Appears in the Proceedings of the International Conference on Computer Design, September, 2002

A Circuit–Level Implementation of Fast, Energy–Efficient CMOS Comparators for High–Performance Microprocessors

Oguz Ergin, Kanad Ghose, Gurhan Kucuk, Dmitry Ponomarev

Department of Computer Science
State University of New York, Binghamton, NY 13902–6000
e–mail:{oguz, ghose, gurhan, dima}@cs.binghamton.edu
http://www.cs.binghamton.edu/~lowpower

Abstract

Datapath components in modern high performance superscalar processors employ a significant amount of associative addressing logic based on the use of comparators that dissipate energy on a mismatch. These comparators are used to detect a full match, but as mismatches are much more common than full matches in some components of the CPU, considerable energy–inefficiencies occur within the associative logic. We propose the design of two new comparator circuits that predominantly dissipate energy on a match, thus resulting in very significant savings in comparator power dissipation. The proposed designs are evaluated using SPICE simulations of actual VLSI layouts of the comparators in 0.18 micron 6–metal layer process and micro–architectural level statistics.

1. Introduction

Contemporary microprocessors, particularly superscalar CPUs, rely on the use of a significant amount of associative matching logic to support register renaming, out–of–order execution and virtual memory mechanisms. The traditional comparator circuit (also known as a pull–down comparator), shown in Figure 1, dissipates energy on a mismatch in the input arguments (comparands). The output is precharged, and pulled down on a mismatch in any bit position during the evaluation phase, causing energy dissipation. Notice that the effective output loading of traditional comparators is high: this is equal to the diffusion capacitances of 2C n–transistors plus the load capacitance, where C is the number of bits compared. This results in considerable power dissipation in the case of a mismatch. As mismatches are much more frequent than matches in some components of superscalar paths that make use of associative addressing, the use of traditional comparators is not an energy–efficient solution.

The need for a power–efficient comparator was first mentioned in [2], but no specific circuit solution was proposed. In this paper, we introduce two fast CMOS comparators, capable of comparing up to 8 bits that dissipate energy predominantly on a full or a partial match in the bit positions of the comparands. (The designs can be extended to compare more than 8 bits; 8–bits suffice for the application studied in this paper.) One of the proposed designs is an improved version of the domino–style comparator circuitry introduced in [5]. The second design does not use the domino logic and results in faster response time and lower energy dissipations.

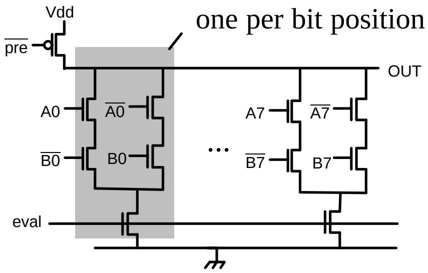

Figure 1. Traditional 8–bit pull–down comparator

2. A Two–Stage Domino–Style Comparator

Figure 2 depicts the first proposed comparator circuit. It represents a noticeable improvement, mainly in terms of response time, over our earlier proposed design based on three stages of domino logic [5].

The circuit of Figure 2 compares two 8–bit comparands, A7A6..A0 and B7B6..B0 using a combination of domino–style logic and pass transistor logic. P–transistor pass logic blocks (such as P in Figure 2) compare two bits of the comparands at a time. A high voltage level V_s is passed on to the right by each of these P–transistor blocks when both input bits that they compare match. Each P–transistor block drives the gate of an n–transistor (such as Q1) that is part of a discharge path of the domino logic stages. The precharging signal is cut off during the evaluation phase and an evaluation signal is applied to each stage of the domino logic. The first domino stage pulls down the output of the first stage during the evaluation phase only if the pass transistor logic using p–devices (greyed box, P) driving the gates of Q1 and Q2 are both on. This happens when all 4 least significant bits are equal. The n–transistors Q3 and Q4 prevent false matches caused by the charge accumulations in previous cycles. During precharging, these transistors turn on (dis is high) for a small time to discharge the charge stored in the previous cycle (which happens when the corresponding block matches) on the gates of transistors Q1 and Q2. The output of the comparator is discharged to ground only when *all* bits of the comparands match.

To reduce the charge stored at the gates of transistors Q1, Q2, Q8 and Q9 and hence, the energy dissipated when these gates are discharged, a voltage lower than V_{dd} can be used as V_s. On the flip side, this increases the circuit delay on a match and complicates the design because the additional voltage source has to be provided (which can be either derived from the outside of the chip or generated locally from the V_{dd}). The effects of various values of V_s on the delay of the circuit and its power dissipation are analyzed in detail in Section 4.

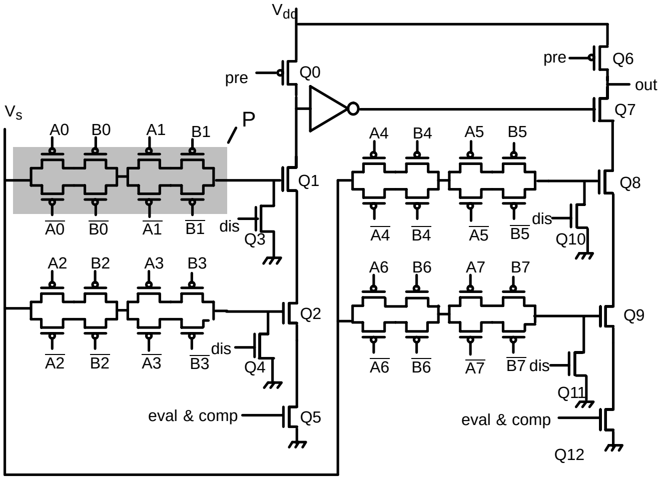

Figure 2. A Domino–Style Comparator

In this circuit, maximum power dissipation occurs on a full match; some power dissipation occurs during the discharge of the corresponding n–device gate when a partial match occurs. Since there is no discharging of the output, except in a full match condition, the dominant power dissipation is the discharge dissipation of the gates of the stacked n–devices. Since the probability of the lower 4 bits matching is small in all of the applications we consider, as revealed later, the comparator of Figure 2 dissipates power predominantly on a full match of all 8 bits of the arguments. The comparator of Figure 2 is effectively a dissipate–on–match comparator.

Compared to an 8–bit traditional comparator (Figure 1), that requires 40 n–devices and a large p–device for the precharger, the comparator of Figure 2 needs 35 p–devices and 12 n–devices. The layout area for the new domino–style comparator is 21% higher than the layout area of the traditional comparator in our implementation. The response time, as well as power dissipation of the new comparator is determined by the value of V_s, as detailed later.

3. A Pass Logic, Single Stage Comparator (PLSSC)

The second proposed comparator design, shown in Figure 3, avoids the use of domino–style logic altogether.

The pass transistor logic shown within the greyed box in Figure 3 passes a high logic level to the gate of the n–transistor Q1 when bits A7 and B7, as well as bits A6 and B6 of the comparands match. The series pulldown structure consisting of the devices Q1, Q2, Q3 and Q4 thus conducts when all 8 bits of the comparands are equal. The output of this comparator, precharged to V_{dd} by Q0 is thus discharged when all bits of the comparands are equal and when the evaluate device, Q5, is on. The n–transistors Q6, Q7, Q8 and Q9 discharge any accumulated charges when partial matches occur, just as in the design of Figure 2. The effective loading of the output is small: the diffusion capacitances of a small p transistor (Q0) and an n–transistor (Q1), plus the gate capacitances of whatever is driven by the output and wire capacitances. As in the case of the domino–style comparator from Section 3, a lower voltage than V_{dd} can be used to pass the high voltage level to the gates of n–devices.

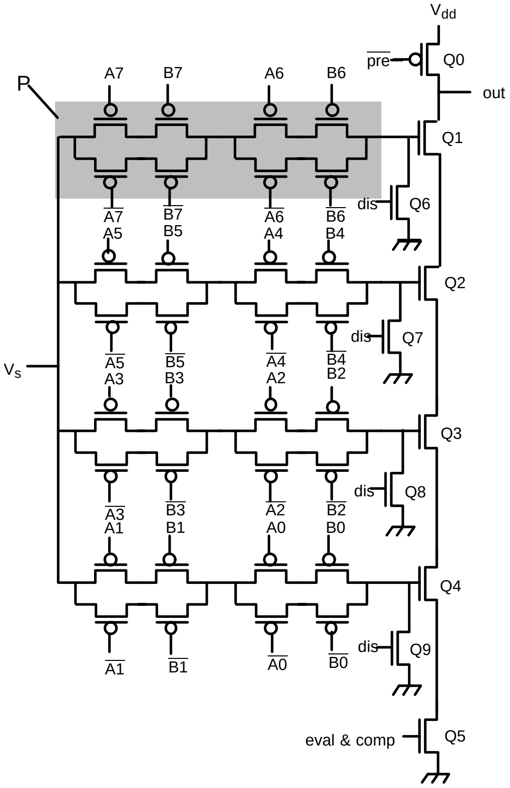

Figure 3. A Pass Logic based Single–Stage Comparator

The total number of devices used to implement the circuit of Figure 3 (33 p–devices and 9 n–devices) is one higher than what is needed to implement a traditional comparator.

The advantage of the domino–style comparator of Figure 2 over the PLSSC of Figure 3 is that the former design is a more scalable solution that can compare wider operands.

Both proposed comparators dissipate energy on partial matches while removing the charge stored on the gates of the pull–down transistors. Since the pull–down path of the PLSSC is not turned on until all of the P–blocks match, the comparator energy dissipation can be roughly estimated as the dissipation occurring during the discharge phase of one P–block in Figure 3, multiplied by the number of matching 2–bit groups. Of course, additional energy dissipation occurs on a full match. For the domino–style comparator of Figure 2, this additional dissipation also occurs in the match of the least significant 4–bits (bits 0 through 3) as well as on a complete match.

4. Timing and Energy Considerations

Figure 4 shows the timing diagrams of the traditional and the proposed comparators. In a typical cycle, the critical path is constrained by the evaluation delay of the traditional comparator, as seen in the bottom part of Figure 4. The delay of the precharge phase can be usually hidden by overlapping the precharge of the comparators with other activities, such as driving the tags or selecting the marked instructions for the execution when the comparators are used within an issue queue. The precharge delay is thus off the critical path.

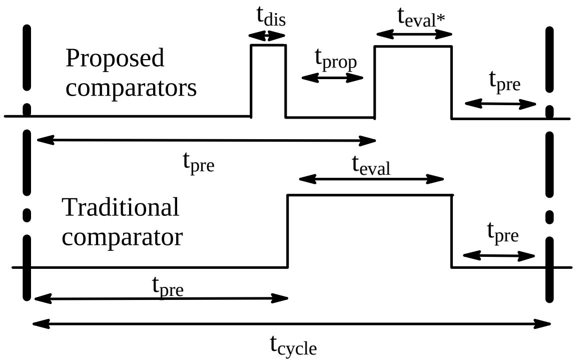

Figure 4. Timing diagrams of the traditional and the proposed comparators

For the proposed comparators, a discharge signal is needed for discharging the gates of pull–down transistors before the next evaluation signal arrives, to avoid false matches. After the discharge signal falls, it takes a finite amount of time (defined as tprop in the top part of Figure 4) for the high voltage level Vs to propagate through the matched comparand blocks to the gates of the pull–down transistors. While this propagation time can be overlapped with precharging, the inputs must be ready and stable before the discharge signal falls. For this reason, the propagation delay of the P–blocks is on the critical path of the proposed comparators. The comparator delay is therefore defined as the sum of the propagation delay and the evaluation delay.

The duration of the discharge signal is also important. It is desirable to keep it short to reduce the energy dissipation during the discharging. On the other hand, the duration of the discharge signal must be long enough to pull down the voltage levels at the gates of n–devices below the threshold voltage. One way to reduce the propagation delay is to increase the width of the p–devices used to implement the comparand's blocks. Doing so requires a higher capacitance to be discharged on partial matches, resulting in a longer discharge signal and higher power dissipation.

Figures 5 and 6 show the delay and power dissipation characteristics of the three comparator circuits discussed in this paper, measured using SPICE for full–custom layouts in the TSMC 0.18 micron CMOS process, with Vdd = 1.8 Volts. As seen from the graphs, increasing the value of Vs results in a reduction in the delay of the two proposed circuits. The area of the traditional comparator was optimized for the best performance. The worst case (1–bit mismatch) evaluation delay for the traditional comparator is 121 ps. At the expense of 7% area increase in the layout of the PLSSC over the traditional comparator, we managed to reduce its delay (as defined by tprop+teval*) to that of the traditional comparator for Vs=1.75V. However, the delay of our proposed domino–style comparator was at best 149 ps for Vs=1.8V because of the delay of inverter and the pass transistors.

5. Application to Superscalar Datapaths: Issue Queue

For estimating the energy savings possible with the use of the proposed comparators within the issue queue of a superscalar processor, we used the AccuPower toolset [7]. A detailed cycle–level and hardware–level superscalar

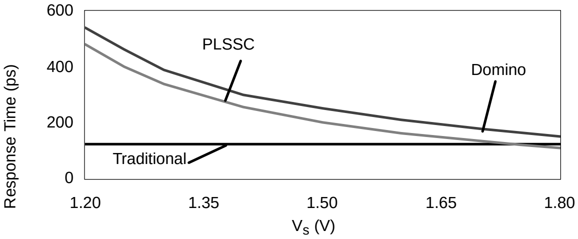

Figure 5. Variation of response time with V_s

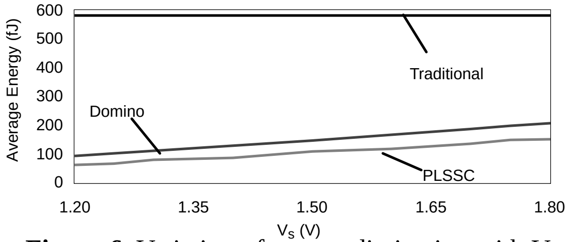

Figure 6. Variation of energy dissipation with V_s

processor simulator, with its origin in the Simplescalar simulator [1], was used to estimate the activities within the issue queue. We assumed 64–entry issue queue, 128–entry reorder buffer, 128–entry load/store queue and separate integer and floating point register files, 128 registers in number each. We simulated the execution of 10 integer (*bzip2*, *gap*, *gcc*, *gzip*, *mcf*, *parser*, *perlbmk*, *twolf*, *vortex* and *vpr*) and 8 floating point (*applu*, *apsi*, *art*, *equake*, *mesa*, *mgrid*, *swim* and *wupwise*) benchmarks from SPEC 2000 suite. Benchmarks were compiled using the Simplescalar GCC compiler that generates code in the portable ISA (PISA) format and run using the reference inputs. For each benchmark, the results from the simulation of the first 1 billion instructions were discarded and the results from following 200 million instructions were used. The microarchitectural–level data collected was exploited to use the new comparators more effectively, as described later.

We consider a 64–entry out–of–order issue queue for a superscalar processor. Our simulation results show that for a 4–way processor configuration studied here (which is different from the configurations studied in [5]), 59 comparators are active in the issue queue in an average cycle – these are the issue queue slots corresponding to invalid source operands of the valid entries. Out of these, only between 1 and 2 comparators produce a match on the average. Table I shows how often the individual 2–bit groups in the comparands used in the issue queue match. This data is obtained from the simulated execution of the SPEC 2000 benchmarks.

Number of matching 2–bit groups ——>	**% of total cases**				
	0	**1**	**2**	**3**	**4**
SPEC2000 average	19.7	38.6	29	9.7	3

Table I. Percentage of partial and full matches

To collect the statistics relevant to the operation of the PLSSC of Figure 3, we divided the 8–bit comparands into four 2–bit groups (each corresponding to a P–structure from Figure 3). Specifically, bits 0 and 1 of the two comparands belong to the first group, bits 2 and 3 belong to the second group etc. Table I shows the percentage of cases when none of the groups match, only one of the groups matches, any two of the groups match, any three of the groups match and all four groups match. As these numbers show, in more than 85% of the cases at most 1 or 2 groups in the comparands match. This is why the average energy dissipation of the PLSSC is small. Additional considerations come into play when the domino comparator of Figure 2 is used within the issue queue. Specifically, the first stage compares the groups that are least likely to match to reduce energy dissipation on partial matches. In our simulations, 4 least significant bits match in only 8.4% of the cases and these bits are thus compared within the first domino stage.

We computed the average energy dissipated within the comparators used within the issue queue by recording the complete statistics of comparand's match patterns (some results were presented in Table I) and measuring the power dissipation in each case individually. For the sake of brevity, we omit the representation of these somewhat large tables here and only show the average results.

Figure 7 shows the energy dissipations for the three comparators for all possible combinations of matches and mismatches in 2–bit groups implemented by the P–blocks in Figures 2 and 3. Each case is denoted by a four–bit binary number, where the value 1 in the bit position *i* means that the two bits constituting group *i* match. The traditional comparator dissipates 582 fJ of energy on the average when used in the issue queue. Note that for traditional comparator the energy dissipation is also slightly dependent on the number of bits that mismatched, because charge has to be replenished on the diffusion capacitances of the n–transistor stacks that were conducting. The domino–style comparator, with Vs=1.8V, dissipates 206 fJ on the average in the issue queue which results in 65% energy savings in comparator power. PLSSC, which has an average dissipation of 150 fJ, performs better than the domino style comparator and results in 75% power savings. If the comparators are used in other datapath components, the energy dissipation will change because of different bit patterns in the input operands. As this paper primarily deals with the circuit–level aspects of the comparators, such studies are beyond the scope of this work.

Even under conservative assumption that comparators expend about 30% of the issue queue power for a datapath with dispatch–bound register file reads [5], the deployment of the PLSSC in the issue queue results in at least 75% comparator energy savings and an overall power savings of about 22% for the queue without lowering the Vs value (In the datapaths that employ issue–bound operand read policy, the contribution of comparator power may be well in excess of 50%, so higher power savings can be can be achieved by the use of new comparators). The extent of the total issue queue energy/power savings also depends on the optimizations that could already be implemented within the issue queue, for example, the ones suggested in [3] and [8].

6. Concluding Remarks

Traditional comparators used in several datapath artifacts of a modern processor are notoriously energy–inefficient as they dissipate energy on a mismatch in one or more bit positions. In scenarios, where matches occur relatively rarely, alternative comparator designs that dissipate energy predominantly on a full match are better alternatives. The issue queue of a modern superscalar processor is a prime example of a datapath artifact where mismatches significantly outweigh the full matches. An additional challenge in the design of the issue queues for superscalar CPUs has to do with the delay of the tag matching and steering logic, which sits on the critical path [3, 4, 6].

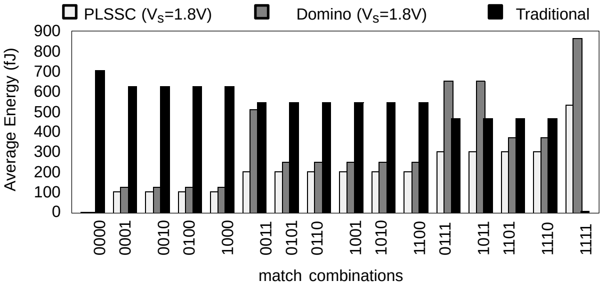

Figure 7. Energy dissipation in various matching cases

We introduced the designs of two comparators that dissipate energy primarily on a full match, including a design (the PLSSC) that has a lower overall response time than the traditional design. Assuming a pipeline cycle time of 500 ps, the traditional comparators of Figure 1 leave about 380 ps for driving the tags and possibly results and for steering in any matched data and/or setting status bits. With the use of our PLSSC, which has a delay of about 108 ps (for V_S = Vdd = 1.8V), additional time is gained for data/tag driving and steering. This significantly eases the circuit design of the issue queue. Our comparator design, particularly the PLSSC, reduces the comparator–related dissipation in an issue queue by 75%.

7. Acknowledgements

We thank anonymous reviewers for their valuable comments. This work is supported in part by DARPA through contract number FC 306020020525 under the PAC–C program, the NSF through award no. MIP 9504767 & EIA 9911099, and by IEEC at SUNY–Binghamton.

References

[1] Burger, D., and Austin, T. M., "The SimpleScalar tool set: Version 2.0", Tech. Report, Dept. of CS, Univ. of Wisconsin–Madison, June 1997 and documentation for all Simplescalar releases (through version 3.0).

[2] Brooks, D.M., Bose, P., Schuster, S.E. et al, "Power–Aware Microarchitecture: Design and Modeling Challenges for Next–Generation Microprocessors", IEEE Micro Magazine, 20(6), Nov./Dec. 2000, pp. 26–43.

[3] Folegnani, D. and Gonzalez, A., "Energy–Effective Issue Logic," in *Proc. ISCA*, 2001, pp. 230–239.

[4] Emer, J. EV8: The post–ultimate Alpha. Keynote at International Conference on PACT, Sep. 2001.

[5] Kucuk, G., Ghose, K., Ponomarev, D. and Kogge, P., "Energy–Efficient Instruction Dispatch Buffer Design for Superscalar Processors", in *Proc. ISLPED*, 2001, pp. 237–242.

[6] Palacharla, S., Jouppi, N. P. and Smith, J.E., "Quantifying the complexity of superscalar processors", Technical report CS–TR–96–1308, Dept. of CS, Univ. of Wisconsin, 1996.

[7] Ponomarev, D., Kucuk, G., Ghose, K, "AccuPower: an Accurate Power Estimation Tool for Superscalar Microprocessors", in Proc. of DATE, 2002, pp.124–129.

[8] Huang, M., Renau, J., Torrellas, J., "Energy–Efficient Hybrid Wakeup Logic", in *Proc. ISLPED*, August 2002.