

## Energy Dissipation in General Purpose Processors\*

Ricardo Gonzalez and Mark Horowitz

Center for Integrated Systems, Stanford University, Stanford, CA 94305-4070

{ricardog,horowitz}@chroma.Stanford.EDU

### Abstract

In this paper we investigate how super-scalar issue and pipelining affect the energy-delay product of general purpose processors. We show that for idealized machines pipelining gives approximately a 2X improvement in energy-delay product, while super-scalar issue only improves the energy-delay product by a small amount. We then show that real processors are roughly half as efficient as the idealized machines. We also show that improving the efficiency will be difficult since the overhead is distributed among many units, none of which represent a significant fraction of the energy dissipation.

### Introduction

The interest in lowering the power of a processor has grown dramatically over the past few years. This has led to an increasing diversity in the processors available. However, processors that vary by an order of magnitude in power and performance have remarkably similar energy-delay products [2]. In this paper we investigate whether exploiting parallelism can significantly improve the energy-delay product of a processor. We first look at three idealized machines that have no overhead to understand what are the intrinsic factors that set the energy and energy-delay product of a processor. We then look at two processors we have designed.

#### Lower Bound on Energy and Energy-Delay Product

All processors fundamentally perform the same operations. They fetch instructions from a cache, use that information to determine which operands to fetch from a register file, then either operate on this data or use it to generate an address to fetch data from the cache, and finally store the result. These operations are sequenced using a global clock signal. High performance processors have sophisticated architectural features that improve the instruction fetch and data cache bandwidth and exploit more instruction level parallelism. In a real processor, of course, there is a lot more overhead, but we ignore this overhead since it is not essential to the functionality of the processor. We aggressively assume that the energy cost of performing computation and the cost of communication between memories is zero. We only consider the energy needed to read and write memories, and, where required, the energy to clock storage elements, such as pipeline latches. Furthermore, these ideal processors only dissipate energy that is absolutely necessary, and there are no unwanted transitions, such as glitches.

The ideal machine is the most basic compute element, similar to DLX [1]. The processor consists of a simple state machine that fetches an instruction, fetches the operands, performs the operation, and stores the result. The processor is not pipelined so we model the execution time by assuming control transfer instructions take 2 cycles, ALU operations take 3 cycles, and cache operations take 4 cycles. Since this

machine is not pipelined, we assume no energy is dissipated in clocking. Only the energy required to read and write the caches and register file is taken into consideration.

The ideal pipelined processor is similar to the one above, except that we use pipelining to exploit instruction level parallelism. Thus the change in energy-delay product of this machine compared to the ideal will be due entirely to pipelining. We use the traditional MIPS [3] or DLX 5-stage pipeline. This processor is very similar to the pipelined version of DLX. Since it is not possible to build a pipelined machine without some kind of storage elements, in addition to the energy required to read and write memories—as above, we also take into account the energy required to clock the latches in the pipeline and the PC chain.

The ideal super-scalar machine is similar to the previous one, except that it can execute a maximum of two-instructions per cycle. This machine is an idealization of the TORCH processor [4] which is described later.

Figure 1 shows the total energy required to complete a benchmark. All numbers in this section have been normalized to the corresponding value for the super-scalar processor. As we would expect the unpipelined machine has the lowest energy per instruction, and the super-scalar machine has the highest. However, energy per instruction is not a

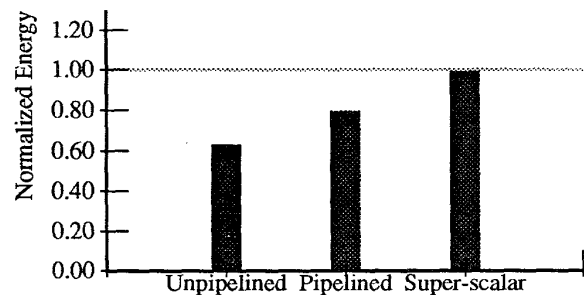


Figure 1. Normalized energy.

good metric of efficiency since it does not factor in the time required to execute the benchmark [2]. If we look instead at the energy-delay product, shown in Figure 2, we can see that the unpipelined machine has lower energy efficiency. By trading excess performance for lower energy—by reducing the supply voltage for example—the other two processors could operate at the same performance as the unpipelined machine but dissipate half the energy.

Pipelining provides a big boost in performance for very little energy cost, so it gives almost a 2X improvement in energy-delay product. Super-scalar issue, on the other hand, only gives a small improvement in energy-delay product. In this simple model, a processor with a wider parallel issue would be of limited utility. The basic problem is that the amount of instruction level parallelism is limited (in the integer applications we have run), so the effective parallelism is

\* Funding for this research provided by ARPA under contract J-FBI-92-194.

small. Yet increasing the peak issue rate increases the energy of every operation, even in this simple machine because the energy of the memory fetches and clocks also increases.

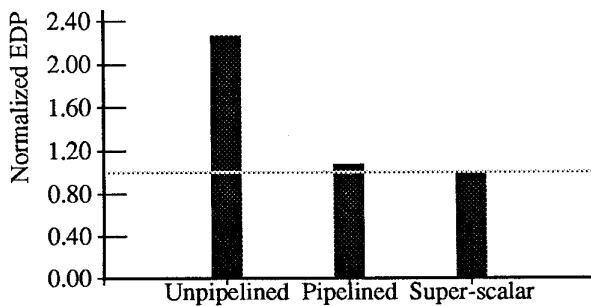


Figure 2. Normalized energy-delay product.

### Current Implementations

To see how real processors compare with the ideal machines described previously, we now look at two processors we have designed, a simple RISC machine and a super-scalar processor called TORCH [4].

The simple RISC machine is similar to the original MIPS R3000 described by Kane [3], except that it includes on-chip caches. This processor is similar to the ideal pipelined machine except that it accounts for all the energy required to complete the instruction and it includes all the overhead associated with the architecture, such as the TLB and co-processor 0.

TORCH is a statically scheduled two-way super-scalar processor. TORCH supports conditional execution of instruction directed by the compiler. The processor includes shadow state that can buffer the results of these conditional instructions until they are committed. The machine is statically scheduled, so the compiler determines which instructions can be executed simultaneously and when NOP's must be inserted into the pipeline. Instructions are encoded in a 40-bit word and are packed in main memory using a special format. Instructions are unpacked as they are written in the first level instruction cache.

Both processors have been optimized for low-energy dissipation. Using simple optimizations that do not adversely affect the critical paths—such as clock gating, selective activation, and removing speculative operations—we were able to reduce the energy dissipation by approximately 25%.

Figure 3 shows the energy-delay product for both processors and the ideal super-scalar machine. Unlike the previous graph, the single issue machine is slightly more efficient than the dual-issue processor. The overhead in a super-scalar machine is worse than that of a simple machine and it overwhelms the modest gains in the energy-delay product gotten from parallelism.

### Energy Breakdown

To understand why the ideal machine is roughly twice as efficient, we investigated the energy breakdown for all three processors. Figure 4 shows the energy dissipation broken down into four different categories. Interconnect represents only the fraction of the energy dissipated in driving

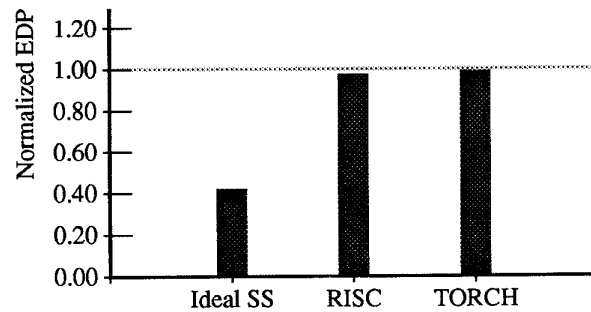


Figure 3. Normalized Energy-delay product.

global wires. Other represents everything else. The figures are normalized to the total energy dissipation of TORCH.

The energy dissipated in the on-chip memories is almost the same in the ideal machines than in our implementations. This means that we were successful in reducing the number of unnecessary cache accesses. The clock power in the real machines is higher since there is a large amount of state that we did not consider in the ideal machines. Global interconnect and computation each represent only about 15% of the total power. Thus reducing the energy dissipated in combinational logic or global interconnect will only provide marginal improvements in energy-delay product.

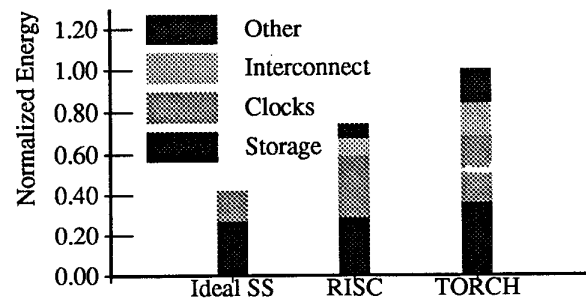


Figure 4. Energy breakdown.

### Conclusions

We have shown that even when we ignore overhead super-scalar issue does not significantly improve the energy-delay product of a processor. Unlike pipelining—which provides almost a factor of 2 improvement—the energy cost of super-scalar issue is almost the same as the increased performance. Reducing the overhead is difficult since it is dissipated in a variety of units, none of which represents a significant fraction of the total energy.

### Bibliography

- [1] J. L. Hennessy, et al., *Computer Architecture A Quantitative Approach*. Morgan Kaufmann Publishers, San Mateo, Ca, first ed., 1990.
- [2] M. Horowitz, et al., "Low-power digital design". In *Symposium on Low Power Electronics*, volume 1, pp. 8–11. IEEE, October 1994.
- [3] G. Kane, *MIPS RISC Architecture*. Prentice Hall, Englewood Cliffs, NJ 07632, 1988.
- [4] M. D. Smith, *Support for Speculative Execution in High-Performance Processors*. PhD thesis, Stanford University, November 1992.