Very Long Instruction Words (VLIW)

6.911 Architectures Anonymous

Aaron Adler

Very Long Instruction Words (VLIW)

- Fisher (HP)
  - Trace Scheduling
    - splits and rejoins
  - ELI-512
    - 500+ bit instruction

VLIW

- Rau (HP):
  - Dynamic Scheduling
    - two phases
    - hardware support
    - paper shows VLIW processors capable of out-of-order instructions and multiple instructions

VLIW - Crusoe

- Klaiber (Transmeta)
- Lots of hype
  - two versions
  - hardware software line
  - system architecture
  - “atoms”
  - “modules”
  - instruction length

Crusoe - CodeMorphing

- Starts conservatively
- Monitors usage patterns
- Caches translations
- Gated store buffer

Crusoe

- Power Management
  - can vary clock speed and voltage
  - changes not noticeable to user
- Benchmarks
  - need to be redone?
  - only translates code once

Comparisons

- Crusoe vs. other VLIW
  - no need to recompile because hardware details not exposed
  - can fix bugs
- VLIW vs. Vector machines
  - more tests required

Crusoe

- Alias hardware
- Performance sacrifice for translation
- Still too early to tell?

Questions

- The benefit of VLIW techniques is *simplicity* due to the compiler-driven static scheduling. however, there seem to be hard limits on how much single-thread parallelism can be extracted statically. HP/Intel's Merced attempts to reintroduce some limited dynamic scheduling to the core, in more or less the manner of their previous superscalar designs. are there other ideas on how to reintroduce dynamism w/o adding overmuch complexity -- a merge of multithreading techniques, say, or Crusoe-like JITs which can adapt to dynamic parallelism opportunities, or something completely different?

Questions

- Crusoe's translation cache strikes me as key to the success of the design-- instruction sequences can be translated just once rather than every time they are executed. How good is Crusoe at preventing cache pollution caused by translating code that ends up not being reused much? Is there a bailout condition at which point translation is abandoned in favor of just interpreting? I'm thinking of the dynamic translation talk one of the HP folks had recently in NE43. How bad do things get in this worst case?

Questions

- VLIW strongly resembles SIMD, in that operations are performed synchronously and in the same context; the difference is that different processing units perform different operations. This leads to a comparative question: are there fine-grained VLIW systems which operate on bit-wide (or few-bit-wide) data pipelines (compare Abacus)? What are they like?

Questions

- Since the translation software for Crusoe processors does not have to change until the VLIW machine itself does, why wouldn't Transmeta put that functionality in hardware?

- People have been saying VLIW is the coming great thing for a long time now... have we gotten to point, with Crusoe et al, where it can actually succeed, or is it just going to fall prey to the same problems as always?

Questions

- What's the current state of the art in VLIW compilers? Who makes them (commercial and research), are any available online, and how well do they work?

- Transmeta's Crusoe has been the buzz of the industry, but it seems that they are trying to cover up a lot of the performance problems of their processors that use their "codemorphing" technique with explanations of why standard benchmarks are not good. Is this indicative of any weaknesses of their dynamic translation approach?

Questions

- What's the deal with Sun's MAJC architecture? Is it serious or another picoJava? How tied is it to the Java language/JVM and the Java threading model?

- How prevalent is trace scheduling nowadays? Does Transmeta's code morphing software implement some form of trace scheduling or does it only look as far ahead as a single translation?

Questions

- What are these condition codes they speak of in the Crusoe translation example?

- A bunch of companies are trying VLIW these days (Transmeta, Intel, etc.), but they all seem to be getting dismal performance. Any clues as to why so many people are failing at VLIW?

- Is there any way to bypass the code-morphing software and write VLIW code that runs directly on the Crusoe?