Summary Part 2

Drowsy Caches: Simple Techniques for Reducing Leakage Power

Mrinal Nath (ID: 3307043)

March 30, 2005

Did this paper address an important issue? Explain.

Now-a-days, on-chip caches (L1, L2) account for the majority of the transistors on a processor chip. Caches have a good property that they do not consume much *dynamic* power, since most of the switching activity tends to be localized. However, as technology scales to smaller dimensions, the *leakage* or *static* power dissipation of a transistor is becoming a significant fraction of the total power consumed by it. Hence, leakage power dissipation in caches is a major fraction of the total power consumption in latest and future generation processors (15% to 25%). This paper proposes techniques by which the leakage power of caches could be reduced. Thus it deals with an important issue.

Are the proposed approaches valid? Describe its strength and weakness.

The basic technique that the authors propose is nothing but DVS. DVS is widely used and well understood. Only, DVS is currently used to reduce *dynamic* power dissipation, while the authors propose to use DVS to reduce *static* power consumption. Hence, the approach is valid.

Strengths:

- The technique is simple and has low area overhead (7.35 memory cells per cache line)
- The main advantage is that the cache lines retain their data even when they are in the drowsy state
- The performance overhead is small (one cycle additional latency) if only the data lines are kept in drowsy state. Hence the entire L2 cache can be kept in drowsy state permanently.
- The *simple* policy can be quite effective. Hence there is no need for having any complex mechanism to decide which cache lines should be put in drowsy state
- This technique is simpler and more efficient than the ABB-MTCMOS technique, since it does not require high V_{DD} values
- This technique reduces the static energy consumption to values close to the theoretical minimum (zero leakage)
- The drowsy bit is found to be stable (does not flip) in the presence of noise and cross-talk due to its neighboring awake bits.

Weaknesses:

- This technique is found to work well for data caches, but not for instruction caches
- A single window size may not work for all types of programs, so there will be a need to determine the window size dynamically. This could add some complexity.

Do the results support the conclusions? Explain.

Their main conclusion is that this technique is very effective in reducing the leakage power consumption. The results definitely support this conclusion, as a large reduction in static power consumption is

observed. Moreover, the performance penalty is acceptably small (around 6%). They also conclude that the *simple* scheme is good enough to achieve sufficient leakage power savings. This is true, since the more complicated scheme does not provide much benefit. Also, though this technique does *not* offer the lowest leakage per bit, it has the advantage of maintaining the stored data. This allows a more aggressive approach while putting the cache lines into drowsy state and thus provides greater overall leakage power reduction.

Describe the potential future works?

- How to dynamically decide the window size which will allow the maximum power savings?
- What are the changes/additions required to make this approach work well for I-caches also?
- There are several other caches in the processor (e.g. branch predictors, TLBs, etc.). Can this technique be applied to such caches? Are any changes required to adapt this technique to such caches?
- This technique is mainly applicable to SRAM cells. Can this be adapted to DRAM cells so that the main memory can be put into drowsy mode? This could potentially save a lot of power, especially in embedded systems, where memory consumes a considerable amount of power.
- L1 I-caches are replaced by trace caches (for e.g. in Pentium4). Does this technique work for trace caches?