Design of FPGAs with Area I/O for Field Programmable MCM

Vijayshri Maheshwari, Joel Darnauer, John Ramirez, Wayne Wei-Ming Dai *

Applied Sciences Building,
Computer Engineering, UC Santa Cruz,
Santa Cruz, CA 95064
Internet: ce.ucsc.edu

Abstract

Area-IO provide a way to eliminate the IO bottleneck of field programmable logic devices (FPLDs) created the mismatch between the ability of perimeter bonds to provide IO and and the propensity of logic to demand it. Whether the incorporation of area IO into FPLD architectures has undesirable side effects is a question that has not yet been answered. In this paper, we examine the architectural impact of area-IO on FPLDs from a theoretical and experimental standpoint and show that the introduction of area IO generally improves *the routability and delay of a set of benchmark circuits.*

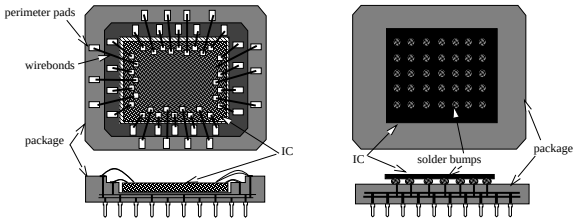
1

1 Introduction

The application of area-IO to Field Programmable Logic Devices (FPLDs) originally stemmed from our work on Field Programmable Multi-Chip Modules (FPMCM).[DGI+94] In this section we introduce area-IO and present reasons why area-IO are desirable even in the context of single-chip FPLDs. We then propose a specific area-IO architecture whose characteristics will be the subject the remainder of the paper.

1.1 Limits of Perimeter Bonding

FPLDs are connected to a digital system using a combination of packaging structures that provide protection, electrical connection, and efficient removal of the by-products of computation.[Tum92] Today, the most common method of making the electrical connection between the FPLD and the package is by wirebonding.[Xil93] Wirebonds are mechanically delicate, and have relatively high electrical parasitics.[Bak90] Because of this, it is usually not recommended to have long wirebonds, limiting the area where wirebonds can be attached to the perimeter of the IC.[Har91] In addition to this, the precision of the machinery requires each bond pad to be placed at a pitch greater than 100 microns in order to ensure high yield.[Har92] Since the area where wirebonds can be attached is limited, and since each bond takes up a certain amount of area, the total number of perimeter bonds that can be placed on a single die is limited. (See figure 1a). Experience suggests this number of IO is not always enough, especially in FPLD-based emulation systems because of the large number of chip to chip signals.[BTA93]



(a) Perimeter-IO (b) Area-IO

Figure 1: Wirebond (a) and flip-chip (b) packaging technologies have an impact on the routability and performance of FPGAs.

1.2 Potential Solutions

There are several ways to increase the number of IO on a single IC:

- Use a larger die with a larger perimeter and more IO.
- Use a very oblong die to increase the perimeter without changing the area.
- Improve bonding technology so that smaller bond pitches are feasible.
- Change bonding processes to allow bonds to be placed away from the perimeter.

Let us consider each of these in turn:

1.2.1 Larger Die

Although increasing the die and leaving a large amount of dead space between the core logic and the pads is a common approach in IO-limited standard cell

*This work supported by ARPA under ONR Grant N00014-93-1-1334.

1

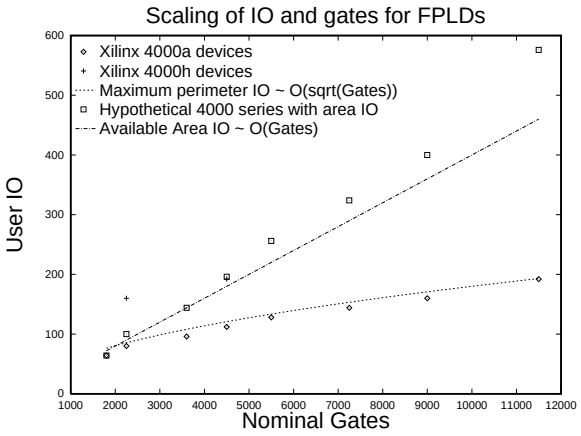


Figure 2: Scaling relationships: Xilinx 4000 series, the $O(N^{1/2})$ growth of perimeter IO, the $O(N)$ growth of area IO, and a hypothetical area-IO 4000 series.

designs[DB93], it is not likely that the same approach will be employed in FPLDs. First, FPLDs are already pushing the limits of what can be done on a single IC. Second, making the FPLD larger results in an exponential increase in chip cost, which is not returned in added functionality because most of the added logic resources, although they have been paid for, will be idle.

1.2.2 Aspect Ratio

It is possible to increase the ratio of perimeter to silicon area as much as desired by merely elongating the die. More rectangular die obviously have a higher ratio of IO to gates. Although this is a useful technique, there are two factors that limit its application. The first is that very oblong die are more difficult to handle and manufacture. The second is that if the die is very oblong, the signal traffic that needs to cross the narrow central axis of the die may begin to exceed the existing wiring resources, creating a bottleneck.

1.2.3 Improve Bond Pitch

Wirebond pitches have been decreasing over time and techniques such as TAB have the potential to increase the number of IO that can connect to a fixed length of die perimeter while simultaneously addressing the problem of yield.[HHYea93] Although there is good evidence that wirebonding technology can be improved, these improvements are likely to be costly to develop and incremental in nature and to break down once again as increasing demands are placed upon them.

1.2.4 Area IO

The crux of the problem is that the demand for IO of a complex of N^2 gates grows faster with respect to N than the perimeter because of Rent's rule. [DGI+94] The central claim of this paper is that area IO provide an immediate, lasting, and un-controversial solution to the IO bottleneck on FPLDs. In area-IO, the IO pads are placed anywhere on the surface of the IC, and the signals are carried to their destinations via wires in a multi-layered interconnect medium (which could be a package, a board, or a module).(Figure 1b). In the case MCM and BGA, the bonding connection is usually achieved by solder bumps between the IC and the interconnect fabric.[Bak90] With area-IO the number of IO pads scales linearly with the chip size, meaning that as chip sizes increase, the IO bottleneck becomes less and less a problem.(Figure 2). [2]

1.3 True Area-IO vs Redistribution

There are several ways to achieve area-IO. One technique is to take a perimeter-IO die and simply place additional layers of interconnect and packaging on it to redistribute the perimeter pads to the center of the die.[DD94, Bin94] This has the advantage that very little re-design is required, but the disadvantage is that the number of IO will still be limited by the number of bond pads at the perimeter. As such, these redistribution techniques are merely a short-term measure.

A more radical technique is to move the pads from the perimeter to the center of the die on the IC itself. Although this requires significant changes to the architectural and physical design of FPLDs, we believe that in the long run such design effort is justified. First, employing area-IO during the chip design process eliminates the need for the costly and performance-limiting redistribution layers discussed above. Secondly, area-IO has some advantages for clock and power distribution.Thirdly, savings in chip area have been reported when switching from perimeter to area IO in physical design of standard cell devices, suggesting that the FPLD layout area will decrease.[DB93]

Although true area-IO technique has some advantages, might not there also be some disadvantages to introducing such radical changes? Perhaps placing the IO in the center of the die will result in routing congestion or very different and difficult placement problems that produce longer routing delays. These are the concerns that the remainder of this paper addresses. The hurried reader can rest assured that placing the IO evenly throughout the FPLD generally makes the routing problem simpler, resulting *less* routing congestion and *smaller* overall delays.

1.4 A Proposed Area-IO Architecture

To make the following discussion more concrete, for the remainder of this paper we will consider two competing FPLD architectures The first architecture is the existing family Xilinx 4000 FPLD with perimeter IO pads.[Xil93] The second is a modified version where the perimeter IO blocks have been removed and each CLB has been supplemented with an IO pad. (Figure 3) [3] Assuming that the new architecture has an equivalent gate count, the gates-IO relation for the area-IO architecture is shown in figure 2.

2 Theoretical Analysis

In the previous section, we described how area-IO could be used to circumvent the perimeter IO bottleneck. In this section, we discuss the possible impact

[2] Although with a planar interconnect fabric the number of IO leaving the die is still bounded by a function of the die perimeter, the fact that thin film wires can be spaced much closer than wirebonds and the fact that layers in the interconnect fabric can be stacked make the problem much less severe.

[3] Some readers may question whether this is feasible, and though we do not have access to die information from Xilinx to make a determination, area-IO can be placed as close as 200μm to effect this change.

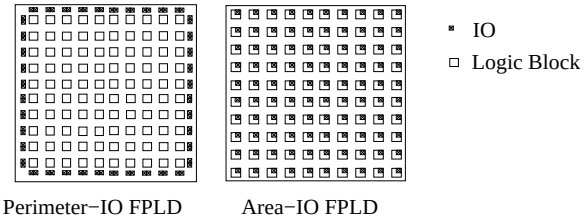


Figure 3: Difference between area-IO and perimeter-IO FPLDs

that an area-IO will have on routability and delay. Our goal is to compare the routability and delay of "typical" circuits on two sets of architectures: architectures with perimeter IO and architectures with area-IO. We will conclude that area-IO are a good feature if we can show that the probability that a circuit will route on the area-IO FPLD is higher than on the standard architecture, and if we can show that the critical delay on the area-IO architecture is likely to be smaller.

2.1 Observables

To keep our intellectual feet on the ground, we should consider what properties we can actually measure and what properties yield to theoretical analysis. Let us assume that we have two architectures to compare and a set of benchmarks circuits which capture the meaning of "typical circuit".

In order to make a comparison of delay, we could route each circuit onto both architectures using some algorithm which is believed to be well-suited to the architecture in question. (This algorithm need not be the same for both architectures). We could then measure the critical path delays and come to a quantitative conclusion about the impact of area-IO on delay on the benchmark circuits. This observation can be generalized to a claim about the delay of typical circuits to the degree that the benchmarks represent typical circuits.

Making quantitative conclusions about the *routability* of an architecture is a much more difficult matter, since routability is a discrete event. For example, given ten circuits and a pair of architectures, we might find that all of the circuits routes on each of the architectures in which case we cannot make any distinction. We might compare routability by using a set of architectures where some routing resource's availability (such as channel width) is parameterized, and asking what the cut-off point for the parameter is. Although this is a tractable approach which has been employed in the past, it requires a large number of experiments and customized routing tools. For the purposes of this paper, it would be useful to find another quantity which can be used as an indicator of routability.

2.2 Wirelength as the Figure of Merit

A connection between the total length of placed wire, routability, and circuit delay has been established.[BRV93, Don79] In general, longer overall wirelength is related to longer average signal delay because longer wires pass through more switches and have a higher capacitive loading. Higher total wire lengths are also related to lower routability, because the chance of contention for each element of the fixed number routing resources is raised. In addition, there is a bulk of theoretical work on wirelength distributions which have been used to obtain routability results and channel capacity estimates.[Gam81, Don79, Don81, Feu82, CC91] Because wirelength is directly observable and a relatively strong theoretical background exists, we will focus on the wirelength distribution in our theoretical comparisons.

Note that it is not the absolute wirelength (in meters) that matters as much as the number of rows and columns in the routing lattice which must be traversed. Thus the small enlargement of the logic block in the area-IO architecture may be ignored for the routability and delay analysis.

2.3 Algorithm 1: Random Placement

Since the wirelength distribution depends on the algorithm as well as the circuit and the architecture, it will be useful at first to consider a very simple placement algorithm: *random placement*. Although we are not suggesting that random placement is a good method for choosing IO or block locations, in the case that the IO positions are arbitrarily pre-assigned, the random model of the placement may be the most appropriate.

Let us assume that our circuit is a collection of N^2 logic blocks with an average of k terminals on each block. The logic blocks are then placed arbitrarily in an $N \times N$ lattice in the core of the FPLD.(Figure 4) If Rent's rule holds, then the number of IO required by this collection of blocks, denoted by P, will be approximately:

$$P \approx kN^{2r} \tag{1}$$

where r is the "Rent exponent", a design dependent parameter between 0 and 1 with a "typical" value of 2/3.[Don81]

To find the number of terminals requiring connection, T, we add the number of external pins and the number of terminals on each of the N^2 blocks:

$$T \approx k(N^2 + N^{2r}) \tag{2}$$

Assuming each net connects $d \geq 2$ terminals, the number of total connections or nets, c, is:

$$c \approx T/d \approx k(N^2 + N^{2r})/d \tag{3}$$

We may partition the nets into two types as in figure 4, based on whether they connect an IO pad or not. Assuming each IO pad appears is used for one net, the number of *external* nets, i.e. those between logic blocks and pads on the perimeter, is:

$$c_{ext} = P \approx kN^{2r} \tag{4}$$

while the remaining number of *internal* nets is just:

$$c_{int} \approx (k(N^2 + N^{2r})/d - kN^{2r} \tag{5}$$

Let us assume that the routing algorithm is optimal in the sense that it successfully connects blocks using some path that has the same length as the Manhattan distance between blocks, or in the case of a multi-terminal net, a path that has the same length as the half of the perimeter of the bounding box enclosing all of the terminals to be connected. Then, the total length of wire generated by random placement is denoted by L and can be decomposed into the sum:

$$L_{rand} = c_{int}\overline{l_{int,rand}} + c_{ext}\overline{l_{ext,rand}} \tag{6}$$

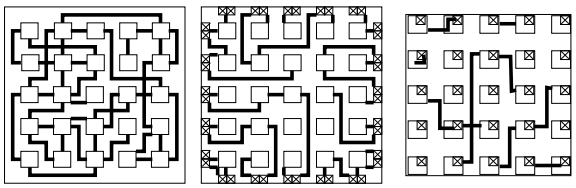


Figure 4: Internal and external nets

where $\overline{l_{int,rand}}$ and $\overline{l_{ext,rand}}$ are the expected lengths of the internal and external nets. Since c_{int} and c_{ext} are already well-understood, to estimate L we need only to evaluate $\overline{l_{int,rand}}$ and $\overline{l_{ext,rand}}$.

Since the choice of architecture does not affect the randomness of the placement of the logic blocks in the core, we can compute $\overline{l_{int,rand}}$ independent of the architecture. Ignoring the discrete nature of the placement, the expected value of the internal net length depends on the number of terminals and is proportional to the array size. Assuming for simplicity that the number of terminals on each net is simply 2, we have:

$$\overline{l_{int,rand}} \approx 2N/3 \tag{7}$$

Note that the internal lengths should be the same regardless of the architecture.

For the external nets, the lengths depend on whether the IO are located on the perimeter or in the core. In the area-IO case, the IO are located in the core. If we adopt the convention that the IOs in the area-IO case are distributed evenly inside the logic blocks in the core, then the lengths are drawn from a random process equivalent to L_{int}:

$$\overline{l_{ext,aio}} = \overline{l_{int}} \tag{8}$$

This similarity exists even if there are fewer IOs than logic blocks, as long as the IOs are placed evenly within the logic blocks.

In the perimeter IO case, however, although the endpoint in the core is drawn from the previous uniform distribution, the endpoint on the IO pad must be distributed evenly along the edge of the N by N shape. We can show that the expected value of the length of such a two-point net is:

$$\overline{l_{ext,rand,pio}} \approx 5N/6. \tag{9}$$

We can see immediately that the average external 2-pin net in the perimeter IO case is slightly (25%) longer than the average internal net. The area IO architecture does not suffer from this disadvantage as its internal and external nets are exactly the same length. How seriously this effect impacts the total chip wiring will depend on the number of internal and external nets, and their respective degrees.

If we again assume that almost all nets are of degree 2 and combine this with (6) we get:

$$L_{rand,pio} \approx (kN)(N^2/3 + N^{2r}/2) \tag{10}$$

for the perimeter case, and

$$L_{rand,aio} \approx kN(N^2 + N^{2r})/3 \tag{11}$$

for the area-IO case.

To consider how these quantities scale with array size, we can compute the fraction of the perimeter IO's wire that we could expect to eliminate by going to area-IO:

$$\delta_{rand} = (L_{rand,pio} - L_{rand,aio})/L_{rand,pio} \tag{12}$$

$$\approx 1/(2N^{2-2r} + 3) \tag{13}$$

This measure of area-IO's advantage decreases with increasing N because the external nets become a smaller fraction of the total interconnect. This analysis suggests that area-IO offer a small improvement on the total wire length, but not much. In the limit as FPGA become very large, the expected improvement in average wirelength under random placement becomes negligible.

2.4 Limitations

Before too much confidence is placed in these models, it should be noted that many of the assumptions that they are based on are very rough. For example, most of the previous theoretical models assume that the routing area is so large that edge effects are insignificant, and gloss over the possibility of hot-spots in the routing architecture. Secondly, real placement programs almost always beat random placements. Finally, we have glossed over the new resource conflicts that placing the IO structure in the logic blocks might create. Although the theoretical models suggest that area-IO improves the wirelength, and by implication the delay and routability, experimental results are clearly needed in order to make a firm determination. It should be stressed that the proper purpose of the above models is to justify the effort involved in the experiments, rather than to produce detailed predictions of wiring savings.

3 Experiment

In the previous sections we introduced the need for area-IO and provided a theoretical model that suggests that area-IO may have advantages over perimeter-IO in routability and delay. In this section we provide experimental evidence to substantiate this claim.

We performed two sets of experiments to see the effect of area I/O on total routing resources, and critical path delay. These experiments were later refined to better understand the mechanism of the wirelength savings.

3.1 Common Considerations

There are a number of procedural factors common to both set of experiments. These experiments were performed on MCNC benchmarks shown in table 1. We used the XACT 5.0.0 software to place and route the designs in order to avoid uncertainties and non-repeatability associated with home-grown software. To simulate the perimeter-IO architecture we used the raw circuit on the existing X4000 series part. In order to simulate the presence of area-IO for the comparison, the *circuits* were modified to fool the software into believing that IO pads can be placed inside a CLB.

The basic concept is that we can pretend that we have replaced one of the flip flops in each CLB with a more complicated circuit which contains both a flip flop and an IO pad without changing the routing architecture. (Figure 5) Then, to simulate the area-IO architecture, we can then modify the netlist as follows. First, each input block or latched input block is replaced by the Q (output) pin of a D flip flop which can be gated to the

Circuit	CLBs	IOs	Part type	Gate Util.
alu2	108	16	4005PQ160	55%
vda	167	56	4005PQ160	85%
x1	79	86	4005PQ160	40%
c1355	83	73	4003PQ100	83%
c3540	256	72	4006PG156	100%
c499	56	73	4003PQ100	56%
c880	85	86	4003HPG191	85%
s1196	144	30	4004APC84	100%
s1423	100	24	4003PC84	100%
s5378	256	86	4006PG156	100%
s9234	400	43	4010DPC84	100%
s13207	576	154	4013MQ208	100%

Table 1: MCNC benchmark characteristics. Circuits in the lower half of the chart are sequential circuits. Part shown is the part used to implement the design. In all cases the smallest feasible part was chosen.

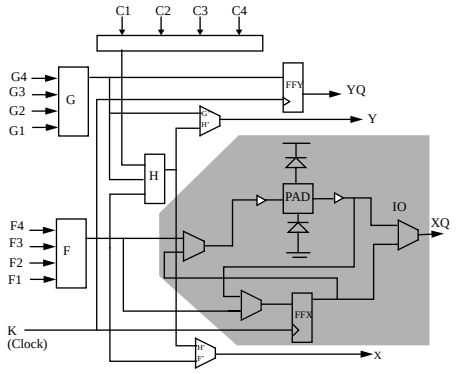


Figure 5: A modified X4000 CLB with IO structures. The grey region consists of added circuitry that replaces the ordinary D-flip flop in the existing Xilinx 4000. The circuit can implement direct and latched output of the function table, direct and latched input from the pad, or the original function of the CLB.

pad in our new CLB. Similarly the pin for each latched or un-latched output block is replaced by the D (input) pin of a D flip flop which can be gated to an output buffer in the new CLB. The CLK pin of the flip-flops created for un-latched inputs was left unconnected so as not to produce additional routing. Finally, constraints are put on the added flip flops to ensure placement of at most one of these added flip flop per CLB.

The modified CLB we used to accomplish this deception will perform rather well, though it should be noted that the design of this CLB was guided by the expediency with which it could be simulated with existing tools, not for its elegance.

3.2 Wiring Resource Comparison

The first experiment was designed to measure the impact of area-IO on the routability of FPGAs. Since routability is difficult to measure directly for the reasons discussed before, we decided to compare the overall wiring resources used in each architecture for each circuit. Our hypothesis was that the area-IO architecture would require fewer overall routing resources for most circuits.

In the perimeter-IO case the circuits were placed and routed normally and the utilization of routing resources was reported by the Xilinx tools. The area-IO case was implemented using the circuitous technique described above. The results after 100% routing completion are reported in Table 3.

Although XACT reports usage of different type of routing resources used, to simplify the comparison, we report a a weighted sum corresponding roughly to the total wire length. The formula used for calculating the cost is:

$$Total = n_{pips} + 5n_{feed} + 5n_{local} + 10n_{dbl} + 15n_{long} + 15n_{global} \quad (14)$$

The last column reports the percentage improvement (or decline) when switching from perimeter to area-IO. We can see there is overall improvement (13%), but that the improvement is greatest for the combinational circuits. This suggests that although area-IO has advantages, our tentative CLB architecture and modified netlist interferes too much with the normal placement of latches.

3.3 Delay Comparison

The method in this set of experiments were the mostly same as in the previous experiments. Since the Xilinx tools do not measure the delay except in reference to a pad or a clocked latch, clock lines had to be added to the IO-flip-flops for the area-IO pads that were not originally latched. Because of this extra routing, the designs were re-placed and routed, and the delays measured with *Xdelay*. The worst case delay for each design was appropriately corrected to account for differences in the PAD and the DFF delays that were introduced artificially by our modifications. For example if t_{cs} is the clock-to-setup delay reported by *Xdelay* for area-IO case, from the replaced DFF to replaced DFF, the corrected pad-to-pad t_{pp} delay is expressed as:

$$t_{pp} = t_{cs} - (t_{clk2q} + t_{setup}) + t_{ibuf} + t_{obuf} \quad (15)$$

Table 2 shows the corrected critical path delay for each design on each architecture.

The data show significant improvement for many designs, and an average improvement of about 23.5%. Most of this improvement is due to the combinational circuits – the sequential circuits show mixed results, with a mean improvement of only 1.4%.

Each experiment showed some overall savings, but most of savings was for the combinational benchmarks. This suggests that while area-IO has substantial benefits for layout in general, the particular architecture we employed for these experiments results in an unhealthy amount of competition between the latches and the IO structures, which greatly reduces the advantages.

3.4 Number of IO vs. Position of IO

The savings detected in the previous results can be due to two separate mechanisms. The first mechanism is that the location of the IO within the area-IO architecture leads to shorter wirelengths because the mean distance from IOB to CLB is decrease. A second and equally plausible mechanism is that the sheer number of IO in the area-IO architecture greatly lowers IO utilization, increasing the chances that a nearby IO will be unused and resulting in shorter wirelengths. We expect that some fraction of the improvement above is

Design	$Delay_{pio}$(ns)	$Delay_{aio}$(ns)	Change
alu2	175.0	121.7	30%
vda	110.5	74.3	32%
x1	60.2	43.9	27%
c1355	102.3	81.5	20%
c3540	191.0	157.9	17%
c499	75.0	42.1	44%
c880	127.6	105.8	17%
s1196	108.4	95.8	11%
s1423	266.5	272.3	-2%
s5378	89.7	87.0	3%
s9234	122.0	130.9	-7%
s13207	166.2	166.3	0%
$Mean_{comb}$			27%
$Mean_{seq}$			1%
$Mean_{all}$			16%

Table 2: Comparison of critical path delay in perimeter (pio) and area (aio) architectures for 12 benchmark circuits adjusted from values reported by Xdelay. The "change" column is the quantity ($Delay_{pio}$-$Delay_{aio}$)/$Delay_{pio}$.

attributable to each of these causes, though it is reasonable to argue that the whole improvement is due only to the larger number of IO in the area-IO case. The distinction is important because real area-IO designs have a large number of IO and cannot benefit from the first mechanism.

In order to show that the savings in resources is also due to the difference in IO placement, we ran the few of the same designs for area-IO on Xilinx FPGA with additional constraints so that the number of CLB'S available for area-IO is almost equal to the one available in the perimeter-IO, but that the IOs are distributed evenly throughout out the chip. The preliminary results indicate that some of the improvement in routing resources usage in area-IO is due to the distribution of the pads.

4 Conclusions

4.1 Costs of Area IO

We see that a shift to area-IO has some benefits. But what about the costs? There are two types of costs that area-IO incurs.

First, since area-IO is solely motivated by the need to increase the number of IO, we would expect that the number of IO would dramatically increase.[4] Although the individual IO structures for flip chip are smaller than wirebond pads, doubling the number of IO would substantially increase the total chip area, and thus the cost of the chip. This "cost" can be small, however, when one considers that the number of high-IO chips needed to implement a complex system may be much smaller than the number of traditional chips needed.

The second costs are more dramatic. Since the number of applications that demand the high-IO capacity of flip-chip is a relatively small fraction of the FPGA market, the engineering costs need to be borne by a small number of users. These engineering costs include redesign of the FPGA core, which requires the development of a new layout style that does not depend on traditional ring-based power distribution. In addition, investment in new manufacturing and testing procedures will be needed. Given all of these development and infrastructure costs, it does not seem likely that vendors will be quick to implement area-IO FPGAs.

How might this situation change as technologies evolve? First, as the MCM and flip-chip infrastructure becomes more mature, the cost of the assembly and test process will eventually surpass wirebonding. In addition, developers of high performance chips that demand area-IO will find design styles that replace the perimeter-based techniques. At some point, the costs of area-IO will gradually decrease. On the demand side, we can expect as linewidths continue their inexorable shrinkage that both perimeter based technologies will need to push hard to keep up with the IO demand on denser and denser chips. Area-based technologies will have to push as well, but not as soon and not nearly as hard since they are starting in an advantaged position. Some have argued that as FPLDs become more and more dense that the demand for IO on large FPLDs may level off since whole systems become integrated. The demand for IO *density* is relatively independent of whether or not the IO demand for large FPLDs increases. Today's large FPLDs will become tomorrows medium sized and eventually small FPLDs, but will require the same number of IO as are needed today. They will simply have to deliver the same number of IO in a smaller area.

4.2 Summary

Area-IO provide a promising way of increasing the number of IO on a chip beyond the perimeter-IO limits. We have shown a theoretical argument that suggests that area-IO will reduce the total wirelength and thus improve the delay and routability of FPLDs. Experimental evidence shows that this is generally true for a particular area-IO architecture we presented, though the architecture we have presented suffers from resource conflicts when applied to sequential benchmarks. The improvements measured exist even in the case the the area-IO architecture is restricted to the same number of IO as its perimeter counterpart. Although, more work is needed to produce a single accurate theoretical model, and to produce an area-IO architecture that performs as well for combinational and sequential circuits alike, area-IO FPLDs clearly have the advantages of more IO per gate, higher routability, and decreased delay. A pressing question that remains to be answered is whether these gains are compelling enough today to motivate the changes required to achieve them.

Acknowledgments

The authors are indebted to Aptix Corporation, and AT&T Bell Laboratories, Quickturn Design Systems, and especially Dr. Kamal Choudhary and others at Xilinx for providing material and technical support for the FPMCM project. We also wish to thank the reviewers for their many suggestions.

References

[Bak90] H.B. Bakoglu. *Circuits, interconnections, and packaging for VLSI.* Addison-Wesley, 1990.

[Bin94] Ashock Bindra. Moto gets 'slicc' with packaging; will detail bga-flip chip combination at nep-

[4] The performance increases can be achieved by running dedicated lines from the periphery into the core of the chip.

Designs	Pips		Feed		Locals		Doubles		Longs		Globals		Total		%
	Pio	Aio	Pio	Aio	Pio	Aio	Pio	Aio	Pio	Aio	Pio	Aio	Pio	Aio	Change
alu2	1273	1260	3	4	457	451	230	228	97	102	0	0	7328	7345	-0.2
vda	2032	1832	5	11	776	678	371	350	215	166	0	0	12872	11267	12
x1	1081	956	0	4	361	348	209	177	118	62	0	0	6746	5416	20
c1355	871	734	1	0	279	234	191	146	60	46	0	0	5081	4054	20
c3540	3711	3559	24	34	1341	1305	586	518	305	280	0	0	20971	19634	6
c499	742	592	2	2	222	195	176	121	71	36	0	0	4687	3327	29
c880	1190	945	2	0	370	348	212	160	106	55	0	0	6760	5110	24
s1196	1773	1671	6	5	549	503	360	326	154	142	12	11	10638	9766	8
s1423	1344	1407	1	5	400	455	189	205	76	62	10	10	6529	6837	-5
s5378	5551	5283	75	52	2230	2122	832	733	396	366	21	16	31651	29213	8
s9234	6588	6386	76	50	2490	2278	927	989	429	395	21	24	35438	34201	3
s13207	14124	14290	350	373	5956	6228	2083	2111	608	616	38	36	76174	78185	-3
Mean_{comb}															16
Mean_{seq}															2
Mean_{all}															10

Table 3: Comparison of wiring resources used for a collection of MCNC benchmarks on perimeter ("pio") and area ("aio") architectures. The "Total" column is a weighted sum of the component routing resources used that approximates the total wirelength. The "Change" column is the ratio $(Pio - Aio)/Pio$.

con. *Electronic Engineering Times*, 786:1, Feb 28 1994.

[BRV93] Stephen D. Brown, Jonathan Rose, and Zvonko G. Vranesic. A stochastic model to predict the routability of field programmable gate arrays. *IEEE Transactions on Computer Aided Design of Integrated Circuits and Systems*, 12(12):1827–1838, December 1993.

[BTA93] J. Babb, R. Tessier, and A. Agrawal. Virtual wires: overcoming pin limitations in fpga-based logic emulators. In *Proceedings IEEE Workshop on FPGAs for Custom Computing Machines*, pages 142–151, 1993.

[CC91] Jeffrey Cotter and Phillip Christie. The analytic form of the length distribution function for computer logic interconnections. *IEEE Transactions in Circuits and Systems*, 38(3):317–320, March 1991.

[DB93] P. Dehkordi and D.W Bouldin. Design for packageability-the impact of bonding technology on the size and layout of vlsi dies. In *IEEE Multi-Chip Module Conference*, pages 153–9, 1993.

[DD94] Joel Darnauer and Wayne Dai. Fast pad redistribution from periphery io to array io. In *IEEE Multi-Chip Module Conference MCMC-94*, pages 38–43, 1994.

[DGI+94] Joel Darnauer, Porfirio Garay, Tsuyoshi Isshiki, John Ramirez, and Wayne Dai. A field programmable multichip module (fpmcm). In *IEEE Workshop on FPGAs for Custom Computing Machines*, pages 1–10, 1994.

[Don79] W. E. Donath. Placement and average interconnection lengths of computer logic. *IEEE Transactions on Circuits and Systems*, CAS-26(4), April 1979.

[Don81] W. E. Donath. Wire length distribution for placements of computer logic. *IBM Journal of Research and Development*, 25(3):152–155, May 1981.

[Feu82] Michael Feuer. Connectivity of random logic. *IEEE Transactions on Computers*, C-31(1):29–33, January 1982.

[Gam81] Abbas A. El Gamal. Two dimensional stochastic model for interconnections in master slice integrated circuits. *IEEE Transactions on Circuits and Systems*, CAS-28(2):127–138, February 1981.

[Har91] Charles A. Harper, editor. *Electronic packaging and interconnection handbook*. McGraw Hill, c1991.

[Har92] George Harman. Wire bonding - towards 6-σ yield and fine pitch. *IEEE Transaction on Components, Hybrids, and Manufacturing, Technology*, 15(6):1005–23, December 1992.

[HHYea93] Y. Hiruta, N. Hirano, Y. Yamaji, and M. Mukai et. al. An 820 pin pga for ultralarge-scale bicmos devices. *IEEE Transactions on Components, Hybrids, and Manufacturing Technology*, 16(8):893–901, Dec. 1993.

[Tum92] Roy Tummala. Multichip packaging-a tutorial. *Proceedings of the IEEE*, 80(12):1924–41, December 1992.

[Xil93] Xilinx. *The Programmable Logic Data Book*. 2100 Logic Drive, San Jose, CA 95124, 1993.