Building and Using the ATLAS Transactional Memory System

Njuguna Njoroge, Sewook Wee, Jared Casper, Justin Burdick, Yuriy Teslyar,
Christos Kozyrakis, Kunle Olukotun

Computer Systems Laboratory
Stanford University
http://tcc.stanford.edu

Today's Agenda

- ATLAS Overview
- ATLAS Status and Roadmap
- Experience of Building ATLAS
- Results and evaluation
- Conclusions

ATLAS Overview

- ATLAS is an UMA implementation of TCC
  - TCC = Transactional Coherence and Consistency
  - Shared memory with continuous transactions

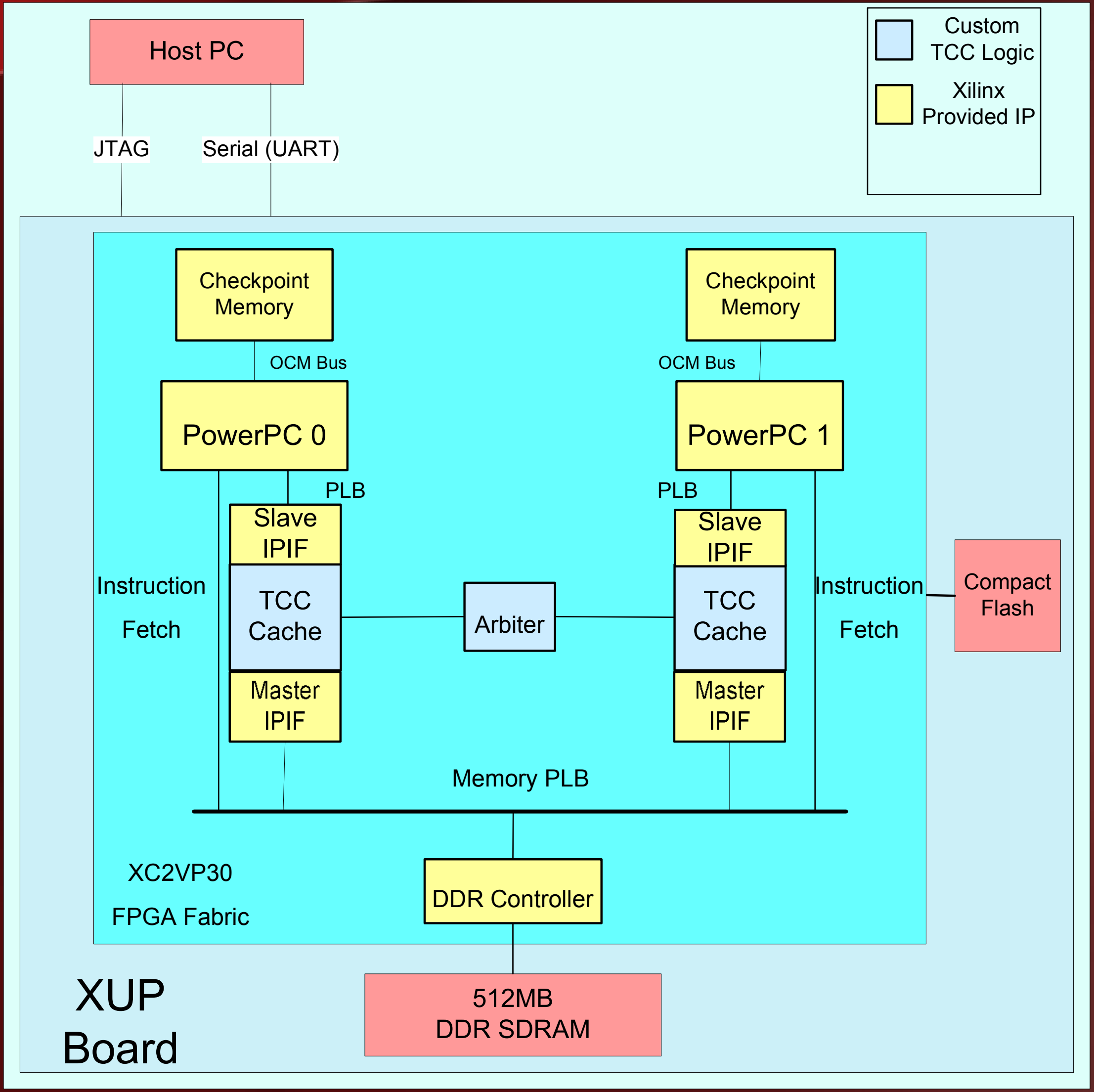
- ATLAS' objectives
  - Provide a fast platform for software development
    - For user applications and system software
    - Direct transactions support, tuning & debugging tools
  - Provide reasonable performance accuracy
    - Compared to ASIC designs or detailed simulation
  - Use commodity FPGA HW/SW for rapid design
    - A tool for research, not a final project demo
  - Not a goal: highest possible GOPS/GFLOPS

ATLAS Status and Roadmap

- ATLAS' status
  - Implemented on XUP board with XC2VP30 FPGA
  - 2-CPU TCC system at 100 MHz
    - Using the built-in PowerPC 405 cores
  - Rich debugging, profiling & tuning environment

- Next → ATLAS on BEE2 board (RAMP-Red)
  - 10x more LUTs/BRAMs than XUP board
  - Allows for 8-CPU TCC system on the 4 user FPGAs
  - DRAM, interconnect, Linux I/O on control FPGA

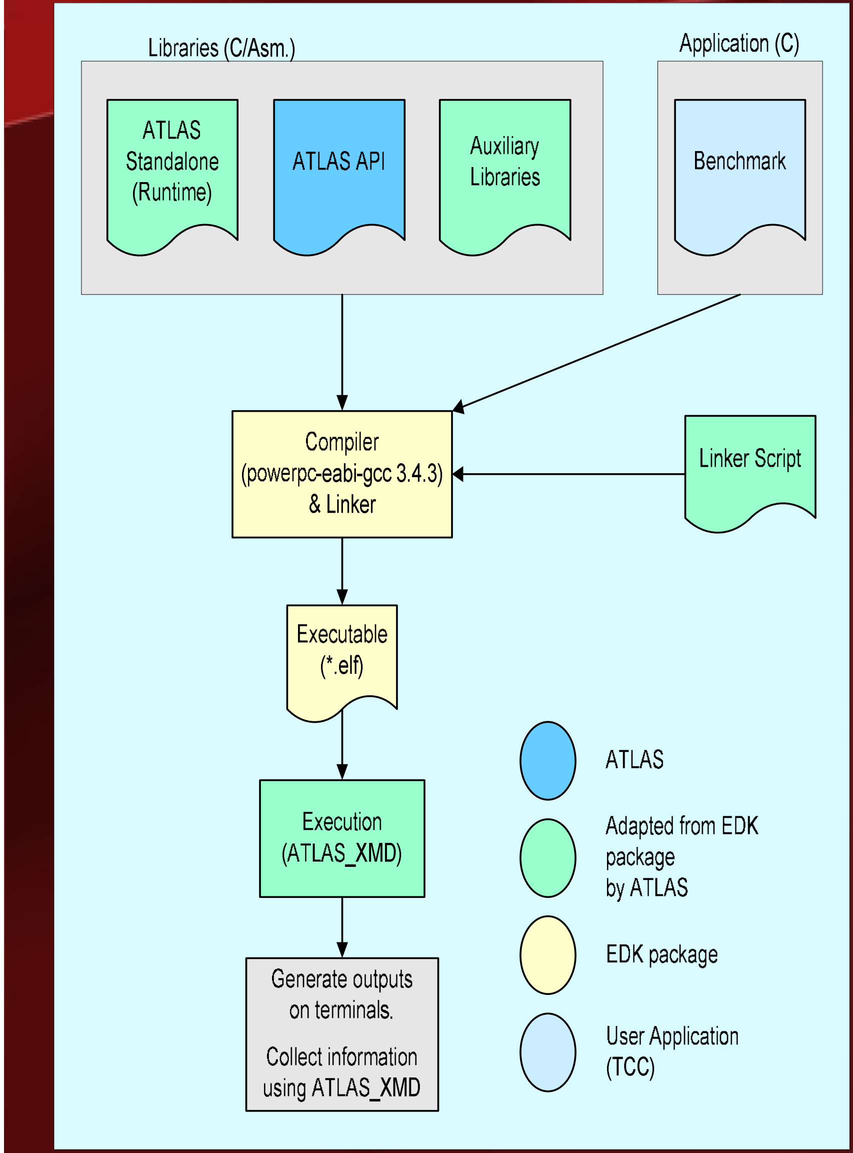
2-way ATLAS Hardware Platform



HW Highlights

- 100 MHz CPU & bus
  - Internal PPC I-Cache on
  - Internal PPC D-caches off
- Transactional cache
  - 8KB DM or 16KB 2-way or 32KB 4-way cache
  - 32B lines
  - 2KB or 4KB or 8KB write address FIFO
- Main memory
  - 512 MB DDR SDRAM
- I/O
  - UART for each PPC
    - PPC0: RS232
    - PPC1: JTAG UART
  - File I/O: Compact Flash
- See [PACT'05] for architectural model

2-way ATLAS Software Platform


Libraries (C/Asm.)
Application (C)
ATLAS Standalone (Runtime)
ATLAS API
Auxiliary Libraries
Benchmark
Compiler (powerpc-eabi-gcc 3.4.3) & Linker
Linker Script
Executable (*.elf)
Execution (ATLAS_XMD)
Generate outputs on terminals.
Collect information using ATLAS_XMD
ATLAS
Adapted from EDK package by ATLAS
EDK package
User Application (TCC)

SW Highlights

– TCC API for parallel programming
- Written in assembly for speed
- See [ASPLOS'04] for prog. model

– Robust debugging infrastructure
- Xilinx's Microprocessor Debugger (XMD)
  - –JTAG port access to PPC debugging ports
  - –GDB stub on-top of XMD
- Extended XMD → ATLAS XMD

– Rich support for intuitive performance profiling & tuning
- Integrated into the API
- See [ICS'05] for tuning process

Come watch our demo!

Experience of using Commodity HW and SW

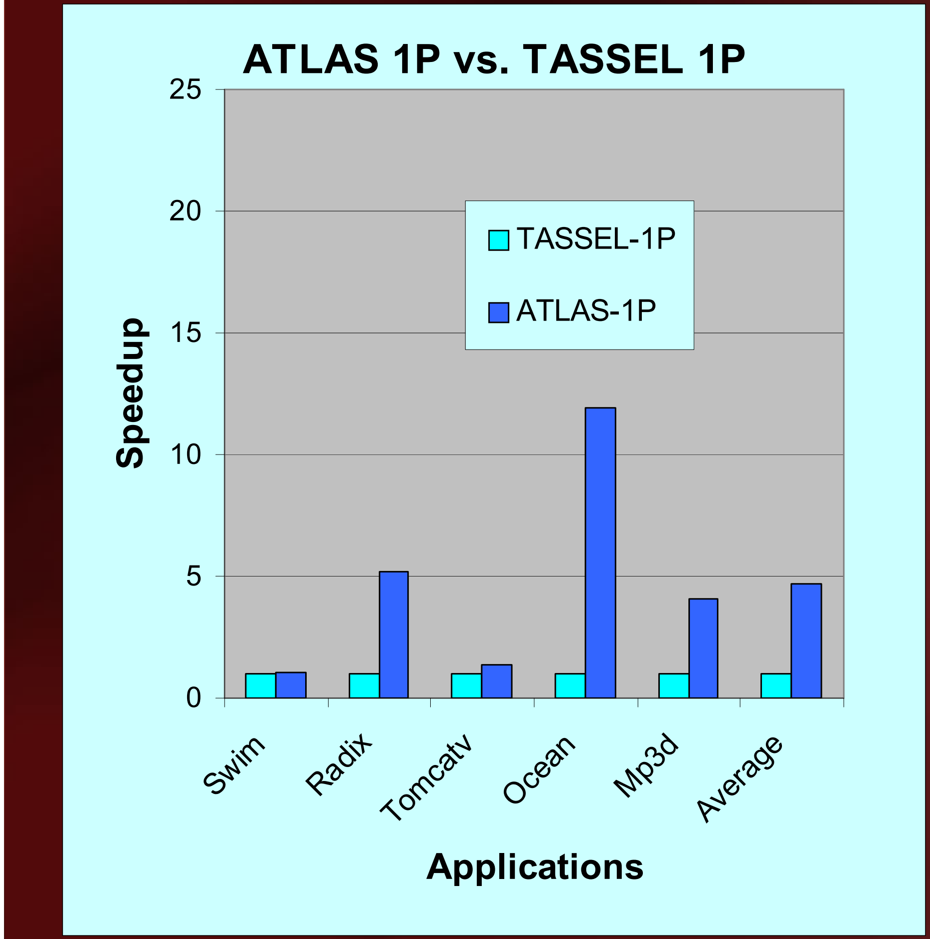
- Tools and Documentation (EDK)
  - Examples & GUI-wizards assume 1-CPU systems
  - ATLAS stresses scarcely documented features
- Provided IP and SW libraries
  - Convenient but often slow or missing functionality
    - PLB DDR can't run below 100 MHz
    - I/O from CF card is too slow
    - Had to implement syscalls from scratch
- Challenging coding API in assembly
  - API tethered to EDK's gcc, which lags latest version

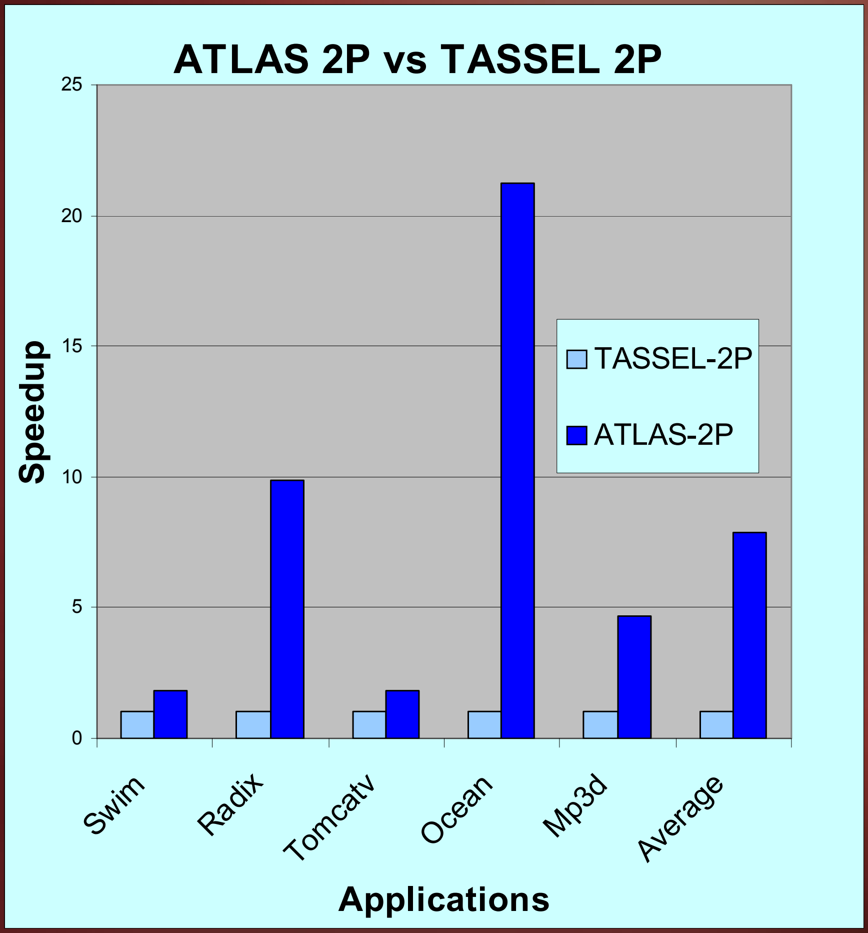
Hardcore vs. Softcore Processor

- Cons
  - Cannot modify CPU internal datapath/cache
    - 10 cycles for TCC cache hit
  - No internal FPU – no interface for external FPU
    - FP operations are emulated
  - Maximum 2 processors per FPGA
- Pros
  - Same ISA with our software simulator
  - Can run full software frameworks
    - PowerPC Linux, PowerPC Jikes RVM
  - Observed similar speedup trends with simulator
    - Despite stalls on cache hits

So how does ATLAS perform?

- Wall-clock time : ATLAS vs. TASSEL (TCC Simulator)
  - –Atlas-1P is ~5x faster Tassel-1P
  - –Atlas-2P is ~8x faster Tassel-2P





- TASSEL runs on a 2.5GHz Apple G5 workstation

Discussion of Results

- TASSEL uses fast-forwarding
  - Significant sections of application skipped
    - Explains small ATLAS gains on swim, tomcatv, mp3d
  - But programmer must be very careful
    - May miss a critical section → meaningless speedups
  - TASSEL does not require such tradeoffs

- FPU emulation is a major bottleneck
  - Radix: 90% to gen FP data, 10% integer sorting
  - ATLAS-2P: 75x speedup in sorting, 22x overall

- Scalability
  - TASSEL gets slower with more processors
  - ATLAS scales with number of FPGAs

Summary of Experience: FPGAs are promising, but...

- CMP research targets 8 to 16 CPUs
  - Desire to scale ATLAS to ≥8 processors
  - XUP boards insufficient for the task
    - Limited to ring topology: high latency, limited bandwidth
    - XC2VP30 FPGA has limited LUT/BRAM resources
  - Need a better platform → BEE2

- Diagnosis:
  - Commodity boards and tools need to mature for CMP research