Leakage Power Modeling and Optimization in Interconnection Networks

Xuning Chen and Li-Shiuan Peh
Dept. of Electrical Engineering,
Princeton University, NJ08544
{xuningc,peh}@ee.princeton.edu

ABSTRACT

Power will be the key limiter to system scalability as interconnection networks take up an increasingly significant portion of system power. In this paper, we propose an architectural leakage power modeling methodology that achieves 95-98% accuracy against HSPICE estimates. When applied to interconnection networks, combined with previous proposed dynamic power models, we gain valuable insights on total network power consumption. Our modeling shows router buffers to be a prime candidate for leakage power optimization. We thus investigate the design space of power-aware buffer policies, propose a suite of policies, and explore the impact of various circuits mechanisms on these policies . Simulations show power-aware buffers saving up to 96.6% of total buffer leakage power.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network architecture and design

General Terms

Measurement, Design

Keywords

Leakage power, interconnection networks, power optimization

1. INTRODUCTION

As power becomes the dominant constraint in many computer systems, research into power-efficient systems has thrived. In many of these systems, the network fabric is a significant consumer of power. This has resulted in researchers modeling [9] and optimizing [8, 10] the dynamic power consumption of interconnection networks. As technology scales to deep sub-micron processes, leakage power becomes increasingly significant as compared to dynamic power. There is thus a growing need to characterize and optimize network leakage power as well.

In this paper, we propose a new architectural methodology for estimating leakage power that distinguishes technology-dependent from technology-independent variables, providing the flexibility of an architecture-level power model where architectural parameters suffice, together with the rigorous accuracy of a low-level model. An accurate model allows architects to rapidly estimate leakage power as they iterate across alternative designs. We applied our methodology to both on-chip and chip-to-chip interconnection networks, and validated our estimates against HSPICE, obtaining 95-98% accuracy.

By combining our proposed leakage power model with a dynamic power model [9], we were able to gather insights on the total power consumption of networks, characterizing the power breakdown of various network components as technology scales. Our modeling guided us to investigate and propose power-aware buffers as a leakage power optimization technique. We then explore the design space of architectural policies for power-aware buffers, and propose a suite of techniques that are able to save up to 96.6% of total buffer leakage power.

ISLPED'03, August 25–27, 2003, Seoul, Korea.

2. AN ARCHITECTURAL LEAKAGE POWER MODELING METHODOLOGY

Leakage current has five basic components: reverse biased pn junction current, sub-threshold leakage current, gated induced drain leakage, punch through current and gate tunneling current. These leakage current components have an almost linear relation with transistor width. For instance, subthreshold current I_{sub} which currently dominates leakage current is defined as follows [1]:

$$I_{sub} = I_0 \left[1 - \exp(-\frac{V_{ds}}{V_t})\right] \exp(\frac{V_{gs} - V_{th} - V'_{off}}{nV_t}) \quad (1)$$

$$I_0 = \mu \frac{W}{L} \sqrt{\frac{q\varepsilon_{si} \cdot NDEP}{2\Phi_s}} V_t^2 \quad (2)$$

For a given circuit type i and input state s at a process technology, subthreshold current is almost proportional to the transistor width W. Although, different components will have different impact on leakage current as technology scales, e.g. gated induced drain current will become more and more significant, the total leakage current still keeps an almost linear relation with transistor width.

So to derive an architectural leakage power model, we can separate the technology-independent variables such as transistor width from those that stay invariant for a specific process technology:

$$I_{leak}(i,s) = \frac{W(type(i,s))}{L} \cdot I'_{leak}(i,s) \quad (3)$$

where I_{leak} is total leakage current. I'_{leak} is leakage current per unit transistor width over length. $W(type(i,s))$ refers to the transistor width of NMOS when NMOS determines the leakage current (*i.e.* $type(i,s)$ is N), or PMOS (when $type(i,s)$ is P). As transistor width has a negligible effect on $I'_{leak}(i,s)$, $I'_{leak}(i,s)$ is fixed for a given circuit type and input state under certain technology and temperature. With this approximation, armed with I'_{leak} for various kinds of circuit components at different input states, architects can estimate the leakage power for architectural units composed from these circuit components. Our proposed modeling methodology is as follows:

1. Identify the fundamental circuit components, and derive $I'_{leak}(i,s)$ for each at different input states. Examples are single NMOS and PMOS transistors, NAND gates, inverters, etc.
2. Define major architectural building blocks. For interconnection networks, typical building blocks will be buffers, crossbars, arbiters and links [9]. For microprocessors, suitable building blocks will be cache lines, adders, etc.
3. Identify the distribution of the input states based on operation characteristics or simulation and derive architectural equations that estimate the leakage power for each building block.

We believe this is the first leakage power modeling methodology that truly separates technology-dependent and independent variables. In [2], a single k_{design} is used to reflect the composition of device types (N/P), geometries (W/L), states (on/off), and stacking factors. As a result, k_{design} is extremely sensitive to changes in any of the variables and the impact of architectural parameters hard to isolate. In [6], $P^{lib}_{leak} = \chi^{lib} \cdot Cells^{S^{lib}}$ is used to estimate the leakage power in an ASIC design environment, where χ^{lib}, S^{lib} are technology-dependent parameters derived through experiments and "Cells" is the number of cells in the design. This model targets a later design stage then the architectural stage, when designers explore various circuit designs for a selected architecture.

2.1 Derivation of I'_{leak}

For each component i, we simulate $I'_{leak}(i,s)$ using HSPICE and the Berkeley Predictive Technology Model [1] for the range of process technologies and associated parameters listed in Table 1. Table 2 lists the $I'_{leak}(i,s)$ simulated for each fundamental circuit component i (Leakage currents differs at different states due to staking and body bias effects). Circuit structures can then be hierarchically composed from these fundamental circuit components.

Table 1: Parameters for various technologies.

	0.18μm	0.10μm	0.07μm
V_{th0}	0.4V	0.3V	0.2V
V_{dd}	1.8V	1.2V	0.9V

Table 2: $I'_{leak}(i,s)$ for each fundamental circuit component i at different input states s at temperature 80^oC. $Type(i,s)$ indicates if the NMOS or PMOS transistor is dominant in determining leakage current.

			$I'_{leak}(i,s)$		
		type	Process Technology, T		
i	s	(i,s)	0.18μm	0.10μm	0.07μm
NMOS	0	N	7.9e-9	10.9e-9	67.6e-9
PMOS	1	P	4.0e-9	9.7e-9	80.4e-9
INV	0	N	7.9e-9	10.9e-9	67.6e-9
	1	P	4.0e-9	9.7e-9	80.4e-9
NAND2	00	N	0.3e-9	0.4e-9	9.6e-9
	01	N	7.9e-9	10.8e-9	46.0e-9
	10	N	4.7e-9	5.1e-9	44.0e-9
	11	P	8.1e-9	19.4e-9	159.5e-9
NOR2	00	N	15.9e-9	21.7e-9	133.8e-9
	01	P	3.6e-9	5.9e-9	45.3e-9
	10	P	4.3e-9	9.7e-9	77.5e-9
	11	P	0.9e-9	0.7e-9	5.9e-9

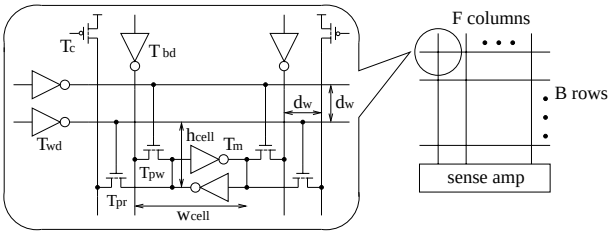

Figure 1: A FIFO buffer with 1 read port and 1 write port (adapted from [4]). T_c is the pre-charging transistor, T_{wd} the wordline driver, T_{bd} the write bitline driver, T_m the memory cell inverter, and T_{pr} and T_{pw} the pass transistors connecting read and write ports to memory cells respectively.

2.2 Leakage power modeling of router buffers

We applied our methodology to the major building blocks of interconnection networks as identified in [9] – buffers, crossbars, arbiters, and links. Here, we walk through our modeling of router buffers to demonstrate the methodology. Fig. 1 sketches the circuit structure of a router buffer pool with B flit[1] buffers, each F bits wide, with P_r read ports and P_w write ports. It shows a FIFO buffer that is composed of the fundamental circuit components of PMOS, NMOS transistors and inverters. Dimensions of the circuit structure such as h_{cell}, w_{cell}, d_w are estimated by Orion [9] from architectural parameters.

Input state probabilistic analysis. Next, we analyze the probability distribution of each input state of a circuit component by examining how architectural units function . For instance, the word line inverter T_{wd} is set ($s = 0$) whenever that buffer/row is read or written. Thus, at any point in time, only one out of B word line inverters will be set. Hence, $I_{leak}(T_{wd}) = \frac{1}{B}\frac{W(type(INV,0))}{L} \cdot I'_{leak}(INV,0) + \frac{B-1}{B}\frac{W(type(INV,1))}{L} \cdot I'_{leak}(INV,1)$.

Basically, given these $I'_{leak}(i,s)$, and the probabilities of

[1]A flit is short for flow control unit, and is a fixed-length segment of a packet.

each input state $Prob(i,s)$, the leakage current for a building block is:

$$I_{leak}(Block) = \sum_i \sum_s Prob(i,s) \frac{W(type(i,s))}{L} \cdot I'_{leak}(i,s) \quad (4)$$

where $W(type(s))$ refers to the transistor width of NMOS (when $type(s)$ is N) or PMOS (when $type(s)$ is P).

Input state simulation. Input states can also be tracked through network simulation.

$$I_{leak}(Block,t) = \sum_i \frac{W(type(i,s(t)))}{L} \cdot I'_{leak}(i,s(t)) \quad (5)$$

where, $I_{leak}(Block,t)$ is the leakage current at time t, and $s(t)$ is the state of circuit type i at time t within this circuit block.

Finally, we can estimate the total leakage current of a router buffer (Eq. 6) while its leakage power is leakage current multiplied by supply voltage (Eq. 7).

$$\begin{aligned} I_{leak}(buffer) =& (P_r + P_w) B I_{leak}(T_{wd}) + 2 P_w F I_{leak}(T_{bd}) \\ &+ 2BF I_{leak}(T_c) + 2BF I_{leak}(T_m) \\ &+ 2BF(P_w I_{leak}(T_{pw}) + P_r \cdot I_{leak}(T_{pr})) \quad (6) \\ P_{leak}(buffer) =& I_{leak}(buffer) \cdot V_{dd} \quad (7) \end{aligned}$$

2.3 Validation

We validated our model with HSPICE simulation of each complete functional unit of a chip-to-chip router (crossbar, arbiter, and buffers) in $0.07\mu m$ technology. Leakage currents under different input states were estimated with our model and compared with the leakage currents obtained from HSPICE simulation for the same functional unit with the same input states, the exact structure and feature sizes. For instance, a 5-by-5 matrix crossbar unit has 5 data inputs and 25 control signals. The combination of their values determine the input state of the crossbar and thus the leakage current. For such functional units with a vast number of possible input states, we select a random sample of typical input states for validation. The accuracy of our model for these functional units is computed by averaging across different input states. Table 3 shows mean and standard deviation of our model's error in $0.07\mu m$ technology compared with HSPICE simulation. Since leakage current is large at $0.07\mu m$, we expect the magnitude of error to be larger than that in earlier process technology.

Table 3: Validation of our model vs. HSPICE simulation for each major building block of a router.

	Buffers(%)	Crossbar(%)	Arbiter(%)
Average error	5.0	1.6	1.8
Standard deviation	1.1	2.9	3.0

3. DYNAMIC AND LEAKAGE POWER CHARACTERIZATION OF INTERCONNECTION NETWORKS

Combined with Orion, an architectural dynamic power model for networks [9], we characterized the total power consumption of both an on-chip network and a chip-to-chip network. The on-chip network is parameterized as in [3], with a 4-by-4 mesh network on a $12mm^2$ chip, each node clocked at 1GHz, with 5 input/output ports (one of which is the injection/ejection port), 64 flit buffers per input port (each flit 128 bits wide), connected with a 5-by-5 matrix crossbar and 5 5:1 arbiters. The router in the chip-to-chip network has 256 128-bit flit buffers per input port instead, other parameters remaining the same as that in the on-chip network. The feature size of the transistors is derived by Orion [9] from architectural parameters based on the timing delay requirements and assuming minimum area.

Effect of process technology. Tables 4 and 5 show the estimates for a router in a chip-to-chip and on-chip network respectively at 50% flit arriving rate in 1s at 80^oC. As technology scales, leakage power becomes increasingly significant, starting from 2.5% of total (leakage+switching) power at current 0.18μm technology, to a hefty 60% at 0.07μm technology if clock frequency is kept invariant for the chip-to-chip network. Even assuming doubling clock frequencies as we scale process technology, leakage power remains a significant 27% at 0.07μm. Though the on-chip network has fewer storage elements, leakage power still rises to a significant 21% at 0.07μm, assuming clock frequency doubles each process generation.

Table 4: Dynamic and leakage power estimates of a router in a chip-to-chip network.

Process	Leakage power(W)	Switching power(W)
0.18μm	0.024	0.950(1GHz)
0.10μm	0.039	0.350(1GHz), 0.700(2GHz)
0.07μm	0.278	0.185(1GHz), 0.740(4GHz)

Distribution of leakage power between router and links. From Table 5, it is evident that full-swing on-chip link drivers and wires consume substantial dynamic power, overwhelming that of the router core in 0.10 and 0.07μm processes. However, when you look at leakage power consumption of router vs. links, the converse is true. As wires do not dissipate leakage power, the leakage power consumption of just the drivers is minimal, compared to that of the router core. This prompted us to delve into a leakage power breakdown of various functional units within an on-chip router.

Breakdown of leakage power within a router. Fig. 2 shows the leakage power consumed by the various major functional units of an on-chip router and its links at different process technologies. It shows buffers consuming approximately 64% percent leakage power of the total node (router+link) for all process technologies, standing as the largest leakage power consumer. Our characterization highlights router buffers as a prime candidate for leakage power optimization.

4. POWER-AWARE BUFFERS

As interconnection networks experience significant temporal and spatial variance in workload that leads to highly varying buffer utilization, we propose power-aware buffers as an architectural technique for leakage power optimization in interconnection networks –*i.e.* buffers that regulate their own leakage power consumption based on actual utilization.

Table 5: Dynamic and leakage power estimates of an on-chip router and its links.

Process	Leakage power		Switching Power	
	Router (W)	Links (W)	Router (W)	Links (W)
0.18μm	0.0090	0.0004	0.9334 (1GHz)	0.7066(1GHz)
0.10μm	0.0151	0.0008	0.3513(1GHz), 0.7025(2GHz)	0.5018(1GHz), 1.0036(2GHz)
0.07μm	0.1087	0.0057	0.1052(1GHz), 0.4108(4GHz)	0.1389(1GHz), 0.5556(4GHz)

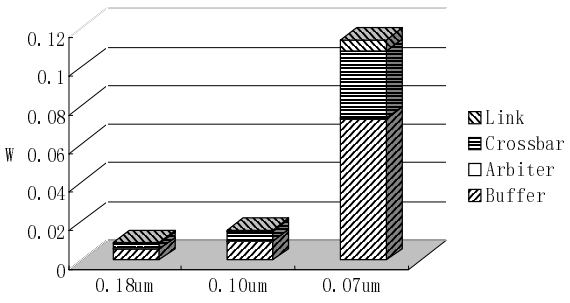

Figure 2: Leakage power distribution across the major functional units of an on-chip router: buffers, arbiters, crossbar and links. Arbiter leakage power is negligible and not visible in the figure.

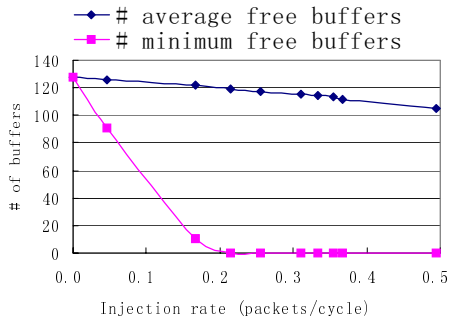

Figure 3: Average and minimum number of idle buffers out of 128flits/buffer.

To explore the potential of power-aware buffers, we first characterize network buffer utilization with the traffic model proposed in [8], with Poisson task inter-arrival rate, and self-similar packet inter-arrival rates within each task session. This workload exhibits the high temporal and spatial variance present in many real-life networks. We simulate the chip-to-chip network described in Sec. 3 (2 virtual channels per port). Fixed-length packets of 20 flits are assumed. Fig. 3 graphs the average and minimum number of idle buffers as traffic increases. As expected, a large number of buffers is left idle at low injection rates. Interestingly, while there are routers whose buffers are fully-occupied (minimum number of idle buffers = 0) at high network load, average buffer utilization remains rather low, with about 85% idle buffers. This is reflective of the high variance in the workload that results in a large gap between average and maximum network utilization that is inherent in many actual workloads. Clearly, placing these idle buffers in an inactive mode that uses less leakage power will result in significant leakage power savings.

4.1 Power-aware buffer policy design

A router buffer is utilized in a stream-like fashion. When a flit enters a router, it gets written into an unoccupied buffer, and sits there while a series of router operations is triggered: routing, virtual-channel allocation and switch allocation. When it is scheduled to leave the router, the flit is read from the buffer pool, and the buffer is then marked as unoccupied and released back to the free list, ready to be reused when a new flit enters the router.

We term a policy that turns a buffer to inactive mode only when it's unoccupied *single* and one that switches a buffer to inactive mode anytime it's not being accessed, *i.e.* when it's both unoccupied and occupied, *double*. To evaluate the effectiveness of any policy, we need a yardstick – we define two theoretically ideal, though unachievable, policies: *Ideal-Single*, that reduces leakage power to zero instantly for buffers that are unoccupied with no additional power overhead, and *Ideal-Double*, that does so similarly for buffers when they are both unoccupied and not being accessed.

A power-aware buffer policy can be oblivious, *i.e.* it does not take current buffer utilization or workload into account; or adaptive, tuning the policy according to current utilization. It can also be conservative, making sure network performance is not impacted, vs. aggressive, targeting as much leakage power savings as possible, even if this comes at the expense of network performance.

Fig. 4 shows the design space of power-aware buffer policies that we envision, and several simple policies that we propose at each design point. Each policy can target either *single* or *double* leakage power savings. First, we propose a conservative policy, Lookahead, that obliviously places buffers in low-leakage mode and wakes them up N cycles before they are accessed. When a flit is read from the buffer queue, that buffer will be switched to the low-leakage inactive mode (if there are more than N free buffers), and when a flit arrives and is written into a router buffer at the tail of the queue, the buffer that is N cells ahead will be switched to normal operating mode. The policy is conservative as it sets the lookahead window of N to the number of cycles needed to switch a buffer from inactive to active mode (transition delay), so a free buffer will always be available when flits arrive, and network performance will never be affected. Clearly, if the buffer size B is less than N, our policy will result in no leakage power savings. An aggressive variant of this policy, Lookahead-Agg simply shortens N to less than the transition delay, trading off performance for higher leakage power savings. Our implementation of Lookahead inserts a newly freed buffer back at the head of the free list, so an active buffer has the highest chance of reuse, minimizing the impact on network performance significantly. A simple adaptive policy, we call Predictive, uses prior buffer utilization history to predict future usage, adjusting the lookahead window N accordingly. We use a

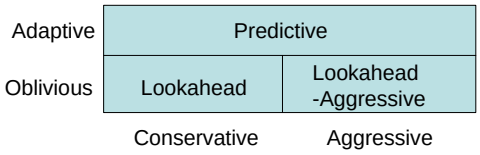

Figure 4: Design space of power-aware buffer policies.

simple statistic – when there are more writes than reads to a buffer in a time window W, N is incremented till it hits an upper-bound N_{high}. Otherwise, it is decremented to a lower-bound N_{low}. The intuition is that when buffer writes outnumber reads, the buffer pool is building up, with fewer and fewer free buffers, so an adaptive policy should be less aggressive in switching buffers to inactive mode in order to enhance network performance. Conversely, when more flits are leaving rather than entering the router, an adaptive policy can more aggressively switch off buffers, guessing that fewer will be needed.

4.2 Circuit-level mechanisms

Power-aware buffers require circuit-level mechanisms that allow buffers to be put into inactive mode for leakage power savings. Several circuit-level mechanisms have been proposed for leakage power savings in SRAMs [4, 7], targeted for microprocessor caches. Since router buffers are usually constructed with SRAMs, these can be readily applied to power-aware buffers.

The characteristics of circuit-level mechanisms that are critical to power-aware buffers are: (1) transition delay - the time it takes to switch a buffer between the normal operating mode and the inactive mode; (2) transition energy - the dynamic energy incurred each time to effect a transition; (3) leakage power savings - the difference between the leakage power incurred at normal operating mode and that at inactive mode; and (4) data preservation - whether the inactive mode preserves the contents of the SRAMs, *i.e.* whether this circuit technique can be applied to both *single* and *double* power-aware buffer policies.

In this paper, we choose two circuit-level mechanisms with fairly different characteristics – Drowsy [4], and Gated V_{dd} SRAMs [7]. Drowsy SRAMs have faster transition delays than Gated SRAMs, preserves data content, but delivers less leakage energy savings in the inactive mode as shown in Table 6[2]. Both techniques have negligible effect on the access time.

5. EXPERIMENTAL RESULTS

We extend a C++ network simulator to investigate the power-performance of power-aware buffers [5]. The same set of router parameters as that in section 3 with an 8-by-8 mesh in $0.07\mu m$ technology is used. Here average latency, throughput and leakage power savings are the metrics used to evaluate our policy. Latency refers to the time from the creation of the first flit of the packet till the ejection of its last flit from the network at the destination, throughput refers to the injection rate at which average network latency exceeds twice the latency at zero network load, and leakage power savings is expressed as a percentage of the total leakage power consumed by router buffers. Simulations are run for 1 million cycles.

Leakage power savings of Lookahead policy. Fig. 5 compares the effectiveness of the conservative Lookahead policy (N = 10 for Gated V_{dd}, and 1 for Drowsy cells) against the ideal policies. Ideal-Double saves close to 100% of buffer leakage power, since it only keeps a buffer active during accesses. Ideal-Single gets savings close to that of Ideal-Double at low traffic workloads as flits do not stay in buffers for long. As traffic increases, however, not shutting buffers off when they are occupied in-between writes and reads result in almost 10% less leakage power savings. A similar difference is observed between Lookahead-Single and Lookahead-Double with Drowsy cells.

With 256 flit-buffers at each router input port, Lookahead-Single saves more leakage power with Gated V_{dd} rather than Drowsy cells. While the long transition delay of Gated V_{dd} results in a large $N = 10$, potentially leading to up to 9 fewer buffers turned inactive, this is overwhelmed by the remaining substantial number of buffers that can still be leveraged. Thus, with large buffers, the higher leakage power savings per SRAM cell of Gated V_{dd} leads to higher overall network power savings as compared to Drowsy SRAMs.

The converse is however true with smaller buffers (Fig. 6). Here, the large N of Lookahead (Gated V_{dd}) constrains the number of buffers that can be turned inactive, and the low transition delay of Drowsy cells win over. Note that as traffic rate increases, however, flits occupy buffers for a longer time, so Lookahead-Single (Drowsy) is unable to exploit its fast transition delay. Lookahead-Double (Drowsy) however leverages this for higher leakage power savings at high traffic injection rates.

Leakage power savings of aggressive and predictive policies. We simulated single Lookahead-Agg policies, with a lookahead window N shortened from 10 to 4 and 2, for Gated V_{dd}. The Predictive policy simulated has W=10, N_{low}=1, N_{high}=2. Fig. 7 shows that as expected, Lookahead-Agg improves the leakage power savings of Lookahead, pushing savings up to 81% at low traffic. Predictive pushes it even further, up to 88% savings at low traffic. Even at very high traffic loads, Predictive still saves 71% leakage power, as it better adapts to actual utilization. This shows that even a simple adaptive policy can outperform oblivious policies.

Performance impact of power-aware buffer policies. Lookahead, being a conservative policy, does not have an impact on performance as it always ensures there will at least be an active buffer available awaiting an arriving flit. However, the aggressive Lookahead-Agg and Predictive policies can potentially cause performance penalties. Fig. 8 simulates the latency-throughput performance of these two policies, showing negligible performance degradation for both policies as compared to a network with no power-aware buffers.

6. CONCLUSIONS

We have proposed a methodology for modeling leakage power on the architecture level. To facilitate the use of this methodology, we will distribute the I'_{leak} tables online. We here also incorporated our network architectural leakage power models into Orion [9] so architects can easily factor in dynamic and leakage power estimates when evaluating

[2]While we assumed the characteristics of Gated V_{dd} SRAMS as published, that it does not preserve data in inactive mode, it can be sized to ensure data preservation though with a poorer transition delay.

Table 6: Characteristics of Drowsy and Gated V_{dd} SRAM cells. The leakage numbers are per line per cycle.

Technique	Normal leakage energy(J)	Inactive leakage energy(J)	Leakage energy savings(%)	Transition energy(J)	Transition delay (cycles)	Data preservation
Drowsy	2.09e-13	3.31e-14	84	3.2e-12	1	Yes
Gated V_{dd}	1.86e-13	9.3e-16	99.5	4.48e-14	10	No

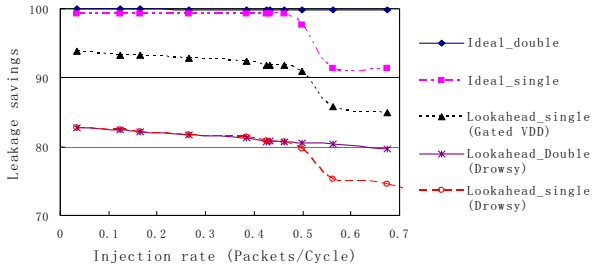

Figure 5: Leakage power savings under Ideal and Lookahead policies for 256-flit buffer.

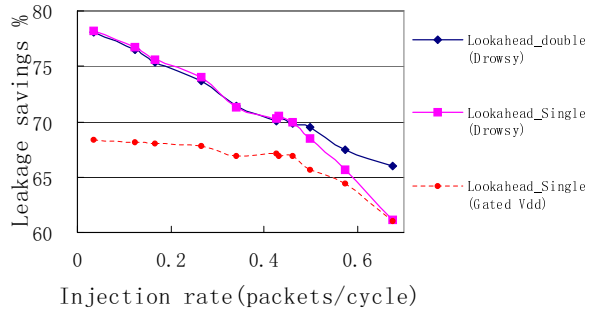

Figure 6: Leakage power savings under Lookahead-Single policy for 64-flit buffer.

network architectures.

By delineating the design space for power-aware buffer policies, and exploring the impact of several simple alternatives, we hope our work will motivate the proposal of sophisticated policies in the future.

Acknowledgments

The authors are grateful to K. Flautner of University of Michigan and S. Kim of Carnegie Mellon for providing detailed parameters of Drowsy Cache and Gated Vdd respectively. At Princeton, we wish to thank Hang-Sheng Wang for his help in characterizing dynamic power using Orion and Li Shang for assistance with the PopNet network simulator. This work is partially funded by NSF CAREER grant CCR-0237540.

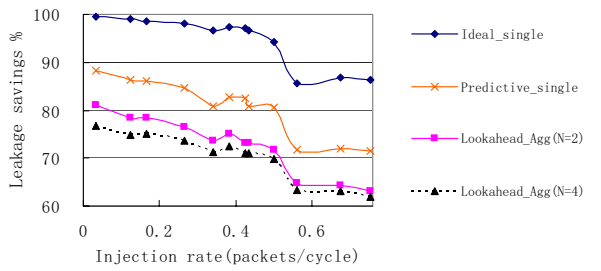

Figure 7: Leakage power savings under Lookahead-Aggressive and Predictive policies for 64-flit buffer.

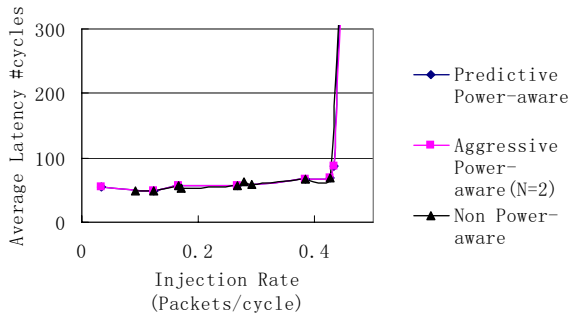

Figure 8: Average latency under different policies for Gated V_{dd} SRAMs.

7. REFERENCES

[1] Berkeley Predictive Technology Model and BSIM4. Available at http://www-device.eecs.berkeley.edu/research.html.

[2] J. A Butts and G. S. Sohi, "A static power model for architects", In *Proc. Intl. Symp. Microarchitecture*, Califonia, Dec. 2000, pp. 191–201.

[3] W. J. Dally and B. Towles. "Route packets, not wires: On-chip interconnection networks", In *Proc. Design Automation Conference*, Las Vegas, June 2001

[4] K. Flautner, N. S. Kim, S. Martin, D. Blaauw, and T. Mudge, "Drowsy caches: simple techniques for reducing leakage power", In *Proc. Intl. Symp. Computer Architecture*, Alaska, May 2002, pp. 219-230.

[5] http://www.ee.princeton.edu/∼lshang/popnet.html

[6] R. Kumar, C. P. Ravikumar, "Leakage power estimation for deep submicron circuits in an ASIC design invironment", In *Proc. ASP-DAC / VLSI Design*, Bangalore, India, 2002. pp. 45-50.

[7] M. Powell, S.-H. Yang, B. Falsafi, K. Roy, and T. N. Vijaykumar, "Gated-V_{dd}: a circuit technique to reduce leakage in deep-submicron cache memories", In *Proc. Intl. Symp. Low Power Electronics and Design*, Italy, July, 2000, pp. 90-95.

[8] L. Shang, L.-S. Peh, and N. K. Jha, "Dynamic voltage scaling with links for power optimization of interconnection networks", In *Proc. Intl. Symp. on High-Performance Computer Architecture*, California, Jan. 2003, pp. 79-90.

[9] H. Wang, X. Zhu, L.-S. Peh, and S. Malik, "Orion: a power-performance simulator for interconnection networks", In *Proc. Intl. Symp. Microarchitecture*, Istanbul, Turkey, Nov. 2002, pp. 294-305. [http://www.ee.princeton.edu/∼peh/orion.html]

[10] F. Worm, P. Ienne, P. Thiran and G. D. Micheli, "An adaptive low power transmission scheme for on-chip networks", In *Proc. Intl. Symp. Systems Synthesis*, Kyoto, Japan, October 2002, pp. 92-100.