Profiling Per-Packet and Per-Byte Energy Consumption in the NetFPGA Gigabit Router

Vijay Sivaraman†, Arun Vishwanath†, Zhi Zhao†, Craig Russell‡
†School of Electrical Engineering & Telecommunications, University of New South Wales, Sydney, Australia
Emails: {vijay@unsw.edu.au, arunv@ee.unsw.edu.au, zhi.zhao@student.unsw.edu.au}
‡CSIRO ICT Centre, Sydney, Email: {craig.russell@csiro.au}

Abstract— **Improving energy efficiency of Internet equipment is becoming an increasingly important research topic, motivated by the need to reduce energy costs (and Carbon footprint) for Internet Service Providers, as well as increase power density to achieve more switching capacity per-rack. While recent research has profiled the power consumption of commercial routing equipment, these profiles are coarse-grained (i.e., at the granularity of per line-card or per port), and moreover such platforms are inflexible for experimentation with new energy-saving mechanisms. In this paper we therefore consider the NetFPGA platform, which is becoming an increasingly popular routing platform for networking research due to its versatility and low-cost. Using a precise hardware-based traffic generator and high-fidelity energy probe, we conduct several experiments that allow us to decompose the energy consumption of the NetFPGA routing card into fine-grained per-packet and per-byte components with reasonable accuracy. Our quantification of energy consumption on this platform opens the doors for estimating network-wide energy footprints at the granularity of traffic sessions and applications (e.g., due to TCP file transfers), and provides a benchmark against which energy improvements arising from new architectures and protocols can be evaluated.**

I. Introduction

Internet traffic has witnessed exponential growth over the past decade, currently in the Exabyte range (10^{18} bytes) per year, and projected to reach Zettabytes (10^{21} bytes) in the next 5 years [1]. To cope with such high traffic demand, Internet Service Providers (ISPs) deploy routers that can today switch data at hundreds of gigabits-per-second, drawing tens of KiloWatts of power [2]. Not only does this power account for a significant fraction of the ISP's operational expenses, but also the high power density necessitates complex cooling systems to manage heat dissipation. Though routing equipment is becoming more power efficient, the increase in efficiency is outpaced by annual increase in throughput capacity [3], meaning that the problem is likely to worsen with time. In recognition of the pressing need to address this problem, a consortium called GreenTouch [4] has recently been launched that aims to improve the ICT sector's energy efficiency by a factor of 1000 from current levels by 2015.

Early works such as [5] highlighted energy efficiency issues in wireline networks, and several recent works have proposed techniques for power-optimising network design [6], and saving energy by putting ports and line-cards to sleep or via adapting link rate [7]. Models of device energy consumption have been put forth [8] using measurements on commercial switches and routers. In this paper, we focus instead on the NetFPGA experimental Gigabit routing platform. Our reasons for choosing this platform are twofold. **First**, commercial routers do not offer sufficient flexibility (or mechanisms) for experimenting with new power-optimised architectures and algorithms, whereas the NetFPGA router is an open reprogrammable hardware platform that is increasingly being used by networking researchers world-wide to rapidly prototype and evaluate new mechanisms. By benchmarking energy performance of this platform, it becomes possible to quantify the energy gains from new mechanisms for improving power efficiency. **Second**, commercial platforms provide only coarse-grained measurements of power (at the granularity of device, line-card or port), whereas in this paper we focus on obtaining *fine-grained* energy measurements of per-packet and per-byte energy, which typically cannot be measured with high confidence in commercial routers today.

Our main contribution in this paper is to isolate the energy components associated with the various operations in the NetFPGA router. We use a high-precision hardware-based traffic generator and analyser that enables us to tightly control the stimulus to the router. In conjunction, we use a high-fidelity multi-channel digital oscilloscope that probes and records the instantaneous power draw of the NetFPGA router card 50 million times-per-second. Using these, we devise a series of experiments that allow us to quantify the per-packet processing energy, per-byte energy for receipt and storage at the ingress to the router, as well as per-byte energy for queueing and transmission at the egress of the router. To the best of our knowledge, ours is the first work (barring our own preliminary abstract [9]) that gives such a fine-grained profile of energy consumption on the NetFPGA platform, which will be of great value to the community. This profile opens the doors to deducing network-wide energy footprints at various levels such as traffic sessions, applications, or user-groups. Researchers who prototype new architectures (e.g., dynamic speed scaling or sleeping) and protocols (e.g., power-aware TCP) for energy-savings in the NetFPGA routing platform can use our profile as a benchmark to quantify the improvements their mechanisms can realise.

The rest of the paper is organised as follows. In Section II, we describe the NetFPGA Gigabit router, outline the life of a packet through the router, and present a simple linear model for the power consumption. Our main contribution is in Section III that isolates the energy components based on a series of experiments, and discusses its implications. The paper is concluded in Section IV with pointers to directions for future work.

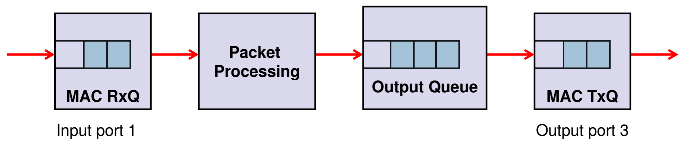

Fig. 1. Life of a packet through the NetFPGA router

II. The NetFPGA Gigabit Router

The NetFPGA is a PCI-based experimental platform developed by Stanford University, which consists of a fully programmable Xilinx FPGA based core with four Gigabit Ethernet interfaces, and functions as an IP router. Like commercial routers, it handles all the packet-processing tasks in *hardware* (i.e., longest-prefix matching, ARP lookups, etc.). The NetFPGA has been used extensively in numerous research studies, such as for router buffer sizing, clean-slate network architecture and design, etc. Additional information can be obtained from the NetFPGA website [10].

A. Life of a Packet Through the NetFPGA Router

The experimental work in this paper uses the NetFPGA revision 2 board along with the standard reference router gateware (*reference_router.bit*) taken from [10]. We use the reference router bit-file as supplied, since that is the base distribution most researchers will build upon. Needless to say, modifying the architecture (for example to store packets in DRAM rather than SRAM) would doubtless change the energy performance. We have not optimised the code for energy-efficiency, and leave that for future work.

Fig. 1 shows a high-level view of the life of a packet through the NetFPGA running the reference router. Consider a packet *received* on port 1 via the Ethernet MAC receive queue (MAC RxQ). The received packet is stored in the Xilinx FPGA's on-chip memory while the output port lookup module inside the FPGA performs the following *packet processing* steps: (1) it does a longest prefix match on the routing table to decide the output port to which the packet must be switched to (in this case port 3), (2) it obtains the next-hop MAC address by performing an ARP lookup, (3) it modifies the source and destination MAC addresses in the packet header, the TTL (time-to-live) is decremented, and the checksum is updated, and, (4) it buffers the packet in the respective output queue. In the reference router, the off-chip SRAM memory serves as the output queue.

When it is time for the packet to be dequeued, the FPGA logic extracts it from the off-chip SRAM memory and feeds it to the Ethernet transmission queue (in this case MAC TxQ port 3) for it to be sent on the wire.

B. A Simple Linear Model of Power Consumption

We use the following simple linear model of power consumption P of the NetFPGA router. When the card is powered on, but has no Ethernet ports connected (and hence no traffic), it consumes a constant power P_C. Each Ethernet port that is connected (i.e., link is up) but has no traffic flowing through it consumes an additional constant power P_E. Each packet that enters the router needs to be processed (parsing, route lookups, ARP lookup, etc.), and is assumed to require energy E_p that is independent of the packet size. Each byte of the incoming packet needs energy to be received (E_{rx}) and stored (E_{rs}) while it is processed; if the packet is not dropped, it needs further energy to be stored in the output queue (E_{ts}) and thereafter transmitted (E_{tx}). This simple linear model can be formally expressed as below:

$$P = P_C + KP_E + N_I E_p + R_I(E_{rx} + E_{rs}) + R_O(E_{ts} + E_{tx}) \quad (1)$$

- P_C is a constant base-line power consumption of the NetFPGA card (without any Ethernet ports connected),
- $K \in [0, 4]$ is the number of Ethernet ports connected,
- P_E is the power consumed by each Ethernet port (without any traffic flowing),
- N_I is the input rate to the NetFPGA card in packets-per-second (pps),
- E_p is the energy required to process each packet (parsing, route lookup, etc.),
- R_I is the input rate to the NetFPGA card in bytes-per-second,
- R_O is the output rate from the NetFPGA card in bytes-per-second,
- E_{rx} is the energy required to receive a byte on the ingress Ethernet interface,
- E_{rs} is the energy required to process/store a byte on the ingress Ethernet interface,
- E_{ts} is the energy required to store/process a byte on the egress Ethernet interface, and
- E_{tx} is the energy required to transmit a byte on the egress Ethernet interface.

For a byte of a packet that does not get dropped, we find it convenient to define the term $E_b = E_{rx} + E_{rs} + E_{tx} + E_{ts}$ to denote the total per-byte energy; this term simplifies notation when the input rate R_I and output rate R_O are equal in Eq. (1).

III. Estimating Per-Packet and Per-Byte Energy Components

Under the assumption that the power consumption of the NetFPGA Gigabit platform running the standard reference router follows the model proposed above, the objective of this section is to determine each of the components in Eq. (1). In the following subsections we first outline our experimental setup and methodology, and then describe in detail each experiment that helps us estimate specific components of the energy model. We note that the results presented here expand vastly on our earlier abstract [9]: we repeat our earlier experiments for a larger parameter set to obtain higher confidence in the results, and additionally conduct new experiments to decompose the per-byte energy into the receive and transmit side components.

A. Experimental Setup

The setup (pictured in Fig. 2) consists of three components: the NetFPGA router itself, a high-precision traffic generator, and a high-fidelity oscilloscope for power measurement, each described in turn next:

Router: We use a NetFPGA revision 2 board having four 1 Gbps Ethernet ports. The gateware on our NetFPGA is the supplied reference router (*reference_router.bit*, v1.0 Beta,

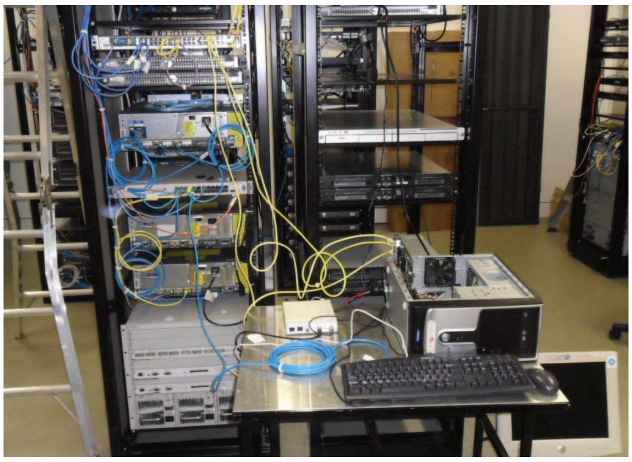

Fig. 2. Experimental setup comprising of the NetFPGA router, riser card, oscilloscope, and traffic generator

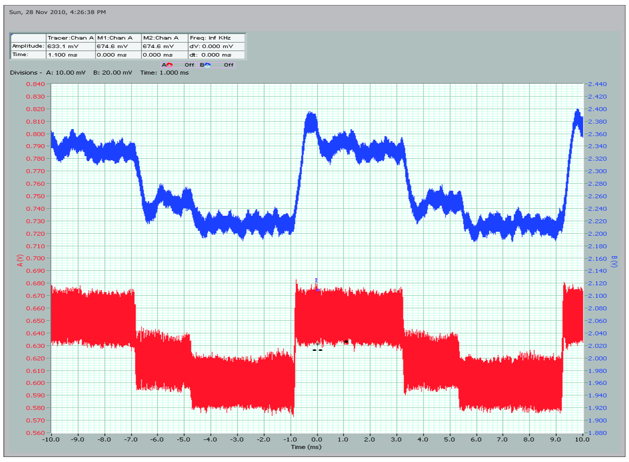

Fig. 3. Oscilloscope output showing waveform of current draw on 3.3V (top) and 5V (bottom) supply

build date 4 Jul 2009) taken from the NetFPGA website; note that we do not make any modifications to the gateware (for energy optimisation or otherwise), since researchers are likely to build upon the given base distribution. The NetFPGA card is housed in a desktop PC with a 3 GHz AMD Athlon processor with 2 GB memory, running CentOS version 5.2, along with Scone (the NetFPGA control software).

Traffic Generator: We use the IXIA [11], which is a high-precision commercial-grade hardware traffic generator with sophisticated capabilities for configuring traffic profiles, synchronising traffic on multiple ports, and accumulating statistics based on pattern filters. We will see further in this section how these capabilities are used to isolate the various energy components. We connected three Gigabit ports of the IXIA to three ports of the NetFPGA (the fourth port could not be connected due to a physical barrier presented by the power measurement apparatus, as described next). In our experiments Ethernet port 1 and 2 on the NetFPGA were typically the ingress ports, while port 3 was the egress feeding back to the IXIA.

Power Measurement: Early in our experimentation we noticed that the power consumed by the host PC (including the NetFPGA card) had a wide range of fluctuation (75 ± 5 Watts) even when there was no network traffic. These fluctuations most likely arise from mechanical components in the PC such as fans, disks, etc., and can mask the fluctuations we want to measure – namely changes in power drawn by the NetFPGA card as traffic patterns change. In order to accurately isolate the power consumed by the NetFPGA card alone, we therefore mounted the NetFPGA card on an Ultraview PCI Smart Extender PCIEXT-64U riser card [12], which is in turn inserted into the PCI slot of the host PC. The riser card has break-out pins for measuring current draw on the several voltage supply pins (3.3V, 5V and 12V) to the NetFPGA card. The current measurement pins (for the 3.3V and 5V supplies only, since the NetFPGA does not use the 12V supply) were connected via leads to a two-channel USB digital oscilloscope, specifically the Cleverscope CS328A [13], which samples the current draw every 20 nanoseconds and records them to a file (Fig. 3 shows a snapshot of the visual output). Each run of each experiment described below collected 100 million samples of the instantaneous current draw (on each supply voltage) to determine the average power consumption, and we typically performed ten runs for each experiment.

B. Baseline Power P_C

We measured several times over the day the baseline power consumed by the NetFPGA card, namely when none of the Ethernet ports are connected. On average the baseline power was $P_C = 6.936$W. However, we noticed that the standard deviation was quite large at about 67mW. In particular, we noted that as the NetFPGA operated continuously (routing traffic) for a few hours, it grew hotter and the baseline power consumption drifted. We believe this is because power consumption is affected by factors such as leakage current that vary with temperature. Since temperature can vary between our experimental runs, we minimise its impact on our estaimtions by concentrating on power differentials (i.e., slope of the power curve) within an experiment rather than using the absolute numbers themselves, as described in more detail further on.

C. Ethernet Per-Port Power P_E

Starting from the baseline above (with no ports connected) we successively connected each Ethernet port (to get link up but with no traffic flowing) and measured the power consumption of the NetFPGA card. The increase in power was almost perfectly linear, with each Ethernet port adding $P_E = 1.102$W. This must come from the PHY and MAC components that activate the link, perform the carrier sensing, check for an incoming preamble, etc. In all our subsequent experiments three of the NetFPGA Ethernet ports are connected, and the baseline power consumption for this 3-port configuration was found to be $P_C + 3P_E \approx 10.242$W.

D. Per-Packet Processing Energy E_p

Our next experiment is designed to estimate E_p, the incremental energy cost of processing each packet, which includes the energy required to parse the packet (to extract Ethernet/IP headers) and to perform route lookups. This energy cost is incurred on a per-packet basis, and should not depend on

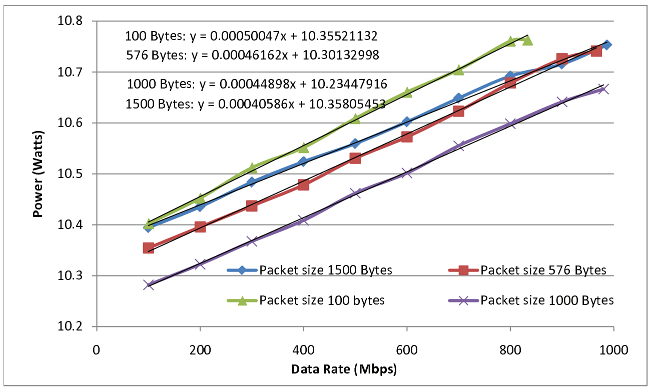

Fig. 4. Power consumption versus data rate for fixed packet size

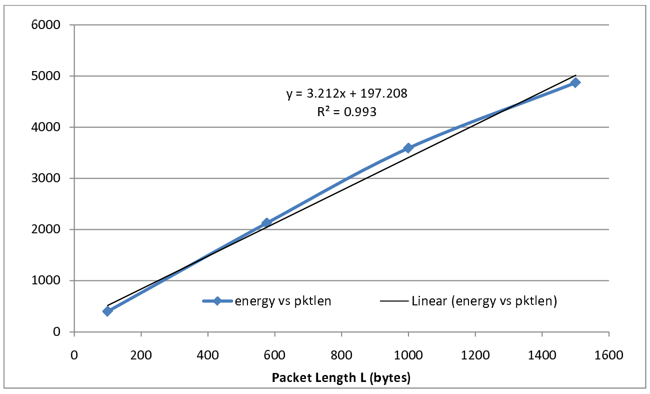

Fig. 5. Fitting data to estimates of energy components

packet size. Our approach is as follows: We fix the packet size (to say L Bytes), and send a constant-rate stream of packets from the IXIA to the NetFPGA on port 1, which gets routed out of port 3 back to the IXIA at a constant rate. There are no packet drops or losses. The experiment is repeated for data rates of 100Mbps, 200Mbps, and so on till the maximum line-rate (close to but not exactly equal to 1Gbps due to inter-packet gaps). The whole process is repeated for several choices of fixed packet size to get more confidence in our estimates.

Fig. 4 shows how the power consumption P of the NetFPGA card changes as a function of data rate R_I for fixed packet size of $L = \{100, 576, 1000, 1500\}$ Bytes. Our first observation from the figure is that the power consumption varies by nearly 400mW as the traffic rate is varied from 10% to 100% of line-rate, which represents an increase of only about 4% on the base-line power consumption of 10W when the NetFPGA router is powered on (with 3 connected ports) but has no traffic. This is in-line with observations made by other researchers (e.g., [6]) that commercial routers consume most of their power just to be on, and the impact of traffic load on power consumption is relatively small. Nevertheless, there are many ongoing research efforts to make the power consumption of chips, devices and systems proportional to their workload, and our study aids their research effort by helping understand the fine-grained dependence of energy on traffic. Our second observation from the figure is that each curve (corresponding to one value of packet size L) is well approximated by a linear fit, also shown in the figure. This lends credence to the linear energy model in Eq. (1).

We now see how this experiment helps us estimate the per-packet energy component. We begin by taking the partial derivative of Eq. (1) with respect to the input rate R_I, noting that the input packet rate $N_I = R_I/L$ where L is the fixed packet size in Bytes, and that $R_O = R_I$:

$$\partial P/\partial R_I = E_p/L + E_b \tag{2}$$

where we have used $E_b = (E_{rx} + E_{rs}) + (E_{tx} + E_{ts})$ to denote the overall per-byte energy. For a given value of packet size L, the left side of the above equation, namely $\partial P/\partial R_I$, can be deduced from the slope of the corresponding curve in the figure. Specifically, if the curve for packet size L is represented by a straight line of the form $m_L x + c_L$, then $\partial P/\partial R_I = 8m_L$ (the multiplier of 8 accounts for the fact that R_I is in bytes-per-second while the figure shows data rate in bits-per-second). At this point we note that the figure shows that the slope m_L decreases with packet size L, confirming that for the same increase in data rate, smaller packets incur a larger incremental energy overhead. We also note that we choose not to use the intercepts c_L since the baseline power consumption was not found to be very stable (as explained earlier in Section III-B) over the several hours we needed to do the various sets of experiments.

Eq. (2) can then be rewritten as the family of equations:

$$8 * L_j * m_{L_j} = E_p + L_j * E_b, \qquad j = 1, 2, 3, 4 \tag{3}$$

where $L_j = \{100, 576, 1000, 1500\}$ denotes the four packet sizes in our experiments, with corresponding slopes $m_{L_j} = \{0.00050047, 0.00046162, 0.00044898, 0.00040586\}$ as shown in the equations for the best linear-fit in Fig. 4. We solve for the energy components E_p and E_b by finding the best-fit to the equations. Specifically, we treat the above equations to be of the form $y = a + bx$, where y corresponds to the left side value in each of the equations and x is the packet length value. We plot the (x, y) values in Fig. 5, and see that the data points fit almost perfectly (with $R^2 = 0.993$) to a straight line. The slope of this straight line corresponds to the per-byte energy $E_b \approx 3.212$nJ, while the intercept gives us the per-packet processing energy $E_p \approx 197.208$nJ.

E. Per-Byte Total Energy E_b

Though our previous experiment also gave us an estimate for the per-byte energy $E_b = (E_{rx} + E_{rs}) + (E_{tx} + E_{ts})$, in this experiment we directly deduce the per-byte energy as follows: our setup still has one traffic stream that enters the NetFPGA on one port (port 1) and egresses on another port (port 3), and there are no losses (so $R_I = R_O$). This time we fix the packet rate N_I, and vary the packet size (from 64 up to 1500 bytes) in order to vary the data rate. We perform our experiment for four packet rates: $N_I = 20, 40, 60, 80$ Kpps, and the corresponding power consumptions of the NetFPGA card as a function of the packet size are shown by the four curves in Fig. 6. Once again we note that each curve is roughly linear, and the slope and intercept of the best linear-fit are also depicted in the figure.

To deduce the per-byte energy, we take the partial derivative of Eq. (1) with respect to packet length L, noting that

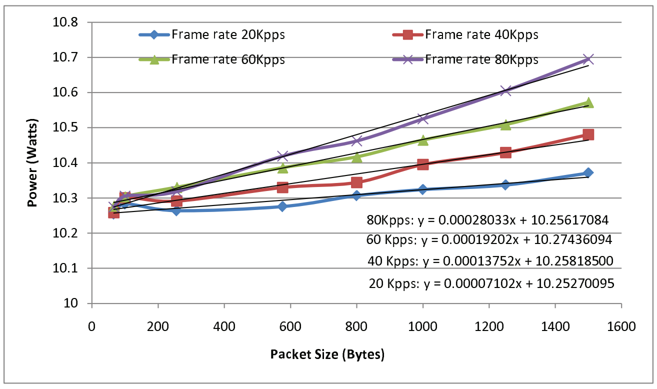

Fig. 6. Power consumption versus packet size for fixed packet rate

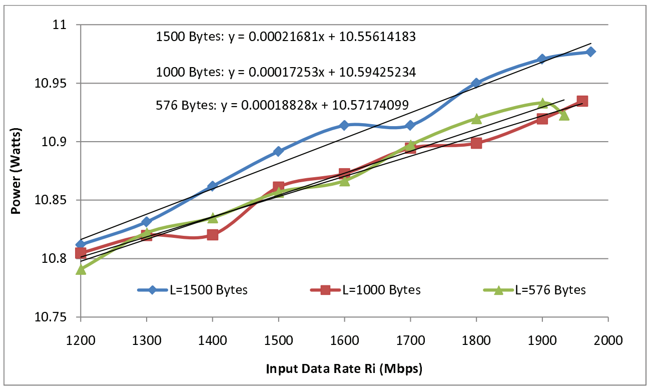

Fig. 7. Power consumption versus input data rate for fixed output rate

N_I is a constant and $R_I = R_O = LN_I$:

$$\partial P/\partial L = N_I * E_b \quad (4)$$

where $E_b = (E_{rx} + E_{rs}) + (E_{tx} + E_{ts})$ denotes the total per-byte energy. Estimating the slope of each curve in Fig. 6 therefore yields the left side of the equation above, which when divided by the packet rate N_I directly yields an estimate of the per-byte energy E_b. For the four curves corresponding to frame rates of $20, 40, 60, 80$ Kpps, using the slopes of the best linear-fit from the figure, we obtain estimates of E_b as $3.551, 3.438, 3.200, 3.504$nJ, which are reasonably consistent, yielding an average estimate $E_b = 3.423$nJ. This is also fairly consistent with the estimate of 3.212nJ we had obtained for E_b in the previous subsection.

F. *Per-Byte Receive-Side Energy* $E_{rx} + E_{rs}$

In this experiment we try to independently deduce the per-byte energy $E_{rx} + E_{rs}$ at the receive (ingress) side, which includes the energy for receiving the bytes and storing them while the packet headers are being processed and a routing decision is being made. Our methodology for estimating the receiver-side energy is as follows: we oversubscribe the output link (port 3) of the NetFPGA by feeding in traffic on two input links (ports 1 and 2). In each experiment, the packet size is fixed, and the input rate is varied such that the output link stays oversubscribed (thus the output traffic rate is fixed). Consequently, a known fraction of packets received by the NetFPGA get dropped, and in the process incur energy cost at the receiver but not at the transmitter side. This lets us isolate the receiver-side energy.

More specifically, we fix packet size at each of three values $L = \{576, 100, 1500\}$ bytes. The total input rate (from two input ports) varies from 1.2 to 2 Gbps, while the output stays saturated at 1 Gbps (less the inter-packet gaps). Fig. 7 shows how the power of the NetFPGA card varies with input data rate R_I for three settings of packet size L. To estimate the receiver-side per-byte energy $E_{rx} + E_{rs}$, we take the partial derivative of Eq. (1) with respect to R_I, bearing in mind that the output data rate R_O and the packet size L are constants, while the input packet rate $N_I = R_I/L$:

$$\partial P/\partial R_I = E_p/L + (E_{rx} + E_{rs}) \quad (5)$$

For each of the three packet sizes used in this experiment, the value of the left side of the above equation is deduced from the slope of the best linear-fit of the corresponding curve in the figure. Using the value of the per-packet processing energy E_p from Section III-D, we deduce that $E_{rx} + E_{rs} \approx 1.317$nJ.

We are not experimentally able to isolate the individual components E_{rx} and E_{rs}, as we do not see a way by which the NetFPGA card can be made to receive the byte without storing it (short of modifying the gateware to explicitly do so, which is beyond the scope of this paper). However, other studies such as [14] have reported that a commercial Ethernet PHY operating at 1 Gbps consumes around 62mW for transmit and receive, which translates to per-byte receive energy of $E_{rx} = 0.496$nJ. Using this value, we can then estimate that the energy of storing a byte in the ingress side of the NetFPGA is $E_{rs} \approx 1.317 - 0.496 = 0.821$nJ.

G. *Per-Byte Transmit-Side Energy* $E_{ts} + E_{tx}$

In this experiment we try to directly deduce the transmit-side per-byte energy, associated with storing the byte in the output queues E_{ts} and transmitting the byte on the output link E_{tx}. To do so, we keep the input rate to the NetFPGA fixed, while trying to vary the output rate. This actually turns out to be quite tricky, and is achieved as follows: we send synchronised bursts of packets from the IXIA to the two input ports of the NetFPGA (ports 1 and 2). Each burst consists of 500 back-to-back packets of size 1000 bytes sent at line-rate (this takes about 4 msec), followed by an off-period (no packets sent) for 6 msec. The bursts are perfectly synchronised on the two input ports, so that when both ports start sending packets (for an on-period of about 4 msec), the output link is oversubscribed and the queue builds up, and when both sources stop (for an off-period of 6 msec), the queue drains out, and the entire process repeats. By adjusting the buffer size on the output link, we can vary the number of packets dropped in each on-off cycle, thereby controlling the output rate, which is measured by the IXIA. This setup therefore lets us control the output rate for fixed input rate, which helps us estimate the transmit-side per-byte energy as follows: taking partial derivates of Eq. (1) with respect to the output rate R_O we get:

$$\partial P/\partial R_O = (E_{ts} + E_{tx}) \quad (6)$$

since the input packet rate N_I and data rate R_I are fixed.

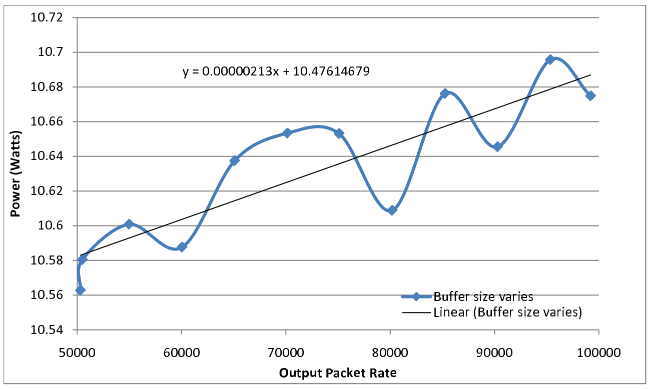

Fig. 8. Power consumption versus output packet rate for fixed input rate

Fig. 8 shows how the power consumption of the NetFPGA card varies as a function of output packet rate (packet size was fixed at 1000 bytes and input packet rate was fixed at $49,603.65$ pps), achieved by varying buffer size from 3 KiloBytes to 500 KiloBytes. The curve has a small range (less than 140mW), and is quite noisy, with $R^2 \approx 0.75$ for a linear fit. Nevertheless, the slope of the curve (when converted to units of bytes rather than packets) directly yields an estimate of the transmit-side per-byte energy $E_{ts} + E_{tx} = 2.130$nJ. This is reasonably close to the estimate $E_b - (E_{rx} + E_{rs}) = 3.423 - 1.317 = 2.106$nJ from our earlier results, and thus provides independent confirmation of the accuracy of our estimates.

Once again, separating E_{ts} from E_{tx} is not possible experimentally without modifying the gateware (beyond the scope of this paper), but we again depend on the estimate of $E_{tx} \approx 0.496$nJ from earlier work [14]. This lets us estimate that $E_{ts} \approx 1.61$nJ. We note that the storage energy E_{ts} on the transmit side is almost twice the storage energy E_{rs} on the receive side, which we believe is because the ingress queues are on-chip on the NetFPGA, whereas the transmit queues are off-chip in the external SRAM.

H. Summary and Discussion

Table I summarises our results on the decomposition of the energy costs in the NetFPGA Gigabit router. We qualify our results with the following notes:

- Fully loading the NetFPGA router with traffic causes its power consumption to increase by a mere 5% over being idle. However, ongoing research in "work proportional" devices and components with very low idling energy will likely amplify the relative impact of traffic load on router power consumption.
- Our experiments had only a handful of routing table entries configured, and it is likely that having hundreds of thousands of routing entries could increase the per-packet processing energy. However, we are limited in the NetFPGA hardware's current ability to store at most 32 routing table entries.
- The ouput queue in the NetFPGA standard reference router is currently implemented in SRAM. Modifying the gateware to use the DRAM for output queueing will likely alter the per-byte power consumption for storage at the egress.

Energy component and description	Our estimate
Power consumed by unconnected NetFPGA card (P_C)	6.936 W
Power consumed per connected Ethernet port (P_E)	1.102 W
Per-Packet processing energy (E_p)	197.2 nJ
Per-Byte energy (E_b)	3.4 nJ
Per-Byte Ingress storage energy (E_{rs})	0.8nJ
Per-Byte Egress storage energy (E_{ts})	1.6 nJ
Per-Byte transmit/receive energy (E_{tx}, E_{rx})	0.5 nJ

TABLE I
SUMMARY OF NETFPGA POWER PROFILE

Not withstanding the above caveats, we believe our work provides valuable information to the research community on the incremental energy costs of switching traffic through the NetFPGA router. We believe our work can be used in at least two ways: to compute network-wide energy footprints of TCP sessions, file transfers, user activity patterns, etc., and to evaluate the performance of new architectures and protocols (such as processor speed scaling, energy-aware transport protocols, etc.) in terms of their energy improvements.

IV. CONCLUSIONS AND FUTURE WORK

To support the increasing research activity in improving the energy-efficiency of Internet routing equipment, in this paper, we experimentally derived fine-grained quantitative estimates of the per-packet and per-byte energy costs in the NetFPGA 4-port Gigabit routing platform. Our study informs the research community on the relative energy costs of the various components in the router, and provides a benchmark against which energy performance of new proposals can be compared. Our future work will extend the study to compare it to other platforms, and study the impact of factors (such as routing table size) and features (e.g., access control lists) on the energy costs.

REFERENCES

[1] Cisco-Systems, "Approaching the Zettabyte Era," www.cisco.com/en/US/solutions/collateral/ns341/ns525/ns537/ns705/ns827/white_paper_c11-481374.pdf, Jun 2008.
[2] ——, "Carrier Routing System (CRS-1)," www.cisco.com/en/US/prod/collateral/routers/ps5763/prod_brochure0900aecd800f8118.pdf.
[3] R. Tucker, J. Baliga, R. Ayre, K. Hinton, and W. Sorin, "Energy Consumption of IP Networks," in *Proc. European Conference on Optical Communications*, Brussels, Belgium, Sep 2008.
[4] GreenTouch, www.greentouch.org.
[5] M. Gupta and S. Singh, "Greening the Internet," in *Proc. ACM SIGCOMM*, Karlsruhe, Germany, Aug 2003.
[6] J. Chabarek *et al.*, "Power Awareness in Network Design and Routing," in *Proc. IEEE INFOCOM*, Phoenix, AZ, USA, Mar 2008.
[7] S. Nedevschi *et al.*, "Reducing Network Energy Consumption via Rate-Adaptation and Sleeping," in *Proc. USENIX NSDI*, San Francisco, CA, USA, Apr 2008.
[8] P. Mahadevan *et al.*, "A Power Benchmarking Framework for Network Devices," in *Proc. IFIP Networking*, Aachen, Germany, May 2009.
[9] A. Vishwanath *et al.*, "An Empirical Model of Power Consumption in the NetFPGA Gigabit Router," in *Proc. IEEE Advanced Networks and Telecommunication Systems*, Mumbai, India, Dec 2010.
[10] "NetFPGA Gigabit Router," www.netfpga.org.
[11] "Ixia traffic generator," www.ixiacom.com.
[12] "Ultraview PCI Smart Extenders," www.ultraviewcorp.com/displayproduct.php?part_id=4&sub_id=2.
[13] "Cleverscope oscilloscope," www.cleverscope.com/products/cs328a.php.
[14] Y. Chen and R. H. Katz, "Energy Efficient Ethernet Encodings," University of California, Berkeley, Tech. Rep. UCB/EECS-2009-68, 2009.