Power/Performance evaluation of Energy Efficient Ethernet (EEE) for High Performance Computing

Karthikeyan P. Saravanan, Paul M. Carpenter, Alex Ramirez
Barcelona Supercomputing Center - Centro Nacional de Supercomputacion (BSC-CNS)
Universitat Politècnica de Catalunya (UPC), Barcelona, Spain
Email: {karthikeyan.palavedu,paul.carpenter,alex.ramirez}@bsc.es

***Abstract*—As of June 2012, 41% of all systems in the TOP500 use Gigabit Ethernet. Ethernet has been a strong contender in the HPC interconnect market for its competitive performance and low cost. However, until recently, little emphasis has been thrown upon bringing about energy efficient HPC interconnects. To illustrate, in a majority if not all Ethernet based systems, the transmitter and receiver operate at full power regardless of any data transmission between them, leading to power inefficiency. The recent standard IEEE 802.3az, Energy Efficient Ethernet (EEE), approved in 2010, solves the above conundrum by introducing "Low-Power-Idle", dynamically turning off unused links to save interconnect power.**

In this paper, we present the first analysis of Energy Efficient Ethernet in the domain of HPC, examining its potential for power savings. Unlike previous proposals, we present a detailed analysis of the impact of additional latency overhead introduced by EEE, using multiple simulated systems running actual HPC application traces. We propose the use of "Power-Down Threshold", as a possible add-on to EEE to mitigate its on/off transition overhead. We find that EEE brings about link power savings of about 70% by switching off links, but at the cost of performance, leading to increased power consumption of the overall system by 15% (average). In contrast, using our proposed "Power-Down Threshold", we demonstrate reduced on/off transition overhead, from 25% to 2%, translating to overall system power savings of about 7.5%. Furthermore, in this work we point out relevant design decisions for future vendors intending to deploy EEE solutions for their HPC systems.

Keywords*—*Energy Efficient Ethernet (EEE), Low-Power Interconnects, Low-Power-Idle (LPI), Power-Down Threshold, On/Off Networks

I. Introduction

Energy efficiency has become the most important design criterion for modern supercomputers. Today's most energy efficient supercomputer (as of June 2012, at 2.1 GFlops/W [4]), if extrapolated to a sustained ExaFlop performance, would have a total power consumption of 500 MWatts. In contrast, US DOE, DARPA and other exascale HPC programs target the deployment of an ExaFlop machine by 2018 with power consumption of a mere 20 MWatts [9, 12, 13]. Building future supercomputers at such stringent power budgets requires eliminating every inefficiency throughout the system. Much effort in bringing about energy efficiency and energy proportionality to systems, has gone into optimizing the compute elements and memory, which contribute to the majority of the system power consumption. However, recent trends show an increased emphasis on bringing about energy efficiency in interconnects. With compute nodes being power optimized and energy proportional, the interconnection network power is becoming increasingly significant. Power consumption of HPC interconnects can contribute to up to 30% of the overall power consumption of the system [10, 14].

As with supercomputers, reducing interconnect power has become a pressing issue for Internet and IT infrastructure as well as servers and high-end data centers. Studies suggest an annual power consumption of 6 TWh consumed by networking devices alone in the US, and this figure is expected to further increase [23]. Since Ethernet is the dominant interconnect technology in commercial and IT infrastructure, improvements in its energy efficiency are estimated to bring about energy savings of over 3 TWh [23].

Interconnection links can be attributed to consuming a substantial portion of the total interconnect power. Links take up 64% of the power budget of the IBM Infiniband 8-port 12X switch [8, 16] and 63%, 65% of the Dell PowerConnect 5324 and 6248 respectively [16]. In addition, conventional network links are essentially always on, thereby dissipating power, regardless of whether or not data is being transmitted. The average power consumption of the IBM Infiniband 12X link, for example, is almost identical to its worst-case power [16, 20]. In the case of Ethernet, its design require both the transmitters and receivers to operate continuously to keep the link aligned. In order to mitigate such waste, the IEEE 802.3az Energy Efficient Ethernet (EEE) standard (approved in Sept 2010) brought about energy saving schemes that make the energy consumption of the link proportional to its utilization.

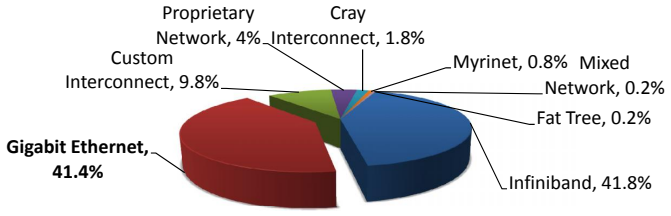

Fig. 1: System share of Interconnects in TOP500 supercomputers (June 12)

While numerous interconnects for HPC exist, Ethernet is a popular choice due to its high performance to cost. Statistics in Figure 1 obtained from TOP500[3] show that 41.4% of all supercomputers use the Ethernet family as their interconnection network. Out of the 41.4%, Gigabit Ethernet holds the majority share with 10-Gigabit Ethernet (10GbE) catching up quickly. Although the IEEE's Energy Efficient Ethernet standard is being incorporated into the network infrastructure in the areas of commercial IT, desktop, servers and high end

data centers, the question of how EEE would perform under HPC workloads requires answering.

HPC workloads are typically characterized by alternating compute and communication phases (as shown in Figure 2). These applications most often contain very large compute phases followed by relatively smaller communication phases, usually bursts of MPI messages. Further, these compute and communication phases usually have dependency patterns characteristic of the HPC applications. These dependencies may cause design decisions such as Frame buffering in EEE [20, 22, 23] to be destructive to both application performance and power (more discussed in later sections). Further, since HPC workloads in general do not overlap communication and compute periods, the time involved in communication is destructive to performance. This results in HPC applications being heavily optimized to keep communication time at its minimum, translating to low interconnect usage. However, HPC systems running these applications with conventional always-on links, do not benefit from this inherent power saving opportunity. Figure 2 shows alternating compute and communication phases of application WRF[40] along with its corresponding interconnect usage. The figure clearly shows that the interconnect links are predominantly idle. Since these links are typically always-on, continually consuming power, they would benefit greatly if built to be energy proportional.

Although EEE as a standard promises energy proportionality by design, little is known of its behavior from the perspective of HPC workloads. The biggest justification for the deployment of the EEE standard came from desktop and IT infrastructure, which remain idle for the majority of the time. EEE saves power by turning off links when they are not in use - a feature known as Low Power Idle (LPI). However, the time involved in the turning *off* and *on* links, adds additional latency to the messages transmitted. This added latency is known to be destructive to performance and power of data center workloads (whose communication is modeled as a Poisson process) and hence, frame buffering as a solution was suggested to tackle this conundrum (which we discuss in the next section). However, it is uncertain as to whether these solutions would also be beneficial to HPC.

Our contributions in this paper are as follows:

1) We conduct the first evaluation of Energy Efficient Ethernet under HPC workloads to determine the potential of EEE for energy proportional supercomputer interconnects.
2) We perform latency sensitivity analysis on HPC workloads to project performance estimates for plausible interconnect latencies determined by EEE's energy savings schemes.
3) We propose the use of *Power-Down Threshold* as a technique to reduce overhead involved with EEE's additional latencies. We experimentally compare and demonstrate how our proposed *Power-Down Threshold* scheme significantly reduces the on/off transition overheads, compared to out-of-the-box Energy Efficient Ethernet. Finally, based on our analysis we propose design recommendations for vendors intending to deploy EEE for HPC systems.

In the following sections, we discuss the background of Energy Efficient Ethernet. Following the background, we discuss related work from literature on similar proposals. We discuss our methodology and evaluate the potential of Energy Efficient Ethernet for HPC applications. We show experimental results for power and performance of Energy Efficient Ethernet and demonstrate the benefit of using our proposed Power-Down Threshold. Finally, we discuss and conclude with recommendations for the design and deployment of EEE for HPC systems.

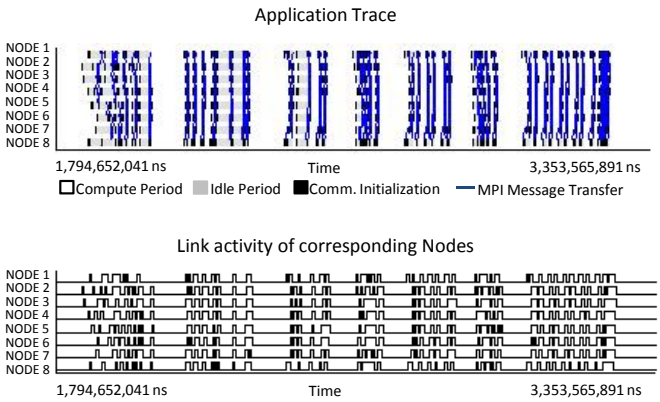

Fig. 2: Execution and Link activity pattern of WRF application. Sub-figure a) A cut of the actual application run showing execution patterns. Sub-figure b) Link activity corresponding to execution - peaks represent data transmission. Note that communication occurs in phases and are correlated.

II. Background: Energy Efficient Ethernet

In this section, we discuss the background of Energy Efficient Ethernet. In order to cut down on the power consumption costs of the Internet and data centers network infrastructures, an IEEE task force was set up in 2007 to design an IEEE standard for energy efficiency. While a number of proposals for energy efficiency were considered, Adaptive Link Rate and Low Power Idle were the most popular choices. In this section, we briefly discuss Adaptive Link Rate and Low Power Idle.

A. Adaptive Link Rate

As we mention in the previous section, interconnects in general are sparsely used. However, the power consumption of links (which takes up majority of the interconnect power) remains the same regardless of whether or not there is data transmission. This has led to studies focusing on bringing about energy consumption roughly proportional to link usage. Many initial proposals in energy proportional interconnects were concepts similar to Adaptive Link Rate (ALR) [27–30]. The idea behind ALR comes from the fact that a link's power increases with bandwidth. Example, a 100Mbps transceiver consumes about 0.25Watts, whereas 1Gbps and 10Gbps transceivers consume about 0.7Watts and 6Watts respectively [18]. Furthermore, since these links typically consume the same power when active and idle, it is beneficial to reduce the Ethernet link rate from 10Gbps link to 100Mbps link (say) during periods of inactivity. To do the above, ALR uses a process known as Auto- Negotiation[27], which changes link rate on demand.

TABLE I
LINK PARAMETERS (WAKE, SLEEP, FRAME TRANSMISSION TIME)[23]

Protocol	T_w	T_s	T_{Frame} (1500B)	Frame eff.
100Base-Tx (100Mbps)	30μs	100μs	120μs	48%
1000Base-T (1Gbps)	16μs	182μs	12μs	5.7%
10GBase-T (10Gbps)	4.16μs	2.88μs	1.2μs	14.6%

Although ALR promised large energy savings, its process of changing link rate required time in the range of milliseconds to a few seconds, which is far too large for many applications. While other alternatives to accelerate rate change were proposed [31], ALR still contained two major disadvantages. Firstly, despite the proposals to accelerate rate change, the time taken for rate change was still higher than acceptable ranges. The second disadvantage is that, although lesser power, Adaptive Link Rate still consumes power during idle periods of the interconnect. To illustrate, if a 10Gbps link is rate changed to 1Gbps during periods of inactivity, the inactive 1Gbps link consumes power, even though no data is being transmitted.

B. Energy Efficient Ethernet: Low Power Idle

To solve the above conundrum, an alternative known as Low Power Idle [2, 20] was proposed. Low Power Idle (LPI) essentially switches *off* the links during periods of inactivity and turns them back *on* when needed. The key point is that switching *on* and *off* links with LPI is relatively much faster (order of microseconds) and the link speed is not changed. This scheme was chosen to be the *de facto* standard accepted to be a part of IEEE 802.3az Energy Efficient Ethernet standard.

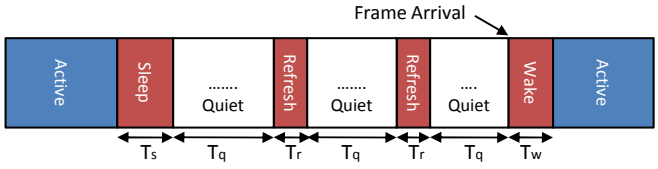

Fig. 3: State transactions between active and low-power modes in Energy Efficient Ethernet (EEE): Low Power Idle (LPI)

The Low Power Idle (LPI) mode of EEE proposes the use of "Sleep" and "Wake" modes to conserve power during periods of inactivity. Unlike complex mechanisms required to change the link speed in the case of ALR, Low Power Idle freezes the states of the transceiver when it enters the low power mode and restores it when links are powered back up. This operation can be performed in a few microseconds compared to milliseconds required for ALR [2]. Figure 3 shows a state transition example of a link that uses Low Power Idle. Here, the T_s, T_w and T_r are the time taken to put the link to sleep, wake the link and refresh the link respectively. The periodic refresh in Figure 3 is to ensure the receiver elements are aligned with the channel during low power mode.

With regard to energy efficiency, link's power consumption during T_s, T_w and T_r consumes the same power as when a link is in its *on* state and T_q consumes about 10% of the total power consumption of the link (here, $T_r << T_q$) [2, 23]. Table I shows the values for T_s, T_w and frame transmission efficiencies based on IEEE 802.3az draft [1]. It is to be noted from Table I that the sleep and wake times are relatively large for small frames. This is to show that, EEE would typically work best for large bursts of communication activity followed by large periods of inactivity.

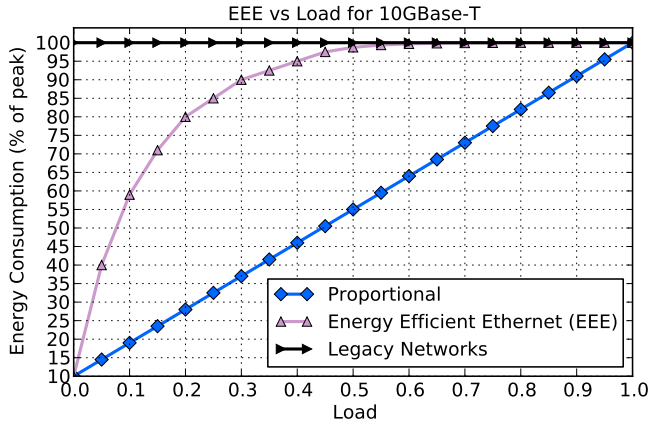

Fig. 4: Energy consumption vs load for 10Gbps Link on data center-like workload (figure reproduced from literature[23]); Figure shows how energy consumption of Energy Efficient Ethernet links quickly increases with increase in load.

P. Reviriego et al., recently published an evaluation of Energy Efficient Ethernet [23]. Their results, as shown in figure 4, suggested that power savings with EEE links decrease quickly with an increase in link utilization above zero. According to their results, when the link utilization is at 20%, the corresponding link power consumption is greater than 70%. The results obtained were for 1000-bit frames arriving into the link following a Poisson process. The poor performance is shown to be a result of large wake-up and sleep overheads compared to actual frame transmission. Essentially, a majority of the time is being spent switching *on* and *off* the links leading to decreased performance. To solve the above problem, frame buffering was proposed which holds frames (without waking up the link) up to a certain number of frames or a time-out period. With an increase in time-out periods and/or by increasing the number of frames held in buffers, the results can be made more energy proportional. However this method comes with the cost of increased packet delay leading to performance degradation of latency sensitive applications. Results show that a time-out period of 120 microsec [20, 23] is required for the link to be energy proportional with increased link utilization. In the later sections, we further analyze HPC workload for the need for frame buffering.

III. RELATED WORK

In this section, we discuss other proposals related to Energy Efficient Ethernet (EEE) and energy-proportional interconnects. Jian Li et al.,[17] proposed a similar on/off based interconnect, which makes use of system events to overlap link *on* transaction delay. Their approach requires a control network which sends messages that switch *on* the links prior to sending the message and thereby overlaps the link *on* transaction delay with the pre-processing of messages. The control network in their approach is required to be always

powered *on*, in order to receive and transmit control signals. They evaluate their methodology over three power *on* modes, each with 1μs, 2μs and 1ms as their wake-up delays over a simulated PowerPC-like 32 node machine. In our approach, we evaluate four different machines, over corresponding EEE standard defined wake-up and sleep delays, their effect on power and performance. Our model uses power estimates and energy efficiency schemes as proposed by EEE which does not assume the need for extra control networks. Alonso et al.,[24] in their work propose the use of traffic information to switch *on* or *off* networks to save power. However, similar to the above, their approach requires an always-on control link to maintain connectivity.

Dannis et al.,[15] proposed energy proportional interconnects based on a similar method of reducing link power consumption. In their work on energy proportional interconnects, they evaluate the idea of reducing power consumption during periods of inactivity. However, their approach towards energy efficiency involves reducing the link rates of aggregated links. Aggregated links are typically networks built using multiple links of lower rate, aggregated to a single logical high bandwidth link. In their approach, during periods of inactivity, link rates are reduced to a lower link bandwidth to save power. This work is similar to Adaptive Link Rate proposals for EEE, mentioned in the background section. This paper does not analyze Adaptive Link Rate due to the scheme's above mentioned technology disadvantages.

In the work by Vassos et al.,[19], they provide a design space analysis for On/Off based links. They evaluate two different machines for the power saving capability of On/Off interconnects. For their first machine, they use a fast interconnect similar to an on-chip interconnect, using corresponding on-chip benchmarks. In their second machine, they evaluate a slow network, similar to cluster networks using synthetic benchmarks, with messages arrival following a Poisson process. Similar Ethernet evaluation reports [20, 23] also use synthetic benchmarks to evaluate on/off networks. In our approach, we evaluate Energy Efficient Ethernet using actual HPC traces collected from a production supercomputer.

J.A. Maestro et al., in their work [18] evaluate EEE for its potential over industrial Ethernet based systems using analytical models. Similarly, Sergio et al., in their work, analytically model and analyze the potential of EEE for energy savings[21]. They provide models for Ethernet standards with frame buffering. Their results suggest that frame buffering offers increased energy savings, however, with the cost of increased packet delays. In our work, our evaluation focuses on the tolerance of HPC applications to such packet delays required by EEE standard for larger energy savings.

Relevant work on Energy Efficient Ethernet [2, 16, 20, 23, 27–31] provide detailed evaluations on EEE for its potential for desktop and IT based systems; however, they do not give a sense of its performance over HPC. To the best of our knowledge, our work is the first at evaluating Energy Efficient Ethernet from the perspective of High Performance Computing. In our work, we compare the latency, power and performance requirements of HPC applications to the potential promise of Energy Efficient Ethernet.

IV. Evaluating Energy Efficient Ethernet for High Performance Computing

In this section, we evaluate the proposed IEEE standard on actual HPC workloads. The below methodology section describes our experimental setup, applications and the power model used. Following the methodology, we first evaluate our HPC workloads for their bandwidth and latency sensitivity and draw conclusions for EEE based on the same. Following which, we examine link activity for their potential for power savings with EEE and finally discuss performance and power evaluation of Energy Efficient Ethernet. To evaluate EEE, we simulate Energy Efficient Ethernet links on various test machines and obtain statistics for various workloads. Further, we compare EEE with our proposed *Power-Down Threshold* and draw power estimates based on the results obtained. Although our study focuses on Energy Efficient Ethernet, the evaluation and discussions can apply to design decisions of other On/Off networks for HPC.

TABLE II
Network and node parameters for EEE evaluation

Test Machine	Node Perf.	Interconnect
Machine-Low	62 GFlops	2x10Gbps links with 1μs latency
Machine-Mid	112 GFlops	40Gbps links with 1μs latency
Machine-High	224 GFlops	100Gbps links with 1μs latency
Machine-Acc	348 GFlops	100Gbps links with 1μs latency

A. Methodology

1) Simulation Infrastructure: For evaluation of Energy Efficient Ethernet, we use an extension of *Dimemas*[5, 6], an event-driven cluster simulator, whose network modules we have modified to support Low Power Idle and the proposed Power-Down Threshold. The simulation infrastructure supports the execution of *Paraver*[5] traces that we obtained from a production supercomputer - MareNostrum built with PowerPC970MC blades. The simulator reconstructs the behavior of the actual application from traces that contain CPU intervals and MPI/communication event information (message size, identifiers, type, source-destination etc) from the original execution. The simulator models simple node modules that contain CPUs, memory and on-board interconnect. Simulated CPUs based on their performance parameters operate relative to the actual application CPU(PowerPC970) intervals recorded in the traces. The cluster network is modeled as a point-to-point network with duplex links, to execute MPI events over a fully-connected mesh. The communication is based on a linear performance model, however non-linear effects such as network conflicts are also considered[6].

Low Power Idle (LPI) is implemented within the link module. Each duplex link contains its corresponding wake, sleep and Power-Down Threshold states. LPI is implemented such that send or receive both triggers a common LPI state within the link. On receiving a frame, both sending and receiving links are synchronously woken up or put into sleep state. Traces replayed in Dimemas contain inherent communication and execution dependencies that are used to maintain coherency and correctness in simulations. Dimemas has been verified for its accuracy over production supercomputers.

TABLE III
HPC WORKLOADS USED IN SIMULATIONS

Name (Nodes Executed)	Class	Description
CPMD (128)[32]	Chemistry	Molecular Dynamics
GADGET (128)[33]	Astro-Physics	Dark-matter simulation
GROMACS (128)[34]	Biology	Biomolecular dynamics
LINPACK (256) [35]	Benchmark	Linear algebra solver
MILC (128)[36]	Physics	Sub-Atomic Interactions
NAMD (64)[37]	Biology	Biomolecular simulations
PEPC (64)[38]	Mathematical	Parallel Coulomb Solver
QUANTUM (128)[39]	Chemistry	Nanomaterials modeling
WRF (128)[40]	Meteorology	Weather Forecasting Model

2) Experimental Setup: For our EEE evaluation, we model four different machines, each with varying network bandwidth and node performance, as shown in Table II. Our choice of node performance for *Machine-Low*, *Machine-Mid*, *Machine-High* and *Machine-Acc* comes from the top four machines of TOP500 (June'12) that use Ethernet. The second fastest Ethernet based machine, Amazon EC2 Cluster, contains Xeon 8-core processor. The sustained node performance of the above machine is estimated to be 112 GFlops/node with one socket/node and 225 GFlops/node with 2 sockets/node (Calculated based on system information provided at TOP500). The fastest Ethernet based system contains Xeon processors with NVIDIA 2090 GPUs; whose node performance we estimate based on a similar machine Tianhe-1A (TOP500 Rank 4) at 349 GFlops/node, for our accelerator based machine. Similarly, we calculate our *Machine-Low* based on a machine ranking 4th among the fastest Ethernet based TOP500 machines at 62GFlops. Note that our chosen interconnect bandwidths for corresponding systems are rather conservative in comparison to high-end TOP500 machines of similar node performance. For example, the Tofu[25] and Arch[26] custom interconnects of the K-supercomputer (128 GFlops/node) and Tianhe-1 (349 GFlops/node) supercomputer have node-to-node bandwidths of 100Gbps and 160Gbps respectively. We assume a node-to-node latency of 1μs across test machines based on relevant literature and specification sheets of products.

Table III provides a description of the HPC workloads used in this study. In general, HPC applications consist of initialization and finalization phases with a large number of iterative execution phases between them. Due to the large sizes of the actual application traces (hundreds of gigabytes), for our analysis we cut out a few of these repetitive execution phases. The applications presented in this study have diverse network requirements; we analyze network latency and bandwidth sensitivity of these applications in the next section.

For our analysis, Low Power Idle mode of Energy Efficient Ethernet is setup with timing information provided by the IEEE 802.3az EEE[1] draft (Table I). EEE standard does not provide timing information on the wake and sleep modes for Ethernet links with bandwidth upwards of 10Gbps. Since the network links we consider for our study are 10Gbps and upwards, we conservatively assume the timing information of 10Gbps links, for 40 and 100 Gbps links as well. However, as shown in Table I, the trend shows decreasing *wake-up* and *sleep* timings with higher bandwidths. When EEE standard proposes timing information for 40Gbps and 100Gbps, we expect the *wake-up* and *sleep* modes to require lesser time implying that EEE would perform proportionally better to the results presented in this paper with the same *wake-up* and *sleep* timings of 10Gbps.

To model power consumption, we use data available in the industry and relevant literature [7, 8, 11, 14, 17]. We model the power consumption of links to consume 10% of the full link power in idle mode based on the IEEE 802.3az EEE [1] draft. To calculate system power consumption, we assume that links consume 65% of the network power and consider total network power to be about 20% of the system power[14, 17]. Consequently we calculate the power consumption of each of the machines (Low, Med, High and Acc) based on the above model and network utilization. It is to be noted that, since we use the same wake-up timing information of 10Gbps in the case of 40 and 100Gbps, our power estimates are an upper bound to actual power consumption of machines using 40 and 100Gbps. This is because, as per trend, when EEE for 40 and 100 Gbps are proposed (by IEEE community) with faster wake-up latencies (than 10Gbps), lesser time is spent on switching *on* the link, resulting in better power savings. Based on previous literature [11, 15] we assume other network elements to consume power proportional to utilization.

B. Evaluation Results

1) Interconnect sensitivity analysis: As shown in Figure 5, the requirements of our workloads vary widely in terms of their sensitivity to network latency and bandwidth. In Figure 5 we show normalized execution time, as a function of latency (Figure 5(a,b)) and bandwidth (Figure 5(c,d)). Figure 5(a,c) shows the execution time for each application, on a fixed machine, *Machine-Low* and Figure 5(b,d) shows the execution time for each machine (Low, Med, High and Acc), with *Max, Min* and *Avg* corresponding to the most sensitive and least sensitive application and the average across applications. Our graphs point out the diversity of our HPC applications with their bandwidth and latency requirements. Furthermore, we show that our choice of interconnects for the corresponding systems fall within a 10% drop in performance in comparison to infinite bandwidth and zero latency, to which the graphs are normalized. A 10% drop in application performance is generally not ideal for state-of-the-art HPC designers. However, the diversity of applications and large interconnect capital costs play a factor, often making it difficult to build the perfect interconnection network for all applications. However, when faster interconnects are used, since EEE is intended to be energy proportional, higher performance will translate to lower average power consumption (we discuss more on this in the next section).

2) Idle link event time analysis: Energy Efficient Ethernet, as we mentioned in previous sections, uses Low Power Idle (LPI) as a proposed mechanism for bringing about energy proportionality. LPI works on the basis that links are turned off during periods of inactivity. While this proposal seems inherently energy efficient, its power saving potential is heavily dependent on the frequency at which the link becomes inactive. Due to a non-negligible increase in latency imposed by LPI every time a link needs to *wake-up* from low power mode, the frequency at which these links are turned *on* and *off* is crucial to power savings. In this regard, the time required for powering the link back to operational mode determines power savings as well as performance overheads. Conventional wisdom suggests that deeper power saving modes require correspondingly large *wake* times. Note that a large *wake* time introduces large latencies, which in turn results in large performance overheads.

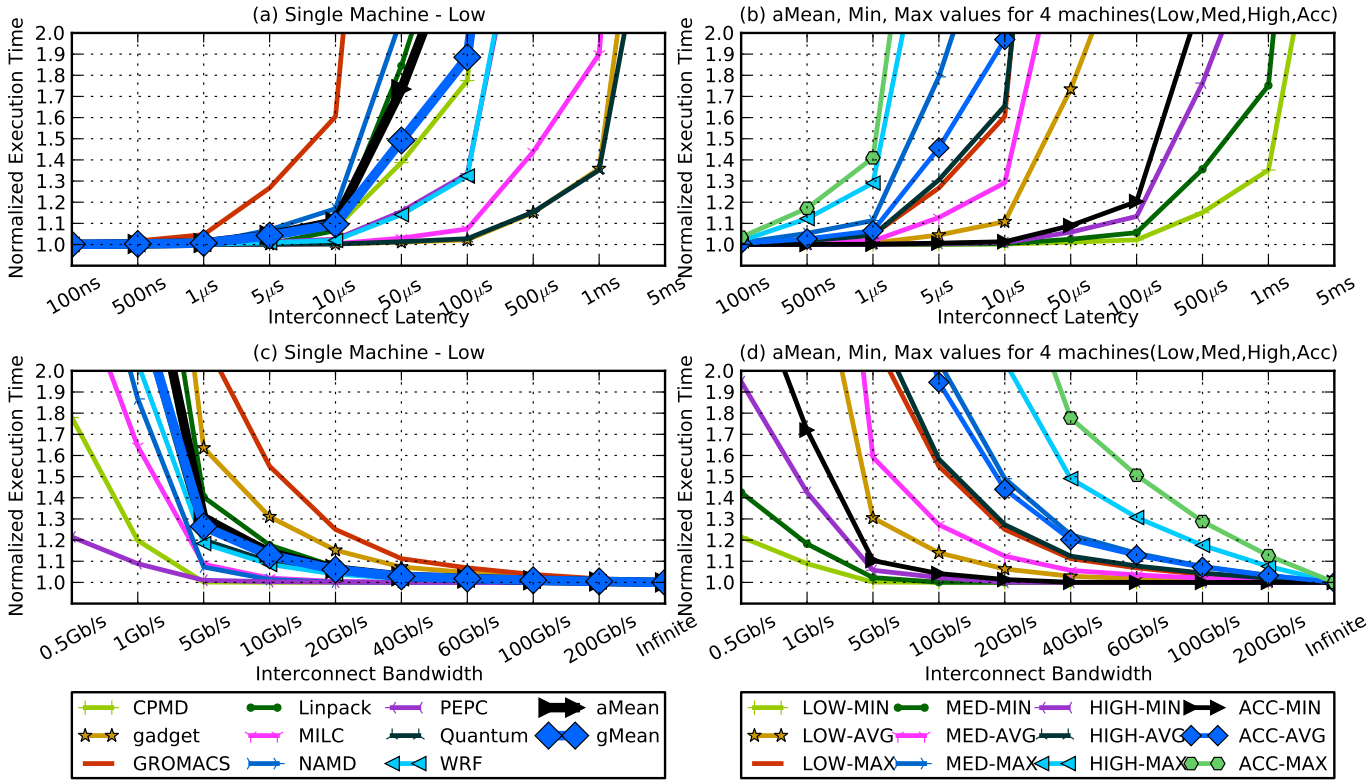

Fig. 5: Bandwidth and Latency sensitivity of the applications executed on the four test machines (Low, Med, High and Acc)

To keep performance overheads at a minimum, networks that require large *wake-up* times would rarely be put into low power modes. To understand and evaluate power saving benefits offered by On/Off based interconnects with specified *wake* times, we analyze for our applications the link idle time distribution.

In Figure 6 we show histograms to illustrate and describe intervals during which links remained idle, running application *GADGET* over *Machine-Mid*. In Histogram 1, we show *idle link events*[1] of various time intervals. In the x-axis of Histogram 1 we show the time distribution and correspondingly for each point in the x-axis the number of events along the y-axis. Our time distribution corresponds to the time interval during which the link remained idle. To illustrate, we see a peak at 100μs with approximately 3,750 events. The 3,750 events in this peak correspond to 3,750 separate intervals during which the link was idle, each of whose duration is close to 100μs. The peaks that we see between 1μs to 4μs and 100μs to 400μs in Histogram 1 of Figure 6 are a result of self-similarity or repetitive patterns with these applications. This essentially shows that the distribution that holds true for one or more iterations of our cut of the *GADGET* application, will remain similar throughout all iterations of the application execution.

In Histogram 2 of Figure 6, we show the above Link Idle Event distribution in actual time periods. Here the bars of the histogram represent the total idle time throughout the application's execution, from all idle intervals of length close to the value on the x-axis. To illustrate, consider the bar at 100 ns which contains an y-axis equivalent of approximately 1μs. This shows that the total time in all idle link events that lasted close to 100ns, accumulated over the whole trace execution is 1μs. Although a majority of the idle link events come from the range of 1μs to 4μs, as shown in histogram 2, their accumulated link idle time does not exceed a few milliseconds. In Histogram 3, we convert Histogram 2 into its cumulative distribution graph to determine the cumulative time in all idle periods of length less than the value on the x-axis. For example, the total time in all intervals of length 100μs or less is above 1 second. As shown in Histogram 3, we calculate the interval length for which 90%, 99% or 99.9% of the total idle link time is in idle intervals that are longer than that value. In this case, for application *GADGET* on *Machine-Med*, 99.9% of the total application execution time that links remained idle, comes from events when a link remained idle for 52μs or longer. The corresponding values for the other applications and machines are listed in table IV. With the exception of GROMACS, all applications on Machine-Mid have 90% of their idle link times in the ranges of milliseconds or above.

From the design perspective of power savings modes for On/Off based interconnects such as Energy Efficient Ethernet, these numbers are relevant in designing *wake* timings and to gauge potential power savings using the above. For 10Gb Ethernet, the link *wake-up* and *sleep* times proposed by the IEEE 802.3az EEE [1] draft are 4.16μs and 2.88μs respectively. This is to show that an EEE link requires approximately 7μs in total to switch off and back on. For applications such as *GADGET*, which contains 99% of its link's total idle time coming from events where the link remains idle for beyond 100μs, 7μs potentially offers enough time to go into and back from low power mode without significant loss in performance.

[1]We assume an *idle link event* to be any interval during which no data is being transmitted on the link

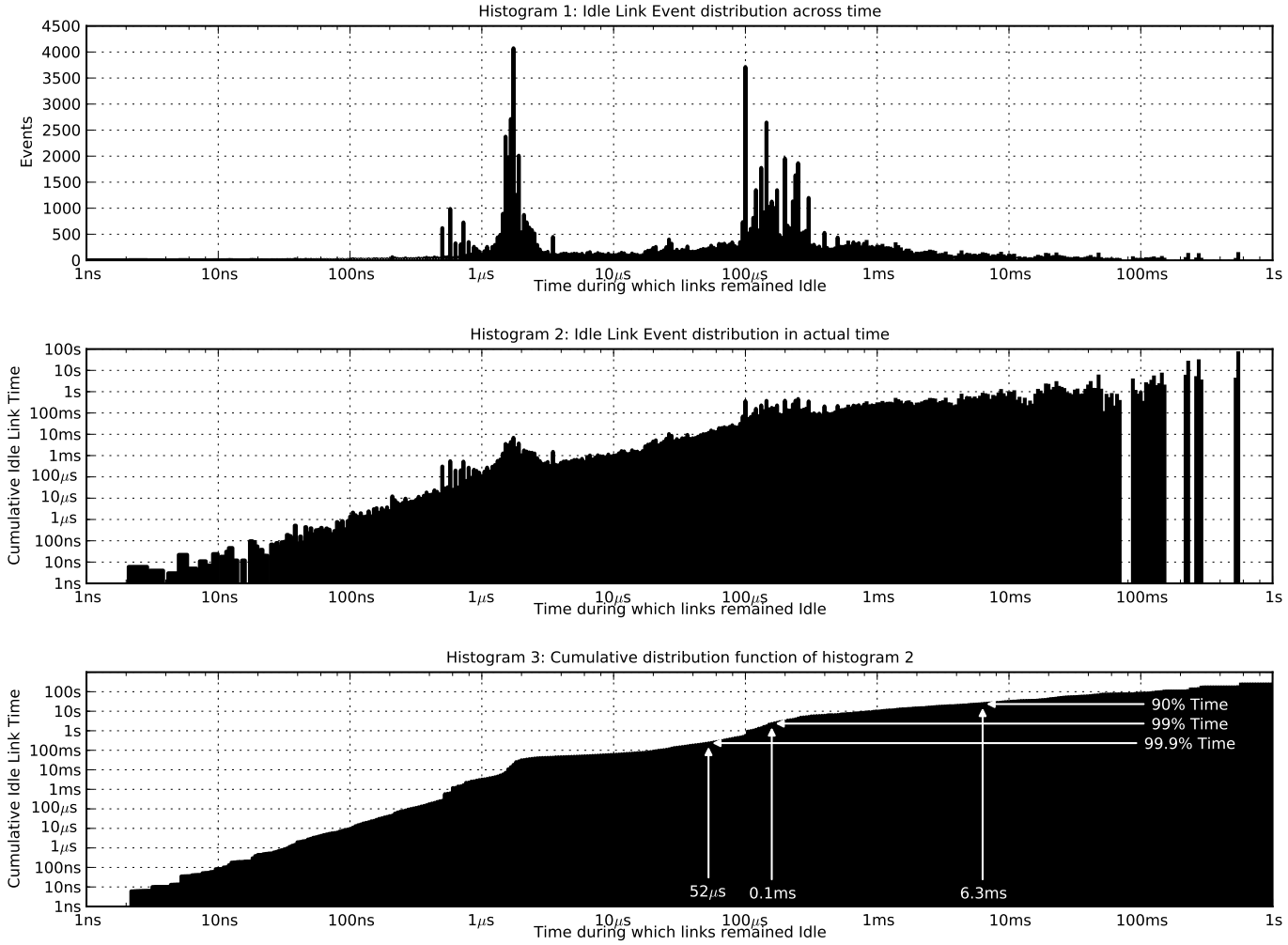

Fig. 6: Histograms show Idle Link Event distribution for application *GADGET* running on Machine-Mid. Histogram 1 (top) shows the number of idle link events as a function of time. Histogram 2 shows a product of total idle link events by idle link time for the corresponding events. Histogram 3 shows the cumulative distribution graph of Histogram 2

Large gaps in the ranges of 70ms to 800ms shown Histogram 2 of Figure 6 represent lengths of idle intervals that never occur. Essentially, if we consider a gap near 100ms (assume the gap is between 70-80ms), this is to show that no link that has gone idle, has had an idle time that has lasted between 70 and 80ms. Note that these gaps occur in the range of 70ms to 800ms, which contribute to a large portion of the total link idle time (82% in the case of the histogram 2 in Figure 6). The reason behind these gaps may be that the majority of the execution time contains distinctively large computation periods unhindered by communication events (during which all links remain idle). Our analysis on experiments over various machines suggests the existence of these patterns in all applications except GROMACS, NAMD and Quantum Espresso. In the case of the above Histogram 2 of Figure 6, the gap between 70 to 80ms suggests that, when a link has been idle beyond 70ms, we can be certain that it would remain idle for another 10ms (until 80ms) in time. These results suggest that even a simple history based prediction algorithms, that dynamically records this information to switch On or Off links would greatly benefit from deeper sleep states of the network, however, this work is beyond the scope of this paper.

3) Performance and power analysis of Energy Efficient Ethernet: In this section, we evaluate Energy Efficient Ethernet, its power and performance with and without the proposed *Power-Down Threshold*. As mentioned in the methodology section, we use *wake-up* latency of 4.16μs and a *sleep* latency of 2.88μs for our simulations. This latency is the standard proposed wake-up and sleep latency of 10Gbps which we use for all test machines (although, as explained earlier, we expect lower wake-up and sleep latencies for faster interconnects).

Power-Down Threshold: From our previous analysis of HPC workloads, we get a clear sense of how latency intensive HPC workloads can be. The previous proposals suggest the use of Frame Buffering, where frames are held back up to 100μs before transmission to improve energy savings. However, with regard to HPC, as shown in Figure 5, a 10μs added latency (due to frame buffering) reduces performance on average by about 7% to 40%. This reduction in performance would essentially negate any power benefits obtained by the use of Energy Efficient Ethernet (we discuss this in more detail

TABLE IV

DISTRIBUTION OF LINK IDLE TIMES: THE NUMBERS INDICATE THAT 90%, 99% OR 99.9% OF THE TOTAL TIME A LINK REMAINS IDLE DURING THE ENTIRE APPLICATION EXECUTION, COMES FROM EVENTS WHERE THE LINK REMAINS IDLE FOR A PERIOD ABOVE THE TABLE SPECIFIED TIME FOR THEIR RESPECTIVE MACHINES

Applications	Machine-Low			Machine-Mid			Machine-High			Machine-Acc		
Idle Link Time	90%	99%	99.9%	90%	99%	99.9%	90%	99%	99.9%	90%	99%	99.9%
CPMD	10s	0.4s	50ms	5.7s	0.22s	22ms	2.8s	0.11s	0.2ms	1.8s	72ms	10μs
GADGET	10ms	0.2ms	91μs	6.3ms	0.1ms	52μs	3.3ms	72μs	19μs	1.9ms	52μs	13μs
GROMACS	3.9μs	1μs	0.3μs	2μs	0.7μs	0.2μs	1.6μs	0.5μs	0.18μs	1.3μs	0.4μs	0.1μs
LINPACK	22ms	0.3ms	2.75μs	11ms	0.1ms	1.8μs	6.6ms	31μs	1μs	9.1ms	3.6μs	72μs
MILC	1.3ms	0.1ms	25μs	0.7ms	79μs	13μs	0.3ms	34μs	6.3μs	0.2ms	30μs	4.7μs
NAMD	0.1ms	26μs	1.4μs	0.1ms	12μs	91μs	45μs	4.1μs	1μs	28μs	1.9μs	75μs
PEPC	0.6ms	45μs	4.1μs	0.3ms	21μs	2.0μs	0.1ms	9.1μs	1.1μs	0.1ms	5.2μs	1.1μs
QUANT.ESP	1.8ms	0.9ms	0.1ms	1ms	0.5ms	87μs	0.4ms	0.3ms	41μs	0.3ms	0.1ms	28μs
WRF	0.2ms	27μs	7.2μs	0.1ms	15μs	3.9μs	79μs	7.5μs	1.3μs	52μs	5.2μs	95μs

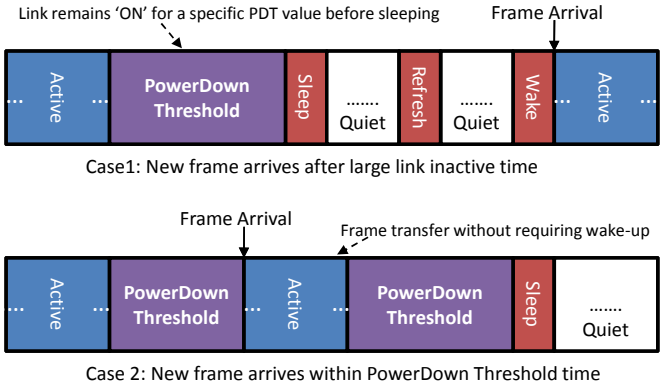

Fig. 7: State transactions of Power-Down Threshold (PDT) for Low Power Idle in Energy Efficient Ethernet - Link remains in 'on' state after frame transmission for specific PDT time. Frame arrival within PDT time does not incur additional latencies for the turning *off* and *on* the link.

below). Hence, as a solution to maintaining performance, as well as saving power, we propose *Power-Down Threshold* as a scheme for EEE for HPC. While in the case of Frame Buffering, frames are buffered and the link does not switch *on* for transmission until time-out or a specific number of frames are obtained. In the proposed *Power-Down Threshold*, as shown in figure 7, after the transmission of a frame, EEE remains in *on* state until a threshold time before shifting back to low power mode. During this duration, frames arriving at the link is transmitted without the need for powering the link back up. This added threshold time would result in increased power consumption, since the link remains *on*, longer than required in the case when no messages arrive before the threshold. However, we argue that this increased power consumption (in an already energy proportional interconnect) is rather small in comparison to power savings achieved by negating any performance overheads due to the use of EEE.

Experimental Results: Figure 8 shows our experimental results on the evaluation of EEE and *Power-Down Threshold*. In Figure 8, sub-figures (a) and (b) corresponds to performance graphs and sub-figures (c) and (d) correspond to the average energy consumption of all links. Further, sub-figures (a) and (c) refer to results for all applications over a single machine and sub-figure (b) and (d) shows results for all 4 machines, with the averages, minimum and maximum values of the applications. In Figure 8(a,b), at *Power-Down Threshold* value zero corresponds to Energy Efficient Ethernet where for every message, links are required to be powered *on* and *off* introducing wake-up and sleep transition delays. As we can see, from the figure, EEE introduces an average of approximately 25% reduction in performance for Machine-High, 35% for Machine-Acc and 10% and lower for Machine-Mid and Machine-Low. Correspondingly, on an average, there is 60% reduction in the link energy consumption. However, as we increase the *Power-Down Threshold* values, we see that at about 50μs the performance overhead of EEE drops down to less than 2% with savings in link energy consumption at 70%.

Total System Power Estimates: The above savings in power correspond to link power; here we calculate the overall system power consumption based on the methodology specified in section 4.1. On considering two cases, where *Power-Down Threshold* is zero (Energy Efficient Ethernet) and 50μs threshold values (Energy Efficient Ethernet + Power-Down Threshold) we calculate power consumption across the overall system. As mentioned above, we project power estimates assuming that node and network elements operate at full power during the entire application execution. However, during real execution this is not necessarily the case, many components including CPUs can enter sleep mode, so the increase in energy may be lower than our estimates. For Machine-High, shown in Figure 8 (a), we see that there is a 25% reduction in performance, with power savings in the interconnect of about 60%. This translates to an overall system power consumption at 115%, suggesting that deploying Energy Efficient Ethernet on Machine-High would lead to a 15% increase in power and 25% decrease in performance. However, with a *Power-Down Threshold* of about 50μs, we see that the performance overhead drops down to less than 2% with interconnect power savings of 70%, translating to overall system power savings of 7.3%.

V. DISCUSSION AND RECOMMENDATIONS

In this section we discuss our recommendations for vendors deploying Energy Efficient Ethernet in HPC.

A. Frame Buffering/Packet Coalescing

Frame buffering, as we mentioned in previous sections, buffers frames or waits until a time-out before frames are issued through the link. Frame buffering time-out of 10-120 microseconds is used to hold frames in previous proposals. Holding frames ensures that link states are not frequently

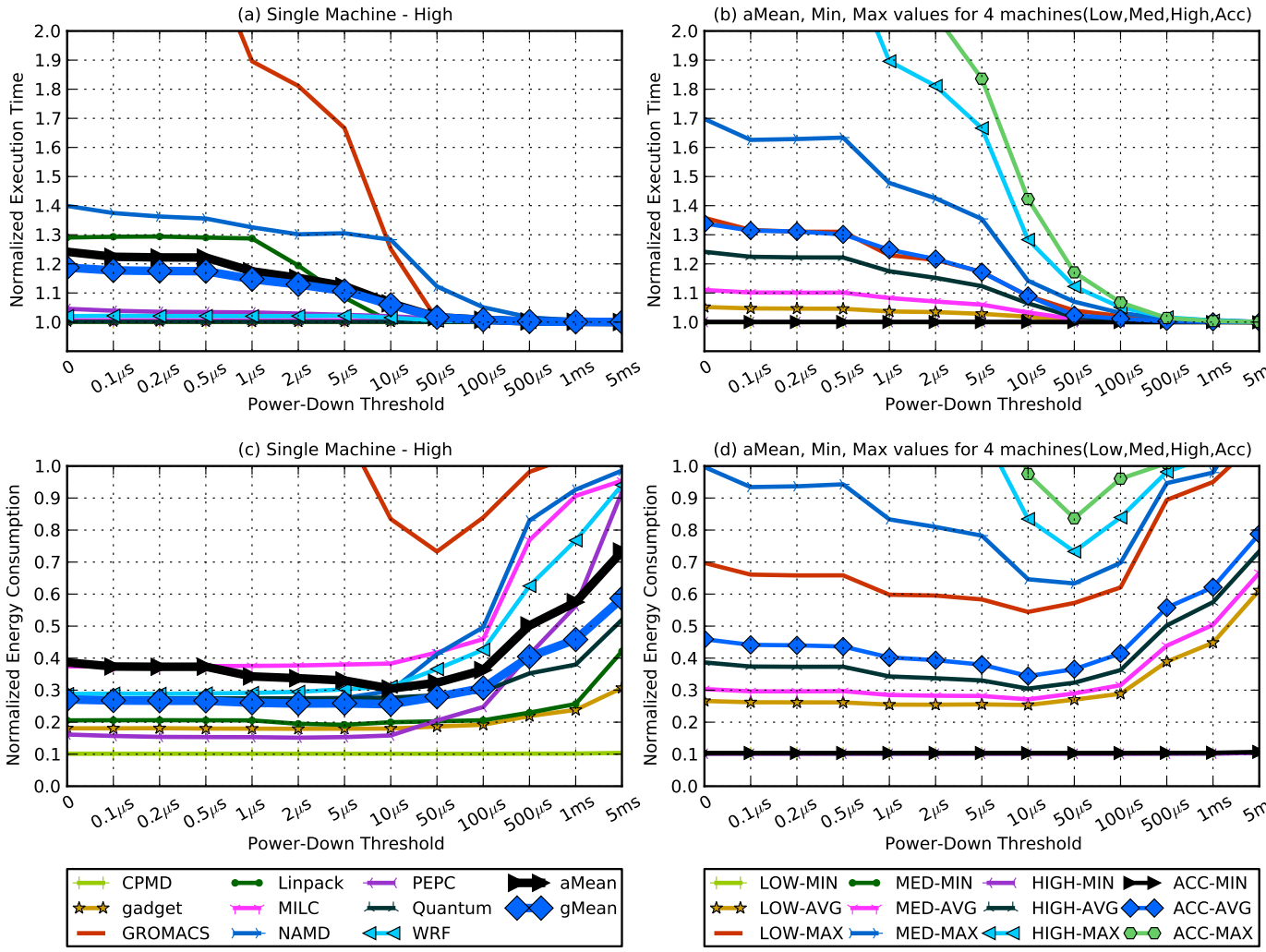

Fig. 8: Performance and power graphs of Energy Efficient Ethernet over proposed *Power-Down Threshold*

toggled due to small frames that arrive at frequent intervals. Two observations from our presented results suggest the frame buffering is not required for HPC. Firstly, as we see in previous results, the majority of the HPC applications are heavily latency sensitive. The latency requirements of our applications, as we find in Figure 5, become more challenging with faster machines. Although HPC interconnect usage contributes to a small part of the application's execution time, they require very high bandwidth and low latency. Hence, frame buffering as a proposal to hold back frames until time-out would degrade performance to negate any power benefits obtained. Secondly, HPC applications due to their complex intercommunication dependencies, do not transmit subsequent messages before acknowledges are received for the first ones sent. This makes holding frames destructive to the performance of HPC workloads.

B. *Discussing On/Off Links for power savings*

While discussing the related work, we point to other similar On/Off based interconnect proposals. These proposals typically switch off links that are aggregated to a larger bandwidth. Hence, when the traffic in these interconnects is low, correspondingly links of the network are switched off to reduce bandwidth and consequently save power. The difference in the case of EEE is that it runs at maximum speed, and the transfer is completed in the shortest possible time. Although the technology behind the methods are different, if we assume that their power consumption and On/Off latencies are the same, both methods would consume the same power. However, trends in EEE suggest that, with an increase in interconnect bandwidth, power increases *sub-linearly*. This is to say that the power consumption of a 10Gb link is less than ten 1Gb links aggregated to form a 10Gb link. Since they transfer data at the same rate, the 10Gb link would consume less power than its aggregated counterpart.

VI. Conclusions and Future Work

In this paper, we presented the first analysis of Energy Efficient Ethernet for HPC workloads. We proposed the use of *Power-Down Threshold* to further increase power savings of HPC systems by having the link in *on* mode until a threshold is reached. Based on our analysis of HPC workloads on EEE, we proposed various recommendations to further increase power savings. Based on results obtained, we conclude that out-of-the-box Energy Efficient Ethernet for HPC does not

provide sufficient power savings to justify its use. However, with further EEE enhancements such our proposed *Power-Down Threshold* makes using Energy Efficient Ethernet for HPC a very promising solution for energy proportionality. We suggest that existing complimentary proposals such as frame buffering that increase the latency of messages are harmful to performance, negating any power benefits obtained by such schemes. As a part of future work, we plan to extend our study of EEE to include Dynamic Power-Down Threshold, effect of prediction methods to overlap link wake-up delay and move into deeper sleep modes with the use of application idle link time information.

ACKNOWLEDGMENT

We thank the anonymous reviewers for their constructive, insightful and detailed reviews. This research was supported by the Ministry of Economy and Competitiveness of Spain under the contract TIN-2007-60625, European HiPEAC-3 Network of Excellence (ICT-287759), European Union's Seventh Framework Programme [FP7/2007-2013] under project Mont-Blanc (grant agreement number: 288777) and finally the Generalitat de Catalunya (FI-AGAUR program 2012 FI B 00644).

REFERENCES

[1] IEEE Draft P802. 3az/D2. 3, Energy Efficient Ethernet, 2010.
[2] Active/Idle Toggling with Low Power Idle, IEEE 802.3az Task Force.
[3] Top500 list, June 2012, url: *http://www.top500.org/list/2012/06/100*
[4] Green500: List of most energy efficient Supercomputers. url: *http://www.green500.org/*
[5] Jesus Labarta *et al.*, "DiP: A Parallel Program Development Environment", In proceedings of the Second International Euro-Par Conference on Parallel Processing, 1996
[6] Rosa M. Badia *et al.*, "Dimemas: Predicting MPI applications behaviour in Grid environments", Workshop on Grid Applications and Programming Tools (GGF8), 2003.
[7] L. Shang, L.-S. Peh, and N. Jha. Dynamic voltage scaling with links for power optimization of interconnection networks. In proceedings of the 9th International Symposium on High-Performance Computer Architecture, 2003
[8] IBM InfiniBand 8-port 12x Switch, url: *http://www-3.ibm.com/chips/ products/infiniband*
[9] A. Sodani. Race to exascale: Opportunities and challenges. Keynote Speech, MICRO 44, Dec, 2011.
[10] D. Turek, Challenges on the road to exascale computing (invited talk), Simulating the Future Workshop, 2008.
[11] H.-S. Wang *et al.*, A power model for routers: modeling alpha 21364 and infiniband routers. In proceedings of 10th Symposium on High Performance Interconnects, 2002.
[12] Peter Kogge et al. ExaScale Computing Study: Technology Challenges in achieving Exascale Systems, June'08.
[13] Scientific grand challenges: Crosscutting technologies for computing at the exascale. Report from the Workshop Held Feb'10, by U.S.DOE.
[14] P. Kogge. Architectural challenges at the exascale frontier (invited talk), Simulating the Future Workshop.2008.
[15] Abts, Dennis and Marty, Michael R. and Wells, Philip M. and Klausler, Peter and Liu, Hong, Energy proportional datacenter networks. In proceedings of the 37th annual international symposium on Computer architecture, 2010
[16] M. Koibuchi, T. Otsuka, H Matsutani and Amano, H, An on/off link activation method for low-power ethernet in pc clusters. In proceedings of 23rd IEEE International Parallel & Distributed Processing Symposium,2009.
[17] Jian Li *et al.*,Power shifting in thrifty interconnection network, In proceedings of the 17th IEEE International Symposium on High-Performance Computer Architecture, HPCA'2011
[18] J. Maestro *et al.*, Energy efficiency in industrial ethernet: The case of powerlink. IEEE Transactions on Industrial Electronics, Aug. 2010.
[19] V. Soteriou and L.-S. Peh. Design-space exploration of power-aware on/off interconnection networks. In proceedings of the IEEE International Conference on Computer Design, 2004.
[20] K. Christensen *et al.*, IEEE 802.3az: the road to energy efficient ethernet. In Communications Magazine, IEEE, Nov 2010.
[21] S. Herreria et al.,How efficient is energy efficient ethernet, in 3rd International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT), 2011
[22] M. Mostowfi and K. Christensen. Saving energy in lan switches: New methods of packet coalescing for energy efficient ethernet, in proceedings of the International Green Computing Conference and Workshops, 2011
[23] P. Reviriego *et al.*, Performance evaluation of energy efficient ethernet. IEEE Communications Letters, Sept 09.
[24] M.Alonso *et al.*, Dynamic power saving in fat-tree interconnection networks using on/off links. In proceedings of 20th IEEE International Parallel & Distributed Processing Symposium, 2006
[25] Ajima *et al.*, "Tofu: A 6D Mesh/Torus Interconnect for Exascale Computers," in journal of Computer, 2009.
[26] Liao XK *et al.*, The TianHe-1A Supercomputer: Its Hardware and Software, in Journal of Computer Science and Technology.
[27] K. J. C.Chamara Gunaratne. Ethernet adaptive link rate: System design and performance evaluation, in proceedings of 31st IEEE Conference on Local Computer Networks, 2006.
[28] H.Anand *et al.*, Ethernet adaptive link rate (ALR): Analysis of a mac handshake protocol, in proceedings of 31st IEEE Conference on Local Computer Networks, 2006.
[29] Zhang, Real-time performance analysis of adaptive link rate, in proceedings of 33rd IEEE Conference on Local Computer Networks, 2008
[30] C. Gunaratne *et al.*, Reducing the energy consumption of ethernet with adaptive link rate (ALR). IEEE Trans. Comput. Apr08.
[31] F. Blanquicet and K. Christensen. An initial performance evaluation of rapid phy selection (rps) for energy efficient ethernet, in proceedings of 32nd IEEE Conference on Local Computer Networks, 2007
[32] D. Marx and J. Hutter. Ab-initio Molecular Dynamics: Theory and Implementation, NIC. Forschungszentrum Jlich, 2000.
[33] V. Springel. The cosmological simulation code GADGET-2. Royal Astronomical Society 2005.
[34] B. Hess and E. A. Kutzner. Gromacs 4: Algorithms for highly efficient, load-balanced, and scalable molecular simulation. Journal of Chemical Theory and Computation, 2008.
[35] T. E. A.Davies. High performance linpack benchmark: a fault tolerant implementation without checkpointing. In ICS'11.
[36] MIMD Lattice Computation (MILC) Collaboration.
[37] J. C. Phillips and R. B. et al. Scalable molecular dynamics with NAMD. Journal of Computational Chemistry, 2005.
[38] P. Gibbon. PEPC: Pretty Efficient Parallel Coulomb-solver. Interner Bericht Zentralinstitut für Angewandte Mathematik, Forschungszentrum Jülich. 2003.
[39] P. Giannozzi *et al.*, Quantum espresso: a modular and open-source software project for quantum simulations of materials. Journal of Physics: Condensed Matter, 2009.
[40] Michalakes, J. *et al.*, "The Weather Reseach and Forecast Model: Software Architecture and Performance", in proceedings of the 11th ECMWF Workshop on the Use of HPC In Meteorology, Oct'04.