Modified CONNECT: New Bufferless Router for NoC-Based FPGAs

Marwa Shaheen[a], Hossam Fahmy[a], and Hassan Mostafa[a, b]

[a]Electronics and Communication Engineering Department, Cairo Univeristy
[b]Nanotechnology and Nanoelectronics Program, Zewail City for Science and Technology

***Abstract*—As Chip Multiprocessors (CMPs) scale to tens or hundreds of nodes, the interconnect becomes a significant factor in cost, energy consumption and performance. Energy efficiency of the underlying communication framework plays a major role in the performance of multi-core systems. Recent work proposes buffer-less deflection routing as a cost effective alternative. A modified version of CONfigurable NETwork Creation Tool(CONNECT) is proposed as a buffer-less router to be lightweight and efficient. Modified CONNECT is based on parallel port allocation mechanism within the routers, that allows maximum number of flits to be deflected in a productive direction. Modified CONNECT is evaluated using uniform, transpose and inverse traffic workloads against CONNECT, BLESS and Cheap- Interconnect Partially Permuting Router (CHIPPER) on a 4x4 mesh network. Modified CONNECT saves 30% area compared to CONNECT while employing lower performance. Modified CONNECT also saves 24% in area against BLESS keeping the same performance.**

I. INTRODUCTION

Network on chip offers component re-use and partial configuration, which opens a new way to dynamic multitasking applications [19]. To solve the problem of frequency requirements for different peripherals; network on chip becomes a parallel, concurrent and scalable alternative to traditional buses. Programmed FPGA is used to provide flexibility and fast prototyping implementation, and therefore it is used in many network on chip designs [16] and [18].

However, Networks On Chips(NoC) occupy large area. Buffer-less routers are proposed to minimize the area and the cost of NoC. Removing buffers yields to a simpler and more energy-efficient NoC; e.g., CHIPPER [5] reduces average network power by 54.9% in a 64-node system compared to a conventional buffered router and reduces router area by 36.2% relative to buffered routing. Although buffer-less architecture seems promising, it faces long critical path because of the complex sequential port allocator e.g., BLESS[4]. The lightweight designs and the simple allocators even suffer from near saturation point e.g.; CHIPPER[5] saturates at injection rate 30%. Buffer-less design performs well at low and medium traffic loads. At high traffic loads, buffer-less designs suffer from performance degradation.

In this work, Modified CONNECT is presented; a buffer-less router based on the architecture of CONfigurable NETwork Creation Tool (CONNECT)[1]. The goal is to obtain a lightweight router. The key insight is: Support a simple port allocator that allows the maximum number of flits to be directed productively. It guarantees that the maximum number of flits are granted to their requested output ports. Modified CONNECT increases the number of flits directed productive direction to their destination. Modified CONNECT is evaluated against CONNECT, and against buffer-less routers such as BLESS and Cheap-Interconnect Partially Permuting Router (CHIPPER). It occupies less area than both pure buffer-less routers and achieves a competitive performance; while achieving less area than CONNECT and lower performance at high traffic loads. The paper is organized as follows. First, section II-A is a brief discussion on previous buffer-less architectures. Section II-B presents the architecture of CONNECT.

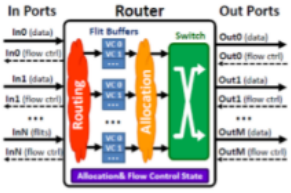


Fig. 1: CONNECT Router

Then, section III presents the architecture of MODIFIED CONNECT. In section IV discusses the simulations and the pros of the proposed router. Finally, MODIFIED CONNECT is evaluated in section V.

II. BACKGROUND

A. Buffer-less NoC Routing

Buffer-less routers eliminate in-network buffers. Flits must be directed to an output port at the end of the router pipeline. Incoming flits are kept in pipeline registers till output ports are allocated. When two flits request the same output port, the allocator either drops one of them or deflects it. This work discusses deflection routing. When flits enter the router pipeline, they are sorted. Flits of lower priority are deflected at contentions. A flit is deflected when the assigned output port is not the requested one, therefore it is deflected away from its route to destination. The deflected flits eventually reach the destination by a livelock prevention algorithm. BLESS[4] uses a sequential port allocator. To give a deterministic bound on network latency, a time stamp is added to each flit. Flits at the allocator are sorted in order of priority. When two flits request the same output port, the oldest is prioritized. The oldest flit gets its requested output port, the other flit is deflected to any unallocated output port. Livelock freedom is guaranteed because the oldest flit is always chosen to make a productive hop towards the destination. However, this router has a long allocation critical path because of the used priority scheme.

CHIPPER[5] proposes a lightweight two stage router. CHIPPER does not use a full sorting algorithm to the incoming flits. It determines the highest priority flit only. Only a single flit is guaranteed to be directed to its productive output port. CHIPPER proposes an efficient simple parallel switch. CHIPPER simplifies the allocation problem into 2x2 arbiter block. Livelock problem is solved by a golden flit; every epoch a single flit is prioritized over all the other flits roaming in the network. The golden flit is prioritized for a period long enough to guarantee its delivery to its destination. However, this router has traded performance with low area. CHIPPER's switch is used later in many designs as in Minimally-Buffered Deflection Routing for Energy-Efficient Interconnect (MINBD). MINBD[6] is a two stage buffer-less router based on CHIPPER's switch. It uses a

978-1-5386-7392-8/18/$31.00 ©2018 IEEE

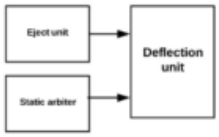
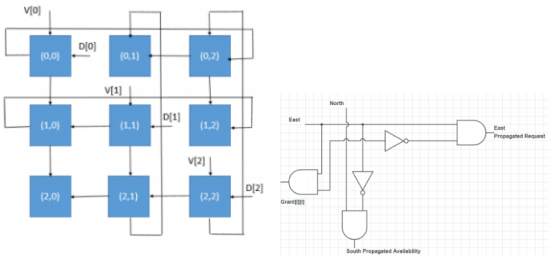


Fig. 2: Proposed Allocation Logic

side buffer to reduce deflections and the dynamic power consumed in the deflections. At contentions the router performs deflection routing, but it can choose to buffer up to one flit per cycle in a small side buffer. MINBD has a better improve in performance than CHIPPER with small area overhead. DEBAR[7] is a buffer-less router, DEBAR offers dual ejection units and dual injection units. To solve livelock problem, DEBAR sorts the incoming flits so that flits closer to destination are given higher priority. DEBAR uses the switch proposed in CHIPPER. DEBAR has higher performance than both CHIPPER and MINBD. SLIDER [11] is a dual stage buffer-less architecture that uses CHIPPER's switch and side buffer. SLIDER proposes late injection where the injection of the flits from the node or the side buffer, is kept at the end of the pipeline to reduce channel wastage. Recently, SCEPTER[10] proposes a buffer-less router that offers bypassing of flits. It allows flits to arbitrate for multiple routers in a single cycle. Also it offers a new throttling algorithm. Finally, Carpool [12] merges flits destined to the same port.

B. CONNECT Router

CONNECT is an open source RTL network on chip designed for FPGAs. The router has a single stage pipeline as shown in Figure1. Packets are divided into flits. The basic unit of routing in the network is the flit. Each flit has the routing information and can travel independently. Routing information is the flit's destination. As the flits enter the router pipeline, they access routing unit "look-up tables" to get the proper output port. Then the flit and the header indicating the appropriate output port, both are buffered in a FIFO corresponding to the input port of the flit and its virtual channel. CONNECT implements flit buffers using Distributed RAM, where each FIFO represents a virtual channel. Each virtual channel is implemented efficiently as a circular FIFO. Multiple VCs increase the performance by reducing the effects of head of line blocking. To determine which flit is leaving the buffer to its output port, Flit header, buffer occupancy of the corresponding input buffer and credit availability are forwarded to the allocation logic to arbitrate for the proper output port. CONNECT can use credit based flow control or peek flow control. CONNECT uses two channels, one for data and the other for control signals used in the flow control. CONNECT supports four allocators separable input first static allocator, separable output first static allocator, separable input first round robin allocator and separable output first round robin allocator. The allocator chooses flits to travel to the output ports respecting the priorities of the input ports and VCs, according to the priority scheme used. The output of the allocator is fed to the encoder. If a flit gets a grant from the allocator to traverse the switch to output ports, the encoder allows the flit's corresponding input buffer to move the header pointer to the next flit. Finally the flit traverses a multiplexer to the output port. CONNECT architecture and the effect of varying the number of virtual channels (VCs), flit data width and buffer depth on the operating frequency, Logic Look-Up Tables (LUTs) and registers were briefly discussed in [13] and [17]. CONNECT router is also used in simulations in [14].



(a) Deflection Unit (b) Single Element

Fig. 3: Deflection Unit

III. PROPOSED BUFFER-LESS ROUTER ARCHITECTURE

This is a single stage pipeline router. During each clock cycle at each input port; the router senses whether there is a valid flit and buffers it in the corresponding pipeline register. At the end of each cycle, every flit should be delivered, despite contentions.

A. Routing

Input flit fetches distributed RAM using its destination header to get the appropriate output port. The requested output port is saved in the pipeline register corresponding to its input port. Also the flit is saved in the pipeline register. Switch enables the pipeline register to receive new data every new cycle. The allocator input is only the output port request for each input flit.

B. Allocation

The presented allocator achieves parallelism, mutual exclusion and high performance. The allocator is divided into two sequential stages as shown in Figure 2. The first includes the eject unit and static output arbiter, the second is the deflection unit.

1) Eject Unit: Flits destined to this local node are ejected. The router can eject only one flit. To solve the livelock problem, eject unit uses round robin to give priority to input ports. The flit with the higher priority is ejected, and others are deflected at the Deflection unit.

2) Static Output Arbiter: This allows the maximum number of flits to be directed to their productive direction in a parallel way. Static Output Arbiter uses a simple fixed priority order of input flits for each output port. Using this priority scheme and round robin at the eject unit guarantees that there is no livelock problem. Eventually, all packets are sent to destination. This static output arbiter is an initial port allocator and is used so that flits of the highest priority are assigned their requested output ports. Flits that are not granted any output ports are directed to the second stage.

C. Deflection Unit

The inputs to this stage are flits of lower priority that should be deflected, and the available output ports. The deflection unit directs deflected flits to the available free ports in an efficient parallelized way. Flags representing the available output ports propagate vertically until they meet an active request. The requests travel horizontally until they meet an available output. The deflected flit is granted an output port when the its request travelling horizontally meets an available output port. The deflection requests are maintained on the diagonal to keep the propagation delay of the deflection requests symmetric. Also, it helps that the deflection requests are granted output ports in parallelized way illustrated in Figure3.

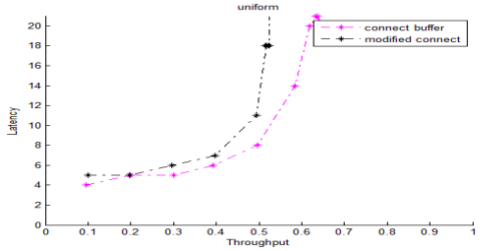


Fig. 4: Load Latency Curve

D. Inject Unit

Local node injects flit whenever one channel is free. Local node injects flits whenever there is an available channel "an empty slot" whether it is in a productive direction or not. Local node injection takes place after the main input ports' flits are assigned output ports.

E. Switch

This is the same crossbar switch used by CONNECT. Only the header bits "control bits" travel through the allocation stage and the switch stage; data payload is buffered at the pipeline register after the routing stage and does not travel down the router units. At the end, switch outputs are forwarded to five multiplexers which direct the data payloads "from pipeline registers" to the appropriate output ports that were granted at the arbitration stage.

IV. SIMULATION RESULTS AND DISCUSSIONS

FPGAs are generally slower than ASIC. Hence, most FPGA router designs are pipelined to improve the frequency. Most of the current buffer-less designs are multi-stage routers. Modified CONNECT is a single stage router. "The shallow pipeline in the CONNECT NoC architecture has the added benefit of reducing network latency as well as greatly reducing the number of precious flip-flops consumed by a router"[1]. This is the main factor that serves our design to be a lightweight router. The design performance is better at wide data path. As wide data paths improve the operating frequency of NoCs on FPGA[1]. It is noticed that implementing the pipeline register at each input port as register is more efficient than implementing it using distributed RAM, for wide data paths. Parallel port allocation improves the performance as it allows the maximum number of flits to be deflected to their productive output ports. Also it prevents unnecessary deflections, where flits take shorter time to destination. Reducing deflection rate decreases the dynamic power dissipated as shown in TableI. Generally, buffer-less routers have performance close to conventional buffered routers at low traffic loads, however the performance degrades at high traffic loads. The design has a competitive performance at high traffic loads as in Figure 4. Load latency curves of CONNECT were briefly discussed in [15]. To decrease the average latency of flits in the network and increase the throughput, flits are injected from the local node after all flits from the four input ports are granted output ports.

V. COMPARISON WITH PREVIOUSLY PUBLISHED ROUTERS

Modified CONNECT is evaluated against two metrics: performance and hardware cost (network power and area). It is compared against buffer-less router BLESS [4], buffer-less CHIPPER [5] and buffered router CONNECT in a 4x4 mesh network. The generated buffered CONNECT [2] uses separable input first static allocator, with 8 slot buffer and credit-based flow control.

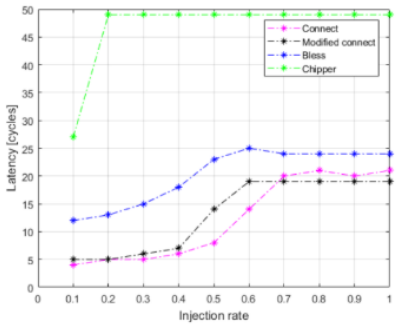


Fig. 5: Average Network Latency vs. Injection rate

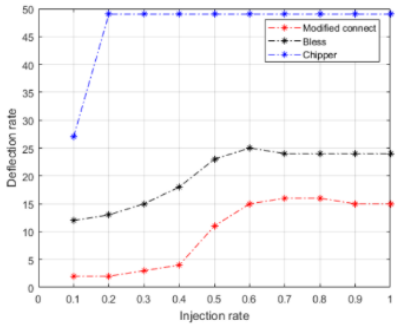


Fig. 6: Deflection rate for Buffer-less Designs

1) Performance Analysis: The design is evaluated using synthesis uniform traffic pattern, inverse traffic pattern and transpose traffic pattern. Modified Connect has lower throughput than CONNECT, the same throughput as BLESS and higher than CHIPPER as illustrated in Figure7. Modified Connect has lower average latency than BLESS as illustrated in Figure 5.

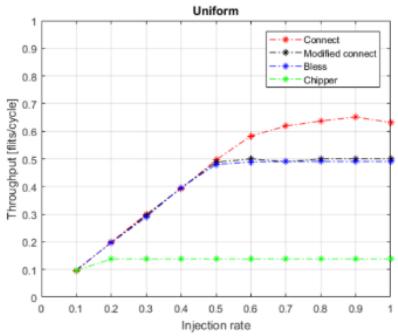
2) Hardware Analysis: Modified CONNECT is implemented in Verilog. Modified CONNECT, the open source model of CONNECT [1] and the available buffer-less open source routers BLESS[4] and CHIPPER [5]; the four designs are synthesized on Virtex-7, evaluation board VC707. The area utilization of the whole 4x4 network for each of the mentioned routers is illustrated in TableI. The estimated FPGA resource area in [3] is used to calculate the approximate gain in area utilization. Modified Connect has 30% area less than CONNECT. Modified CONNECT reduces area compared to BLESS by 24% and reduces the area compared to CHIPPER by 18%. The reduction in area of Modified CONNECT is due to the simple pipeline used in Modified CONNECT. Power analysis is illustrated in TableI. Dynamic power is calculated at maximum injection rate.

VI. CONCLUSION

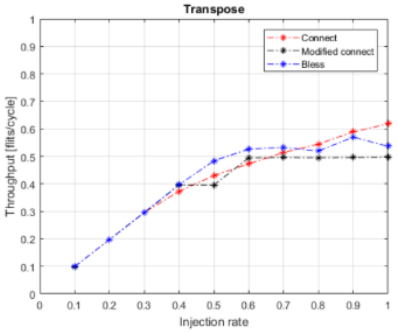
In this work, a new lightweight buffer-less router for NoCs on FPGAs is presented based on the architecture of CONNECT. Modified CONNECT replaces the allocator used in CONNECT with a parallel port allocation logic. Modified CONNECT achieves competitive

Router	LUT	register	Dynamic Power
Connect	14211	2624	0.003 (W)
Modified-Connect	8850	4144	0.035 (W)
BLESS	24814	33584	0.34 (W)
CHIPPER	20852	26784	0.075 (W)

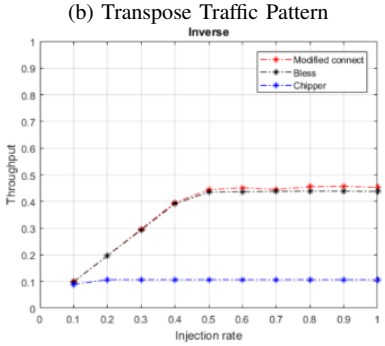
TABLE I: Area and Power Analysis



(a) Uniform Traffic Pattern



(b) Transpose Traffic Pattern



(c) Inverse Traffic Pattern

Fig. 7: Traffic Evaluations for Network-Level Throughput

performance to other buffer-less designs, with minimal cost, where it achieves 30% reduction in area compared to CONNECT, 24% compared to BLESS and 18% compared to CHIPPER.

VII. ACKNOWLEDGEMENT

This research was partially funded by ONE Lab at Cairo University, Zewail City of Science and Technology, and KAUST.

REFERENCES

[1] M. K. Papamichael and J. C. Hoe, "CONNECT: Re-examining conventional wisdom for designing NoCs in the context of FPGAs", in FPGA, pp. 37-46, 2012.
[2] M. K. Papamichael and J. C. Hoe, http://users.ece.cmu.edu/~mpapamic/connect/
[3] N. Gamal, and H. Mostafa, "Design guidelines for embeded NoCs on FPGA", in ISQED, pp. 69-74, 2012.
[4] T. Moscibroda and O. Mutlu, "A case for buffer-less routing in on-chip networks", in ISCA, pp. 196-207, 2009.
[5] C. Fallin et al, "CHIPPER: A low complexity buffer-less deflection router", in HPCA, pp. 144-155, 2011.
[6] C. Fallin et al," MinBD: Minimally-buffered deflection routing for energy- efficient interconnect", in NOCS, pp. 1-10, 2012.
[7] J. Jose et al., "DeBAR: Deflection based adaptive router with minimal buffering", in DATE, pp. 1583-1588, 2013.
[8] M. Hayenga et al., "SCARAB: A single cycle adaptive routing and buffer-less network", in MICRO,pp. 244-254, 2009.
[9] N. Kapre and J. Gray, "Hoplite: building austere overlay NoCs for FPGAs", in FPL, pp. 1-8, 2015.
[10] B. K. Daya, L.-S. Peh, and A. P. Chandrakasan, "Quest for high-performance bufferless nocs with single-cycle express paths and self-learning throttling", in DAC, pp. 36:1-36:6, 2016.
[11] B. Nayak et al., "Slider: Smart late injection deflection router for mesh NoCs", in ICCD, pp. 377-383, 2013.
[12] X. Xiang, W. Shi, S. Ghose, L. Peng, O. Mutlu, and N.-F. Tzeng, "Carpool: A buffer-less on-chip network supporting adaptive multicast and hotspot alleviation", in ICS, pp. 1-11, 2017.
[13] A. Salaheldin, K. Abdallah, N. Gamal, and H. Mostafa, "Review of noc-based FPGAs architectures", in ICEAC, pp. 1-4, 2015.
[14] A. Salaheldin, H. Mostafa, and A. M. Soliman, "A CODEC, tiles to NoC router interface, for next generation FPGAs with embedded NoCs", in MWSCAS, pp. 1228-1231, 2017.
[15] K. Helal, S. Attia, T. Ismail, and H. Mostafa, "Comparative review of NOCs in the context of ASICs and FPGAs", in ISCAS, pp. 1866-1869, 2015.
[16] M. Beheiry, H. Mostafa, Y. Ismail, and A. M. Soliman, "3D-NOCET: a tool for implementing 3D-noCs based on the direct-elevator algorithm", in ISQED, pp. 144-148, 2017.
[17] N. Gamal, H. A. H. Fahmy, Y. Ismail, T. Ismail, M. Mohie-Eldin, and H. Mostafa, "Design guidelines for soft implementations to embedded NoCs of fPGAs", in IDT, pp. 37-42, 2016.
[18] M. Beheiry, A. Aly, H. Mostafa, and A. M. Soliman, "Direct-elevator: a modified routing algorithm for 3D-noCs",in ICM 2015, pp. 1-4, 2015.
[19] A. Hassan, H. A. H. Fahmy, Y. Ismail, and H. Mostafa, "Exploiting the dynamic partial reconfiguration on noc-based fpga", in NGCAS, pp. 277-280, 2017.