

RAMP Gold Wrap

Krste Asanovic

RAMP Wrap

Stanford, CA

August 25, 2010

Graduate Students

- Zhangxi Tan
- Andrew Waterman
- Rimas Avizienis
- Yunsup Lee
- Henry Cook
- Sarah Bird

Faculty

- Krste Asanovic
- David Patterson

- Bigger, more complex target system
 - Many cores, more components, cache coherence, ...
 - Need to run OS scheduler, or multithreaded run time
- Sequential software simulator performance no longer scaling
 - More cores, not faster cores
- Detailed software simulators don't parallelize well
 - Cycle-by-cycle synchronization kills parallel performance
- Parallel code has non-deterministic performance
 - Need multiple runs to get error bars
- Software more dynamic, adaptive, autotuned, JIT, ...
 - Can't use sampling
- ~No legacy parallel software
 - Need to write/port entire stack.



RAMP Blue, July 2007



- 1,008 modified MicroBlaze cores
- FPU (64-bit)
- RTL directly mapped to FPGA
- 90MHz
- Runs UPC version of NAS parallel benchmarks.
- Message-passing cluster
- No MMU
- Requires lots of hardware
 - 21 BEE2 boards / 84 FPGAs
- Difficult to modify
- FPGA computer, not a simulator!

Direct: One target cycle executed in one FPGA host cycle

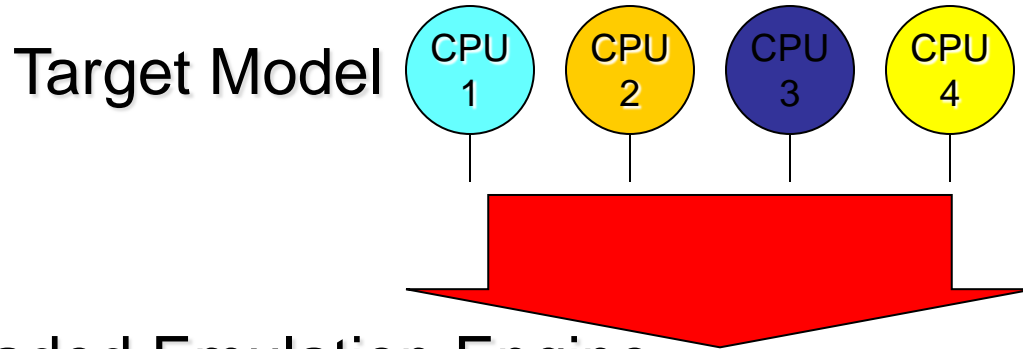
Decoupled: One target cycle takes one or more FPGA cycles

Full RTL: Complete RTL of target machine modeled

Abstract RTL: Partial/simplified RTL, split functional/timing

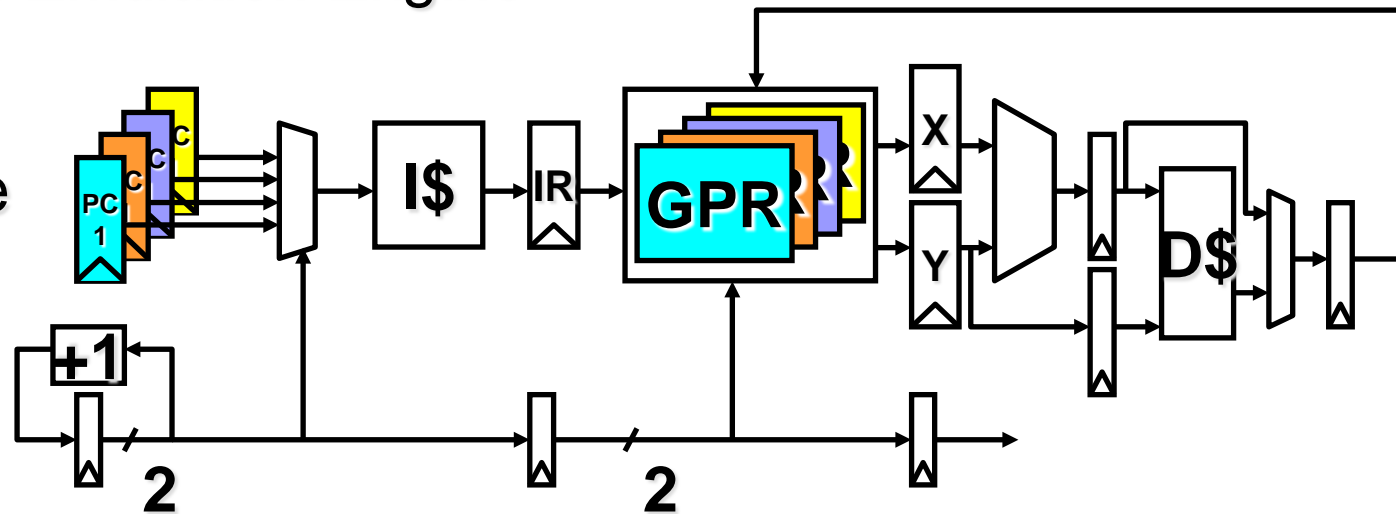
Host single-threaded: One target model per host pipeline

Host multi-threaded: Multiple target models per host pipeline



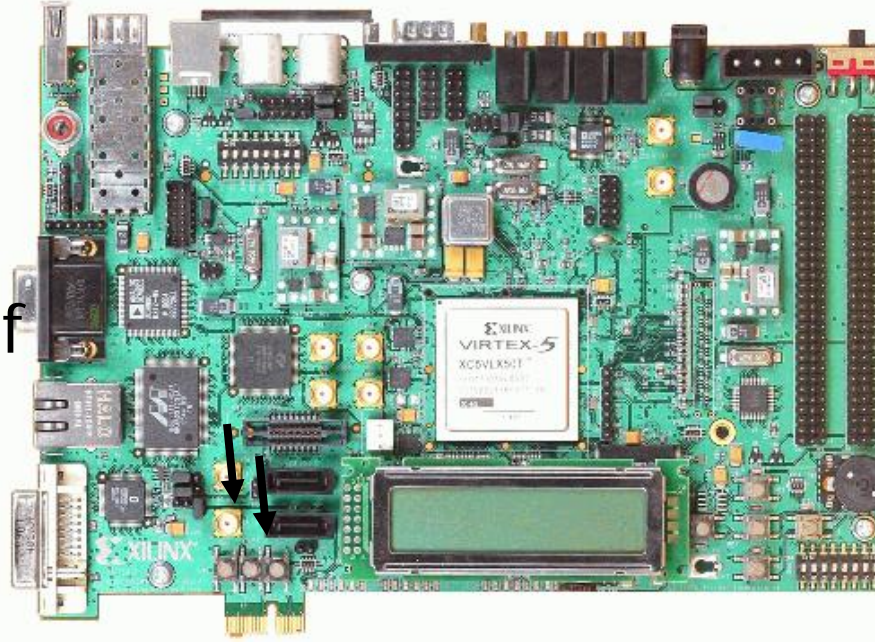
Multithreaded Emulation Engine (on FPGA)

Single hardware pipeline with multiple copies of CPU state

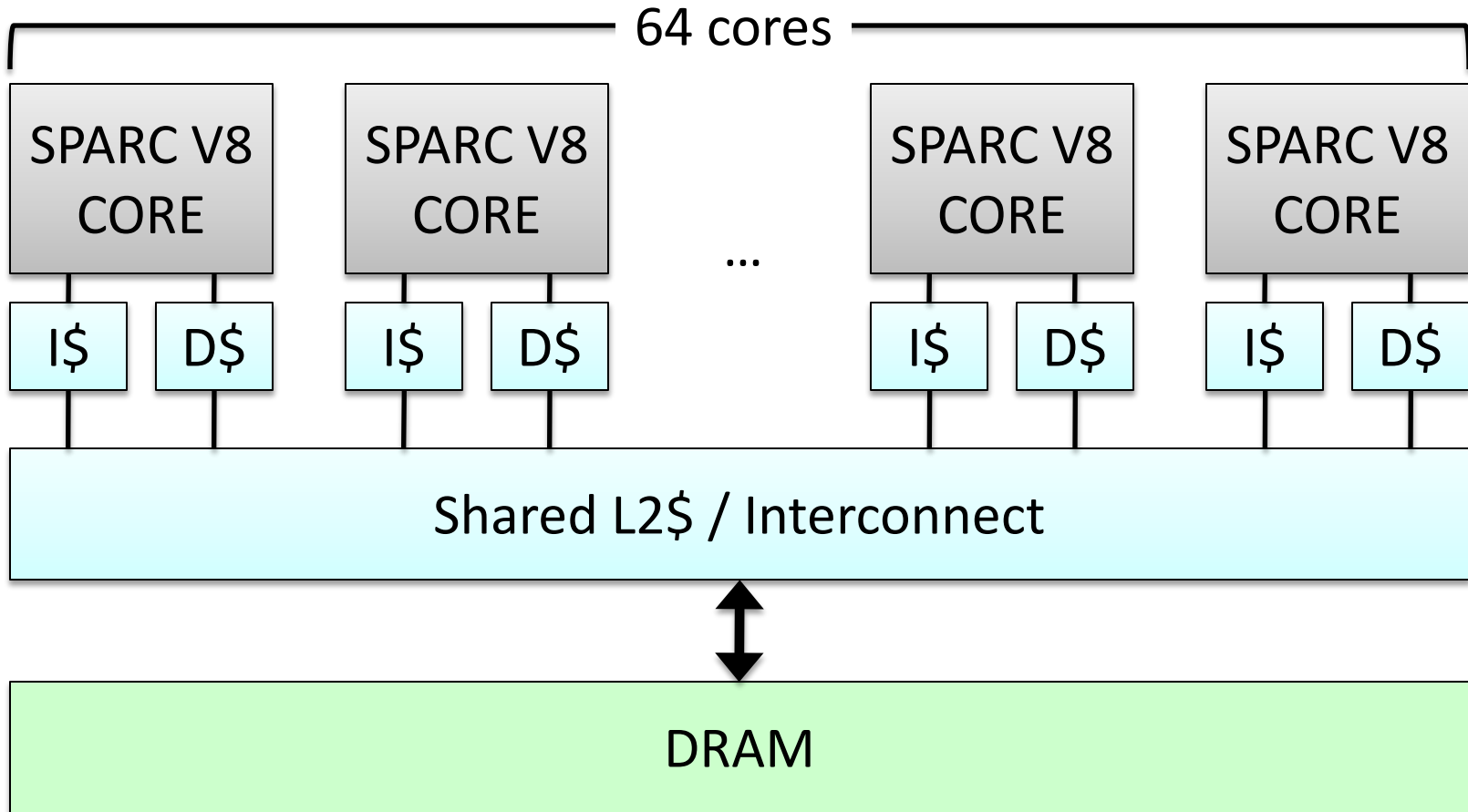


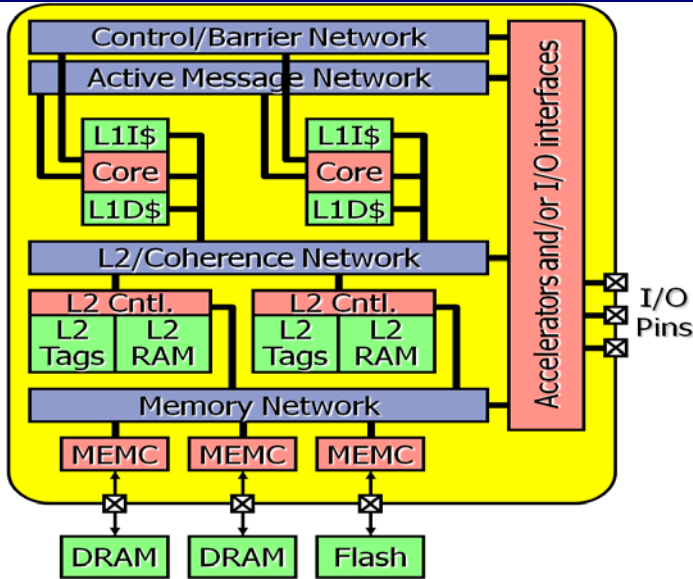
- Multithreading emulation engine reduces FPGA resource use and improves emulator throughput
- Hides emulation latencies (e.g., communicating across FPGAs)

- Rapid accurate simulation of manycore architectural ideas using FPGAs
- Initial version models 64 cores of SPARC v8 with shared memory system on \$750 board
- Hardware FPU, MMU, boots OS.

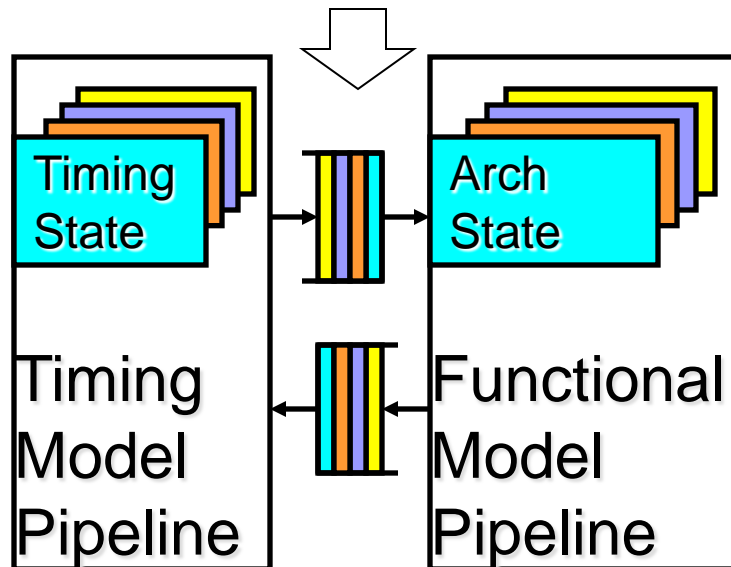


	Cost	Performance (MIPS)	Simulations per day
Simics Software Simulator	\$2,000	0.1 - 1	1
RAMP Gold	\$2,000 + \$750	50 - 100	100



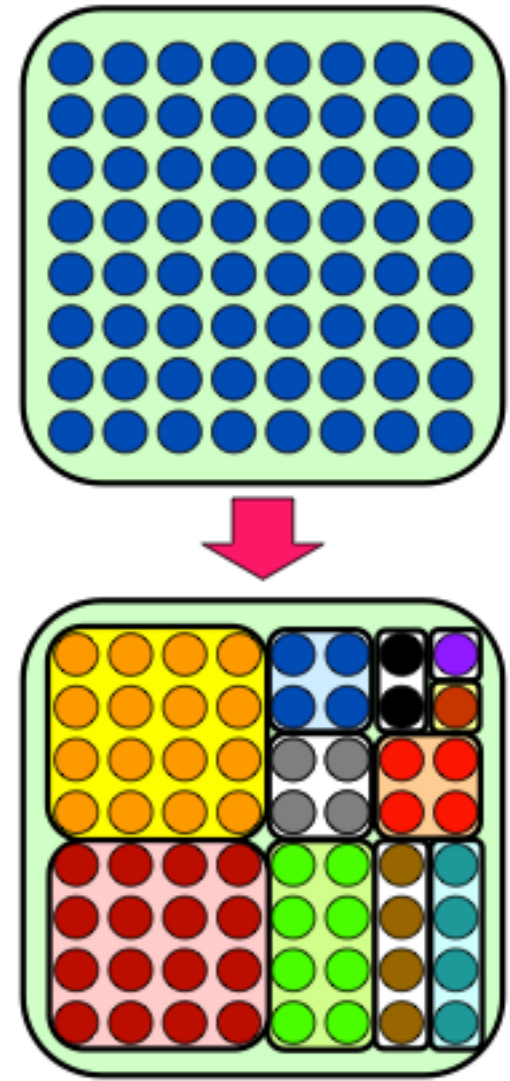


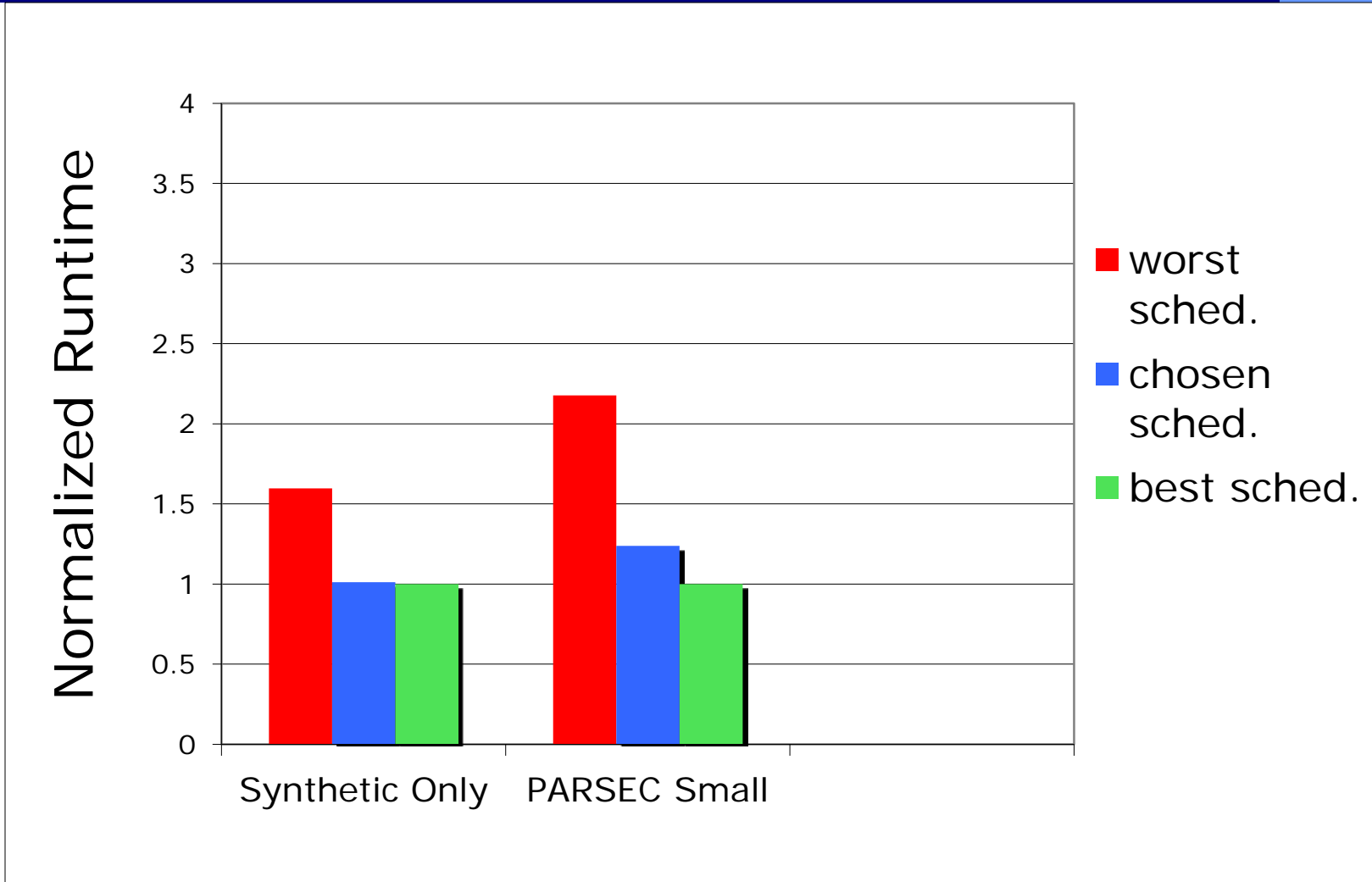
- RAMP emulation model for Parlab manycore
- SPARC v8 ISA
- Single-socket manycore target
- Split functional/timing model, both in hardware

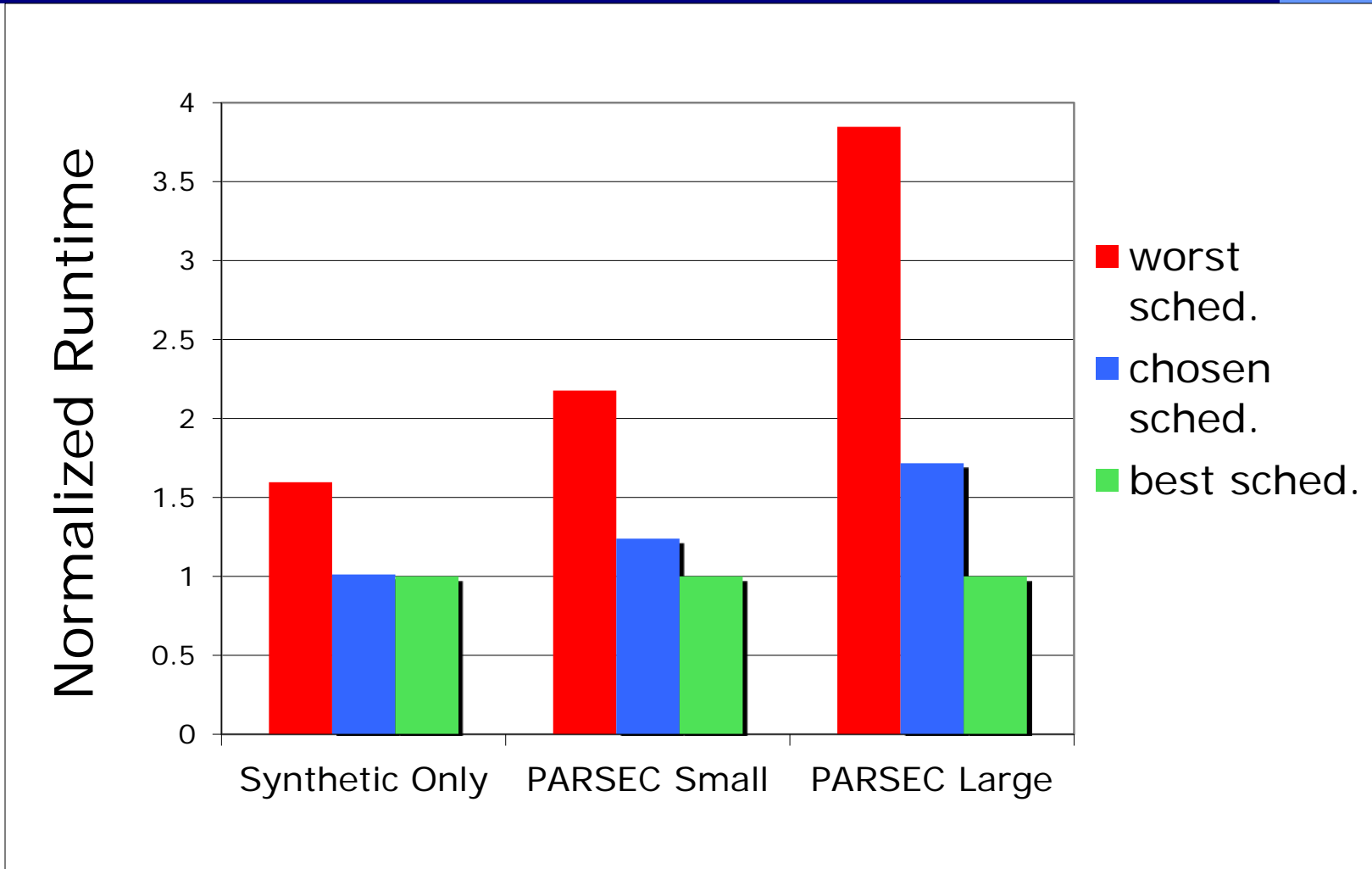


- Functional model: Executes ISA
- Timing model: Capture pipeline timing detail (can be cycle accurate)
- **Host multithreading** of both functional and timing models
- Built for Virtex-5 systems (ML505,

- Spatial resource allocation in a manycore system is hard
 - Combinatorial explosion in number of apps and number of resources
- Idea: use predictive models of app performance to make it easier on OS
- HW partitioning for performance isolation (so models still work when apps run together)
- Problem: evaluating effectiveness of resulting scheduling decisions requires running hundreds of schedules for billions of cycles each
- Simulation-bound: 8.3 CPU-years for Simics!
- Read ISCA'10 FAME paper for details







- The technique appears to perform very well for synthetic or reduced-input workloads, but is lackluster in reality!

- <http://sites.google.com/site/rampgold/>
- BSD/GNU licenses
- Many (100?) downloads

- Used by Xilinx tools group as exemplar System Verilog design

- Architects will use it!
 - Actually, don't want to use software simulators now
- Make everything a run-time parameter
 - Avoid resynth + P&R
- Difficult to modify highly tuned FPGA designs
- Functional/Timing split should be at μ arch. block level
 - Really build a *microfunctional* model
- Standard ISA only gets you so far in research
 - Research immediately changes ISA/ABI, so lose compatibility
- Target machine design vital part of architecture research
 - Have to understand target to build a model of it!
 - Need area/cycle-time/energy numbers from VLSI design
- FAME-7 simulators are very complex hardware designs
 - Much more complicated than a processor

- Richer set of target machines
 - In-order scalar cores, possibly threaded, + various kinds of vector unit
 - Various hardware-managed plus software-managed memory hierarchies
 - Cross-chip and off-chip interconnect structures
- More modular design
 - Trade some FPGA performance to make modifications easier
- Better/more timing models
 - Especially interconnect and memory hierarchy
- Better I/O for target system
 - Timed I/O to model real-time I/O accurately
 - Paravirtual-style network/video/graphics/audio/haptic I/O devices
- Better scaling of simulation performance
 - Weak scaling – more FPGA pipelines allow bigger system to run faster
 - Strong scaling – more FPGA pipelines allow same system to run faster
- *First target machine running apps on Par Lab stack in 2010*

- RISC-V: A new home-grown RISC ISA
 - V for Five, V for Vector, V for Variants
 - Inspired by MIPS but much cleaner
 - Supports 32-bit or 64-bit address space
 - 32-bit instruction format plus optional variable length (16/32/>32)
 - Designed to support easy extension
 - Completely open (BSD)
- Are we crazy not to use a standard ISA?
 - Standard ISA not very helpful, but standard ABI is
 - Standard ABI effectively means running same OS
 - Running same OS means modeling hardware platform – too much work for university
 - Also, only interesting standard ISAs are x86 & ARM (+GPU), both are too complex for university project to implement in full
 - We're going to change ISA plus OS model anyway
 - Midas modularity should make it easy to add other ISAs as well

- **Scalar cores**
 - In-order (out-of-order possible later)
 - Different issue widths, functional-unit latencies
 - Decoupled memory system (non-blocking cache)
 - Optional multithreading
 - New translation/protection structures for OS support
 - Performance counters
- **Attached data-parallel accelerator options**
 - Traditional vector (a la Cray)
 - SIMD (a la SSE/AVX)
 - SIMT (a la NVIDIA GPU)
 - Vector-threading (a la Scale/Maven)
 - All with one or more lanes
- Should be possible to model mix of cores for asymmetric platforms (Same ISA, different architecture)

- Distributed coherent caches
 - Initially, using reverse-map tag directory
 - Flexible placement, replication, migration, eviction policies
- Virtual local stores
 - Software-managed with DMA engines
- Multistage cross-chip interconnect
 - Rings/torus
- Partitioning/QoS hardware
 - Cache capacity
 - On-chip and off-chip bandwidth
- DRAM access schedulers
- Performance counters
- All fully parameterized for latency/bandwidth settings

- 1W – handheld
- 10W – laptop/settop/TV/games/car
- 100W – servers

- Gcc+binutils as efficiency-level compiler tools
 - Already up and running for draft RISC-V spec
- Par Lab OS + software stack for the rest
- Par Lab applications
- Pattern-specific compilers build with SEJITS/Autotuning help get good app performance quickly on large set of new architectures

- New DoE-funded Project Isis (with John Wawrzynek) on developing high-level hardware design capability
- Integrated approach spanning VLSI design and simulation
- Building ASIC designs for various scalar+vector cores
 - Yes, we will fab chips
- Mapping RTL to FPGAs
 - FPGA computers help software developers, helps debug RTL
- Why #1. For designing (not characterising) simple cores, power model abstractions don't work (e.g., 2x difference based on data values).

- Sponsors
 - ParLab: Intel, Microsoft, UC Discovery, National Instruments, NEC, Nokia, NVIDIA, Samsung, Sun Microsystems
 - Xilinx, IBM, SPARC International
 - DARPA, NSF, DoE, GSRC, BWRC