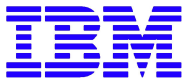


Out Of Steam ?
From "Hardware Verification Crisis"
to
"Crisis of Verification" ?

HVC 2007

Wolfgang Roesner

Distinguished Engineer
Hardware Verification
IBM Systems & Technology Group
Austin, TX. USA
wolfgang@us.ibm.com

Overview

1. Context : High-End Server Micros/Systems (POWER6)

2. POWER6 Verification Experiences

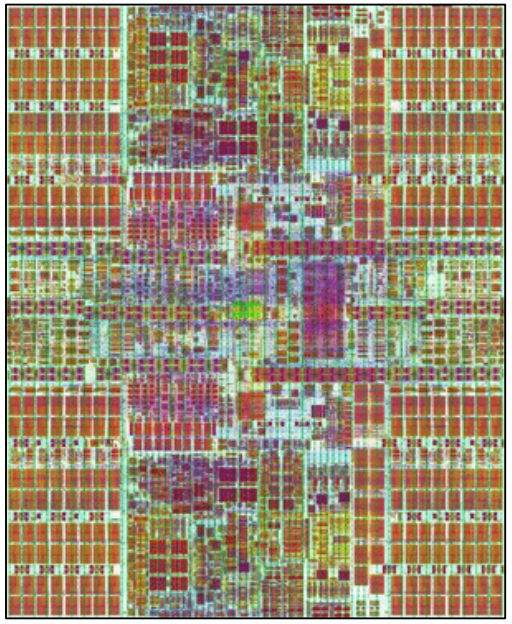
3. Crisis – What Crisis ?

4. Open Problem Areas and Unfulfilled Solutions

5. Conclusion

IBM High-End Server: new POWER6 microprocessor

- Topology:
  - Two cores on chip, a 2-way SMP.
  - Core private L1s (64KB I, 64KB D)
  - Superscalar, SMT cores
  - Chip private 8MB L2 cache
  - L3 32MB off chip
  - Two-tier SMP fabric



- Technology:
  - 65nm SOI
  - 341 mm^2 die size
  - 10 Layers of metal
  - 790 million transistors on chip
  - Frequency : 3.5, 4.2, 4.7 GHz

POWER6™ BUILT ON Power™

- Custom & Semi-Custom Design Style
  - High Frequency Constraints

3.3 M Lines of VHDL

IBM POWER SYSTEMS

Consistent Predictable Delivery

2001 POWER4

2003 POWER4+

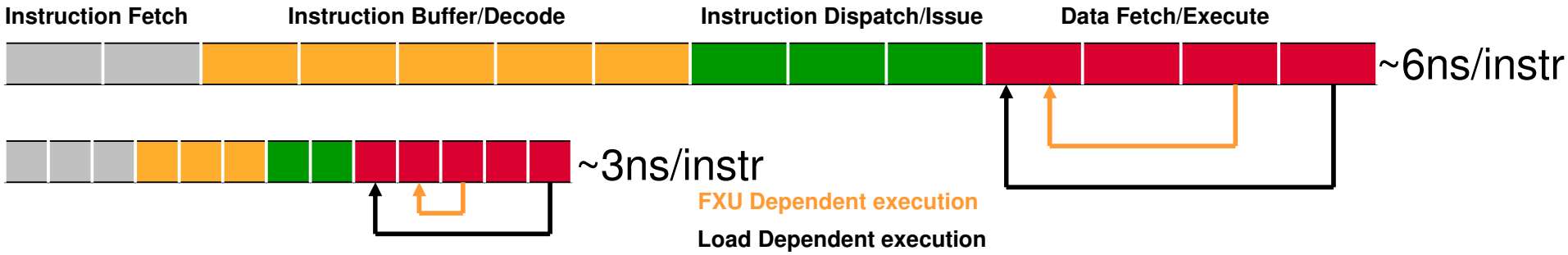
2004 POWER5

2006 POWER5+

2007 POWER6

POWER6 Core

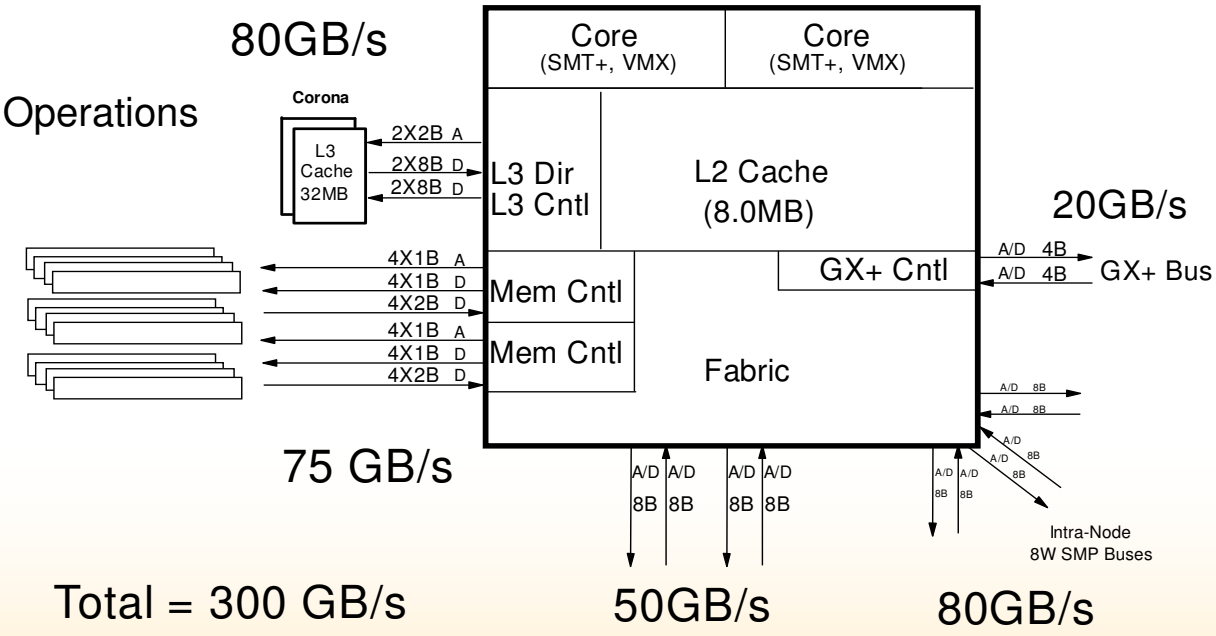
- **POWER6 processor is ~2X frequency of POWER5 (4-5GHz)**
- **POWER6 instruction pipeline depth equivalent to POWER5**
  - Minimize power
  - Scale performance with frequency



- **POWER6 Extends functionality of POWER5 Core**
  - 64K I Cache, 64K D Cache, 2 FXU, 2 FPU, 1 Branch execution unit
  - Two way SMT with 7 instruction dispatch from 2 threads (maximum of 5 instructions per thread)
  - Decimal Unit
  - VMX Unit
  - Recovery Unit

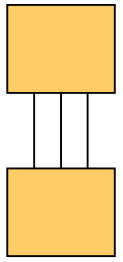
POWER6 scales chip capabilities with core performance

- **Cache highlights**
  - 4MB Private L2 Cache per Core
  - 32MB Non-sectored L3 Cache per chip
- **Fabric highlights**
  - Three Intra-Node SMP buses for 8-way Node
  - Two Inter-Node SMP buses for up to 8 Nodes
  - Multiplexed Address/Data SMP buses
- **New prefetching capabilities**
  - Coherent Multi-Cacheline Data Prefetch Operations
  - Prefetching on stores

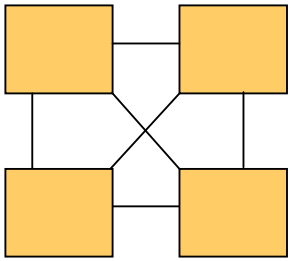


Flex System to optimize low end to high end server designs

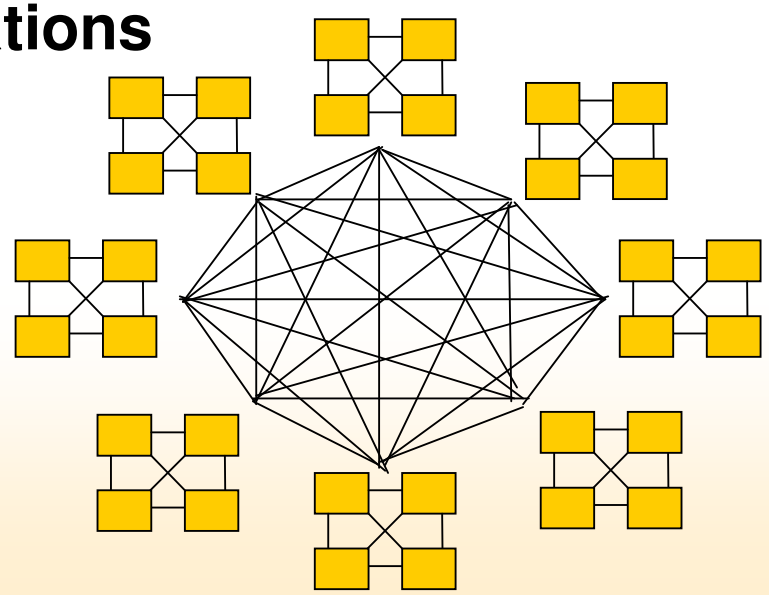
- **SMP busses can be configured in two modes**
  - Cost/performance trade-offs
  - On node busses are 8B or 2B
  - Off node busses are 8B or 4B
- **Numerous memory controller BW options**
  - 1 or 2 memory controllers are available
  - Memory controllers can be configured to full width or ½ width
- **L3 cache is supported in three configurations**
  - On module High Bandwidth configuration
  - Optional off module configuration
  - No L3 option
- **Fully interconnected two-tier SMP fabric**
  - Reduced latencies vs. POWER5
  - New two tier memory coherency protocol



2 socket



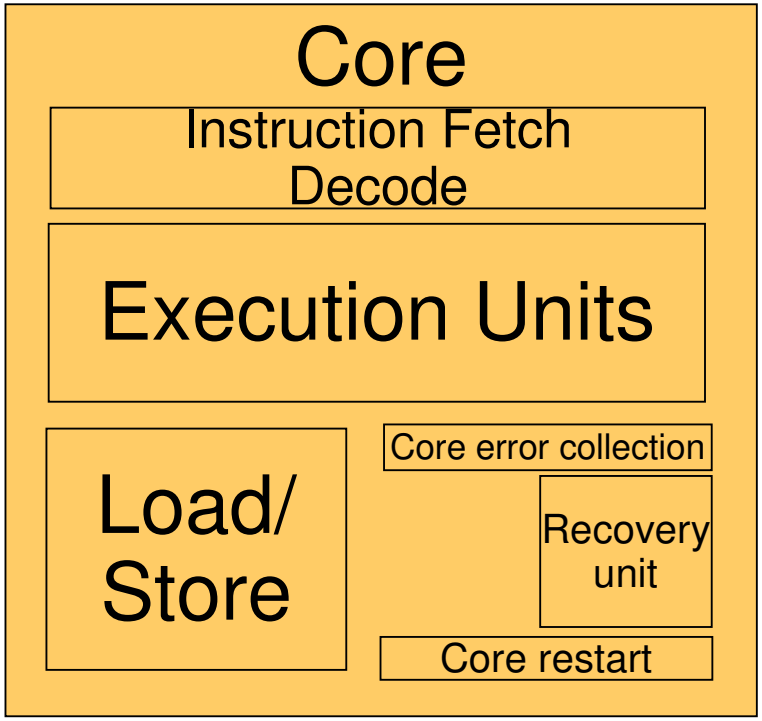
4 socket



32 socket / 64way SMP

Bullet-proof computing

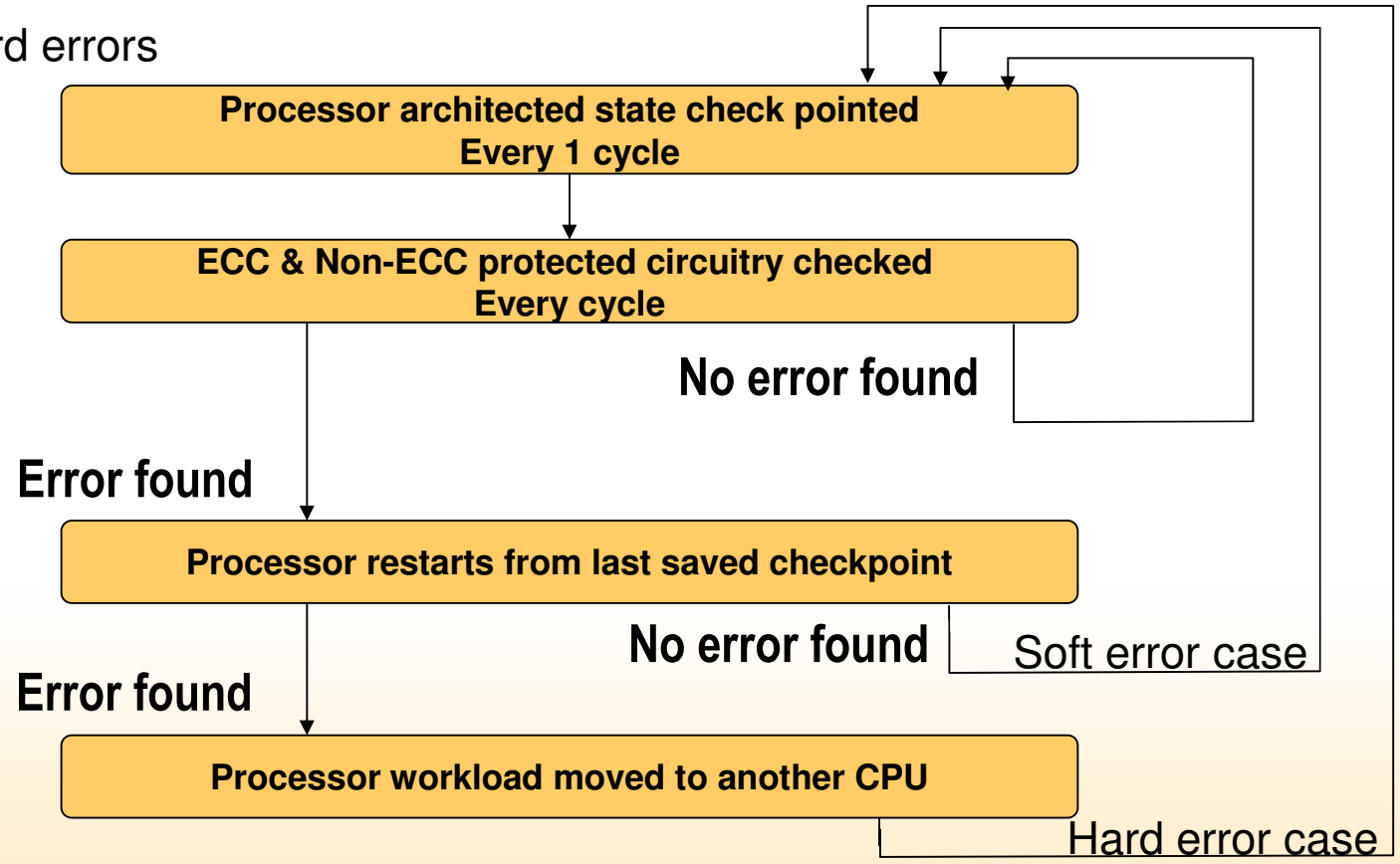
- **Recovery Capability**
  - Array error
    - Error correction (ECC)
    - Arrays with parity
      - Processor restarts
  - Instruction flow and Data flow Error
    - Processor restarts
  - Control Error
    - Processor restarts



- **System Resiliency**
  - Processor states are check pointed and protected with ECC
  - Processor states can be moved from one processor to another upon unsuccessful recovery restart

Bullet-proof computing

- **System reliability with recovery unit**
  - Every measure possible taken to preserve application execution
  - Retry soft errors
  - Change hardware for hard errors



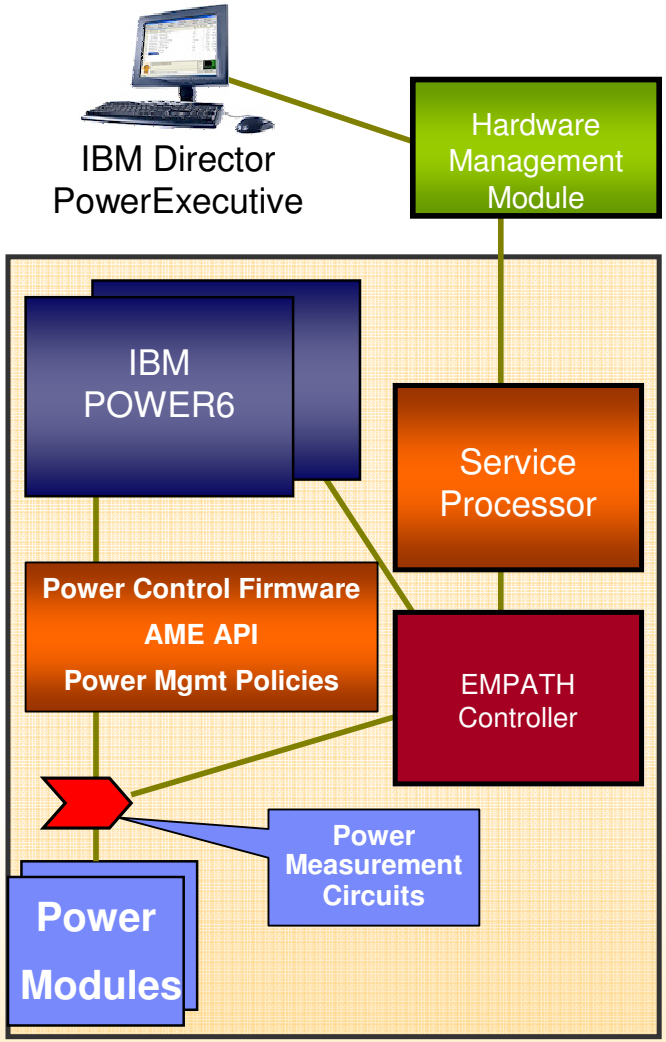
PowerExecutive Extensions for POWER6 Energy Management Policies

Example Energy Management Policies:

- **Energy cost management**
  - Monitor System workloads/power consumption
    - If: System utilization reduces reduce system power/performance
    - If: Multiple Systems go below utilization threshold consolidate workloads
    - If: System power budgets exceed allocation cap power
- **Acoustic optimization**
  - Monitor Systems temperature
    - If system temperatures go below threshold reduce fan speeds
- **Performance optimization**
  - Monitor system temperature/power consumption
    - If temperatures/power consumption go below threshold increase performance

Energy Management Policies Enable Customers To Maximize The Compute Capability Of Their Datacenter Or Minimize Energy Costs

Power6 EMPATH System Control
Extended System Functions For PowerExecutive Policies



- **Thermal / Power Measurement**
  - Read thermal data from processor chip thermal sensors
  - Measure power data from system level sensors
  - Report data via PowerExecutive
- **Power Capping**
  - Use of Hardware controls to keep system power under a specified limit
- **Power Saving**
  - Operation at reduced power when workload and policy allows
    - Can be a static policy (e.g. overnight reduction)
    - Can be dynamic (when absolute max performance is not always required)
- **System health monitoring**
  - Use of hardware sensors to ensure system is operating within safe predefined bounds
- **Performance-Aware Power Management**
  - Use of dedicated performance counters to guide power and thermal management tradeoffs

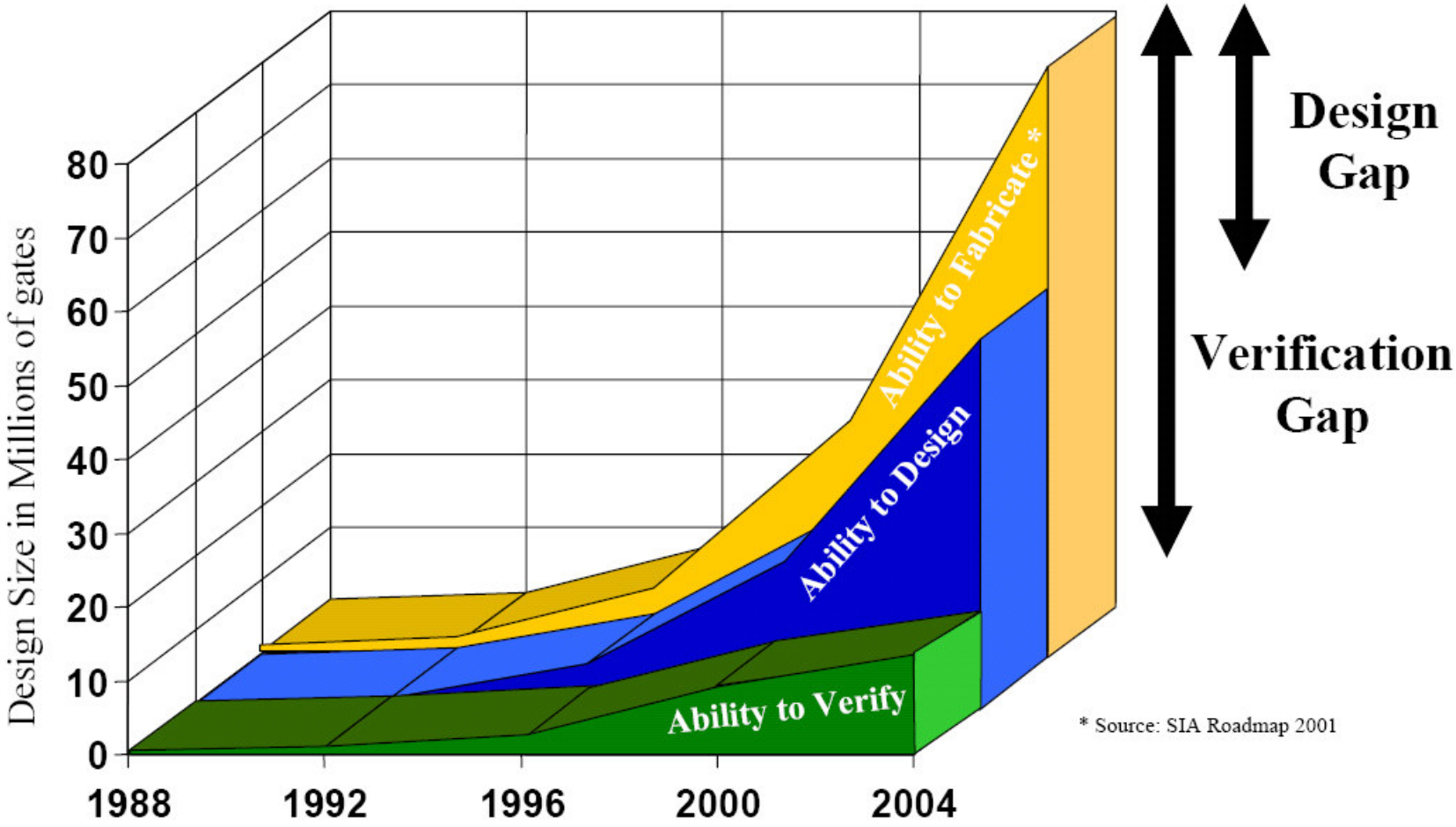
Power6 Summary & Conclusions

- **POWER6 doubles frequency and bandwidth of POWER5**
  - Same pipe depth
  - Same power envelope
- **POWER6 scales chip/system performance with core performance**
- **POWER6 provides new capabilities**
  - Decimal Floating Point
  - Processor recovery
- **POWER6 provides “mainframe”-like reliability for Unix platform**
- **System P will begin delivery of system power management with POWER6**
- **POWER6 systems shipping since mid 2007**

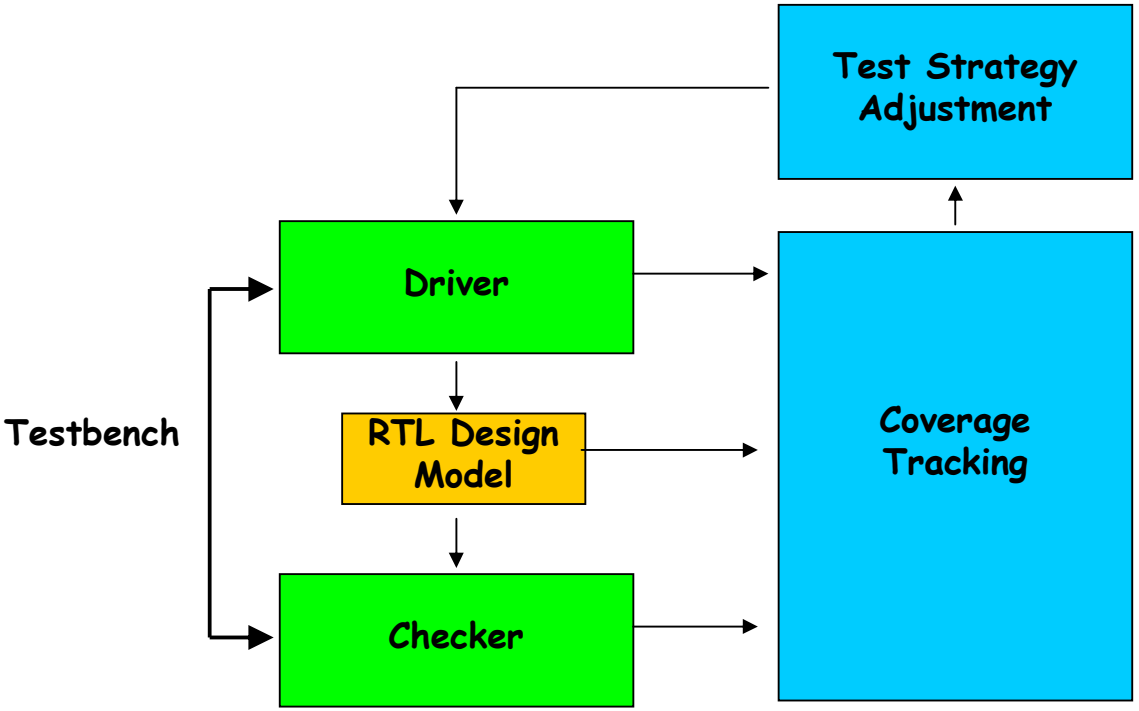
Overview

1. Context : High-End Server Micros/Systems (POWER6)
2. **POWER6 Verification Experiences**
3. Crisis – What Crisis ?
4. Open Problem Areas and Unfulfilled Solutions
5. Conclusion

Crisis Prediction – How did we do POWER6 ?

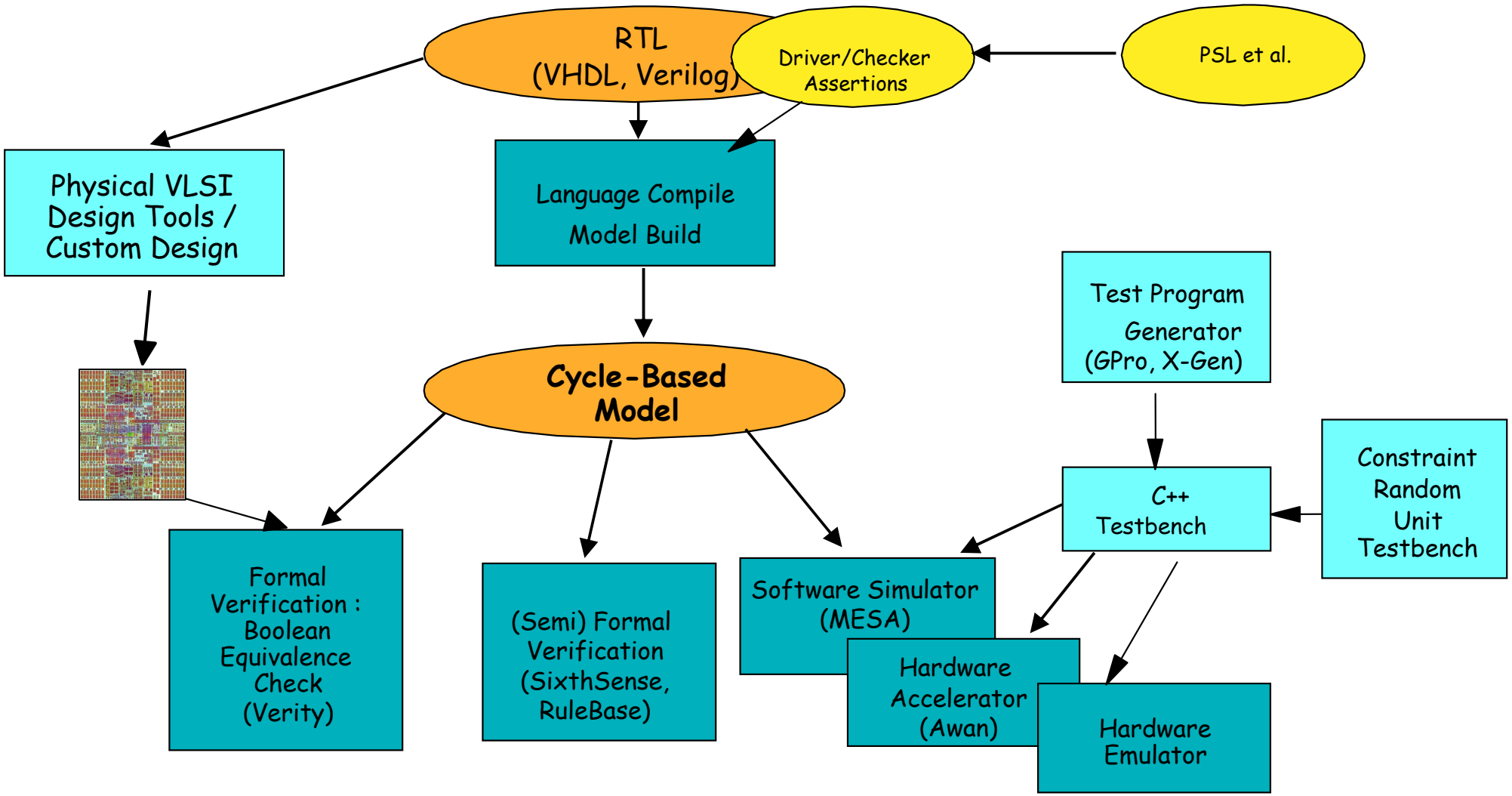


POWER6 Verification Process

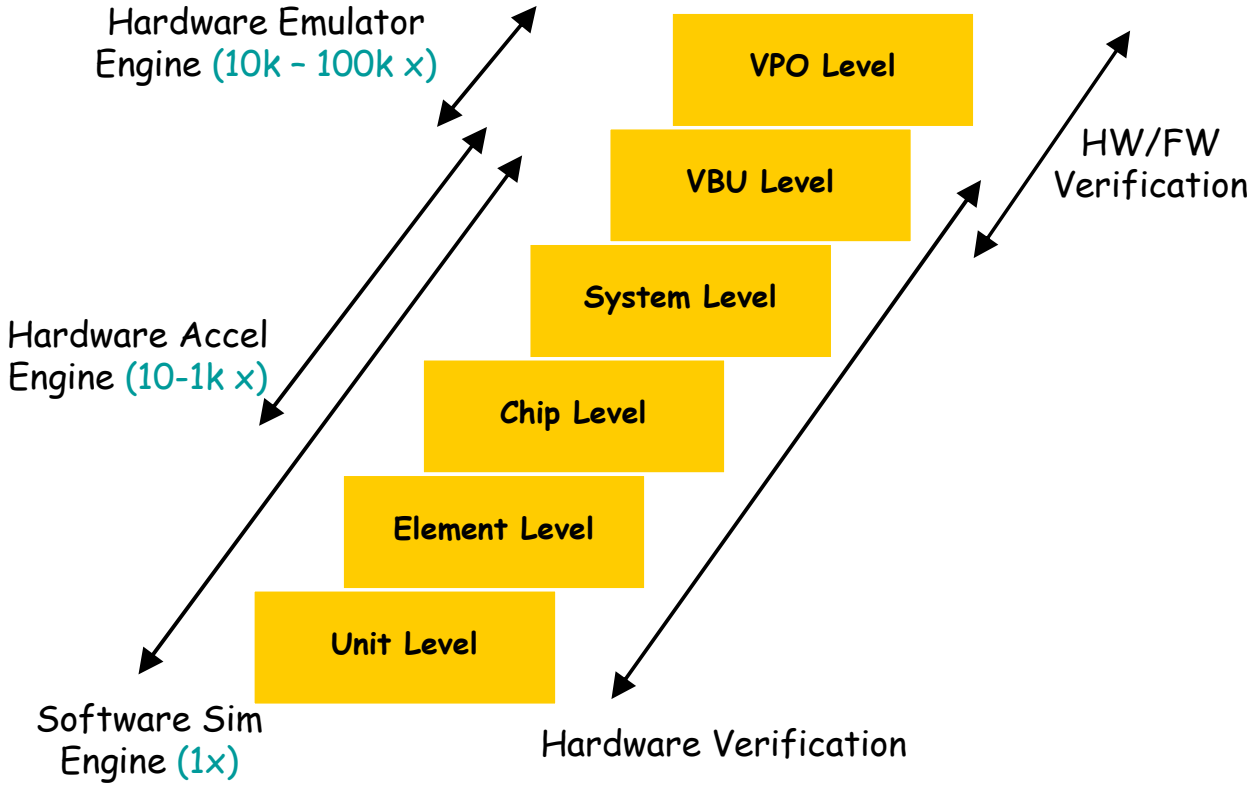


- **The different verification engines have different strengths related to the verification tasks**
- **Software simulation**
  - Slow, but low penalty for highly intrusive checking of model internals. Total model visibility.
- **Hardware-accelerated simulation**
  - Faster, but need less intrusive driving/checking to not slow down hardware box.
- **(Semi)-Formal verification**
  - (High to) Exhaustive coverage, but higher skill needed to drive. Scaling problems w/ model size.
- **Hardware Bring-Up**
  - Ideal speed, very limited visibility/controllability

POWER6 RTL Verification Technology



POWER6 Scaling the Simulation-Based Methodology



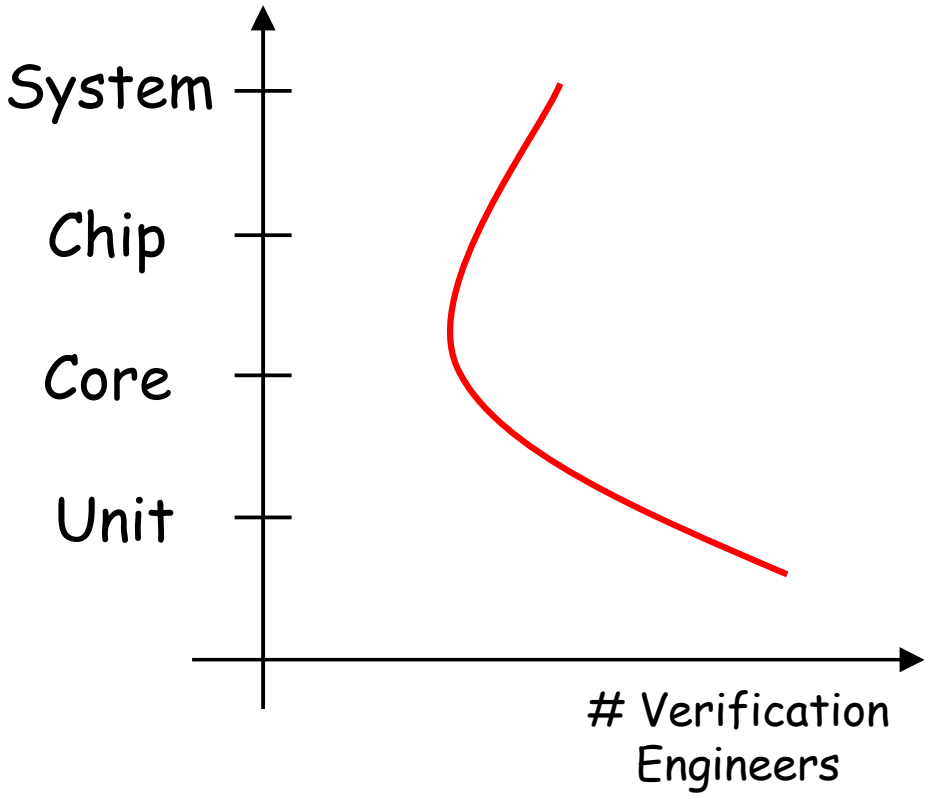
Gen: POR + Firmware + O/S Boot Check: pre-calc exp. Results
Gen: POR + "bare-metal exerciser" + Linux Check: pre-calc exp. Results
Gen: random-biased test pgm + rand. irritation Check: some checker re-use + arch rules check
Gen: random-biased test pgm + rand. irritation Check: re-use element check + arch rules check
Gen: random-biased, transactions, test pgm Check: re-use unit check + pre-calc exp. results
Gen: random-biased, on-the-fly Check: detailed, score-boarding, rules

VBU = Virtual Bring-up (chip)
VPO = Virtual Power-On (system)

Scaling the Simulation-Based Methodology (cont.)

- **Success Factors:**
  - **Simulation Technology scales well**
    - cycle-based streamlined algorithms
    - super-linear scaling with "parallel instance support"
    - large workstation "farms"
    - Hardware engines for acceleration/emulation
  - **Layered Re-use of checking components**
    - "vertical re-use" -> major productivity
    - unit->core->chip->system
  - **Sophisticated generation technology**
    - simple constraint-based random at unit level
    - constraint-based, on-the-fly transaction generation for data-mover units (e.g. memory sub-system)
    - heavy constraint-based test program generation at core/chip/system
  - **Coverage Technology**
    - Heavy emphasis on functional coverage

Staffing



- **Unit Level :**
  - Interface-specific checking
  - Intrusive checking
  - Implementation dependent
  - Lots of code (> 1M LOC C++)
  - Score-boarding
    - accumulate state in data structures
  - Sophisticated drivers (constraint random)
- **Core Level :**
  - Integrate unit level checkers
  - Re-use drivers or use architecture testgen
- **Chip Level :**
  - Integrate cores + select/re-use
  - Architecture testgen
- **System Level :**
  - Big environment integration job (several chip env)
  - Architecture testgen

Formal Methods

- **Formal Methods are a vital complement to simulation flow:**
  - Abstract Coherency Protocol verification
  - Equivalence Checking (Boolean & Sequential)
    - almost total elimination of gate-level verification
  - Coverage Reachability Analysis
  - RTL - Special Focus areas
    - sub-unit level, macro, multi-macro
    - designer-assertions and FV-expert testbenches
  - FPU – unit data-path verification
  - Lab Bring-up Bug re-creation
    - often faster repro
    - high-coverage/proof of side-effect-free fixes

POWER6 Verification Results

- Successes:
  - Methodology scaled well even for POWER6-size project
  - POWER(4/5/)6 systems booted O/S with first pass hardware
  - Verification was never the single bottle-neck for the project
  - <2% of design bugs found post-silicon
  - RTL abstraction worked well, even for full custom chip design style
  - Very repeatable, disciplined, verifiable methodology
  - Extension of verification with (S)FV helped in many areas of critical complexity

1. Context : High-End Server Micros/Systems (POWER6)

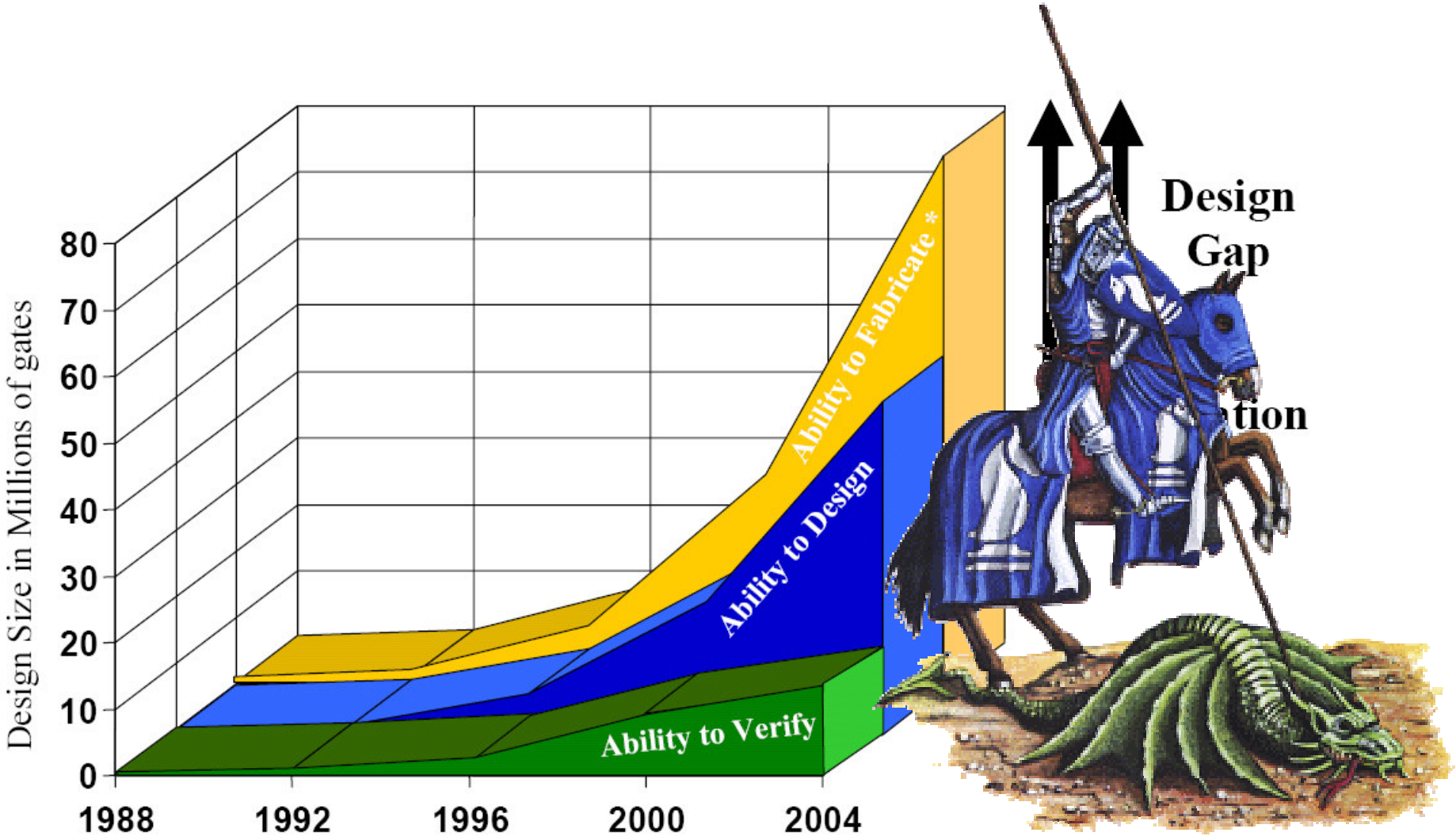
2. POWER6 Verification Experiences

3. **Crisis – What Crisis ?**

4. Open Problem Areas and Unfulfilled Solutions

5. Conclusion

Crisis Prediction – Is the dragon dead ?

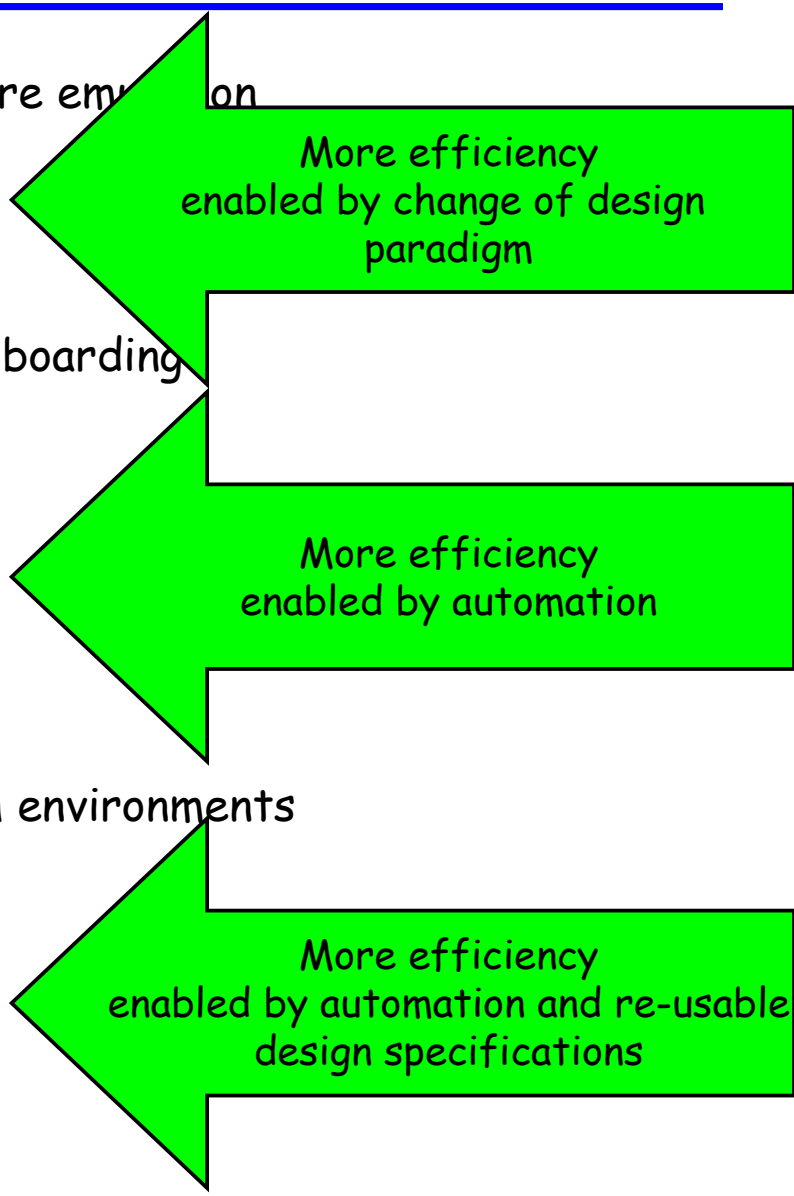


Key Verification Innovations that Avoided the Crisis So Far (1)

- Fast RTL simulation - hardware acceleration – hardware emulation
- Cycle-based execution
- Boolean Equivalence Proof

- End-to-end checking on-the-fly – golden model – score-boarding
- Grey-box checking on-the-fly
- Constrained-random generation
- On-the-fly constrained-random generation
- On-the-fly constrained-random transaction generation
- Constrained-random libraries
- High-level verification languages
- Re-use methodology for constrained-random testbench environments

- Instruction-stream test generators
- Multi-instruction-stream test generators
- System transaction-stream test generators



Key Verification Innovations that Avoided the Crisis So Far (2)

- BDD-based formal verification
- Multi-engine formal verification
- Semi-formal verification

More efficiency
enabled by deeper coverage
with new algorithms

- Structural coverage collection
- Functional coverage collection

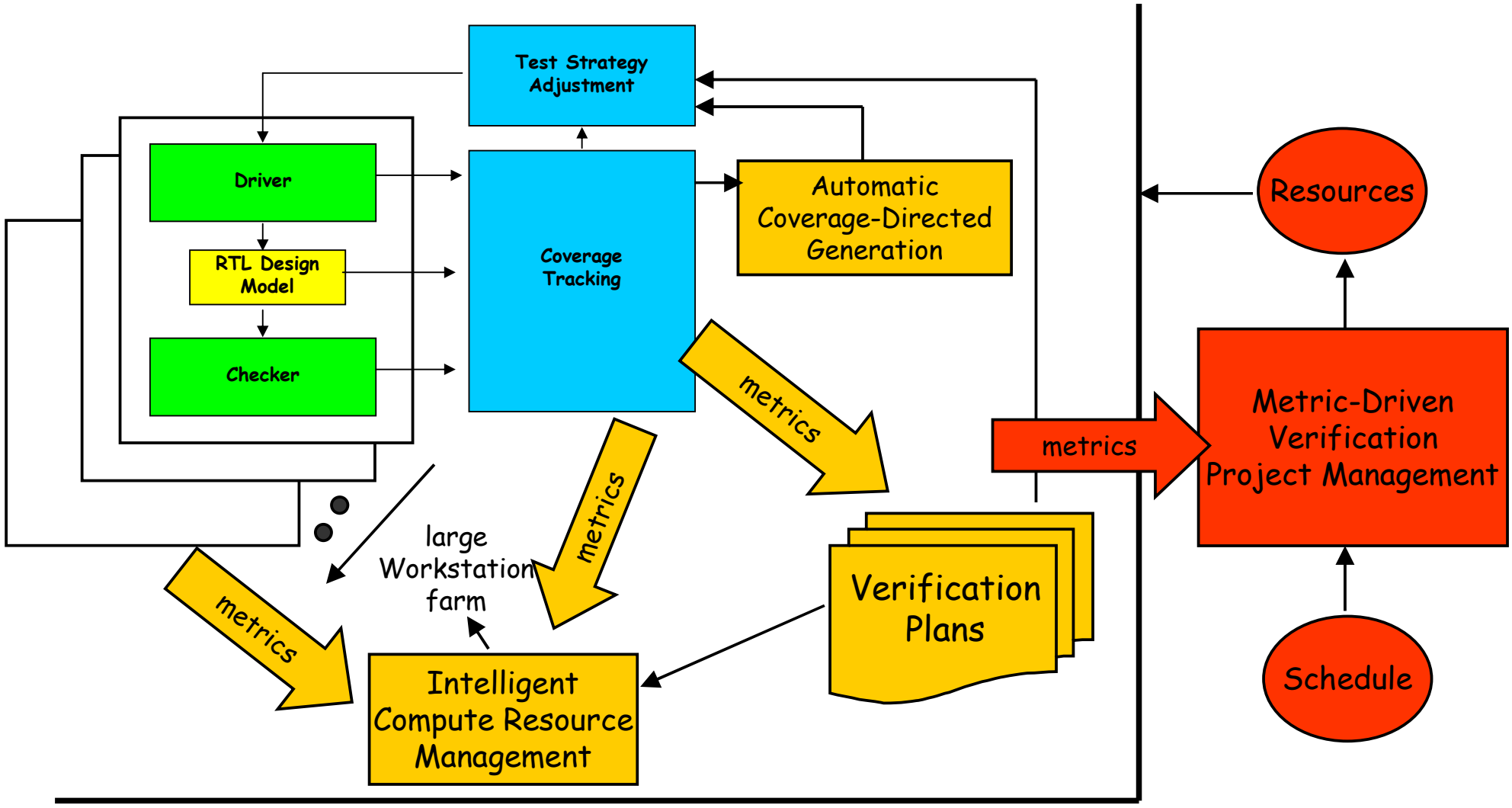
More efficiency
enabled by coupling verif process with
deeper semantic insight into design

- Assertion languages
- Property specification languages
- Assertion-based verification

More efficiency
enabled by shifting error detection
closer to the design(er) source

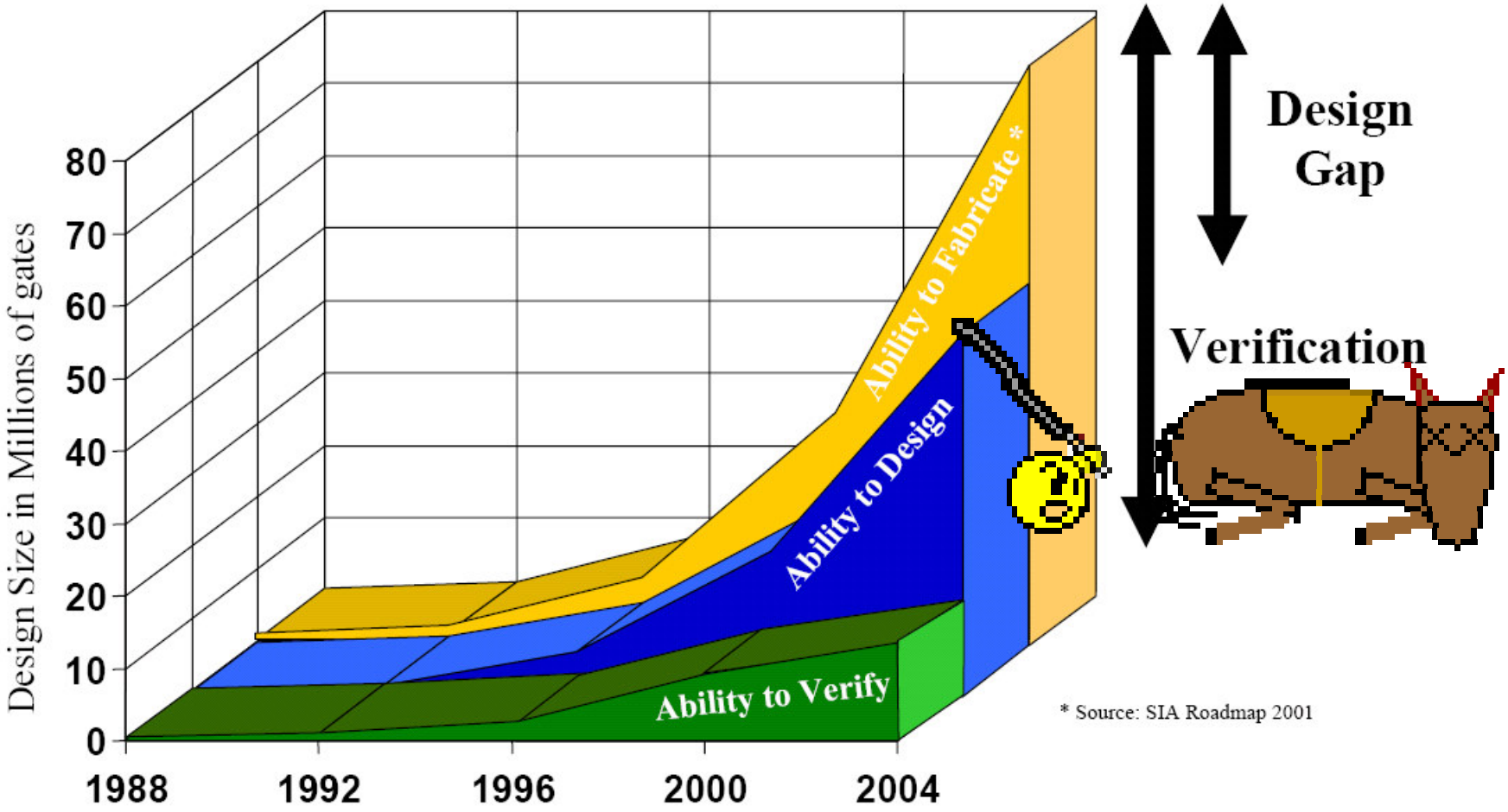
> Everybody seems to have arrived at more or less the same methodology
> The difference is in the strength of algorithms and implementations

Current Industry Trends in the Quest for More Efficiency



Verification turns from a service to a first-class production discipline

Crisis Prediction – Or is the horse dead ?



- 2006 Update :

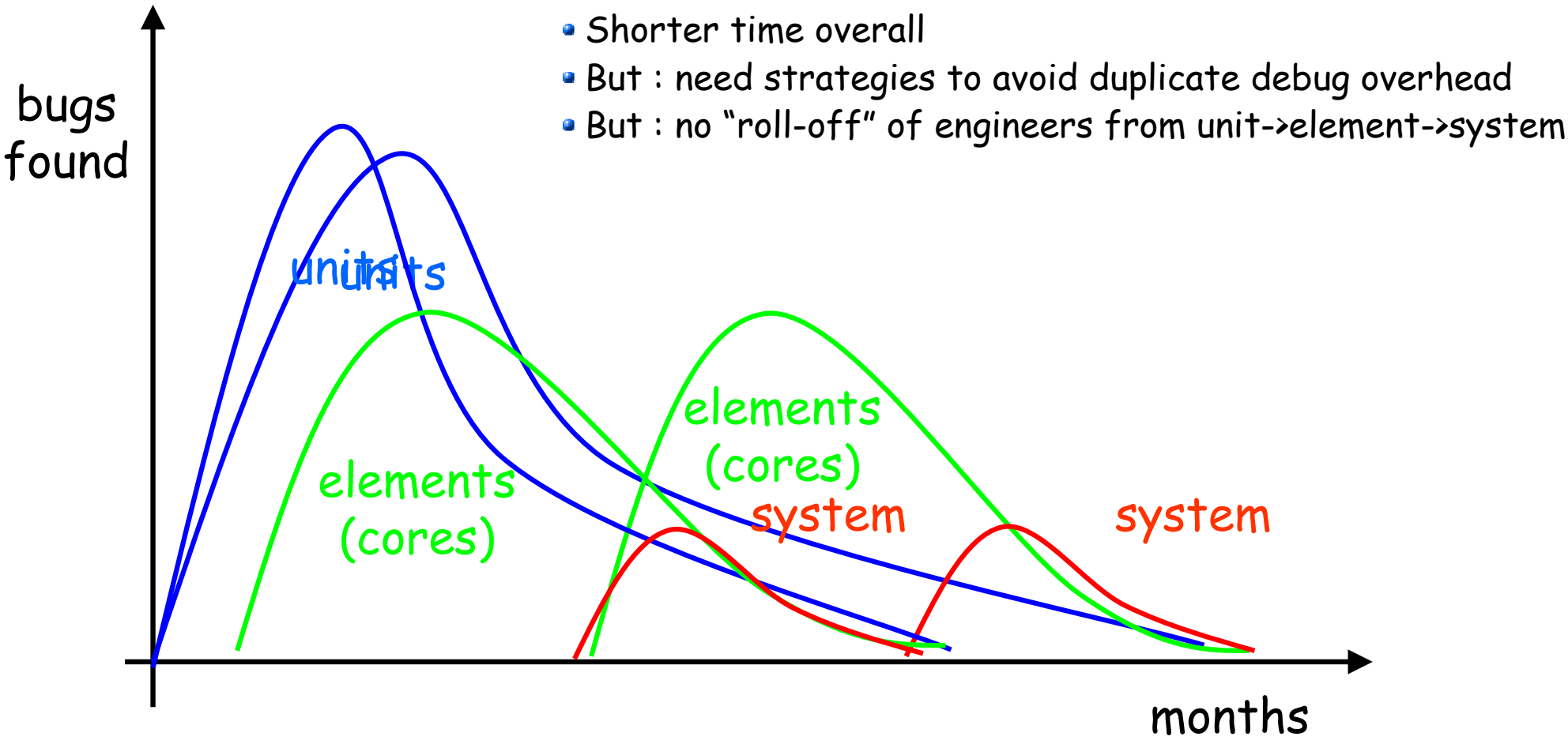
The biggest issue in design verification is that all currently known algorithmic solutions are running out of capacity with respect to the designs being developed today. The only foreseeable way to overcome this issue in the short term is with an adequate verification methodology.

Many challenges are still to be solved to obtain a sufficiently robust and complete methodology. For instance, there is a need for ways to obtain consistent abstraction techniques of design components, interfaces, etc., that do not drop key aspects of the design in the abstraction

1. Context : High-End Server Micros/Systems (POWER6)
2. POWER6 Verification Experiences
3. Crisis – What Crisis ?
4. **Open Problem Areas and Unfulfilled Solutions**
5. Conclusion

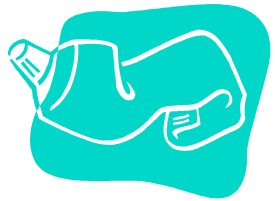
Open Problem Areas (1) : Schedule Compression

- Parallelization of work
  - Shorter time overall
  - But : need strategies to avoid duplicate debug overhead
  - But : no "roll-off" of engineers from unit->element->system



Open Problem Areas (2)

- **Areas of heavy complication during POWER6 verification**
  - Error Detection and Soft Error Recovery
    - the design has lots of error detection logic
      - easy for data paths (parity, ECC)
      - ad-hoc for control logic
    - multitude of error recovery schemes
      - centralized – checkpoint/restart from recovery unit
      - decentralized – error check/correction
    - **Verification strategy** : dynamic on-the-fly error injection
    - **Problems:**
      - myriads of injection points in almost any "mainline state" of the design
      - end-to-end spec easy to check (arch. correctness; despite recovery)
      - unit-level specification is extremely complex and always in flux
      - huge effort to make testbench code robust
        - recovery acts like an **exception** for the testbench code
        - invalidate/re-sync data containers, score-boards etc.



Open Problem Areas (3)

- "Pervasive" functions / Non-mainline functions
  - pervasive functions are
    - sets of registers
    - sets of access mechanisms
  - used to configure various system capabilities and read system state during debug activities
  - debug control, trace arrays
  - Core sparing, fencing
    - support for partition/job migration
  - pervasive capability is highly optimized to limit impact on area/timing
  - pervasive and DFT functionality are often interleaved
  - specification late, in-flux and hard to keep complete

  - **Verification strategy:**
    - specialized unit/chip/system test environments
    - highly customized mix of structural, (semi-)formal and simulation environments
  - **Problems:**
    - mainline test benches and pervasive environment are developed in parallel
      - very costly to build & run efficient integrated cross-checking environment
    - hard to cover all unwanted side-effects of pervasive function to "mainline" state

Open Problem Areas (4)

- Power Save Modes
  - clock gating
    - treated as normal functional mode in all verification
  - sleep, nap, doze
  - new challenge : functional power gating
    - treating as normal functional mode has very questionable coverage
    - wake-up is mini-POR
  - **Verification strategy**
    - combination of structural checking, simulation and formal techniques
    - randomized injection to validate function of safety overrides ("disables")
  - **Problems:**
    - myriads of combinations of design partitions powered up/down
    - enforcement of strong design rules for verification very expensive for design
      - e.g. require fully-defined state of re-powered partition

Open Problem Areas (5)

- Configurations
  - Designs increasingly support many different configurations
  - Part of the re-usability drive (see below)
  - How to specify & verify configurable aspect of design?
  - How to predict & verify all possible usage scenarios of a configurable design apriori
  - **Verification strategy**:
    - randomize / non-deterministic configuration settings
  - **Problem:** coverage

Unfulfilled Solutions (1)

- Re-use in SoC design resolves the verification crisis
  - pre-verified design IP
  - re-usable verification IP
  - ITRS 2006 predicts need of 3rd party verif IP re-use to grow from
    - ~17% in 2007
    - ~43% in 2016
    - assessment is there are no known solutions to attain this percentage
      - key issues:
        - how to rigorously/completely describe abstract behavior of IP
        - how to generally specify environment constraints assumed
        - how to exploit hierarchy to simplify verification (see "i/f compliance vs. interoperability
  - Will slow-down of Moore's Law cause a bifurcation ?
    - ASICs turn into SoC
    - High-margin/performance designs will even more aggressively optimize unique implementation
  - Interface compliance does not necessarily imply interoperability
    - transport-level vs. protocol level

Unfulfilled Solutions (2)

- New language "x" will resolve the verification crisis

1. New languages sometimes can provide significant productivity gains
2. New languages always provide job-security for tool developer (companies)

- New languages that focus more on 2.) in balance, are inhibitors of progress in the industry
  - Switch of language platform creates artificial problem that is predictably solvable
  - Drain of innovation bandwidth
  - Work hard, pay money to get back where we started

Unfulfilled Solutions (3)

- Higher levels of abstraction will resolve the verification crisis
  - High-level modelling
  - ESL
  - Various languages … again
  - Industry-wide search > 10 years
  - **Why is this hard ?**
    - This is a multi-discipline problem – one-dimensional optimization is **bad**
    - Verification is only one concern of several (performance, power, timing, yield)
    - Automation of implementation of all other concerns **ex** elusive
      - Specific optimization is a key differentiator

Law of leaky abstractions

Joel Spolsky, "Joel On Software"

The Law Of Leaky Abstractions (1)

- "**All non-trivial abstractions, to some degree, are leaky**." – Joel Spolsky
- When abstractions fail, the underlying, hidden structure breaks through
- **Bad abstractions** neglect or suppress relevant areas of concern
  - Hiding a relevant area of concern, i.e. making it inaccessible lead directly to a leak
- Examples:
  - High-frequency design
    - disregard of physical partitioning in high-level design leads to severe problems in timing closure
  - Low-power design
    - leakage power, dynamic power estimates need to be part of high-level design – defines granularity of a model
  - Design for verification
    - limit complexity of implementation (e.g. async domain crossing interfaces) to simplify verification space

The Law Of Leaky Abstractions (2)

- High-level design is a multi-dimensional, multi-disciplinary set of **engineering trade-off decisions**
- A high-level modelling abstraction must be able to express constraints for all relevant **dimensions of** the trade-off solution space

- Remember the **2006 ITRS Update** :

Many challenges are still to be solved to obtain a sufficiently robust and complete methodology. For instance, there is a need for ways to obtain consistent abstraction techniques of design components, interfaces, etc., that do not drop key aspects of the design in the abstraction

Unfulfilled Solutions (6)

- Post-silicon Verification
  - Post-silicon hardware is the fastest simulation platform possible
  - Problem : how to apply pre-silicon "science" to post-silicon
    - significant shift of constraints:
      - extremely long tests preferred
      - visibility into "model" very limited
      - coverage collections?
      - debug?
  - design for post-silicon verification (a la DFT) ?
    - what are the regular, generic hardware structures to support the post-silicon verification effort?

Overview

1. Context : High-End Server Micros/Systems (POWER6)
2. POWER6 Verification Experiences
3. Crisis – What Crisis ?
4. Open Problem Areas and Unfulfilled Solutions
5. **Conclusion**

Conclusion

- The verification field can take credit for avoiding the predicted major verification crisis with a long series of practical innovations
  - Don't let the decade-old crisis talk overshadow significant innovation breakthroughs
- The uniform marketing & methodology talk across the industry indicates a level playing field
  - Harder to assess specific strengths of algorithms and implementations
- Metric-driven verification project management is not the endpoint of maturation of verification technology
- In the post 32nm era
  - verification advancements need to address many **non-mainline** problems
  - verification research cannot afford dead-ends in **leaky abstractions**