Power Considerations in Network Processor Design

Mark A. Franklin[1*] and Tilman Wolf[2]

[1] Department of Computer Science and Engineering
Washington University in St. Louis
jbf@ccrc.wustl.edu

[2] Department of Electrical and Computer Engineering
University of Massachusetts at Amherst
wolf@ecs.umass.edu

Abstract

Network processors are commonly implemented as systems-on-a-chip with multiple processors, caches, memory interfaces and I/O components on a single chip. Networking workloads lend themselves to exploiting high levels of parallelism with these chip-multiprocessors. The constraints of such a system lie in the maximum chip area and the maximum power consumption that are permissible for economic and technical reasons. We develop an analytic performance model that captures the processing performance and power consumption of such a system. Using a variety of metrics, we explore the design space of network processors and show the performance impact of different system and memory configurations.

1 Introduction

Over the last several years, network processors (NPs) have become important components in router designs. By providing for programmability, they permit adaptation to new functional requirements and standards. Additionally, network processors provide a powerful single (or a few) chip multiprocessor architecture, typically containing logic components and instructions specialized to the networking environment, to satisfy a range of performance requirements. At this point there are over two dozen companies producing a variety of network processors [9] [10] [13] [5].

*This research has been supported in part by National Science Foundation grant CCR-0217334.

At the hardware level, there are four key concerns in the design of NPs.

- Computational Power: The NP must be able to perform the required computational tasks fast enough to keep up with input line speeds.
- Functional Power: The NP must be able to perform the required functional tasks associated with its targeted environment (e.g., packets, cells, IPv4, IPv6, MPLS, etc.).
- Cost: The cost of the chip should be reasonable. In this paper we deal with only manufacturing costs and consider chip area to be a proxy for these costs.
- Electrical Power Dissipation: The NP must not consume an excessive amount of power.

In this work we consider the prototypical NP architecture shown in Figure 1. It contains a number of identical multithreaded general-purpose processors, each having its own instruction and data caches. To satisfy off-chip memory bandwidth requirements, groups of processors are clustered together and share a memory interface. A scheduler assigns packets from independent flows to the different processors. Thus, after assignment of a flow to a processor, all packets of the same flow are routed to the same processor. Speedup and computational power is achieved by exploiting parallelism at the flow level. Note that additional speedup can be obtained by also exploiting packet level parallelism, however, this is not considered here. All of the processors are assumed to be identical and capable of executing the necessary NP functions.

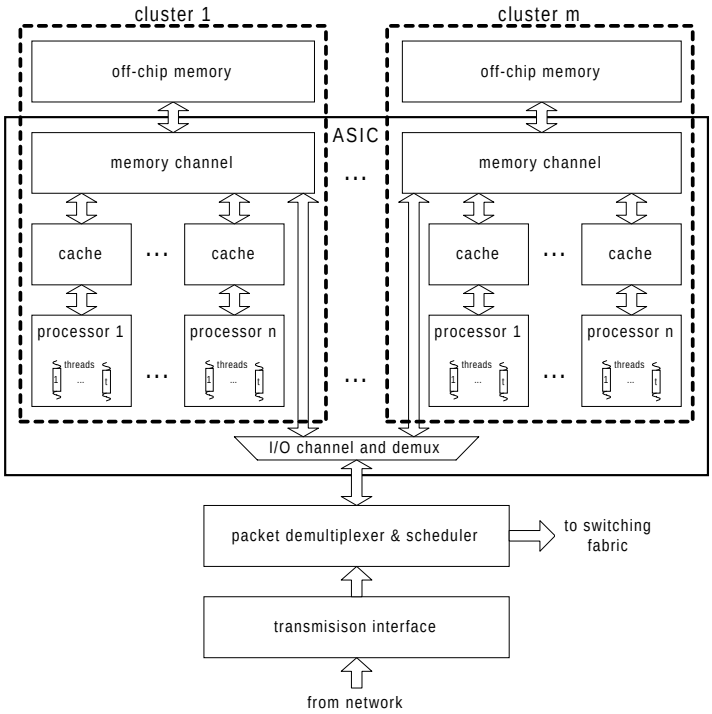


Figure 1. Overall Network Processor Architecture.

In our previous work [6] [16] [14] we have developed performance models in order to find the optimal configuration of components associated with this architecture. The performance metric utilized in this prior work involved both computational power and chip area. Computational power is measured by the total IPS (Instructions per Second) available from the NP, and area is measured by the number of square centimeters required for a given chip configuration. A configuration consists of a selection of the instruction and data cache sizes, the number of processors, the number of clusters, and the multithreading level associated with each processor[1]. Other design options such as channel bandwidth and use of on-chip DRAM were also considered.

In this paper we extend the model presented earlier to include the important component of power dissipation. As line rates and clock frequencies have increased, power dissipation considerations often effect design decisions. In a router environment, there may be one or two NPs per line card with the card holding various other components (e.g., optical-electrical converters, line drivers, memories, CAMs, various interfacing chips, etc.). A group of line cards (e.g., 16, 32) are generally placed within a single rack or cabinet, and in such an arrangement aggregate heat dissipation issues become important. Thus, although many current commercial NPs consume ten or more watts, designing for increased performance while restraining power dissipation is a constant concern.

This paper presents the development of optimal designs that provide for the maximum IPS while at the same time minimizing metrics involving power consumption, chip area, or a combination of the two. The components involved in the process are shown in Figure 2. Using a benchmark of networking oriented programs called CommBench [15], we obtain an application workload that is representative of the network processor environment. This workload is simulated with the SimpleScalar [2], Wattch [1], and Cacti [12] tool sets to derive workload and power parameters, which are necessary for the analytic models. The overall analytic models consists of a model for processing power and chip area and a model for power consumption. Individual analytic power dissipation models for the main architecture components (e.g., ALUs, clocks, caches, etc.) are developed in this work. The results of various performance metrics from the models are used to find the optimal configurations for the system of Figure 1 by iterating over the design space. The simulation environment is also used to verify the accuracy of the analytic models derived in our work.

[1]While most commercial NPs employ multithreading, for simplicity, here we consider single threaded processors. The processor model can be readily extended to the multithreaded case [6].

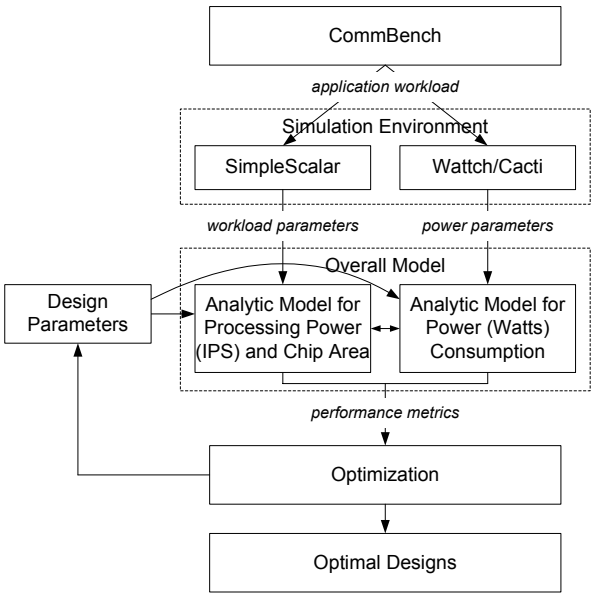


Figure 2. Model Development and Optimization Process.

Section 2 presents the model used in determining the processing power for the NP. The workload for our analysis is based on the CommBench benchmark and is briefly discussed in Section 2.1. Section 3 develops the power model and explains the usage of various simulation tools to obtain model parameters. Section 4 describes the area model utilized and the set of performance metrics to be considered. Section 5 presents the results of overall model optimization and examines how performance changes as selected parameters are varied. The final section contains a summary of the results and a number of design conclusions that follow from the analysis.

2 Computational Performance Model

For a single processor, processing power can be expressed as the product of the processor's utilization, ρ_p, and its clock rate, clk_p. The processing power of the entire NP can be expressed as the sum of processing power of all the processors on the chip. If all processors are identical and run the same workload, then in a NP with m clusters and n processors per cluster, on average the processing power is:

$$IPS = m \cdot n \cdot \rho_p \cdot clk_p. \tag{1}$$

A key question is how to determine the utilization of the processors. For ideal RISC processors where significant hazards result principally from cache misses, processor utilization can be expressed as:

$$\rho_p = \frac{1}{1 + \tau_{mem} \cdot p_{miss}}. \tag{2}$$

where τ_{mem} is the memory access time and p_{miss} is the cache miss rate.

We assume the memory channel implements a FIFO service order on the memory requests in such a way that they can be interleaved in a split-transaction fashion. The total off-chip memory request time, τ_{mem}, thus has three components: the bus access time, τ_Q, the physical memory access time, τ_{DRAM}, and the cache line transmission time, $\tau_{transmit}$ (all represented in terms of numbers of processor clock cycles):

$$\tau_{mem} = \tau_Q + \tau_{DRAM} + \tau_{transmit}. \tag{3}$$

The DRAM access time, τ_{DRAM}, is determined by the external DRAM specifications. The cache line transmission time, $\tau_{transmit}$, depends on the cache line size, $linesize$, the memory channel width, $width_{mchl}$, the processor clock frequency, clk_p, and the memory channel clock frequency, clk_{mchl}. The queuing time, however, depends on the load on the memory channel. We have shown earlier [6] that the M/D/1 queuing model is a reasonable approximation of the memory channel queuing time τ_Q. Thus, for a channel utilization of ρ_{mchl} and an average service time of $E(s)$, the bus access time, τ_Q, is given by:

$$\tau_Q = \frac{\rho_{mchl}^2 \cdot E(s)}{2(1-\rho_{mchl})} = \frac{\rho_{mchl}^2}{2(1-\rho_{mchl})} \cdot \frac{linesize}{width_{mchl}} \cdot \frac{clk_p}{clk_{mchl}}. \quad (4)$$

With a fixed DRAM access time, τ_{DRAM}, and a transmission time of

$$\tau_{transmit} = \frac{linesize}{width_{mchl}} \cdot \frac{clk_p}{clk_{mchl}}, \quad (5)$$

we can substitute in Equation 3 to obtain the memory access time:

$$\tau_{mem} = \tau_{DRAM} + \left(1 + \frac{\rho_{mchl}^2}{2(1-\rho_{mchl})}\right) \cdot \frac{linesize}{width_{mchl}} \cdot \frac{clk_p}{clk_{mchl}}. \quad (6)$$

The remaining component needed to evaluate the utilization expression (Equation 1) is the cache miss rate, p_{miss}. For a simple RISC style load-store processor running application a, the miss probability is given as [8]:

$$p_{miss,a} = mi_{ci,a} + (f_{load_a} + f_{store_a}) \cdot md_{cd,a} \cdot (1 + dirty_{cd,a}), \quad (7)$$

where $mi_{ci,a}$ and $md_{cd,a}$ are the instruction and data cache miss rates for application a with respective cache sizes ci and cd. The parameters f_{load_a} and f_{store_a} are the frequency of occurrence of load and store instructions also for application a. The parameter $dirty_{cd,a}$ is the probability of the dirty bit being set on a cache line requiring that the cache line be written back to memory. Section 2.1 discusses the applications from which these parameter values are derived.

The expression for miss rate, p_{miss}, (Equation 7) and for total memory access time, τ_{mem}, (Equation 3) can now be substituted into Equation 2 to obtain processor utilization. To do this, the memory channel load, ρ_{mchl}, needs to be fixed because τ_Q depends on ρ_{mchl}. Thus, with the memory channel load given, we can determine the utilization of a single processor. This gives us the memory bandwidth, $bw_{mchl,1}$, required by a single processor:

$$bw_{mchl,1} = \rho_p \cdot clk_p \cdot linesize \cdot p_{miss}. \quad (8)$$

With $width_{mchl} \cdot clk_{mchl} \cdot \rho_{mchl}$ being the bandwidth associated with the selected memory channel utilization, the number of processors, n, in a cluster corresponds to the number of processors that can share the memory channel without exceeding the specified load. Thus n is given by:

$$n = \left\lfloor \frac{width_{mchl} \cdot clk_{mchl} \cdot \rho_{mchl}}{bw_{mchl,1}} \right\rfloor. \quad (9)$$

Having considered the memory channel, we now turn our attention to the I/O channel that is used to input and output packets from the network. From monitoring execution of an application a (or a benchmark of applications), one can obtain a parameter, $compl_a$, referred to as "*complexity*". The application complexity corresponds to the number of instructions that are required to process a packet of a certain length. That is:

$$compl_a = \frac{instr.\ executed\ in\ the\ application}{packet\ size} \quad (10)$$

For an I/O channel operating at a load of ρ_{IO}, the I/O channel bandwidth, bw_{IO}, for the entire NP is:

$$bw_{IO} = 2 \cdot \frac{IPS}{compl_a \cdot \rho_{IO}}. \quad (11)$$

The number of pins on the NP package can also be determined by summing the pins required for I/O and memory channels with the pins required for control and power. The number of memory channel pins are obtained directly from $width_{mchl}$, while the number of I/O memory pins can be obtained from a knowledge of bw_{IO} and the I/O channel clock frequency.

2.1 The Benchmark

To properly evaluate and design NPs it is necessary to specify a workload that is typical of that environment. This has been done in the development of the benchmark CommBench [15]. CommBench applications represent typical workloads for both traditional routers (focus on header processing) and programmable routers (perform both header and stream processing). Thus, the applications can be divided into two groups: *Header-Processing Applications* (HPA) and *Payload-Processing Applications* (PPA). For our model, we use two workloads, W1 and W2, which are aggregates of the applications in CommBench. A list of the applications is given in Table 1. Workload W1 is a combination of the four header-processing applications. Workload W2 consists of the four payload processing applications. The applications within the workloads are weighted such

that each application processes the same number of instructions over time. W1 applications process only packet headers and are generally less computationally demanding than W2 applications that process all of the data in a packet.

A desirable property of any application in a benchmark is its representativeness of a wider class of applications in the domain of interest. Therefore, a key focus is on the "kernels" of the applications, which are the program fragments containing the set of dynamically frequently used instructions. The application kernels associated with W1 and W2 applications are shown in Table 1.

For each application, the properties required for the performance model have been measured experimentally: computational complexity ($compl_a$), load and store instruction frequencies (f_{load_a}, f_{store_a}), instruction cache and data cache miss rate ($mi_{ci,a}, md_{cd,a}$), and dirty bit probability ($dirty_{cd,a}$). These parameter values were obtained with a processor and cache simulator (Shade [3] and Dinero [4] and verified with SimpleScalar [2]) for cache sizes ranging from 1kB to 64kB. A 2-way associative write-back cache with a linesize of 32 bytes was simulated. The cache miss rates were measured such that cold cache misses were amortized over a long program run. Thus, they can be assumed to represent the steady-state miss rates of these applications. The average values for the parameters were obtained for each of the benchmarks (W1 and W2) by averaging over the benchmark application values assuming equal probabilities for each application. The parameter values and miss probability curves can be found in [15].

3 Power Model

3.1 Overall Power Model

The IPS metric is one of three that must be obtained in determining the "best" NP architecture configuration. The second critical metric relates to the power consumption (watts) associated with the design. The third is the NP chip area which is considered in the next section.

The principal components considered in the power calculations are:

- processor ALUs
- processor clock
- processor instruction and data caches (level 1, on-chip)
- off-chip memory and I/O bus

Since we are interested in relative performance of alternative configurations for the architecture of Figure 1, power associated with off-chip components and with driving the chip pins are not considered. Additionally, the contribution of the branch predictor is ignored since, for simple NP RISC cores, it is not necessary to perform complex branch prediction and, overall, system power is dominated by memory accesses and I/O operations. Complex superscalar processors, where a mispredicted branch may have a significant performance impact, are not considered.

For all our simulations, we model the overall network processor power consumption, P_{NP}, as a sum of the four components listed above (scaled to the appropriate number of processors and sizes of caches). This makes up for 94%-97% of the overall power consumptions (ignoring the branch predictor). The remaining 3%-7% are consumed by register files and miscellaneous control components.

For CMOS technology, dynamic power consumption P_d is defined as:

$$P_d = C \cdot V_{dd}^2 \cdot a \cdot f. \tag{12}$$

where C is the aggregate load capacitance associated with each component, V_{dd} is the supply voltage, a is the switching activity for each clock tick ($0 \leq a \leq 1$ and can be considered to be the utilization of the component) and f is the clock frequency. The energy expended per cycle is[2]:

$$E_d = C \cdot V_{dd}^2 \cdot a. \tag{13}$$

By obtaining parameter values for Equations 12 and 13, the power consumption models for each of the components is determined. The sum of these models yields an overall power consumption model for the NP. Most of the parameters are based usage of the Wattch toolkit [1] and CACTI [11] [12]. These values correspond to the use of an Alpha 21264 [7] processor and a .35μm technology. Since we are primarily interested in comparative NP configurations and what they illustrate about NP design, smaller feature size technologies are not initially considered. However, the analytic models presented apply with adjustments of the parameter values for other technologies (e.g., .18μm and $V_{dd} = 2.0$ volts).

To verify the analytic power model, power results are compared to the power results obtained from executing Wattch over the benchmark discussed. This is considered in Section 3.6. Once the model has been verified, optimal NP configurations are then obtained

[2]The power modelling does not account for leakage currents and associated power which will become more important as feature sizes shrink below .15μm.

Workload	Name	Type	Application	Kernel
W1	RTR	HPA	Radix tree routing	Lookup on tree data structure
	FRAG	HPA	IP header fragmentation	Packet header checksum computation
	DRR	HPA	Deficit round robin	Queue maintenance
	TCP	HPA	TCP filtering	Pattern matching on header fields
W2	CAST	PPA	Encryption	Encryption arithmetic
	ZIP	PPA	Data compression	Compression arithmetic
	REED	PPA	Reed-Solomon FEC	Redundancy coding
	JPEG	PPA	JPEG Compression	DCT and Huffman coding

Table 1. Benchmark Applications.

analytically without resorting to the use of Wattch simulations.

3.2 ALU Power Model

ALU power depends on the voltage, V_{dd}, processor clock frequency, f, the ALU utilization, a_{ALU}, and its capacitance:

$$P_{ALU} = C_{ALU} \cdot V_{dd}^2 \cdot a_{ALU} \cdot f. \quad (14)$$

Using Wattch, the capacitance for .35 μm technology (the process specification of an Alpha 21264 [7] that is simulated by Wattch) can be obtained as 310*p*F. V_{dd} for this case is 2.5 volts.

The value for a_{ALU} (that corresponds to the ALU utilization, ρ_{ALU}) used by Wattch is 1. As discussed later, this value is used to verify the analytic power model by comparing model results with the results obtained from Wattch. However, by using a value of 1, the Wattch simulator assumes that the ALU is busy on every cycle. This is not true during stalls due to cache misses. Thus, the value used in our optimization studies (as contrasted with the power model verification work) is obtained from Equation 2 ($a_{ALU} = \rho_p$) and reflects the effects of cache misses on component utilization.

3.3 Clock

In a similar fashion clock power consumption can be obtained:

$$P_{clk} = C_{clk} \cdot V_{dd}^2 \cdot a_{clk} \cdot f. \quad (15)$$

Since the clock is changing state in every cycle, $a_{clk} = 1$. From Wattch, we obtain $C_{clk} = 3.33nF$. With differing cache configurations, the clock power consumption can vary by up to $\pm 8\%$, however the model does not consider this effect. As will be shown in Section 3.6, overall power consumption that is predicted corresponds well to that obtained with Wattch.

3.4 Caches

The expression for cache power consumption is:

$$P_{c_i} = C_{c_i} \cdot V_{dd}^2 \cdot a_{c_i} \cdot f. \quad (16)$$

The dynamic power consumption of caches is due to memory accesses. For the instruction cache, the i-cache is accessed for each instruction. Additionally, the i-cache is accessed after each pipeline stall due to i-cache misses or branch misprediction (we do not consider misprediction effects on cache power in this analysis). Adding in the effects of cache usage occurring after a miss, one obtains:

$$a_{c_i} = \rho_p \cdot (1 + mi_{ci,a}). \quad (17)$$

where $mi_{ci,a}$ is the instruction cache miss probability associated with application a and instruction cache size ci.

The data cache is accessed for each read/write (load/store) instruction and for each d-cache miss, thus:

$$a_{c_d} = \rho_p \cdot ((f_{load_a} + f_{store_a}) \cdot (1 + md_{cd,a})). \quad (18)$$

The cache capacitance, C_{c_i} and C_{c_d}, is shown in Table 2. These numbers are given by the CACTI tool [12] for .35μm technology.

3.5 Memory and I/O Bus

The same approach taken in Wattch is used to calculate the power consumption of the memory and I/O busses. The memory channel is characterized by its width, $width_{mchl}$, its physical length on the chip, $length_{mchl}$, its clock frequency, f_{mchl}, and its utilization $a_{mchl} = \rho_{mchl}$. As part of the optimization procedure the channel utilization, as used in performance model equations 4 to 9, is varied to find its value associated with the optimal configuration.

The capacitance, C_{mchl}, is based on the width and the length parameters and is given by:

Cache size in kB	i-cache capacitance in nF	d-cache capacitance in nF
1	0.369	0.378
2	0.397	0.406
4	0.440	0.450
8	0.541	0.570
16	0.708	0.739
32	0.957	1.030
64	1.368	1.412

Table 2. Cache Capacitance for .35 μm Technology. The cache line size is 32 bytes and associativity level is 2. For instruction caches one read/write port and one read port are assumed. For data caches two read/write ports are assumed.

$$C_{mchl} = 2 \cdot C_{.35\mu m} \cdot width_{mchl} \cdot length_{mchl}. \quad (19)$$

The factor of 2 is due to the coupling capacitance between wires. The length of the memory channel is taken to be $length_{mchl} = 5$mm, which is the expected distance to a processor from the edge of a chip. We also explored a larger channel length of 20mm. This, however, only affects the overall results shown below by about 1%. The width is set to 32 bits. The capacitance parameter associated with using .35μm technology is obtained from scaling the capacitance associated with Wattch's "result bus," yielding $C_{.35\mu m} = 0.275$fF/μm.

3.6 Validation

To compare the validity of the above power model, the energy results obtained with Wattch are compared with the model results. In the validation experiment, all applications in the benchmark were executed for cache configurations ranging from 1kB to 64kB. Figure 3 shows the Wattch results versus the model results. Ideally, each cross point would lie on the dashed line which corresponds to the model and Wattch having the same results. It should be noted that Wattch simulates a complex superscalar processor. To make a reasonable comparison to the RISC core that we are modelling, only the ALU, clock and cache access power from Wattch was considered. Since there is no shared memory bus modelled in Wattch, we cannot compare the results for this component.

The maximum error is 15.8% for the smallest cache size. This is due to differences in the results from the Cacti toolkit versus the Wattch results. For larger caches the differences are much smaller. With an average error of only 8%, the analytic approximation of power consumption is a useful tool for NP design space exploration.

4 Performance Metrics

We use several performance metrics to evaluate design choices. Processing performance comes at the cost of power consumption and chip area. To be able to capture the chip area cost, A, we use the following expression (see [6]):

$$A = s(io)+m\cdot(s(mchl)+n\cdot(s(p)+s(ci)+s(cd))), \quad (20)$$

where s is the area of a processor ($s(p)$), the caches ($s(ci)$ and $s(cd)$), the memory channel ($s(mchl)$) and the I/O channel ($s(io)$). For .35μm CMOS technology, we assume $s(p) = 4$mm^2, $s(c) = 0.5$mm^2/kB, $s(mchl) = 28$mm^2 (20mm^2 for the channel and 8mm^2 for memory channel logic).

With an expression for processing performance (IPS), power consumption (P), and chip area (A), performance metrics of the following form can be derived:

$$Performance = IPS^{\alpha} \cdot A^{\beta} \cdot P^{\gamma}. \quad (21)$$

In particular, we are interested in the metrics that consider area and power consumption as a cost ($\beta, \gamma \leq 0$) and $\alpha < 0$. For the design results in Section 5, the following common processor performance metrics are used:

- Processing/power or $IPS \cdot P^{-1}$: this metric assumes an equal weight to processing performance and power consumption.
- Processing/(power)^2 or $IPS \cdot P^{-2}$: in this case, power consumption is weighted higher.
- Processing/area or $IPS \cdot A^{-1}$: this metric considers only area and no power consumption (as used in [6]).
- Processing/area/power or $IPS \cdot A^{-1} \cdot P^{-1}$: this combines both area and power costs.

5 Design Results

In this section design results based on the "optimal" design for a given metric are considered. To obtain this "optimal" design, the entire design space

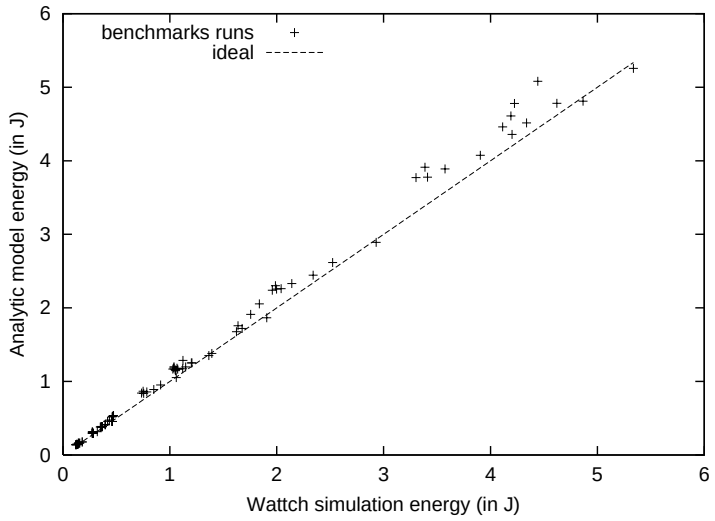


Figure 3. Comparison of Benchmark Application Execution Energy for Wattch Toolkit and Analytic Model.

is examined and the best configuration of cache sizes ($ci, cd = 1\text{kB}\ldots 64\text{kB}$), number of processors ($n \geq 1$, limited by maximum memory channel load) and memory channel utilization ($rho_{mchl} = 0 \ldots 0.999$) is obtained. The processor clock frequency is 600MHz and the memory channel clock frequency is 240MHz.

5.1 Performance Trends

Figure 4 illustrates the basic trends for the components of the performance metrics from Equation 21. To illustrate basic trends, the cache sizes in this figure are set to 8kB for both the instruction and data caches. The number of processors, n, that share a memory channel (i.e., processors in a cluster) is shown on the x-axis. The y-axis shows the increase in processing performance, power consumption, and area relative to a configuration with a single processor ($n = 1$).

As expected, the area curve, A, increases linearly with the number of processors (the slope depends on the proportion of processor and cache sizes to the memory channel). The instructions per second curve, IPS, initially increases more rapidly then A, but then at about 6 processors levels out. This is due to the fact that with increasing numbers of processors, the shared memory channel load, ρ_{mchl}, increases due to processor contention for use of the channel. However, at saturation, the memory responds to requests at its maximum rate and hence the IPS remains steady.

The trends on Figure 4 show that power consumption grows fastest. The faster growth of power is due to memory channel contention. If more processors share a memory channel, the processor stall time on a cache miss increases. During a stall, the processor does no useful computation, but still consumes energy. As a result the total processing performance does not increase very much, but power consumption does. These trends are very similar for all cache configurations. The plateaus for processing performance are higher for larger caches since miss rates are lower and thereby contention on the memory channel is less. In all cases, however, power consumption grows faster than processing performance.

The effect on the performance metrics is shown in Figure 5, where each metric is shown versus a range of processors for both workloads. Figure 5(a) shows the trends for both power-related metrics (IPS/P and IPS/P^2). Because power increases faster than processing performance, the performance drops with higher number of processors. This means that from the point of view of power consumption, fewer processors per memory channel are preferable. Looking at the impact of area in Figure 5(b), however, fewer processors are not necessarily best. There is a clear optimum for three (workload W1) or six (workload W2) processors. The differences between the workloads are due to different cache miss rates. When combining both area and power, again, power consumption dominates the cost and causes a clear drop in performance for more processors.

The implications for network processor design are the following:

- More processors per memory interface increase the relative power consumption for the network

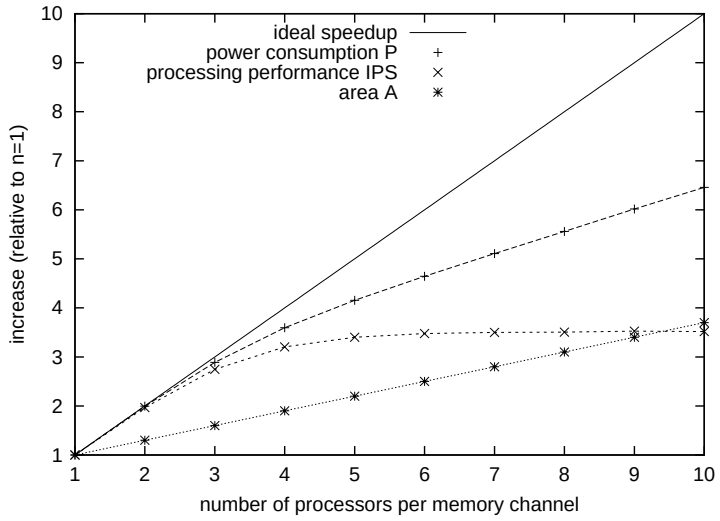
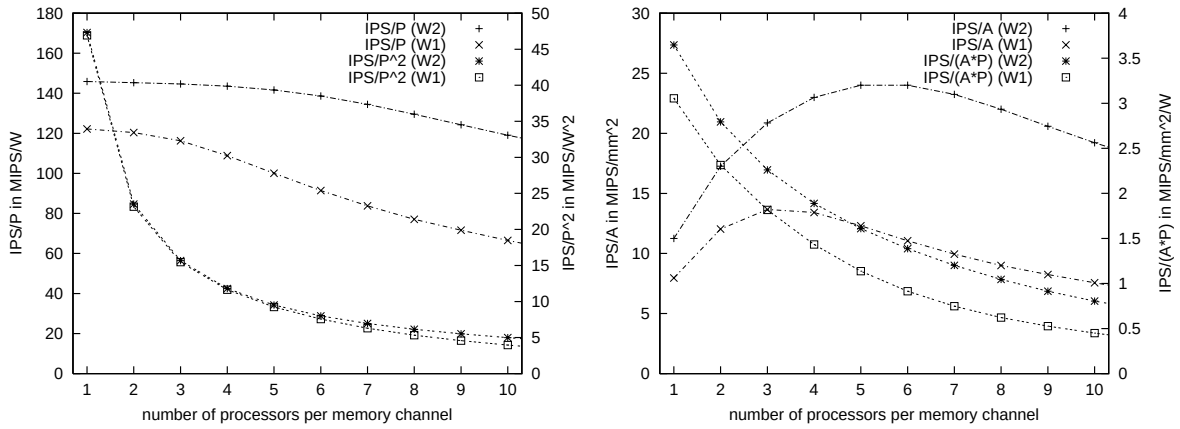


Figure 4. Trends of Processing Performance, Area, and Power. The workload is W1 and cache sizes are c_i=8kB and c_d=8kB.



(a) IPS/P and IPS/P^2

(b) IPS/A and $IPS/(A \cdot P)$

Figure 5. Performance Trends for Various Performance Metrics and Workloads. The cache sizes are set to c_i=8kB and c_d=8kB.

processors. This comes from power dissipation of the clock during stall cycles while waiting for memory access and suggests that fewer processors (ideally one) per memory interface is best in terms of power consumption. However, that is not realistic from the point of view of the number of external memory chips that would be required. Thus, there is the tradeoff between power dissipation, which requires few processors with high utilizations and many interfaces, and the costs and engineering constraints (e.g., pin limitations) associated with having many memory interfaces.

- When considering area constraints in the design, having only one processor per interface is not optimal. Instead, the optimum is reached when a few processors are clustered to share a memory interface. The optimal configuration depends on the workload and technology parameters.

- Other measures can be taken to avoid energy consumption during memory stalls: Multithreading allows to a processor to switch to a different task when encountering stalls. Clock

gating can be used to reduce the power consumption of components that are not in use.

One main observation from our design results is that a significant amount of power is lost through processor stalls. For the optimal configurations shown in Table 3, the processor utilization, ρ_p ranges from 30% to 50%. This emphasizes the importance of multithreading support in network processors. With additional hardware threads and zero-overhead context switching, the processing power can be increased significantly. In our previous work [6], we have shown that the processing power improvement for two threads easily makes up for the additional area cost associated with multiple thread register files.

5.2 Optimal Cache Configuration

One key question for system-on-a-chip design is how to find a good balance between processing logic and on-chip memory. Network processors designs are constrained by the maximum chip size. More processing engines mean more processing cycles, but also smaller caches, higher cache miss rates, more memory contention and higher energy consumption. Using our model, we can find the optimal cache configuration for a given metric. The design space is relatively small and a exhaustive enumeration of the design options can be used to obtain the optimum design. Figures 6(a)–6(d) show the performance of various cache configurations for the different performance metrics.

The following observations can be made regarding the optimal cache size:

- For IPS/P (Figure 6(a)), the optimum lies at $ci = 8$kB and $cd = 32$kB. Since processing power increases with larger caches (due to fewer memory stalls), the optimum configuration uses a large data cache.
- For IPS/P^2 (Figure 6(b)), the optimum lies at $ci = 4$kB and $cd = 4$kB. Even though the optimization metric is based on power (as is IPS/P), the optimum configuration yields small caches, which is quite different from the optimum for IPS/P. Because of the quadratic cost for power consumption, larger caches cost more than they can contribute in terms of processing power.
- For IPS/A (Figure 6(c)), the optimum lies at $ci = 16$kB and $cd = 8$kB. For this metric, small caches cause inefficient processing and large caches cost too much in terms of area. Thus, there is a clear optimum for a medium configuration.
- For $IPS/(A \cdot P)$ (Figure 6(d)), the optimum lies at $ci = 4$kB and $cd = 4$kB. Here the optimum configuration again uses small caches, because both area and power contribute to the cost. The larger caches contribute to a better IPS performance but at the same time cost in terms of area and power.

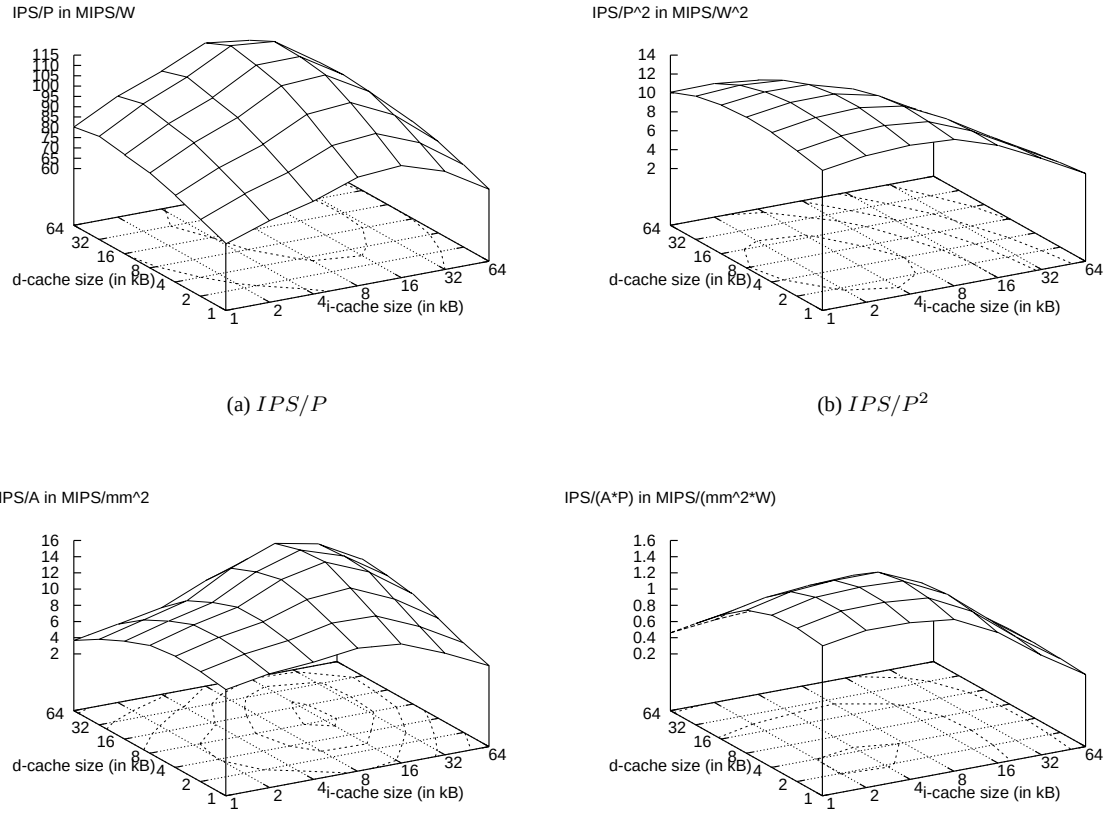
From these observations, we can conclude that for both IPS/P and IPS/A, there are clear optima for which the network processor can be configured. Using any combination metric involving power as a cost function (e.g., P^2 or $A \cdot P$) yields very small cache configurations since the IPS improvements cannot keep up with the cost for larger caches. If a metric for both area and power is desired, it might be more suitable to use $IPS/\sqrt{A \cdot P}$ as it keeps a balance between performance and total cost.

The values for the optimal instruction and data cache sizes as a function of the number of processors per cluster is shown in Figures 7(a) and 7(b). There is a slight trend towards larger caches for more processors as again more processors cause more load on the memory channel.

5.3 Chip Configurations

Table 3 shows overall chip configurations for a 400mm^2 chip. The table shows the optimal configurations in terms of number of memory interfaces, m, and processors per memory channel, n. For all metrics and workloads, the overall throughput of such a system is also shown, which is determined by the complexity of the workload and the overall processing power ($IPS/compl$). Note that the complexity for workload W2 is about 50 times higher than that of W1, which results in the large differences in throughput. Thus, while header processing applications can achieve throughput rates in the gigabit range, payload processing applications have rates well under a gigabit for all performance metrics. This is consistent with the notion that these types of applications (e.g., encryption) often require special purpose processors and logic to achieve high throughput rates.

For power-related metrics, the trends in Figure 5 result in optimal configurations with only one processor per interface. This however yields a lower throughput than when optimizing for area only. On the other hand, power consumption for the area-optimized configuration is about twice as high as that for power-optimized configurations. The $IPS/\sqrt{A \cdot P}$ metric is a good combination of area



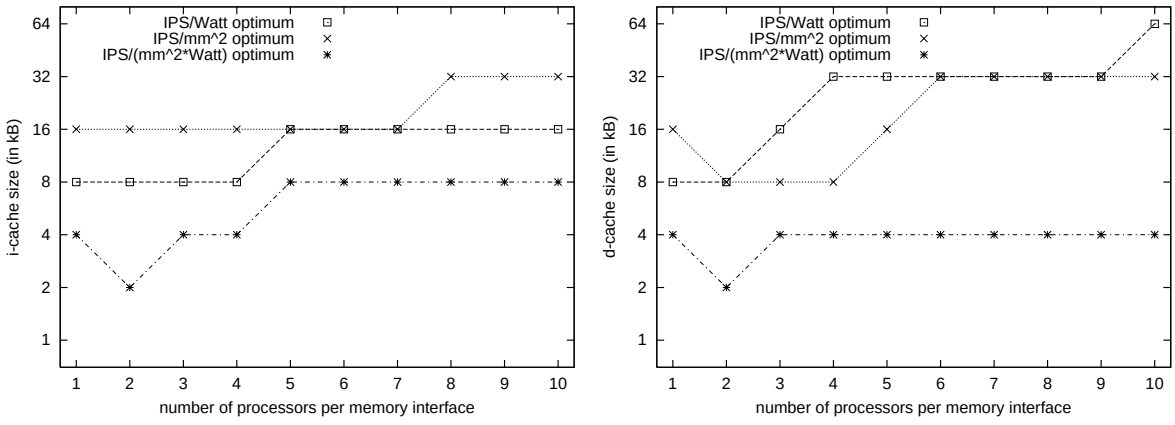
(a) IPS/P

(b) IPS/P^2

(c) IPS/A

(d) $IPS/(A \cdot P)$

Figure 6. Performance of Cache Configurations for Various Performance Metrics. The workload is W1 and the number of processors per memory channel is set to four.



(a) Optimal Instruction Cache Size

(b) Optimal Data Cache Size

Figure 7. Optimal Cache Configuration for Different Number of Processors per Memory Channel. The workload is W1.

Metric	workload	chip configuration					total IPS (in MIPS)	total P (in W)	throughput (in Gbps)
		m	n	$m \cdot n$	ci	cd			
IPS/P	W1	10	1	10	8	8	3181	26.0	6.75
	W2	10	1	10	8	8	4499	30.8	0.18
IPS/P^2	W1	10	1	10	1	4	2095	19.7	4.45
	W2	11	1	11	1	4	2293	19.9	0.09
IPS/A	W1	4	4	16	16	8	5360	48.7	11.37
	W2	4	6	24	8	8	9603	69.2	0.38
$IPS/(A \cdot P)$	W1	11	1	11	4	4	2652	22.6	5.63
	W2	10	1	10	8	8	4499	30.8	0.18
$IPS/\sqrt{A \cdot P}$	W1	5	3	15	16	8	5406	47.7	11.47
	W2	4	5	20	8	8	8448	59.6	0.33

Table 3. Chip Configurations and Throughput Results. This table shows the optimal configurations for various optimization metrics for a 400mm^2 chip.

and power. It yields configurations with four to five memory interfaces and good throughput (e.g., for workload W1, $IPS/\sqrt{A \cdot P}$ yields higher throughput with less power consumption than IPS/A).

The overall power consumption of the optimal configurations with 20W to 70W is higher than current commercial systems, which consume on the order of 10W. This is due to commercial NPs using more advanced CMOS technologies with smaller feature and overall chip sizes (e.g., Intel IXP2400: $.18\mu$m vs. $.35\mu$m and 1.3V vs. $2.5V$ [10]).

6 Summary and Conclusions

This work develops an analytic model for power consumption of network processor systems-on-a-chip. Combining this with the performance model that we have developed in previous work, we show how both models can be used to yield an understanding of power issues for these systems. Our power model was verified through comparison with the Wattch toolkit with an average error of only 8%. Using a workload that is derived from our CommBench benchmark for model parametrization, we obtain quantitative results for different performance metrics. This enabled the determination of optimal network processor configurations in terms of cache configurations and number of processors per memory interface. We believe this is an important step towards developing network processor architectures that yield high processing power, but are also within the power constraints of realistic systems.

Currently, we are refining the models and methodology presented. In particular we are expanding the analysis to reflect multithreading. Additionally, we are investigating the use of more accurate power, associated capacitance models and incorporation of limitations on the number of external memory chips that can be used.

References

[1] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: A framework for architectural-level power analysis and optimizations. In *Proc. of ACM ISCA-27*, pages 83–94, Vancouver, BC, June 2000.

[2] D. Burger and T. M. Austin. The SimpleScalar tool set, version 2.0. Technical Report 1342, Department of Computer Science, University of Wisconsin in Madison, June 1997.

[3] R. F. Cmelik and D. Keppel. Shade: A fast instruction-set simulator for execution profiling. In *Proc. of ACM SIGMETRICS*, pages 128–137, Nashville, TN, May 1994.

[4] J. Edler and M. D. Hill. *Dinero IV Trace-Driven Uniprocessor Cache Simulator*, 1998. http://www.cs.wisc.edu/~markhill/DineroIV/.

[5] EZchip Technologies Ltd., Yokneam, Israel. *NP-1 10-Gigabit 7-Layer Network Processor*, 2002. http://www.ezchip.com/html/pr_np-1.html.

[6] M. A. Franklin and T. Wolf. A network processor performance and design model with benchmark parameterization. In *Network Processor Workshop in conjunction with Eighth International Symposium on High Performance Computer Architecture (HPCA-8)*, Cambridge, MA, Feb. 2002.

[7] M. K. Gowan, L. L. Biro, and D. B. Jackson. Power considerations in the design of the Alpha 21264 microprocessor. In *Proc. of 35th Design Automation Conference*, pages 726–731, San Francisco, CA, June 1998.

[8] J. L. Hennessy and D. A. Patterson. *Computer Architecture – A Quantitative Approach*. Morgan Kaufmann Publishers, Inc., San Mateo, CA, second edition, 1995.

[9] IBM Corp. *IBM Power Network Processors*, 2000. http://www.chips.ibm.com/products/wired/communications/network_processors.html.

[10] Intel Corp. *Intel IXP2800 Network Processor*, 2002. http://developer.intel.com/design/network/products/npfamily/ixp2800.htm.

[11] G. Reinman and N. P. Jouppi. CACTI 2.0: An integrated cache timing and power model. Technical Report WRL Research Report 2000/7, Western Research Laboratory, Palo Alto, CA, Feb. 2000.

[12] P. Shivakumar and N. P. Jouppi. CACTI 3.0: An integrated cache timing, power and area model. Technical Report WRL Research Report 2001/2, Western Research Laboratory, Palo Alto, CA, Aug. 2001.

[13] Silicon Access Networks, San Jose, CA. *iFlow Family Overview*, 2002. http://www.siliconaccess.com/products/iFlow_DP3_PB_2.6.pdf.

[14] T. Wolf. *Design and Performance of a Scalable High-Performance Programmable Router*. PhD thesis, Washington University, St. Louis, MO, May 2002.

[15] T. Wolf and M. A. Franklin. CommBench - a telecommunications benchmark for network processors. In *Proc. of IEEE International Symposium on Performance Analysis of Systems and Software (ISPASS)*, pages 154–162, Austin, TX, Apr. 2000.

[16] T. Wolf and M. A. Franklin. Design tradeoffs for embedded network processors. In *Proc. of International Conference on Architecture of Computing Systems (ARCS) (Lecture Notes in Computer Science)*, volume 2299, pages 149–164, Karlsruhe, Germany, Apr. 2002. Springer Verlag.