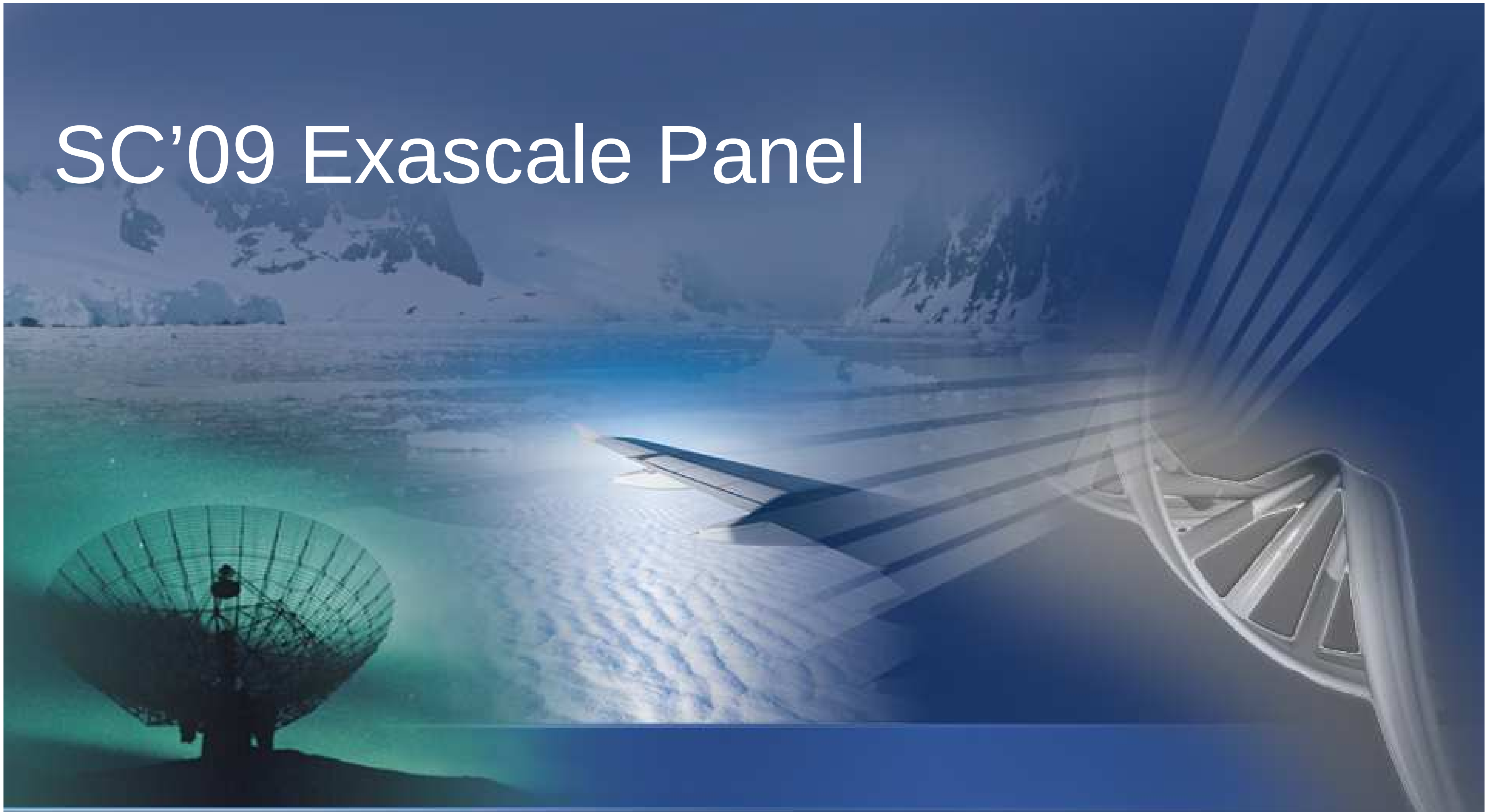
SC’09 Exascale Panel



Steve Scott
Cray Chief Technology Officer

Exhibitor Forum,
SC’09

THE SUPERCOMPUTER COMPANY

My Exascale(flop) Assumptions

- A plausible first exaflop:
  - 2017, 16nm IC technology
  - 8 TF per socket ⇒ 125K sockets
  - 250 watts/socket system power ⇒ 31 MW (PUE will be < 1.33)
  - 384 sockets/cabinet ⇒ 325 cabinets ⇒ 10,000 sq. ft.
- Main memory
  - Bandwidth/flop *will* be significantly lower than today
  - But we'll have another layer in-between LLC and main memory
- Network
  - All optics (transceivers integrated in package)
  - High radix routers with very low network diameter (flattened butterfly, dragonfly, etc.)
  - Bandwidth/flop *will* be lower than today
- Processor architecture
  - Heterogeneous (most of the flops will be in "accelerators")
- Bill asked me to focus on scalability and reliability…

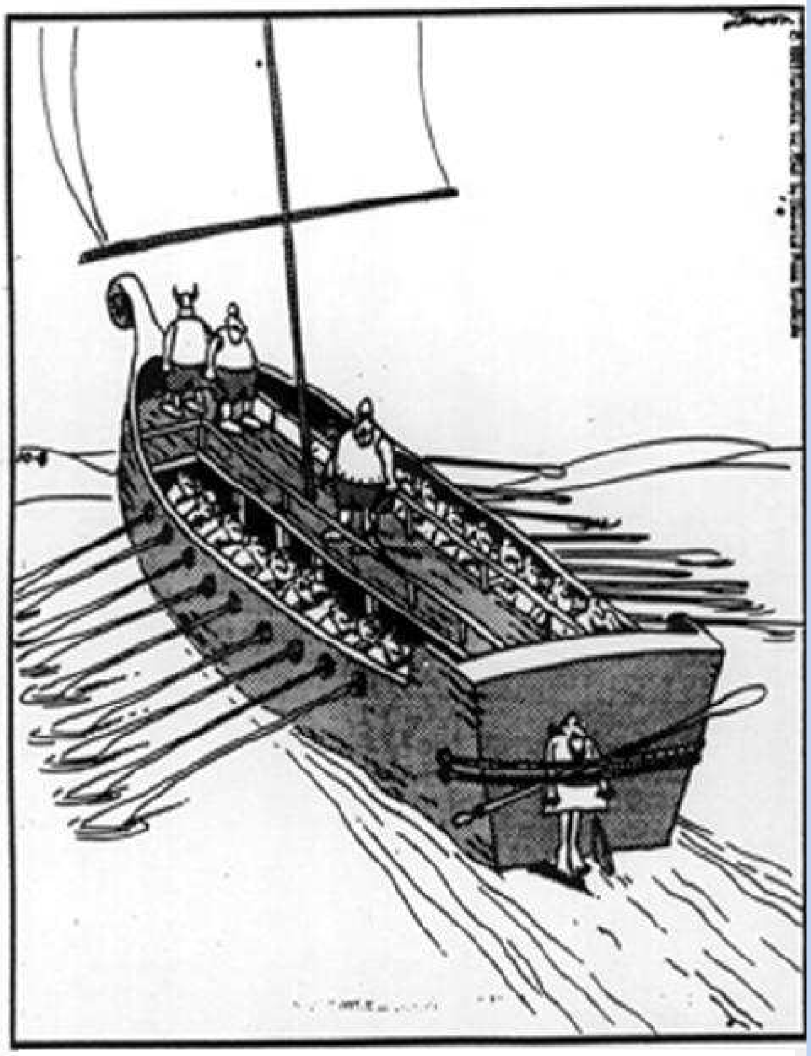
Scaling

- Building a physical system of 10K sq. ft., 31 MW and 325 cabinets is quite feasible
  - About 3x of where we are today
  - Optics makes the cabling manageable
- Scaling the OS doesn't particularly concern me either
  - One instance of the OS per "node" (1-4 sockets)
  - Do not need to run the OS on every "core"
    - OS runs on the serial processors
    - User-level (or partially privileged) runtime on the accelerator
    - OS instances on the compute nodes are very lightweight/low noise
- Application scaling and resiliency are another matter…

Reliability

- Yesterday's approach was to make the underlying system reliable through good engineering and redundancy

- This is simply not going to work at the Exascale
  - Component counts going way up
  - Underlying components getting less reliable
  - (of course we'll still use redundancy extensively)

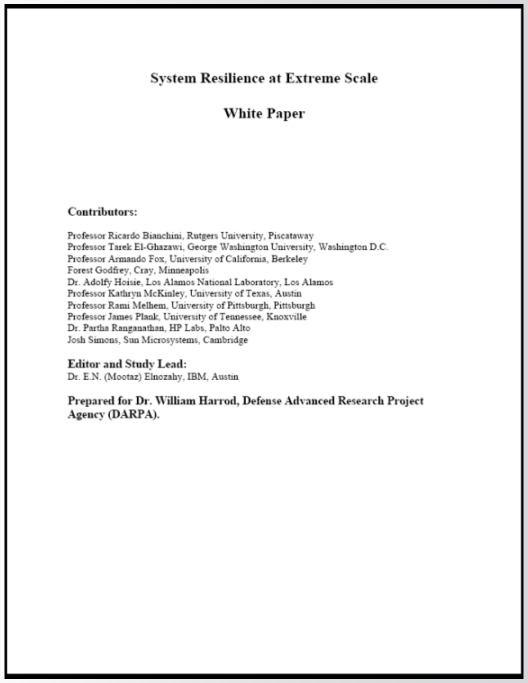


The better equipped slave ships, of course, always carried a spare

DARPA Exascale Resiliency Study

April–November 2008

Objective: Analyze the current problem in reliable systems and suggest new avenues for research in resilient systems at large scale.



System Resilience at Extreme Scale

White Paper

Contributors:

Professor Ricardo Bianchini, Rutgers University, Piscataway
Professor Tarek El-Ghazawi, George Washington University, Washington D.C.
Professor Armando Fox, University of California, Berkeley
Forest Godfrey, Cray, Minneapolis
Dr. Adolfy Hoisie, Los Alamos National Laboratory, Los Alamos
Professor Kathryn McKinley, University of Texas, Austin
Professor Rami Melhem, University of Pittsburgh, Pittsburgh
Professor James Plank, University of Tennessee, Knoxville
Dr. Partha Ranganathan, HP Labs, Palto Alto
Josh Simons, Sun Microsystems, Cambridge

Editor and Study Lead:
Dr. E.N. (Mootaz) Elnozahy, IBM, Austin

Prepared for Dr. William Harrod, Defense Advanced Research Project Agency (DARPA).

- Estimate 20% of today's computing capacity in large HPC systems is wasted due to failures and recoveries
  - Will get worse as system size continues to scale
- Example:
  - 1000 FIT processor has an expected lifetime of > 100 years
  - 125,000 processors ⇒ three failures per day
  - *The market doesn't need to solve this problem*
- Can make system resilient
  - It's the applications that are hard
- Checkpoint/Restart is a temporary solution
  - MTBF is shrinking due to scale and technology
  - Checkpoint times growing due to larger memory
  - As MTBF approaches checkpoint time, will no longer work!

Available at www.darpa.mil/ipto/personnel/docs/ExaScale_Study_Initial.pdf

Application Resiliency

- Checkpoint/Restart and resilient communication protocols will help
  - But need new approaches to make applications resilient at extreme scale
- We need community engagement and collaboration
  - Need better understanding of classes of apps and resiliency attributes
  - Need standards for applications
    - How to specify variable resiliency requirements (e.g.: reliability critical sections)
    - APIs for the system to provide failure information to the application
    - APIs for the applications to specify actions to the system (e.g.: restart *this* piece)
- May be potential for (semi-)automatic application resiliency
  - Use compiler techniques to decompose applications into sufficiently constrained work items
    - Some such decomposition already occurs as part of parallelization
    - Expect that user directives will be needed to make this work well enough
  - Use runtime techniques to reliably execute these work items
    - Work distribution is already done by some runtimes
    - Need to add reliability and encapsulation aspects
    - In-memory checkpoints and transactional techniques may apply

Thank You

- Steve, could you concentrate on scalability and reliability challenges at this scale and this power?

- We should not assume more than 2 GHz for the clock rate and probably well less—perhaps 500MHz to 1 GHz, given the aggressive signaling we are going to need to get down into the few pJ per operation range implied budgets.
- This means O(100M—1B) threads. Is that even programmable?
- If we want to have the system provide ~20 hours between losses of an application using most of the resources and say 15+ days between unscheduled system reboots, we are going to face unprecedented reliability and resiliency requirements.
- I personally would like to see the system be balanced in that we do not make memory and interconnect balance factors worse than they are on today’s best machines.