Design and Implementation of Three Dimensional Arithmetic Units

J. Ouyang, G. Sun, Y. Chen, L. Duan, T. Zhang, Y. Xie
Department of Computer Science and Engineering, The Pennsylvania State University
351 IST Building, State College, PA 16802 USA

Abstract— **We describe the design and the implementation of two three dimensional arithmetic units: a 3D adder and a 3D multiplier. Compared to their 2D counterparts, our 3D adder incurs 10.6–34.3% less delay and 11.0–46.1% less energy when the width increases from 12-bit to 72-bit; the 32×32 3D multiplier incurs 14.4% less delay and 6.8% less energy, according to the post place and route results. The prototype chip including the implementations of a 3D adder, a 3D multiplier, and simple test interface has been delivered for fabrication in MIT Lincoln Laboratory.**

I. Introduction

Arithmetic unit is an indispensable part of today's processor. As an example, adders and multipliers not only serve as the key functional units in the pipeline, but are also typically used in MMU, DMA controller, and memory controller for address generation or burst transfer control. Because of the importance of these components, a great amount of research efforts have been devoted to designing fast, power- and area-efficient adders and multipliers. For adders, however, conventional design space exploration have been focused on logic depth, radix, sparsity, and regularity [1]. In contrast, as the feature size of wires kepdf shrinking, the interconnect effect starts to limit and dominate latency, power, and area of the adder [2]. The interconnect effect is, if not worse, equally severe for multipliers, as they are often very complex and incurs high interconnect density. Three dimensional integrated circuits (3DIC) is proposed to reduce the interconnect pressure and improve the performance of interconnect intensive designs. The idea of 3DIC is to partition and distribute a large 2D design onto a number of small dies, and stack them on top of each other. Through silicon vias (TSV) are fabricated to carry signals propagating from one die to another. The advantage of using 3DIC is often twofold: firstly, the device density and routing capacity per unit area is effectively multiplied by the number of dies; secondly, if aligned carefully, components originally distant from each other on a 2D plane can be stacked in a cylinder. In the second case, long 2D interconnects are replaced by TSVs, which typically have lengths at order of the die thickness and hence incur much lower parasitics. A conceptual view of a 3DIC with three device layers is shown in Figure 1.

The benefits of 3DIC has been explored and applied to various types of integrated circuits. Previous work can be categorized according to the granularity at which the architecture is partitioned into sub-designs. At the coarse-granularity level, the cores and cache banks of a CMP are placed onto different layers to reduce latency of on-chip communication [3]. At a finer-granularity level, the cache [4] and the register file [5] is partitioned along the word- or bit-line to reduce the latency and energy. Yet delicate components like arithmetic units can also benefit from 3D partitioning. In [6], each logic level of the Kogge-Stone adder (KS Adder) is placed onto a separate die and the delay is reduced by 32.7% on a 3-layer 3D design. Puttaswamy et al [7], in contrast, partition the KS adder into odd/even bits on a 2-layer 3DIC. Their results show less aggressive improvement (15.4–30.2%). However, when the issue width of the processor increases, the 3D adder is shown to be more scalable than the 2D adder.

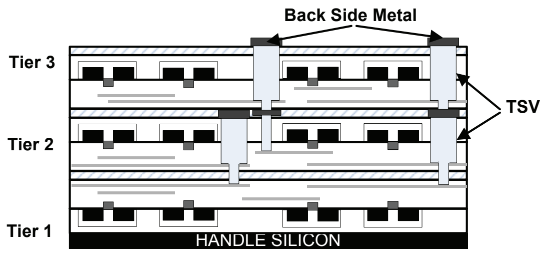

Fig. 1. Cross-sectional view of the 3D stack fabricated by MITLL 3D-FDSOI process

In this paper, we leverage 3DIC to improve the performance of arithmetic units. More specifically, we design and implement a) a Kogge-Stone [8] style 3D adder, and (b) a Wallace Tree style 3D multiplier [9]. The adder and the multiplier, together with simple test interface, are designed for and implemented in MIT Lincoln Laboratory 0.18μm 1.5V 3D process [10]. The prototype chip with dimensions of 1.3mm×1.3mm have been delivered for fabrication.

II. MIT Lincoln Laboratory (MITLL) 3D-FDSOI Process

A. Process

The underlying technology of our design is MIT Lincoln Laboratory (MITLL) 3D fully depleted SOI process (3D-FDSOI) [10]. This is a top-down three-tier process, where each of the three tiers are first fabricated separately, and then assembled into a stack. During chip assembly, TSVs are inserted by etching through the silicon and filling tungsten. The conceptual cross-sectional view of the 3D chip is shown

TABLE I

TECHNOLOGY PARAMETERS OF MITLL 3D-FDSOI PROCESS.

Device Channel Length (μm)	0.18
No. of Routing Layers	3
Poly Feature Size (μm)	0.2
Metal Feature Size (μm)	0.25
TSV Size (μm^2)	3$\times$3
TSV Spacing (less than 5 TSVs) (μm)	1.450
TSV Spacing (5 or more TSVs) (μm)	2.650
Tier Thickness (TSV length) (μm)	7.470
TSV resistance (Ω)	0.115
TSV capacitance (shielded by routing metal) (fF)	4.34
TSV capacitance (isolated from routing metal) (fF)	1.01

in Fig. 1. The important process parameters are generalized in Table I [10] [11].

B. Design Tools

In our design flow, the 3D design kit developed by North Carolina State University specifically for MITLL 3D-FDSOI technology is used. The design kit includes standard cell libraries, utilities, and scripts for adapting conventional 2D EDA tools to design the 3D-chip. In addition to the design kit, a number of in-house scripts are developed to customize and automate design flow. For example, inserting TSVs into the layout is a tedious and error-prone process, and can be largely accelerated by scripts.

III. 3D KOGGE-STONE ADDER

A. Architecture

As can be seen from Fig. 2, we take every triple of bits of a 2D KS adder and place each of them on a different tier. As a result, the KS adder is effectively partitioned into three sub-adders on each tier. In this figure, different types of units are represented by nodes with different shapes: circles represent function units producing initial generation and propagation terms; solid triangles represent carry-merge units while empty triangles represent buffers. The orange, green, and blue colored logic units are placed on different tiers respectively. The levels of the adder are numbered from 0 to l_{max}, and the total width of the adder is $3 \bullet 2^{l_{max}-1}$. The nodes on level 0 produce initial generation and propagation terms. It is interesting to notice that the nodes on levels 1 to l_{max} (carry-merge units and buffers) within each tier form an independent 2D KS adder of height $l_{max} - 1$.

According to Fig. 2 and the above analysis, it can be seen that the only signals crossing tiers are between level 0 and level 1. Different from [7], our adder actually embraces a hybrid architecture where the arities of the first merge stage and other stages are different. Note that we also flip the direction of carry forwarding in the first merge stage, in order to conveniently generate the final sum.

B. Cross-Tier Carry-Merge

We call the first merge stage (level 1) the *Cross-Tier Carry-Merge* stage, where signals have to propagate across the tier boundaries. Directly implementing the architecture would introduce a large number of TSVs and high fanout numbers.

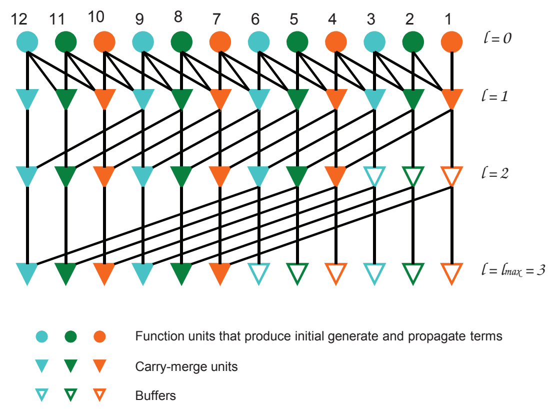

Fig. 2. 3D Kogge-Stone Adder

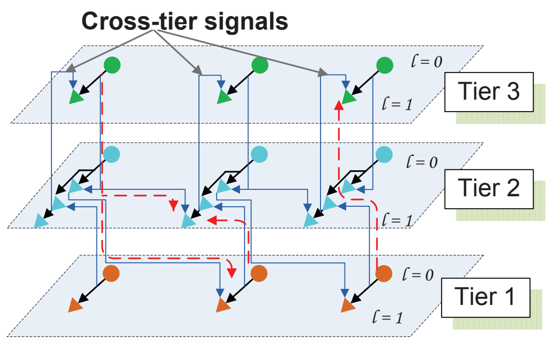

Fig. 3. *Cross-Tier Carry-Merge*. The red dashed arrows indicate how carries are propagated along the short carry chains. For clarity, only three bits on levels 0 and 1 on each tier are shown.

In addition, high TSV density will increase variation and the risk of failing chips, and require tighter design rules [10] that incur additional area overheads.

Taking all the effects into consideration, we choose not to directly implement the arity-of-three carry-merge, but use length-of-three carry-chains for *Cross-Tier Carry-Merge*. A conceptual view of this scheme is shown in Figure 3. Note that this method has one logic stage overhead, however, it still has the same logic depth as a 2D KS adder (a 2D KS adder has a logic depth of $\lceil \log_2(3 \bullet 2^{l_{max-1}}) \rceil = l_{max} + 1$). In the meantime, the layout width and longest global interconnect length of the 3D adder are both approximately one third of its counterpart.

C. Flexibility of the Architecture

Another view to consider our 3D adder architecture proposed in this section is that the *Cross-Tier Carry-Merge* stage is part of the preprocessing of an ordinary 2D Parallel Prefix Adder (PPA). Then the 3D adder can be viewed as a collection of 2D PPAs sharing a common preprocessing stage. Additional benefits can be obtained from this observation. Firstly, some simple modifications to the carry-merge unit allows for dynamically reconfiguring the adder into three independent sub-adders. This may be beneficial for situations where resource sharing is desired. Secondly, other architectures can be used by the sub-adders. For example, the three sub-adders can comprise a carry-select adder, a Kent-Brunt adder, and a carry-propagate adder. This allows for trading off the speed of part of the design with power or area. Finally, even the width of the sub-adders does not have to be the same. In sum, this architecture allows for large design space to be explored for the optimal design.

IV. 3D Wallace Tree Multiplier

Reduction tree based multiplier is another example of minimum logic depth ($O(\log N)$), large area, and high interconnect density arithmetic units. Fig. 4 shows the architectural view of a 16×16 2D Wallace Tree multiplier, which has three levels of four-two counters to reduce a 16-row partial product array into two addends, and a PPA to generate the final product. It is not difficult to identify the long wires of the 2D multiplier (blue arrows shown in the figure).

To architect the 3D Wallace Tree multiplier, we split the last two levels of the multiplier horizontally. For example, the 16×16 multiplier shown in Fig. 4 falls into three parts as the result of this splitting: two sub-trees consisting of levels 1 and 2 four-two counters, and one root level containing the last level four-two counters and the PPA. Then the two sub-trees are placed on the top and bottom tiers respectively, and the root level is placed on the middle layer. The long wires (blue arrows) are replaced by the TSVs in the 3D multiplier, and hence the latency and power incurred by long wires are both reduced.

In addition to the delay and power benefit, the 3D Wallace Tree multiplier also shares the similar design flexibility with the 3D KS adder. If separate adders are added to tier 3 and tier 1 respectively, the 3D multiplier in Fig. 4 can also work as two 16×8 multipliers. This allows for dynamically reconfiguring a wide multiplier into short multipliers. Note that although dynamic reconfiguration is also possible with 2D multiplier, it is relatively more difficult to squeeze two more PPAs into the design since the area and the interconnect complexity is already trepidating. Whereas 3DIC provides ample space to hold these additional logics and interconnects.

V. Experiment

A. Implementation

1) Methodology: Both the 3D adder and the 3D multiplier are designed and implemented based on MITLL 3D-FDSOI process as mentioned in Section II. The arithmetic units are first partitioned in to sub-designs in the way described in Section III and IV, and compiled separately in Design Compiler. The 3D adder is firstly manually placed, and then routed by the timing-driven router of SoC Encounter. Due to time limitation, we only manually floorplan the multiplier according to its hierarchical structure, and then use SoC Encounter to do place and route. The above approaches are also used to layout a 2D adder and a 2D multiplier to evaluate performance differences.
TSV insertion is one of the most important stpdf and unique from traditional 2D flow. After synthesis and before place-and-route, the TSV instances have to be added to the netlist and fully connected. Then in SoC Encounter, these instances

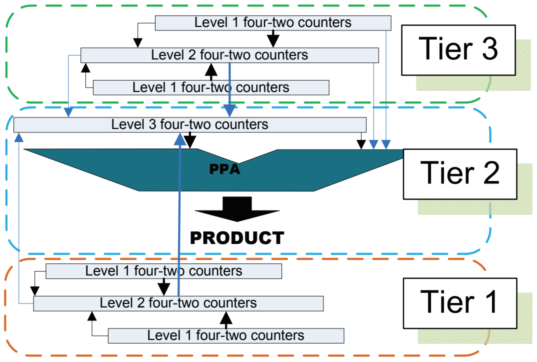
Level 1 four-two counters
Level 2 four-two counters
Level 1 four-two counters
Tier 3
Level 3 four-two counters
PPA
Tier 2
PRODUCT
Level 1 four-two counters
Level 2 four-two counters
Level 1 four-two counters
Tier 1

Fig. 4. The architecture of a 16×16 2D Wallace Tree multiplier with a PPA to form the final product. For clarity, the partial product array is neglected. We partition the last two levels of the multiplier and put them onto different tiers as is shown. The blue arrows indicate the long wires of the 2D multiplier, which are TSVs in the 3D case

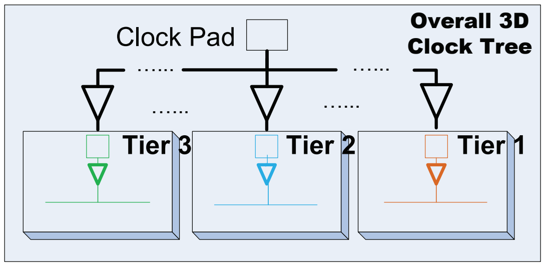

Fig. 5. Hierarchical 3D clock tree. Note that the root of the entire tree is the clock pad, while the root of each subtree is a TSV carrying the clock signal.

are manually placed and aligned[1] in the manual placement and floorplan phase as mentioned above. In all stpdf and in the post place-and-route evaluation, the TSV parasitics in Table I [10] [11] are back-annotated in the constraint files. The photograph of tier 3 of the chip is shown in Fig. 6

2) Pad Limited Design: Our prototype chip is a pad limited design, since both arithmetic units have relatively small footprints. For this reason we use shift registers shared by both the multiplier and the adder to load test vectors and output results. The chip has 21 signal I/Os, in which 16 pins are used to shift in/out to and from the on-chip registers, 4 pins for control signals, and 1 pin for clock. The number of total pins amounts to 43 pins including power pins, increasing the chip size to 1.3mm$\times$1.3mm.

Even with only 43 pins, the layout is still quite sparse. Although the total power of the chip is only about 30mW, we insert about 2000 power TSVs to utilize the unused regions. The reason of using far more than required TSVs is increasing the yield of the prototype chip and helping heat dissipation.

3) 3D Clock Tree Insertion: The most challenging part in our flow is confining the clock skews of the 3D clock tree to an acceptable level, since there is no existing commercial tool to do 3D clock tree synthesis. The problem is handled by a bottom-up clock tree synthesis approach. Firstly, we synthesize a 2D clock-tree on each tier separately. The 2D clock tree is

[1]This process is actually automated by in-house scripts

grown from a clock buffer placed side-by-side with a TSV carrying the cross-tier clock signal, so as to easily determine the RC parasitics from the TSV to the clock buffer. We can get a detailed report of the delay, the skew, and the load on the root of each clock tree after the 2D clock tree synthesis. As a final step, each tier is treated as a hard macro with a clock pin, and a final clock tree synthesis is run to balance the skew across these macros, according the information gathered in the first step. This approach is illustrated in Fig. 5. The resulting 3D clock tree has a 770ps delay and a 91.2ps max skew.

B. Evaluation

1) The Prototype Chip: Although 3D arithmetic units show advantages in performance and power, as will be seen shortly, the chip we designed is only for prototyping idea and design flow, but not for performance evaluation. As mentioned above, the chip is pad limited, and hence the advantage of 3DIC to reduce interconnect length is largely compensated. Our chip has a size of 1.3mm$\times$1.3mm, containing a 36-bit 3D KS adder, a 32$\times$32 3D Wallace Tree multiplier with the architectures described in Sections III and IV. We constrain the chip during the synthesis and place-and-route phase so that it is expected to run at 200Mhz according to post place-and-route result.

2) Performance Comparison: We conduct experiment to compare the performance of standalone 3D and 2D arithmetic units. Specifically, we also lay-out a 2D KS adder and a 2D Wallace Tree multipler, and use the post place-and-route result to study their performance.

The results are generalized in Table II. The first six rows show the comparison between the 3D and 2D KS adder with three different widths. It is clear that the 3D KS adder outperforms the 2D adders for all three widths, with the delay improvement ranging from 10.6% to 34.3% when the width increases from 12-bit to 72-bit. The energy reduction is even more significant, with the improvements ranging from 11.0% to 46.1%. Hence we see that the 3D KS adder not only outperforms the 2D one in both performance and power, but is also more scalable. For the 3D multiplier, there is also a performance improvement of 14.4%, and an energy improvement of 6.8%, as can be seen from the last two rows in Table II. Due to time limitation, we have not done scalability study on the 3D multiplier. However, it is expected that the scalability of the 3D multiplier is also better than the 2D one.

VI. Conclusion

In this paper we describe the design and the implementation of two 3D arithmetic units, a 3D Kogge-Stone adder and a 3D Wallace Tree multiplier. The adder uses a bit-wise partitioning scheme; the multiplier are partitioned along the level boundaries of the last two levels. Taking advantage of the additional interconnect and device capacities provided by 3DIC, we show that there are 10.6% to 34.3% and 14.4% performance improvements for the adder and the multiplier compared to their 2D counterparts respectively. The energy improvements are 11.0% to 46.1% for the adder and 6.8% for the multiplier. The prototype chip that contains both arithmetic units consumes 30mW power and has a size of 1.3mm$\times$1.3mm are being fabricated by MIT Lincoln Laboratory's 3D-FDSOI process.

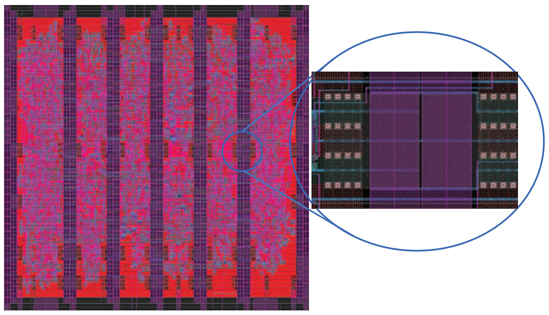

Fig. 6. The topmost tier of the prototype chip. The inset in the figure shows the power TSVs. For clarity, pads are neglected in this figure.

TABLE II
Delay and Power Comparison of 2D and 3D Arithmetic Units

	2D		3D			
	Delay (ps)	Energy (pJ)	Delay (ps)	Delay Reduction (%)	Energy (pJ)	Power Reduction (%)
KS Adder						
12-bit	470	0.860	420	10.6%	0.765	11.0%
36-bit	830	3.86	660	20.4%	3.03	21.5%
72-bit	1280	9.95	840	34.3%	5.63	46.1%
Wallace Tree Multiplier						
32$\times$32	2852	147	2440	14.4%	137	6.8%

References

[1] M.M. Ziegler and M.R. Stan, A unified design space for regular parallel prefix adders, *Design, Automation and Test in Europe Conference and Exhibition, 2004. Proceedings*, Vol.2, pp.1386–1387, 2004.

[2] Z. Huang and M.D. Ercegovac, Effect of wire delay on the design of prefix adders in deep-submicron technology, *Signals, Systems and Computers, Asilomar Conference on*, Vol.2, pp.1713–1717, 2000.

[3] F. Li, C. Nicopoulos, T. Richardson, Y. Xie, V. Narayanan, and M. Kandemir, Design and management of 3D chip multiprocessors using network-in-memory, *Computer Architecture, International Symposium on*, pp. 130–141, 2006.

[4] Y.-F. Tsai, F. Wang, Y. Xie, N. Vijaykrishnan, and M.J. Irwin. Design space exploration for 3-D cache. *Very Large Scale Integration Systems, IEEE Transactions on*, Iss.16, Vol.4, pp.444–455, 2008.

[5] K. Puttaswamy and G.H. Loh, Implementing register files for high-performance microprocessors in a die-stacked (3D) technology, *Emerging VLSI Technologies and Architectures, IEEE Computer Society Annual Symposium on*, 2006.

[6] B. Vaidyanathan, W.-L. Hung, F. Wang, Y. Xie, V. Narayanan, and M.J. Irwin, Architecting microprocessor components in 3D design space, *VLSI Design, International Conference on*, pp.103–108, 2007.

[7] K. Puttaswamy and G.H. Loh, Scalability of 3d-integrated arithmetic units in high-performance microprocessors, *Design Automation Conference*, pp.622–625, 2007.

[8] P.M. Kogge and H.S. Stone, A parallel algorithm for the efficient solution of a general class of recurrence equations, *Computers, IEEE Transactions on*, Vol.22, pp.786–793, 1973.

[9] K.F. Pang, H.-W. Soong, R. Sexton, and P.-H. Ang, Generation of high speed CMOS multiplier-accumulators, *Electronic Devices Meeting, 1963 Internationl*, pp.217–220, 1988.

[10] MIT Lincoln Lab, *MITLL Low Power FDSOI CMOS Process Design Guide*, 6th Edition, 2008.

[11] H. Hao, *Design and verification methodology for complex three-dimensional digital integrated circuit*, PhD thesis, North Carolina State University, 2006.