CINEFORUM
ISSN: 0009-7039
Vol. 66. No. S6, 2026 (Special Issue 6)

Configurable Hierarchical Wafer scale Switch Architecture for Scalable High-Bandwidth Interconnect Systems

Pandala Lekhyasree[1*], Dr.I.V.Prakash[2*] Mr.D. Satheesh[3*], Mr.P.Sharath Power[4*]

[1]Student of MTech in VLSI System Design, Department of Electronics and Communication Engineering, Sree Dattha Institute of Engineering and Science, Hyderabad, Telangana, India.
[2]Assistant Professor, Department of Electronics and Communication Engineering, Sree Dattha Institute of Engineering and Science, Hyderabad, Telangana, India
[3]Assistant Professor, Department of Electronics and Communication Engineering, Sree Dattha Institute of Engineering and Science, Hyderabad, Telangana, India
[4]Assistant Professor, Department of Electronics and Communication Engineering, Sree Dattha Group of Institutions, Hyderabad, Telangana, India
Corresponding Author
Mail Id:pandalalekhyasree1911@gmail.com

Abstract

This paper provided a hierarchical wafer scale switching architecture which is configurable and scalable and allows communication over a large number of dies at high bandwidth. The structure of the proposed fabric is legislation into the form of Outside, Middle, and Inside Switch Clusters connected with the help of special inter-cluster wiring modules. The architecture, unlike a classic flat two-dimensional mesh, separates the boundary traffic processing, middle redistribution and inner packet forwarding into two layers of communication. Every switch die has three programmable states of the ports unused, packet-switched, and circuit-switched. Packet-switched packet is handled by destination extraction, route-table shared, arbitration and crossbar forwarding, whereas circuit-switched packet take a low-overhead fixed route. Port mode, circuit destination and route-table entries are programmed by a global configuration controller. A shared, dual-table routing structure is used at die level in order to minimize hardware duplication and be available to all active input ports. The RTL-based modular design offers a platform that is sensible to assess hierarchical routing, customizable communication, and a huge scale switchover. A description of the system architecture, packet flow, routing strategy, arbitration mechanism, implementation methodology and evaluation metrics are provided in the paper. The results of simulations are deliberately intended to be inserted only after verification of simulation and synthesis.

Keywords—wafer scale integration, chiplet interconnect, network-on-chip, hierarchical switching, packet switching, circuit switching, route table, crossbar, RTL, VLSI.

1. Introduction

High-performance computing today along with artificial-intelligence acceleration, and data-intensive systems increasingly rely on bandwidth in communications, as opposed to arithmetic capability alone. With the rising cost and capability of producing very large areas of monolithic dies, alternatives have come by in the form of chiplet-based and wafer scale integration. These technologies place together several known-good dies on top of a platform built of interposers (or a package substrate or a wafer substrate) and count on the high density die-to-die connections to supply system-level connectivity.

A two-dimensional mesh is appealing since it is consistent with physical placement of die and constrains length of individual links. However, the communication diameter is increased with network size. When packets are propagating between nodes that are far apart, they are faced with an increased number of routers, increased arbitration delay and increased contention. These constraints get worse when tens or hundreds of switching dies are on the wafer. Thus, a high-performance wafer scale interconnect needs to maintain physical regularity and offers a more advanced logical communication structure.

This paper suggests a hierarchical architecture (made programmable), which breaks down the communication fabric into Outside, Middle and Inside Switch Clusters. The Outside Cluster offers ingress and egress connectivity, the Middle Cluster redistributes traffic and the Inside Cluster is used to support further progression within the internal. Port-mode controls and programmable route tables enable one switch fabric, physically same, to be used to implement different logical paths. It is also a design that merges general-purpose communication by use of packet switching together with circuit switching which is deterministic flows.

The main deliverable is a reusable modular switching architecture using RTL, consisting of reusable switch dies,

shared route tables, arbitration logic, crossbars, wiring modules, and a single central controller to configure the architecture. The design aims to minimize redundant routing memory, ease architecture scaling, and have a solid base for simulating, synthesizing and subsequent FPGA or ASIC implementation.
Structured packet progression with an Outside–Middle=Inside cluster hierarchy three levels. Tri-mode configurable ports can operate in unused mode, packet-switched mode and circuit-switched mode. A common

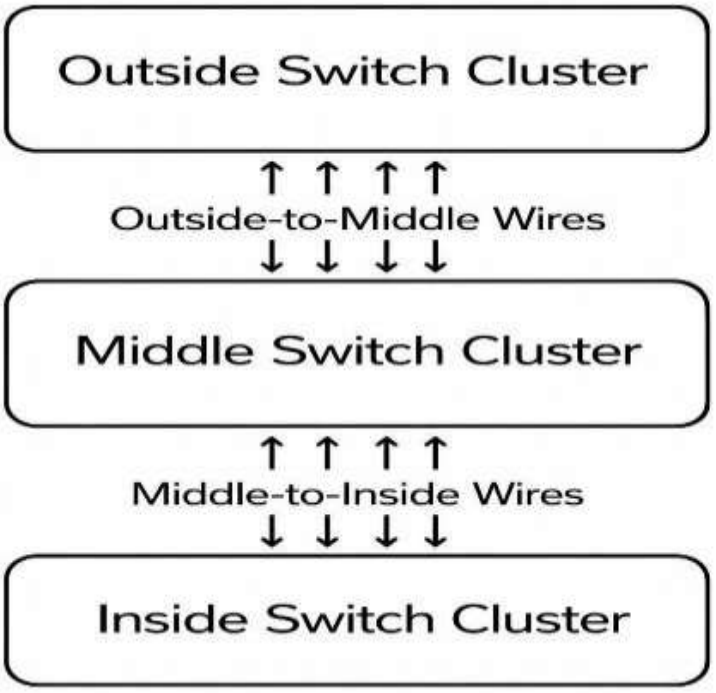


Fig. 1. Hierarchical interconnection of Outside, Middle, and Inside Switch Clusters.

dual-route-table format which does not replicate route-memories at each input port. A centralized route programming/circuit-path selection interface. An RTL architecture framework that can be easily verified and translated into hardware.

2. Related Work and Research Gap

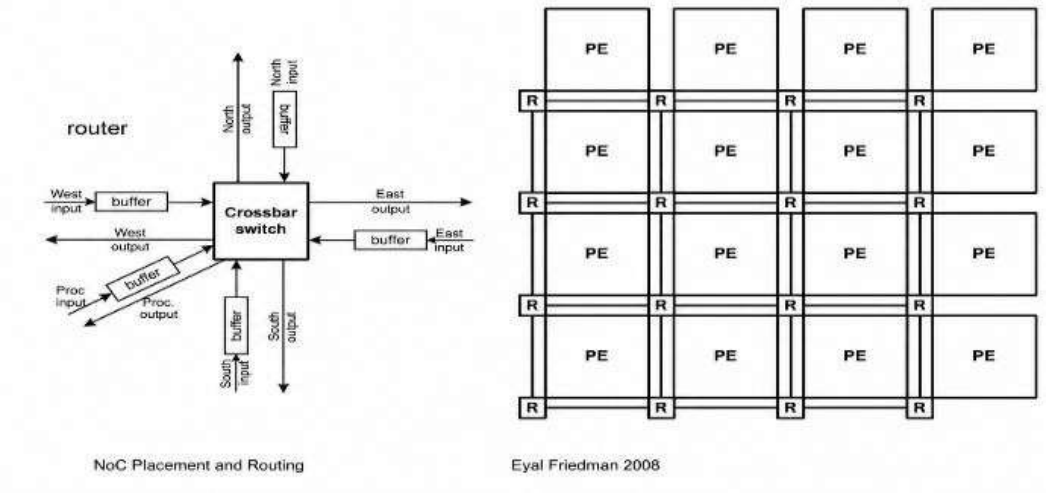
Heterogeneous systems have been achieved by chiplet integration which is a complex packaging of compute, memory, I/O and specialized accelerators. This idea is applied on a much larger scale to scale in wafer-scale integration, with large scale offering high aggregate bandwidth and short-reach die-to-die interconnections. Prior literature has shown massive chiplet arrays, wafer-based system-on-a-wafer packing and wafer-based machine-learning and high-performance workload compute fabrics [1]-[05].
A significant determinant of scalability is the network topology. Mesh networks are easy and layout adaptable however the number of hops and congestion in mesh networks are dependent on the diameter of the system. The tree and fat-tree networks have better logical communication, but are more difficult to layout on standard silicon maps [06]– [09]. Recent work thus highlights reconfigurable routing and logical-topology transformation where a conventional physical layout is programmed to make it appear like a more efficient network [10], [11], [12].
In hierarchical systems, routing-table-directed forwarding is desirable since the next hop is determined based on programmed topology, and not just geometric coordinates. Deterministic routing provides predictable (manageable) behavior, and limited path diversity is able to spread traffic without the complexity of completely adaptive algorithms [13].
The other missing piece is the absence of a well-defined modular RTL implementation taking into account wafer scale hierarchy, shared route memory, packet/circuit mode selection, inter-cluster wiring and centralized configuration within a single architecture. This gap has been closed by the proposed design which translates these principles into a top-level switching system and reusable hardware blocks.

3. Conventional Mesh-Based System

The baseline system is a system where every processing or switching node interfaces with their neighbors to the north, south, east and west. This architecture provides local connections that are local and low- frequency in nature and coordinate-based routing. Many intermediate routers may, however, be required to transfer a global transfer

of a packet. With increasing number of nodes, there is increased contention in central links, there is increased average throughput and the hop count goes higher. The deterministic bypass paths and workload-specific logical connectivity is also limited in a fixed mesh. It is these constraints that drive the hierarchical approach that can be configured.

Fig. 2. Conventional 2-D mesh communication topology and router organization.



4. PROPOSED HIERARCHICAL WAFER SCALE SWITCH ARCHITECTURE

The proposed system breaks down the switching fabric, into three communication layers. The layers are linked by delimited wiring modules and there is a clear delineation of ingress/egress handling, redistribution, and delivery within. Reusable configurable switch-die modules are used to construct all the clusters. The logical behavior is programed using programs in the global which do not alter the actual RTL structure.

4.1. Outside Switch Cluster

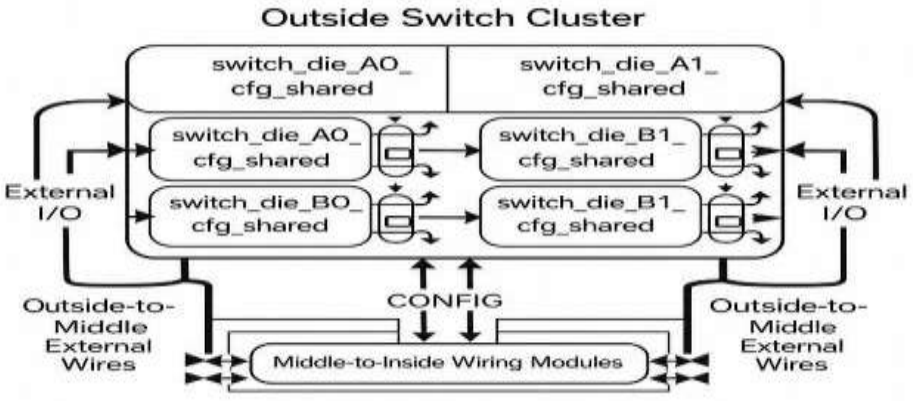


Fig. 3. Outside Switch Cluster.

The Outside Switch Cluster provides connection between external sources and destinations with the internal hierarchy. The external-facing connection of type A dies is supported, and the local distribution and transfer to the Middle Cluster are favored by B-type dies. Packet-switched traffic coming in is treated to route look up, arbitration and crossbar forwarding. CSTA is directly linked to a programmed output.

4.2. Middle Switch Cluster

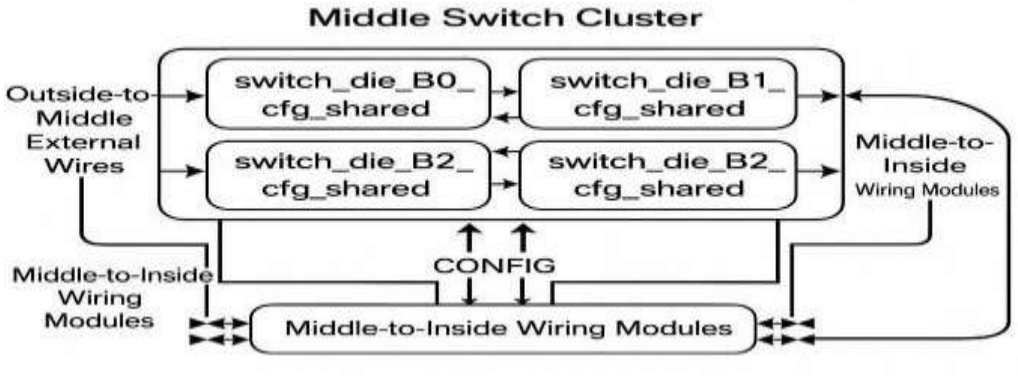


Fig. 4. Middle Switch Cluster.

The main stage of aggregation and redistribution is the Middle Switch Cluster. It is fed by one or more outer paths, and implements the programmed route policy, both contention resolution and packet forwarding, to the Inside Cluster or a paths in another middle-layer. Future load balancing and quality-of-service additions should be located in this layer.

4.3. Inside Switch Cluster

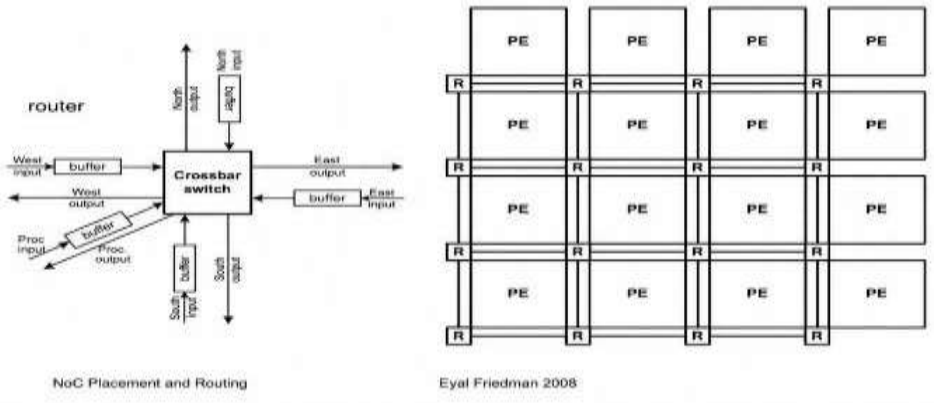


Fig. 5. Inside Switch Cluster.

This is the Inside Switch Cluster that constitutes the innermost switching zone. To maintain the internal hierarchy while making the RTL manageable a simplified 2x2 B-die structure is maintained. The routing and arbitration and crossbar principles are retained as well and the routing principle is unified across the hierarchy levels.

5. SWITCH-DIE MICROARCHITECTURE

5.1. Port-Mode Controller

The programmable mode value is stored in each physical port. In the unused mode, input requests do not get processed and an output activity is unavailable. When using the mode of packet-switch it is the packet that is inserted into the routing pipeline. The input in the mode of circuit switching is attached to a predetermined destination port. The mode encoding will be 00: unused, 01: packet switching, and 10: circuit switching.

5.2. Shared Dual Route Table

A single copy of a route is kept in a die, rather than being replicated in every input. The active inputs enter their destination address and packet id to the shared look up table. One of the two route tables is picked by a selector based on the context of a packet. The configuration minimizes redundancy of memories and still maintains a variety of programmable forwarding options.

5.3. Request Generation and Arbitration

In the case of packet-switched traffic, the output of a route-table identification detects the output requested of a route-table. In cases where more than one priority is asked on the part of the input, an allocator picks a requester.

A round-robin allocator is more fair but constant priority could be employed in a simplified system. Grant vector: It is necessary to make sure that one input in a cycle drives a specific output.

5.4. Crossbar Forwarding

The crossbar maps received the input words onto the chosen outputs. It is determined by the outcome of the arbitration and it is independent of routing policy. This separation enables the route lookup and allocator to be changed autonomously of the physical data-transfer structure.

5.5. Wiring Module

Dedicated Outside-to-Middle and Middle-to-Inside wiring modules transfer channels between adjacent clusters. The module may support pass-through, reversal, Channel-to-Channel: Between clusters, there is dedicated Outside-to-Middle and Middle-to-Inside wiring of channels. The module can support pass through mappings, reversal, rotation mappings or disabled mappings. Modeled as a distinct block, inter-layer wiring can be much easier to debug, and allows alternative physical mappings without having to re-design the switch dies.

5.6. Global Configuration Controller

Port modes, circuit destinations and route-table entries are written out by the controller. It decouples management policy and data path and enables a common hardware to support alternate logical topologies. In a future implementation, configuration can be loaded at booting or via a serial interface, or by a dedicated management processor.

6. Packet Processing and Routing Method

A packet will include fields consisting of a payload at least, destination and packet-identification. Upon a valid input being received the port controller decides the mode of active switching. The packets mode rips out the destination and the packet number which is utilized in routing. During circuit mode the route lookup is bypassed.

$r_i = T_s[d_i], \quad s = f(d_i, p_i)$

In this case, d i is where input i is going, p i is the packet identifier of input i, S is to choose between the two route tables T n is the chosen table, and r i is the output requested as the next-hop. The requests are transformed in grants through arbitration:

g simultaneously equal to 1 and 1 implies that there is some choice of input i to be sent to and read by output j. The transfer y = x i of the input-output pair that is granted is then carried out by the crossbar. The hierarchical route through the Inside, Outside and Middle layers follows the programmed route entries. This process is reciprocated in the opposite direction with packets in the internal region.

7. Performance Metrics

The architecture can be measured with respect to functional and hardware-oriented measures. The key network level measures are defined below using equations.

Average hop count: $H = \frac{1}{N_r} \sum^{N_r} H$

Average packet delay: $\overline{D}_{avg} = \frac{1}{N_r} \sum_{i=1}^{N_r} (t - t)$

Throughput $T = \frac{\sum_{i=1}^{N_r} r, s,i}{N_r \times L_p}$

Δt

Packet delivery ratio: PDR = Nr / Ns × 100%
Ns and Nr are the number of packets sent and received respectively, H i is the count of hops a packet has travelled, Lp is the length of packet, and dt is observation time. Added to RTL implementation should be reporting of lookup-table or gates usage, the number of registers, memory usage, maximum frequency, critical path delay, and estimated power.

8. RTL Implementation and Verification Strategy

Test and debug the route-table memory without configuration writes, or mixed configuration write/read requests. Test all port modes with directed tests: disabled port, packet-switched forwarding and direct circuit-switched bypass. Design instances of contention where, with an identical input, two or more inputs demand the same output and establishing deterministic arbitration. Install switch dies, in each cluster and check intra cluster transfers and connect inter-cluster wiring. Go between routes via the global controller and test end-to-end Outside-to-Inside and Inside-to-Outside packet delivery. Check used assertions or scoreboard logic to verify the integrity of a payload, its destination, lack of duplicate grants, and correct handshaking. Compile a full top-level module and generate area, timing and power reports against which to compare with a mesh-based baseline.

9. Results and Discussion

This section includes behavioral, synthesized device, RTL schematic, power estimate as well as the synthesis status of the developed switch-cluster design (configurable) design. The provided Vivaldo outputs will confirm the design hierarchy can be elaborated and synthesized, but the behavioral waveform also shows that the testbench did not drive a number of top-level inputs. The results of the synthesis therefore can be considered valid, but the current simulation cannot yet be considered general functional verification.

9.1. Behavioral Simulation Result

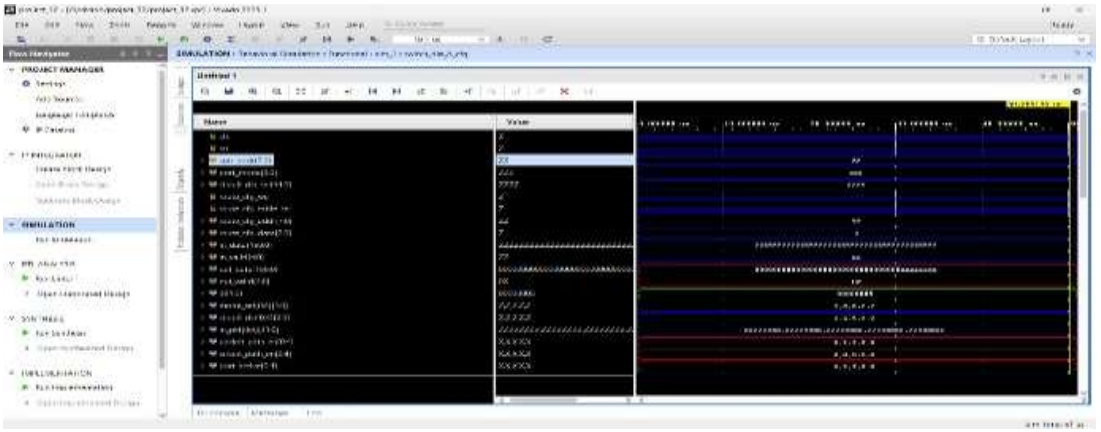


Fig 6. Behavioral simulation waveform of the configurable switch-die module.

The behavior simulation was run about 51 u s. The clock, reset, configuration address, port mode, circuit-destination selection, routing configuration, input data, valid signals, output data, output-valid signals and internal routing-control signals are provided in the waveform window.
The most valuable observation is that both clock and reset signals are shown as high value of impendence (Z). Most of the input buses are also at Z and a number of internal control signals will be unknown (X). The output-data bus initially has zeros but subsequently acquires unknown bits and also the output-valid bus has an unknown state. These waveforms show that the testbench is not supplying completely defined stimulus to the design-under-test. There is no clock generator assigned or connected to the DUT clock port or no clock generator at all. There is no known logic level to which Reset is driven to in the course of initializing. Part or all of the simulation has configuration, routing, packet and valid inputs that are left floating. Unknown X values can be propagated in the routing logic that is combinational and they are shown on the outputs.
Thus, Figure 1 can only be viewed as the results of the design being elaborated and implemented successfully and not to establish the design to have actually worked with its routing capabilities. A valid verification waveform should indicate a toggling clock, specified reset sequence, known configuration values, valid input packets and deterministic output transitions.

9.2. Synthesized Device Placement

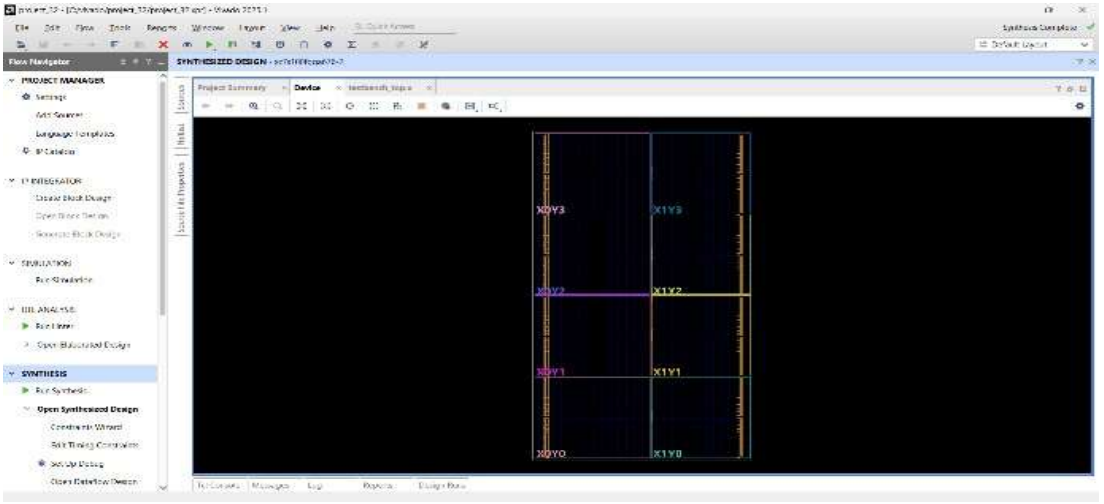


Figure 7. Synthesized device view showing the placement regions used by the design.

The generated device view will display the target FPGA broken into several clock areas or placement areas and the areas are named X0Y0 up to X1Y3. The horizontal and vertical highlights of the routing lines show that the switch-cluster logic covers multiple areas on the device. This is the expected distribution with a large number of ports with large interconnection resources in a design.

The success of technology mapping to the chosen FPGA device is verified by the device view. This screen is, however, a post-synthesis view and not a conclusive post implementation placement. Conclusion regarding congestion, timing and the length of the route will only be accurate once implementation stage is completed and implemented-device view inspected.

9.3. RTL Elaborated Schematic

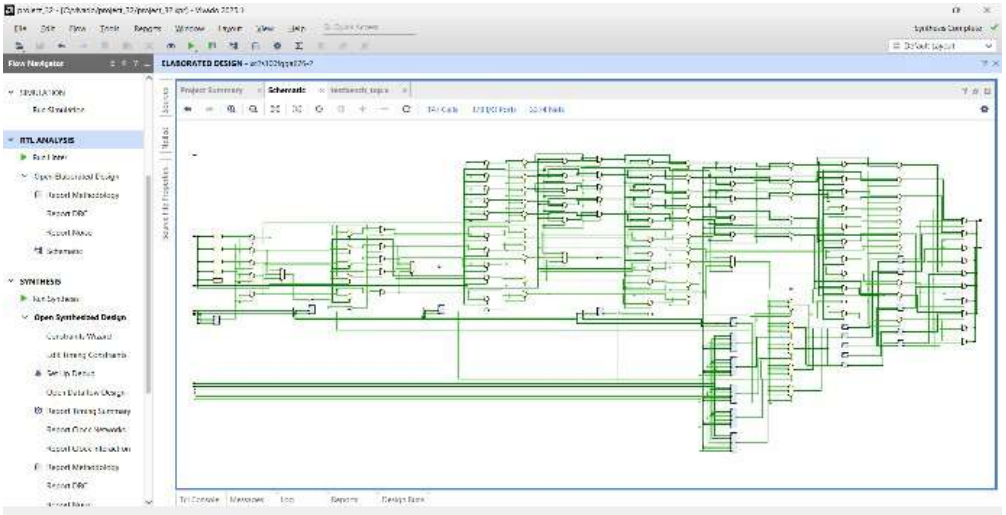


Fig 8. Elaborated RTL schematic of the configurable switch architecture.

The elaborated schematic has 147 cells, 378 I/O ports and it has 2274 nets. The huge array of nets and I/O connections is compatible with a configurable multi-port routing arrangement whereby input data, routing modes, destination selections, packet-path enables and configuration words are replicated over multiple identical routing blocks.

We can observe repeated structures of logic throughout the schematic as an indication that the design has been

synthesized into a number of parallel routing channels (or switch elements). The hierarchical and scalable architecture is wholly backed by the intended schematic. The fan-out and lengthy interconnection tracks evident in the schematic might actually become significant sources of timing and dynamic power once made real.

9.4. Synthesis Status

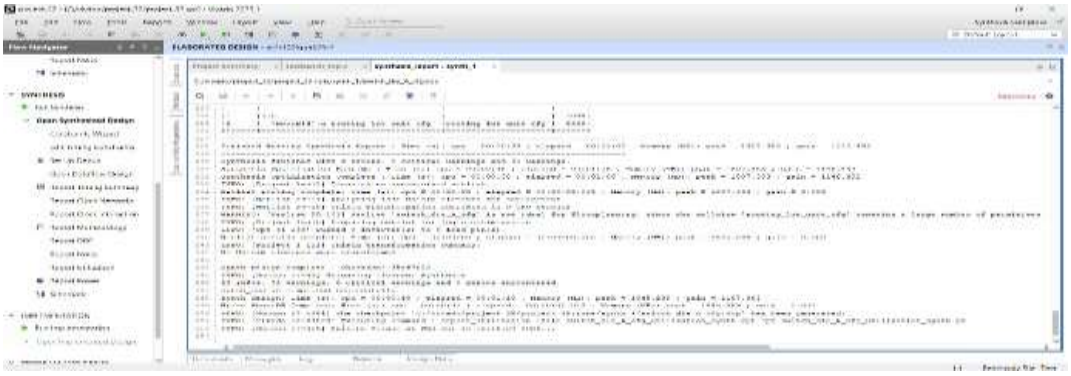


Fig 9. Vivado synthesis completion report.

Synthesis by Vivado was successful and had no errors or critical warnings. The report indicates that the checkpoint and utilization report created was done correctly. It took a total of roughly 1 minute and 10 seconds of elapsed time to complete the synthesis process consuming an average of 1845 MB of memory.

It includes a warning that the netlist generated is not optimal to use in floor planning since the routing lut unit cfg cell view has a high number of primitives. It is not a failure of synthesis. It implies that hierarchy is rather thick and might not be convenient to deal with as one floor planning unit. When physical optimization is necessary hierarchy preservation, modular synthesis, or smaller floor planning partitions can be taken into account.

Parameter	**Observed result**
Synthesis completion	Successful
Errors	0
Critical warnings	0
Warnings	Approximately 31-32
Elapsed synthesis time	Approximately 00:01:10
Peak memory	Approximately 1845 MB
Generated checkpoint	Switch die A cfg.dcp

9.5. Power Estimation

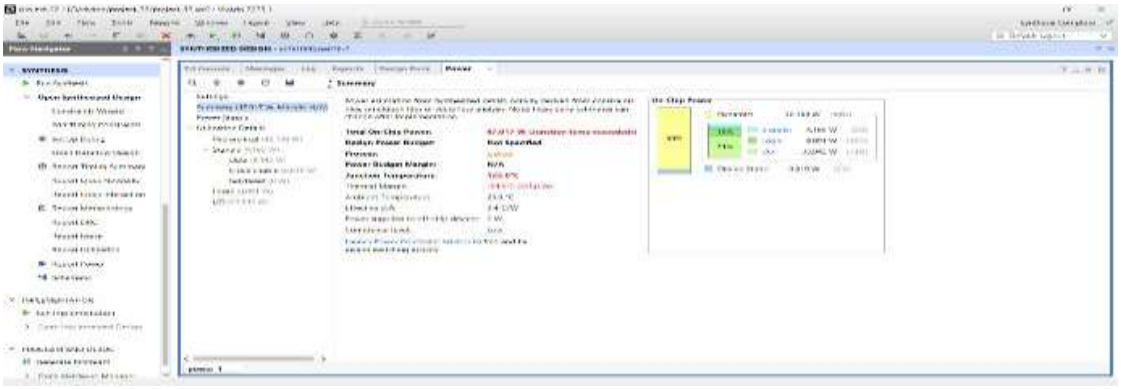


Fig 10. Post-synthesis power estimation summary.

The power report post-synthesis estimates 47.017 W of total on-chip power, and it is broken down to 46.198 W which is dynamic power (about 98 percent) and 0.819 W (about 2 percent) which is device-static power.

Power component	Power	Share / observation
Signals	4.165 W	9% of dynamic power
Logic	8.091 W	18% of dynamic power
I/O	33.942 W	73% of dynamic power
Device static	0.819 W	2% of total power
Total on-chip power	47.017 W	Junction-temperature limit exceeded

The report will give a junction temperature estimate of 125.0 o C and a negative thermal margin of -54.6 o C. Confidence level is denoted by low. The values suggest that the final hardware power result obtained should not be relying on the current estimate.
The overwhelming reason of the total power estimate is due to the exceptionally high I/O power. This is usually due to the opening up of hundreds of wide top-level ports in a design and the default switching activity being applied by Vivado, I/O standards, clock assumptions, or unconstrained toggle rates. The 378 I/O ports of the design presented in the RTL view correlate with why the I/O contribution is estimated to be huge.
Set the XDC clock constraint of use with create clock. Provide realistic I/O standards, drive strengths and output loads. Simulate an entire behavior using known stimulus and produce a SAIF file of switching-activity. Introduce the information on activities in the power analysis and repeat the power estimation once the acts have been implemented.Where it is allowed in the architecture, open excessive top-level ports or serialize/group wide interfaces.
The Vivado models prove the configurable switch-cluster architecture to be syntactically sound, able to be elaborated into a large, connected RTL network, and successfully synthesized to the FPGA desired. The resulting design has 147 cells, 378 I/O ports and 2274 nets which in turn validates the adoption of strong multi-port routing structure.
Meanwhile, the present stature of the behavioral waveform carries both Z and X states as the important testbench inputs are not driven. Therefore, the functionality of packet routing, configuration choice and output-valid behavior is still yet to be functionally checked. The power and 125 C junction temperature are also tentative and most likely exaggerated as I/O power is in control of the report and the tool suggests the estimation to be low-confidence.

10. Conclusion and Future Work

An architecture of a hierarchical wafer scale switch system has been proposed to design scalable high-bandwidth interconnect systems. The layout plans traffic among Outside, Middle, and Inside Switch Clusters and has special inter-layer wiring to provide modularity. Tri-mode ports enable the physical fabric to take care of packet switching, direct circuit paths or disabled links. Dual route tables are shared to minimize replication of memory and a global configuration controller is used to provide programmable logical connectivity. The architecture converts concepts of wafer scale switching to reusable RTL implementation (based) on route lookup, arbitration, and crossbar modules. The final experiment that is done ought to prove the functional correctness and quantitate the hop count, delay, throughput, area, timing and power behavior improvement over a conventional mesh baseline.
It may be extended in the future with virtual channel, credit-based flow control, arbitrage based on traffic awareness, adaption of routing, fault-tolerant reconfigurations, quality-of-service arbitration, and deadlock verification. It can also be scaled to bigger arrays of clusters and mapped to both FPGA and ASIC platforms. It could be linked to a software route compiler to allow automatic creation of contents of route tables to apply to application-specific logical topologies.

References

[1] T. Li, J. Hou, J. Yan, R. Liu, H. Yang, and Z. Sun, “Chiplet heterogeneous integration technology—Status and challenges,” Electronics, vol. 9, no. 4, p. 670, 2020.
[2] S. Y. Hou et al., “Wafer-level integration of an advanced logic-memory system through the second-generation CoWoS technology,” IEEE Transactions on Electron Devices, vol. 64, no. 10, pp. 4071–4077, 2017.
[3] S.-R. Chun et al., “InFO_SoW (system-on-wafer) for high performance computing,” in Proc. IEEE ECTC, 2020, pp. 1–6.
[4] R. Mahajan et al., “Embedded multi-die interconnect bridge (EMIB)—A high density, high bandwidth

packaging interconnect," in Proc. IEEE ECTC, 2016, pp. 557–565.
[5] S. Pal et al., "Architecting waferscale processors—A GPU case study," in Proc. IEEE HPCA, 2019, pp. 250–263.
[6] W. Wang et al., "Demonstration of a wafer-level integration for system-on-wafer architecture," in Proc. ICEPT, 2023, pp. 1–4.
[7] S. Lie, "Cerebras architecture deep dive: First look inside the HW/SW co-design for deep learning," IEEE Hot Chips 34, 2022.
[8] B. Chang, R. Kurian, D. Williams, and E. Quinnell, "DOJO: Supercompute system scaling for ML training," IEEE Hot Chips 34, 2022.
[9] Y. Feng and K. Ma, "Chiplet actuary: A quantitative cost model and multi-chiplet architecture exploration," in Proc. ACM/IEEE DAC, 2022, pp. 121–126.
[10] S. Pal et al., "Designing a 2048-chiplet, 14336-core waferscale processor," in Proc. ACM/IEEE DAC, 2021, pp. 1183–1188.
[11] J. Flich et al., "A survey and evaluation of topology-agnostic deterministic routing algorithms," IEEE Transactions on Parallel and Distributed Systems, vol. 23, no. 3, pp. 405–425, 2012.
[12] J. Chen, C. Li, and P. Gillard, "Network-on-chip topologies and performance: A review," in Proc. NECEC, 2011, pp. 1–6.
[13] L. Benini and G. De Micheli, "Networks on chips: A new SoC paradigm," Computer, vol. 35, no. 1, pp. 70–78, 2002.
[14] C. E. Leiserson, "Fat-trees: Universal networks for hardware-efficient supercomputing," IEEE Transactions on Computers, vol. C-34, no. 10, pp. 892–901, 1985.
[15] W. J. Dally and B. Towles, Principles and Practices of Interconnection Networks. Elsevier, 2003.