**Framework for performing rapid evaluation of 3D SoCs**

D. Diamantopoulos, K. Siozios and D. Soudris

Integrating more functionality in a smaller form factor with lower power consumption pushes traditional semiconductor technology scaling to its limits. Three-dimensional (3D) chip stacking is touted as the silver bullet technology that can keep Moore’s momentum and fuel the next wave of consumer electronic products. Introduced is a framework that enables rapid evaluation of 3D SoCs with existing physical design tools.

**Introduction:** In recent years, the 3D IC has attracted increasing attention. Along with technology updates, there are several published works dealing with the 3D physical design problem. Among others, tools for partitioning, floor-plan, placement and routing for 3D architectures, have been proposed. These approaches are based almost exclusively on academic tools. On the other hand, the only known commercial framework for supporting the design of 3D SoCs is provided by R3Logic Corp. [1]. In this Letter we introduce a novel framework for supporting rapid evaluation of 3D SoCs with the usage of existing CAD tools. Such a framework is crucial even before physical design tools for the 3D domain become commercially available, since it provides a good estimation of the potential benefits from designing 3D chips.

**Proposed framework:** This Section introduces the proposed framework for performing rapid evaluation of 3D SoCs. This framework, depicted in Fig. 1, consists of three modular steps in order to enable interaction with tools from similar and/or complementary flows. More specifically, the steps of our framework are summarised as follows:

- **Pre-processing step:** Verification of functional integrity for the design and extraction of its XML description.
- **3D stack generation:** Generation of the 3D stack and determination of the communication (routing paths) among layers.
- **3D system prototyping:** Physical implementation of 3D SoC and evaluation of the derived solution.

**Fig. 1 Proposed framework for supporting rapid evaluation of 3D SoCs**

Initially, the architecture’s HDL description (i.e. VHDL, Verilog) is simulated under various parameters and constraints (e.g. clock period, on-chip memories organisation) in order to verify the system’s functionality. For this purpose, we employ the Cadence NC-sim simulator. Then, we determine the desired hierarchy for target 3D architecture. Our framework can handle different levels of hierarchy. More specifically, a block-based system’s description leads to a coarse-grain solution, whereas a gate-level netlist comes with a finer system implementation. In other words, the fine-grain approach imposes the highest performance enhancement for the 3D architecture, but it also introduces the maximum computational complexity for performing architecture-level exploration. For the scope of this Letter, we choose (without affecting the generality of the proposed framework) to maintain the system’s hierarchy among heterogeneous modules (e.g. logic, memory), while each module is being flattened in order to maximise the performance enhancement.

After defining the SoC’s hierarchy, the HDL description is synthesised with Synopsys Design Compiler. As long as design constraints (e.g. timing slacks, DRCs etc.) are met, the output from synthesis is translated to an equivalent XML description, which corresponds to the system’s hypergraph representation. This task is software supported by our newly publicly available tool, named Net2XML. The derived XML description is fed as input to the second step of our proposed framework, which deals with the 3D stack generation under the selected design constraints.

The second step involves the application’s partitioning into a number of subsets, each of which is assigned to a different layer of the 3D architecture. During this task, both fabrication and cost parameters are taken into consideration. More specifically, for a given layer, only technology compatible components can be assigned, while the layers have to exhibit sufficient area utilisation. Then, the layers are appropriately ordered to maximise the performance of the derived 3D stack. This is feasible by assigning to adjacent spatial locations over the z-axis layers with incompatible interlayer signal activity. The application’s partitioning and layer ordering is software-supported by our previously published TABU algorithm [2]. Rather than similar approaches, which mainly perform mincut partitioning [3], our solution provides additional flexibility, since it is aware also of the selected bonding technology (e.g. TSV, Face-to-Face etc), the desired density of TSVs (per layer) and the shape of the 3D stack (e.g. cube, pyramid etc.). Finally, the derived solutions are evaluated with models for wire-length [4], delay [5] and power consumption [6].

The output from the second step provides sufficient information about the application’s functionality assigned to each layer, as well as the required connectivity among layers. This information is appropriately handled by our new tool, named XML2Net, in order to attach an array of TSVs to every bus that connects the architecture’s components assigned to different layers. Note that whenever a bus needs to be routed from layer i to layer j, the silicon area that equals the area occupied by the TSV array has to be reserved in both layers (we have to preserve that no block will be assigned to the TSV’s landing area) [7]. Even though our framework can also handle distinct TSVs, throughout this study we select to employ arrays of TSVs because they introduce fewer constraints to the routing algorithm [7]. Then, pairs of TSV array and the block that corresponds to its landing area are connected through special purpose routing paths, named TSV networks.

The last step in our framework deals with the system prototyping. More specifically, during this step we perform floor-planning, power and ground network generation, placement of physical library cells, clock tree synthesis and global/detail signal routing with the Cadence SoC Encounter. Since the Cadence tools do not support 3D architectures, we have to make them aware of the additional flexibility imposed by the third dimension through appropriate design encoding. For this purpose we introduce:

- **Virtual layers:** Our framework assumes that the target architecture incorporates a number of virtual layers, each of which contains hardware resources assigned to different physical layers of the 3D SoC.
- **TSV networks:** These networks represent routing paths that provide signal connectivity between a TSV array and its corresponding landing area (assigned to adjacent virtual layers). Note that during physical implementation, our framework preserves that these two architectural entities are aligned over the z-axis (have the same relative (x, y) coordinates at virtual layers). The TSV networks are actually implemented through additional metal layers inserted into the technology library file, while their total resistance (R), capacitance (C) and inductance (L) values per unit length are automatically annotated to represent the TSV’s LRC parameters [8, 9].

The performance of the derived 3D physical prototype is evaluated by the Cadence Static Timing Analysis Engine, while for the sake of completeness this analysis is performed both in advanced, as well as after clock tree synthesis and the architecture’s detailed routing. Then, we verify the functional integrity of the physical design by applying a post-layout simulation with the Cadence Incisive Simulator. For this purpose we extract the delay for all the architecture’s routing paths in SDF format (standard delay format) and then our framework automatically annotates the delay values for the TSV networks. For the scope of this Letter, the electrical characteristics of TSV networks are retrieved from models published in [8, 9]. The evaluation of derived 3D SoCs can also be performed in terms of power consumption by applying a post-layout analysis with the Synopsys PrimeTime PX tool. The inputs to this analysis are the trace file that contains signal activities in VCD (value change dump) format, as well as the annotated SPEF (standard parasitic exchange format) file with extracted parasitic values for all the design’s resources (logic and interconnect).

**Experimental results:** This Section shows how it is possible to employ the proposed framework for designing a 3D instantiation (with two
The synthesis of the LEON3 processor is performed with Synopsys Design Compiler at 130nm CMOS technology under a timing constraint of 4.35ns (or 230MHz). The derived netlist consists of 38988 standard cells, 42626 nets, and 110 I/O ports. Fig. 3 gives the output from floor-planning, assuming a 3D device consisted of two layers. In this Figure, red and green colour dots (see online version) denote arrays of TSV and their landing blocks assigned to virtual layer1 and layer2, respectively, whereas the TSV networks are depicted with blue colour lines. Similarly, red and green colour lines correspond to intralayer connections among arrays of TSV and the rest of the hardware components found in virtual layer1 and layer2, respectively.

![Fig. 3 Example of designing 3D instantiation of LEON3 processor](image)

Conclusion: A novel framework for supporting the rapid evaluation of 3D SoCs is introduced. For the scope of this Letter, the proposed methodology was applied to the design of a 3D instantiation of the LEON3 processor under low-power constraints. Experimental results with various DSP kernels prove the effectiveness of the proposed solution, since it leads to average power savings of 20% without any performance degradation.

© The Institution of Engineering and Technology 2012
19 April 2012
doi: 10.1049/el.2012.1321
One or more of the Figures in this Letter are available in colour online.

D. Diamantopoulos, K. Siozios and D. Soudris (School of Electrical and Computer Engineering, National Technical University of Athens, 9 Heron Polytechneiou, Zographou, Athens 15780, Greece)

E-mail: ksiop@microlab.ntua.gr

References
1 http://www.r3logic.com
5 Okamoto, T., and Cong, J.: ‘Buffered Steiner tree construction with wire sizing for interconnect layout optimization’. Int. Conf. on Computer-Aided Design (ICCAD), San Jose, CA, USA, 1996, pp. 44–49
8 http://3d-performance.lancs.ac.uk/