Università degli Studi di Bologna

FACOLTA' DI INGEGNERIA

DOTTORATO DI RICERCA IN INGEGNERIA ELETTRONICA,
INFORMATICA E DELLE TELECOMUNICAZIONI
Ciclo XXII

ARCHITETTURE DI INTERCONNESSIONE PER SISTEMI SU SINGOLO CHIP E PER SISTEMI AD INTEGRAZIONE TRIDIMENSIONALE

Tesi di Dottorato di:
IGOR LOI

Relatori:
Chiar.mo Prof. **LUCA BENINI**

Coordinatore:
Chiar.mo Prof. **PAOLA MELLO**

Settore Scientifico Disciplinare: ING/INF01 Elettronica
Anno Accademico 2008/09

Architetture di interconnessione per sistemi su singolo chip e per sistemi ad integrazione tridimensionale

Igor Loi

DEIS

Universita' di Bologna

A thesis submitted for the degree of

PhilosophiæDoctor (PhD) in Electronic Engineering

2010 March

Abstract

I continui sviluppi nel campo della fabbricazione dei circuiti integrati hanno comportato frequenti stravolgimenti nel design, nell'implementazione e nella scalabilitá dei device elettronici, cosí come nel modo di utilizzarli. Anche se la legge di Moore ha anticipato e caratterizzato questo trend nelle ultime decadi, essa stessa si trova a fronteggiare attualmente enormi limitazioni, superabili solo attraverso un diverso approccio nella produzione di chip, consistente in pratica nella sovrapposizione verticale di diversi strati collegati elettricamente attraverso speciali vias. Sul singolo strato, le network on chip sono state suggerite per ovviare le profonde limitazioni dovute allo scaling di strutture di comunicazione condivise. Questa tesi si colloca principalmente nel contesto delle nascenti piattaforme multicore ad alte prestazioni basate sulle 3D NoC, in cui la network on chip viene estesa nelle 3 direzioni. L'obiettivo di questo lavoro é quello di fornire una serie di strumenti e tecniche per poter costruire e caratterizzare una piattaforma tridimensionale, cosi come dimostrato nella realizzazione del testchip 3D NOC fabbricato presso la fonderia IMEC. Il primo contributo é costituito sia una accurata caratterizzazione delle interconnessioni verticali (TSVs) (ovvero delle speciali vias che attraversano l'intero substrato del die), sia dalla caratterizzazione dei router 3D (in cui una o piú porte sono estese nella direzione verticale) ed infine dal setup di un design flow 3D utilizzando interamente CAD 2D. Questo primo step ci ha permesso di effettuare delle analisi dettagliate sia sul costo sia sulle varie implicazioni. Il secondo contributo é costituito dallo sviluppo di alcuni blocchi funzionali necessari per garantire il corretto funziomento della 3D NoC, in presenza sia di guasti nelle TSVs (fault tolerant links) che di deriva termica nei vari clock tree dei vari die (alberi di clock indipendenti). Questo secondo contributo é costituito dallo sviluppo delle seguenti soluzioni circuitali: 3D fault tolerant link, Look Up

Table riconfigurabili e un sicnronizzatore mesocrono. Il primo é costituito fondamentlmente un bus verticale equipaggiato con delle TSV di riserva da utilizzare per rimpiazzare le vias guaste, piú la logica di controllo per effettuare il test e la riconfigurazione. Il secondo é rappresentato da una Look Up Table riconfigurabile, ad alte prestazioni e dal costo contenuto, necesaria per bilanciare sia il traffico nella NoC che per bypassare link non riparabili. Infine la terza soluzione circuitale é rappresentata da un sincronizzatore mesocrono necessario per garantire la sincronizzazione nel trasferimento dati da un layer and un altro nelle 3D Noc. Il terzo contributo di questa tesi é dato dalla realizzazione di un interfaccia multicore per memorie 3D (stacked 3D DRAM) ad alte prestazioni, e dall'esplorazione architetturale dei benefici e del costo di questo nuovo sistema in cui il la memoria principale non é piu il collo di bottiglia dell'intero sistema. Il quarto ed ultimo contributo é rappresentato dalla realizzazione di un 3D NoC test chip presso la fonderia IMEC, e di un circuito full custom per la caratterizzazione della variability dei parametri RC delle interconnessioni verticali.

Contents

List of Figures

1

Introduction

The recent electronics revolution has been fueled by the decades-long trend of exponential growth in circuit performance through device scaling. In current and future technologies, simple device scaling does not result in the same performance improvements. For deeply scaled technologies, optimizing interconnect performance is of equal importance as device performance. Following the International Technology Roadmap for Semiconductors (ITRS) specifications for scaled interconnect, worst-case and even average-case interconnect performance decreases with each technology generation. Because the performance improvement achievable through Moores law scaling is bottoming out, new technologies and methodologies to achieve performance enhancement are being explored. Networks on chip and Three-dimensional integration are one such technology/methodology.

The Network of Chip paradigm have been proposed to tackle the interconnect bottleneck at system level. Cores are atteched to the network nodes, and the path between these nodes is segmented using one or more repeaters. This methodologies is architecturable scalable with the increasing number of cores of modern multicores, and physically scalable with the CMOS scaling trend.

Three-dimensional integration offers several performance improvements for electronic systems. First, 3D integration offers a greater device density for a given footprint area. Essentially, more functionality can be packed into a given area. Second, long, global wires can be replaced with shorter, local wires by exploiting vertical interconnect used to connect device layers. This results in lower power dissipation and shorter timing delays. Finally, a 3D technology permits the integration of heteroge-

neous technologies. Using a wafer-bonding approach to 3D integration, each device layer is fabricated independently of every other device layer allowing each system to be fabricated in a different technology. Circuits that are part of system-on-a-chip (SOC) applications can be fabricated in the device technology that yields optimal performance and then subsequently integrated together using a 3D integrated technology. Because each subsystem is fabricated in the optimal technology, optimal overall system performance can be achieved.

Actually the numbers of cores integrated in a typical Multi Processor System on Chip (MPSoC) ranges to 2 (INTEL core Duo) to 8 (UltraSparc T1-T2), with few excpetion with more than 8 cores. Since the number of cores is limited, they are simply interconnected through a crossbar or shared bus. The memory system is always hierarchical, composed by private L1 and shared L2 caches, while the L3 level (DRAM) is always off chip, due the logic-memory process incompability. Only few L3 memory channels are available because they require hundreds of IO pins per channel. The system performances strongly depend on the memory access time, which actually, it represents the main bottleneck even for a dual core. Although this approach is acceptable for a limited number of cores, will be infeasible will the next generation of multicore, because to sustain the same system performances, the L2 cache must be increased with a cubic dependence with respect to the number of cores. This requirement leads to adopt huge L2 caches (SRAM), which clearly will be not feasible with the acual trend.

As the number of cores will significantly increases in the next years, a NoC based approach will be mandatory to keep performance requirements within each processing elements (PEs). Merging NoC paradigm with the three-dimensional integration will be a strategic choice to sustain the increasing demand of power computation and power efficiency. Three-dimesional NoCs therefore will gain the benefits of the scalability, modularity and shorter interconnects. Moreover, removing the L2 chaches and stacking DRAM on top of the logic die (heterogenous systems) will remove the main bottleneck from the memory, therefore balancing the interconnect infrastructure and memory latencies. This strategy is the only way to sustain the Moore's law in the next 10 years.

This thesis has four major objectives: first a novel design flow for 3D integration using 2D CAD; second to propose an high performance multicore platform based on 3D NoCs; third to propose some architectural blocks to overcome some new challenges that

were not present in the 2D fabrication; fourth, to propose a novel memory interface for 3D Stacked DRAM and many core platform.

The organization of the thesis will be outlined in this chapter with the a summary of contributions in design methodology, VLSI SoC architectures and joint optimization of design methodology and architectures.

Chapter 2 describes an automated design flow for 3D NoCs, starting from the RTL to the GDSII. A Through-Silicon Vias (TSVs) interconnects modeling are presented. Extracted models are used to evaluate the design implications of extending switch architectures with ports in the vertical direction. In addition, it is presented a design flow allowing for post-layout simulation of NoCs with links all three physical dimensions.

The synchronization in 3D integrated circuits is an undiscovered field. GALS approach based on dual clock fifos are often used to perform this task, but the power and area cost is heavy. Chapter 3 presents a scheme to handle mesochronous communication in 3D NoCs with minimal area and power impact, and analyze the circuit design, the timing properties, the requirements to support flow control across mesochronous links, and the implementation cost of such a scheme after placement and routing.

From a product development viewpoint 3D technologies are still in their infancy, and are only expected on market in 2012-2015 timeframe. Many questions remain as how to make this technology low cost, reliable and yielding. Chapter 4 presents a defect-tolerance technique for TSVs-based multi-bit links through an efficient and effective use of redundancy. Using few spares vertical interconnects, the overall yield is increased up to 98% ,with minimal hardware overhead and with the minimal timing impact.

Chapter 5 is stricly connected with chapter 4. Link failures and dynamically changing application scenarios represent demanding constraints for the provision of suitable Quality of Service. NoCs with configurable routing, whereby the communication routes are explicitly chosen at runtime out of a set of statically predefined alternatives, provide intelligent adaptation without impacting the consistency of traffic flows. In this chapter is presented an exploration and synthesis approach that, depending on the required amount of routing flexibility, can for example reduce the area cost of the NoC routing tables by adopting partially reprogrammable routing logic instead of fully reprogrammable tables.

Chapter 6 describes our effort in establishing a 3DNoC design flow and in designing circuits and architectural solutions for variability and reliability characterization and tolerance.

Chapter 7 presents the 3D NoCs testchip developed using the IMEC 3D design kit. The 3D NoC is equipped with fault tolerant links and JTAG interface to test each blocks of the chip, to get TSV yield information, and to characterize the power impact 2D vs 3D traffic.

To tackle the memory wall in complex SoC or MPSoCs, 3D Stacked DRAM is the most promising solution. To fully exploit the potential benefits of stacked memories, the architectural interface to vertically stacked memory must be streamlined. In chapter 8 we present an efficient and flexible distributed memory interface for 3D-stacked DRAM. An efficient memory interface is presented, ensuring ultra-low-latency access to the memory modules on top of each processing element. Communication to these local modules do not travel through the NoC and takes full advantage of the lower latency of vertical interconnect, thus speeding up significantly the common case.

Finally, chapter 8 presents the conclusion and overall contribution in this research field.

2

Vertical interconnect modeling and 3D NoCs

Three-dimensional (3D) manufacturing technologies are viewed as promising solutions to the bandwidth bottlenecks in VLSI communication. At the architectural level, Networks-on-chip (NoCs) have been proposed to address the complexity of interconnecting an ever-growing number of cores, memories and peripherlas. NoCs are a promising choice for implementing scalable 3D interconnect architectures. However, the development of 3D NoCs is still at an early development stage. In this chapter, we present a semi-automated design flow for 3D NoCs. Starting from an accurate physical and geometrical model of Through-Silicon Vias (TSVs), we extract a circuit-level model for vertical interconnections, and we use it to evaluate the design implications of extending switch architectures with ports in the vertical direction. In addition, we present a design flow allowing for post-layout simulation of NoCs with links all three physical dimensions.

2.1 Introduction

Over the years, advances in silicon technology have enabled the integration of larger and larger amounts of processing elements and memories, with increasing communication requirements at their interfaces. Simultaneously, there has been a strong push towards the mixing of functional blocks which may require some processing steps differentiating them from plain CMOS, such as DRAM, MEMS, passive and active analog circuitry,

optoelectronic elements, chemical sensors, actuators, *etc.*. Vertically stacking multiple layers of silicon is an attractive way of sustaining the pace of the improvement in functionality, while providing sufficient communication bandwidth and pursuing design objectives such as small package sizes, minimum footprints and modularity.

In planar implementations, interconnects are becoming a limiting factor to achieve design closure. This is due to several issues, such as the growing ratio of wire delay *vs.* logic delay, signal integrity concerns and stringent bandwidth requirements. At the system level, the key challenge is configuring, optimizing and verifying the communication architecture across many degrees of freedom in terms of topology, architecture and interface protocols. The Network-on-Chip (NoC) paradigm, which brings packet-switching networking concepts to the on-die level, has been proposed (1, 2) to systematically tackle these challenges. NoCs are a structured, predictable and scalable approach to the problem, centered around wire segmentation and point-to-point signaling.

The simultaneous emergence of 3D integration technologies and NoCs exposes new opportunities and new challenges to system designers. The structured nature of NoCs seem to be an ideal way of encapsulating the design properties and requirements (such as heterogeneous wiring resources, large degree of parallelism, architectural heterogeneity) of three dimensional integration. However, design tradeoffs and design technology support for 3D NoCs are yet to be explored in depth.

The first contribution of this chapter is the construction of a circuit-level model for vertical inteconnects (through-silicon vias), based on accurate 3-dimensional parasitic extraction. Comparative analysis demonstrates that not only vertical interconnects are usable, but that they are highly competitive with horizontal wires in terms of delay and power, with a reasonable area overhead. As a second main contribution, we extend a 2-dimensional NoC switch architecture to deal with vertical links. Our third contribution is the development of a prototype design flow for automatic instantiation of 3-dimensional NoCs. Finally, we present a case study where a planar NoC topology is folded and implemented across two chip layers.

2.2 Previous Work

2.2.1 Vertical Stacking

A number of technologies for 3D chip manufacturing have been explored in recent years, including transistor stacking (3), die-on-wafer stacking (4), wafer stacking (5), chip stacking (6). In this chapter we focus on wafer stacking approaches, as one of the most promising avenues for the implementation of high-performance yet inexpensive (multiple 3D chips can be processed in a single pass) three-dimensional ICs. Wafer stacking relies on Through-Silicon Vias (TSVs) (7) for vertical connectivity, guaranteeing low parasitics (*i.e.* low latency and power) and, if needed, extremely high densities of vertical wires (*i.e.* high bandwidth). Tezzaron Semiconductor Corporation (8) and IBM Technologies (9) are active players in this field; the major differences between their processes are in wafer bonding methodologies and TSV formation. The former resorts to via formation followed by high-temperature wafer bonding, so that electrical connectivity and bonding strength are guaranteed by thermocompression. The latter uses oxide fusion bonding at room temperature, allowing a very high precision alignment, while vias are formed after the wafers have been bonded together. In this chapter, we will use fabrication technology parameters disclosed in previous literature by these manufacturers (8, 9).

2.2.2 Networks-on-Chip

NoCs have been suggested as a scalable communication fabric (1, 2). From the architectural point of view, a complete scheme is presented for example in (10), while specific topics are tackled in several works: flow control protocols (11), router power estimations (12), Quality of Service (QoS) provisions (13, 14), asynchronous implementations (15, 16, 17). CAD tools for NoC instantiation and optimization can be found for example in (18, 19).

The synthesis flow of NoCs has been explored by several groups. Layouts are presented in (20, 21), a test chip is shown in (22), and an FPGA target is provided for (23, 24). Synthesis and layout results for the ×pipes library of component blocks that we will leverage upon are detailed in (25, 26). While most efforts have been aiming at standard cell ASIC targets, some groups have been doing custom design (22).

Some research is being undertaken on 3D NoCs. For example, in (27, 28) alternate ways of interconnecting 3D chips are contrasted; namely, the authors focus on several variants of 3D meshes, stacked meshes, stacked tori, *etc.*. The main focus of the authors is on topologies and on performance metrics, while the physical implementation is not studied in depth. Our work is orthogonal and complementary, as we provide accurate characterization of physical effects and parasitics, including coupling capacitances, and discuss a complete flow to implement a 3D NoC at the layout level. In (29), the authors propose a dimension decomposition scheme to optimize the cost of 3D NoC switches, and present some area and frequency figures derived from a physical implementation. The fundamental assumption of their work is that a regular, homogeneous NoC is the best solution for a 3D design, and therefore the next logical step is to reduce the cost of each required building block. However, we believe that, for such complex designs as stacked 3D chips, which are likely to mix logic layers with memory layers and even more uncommon functionality, heterogeneity will likely be significant, especially along the vertical axis. For this reason, we propose a more general approach, where the designer is allowed to choose among planar and vertical communication on a switch-by-switch basis, without any topological constraint. Post-silicon nano-scale 3D interconnections have also been recently investigated (30), but large scale availability of these technologies in the near future is uncertain. To the best of our knowledge, no previous work fully characterizes the vertical interconnections for use in NoCs, especially with respect to physical implementation and timing requirements.

2.3 Physical Modeling of Vertical TSVs

To be useful for a NoC infrastructure, a vertical wire should not be used in isolation; instead, to simplify routing, it is better to create buses of such wires. The geometry of a TSV bus connecting adjacent stacked wafers is shown schematically in Figure 2.1 for two manufacturing scenarios: Silicon on Insulator (SOI) and bulk-silicon technologies. Given the physical proximity of the TSVs, concerns related to capacitive coupling within such buses may arise. In this section, we quantify the delay in a bus formed by vertical TSVs for both the SOI and bulk-silicon cases.

TSV models are obtained with the Ansoft Q3D extractor (31), a quasi-static electromagnetic-field simulation for parasitic extraction of electronic components, which utilizes finite

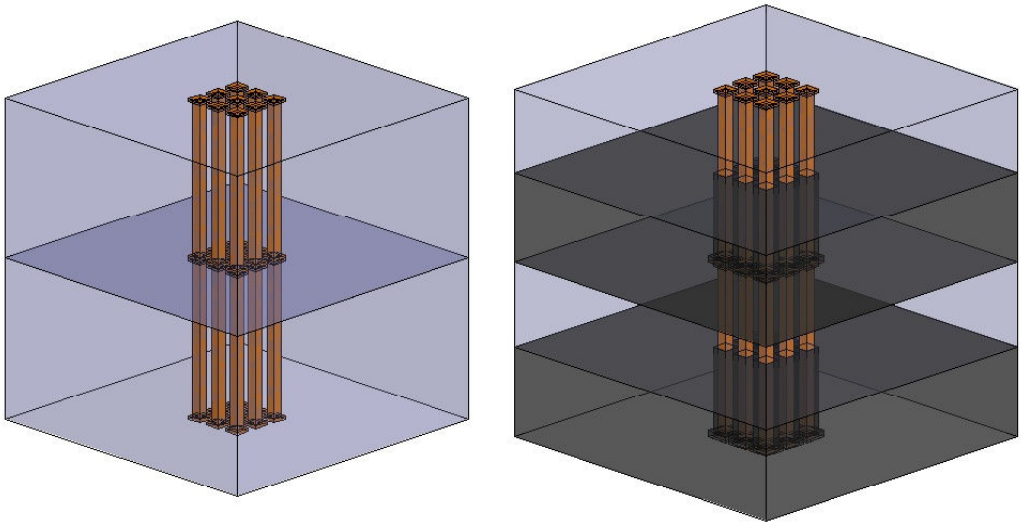


Figure 2.1: Through-Silicon Vias in SOI and bulk-silicon technologies.

element algorithms and the Method of Moments to compute the RLC parameters of a 3D structure. This makes the study of signal integrity (crosstalk, ground bounce) and delay possible.

The starting point of our analysis is a simple configuration composed of nine TSVs placed in a 3x3 grid structure. The baseline configuration we study can be summarized as (Figure 2.2):

Copper vias

$4\mu m \times 4\mu m$ via cross-section ($W \times L$)

$5\mu m \times 5\mu m$ pads at via extremities

$8\mu m$ via pitch

$1\mu m$ oxide thickness (t_{OX}) (only for Si-Bulk)

$50\mu m$ layer thickness ($25\mu m$ bulk-Si and $25\mu m$ SiO_2)

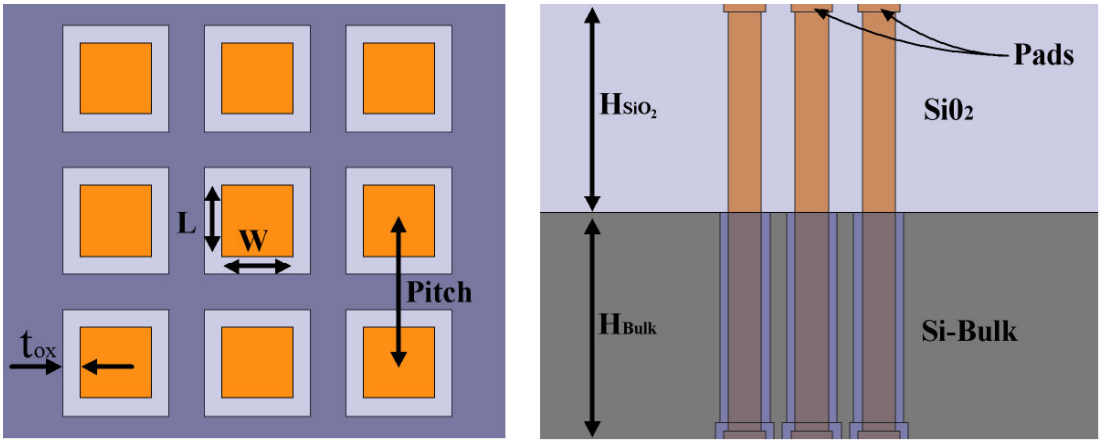


Figure 2.2: Schematic representation of a bundle of 3D vias.

Delay is a function of resistance and capacitance. Resistance can be described with a single parameter as a function of via length and cross-section:

$$R = \frac{\rho \times l}{\sigma} \tag{2.1}$$

For example, copper Vias with $4 \times 4\mu m$ diameter show a resistance per μm around $1.18m\Omega/\mu m$. Skin effect is negligible at few GigaHertz with these dimensions, and a comparison between Vias and top metal wires (Metal8, 130nm tecnology node) which have $0.4 \times 0.8\mu m$ Cross section, shows that Vias Resistance is fifty times smaller than Metal8, with equal length.

Capacitance, on the other hand, due to coupling effects, poses several more issues. Therefore, we resort to a capacitance matrix $\overline{\overline{C}}$ (Equation 2.2):

$$\overline{\overline{C}} = \begin{pmatrix} C_{1,1} & -C_{1,2} & ... & -C_{1,n} \\ -C_{2,1} & C_{2,2} & ... & -C_{1,n} \\ ... & ... & ... & ... \\ -C_{n,1} & -C_{n,2} & ... & C_{n,n} \end{pmatrix} \tag{2.2}$$

In this matrix, the elements outside of the diagonal represent inter-via coupling, with inverted sign, while the ones along the diagonal are the sum of the capacitances towards the ground plane ($C_{i,0}$ - not explicitly reported in the matrix) plus the coupling capacitances:

$$C_{ii} = C_{i,0} + C_{i,1} + ... + C_{i,i-1} + C_{i,i+1} + ... + C_{i,n} \tag{2.3}$$

In Tables 2.1 and 2.2 we report extraction results for the capacitance of vias in SOI and bulk-silicon TSVs, respectively, for the reference case. The capacitance towards

C [fF]	Ground	M	N	S	W	E	SW	NW	NE	SE
M	0.00	11.41	-2.43	-2.43	-2.43	-2.43	-0.41	-0.42	-0.41	-0.42
N	0.00	-2.43	10.13	-0.03	-0.47	-0.47	-0.07	-3.19	-3.18	-0.07
S	0.00	-2.43	-0.03	10.13	-0.47	-0.47	-3.19	-0.07	-0.08	-3.18
W	0.00	-2.43	-0.47	-0.47	10.13	-0.03	-3.18	-3.19	-0.07	-0.07
E	0.00	-2.43	-0.47	-0.47	-0.03	10.13	-0.08	-0.07	-3.19	-3.18
SW	0.00	-0.41	-0.07	-3.19	-3.18	-0.08	8.32	-0.40	-0.11	-0.41
NW	0.00	-0.42	-3.19	-0.07	-3.19	-0.07	-0.40	8.31	-0.40	-0.12
NE	0.00	-0.41	-3.18	-0.08	-0.07	-3.19	-0.11	-0.40	8.32	-0.41
SE	0.00	-0.42	-0.07	-3.18	-0.07	-3.18	-0.41	-0.12	-0.41	8.31

Table 2.1: Capacitance matrix of TSVs in SOI technology. M = middle via; the other vias are labeled according to their positioning with respect to it (N = north, *etc.*). "Ground" refers to the ground plane ($C_{i,0}$).

C [fF]	Ground	M	N	S	W	E	SW	NW	NE	SE
M	-17.7	23.89	-1.20	-1.21	-1.20	-1.20	-0.33	-0.33	-0.36	-0.36
N	-18.1	-1.20	23.26	-0.09	-0.39	-0.34	-0.05	-1.58	-1.52	-0.05
S	-18.3	-1.21	-0.09	23.39	-0.35	-0.33	-1.47	-0.06	-0.05	-1.56
W	-18.1	-1.20	-0.39	-0.35	23.25	-0.09	-1.57	-1.48	-0.05	-0.05
E	-18.3	-1.20	-0.34	-0.33	-0.09	23.42	-0.05	-0.06	-1.55	-1.52
SW	-18.6	-0.33	-0.05	-1.47	-1.57	-0.05	22.23	-0.11	0.00	-0.13
NW	-18.5	-0.33	-1.58	-0.06	-1.48	-0.06	-0.11	22.16	-0.11	-0.01
NE	-18.5	-0.36	-1.52	-0.05	-0.05	-1.55	0.00	-0.11	22.24	-0.13
SE	-18.3	-0.36	-0.05	-1.56	-0.05	-1.52	-0.13	-0.01	-0.13	22.07

Table 2.2: Capacitance matrix of TSVs in bulk-silicon technology. M = middle via; the other vias are labeled according to their positioning with respect to it (N = north, *etc.*). "Ground" refers to the ground plane ($C_{i,0}$).

the ground plane is negligible in the SOI case, since the whole structure is "floating", but it is the dominant element in bulk-silicon technology. On the other hand, due to the presence of a passivation coating around the TSVs in the bulk-silicon case, the SOI scenario exhibits much larger coupling capacitances among the vias.

We can analyze the behavior of TSVs in different geometries using our geometric model. In Figure 2.3 we sweep the TSV diameter, from 0.5 μm to 6 μm, while keeping the TSV pitch constant at 8 μm. Capacitance in the bulk-silicon case increases linearly with the diameter, while the increase is steeper for SOI. This is due to the fact that, in both technologies, the lateral via surface, which determines the coupling, is becoming larger. Further, the distance among the lateral surfaces decreases, since the pitch is constant. However this effect is most relevant in the SOI scenario, whereas, in bulk-silicon, the passivation layer surrounding each TSV (t_{OX} thickness) dampens the increase in coupling.

It is also interesting to sweep via pitch while keeping the TSV diameter constant (*e.g.* at $4\mu m$. The curves are dual with respect to the previous plot, since increasing via

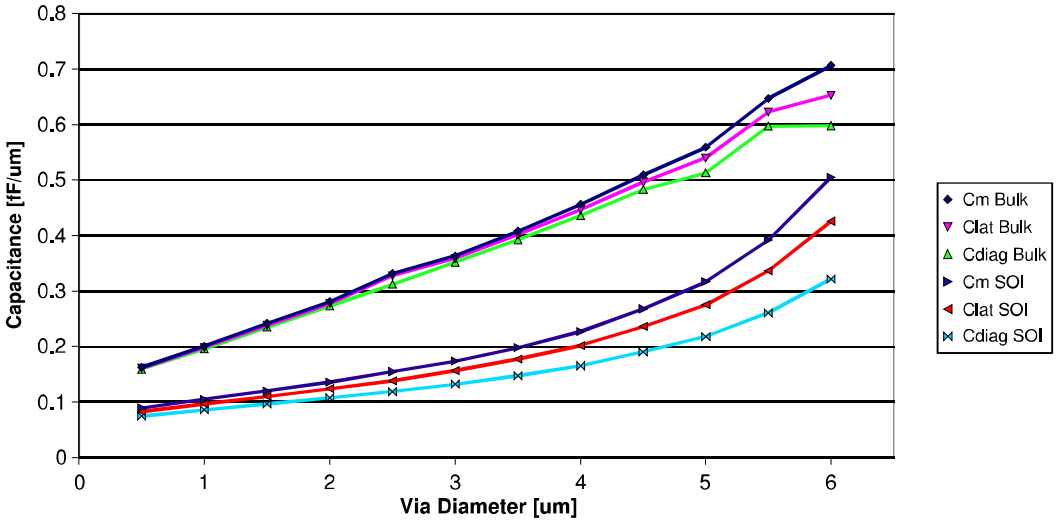


Figure 2.3: Capacitance trend when sweeping the diameter of vias having a constant pitch. Figures are reported for SOI and bulk-silicon. C_m: $C_{1,1}$; C_{lat}: average of $C_{2,2}$ to $C_{5,5}$ (N, S, W, E vias); C_{diag}: average of $C_{6,6}$ to $C_{9,9}$ (SW, NW, NE, SE vias).

diameters has a similar effect as decreasing via pitches. The most interesting property to be observed is the discontinuity in the bulk-silicon curves at the 6 μm pitch threshold, which represents the point where two adjacent TSVs are actually in contact. This is because vias have a 4 μm diameter, plus, only for the bulk-silicon case, an insulating coating 1 μm thick. Below the 6 μm threshold, we assume that TSVs are dug into a solid SiO_2 structure, and are therefore only separated by a thin oxide layer; above the threshold, a Si "screen" appears in the middle as each TSV is the result of a separate etching in the Si substrate. The presence or absence of the Si layer changes substantially the parasitic capacitance behaviour.

Even though the complete extracted circuit model gives maximum accuracy in electrical simulation, good insight can be gained by analyzing delay with the well-known RC time constant approximation (Equation 2.4).

$$t_D = 0.35 \times R \times C \tag{2.4}$$

In the formula, contact resistance and load capacitance (*e.g.* buffers or flip flop at the end of the line) should be taken into account. Since TSVs are interconnected by means of metal bonding, we estimate the contact resistance (32) to be $100m\Omega$ per layer. Delay

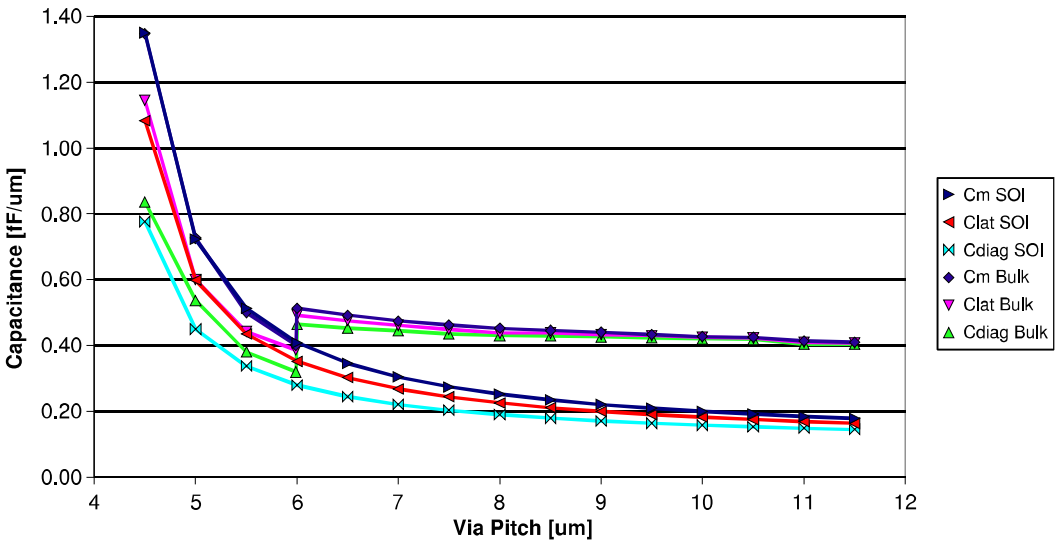


Figure 2.4: Capacitance trend when sweeping the pitch of vias having a constant diameter. Figures are reported for SOI and bulk-silicon. C_m: $C_{1,1}$; C_{lat}: average of $C_{2,2}$ to $C_{5,5}$ (N, S, W, E vias); C_{diag}: average of $C_{6,6}$ to $C_{9,9}$ (SW, NW, NE, SE vias).

estimates using Equation 2.4 are in good agreement with SPICE simulation and are around 16ps for SOI and 18.5ps for Si-Bulk and Vias Diameter set to 4um and Pitch 8um.

To put these results in perspective, the maximum un-repeated planar line length in metal2 and metal3 is 1.5mm. If we take 1.5mm as a resonable planar inter-switch link length, we observe that vertical links exhibit roughly one order of magnitude lower capacitive load. Roughly the same ratio can be found for resistance. As a consequence, even after taking coupling effects of tightly packed TSV bundles into account, vertical links turn out to be substantially faster and more energy efficient than moderate size planar links.

2.4 Integration of TSVs within NoC Switches

NoC components and NoC design tools require modifications to support vertical links made by TSVs. As discussed in Section 2.2, 3D designs are likely to expose a large degree of heterogeneity, especially along the vertical axis. Therefore, we choose to base our integration effort on the ×pipes (26) NoC library, which supports arbitrary

connectivity, and on its instantiation toolchain (33). Thus, we can leverage a semi-automatic design flow, from RTL description to layout-level verification.

×pipes switches come in two radically different variants, conceived to best match two flow control protocols. The first is ACK/NACK, a retransmission-based protocol featuring increased error resilience. The second is STALL/GO, a simple variant of credit-based flow control allowing for pipelined links to be transparently deployed. In the ACK/NACK case, output buffers need to be inserted within switches, since any transmitted packet should be stored for potential retransmission. This implies a hardware cost, but it also means that NoC links are enclosed between two clocked buffers at the sending and receiving ends. Hence, a whole clock period is available for signal propagation along the wires of the inter-switch links. In any case, the link length and the switch logic are decoupled by the output buffer.

In contrast, in STALL/GO, low switch latency reduced buffer cost are the main goals. ×pipes STALL/GO switches therefore adopt a lean architecture, where only switch inputs are buffered. In other words, the switch logic and the link propagation time (up to the following switch or to the first link pipeline stage) contribute to a the same timing path, which becomes the bottleneck for the system. While ACK/NACK transparently allows for links of arbitrary propagation time, possibly just requiring the insertion of pipeline stages, with STALL/GO the link propagation time directly impacts the maximum operating frequency of the switches and so of the whole NoC.

We leverage the information gathered in Section 2.3 to build LEF (Library Exchange Format) descriptions of vertical vias. LEF macros are standard hardware descriptions at the layout level, including information about process technology, cell placement, routing and pins/pads. Based on these macros, TSVs can be accurately inserted within the design during the placement and routing stage; they are simply attached to the input or output pins of a switch port, just as a horizontal bus would. At the RTL level, on the other hand, the design can still be unchanged with respect to a 2D implementation. This brings several advantages: (i) the presence of vertical wires is totally transparent to the architectural and functional views of the architecture; (ii) a chip may feature any degree of connectivity heterogeneity since vertical links can be added or exchanged for horizontal ones; (iii) vertical bandwidth can be added only where needed in the chip, saving switch ports everywhere else; (iv) building upon the savings brought by the previous item, the set of switches with vertical ports, *i.e.* the ones located where

vertical bandwidth is really needed, can have ideal performance because they can be implemented as full crossbars.

Thanks to this approach, a complete flow is achieved; this includes the ability to extract and simulate a 3D layout, where all switch ports are exposed to proper timing constraints and load information is available for both horizontal and vertical connections. A depiction of a sample layout featuring a 3x3 switch with vertical ports is presented in Figure 2.5. The arrangement of the TSV macros is the one we identified to offer the best timing requirements: close to the pinout of the switch, so as to guarantee minimum length of the wire from the switch to the base of the via, thus reducing parasitics.

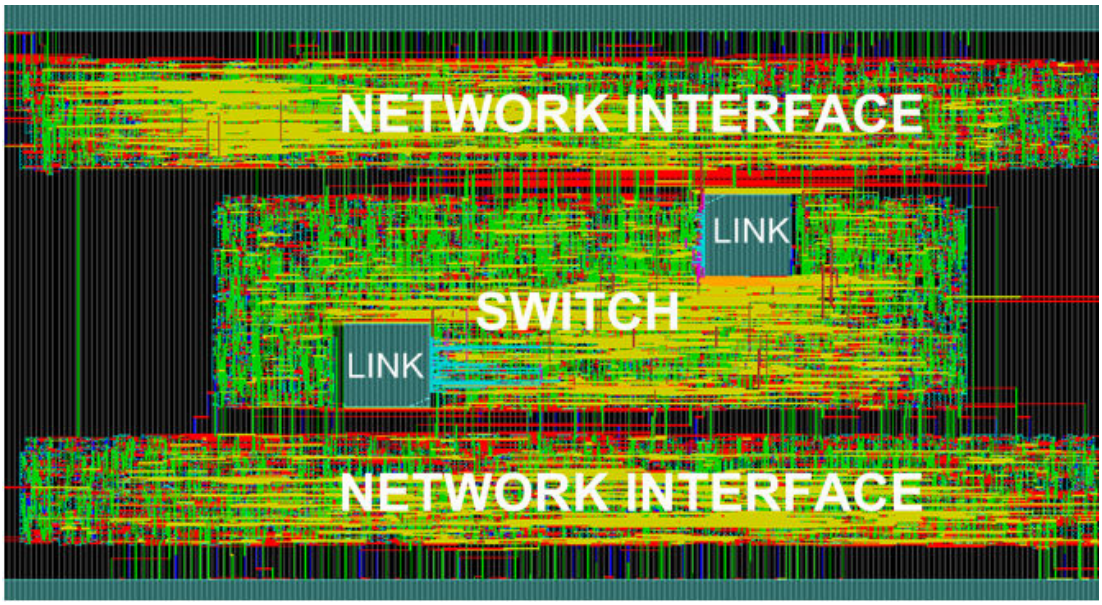


Figure 2.5: Layout detail: a switch is attached to the LEF macros of two vertical links.

The choice of a NoC topology must be performed by taking into account available performance information. Therefore, it is important to build a timing model of the switches. In Figure 2.6, we explore the frequency that STALL/GO and ACK/NACK switches of different cardinalities can achieve when driving horizontal (1.5mm) or vertical (50um) links. As expected, ACK/NACK switches don't change operating frequency when moving to 3D structures, since its frequency bottleneck is given by the switch logic and is not affected by link performance. STALL/GO is, in general, slightly slower than ACK/NACK due to the contribution of link delay on critical paths; however,

when used in combination with TSVs, it regains 30-50 MHz, *i.e.* at 50 to 75% of the frequency gap, while maintaining its low-overhead properties (and single-cycle latency). In other words, the NoC can be clocked faster when the slowest horizontal links are replaced by fast vertical links.

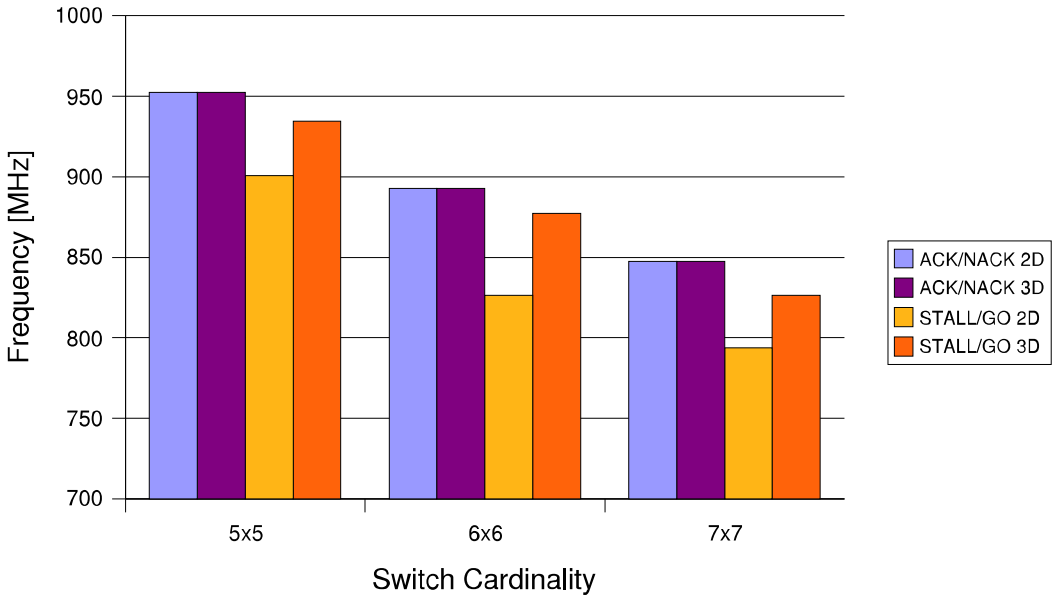


Figure 2.6: Maximum frequency achievable by STALL/GO *vs.* ACK/NACK switches in 2D and 3D flows, for varying switch cardinalities.

2.5 Implementation of TSV-based NoCs

As a validation of our flow, we present a NoC implementation based on a 2D 3x2 quasi-mesh (called simply mesh in the following) and migrate it to a 3D arrangement (Figures 2.7 and 2.8). The 3D mapping is achieved by splitting in two halves the mesh and overlapping them in separate chip layers, with communication achieved through TSVs. The stacked topology has exactly the same functionality of the bidimensional implementation.

As a first step, we leverage SunFloor (33) to instantiate the 2D mesh. There is no need to modify the RTL output of SunFloor in any way. Next, we identify the best partitioning for mapping onto the layer stack. This task is, at present, done manually, due to the large set of constraints involved. These include manufacturing limitations,

chip pinout, area considerations, bandwidth demands, thermal requirements, *etc.*. For example, our test 3x2 mesh connects three processors and three memories; since we assume that processors cannot be stacked on top of each other, to avoid the formation of hot spots, we interleave processors and memories. ×pipes links connect either two different switches or a switch and a network interface; our choice is to cut bidimensional topologies across switch-to-switch links, replacing the latter with an upstream and a downstream port.

Then we perform synthesis, placement and routing of the RTL in two separate runs, one per design partition. During placement, we insert TSV macros at the proper switch boundaries. We choose minimum TSV diameter (4 μm) and pitch achievable in current technologies. The area overhead of each Via is 64 μm^2 (8x8) and for a couple of swicth in/out port (e.g. UP or Down) we have $2(5 + Flith\_Width)$ Vias, where 5 is the number of control signals, $Flit\_Width$ is the flit width, and the multiplicative factor 2 is introduced since we consider a couple of ports (in and out). When Flit_Width is set to 28, the area overhead for a 6x6 Switch is about 6% on ACK/NACK and 9% on STALL/GO. In exchange for this small area cost, switches run around 10% faster and less buffering is required (saving up to 13% of the combinational area).

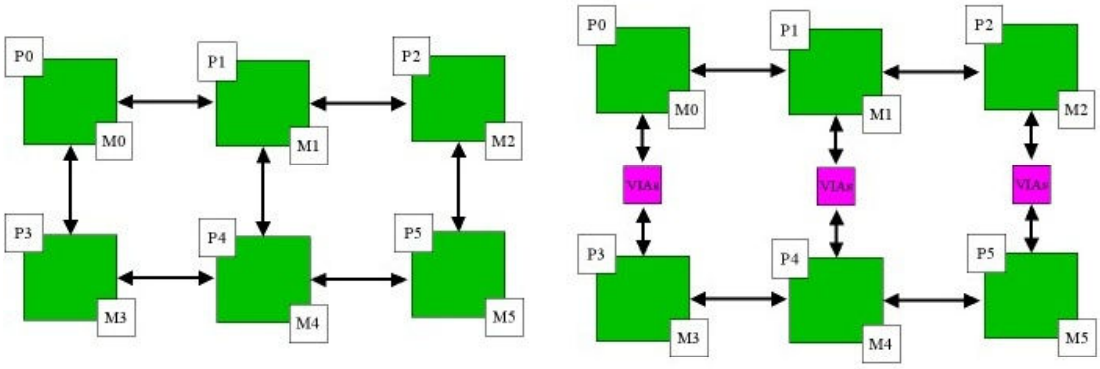


Figure 2.7: 2D 3x2 mesh NoC topology and one possible 3D re-implementation.

2.6 Conclusions

In this chapter, we have studied the performance and system-level impact of through-silicon vias as one of the possible ways to implement high-density vertical NoC links.

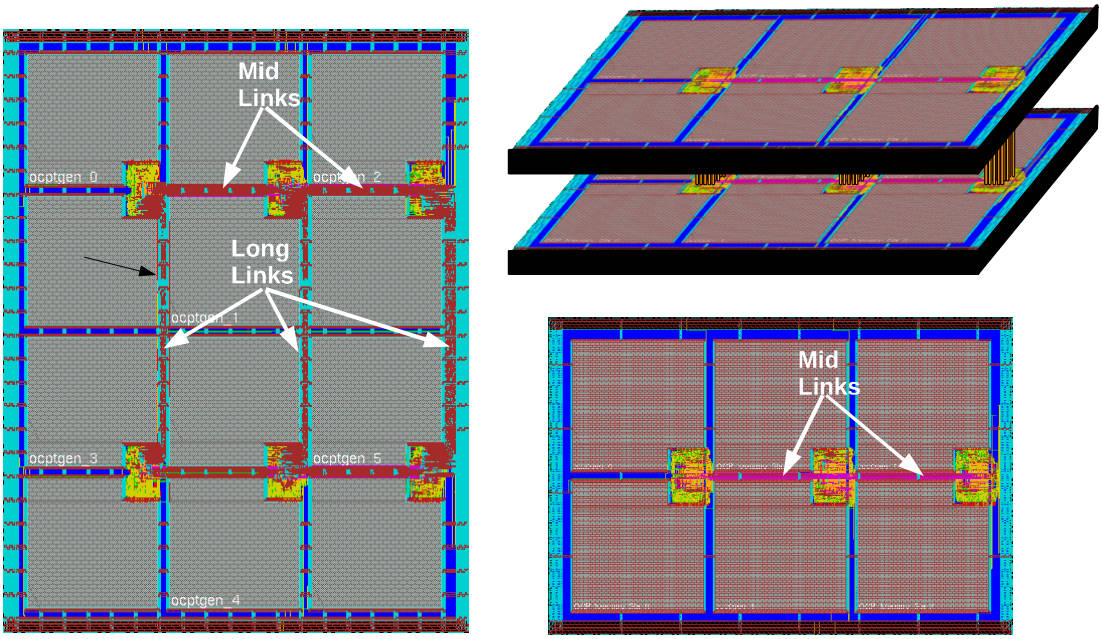


Figure 2.8: Layouts for (a) the 2D 3x2 mesh, (b) one of the halves of its 3D re-implementation, and (c) the 3D view of the two stacked halves (vertical axis not to scale).

We have shown that, even when accounting for the coupling effects in dense vertical link bundles, the parasitics associated with TSVs are one order of magnitude smaller than traditional horizontal wires, making 3D NoCs a very promising approach. We have shown how to design NoC switches with up/down ports. Finally, we have shown that our semi-automated flow is capable of generating layouts of 3D NoCs which are fully compatible with accurate post-layout timing, area and power analysis.

Bibliography

[1] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proceedings of the 38th Design Automation Conference*, June 2001, pp. 684–689. 6, 7

[2] L. Benini and G. De Micheli, "Networks on chips: A new SoC paradigm," *IEEE Computer*, vol. 35, no. 1, pp. 70 – 78, January 2002. 6, 7

[3] B. Rajendran, R. S. Shenoy, D. J. Witte, N. S. Chokshi, R. L. DeLeon, and G. S. Tompa, "Cmos transistor processing compatible with monolithic 3-d integration," in *Proc. VLSI Interconnection (VMIC)*, 2005, pp. 76–82. 7

[4] Ziptronix, *Ziptronix target vertical scalability*, 2005. 7

[5] S. Christiansen, R. Singh, and U. Gosele, "Wafer direct bonding: From advanced substrate engineering to future applications in micro/nanoelectronics," in *Proceedings of the IEEE*, December 2006, pp. 2060–2106. 7

[6] K. Lee, "Wafer-stacked package technology for high-performance system," in *RTI Int. technology Venture Forum*, 2005. 7

[7] S.Spiesshoefer and et al, "Z-axis interconnects using fine pitch, nanoscale through-silicon vias: Process development," in *Electronic Components and Technology Conference*, 2004. 7

[8] R. S. Patti, "Three-dimensional integrated circuits and the future of system-on-chip designs," *Proceedings of the IEEE*, vol. 94, no. 6, June 2006. 7

[9] A. W. Topol, J. D. C. La Tulipe, L. Shi, D. J. Frank, K. Bernstein, S. E. Steen, A. Kumar, G. U. Singco, A. M. Young, K. W. Guarini, and M. Ieong,

"Three-dimensional integrated circuits," *IBM Journal of Research and Development*, vol. 50, no. 4/5, pp. 491–506, July/September 2006. 7

[10] F. Karim, A. Nguyen, S. Dey, and R. Rao, "On-chip communication architecture for OC-768 network processors," in *Proceedings of the Design Automation Conference (DAC)*, 2001, pp. 678 – 683. 7

[11] A. Pullini, F. Angiolini, D. Bertozzi, and L. Benini, "Fault tolerance overhead in network-on-chip flow control schemes," in *Proceedings of the 18th Annual Symposium on Integrated Circuits and System Design (SBCCI)*, 2005, pp. 224–229. 7

[12] W. Hang-Sheng, Z. Xinping, P. Li-Shiuan, and S. Malik, "Orion: a power-performance simulator for interconnection networks," in *Proceedings of 35th Annual IEEE/ACM International Symposium on Microarchitecture (MICRO)*. IEEE/ACM, November 2002, pp. 294–305. 7

[13] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "QNoC: QoS architecture and design process for network on chip," in *Journal of Systems Architecture*. Elsevier, 2004. 7

[14] D. Wiklund and D. Liu, "SoCBUS: Switched network on chip for hard real time embedded systems," in *Proceedings of the International Parallel and Distributed Processing Symposium (IPDPS03)*. IEEE, 2003. 7

[15] T. Bjerregaard and J. Sparsø, "Scheduling discipline for latency and bandwidth guarantees in asynchronous network-on-chip," in *Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, 2005, pp. 34–43. 7

[16] A. Sheibanyrad, I. M. Panades, and A. Greiner, "Systematic comparison between the asynchronous and the multi-synchronous implementations of a network on chip architecture," in *Design, Automation & Test in Europe Conference & Exhibition*, April 2007, pp. 1–6. 7

[17] S. Furber and J. Bainbridge, "Future trends in soc interconnect," in *Proceedings of the International Symposium on System-on-Chip (SoC)*. IEEE Computer Society, 2005. 7

[18] S. Murali, M. Coenen, A. Radulescu, K. Goossens, and G. D. Micheli, "Mapping and configuration methods for multi-use-case networks on chips," in *Proceedings of the 2006 conference on Asia South Pacific design automation (ASP-DAC)*. New York, NY, USA: ACM Press, 2006, pp. 146–151. 7

[19] K. Srinivasan and K. Chatha, "A methodology for layout aware design and optimization of custom network-on-chip architectures," in *Proceedings of the 7th International Symposium on Quality Electronic Design (ISQED)*. IEEE Computer Society, 2006. 7

[20] A. Radulescu, J. Dielissen, K. Goossens, E. Rijpkema, and P. Wielage, "An efficient on-chip network interface offering guaranteed services, shared-memory abstraction, and flexible network configuration," in *Proceedings of the 2004 Design, Automation and Test in Europe Conference (DATE)*. IEEE, 2004. 7

[21] A. Andriahantenaina and A. Greiner, "Micro-network for SoC: Implementation of a 32-port SPIN network," in *The Proceedings of Design, Automation and Test in Europe Conference and Exhibition*. IEEE, 2003, pp. 1128– 1129. 7

[22] K. Lee, S.-J. Lee, S.-E. Kim, H.-M. Choi, D. Kim, S. Kim, M.-W. Lee, and H.-J. Yoo, "A 51mW 1.6GHz on-chip network for low-power heterogeneous SoC platform," in *Digest of Technical Papers of the 2004 IEEE International Solid-State Circuits Conference (ISSC)*. IEEE Computer Society, 2004, pp. 152–518. 7

[23] C. A. Zeferino and A. A. Susin, "SoCIN: A parametric and scalable network-on-chip," in *Proceedings of the 16th Symposium on Integrated Circuits and Systems Design (SBCCI03)*, 2003, pp. 34–43. 7

[24] P. T. Wolkotte, P. K. Holzenspies, and G. J. Smit, "Fast, accurate and detailed noc simulations," in *Proceedings of the First International Symposium on Networks-on-Chip (NOCS)*. IEEE Computer Society, 2007. 7

[25] F. Angiolini, P. Meloni, D. Bertozzi, L. Benini, S. Carta, and L. Raffo, "Networks on chips: A synthesis perspective," in *Proceedings of the 2005 ParCo Conference*, 2005. 7

[26] F. Angiolini, P. Meloni, S. Carta, L. Raffo, and L. Benini, "A layout-aware analysis of networks-on-chip and traditional interconnects for mpsocs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 3, pp. 421–434, March 2007. 7, 13

[27] V. F. Pavlidis and E. G. Friedman, "3-d topologies for networks-on-chip," in *Proceedings of the IEEE SoC Conference (SOCC)*. IEEE Computer Society, 2006, pp. 285–288. 8

[28] B. Feero and P. P. Pande, "Performance evaluation for three-dimensional networks-on-chip," in *Proceedings of the IEEE Annual Symposium on VLSI (ISVLSI)*. IEEE Computer Society, 2007, pp. 305–310. 8

[29] J. Kim, C. Nicopoulos, D. Park, R. Das, Y. Xie, N. Vijaykrishnan, M. S. Yousif, and C. R. Das, "A novel dimensionally-decomposed router for on-chip communication in 3d architectures," in *Proceedings of the 34th International Symposium on Computer Architecture (ISCA)*, 2007. 8

[30] S. Fujita, K. Nomura, K. Abe, and T. Lee, "3d on-chip networking technology based on post-silicon devices for future networks-on-chip," in *Nano-Networks and Workshops*, September 2006, pp. 1–5. 8

[31] A. Corp., "Q3d extractor," 2007, http://www.ansoft.com/products/si/q3d_extractor/. 8

[32] K. N. Chen, A. Fan, and C. S. T. ans R. Reif, "Contact resistance measurament of bonded copper interconnects for three-dimensional integration technology," *IEEE ELECTRON DEVICE LETTERS*, vol. 25, no. 1, January 2005. 12

[33] S. Murali, P. Meloni, F. Angiolini, D. Atienza, S. Carta, L. Benini, and G. D. Micheli, "Designing application-specific networks on chips with floorplan information," in *Proceedings of the 2006 International Conference on Computer-Aided Design (ICCAD)*. New York, NY, USA: ACM Press, 2006, pp. 355–362. 14, 16, 27

3

Synchronization in 3D NoCs

The Network-on-Chip (NoC) interconnection paradigm has been gaining momentum thanks to its flexibility, scalability and suitability to deep submicron technology processes. The next challenge is to use NoCs as the backbones of the upcoming generation of 3D chips, assembled by stacking multiple silicon layers. Multiple technical issues have to be tackled in this respect. One of the foremost is the unsuitability of a purely synchronous design style, as it is not straightforward to impose a strict bound on the clock skew among multiple clock trees across different layers. In this chapter, we present a scheme to handle mesochronous communication in 3D NoCs and analyze (i) the circuit design, (ii) the timing properties, (iii) the requirements to support flow control across mesochronous links, (iv) the implementation cost of such a scheme after placement and routing.

3.1 Introduction

Over the years, advances in silicon technology have enabled the integration of larger and larger amounts of processing elements and memories into chips, increasing processing performance. This trend is still desirable, but new technological hurdles may prevent designers from being able to sustain the current pace of miniaturization. Simultaneously, there has been a strong push towards the mixing of functional blocks which may require a variety of processing steps, such as plain CMOS, DRAM, MEMS, passive and active analog circuitry, optoelectronic elements, chemical sensors, actuators, *etc.*. Unfortunately, each extra manufacturing step increases costs and decreases

yield, imposing a limit on the heterogeneity of each silicon die. Vertically stacking multiple layers of silicon is an answer to both concerns: it represents a sustainable way of continuously adding functionality by the integration of more computing blocks, while pursuing design objectives such as ease of layer manufacturing, small package sizes, minimum footprints and modularity.

In planar implementations, interconnects are becoming a limiting factor to achieve design closure. This is due to several issues, such as the growing ratio of wire delay *vs.* logic delay, signal integrity concerns and stringent bandwidth requirements. At the system level, the key challenge is configuring, optimizing and verifying the communication architecture across many degrees of freedom in terms of topology, architecture and interface protocols. The Network-on-Chip (NoC) paradigm, which brings packet-switching networking concepts to the on-die level, has been proposed (1, 2) to systematically tackle these challenges. NoCs are a structured, predictable and scalable approach to the problem, centered around wire segmentation and point-to-point signaling.

3D manufacturing and NoCs represent clearly synergistic techniques. The structured nature of NoCs ideally encapsulates the design properties and requirements (such as heterogeneous wiring resources and architecture, large degree of parallelism, high routing and topological complexity) of three-dimensional integration of two-dimensional building blocks. However, the implementation of 3D NoCs is not completely straightforward, as testified by the research papers on the topic that have been appearing recently (3, 4, 5, 6, 7).

One of the most obvious issues is the handling of clock domains. A completely synchronous approach to NoC design is feasible, even at frequencies around 1 GHz, in 2D chips (8). On the other hand, minimizing the clock skew of a clock tree in a complex 3D structure becomes at least a challenging task. Two of the possible ways to implement a 3D clock tree are (i) the design of a separate clock tree per each 2D layer, and the insertion of a single vertical structure to which to attach all the tree roots, (ii) the design of a single clock tree in one of the layers, and the deployment of many vertical vias at the terminal nodes of this tree, thus distributing the clock to all the layers. According to (9, 10), solution (ii) is better in terms of power and resulting skew, but requires many more vertical vias, which is likely to be expensive or impractical. Further, since solution (ii) requires a large amount of connectivity among

the layers, it does not readily apply to the very desirable scenario where 3D chips are assembled by stacking layers provided by different vendors and possibly built with completely different processes. Since there is no clear solution to the problem of skew-free and modular clock distribution in 3D chips, the need for clock synchronization at the inter-layer boundaries is well motivated.

Several possible solutions to this issue are available. Totally asynchronous NoCs are, unfortunately, very complex to design, validate and implement. Generic dual-clock FIFOs could be deployed, but their high implementation cost suggests using them only where absolutely needed; for example, instead of using them inside of the NoC topology for mere synchronization among clusters of routers, it may be wiser to instantiate them only at the edges of the NoC, *e.g.* at the interface of a core which is able to perform frequency scaling. To achieve functionality and flexibility at the minimum cost, mesochronous schemes are probably the most effective. This paper focuses on the implementation of mesochronous adapters for 3D NoCs, with emphasis on circuit design, timing properties, flow control support, and implementation cost.

The rest of the paper is organized as follows. Section 2 presents the related work, given a overview of the possible techniques, Section 3 describes the reference NoC architecture, Section 4 describes the mesochronous Synchronizer architecture and timing analysis. Section 5 analyzes the physical implementation and experimental data, and finally in section 6 provides the conclusion and future work.

3.2 Related Work

A number of technologies for 3D chip manufacturing have been explored in recent years. In this chapter we focus on wafer stacking approaches, as one of the most promising avenues for the implementation of high-performance yet inexpensive (multiple 3D chips can be processed in a single pass) three-dimensional ICs. Wafer stacking relies on Through-Silicon Vias (TSVs) (11) for vertical connectivity, guaranteeing low parasitics (*i.e.* low latency and power) and, if needed, extremely high densities of vertical wires (*i.e.* high bandwidth). Tezzaron Semiconductor Corporation (12) and IBM Technologies (13) are active players in this field.

NoCs have been suggested as a scalable communication fabric (1, 2). CAD tools for NoC instantiation and optimization can be found for example in (14, 15). The

synthesis flow of NoCs has been explored by several groups, including full custom (16) (with actual test chips), FPGA targets (17, 18) and plain logic synthesis (19).

Some research is being undertaken on 3D NoCs. For example, in (3, 4), the authors focus on topologies (meshes, stacked meshes, *etc.*) and on performance metrics. In (5), the authors propose a dimension decomposition scheme to optimize the cost of 3D NoC switches, and present some area and frequency figures derived from a physical implementation. Post-silicon nano-scale 3D interconnections have also been recently investigated (6), but large scale availability of these technologies in the near future is uncertain.

A large body of research exists on asynchronous NoC design styles. For example, the CHAIN network (20) is completely based on clockless circuit design techniques. Other asynchronous NoC libraries include MANGO (21) and NEXUS (22). ANOC (23) is based on a Quasi-Delay-Insensitive circuit design. Specific network building blocks are presented for example in (24) and asynchronous link design is tackled in (25).

The main goals of asynchronous NoCs have traditionally been lower power consumption than synchronous alternatives, increased tolerance to delay variability, and reduced electromagnetic emissions (26). Despite all the research efforts, however, the actual physical implementations of asynchronous NoCs (27, 28) are few and limited in complexity (few millions of gates, $0.13\mu m$ technology). This is often attributed to the current lack of fully mature synthesis toolchains, simulation environments and testing infrastructures, hindering industrial implementations. Suitable component libraries are also very difficult to build and characterize.

Globally Asynchronous Locally Synchronous (GALS) approaches do not disrupt as much the existing design flows. GALS systems (29, 30, 31) attach together a number of synchronous building blocks, and provide asynchronous facilities for the inter-block communication. While some of the tool maturity issues mentioned above still hold, the encapsulation of mixed-clock concerns within well-defined boundaries, which can be validated separately, provides a more conservative, and possibly more promising, solution to the interconnection issue. Several ways to synchronize clock domains at the boundaries exist, such as interleaving pipeline registers, using dual-clock FIFOs, adding programmable delays (32), deploying synchronous-to-asynchronous wrappers (29). Although some of these solutions (for instance, dual-clock FIFOs) are very flexible, allowing for arbitrary clock frequencies in the sender

and receiver domain, they all have one or more drawbacks, ranging from robustness to implementation complexity, from high latency to large area overhead. Some solutions have instead been specifically tuned only for the relatively simpler problem of mesochronous signaling, and have therefore been focused on low complexity and ease of implementation in existing tool flows. Two recent papers (33, 34) both suggest to implement the boundary interface with a source-synchronous design style, and propose some form of ping-pong buffering to counter timing and metastability concerns. We improve on these papers by studying such synchronizers inside of a NoC layout for a 3D chip, and considering full duplex communication with flow control, as discussed later.

3.3 Reference NoC Architecture

To make our study realistic, we integrate it on the ×pipes (19) NoC library. This NoC infrastructure is best suited for several reasons. First, it provides facilities for arbitrary switch connectivity, allowing the designer to easily deploy topologies for any kind of 3D arrangement of computing cores. Second, it already leverages a semi-automated design flow (33) spanning from RTL description to layout-level verification; this makes it possible for us to explore and validate the design down to the placement and routing steps. Third, it provides an interesting case study due to its configurability in terms of flow control. ×pipes switches come in two radically different variants, conceived to best match two flow control protocols (see Figure 3.1). The first is ACK/NACK, a retransmission-based protocol featuring increased error resilience. The second is STALL/GO, a simple variant of credit-based flow control allowing for pipelined links to be transparently deployed. In the ACK/NACK case, output buffers need to be inserted within switches, since any transmitted packet should be stored for potential retransmission. This implies a hardware cost, but it also means that NoC links are enclosed between two clocked buffers at the sending and receiving ends. Hence, a whole clock period is available for signal propagation along the wires of the inter-switch links. In any case, the link length and the switch logic are decoupled by the output buffer.

In contrast, in STALL/GO, low switching latency and reduced buffer cost are the main goals. The ×pipes STALL/GO switches therefore adopt a lean architecture, where only switch inputs are buffered. In other words, the switch logic and the link propaga-

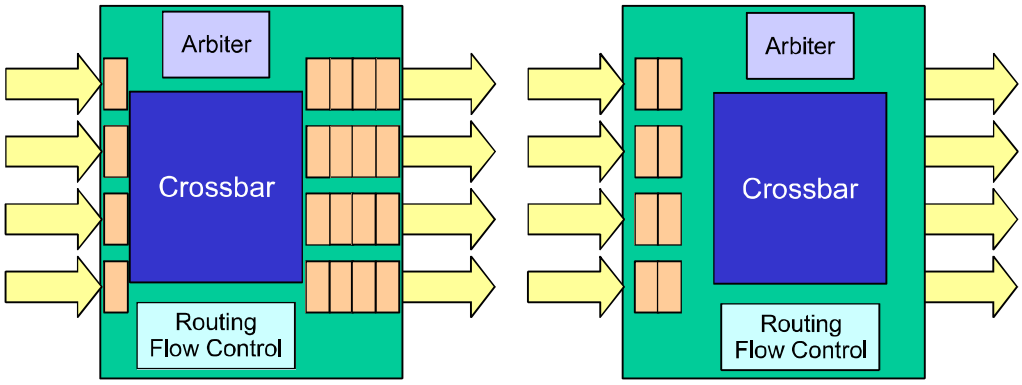


Figure 3.1: Block diagram of two switches, (a) with ACK/NACK (inputs and outputs are registered), (b) STALL/GO (only inputs are registered).

tion time (up to the following switch or to the first link pipeline stage) contribute to the same timing path, which can become the bottleneck for the system.

3.4 Architecture of a Mesochronous Synchronizer for 3D NoCs

In this chapter, we leverage the baseline architecture proposed in (33). This choice features substantial pros, including minimal complexity, ease of implementation in traditional design flows, and ability to function even during chip testing (which is typically performed at a lower frequency than the target operating one). It is important to notice, however, that the reference paper is aimed especially at handling mesochronous communication over very long and slow links (where it provides variation tolerance and high performance as additional benefits), but does not focus on short-range mesochronous synchronization, such as the one likely to be happening across 3D NoC vertical links. Therefore, it does not provide a sufficiently in-depth discussion about two issues that are crucial for any such implementation:

Timing margins, which are key to assessing circuit robustness and to the tuning of the low-level details of the design, are not studied in enough depth in a real NoC test case, therefore preventing the related optimizations.

Support for bidirectional communication, *i.e.* for flow control, is lacking. Mesochronous signaling is useless if proper backwards flow control cannot be issued.

Exploring and quantifying the tradeoffs required by these features is clearly key to assessing the viability of the overall approach.

3.4.1 Circuit Description

The proposed scheme is based on a synchronization circuit at the receiving end of a mesochronous link (see Figure 3.2 for a slightly simplified depiction) (33). The circuit receives as its inputs a bundle of NoC wires representing a regular NoC link, carrying data and/or flow control commands, and a copy of the clock signal of the sender. Since the latter wire experiences the same propagation delay as the data and flow control wires, it can be used as a strobe signal for them.

The circuit is composed of a *front-end* and a *back-end*. The front-end is driven by the incoming clock signal, and strobes the incoming data and flow control wires onto a set of parallel latches in a rotating fashion, based on a counter. The back-end of the circuit leverages the local clock, and samples data from one of the latches in the front-end thanks to multiplexing logic which is also based on a counter. The rationale is to temporarily store incoming information in one of the front-end latches, using the incoming clock wire to avoid any timing problem related to the clock phase offset. Once the information stored in the latch is stable, it can be read by the target clock domain and sampled by a regular flip-flop. The counters in the front-end and back-end are initialized upon reset, after observing the actual clock skew among the sender and receiver with a phase detector (33), so as to establish a proper offset. This is to guarantee that information can safely settle in the front-end latches before being sampled on the target domain clock. The phase detector only operates upon the system reset, but given the mesochronous nature of the link, its findings hold equally well during normal operation; the advantage is that power consumption in normal mode is negligible.

With respect to the baseline scheme (33), we apply several changes, tuning the architecture to the problem at hand. One clear feature of the 3D NoC scenario, for example, is that vertical inter-switch links are typically short and feature extremely small propagation delays, in the range of tens of picoseconds (7). Therefore, there is typically no need for the synchronizer to support multi-cycle propagation delays. As a result, one of the most notable architectural changes is the presence of only two latches, thus also dramatically simplifying the structure of the front-end and back-end counters

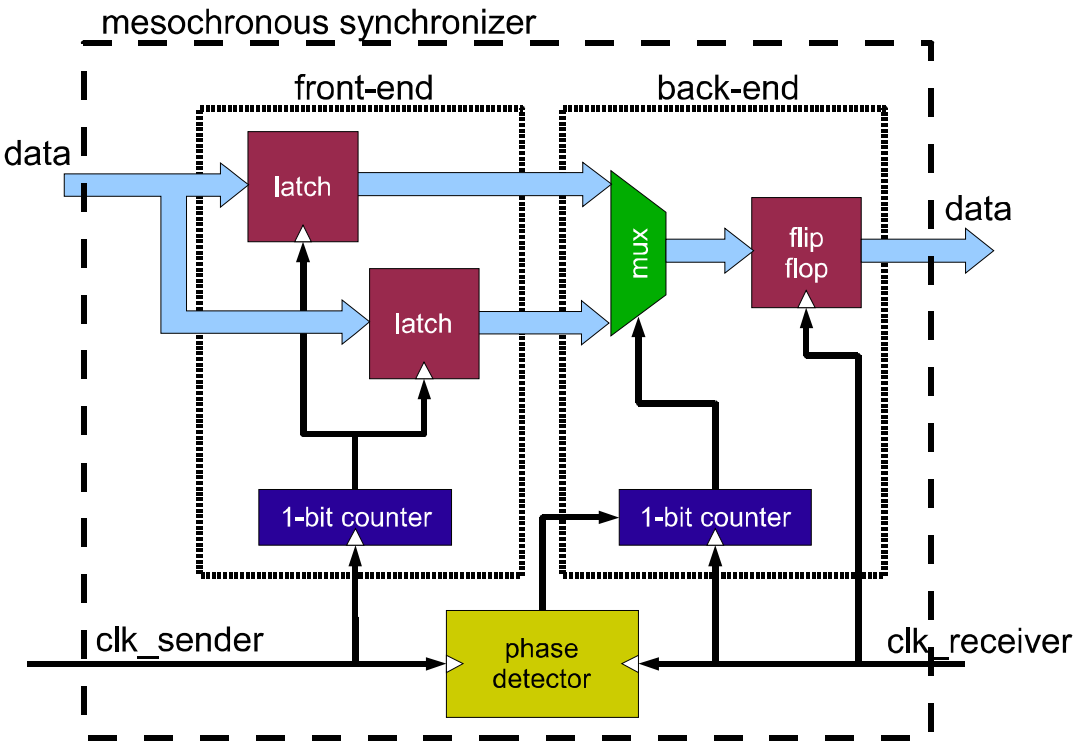


Figure 3.2: Proposed mesochronous synchronizer circuit.

to 1-bit elements. This change, which allows for large area savings, is allowed by the timing properties discussed in the following. Shall the need arise, more latches could still be deployed in case of a mesochronous link spanning over a very long distance, and requiring multiple clock cycles for signal propagation.

Figure 3.3 summarizes the intended configuration for a system with two layers and two vertical links, one going upwards, one going downwards. For each such link, one main synchronizer ("RX Synchronizer") must be deployed to adjust the incoming information to the new clock signal. Since few flow control wires are travelling backwards, a smaller "TX Synchronizer" is also needed to handle them.

3.4.2 Timing Margins of the Proposed Circuit

In order not to incur metastability and not to lose data within the mesochronous synchronizer, timing constraints must be met at two points in the circuit: (i) the front-end must latch incoming data safely, (ii) the back-end must sample incoming data when it is stable.

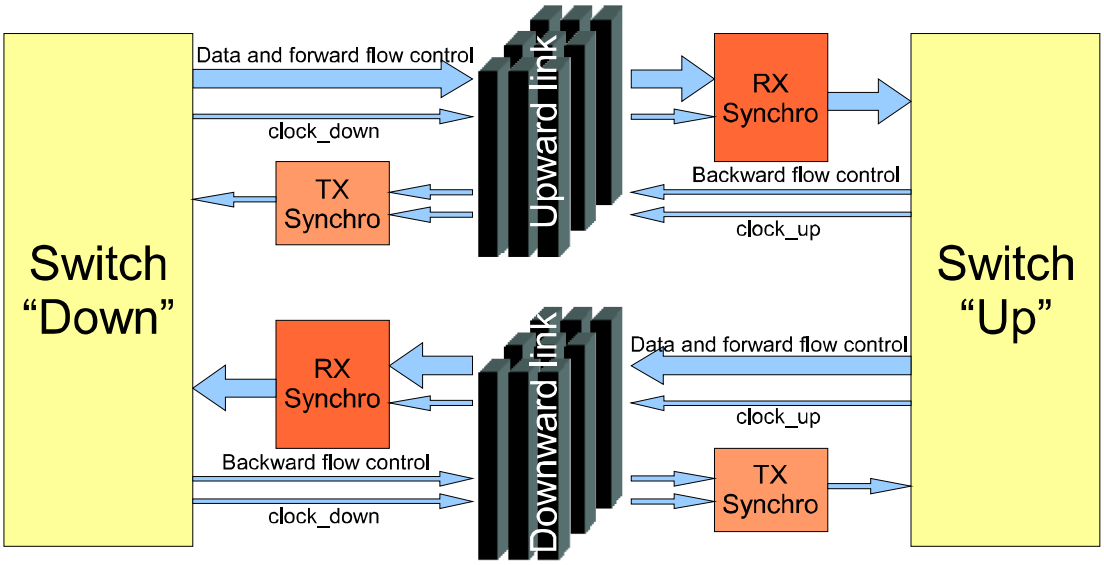


Figure 3.3: Proposed scheme for two-way synchronization across two layers.

To fulfill condition (i), the latches must become transparent at the right point in time. The ideal control signal `latch_enable` to do so would be perfectly aligned with the strobe clock `clk_sender`, upon which `data` is designed to be sampled. Unfortunately, such an ideal condition is impossible to reproduce. First, `clk_sender` must be conditioned by local signals in the mesochronous synchronizer (namely, the output `count` of the front-end counter), which introduces a delay t_{cond}. Second, `clk_sender` and `data` may not be perfectly in sync any more if the vertical link among the sending switch and the mesochronous synchronizer is not ideal, *e.g.* if the wires/vertical vias carrying `clk_sender` are slower than those carrying `data` by $t_{routingskew}$. This means that `latch_enable` has a worst-case offset, with respect to the ideal edge on which `data` should be sampled, of $t_{cond} + t_{routingskew}$; this is an advance if $t_{routingskew}$ is negative and larger than t_{cond} (`clk_sender` wires much faster than `data` wires), and a delay otherwise.

On the other hand, the good news is that `data` is not supposed to be switching extremely close in time to the clock edges of `clk_sender`. Even if `data` were to be the direct output of registers in the sending switch, it would still take the propagation time of a flip flop before any transition could be noticed. In practice, it is likely that output buffers in the sending switch may also have some additional logic downstream of such

registers, such as multiplexers to select the output of one of multiple buffer locations. Similarly, the `data` propagation delay must be designed to allow for at least a flip-flop setup time before the following clock edge, and probably a bit more to account for a bit of extra logic at the receiving buffer, such as multiplexers again. In general, the minimum transition delay of `data` after the previous clock edge of `clk_sender` can be called $t_{data_{min}}$, and the maximum can be called $t_{data_{max}}$.

In order to generate as robust a circuit as possible, we propose the circuit of Figure 3.4 to generate `latch_enable`; example waveforms are in Figure 3.5. This circuit is an improvement with respect to (33). Since two latches are enough to implement the front-end (see below), the counter is 1-bit, and therefore a single flip-flop, while the logic to check the counter output against a fixed value becomes a single XOR. The circuit evaluates the counter output `count` on the positive edges of `clk_sender`, but only asserts `latch_enable` when `clk_sender` goes low, *i.e.* half a clock cycle later. This shortens the critical path among `clk_sender` edges and `latch_enable` edges, *i.e.* t_{cond}, to the delay of a single NOR gate, irrespective of the delay of the counter and comparison logic - as long as these fit within a clock semiperiod, which is trivial. With this arrangement, the latches in the front-end are only transparent for one clock semiperiod every two clock periods. The conditions for correct functionality can then be summarized as:

$$t_{cond} + t_{routingskew} + t_{latchhold} < t_{data_{min}} \tag{3.1}$$

$$t_{clk} + t_{cond} + t_{routingskew} > t_{data_{max}} + t_{latchsetup} \tag{3.2}$$

$$t_{counter} + t_{comp} < \frac{t_{clk}}{2} \tag{3.3}$$

Equation 3.1 expresses the fact that `latch_enable` should come early enough not to let the following piece of `data` slip in the front-end latch by mistake. Equation 3.2 ensures that `latch_enable` comes late enough to actually let `data` settle down before latching it in the front-end. Finally, Equation 3.3 ensures that the critical path for the generation of `latch_enable` is indeed determined by the edges of `clk_sender` plus a NOR delay. Experimental results validating that these equations are actually holding will be presented in Section 3.5.2.

Condition (ii) is easy to fulfill given a proper initialization of the counters at reset. It is a degree of freedom whether to have the back-end sample data from the upper latch

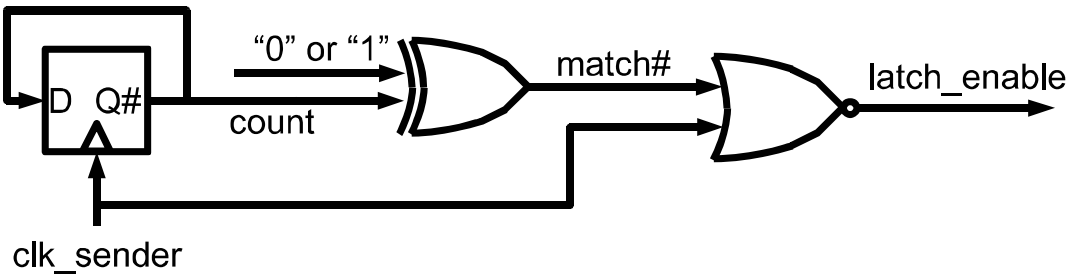


Figure 3.4: Circuit to generate the `3/latch_enable` control wire.

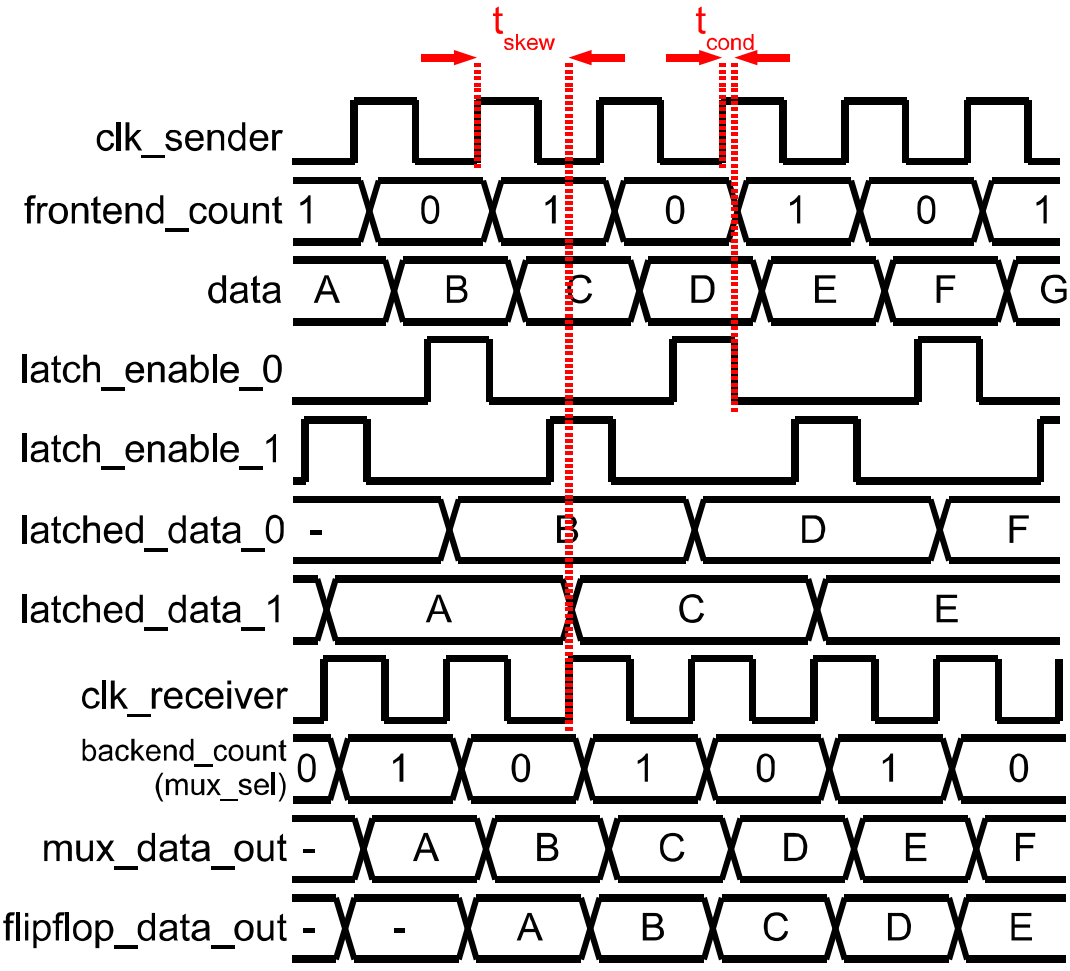


Figure 3.5: Example of the waveforms in the proposed synchronizer.

at "even" clock edges and the lower latch at "odd" clock edges, or vice versa, based on the initial value imposed to the back-end counter during reset. Since the latches in the front-end are transparent one semiperiod every two periods, and opaque (frozen) for the remaining three semiperiods, it is always possible to choose a counter setup where the sampling clock edge in the back-end captures the output of the latches in a stable condition, even accounting for a large timing margin to neutralize jitter. Please note that this discussion also proves that no more than two latches in parallel are needed in the front-end, at least as long as the link propagation time remains shorter than a single clock period.

3.4.3 Adding Support for Backwards Flow Control

A key open issue to understanding whether the circuit can be used to implement a useful link for a 3D NoC is to check the overhead it mandates for a design with flow control. In fact, a unidirectional mesochronous link is relatively straightforward to design; once bidirectional communication must be taken into account, the implementation details and the related resource overhead become crucial. We see two properties that the system must feature to define a proper implementation of flow control over mesochronous links: (i) the system must never incur data loss or corruption, (ii) if the receiver is not busy for independent reasons (such as contention for the same switch output port), the system must be able to sustain a transfer bandwidth of one flit per clock cycle.

The solution to be applied depends on the flow control deployed in the platform, but is anyway based on the main observation that the maximum added time to convey flow control signals across a vertical link, and to resynchronize them, is in any case less than two clock cycles. Based on this information, the following solutions can be envisioned.

3.4.3.1 Backwards Flow Control in ACK/NACK

In the ACK/NACK flow control, in absence of flow control information heading back (either ACKs or NACKs), the sender "optimistically" pushes flits out. Since a copy of each flit must be stored locally, the maximum number of outstanding flits is as many as the output buffer can hold. When flow control information is eventually received, in case of NACKs, old flits are resent; if, on the other hand, it is an ACK which makes

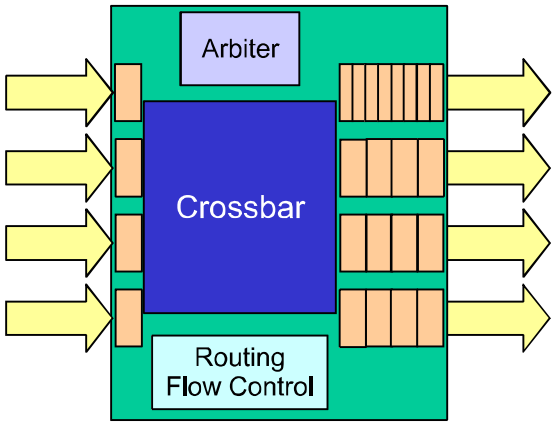


Figure 3.6: ACK/NACK modified switch block diagram. The port upstream of the vertical link has a deeper output buffer.

its way back to the sender, an old flit can be discarded from the output buffer, and a new one can be stored and sent.

Strictly speaking, the ACK/NACK flow control protocol does not require any corrective action to handle the timing changes introduced by a mesochronous synchronizer. The synchronizer merely delays the reception of flow control signals; this introduces no critical change in behaviour, and data safety is still guaranteed. However, changes need to be performed to support maximum bandwidth over the mesochronous link. The added latency of two clock cycles on each way (forwards and backwards) means that flits will reach their destination two cycles later, and ACKs will bounce back four cycles later than normal. To cope with this condition, output buffers in the sender need to be extended by four entries, *e.g.* from four (the minimum buffer depth to support maximum throughput in normal circumstances) to eight. This does not require any architectural change; a parameter adjustment in the output buffer is sufficient. A block diagram of the modified ACK/NACK switch is presented in Figure 3.6. The area cost related to this change will be presented in Section 3.5.3. No changes are required to the receiving switch, at the other side of the vertical link, unless a link in the opposite direction is also desired.

3.4.3.2 Backwards Flow Control in STALL/GO

In STALL/GO, flits are sent only as long as the STALL feedback wire is deasserted. This has two implications, which are the opposite of the ones for the ACK/NACK case.

On the one hand, if STALLs are never injected by the receiver, the sender never receives them, and full transmission bandwidth can always be sustained; no circuit change is needed to meet this criterion. On the other hand, data safety is critical. STALLs are the only way the receiver can withhold the flow of flits from the sender in case they cannot be processed (such as in case of lost arbitration for a switch output port). If STALLs cannot reach the sender in time, namely within one clock cycle, flits leaving the sender while the receiver is busy simply get lost.

To cope with this situation, we extend regular input buffers by two entries (from two, which is the minimum to provide full bandwidth, to four) and change their control logic. Instead of raising the STALL wire when the buffer is actually full, we raise it when two locations are still available. This approach is conservative; for example, a 4-deep STALL/GO buffer could in principle operate forever and at full bandwidth with three or four of its locations full, provided that, at each clock cycle, a flit can be extracted to make room for a new incoming one. However, if the same buffer were to be this full and were to experience further downstream congestion, there would simply be no way to notify the sender in time and to store the flits in flight anywhere. Thus, we choose instead to raise STALLs in advance, so that, by the time the sender is notified of the congestion, at most two flits are in flight, and they can still be stored. A block diagram of the modified STALL/GO switch is presented in Figure 3.7. The area cost related to this change will be presented in Section 3.5.3. No changes are required to the sending switch, at the other side of the vertical link, unless a link in the opposite direction is also desired.

3.5 Implementation and Experimental Results

3.5.1 Example Layout of Mesochronous Link Implementation

We synthesize the proposed circuit scheme with the UMC $0.13\mu m$ technology library and insert it into the floorplan for a 3D chip stack, then perform the routing. Figure 3.8 summarizes the result. It is possible to see the layout of the upper and lower layers. Both layers feature a switch and two NIs. Two obstructions model the vertical vias (one for the “Up” link, one for the “Down”) interconnecting the layers. The RX and TX synchronizers are wrapped just around the via bases, and are swapped among the layers. This layout is found to be very efficient from the area occupation viewpoint.

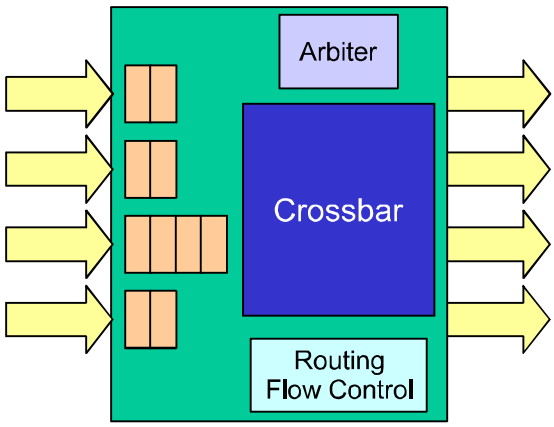
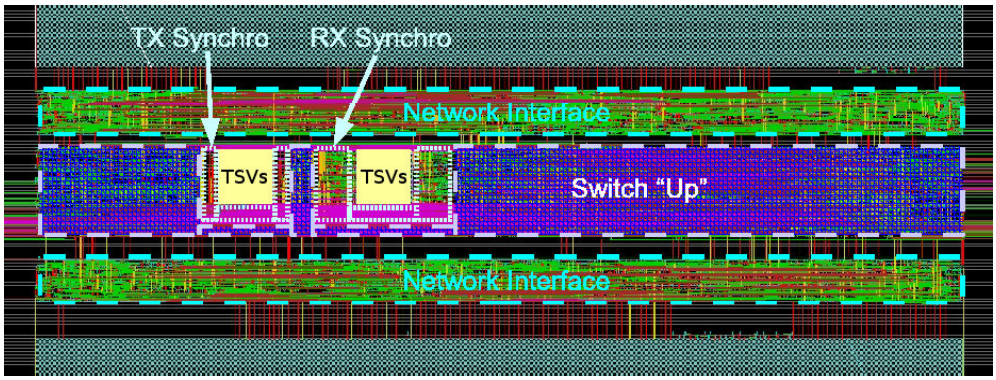
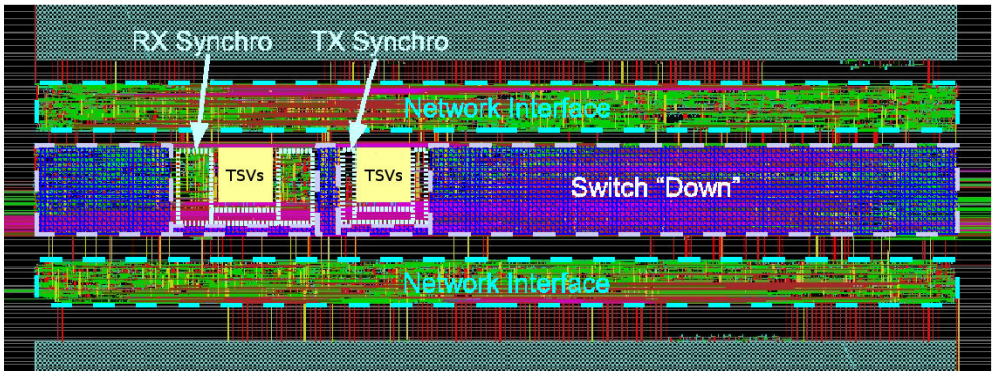


Figure 3.7: STALL/GO modified switch block diagram. The port downstream of the vertical link has a deeper input buffer and modified control logic.



(a) upper layer



(b) lower layer

Figure 3.8: Layout of a 3D chip stack with a mesochronous NoC link.

3.5.2 Timing Properties of the Mesochronous Synchronizer Front-End

In Section 3.4.2, a set of conditions to be fulfilled for proper operation have been presented. Our experimental results on a post-routing netlist show the following:

Equation 3.1 is easily fulfilled. Thanks to our optimized design (Section 3.4.2), we measure t_{cond} values of about $60ps$. The propagation time skew among different wires of a NoC link is very low, typically yielding a $t_{routingskew}$ below $20ps$. The typical latch hold time $t_{latchhold}$ is roughly $60ps$. On the other hand, for our NoC, we measure a $t_{data_{min}}$ of about $370ps$, irrespective of whether the flow control is ACK/NACK or STALL/GO. The constraint is therefore fulfilled. (Please note that $t_{data_{max}}$, however, is dependent on the chosen flow control due to the reasons explained in Section 3.3, and can be of up to $900ps$, imposing an operating frequency of $1GHz$ at most).

Equation 3.2 poses no issue. This is because the condition $t_{clk} > t_{data_{max}} + t_{latchsetup}$ is automatically met by any fully synchronous circuit. On the other hand, the term $t_{cond} + t_{routingskew}$, which appears because of the mesochronous synchronizer logic, never becomes negative in any of our test layouts. In other words, the propagation time difference among `data` and `clk_sender` is normally negligible, and in no case `clk_sender` is so much faster than `data` so as to more than offset t_{cond} (also see the bullet above). Therefore Equation 3.2 is always verified in our tests. Please note that, even in case of a violation of this condition, the circuit could still be made to work safely by slightly increasing t_{clk}, *i.e.* slightly decreasing the operating frequency.

Equation 3.3 is fulfilled by a very large margin. The typical clock period of our reference NoC is larger than $1ns$ in $0.13\mu m$ technology, yielding a semiperiod of at least $500ps$. Given the simplicity of the counter and comparator logic for a front-end with just two latches, we observe $t_{counter} + t_{comp}$ times of less than $200ps$, well within the desired range.

Given these results, the proposed architecture proves robust under all circumstances.

3.5.3 Silicon Cost of Proposed Synchronizer and Related Flow Control Adjustments

Figure 3.9 summarizes the overhead, in terms of area, for the implementation of the proposed mesochronous synchronization scheme. The numbers are computed for a post-routing circuit model. The "Synchronous" baseline comprises the area of two 32-bit 5x5 switches, with the minimum amount of buffering required for sustaining maximum throughput under no congestion, and that of the vertical obstruction required for a unidirectional vertical link (counted twice: once per chip layer). The "Mesochronous" figures add the area overhead for supporting mesochronous clocking over such a link, namely, the buffer depth increase in one of the switches and the two TX and RX Synchronizers. It is possible to notice that the synchronizers themselves feature minimal overhead, thanks to the drastic simplification in logic allowed by the implementation of 2-latch front-ends. The RX Synchronizer is about five times larger than the TX Synchronizer, since it must handle many more wires. The largest area overhead for mesochronous clocking support is within the switches themselves, and, as expected, is mostly accounted for in the sequential area budget. ACK/NACK incurs a much larger penalty than STALL/GO, since four extra buffers have to be deployed instead of just two. Overall, the global area overhead is about 13% of the baseline configuration in the STALL/GO scenario and about 40% of the baseline configuration in the ACK/NACK scenario. Especially in the STALL/GO case, the area cost is minimum and the implementation seems to be clearly affordable.

3.6 Conclusions

In this chapter, we have shown a detailed implementation of a mesochronous synchronizer for usage in a three-dimensional chip with a NoC backbone. In this context, since completely synchronous designs are hard or impossible to achieve, such a device is key to correct functionality. Starting from a baseline circuit scheme, we have customized it, verified its timing properties, added flow control facilities on top of the basic circuit, and assessed the area overhead of the whole. Key advantages of the baseline circuit have been kept, such as simplicity and ability to operate correctly even during chip test at a low frequency. The experimental results show that the proposed scheme is robust

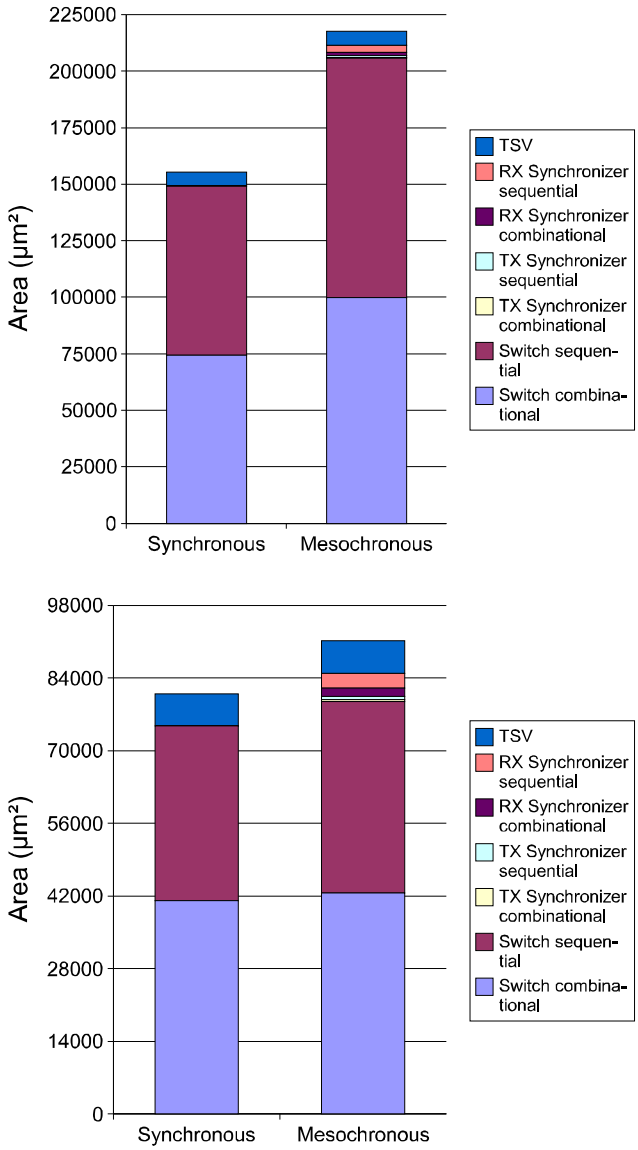


Figure 3.9: Area cost to implement mesochronous synchronization, (a) with ACK/NACK, (b) with STALL/GO.

and that its area cost is minimal, proving the viability of this architecture for 3D chip implementations based on NoCs.

Bibliography

[1] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proceedings of the 38th Design Automation Conference*, June 2001, pp. 684–689. 24, 25

[2] L. Benini and G. De Micheli, "Networks on chip: a new SoC paradigm," *IEEE Computer*, vol. 35, no. 1, pp. 70–78, January 2002. 24, 25

[3] V. F. Pavlidis and E. G. Friedman, "3-D topologies for networks-on-chip," in *Proceedings of the IEEE SoC Conference (SOCC)*. IEEE Computer Society, 2006, pp. 285–288. 24, 26

[4] B. Feero and P. P. Pande, "Performance evaluation for three-dimensional networks-on-chip," in *Proceedings of the IEEE Annual Symposium on VLSI (ISVLSI)*. IEEE Computer Society, 2007, pp. 305–310. 24, 26

[5] J. Kim, C. Nicopoulos, D. Park, R. Das, Y. Xie, N. Vijaykrishnan, M. S. Yousif, and C. R. Das, "A novel dimensionally-decomposed router for on-chip communication in 3d architectures," in *Proceedings of the 34th International Symposium on Computer Architecture (ISCA)*, 2007. 24, 26

[6] S. Fujita, K. Nomura, K. Abe, and T. Lee, "3d on-chip networking technology based on post-silicon devices for future networks-on-chip," in *Nano-Networks and Workshops*, September 2006, pp. 1–5. 24, 26

[7] I. Loi, F. Angiolini, and L. Benini, "Supporting vertical links for 3d networks-on-chip: Toward an automated design and analysis flow," in *Proceedings of the Nano-Net Conference 2007*, 2007. 24, 29

[8] F. Angiolini, P. Meloni, S. Carta, L. Raffo, and L. Benini, "A layout-aware analysis of networks-on-chip and traditional interconnects for MPSoCs," *IEEE Transactions on Computer-Aided Design*, vol. 26, no. 3, pp. 421–434, March 2007. 24

[9] C. A. Mineo, "Clock tree insertion and verification for 3D integrated circuits," North Carolina State University at Raleigh, Tech. Rep., 2005. 24

[10] M. Mondal, A. J. Ricketts, S. Kirolos, T. Ragheb, G. Link, N. Vijaykrishnan, and Y. Massoud, "Thermally robust clocking schemes for 3D integrated circuits," in *Proceedings of the 2007 Design, Automation and Test in Europe conference (DATE)*. New York, NY, USA: ACM Press, 2007, pp. 1206–1211. 24

[11] S.Spiesshoefer and et al, "Z-axis interconnects using fine pitch, nanoscale through-silicon vias: Process development," in *Electronic Components and Technology Conference*, 2004. 25

[12] R. S. Patti, "Three-dimensional integrated circuits and the future of system-on-chip designs," *Proceedings of the IEEE*, vol. 94, no. 6, June 2006. 25

[13] A. W. Topol, J. D. C. La Tulipe, L. Shi, D. J. Frank, K. Bernstein, S. E. Steen, A. Kumar, G. U. Singco, A. M. Young, K. W. Guarini, and M. Ieong, "Three-dimensional integrated circuits," *IBM Journal of Research and Development*, vol. 50, no. 4/5, pp. 491–506, July/September 2006. 25

[14] S. Murali, M. Coenen, A. Radulescu, K. Goossens, and G. D. Micheli, "Mapping and configuration methods for multi-use-case networks on chips," in *Proceedings of the 2006 conference on Asia South Pacific design automation (ASP-DAC)*. New York, NY, USA: ACM Press, 2006, pp. 146–151. 25

[15] K. Srinivasan and K. Chatha, "A methodology for layout aware design and optimization of custom network-on-chip architectures," in *Proceedings of the 7th International Symposium on Quality Electronic Design (ISQED)*. IEEE Computer Society, 2006. 25

[16] K. Lee, S.-J. Lee, S.-E. Kim, H.-M. Choi, D. Kim, S. Kim, M.-W. Lee, and H.-J. Yoo, "A 51mW 1.6GHz on-chip network for low-power heterogeneous SoC platform," in *Digest of Technical Papers of the 2004 IEEE International Solid-State Circuits Conference (ISSC)*. IEEE Computer Society, 2004, pp. 152–518. 26

[17] C. A. Zeferino and A. A. Susin, "SoCIN: A parametric and scalable network-on-chip," in *Proceedings of the 16th Symposium on Integrated Circuits and Systems Design (SBCCI03)*, 2003, pp. 34–43. 26

[18] P. T. Wolkotte, P. K. Holzenspies, and G. J. Smit, "Fast, accurate and detailed noc simulations," in *Proceedings of the First International Symposium on Networks-on-Chip (NOCS)*. IEEE Computer Society, 2007. 26

[19] F. Angiolini, P. Meloni, S. Carta, L. Raffo, and L. Benini, "A layout-aware analysis of networks-on-chip and traditional interconnects for mpsocs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 3, pp. 421–434, March 2007. 26, 27

[20] J. Bainbridge and S. Furber, "Chain: a delay-insensitive chip area interconnect," *IEEE Micro*, vol. 22, no. 5, pp. 16–23, Sep/Oct 2002. 26

[21] T. Bjerregaard and J. Sparsø, "Scheduling discipline for latency and bandwidth guarantees in asynchronous network-on-chip," in *Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems (ASYNC)*, 2005, pp. 34–43. 26

[22] A. Lines, "Asynchronous interconnect for synchronous soc design," *IEEE Micro*, vol. 24, no. 1, pp. 32–41, Jan-Feb 2004. 26

[23] E. Beigne, F. Clermidy, P. V. A. Clouard, and M. Renaudin, "An asynchronous noc architecture providing low latency service and its multi-level design framework," in *11th IEEE International Symposium on Asynchronous Circuits and Systems*, 2005, pp. 54–63. 26

[24] D. Rostislav, V. Vishnyakov, E. Friedman, and R. Ginosar, "An asynchronous router for multiple service levels networks on chip," in *Proceedings of the 11th IEEE International Symposium on Asynchronous Circuits and Systems*, 2005, pp. 44–53. 26

[25] E. Nigussie, J. Plosila, and J. Isoaho, "Delay-insensitive on-chip communication link using low-swing simultaneous bidirectional signaling," in *Proceedings of the 2006 Emerging VLSI Technologies and Architectures (ISVLSI06)*, 2006, p. 217. 26

[26] E. Grass, F. Winkler, M. Krstic, A. Julius, C. Stahl, and M. Piz, "Enhanced gals techniques for datapath applications," in *PATMOS*, 2005, pp. 581–590. 26

[27] M. B. Stensgaard, T. Bjerregaard, J. Sparso, and J. H. Pedersen, "A simple clockless network-on-chip for a commercial audio DSP chip," in *Proceedings of the 9th EUROMICRO Conference on Digital System Design*, 2006, pp. 641–648. 26

[28] L. A. Plana, W. J. Bainbridge, and S. B. Furber, "The design and test of a smartcard chip using a CHAIN self-timed network-on-chip," in *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*, 2004, p. 274. 26

[29] M. Krstic, E. Grass, C. Stahl, and M. Piz, "System integration by request-driven gals design," *IEE Proceedings on Computers and Digital Techniques*, vol. 153, no. 5, pp. 362–372, September 2006. 26

[30] D. Lattard and et al, "A telecom baseband circuit-based on an asynchronous network-on-chip," in *Proc. of the International Solid State Circuits Conference, ISSCC2007*, 2007, pp. 258–259. 26

[31] K. Lee, S.-J. Lee, and H.-J. Yoo, "Low-power network-on-chip for high-performance soc design," *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, vol. 14, no. 2, pp. 148–160, February 2006. 26

[32] S.-J. Lee, "Cost-optimization and chip implementation of on-chip network," KAIST, Tech. Rep., 2005. 26

[33] M. Ghoneima, Y. Ismail, M. Khellah, and V. De, "Variation-tolerant and low-power source-synchronous multicycle on-chip interconnection scheme," *VLSI Design*, vol. 2007, 2007. 27, 28, 29, 32

[34] D. Mangano, R. Locatelli, A. Scandurra, C. Pistritto, M. Coppola, L. Fanucci, F. Vitullo, and D. Zandri, "Skew insensitive physical links for network on chip," in *1st International Conference on Nano-Networks (NanoNet)*, 2006, pp. 1–5. 27

[35] S. M. et all, "Designing message-dependent deadlock free networks on chips for application-specific systems on chips," in *VLSI-SoC*, 2006, pp. 158–163.

4

3D NoCs fault tolerant links

Three-dimensional die stacking integration provides the ability to stack multiple layers of processed silicon with a large number of vertical interconnects. Through Silicon Vias (TSVs) provide a promising area- and power-efficient way to support communication between different stack layers. Unfortunately, low TSV yield significantly impacts design of three-dimensional die stacks with a large number of TSVs. This chapter presents a defect-tolerance technique for TSVs-based multi-bit links through an efficient and effective use of redundancy. This technique is ideally suited for three-dimensional network-on-chip (NoC) links. Simulation results demonstrate significant yield improvement, from 66% to 98%, with a low area cost (17% on a vertical link in a NoC switch, which leads a modest 2.1% increase the total switch area) in 130nm technology, with minimal impact of VLSI design and test flows.

4.1 Introduction

For future integrated system design, two major trends are emerging. Communication-centric architectures based on the Network on Chip (NoC) design paradigm (1, 2) to tackle interconnect and architectural scalability challenges. Three-Dimensional Integrated Circuits (3DICs) that provide a promising technological solution to alleviate the interconnect, I/O bandwidth and latency bottlenecks.

3DICs may enable heterogeneous integration and new classes of applications through significantly improved performance and energy efficiency of complex system architectures (e.g. technologies from Tezzaron Semiconductor Corporation (3), IMEC, MIT

Lincoln Labs, and IBM (4)). One of the most promising technologies for 3D integration is based on Through Silicon Vias (TSVs), which cut across thinned silicon substrates to establish inter-die connectivity after die-bonding.

Three-Dimensional Network on Chips (3DNoCs) combine the benefits of short vertical interconnects of 3DICs and the scalability of NoCs. 3DNoCs support both horizontal and vertical links. A vertical link can be physically implemented as a cluster of TSVs. TSVs allow fine pitch, high density and high compatibility with the standard CMOS process. Unfortunately, currently available processes for TSV fabrication have relatively low yield (compared to standard 2D processes). Figure 4.1 shows limited yield of TSVs from three different process technologies: HRI (5), IMEC (6) and IBM (7).

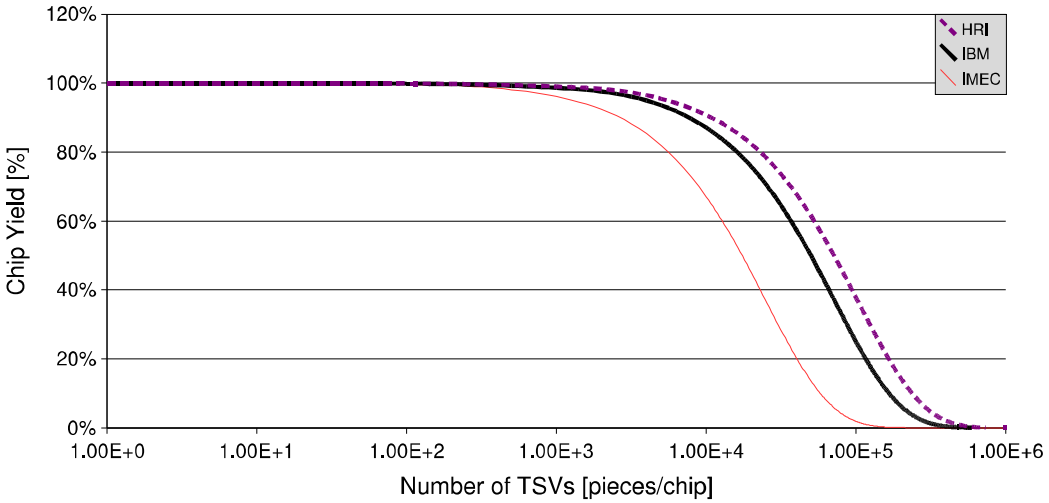


Figure 4.1: Yield trend for TSVs in three different processes: IBM, HRI and IMEC. Only random (complete or partial) open defects are considered in this figure, since misalignments are well controlled during the bonding phase. Yield is evaluated using the Poisson distribution.

In this chapter, we describe the design of a defect-tolerant TSV-based multi-bit vertical link which enables significant yield improvement with respect to random (complete or partial) open defects at an extremely low cost. Like traditional defect-tolerance techniques (such as those used for memories), our technique also relies on redundancy. Our major contribution is in a simple and efficient design of such a defect-tolerant TSV-based link at lowest cost, and also with minimal impact on the overall integrated system design and production test flows. While this TSV-based link design is generally applicable for both NOC-based and bus-based 3D interconnects, it is especially useful

for 3DNoCs because it takes advantage of the NoC switch architecture to introduce minimal system-level area impact.

The main contributions of this chapter are:

Introduction of a robust, defect-tolerant, vertical link architecture (for TSVs) to overcome challenges of low yield for current TSV fabrication processes;

Integration of the defect-tolerant 3D link into a complete three-dimensional Network on Chip design flow;

Experimental evaluation, performed at the layout level, including full placement and routing, to evaluate benefits, feasibility and hardware costs.

In our experiments, we achieve significant yield improvements (from 66% to 98% for 4.2M TSVs design, arranged in 100K spots made up of 42 TSVs each) for random (complete and partial) open defects that pose major challenges for TSVs. Our layout results demonstrate the feasibility of this approach and its low cost (17% on a vertical link in a NoC switch, which leads a modest 2.1% increase in the switch area).

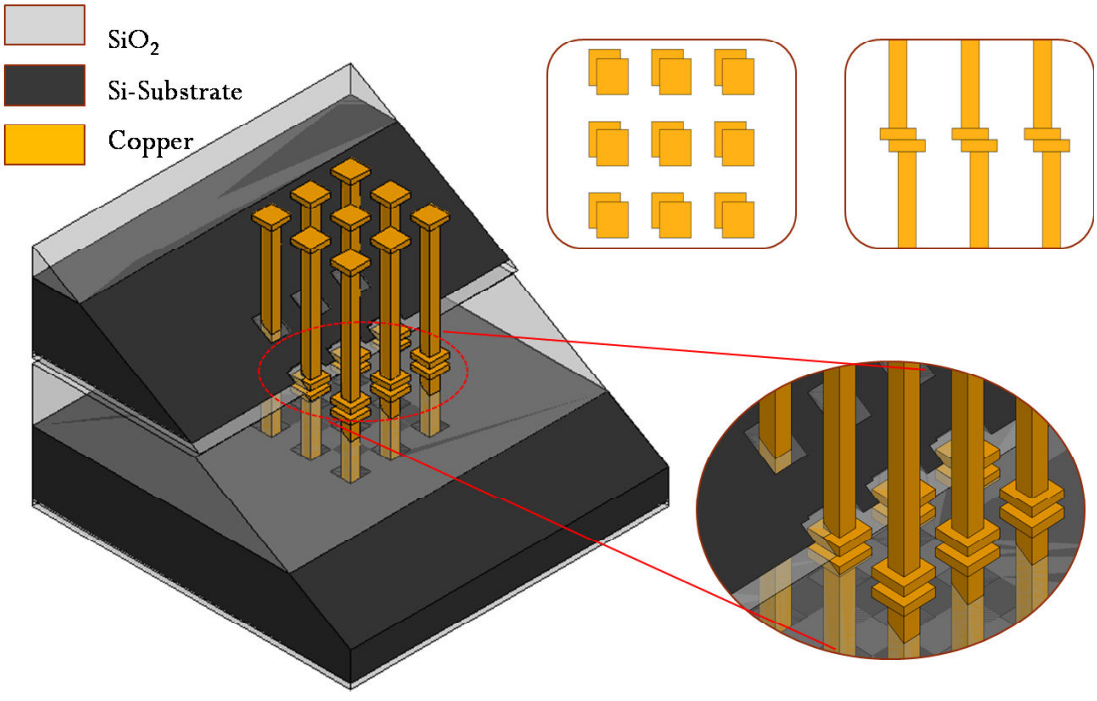


Figure 4.2: Cross-section of a vertical link across two tiers. The figure also shows the worst-case misalignment scenario

4.2 Related Work

Interconnect scaling has become one of the most crucial challenges in chip design, and is expected to get worse in the future. 3D integration and Network on Chip design methodologies are expected to overcome many of these challenges. NoCs have been suggested as a scalable communication fabric (1, 2). 3D integration has been proposed in different ways (e.g. Tezzaron Semiconductor Corporation (3), IMEC, MIT Lincoln Labs, and IBM Technologies (4)) providing promising solutions to enable connectivity along the vertical direction.

Recently, some research has been undertaken on 3DNoCs. In (8), the authors propose a dimension decomposition scheme to optimize the cost of 3D NoC switches, and present some area and frequency figures derived from a physical implementation. Post-silicon nano-scale 3D interconnections have also been recently investigated (9, 10), but large scale availability of these technologies in the near future is uncertain.

As technology scales, fault tolerance is becoming a key concern in on-chip communication. Optical Proximity Correction (OPC) and redundant via placement (11) have solved a huge number of cases of faults related, mainly, to interconnects.

Recent experiments by HRI on 3DICs report very high yields of over 60%, and the redundancy scheme used realizes each vertical interconnect as a pair of vias (twins) (5).

Despite the research undertaken on 3DICs and recently on 3DNoCs, to date, yield improvements for vertical links of 3DNoCs have never been studied. In this paper, we propose a novel scheme to overcome this limitation. The starting point of this work is (12, 13), where a thorough physical and timing analysis of the vertical links has been conducted on a real 3DNoC. Further, it is worth stressing that the proposed scheme can also be applied successfully to alternative interconnection schemes, such as buses.

4.3 Physical Level Modeling and Analysis of TSV Fault Impact

In this paper we focus on the wafer stacking approach since it is very promising for the implementation of high-performance yet inexpensive 3DICs. Wafer stacking relies on Through-Silicon Vias (TSVs) (14) for vertical connectivity, guaranteeing low parasitics (i.e. low power and propagation delay) and, if needed, extremely high densities of

vertical wires (i.e. high bandwidth-per-area ratio). The electrical connectivity between different tiers is provided by creating pads on the wafer surface, and then performing bonding by mechanical thermo-compression.

The primary failure mechanisms for TSVs are misalignment and random (complete or partial) open defects (15). Misalignment refers to unsuccessful wafer alignment prior to and during wafer bonding process (Figure 4.2), and is caused by shifts of bonding pads with respect to their nominal positions. Random defects comprise a variety of unpredictable physical phenomena related to the thermal compression process used in wafer stacking.

Starting from these considerations and based on (12), we have conducted a detailed study to quantify the impact of TSV failures on overall chip yield. We use an electrical model of TSVs and the bonding mechanisms for this purpose. Figure 4.3 shows the electrical model of two stacked vias (rendered as a T network). The vias are driven by one inverter followed by a stretch of planar interconnect (global routing). The contact resistance is related to the quality and area of bonding.

In case of misalignments (e.g. top wafer shifts along the X or Y axes or a small rotation), the bonded area decreases. This phenomenon has been modeled as a variable resistance (central resistor in Figure 4.3) between the two T networks, and the outcome is summarized in Table 4.1. As can be seen, misalignments of even noticeable entity do not normally compromise functionality (which is dominated by the overall planar routing parasitics (12)) and have a minimum impact on delay. Extreme misalignment, like in the last row of table 4.1 are highly unlikely in state-of-the-art wafer bonding processes (3, 4, 16). This motivates special emphasis on workarounds for the other main source of yield losses: random defects.

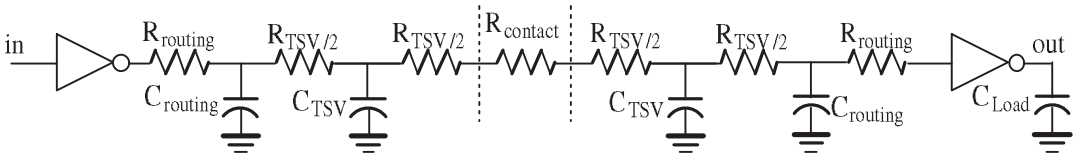


Figure 4.3: TSVs and global wire electrical model for two stacked vias(refer to Figure 4.2)

Random (complete or partial) open defects affect single vias or a small area of the interface because of failure mechanisms such as dislocations, 0_2 trapped on the surface, void formation, or even mechanical failures in TSVs (4, 17, 18, 18, 19). To model the

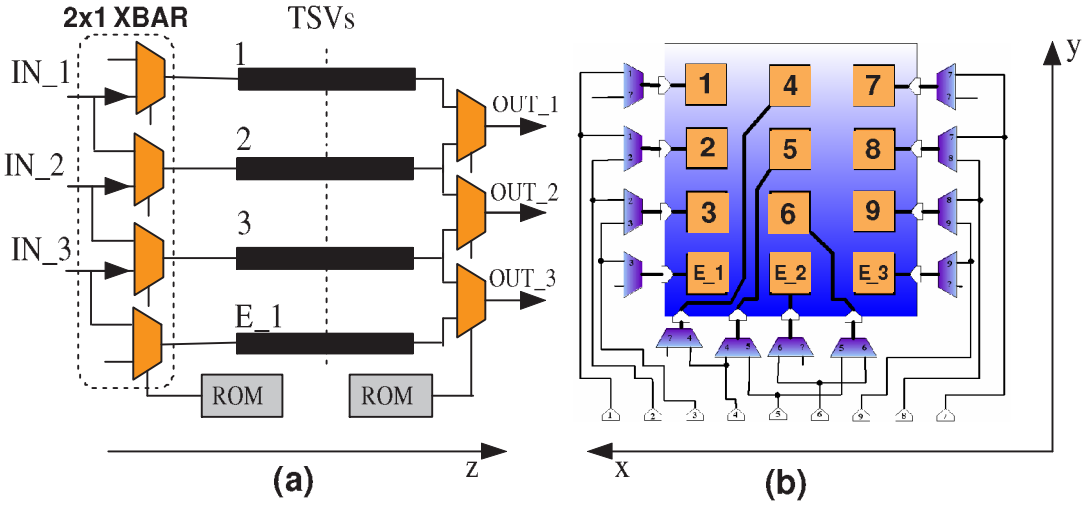


Figure 4.4: Redundant Routing scheme. (a) shows a simplified crossbar scheme for dynamic routing (functional scheme). (b) shows the TSVs obstruction and the routing crossbar (the orange squares are the TSV pads). Extra pads (E_1 E_2 ...) are spread around the TSV cluster, simplifying fault bypassing by means of a 2X multiplexers.

effects of these defects, we assumed a uniform TSV defect distribution and performed several Monte Carlo simulations. Based on our results (Section 4.5), we concluded that random (complete or partial) open defects are far more relevant compared to misalignment problems. For this reason, we focus on these defects in the following sections.

4.4 Yield Enhancements for 3DNoCs

In this section, we describe the target 3DNoC (12, 13) used for our experiments, and present our defect-tolerant solution for TSV-based vertical link design. As pointed out earlier, our solution can be applied not only to 3DNoCs, but also, more generally, to regular structures such as buses.

4.4.1 The reference NoC architecture

To make our study realistic, we developed our approach within the ×pipes (12, 13, 21) NoC library. To enable our NoC for 3D technology we extended the ×pipes switches by adding a couple of vertical ports, and we developed hard macros for the TSVs obstruction (12). Vertical links are unidirectional, and are composed (as planar links)

Misalignment [μm] in X-Y	Contact Area [μm^2]	Contact Resistance [Ω]	Δ Delay [%]
0	4x4	10m	0
1	3x3	19m	$< 1\%$
2	2x2	40m	$< 1\%$
3	1x1	160m	$< 1\%$
3.98	0.02x0.02	1K	22%

Table 4.1: Pad Contact resistance and delay increasing for CU-CU wafer metal bonding under different misalignment cases (17, 20)

of data and flow control signals, traveling in opposite directions. For this work we selected a data width of 32 bit, therefore for a pair of 3D links, 76 different signals are needed overall.

4.4.2 Yield Enhancement Approaches

Among the numerous techniques to increase wafer yield of VLSI designs, we focus on hardware redundancy, deployed at design time, with some amount of post-manufacturing configuration. We use active redundancy in the form of spare pads and reconfigurable routing hardware in order to minimize the overall complexity, while gaining maximum benefits in term of efficiency (Figure 4.4).

The dynamic routing solution is designed to leverage post-manufacturing configurability of the TSV interconnect map. This allows us to achieve high yield while minimizing the overhead in terms of the number of pads and extra logic. Combining testing resources (e.g., scan chains [1]) with such reconfigurability plays the key role in achieving yield. This solution allows us to test each vertical interconnect and diagnose defects, to isolate any failed TSV, and finally to restore functionality through reconfiguration by routing the affected signals over to the spare pads.

As we see in Figure 4.4 (a), in our proposed Dynamic Routing scheme, all pads are driven by a 2 1 crossbar, and each signal can be routed to two different TSVs. We explore configurations with one extra pad for each cluster (i.e. for each pad column). The crossbar is extremely small, as a strategic choice to keep the area overhead as low as possible - for each additional re-routing degree of freedom, the crossbar radix increases by a factor of one. With this lean architecture, faults are recovered by shifting

[1] The use of scan chains does not normally imply any extra cost, as they are typically integrated in every design

affected signals to the neighboring pads, and further shifting the displaced connections over to other adjacent pads until all connections are across safe electrical structures. To clarify the recovery scheme, we shall consider Figure 4.4 (b). Supposing that pad 2 is affected by some defects (resulting e.g. in an open circuit), we route signal 3 normally through its associated pad 3, while signal 2 gets rerouted through pad 1, and therefore signal 1 gets remapped to pad E_1. Signals outside this column are not shifted since the defect is contained inside the first cluster; the recovery process is performed locally. The proper routing information is elaborated off-chip (to minimize hardware complexity and overhead) during chip testing, and is then stored on-chip into a small One Time Programmable (OTP) memory (e.g. a fuse ROM).

The importance of the testing stage is evident, as it determines all the necessary inputs to correctly set the crossbar up. To test the physical interconnect, we reuse the scan chains which are normally inserted anyway in the design, thus incurring no overhead for this. Figure 4.5 illustrates the hardware facilities used to test the TSVs. The TSVs are tested by injecting Test Vectors (TVs) in one tier (e.g. the bottom one). The TV is propagated to the destination tier (e.g. the top one), where it is captured and transmitted off-chip. In summary, the approach is split into five steps:

1. Inject test vectors (e.g. bottom tier);
2. Propagate test vectors across TSVs and capture them (e.g. top tier);
3. Scan out the captured data (e.g. top tier);
4. Elaborate off-chip the interconnect map;
5. Reconfigure the crossbar (both bottom and top tier);

The process can be performed at any speed allowed by the external I/O pins. Since the interconnect map is devised off-chip, minimal logic is required on-chip for the mapping procedure - mostly, the OTP memory to store the crossbar configurations.

4.5 Experimental Results

4.5.1 Yield and Hardware Cost of the Redundant Solutions

The alternative solutions, and a non-redundant baseline case, have been synthesized with the UMC 130nm technology library and inserted into the floorplan for a 3D

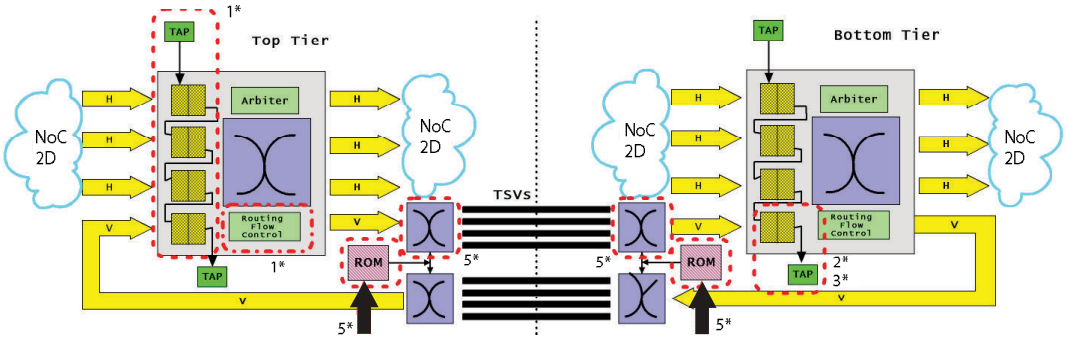


Figure 4.5: TSV NoC Test Environment: in test mode, test vectors are injected from the Test Access Point (1*) into the switch input buffer (scan), then the path through the crossbar is enabled (1*) and flow control is disabled. After some cycles the stimuli reach the next tier where they are captured (2*) from the input buffer, and then shifted out through the TAP (3*). This stream is analyzed off chip then, based upon the failure map the OTP

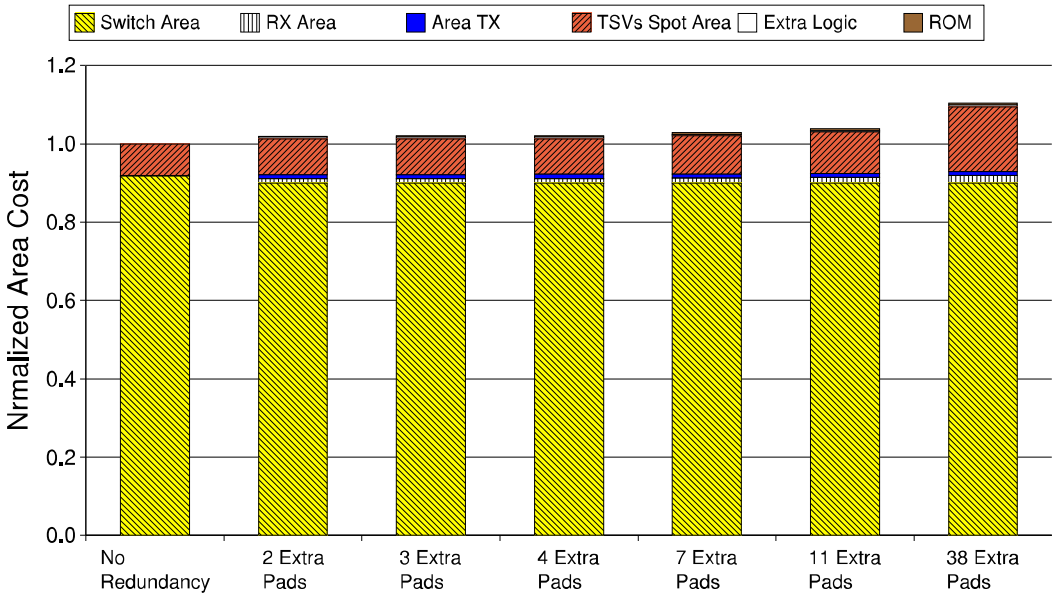


Figure 4.6: Normalized area cost in case of No Redundancy and Dynamic Routing with 2, 3, 4, 7, 11 and 38 extra pads. The main contribution of this paper is resumed starting from the 2nd bar, which shows only 1.6% area overhead for 2 extra pads, 2.1% for 4 extra pads and 10.5% for full redundancy (38 extra pads)

chip stack. Placement, routing and post layout verification have been performed. As depicted in Figure 4.8, the planar topology has been partitioned in two parts (dotted line), between the central routers. The topology under test (see Figure 4.8) includes six processors and six memories, placed on two layers. Vertical communication is achieved

through the two central switches which act as a gateway for 3DNoC traffic. The reconfigurable crossbars have been inserted between the TSV pads and the switch. For a 32-bit link, the NoC protocol uses 38 bits, where the remaining 6 bits belong to flow control signaling and mesochronous handling (i.e. the clock and reset signals which are forwarded along with the data).

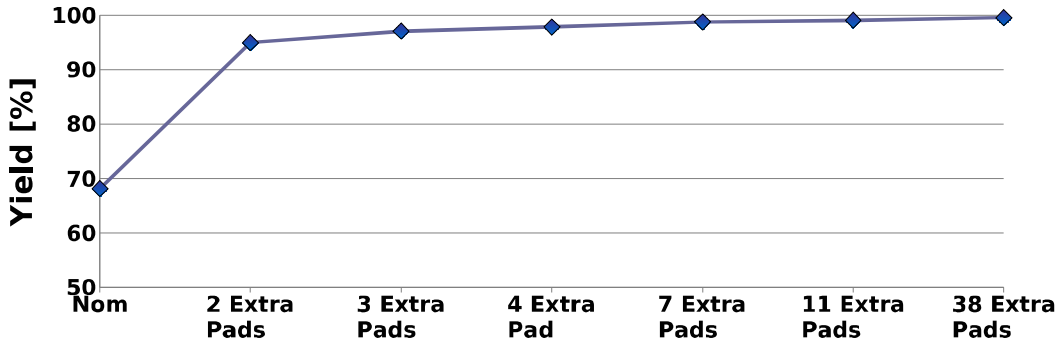


Figure 4.7: Yield improvement over seven different hardware configurations: no-redundancy, 2, 3, 4, 7, 11 and 38 extra pads, which correspond to 38, 40, 41, 42, 45, 49, and 76 TSVs per 3D link. A fixed defect frequency of 9.75 Defects Per Million Opportunities (DPMO) is assumed, and 4.2M TSVs design has been analyzed.

The nature of the reference NoC switches, namely their flow control, have influenced the adopted testing solution. During testing, a portion of the hardware works in scan mode (inject) and the other in capture mode; the flow control has to be explicitly managed to avoid the formation of communication stalls. Four scan chain groups have been inserted, driven by a simple Finite State Machine (FSM), accomplishing high efficiency and reliability. The overhead of this approach is mainly due to the crossbar logic around the via bundles, to the OTP memory and to the small FSM. The scan chain cost is not taken into account since, as mentioned before, the design must be testable anyway, and this contribution is present as well on planar ICs.

Several experiments have been conducted, especially with the dynamic routing technique, in order to evaluate how many extra pads and area may be needed for implementation, and in order to explore the trade-offs between yield and cost. We implemented six different configurations, respectively with 2, 3, 4, 7, 11 and 38 extra pads. It is worth noting that, in each unidirectional link of 38 signals, spare pads are separately needed for incoming (mostly, flow control) and outgoing (mostly, data) wires; hence

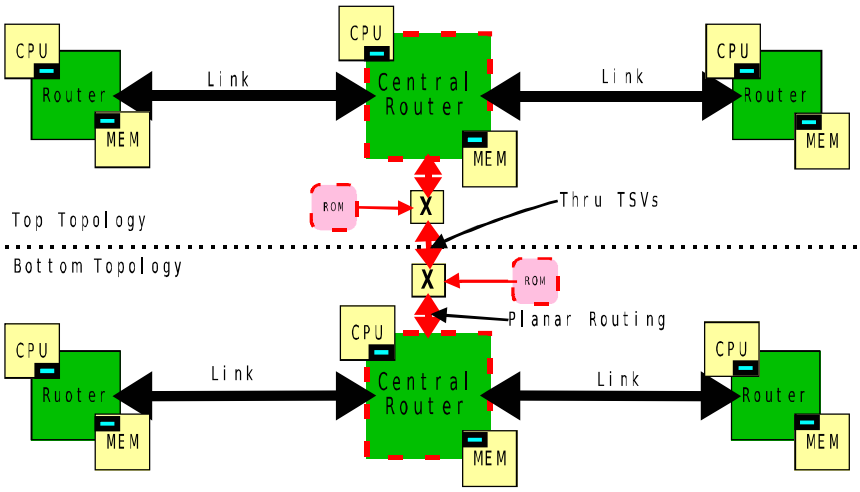


Figure 4.8: 3D NoC topology. Dash boxes indicate the resources involved in the TSV test process.

the need for at least 2 spares. The latter group typically features many more wires than the former (35 vs. 3 in our example), so the correction performance is maximized with an asymmetric assignment of spares to the two groups. For example, with only 2 extra pads, no choice is available; there is only one spare for 35 outgoing signals, while the 3 incoming wires share the second spare. With 4 spares, the optimal arrangement is to assign 3 to the outgoing bundle, and the fourth to the incoming bundle. In the extreme case of 38 spares, each TSVs has a backup.

Figure 4.7 illustrates the yield improvement in case of 2, 3, 4, 7, 11 and 38 extra pads and based on experimental data, assuming a fixed defect frequency of 9.75 Defect Per Million Opportunities (HRI TSV process (5)). We emulated 100K TSVs links with and without redundancy. Without post-manufacturing processing, the system is unable to recover damaged vias, and tolerates only small misalignments, thus exhibiting a yield of only 68%. When Dynamic Routing redundancy is adopted, the recovery algorithm shows excellent results, especially with 2 to 7 extra pads. Further increasing the number of extra pads brings minimal yield benefits, and the increase in cost of TSV obstructions, TSV crossbar and the OTP memory may be unjustified. With only four extra pads per 3D link, yield increases from 68% to 98%.

Concerning the silicon cost, Figure 4.6 shows the normalized area cost in case of different degrees of redundancy applied to a single 3D link. As the number of extra pads increases, the TSVs spot and the routing logic grow in a linear fashion. The

increasing area, with reference to a baseline composed of the Switch and the non-redundant TSVs link is 1.6% in case of 2 extra pads, 2.1% on 4 extra pads, and 10.5% in case of 38 extra pads. As a stand-alone component, the redundant links with 2 extra pads impacts for the 17% on a non-redundant Link. The physical implementation of the redundant hardware is depicted in Figure 4.9, where a pair of 3D links (42 TSVs each) is surrounded by the routing crossbar, guaranteeing low latency and better area utilization.

To evaluate the impact of the Dynamic Routing solution uses advanced technology nodes, we performed an experiment using 65nm technology library. As we can see in Table 4.2, by scaling the technology the Dynamic Routing logic scales as well. But, the TSV obstructions show the same area, since we conservatively assumed that the TSV process is independent from the technology node used for the 2D chip and it does't scale. Therefore, the area overhead of our solution increases from 2.1% on 130nm to 3.8% on 65nm, which is still very affordable.

	130nm				
	SW Area	TSVs Area	Routing HW	Total Area	Total Area increase
NoRedundancy	54000	4864	-	58864	-
With Redundancy	53000	5376	1713	60090	+1225 (2.1%)
	65nm				
	SW Area	TSVs Area	Routing HW	Total Area	Total Area increase
NoRedundancy	13500	4864	-	18364	-
With Redundancy	13250	5376	430	19056	+692 (3.8%)

Table 4.2: Silicon Cost [μm^2] of TSV redundancy solution with 4 extra pads in 2 technology nodes

4.6 Conclusions

3DICs, especially those based on Through-Silicon Vias, are gaining traction as a workaround against the increasing costs of chip miniaturization. However, the manufacturing technology is not mature enough, resulting in issues such as misalignments and random defects. Misalignment-reduction techniques have undergone significant improvements, so that today random defects must be considered the main source of yield losses. For this reason, minimizing their impact is crucial. In this chapter, we study some baseline redundancy schemes and we notably propose a novel Dynamic Routing approach. The latter scheme is based on post-manufacturing study and reconfiguration of the electrical resources, leveraging a small amount of on-chip spares. The scheme proves capable of

yields up to 98% with a minimum silicon cost of just 17% per TSV link in 130nm. This cost is further projected to decrease to just 12% in the newest 65nm technologies.

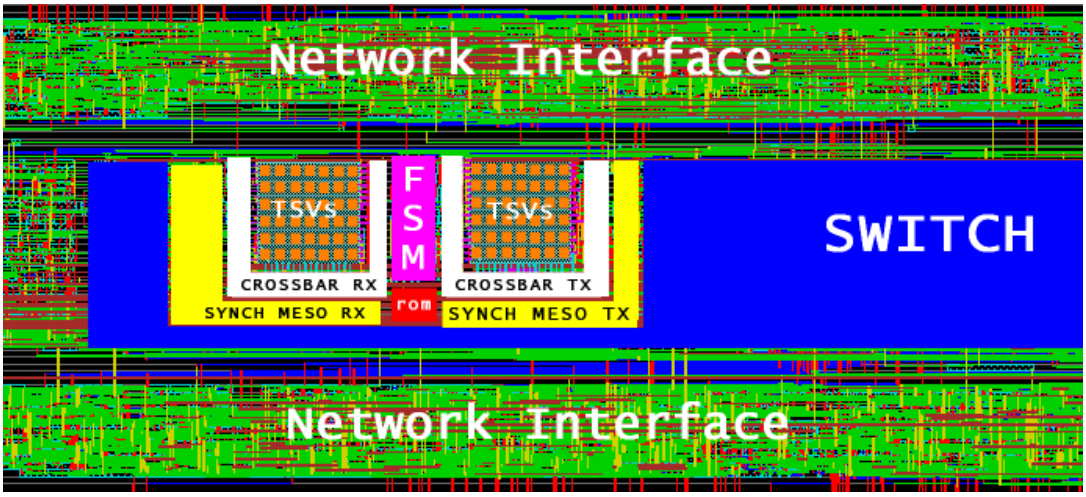


Figure 4.9: Layout detail of the bottom tier (3DICs) with emphasis on TSVs guide and configurable crossbar

Bibliography

[1] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proceedings of the 38th Design Automation Conference*, June 2001, pp. 684–689. 47, 50

[2] L. Benini and G. De Micheli, "Networks on chip: a new SoC paradigm," *IEEE Computer*, vol. 35, no. 1, pp. 70–78, January 2002. 47, 50

[3] R. S. Patti, "Three-dimensional integrated circuits and the future of system-on-chip designs," *Proceedings of the IEEE*, vol. 94, no. 6, June 2006. 47, 50, 51

[4] A. W. Topol, J. D. C. La Tulipe, L. Shi, D. J. Frank, K. Bernstein, S. E. Steen, A. Kumar, G. U. Singco, A. M. Young, K. W. Guarini, and M. Ieong, "Three-dimensional integrated circuits," *IBM Journal of Research and Development*, vol. 50, no. 4/5, pp. 491–506, July/September 2006. 48, 50, 51

[5] N. Miyakawa, T. Maebashi, N. Nakamura, S. Nakayama, E. Hashimoto, and S. Toyoda, *New Multi-Layer Stacking Technology and Trial Manufacture*, November 2007, honda Research Institute Japan Co. Ltd. 48, 50, 57

[6] B. Swinnen, W. Ruythooren, P. D. M. L. Bogaerts, L. Carbonell, K. D. Munck, B. Eyckens, S. Stoukatch, Tezcan, D. Sabuncuoglu, Z. Tokei, J. Vaes, J. V. Aelst, and E. Beyne, "3d integration by cu-cu thermo-compression bonding of extremely thinned bulk-si die containing 10 um pitch through-si vias," pp. 1–4, 2006. 48

[7] A. W. T. et all, "Enabling soi based assembly technology for three dimensional integrated circuits," pp. 352–355, IEDM 2005. 48

[8] J. Kim, C. Nicopoulos, D. Park, R. Das, Y. Xie, N. Vijaykrishnan, M. S. Yousif, and C. R. Das, "A novel dimensionally-decomposed router for on-chip communication in 3d architectures," in *Proceedings of the 34th International Symposium on Computer Architecture (ISCA)*, 2007. 50

[9] S. Fujita, K. Nomura, K. Abe, and T. Lee, "3d on-chip networking technology based on post-silicon devices for future networks-on-chip," in *Nano-Networks and Workshops*, September 2006, pp. 1–5. 50

[10] S. Fujita, K. Nomura, K. Abe, and T. H. Lee, "3-d nanoarchitectures with carbon nanotube mechanical switches for future on-chip network beyond cmos architecture," *IEEE Transactions on Circuits and Systems Part I: Regular Papers*, vol. 54, no. 11, pp. 2472–2479, November 2007. 50

[11] M. Rencher and F. Schellenberg, "Why interconnect and litography modeling impacts yield," in *What's Yield got to do with IC*. 50

[12] I. Loi, F. Angiolini, and L. Benini, "Supporting vertical links for 3d networks-on-chip: Toward an automated design and analysis flow," in *Proceedings of the Nano-Net Conference 2007*, 2007, pp. 23–27. 50, 51, 52

[13] I.Loi, F.Angiolini, and L.Benini, "Developing mesochronous synchronizer to enable 3d nocs," in *Proceedings of the Date Conference 2008*, 2008, pp. 1414–1419. 50, 52

[14] S.Spiesshoefer and et al, "Z-axis interconnects using fine pitch, nanoscale through-silicon vias: Process development," in *Electronic Components and Technology Conference*, 2004. 50

[15] R. Patti, "Impact of wafer-level 3d stacking on the yield of ics," in *Future Fab Intl*, September 2007. [Online]. Available: http://www.future-fab.com/documents.asp?d_id=4415 51

[16] N.Miura, D.Mizoguchi, M.Inoue, T.Sakurai, and T.Kuroda, "A 195-gb/s 1.2-w inductive inter-chip wireless superconnect with transmit power control scheme for 3-d-stacked system in a package," *IEEE journal of solid state circuits*, vol. 41, no. 1, pp. 23–34, january 2006. 51

[17] K.-N. Chen, A. Fan, and R. Reif, "Microstructure examination of copper wafer bonding," in *http://www-mtl.mit.edu/ reif/papers/2001-knchen-JEM-manuscript.pdf.* 51, 53

[18] K.-N. Chen, C. Tan, A. Fan, and R. Reif, "Morphology and bond strength of copper wafer bonding," in *Electrochem. Solid-State Lett.*, vol. 7, 2004, pp. 14–16. 51

[19] A. Papanikolaou, M. Miranda, H. Wang, F. Catthoor, M. Satyakiran, , P.Marchal, B. Kaczer, C. Bruynseraede, and Z. Tokei, "Reliability issues in deep deep submicron technologies: time-dependent variability and its impact on embedded system design," pp. 342–347, IEDM 2006. 51

[20] K. N. Chen, A. Fan, and R. Reif, "Interfacial morphologies and possible mechanisms of copper wafer bonding," in *http://www-mtl.mit.edu/users/reif/papers/2002-knchen-JMS-manuscript.pdf.* 53

[21] F. Angiolini, P. Meloni, S. Carta, L. Raffo, and L. Benini, "A layout-aware analysis of networks-on-chip and traditional interconnects for mpsocs," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 26, no. 3, pp. 421–434, March 2007. 52

Future work may revolve around timing faults, which are an often underestimated source of failures.

5

Reconfigurable Source Routing Tables for unreliable NoCs

In on-chip multiprocessor communication, link failures and dynamically changing application scenarios represent demanding constraints for the provision of suitable Quality of Service. Networks-on-Chip (NoCs) featuring dynamic routing are a known way to tackle these issues, but deadlock freedom and message ordering concerns arise. NoCs with configurable routing, whereby the communication routes are explicitly chosen at runtime out of a set of statically predefined alternatives, provide intelligent adaptation without impacting the consistency of traffic flows.

However, configurable source routing on a NoC platform requires a design that provides fast path lookup coupled with low area and power consumption. This chapter presents an exploration and synthesis approach that, depending on the required amount of routing flexibility, can for example reduce by 3 to 15 times the area cost of the NoC routing tables by adopting partially reprogrammable routing logic instead of fully reprogrammable tables. Further optimizations based on path redundancy allow to reduce up to 17 times the silicon cost.

5.1 Introduction

Global on-chip communication is becoming a problem as silicon chips become larger, technology scales down, and the clock frequency increases. Signals are predicted to take several clock cycles to travel over the longest distances from corner to corner of a

chip (1). Simultaneously, the increasing performance requirements of highly parallel on-chip architectures are unmet due to the bottlenecks imposed by traditional, bus-based on-chip interconnects.

The Network-on-Chip (NoC) (2, 3) paradigm, which brings packet-switching networking concepts to the on-die level, has been proposed to systematically tackle these challenges. NoCs are a structured, predictable and scalable approach to the problem, centered around wire segmentation and point-to-point signaling.

Generally, cores attached to a NoC do not have any information about the NoC topology - only the destination addresses are known. One of the key functional features of NoCs is providing routing services among these endpoints. In a basic implementation, NoC routing can be extremely simple. For example, several proposed approaches just tag every packet with a routing field in the header. The tagging is done at the network endpoints (Network Interfaces or Network Adapters) by leveraging routing Look-Up Tables (LUTs). The route is deterministic, decided at design time. Yet, this minimal approach is not able to tolerate changes in the operating conditions at runtime, such as:

Intervening faults (*e.g.* switch or link failures)

Application switching (*e.g.* task migration or task switching, which may induce critical localized congestion)

Power management events (*e.g.* power down of a portion of the NoC)

Dynamic routing has often been suggested as the answer to best handle these scenarios. Dynamic routing involves forwarding packets along different paths depending on a choice of decision variables, which are evaluated cycle-by-cycle at runtime. For example, with dynamic routing, packets would automatically find an alternate way around a faulty NoC node. Unfortunately, dynamic routing introduces two major problems: deadlocks and packet ordering. Since every packet may follow a different route, it becomes hard to avoid routing loops, which induce deadlocks. Sequential packets travelling among the same endpoints on different routes may also encounter different congestion, and reach their common destination in swapped order - a condition which is forbidden in several implementations, requiring reordering queues. Therefore, dynamic routing can become impractical in practice.

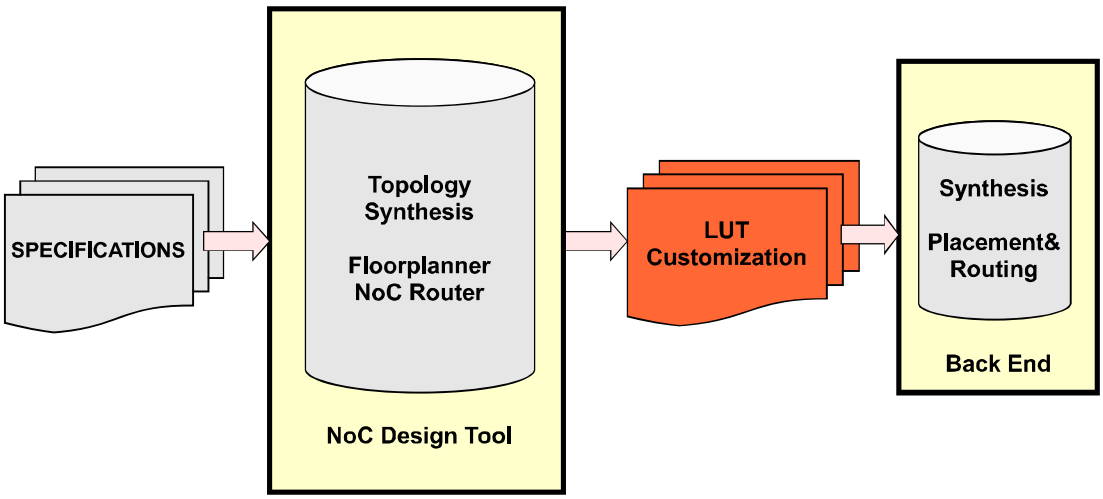


Figure 5.1: Reference NoC design flow. The routing customization proposed in this chapter operates on given NoC topologies to improve the results upon physical synthesis.

In this chapter, we follow an approach in between deterministic and dynamic routing, which we will call *configurable routing* in the remainder of the chapter. We acknowledge the importance of adding reconfiguration capabilities into the NoC routing policies, but we adopt an architectural design flow which achieves them without incurring any of the major downsides of dynamic routing. Referring to Figure 5.1 (left side), we operate by leveraging one of the several proposed approaches for NoC topology and route design (4, 5, 6, 7, 8). These works specifically focus on the generation of NoCs whereby, depending on application needs, routes are established to provide connectivity among communicating nodes. We assume that multiple sets of routing schemes are available as an input. We then support runtime reconfiguration of the NoC routes among one of the possible alternatives, with the idea of performing a reconfiguration only upon one of the rare events which really demand it (*e.g.* a power down message), and not cycle-by-cycle. The rationale is that, by pre-characterizing a finite set of possible routes, these can be verified to be deadlock-free. Further, packets are again delivered in-order by construction.

A remaining issue with configurable routing is that programmable NoC routing tables can have a prohibitive hardware cost. The main novel goal of this chapter is to mitigate this issue. In order to do so, we first observe that configurable routing crucially requires a certain amount of design-time activities - for example, coming up with the

static routing sets. Therefore, in this chapter, we present a design-time analysis step whose purpose is to identify, out of the various alternatives, the cheapest architecture to support variable amounts of routing reconfigurability, and to further optimize it. As can be seen again in Figure 5.1 (center), the proposed novelty plugs into existing NoC design flows: given a NoC optimized for a certain application, and namely given a set of possible routes over a topology, our objective is to assess and optimize the cost for rendering those routes into configurable routing LUTs.

In the following, we will present a detailed study of various possible mechanisms to support the routing LUT reconfigurability, keeping in mind the area cost metric. Such mechanisms include RAM- and register-based LUTs, or cheaper solutions whereby the routing LUTs can be only partially reconfigured. Completely static routing is kept as a benchmark. We will also present a comparison of three different synthesis flows of these architectures, with the goal of reaching the most effective physical implementation. Our experiments show that it is critical to pick the right architecture for a given set of requirements. For example, across a wide range of numbers of nodes in the NoC, partially configurable LUTs have shown an excellent trade off between route reconfigurability, area-power cost, and performance. Area savings from 3X to 15X can be achieved compared to fully reconfigurable circuitry, depending on the routing flexibility requirements. A further proposed optimization allows to drastically reduce the programmable elements, additionally saving up to 20% of the area cost in our test case.

5.2 Related Work

Networks-on-Chips (NoCs) have been proposed by numerous authors (2, 3, 9, 10) as a way to tackle multiple on-chip interconnection challenges of multicore devices, such as scalability to ever larger numbers of IP cores, increasing bandwidth demands, and worsening propagation delays of global on-chip wires.

As NoCs are becoming the focal point of the system integration process, several authors have investigated ways to embed advanced features into them. For example, work has been done on NoC-centered mechanisms for fault tolerance, power management and performance optimization. One of the crucial degrees of freedom in NoC design, which has been leveraged to solve some of the above problems, is routing. Routing is normally categorized as either deterministic (packets follow statically know routes) or adaptive,

also called dynamic (packets follow different routes over time depending variables such as congestion states or fault conditions). Deterministic routing is often implemented with lookup tables at the NoC endpoints (11), although in some regular topologies, such as spidergon or mesh, it can be performed in switches based on a destination tag (12, 13). The work in (14) proposes a mixed approach based on irregular meshes. Dynamic routing needs decentralized decision processes and is therefore often achieved with dedicated logic in every router (15). Deterministic routing is characterized by its simplicity and minimal overhead; it can easily be configured to avoid deadlocks (16) and natively guarantees in-order delivery. Unfortunately, deterministic routing does not adjust to the system evolution over time; on the contrary, dynamic routing has been proposed to achieve goals such as bypassing faulty nodes and minimizing congestion (15, 17).
Unfortunately, dynamic routing generally induces deadlock conditions, which must be resolved, and, in many implementations, can deliver packets out-of-order, mandating the presence of reordering buffers. These provisions can become impractically expensive on silicon. In (18), a mechanism that compresses lookup tables for adaptive routing has been presented. This solution is however not suitable for irregular topologies. The work in (19) proposes a region-based routing mechanism (adaptive routing) to tackle unreliable hardware in network on chips.

In this chapter, in order to support configurable routing instead, first of all we leverage several previous efforts aimed at NoC topology synthesis for a given application (4, 5, 6, 7, 8). Configurable routing allows bypassing faulty nodes or links, safely shutting down chip regions, and readjusting traffic patterns upon a change in the software application running on the chip. Configurable routing has been proposed in several forms; for example, in (20), the authors propose a custom methodology, based on packet rerouting, to handle data transfers upon power management events or system faults. However, their approach is not general and is actually working around faults in the attached cores, not in the NoC itself. More in general, a novel contribution of this chapter is an exploration of the design space for configurable routing implementation.

In this chapter, we will explore the area impact of synthesizing routing tables in multiple ways. To this extent, we will use the built-in logic optimizer of industry-standard Synopsys tools (21), as well as BOOM II (22) and ABC (23). BOOM is a

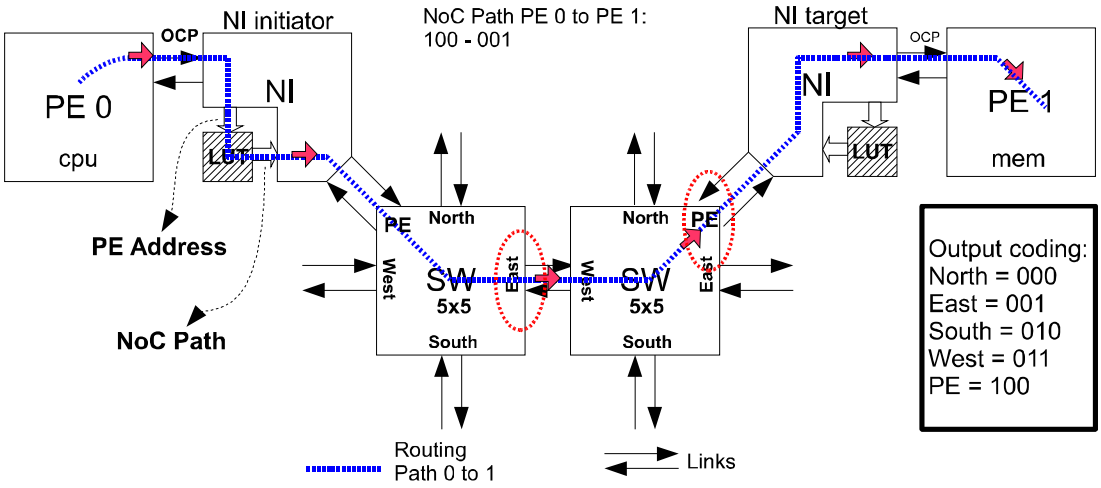


Figure 5.2: Example routing mechanism for deterministic source routing NoCs. For each transaction, the Processing Element provides a destination address. This address is then translated into a NoC path, which is merely the ordered sequence of bits representing the codes of the switch ports that packets need to take to reach their destination. The NI uses a LUT to store the route map.

tool specialized in minimizing multiple-output combinational logic, and generates two-level logic as an output. ABC is more flexible and can handle generic circuits, including sequential and multi-level logic.

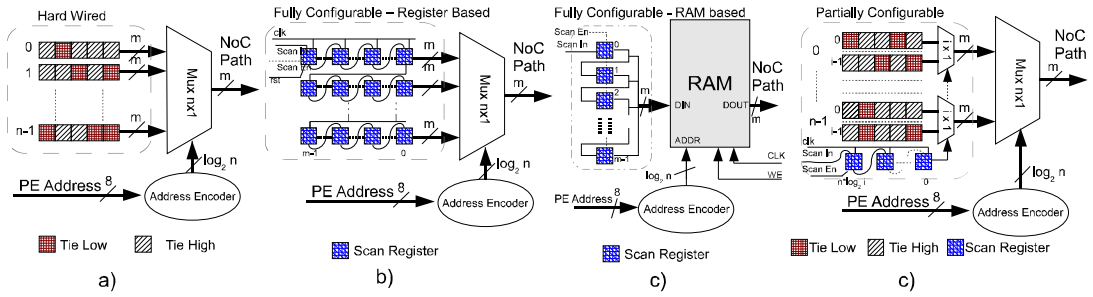


Figure 5.3: Different logic implementation of the Source Routing LUT: a) Hard Wired: each route is encoded as hard wired sequence of 0s and 1s; b) Fully Configurable RAM based: each route is stored in a memory cells and can be fully reprogrammed at any time with a scan chain; c) Fully Configurable register based: each route is stored in a register and can be fully reprogrammed at any time with a scan chain; d) Partially Configurable: alternate fixed routes are available for each destination and can be selected by programming a few control bits via a scan chain.

5.3 Configurable Source Routing NoCs

Network Interfaces (NI) seamlessly connect existing IP modules to a Network-on-Chip. They play a crucial role in a NoC context, determining the performance of the whole system. NIs, given a request from the attached processing element, generate packets that will be sent to the destination core, and all the information needed to manage the flow control. As commonly seen, we assume as a reference a NoC with deterministic source routing. For each message coming from the attached core, the NI generates in a deterministic manner the routing bits needed to traverse the NoC switches. An LUT is used for this purpose, converting the memory-mapped address into routing bit sequences. Figure 5.2 shows the routing mechanism: the core request is processed, and depending on the destination address, the NI generates the path across the NoC switches, up to the destination element. The routing bits are stored in the packet header. When the packet is sent through the NoC switches, a physical channel is created between the packet source and destination.

Depending on how reconfigurable they are, we classify routing LUTs in three categories:

Hard Wired

Fully Configurable

Partially Configurable

As LUT configurability increases, the system becomes more flexible. A highly configurable LUT could for example be reprogrammed to route packets around a large number of NoC faults, while a less configurable LUT may not be able to work around more than one fault, and a hard wired LUT may not tolerate any single faulty link in the NoC. Similarly, more configurability could lead to better performance in a system where many different tasks may be switched over time, *etc.*. Unfortunately, in general, more configurable LUTs are also more expensive in area. This section gives an overview of various possible architectural implementations of NI LUTs. The designer is ultimately in charge of picking one alternative, but our approach makes clear the trade-off among the area cost and the achievable flexibility for the given NoC topology. In the following we explain in detail the possible architectural implementations of the LUTs.

5.3.1 Hard Wired

This first solution is presented as a reference. Logically, it is simply a ROM; circuit synthesis tools will actually render it with a netlist of combinational gates. The routing information is permanently stored and no changes are possible at runtime. The address decoder translates the addresses issued by the attached core to properly drive the multiplexer selector. Figure 5.3a illustrates in detail the logic implementation.

5.3.2 Fully Configurable

In this architecture, the routing information is stored on either registers or memory banks. Unlimited remapping of NoC routes is allowed at run time, with maximum routing flexibility. Very different reconfiguration circuits are needed depending on whether the memory elements are rendered as plain flip-flops or whether the designer instantiates a RAM macro instead. The register programming can take a place by injecting a setup vector through a scan chain, depicted in Figure 5.3b. On the contrary, the RAM-based LUT uses a dedicated data structure, and as depicted in Figure 5.3c, a serial-to parallel converter is needed to properly initialize the memory. The number of memory elements is in both cases $n * m$, n being the number of destinations (LUT entries) and m being the number of bits required to encode the longest path. Both solutions use a large amount of scan registers - namely $n * m$ for the former and m for the latter - for the programming operations. This also means that several clock cycles are needed to change the route map, during which the NI is forced to stay idle. This high programming latency is, however, consistent with the fact that the LUT reprogramming is expected to happen only upon rare events, such as upon failures or power downs.

5.3.3 Partially Configurable

Partially configurable LUTs represent a hybrid solution between the previous schemes. Figure 5.3d gives an overview of this solution. Up to i NoC paths are allowed for each possible destination. These are hard coded, and through a setting logic block, the desired path is enabled. No memory element is required to store the LUT content itself; however, some flip-flops are still needed as the choice of which route to enable is again performed via a scan chain. Since the scan chain injects only the ID of the

desired configuration, and not the entire NoC map connectivity, sequential resources decrease drastically and the total amount of clock cycles needed to reprogram the table is much lower compared to the fully configurable scheme.

5.4 Synthesis of Configurable Source Routing Logic

We chose to base our integration effort on the ×pipes (8), which supports arbitrary connectivity, and on its instantiation toolchain (24). Thus, we can leverage a full design flow up to the layout level, as depicted in Figure 5.1.

For the sake of simplicity, and without losing generality, we choose a simple and regular NoC topology, a mesh. We assume a full connectivity among the cores: each core communicates with all others. An internal tool outputs LUT description, in Verilog, for each NI. As per the default, the tool creates a code which is interpreted as a hard wired structure, with each reachable endpoint node hard coded as one entry of the LUT.

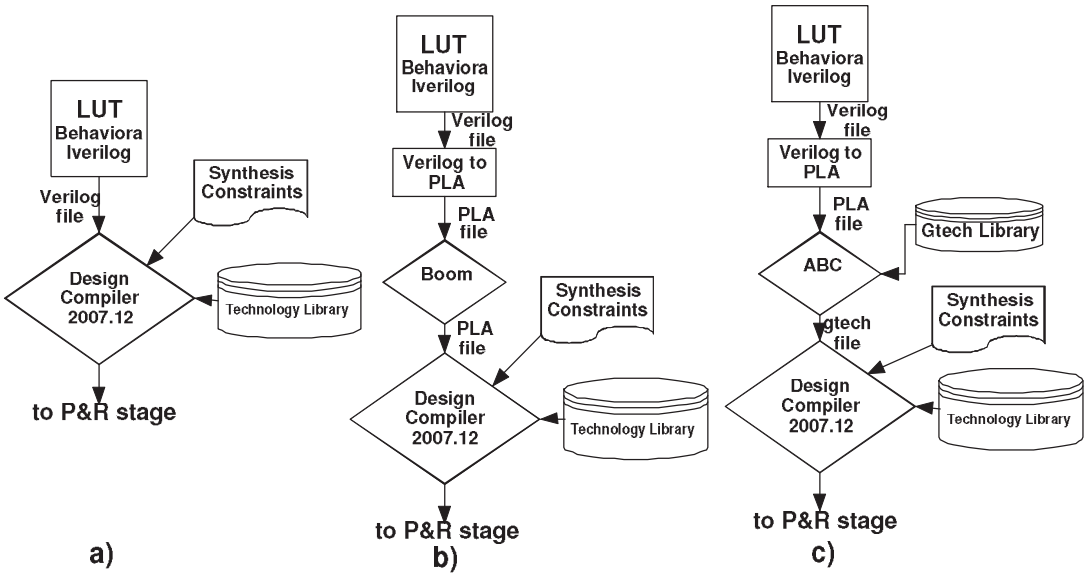


Figure 5.4: Three different synthesis flow have been adopted; a) standard synthesis flow; b) Boolean Optimization before the standard synthesis flow; c) And-Inverter Graphs (AIG) Optimization and GTECH mapping before the standard synthesis flow

In order to evaluate the impact of synthesis tools on the cost of routing LUTs, three different synthesis flows have been explored as a contribution of this work:

Standard synthesis flow with Synopsys tools (21)

Boolean Optimization with BOOM II (22) before the standard synthesis flow

And-Inverter Graphs (AIG) Optimization with ABC (23) and GTECH mapping before the standard synthesis flow

In the standard Synopsys synthesis flow, the LUTs come as a Verilog behavioral description, where a parallel switch statement implements the LUT logic. Since the NoC paths may show different depth, *don't care* usage may improve the quality of synthesis results. During the synthesis stage, Design Compiler performs a preliminary optimization check by using a PRESTO boolean minimizer.

As the number of paths grows, the LUT increases in complexity, suggesting that a preliminary optimization before the logical synthesis may be useful. This motivates our effort to explore different optimization alternatives, like BOOM II (22) and ABC (23).

BOOM II is a heuristic two-level multiple-output boolean minimizer. As it is compatible with the Berkeley standard PLA format, we use a Perl script to translate the behavioral Verilog description into a PLA file, getting an optimized PLA file as the output of BOOM II. This file is then used for the subsequent synthesis step.

We also try ABC (23), a powerful tool for the synthesis of combinational or sequential logic circuits. With a modification to its generic technology library, ABC can be asked to synthesize the same PLA files that we generate for BOOM II into a netlist of GTECH cells. GTECH is the generic netlist format used by Design Compiler to describe circuits just before mapping them onto a specific technology library, and can therefore be passed as an input to Design Compiler for the final synthesis.

5.5 Experimental Results

The proposed solutions have been synthesized and mapped on the UMCE-Faraday 130nm CMOS Technology library, then the Place and Route and the verification have been performed to check the correctness and evaluate the area cost, timing performance and power consumption.

With respect to partially configurable LUTs, we identify two scenarios of interest. The first is the minimum cost one, when only one redundant path per possible NoC destination is provided. The maximum number of redundant entries, on the other hand, is potentially almost unbounded, depending on the topology and on its size. However,

due to deadlock freedom constraints, many fewer routing combinations will be valid. If the fault tolerance is factored in, alternate routes should use as disjoint sets of links as possible, further reducing the solution space. It is not the focus of this chapter to calculate how many possible route sets may exist in any given topology; for this specific experiment, based on our experience with enumeration in small meshes, we consider at most six such alternate routes. We present experimental results sweeping among these bounds - two to six alternate routes per LUT entry.

Our first experiment is summarized in Figure 5.5, where we compare the silicon cost, expressed in equivalent gates, of three possible LUT implementations. We assume 14 bits of routing information per LUT entry. The hard wired LUT shows a very efficient area utilization, respectively 18 and 30 times smaller compared to the fully configurable register- and RAM-based LUTs for 6 to 8 LUT entries (the lower bound). This can be explained by the fact that in completely static LUTs the synthesis tool can tap into a huge optimization potential, as the address decoder and the multiplexer can be merged as glue logic. As the number of LUT entries increases, the silicon cost increase in a linear fashion, mainly due to the multiplexer. Concerning the RAM-based solution, four memories have been generated with the UMCE-FARADAY 130nm memory compiler. This tool allows the designer to specify some parameters (word length, number of columns, number of words, *etc.*). We selected four values for the number of words in the LUT - 8, 16, 32 and 64. As expected, RAMs of few words experience a high hardware overhead due to memory handling logic, but this solution is quite efficient when the number of entries grows, since in fact its cost slope is comparable to the hard wired solution. Finally, the register-based fully configurable LUT is very efficient for small LUTs, but its cost slope is steep. Beyond about 10 LUT entries, this solution shows poor area efficiency as compared to the RAM-based solution.

In many designs, the LUT may fall in the critical path of the Network Interface block, therefore the timing performance of the whole NoC may degrade if the LUT is too slow. Figure 5.5b summarizes the timing cost: hard wired and register-based solution show the same timing properties, since the latency is dominated by the multiplexer, while in case of a RAM-based LUT the latency is dominated by the memory access time. This cost however grows slowly as compared to the other solutions, confirming the intuitive finding that the RAM usage becomes suitable for very complex designs with large LUTs.

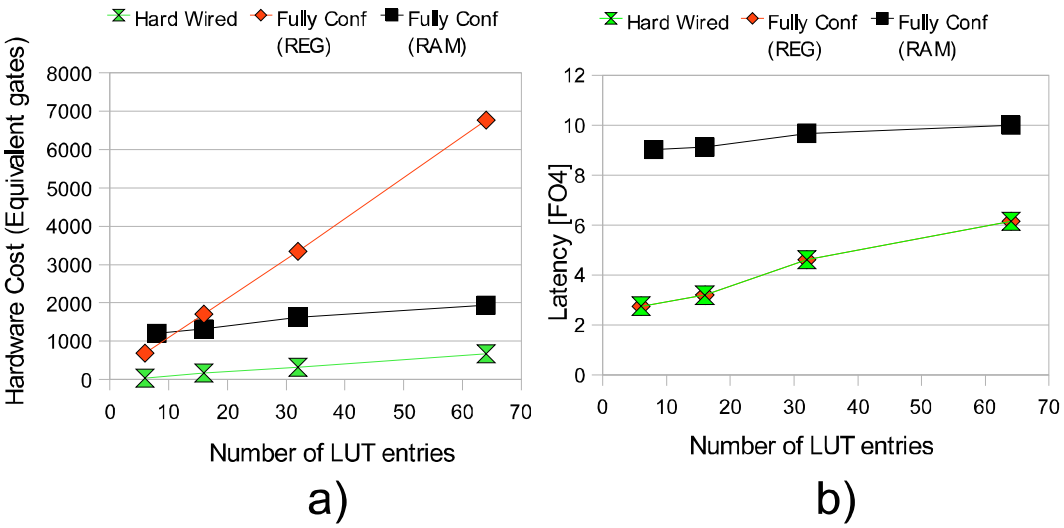


Figure 5.5: Cost of hard wired and fully configurable LUTs for a 3x3 mesh: a) area cost (equivalent NAND2 gates); b) propagation delay between input and output ports (FO4 delays)

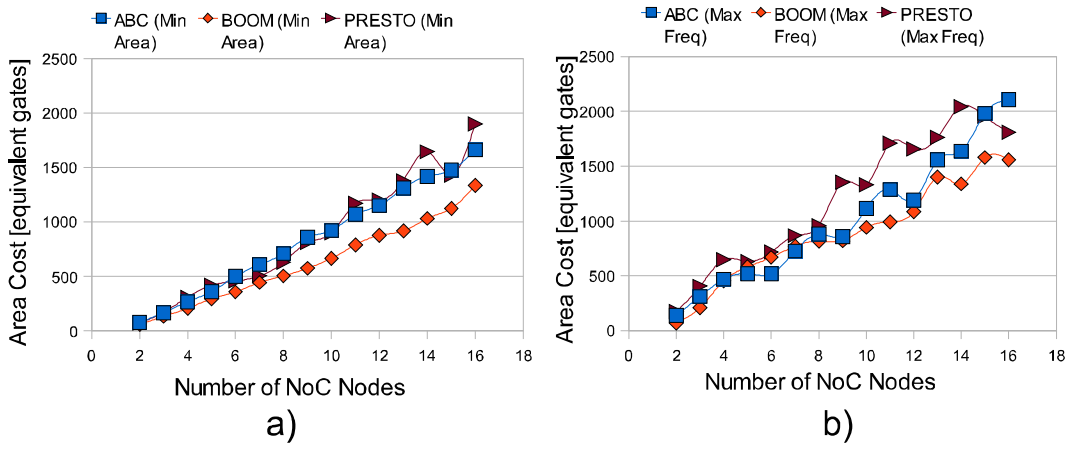


Figure 5.6: Area Cost for three synthesis flows: standard flow (PRESTO), boolean minimization (BOOM) and AIG optimization (ABC). a) The area is optimized with fixed timing constraints; b) the area is optimized for maximum operating frequency

As we introduced in the previous section, we explored three different synthesis flows for partially configurable LUTs. A detailed comparison of the outcome is presented in Figure 5.6. BOOM II demonstrates clear area savings. Figure 5.6a illustrates the silicon cost for a fixed latency between input and outputs, while Figure 5.6b shows the

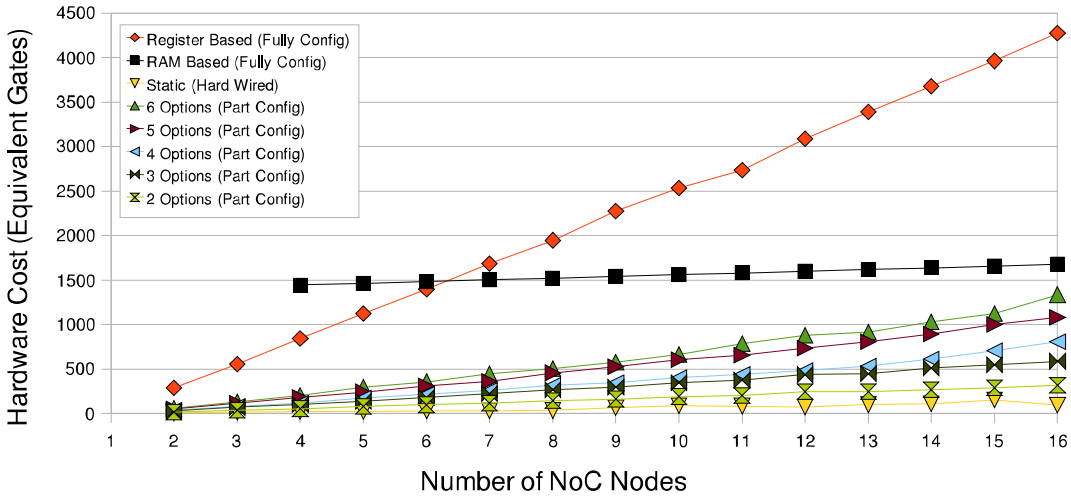


Figure 5.7: Area cost of partially configurable LUTs for a 4x4 mesh when sweeping both the number of entries and the number of possible alternatives per entry

area cost with a maximum operating frequency goal. Thanks to boolean minimization the area reduction ranges from 7% up to 20% with respect to the standard flow. ABC optimization on the other hand does not bring any significant benefits and shows results that are comparable whith a standard Synopsys flow.

Figure 5.7 illustrates the silicon cost to implement the partially configurable approach. This experiment is based on a 4x4 mesh topology, with a routing depth of 21 bits and boolean optimization before the synthesis. We sweep the number of nodes and redundant paths to investigate the trade offs in hardware complexity. Figure 5.7 also presents an exhaustive comparison between partially configurable, fully configurable and hard wired routing. Partially configurable solutions exhibit clearly better results across all the design points we consider here. LUTs with 6 and 2 alternate entries are respectively 3 and 15 times smaller than the equivalent register-based fully configurable LUT, and up to 40% of their area is required by the setting registers. The total area increase compared to a hard wired routing LUT is just approximately 20% for 6 options, and 7% in case of 2 options. Factoring in the area cost of the rest of the NI, the gain is still very clear. An NI equipped with partially configurable LUTs needs approximaltely 79% less area in case of 6 option, and 103% in case of 2 options, compared to one with fully configurable register-based LUTs. This proves the importance of the study in this

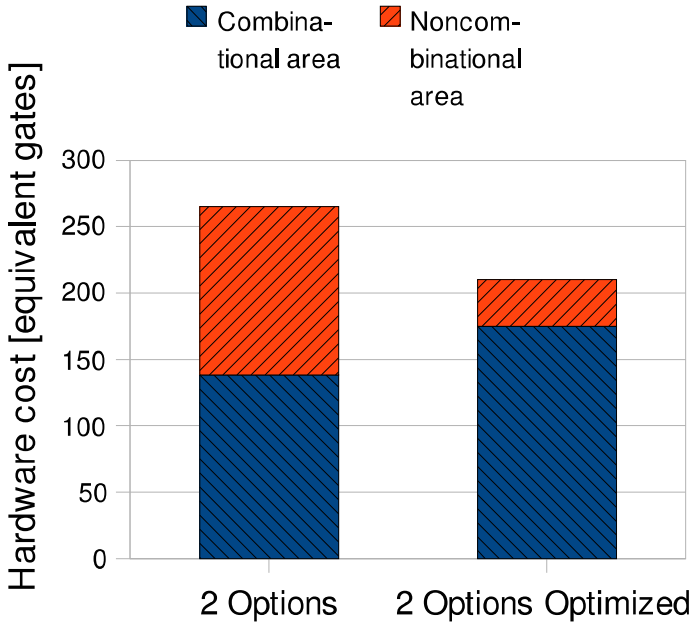


Figure 5.8: 2-option and optimized partially configurable LUT. Sequential area is reduced by 3.6 times, with a total cost decrease of 20%

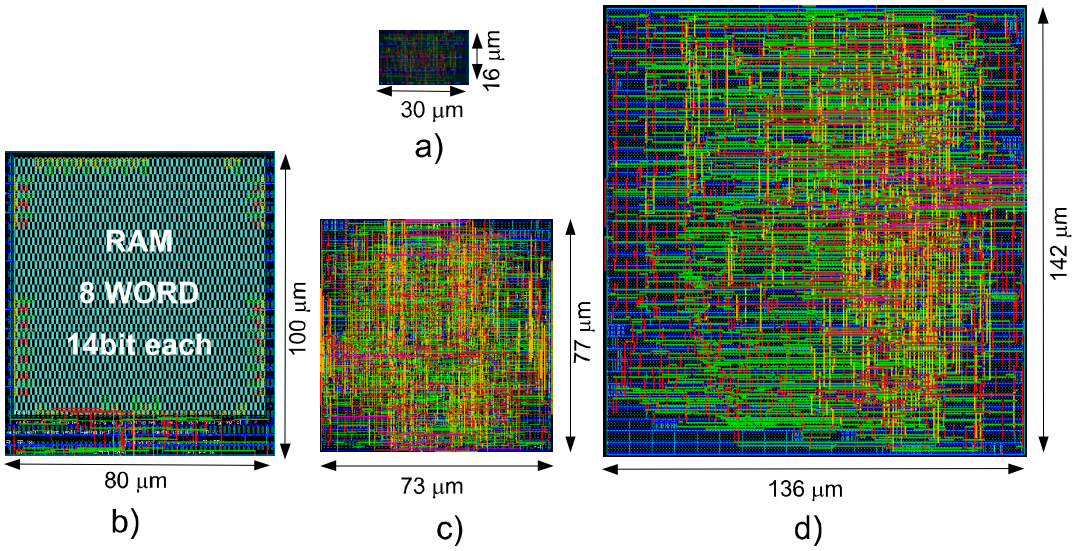


Figure 5.9: Physical implementation in 130nm of the a) Hard wired LUT, b) Fully configurable RAM-based LUT, c) Partially configurable (with 6 options per path) LUT, and d) Fully configurable register-based LUT

chapter and that a considerable amount of routing flexibility can be supplied at a low cost.

As a last experiment, we try to minimize the configuration register cost for partially configurable LUTs. n configuration bits (assuming just one per LUT entry) allow for 2^n possible route combinations, a number which is likely to be much larger than needed. A partially configurable LUT of 16 entries with two options each includes 15 registers used to store the setup information, therefore 2^{15} different configurations are possible. Since many of these configurations are certainly invalid, for example due to deadlocks, we now decrease the number of legal configurations, removing 10 of the 15 setting flip-flops. The 15 configuration bits will therefore be generated with 5 flip-flops and combinational gates.

Figure 5.7 illustrates the area savings by adopting this optimization. The sequential area becomes 3.6 times smaller, and the total cost is reduced by 20%.

A visual comparison of the considered alternative architectures is given in Figure 5.9, which presents layout screenshots.

5.6 Conclusions

In this chapter we have presented an approach which allows NoC designers to deploy NoCs based on configurable routing. Our contributions include a way to evaluate configurability/cost/latency trade-offs among different alternate architectures, plus several investigations on how to optimize the quality of the physical-level results.

In our tests, hard wired LUTs prove the least expensive. But if routing flexibility is needed, reconfigurable tables are mandatory. Among register-based, RAM-based, and partially configurable LUTs, the first shows better performance for small tables (*e.g.* up to 10 words of 14 bits each). Beyond that size, RAM-based solutions are better in area. The area overhead however is still high. Partially configurable LUTs offer a good trade off between routing flexibility, area cost and performance. The area gain ranges from 3x to 15x (1,2x to 5,2x as regard to RAM based scheme) depending on the routing flexibility requirements. Further optimizations allows to drastically reduce the programmable elements, saving up to 20% extra area cost in our test case.

Bibliography

[1] D. Sylvester and K. Keutzer, "A global wiring paradigm for deep submicron design," *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, vol. 19, pp. 242–252, Feb 2000. 66

[2] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proceedings of the 38th Design Automation Conference*, June 2001, pp. 684–689. 66, 68

[3] L. Benini and G. De Micheli, "Networks on chip: a new SoC paradigm," *IEEE Computer*, vol. 35, no. 1, pp. 70–78, January 2002. 66, 68

[4] A. Pinto, L. Carloni, and A. Sangiovanni-Vincentelly, "Efficient synthesis of networks on chip," in *Proceedings of 21st International Conference in Computer Design*, October 2003, pp. 146 – 150. 67, 69

[5] K. Srinivasan, K. S. Chatha, and G. Konjevod, "An automated technique for topology and route generation of application specific on-chip interconnection networks," in *Proceedings of the 2005 IEEE/ACM International conference on Computer-aided design (ICCAD '05)*. Washington, DC, USA: IEEE Computer Society, 2005, pp. 231–237. 67, 69

[6] W. H. Ho and T. M. Pinkston, "A Methodology for Designing Efficient On-Chip Interconnects on Well-Behaved Communication Patterns," in *The Ninth International Symposium on High-Performance Computer Architecture (HPCA'03)*, Feb. 2003, p. 377. 67, 69

[7] T. Ahonen, D. A. Siguenza-Tortosa, H. Bin, and J. Nurmi, "Topology optimization for application-specific networks-on-chip," in *Proceedings of the 2004 international*

workshop on System level interconnect prediction (SLIP '04). New York, NY, USA: ACM Press, 2004, pp. 53–60. 67, 69

[8] S. Murali, P. Meloni, F. Angiolini, D. Atienza, S. Carta, L. Benini, , and G. D. Micheli, "Designing application-specific networks on chips with floorplan information," in *Proceedings of the 2006 International Conference on Computer-Aided Design (ICCAD)*, 2006, pp. 355–362. 67, 69, 73

[9] P. Guerrier and A. Greiner, "A generic architecture for on-chip packet-switched interconnections," in *Design Automation and Test in Europe, DATE'00*, March 2000, pp. 250 – 256. 68

[10] E. Salminen, A. Kulmala, and T. D. Hmlinen, "Survey of network-on-chip proposals," Tech. Rep., March 2008. [Online]. Available: www.ocpip.org/socket/whitepapers 68

[11] D. Bertozzi and L. Benini, "×pipes: A network-on-chip architecture for gigascale systems-on-chip," *IEEE Circuits and Systems Magazine*, vol. 4, no. 2, pp. 18–31, 2004. 69

[12] M. Coppola, R. Locatelli, G. Maruccia, L. Pieralisi, and A. Scandurra, "Spidergon: a novel on-chip communication network," in *Proceedings of 2004 International Symposium on System-on-Chip*, 2004, pp. 15–. 69

[13] F. Moraes, N. Calazans, A. Mello, L. Moller, and L. Ost, "Hermes: an infrastructure for low area overhead packet-switching networks on chip," *Integration, the VLSI Journal*, April 2004. 69

[14] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny, "Routing table minimization for irregular mesh nocs," in *Proceedings of the conference on Design, Automation and Test in Europe*, 2007, pp. 942–947. 69

[15] E. Beigne, F. Clermidy, P. V. A. Clouard, and M. Renaudin, "An asynchronous noc architecture providing low latency service and its multi-level design framework," in *11th IEEE International Symposium on Asynchronous Circuits and Systems*, 2005, pp. 54–63. 69

[16] D. Starobinski, M. Karpovsky, and L. A. Zakrevski, "Application of network calculus to general topologies using turn-prohibition," *IEEE/ACM Transactions on Networking*, vol. 11, no. 3, pp. 411–421, Jun. 2003. 69

[17] M. Ali, M. Welzl, and S. Hellebrand, "A dynamic routing mechanism for network on chip," in *Proceedings of the 23rd NORCHIP Conference*, 2005, pp. 70–73. 69

[18] M. Palesi, S. Kumar, and R. Holsmark, "A method for router table compression for application specific routing in mesh topology noc architectures," in *SAMOS*, 2006, pp. 373–384. 69

[19] J. Flich, A. Mejia, P. Lopez, and J. Duato, "Region-based routing: An efficient routing mechanism to tackle unreliable hardware in network on chips," in *NOCS '07: Proceedings of the First International Symposium on Networks-on-Chip*. Washington, DC, USA: IEEE Computer Society, 2007, pp. 183–194. 69

[20] F. Angiolini, D. Atienza, S. Murali, L. Benini, and G. D. Micheli, "Reliability support for on-chip memories using networks-on-chip," in *ICCD*, 2006. 69

[21] Synopsys, "Physical Compiler," http://www.synopsys.com. 69, 73

[22] P. Fiser and H. Kubatova, "Two-level boolean minimizer boom-ii," in *Proceedings of 6th Int. Workshop on Boolean Problems (IWSBP'04)*, 2004, pp. 221–228. 69, 74

[23] B. L. Synthesis and V. Group, "Abc: A system for sequential synthesis and verification," Berkeley, Tech. Rep. 69, 74

[24] F. Angiolini, P. Meloni, S. Carta, L. Raffo, , and L. Benini, "A layout-aware analysis of networks-on-chip and traditional interconnects for mpsocs," vol. 26, Mar 2007, pp. 421–434. 73

6

3D NoCs - Unifying Inter & Intra chip Communication

Networks-on-chip have been developed in the last few years to address the scalability challenges of global on-chip communication. VLSI technology is now rapidly moving into vertical stacking to overcome fundamental communication and integration bottlenecks, however this technology is not mature yet, and significant reliability challenges must be overcome. In this chapter we describe our effort in establishing a 3DNoC design flow and in designing circuits and architectural solutions for variability and reliability characterization and tolerance.

6.1 Introduction

Interconnect scaling in nanometer technology brings limited advantages, and VLSI systems are becoming increasingly interconnect-dominated. Three-Dimensional Integrated Circuits (3DICs) alleviate the interconnect I/O bandwidth and latency bottlenecks, by leveraging the vertical axis to minimize communication distances and to provide more connectivity among blocks. 3DICs may also enable heterogeneous integration with improved performance and energy efficiency (e.g. technologies from Tezzaron Semiconductor Corporation (1), IMEC, MIT Lincoln Labs, and IBM (2)). One of the most promising approaches for 3D integration is based on Through Silicon Vias (TSVs), pillars manufactured across thinned silicon substrates to establish inter-die connectivity after die bonding. Salient TSVs features include fine pitches, high densities and high

compatibility with the standard CMOS process.

Three-Dimensional Networks-on-Chip (3DNoCs) combine the benefits of short vertical interconnects of 3DICs and the scalability of NoCs. A vertical link can be physically implemented as a cluster of TSVs. Unfortunately, currently available processes for TSV fabrication have low yield relative to standard 2D processes, thus fault-tolerance schemes and built-in sensor are needed to alleviate and quantify the 3D process variation. 3D-NoCs have a significant advantage with respect to unstructured interconnects, as they enable to focus yield-loss countermeasures at the vertical NoC-link level of granularity, thereby enabling efficient sharing of the redundant circuits required to support self-calibration and fault tolerance.

In this work we first introduce a simple circuit-level model for vertical TSV-based interconnects based on accurate 3D parasitic extraction, and we describe an advanced 3D NoC design flow that allows post layout verification of the 3D stack. We then focus on reliability enhancement techniques for 3D NoCs based on fault-tolerance and post-silicon calibration. We propose a defect-tolerant multi-bit vertical link interface and circuits for TSV process characterization, that allow accurate post-silicon characterization of TSV RC parasitics and process variability, as needed to calibrate NoC operating frequency.

6.2 Related Work

NoCs have been suggested as a scalable communication fabric (3), but research in 3D NoC is only starting. A few works partially address the characterization the vertical interconnects for use in 3DNoCs with respect to physical implementation and timing requirements. In (4) the authors present various possible 3D topologies for 3D NoC, considering power an latency cost. In (5) a comparison of three 3D clock distribution network topologies is presented. In (6) and (7) the author presents a design tool to synthesize application-specific 3D-NoCs. This tool is able to find the best NoC topology, and to assign the network components on to the 3D layers and performs a placement of them in each layer.

As technology scales, fault tolerance is becoming a key concern for on-chip communication. Optical Proximity Correction (OPC) and redundant via placement (8) have solved a huge number of cases of faults related, mainly, to interconnects. Several fault

tolerant algorithms for on-chip interconnects have been presented by (9), but as the authors emphasize, this approach is not well suited to the NoC context due to the large area cost. Experiments by HRI on 3DICs report relatively low yield, at around 60%; the redundancy scheme used realizes each vertical interconnect expensively as a pair of vias (twins) (10).

Several significant achievements have been recently published, confirming the rapidly increasing industrial R&D effort in this area. In (11) an 8Gb 3D DDR3 using TSVs to stack 4 DRAM dies is presented. This memory uses a set of redundant TSVs with a check-and-repair scheme to increase the chip yield. Vertical vias are overprovisioned by 2:1 and 4:2, achieving a target yield of 95% and 99.8% respectively. An operational 3DNoC prototype is described in (12), but its frequency of operation is severely limited by unknown process-related issues in 3D stacking. These early prototypes confirm that vertical interconnects should be protected from both hard faults and parametric reliability losses.

In this chapter, we propose both a detailed characterization of the vertical links and switches, and novel scheme to overcome the yield limitation and to perform post-silicon characterization of TSVs. The starting point of this work is (13, 14), where a thorough physical and timing analysis of the vertical links has been conducted on a real 3DNoC.

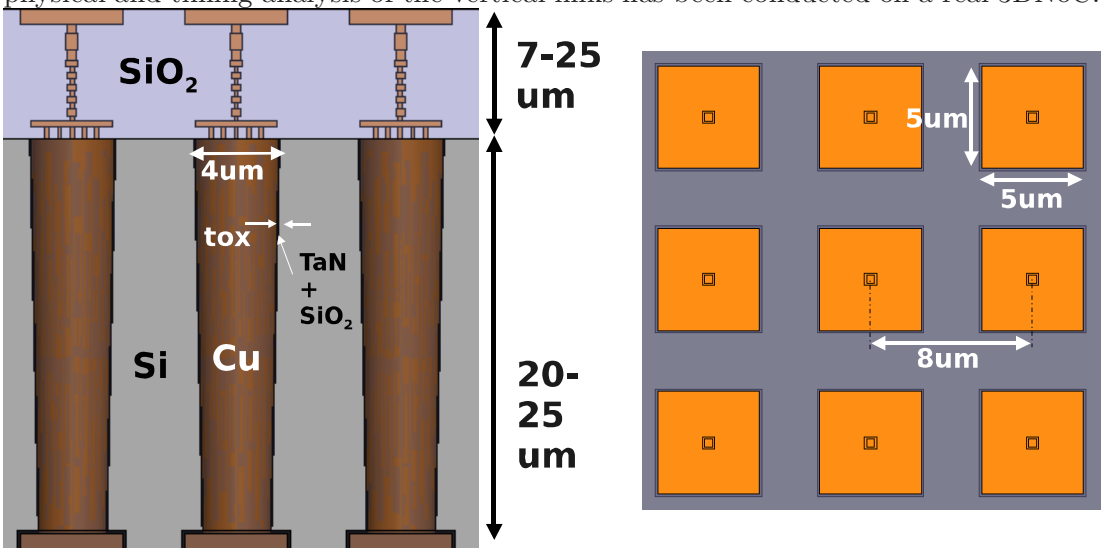


Figure 6.1: Schematic representation of a bundle of 3D Vias

6.3 3D NoC Design

In this section, we discuss the main changes in the NoC switches and finally we present the 3D flow for a small NoC topology. We tuned our flow on the results presented in (13), where we described the modeling of the performance of vertical interconnects (Figure 6.1) to assess 3DNoC implementation tradeoffs. Delay estimates through a SPICE simulation result in 18.5ps for TSVs of $4\mu m$ diameter and $8\mu m$ pitch. As a consequence, even after taking coupling effects of tightly packed TSV bundles into account, vertical links turn out to be substantially faster and more energy efficient than moderate size planar links.

6.3.1 3DNoC Architectural and Physical Design

NoC components and NoC design tools require modifications to support vertical links made of TSVs. We choose to base our integration effort on the ×pipes (15) NoC library, which supports arbitrary connectivity, and on its instantiation toolchain (16).

×pipes switches adopt a lean architecture, where only switch inputs are buffered. The switch logic and the link propagation time (up to the following switch or to the first link pipeline stage) contribute to a same timing path, which becomes the bottleneck for the system. The link propagation time directly impacts the maximum operating frequency of the switches and so of the whole NoC.

We leverage the models described at the beginning of this section to build LEF (Library Exchange Format) and LIB descriptions of vertical vias and pads. Based on these models, TSVs can be accurately inserted within the design during the placement and routing stage; they are simply attached to the input or output pins of a switch port, just as a horizontal wire would. At the RTL level, the design remains completely unchanged with respect to a 2D implementation. This brings several advantages: (i) the presence of vertical wires is totally transparent to the architectural and functional views of the architecture; (ii) a chip may feature any degree of connectivity heterogeneity since vertical links can be added or exchanged for horizontal ones; (iii) vertical bandwidth can be added only where needed in the chip, saving switch ports everywhere else; (iv) building upon the savings brought by the previous item, the set of switches with vertical ports, *i.e.* the ones located where vertical bandwidth is really needed, can have ideal performance because they can be implemented as full crossbars.

Thanks to this approach, a complete flow is achieved; this includes the ability to extract and simulate a 3D layout, where all switch ports are exposed to proper timing constraints and load information is available for both horizontal and vertical connections. A depiction of a sample layout featuring a 5x5 switch with vertical ports (UP direction with 64-bit data width) is presented in Figure 6.2. The TSV macros (or top pads) are placed close to the pin-out of the switch block, shortening the wires leading to the base of the via, thus reducing parasitics and improving timing.

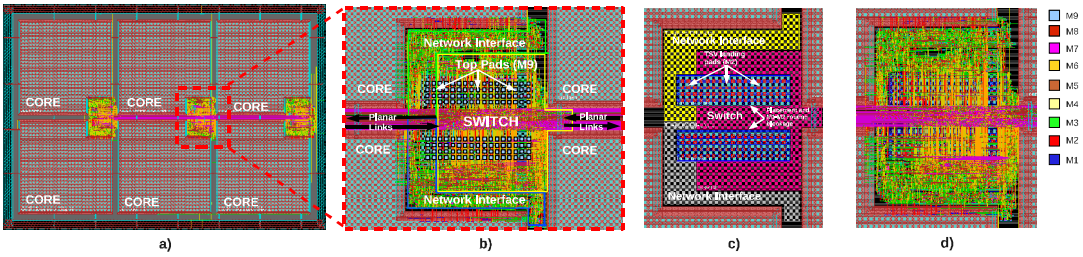


Figure 6.2: Layout details (65nm) of a NoC topology and switches with 3D ports. a): a topology where switches feature the UP port. b): detail of a switch with an UP port (IO pads on M9): Metal 8 and 9 are reserved for the vertical link routing and bonding. c): floorplan of a switch with a DOWN port. The TSV hard macros are placed close to the switch. d): post-place&route detail of a switch with a DOWN port.

6.4 Addressing 3DIC reliability and Variability challenges

As we seen in Section 6.3, a vertical link is physically implemented as a cluster of TSVs. Unfortunately, the available processes for TSV fabrication have relatively low yield (compared to standard 2D processes), and the development of 3D ICs results in a significant yield loss. In this context, the adoption of a solution that acts to mitigate the effects of the low reliability of the 3D process is mandatory. Many types of failure mechanisms can take a place, and they are categorized in two classes: systematic and random faults.

While systematic defects affect the whole wafer in the same way, random defects affect only some random spots. Random defects comprise a variety of physical phenomena during e.g. the thermal compression process used in wafer stacking, TSV insulating and filling, that eventually leading to opens and shorts along TSVs and variation in the nominal resistance and capacitance. These defects can be recognized as stuck-at,

stuck-open and delay fault. Stuck-at happens when the TSV is stuck on a logic value, and usually is due to short between the TSV and the substrate. Stuck-opens can be produced by void formation during metal filling or missing bonding on the TSV interface. These defect can also manifest themselves in a gradual fashion, where TSVs are still functional, but with much degraded speed. In the following we focus on countermeasures for stuck-at and stuck-open faults.

6.4.1 Stuck-at and stuck-open remedies

A TSV that is shorted to the substrate or open, must be replaced with a spare. Extra TSV may be available in a limited number, or in the best case each TSV has a backup. The simplest solution relies on via duplication, where each signal is assigned to two TSV, but only one is used effectively at the same time. A more efficient solution shares some spares between a wide number of TSV (usually with a ratio signal/spares greater than 4), like the technique that has been used for many years in the DRAM manufacturing. The 3D link interface must provide the right connectivity and fault isolation, in order to operate in a safe condition. It is clear that despite the first solution is the most expensive, whereas each TSV is duplicated and most of the overhead is due TSV obstruction, it can take advantages of the simple and lean architecture needed to enable the TSV test and reconfiguration. Figure 6.3 shows a detail of the fault

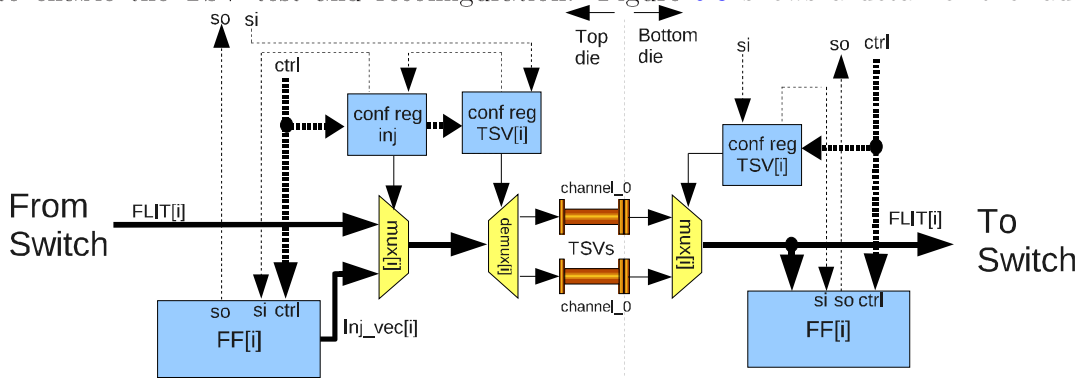


Figure 6.3: Detail of the fault tolerant 3D link interface

tolerant 3D link interface. During the operative mode, the signal $FLIT[i]$ is routed through one of the two TSVs, and the selected via depend on the setup stored inside the configuration registers. During the test mode, these registers are programmed systematically, in order to select one of the two channels and inject the stimulus through

the serial input. After the stimulus is propagated, it is captured and scanned out and the response analysis is performed off chip.

6.4.2 TSV Variability

As outlined in section 6.2 hard TSV failures are only one part of the problem. The immature TSV fabrication processes are also characterized by a wide variability range. Hence, we need approaches to efficiently characterize TSV performance after manufacturing. Since a TSV can be considered as a capacitive load, the simplest way to measure the parasitics and the variability of a TSV is to insert it in a ring oscillator. The frequency of a balanced (all stages are equivalent) ring oscillator depends on two factors: the number of stages and the sum of the propagation delays low to high and high to low.

$$f_{osc} = \frac{1}{n \times (t_{LH} + t_{HL})} \tag{6.1}$$

In our context, the ring oscillator is not balanced, and therefore the frequency can be written in this way:

$$f_{osc} = \frac{1}{(n-1) \times (t_{LH} + t_{HL}) + (t_{LH_{TSV}} + t_{HL_{TSV}})} \tag{6.2}$$

where the first term on the denominator, depends only on the number of stages, while the second term is directly related to the TSV load. Whereas we want to analyze TSV variation in the range of $\pm 10\%$ we maximized the frequency sweep due variation, therefore as we can see in the Equation 6.2, we decreased the number of stages, until we noticed swing penalties on the the internal nets. In order to be readable from the Automatic Test Equipment (ATE), the ring oscillator frequency is managed by a divider. It has been built with toggle flip flop ad it is composed by 8 stages, which divides the input frequency by 256, therefore ensuring that the frequency on the output pad is in the range 0 to 25MHz. Each ring oscillator has an enable signal that allows to stop the oscillations after the characterization phase, to reduce power consumption and heat. The divider can be shared between several ring oscillators in order to increase the number of measurement per die, therefore each ring output is multiplexed to the divider, and the enables are provided coherently.

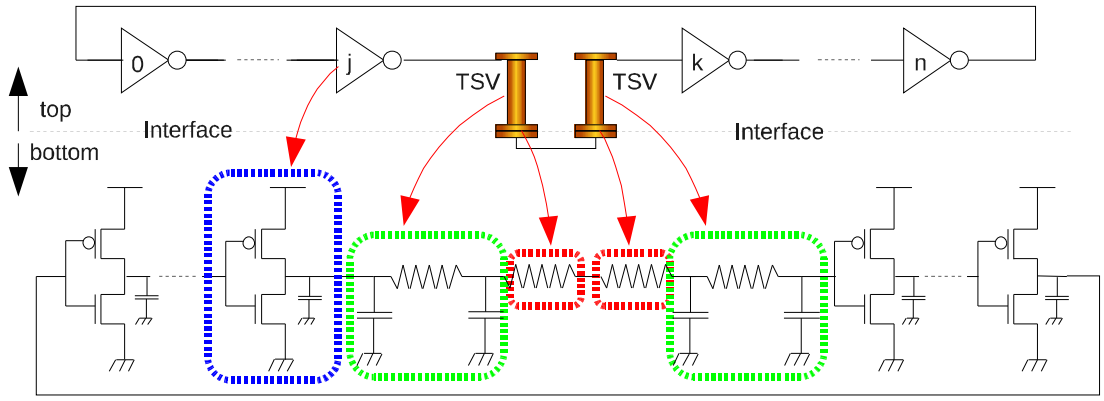


Figure 6.4: Ring Oscillator with two TSVs load and schematic

6.4.3 Coupling with substrate

A second issue in 3D stacked dies is the presence of a non negligible coupling between top metal layer and the substrate of the die that is stacked on top of it. In a 2D die, the space above the top metal is covered by insulator, therefore most of the coupling is due cross capacitance between on the same layer. The stacking of a silicon substrate creates additional cross capacitance, increasing the net load and therefore degrading the timing performance of the 3D integrated circuit. Quantify this overhead is very important to tune the back end tools for accurate post place and route simulation. A wrong characterization of the 3D geometries may lead in a underestimated cross-capacitance, and a sensitive mismatch between simulations and silicon measurement.

A simple way to quantify this overhead, is to create two identical simple structures, and replicated in two different environment, such as, in between the die stack, or in the top die. Whereas the variation will be in terms of coupling capacitance, we used a ring oscillator, similar to the ones used to characterize the TSV variations, but tuned on the expected value of the cross-capacitance. We built several ring oscillators, each made of 33 inverter stages and one fat and long metal track with a variable width. As seen in the previous subsection, the 8 stage divider may be shared and the ring oscillator output multiplexed. To avoid thermal gradients, rings are normally turned off, and activated one at a time for several milliseconds to perform measurement. The equations 6.1 and 6.2 are still valid, but in this case the load variation is attributable to coupling capacitance between top metal and substrate.

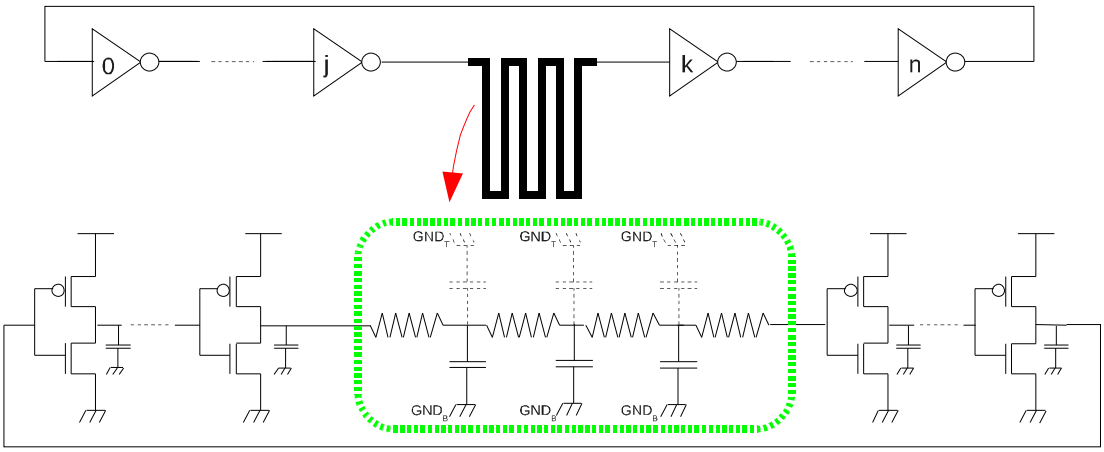


Figure 6.5: Ring Oscillator with a long & fat top metal track and schematic

6.5 3D NOC silicon demonstrator

To demonstrate the feasibility of NoCs in 3D technology, we designed and taped out a 16-bit 3D NoC distributed across two tiers using the ×pipes synthesizable NoC IP and tool chain (15), with extensions for supporting vertical links, as described in section 6.4. Each tier consists of a traffic generator, a slave memory, a 3x3 switch and a JTAG controller and the above-mentioned test structures for TSV characterization and 3D technology tuning. Figure 6.6 shows the block diagram of the designed test chip.

6.5.1 Traffic Generators

The traffic generators (TGEN) mimic logic IP components. They can send/receive flits across the NoC at speed to and from each memory on each tier and the type of transaction can be programmed via JTAG controller. For sake design simplicity, we implement the network interface within these blocks, therefore the TGEN injects flits instead of data. Our traffic generator is composed by two sections, namely request and response. The former is responsible to pack transactions into flits that correspond to read and write transaction and injects it into the router. The latter is responsible to collect incoming flits (only for read transactions), to unpack them and to store it inside a local register called RDATA. The data width is 8 bit. The request section is fully configurable, through the JTAG controller. Serial input, serial output and control signals are used to program the traffic generator or scan out the RDATA register.

For each parameter(burst type, burst length, data to be written etc.), we create a register that at boot time is initialized with the default transaction, while in the programming mode, are connected in a single chain (even RDATA), allowing to inject the setup configuration through the serial input or scan out the RDATA through the serial output.

6.5.2 Memory IP

The Memory block acts as a small memory with an embedded network interface. The memory size is 64bit arranged in 8 words wide 8 bit. This block is divided in two parts: receive and resend. The first is responsible for incoming flits collection, and packing them to create the proper informations. In case of write transaction, the data is written in the memory, while in case of read, the request is forwarded to the resend stage that create the response flits containing the read data. The memory is implemented with registers, and during testing stage, these registers are linked together to form a single scan chain, therefore allowing to read the memory content through the serial output.

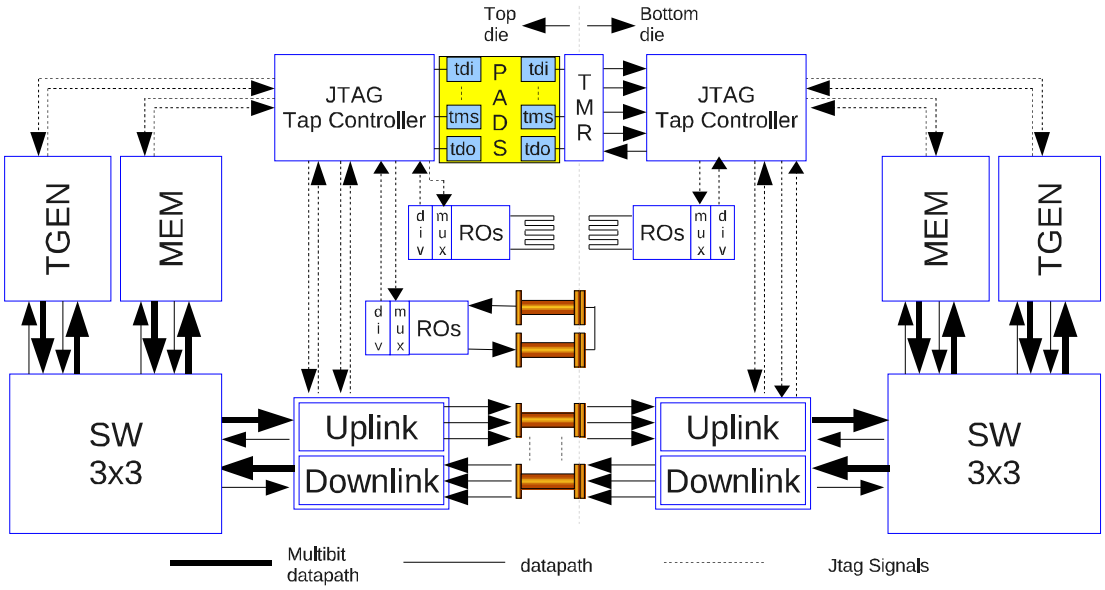


Figure 6.6: Block Diagram of the Test Chip

6.5.3 3D Link

The 3D link, as introduced in Section 6.4.1, ensures the point-to-point connectivity between the in-to-out and vice versa switch's port, and it is composed by the TSVs, the logic for fault bypassing, and obviously, the overhead to support the vertical interconnect test. Our links are unidirectional, therefore the channels are divided in two parts: the uplink, which manages the outgoing traffic, and the downlink that operates on the incoming traffic. Our 3D link is targeted to tolerate static faults like stuck at and stuck open. Both the downlink and the uplink are equipped with a register (18 bit wide) used only for testing purposes, and some registers (19 bit) to store the desired TSVs configuration. The Link is also provided with serial input and serial output, necessary to inject test vectors and to select one of the two TSV channels for each bit.

6.5.4 JTAG Controller

To enable testing and programming and to reduce the number of IO pads we inserted two JTAG controllers, respectively one for the top and one for the bottom tier. The tap controller is compliant with the IEEE 1149.1 standard, and allows to select one of the four testable/programmable block (TGEN, memory, uplink and downlink), to inject a new configuration vector and scan out the scan chains. Te selection of one of these blocks comes in two phases: capture the block ID, and then scan-in vector or scan-out the scan chain. For sake simplicity, we targeted the design to run at 100MHz (our equipment has a limitation of 25MHz). JTAG clock and reset are shared with the whole design (TCK and TRST) and implemented as a synchronous design. The clock is provided from an off-chip source.

While the 3D link provides connectivity between the NoC, the bottom tier JTAG signals are feed from the top tier (where the global IO pads reside) and routed to the bottom tier using TSV and triple modular redundancy. For the obvious reasons explained in the previous section, each TSV is decoupled from the others.

Our 3D NoC Design not only allows to run functional test, but permits to collect statistics on TSV yield, whereas each TSV can be individually tested at boot time, and even allows to collect data about the variability of the TSV parasitics and the extra coupling capacitance needed for early time budgeting and technology tuning.

The design has been implemented using the IMEC 130nm process, using one poly and two metal layers. The die size is roughly one square millimeter, and includes 69K transistors, 148 TSVs and 12 IO pads. Silicon measurement on the fabricated dies are under way at the time of writing.

6.6 Conclusions

In this chapter, we have presented a characterization of 3D link and a 3D NoC design flow. We then proposed a simple but efficient reconfiguration scheme capable to isolate faulty TSV. Furthermore we introduced two test structures used to collect data about TSV variability and cross-coupling, needed for tech tuning and post-silicon calibration. Finally we described a test chip containing a small 3D NoC, equipped with fault tolerant 3D link interface, end embedded test structures for TSV variability measurement and metal-substrate coupling.

Bibliography

[1] R. S. Patti, "Three-dimensional integrated circuits and the future of system-on-chip designs," *Proceedings of the IEEE*, vol. 94, no. 6, June 2006. 85

[2] A. W. Topol et al. , "Three-dimensional integrated circuits," *IBM Journal of Research and Development*, vol. 50, no. 4/5, pp. 491–506, July/September 2006. 85

[3] L. Benini and G. De Micheli, "Networks on chip: a new SoC paradigm," *IEEE Computer*, vol. 35, no. 1, pp. 70–78, January 2002. 86

[4] V.F. Pavlidis et al., "Clock distribution networks for 3-d integrated circuits," in *CICC*, 2008, pp. 651–654. 86

[5] V. F. Pavlidis et al., "3-d topologies for networks-on-chip," in *IEEE transactions on VLSI*, October 2007, pp. 1081–1090. 86

[6] C. Seiculescu et al., "Sunfloor 3d: A tool for networks on chip topology synthesis for 3d system on chips," in *Proceding of DATE09*, 2009, pp. 9–14. 86

[7] S. Murali et al., "Synthesis of networks on chips for 3d systems on chips," in *ASP-DAC*, 2009, pp. 242–247. 86

[8] Mark Rencher et al., "Why interconnect and litography modeling impacts yield," in *What's Yield got to do with IC*, vol. 1, 2002. 86

[9] M. Pirretti, G. M. Link, R. R. Brooks, N. Vijaykrishnan, M. Kandemir, and M. J. Irwin, "Fault tolerant algorithms for network-on-chip interconnect," in *ISVLSI*, vol. 26, 2004, pp. 46–51. 87

[10] N. Miyakawa, T. Maebashi, N. Nakamura, S. Nakayama, E. Hashimoto, and S. Toyoda, *New Multi-Layer Stacking Technology and Trial Manufacture*, November 2007, honda Research Institute Japan Co. Ltd. 87

[11] Uksong Kang et al., "8gb 3d ddr3 dram using through-silicon-via technology," in *ISSCC*, 2009, pp. 130–132. 87

[12] C. Mineo et al., "Inter-die signaling in three dimensional integrated circuits," in *Procedings of CICC*, 2008, pp. 655–658. 87

[13] I. Loi et al., "Supporting vertical links for 3d networks-on-chip: Toward an automated design and analysis flow," in *Proceedings of Nano-Net*, 2007, pp. 23–27. 87, 88

[14] I.Loi et al., "Developing mesochronous synchronizer to enable 3d nocs," in *Proceedings of DATE conference*, 2008, pp. 1414–1419. 87

[15] F. Angiolini et al., "Contrasting a noc and a traditional interconnect fabric with layout awareness," in *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition 2006*, 2006, pp. 124–129. 88, 93

[16] S. Murali et al., "Designing message-dependent deadlock free networks on chips for application-specific systems on chips," in *VLSI-SoC*, 2006, pp. 158–163. 88

7

A 3D Network-on-Chip to Unify Inter/Intra-Die Communication With 5 m TSVs

3D TSV technologies enable the heterogeneous integration of systems, where IP components can be distributed across multiple tiers, but remain tightly interconnected. A 3D Network-on-Chip unifying both inter/intra-chip communication between the IP components will hereto be presented. The 3D link consists of 100 TSVs of 5um diameter and 10um pitch. TSV (through silicon via) are an essential technology towards higher and more heterogeneous system integration. Figure 7.1 shows several existing 3D Integration schemes proposed in the last years. These schemes mainly differ in via diameter/pitch and via aspect ratio. In this chapter, we focus on Cu 3D SIC TSV technology (1), as it hits the sweet-spot between cost and application flexibility. Die-to-wafer stacking is great for cost as it creates the possibility to stack dies of different sizes, and thus not add unnecessary constraints during floor planning, which may limit die utilization. It also allows Known-Good-Die testing before stacking, thereby increasing compound yield significantly, particularly when dealing with new designs/technologies where initial 2D yield may be low. The higher interconnect density with respect to WLP enables system integration across multiple dies at the level of IP components rather than bond-pads. The least intrusive way to distribute IP components is then to partition the design at the level of the communication architecture, which then must support both inter and intra die communication.

7. A 3D NETWORK-ON-CHIP TO UNIFY INTER/INTRA-DIE COMMUNICATION WITH 5 M TSVS

In order to enable heterogeneous integration with TSVs, a communication architecture that unifies on-chip and off-chip communication must be developed. The communication architecture of choice in today's state-of-the-art designs are structured and scalable Networks-on-Chip (2, 3). The extension of the NoC paradigm to 3D integrated circuits is very promising, as modularity and scalability are even more critical for future three-dimensional integrated systems (4). To demonstrate the feasibility of this technology, we designed and manufactured a 16-bit 3D NoC distributed across 2 tiers using the iNoCs IP and tool chain. Each tier consists of a traffic generator, a slave memory, a 3x3 switch and a JTAG controller. The traffic generators mimic logic IP components. They can send/receive data packets to and from each memory on each tier. The JTAG controllers are used to program the traffic generators, to structurally test the TSVs and read/write from the memories. To enable testing of bottom tier after stacking, the JTAG pads of the bottom tier are replicated on the top tier. The PAD SELECT block connects these replicated pads to the JTAG bottom block if the TOP_PRESENT signal is high, and thus the top die is present. Otherwise, the bottom pads are selected.

In our process TSVs technology, the most frequently occurring fault are opens. To day, no shorts to bulk have been observed. Therefore, we connect two TSVs in parallel for each logic wire to increase yield. The NoC must be testable prior to assembly to enable KGD testing. Hereto, a diode in inversion acting as a weak pull down is attached to each TSV at the receiver side. The leakage current of the diode ensures that input of the TSV_SLECT_MUX is never floating but always driven to a logic value. After assembly, the TSV_SLECT_MUX allows us to collect statistics on TSV yield, as at boot time each TSV can be individually tested. Hereto, a test pattern is injected in to the Data_Scan_Top_DFF1 registers, and this is applied to the TSVs by setting the DATA_SLCT_MUX. After transmitting the data, the receiving register can be sampled and scanned in order to detect faulty TSVs (see Figure 7.8). The diagnostic information on the TSV can be used to monitor and improve the yield of the TSV process. The performance overhead is limited as only a single multiplexer is added to the 3D link. The JTAG controllers on both tiers play a crucial role for performing KGD test. To guarantee that correct test signals are transmitted between two tiers, the inter-chip signals TMS/TRST/TDO/TDI are protected with triple modular redundancy. In case of a TSV process technology where shorts to bulk frequently occur, the above TSV

test scheme can be adapted hereto by replacing the buffer driving both TSVs with a DEMUX controlled by additional configuration memory. 3D NoC operation is depicted in Figure 4. After power on, the 32 TSV SLCT MUXs and traffic generators are programmed in test mode using the JTAG scan-chains. Thereafter, the NoC is switched in operation mode (step 3): short bursts of 32-bits are transmitted in 96bit network packets. The packets are respectively sent across the TSV links from top(bottom) traffic generator to bottom(top) memory and in 2D from bottom(top) traffic generator to bottom(top) memory. Finally, the data is scanned out again. The output (tdo[2]) corresponds with the expected output (tdo[2]_exp). A separate test indicated that in both prototypes no TSV failures were found on the 3D NoC link (see Figure 7.8). To characterize the performance limits of the 3D data links, , we measure the performance of 41-stage ring-oscillators with and without TSVs Figure 5. The ring oscillators were sized to drive an expected load of 37fF per TSV. The results indicate that data can be transferred across a TSV in less than 150ps while consuming less power than 2pJ/bit. TSVs thus enable to transfer data between dies at a similar speed than intra-die data. The 3D NoC implementation demonstrates that 2D and 3D communication can be unified (Figure 6). The additional area penalty for the TSVs in the 3D is limited to 0.018mm2. The power penalty for 3D data transfers is only 3% higher with respect to 2D traffic, as the power spent per bit in each link is limited to 2pJ/bit. The delay of the 3D link is 183ps at 1.2V, enabling high speed data transfers. In this paper, we have demonstrated a testable 3D NoC manufactured with a low cost 3D TSV Stacked-IC technology. This result indicates that integration of systems with IP components distributed across multiple dies has become possible.

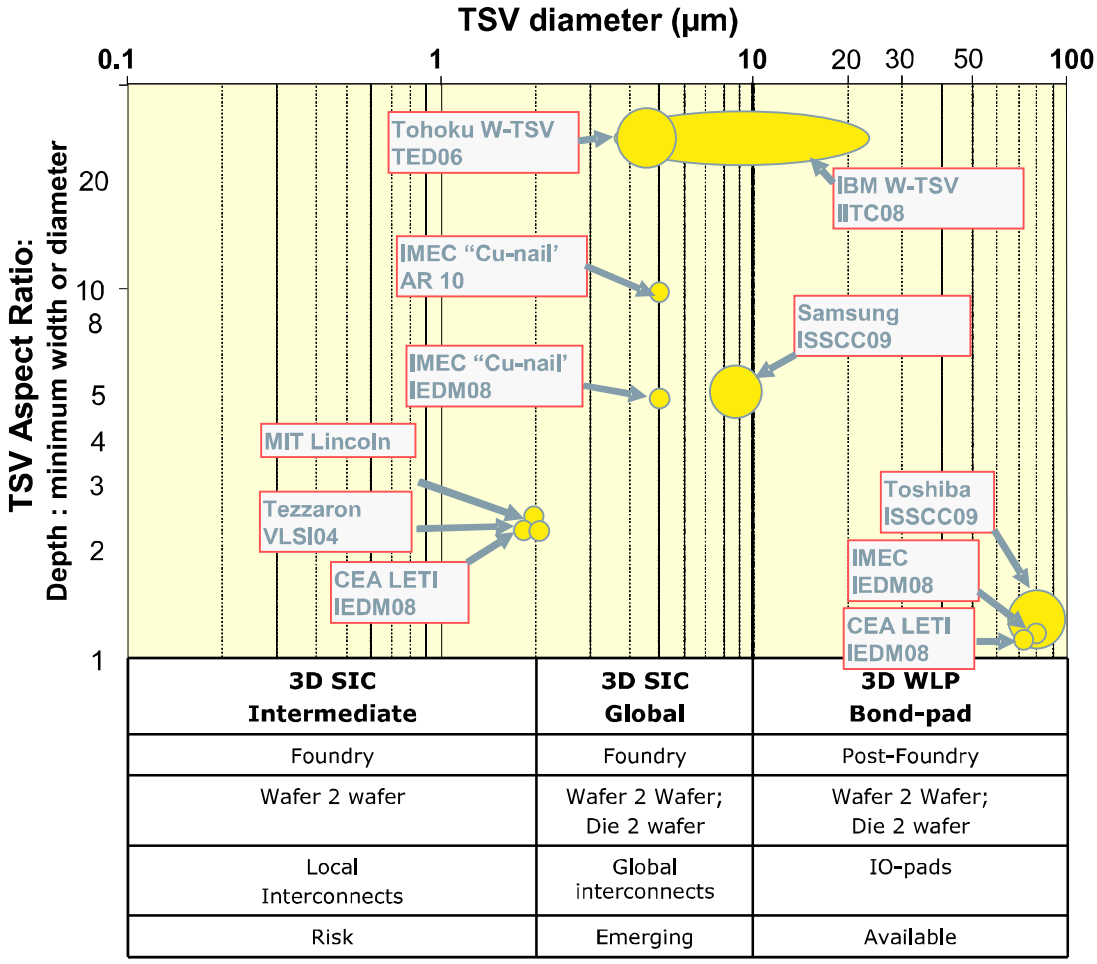


Figure 7.1: 3D SIC Technology for Global Wires Provides Highest Application Flexibility at Lowest Cost

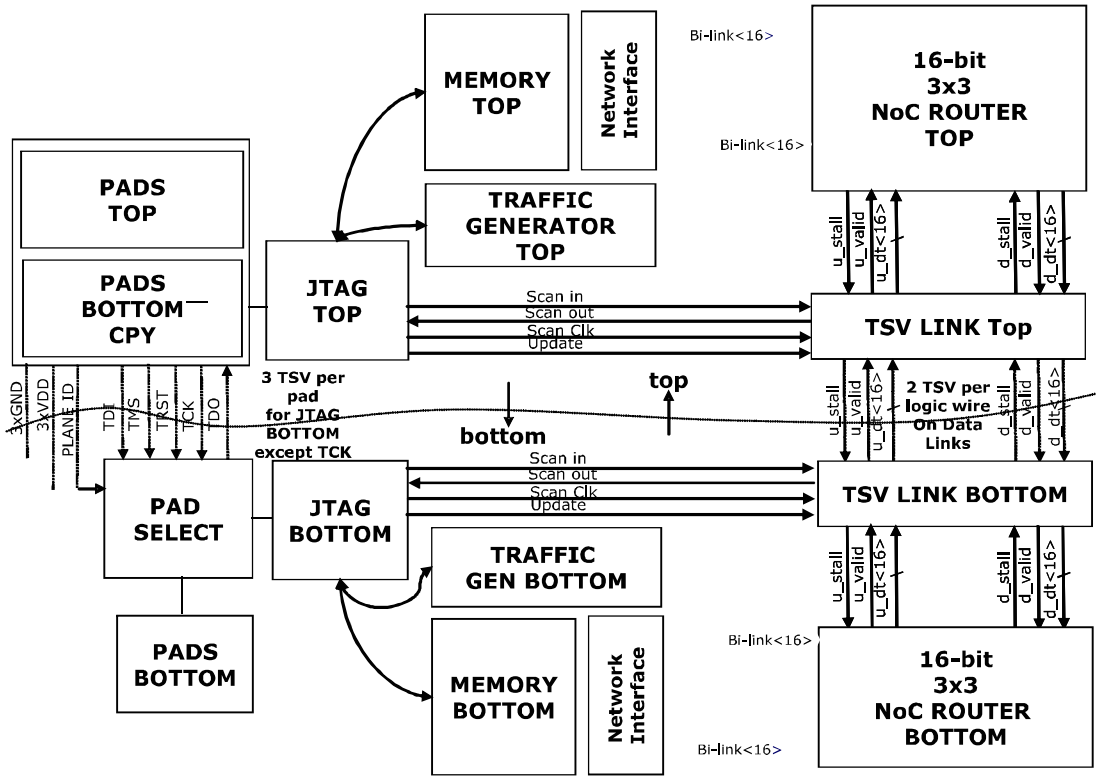


Figure 7.2: 3D NoC Schematic

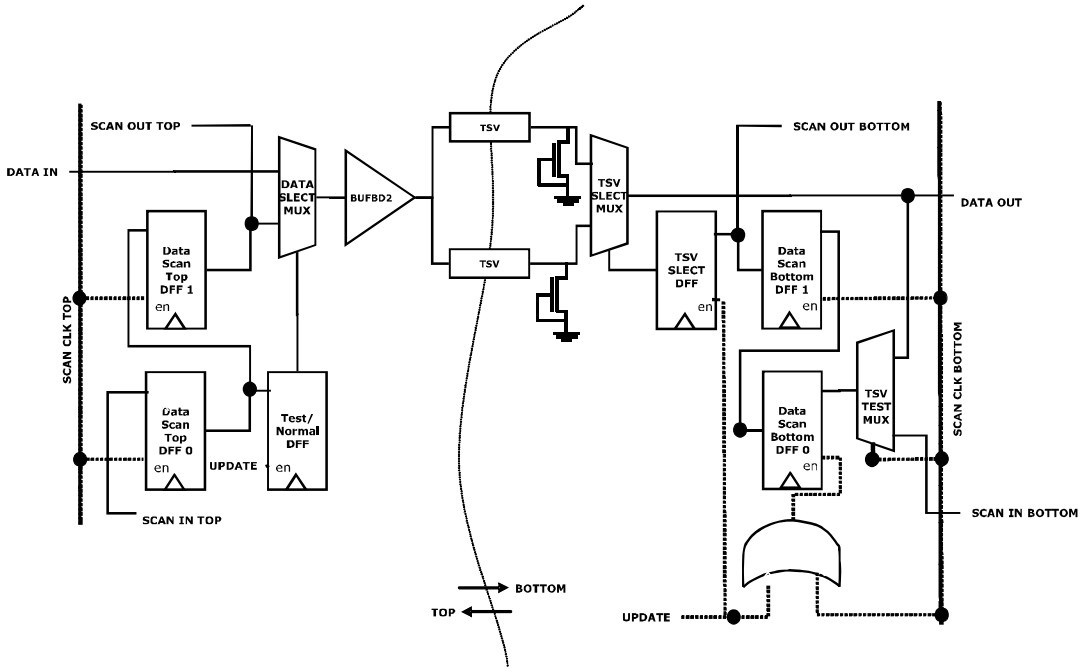


Figure 7.3: Test of TSV Data Links

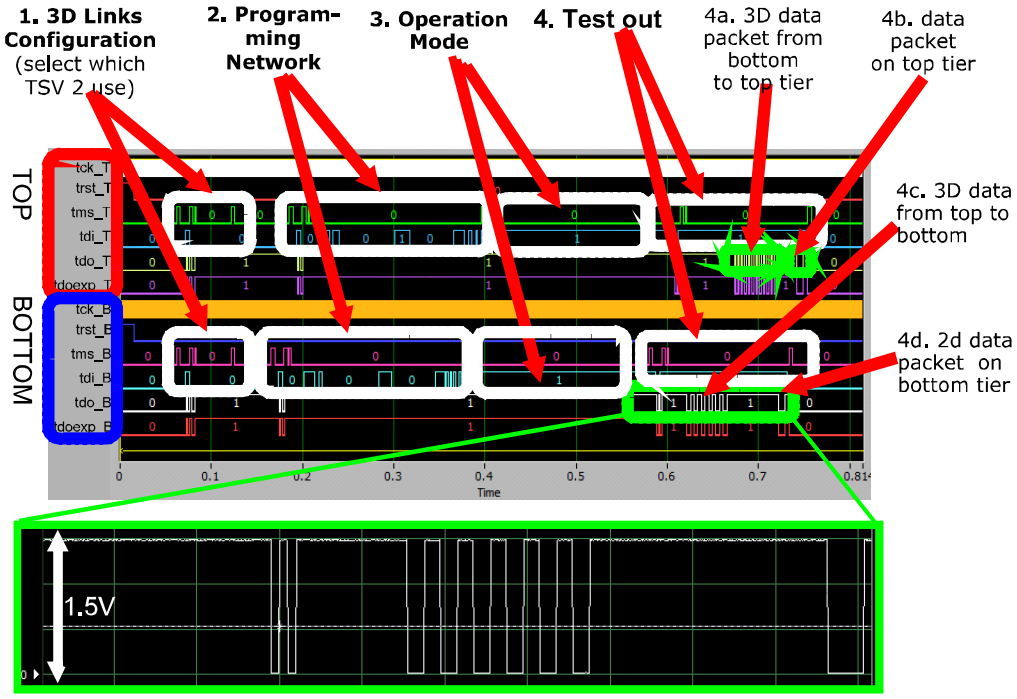


Figure 7.4: Wafer-level test of 3D NoC. Measured results (tdo[2]) correspond with logic simulation results (tdo_exp). Measured max. speed 25Mhz@0.4-1.2V/25C/130nm/200mm limited by logic analyzer

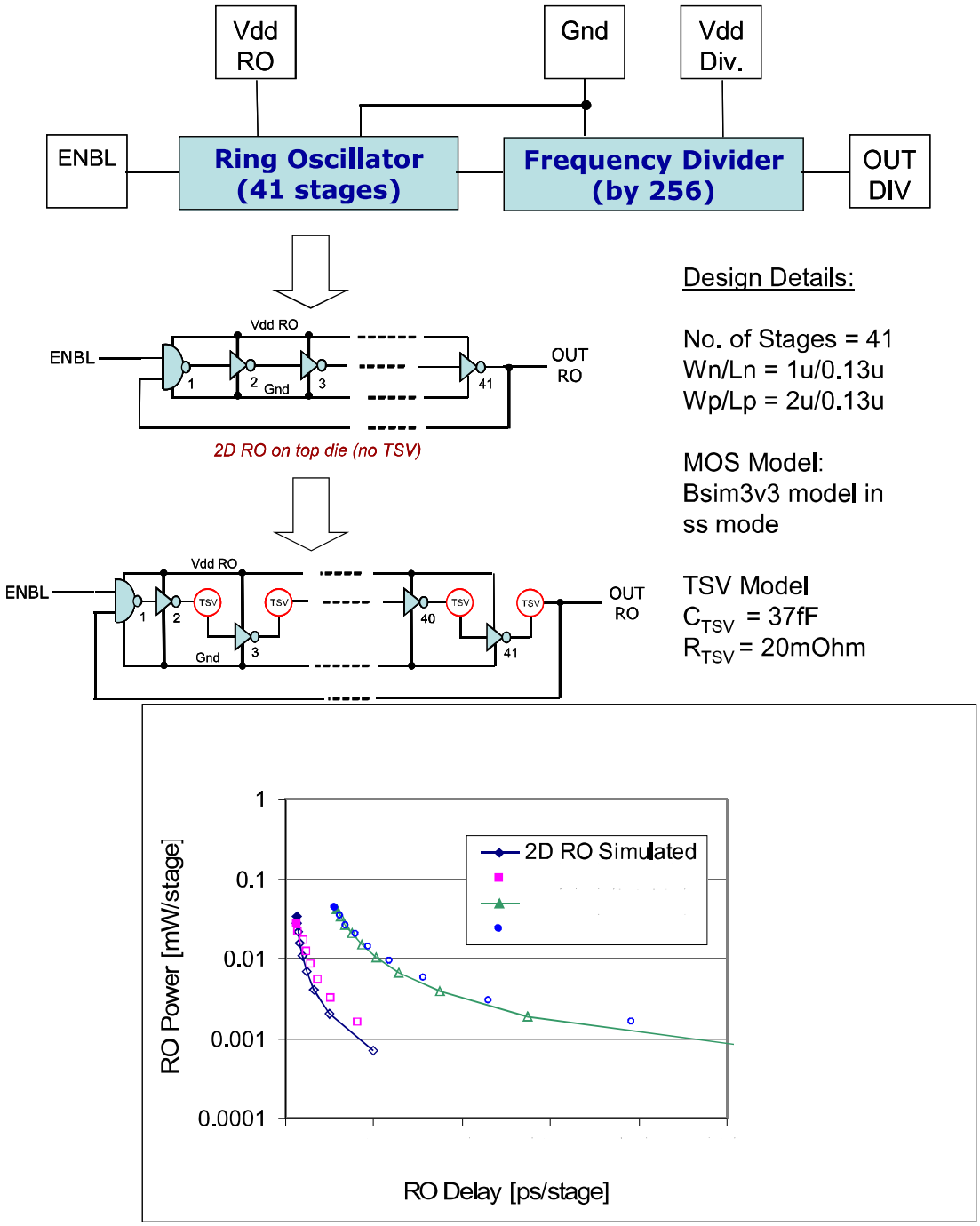
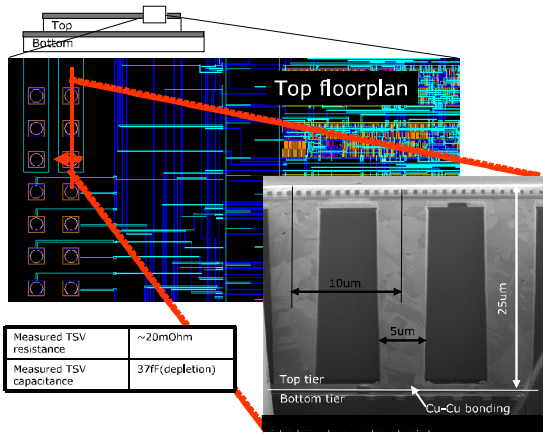


Figure 7.5: Performance of 2D versus 3D Ring Oscillator



3D NOC Test Chip		
IMEC 130nm/200mm/2 metal layers		
Gate count	2x60kGates	
Chip Area	981umx1003um	
TSV Area of 3D NoC	700umx26um	
Simulated Power 2D traffic @1.2V/50Mhz/25C*	1.34mW	
Simulated Power 3D traffic @1.2V/50Mhz/25C *	1.39mW	
Simulated 3D Link Performance		
Energy/Delay 3D link@1.2V (combinational logic between as in Figure 3)	2pJ/bit	183 ps

Figure 7.6: Manufactured 3D stack & Figures of Merit Ẽmonstrating Unification of Intra-Inter-chip Communication

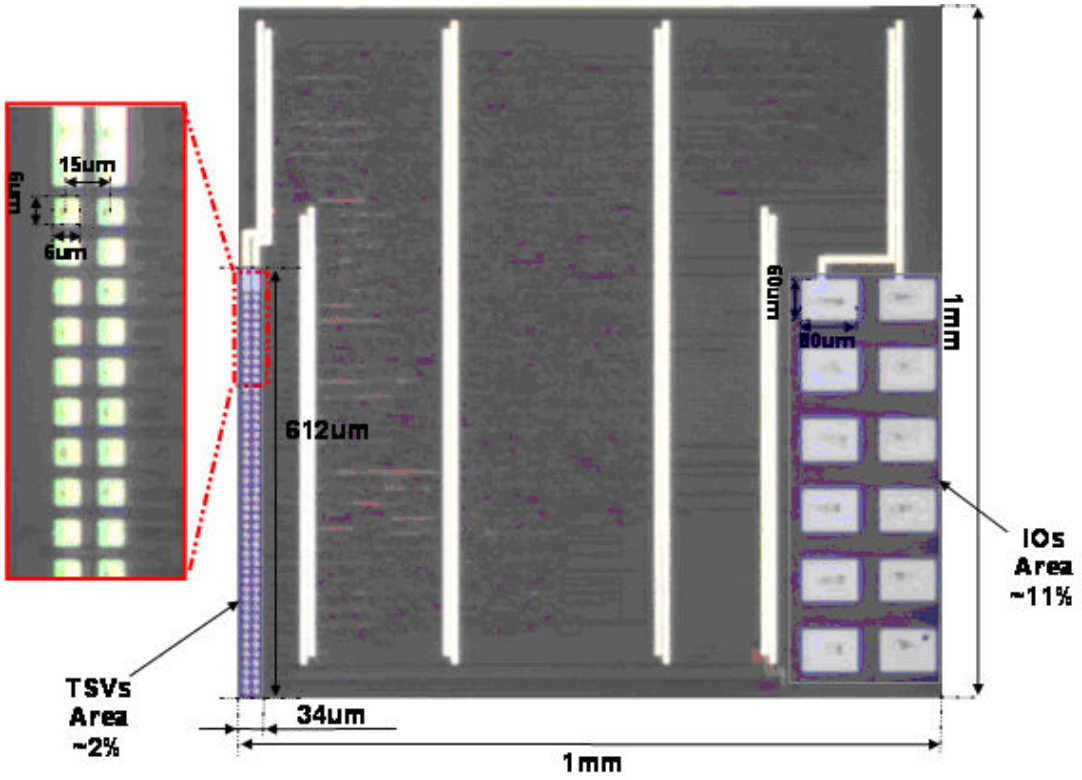
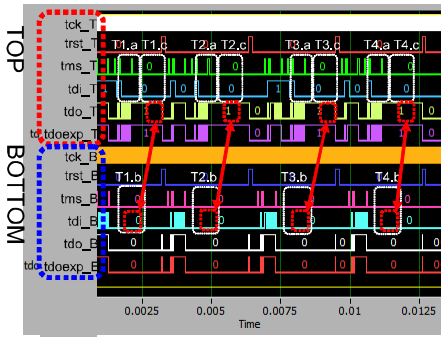


Figure 7.7: Die Picture



	Description	Note
T1a	Select TSV1 on TSV_SLECT_MUX1 (top tier)	
T1b	Inject data on TSV via DATA_SLECT_MUX (bottom tier)	Data Pattern = 01010101..01
T1c	Capture and ScanOut data on top tier	
T2a	Select TSV2 on TSV_SLECT_MUX1 (top tier)	
T2b	Inject data on TSV via DATA_SLECT_MUX (bottom tier)	Data Pattern = 01010101..01
T2c	Capture and ScanOut data on top tier	
T3a	Select TSV1 on TSV_SLECT_MUX1 (top tier)	
T3b	Inject data on TSV via DATA_SLECT_MUX (bottom tier)	Data Pattern = 10101010..10
T3c	Capture and ScanOut data on top tier	
T4a	Select TSV2 on TSV_SLECT_MUX1 (top tier)	
T4b	Inject data on TSV via DATA_SLECT_MUX (bottom tier)	Data Pattern = 10101010..10
T4c	Capture and ScanOut data on top tier	

Figure 7.8: JTAG test results of TSVs in d_link (from bottom to top tier); all 38 TSVs in this link are working

Bibliography

[1] J. Van Olmen et al., "3D stacked IC demonstration using a through Silicon Via First approach," in *Proceedings of IEDM 2008* pp.303–306. 99

[2] S. Vangal, et al., "An 80-Tile 1.28TFLOPS Network-on-Chip in 65nm CMOS," in *Proceedings of ISSCC 2007* pp.98–99. 100

[3] K. Kim et al., "A 125GOPS 583mW Network-on-chip Based Parallel Processor with Bio-inspired Visual Attention Engine," in *Proceedings of ISSCC 2008* pp.308–309. 100

[4] I.Loi et al., "A low-overhead fault tolerance scheme for TSV-based 3D network on chip links," in *Proceedings of ICCAD 2008* pp. 592–602. 100

8

Memory interface for Many-Core Platform with 3D stacked DRAM

Historically, processor performance has increased at a much faster rate than that of main memory and up-coming NoC-based many-core architectures are further tightening the memory bottleneck. 3D integration based on TSV technology may provide a solution, as it enables stacking of multiple memory layers, with orders-of-magnitude increase in memory interface bandwidth, speed and energy efficiency. To fully exploit this potential, the architectural interface to vertically stacked memory must be streamlined. In this chapter we present an efficient and flexible distributed memory interface for 3D-stacked DRAM. Our interface ensures ultra-low-latency access to the memory modules on top of each processing element (vertically local memory neighborhoods). Communication to these local modules do not travel through the NoC and takes full advantage of the lower latency of vertical interconnect, thus speeding up significantly the common case. The interface still supports a convenient global address space abstraction with high-latency remote access, due to the slower horizontal interconnect. Experimental results demonstrate significant bandwidth improvement that ranges from 1.44x to 7.40x as compared to the JEDEC standard, with peaks of 4.53GB/s for direct memory access, and 850MB/s for remote access through the NoC.

8.1 Introduction

For many years DRAM access latencies have not decreased at the same rate as microprocessor cycle times. Hence, relative memory access time (in CPU cycles) have increased from one generation to the next. This unavoidable bottleneck, often called the "memory wall", has caused the development of complex and expensive memory hierarchies with extensive data replication, and it is now one the main hurdles to the successful transition from multi-core to many-core computing platforms.

Memory access speed is not the only cause of the memory wall, which is also tied to memory-to-logic interfacing issues. DRAMs currently are developed using high-density NMOS process optimized to create high-quality capacitors and low-leakage transistors. On the other hand, logic chips are manufactured in high-speed CMOS processes optimized for transistor performance and complex multi-level metalizations. The two processes are not compatible, therefore highly optimized DRAM and logic cannot coexist on the same die[1] and they must be interfaced through off-chip interconnects. This imposes tight constraints on the maximum DRAM pin count resulting in a limited-bandwidth interface between DRAM and Logic chips. A number of countermeasures have been adopted to overcome this problem, such as multiplexing the address in two phase (RAS and CAS) and fast burst-transfer modes to increase per-pin bandwidth. This has lead to an ever-increasing power and signal integrity bottleneck in memory interfaces (1) (2), which is a major concern for bandwidth-hungry multicore architectures.

In the last few years, three-dimensional die-stacking has received a great deal of attention (3) (4). 3D stacking enables the construction of circuits using multiple layers of active silicon bonded with low-latency, high-bandwidth and very dense vertical interconnects (5). 3D stacking also enables mixing heterogeneous process technologies such as high-speed CMOS with high-density DRAM. Stacking DRAM directly on top of a processor is a natural way to attack the memory bottleneck.

In this chapter we consider a 3D-stacked DRAM on a multi-core logic die. On the horizontal logic plane cores are connected through a Network-on-chip (6, 7), which provides bandwidth in a scalable fashion, at the price of non-negligible latency, caused

[1]Embedded DRAM processes have been developed, but they are characterized by lower-density DRAMs and higher cost, hence they have not succeeded in replacing dual-chip DRAM+Logic solution for mainstream applications

by protocol translation and packetization, as well as network traversal time. In this context, a vertically stacked memory system can be modeled with the abstraction of *memory neighborhood*: each physical processing element in a large many-core array has fast, large-bandwidth access to a vertical stack of memory banks on top of it. The processor can address vertical stacks on top of other processors, but corresponding memory transactions will have to be transported through the horizontal NoC. This implies a notion of distance: the cost (increased latency and decreased bandwidth) of a memory access sharply increases as we move to memory neighborhoods to far away processors.

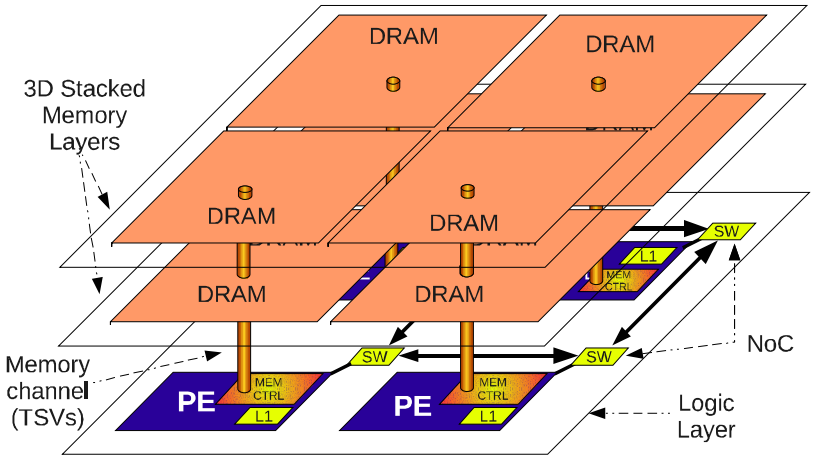


Figure 8.1: Target 3D hardware architecture

Figure 8.1 depicts a high-level view of a 3D integrated architecture and its memory neighborhoods. Note that stacked DRAM memory is still accessed using standard DRAM interfacing protocols. Hence protocol translation between processor memory accesses and DRAM transactions is still needed. The standard approach is to implement protocol translation within a memory controller connected to the NoC via a slave port. However, this is an extremely inefficient way to access a local memory neighborhood, as NoC protocol translation and packetization cost has to be paid at the master and slave interface of the NoC. Hence, we want the common case of accesses to the local memory neighborhood to be as fast as possible, bypassing the NoC interface. For remote memory accesses request, data are routed through the NoC.

The main contribution of this chapter is the development of a specialized 3D-DRAM controller featuring a very fast path to the local memory neighborhood and a standard

NoC communication facility for accesses to remote memory neighborhoods. The controller is also capable of handling requests coming from remote processors and directing them to the local memory neighborhood, while arbitrating potential conflicts with the local traffic. We also present results on the hardware cost of our memory controller, and an accurate performance analysis for a system composed by four cores, a 2x2 NoC mesh and four 3D DRAM modules. Our results demonstrate that even for such a small and low-congestion system, memory access performance is greatly increased by our distributed, asymmetric memory controller with respect to a standard NoC-interfaced distributed memory controller.

8.2 Related Work

Interconnect scaling has become one of the most crucial challenges in chip design, and is expected to get worse in the future. 3D integration and Network on Chip design methodologies are expected to overcome many of these challenges. NoCs have been suggested as a scalable communication fabric (8, 9). 3D integration has been proposed in different ways (e.g. Tezzaron Semiconductor Corporation (3), IMEC, MIT Lincoln Labs, and IBM (4)) providing promising solutions to enable connectivity along the vertical direction.

In recent years, several authors have been exploring the benefits of memory and logic stacking. Significant achievements have been announced in the last few months, confirming the rapidly increasing industrial R&D effort in this area. In (10) a 8GB 3D DDR3 using TSVs to stack 4 DRAM dies is presented. The first die includes both DRAM core, R/W buffers and IO circuitry. Redundant TSVs with check and repair scheme, and power-noise reduction method are also presented. Read and write buses are independent, but row and column address are still multiplexed as in the conventional DRAM. As the author remarks, the DRAM module are simply added on each tiers, therefore this results in increased power and area waste due to duplication of circuit components.

3D bitcells in DRAM are presented in (11). FaStack memory devices contain up to four layers of stacked DRAM on top of one control and interface layer, greatly increasing memory density up to 300% as compared to current commercial DRAM. Individual bitcell arrays are stacked in a 3D fashion, therefore reducing length of internal buses,

wordlines and bitlines, which in turn reduces the access latency of the memory. The I/O capacitance is $25fF$, or about 100x less than off-chip memory. Latency is 8-12ns, which is near SRAM speed.

A few prior studies have already started to investigate the potential of 3D-stacked memory. (12) and (13) independently researched the performance impact of placing the main memory system on top of the processor using 3D fabrication technologies. These studies report impressive performance speedups for 3D-stacked memories (92% performance gain in (12) and 65% for (13)).

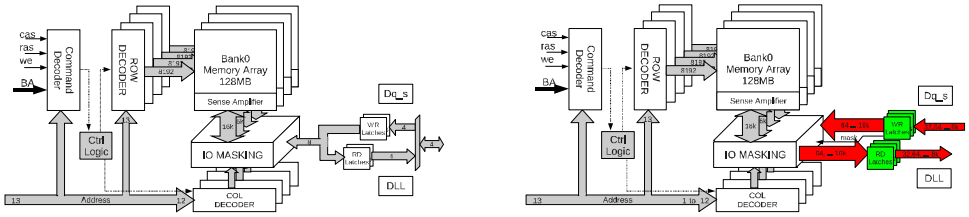
An early approach in architectural exploration for 3D DRAM stacking on a logic die is presented in (14). The authors modeled a web server as Chip Multi Processor (CMP): the logic has been placed in the first die, and the memory (DRAM) has been stacked on 4-8 dies on top of the logic. Logic layers include up to 8 cores, wide shared bus and memory controller. The connections between dies are assumed to be provided by Through Silicon Vias (TSVs). Overall power improvements is 2-3x better than multi-core architecture without 3D stacking technology. The memory interface also is modified according to the fact that there is no need to use narrow interfaces (pin-out) and address multiplexing with the familiar two phases CAS/RAS, therefore the logic for latching and multiplexing address/data can be removed (RLDRAM and NetDRAM like). However this solution lacks in scalability, due to the usage of shared bus. In addition, since all the blocks have been modeled at high level and at a very coarse-grained abstraction, the results are not validated with accurate and technology-calibrated models.

In (15) the authors present a 3D stacked memory architecture for CMP. Four DRAM layers stacked on top of logic layer that include 4 Cores (Intel QuadCore Penryn 45nm) and L2 cache. They presented a 3D internal DRAM architecture (based on true-3D memory organization proposed by Tezzaron) to better exploit the possibilities enabled by 3D technologies. Whit these changes up to 280% speedup is achievable over the baseline 2D system with off-chip DRAMs.

Physical connectivity to DRAM is only a facet of the problem. Memory controllers manage the architectural and circuit level interface between processors and DRAM. State of-the art memory controllers (16) are still designed for narrow off-chip interfaces. They are quite complex component and they deploy many complex features to maximize exploitable interface bandwidth. The front end includes a multiport arbitration

interface and I/O queues with reordering capabilities, to improve power consumption and access latency.

The idea of distributing memory controllers to manage multiple DRAM channels is starting to make inroads, as the number of processors per chip increases. In (17) a novel scheme for memory controller placement in many-core CMP is presented. Memory controllers are attached on 2D mesh NoC nodes, therefore achievable bandwidth and numbers of on-chip memory controllers are constrained due to pin limitations. To our knowledge our work is the first to present a distributed memory controller architecture for vertically stacked memory.



(a) Conventional DRAM (b) Modified DRAM

Figure 8.2: 512MB DDR SDRAM Functional Block with standard a) and modified b) interface: In a) the data bus is bidirectional and the bus width is 4. In b) the bidirectional data bus has been replaced with two independent buses for Read and Write, and the bus width now range from 32bit to an half of the row size. CAS latency is also expected to decrease with the number of columns

8.3 3D DRAM Memory Interface

In this section we discuss the changes in the DRAM interface when moving from 2D to 3D domain. Figure 8.2 a illustrates the functional block diagram for a conventional 512MB DDR SDRAM. As shown, the data bus is both bidirectional and narrow. The former implies three-state buffers both in the DRAM and in the memory controller, while the latter implies low throughput. Moreover, since conventional DRAM is packaged separately from the processor and accessed through IO pad pins and wires on a PCB, synchronization and noise are the main concern. To tackle synchronization issues, DRAM is equipped with data strobe circuitry and DLL, since the propagation delay between memory controller and memory module is unknown at design time. Signal

integrity is an important issue that limits to reach higher frequencies on the IOs. To tackle it, the output impedance of the drivers must be matched with the characteristic impedance of the line.

In the 3D context, as mentioned in Section 8.1, the limitation on IOs count is no longer an issue. Owning to the low capacitance of the connection achieved by stacking a memory die on top of a processor die, power consumption is reduced about $24\mu W$ per pin compared to 30-40 mW per pin for DDR (18). Area cost is well limited, and since TSV process allows fine pitch interconnects, up to 10K TSVs can be fitted in a space of $1mm^2$ (19), compared to 2K ultrafine microbumps (20)(die to package), and only 11 IO's microbumps (BGA $300\mu m$ pitch).

As shown in Figure 8.2 b, theoretically, the data bus width can range from a single word, to the half width of the row. Moreover, there is no need to have a single bidirectional bus, thus write and read data are routed on separated buses. This implies that output drivers are simple buffers (three-state buffers are 3 times bigger in the same condition load) and the driven load is simply the capacitance of a TSV. The parasitic capacitance and resistance for 3D vias are negligible compared to off-chip interconnects, which are 3 order of magnitude bigger than the typical on-chip interconnect based on TSVs (19) (14).

The increasing in data bus width results in the explosion of the achievable bandwidth. Moreover the number of columns for each row is going to decrease, therefore the column address decoder becomes smaller and faster, and the CAS latency decreases. On the other hand this solution requires additional buffering resources to load the data before the movement into the prefetch buffer. Figure 8.3 shows the timing diagram for a write request in two scenarios: on top, the JEDEC compliant model and on the bottom the 3D compliant model. During the first phase, the row and bank are activated. In the second step the column is selected while the data is written in the internal write buffer at rate of 2 word per cycle (DDR), and then committed in the prefetch buffer. Finally if the row must be deactivated, the precharge command writes back the data in the bank row, and then bank is ready for a new request in a different row. The read request is similar, but during the second step, there is latency of several clock cycle before the data reaches the IO. This latency is called CAS Latency (CL) and it depends on the complexity of the column address decoder. This parameter can be tuned during the setup stage, accordingly with the target frequency. In the 3D version, Activate and

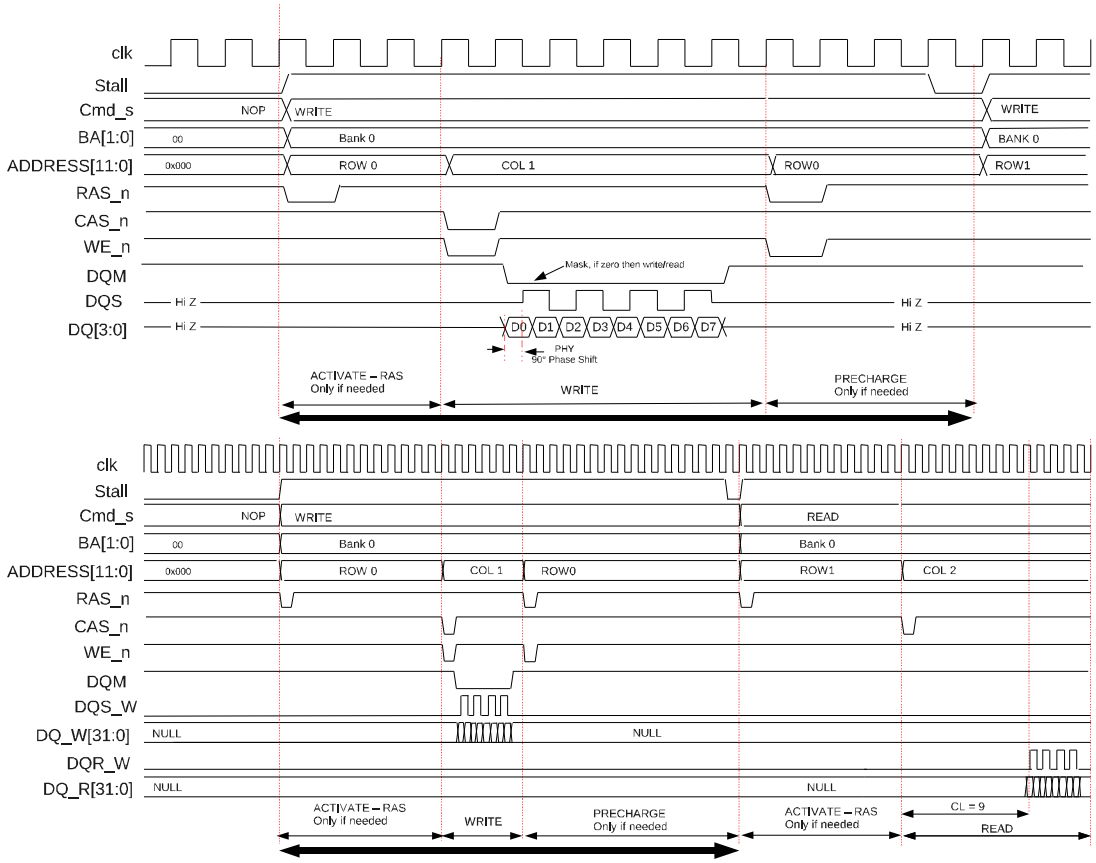


Figure 8.3: DRAM timing diagram in two scenarios: On top, the JEDEC compliant model and on the bottom the 3D compliant model. During the first phase, the row and bank are activated. In the second step the column is selected and data are written in the internal write buffer, and then committed in the prefetch buffer. Finally, if the row must be deactivated, the precharge command writes back the data in the bank row and closes the bank

Precharge show the same latency since they depend on the internal memory structure, but the R/W phase is shorter due to improvement on the IO interface and lean column decoder architecture.

In our 3D implementation, the CL decreases with the number of columns. Roughly speaking, in the conventional DRAM depicted in Figure 8.2 a the column address decoder manages 2048 columns while in our implementation (Figure 8.2 b) the columns are 256.

8.4 3D Memory System

In the Section 8.3, we discussed the improvement in latency and bandwidth achievable with the 3D Stacked DRAM. In this section we discuss an efficient memory interface and infrastructure to sustain the huge memory bandwidth available. Figure 8.4 shows the reference architecture for a multicore platform and shared DRAMs. The memory controllers are attached to the NoC, therefore every access to the DRAM must travel through the NoC. It is clear that, when the memory is off chip, the available bandwidth within the NoC is by far bigger than the memory bandwidth system. When moving to 3D memory stacking, the bottleneck is no longer the memory system, but the communication infrastructure in the logic layer, thus the reference architecture might be inefficient. A better access scheme is depicted in Figure 8.5, where local accesses are routed directly to the memory, while remote accesses are routed to the NoC. On local access, the transfer takes the advantage of the low latency and high bandwidth of the direct access, therefore an entire cache line (or simply a single word) can be written or read in few clock cycles. A remote request must be serialized as burst access, therefore increasing the access time both for the nature of the transfer and because the NoC is subject to traffic congestion.

In the following subsection we present the main blocks of the proposed architecture.

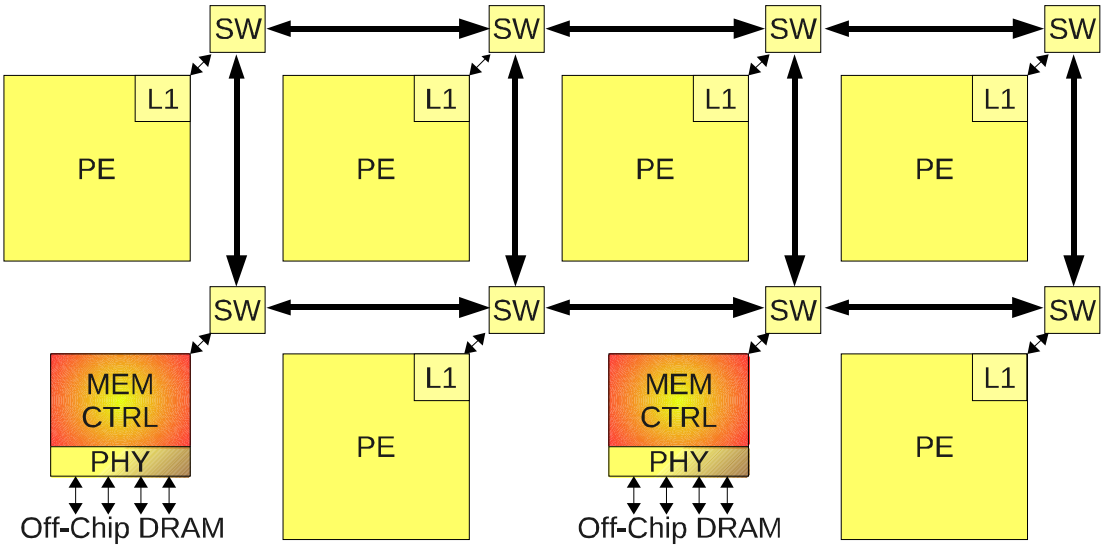


Figure 8.4: Reference architecture for multicore and shared offchip DRAMs

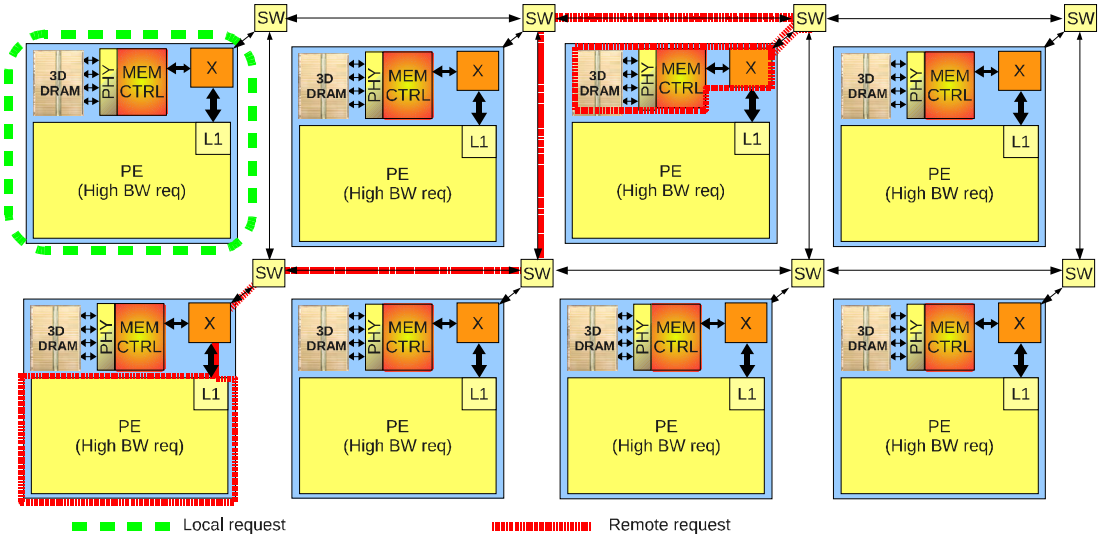


Figure 8.5: Proposed architecture for multicore and 3D Stacked memory

8.4.1 Processing Element Interface

We supposed a simple processing element interface with independent buses for read and write and with a Stall/GO flow control policy. We also assumed that the width of these buses are 256 bit (8 words) and that all the memory requests come from L1 cache misses. Since our bus provides up to 8 words, during the read or write requests, the processing element provides a data mask to inform the network interface or the DRAM controller that only few words must be read/written. We also suppose that the processing element generates outstanding transactions, therefore we developed both the memory controller and the NoC interfaces keeping in mind these important features.

8.4.2 NoC

Each processing element is connected to a 4 way custom crossbar, that allows to route incoming traffic both through the NoC (master and slave NoC port) for global request and directly to the memory controller for local accesses. Similarly, memory responses are managed in a comparable way. The custom crossbar includes a programmable built-in arbiter which supports 3 different priority policies: fixed priority, round robin and TDMA. The arbiter captures the requests and generates the grant signals both for the requests and for the responses, in order to reconfigure the crossbar in the right way.

A translation protocol layer is needed to attach the NoC to the crossbar, since the Network on chip is not capable of transferring data bigger than one word. Global

write requests are serialized as burst access with the OCP protocol, and then fed to the Network Interface (NI) initiator. Packets are collected by the NI target, and data are sent to the deserializer. Once that all the burstiness data have been collected, the deserializer forward the request for exclusive access to the memory controller (through the destination crossbar). For read operations, the request first must reach the memory controller, and then read data are serialized - deserialized through the NoC and finally delivered to the related processing element.

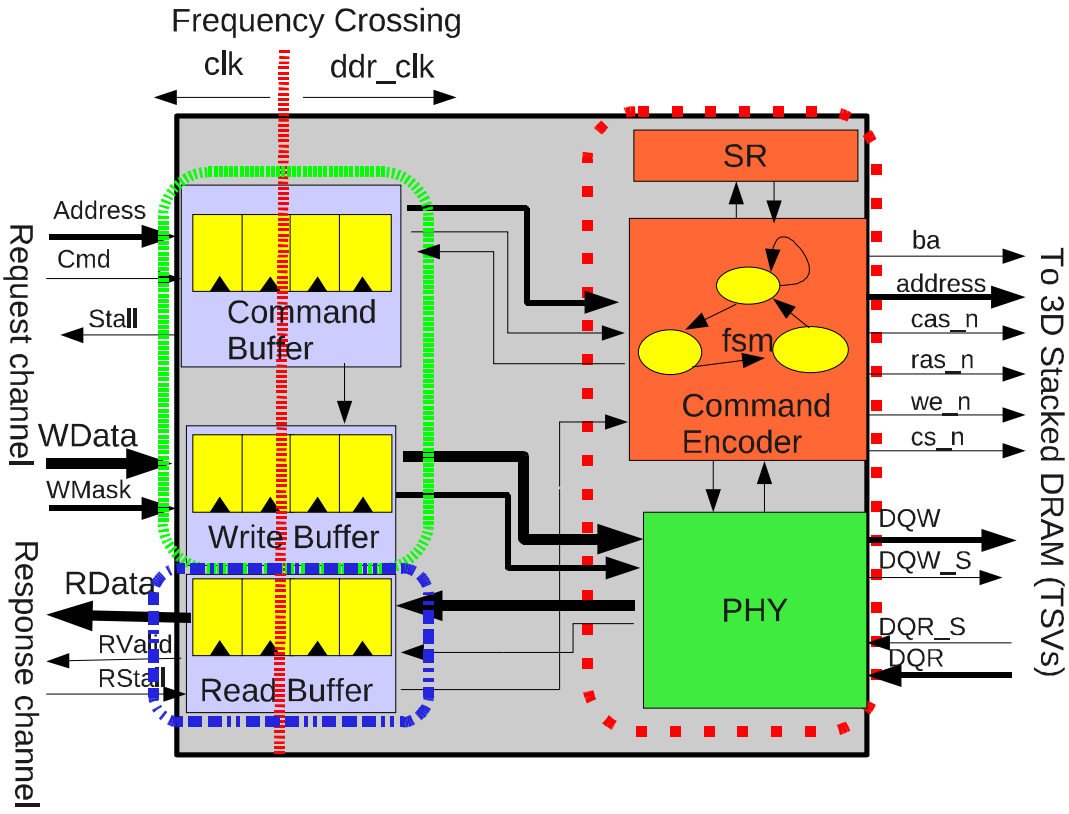


Figure 8.6: Architecture of the DDR Memory Controller

8.4.3 3D DDR Controller

The memory controller handles one of the toughest problems in complex multicore design: sustain the on-chip data bandwidth requirements of a high performance communication infrastructure. Starting from this idea, we developed a flexible memory

controller that can be tuned to achieve the desired performance. The memory controller is composed by three main blocks:

the front end includes the buffering stage for the incoming and outgoing PE signals and queues for asynchronous synchronization. The minimum depth for these buffers is 4 in case of asynchronous interface, and 2 in case of synchronous system.

the back end is composed by the DRAM command encoder and the Status Register (SR) that tracks open banks/rows and also includes counters for DRAM refresh.

the physical interface, generates the data strobes for write operation (through a DLL or delay chain), forwards the data to the DRAM at double data rate and collects the incoming chunks of the read data.

Figure 8.6 shows the architecture of the memory controller at block level. Request and response channels are independent and the implemented flow control policy is the STALL/GO. The command encoder takes the command/address from the request buffer and generates the proper signals to drive the DRAM, with the JEDEC standard or 3D interface compatibility. Since the propagation time along the TSVs is negligible as compared to delays on 2D routing, the role of the physical interface is merely to align data strobe ad data for safe data sampling in the memory controller buffers.

8.5 Experimental Results

This section provides the experimental results for the proposed memory interface and the bandwidth/latency improvement that we achieved. We first quantify the cost for remote requests across the NoC. Then we quantify the local/remote latency cost of the whole system when sweeping the frequency in two corner case: with and without memory contentions on a synchronous and asynchronous system. Finally we present the area and power cost and the impact on the whole system.

8.5.1 Timing Analysis

In a remote access, packets travel within the NoC until they reach the destination (memory controller), therefore the NoC crossing latency is strictly correlated on the topology parameters: flit width, burst length, number of hops and number of repeaters. This latency (not includes the memory access time) can be expressed in an analytic form as follow:

$$NoC\_Lat_w = N_H + N_R + BL \cdot F_P(fw) + F_H(fw) + 2 \tag{8.1}$$

$$NoC\_Lat_r = 2 \cdot N_H + 2 \cdot N_R + BL \cdot F_P(fw) + 2 \cdot F_H(fw) + 4 \tag{8.2}$$

N_H and N_R mean respectively the number of NoC hops and the number of repeaters between source and destination. BL stands for burst length (number of words to write or read), and $F_P(fw)$ and $F_H(fw)$ represent respectively the number of flits needed to encode respectively payload and header which are a function of the flit width of the NoC. The constant additive terms (2 and 4) represent the latency due serializer and deserializer pipeline stages. In equation 8.2 there is a round-trip path since the request first must reach the memory controller and then data must come back to the processing element. For that reason the read latency shows double latency for several terms of the Formula.

In case of low traffic injection (no memory conflicts) the overall latency for a remote request can be expressed as the sum of two terms: the NoC crossing latency and the memory access time, that includes the latency for buffering and the access RAS/CAS sequence. Latency for local request is simply the memory access time. Starting from the assumptions that we discussed in section 8.3 and 8.4, we simulated several configurations, sweeping both the memory and the NoC clock frequency (synchronous system). Figure 8.7 shows the latency trend for a synchronous memory system (without conflicts) in four cases: best and worst case for both local and remote accesses. The best case means that all the requests are issued within the same open rows (this features is denoted as “locality”), therefore, there is no need to precharge and activate a different row whenever a command is issued. On the other hand, the worst case is given by accessing on random rows, that results in a sequence of activations and precharges. It is clear that when there is no data locality the latencies increases quickly. To improve the system efficiency we increase the clock frequency of the whole system (as shown

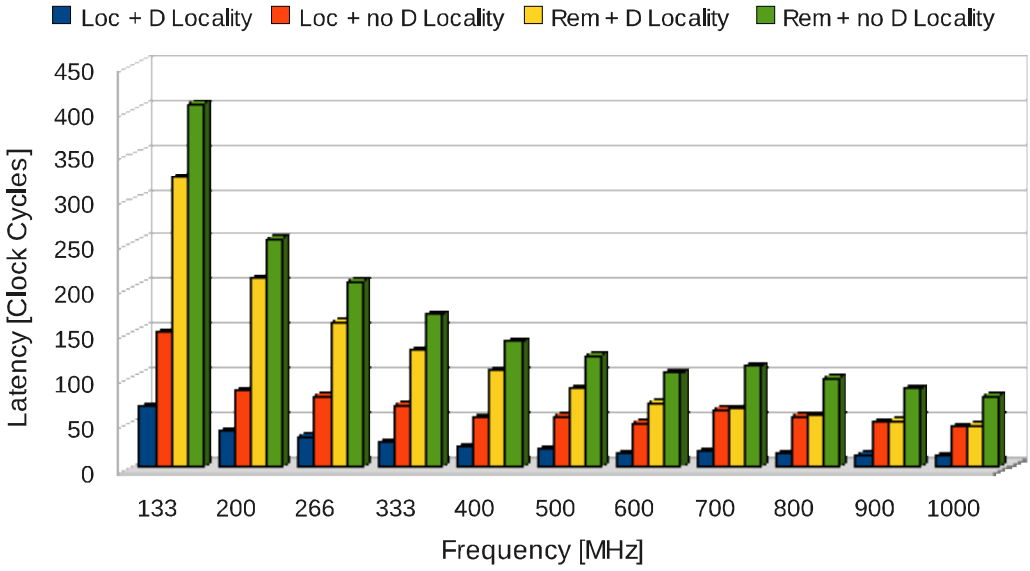


Figure 8.7: Synchronous memory system (NoC and DRAM Controller are clocked at the same frequency): max and min latency for local and remote read request, when sweeping system clock frequency (without memory conflicts). The first group of bars represent the JEDEC compliant system

in Figure 8.7) from 133 MHz to 1 GHz. The remote requests gain benefits from the overclocking, since the NoC latency is proportional to the clock speed. The memory access time on the other hand is locked on the timing parameters of the DRAM, just overclocking influences only the rate at which the data is read/written in the DRAM prefetch buffer. Activation and precharge latencies remain unchanged. We achieving a speed-up (as compared to the JEDEC standard) that ranges from 7.6× for local write request (with data locality) to 3.3× for local read request (without data locality).

Moreover, we swept the memory system clock frequency, keeping fixed the NoC frequency at 1GHz. Experimental results are shown in Figure 8.8. The speed-up ranges from 6.4× for local write request (with data locality), to 2.3× for remote request (with data locality). As shown in the figure, the latencies for remote accesses saturate for frequencies above 333MHz (NoC is dominant), while local requests latencies scale well in the whole frequency span.

The memory system bandwidth for the synchronous memory system ranges from 4.53GB/s (7.6× better than the JEDEC DDR compliant case) to 848MB/s (which

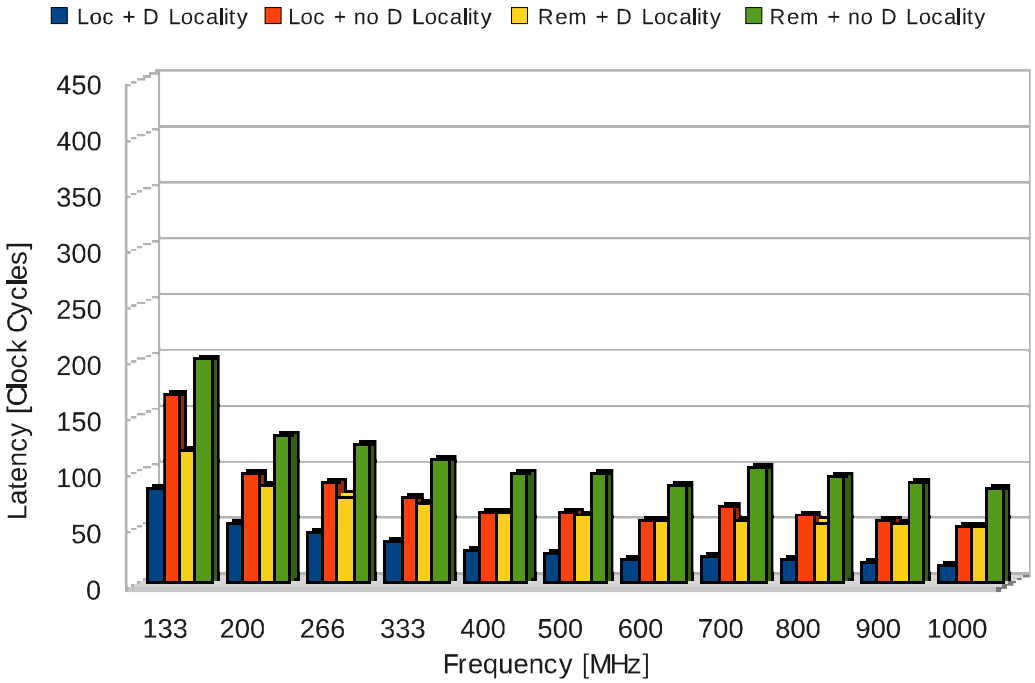


Figure 8.8: Asynchronous memory system: max and min latency for local and remote read request, when sweeping memory clock frequency (without memory conflicts). The first group of bars represent the JEDEC compliant system. NoC is Clocked at 1GHz

leads an improvement of 3.3×). The bandwidth improvement in the asynchronous case ranges from 4.53GB/s (5× better than the JEDEC DDR compliant case) to 478MB/s. (1.6×).

Finally we evaluated the bandwidth performance when the system is under high traffic injection, on the same target memory. As discussed in section 8.4, the arbiter supports three priority policies: fixed priority, round robin and TDMA. We simulated the system under different configuration and we extracted, the relative bandwidth achievable for both the local and remote accesses. Figure 8.9 shows the relative bandwidth for remote accesses, expressed in %. Round robin arbitration shows the best performance, while the TDMA is a little worse. Fixed priority, with priority set to the local accesses show the worst bandwidth result as depicted in the Figure.

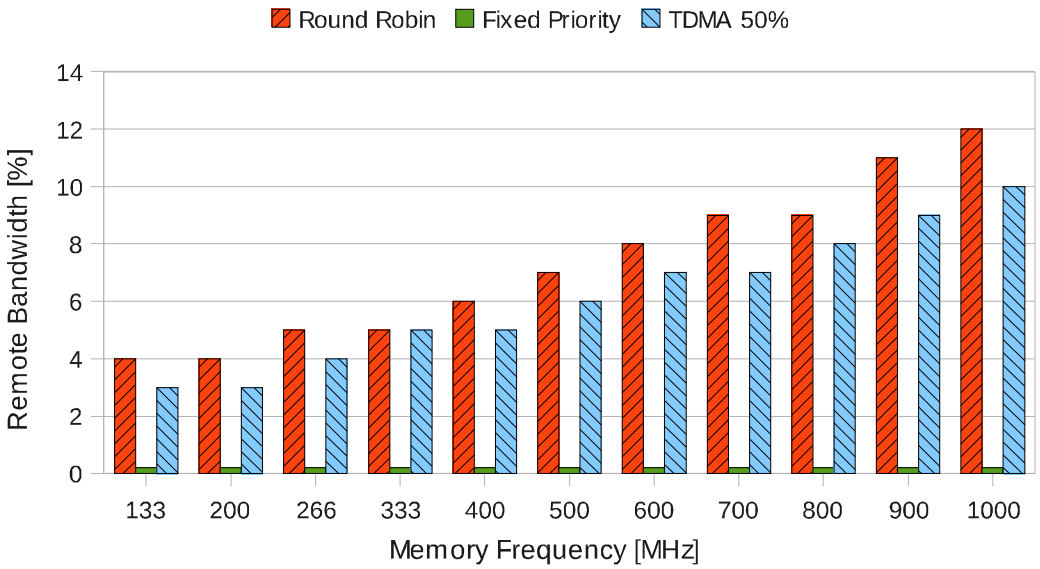


Figure 8.9: Relative bandwidth (average) for remote request, for the Asynchronous system. NoC is clocked at 1GHz

8.5.2 Physical Analysis

We synthesized the whole asynchronous platform (Mesh 2x2 NoC with 4 Memory controllers and 4 custom interfaces for low-latency local accesses) with the TSMC 65nm technology library (general purpose process). The front-end flow (Multi Vth) has been performed with Synopsys Design Compiler in topographical mode, while the back end with Cadence SoC Encounter.

Figure 8.10 (a) shows the silicon and power cost for the proposed asynchronous system. The total silicon cost is about 127K equivalent gates (NAND2) and includes the cost of NoC, data serializer and deserializer, custom crossbar-arbiter and memory controllers (PEs and DRAMs are not included in this chart). The custom crossbars (CC) require only the 5% of the total area (each is around 1.25%) while the expensive components are the memory controller, which consumes about 43% of the total area (each is around 11%). This high cost is mainly due to front end buffering (wide buses) and requires about 40% of the total area. The four TSV channels, each made by 89 TSVs (two independent buses for read and write) consumes about 8% of the silicon cost.

Figure 8.10 (b) illustrates the relative power cost of the proposed asynchronous

architecture. The total power consumption is 33mW at 1GHz for both NoC and DRAM controllers. The custom crossbar shows a negligible power consumption (about 1.25% each) while the NoC is the most power hungry block 51% due the large amount of switching activity. On the other hand, each memory controller consume only 2.5% since the clock is gated due low switching activity on the front end side. Similarly, the power consumption on the TSV drivers is around 1.25% for each link.

8.6 Conclusions

In this work, we have presented a specialized 3D-DRAM controller featuring a very fast path to the local memory neighborhood and a standard NoC communication facility for accesses to remote memory neighborhoods. We have shown hardware cost of our memory controller, and an accurate performance analysis for a system composed by four cores, a 2x2 NoC mesh and four 3D DRAM modules. Our results demonstrate that even for such a small and low-congestion system, memory access performance is greatly increased by our distributed, asymmetric memory controller with respect to a standard NoC-interfaced distributed memory controller. We have demonstrate significant bandwidth improvement that ranges from 1.44x to 7.40x as compared to the JEDEC standard, with peaks of 4.53GB/s for direct memory access, and 850MB/s for remote access through the NoC. Finally we have shown feasibility, and we quantified silicon and power cost for both the DRAM controller and the low latency interface for local accesses.

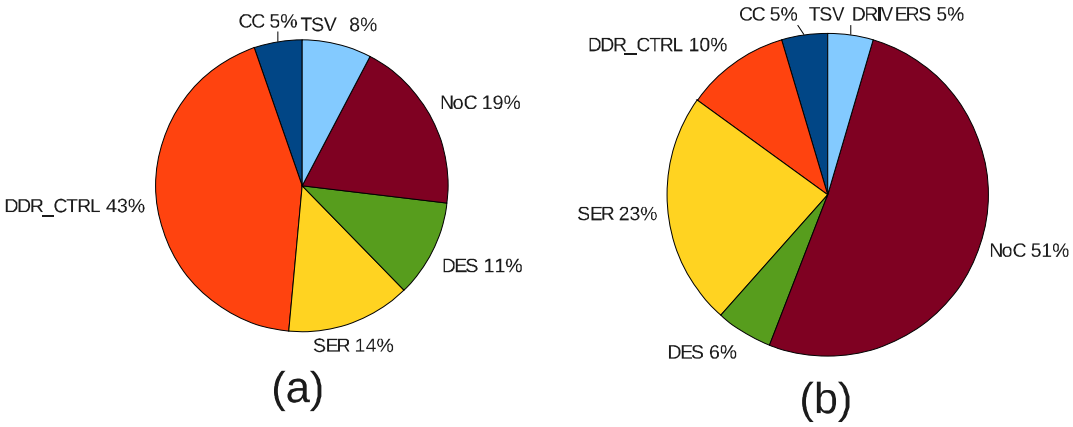


Figure 8.10: Area (a) and power (b) cost for the asynchronous platform. The design includes 4 memory controllers, 4 data serializers and deserializers 4 custom crossbars and the NoC. In (a) the NoC is assumed with a flit width of 35bit while in (b) the power estimation is done at the maximum frequency (1GHz)

Bibliography

[1] Micron Technology Inc, "Ddr sdram point-to-point simulation process," 2005, $http://download.micron.com/pdf/technotes/DDR/TN4611.pdf$. 112

[2] Micron Technology, "Ddr sdram system-power calculator," 2009, $http://www.micron.com/support/part_info/powercalc$. 112

[3] R. S. Patti, "Three-dimensional integrated circuits and the future of system-on-chip designs," *Proceedings of the IEEE*, vol. 94, no. 6, June 2006. 112, 114

[4] A. W. Topol et al, "Three-dimensional integrated circuits," *IBM Journal of Research and Development*, vol. 50, no. 4/5, pp. 491–506, July/September 2006. 112, 114

[5] B. B. et al, "Die stacking (3d) microarchitecture," in *39th International Symposium on Microarchitecture*, December 2006, pp. 469–479. 112

[6] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proceedings of the 38th Design Automation Conference*, June 2001, pp. 684–689. 112

[7] L. Benini and G. De Micheli, "Networks on chips: A new SoC paradigm," *IEEE Computer*, vol. 35, no. 1, pp. 70 – 78, January 2002. 112

[8] W. J. Dally and B. Towles, "Route packets, not wires: On-chip interconnection networks," in *Proceedings of the 38th Design Automation Conference*, June 2001, pp. 684–689. 114

[9] L. Benini and G. De Micheli, "Networks on chip: a new SoC paradigm," *IEEE Computer*, vol. 35, no. 1, pp. 70–78, January 2002. 114

[10] U. K. et al, "8gb 3d ddr3 dram using through-silicon-via technology," *2009 IEEE International Solid-State Circuit Conference*, no. 1, pp. 130 – 132, 2009. 114

[11] R. Patti, "Tezzaron semincoductor," 2006, *http* : $//www.tezzaron.com/memory/TSC_LeoI I.htm$. 114

[12] Christianto C. Liu et al, "Bridging the processor-memory performance gap with 3d ic technology," in *IEEE Design and Test of Computers*, Nov 2005, pp. 556 – 564. 115

[13] Gianluca Loi et al, "A thermally-aware performance analysis of vertically integrated (3-d) processor-memory hierarchy," in *Proceedings of the 43rd annual Design Automation Conference*, Aug 2006, pp. 991 – 996. 115

[14] Kgil Taeho et al, "Picoserver: using 3d stacking technology to enable a compact energy efficient chip multiprocessor," *Proceedings of the 2006 ASPLOS Conference*, vol. 41, no. 11, November 2006. 115, 117

[15] Gabriel H Loh et al, "3d-stacked memory architectures for multi-core processors," in *International Symposium on Computer Architecture*, June 2008, pp. 453–464. 115

[16] Denali Software Inc, "Databahn dram memory controller ip," 2009, *https* : $//www.denali.com/en/products/databahn_dram.jsp$. 115

[17] Dennis Abts et al, "Achieving predictable performance through better memory controller placement in many-core cmps," in *The 36th International Symposium on Computer Architecture*, June 2009, pp. 451–461. 116

[18] 3D-IC Alliance, "Imis - intimate memory interface specification," 2009, *http* : $//www.3d-ic.org/standards.html$. 117

[19] I. Loi, F. Angiolini, and L. Benini, "Supporting vertical links for 3d networks-on-chip: Toward an automated design and analysis flow," in *Proceedings of the Nano-Net Conference 2007*, 2007, pp. 23–27. 117

[20] Yu Aibin et al, "Development of fine pitch solder microbumps for 3d chip stacking," in *Electronics Packaging Technology Conference*, December 2008, pp. 387–392. 117

9

Conclusions

Integrated Fabless Manufacturers must innovate on system integration rather than chip design.The future of the semiconductor industry strongly depends on its agility to adapt to the changing market situation. Large OEMs are shifting their focus to more customer oriented services rather than system integration. The drive for more system integration is catalyzed by the decreasing return on investment for classical CMOS scaling. Innovative system design therefore must come from heterogeneous integration of technologies.

The main objective of this PhD research has been the exploration of high performance multicore platforms based on 3D NoCs and 3D stacked memory, and the development of a novel design flow for 3D integrated circuits. A RTL to GDSII flow have been presented, and several architectual blocks have been proposed to tackle new challenges in three-dimensional process integration. Finally, A 3DNoC test chip based on the proposed architectures/blocks have been manufactured at the IMEC labs (Leuven-Belgium)

In this thesis we have contributed tackling some of the numerous open research challenges in high performance and low-power multi-processor-system-on-chip domain.

9.1 Perspectives

Research on 3D NoCs is just now beginning, and much work remains to be done. Among the areas requiring more attention, we plan on focusing on three main areas: testing, characterization and 3D memory system exploration. The first topic will cover

the detection “at speed” of hardware faults in 3D links and the automatic link reconfiguration; the second topic will manage the 3D link characterization, in order to quantify the variability along the links (technology tuning); the third and last topic will cover the 3D system exploration for NoC based platforms with 3D stacked DRAM on top of the logic die.

10

Publications

P. Meloni, I. Loi, F. Angiolini, S. Carta, M. Barbaro, L. Raffo and L. Benini, "Area and Power Modeling for Networks-on-Chip with Layout Awareness" VLSI Design, March 2007.

I. Loi, F. Angiolini and L. Benini "Supporting vertical links for 3D networks on chip: toward an automated design and analysis flow" Proceedings of the Nano-Net Conference 2007, Catania, Italy, Sep 24-26, 2007

I.Loi, F.Angiolini and L.Benini "Developing Mesochronous Sychronizer to enable 3D NoCs", Procedings of the Date Conference, 10-14 March 2008, Munich, Germany

I.Loi, S.Mitra, T.H.Lee S.Fujita and L.Benini "A Low-overhead Fault Tolerance Scheme for TSV-based 3D Network on Chip Links", ICCAD conference, 9-13 Nov 2008, San Jose, CA, USA

I. Loi, F. Angiolini and L. Benini "Synthesis of Low-Overhead Configurable Source Routing Tables for Network Interfaces" DATE09, Nizza, France

I. Loi and L. Benini "An effcient distributed memory interface for Many-Core Platform with 3D stacked DRAM" DATE10, Dresden, Germany

I. Loi, A. Pullini, P. Marchal and L. Benini "3D NoCs - Unifying Inter & Intra chip Communication" ISCAS10, Paris, France

M.R. Kakoee, I. Loi and L. Benini "A New Physical Routing Approach for Robust Bundled Signaling on NoC Links", GVLSI10

I. Loi, F. Angiolini, S. Mitra, S. Fujita and L. Benini "Characterization and Implementation of Fault-Tolerant Vertical Links for 3D Networks-on-Chip", Under revision on TCAD journal