An Open-Source Hardware Module for High-Speed Network Monitoring on NetFPGA

Gianni Antichi
Dept. Information Engineering
University of Pisa
gianni.antichi@iet.unipi.it

David J. Miller
Computer Laboratory
University of Cambridge
david.miller@cl.cam.ac.uk

Stefano Giordano
Dept. Information Engineering
University of Pisa
stefano.giordano@iet.unipi.it

ABSTRACT

We present a passive network measurement solution based on the low-cost NetFPGA — suitable for network research, security applications, and traffic engineering and management. Key features include accurate timestamping, and the ability to filter traffic based on flow. In this paper, we describe our implementation.

Keywords

High Performance, FPGA, Low-cost, Monitoring, Timestamping

1. INTRODUCTION

Network measurement and monitoring has been an active area of research for at least the past 15 years. Applications include academic research, security, and traffic engineering and management.

An ideal measurement and monitoring solution would be accurate, guarantee no loss of information, and be inexpensive. To be cheap, a solution must use off-the-shelf components such as common PCs with their built-in network adaptors. Software running on the host timestamps each packet as it arrives, and stores it for later analysis. Applications such as tcpdump [11], Wireshark [5], nTop [3] *et al.* demonstrate how effective and flexible this approach can be for a large variety of monitoring tasks.

Many of the first solutions were like this, and some still are. This approach works well enough for low speed networks, or when timestamp accuracy is not too important, but it doesn't scale to high speed networks.

High speed network adaptors must employ a variety of techniques to manage the load imposed upon the host processor. One such technique — interrupt mitigation — notifies the host only about groups of packets, rather than when each individual packet arrives. Since the software (usually, an OS kernel) can't timestamp packets until notified of their arrival, interrupt mitigation and interrupt latency both contribute to poor time-stamp quality.

If traffic load exceeds a host's capacity to process and store it, data are lost — usually with no record of it ever having existed. Even when the host can keep up, time taken to process packets that are of no interest is time not spent on analysis of the traffic of interest [6, 7].

Host load can be relieved if the network interface card relays only traffic of interest, but most typical network interfaces aren't capable of filtering traffic based on flow (or class) — only on whether the packet is addressed to the host or not, which is never the case for a monitor.

As described later, a variety of specialised hardware-based solutions exist — but they come at a cost. The NetFPGA platform, which is open and low-cost, offers a new opportunity to achieve the performance of hardware-based measurement solutions, but at costs closer to that of software-only solutions.

Our monitoring solution addresses the criterion of cost by using the NetFPGA. Achieving high quality timestamp data is easy in hardware, and the degree to which an FPGA-based solution can be customised makes it possible to provide filtration such that only traffic of interest is sent to the host.

We describe a flexible monitoring solution offering high quality timestamps and flow-based filtration at line-rate Gigabit Ethernet based on NetFPGA.

2. RELATED WORK

Passive measurement systems have been an active area of research since at least the mid 1990s. The work done by Ian Graham, Stephen Donnelly [9], Jörg Micheel, their colleagues in the WAND Group at Waikato University, and then later Endace [1], produced the excellent, but expensive DAG card.

A similar FPGA-based monitoring and filtering solution was employed by the SCAMPI project [4] using the COMBO6 card from CESNET. Filtration is performed by the FPGA, and only matching packets are forwarded to the host.

Ficara *et al.* [10] present an Intel IXP2400-based traffic monitoring device for Gigabit Ethernet capable of analysing up to 50,000 filtering rules at line rate.

Luca Deri's nCap [8] and related works provide software-based measurement techniques that work well, but are at the mercy of kernel-based timestamping.

Wolf *et al.* [13] propose Distributed On-line Measurement Environment (DOME), a distributed network of passive measurement nodes based on an Intel IXP2400 Network Processor. DOME includes header anonymization, and performance is comparable with that of Endace DAG 4.3 cards. Both the previous systems are able to analyse up to 500 MB/s of minimum-sized packets (64 bytes).

These software solutions are both inexpensive and flexible, but traffic load and timestamp quality are both limited by NIC hardware and kernel performance. Hardware solutions typically provide very good timestamp quality, but hardware is typically expensive and offer limited flexibility (especially in the case of proprietary offerings).

A NetFPGA-based solution offers the accuracy of hardware timestamping on inexpensive hardware (thanks to support from Xilinx) with the flexibility of open firmware, together with a rapidly growing community of developers and academics.

3. THE NETFPGA NETWORK MONITOR

This section describes the design and implementation of our NetFPGA-based time-stamping network monitor.

3.1 Deployment

A network monitor may either be installed in-series with the link to be monitored, or connected by means of a network tap. Optical network links make the choice is easy: passive optical splitters are inexpensive, and other than during initial installation, offer no possibility of interruption of the link.

Copper network links, on the other hand, are more challenging. Some protocols, such as 10/100 Ethernet can be tapped using a passive resistive network — but others (including Gigabit Ethernet) require an expensive active tap, such as the NetOptics TP-CU3, or installation of the monitor in-line.

In-line monitoring is cheap, and offers the possibility of building an Intrusion Prevention System) system, but comes at the cost of significant extra latency and the risk of interruption of the link, should the monitor lose power, be misconfigured, or otherwise fail.

Since the NetFPGA has four ports, but supports only copper Ethernet, our monitoring solution integrates the function of an active copper tap by internally coupling two ports of the card. Traffic received on one port is retransmitted out of the other, and visa versa.

Where a deployment is especially cost-sensitive, a single NetFPGA-based monitor is sufficient for a single full-duplex link. (Our solution could be modified to monitor two full-duplex links with ease.) Where uptime is more critical, our monitor may also be used with a conventional active copper Gigabit Ethernet tap.

3.2 NetFPGA data path

Our monitor adds two new modules (shown in Figure 2) to the standard NetFPGA pipeline (Figure 1) described as follows, as well as making minor modifications to the input arbiter, and `nf2_mac_grp.v`.

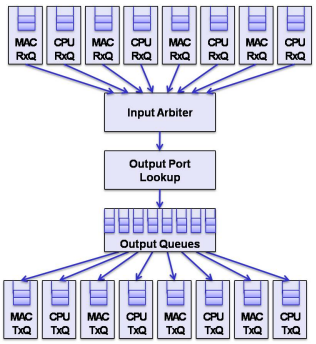

Figure 1: The NetFPGA reference pipeline

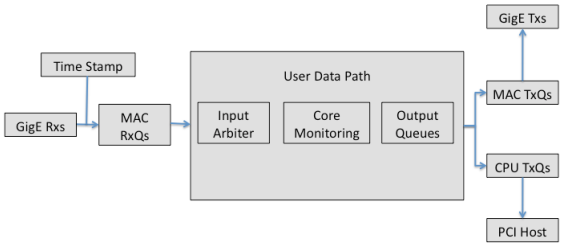

Figure 2: Data path with timestamp modules

3.2.1 Timestamping module

The time stamping module attaches to the RGMII (Reduced Gigabit Media Independent Interface), as near as possible to the MAC (Media Access Controller) so that timestamps are recorded as soon as packets are received, in order to minimise timestamp error and jitter. The RGMII asserts its "data valid" signal when the SFD (Start of Frame Delimiter) of a frame is received at a physical interface. We sample the free-running timestamp counter on the low-to-high transition of this signal.

Timestamps are sampled from a 64-bit, free-running counter driven by the 125 MHz system clock, which increments by 8 once every 8 ns. By using the system clock, the time stamp module can be made synchronous with receive logic, and thereby avoid additional error associated with crossing clock domains.

At start-of-day, the timestamp counter can be initialised with current time derived from the PC real-time clock (which may optionally be maintained using NTP) yielding absolute timestamps.

This timestamp counter is easy to implement, but provides no means of correcting for oscillator drift and

yields data in units of whole nanoseconds. Since standard formats record time in units of seconds, conversion by a floating-point division is required. Both of these drawbacks can be addressed by means of using Direct Digital Synthesis [12], a technique of producing arbitrarily variable frequencies using FPGA-friendly, purely synchronous, digital logic.

By means of an external time reference, such as either NTP, or GPS, the rate of the DDS clock can both be corrected in real-time as well as yield a fixed-point representation in seconds. We plan to add such a system, similar to that described by Stephen Donnelly [9] in a future revision of our measurement solution.

3.2.2 *Selective, flow-based monitoring*

Because the NetFPGA PCI interface lacks the bandwidth to record all traffic, we provide a 5-tuple (IP address pair, protocol and port pair) filter. As described in Section 3.1, all packets received are retransmitted. Packets that match one of up to 32 filter rules are also copied verbatim, with their timestamp prepended (as shown in Figure 3), to the host. The timestamp is converted to Intel (little-endian) byte order in the card to save the host most commonly used in these applications from having to do so.

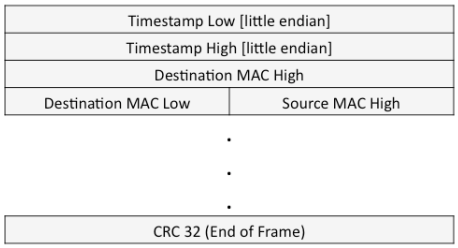

Figure 3: Format of packets sent to host

The timestamps of packets that match no rule are made available to the host via registers. In future versions, we plan to pass unmatched timestamps — perhaps along with minimal information about the packet — to the host, in-band with those packets that did pass the filter.

Our initial implementation uses the TCAM (Ternary Content Addressable Memory) modules available in Xilinx CoreGen. Although these TCAMs are fast (one search every clock cycle), and permit on-the-fly rule updates, they are low density. Owing to problems with timing closure, we found it necessary to implement the filter using two 16-entry TCAMs, rather than one 32-entry TCAM.

In addition, our initial implementation doesn't automatically try both combinations of source and destination port and address, requiring two rule slots to specify a complete flow. Rather than try to address this limitation, we feel that a Bloom filter would provide considerably greater density, while also providing the flexibility of specifying only one half of a flow, should that be desirable.

Since the object of the filter is to manage PCI interface throughput by limiting irrelevant traffic, any false positive matches from the bloom filters are harmless, and the host can simply throw them away.

3.2.3 *Pipeline changes*

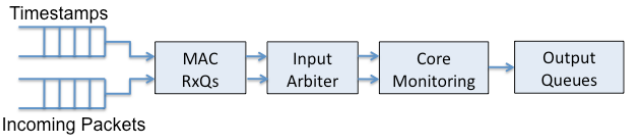

Figure 4: Packet data and side-band timestamps

Our initial implementation passes timestamps in a side-channel, parallel with the main packet data path, as shown in Figure 4. This required minor modifications to parts of the standard data path. Future versions of this code will probably pass timestamp information in-band by means of a new module header.

3.3 Software

`libpcap` is the *de facto* standard capture API but, for now, `libpcap` applications cannot yet directly be used with our monitoring solution. Simple packet recorders should work, but the 8-byte timestamp prepended to each packet will confound protocol analysis.

Instead, we include a `libpcap`-based capture programme which converts and removes the hardware timestamp, overwrites the PCAP timestamp, and records a standard PCAP trace. It is our ambition to support a live PCAPng-style capture interface, as well as `libtrace` [2] (and thereby Endace [1] ERF format).

We also provide auxiliary command-line tools for TCAM rule management, and the initialisation of the hardware timestamp.

4. PRELIMINARY RESULTS

We characterised the latency through the NetFPGA with an Endace DAG 4.3ge SX, as shown in Figure 5. Being optical, we were obliged to use a pair of media converters (Allied-Telesyn AT-MC-1004), and we didn't have the means at our disposal to calibrate out the latency contributed by these devices.

We measured latency through the two converters and the NetFPGA card at a constant 2.4 μs, irrespective of whether the test packets matched a filter rule, or how many rules were programmed into the filter.

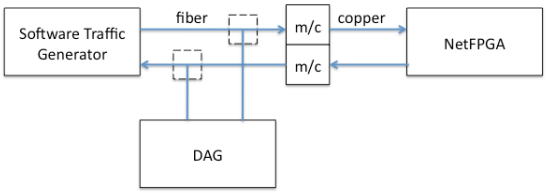

Figure 5: Latency measurement apparatus

At the time of writing, we are setting up an experiment in which we will test the quality of timestamps returned to the host against the DAG card.

5. CONCLUSION AND FUTURE WORK

We present a flexible and cheap passive NetFPGA-based monitoring system. Our proof-of-concept implementation so far shows promising results.

Further development and rigorous quality evaluation against the respected industry standard Endace DAG card are on-going. We identify a number of extensions and enhancements, many of which the implementation is underway:

- Use of Direct Digital Synthesis, together with an external time-base to provide error-corrected timestamps in a convenient format;
- Re-implementation of the flow filter using Bloom filters in order to support substantially more than 32 flows;
- Optional in-band markers for packets belonging to unmatched flows;
- Refactor to include timestamp in a module-header;
- Live `libpcap` support with extended precision timestamps; and
- Endace ERF format and `libtrace` support.

The NetFPGA is a promising platform on which to develop exciting new, low-cost instrumented devices. For example, whereas by conventional techniques, only link-level behaviour can be instrumented, the NetFPGA monitoring platform offers the possibility of instrumentation of router behaviour right at the routing plane.

6. REFERENCES

[1] Endace. `http://www.endace.com`.
[2] libtrace. `http://research.wand.net.nz/software/libtrace.php`.
[3] nTop network traffic probe. `http://www.ntop.org`.
[4] Scampi project. `http://www.ist-scampi.org`.
[5] Wireshark protocol analyzer (was ethereal). `http://www.wireshark.org`.
[6] L. Deri. Improving passive packet capture: Beyond device polling. In *SANE 2004*.
[7] L. Deri. Passively monitoring networks at gigabit speeds using commodity hardware and open source software. In *PAM 2003*.
[8] L. Deri. nCap: Wire-speed packet capture and transmission. In *End-to-End Monitoring*, May 2005.
[9] S. F. Donnelly. *High precision timing in passive measurements of data networks*. PhD thesis, University of Waikato, 2002.
[10] D. Ficara, S. Giordano, F. Oppedisano, G. Procissi, and F. Vitucci. A cooperative pc/network-processor architecture for multi gigabit traffic analysis. In *QoS-IP 2008*.
[11] N. R. G. Lawrence Berkeley National Labs. tcpdump, libpcap. `http://www.tcpdump.org`.
[12] P. Saul. Direct digital synthesis. *Circuits and systems tutorials*, page 393, 1996.
[13] T. Wolf, R. Ramaswamy, S. Bunga, and N. Yang. An architecture for distribuited real-time passive network measurement. In *MASCOTS 2006*.