Dynamic Voltage Scaling with Links for Power Optimization of Interconnection Networks

Li Shang   Li-Shiuan Peh   Niraj K. Jha

Department of Electrical Engineering
Princeton University, Princeton, NJ 08544

Abstract

Originally developed to connect processors and memories in multicomputers, prior research and design of interconnection networks have focused largely on performance. As these networks get deployed in a wide range of new applications, where power is becoming a key design constraint, we need to seriously consider power efficiency in designing interconnection networks. As the demand for network bandwidth increases, communication links, already a significant consumer of power now, will take up an ever larger portion of total system power budget. In this paper, we motivate the use of dynamic voltage scaling (DVS) for links, where the frequency and voltage of links are dynamically adjusted to minimize power consumption. We propose a history-based DVS policy that judiciously adjusts link frequencies and voltages based on past utilization. Our approach realizes up to 6.3X power savings (4.6X on average). This is accompanied by a moderate impact on performance (15.2% increase in average latency before network saturation and 2.5% reduction in throughput.) To the best of our knowledge, this is the first study that targets dynamic power optimization of interconnection networks.

1 Introduction

In recent years, interconnection network fabrics, historically used in high-end multiprocessor systems [17], have proliferated to a wide range of communication systems – clusters [1], terabit Internet routers [5], server blades [9, 16], and on-chip networks [6, 24]. These myriad applications place new demands on the network fabric. In the past, a low-latency and high-throughput fabric was the ultimate goal. This is no longer sufficient. Applications of interconnection networks are becoming heavily power-constrained.

While there have been extensive architectural research into power-aware processors and memories, research into network power efficiency has been lagging. Some prior studies on network power efficiency focused on power modeling and simulation [18, 28]. While circuit-level techniques have been studied, such as multi-gigabit low-power links [8, 14], architectural mechanisms targeting network power efficiency have yet to be explored. This needs to be urgently addressed, as interconnection networks consume a substantial fraction of total system power. In a Mellanox server blade, the router and links are estimated to dissipate 15W out of the total budget of 40W (37.5%), with the processor allocated the same power budget of 15W [16]. The integrated router and links of the Alpha 21364 processor [17] consume 23W, where 58% of the power is consumed in the link circuitry[1]. Designers of the IBM InfiniBand 8-port 12X switch estimate it consumes 31W, with links taking up 65% (20W). In a terabit Internet router, the power consumed by the interconnection network circuitry is about a third of that dissipated by the entire line card. This will only worsen as the demand for network bandwidth increases. Off-chip memory bandwidth has been increasing fast in multiprocessor systems – while Alpha 21164 had just 1.2GB/s bus bandwidth to off-chip memory, the newest Alpha 21364 router provides a whopping 22.4GB/s raw network bandwidth. In Internet routers, a similarly increasing trend of network bandwidth is observed, with port bandwidths scaling from OC48 (2.5Gb/s) to OC768 (40Gb/s).

Dynamic voltage scaling (DVS) has been proposed and widely deployed for microprocessors [2, 10, 25], uncovering significant power savings [11]. DVS in microprocessors exploit the variance in processor utilization, lowering the frequency (and voltage) when the processor is lightly loaded, and running at maximum frequency (and voltage) when the processor is heavily executing. Like microprocessors, there is a wide variance in link utilization in interconnection networks, depending on applications' communication patterns. This can lead to huge power savings if network bandwidth can be tuned accurately to track usage. Recently, variable-frequency links were proposed [12, 29]. These links track and adjust their voltage levels to the minimum supply voltage as link frequency is changed, providing a mechanism that can potentially lower

[1]Alpha 21364 designers estimate that the integrated router and links dissipate 23W out of the total chip power of 125W (20%), with the router core consuming 7.6W, link circuitry (drivers, pad logic) consuming 13.3W, clocks taking up 2W and miscellaneous circuitry the remaining.

link power dissipation by up to 10X. Variable-frequency links demonstrate the feasibility of applying DVS to links, opening up exciting avenues for architectural policies that control DVS links to trade off overall network power and performance.

In this paper, we propose a history-based DVS policy that uses past network utilization to predict future traffic and tune link frequency (and voltage) dynamically to minimize network power consumption while maintaining high performance. We first discuss the components of a DVS link, and DVS link characteristics that are critical to architectural DVS policies, in Section 2. Next, we explain in detail our history-based DVS policy, analyzing its hardware implementation cost and power consumption overhead in Section 3. In Section 4, we present the experimental setup and workload model used to evaluate the performance of history-based DVS. Our experiments show history-based DVS reducing power significantly by up to 6.3X. This is accompanied with 15.2% degradation in average network latency before network saturation and 2.5% reduction in network throughput. These results are realized despite making very conservative assumptions about DVS link characteristics. We believe this first study demonstrates the potential of using DVS links in high-performance power-efficient interconnection networks.

2 DVS Link Architecture and Modeling

Wei, Kim and Horowitz investigated voltage scaling for chip-to-chip parallel [29] and serial links [12], where the link automatically adjusts itself to the minimum possible supply voltage at different link frequencies. Their variable-frequency link was fabricated in 0.25 μm CMOS technology, dissipating power varying from 21mW at 1Gb/s to 197mW at 3.5Gb/s, providing a potential 10X improvement in power. While variable-frequency links were originally designed for offline frequency setting, and not for dynamic frequency and voltage scaling, the link architecture can be readily extended for DVS.

A variable-frequency link consists of the components of a typical high-speed link: a transmitter to convert digital binary signals into electrical signals; a signaling channel usually modeled as a transmission line; a receiver to convert electrical signals back to digital data; and a clock recovery block to compensate for delay through the signaling channel. In addition, it needs an adaptive power-supply regulator that tracks link frequency, regulates voltage to the minimum level required, and feeds regulated supply voltage to multiple links (say, the links of a network channel), amortizing its area and power costs. To extend variable-frequency links to DVS links, an additional frequency synthesizer that supplies the user-controlled frequency to the power-supply regulator is needed, as shown in Figure 1.

The important characteristics of a DVS link are (i) transition time - how long it takes a link to change from voltage level V_1 to V_2, (ii) transition energy - the overhead energy consumed

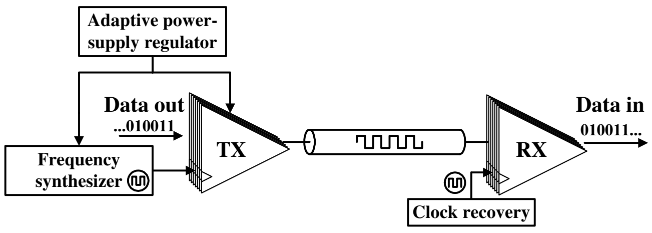


Figure 1. Components of a DVS link.

for a transition from V_1 to V_2, (iii) transition status - whether the link functions during a transition, and (iv) transition step - whether the link supports a continuous range of voltages, or if it only supports a fixed number of voltage levels.

We construct a multi-level DVS link model, as shown in Figure 2, which supports ten discrete frequency levels and corresponding voltage levels. At each frequency level, the link circuitry can function within a range of voltages. The voltage and frequency transitions between adjacent levels occur separately. When increasing link speed, the voltage is increased first, then the frequency. Conversely, when decreasing link speed, the frequency decreases first, followed by the voltage. The link functions during voltage transition but not during frequency transition. Based on the current circuit technology [2, 3, 7, 13, 22], the latency of voltage (frequency) transition between adjacent levels is set to 10 μs (100 link clock cycles).

Transition energy is derived based on Stratakos's analysis [23], where energy overhead when voltage transitions from V_1 to V_2 is calculated with the following first-order estimation equations:

$$Energy_{overhead} = (1 - \eta) \cdot C \cdot |V_2^2 - V_1^2| \qquad (1)$$

where C is the filter capacitance of the power-supply regulator and η is the power efficiency. In our experiments in Section 4, we assume 5μF capacitance and 90% power efficiency, based on the variable-frequency link presented in [12].

Noise is another issue in the design of link circuitry. Different noise sources, power supply noise, crosstalk, clock jitter, deviation of parameters, device mismatches etc., could result in voltage and timing uncertainty. For DVS links, supply voltage reduction magnifies the noise sensitivity of link circuitry. Since a lower link frequency decreases the ratio of timing uncertainty to bit time, frequency reduction improves communication reliability. Bit error rate (BER) is a measure of perfor-

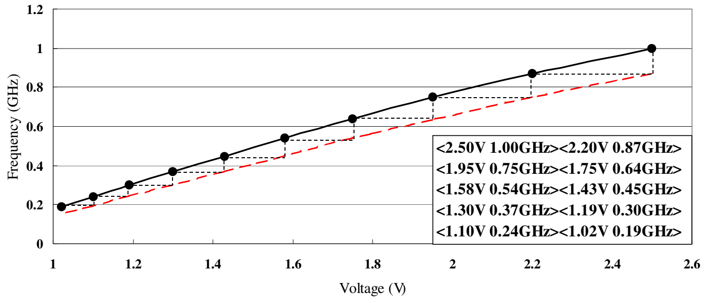


Figure 2. Multi-level DVS link model.

mance in link circuitry design. Current link designs [12] can achieve 10^{-15} BER over a wide range of voltages (0.9V-2.5V) and frequencies (200MHz-700MHz). In this work, we assume that within the range of multiple voltage and frequency levels, link circuitry can always function above the noise margin and achieve low BER. Due to the timing uncertainty, during frequency transition, when the receiver is trying to lock in the input clock, link circuitry is disabled.

3 History-based DVS

The policy controlling DVS links has to judiciously trade off power and performance, minimizing the power consumption of the network while maintaining high performance. We propose a distributed history-based DVS policy, where each router port predicts future communication workload based on the analysis of prior traffic, then dynamically adjusts the frequencies (and corresponding voltages) of its communication links to track network load.

3.1 Communication traffic characterization

Communication traffic characteristics can be captured with several potential network measures. We need to explore suitable indicators that are useful for predicting network load. An obvious measure for DVS links is link utilization, defined as follows:

Link utilization

$$LU = \frac{\sum_{t=1}^{H} A(t)}{N}, \quad 0 \leq LU \leq 1 \tag{2}$$

where $A(t) = \begin{cases} 1 & \text{if traffic passes link } i \text{ in cycle } t \\ 0 & \text{if no traffic passes link } i \text{ in cycle } t \end{cases}$
and N is the number of link clock cycles, which is sampled within a history window size H defined in router clock cycles.

Link utilization is a direct measure of traffic workload. A higher link utilization reflects that more data are sent to the next router. Assuming history is predictive, this indicates a higher link frequency is needed to meet the performance requirement. Conversely, lower link utilization implies the existence of more idle cycles. Here, decreasing the link frequency can lead to power savings without significantly hurting performance.

To investigate how predictive link utilization is of network load, we track the utilization of a link within a two-dimensional 8x8 mesh network. Network traffic is generated based on the two-level task workload model discussed in Section 4.3. Figure 3 shows the utilization of this link as sampled every 50 cycles (H=50), across the entire timing simulation, as we increase the load on the network. At low traffic workloads, contention for buffers and links is rare. In this case, link utilization is bounded by flit[2] arrival rate which is slow at low traffic workloads (see Figure 3(a)). As network traffic increases, more flits are relayed between adjacent routers, and the utilization of each corresponding link also increases, as reflected in Figures 3(b) and 3(c). When the network traffic nears the congestion point, resource contention results in flits being stalled in input buffers, since they can be relayed to the next router only if free buffers are available. Limited available buffer space in the succeeding router begins to be a tighter constraint, causing link utilization to decrease. When the network is highly congested, inter-router flit transmission is totally constrained by the availability of free buffers. Link utilization thus starts to dip (Figure 3(d)).

The histograms show that whether the network is lightly loaded or highly congested, link utilization is low. These two extreme network scenarios have vastly different requirements on network latency though. At low network loads, since the flit will not be stalled in the succeeding router, any increase in link delay directly contributes to overall packet latency. At high network loads, flits will be stalled in the next router for a long time anyway. Getting there faster will not help. In this case, link frequency can be decreased more aggressively with minimal delay overhead. Hence, link utilization alone will not be sufficient for guiding the history-based DVS policy.

We thus investigate two other measures – input buffer utilization and input buffer age:

Input buffer utilization

$$BU = \frac{\sum_{t=1}^{H} (F(t)/B)}{H}, \quad 0 \leq BU \leq 1 \tag{3}$$

where $F(t)$ is the number of input buffers that are occupied at time t, and B is the input buffer size.

Input buffer age

$$BA = \frac{\sum_{t=1}^{H} \sum_{i=1}^{D(t)} (t_{d_i} - t_{a_i})}{\sum_{t=1}^{H} D(t)} \tag{4}$$

where $D(t)$ is the number of flits that left the input buffer at cycle t of a history interval H, t_{d_i} is the departure time of flit i from this buffer, and t_{a_i} is the arrival time of flit i at the input buffer.

Input buffer utilization tracks how many buffers in the succeeding router of the link are occupied. Input buffer age determines how long flits stay in these input buffers before leaving. These measures reflect resource contention in the succeeding router. We track the input buffers downstream from the same link reflected in Figure 3. Figures 4 and 5 show their profile as we increase the network load. Under low network traffic, resource contention is low. Few buffers are occupied (Figure 4(a)). Flits also do not stay in the input buffers for long (Figure 5(a)). Hence, both input buffer utilization and input

[2]Flits stand for flow control units, and are fixed-size segments of a packet.

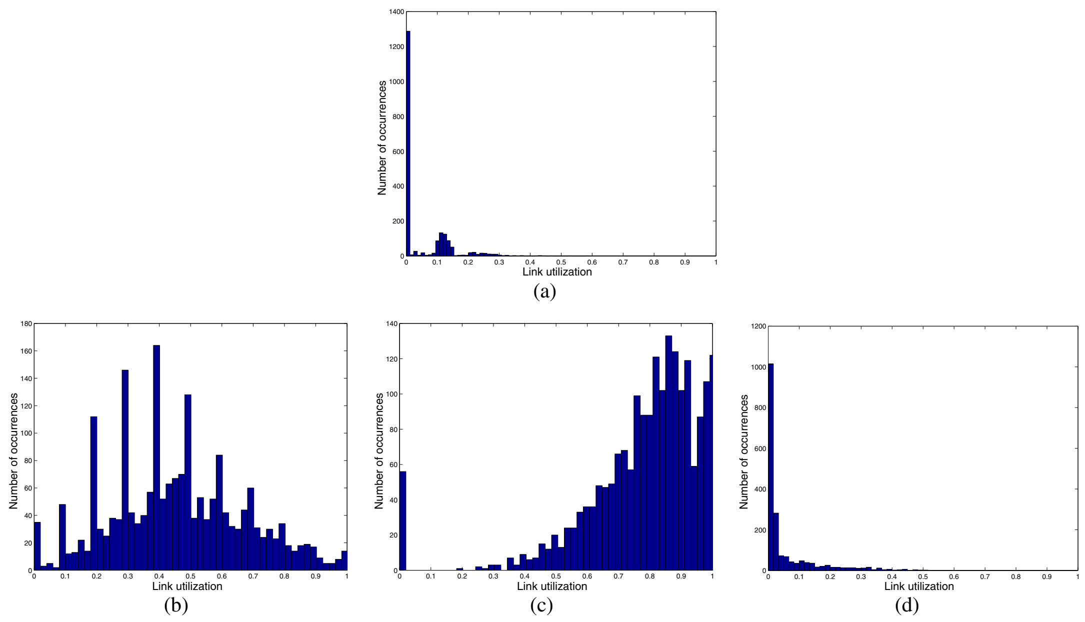


Figure 3. Link utilization profile. From (a) to (d), network traffic increases, with the network congested in (d).

buffer age are low. As input traffic increases, more flits are relayed between adjacent routers, and resource contention increases, being reflected in higher input buffer utilization and age (Figures 4(b) and 5(b)). When the network is highly congested, most of the buffers are filled, and flits are stalled within a router for a long time. Both buffer utilization and age thus rise dramatically (see Figures 4(c) and 5(c)).

Both input buffer utilization and age track the network congestion point well. They behave like an indicator function that rises sharply at high network loads. However, compared with link utilization, input buffer utilization and age are much less sensitive to changes in traffic. From Figure 4, we can see that, from lightly loaded traffic (Figure 4(a)) to high network loads (Figure 4(b)), the average buffer utilization only increases by about 0.1. The average link utilization, on the other hand, changes by more than 0.8 (see Figure 3). Hence, link utilization is much better at tracking nuances in network traffic.

We select link utilization and input buffer utilization as the relevant measures for guiding the history-based DVS policy, as input buffer age has similar characteristics as input buffer utilization, and is harder to capture. We use link utilization as the primary indicator, while input buffer utilization is used as a litmus test for detecting network congestion. Both measures can be efficiently measured and quantified. The hardware implementation for doing so is detailed in Section 3.3.

3.2 History-based DVS policy

Network traffic exhibits two dynamic trends: transient fluctuations and long-term transitions. Our history-based DVS policy filters out short-term traffic fluctuations and adapts link frequencies and voltages judiciously to long-term traffic transitions. It does this by first sampling link (input buffer) utilization within a predefined history window, and second, using exponential weighted average utilization to combine current and past utilization history:

$$Par_{predict} = \frac{weight \times Par_{current} + Par_{past}}{weight + 1} \qquad (5)$$

where $Par_{predict}$ is the predicted communication link (input buffer) utilization. $Par_{current}$ is the link (input buffer) utilization in the current history period, and Par_{past} is the predicted link (input buffer) utilization in the previous history period.

Given the predicted communication link utilization, $LU_{predicted}$, and input buffer utilization, $BU_{predicted}$, the DVS policy dynamically adapts its voltage scaling to achieve power savings with minimal impact on performance. It prescribes whether to increase link voltage and frequency to next higher level, decrease link voltage and frequency to next lower level, or do nothing. Intuitively, when a link is going to be highly utilized, voltage scaling is enabled so that link frequency can be increased to handle the load. Similarly, if a link will be mostly idle, voltage scaling is carried out so that link frequency can drop to save power. Otherwise, voltage scaling is conservatively carried out to minimize impact on performance. The prescribed action depends on four thresholds, two of which are used when the network is lightly loaded (TL_{high}, TL_{low}), and the other two when the network is highly congested (TH_{high}, TH_{low}). In the latter case, since link delay can be hidden, the thresholds prescribe more aggressive power savings. The pseudo-code of our proposed DVS policy is shown in Algorithm 1.

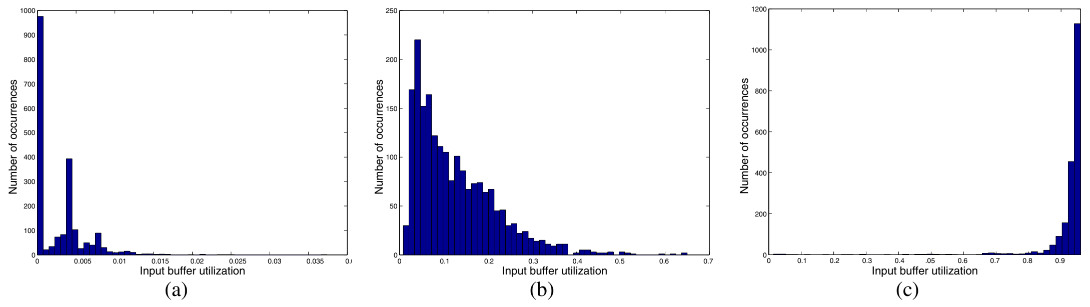


Figure 4. Input buffer utilization profile. From (a) to (c), network traffic increases, with the network congested in (c).

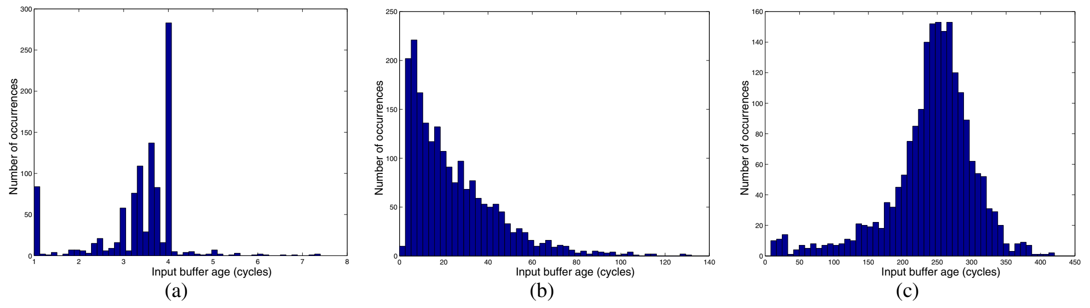


Figure 5. Input buffer age profile. From (a) to (c), network traffic increases, with the network congested in (c).

Algorithm 1 *History-based Dynamic Voltage Scaling*

```
while ($Signal_{VS}$) do
    $LU_{predicted} = (W * LU_{current} + LU_{past})/(W+1)$
    $LU_{past} = LU_{predicted}$
    $BU_{predicted} = (W * BU_{current} + BU_{past})/(W+1)$
    $BU_{past} = BU_{predicted}$
    if ($BU_{predicted} < B_{congested}$) then
        $T_{low} = TL_{low}$    $T_{high} = TL_{high}$
    else
        $T_{low} = TH_{low}$    $T_{high} = TH_{high}$
    end if
    if ($LU_{predicted} < T_{low}$) then
        $NewVol_{link} = Voltage\_table[CurLevel_{link} + 1]$
        $NewFre_{link} = Frequency\_table[CurLevel_{link} + 1]$
    else if ($LU_{predicted} > T_{high}$) then
        $NewVol_{link} = Voltage\_table[CurLevel_{link} - 1]$
        $NewFre_{link} = Frequency\_table[CurLevel_{link} - 1]$
    else
        $NewVol_{link} = Voltage\_table[CurLevel_{link}]$
        $NewFre_{link} = Frequency\_table[CurLevel_{link}]$
    end if
end while
```

3.3 Hardware Implementation

The proposed DVS policy relies only on local link and buffer information. This avoids communication overhead in relaying global information, and permits a simple hardware implementation. Figure 6 shows the hardware realization of our history-based DVS policy. To measure link utilization, a counter at each output port gathers the total number of cycles that are used to relay flits in each history interval. Another counter captures the ratio between the router and link clocks. A simple Booth multiplier combines these two counters to calculate link utilization. For the calculation of the exponential weighted average (see Equation (5)), we set W to 3 so that the division can be implemented as a shift, and the numerator as a shift-and-add operation. Most interconnection network routers use credit-based flow control. Current buffer utilization is thus already available. Two registers store LU_{past} and BU_{past}, which feed the circuit module calculating the exponential weighted average. Finally, some combinational logic performs the threshold comparisons and outputs signals that control the DVS link.

Synthesis with Synopsys Design Compiler results in a final area of about 500 equivalent logic gates per router port. Since this circuit does not lie on the critical path of the router, its delay can be ignored. Combined with real-delay timing simulation, we used Synopsys Power Compiler, a gate-level power simulator, to estimate the power dissipation of this circuit. The power overhead is negligible (less than 3mW per router port).

4 Experimental Results

We next present experimental results to evaluate the efficacy of our approach.

4.1 Network simulator

We implemented an event-driven flit-level interconnection network simulator to evaluate the power-performance of the

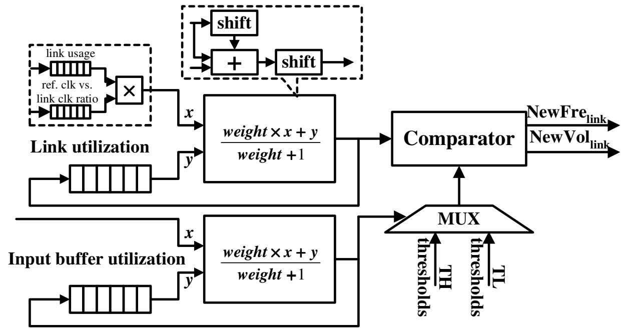


Figure 6. The hardware implementation of the history-based DVS policy. This circuitry sits at each output port of a router, tracking and controlling the multiple links of that port.

proposed history-based DVS policy. The simulator supports *k*-ary *n*-cube network topologies consisting of pipelined virtual-channel (VC) routers [4]. Different routing protocols, both deterministic and adaptive, are supported, with credit-based flow control. Channels are made up of DVS links, with detailed transition delay and energy overheads modeled as described in Section 2.

We assume separate clock domains for the router core and its links. Functional modules within a router operate synchronously and each DVS link has its own local clock, which is dynamically regulated by the history-based DVS policy during simulation. Inter-router flit traversal via DVS links is emulated with message passing to accurately characterize the impact of link delay on the network latency/throughput performance. The simulator has been implemented in C++ (5,200 lines of code) based on standard template libraries (STL), running under Linux OS.

4.2 Experimental Setup

In our experiments, we assume a two-dimensional 8x8 mesh network with 64 1GHz routers, each with two virtual channels and 128 flit buffers per input port. Fixed-length packets of five flits are assumed, with a head flit leading four body flits, and each flit being 32-bits wide. The router is pipelined in a fashion similar to the integrated router of Alpha 21364 processor [17] with 13 pipeline stages. Each port of the router has a channel bandwidth of 32 Gb/s so that a 32-bit flit can be transported across the channel every cycle[3]. The channel consists of eight serial links, each link's frequency (voltage, power) can be scaled from 125MHz (0.9V, 23.6mW) to 1GHz (2.5V, 200mW), with a corresponding bandwidth of 0.5Gb/s to 4Gb/s with 4:1 multiplexing.

[3]We ignore the encoding overhead which requires a higher channel bandwidth, say 40Gb/s for 8b/10b encoding.

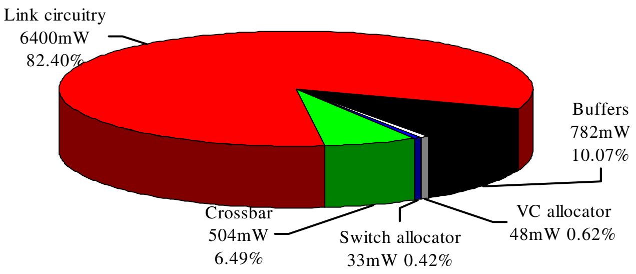


Figure 7. Router power consumption distribution.

Table 1. Parameters of the history-based DVS policy

W	H	$B_{congested}$	TL_{low}
3	200	0.5	0.3
TL_{high}		TH_{low}	TH_{high}
0.4		0.6	0.7

We synthesized the Verilog description of our router to a gate-level netlist in TSMC 0.25μm technology [26] to characterize its power consumption. Combined with real-delay timing simulation, we used Synopsys Power Compiler to estimate the power dissipated by various parts of the router. The profile is shown in Figure 7, based on maximum power consumed by a channel of eight links [12]. In our router, 82.4% of the total power is consumed by the link circuitry. This is higher than the 60% consumed by links in Alpha 21364 routers. The percentage of power consumed by links versus nodes depends largely on the amount of buffering in a router, relative to channel bandwidth. The main purpose of our power characterization is to determine the additional power consumed as a result of a flit staying longer in a router due to slower links. A flit that stays in a router can potentially trigger more arbitrations. However, it does not increase buffer read/write power, nor crossbar power. Since the allocators consume minimal power (81mW), router power consumption does not vary much with and without DVS links. We thus ignore router power consumption when evaluating our history-based DVS policy.

The experimental parameters of our policy are given in Table 1. Latency, throughput and power consumption are the metrics we use to evaluate our technique. Latency spans the creation of the first flit of the packet to ejection of its last flit at the destination router, including source queuing time and assuming immediate ejection. Each simulation is run for 10 million cycles, and the latency of all packets averaged. The saturation throughput of the network is where average packet latency worsens to more than twice the zero-load latency (*i.e.*, the average packet latency when there is no network congestion). Power consumed by the network is derived based on the frequency and voltage levels set for all the channels in the network. Hence, without DVS links, the power consumed by a network is 64 routers $*$ 4 ports $*$ 8 links $* \; 0.2W = 409.6W$.

4.3 Communication traffic modeling and generation

In order to evaluate the performance of our history-based DVS policy, workload models that accurately characterize real-world communication traffic are required. In many applications, communication traffic has high temporal and spatial variance, with dynamic fluctuations and bursts. Workloads that are commonly used in interconnection networks do not capture such phenomena. Random uniformly distributed traffic does not exhibit any spatial or temporal variance, other than that brought about by the topology; and while permutation traffic stresses routing protocols with spatial variance in the workload, it does not capture any temporal variance.

A number of recent empirical studies of traffic measurements from wide-area networks [19], local-area networks [30], and on-chip interconnection networks [27], have convincingly demonstrated that actual network traffic is self-similar or long-range dependent (LRD), which is defined as follows:

Self-similar: Let $(X_t)_{t\in \mathbb{Z}_+}$ be a discrete time stochastic process. Let $r(k)$ denote the autocorrelation function of X_t. X_t is self-similar, or more precisely, asymptotically second-order self-similar, if the following conditions hold:

$$r(k) \sim const \times k^{-\beta} \text{ as } k \to \infty, 0 < \beta < 1 \qquad (6)$$

This autocorrelation function decays polynomially rather than exponentially. LRD processes indicate that while long-range correlations are individually small, the cumulative effect is non-negligible. Such a traffic preserves burstiness over a wide range of time scales, while using short-range dependent processes, such as Poisson, to model the packet injection process results in substantial underestimations of burstiness.

We propose a two-level workload model. The first level consists of concurrent communication task sessions that are generated at random nodes based on the model of sphere of locality [21]. Their inter-arrival time has a Poisson distribution. Task durations are uniformly distributed within a specified range to model interleaved heterogeneous workloads, with tasks lasting on average from 1μs to 1ms. This first-level task model allows us to approximate parallel task communication through an interconnection network, and is widely used to model workloads in distributed embedded systems.

At the second level, within each task session, we assume that packet inter-arrivals are self-similar. According to [15], self-similar traffic can be generated by multiplexing ON/OFF sources that have Pareto-distributed ON and OFF periods. A Pareto distribution is defined as follows:

Pareto distribution: The Pareto distribution with shape parameter β and location parameter a has the following cumulative distribution function:

$$F(x) = P[X \leq x] = 1 - (a/x)^{\beta}, a, \beta \geq 0, x \geq a \qquad (7)$$

In our generation of self-similar traffic for packet injections within a task, we multiplex 128 ON/OFF traffic sources,

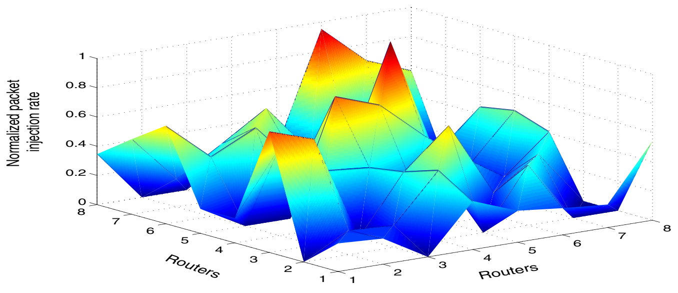


Figure 8. Spatial variance of the injected workload model.

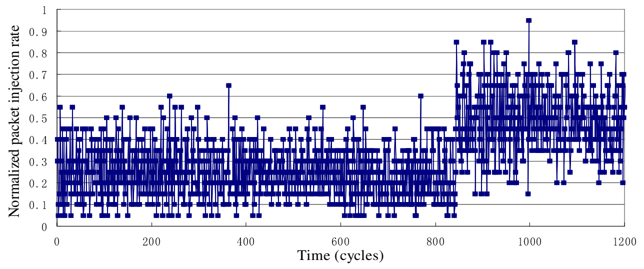


Figure 9. Temporal variance of the injected workload model at one router.

and choose ON and OFF shape parameters to be 1.4 and 1.2 correspondingly. The choice of the above parameters is prompted by measurements on an actual Ethernet traffic performed in [15]. The average packet injection rate across different communication task sessions is uniformly distributed within a specified range.

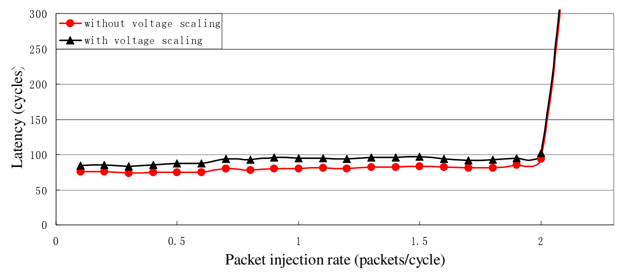
Snapshots shown in Figures 8 and 9 demonstrate the spatial and temporal variance of the workload.

4.4 Performance evaluation of the history-based DVS policy

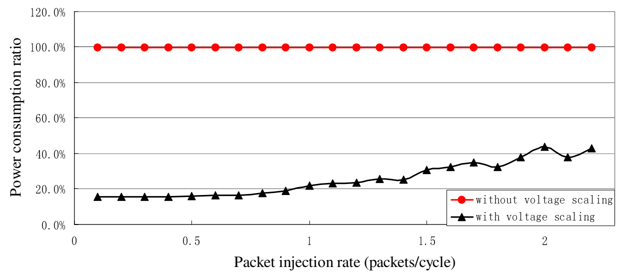
We next evaluate the performance of our policy with conservative DVS links, studying the trade-off between network latency/throughput degradation and dynamic power savings. We then explore the sensitivity of the policy to the thresholds, and investigate the impact of dynamically adjusted thresholds. One of the key goals of this study is to uncover the effect of DVS link characteristics on network power and performance, providing insights that will guide future design of DVS links. We thus explore the effect of varying link characteristics with history-based DVS policy on diverse network workloads.

4.4.1 Performance of history-based DVS with conservative link assumptions

As mentioned before, we generate the communication traffic based on a two-level traffic model. The average number of tasks per cycle is set to 50 or 100. The average task duration is set to $1ms$, and the voltage (frequency) transition la-

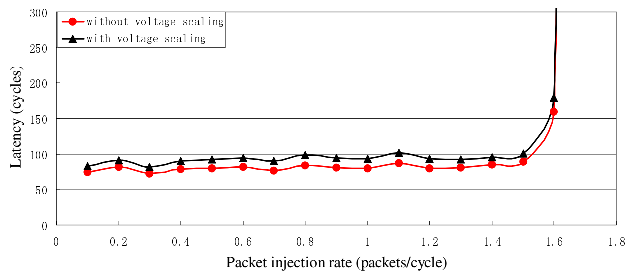


(a) Latency-throughput performance

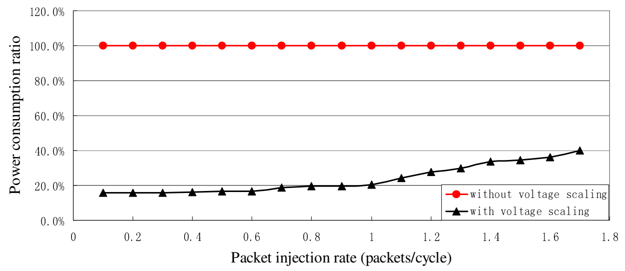


(b) Normalized dynamic power consumption

Figure 10. Network power and performance with and without history-based DVS for 100 tasks.



(a) Latency-throughput performance



(b) Normalized dynamic power consumption

Figure 11. Network power and performance with and without history-based DVS for 50 tasks.

tency between adjacent voltage (frequency) levels to $10\mu s$ (100 link clock cycles). Figure 10(a) shows the latency/throughput performance at increasing packet injection rates while Figure 10(b) plots the power consumption of the network normalized to the non-DVS case. With 100 tasks, history-based DVS increases zero-load latency by 10.8%, and average latency before congestion by 15.2%, while decreasing throughput by less than 2.5%. This moderate impact on performance is accompanied by a large power saving of up to 6.3X (4.6X average).

With 50 tasks, history-based DVS has a moderate impact on performance - increasing latency by 14.7% on average before congestion, and lowering throughput by less than 2.5%, while realizing up to 6.4X power saving (4.9X average) (see Figures 11(a) and 11(b)). The lower throughput as compared to the 100-task workload is due to a higher imbalance in traffic.

When the network is saturated, flits are stalled for a long time in input buffers. Such congested routers show high input buffer utilization and low communication link utilization. In such a scenario, our history-based DVS policy will try to dynamically reduce the frequencies of affected links to decrease power consumption. We track the network with 100 tasks, beyond saturation. Figure 12 shows the power consumption of the network increasing initially as network throughput increases, and dipping thereafter as network throughput decreases. This interesting phenomenon is largely due to the very bursty nature of our communication workload that results in routers in different parts of the network experiencing

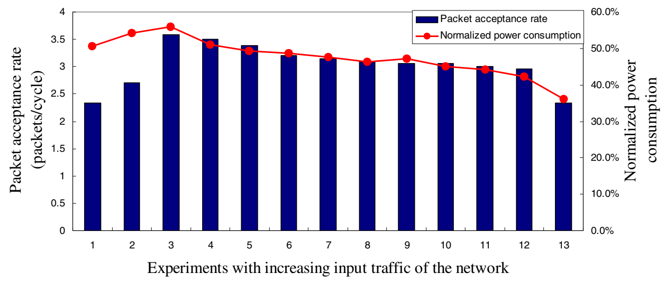


Figure 12. Power consumption and network throughput under network congestion.

widely varying loads over time. When some of the routers are congested, traffic through other routers may still be relatively light. Therefore, as the packet injection rate increases, the overall network throughput may still increase. Link utilization is strongly correlated with network throughput. A higher throughput implies a higher average link utilization in the network. Our distributed history-based DVS policy only decreases the frequencies and voltages of links that are lightly utilized – those connected to the congested routers. It increases the frequencies and voltages of the heavily-used links to meet performance requirements. Hence, only when the entire network becomes highly congested, and overall network throughput starts to decrease, do we see overall reduction in network power.

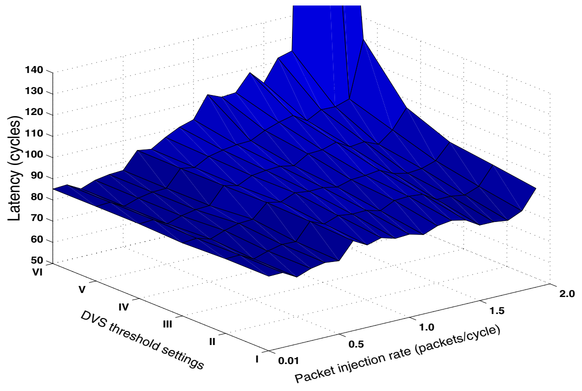


Figure 13. Latency profile under different DVS thresholds.

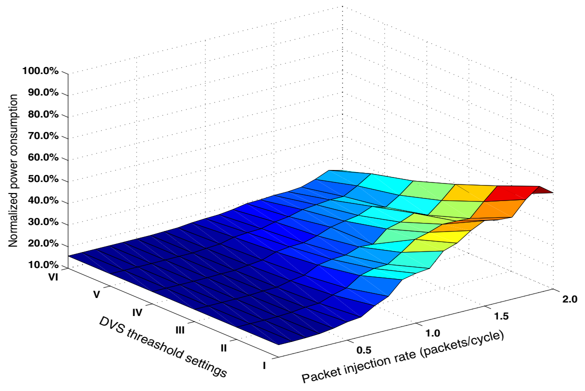


Figure 14. Power consumption profile under different DVS thresholds.

Table 2. Thresholds used in trade-off analysis.

Threshold settings	I	II	III	IV	V	VI
TL_{low}	0.2	0.25	0.3	0.35	0.4	0.5
TL_{high}	0.3	0.35	0.4	0.45	0.5	0.6

By dynamically adjusting the DVS policy to maximize power savings when the network is lightly loaded, and minimizing performance impact when the network is congested, the policy is able to realize substantial power savings without a significant impact on performance. It should be noted that part of the impact on performance is due to our assumptions for DVS links. First, the link is down during frequency scaling. Second, when increasing voltage and frequency, voltage increases first, which takes a long time. Frequency is kept at the original low level during voltage transition.

4.4.2 Effect of thresholds on power-performance of the policy

Power savings and latency-throughput are conflicting optimization metrics. Higher power savings require lower link voltage and frequency. A lower link frequency increases the inter-router traversal latency of packets and flits, and also the buffer turnaround time, which was shown to degrade network throughput [20]. Different applications have different power and performance demands. The policy should be flexible enough to adapt and strike an appropriate balance between these two metrics.

In the history-based DVS policy, a set of threshold parameters can be dynamically changed to adjust the voltage scaling policy. Since latency/throughput performance is the key issue when network traffic is below the congestion point, we change the thresholds TL_{low} and TL_{high} to study the trade-off between power savings and latency/throughput performance under those network traffic conditions. In Table 2, we show the values of the thresholds we set during each simulation.

Under different threshold settings, latency and dynamic power savings are shown in Figures 13 and 14, respectively. The thresholds are used to adjust the voltage scaling policies. Threshold II is a more aggressive voltage scaling policy than threshold I, and so on. From the simulation results, we can see that with more aggressive thresholds, we can obtain more power savings but at the cost of a higher latency and throughput degradation. This points to the possibility of dynamically adjusting threshold settings to trade off power savings and latency/throughput performance.

In Figure 15, we show latency vs. dynamic power savings under the average network packet injection rate of 1.7 packets per cycle. From the figure, we see that an improvement in one metric can only be obtained by degrading the other, verifying that the central goal of DVS link policies is to strike a good balance between power and performance.

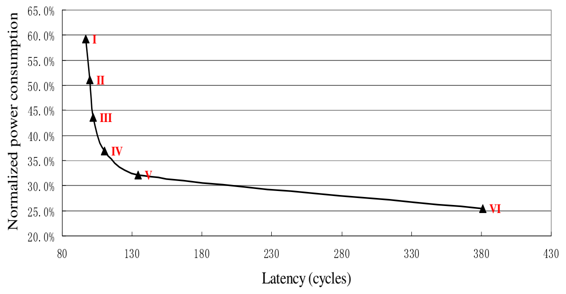
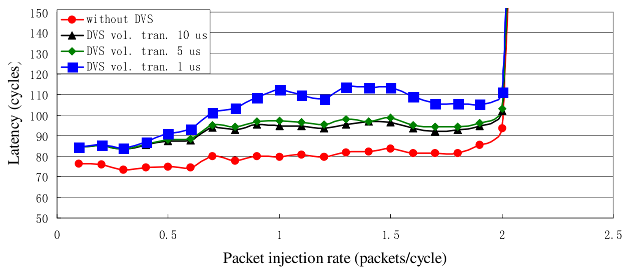


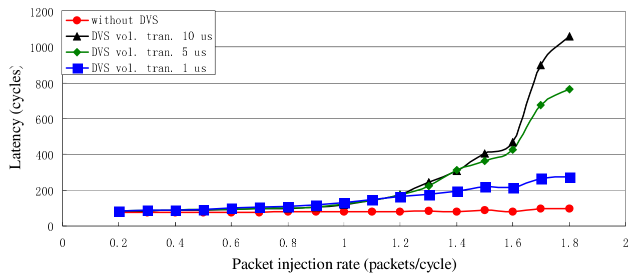
Figure 15. The Pareto curve of latency vs. dynamic power savings.

4.4.3 Effect of DVS links with varying characteristics

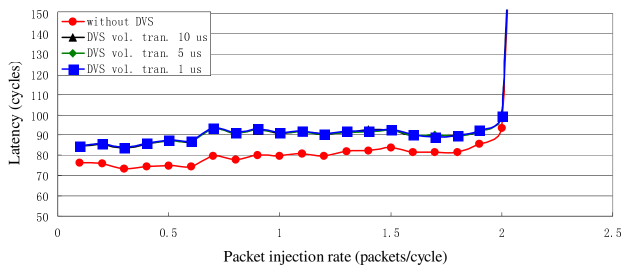
With current DVS technology, voltage transitions may take tens of microseconds. Most dynamic voltage regulators use a Buck converter, where the transition latency is dominated by voltage charge and discharge of the off-chip capacitor. In our DVS model, voltage and frequency transitions are distinct. When network traffic requires higher frequency levels,



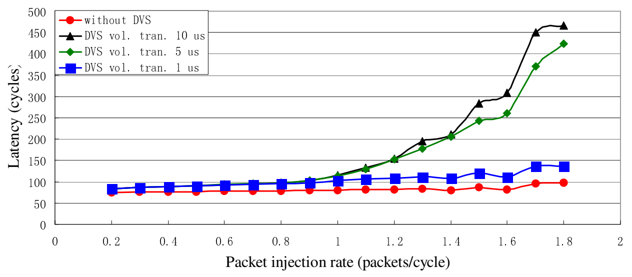
(a) Average task duration of $1ms$, frequency transition takes 100 cycles



(b) Average task duration of $10\mu s$, frequency transition takes 100 cycles



(c) Average task duration of $1ms$, frequency transition takes 10 cycles



(d) Average task duration of $10\mu s$, frequency transition takes 10 cycles

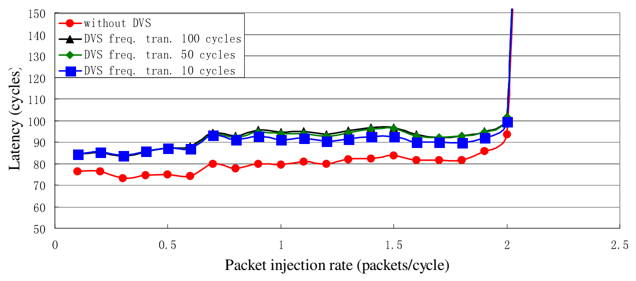
Figure 16. Network performance with DVS links of varying voltage transition rates.

frequency transition can only start after the long voltage transition - links at low frequency levels thus degrade network performance. Although frequency transitions are much faster compared with voltage transitions, the link stops functioning when the receiver is tracking the input clock. Under highly fluctuating network traffic, fast voltage and frequency transitions are critical for quick responses to traffic changes, to maximize dynamic power savings and minimize latency/throughput degradation. This prompted us to explore links with varying voltage and frequency transition rates and their impact on different traffic patterns. In order to generate a range of traffic patterns, we changed the average duration of concurrent task sessions. The motivation is that communication traffic composed of short duration tasks shows a much higher temporal variance than longer duration tasks. The voltage transition delay range is $[1\mu s, 10\mu s]$, frequency transition delay range is [10 cycles, 100 cycles], and average task duration varies within $[10\mu s, 1ms]$.

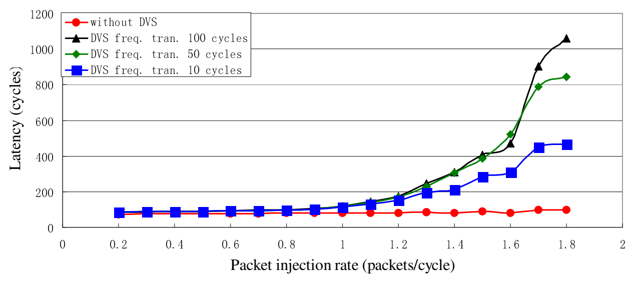
First, we study the impact of voltage transition delay on network performance. In Figures 16(a) and 16(c), the frequency transition delays are 100 and 10 cycles. The average task duration is $1ms$. In this setup, long-range traffic transition is slower than the voltage and frequency transitions. A slow voltage transition only introduces extra latency overhead. In Figure 16(a), with the decrease of voltage transition delay from $10\mu s$ to $1\mu s$, the average network latency increases. This strange phenomenon is due to the fact that the performance of DVS policy is dependent on the ratio of voltage and frequency transition delays. With the link disabled during frequency transition, a more frequent DVS policy may result in worse performance. In current link design, voltage transition is very slow, i.e. 10 μs, which dominates performance effects and constrains the shortest DVS interval. When faster voltage transition becomes possible, determination of DVS intervals should take into account such delay ratios. Compared with Figure 16(a), in Figure 16(c), frequency transition is much faster. Hence, we cannot see such performance degradation. In Figures 16(b) and 16(d), frequency transition takes 100 and 10 cycles respectively, while tasks are much shorter, at $10\mu s$. Here, the impact of voltage transition delay on network throughput becomes pronounced. A longer voltage transition further postpones the increase of link frequency, which exacerbates network congestion and affects network throughput.

Second, we study the impact of frequency transition delay on network performance. In Figures 17(a) and 17(c), the voltage transition delay are $10\mu s$ and $1\mu s$ respectively. It demonstrates that when the average task duration is $1ms$, voltage and frequency transitions are still fast enough to track traffic changes. Frequency transition only introduces extra latency overhead. In Figures 17(b) and 17(d), when the average task duration is $10\mu s$, voltage and frequency transitions respond more slowly to traffic changes, thus degrading network throughput.

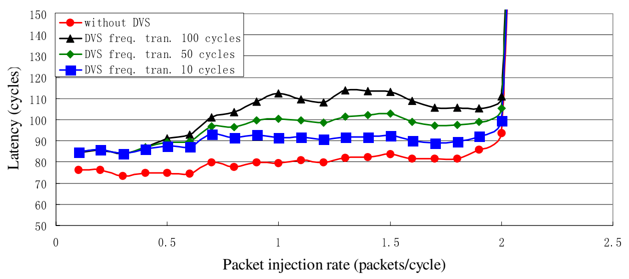
Network power is much less sensitive to varying voltage and frequency transition rates than network latency and throughput. The reason is that the same history-based DVS policy is used here for comparison purposes. Faster voltage and frequency transition rates can respond more quickly



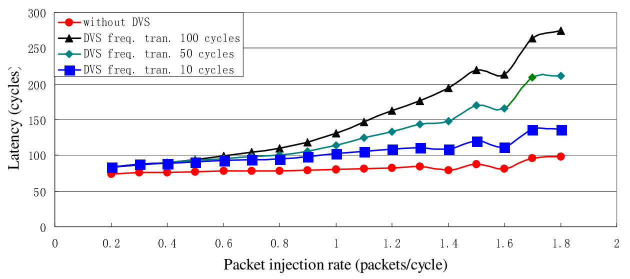
(a) Average task duration of $1ms$, voltage transition takes 10 μs



(b) Average task duration of $10\mu s$, voltage transition takes 10 μs



(c) Average task duration of $1ms$, voltage transition takes 1 μs



(d) Average task duration of $10\mu s$, voltage transition takes 1 μs

Figure 17. Network performance with DVS links of varying frequency transition rates.

to both a decrease and increase of traffic workload. More power savings achieved in the former case are offset in the latter. However, under the same latency/throughput penalty constraint, more aggressive power optimization policies can be applied to faster DVS links. This will lead to more power savings as DVS link technology advances.

5 Conclusions

In this paper, we motivated the use of DVS with links as a power optimization mechanism for interconnection networks. Just as DVS in microprocessors exploits the variance in processor utilization to tune processor frequency for power savings, DVS in links allows link frequency to be tuned for power efficiency as network utilization varies. Links that automatically track the lowest suitable supply voltage at different link frequencies were recently proposed [12, 29], and shown capable of delivering up to 10X power savings. These variable-frequency links demonstrate the feasibility of DVS links for power optimization of interconnection networks.

DVS links need to be regulated by a good DVS policy that maximizes dynamic power savings while minimizing latency-throughput degradation. We proposed a history-based DVS policy that leverages past network history to predict future network needs, judiciously controlling the frequency (and voltage) of links to track actual network utilization. We first studied the behavior of various measures under varying network workloads to identify indicators that are predictive of network utilization. Our findings point to a combination of two measures – link utilization and input buffer utilization, to characterize network traffic and guide the DVS policy. Our policy is distributed, implemented at each output port of a router. This avoids the communication overhead of relaying global information and permits a simple hardware implementation which is both power-and area-efficient.

Our evaluation of history-based DVS uses a realistic two-level task workload model that effectively captures the high spatial and temporal variance in real network traffic. To explore the potential of DVS links, we make very conservative assumptions of DVS link characteristics, assuming it transitions very slowly between voltage levels, and that it does not function during a transition. Despite these conservative assumptions, experiments show history-based DVS providing up to 6.3X dynamic power savings (4.6X on average). This is at the cost of a small impact on performance – 15.2% degradation in average network latency before network saturation and 2.5% reduction in network throughput.

We investigated the impact of DVS links with different voltage and frequency transition rates on overall network power-performance and found that a faster transition rate more effectively tracks bursty traffic workloads, enabling a lower latency-throughput penalty. Hence, future technology improvements in DVS links can yield even better interconnection network power-performance. Designers of variable-frequency links are now actively researching extensions of their links to incorporate the fast, dynamic transitions of DVS links.

As power becomes increasingly as, if not more, important than performance in interconnection networks, there is a clear need for mechanisms that target network power efficiency while maintaining good latency/throughput performance. Our research demonstrates the effectiveness of DVS with links as a power optimization mechanism for interconnection networks.

Acknowledgments

The authors would like to thank Gu-Yeon Wei of Harvard University and Jaeha Kim of Stanford University for their help in our understanding of variable-frequency links, and for validating and extracting DVS link characteristics. We also wish to thank Edward Lee of Velio Communications for insightful technical discussions on high-speed signaling. In addition, we appreciate the use of Kai Li's research group's clusters of workstations at Princeton for our network simulations. This work was supported in part by DARPA under contract no. DAAB07-00-C-L516.

References

[1] N. J. Boden, D. Cohen, R. E. Felderman, A. E. Kulawik, C. L. Seitz, J. N. Seizovic, and W.-K. Su. Myrinet - A Gigabit-per-second local-area-network. *IEEE Micro*, 15(1):29–36, Feb. 1995.

[2] T. Burd and R. Brodersen. Design issues for dynamic voltage scaling. In *Proc. International Symposium on Low Power Electronics and Design*, pages 9–14, July 2000.

[3] A. Chandrakasan, W. Bowhill, and F. Fox. Design of high-performance microprocessor circuits. *IEEE Press and John Wiley & Sons, Inc.*, Sept. 2001.

[4] W. J. Dally. Virtual channel flow control. *IEEE Transactions on Parallel and Distributed Systems*, 3(2):194–205, Mar. 1992.

[5] W. J. Dally, P. Carvey, and L. Dennison. The Avici Terabit switch/router. In *Proc. Hot Interconnects 6*, Aug. 1998.

[6] W. J. Dally and B. Towles. Route packets, not wires: On-chip interconnection networks. In *Proc. Design Automation Conference*, pages 684–689, June 2001.

[7] S. Dhar, D. Maksimovic, and B. Kranzen. Closed loop adaptive voltage scaling controller for standard-cell ASICs. In *Proc. International Symposium on Low Power Electronics and Design*, pages 120–125, Aug. 2002.

[8] IBM blue logic high-speed SERDES family of cores. `http://www.ibm.com`.

[9] The InfiniBand Trade Alliance architecture. `http://www.infinibandta.org`.

[10] Intel XScale microarchitecture. `http://developer.intel.com/design/intelxscale/`.

[11] N. K. Jha. Low power system scheduling and synthesis. In *Proc. International Conference on Computer-Aided Design*, pages 259–263, Nov. 2001.

[12] J. Kim and M. Horowitz. Adaptive supply serial links with sub-1V operation and per-pin clock recovery. In *Proc. International Solid-State Circuits Conference*, Feb. 2002.

[13] J. J. Kim, S.-B. Lee, T.-S. Jung, C.-H. Kim, S.-I. Cho, and B. Kim. A low-jitter mixed-mode DLL for high-speed DRAM applications. *IEEE Journal of Solid-State Circuits*, 35(10):1430–1436, Oct. 2000.

[14] M.-J. E. Lee, W. J. Dally, and P. Chiang. Low-power area-efficient high-speed I/O circuit techniques. *IEEE Journal of Solid-State Circuits*, 35(11):1591–1599, Nov. 2000.

[15] W. Leland, M. Taqqu, W. Willinger, and D. Wilson. On the self-similar nature of ethernet traffic (extended version). *IEEE/ACM Transactions on Networking*, 2(1):1–15, Feb. 1994.

[16] Mellanox Technologies performance, price, power, volume metric (PPPV). `http://www.mellanox.com/products/shared/PPPV.pdf`.

[17] S. S. Mukherjee, P. Bannon, S. Lang, A. Spink, and D. Webb. The Alpha 21364 network architecture. *IEEE Micro*, 22(1):26–35, Jan./Feb. 2002.

[18] C. Patel, S. Chai, S. Yalamanchili, and D. Schimmel. Power-constrained design of multiprocessor interconnection networks. In *Proc. International Conference on Computer Design*, pages 408–416, Oct. 1997.

[19] V. Paxson and S. Floyd. Wide-area traffic: The failure of Poisson modeling. *IEEE/ACM Transactions on Networking*, 3(3):226–244, June 1995.

[20] L.-S. Peh and W. J. Dally. Flit-reservation flow control. In *Proc. International Symposium on High Performance Computer Architecture*, pages 73–84, Jan. 2000.

[21] D. A. Reed and D. C. Grunwald. The performance of multicomputer interconnection networks. *IEEE Computer*, 20(6):63–73, June 1987.

[22] S. Sidiropoulos, D. Liu, J. Kim, G. Wei, and M. Horowitz. Adaptive bandwidth DLLs and PLLs using regulated supply CMOS buffers. In *Proc. IEEE Symposium on VLSI Circuits*, pages 124–127, June 2000.

[23] A. Stratakos. High-efficiency low-voltage DC-DC conversion-for portable applications. Ph.D. Thesis, Univ. of California, Berkeley, June 1998.

[24] M. B. Taylor *et al.*. The RAW microprocessor: A computational fabric for software circuits and general-purpose programs. *IEEE-MICRO*, 22(2):25–35, Mar./Apr. 2002.

[25] Transmeta Crusoe microarchitecture. `http://www.transmeta.com`.

[26] TSMC 0.25 μm process 2.5-Volt SAGE$^{\text{TM}}$ standard cell library. `http://www.mosis.com`.

[27] G. Varatkar and R. Marculescu. Traffic analysis for on-chip network design of multimedia applications. In *Proc. Design Automation Conference*, pages 795–800, June 2002.

[28] H.-S. Wang, X.-P. Zhu, L.-S. Peh, and S. Malik. Orion: A power-performance simulator for interconnection networks. In *Proc. International Symposium on Microarchitecture*, Nov. 2002.

[29] G. Wei, J. Kim, D. Liu, S. Sidiropoulos, and M. Horowitz. A variable-frequency parallel I/O interface with adaptive power-supply regulation. *Journal of Solid-State Circuits*, 35(11):1600–1610, Nov. 2000.

[30] W. Willinger, M. S. Taqqu, R. Sherman, and D. V. Wilson. Self-similarity through high-variability: Statistical analysis of ethernet LAN traffic at the source level. In *Proc. ACM SIGCOMM*, pages 100–113, Sept. 1997.