When Will SystemC Replace gcc/g++?

Bodo Parady, CTO
Pentum Group Inc.
Sunnyvale, CA 94087
bparady@pentum.com

SystemC: The Next Programming Language

- Who
  - Who will support future system development?
- What
  - What is SystemC?
- Why
  - Why adopt SystemC?
- How
  - How will SystemC be adopted?
- When
  - When Will SystemC be adopted?

When Will SystemC Replace gcc/g++?

- Confluence of hardware and software development
- SystemC maintains application development at a high level of abstraction -- co-design
- Key issues are system communications and upgradeability of systems
- Hardware is changing so fast and suffering multiple limitations that programming models need to adapt
- Computer industry is consolidating, driving Open Source

Software versus Hardware Engineers

- Hardware design complex heterogeneous systems
  - 58,490 Computer Hardware Engineers@$76K
- Software implementation & programming applications for complex heterogeneous systems
  - 261,520 software engineers bls.gov @ $73K
- Transmeta: Instead of expensive hardware engineering to replace X86 chips, used software
- System Architects: Join and separate Hardware and Software engineering
- In the end, even hardware is defined in software

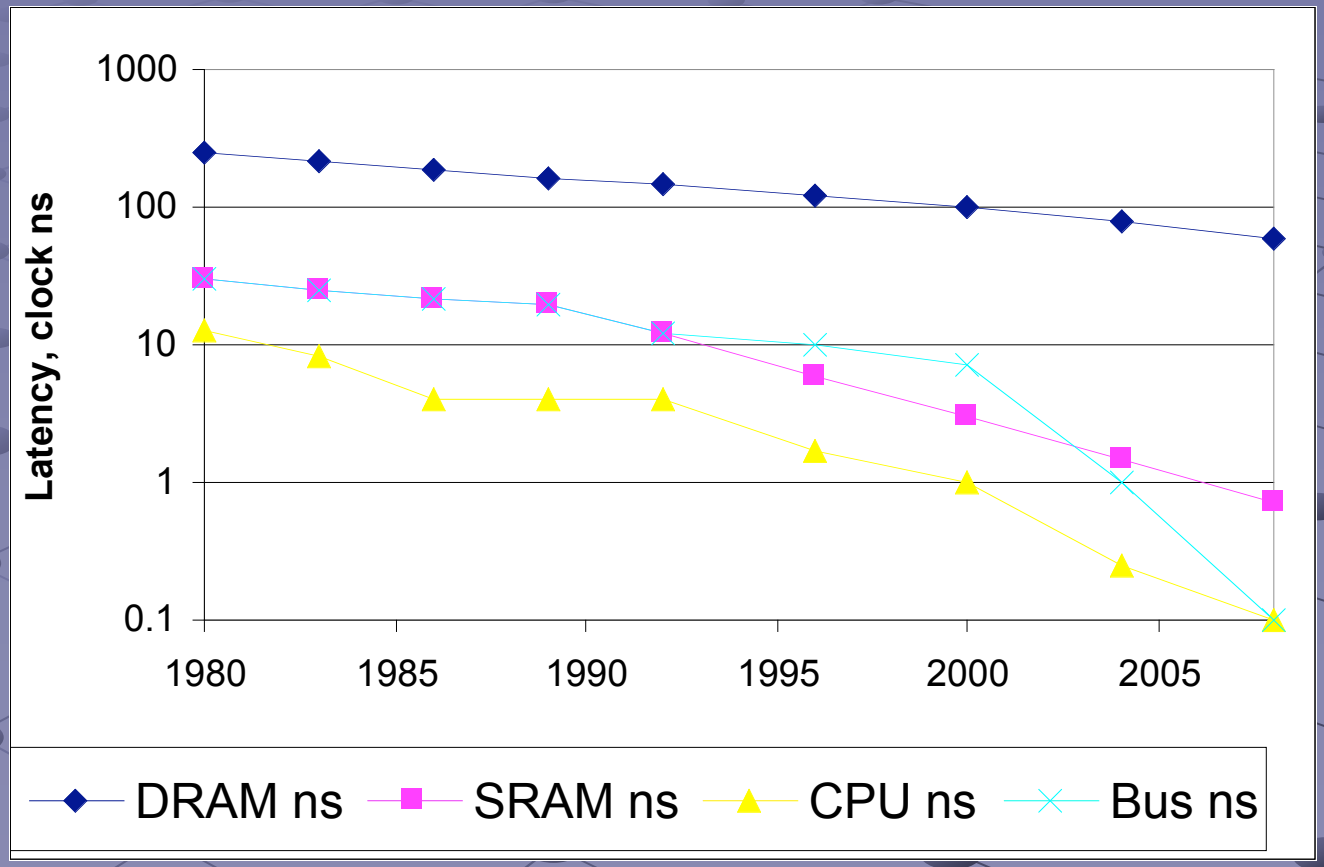
SystemC Models of Computation: Not Explicitly Hardware nor Software

- Register Transfer Level
- Behavioral level
- Functional level
  - Timed
  - Untimed
  - Dynamic vs. Static Dataflow Models
  - Untimed vs. Timed Functional Models
  - Kahn Process Network
- (TFM)Transaction level
- It is the models of computation that bridge hardware and software components, making them equivalent

Software Production Crisis

- No complete parallel programming model
- Trend to more CPU cores/die
  - Intel, Sun, IBM, Tensilica, Broadcom
- Trend to heterogeneity
  - UMA has become NUMA: Sun, SGI, IBM
  - Cray buys Octiga Bay with its Opterons+FPGAs
- Lots of solutions:
  - Distributed memory: MPI, sockets
  - Shared memory: OpenMP, threads, shmem, UPC, COF
- Standard programming languages lack implicit concurrency of SystemC

Memory/Clock/Bus Speed Trends Point to Changing Programming Models and System Architectures



The Hardware Industry Design Production Crisis

- High cost software tools impede development of new hardware
- Current application requirements for improved hardware are outstripping the productivity of programmers, architects, and designers.
- Systems becoming more complex with more heterogeneous components, e.g.. FPGAs
- Custom SOCs are becoming order of the day
- Not enough designers to handle the drive to eke out power,performance, and compatibility

What Will Hardware Industry Do?

- Too many software aspects on the plate:
  - Need to handle hardware upgrades
  - HDLs
  - SMP parallel programming models
  - Distributed memory
  - FPGA programming
- To speed adoption and development of complex heterogeneous systems:
  - SystemC into the open source/FSF release model
  - Need for multiple contributors

Conclusion: Software Needs to be Hardware Aware -- Co-Design

- Implicit concurrency
- Applications need means to handle hardware upgrades -- configuration hierarchy
- Must describe interconnects for optimization
- Synthesizeable, at least for FPGAs
- Some HDL capability
- SystemC: parallel and system oriented,
  - Hardware/Software co-design, simulation
  - Lacks tools for debugging concurrent processes, not breakpoints
  - What is needed is something similar to a RTL waveform viewer, but extended to plot complex data types.

Proprietary Compilers vs gcc/g++

- Top benchmark results still use proprietary compilers
  - gcc results within 10% of proprietary performance
- Most developers on Linux use gcc/g++
- Trend is to push proprietary compilers and IDEs to lock users in, e.g.. Microsoft
- Linux lesson: It took a major sponsor -- IBM -- to make it take hold
- Laptop battery lesson: Money in proprietary
- Opencores.org, hardware offering open IP

Computer Language Timelines

- Programming high level languages started with Fortran in 1957 after development of IBM 701 in 1953, acceptance in 1964
- C developed in1973, PDP-8 developed in 1965, acceptance 1985.
- C++ developed in 1985, PDP-11 in 1970, acceptance in 1995.
- Makimoto's wave:
  - Mainstream silicon application is changing every 10 years

SystemC: The Language for the New System Architect

- Hardware and Software are treated as one system, separable at lower levels
- Models of computation and large scale components comprise the system
  - Not explicit procedures, objects, FPGAs, CPUs, controllers, memory, etc.
- All hardware/software development and co-design take place under a SystemC implementation

2010 System SystemC Prediction

- Foundries & FPGA vendors will issue proprietary languages and compilers to lock in users
- Look for Intel, AMD, IBM, Xilinx, Altera and VIA/TSMC to issue their own HDLs by buying CAD software suppliers
- SystemC or its successor will become the lingua franca to develop system prototypes
  - It will not be SystemVerilog
  - SystemC will change
- A major open source sponsor will be needed for SystemC's long-term success

END

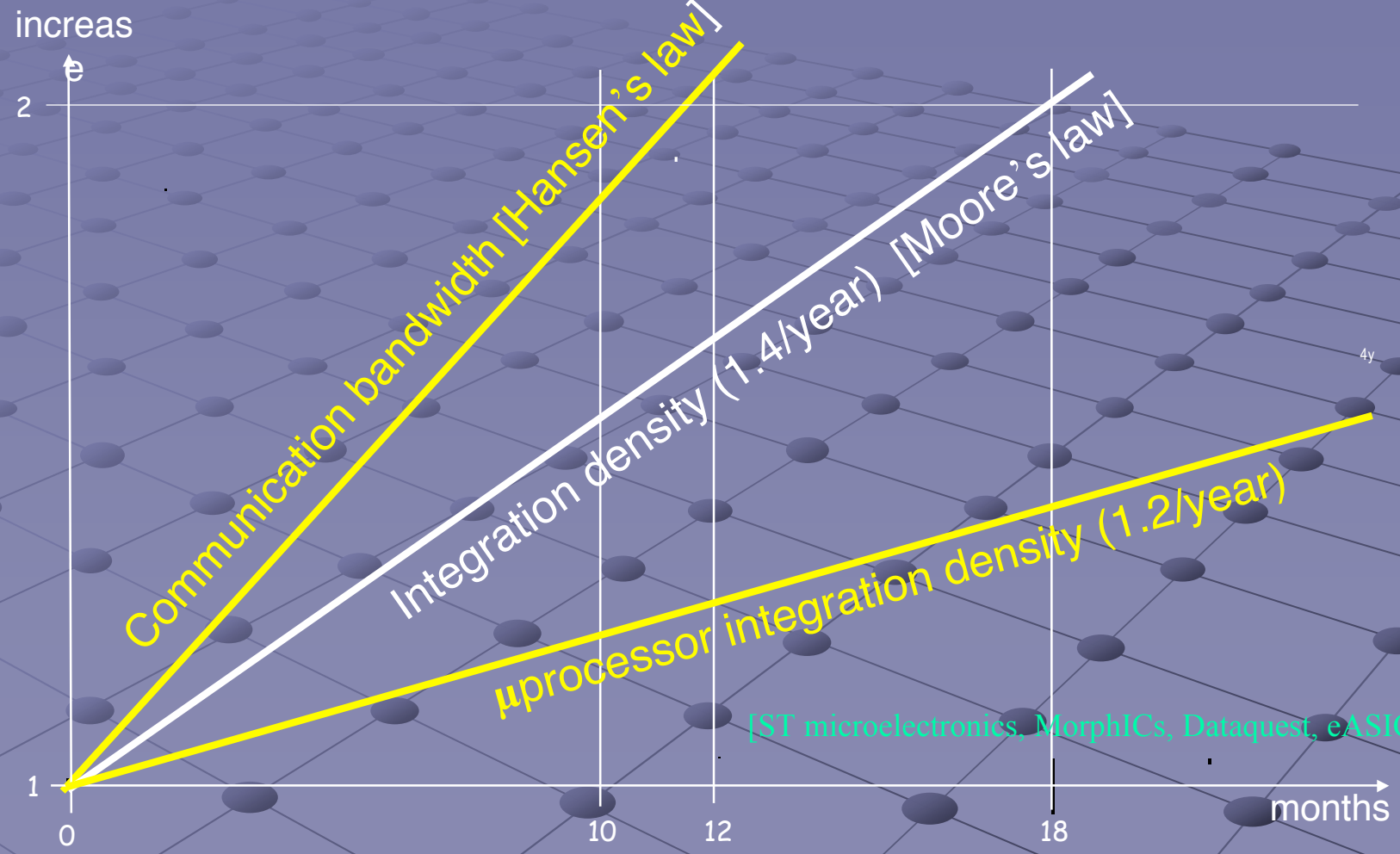
FINIS

BACKUP SLIDES

BLS Statistics

- The BLS uses term "system software engineers"
  - http://www.bls.gov/oes/2003/may/oes151032.htm
- Includes driver, kernel, OS, compiler, embedded?, engineers.
- It excludes the class "programmer" which means Java, J2EE, HTML, apps. etc.
- The hardware engineer group is at:
  - http://www.bls.gov/oes/2003/may/oes172061.htm
- I tend to run the former group together
- Narrow class of embedded programmer tends to fit more into the hardware engineer class.

Programming Needs to Catch up to Communications



Opencores.org

- OpenCores: people who are interested in developing hardware, with a similar ethos to the free software movement.
- Emphasis is on digital modules called 'cores', since FPGAs have reduced the incremental cost of a core to approximately zero.
- Example of the drive to open source, but now in hardware -- Hardware is software

Software/Hardware Industry Trends

- Software industry handles consolidation of hardware CPUs by open source release of gcc/g++ compilers
  - Hardware vendors cannot afford state-of-the-art compiler development
  - Puts the tools of application development in everyone's hands very quickly and cheaply.
- To speed implementation and development of complex heterogeneous systems, it will take the adoption of SystemC into the open source/Free Software Foundation release model.

Growing Physical Constraints Mean More Architectural Optimization

- Physics trumps Computer Science
  - Heat - limits proximity and adds costs
  - Interconnect latency - clock speed limits
  - Interconnect bandwidth - Rent's rule $IOs=kN^{1/4}$
  - Physical distance between nodes - c
  - Memory performance - DRAM stuck
  - Processor speed - Limited by future tech
  - System costs - Limits number of CPUs
  - Non-portable software - Development takes time

When Do Software Engineers and SystemC Take Over?

- Lesson: It takes about 10 years for a new computer language to be fully accepted
- Makimoto's wave:
  - Mainstream silicon application is changing every 10 years
- Computer language lifetimes are about 25 years to over 50 years
- Look for SystemC to take over from gcc/g++ in 2010 since SystemC emerged in 2000