Critiquer#2: Stephen Orchowski
Paper: “Power Savings in Embedded Processors through Decode Filter Cache”
Presenter: Fei Hong, presented on November 11, 2008

This paper proposes a new cache mechanism for handling instruction fetches outside of a conventional power-hungry cache. The decode cache substitutes fetches to a regular cache so that the actual fetch and subsequent decode is avoided. A predictor mechanism is also used to further improve the likelihood of successfully accessing the “Decode Fetch Cache”.

Strengths of Paper:

If the researchers’ claims hold true, the cache mechanism reduces power consumption by 34% on average for the processor without degrading overall application performance by more than 1%. This is important in embedded environments where power is at a premium and thus such a mechanism shows promising results for many embedded applications and systems.

There is clear evidence that the researchers have test the mechanism in a simulated environment and have produced meaningful results. There is plenty of real data to analyze.

Weaknesses of Paper:

The paper itself is somewhat short compared to that which we have become accustomed. The main points concerning background work, mechanism description, experimental setup and results are all present. However, the paper lacks details on many of these things. Thus, the reader is left wondering about many of the finer points of the cache mechanism details and motivations behind the design. For example, the paper does not detail exactly how profiling was accomplished on the instruction set of the modeled processor (see section 3.2). It states “execution frequencies of all instructions are obtained from a set of benchmarks”, but it does not explicitly elaborate on this. Were the benchmarks, the same benchmarks used to test the mechanism? The paper is full of rather large leaps like this.

No consideration is given to possible implementation details of adding such a mechanism to the processor pipeline and how this might affect overall cycle times.

No summary is given to characterize the reasons why particular applications tested offer a performance improvement. The paper does not offer why one application improves with the mechanism and another does not. There is a raw presentation of the benchmark results, but little analysis to show why applications run better or not.

The paper does not consider applications or at least identify applications which might be more dependent on the set of “uncacheable” instructions.