A Spiffy tool for the simultaneous placement and global routing for three-dimensional field-programmable gate arrays. *

John Karro and James P. Cohoon
Department of Computer Science, University of Virginia
Charlottesville, VA 22903

Abstract

FPGAs are a useful and flexible alternative to custom design chips, but can suffer from severe interconnection delay. The 3D-FPGA is an alternative to the two-dimensional architecture that has been proposed to reduce these delay problems [2]. Here we present Spiffy – the first tool specifically designed for the placement and global routing of 3D-FPGAs. Spiffy produces some of the best results in the literature, and using Spiffy we can show that when mapped to the 3D-FPGA architecture, circuits tend to have considerably shorter net-length, making this new chip an improvement over the standard architecture.

1 Introduction

Field-Programmable Gate Arrays (FPGAs) are reprogrammable chips that can be used to implement arbitrary logic functions, leading to a far more efficient design cycle than that of custom design chips. However, this flexibility often comes with a substantial performance cost. Recently, the concept of a *three-dimensional* (3D) FPGA was proposed [2] with the idea of reducing this cost and making programmable gate arrays a more useful technology. In [1], the first placement statistics were given for any 3D benchmarks, but these were only crude estimates taken from standard 2D algorithms quickly adopted for the 3D architecture.

We have developed Spiffy: a new tool specifically designed for the simultaneous placement and global routing of 3D-FPGAs. The major contributions of this tool are:

- Spiffy is the first tool to produce complete, competitive circuit mappings for the new architecture.
- The quality of solutions produced by Spiffy for the 2D-FPGAs surpasses that of any tool in the literature.
- Using Spiffy, we were able to experimentally verify our claim that 3D-FPGAs are in fact superior to the standard architecture – finding that the addition of only three extra levels led to an average improvement of over 10% in terms of both net-length and congestion.

*John Karro is supported by a grant from the Virginia Aerospace Consortium Fellowship Program. A full version of this paper may be found at http://www.cs.virginia.edu/vlsicad/gl-complete.pdf

2 3D-FPGAs

Our 3D-FPGAs will be constructed by placing a number of 2D-FPGAs in layers, providing vertical interconnections between adjacent switch-blocks through the use of solder bumps. Hence the switch-block configuration is generalized to the third dimension, while the structure of the individual logic blocks remains unchanged.

We expect 3D-FPGAs to prove superior over the standard architecture in two categories: performance and logic density. In terms of performance, we get a win because of the decreasing average path length between points. In terms of logic density, the increased number of paths between any two points allow us to decrease the maximum number of wires required in any one channel, hence allowing us to use more of an individual layer's area for logic.

The idea of three dimensional chip architectures is certainly not new; MVM-Vs have been studied in detail, and other schemes for 3D architectures have also been proposed. Since our initial proposal for a 3D FPGA architecture [1, 2], others have since been proposed [5, 6].

Certainly there will be problems with such an architecture, such as heat dissipation, and the difficulties associated with efficient construction. However, similar problems have been encountered and explored for other 3D packaging technologies. We believe that the 3D-FPGA architecture will not pose any additional problems.

3 Spiffy

Spiffy is the first tool created for the physical design of 3D-FPGAs, using a divide and conquer approach to perform both placement and global routing *simultaneously*. It

0-7695-0104-4/99 $10.00 © 1999 IEEE

begins by dividing the chip into a grid of some specified size ($2 \times 2 \times 2$ in our experiments), and partitioning the logic blocks between the grid blocks. When we consider an individual net, we would like to place the logic blocks of the net entirely within one grid block if possible, and failing that, we would like to keep them in neighboring grid blocks. This preference is heuristically achieved by considering the *thumbnail* for each net: the minimum steiner tree connecting the grid blocks the net occupies. By partitioning the logic blocks in such a way as to minimize the total length of the thumbnails, we tend to achieve our objective of keeping the logic blocks of a given net close together.

After this partitioning has been performed, Spiffy considers those nets that have been assigned to multiple partitions, and assigns them the exact spots at which they will cross between grid blocks. It can then consider each grid block recursively. Once we have placed and routed the logic blocks within the grid block, we need only connect each net with its "entry points" to the grid block, and are left with a full placement and global routing upon termination.

The algorithm iterates between placement and global routing decisions, allowing each to influence the other. By allowing this interaction, and thus allowing the placement phase to take routing issues into consideration, we believe we will achieve better results than the traditional model of fixing a placement and then computing a routing for it. After Spiffy produces the placement and global route, the graph-based tool of Alexander et al [3] determines the detailed routing.

4 Results

To verify that quality of our tool, we first compared it against Mondrian, a leading tool for the placement and global routing of FPGAs[4]. Using a series of standard benchmarks for 2D architectures, we found our tool produced mappings of better net length (an average improvement of 20%) in considerably better time (running about 83% faster).

The true importance of Spiffy is its ability to map 3D-FPGAs, thus allowing us to determine whether this new architecture actually does bring about improvements over the standard 2D-FPGA. We mapped each benchmark to four different 3D architectures, ranging from one to four layers. In Table 1 we see the results of these tests, and a clear improvement in each category as the number of layers is increased.

	Two Levels	Three Levels	Four Levels
Congestion	3.97%	6.09%	11.9%
Net Length	3.27%	6.95%	10.7%
Run Time	23.8%	23.6%	20.0%

Table 1: The average percentage gains of the 3D architecture over the 2D architecture.

5 Conclusion

In creating Spiffy, we achieved three goals. First, we created the first tool suite to produce full circuit mappings for the proposed 3D-architecture. Second, we created a tool which performs placement and global routing simultaneously, and produces better results for the standard architecture than the state-of-the-art in the literature. Third, we were able to use this tool to produce the first actual circuit mappings for the 3D architecture, and hence empirically verify that the extra layers would actually lead to more efficient mappings.

At this point, we have a number of directions we are taking with Spiffy and the concept of the 3D-FPGA. With regard to the tool, we are looking at ways to improve certain aspects of algorithm, as well as using a detailed router currently being worked on by McCullouch at the University of Virginia, or possibly incorporating a detailed routing method into the simultaneous paradigm.

In terms of the architecture, there are a number of questions we are investigating concerning its ideal structure. Such as the necessity of including all possible vertical interconnections between layers, and ideal ways to structure the interconnection block to balance flexibility against construction complexity and size.

References

[1] Alexander, M.J., Cohoon, J.P., Colflesh, J.L., Karro, J., Peters, E.L., and Robins, G. Physical layout for three-dimensional fpgas. *Fifth ACM/SIGDA Physical Design Workshop*, pages 142–149, April 1996.

[2] Alexander, M.J., Cohoon, J.P., Colflesh, J.L., Karro, J., and Robins, G. Three-dimensional field-programmable gate arrays. *IEEE ASIC Conference*, pages 253–256, September 1995.

[3] Alexander, M.J., Cohoon, J.P., Ganley, J.L., and Robins, G. Placement and routing for high-performance fpga layout. *VLSI Design: International Journal of Custom-Chip Design, Simulation, and Testing*, 7:97–110, January 1998.

[4] J. Ganley. *Geometric Interconnections and Placement Algorithms*. PhD thesis, University of Virginia, Department of Computer Science, Charlottesville, VA, 1995.

[5] Lesser, M., Meleis, W., and Vai, M. *Rothko: 3-Dimensional FPGA's*. http://www.ece.neu.edu/research/rothko/.

[6] Seed, Luke, Meacham, R., Zawads, A., and Thorne, P. *TriMorph: 3D Computational Structures*. http://www.shef.ac.uk/ uni/academic/D-H/eee/esg/research/trimorph.html.