A 3D NETWORK-ON-CHIP FOR STACKED-DIE TRANSACTIONAL CHIP MULTIPROCESSORS USING THROUGH SILICON VIAS

Sumeet S. Kumar, Rene van Leuken
Delft University of Technology
EEMCS, Circuits and Systems group,
Mekelweg 4, 2628CD Delft, The Netherlands
{s.s.kumar, t.g.r.m.vanleuken}@tudelft.nl

***Abstract*—Effective utilization of computing power offered by modern chip multiprocessors (CMP) depends on the design and performance of the interconnect that connects them. We present a three-dimensional Network-on-Chip (NoC) based on the R3 router architecture for transactional CMPs utilizing advanced Through Silicon Vias (TSV) in a stacked-die architecture, facilitating low latency and high throughput communication between CMP nodes. We report the performance of an R3 based three-dimensional mesh in a stacked-die transactional CMP highlighting the limitations of performance scale-up with stacking. Furthermore, we present data on area penalty associated with the use of TSVs in different configurations in 90nm UMC technology.**

***Index Terms*—Multiprocessor interconnection networks, Multicore processing, Through-Silicon Vias, Three-dimensional integrated circuits**

I. INTRODUCTION

Increasing integration densities have allowed for an increase in the processing power available on chip over the last decade. Effectively leveraging this additional processing power is a critical design issue in high performance systems. *Chip Multiprocessors (CMP)* achieve this by exploiting *Thread Level Parallelism (TLP)* in programs, executing threads on different processor cores on chip. However, the gain in performance obtained from using such an array of processor cores is limited by cache misses, coherence updates and maintaining memory consistency, all of which depend on interconnect performance. Consequently, the interconnect must offer low latency communication between processor cores, as well as the shared L2 cache.

A large CMP may occupy a significant die area while requiring an extensive interconnect for communication between cores and the L2 cache. The size of the interconnect influences communication latencies, consequently limiting the performance gains obtained from adding additional cores to the system. Die stacking utilizing *Through Silicon Vias (TSV)* allows for a higher level of integration with low cost in terms of chip area. In a stacked-die architecture TSVs act as vertical links in the three-dimensional interconnect, linking routers on adjacent layers in the stack. TSVs facilitate shorter interconnect line lengths between routers in the vertical dimension due to their smaller height when compared to the length of ordinary lateral wires [1][2], and improve propagation delay on account of their superior electrical behaviour [3][4].

We present the baseline R3 router architecture for a three-dimensional *Network-on-Chip (NoC)* for transactional CMPs. The R3 is deployed in a 3D NoC composed of stacked 3x2 meshes. Each mesh contains four processing elements, a shared L2 data cache, and a speculative scheduler tightly coupled with an L2-instruction memory, as illustrated in Fig. 1. Each processing element resides inside a tile, with a local data cache, local instruction memory and their corresponding controllers, along with a *Network Interface (NI)* to connect the tile to the NoC. Application code is coarsely divided into atomic *transactions* that are executed speculatively on processing elements. During execution, each transaction's *write-set* is isolated until the transaction completes, at which time it is *validated* against other active transactions. In the absence of dependencies, the oldest transaction is allowed to *commit* its write-set to the shared L2 data cache, thus making its writes visible to all transactions [5][6][7].

Using the described transactional CMP architecture, we evaluate the performance of the baseline R3 router deployed in a stacked 3D NoC, highlighting the limitations of performance scale-up in terms of average packet latency and throughput with stacking. In order to provide perspective on the use of TSVs, we present data on the area penalty associated with their use in different configurations in 90nm UMC technology.

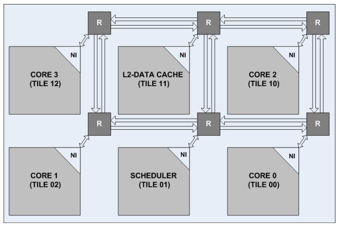

Fig. 1. Single layer 3x2 mesh

This paper is organised as follows. Section II provides an overview of related work. Section III describes the network architecture of the R3 based NoC, Section IV examines the R3 router architecture, highlighting its routing algorithm, flowcontrol and arbitration schemes. Section V describes the Through Silicon Via configurations in 90nm UMC technology

This research was supported in part by the CATRENE programme under the Computing Fabric for High Performance Applications (COBRA) project CA104.

used in the R3 layout. Section VI describes the simulation environment used for performance analysis of single layer and stacked meshes in the context of a transactional CMP. Section VII examines the performance of the R3 router deployed in a single layer and stacked mesh configuration, and provides insight into the stacking limit. Section VII also outlines the details of the synthesis-place and route carried out for the R3, and presents information on area penalty for various TSV combinations.

II. Related Work

Three-dimensional interconnect architectures have received immense attention over the last few years with the advancement of die-stacking technology. Pavlidis and Friedman in [8] provided evidence of the performance benefits of 3D NOCs by comparing the zero-load network latency of different interconnect configurations. The increase in average network latency in the vertical dimension was found to follow the same trend as observed in scaling 2D networks. More significantly, the increase in hop count per transmission was observed to be considerably lower while scaling vertically. Further to the work by Pavlidis, Weldezion et al. in [9] explored the scalability of 3D buffer-less networks-on-chip, analyzing their performance benefits over 2D meshes. While their work served to highlight the performance benefits of scaling in the vertical dimension, it did not examine the penalties incurred in the process.

Patti in [10] discussed the processes involved in fabricating TSVs in a real stacked system-on-chip. While this work demonstrated the ease of integrating TSVs into standard digital designs, a methodology for their integration was provided by Loi et al. in [11], which described the design flow for 3D NoCs using TSVs. The work presented an electrical model for TSVs based on extracted parasitics and described an implementation of a TSV based 3D NoC. However, no process technology was specified, making it difficult to compare the TSV size with interconnect dimensions and area metrics for the implemented logic.

Apart from the 3D NoC itself, router architectures utilizing TSVs were investigated in MIRA [12] and Picoserver [13]. Park et al. presented a 3D router architecture decomposed into blocks and placed on separate layers in a stacked configuration. MIRA represented the first such architecture spanning several layers in a stack. However, inspite of its design complexity, MIRA performed only marginally better than a baseline 3D network in terms of hop-count and latency. Additionally, MIRA assumed that processor cores themselves are decomposed into multiple layers. Such processor cores do not find their way into the mainstream very often due to the relative infancy of the design flow for such decomposition over multiple layers. This effectively limits the potential application of MIRA as a 3D interconnect architecture for CMPs.

Picoserver on the other hand was designed for CMPs in a stacked configuration. It used a shared bus architecture composed primarily of TSVs to enable communication between cores and shared memory in the CMP. However, its primary limitation lies in its scalability since it uses a bus. The authors note that system performance would saturate at 16 cores with their bus architecture.

III. Network Architecture

The network is organised as a packet-switched stack of 3x2 meshes with 36-bit unidirectional input and output links interconnecting nodes, as shown in Fig. 1. Packets may contain upto 64 bytes of payload data, and are transferred as 36-bit flits. Packets are routed across the network based on the *Destination Network Address* they carry. Each network router and its local tile are assigned a unique *Local Network Address* in the network. When a packet is received at a router with a destination network address that matches the local network address, it is forwarded to the local tile.

The *header flit* contains source and destination node addresses, packet size and transmission type. Transmissions may be of 3 types: Coherence Update (01), Cache Miss (10) and Instruction Block Transfer (11). A packet may consist of a number of *body flits*, depending on its size. Each body flit carries upto 32-bits of the payload. The *tail flit* is similar to the body flit, and in addition to carrying a 32-bit payload, it marks the end of a packet.

IV. R3 Router Architecture

The R3 router was designed to facilitate a three dimensional interconnect utilizing advanced TSVs. This input buffered wormhole router contains five lateral ports and two vertical ports, *Up* and *Down* for routing between layers in the stack, as illustrated in Fig. 2 and Fig. 3. It is primarily composed of three blocks: input block, switch and output block. This section examines the architecture of the R3.

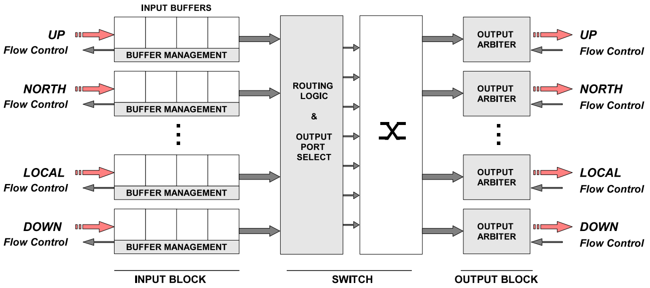

Fig. 2. R3 Router Architecture

A. Routing Algorithm

The R3 implements a dimension ordered static Z-X-Y routing algorithm that routes packets along the shortest path from source to destination. Packets are first routed in the Z-dimension, and consequently in the X and Y-dimensions. This ensures that packets in transit to other layers in the stack are routed through the TSVs onto their destination layer immediately upon injection into the network, reducing congestion in the layer meshes. Once packets arrive in their destination layer,

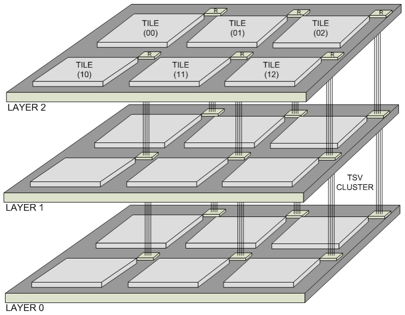

Fig. 3. Illustration of vertical interconnects in the stacked-die CMP

they are routed first in the X-dimension, followed by the Y-dimension. The nature of the algorithm ensures that incoming packets are never routed through the same port that they were received at, nor misrouted. Consequently, conditions such as *Deadlock* and *Livelock* are prevented from occurring in the network.

B. Flowcontrol

The R3 implements On-Off flowcontrol between nodes in the network. This is achieved by means of a flow control line between upstream and downstream nodes, which switches to a high state when the occupancy of the input buffer at the downstream router reaches a certain threshold. The pipelined nature of the router induces a two cycle latency for transmission stalls in the event of congestion at downstream routers. The *Buffer Management* module is designed to take this latency into account, and consequently, the flowcontrol threshold is *N-2*, N being the input buffer depth at each port. The R3 provides 12-flit deep input buffers at each of its seven input ports. This flowcontrol mechanism ensures that flits are never dropped, thereby eliminating the need for retransmissions.

C. Output Arbiter

The R3 uses dedicated arbiters for each output port, implementing a *Round Robin* arbitration scheme in order to ensure fair, *Best Effort* service to all input ports. Due to the routing restriction that prevents packets from being routed through the same port they arrived at, each output arbiter only polls six out of the seven input ports. When a header flit arrives at an input port, the Routing Logic & Output Port Select block raises a request to the appropriate output arbiter. If the requested output port is idle and the downstream router has free input buffer slots, the arbiter grants the request by asserting the *Drain* signal for the input buffer. Round robin arbitration is deactivated, and remains in that state until the tail flit of the packet has advanced through the output port. This ensures the integrity of routed packets, and prevents flits from contending packets merging in the output stream.

Contending input ports now wait for access to the output port, keeping their request lines asserted, and input buffers in the stalled state. In the event of input buffers becoming empty while flits of a packet are advancing through the router, the output arbiter sets the link as idle while maintaining the round robin arbitration in the deactivated state. This state of the output port holds until the tail flit has advanced and the complete packet has been routed. Upon completion, the arbiter asserts the *Next* signal and resumes round robin arbitration to service the next waiting input stream. The maximum packet size of 17 flits, i.e. a 64 byte payload, imposed by the network architecture ensures fair and timely arbitration to all input ports, and places a limit on the time the arbiter spends servicing any particular input. The three-stage R3 router thus has a minimum fall through latency of four cycles.

V. Through Silicon Vias

Through Silicon Vias facilitate inter-layer links between routers in the stack. We consider a face-to-back stacking scheme since it allows stack heights of more than two layers. TSVs consist of a central via with a keep-out area to ensure signal integrity in neighbouring interconnect lines. In order to estimate the area penalty associated with the use of TSVs, three full-custom configurations were designed in 90nm UMC technology with $5.6\mu m$ TSV width [2] and pitch as detailed in Table I. Subsequently, the cells were instantiated for each of the 74 lines in the Up and Down ports of the R3 to determine the area penalty for various TSV configurations. The TSVs for the Down port were placed under the Up port and aligned correctly to ensure proper contact when stacked. Fig. 4 shows the layout of the TSV, with the via surrounded by a keep-out area and a guard ring.

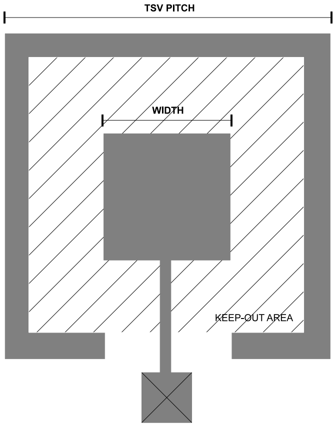

Fig. 4. Layout of TSV Cell

TABLE I
TSV Configurations

TSV Width (μm)	5.6		
TSV Pitch (μm)	11.2	25.2	50.4
Aggregate Cell Area (μm^2)	282.4	953.5	3144.96

VI. SIMULATION ENVIRONMENT

For performance evaluation, the R3 was deployed in stacked 3x2 meshes with traffic generators connected to the local port of each instance. Each traffic generator is tunable over a range of injection rates with variable packet sizes. In order to simulate traffic conditions in a transactional CMP, a network transaction was created comprising of:

1) *Instruction block transfer from scheduler to cores*
2) *Read from L2 in the event of a cache miss at a core*
3) *Commit arbitration*
4) *Coherence updates to cores and writes to L2*

As explained in the Introduction, transactions are atomic sections of program code which are transferred to the instruction memory of a free processor core from the scheduler. When execution begins, cache misses may occur in the L1 data cache resulting in accesses to the L2. During execution, writes to addresses in the L1 cache are buffered in a *Speculative Write Buffer*. Upon completion, these writes must be made visible to all processors in the system in the same order as they occur in the program code, and older shared copies invalidated. A transaction ready to commit must therefore arbitrate to make its writes global, and write back all modified L1 data to the L2 if it does not violate program order for shared data [5]. Transactions using data invalidated by a committing transaction are forced to restart. The network transaction was therefore configured with the parameters listed in Table II.

In our transactional CMP, transactional instruction blocks are similarly sized to equalize the core programming latency and to pipeline the start of transaction execution across cores. Cache writes by transactions are held in buffers till the entire transaction completes execution, and subsequently, all writes are committed to the L2 cache with cache line granularity. Consequently, writes to the L2 cache occur at a frequency dependent on transaction size, and the amount of data written into this cache depends on the amount of data modified during execution of the transaction. Therefore, communication between nodes in the CMP occurs during specific phases.

A majority of the traffic from the scheduler is injected immediately after the transactional application code is received through the programming interface. Since our CMP uses similarly sized transactions, traffic from the scheduler to the four cores consists of equal length instruction blocks, transferred in a pipelined manner. The processing cores on the other hand, inject traffic in a different pattern into the network. During execution of the transaction, traffic is directed only to the L2 data cache in the event of L1 data cache read misses. At the end of execution, the core requests the scheduler for permission to verify its write-set against other transactions executing in the CMP. This verification is performed by transmitting the addresses of modified cache lines to all cores in the system, as an invalidation packet. Modified cache lines are transferred to the L2 data cache after successfully completing verification. Finally, the scheduler is notified of completion of the transaction. These communications are illustrated in Fig. 5 using the traffic destination distribution of *Core 0* as an example. This also provides an overview of the injected traffic distribution from cores. Our simulations considered four combinations of transaction and write set sizes - large transaction with large write set, large transaction with small write set, small transaction with large write set and small transaction with a small write set. Small write sets are assumed with a worst case 100% L1 miss rate, while large write sets are simulated with a 35% miss rate. The simulation was therefore driven by synthetic traffic, with a packet destination distribution as given by the network transaction parameters in Table II. Incoming traffic at local ports of each router was analyzed by a Traffic Evaluator to determine average end-to-end packet latency and average throughput during simulation.

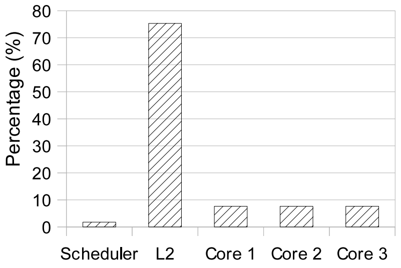

Fig. 5. Traffic destination distribution from Core 0 to each tile illustrated as a percentage of total injected traffic per transaction

TABLE II
NETWORK TRANSACTION PARAMETERS

	Dependence	Worst Case	Best Case
Instruction Transfer	Transaction Size*	192 byte	12 byte
L2 Reads**	L1-D Miss Rate	100%	35%
Commit Arbitration	Write-set size	192 byte	64 byte
L2 Write	Write-set size	192 byte	64 byte

* 4-byte instructions ** Percentage of write-set size

VII. RESULTS

Network performance was evaluated based on average throughput and average end-to-end packet latency obtained from simulation with the described network transactions over a range of injection rates after 500K cycles. A single layer 3x2 mesh was first simulated, with the vertical ports on each router disabled. Fig. 6 shows the variation in average packet latency with increasing injection rates, with average payload size of 32 bytes per packet for the single layer mesh. The flat shape of the curve at lower injection rates is due to the fair arbitration scheme and fixed maximum packet size that ensure that large transfers are divided into packets upto the maximum allowed size in order to prevent domination of any particular input stream at the output arbiter of routers. At higher injection rates, a brief dip is observed in average packet latency due to traffic conditions in the network which cause packet header flit arrival times to match their destination output port's arbitration cycle, resulting in the packet being routed through faster. The single layer R3 mesh exhibits an maximum aggregate raw throughput of 2.85GBps at 200MHz for the given synthetic traffic conditions.

The test setup was subsequently extended to a multilayer stacked mesh consisting of R3 routers with their vertical

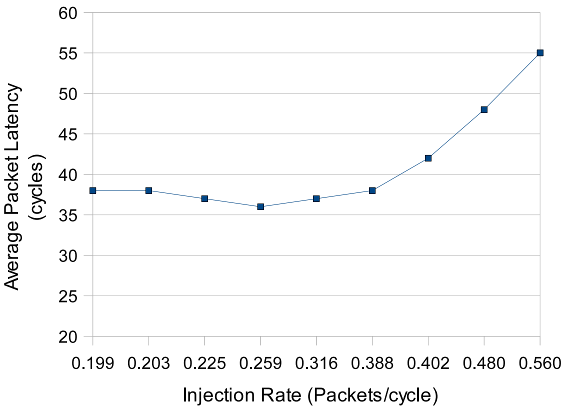

Fig. 6. Variation in average packet latency with injection rate for R3 in single layer 3x2 mesh with average payload size of 32 bytes/packet

ports enabled. Fig. 7 shows the variation in average packet latency with increasing injection rates in a three-layer stack of 3x2 meshes. The average payload size for packets was set to 64 bytes. The overall trend from the graph shows an increasing average latency with increasing injection rates. At low injection rates in the stack, we observe average latencies of approximately ten cycles higher than the latencies observed for the single layer R3 mesh at the appropriate injection rates. This increase in latency is due to the longer round robin arbitration cycle at the output arbiters on account of the enabled vertical ports in the R3.

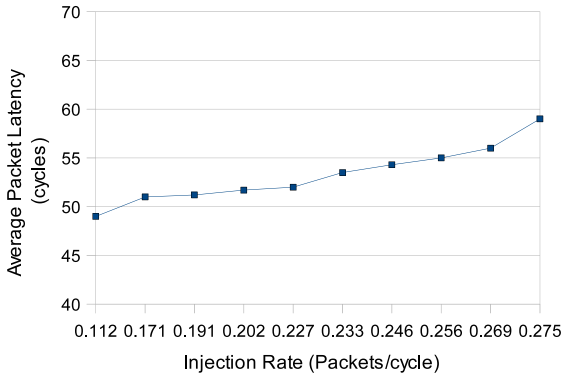

Fig. 7. Variation in average packet latency with injection rate for R3 in stacked 3x3x2 mesh with average payload size of 64 bytes/packet

The Z-first nature of the routing algorithm ensures that injected traffic is first routed in the Z-dimension to its destination layer thereby reserving X-Y routing resources for packets in their destination layer. Consequently, the stacked mesh exhibits network performance similar to the single layer mesh. While the two-layer R3 mesh delivered an aggregate raw throughput of 5.9GBps, the 3-layer mesh delivered a maximum of 8.3GBps. A graph of variation in throughput and average end-to-end latency versus the number of stacked layers is shown in Fig. 8. We observe an expected increase in latency as the stack height increases. Consequently, the rate of increase in average throughput becomes less rapid, evidenced by the flattening of the throughput curve in Fig. 8. For stack heights greater than five layers, the throughput curve becomes flatter and the incurred latency penalty begins to outweigh the increase in throughput obtained from adding additional layers to the stack. The flattening of the throughput curve may be postponed by prioritizing inter-layer traffic, by separating the arbitration cycles for lateral and vertical ports, thereby reducing the buffering requirements in vertical ports. Inter-layer latency may be further decreased by exploiting the superior parasitic performance of the TSVs. Both these approaches are being investigated for performance improvement of the R3. The R3 was synthesized for a Xilinx Virtex 6 device (XC6VLX75TFF784-3) and resulted in a resource utilization of 5942 LUTs (6%) at 200MHz. The design was further synthesized using Faraday standard cells in 90nm UMC technology with custom TSVs. The core router logic occupied an area of 0.158mm^2 including an area of 0.0177mm^2 for 12-flit deep input buffers at each port. A total of 148 TSVs were inserted in different configurations into the design for the Up and Down ports. The design was resynthesized for each configuration. The resulting area for each configuration is listed in Table III.

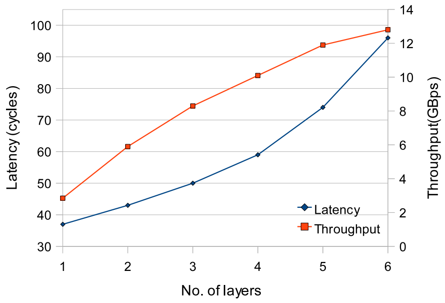

Fig. 8. Variation in average packet latency and aggregate raw throughput with increase in stack height.

From Table III, we see that small TSV pitches cause a minimal area overhead in the R3 design. Larger pitches imply a larger keep-out area, reducing the possibility of occurrence of signal integrity issues in neighbouring interconnect lines. However, larger pitches also increase the incurred area overhead from the use of TSVs. Fig. 9 shows the layout of the R3 in 90nm UMC technology, with the router core in the center surrounded by 11.2μm TSVs of the Up and Down ports. Only TSVs for the Up port are visible in the layout, while those for the Down port are on lower metal layers.

TABLE III
ROUTER AREA WITH VARYING TSV PITCHES

TSV Pitch (μm)	11.2	25.2	50.4
Router Area without TSV (mm^2)	0.158		
Router Area with TSV (mm^2)	0.199	0.298	0.62

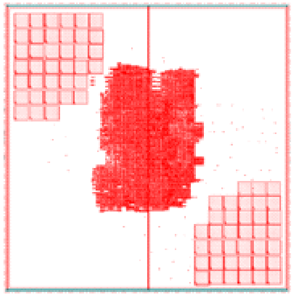

Fig. 9. Layout of R3 in 90nm UMC with custom TSVs

VIII. Conclusion

Maturing die-stacking technologies have allowed an increase in the computing power available on-chip, with transactional CMPs providing a promising platform for exploiting parallelism in application code. The need for communication between processor cores and the L2 cache places certain requirements on the latency, throughput and scalability of the interconnect that connects them. We presented the R3, a router architecture for three-dimensional NoCs in die-stacked transactional CMPs. Employing a simulation environment customized to emulate the transactional behaviour of processor cores in a transactional CMP, we evaluated the performance of the R3 in single and multilayer stacked configurations, reporting a low end-to-end latency and high throughput. We highlighted the stacking limit, observing a gradual decrease in the gained throughput with stack heights greater than five-layers. We attributed this to the increased latency, suggesting the separation of arbitration cycles for the lateral and vertical ports of the R3. Furthermore, with three configurations of full-custom TSVs in 90nm UMC technology, we highlighted the area penalty associated with the use of TSVs in a 3-dimensional network-on-chip. This paper presents the first part of our work with transactional CMPs and thus, topics such as separated arbitration cycles for vertical ports, schemes to exploit TSV performance along with a broader description of the transactional CMP will be included in our future work.

References

[1] S. W. Yoon, D. W. Yang, J. H. Koo, M. Padmanathan, and F. Carson, “3d tsv processes and its assembly/packaging technology,” in *3D System Integration, 2009. 3DIC 2009. IEEE International Conference on*, pp. 1 –5, sept. 2009.

[2] P. Franzon, W. Davis, and T. Thorolffson, “Creating 3d specific systems: Architecture, design and cad,” in *Design, Automation Test in Europe Conference Exhibition (DATE), 2010*, pp. 1684 –1688, march 2010.

[3] G. Loi, B. Agrawal, N. Srivastava, S.-C. Lin, T. Sherwood, and K. Banerjee, “A thermally-aware performance analysis of vertically integrated (3-d) processor-memory hierarchy,” in *Design Automation Conference, 2006 43rd ACM/IEEE*, 0 2006.

[4] K. Puttaswamy and G. Loh, “3d-integrated sram components for high-performance microprocessors,” *Computers, IEEE Transactions on*, vol. 58, no. 10, pp. 1369 –1381, 2009.

[5] M. Herlihy, J. Eliot, and B. Moss, “Transactional memory: Architectural support for lock-free data structures,” in *Computer Architecture, 1993., Proceedings of the 20th Annual International Symposium on*, pp. 289 –300, may 1993.

[6] C. Fu, H. Liu, X. Wang, D. Wen, and X. Yang, “Hardware transactional memory in multicore processors,” in *Information Engineering and Computer Science, 2009. ICIECS 2009. International Conference on*, pp. 1 –4, dec. 2009.

[7] L. Hammond, B. Carlstrom, V. Wong, M. Chen, C. Koryrakis, and K. Olukotun, “Transactional coherence and consistency: simplifying parallel hardware and software,” *Micro, IEEE*, vol. 24, pp. 92 –103, nov.-dec. 2004.

[8] V. F. Pavlidis and E. G. Friedman, “3-d topologies for networks-on-chip,” *IEEE Trans. Very Large Scale Integr. Syst.*, vol. 15, pp. 1081–1090, October 2007.

[9] A. Y. Weldezion, M. Grange, D. Pamunuwa, Z. Lu, A. Jantsch, R. Weerasekera, and H. Tenhunen, “Scalability of network-on-chip communication architecture for 3-d meshes,” in *Proceedings of the 2009 3rd ACM/IEEE International Symposium on Networks-on-Chip*, NOCS ’09, (Washington, DC, USA), pp. 114–123, IEEE Computer Society, 2009.

[10] R. Patti, “Three-dimensional integrated circuits and the future of system-on-chip designs,” *Proceedings of the IEEE*, vol. 94, no. 6, pp. 1214 –1224, 2006.

[11] I. Loi, F. Angiolini, and L. Benini, “Supporting vertical links for 3d networks-on-chip: toward an automated design and analysis flow,” in *Proceedings of the 2nd international conference on Nano-Networks*, Nano-Net ’07, (ICST, Brussels, Belgium, Belgium), pp. 15:1–15:5, ICST (Institute for Computer Sciences, Social-Informatics and Telecommunications Engineering), 2007.

[12] D. Park, S. Eachempati, R. Das, A. Mishra, Y. Xie, N. Vijaykrishnan, and C. Das, “Mira: A multi-layered on-chip interconnect router architecture,” in *Computer Architecture, 2008. ISCA ’08. 35th International Symposium on*, pp. 251 –261, 2008.

[13] 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,” *SIGPLAN Not.*, vol. 41, pp. 117–128, October 2006.