ARCHITECTURE AND EDA FOR 3D FPGAS

HOU JUNSONG

(B. Eng.(Hons.), NUS)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

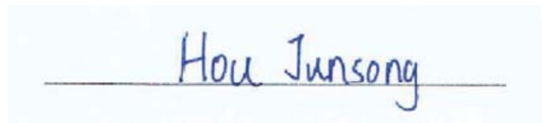
NATIONAL UNIVERSITY OF SINGAPORE

2013

DECLARATION

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has not been submitted for any degree in any university previously.



Hou Junsong

27 Dec 2013

Acknowledgments

I would like to take this opportunity to thank my main supervisor, Dr. Ha Yajun, for his guidance and support. Although I am a part time student, he always encouraged me to discuss my progress and advised me with his professional opinions. Under his guidance, I gradually learned how to pursue my research topic.

Also, I would like to thank my co-supervisor, Dr Liu Xin, for providing me many useful research materials and information. The time that I worked with Dr Liu is relatively short. However, he offered me a lot of information on the 3D IC design to expand my knowledge scope.

Lastly, I would like to thank Dr. Zhou Jun, Zhao Wenfeng, Dr. Yu Heng and Syed Rizwan for their advice and help on my research work.

Contents

Declaration i

Acknowledge ii

Table of Content iii

Summary viii

List of Tables x

List of Figures xi

List of Acronyms xvi

Chapter 1 Introduction 1

1.1 2D FPGA Overview 3

1.2 Overview of 3D FPGA 5

1.3 3D FPGA Design Issues 8

1.4 Our Solutions 9

1.5 Thesis Contributions 11

1.6 Organization 13

Chapter 2 Background and Related Work 14

2.1 2D FPGA Architecture 15

2.1.1 Cluster-based Logic Block 15

2.1.2 Programmable Routing 19

2.2 3D IC Architecture 22

2.3 Electronic Design Automation (EDA) Framework 24

2.3.1 VPR Framework 24

2.3.1.1 VPR Based EDA Flow 25

2.3.1.2 VPR Placement 26

2.3.1.3 VPR Power Model 28

2.3.2 HotSpot Thermal Framework 30

2.4 Related Work 31

2.4.1 3D FPGA Architecture 31

2.4.2 3D FPGA EDA Software Tools 33

2.4.3 Thermal Estimation 35

Chapter 3 3D FPGA Architecture **36**

3.1 Overview of 3D FPGA Architecture 36

3.2 Uni-directional Routing Based 3D FPGA 41

3.2.1 2D Uni-direction Routing Architecture 42

3.2.2 3D-SB-1 and 3D-SB-2 3D Switch Box (SB) 43

3.2.3 The “Uni-Bi” Multiplexer Switch 48

Chapter 4 3D FPGA EDA 52

4.1 General Flow 53

4.2 Uni-directional Placement Algorithm 55

4.2.1 Placement Algorithm for UNI-3D 56

4.2.2 Placement Algorithm UNI-3D with UB Switch 61

4.3 Thermal Placement 63

4.3.1 Physical Model 65

4.3.2 Power Model 72

4.3.3 Thermal Model 76

Chapter 5 Design Space Exploration 78

5.1 Architecture Exploration on Input Connection Box (CB) Flexibility, C_{in} 79

5.2 Placement Algorithm Exploration on The Weight of C_{uni} . . 85

5.3 3D Architecture with UB Switch V.S. 3D Architecture without UB Switch 89

5.4 Thermal Placement Exploration 94

Chapter 6 Conclusion and Future Work **99**

Publicaton **101**

References **102**

Summary

Research on 3D IC design is actively conducted for its high logic density and excellent performance, compared with conventional 2D Integrated Circuit (IC) design. In this study, we explore the architecture and the EDA on 3D Field Programmable Gate Array (FPGA) which is implemented by stacking the conventional 2D FPGAs. Thermal and Through Silicon Via (TSV) reduction are two of the major design challenges we focus on in this thesis.

As the uni-directional based routing architecture becomes dominant, we use the uni-directional based routing architecture as the base to build the 3D FPGA architecture in this study. For the uni-directional routing based 3D architecture, we believe that the fixed direction in the vertical channel may waste the TSV usage. Therefore we put some effort to the routing switch design in order to use TSV more efficiently besides the 3D SB design.

On the other hand, we develop the evaluation software based on VPR 5 at the EDA side. Also we improve the placement algorithm so that fewer number of TSV is required for the 3D FPGA placement and routing. Furthermore, we add the HotSpot thermal model into the software to implement the thermal aware placement to tackle the thermal issue cause by the

high power density of 3D stacking.

Our experiment with 20 MCNC benchmarks shows that our 3D architecture is able to reduce about more than 25% delay and planar channel width compared with the 2D baseline. In addition, our proposed architecture with corresponding placement algorithm is able to use 16% fewer number of TSV in average compared with the state-of-the-art work. As for the thermal issue, our proposed placement algorithm is able to reduce the temperature but the run time needs to be improved.

List of Tables

5.1 **Placement Method Compare** 98

List of Figures

1.1 **FPGA EDA Flow [1]** 4

1.2 **Two Different 3D FPGA Architectures** 6

1.3 **Evolution of 2D SB to 3D SB** 7

2.1 **2D FPGA in Abstract Level [2]** 16

2.2 **Cluster-based Logic Block** 17

2.3 **Basic Logic Element** 18

2.4 **2D FPGA Architecture (Island Style)** 19

2.5 **Internal Population** 20

2.6 **Bi-directional and Uni-directional Wires [3]** 22

2.7 **Three Types of Die Bondings and Two Types of TSVs [4]** 23

2.8 **VPR Flow [5]** 25

2.9 **VPR EDA FLOW [6]** 26

2.10 **Pseudo Code of a Generic SA Based Placer [1]** 27

2.11 **VPR 5 Power Model Software Flow** 29

2.12 **HotSpot RC thermal model [7]** 30

3.1 **3 layer stacked 3D FPGA** 37

3.2 **TSV Layout [8]** 38

3.3 **TSV Cell** 40

3.4 **2D FPGA and 3D FPGA Tile** 41

3.5 **2D Uni-directional Routing Architecture** 43

3.6 **3D-SB-1 SB** 44

3.7 **Output Pin to Vertical Track Connection** 46

3.8 SB Pseudo Code 47

3.9 **Unbalance Assignment to Reduce Vertical Channel Width** 49

3.10 **"Uni-Bi" Switch** 50

3.11 **Schematic of UNI-3D with UB Switch** 51

4.1 **EDA Flow** 54

4.2 **Example of Nets Spanning in the Vertical Direction by Using Cost Equation 4.1** 58

4.3 **Modified TSV Aware Placement Algorithm for Unidirectional Routing Based 3D FPGA** 61

4.4 **Modified EDA Flow for Further TSV Reduction** . . . 64

4.5 **Thermal Profile Calculation Flow** 65

4.6 **Thermal Aware Placement Algorithm (Also Considering TSV Reduction)** 66

4.7 **Thermal Tile and Thermal Area** 68

4.8 **Heat Flow and Conductive Layers** 70

4.9 **Thermal Block** 72

4.10 **Example of Static Power and Short Circuit Power** . . 73

5.1 **Average Planar Channel Width Variation Due to $\mathbf{C_{in}}$** 80

5.2 **Average Routing Area Due to $\mathbf{C_{in}}$** 81

5.3 **Average TSV Used Per Tile Due to $\mathbf{C_{in}}$** 81

5.4 **Average Delay Due to $\mathbf{C_{in}}$** 83

5.5 **RADP Product Due to $\mathbf{C_{in}}$ (Norm to 2D Architecture while $C_{in} = 0.2$)** 84

5.6 **Average Planar Channel Width Variation Due to $\mathbf{C_{uni}}$ Weight γ** 86

5.7 **Average Routing Area Due to $\mathbf{C_{uni}}$ Weight γ** 87

5.8 **Average TSV Used Per Tile Due to $\mathbf{C_{uni}}$ Weight, γ** . 87

5.9 **Average Delay Due to $\mathbf{C_{uni}}$ Weight, γ** 88

5.10 **RADP Due to C_{uni} Weight, γ (Norm to 3D Architecture while $\gamma = 0.2$)** 89

5.11 **Average Delay** 90

5.12 **Average Planar Channel Width** 91

5.13 **Average Vertical Channel Width** 92

5.14 **Average Routing Area** 92

5.15 **RADP** 93

5.16 **Thermal Profile of Layer 0** 94

5.17 **Thermal Profile of Layer 1** 95

5.18 **Thermal Profile of Layer 2** 96

5.19 **Thermal Profile of Layer 3** 96

5.20 **Thermal Profile of Layer 4** 97

List of Accronyms

ADP Area Delay Product

ASIC Application Specific Integrated Circuit

BLE Basic Logic Element

CB Connection Box

CLB Cluster-based Logic Block

EDA Electronic Design Automation

FDM Finite Difference Method

FEM Finite Element Method

FPGA Field Programmable Gate Array

IC Integrated Circuit

I/O input/output

LUT Look-up Table

NEM Nanoelectromechanical

RADP Routing Area Delay Product

SA Simulated Annealing

SB Switch Box

SOC System On Chip

TSV Through Silicon Via

UB "Uni-Bi"

1 | Introduction

In the modern electronic design, FPGAs are favourable for digital circuit implementation in many key areas, for example Application Specific Integrated Circuit (ASIC) prototyping, data processing, and communication, because of its programming flexibility and fast time-to-market over ASICs.

It is well studied that the interconnect routing resource of FPGA brings programming flexibility but leads to poor logic density, low circuit speed, and high power consumption. Several studies indicate that the routing resource takes the dominant portion in total power consumption, chip area, and circuit delay. In [9], the study of Xilinx XC4003A power consumption shows that interconnect routing resources would consume up to 65% of total power. In addition, it is reported that more than 90% of total chip area would be taken by routing area [10] and up to 80% of total delay is from interconnect delay [11]. As a result, FPGA's performance is still far

behind ASIC.

Although those disadvantages limit the FPGA's application, the trend of demand is continually strong in the near future. It motivates the researchers in institutes as well as in industries to enhance FPGA's performance through both hardware and EDA improvement.

The rise and popularity of 3D integrated chip (IC) design technology in the recent years, like a strong dose, is now pushing FPGA performance much closer to ASIC-like. The 3D IC design is to stack the conventional 2D dies vertically so that each component has more near neighbours. It enhances the logic density, and reduces wire delay and routing power consumption significantly. Many studies on 3D IC integration design have proved its excellent performance over 2D conventional IC design. In [12, 13], the study on performance of 3D architecture shows that the delay can be improved more that 20%. The study of power consumption from [14] shows that up to 77.5% read power can be saved for 3D stacked DRAMs in the Tezzaron systems. Giving those benefits, we are motivated to study the 3D FPGA. In this thesis we focus on exploring the architecture and EDA of 3D FPGA which is implemented by stacking conventional 2D FPGA dies. Furthermore, we pay our attentions more on the TSV reduction and chip temperature reduction which are the two major design issues in the 3D IC

design.

1.1 2D FPGA Overview

Unlike early FPGA design, modern FPGAs have many different module blocks, such as DSP blocks, analog blocks, memory blocks and high speed communication modules, in order to adapt the complexities, varieties, and low cost of modern electronics design. Nonetheless, the dominant part of FPGA is always its programmable array. FPGA's programmable array is composed of two parts, the Cluster-based Logic Block (CLB) and routing resources which can be further divided into, routing track, SB and CB.

Any digital circuit can be implemented on the FPGA by properly synthesising and mapping the circuit to the programmable array. In addition, the circuit can be erased and be reprogrammed for multiple times. FPGA is favourable as a balanced solution between microprocessor and ASIC because the application on FPGA can run faster than on microprocessor and time-to-market is much shorter than ASIC design. However, heavy price is paid on power, area, and delay in order to achieve the programming flexibility. Therefore, research is constantly conducted in order to reduce those costs.

Beside optimization of the hardware architecture, software EDA for the circuit placement and routing is another part of FPGA research which can reduce those costs as well. A typical EDA flow has 5 stages described in Figure 4.1.

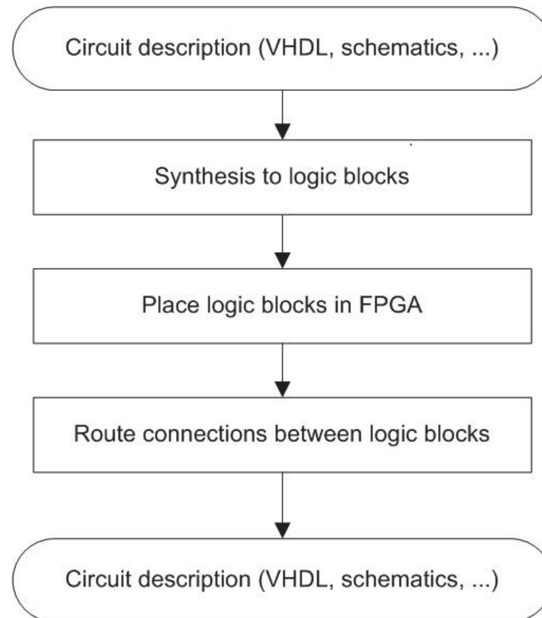


Figure 1.1: FPGA EDA Flow [1]

The description of circuit either in HDL or schematics is firstly translated and synthesized to CLB based circuit. Then each CLB's location on FPGA array is decided at placement stage. A detailed routing is later carried out to connect the input/output (I/O) of CLB in the routing stage. Lastly the FPGA programming file normally called Bit-stream is generated. Once the Bit-stream is programmed into the FPGA through JTAG interface, FPGA

can perform tasks as described in the first stage.

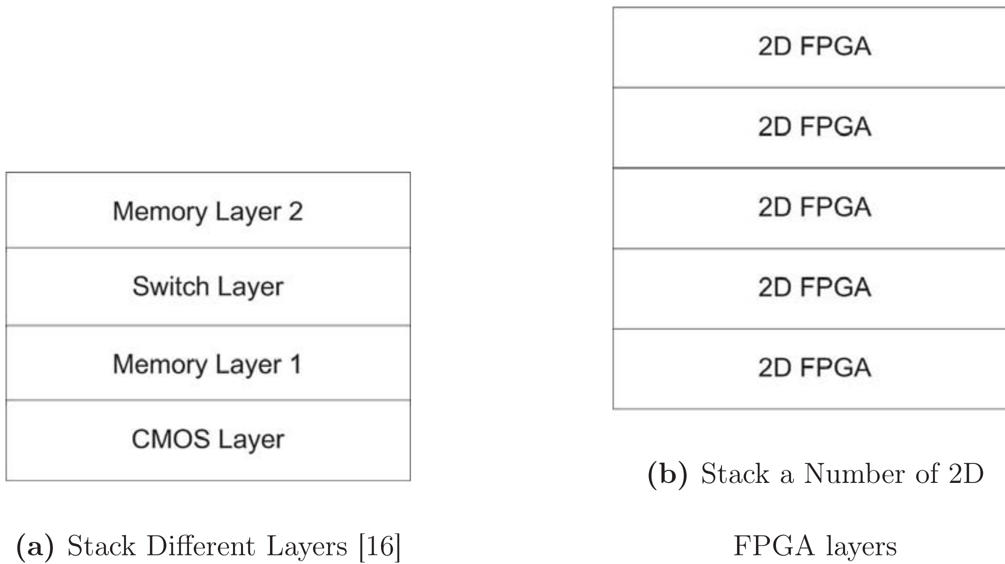
The architecture together with its EDA software decides the final circuit performance on FPGAs. Therefore, when a new architecture is designed, the corresponding EDA software should also be implemented accordingly to optimize the performance.

1.2 Overview of 3D FPGA

Same as the 3D ASIC design, 3D FPGA design aims to improve the circuit speed as well as the delay and routing power consumption. So far, a commercial 3D FPGA has not been released to the market, but, in the research area, various design issues regarding to the 3D FPGA architectures and EDA have been actively discussed in the past a few years.

At the architecture level, 3D FPGA can be categorized into two types, stacking the components such as [15, 16, 17] and stacking the conventional 2D FPGA such as[18, 19, 20]. The later 3D architecture is the type we discuss in the rest of the thesis because we believe this kind of architecture is closer to the future commercial 3D FPGA.

Figure 1.2a is one example of stacking components designs. The technology



(a) Stack Different Layers [16]

(b) Stack a Number of 2D FPGA layers

Figure 1.2: Two Different 3D FPGA Architectures

used by those designs is different from 3D stacking technology since those designs require the interlayer connection and landing pads with comparable size to the metal wire and metal via. Instead of stacking the traditional full CMOS process at each layer, the technology used in [15, 16] stacks components in a way such that the switch layer is uniformly comprised with one MOS transistor while the memory layer is implemented by 2-T flash or a programmable solid-electrolyte switch. The simulation based on VPR shows it can achieve much better performance than a baseline 2D FPGA [15, 16]. Another kind of 3D FPGA based on stacking components utilizes Nanoelectromechanical (NEM) relay technology to spread the CLB, SB and CB into different layers. It is quite similar to a face-to-face die

stacking based 3D FPGA. However, both designs later diminish. One of reasons is that the FPGA volume is not enough to drive the development of these technologies into the industry.

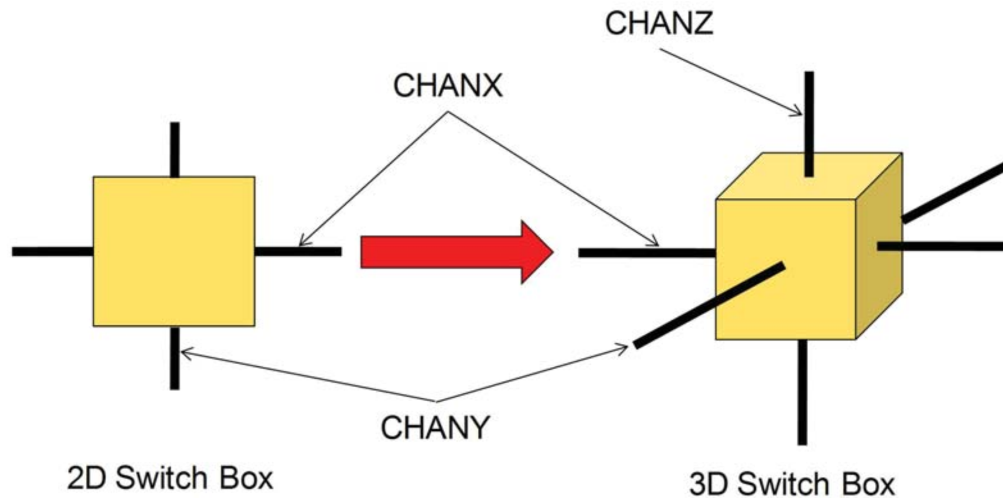


Figure 1.3: Evolution of 2D SB to 3D SB

Stacking the traditional 2D FPGA using TSV for interlayer connection as plotted in Figure 1.2b is discussed frequently. At architecture level, this approach keeps the general architecture of 2D FPGA with the extension of the SB from 2D to 3D as shown in Figure 1.3. Since every layer is homogeneous, the design cost is low and scalability to any number of layers is relatively flexible. However, due to the technology constraint on the size of TSV, vertical channel width, CHANZ, cannot be as wide as the planar channel width, CHANX and CHANY. Many efforts have been put to TSV reduction while keeping the cost on delay and area to be as low as possible.

On the other hand, with the increment of the logic density, the power density is increased at the same time. This leads to the temperature increase dramatically compared with the 2D FPGA. Therefore, thermal issue should be considered in the design of both hardware and software.

In the rest of this thesis, we focus the discussion on the die stacking based 3D FPGA because the technology gets mature and the development is frequently conducted in both research and industry.

1.3 3D FPGA Design Issues

Although the die stacking based 3D FPGA can bring great benefits to performance, various issues need to be addressed in order to optimize the performance. In this thesis, we present our solution to two major issues:

1. Due to the TSV technology constraints, a TSV usually occupies very large area compared with the finest process technology, such as the diameter of TSV and the corresponding keep out zone. Furthermore, TSV pitch which means the distance between the center of two adjacent TSV is still too wide and limits the TSV density. Therefore, how to use TSV more efficiently or how to reduce the number of TSV

used for the circuit placement is a great challenge for the survival of the 3D FPGA.

2. By stacking dies vertically, the power density per area increases rapidly with the number of stacking layers. How to lower down the temperature with less penalty on performance is another issue to maintain system stability and to keep the circuit performance.

1.4 Our Solutions

In order to solve the issues we mentioned in the previous Section 1.3, we propose our solutions from the architecture and the EDA design respectively. However, it does not mean that there is no relationship between the architecture and EDA software. On the contrary, the architecture and EDA are close related. The architecture must have the corresponding EDA to optimize the performance.

In this study, we discuss the performance of the uni-directional routing based 3D FPGA with 2 types of 3D SBs aiming to optimize the delay and to reduce the number of TSVs needed for the circuit placement. Unlike previous studies focusing frequently on 3D SB topology for the architecture

exploration, we take a different step to offer a direction configurable switch to further reduce the TSV demand.

Following, we present our EDA software which includes TSV aware placement algorithm and thermal aware placement algorithm to evaluate the architecture and also the placement algorithms themselves. The EDA software is developed based on two open source software, VPR 5 [21] with its corresponding power model [22], and HotSpot [23]. By integrating those software, we are able to introduce the additional thermal cost in the placement stage to optimize the circuit performance and to reduce the thermal impact.

During the design space exploration, we use the EDA software we develop and 20 MCNC benchmark circuits to explore the performance and enhancement of 3D FPGA with various configurations and parameter values. For the TSV aware placement, we compare with the results under different values of cost weights in the cost function. Regarding to thermal aware placement, we make the compare between the placement without the presence of the thermal cost and with the thermal cost.

1.5 Thesis Contributions

This thesis investigates two issues regarding to the 3D FPGA, the TSV reduction and the thermal alleviation through both the hardware approach and the software approach. Our results show that we can resolve those two issues with reasonable performance, such as circuit speed, planar channel width and TSV reduction. The contributions of this thesis are as follow:

1. From the hardware perspective, a uni-directional routing based 3D FPGA is proposed. In addition, we design two different types of 3D SBs aiming to reduce TSV usage while keeping good circuit speed. Furthermore, we also design a new switch, Uni-Bi (UB) switch, for the vertical channel (TSV) such that we can use the TSV more efficiently and hence reduce the number of TSVs placed per FPGA tile in average.

2. For the EDA software perspective, we implement placement and routing software based on VPR 5 for the uni-directional routing based 3D FPGA. Based on the characteristics of the uni-directional 3D FPGA, we develop the new placement algorithm which is used to reduce the TSV needed for the circuit placement and routing. We call this place-

ment algorithm "even number". It means the algorithm is based on the architecture that the number of TSVs used to transmit the signals in the downward and upward direction is the same. By using the UB switch, we also implement the corresponding EDA flow, "place twice", aiming to further reduce the number of TSV per FPGA tile. Besides, we implement the thermal aware placement by combining VPR power model and HotSpot.

3. From the design space exploration, it shows that our proposed 3D architecture with the corresponding EDA software is able to place and route the circuit with the delay less than 75% of the 2D baseline FPGA. The average planar channel width of the 3D FPGA is also less than 75% of the 2D baseline. Our best result shows that we can use 16% of fewer number of TSVs compared with the state-of-the-art work in average. Our thermal aware placement algorithm shows that the temperature of the chip can be reduced but the time of completing the placement is much longer than the placement without thermal aware.

1.6 Organization

The rest of the thesis is organized as follows. Firstly, the background of FPGA, EDA and related works are discussed in Chapter 2. In Chapter 3, the 3D FPGA architecture is presented for the uni-directional routing based with design of 3D SBs and "Uni-Bi" (UB) switch. In Chapter 4, we present our 3D EDA software and flow. We explore various design configuration and parameter values in the Chapter 5. The conclusion and future work is discussed in Chapter 6.

2 | Background and Related Work

FPGA is a kind of ICs whose functionality is implemented by programming its reconfigurable circuit. It can be configured multiple times for the purpose of system upgrade or function changes. Nowadays, FPGAs are widely used in the area of ASIC prototype, medical, automotive, and automation because of its ability on quick implementation with faster processing speed than microcontrollers.

In this chapter, we review the architecture and EDA development of the 3D FPGA. In Section 2.1, we introduce the 2D FPGA architecture which is the base model for the 3D stacking architecture. Then we introduce the 3D IC architecture in Section 2.2. Section 2.3 presents the base software and tool chain that we modify and implement for the 3D architecture. At last, we review the previous works on the 3D FPGA and thermal modelling.

2.1 2D FPGA Architecture

In addition to programmable logic arrays, a modern FPGA is composed of many different types of modules, such as digital signal processing (DSP) unit, arithmetic logic unit (ALU), memory, or high-speed communication interface. Nonetheless, the programmable array still takes the dominant portion in the FPGA design because it is the key component to offer the programming flexibility in the circuit implementation and short time-to-market.

The fundamental architecture of a FPGA is composed of three parts, I/O block, CLB, and programmable routing which are shown in Figure 2.1. Each CLB can implement a small portion of the circuit. I/O block can be used as either input or output. The programmable routing is used to link all the CLB and I/O block together as the netlist requires. In the Section 2.1.1, a more detailed view of each part is introduced.

2.1.1 Cluster-based Logic Block

The CLB as stated before, is the abbreviation of cluster-based logic block shown in Figure 2.2. It is a collection of smaller logic blocks, Basic Logic

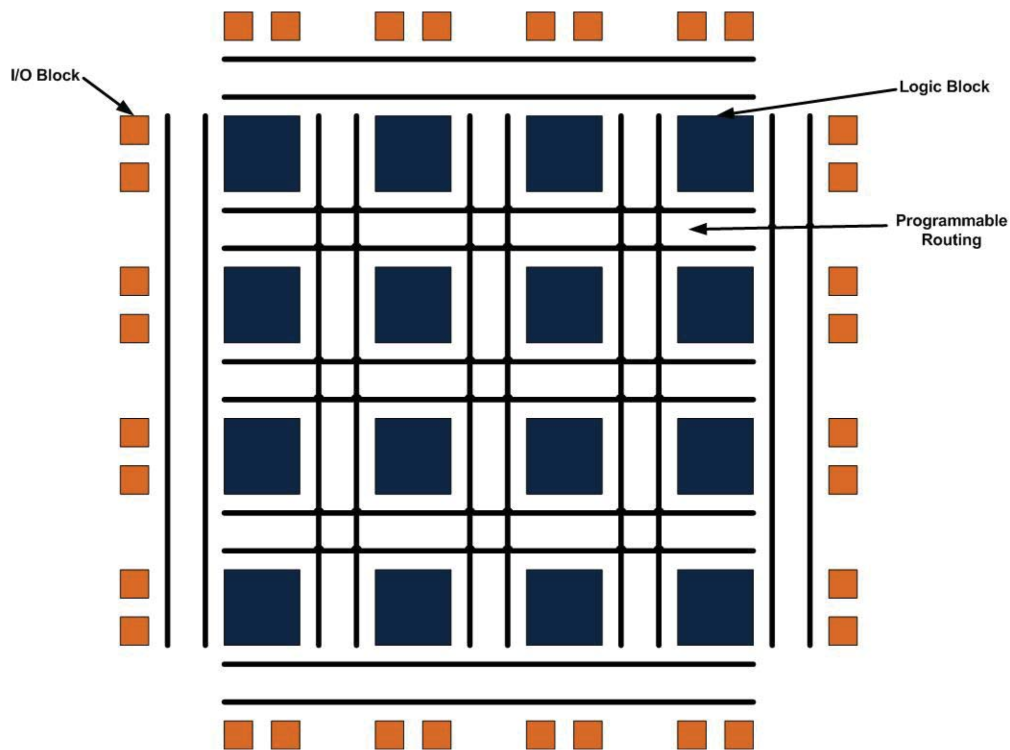


Figure 2.1: 2D FPGA in Abstract Level [2]

Element (BLE). Each BLE can implement a smaller portion of logic than the CLB. With local interconnect network within a CLB, it can form more complex logic circuit.

By closely looking at the inner part of the BLE, it can be further decomposed into a Look-up Table (LUT), a register and a two-input multiplexer as shown in Figure 2.3. The LUT is in fact a multiplexer with SRAM storing the output values. The number of a LUT inputs (the same as BLE inputs) K decides the number of data bit that a LUT can store as 2^K. A simple logic such as AND gate can be implemented by mapping the truth

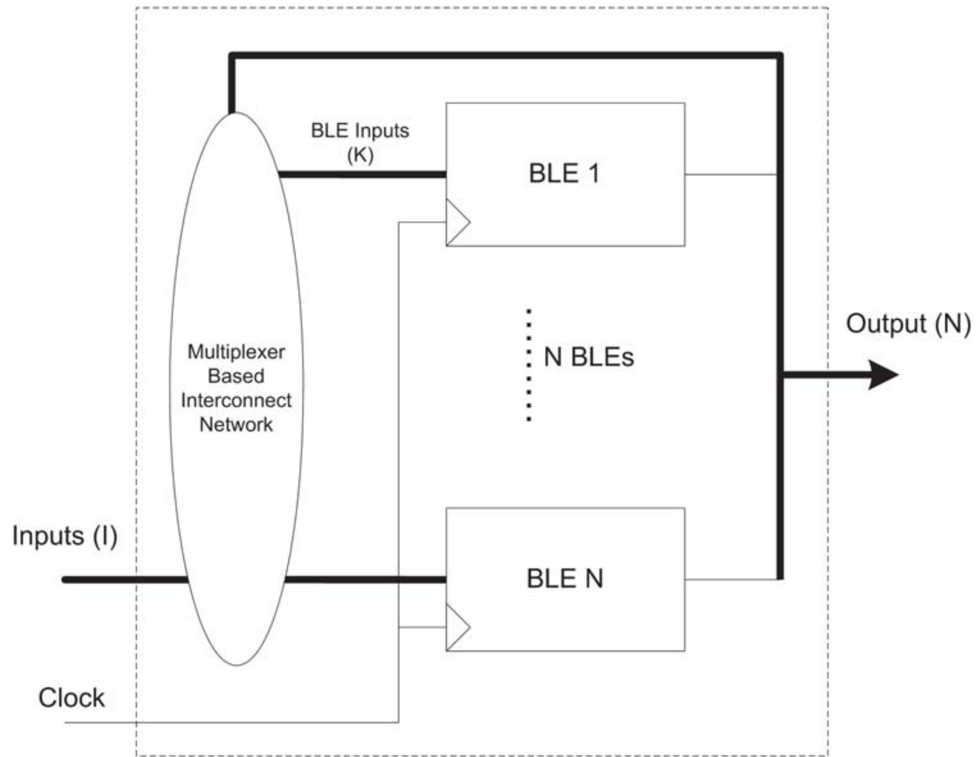


Figure 2.2: Cluster-based Logic Block

table to the corresponding LUT data bit. The respective AND gate output can be generated according to the LUT inputs. If required, a BLE can be implemented as either sequential circuit or combinational circuit by choosing the LUT output or clock based register output accordingly as shown in Figure 2.3.

For the purpose of the simplification, we use following parameters, which

are the same as in [1] to describe the CLB:

- I: the number of a CLB inputs
- N: the number of BLEs in a CLB or number of a CLB output
- K: the size of a LUT or the number of a BLE inputs

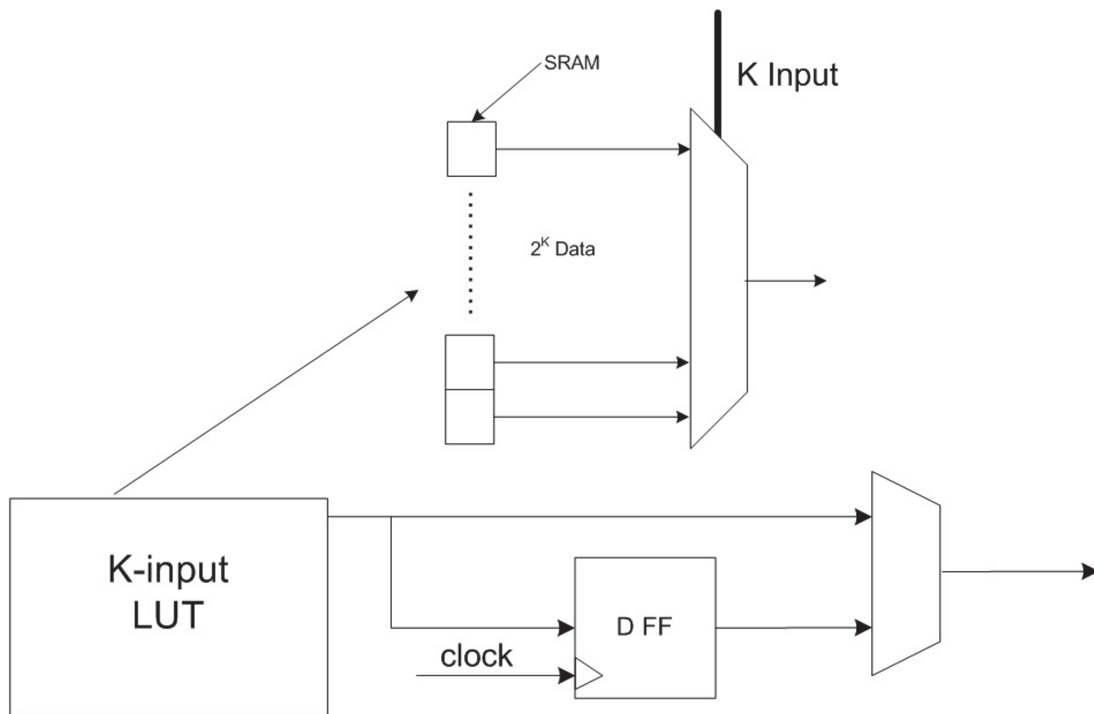


Figure 2.3: Basic Logic Element

For the rest of the thesis, our discussion on the FPGA architecture always uses the LUT with its size, $K = 4$.

2.1.2 Programmable Routing

The programmable routing is composed of three parts, SB, CB, and routing channel as shown in Figure 2.4. SB is used to connect routing tracks from one to another at every crossing point of routing channel CHANX and CHANY [24]. CB is used to connect the I/O of CLB or I/O block to the routing track adjacent to it [24]. The routing channels surround the four side of a CLB and are used to transmit signal among CLBs and I/O blocks.

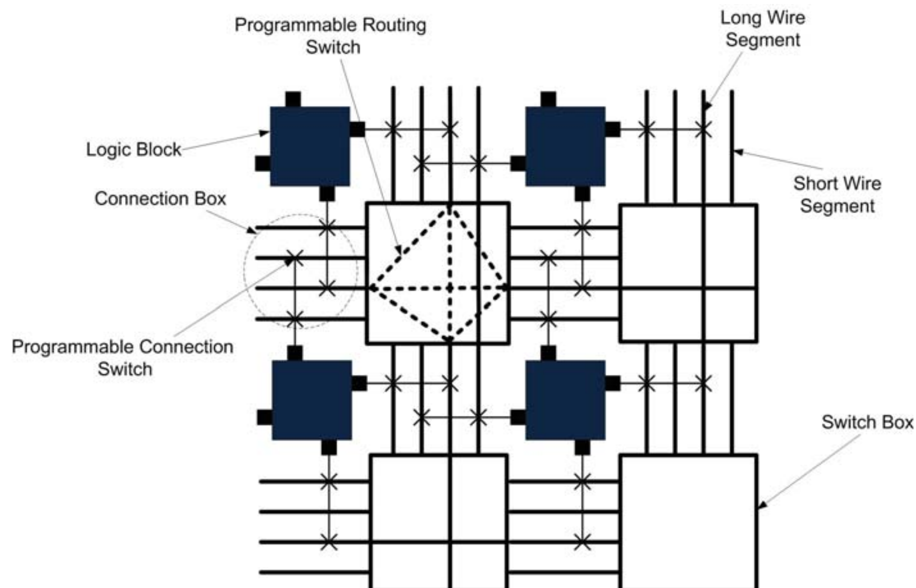


Figure 2.4: 2D FPGA Architecture (Island Style)

To describe the routing resources, we have the following parameters which are also used in [24, 1]:

- F_c(CB flexibility): the number of tracks or (percentage of total channel width) that an I/O of CLB or the I/O block can connect to in a CB

- F_s(SB flexibility): the number of tracks that an incoming tracks can connect to in a SB

- W(channel width): the number of tracks in a routing channel

- L(length of track): the number of CLBs that a track spans.

In addition, another programming routing parameter discussed in [25] and [26], called internal population, is used to describe the connectivity of track with length more than 1 as shown in Figure 2.5.

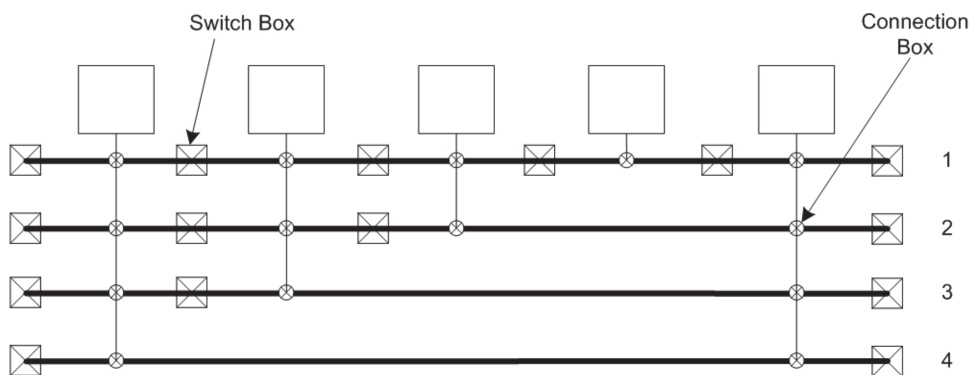


Figure 2.5: Internal Population

In Figure 2.5, it shows four different population cases for the SB and CB with wire length $L = 5$. For the track 1, the population of the wire for the

SB and CB is 100%. For the track 2, the population of the wire for CB is 80% since 4 CBs out of 5 are connected and 67% of the population of the wire for the SB as 4 SBs out of 6 are connected. For any routing track with length more than 1, it should have the population rate of the SB greater than 0% due to the fact that the two ends of the track should connect to the SB. We use P_c and P_s to represent the CB and SB internal population of the corresponding tracks.

In the early time, FPGAs use a bidirectional base programmable routing architecture, such as Xilinx XC4000 [27]. In the current commercial FPGA like [28] and [29], uni-directional routing base architecture becomes popular because of its better performance in power, delay and area. For the bidirectional routing architecture, signal can be transmitted in either direction in a specific track. However, for the uni-directional routing architecture, the signal can be transmitted in one direction only for a specific track as shown in Figure 2.6 from [3].

According to the study of [3], 32% of Area Delay Product (ADP) saving can be achieved by adapting the uni-directional routing architecture.

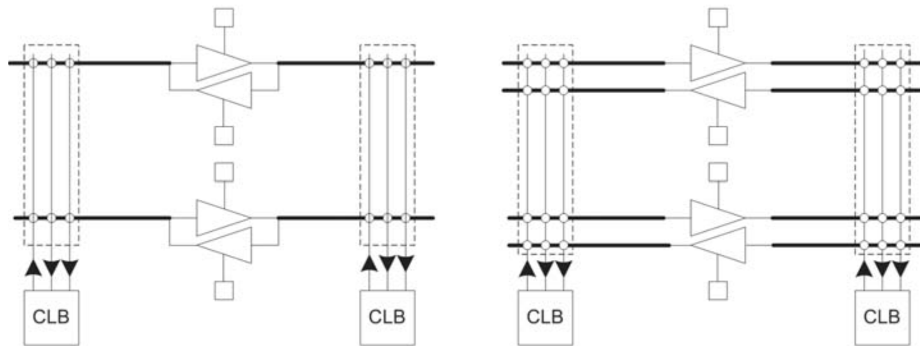


Figure 2.6: Bi-directional and Uni-directional Wires [3]

2.2 3D IC Architecture

Pursuing for faster speed with lower power consumption, 3D IC design is actively developed in both research and industry when there is little space for shrinking transistor size to continuously keeping the validity of Moore's Law. By stacking the dies vertically, 3D IC can improve the interconnect delay and system performance significantly [30], compared with the 2D design.

In a simplified model, a 3D IC is implemented by stacking the conventional 2D IC die vertically. Among each dies, the TSV is used for interlayer connection. In [31] and [4], it describes three types of die bondings and two types of TSVs as shown in Figure 2.7.

Face-to-face bonding is to place the device layer of two dies towards each

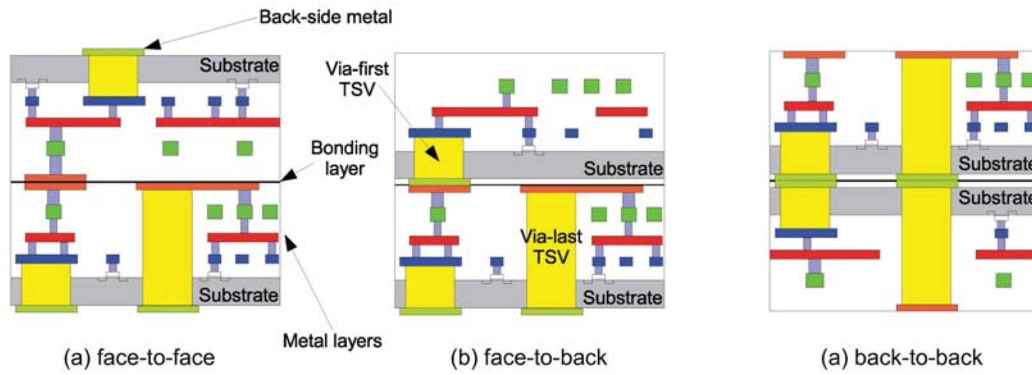


Figure 2.7: Three Types of Die Bondings and Two Types of TSVs [4]

other. The back-to-back bonding is similar to face-to-face bonding except the substrate layers being placed towards each other. The face-to-back bonding is to place one die on top of the other with same layer sequence. Via-first TSV and via-last TSV are two fabrication technologies that fabricate TSV before fabricating device and metal layer or not accordingly.

For the rest of this thesis, face-to-back bonding technology is applied for 3D FPGA architecture since it has good scalability to any number of layers. In addition, TSV fabrication technologies are not in the scope of this thesis. However, in our discussion, we will consider the delay and area cost of TSV which is our focus.

2.3 EDA Framework

VPR [5] and HotSpot [7] are two open source programs which are widely used in the academic research for 2D conventional FPGA architecture evaluation and IC thermal profile simulation accordingly. In Section 2.3.1 and Section 2.3.2, we briefly introduce these two programs.

2.3.1 VPR Framework

VPR, the abbreviation of Versatile Place and Route, is the EDA program to perform circuit placement and routing on FPGA. The general flow of VPR is to firstly read the technology mapped netlist and FPGA architecture file and then to perform the placement followed by routing. The results are stored in the output file as shown in Figure 2.8.

Since 1997, VPR has been developed and updated for several versions. In this thesis, we use VPR 5 [21] as the base software framework which is not the latest one but enough for us to use. VPR, only itself, is not sufficient to implement a complete flow from HDL circuit description file to the final placement and routing. In Section 2.3.1.1, a briefly EDA flow is discussed. In addition, a power estimation model [22] is developed based

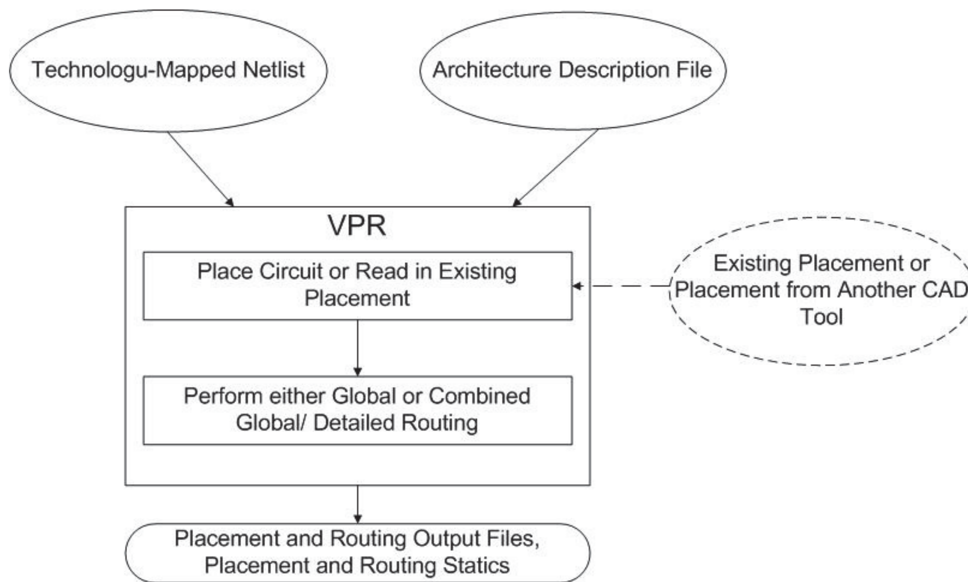


Figure 2.8: VPR Flow [5]

on VPR 5.0 is briefly discussed also. VPR 5 uses Simulated Annealing (SA) algorithm to perform the placement. Section 2.3.1.2 discusses VPR placement algorithm since the placement has significant impact on the 3D FPGA circuit performance.

2.3.1.1 VPR Based EDA Flow

Figure 2.9 shows the EDA flow that VPR uses. A circuit described in Verilog HDL is translated into flattened netlist of logic block and black boxes by ODIN [32]. ABC [33] then synthesizes the output of ODIN to the .blif format with LUTs and flop flops. T-Vpack [1] packs the LUTs and flip

flops into logic blocks. At last VPR performs the placement and routing.

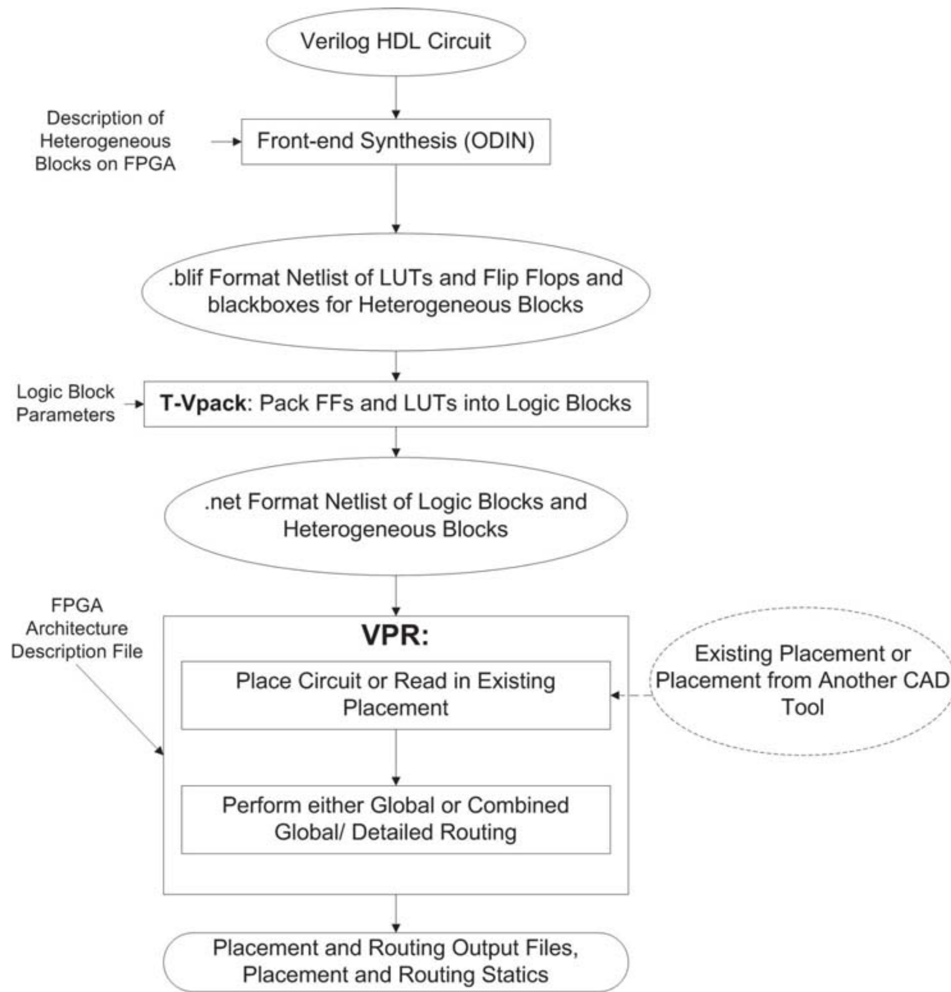


Figure 2.9: VPR EDA FLOW [6]

2.3.1.2 VPR Placement

The placement algorithm is used to identify the location of each CLB on FPGA. The object of placement is to move CLBs to a proper location in

order to achieve various optimization purposes, such as delay minimization, wire length minimization or power minimization. In VPR, a SA based placement algorithm is developed to deliver a good placement result.

$S = RamdomPlacement();$

$T = InitialTemperature();$

$R_{limit} = InitialR_{limit};$

while $ExitCirterion() == False$ **do** ▷ Outer loop

  while $InnerLoopCriterion() == False$ **do** ▷ Inner loop

    $S_{new} = GenerateViaMove(S, R_{limit});$

    $\triangle C = Cost(S_{new}) - Cost(S);$

    $r = random(0, 1);$

    if $r < e^{-\triangle C/T}$ **then**

      $S = S_{new};$

    end if

    $T = UpdateTemp();$

    $R_{limit} = UpdateR_{limit}();$

  end while ▷ Inner loop

end while ▷ End Outer loop

Figure 2.10: Pseudo Code of a Generic SA Based Placer [1]

Figure 2.10 lists the pseudo code of a SA placement. A cost function, $Cost()$, is developed to access the cost variation for any CLB placement change. An initial cost can be calculated by placing each CLB randomly. The cost can be changed by moving a CLB to a new location or exchanging the locations of any two CLBs. Such movement is accepted if $\triangle C$ is negative at the tendency of cost reduction. However, if $\triangle C$ is positive, the acceptance of the movement is depending on the value of $e^{-\triangle C/T}$. T (temperature) controls the probability that a movement can be accepted while $\triangle C > 0$. The inner loop exits criterion and inner loop exit criterion are defined by the annealing scheduler.

2.3.1.3 VPR Power Model

VPR power model [34, 35] is firstly developed under VPR 4 and is upgraded to VPR 5 later with power estimation for uni-directional routing architecture [22]. Figure 2.11 shows the software flow of VPR 5 power model which is extracted from [36].

The benchmark circuit is in the .blif format. The ABC is used to map the circuit to the LUT of different number of input such as 4, 5, or 6. The Input simulation of circuits together with CLB based .net file and LUT

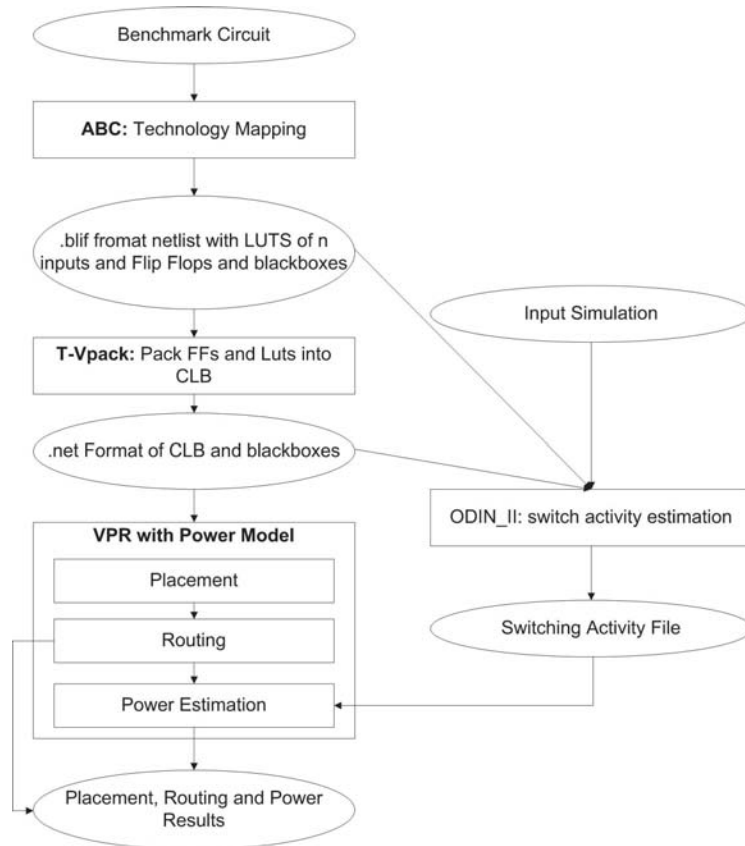


Figure 2.11: VPR 5 Power Model Software Flow

based .blif file is used by ODIN II to generate the switching activity file which is used after post routing stage for power calculation. In this thesis, we modify the VPR 5 power model to estimate the power consumption for individual FPGA thermal tile at the placement stage since the power profile is essential for the thermal estimation.

2.3.2 HotSpot Thermal Framework

HotSpot is an open source program which is used to study the thermal profile at the architectural level. It uses Finite Element Method (FEM) to build the equivalent thermal RC circuit network as shown in Figure 2.12. Based on HotSpot model, Hotfloorplan [37] is developed. It is a SA based program for thermal aware floorplan. In this thesis, we develop the 3D FPGA thermal profile based on Hotfloorplan.

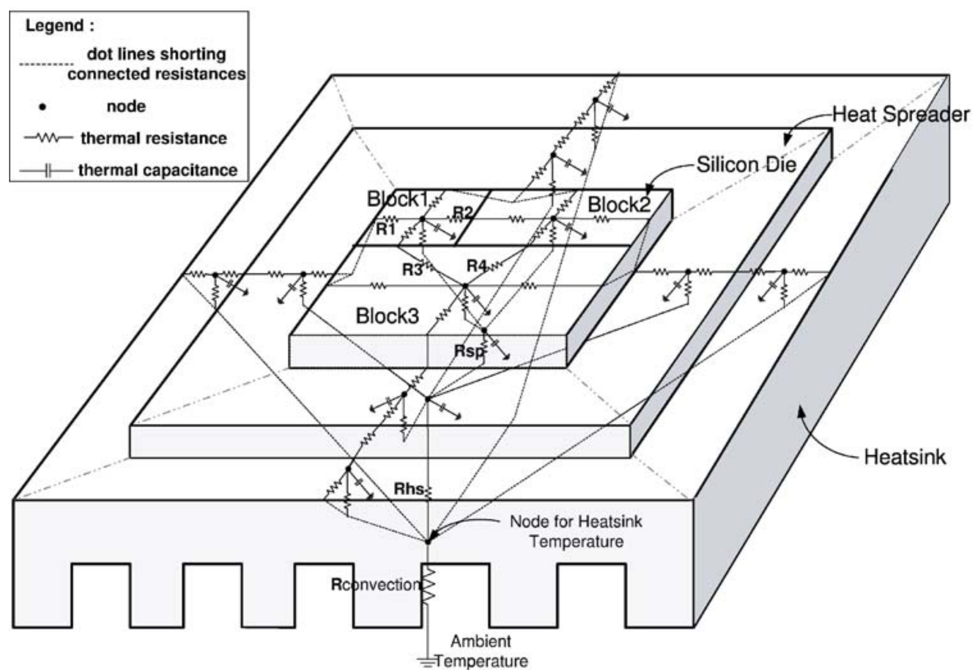


Figure 2.12: HotSpot RC thermal model [7]

2.4 Related Work

In this section, we review the related works from three fields. In Section 2.4.1, we discuss the related work on 3D FPGA architecture. In Section 2.4.2, we review the EDA development of 3D FPGA in the previous work. At last, we review the thermal model related work for 3D IC modelling.

2.4.1 3D FPGA Architecture

As we know [38] is the first paper which discusses 3D FPGA architecture. According to [38], an average of 13.8% of wire reduction can be achieved when changing from 2D FPGA to 3D FPGA architecture. However, it does not reveal the circuit delay improvement and detailed architecture. In addition, it does not involve the consideration of the TSV size which still takes large area in μm^2 scale.

[39] and [40] propose a 3D FPGA architecture (Rothko) which is developed based on the Triptych architecture [41]. Although, it is reported by [41] that the Triptych architecture yields better logic density and performance than the traditional architecture, it is not adapted by the major FPGA vendor. On the other hand, [39] and [40] do not offer the performance

improvement of the 3D architecture.

[42] and [43] propose a 3D Tree-based FPGA. In their discussion, they analyze the design constraint on the number of TSV with thermal issue and propose their solutions which show significant reduction on the number of TSV with little performance penalty. However, the architecture they propose is quite different from the main stream. As a result, adopting their 3D architecture in the industry may result in very high cost compared with using the existing uni-directional routing based architecture.

[15] and [16] propose monolithically stacked 3D FPGA which distributes components into different layers. The 3D architecture described in [16] shows that the logic density is 3.3 times of the baseline 2D FPGA with 2.35 times lower critical path delay and 2.82 times lower dynamic power consumption.

[17] proposes a 3D FPGA architecture utilizing NEM relay technology, called 3D CMOS-NEM FPGA. It is reported that up to 41.9% delay reduction can be achieved compared with the baseline.

However, both monolithically stacked 3D FPGA and 3D CMOS-NEM FPGA require special fabrication technologies which are not widely used in industry or research.

Focusing on the more common die stacking based 3D architecture, [18] [19] [44] [20] and [45] take efforts on the stacking traditional 2D FPGA with 3D SB. In [18] [19] [44] and [20], TSV issue regarding to size and area cost is considered and studied. Those works concentrate more on the 3D SB design to raise the TSV utilization and to reduce number of TSVs required per 3D SB. [44] and [20]'s results show good 3D SB topology can reduce not only the TSV demand but also the critical path delay. In addition, delay cost is also considered in their design to balance the cost with area. [45], on the other hand, proposes a new architecture for TSV reduction by using heterogeneous SB architecture (2D and 3D SB exist at the same time). However, such architecture has difficulties of physical layout since it assigns large number of TSVs in a few 3D SB.

2.4.2 3D FPGA EDA Software Tools

[18] and [45] develop their own EDA flow and software for architecture evaluation referring VPR flow.

The EDA software described in [18] is called TPR which is evolved from VPR 4.3. The placement of TPR firstly uses hMetis [46] to partition the circuit into several parts (according to number of layers) with almost equiv-

alent number of CLBs while the number of interconnect among any of two parts is almost the same. After the initial layer assignment, TPR then starts the placement at each layer from the first to the last according to the order of the layer. The routing program of TPR is almost the same as VPR. As indicated in [18], TPR's placement algorithm time is much faster than SA based. However, regarding to the delay, a SA based 3D placement algorithm which is also proposed in [18] can yield better result.

3D MENADER, another 3D FPGA EDA framework, is proposed in [45] and [47] by combining the EX-VPR[48] and TPR framework. Compared with TPR, 3D MEANDER has more functions and options of configuration, such as the 3D SB configuration, power estimation and option for different 3D layout. The EDA flow is more or less the same as TPR.

Nonetheless, both of them investigate the bidirectional routing based 3D FPGA. Since the uni-directional routing based FPGA is dominant in the market. It is necessary for us to implement a new EDA software to explore the uni-directional routing based 3D FPGA.

2.4.3 Thermal Estimation

The compact thermal model is nowadays used in many research for thermal modelling. In [49], a cell level 2D IC thermal model and thermal-aware placement algorithm are proposed. The author builds the thermal resistor network by applying the Finite Difference Method (FDM) which can be considered as a special case of FEM. The thermal placement algorithm used by [49] is developed from SA, and it is similar to the one we introduce in Section 2.3.1.2.

[37]'s thermal placement algorithm is quite similar to [49]. However, it builds its thermal resistor network by applying FEM because it optimizes the thermal issue at the floorplan level and typically the size and the shape of each block are quite different, compared with a cell level placement.

Several thermal modelling works related to FPGA are also conducted. [50] implements a System On Chip (SOC) application on FPGA with ring oscillator thermal sensors to validate the corresponding Hotspot thermal model. [51] develops the CLB thermal placement and validates its model with HotSpot.

3 | 3D FPGA Architecture

In this chapter, we present our proposed 3D FPGA architecture based on the island style 2D FPGA. In Section 3.1, an overview of 3D FPGA architecture is presented. In Section 3.2, we present the architecture of the uni-directional routing based 3D FPGA we propose.

3.1 Overview of 3D FPGA Architecture

In this section, we discuss the basic 3D architecture that we use. Figure 3.1 is a three-layer stacked 3D FPGA which uses face-to-back layout. As mentioned in the earlier section, this 3D architecture is obtained by stacking the traditional 2D FPGA. It consists of I/O pad, CLB and 3D SB, CHANX, CHANY, and CHANZ. Compared with the traditional 2D FPGA architecture, the most obvious difference is the 3D SB and CHANZ

which consists of TSV.

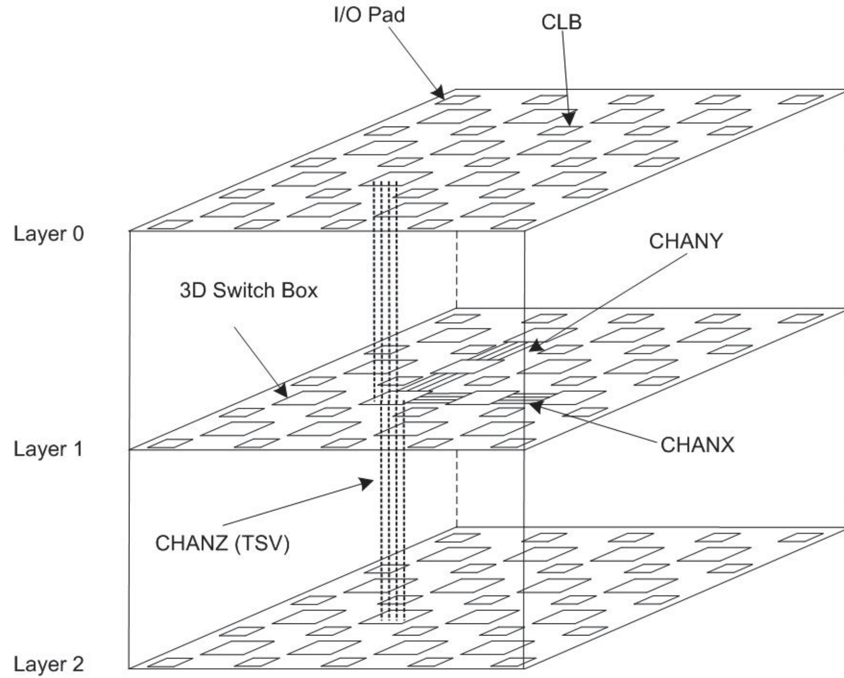


Figure 3.1: 3 layer stacked 3D FPGA

Instead of the conventional 2D SB which consists of four directions, 3D SB has six directions. The additional 2 dimensions are in vertical direction. We follow the direction definition in TPR to name the direction as LEFT, RIGHT, TOP, BOTTOM, ABOVE, and BELOW. The LEFT and RIGHT directions are along CHANX. The TOP and BOTTOM directions are along CHANY. The ABOVE and BELOW directions are along CHANZ.

CHANZ is implemented by TSV which usually takes up large area compared with the size of the unit transistor. Therefore, the number of TSV

in CHANZ is much fewer than the CHANX and CHANY. Figure 3.2 shows TSV layout.

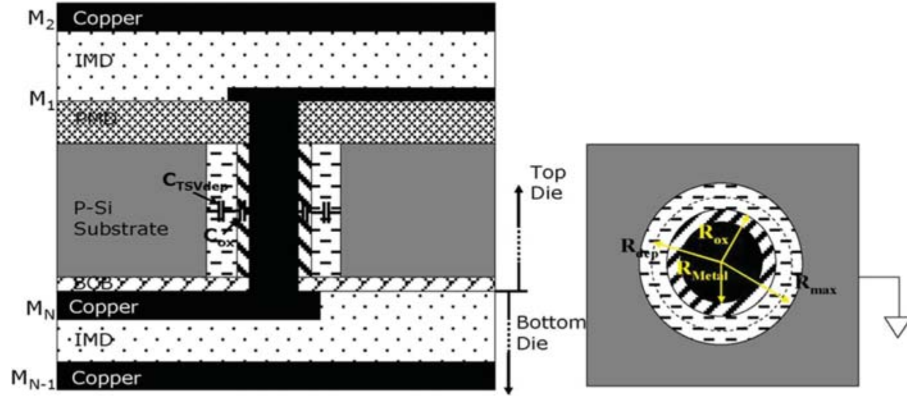


Figure 3.2: TSV Layout [8]

In [8], it provides a simplified model for the TSV's resistor and capacitor estimation with corresponding examples. In this thesis, we refer the data in [8] and come out the following parameters for the TSV we use:

- $D_{TSV} = 2\,\mu m$
- $R_{TSV} = 119\,m\Omega$
- $C_{TSV} = 52.37fF$

In additional, we assume that one TSV takes the area cost, $A_{TSV} = 16\,\mu m^2$ considering the landing pad and keep out zone as shown in Figure 3.3. Since we do not have any software tools or reliable resource of TSV data, in this

thesis, we only can estimate the value from other's research work which may not be accurate but should be more or less close to the quantity grade of the accurate data.

In Chapter 2, we have introduced some parameters to describe the 2D FPGA architecture. In order to describe the 3D architecture, we add some additional parameters to describe the 3D architecture as below:

- W_h(planar channel width): the channel width of CHANX and CHANY. In this thesis, the channel width of CHANX and CHANY always equals to each other.
- W_v(vertical channel width): the channel width of CHANZ which indicates the number of TSVs that connects two vertically adjacent 3D SB.
- N_{layer}(number of layers): the number of stacked conventional 2D FPGA dies.

Since the channel width of CHANZ and channel width of CHANX and CHANY are located in the different surfaces, horizontal surface and vertical surface, it is necessary to differentiate them with subscripts, h and v. By using the additional parameters, we now can describe a complete 3D FPGA

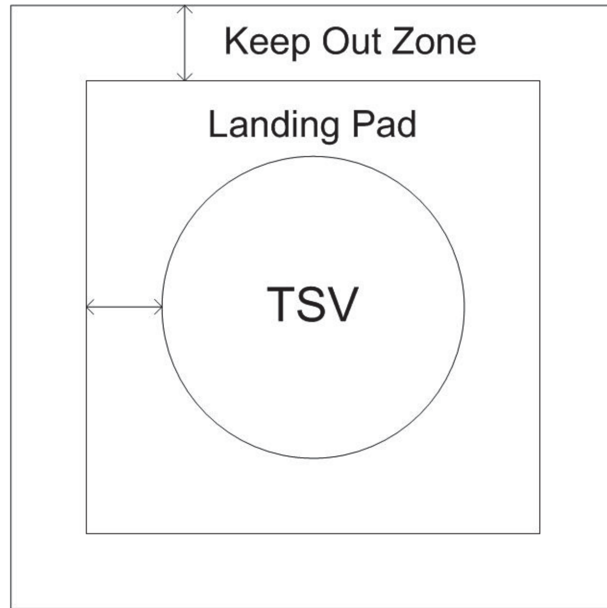
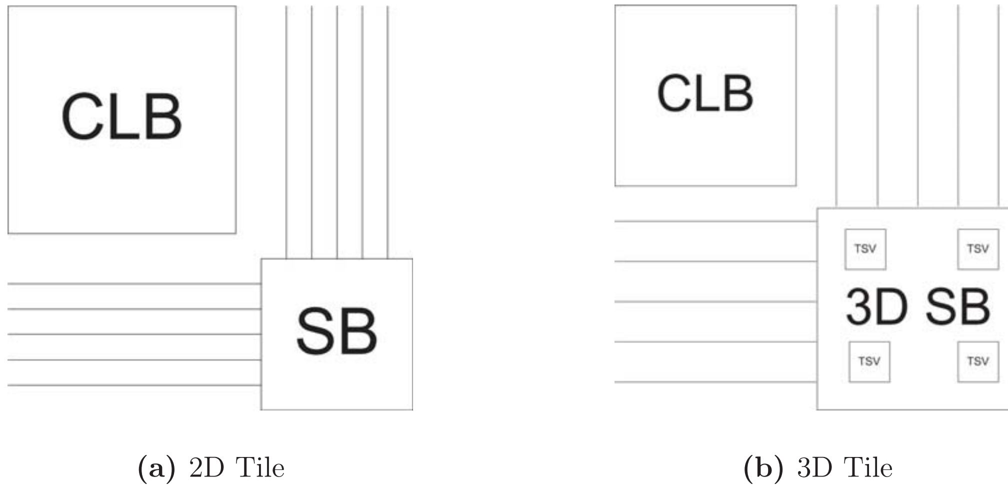


Figure 3.3: TSV Cell

architecture.

Tile is an important concept to describe the FPGA's architecture. In the 2D architecture, a tile contains a CLB and routing resources associated with it. If a FPGA can be built by repeating tile, this means the architecture can be implemented with less design cost and has the good scalability to any array size. In the 3D architecture, the scalability not only means expansion to any number of layers in the vertical direction but also means the spanning in the horizontal direction. Therefore, we also introduce the 3D tile concept as shown in Figure 3.4.

In each 3D FPGA tile, there are a few TSVs which are used to construct



(a) 2D Tile **(b)** 3D Tile

Figure 3.4: 2D FPGA and 3D FPGA Tile

the vertical channel, CHANZ. Due to the TSV fabrication constraint, the number of TSVs in each tile is limited. Furthermore, placing too many TSVs will increase area of the 3D tile such that it affects the logic density. Therefore, from the architecture perspective, maintaining the routability between the CHANZ and planar channels, CHANX and CHANY, will increase the utilization rate of the TSV so that we can place fewer TSVs in each tile. As a result, we can lower area cost caused by TSVs.

3.2 Uni-directional Routing Based 3D FPGA

Nowadays, the uni-directional routing architecture has replaced the bi-directional routing architecture in the commercial FPGA design due to

its better performance of ADP. In this section, we propose the 3D FPGA architecture, called UNI-3D, based on the uni-directional routing architecture. Unlike the bidirectional routing architecture, the uni-directional routing architecture requires equal number of tracks whose signals propagate on it in the opposite directions. Due to the scarcity of TSVs, maximizing the routability "from and to" the TSV in the hardware design tends to reduce the number of TSVs demand for placing and routing the netlist and to improve the delay. In this section, we present our design on raising the routability for the TSV related connection.

3.2.1 2D Uni-direction Routing Architecture

Figure 3.5 is an example of 2D uni-directional routing based SB with $W_h = 6$. For a uni-directional routing architecture, the number of tracks in a channel must be an even number to pair the tracks carrying signals either forward or backward. For instance, CHANX's channel width is 6. Then there are 3 tracks in which signals transmit to LEFT direction only and another 3 to the RIGHT direction only. This is different from the bidirectional routing architecture where signal can transmit in either direction. On each side of the SB, the outgoing tracks use multiplexers to switch the connection with the incoming wire or output pins. The output pins of the

CLB directly attach to the multiplexer of the corresponding outgoing wire. The input pins of CLB, the same as the bidirectional case, connect to the routing tracks by multiplexers.

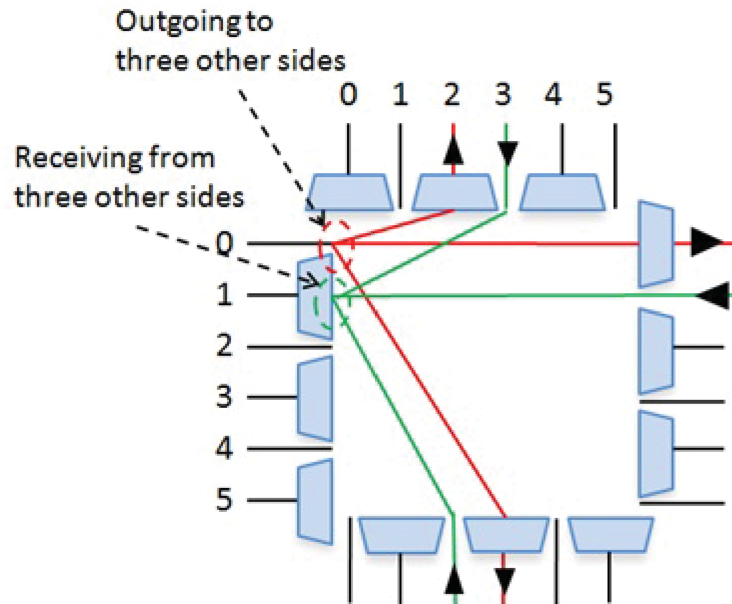


Figure 3.5: 2D Uni-directional Routing Architecture

3.2.2 3D-SB-1 and 3D-SB-2 3D SB

In this section, we introduce two 3D SBs, 3D-SB-1 and 3D-SB-2, aiming to increase the routability and to reduce the delay. Compared with the 2D SB, The 3D SB for the uni-directional routing based 3D FPGA involves the TSV connection to enable the signal transmission in ABOVE and BELOW direction.

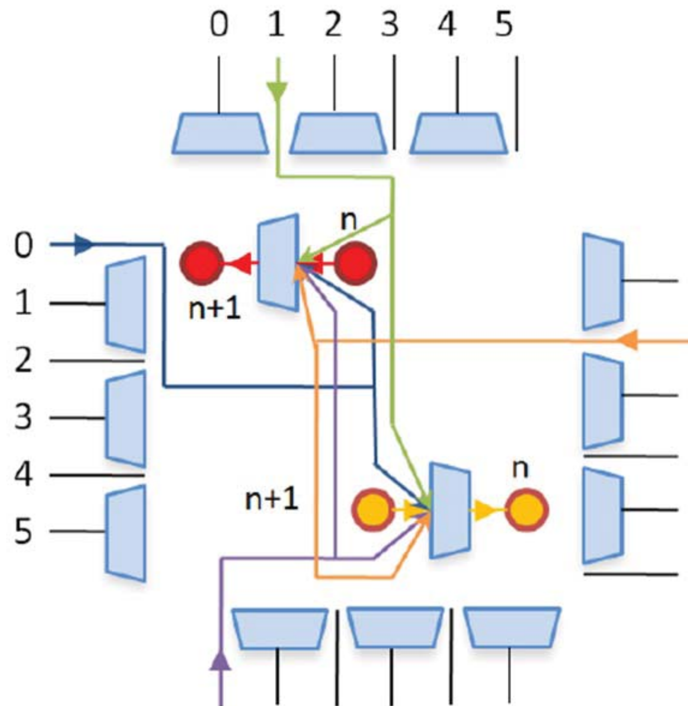


Figure 3.6: 3D-SB-1 SB

The 3D-SB-1 is implemented by further adding the connections between the vertical track and planar track into the 2D Wilton SB, which is introduced in the previous section. The logic connection of 3D-SB-1 is shown in Figure 3.6. For clarity purpose, we remove the planar track connections and vertical track connections to planar track connection.

In Figure 3.6, the solid circles in orange color represent the vertical track for ABOVE connection, and the red color circles are for BELOW connection. Two mux-separated circles are associated with each color, to represent the

TSV associated with different segments. We use blue, green, orange, and purple colors to represent the first incoming wire at each direction according to the track numbering sequence. For the planar track to vertical track multiplexer, each track, green color for example, connects to the vertical channel multiplexer to propagate signals to the ABOVE or BELOW direction. The connection sequence is based on the numbering sequence. For example, the first incoming track from all the four planar sides connects to the first outgoing multiplexer at both the BELOW and ABOVE direction sides. There is no connection if the sequence number of the n-th incoming planar track is greater than the total number of the outgoing multiplexer at the vertical direction. However, for connection from vertical track to the planar multiplexer, we need to carefully connect to the multiplexers of the planar side to avoid two vertical tracks attaching to the same multiplexer so that we can maximize the routability.

On the other hand, for the source of an interlayer net, routing to the vertical track in 3D-SB-1 depends highly on whether there is a planar path available to the vertical channel. It not only needs additional planar bandwidth to ensure the routing, but also increases the delay of the path. To further improve delay and routability, we propose a new SB named 3D-SB-2, by adding the output pin to the vertical track connection as shown in Figure

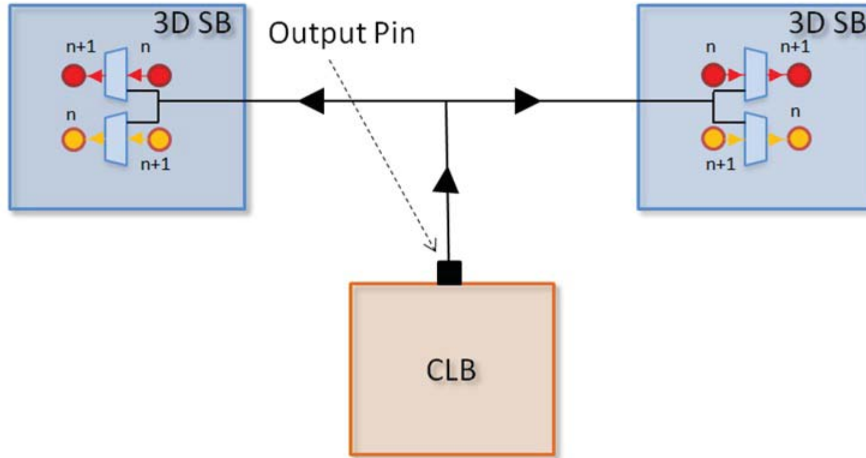


Figure 3.7: Output Pin to Vertical Track Connection

3.7. For any output pin, we add the switch connection for the vertical tracks at two adjacent 3D SB. In such way, we expect the routability and delay can be further improved.

The pseudo code of the 3D-SB-1 and 3D-SB-2 without the output pin connection is shown in the Figure 3.8. For planar-to-planar track routing, we adopt the 2D Wilton switch mechanism. If it is planar-to-vertical track (or outgoing multiplexer) transmission, then the connection follows the number sequence according to what we explained earlier: the incoming track of the from side, fr_in_track, connects to the vertical outgoing multiplexer, to_mux, with the same sequence number unless the sequence number of the incoming track is unable to match a corresponding outgoing multiplexer. If it is vertical-to-planar track, we align the track-

The switching algorithm

```
1:   if all tracks planar
2:      Use Wilton
3:   else if planar-to-vertical track
4:          && satisfy number sequence constraint
5:          /*from_incoming_track*/
6:      to_mux = fr_in_track
7:   else if vertical-to-planar track
8:      if fr_in_track in BELOW direction
9:          to_mux = num_fr_in_track × to_side_index
10:         to_mux += fr_in_track
11:         to_mux %= num_to_side_mux
12:     if fr_in_track in ABOVE direction
13:         to_mux = n_fr_in_track × (to_side_index + 1)
14:         to_mux += fr_in_track
15:         to_mux %= num_to_side_mux
16:  else to_mux = fr_in_track
```

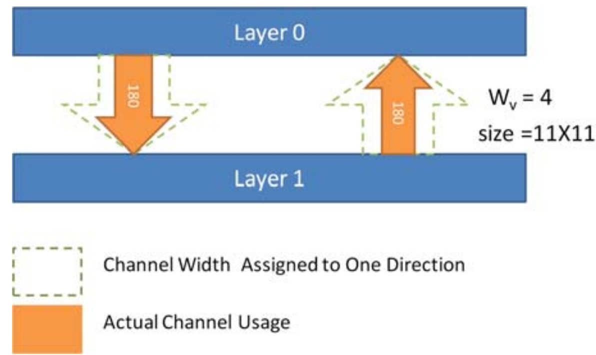
Figure 3.8: SB Pseudo Code

to-multiplexer connection as our formula stated in the pseudo code line 8 to line 15, where $num_fr_in_track$ is the total number of incoming track, $num_to_side_mux$ is the total number of outgoing multiplexer, to_side_index is the respective sequence number of the "to" side (where we define from 0 to 5 as the index of the six directions respectively). If the connection is between vertical tracks, then it travels from the n-th incoming track to the n-th outgoing multiplexer.

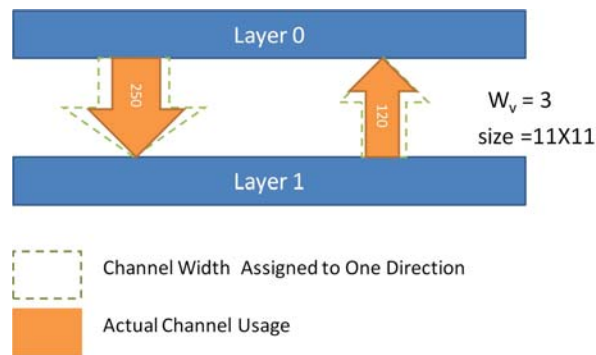
3.2.3 The "Uni-Bi" Multiplexer Switch

Considering a simple scenario that a circuit maps to a 2-layer 3D uni-directional FPGA, as shown in Figure 3.9a, if, for example, the placement result shows that 180 nets need to propagate signal from Layer 0 to Layer 1 and the other 180 from layer1 to Layer 0. On the other hand, this architecture can provide up to 576 vertical tracks of which 288 are for the direction from Layer 0 to Layer 1 and 288 are for the direction from Layer 1 to Layer 0. Assuming each interlayer net only uses one vertical track, then there will be $576 - 180 - 180 = 216$ unused vertical tracks. That is 37.5% of the total vertical tracks.

Instead of evenly assigning the vertical channel capacity to each direction in



(a) Balancing Direction Assignment



(b) Unbalance Direction Assignment

Figure 3.9: Unbalance Assignment to Reduce Vertical Channel Width

the conventional uni-directional routing architecture, assigning the capacity according to the best optimization result of vertical channel width is able to reduce the number of TSVs required in the circuit. It means the need for the flexibility of vertical channel width assignment. In this thesis, we introduce a new switch called UB switch which is the modification from the conventional wire multiplexer switch used in the uni-directional routing

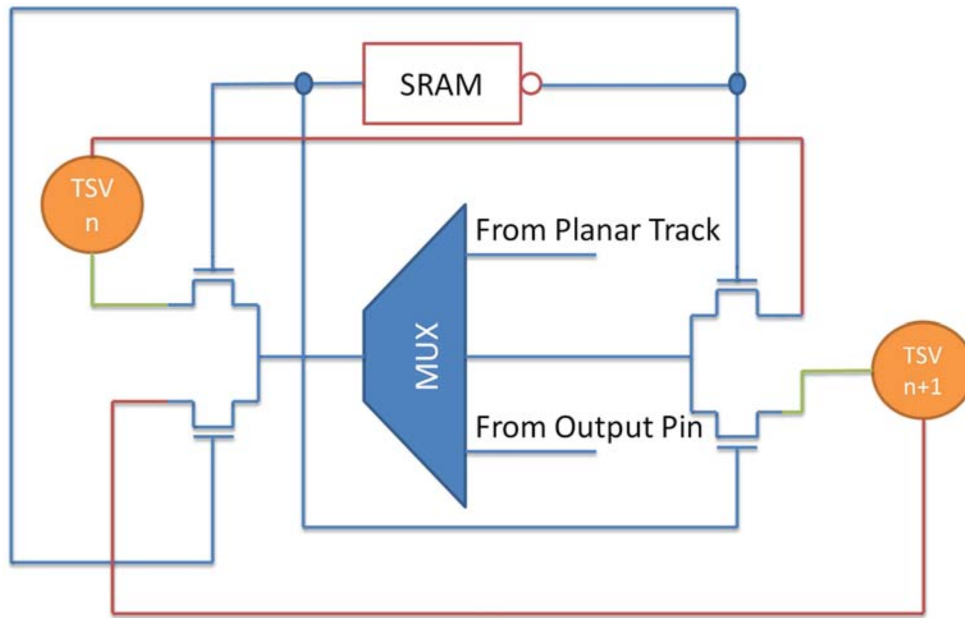


Figure 3.10: "Uni-Bi" Switch

architecture as shown in Figure 3.10.

We add 10 more transistors per multiplexer for the vertical track, including 4 pass transistors and one SRAM (comprised of another 6 transistors) assuming the output of the multiplexer has a buffer. We call it "Uni-Bi" switch because this switch acts as a uni-directional multiplexer switch but it is configured from a bi-directional switch. For example, at a random layer n, TSV n and TSV $n+1$ have the same track number but in adjacent segments, n and $n+1$. We define from n to $n+1$ as BELOW direction and from $n+1$ to n as ABOVE direction. By assigning "1" to the SRAM, the path in green color is turned on and the path in the red color is off. The

track acts as an ABOVE direction track. By assigning "0" to the SRAM, it acts as a BELOW direction track. As a result, we can change the track direction according to the actual situation. Unlike the bidirectional architecture in which each track direction is decided at the routing stage, we decide the track direction right after the placement stage based on the placement results.

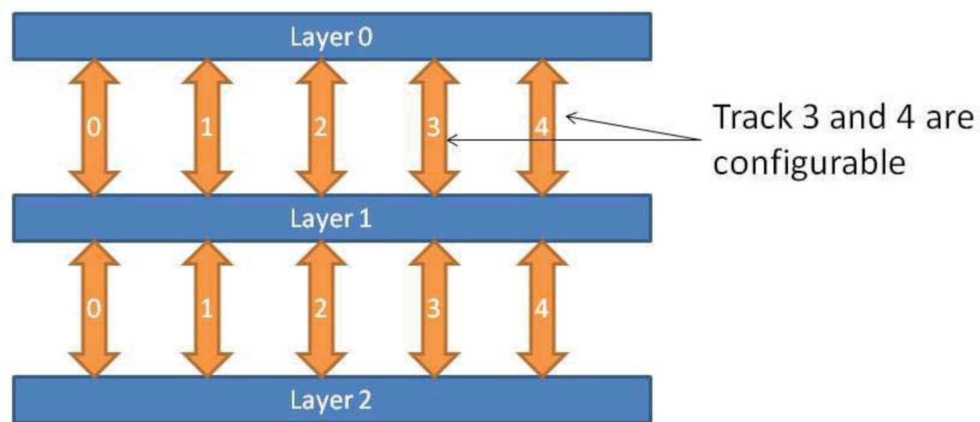


Figure 3.11: Schematic of UNI-3D with UB Switch

By using UB switch in uni-directional architecture, the vertical channel width and the respective channel capacity assignment can be decided by the best optimized placement results on the vertical width. The schematic of the new architecture is shown in Figure 3.11. We can find that, instead of fixed direction of the signal that the vertical track can carry, the new architecture allows the direction to be configurable.

4 | 3D FPGA EDA

In this chapter, we present our 3D EDA software which is developed from VPR 5 and HotSpot. Compared with previous work on the EDA development, Our EDA software has two contributions. The first contribution is that our software is able to place and route the circuits on the 3D FPGA for uni-directional routing based 3D FPGA. Furthermore, we propose a new placement algorithm which caters for the uni-directional routing based 3D architecture because the previous placement algorithm for bidirectional routing based 3D architecture in [44, 18] is not suitable for the uni-directional architecture. The second contribution is that we add the thermal placement algorithm to alleviate the thermal side effect to 3D architecture.

4.1 General Flow

Our EDA flow is similar to the VPR power model flow with our modification on the VPR 5 for the 3D FPGA architecture as shown in Figure 4.1. Compared with Figure 2.11, the switching activity file is used in the placement stage for early power estimation since the thermal calculation requires the power consumption at each part of the chip. The thermal model is implemented by referring the HotSpot framework. The thermal configuration file is added to build the thermal model. Furthermore, the thermal aware placement can be disabled and enabled at source code level so that we can also conduct the placement without thermal aware consideration.

We have implemented two kinds of placement. The first one is the ordinary placement which is used for the 3D FPGA placement without thermal cost consideration. The implementation is similar to [18]. We change the 2D grid array to the 3D grid array by adding the vertical direction to the coordinate. In the ordinary placement, we use critical path delay, TSV placement cost, and 2D bounding box as the circuit placement cost which is similar to [44] for the bidirectional 3D architecture placement. For the 3D uni-directional placement, we have implemented a new cost function which will be introduced in the later section.

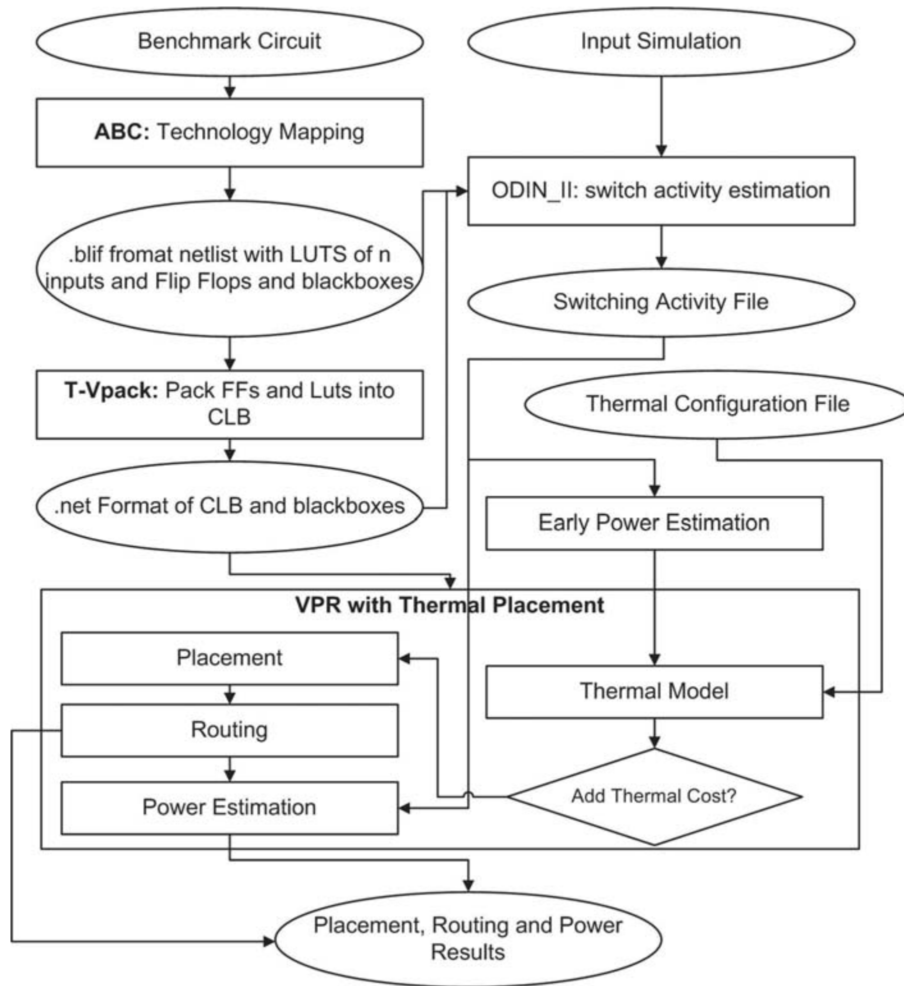


Figure 4.1: EDA Flow

The second is the thermal aware placement. The thermal placement is implemented by referring and combining two programs from the HotSpot framework. For the first step, we use ODIN II to calculate the switching activity from the input simulation and then we modify the VPR power model such that we can estimate power consumption of each CLB with the nets associated with it. Based on power profile estimation, we can further

obtain the thermal profile we want for thermal cost calculation.

4.2 Uni-directional Placement Algorithm

In this section, we introduce a new placement algorithm improved from VPR SA algorithm to optimize the TSV usage as well as the vertical channel width in the 3D uni-directional architecture. Based on our studies, the previous SA algorithm, such as [44, 18], for 3D bidirectional architecture is not suitable for the uni-directional architecture because the algorithm does not consider the characteristic of uni-directional architecture that the channel capacity is evenly distributed to the ABOVE direction and the BELOW direction. As a result, it wastes vertical channel capacity if nets propagate in the BELOW direction are more than the nets propagate in the ABOVE direction along the vertical channel and vice versa.

Therefore, the uni-directional routing based 3D architecture should have a new placement algorithm to ensure the vertical channel capacity can be used efficiently such that the demand for TSV is as low as possible with less penalty on circuit delay or total wire length. For the uni-directional routing based 3D architecture, we have modify the placement algorithm to cater it and the result from Chapter 5 shows that the algorithm can achieve

our optimization goal of TSV reduction.

4.2.1 Placement Algorithm for UNI-3D

Considering UNI-3D architecture, if we can place the circuit in a way such that the number of signals propagating in the ABOVE and BELOW direction could be distributed more evenly in each segment along the vertical channel, it may reduce the vertical channel width. In the previous study, the SA based placement algorithm is used to place the circuit in the 3D FPGA to find a well trade off point among the circuit speed, bounding box, and net spanning in the vertical direction. As shown in Equation (4.1) (from [44]), C_{timing} is the critical path delay cost, $C_{cong-2D}$ is the FPGA sum of all 2D bounding box cost of each net and C_{spanz} is the sum of the each net cost spanning in the vertical direction. α, β, and γ are respectively the weights of each cost towards the total cost.

$$C_{3D} = \alpha C_{timing} + \beta C_{cong-2D} + \gamma C_{spanz} \tag{4.1}$$

However, C_{spanz} aims to reduce the total number of interlayer nets or the number of layers that the net can span. Without controlling the specific

vertical segments in which a net can span and corresponding propagation direction, it can lead to several uncertainties in the UNI-3D architecture as summarized below:

1. At certain vertical segment, the total number of interlayer nets may far exceed the number of interlayer nets spanning in other segment. It may lead to the TSV waste at the segment which has fewer number of nets spanning.

2. At certain vertical segment, the number of nets in the ABOVE direction may far exceed the number of nets in the BELOW direction and vice versa. The vertical channel capacity must satisfy largest number of the nets spanning in the ABOVE direction. This may also lead to low TSV usage, depending on the gap between the number of nets spanning in the ABOVE and the BELOW direction.

These two uncertainties can be explained in Figure 4.2 for example. The solid arrow represents the usage of the channel capacity and the dotted arrow is the the channel capacity assigned to each direction. Suppose using Equation 4.1 as the cost function can get the following placement result. We can find that there are more nets spanning in the Segment 2 more than in Segment 1 and Segment 3. The vertical channel width must satisfy the

largest number of nets spanning in Segment 2. For some other circuits, it may be Segment 1 or other segment but we don't know as C_{spanz} do not control net spanning in each vertical segment. This is applied to both the bi-directional and uni-directional 3D architecture. It is the first uncertainty that Equation 4.1 brings.

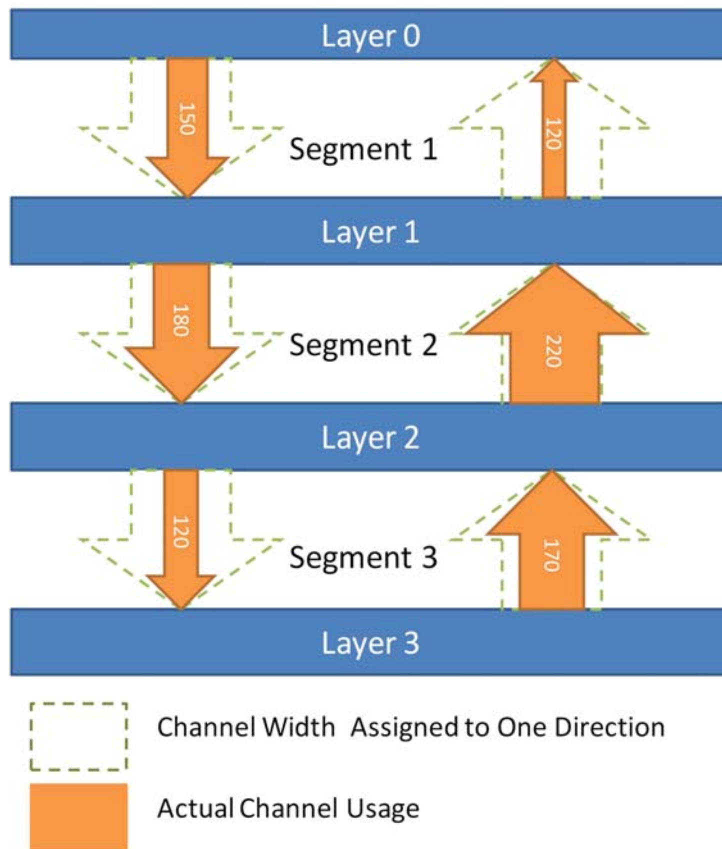


Figure 4.2: Example of Nets Spanning in the Vertical Direction by Using Cost Equation 4.1

On the other hand, if the placement algorithm assigns the vertical channel capacity to the ABOVE and BELOW direction not equally, it may also

cause the TSV usage inefficient. In Segment 2, we can find that there are more nets in the ABOVE direction than in BELOW direction and it cause the waste of the channel capacity in the BELOW direction. This is the second uncertainty if we use Equation 4.1 as the cost function for the uni-directional 3D architecture placement.

Therefore, we need to revise the SA algorithm to reduce the TSV demand. Equation 4.1 is modified to implement the balance control mechanism in our algorithm. We remove C_{spanz} since it only controls the individual nets spanning in the vertical direction without adjusting the number of nets travelling in the ABOE and BELOW directions. We add a new cost called C_{uni} which controls nets spanning in the vertical direction regarding to the number of nets at each segment and their respective propagation direction. The revised cost function is expressed as follows,

$$C_{total} = \alpha C_{timing} + \beta C_{2D_BB} + \gamma C_{uni} \tag{4.2}$$

where α, β, and γ are the respective weights of each cost towards the total cost. C_{timing} and C_{2D_BB} is the same as C_{timing} and $C_{cong-2D}$ in Equation 4.1 respectively. Furthermore, C_{uni} is the largest number of nets travelling in the same direction among all vertical segment. Equation 4.2 is called

the balance cost function. Similar to Equation 4.1, it represents the trade off among delay, wire length, and the net spanning in the vertical direction with additional concern on the TSV usage for the uni-directional routing based 3D architecture. The modified TSV aware placement algorithm is shown in Figure 4.3.

By applying the balance cost function, the number of nets travelling in ABOVE or BELOW direction is assigned to almost equal to each other. Since C_{uni} is the largest number of nets travelling in one direction among all vertical segment, the number of nets travelling in each segment is also balanced to almost equal to each other. As a result, the new cost function can achieve the purpose to reduce the vertical channel width through this balancing mechanism. Accordingly, we call this placement algorithm as "even number", because the vertical channel width is even number based on the conventional uni-directional routing architecture. Regarding to the run time, the simulated annealing is highly based the cooling scheduler, if the scheduler tends to cool down slowly, the run time can be very long with the large size of the circuit placement. For the individual cost computation, it does not involve any complex computation, $O(n)$ can be used to represent the computation complexity, depending on the number of affected nets in each movement.

$S = RamdomPlacement();$

$T = InitialTemperature();$

$R_{limit} = InitialR_{limit}$ (for planar bounding box only);

while $ExitCirterion() == False$ **do** ▷ Outer loop

while $InnerLoopCriterion() == False$ **do** ▷ Inner loop

$S_{new} = GenerateViaMove(S, R_{limit});$

Calculate $\triangle C_{uni}, \triangle C_{timing}, \triangle C_{2D_BB}$

$\triangle C = \alpha \triangle C_{timing} + \beta \triangle C_{2D_BB} + \gamma \triangle C_{uni}$

$r = random(0, 1);$

if $r < e^{-\triangle C/T}$ **then**

$S = S_{new};$

end if

$T = UpdateTemp();$

$R_{limit} = UpdateR_{limit}();$

end while ▷ Inner loop

end while ▷ End Outer loop

Figure 4.3: Modified TSV Aware Placement Algorithm for Uni-directional Routing Based 3D FPGA

4.2.2 Placement Algorithm UNI-3D with UB Switch

On the other hand, sometimes, the placement algorithm we propose in the Section 4.2.1 may also waste vertical track resource due to the fact which is

shown in Figure 3.9a. Therefore we propose a more flexible vertical capacity assignment architecture by involving UB switch described in Section 3.2.3. Accordingly, we need to develop the placement algorithm accordingly to achieve our optimization goal for TSV reduction as shown in Figure 3.9b. This is to be achieved by modifying C_{uni} such that the number of nets travelling in the BELOW (S_{down}) and the ABOVE (S_{up}) direction are in a controlled ratio as shown in Equation 4.3 where η is the ratio. The new cost computation function allows more interlayer nets travelling in the BELOW direction than in the ABOVE direction, based on the ratio, η.

$$C_{uni} = \begin{cases} S_{up} & \text{if } S_{down} <= \eta S_{up} \\ (S_{up} + (S_{down} - \eta S_{up})) & \text{if } S_{down} > \eta S_{up} \end{cases} \tag{4.3}$$

Accordingly, we also need to modify the EDA flow as shown in Figure 4.4. We name this EDA flow as "place twice". As indicated by the name of this EDA flow, we conduct two times placements. The first placement uses the same placement algorithm as “even number”. If the results show that there is possibility to reduce vertical channel width, it will conduct a second placement. In the second placement, C_{uni} is modified such that $S_{down_max} : S_{up_max} = 1.3$ (It is logically equivalent to using $S_{up_max} : S_{down_max} = 1.3$. The ratio can be altered and 1.3 is for demonstration

purpose.) to adjust signal propagation directions, allowing more signals travelling in the BELOW direction and reducing the number of signals travelling in ABOVE direction. If this placement result can fit into the vertical channel capacity which reduces the assignment to the ABOVE direction and maintains the assignment for the BELOW direction, then the program will proceed to route the circuit with the second placement result. Otherwise, the program will use the first placement result for routing.

4.3 Thermal Placement

In the 3D IC design, thermal issue becomes critical and needs to be resolved because stacking IC dramatically increases the power density and hence the chip temperature. Properly placing the circuit can alleviate the thermal effect and remove the hot spot while with the reasonable delay trade off. In this section, we present our thermal placement algorithm framework.

In order to add the thermal cost in the placement algorithm, we need to know the thermal profile which can be acquired from thermal model. Furthermore, thermal model requires the input of the power model and physical model for its calculation as shown in Figure 4.5. Once we can get the highest temperature of the chip from the thermal profile, it can

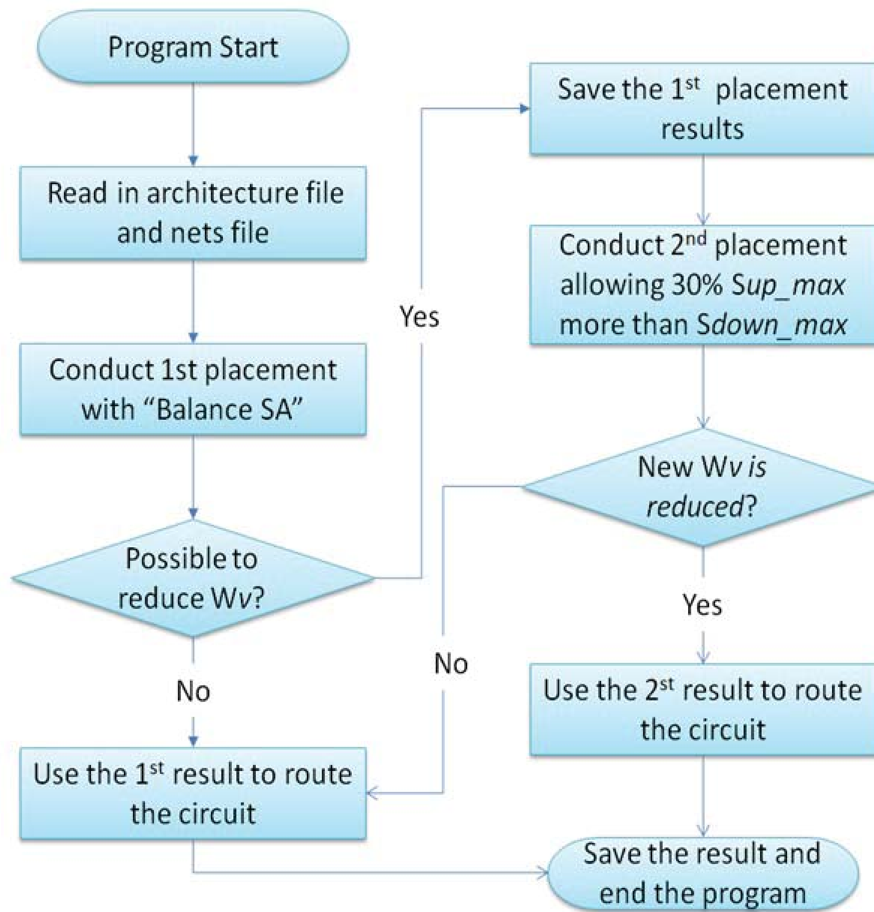


Figure 4.4: Modified EDA Flow for Further TSV Reduction

be used as the thermal cost in the SA cost function. Equation 4.4 is one of example for adding thermal cost in the uni-directional cost function we propose in Section 4.2.2. It represents the trade off among four placement costs, delay, wire length, number of nets spanning in the vertical direction, and the thermal cost of the chip.

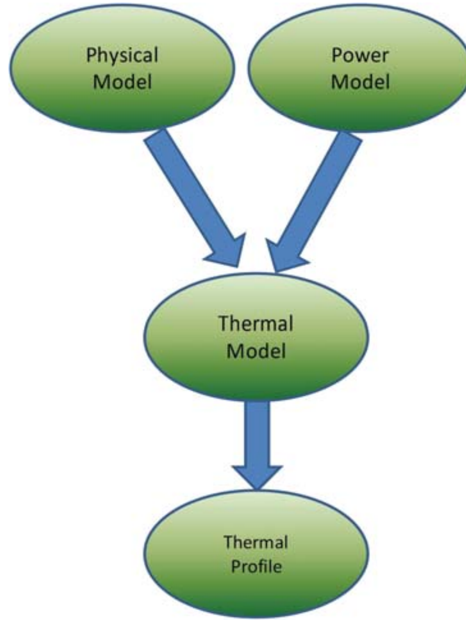


Figure 4.5: Thermal Profile Calculation Flow

$$C_{total} = \alpha C_{timing} + \beta C_{2D_BB} + \gamma C_{uni} + \theta C_{thermal} \tag{4.4}$$

where $C_{thermal}$ is the thermal cost and θ is the weight of the thermal cost in the cost function. Accordingly, the placement algorithm further involves the thermal model construction and calculation as shown in Figure 4.6.

4.3.1 Physical Model

The thermal flow is analogous to the electrical current flow. This relationship can be expressed by Equation 4.5 and Equation 4.6.

$S = RamdomPlacement();$

$T = InitialTemperature();$

$R_{limit} = InitialR_{limit}$ (for planar bounding box only);

Build Thermal RC Matrix Equations;

while $ExitCirterion() == False$ **do** ▷ Outer loop

  while $InnerLoopCriterion() == False$ **do** ▷ Inner loop

    $S_{new} = GenerateViaMove(S, R_{limit});$

    Calculate $\triangle C_{uni}, \triangle C_{timing}, \triangle C_{2D_BB}, \triangle C_{thermal}$

    $\triangle C = \alpha \triangle C_{timing} + \beta \triangle C_{2D_BB} + \gamma \triangle C_{uni} + \theta \triangle C_{thermal}$

    $r = random(0, 1);$

    if $r < e^{-\triangle C/T}$ **then**

      $S = S_{new};$

    end if

    $T = UpdateTemp();$

    $R_{limit} = UpdateR_{limit}();$

  end while ▷ Inner loop

end while ▷ End Outer loop

Figure 4.6: Thermal Aware Placement Algorithm (Also Considering TSV Reduction)

$$V_{diff} = IR, \quad I(t) = C\frac{dV(t)}{dt} \tag{4.5}$$

$$T_{diff} = QR_{th}, \quad Q(t) = C_{th}\frac{dT(t)}{dt} \tag{4.6}$$

where V_{diff} is the potential difference and T_{diff} is the temperature difference. I is the current and Q is the power. R and C are the electrical resistance and capacitance. Accordingly, R_{th} and C_{th} are the thermal resistance and capacitance respectively. By constructing the thermal "RC" network, we can calculate the thermal profile through the similar method for solving the electrical current flow.

In the previous works on thermal profile, compact thermal model is widely used to construct the thermal "RC" network for thermal modelling. In this thesis, we also use this model for the thermal modelling. Referring the work in [52], we use the HotSpot to construct 3D FPGA's thermal "RC" network in our 3D architecture. In additional, we define thermal tile which is similar to the 2D FPGA tile defined in [1], as the basic thermal block in the thermal calculation.

Figure 4.7 is one layer of 3D FPGA. For clarification purpose, we use one

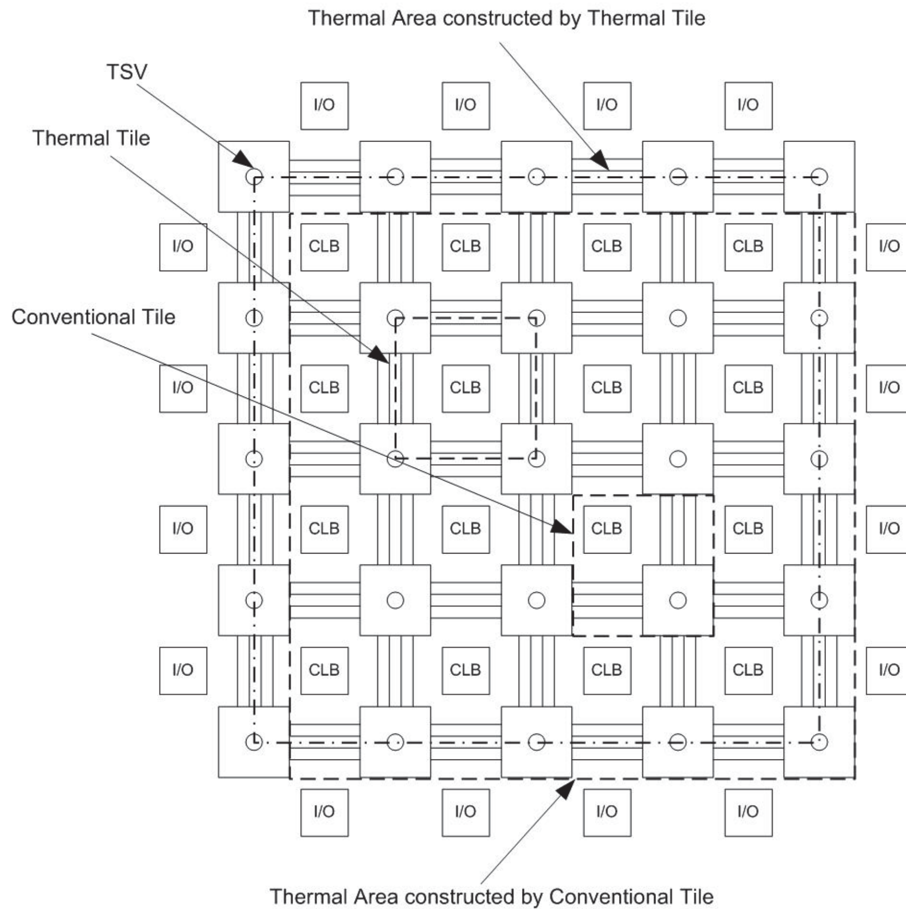


Figure 4.7: Thermal Tile and Thermal Area

circle to represent all TSVs that connect to the respective 3D SB. A thermal tile is defined as the chip area which contains a CLB and routing area surrounding it. Each SB adjacent to a CLB contributes one quarter of its area to the thermal tile. Compared with the conventional tile defined in [1] whose definition focuses on the components and repeatable in each tile, the thermal tile's definition focuses more on constructing homogeneous thermal resistor for each block to simplifying the calculation. From the

area perspective, the area of a thermal tile is the same as the area of a conventional tile.

We use the lumped model to calculate the thermal resistance of the each thermal tile in both vertical and horizontal direction and we assume that the thermal resistance is homogeneous across each die which is shown in Figure 4.7. Although in this work we do not explicitly shows the thermal resistance differences caused by TSV which is mainly composed by copper, we think our model is a valid assumption. It is because, in our 3D architecture, the TSV is homogeneously distributed among each SB, and we use a repeatable thermal tile as the grid to calculate the thermal profile so that the lumped resistance in each thermal tile should be the same.

We exclude the the area of the 4 edges of FPGA which is the parts outside the dotted-line square in Figure 4.7. It is because as the size of the FPGA grows, the weight of those areas becomes less significant. In addition, power density at those areas is not high compared with the power density at the central part. For the actual 3D layout, the I/O pad may not be allocated in every layer. Due to above reasons, we think exclusion of those areas only cause little impact on the result accuracy.

In the vertical direction, there are several thermal conductive layers, die

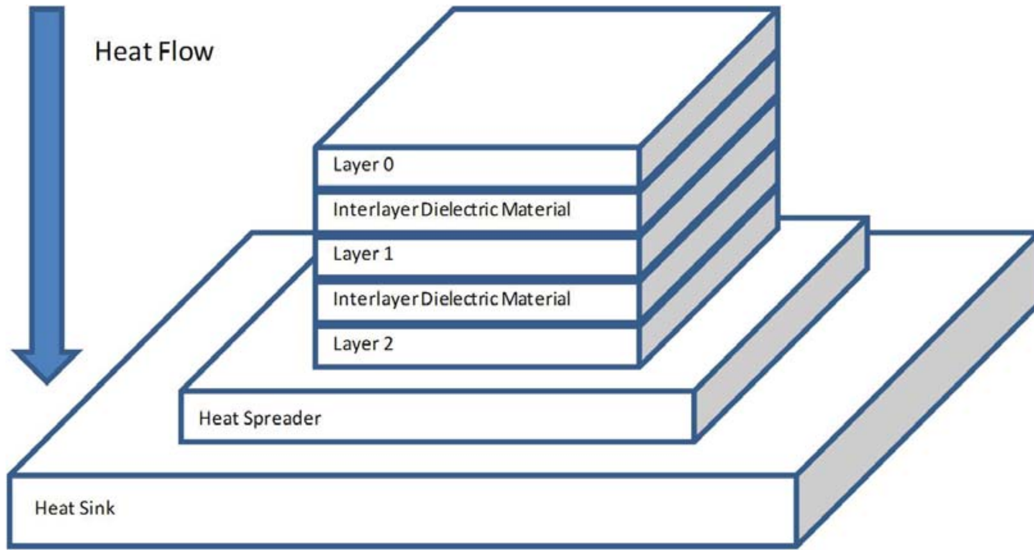


Figure 4.8: Heat Flow and Conductive Layers

layer, dielectric material layer between two adjacent die layers, heat spreader layer and heat sink layer. Figure 4.8 is a three-die stacked 3D FPGA for example. It is composed of 6 thermal conductive layers, three silicon layers (Layer 0, Layer 1, and Layer 2), two interlayer dielectric material layers, one heat spread layer, and one heat sink layer. If there are more dies stacked, additional layers should be added according to the example of Figure 4.8 for the die layer and interlayer dielectric material layer. For a N_{layer} stacked 3D FPGA, the number of thermal conductive layer is based the following equation,

$$N_{thermal} = 2 + 2 \times N_{layer} - 1 \tag{4.7}$$

where, $N_{thermal}$ is the number of thermal conductive layer. In this work, we assume that the major heat flow happens in the vertical direction indicated by the heat flow arrow in Figure 4.8. It is because chip is usually packed with dielectric material whose thermal resistivity is also high and the thermal resistance in the lateral direction is usually higher than the vertical direction. We can easily conclude it by considering the cross area and the following equation for thermal resistance calculation:

$$R_{th} = t/(k \times A) \tag{4.8}$$

where k is thermal resistivity. A is the cross section and t is the length. Similar to the calculation of the electrical resistor, the thermal resistance decreases with the increment of the area of the cross section and increases with the increment of the length. Figure 4.9 shows a thermal block whose dimension is $2a \times 2a \times b$ (b is a few times smaller than $2a$). Suppose the center of the top surface is the base point and we apply the finite difference method. Then the lateral thermal resistance due to the base point is $\frac{1}{2bk}$ and the vertical thermal resistance due to the base point is $\frac{b}{4a^2k}$. As a result, we can find that the lateral thermal resistance is larger than the vertical resistance and the heat flow is mostly towards the direction with less thermal resistance. In addition, length of a chip die is much longer

than its thickness. Based on above reasons, it is valid to assume that there is no heat exchange between the thermal layers we define and the chip package material in the lateral direction. However, we do consider the heat exchange in the lateral direction among each thermal tile.

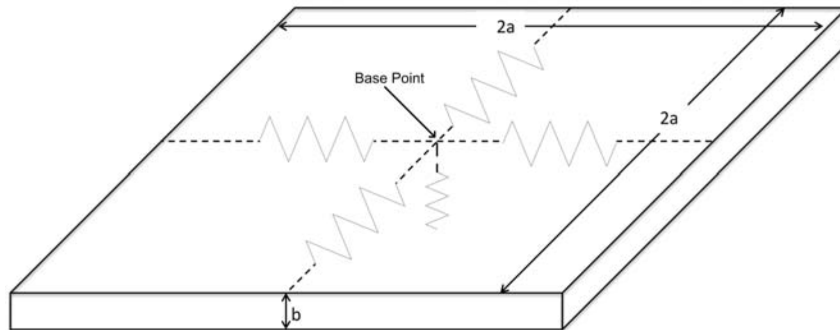


Figure 4.9: Thermal Block

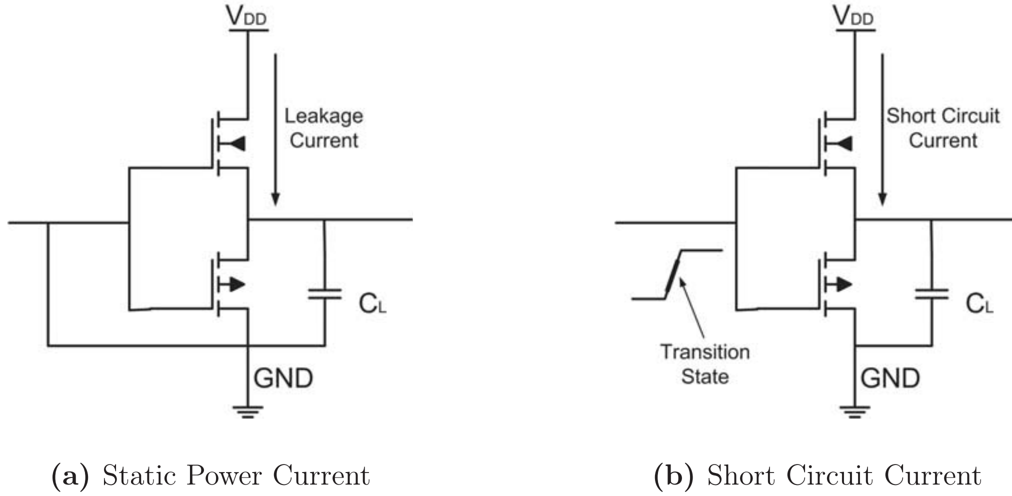
4.3.2 Power Model

The power modelling is important for the thermal modelling. It is because the temperature variation is, in fact, caused by the different power density at each part. Since we use the thermal tile as the unit for thermal "RC" network construction, we calculate the power based on each thermal tile.

In either 2D or 3D chip, the power consumption can be categorized into three parts, dynamic power, static power and short circuit power. The dynamic power is generated by charging and discharging the capacitors in

the circuit. It can be expressed as the following equation,

$$P_{dyn} = C_L V_{DD}^2 f_{0\to1} \tag{4.9}$$



(a) Static Power Current

(b) Short Circuit Current

Figure 4.10: Example of Static Power and Short Circuit Power

where P_{dyn} is the dynamic power consumption, C_L is the load capacitance of the device output which is contributed from the parasitic capacitance of the input/output of the gate and metal wire. V_{DD} is the voltage applied to the circuit. $f_{0\to1}$ is the dynamic power transition frequency at each device node, such as input or output of a gate. Due to the delay of the metal wire and the gate delay, the dynamic power transition frequency at each node may not be the same. In addition, the dynamic power transition frequency is also called the switching activity which is computed by ODIN

II as shown in Figure 4.1. The static power is also called leakage power, which is caused by a small amount of current flow from V_{DD} to GND while there is no switching activities as shown in Figure 4.10a. The short circuit power is caused also by the current flow from V_{DD} to GND. It happens in the transition state of high-to-low or low-to-high in which both NMOS and PMOS are not completely turned on or off and causes the current flow from V_{DD} to GND as shown in Figure 4.10b. The detailed calculation of the three types of power is from the VPR power model [35, 22].

From the aspect of the FPGA components, the power of each thermal tile can be classified as the CLB power and routing power. The CLB power is relatively easy to be computed. The most difficult power estimation is the routing power. It is because how each net routes in the FPGA can only be decided after the routing. During the placement, the routing power can only be estimated, though it is not accurate.

To calculate the net power, we firstly define the shortest distance between the source and the sink. The shortest distance is defined as the sum of coordinate difference in the absolute value form along each axis in Cartesian coordinate system, as shown in Equation 4.10.

$$D_{shortest} = |\Delta x| + |\Delta y| + |\Delta z| \tag{4.10}$$

where $D_{shortest}$ is the shortest distance between source and sink. Δx, Δy, and Δz are the difference of the source and sink along x, y, and z direction respectively. Follow the shortest path between the source to its respectively individual sink, we can calculate the corresponding power consumption. The net power is redistributed in a way such that for each source to its corresponding sink path, we assign half of power to the thermal tile where the source is located and the other half to the thermal tile where the sink is located. If a source has multiple sinks. Then, we sum each half power between the source and each sink. Although this calculation for the net power is inaccurate, we think this way can generally indicate the power distribution in the 3D chip. For example, if the source has multiple sinks, it has multiple paths in order to distribute the signal to each sinks. As a result, the net power density around this source should be higher than the source with only one or two sinks. The power consumption in each thermal tile can be calculated by summing up the power consumption in the routing area and the CLB in the thermal tile.

4.3.3 Thermal Model

The thermal model is referred from HotFloorplan and HotSpot 3D thermal model which are part of HotSpot framework. The HotFloorplan is used for the thermal aware 2D floor planning and HotSpot 3D thermal model is used to estimate the temperature variation for the 3D IC. Considering the Hotfloorplan uses SA based algorithm for the thermal aware floor planning. We use the HotFloorplan as the base and add the layer structure of HotSpot 3D thermal model.

Different from the HotFloorplan, each block size, the size of thermal tile, in our model is much smaller than the block size used for floor planning. In addition, the dimension of each block is fixed and we assume that each thermal tile is a square with identical width.

In order to average the temperature, we use the highest temperature as the thermal cost. For each block movement in the 3D FPGA placement, we update the estimated power consumption in each thermal tile and then calculate the temperature profile through the thermal model. The cost function is Equation 4.4.

However, one drawback is that the time consumed by the thermal aware

placement is much longer than the placement without the thermal aware. Besides of possible long run time from the SA itself, the computation complexity for the thermal cost is $O(n^2)$. As a result, in each movement, the cost function complexity becomes $O(n^2)$. It is much slower than the cost function computation without thermal cost. We will improve the speed in the future work.

5 | Design Space Exploration

In this chapter, we present our finding on the uni-directional routing based 3D FPGA, the placement algorithm, and thermal evaluation by using EDA software we develop. The architecture parameters we extract are based on $65nm$ process technology. In section 5.1, we use different CB flexibility of the input to investigate the variation of delay and area relationship. In section 5.2, we evaluate the placement algorithm we develop with different cost weight on C_{uni}. In Section 5.3, we evaluate the architecture with different 3D SB and the architecture with UB switch. In section 5.4, we evaluate the thermal placement algorithm and the corresponding thermal profile.

5.1 Architecture Exploration on Input CB Flexibility, C_{in}

In the previous study on the 3D bidirectional architecture, the number of stacked layers associated with the delay reduction has already been studied. Because the uni-directional routing based 3D architecture is similar to the bidirectional routing based 3D architecture. Therefore the bidirectional and the uni-directional routing based 3D FPGA architecture should have similar delay reduction associated with the number of layer stacked. Although some variation may come from the differences of the routing topology and placement algorithm, the tendency should be similar to the observation from [18]. Yet none of previous works reveals delay, area, and channel width change associated with the change of CB flexibility which is an important factor of the variation among area, delay and routability. In this section, we explore how the CB flexibility varies area and delay in the uni-directional based routing architecture.

Regarding to the experiment setup, we use the uni-directional routing based 3D FPGA with $N_{layer} = 5$, $n = 4$ and unit segment length as the base 3D FPGA. We also compare with the 2D uni-directional routing based baseline

FPGA. The modelling is under 65nm process technology and we assume that a TSV costs 16 μm^2 chip area.

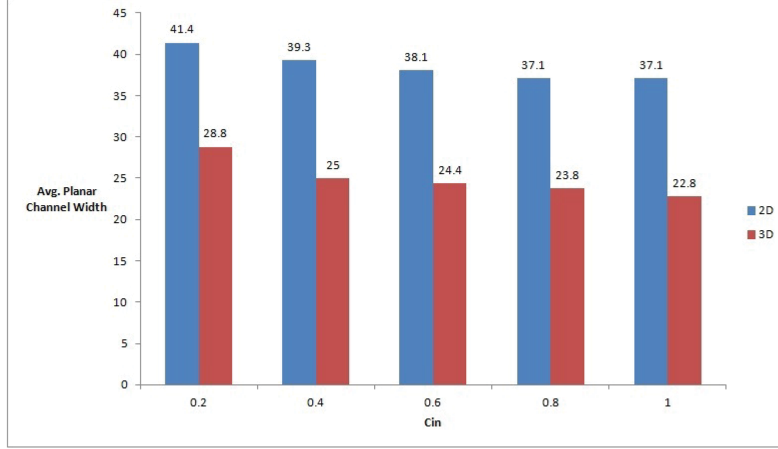


Figure 5.1: Average Planar Channel Width Variation Due to C_{in}

To evaluate C_{in}, we fix C_{out} to be 1 and increase C_{in} from 0.2 to 1 with 0.2 increment each time for both 2D and 3D architecture. In addition for the 3D architecture, we use 3D-SB-2.

Figure 5.1 is the average planar channel width variation for both 2D baseline FPGA and 3D FPGA. We can see that the channel width for 2D FPGA reduces almost linearly with the C_{in} increasing from 0.2 to 1. In addition, the channel width of $C_{in} = 1$ uses about 90% of the channel width while $C_{in} = 0.2$. Similar relationship also holds for the 3D architecture. We can see that the widest channel width happens while $C_{in} = 0.2$. Compared with $C_{in} = 0.2$, the average channel width is reduced about 20% when $C_{in} = 1$.

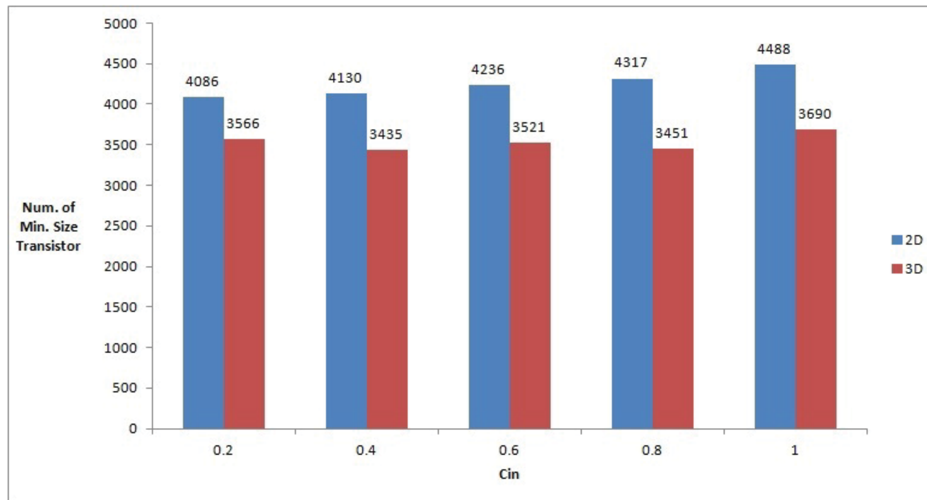


Figure 5.2: Average Routing Area Due to C_{in}

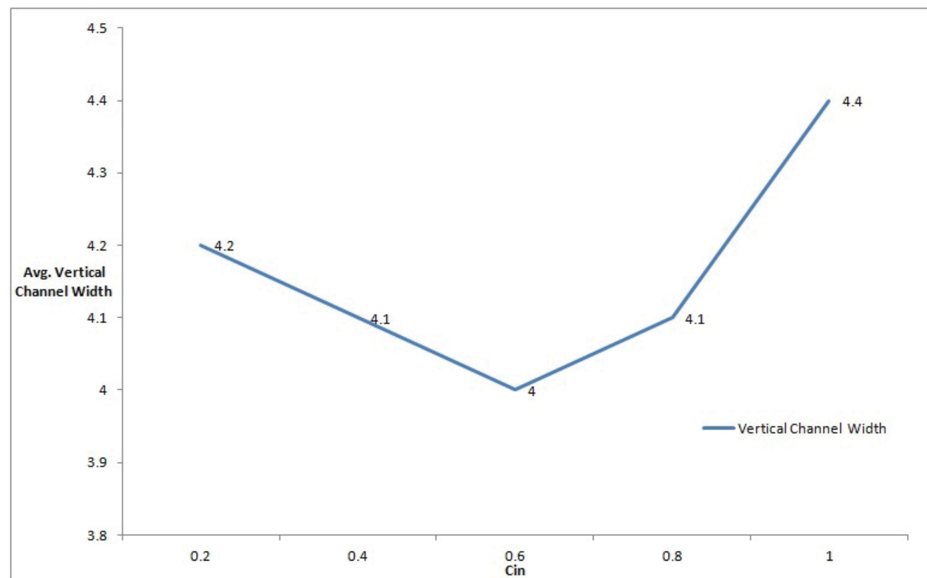


Figure 5.3: Average TSV Used Per Tile Due to C_{in}

On the other hand, Figure 5.2 shows the variation of routing area in a tile translated into the number of minimum size transistors. We exclude the

area of the CLB since both 2D and 3D FPGA architecture uses the same CLB architecture so that it can be more clear to see the changes in the routing area. We can find that, for the 2D baseline FPGA, the routing area increases according to the increase of C_{in}. For $C_{in} = 1$, it requires about 10% more routing area than the architecture of $C_{in} = 0.2$. The routing area of 3D architecture does not increase with the increase of C_{in}.

Figure 5.3 is the average vertical channel width (same as the number of TSV in a tile) required for routing the benchmark circuits. We can find that under the same cost function parameters, the required vertical channel width behaves like a second order curved and achieves its minimum value at $C_{in} = 0.6$. The highest average number of TSV per tile appears when $C_{in} = 1$. It is somehow beyond our expectation that the average number of TSV per tile should reach its minimum value when $C_{in} = 1$. From Figure 5.3, we can preliminarily conclude that under different 3D FPGA architecture parameters, it requires corresponding cost function and parameters to be adjusted in order to achieve fewer TSV demand since it is favorable for the physical layout and logic density improvement.

Figure 5.4 shows the both 2D and 3D FPGA delay vary according to C_{in}. We can find that there is no much difference on the delay improvement with the increase of inputs flexibility, except for the 3D architecture when

$C_{in} = 1$. For $C_{in} = 1$, we can see that there is the obvious delay drops in the 3D architecture. Compared with its 2D counterpart, the delay can be reduced by 31.8%.

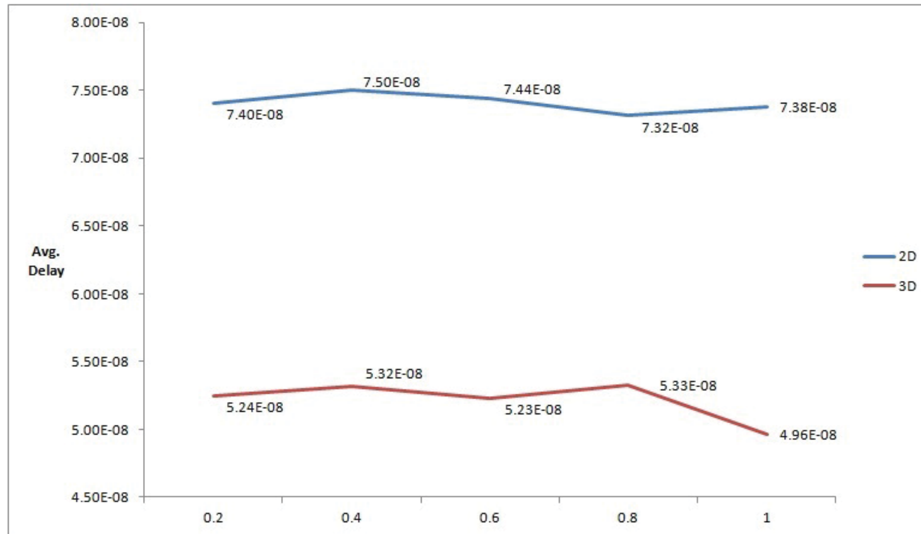


Figure 5.4: Average Delay Due to C_{in}

Figure 5.5 shows Routing Area Delay Product (RADP). We normalize all the data to the value of 2D architecture while $C_{in} = 0.2$. Since in previous studies, the conclusion of routing area is dominant is based on the evaluation of commercial FPGA which has very large channel width, this conclusion may not be held in our evaluation because the channel width we use is around the minimum routable channel width for different benchmark circuits.. Therefore adding the area of CLB may not be convenient for us to observe the changes due to routing area and we exclude the area of CLB

and focus more on the routing area. Based on the data shown in Figure 5.5, we can see that the RADP value achieves its minimum, 0.6, for the 3D architecture while $C_{in} = 0.4$. But the variation for the 3D architecture is not so obvious for 3D architecture compared with 2D architecture. For the 2D architecture, RADP value tends to increase with the C_{in}. In general, the RADP value of 2D baseline architecture is about $1.6 - 1.7$ times of the 3D architecture.

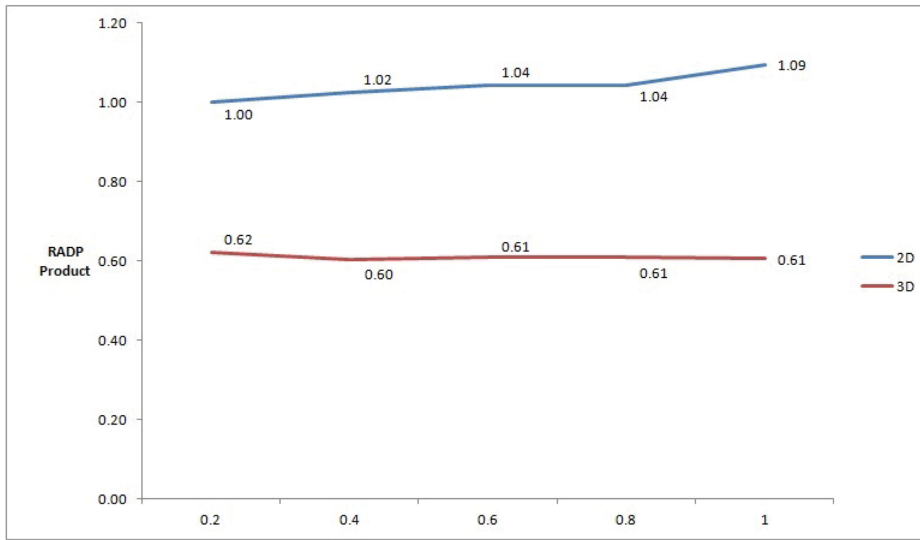


Figure 5.5: RADP Product Due to C_{in} (Norm to 2D Architecture while $C_{in} = 0.2$)

By crossing comparison among Figure 5.2, 5.4, and 5.5, we can expect that the ADP which considers the whole tile area would achieve its minimum value with $C_{in} = 1$ due to minimum value of the average delay.

5.2 Placement Algorithm Exploration on The Weight of C_{uni}

In this section, we look into effect caused by varying the weight of C_{uni} in the placement cost function. The weight of C_{uni}, γ, is varied from 0.2 to 1. At the same time, we keep the following conditions so that the sum of each weight is 1 and the cost weights of the delay and 2D bounding box are the same.

$$\alpha + \beta + \gamma = 1 \tag{5.1}$$

$$\alpha = \beta \tag{5.2}$$

At the 3D architecture side, we use a 5-layer 3D FPGA with $C_{in} = 0.2$, $C_{out} = 1$, 3D-SB-2 SB, and unit length segment.

Figure 5.6 shows the planar channel width varies according to the γ. From the graph, we can find that at $C_{uni} = 1$ the largest channel width appears due to the absence of bounding box controlling parameter C_{2D_BB}. Except

for the planar channel width when $\gamma = 1$, the next largest channel width happens at $\gamma = 0.4$. It is about 1.23 times of the smallest planar channel width at $\gamma = 0.6$. However, we cannot find any obvious trends of the planar channel width variation, except the result at $\gamma = 1$ when the weight of C_{2D_BB}, β, becomes 0 as well as the weight of C_{timing}, α.

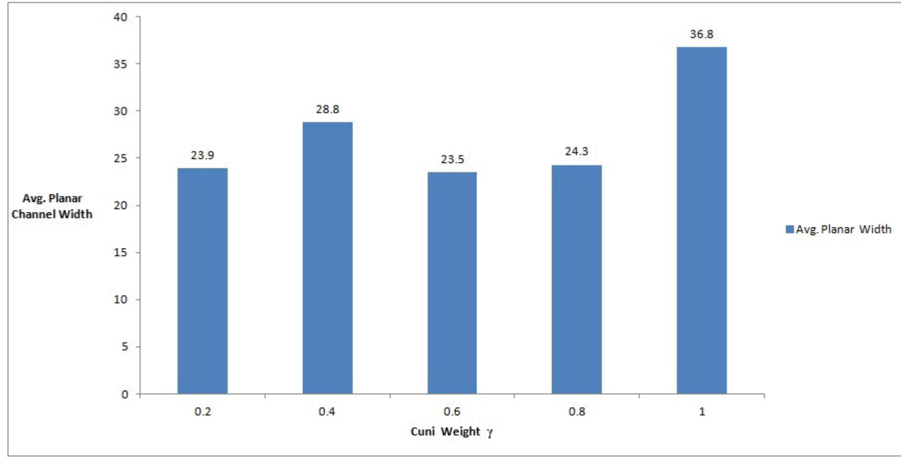


Figure 5.6: Average Planar Channel Width Variation Due to $\mathbf{C_{uni}}$ Weight γ

Accordingly, the routing area follows the similar trend shown in Figure 5.7. It is because the number of transistors in a tile is largely decided by the channel width and the average TSV used in a tile is not varied significantly at the same time as shown in Figure 5.8.

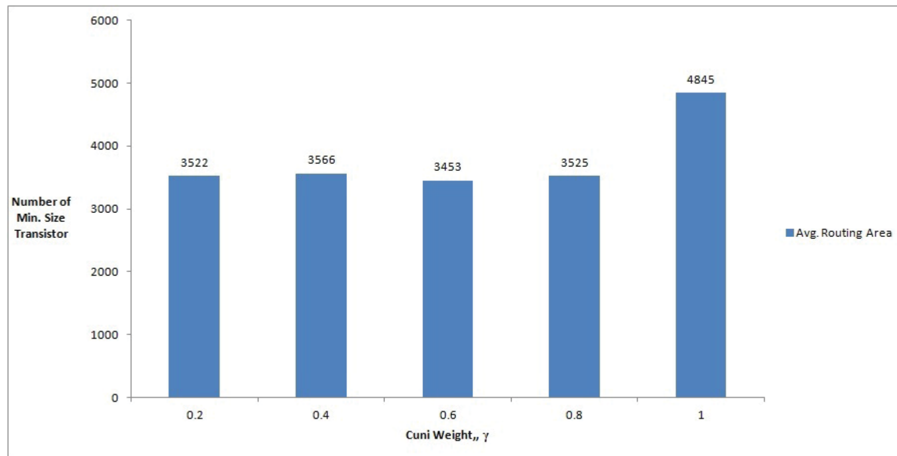


Figure 5.7: Average Routing Area Due to C_{uni} Weight γ

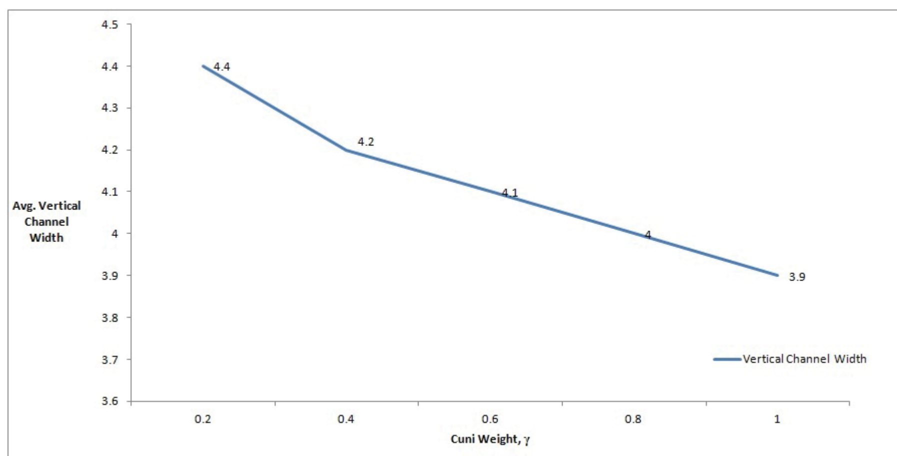


Figure 5.8: Average TSV Used Per Tile Due to C_{uni} Weight, γ

In Figure 5.8, we can see that the vertical channel width declines with increase of the γ. At $\gamma = 1$, the vertical channel width achieves its minimum which is about 12% fewer than the vertical channel width when $\gamma = 0.2$.

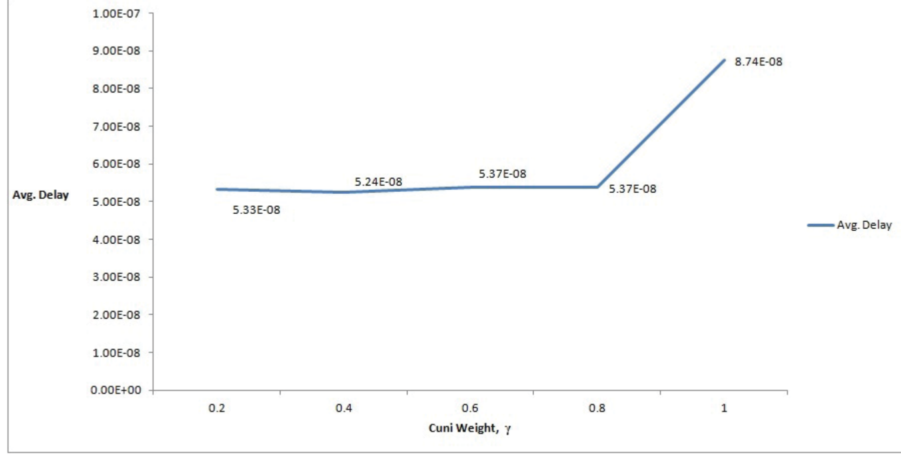


Figure 5.9: Average Delay Due to $\mathbf{C_{uni}}$ Weight, γ

In Figure 5.9, we can observe that the average delay variation does not changes much with increase of γ except at the points where $\alpha = 0, \beta = 0$. This is in the prediction since at $\gamma = 1$, there is no cost parameters to optimize the circuit delay. Compared with delay values in the Figure 5.4, the average delay at $\gamma = 1$ even exceeds the value of 2D baseline FPGA.

Figure 5.10 indicates the variation of RADP. We can find that, as far as the weights, α and β, are none zero, the RADP does not vary too much with different value of γ. As a result, we can conclude that the value of C_{uni} itself is much larger than the other two costs presented in Equation 4.2 such that the variation of the corresponding weight does not affect much on the RADP when the weights of the other two costs are none zero. With

C_{uni} presenting only in the cost function, it has poor critical path delay. Hence, C_{uni}'s effect is limited to regulate the vertical channel width. The cost of C_{timing} and C_{2D_BB} must be kept in the cost function in order to achieve better delay results.

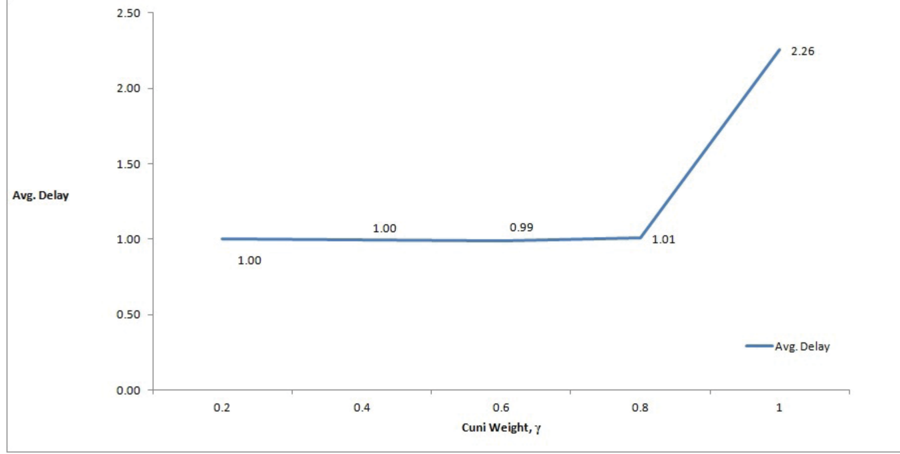


Figure 5.10: RADP Due to $\mathbf{C_{uni}}$ Weight, γ (Norm to 3D Architecture while $\gamma = 0.2$)

5.3 3D Architecture with UB Switch V.S. 3D Architecture without UB Switch

As introduced in Section 4.2.2 and 3.2.3, the direction of the UB switch is able to be configured so that the TSV can be used more efficiently without the constraint of predefined propagation direction. In this section, we con-

duct a preliminary study by making the compare between 3D architecture without UB switch and 3D architecture with UB switch.

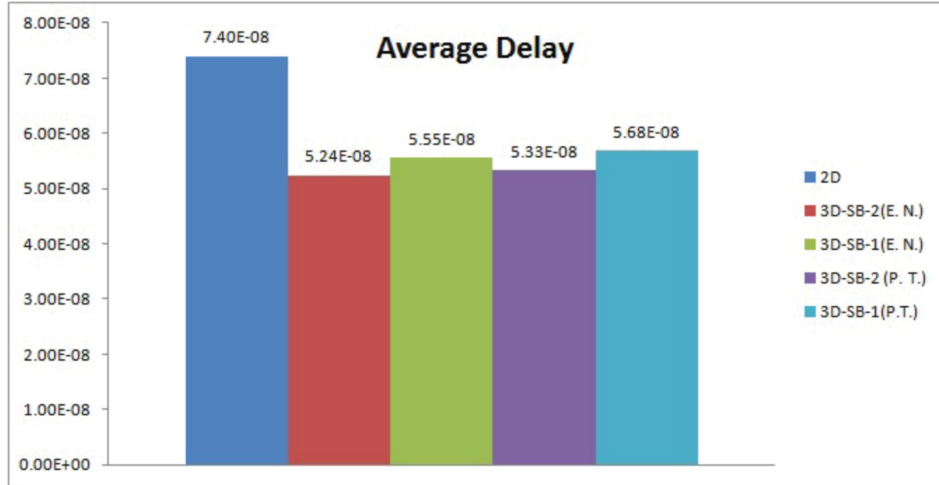


Figure 5.11: Average Delay

Figure 5.11 describes the average delay comparison among 2D baseline FPGA, 3D FPGA using SB 3D-SB-1 and 3D-SB-2 with "even number" placement, and 3D FPGA using 3D-SB-1 and 3D-SB-2 but with UB switch and "place twice" placement. As a result, we can observe that the 3D architecture shows significant delay reduction compared with the 2D baseline FPGA. However, the differences of delay among the 4 cases of 3D architecture are not significant. Similar trends of the average planar channel width can be observed from Figure 5.12.

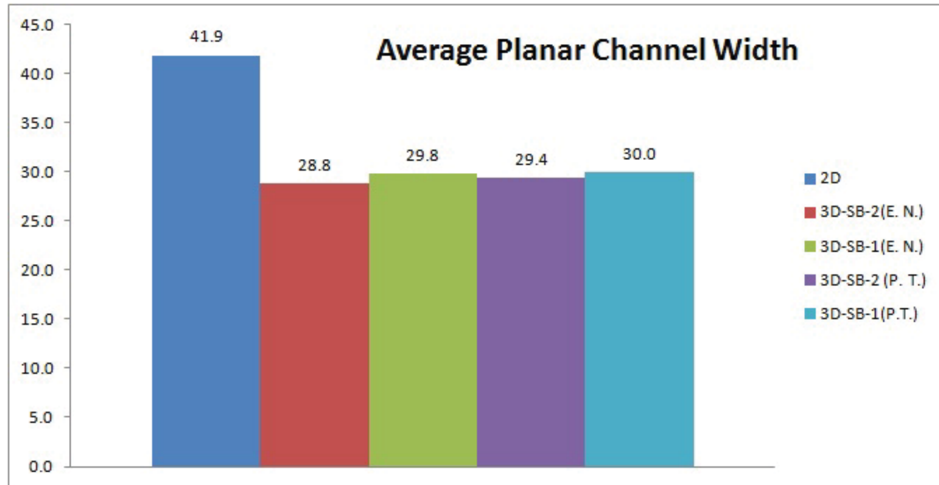


Figure 5.12: Average Planar Channel Width

On other hand, we can find that the vertical channel width which indicates the number of TSV placed per FPGA tile shows some obvious improvement over the previous work. The previous work indicates that in average, 4.5 TSVs per tile are required. We also use the previous 3D SA placement algorithm to place the 3D uni-directional architecture with 3D-SB-2 and it shows that it requires more TSV per tile. By adopting our "even number" placement, we can find that the number of TSV required per tile can be reduced below the value of previous work and even lower while using UB switch and "place twice" placement. However, the "place twice" placement in trade off requires more time for the circuit placement.

Regarding to the routing area, we can find that 3D architecture tends to

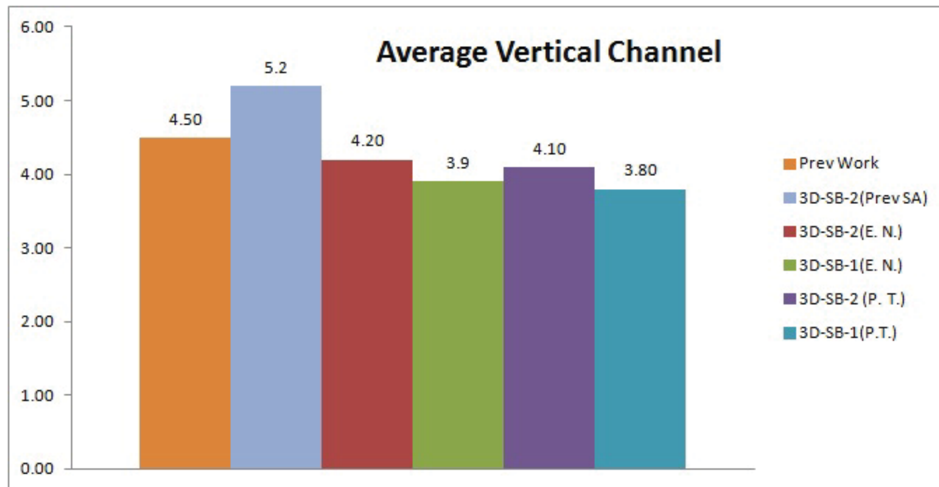


Figure 5.13: Average Vertical Channel Width

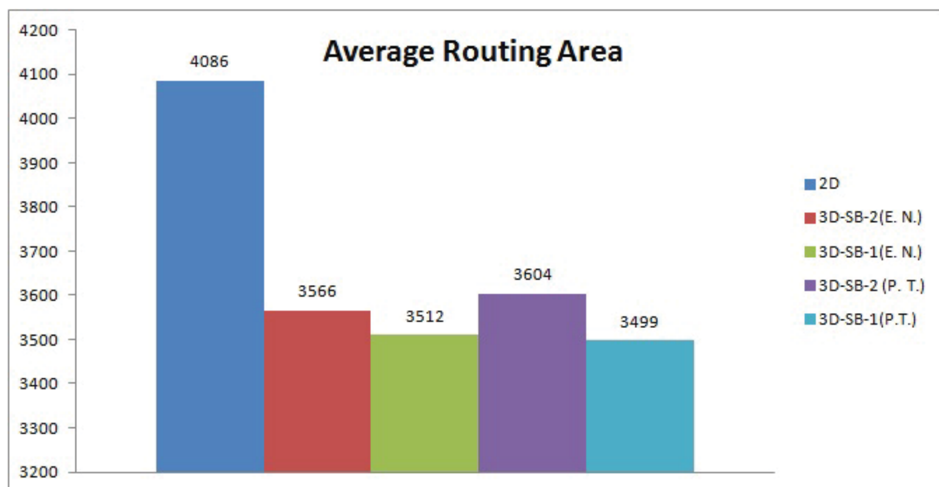


Figure 5.14: Average Routing Area

use less area compared with 2D baseline. It is because of the significant reduction of planar channel width. In addition, we must admit that the TSV area used here is an ideal case. In the actual physical layout the area of 3D architecture per tile may even exceed the 2D baseline architecture's

tile due to the TSV's present. But we cannot estimate the differences here since it depends on various factors such as fabrication process and diameter of the TSV. Due to the constraint of TSV fabrication technology, the fewer number of TSVs in the circuit, the better for the layout and area saving.

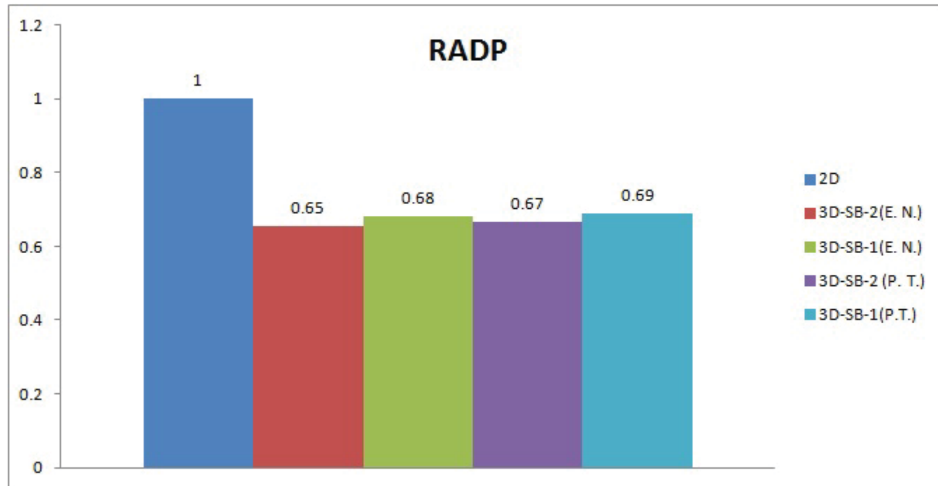


Figure 5.15: RADP

Figure 5.15 shows the routing area delay product. We can observe that significant less RADP value in 3D architecture can be achieved, compared with the 2D baseline architecture. Since we expect more area penalty of TSV placement while using, the RADP value should be higher than the value indicated in Figure 5.15. It is because the TSV area calculation is based on the ideal case.

5.4 Thermal Placement Exploration

The thermal aware placement algorithm is the combination of HotSpot thermal model with VPR so that we can alleviate the temperature issue raised by the high power density. In the 3D stacking architecture the temperature raises faster than the 2D architecture since the 3D architecture dramatically increases the logic density as well as the power density. Therefore, in the 3D architecture, the thermal issue needs to be resolved in order to maintain the circuit reliability.

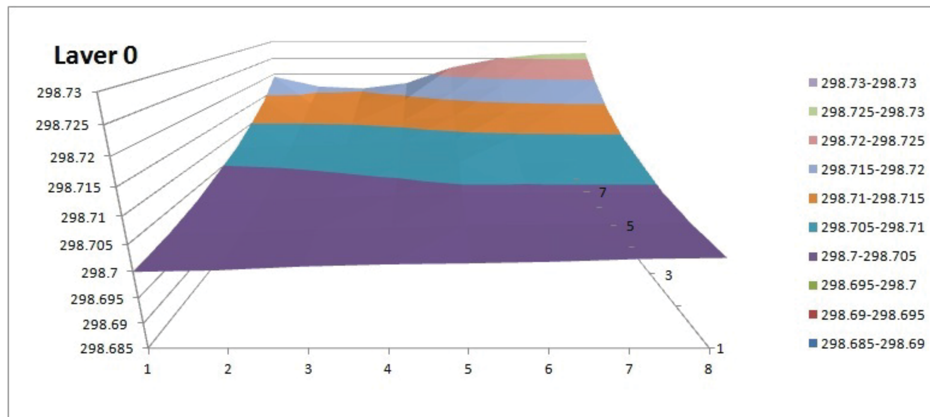


Figure 5.16: Thermal Profile of Layer 0

Equation 4.4 is the cost function of the thermal aware placement algorithm. We use $\alpha = 0.2$, $\beta = 0.2$, $\gamma = 0.4$, and $\theta = 2$ as the weight of each cost. $\theta = 2$ is used here because the power generated by the benchmark circuit

cannot generate enough difference and we need increase the cost through its weight so that the effect can be more obvious. One of drawback of this thermal aware placement is that the time needed to finish is too long. For example, we use the Linux virtual machine with I5 dual core laptop, the running time for the placement is about 4 to 5 hours for the benchmark circuit, bigkey. It is not practical for industry design since the time cost is too much. In this section, we only demonstrate how it works and what is the result looks like.

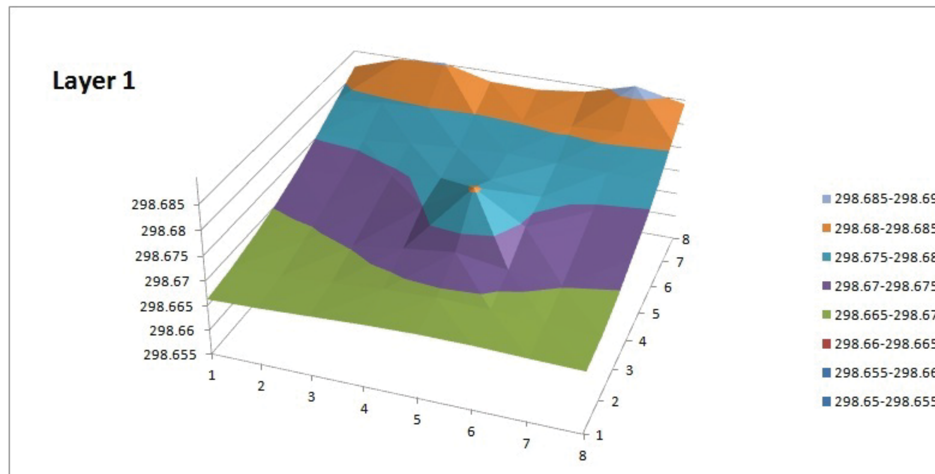


Figure 5.17: Thermal Profile of Layer 1

Figure 5.16 to 5.20 show the profile of each layer in a 5-layer stacked 3D FPGA by using thermal aware placement algorithm while placing bigkey benchmark circuit. We record the estimated thermal profile after the complete of the placement stage. Since we do not involve the thermal aware

routing, we think plot the estimated profile here should be fair to evaluate the thermal placement algorithm. Through Figure 5.16 to 5.20, we can find that temperature drops from Layer 0 down to Layer 4.

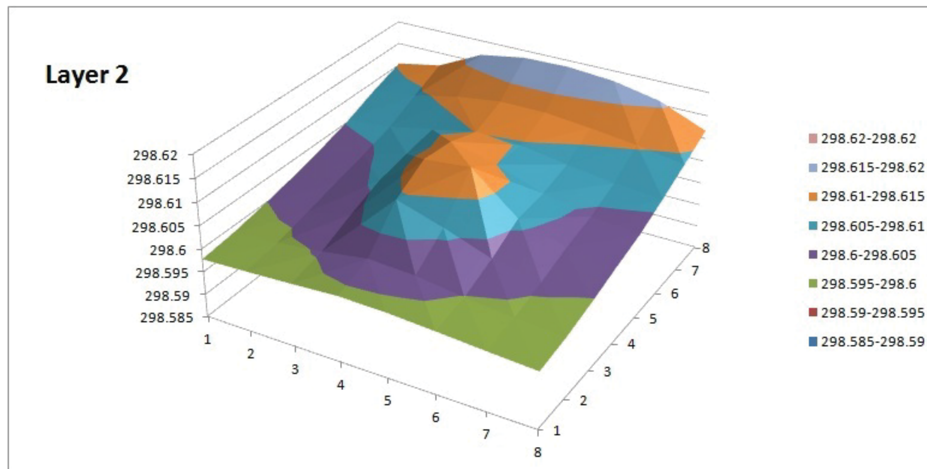


Figure 5.18: Thermal Profile of Layer 2

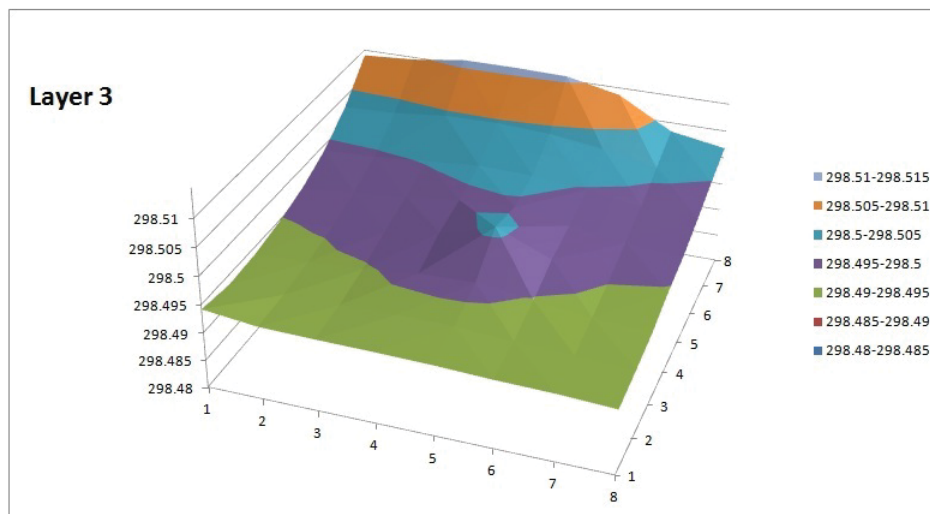


Figure 5.19: Thermal Profile of Layer 3

In overall, based on our simulation, the highest temperature in the 3D

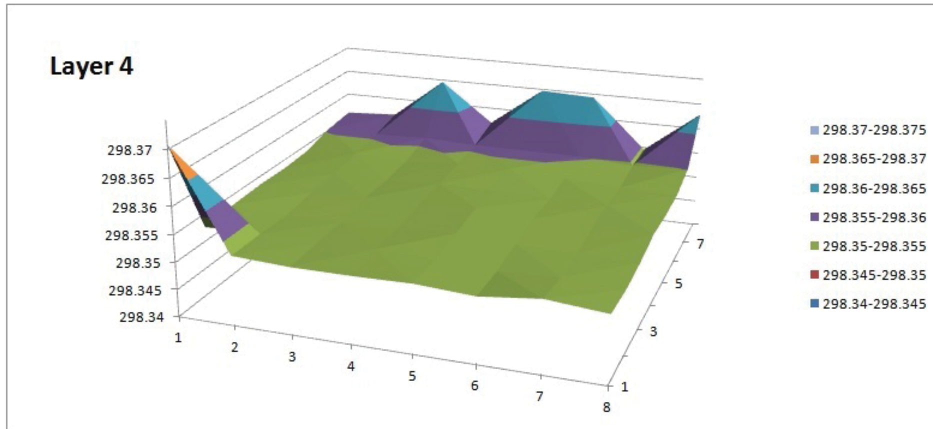


Figure 5.20: Thermal Profile of Layer 4

architecture is 298.727K at Layer 0 and the lowest temperature is 298.352K at Layer 4. It has about 0.375K difference. On the other hand, we also try to compare with estimated thermal profile without the temperature cost. It shows that the hottest temperature is 298.731 at Layer 0 and the lowest temperature is 298.355K. The difference is 0.376K. At the same time, we record other factors such as delay and channel width as shown in Table 5.1.

From Table 5.1 we can find that the trade off is the increment of the delay. On the other hand, bigkey is not a good benchmark for demonstration. Larger size of the benchmark may be better. But it takes too long time for the thermal aware placement. We need to research on a new solution in the future to achieve more practical run time.

Placement Method	Delay (ns)	Max. Temp.(K)	Planar Channel Width	Vertical Channel Width
With Thermal Aware	27.4	298.727	20	2
Without Thermal Aware	23.0	298.731	20	2

Table 5.1: Placement Method Compare

6 | Conclusion and Future Work

In this thesis, we present our work on architecture and EDA of a uni-directional routing based 3D FPGA aiming to reduce the number of TSV placed between two layers and to resolve the thermal issue caused by the high power density of 3D stacking architecture.

Our result shows that, by applying the 3D architecture, the delay and planar channel can be significantly reduced by 25% to 30%, compared with the 2D baseline architecture. Regarding to the TSV usage, our improved placement algorithm together with the new UB switch is able to use about 16% less TSV for the signal propagation in the vertical direction. The RADP value shows that more than 30% of reduction can be achieved. However, due to the ideal assumption on the area cost of TSV, the value needs to be adjusted by the actual physical layout parameters.

On the other hand, we combined two open source software VPR and Hot-Sport to enable the thermal aware placement. Based on the simulation result, we can find that the thermal issue can be alleviated. However, this placement algorithm is not practical due to a very long processing time for the placement. For the purpose of research, it is good for us to know what happens while applying the thermal placement algorithm.

For the future work, we will focus on speeding up the placement time while keeping reasonable circuit speed. Furthermore, we will continue to investigate the UB switch and configurable direction in the vertical channel for both hardware and EDA software to further improve the TSV usage as well as the delay.

Publication

[1] J. Hou, H. Yu, Y. Ha, and X. Liu, "The architecture and placement algorithm for a uni-directional routing based 3d fpga," in *Field-Programmable Technology (FPT), 2013 International Conference on*, pp. 28–33, Dec 2013.

References

[1] J. R. Vaughn Betz and A. Marquardt, *Architecture and CAD for Deep-submicron FPGAs.* Norwell: Kluwer Academic Publisher, 1999.

[2] J. Rose, A. El Gamal, and A. Sangiovanni-Vincentelli, "Architecture of field-programmable gate arrays," *Proceedings of the IEEE*, vol. 81, no. 7, pp. 1013–1029, 1993.

[3] G. Lemieux, E. Lee, M. Tom, and A. Yu, "Directional and single-driver wires in fpga interconnect," in *Field-Programmable Technology, 2004. Proceedings. 2004 IEEE International Conference on*, pp. 41–48, 2004.

[4] D. H. Kim, S. Mukhopadhyay, and S.-K. Lim, "Fast and accurate analytical modeling of through-silicon-via capacitive coupling," *Components, Packaging and Manufacturing Technology, IEEE Transactions on*, vol. 1, no. 2, pp. 168–180, 2011.

[5] V. Betz and J. Rose, "Vpr: A new packing, placement and routing tool for fpga research," in *International Workshop on Field-Programmable Logic and Applications*, 1997.

[6] V. Betz, T. Campbell, W. M. Fang, P. Jamieson, I. Kuon, J. Luu, A. Marquardt, J. Rose, and A. Ye, "Vpr and t-vpack user's menu." `http://www.eecg.utoronto.ca/vpr/VPR_5.pdf`, July 2009.

[7] K. Skadron, M. R. Stan, K. Sankaranarayanan, W. Huang, S. Velusamy, and D. Tarjan, "Temperature-aware microarchitecture: Modeling and implementation," *ACM Trans. Archit. Code Optim.*, vol. 1, pp. 94–125, Mar. 2004.

[8] G. Katti, M. Stucchi, K. De Meyer, and W. Dehaene, "Electrical modeling and characterization of through silicon via for three-dimensional ics," *Electron Devices, IEEE Transactions on*, vol. 57, no. 1, pp. 256–262, 2010.

[9] E. Kusse and J. Rabaey, "Low-energy embedded fpga structures," in *Low Power Electronics and Design, 1998. Proceedings. 1998 International Symposium on*, pp. 155–160, 1998.

[10] V. George, *Low Energy Field-Programmable Gate Array.* PhD thesis, University of California, Berkeley, 2000.

[11] A. Dehon, *Reconfigurable Architectures for General-Purpose Computing*. PhD thesis, Massachusetts Institute of Technology, 1996.

[12] T. Thorolfsson, G. Luo, J. Cong, and P. Franzon, "Logic-on-logic 3d integration and placement," in *3D Systems Integration Conference (3DIC), 2010 IEEE International*, pp. 1–4, 2010.

[13] K. Nomura, K. Abe, S. Fujita, Y. Kurosawa, and A. Kageshima, "Performance analysis of 3d-ic for multi-core processors in sub-65nm cmos technologies," in *Circuits and Systems (ISCAS), Proceedings of 2010 IEEE International Symposium on*, pp. 2876–2879, 2010.

[14] T. Thorolfsson, S. Melamed, G. Charles, and P. Franzon, "Comparative analysis of two 3d integration implementations of a sar processor," in *3D System Integration, 2009. 3DIC 2009. IEEE International Conference on*, pp. 1–4, 2009.

[15] M. Lin, A. El Gamal, Y.-C. Lu, and S. Wong, "Performance benefits of monolithically stacked 3d-fpga," in *Proceedings of the 2006 ACM/SIGDA 14th international symposium on Field programmable gate arrays*, FPGA '06, (New York, NY, USA), pp. 113–122, ACM, 2006.

[16] M. Lin and A. El Gamal, "A routing fabric for monolithically stacked 3d-fpga," in *Proceedings of the 2007 ACM/SIGDA 15th international symposium on Field programmable gate arrays*, FPGA '07, (New York, NY, USA), pp. 3–12, ACM, 2007.

[17] C. Dong, C. Chen, S. Mitra, and D. Chen, "Architecture and performance evaluation of 3d cmos-nem fpga," in *System Level Interconnect Prediction (SLIP), 2011 13th International Workshop on*, pp. 1–8, 2011.

[18] C. Ababei, H. Mogal, and K. Bazargan, "Three-dimensional place and route for fpgas," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 25, no. 6, pp. 1132–1140, 2006.

[19] A. Gayasen, N. Vijaykrishnan, M. Kandemir, and A. Rahman, "Switch box architectures for three-dimensional fpgas," in *Proceedings of the 14th Annual IEEE Symposium on Field-Programmable Custom Computing Machines*, FCCM '06, (Washington, DC, USA), pp. 335–336, IEEE Computer Society, 2006.

[20] J. Soofiani and N. Masoumi, "Area efficient switch box topologies for 3d fpgas," in *New Circuits and Systems Conference (NEWCAS), 2011 IEEE 9th International*, pp. 390–393, 2011.

[21] J. Luu, I. Kuon, P. Jamieson, T. Campbell, A. Ye, W. M. Fang, K. Kent, and J. Rose, "Vpr 5.0: Fpga cad and architecture exploration tools with single-driver routing, heterogeneity and process scaling," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 4, pp. 32:1–32:23, Dec. 2011.

[22] P. Jamieson, W. Luk, S. J. E. Wilton, and G. Constantinides, "An energy and power consumption analysis of fpga routing architectures," in *Field-Programmable Technology, 2009. FPT 2009. International Conference on*, pp. 324–327, 2009.

[23] W. Huang, M. Stan, K. Skadron, K. Sankaranarayanan, S. Ghosh, and S. Velusamy, "Compact thermal modeling for temperature-aware design," in *Design Automation Conference, 2004. Proceedings. 41st*, pp. 878–883, 2004.

[24] J. Rose and S. Brown, "Flexibility of interconnection structures for field-programmable gate arrays," *Solid-State Circuits, IEEE Journal of*, vol. 26, no. 3, pp. 277–282, 1991.

[25] S. Brown, M. Khellah, and G. Lemieux, "Segmented routing for speed-performance and routability in field-programmable gate arrays," *Journal of VLSI Design*, vol. 4, pp. 275–291, 1996.

[26] P. Chow, S. O. Seo, J. Rose, K. Chung, G. Paez-Monzon, and I. Rahardja, "The design of an sram-based field-programmable gate array. i. architecture," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 7, no. 2, pp. 191–197, 1999.

[27] Xilinx Inc., "Xc4000xla/xv field programmable gate arrays." `http://www.xilinx.com/support/documentation/data_sheets/ds015.pdf`, March 2013.

[28] Altera Corporation, "Stratix iv device handbook (volume 1)." `http://www.altera.com/literature/hb/stratix-iv/stratix4_handbook.pdf`, September 2012.

[29] Xilinx Inc., "Vertex-5 family overview." `http://www.xilinx.com/support/documentation/data_sheets/ds100.pdf`, February 2009.

[30] B. Black, D. Nelson, C. Webb, and N. Samra, "3d processing technology and its impact on ia32 microprocessors," in *Computer Design: VLSI in Computers and Processors, 2004. ICCD 2004. Proceedings. IEEE International Conference on*, pp. 316–318, 2004.

[31] D. H. Kim, S. Mukhopadhyay, and S.-K. Lim, "Tsv-aware interconnect length and power prediction for 3d stacked ics," in *Interconnect Tech-*

nology Conference, 2009. IITC 2009. IEEE International, pp. 26–28, 2009.

[32] P. Jamieson and J. Rose, "A verilog rtl synthesis tool for heterogeneous fpgas," in *Field Programmable Logic and Applications, 2005. International Conference on*, pp. 305–310, 2005.

[33] J. Pistorius and M. Hutton, "Benchmarking method and designs targeting logic synthesis for fpgas," in *Proc. IWLS '07*, 2007.

[34] K. K. W. Poon, A. Yan, and S. J. E. Wilton, "A flexible power model for fpgas," in *Proceedings of the Reconfigurable Computing Is Going Mainstream, 12th International Conference on Field-Programmable Logic and Applications*, FPL '02, (London, UK, UK), pp. 312–321, Springer-Verlag, 2002.

[35] K. K. W. Poon, "Power estimation for field programmable gate arrays," Master's thesis, University of British Columbia, 2002.

[36] P. A. Jamieson, "vpr_5_pow_dist.tar.gz." `http://www.users.muohio.edu/jamiespa/vpr_5_pow.html`.

[37] K. Sankaranarayanan, S. Velusamy, M. Stan, C. L, and K. Skadron, "A case for thermal-aware floorplanning at the microarchitectural level," *Journal of ILP*, vol. 7, 2005.

[38] M. J. Alexander, J. P. Cohoon, J. L. Colflesh, J. Karro, E. L. Peters, and G. Robins, "Placement and routing for three-dimensional fpgas," 1996.

[39] M. Leeser, W. Meleis, M. Vai, S. Chiricescu, W. Xu, and P. M. Zavracky, "Rothko: a three-dimensional fpga," *Design Test of Computers, IEEE*, vol. 15, no. 1, pp. 16–23, 1998.

[40] W. Meleis, M. Leeser, P. Zavracky, and M. Vai, "Architectural design of a three dimensional fpga," in *Advanced Research in VLSI, 1997. Proceedings., Seventeenth Conference on*, pp. 256–268, 1997.

[41] G. Borriello, C. Ebeling, S. Hauck, and S. Burns, "The triptych fpga architecture," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 3, no. 4, pp. 491–501, 1995.

[42] V. Pangracious, H. Mehrez, and Z. Marakchi, "Tsv count minimization and thermal analysis for 3d tree-based fpga," in *IC Design Technology (ICICDT), 2013 International Conference on*, pp. 223–226, 2013.

[43] V. Pangracious, H. Mehrez, and Z. Marakchi, "Designing 3d tree-based fpga: Interconnect optimization and thermal analysis," in *New Circuits and Systems Conference (NEWCAS), 2013 IEEE 11th International*, pp. 1–4, 2013.

[44] A. Gayasen, V. Narayanan, M. Kandemir, and A. Rahman, “Designing a 3-d fpga: switch box architecture and thermal issues,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 16, pp. 882–893, July 2008.

[45] K. Siozios, V. F. Pavlidis, and D. Soudris, “A novel framework for exploring 3-d fpgas with heterogeneous interconnect fabric,” *ACM Trans. Reconfigurable Technol. Syst.*, vol. 5, pp. 4:1–4:23, Mar. 2012.

[46] G. Karypis, R. Aggarwal, V. Kumar, and S. Shekhar, “Multilevel hypergraph partitioning: application in vlsi domain,” in *Proceedings of the 34th annual Design Automation Conference*, DAC ’97, (New York, NY, USA), pp. 526–529, ACM, 1997.

[47] K. Siozios, K. Sotiriadis, V. F. Pavlidis, and D. Soudris, “A software-supported methodology for designing high-performance 3d fpga architectures,” in *Very Large Scale Integration, 2007. VLSI - SoC 2007. IFIP International Conference on*, pp. 54–59, 2007.

[48] K. Siozios, K. Tatas, G. Koutroumpezis, D. Soudris, and A. Thanailakis, “An integrated framework for architecture level exploration of reconfigurable platform,” in *Field Programmable Logic and Applications, 2005. International Conference on*, pp. 658–661, 2005.

[49] C.-H. Tsai and S.-M. Kang, "Cell-level placement for improving substrate thermal distribution," *Computer-Aided Design of Integrated Circuits and Systems, IEEE Transactions on*, vol. 19, no. 2, pp. 253–266, 2000.

[50] S. Velusamy, W. Huang, J. Lach, M. Stan, and K. Skadron, "Monitoring temperature in fpga based socs," in *Computer Design: VLSI in Computers and Processors, 2005. ICCD 2005. Proceedings. 2005 IEEE International Conference on*, pp. 634–637, 2005.

[51] S. Bhoj and D. Bhatia, "Thermal modeling and temperature driven placement for fpgas," in *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on*, pp. 1053–1056, 2007.

[52] W. Huang, S. Ghosh, S. Velusamy, K. Sankaranarayanan, K. Skadron, and M. Stan, "Hotspot: a compact thermal modeling methodology for early-stage vlsi design," *Very Large Scale Integration (VLSI) Systems, IEEE Transactions on*, vol. 14, no. 5, pp. 501–513, 2006.