ESTIMA: An Architectural-Level Power Estimator for Multi-Ported Pipelined Register Files

Kavel M. Büyükşahin
ECE Dept., University of Illinois at Urbana-Champaign
Urbana, Illinois 61801, USA
buyuksah@uiuc.edu

Priyadarsan Patra
Intel Labs, Intel Corporation
JF4-211, NE 25th Avenue
Hillsboro, OR 97124, USA
priyadarsan.patra@intel.com

Farid N. Najm
ECE Department
University of Toronto
Toronto, Ontario, Canada
f.najm@utoronto.ca

ABSTRACT

We introduce an architectural-level power, area, and latency estimator for multi-ported, pipelined register files. Strengths of the proposed approach include the handling of pipelined operation and clock power, the simulation-based device size estimation, and the ability to handle user-specified timing constraints. The model proposed can be used as a stand-alone estimation and design exploration tool for register files and register-file type structures, or it can be incorporated into a high-level performance simulator to add power estimation capabilities.

Categories and Subject Descriptors

B.7 [**Integrated Circuits**]: Design Aids

General Terms

Design

Keywords

Area estimation, power estimation, register files, processor

1. INTRODUCTION

The recent accelerated increase in the power consumption of modern microprocessors has resulted in power becoming one of the most important, if not the most important, criteria for design exploration. It is no longer sufficient to explore the design space with the purpose of high performance alone. Decisions made at the architectural level have the potential of biggest impact on the power consumption of the final chip. Therefore, it is no longer acceptable to postpone the design decisions that affect the power until the back-end design phase. It is very important to expose the architects to the power/performance and floorplan tradeoffs. This means that there is a need for architectural-level power models for the building blocks of a microprocessor.

Multi-ported register-file-type arrays are very commonly used structures in modern microprocessors. The architectural register files (integer, floating point, etc.), the data and instruction caches, cache tag arrays, register alias table, branch predictors, and the instruction queue are all examples of multi-ported arrays. For the remainder of the paper, the abbreviation RF stands for only register file type arrays. Their large number, and usually considerable size, make RFs very important structures in terms of the power dissipation of the processor. Furthermore, their regular structure makes them very good candidates for architectural level power modeling. RFs in modern microprocessors are characterized by their large signal voltage swings (in contrast to cache arrays), and by their large number of ports (in contrast to their ASIC counterparts, microprocessor RFs may have anywhere from 2 to over 15 read/write ports).

This paper describes *Estima*, an architectural-level power, area and latency model for multi-ported, pipelined register files. *Estima* can be used as a stand-alone estimation tool for array-type structures, or it can be incorporated into an architectural-level performance simulator to add power estimation capabilities.

1.1 Existing work

There has been considerable effort spent on creating energy models for RFs in the literature [5, 3, 1, 4]. All of these models are based on some assumptions for the structure and design style of the RFs. They model the energy dissipation of a RF in terms of read- and write-access energies based on the capacitance switched at the energy dissipating nodes. Zyuban's work [5] concentrates on the energy complexity of the various RF styles, and introduces an energy model for multi-ported RFs. Although the model is useful for a first order comparison of different RF styles, it is not detailed enough to be used for a more accurate analysis. The main shortcomings of the model introduced are the lack of clock power, sizing of the various devices, a non-flexible timing structure, and the lack of consideration for pipelined operation. Furthermore, the model is based on the assumption that the cell dimensions are wire-limited. This assumption holds only for RFs with a large number of ports, and breaks down for moderate or low numbers of ports.

Brooks *et. al.* [1] introduce *Wattch*, an architectural-level power simulator based on the SimpleScalar [2] framework and various architectural-level power models for the building blocks of a processor. Their register-file model is derived from Zyuban's model, and has the same shortcomings.

Kamble *et. al.* [3] introduce an analytical energy model for low-power caches based on the hit/miss rates and number of cache read and write accesses reported by a detailed performance simulator. Their cache energy model assumes the capacitances of the power hungry nodes (such as bit-lines and word-lines) are available, and concentrate on estimating the number of transitions on these nodes. This model also lacks the consideration for clock power and pipelined operation.

Cacti [4] is a well-known cache delay, area, and power analysis tool. As the previous models introduced in this section, *Cacti* also lacks the consideration of clock power and the pipelined operation. Furthermore, it uses very simple first-order transistor models to size the various devices.

1.2 Contributions of this work

As mentioned above, *Estima* is a parametrized power, area, and latency model for multi-ported, pipelined register files. Its basic approach to power estimation is similar to the above mentioned cache and RF models. The main contributions of *Estima* that make it potentially more useful than these other models can be summarized under the following four headings:

Pipelined operation

With the clock period decreasing and the RF size going up, it is no longer possible to assume that RF accesses occur in a single cycle (or phase). This introduces the need for hierarchical bit-line schemes and multi-cycle access. *Estima* can handle up to three levels of bit-line hierarchy, and it can be easily extended if more levels are needed.

Clock power

Clock nodes (precharge nodes for bit-lines, and clock inputs of domino gates) contribute significantly to the overall power consumption of even a single cycle RF. When pipelined operation is introduced, this contribution becomes even more pronounced as the pipeline latches are inserted at pipe boundaries. *Estima* handles latch clock power as well as the precharge and domino clock power.

User-defined timing constraints

Timing constraints are very influential in determining the power consumption of any circuit. RFs are no different in this aspect. It is possible to build a lower power RF by relaxing the timing constraints, and vice versa. *Estima* gives complete control of timing constraints to the user, making the timing scheme user-configurable as opposed to making assumptions on how long a particular stage will take during a read/write operation.

Simulation-based device sizing

Device sizes affect the power consumption of a RF by determining the switched capacitance at a node along with the interconnect capacitance. Therefore, it is very important to have realistic device sizes to have any kind of accuracy (absolute or relative) in the power model. *Estima* uses a simulation-based library-independent device sizing algorithm, where it actually runs circuit level simulations of power nodes with the supplied timing constraints to size the various devices (read access transistors, precharge devices, word-line drivers, etc.).

2. METHODOLOGY

In the following sections, we will describe the methodology used in the development of *Estima*.

2.1 Preliminary studies

There are two very important questions that have to be answered before starting the development of a parametrized architectural-level power model for RFs: *(i)* What parameters will the model be based on, and *(ii)* where is the power consumed in a real RF.

To answer the first question, we sought input from architects and circuit designers on what parameters they think are relevant to power consumption, and what features should be included in a useful power model. As a result of these interactions, we have determined that relevant parameters at the architectural level are: number of registers, bit-width of registers, number of read and write ports, and architectural and data activity factors. Consequently, these became the architectural level parameters our model is based on.

To answer the second question, we studied a number of existing RFs in a recent microprocessor, and determined the power breakdown of different nodes. Fig. 1 shows the approximate power breakdown for a number of large signal, single-cycle RFs.

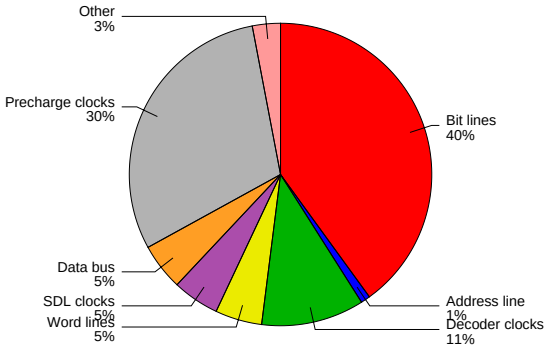

Figure 1: Power breakdown of register files.

This figure clearly shows that the dominant components of power consumption in the RF are the bit-lines and the clock lines. Once again, this shows that it is very important to account for clock power as well as the bit-line and word-line power.

2.2 Basic power model

Our basic power model is based on computing the energy-per-access (EPA) for read and write operations on a single port. This number, combined with the architectural read and write activity factors and the clock frequency, gives the power consumption of the register file.

The EPA consumed at a single node, i, for charging and discharging the node can be written as

$$EPA_i = C_i \cdot V^2 \tag{1}$$

where C_i is the total device and interconnect capacitance at node i and V is the supply voltage. The total EPA for a read (write) access in a RF can be written as

$$EPA_{rd(wr)} = \sum_i EPA_i \cdot \#nodes_i \tag{2}$$

where i enumerates all the different types of nodes that switch during a read (write) operation, EPA_i is computed

by (1), and $\#nodes_i$ is the number of nodes of type i that are switching for that particular access. This number is determined by considering the total number of such nodes in the RF and the input data statistics.

Once we have the read and write EPAs of the RF, getting the power consumption is a simple task of multiply and add

$$P_{RF} = (\alpha_{rd} \cdot EPA_{rd} + \alpha_{wr} \cdot EPA_{wr}) \cdot f_{clk} \quad (3)$$

where α_{rd} and α_{wr} are architectural read and write activity factors, f_{clk} is the clock frequency, and EPA_{rd} and EPA_{wr} can be computed using (2).

Node capacitance

At the heart of the power estimation algorithm lies the computation of the capacitance of a specific node. Our node capacitance model lumps the capacitance of all the devices connected to a node and the interconnect capacitance in a single number for power computation purposes. This number is computed by

$$C_i = l_i \cdot c_{mi} + W_{gi} \cdot c_g + W_{di} \cdot c_d \quad (4)$$

where l_i is the interconnect length at node i, c_{mi} is the capacitance-per-unit-length of the metal layer that node i is on, W_{gi} and W_{di} are the total width of gate and diffusion connected devices at i, and c_g and c_d are the unit gate capacitance and the unit diffusion capacitance respectively. In this equation, c_{mi}, c_g and c_d are determined by the particular process technology, and lengths and widths are determined by the physical dimensions of the register file and the device sizes respectively. Once again, we would like to emphasize that the lumped capacitance approach is used only for power computations. For device sizing and timing analysis, the nodes are modeled as distributed capacitance wires.

Let us demonstrate this with an example node. Fig. 2 shows the model of a local read bit-line from a hypothetical RF. This structure is essentially a distributed domino AND-OR-INVERT gate driving a static NAND gate. In this particular RF, there are 4 entries on a local bit-line, and hence the 4 pull-down structures. The length of the interconnect is l_m, there is a pre-charge/keeper structure, and a NAND gate at the end to combine the read value with another local bit-line. Assuming the interconnect is on metal layer 2, we can get the capacitance of this node using (4) as

$$C_{lrdbl} = l_m \cdot c_{m2} + (W_{kpinv} + W_{nand}) \cdot c_g + (4 \cdot W_{txrd1} + W_{txpre} + W_{txkp}) \cdot c_d \quad (5)$$

2.3 Pipelined register files

As the sizes of the RFs increase (on-chip cache sizes have increased dramatically in recent years), and the clock periods decrease, it becomes more and more difficult to read (write) from (to) a RF in a single phase or cycle of the clock. One may still try to perform reads or writes in a single cycle, but this would require stronger (i.e., larger) devices, and hence, higher dynamic and leakage power. In a small RF, one may have local read bit-lines driving the output data bus directly in one cycle, whereas in a larger RF, local read bit-lines would usually be driving a higher level global read bit-line, which in turn drives an even higher level global-global read bit-line, which drives the output data bus. This will introduce new power consuming nodes to the RF; namely the new hierarchy of bit-lines, and the latches, and latch clocks introduced at the pipe boundaries. The pipelining will also introduce latency of the reads (writes), which may effect the performance of the RF, and consequently the whole chip. Therefore, an estimate of power without an appropriate estimate of latency does not mean much.

Our model is designed from ground up to handle multiple levels of bit-line hierarchy and pipelining. As it has the capability of handling clock lines, addition of the latch clocks to the RF does not require any special handling.

In the next section, we will discuss the algorithm used to compute the physical dimensions and the latency of the RF.

2.4 Physical dimensions and latency

Physical dimensions play a very important role in determining the power consumption of an RF. Actually, they influence the power consumption in more than one way: *(i)* they determine the length of the wires in the RF, hence directly affect the power consumption by determining the capacitance of the nodes, and *(ii)* they impose pipelining constraints, indirectly affecting power by introducing additional power consuming nodes. Therefore, it is critical to have a good model to estimate the physical dimensions of the RF.

High-level RF model

The high-level RF model used in *Estima* is illustrated in Fig. 3. This model is of a single pipe in a pipelined configuration. A multi-cycle RF may have more than one of these "blocks" in either physical dimension.

Physical dimensions of the memory cell

Memory cell size is the single most important factor in determining the physical dimensions of the RF. Although a single memory cell is usually small, they are replicated many times in both dimensions, and hence dominate the other blocks in the RF.

For RFs with a large number of ports, it is possible to claim that the memory cell size is metal limited, and approximate the cell height and width as $mp \cdot (N_{wr} + N_{rd})$, where mp is the metal pitch in the particular dimension, and N_{wr} and N_{rd} are the number of write and read ports respectively. However, for register files with a small or moderate number of ports, this model does not quite hold. Therefore, we have chosen not to use this approximation for the cell size, but to study existing RFs for memory cell sizes, and derive an empirical relationship between the cell dimensions and the number of ports. We have found that it is possible to approximate this relationship with a simple first order polynomial

$$height = a_h + b_h \cdot \#ports$$
$$width = a_w + b_w \cdot \#ports \quad (6)$$

where a_h, b_h, a_w and b_w can be obtained from correlation analysis of existing RFs, and scaled for new process technologies. $\#ports$ is the total number of read and write ports of the RF. As one can see, (6) reduces to the metal limited form for a large number of ports.

Physical dimensions of segment drivers and decoder

As mentioned in the previous section, the dimensions of the segment drivers (local-to-global, global-to-global-global, etc.) and the decoder block are not critical in determining

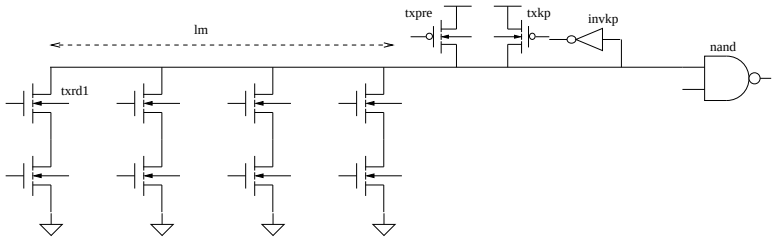

Figure 2: Local read bit-line.

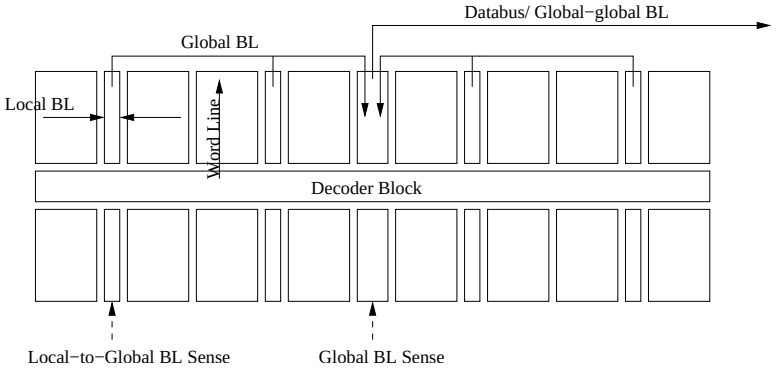

Figure 3: High-level model of the RF.

the dimensions of the RF. However they can be included in the model for completeness and greater accuracy. We have performed a similar correlation analysis for the segment drivers and the decoders to determine the dimensions as a function of number of ports.

Area and latency computation

Arguably, one of the most important part of *Estima* is the algorithm to determine the configuration of the RF. Configuration of a RF has many aspects:

- *Number of memory cells on a local bit-line*: This determines the length of the local bit-lines and the need for higher levels in bit-line hierarchy, and therefore affects the power consumption of the RF. It is a design decision, but cannot be assigned arbitrarily. A large number of cells on a bit-line will cause the bit-line charge to leak too rapidly and result in incorrect operation.

- *Number of pull-down devices on a global bit-line*: Similarly, this parameter determines the need for higher levels of bit-line hierarchy, and affects power consumption. This too, cannot be assigned arbitrarily due to leakage considerations.

- *Maximum width and height of a pipe-stage*: This configuration parameter determines whether we need a pipelined implementation. It affects the power indirectly by introducing more levels of bit-line hierarchy and latches in pipe boundaries. It is a function of the process technology.

- *Bit-line folding*: As seen in Fig. 1, most of the power in a RF is consumed in the bit-line hierarchy. Therefore, bit-lines are the natural targets for the power reduction techniques. One such technique that can be applied to RFs with a very high aspect ratio to decrease the bit-line length is bit-line folding. A RF can be folded multiple times to correct the aspect ratio, or to make the RF fit in a constrained space. Folding is a design decision that can be investigated in the design exploration phase.

Our "configuration algorithm" takes all these configuration parameters, and tries to come up with the optimum configuration that satisfies these constraints. It computes the number of local bit-lines in a bit-slice, the number of global bit-lines and the number of pipe stages in the bit-line and word-line dimensions. The last two numbers, when added, give the latency of the RF. Combining all these numbers with the cell, segment driver, and decoder sizes allows one to compute the height, width, and area of the RF.

The configuration process completely defines the high level model of the RF at hand. The numbers and types of nodes, and the lengths of interconnect wires are all available at this point. The only component missing in the EPA computation performed by (2) is the device sizes of (4). In the next section, we will introduce our device size estimation methodology, wherein lies another strength of *Estima*.

2.5 Device sizes

We start this section by introducing the devices that are relevant to power consumption of the RF, and hence need to be estimated. To do this, we should once again look at the power breakdown of Fig. 1. This figure suggests that, one can actually get a very good estimate of the power consumption if one can estimate the power consumed by the bit-lines and various clock nodes. This means that the size of the pull-down and precharge devices on the bit-lines (refer to Fig. 2 for an example), and all clocked devices in the design should be known in order to get an accurate estimate of the power consumption. Unfortunately, the circuit level

details of the RF are usually not available at the architectural design exploration phase. This means that, in order to be able to estimate the power of a RF, one needs an accurate device size estimation method.

One way to tackle this problem would be to impose artificial timing constraints on rise and fall times of various lines, and to estimate the device sizes that will result in such timing behaviour using simple analytical formulas. This is the approach taken in [5, 4]. This approach would work as a first approximation, but a more thorough approach is needed for greater accuracy.

In *Estima*, we tackle the device size estimation problem by using a simulation-based sizing algorithm to size various devices. At the core of the method, there is a library-independent, technology-dependent, iterative device sizer. Wrapping this sizer is a simulated-annealing based optimization loop that tries to balance the device and wire delays for obtaining minimal sized devices that satisfy the timing constraints. As of the writing of this paper, the timing constraints are user-defined. This can be viewed both as a strength and a weakness of the approach. On the plus side, it gives the user full control on the power/performance trade-off by letting him/her set different timing constraints and obtain different power results. On the negative side, it may be too much information to ask at such a high-level of abstraction.

The way the sizer works is as follows: the user is asked to provide the length and load of the data bus that the RF needs to drive. He/she also has to provide the detailed timing requirements of individual stages (such as the local bit-line, global bit-line, read word-line, etc.). Using the load information of the data bus and the required timings, the sizer sizes the output drivers of the RF. This value is then assigned as the load for the final stage of the RF, and, together with the timing requirements, used to size the devices on this stage. This exercise is repeated until all the relevant devices in the register file are sized according to the timing requirements and the output load.

Finally, with the configuration information, the device sizes, and the activity factors at hand, the power consumption of the RF is computed using (3).

2.6 Implementation details

We can summarize the power estimation process introduced in this paper as follows:

```
Read in the architectural parameters
Read in the configuration constraints
Determine the configuration of the RF
Compute the area and latency of the RF
Read in timing constraints
Run the device sizer to estimate sizes
Read in the data statistics
Compute the EPA using (2)
Read in the architectural activity factors
Compute the power using (3)
```

We have implemented this RF model as a stand-alone tool called *Estima*. *Estima* is mainly coded in Ruby, a powerful, object-oriented scripting language. The reasons for choosing Ruby to implement the tool was merely practical. Ruby has a clean syntax and full support for object oriented design, making it very suitable for fast-prototyping. It has powerful extensions to facilitate implementation of graphical user interfaces, which is critical for a stand-alone tool. And the simulation based device sizer we used, although written in C++, interfaces very nicely with Ruby.

In the next section, we will introduce some examples of what *Estima* can do, and how it can be used for design exploration.

3. RESULTS

3.1 Power breakdown

To verify the relative accuracy of our model, we conducted an experiment on an existing RF from a recent microprocessor. To obtain a reference point to compare our model with, we ran an industrial switch-level simulator on the RF circuit with a particular input stream and obtained the power breakdown. Then, we ran our model with the parameters of the RF and the statistics of the input stream used. In Table 1, one can see the results of this experiment. There is a very good agreement between the power breakdown obtained by using *Estima* and the switch level simulator (column heading SLS in the table).

Table 1: Power breakdown by node type

Node type	SLS	*Estima*
data bus	5%	5.27%
bit-lines	40%	36.6%
word-line	5%	7.3%
pre-charge ck	30%	31.5%
SDL ck	5%	5.23%
Address line	1%	1.11%
Decoder ck	11%	12.9%
Others	3%	-

3.2 Effect of timing on RF power

In this subsection, we will introduce the results of an experiment conducted to support our claim that the RF power depends very much on the timing constraints imposed on the RF, and also to demonstrate how *Estima* can be used as a means to observe the power/performance tradeoff. For this experiment, we fixed the architectural parameters and the input statistics of a hypothetical RF. Then, we chose a particular timing scheme as our reference timing, and ran *Estima* to estimate the power consumption. To see the effect of timing constraints, we then changed the timing constraints by 5% and 10%, and estimated the power consumption at these timing points. Fig. 4 shows the results of this experiment. As one can see, even a modest reduction of 10% in timing can increase the power consumption by more than 25%. Similarly, a 10% relaxation in constraints can reduce the power consumption by as much as 15%.

3.3 Effects of bit-line folding in power consumption

In this section, we will demonstrate how *Estima* can be used to comprehend the effects of bit-line folding on power. For this experiment, we once again fixed the architectural parameters and the input stream of the RF and obtained the reference power consumption value. Then, we used *Estima* to fold the bit-lines once, twice and three times, consecutively. Fig. 5 shows the results of this experiment.

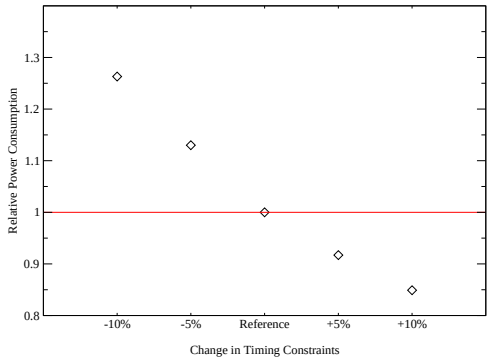

Figure 4: Effect of timing on RF power.

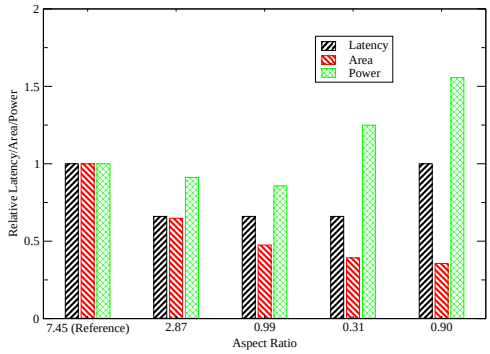

Figure 5: Effect of bit-line folding on RF latency, area and power.

This experiments points out an interesting fact. The power consumption of a particular RF is lowest when the layout aspect ratio is close to 1. The explanation of this lies in understanding the power consumption model of the RF. When the register file is skewed in the bit-line direction (has more registers than the number of bits in a register), the bit-lines are long, the devices are larger to meet the timing requirements, and therefore they consume most of the power. As we fold the RF, the bit-lines become shorter, bit-line devices get smaller, the word-lines and the clock lines get longer. This reduces the power consumption while shifting it from bit-lines to word-lines and clock lines. If we keep folding the RF, the word-lines and the clock lines become very long, and the power consumption goes up again.

3.4 Number of ports vs. area and power

To observe the effect of the number of read/write ports on RFs, we fixed all parameters except for the number of read and write ports and ran *Estima* for different combinations of read and write port numbers. Fig. 6 shows the results of this experiment.

As expected, both the layout area and the power consumption of the RF increase with increasing number of ports.

4. CONCLUSION AND FUTURE WORK

In this paper, we have introduced an architectural-level, power, area, and latency estimator for multi-ported, pipelined register files. The strengths of the proposed approach are the handling of the pipelined operation and clock power, the simulation based device size estimation, and the ability to handle user-specified timing constraints. The proposed model can be used as a stand-alone estimation and design

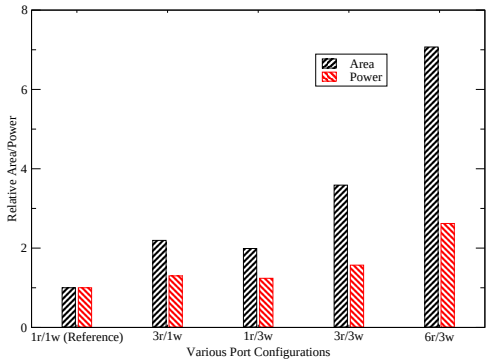

Figure 6: Effect of number of ports on RF latency, area and power.

exploration tool for register files and register-file type structures, or it can be incorporated into a high-level performance simulator to add power estimation capabilities.

There are two main areas that need attention to make the model even more practical and useful at the architectural level. At the moment, the model needs the user to supply the maximum width and height of a pipe-stage in the particular process technology at hand. As the model has access to the process details (the sizer is simulation based, and therefore "technology aware"), it should be possible to determine these values automatically. We are currently working on this problem, and will incorporate it in the model in the near future. The second area that can be improved upon is the notion of user-defined timing constraints. The model should be able to work even if the user does not want to provide detailed timing requirements by using self-computed timing requirements based on the technology and configuration. This problem is a bit involved, and its solution requires more studies. We are investigating this issue as well.

5. ACKNOWLEDGEMENTS

Omitted for blind review

6. REFERENCES

[1] D. Brooks, V. Tiwari, and M. Martonosi. Wattch: A framework for architectural-level power analysis and optimizations. In *Proc. Intl Symposium on Computer Architecture*, pages 83–94, Vancouver, BC, June 2000.

[2] D. Burger and T. M. Austin. The SimpleScalar Tool Set, Version 2.0. In *Computer Architecture News*, pages 13–25, June 1997.

[3] M. B. Kamble and K. Ghose. Analytical energy dissipation models for low power caches. In *Proc. International Symposium on Low Power Electronics and Design (ISLPED)*, pages 143–148, 1997.

[4] P. Shivakumar and N. P. Jouppi. CACTI 3.0: An integrated cache timing, power and area model. Technical report, Compaq Western Research Laboratory, 250 University Avenue Palo Alto, California 94301 USA, Aug. 2001.

[5] V. Zyuban and P. Kogge. The energy complexity of register files. In *Proc. International Symposium on Low Power Electronics and Design (ISLPED)*, pages 305–310, 1998.