ReRISC

Reconfigurable Reduced Instruction Set Computer

Andrew S. Huang and Edward H. Kim
ReRISC Motivation

• General observations
  – specialized hardware yields greater performance
    • ASICs, DSPs
  – generalized hardware is more versatile, less risky, and often easier to use
    • FPGAs, microprocessors
  – microprocessor architecture trends
    • more specialized hardware and instructions (MMX, Sparc gfx extensions)
    • more reliance upon compilers to extract parallelism and manage data dependencies
  – CMOS technology trends
    • wire-delay dominated
    • mega-gate level densities
      – difficult to verify and test
      – favors array architectures with redundancy capabilities
ReRISC Solution

• “Performance where you need it, convenience when you want it”
  – Convenience of a conventional, single-threaded instruction-based computational model
    • for those days when you’re just writing UI code or on-line help
  – Performance of specialized hardware, configurable as a systolic array, an in-place computation, a vector processor, or a single complex operator
    • for those days when you just can’t seem to get real-time performance out of your graphics engine or database engine

• Reconfigurable instruction set
  – default instruction set of a simple RISC microprocessor
  – user can augment or replace the instruction set with new instruction set configurations (ISCs) for application specific performance
  – fast context switching
    • allows for practical integration into standard workstation environment

• Physical design in arrays
  – allows for easier verification and post-fabrication redundancy
ReRISC Programming Model

- ReRISC machine state
  - registers (r31=0)
    - 36-bit registers for tagged datatypes
  - ISC
    - Instruction set configuration
    - 8 38-bit computational elements, e0-e7, each with
    - 8 contexts, c0-c7, switchable on a cycle by cycle basis
  - Opcode mapping
    - one map per opcode, o0-o63
    - controls timing and activation of elements and contexts
  - NOR Plane
    - 16 contexts, x0-x15, switchable on a cycle by cycle basis plus hardwired control
ReRISC Hardware Architecture

- Based on MIT Beta architecture
- Tagged datatypes in hardware
- 3R/1W register file
- 32x32 NOR Plane with hardwired control and masking
  - hardwire control lets NOR plane function as rotator or barrel shifter
  - mask allows for fast assembly of bit-field data structures, enhances performance as an emulation machine
- Computational array
  - another slide
- IPR - In-Place computation register
  - another slide
NOR Plane

- 1/2 PLA NOR Plane
  - used for complex bit-twiddles and decoding applications
  - dual mode operation
    - context RAM driven
      - decoding
      - permutations (DES)
      - complex condition testing
    - shift code driven
      - barrel shifter (RC-5, multiply and divide by powers of 2)
      - bit field packing and unpacking (fast emulation of non-native binaries)
      - packed rotates/shifts (multiple byte or word wide ops shifted per cycle, from the MMX ISA)
  - post-masking device
    - simplifies shift code decoder design significantly
• Insert adobe illustrator slides here
In-Place Computation Register

- “In-place” computation is a term borrowed from the DSP world
  - FFTs are in-place computations
  - Output of one stage is the input to an identical stage
- Multiplies are in-place computations
  - Implement 32x32 multiply in a 38x8 array of processing elements
- IPR allows for storage of intermediate results in in-place computations without disturbing the register file contents
- IPR plus context switches allows for multi-function instructions