Multidisciplinary Journal for Applied Research in Engineering and Technology (MJARET)

Vol. 2 Issue 5, July – 2022, Pages: 60-67

DOI : https://doi.org/10.54228/mjaret07220013

ISSN : 2583-1100

AN EFFICIENT STROBE SWITCH BASED CHIP-TO-CHIP INTERCONNECT FOR NNA's

Pavithra K
M.E.(Applied Electronics) - Department of ECE
CK College of Engineering & Technology
Cuddalore, India
pavithrakrishnan06@gmail.com
Manikannan G
Assistant Professor - Department of ECE
CK College of Engineering & Technology
Cuddalore, India
pavithrakrishnan06@gmail.com

Abstract: Modern multi- or many-core devices may handle complicated data flows with energy efficiency by using network-on-chips (NoCs). A Neural Network Accelerators (NNA) is a linked collection of nodes called neurons that generally consists of many processing nodes. To manage the high neuron-to-neuron traffic, a multicore device might be used. Massive volumes of multicast-based traffic are sent on-chip or between chips, making the interconnection network design more difficult and posing a performance and energy barrier for NN systems. For virtual-channel routing, the proposed system combines scoring crossbar arbitration, arbitration interception, and route calculation parallelization approaches, resulting in a high-throughput NoC with less expensive hardware for multicast-based traffic. Every time this system connects and disconnects, it will automatically self-repair the connection and give a workable connection route. Additionally, the network configuration will often alter depending on the cost value. With that route controller and switch assigned, a strobe cross bar switch is implemented for improved speed and performance. The power, area, logical, and speed of the proposed architecture are contrasted with those of the current design.

Keywords— Chip-to-chip interconnection, deep neural network (DNN), hardware accelerator, interconnection architecture, network-on-chip (NoC).

1. INTRODUCTION

In addition to expanding the number of nodes, designers now must deal with the issue of on-chip interconnects. The inability of systems employing conventional bus systems to scale means that they are unable to meet demands for future SoC in terms of hardware usage, performance, predictability, power consumption, and timing factors. Data collision risks are decreased since each node releases a data packet after receiving the token. Under high traffic, ring topology outperforms bus topology thanks to token passing. No server is required to manage communication between nodes. A network arrangement known as a ring topology uses device connections to create a circular data flow. Every networked device is linked to two other ones by two points on a circle. A ring network is the collective name for the devices arranged in a ring topology. Data packets go from one device to the next in a ring network until they arrive at their final destination. A unidirectional ring network, which characterises the majority of ring topologies, allows packets to go only in one direction. Others, known as bidirectional, allow data to flow in either way. The main drawback of a ring topology is that if any one link inside the ring fails, the network as a whole is impacted. Both local area networks (LANs) and wide area networks (WANs) may use ring topologies (wide area

networks). A coaxial cable or an RJ-45 network cable is used to link computers together in a ring topology, depending on the kind of network card each computer has.

2. RELATED WORKS

In recent years, hardware acceleration of NNs has attracted a lot of attention. There are, however, surprisingly few efforts on the interconnection networks required to execute DNN. A hybrid ring-mesh NoC called Neu-NoC [17] was suggested for neuromorphic devices with a focus on speeding multilayer perceptrons. To implement multicast traffic, Neu-NoC connects the neurons in the same layer via a single ring, and the neurons in that ring share the same data. For the purpose of implementing data transfers across various levels, these local rings connect to one another through a mesh NoC[9]. Ring topology, however, generally has performance and latency issues. A unique, clos topology-based direct interconnection network for feed-forward NNs was suggested by ClosNN [18].

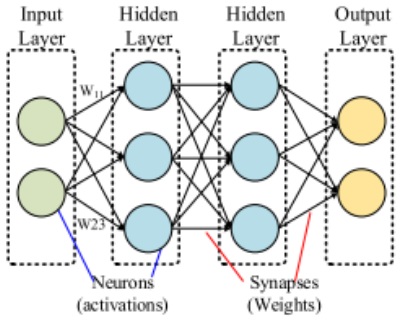

Fig. 1. *Simple Neural Network.*

It tries to solve a tree's low bisection bandwidth and a mesh topology's large diameter, both of which exhibit energy efficiency while processing multicast-based traffic. The design, however, is constrained by wire physical restrictions. A hierarchical mesh network (HM-NoC) for DNNs was suggested by Eyeriss-v2 [14]. Clusters are formed from the processing elements (PEs) and global buffers (GLBs), which are linked by HM-NoC. With circuit-switched routing, the NoC may be set up in a variety of configurations depending on the kind of sent data. The customised NoC enables the transmission of various data kinds, including input activation, weight, and partial sums, between PEs and GLBs while providing high bandwidth and high data reuse. DaDianNao [16] used fat-tree topology for intrachip connectivity and mesh through Hyper Transport 2.0 to control data flows between chips for large-scale NN systems. However, since the intrachip and interchip designs are different, they do not completely account for the multicast traffic within DNNs, which makes the interconnection network the acceleration system's bottleneck.

3. BACKGROUND AND MOTIVATION

A. DNNs

An NN is an interconnected group of nodes called neurons. The operation of a neuron can be expressed as follows:

$$\mathrm{OUT} = \Psi\left(\sum_{i=0}^{N-1} W_i * \mathrm{IN}_i + b\right) \quad (1)$$

It tries to solve a tree's low bisection bandwidth and a mesh topology's large diameter, both of which are effective at managing multicast-based traffic while using little energy. However, wire physical restrictions are a problem for the design. A hierarchical mesh network (HM-NoC) was suggested for DNNs by Eyeriss-v2 [14]. Global buffers (GLBs) and processing elements (PEs) are arranged into clusters and linked by HM-NoC. Depending on the kind of sent data, the NoC may be configured into a variety of modes using circuit-switched routing. Different data kinds, including input activation, weight, and partial sums, are exchanged between PEs and GLBs via the customised NoC, enabling high bandwidth and high data reuse. For large-scale NN systems, DaDianNao [16] used mesh through Hyper Transport 2.0 for intrachip connection and fat-tree topology for interchip communication. The interconnection network becomes the acceleration system's bottleneck since the distinct intrachip and interchip architectures do not completely account for the multicast traffic within DNNs.

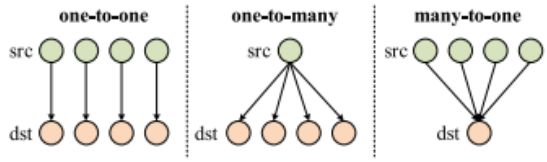

Fig. 2. Traffic patterns inside DNN accelerators.

TABLE I
SPECIFICATION OF TRAFFIC PATTERNS INSIDE DNN ACCELERATORS

Traffic Pattern	Operations in DNNs
One-to-One	Sending input activation/weight from GLB to a PE.
One-to-Many	Sending input activations/weighs from GLB to PEs.
Many-to-One	I. Adding partial sums to generate output activation. II. Moving output activations from PEs to GLB.

B. TRAFFIC PATTERNS

Each hidden/output neuron in dnn accelerators in dnns is linked to a limited area or all of the input neurons (for example, convolutional layers/fully connected layers). the neurons in the layer above get their input from the neurons in the layer below through their outputs. As shown in fig. 2, communications take

Multidisciplinary Journal for Applied Research in Engineering and Technology (MJARET)

Vol. 2 Issue 5, July – 2022, Pages: 60-67

DOI : https://doi.org/10.54228/mjaret07220013 **ISSN : 2583-1100**

place across several levels, mostly using one-to-one, one-to-many, and many-to-one patterns.
Without losing generality, we assume a dnn accelerator with a large pes for neuron computation and a glb for inputs, weights, and intermediate values coming from dram. It is possible to abstract the mapping of dnn dataflows across these pes to carry out one of the three communications shown in table i.

C. MOTIVATION

Two important insights serve as our inspiration. First, from the standpoint of algorithms, there is a clear tendency toward bigger nns, as shown by the accuracy gains on the image net visual recognition challenge with larger models (see fig. 3 [1], [18], [19]-[26]). The 2014 ImageNet Challenge was won by Google [19]. With almost 4 million parameters, it attained a top-1 accuracy of 74.8 percent. Senet [23], which has 82.7 percent top-1 accuracy with over 145.8 million parameters, won the 2017 ImageNet Challenge. The model's dimensions were raised from 36. Amoeba net-b [26], a newer model, even has over 550 million parameters.

Second, long-range connectivity among cores/chips arises as a new issue for high-performance and energy efficiency from a hardware standpoint as nn's size increases. As the distance between the source and the destination grows, core-to-core/chip-to-chip communications' energy usage rises fast. For instance, an inter-core/inter-chip data transfer in the IBM True North system [27] might use 224 more energy than an intra-core/intra-chip one (i.e., 894 versus 4 pj per spike per hop).

4. NEURON LINK

Two important insights serve as our inspiration. First, from the standpoint of algorithms, there is a clear tendency toward bigger nns, as shown by the accuracy gains on the image net visual recognition challenge with larger models (see fig. 3 [1], [18], [19]-[26]). The 2014 ImageNet Challenge was won by Google [19]. With almost 4 million parameters, it attained a top-1 accuracy of 74.8 percent. Senet [23], which has 82.7 percent top-1 accuracy with over 145.8 million parameters, won the 2017 ImageNet Challenge. The model's dimensions were raised from 36. Amoeba net-b [26], a newer model, even has over 550 million parameters.

Second, long-range connectivity among cores/chips arises as a new issue for high-performance and energy efficiency from a hardware standpoint as nn's size increases. As the distance between the source and the destination grows, core-to-core/chip-to-chip communications' energy usage rises fast. For instance, an inter-core/inter-chip data transfer in the IBM True North system [27] might use 224 more energy than an intra-core/intra-chip one (i.e., 894 versus 4 pj per spike per hop).

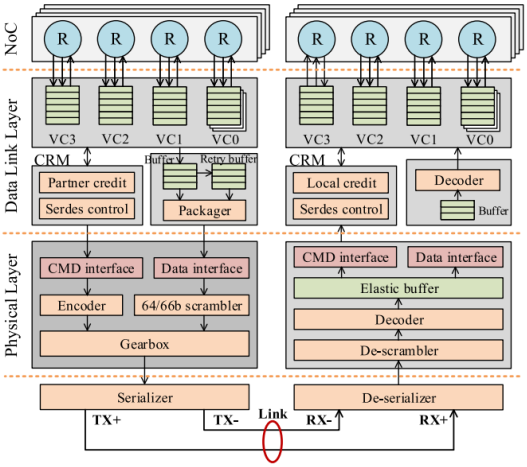

Fig. 3. *Overall interconnect architecture. It consists a physical layer, a data link layer, and a transaction layer (NoC).*

to the flit address within the retry data command. At the same time, the CRM unit sends related commands to indicate the retrying operation.
The data connection layer sends instructions and data to the physical layer. For data transmission and command transfer, we use asynchronous FIFOs and asynchronous handshake mechanisms. The problem that instructions need to have a higher priority than data is addressed by the use of an asynchronous handshake technique (e.g., for retransmission, commands are required to transmit before the data). The encoder then adds the synchronous header, check header, and package header to the data or instructions. The scrambler then makes use of the IEEE 802.3 standard to equalise the number of "0" and "1" values and prevent consecutive "0" or "1" values. There will be no scrambling of the encoded message. Therefore, to increase parallelism, these two steps can be carried out concurrently. In order to prevent errors on the chips, control fields are lastly created by the byte striping module.
After block boundary alignment and channel bonding, the data processed by the physical medium attachment (PMA) sublayer is recovered to normal data at the receiving end. The information is then returned to its original sequence and delivered to the decoder. Data from the recovery clock is then synchronised to the

local clock using the elastic buffer. By identifying the sync header, command interfaces and data interfaces transmit commands or data to the data link layer. The CRM at the data connection layer examines a command and replies appropriately. When sending data, it is prioritised and addressed before being buffered and transmitted to other VCs. The CRM creates a retry command to demand that the transport side retry data and relinquish any data until it gets the retry data after the data has failed the decoder's check.
Optimizing the routing of on-chip VCs
Fig. 5 depicts our router architecture, which is an improvement on a standard router. There are five input ports.

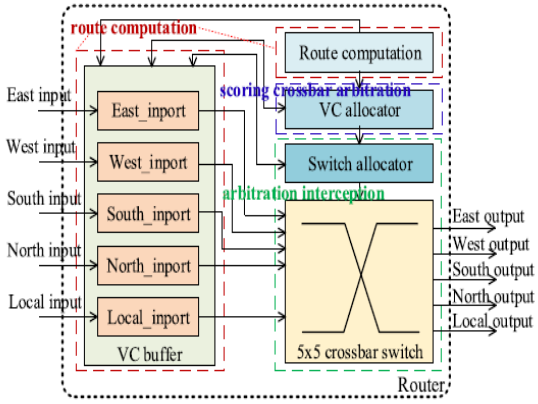

Fig. 4. *Structure of our router*

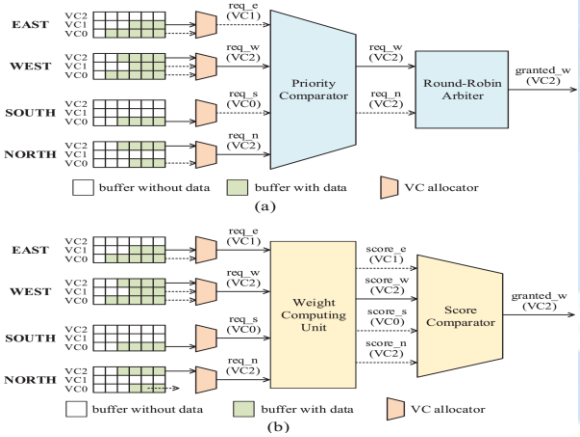

Fig. 5. *VC routing optimization I. (a) Conventional round-robin crossbar arbitration. (b) Our scoring crossbar arbitration.*

five output ports, VC buffers, a route computation unit, a VC allocator, a switch allocator, and a 5 × 5 crossbar switch. The dashed rectangle indicates that the component is with our optimization. Details about the VC routing optimizations are given in the following.

1) Scoring Crossbar Arbitration: Fig. 6(a) illustrates a conventional round-robin crossbar arbitration method. For each input port, if there exist packets of different priorities in the VCs, the highest priority one will be forwarded to the crossbar arbitration unit by the VC allocator. Then, packets from different input ports are granted by priority comparator and round-robin arbitration to access the routing resource. This multilevel arbitration method would introduce long latency and consume more hardware resources.

To reduce latency, we propose a scoring arbitration method as shown in Fig. 6(b). The main feature of our method

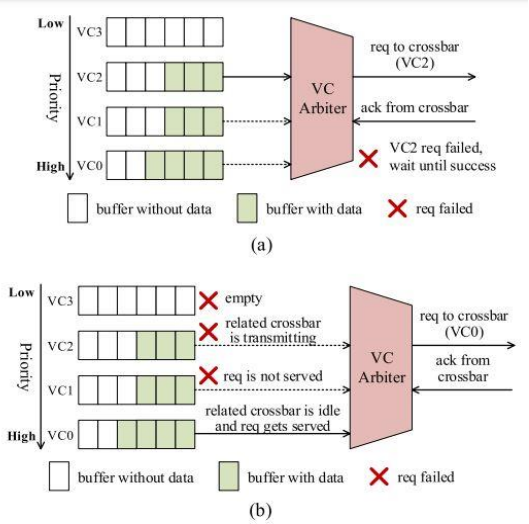

Fig. 7. *VC routing optimization II. (a) Conventional routing without arbitration interception. (b) Our proposed routing with arbitration interception.*

It is performing crossbar priority arbitration and round-robin arbitration in parallel with the scoring mechanism. The related score of the packets from each port is computed based on both their priority (cls) and the current round-robin factor (reslast) as weight, and the one gets the highest score will be granted and transmitted to the next router or IP core. The computation of the score of the packets is defined as

$$Sn = (cls * req) + (req - reslast + reqn)\%req \quad (2)$$

where Sn, cls, req, reslast, and reqn are the score of nth request input, the packet priority of the request input, the total number of arbitration, the last arbitration result, and the input port number of current request, respectively.

2) Arbitration Interception: As Fig. 7(a) depicts, packets from the same upstream node are stored in the related VCs (VC0–VC3) of the input port based on their priority. When multiple input ports apply to the crossbar arbitration of the same output port, ungranted packets are congested in the buffers until the next arbitration after the previously granted packets are

Multidisciplinary Journal for Applied Research in Engineering and Technology (MJARET)

Vol. 2 Issue 5, July – 2022, Pages: 60-67

DOI : https://doi.org/10.54228/mjaret07220013

ISSN : 2583-1100

transmitted to the next router or IP core. It is no doubt that it aggravates the congestion and increases the latency of the NoC.

Therefore, we propose an arbitration interception to ease the congestion. When the high-priority channel from the same input port fails to go through the crossbar arbitration, the crossbar will return an arbitration interception signal to the VC allocator. After receiving the signal, the VC allocator disables the request of the high-priority channel and grants the lower priority channel to apply to the crossbar arbitration, as Fig. 7(b) illustrates. This approach prevents the situation that lower priority packets have to wait in the VCs until higher priority packets get granted and transmitted to the

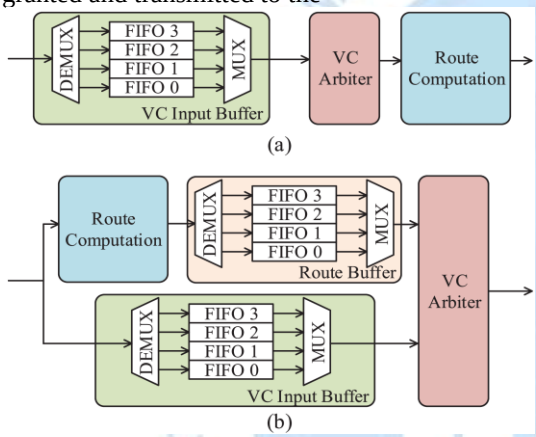

Fig. 7. *VC routing optimization III. (a) Conventional sequential route computation. (b) Our proposed parallel route computation.*

next router or IP core, which effectively eases transmission congestion and improves the performance of NoC.

3) Route Computation Parallelization: Fig. 8(a) shows the route computation process in a conventional router. The VC input buffering and route computation are processed in sequence. When the route computation involves a multicast lookup table, it may take a longer time to query the routing destination address in the lookup table. This would increase the delay of data packet transmission and greatly lower the performance of the on-chip network.

Fig. 8(b) shows our route computation. Unlike the process shown in Fig. 8(a), VC input buffering and route computation are processed in parallel. The packet header enters the VC FIFO unit and the route computation unit simultaneously. The route computation unit performs corresponding route calculation based on the destination address, packet type, and priority information contained in the packet header. In the route computation unit, we adopt the XY routing algorithm for unicast packets and lookup table-based routing algorithm for multicast packets. The routing algorithms are deterministic and effective in preventing deadlock. The body and tail of the packets only enter the VC FIFO unit.

5. Results and Discussion

Simulation was done in MODELSIM simulator with different input combination of NNA selection.

INPUT VALUES

The suggested crossbar arbitration with input waveform is shown in the image below. Different input signals will be provided in this section. In traditional round robin crossbar arbitration, the VC allocator will pass the highest priority packet to the crossbar arbitration unit for each input port if there are packets with differing priorities in the VCs. Then, a priority comparator and round-robin arbitration allow packets from various inputports access to the routing resource. Long delay would be introduced by using this multilayer arbitration mechanism, and more hardware resources would be used. Crossbar priority arbitration and round-robin arbitration are carried out concurrently with the scoring mechanism in the proposed arbitration in order to decrease latency. The related score of each port's packets is calculated based on both its priority and the weight of the current round-robin factor, and the packet with the highest related score will be allowed and sent to the next router or IP core.

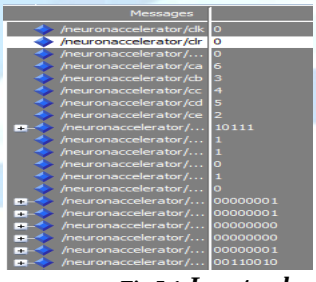

Fig 5.1 ***Input values for simulation***

SIMULATED OUPTUT

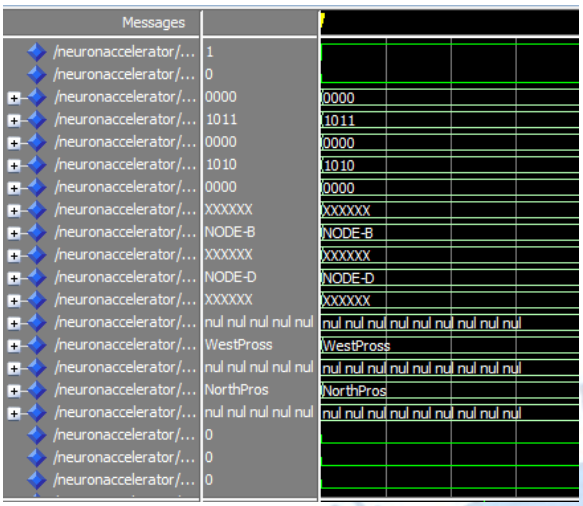

Fig 5.2 *Selected Nodes*

The figure 5.2 describes about the VC selection based on the input. Here Neuron Accelerator selects the North Pros and West Pros depending on the given inputs. In the same manner, the depends on the priority one gets the highest score will be granted and transmitted to the next router or IP core by the VC allocator.

POWER LEAKAGE

The proposed design consumes very minimum power for the crossbar arbitration design. The screenshot of the power consumption report is given below in Table 1.

Table 1. Total Power Leakage

On-Chip	Power (W)	Used	Available	Utilization (%)
Logic	0.000	5	2400	0
Signals	0.000	25	---	---
IOs	0.000	25	102	25
Leakage	0.011			
Total	0.011			

FLOOR PLAN

It is the floorplan ofinternal structure of Field programmable Grate Array Architecture. The screenshot of floor planner is given below in fig 5.3

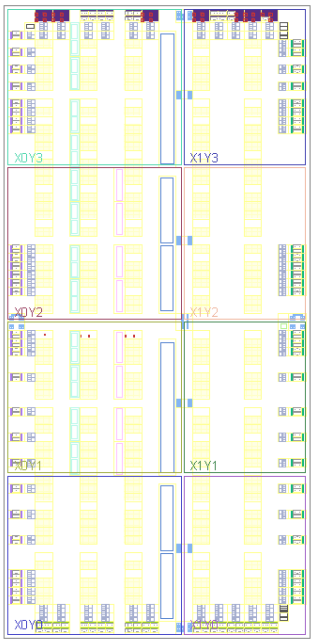

Fig 5.3 *Floor plan*

STATIC TIME ANALYSIS

By examining all potential avenues for timing violations, static timing analysis (STA) is a technique for evaluating the timing performance of a design. In STA, a design is divided into timing pathways, the signal propagation delay along each path is calculated, and timing restrictions within the design and at the input/output interface are checked for violations. By examining all potential avenues for timing infractions, static time analysis is a technique for evaluating the timing performance of a design.

The time analysis is mentioned in the below figure 5.4.

	Source Pad	Destination Pad	Delay
1	nodea<0>	East_Import	17.555
2	nodea<1>	East_Import	17.429
3	nodea<2>	East_Import	17.348
4	nodea<3>	East_Import	17.316
5	nodeb<0>	West_Import	19.129
6	nodeb<1>	West_Import	19.093
7	nodeb<2>	West_Import	18.982
8	nodeb<3>	West_Import	18.979
9	nodec<0>	South_Import	17.671
10	nodec<1>	South_Import	18.031
11	nodec<2>	South_Import	17.973
12	nodec<3>	South_Import	17.572
13	noded<0>	North_Import	18.391
14	noded<1>	North_Import	18.359
15	noded<2>	North_Import	18.301
16	noded<3>	North_Import	18.449
17	nodee<0>	Local_Import	18.127
18	nodee<1>	Local_Import	17.996

Fig 5.4 *Time analysis*

Multidisciplinary Journal for Applied Research in Engineering and Technology (MJARET)

Vol. 2 Issue 5, July – 2022, Pages: 60-67

DOI : https://doi.org/10.54228/mjaret07220013 **ISSN : 2583-1100**

RTL STRUCTURE OF NEURON LINK

Register Transfer Level (RTL) is an abstraction for defining the digital portions of a design. It is the principal abstraction used for defining electronic systems today and often serves as the golden model in the design and verification flow. RTL structure of neuron link screenshot is given below in fig 5.5

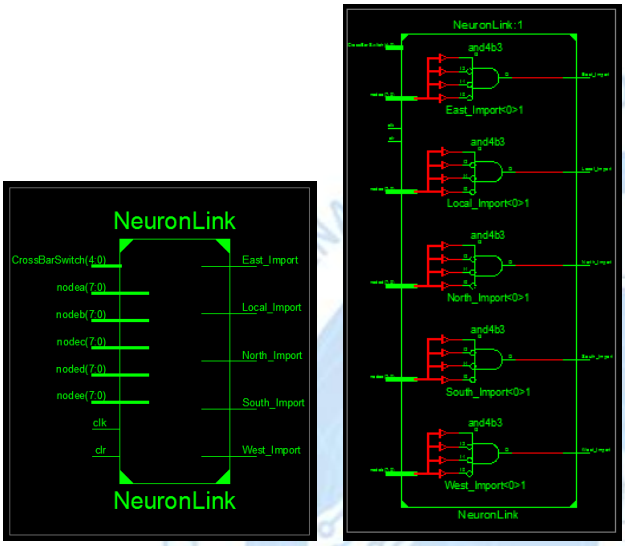

FIG 5.6 *Overall RTL structure and internal ucture of Neuron Link*

6. CONCLUSION

Xilinx's Model Sim Simulator was used to create and test the suggested strobe switch utilising various combinations of input values. Channel routing was carried out effectively with this crossbar arbitration, resulting in a high throughput NOC with reduced hardware costs for multicast-based traffic. This strobe switch offers a viable connection method, and in addition, the network configuration is often altered depending on the cost. The connection aims to manage the massive volume of multicast-based traffic within DNN accelerators effectively. The suggested connection, NeuronLink, may achieve high throughput and low bit error rate, according to the findings of experiments..

REFERENCES

[1] [1] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in Proc. Eur. Conf. Comput. Vis. (ECCV), Oct. 2016, pp. 630–645.

[2] [2] G. Hinton et al., "Deep neural networks for acoustic modeling in speech recognition: The shared views of four research groups," IEEE Signal Process. Mag., vol. 29, no. 6, pp. 82–97, Nov. 2012.

[3] [3] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierar- chies for accurate object detection and semantic segmentation," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit., Jun. 2014, pp. 580–587.

[4] [4] C. Zhang, P. Li, G. Sun, Y. Guan, B. Xiao, and J. Cong, "Optimizing FPGA-based accelerator design for deep convolutional neural networks," in Proc. ACM/SIGDA Int. Symp. Field-Program. Gate Arrays (FPGA), Feb. 2015, pp. 161–170.

[5] [5] S. Gupta, A. Agrawal, K. Gopalakrishnan, and P. Narayanan, "Deep learning with limited numerical precision," in Proc. Int. Conf. Mach. Learn. (ICML), Jul. 2015, pp. 1737–1746.

[6] [6] X. Zhou et al., "Cambricon-S: Addressing irregularity in sparse neural networks through a cooperative software/hardware approach," in Proc. 51st Annu. IEEE/ACM Int. Symp. Microarchitecture (MICRO), Oct. 2018, pp. 15–28. [7] S. Yin et al., "An energy-efficient reconfigurable processor for binary- and ternary-weight neural networks with flexible data bit width," IEEE J. Solid-State Circuits, vol. 54, no. 4, pp. 1120–1136, Apr. 2019.

[7] [8] P. Ou et al., "A 65 nm 39 GOPS/W 24-core processor with 11Tb/s/W packet-controlled circuit-switched double-layer network-on-chip and heterogeneous execution array," in IEEE Int. Solid-State Circuits Conf. (ISSCC) Dig. Tech. Papers, Feb. 2013, pp. 56–57.

[8] [9] M. Arulaalan, K. Mahendran, P. Prabakaran & G. Manikannan, "Design of MIMO Triangular Microstrip Patch Antenna for IEEE 802.11a Application", Inventive Communication and Computational Technologies pp 369–379.

[9] [10] A. Touzene, "On all-to-all broadcast in dense Gaussian network on- chip," IEEE Trans. Parallel Distrib. Syst., vol. 26, no. 4, pp. 1085–1095, Apr. 2015.

[10] [11] B. Bohnenstiehl et al., "KiloCore: A 32-nm 1000-processor computa- tional array," IEEE J. Solid-State Circuits, vol. 52, no. 4, pp. 891–902, Apr. 2017.

[11] [12] A. Firuzan, M. Modarressi, M. Daneshtalab, and M. Reshadi, "Recon- figurable network-on-chip for 3D neural network accelerators," in Proc. 12th IEEE/ACM Int. Symp. Netw.-Chip (NOCS), Oct. 2018, pp. 1–8.

[12] [13] H. Kwon, A. Samajdar, and T. Krishna, "MAERI: Enabling flexible dataflow mapping over DNN accelerators via reconfigurable intercon- nects," in Proc. 23rd Int. Conf. Archit. Support Program. Lang. Oper. Syst. (ASPLOS), Mar. 2018, pp. 461–475.

[13] [14] M. F. Reza and P. Ampadu, "Energy-efficient and high-performance NoC architecture and mapping solution for deep neural networks," in Proc. 13th IEEE/ACM Int. Symp. Netw.-Chip, Oct. 2019, pp. 1–8.

[14] [15] Y.-H. Chen, T.-J. Yang, J. S. Emer, and V. Sze, "Eyeriss v2: A flexible accelerator for

emerging deep neural networks on mobile devices," IEEE J. Emerg. Sel. Topics Circuits Syst., vol. 9, no. 2, pp. 292–308, Jun. 2019.

[15] [16] K. Bhardwaj and S. M. Nowick, "A continuous-time replication strategy for efficient multicast in asynchronous NoCs," IEEE Trans. Very Large Scale Integr. (VLSI) Syst., vol. 27, no. 2, pp. 350–363, Feb. 2019.

[16] [17] Y. Chen et al., "DaDianNao: A machine-learning supercomputer," in Proc. 47th Annu. IEEE/ACM Int. Symp. Microarchitecture, Dec. 2014, pp. 609–622.

[17] [18] G. Manikannan; P. Prabakaran; M. Selvaganapathy, "IoT enabled Smart Switch With user-friendly Electrical Interfacing", 2022 International Conference on Smart Technologies and Systems for Next Generation Computing (ICSTSN), April 2022

[18] [19] X. Liu, W. Wen, X. Qian, H. Li, and Y. Chen, "Neu-NoC: A high- efficient interconnection network for accelerated neuromorphic systems," in Proc. 23rd Asia South Pacific Design Automat. Conf. (ASP-DAC), Jan. 2018, pp. 141–146. [20] R. Hojabr, M. Modarressi, M. Daneshtalab, A. Yasoubi, and A. Khonsari, "Customizing clos network-on-chip for neural networks," IEEE Trans. Comput., vol. 66, no. 11, pp. 1865–1877, Nov. 2017.

[19] [21] C. Szegedy et al., "Going deeper with convolutions," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2015, pp. 1–9.

[20] [22] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2016, pp. 2818–2826.

[21] [23] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He, "Aggregated residual transformations for deep neural networks," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 5987–5995. Authorized licensed use limited to: FLORIDA INTERNATIONAL UNIVERSITY.

[22] [24] X. Zhang, Z. Li, C. C. Loy, and D. Lin, "PolyNet: A pursuit of structural diversity in very deep networks," in Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR), Jul. 2017, pp. 3900–3908.

[23] [25] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2018, pp. 7132–7141.

[24] [26] B. Zoph, V. Vasudevan, J. Shlens, and Q. V. Le, "Learning transferable architectures for scalable image recognition," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit., Jun. 2018, pp. 8697–8710.

[25] [27] E. D. Cubuk, B. Zoph, D. Mane, V. Vasudevan, and Q. V. Le, "AutoAug- ment: Learning augmentation strategies from data," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), Jun. 2019, pp. 113–123. [26] E. Real, A. Aggarwal, Y. Huang, and Q. V. Le, "Regularized evolution for image classifier architecture search," in Proc. AAAI Conf. Artif. Intell., vol. 33, 2019, pp. 4780–4789.

[26] [28] G. Manikannan ; K. Mahendran; P. Prabakaran, "Low Power High Speed Full Adder Cell with XOR/XNOR Logic Gates in 90nm Technology", 2017 International Conference on Technical Advancements in Computers and Communications (ICTACC), April 2017.

[27] [29] P. A. Merolla et al., "A million spiking-neuron integrated circuit with a scalable communication network and interface," Science, vol. 345, no. 6197, pp. 668–673, Aug. 2014.

[28] [30] W. Liao, Y. Guo, S. Xiao, and Z. Yu, "A low-cost and high-throughput NoC-aware chip-to-chip interconnection," in Proc. IEEE Int. Symp. Circuits Syst. (ISCAS), Oct. 2020.

[29] [29] F. Mireshghallah, M. Bakhshalipour, M. Sadrosadati, and H. Sarbazi-Azad, "Energy-efficient permanent fault tolerance in hard real-time systems," IEEE Trans. Comput., vol. 68, no. 10, pp. 1539–1545, Oct. 2019.

[30] [30] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in Proc. Adv. Neural Inf. Process. Syst. (NIPS), Dec. 2012, pp. 1097–1105.