Computer Architecture Reading Group Notes

Date: 1/22/04
Discussion Leader: Francois
Notes: Mattan
Topic: Power
Papers:

1. A.P. Chandrakasan, S. Sheng, and R.W. Brodersen. Low-power CMOS digital design. *IEEE Journal of Solid-State Circuits*, Vol.: 27, Issue: 4, April 1992. Pages:473 – 484.
2. Dan Ernst, Nam Sung Kim, Shidhartha Das, Sanjay Pant, Toan Pham, Rajeev Rao, Conrad Ziesler, David Blaauw, Todd Austin, Trevor Mudge, and Krisztián Flautner, "Razor: A Low-Power Pipeline Based on Circuit-Level Timing Speculation," in the *36th Annual International Symposium on Microarchitecture (MICRO-36)*, December 2003

Administrative

Vicky will take notes and choose a topic next week.

Low-power CMOS digital design

Summary:
A seminal paper on low power design (the most sited in JSSC). It discusses voltage scaling as a general low power circuit technique, analyzes several circuit families, and finds NMOS pass-gate to be the most efficient one. The paper continues by making technology independent predictions, and suggests a framework and architecture for low-voltage low-power circuits that maintain a required performance level.

Discussion:

1. This is paper was published in 1992 and although velocity saturation is mentioned, it is not discussed much.
2. Transistor sizing is key in achieving low power design.
3. Suggests that voltage and frequency can be scaled down, and performance met by taking advantage of parallelism (parallel circuits, or pipelining).
4. Only considers pipelining penalty only due to extra capacitance of latches, doesn't mention the fact that some circuits cannot be broken down into equal pipeline stages. Also doesn't mention latch delays.
5. Intro seems completely valid today - didn't we change anything in computers in the last 10 years?
   This paper was part of Berkeley's InfoPad project, which was a wireless, mobile, internet device. The concept is still what is driving applications today, the integration level and size are where most of the changes happened.
   The paper dealt with low-level circuits so is relevant today as well.
6. doesn't talk about how to calculate the extra capacitance of parallelism, but they did evaluate what happens in extreme overheads ($\sim N^2$ and $\sim N^3$)

7. Q: do we need low-swing buses after we drop V_{dd} as much as possible?
8. Pass gate turned out best but how does this look today when leakage current is more critical. Mark Horowitz claims that you can always control leakage and limit it to ~15% of the power. For example, by using low V_t transistors most places and high V_t where it's critical.
9. Did they consider leakage when putting in parallelism?
10. The paper doesn't talk much about asynchronous circuits - maybe more important today where so much power goes into the clock.
    Amulet from Manchester didn't seem to achieve much lower power than an industry low-power ARM. Perhaps it was because it was too simple a processor and clock power was not an issue.
    GALS (Globally Asynchronous Locally Synchronous) can also be used to reduce clock power. Industry leaning that way, and kind of doing it anyways because you need to maintain some clocks even when in low-power mode (or sleep mode) or clock gating. Also have different domains because of memory controller on-chip and other system components.

Razor: **A Low-Power Pipeline Based on Circuit-Level Timing Speculation**

Summary:
Dynamic voltage scaling (DVS) is used today but with large margins to account for fab and operating environment variations. In this paper they want to scale the voltage down to no margin and recover from the timing errors introduced. The way they do this is by detecting errors using a delayed flip-flop and compare to the normal flip-flop. Basically they increase the timing margin by introducing the shadow latch. A control loop adjusts the voltage to maintain a non-zero but low error rate.

Discussion:

1. Meta-stable states are a problem. They introduce a circuit to deal with meta-stable states. Two skewed invertors and check to see whether they sense a difference (still won't catch all occurrences). So you add a set of "double-latches" to detect that, but somehow they do it in one cycle.
2. At the arch level you don't put these flip-flops everywhere - only in timing critical sections.
3. They implemented an Alpha with these techniques but haven't shown results yet.
4. they can tolerate about 2% errors and get a 40% reduction in energy.
5. This scheme makes timing verification a lot more difficult.
6. They basically allow for timing margin modification, but the general rule is to make all paths equal so you shouldn't need this variability.
7. The delayed clock is .5 cycles (clock-bar), not clear how to do anything else.
8. The control loop is not well described, and not clear if this will actually work. Is the measurement still valid by the time the control is applied.
9. Can you use the same technique to over-clock?
   The real question is how low can we drop the voltage for a predetermined amount of performance (frequency).
10. Will the unpredictable performance due to errors impact real-time systems?

11. This was not compared to lowering the frequency a little bit and reducing the voltage.
12. Didn't talk about dual-voltage designs, where the critical paths are run at a higher voltage than everything else.
13. How does the counter-flow pipeline recovery actually work, and is it better than the first method?
    It's not better, it's required when you can't have a global signal to gate the clock. CF eliminates the global control.
    They also introduced an extra pipeline stage or two before modifying architectural state.
14. It seems like you wouldn't be able to make all flip-flops Razor flip-flops because the power overhead will outweigh the energy reduction benefit.
15. Not clear how this applies to deeply pipelined architectures (higher overhead for Razor).
16. They suggest Razor for sense-amps. How will these work? Will there still be a single cycle latency?
    What happens when you need to pre-charge the bit-lines? Is this equivalent to just increasing the clock cycle because you can't start the pre-charge until you made sure there is no Razor error?
    Can this be solved by interleaving banks? If the access pattern allows it then probably yes.
    Also, is there any slack in the SRAM timing to take advantage of?