

# Circuits for Low Power Bus Traffic Encoding

Spring 2003 EE 241

Semester Project

Yatish Patel, Yury Markovskiy, Victor Wen  
{yatish, yurym, vwen}@cs.berkeley.edu

May 9, 2003

## Abstract

In modern circuits the fraction of total power attributed to wires is increasing. One method to decrease the power consumption is to reduce the number of transitions on long bus wires by encoding bus traffic. The presented circuit reduces the energy consumed by buses on a single die by caching the eight most recent values on both sides of a long bus and transmitting a low weight code word on a cache hit. A custom layout in  $.13\mu$  process technology extracted and simulated indicates that our current optimized implementation offers net energy savings on buses longer than 15mm.

## 1 Introduction

Improvements in process technology are making the fabrication of smaller transistors possible. Circuit designers are using this opportunity to make larger, more complicated circuits while maintaining similar die sizes. Design performance, complexity, and power consumption begin to be increasingly dominated by on-chip wires, shifting the emphasis on optimizations away from the logic to wires. The inability of wires (particularly long wires) to scale as well as transistors results in a greater fraction of total circuit power consumed by wires.

A large number of long wires on a chip belong to buses such as the ones required for registers and memory in a microprocessor. One approach to reducing the amount of power used by these buses is to decrease the total number of transitions. The equation for power consumption in CMOS circuits reveals that a smaller activity factor, or fewer transitions, will result in lower power. One method of accomplishing this is to code the bus values so that consecutive transmitted values have fewer transitions. Another method is to simply reduce the number of times a value needs to be transferred over the bus.

We propose a simple method of caching bus values to

decrease the number of transitions on the bus. By inserting a transmitter circuit on one end of a long bus, and a receiver circuit on the other end, one could maintain a synchronized state of recently transmitted bus values. When a reoccurring value appears, a low weight code word can be sent rather than the original value. This effectively decreases the number of transitions on the bus by simply decreasing the number of times frequently appearing data needs to be sent.

In section 2 we evaluate some approaches presented by others to reduce power consumed by buses. In section 3 we present the caching circuit mentioned above. An evaluation of the circuit performance and a discussion of various improvements made to the initial design are in section 4.

## 2 Related Work

Reduction of bus power consumption is not a novel idea and has been successfully applied to off-chip communication. Off-chip buses have capacitance to bulk that is several orders of magnitude greater than the gate and interconnect capacitance on-chip. For example, compare 10pF capacitance of an off-chip wire with 0.1pF/mm of on-chip interconnect. Stan and Burleson in [4] make those exact assumptions and employ encoding to reduce the number of transitions on off-chip bus wires, which leads to energy savings. The authors propose inversion coding, which is a simple and provably optimal scheme to minimize transitions on bus lines. The encoder compares the last transmitted value with the current value, and inverts the current value before sending it if the majority of bits have changed from previous to the current word, or sends the current value unchanged.

The savings in number of transitions for such inversion coding range from 12% - 15% on a narrow 8 bit bus, and unfortunately are even lower for a wide 32 bit bus, targeted in this work. [4] does not account for any encoder/decoder

overhead because the difference between on-chip and off-chip capacitances driven is around two orders of magnitude, thus the authors sacrifice the energy consumption on the “low” capacitance side to improve the “high” capacitance side.

As technology scales to sub-micron, the cross-coupling capacitance between adjacent wires becomes increasingly important and thus encoding must become more complex and a simple reduction in the number of transitions on individual bus wires is no longer effective. Sotiriadis *et al* in [3] and Henkel *et al* in [1] have extended the work on encoding bus traffic to account for inter-wire wire capacitances. They propose extensive encoding techniques that attempt to minimize “observed” capacitance between bus wires. The difference between the measured and “observed” inter-wire capacitance has to do with the actual traffic. If the switching activity on two capacitively coupled wires is in phase, the “observed” capacitance is effectively 0, while out of phase transitions (*e.g.* wire 1 transitions  $0 \rightarrow 1$  and wire 2 the opposite  $1 \rightarrow 0$ ) create measurable (energy consuming) capacitance.

Both aforementioned works promise 40% – 50% energy savings on buses without clearly specifying the length of the bus. Sotiriadis *et al* only presents the results of the simulations. While Henkel *et al* states the existence of 400 gate implementation for their adaptive encoding scheme, the actual encoder and decoder power consumption, nor available power budget are evaluated. Hence it is the focus of this work to study the viability of bus encoding for reduction of energy consumption on on-chip buses. Rather than focusing on particular encoding algorithms and their compression bounds, we attempt to analyze and implement an actual encoder/decoder that can offer net energy savings in a practical implementation.

### 3 Transcoder

To determine a viable encoder implementation, the available energy budget was carefully analyzed. First the actual wire capacitances were obtained from layout extractions of minimum spaced minimum width wires. This results in the highest inter-wire capacitance, the best case for our purposes. Figure 1 shows the energy savings offered by two types of encodings: inversion coding [4] and a cache based encoder that will be analyzed in this work. The graph shows the average amount of energy saved per cycle (one transmitted 32 bit word) by each type of encoding for a range of wire lengths. The values were obtained by running the SPEC95 benchmarks to measure the energy saved for a microprocessor register bus.

While the results in the graph are very intuitive, it is also painfully obvious that very little energy is available for encoder/decoder circuit to break even transmitting along a short

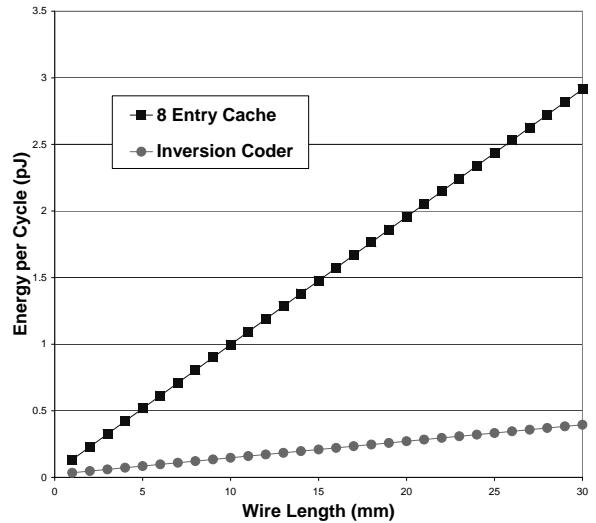


Figure 1: Energy saved on average per cycle given a length of wire using either Caching or Inversion Coding.

on-chip buses. [4] provides evidence that the effectiveness of inversion coding is diminished on wider buses. Figure 1 confirms the analysis on our 32 bit bus and leaves the cache based encoder as the practical implementation to build, since it is easier to meet the energy budget constraint. Even with our earlier mentioned capacitance extraction, which results in higher capacitance when compared to a well laid out bus, the energy budget is already overly constraining.

A high level block diagram of the Transmitter half of our design is shown in figure 2. The Receiver portion is very similar. Both sides maintain a synchronized cache of recently seen bus values. When data appears at the Transmitter it is stored in the FIFO cache. If the value is not present in the cache, the multiplexor sets the bus value to be the data value and the match control signal is not asserted. The data value is sent across the bus and inserted into the Receiver cache. When a value that is already present in the cache appears, the match signal is asserted telling the multiplexor to send the entry index of the data rather than the actual data value. The receiver end reads the match signal and uses the entry number sent across the bus to choose the data value from its own cache.

With a very tight energy budget the energy consumed by the transmitter and receiver circuit must be delicately balanced with the energy consumed by the bus. After extensive simulation of the SPEC95 benchmarks with caches varying in size from 2 to 32 entries, we settled on an 8 entry cache. Larger caches do not provide justifiable reduction in the number of transitions to warrant their use.

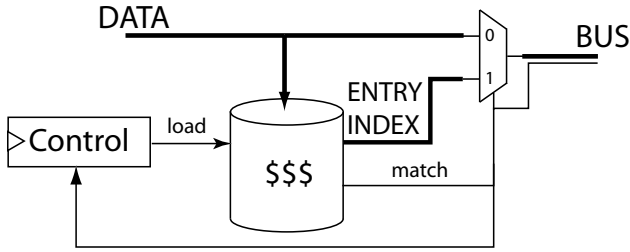


Figure 2: Transmitter Block Diagram

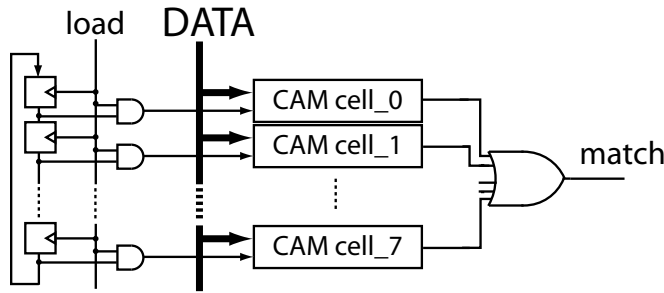


Figure 3: Cache Implementation

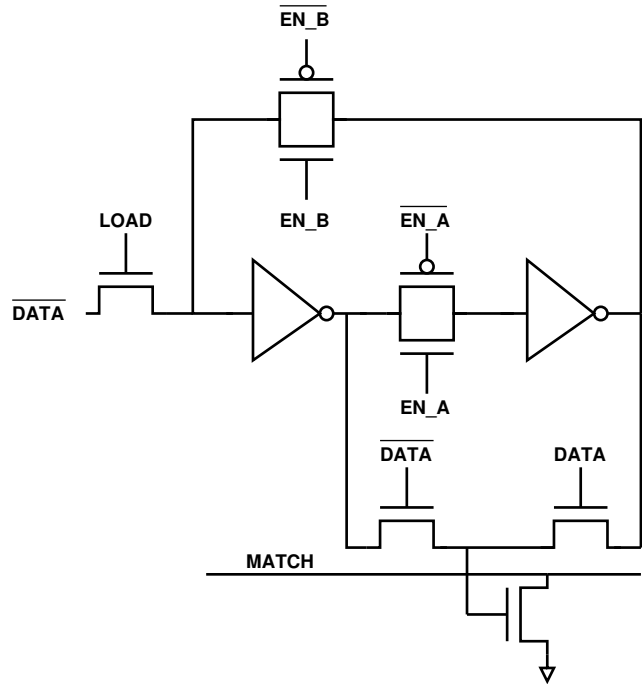


Figure 4: Original CAM Cell Schematic

Our implementation of the cache block is shown in figure 3. Eight content addressable memory (CAM) cells are controlled by a pointer bit that signifies which cell will be overwritten on the next write to the cache. The pointer bit simply rotates down the chain in a round robin fashion to keep track of the write location. The circuit uses a two phase clock with a period of  $2.7ns$ , consisting of a match evaluation phase and a cache write phase. If any of the CAM cells match the current data value, the match signal is asserted to prevent any cache writes from occurring.

**CAM circuit:** Figure 4 is the schematic of our original CAM cell. The circuit is a simple inverter chain with pass transistors used to break the feedback when a new value needs to be loaded in. Five control signals are required to operate the cell. The three transistors on the bottom are used for the matching logic.

**Match Logic:** The match line is precharged and discharged if any of the CAM cells do not match the input data. Due to the high cost of charging and discharging a long wire connected to all the CAM cells, we choose to implement a selective precharge circuit as described in [5]. The low five bits are initially charged to check for a match. Only after all five of the low bits are detected as matching does the match line of the remaining 27 bits charge up for evaluation. This reduces the probability of a full match line discharge to only  $1/32$ .

## 4 Analysis

In order to evaluate the performance of our circuits, we created a full custom layout in a  $.13\mu$  process technology. The layout was extracted using a lump capacitance model where each net has a capacitor to ground. This extraction method was chosen over a more detailed distributed resistor-capacitor model due to the reduction in simulation time. A couple circuits were extracted and simulated with both extraction methods. The distributed resistor-capacitor model consistently resulted in lower energy numbers than the lump capacitor model. Since we used the lump-capacitor model, all the numbers presented in this paper will be on the conservative side.

The extracted netlist was modified to output the energy used by different functions of the circuit and then simulated using SPICE. The input values used were obtained from a 50 cycle trace of the SPEC95 gcc benchmark of the register bus.

### 4.1 Control Lines

As shown in the first column of figure 7 the initial circuit labeled *Five Clock* had a large portion of its total energy consumption going to control. In this case, control is defined as the energy used by the logic that generates the various enable signals ( $EN\_A, EN\_B, \dots$ ) and the wires associated with

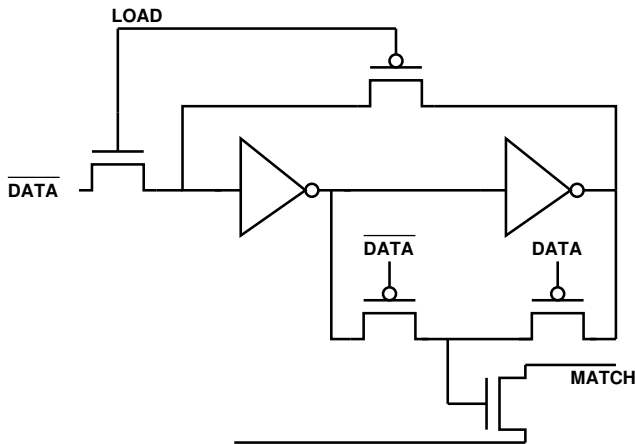


Figure 5: Improved CAM Cell Schematic

them. Since the layout required five long wires with high capacitance to be switched every cycle, the control became a good target for potential energy reduction. A new CAM cell circuit, figure 5, was created to decrease the number of control lines. Rather than requiring two separate pass gates to disable the inverter feedback, only one path truly needed to be disabled to load in a new value. In order to disable the feedback path, the pass gate was changed to just a single PMOS transistor which allows us to use the same **LOAD** signal for the NMOS input gate and the feedback gate. Careful simulation was conducted to verify the inverter would correctly detect the level of the PMOS feedback. The second column in figure 7 clearly shows that the new cell resulted in significantly less energy going to control.

## 4.2 Match Circuitry

After the new CAM cell was inserted into the design the matching logic started to play a bigger role in the energy breakdown. The match logic was designed as a dynamic OR gate where if any cell does not match, the match line is discharged. Due to the fact that a data value can only be cached in one CAM cell at any given time, when a match occurs, seven of the eight CAM cells will charge and then discharge their respective match lines. This behavior results in a tremendous amount of energy wasted to conduct matching.

An alternative approach to detect if a match exists is to use a series pull down rather than a parallel pull down, similar to what is described in [2]. In a series pull down, the match line will only discharge when a match is detected allowing it to stay charged in the more prevalent case of no match. Unfortunately, a 32 transistor stack used as the pull down

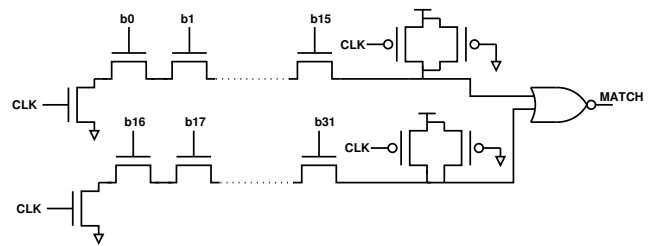


Figure 6: New Match Logic

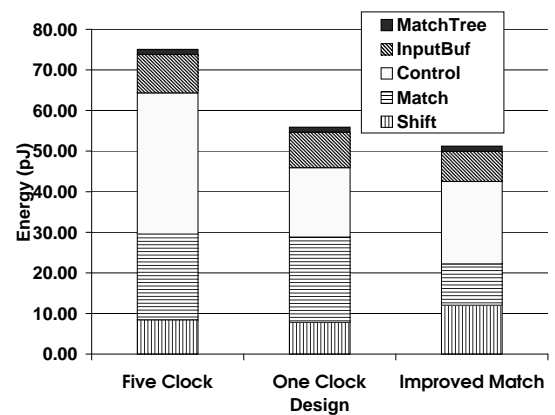


Figure 7: Energy breakdown by circuit design for a 50 cycle trace. The first column represents the original design, the second column is the design with a smaller CAM cell and fewer clock lines, and the the third column is the smaller CAM cell coupled with improved match logic. Each column is broken down energy consumed for each function

in dynamic logic will result in a significant charge sharing problem. After careful simulation of various configurations of stacks and bleeder transistors, we settled on a design that uses two stacks of sixteen transistors with a bleeder on each output. In order to make sure the NMOS pull down transistors are completely on, the two NMOS match logic transistors in the CAM cell were replaced with PMOS transistors.

Those outputs are then NOR'd to generate the match signal as shown in figure 6. The third column in figure 7 shows the decrease in energy used by matching. This improvement, however, comes at a cost of a slightly increased clock period for the circuit resulting in an increase from 2.7ns to 3.0ns.

## 5 Conclusion

This work provides a detailed energy consumption analysis of a fully functional cache based bus traffic encoder. We have identified two dominant circuit components, long clock/control wires and match logic, that consume the majority of energy and attempted to aggressively optimize them. The results, however, can be considered a successful failure. While with each optimization the improvement in overall circuit power consumption was demonstrated, the encoder circuit still exceeds the power budget to be used in a practical design, because the net energy savings will only be observed for interconnect buses of length longer than 15mm.

The focus of this work was reduction of energy consumed by on-chip buses through algorithmic means (coding), not by introducing modifications to devices or bus topology. While energy savings can be increased through shielding and spacing of bus wires, these will only reduce the effective bus capacitance, further hindering our ability to fit encoder circuit in already constrained energy budget. The evidence from this work suggests that traffic encoding is not yet a practical technique for energy saving on on-chip buses.

However, it is important to look to the future and carefully analyze whether scaling down to sub-micron technologies will result in the same discrepancy between the capacitance of on-chip buses and logic as was observed by Stan and Burleson in their work with off-chip buses. These authors justified ignoring the encoder energy overhead based on the order of magnitude gap in capacitance that exists between on-chip and off-chip loads.

## References

- [1] Jrg Henkel and Haris Lekatsas. A<sup>2</sup>BC: adaptive address bus coding for low power deep sub-micron designs. In *Proceedings of the 38th conference on Design automation*, pages 744–749. ACM Press, 2001.
- [2] Kun-Jin Lin and Cheng-Wen Wu. A Low-Power CAM Design for LZ Data Compression. *IEEE Transactions on Computers*, 49(10):1139–1145, October 2000.
- [3] P. P. Sotiriadis and A. Chandrakasan. Low power bus coding techniques considering inter-wire capacitances. In *Proceedings of the IEEE Custom Integrated Circuits Conference*, pages 507–510, 2000.
- [4] M.R. Stan and W.P. Burleson. Bus-invert coding for low-power I/O. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 3(1):49–58, March 1995.
- [5] C. Zukowski and S. Wang. Use of Selective Precharge for Low-power Content-addressable Memories. In *Proceedings of the IEEE Int'l Symp. on Circuits and Systems*, 1997.