CONNECT: Fast Flexible FPGA-Tuned Networks-on-Chip

Michael K. Papamichael
Computer Science Department
Carnegie Mellon University
Pittsburgh, PA, USA
<papamix@cs.cmu.edu>

James C. Hoe
Electrical & Computer Engineering Department
Carnegie Mellon University
Pittsburgh, PA, USA
<jhoe@ece.cmu.edu>

ABSTRACT

In this paper we present CONNECT, a flexible NoC architecture and RTL generation engine for fast, FPGA-tuned Networks-on-Chip (NoCs). The CONNECT NoC architecture embodies a set of FPGA-motivated design principles that uniquely influence key NoC design decisions, such as topology, link width, router pipeline depth, network buffer sizing, and flow control. The flexibility, lightweight nature and high performance of CONNECT-based NoCs makes them ideal candidates for use in FPGA-based research studies. We evaluate CONNECT against a high-quality publicly available synthesizable RTL-level NoC design intended for ASICs. Our results show a significant gain in specializing NoC design decisions to FPGAs' unique mapping and operating characteristics. For example, in the case of a 4x4 mesh configuration evaluated using a set of synthetic traffic patterns, we obtain comparable or better performance than the state-of-the-art NoC while reducing logic resource cost by 58%, or alternatively, achieve 3-4x better performance for approximately the same logic resource usage. To demonstrate CONNECT's flexibility and extensive design space coverage, we also report synthesis and network performance results for a variety of different CONNECT networks. This paper is based on [13] previously published at FPGA 2012.

1. INTRODUCTION

The rapidly-growing capacity of Field Programmable Gate Arrays (FPGAs), combined with the steady introduction of hardwired support for a multitude of diverse interfaces and functionalities, has promoted FPGAs to an attractive and capable platform for architectural research and extended System-on-Chip (SoC) prototyping and implementation. As the scale of designs targeting FPGAs grows, designers need a systematic and flexible Network-on-Chip (NoC) infrastructure to support communication between the tens and in the future potentially hundreds of interacting modules.

In this paper, we present the CONNECT NoC design generator that can produce synthesizable RTL-level designs of multi-node NoCs based on a simple but flexible fully-parameterized router architecture specifically designed and tuned for use in FPGA-based environments. The CONNECT NoC architecture embodies a set of FPGA-motivated design principles that uniquely influence key NoC design decisions, such as topology, link width, router pipeline depth, network buffer sizing, and flow control.

The rest of this paper is organized as follows. Section 2 introduces the motivations behind the CONNECT NoC architecture, and Section 3 presents the architecture of CONNECT-based routers and NoCs. In Section 4 we evaluate a CONNECT-based 4x4 mesh network against an equivalent NoC implemented using publicly available high-quality state-of-the-art RTL and show FPGA synthesis and network performance results for various CONNECT NoC configurations. Finally, we examine related work in Section 5, present our recently released online NoC generation tool in Section 6 and conclude in Section 7. Readers not familiar with basic NoC terminology and concepts are encouraged to read the background section in our related FPGA 2012 paper [13].

2. TAILORING TO FPGAS

Compared to ASICs, an FPGA is a peculiar hardware realization substrate because it dictates a very different set of design tradeoffs between logic, wires, memory and clock frequency. In this work, we take full consideration of FPGAs' special hardware mapping and operating characteristics to identify their own specialized NoC design sweet spot, which we will show is very different from the conventional wisdom stemming from NoC designs on ASICs.

Specifically, we focus on these FPGA characteristics that influence fundamental CONNECT NOC design decisions: (1) the relative abundance of wires compared to logic and memory; (2) the scarcity of on-die storage resources in the form of a large number of modest-sized buffers; (3) the rapidly diminishing return on performance from deep pipelining; and (4) the field reconfigurability that allows for an extreme degree of application-specific fine-tuning.

Abundance of Wires. As previously also noted by other work [10], FPGAs are provisioned, even over-provisioned, with a highly and densely connected wiring substrate. As a result wires are plentiful, or even "free", especially relative to the availability of other resources like configurable logic blocks and on-chip storage. In CONNECT we make the datapaths and channels between routers as wide as possible to consume the largest possible fraction of the available (otherwise unused) wires. In addition, we also adapt the format of network packets; information that would otherwise be carried in a separate header flit is carried through additional dedicated control wires that run along the data wires. Finally, we also adapt flow control mechanisms to occupy fewer storage resources by using wider interfaces (please see our related FPGA 2012 paper [13] for more information).

Storage Shortage. Modern FPGAs provide storage in two forms: (1) Block RAMs with tens of kilo-bits of capacity, and (2) small tens-of-bits Distributed RAMs built using logic Look-Up Tables (LUTs). Both of these monolithic memory macros can not be subdivided, which can lead to inefficiencies. This sets up a situation where NoCs on FPGAs pay a disproportionately high premium for storage because NoCs typically require a large number of buffers whose capacities are each much bigger than Distributed RAMs but much smaller than Block RAMs. To make the most efficient use of storage resources, CONNECT only uses Distributed RAM and implements multiple logical flit buffers in each physically allocated buffer on the FPGA. CONNECT does not use any Block RAMs, which are typically in high demand from the rest of the FPGA-resident user logic.

Low Clock Frequency. FPGA designs tend to operate at significantly lower clock frequencies compared to ASIC designs, which was one of the gaps studied in [9]. This frequency gap can be attributed to the use of LUTs and

long interconnect wires and results in rapidly diminishing returns when attempting to deeply pipeline a FPGA design to improve its frequency. To minimize FPGA resource usage and network latency, CONNECT routers are based on a shallow single-cycle pipeline architecture. As we'll see later, the single-stage router used in CONNECT reaches lower, but still comparable frequency as an ASIC-tuned 3-stage-pipelined router. The FPGA's performance penalty from running at a lower frequency can be much more efficiently made up by increasing the width of the datapath and links or even switching to an entirely different topology.

Reconfigurabilitiy. Given the flexibility of FPGAs stemming from their reconfigurable nature, an effective NoC design is likely to be called to match up against a diverse range of applications. To cover the needs of such a diverse and rapidly changing set of applications, the CONNECT NoC generator is fully parameterized and more importantly topology-agnostic, which means that individual routers can be composed to form arbitrary custom network topologies. Moreover, to minimize changes in the user logic, all CONNECT networks adhere to the same simple standard common interface. From the user's perspective the NoC appears to be a plug-and-play black box device that receives and delivers packets. Rapid prototyping and design space exploration become effortless as any CONNECT network can be seamlessly swapped for another CONNECT network that has the same number of endpoints.

3. CONNECT NOC ARCHITECTURE

CONNECT-based NoCs are meant to be part of larger FPGA-based systems and, as such, must co-exist with the rest of the FPGA-resident components. This means that CONNECT NoCs need to balance between two conflicting goals: (1) provide sufficient network performance to satisfy the communication requirements of the target application; and (2) minimize the use of FPGA resources to maximize the resources available to the rest of the system. CONNECT addresses both goals by making the NoC implementation as efficient as possible, following the principles discussed in the previous section. When compared to ASIC-optimized NoC designs, in many places, the CONNECT NoC architecture goes directly against conventional NoC design wisdom. These differences can be attributed to two fundamental CONNECT design decisions, which are summarized below:

- **Single pipeline stage.** Instead of the typical three to five stage pipeline found in most contemporary Virtual-Channel-based router designs, CONNECT employs a single stage router pipeline, leading to lower hardware cost, lower latency and opportunities for simpler flow control and more efficient buffer usage, due to the reduced round-trip time between routers.
- **Tightly-Coupled Routers.** Instead of treating the NoC as a collection of decoupled routers connected through narrow links, CONNECT tries to maximize wire usage, by using wider interfaces, leading to tighter coupling between routers. This includes carrying flit control information (that would traditionally be carried in separate header flits) on additional wires that run along the data wires. This decoupling is also the driving idea behind CONNECT's "peek" flow control mechanism, that allows routers to directly peek at the buffer occupancy information of their downstream receiving routers and is explained in more detail in [13].

Driven by the special characteristics of FPGAs, we developed a simple router architecture to serve as the basic building block for composing CONNECT networks. Our router design, shown in Figure 1 was implemented using Bluespec

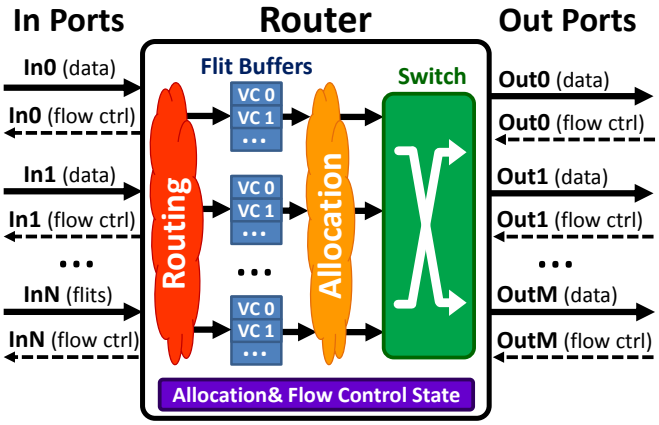

Figure 1: CONNECT Router Architecture

System Verilog (BSV) [3], which allowed us to maintain a flexible parameterizable design. CONNECT routers are heavily configurable and among other parameters, they support:

- Variable number of input and output ports
- Variable number of virtual channels (VCs)
- Variable flit width
- Variable flit buffer depth
- Two flow control mechanisms
- Flexible user-specified routing
- Four allocation algorithms

Below, we focus on some of the most interesting features of the CONNECT NoC Architecture.

Topology-agnostic. A major benefit of allowing any number of input or output ports and being flexible with respect to the routing algorithm is the ability to support arbitrary topologies. As long as flit widths match, CONNECT routers can be hooked to each other and form custom topologies that can better serve the needs of the application at hand. Similarly, all CONNECT networks that connect the same number of endpoints are interchangeable, which can greatly accelerate design space exploration.

Virtual Channels. In order to meet the diverse communication requirements of various applications, CONNECT has support for multiple VCs[1] , which, as explained earlier, are implemented in a very FPGA-efficient manner. Multiple VCs are fundamental for ensuring deadlock freedom, implementing protocols that require traffic isolation between different message classes (e.g., memory requests and responses) and can also be used to increase network performance by reducing the effects of head-of-line blocking [12].

Virtual Links. In order to ease the implementation of receive endpoints in NoCs that use multi-flit packets and employ multiple VCs, CONNECT offers a feature called "Virtual Links". When enabled, this feature guarantees contiguous transmission and delivery of multi-flit packets. In other words, this guarantees that once a packet starts being delivered it will finish before any other packet is delivered. Enabling virtual links can cause a slight increase in hardware cost, but, in return, can significantly reduce the reassembly buffering and logic requirements at the receive endpoints.

[1]In addition to user-exposed VCs (a.k.a. message classes), NoCs often also employ a number of internal VCs within each router to improve network performance. Such VCs are typically only visible and allocated within the network and are not exposed to the network clients. To reduce hardware cost, CONNECT exposes all VC handling to the network clients. As a result, applications seeking to use additional VCs for performance improvements need to manually handle VC allocation at the NoC endpoints.

4. EVALUATION AND RESULTS

To demonstrate the effectiveness of CONNECT's FPGA-centric design choices, we first compare FPGA synthesis results and network performance of a CONNECT-based 2D mesh NoC against a high-quality state-of-the-art ASIC-oriented NoC [14] after modifying their ASIC-style RTL for efficient FPGA synthesis while maintaining bit and cycle accuracy to their original RTL. To further evaluate the CONNECT NoC architecture and highlight its flexibility and extensive design space coverage, we examine multiple CONNECT networks and report FPGA synthesis results and network performance results.

Methodology. Synthesis results are obtained using Xilinx XST 13.1i targeting a moderately sized Xilinx Virtex-6 LX240T FPGA (part xc6vlx240t, speed grade -1) and a large Xilinx Virtex-6 LX760 FPGA (part xc6vlx760, speed grade -2). To assess network performance, we drive the NoCs with various traffic patterns and show the resulting load-delay curves, which are generated through multiple simulations that sweep a range of different network loads. For more information on our methodology, please see our related FPGA 2012 paper [13].

4.1 Comparing to ASIC State-Of-The-Art

To put the FPGA hardware cost and network performance of CONNECT into perspective, we compare it against publicly available RTL of a high-quality state-of-the-art VC-based router [14], which we will refer to as SOTA. This router is written in highly-parameterized Verilog and is modeled after the VC-based router described in [4]. SOTA only supports single or multi-dimensional mesh and torus topologies, as well as the flattened butterfly topology [8].

We compare the two designs at the network level by using CONNECT and SOTA routers to build three 4x4 mesh networks with 4 VCs, 8-entry flit buffers and seperable allocators. Table 1 shows synthesis results for the resulting networks targeting Xilinx Virtex-6 LX240T and LX760 FPGAs. When configured with the same 32-bit flit width (SOTA and CONNECT_32), the SOTA network is more than twice as costly in terms of LUT usage, but can achieve approximately 50% higher clock frequency. The potential performance loss due to the maximum clock frequency difference can be easily regained by adapting other CONNECT NoC parameters, such as flit width. To demonstrate this, we also include results for a 128-bit wide version of the CONNECT mesh NoC (CONNECT_128), that uses aproximately the same FPGA resources as its SOTA 32-bit counterpart, but offers three to four times higher network performance.

	Xilinx LX240T		Xilinx LX760	
4x4 Mesh w/ 4VCs	**%LUTs**	**MHz**	**%LUTs**	**MHz**
SOTA (32-bit)	36%	158	12%	181
CONNECT_32 (32-bit)	15%	101	5%	113
CONNECT_128 (128-bit)	36%	98	12%	113

Table 1: Synthesis Results for CONNECT and SOTA Mesh Network.

To compare the example CONNECT and SOTA NoCs in terms of network performance, we examine the load-delay behavior of the networks under uniform random traffic, where the destination for each packet is randomly selected, and an instance of the unbalanced traffic pattern, where a fraction of the generated packets determined by the *Unbalance Factor* are local and are sent to neighboring nodes. In our experiments we set the *Unbalance Factor* to 90%, which represents a system where nodes communicate heavily with their neighbors and occasionally also send packets to other randomly chosen nodes in the system. We size packets to half the flit buffer depth, which corresponds to 4 flits, and pick the VC randomly for each injected packet.

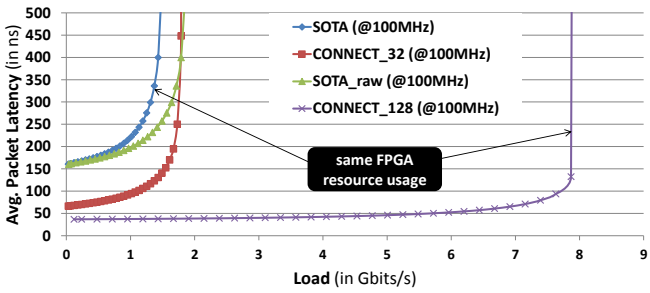

Figure 2: Load-Delay Curves for SOTA & CONNECT @ 100MHz with Unif. Random Traffic.

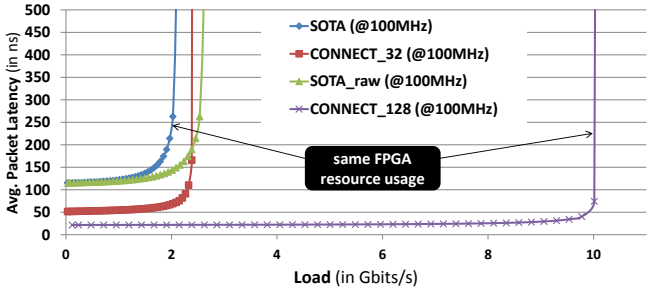

Figure 3: Load-Delay Curves for SOTA & CONNECT @ 100MHz with Unbalanced 90% Traffic.

Since NoCs are typically used within larger systems hosted on an FPGA, their clock frequency is oftentimes dictated by other components and constraints in the system. This is especially true in FPGA environments, where the clock frequency gains of running each component at its maximum frequency are likely to be outweighed by the added synchronization latency increase and hardware cost. In this paper we report network performance results assuming that all networks are running at a common clock frequency of 100 MHz, possibly dicated by some other system component. For additional network performance results that also consider running each network in isolation at its maximum frequency please see our related FPGA 2012 paper [13].

All packets in the SOTA network require an additional header flit that carries control information, which brings the total number of flits per packet to five; one header flit and four data flits. CONNECT does not require this extra header flit; instead it carries flit control information "on the side" using wider links. Since the header overhead can change depending on the specific packet size, we also report the SOTA_raw curve, which eliminates SOTA's header overhead and captures raw flit throughput, providing an upper bound for the fully amortized performance of SOTA.

Figures 2 and 3 present load-delay curves for the CONNECT and SOTA networks all running at the same frequency of 100MHz under the two traffic patterns introduced above. Interestingly, even CONNECT_32, which shares the same 32 bit flit width with SOTA and occupies about half the FPGA resources, yields better network performance, both in terms of latency and saturation throughput. This is due to the additional header flit overhead on the SOTA network. When compared to SOTA_raw, which excludes the header overhead, CONNECT's performance is comparable to SOTA.

However, notice that in all cases CONNECT_128, which occupies about the same FPGA resources as SOTA, can easily outperform all other networks by a wide margin across all traffic patterns and regardless of frequency adjustments; it consistently offers three to four times higher saturation throughput and more than three times lower latency.

Overall, for comparable configurations, CONNECT can offer similar network performance to SOTA with consistently

Network	Routers	Ports/Router	VCs	Width
Ring64	64	2	4	128
DoubleRing16	16	3	4	32
DoubleRing32	32	3	2	32
FatTree16	20	4	2	32
Mesh16	16	5	4	32
Torus16	16	5	2	64
HighRadix8	8	8	2	32
HighRadix16	8	9	2	32

Table 2: Sample Network Configurations.

lower latency at approximately half the FPGA resource usage. Alternatively, for the same FPGA resource budget, CONNECT can offer much higher performance than SOTA – three to four times higher saturation throughput and more than three times lower latency. In all cases the unbalanced traffic pattern, which consists of mostly local traffic, increases the saturation throughput across all networks, which is expected for the mesh topology that performs better under increased neighbor-to-neighbor traffic.

4.2 CONNECT Network Synthesis Results

In this section, to demonstrate the flexibility and extensive design space coverage of CONNECT, we examine a few different examples of CONNECT-based networks in terms of hardware cost and network performance. Table 2 lists the selected network configurations, which range from a low-cost low-performance ring network (Ring16) all the way to a high-performance fully-connected network (HighRadix16), as well as an indirect multistage network (FatTree16). The number next to each network name indicates the number of supported network endpoints. The HighRadix16 network corresponds to a network with eight fully connected routers, where each router is shared by two network endpoints, i.e. with a concentration factor of two.

Table 3 shows synthesis results for these eight sample network configurations targeting a moderately sized Xilinx Virtex-6 LX240T FPGA, as well as a larger Xilinx Virtex-6 LX760 FPGA. For each network we report the LUT usage as a percentage of the total amount of LUTs on the respective FPGA, as well as synthesis clock frequency.

The synthesis results indicate that all networks easily fit within both FPGAs, with plenty of room to spare for placing many other pieces of user logic. In fact, when targeting the LX760 FPGA, all networks occupy less than 10% of the available LUTs. Finally, it is also worth mentioning that CONNECT networks do not occupy even a single Block RAM, which leaves a great amount of on-chip storage available to other FPGA-resident components.

	Xilinx LX240T		Xilinx LX760	
Network	%LUTs	MHz	%LUTs	MHz
Ring64	30%	175	9%	200
DoubleRing16	9%	139	3%	158
DoubleRing32	11%	146	4%	169
FatTree16	12%	117	4%	143
Mesh16	15%	101	5%	113
Torus16	25%	91	8%	100
HighRadix8	20%	73	5%	76
HighRadix16	28%	67	9%	75

Table 3: Synthesis results for sample networks.

4.3 CONNECT Network Performance

In this section, we focus on a subset of four networks (DoubleRing16, Mesh16, FatTree16 and HighRadix16), that all support 16 network clients, and as such would be interchangable when used as the interconnect within an FPGA-based system. To study network performance we use the same two traffic patterns described earlier, uniform random and unbalanced (with an UnbalanceFactor of 90%), which

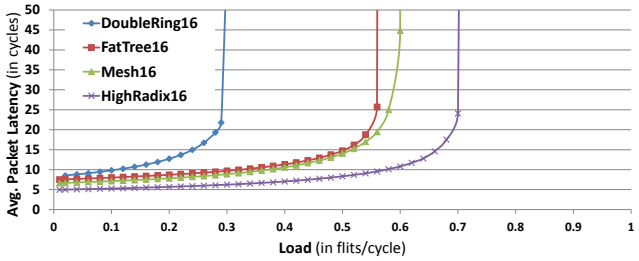

Figure 4: Load-Delay Curves for CONNECT Networks with Uniform Random Traffic.

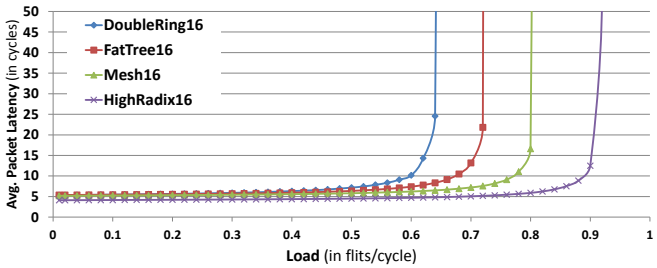

Figure 5: Load-Delay Curves for CONNECT Networks with Unbalanced 90% Traffic.

can be thought of as corresponding to two different classes of FPGA applications, each with different degrees of local communication. Once again, we size packets to half the flit buffer depth, which corresponds to 4 flits, and pick the VC randomly for each injected packet.

Figure 4 shows the load-delay curves for the four selected networks under uniform random traffic. Given the low bisection bandwidth of the double ring topology, the DoubleRing16 network is the first to saturate at a load of approximately 30%. The Mesh16 and FatTree16 networks can sustain much higher loads before they saturate at roughly 55% load. This can be both attributed to the higher connectivity and bisection bandwidth of the mesh and fat tree topology, as well as the higher number of VCs in the case of the Mesh16 network (4 instead of 2). Finally, as expected, the HighRadix16 network achieves the highest performance, offering lower latency across all loads and saturating at a load of 70%. This should come as no surprise, given that the HighRadix16 network is fully-connected (maintains single-hop point-to-point links between all routers in the network), which means that the only source for loss of performance is output contention [4].

Figure 5 shows the equivalent load-delay curves under the unbalanced traffic pattern, which favors mostly neighbor-to-neighbor communication. As expected, the increased locality allows all networks to perform better, with the DoubleRing16 network experiencing the largest relative performance gains. In fact, under unbalanced traffic the DoubleRing16 network outperfoms the more FPGA resource intensive Mesh16 and FatTree16 networks.

Even though these results are mainly presented to demonstrate the flexibility of CONNECT and, as such, are not exhaustive or might omit other implementation details, such as frequency-related constraints, they do show that NoC performance can be highly dependent on network topology and configuration, but more importantly on the specific traffic patterns and requirements of an application. This observation is especially important in the context of FPGAs, where NoC topology and configuration can be easily adapted to suite the requirements of the given application.

5. RELATED WORK

There is only a limited amount of previous studies that focus on FPGA-tailored NoC architectures to support FPGA-based architectural research or SoC prototyping.

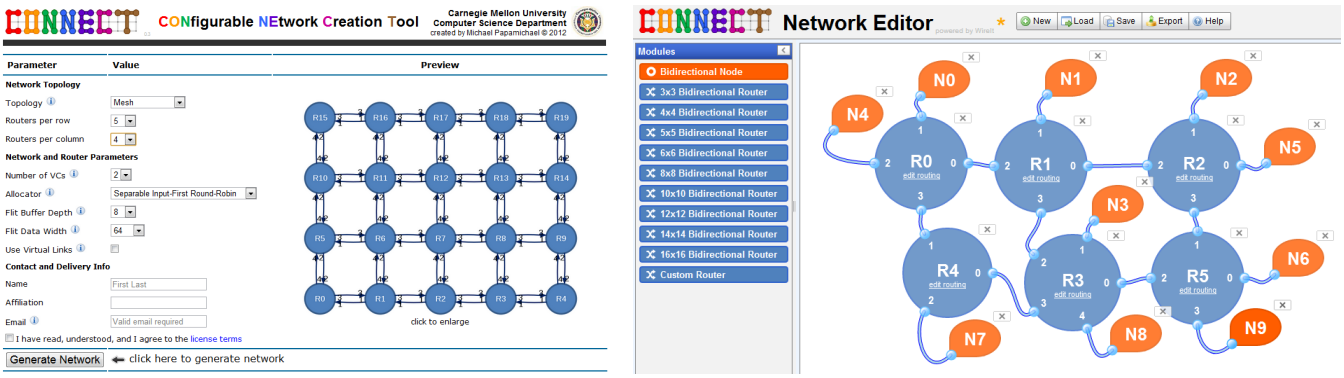

Figure 6: Screenshots of our publicly released web-based CONNECT NoC Generator and Network Editor.

In the context of FPGA-oriented NoC architectures, NoCem [5] presents a very simple router block that can be used to compose larger networks on FPGAs. Compared to CONNECT it lacks more advanced features, such as support for virtual channels or selectable allocation and flow control schemes. More importantly, it appears to incur a much larger number of FPGA resources for equivalent networks. PNoC [6] is an interesting proposal for building lightweight networks to support FPGA-based applications. Even though PNoC can also yield low-cost FPGA-friendly networks, the fundamental difference compared to CONNECT is that it can only be used to create circuit-switched networks, instead of packet-based. Circuit-switched networks can be useful for those classes of FPGA applications that have structured non-conflicting traffic patterns and are willing to tolerate the additional setup and tear-down delay and connection management associated with circuit-switched networks.

Finally, there is also a large body of commercial interconnect approaches, such as Spidergon STNoC [11], ARM's AMBA [2] or even FPGA-specific approaches, such as the CoreConnect [7] PLB and OPB buses, commonly found in Xilinx FPGAs, or Altera's Qsys [1]. CONNECT offers a more lightweight, fine-grain and flexible FPGA-tailored solution for building soft NoCs, that can synergistically coexist with the above approaches to cover the diverse communication needs of FPGA-based research and emerging SOCs.

6. CONNECT ONLINE NOC GENERATOR

In an effort to create a useful research tool we recently released a web-based front-end to CONNECT's NoC RTL generation engine, hosted at `http://www.ece.cmu.edu/~mpapamic/connect`. Figure 6 shows two screenshots of CONNECT's web-based interface. CONNECT supports a variety of common network topologies, as well as custom user-configured networks, that can be either specified through special configuration files or created through a web-based graphical network editor (Figure 6 - right). A variety of network and router parameters, such as the number of virtual channels or allocator type, allow the user to further customize the network and trade-off between resource usage and performance to satisfy application-specific requirements.

7. CONCLUSION

In this paper, we presented CONNECT, a flexible and efficient approach for building NoCs for FPGA-based systems. CONNECT embodies a set of design guidelines and disciplines to make the most efficient use of the FPGA substrate and in many cases go against ASIC-driven conventional wisdom in NoC design. We compare a high-quality ASIC-oriented NoC design against our design both in terms of FPGA cost, as well as network performance for a similarly configured 4x4 mesh NoC. Across a wide range of configuration parameters, we find that CONNECT consistently offers lower latencies and can achieve comparable network performance at one-half the FPGA resource cost; or alternatively, three to four times higher network performance at approximately the same FPGA resource cost. Moreover, to demonstrate the flexibility and extensive design space coverage of CONNECT we report synthesis and network performance results for a variety of diverse CONNECT-based networks.

8. REFERENCES

[1] Altera. Qsys System Integration Tool. `http://www.altera.com/products/software/quartus-ii/subscription-edition/qsys/qts-qsys.html`.

[2] ARM. AMBA Open Specifications. `http://www.arm.com/products/solutions/AMBAHomePage.html`.

[3] Bluespec, Inc. Bluespec System Verilog. `http://www.bluespec.com/products/bsc.htm`.

[4] W. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann, 2004.

[5] G. Schelle and D. Grunwald. Exploring FPGA Network on Chip Implementations Across Various Application and Network Loads. In *FPL*, 2008.

[6] C. Hilton and B. Nelson. PNoC: A Flexible Circuit-Switched NoC for FPGA-based Systems. *IEE Proceedings Computers and Digital Techniques*, 2006.

[7] IBM. The Coreconnect Bus Architecture. `https://www-01.ibm.com/chips/techlib/techlib.nsf/products/CoreConnect_Bus_Architecture`, 1999.

[8] J. Kim, J. Balfour, and W. Dally. Flattened Butterfly Topology for On-Chip Networks. In *MICRO*, 2007.

[9] I. Kuon and J. Rose. Measuring the Gap Between FPGAs and ASICs. *IEEE TCAD*, 2007.

[10] J. Lee and L. Shannon. The Effect of Node Size, Heterogeneity, and Network Size on FPGA based NoCs. In *International Conference on Field-Programmable Technology (FPT)*, 2009.

[11] M. Coppola et al. Spidergon: A Novel On-Chip Communication Network. In *SoC*, 2004.

[12] M. Karo; M. Hluchyj; S. Morgan. Input Versus Output Queuing on a Space-Division Packet Switch. In *IEEE Transactions on Communications*, 1987.

[13] M. K. Papamichael and J. C. Hoe. CONNECT: Re-Examining Conventional Wisdom for Designing NoCs in the Context of FPGAs. In *FPGA*, 2012.

[14] Stanford Concurrent VLSI Architecture Group. Open Source Network-on-Chip Router RTL. `https://nocs.stanford.edu/cgi-bin/trac.cgi/wiki/Resources/Router`.