Amir-Mohammad Rahmani-Sane

Exploration and Design of Power-Efficient Networked Many-Core Systems

Turku Centre *for* Computer Science

TUCS Dissertations
No 153, December 2012

Exploration and Design of Power-Efficient Networked Many-Core Systems

Amir-Mohammad Rahmani-Sane

To be presented, with the permission of the Faculty of Mathematics and Natural Sciences of the University of Turku, for public criticism in Auditorium Lambda on December 14th, 2012, at 12 noon.

University of Turku
Department of Information Technology
20014 Turun yliopisto

2012

Supervisors

Professor Hannu Tenhunen
Associate Professor Juha Plosila
Associate Professor Pasi Liljeberg
Department of Information Technology
University of Turku
Turku, Finland

Reviewers

Professor José Flich Cardo
School of Engineering in Computer Science
Technical University of Valencia
Valencia, Spain

Ph.D. Dinesh Pamunuwa
Reader in Microelectronics
Department of Electrical and Electronic Engineering
University of Bristol
Bristol, UK

Opponent

Professor José L. Ayala
Facultad de Informática
Complutense University of Madrid
Madrid, Spain

ISBN 978-952-12-2817-9
ISSN 1239-1883

Abstract

Multiprocessing is a promising solution to meet the requirements of near future applications. To get full benefit from parallel processing, a many-core system needs efficient, on-chip communication architecture. Network-on-Chip (NoC) is a general purpose communication concept that offers high-throughput, reduced power consumption, and keeps complexity in check by a regular composition of basic building blocks. This thesis presents power efficient communication approaches for networked many-core systems. We address a range of issues being important for designing power-efficient many-core systems at two different levels: the network-level and the router-level.

From the network-level point of view, exploiting state-of-the-art concepts such as Globally Asynchronous Locally Synchronous (GALS), Voltage/Frequency Island (VFI), and 3D Networks-on-Chip approaches may be a solution to the excessive power consumption demanded by today's and future many-core systems. To this end, a low-cost 3D NoC architecture, based on high-speed GALS-based vertical channels, is proposed to mitigate high peak temperatures, power densities, and area footprints of vertical interconnects in 3D ICs. To further exploit the beneficial feature of a negligible inter-layer distance of 3D ICs, we propose a novel hybridization scheme for inter-layer communication. In addition, an efficient adaptive routing algorithm is presented which enables congestion-aware and reliable communication for the hybridized NoC architecture. An integrated monitoring and management platform on top of this architecture is also developed in order to implement more scalable power optimization techniques.

From the router-level perspective, four design styles for implementing power-efficient reconfigurable interfaces in VFI-based NoC systems are proposed. To enhance the utilization of virtual channel buffers and to manage their power consumption, a partial virtual channel sharing method for NoC routers is devised and implemented.

Extensive experiments with synthetic and real benchmarks show significant power savings and mitigated hotspots with similar performance compared to latest NoC architectures. The thesis concludes that careful co-designed elements from different network levels enable considerable power savings for many-core systems.

Acknowledgements

This work was carried out at the Department of Information Technology, University of Turku, during September 2009 and November 2012. This dissertation would not have been possible without the guidance and the help of several individuals who in one way or another contributed and extended their valuable assistance in the accomplishment of this research work.

First and foremost, my utmost gratitude to my supervisors Associate Prof. Pasi Liljeberg, Associate Prof. Juha Plosila, and Prof. Hannu Tenhunen for their inspiration, guidance and support. I attribute the level of my Ph.D. degree to their encouragement and effort and without them this thesis, too, would not have been completed or written. One simply could not wish for better or friendlier supervisors.

I also wish to thank Prof. José Flich Cardo from Technical University of Valencia - Spain and Ph.D. Dinesh Pamunuwa from University of Bristol - UK for their detailed reviews of this dissertation and for providing constructive comments and suggestions for improvement.

The Turku Centre for Computer Science (TUCS) Graduate School is gratefully acknowledged for funding my doctoral studies. This research work was financially supported by the Academy of Finland, the Nokia Foundation, and the Ulla Tuominen Foundation. I would like to also acknowledge the Business and Innovation Development (BID) unit at the University of Turku and the European Institute of Innovation & Technology (EIT) ICT labs for providing the joint MBA programme for doctoral students. This programme indeed gave me a thorough experience on business competence and valuable industry exposure.

I would like to thank all my colleagues and staff at the IT department and TUCS graduate school. I am grateful to everyone who has co-authored papers with me specially Khalid Latif and Kameswar Rao Vaddina for the insights they have shared. I would also like to thank my M.Sc. thesis supervisor at University of Tehran, Prof. Ali Afzali-Kusha. I am indebted to him for his help since I learned fundamentals of this research work under his supervision.

I would like to express my deepest gratitude to my parents for all the help during different phases of my life and studies. All the support they have provided me over the years was the greatest gift anyone has ever given me. My brothers and sisters also deserve thanks for their encouragement and most importantly taking care of my responsibilities in my absence at home.

Finally, my warmest and heartfelt thanks go to my wife, Sanaz, for her love and standing beside me throughout my study and writing this thesis. She has been my inspiration and motivation for continuing to improve my knowledge and move my research forward. She is my rock.

Turku, November 2012

Amir-Mohammad Rahmani

List of original publications

The work discussed in this thesis is based on the publications listed below:

Journal publications (JNL):

1. Amir-Mohammad Rahmani, Kameswar Rao Vaddina, Khalid Latif, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "High-Performance and Fault-Tolerant 3D NoC-Bus Hybrid Architecture Using ARB-NET Based Adaptive Monitoring Platform," IEEE Transactions on Computers (*IEEE-TC*), - (To appear).

2. Amir-Mohammad Rahmani, Kameswar Rao Vaddina, Khalid Latif, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "Design and Management of High Performance, Reliable, and Thermal-Aware 3D Networks-on-Chip," IET Circuits, Devices & Systems Journal (*IET-CDS*), Vol. 6, No. 5, pp. 308-321, 2012. doi: 10.1049/iet-cds.2011.0349

3. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "Developing a Power-Efficient and Low-Cost 3D NoC Using Smart GALS-Based Vertical Channels," Journal of Computer and System Sciences (*Elsevier-JCSS*), 2012. doi: 10.1016/j.jcss.2012.09.004

4. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "Design and Implementation of Reconfigurable FIFOs for Voltage/Frequency Island-Based Networks-on-Chip," Journal of Microprocessors and Microsystems (*Elsevier-MICPRO*), 2012. doi: 10.1016/j.micpro.2012.07.003

5. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "Exploring a Low-Cost and Power-Efficient Hybridization Technique for 3D NoC-Bus Hybrid Architecture using LastZ-Based Routing Algorithms," Journal of Low Power Electronics (*American Scientific Publishers - JOLPE*), Vol. 8, No. 4, pp. 403-414, 2012. doi: 10.1166/jolpe.2012.1202

6. Khalid Latif, Amir-Mohammad Rahmani, Tiberiu Seceleanu and Hannu Tenhunen, "Cluster based Networks-on-Chip: An Efficient and Fault-Tolerant Architecture using Network Interface Assisted Routing," International Journal of Adaptive, Resilient and Autonomic Systems (*IGI-Global-IJARAS*) - (To appear).

7. Khalid Latif, Amir-Mohammad Rahmani, Ethiophia Nigussie, Tiberiu Seceleanu, Martin Radetzki, and Hannu Tenhunen, "PVS: A Generic Methodology to Enhance Resource Management and Reliability of NoCs," submitted to Journal of Parallel Computing (*Elsevier-PARCO-M*).

Conference publications (CNF):

8. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "Developing Reconfigurable FIFOs to Optimize Power/Performance of Voltage/Frequency Island-Based Networks-on-Chip," in *Proc. of IEEE International Symposium on Design and Diagnostics of Electronic Circuits and Systems* (DDECS'10), pp. 105-110, 2010, Austria.

9. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "Power and Performance Optimization of Voltage/Frequency Island-Based Networks-on-Chip Using Reconfigurable Synchronous-/Bi-Synchronous FIFOs," in *Proc. of ACM International Conference on Computing Frontiers* (CF'10), pp. 267-276, 2010, Italy.

10. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "An Efficient 3D NoC Architecture Using Bidirectional Bisynchronous Vertical Channels," in *Proc. of 2010 IEEE Computer Society Annual Symposium on VLSI* (ISVLSI'10), pp. 452-453, 2010, Greece.

11. Amir-Mohammad Rahmani, Khalid Latif, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "Research and Practices on 3D Networks-on-Chip Architectures," in *Proc. of IEEE International Norchip Conference* (NORCHIP'10), 2010, Finland.

12. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "Exploring a Low-Cost Inter-layer Communication Scheme for 3D Networks-on-Chip," in *Proc. of IEEE International Symposium on Computer Architecture & Digital Systems* (CADS'10), pp. 183-186, 2010, Iran.

13. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "An Efficient VFI-Based NoC Architecture Using Johnson-Encoded Reconfigurable FIFOs," in *Proc. of IEEE International Norchip Conference* (NORCHIP'10), 2010, Finland.

14. Amir-Mohammad Rahmani, Khalid Latif, Pasi Liljeberg, Juha Plosila, Hannu Tenhunen, "A Stacked Mesh 3D NoC Architecture Enabling Congestion-Aware and Reliable Inter-Layer Communication," in *Proc. of IEEE/Euromicro International Conference on Parallel, Distributed, and Network-Based Processing* (PDP'11), pp. 423-430, 2011, Cyprus.

15. Amir-Mohammad Rahmani, Khalid Latif, Kameswar Rao Vaddina, Pasi Liljeberg, Juha Plosila, Hannu Tenhunen, "Congestion Aware, Fault Tolerant, and Thermally Efficient Inter-Layer Communication Scheme for Hybrid NoC-Bus 3D Architectures," in *Proc. of IEEE/ACM International Symposium on Networks-on-Chip* (NOCS'11), pp. 65-72, 2011, USA.

16. Khalid Latif, Amir-Mohammad Rahmani, Ethiopia Nigussie, Tiberiu Seceleanu, Hannu Tenhunen, "A Novel Topology-Independent Router Architecture to Enhance Reliability and Performance of Networks-on-Chip," in *Proc. of IEEE International Symposium on Defect and Fault Tolerance in VLSI and Nanotechnology Systems* (DFT'11), pp. 454-462, 2011, Canada.

17. Amir-Mohammad Rahmani, Kameswar Rao Vaddina, Pasi Liljeberg, Juha Plosila, Hannu Tenhunen, "Power and Area Optimization of 3D Networks-on-Chip Using Smart and Efficient Vertical Channels," in *Proc. of International Conference on Power and Timing Modeling, Optimization and Simulation* (PATMOS'11) as Lecture Notes in Computer Science (LNCS-Springer), pp. 278-287, 2011, Spain.

18. Khalid Latif, Amir-Mohammad Rahmani, Kameswar Rao Vaddina, Tiberiu Seceleanu, Pasi Liljeberg, and Hannu Tenhunen, "Enhancing Performance of NoC-Based Architectures using Heuristic Virtual-Channel Sharing Approach," in *Proc. of Real-Time, Embedded and Physical Systems track of IEEE Computer Software and Applications Conference* (COMPSAC'11), pp. 442-447, 2011, Germany.

19. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "LastZ: An Ultra Optimized 3D Networks-on-Chip Architecture," in *Proc. of IEEE/Euromicro International Conference on Digital System Design* (DSD'11), pp. 173-180, 2011, Finland.

20. Kameswar Rao Vaddina, Amir-Mohammad Rahmani, Khalid Latif, Pasi Liljeberg and Juha Plosila, "Thermal Analysis of Job Alloca-

tion and Scheduling Schemes for 3D Stacked NoC's," in *Proc. of IEEE/Euromicro International Conference on Digital System Design* (DSD'11), pp. 643-648, 2011, 2011, Finland.

21. Amir-Mohammad Rahmani, Khalid Latif, Kameswar Rao Vaddina, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "ARB-NET: A Novel Adaptive Monitoring Platform for Stacked Mesh 3D NoC Architectures," in *Proc. of IEEE/ACM International Asia and South Pacific Design Automation Conference* (ASP-DAC'12), pp. 413-418, 2012, Australia.

22. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "An Efficient Hybridization Scheme for Stacked Mesh 3D NoC Architecture", in *Proc. of IEEE/Euromicro International Conference on Parallel, Distributed, and Network-Based Processing* (PDP'-12), pp. 507-514, 2012, Germany.

23. Amir-Mohammad Rahmani, Kameswar Rao Vaddina, Khalid Latif, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "Generic Monitoring and Management Infrastructure for 3D NoC-Bus Hybrid Architectures," in *Proc. of IEEE/ACM International Symposium on Networks-on-Chip* (NOCS'12), pp. 177-184, 2012, Denmark.

24. Amir-Mohammad Rahmani, Kameswar Rao Vaddina, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "Power and Thermal Analysis of Stacked Mesh 3D NoC Using AdaptiveXYZ Routing Algorithm," in *Proc. of IEEE/Euromicro International Conference on Digital System Design* (DSD'12), pp. 208-215, 2012, Turkey.

25. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, Ka Lok Man, Youngmin Kim, and Hannu Tenhunen, "Partial-LastZ: An Optimized Hybridization Technique for 3D NoC Architecture Enabling Adaptive Inter-Layer Communication," in *Proc. of IEEE International SoC Design Conference* (ISOCC'12), pp. 281-284, 2012, South Korea.

Book chapters (BC):

26. Khalid Latif, Amir-Mohammad Rahmani, Tiberiu Seceleanu and Hannu Tenhunen, "An Autonomic NoC Architecture using Heuristic Technique for Virtual Channel Sharing," in Phan Cong-Vinh (Eds.), *Autonomic Networking-on-Chip: Bio-inspired Specification, Development, and Verification.* Part of the Embedded Multi-core Systems (EMS) Book Series, CRC Press, December 2011.

Workshop publications (WS):

27. Amir-Mohammad Rahmani, Khalid Latif, Kameswar Rao Vaddina, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "Power-Efficient Inter-Layer Communication Architectures for 3D NoC," in *Proc. of 2011 IEEE Computer Society Annual Symposium on VLSI* (ISVLSI'11-), Ph.D. Forum, pp. 355-356, 2011, India.

28. Amir-Mohammad Rahmani, Pasi Liljeberg, Juha Plosila, and Hannu Tenhunen, "High-performance, Power-Aware and Reliable 3D NoC Architectures," in *PhD Forum Booklet of IEEE/ACM International Asia and South Pacific Design Automation Conference* (ASP-DAC'12), 2012, Australia.

Contents

1 Introduction **1**
- 1.1 Scope of Thesis and Contributions . . . 3
- 1.2 Thesis Organization . . . 4

2 Networks-on-Chip **7**
- 2.1 Topology . . . 7
  - 2.1.1 Direct Topologies . . . 7
  - 2.1.2 Indirect Topologies . . . 8
  - 2.1.3 Regular Vs. Irregular Topologies . . . 9
- 2.2 Switching Strategies . . . 10
  - 2.2.1 Circuit Switching . . . 10
  - 2.2.2 Packet Switching . . . 11
- 2.3 Flow Control . . . 12
- 2.4 Virtual Channels . . . 12
- 2.5 Router Architecture . . . 13
- 2.6 Routing . . . 14
  - 2.6.1 Unicast and Multicast Routing . . . 15
  - 2.6.2 Distributed and Source Routing . . . 15
  - 2.6.3 Static and Dynamic Routing . . . 15
  - 2.6.4 Minimal and Non-Minimal Routing . . . 16
  - 2.6.5 Deadlock and Livelock . . . 16
  - 2.6.6 Starvation . . . 16
- 2.7 Clocking Techniques . . . 16
- 2.8 Networks-on-Chip Architectures . . . 17
- 2.9 Summary . . . 18

3 Research and Practices on 3D Networks-on-Chip **19**
- 3.1 Three-Dimensional Integrated Circuits . . . 19
- 3.2 Opportunities and Challenges of TSV . . . 20
  - 3.2.1 3D Design Issues . . . 20
  - 3.2.2 Benefits of Vertical Channels . . . 21
- 3.3 3D Networks-on-Chip Architectures . . . 24

3.3.1 Symmetric 3D Mesh . . . 24
3.3.2 Ciliated 3D Mesh . . . 25
3.3.3 Hybrid 3D NoC-Bus . . . 26
3.3.4 True 3D NoC . . . 26
3.3.5 Tree-Based 3D NoCs . . . 27
3.3.6 XNoTS . . . 28
3.3.7 MIRA . . . 28
3.3.8 De-Bruijn Graph-Based 3D NoC . . . 29
3.3.9 Serialized Vertical Channel Architecture . . . 29
3.3.10 Honeycomb 3D Architecture . . . 30
3.3.11 Low-Radix 3D NoC . . . 30
3.3.12 Layer-Multiplexed Mesh Architecture . . . 31
3.3.13 Special Purpose 3D NoCs . . . 32
3.4 Summary . . . 32

4 Reconfigurable Voltage/Frequency Island-Based NoC 35
4.1 Power-Aware VFI-Based Frequency-Voltage Assignment . . . 36
4.1.1 Dynamic Voltage Frequency Assignment . . . 37
4.1.2 Static Voltage Frequency Assignment . . . 40
4.2 Reconfigurable Synchronous/Bi-Synchronous FIFO . . . 40
4.2.1 Ring Counter Based RSBS FIFO . . . 41
4.2.2 Modular RSBS FIFO . . . 45
4.2.3 Gray-Encoded RSBS FIFO . . . 47
4.2.4 Johnson-Encoded RSBS FIFO . . . 49
4.2.5 Mesochronous Adaptation . . . 52
4.3 Analysis and Case Study . . . 53
4.3.1 Latency analysis . . . 54
4.3.2 Throughput analysis . . . 56
4.3.3 Area Calculation . . . 56
4.3.4 Power Consumption and Latency Evaluation of a VFI-based NoC Running MPEG-4 Decoding System . . . 57
4.4 Summary . . . 60

5 Bidirectional Bisynchronous Vertical Channels for 3D NoC 61
5.1 Motivation . . . 62
5.2 BBVC-Based 3D NoC Architecture . . . 63
5.2.1 Router Architecture . . . 64
5.2.2 Inter-layer Transmission Scheme . . . 65
5.3 Forecasting-Based Dynamic Frequency Scaling . . . 68
5.4 Experimental Results . . . 71
5.4.1 TSV Area Footprint Analysis . . . 71

5.4.2 Synthetic Traffic Analysis . . . 71
5.4.3 Videoconference Application . . . 79
5.4.4 Hardware Overhead . . . 81
5.5 Summary . . . 82

6 Efficient Hybridization Scheme for 3D NoC 83
6.1 LastZ Routing Policy . . . 83
6.1.1 LastZ Rule . . . 85
6.2 Proposed Hybridization Architecture . . . 86
6.2.1 Wrapper Architecture . . . 88
6.3 Experimental Results . . . 90
6.3.1 Synthetic Traffic Analysis . . . 91
6.3.2 Videoconference Application . . . 94
6.3.3 Area Calculation . . . 95
6.4 Discussion . . . 96
6.5 Summary . . . 100

7 Adaptive Inter-Layer Communication 101
7.1 Motivation and Contribution . . . 101
7.2 Proposed Architecture . . . 103
7.3 Inter-layer Fault Tolerant Routing . . . 106
7.4 Experimental Results . . . 109
7.4.1 Synthetic Traffic Analysis . . . 109
7.4.2 Videoconference Application . . . 112
7.5 Summary . . . 113

8 ARB-NET: A Novel Adaptive Monitoring Platform 115
8.1 Background and Related Work . . . 116
8.2 Motivation and Contribution . . . 117
8.3 ARB-NET Architecture . . . 118
8.3.1 Packet Format . . . 119
8.3.2 ARB-NET Node Architecture . . . 121
8.3.3 AdaptiveXYZ Routing Algorithm . . . 123
8.3.4 Fault Tolerant Inter-Layer Routing . . . 126
8.4 Thermal Analysis Of AdaptiveXYZ Routing . . . 127
8.4.1 Junction Temperature and Thermal Resistance for a 3D System . . . 128
8.4.2 Thermal Modeling for 3D NoC . . . 129
8.5 Experimental Results . . . 131
8.5.1 Synthetic Traffic Analysis . . . 131
8.5.2 Videoconference Application . . . 132
8.5.3 Thermal Analysis . . . 134
8.5.4 Hardware Overhead . . . 137

8.6 Summary 138

9 Partial Virtual-Channel Sharing NoC Architecture 139
9.1 Resource Management Architectures 139
9.2 Resource Utilization Analysis 140
9.2.1 Link Load Analysis with Synthetic Traffic 140
9.2.2 Application Specific Link Load Analysis 143
9.3 The Proposed Partial Virtual Channel Sharing Approach . . 145
9.3.1 The Input Controller and Buffer Allocation 146
9.3.2 The Output Controller 148
9.3.3 Comparison with Existing Architectures 148
9.4 Simulation results 149
9.4.1 Synthetic Traffic Analysis 150
9.4.2 Videoconference Application 150
9.4.3 Area Comparison 152
9.5 Trade-offs 152
9.6 Summary 153

10 Conclusions 155
10.1 Future Work 156

List of Figures

1.1 An example of a 5×5 mesh-based NoC system 2

2.1 Examples of 2D direct NoC topologies 8
2.2 Examples of indirect NoC topologies 9
2.3 Examples of irregular NoC topologies 10
2.4 Typical virtual channel input port architecture 13
2.5 Typical virtual channel router architecture [100] 14

3.1 Schematic representation of a group of TSVs 21
3.2 Symmetric 3D Mesh NoC architecture 25
3.3 Ciliated 3D Mesh NoC architecture 26
3.4 Hybrid 3D NoC-Bus mesh architecture 27
3.5 NoC architecture with true 3D crossbars [78] 28
3.6 De-Bruijn graph-based 3D NoC architecture 29
3.7 Honeycomb 3D NoC architecture 31
3.8 Low-Radix 3D NoC architecture 31

4.1 Frequency level of the nodes in a sample 2D Mesh network with 3 VFIs for two consequent periods 38
4.2 Input channel module architecture with support of Reconfigurable Syn/Bi-Syn FIFOs 39
4.3 Reconfigurable Syn/Bi-Syn FIFO style #1 architecture 42
4.4 Full and empty detector details of the design style #1 43
4.5 Reconfigurable Syn/Bi-Syn FIFO style #2 architecture 45
4.6 RSBS FIFO put control block 46
4.7 Cycle synchronizer with asynchronous set [113] 47
4.8 Reconfigurable Syn/Bi-Syn FIFO style #3 architecture 48
4.9 FIFO wptr & full block diagram 49
4.10 Reconfigurable Syn/Bi-Syn FIFO style #4 architecture 50
4.11 4-bit Johnson encoding implementation 51
4.12 FIFO wptr & full block diagram 52
4.13 Mesochronous adaptation for the design style #1 53
4.14 Mesochronous adaptation for the design style #3 54
4.15 Latency analysis of conventional bi-synchronous FIFOs 55

4.16 Latency analysis of the RSBS FIFO 55
4.17 Mapping of MPEG-4 decoding system on a mesh NoC: Average bandwidths between two nodes are presented. (MB/s) . 58
4.18 Average power savings for the NoC switches used in MPEG Encoder 59
4.19 Average latency improvement for the NoC-based system running MPEG Encoder 59

5.1 Example of a 3D NoC with three 3×3 layers 63
5.2 Example of the transmission process of BBVC-based communication scheme 64
5.3 Up and Down ports of the proposed router architecture supporting bidirectional bisynchronous vertical channels 66
5.4 Example of physical interface between routers 66
5.5 Inter-layer data transmission scheme 67
5.6 The hardware diagram of the forecasting-based DFS policy . 70
5.7 Percentage of area saving with BBVC-based communication scheme for different TSV pitches and link sizes for 65nm, 90nm, and 130nm technologies 72
5.8 Forecasting sum square error as a function of α 74
5.9 Latency versus average packet arrival rate results 75
5.10 Average power consumption versus average packet arrival rate results 76
5.11 % performance overhead versus average packet arrival rate results 77
5.12 Bandwidth utilization versus average packet arrival rate results 78
5.13 Communication trace of encoding part of a H.264 video conference 80
5.14 Partition and core mapping of the H.264 encoder 81

6.1 High level overview of the Hybrid 3D NoC-Bus mesh router of a pillar node [88] 84
6.2 Side view of the conventional Hybrid 3D NoC-Bus mesh architecture 85
6.3 Side view of the proposed Hybrid 3D NoC-Bus mesh architecture 87
6.4 High level overview of the proposed Hybrid 3D NoC-Bus mesh router of a pillar node 88
6.5 Proposed 5×6 router architecture 89
6.6 Example of a 2×1 credit-based wrapper 89
6.7 Finite state machine for the proposed wrapper controller . . . 90
6.8 Latency versus average packet arrival rate on a 3×3×3 mesh using XYZ routing without virtual channel 92

6.9 Latency versus average packet arrival rate on a 3×3×4 mesh using (DyXY)Z routing with virtual channels 93
6.10 Latency versus average packet arrival rate on a 6×6×3 mesh under uniform traffic profile 94
6.11 Partition and mapping of the video conference encoding application 95
6.12 3D Mesh NoC with TSV Squeezing [91] 99
6.13 Top view of a TSV sharing region using the proposed hybridization scheme 100

7.1 Side view of the 3D NoC with the dTDMA bus 102
7.2 Example of *AdaptiveZ* routing 103
7.3 VC architecture for the stacked mesh architecture 106
7.4 Side view of the proposed stacked mesh architecture 107
7.5 Example of modifications to the proposed routing algorithm to guarantee single bus-link failure tolerance 108
7.6 Latency versus average packet arrival rate for a 3×3×3 NoC . 110
7.7 Latency versus average packet arrival rate for a 3×3×4 NoC . 111
7.8 3D NoC running the video conference encoding application with one faulty bus 112

8.1 ARB-NET-based Hybrid 3D NoC-Bus mesh architecture . . . 119
8.2 Packet format supporting ARB-NET monitoring platform . . 120
8.3 ARB-NET node architecture 121
8.4 Cross-Sectional view of a modern 3D Flip-Chip package . . . 128
8.5 Average throughput for different α values 132
8.6 Latency versus average packet arrival rate on a 3×3×4 mesh for different Hybrid 3D NoC-Bus mesh architectures 133
8.7 Steady-state grid level thermal maps for the die 1 (layer 0) . 136

9.1 Traffic load analysis for XY-routing 142
9.2 MPEG4 application [77] 144
9.3 Proposed PVS approach for conventional Virtual Channel Architecture 147
9.4 Average packet latency (APL) vs. Packet injection rate for 5×5 Mesh 2D NoC with (2, 2, 1) combination of PVS approach 151

List of Tables

4.1 4-bit Johnson encoding 51
4.2 Minimum FIFO depth in function of the clock relation and required throughput 56
4.3 Area and overhead comparison between the RSBS and RMBS design styles and their baseline designs (buffer length × buffer width) 57
4.4 Total number of synchronization components 57

5.1 Power consumption and average packet latency running H.264 traces 81
5.2 Hardware implementation details 81

6.1 Power Consumption and Average Packet Latency 96
6.2 Hardware implementation details 96

7.1 Power Consumption and Average Packet Latency 113

8.1 $X_{dir}Y_{dir}$ Lookup Table 120
8.2 Power Consumption and Average Packet Latency 135
8.3 Layer temperature profile of the Hybrid 3D NoC-Bus Mesh based system running the video conference application 135
8.4 Layer temperature profile of the proposed Hybrid 3D NoC-Bus Mesh-based system running the video conference application (*AdaptiveXYZ* routing) 135
8.5 Hardware Implementation Details 137

9.1 Comparison with existing NoC router architectures 149
9.2 Experimental result for average power consumption and APL of video conference encoder application 152
9.3 PVS-NoC router silicon area for different grouping combinations 153

List of Abbreviations

2D Two-Dimensional

3D Three-Dimensional

ACK Acknowledgement

APAR Average Packet Arrival Rates

APL Average Packet Latency

AR AND Read pointer

ARB-NET Arbiter Network

ATM Asynchronous Transfer Mode

BBVC Bidirectional Bisynchronous Vertical Channels

BFT Butterfly Fat Tree

BiNoC Bidirectional Network-on-Chip

BOP Begin-of-Packet

CMOS Complementary Metal-Oxide-Semiconductor

CMP Chip Multiprocessors

CRC Cyclic Redundancy Check

DARPA Defense Advanced Research Projects Agency

DFS Dynamic Frequency Scaling

DMesh Diagonally-Linked Mesh

DOR Dimension-Order Routing

DPM Dynamic Power Management

DSB Distributed Shared Buffer

DSP Digital Signal Processor

dTDMA Dynamic Time Division Multiple Access

DVFA Dynamic Voltage Frequency Assignment

DVFS Dynamic Voltage and Frequency Scaling

DyXY Dynamic XY

EDA Electronic Design Automation

EDXY Enhanced Dynamic XY

EOP End-of-Packet

FCBGA Flip-Chip Ball Grid Array

FFVL Fully Faulty Vertical Link

FIFO First In, First Out

flit flow control unit

FLOPS Floating-Point Operations per Second

FSM Finite State Machine

FVS Fully Virtual Channel Sharing

GALS Globally Asynchronous Locally Synchronous

GPLVT General-Purpose, Low-Voltage Threshold

HAM Hierarchical Agent Monitoring

HDL Hardware Description Language

HLP Higher Level Protocols

IB Input Buffer

IBM International Business Machines

IC Input Controller

IC Integrated Circuit

ICG Integrated Clock Gating

ID Identifier

IFC Input Flow Controller

IL Inter-Layer

ILM Inter-Layer Material

IMEC Interuniversity Microelectronics Centre

IP Intellectual Property

IRS Input Read Switch

LM Layer-Multiplexed

LPLVT Low-Power, Low-Voltage Threshold

LU Link Utilization

MI Monitoring Information

MIRA Multi-layered on-chip Interconnect Router Architecture

MPEG Moving Picture Experts Group

MPSoC Multi-Processor System-on-Chip

MSB Most Significant Bit

MV Motion Vector

NACK Negative Acknowledgement

NED Negative Exponential Distribution

NePA Networked Processor Array

NI Network Interfaces

nMOS n-type Metal-Oxide-Semiconductor

NoC Network-on-Chip

NUCA Non-Uniform Cache Architecture

OB Output Buffer

OFDM Orthogonal Frequency-Division Multiplexing

OTP Open Telecom Platform

PC Personal Computer

PDA Personal Digital Assistant

PDP Power-Delay Product

PE Processing Elements

PFL Partially-Faulty Link

pMOS p-type Metal-Oxide-Semiconductor

PVS Partial Virtual-Channel Sharing

QoS Quality of Service

RAM Random-Access Memory

RASoC Router Architecture for SoC

RI Routing Information

RILM Reconfigurable Inter-Layer Routing Mechanism

RMBS Reconfigurable Mesochronous/Bi-Synchronous

RSBS Reconfigurable Synchronous/Bi-Synchronous

RTL Register-Transfer Level

SCC Single-chip Cloud Computer

SMS Short Messages

SNSW Synchronized/Not Synchronized Write pointer

SNUCA Static Non-Uniform Cache Architecture

SoC System-on-Chip

SSE Sum Square Error

SVFA Static Voltage Frequency Assignment

SW Synchronized Write pointer

TLL Total Link Load

TLM Thermal Interface Material

TNL Total Network Link Load

TSV Through Silicon Via

UBS Unified Buffered Structure

VC Virtual Channel

VFI Voltage/Frequency Island

ViChaR Virtual Channel Regulator

VL Via Last

VLSI Very Large Scale Integration

VM Via Middle

XNoTs Xbar-connected Network-on-Tiers

Chapter 1

Introduction

Following Moore's law, on-chip transistor densities have increased steadily, enabling the integration of many cores on a single die. This includes regular arrays of processors and cache banks in tiled chip multiprocessors (CMPs) and heterogeneous resources in system-on-chip (SoC) designs. For example, the Intel SCC [97] incorporates 48 cores, the Tilera TILE-Gx family processor has 9 to 100 identical cores [152] and the Intel Polaris chip contains 80 cores [160]. Integrating all of these computational resources requires efficient communication resources [70]. Scaling of the CMOS feature sizes into nano-scale regime makes the interconnect delay and power consumption critical parameters in the optimization of the many-core systems. The interconnect optimization in this regime is a challenging task due to the increase in crosstalk and electromagnetic interference effects [70][10].

Networks-on-Chip was proposed to cope with the mentioned issues by scaling down the concepts of macro- and tele-networks, and applying them to the system-on-chip domain [12][59]. Instead of connecting top-level modules (i.e. processing elements) by routing dedicated wires, NoCs use packets to route data from the source to the destination component, via an on-chip network that is composed of a number of routers and interconnection links. Replacing global wires with an on-chip network offers advantages of structure, modularity, better performance and power consumption and can scale efficiently as the number of processing elements in a system increases. An example of a 3×3 mesh-based NoC system is illustrated in Fig. 1.1. In general, a NoC system contains four fundamental components: processing elements (PEs), network interfaces (NIs), routers, and links. A PE can be any component such as a microprocessor, digital signal processor (DSP), memory, custom logic, or a peripheral controller. A network interface (NI) at the boundary of each PE implements the interface by which PEs connect to the NoC. Their function is to packetize any data generated by the PE. Routers send the data from the input buffers to the appropriate output link

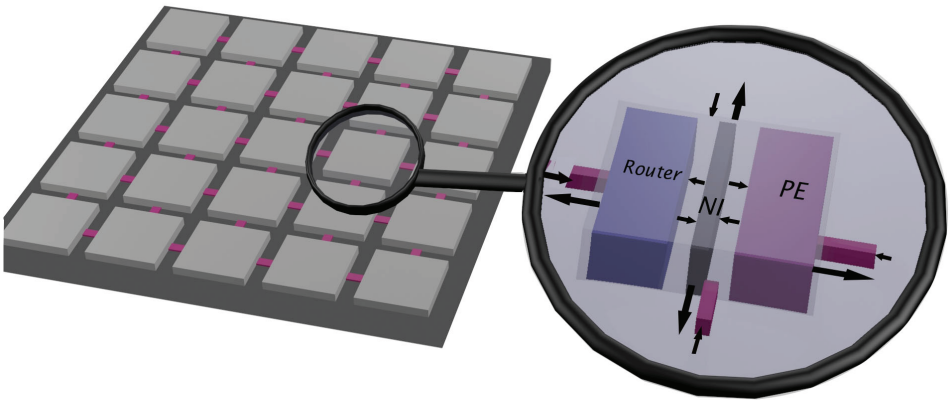


Figure 1.1: An example of a 5×5 mesh-based NoC system

according to chosen routing strategy. While moving from the source to the destination, a packet traverses multiple links which consist of one or more logical or physical channels.

The advent of three-dimensional (3D) stacked technologies provides new avenues for on-chip interconnect design. 3D ICs, which contain multiple layers of active devices, have the potential for enhancing system power and performance characteristics [47][154][44][45][66]. 3D ICs allow for performance enhancements even in the absence of scaling because of the reduced length of interconnects [154][124]. Besides this clear benefit, the package density is increased significantly, power is reduced due to shorter wires, and circuitry is more immune to noise [154]. The power and performance improvement arising from the architectural advantages of NoCs will be significantly enhanced if 3D ICs are adopted. The amalgamation of two emerging paradigms, NoC and 3D IC, allows for the creation of new structures that enable significant power and performance enhancements over more traditional solutions. With freedom in the third dimension, architectures that were prohibitive due to wiring constraints in 2D ICs are now possible, and many 3D implementations can outperform their 2D counterparts [47].

Even though both 3D integrated circuits and NoCs are proposed as alternatives for the interconnect scaling demands, there are several challenges in combining both approaches to design three-dimensional NoCs. There are few commercially available EDA tools and design methodologies, high peak temperatures, increased power densities and large area footprints of vertical interconnects. To address these issues and get the most benefit out of 3D chip stacks in multiprocessor systems, new architectures and design methods are needed.

Recently, the NoC-based computing systems are increasingly becoming power-constrained. Power consumption has become a primary issue in em-

bedded systems as well as enterprise and desktop environments. Today's PDAs, laptops and other mobile devices require considerable processing power while keeping the battery life unchanged. On the other hand, there is a large demand of power for server blades, storage bricks, and PCs due to cooling and packaging costs [164]. With an increasing number of cores and demand for interconnect bandwidth, on-chip networks have become a major source of power consumption. In addition, a rapid increase in chip operating frequencies further increases the amount of dynamic power dissipated by the on-chip communication network. For example, the 16-tile MIT Raw on-chip network [149] consumes 36% of total chip power, with each router dissipating 40% of individual tile power [164]. Another example is the TeraFLOPS [160] in which the network consumes up to 39% of the total chip power (76 W when operating at 5.1 GHz) [159]. Therefore, more emphasis should be put on networked multi-core system design that reduce the power consumption of the on-chip network. The objective of this thesis is to develop novel power-efficient communication platforms for achieving maximum power reduction and optimization for networked many-core systems.

1.1 Scope of Thesis and Contributions

This thesis addresses the challenges in the area of NoC power optimization at the different network layers. Different novel techniques to reduce or manage the excessive power consumption of the on-chip communication infrastructure of many-core systems are developed and integrated. Since achieving power efficiency has become an increasingly difficult challenge in fully synchronous 2D many-core systems, especially in the presence of increasing die sizes and high clock frequencies, the focus of this thesis is on design and optimization of NoC-based communication platform using the globally asynchronous locally synchronous and 3D NoC concepts. In the following, the author's contributions are summarized. We use JNL, CNF, BC, and WS prefixes to refer to the related journal publications, conference publications, book chapters, and workshop publications, respectively. These papers have been indexed in the list of original publications section.

- *Power-efficient link-level on-chip communication*: Different scalable and synthesizable reconfigurable FIFOs for voltage-frequency islands based systems are developed. These FIFOs can adapt their operations to either synchronous or bi-synchronous mode. We also present a mesochronous adaptation method for the proposed Reconfigurable FIFOs [JNL-4, CNF-8, CNF-9, CNF-13].

- *Power-efficient inter-layer communication for 3D NoC*: We utilize a dynamically self-configurable bidirectional bisynchronous vertical

channels to mitigate high peak temperatures, power densities and area footprints of vertical interconnects in 3D NoC structures. In addition, based on the GALS implementation approach of the proposed channels, a forecasting-based dynamic frequency scaling technique for reducing the power consumption of the inter-layer communication is introduced [JNL-3, CNF-10, CNF-12, CNF-17].

- *Efficient hybridization scheme for Hybrid 3D NoC-Bus architectures*: The hybridization mechanism benefiting from our proposed routing rule called *LastZ* enables low-cost inter-layer communication architecture. In the proposed architecture, instead of using the conventional 6×6 routers, 5×6 routers were utilized which offers many advantages [JNL-5, CNF-19, CNF-22, CNF-25].

- *Proposing adaptive routing algorithms for 3D NoC*: The contribution includes developing an efficient 3D NoC architecture to optimize performance, power consumption, and reliability of Hybrid 3D NoC-Bus Mesh system. The mechanism benefits from a congestion-aware and bus failure tolerant routing algorithm, which we call *AdaptiveZ*, for vertical communication [JNL-2, CNF-14, CNF-15, WS-27].

- *Autonomic monitoring and management platform*: We develop a monitoring platform on top of the Hybrid 3D NoC-Bus mesh architecture which can be efficiently used for various system management purposes such as traffic monitoring, fault tolerance, and thermal management. As a test case, based on the proposed monitoring platform, a fully congestion-aware and inter-layer fault tolerant routing algorithm named *AdaptiveXYZ* is presented [JNL-1, CNF-20, CNF-21, CNF-23, CNF-24, WS-28].

- *Low-power and high-performance flow control techniques*: A novel technique aiming for ideal performance and power consumption trade-off, partial virtual channel sharing NoC, is presented. We propose that sharing the virtual channel buffers among multiple inputs, provides a performance advantage to conventional virtual channel based routers with minimized overhead. We demonstrate quantitatively the benefits of proposed architecture by comparing to the state-of-the-art NoC routers, with various traffic patterns [JNL-6, JNL-7, CNF-16, CNF-18, BC-26].

1.2 Thesis Organization

The thesis is organized as follows. In Chapter 2, the concept of Networks-on-Chip is introduced and its main design characteristics are elaborated.

Chapter 3 identifies general trends and explains a range of issues which are important for designing efficient 3D NoC architectures. In addition, this chapter highlights the significance of difference network characteristics in on-chip networks, and presents different existing 3D on-chip network architectures. In Chapter 4, we present four design styles for implementing reconfigurable synchronous/bi-synchronous FIFOs. Exploration of a mechanism using Bidirectional Bisynchronous Vertical Channels to reduce TSV area footprint and optimize 3D NoC power consumption is presented in Chapter 5. In Chapter 6, we propose a novel hybridization scheme for inter-layer communication to enhance the overall system power, performance, and area characteristics. Chapter 7 addresses the routing issues and buffer utilization to improve the overall system power efficiency, performance, and reliability of existing stacked mesh architectures. An integrated low-cost monitoring and management platform for 3D stacked mesh architectures is proposed in Chapter 8. In Chapter 9, we present partial virtual-channel sharing NoC architecture with an ideal tradeoff among VC utilization, system performance, and power consumption. Finally, Chapter 10 concludes the thesis and presents future direction of this research.

Chapter 2

Networks-on-Chip

Networks-on-Chip shares the same characteristics in network topology, switching strategy, routing algorithm, and flow control policy with large scale networks. The main difference is that NoC architectures are supposed to offer high and predictable performance while constricted to limited area and power budget. In this chapter, we elaborate on the NoC characteristics and overview a few examples of NoC architectures that have been presented in literature.

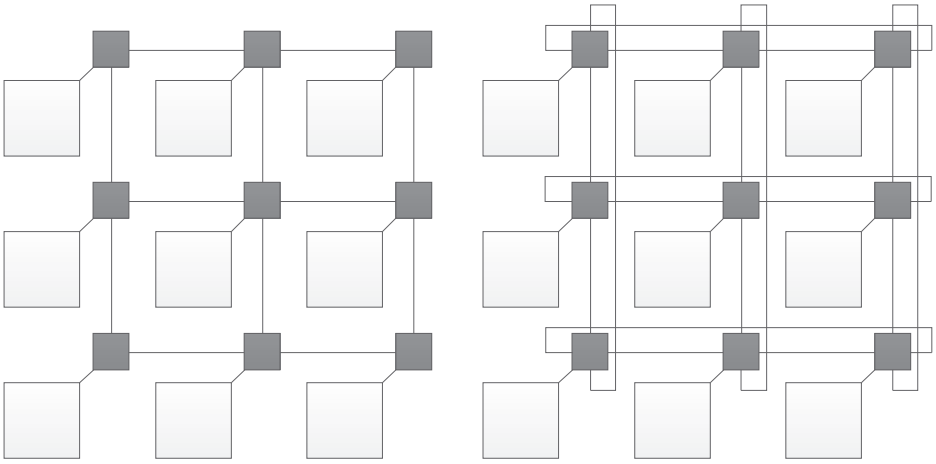
2.1 Topology

The physical organization of an on-chip network is defined mainly by its topology. Topology concerns the connectivity of nodes, switches, and links. In NoCs, it can be classified into four main divisions: direct, indirect, regular, and irregular networks [21].

2.1.1 Direct Topologies

In a direct network topology, each node has a point-to-point, or direct, link to some number of other (neighboring) nodes. In this topology, as the number of nodes in the system increases, the total available communication bandwidth increases as well. This offers a desirable scalability to implement massively parallel computers. However, an essential tradeoff between the offered bandwidth and area of routers should be taken into account. Higher bandwidth results in lower latency, but increases the energy and area costs for the router and link implementations.

As a result of N-dimensional orthogonal characteristic of most of the direct topologies, routing mechanism can be implemented simply and cost-effective in terms of area footprint. The N-dimensional mesh and torus topologies are two well-known direct topologies. Fig. 2.1(a) shows a 3×3 2D mesh topology. The reason for the popularity of the 2D mesh topology is



(a) Mesh (b) Torus

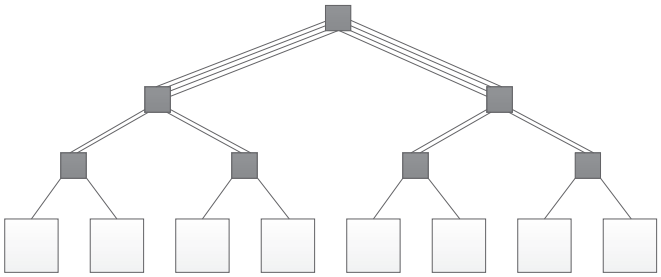
Figure 2.1: Examples of 2D direct NoC topologies

that it benefits from links all having the same length, which can be laid out on a chip surface. Avoiding congestion in the center of the network in the mesh topology should be considered to increase the system performance.

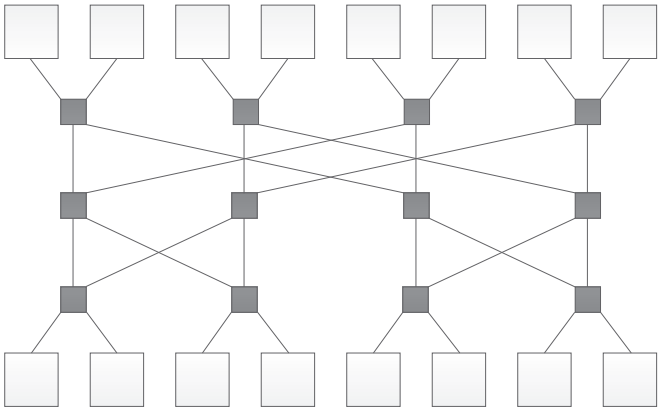
The torus topology is similar to the regular Mesh topology. The only difference is that the nodes at the edges are connected to the nodes at the opposite edge via long links [122]. An example of 2D torus topology is shown in Fig. 2.1(b). The torus topology takes advantage of higher bandwidth, however, the long wrap-around link can lead to excessive delays due to needing repeater insertion. These long links also increase the complexity of layout process.

2.1.2 Indirect Topologies

In indirect network topologies, switches have point-to-point connections to other switches. More precisely, nodes (PEs) are connected to external switches and perform data processing. On the other hand, intermediate switches just handle the packet switching. Examples of popular multi-stage indirect networks are shown in Fig. 2.2 [40]. Fig. 2.2(a) shows an example of the fat-tree topology with 8 nodes [57][87]. In the fat-tree topology, PEs are connected to the leaves of the tree. Since for a typical tree topology, root and its neighbors have higher traffic, in the fat-tree topology, links among adjacent switches are increased as they get closer to the root. This approach can cope with the bandwidth limitation of the simple-tree topology. The fat-tree topology has lower diameter as compared to a mesh and benefits from efficient VLSI layout [2]. Butterfly topology is another popular indirect



(a) Fat-tree



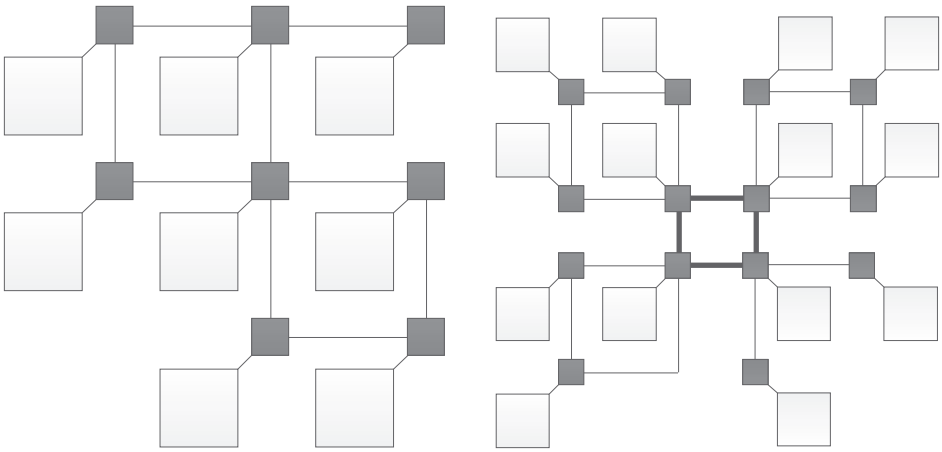
(b) Butterfly

Figure 2.2: Examples of indirect NoC topologies

topology. Generally, a network using k-ary, n-fly butterfly topology comprises of k^n nodes and n stages of crossbar switches. A 2-ary, 3-fly butterfly topology is illustrated in Fig. 2.2(b). In this figure, despite that the source and destination nodes are shown separately, they are the same. One characteristic of the butterfly topology is in the case of contention when packets may be temporarily blocked due to the blocking multi-stage attribute of this topology [122].

2.1.3 Regular Vs. Irregular Topologies

A regular network topology is formed in terms of well-defined regular graph structures such as rings, meshes, hypercubes, etc. However, irregular topologies are defined when there is no regularity in the topology and can be derived by mixing different topologies such as shared bus, direct, and indirect network topologies in an asymmetric way. These topologies scale nonlinearly with regards to area and power and are typically customized for specific applications. The main goal of irregular topologies is to reduce



(a) Optimized mesh (b) Hybrid

Figure 2.3: Examples of irregular NoC topologies

network power consumption and cost while maintaining the available bandwidth. The reduced mesh topology shown in Fig. 2.3(a) is an example of irregular topologies. As can be seen from the figure, a number of routers and links are removed based on the target application. The hybrid topology is another example of an irregular topology. The hybrid topology is the combination of other topologies as shown in Fig. 2.3(b). Hybrid topologies are usually based on the concept of clustering [12].

2.2 Switching Strategies

The switching mechanism determines how and when messages cross the routers in the network. In NoCs, messages that are divided into packets are generated by the nodes. A packet can be further partitioned into a number of flits (flow control unit). Different flit, packet, and message sizes can be chosen for the network. A proper sizing mechanism can considerably improve the system cost, performance, and power consumption. Circuit switching and packet switching are two main forms of switching mechanism as described below.

2.2.1 Circuit Switching

In circuit switching, a physical path between a source and a destination is setup and reserved until the transport of data is complete. Circuit switching provides strong guarantees at the cost of dimensioned connections for the worst case traffic. Circuit switching suffers from the poor scalability as the

size of the network grows as well as long initialization time for connection setup. The reason is that several links should be reserved for the whole data transmission process (even when no data is being transmitted) [34].

2.2.2 Packet Switching

In contrast with the circuit switching in which establishing a path before transmission process is done, packet switching benefits from a per-hop basis to forward traffic. In this mechanism, packets comprise of routing information, as well as data, and independently traverse their path towards their destinations possibly along different routes. Packets can also have variable delay depending on the traffic load in the network. Packet switching improves the bandwidth utilization, transmission latency, and communication robustness of the NoC [10]. There are three well-known and commonly-used buffered packet switching schemes: store and forward, virtual cut through, and wormhole switching as described below:

Store and forward

Store-and-forward is a simple switching protocol in which a node stores the complete packet and forwards it only when it has been received in its entirety based on the information within its header. The buffer size in the router is at least equal to the size of a packet otherwise the packet has to be stalled until the router in the forwarding path provides sufficient buffer space. This switching technique necessitates large buffer size making it less attractive for NoC architectures.

Virtual cut through

In this technique, just after the reception of the message header, if its selected outgoing channel is free (space for the entire packet is available in the next router), the message is forwarded to the adjacent node towards its destination. Following the header flit, the other flits (payload) are sent out without any delay. This technique decreases network latency and increases the network throughput because it does not wait for the entire packet to be received. In addition, if the packet stalls, it will stay in the current node and will not block any link. Since the buffer requirement of this technique is the same as the store and forward switching, it is not commonly used in NoC architectures.

Wormhole

Wormhole switching takes advantage of the concept of circuit switching mixed with the packet switching strategy to reduce the packet latency over

store and forward and virtual cut through switching. In this technique, the node processes the header of the packet to determine its next hop and immediately forwards it. The other flits follow the header flit as they arrive. If there is insufficient space in the next router to store the entire packet, the packet can worm its way through the network, possibly distributing among a number of routers. A stalling packet, however, has the unpleasantly expensive side effect of blocking the intermediate links occupied by a worm and susceptibility to deadlocks. Nevertheless, it is the most attractive and cost-effective switching mechanism for the NoC architectures.

2.3 Flow Control

In on-chip networks, flow control is defined as a policy that determines how the packet moves along the network path [126]. Flow control policy plays a significant role in enhancing the system performance and reliability by managing buffers and links allocated to packets. It can be discussed as both global and local perspectives. It can be also viewed as a problem of ensuring correct operation of the network-on-chip by considering optimal network resource utilization and performance predicability of communication services. At the data link-layer level, flow control policy also provides error recovery when transmission errors occur. In this layer, there are various popular flow control schemes for NoCs offering different reliability features in terms of power, performance, and area overhead, such as STALL/GO, T-Error, credit-based, and ACK/NACK [148]. In the network and transport layer, the flow control policy is implemented to handle flows between senders and receivers. More precisely, flow control guarantees that the generated packets by a sender can be accepted by a receiver without any contention. The network and transport layers' flow control techniques are classified based on their need for resource allocation [122].

2.4 Virtual Channels

As discussed before, wormhole flow control has been proposed to reduce the buffer requirements and enhance the system throughput. However, one packet may occupy several intermediate switches at the same time. This introduces the problem of deadlocks [10]. To avoid this problem, the use of a virtual channel is introduced. A typical virtual channel architecture [101] for input port of a router is shown in Figure 2.4 [85]. Virtual channel flow control exploits an array of buffers at each input port. By allocating different packets to each of these buffers, flits from multiple packets may be sent in an interleaved manner over a single physical channel. This improves the throughput and reduces the average packet latency by allowing blocked

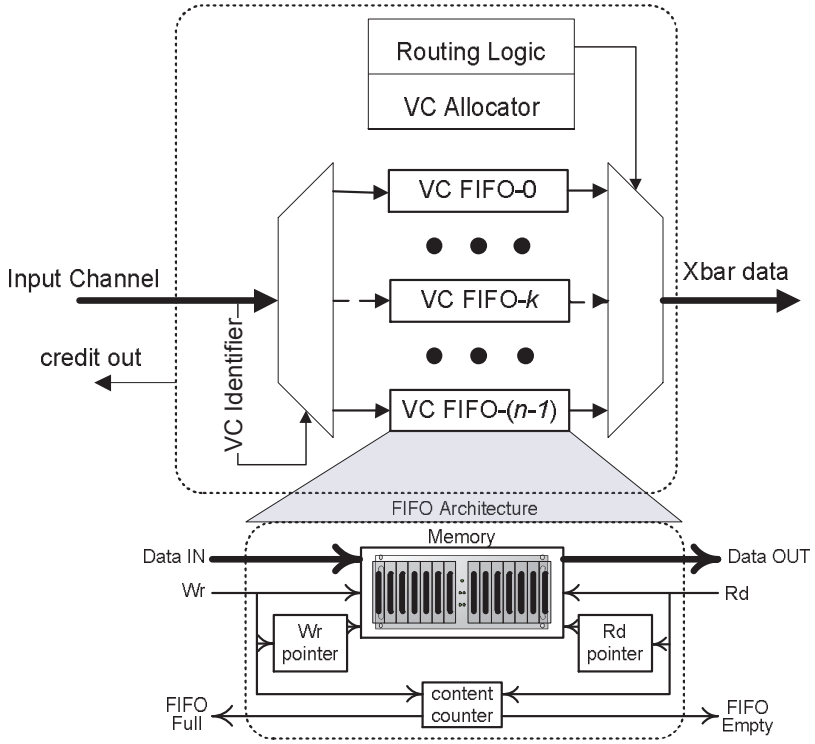


Figure 2.4: Typical virtual channel input port architecture

packets to be bypassed.

In general, virtual channels offer many advantages such as avoiding deadlocks by breaking cycles in the resource dependency graph [38], optimizing wire utilization by letting several logical channels share the physical wires [158], improving performance by minimizing the frequency of stalls [37], and providing quality-of-service (QoS) by assigning different priorities to different data streams [48].

2.5 Router Architecture

In this section, a typical router architecture for Networks-on-Chip using virtual channels is briefly presented. The router consists of crossbar switch, buffers and routing logic as shown in Figure 2.5. If a flit needs be stalled to get an output channel access, it will be stored in an input buffer. Virtual channels provide multiple input buffers for each physical input port so that flits can move as if there are multiple virtual channels. When a flit is ready to be forwarded, the crossbar switch links an input buffer to a selected output port. Routing logic, virtual channel (VC) allocator, and switch allocator govern the router data path by deciding the direction of the next router, the

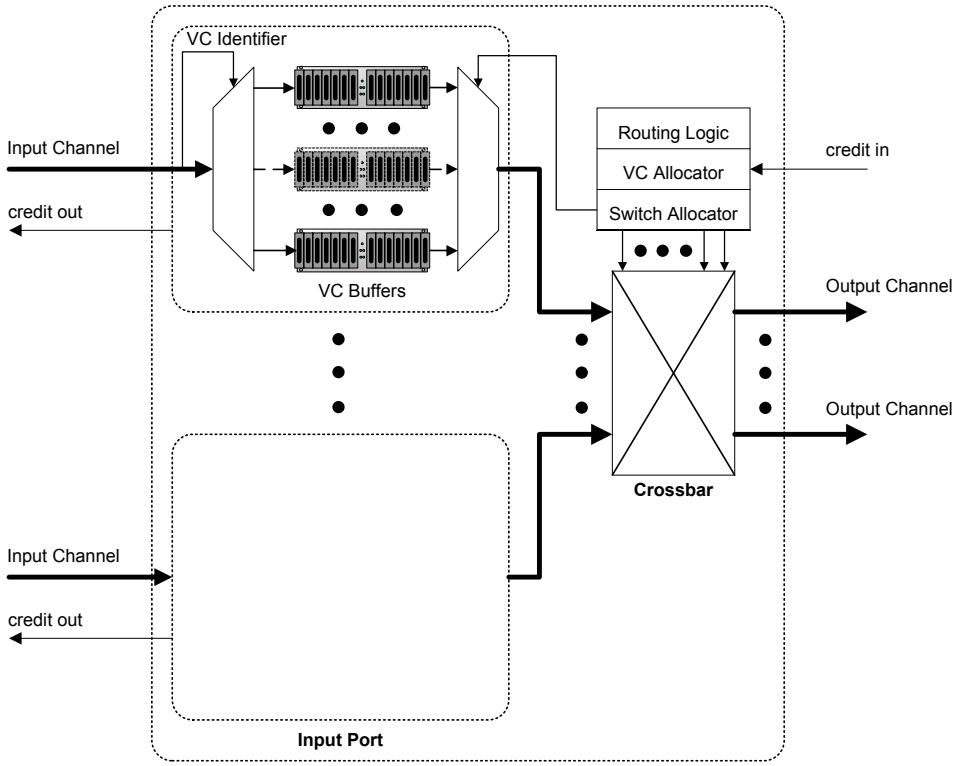


Figure 2.5: Typical virtual channel router architecture [100]

output virtual channel, and availability of the switch for each flit. When a head flit arrives at an input channel, the router stores it in the buffer for its corresponding virtual channel and the routing field is examined by the routing computation module to determine the appropriate output port. Then the VC allocator designates an output virtual channel for the packet. After that the header flit competes to get access to the switch and move to the determined output port. When the output channel is available the flits will traverse to the output channel towards the next hop.

2.6 Routing

A routing algorithm has the purpose of correctly and efficiently routing packets or circuits, from where it was injected into the network, the source node, up to its destination node. Before presenting different routing algorithms, we present the taxonomy for routing algorithms. Efficient routing of messages is critical to the performance, power consumption, area footprint, and robustness of NoC-based systems. Routing algorithms can be categorized into several classes such as unicast or multicast routing, distributed

or source routing, static or dynamic routing, and minimal or non-minimal routing. The taxonomy for routing algorithms are presented below [46]:

2.6.1 Unicast and Multicast Routing

A routing algorithm is called unicast when it has a single destination. Multicast (one-to-many) is defined as delivering of the same message from a source node to an arbitrary number of destination nodes.

2.6.2 Distributed and Source Routing

Routing can be classified depending on the location of routing information and routing decision logic. In source routing, the routing path is determined at the source node (before the packet is injected) using pre-computed routing tables which are stored at the associated network interface. Therefore, routing information is inserted to the header of packet and no look-up table or routing logic is needed in the intermediate nodes. The routing path may also be determined in each router while the packet travels from router to router by looking up the destination addresses in a routing table or by executing a hardware function. In such cases, the routing is distributed. More precisely, a routing logic/table at each input port determines the appropriate output port for packets based on the destination address.

2.6.3 Static and Dynamic Routing

Based on the adaptivity of the routing decision making, routing policy in an NoC router can be static (deterministic) or dynamic (adaptive). In static routing, the same path is chosen every time a packet must be routed between a particular source and destination. It is suitable for on-chip networks where network traffic is predictable and relatively simple since the current state of the network in terms of traffic load, faulty link, temperature is not considered. Static routings provide in-order-delivery of packets and low-cost router implementation for NoC architectures. In contrast, dynamic routing, as the name suggests, uses information about the network state to dynamically discover routes in case of path changes as traffic conditions and requirements of the application change. The provided adaptivity results in more efficient traffic distribution in a network at the cost of additional run-time monitoring and management logic. This routing is desirable for networks in which traffic conditions show irregularity and unpredictability. A hybrid between deterministic and adaptive routing has been also presented which is called oblivious routing [40]. In oblivious routing, information about the network state is not utilized and the same routing path may not be chosen every time. Instead, it benefits from stochastic mechanisms such as a random algorithm to generate routing path.

2.6.4 Minimal and Non-Minimal Routing

Based on the length of the routing paths, the routing may be minimal or non-minimal. A minimal routing always generates the shortest possible path between the source and destination nodes. Non-minimal means that the paths may be longer than the minimum path which can be useful for avoiding congestion and critical (faulty or hotspot) regions. Non-minimal paths can result in overhead of power consumption in NoC architectures.

2.6.5 Deadlock and Livelock

A deadlock occurs when one or more packets in a network wait for an event that cannot occur and remain blocked for an indefinite amount of time. For instance, when no message can be forwarded towards its destination because one path is blocked leading to other being blocked in a cyclic fashion. Deadlock can be avoided by analyzing and breaking cyclic dependencies in the dependency graph of the shared network resources. Breaking cyclic dependencies can be done by applying routing restrictions or using additional hardware resources such as virtual channels [105].

Livelock occurs when routing a packet never leads to its destination and resources constantly change state waiting for the other to finish. In adaptive non-minimal routing, livelock can be seen because of its flexible capability of re-directing routes characteristic. Non-minimal routing grants a packet to choose alternative longer paths to reach its destination depending on the current network traffic condition, resulting in this packet aimlessly wandering in the system without ever entering the destination node. Applying priority rules can simply avoid livelock conditions.

2.6.6 Starvation

Starvation refers to a case when a packet never reaches its destination because it has low priority in the packet prioritization process. More precisely, it occurs when high-priority packets overdemand the network resources. A fair routing algorithm or allocating part of the resources for low-priority packets are solution to starvation problems [10].

2.7 Clocking Techniques

The trend of scaling down to smaller geometric dimensions in deep submicron technologies has made the clock distribution tree a crucial component of modern synchronous digital system design. In these systems, the clock distribution tree consumes a significant amount of the total power of a chip. Distributing a skew-free synchronous clock over the whole chip has also become a challenging problem in today's many-core CMPs [70].

In networks-on-chip platforms different possible clocking techniques can be used. Synchronous, mesochronous, pleisochronous, and asynchronous are the most popular clocking schemes in NoC design. A fully synchronous system needs to have clock signal connected to every flip-flop and latch all over the chip. As mentioned, it suffers from the clock skew and is difficult to achieve in practice. To cope with this problem, mesochronous clocking schemes can be used in which nodes receive clock signals having the same clock frequency but different phases. This technique can resolve the clock skew and is suitable for regular NoC topologies with deterministic phase difference. However, for irregular NoC architectures, synchronizers are needed to manage the non-deterministic phase difference. Pleisochronous is another clocking scheme which provides locally generated clock signal for each clock domain. The goal is to produce clock signals with almost the same frequency. Since the clocks are generated separately, the receiver node should handle the variable phase margin leading to a large implementation overhead. In the asynchronous clocking scheme, a clock signal is not needed to be present at all. Regions communicate with each other using some asynchronous protocols such as two- or four-phase handshaking. In asynchronous designs, metastability avoidance in the sampling latches must be taken into account. The Globally-Asynchronous Locally-Synchronous (GALS) [24][104] paradigm is an extension of this clocking scheme, in which different subsystems use unrelated clocks and communicate asynchronously. The use of GALS architectures provides the chance to completely decouple timing constraints of different modules and to connect them without considering metastability issues. It also enables the modules in the system to be upgraded simply without reassembling the whole system.

2.8 Networks-on-Chip Architectures

In recent years, the interest in NoC has increased and several architectures have been presented using different schemes for switching, routing, interfacing, clock distribution, and flow control. There are various architectures presented in literature which are suitable for different types of applications. The most poplar NoC architectures are Æthereal [52], Butterfly Fat Tree (BFT) [118], CHAIN [6], CLICHÉ [80], 2D Torus [39], HERMES [114], MANGO [12], Nostrum [81], Octagon [74], Proteo [145], QNoC [16], SPIN [16], SOCBUS [165], and Xpipes [36]. These architectures provides abundant choices for designers to implement efficient NoC-based platforms in terms of performance, power consumption, reliability, and hardware cost. This is by no means a complete compilation of existing NoCs, since there are many other recently proposed NoC architectures.

2.9 Summary

In this chapter, we have given an overview of the network-on-chip platforms. We discussed different aspects in NoC design, such as the network topology, switching techniques, routing algorithms, flow control mechanisms, and clocking schemes. The efficiency of virtual channels in avoiding deadlocks, enhancing network throughput, and providing different services was presented and a typical VC-based router architecture for NoC was illustrated. The most popular NoC architectures were listed offering various design choices of NoC implementations.

Chapter 3

Research and Practices on 3D Networks-on-Chip

In the last few years, there have been many efforts in interconnection network design for 3D stacked CMPs. The purpose of this chapter is to clarify the 3D NoC concept and to map the scientific efforts made in the area of architectural and topological optimizations of NoCs. We identify general trends and explain a range of issues which are important for designing efficient 3D NoC architectures. Moreover, this chapter highlights the significance of different network characteristics (e.g. power dissipation, thermal issues, etc) in on-chip networks, and brings insights and comparisons of different existing architectures in the area of 3D on-chip network design.

3.1 Three-Dimensional Integrated Circuits

Recently, there have been many efforts to design and fabricate 3D ICs for different applications. The designs include a NAND Flash using double stack S3 technology from Samsung [73], 3D-OTP memory from Matrix [89], the DARPA 3D IC project [27] from MIT Lincoln Laboratories, and also IBM [11] and Tezzaron [123] recent projects. These 3D integrated circuits have emerged to overcome the limitations of interconnect scaling [49] by stacking active silicon layers [14][75]. Compared with traditional 2D design, 3D ICs offer a number of advantages such as shorter global interconnects, higher performance, lower interconnect power consumption due to wire-length reduction, higher packing density and smaller footprint, and support for the implementation of mixed-technology chips [155][127][1]. In this context, several 3D designs have been recently presented. This section gives a brief introduction to possible technologies.

In a 3D chip, multiple device layers are stacked together [42][98]. Several vertical interconnect technologies have been explored including wire

bonding, microbump, contactless (capacitive or inductive), and through-silicon-via (TSV) vertical interconnects [44]. Among several 3D integration technologies, TSV is the most promising one, hence the majority of 3D integration R&D activities concentrate on this approach [49].

Utilizing 3D design offers increased bandwidth [92] and a reduced length of average interconnection wire [72], which results in considerable savings in overall power consumption. It has been also demonstrated that 3D design can be utilized to improve reliability [95]. However, the adoption of a 3D integration technology faces the challenges of increased chip temperature due to increased power density compared to a planar 2D design [32]. In this context, several techniques have been proposed for 3D architectures such as physical design optimization through intelligent placement [53], increasing thermal conductivity of the stack through insertion of thermal vias [32], and use of novel cooling structures [41]. In addition, a recent work reveals that in placement of the processing cores in a 3D chip, the areal power density is a significant design constraint [65]. Consequently, thermal concerns can be managed as long as components with high power density are not stacked on top of each other. To prevent severe thermal problems, architectures that stack memory on top of processor cores, or those that rely on low-power processor cores, have been developed [13]. It should be noted that increased temperatures increase wire resistances, and consequently the interconnect delays [19].

3.2 Opportunities and Challenges of TSV

3D NoCs are viewed as promising solutions to the bandwidth bottlenecks in communication and complexity of interconnecting an ever-growing number of cores, memories and peripherals. However, the development of 3D NoCs is still at an early stage because of a number of challenges such as thermal, area, and power density issues.

3.2.1 3D Design Issues

TSVs in die stacking and wafer stacking technologies are one of the most important design components. 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 groups or buses of such wires. As an instance, Fig. 3.1 shows a simple configuration composed of four TSVs placed in a 2×2 grid structure where L and W are respectively length and width of each TSV and t_{ox} is thickness of the oxide coating around each pillar. For a fixed chip area, as the number of stacked layers increases, the power density within the volume could also increase, thus raising the temperature of 3D ICs. One solution to relieve temperature is to exploit thermal TSVs [32][166]. However, it

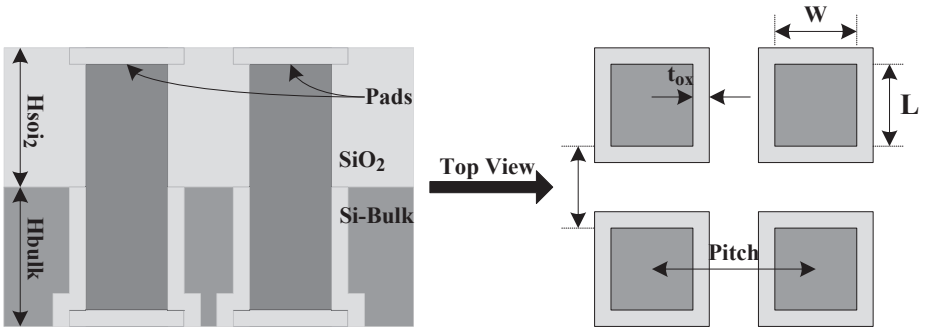


Figure 3.1: Schematic representation of a group of TSVs

is noteworthy to mention that they take up more area than normal TSVs. For example, consider a 3D NoC-based CMP with hundred links of 64-bit vertical TSV between layers. In this case, the TSVs will take up an area of approximately equivalent to the size of a computation core. Furthermore, regarding the fact that these interconnect TSVs are spread out in each layer, they will have a consequential negative impact on floorplanning and routing. The need to use thermal TSVs will make this scenario even more critical. On the other hand, it is shown in [161] that based on IMEC 3D cost model, TSV processing cost is the dominating cost for a 3D wafer. Assuming a CMOS processing technology of $65nm$ with $200mm$ silicon wafers, 46% to 65% of the costs are spent on TSV processing with annual production volume varying from 10,000 to 300,000 wafers respectively. Moreover, it is demonstrated that the manufacturing cost raises significantly beyond 1,000 TSVs per chip stack for via last (VL), while it raises significantly beyond 10,000 TSVs for via middle (VM) [161]. Obviously, in terms of chip area and manufacturing cost, the number of TSVs should be limited to a practical value.

3.2.2 Benefits of Vertical Channels

The major advantage of 3D ICs is the considerable reduction in the length and number of global interconnects, resulting in increased performance and decreased power consumption and area of wire limited systems. To clarify this significant advantage let us first compare the zero-load latency model of vertical and horizontal interconnects in terms of wire length and propagation delay. According to [40], the zero-load latency of an NoC with wormhole switching is

$$T_{Network} = hops \cdot t_r + t_c + \frac{L_p}{b} \tag{3.1}$$

where t_c is the propagation delay along the wires of the communication

channel, *hops* is the average number of switches that a packet traverses from a source to a destination node, t_r is the router delay, L_p is the packet length in bits, and b is the communication channel bandwidth. In this case we are interested in t_c which is [125]

$$t_c = t_v \cdot hops_{3D} + t_h \cdot hops_{2D} \tag{3.2}$$

where t_v and t_h are the delay of the vertical and horizontal channels, respectively, $hops_{2D}$ is the average number of hops within X and Y dimensions, and $hops_{3D}$ is the average number of hops within the third dimension. To describe t_v and t_h, the models presented in [140] can be used. According to these models, for a step input, the propagation delay and rise time of a single interconnect stage, respectively, can be described as

$$t_{di} = 0.377\frac{r_i c_i {l_i}^2}{{k_i}^2} + 0.693(R_{d0}C_0 + \frac{R_{d0}c_i l_i}{h_i k_i} + \frac{r_i l_i C_{g0} h_i}{k_i}) \tag{3.3}$$

$$t_{ri} = 1.1\frac{r_i c_i {l_i}^2}{{k_i}^2} + 2.75(R_{r0}C_0 + \frac{R_{r0}c_i l_i}{h_i k_i} + \frac{r_i l_i C_{g0} h_i}{k_i}) \tag{3.4}$$

where r_i *(c_i)* is the per unit interconnect length resistance (capacitance) and l_i is the total interconnect length. The index i represents the interconnect type (v or h). h_i and k_i respectively indicate the number and size of the repeaters and C_{g0} and C_0 denote the gate and total input capacitance of a minimum sized device, respectively. Equivalent output resistances for the propagation delay and transition time of a minimum sized inverter are respectively denoted by R_{r0} and R_{d0}, and hence the overall interconnect delay can be characterized as [125]

$$t_i = k_i(t_{di} + \gamma t_{ri}) \tag{3.5}$$

where coefficient γ is described in [140] and index i denotes the interconnect type (v or h).

For the vertical interconnects, repeaters are not necessary due to the short length, therefore in (3.3) and (3.4) $k_v = 1$ and $h_v = 1$, meaning that no repeaters are inserted. Additionally, the length of the vertical communication channel (l_v) for the 3D NoC is defined as the length of a TSV connecting two switches on adjacent physical planes. In contrast, horizontal interconnects need a number of repeaters and they are much longer than their vertical counterparts (assumed to be equal to square root of connected processing element area). For instance, in [47], the reported wire length for inter-layer and horizontal interconnects are $20\mu m$ and $2.5mm$, respectively, and consequently their delays respectively are $16ps$ and $219ps$.

As discussed in Chapter 1, power dissipation is a critical issue in 3D circuits. In 3D systems, since the global interconnects are shorter [119][64],

total power consumption is anticipated to be lower compared with the mainstream 2D systems. However, the increased power density is a challenging issue for this novel design paradigm. Therefore, those 3D NoC topologies that offer low power characteristics should be of significant interest.

In [124], a brief discussion on the different power consumption components of an interconnect line with repeaters has been presented, which is adopted from [26].

According to Pavlidis *et al.*, the power consumption components of an interconnect line with repeaters are as follows.

1. *Dynamic power consumption* is the dissipated power due to the charge and discharge of the interconnect and input gate capacitance during a signal transition, and can be described by

$$P_{di} = a_s f (c_i l_i + h_i k_i C_0) V_{dd}^2 \tag{3.6}$$

   where f is the clock frequency, a_s is the switching factor and h_i and k_i denote the number and size of repeaters, respectively, C_0 represents the total input capacitance of a minimum sized device, c_i is the per unit length capacitance of the interconnect and l_i is the total length of the interconnect [8].

2. *Short-circuit power* is due to the DC current path that exists in a CMOS circuit during a signal transition when the input signal voltage changes between V_{tn} and $V_{dd} + V_{tp}$. The power consumption is modeled by

$$P_{si} = \frac{4 a_s f I_{d0}^2 t_{ri}^2 V_{dd} k_i h_i^2}{V_{dsat} G C_{effi} + 2 H I_{d0} t_{ri} h_i} \tag{3.7}$$

   where t_{ri} is the delay of the switch, I_{d0} is the average drain current of the nMOS and pMOS devices operating in the saturation region and the value of the coefficients G and H are described in [109]. Due to resistive shielding of the interconnect capacitance, an effective capacitance (C_{effi}) is used which is determined from the methodology described in [109], [110].

3. *Leakage power* is comprised of two power components, the subthreshold and gate leakage currents and can be described as

$$P_{li} = h_i k_i V_{dd} (I_{sub0} + I_{g0}) \tag{3.8}$$

   where the average subthreshold I_{sub0} and gate I_{g0} leakage current of the nMOS and pMOS transistors is used in this equation. The total

power consumption with delay constraint T_0 for a single line of a switch P_{stotal}, horizontal bus P_{htotal}, and vertical bus P_{vtotal} is, respectively,

$$P_{stotal}(T_0 - t_a) = P_{ds} + P_{ss} + P_{ls} \tag{3.9}$$

$$P_{htotal}(T_0) = P_{dh} + P_{sh} + P_{lh} \tag{3.10}$$

$$P_{vtotal}(T_0) = P_{dv} + P_{sv} + P_{lv} \tag{3.11}$$

Consequently, the minimum power consumption per bit between a source destination node pair in a NoC with a delay constraint is

$$P_{bit} = hops P_{stotal} + hops_{2D} P_{htotal} + hops_{3D} P_{vtotal} \tag{3.12}$$

This reveals that the choice of network topology can significantly reduce the power consumption. It is now clear that a strong correlation exists between power consumption and the number of hops from source to destination as well as the channel length. Since 3D NoC architectures offer reduced average number of hops traversed by a packet as well as short vertical links, they enable low power consumption.

3.3 3D Networks-on-Chip Architectures

Recently, various 3D NoC architectures have been proposed which have different impacts on design metrics. This part is concerned with the existing architectures for 3D NoC design and reviews their influences on different characteristics of the system.

3.3.1 Symmetric 3D Mesh

In order to integrate many nodes into a 3D chip, the simplest approach is to group the nodes into multiple layers and basically stack them on top of each other. Fig. 3.2 shows an example of this architecture with 3 layers stacked together, each with 9 nodes, totaling 27 nodes. We call this architecture a Symmetric 3D Mesh NoC, since both intra- and inter-layer movement bear identical characteristics: hop-by-hop traversal. Despite of simplicity, this architecture has two major inherent drawbacks. Firstly, it does not exploit the beneficial attribute of a negligible inter-wafer distance (around 50 μm per layer) in 3D chips [78]. Since traveling in the vertical dimension is multi-hop, it takes as much time as moving within each layer. In this architecture, inter-layer and intra-layer hops are indistinguishable. However,

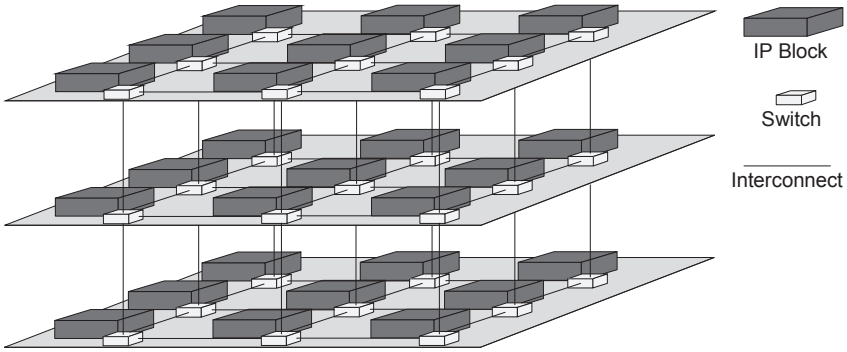


Figure 3.2: Symmetric 3D Mesh NoC architecture

the average number of hops between a source and a destination decreases as a result of reduced network diameter. In addition, buffering and arbitration delay of each flit at every hop add to the overall delay in moving within the layers. Secondly, a larger 7×7 crossbar is required as a result of two extra ports. According to [78], crossbars scale upward very inefficiently and it can be seen from their reported results that compared with 5×5 crossbar, a 6×6 crossbar consumes about 21% more power, and the power consumption of a 7×7 crossbar is approximately 2.25 times higher than that of the 5×5 counterpart. Therefore, the Symmetric 3D Mesh implementation is a somewhat naive extension to the baseline 2D network because of its excessive area and power overhead.

3.3.2 Ciliated 3D Mesh

Another proposed method of constructing a 3D NoC is by adding layers of functional IP blocks and restricting the switches to one layer or a small number of layers. In this context, Feero *et al.* [47] introduce new architecture called ciliated 3D mesh. This structure is essentially a 3D mesh network with multiple IP blocks per switch. For the ciliated 3D mesh, a 3×3×3 3D mesh network with three IPs per switch, is shown in Fig. 3.3. In the ciliated 3D mesh network, each switch contains at most $5+k$ ports (one for each cardinal direction, two for up and down (one either up or down in two-layer 3D mesh), and one to each of the k IP blocks. In consequence of multiple IP cores per switch and diminished connectivity, this architecture presents lower overall bandwidth compared to the Symmetric 3D Mesh. Nonetheless, it was shown that this type of network offers an advantage in terms of energy dissipation, especially in the presence of specific traffic patterns.

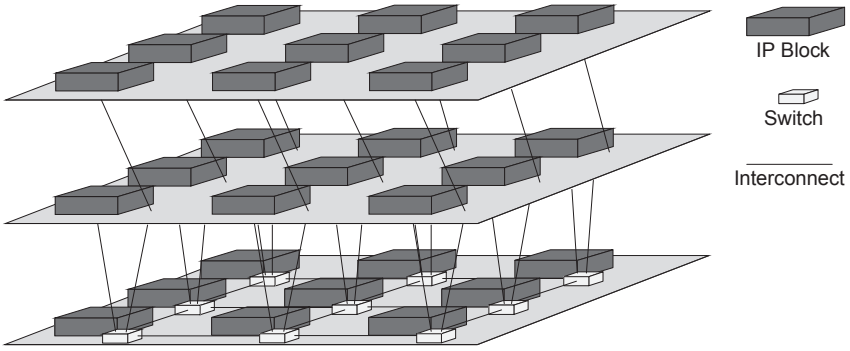


Figure 3.3: Ciliated 3D Mesh NoC architecture

3.3.3 Hybrid 3D NoC-Bus

One of the main characteristics of the 3D ICs is the short interlayer distances. In order to take advantage of this property, a Hybrid 3D NoC-Bus architecture has been proposed that is a hybrid between a packet-switched network and a bus. In this architecture, given the very small inter-layer distance, single-hop communication is, in fact, feasible. As can be seen from Fig. 3.4, the NoC router can be hybridized with a bus link in the vertical dimension to create a Hybrid 3D NoC-Bus structure. This approach was first used in a 3D NUCA L2 cache for CMPs [88]. This hybrid system provides both performance and area benefits. It requires a 6×6 crossbar, since the bus adds a single additional port to the generic 2D 5×5 crossbar. Compared to 7×7 crossbars in symmetric architecture, it is less power-hungry and occupies less area. The additional link forms the interface between the NoC domain and the bus domain. The bus link has its own dedicated queue, which is controlled by a central arbiter. Flits from different layers wishing to move up/down should arbitrate for access to the shared medium. Furthermore, each bus has only a small number of nodes, keeping the overall capacitance on the bus small and considerably simplifying bus arbitration.

Since the bus is a shared medium it does not allow concurrent communication in the third dimension, however it has been shown that the dTDMA bus outperforms an NoC for the vertical communication as long as the number of 2D planes is less than 9 [88]. Thus, the bus medium offers a sufficient degree of scalability for the third dimension.

3.3.4 True 3D NoC

The implementation of a true 3D crossbar with the target of seamless integration of the vertical links in the overall router operation is desirable. Based upon this envision, in [78] an efficient router structure has been presented. The illustration of such a 3D crossbar layout can be seen from Fig. 3.5. It

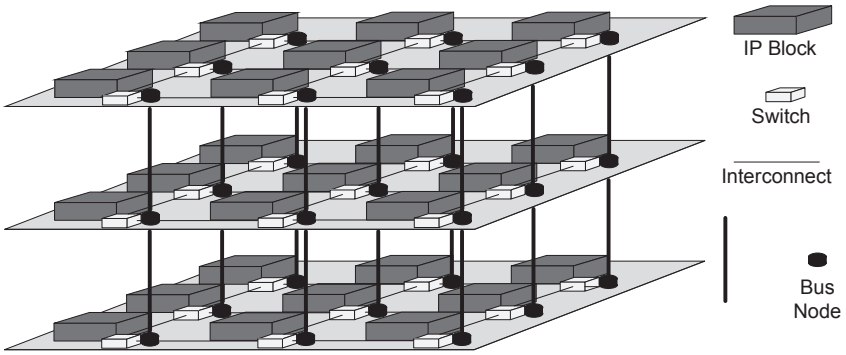


Figure 3.4: Hybrid 3D NoC-Bus mesh architecture

should be noted that according to the traditional definition of a crossbar, in the context of a 2D physical layout, it is a switch in which each input is connected to each output through a single connection point. However, extending this definition to a physical 3D structure would imply a switch of enormous complexity and size (given the increased numbers of input and output port pairs associated with the various layers). Therefore, in this architecture a simpler structure was chosen which can accommodate the connection of an input to an output port through more than one connection points.

The vertical links are embedded in the crossbar and extended to all layers. This implies the use of a 5×5 crossbar, since no additional physical channels need to be dedicated for interlayer communication.

Interconnection between the various links in a 3D crossbar would have to be provided by dedicated connections at each layer. These connecting points can facilitate linkage between vertical and horizontal channels, allowing flexible flit traversal within the 3D crossbar. The improved architecture (particularly on crossbar structure) of the True 3D NoC router called *DimDe* has been proposed which reveals the rather enhanced energy-delay product characteristic [78].

Despite this encouraging result, there are some significant drawbacks. Adding a large number of vertical links in a 3D crossbar to increase NoC connectivity leads to increased path diversity and means multiple possible paths between source and destination pairs, and results in a dramatic increase in the complexity and power consumption of the central arbiter.

3.3.5 Tree-Based 3D NoCs

Butterfly fat tree (BFT) [55][54] and the generic fat tree, or SPIN [57] are the two types of tree-based interconnection networks that have been considered for NoC applications. It has been shown that considerable enhancements can

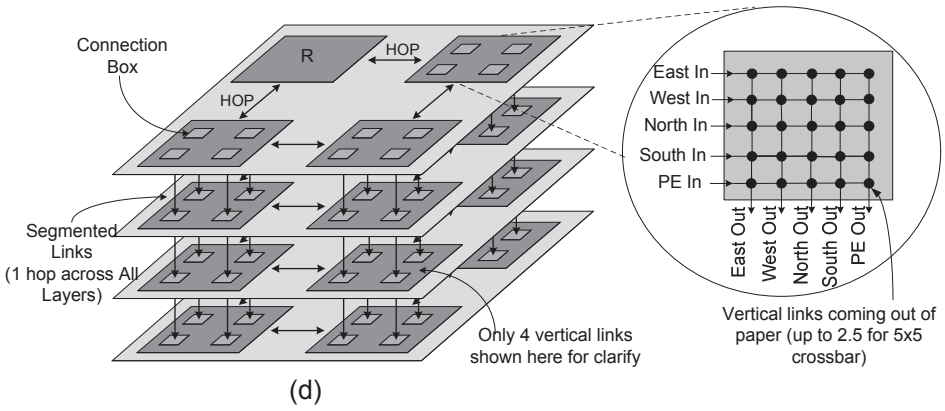


Figure 3.5: NoC architecture with true 3D crossbars [78]

be achieved when these networks are instantiated in a 3D IC environment [47]. Unlike the work with mesh-based NoCs, no new topologies for tree-based systems have been proposed. Instead, the achievable performance is demonstrated by instantiating already-existing tree-based NoC topologies in a 3D environment.

In can be concluded that when the 2D BFT network is mapped onto a multi-layer 3D SoC, wire routing becomes simpler, and the longest inter-switch wire length is reduced by at least a factor of two, in comparison with the one-layer 2D implementation. This leads to reduced energy dissipation as well as smaller area overhead. The fat tree topology has the same advantages as the BFT when mapped onto a 3D IC.

3.3.6 XNoTS

To make the best utilization of the small delay and high density of inter-wafer links, Xbar-connected Network-on-Tiers (XNoTs), which consist of multiple network layers tightly connected via crossbar switches, has been proposed [96]. XNoTs-based architectures have crossbar switches that connect different layers and their cores, in such a way that the 2D topology on every layer can be independently customized. This architecture is not power-efficient, because it requires large vertical switches.

3.3.7 MIRA

Park *et al.* [120] propose a Multi-layered on-chip Interconnect Router Architecture (MIRA), which is based on implementing a 2D mesh chip multiprocessor in three dimensions. Unlike the explained 3D routers, MIRA is a 3D NoC router architecture which is stacked into multiple layers and optimized to reduce the overall area requirements and power consumption. The

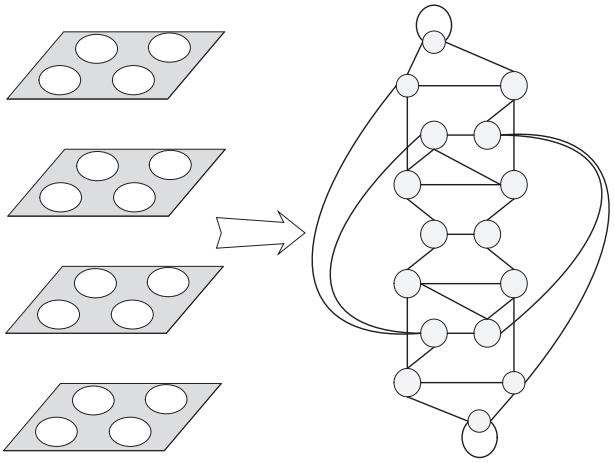


Figure 3.6: De-Bruijn graph-based 3D NoC architecture

major drawback of the architecture is that it assumes the processor cores are designed in 3D. This makes it difficult to reuse existing highly optimized 2D processor core designs.

3.3.8 De-Bruijn Graph-Based 3D NoC

Architectures like symmetric 3D mesh, ciliated 3D mesh or tree based 3D NoC have a common disadvantage: the large network latency because of the large network diameter in the mesh topology. In [29], Chen *et al.* propose a novel 3D NoC architecture based on the De-Bruijn graph. The De-Bruijn topology is an efficient topology for parallel processing purposes. The De-Bruijn graph network topology offers a smaller diameter, high connectivity and high reliability [61]. The degree of a NoC based on the De-Bruijn graph does not change with an increase in the size of the network.

This architecture benefits from a simple routing algorithm. The De-Bruijn architecture provides better performance as compared to the Symmetric 3D Mesh NoC because of shorter diameter. On the other hand, this architecture is not power-efficient as compared to the Symmetric 3D Mesh NoC, because a shorter routing path cannot be achieved in most cases. The proposed topology is shown in Fig. 3.6.

3.3.9 Serialized Vertical Channel Architecture

In 3D ICs, vertical TSVs take a significant chip area because of their typically spread-out distribution. Pasricha [121] proposes the serialization of vertical TSV interconnects to reduce their area footprint and avoid the routing congestion of interconnects. Such serialization can lead to a better thermal

TSV distribution resulting in lower peak temperatures. The extra space made available on each layer due to serialization can be used for efficient core layout across multiple layers and routing, as well as more efficient thermal TSV insertion for temperature management. Such area savings and other benefits come at the cost of nominal power and performance overhead.

The impact on performance with varying degrees of serialization was presented in [121]. The performance degradation is about 1.7% on average for various applications for 4:1 serialization (64 $\rightarrow$ 16 wires) but reaches around 16.1% for 64:1 serialization (64 $\rightarrow$ 1 wire). Performance degradation depends on the frequency of vertical transfers as well. The lower degrees of serialization like 4:1 or 2:1 are more practical because of a negligible penalty in performance and lower power consumption overhead but a significant reduction in the footprint area.

3.3.10 Honeycomb 3D Architecture

For a topology design, there is always a tradeoff between degree and diameter. The degree refers to the hardware cost. Mesh and torus are the main stream topologies because of their high regularity, symmetry and scalability but with extra hardware cost (degree). Yin *et al.* [171] propose the honeycomb interconnect topology as an alternative for NoC based designs. Honeycomb topology reduces the network cost significantly, while maintaining the positive characteristics of typical mesh and torus topologies.

For regular mesh and torus topologies in two dimensional domains, the honeycomb mesh and torus provide an approximate 40% reduction in terms of network cost. Smaller network degree for honeycomb topology makes the router architecture simpler, reliable and power efficient. The 3D honeycomb mesh topology with network degree of '5' is shown in Fig. 3.7. To further reduce the network cost, vertical links can be removed by systematically bi-partitioning of the routers into odd and even groups. The authors present the deadlock free routing algorithm for 3D honeycomb topology as well.

3.3.11 Low-Radix 3D NoC

The key problem faced by current 3D stacking technology is that only vertical inter-layer links are allowed. Due to which, the direct connection between arbitrary nodes located at different layers is not allowed. In case of 3D NoC architectures, the system design is highly constrained by the complexity and power of routers and links. Thus, the low radix routers are preferred due to lower power consumption and better heat dissipation. This makes the latency value higher due to high hop counts in network paths.

Xu *et al.* [169] present an efficient network topology for 3D NoCs by using the long range links. As shown in Fig. 3.8, utilization of long-range

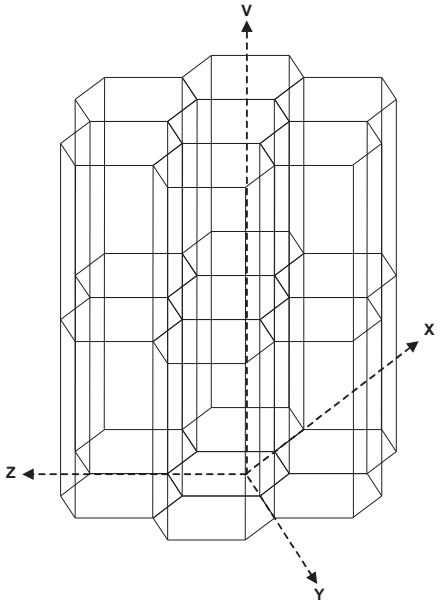


Figure 3.7: Honeycomb 3D NoC architecture

links makes the topology low diameter but requires only low-radix routers to implement. The long range links show significant reduction in latency even for the higher operating frequency and pipelined wires. The increase in power consumption is sub-linear to the increase in length. The authors present an optimal operating frequency of 1GHz for 3D NoCs because higher clock frequency for 3D chip brings the concern of high heat dissipation.

3.3.12 Layer-Multiplexed Mesh Architecture

As discussed before, architectures like Hybrid 3D NoC-Bus can reduce the network cost, but there are some other performance bottlenecks like bus

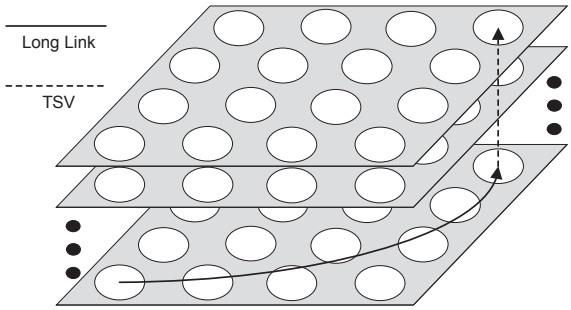


Figure 3.8: Low-Radix 3D NoC architecture

bandwidth limitations. Another issue is the layer load-balancing. If one layer is congested and other layers have very light load, the system is not power and performance efficient.

Ramanujam *et al.* [136] present an efficient Layer-Multiplexed (LM) 3D architecture for vertical communication with the consideration of load balancing. More precisely, the layer-multiplexed architecture replaces the one-layer-per-hop routing in a conventional 3D mesh with simpler vertical de-multiplexing and multiplexing stages. There are two major drawbacks for the proposed architecture. This architecture suffers from two major disadvantages. First is to traverse packets through two stage crossbar, which makes it less power efficient. Second is to use two hops per packet for vertical communication, which degrades system throughput.

3.3.13 Special Purpose 3D NoCs

Design of 3D SoCs which satisfies the application performance requirements with minimum power consumption, while meeting the 3D technology constraints, is a serious challenge. A synthesis based power-performance efficient design approach for 3D NoCs can deal with such issues.

Seiculescu *et al.* [141] present a tool for NoC topology synthesis for 3D ICs named SunFloor 3D. Path computation and assignment and placement of network elements in 3D layers are also important tasks of the tool. A comprehensive comparison between 2D and 3D NoCs shows that 3D integration can significantly reduce the latency and power consumption as compared to the 2D interconnects. The topologies produced by SunFloor tool show significant power and latency savings as compared to the standard topologies. Similar approach has been adopted by [102].

3.4 Summary

Recently, Networks-on-Chip architectures have gained popularity to address the interconnect delay problem for designing CMP/multi-core/SoC systems in deep sub-micron technology. Since three dimensional (3D) integration has emerged to mitigate the interconnect delay and power problem, exploring the NoC design space in 3D can provide ample opportunities to design high performance and energy-efficient NoC architectures. In this chapter, we have given an overview of the existing architecture for 3D NoC and highlighted their impact on network characteristics. We have first stated the motivation for 3D NoC and given an introduction of the basic concepts. Furthermore, we investigate various architectural alternatives for designing a high-performance and energy-efficient 3D NoC system. We have demonstrated that besides reducing the footprint in a fabricated design, 3D network structures provide better power consumption and performance characteristics

compared to traditional, 2D NoC architectures. It has been shown that most NoC architectures are capable of achieving better power/performance when instantiated in a 3D IC environment compared to more traditional 2D implementations.

Chapter 4

Reconfigurable Voltage/Frequency Island-Based NoC

Achieving power efficiency has become an increasingly difficult challenge, especially in the presence of increasing die sizes, high clock frequencies and variability driven design issues. Globally Asynchronous Locally Synchronous [24][104] (GALS)-based NoCs implemented using a Voltage Frequency Island (VFI) design style have become an attractive alternative to traditional designs [82]. In fact, VFI-based designs could be used for managing local frequencies or voltages to match prescribed performance limits, while minimizing energy consumption.

Assignment of frequencies and voltages to VFIs can be done by using either offline or online methods [31]. Offline methods can be used when the behavior of an application is very predictable for various input conditions [30]. However, such an approach is not well suited for applications that show large variations in their behavior for different input conditions. For such systems, online methods are more suitable [31][108]. Dynamic Voltage and Frequency Scaling (DVFS) schemes can be used to adapt the system to meet the performance requirements caused by a dynamically changing workload, while minimizing power consumption. On the other hand, in order to benefit from the GALS scheme, communication between islands should be carried out by using mixed-timing (bi-synchronous) FIFOs [25] which adapt clock frequency discrepancy; however, due to the overhead in implementing these FIFOs in terms of latency, area, and power consumption, the associated design complexity increases.

The contribution of this chapter is twofold. By exploiting reconfigurable FIFOs, the latency and power consumption overhead of bi-synchronous FIFOs can be considerably decreased in such cases that synchronization is

not required. Because the NoC partitioned into VFIs requires a method to embed and take advantage of these FIFOs, we have developed an online controller for input channels of a NoC switch which can adaptively decide the operating mode of the FIFOs. We further present four design styles for implementing Reconfigurable Synchronous/Bi-Synchronous (RSBS) FIFOs enabling the bi-synchronous FIFOs to work dynamically in either synchronous or bi-synchronous mode.

4.1 Power-Aware VFI-Based Frequency-Voltage Assignment

GALS-based communication mechanisms has been the subject of extensive prior research [43]. In [25], authors utilizes two flip-flop synchronizers to implement mixed-timing FIFO in order to interface systems on a chip working at different frequencies. The FIFOs proposed in [35] and [151] benefit from Gray and Johnson coding for read and write pointers, respectively, while ring counter has been used for the bi-synchronous FIFOs presented in [116] and [113].

Introducing an efficient technique called voltage islands [83] makes the chip design more efficient by reducing both static and dynamic power consumption of the system. In the voltage island concept, processing elements can have power characteristics independently from the rest of the design, however they use the same voltage source. This technique enables designers to optimize the power dissipation of each domain according to system requirements. As the design complexity of such systems increases with the number of island, elaborated design methodology is needed to provide efficient voltage island partitioning, voltage level assignment and physical-level floorplanning. In [63], Hu *et al.* addressed these issues by proposing a annealing-based algorithm.

The concept of voltage island has been expanded to offer further power saving by providing independent frequency level for each island (i.e. Voltage Frequency Island) thanks to the GALS technique. There have been several proposals to combine the benefits of GALS-based NoC interconnect mechanism with VFI design methods [129][18]. The study of voltage and frequency assignment has become an important aspect of VFI-based systems. For instance the authors of [111] present design methodologies for partitioning an NoC architecture into multiple VFIs and assigning frequency, supply voltage, and threshold voltage levels to each VFI according to given performance constraints at design time. Jang *et al.* [69] enchanted this methodology by proposing a systematic VFI-aware energy optimization framework being able to consider VFI-aware partitioning, VFI-aware mapping, and VFI-aware routing together.

In recent years, heuristical task mapping techniques have generated interest in VFI-based systems by taking into account other design constraints such as area and temperature characteristics. In [50], in order to minimize energy consumption while satisfying the performance constraints, a task mapping technique for heterogeneous NoC-based system operating at multiple voltage levels was presented. In this work, a voltage assignment technique has been also proposed which has one-to-one correspondence with the minimum weight graph coloring theory. In [4], Arjomand *et al.* proposed a thermal-aware voltage-frequency assignment approach for three dimensional NoCs. The methodology also utilizes an offline technique to map application tasks to processing elements of a 3D NoC platform considering performance requirements of the tasks and thermal conductivity to the heat sink.

Recently, there has been wide interest in proposing hardware-based approaches to dynamically change the frequencies and potentially voltages of a VFI system driven by a dynamic workload [31][108]. For instance, Ogras *et al.* [112] proposed a voltage-frequency control system to monitor and manage network workload dynamically to handle workload variability issues. In other recent work, Bogdan *et al.* [15], proposed a fractal-based control approach to manage the power consumption of the VFI-based systems. This technique enables the system to capture fractality and non-stationarity characteristics of workload.

Partitioning a NoC to several VFIs for assigning different voltages/frequencies is of increasing concern in today's technologies. To this end, a system comprised of a number of synchronous cores, IPs or processing elements (PEs) that can be assigned to VFIs is considered. A VFI can consist of a single PE or, depending on the physical or design considerations, may contain a group of PEs. Each VFI is assumed to have a voltage level above a certain value V_{min} and, since the architecture is globally asynchronous, locally synchronous, each module or core is assumed to be clocked by a local ring oscillator or a central clock generator controlled by the variable intra-island supply voltage [104][107]. In such systems, the assignment of voltage/frequency to each island can be classified into Static and Dynamic Voltage/Frequency Assignment techniques. In the rest of this section, we will describe the general concept of these techniques and discuss their main shortcomings.

4.1.1 Dynamic Voltage Frequency Assignment

In such systems, usually each individual processing element is a locally synchronous module operating with its own clock and either being a single VFI or forming a VFI with another synchronous module. This enables dynamic voltage/frequency scaling in each synchronous module using a DC-DC voltage regulator and a central or local variable delay ring oscillator maintaining

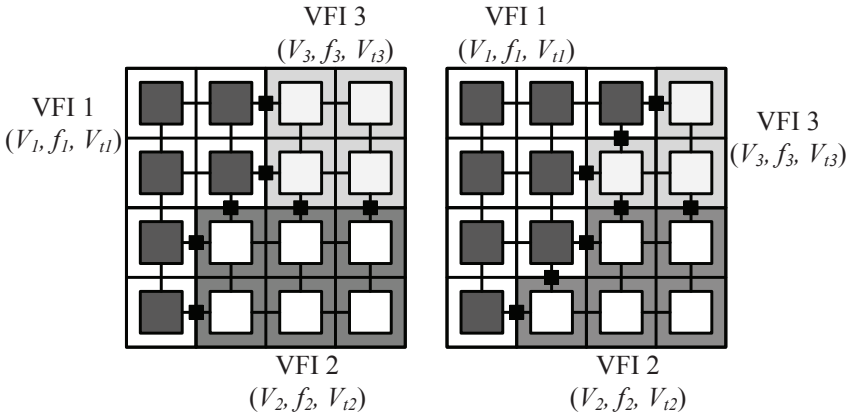


Figure 4.1: Frequency level of the nodes in a sample 2D Mesh network with 3 VFIs for two consequent periods

the clock of the VFI. There are usually a discrete set of frequency and voltage levels (usually 2 to 6 levels) assigned by some methods such as forecasting. As an instance, in prediction based algorithms, based on some training data such as the number of cycles required by the PE to process past samples, the workload requirement of the PE for the next sample may be predicted and the local voltage/frequency can be scaled accordingly.

For example, Figure 4.1 shows frequency levels in a sample 4×4 mesh-based NoC system for two subsequent periods. Such a scheme would be very useful in applications where the workload is not overwhelming. Note that the prediction generally is performed at the start of a predefined timing window having length of w packets.

For the sake of adaptivity in voltage/frequency selection, all nodes should benefit from bi-synchronous FIFOs. Let us consider a frequent case in which adjacent cores in a NoC work at a same frequency level (e.g., they belong to same VFI in current timing window). In this situation, despite both read and write clock signals of their FIFOs have equal frequencies, they are still synchronized by passing through synchronizer blocks for asserting full and empty signals. However, these FIFOs can be informed about their equal read and write clock frequencies. A reconfigurable FIFO capable to operate in both synchronous and bi-synchronous modes, can cope with this by bypassing and switching off the unused components (e.g., synchronizers, code converters) and result to considerable improvement in terms of latency, throughput, and power consumption.

In order to highlight the importance of reconfigurable FIFOs, we have embedded a simple hardware called *Mode Selector* in the input channel of a RASoC-based [172] NoC switch. As can be seen from Figure 4.2, this mod-

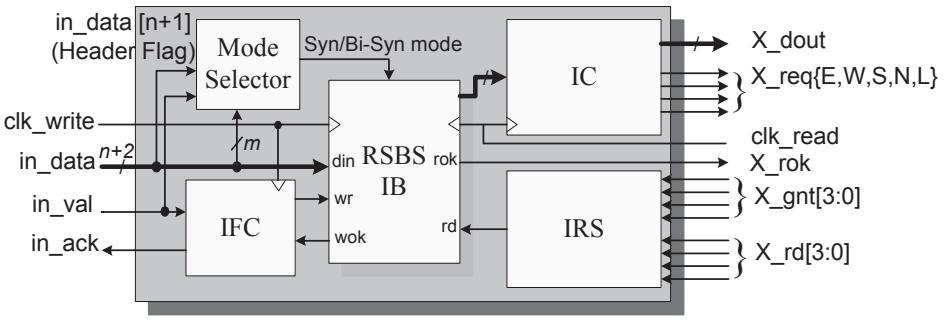


Figure 4.2: Input channel module architecture with support of Reconfigurable Syn/Bi-Syn FIFOs

ule is responsible for recognizing the equality of write (provided by output channel of the adjacent switch) and read clock frequencies and directing the buffer to operate in synchronous or bi-synchronous mode.

The input channel module shown in Figure 4.2 consists of five different units: *IFC* (Input Flow Controller), *IC* (Input Controller), *RSBS IB* (Reconfigurable Synchronous/Bi-Synchronous Input Buffer), *IRS* (Input Read Switch), and *Mode Selector* (which is added to the basic architecture of input channel module to support RSBS FIFO). The *RSBS IB* block is a dual mode FIFO buffer, while the *IC* block of each input channel performs the routing function, its *IRS* block receives *x_rd* and *x_gnt* signals and triggers the *rd* signal of the *RSBS IB* block, and the *IFC* block implements the logic that performs the translation between the handshake and the FIFO flow control protocol. Each channel includes n bits for data and two bits for packet framing: begin-of-packet ($(n+2)^{th}$ bit), and end-of-packet ($(n+1)^{th}$ bit). *IFC*, *IC*, and *IRS* modules are described in detail in [172].

To clarify the operation of this system, let us consider a dynamic voltage frequency assignment (DVFA) unit which can be implemented either locally or globally. This unit depends on its decisions at the start of each timing window H (which may have length of 1 to j packets) determines the frequency level of each switch for the next period. In the current timing window, each switch puts an m-bit number ($m{=}\lceil log_2^{Number\ of\ frequency\ levels} \rceil$) corresponding to its frequency level in the header of each packet. The *Mode Selector* block through *in_val* signal detects a flit is coming and by $(n+2)^{th}$ bit of *in_data* signal knows that it is a header flit. Then it compares the respective m bits of the header flit representing the frequency level of the adjacent switch with frequency level of the enclosing router. If the frequency levels are the same (different), it will inform *RSBS IB* block to work in the synchronous (bi-synchronous) mode. For each timing window, there is a slight switching penalty to change the FIFO mode, however the overall net-

work latency improvement achieved in the synchronous mode considerably dominates the switching penalty. In addition, since *Mode Selector* and *IFC* blocks work in parallel manner, in fact the embedded block does not affect the router critical-path delay.

Notice that the architecture shown in Figure 4.2 is an uncomplicated example of utilizing RSBS FIFOs. It could be considered as a novel technique which can complement any switch architecture. For instance, in order to trigger the *Mode Selector* block even during sending packet payload, instead of occupation m header bits, m extra wires between each adjacent switches in each direction can be added. Moreover, if there is a central DVFA unit, it can be used for assigning frequency to each island as well as selecting the operation mode of the FIFOs.

4.1.2 Static Voltage Frequency Assignment

In the static voltage frequency assignment techniques (SVFA), voltage and frequency assignment process is done for each island at the design time. Then the VFIs are created after which bi-synchronous and synchronous FIFOs are respectively employed for inter-island and intra-island communications. According to the recent work in this field [112][30][108], a VFI-based NoC is generally partitioned into 2 to 4 islands because if a larger number is used, the overhead of the bi-synchronous FIFOs will diminish the power savings gained by VFI architecture.

Designs using SVFA techniques are advantageous in the case of a system where an oracle has pre-existing knowledge of the number of run time cycles used in each PE for processing each sample of the application under consideration. Moreover, since all of the SVFA stages are done at the design time and for a specific application, this system is not practical for other applications. Thus, the general purpose multicore systems such as Multi-Processor System-on-Chip's (MPSoC's) cannot efficiently benefit from the SVFA techniques. Our proposed reconfigurable RSBS FIFOs can cope with this issue by providing sufficient reconfigurability for the system.

In the next section, we present the architecture of the proposed RSBS FIFO. It can be seen how this simple technique can significantly optimize the overall NoC power consumption as well as latency and throughput.

4.2 Reconfigurable Synchronous/Bi-Synchronous FIFO

In this section, we first present three reconfigurable FIFO design styles (#1, #2, #3) inspired by [116], [113], and [35], respectively and discuss their utilizations based on benefits and drawbacks of each design style. These FIFOs

are frequently used in the recently proposed prominent work pertaining to GALS- and especially VFI-based systems. However, to support different read and write clock frequencies, the previously proposed FIFOs cause considerable performance and area overhead to a NoC switch. To this end, we improve their architectures to provide reconfigurability and present three reconfigurable FIFOs by adding low-cost components. Meanwhile, since the explanation of all the components of each design style is beyond the scope of this work, only the parts of the baseline FIFOs which should be modified to support reconfigurability are described. More information about the other components and signals can be found in [35], [116], and [113]. For further optimization, we propose a novel RSBS FIFO based on Johnson-encoded pointers (Design Style #4). It should be noted that all the proposed design styles are scalable and synthesizable in synchronous standard cells. Finally, we present a technique for mesochronous adaptation of the RSBS FIFOs in the case where a sender and a receiver have the same clock frequency but different phases.

4.2.1 Ring Counter Based RSBS FIFO

The first FIFO presented in this section (Design Style #1) is a bi-synchronous FIFO [116] being able to interface two synchronous systems with independent clock frequencies and phases which has been used on the DSPIN NoC [144]. For the sake of metastability [51] avoidance and synchronization of pointers between two independent clock domains, it takes advantage of a token-ring-based bubble encoding technique. Note that for better reliability in metastability avoidance they use two tokens per token ring.

As shown in Figure 4.3, similar to most bi-synchronous FIFOs, five modules compose the RSBS FIFO architecture: *Write pointer*, *Read pointer*, *Data buffer*, *Full detector*, and *Empty detector*. The *Write* and *Read pointers* indicate the read and write positions, the *Data buffer* contains the buffered data of the FIFO, and the *Full* and *Empty detectors* signal the status of the FIFO.

The FIFO protocol is synchronous; all input and output signals in the sender and receiver interfaces are synchronous to their respective clock signal, *Clk_write* or *Clk_read*. In order to provide reconfigurability, we have exploited some multiplexers and flip-flops for the *Full* and *Empty detector* units used to bypass non-functional components in the synchronous mode. For this purpose, we added *Syn/Bi-Syn_Mode* signal indicating the operation mode of the FIFO (synchronous or bi-synchronous). The *Write* and *Read pointers* are implemented using the described token rings with the bubble-encoding technique. The position of the tokens determines the position of the pointer; therefore in this FIFO, length of *Write* or *Read* pointers are equal to FIFO depth, instead of $\lceil log_2^{FIFO\ depth} \rceil$. In the design style #1,

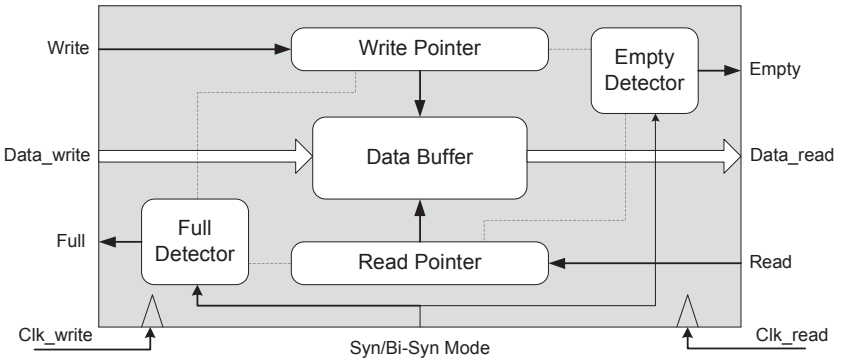


Figure 4.3: Reconfigurable Syn/Bi-Syn FIFO style #1 architecture

in order to design a RSBS FIFO based on [116], just the *Full* and *Empty detector* components should be modified, as a result, the further description of the other components has been omitted.

The *Full detector* computes the *Full* signal using the *Write_pointer* and *Read_pointer* contents. As can be seen from Figure 4.4, the *Full detector* performs the logic AND operation between the *Write* and *Read* pointers and then collects them with an OR gate, obtaining logic value '1' if the FIFO is Full or quasi-Full, otherwise it is '0'. If the FIFO operates in the bi-synchronous mode, this value will be synchronized to the *Clk_write* clock domain into *Full_s* otherwise the synchronizer can be bypassed.

The implementation of the *Empty detector* is similar to the *Full detector* in that both employ the *Write* and *Read* pointer contents. As the *Empty detector* output is correlated to the FIFO throughput, its detection logic has to be optimized, and no anticipation detector should be used. Figure 4.4 shows the *Empty detector* for a four word FIFO. In the proposed design style, first, in case that the *Syn/Bi-Syn_Mode* signal indicates that the FIFO must work in the bi-synchronous mode, the *Write_pointer* will be synchronized with the read clock into the *Synchronized_Write_pointer* (SW) using a parallel synchronizer. Otherwise the *Write_pointer* will be transferred for empty detection to respective logic gates without any synchronization (W). Next, the *Read_pointer* is recoded into the *AND_Read_pointer* (AR) using two-input AND gates. The output of AR is a one-hot encoded version of the *Read_pointer*. Finally, the Empty condition is detected comparing the $SNSW$ and AR values using three-input AND gates.

In the synchronous mode, the synchronizer blocks used to synchronize *Write_pointer* to *Clk_read* signal (Figure 4.4) and *Full_signal* to *Clk_write* signal can be switched off to prevent unnecessary energy dissipation. Depending on the design characteristic, different power reduction techniques can be utilized to switch off the idle components such as clock gating [117]

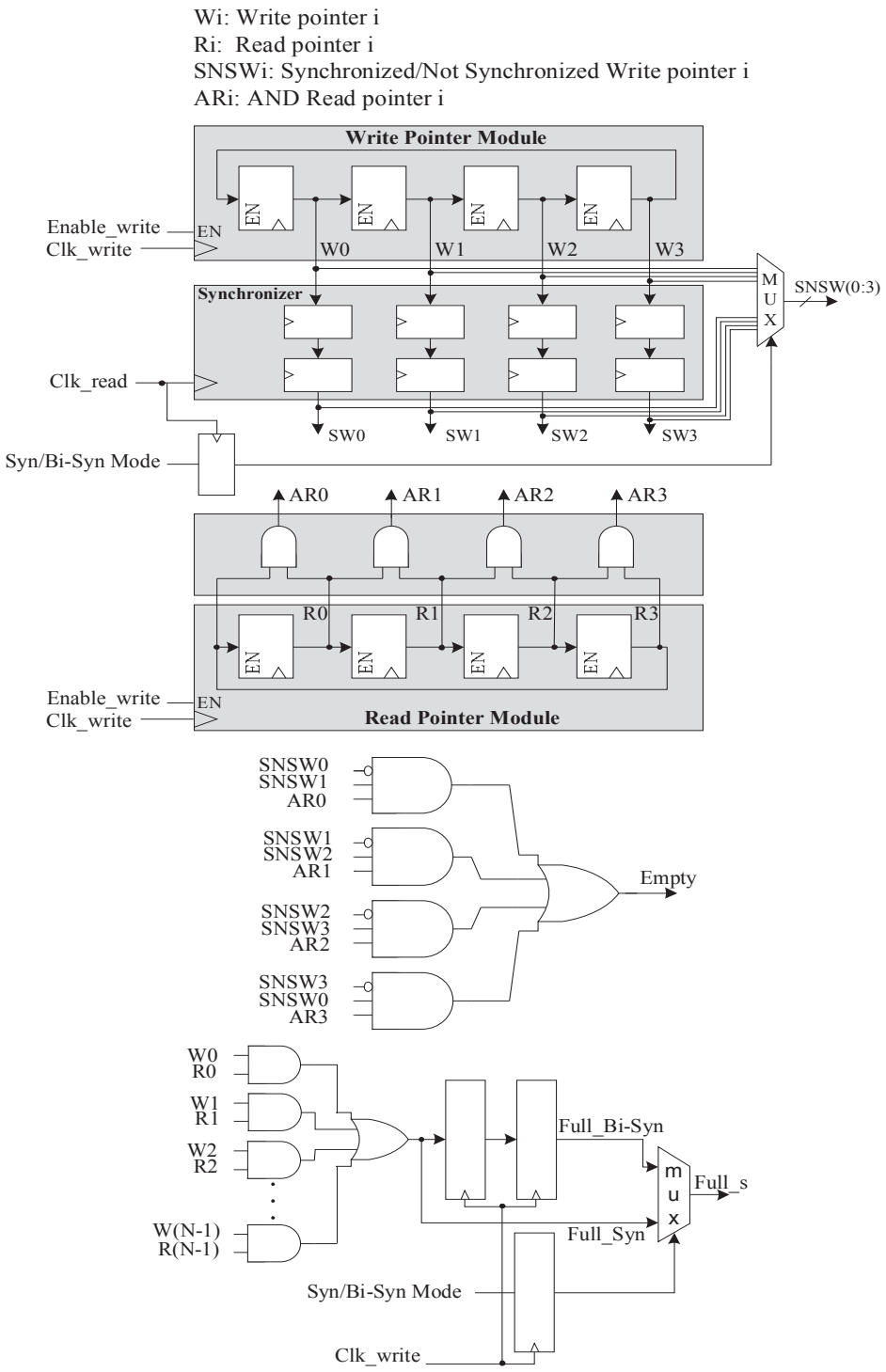


Figure 4.4: Full and empty detector details of the design style #1

and power gating [103]. These techniques have different overheads and implementation requirements. Clock gating is one of the most commonly used techniques for reducing dynamic power consumption by taking the enable conditions attached to clocked elements and deactivating them. Therefore, the switching power consumption is saved and only static power is incurred. There are different ways to apply clock gating technique such as including enable conditions in the RTL code or inserting Integrated Clock Gating (ICG) cells into the design manually or automatically. On the other hand, there is variety of ways for reducing leakage power consumption. Power gating is a popular technique in which a sleep transistor is inserted between actual ground rail and circuit ground. The technique shuts down idle blocks in order to cut-off the leakage path.

There are system trade-offs between achieving the leakage power saving in the low power mode and the energy dissipation to switch between normal and low power modes. For the RSBS FIFOs, the power reduction technique should be chosen based on different factors such as length of the timing window and area of the components needed to be switched off. For instance, for short FIFOs with a short rows of synchronizers, the clock gating approach is recommended, while power gating can be utilized for large FIFOs used in lengthy timing windows. It should be noted that as will be shown in the experimental results, RSBS FIFOs offer better latency and throughput characteristics for the system leading to reduced average packet latency (APL) of networks. The APL improvement itself results in reduced energy consumption per packet. It also decreases the static power consumption because packets get stuck in FIFOs for a shorter period of time. In fact, even without using any power reduction techniques, RSBS FIFOs are beneficial in terms of energy consumption as well.

The first proposed design style is a simple way to implement RSBS FIFOs and it has negligible area overhead compared to [116] as a result of the maximum employment of existing components. In addition, as a result of bypassing synchronizers in both full and empty detection stages, it has considerable latency and throughput improvements when it operates in the synchronous mode; hence it can be an applicable FIFO architecture to utilize in DVFA methods. However, it is not an appropriate option for SVFA techniques, because it uses ring counters for write and read pointers and compared to typical synchronous FIFOs, it requires p Flip-Flops instead of $\lceil log_2(p) \rceil$ where p is the buffer depth. As a result, it consumes more power and is not reasonable to use with all NoC switches. In addition, to differentiate between a full and an empty FIFO, this FIFO only can be filled up-to the second-to-last position, which means that a p stage FIFO can only hold p-1 items.

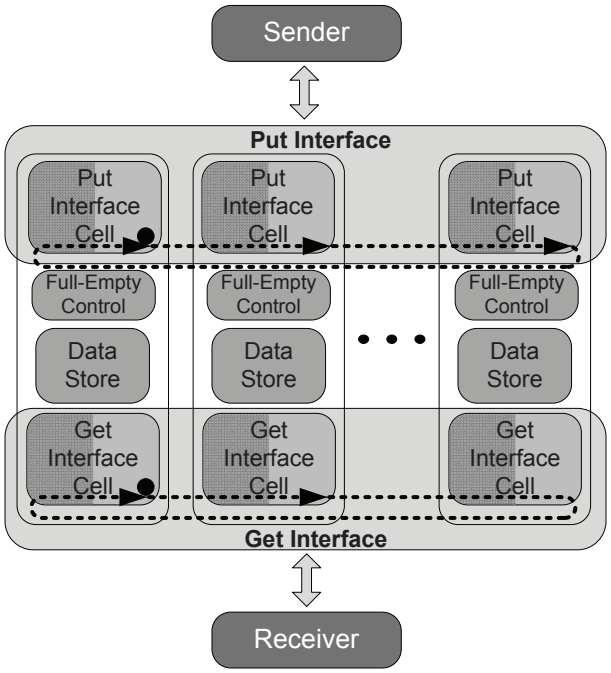


Figure 4.5: Reconfigurable Syn/Bi-Syn FIFO style #2 architecture

4.2.2 Modular RSBS FIFO

Figure 4.5 shows the Modular RSBS FIFO (Design Style #2) based on [113]. In this style, a sender communicates with the FIFO through the *put interface* and the receiver through the *get interface*. The FIFO consists of a ring of stages, where each stage has its own *put interface*, *get interface*, *full-empty control* and *data storage* cells. The *put interface* cells implement a token ring where a token is passed from one cell to the next each time a data value is written into the FIFO. Likewise, the *get interface* cells implement a token ring whose token marks the cell for the next data read operation.

In this design style, to construct a RSBS FIFO, we have modified *Get* and *Put Interface* Cells of [113]. In each *Put* (*Get*) *Interface* Cell, there is a block called *Put* (*Get*) *Control Block* which handles synchronization of the *full* (*empty*) signal that can fall asynchronously with respect to *put_clk*. Figure 4.6 shows the *Put Control Block* of a *Put Interface* of a RSBS FIFO. It raises *write_enable* if the *full* signal is low, and consists of a synchronizer, a Flip-Flop, an AND gate, and a multiplexer. The *full* signal, depending on *Syn/Bi-Syn_Mode* signal, can be synchronized or not by the multiplexer. The *write_enable* outputs from all FIFO stages are combined together into the *space_avail* signal for the sender (see Figure 10 of [113]). Note that, to prevent asynchronous change of the value of *Syn/Bi-Syn_Mode* signal in

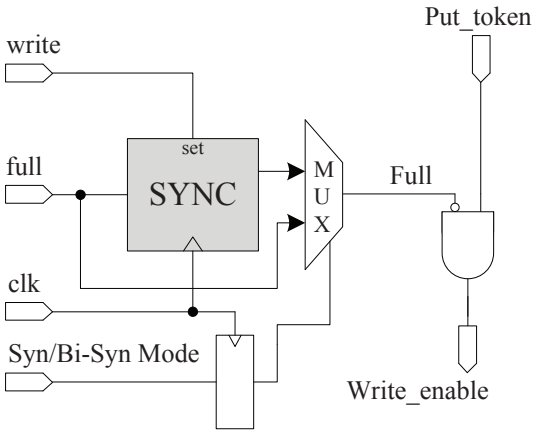


Figure 4.6: RSBS FIFO put control block

put (*get*) *control block*, a register clocked by *put_clk* (*get_clk*) is required. As shown in Figure 4.7, the synchronizer can consist of any number of half-cycle and full-cycle synchronizing stages.

The bypassing technique used in the design style #2 is very similar to its #1 counterpart; however it has some advantages and drawbacks compared to the prior style. Likewise the design style #1, this style has negligible area overhead in comparison with its baseline FIFO; however in the design style #1, in order to sample the pointers properly and consequently prevent metastability, we use two tokens for the read pointer and two tokens for the write pointer. This makes the empty detector a bit more complicated than the design style #2, as it requires comparing two synchronized write pointer bits with two read pointer bits for each stage. On top of that, in long-depth FIFOs, the large-fan-in OR and AND logic gates in full and empty detectors have negative influence on both footprint and latency. Therefore, as can be seen in the Section 4.3, the design style #2 has smaller footprint than the previous style for large FIFOs. In addition, a p stage FIFO using design style #2 can hold p items, which is more efficient than the style #1. On the other hand, per-stage synchronizers in both *put* and *get control blocks* makes the style #2 more complicated to exploit in modest FIFO designs.

Similar to the design style #1, due to use of a token ring for full and empty detectors, this proposed style is suitable only for DVFA techniques, especially when high-speed and large FIFOs are required (8×32 or larger). Using this FIFO instead of [113] in DVFA techniques results in considerable power savings and latency improvement in VFI-based systems because there are two synchronizer units in each buffer stage (slot) which can be bypassed and turned off during the synchronous mode.

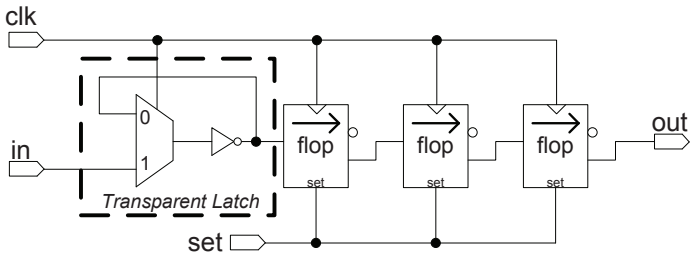


Figure 4.7: Cycle synchronizer with asynchronous set [113]

4.2.3 Gray-Encoded RSBS FIFO

So far, two alternative design styles with their pros and cons were described. In this part, we present the third design style for implementing RSBS FIFOs based on [35]. In the third proposed design style, after bypassing and switching off the useless components in the synchronous mode, in order to have a simple synchronous FIFO (not anything else such as token ring, code converter, etc) we have added an extra component to detect fullness and emptiness. Figure 4.8 shows the design style #3 architecture consisting of six main modules. The *FIFO Memory* block is a buffer accessed by both the write and read clock domains. This buffer is most likely an instantiated, synchronous dual-port RAM but other memory styles can be adapted to function as the FIFO buffer. The *sync_r2w* (*sync_w2r*) module is a synchronizer used to synchronize the *read* (*write*) pointer into the write(read)-clock domain in the bi-synchronous mode. The *FIFO rptr & empty* block is completely synchronous to the read-clock domain and contains the *FIFO read pointer* and *empty-flag* logic. Similarly, the *FIFO wptr & full* block is completely synchronous to the write-clock domain and contains the *FIFO write pointer* and *full-flag* logic. While the embedded *unsynchronized full-empty detector* module is responsible for generating full and empty flags in the synchronous mode.

The architecture of the *FIFO rptr & empty* (*FIFO wptr & full*) block is shown in Figure 4.9. This module consists of a counter and a register (*Binary register*) used to address the FIFO memory directly without the need to translate memory addresses into Gray codes. In addition, it exploits a binary to Gray converter and another register (*Gray register*) which generates the n-bit Gray-code pointer to be synchronized into the opposite clock domain and also one *Empty* (*Full*) *Detector* block to check emptiness (fullness) of the FIFO in the bi-synchronous mode.

In our third design style, once the FIFO receives the command to operate in the synchronous mode via *Syn/Bi-Syn Mode* signal, the blocks in Figure 4.8 and Figure 4.9 highlighted with gray circles are removed from

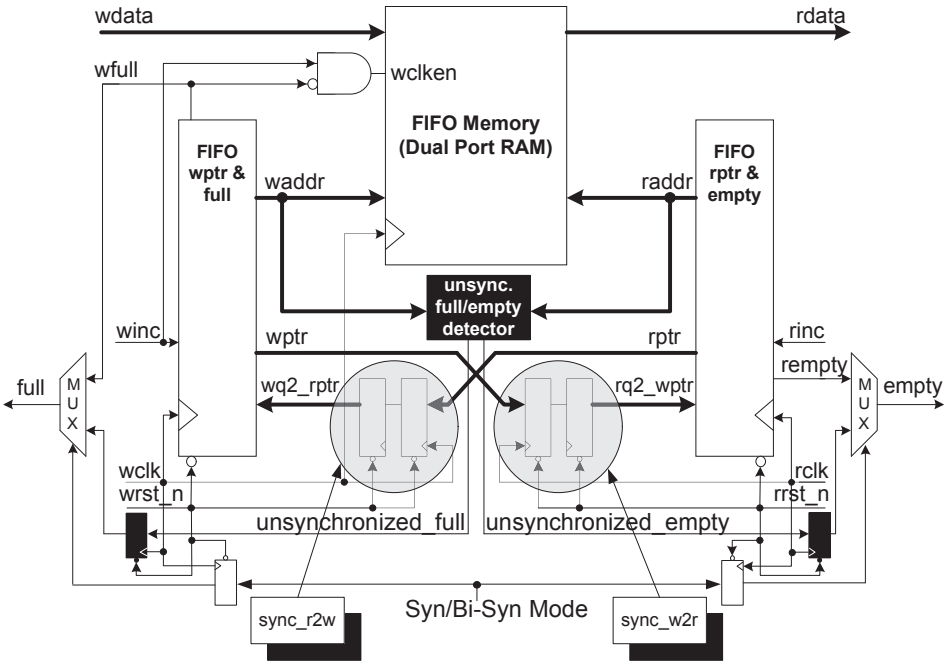


Figure 4.8: Reconfigurable Syn/Bi-Syn FIFO style #3 architecture

the FIFO path and switched off. Since in the synchronous mode it is not necessary to synchronize the pointers into opposite clock domains, we add the *unsynchronized full-empty detector* block to produce the *empty* and *full* flags using only the binary pointers. In contrast, as the mode changes to bi-synchronous, the blocks in Figure 4.8 shown in black color are bypassed and turned off and the fullness and emptiness are again checked using the gray pointers and synchronizers.

FIFOs based on the design style #3 can be filled up to the last position, because *empty* and *full detectors* utilize an extra bit in each pointer. When the write pointer is incremented past the final FIFO address, the write pointer will assert the unused MSB while setting the rest of the bits back to zero and the same is done with the read pointer. As can be seen from Figure 4.9, in contrast with the previous styles, the design style #3 uses pointers with $\lceil log_2^{FIFO\ depth} \rceil + 1$ bits and therefore it can be argued that this design style is practically a simple synchronous FIFO when the unused components are turned off in the synchronous mode. In addition, although in this style some small components are added, it still has a smaller footprint than other mentioned styles (see Section 4.3).

This style supports only FIFO capacities that are powers of two, but in case that this condition can be tolerated by the system, it could be a suitable option for VFI-based NoC systems when using either DVFA or

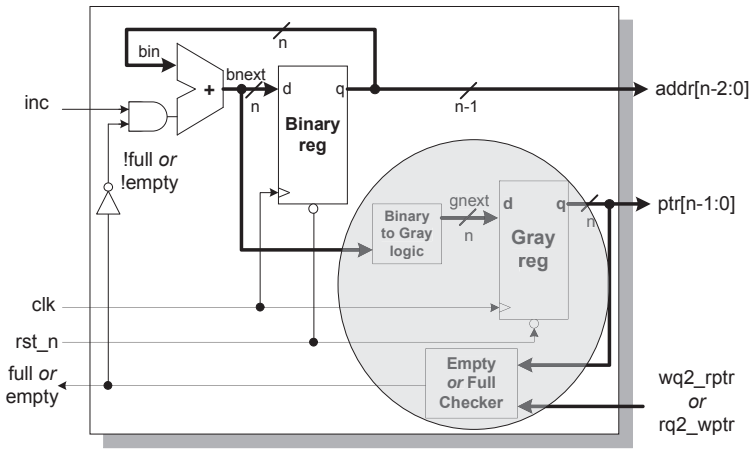


Figure 4.9: FIFO wptr & full block diagram

SVFA techniques. It should be emphasized that in such NoC systems which benefit from SVFA techniques, the position of islands are constant. These NoC systems are not appropriate to be mapped for various applications at different times. Therefore, it is desirable to have such synchronous FIFOs for intra-island communication which do not have extra latency and power consumption overhead. As a result, if the area overhead of the inactive components is acceptable for the system, exploiting a FIFO using the design style #3 is quite effective.

4.2.4 Johnson-Encoded RSBS FIFO

The Gray-encoded reconfigurable FIFO for the design style #3 offers many advantages in terms of power consumption and performance. However it still suffers from latency and power overheads due to existence of pointer counters and complexity of fullness/emptiness checking logic. Moreover, as mentioned before, the Gray-encoded design style can only support FIFO capacities that are powers of two. The presented FIFO in this section is capable of overcoming the aforementioned issues. In contrast with all the presented design styles, in this section we illustrate novel emptiness and fullness detectors (FIFO *wptr* & *full* and *rptr* & *empty* blocks) which are not based on the existing bisynchronous FIFOs. The design style #4 is based on Johnson encoding and has substantial advantages over the aforementioned design styles. Firstly, it provides more efficient reconfigurability to bi-synchronous FIFOs to prevent their associated power and latency overheads in such cases that their synchronizers are not needed. Secondly, in addition to register based implementation of Johnson-based FIFOs using one-hot addressing, it supports standard memory based implementation addressed by normal

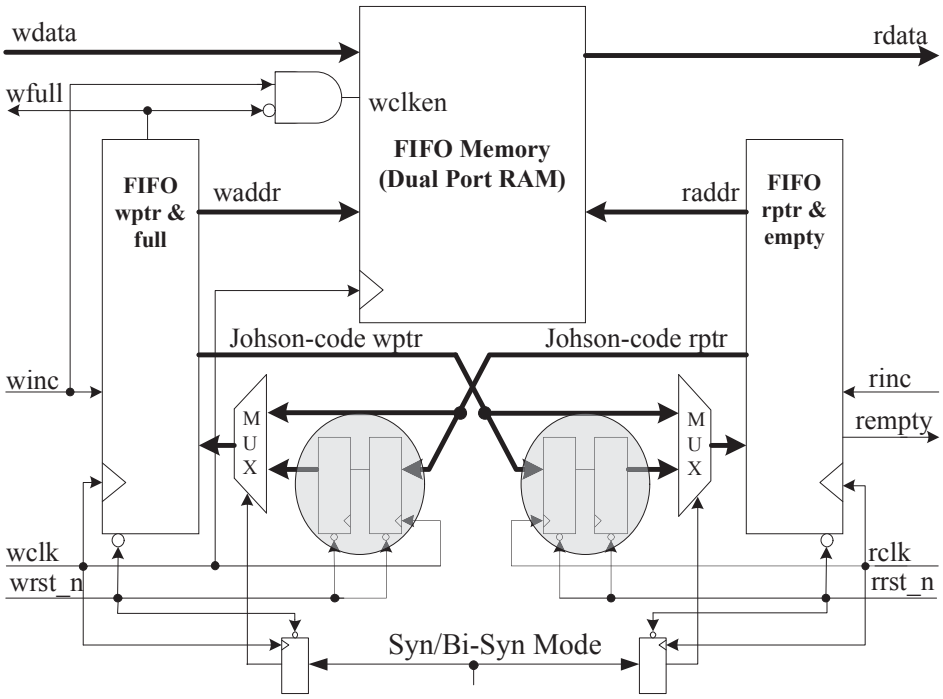


Figure 4.10: Reconfigurable Syn/Bi-Syn FIFO style #4 architecture

binary code.

As shown in Figure 4.10, similar to most bi-synchronous FIFOs, five typical modules compose the Johnson-based RSBS FIFO architecture: FIFO Memory block, *sync_r2w*, *sync_w2r*, *FIFO rptr & empty*, and *FIFO wptr & full*. In the proposed design style, to provide reconfigurability, we have exploited two multiplexers and two flip-flops to bypass unused components in the synchronous mode. For this purpose, we added the *Syn/Bi-Syn_Mode* signal indicating the operation mode of the FIFO. Before describing the main function of the RSBS FIFO, let us first focus on the properties of Johnson encoding [151] for the FIFO read and write pointers and the internal structure of the *FIFO wptr & full* (*FIFO rptr & empty*) block.

As discussed, Gray code poses some limitations in terms of the implementation complexity. The first reason is that Gray code allows encoding only "power of two" ranges, while the FIFO size may be optimal at a value that is not a power of two. The second limitation is that in contrast to binary encoding with the full-adder standard-cell, there is no elementary logical operator to perform an addition in Gray encoding. Hence, the increment of the pointers needs to be hardwired at the cost of more area and lower performance. In order to cope with this issue, we use Johnson encoding for read and write pointers in this design style.

Table 4.1: 4-bit Johnson encoding

Count	Q_0	Q_1	Q_2	Q_3
0	0	0	0	0
1	1	0	0	0
2	1	1	0	0
3	1	1	1	0
4	1	1	1	1
5	0	1	1	1
6	0	0	1	1
7	0	0	0	1

Johnson encoding is also a code with a Hamming distance of 1 between consecutive elements which allows a safe synchronization of the pointers. To implement the sequence, bits are chained in series as in a shift register, and the loop is closed using an inverter, so that the least significant bit is implemented as the negation of the most significant bit. Table 4.1 and Figure 4.11 show an example of 4-bit Johnson encoding and implementation with initial register values of 0000, respectively. As shown in Figure 4.11, a N-stage Johnson encoder circulates a single data bit giving sequence of $2N$ different states and can therefore encode $2N$ values by N bits. To differentiate the FIFO fullness and emptiness, a parity bit to the binary pointers is added for virtually doubling the addressing range of the pointers [3]. This parity method is extensible both to Gray and Johnson encodings, but for Johnson encoding, because of the twisted-ring sequence, it is simpler. When Johnson encoding is used, the buffer is empty if $write\_pointer = read\_pointer$, and it is full if $write\_pointer = NOT\ read\_pointer$.

The architecture of the *FIFO wptr & full* (*FIFO rptr & empty*) block is shown in Figure 4.12. This module consists of a *Johnson-encoded register* to generate the n-bit pointer to be synchronized into the opposite clock domain. In addition, it exploits a Johnson to binary converter and another register (*Binary register*) used to address the FIFO memory directly without the need to translate memory addresses and also one *Full* (*Empty*) *Detector*

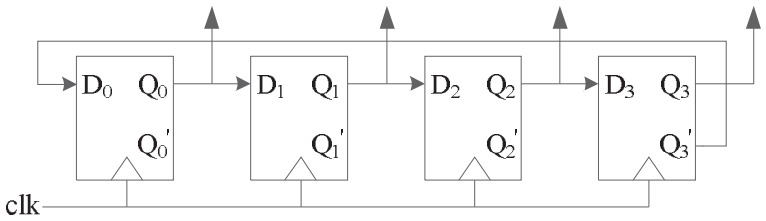


Figure 4.11: 4-bit Johnson encoding implementation

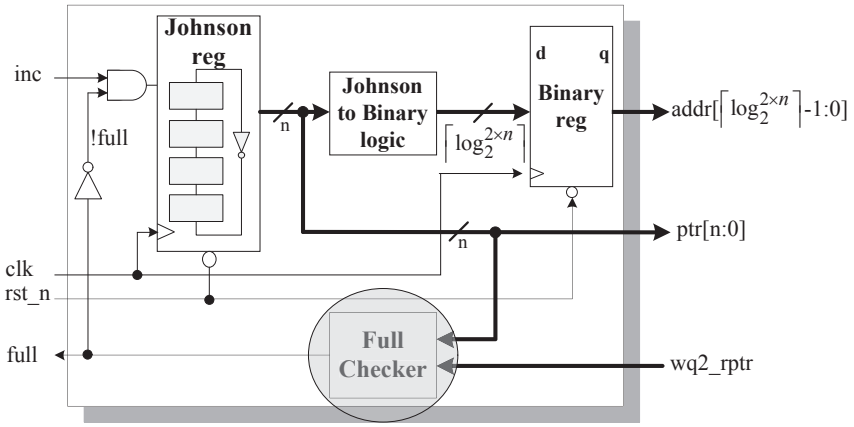


Figure 4.12: FIFO wptr & full block diagram

block to check fullness (emptiness) of the FIFO.

As a conclusion of this section, we now list the contributions of our proposed RSBS FIFOs and highlight their advantages over other available bi-synchronous FIFOs. Firstly, we have presented a generic technique which is capable of providing reconfigurability for existing bi-synchronous FIFOs which leads to considerable latency improvement and power savings for the VFI-based systems. We applied this technique to three popular bi-synchronous FIFOs as presented in the above Design Style #1 to #3 subsections. Each of the three enhanced reconfigurable FIFOs are suitable for different types of applications. Secondly, we have proposed our own reconfigurable FIFO which is based on Johnson encoding. Since the proposed Johnson-encoded FIFO has been redesigned from grounds up to provide reconfigurability, the shortcomings of the previous RSBS FIFOs have been addressed in this design style. Our methodology enables designers to add reconfigurability to their existing designs or choose our proposed Johnson-encoded RSBS FIFO. Finally, in the following, it is shown that the proposed RSBS FIFOs can be easily adapted to be used in the mesochronous mode.

4.2.5 Mesochronous Adaptation

In some cases, it is not possible to exploit a central clock generator for NoC-based systems (specifically for DVFA-based ones). In these situations, each node has its own clock generator (phase-locked loop) and the FIFO architecture should be adapted to interface mesochronous clock domains where the sender and the receiver have the same clock frequency but different phases. To this end, the RMBS (Reconfigurable Mesochronous/Bi-Synchronous) FIFO should be utilized instead of the RSBS one. The phase difference can be constant or slowly varying. According to [99], metastabil-

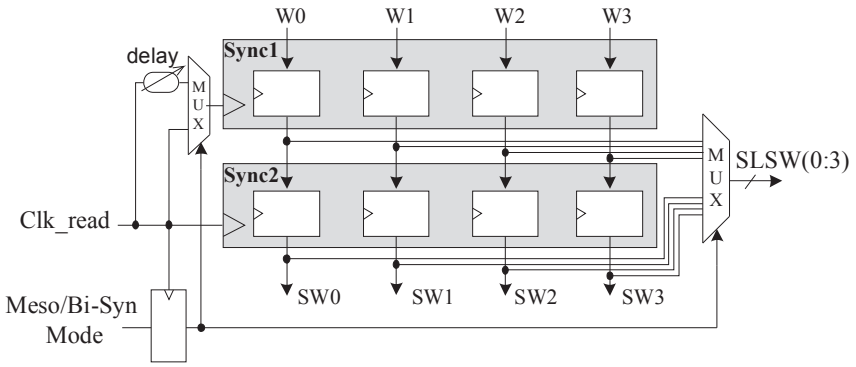


Figure 4.13: Mesochronous adaptation for the design style #1

ity can be avoided when the rising edges of the clock signals are predictable, and the two rows of registers in the synchronizer can be reduced to a single row of registers.

As an example, Figure 4.13 and Figure 4.14 show part of the proposed design styles #1 and #3 which we have modified for correct emptiness detection in the mesochronous mode, respectively. In this case, the first row of registers is clocked using a delayed version of the read clock in the mesochronous mode. This delay must be chosen to exchange the data without metastable situations. The delay can be a programmable delay, or any other metastability-free solution, as for example in the Chakraborty-Greenstreet [20] architecture which allows the FIFO to work also on plesiochronous (small difference of frequency) clocks. Likewise, if the write and read clocks are out of phase by 180° (clock-inverter), no programmable delay is needed because, by-construction, the communication is free of metastability. If the difference of phase varies between 90° and 270°, the interface is also free of metastability. This mesochronous adaptation of the bi-synchronous FIFO is simple and allows switching between the mesochronous and bi-synchronous modes.

4.3 Analysis and Case Study

To assess the efficiency of the proposed RSBS FIFOs, a synthesizable model for each style and its baseline FIFO has been developed. We have simulated the reconfigurable FIFOs based on each design style to characterize their latency, throughput, area, and power consumption. Note that, to observe the power efficiency of the FIFOs, we have employed them in a NoC-based MPEG system as a case study and compared to a similar system using conventional bi-synchronous FIFOs.

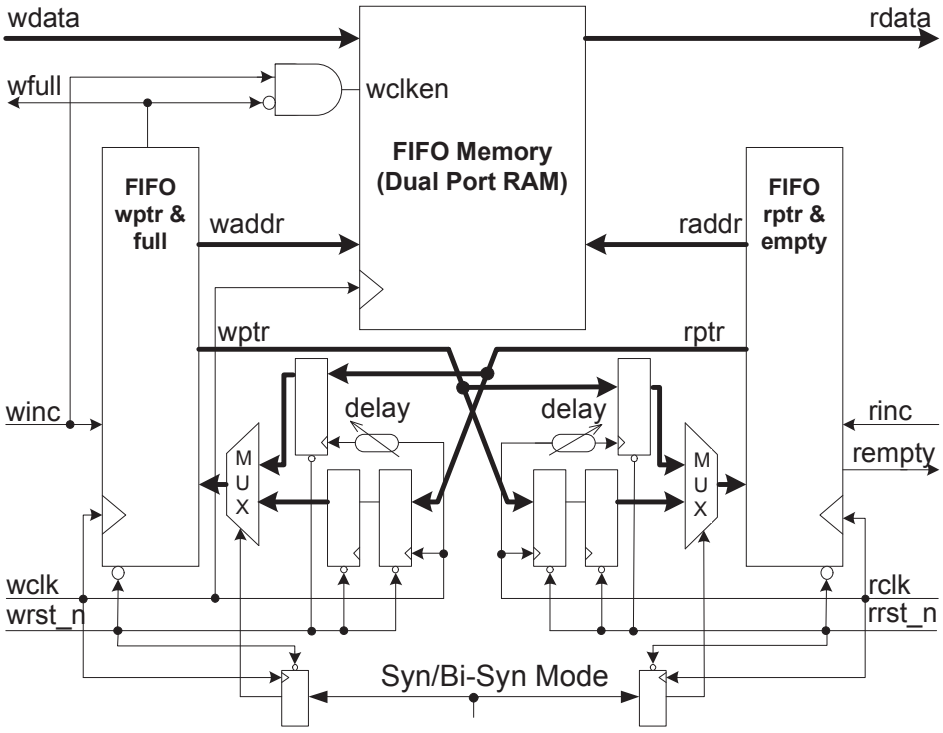


Figure 4.14: Mesochronous adaptation for the design style #3

4.3.1 Latency analysis

As the sender and the receiver have different clock signals, the latency of the FIFO depends on the relation between these two signals. The latency can be decomposed in two parts: the state machine latency and the synchronization latency. As the state-machine is designed using a Moore automaton, its latency is one clock cycle. In bi-synchronous mode, s rows of registers compose the synchronizers and its latency is ΔT plus one clock cycle, where ΔT is the difference, in time, between the rising edges of sender and receiver clocks. As this difference is between zero and one *Clk_read* clock cycle, the latency of the RSBS FIFO for all three styles is between s and s+1 *Clk_read* clock cycles in the bi-synchronous mode. Obviously, in the synchronous mode, there is no difference between *Clk_read* and *Clk_write* and also there is not any synchronizer, and hence data can be fetched by the receiver on the next rising/falling edge of *Clk_read*.

As an example, Figure 4.15 shows a simulation plot of a three stage FIFO with a 2-cycle synchronizer. The *put* and *get* clocks have the same frequency and zero phase offset. Once the first *put* request is issued, data is written to the first FIFO stage on the next rising clock edge. Three clock cycles later

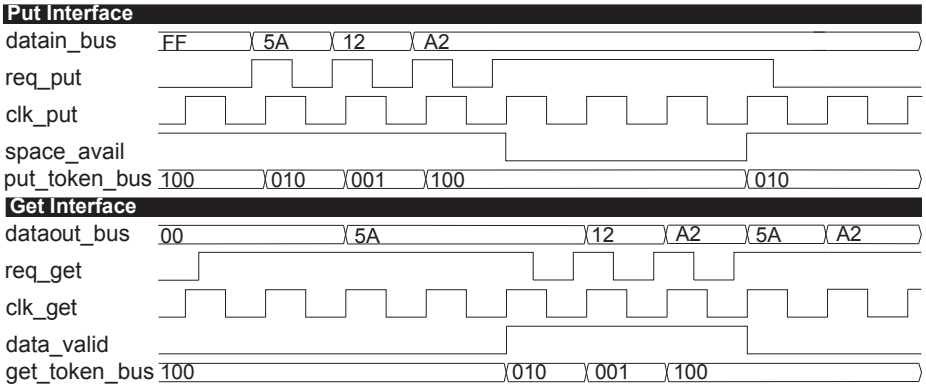


Figure 4.15: Latency analysis of conventional bi-synchronous FIFOs

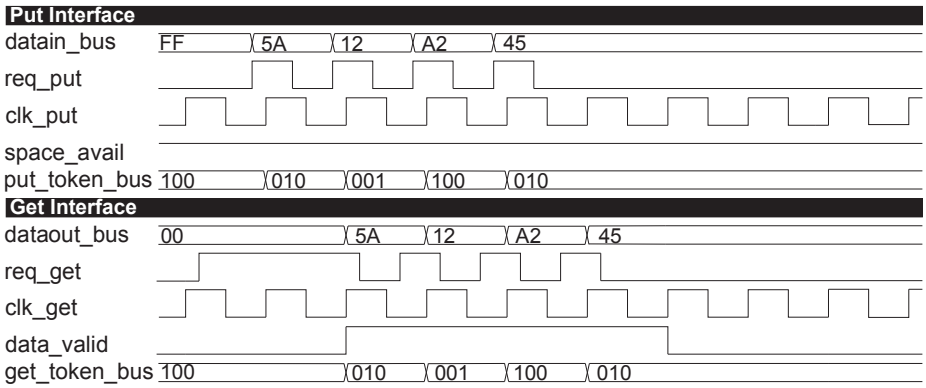


Figure 4.16: Latency analysis of the RSBS FIFO

data_valid rises, allowing the receiver to consume the data. By this time, three more FIFO writes have taken place and the *space_avail* signal drops to notify the sender that the FIFO is full. Another three clock cycles later, signal *space_avail* rises and the sender can write to the first FIFO stage again. The *put* request is not serviced until this time. On the contrary, Figure 4.16 shows a simulation plot of a three stage RSBS FIFO used in the synchronous mode. At the moment the data is written to the first FIFO stage, one clock cycle later *data_valid* rises allowing the receiver to consume the data.

When a RMBS FIFO is used, the latency of the bi-synchronous FIFO is reduced, because a single register can replace the two-register synchronizer. In addition, the ΔT is constant as the phase difference is constant. In that case, the latency of the FIFO is one clock cycle plus ΔT.

4.3.2 Throughput analysis

The throughput of the FIFOs for each design style and each operation mode has been analyzed as a function of the FIFO depth. For all the design styles in the bi-synchronous and mesochronous modes, as the synchronizers add latency, the performance of flow control of the FIFO is penalized. In the case of a deep FIFO, those latencies do not decrease the FIFO throughput since the buffered data compensate the latency of the flow control. As the FIFO, based on any of the design styles, operates on the synchronous mode, the minimum FIFO depth required to provide maximum throughput decreases because there is no need for synchronizers. Table 4.2 shows the minimum FIFO depth for 50% and 100% throughput as a function of the clocking mode. Note that for the bi-synchronous mode analysis, the write and read clock frequencies are equal, otherwise it is not possible to obtain 100% throughput. Meanwhile, there is an empty slot in the table because the minimum possible FIFO depth for the design style #1 is four.

Table 4.2: Minimum FIFO depth in function of the clock relation and required throughput

Style	**Mode**	**Minimum depth for 50% throughput**	**Minimum depth for 100% throughput**
Design Style #1	Bi-syn. Mode	5	6
	Meso. Mode	4	5
	Syn. Mode	-	4
Design Style #2 & #3 & #4	Bi-syn. Mode	3	6
	Meso. Mode	2	4
	Syn. Mode	1	2

4.3.3 Area Calculation

The area of the FIFOs was computed once synthesized on CMOS 90nm *GPLVT STMicroelectronics* standard cells using Synopsys Design Compiler. Different FIFO depths are used to illustrate the scalability of the architectures. Table 8.5 shows the area of the 16 and 32-bit RSBS FIFOs for each design style and the baseline bi-synchronous FIFOs as a function of the FIFO depth. Note that for the design style #4, there is no comparison to any baseline FIFO owing to its novelty. In addition, to clarify the impact of synchronizers on each design style, Table 4.4 formulizes the total number of synchronization components used in each design style; let p be the buffer depth, z be the number of full synchronizer (registers) rows, and f be the number of half synchronizer (transparent latches) rows.

For the design styles #1 and #2, even though there is no need for additional components to detect emptiness and fullness, there is still a slight area overhead compared with the baseline FIFOs. In the case of the design style

Table 4.3: Area and overhead comparison between the RSBS and RMBS design styles and their baseline designs (buffer length × buffer width)

Style	**4×16** (μm^2)	**4×32** (μm^2)	**8×16** (μm^2)	**8×32** (μm^2)
Panades et al. [116]	3600	6511	7117	12939
Design style #1 RSBS FIFO	3635 (0.98%)	6546 (0.54%)	7187 (0.99%)	13009 (0.54%)
Design style #1 RMBS FIFO	3664 (1.79%)	6570 (0.91%)	7264 (2.06%)	13089 (1.16%)
Ono et al. [113]	4266	6866	7004	10906
Design style #2 RSBS FIFO	4332 (1.54%)	6933 (0.97%)	7105 (1.43%)	11007 (0.93%)
Design style #2 RMBS FIFO	4382 (2.73%)	6992 (1.84%)	7319 (3.01%)	11123 (1.99%)
Cummings [35]	3326	5779	6237	11298
Design style #3 RSBS FIFO	3434 (3.25%)	5888 (1.88%)	6354 (1.88%)	11415 (1.04%)
Design style #3 RMBS FIFO	3530 (6.13%)	5983 (3.53%)	6509 (4.36%)	11570 (2.41%)
Design style #4 RSBS FIFO	3331	5795	6267	11332
Design style #4 RMBS FIFO	3420	5877	6416	11463

#3 there is a negligible area overhead which decreases when the FIFO depth increases. Compared to the design style #3, the Johnson-based RSBS FIFO (design style #4) has lower area overhead due to removed pointer counters and simplified fullness/emptiness detection logic. Note that for the design style #1, a p stage FIFO can only hold *p-1* items.

Table 4.4: Total number of synchronization components

	Design style #1	**Design style #2**	**Design style #3 & #4**
Number of synchronization components	$(p+1)\times(z+f)$	$2\times p\times(z+f)$	$2\times\lceil log_2^p+1\rceil\times(z+f)$

4.3.4 Power Consumption and Latency Evaluation of a VFI-based NoC Running MPEG-4 Decoding System

We apply the proposed RSBS-FIFO-based switch to the NoC-based MPEG-4 decoder shown in Figure 4.17 [153] and compare it with a similar system which does not benefit from the reconfigurable FIFOs. The MPEG-4 decoder system is modeled and mapped on a 5×3 NoC. In the system of Figure 4.17, each node has a 5×5 crossbar switch based on the Router Architecture for SoC (RASoC) [172] NoC switch; one port of the crossbar switch is connected to the functional block and the other four ports are used to interconnect neighboring modules. In the figure, the numbers above links connecting nodes present average bandwidth between them. Since MPEG videos show

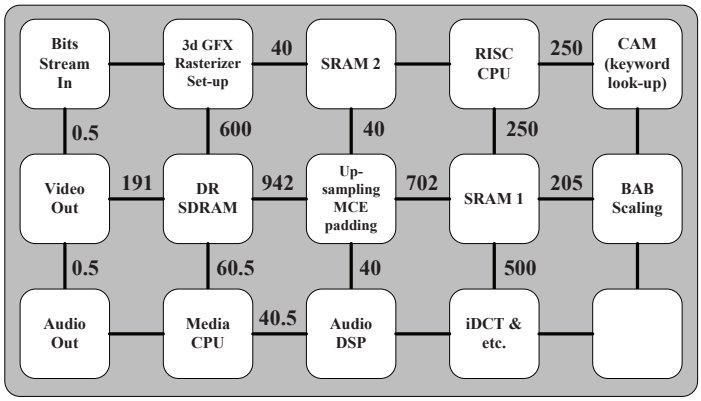


Figure 4.17: Mapping of MPEG-4 decoding system on a mesh NoC: Average bandwidths between two nodes are presented. (MB/s)

a lot of variability in processing time depending on the type of frame being processed, we perform prediction-based dynamic voltage/frequency assignment on each node based on the DVFA algorithm proposed in [108]. The prediction decision is taken at the start of processing of a new macroblock at each node, and for each input channel we add a synchronous/bi-synchronous mode selector unit based on the architecture explained in Section 4.1.1. The simulation is performed for three different frequency sets having 2, 4, and 6 frequency levels. We assume that the switches are clocked by a central clock generator block; therefore they do not need the mesochronous adaptation. In addition, for all the RSBS FIFOs, we consider two-row synchronizers. In this work, clock gating is used as power saving method; although more power could be saved by exploiting transistor level techniques as power gating (note that the loss of states in registers in sleep mode should be considered).

Figure 4.18 shows the average power saving percentages of the NoC switches achieved by exploiting the RSBS FIFOs instead of the conventional bisynchronous FIFOs. The comparison is made between each design style and its baseline counterpart for three different frequency sets. We compare the design style #4 with the conventional bisynchronous FIFO presented in [35] because of their similarity in the main structure. As the results show, we get around 3.5-17% savings over the baseline architectures. As expected, when there are fewer frequency levels, the possibility of operating in the synchronous mode increases and more power can be saved. Note that in this simulation, all the FIFOs have eight 16-bit slots.

The simulation also has been performed to determine the percentage of average packet latency improvement for two different data widths. The packet latency is defined as number of cycles from the creation of the first flit at a source node to the moment when the last flit is delivered to a

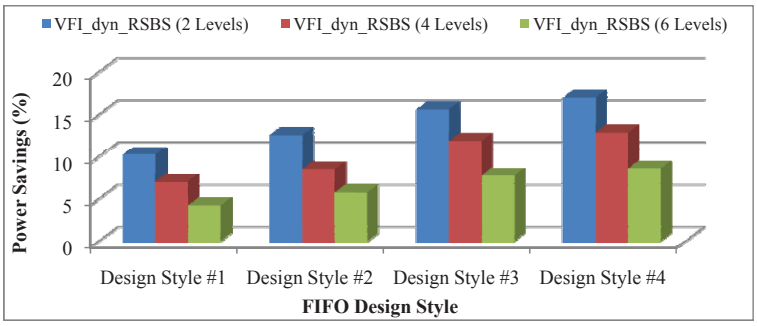


Figure 4.18: Average power savings for the NoC switches used in MPEG Encoder

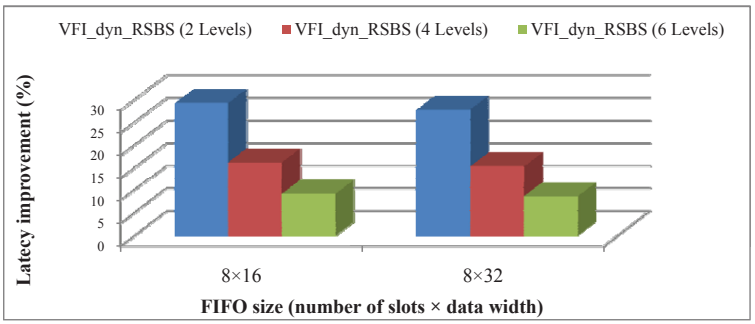


Figure 4.19: Average latency improvement for the NoC-based system running MPEG Encoder

destination node. Whereas, each packet during the traversal from a source to a destination may pass from several VFIs having different frequencies, therefore to perform a correct latency assessment, we measure the latency of a packet for each region separately, and subsequently merge the measured latency values. It was assumed that packets had a fixed length of five flits, and the data width was set to 16 and 32 bits for each set of results. As the latency is measured by the number of cycles and all the design styles have the same number of synchronizer rows, the simulation has been performed only for the NoC using the design style #3 and its baseline for three different frequency levels. It can be concluded from the simulation results shown in Figure 4.19 that around 8.5-29% latency improvement over the baseline architecture can be gained especially for the VFI-based system with a few number of frequency levels.

4.4 Summary

In this chapter, the reconfigurable synchronous/bi-synchronous FIFO that can be used to improve power and performance characteristics of the VFI-based NoC, was presented. The FIFO, which is able to work in the synchronous and bi-synchronous mode, address the squandered synchronization power and latency overhead for the case that adjacent switches in the NoC system operate in the same clock frequency but suffer from unnecessary synchronization. The FIFOs are scalable and synthesizable in synchronous standard cells. A technique for mesochronous adaptation of the FIFOs was suggested. Moreover, we presented different techniques to describe how the FIFO could be utilized in the VFI-based NoC. For this purpose, four different design styles were developed and synthesized with 90nm process library and thoroughly analyzed in terms of power consumption, latency, throughput, and area. Our results revealed that compared to a non-reconfigurable system architecture, a NoC system using the proposed FIFOs and the operation mode controller is able to run MPEG-4 encoder application with considerably higher performance and lower power consumption at a cost of negligible area overhead.

Chapter 5

Bidirectional Bisynchronous Vertical Channels for 3D NoC

The three-dimensional Networks-on-chip approach offers a matchless platform to implement the globally asynchronous, locally synchronous design paradigm [104]; this makes the clock distribution and timing closure problems more manageable and enables 3D technology to be suitable for heterogeneous integration. In addition, intrinsically, the GALS-based approach presents a suitable platform to implement Dynamic Power Management (DPM) techniques such as dynamic frequency scaling (DFS).

Recently, a number of technologies for 3D chip manufacturing have been presented, including package stacking, die stacking, wafer stacking, and device stacking. Out of these, wafer stacking is one of the most promising yet inexpensive implementation technologies for 3D ICs [125], and the focus of this work. Wafer stacking relies on TSVs [147] for vertical connectivity, guaranteeing low parasitics and high density of vertical wires. In 3D NoCs, as the number of cores increases in each layer, the amount of communication between layers is also expected to grow, and consequently the number of interconnect TSVs will get higher. Since each TSV requires a pad for bonding to a wafer layer, the area footprint of TSVs in each layer should be considered.

In this chapter, we explore a mechanism to reduce TSV area footprint and optimize 3D NoC power consumption, and thus improving 3D IC cost, routability, and thermal efficiency. Specifically, we propose a novel technique to replace the pairs of unidirectional vertical channels between layers by a Bidirectional Bisynchronous Vertical Channels (BBVC) that is dynamically self-reconfigurable to be used in either out-going or incoming direction. To compensate the bandwidth exacerbation, we exploit the low-latency na-

ture of vertical TSVs by establishing high-speed inter-layer communication using our proposed RSBS FIFOs presented in Chapter 4. Moreover, we enhance the power characteristic of the proposed mechanism and present a forecasting-based dynamic power management scheme being able to adjust frequency of vertical channels based on the inter-layer communication traffic.

5.1 Motivation

As discussed in Chapter 3, the major advantage of 3D ICs is the considerable reduction in the length and number of global interconnects, resulting in an increase in the performance and decrease in the power consumption and area of wire limited circuits. This astonishing difference should be somehow exploited. With this in mind, it was mentioned that TSVs are limited resources in 3D-ICs. In addition, reducing the number of TSVs leads to crosstalk capacitance reduction, which in turn reduces propagation delay. This can enable even higher clock frequencies on the wires. These limitations could be alleviated by utilizing high-speed vertical communication.

In order to reduce the interconnect TSV footprint, Pasricha proposed a technique for serialization of vertical TSV interconnects [121]. Although, this can lead to a better thermal TSV distribution resulting in more efficient core layout across multiple layers due to the reduced routing complexity, it degrades the performance due to serialization overhead and low bandwidth utilization. It should be noted that serialization solely cannot offer any improvements in terms of dynamic power consumption. As will be shown later, our technique can achieve their improvements with less or in some cases without any performance degradation by utilizing high speed bidirectional bisynchronous vertical channels. In addition to this important enhancement, our proposed forecasting-based dynamic power management technique can considerably optimize the power consumption and mitigate thermal issues.

In [84], Lan *et al.* propose a bidirectional channel based NoC architecture to enhance the performance of 2D on-chip communication. This architecture does not support GALS-based communication and addresses only 2D communication. In addition, in order to increase the performance, they exploit ten bi-directional channels instead of five input and five output channels for a 5×5 mesh-based switch. To support this adaptivity, they extend the crossbar to a 10×10 one, add complexity to the arbitration unit, and embed a central channel control module to each router. In contrast, we avoid modifying the main router structure because the BBVC is only responsible for inter-layer communication and there is only one BBVC between two adjacent routers located in different layers. In other words, the proposed inter-layer communication scheme is independent of the intra-layer

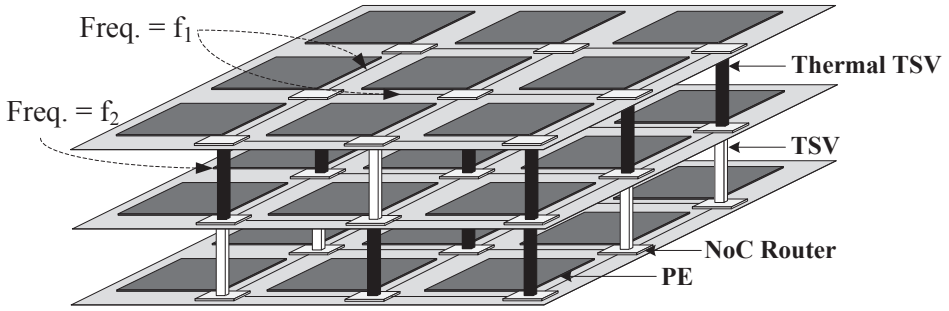


Figure 5.1: Example of a 3D NoC with three 3×3 layers

topology.

In the next section, we present an efficient 3D NoC architecture in which Processing Elements (PEs) in each layer operate synchronously at a same frequency but communicate with PEs in adjacent layers by a bidirectional channel operating at a different (higher) frequency.

5.2 BBVC-Based 3D NoC Architecture

Generally the bandwidth utilization of vertical interconnects in a conventional NoC architecture is low [84] due to congestions and competitions within routers to get access to output ports. Even in a highly loaded NoC, because the input buffers get congested as a result of high competitions, link utilization for some directions frequently drops. Therefore, it can be speculated that many vertical channels are idle during each cycle because of the fixed channel directions. Regarding this fact and the aforementioned pros and cons of 3D NoC design, to reduce the number of interconnect TSVs, an architecture using high-speed BBVCs is proposed. Fig. 5.1 shows the schematic representation of such a system. The main idea of the proposed 3D NoC system is to exploit a bidirectional channel for inter-layer communication operating at a higher frequency compared to intra-layer communication ($f_2 > f_1$) and being capable of dynamically changing the channel direction between routers in neighboring layers based on the real time bandwidth requirement.

For the desired system, a bisynchronous FIFO for Up and Down ports of each 3D router is needed to enable vertical channels to operate at different speeds. In addition, for each vertical port of the routers, a module is required to control the inter-layer communication flow and the direction of a respective BBVC.

To clarify the general process, Fig. 5.2 exemplifies a frequent case of inter-layer communication. In this figure, which depicts the transmission

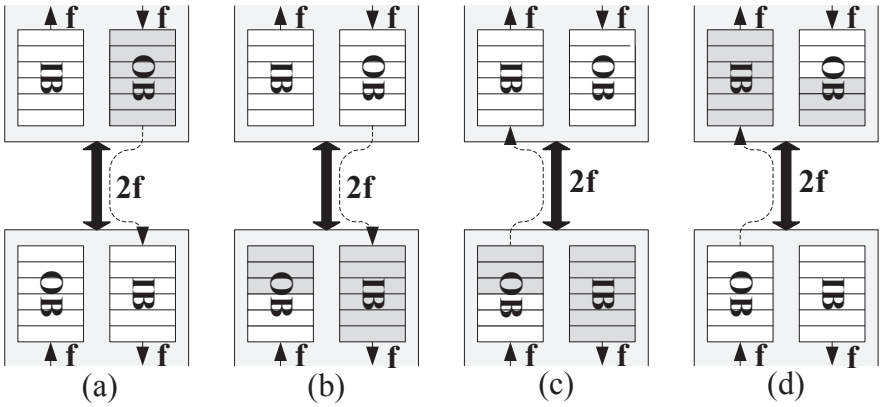


Figure 5.2: Example of the transmission process of BBVC-based communication scheme

process between two routers in adjacent layers, the operating frequency of the BBVC is assumed to be two times faster than the horizontal one. The buffer depth and packet length are set to six and the direction of the vertical channel is from up to down initially. In Fig. 5.2(a), the Output Buffer (OB) of the upper router starts to transmit a packet to the Input Buffer (IB) of the lower router, while the OB of the lower router is storing its output stream with lower speed. As shown in the Fig. 5.2(b), after whole transmission of the packet to the lower router, half of the upward packet has been stored in the lower OB. At this time, depicted in Fig. 5.2(c), the channel direction is reversed and the upward transmission starts. As can be seen from Fig. 5.2(d), while the first half of the stored packet is being sent, the second half will be received and becomes ready to be sent afterwards. It can be concluded from this example that with almost half of the vertical links, the performance of this high-speed BBVC and a pair of typical channels is almost the same. In the following section, we will describe in detail the router architecture and the inter-layer transmission scheme.

5.2.1 Router Architecture

The main objective of an on-chip router is to provide correct transfer of messages between IP cores. Routers in a mesh-type 3D NoC usually have routing logic, arbitration logic, crossbar and seven pairs of unidirectional ports: East, West, North, South, Local, Up, and Down. The Local port establishes communication between the router and its local core, while the other ports of the router are connected to the neighbor routers. Each port has an input buffer for temporary storage. First, each routing module will make a routing decision for each packet and generate a channel request

to the arbiter individually. If the requested channel is available, it means that the corresponding downstream buffer at the neighboring router has enough buffer space. Then the arbiter can begin to allocate these channels to let the granted packets traverse the crossbar to the neighboring routers simultaneously. The crossbar is a large switch fabric which is used to connect each input and output channels. The size of the crossbar in this typical router is 7×7.

To implement a 3D NoC architecture using BBVCs, we modify the configuration of the Up and Down input/output ports and replace their two unidirectional channels by a BBVC, as can be seen from Fig. 5.3. We avoid changing the structure and operating frequency of the routing modules, the arbiter, and the crossbar of a conventional router. As shown in the figure, in order to dynamically adjust the direction of vertical bidirectional channels at runtime, we add a control module to both Up and Down ports to arbitrate the authority of the channel direction. These two bidirectional channels which are composed of in-out type ports are the main difference from the typical router design in which a unidirectional channel employs a hardwired input or output port. The control module of the Up (Down) port communicates with its counterpart in the Down (Up) port of the neighbor router using a token-based communication scheme via *put/get-token* signals.

In order to support different clock frequencies for inter-layer communication, we replace the synchronous input FIFOs of the Up and Down ports by bisynchronous FIFOs. In addition, a bisynchronous FIFO is required for each vertical output port to enable a different (higher) clock frequency for inter-layer communication. If needed, this also offers to have 3D NoC supporting different clock frequency for each layer. Since in this case the number of slots in the bisynchronous FIFO does not affect the general architecture, to reduce the area overhead caused by two additional output FIFOs, the FIFO length can be reduced. For example, if 8-slot FIFOs are used for non-vertical input ports; two bisynchronous FIFOs with length of 4 can be utilized for vertical input and output ports.

5.2.2 Inter-layer Transmission Scheme

According to the OSI model, the network physical layer is responsible for providing mechanical and electrical media to connect different entities at bit level. In the present work, this layer corresponds to the communication between routers. The physical interface between routers for intra-layer communication has been exemplified in Fig. 5.4. The physical data bus width must be chosen according to the available routing resources and available memory to implement buffering. The output port in the example is composed by the following signals: (i) *tx*: control signal indicating data availability; (ii) *data_out*: data to be sent; (iii) *credit_i*: control signal indicating

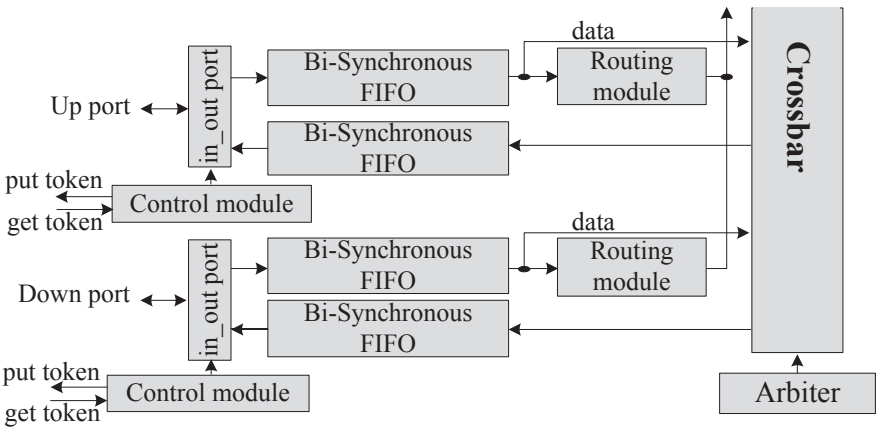


Figure 5.3: Up and Down ports of the proposed router architecture supporting bidirectional bisynchronous vertical channels

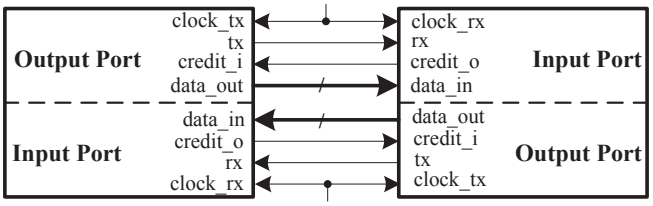


Figure 5.4: Example of physical interface between routers

space availability in the target buffer; (iv) *clock_tx*: clock signal. The input port in the example is composed by the following signals: (i) *rx*: control signal indicating data availability; (ii) *data_in*: data to be received; (iii) *credit_o*: control signal indicating space availability in the input buffer; (iv) *clock_rx*: clock signal. In this protocol, when a router needs to send data to a neighbor router, once the neighbor router assert the *credit_o* signal indicating that there is at least one empty slot in its buffer, it puts the data in the *data_out* signal and asserts the tx signal synchronized with *clock_tx*, then the neighbor router stores the data from the *data_in* signal, and the transmission is complete.

For inter-layer communication, the physical interface is different but it still works based on the described protocol. Fig. 5.5 discloses details of the proposed communication scheme and signals between the two adjacent routers in different layers. As can be seen from the figure, there is only one data bundle shared between the routers. In the proposed scheme, the bidirectional channel is managed by a finite state machine. To dynamically

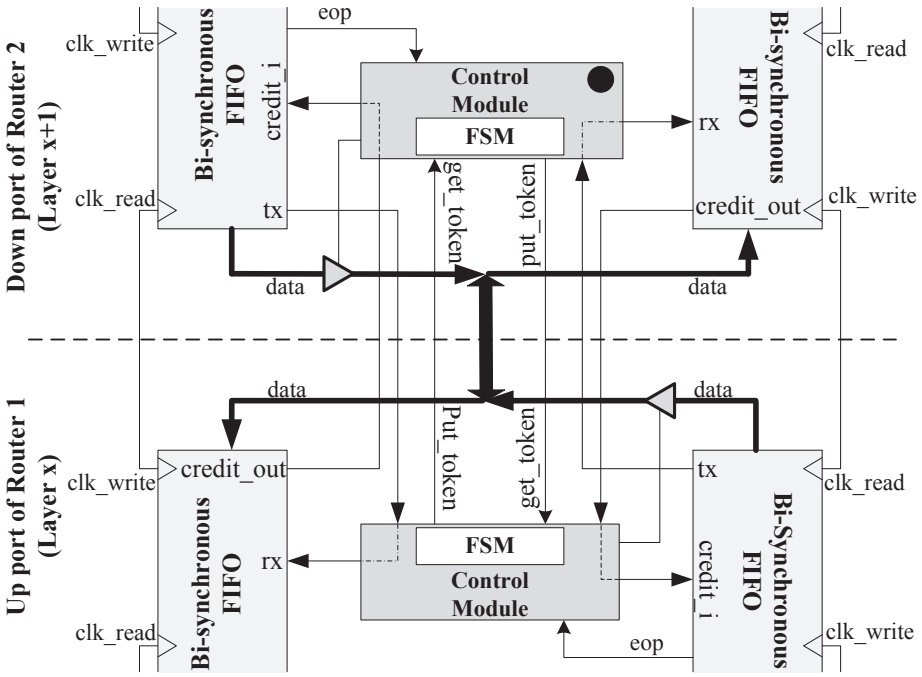


Figure 5.5: Inter-layer data transmission scheme

change the channel direction, a token-passing-based technique is utilized, therefore in addition to described control signals; one extra signal for both directions is needed to pass the token between control modules. It should be noted that since the vertical ports are benefiting from bisynchronous FIFOs, they can transmit data with a higher speed compared to intra-layer communication.

In the proposed communication scheme, the control module possessing the token, in the case there are one or more available packets in its input port FIFO, starts to send the packet. After sending the last flit indicated by the *eop* signal and if there is a pending request from the other router, it passes the token to the neighboring module via the *put_token* signal, changes its status to *receive_mode*, and disconnects the *data_out* port from the bidirectional link. Next, the control module of the adjacent router changes its mode to *send_mode*, connects the received *credit_o* coming from input port buffer of the opposed router to *credit_i* of its output port buffer, and starts the transmission. Note that because passing the token is based on round-robin policy (fair arbitration), there is no starvation and accordingly no deadlock (depending on routing algorithm) for the inter-layer communication.

5.3 Forecasting-Based Dynamic Frequency Scaling

As illustrated via a simple scenario in Fig. 5.2, the BBVCs can operate at a higher frequency compared to the horizontal channels. We showed that, in the worst case, when a BBVC has the maximum traffic load and operates with two-times faster clock frequency; it can offer the same performance as two unidirectional channels. If we always set the inter-layer frequency level based on the worst case scenario, there will be no improvement in terms of power consumption. To this end, we propose a dynamic power management technique to dynamically determine the vertical channel frequency level. The DPM technique is based on a distributed forecasting-based policy, where each BBVC predicts its future communication workload and the required operating frequency based on the analysis of the prior traffic, thanks to the bi-synchronous FIFOs for enabling the DFS technique.

In order to assess the communication traffic characteristics of a channel several potential network parameters can be utilized. In this case, we employ the link utilization parameter as the network load indicator because of its simplicity and efficiency. The Link Utilization (LU) parameter is defined as [142]

$$LU = \frac{\sum_{t=1}^{H} A(t)}{H}, \quad 0 \leq LU \leq 1 \tag{5.1}$$

where $A(t)$ is equal to 1, if the traffic passes through the link i in the cycle t and 0 otherwise, and H is the window size. The link utilization is a direct measure of the traffic workload.

In the proposed technique, the DPM unit uses the LU parameter for measuring the past communication traffic. Based on this analysis, the LU of the next period and the required frequency level are determined. To make the forecasting formula reliable, we use an exponential smoothing function. This is a simple and popular forecasting (weighted average) formula, which is commonly used in programming and inventory control science and is defined as [150][131]

$$LU_{predict} = LU_{past} + \alpha \times (LU_{actual} - LU_{past}) \tag{5.2}$$

where α is the forecasting weight, $LU_{predict}$ is the predicted link utilization, LU_{actual} is the actual link utilization in the current history period, and LU_{past} is the predicted link utilization in the previous history period. The accuracy of the prediction is a function of α, and hence, its value should be selected such that the prediction error is minimized (see Section 5.4).

The network traffic profile has short-term and long-term fluctuations. The proposed technique in this work filters out the short-term traffic fluc-

Algorithm 1 *Forecasting-Based Dynamic Frequency Scaling*

1: $LU_{predict} = LU_{past} + \alpha \times (LU_{actual} - LU_{past})$;
2: **if** $(LU_{predict} < Threshold_1)$ **then**
3: $Freq\_Level = 1$;
4: $New\_BBVC\_Freq = Frequency\_table[Freq\_Level]$;
5: **if** $(Freq\_Level = Intra\_layer\_Freq\_Level)$ **then**
6: Syn/Bi-Syn_Mode = Syn;
7: **else**
8: Syn/Bi-Syn_Mode = Bi-Syn;
9: **end if**
10: **else if** $(LU_{predict} > Threshold_1$ **and** $LU_{predict} < Threshold_2 = 0)$ **then**
11: Freq_Level=2;
12: New_BBVC_Freq = Frequency_table[Freq_Level];
13: **if** $(Freq\_Level = Intra\_layer\_Freq\_Level)$ **then**
14: Syn/Bi-Syn_Mode = Syn;
15: **else**
16: Syn/Bi-Syn_Mode = Bi-Syn;
17: **end if**
18: ...
19: ...
20: **else if** $(LU_{predict} > Threshold_{n-1} = 0)$ **then**
21: Freq_Level=n;
22: New_BBVC_Freq = Frequency_table[Freq_Level];
23: **if** $(Freq\_Level = Intra\_layer\_Freq\_Level)$ **then**
24: Syn/Bi-Syn_Mode = Syn;
25: **else**
26: Syn/Bi-Syn_Mode = Bi-Syn;
27: **end if**
28: **end if**
29: $CT_{past} = CT_{predict}$;

tuations, adapting the frequency level of a BBVC based on the long-term traffic profiles. Based on the prediction, the controller decides to increase, decrease, or keep the same frequency level. The pseudo-code for our proposed forecasting-based DFS policy for the case of n frequency levels is shown in Algorithm 1.

In this algorithm, when the frequency level is equal to the intra-layer frequency level, it means that the inter-layer and intra-layer communications are using the same clock frequencies. In other words, the bi-synchronous FIFOs are not needed. To avoid the power/performance overhead of the bi-synchronous FIFOs in such cases, we exploit Reconfigurable Synchronous/Bi-Synchronous (RSBS) FIFOs presented in [135] instead of simple bi-synchron-

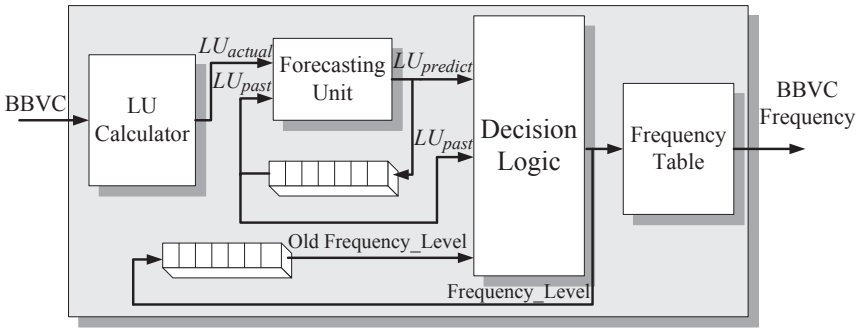


Figure 5.6: The hardware diagram of the forecasting-based DFS policy

ous FIFOs. When the read and write clock signals have the same frequency level; the controller changes the RSBS FIFO mode to synchronous, otherwise asynchronous.

The hardware realization of the proposed forecasting-based DFS policy which relies on the local link is shown in Fig 5.6. Because the communication overhead of relying on global information is avoided, a simple hardware implementation can be invoked. To measure the link utilization (LU), a counter at each BBVC gathers the total number of flits that are passed from the link in each history interval. The Forecasting Unit uses the calculated LU and previous predicted LU from the previous interval (LU_{past}) to predict the new LU ($LU_{predict}$) for the next H cycle interval. A register stores the $LU_{predict}$ to be used as the LU_{past} in the next interval. The LU calculation is cycle-based using simple counters, but the forecasting-based algorithm which is the major part of the DFS technique is window-based. The algorithm only comprises number of conditional statements being equal to number of frequency levels and being called for each timing window (H) consisting of many cycles. In addition, the execution time of the forecasting-based algorithm is predictable owing to its non-iterative structure.

To reduce the hardware overhead, we set α to 3/4, which is very close to the optimal values (see Section 5.4) of this parameter for the traffic profiles used in this work. We implemented the division and multiplication operations by right and left shifts, respectively. It should be noted that the number of DFS units is equal to the number of vertical links (i.e., if there are k layers, and each of which contains $m \times p$ nodes, (k-1)$\times m \times p$ DFS units are needed).

The Decision Logic unit determines the required frequency level using $LU_{predict}$, LU_{past} and the required frequency level of the previous history window. Finally, note that we simplify the division and multiplication operations by setting H value as power-of-two numbers. Otherwise, the overhead of the DFS unit will become slightly larger.

5.4 Experimental Results

To assess the efficiency of the proposed BBVC-based 3D NoC, several simulations were conducted. We have simulated the Typical-3D-NoC, BBVC-3D-NoC, and DFS-BBVC-3D-NoC to characterize their area, latency and power consumption. To demonstrate the amount of power savings and the negligible performance overhead of the proposed inter-layer communication scheme, a cycle-accurate NoC simulation environment was implemented along with three different inter-layer channel designs (Typical, BBVC-based, and DFS-BBVC-based) in VHDL. Wormhole packet switching is adopted. The implemented Symmetric 3D NoC consisted of typical state-of-the-art routers including input buffers, centralized control logic, with a round-robin arbiter and Dimension-Order Routing (DOR), credit based flow control, internal crossbar to connect input to output ports and priority-based output scheduling. Owing to utilizing DOR-based algorithm, deadlock avoidance is guaranteed. In this architecture routers have at most seven input/output ports.

5.4.1 TSV Area Footprint Analysis

The area of the BBVC-based communication scheme was computed once synthesized on CMOS 130, 90, and 65nm standard cells by Synopsys Design Compiler. Different bandwidths are used to illustrate the scalability of the architecture. In Fig. 5.7, the first and second groups of bars show the area savings when using the proposed communication scheme for a 5μm$\times5\mu$m TSV pads and an 8μm pitch for 32-bit and 64-bit links, respectively. It can be seen that the proposed communication scheme can significantly reduce the area footprint of TSVs in 3D ICs. As technology scales, the savings in area increase as a result of lower area footprint of the control module logic.

As discussed before, it is a common practice to increase TSV pad area to mitigate thermal issues and enhance the system reliability. To this end, the area saving was calculated for the 10μm$\times10\mu$m TSV pad dimensions and 16μm pitch, while TSV dimensions were kept unchanged. The third and fourth groups of bars in Fig. 5.7 illustrate the area savings for 32-bit and 64-bit links in the case when the proposed BBVC-based communication scheme is applied. As shown in the figure, compared to the previous non fault-tolerant case, the design is more area efficient. For instance, the area savings are greater than 47% for an implementation based on a 65nm library and 64-bit links.

5.4.2 Synthetic Traffic Analysis

Generally the area savings due to reduction in the number of TSVs come at an expense of reduction in performance. For example, for the serialization

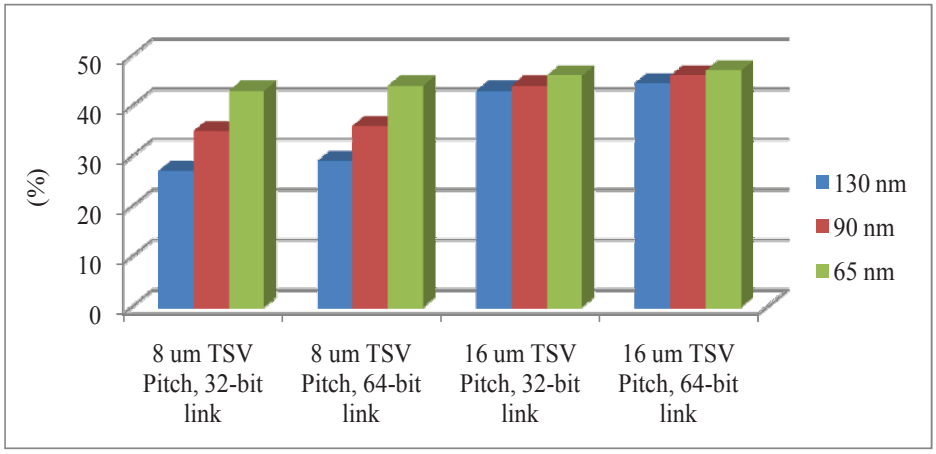


Figure 5.7: Percentage of area saving with BBVC-based communication scheme for different TSV pitches and link sizes for 65nm, 90nm, and 130nm technologies

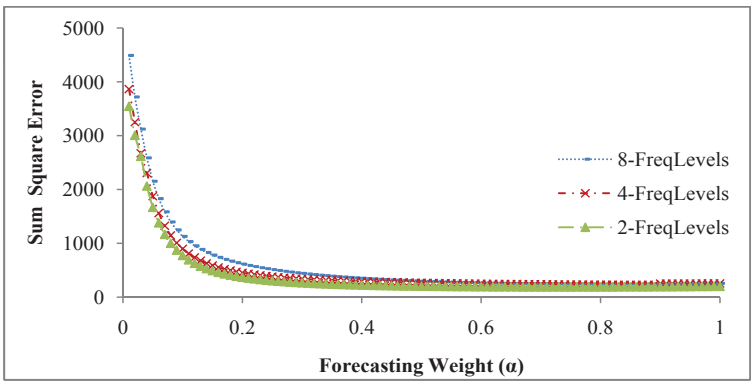
technique presented in [121], as the degree of serialization increases, and the number of wires in the links are reduced, the performance degrades. In synthetic traffic analysis, the 3D NoC of our simulation environment consists of 3×3×3 nodes connected as a symmetric 3D mesh. The performance of the network was evaluated using latency curves as a function of the packet injection rate (i.e., the number of packets injected into the network per cycle). The packet latency was defined as the time duration from when the first flit is created at the source node to when the last flit is delivered to the destination node. For each simulation, the packet latencies were averaged over 500,000 packets. It was assumed that the buffer size of each synchronous FIFO was eight fits, and the data width was set to 64 bits. To support the bisynchronous inter-layer communication, 6-stage modular synchronizing FIFOs presented in [113] were exploited. The FIFO had a maximum read and write clock speed of 2.33GHz at full throughput. A 500 MHz clock frequency was applied to the typical NoC and for the intra-layer communication of the DFS-BBVC-Based NoC, resulting in a maximum transmission rate per link of 8 Gbps. For the inter-layer communication, following frequency levels were available to be assigned based on the DFS decision: 250MHz, 500MHz, 750MHz, and 1 GHz. Note that the clock cycle used to measure average packet latency, is based on the fixed intra-layer communication frequency.

To see the impact of α on the prediction accuracy, we have used the sum square error (SSE) [17]

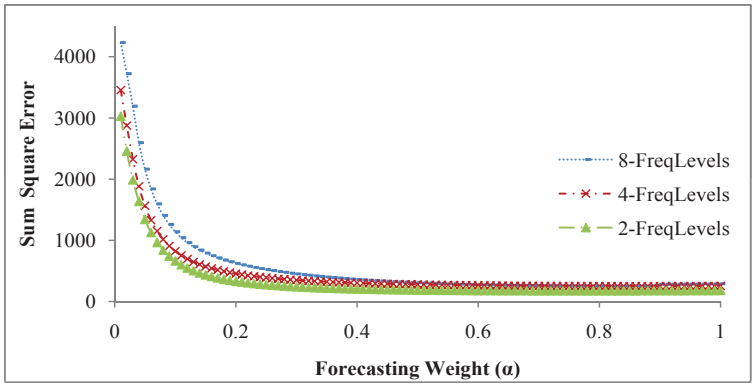
$$SSE = \sum_{i=1}^{k} \sum_{j=1}^{H} (y_{ij} - \widehat{y}_{ij})^2 \tag{5.3}$$

where y_{ij} is the actual required frequency level, $\widehat{y}_{ij}$ is the predicted required frequency level, and k is the total number of the prediction windows that the simulation has been run. To perform the simulations, we used an XYZ wormhole routing algorithm [137] under uniform, hotspot 10% and Negative Exponential Distribution (NED) [130] traffic patterns. In the uniform traffic pattern, a node sends a packet to any other nodes with an equal probability while in the hotspot traffic pattern, messages are destined to a specific node (the node at (2, 2, 2)) with a certain (10% higher than average) probability and are otherwise uniformly distributed. The NED is a synthetic traffic model based on Negative Exponential Distribution where the likelihood that a node sends a packet to another node exponentially decreases with the hop distance between the two cores. This synthetic traffic profile accurately captures key statistical behavior of realistic traces of communication among the nodes. As shown in Fig. 5.8, for the uniform traffic, the optimum forecasting weights for *2FreqLevels*, *4FreqLevels*, and *8FreqLevels* were equal to 0.86, 0.76 and 0.82, respectively while for the NED traffic, the optimum forecasting weights for *2FreqLevels*, *4FreqLevels* and *8FreqLevels* were equal to 0.85, 0.77 and 0.82, respectively. The optimum forecasting weights for the hotspot 10% traffic were equal to 0.82, 0.79 and 0.84 for *2FreqLevels*, *4FreqLevels* and *8FreqLevels*, respectively. The error, however, is not much higher than the minimum value if we select α as 0.75 for all the cases. This can simplify the implementation of the DFS unit. Fig. 8.6 shows that our DFS-BBVC-3D-NoC with half of vertical channels has still negligible performance overheads when compared to the Typical-3D-NoC under all uniform, NED, and hotspot 10% traffic patterns. The overhead originates mainly from the prediction error. Note that the proposed architecture still significantly outperforms 2D NoC architectures in terms of system power and performance. It can be seen that as the injection rate increases, the performance overhead slightly grows.

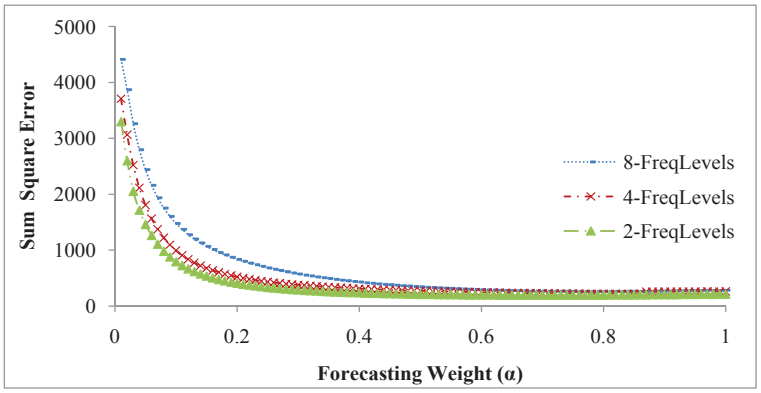
To estimate the power consumption, we adapted the high level NoC power simulator presented in [58] to support the 3D NoC architectures. The power simulator is a precise NoC power estimation model extracted from Synopsys PrimePower which is based on buffer reception rates. To extend the high-level simulator, we used the vertical interconnect parameters and equations presented in [124]. The average power consumption of the routers (getting average of each router and its connected links) which are based on 35nm standard CMOS technology are presented in Fig. 5.10. As the results reveal, the average power consumption of the routers using DFS-based BBVCs are considerably lower than those of the correspond-



(a) Under uniform traffic profile

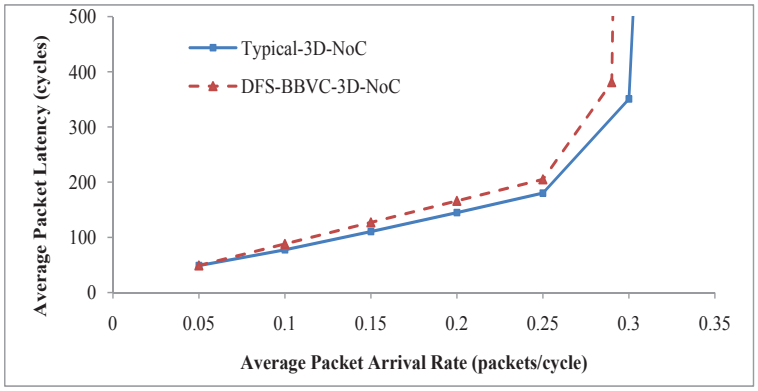


(b) Under NED traffic profile

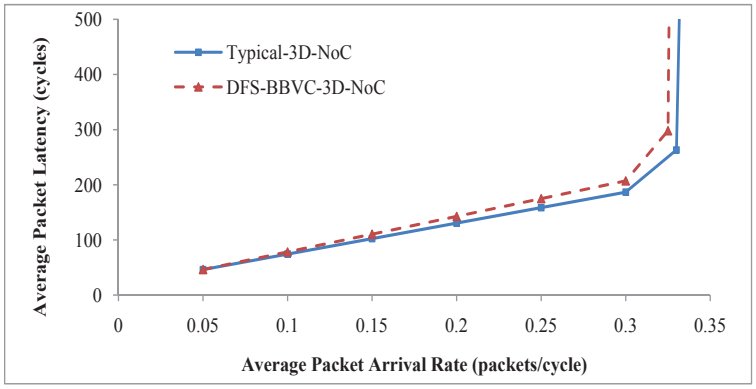


(c) Under hotspot 10% traffic profile

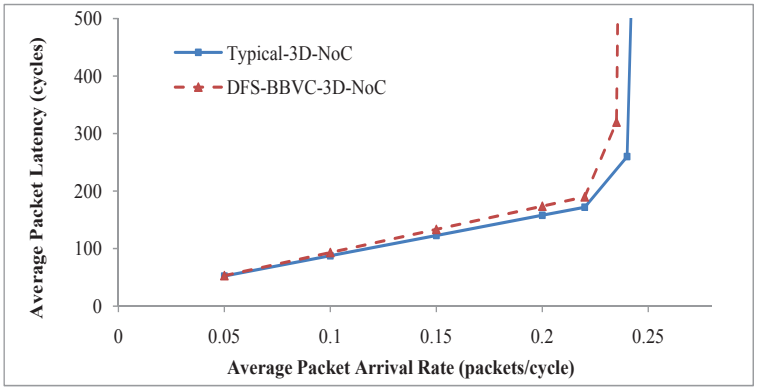
Figure 5.8: Forecasting sum square error as a function of α



(a) Under uniform traffic profile

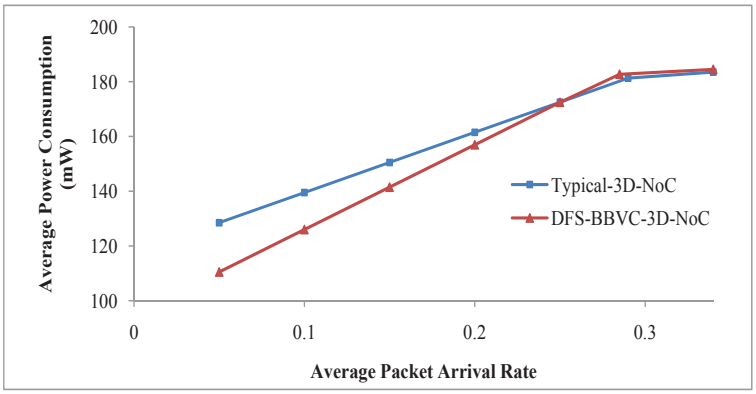


(b) Under NED traffic profile

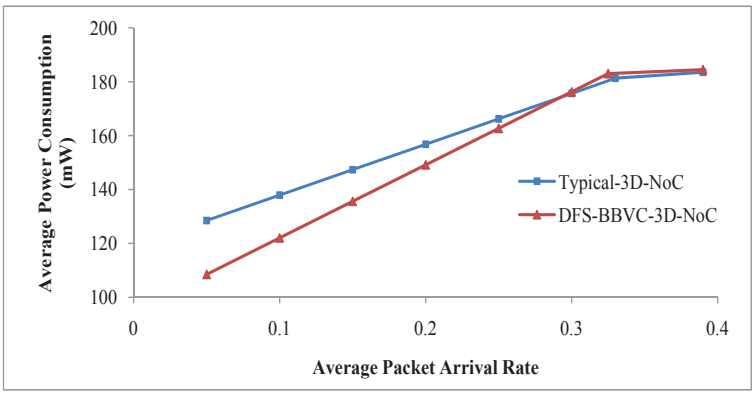


(c) Under hotspot 10% traffic profile

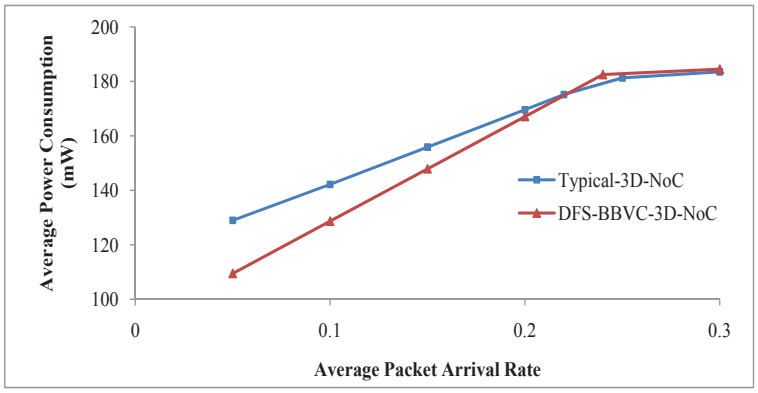
Figure 5.9: Latency versus average packet arrival rate results



(a) Under uniform traffic profile

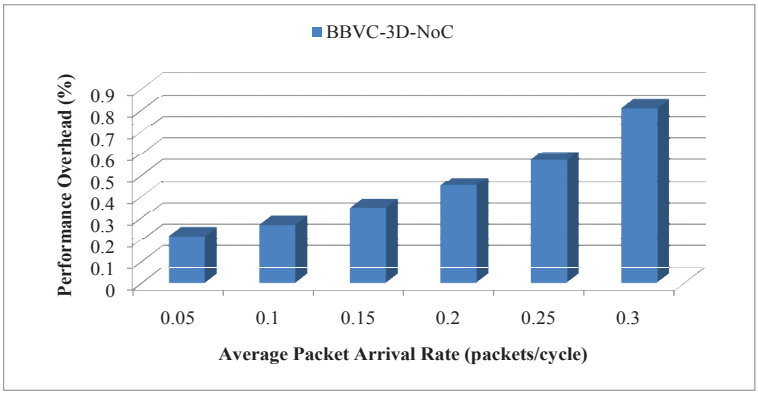


(b) Under NED traffic profile

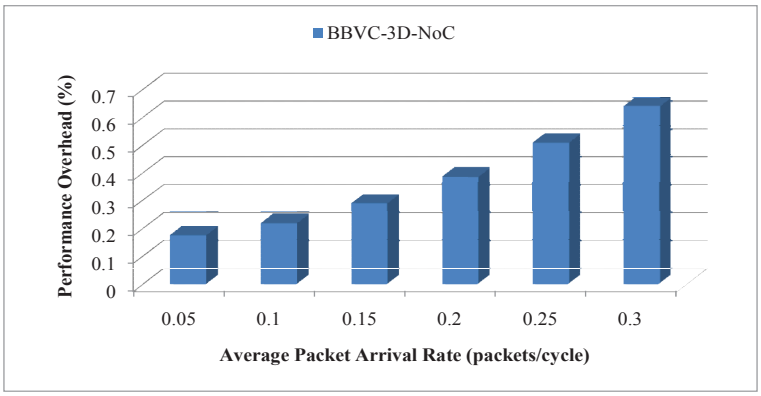


(c) Under hotspot 10% traffic profile

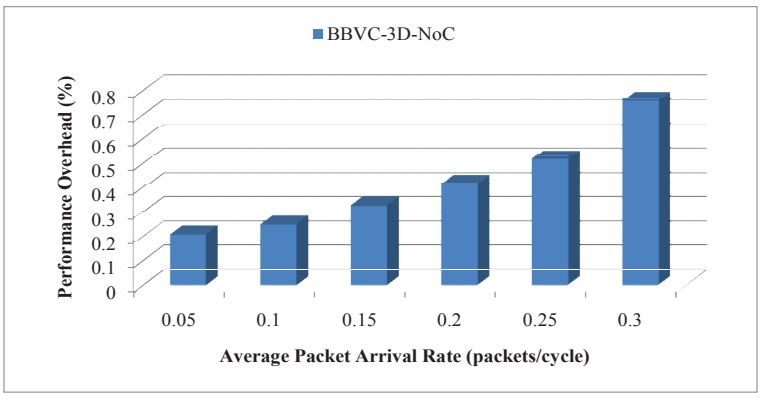
Figure 5.10: Average power consumption versus average packet arrival rate results



(a) Under uniform traffic profile

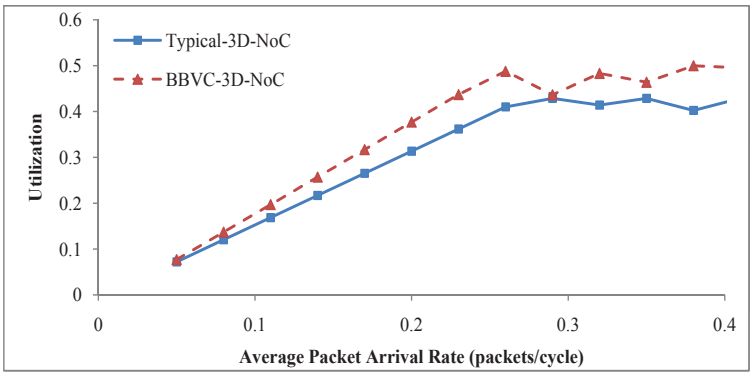


(b) Under NED traffic profile

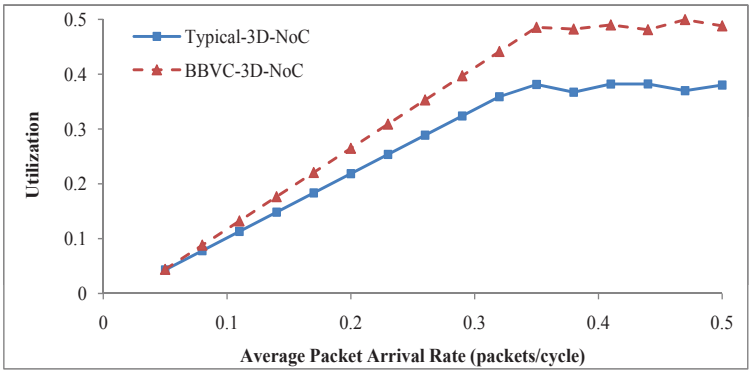


(c) Under hotspot 10% traffic profile

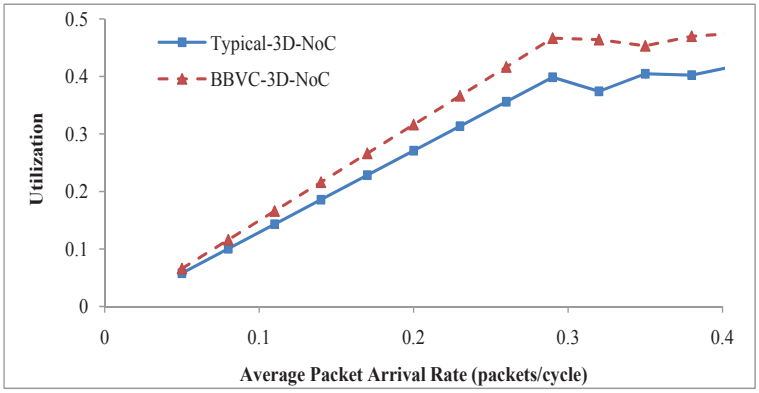
Figure 5.11: % performance overhead versus average packet arrival rate results



(a) Under uniform traffic profile



(b) Under NED traffic profile



(c) Under hotspot 10% traffic profile

Figure 5.12: Bandwidth utilization versus average packet arrival rate results

ing conventional routers at low traffic loads. The reason is that when the forecasting-based DFS technique is used, vertical links and the connected buffers operate at a lower frequency level, leading to some power saving. As the traffic load increases the difference between the average power consumptions of the routers decreases till they eventually become almost the same at the congestion injection rate. For all the synthetic traffic patterns, there is a slightly higher power consumption at the saturation point. The reason is the power consumption of the extra components (proposed BBVC Control Module and forecasting-based DFS Controller). However, summing up the power consumption at each injection rate leads to power saving of 17% in the overall router power consumption and up to 39% savings in vertical channels power consumption.

To demonstrate the negligible performance drop of the proposed channels, we have also simulated a typical BBVC-based system which does not benefit from the DFS technique. The same configuration was used to extract the average packet latency (APL) and bandwidth utilization of the network. In this case, the clock frequency for intra-layer and inter-layer communication was set to 1 and 2 GHz, respectively. Fig. 5.11(a), 5.11(b), and 5.11(c) show the reduction in overall 3D-NoC performance for the synthetic traffic patterns, with varying average packet arrival rates. The figures reveal that the BBVC-based communication scheme has negligible performance overhead when compared to the Typical-3D-NoC under these three traffic patterns. It can be seen that as the injection rate increases, the performance degrades slightly compared to the typical case due to the slight latency of changing a BBVC direction.

The bandwidth utilization experiments versus average packet arrival rate under synthetic patterns are presented in Fig. 5.12. We can find that BBVC-based 3D NoCs always have better bandwidth utilizations due to their flexibilities. It can be concluded that many vertical channels are idle during each processing cycle in Typical-3D-NoC because of the fixed channel direction.

5.4.3 Videoconference Application

For realistic traffic analysis, the encoder part of video conference application with sub-applications of H.264 encoder, MP3 encoder and OFDM transmitter was used. The size of video stream used for simulation was 300×225 pixels and each pixel consisted of 24 bits. Therefore, each video frame was composed of 1.62 Mbits and could be broken into 8400 data packets each of size 7 flits including the header flit. The data-width was set to 32-bits. The application graph with 26 nodes is shown in Fig. 5.13. In this application, the *Mem_in_Video* component generates 8400 packets for one application cycle equivalent to the one video frame. The frame rate for the video stream

was 30 frames/second and the data rate for the video stream was 49336 kbps. The application graph consists of processes and data flows; data is, however, organized in packets. Processes transform input data packets into output ones, whereas packet flows carry data from one process to another. A transaction represents the sending of one data packet by one source process to another, target process, or towards the system output. A packet flow is a tuple of two values (P, T). The first value 'P' represents the number of successive, same size transactions emitted by the same source, towards the same destination. The second value 'T' is a relative ordering number among the (packet) flows in one given system. For simulation purposes, all possible software procedures are already mapped within the hardware devices.

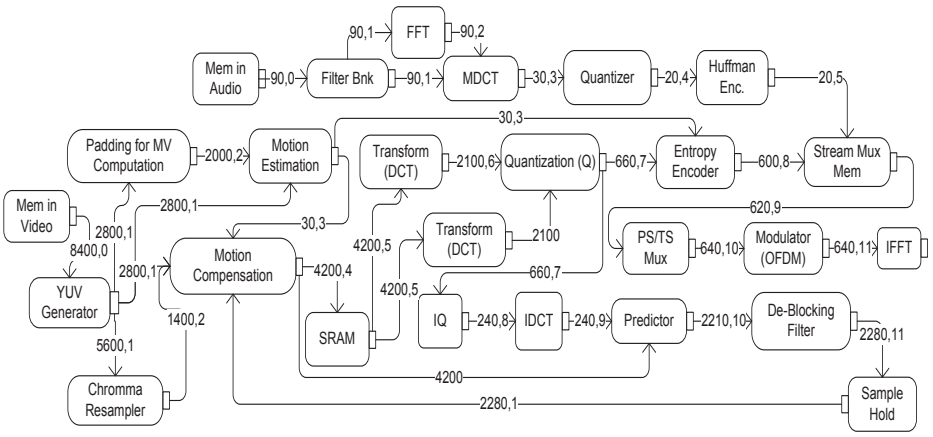


Figure 5.13: Communication trace of encoding part of a H.264 video conference

We applied the proposed power-aware BBVC-based scheme to the encoder and compared it with a similar system which did not benefit from the proposed BBVCs. The encoder system was modeled and mapped on a 3×3×3 3D NoC shown in Fig. 5.14. The central node (1, 1, 1) was used as a platform agent for monitoring purposes. A 200 MHz clock frequency was applied to the typical NoC and the intra-layer communication of the DFS-BBVC-based NoC. For the inter-layer communication, the following frequency levels were applied: 100, 200, 300, and 400 MHz. The buffer sizes of the synchronous and bi-synchronous buffers were set to eight and six fits, respectively, while the forecasting weight (α) was selected as 0.75. Deterministic routing algorithm (XYZ) has been selected to avoid deadlocks.

The power and latency results which were estimated the same as the synthetic traffic analysis are given in Table 5.1. The figures given in these tables demonstrate almost 46% and 18% reduction in terms of number of TSVs and power consumption, respectively. As can be seen, the performance

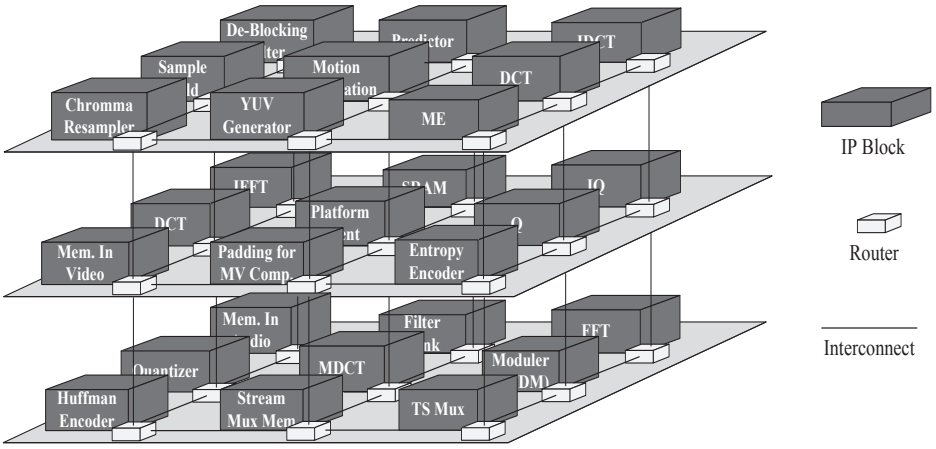


Figure 5.14: Partition and core mapping of the H.264 encoder

NoC Architecture	**Average Power Consumption** (mW)	**Average Packet Latency (cycles)**	**Number of TSVs**
Typical 3D NoC	61.8	157	2394
DFS-BBVC-3D-NoC	50.5	161	1296

Table 5.1: Power consumption and average packet latency running H.264 traces

overhead of the proposed forecasting-based technique is negligible.

5.4.4 Hardware Overhead

The area of the main components was computed once synthesized on CMOS 65nm LPLVT STMicroelectronics standard cells using Synopsys Design Compiler. Different subcomponents were also synthesized to illustrate the area overhead of the controllers and the extra hardware. The layout area of the typical 7-port NoC router, the proposed router, and the utilized controllers are listed in Table 5.2. The figures given in the table reveal, the area overheads of the proposed forecasting-based DFS unit and BBVC control module are negligible.

Component	**Area**(μm^2)
Typical-3D-NoC 7-Port Router	43506
DFS-BBVC-3D-NoC 7-Port Router	46722
BBVC Control Module	304
Forecasting-Based DFS Unit	1392

Table 5.2: Hardware implementation details

5.5 Summary

In this chapter, we proposed a power and area efficient 3D NoC architecture exploiting bidirectional bisynchronous vertical channels. By replacing a pair of unidirectional vertical channels by a power-aware high-speed bidirectional channel, this communication scheme can remarkably reduce the number of TSVs and considerably mitigate high power densities and peak temperatures. To compensate the performance loss caused by reduction in the number of vertical links, a GALS-Based approach for vertical communication was utilized. In this architecture, the high-speed nature of the vertical links in 3D ICs enables the system to support a higher frequency for vertical channels. Moreover, we presented a technique to manage power consumption of the inter-layer communication using a forecasting-based DFS technique. The approach makes use of the link utilization in predicting the inter-layer communication traffic. Based on the predicted traffic, the frequency level of the respective vertical channel may be increased, decreased, or remain the same. Our extensive simulations with synthetic and real benchmarks, including the one with an integrated videoconference application, demonstrate that the proposed architecture can astonishingly reduce interconnect TSV area footprint (up to 47%) and NoC power consumption (upto 18%) with a negligible performance exacerbation compared to a symmetric 3D NoC. Compared to the serialization technique, the proposed architecture shows considerable improvements in terms of power consumption and average packet latency.

Chapter 6

Efficient Hybridization Scheme for 3D NoC

As discussed in Chapter 3, the 3D Symmetric NoC architecture can be simply created by adding two additional physical ports to each router for inter-layer communication [19]. However, due to its major inherent drawbacks, the stacked mesh 3D NoC (Hybrid 3D NoC-Bus mesh) architecture was presented in [88] to overcome the mentioned 3D Symmetric NoC challenges. By using the stacked mesh architecture, six-port router is required instead of seven ports for typical 3D NoC router. However, the additional input port still imposes considerable extra logic to a NoC router, especially when complex routers with support of virtual channels and load management schemes are required.

In this chapter, we propose a novel hybridization scheme for inter-layer communication using an efficient 5-input router to enhance the overall system power, performance, and area characteristics of the existing Hybrid 3D NoC-Bus mesh architecture. By defining a rule for routing algorithms called *LastZ*, the proposed area-efficient architecture decreases the overall average hop count of a NoC-based system compared to the existing architectures. In addition, an extremely low-cost wrapper is presented to provide the compatibility of the proposed vertical communication scheme with existing network interfaces.

6.1 LastZ Routing Policy

The 3D NoCs are extension to the 2D NoC architecture. For each NoC router of mesh topology, two extra ports are needed resulting a 7×7 crossbar instead of the 5×5 crossbar of the 2D mesh architecture. Since, the crossbar power increases quadratically with number of ports, the power consumption of a 3D router is much higher than a 2D router [136]. The solution to the

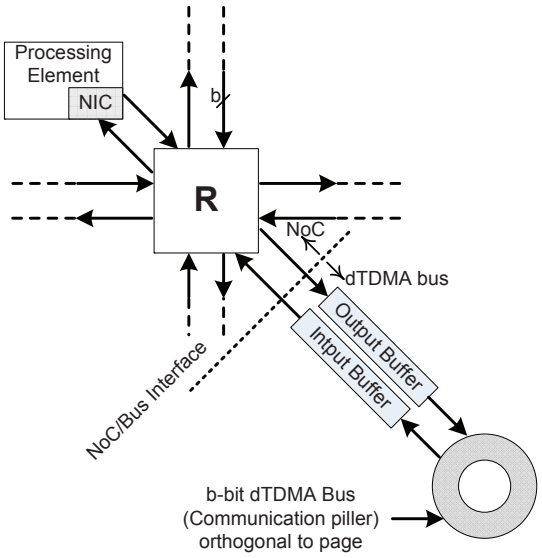


Figure 6.1: High level overview of the Hybrid 3D NoC-Bus mesh router of a pillar node [88]

power consumption and inefficient utilization of vertical links for a 3D router has been proposed by Li *et al.* [88]. The proposed architecture is Hybrid 3D NoC-Bus mesh architecture. The dynamic Time-Division Multiple Access (dTDMA) bus was used as a communication pillar as shown in Fig. 6.1. In this architecture, an interface between the dTDMA pillar (vertical link) and the NoC router must be provided to enable seamless integration of the vertical links with the 2D network within the layers. An extra physical channel is added to the router for the vertical communication. This channel has its own dedicated buffers, and is indistinguishable from the other channels. Bus arbiters should be placed in the middle layer to keep wire lengths as uniform as possible. Due to one hop vertical communication and 6×6 routers, proposed architecture offers better characteristics in terms of power consumption and latency. The issue with this architecture is that it still has a large crossbar and low throughput due to inefficient hybridization of NoC and Bus media.

A 6×6 router has many disadvantages over its 5×5 counterpart. For instance, based on the reported results in [78] for 90nm CMOS technology, a 6-port router incurs around 36% area and 20% power overheads compared to a 5-port one. On the other hand, the packet delay to cross the router increases because more input channels compete to get access to the target output port. The larger crossbar also increases the router critical path and thereby reduces the router maximum operating frequency.

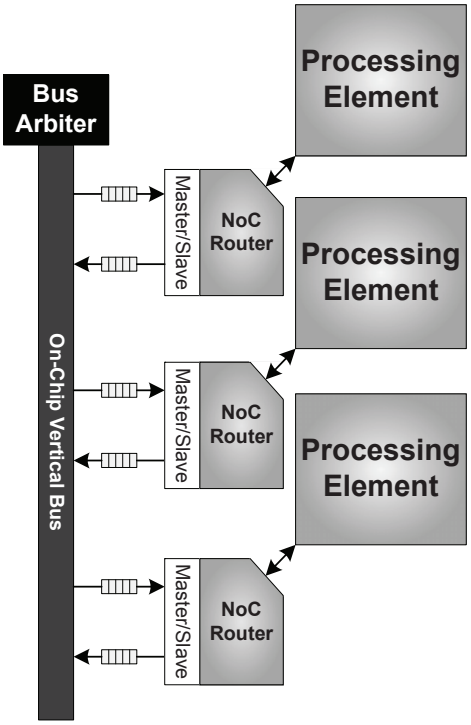


Figure 6.2: Side view of the conventional Hybrid 3D NoC-Bus mesh architecture

Our investigation shows that a straightforward hybridization of two different communication media (i.e. NoC and Bus) without considering their intrinsical characteristics is not an efficient strategy. Fig. 6.2 shows the conventional hybridization style of a Hybrid 3D NoC-Bus mesh-based system. In this architecture, stacked routers in different layers are connected to a vertical bus. The routers are able to serve as either a master or a slave depending on the arbitration decision. The bus protocol is based on the work presented in [88]. The dTDMA policy is used for arbitration and the bus arbiters are placed in the middle layer to keep wire distances as uniform as possible. TSVs are used to implement interlayer control signals. The number of control signals grows with the number of layer. Surprisingly, as will be shown later, just by following one basic rule in routing algorithm policy, it is possible to remove one input port and substitute 6×6 routers with 5×6 ones.

6.1.1 LastZ Rule

In 3D NoC, the packet routing process is classified into two different categories: intra-layer routing and inter-layer routing. For the Hybrid 3D NoC-

Bus mesh architecture, the intra-layer packet routing is multi-hop because traditional NoC architecture is utilized for communication. In multi-hop communication, packet routing plays a crucial role because there are many minimal or non-minimal paths to send a packet from a source to a destination. In contrast, for inter-layer communication, the Hybrid 3D NoC-Bus mesh architecture benefits from bus-based one-hop communication. In this work, we define a rule which we call *LastZ*.

Definition 1 (LastZ): In the wormhole packet switching approach messages are sent by means of packets composed of flits (flow control unit) [131]. A 3D routing algorithm is *LastZ*-based if the intra-layer routing process is completed before the inter-layer routing. In other words, in a *LastZ*-based routing algorithm, when a node N_{source} sends a flit to a node $N_{destination}$, the flit will first travel along the X or Y direction (statically or adaptively) in N_{source} dimension until the flit reaches the pillar (vertical inter-layer bus) having the same x- and y- coordinates as the destination node ($Flit_{xy}$=$Pillar_{xy}$). If a TSV sharing technique [91] is used to provide one bus for a group of destination nodes, the condition can be revised as $Flit_{xy}$=$Pillar\_Group_{xy}$. Then the flit will traverse the last hop in the Z direction.

As will be shown later, this rule is considerably beneficial to improve system characteristics. It is noteworthy that this rule does not force significant limitations to a routing algorithm owing to the one-hop bus-based vertical communication. Many of recently proposed architectures benefit from routing algorithms being by default *LastZ*-based such as [169][167][128][71][134].

6.2 Proposed Hybridization Architecture

Based on the defined *LastZ* rule, assume that a packet, after complete intra-layer routing, has reached the destination pillar (vertical bus). In this case, it is obvious that one of the connected routers to the vertical bus is the last router (hop) for delivering the packet to the target processing element (PE). Based on the fact that the destination is already known, it is not wise to send the packet again to the respective router for the routing decision making. Instead, it is more efficient to deliver the packet to the connected PE directly.

The explained scenario was the motivation to propose a new hybridization scheme for connecting components to a vertical bus. The proposed architecture is shown in Fig. 6.3. As can be seen in the figure, it is practical to establish a more efficient inter-layer communication scheme without adding any extra workload and hardware to bus arbiters. Based on the proposed hybridization architecture, routers just serve as masters to initiate the transaction and PEs play the slave role and via the intermediate buffers

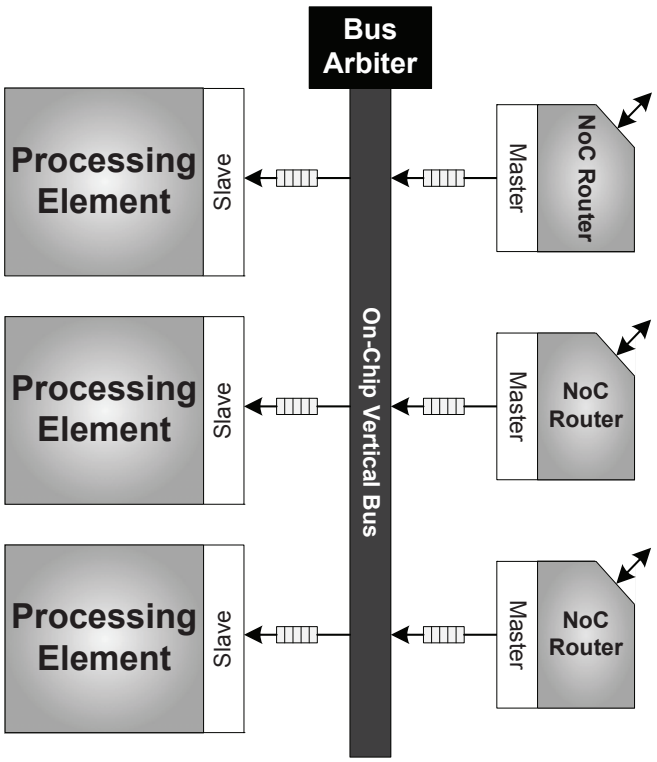


Figure 6.3: Side view of the proposed Hybrid 3D NoC-Bus mesh architecture

directly receive their own packets.

In the proposed communication scheme, routers do not need any input port for the Up/Down direction, because they are bypassed by intermediate buffers. Fig. 6.4 shows the high level view of the proposed Hybrid 3D NoC-Bus mesh architecture. As can be seen in the figure, the intermediate input buffer which was used as an interface between a router and a dTDMA bus, in this architecture connects a PE directly to the vertical bus. Bypassing routers enables a 3D NoC to utilize a 5×6 router instead of a larger 6-input port router. As will be demonstrated later, this also offers many advantages. It should be mentioned that in the proposed architecture, PEs deliver packet from two different sources: PEs in the same layer or from other layers. If it is desirable to use the existing network interfaces (NI), it will necessitate to benefit from a small wrapper. Later, we will present the implementation of a 2×1 wrapper, however a dual-input-port NI can more efficiently exploit the offered bandwidth. This type of NI has been presented for the number of NoC architectures such as NePA [5] and DMesh [163].

Fig. 6.5 illustrates the main components of a generic 5×6 router for the proposed 3D Mesh-based architecture. In this case, a router has 5 input

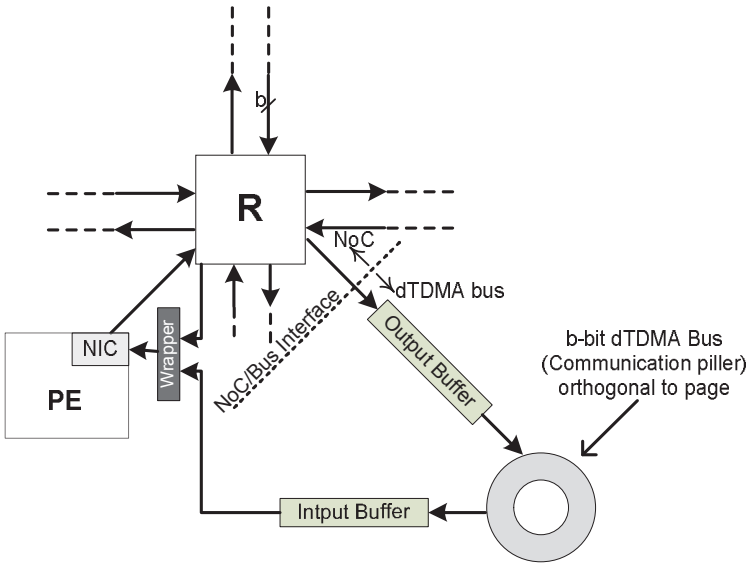


Figure 6.4: High level overview of the proposed Hybrid 3D NoC-Bus mesh router of a pillar node

ports (four for cardinal directions and one for a local PE) and 6 output ports (one additional channel for vertical communication), supporting V virtual-channels per port. Virtual-channel flow control exploits an array of buffers at each input port to improve both throughput and latency by allowing blocked packets to be bypassed. When a flit is ready to be sent, the switch connects an input buffer to an appropriate output channel. The router also comprises of three main modules to control the datapath: a router, a virtual-channel allocator, and a switch allocator. These components decide the direction of next hop, the next virtual channel, and when a switch is available for each flit.

6.2.1 Wrapper Architecture

In order to utilize the existing single-input-port NIs and also to benefit from the proposed architecture, a simple 2×1 wrapper is required. In such cases that there are two concurrent requests for sending packets to a NI buffer, this wrapper performs the arbitration process. More precisely, the wrapper comprises of a simple finite state machine (FSM) and multiplexers to connect the respective input channel to the output channel based on an arbitration policy such as round-robin, priority-based, etc. Fig. 6.6 shows the high-level block diagram of a 2×1 credit-based wrapper.

The state machine in our wrapper is quite simple and area efficient. To be fair, we choose the round-robin policy for arbitration. It should be noted

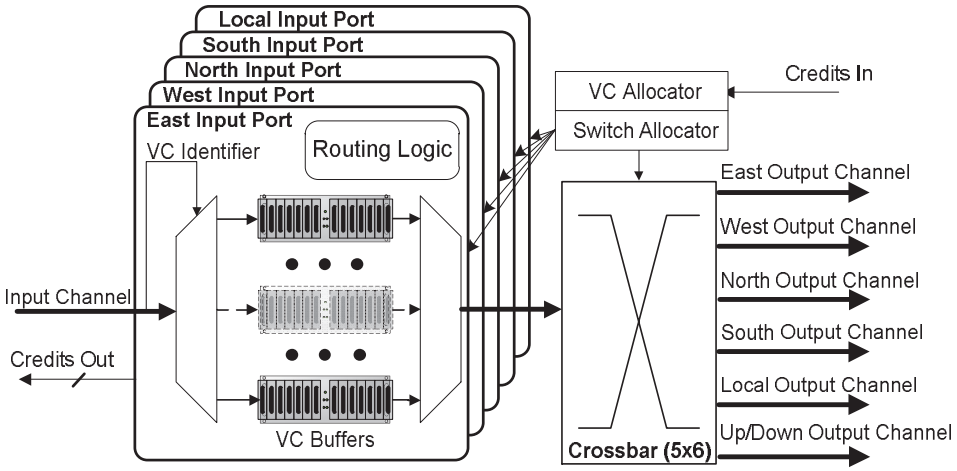


Figure 6.5: Proposed 5×6 router architecture

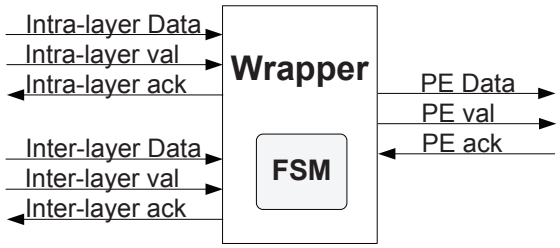


Figure 6.6: Example of a 2×1 credit-based wrapper

that to maintain full compatibility with the existing NI transmission protocols, a packet-based arbitration is used. This means that the arbitration is performed for each packet, not for each flit. The state machine contains four main states, which are *XY-Turn*, *XY-Sending*, *Z-Turn* and *Z-Sending* as shown in Fig. 6.7. At the initialization time of the whole system, access to the output port is granted to the intra-layer communication (*XY-Turn*). In this state, if there is a request from an intra-layer router (Req_XY = '1'), it will be granted and the current state will be changed to *XY-Sending*. The current state remains in *XY-Sending* as long as the packet is being sent. *End-of-packet* signal (eop_XY = '1') indicates the transmission completion for intra-layer communication. At this time, the current state switches to *Z-Turn* indicating that this is the inter-layer buffer turn to send a packet (Req_XY = '0' and Req_Z = '0'). Alternatively, it will directly get the *Z-Sending* state, on condition that there is a request from the inter-layer buffer (Req_Z = '1'). At the initial stage (*XY-Turn*), providing that there is no request from the router (Req_XY = '0'), the request signal from the inter-

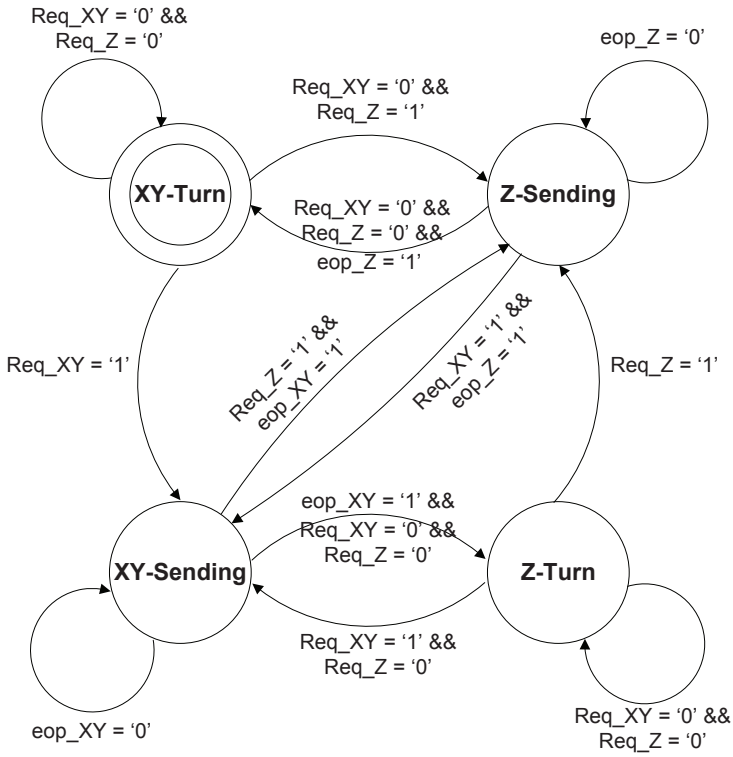


Figure 6.7: Finite state machine for the proposed wrapper controller

layer buffer will be checked. If there is a request from the buffer (Req_Z = '1'), the current state will switch to *Z-Sending* and the packet transmission will be initiated. A similar procedure is followed when the current state is *Z-Turn*.

Our experiments reveal that typically the competition on the wrapper side is not too high even when a network is fully loaded. Based on the fact that bus is not a concurrent medium, it is not common for an intermediate buffer to receive grants from a bus arbiter unceasingly. This fact indicates that the presented wrapper has a negligible impact on network average packet latency.

6.3 Experimental Results

To demonstrate the better performance, power, and area characteristics of the proposed 3D NoC, a cycle-accurate NoC simulation environment was implemented in HDL. Hybrid 3D NoC-Bus mesh [93], and the proposed architecture using *LastZ*-based routing algorithms were analyzed for different traffic patterns. For the typical NoC, the on-chip network considered for

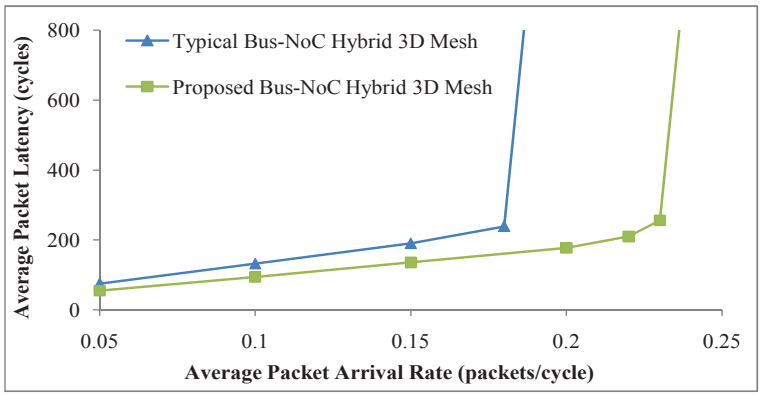
experiment is formed by a typical state-of-the-art router structure including buffers, a routing unit, a switch allocator, VC allocators (for the second and third experiment sets) and a crossbar. For the Hybrid 3D NoC-Bus based System, routers have 6 input/output ports and for the proposed architecture, routers have 5 input and 6 output ports. The arbitration schemes for the switch allocator and the wrappers are based on round-robin.

6.3.1 Synthetic Traffic Analysis

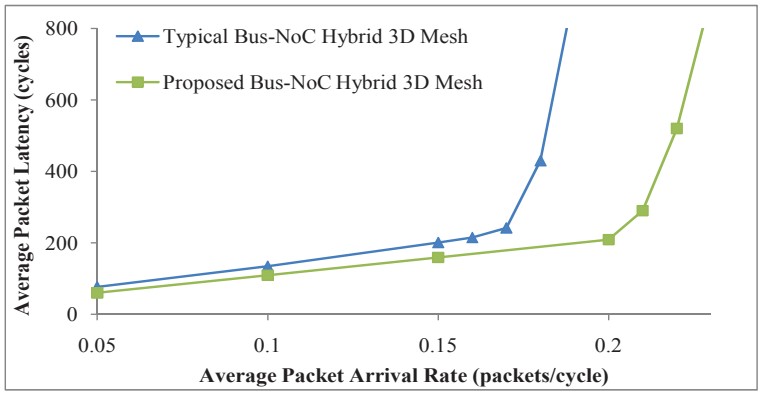
To perform the simulations under synthetic traffic profiles, we used uniform, hotspot 10%, and Negative Exponential Distribution (NED) [130] traffic patterns. For each simulation, the packet latencies were averaged over 50,000 packets. Latencies were not collected for the first 5,000 cycles to allow the network to stabilize. It was assumed the buffer size of each FIFO was eight flits, and the data width was set to 64 bits.

In the first experiment set, 3×3×3 3D meshes and packets with a length of nine-flits were used. For average packet latency analysis, the static *XYZ* [137] routing algorithm was used. Owing to the deadlock-free nature of this routing algorithm, we used routers without virtual channel for this experiment set. The packet latencies for different traffic profiles are shown in Fig. 6.8. For the hotspot 10% traffic pattern, the node at (2, 2, 2) is destined as the hotspot node. It can be observed for all the traffic patterns, the network with the proposed architecture saturates at higher injection rates and always offers reduced average packet latency compared to the typical Hybrid 3D NoC-Bus mesh. The more important reasons for this performance improvement are the reduced total average hop count, the high-speed router, and the high-throughput network.

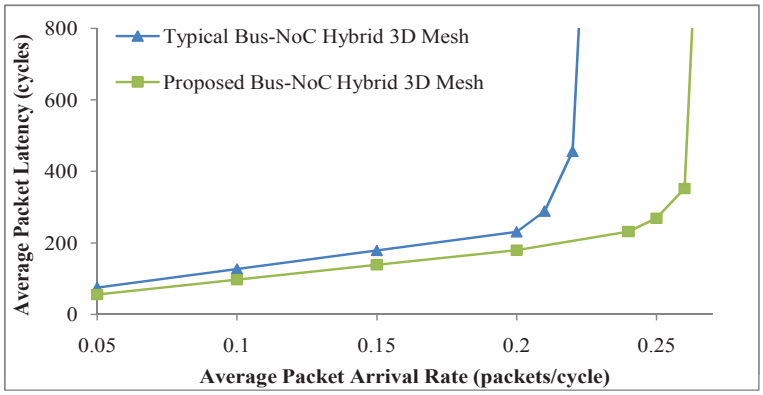
In the second experiment set, we changed some of the NoC parameters considered in the first set of experiments. For the simulations, 3×3×4 (4 layers) 3D meshes and packets with a length of twelve-flits were used. To assess the average packet latency, *(DyXY)Z* [90] routing algorithm was utilized. More precisely, firstly packets follow the adaptive *DyXY* routing algorithm to reach target buses, and then via the inter-layer bus they are delivered to target PEs. In *DyXY* algorithm, which is based on the static XY algorithm, a packet is sent either to the X or Y direction depending on the congestion condition. It uses local information which is the current queue length of the corresponding input port in the neighboring routers to decide on the direction of next hop. Because this routing algorithm is not deadlock-free, we used routers with two virtual channels per input port. The nodes at (2, 2, 2) and (2, 2, 3) are chosen to receive more packets for the hotspot 10% traffic pattern. The packet latency for different traffic profiles are shown in Fig. 6.9. For this experiment set, our results also reveal improved average packet latency for varying average packet arrival rates compared to the



(a) Under uniform traffic profile

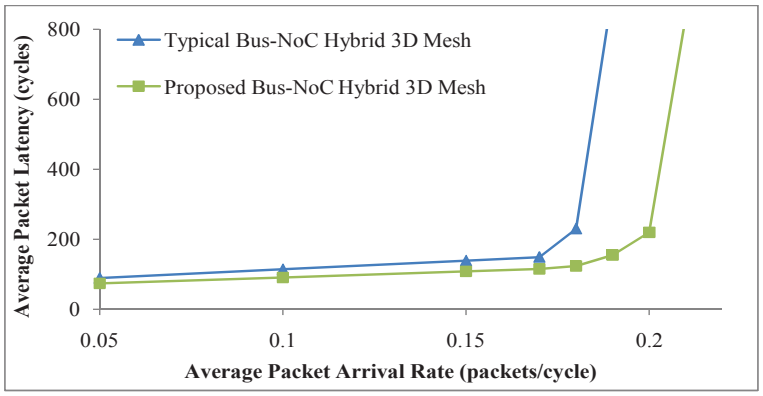


(b) Under hotspot 10% traffic profile

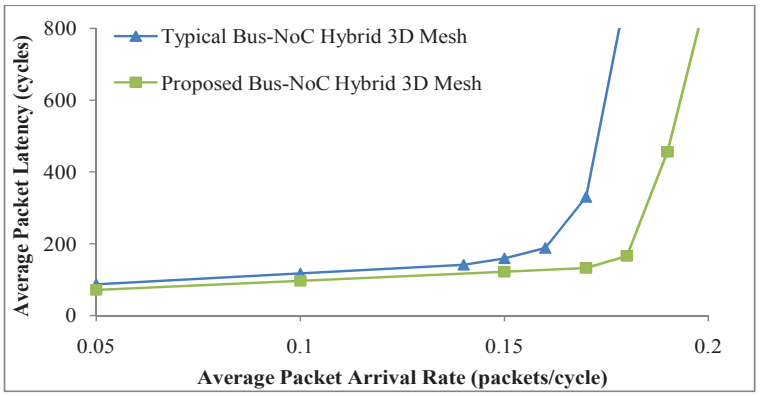


(c) Under NED traffic profile

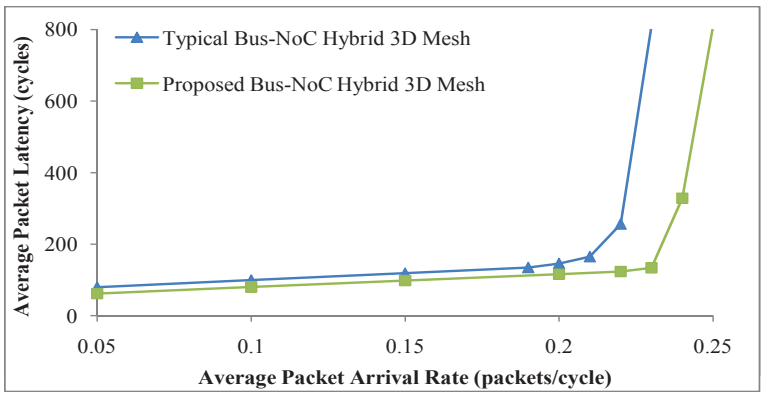
Figure 6.8: Latency versus average packet arrival rate on a 3×3×3 mesh using XYZ routing without virtual channel



(a) Under uniform traffic profile

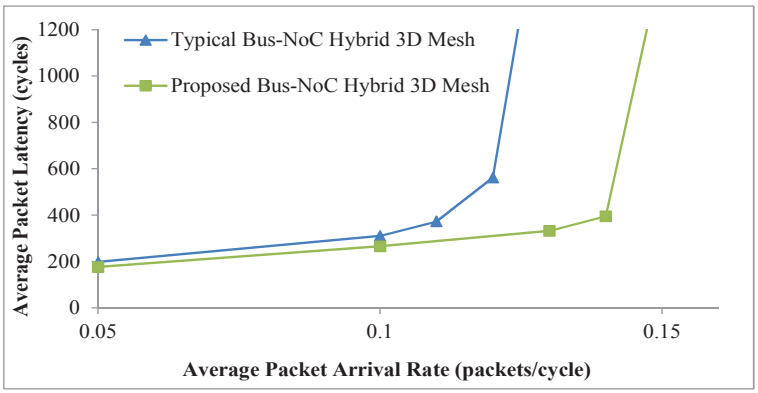


(b) Under hotspot 10% traffic profile

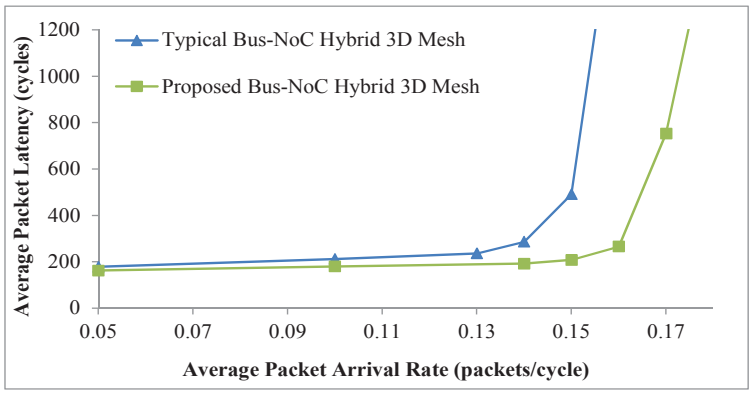


(c) Under NED traffic profile

Figure 6.9: Latency versus average packet arrival rate on a 3×3×4 mesh using (DyXY)Z routing with virtual channels



(a) Using XYZ routing without virtual channel



(b) Using (DyXY)Z routing with virtual channels

Figure 6.10: Latency versus average packet arrival rate on a 6×6×3 mesh under uniform traffic profile

typical Hybrid 3D NoC-Bus mesh.

In the third experiment set, large 6×6×3 3D meshes were analyzed under uniform traffic profile using XYZ routing without virtual channel and *(DyXY)Z* routing with 2 virtual channels. Packets length was also changed to six-flits. The average packet latency of the networks using different routing algorithms under the uniform traffic profile are shown in Fig. 6.10. As can be seen from the figure, in comparison with the typical hybrid architecture, the proposed hybridization architecture reduces the average packet latency also for large networks with more than hundred cores.

6.3.2 Videoconference Application

For realistic traffic analysis, we used the encoding part of videoconference application with sub-applications of H.264 encoder, MP3 encoder and OFDM

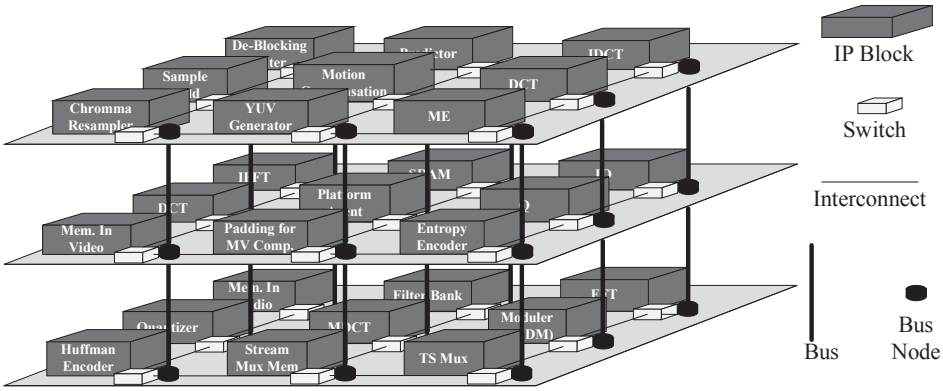


Figure 6.11: Partition and mapping of the video conference encoding application

transmitter presented in Chapter 5. The video stream used for simulation purposes was of size 300×225 pixels and each pixel consists of 24 bits. Thus each video frame is composed of 1.62 Mbits and can be broken into 8400 data packets each of size 7 flits including the header flit. The data width was set to 64 bits. We modeled the application graph, mapping strategy, frame rate, buffer size, number of nodes, layers and generated packets, supply voltage and clock frequency used in Chapter 5 for the real application simulation. In order to implement adaptive routing, routers with 2 VCs per input port were used. The application that is mapped to 3×3×3 3D-mesh NoC is shown in Fig. 6.11.

To estimate the power consumption, we used the high-level NoC power simulator presented in Chapter 5. The simulation results for power and performance of the videoconference encoding application are shown in Table 6.1. The proposed *LastZ-based* architecture showed about 24% and 16% drop in power consumption over the Symmetric 3D-mesh NoC and Hybrid 3D NoC-Bus mesh architectures, respectively. Similarly, 16% and 5% reduction in APL over the Symmetric 3D-mesh NoC and Hybrid 3D NoC-Bus mesh architectures was also observed for our proposed architecture. Note that for the realistic application the simulated 3D architectures do not benefit from VCs. More power and performance improvements could be speculated for the proposed architecture compared to other 3D architectures in case of utilizing VC-based routers.

6.3.3 Area Calculation

Finally, the area of the different routers was computed once synthesized on CMOS 65nm *LPLVT STMicroelectronics* standard cells using Synopsys Design Compiler. The 2×1 32-bit round-robin-based wrapper was also syn-

Table 6.1: Power Consumption and Average Packet Latency

3D NoC Architecture	**Power Consumption (W)**	**Average Packet Latency (cycles)**
Symmetric NoC 3D Mesh	1.587	186
Hybrid 3D NoC-Bus Mesh	1.439	166
LastZ-based Hybrid 3D Bus-NoC Mesh	1.204	157

Table 6.2: Hardware implementation details

Component	**Area (μm^2)**
2×1 32-bit Wrapper	203
5×5 Conventional 2D NoC Router Without VC	33678
5×6 Proposed *LastZ-based* NoC Router Without VC	35230
6×6 Hybrid 3D NoC-Bus Router Without VC	42764
7×7 3D Symmetric NoC Router Without VC	58973
5×5 Conventional 2D NoC Router With VC	70675
5×6 Proposed *LastZ-based* NoC Router With VC	73013
6×6 Hybrid 3D NoC-Bus Router With VC	91951
7×7 3D Symmetric NoC Router With VC	122779

thesized to illustrate the area overhead of the interface. To observe the area savings of more complex routers, we synthesized routers supporting virtual channels as well. For these routers, we set the number of virtual channels to 2. The routing logic for all routers was set to XYZ. The layout area of a conventional 2D NoC router, the proposed *LastZ-based* NoC router, a conventional Hybrid 3D NoC-Bus router, a 3D Symmetric NoC router and the wrapper are listed in Table 6.2. For all the routers, the data width and buffer depth were set to 32 bits and 8 slots, respectively. The figures given in the table reveal that compared to a conventional Hybrid 3D NoC-Bus router, the area savings for the proposed *LastZ-based* router is around 18% and 21% for without- and with-VC implementations, respectively. Additionally, Table 6.2 shows the area overhead of the proposed wrapper is negligible. For more complex routers supporting a large number of VCs, complex VC management techniques [106][131], and wider data width, it is expected to have more area savings.

6.4 Discussion

The proposed hybridization scheme offers many advantages. Obviously, there are some drawbacks associated with this architecture. As described in Fig. 6.5, input ports are complex and area-hungry. Hence, removing one of

them could be significantly beneficial. In this part, we enumerate the pros and cons of the proposed hybridization scheme:

- **Low-cost router.** The router area and cost in this architecture is reduced because of removing one complex input port.
- **Reduced average hop count.** Applications under this architecture have a lower average packet latency due to the fact that the total average hop count of the system is decremented by 1. This solely can significantly reduce the average packet latency.
- **High-speed router.** Removing one port from input channel reduces the crossbar size and consequently router critical path. This allows a router to operate at a higher frequency.
- **Fast intra-layer packet transmission.** Packet transmission within layers is performed faster as a result of less competition between input ports and no interference from the packets issued from other layers.
- **Fast inter-layer packet transmission.** In this architecture, packets reach destination pillars when they are located exactly above or below target PEs. In this case, if the PEs are fast enough to store and consume packets, there will not be any congestion in the slave part of vertical buses. In contrast, in typical Hybrid 3D NoC-Bus mesh architecture, if a router connected to a bus and serving as a slave, is congested, packets will be delayed in the master routers for an undetermined length of time, even if the buses are free to serve.
- **High-throughput network.** This architecture enables network interfaces to read from two different channels concurrently. In this situation, a customized dual-input-port NI can exploit the offered bandwidth. This feature could be also advantageous in mixed-technology implementations when prioritization between inter-layer or intra-layer streams are needed due to different packet types. However, we propose a wrapper to preserve the compatibility of our architecture with the existing NIs.
- **Utilization of the existing resources.** The proposed hybridization scheme utilizes the available intermediate buffers (see Fig. 6.1) for connecting PEs to Buses. The only additional hardware is a 2×1 wrapper which is around 30 times smaller than a typical input port.
- **Reduced power consumption and mitigated thermal issues.** The proposed smaller routers consume less static and dynamic power consumption. Additionally, the reduced average hop count and average packet latency enable faster packet transmission. Based on this

fact, packets stay in intermediate routers for a shorter period of time and further reduce the static power. The overall average power reduction is beneficial to mitigate thermal issues as well.

- **Independency of intra-layer topology and routing algorithm.** Since the proposed hybridization scheme only deals with inter-layer communication, it is independent of the intra-layer topology. The mesh topology for intra-layer communication is only an example in this work. In addition, the architecture is independent of intra-layer routing algorithm as well. In other words, there is no limitation to use static/adaptive or minimal/non-minimal routing algorithms for intra-layer communication and load balancing. The algorithm just should follow the *LastZ* rule.

- **Efficient platform to implement Processor-Memory architectures.** In Processor-Memory architectures, one solution to reduce the average data access latency is to divide caches into smaller banks. These caches have different access latencies, from a given processor's perspective. This type of cache architecture is called a nonuniform access cache architecture (NUCA) [76]. The 3D multi-processor design is a promising approach to NUCA to reduce wire delays [88][93]. It is expectable that since the processors are power-hungry components in a chip, stacking multiple processor layers could be unwise for heat dissipation. A typical way of applying 3D integration for a chip is to partition all the processors to one layer and other components to the other die layer [168]. In consideration of heat dissipation, a processor connected to a bus is placed on the top layer (near heatsink) and its most influential (low-access latency) caches are connected to the same bus in different layers. Our proposed approach results in significantly reduced latencies between processors and memories, and a corresponding increase in NoC bandwidth due to the low-latency inter-layer communication.

- **Applicable for TSV grouping approaches.** In 3D ICs, TSV pads occupy significant chip area and result in routing congestions. Moreover, as the number of TSVs grows the yield of 3D ICs decreases significantly. Based on these facts, when utilizing full layer-layer connection is not affordable for a system, TSV squeezing technique is used [91]. The idea is to share pillars among neighboring routers to improve TSV utilization and cost. Fig. 6.12 shows such a system in which four nodes share a single TSV bundle. As shown in Fig. 6.13, the proposed hybridization scheme will be applicable and effective for such 3D architectures as well, if the intra-layer routing process is completed before the inter-layer routing.

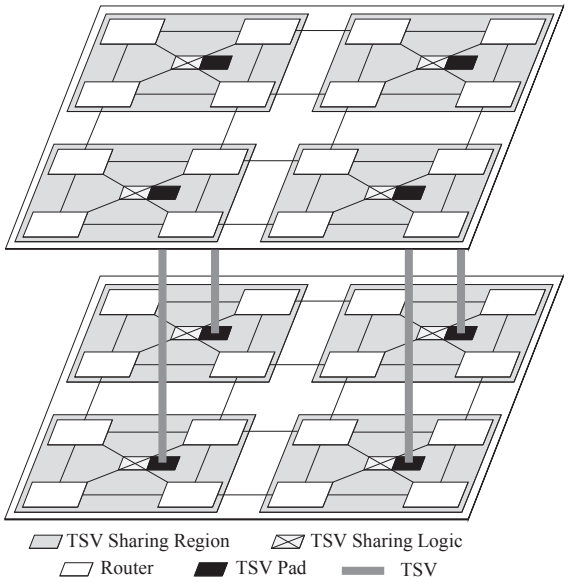


Figure 6.12: 3D Mesh NoC with TSV Squeezing [91]

- **Capability of amalgamation with faulty channel recovery techniques.** A single link failure within inter-layer pillars can obstruct or halt inter-layer communication. Although fault tolerant routing algorithms partly try to tackle this issue, they can usually manage only a very limited amount of faults. Recently, many novel techniques have been presented to handle very large numbers of permanent and transient wire failures that occur either at manufacture-time or at run-time [162][115][86]. Their major contribution is to recover Partially-Faulty Links (PFLs) in order to enable functionally correct transmission of data in the presence of faulty wires. These techniques enable the system to use non-fault tolerant routing algorithms. By forcing the Z plane routing last, we may constrain other available routes to transfer packets in the event of faults on a bus. Amalgamation of these recovery techniques with the proposed architecture enables a system to benefit from *LastZ*-based routing even in the presence of faults on pillars. It should be noted that at the presence of a Fully Faulty Vertical Link (FFVL), the *LastZ*-based routing is not efficient due to lack of inter-layer adaptivity.

- **Inability to support adaptivity for inter-layer communication.** Based on the *LastZ* rule, the vertical hop must be taken as the last hop. This rule leads to a limitation that the routing adaptivity is restricted to the intra-layer routing. Therefore, if a totally adaptive routing

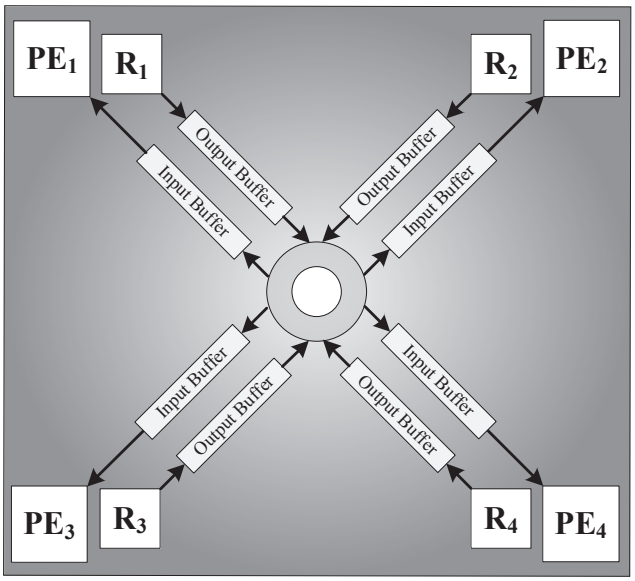


Figure 6.13: Top view of a TSV sharing region using the proposed hybridization scheme

algorithm is desired being able to balance the load across all layers, then this architecture is not recommended. However as mentioned in Section 6.1.1, there are many available and recently proposed 3D NoC systems using *LastZ*-based routing algorithms.

6.5 Summary

In this chapter an efficient hybridization scheme was proposed to address the naive and straightforward hybridization between NoC and bus media in the Hybrid 3D NoC-Bus Mesh architecture. By utilizing a routing rule, namely *LastZ*, the proposed hybridization scheme offered many advantages in terms of system performance, power consumption, and area footprint of the system. Moreover, to guarantee the compatibility of the proposed hybridization scheme with the existing network interfaces, a low-cost wrapper was presented. Our extensive simulations with synthetic and real benchmarks, including the one with an integrated videoconference application demonstrated that compared to a typical Hybrid 3D NoC-Bus Mesh architecture, our hybridization scheme achieves significant power, performance, and area improvements.

Chapter 7

Adaptive Inter-Layer Communication

The Stacked (Hybrid 3D NoC-Bus) mesh architecture presented in [88] is a hybrid architecture between the packet switched network and the bus architecture to overcome the mentioned 3D Symmetric NoC challenges. As the inter-layer distance for 3D ICs is small, the bus length will also be smaller; approximately (n-1) $\times$ 20μm, where n is the number of layers. This makes the bus suitable for inter-layer communication in vertical direction.

In this chapter, we address the routing issues and buffer utilization to enhance the overall system power, performance, and reliability of existing stacked mesh architectures. The proposed architecture increases tolerance against single bus failure compared to the existing architectures.

7.1 Motivation and Contribution

As discussed before, the NoC router can be hybridized with a bus link in the vertical dimension to create a Hybrid 3D NoC-Bus structure. In the static XYZ routing algorithm, consider that R_{XYZ} is the source, which needs to send a data packet to the destination R_{XYZ+1} as shown in Fig. 7.1. This particular architecture is neither power nor performance efficient because of inefficient bus utilization. This can be justified by considering different communication scenarios as follows.

- *Bus is busy but input buffer of R_{XYZ+1} is free*: The packet will get stuck in R_{XYZ} output buffer as long as the bus is busy and arbiter does not give bus access to R_{XYZ}. If the neighboring bus becomes free at a later stage to route the packet, the routing cannot take place even with adaptive routing algorithms after the packet has been written to the output port buffer of R_{XYZ}.

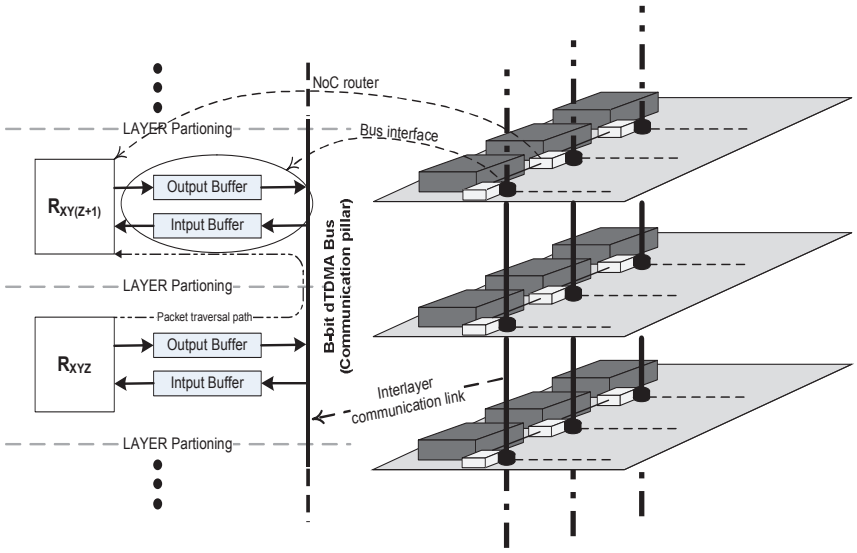


Figure 7.1: Side view of the 3D NoC with the dTDMA bus

- *Bus is free but input buffer of R_{XYZ+1} is full*: The packet will be delayed for an undetermined length of time, because another transaction may start at the bus. In the meanwhile destination buffer may be available. Also, if another node with a high priority packet requests access to the bus or even worse the R_{XYZ+1} input buffer, the delivery time of packet becomes completely uncertain.

- *Bus is busy and also the input buffer of R_{XYZ+1} is full*: For the stacked mesh architecture, if the destination node is not exactly above/below the current node, packets will wait unnecessarily even if the rest of the network resources are available.

According to the communication scenarios discussed above, the stacked mesh architecture has two major drawbacks. First, each packet is traversed through two buffers, which increases the dynamic power consumption. Secondly, the output buffers are not beneficial in reducing the latency. They also hinder the on-chip network from implementing adaptive and congestion-aware routing algorithms. This happens because when the packet is written to an output buffer, it is not possible for the connected router to bring it back and re-route it. The presence of the output buffer also affects the static power consumption.

To cope with these challenges, we propose an efficient inter-layer communication scheme and routing algorithm which enable congestion-aware communication. The novel contribution of this chapter is to address the issues arising when two different communication media (NoC and Bus) are

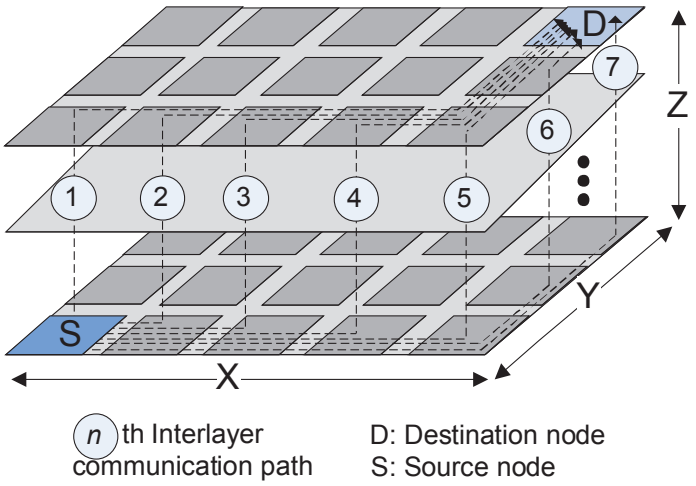


Figure 7.2: Example of *AdaptiveZ* routing

hybridized to form a 3D communication scheme. As we will present in Section 7.2 and Section 7.3 in detail, by removing the output buffers, setting up a smart inter-layer communication, and exploiting an appropriate adaptive routing algorithm, the proposed architecture will improve the average packet latency (APL), power consumption, and fault tolerance.

7.2 Proposed Architecture

As discussed in Section 7.1, it is not efficient to connect a NoC router to a vertical bus without considering their characteristics. We showed the extra output buffer even can obstruct routing adaptivity and load balancing. In this section, we approach the problem in different phases. Initially, the buffer on output port is removed to reduce power consumption and to enable implementing an adaptive routing algorithm. Then, an adaptive inter-layer routing algorithm called *AdaptiveZ* ($\widetilde{Z}$) is applied. For the sake of simplicity, it is assumed that for intra-layer communication a static XY routing algorithm is used. Note that the proposed inter-layer routing algorithm is not dependent to intra-layer routing policy. Therefore, in $\widetilde{Z}$XY routing algorithm, the XY routing is projected in different layers identically as shown in Fig. 7.2 with the adaptive inter-layer communication. The *AdaptiveZ* routing algorithm is elaborated in Algorithm 2.

As can be seen from Fig. 7.2, for inter-layer communication, the first bus pillar that is available on the way for the vertical communication will be used. For this purpose, the router sends a request signal *req* with the destination layer ID to the bus arbiter after each hop on its way until it

reaches the target layer or being exactly below/above the destination router. Based on the status of the target buffer and bus, the arbiter decides to grant, reject, or suspend the request. Once the *grant* or *wait* signals are received, the packet waits there accordingly, to be delivered to the intermediate node in the destination layer. In order to determine if the destination is free or busy, we use the stress values of the bus which is calculated based on the number of the requests pending on the bus. In this work, the wait signal is deasserted when the number of pending requests are more than $\lfloor n/2 \rfloor$. However, other criteria to calculate the stress value of the bus can be used. Once, the packet is in the destination layer ($Z_{diff} = 0$), normal XY (or any other) routing algorithm is used. The packet is delivered to the destination node, once X_{diff} =Y_{diff} =Z_{diff} = 0. Thus, in Fig. 7.2 vertical link #1 has the highest probability to serve the packet under consideration and link #7 has the lowest probability to be used for routing of the current packet. Routing in the vertical dimension is done adaptively by considering the load on vertical links for load balancing purposes. This was done because the bus is a shared medium and adaptive vertical link communication can increase the bus utilization. Note that if the output buffer existed, it would not be possible to bring back the packet and re-route it.

This solution is not deadlock free due to adaptivity. To deal with deadlock, typical virtual channel architecture is used as shown in Fig. 7.3. Consider that the input buffer of router R_{001} contains a packet, which needs to be transmitted to the neighboring node R_{101}. Router R_{101} is already waiting for the availability of dTDMA bus to deliver a packet at the node R_{100}. Also, consider that the node R_{000} and R_{100} have the same situations. In case of typical stacked mesh architecture with adaptive vertical routing and without virtual channel, there will be a deadlock. In the proposed architecture, with the same number of buffers as compared to stacked mesh with dTDMA bus architecture [88], we modify the inter-layer communication scheme to support the VC architecture. The output buffer from the router for bus communication was removed. An extra input buffer is used to receive the data packets and support the VC concept. There is very small area overhead of one 2×1 multiplexer, one 1×2 demultiplexer and few signaling wires. Reduction in dynamic power consumption due to removal of the buffer from the traversal path of packet and also static power optimization due to routing adaptivity dominates the power consumption of this small extra logic. In addition, the added VC not only avoids deadlock, but it also improves the throughput. Now the intermediate buffers are used in a more efficient way and enhance the system power and performance characteristics.

The detailed stacked mesh architecture is shown in Fig. 7.4. The bus arbiter receives the bus request from different nodes by *req* signal along with destination layer ID (*dest_layerID*). The bus arbiter deals with bus request by considering the different parameters like stress value (indicating availabil-

Algorithm 2 *AdaptiveZ Routing Algorithm*

Input: ($X_{current}$, $Y_{current}$, $Z_{current}$), ($X_{destination}$, $Y_{destination}$, $Z_{destination}$), *wait*

Output: Next Hop (E, W, N, S, L, U/D)

1: $X_{diff} = X_{current} - X_{destination}$;
2: $Y_{diff} = Y_{current} - Y_{destination}$;
3: $Z_{diff} = Z_{current} - Z_{destination}$;
4: **if** ($X_{diff} = Y_{diff} = Z_{diff} = 0$) **then**
5: Deliver the packet to the local node and **exit**;
6: **end if**
7: **if** ($Z_{diff} = 0$) **then**
8: Use a 2D intra-layer routing algorithm;
9: **else if** ($X_{diff} = Y_{diff} = 0$) **then**
10: Send a request to the bus arbiter along with the destination layer ID;
11: Wait until receiving the grant;
12: **else if** ($X_{diff} \neq 0$ **or** $Y_{diff} \neq 0$) **then**
13: Send a request to the bus arbiter along with the destination layer ID;
14: **if** (*wait* = '1') {*the destination is free, but the bus is busy*} **then**
15: Wait until receiving the grant;
16: **else** {*the destination is busy*}
17: Withdraw the request;
18: Use a 2D intra-layer routing algorithm;
19: **end if**
20: **end if**

ity of buffer slots) of VC buffers, ongoing transactions and the transactions in the queue waiting for the bus grant. We use local information which is the current queue length of the corresponding VCs connected to the target router to generate the stress value. Depending on the situation, the bus arbiter generates *grant* and *wait* signals. If the node receives a *grant* signal, then it will proceed with the transaction. Instead, if it receives a *wait* signal, the node will continue to wait for the *grant* signal and the bus arbiter will put the request in the queue. If no *grant* and no *wait* signals are received, the node will withdraw the request and move the packet within the layer using any of the 2D routing algorithms. On the next node, again same procedure will be repeated. On the same node, if the bus is granted for vertical communication, the packet will be delivered to the intermediate node in the destination layer. If the packet does not get any bus access on its way for vertical communication until it reaches exactly below or above the destination node, it will wait there without any further routing to get the bus arbitration and will be delivered directly to the destination node.

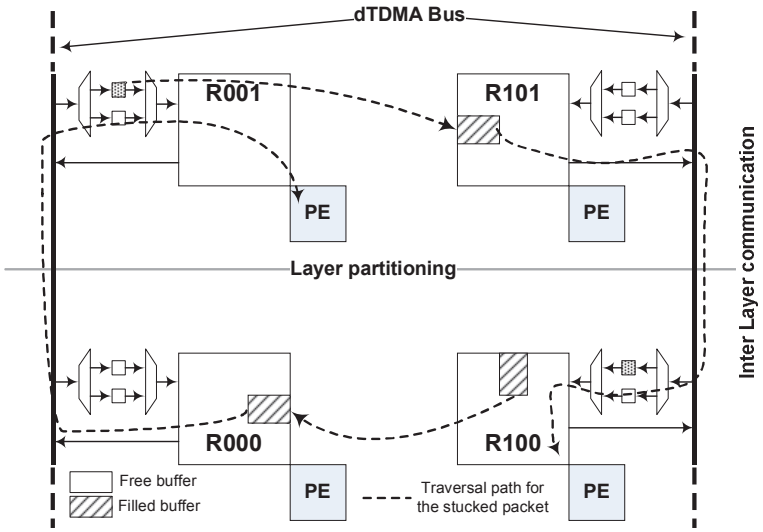


Figure 7.3: VC architecture for the stacked mesh architecture

This is the situation for path #7 in Fig. 7.2.

7.3 Inter-layer Fault Tolerant Routing

The silicon via's, which provide communication links for dies in the vertical direction, is a critical design issue in 3D integration. Like other physical components, the fabrication and bonding of TSVs can fail. A failed TSV may cause a number of stacked known-good-dies to be discarded. As the number of dies to be stacked increases, the failed TSVs increase the manufacturing cost and decrease the yield [62]. A reliable inter-layer communication scheme can considerably cope with these issues. In this section, we explore how the available signals can enable the routing algorithm to avoid these paths (faulty buses) when there are other available paths between the source and destination pair. In this work, we assume that all intra-layer communication links are free from faults.

In [94], we proposed an enhanced decision making routing algorithm named EDXY to avoid congestion in 2D NoC architectures. In addition, the proposed dynamic routing approach can tolerate a single link failure by utilizing the available communication resources. In this section, we enhance our proposed *AdaptiveZ* algorithm to deal with fault tolerance for inter-layer communication of stacked mesh architecture. Similar to the EDXY routing algorithm, the existing signaling used for adaptive inter-layer communication is reused to achieve fault tolerance without adding any extra wires. As explained in Section 7.2, *wait* signal acts as a congestion flag in normal

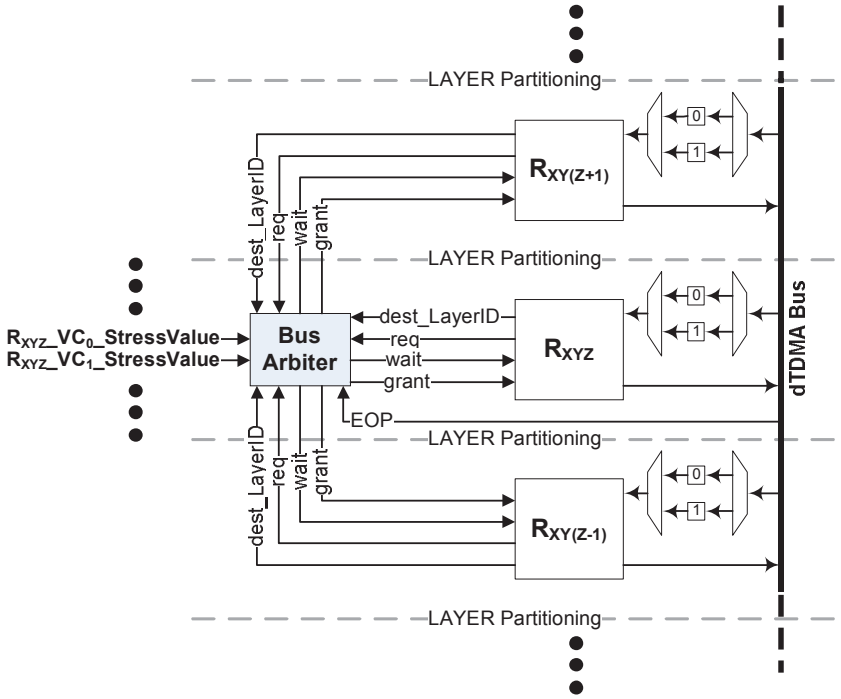


Figure 7.4: Side view of the proposed stacked mesh architecture

conditions. In case of any fault on the dTDMA bus, *wait* signal is permanently asserted to '0' and bus arbiter does not serve any request raised by *req* signal. Thus, routing adaptivity is not affected by introducing the fault tolerance in the existing architecture.

The proposed architecture employs Cyclic Redundancy Check (CRC) codes to detect possible faults in a received flit that may have been corrupted. Before traversing the bus towards the input of the downstream intermediate buffer, each flit is error encoded at the upstream router using payload bits to store a CRC checksum for error detection. The CRC characteristic polynomial that has been used in this work is $g(x) = x^8 + x^2 + x + 1$. The polynomial adds 8 check bits to flit data. The same CRC code has been used in the Header Error Control of ATM networks, and is able to correct single-bit errors (we do not make use of this correction capability) and detect many multiple bit errors [79]. If the error is detected in received flit, the bus arbiter asserts '0' on the *wait* signal. When router checks the *wait* signal, it will re-route the packets within the layer and in future, will not drive any traffic for inter-layer communication to the corresponding dTDMA bus. The minimal path routing will be used even in the case of faulty links without any modification in the proposed routing algorithm. However there are the cases, when routing mechanism needs to be modified to follow

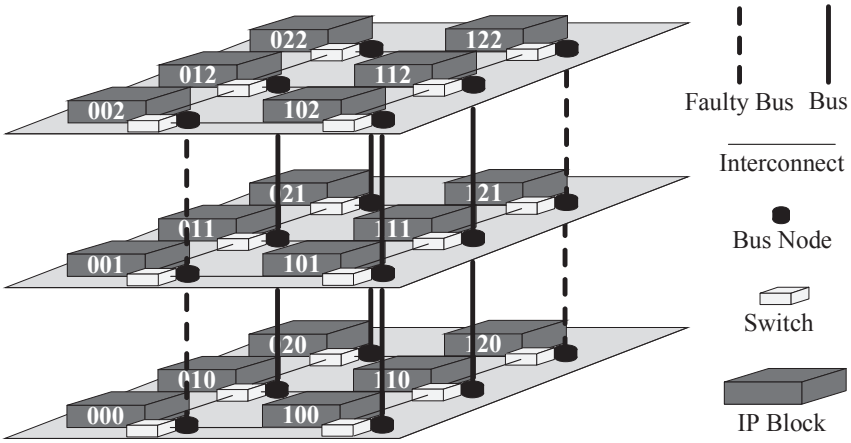


Figure 7.5: Example of modifications to the proposed routing algorithm to guarantee single bus-link failure tolerance

a non-minimal path to reach the destination node.

If the destination node is not above/below the current node or if they are, the bus is not faulty then the packet will be routed via the minimal path. Consider the node '000' in Fig. 7.5 which needs to send a packet to node '112'. The vertical links on two paths are faulty as shown in Fig. 7.5. According to the proposed routing algorithm in Section 7.2, the vertical dTDMA bus connecting the nodes '000', '001' and '002' should be tried first, but that bus link is faulty in the current situation. So, the packet will be routed to either of the nodes '010' or '100' according to the routing algorithm.

We route the packet via non-minimal path through one of the neighboring nodes of the current node, when both the current and the destination nodes are connected to the same faulty pillar. Consider the situation that the packet is exactly below or above the destination node. In this case, it cannot be delivered to the destination if the bus link is faulty. So, the packet will be rerouted to the neighboring node within the layer, where the packet is currently residing and then will be delivered to the destination. This is the situation, when the node '120' contains a packet for node '122'. For the proposed routing algorithm, the packet can only be routed through the vertical dTDMA bus connecting the nodes '120', '121' and '122', which is faulty. In this situation, the packet will be routed to the nodes '110' or '020' via a non-minimal path using one of the neighboring pillars.

This is especially important in future 3D integration in which the failure rate is high. If the interlayer channels are fabricated correctly, the added wires used to reduce network latency, can guarantee packet transmission in the vertical dimension in the case of a bus link failure.

7.4 Experimental Results

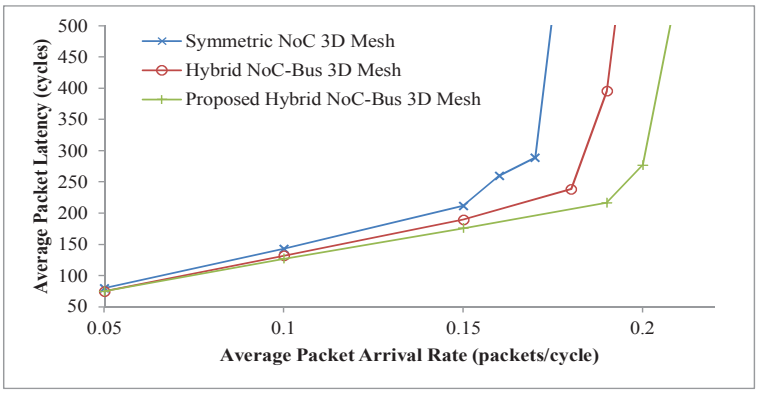
In order to demonstrate the better performance characteristic of the proposed high-speed inter-layer communication scheme, a cycle-accurate NoC simulation environment was implemented in VHDL. Symmetric 3D-mesh NoC [19], Hybrid 3D NoC-Bus mesh [88], and the proposed architecture using *AdaptiveZ* routing algorithm were analyzed for synthetic and realistic traffic patterns. Static ZXY routing was used for Symmetric and Hybrid 3D NoC-Bus mesh. For the proposed architecture, *AdaptiveZ*($\widetilde{Z}$XY) routing was used.

7.4.1 Synthetic Traffic Analysis

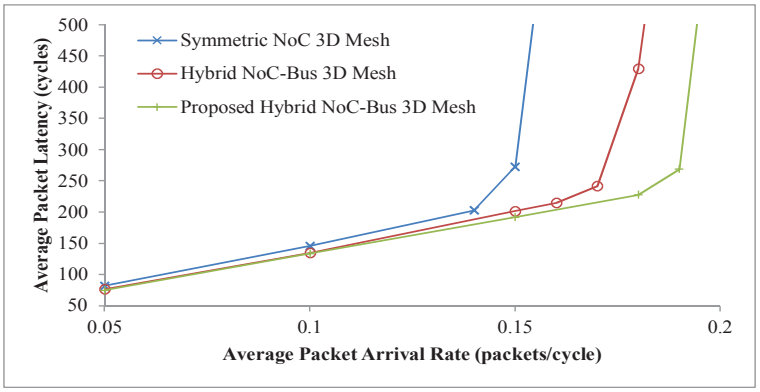
To perform the simulations under synthetic traffic profiles, we used uniform, hotspot 10%, and Negative Exponential Distribution (NED) [132][130] traffic patterns. For each simulation, the packet latencies were averaged over 50,000 packets. Latencies were not collected for the first 5,000 cycles to allow the network to stabilize. It was assumed the buffer size of each FIFO was eight flits, and the data width was set to 64 bits.

In the first experiment set, 3×3×3 3D meshes and packets with a length of seven-flits were used. The average packet latency (APL) curves for uniform, hotspot 10% and NED traffic patterns with varying average packet arrival rates (APAR) are shown in Fig. 7.6(a), 7.6(b), and 7.6(c), respectively. For the hotspot 10% traffic pattern, the node at (2, 2, 2) is destined as the hotspot node. It can be observed for all the traffic patterns, that the network with proposed architecture saturates at higher injection rates. The reason being that the *AdaptiveZ* routing for inter-layer communication increases the bus utilization and makes the load, balanced. In the case of static ZXY routing, the Hybrid 3D NoC-Bus architecture cannot deliver the desired throughput because of bandwidth limitations. For the proposed architecture, bandwidth limitations are managed by proper buffer utilization without increasing the communication resources. For NED traffic, the throughput curves show a significant difference in performance as shown in Fig. 7.6(c).

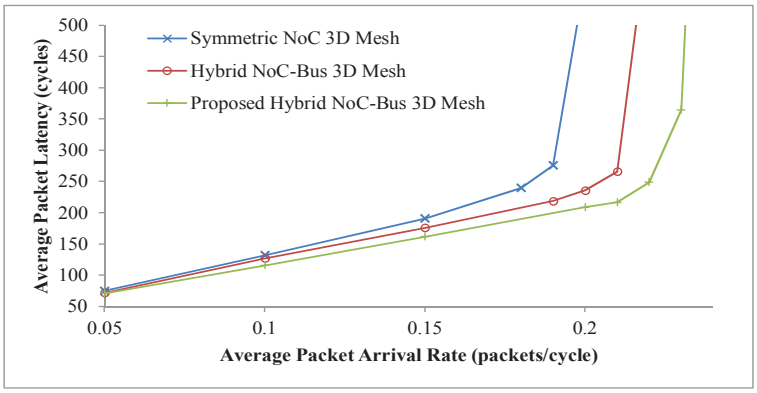
In the second experiment set, we changed some of the NoC parameters considered in the first set of experiments. For the simulations, 3×3×4 3D meshes and packets with a length of ten-flits were used. The nodes at (2, 2, 2) and (2, 2, 3) are chosen to receive more packets for the hotspot 10% traffic pattern. The packet latency for uniform, hotspot 10% and NED traffic patterns with varying APAR are shown in Fig. 7.7(a), 7.7(b), and 7.7(c), respectively. For this experiment set, our results also reveal improved average packet latency for varying average packet arrival rates compared to the typical Hybrid 3D NoC-Bus mesh and the Symmetric 3D-mesh NoC.



(a) Under uniform traffic profile

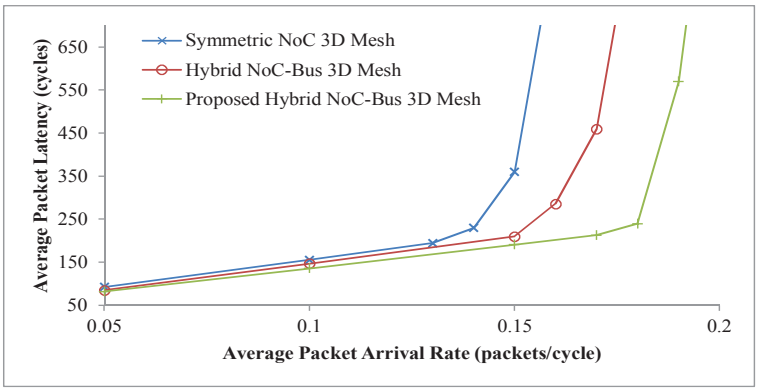


(b) Under hotspot 10% traffic profile

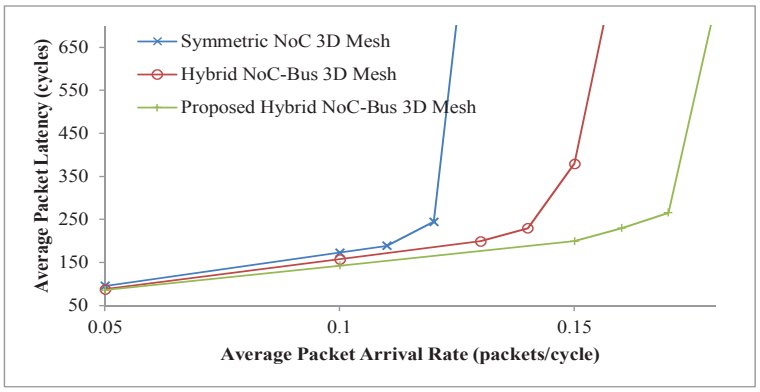


(c) Under NED traffic profile

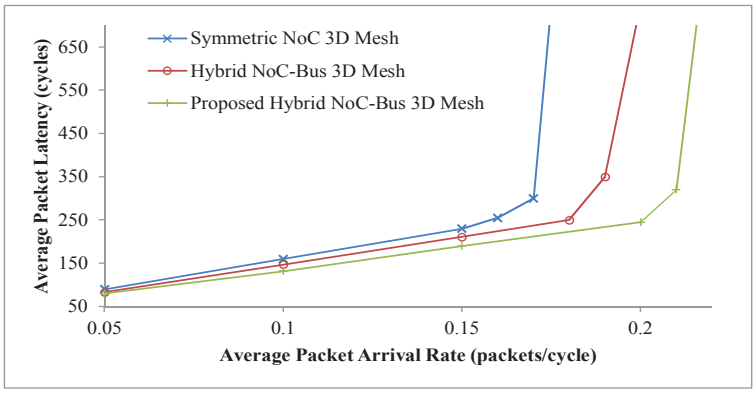
Figure 7.6: Latency versus average packet arrival rate for a 3×3×3 NoC



(a) Under uniform traffic profile



(b) Under hotspot 10% traffic profile



(c) Under NED traffic profile

Figure 7.7: Latency versus average packet arrival rate for a 3×3×4 NoC

7.4.2 Videoconference Application

For realistic traffic analysis, we used the encoding part of videoconference application with sub-applications of H.264 encoder, MP3 encoder and OFDM transmitter presented in Chapter 5. The video stream used for simulation purposes was of size 300×225 pixels and each pixel consists of 24 bits. Thus each video frame is composed of 1.62 Mbits and can be broken into 8400 data packets each of size 7 flits including the header flit. The data width was set to 64 bits. We modeled the application graph, mapping strategy, frame rate, buffer size, number of nodes, layers and generated packets, supply voltage and clock frequency used in Chapter 5 for the real application simulation. The application that is mapped to 3×3×3 3D-mesh NoC is shown in Fig. 6.11.

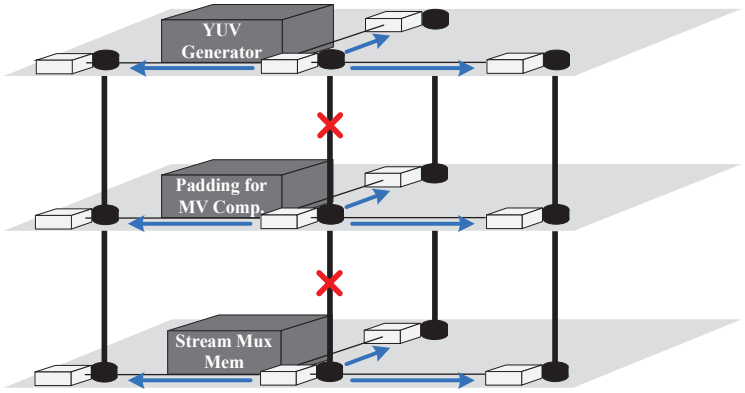


Figure 7.8: 3D NoC running the video conference encoding application with one faulty bus

To estimate the power consumption, we used the high-level NoC power simulator presented in Chapter 5. The simulation results for power and performance of the videoconference encoding application are shown in Table. 7.1. The proposed architecture using the *AdaptiveZ* routing algorithm showed 13% and 4% drop in power consumption over the Symmetric 3D-mesh NoC and Hybrid 3D NoC-Bus mesh architectures, respectively. Similarly, 18% and 8% reduction in APL over the Symmetric 3D-mesh NoC and Hybrid 3D NoC-Bus mesh architectures was also observed for our proposed architecture. Note that the proposed technique only deals with inter-layer communication, but these values are calculated for the whole system. It could be deduced that the proposed architecture has considerably optimized the inter-layer communication with negligible area overhead.

As discussed in Section 7.3, if a fault occurs on a vertical bus, the proposed architecture can tolerate the fault and packets will be re-routed to one of neighboring nodes. To demonstrate that, the system running the

videoconference (Fig. 6.11) with one faulty vertical bus was simulated. The rest of the system parameters were kept unchanged. As shown in Fig. 7.8, we assumed that the bus connecting *Stream Mux Mem*, *Padding for MV Computation*, and *YUV Generator* components is faulty because there is a high inter-layer communication via this bus. Thus, the arbiter of the faulty bus asserts '0' on the *wait* signal and the connected routers re-route the packets within the layer and do not drive any traffic for inter-layer communication to this bus. The power consumption and average packet latency of the proposed architecture using the *AdaptiveZ* routing with one faulty bus are shown in Table 7.1 as well. As mentioned, the proposed architecture can tolerate a single faulty bus using the congestion signal triggered by the bus arbiter with a negligible latency and power overhead.

Table 7.1: Power Consumption and Average Packet Latency

3D NoC Architecture	**Power Consumption (W)**	**Average Packet Latency (cycles)**
Symmetric NoC 3D Mesh	1.587	186
Hybrid 3D NoC-Bus Mesh	1.439	166
Proposed Hybrid 3D NoC-Bus Mesh (*AdaptiveZ* routing)	1.382	153
Proposed Hybrid 3D NoC-Bus Mesh with 1 faulty bus (*AdaptiveZ* routing)	1.491	179

7.5 Summary

In this chapter, a novel inter-layer communication scheme for Hybrid 3D NoC-Bus Mesh architecture was proposed to enhance system performance, reduce power consumption, and improve the system reliability. To this end, a congestion aware adaptive inter-layer communication algorithm was introduced. To deal with deadlock, an appropriate VC architecture was used with same number of buffers as compared to the existing stacked mesh architectures. In addition, the congestion signal triggered by the bus arbiter was used to deal with fault tolerance. The effectiveness of the proposed architecture regarding average packet latency and power consumption has been demonstrated by experimental results using the synthetic as well as the realistic traffic loads. The results showed lower latencies for the proposed *AdaptiveZ* algorithm at places where the congestion in the NoC occurs.

Chapter 8

ARB-NET: A Novel Adaptive Monitoring Platform

To obtain high performance on large on-chip systems, a careful design of on-chip communication platform and efficient utilization of available resources is required. The effective resource utilization needs an efficient system monitoring platform, which can monitor the system at run-time and manage the resource utilization dynamically. Due to large number of cores in 3D NoC based system, heavy traffic loads can create congestion. Similarly, over-utilization of one core can produce thermal hotspots. Moreover, retaining system performance under faults is an essential challenge for 3D NoCs. Such hotspots and congestions can halt the system operation. All these issues direct designers to follow a sophisticated monitoring platform. Except the thermal management system, the existing monitoring schemes for 2D NoCs can be extended for typical 3D NoC architectures. However, Hybrid 3D NoC-Bus mesh architecture necessitates its own customized monitoring platform considering the hybridization structure.

In this chapter, we address the monitoring issues of Hybrid 3D NoC-Bus mesh architectures to enhance the overall system performance and reliability and reduce the power consumption. The proposed platform can be efficiently used for any kind of system management and monitoring such as traffic monitoring and fault tolerance. As a test case, the traffic and inter-layer fault monitoring and management infrastructure is presented. The proposed platform gracefully monitors and extracts traffic and bus failure information from bus arbiters and provides valuable information for connected routers. Based on this information, for the first time, a fully adaptive and inter-layer fault tolerant 3D routing algorithm called *AdaptiveXYZ* is implemented to globally balance the load.

8.1 Background and Related Work

One of the many challenges for NoC community is to monitor the system with minimum possible overheads in power, performance, latency and area. It is not a simple task to monitor the traditional NoC-based systems especially the 3D NoC architectures because of their massively parallel and distributed nature.

The stacked mesh architecture [88] suffers from an important drawback. The issue with this architecture is that, each packet is traversed through two intermediate buffers: the source output buffer and destination input buffer. The output buffer hinders implementing congestion-aware inter-layer communications.

In Chapter 7, we resolved the problems associated with the stacked mesh architecture [88] by removing the source output buffers and rearranging the buffers placement at destination input. Based on the new hybridization scheme, we introduced a congestion-aware inter-layer routing algorithm called *AdaptiveZ* for 3D stacked mesh NoC architectures.

Chao *et al.* [23] proposed a thermal-aware adaptive routing using a proactive downward routing to ensure thermal safety for throttled 3D NoCs. The routing technique is limited to symmetric 3D NoCs, uses non-minimal path and increases the zero load latency. The work presented in [133] addressed these issues and proposed a hybrid adaptive routing algorithm to mitigate the thermal issues. A reconfigurable inter-layer routing mechanism (RILM) for irregular 3D NoCs was presented in [139]. The proposed architecture exchanges a set of messages during initialization to generate an information tree containing location of 3D routers. This information tree is stored at each node. The mechanism is repeated, whenever a fault on any vertical link occurs. Storing the tree information on each node and message exchanging impose considerable overheads in terms of area, power and latency.

In [56] a Hierarchical Agent Monitoring (HAM) for complex, parallel and distributed systems was proposed. The authors proposed to have a separate design layer for monitoring operations, separated from computation and communication because adaptive monitors are required for adjusting the system performance under unpredictable situations. The approach provides high level of abstraction for system designers. However, there is no available physical platform to implement HAM approach for Hybrid 3D NoC-Bus mesh architectures.

For all the discussed architectures, the authors address a specific issue such as fault tolerance, thermal management or traffic monitoring. From an implementation point of view, if a designer considers all the system monitoring and management techniques, the design complexity and overheads will grow dramatically. Apart from design complexity, most of the afore-

mentioned techniques were designed for symmetrical 3D NoCs without consideration of traffic congestion. These issues necessitate an efficient system monitoring platform with minimum overheads capable of considering all the system constraints in parallel without adding extra workloads to the main interconnection network.

8.2 Motivation and Contribution

In Chapter 7, a stacked mesh 3D NoC architecture enabling congestion-aware interlayer communication was presented. The detailed stacked mesh architecture was shown in Fig. 7.4. In this architecture, after receiving a bus request (*req*) along with destination layer ID (*dest_layerID*) from the router, the bus arbiter generates the *grant* and *wait* signals on the basis of different parameters such as stress value of VC buffers, transactions going on and the transactions in the queue waiting for the bus grant. If the router receives the *grant* signal, it will proceed with the transaction. Instead of *grant*, if *wait* signal is received, the router will wait for the grant signal and the bus arbiter will put the request in the queue. The problem arises when no *grant* or *wait* signal is received. In this case, the router will withdraw the request and send the packet to one of the neighboring routers without any information of the status of their connected buses. The packet may have to be rerouted from next hop due to the similar situation. Forwarding a packet without having enough information about the next vertical hop (bus) increases average packet latency and power consumption of the system. In this kind of situations, if the bus arbiter informs the router to choose proper direction for the next hop, it will globally balance the load across all vertical buses. This issue necessitates bus arbiters to have some monitoring information about their neighboring buses.

The same kind of problems may arise for fault tolerant routing or during dynamic thermal management. In fault tolerant routing techniques for 3D NoC architectures, such as the work presented in [139], a lot of information is transferred using the data channels and thus hindering the system performance. For dynamic thermal management, there could be sensors on the silicon die, which needs to exchange thermal data using the existing network and further exacerbates the system performance.

Independent implementation of the mentioned monitoring services imposes a considerable area, power, latency and design complexity overheads on the system. Among the available options, an optimal approach is to exploit a dedicated generic monitoring platform, where different monitoring and management services can be integrated. The available monitoring frameworks for 3D NoC architectures have significant overheads because monitoring data needs to be exchanged across multiple layers. In addition,

the available frameworks cannot efficiently utilize the benefits of hybrid architectures such as Hybrid 3D NoC-Bus mesh architecture.

To deal effectively with the mentioned challenges, we propose a system monitoring platform for Hybrid 3D NoC-Bus mesh architectures. To the best of our knowledge, this is the first effort in implementing generic monitoring platform for Hybrid 3D NoC-Bus mesh architectures. In hybrid NoC-Bus architectures, the bus arbiters that are located in the middle layer are a precious source of information that can be used for monitoring purposes. This monitoring data is available within the same layer. Bus arbiters can exchange the monitoring information directly with each other and form a monitoring platform called ARB-NET without using the data network.

8.3 ARB-NET Architecture

The proposed architecture of our 3D NoC including the ARB-NET monitoring platform is shown in Fig. 8.1. It is basically a 3D stacked mesh architecture which is a hybrid between packet-switched network and a bus. The arbiters resolve the contention between different IP blocks for bus access and in our view are a better source to keep track of monitoring information. Hence, the arbiters can be prudently used by bringing them together to form a network and thereby creating an efficient monitoring and controlling mechanism. The arbiters exchange very short messages (SMS) among themselves regarding various monitoring services that are on offer. The calculation of SMS values are updated every cycle. In this work, we deal with SMS messages which include traffic and thermal stress values (its a measure of how much the bus is busy and thermally stressed respectively) of the bus pillars and possible failures on the bus in order to effectively monitor and control the network of a 3D NoC system.

If n is the number of layers in a 3D stacked mesh NoC and is an odd number, then the arbiter network is placed in the $\lceil n/2 \rceil$ layer. If n happens to be an even number then the arbiter network can be placed in either $n/2$ or $(n/2) + 1$ layer. The main reason for placing the bus arbiters in the middle of the stacked 3D NoC is to reduce the number of vertically connected TSVs by keeping wire distances as uniform as possible [88]. Since, the monitoring platform infrastructure is low cost in terms of area and power, it does not need any dedicated layer and can cohabit with other on-chip logic. Since the routers are mandated to send extra packet information (like X_{dir} and Y_{dir} which define possible directions a packet can take during the minimal path routing), the number of TSVs that are being used are reduced drastically. Our monitoring mechanism that is being proposed and implemented in this work is a low-cost and an optimal approach, as the control network is required to spawn across in only one layer. The proposed

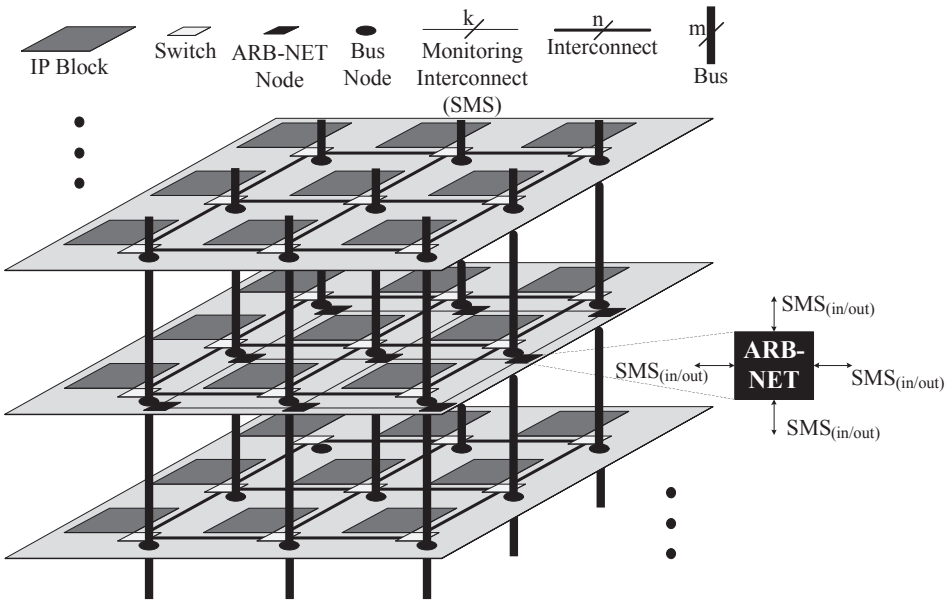


Figure 8.1: ARB-NET-based Hybrid 3D NoC-Bus mesh architecture

monitoring network which uses very short messages, can also be used for fault tolerance as well as making systems thermally efficient. Since, the proposed monitoring infrastructure differentiates between data network and monitoring network, the system performance is not degraded.

Since, the proposed monitoring infrastructure differentiates between data network and monitoring network, the system performance is not degraded. In the monitoring techniques using the data network, monitoring packets are sent throughout the whole network. Generally due to their higher priority, these packets increase the average packet latency of the network particularly at places where the congestion in the NoC occurs. On the other hand, the overdue delivery of these critical monitoring packets can lead to late reactions and further negative impacts on the network. Handling all the monitoring information in a low-cost network separated from data network and located only in one layer deals with the both mentioned issues.

8.3.1 Packet Format

The packet format that is being used in our 3D NoC is shown in Fig. 8.2. It can be seen that the packet has a header flit and a number of payload flits. Each flit is n bits wide. The first bit in the flits is reserved for the *bop* (begin-of-packet) flag and the second bit for the *eop* (end-of-packet) flag respectively. In the header flit, m bits are reserved for the routing information (RI) and k bits are reserved for the monitoring information

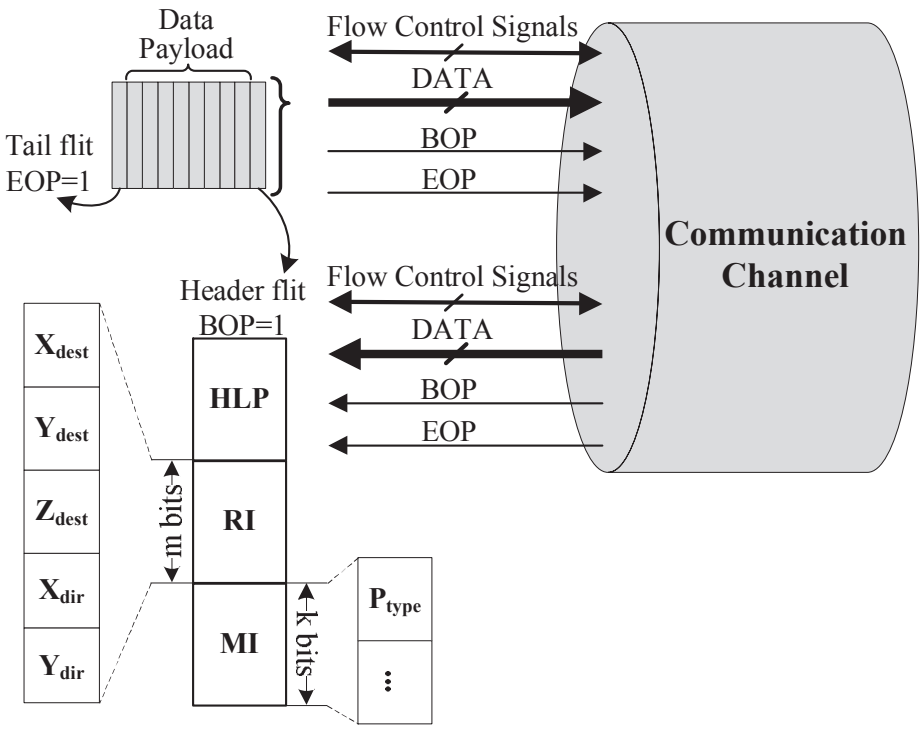


Figure 8.2: Packet format supporting ARB-NET monitoring platform

Table 8.1: $X_{dir}Y_{dir}$ Lookup Table

$X_{dir}Y_{dir}$	00	01	10	11
Possible Directions	$\rightarrow\uparrow$	$\rightarrow\downarrow$	$\leftarrow\uparrow$	$\leftarrow\downarrow$

(MI). The remaining bits are allocated for *Data* and to implement some higher level protocols (HLP). Although such a protocol is not defined in this work, it is reserved for our derivative and future work.

The RI field consists of five fields: X_{dest}, Y_{dest}, Z_{dest}, X_{dir} and Y_{dir}. The X_{dest}, Y_{dest} and Z_{dest} fields indicate the destination address. The X_{dir} and Y_{dir} bits define the possible directions a packet can take during the minimal path routing within the layer of the network, and is described in Table 8.1. The MI field consists of several bits which can be used to effectively monitor the network. One such possible MI field information can be the packet length bits (P_{type}). In this work, in order to classify different types of packets in the network, we use the packet length field (P_{type}) as a differentiating unit between packets instead of packet type. The packet length bits can be ignored in our network for packets which are of constant length.

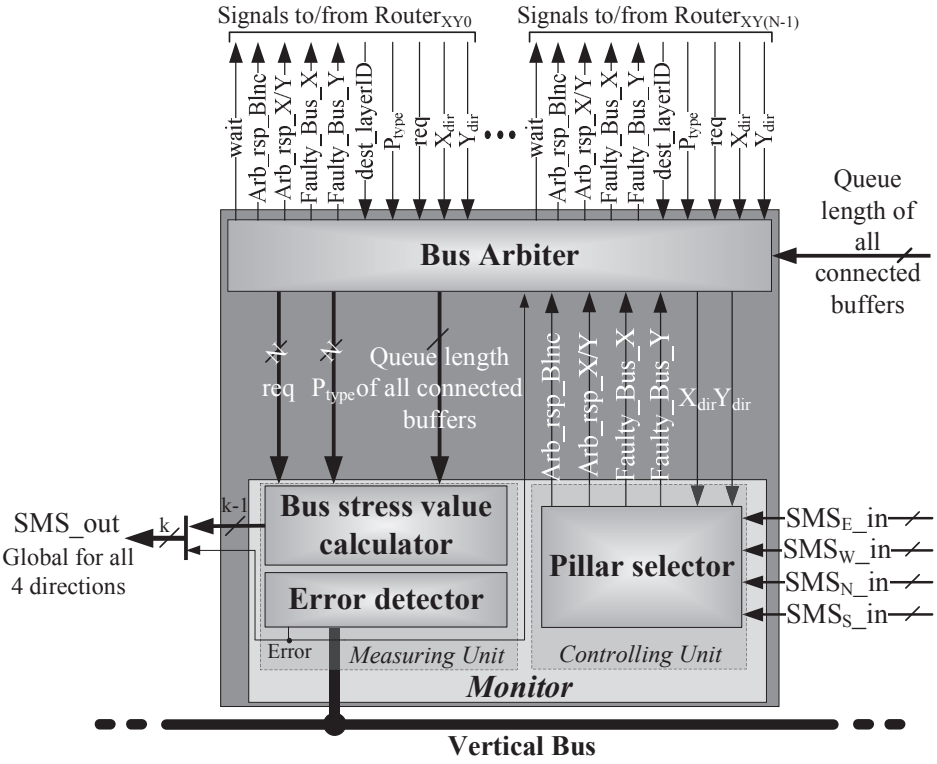


Figure 8.3: ARB-NET node architecture

8.3.2 ARB-NET Node Architecture

The ARB-NET node that is used in our 3D NoC is a central building block for our arbiter network. In its general form, it consists of a measuring unit, control unit and a unit which does the actual arbitration. The measuring unit senses different monitoring parameters (like thermal, faults, traffic, etc), whereas a control unit acts on the parameters that were measured by the measuring unit, by passing actionable control signals to the bus arbiter. In this section, in the case of no faulty buses, we use bus stress values to select least stressed bus pillars (bus pillars are depicted in Fig. 8.1), while adaptively routing data packets in a more efficient way. The measuring unit (consisting of traffic stress value calculator module along with the error detector module) and control unit (consisting of pillar selector module) is represented in Fig. 8.3. The signals to/from the routers to the ARB-NET node are also depicted in the Fig. 8.3.

Algorithm 3 *Bus Stress Value Calculation Algorithm*

Input: req_{1-N}, $P_{type_{1-N}}$, *queue length of all connected output buffers*
Output: $Stress\_value_Z$
N: Number of layers
α: *Constant weight factor*, $0 \leq \alpha \leq 1$

1: $Stress\_value_Z = 0$;
2: **for** $i = 1 \rightarrow N$ **do**
3: **if** ($req_i =$ '1') **then**
4: $Stress\_value_Z = Stress\_value_Z + (P_{type_i} + \alpha \times \mathbf{min}\{queue\_length_{req_i\_dest}\})$;
5: **end if**
6: **end for**
7: $SMS[STV_Z] = Stress\_value_Z$;

Bus Stress Value Calculator and Error Detector

The measuring unit which senses different monitoring parameters calculates stress value of the bus and detects possible failures on the bus and then propagates them to the neighboring arbiters as short messages over the network. Such a measuring unit is depicted as part of the ARB-NET node in Fig. 8.3. The bus stress value is calculated according to Algorithm 3 and is based on the request signal (*req*) and packet type (P_{type}) from the router, and queue length of all output buffers connected to the bus within the destination layer. The total stress value of the bus pillar is calculated by summing up packet type (e.g. length) and the status of the destination buffer associated with the routers request signal. The weight factor (α) is used to regulate the impact of the destination queue length in our algorithm, the optimal value of which is determined based on a series of experiments conducted as described in the experimental results section of this chapter.

The proposed ARB-NET monitoring platform employs CRC codes to detect possible faults in a received flit that may have been corrupted. Before traversing the bus towards the input of the downstream intermediate buffer, each flit is error encoded at the upstream router using payload bits to store a CRC checksum for error detection. The CRC characteristic polynomial that has been used in this work is $g(x) = x^8 + x^2 + x + 1$. The polynomial adds 8 check bits to flit data. When the *Error Detector* unit signals that the bus link is not able to provide services because of either transient or permanent failures, the *Error* signal is asserted to '1'. When the bus arbiter receives any such signals it asserts '0' on the *wait* signal accordingly. When the router checks the *wait* signal, it will not drive any more traffic for inter-layer communication to the corresponding vertical bus.

Pillar Selector

The controlling unit (pillar selector unit) which selects the next bus pillar the packet has to take en route to its destination is being governed by Algorithm 4 and illustrated in Fig. 8.3. In this section it takes into account the stress and fault values of the neighboring buses, as well as the X_{dir} and Y_{dir} bits which give two possible directions the packet can take depending on the response of the Arb_rsp_X/Y signal.

For example if both the X_{dir} and Y_{dir} bits are low, then the pillar selector unit first assigns the bits indicating the direction of faults (east and north bound) to the outputs $Faulty\_Bus\_X$ and $Faulty\_Bus\_Y$, respectively. It then checks the stress values of the buses which are north and east bound to make a decision on the direction the packet has to take. Depending on the Arb_rsp_X/Y signal, the arbiter guides the router to send the packet towards the direction of the bus which is least stressed. When the stress values of buses in both the possible directions is same, then $Arb\_rsp\_Balance$ signal is asserted and the choice of the direction which the packet has to take is left to the router. Similarly, for other values of X_{dir} and Y_{dir}, the bus pillar is selected in accordance with Algorithm 4.

8.3.3 AdaptiveXYZ Routing Algorithm

The routing of packets that takes place in the 3D NoC architecture that has been described above is based on our *AdaptiveXYZ* routing algorithm which has been elaborated in Algorithm 5. In the *AdaptiveXYZ* routing, the direction the data packet has to traverse, depends on the location of the current node, location of the destination, queue length of the buffers in the x-direction and in the y-direction, as well as the response of the arbiter which will be guiding the packet through the network. If the destination node is the same as the current node, then the packet is delivered to the local node and the algorithm exits. If the packet has to be delivered within the same layer, then first stress values which are nothing but the queue lengths of the buffers in the neighboring nodes are calculated, and the node with least stress value is chosen to transmit the packet while on its journey to its destination. If the destination node is connected to the same bus as the current node, then the current node sends a request signal to the bus arbiter along with the destination layer ID. The packets waits at the current node until it receives a grant signal from the bus arbiter.

If the destination node is not on the same layer and is not connected to the same bus as the current node, then the packet is routed in the network after the current node sends a request signal to the bus arbiter along with the destination layer ID, packet type and X_{dir} and Y_{dir} information. The packet either waits upon the receipt of the grant signal, or withdraws the

Algorithm 4 *Pillar Selection Algorithm*

Input: X_{dir}, Y_{dir}, $Stress\_value_{Z_{\{E,W,N,S\}}}$, $Fault_{\{E,W,N,S\}}$
Output: Arb_rsp_X/Y, $Arb\_rsp\_Balance$, $Faulty\_Bus\_X$, $Faulty\_Bus\_Y$

1: **if** ($X_{dir}Y_{dir}$ = "00") **then**
2: $Faulty\_Bus\_X = Fault_E$; $Faulty\_Bus\_Y = Fault_N$;
3: **if** ($Stress\_value_{Z_E} > Stress\_value_{Z_N}$) **then**
4: Arb_rsp_X/Y = '1'; $Arb\_rsp\_Balance$ = '0';
5: **else if** ($Stress\_value_{Z_E} < Stress\_value_{Z_N}$) **then**
6: Arb_rsp_X/Y = '0'; $Arb\_rsp\_Balance$ = '0';
7: **else**
8: Arb_rsp_X/Y = 'X'; $Arb\_rsp\_Balance$ = '1';
9: **end if**
10: **else if** ($X_{dir}Y_{dir}$ = "01") **then**
11: $Faulty\_Bus\_X = Fault_E$; $Faulty\_Bus\_Y = Fault_S$;
12: **if** ($Stress\_value_{Z_E} > Stress\_value_{Z_S}$) **then**
13: Arb_rsp_X/Y = '1'; $Arb\_rsp\_Balance$ = '0';
14: **else if** ($Stress\_value_{Z_E} < Stress\_value_{Z_S}$) **then**
15: Arb_rsp_X/Y = '0'; $Arb\_rsp\_Balance$ = '0';
16: **else**
17: Arb_rsp_X/Y = 'X'; $Arb\_rsp\_Balance$ = '1';
18: **end if**
19: **else if** ($X_{dir}Y_{dir}$ = "10") **then**
20: $Faulty\_Bus\_X = Fault_W$; $Faulty\_Bus\_Y = Fault_N$;
21: **if** ($Stress\_value_{Z_W} > Stress\_value_{Z_N}$) **then**
22: Arb_rsp_X/Y = '1'; $Arb\_rsp\_Balance$ = '0';
23: **else if** ($Stress\_value_{Z_W} < Stress\_value_{Z_N}$) **then**
24: Arb_rsp_X/Y = '0'; $Arb\_rsp\_Balance$ = '0';
25: **else**
26: Arb_rsp_X/Y = 'X'; $Arb\_rsp\_Balance$ = '1';
27: **end if**
28: **else**
29: $Faulty\_Bus\_X = Fault_W$; $Faulty\_Bus\_Y = Fault_S$;
30: **if** ($Stress\_value_{Z_W} > Stress\_value_{Z_S}$) **then**
31: Arb_rsp_X/Y = '1'; $Arb\_rsp\_Balance$ = '0';
32: **else if** ($Stress\_value_{Z_W} < Stress\_value_{Z_S}$) **then**
33: Arb_rsp_X/Y = '0'; $Arb\_rsp\_Balance$ = '0';
34: **else**
35: Arb_rsp_X/Y = 'X'; $Arb\_rsp\_Balance$ = '1';
36: **end if**
37: **end if**

Algorithm 5 *AdaptiveXYZ Routing Algorithm*

Input: ($X_{current}$, $Y_{current}$, $Z_{current}$), ($X_{destination}$, $Y_{destination}$, $Z_{destination}$), $queue\_length_X$, $queue\_length_Y$, Arb_rsp_X/Y, $wait$
Output: Next Hop (E, W, N, S, L, U/D)

1: $X_{diff} = X_{current} - X_{destination}$;
2: $Y_{diff} = Y_{current} - Y_{destination}$;
3: $Z_{diff} = Z_{current} - Z_{destination}$;
4: **if** ($X_{diff} = Y_{diff} = Z_{diff} = 0$) **then**
5: Deliver the packet to the local node and **exit**;
6: **end if**
7: **if** ($Z_{diff} = 0$) **then**
8: $Stress\_value_X = queue\_length_X$;
9: $Stress\_value_Y = queue\_length_Y$;
10: Use $Stress\_value_X$ and $Stress\_value_Y$ to choose the destination port;
11: **else if** ($X_{diff} = Y_{diff} = 0$) **then**
12: Send a request to the bus arbiter along with the destination layer ID;
13: Wait until receiving the grant;
14: **else if** ($X_{diff} \neq 0$ **or** $Y_{diff} \neq 0$) **then**
15: Send a request to the bus arbiter along with the destination layer ID, X_{dir}, Y_{dir}, P_{type};
16: **if** ($wait$ = '1') **then**
17: Wait until receiving the grant;
18: **else**
19: Withdraw the request;
20: **if** ($X_{diff} = 0$) **then**
21: Send the packet to *Y_Axis* towards the destination;
22: **else if** ($Y_{diff} = 0$) **then**
23: Send the packet to *X_Axis* towards the destination;
24: **else if** ($X_{diff} \neq 0$ **and** $Y_{diff} \neq 0$) **then**
25: **if** ($Arb\_rsp\_Balance$ = '0') **then**
26: $Stress\_value_X = \{Arb\_rsp\_X/Y, queue\_length_X\}$;
27: $Stress\_value_Y = \{Arb\_rsp\_X/Y, queue\_length_Y\}$;
28: **end if**
29: Use $Stress\_value_X$ and $Stress\_value_Y$ to choose the output port;
30: **end if**
31: **end if**
32: **end if**

request and then proceeds to send the packet towards the destination along the x or y axis. It may also withdraw the request and use the Arb_rsp_X/Y and $Arb\_rsp\_Balance$ signals provided by the ARB-NET node, to choose

one output port among them through which the packet is propagated in the network. This is done by calculating the stress values of the neighboring nodes as a function of the queue length and the Arb_rsp_X/Y value. This solution is deadlock–free as a result of using typical virtual channel architecture for the interlayer routing as shown in Fig. 7.4. In order to simplify the hardware infrastructure, we simply ignore the *queue_length* of the buffers in the neighboring buses and just use the Arb_rsp_X/Y value provided by the ARB-NET node. In this way, the problem which has been described in the motivation section of this chapter about the routing issue when both the *wait* signal and the *grant* signal are not received by the router has been solved in an innovative way with the help of the pillar selector unit.

8.3.4 Fault Tolerant Inter-Layer Routing

The fault tolerance feature being incorporated in our 3D architecture is governed by a fault tolerant inter-layer routing algorithm as enumerated in Algorithm 6. It should be noted in here that the Algorithm 6 is not the complete fault tolerant inter-layer routing algorithm, but actually represents the last portion of Algorithm 5 (i.e. lines 25-28 should be replaced by Algorithm 6 in order to arrive at a complete fault tolerant inter layer routing algorithm). In this subsection we will be describing only the last portion of Algorithm 5 after replacing it with Algorithm 6.

The packet either waits upon the receipt of the *grant* signal, or withdraws the request and then proceeds to send the packet towards the destination along the x or y axis. It may also withdraw the request and depending on the direction of faulty buses will route packet data accordingly. So, if neither of the buses are faulty, then depending on the Arb_rsp_X/Y and $Arb\_rsp\_Balance$ signals provided by the ARB-NET node, the output port through which the packet has to be sent is chosen. This is done by calculating the stress values of the neighboring nodes as a function of the queue length and the Arb_rsp_X/Y value. If either one of the bus is faulty, then the packet is sent in the other direction towards the destination; or if both the buses are faulty, then the output port is chosen depending on the stress values of their respective routers.

Normally, the minimal path routing will be used even in case of faulty bus without any modification in the proposed routing algorithm. However there are few cases, when routing mechanism needs to be modified. Situations, wherein the current node is exactly below or above the destination node, packets cannot be delivered to the destination if the bus link is faulty. So, the packets will be rerouted to the neighboring node within the layer, where the packet is currently residing. Then they will be delivered to the destination. Because we only forward a packet through a non-minimal path in case of a bus failure and only for source–destination pairs connected to a same bus,

Algorithm 6 *Inter-Layer Fault Tolerant AdaptiveXYZ*
Additional Input: $Faulty\_Bus\_X$, $Faulty\_Bus\_Y$

```
1: if (Faulty_Bus_X = '0' and Faulty_Bus_Y = '0') then
2:    if (Arb_rsp_Balance = '0') then
3:       Stress_value_X = {Arb_rsp_X/Y, queue_length_X};
4:       Stress_value_Y = {Arb_rsp_X/Y, queue_length_Y};
5:    end if
6: else if (Faulty_Bus_X = '1' and Faulty_Bus_Y = '0') then
7:    Send the packet to Y_Axis towards the destination and exit;
8: else if (Faulty_Bus_X = '0' and Faulty_Bus_Y = '1') then
9:    Send the packet to X_Axis towards the destination and exit;
10: end if
```

the lack of livelock in the routing algorithm is preserved.

8.4 Thermal Analysis Of AdaptiveXYZ Routing

Recent advances in packaging technology have lead to Flip-Chip Ball Grid Array (FCBGA) packages being extensively used. In an FCBGA the die is mounted upside-down (flipped) and connects to the package balls (lead-free solder bumps) via a package substrate. The cross-sectional view of a modern 3D flip-chip package is shown in the Fig. 8.4 whose primary consideration will be its ability to transfer heat from the silicon die to the ambient. Unlike the traditional wire-bonding technology, the electrical connection of a face-down (or flipped) integrated circuit onto the substrate is done with the help of conductive bumps on the chip bond pads. The conductive bumps are initially deposited on the top-side of the die during the fabrication process. It is then flipped over so that its top side faces down, and aligned with the marching pads on the substrate. The solder is then flown to complete the interconnection. The advantages of flip-chip interconnect include reduced signal inductance, power/ground inductance, and package footprint, along with higher signal density [67].

In a stacked mesh 3D architecture, the thermal coupling of vertically aligned tiles is larger than the horizontally aligned tiles [22]. This is because the thickness of the silicon dies is much smaller than the lateral dimensions, the lateral heat flow is usually lower than the vertical heat flow. Also, having interface materials with lower thermal conductivities does contribute to this issue. The thermal impact of on-chip 3D NoCs are governed by various non-design issues like the ambient temperature, cooling solutions and the package solutions. In this work we assume that the size of the heat sink is fixed, the ambient temperature around the chip is constant and the velocity

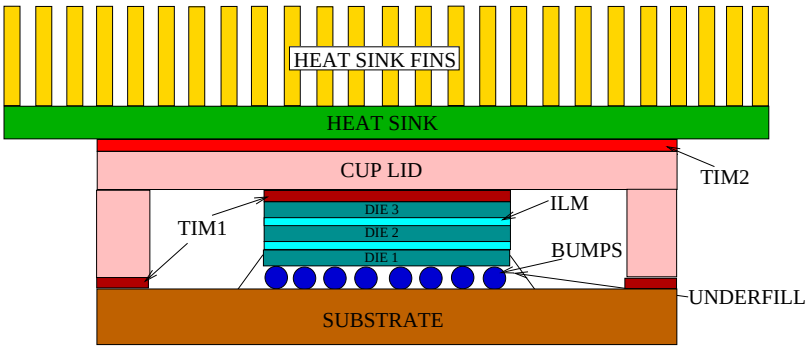


Figure 8.4: Cross-Sectional view of a modern 3D Flip-Chip package

of air-flow is set [143]. We also assume that the application mapping is fixed and just focus on the routing based approach.

8.4.1 Junction Temperature and Thermal Resistance for a 3D System

The two most important thermal parameters for any semiconductor device are the junction temperature (T_J) and thermal resistance (R_{JX}). The junction temperature is usually the highest temperature on a silicon die, whereas the thermal resistance is quantified as the rate of heat transfer between two layers in a package. The junction-to-ambient thermal resistance (R_{JA}) which is a measure to evaluate the thermal performance of a flip-chip package is determined from equation (8.1).

$$R_{JA} = \frac{T_J - T_A}{Q} \tag{8.1}$$

The single-valued junction-to-ambient thermal resistance which has been used traditionally to describe the thermal characteristics of a silicon die is not sufficient enough to describe the thermal performance of a 3D system, due to the presence of multiple heat sources and multiple thermal resistances. Hence, Jain *et al.* [68] have suggested a matrix representation for the junction-to-ambient thermal resistance. In this regard R_{ij} represents the temperature rise in the i^{th} layer per unit heat dissipation in the j^{th} layer. This is represented in the equation (8.2).

$$R_{ij} = \frac{\theta_i}{Q_j} \tag{8.2}$$

Where, θ_i is the temperature rise above ambient of the i^{th} node and Q_j is the heat generated at the j^{th} node. The equation (8.2) can be rewritten as follows.

$$R_{ij} = \frac{T_i - T_A}{Q_j} \tag{8.3}$$

Where, T_i is the junction temperature of the i^{th} layer. So, for a simple two-die stack, where one layer is the processing layer (denoted by subscript 'p') and the other a memory layer (denoted by subscript 'm'), we have 4 different thermal resistance values namely $R_{pp},\ R_{pm},\ R_{mp}\ and\ R_{mm}$ and the junction-to-ambient thermal resistance can be represented as shown below.

$$R_{JA} = \begin{bmatrix} R_{pp} & R_{pm} \\ R_{mp} & R_{mm} \end{bmatrix}$$

In [157], we have modelled a 3D multicore system in a flip-chip package using a commercial finite element based multiphysics modelling and simulation software called COMSOL. In there, we have shown how the thermal resistance and maximum temperature values of silicon layers is greatly improved by optimally placing them in a flip-chip package. In this chapter, we focus on a congestion aware and reliable inter-layer communication scheme for Hybrid 3D NoC-Bus architectures.

8.4.2 Thermal Modeling for 3D NoC

Although 3D NoC systems take advantage of dimensional scaling and are seen as a natural progression towards future large and complex systems, they suffer from significant thermal challenges. These thermal issues caused by device power density and ambient conditions are an important performance and reliability concern. Instantaneous high temperature rises in the devices can possibly cause catastrophic failure, as well as long-term degradation in the chip and package materials, both of which may eventually lead to system failure [67].

We have developed a thermal model for 3D NoC using HotSpot v.5.0. [146]. Hotspot is an accurate and fast thermal model suitable for use in architectural studies and is based on an equivalent circuit of thermal resistances and capacitances that correspond to micro-architecture blocks and essential aspects of the thermal package. We have exploited Hotspot's grid model which is capable of modeling stacked 3D chips for our thermal simulations. Hotspot takes a power trace file and floorplan file as inputs apart from a lot of other model configurations and parameters which can be modified. We obtained the power trace file from our in-house cycle accurate NoC simulator which was implemented in HDL. Like [23], the tile geometry and power model has been adopted from Intel's 65nm based 80-core processor [60].

We have modeled a 3×3×3 NoC using Hotspot. The sizes of the silicon die's 1, 2 and 3 are 4.5 mm × 6.0 mm × 0.15 mm. The convection capacitance and convection resistance of the heat sink are 140.4 J/K and 0.1 K/W respectively. The layers of silicon die are separated by an interlayer material (ILM) whose thickness is around 0.02 mm. The cup lid which acts as the heat spreader and whose thermal conductivity is very high is placed on top of the silicon die. The thermal interface material (TIM) which is some sort of a thermal grease and has very good adhesive properties is being used as the filler material in between the heat spreader and the silicon die. The resistivity of the interlayer material is around 0.249 mK/W (i.e. thermal conductivity = 4.016 W/mK) [33]. Other parameters are left unchanged from Hotspot's configuration file.

Three effective thermal conductivities are used for the lead solder bumps/ underfill layer, substrate layer and the interlayer material respectively. The interlayer material in between the silicon dies is modelled as a homogeneous layer in our thermal model. We assumed a uniform through-silicon-via (TSV) distribution on the die and obtained the effective interlayer material resistivity based on the TSV density (d_{TSV}) values from [33], where d_{TSV} is the ratio of total TSV's area overhead to the total layer area. Coskun *et al.* [33] have observed that even when the TSV density reaches 1-2%, the temperature profile of the silicon die is only limited by a few degrees, thus justifying the use of homogeneous TSV density in our thermal model. According to the current TSV technology [22], the diameter of each via is 10μm, and the spacing required around the TSV's is assumed to be 10μm [33]. For our experiments we have assumed around 8 via's/mm^2, that is around 216 vias spread across the 27 mm^2 area of the silicon die. Hence the TSV density is around 0.062% and the resistivity of the interlayer material is around 0.249 mK/W (i.e. thermal conductivity = 4.016 W/mK) [33].

The presence of high power densities in the routers leads to the increase in hotspots. The information that is being provided by the ARB-NET platform enables the implementation of our congestion aware routing strategy which when used will balance the load in such a way that it increases the utilization of routers which are less active. Also, due to the improved average packet latency, the packet will tend to stay for shorter amount of time in the buffers of intermediate routers and consequently the static power consumption (most of which is leakage power) decreases. The decrease in leakage power leads to decrease in temperatures on the chip. Based on the presented model, power and thermal analysis of the stacked mesh 3D NoC using *AdaptiveXYZ* routing strategy under real application has been performed. In the following section, the efficiency of our proposed *AdaptiveXYZ* routing strategy to mitigate the hotspots has been presented.

8.5 Experimental Results

In order to demonstrate the efficiency of the proposed monitoring platform vis-a-vis its average network packet latency and power reduction, a cycle-accurate NoC simulation environment was implemented in VHDL. Symmetric 3D-mesh NoC [19], Typical 3D NoC-Bus Hybrid mesh [88], *AdaptiveZ*-based Hybrid 3D NoC-Bus mesh [133] and the proposed architecture were analyzed for synthetic and realistic traffic patterns. For Symmetric and Typical Hybrid 3D NoC-Bus mesh architectures, *Z-DyXY* routing was used. In *Z-DyXY* routing, a flit will first travel statically along the Z direction, then *DyXY* algorithm [90] will be used for the intra-layer routing. For the architecture proposed in [133], *AdaptiveZ-DyXY* routing was used. We used the proposed *AdaptiveXYZ* routing for the presented ARB-NET 3D NoC. For all the four architectures, the routers have used the minimal, fully adaptive, reserved VC deadlock avoidance technique as discussed in [105]. We take advantage of the presence of two VCs per each input port that can be effectively utilized.

8.5.1 Synthetic Traffic Analysis

In synthetic traffic analysis of our experiments, we use a 36-node (four layers of 3×3 mesh) network which models a single-chip CMP using network-in-memory. We have assumed that the first layer consists of processor arrays and other layers are shared cache memories. The flit size was set to 128 bits. Our cache coherency protocol is a 3-phase protocol, which means a request(R)/response(S) control message is issued to the memory/cache. The cache responds, and then the cache controller sends the actual data. This avoids any race conditions. In conclusion, each of the R/S message is used just for control, short enough to fit a flit (e.g. 1 R/S message = 1 flit). We used 5-flit packets to transfer on each 64-Byte cache line (read response or write request data). Because there are only two packet types (1-flit and 5-flit packets), we used the actual packet length instead of packet type (P_{type}) in packet headers. For each simulation, the packet latencies were averaged over 500,000 packets. It was assumed that the buffer size of each FIFO is four flits.

In Algorithm 3 line 4, α is used as a criterion to regulate the impact of destination queue length on the bus stress value calculation. The greater the value of α, the greater is the stress value that is assigned to the corresponding bus. Balancing the influence between the source part (req and P_{type}) and the destination part ($queue\_length_{req_i\_dest}$) is a critical issue for optimizing bus stress value. Fig. 8.5 shows throughput comparison between different α values under the uniform random traffic pattern. For this analysis, STV_α_0 denotes parameter $\alpha = 0$. Small α leads to reduction in network throughput

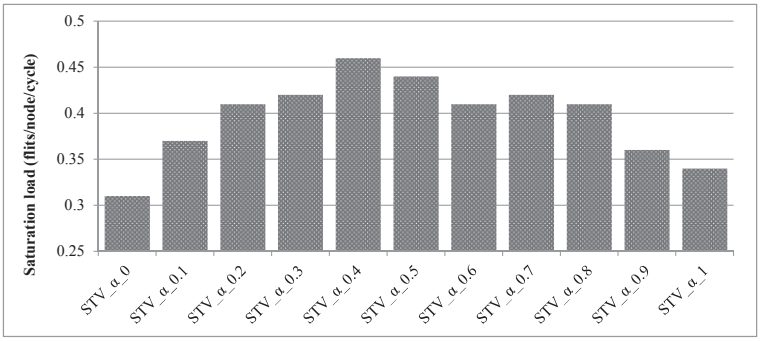


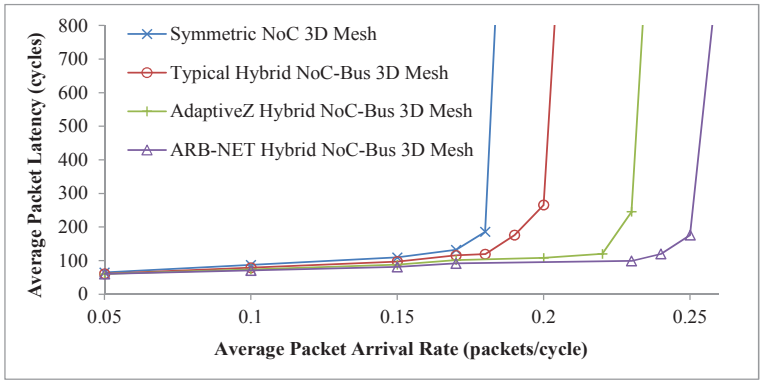
Figure 8.5: Average throughput for different α values

due to the reduction in the impact of FIFOs in the destination part. Large α might excessively reduce the source part impact and negatively affect the throughput. The case where $\alpha = 0.4$, sustains the most throughput for the proposed architecture. Since a comparison between the neighboring bus stress values is needed the division operation can be eliminated by quantizing the variables used in Algorithm 3. Based on the number of layers, packet types, intermediate buffer depth, and the selected alpha value in our simulation environment, after quantization, 11 bits are enough to exchange the monitoring messages (SMS).

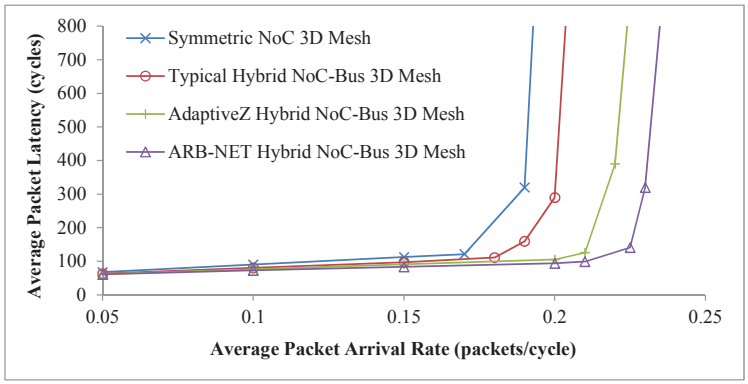
To perform the simulations, we used uniform, hotspot 10% (2 nodes) and Negative Exponential Distribution (NED) [130] traffic patterns. The average packet latency (APL) curves for uniform, hotspot 10% and NED traffic patterns with varying average packet arrival rates (APAR) are shown in Fig. 8.6. It can be observed for all the traffic patterns, that the network with proposed architecture saturates at higher injection rates. The reason being that, by utilizing the provided information via the ARB-NET, the *AdaptiveXYZ* routing increases the bus utilization and makes the load, balanced. In the case of *Z-DyXY* routing and *AdaptiveZ-DyXY* routing, the Hybrid 3D NoC-Bus mesh architecture cannot deliver the desired performance because of bandwidth limitations. For the proposed architecture, bandwidth limitations are managed by the separated monitoring network without increasing the main 3D NoC communication workload.

8.5.2 Videoconference Application

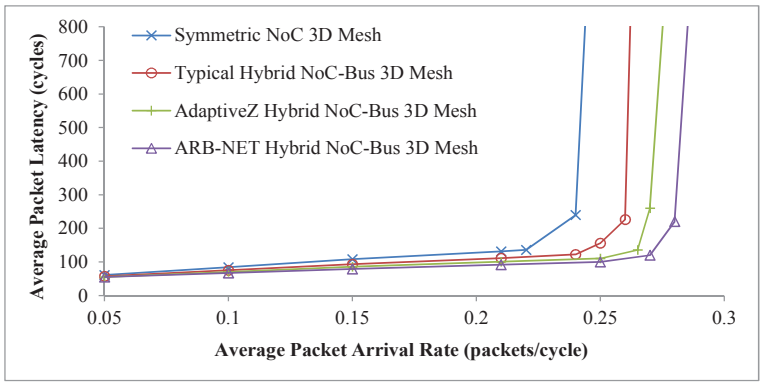
For realistic traffic analysis, we used the encoding part of videoconference application with sub-applications of H.264 encoder, MP3 encoder and OFDM transmitter presented in Chapter 5. The video stream used for simulation purposes was of size 300×225 pixels and each pixel consists of 24 bits. Thus each video frame is composed of 1.62 Mbits and can be broken into 8400



(a) Under uniform traffic profile



(b) Under hotspot 10% traffic profile



(c) Under NED traffic profile

Figure 8.6: Latency versus average packet arrival rate on a 3×3×4 mesh for different Hybrid 3D NoC-Bus mesh architectures

data packets each of size 7 flits including the header flit. The data width was set to 64 bits. We modeled the application graph, mapping strategy, frame rate, buffer size, number of nodes, layers and generated packets, supply voltage and clock frequency used in Chapter 5 for the real application simulation. In order to implement adaptive routing, routers with 2 VCs per input port were used. The application that is mapped to 3×3×3 3D-mesh NoC is shown in Fig. 6.11.

To estimate the power consumption, we used the high-level NoC power simulator presented in Chapter 5. The simulation results for power and performance of the video conference encoding application are shown in Table 8.2. The proposed ARB-NET-based architecture using the *AdaptiveXYZ* routing algorithm showed 19%, 9%, and 4% drop in power consumption over the Symmetric 3D-mesh NoC, typical Hybrid 3D NoC-Bus mesh, and *AdaptiveZ* Hybrid 3D NoC-Bus mesh architectures, respectively. Similarly, 29%, 17%, and 10% reduction in APL over the Symmetric 3D-mesh NoC, typical Hybrid 3D NoC-Bus mesh, and *AdaptiveZ* Hybrid 3D NoC-Bus mesh architectures was also observed for our proposed architecture. Since the proposed architecture using *AdaptiveXYZ* routing algorithm can balance the load distribution among all vertical buses much better than *Z-DyXY* and *AdaptiveZ-DyXY* routing, the average packet latency for all routers is smaller when compared with other counterparts. The achieved latency and power reductions also lead to a considerable decrease in power-delay product (PDP) of the system.

To demonstrate the efficiency of the ARB-NET monitoring platform for system reliability, the network running the video conference application presented in Chapter 7 with one faulty vertical bus was simulated. The rest of the system parameters were kept unchanged. As shown in Fig. 7.8, we assumed that the bus connecting *Stream Mux Mem*, *Padding for MV Computation*, and *YUV Generator* components is faulty because there is a high inter-layer communication via this bus. Thus, the ARB-NET node of the faulty bus asserts '0' on the *wait* signal and informs the neighboring ARB-NET nodes not to consider this faulty bus in their intra-layer routing. The power consumption and average packet latency of the *AdaptiveZ* Hybrid 3D NoC-Bus mesh and the proposed ARB-NET-based architectures with one faulty bus are shown in Table 8.2 (last two rows of the table). The figures given in the table show that, the proposed architecture can tolerate a single faulty bus with lower latency and power overhead.

8.5.3 Thermal Analysis

We study the impact of the proposed *AdaptiveXYZ* routing algorithm on the chip temperature of a 3×3×3 NoC-based system using the thermal model presented in Section 8.4. To this end, we have imported the physical floor-

Table 8.2: Power Consumption and Average Packet Latency

3D NoC Architecture	**Avg. Power Cons. (W)**	**APL (cycles)**
Symmetric NoC 3D Mesh	3.195	117
Hybrid 3D NoC-Bus	2.832	100
Rahmani et al. [133] Hybrid 3D NoC-Bus Mesh	2.671	92
ARB-NET Hybrid 3D NoC-Bus Mesh using *AdaptiveXYZ* routing	2.603	86
Rahmani et al. [133] Hybrid 3D NoC-Bus Mesh (1 faulty bus)	2.847	103
ARB-NET Hybrid 3D NoC-Bus Mesh using IL Fault Tolerant *AdaptiveXYZ* routing (1 faulty bus)	2.663	89

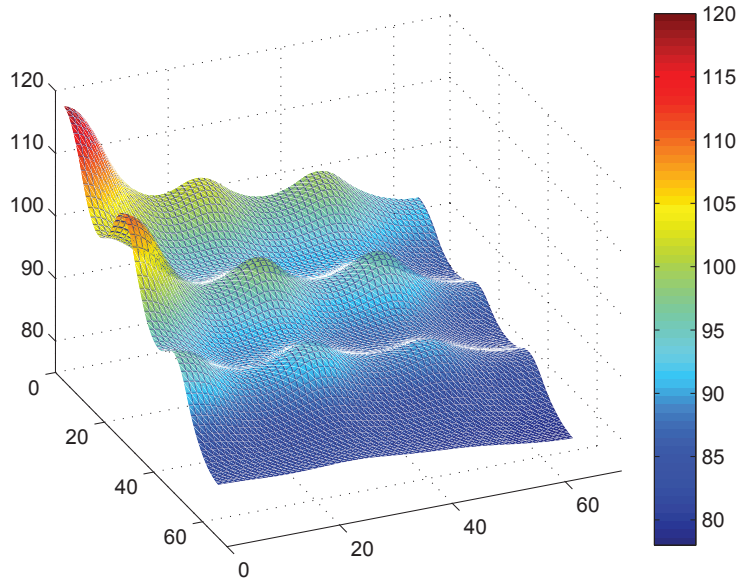
Table 8.3: Layer temperature profile of the Hybrid 3D NoC-Bus Mesh based system running the video conference application

Layer ID	**Peak Temperature (°C)**	**Min Temperature (°C)**
Layer 0	114.0	80.8
Layer 1	113.5	77.0
Layer 2	93.5	69.6

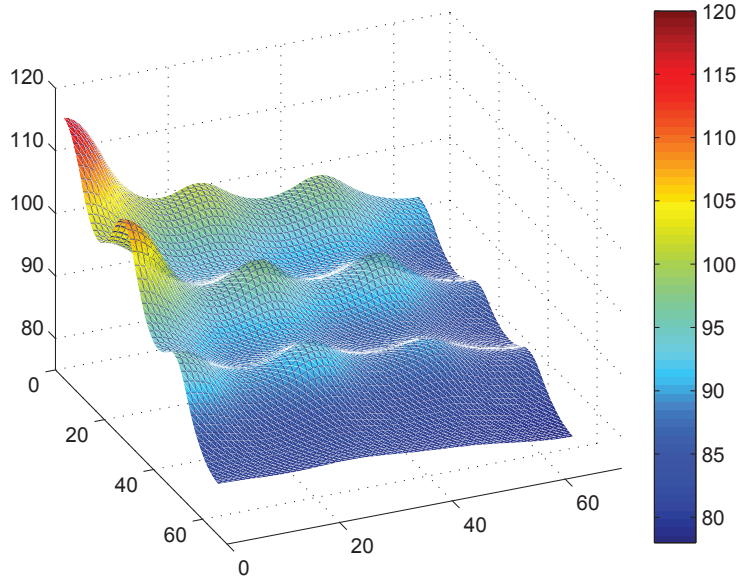
Table 8.4: Layer temperature profile of the proposed Hybrid 3D NoC-Bus Mesh-based system running the video conference application (*AdaptiveXYZ* routing)

Layer ID	**Peak Temperature (°C)**	**Min Temperature (°C)**
Layer 0	111.9	80.6
Layer 1	111.3	76.8
Layer 2	92.1	69.4

plan and the obtained power trace file to the thermal simulator, and estimate the temperature profile for each layer. The results of the thermal simulations on normal Hybrid 3D NoC-Bus Mesh based and the proposed ARB-NET based Hybrid 3D NoC-Bus Mesh-based (*AdaptiveXYZ* routing) systems running the videoconference encoding application are shown in Table 8.3 and Table 8.4, respectively. These tables depict the peak and minimum temperatures for each silicon layer after the transient simulation run. Layer 0 is considered to be the one which is farther from the heatsink in our thermal model. The comparison between these tables shows the effectiveness of temperature optimization of the proposed routing strategy. As expected, moving from the traditional 3D NoC to the proposed ARB-NET based 3D NoC causes the peak temperature of the chip to decrease. The significant importance of the proposed routing algorithm is the mitigation of



(a) For the normal routing



(b) For the proposed *AdaptiveXYZ* routing

Figure 8.7: Steady-state grid level thermal maps for the die 1 (layer 0)

noticeably exacerbate performance and reduce the lifetime of the chip. The figures given in Table 8.3 and Table 8.4 show that the proposed technique improves the peak chip temperature with negligible area overhead. The

Table 8.5: Hardware Implementation Details

Component	**Area (μm^2)**
Typical 6-Port Router with 2 VCs (*Z-DyXY*)	92194
Rahmani et al. [133] 6-Port Router with 2 VCs (*AdaptiveZ-DyXY*)	93591
Proposed 6-Port Router with 2 VCs (*IL FT AdaptiveXYZ*)	93914
Typical Bus Arbiter for a 3-layer NoC	267
Rahmani et al. [133] Bus Arbiter for a 3-layer NoC	694
ARB-NET Bus Node for a 3-layer NoC (Arbiter + Monitoring)	1534

improvement is up to 2.1°C for the realistic application. For our system with not much interlayer communication, the reduction of 2.1°C is quite an achievement. We expect more improvement for an application with more interlayer communication. Assuming that the mapping of task is predefined and the computation power cannot be migrated, the *AdaptiveXYZ* routing offers a sensible peak temperature improvement by globally balancing the load across all dimensions.

Fig. 8.7 shows the steady state grid level thermal maps of the die 1 (Layer 0) for both the normal and our proposed adaptive routing approach. On this layer the efficacy of our proposed adaptive routing approach is seen quite clearly. It can be observed that the drop in the maximum temperature in this layer with our proposed routing approach is around 2.3°C. Other layers are not shown here because the maximum temperature differences in those layers with normal and *AdaptiveXYZ* routing is not much. This is due to the presence of a very efficient heatsink which takes most of the heat generated in the system and transfers it to the ambient.

8.5.4 Hardware Overhead

We conclude the simulation results with a qualitative area comparison with the other work, most closely related to ours. The area of the different routers was computed once synthesized on CMOS 65*nm LPLVT STMicroelectronics* standard cells using Synopsys Design Compiler. The typical, the *AdaptiveZ-based*, and the proposed bus arbiters were also synthesized to illustrate the area overhead of the controllers. The layout area of a typical 6-port NoC router, the *AdaptiveZ* router, the proposed *AdaptiveXYZ* router, and the bus controllers are listed in Table 8.5. Note that for all the routers, the data width and buffer depth were set to 32 and 8 bits, respectively. The figures given in the table reveal that the area overheads of the proposed routing unit and the bus control module (ARB-NET node) are negligible.

8.6 Summary

In this chapter, a generic monitoring and management infrastructure for Hybrid 3D NoC-Bus mesh architectures was proposed to enhance the system management capability, thereby improving the overall NoC performance, power consumption, and reliability. The proposed monitoring platform called ARB-NET comprises of a network of monitoring nodes in the middle layer of the 3D NoC, which efficiently manages the entire network without any overhead on the main data network. The monitoring nodes are integrated in the bus arbiters and benefit from the available monitoring information generated by each transaction as well as the information provided by the connected routers. Further, to assess the efficiency of the proposed ARB-NET infrastructure, a fully adaptive and bus failure tolerant routing algorithm named *AdaptiveXYZ* was demonstrated. The results showed lower latencies and increased reliability for the proposed routing algorithm for high packet injection rates at places where the congestion in the NoC occurs.

Chapter 9

Partial Virtual-Channel Sharing NoC Architecture

It is generally accepted that buffers consume the largest fraction of dynamic and leakage power of a NoC node (router + link) [28][9]. Packet buffers consumes far more power than packet transmission [170]. Thus, increasing the utilization of buffers and reduction in their number and size reduces area and power consumption. As discussed in Chapter 2, to avoid potentially resulting deadlocks in wormhole switching policy, virtual channels are introduced. By allocating each packet to an individual buffer, flits from multiple packets may be sent in an interleaved manner over a single physical channel. This improves the throughput and reduces the average packet latency by allowing blocked packets to be bypassed.

A well designed network exploits available resources to improve performance while incurring minimum overhead [7]. Buffer utilization can be enhanced by sharing buffers among ports. Router architectures with full buffer sharing can deliver high throughput, but this comes at the expense of area and power consumption. Hence, it is necessary to devise a technique that enables judicious tradeoff between performance, power and area. In this chapter, a novel architecture for NoC routers with partial virtual channel sharing is proposed.

9.1 Resource Management Architectures

There have been many efforts to enhance resource utilization in NoC based systems. Lan *et al.* [84] address buffer utilization by making the channels bidirectional and show significant improvement in system performance. In this case, each channel controller has two additional tasks: dynamically configuring the channel direction and allocating the channel to one of the routers. These additional tasks make the controller circuit complex. There

is a 40% area overhead over the typical NoC router architecture due to double crossbar design and control logic. This also causes additional power consumption.

Nicopoulos *et al.* [106] present a Dynamic *Virtual Channel Regulator* (ViChaR) for NoC routers. The authors improve buffer utilization by using the unified buffered structure (UBS) instead of individual, statically partitioned FIFO buffers. UBS provides each router port with a variable number of VCs, depending to the traffic load. The architecture achieves around 25% improvement in system performance at a small cost of power consumption. However, the architecture enhances buffer utilization only when a port is under heavy traffic load. If there is no load, the buffer resources cannot be utilized by neighboring overloaded ports.

Ramanujam *et al.* [138] introduce a distributed shared buffer (DSB) NoC router architecture. The proposed architecture shows a significant improvement in throughput at the expense of area and power consumption due to its extra crossbar and complex arbitration scheme.

The main goal of this work is to design a NoC architecture which offers good communication performance (throughput, low latency) and minimizes design overheads through efficient resource utilization. The proposed architecture has a well balanced tradeoff between performance, power consumption and area.

9.2 Resource Utilization Analysis

Efficient resource utilization is necessary to execute a given application with minimized overhead. The first step towards enhancing the utilization of interconnection resources consists of examining the purpose of each network resource individually. If multiple applications execute on an MPSoC simultaneously, the traffic pattern is unpredictable and makes it difficult to analyze utilization of individual resources. The routing algorithm controls the utilization of communication channels. The system then utilizes the router resources according to the load on the incoming channels. We employ link load analysis to provide the basis for determine the utilization of these router resources. To do so, synthetic and application based benchmarks are used.

9.2.1 Link Load Analysis with Synthetic Traffic

In synthetic traffic analysis, the average load for each link is determined for a variety of traffic patterns. In our case, uniform, transpose, bit complement and negative exponential distribution (NED) traffic is analyzed with XY routing. In the uniform traffic pattern, a node sends a packet to any other node with an equal probability while in the transpose traffic pattern, each node (i,j) communicates only with node (j,i). For bit complement traffic

load, each node (i,j) communicates only with node $(M\text{-}1\text{-}i,N\text{-}1\text{-}j)$, if mesh size is $M \times N$. The NED is a synthetic traffic model based on Negative Exponential Distribution where the likelihood that a node sends a packet to another node exponentially decreases with the hop distance between the two cores [130]. Figure 9.1 shows the percentage load for each link on the network for different traffic patterns, measured by Equation 9.1.

$$L_{(i,j)\rightarrow(k,l)} = \frac{\text{TLL: } (i,j) \rightarrow (k,l)}{\text{TNL}} \tag{9.1}$$

where,

$$\text{TLL: } (i,j) \rightarrow (k,l) = \sum_{\substack{0<x<(M-1)\\0<x'<(M-1)}} \sum_{\substack{0<y<(N-1)\\0<y'<(N-1)}} \begin{cases} (S(x,y), D(x',y') \mid \\ via(i,j)) \text{ Then } via(k,l) \end{cases}$$

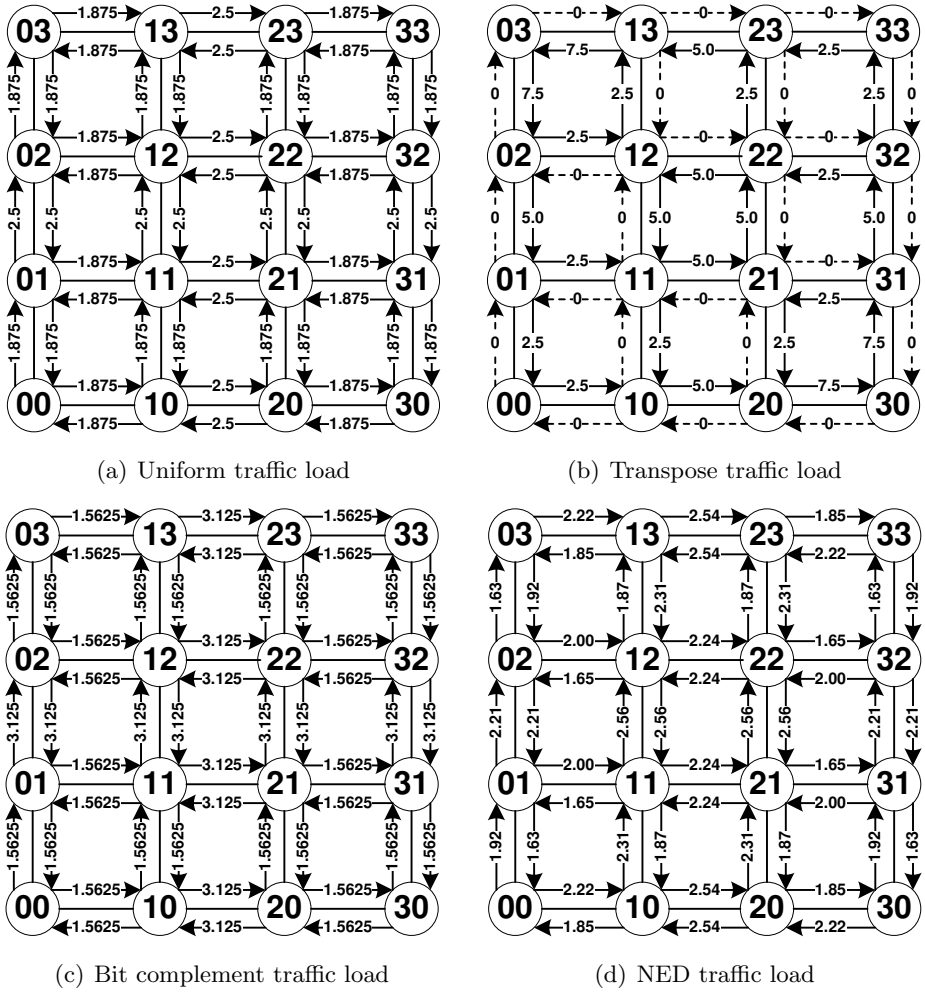
and

$$\text{TNL} = \sum_{\substack{0<m<(M-1)\\0<n<(N-1)}} \sum_{\substack{0<o<(M-1)\\0<p<(N-1)}} L_{(m,n)\rightarrow(o,p)}$$

To measure the total link load (TLL) on a specific link directed from node (i, j) towards node (k, l), the traffic load from the source nodes represented by S(x, y) routed via node (i, j) and then via node (k, l) towards destination nodes represented by D(x', y') is considered. The destination node D(x', y') could be the node (k, l) because these packets will contribute to the link load for the link directed from node (i, j) towards node (k, l). For total network link load (TNL), the link load of all the interconnection links is summed up. The expression is topology independent and can be extended to any number of dimensions.

The normalized link load percentage computed using Equation 9.1 for uniform, transpose, bit complement and NED traffic loads are shown in Figures 9.1(a), 9.1(b), 9.1(c) and 9.1(d), respectively with XY routing logic. Consider the node '12' in Figure 9.1(b). The input ports to receive data from nodes '11' and '13' are not used at all during the whole simulation independent of the total simulation time. But the input ports from left and right receive the traffic load. The traffic load from node '22' is two times the load from node '02'. The link from node '22' towards '12' is overloaded but cannot utilize the available resources of other ports. Similar behavior can be observed for odd-even routing.

In all investigated cases, some input ports are overloaded as compared to other ports. In order to balance the load and to enhance the resource



(a) Uniform traffic load

(b) Transpose traffic load

(c) Bit complement traffic load

(d) NED traffic load

Figure 9.1: Traffic load analysis for XY-routing

utilization, resources can be shared among over- and underutilized input ports. The threshold for distinguishing between over- and underutilization is selected in such a way that half of the ports have a higher load value and the other half has a lower load value.

The resources could be shared among all the input ports, but this would require large crossbar switches, which increases power consumption, area and switching delay. The other option is sharing the resources among multiple ports (but not among all ports) so that loads are balanced, resource utilization is improved, and throughput is close to an architecture with full VC buffer sharing.

In higher-dimensional NoCs (e.g. 3D NoCs), the partial virtual-channel

sharing (PVS) approach benefits from the increased number of router ports, which opens more grouping options as listed below:

$$\begin{aligned}
\text{Typical 2D-mesh:}\mathtt{R}(5) &:< (5), (4,1), (2,2,1), \ldots, (1,1,1,1,1) > \\
\text{Stacked 3D-mesh[88]:}\mathtt{R}(6) &:< (6), (5,1), (2,2,2), \ldots, (1,1,1,1,1,1) > \\
\text{Typical 3D-mesh:}\mathtt{R}(7) &:< (7), (6,1), (3,3,1), \ldots, (1,1,1,1,1,1,1) >
\end{aligned}$$

Where R(n) represents the router with n ports. $< (p, q, ...) \ldots (f, g, ...) >$ represents the set of different grouping options. Each grouping option is denoted as a tuple a group sizes; for example, (3, 2, 2) represents one group of 3 ports and two groups of 2 ports. Ports in the same group can share their resources.

9.2.2 Application Specific Link Load Analysis

The MPEG4 application introduced in [77] has been selected for resource utilization analysis. The NoC-mapped application and its bandwidth requirements are shown in Figure 9.2. Consider, for example, the link loads of the DR-SDRAM node. If its East and South ports share their resources, the heavy load value 942 MB/s from East can utilize the resources of the South port, which receives a smaller load value of 60.5 MB/s. Thus, sharing communication resources among multiple ports can balance the input load on all ports without increasing crossbar size too much. The average load on input ports is comparable to what can be achieved with full sharing.

The ports are grouped so that the load sums of the different groups are balanced. The port with maximum load should be grouped together with the port of minimum load. Grouping it with an average load port would not make sense because such port does not have free resources or does not require extra resources. For load balancing, the selection of ports to share the resources should be made at design time according to Algorithm 7, described and analyzed below.

The input parameters of the algorithm are the number of router ports, P, the number of VCs per port, V, the vector representing the input bandwidth requirements for each router port, L, and the number of partitions (groupings) of ports, d.

The input ports are grouped to share the buffers in such a way that the total incoming load for the whole group approaches the value of average bandwidth requirements per VC times the number of VC buffers in the current group. The average bandwidth requirements per VC are represented by l in algorithm 7, which can be computed by summation of the bandwidth requirements of all the router input ports divided by the total number of

available VC buffers. The total number of VC buffers in the router are $P \times V$. The number of ports in a group, sharing the VC buffers is represented by vector S. The sum of all the S values is equal to P. The total bandwidth requirements for each group of ports sharing the VC buffers are represented by vector W. Each value of W is equal to l times the corresponding value of S. Different combinations of ports are tested by using a *for* loop such that the sum of the bandwidth requirements of the ports in the combination is close to the corresponding value of W. The whole process is repeated until the best combination is achieved and the difference between W and the sum of communication bandwidth requirements of the ports in the group is minimal. Finally, the grouping combinations, C, are returned.

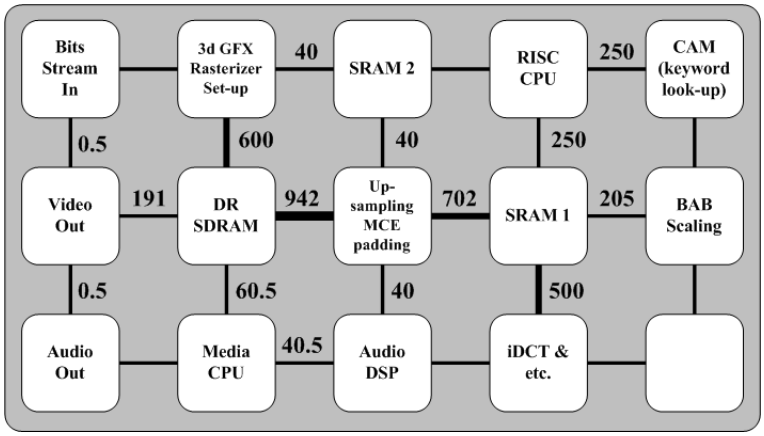


Figure 9.2: MPEG4 application [77]

The proposed algorithm can support any topology including irregular topologies with any number of ports, P. For example, Murali *et al.* [102] propose an application specific power efficient topology which requires an eleven port router. If algorithm 7 is used to generate the grouping combinations according to the input load requirements, further system performance enhancement can be achieved as the algorithm chooses the optimum from the following set of potential groupings:

$$\begin{aligned} \mathtt{R}(11) :< & (11), (10,1), (6,5), (5,5,1), (4,4,3), \\ & (3,3,3,2), \ldots, (1,1,1,1,1,1,1,1,1,1,1) > \end{aligned}$$

Algorithm 7 Grouping $(P.V, d, \mathbf{L})$

P = Number of router ports.
V = Number of VCs per port.
d = Number of partitions.
$\mathbf{L} = [L_1, L_2, \ldots, L_P]$;
$//L_i$ is the input bandwidth requirement for port i.

define $\mathbf{S} = [S_1, S_2, \ldots, S_d]$;
$//S_i$ is the number of ports sharing the VCs at segment i.

define l = Average bandwidth requirement per VC.

define $\mathbb{C} = [\mathbf{C_1}, \mathbf{C_2}, \ldots, \mathbf{C_P}]$;
$//\mathbf{C_i}$ contains the port IDs in segment i.

define $\mathbf{W} = [W_1, W_2, \ldots, W_d]$;
$//W_i$ is total bandwidth requirement for partition i.

$L_{ACC} = 0$; $//\mathbf{L}$ Accumulator
for $i = 1$ to P **do**
    $L_{ACC} \leftarrow L_{ACC} + L_i$;
end for
$l \leftarrow L_{ACC}/(P \times V)$;
loop
  Random $\mathbf{S} \mid \sum_{i=0}^{d} S_i = P$;
  $\mathbf{W} \leftarrow l \times \mathbf{S}$;
  for $j = 1$ to d **do**
    $\mathbf{C_j} \leftarrow$ Combination of ports $\mid \sum_i \mathbf{L}(\mathbf{C_i}) \sim W_j$;
  end for
  if $\forall(l \times sizeOf(\mathbf{C_i})) \sim \sum \mathbf{C_i}$ **then**
    exit;
  end if
end loop
return $\mathbb{C}$;

9.3 The Proposed Partial Virtual Channel Sharing Approach

Maximum VC utilization could be achieved by sharing among all the input ports. However, full sharing increases the control logic complexity and power consumption. Thus, a tradeoff between resource utilization and power consumption is needed.

This tradeoff can be achieved with the PVS approach by forming groups with a limited number of input ports that share resources according to the communication requirements. With this technique, the buffer utilization is increased and comes close to the utilization level of the fully shared architecture without suffering its significant silicon area and power consumption overhead.

9.3.1 The Input Controller and Buffer Allocation

The PVS approach is implemented on the input ports of the router. The buffer utilization is enhanced by dynamically allocating free buffers to overloaded ports. The definition of which buffers are shared among which ports is parameterized and can be adjusted to match any number of input ports according to the topology requirements. Only the processing element uses dedicated buffers for packet injection which are not shared with any other router ports.

In the PVS approach, the input control logic is responsible for buffer allocation and receiving the data packets. An example of the PVS architecture, with two groups of two channels sharing VC buffers, is shown in Figure 9.3. Within each group, each port has its own (distributed) routing logic whereas VC allocation is centralized. Both, VC allocator and routing logic operate independently, without communicating with control logic of other groups. The task of the VC allocator is to keep track of free buffers and to allocate them to the incoming traffic. After allocation, the routing logic computes the route for the packet and controls the crossbar for packet switching. In Figure 9.3, *Routing Logic_U* refers to the control logic for the upper group of virtual channel buffers and *Routing Logic_L* refers to the control logic for the lower group.

Virtual Channel Selection for Sharing

For load balancing, the selection of ports to share their VC buffers should be made on the basis of the number of router ports, the number of VCs per port, input bandwidth requirements for each input port and the number of groups (sharing VCs). The input ports are grouped to share the buffers in such a way that the total incoming load for the whole group approaches the average value of bandwidth requirements per VC times the number of VC buffers in the current group as described in Algorithm 7.

The MPEG4 application presented by [77] was discussed in section 9.2.2 for link load analysis. In the scenario described, the East and South ports of the DR-SDRAM node should share their VC buffers, and the West and North ports should form a second sharing group. The heavy traffic amounting to 942 MB/s at the East port can utilize the resources of the South port

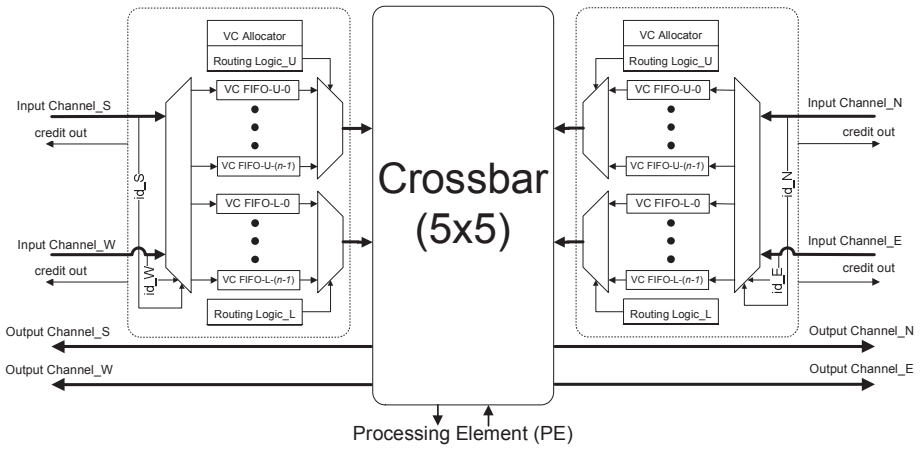


Figure 9.3: Proposed PVS approach for conventional Virtual Channel Architecture

which receives less traffic, 60.5 MB/s only.

Flow Control

The PVS approach uses wormhole switching with partial sharing of virtual channel buffers. Due to sharing, race conditions may occur. This happens, for example, if only one buffer is available and multiple channels in the sharing group request ownership of the same buffer. To avoid such situations, the channels in a sharing group are assigned default priority values. For application specific NoCs, the priority is proportional to the channel bandwidth requirements. The channel with higher priority value is allowed to use the buffer.

Routing Algorithm

Different routing algorithms can be used with the PVS technique. However, there is a possibility of deadlock when more than one input ports share their VCs. For instance, if the North and East input ports share their resources and all the VCs are occupied by the flits coming from the East and going to the West, then the flits traversing from the North to the South direction in the upstream router (i.e., North router) have to wait. If the scenario results in a cyclic dependency, then a deadlock will occur.

In order to avoid deadlock in static routing algorithms such as static XY, at least one VC should be dedicated for each input port. Therefore, for a PVS unit which has w number of VCs being shared among u ports, u dedicated VCs are needed and the remaining (w-u) VCs can be shared. Similarly, since for the dynamic routing algorithms at least two input VCs

are required for each input port to avoid deadlock [105], *w-2u* VCs can be shared.

9.3.2 The Output Controller

The *Output* part consists of a typical $N \times N$ crossbar switch with central control logic, where N is the total number of ports including the local PE port. The crossbar size can be customized according to the topology requirements. Wormhole switching is used for packet transmission, which makes efficient use of buffer space as the number of flit buffers per VC can be less than the packet size [40].

9.3.3 Comparison with Existing Architectures

A 5-port router with two unidirectional links per port and 10 internal buffers has been investigated and has been compared with other NoC architectures. Table 9.1 shows the result of this comparison.

The typical VC NoC represents a conventional virtual channel NoC architecture with 2 VCs per port which uses unidirectional channels to communicate with neighboring routers. Thus, two channels are required between two neighboring routers for two way communication. In case of heavy traffic load on a certain port, the typical virtual channel architecture can provide only 2 VCs to receive the packets on that port. The PVS-NoC can provide 4 VCs to the same port under heavy traffic load on the same port. Thus, the VC availability has been doubled with slight overhead of crossbar size.

BiNoC has two bidirectional channels per port, whose direction can be switched at run-time to meet communicate requirements [84]. As compared to the BiNoC architecture with 10 in-out channels, the PVS-NoC approach provides 5 input and 5 output physical channels. PVS-NoC can provide 4 input and 4 output VCs per physical port whereas BiNoC has only two physical channels per port, without VCs. The option of direction selection is provided at the cost of a large crossbar switch. Another issue to be addressed here is scalability. The number of VC buffers can be selected according to the application and topology requirements for our proposed architecture. To insert a new VC, the buffer and a controller are needed without any modification in existing logic, and at the cost of only a slight increase in crossbar resources. To insert a new buffer in the BiNoC architecture, a separate buffer allocator is required and the crossbar is significantly larger.

The Distributed Shared Buffer (DSB) router architecture has been compared with the other architectures. The DSB-175 and DSB-300 router architectures have been described in [138]. To make an exact comparison, we define a DSB-160 architecture in accordance with [138]. The DSB-160 is a router with 160 flits of aggregate buffering. The buffers are divided between

Table 9.1: Comparison with existing NoC router architectures

Architecture⇒ Resource⇓	Typical VC NoC	BiNoC	DSB-160	FVS-NoC	PVS-NoC
Number of Buffers	10	10	10	10	10
Channels/Direction	1-in 1-out	2-inout	1-in 1-out	1-in 1-out	1-in 1-out
Max. VCs/Channel	2	1	2	10	4
Buffer Size	16 flits	16 flits	8 flits	16 flits	16 flits
Total Buffer Size	160 flits	160 flits	160 flits	160 flits	160 flits
Crossbar Size	4(1×2) + 5×5	10×10	2(5×5)	5×10 + 10×5	2(2×4) + 5×5

5 middle memory banks with 16-flit buffers per bank and aggregate 80-flit input buffers comprising two 8-flits buffers (VCs) at each input port. The five memory banks are not considered in comparison Table 9.1 because only one flit can be written into and read from a middle memory in the DSB architecture, which reduces the utilization of memory banks. Thus the static power consumption without increasing the system performance is the major overhead of DSB architecture as compared to the PVS-NoC.

For further comparison, the Fully Virtual channel shared NoC (FVS-NoC) architecture has been investigated. In this architecture, any of the 10 VC buffers can be allocated to any input port. FVS-NoC channels are unidirectional. The architecture provides the maximum utilization of VC buffers at the cost of significantly larger crossbars, which makes the solution area and power expensive as compared to the proposed architecture.

9.4 Simulation results

To demonstrate performance characteristics of the proposed architecture (PVS-NoC), a cycle-accurate NoC simulation environment has been implemented in VHDL. The packets have a fixed length of seven flits, the buffer size is eight flits, and the data width is set to 32 bits. The 5×5 2D mesh topology is used for interconnection. Each input port has 4 VCs. With the same parameters, typical virtual channel and FVS-NoC architectures are analyzed. The static XY routing algorithm is used.

The PVS approach with grouping combination of (2, 2, 1) is used in the simulation, where '1' represents the buffer dedicated to the local PE. The critical path limits the operating frequency of the PVS router, which is 3.5% less than the operating frequency of the baseline router. For the grouping combination of (3, 1, 1), the maximum operating frequency is around 9% less than the maximum operating frequency of the baseline router.

9.4.1 Synthetic Traffic Analysis

We compare the simulation results in terms of average packet latency (APL) and saturation points for a network using typical, PVS, and FVS virtual channel management policies.

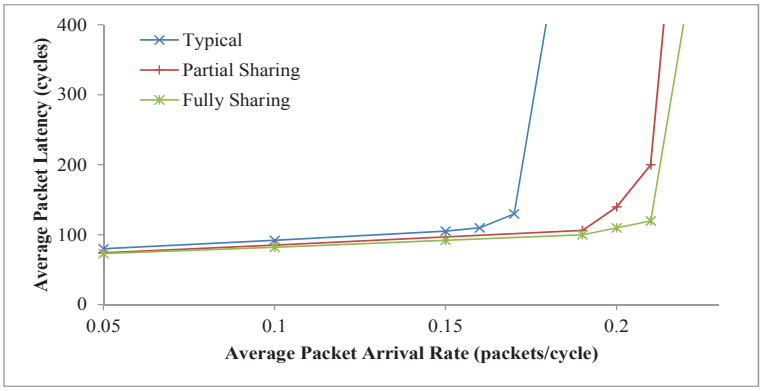
In traffic analysis, we have evaluated the performance of the network using latency curves as a function of the packet injection rate. For each simulation, the packet latencies are averaged over 50,000 packets. Latencies are not recorded for the first 5,000 cycles to allow the network to stabilize. In the simulations, uniform, transpose and NED [130] traffic patterns are used.

The latency curves for uniform, transpose and NED traffic patterns are shown in Figure 9.4. It can be observed for all the traffic patterns, the PVS-NoC architecture saturates at higher injection rates as compared to the typical VC architecture, but at slightly lower rates than FVS-NoC architecture. The proposed architecture manages bandwidth limitations by proper resource utilization and by making the load more balanced, without increasing the communication resources. The saturation point of PVS-NoC is just before FVS-NoC because FVS-NoC provides more buffer utilization by sharing the VC buffers among all input ports. However FVS-NoC is not a power efficient solution as verified below with application traffic.

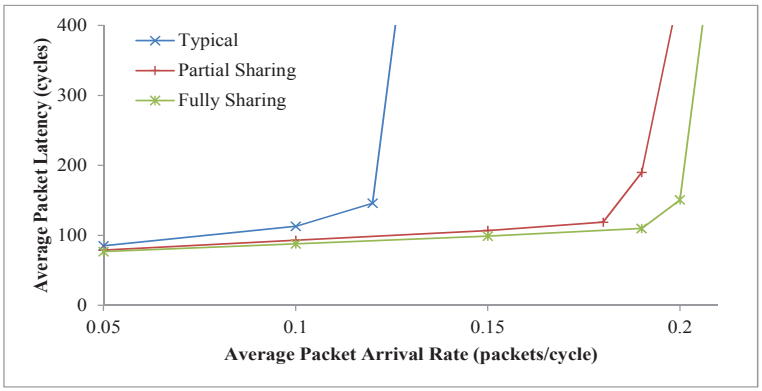
9.4.2 Videoconference Application

For realistic traffic analysis, we used the encoding part of videoconference application with sub-applications of H.264 encoder, MP3 encoder and OFDM transmitter presented in Chapter 5. The video stream used for simulation purposes was of size 300×225 pixels and each pixel consists of 24 bits. Thus each video frame is composed of 1.62 Mbits and can be broken into 8400 data packets each of size 7 flits including the header flit. The data width was set to 64 bits. We modeled the application graph, mapping strategy, frame rate, buffer size, number of nodes, layers and generated packets, supply voltage and clock frequency used in Chapter 5 for the real application simulation. In order to implement adaptive routing, routers with 2 VCs per input port were used. The application that is mapped to 3×3×3 3D-mesh NoC is shown in Fig. 6.11. By applying Algorithm 7, an individual grouping combination has been determined for each node according to the bandwidth requirements.

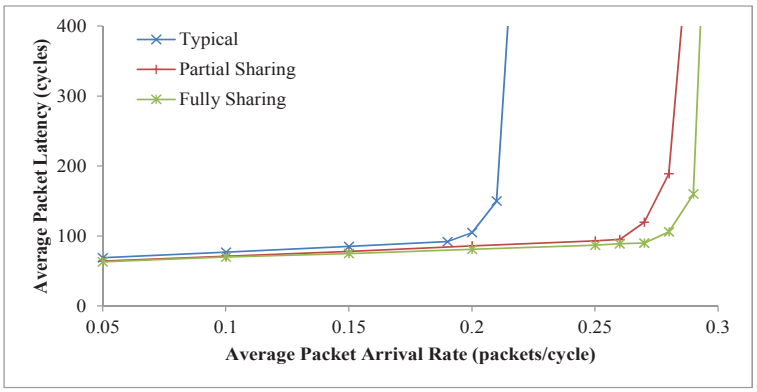
To estimate the power consumption, we used the high-level NoC power simulator presented in Chapter 5. The simulation results for average packet latency (APL), power consumption and average router silicon area for the video conference encoding application are shown in Table 9.2.



(a) Uniform traffic load



(b) Transpose traffic load



(c) NED traffic load

Figure 9.4: Average packet latency (APL) vs. Packet injection rate for 5×5 Mesh 2D NoC with (2, 2, 1) combination of PVS approach

Table 9.2: Experimental result for average power consumption and APL of video conference encoder application

3D NoC Architecture	Power Consumption (W)	Average Packet Latency (cycles)	Average Silicon Area (μm^2)
Typical Symmetric 3D NoC (7x7)	1.587	186	195154
PVS-3D-NoC	1.713	144	203596
FVS-3D-NoC	2.182	136	220282

9.4.3 Area Comparison

The area of the 3D-symmetric-mesh-based routers (with 7×7 crossbars) have been determined by synthesis to CMOS 65*nm LPLVT STMicroelectronics* standard cells using Synopsys Design Compiler. The results for the average router silicon area of a 3×3×3 3D NoC are shown in Table 9.2. Each input port has 4 VC buffers. The buffers' size is 8 flits and the data width is set to 32 bits. Here, the average silicon area is reported because different sharing combinations for PVS according to algorithm 7 have different crossbar sizes and thus different silicon area. The figures given in the table demonstrate that the area overhead of the proposed PVS technique more reasonable compared to a fully shared virtual channel technique.

The PVS-NoC shows around 21% reduction in power consumption and around 7% reduction in silicon area but around 6% higher APL over the FVS-NoC architecture. On other hand, the PVS-NoC shows approximately 22% reduction in APL value but around 8% higher power consumption and around 4% larger silicon area over the symmetric 3D-NoC architecture. Thus, the proposed PVS-NoC architecture provides a superior tradeoff between APL, power consumption and silicon area.

9.5 Trade-offs

As already discussed in previous sections, the proposed PVS approach makes a tradeoff between system performance, area and power consumption. As shown with synthetic traffic analysis for a 2D mesh NoC, the saturation point of the PVS latency curves comes close to the fully shared virtual channel architecture. On the other hand, PVS saturates at significantly higher packet arrival rate as compared to the typical virtual channel architecture. For uniform traffic load, the PVS architecture does not show significant improvement in saturation point for packet injection rate compared to typical VC architectures. The reason for this is that all the resources are equally

Table 9.3: PVS-NoC router silicon area for different grouping combinations

Grouping Combination	Silicon Area (μm^2)
Typical VC router (1,1,1,1,1)	145712
PVS router (2,2,1)	151412
PVS router (3,1,1)	153338
PVS router (3,2)	156180
PVS router (4,1)	159208
Full VC Shared router (5)	163742

loaded. The PVS based network has smaller average packet latency for transpose and NED traffic loads than the typical VC architecture because it is better able to balance link load as discussed in section 9.2.1. PVS with (2,2,1) sharing groups requires 7% less area than FVS. Compared to the typical VC architecture, PVS has only 3% area overhead. Silicon area with different PVS sharing combinations, again synthesized on CMOS $65nm$ *LPLVT STMicroelectronics* standard cells using Synopsys Design Compiler, are presented in Table 9.3. It can be observed that PVS area significantly increases as the VC sharing group size increases.

In case of video conference encoder application, the PVS architecture shows significant reduction in average packet latency with minor overhead of silicon area and power consumption. Moreover, the proposed architecture shows significant reduction in area and power consumption over the FVS architecture with minor overhead of average packet latency.

9.6 Summary

A novel NoC architecture with a better tradeoff between resource utilization, system performance and power consumption than conventional VC based architectures was presented. The proposed architecture has been simulated with uniform, transpose, and negative exponential distribution (NED) synthetic traffic. Also, the architecture has been simulated with video conference encoder application traffic. Simulation results show that the average packet latency of PVS-NoC is significantly lower compared to typical VC based NoC architecture and comes close to an architecture with full buffer sharing (FVS). In case of the video conference encoder application, the PVS-NoC consumes 21% less power and takes 7% less area than FVS. On the other hand, the PVS-NoC consumes only 8% more power and takes only 4% more area than a typical virtual channel architecture. We thus conclude that PVS offers a superior trade-off in providing near-maximum performance at near-minimum area and power cost.

Chapter 10

Conclusions

The advances in the semiconductor technology and the shrinking feature size in the deep submicron era enable the design of complex systems-on-chips (SoCs) composed of tens or hundreds of IP cores. At the same time, the applications are becoming more and more complex and have tight power and performance requirements. Achieving a satisfactory design quality under these circumstances requires higher degrees of support from communication resources. There is now much evidence to support the fact that the targeted on-chip multi-core systems are becoming increasingly power-constrained. Consequently, there is a great demand for power-efficient on-chip communication architectures in future multi-core design. Power consumption, as the substantial bottleneck of today's on-chip communication, has been the primary focus of this thesis, and to address this issue several power reduction and optimization techniques have been proposed and developed.

The GALS-based design styles can provide multiple Voltage/Frequency Islands by partitioning a system into isolated synchronous islands. The concept of Voltage/Frequency Islands has a potential to significantly reduce the system overall power consumption. To this end, we presented four efficient reconfigurable synchronous/bi-synchronous buffers for NoC-based multicore systems which can adapt their operation modes to either synchronous or bi-synchronous. Our analysis revealed that the reconfigurable FIFOs can help to achieve considerable savings in the average power consumption of NoC-based systems and latency improvement, when compared to a non-reconfigurable architecture.

As a viable alternative to the 2D planar chip, 3D integration technology offers greater device integration and shorter interlayer interconnects. In order to take advantage of the proposed low-power FIFOs, a low-cost 3D NoC architecture based on bidirectional bisynchronous vertical channels was proposed to mitigate high peak temperatures, power densities and area

footprints of vertical interconnects in 3D ICs. We achieved considerable improvements in terms of area of vertical interconnects. We further improved the architecture by presenting a forecasting-based dynamic frequency scaling technique for reducing the power consumption of the inter-layer communication.

The Hybrid 3D NoC-Bus Mesh multicore architecture, which is a hybrid between packet-switched network and a bus, was proposed to take advantage of the intrinsic properties of 3D ICs. In the conventional Hybrid 3D NoC-Bus Mesh, typical 6-port routers are utilized. We showed that by benefiting from the *LastZ* routing rule, it is possible to reduce the router size and accordingly router power consumption. Based on this rule, for the first time, we proposed ultra-optimized 5-port routers for 3D NoC. To further enhance our NoC-based multicore architecture, we proposed an efficient 3D NoC architecture to optimize performance, power consumption, and reliability of Hybrid 3D NoC-Bus Mesh system. The mechanism benefits from a congestion-aware and bus failure tolerant routing algorithm called *AdaptiveZ* for vertical communication.

The Hybrid 3D NoC-Bus Mesh architecture also provides a unique and hitherto previously unexplored way to implement an efficient system-wide monitoring network. In order to implement more scalable power optimization techniques, we proposed an integrated low-cost monitoring platform called ARB-NET for 3D NoC architectures which can be efficiently used for various system management purposes such as traffic monitoring, power and thermal management, and fault tolerance. To assess the efficiency of the proposed ARB-NET platform, a fully adaptive and bus failure tolerant routing algorithm named *AdaptiveXYZ* was demonstrated. The results showed lower latencies and power consumption and increased reliability for the proposed routing algorithm for high packet injection rates at places where the congestion in the NoC occurs.

Furthermore, we optimized the buffer utilization of the NoC routers because of the fact that buffers consume a large amount of power in the routers. In order to manage the power consumption of the virtual channel buffers, we presented a partial virtual channel sharing method for 2D NoC routers. The PVS approach was then extended for other network topologies, especially for 3D NoC routers.

10.1 Future Work

Our research will be extended by introducing a comprehensive power and thermal management hardware infrastructure which monitors the communication activity over time and provides finer granularity DVFS-based power and thermal control. The power management has a close correlation with

thermal characteristics of a system. This fact necessitates comprehensive thermal analysis of the implemented 3D NoC. To this end, we have carried out thermal analysis for 3D NoC-based chips. We published the achievements in the IEEE/EUROMICRO DSD conference [156]. Complementary thermal analysis considering the chip power density has been also done and the results have been published in the Elsevier ICCTSD conference [157]. These results are considerably essential to extend the proposed autonomic ARB-NET monitoring platform. We plan to implement more sophisticated monitoring and management techniques.

It is also noteworthy that in parallel with developing a monitoring platform for our 3D NoC, architectural refinement is required as well in many aspects. One of the most important issues is system reliability. For this purpose, we are improving performance sustainability and power consumption of the system under faulty situations by addressing fault tolerance issues for the proposed 3D NoC. We published the preliminary promising results in the IEEE/EUROMICRO DSD conference [85]. In addition, we plan to develop ways to recover processing elements in faulty situations in the network with minimum energy dissipation by using the proposed partial virtual-channel sharing approach.

Bibliography

[1] C. Ababei, Y. Feng, B. Goplen, H. Mogal, T. Zhang, K. Bazargan, and S. Sapatnekar. Placement and Routing in 3D Integrated Circuits. *IEEE Design Test of Computers*, 22(6):520–531, 2005.

[2] A. Adriahantenaina, H. Charlery, A. Greiner, L. Mortiez, and C.A. Zeferino. SPIN: a scalable, packet switched, on-chip micro-network. In *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*, pages 70–73, 2003.

[3] R.W. Apperson, Z. Yu, M.J. Meeuwsen, T. Mohsenin, and B.M. Baas. A scalable dual-clock FIFO for data transfers between arbitrary and haltable clock domains. *IEEE Transaction on Very Large Scale Integration Systems*, 15(10):1125–1134, 2007.

[4] M. Arjomand and H. Sarbazi-Azad. Voltage-Frequency Planning for Thermal-Aware, Low-Power Design of Regular 3-D NoCs. In *Proceedings of the International Conference on VLSI Design*, pages 57–62, 2010.

[5] J.H. Bahn, S.E. Lee, Y.S. Yang, J Yang, and N. Bagherzadeh. On Design and Application Mapping of a Network-on-Chip (NoC) Architecture. *Parallel Processing Letters*, 18:239–255, 2008.

[6] J. Bainbridge and S. Furber. Chain: A Delay-Insensitive Chip Area Interconnect. *IEEE Micro*, 22(5):16–23, 2002.

[7] J. Balfour and W.J. Dally. Design tradeoffs for tiled CMP on-chip networks. In *Proceedings of the 20th annual international conference on Supercomputing*, pages 187–198, 2006.

[8] K. Banerjee and A. Mehrotra. A power-optimal repeater insertion methodology for global interconnects in nanometer designs. *IEEE Transactions on Electron Devices*, 49(11):2001–2007, 2002.

[9] N. Banerjee, P. Vellanki, and K.S. Chatha. A power and performance model for network-on-chip architectures. In *Proceedings of the Design,*

Automation and Test in Europe Conference and Exhibition, volume 2, pages 1250–1255, 2004.

[10] L. Benini and G. De Micheli. Networks on chips: a new SoC paradigm. *Computer*, 35(1):70–78, 2002.

[11] K. Bernstein, P. Andry, J. Cann, P. Emma, D. Greenberg, W. Haensch, M. Ignatowski, S. Koester, J. Magerlein, R. Puri, and A. Young. Interconnects in the Third Dimension: Design Challenges for 3D ICs. In *Proceedings of the 44th ACM/IEEE Design Automation Conference*, pages 562–567, 2007.

[12] T. Bjerregaard and J. Sparso. A Router Architecture for Connection-Oriented Service Guarantees in the MANGO Clockless Network-on-Chip. In *Proceedings of the conference on Design, Automation and Test in Europe - Volume 2*, pages 1226–1231, 2005.

[13] B. Black, M. Annavaram, N. Brekelbaum, J. DeVale, L. Jiang, G. H. Loh, D. McCaule, P. Morrow, D. W. Nelson, D. Pantuso, P. Reed, J. Rupley, S. Shankar, J. Shen, and C. Webb. Die Stacking (3D) Microarchitecture. In *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 469–479, 2006.

[14] B. Black, D.W. Nelson, C. Webb, and N. Samra. 3D Processing Technology and Its Impact on iA32 Microprocessors. In *Proceedings of the IEEE International Conference on Computer Design*, pages 316–318, 2004.

[15] P. Bogdan, R. Marculescu, S. Jain, and R.T. Gavila. An Optimal Control Approach to Power Management for Multi-Voltage and Frequency Islands Multiprocessor Platforms under Highly Variable Workloads. In *Proceedings of the Sixth IEEE/ACM International Symposium on Networks on Chip*, pages 35–42, 2012.

[16] E. Bolotin, I. Cidon, R. Ginosar, and A. Kolodny. QNoC: QoS architecture and design process for network on chip. *Journal of Systems Architecture*, 50(2-3):105–128, 2004.

[17] A.H. Bowker and G.J. Lieberman. *Engineering Statistics*. Prentice Hall PTR, 1972.

[18] G. Campobello, M. Castano, C. Ciofi, and D. Mangano. GALS Networks on Chip: A New Solution for Asynchronous Delay-Insensitive Links. In *Proceedings of the Design, Automation and Test in Europe Conference and Exhibition*, pages 1–6, 2006.

[19] L.P. Carloni, P. Pande, and Y. Xie. Networks-on-chip in emerging interconnect paradigms: Advantages and challenges. In *Proceedings of the 2009 3rd ACM/IEEE International Symposium on Networks-on-Chip*, pages 93–102, 2009.

[20] A. Chakraborty and M.R. Greenstreet. Efficient self-timed interfaces for crossing clock domains. In *Proceedings of the 9th International Symposium on Asynchronous Circuits and Systems*, pages 78–88, 2003.

[21] C.S. Chang, K.A. Monnig, and M. Melliar-Smith. Interconnection challenges and the National Technology Roadmap for Semiconductors. In *Proceedings of the IEEE 1998 International Interconnect Technology Conference*, pages 3–6, 1998.

[22] Z. Changyun, G. Zhenyu, S. Li, R.P. Dick, and R. Joseph. Three-Dimensional Chip-Multiprocessor Run-Time Thermal Management. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(8):1479–1492, 2008.

[23] C.-H. Chao, K.-Y. Jheng, H.-Y. Wang, J.-C. Wu, and A.-Y. Wu. Traffic- and Thermal-Aware Run-Time Thermal Management Scheme for 3D NoC Systems. In *Proceedings of the 2010 4th ACM/IEEE International Symposium on Networks-on-Chip*, pages 223–230, 2010.

[24] D.M. Chapiro. *Globally-Asynchronous Locally-Synchronous Systems*. PhD thesis, Stanford University, October 1984.

[25] T. Chelcea and S.M. Nowick. Robust interfaces for mixed-timing systems. *IEEE Transaction on Very Large Scale Integration Systems*, 12(8):857–873, 2004.

[26] G. Chen and E.G. Friedman. Low-power repeaters driving RC and RLC interconnects with delay and bandwidth constraints. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 14(2):161–172, 2006.

[27] T.W. Chen, J.-H. Chun, Y.-C. Lu, R. Navid, W. Wang, C.-L. Chen, and R.W. Dutton. Thermal Modeling and Device Noise Properties of Three-Dimensional-SOI Technology. *IEEE Transactions on Electron Devices*, pages 656–664, 2009.

[28] X. Chen and L.S. Peh. Leakage power modeling and optimization in interconnection networks. In *Proceedings of the 2003 International Symposium on Low Power Electronics and Design*, pages 90–95, 2003.

[29] Y. Chen, J. Hu, and X. Ling. De Bruijn graph based 3D Network on Chip architecture design. In *Proceedings of the International Conference on Communications, Circuits and Systems*, pages 986–990, 2009.

[30] C.L. Chou, U.Y. Ogras, and R. Marculescu. Energy- and Performance-Aware Incremental Mapping for Networks on Chip With Multiple Voltage Levels. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 27(10):1866–1879, 2008.

[31] P. Choudhary and D. Marculescu. Power management of voltage/frequency island-based systems using hardware-based methods. *IEEE Transactions on Very Large Scale Integration Systems*, 17(3):427–438, 2009.

[32] J. Cong and Y. Zhang. Thermal via planning for 3-D ICs. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pages 745–752, 2005.

[33] A.K. Coskun, J.L. Ayala, D. Atienza, T.S. Rosing, and Y. Leblebici. Dynamic thermal management in 3D multicore architectures. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 1410–1415, 2009.

[34] D.E. Culler, J.P. Singh, and A. Gupta. *Parallel Computer Architecture: A Hardware/Software Approach*. Morgan Kaufmann, 1998.

[35] C. Cummings and P. Alfke. Simulation and synthesis techniques for asynchronous FIFO design with asynchronous pointer comparison. In *Proceedings of the SNUG-2002*, 2002.

[36] M. Dall'Osso, G. Biccari, L. Giovannini, D. Bertozzi, and L. Benini. Xpipes: a latency insensitive parameterized network-on-chip architecture for multiprocessor SoCs. In *Proceedings of the International Conference on Computer Design*, pages 536–539, 2003.

[37] W.J. Dally. Virtual-channel flow control. *IEEE Transactions on Parallel and Distributed Systems*, 3(2):194–205, 1992.

[38] W.J. Dally and C.L. Seitz. Deadlock-Free Message Routing in Multiprocessor Interconnection Networks. *IEEE Transanctions on Computers*, 36(5):547–553, 1987.

[39] W.J. Dally and B. Towles. Route packets, not wires: on-chip inteconnection networks. In *Proceedings of the 38th annual Design Automation Conference*, pages 684–689, 2001.

[40] W.J. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2004.

[41] B. Dang, P. Joseph, M. Bakir, T. Spencer, P. Kohl, and J. Meindl. Wafer-level microfluidic cooling interconnects for GSI. In *Proceedings of the IEEE 2005 International Interconnect Technology Conference*, pages 180–182, 2005.

[42] S. Das, A. Fan, K.-N. Chen, C.S. Tan, N. Checka, and R. Reif. Technology, performance, and computer-aided design of three-dimensional integrated circuits. In *Proceedings of the 2004 International Symposium on Physical Design*, pages 108–115, 2004.

[43] S. Dasgupta and A. Yakovlev. Comparative analysis of GALS clocking schemes. *IET Computers Digital Techniques*, 1(2):59–69, 2007.

[44] W.R. Davis, J. Wilson, S. Mick, J. Xu, H. Hua, C. Mineo, A.M. Sule, M. Steer, and P.D. Franzon. Demystifying 3D ICs: The Pros and Cons of Going Vertical. *IEEE Design Test of Computers*, 22(6):498–510, 2005.

[45] Y. Deng and W. Maly. 2.5D system integration: a design driven system implementation schema. In *Proceedings of the 2004 Asia and South Pacific Design Automation Conference*, pages 450–455, 2004.

[46] J. Duato, S. Yalamanchili, and N. Lionel. *Interconnection Networks: An Engineering Approach*. Morgan Kaufmann Publishers Inc., 2002.

[47] B.S. Feero and P.P. Pande. Networks-on-Chip in a Three-Dimensional Environment: A Performance Evaluation. *IEEE Transactions on Computers*, 58(1):32–45, 2009.

[48] F. Feliciian and S.B. Furber. An asynchronous on-chip network router with quality-of-service (QoS) support. In *Proceedings of the IEEE International SOC Conference*, pages 274–277, 2004.

[49] P. Garrou, C. Bower, and P. (Eds.). Ramm. *Handbook of 3D Integration*. Wiley-VCH, 2012.

[50] P. Ghosh, A. Sen, and A. Hall. Energy efficient application mapping to NoC processing elements operating at multiple voltage levels. In *Proceedings of the ACM/IEEE International Symposium on Networks-on-Chip*, pages 80–85, 2009.

[51] R. Ginosar. Fourteen Ways to Fool Your Synchronizer. In *Proceedings of the 9th International Symposium on Asynchronous Circuits and Systems*, pages 89–97, 2003.

[52] K. Goossens, J. Dielissen, and A. Radulescu. Æthereal network on chip: concepts, architectures, and implementations. *IEEE Design Test of Computers*, 22(5):414–421, 2005.

[53] B. Goplen and S. Sapatnekar. Efficient thermal placement of standard cells in 3D ICs using a force directed approach. In *Proceedings of the International Conference on Computer Aided Design*, pages 86–89, 2003.

[54] C. Grecu, P.P. Pande, A. Ivanov, and R. Saleh. A Scalable Communication-Centric SoC Interconnect Architecture. In *Proceedings of the 5th International Symposium on Quality Electronic Design*, pages 343–348, 2004.

[55] R.I. Greenberg and L. Guan. An Improved Analytical Model for Wormhole Routed Networks with Application to Butterfly Fat-Trees. In *Proceedings of the international Conference on Parallel Processing*, pages 44–48, 1997.

[56] L. Guang, E. Nigussie, P. Rantala, J. Isoaho, and H. Tenhunen. Hierarchical agent monitoring design approach towards self-aware parallel systems-on-chip. *ACM Transactions in Embedded Computing Systems*, 9(3):25:1–25:24, 2010.

[57] P. Guerrier and A. Greiner. A generic architecture for on-chip packet-switched interconnections. In *Proceedings of the conference on Design, automation and test in Europe*, pages 250–256, 2000.

[58] G. Guindani, C. Reinbrecht, T. Raupp, N. Calazans, and F.G. Moraes. NoC Power Estimation at the RTL Abstraction Level. In *Proceedings of the IEEE Computer Society Annual Symposium on VLSI*, pages 475–478, 2008.

[59] A. Hemani, A. Jantsch, S. Kumar, A. Postula, J. ?berg, M. Millberg, and D. Lindqvist. Network on chip: An architecture for billion transistor era. In *Proceedings of the IEEE NorChip Conference*, 2000.

[60] Y. Hoskote, S. Vangal, A. Singh, N. Borkar, and S. Borkar. A 5-GHz Mesh Interconnect for a Teraflops Processor. *IEEE Micro*, 27(5):51–61, 2007.

[61] M. Hosseinabady, M.R. Kakoee, J. Mathew, and D.K. Pradhan. Reliable network-on-chip based on generalized de Bruijn graph. In *Proceedings of the IEEE International High Level Design Validation and Test Workshop*, pages 3–10, 2007.

[62] A.-C. Hsieh, T. Hwang, M.-. Chang, M.-H. Tsai, C.-M. Tseng, and H.-C. Li. TSV redundancy: architecture and design issues in 3D IC. In *Proceedings of the Conference on Design, Automation and Test in Europe*, pages 166–171, 2010.

[63] J. Hu, Y. Shin, N. Dhanwada, and R. Marculescu. Architecting voltage islands in core-based system-on-a-chip designs. In *Proceedings of the 2004 International Symposium on Low Power Electronics and Design*, pages 180–185, 2004.

[64] H. Hua, C. Mineo, K. Schoienfliess, A. Sule, S. Melamed, and W.R. Davis. Performance Trend in Three-Dimensional Integrated Circuits. In *Proceedings of the International Interconnect Technology Conference*, pages 45–47, 2006.

[65] W.-L. Hung, G. M. Link, Y. Xie, N. Vijaykrishnan, and M. J. Irwin. Interconnect and Thermal-aware Floorplanning for 3D Microprocessors. In *Proceedings of the 7th International Symposium on Quality Electronic Design*, pages 98–104, 2006.

[66] M. Ieong, K.W. Guarini, V. Chan, K. Bernstein, R. Joshi, J. Kedzierski, and W. Haensch. Three dimensional CMOS devices and integrated circuits. In *Proceedings of the IEEE Custom Integrated Circuits Conference*, pages 207–213, 2003.

[67] Texas Instruments. *Flip Chip Ball Grid Array Package Reference Guide*. Literature Number: SPRU811A, May 2005.

[68] A. Jain, R.E. Jones, R. Chatterjee, and S. Pozder. Analytical and Numerical Modeling of the Thermal Performance of Three-Dimensional Integrated Circuits. *IEEE Transactions on Components and Packaging Technologies*, 33(1):56–63, 2010.

[69] W. Jang, D. Ding, and D.Z. Pan. A voltage-frequency island aware energy optimization framework for networks-on-chip. In *Proceedings of the International Conference on Computer-Aided Design*, pages 264–269, 2008.

[70] A. Jantsch and H. Tenhunen (Eds.). *Networks on Chip*. Kluwer Academic Publishers, 2003.

[71] J. Jiao, Y. Fu, T. Liu, H. Wang, X. Han, and J. Wang. Performance analysis and optimization for homogenous multi-core system based on 3D Torus Network on Chip. In *Proceedings of the 8th IEEE International NEWCAS Conference*, pages 313–316, 2010.

[72] J.W. Joyner, P. Zarkesh-Ha, and J.D. Meindl. A stochastic global net-length distribution for a three-dimensional system-on-a-chip (3D-SoC). In *Proceedings of the 14th Annual IEEE International ASIC/SOC Conference*, pages 147–151, 2001.

[73] S.-M. Jung, J. Jang, W. Cho, H. Cho, J. Jeong, Y. Chang, J. Kim, Y. Rah, Y. Son, J. Park, M.-S. Song, K.-H. Kim, J.-S. Lim, and K. Kim. Three Dimensionally Stacked NAND Flash Memory Technology Using Stacking Single Crystal Si Layers on ILD and TANOS Structure for Beyond 30nm Node. In *International Electron Devices Meeting*, pages 1–4, 2006.

[74] F. Karim, A. Nguyen, and S. Dey. An interconnect architecture for networking systems on chips. *IEEE Micro*, 22(5):36–45, 2002.

[75] T. Kgil, S. D'Souza, A. Saidi, N. Binkert, R. Dreslinski, T. Mudge, S. Reinhardt, and K. Flautner. PicoServer: using 3D stacking technology to enable a compact energy efficient chip multiprocessor. In *Proceedings of the 12th international conference on Architectural support for programming languages and operating systems*, pages 117–128, 2006.

[76] C. Kim, D. Burger, and S.W. Keckler. An adaptive, non-uniform cache structure for wire-delay dominated on-chip caches. *SIGOPS Operating Systems Review*, 36(5):211–222, 2002.

[77] D. Kim, K. Lee, S.J. Lee, and H.Y. Yoo. A reconfigurable crossbar switch with adaptive bandwidth control for networks-on-chip. In *Proceedings of the IEEE International Symposium on Circuits and Systems*, pages 2369–2372, 2005.

[78] J. Kim, C. Nicopoulos, D. Park, R. Das, Y. Xie, V. Narayanan, M. S. Yousif, and C. R. Das. A novel dimensionally-decomposed router for on-chip communication in 3D architectures. In *Proceedings of the ACM international symposium on Computer architecture*, pages 138–149, 2007.

[79] P. Koopman and T. Chakravarty. Cyclic Redundancy Code (CRC) Polynomial Selection For Embedded Networks. In *Proceedings of the International Conference on Dependable Systems and Networks*, pages 145–154, 2004.

[80] S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani. A network on chip architecture and design methodology. In *Proceedings of the IEEE Computer Society Annual Symposium on VLSI*, pages 105–112, 2002.

[81] S. Kumar, A. Jantsch, J.-P. Soininen, M. Forsell, M. Millberg, J. Oberg, K. Tiensyrja, and A. Hemani. A network on chip architecture and design methodology. In *Proceedings of the IEEE Computer Society Annual Symposium on VLSI*, pages 105–112, 2002.

[82] D.E. Lackey, P.S. Zuchowski, Thomas R. Bednar, D.W. Stout, S.W. Gould, and J.M. Cohn. Managing power and performance for System-on-Chip designs using Voltage Islands. In *Proceedings of the 2002 IEEE/ACM international conference on Computer-aided design*, pages 195–202, 2002.

[83] D.E. Lackey, P.S. Zuchowski, T.R. Bednar, D.W. Stout, S.W. Gould, and J.M. Cohn. Managing power and performance for System-on-Chip designs using Voltage Islands. In *Proceedings of the 2002 IEEE/ACM International Conference on Computer-Aided Design*, pages 195–202, 2002.

[84] Y.-C. Lan, S.-H. Lo, Y.-C. Lin, Y.-H. Hu, and S.-J. Chen. BiNoC: A bidirectional NoC architecture with dynamic self-reconfigurable channel. In *Proceedings of the 3rd ACM/IEEE International Symposium on Networks-on-Chip*, pages 266–275, 2009.

[85] K. Latif, A.-M Rahmani, K.R. Vaddina, T. Seceleanu, P. Liljeberg, and H. Tenhunen. Enhancing Performance Sustainability of Fault Tolerant Routing Algorithms in NoC-Based Architectures. In *Proceedings of the Euromicro Conference on Digital System Design*, pages 626 – 633, 2011.

[86] T. Lehtonen, D. Wolpert, P. Liljeberg, J. Plosila, and P. Ampadu. Self-Adaptive System for Addressing Permanent Errors in On-Chip Interconnects. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 18(4):527–540, 2010.

[87] C.E. Leiserson. Fat-trees: universal networks for hardware-efficient supercomputing. *IEEE Transactions on Computers*, 34(10):892–901, 1985.

[88] F. Li, C. Nicopoulos, T. Richardson, Yuan X., V. Narayanan, and M. Kandemir. Design and Management of 3D Chip Multiprocessors Using Network-in-Memory. In *Proceedings of the 33rd International Symposium on Computer Architecture*, pages 130–141, 2006.

[89] F. Li, X. Yang, A.T. Meeks, J.T. Shearer, and K.Y. Le. Evaluation of SiO2 antifuse in a 3D-OTP memory. *IEEE Transactions on Device and Materials Reliability*, 4(3):416–421, 2004.

[90] M. Li, Q.-A. Zeng, and W.-B. Jone. DyXY: a proximity congestion-aware deadlock-free dynamic routing method for network on chip. In *Proceedings of the 43rd annual Design Automation Conference*, pages 849–852, 2006.

[91] C. Liu, L. Zhang, Y. Han, and X. Li. Vertical interconnects squeezing in symmetric 3D mesh network-on-chip. In *Proceedings of the 16th Asia and South Pacific Design Automation Conference*, pages 357–362, 2011.

[92] C.C. Liu, I. Ganusov, M. Burtscher, and Sandip T. Bridging the processor-memory performance gap with 3D IC technology. *IEEE Design Test of Computers*, 22(6):556–564, 2005.

[93] G.H. Loh, Y. Xie, and B. Black. Processor Design in 3D Die-Stacking Technologies. *IEEE Micro*, 27(3):31–48, 2007.

[94] P. Lotfi-Kamran, A.-M. Rahmani, M. Daneshtalab, A. Afzali-Kusha, and Z. Navabi. EDXY - A low cost congestion-aware routing algorithm for network-on-chips. *Journal of Systems Architecture*, 56(7):256–264, 2010.

[95] N. Madan and R. Balasubramonian. Leveraging 3D Technology for Improved Reliability. In *Proceedings of the 40th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 223–235, 2007.

[96] H. Matsutani, M. Koibuchi, and H. Amano. Tightly-Coupled Multi-Layer Topologies for 3-D NoCs. In *Proceedings of the International Conference on Parallel Processing*, pages 75–84, 2007.

[97] T.G. Mattson, R.F. Van der Wijngaart, M. Riepen, T. Lehnig, P. Brett, W. Haas, P. Kennedy, J. Howard, S. Vangal, N. Borkar, G. Ruhl, and S. Dighe. The 48-core SCC Processor: the Programmer's View. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–11, 2010.

[98] P. Morrow, M. Kobrinsky, S. Ramanathan, C.-M. Park, M. Harmes, V. Ramachandrarao, H. Park, G. Kloster, S. List, and S. Kim. Wafer-Level 3D Interconnects Via Cu Bonding. In *Proceedings of the 21st Advanced Metallization Conference*, pages 125–130, 2004.

[99] F. Mu and C. Svensson. Self-tested self-synchronization circuit for mesochronous clocking. *IEEE Transactions on Circuits and Systems II: Analog and Digital Signal Processing*, 48(2):129–140, 2001.

[100] R. Mullins, A. West, and S. Moore. Low-Latency Virtual-Channel Routers for On-Chip Networks. In *Proceedings of the 31st Annual International Symposium on Computer Architecture*, pages 188–197, 2004.

[101] R. Mullins, A. West, and S. Moore. Low-latency virtual-channel routers for on-chip networks. In *Proceedings of the International Symposium on Computer Architecture*, pages 188–197, 2004.

[102] S. Murali, C. Seiculescu, L. Benini, and G. De Micheli. Synthesis of networks on chips for 3D systems on chips. In *Proceedings of the 2009 Asia and South Pacific Design Automation Conference*, pages 242–247, 2009.

[103] S. Mutoh, T. Douseki, Y. Matsuya, T. Aoki, S. Shigematsu, and J. Yamada. 1-V power supply high-speed digital circuit technology with multithreshold-voltage CMOS. *IEEE Journal of Solid-State Circuits*, 30(8):847–854, 1995.

[104] J. Muttersbach, T. Villiger, and W. Fichtner. Practical Design of Globally-Asynchronous Locally-Synchronous Systems. In *Proceedings of the 6th International Symposium on Advanced Research in Asynchronous Circuits and Systems*, pages 52–59, 2000.

[105] L.M. Ni and P.K. McKinley. A survey of wormhole routing techniques in direct networks. *IEEE Computer*, 26(2):62–76, 1993.

[106] C.A. Nicopoulos, D. Park, J. Kim, N. Vijaykrishnan, M.S. Yousif, and C.R. Das. ViChaR: A Dynamic Virtual Channel Regulator for Network-on-Chip Routers. In *Proceedings of the 39th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 333–346, 2006.

[107] L.S. Nielsen and C. Niessen. Low-power operation using self-timed circuits and adaptive scaling of the supply voltage. *IEEE Transaction on Very Large Scale Integration Systems*, 2(4):391–397, 1994.

[108] K. Niyogi and D. Marculescu. Speed and voltage selection for GALS systems based on voltage/frequency islands. In *Proceedings of the 2005 Asia and South Pacific Design Automation Conference*, pages 292–297, 2005.

[109] K. Nose and T. Sakurai. Analysis and future trend of short-circuit power. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 19(9):1023–1030, 2000.

[110] P.R. O'Brien and T.L. Savarino. Modeling the driving-point characteristic of resistive interconnect for accurate delay estimation. In *Proceedings of the IEEE International Conference on Computer-Aided Design, Digest of Technical Papers.*, pages 512–515, 1989.

[111] U.Y. Ogras, R. Marculescu, P. Choudhary, and D. Marculescu. Voltage-Frequency Island Partitioning for GALS-based Networks-on-Chip. In *Proceedings of the IEEE Design Automation Conference*, pages 110–115, 2007.

[112] U.Y. Ogras, R. Marculescu, D. Marculescu, and E.G. Jung. Design and management of voltage-frequency island partitioned networks-on-chip. *IEEE Transactions on Very Large Scale Integration Systems*, 17(3):330–341, 2009.

[113] T. Ono and M. Greenstreet. A modular synchronizing FIFO for NoCs. In *Proceedings of the 3rd ACM/IEEE International Symposium on Networks-on-Chip*, pages 224–233, 2009.

[114] L. Ost, A. Mello, J. Palma, F. Moraes, and N. Calazans. MAIA - a framework for networks on chip generation and verification. In *Proceedings of the Asia and South Pacific Design Automation Conference*, volume 1, 2005.

[115] M. Palesi, S. Kumar, and V. Catania. Leveraging partially faulty links usage for enhancing yield and performance in networks-on-chip. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 29(3):426–440, 2010.

[116] I.M. Panades and A. Greiner. Bi-Synchronous FIFO for Synchronous Circuit Communication Well Suited for Network-on-Chip in GALS Architectures. In *Proceedings of the First International Symposium on Networks-on-Chip*, pages 83–94, 2007.

[117] P.R. Panda, B.V.N. Silpa, A. Shrivastava, and K. Gummidipudi. *Power-efficient System Design*. Springer, 2010.

[118] P.P. Pande, C. Grecu, A. Ivanov, and R. Saleh. Design of a switch for network on chip applications. In *Proceedings of the International Symposium on Circuits and Systems*, volume 5, pages V–217–V–220 vol.5, 2003.

[119] P.P. Pande, C. Grecu, M. Jones, A. Ivanov, and R. Saleh. Performance Evaluation and Design Trade-Offs for Network-on-Chip Interconnect Architectures. *IEEE Transactions on Computers*, 54(8):1025–1040, 2005.

[120] D. Park, S. Eachempati, R. Das, A.K. Mishra, Y. Xie, N. Vijaykrishnan, and C.R. Das. MIRA: A Multi-layered On-Chip Interconnect Router Architecture. In *Proceedings of the 35th International Symposium on Computer Architecture*, pages 251–261, 2008.

[121] S. Pasricha. Exploring serial vertical interconnects for 3D ICs. In *Proceedings of the 46th ACM/IEEE Design Automation Conference*, pages 581–586, 2009.

[122] S. Pasricha and N. Dutt. *On-Chip Communication Architectures: System on Chip Interconnect*. Morgan Kaufmann, 2008.

[123] R.S. Patti. Three-Dimensional Integrated Circuits and the Future of System-on-Chip Designs. *Proceedings of the IEEE*, pages 1214–1224, 2006.

[124] V.F. Pavlidis and E.G. Friedman. 3-D topologies for networks-on-chip. *IEEE Transactions on Very Large Scale Integration Systems*, 15(10):1081–1090, 2007.

[125] V.F. Pavlidis and E.G. Friedman. *Three-dimensional Integrated Circuit Design*. Morgan Kaufmann, 2008.

[126] L.S. Peh and W.J. Dally. A delay model for router microarchitectures. *IEEE Micro*, 21(1):26–34, 2001.

[127] K. Puttaswamy and G.H. Loh. Thermal Herding: Microarchitecture Techniques for Controlling Hotspots in High-Performance 3D-Integrated Processors. In *Proceedings of the 13th IEEE International Symposium on High Performance Computer Architecture*, pages 193–204, 2007.

[128] Y. Qian, Z. Lu, and W. Dou. From 2D to 3D NoCs: A case study on worst-case communication performance. In *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design - Digest of Technical Papers*, pages 555–562, 2009.

[129] J. Quartana, S. Renane, A. Baixas, L. Fesquet, and M. Renaudin. GALS systems prototyping using multiclock FPGAs and asynchronous network-on-chips. In *Proceedings of the International Conference on Field Programmable Logic and Applications*, pages 299–304, 2005.

[130] A.-M. Rahmani, A. Afzali-Kusha, and M. Pedram. NED: A Novel Synthetic Traffic Pattern for Power/Performance Analysis of Network-on-Chips Using Negative Exponential Distribution. *Journal of Low Power Electronics*, 5(3):396–405, 2009.

[131] A.-M. Rahmani, M. Daneshtalab, A. Afzali-Kusha, and M. Pedram. Forecasting-Based Dynamic Virtual Channel Management for Power Reduction in Network-on-Chips. *Journal of Low Power Electronics*, 5(3):385–395, 2009.

[132] A.-M. Rahmani, I. Kamali, P. Lotfi-Kamran, A. Afzali-Kusha, and S. Safari. Negative Exponential Distribution Traffic Pattern for Power/Performance Analysis of Network on Chips. In *Proceedings of the International Conference on VLSI Design*, pages 157–162, 2009.

[133] A.-M. Rahmani, K. Latif, V. K. Rao, P. Liljeberg, J. Plosila, and H. Tenhunen. Congestion Aware, Fault Tolerant, and Thermally Efficient Inter-Layer Communication Scheme for Hybrid NoC-Bus 3D Architectures. In *Proceedings of the 5th IEEE/ACM International Symposium on Networks-on-Chip*, pages 65–72, 2011.

[134] A.-M. Rahmani, P. Liljeberg, J. Plosila, and H. Tenhunen. BBVC-3D-NoC: An Efficient 3D NoC Architecture Using Bidirectional Bisynchronous Vertical Channels. In *Proceedings of the IEEE Computer Society Annual Symposium on VLSI*, pages 452–453, 2010.

[135] A.-M. Rahmani, P. Liljeberg, J. Plosila, and H. Tenhunen. Power and performance optimization of voltage/frequency island-based networks-on-chip using reconfigurable synchronous/bi-synchronous FIFOs. In *Proceedings of the 7th ACM international conference on Computing frontiers*, pages 267–276, 2010.

[136] R. S. Ramanujam and B. Lin. A Layer-Multiplexed 3D On-Chip Network Architecture. *Embedded Systems Letters, IEEE*, 1(2):50–55, 2009.

[137] R.S. Ramanujam and B. Lin. Near-optimal oblivious routing on three-dimensional mesh networks. In *Proceedings of the IEEE International Conference on Computer Design*, pages 134–141, 2008.

[138] R.S. Ramanujam, V. Soteriou, B. Lin, and L.-S. Peh. Design of a High-Throughput Distributed Shared-Buffer NoC Router. In *Proceedings of the Fourth ACM/IEEE International Symposium on Networks-on-Chip*, pages 69–78, 2010.

[139] C. Rusu, L. Anghel, and D. Avresky. RILM: Reconfigurable inter-layer routing mechanism for 3D multi-layer networks-on-chip. In *Proceedings of the International On-Line Testing Symposium*, pages 121–126, 2010.

[140] T. Sakurai. Closed-form expressions for interconnection delay, coupling, and crosstalk in VLSIs. *IEEE Transactions on Electron Devices*, 40(1):118–124, 1993.

[141] C. Seiculescu, S. Murali, L. Benini, and G. De Micheli. SunFloor 3D: A tool for Networks On Chip topology synthesis for 3D systems on chips. In *Proceedings of the Design, Automation Test in Europe Conference and Exhibition*, pages 9–14, 2009.

[142] L. Shang, L.-S. Peh, and N.K. Jha. Dynamic voltage scaling with links for power optimization of interconnection networks. In *Proceedings of the 9th International Symposium on High-Performance Computer Architecture*, pages 91–102, 2003.

[143] L. Shang, L.-S. Peh, A. Kumar, and N.K. Jha. Thermal Modeling, Characterization and Management of On-Chip Networks. In *Proceedings of the 37th annual IEEE/ACM International Symposium on Microarchitecture*, pages 67–78, 2004.

[144] A. Sheibanyrad, I. Miro Panades, and A. Greiner. Systematic comparison between the asynchronous and the multi-synchronous implementations of a network on chip architecture. In *Proceedings of the conference on Design, automation and test in Europe*, pages 1090–1095, 2007.

[145] D. Sigüenza-Tortosa, T. Ahonen, and J. Nurmi. Issues in the development of a practical NoC: the Proteo concept. *Integration, the VLSI Journal*, 38(1):95–105, 2004.

[146] K. Skadron, M.R. Stan, W. Huang, S. Velusamy, K. Sankaranarayanan, and D. Tarjan. Temperature-aware microarchitecture. In *Proceedings of the 30th ACM International Symposium on Computer Architecture*, pages 2–13, 2003.

[147] S. Spiesshoefer, L. Schaper, S. Burkett, G. Vangara, Z. Rahman, and P. Arunasalam. Z-axis interconnects using fine pitch, nanoscale through-silicon vias: Process development. In *Proceedings of the 54th Electronic Components and Technology Conference*, volume 1, pages 466–471, 2004.

[148] R.R. Tamhankar, S. Murali, and G. De Micheli. Performance driven reliable link design for networks on chips. In *Proceedings of the Asia and South Pacific Design Automation Conference*, pages 749–754, 2005.

[149] M.B. Taylor, J. Kim, J. Miller, D. Wentzlaff, F. Ghodrat, B. Greenwald, H. Hoffman, P. Johnson, Jae-Wook Lee, W. Lee, A. Ma,

A. Saraf, M. Seneski, N. Shnidman, V. Strumpen, M. Frank, S. Amarasinghe, and A. Agarwal. The raw microprocessor: a computational fabric for software circuits and general-purpose programs. *IEEE Micro*, 22(2):25–35, 2002.

[150] R.J. Tersine. *Principles of Inventory and Material Management*. Prentice Hall PTR, 1994.

[151] Y. Thonnart, E. Beigné, and P. Vivet. Design and Implementation of a GALS Adapter for ANoC Based Architectures. In *Proceedings of the 15th IEEE Symposium on Asynchronous Circuits and Systems*, pages 13–22, 2009.

[152] Tilera. TILE-Gx processors family @ONLINE, http://www.tilera.com/products/processors/TILE-Gx_Family, 2012.

[153] Erik B. Van Der Tol and Egbert G. T. Jaspers. Mapping of MPEG-4 decoding on a flexible architecture platform. In *Media Processors 2002*, pages 1–13, 2002.

[154] A.W. Topol, D.C. La Tulipe, Jr., 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*, 50(4/5):491–506, 2006.

[155] Y.-F. Tsai, Y. Xie, N. Vijaykrishnan, and M. J. Irwin. Three-Dimensional Cache Design Exploration Using 3DCacti. In *Proceedings of the 2005 International Conference on Computer Design*, pages 519–524, 2005.

[156] K.R. Vaddina, A.-M. Rahmani, K. Latif, P. Liljeberg, and J. Plosila. Thermal Analysis of Job Allocation and Scheduling Schemes for 3D Stacked NoC's. In *Proceedings of the Euromicro Conference on Digital System Design*, pages 643–648, 2011.

[157] K.R. Vaddina, A.-M. Rahmani, K. Latif, P. Liljeberg, and J. Plosila. Thermal modeling and analysis of advanced 3D stacked structures. In *Proceedings of the Elsevier International Conference on Communication Technology and System Design*, page 248257, 2011.

[158] A.S. Vaidya, A. Sivasubramaniam, and C.R. Das. Impact of virtual channels and adaptive routing on application performance. In *Proceedings of the ACM SIGCPR conference on Computer personnel research*, pages 223–237, 2001.

[159] S. Vangal, A. Singh, J. Howard, S. Dighe, N. Borkar, and A. Alvandpour. A 5.1GHz 0.34mm^2 Router for Network-on-Chip Applications. In *IEEE Symposium on VLSI Circuits*, pages 42–43, 2007.

[160] S.R. Vangal, J. Howard, G. Ruhl, S. Dighe, H. Wilson, J. Tschanz, D. Finan, A. Singh, T. Jacob, S. Jain, V. Erraguntla, C. Roberts, Y. Hoskote, N. Borkar, and S. Borkar. An 80-Tile Sub-100-W TeraFLOPS Processor in 65-nm CMOS. *IEEE Journal of Solid-State Circuits*, 43(1):29–41, 2008.

[161] D. Velenis, M. Stucchi, E.J. Marinissen, B. Swinnen, and E. Beyne. Impact of 3D design choices on manufacturing cost. In *Proceedings of the IEEE International Conference on 3D System Integration*, pages 1–5, 2009.

[162] A. Vitkovskiy, V. Soteriou, and C. Nicopoulos. A fine-grained link-level fault-tolerant mechanism for networks-on-chip. In *Proceedings of the IEEE International Conference on Computer Design*, pages 447–454, 2010.

[163] C. Wang, W.-H. Hu, S.E. Lee, and N. Bagherzadeh. Area and power-efficient innovative congestion-aware Network-on-Chip architecture. *Journal of Systems Architecture*, 57(1):24–38, 2011.

[164] H. Wang, L.S. Peh, and S. Malik. Power-driven design of router microarchitectures in on-chip networks. In *Proceedings of the 36th Annual IEEE/ACM International Symposium on Microarchitecture*, pages 105–116, 2003.

[165] P.T. Wolkotte, G.J.M. Smit, G.K. Rauwerda, and L.T. Smit. An Energy-Efficient Reconfigurable Circuit-Switched Network-on-Chip. In *Proceedings of the IEEE International Parallel and Distributed Processing Symposium*, page 155a, 2005.

[166] E. Wong and S.K. Lim. 3D Floorplanning with Thermal Vias. In *Proceedings of the Design, Automation and Test in Europe*, volume 1, pages 1–6, 2006.

[167] T.C. Xu, P. Liljeberg, and H. Tenhunen. Exploring DRAM Last Level Cache for 3D Network-on-Chip Architecture. In *Proceedings of the IEEE International Conference on Embedded System and Microprocessors*, pages 39–44, 2010.

[168] T.C. Xu, A.W. Yin, P. Liljeberg, and H. Tenhunen. A study of 3D Network-on-Chip design for data parallel H.264 coding. In *Proceedings of the IEEE International NORCHIP Conference*, pages 1–6, 2009.

[169] Y. Xu, Y. Du, B. Zhao, X. Zhou, Y. Zhang, and J. Yang. A low-radix and low-diameter 3D interconnection network design. In *Proceedings of the IEEE 15th International Symposium on High Performance Computer Architecture*, pages 30–42, 2009.

[170] T.T. Ye, L. Benini, and G. De Micheli. Analysis of power consumption on switch fabrics in network routers. In *Proceedings of the International Design Automation Conference*, pages 524–529, 2002.

[171] A.W. Yin, T.C. Xu, P. Liljeberg, and H. Tenhunen. Explorations of Honeycomb Topologies for Network-on-Chip. In *Proceedings of the International Conference on Network and Parallel Computing*, pages 73–79, 2009.

[172] C.A. Zeferino, M.E. Kreutz, and A.A. Susin. RASoC: A Router Soft-Core for Networks-on-Chip. In *Proceedings of the conference on Design, automation and test in Europe - Volume 3*, pages 198–203, 2004.

Turku Centre for Computer Science
TUCS Dissertations

1. **Marjo Lipponen**, On Primitive Solutions of the Post Correspondence Problem
2. **Timo Käkölä**, Dual Information Systems in Hyperknowledge Organizations
3. **Ville Leppänen**, Studies on the Realization of PRAM
4. **Cunsheng Ding**, Cryptographic Counter Generators
5. **Sami Viitanen**, Some New Global Optimization Algorithms
6. **Tapio Salakoski**, Representative Classification of Protein Structures
7. **Thomas Långbacka**, An Interactive Environment Supporting the Development of Formally Correct Programs
8. **Thomas Finne**, A Decision Support System for Improving Information Security
9. **Valeria Mihalache**, Cooperation, Communication, Control. Investigations on Grammar Systems.
10. **Marina Waldén**, Formal Reasoning About Distributed Algorithms
11. **Tero Laihonen**, Estimates on the Covering Radius When the Dual Distance is Known
12. **Lucian Ilie**, Decision Problems on Orders of Words
13. **Jukkapekka Hekanaho**, An Evolutionary Approach to Concept Learning
14. **Jouni Järvinen**, Knowledge Representation and Rough Sets
15. **Tomi Pasanen**, In-Place Algorithms for Sorting Problems
16. **Mika Johnsson**, Operational and Tactical Level Optimization in Printed Circuit Board Assembly
17. **Mats Aspnäs**, Multiprocessor Architecture and Programming: The Hathi-2 System
18. **Anna Mikhajlova**, Ensuring Correctness of Object and Component Systems
19. **Vesa Torvinen**, Construction and Evaluation of the Labour Game Method
20. **Jorma Boberg**, Cluster Analysis. A Mathematical Approach with Applications to Protein Structures
21. **Leonid Mikhajlov**, Software Reuse Mechanisms and Techniques: Safety Versus Flexibility
22. **Timo Kaukoranta**, Iterative and Hierarchical Methods for Codebook Generation in Vector Quantization
23. **Gábor Magyar**, On Solution Approaches for Some Industrially Motivated Combinatorial Optimization Problems
24. **Linas Laibinis**, Mechanised Formal Reasoning About Modular Programs
25. **Shuhua Liu**, Improving Executive Support in Strategic Scanning with Software Agent Systems
26. **Jaakko Järvi**, New Techniques in Generic Programming – C++ is more Intentional than Intended
27. **Jan-Christian Lehtinen**, Reproducing Kernel Splines in the Analysis of Medical Data
28. **Martin Büchi**, Safe Language Mechanisms for Modularization and Concurrency
29. **Elena Troubitsyna**, Stepwise Development of Dependable Systems
30. **Janne Näppi**, Computer-Assisted Diagnosis of Breast Calcifications
31. **Jianming Liang**, Dynamic Chest Images Analysis
32. **Tiberiu Seceleanu**, Systematic Design of Synchronous Digital Circuits
33. **Tero Aittokallio**, Characterization and Modelling of the Cardiorespiratory System in Sleep-Disordered Breathing
34. **Ivan Porres**, Modeling and Analyzing Software Behavior in UML
35. **Mauno Rönkkö**, Stepwise Development of Hybrid Systems
36. **Jouni Smed**, Production Planning in Printed Circuit Board Assembly
37. **Vesa Halava**, The Post Correspondence Problem for Market Morphisms
38. **Ion Petre**, Commutation Problems on Sets of Words and Formal Power Series
39. **Vladimir Kvassov**, Information Technology and the Productivity of Managerial Work
40. **Frank Tétard**, Managers, Fragmentation of Working Time, and Information Systems

41. **Jan Manuch**, Defect Theorems and Infinite Words
42. **Kalle Ranto**, Z_4-Goethals Codes, Decoding and Designs
43. **Arto Lepistö**, On Relations Between Local and Global Periodicity
44. **Mika Hirvensalo**, Studies on Boolean Functions Related to Quantum Computing
45. **Pentti Virtanen**, Measuring and Improving Component-Based Software Development
46. **Adekunle Okunoye**, Knowledge Management and Global Diversity – A Framework to Support Organisations in Developing Countries
47. **Antonina Kloptchenko**, Text Mining Based on the Prototype Matching Method
48. **Juha Kivijärvi**, Optimization Methods for Clustering
49. **Rimvydas Rukšėnas**, Formal Development of Concurrent Components
50. **Dirk Nowotka**, Periodicity and Unbordered Factors of Words
51. **Attila Gyenesei**, Discovering Frequent Fuzzy Patterns in Relations of Quantitative Attributes
52. **Petteri Kaitovaara**, Packaging of IT Services – Conceptual and Empirical Studies
53. **Petri Rosendahl**, Niho Type Cross-Correlation Functions and Related Equations
54. **Péter Majlender**, A Normative Approach to Possibility Theory and Soft Decision Support
55. **Seppo Virtanen**, A Framework for Rapid Design and Evaluation of Protocol Processors
56. **Tomas Eklund**, The Self-Organizing Map in Financial Benchmarking
57. **Mikael Collan**, Giga-Investments: Modelling the Valuation of Very Large Industrial Real Investments
58. **Dag Björklund**, A Kernel Language for Unified Code Synthesis
59. **Shengnan Han**, Understanding User Adoption of Mobile Technology: Focusing on Physicians in Finland
60. **Irina Georgescu**, Rational Choice and Revealed Preference: A Fuzzy Approach
61. **Ping Yan**, Limit Cycles for Generalized Liénard-Type and Lotka-Volterra Systems
62. **Joonas Lehtinen**, Coding of Wavelet-Transformed Images
63. **Tommi Meskanen,** On the NTRU Cryptosystem
64. **Saeed Salehi**, Varieties of Tree Languages
65. **Jukka Arvo**, Efficient Algorithms for Hardware-Accelerated Shadow Computation
66. **Mika Hirvikorpi**, On the Tactical Level Production Planning in Flexible Manufacturing Systems
67. **Adrian Costea**, Computational Intelligence Methods for Quantitative Data Mining
68. **Cristina Seceleanu**, A Methodology for Constructing Correct Reactive Systems
69. **Luigia Petre**, Modeling with Action Systems
70. **Lu Yan**, Systematic Design of Ubiquitous Systems
71. **Mehran Gomari**, On the Generalization Ability of Bayesian Neural Networks
72. **Ville Harkke**, Knowledge Freedom for Medical Professionals – An Evaluation Study of a Mobile Information System for Physicians in Finland
73. **Marius Cosmin Codrea**, Pattern Analysis of Chlorophyll Fluorescence Signals
74. **Aiying Rong**, Cogeneration Planning Under the Deregulated Power Market and Emissions Trading Scheme
75. **Chihab BenMoussa**, Supporting the Sales Force through Mobile Information and Communication Technologies: Focusing on the Pharmaceutical Sales Force
76. **Jussi Salmi**, Improving Data Analysis in Proteomics
77. **Orieta Celiku**, Mechanized Reasoning for Dually-Nondeterministic and Probabilistic Programs
78. **Kaj-Mikael Björk**, Supply Chain Efficiency with Some Forest Industry Improvements
79. **Viorel Preoteasa**, Program Variables – The Core of Mechanical Reasoning about Imperative Programs
80. **Jonne Poikonen**, Absolute Value Extraction and Order Statistic Filtering for a Mixed-Mode Array Image Processor
81. **Luka Milovanov**, Agile Software Development in an Academic Environment
82. **Francisco Augusto Alcaraz Garcia**, Real Options, Default Risk and Soft Applications
83. **Kai K. Kimppa**, Problems with the Justification of Intellectual Property Rights in Relation to Software and Other Digitally Distributable Media
84. **Dragoş Truşcan**, Model Driven Development of Programmable Architectures
85. **Eugen Czeizler**, The Inverse Neighborhood Problem and Applications of Welch Sets in Automata Theory

86. **Sanna Ranto**, Identifying and Locating-Dominating Codes in Binary Hamming Spaces
87. **Tuomas Hakkarainen**, On the Computation of the Class Numbers of Real Abelian Fields
88. **Elena Czeizler**, Intricacies of Word Equations
89. **Marcus Alanen**, A Metamodeling Framework for Software Engineering
90. **Filip Ginter**, Towards Information Extraction in the Biomedical Domain: Methods and Resources
91. **Jarkko Paavola**, Signature Ensembles and Receiver Structures for Oversaturated Synchronous DS-CDMA Systems
92. **Arho Virkki**, The Human Respiratory System: Modelling, Analysis and Control
93. **Olli Luoma**, Efficient Methods for Storing and Querying XML Data with Relational Databases
94. **Dubravka Ilić**, Formal Reasoning about Dependability in Model-Driven Development
95. **Kim Solin**, Abstract Algebra of Program Refinement
96. **Tomi Westerlund**, Time Aware Modelling and Analysis of Systems-on-Chip
97. **Kalle Saari**, On the Frequency and Periodicity of Infinite Words
98. **Tomi Kärki**, Similarity Relations on Words: Relational Codes and Periods
99. **Markus M. Mäkelä**, Essays on Software Product Development: A Strategic Management Viewpoint
100. **Roope Vehkalahti**, Class Field Theoretic Methods in the Design of Lattice Signal Constellations
101. **Anne-Maria Ernvall-Hytönen**, On Short Exponential Sums Involving Fourier Coefficients of Holomorphic Cusp Forms
102. **Chang Li**, Parallelism and Complexity in Gene Assembly
103. **Tapio Pahikkala**, New Kernel Functions and Learning Methods for Text and Data Mining
104. **Denis Shestakov**, Search Interfaces on the Web: Querying and Characterizing
105. **Sampo Pyysalo**, A Dependency Parsing Approach to Biomedical Text Mining
106. **Anna Sell**, Mobile Digital Calendars in Knowledge Work
107. **Dorina Marghescu**, Evaluating Multidimensional Visualization Techniques in Data Mining Tasks
108. **Tero Säntti**, A Co-Processor Approach for Efficient Java Execution in Embedded Systems
109. **Kari Salonen**, Setup Optimization in High-Mix Surface Mount PCB Assembly
110. **Pontus Boström**, Formal Design and Verification of Systems Using Domain-Specific Languages
111. **Camilla J. Hollanti**, Order-Theoretic Mehtods for Space-Time Coding: Symmetric and Asymmetric Designs
112. **Heidi Himmanen**, On Transmission System Design for Wireless Broadcasting
113. **Sébastien Lafond**, Simulation of Embedded Systems for Energy Consumption Estimation
114. **Evgeni Tsivtsivadze**, Learning Preferences with Kernel-Based Methods
115. **Petri Salmela**, On Commutation and Conjugacy of Rational Languages and the Fixed Point Method
116. **Siamak Taati**, Conservation Laws in Cellular Automata
117. **Vladimir Rogojin**, Gene Assembly in Stichotrichous Ciliates: Elementary Operations, Parallelism and Computation
118. **Alexey Dudkov**, Chip and Signature Interleaving in DS CDMA Systems
119. **Janne Savela**, Role of Selected Spectral Attributes in the Perception of Synthetic Vowels
120. **Kristian Nybom**, Low-Density Parity-Check Codes for Wireless Datacast Networks
121. **Johanna Tuominen**, Formal Power Analysis of Systems-on-Chip
122. **Teijo Lehtonen**, On Fault Tolerance Methods for Networks-on-Chip
123. **Eeva Suvitie**, On Inner Products Involving Holomorphic Cusp Forms and Maass Forms
124. **Linda Mannila**, Teaching Mathematics and Programming – New Approaches with Empirical Evaluation
125. **Hanna Suominen**, Machine Learning and Clinical Text: Supporting Health Information Flow
126. **Tuomo Saarni**, Segmental Durations of Speech
127. **Johannes Eriksson**, Tool-Supported Invariant-Based Programming

128. **Tero Jokela**, Design and Analysis of Forward Error Control Coding and Signaling for Guaranteeing QoS in Wireless Broadcast Systems
129. **Ville Lukkarila**, On Undecidable Dynamical Properties of Reversible One-Dimensional Cellular Automata
130. **Qaisar Ahmad Malik**, Combining Model-Based Testing and Stepwise Formal Development
131. **Mikko-Jussi Laakso**, Promoting Programming Learning: Engagement, Automatic Assessment with Immediate Feedback in Visualizations
132. **Riikka Vuokko**, A Practice Perspective on Organizational Implementation of Information Technology
133. **Jeanette Heidenberg**, Towards Increased Productivity and Quality in Software Development Using Agile, Lean and Collaborative Approaches
134. **Yong Liu**, Solving the Puzzle of Mobile Learning Adoption
135. **Stina Ojala**, Towards an Integrative Information Society: Studies on Individuality in Speech and Sign
136. **Matteo Brunelli**, Some Advances in Mathematical Models for Preference Relations
137. **Ville Junnila**, On Identifying and Locating-Dominating Codes
138. **Andrzej Mizera**, Methods for Construction and Analysis of Computational Models in Systems Biology. Applications to the Modelling of the Heat Shock Response and the Self-Assembly of Intermediate Filaments.
139. **Csaba Ráduly-Baka**, Algorithmic Solutions for Combinatorial Problems in Resource Management of Manufacturing Environments
140. **Jari Kyngäs**, Solving Challenging Real-World Scheduling Problems
141. **Arho Suominen**, Notes on Emerging Technologies
142. **József Mezei**, A Quantitative View on Fuzzy Numbers
143. **Marta Olszewska**, On the Impact of Rigorous Approaches on the Quality of Development
144. **Antti Airola**, Kernel-Based Ranking: Methods for Learning and Performace Estimation
145. **Aleksi Saarela**, Word Equations and Related Topics: Independence, Decidability and Characterizations
146. **Lasse Bergroth**, Kahden merkkijonon pisimmän yhteisen alijonon ongelma ja sen ratkaiseminen
147. **Thomas Canhao Xu**, Hardware/Software Co-Design for Multicore Architectures
148. **Tuomas Mäkilä**, Software Development Process Modeling – Developers Perspective to Contemporary Modeling Techniques
149. **Shahrokh Nikou**, Opening the Black-Box of IT Artifacts: Looking into Mobile Service Characteristics and Individual Perception
150. **Alessandro Buoni**, Fraud Detection in the Banking Sector: A Multi-Agent Approach
151. **Mats Neovius**, Trustworthy Context Dependency in Ubiquitous Systems
152. **Fredrik Degerlund**, Scheduling of Guarded Command Based Models
153. **Amir-Mohammad Rahmani-Sane**, Exploration and Design of Power-Efficient Networked Many-Core Systems

Turku Centre *for* Computer Science

Joukahaisenkatu 3-5 B, 20520 Turku, Finland | www. tucs.fi

University of Turku
Faculty of Mathematics and Natural Sciences
- Department of Information Technology
- Department of Mathematics and Statistics

Turku School of Economics
- Institute of Information Systems Science

Åbo Akademi University
Division for Natural Sciences and Technology
- Department of Information Technologies

ISBN 978-952-12-2817-9
ISSN 1239-1883

Amir-Mohammad Rahmani-Sane

Exploration and Design of Power-Efficient Networked Many-Core Systems