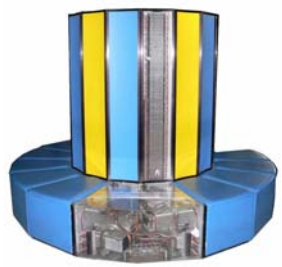
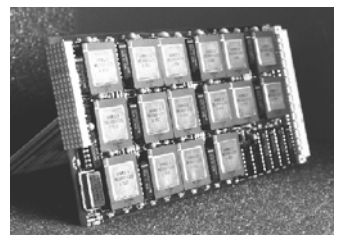
Better Multicore Software Debugging using Virtualization

Jakob Engblom, PhD
Business Development Manager
Virtutech
jakob@virtutech.com

Multiprocessing is the Future

- I guess we all know this since we are here today
  - So I'll spare you the ten slides I have at the ESC explaining why embedded programmers need to care about MP
- Note that I make no real distinction between multicore and discrete multiprocessor systems in this talk
  - From a software perspective, they are the same
  - The difference is in the details and timing, not essentials



It’s All About Software

virtutech

Electronics Is Software

Electronics is software. Shipping a system is largely about identifying and removing defects from the software and keeping them from creeping back in as the product evolves.

St. Jude Medical Cardiac Defibrillators

St. Jude Medical Inc., a Canada-based medical technology company, announced in June 2005 that some of its implantable defibrillator models, or ICDs, have a software problem that could cause the heart-shocking device to malfunction.

The Explosion of the Ariane 5

On June 4, 1996 an unmanned Ariane 5 rocket launched by the European Space Agency exploded just forty seconds after its lift-off from Kourou, French Guiana. The rocket was on its first voyage, after a decade of development costing $7 billion. The destroyed rocket and its cargo were valued at $500 million. A board of inquiry investigated the causes of the explosion and in two weeks issued a report. It turned out that the cause of the failure was a software error in the inertial reference system. Specifically a 64 bit floating point number relating to the horizontal velocity of the rocket with respect to the platform was converted to a 16 bit signed integer. The number was larger than 32,767, th integer storeable in a 16 bit signed integer, and thus the conversion failed.

Los Angeles Times latimes.com

2:43 PM PDT, October 13, 2005

Software Glitch Triggers Toyota Prius Recall

By James F. Peltz, Times Staff Writer

In what's believed to be the first recall of hybrid cars for engine-related problems, Toyota Motor Corp. said today that it is notifying about 75,000 owners of its hot-selling Prius about a potential software glitch that could cause the car to stall or shut down.

The voluntary recall dented the good reliability record of the Prius, whose sales have jumped in the past two years as drivers sought better fuel economy in the face of soaring gasoline prices.

ADVERTISEMENT

The problem involves the hybrid's computer software, rather than mechanical parts,

The Register

Software glitch blamed for CryoSat loss

Rockot still ok to fly again

By Lucy Sherriff

Published Thursday 27th October 2005 14:30 GMT

Get breaking Reg news straight to your desktop - click here to find out how

Officials investigating the loss of the CryoSat mission have revealed that a software glitch in the on board flight control system on the new, upper stage of the rocket was to blame.

There is no fault with the Rockot launcher itself, Russian officials said,

virtutech

Software gets Even Harder on MP

- Parallelism required to gain performance
  - Parallel hardware is “easy” to design
  - Parallel software is **hard** to write
- Existing software assumes single-processor
  - Shared-memory multiprocessors very rare previously
  - Multitasking != multiprocessor-ready
  - Multiprocessors expose latent problems in code
  - Might break in new interesting ways on multiprocessor
  - Multitasking no guarantee to run on multiprocessor
- Fundamentally hard to grasp true concurrency
  - Especially in complex software environments
  - Some new phenomena cannot occur on a single processor running multiple threads

Multiprocessors & Debug

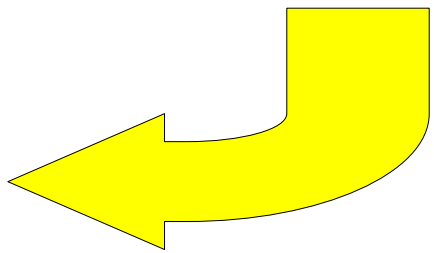
- Limited visibility into hardware
  - Single debug port, multiple processors
  - High speed, concurrent execution
- Timing-sensitive chaotic behavior
  - Small changes in timing alters system behavior radically
  - Hardware variations impact software behavior
- Lack of determinism
  - Rerunning a program gives different results
  - Hard to reproduce bugs
- Heisenbugs
  - Inserting probes to trace behavior alters behavior
  - Bugs hide when they are being debugged
- System keeps running even if one core stopped

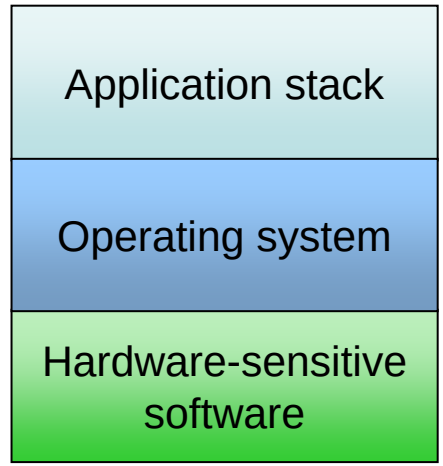
Virtual Software Development

In 60 seconds

Traditional Software Development

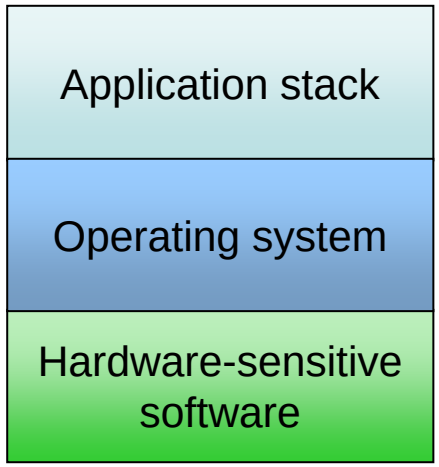
- Software developers create a production binary
- Production binary runs on the physical hardware





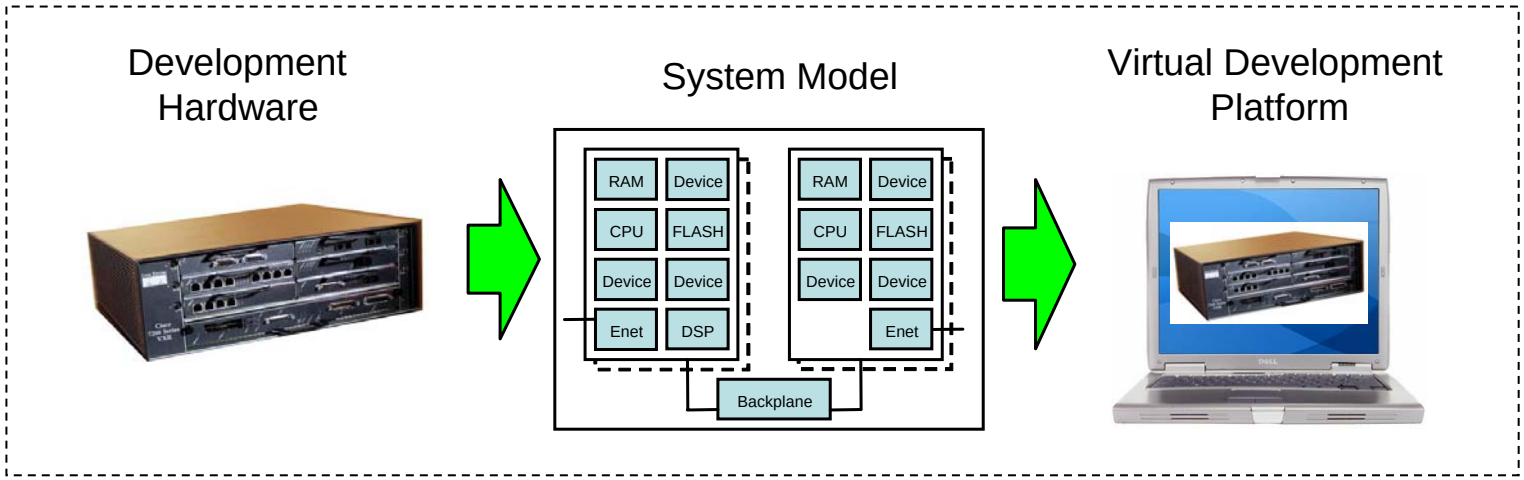
Virtualized Software Development

- Same binary runs inside virtualized software development environment
  - On a virtual model of the hardware



Virtual Software Development

- Virtual software development:
  - Replace development hardware with a simulation in a PC
  - Run the complete same binary stack as the physical target
  - Same compilers, same build setttings
  - “One-track development”



How Can Virtualization Help with Multiple Processors?

virtutech

Three Steps of Debugging

1. Provoking errors
   - Forcing the system to a state where things break
2. Reproducing errors
   - Recreating a provoked error reliably
3. Locating the source of errors
   - Investigating the program flow and data
   - Depends on success in reproduction

Virtualized hardware helps with all three steps

Provoking Errors

- Virtualization provides unparalleled **control** over system configuration and execution
- Vary system configuration
  - Number of processors
  - Speed of processors
  - Software loads on different boards
  - Like testing on a range of real hardware
- Systematically search for error cases
  - Make a processor slower or stop it entirely
  - Increate communication latencies
  - Slow down individual processors to increase perceived load
  - Inject actual faults in the system
  - Work like hardware engineers: set up bad cases in the system to expose likely bugs

virtutech

Reproducing Errors

- Virtual hardware state can be **checkpointed**
- Virtual hardware is **deterministic**
  - Variability explicitly programmed
  - Timing of execution on each processor
  - Communication between processors
  - Recording asynchronous inputs to simulation session
- Then, if an error has been provoked in simulation, it can be reproduced with ease
  - Reset back to initial state (or restore a checkpoint)
  - Rerun test case that ended in error
  - Same error state results
  - ...any number of times
  - ...on any machine running the simulator
  - ...from a checkpoint distributed to multiple developers

virtutech

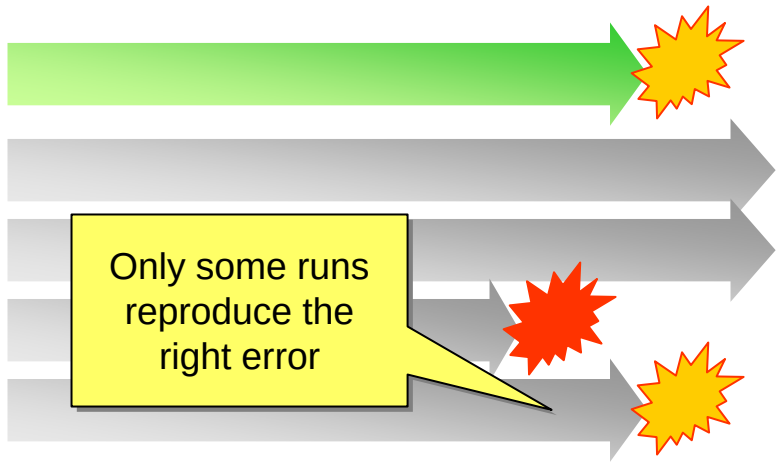
Locating and Fixing Errors

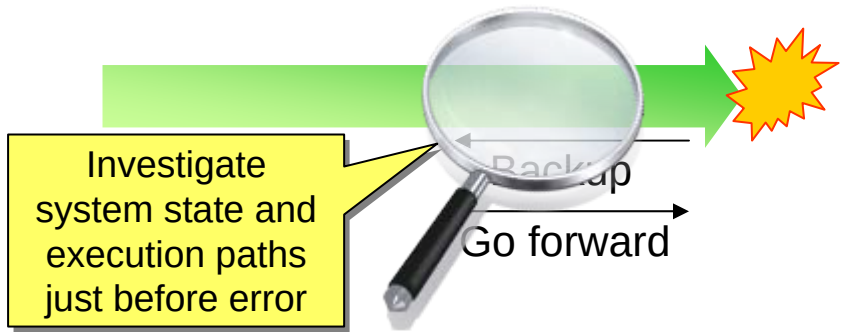
- Virtual hardware provides superior visibility
  - Memory, processor registers, device state
  - For all processors in a system at a synchronized time
- Global system stop
  - All processors stop simultaneously
- No probe effect from instrumentation
  - Tracing and observation from the outside, not by code instrumentation or modification
- No timing disturbance from breakpoints
- Single-step synchronously
  - All other processors wait patiently while one is stepping
- Heisenbugs cannot occur
  - Determinism helps us avoid Heisenbugs

Reverse Debugging

- Stop & go back in time
  - Instead of rerunning program from start
  - No need to rerun and hope for bug to reoccur
  - Investigate exactly what happened this time
  - Breakpoints & watchpoints backwards in time
  - Very powerful for parallel programs

  - Very hard to do for multiple processors using trace on physical hardware





Virtual Validity

- Fidelity of virtual models:
  - We have to admit that virtual models are never quite like the real thing
  - But they are close enough to provide very useful service
- Any bugs found in simulation are valid bugs
  - Simulation does explore a range of behaviors of the real system. If desired, simulation can force certain behavior.
  - Note that precise timing simulation is not really possible
    - "Cycle Accuracy" is really a "Cycle Approximate"

See Ekblom and Engblom, "Simics: a commercially proven full-system simulation framework", *SESP 2006*, for a deeper discussion

Some Examples

virtutech

Divide-by-zero in OS Kernel

- Operating-system kernel crash in virtual model
  - Divide-by-zero right in the kernel
  - Algorithm to determine and compensate for clock skew
  - Division by difference in time between two processors
- Virtual model had zero clock skew = provoked error
  - Could have happened on a real system
  - Just not very likely
  - Typical rare problem in the field
  - Essentially testing a rare corner case in system state

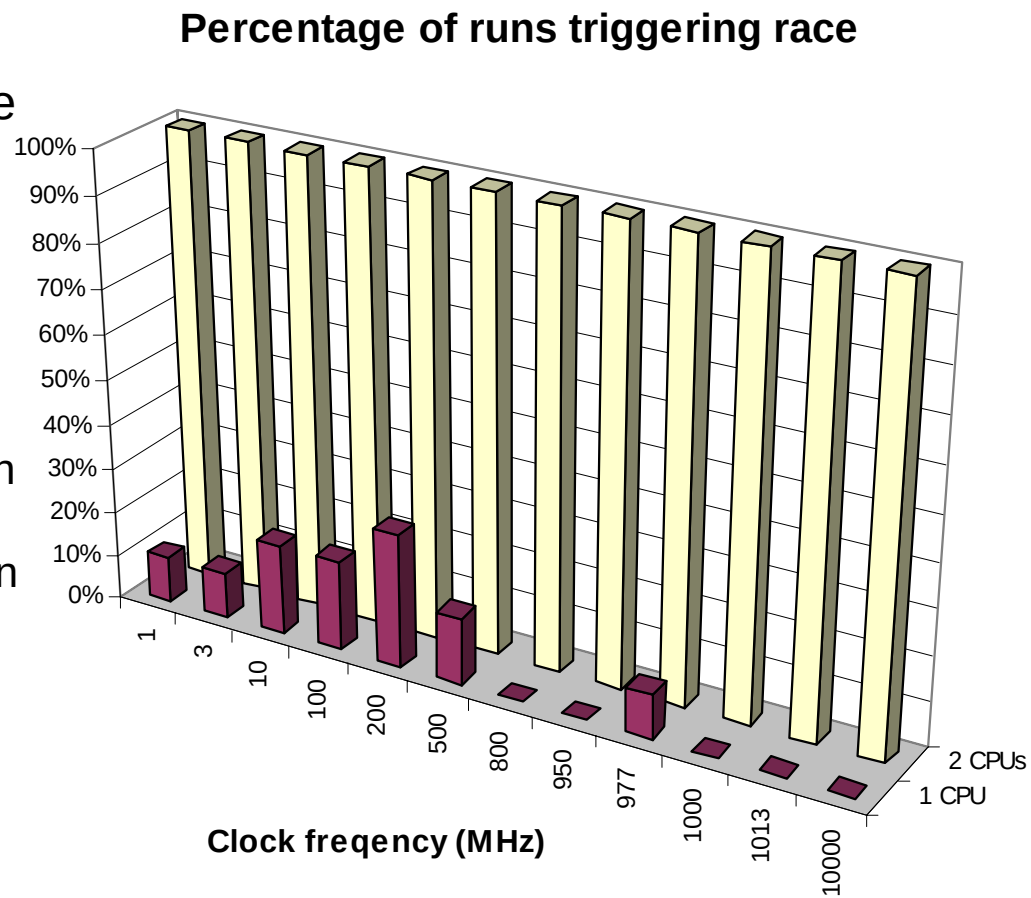
virtutech

Race Condition in Serial Driver

- Changed clock frequency of virtual MPC8641D
  - From 800 to 833 Mhz
  - OS froze on startup – quite unexpectedly
- Investigation:
  - Only happened at 832.9 to 833.3 MHz
  - Determinism: 100% reproduction of error trivial
  - Time control: single-step code feasible
  - Insight: look at complete system state, log interrupts, check the call stack at the point of the freeze, check lock state
- What we found:
  - ISR takes a lock on entry, and then expect a second external interrupt to occur to unlock the data structure. But this interrupt arrives before interrupts are reenabled, and thus we are stuck in deadlock. Took a few hours.

Multipro vs Multitasking

- Simulated single-CPU and dual-CPU
- Test program with embarrassingly bad race run 20 times on a range of clock frequencies
- Count percentage of runs triggering race
- Results:
  - Race always triggers in dual-CPU mode
  - Triggers around 10% in single-CPU mode
  - Higher clock = less chance to trigger
  - **Multitasking hides errors quite well**



Questions?