3D FPGA versus Multiple FPGA System: Enhanced Parallelism in Smaller Area

Krishna Chaitanya Nunna
Department of Advanced Informatics
Kyushu University
Fukuoka, Japan
krishna@soc.ait.kyushu-u.ac.jp

Farhad Mehdipour
E-JUST Center
Kyushu University
Fukuoka, Japan
farhad@ejust.kyushu-u.ac.jp

Kazuaki Murakami
Department of Advanced Informatics
Kyushu University
Fukuoka, Japan
murakami@ait.kyushu-u.ac.jp

Abstract

Handling large amounts of data is being limited by bandwidth constraint between processors components and their memory counterparts. Three-dimensional integration (3D) is providing possible solution to handle such critical applications. Especially for running larger designs when implementing on multi-FPGA platform, which can produce huge amount of fine-grain parallelism, for satisfying the speed and reliability needs 3D FPGA can be considered as a close candidate to choose. In this paper we tried to show the benefits of running larger designs on 3D FPGA compared to running on multi-FPGA systems using benchmark simulation. Results showed that a TSV-based 3D FPGA achieved better performance and area results.

Keywords: 3D FPGA, TSV, Multi-FPGA System

1 Introduction

One of the vital applications of field-programmable gate arrays (FPGAs) is rapid application specific integrated circuit (ASIC) prototyping. An FPGA is an array of programmable logic blocks (LBs), configurable interconnect, and I/O blocks which can be user-configured to implement complex digital circuits. This highly reprogrammable structure enables FPGAs to exploit parallelism at different granularities. ASIC designers are turning to FPGA for prototyping their designs to take the advances of their low cost and fast prototyping. Especially for simulations and design verification of larger circuits multiple FPGA platforms became common. A multi-FPGA system, as shown in Fig. 1(a) contains multiple reprogrammable devices on a PCB. A system of FPGAs can be seen as a computing substrate with somewhat different properties than standard microprocessors. It provides a huge amount of fine-grain parallelism. For example, as presented by Nechma (2012) presented a parallel sparse matrix solution for direct circuit simulation on multi-FPGA system. The re-programmability of the FPGAs allows one to download algorithms onto the FPGAs, and change these algorithms just as PC can change programs. When a circuit is needed to implement on to a multi-FPGA platform, it is partitioned into number of parts equal to number of FPGA chips on the system. Then these partitions will be mapped on those FPGAs separately. The communication between the chips is performed by inter-chip connects. Note that the bus shown in Fig 1(a) is an example representation of such communication. In real, the way the FPGAs connected depends totally on type of chip package used. For example SPP-2K reconfigurable platform shown in HitechGlobal (2012) using nine Xilinx Virtex-4 FPGAs with external interface for use with a PC and general purpose CPU boards can be used for ASIC and IP development. Even though multi-FPGA platform is able to handle larger designs, due to their off-chip communication strategy the communication between the chips is limited by the bandwidth constraints from the interface side. With the constraint as limited number of I/O pads on a FPGA, it can also be necessary to multiplex the FPGA-to-FPGA signals, which further reduces the performance. One possible solution to achieve higher speed at the same level of circuit complexity is three-dimensional (3D) integration of FPGAs.

3D FPGA is one of the promising innovations which can provide benefits like increasing transistor density, reduced form factor, heterogeneous architectures and improvement in delay by significantly reducing the wire lengths of integrated circuits (J. Alexander *et al*. 1996). It is a multi-layer device stacked using through-silicon via (TSV) technology. That means the communication between the layers is done by using TSVs as shown in Fig. 1(b). The communication between the layers in 3D FPGA is on-chip, and hence it is quite obvious from the implementation perspective to expect higher speed compared to the off-chip communication platform. An interesting work is presented by Zied *et al*. (2012) on performance comparison between multi-FPGA system and a Xilinx 3D FPGA (rather 2.5D). In their experiments they compared two ways of implementing specific ASIC designs. The first used the DN2076K10 DINI board (DiniGroup 2013), which contains six FPGA Virtex-6 LX760 FPGAs (each one has 1 die/chip), with a total logic capacity of around 2.8 million LUTs. The second used a board (XCVirtex7 2013) designed with two Virtex-7 2000T FPGAs (each one has 4 dies/chip), with a total logic capacity of around 4 million LUTs and a maximum logic utilization of 70%. Virtex-6 is a 2D FPGA where as Virtex-7 is 2.5D/3D FPGA in which multiple FPGA dies in each package are linked using stacked silicon interconnect layer. They have showed that the clock frequency is increased in the case of 3D FPGA

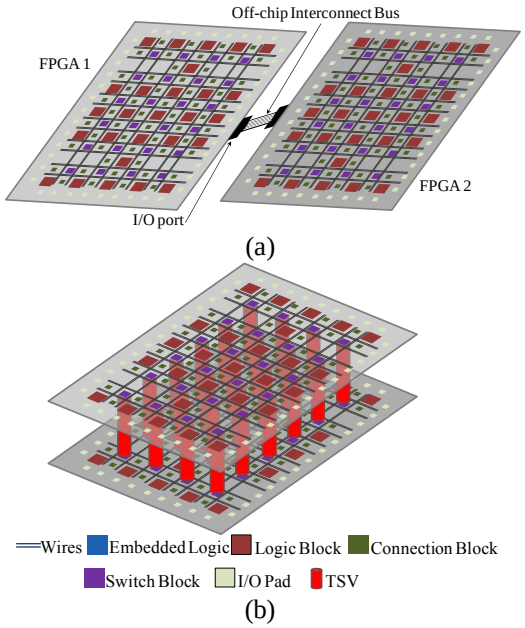

Fig. 1 (a) Multi-FPGA platform using off-chip interconnect bus (b) TSV-based 3D FPGA

against multi-FPGA platform. Even though they tried to show the benefits pertained to 3D FPGA, Virtex-7 is not a fully 3D rather it is a 2.5D FPGA having almost same footprint area as multi-FPGA system and hence more than half footprint reduction in real 3D FPGA is not visible in their experimental results. For example, Davis, *et al.* (2001) reported a 3x reduction in total silicon area and a 12x reduction in chip footprint for a monolithic 3D IC with 4 device layers when compared to a 2D IC. The reduction in wire lengths enabled a size decrease of the logic gate drivers for these wires, which reduced the distance between logic gates further, causing a significant reduction in silicon area. Earlier works (Swaminathan *et al.* 2012, Kim *et al.* 1995 and Roy *et al.* 1995) introduced a novel partitioning techniques for multi-FPGA systems. Most of the earlier research work, except Zied *et al.* (2012), concentrated on benefits attained by multi-FPGA and different techniques to improve the performance metrics for the same.

This research work presents emphatic analysis on benefits attained by TSV-based stacked 3D FPGA against the multi-FPGA platform specifically for rapid prototyping. The key contributions are: performance comparison against 14 largest MCNC benchmarks implemented on to 2D and 3D FPGAs; area comparison. For a fair comparison between multi-FPGA system and 3D FPGA, real time board-based experimental environment is not available either in academia or domestic market. It is quite reasonable to show the benefits of 3D FPGA as an alternative through potential CAD tool environment which can cope with design and development of 3D integration. Up to our knowledge our work presented in this paper is first of its kind.

The remainder of the paper is followed by background in section 2, methodology for proposed work in section 3 and finally concluded along with results in section 4.

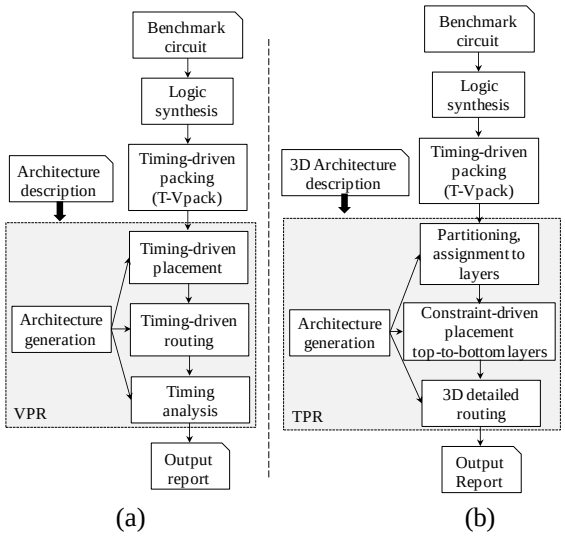

Fig. 2 (a) VPR CAD flow for 2D FPGAs (Betz *et al.* 1997) (b) TPR CAD flow for 3D FPGAs (Ababei *et al.* 2006)

2 Background

As a part of the analysis on MCNC benchmark circuits, we are using Versatile Place and Route (VPR) (Betz *et al.* 1997) and Three-dimensional Place and Route (TPR) (Ababei *et al.* 2006) which are widely used research and academia placement and routing tools for 2D and 3D FPGAs respectively. VPR is an open-source place and route tool intended to research in CAD and architecture for island-style FPGA architectures. These FPGAs contain I/O blocks and logic blocks surrounded by programmable routing. As the logic blocks are all assumed to be identical, a single logic block and its adjacent routing can be combined to form a tile that can be replicated to create the full FPGA. The experimental process used in VPR-based CAD and architecture research is illustrated in Fig. 2(a). This flow takes an input circuit at the logic level and FPGA architecture along with necessary design specifications and implements the circuit on the specified FPGA. The output generates all the required parameters starting from the number of interconnects occupied by the nets in circuit to the critical path delay of the implemented circuit. It can also model an approximation of the area taken up by that circuit in that FPGA. In the first step of the flow, the circuit is synthesized and technology mapped. The output after technology mapping must then be packed into the logic clusters available on the FPGA. Once packing is complete, VPR is then used to perform placement and routing for the circuit. Finally, timing analysis is completed to determine the performance of the circuit on the FPGA.

TPR is a partitioning-based placement and routing toolset. Its purpose is to serve the research community in predicting and exploring potential gains that the 3-D technologies for FPGAs have to offer (similar to the role VPR played in the development of FPGA physical-design algorithms). It can be used as a platform for development and implementation of new ideas in placement and routing for 3-D FPGAs. The philosophy of TPR closely follows that of its 2-D counterpart, VPR. The flow of the TPR

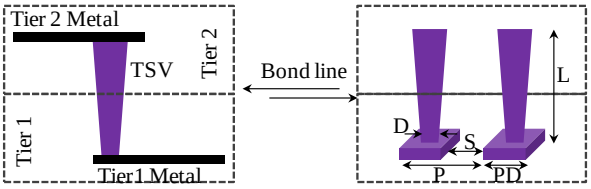

Fig. 3 (a) Typical TSV representation fabricated on a 2-layer stacked IC (b) Typical TSV parameters representation (Papanikolaou *et al.* 2011)

Parameter	2008	2009	2010	2011	2012	2013	2014	2015
TSV diameter, D (um)	1.6	1.5	1.4	1.3	1.3	1.2	1.2	1
TSV pitch, P (um)	5.6	5.5	4.4	3.8	3.8	2.7	2.6	2.5
Pad spacing, S (um)	1	1	1	0.5	0.5	0.5	0.5	0.5
Pad diameter, PD (um)	4.6	4.5	3.4	3.3	3.3	2.2	2.1	2

Table 1 TSV parameters reported by 2008 ITRS data

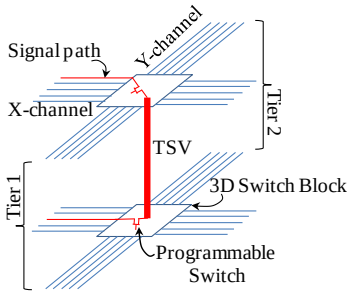

Fig. 4 Net connected between two layers using TSV in association with programmable switch

based placement and routing CAD tool is shown in Fig. 2(b). The design flow starts with a technology mapped netlist in .blif format. The .blif netlist is converted to a netlist composed of more complex logic blocks. The .net netlist as well as the architecture description file are inputs to the placement algorithm. The placement algorithm first partitions the circuit into a number of balanced partitions equal to the number of layers for 3-D integration. The goal of this first min-cut partitioning is to minimize the connections between layers, which translates into minimum number of vertical (i.e., interlayer) wires. After dividing the netlist into layers, TPR continues with the placement of each layer in a top-down fashion.

The total interconnect length of 3D chip is expected to be shorter than that of 2D chip so that the footprint area of 3D IC becomes smaller than that of 2D IC. Wirelength reduction in 3D ICs has been demonstrated in earlier studies including real chip design (Joyner *et al.* 2000, Kim *et al.* 2009 and Thorolfsson *et al.* 2009). Even though the total wirelength is reduced, the delay associated with the TSV is affected by its large capacitance which one should consider in delay calculations. For example, 20fF TSV capacitance is comparable to the capacitance of a 120µm-long intermediate-layer wire (e.g. M4 to M6) in 45nm technology (Kim *et al.* 2010). In addition, TSV RC has different characteristics from wire RC. Therefore, it is shown that if we use wire RC models for TSV parasitics, there will be non-negligible amount of errors in computation and prediction of TSV-related delay and power consumption in 3D chips. A TSV connected between two tiers in a 2-layer 3D device is shown in Fig. 3(a). The width or diameter of a TSV is larger than a normal metal interconnect which we can find on a single die. Typical TSV parameters are represented as shown in Fig. (3b) and their predicted values by ITRS are shown in Table 1. The delay pertained to TSV can be calculated using Elmore delay model as:

$$T_{TSV} = 0.5 * R_{TSV} * C_{TSV} * L_{TSV} + R_{swt} * C_{metal} + T_{swt} \quad (1)$$

The above equation represents the delay associated with TSV in combination with the switch connected to it as shown in Fig. 4. In Eq. (1), R_{TSV} is resistance of TSV, C_{TSV} is capacitance of TSV, L_{TSV} is length of TSV, R_{swt} is the resistance of switch, C_{metal} is capacitance of the metal, T_{swt} is switch delay associated with TSV.

3 Methodology

The proposed methodology is specific to logic simulation and uses both VPR and TPR as and when required. Here the partitioning technique is the significant step for knowing the number of connects between layers (or chips) by partitioning the input circuit equal to the number of layers (or chips), which we assumed as two in our experiments. We used ParKway presented by Trifunovic *et al.* (2004) for partitioning the circuit. The partitioning tool used by TPR is hMetis presented by Karypis *et al.* (1997) which is a closed-source tool. In order to have better control on the partition process we have replaced hMetis with ParKway which is an open-source tool. Since the Parkway is a heuristic based approach, we performed partitioned more than two times and the best result among these is selected.

As a part of the off-chip communication delay calculation, we will extract the required information such as cut size from the partitioned circuit. By utilizing the extracted parameters, we then calculate the off-chip interconnect delay for every signal that traverse between two FPGAs same as given by Swaminathan *et al.* (2012).

$$T_{off-chip} = \left\lceil \frac{N_{cuts}}{N_{pin_{avail}}} \right\rceil * T_{clk} + (T_{wire\_max} * d * N_{arch}) \quad (2)$$

In Eq. (2) N_{cuts} is the number of cuts between two partitions, N_{pin_avail} is the number of pins available to each partition, T_{clk} is the critical path delay obtained when the input circuit is actually implemented using VPR on a single FPGA (to decide a maximum clock frequency that a given circuit can operate), T_{wire_max} is the maximum wire delay that can be obtained for a given FPGA array, d is the maximum distance that the signal can travel from a certain point on FPGA array to the edge of the chip surface (to connect IO pad) and N_{arch} is the number of FPGAs used in the multi-FPGA system which 2 in our experiments.

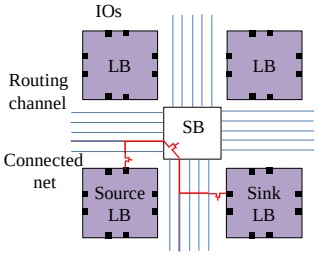

Fig. 5 LB-LB connection for delay calculation

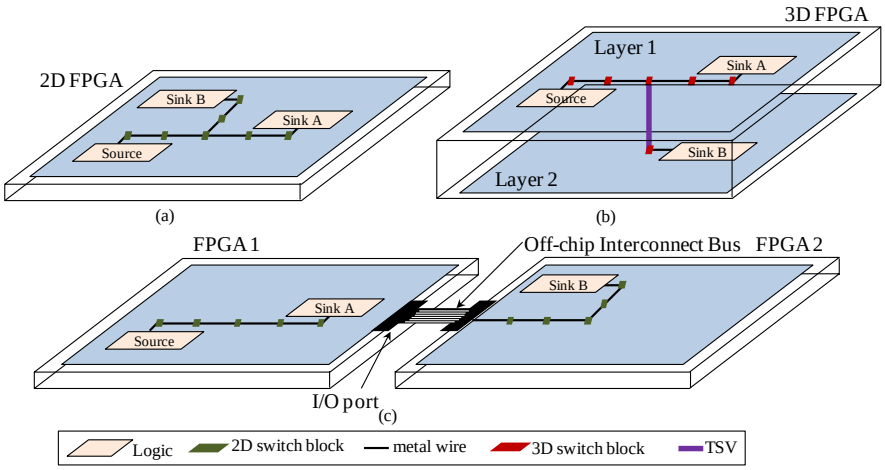

Fig. 6 (a) Circuit placed on 2D FPGA (b) Partitioned circuit placed on a TSV based 2-layer stacked 3D FPGA (c) Partitioned circuit placed on multi-FPGA system with 2 FPGAs connected using an off-chip interconnected bus.

Here T_{wire_max} can be obtained by finding maximum wirelength across a given FPGA area as a multiplication of number of LBs in a row of LBs on a square shaped FPGA with the delay between any two LBs. the delay between any two LBs can be calculated as shown in Fig. 5. The connected path shown in red colour represented the communication between any two neighbouring LBs. It is clear from the figure that the delay calculated between these neighbours involves 3 switch delay associated with SB and two IO connectors to the routing channel.

We are representing the real time multi-FPGA platform using TPR to calculate the delay for all benchmark circuits by replacing the TSV delay with off-chip interconnect bus delay calculated in Eq. (2). Measuring the critical path in 3D FPGA is easy as it can be directly calculated by TPR itself. For multi-FPGA case, we are using the same delay information obtained by TPR with the following approach. We first separate the total delay into 3 parts: layer 1delay, TSV delay, layer 2 delay as shown in Eq. 3.

$$T_{total} = T_{layer1} + T_{TSV} + T_{layer2} \quad (3)$$

$$T_{total} = T_{layer1} + T_{off-chip} + T_{layer2} \quad (4)$$

These separate delay variables in Eq. (3) can be obtained by parsing the TPR source code and keeping the watch-dogs to read the information at the necessary points. Now the TSV delay is replaced by the off-chip interconnect delay calculated by using Eq. (2) as shown in Eq. (4). In TPR case, the goal of placement and routing is to reduce the wirelength as much as they can. That means any connection inside each layer will be made by utilizing the nearest TSV such the logic blocks placed in opposite layers will be connected with shorter wirelength. For multi-FPGA system, the same phenomenon can be applicable. Because for multi-FPGA system, the entire chip will be occupied such that the logic that need to be connected through off-chip bus will be placed near to the I/O pins such the wirelength in each FPGA will be minimized by satisfying the area constraint. Fig. 6 represents the circuit placed and routed on a 2D FPGA, 2-layer 3D FPGA and multi-FPGA system with 2 FPGAs connected using an off-chip interconnected bus. As given in Eq. (2), the off-chip interconnect bus delay depends on the cut size or number of cuts obtained after partitioning the input circuit targeting the minimum cuts between the partitions. The clock is assumed by calculating the critical path delay obtained after doing placement and routing each circuit on a single 2D FPGA using VPR. Both the FPGAs operate at the same clock. Now these values are fed to Eq. (4).

4 Experiment Results

We have performed all experiments on Ubuntu Linux machine. The assumptions that we use for all the simulations are shown in Table 2. As part of the experiments the steps involved are:

- Implement the input circuit on TPR to find the best fit FPGA array size. Thus gives: array size for each layer; number of IOs for each layer; number of inter-communication nets; chip area in terms of wirelength, circuit delay and TSV delay.
- Calculate the off-chip communication delay using the Eq. 2 for the input circuit based on the parameters obtained in step 1.
- Now calculate the delay for the input circuit using Eq. 4 for multi-FPGA system.

4.1 Effective area utilization

Circuit implemented on 3D FPGA requires less footprint area compared when implemented on 2D FPGA. We first implemented all MCNC benchmark circuits on 2-layer 3D FPGA using TPR to get the best fit array size in terms LBs (row x column). For better understanding to the reader we are also showing the best fit FPGA array sizes when implemented on 2D FPGA using VPR. All the 14 MCNC benchmark circuits are implemented on these 2D and 3D FPGAs such that the overall chip utilization will be more than 90%. This is because of the reason that the fundamental application of multi-FPGA is implementing

larger circuits. That means each FPGA in a given multi-FPGA system will occupied almost fully by the input circuit. Hence during our experiments we make ensure to achieve more than 90% area utilization on each layer of the 3D FPGA. For a fair comparison on advantages of 3D FPGA over its counterpart, Table 4 shows the wirelength and critical path delay obtained for 2D and 3D FPGAs. As a target work, the critical path delay obtained only for 3D FPGA is compared against the delay obtained for multi-FPGA system as explained below.

4.2 Performance comparison

The circuit delay involving off-chip communication delay for 14 MCNC benchmark circuits implemented on a multi-FPGA system with 2 FPGAs is shown in Table 5. The clock speed is obtained by running the input circuit for one time on VPR. This will ensure to find the best timing results such that the maximum operating frequency will be decided based on this. Results show that the delay is reduced by more than 50% on average for all the circuits.

4.3 Area comparison

Fig 7 shows the comparison of normalized chip footprint area attained when the circuit implemented on 3D FPGA and multi-FPGA system of 2 FPGAs. Footprint area for 3D FPGA is generated by TPR itself in terms of physical units. For 2D FPGA case, total footprint area equal to 3D FPGA multiplied by a minimum of 2 (due to the active silicon area). In addition to this, area occupied by the off-chip interconnect bus is also added which is not shown in the figure.

RESULT WILL BE ADDED HERE

Table 2 Parameters assumed for experiments

Table 3 FPGA chip area utilization rates for 2D FPGA and a 2-layer 3D FPGA

Table 4 Wirelength achieved for 2D FPGA and a 2-layer 3D FPGA

Table 5 Off-chip interconnect delay for two-FPGA system

5 Conclusion

In this paper, we have investigated the delay characteristics of multi-FPGA system and TSV-based 3D FPGA. We have showed that the delay improvements achieved by 3D FPGA is on par compared to multi-FPGA system majorly due its on-chip communication facility via TSVs. This kind of motivation can add extra potential to the already available feature of fine-grain parallelism of FPGAs which may results in much faster functional modelling. For the experiments we have consider square-shaped TSV with parameters similar to industry standards. We believe that the in-detail modeling of TSV along with its effect on the material around it could help in showing added benefits in terms of the delay numbers. As a part of the future work, heterogeneous FPGA architectures with more than two layers can be considered for targeting design flexibility towards performance improvements.

6 References

Tarek Nechma (2012): Parallel Sparse Matrix Solution for Direct Circuit Simulation on a Multiple FPGA System. Ph.D. thesis. University of Southampton.

HitechGlobal (2012):

http://hitechglobal.com/Boards/MultiFPGA.htm

J. Alexander, J. P. Cohoon, J. L. Colflesh, J. Karro, E. L. Peters, and G. Robins (1996): Placement and Routing for Three-Dimensional FPGAs. *Proc. 4th Canadian Workshop Field-Programmable Devices*, pp. 11–18.

Zied Marrakchi, Ramsis Farhat and Ramine Roane (2012): Improving ASIC prototyping on multiple FPGAs through better Partitioning. *An article in Tech Design Forum*.

DiniGroup (2013):

http://www.dinigroup.com/new/DN2076k10.php

XCVirtex7 (2013): http://www.xilinx.com/products/silicon-devices/fpga/virtex-7/index.htm

J. Davis, et al. (2001): Interconnect limits on gigascale integration (GSI) in the 21st century. *Proc. IEEE*, vol.89, no.3, pp.305-324.

Swaminathan P. S., Lin P.-C.K., and Kathri S.P. (2012): Timing Aware Partitioning for Multi-FPGA Based Logic Simulation Using Top-Down Selective Hierarchy Flattening. *Proc. 30th IEEE International Conference on Computer Design*, pp. 153-158.

C. Kim, H. Shin, and Y. Yu (1995): Performance-driven circuit partitioning for prototyping by using multiple FPGA chips. *Proc. Design Automation Conference*.

K. Roy-Neogi and C. Sechen (1995): Multiple FPGA partitioning with performance optimization. *Proc. Field-Programmable Gate Arrays*, pp. 146 – 152.

V. Betz and J. Rose (1997): VPR: A New Packing, Placement and Routing Tool for FPGA Research. *International Workshop Field Programmable Logic and Applications*, pp. 213–222.

Cristinel Ababei, Hushrav Mogal and Kia Bazargan (2006): Three-Dimensional Place and Route for FPGAs. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD)*, Vol. 25, Issue 6, pp. 1132-1140.

A. Papanikolaou et al. (2011): *Three Dimensional System Integration: IC Stacking Process and Design*, Springer.

J. W. Joyner, P. Zarkesh-Ha, J. A. Davis, and J. D. Meindl (2000): A Three-Dimensional Stochastic Wire-Length Distribution for Variable Separation of Strata. *Proc. IEEE Int. Interconnect Technology Conference*, pp. 126–128.

D. H. Kim, S. Mukhopadhyay, and S. K. Lim (2009): TSV-aware Interconnect Length and Power Prediction for 3D Stacked ICs. *Proc. IEEE Int. Interconnect Technology Conference*, pp. 26–28.

T. Thorolfsson, K. Gonsalves, and P. D. Franzon (2009): Design Automation for a 3DIC FFT Processor for Synthetic Aperture Radar: A Case Study. *Proc. ACM Design Automation Conf.*, pp. 51–56.

D. H. Kim and and S. K. Lim (2010): Through-Silicon-Via-aware Delay and Power Prediction Model for Buffered Interconnects in 3D ICs. *Proc. IEEE Int. workshop on System Level Interconnect Prediction Conference*, pp. 25–32.

Trifunovic. A and Knottenbelt. J (2004): A Parallel Algorithm for Multilevel K-way Hypergraph Partitioning. *Proc. 3rd International Symposium on Algorithms, Models and Tools for Parallel Computing on Heterogeneous Networks*, pp. 114 – 121.

G. Karypis, R. Aggarwal, V. Kumar and S. Shekhar (1997): "Multi-level Hypergraph Partitioning: Applications in VLSI design. *Proc. ACM/IEEE Design Automation Conf. (DAC)*, pp. 526–529.

FreePDK: http://www.eda.ncsu.edu/wiki/FreePDK.