Languages and Compilers for Multicore Computing Systems

FRAN ALLEN
allen@watson.ibm.com

Workshop Keynote
IIT Kanpur, India
December 13, 2007

Topics

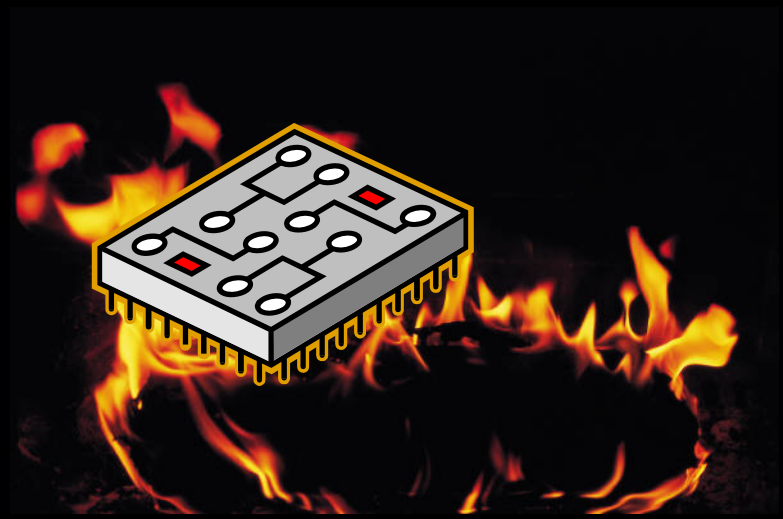
- Multicore Systems
- Parallelism
- The Challenge and the Opportunity

The Problem and Why Does It Matter?

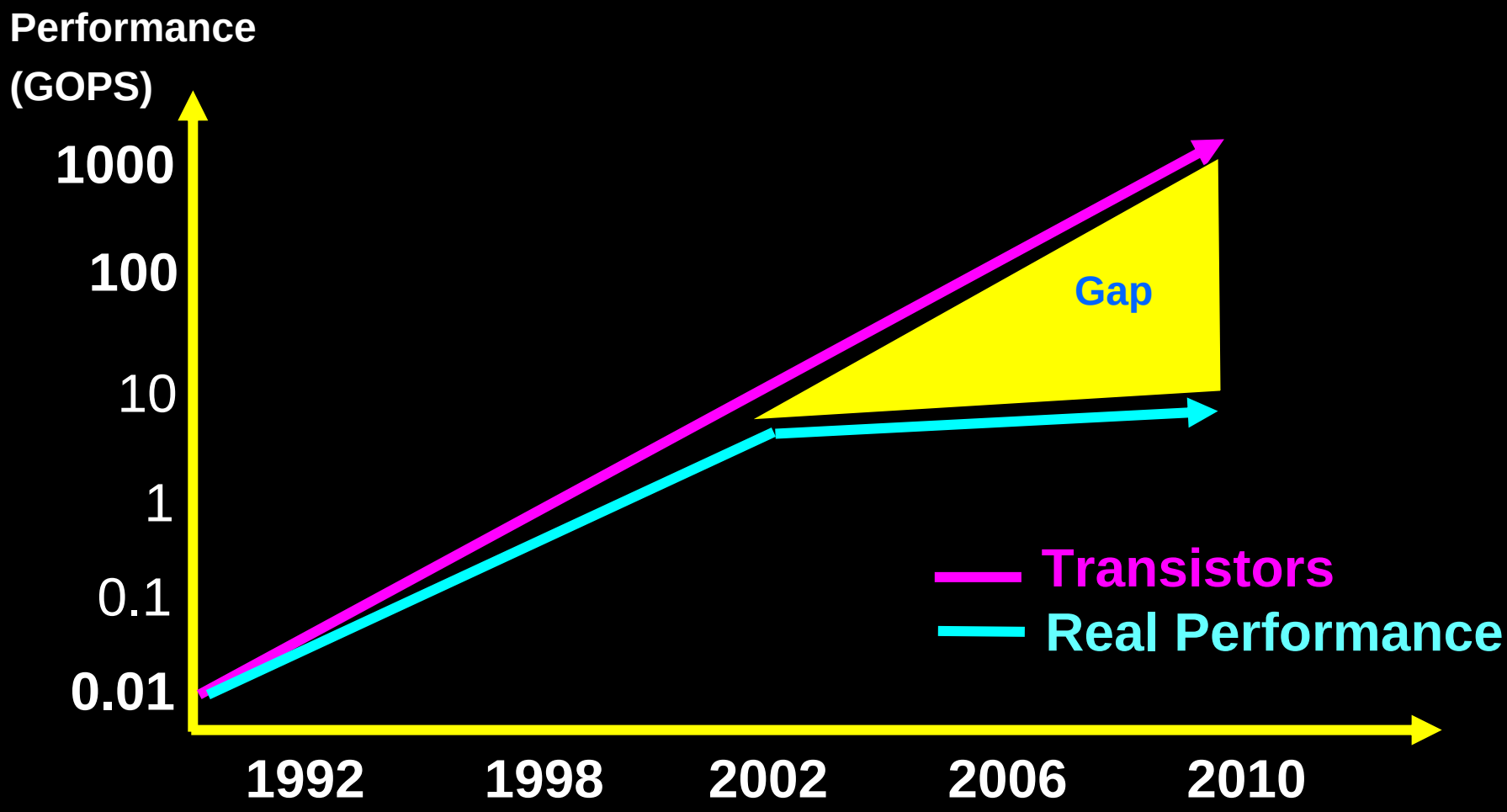
- Computers are hitting a performance limit
- “The biggest problem Computer Science has ever faced.” John Hennessy
- “The best opportunity Computer Science has to improve user productivity, application performance, and system integrity.” Fran Allen

The Problem

- Transistors continue to shrink
- More and more transistors fit on a chip
- The chips are faster and faster
- Result: HOT CHIPS!



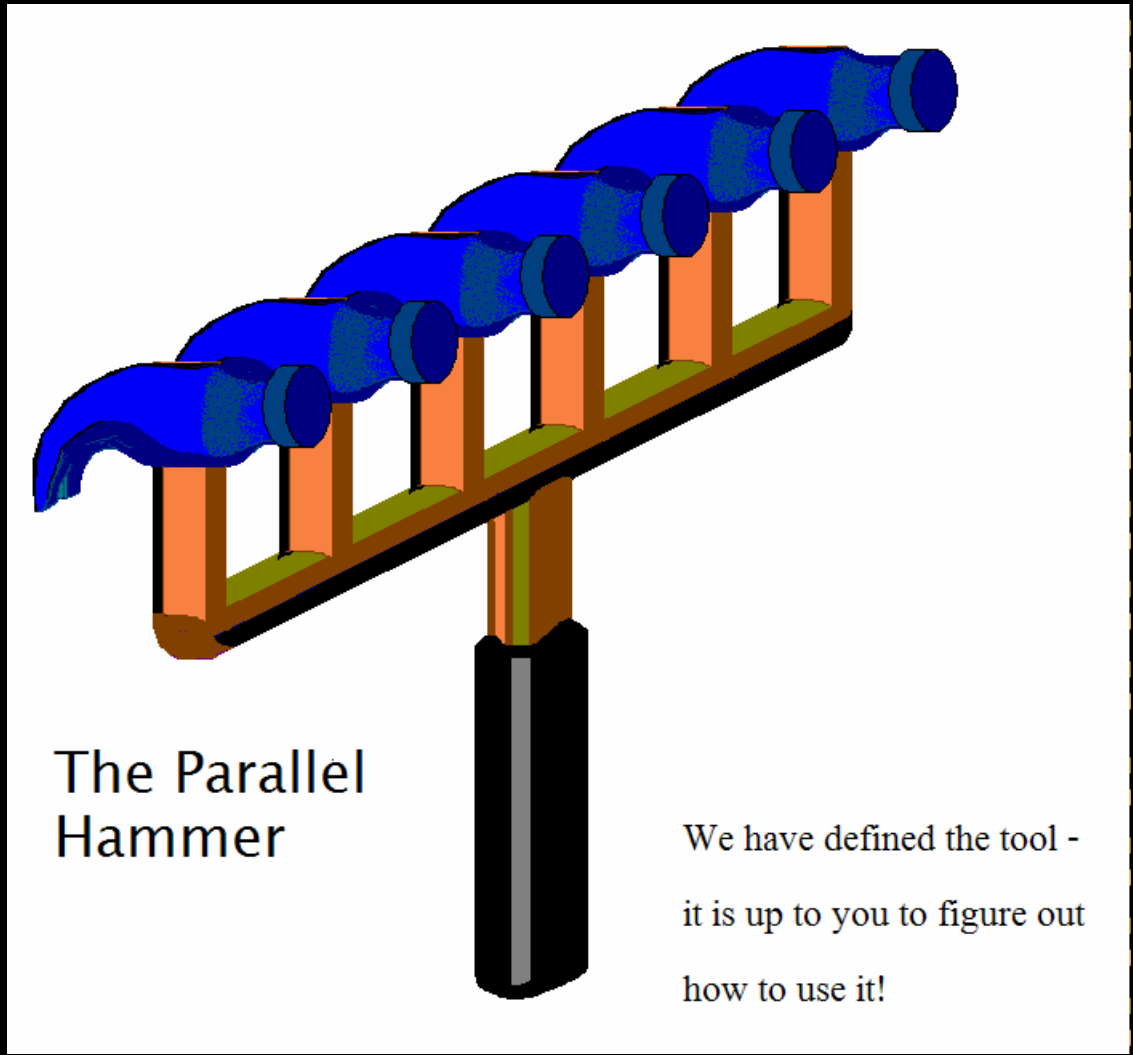
Real Performance Stops Growing as Fast



Solution: Multicores and Parallelism

- Simpler, slower, cooler processors
- Multiple processors on a chip
- Processors can work on independent parts of the same task
- Software and users organize work to maximize **PARALLELISM**

Parallelism Solves the Performance Problem! (or does it?)



Future of Multicores (An Early 2007 Prediction

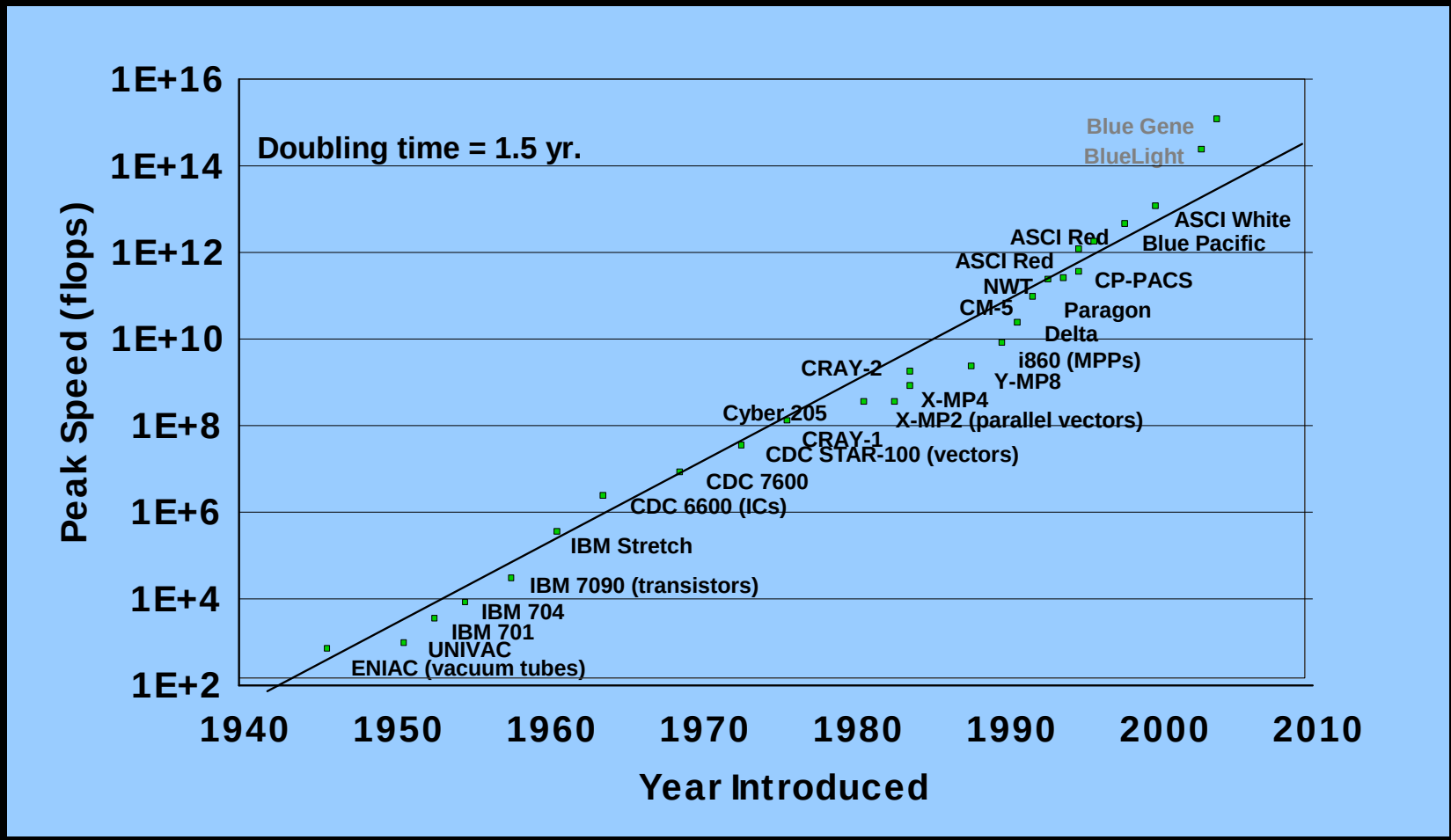
- Moore's Law predicts that the number of cores will double every 18 - 24 months
  - 2007 - 8 cores on a chip
  - 2009 - 16 cores
  - 2013 - 64 cores
  - 2015 - 128 cores
  - 2021 - 1k cores

20?? - LUNATIC LEVELS OF PARALLELISM!!

Parallelism is Moving to the Desktop, the Laptop, Handhelds…. !!

- High performance computing applications and computers have long used parallelism for performance.
- Microprocessors now need parallelism to stay on the performance curve.

Peak Performance Computers



Software Capability is Way Behind Hardware
Multi-core processors are here now

- Can compilers be constructed that abstract thread and data level parallelism from today's sequential languages?
- Do we need new programming languages?
- Is just more threads the answer?
- Is data parallelism the low hanging fruit?
- Is the answer specialized runtime, middleware, programming models?

Research in these areas is critical

Automatic Parallelization is Hard

- Disambiguating data references is hard:
  - Storage reuse
  - Procedure boundaries
  - Pointers
  - Caches
- Forming useful parallelism is a challenge
  - Data is a problem
  - Multiple models of parallelism

My Wish List of Changes and Opportunities

- Establish what we mean be PARALLELISM
- Establish new clearly defined hierarchical computational models
- Application needs and software capabilities determine the computer architecture not the other way around
- Eliminate caches
- Eliminate pointers

OPPORTUNITIES

- New very high level languages
- New compiler techniques to manage data locality, integrity, ownership, … in the presence of parallelism.
- Influence the architects before it is too late
- Rebuild the software stack
- Establish overall system goals:
  - User Productivity
  - Application Performance

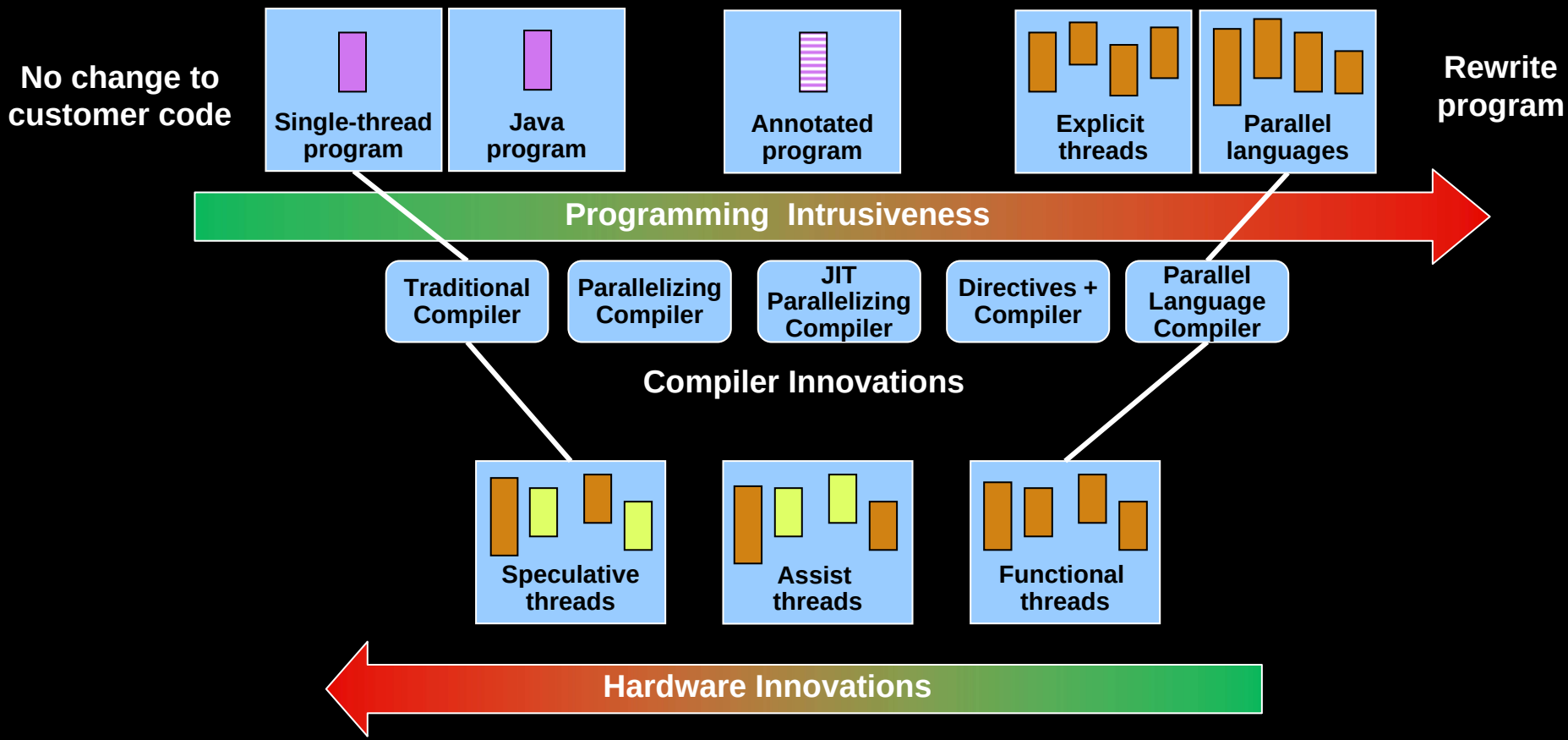
END OF TALK

BEGINNING OF DISCUSSION

PTRAN (mid -1980s to mid -1990s)

- Research on automatic parallelization
  - Program Dependence Graphs
  - Constructing Useful Parallelism
  - Static Single Assignment (SSA)
  - Whole Program Analysis Framework
- Compilers for RP3, 3090
  - IBM's XL Family of Compilers
  - Fortran 90
- Run-time technologies for parallel code
  - Dynamic Process Scheduling
  - Debugging
  - Visualization

Exploiting Multi-Core Multi-Function Chips: Some Options



The Anatomy of a Supercomputer