FPGA Research Design Platform Fuels Network Advances

Xilinx and Stanford University are teaming up to create an FPGA-based reference board and open-source IP repository to seed innovation in the networking space.

by Michaela Blott
Senior Research Engineer
Xilinx, Inc.
mblott@xilinx.com

Jonathan Ellithorpe
PhD Candidate
Stanford University

Nick McKeown
Professor of Electrical Engineering and Computer Science
Stanford University

Kees Vissers
Distinguished Engineer
Xilinx, Inc.

Hongyi Zeng
PhD Candidate
Stanford University

Stanford University, together with Xilinx Research Labs, is building a second-generation high-speed networking design platform called the NetFPGA-10G specifically for the research community. The new platform, which is poised for completion this year, uses state-of-the art technology to help researchers quickly build fast, complex prototypes that will solve the next crop of technological problems in the networking domain. As with the first-generation platform, which universities worldwide have eagerly adopted, we hope the new platform will spawn an open-source community that contributes and shares intellectual property, thus accelerating innovation.

An open-source hardware repository facilitates the sharing of software, IP and design experiences, promoting technological solutions to the next generation of networking problems.

This basic platform will provide everything necessary to get end users off the ground faster, while the open-source community will allow researchers to leverage each other's work. The combination effectively reduces the time spent on the actual implementation of ideas to a minimum, allowing designers to focus their efforts on the creative aspects.

In 2007, we designed the first-generation board, dubbed NetFPGA-1G, around a Xilinx® Virtex®-II Pro 50, primarily to teach engineering and computer science students about networking hardware. Many EE and CS graduates go on to develop networking products, and we wanted to give them hands-on experience building hardware that runs at line rate, uses an industry-standard design flow and can be placed into an operational network. For these purposes, the original board had to be low in cost. And with generous donations from several semiconductor suppliers, we were able to bring the design in at an end price of under $500. As a result, universities were quick to adopt the board and today about 2,000 NetFPGA-1Gs are in use at 150 schools worldwide.

But the NetFPGA very quickly became more than a teaching vehicle. Increasingly, the research community began to use it too, for experiments and prototypes. For this purpose, the NetFPGA team provided free, open-source reference designs and maintains a repository of about 50 contributed designs. We support new users, run online forums and offer tutorials, summer camps and developers' workshops.

Trend Toward Programmability

For more than a decade, networking technology has trended toward more-programmable forwarding paths in switches, routers and other products. This is largely because networking hardware has become more complicated, thanks to the advent of more tunneling formats, quality-of-service schemes, firewall filters, encryption techniques and so on. Coupled with quickly changing standards, those factors have made it desirable to build in programmability, for example using NPUs or FPGAs.

Researchers often want to change some or all of the forwarding pipeline. Recently, there has been lots of interest in entirely new forwarding models, such as OpenFlow. Researchers can try out new network architectures at the national scale at national testbeds like GENI in the United States (*http://geni.net*) and FIRE in the EU (*http://cordis.europa.eu/fp7/ict/fire/calls_en.html*).

Increasingly, researchers are embracing the NetFPGA board as a way to prototype new ideas—new forwarding paradigms, scheduling and lookup algorithms, and new deep-packet inspectors—in hardware. One of the most popular reference designs in the NetFPGA canon is a fully functional open-source OpenFlow switch, allowing researchers to play with variations on the standard. Another popular reference design accelerates the built-in routing functions of the host computer by mirroring the kernel routing table in hardware and forwarding all packets at line rate.

NetFPGA, Take Two

For the second-generation platform, the so-called NetFPGA-10G, we have expanded our original design goals to also include ease of use, with the aim of supplying end customers with a basic infrastructure that will simplify their design experience. This goal is closely aligned with the objective of Xilinx's mainstream Targeted Design Platforms, which provide users with tools, IP and reference designs in addition to FPGA silicon so as to speed up the design process.

To realize this vision, we will deliver a board with matching FPGA infrastructure design in the form of both basic and domain-specific IP building blocks to increase ease of use and accelerate development time. We further will develop reference designs, such as a network interface card and an IPv4 reference router, as well as basic infrastructure that assists with building, simulating and debugging designs. The idea is to allow users to truly focus their development time on their particular area of expertise or interest without having to worry about low-level hardware details.

Unlike the mainstream Targeted Design Platforms, our networking platform targets a different end-user group, namely, the larger research community, both academic and commercial. Semiconductor partners are heavily subsidizing this project to assist in the effort to keep the board cost to an absolute minimum so as to encourage broad uptake. Not only Xilinx but other key component manufacturers such as Micron, Cypress Semiconductor and NetLogic Microsystems are donating parts for the academic end user. (Commercial researchers will pay a higher price.)

Part of the project's strength is the fact that this platform is accompanied by a community as well as an open-source hardware repository that facilitates the sharing of software, IP and experiences beyond the initial base package. The result is an ever-growing IP library that we hope will eventually encompass a wide range of reference components, networking IP, software and sophisticated infrastructure thanks to contributions from many well-known universities, research groups and companies. We hope that by providing a carefully designed framework, some lightweight coordination to share expertise and IP

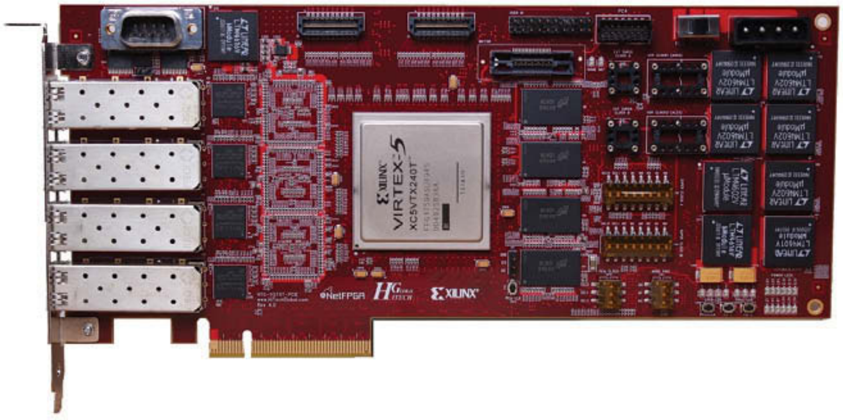

Figure 1 – The NetFPGA-10G board is built around a Virtex-5 FPGA.

in a systematic way, and a well-designed plug-and-play architecture with standardized interfaces, the open-source hardware repository will grow, promoting technological solutions to the next generation of networking problems.

The NetFPGA-10G is a 40-Gbit/second PCI Express® adapter card with a large FPGA fabric that could support as many applications as possible. As shown in Figure 1, the board design revolves around a large-scale Xilinx FPGA, namely a Virtex-5 XC5VTX240T-2 device [1].

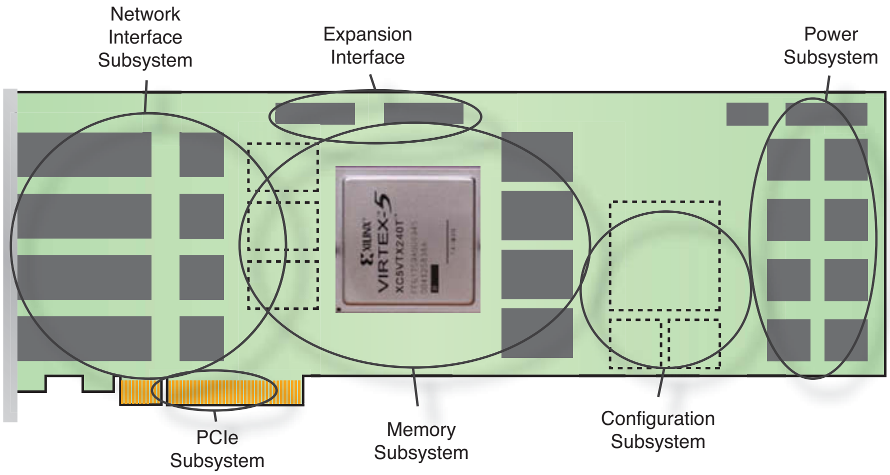

Figure 2 – The board connects to five subsystems: network interface, PCIe, expansion, memory, configuration and power.

The FPGA interfaces to five subsystems (see Figure 2). The first of these, the network interface subsystem, hosts four 10-Gbps Ethernet interfaces with PHY devices through which the network traffic enters the FPGA. The memory subsystem, for its part, consists of several QDRII and RLDRAMII components. The majority of the I/Os are used for this interface, to maximize the available off-chip bandwidth for functionality such as routing tables or packet buffering. The FPGA also interfaces to the PCIe subsystem.

The fourth subsystem is an expansion interface designed to host a daughter card or to communicate with another board. For that purpose we brought all remaining high-speed serial I/O connections out to two high-speed connectors. Finally, the fifth subsystem is concerned with the configuration of the FPGA.

Overall, the board is implemented as a dual-slot, full-size PCIe® adapter. Two slots are required for reasons of heat/power and height. Like high-end graphics cards, this board needs to be fed with an additional ATX power supply, since power consumption—given absolute maximum load on the FPGA—could exceed the PCIe single-slot allowance of 50 watts. However, the board can also operate standalone outside a server environment.

Memory Subsystem

A central focus in our design process was the interface to SRAM and DRAM components. Since the total number of I/Os on the FPGA device limits the overall available off-chip bandwidth, we had to strike a carefully considered compromise to facilitate as many applications as possible. Trying to support applications ranging from network monitoring through security and routing to traffic management imposes greatly varying constraints.

In regard to external memory access, for example, a network monitor would require a large, flow-based statistics table and most likely a flow classification lookup. Both accesses would require short latencies, as the flow classification needs more than one lookup with intradependencies, and the update in a flow statistics table would typically encompass a read-modify-write cycle. Hence, SRAM would be an appropriate device selection. However, a traffic manager would mainly need a large amount of memory for packet buffering, typically implemented through DRAM components due to density requirements. As a final example, consider an IPv4 router that needed a routing-table lookup as well as packet buffering in respect to external memory.

Summing up the requirements from a range of applications, we realized that certain functionality would consume external memory bandwidth, whether it be SRAM or DRAM. Where packet buffering (requiring a large storage space) would point to DRAM, SRAM would be the choice for flow classification search access, routing-table lookup, flow-based data table for statistics or rule-based firewall, memory management tables for packet buffer implementations and header queues.

All of this functionality needs to be carried out on a per-packet basis. Therefore, given the worst case of minimum-size packets of 64 bytes with 20 bytes of overhead, the system needs to service a packet rate of roughly 60 Megapackets per second. Second, we need to differentiate the accesses further. To begin with, many memory components such as QDR SRAM and RLDRAM SIO devices have separate read and write data buses. Since the access patterns cannot be assumed to be symmetrically distributed, we cannot pool the overall access bandwidth, but must consider the operations individually.

What's more, there is a third type of access, namely "searches." Searches can be ideally implemented through TCAM-based devices, which give guaranteed answers with fixed latencies. However, we ruled out this type of device for our board for price and power reasons, along with the fact that TCAMs further constrain the use of the I/O. Searches can also be implemented in many other ways such as decision trees, hash algorithms and decomposition approaches, with or without caching techniques, to name a few [2]. For the purpose of this discussion, we assumed that a search can be approximated through two read accesses on average. Given these facts and making some common assumptions on data widths, we refined our requirements further as seen in Table 1.

	Data width (bits)	#Reads per packet	#Writes per packet	#Reads per x36 interface	#Writes per x36 interface
Flow classification (5tupel + return key)	144	2	0	8	0
Routing-table lookup (dip + next hop)	96	2	0	6	0
Flow-based information	128	2	0	8	0
Packet buffer memory management	32	2	2	2	2
Header queues	32	1	1	1	1
Total number of accesses				**25**	**3**
Total number of accesses per second (MAps)				**1,500**	**180**

Table 1 – SRAM bandwidth requirements

Assuming a clock speed of 300 MHz on the interface, a QDRII interface can service 2*300 = 600 million accesses/second for read and for write operations. Hence, three QDRII x36-bit interfaces could fulfill all of our requirements.

In regard to DRAM access, we considered mainly the case of packet storage, where each packet is written and read once from memory. This translates into a data access bandwidth of roughly 62 Gbps once you have removed the packet overhead from the originally incoming 2*40 Gbps. In terms of physical resources, an RLDRAMII access can probably achieve an efficiency of around 97 percent, whereas DDR2 or DDR3 devices would more likely come in at around 40 percent [3], hence requiring significantly more I/O. We therefore chose RLDRAMII CIO components. Two 64-bit RLDRAMII interfaces at 300 MHz deliver a combined bandwidth that is roughly enough to service the requirement.

Network Interface

The network interface of the NetFPGA-10G consists of four subsystems that can be operated as 10-Gbps or 1-Gbps Ethernet links. To maximize usage of the platform as well as minimize power consumption, we chose four Small Form-Factor Pluggable Plus (SFP+) cages as the physical interface. Compared with other 10G transceiver standards such as XENPAK and XFP, SFP+ has significant advantages in terms of power consumption and size. With SFP+ cages, we can support a number of interface standards, including 10GBase-LR, 10GBase-SR, 10GBase-LRM and low-cost direct-attach SFP+ copper cable (Twinax). Furthermore, SFP modules for 1-Gbps operation can be utilized, thereby supporting the 1000Base-T or 1000Base-X physical standards as well.

Each of the four SFP+ cages connects to a NetLogic AEL2005 device via the SFI interface. The AEL2005 is a 10-Gigabit Ethernet physical-layer transceiver with an embedded IEEE 802.3aq-compliant electrical-dispersion compensation engine. Besides regular 10G mode, the PHY device can support Gigabit Ethernet (1G) mode. On the system side, these PHY devices interface to the FPGA via a 10-Gigabit Attachment Unit Interface (XAUI). When operating in 1G mode, one of the XAUI lanes works as a Serial Gigabit Media Independent Interface (SGMII).

Suitable interface IP cores are available inside the Virtex-5 FPGA. For 10-Gbps operation, Xilinx markets the XAUI LogiCORE™ IP as well as a 10-Gigabit Ethernet media-access controller (10GEMAC) LogiCORE IP [4, 5]. For 1G operation, the interfaces can be directly connected to the embedded Xilinx Tri-Mode Ethernet MAC core [6].

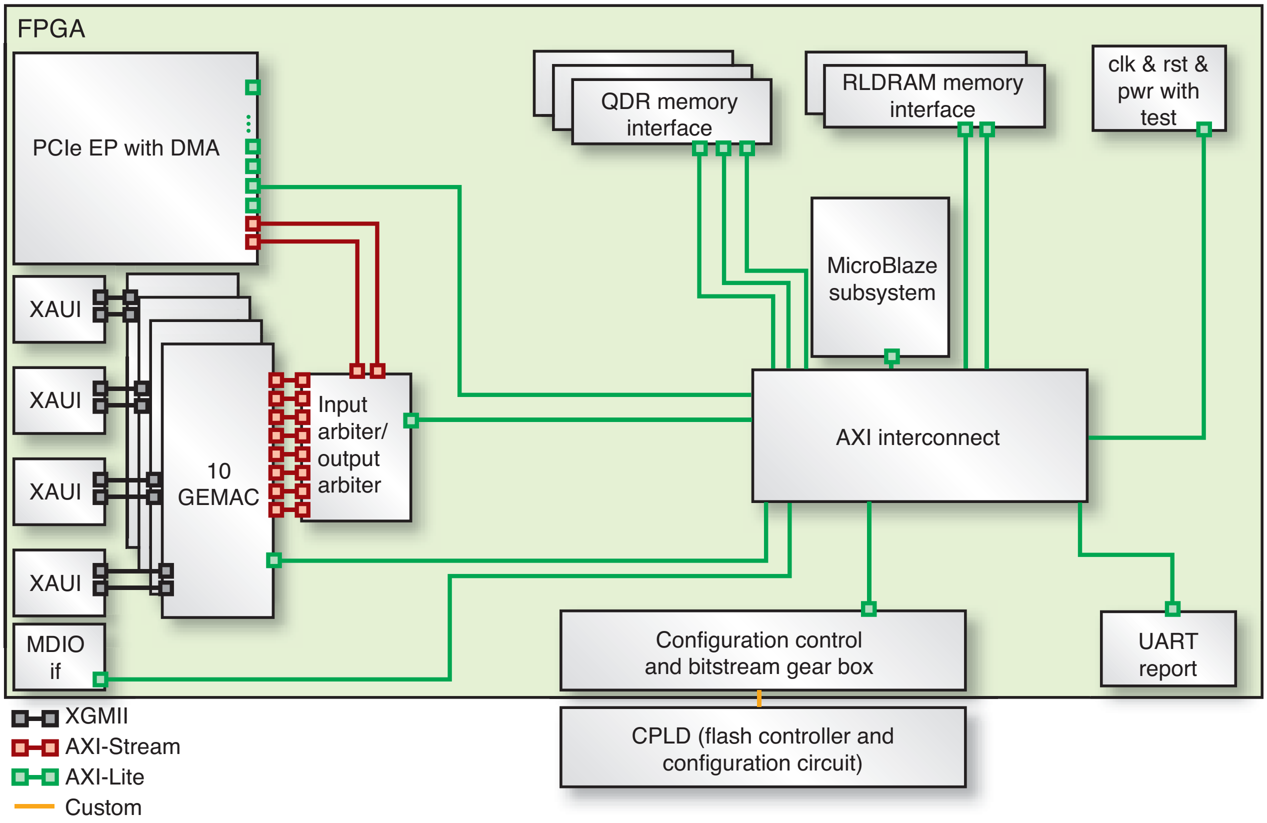

Figure 3 – The FPGA design architecture for 10G operation relies on the AXI protocol.

PCIe Subsystem

The NetFPGA-10G platform utilizes the Open Component Portability Infrastructure (OpenCPI) as its main interconnect implementation between the board and the host computer over PCIe [7]. Presently we support the x8 first generation, but might potentially upgrade to second generation in the future. OpenCPI is a generalized, open-source framework for interconnecting blocks of IP that may each use varied types of communication interfaces (streaming, byte-addressable and so on), with varied bandwidth and latency requirements. In its essence, OpenCPI is a highly configurable framework that provides the critical communication “glue” between IP that is needed to realize a new design quickly.

OpenCPI delivers a few key features for the NetFPGA-10G platform. On the software side, we are able to provide a clean DMA interface for transferring data (primarily including packets, although not limited in any way to this type of information) and for controlling the device via programmed input/output. For networking applications, we provide a Linux network driver that exports a network interface for each of the physical Ethernet ports on the NetFPGA device. This allows user-space software to transfer network packets to the device as well as read any host-visible register in the design.

On the hardware side, OpenCPI provides us with a clean, or even multiple, data-streaming interfaces, each configurable through the OpenCPI framework. In addition, OpenCPI handles all of the host-side and hardware-side buffering and PCIe transactions, so that users can focus on their particular application rather than the details of device communication.

Expansion Interface, Configuration Subsystem

The purpose behind the expansion subsystem is to allow users to increase port density by connecting to a second NetFPGA-10G board—to add different flavors of network interfaces through an optical daughter card, for example—or to connect additional search components such as knowledge-based processors with high-speed serial interfaces. We bring out 20 GTX transceivers from the FPGA and connect them through AC-coupled transmission lines to two high-speed connectors. Designed for transmission interfaces such as XAUI, PCIe, SATA and Infiniband, these connectors can link to another board either directly, with mating connector, or via cable assemblies. Each transmission line is tested to operate at 6.5 Gbps in each direction, thereby providing an additional 130-Gbps data path in and out of the FPGA.

Configuring FPGAs with ever-increasing bitstream size can potentially be a problem when the configuration time of the device exceeds the PCIe limits. This is the case with the V5TX240T. The access speed of the platform flash devices, which is significantly below what the V5TX240T can theoretically handle, imposes a bottleneck. As countermeasures, designers might consider configuration of partial bitstreams, as well as accessing platform flash devices in parallel and configuring the FPGA at maximum speed. To facilitate the latter possibility, we equipped the board with two platform flash devices that connect through a CPLD to the FPGA's configuration interface. In addition, the board also supports the standard JTAG programming interface.

The FPGA Design

A key aspect behind a successful open-source hardware repository must be a clean architectural specification with standardized, abstracted and well-defined interfaces. In fact, we believe these interfaces to be crucial to providing a building-block system that enables easy assembly of components contributed from a large global community. Standardization ensures physical compatibility, and a rigorous definition that goes beyond physical connectivity reduces the potential for misunderstanding on the interface protocols. Ideally, users can then deploy abstracted components without further knowledge of their intrinsic implementation details.

As part of the full platform release, we will provide the complete architectural specification as well as a reference design with key components. These key components include:

- PCIe endpoint from OpenCPI [4]
- Four XAUI LogiCORE IP blocks and four 10GEMAC LogiCORE cores (the latter under licensing agreement) in 10G mode [7,8]
- Two RLDRAMII memory controllers based on the XAPP852 [8]
- Three QDRII memory controllers based on MIG3.1 [9]
- Clocking and reset block that checks clock frequencies and power-OK signals, and generates all required system clocks
- Tri-Mode Ethernet MAC (TEMAC) for 1G operation [9]
- MDIO interface for configuration of the PHY devices
- Control processor in the form of a MicroBlaze®, with a support system that handles all administrative tasks
- UART interface
- Configuration control and bitstream gear box, which provides a programming path to the platform flash devices
- Input arbiter that aggregates all incoming traffic into one data stream
- Output arbiter that distributes the traffic to the requested output port

Figure 3 illustrates how these components are interconnected for 10G operation. For data transport we chose the AMBA®4 AXI streaming protocol, and for control traffic the interface is based on AMBA4 AXI-Lite. You can find further details on these interface specifications on *www.arm.com* (and on the Xilinx webpage in the near future).

Status and Outlook

Design verification of the board is complete. Once production test development is finished, the board should shortly be released for manufacturing. Initial FPGA designs are running and we are working now toward a first code release.

You can order through HiTech Global, which is manufacturing and distributing the board (*http://www.hitechglobal.com/Boards/PCIExpress_SFP+.htm*). Different pricing models apply to academic and commercial customers. For the most up-to-date information on the project, please visit our website, *www.netfpga.org*, or subscribe to the netfpga-announce mailing list: *https://mailman.stanford.edu/mailman/listinfo/netfpga-announce*.

The first code releases, planned for the coming months, will open up the source code to the wider community. It basically contains the design detailed in the FPGA design section and implements the introduced architecture. In addition, we will put significant effort into providing the right type of infrastructure for building, simulating and debugging the design. A last focus point in the coming months will be the repositories and framework that enable the efficient sharing of experience, expertise and IP within the NetFPGA community.

Acknowledgements

Many thanks to the following people for their help with this project: Shep Siegel, CTO of Atomic Rules and driving force behind OpenCPI; John Lockwood, CEO of Algo-Logic Systems, driving force behind the first generation of NetFPGA and an active contributor to the current platform; Paul Hartke from Xilinx University Program (XUP) for his endless energy and support; Rick Ballantyne, senior staff engineer at Xilinx, for his invaluable advice on board design; Adam Covington and Glen Gibb at Stanford for their support; Fred Cohen from HiTech Global for his cooperation.

Furthermore, we would like to thank the National Science Foundation and the following sponsors for donating their components, namely Micron, Cypress Semiconductor and NetLogic Microsystems.

References

[1] XC5VTX240T data sheet: http://www.xilinx.com/support/documentation/data_sheets/ds202.pdf

[2] David E. Taylor, "Survey and Taxonomy of Packet Classification Techniques," ACM Computing Surveys (CSUR), volume 37, No. 3, September 2005

[3] Bob Wheeler and Jag Bolaria, Linley Report: "A Guide to Network Processors," 11th Edition, April 2010

[4] XAUI data sheet: http://www.xilinx.com/products/ipcenter/XAUI.htm

[5] 10GEMAC data sheet: http://www.xilinx.com/products/ipcenter/DO-DI-10GEMAC.htm

[6] Tri-Mode Ethernet MAC http://www.xilinx.com/support/documentation/ip_documentation/hard_temac.pdf

[7] www.opencpi.org

[8] XAPP852 (v2.3) May 14, 2008: http://www.xilinx.com/support/documentation/application_notes/xapp852.pdf

[9] MIG data sheet: http://www.xilinx.com/products/ipcenter/MIG.htm