Performance Benefits of Monolithically Stacked 3D-FPGA

Mingjie Lin, Abbas El Gamal, Yi-Chang Lu, and Simon Wong
Department of Electrical Engineering
Stanford University, CA 94305
{ mingjie, abbas, yizanglu, wong }@stanford.edu

ABSTRACT

The performance benefits of a monolithically stacked 3D-FPGA, whereby the programming overhead of an FPGA is stacked on top of a standard CMOS layer containing the logic blocks and interconnects, are investigated. A Virtex-II style 2D-FPGA fabric is used as a baseline for quantifying the relative improvements in logic density, delay, and power consumption achieved by such a 3D-FPGA. It is assumed that only the pass-transistor switches and configuration memory cells can be moved to the top layers and that the 3D-FPGA employs the same logic block and programmable interconnect architecture as the baseline 2D-FPGA. Assuming a configuration memory cell that is ≤ 0.7 the area of an SRAM cell and pass-transistor switches having the same characteristics as nMOS devices in the CMOS layer are used, it is shown that a monolithically stacked 3D-FPGA can achieve 3.2 times higher logic density, 1.7 times lower critical path delay, and 1.7 times lower total dynamic power consumption than the baseline 2D-FPGA fabricated in the same 65nm technology node.

Categories and Subject Descriptors

B.7.1 [**Integrated Circuits**]: [Types and Design Styles]

General Terms

Design, Experimentation, Measurement, Performance

Keywords

FPGA, 3D monolithically stacked, performance analysis.

1. INTRODUCTION

Cell-based design technology has dominated ASIC implementation over the past 20 years by offering an economically compelling combination of low manufacturing cost and acceptable design and prototyping costs. With the advent of sub-100nm CMOS technologies, the design and prototyping costs of cell-based implementation have become prohibitive for most ASICs, making FPGAs increasingly popular. Current FPGAs, however, cannot meet the performance requirements of many ASICs due to their high programming overhead. As discussed in [1], as much as 90% of the FPGA area is occupied by programmable routing resources. In addition to consuming most of the die area, programmable routing also contributes significantly to total path delay in FPGAs. In [2], interconnect delays are estimated for the Altera's 8K series and the MIT DPGA and found to account for roughly 80% of total path delay. Programmable routing also contributes to the high power consumption of FPGAs, a problem that has recently become a significant impediment to their adoption in many applications. Power consumption measurements of the Xilinx XC4003A and Virtex-II FPGAs [3, 4, 5] have shown that programmable routing contributes over 60% of the total dynamic power consumption. As a result of these performance degradations, FPGA performance is significantly worse in terms of logic density, delay, and power than cell-based implementations. Studies [1, 2] have estimated FPGAs to be over 10 times less efficient in logic density, 3 times larger in delay, and 3 times higher in total power consumption than cell-based implementations.

Although CMOS technology scaling has greatly improved the overall performance of FPGAs, the performance gap between them and ASICs has remained very wide mainly because the FPGA programming overhead shares the same layers as the logic and interconnect. In [6] it is argued that the performance gap between FPGAs and cell-based is becoming even greater in sub-100nm technologies. While the rate of increase in FPGA logic density has tracked that of cell-based, the system frequency has scaled at a lower pace and power consumption has risen to unacceptable levels.

Monolithically Stacked 3D-FPGA

A conceptually appealing approach to closing the performance gap between FPGAs and cell-based ASICs is to stack the programming overhead of an FPGA on top of the logic blocks and interconnect layers that would be implemented in a state-of-the-art CMOS technology (see Figure 1). Besides the obvious benefit of higher logic density, vertical stacking reduces interconnect length, hence reducing signal path delay and power consumption. However, implementing such an approach requires the top layers to have comparable vertical interconnect density to that of the CMOS technology used to implement the logic blocks and interconnects. Several approaches to chip and wafer stacked 3D integrated cir-

This work was partially supported by DARPA and SPAWAR System Center (Grant number N66001-04-1-8916).

FPGA'06, February 22–24, 2006, Monterey, California, USA.

cuits (3D-IC) have been recently developed [7, 8]. The vertical via densities achieved by these technologies, however, are over an order of magnitude lower than in a state-of-the-art CMOS technology, and they are not expected to scale much.

A more promising 3D-IC approach for implementing such a 3D-FPGA is monolithic stacking, whereby active devices are lithographically built in between metal layers. The main advantage of such approach is that, in principle, it can achieve comparable vertical via density and scale at the same rate as the base CMOS technology. Although this approach has yet to be developed for the FPGA application, there is much evidence that forming transistors on a dielectric with low thermal budget is quite feasible [9]. The process technology for the added layers can be much simpler than a full CMOS process. Specifically, the switch layer only needs one type of MOS transistors, while the memory layer can be implemented using a 2-T flash technology [10] or a programmable solid-electrolyte switch [11], both of which promise to achieve higher densities than SRAM with much simpler processes.

Note that our proposed approach to 3D-FPGA is significantly more constrained than the approach investigated in [12], where 3-D switch boxes that require much more complex 3D technology to implement are proposed and evaluated.

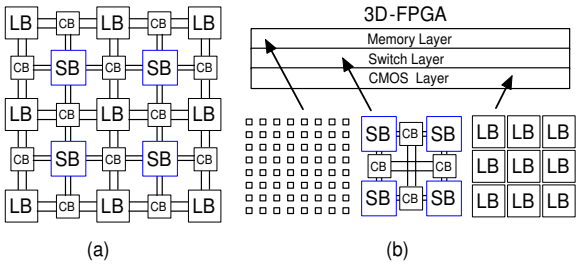


Figure 1: (a) 2D-FPGA (LB: logic block, CB: connection box, SB: switch box). (b) 3D Monolithically stacked 3D-FPGA.

Under a 3D-IC research program, an interdisciplinary team of researchers at Stanford University and several other institutions has been developing the monolithically stacked technologies needed to implement a 3D-FPGA as well as the architecture and circuit designs of such an FPGA. In this paper, we describe the results of a study we have conducted under this program to quantify the potential improvements in logic density, delay, and power of a monolithically stacked 3D-FPGA over conventional 2D-FPGAs. To perform the comparison, we assume a Virtex-II style 2D-FPGA architecture as a baseline. It is assumed that only the pass-transistor switches and configuration memory cells can be moved to the top layers (see Figure 1) and that a Virtex-II style logic block and switch box designs are used. A technology independent FPGA area model is developed and used to compare the logic density of a stacked FPGA to the baseline FPGA as a function of configuration memory element size. RC circuit models for interconnect segments are developed and used to estimate the improvements in interconnect delay in the 3D-FPGA relative to the baseline FPGA for four deep submicron CMOS technology nodes. The interconnect delay results are then used to estimate the relative improvements in the geometric average net delays and critical path delays achieved by the 3D-FPGA for 20 MCNC benchmark circuits placed and routed using VPR [13]. Finally a model for dynamic power consumption is developed and used to quantify the relative improvement in power consumption.

Summary of Results and Outline of the Paper

Assuming a configuration memory cell that is ≤ 0.7 the area of an SRAM cell, e.g., a 2-T flash or programmable solid-electrolyte switch [10, 11], and pass-transistor switches having the same characteristics as nMOS devices in the CMOS layer are used, we show that a monolithically stacked 3D-FPGA can achieve 3.2 times higher logic density, 1.7 times lower critical path delay, and 1.7 times lower dynamic power consumption than the baseline 2D-FPGA implemented in the same 65nm technology node. Since, the logic density improvement can be achieved with the addition of only a few mask layers on top of a standard CMOS technology, a monolithically stacked FPGA is expected to have a lower manufacturing cost than an FPGA with the same logic capacity fabricated using only the standard CMOS technology. It is also expected that additional performance improvements can be achieved by re-architecting the 3D-FPGA to take full advantage of the added layers.

The next section presents the baseline 2D-FPGA architecture, the FPGA area model we use, and the logic density improvements achieved using a 3D-FPGA. In Section 3, we describe the analytical interconnect model and the methodology we use to estimate delay. The model is then used to quantify the delay reduction achieved using a 3D-FPGA for several submicron CMOS technology nodes. In Section 4, we quantify the reduction in dynamic power consumption achieved using a 3D-FPGA.

2. 3D-FPGA LOGIC DENSITY

We choose a Virtex-II island-style FPGA logic fabric as a baseline architecture for our study (see [14] for more details on the Virtex-II architecture). The fabric consists of a 2D array of logic blocks (LBs) that can be interconnected via programmable routing. Each LB contains four slices, each consisting of two 4-input Lookup Tables (LUTs), two flip-flops (FFs), and programming overhead. A *segmented* programmable routing architecture is used to minimize the number of transistors and wires that a signal needs to traverse to reach its destination. Specifically, the programmable routing comprises different length horizontal and vertical *interconnect segments* that can be connected to the LBs via *connection boxes* and to each other via *switch boxes*. For the purpose of our study we assume that the FPGA consists of square tiles of width L as shown in Figure 2. As in the Virtex-II, we assume sets of one (referred to as Single), two (Double), three (HEX-3), and six (HEX-6) FPGA tile width segments in addition to interconnects that span the entire array width (Global). The longer segments (HEX-3, HEX-6, and Global) also include pass-transistor switches and buffers that will be included in delay and power consumption estimation. Each connection box comprises pass transistor switches to connect the LBs' inputs and outputs to two neighboring switch boxes through various interconnects. We assume the MUX-based switch box design described in [15] (see Figure 2(b)). In addition to logic and programmable interconnect, our study will consider the global resources and switches used for clocking.

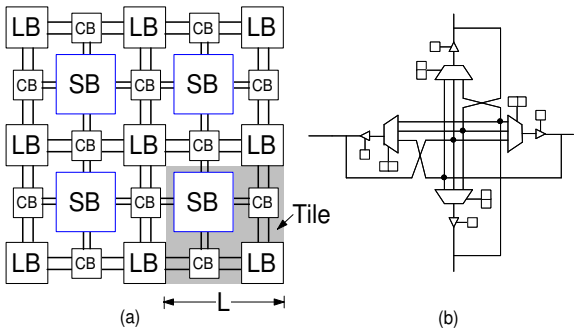


Figure 2: (a) FPGA architecture. (b) Schematic of a switch point (SP).

In the next subsection, we introduce the technology independent model we use to estimate the area breakdown of the baseline FPGA.

2.1 FPGA Area Model

To estimate an FPGA layout area, one needs to estimate the area of a tile, i.e., a logic block and associated programmable routing resources. In previous studies [16, 17, 18, 19], the layout area of the logic block is estimated by counting the number of equivalent minimum-width transistors required for its implementation and multiplying it by the layout area occupied by a minimum-width transistor taking into consideration contact area and spacing to adjacent transistors. The area is computed in terms of λ^2, where λ is half of the minimum feature size of the technology. The advantage of this approach is its simplicity and general applicability. The layout area, however, tends to be overestimated for two reasons. First, individual transistor areas are added up. This overestimates the area when multi-finger transistor layouts are used. Second, the area of a non-minimum width transistor is estimated by multiplying its sizing ratio by the area of a minimum-width transistor. This is clearly an overestimate, since increasing the width of a transistor only increases the device size and the spacing along its width. The area for the programmable routing per logic block is estimated as a function of the track pitch and the dimensions of the logic block. With the availability of eight or more layers of metal, transistor area must also be considered.

In this study, we estimate the logic block area by first decomposing it into smaller components similar in granularity to standard-cell library elements, such as inverters, buffers, 2-MUXs, etc.. A stick diagram for each component and the Magic-8 rules are then used to estimate its area in λ^2. To estimate the total circuit area we add up the areas of its components. We found that this approach yields estimates that are within $\pm 10\%$ of actual layouts. Our approach to estimating the programmable routing area is also different from those in previous studies [16, 17, 18, 19]. To obtain an accurate estimate of the programmable routing area, we treat the routing resources, including the switch boxes, connection boxes, and buffers as logic resources and estimate their area in the same way as for the logic block.

The area breakdown of the various components of the baseline 2D-FPGA architecture estimated using our area model is illustrated in Figure 3. Note that the configuration memory occupies roughly half the area in both the logic blocks and the routing resources. The logic blocks occupy only 22% of the total area (or 14% excluding configuration memory), which matches with previous studies (e.g., see [1]).

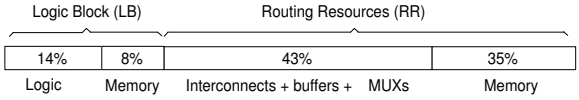


Figure 3: Area breakdown of the baseline 2D-FPGA.

2.2 Logic Density Improvement

In this section, we quantify the potential logic density improvement using a monolithically stacked 3D-FPGA over the baseline architecture described in the previous section. We assume that the 3D-FPGA employs the same logic block and general routing architecture as the baseline FPGA. We also assume that only the pass-transistor switches and the configuration memory may be stacked on top of a standard CMOS technology (see Figure 1). We denote the bottom layer as the *CMOS layer*, the second layer as the *switch layer*, and the top layer as the *memory layer*. Note that any element of the FPGA can be implemented in the CMOS layer. However, only pass-transistor switches can be implemented in the switch layer and configuration memory can be implemented in the top memory layer. Since the CMOS layer is by far the most costly in terms of area, it is important that it be fully utilized in any 3D implementation. Our goal here is to distribute the FPGA resources among the three layers under these constraints to maximize logic density.

Since configuration memory occupies a very large fraction of the FPGA area, the logic density achieved in any implementation depends heavily on the size of the memory cell used. To illustrate this point, consider the three 3D-FPGA scenarios, whose layer area breakdowns relative to the area of the baseline 2D-FPGA, denoted by A, are given in Figure 4. Scenario (a) assumes that an SRAM memory cell is used and that all switches and configuration memory cells are moved to the top layers. In this case, the overall 3D-FPGA area is reduced to $0.43A$. Note, however, that the area in this case is limited by the area of the memory layer and that the CMOS layer is only 72% utilized. The area can be reduced by moving some of the switches and their corresponding SRAM cells back to the CMOS layer. Figure 4(b) shows the results for this scenario. Note that by redistributing the resources, the 3D-FPGA area is reduced to $0.38A$. The area can be further reduced by using a smaller memory cell, e.g., [10]. Scenario (c) shows using a memory cell that is 0.7 the size of an SRAM cell, the 3D-FPGA area can be reduced to $0.31A$, which corresponds to 3.23X increase in logic density.

The above examples motivate us to quantify the logic density improvement of a 3D-FPGA in terms of the configuration memory cell area. We define the parameter $0 < \eta \leq 1$ to be the memory cell size normalized with respect to the size of an SRAM cell. Figure 5 plots the logic density improvement of a monolithically stacked 3D-FPGA over the baseline 2D-FPGA as a function of η assuming that the CMOS layer is fully utilized. Note that using a standard SRAM cell as configuration memory, 3D monolithically stacking can im-

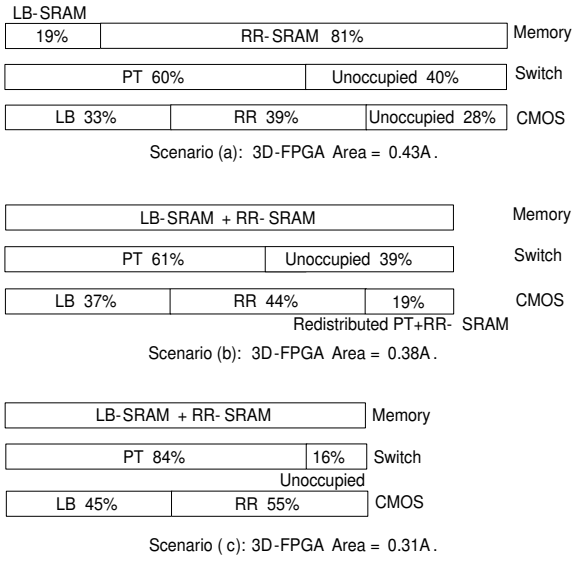


Figure 4: Area breakdown for three FPGA stacking scenarios: (a) SRAM cells are used and all configuration memory cells and pass-transistor switches are moved to the top layers. (b) The SRAM cells and pass-transistor switches are distributed among the three layers to reduce the overall 3D-FPGA area. (c) Smaller configuration memory cells ($\leq$ 0.7 the size of an SRAM cell) are used and all configuration memory cells and pass transistors are moved to the upper layers. Definitions of terms are as follows: A: Area of baseline 2D-FPGA, LB: Logic Block, RR: Routing Resource, LB-SRAM: Configuration Memory Cells in LB, RR-SRAM: Configuration Memory Cells in RR, PT: Pass transistor.

prove logic density by over 2X. For $\eta \leq 0.7$, the logic density improvement stays at around 3.23X because the area becomes limited by the logic and interconnect that can be implemented only in the CMOS layer. Smaller memory cells, however, can be useful. Since the switch layer at $\eta = 0.7$ is only 84% full, more programmability can be provided by having a larger number of smaller memory cells and adding more switches. To take full advantage of such additional programmability, the FPGA architecture would need to be optimized for 3D implementation.

3. 3D-FPGA DELAY

In the previous section, we quantified the potential improvement in logic density of a monolithically stacked 3D-FPGA over the baseline 2D-FPGA. This improvement is obtained by stacking the programming overhead, which is interspersed with the logic blocks in the 2D-FPGA, on top of the logic blocks. Stacking also makes interconnect lengths shorter, which in turn results in lower interconnect delay and dynamic power consumption. In this section we quantify the improvement in delay of a 3D-FPGA relative to the baseline 2D-FPGA for four CMOS technology generations.

Our methodology for comparing delay is as follows:

1. We derive analytical models for interconnect delays and use them to determine optimized interconnect parameter values, i.e., values for the pass-transistors in the connection boxes, buffer sizes in the switch boxes, and the number and sizes of buffers inserted in long interconnects.

   (a) We assume simple RC circuit models for transistors and interconnects and use the Elmore delay as a measure of circuit delay. This approach yields simple analytical expressions that can be easily optimized to determine values for the interconnect parameters.

   (b) We check the accuracy of the results by performing HSPICE simulations in a 65nm CMOS technology.

2. We quantify the interconnect delay improvement achieved using the 3D-FPGA relative to the baseline 2D-FPGA as follows:

   (a) We compute the interconnect parameter values for a 64×64 LB baseline 2D-FPGA implemented in four deep submicron CMOS technology nodes (180nm, 130nm, 90nm, and 65nm).

   (b) We scale the length of each interconnect type in the 2D-FPGA by the *3D wire scaling factor* $0 < r < 1$ (see Subsection 3.2), and use the Elmore delay expressions to select optimized interconnect parameter values for each interconnect and in each technology.

   (c) We compute the interconnect delays for the 2D and 3D-FPGAs using the optimized interconnect parameter values for the four technology nodes.

3. We quantify the improvement in the overall system performance as follows:

   (a) We use VPR to place and route the 20 largest MCNC benchmark circuits in the baseline 2D-FPGA.

   (b) We then modify VPR to take into consideration the inserted buffers in long interconnects and use it to compute the net delays assuming the optimized transistor and buffer sizes for the 65nm technology.

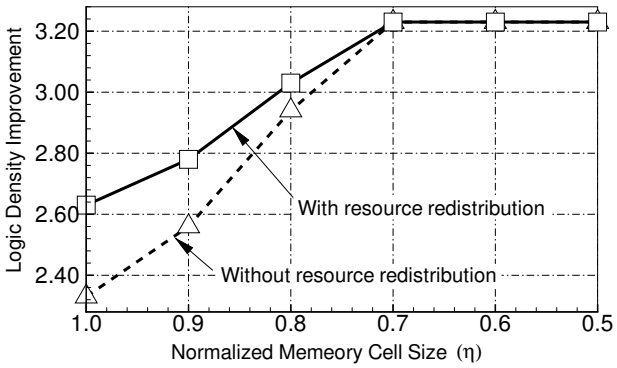


Figure 5: Logic density improvement of 3D-FPGA as a function of the normalized configuration memory cell size.

(c) Assuming the same routing as in the 2D-FPGA, we compute the corresponding net delays in the 3D-FPGA as a function of r for each benchmark circuit.

(d) Finally, we compute the improvements in the geometric average of the point-to-point delay and the critical path delay for each design.

In the following subsections we provide details of the above methodology and present the results and conclusions.

3.1 Interconnect Delay Modeling and Optimization

In this subsection we develop analytical delay models for interconnects and use them to obtain optimized interconnect parameter values.

To develop the analytical delay models, we construct RC-circuit models for each interconnect type and use Elmore delay [20] as a measure of circuit delay. As discussed in [21], this approach yields delay estimates that are accurate to within $10-20\%$ of true delays. We assume the transistor and metal wire RC models shown in Figure 6.

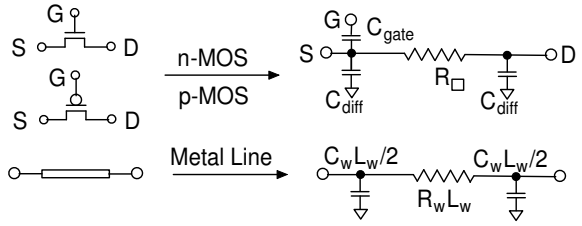


Figure 6: RC circuit model for CMOS transistors and metal wires. C_{gate} is the equivalent transistor gate capacitance (in fF/□), C_{diff} is the transistor diffusion capacitance (in fF/μm), $R_{\square}$ is the transistor channel resistance (in Ω/□), C_{w} is the metal wire capacitance (in fF/mm), R_{w} is the metal wire resistance (in Ω/mm), and L_{w} is the length metal wire length.

To find the values of the equivalent transistor parasitic parameters C_{gate}, C_{diff}, and $R_{\square}$, we use the calibration circuits depicted in Figure 7 and HSPICE simulations. For example, to find C_{gate}, we perform HSPICE simulations to determine the value of C_1 in the figure such that $t_1 = t_2$. The device models used in HSPICE are based on the Berkeley Predictive Technology Model (BPTM). To quantify the impact of technology scaling on FPGA performance, we consider four technology nodes, 180nm, 130nm, 90nm, and 65nm. We then validate the models by performing HSPICE simulations using a foundry supplied model for a 65nm node. The BPTM model is also used to determine R_{w} and C_{w} for the metal wires.

We classify the interconnects into two groups, *short*, which includes Single and Double interconnects, and *long*, which includes HEX-3, HEX-6, and Global interconnects. We measure the interconnect length by the number of tiles it spans, N. Thus for Single interconnect $N = 1$, etc. The interconnect parameters that we aim to optimize are listed in Table 1. Figure 8 depicts the circuits we use to determine the interconnect parameters in the table.

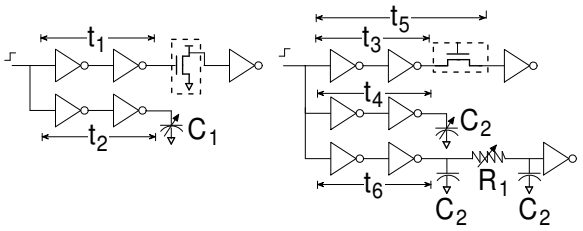


Figure 7: Calibration circuits to determine transistor parasitic values in Figure 6. The left circuit is to determine C_{gate} and the right circuit is to determine C_{diff} and $R_{\square}$.

First, we consider the delay for a short interconnect (Single or Double) (Figure 8(a)). The Elmore delay is given by

$$\begin{aligned} d_{\text{s,N}} = {} & R_{\text{in}}\left(C_{\text{in}} + 3C_{\text{MUX}}\right) + \left(NR_{\text{in}} + \sum_{k=0}^{N-1} kR_{\text{w}}L\right)\frac{C_{\text{w}}L}{4} \\ & + \left(N\left(R_{\text{in}} + \frac{R_{\text{w}}L}{2}\right) + \sum_{k=0}^{N-1} kR_{\text{w}}L\right)\left(\frac{C_{\text{w}}L}{2} + C_{\text{load,int}}\right) \\ & + \left(N\left(R_{\text{in}} + R_{\text{w}}L\right) + \sum_{k=0}^{N-1} kR_{\text{w}}L\right)\frac{C_{\text{w}}L}{4} \\ & + \left(R_{\text{in}} + NR_{\text{w}}L + R_{\text{MUX}}\right)\left(C_{\text{out}} + C_{\text{MUX}}\right) \\ & + \left(R_{\text{in}} + NR_{\text{w}}L\right)3C_{\text{MUX}}. \end{aligned} \quad (1)$$

Here $C_{\text{in}} = m_N C_{\text{diff}}\delta(1+\beta)$, where β is the ratio of the PMOS to NMOS buffer transistor widths, δ is the gate width of a minimum-size transistor, $R_{\text{in}} = R_{\square}/m_N$, $R_{\text{MUX}} = R_{\square}$, $C_{\text{MUX}} = C_{\text{diff}}\delta$, $C_{\text{load,int}} = \gamma_{\text{int}}C_{\text{diff}}\delta$, where γ_{int} depends on the number of LB inputs and outputs and the connectivity parameter F_c [13], and $C_{\text{out}} = m_N C_{\text{gate}}\delta(1+\beta)$.

m_{N}	SP buffer size for interconnect of length N
x	PT size from LB output to CB
y	PT size from interconnect to LB input
l_N	Number of buffers inserted in a long interconnect
n_{N}	Size of inserted buffer in long interconnect
$\gamma_{\text{o}}\delta$	Total PT diffusion width on segment from LB output
$\gamma_{\text{i}}\delta$	Total PT diffusion width on segment to LB input
$\gamma_{\text{int}}\delta$	Total PT diffusion width on Short interconnect
F_{c}	Connection box connectivity [13]

Table 1: Definitions of interconnect parameters.

Now, we consider the following dimensionless parameters that represent the relative values of the transistor parasitics to the wire parasitics.

$$\alpha_1 = \frac{R_{\square}}{R_{\text{w}} \times 1\text{mm}},\ \alpha_2 = \frac{C_{\text{gate}}\delta(1+\beta)}{C_{\text{w}} \times 1\text{mm}},\ \alpha_3 = \frac{C_{\text{diff}}\delta(1+\beta)}{C_{\text{w}} \times 1\text{mm}}. \quad (2)$$

Substituting from the definitions and (2) for a short interconnect, the Elmore delay, normalized with respect to

$R_{\rm w}C_{\rm w}$, can be expressed as

$$\frac{d_{\rm s,N}}{R_{\rm w}C_{\rm w}} = \frac{N^2L^2}{2} + \left(\frac{3\alpha_1}{4m_N} + m_N\alpha_2 + \frac{(N-1)\gamma_{\rm int}+8}{2(1+\beta)}\alpha_3\right)NL + \left(1 + \frac{4}{1+\beta} + \frac{(4+N)\gamma_{\rm int}}{m_N(1+\beta)}\right)\alpha_1\alpha_3 + (1+m_N)\,\alpha_1\alpha_2. \quad (3)$$

Next consider the circuit in Figure 8(b), where an LB output is connected to a Double interconnect. The Elmore delay for this circuit is given by

$$\begin{aligned} d_{\rm l,o} = {} & R_{\rm LB,buf}\left(C_{\rm LB,buf} + C_{\rm PT,diff}\right) \\ & + \left(R_{\rm LB,buf} + R_{\rm PT}\right)\left(C_{\rm PT,diff} + 3C_{\rm w}L/4 + C_{\rm load,o}\right) \\ & + \left(R_{\rm LB,buf} + R_{\rm PT} + R_{\rm w}L\right)\left(C_{\rm load,int} + 3C_{\rm w}L/4\right) \\ & + \left(R_{\rm LB,buf} + R_{\rm PT} + 3R_{\rm w}L/2\right)\left(3C_{\rm MUX} + C_{\rm w}L/4\right) \\ & + \left(R_{\rm LB,buf} + R_{\rm PT} + \frac{3R_{\rm w}L}{2} + R_{\rm MUX}\right)\left(C_{\rm out} + C_{\rm MUX}\right) \\ & + \left(R_{\rm LB,buf} + R_{\rm PT}\right)\left(C_{\rm w}L/2 + 3C_{\rm MUX}\right). \end{aligned} \quad (4)$$

Here $R_{\rm LB,buf} = R_{\square}/b$, $C_{\rm LB,buf} = bC_{\rm diff}\delta(1+\beta)$, where b is the LB buffer size, $R_{\rm PT} = R_{\square}/x$, $C_{\rm PT,diff} = xC_{\rm diff}\delta$, $C_{\rm load,o} = \gamma_{\rm o}C_{\rm diff}\delta$, and m_2 is SP buffer size for a Double interconnect. Substituting from these definitions and (2), the normalized Elmore delay is given by

$$\frac{d_{\rm l,o}}{R_{\rm w}C_{\rm w}} = \frac{9L^2}{8} + \left(\frac{9}{4}\left(\frac{1}{b}+\frac{1}{x}\right)\alpha_1 + \frac{3m_2\alpha_2}{2} + \frac{\gamma_{\rm o}+6}{1+\beta}\right)L + \frac{2b+b\beta+x}{b(1+\beta)} + \left(\frac{1}{b}+\frac{1}{x}\right)\frac{x+2\gamma_{\rm int}+7}{1+\beta}\alpha_1\alpha_3 + m_2\left(1+\frac{1}{b}+\frac{1}{x}\right)\alpha_1\alpha_2. \quad (5)$$

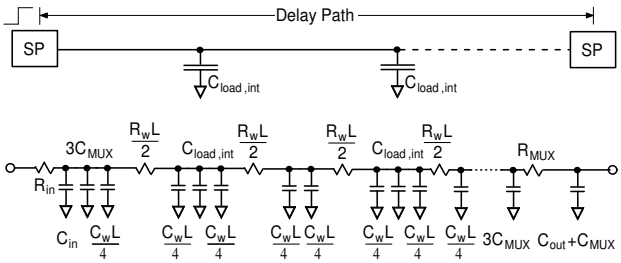
Next we consider the circuit in Figure 8(c), where an LB input is connected to a Double interconnect. The Elmore delay is given by

$$\begin{aligned} \frac{d_{\rm l,i}}{R_{\rm w}C_{\rm w}} = {} & R_{\rm in}\left(C_{\rm in} + 3C_{\rm MUX} + \frac{C_{\rm w}L}{4}\right) \\ & + \left(R_{\rm in} + \frac{R_{\rm w}L}{2}\right)\left(\frac{C_{\rm w}L}{2} + C_{\rm load,int}\right) \\ & + \left(R_{\rm in} + R_{\rm w}L\right)\left(\frac{C_{\rm w}L}{2} + C_{\rm PT,diff}\right) \\ & + \left(R_{\rm in} + R_{\rm PT} + R_{\rm w}L\right)\left(C_{\rm PT,diff} + C_{\rm load,i} + C_{\rm LB,MUX}\right) \\ & + \left(R_{\rm in} + R_{\rm PT} + R_{\rm w}L + R_{\rm LB,MUX}\right)C_{\rm LB,MUX} \\ & + \left(R_{\rm in} + R_{\rm w}L\right)\left(\frac{C_{\rm w}L}{4} + 3C_{\rm MUX}\right). \end{aligned} \quad (6)$$

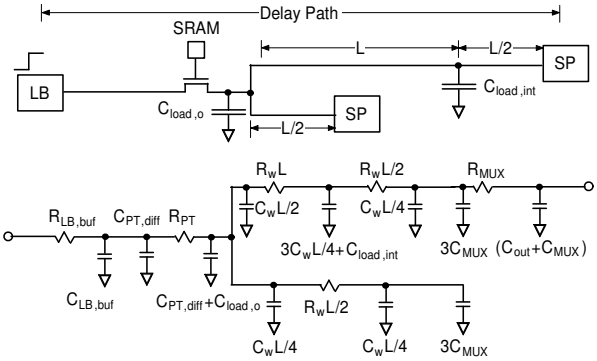
Here $R_{\rm LB,MUX} = R_{\square}$, $C_{\rm LB,MUX} = C_{\rm diff}\delta$, and $C_{\rm load,i} = \gamma_{\rm i}C_{\rm diff}\delta$. Equation (6) can be rewritten as

$$\frac{d_{\rm l,i}}{R_{\rm w}C_{\rm w}} = L^2 + \left(\frac{3\alpha_1}{2m_2} + \frac{y+5}{1+\beta}\alpha_3 + \frac{\gamma_{\rm i}+2x}{2(1+\beta)}\right)L + \left(1 + \frac{1}{1+\beta} + \frac{y+8}{m_2(1+\beta)} + \frac{y+2}{y(1+\beta)}\right)\alpha_1\alpha_3 + \frac{\gamma_{\rm int}+x}{m_2(1+\beta)}\alpha_1. \quad (7)$$

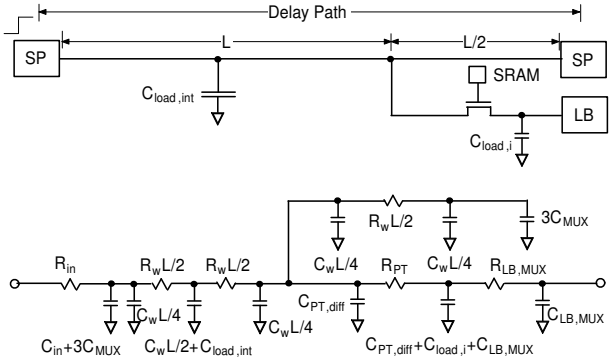
The delay Equations (3), (5) and (7) are functions of several parameters. We assume given are the number of LB inputs and outputs, the LB buffer size b, the number of each type of interconnect in the routing channel, and F_c.



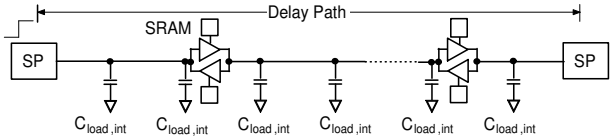
(a) Short interconnect.



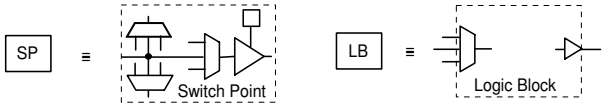
(b) Interconnect from the output of an LB to a neighboring switch box, SP : switch point.



(c) Interconnect from a switch box to the input of an LB.



(d) Long interconnect, RC model omitted for space limitation.



(e) Legends in the above diagrams.

Figure 8: Logic view and circuit model of various FPGA interconnects.

The remaining unknowns, thus, are the switch point buffer sizes m_N for the short interconnects and the pass-transistor sizes x and y. To determine optimized values for these parameters, we use the following heuristic, which is motivated by the fact that in deep submicron technologies the pass transistor loading on the interconnects is significantly lower than the metal wire loading. First we assume nominal values for γ_{int} and determine m_N, $N = 1, 2$, from (3). We then substitute the value of m_2 in (4) and (6) and optimize for x in (4) assuming nominal value of y and for y in (6) assuming nominal value for x.

Finally we consider the delay of a long interconnect, where we allow buffer insertion to reduce delay. We assume l_N bi-directional buffers each of size n_N inserted at regular intervals along the interconnect of length N as depicted in Figure 8(d). The total Elmore delay can be derived as for previous cases. The Elmore delay is then optimized in l_N and n_N for each long interconnect type.

3.2 Interconnect Delay Improvement

In the previous subsection, we developed analytical expressions for interconnect delays and showed how they can be used to optimize the selection of various interconnect parameters. In this section, we use these results to compare the delay of each interconnect type in the monolithically stacked 3D-FPGA to its counterpart in the baseline 2D-FPGA. We parametrize the results by the *wire scaling factor* $0 < r < 1$, which is the ratio of the 3D-FPGA tile width to the baseline 2D-FPGA tile width. Since, as we discussed, the area of the 3D-FPGA depends on the size of the configuration memory cell used, r also depends on the size of the memory cell used. Figure 9 plots the 3D wire scaling factor as a function of the normalized memory cell size η. Note that r monotonically decreases down to 0.56 at $\eta = 0.7$ and then stays constant for $\eta \leq 0.7$.

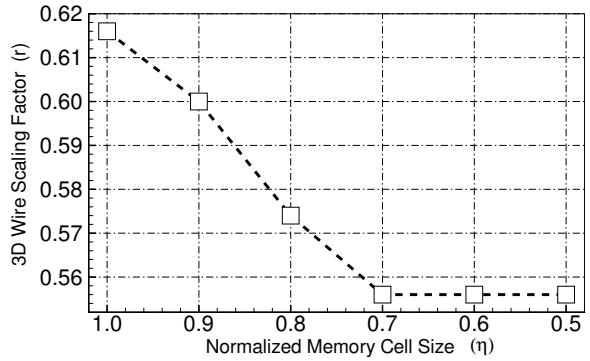


Figure 9: Relationship between the 3D wire scaling factor r and the normalized memory cell size η.

To quantify the interconnect delays, we need to know the FPGA tile width L, the size of the FPGA (i.e., number of LBs), and the specific design of routing resource. As an illustration, we assume $L = 8215\lambda$ (as estimated in our study), an array size of 64×64 LBs, LB buffer size $b = 4$, and 24 Single, 20 Double, 12 HEX-3, 16 HEX-6, and 4 Global interconnects in each routing channel. Each LB is assumed to have $K_{\text{i}} = 16$ input pins and $K_{\text{o}} = 4$ output pins that can be connected to routing channel. Let W be the total number of interconnects in each routing channel, which is

	CB	Single	Double	HEX-3	HEX-6	Global
	x,y	m_{N}		m_{N}, l_{N}, n_{N}		l_{64}, n_{64}
2D	11,7	21	27	25,1,17	26,2,19	32,18
3D	7,6	15	19	17,0,0	19,1,14	16,12

Table 2: Pass-transistor and buffer sizes for baseline 2D-FPGA and 3D-FPGA with $r = 0.56$ assuming a 65nm technology node.

76 in our study. We further assume that the connection box connectivity $F_{\text{c}} = 0.5W = 38$ and the routing matrix of the switch box in [22]. Note that our assumptions yields $\gamma_{\text{o}} = F_{\text{c}}x$, $\gamma_{\text{i}} = F_{\text{c}}y$, and $\gamma_{\text{int}} = \frac{40F_{\text{c}}}{W}\left(\frac{16x+4y}{20}\right) = 16x+4y$. Table 2 lists the values for the various pass-transistor and buffer size parameters as determined by the procedure mentioned in the previous subsection for the baseline 2D-FPGA array and 3D-FPGA with $r = 0.56$ assuming a 65nm CMOS technology node.

Figure 10 is a log-log scale plot of the delay improvement for each interconnect type in the baseline 2D-FPGA under these assumptions, i.e., the ratio of each interconnect delay in the 2D-FPGA to its counterpart in the 3D-FPGA, as a function of r assuming a 65nm CMOS technology. Note that the interconnect delays follow an exponential law (linear on the log-log scale) with exponent ranging from around -0.2 for Single interconnects to -1.1 for Global interconnects. As expected, long interconnect delays are reduced much more by 3D than short interconnects. To explore the effect of

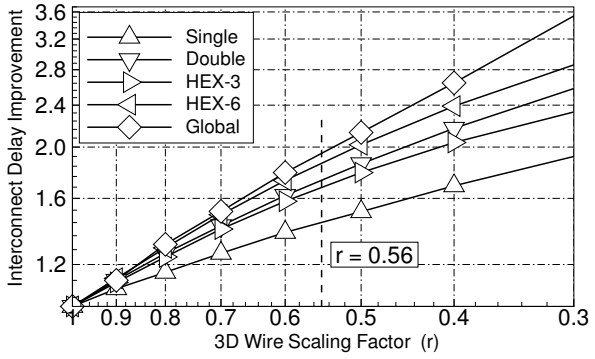


Figure 10: Delay Reduction using 3D at 65nm technology node.

technology scaling on delay improvement, Figure 11 plots the interconnect delay improvements for different interconnect types and different technology nodes at $r = 0.56$. Note that the improvement in delay increases with technology scaling mainly due to the degradation in wire performance. The dips between Double and HEX-3 interconnects is due to the effect of buffer insertion on delay of long interconnects.

3.3 System Performance Improvement

In the previous subsection, we quantified the reduction in interconnect delays achieved using a monolithically stacked 3D-FPGA for different technology nodes. In this section we quantify the impact of these delay reductions on the overall performance of application designs implemented in such FPGA. Our approach is to map the 20 largest MCNC benchmark circuits [23] into the baseline 2D-FPGA, determine the

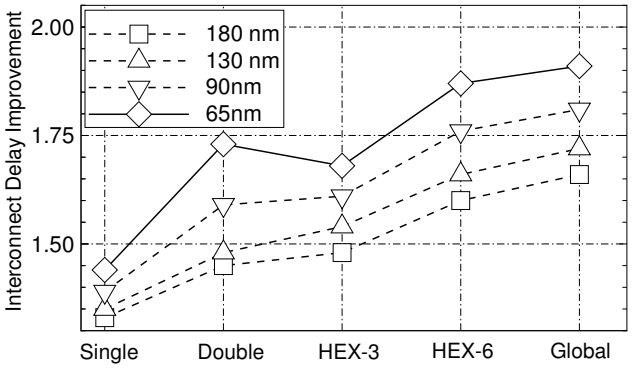


Figure 11: Interconnect delay improvements for different submicron technology nodes.

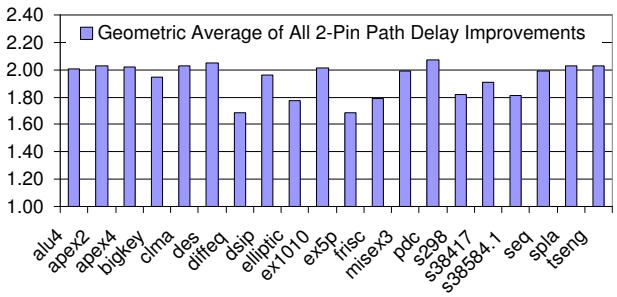


Figure 12: Improvement in geometric average pin-to-pin delay for MCNC benchmark circuits mapped into a 64×64 LB FPGA implemented in 65nm technology.

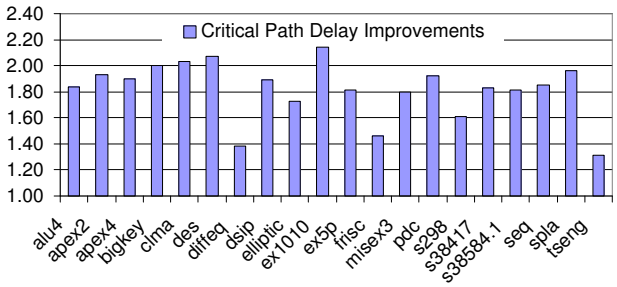


Figure 13: Critical path delay improvement for MCNC benchmark circuits mapped into a 64×64 LB FPGA implemented in 65nm technology.

pin-to-pin delays for each net, i.e., the delay from the net output to each of its inputs, then scale the interconnects by the wire scaling factor r and determine the corresponding delays for the 3D-FPGA.

To map a benchmark circuit into the baseline 2D-FPGA, we first use T-VPACK [13] to pack its LUTs (lookup tables) and FFs (flip-flops) into logic clusters that can each be mapped into an LB. VPR [13] is then used to perform placement and routing. Using the approach described in [22], we modified both T-VPACK and VPR to handle the Virtex-II style LB and to create the correct routing graph for our baseline 2D-FPGA. To compute net delays from the placed and routed designs, we modified the timing analysis code of VPR to take into consideration the inserted buffers in long interconnects.

To compare the system performance of the 3D-FPGA to that of the baseline 2D-FPGA, we use two metrics; the improvement in the geometric average of the pin-to-pin delays, and the improvement in critical path delay, which includes the LB delays along the path. By improvement here we mean the ratio of the delay in the baseline 2D-FPGA to that in the 3D-FPGA. Results for the largest 20 MCNC benchmark circuits are plotted in Figures 12 and 13. Note that the improvement range between 1.7 and 2.05 for the geometric average and between 1.31 and 2.14 for the critical path delay. The reason the improvement in critical path delay on average is slightly lower is that although a critical path is more likely to contain more long interconnects than a point-to-point path delay, the added LB delays, which do not change from 2D to 3D, reduce the overall critical path delay improvement. In general the improvement numbers are quite consistent with the range of interconnect delay improvements in Figure 11.

4. 3D-FPGA POWER CONSUMPTION

In this section, we quantify the reduction in dynamic power consumption achieved using the 3D-FPGA relative to the baseline 2D-FPGA. The power consumed in an FPGA can be divided into static and dynamic power components. Static power represents a growing fraction of the total power consumed in an FPGA [4, 19]. It can be addressed by several techniques including using multiple supply voltages, multiple transistor threshold voltages, and power gating. Our 3D approach may help reduce the area overhead required to implement such techniques. Since this is difficult to quantify at this point, we focus our discussion on dynamic power.

Dynamic power consumption is due to charging and discharging of circuit parasitics as well as the short-circuit currents during signal switching. Previous study [4] has shown that only 10% of the total current in an FPGA interconnect is due to short-circuit currents. To simplify our analysis we only consider the dynamic power due to the parasitics and emulate the contribution of short-circuit currents by appropriately increasing the values of the parasitic capacitances.

The dynamic power consumed in an FPGA can be divided into three components, the dynamic power consumed in the logic blocks P_{LB}, the dynamic power consumed in the interconnects P_{int}, and the dynamic power consumed in the clock networks P_{clk}. Previous studies [1, 4] have shown that 15–20% of the total dynamic power is consumed in the logic blocks, 60–80% is consumed in the interconnects, and around 15% is consumed in the clock networks. Since in this study we assume that the 3D-FPGA uses the same logic block architecture as the baseline 2D-FPGA, the dynamic power consumed by the logic blocks in the 3D-FPGA is the same as that in the baseline 2D-FPGA.

Now, let ψ to be the average activity factor of the signal nets, C_{net} be the equivalent capacitance of the signal net, which includes the capacitances of the wire, side-loads, switch point, and inserted buffers, C_{clk} be the equivalent capacitance of the clock network, which again includes the

wire and buffer capacitances in the network, f_{net} be the interconnect operating frequency, and f_{clk} be the clock frequency. With these definitions, the total dynamic power consumed in the FPGA can be expressed as

$$\begin{aligned} P &= P_{\text{LB}} + P_{\text{int}} + P_{\text{clk}} \\ &= P_{\text{LB}} + \psi V_{\text{DD}}^2 f_{\text{net}} \sum_{\text{all nets}} C_{\text{net}} + C_{\text{clk}} V_{\text{DD}}^2 f_{\text{clk}}. \end{aligned} \quad (8)$$

We compare the dynamic power consumption of the 3D-FPGA to that of the baseline 2D-FPGA as follows.

1. We denote by ϕ_{LB}, ϕ_{int}, and ϕ_{clk}, respectively, the fraction of dynamic power consumed in the logic blocks, the interconnect, and the clock network of the baseline 2D-FPGA. Thus, $\phi_{\text{LB}} + \phi_{\text{int}} + \phi_{\text{clk}} = 1$. We choose ϕ values that are consistent with recent studies [1, 4].

2. We added code to VPR to extract the equivalent capacitance, C_{net}, of each signal net in the placed and routed benchmark circuit, for both the baseline 2D-FPGA and the 3D-FPGA with $r = 0.56$. The equivalent transistor gate and diffusion capacitances are obtained using the calibration circuits in Figure 7. However, instead of matching delays, we match charge stored on the capacitances over a 1μsec period, which yields slightly larger capacitances. We then find the dynamic power consumption improvement factor for interconnects, $\xi_{\text{int}} \geq 1$, which is the ratio of the dynamic power consumed by the interconnects in the 2D-FPGA to that in the 3D-FPGA for a particular benchmark circuit in MCNC suite.

3. We repeat the same procedure for the clock network to find the dynamic power improvement factor for the clock network, $\xi_{\text{clk}} \geq 1$.

4. The results are combined to find the improvement in the total dynamic power consumption, ξ, given by

$$\xi = 1 \Big/ \left(\phi_{\text{LB}} + \frac{\phi_{\text{int}}}{\xi_{\text{int}}} + \frac{\phi_{\text{clk}}}{\xi_{\text{clk}}} \right). \quad (9)$$

5. The above procedure is repeated for each of the 20 MCNC benchmark circuits and ξ is computed for each circuit. Finally, the geometric average of each ξ for different designs, Ξ, is determined and used as an overall measure of power saving.

To find the equivalent capacitance for the global clock network, we assume, as has been done in previous studies (e.g., [19]), that an H-tree distribution network is employed in both the baseline 2D-FPGA and the 3D-FPGA. We assume a distributed buffer scheme, since it achieves lower delay and lower skew. Each LB is served by a clock buffer. The size of the chip in tiles determines the depth of the clock tree.

The equivalent capacitance of the clock distribution network is obtained by adding the total wire capacitance $C_{\text{clk,wire}}$, buffer capacitance $C_{\text{clk,driver}}$, and load capacitances $C_{\text{clk,load}}$ for the network. To obtain numerical values for the equivalent capacitance in the baseline 2D-FPGA, we assume a 64 $\times$ 64 LB array and optimize the buffer sizes for each of the four technology nodes. To estimate the equivalent capacitance for the 3D-FPGA, we scale the wire lengths of the network in the 2D-FPGA by the 3D wire scaling factor r and re-optimize the buffer sizes as for long interconnects. Since the clock network load capacitance is primarily due to the input capacitance of the flip-flops residing in the LBs, we assume that $C_{\text{clk,load}}$ remains the same for 3D-FPGA independent of the value of r.

Now, we are ready to compute the improvements in dynamic power savings achieved by the 3D-FPGA. We assume that $\phi_{\text{LB}} = 0.15$, $\phi_{\text{int}} = 0.65$, and $\phi_{\text{clk}} = 0.2$ and compute ξ_{int} and ξ_{clk} for different values of r and for each of the four technology nodes (180nm, 130nm, 90nm, and 65nm) and then compute ξ_{total} using (9). The results are plotted in Figure 14. As expected, the total improvement in dynamic power, however, depends strongly on r. Note, however, that the improvement in dynamic power consumption does not change much with technology. This is mainly due to the fact that the results are normalized with respect to wire length, supply voltage, and operating frequency. Additionally, C_{gate}, C_{diff}, and C_{w} do not change much with scaling from 180nm down to 65nm.

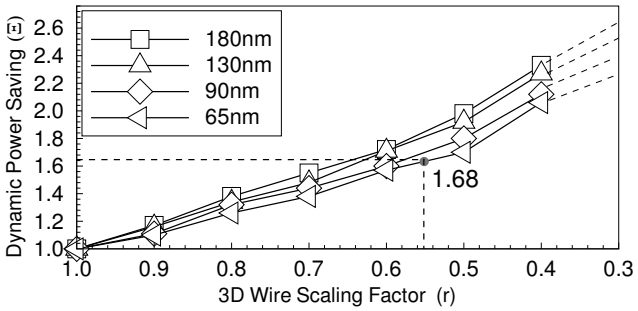


Figure 14: Relative power saving of 3D-FPGA for different technologies.

5. CONCLUSION

The paper discussed the performance benefits of a monolithically stacked 3D-FPGA. A Virtex-II style 2D-FPGA fabric is used as a baseline for comparison. A technology independent FPGA area model is used to compare the logic density of the 3D-FPGA to the baseline 2D-FPGA as a function of programming memory element size. An analytical model for interconnect is used to estimate the delay and dynamic power consumption of the 3D-FPGA compared to the baseline FPGA implemented in several deep submicron CMOS technology nodes and the results are corroborated with HSPICE simulations.

It is shown that the size of the configuration memory cell plays a key role in the degree of performance improvement achieved by a monolithically stacked 3D-FPGA. For a memory cell that is ≤ 0.7 the area of an SRAM cell, we showed that a 3D-FPGA can achieve 3.2 times higher logic density, 1.7 times lower critical path delay, and 1.7 times lower total dynamic power consumption than the baseline 2D-FPGA at the 65nm technology node. Since the 3.2X improvement in logic density requires the addition of only a few mask layers to a standard CMOS technology, a monolithically stacked 3D-FPGA should have significantly lower manufacturing cost than a conventional 2D-FPGA for the same logic capacity.

The improvement results reported are based on several assumptions and approximations that warrant further investigation.

- The 3D-FPGA area results we reported need to be verified by performing some detailed layouts in a 3D technology.
- The delay and power improvements assumed that the pass-transistor switches have the same characteristics as the nMOS devices in the CMOS layer. The accuracy of this assumption depends on the technology used to build these devices. The analytical models we used for delay and dynamic power consumption, however, can be readily used to quantify the improvements for any given pass-transistor switch characteristics.
- The RC models for the interconnects ignored the parasitics of the 3D via. Depending on the 3D technology used, this may need to be taken into consideration.
- The delay and power results assumed that all transistors have the same threshold voltage and that a single supply voltage is used. In the most advanced technologies, devices with different threshold voltages are available.

Finally, our analysis did not assume any optimization of the 3D-FPGA architecture to take better advantage of the added layers. It is expected that significant additional improvements in interconnect delays and dynamic power consumption can be obtained by optimizing the programmable routing architecture beyond merely optimizing buffer insertion and device sizes.

6. REFERENCES

[1] V. George, *Low Energy Field-Programmable Gate Array*. PhD thesis, UC Berkeley, 2000.

[2] A. DeHon, *Reconfigurable Architectures for General-Purpose Computing*. PhD thesis, MIT, 1996.

[3] E. Kusse and J. Rabaey, "Low-energy embedded FPGA structures," in *Proceedings of International Symposium on Low Power Electronics and Design*, August 1998.

[4] L. Shang, A. S. Kaviani, and K. Bathala, "Dynamic power consumption in Virtex-II FPGA family," in *Proceedings of the 2002 ACM/SIGDA Tenth International Symposium on Field-Programmable Gate Arrays*, pp. 157 – 164, 2002.

[5] V. Degalahal and T. Tuan, "Methodology for high level estimation of FPGA power consumption," in *Proceedings of the Design Automation Conference*, pp. 657 – 660, Jan. 2005.

[6] P. S. Zuchowski, C. B. Reynolds, R. J. Grupp, S. G. Davis, B. Cremen, and B. Troxel, "A hybrid ASIC and FPGA architecture," in *Proceedings of the 2002 IEEE/ACM International Conference on Computer-Aided Design*, pp. 187 – 194, May 2002.

[7] J. Burns, L. McIlrath, J. Hopwood, C. Keast, D. Vu, K. Warner, and P. Wyatt, "An SOI-based three-dimensional integrated circuit technology," in *Proceedings of the SOI Conference*, pp. 20 – 21, Oct. 2000.

[8] J. Burns, L. McIlrath, C. Keast, C. Lewis, A. Loomis, K. Warner, and P. Wyatt, "Three-dimensional integrated circuits for low-power, high-bandwidth systems on a chip," in *Proceedings of the Solid-State Circuits Conference*, pp. 268 – 269, Feb 2001.

[9] A. R. Joshi and K. C. Saraswat, "Nickel induced crystallization of a-Si gate electrode at 500C and MOS capacitor reliability," *IEEE Trans. Electron Devices*, vol. 50, pp. 1058 – 1062, April 2003.

[10] M. Cao, T. Zhao, K. C. Saraswat, and J. D. Plummer, "A simple EEPROM cell using polysilicon thin film transistors," *IEEE Electron Device Letter*, vol. 15, pp. 304 – 306, Aug. 1994.

[11] S. Kaeriyama, T. Sakamoto, H. Sunamura, M. Mizuno, H. Kawaura, T. Hasegawa, K. Terabe, T. Nakayama, and M. Aono, "A nonvolatile programmable solid-electrolyte nanometer switch," *IEEE J. Solid State Circuits*, vol. 40, pp. 168 – 176, Jan. 2005.

[12] A. Rahman, S. Das, A. Chandrakasan, and R. Reif, "Wiring requirement and three-dimensional integration technology for field programmable gate arrays," *IEEE Trans. on VLSI Systems*, vol. 11, pp. 44 – 54, Feb. 2003.

[13] V. Betz and J. Rose, "VPR: A new packing, placement and routing tool for FPGA research," in *Proceedings of the 7th International Workshop on Field-Programmable Logic and Applications*, pp. 213 – 222, 1997.

[14] Xilinx, Inc., *Virtex-II Platform FPGA Handbook*, April 2000.

[15] G. Lemieux and D. Lewis, "Circuit design of routing switches," in *Proceedings of the 2002 ACM/SIGDA Tenth International Symposium on Field-Programmable Gate Arrays*, pp. 19 – 28, 2002.

[16] J. Rose, R. Francis, D. Lewis, and P. Chow, "Architecture of field-programmable gate arrays: the effect of logic block functionality on area efficiency," *IEEE Journal of Solid-State Circuits*, vol. 25, no. 5, pp. 1217–1225, 1990.

[17] E. Ahmed and J. Rose, "The effect of LUT and cluster size on deep-submicron FPGA performance and density," in *the 2000 ACM/SIGDA Eighth International Symposium on Field Programmable Gate Arrays*, Feb. 2000.

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

[19] F. Li, D. Chen, L. He, and J. Cong, "Architecture analysis and automation: Architecture evaluation for power-efficient FPGAs," in *Proceedings of the 2003 ACM/SIGDA tenth international symposium on Field-programmable gate arrays*, 2003.

[20] W. C. Elmore, "The transient analysis of damped linear networks with particular regard to wideband amplifiers," *Jour. of Applied Physics*, vol. 19, no. 1, pp. 55–63, 1948.

[21] N. Nassif, M. Desai, and D. Hall, "Robust Elmore delay models suitable for full chip timing verification of a 600 MHz CMOS microprocessor," in *Design Automation Conference*, pp. 15–19, Jun. 1998.

[22] W. Xu, "VPR for Virtex." Available on: `http://www-unix.ecs.umass.edu/~wxu/jbits/VPR_for_Virtex.htm`.

[23] S. Yang, "Logic synthesis and optimization benchmarks, version 3.0," tech. rep., Microelectronics Center of North Carolina, 1991.