Simulating the Tensilica-based SM Design

- Processor simulation model is made automatically
  - By unmodified Tensilica tools
- Memory models are being added by us
  - Using XTMP API
- Overall framework is flexible and modular:
  - Using Tensilica tools
  - Fully equipped tiles
  - Sophisticated quad interface
  - Small version of a potential Smart Memories chip

Overall Design

- Array of reconfigurable processing and memory tiles
- Thread-parallel applications
  - Use multiple cores and multiple contexts
  - Use cache coherence, fast interconnect, fast synchronization
- Data-parallel and streaming applications
  - Use multiple cores in SPMD manner
  - Use HW FIFOs for streaming buffers
  - Use DMA capabilities for fast data streaming
- Control-dominated applications use thread-level speculation (TLS)

Project Motivation

- VLSI technology scaling is driving changes:
  - Computation is getting cheaper
  - Designs are getting more complex
  - Design costs are increasing
  - Gate and wire delay balance is changing
- Current general purpose architectures are not sustainable:
  - Communication speed is not scaling
  - Poor modularity

Adding Hardware Support for TLS

- Working to build a silicon prototype with Tensilica cores
- Small version of a potential Smart Memories chip
  - 16 tiles in four quads, four tiles per quad
  - Two off-chip memory I/O controllers
  - Small, fully-connected global network
- Sophisticated quad interface
  - Cache controller / DMA engine to handle misses and refills in the quad
  - Set of 4 dedicated 32-bit buses to other tiles in quad
  - Global network access interface
- Fully equipped tiles
  - Two Tensilica cores per tile
    - Our fast synchronization instructions added to accelerate multithreaded execution
    - 16 memory mats, 1024 x 32b+5b (data + control) words
    - FIFO mode and DMA FIFOs for streaming execution
    - Control bits can be used for multithreaded or transactional execution
    - Crossbar interconnect

Current Simulator Status

- Developing Tensilica-based simulation environment
  - Steps #1 and #2 complete
  - Cache controllers for Step #3 are in progress
  - Will continue with Step #4 when simulation works
- Thread-level speculation (TLS) works on the ideal memory system
- Ported runtime environments for multithreaded operation
  - Running kernels and benchmarks from SPLASH-2 using ANL macros
  - TLS runtime is complete and working
- Plan to develop a transactionally coherent runtime soon

Current Hardware Status

- Designing memory system control support for:
  - Cache coherence
  - Special synchronization operations
  - Special memory modes like TLS
- Designing tile and quad Verilog models
  - Uses Tensilica-produced processor core model
  - Adds some TIE synchronization instructions right to the core
  - Wraps our memory model around the cores
  - Including SPLASH-2 and interconnect support for TLS
- Working with Reshape tools to automate backend tool flow

The Stanford Smart Memories Project: Hardware

M. Horowitz, D. Engler, P. Hanrahan, C. Kozyrakis, K. Olukotun, M. Rosenblum