A Novel Scalable On-chip Switch Architecture with Quality of Service Support for Hardware Accelerated Cloud Data Centers

Fatih Yazıcı[†], Ayhan Sefa Yıldız[*†], Alper Yazar[*†], Ece Güran Schmidt[†]

[†] Department of Electrical and Electronics Engineering, METU, Ankara, Turkey
{fatih.yazici, yildiz.ayhan, alper.yazar, eguran}@metu.edu.tr
[*] ASELSAN, Ankara, Turkey
{asyildiz, ayazar}@aselsan.com.tr

2020 IEEE 9th International Conference on Cloud Networking (CloudNet) | 978-1-7281-9486-8/20/$31.00 ©2020 IEEE | DOI: 10.1109/CLOUDNET51028.2020.9335788

***Abstract*—This paper proposes a scalable on-chip packet switch architecture, ACCLOUD-SWITCH, for hardware accelerated cloud data centers. The proposed switch architecture adopts architectural features from high-speed computer network and network on chip (NoC) routers. ACCLOUD-SWITCH interconnects heterogeneous high-speed interfaces and is implemented on FPGA. The switch fabric runs at line speed for scalability. We propose a new work-conserving fabric arbiter that can allocate bandwidth to input/output pairs by prioritizing the switch ports and a new hybrid buffer structure for ports connected to reconfigurable regions for more efficient memory use. The switch is implemented for Xilinx Zynq SoC device to work at 40 Gbps. Our simulation results demonstrate the benefits of the proposed arbiter and the hybrid buffer structure.**

***Index Terms*—hardware accelerated cloud data center, on-chip switch, switch fabric arbitration.**

I. Introduction

Hardware Accelerated Cloud Data Centers (HACDC) offer hardware accelerators (HA) as computing resources in addition to memory, processor (CPU) and disk [1], [2]. To this end, FPGA Accelerator Cards (FAC) can be installed in the servers or FACs with an SoC processor can be directly connected to the HACDC network. Multiple HAs can be instantiated on partially Reconfigurable Regions (RR) on the same FPGA [3]. Such organization requires high-speed data exchange among FAC components and *Quality of Service (QoS)* support for both satisfying the requirements of the applications and enabling bandwidth allocation for the Virtual Machines [4].

In this paper, we propose ACCLOUD-SWITCH (ACcelerated CLOUD Switch) to interconnect the hardware modules on the FAC including HAs, the SoC processor and 40Gbps Ethernet with the cloud server over PCIe. ACCLOUD-SWITCH architecture complies with both high-speed computer network and Network on Chip (NoC) routers. Different than previous work, the QoS support is achieved without a decrease in throughput with a novel work-conserving switch fabric arbiter that we call *CreditArbiteR* (CAR). Furthermore ACCLOUD-SWITCH features a new *hybrid buffer* organization specifically designed for the ports connected to RRs for efficient use of memory and a simple signal interface. ACCLOUD-SWITCH is *scalable* as it operates at line speed with a fully distributed architecture.

To the best of our knowledge there is no work in the literature that is directly comparable to ACCLOUD-SWITCH. Arbiters for computer network switches with performance goals of maximizing the throughput and improving the latency are proposed in [5]–[7]. [8], proposes bandwidth allocation to connections in the expense of not fully utilizing the available bandwidth. [9] suggests an on-chip router with a linked-list based buffer management with a special register structure to improve latency and throughput. A four-port switch implementation is evaluated with a soft simulator without a hardware implementation. [10] proposes switching between wormhole and virtual cut-through switching schemes at run-time. Fixed prioritization is applied to certain packet types. The implementation is on Virtex-7 device resulting in a data rate less than 4 Gbps. [11] dynamically manages buffer memory allocated to each port for better utilization. However, the packets are in a FIFO and it is difficult to compute the memory amount when packet boundaries are not observed. They assume fixed size packets of 16 Bytes. [12] proposes a memory access organization for efficient BRAM use. The architecture and the flow control are strictly for NoC. The implementation is on Zynq SoC with similar resource utilization to our implementation in Section II.

II. ACCLOUD-SWITCH

A. Architecture

We consider an FPGA Accelerator Card (FAC) with two 40 Gbps Ethernet interfaces. These interfaces are implemented as IP cores and interconnect the cloud server and the data center network [13]. There are four Reconfigurable Regions (RR) to implement hardware accelerators. There is an SoC processor and a PCIe interface. Accordingly, the switch is designed with $N = 8$ input/output lines with a line rate of 40 Gbps. ACCLOUD-SWITCH runs with 256 bit-payload *flits* to comply with the 40 Gbps Ethernet IP Core interface.

978-1-7281-9486-8/20/$31.00 ©2020 IEEE

Fig. 1 shows the block level design of ACCLOUD-SWITCH with the detailed architecture for input port i and output port j. Each input port i of ACCLOUD-SWITCH is connected to the module Mod_i and has a dedicated *Virtual Output Queue (VOQ)* VOQ_i_j for each output port j. If Mod_i is a HA, we propose the *hybrid buffer* (HB) architecture as shown in Fig. 1. The HB at input port i consists of the FCFS Receiver Buffer RecBuf_i between Mod_i and the VOQs. The HAs are expected to communicate with a specific destination module accordingly rather than uniformly distributed destinations. Hence, the Receiver Buffer mostly serves as an extension of the VOQ for the specific destination module resulting in a better memory utilization compared to the dedicated VOQ buffers as we demonstrate with our experimental evaluation in Section II-B. The HA is custom designed and can be interfaced with a single bit stop_i if RecBuf_i is full which is a more scalable solution than distinct signals that show the status of each VOQ_i_j. ACCLOUD-SWITCH inputs that are connected to system interfaces such as Ethernet feature pure VOQ buffers as they are more likely to communicate with many destination modules and custom interfacing with a single bit is not possible with standard IP cores such as Ethernet. Switch Fabric is a crossbar.

We implement our novel work-conserving switch fabric arbitration method that we call CreditArbiteR (CAR) and Dual Round Robin (DRR) [6] for the Arbiter as a comparison basis. CAR achieves *switching service differentiation* and *proportional bandwidth allocation* for inputs and outputs. CAR is a three step arbitration similar to [5]: 1) inputs set $Request(i, j) \leftarrow 1$ if VOQ_i_j is not empty (Ready[i][j]), 2) Arbiter selects a request and sets $Grant(i, j) \leftarrow 1$ (i can be granted more than once), 3) Arbiter selects a grant and marks i, j pair for the next data transfer over the fabric. Arbitration is repeated until no new connections can be made or $MaxIters$ is reached. CAR works in parallel for input/output ports compatible with the architecture in Fig 1. CAR achieves service differentiation by prioritization of the granted inputs and accepted outputs. We allocate each input i and output j $GrantFullCredit(i, j)$ and $AcceptFullCredit(i, j)$ *credits* respectively in accordance with the desired level of service for connection i, j from input and outputs perspective. We define an *arbitration round* for output j with $T_{OUT,j} = \sum_i GrantFullCredit(i, j)$ and for input i with $T_{IN,i} = \sum_j AcceptFullCredit(i, j)$ as cycles where all j's and i's get prioritized respectively. Let the line rate be C and desired bandwidth for connection (i, j) be $B_{in,i}(j)$ (for input) and $B_{out,j}(i)$ (for output). Credits should be selected using equations 1, 2 and 3.

CAR tracks the amount of service distribution received by input i and output j during the arbitration round by state variables $GrantCredit(i, j)$ and $AcceptCredit(i, j)$ which are re-initialized with full credits each arbitration round and decremented when (i, j) are connected. When credits are depleted, the respective $GrantPtr(j)$ or $AcceptPtr(i)$ is advanced to the next port to be prioritized. Ports with depleted credits are still connected if there are no alternative matches, thus maintaining the work conserving behaviour different than [8]. We randomly select the next port to prioritize when advancing the pointers for a proportionally fair distribution of credits of unmatched ports.

$$\sum_{j'} B_{in,i}(j') = C, \forall i \text{ and } \sum_{i'} B_{out,j}(i') = C, \forall j. \quad (1)$$

$$GrantFullCredit(i, j)/T_{OUT,j} = B_{out,j}(i)/C. \quad (2)$$

$$AcceptFullCredit(i, j)/T_{IN,i} = B_{in,i}(j)/C. \quad (3)$$

B. Implementation and Performance Evaluation

ACCLOUD-SWITCH is implemented for Xilinx XC7Z100 SoC running on single 156.25 MHz clock using Xilinx Vivado 2016.4. The selected SoC has available 277400 LUTs, 554800 FFs and 755 BRAMs (with 36 Kb capacity). CAR uses approximately 24k LUTs, 45k FFs and 224 BRAMs. DRR uses slightly less logic resources as expected at 20k LUTs and 43k FFs where BRAM usage is the same and consistent with the system architecture in Fig.1. The estimated power consumption is around 0.7 Watts for both implementations.

We developed an event-based cycle-accurate simulator in C++ language that models the implementation in Section II-A to measure the performance of the developed switch architecture and CAR fabric arbitration. We verify our simulator by repeating the DRR experiment in [6] that can be seen in Fig.2 (a). Next, we implement CAR in our simulator with an 8x8 switch with 40 Gbps line rate on all ports. We demonstrate the service differentiation capability of CAR with unlimited VOQs and without RecBuf_i under uniform destination ports. 1% of the packets are 40 Bytes and the remaining packets are 1500 Bytes as we assume that HAs will produce streams of data. We set; $B_{out,j}$=8, 6, 4 and 2 Gbps for ports 0-1, 2-3, 4-5 and 6-7. The respective CAR configuration is $\forall j$, $GrantFullCredit(i, j) = [8, 8, 6, 6, 4, 4, 2, 2]$ and $\forall i, j$, $AcceptFullCredit(i, j) = 2$. Fig.2 (b) shows the throughput of an output port j with CAR and DRR. Both arbiters work with $MaxIters = 3$ as we observe that CAR achieves convergence with an average number of iterations of 2.74 at 90% load. We observe that both CAR and DRR are work conserving: throughput is equal to offered load until 100%, and very close to 40 Gbps (39.7 Gbps) after. Next, we evaluate two service levels with $B_{out,j} = B_{in,i} = 2$ Gbps (low) and $B_{out,j} = B_{in,i} = 8$ Gbps (high) under a hot-spot destination scenario where all source ports contend to reach four destination ports. The simulation results in Fig.2 (c) show that CAR achieves proportional bandwidth allocation is achieved while.the matching efficiency of arbitration is affected from the non-uniform traffic for DRR. We evaluate the HB architecture with two RR ports contend to access two outputs. All RR (port 0-3) are assigned a service level $B_{in,i} = 8$ Gbps and the rest (port 4-7) $B_{in,i} = 2$ Gbps. Pure VOQ and HB architecture both have 640 flits of memory. The HB has 60 flits for each VOQ and 160 flits for the shared buffer in front. Fig.3 (a) shows the total throughput an input

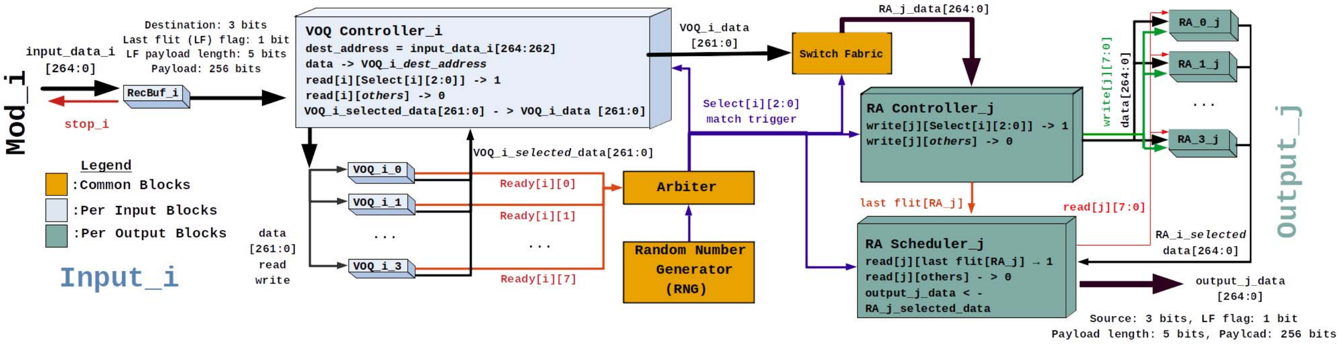
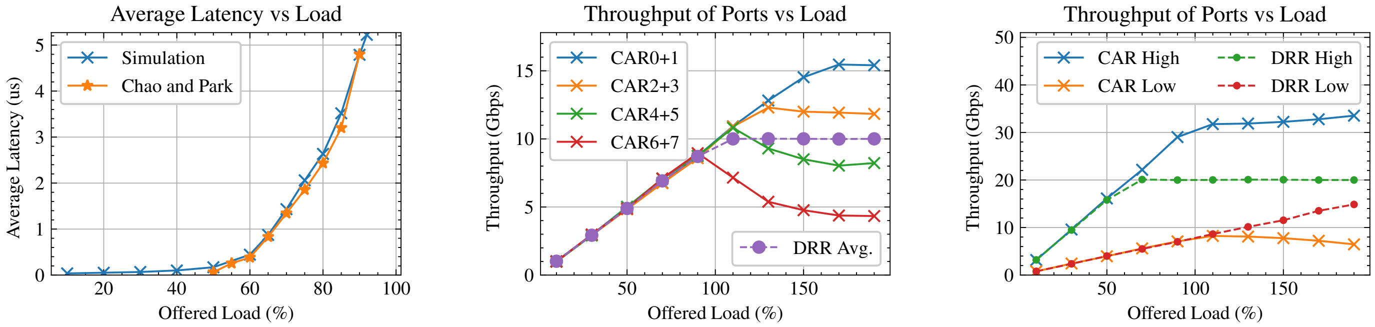


Fig. 1: ACCLOUD-SWITCH architecture.

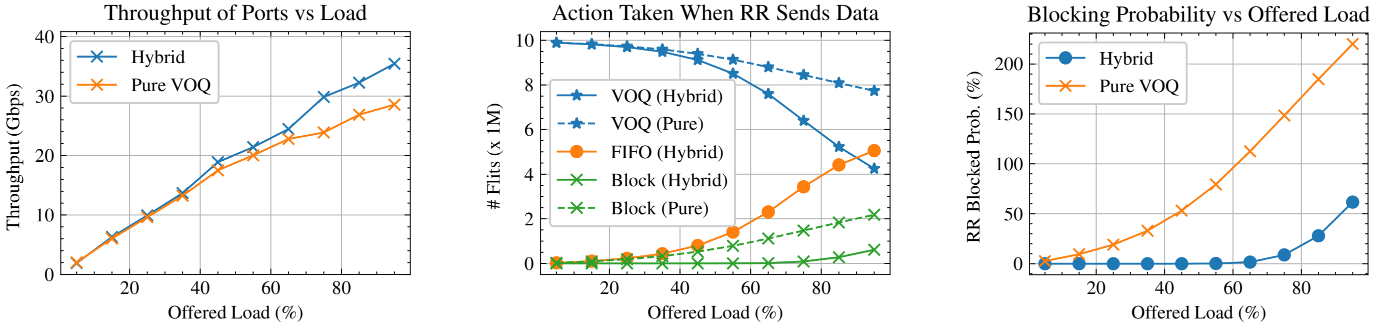


(a) Simulator verification (16x16, uniform tf.).

(b) CAR bandwidth allocation (uniform tf.).

(c) CAR bandwidth allocation (nonuniform tf.).

Fig. 2: Quality of service simulator experiments.



(a) Port throughput under nonuniform traffic.

(b) Distribution of received data action.

(c) Blocking probability of RR.

Fig. 3: Hybrid architecture simulator experiments.

port achieves under this imbalanced loading scenario. Under low load, Pure VOQ and HB both stay in the allocated VOQ memory for queuing. After 60% throughput, 80 flits of VOQ memory of the Pure VOQ model are filled mostly and RRs are blocked more frequently whereas the HB model continues using the shared buffer region as an extension to its VOQs. At 95% load, (sent $40 \cdot 0.95 = 38$ Gbps), the HB model reaches 35.4 Gbps whereas the Pure VOQ model tops at 28.5 Gbps, demonstrating the advantage of HB under expected traffic patterns. Fig.3 (b) shows the buffering actions taken. Under Pure VOQ, at 65% load, 89% of the data are accepted to VOQs and 11% of the incoming data are blocked. In HB, 76.5% of the data are accepted to VOQs, 23.3% are redirected to the shared buffer and only 0.2% is blocked. Fig.3 (c) illustrates that HB significantly decreases blocking probability.

III. Conclusion

This paper proposes ACCLOUD-SWITCH, an on-chip switch architecture to interconnect the modules on the FPGA Accelerator Card including Hardware Accelerators and 40 Gbps Ethernet interfaces. Our experiments show that ACCLOUD-SWITCH achieves 40 Gbps throughput with line speed operation while offering QoS thanks to our novel fabric arbiter and hybrid input buffer organization. The switch is implemented on the Zynq-7000 SoC XC7Z100 at 40 Gbps line rate to demonstrate the resource use. As the next step of our research, the switch will be employed in an laboratory test bed with cloud servers and tested under real traffic.

Acknowledgment

This work was supported by the Scientific and Research Council of Turkey (TUBITAK) [Project Code 117E667-117E668]. The authors would like to thank ASELSAN for their support.

References

[1] "Amazon ec2 f1 instances," accessed: 2020-08-20. [Online]. Available: https://aws.amazon.com/ec2/instance-types/f1/

[2] A. M. Caulfield, E. S. Chung, A. Putnam, H. Angepat, J. Fowers, M. Haselman, S. Heil, M. Humphrey, P. Kaur, J.-Y. Kim *et al.*, "A cloud-scale acceleration architecture," in *The 49th Annual IEEE/ACM International Symposium on Microarchitecture*. IEEE Press, 2016, p. 7.

[3] A. Yazar, A. Erol, and E. G. Schmidt, "Accloud (accelerated cloud): A novel fpga-accelerated cloud archictecture," in *2018 26th Signal Processing and Communications Applications Conference (SIU)*. IEEE, 2018, pp. 1–4.

[4] N. U. Ekici, K. W. Schmidt, A. Yazar, and E. G. Schmidt, "Resource allocation for minimized power consumption in hardware accelerated clouds," in *2019 28th International Conference on Computer Communication and Networks (ICCCN)*. IEEE, 2019, pp. 1–8.

[5] N. McKeown, "The islip scheduling algorithm for input-queued switches," *IEEE/ACM transactions on networking*, no. 2, pp. 188–201, 1999.

[6] H. J. Chao and J.-S. Park, "Centralized contention resolution schemes for a large-capacity optical atm switch," in *1998 IEEE ATM Workshop Proceedings.'Meeting the Challenges of Deploying the Global Broadband Network Infrastructure'*. IEEE, 1998, pp. 11–16.

[7] B. Hu, F. Fan, K. L. Yeung, and S. Jamin, "Highest rank first: A new class of single-iteration scheduling algorithms for input-queued switches," *IEEE Access*, vol. 6, pp. 11 046–11 062, 2018.

[8] D. Stiliadis and A. Varma, "Providing bandwidth guarantees in an input-buffered crossbar switch," in *Proceedings of INFOCOM'95*, vol. 3. IEEE, 1995, pp. 960–968.

[9] C. Li, D. Dong, Z. Lu, and X. Liao, "Rob-router: A reorder buffer enabled low latency network-on-chip router," *IEEE Transactions on Parallel and Distributed Systems*, vol. 29, no. 9, pp. 2090–2104, 2018.

[10] H. K. Nguyen and X.-T. Tran, "A novel reconfigurable router for qos guarantees in real-time noc-based mpsocs," *Journal of Systems Architecture*, vol. 100, p. 101664, 2019.

[11] A. Mirhosseini, M. Sadrosadati, F. Aghamohammadi, M. Modarressi, and H. Sarbazi-Azad, "Baran: Bimodal adaptive reconfigurable-allocator network-on-chip," *ACM Transactions on Parallel Computing (TOPC)*, vol. 5, no. 3, pp. 1–29, 2019.

[12] K. Parane and B. Talawar, "Lbnoc: Design of low-latency router architecture with lookahead bypass for network-on-chip using fpga," *ACM Transactions on Design Automation of Electronic Systems (TODAES)*, vol. 25, no. 1, pp. 1–26, 2020.

[13] "40Gbps Ethernet solution." [Online]. Available: http://hiteksys.com/pdf/40G-Ethernet-Verification-Report.pdf