Model checking in the cloud

VIGYAN SINGHAL

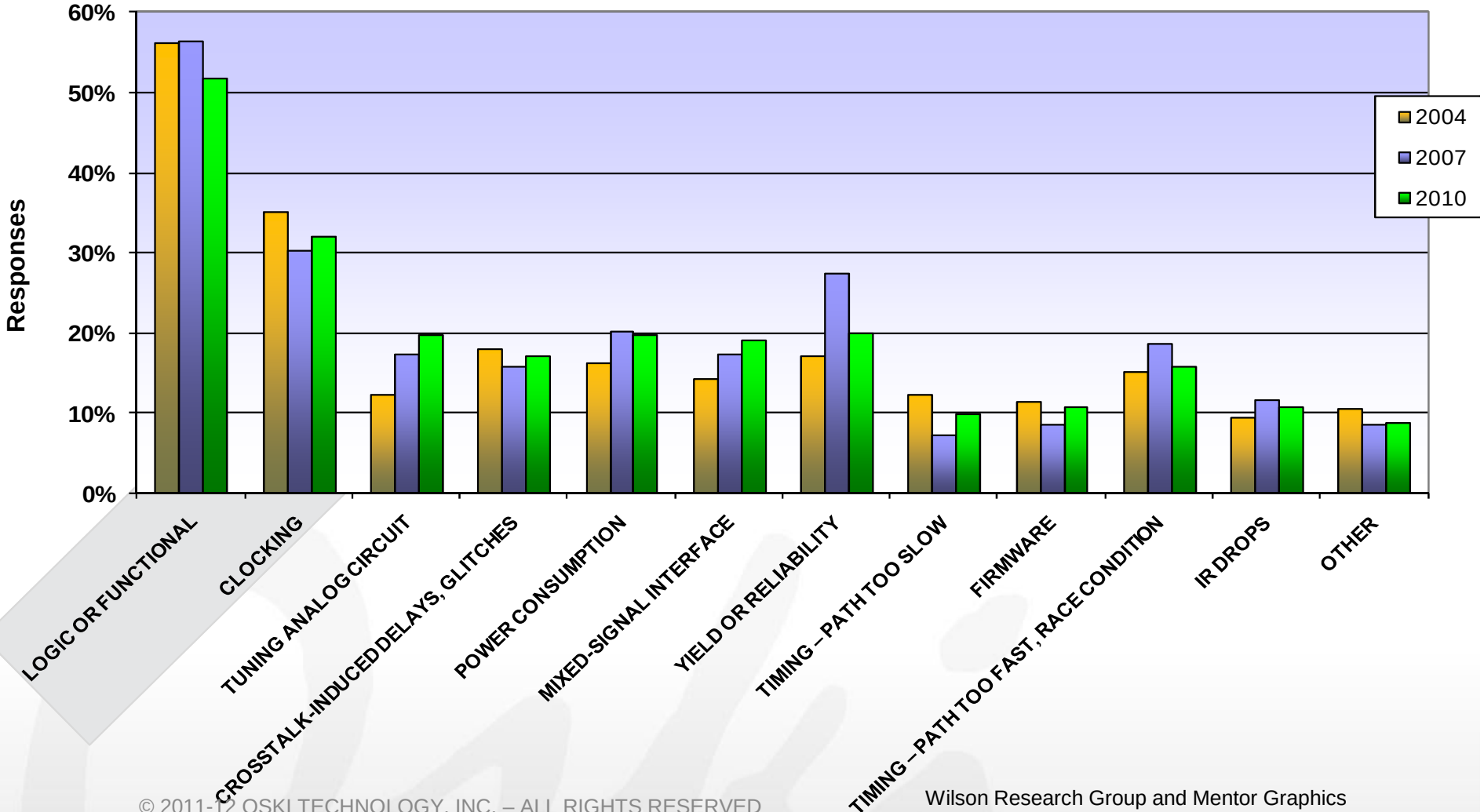
OSKI TECHNOLOGY

Views are biased by Oski experience

- Service provider, only doing model checking
  - Using off-the-shelf tools (Cadence, Jasper, Mentor, OneSpin Synopsys)
    - Have built in the past (UC Berkeley, Cadence, Jasper)
  - 15+ full-time model checking users
  - Customers like NVIDIA, AMD, Cisco, Huawei, Synopsys, Xilinx
- Most projects are set up as milestone-based
  - Milestones have to show value in a simulation-based plan
- Have to fit in with the chip schedule
  - Predicting the user and tool run-times is a requirement
  - Hope (a.k.a “bug hunting”) is not a strategy

Types of post-silicon flaws

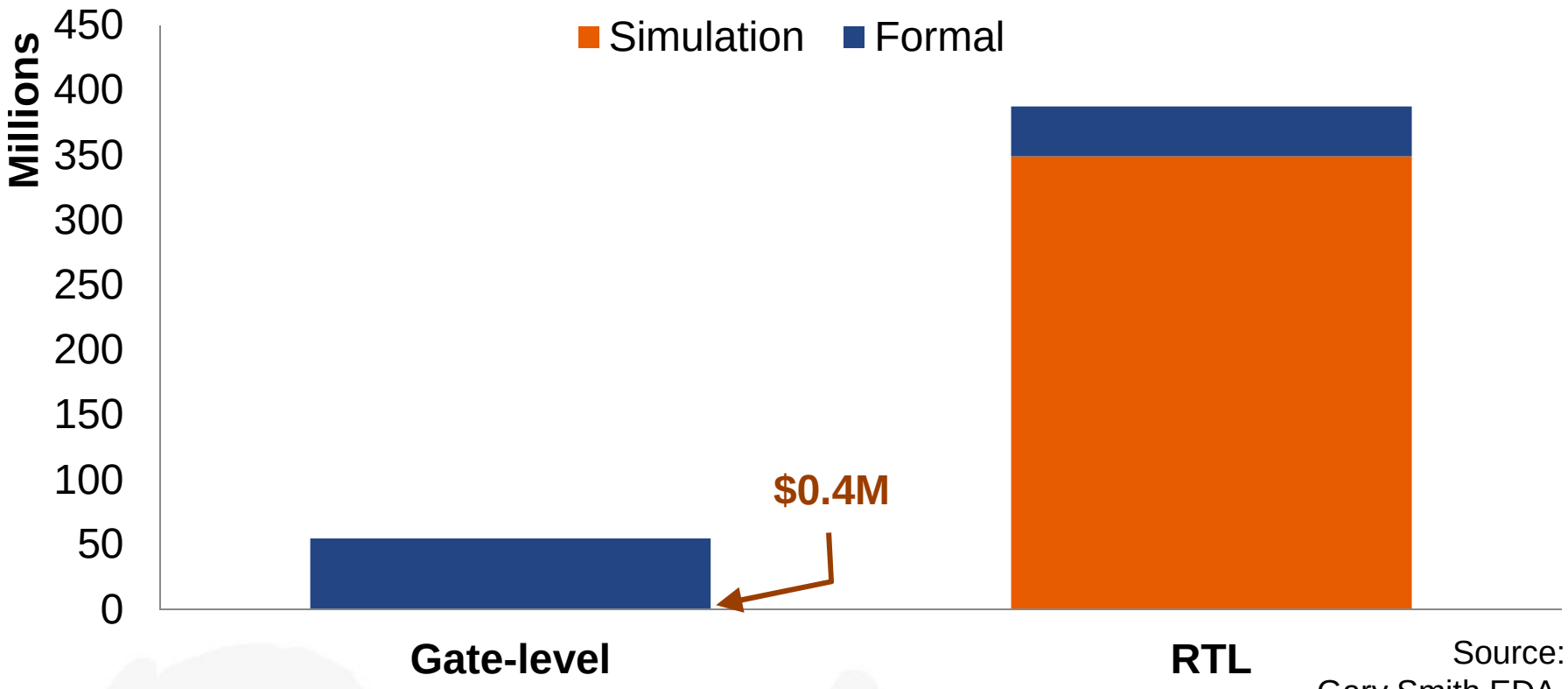
Verification is the still the largest problem



Wilson Research Group and Mentor Graphics
2010 Functional Verification Study. Used with permission.

Verification market size (2009)*

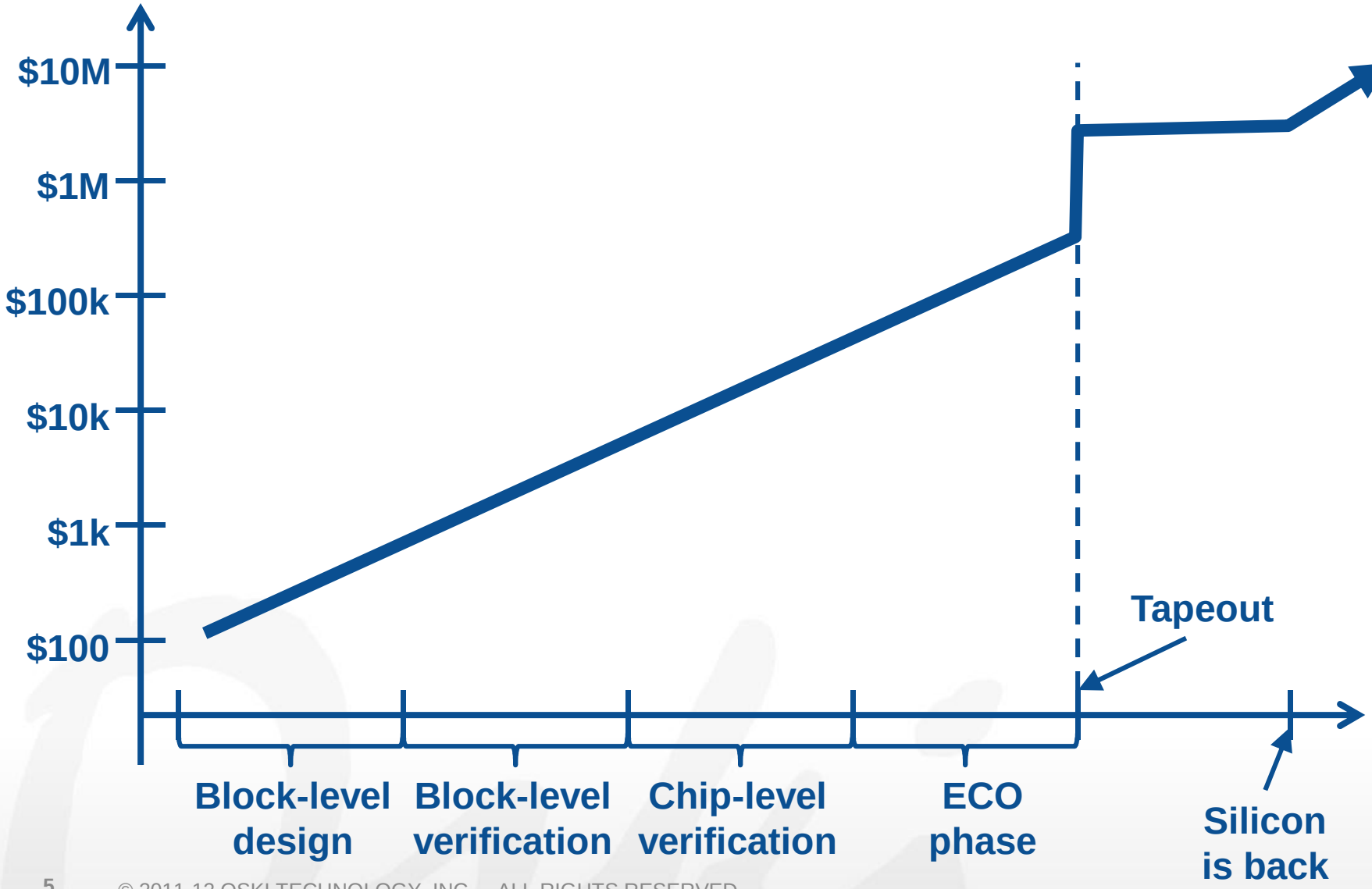
* excluding analog



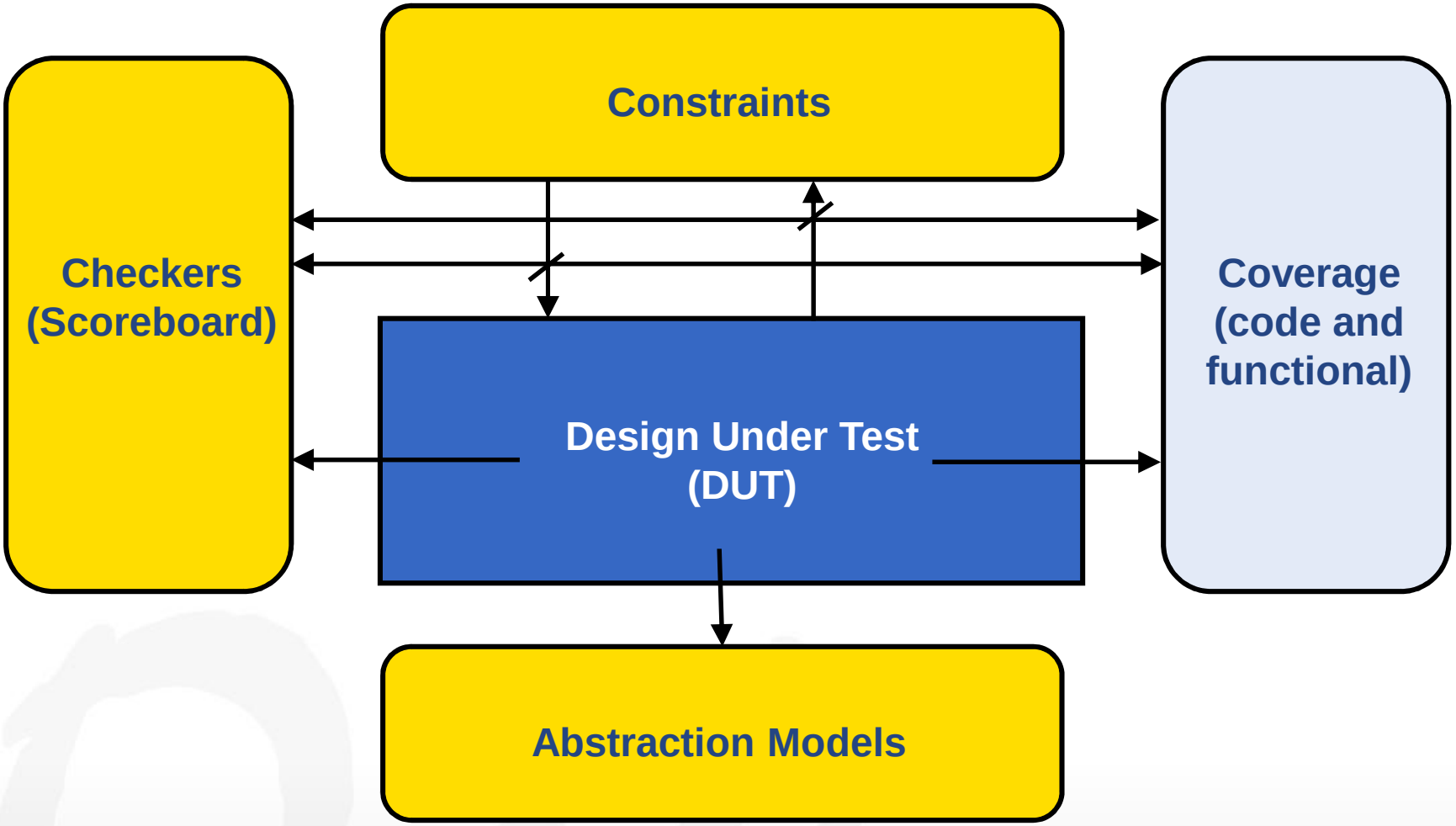
Source: Gary Smith EDA, October 2010

- **Gate-level formal (equivalence checking)**
  - **Then (1993):** Chrysalis; **Now:** Cadence, Synopsys
- **RTL formal (model checking)**
  - **Then (1994):** Averant, IBM; **Now:** Cadence, Jasper, Mentor, OneSpin, Synopsys

Motivation: exponential rise in bug-fix cost

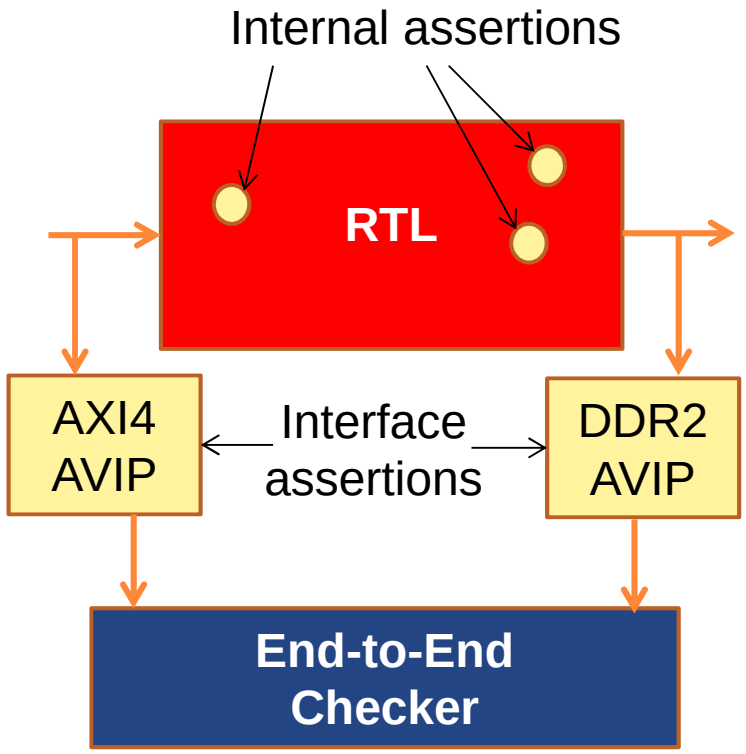


A model checking testbench



Cloud applicability depends on what you check

- Internal assertions, automatic checks
  - Relate RTL internals, embedded in RTL
    - E.g. “sm[7:0]” is one-hot
  - X-propagation, clock gating checks
  - Many, usually easier
- Interface assertions
  - Relate I/Os on one interface
  - E.g. valid-ack, AMBA AXI4
  - Fewer, harder
- End-to-end checkers
  - Models end-to-end functionality
  - Replaces simulation
  - Often requires manual abstractions



Where is the bar (for end-to-end formal)?

- Formal has to be more cost-effective than the alternative
- Usually bounded proofs are good enough

  (if bound is good enough!)
- Need to commit to what can be verified (and not), up front
  - Backed by “Coverage” (measurable and/or argumentative)

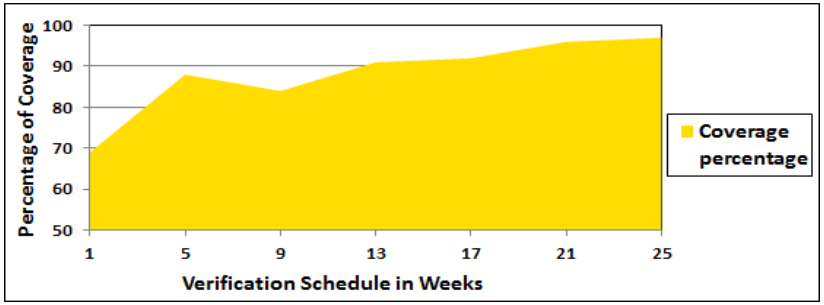
Am I done with model checking? (three C's)

- Is my list of Checkers complete?
- Are my Constraints not over-constrained?
- Is my Complexity strategy complete?
  - (are my proof bounds good enough)

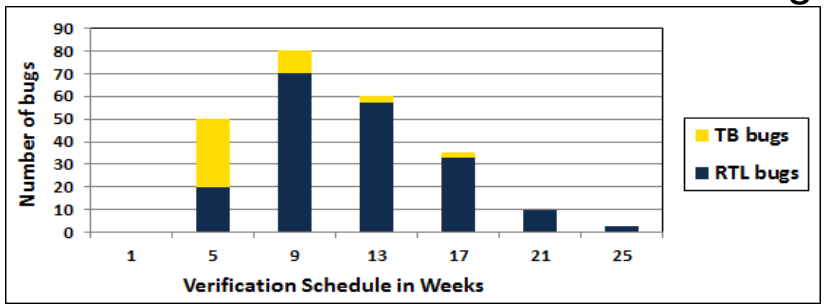
"Coverage" is the missing link

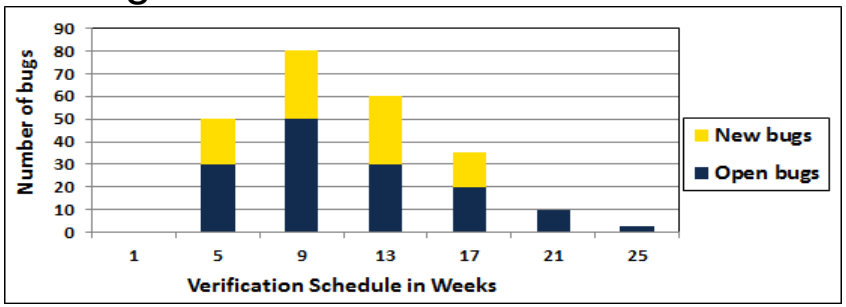
Verification manager's dashboard

Coverage tracking

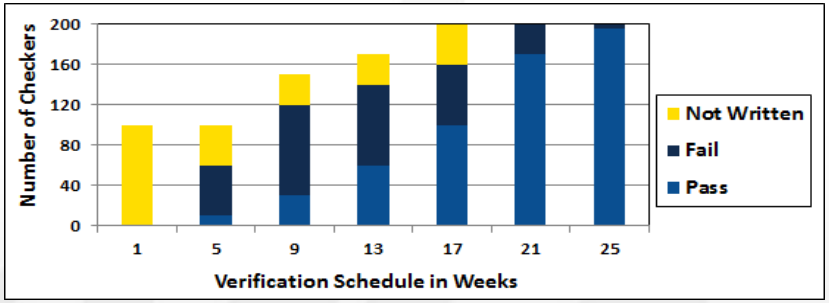


Bug tracking

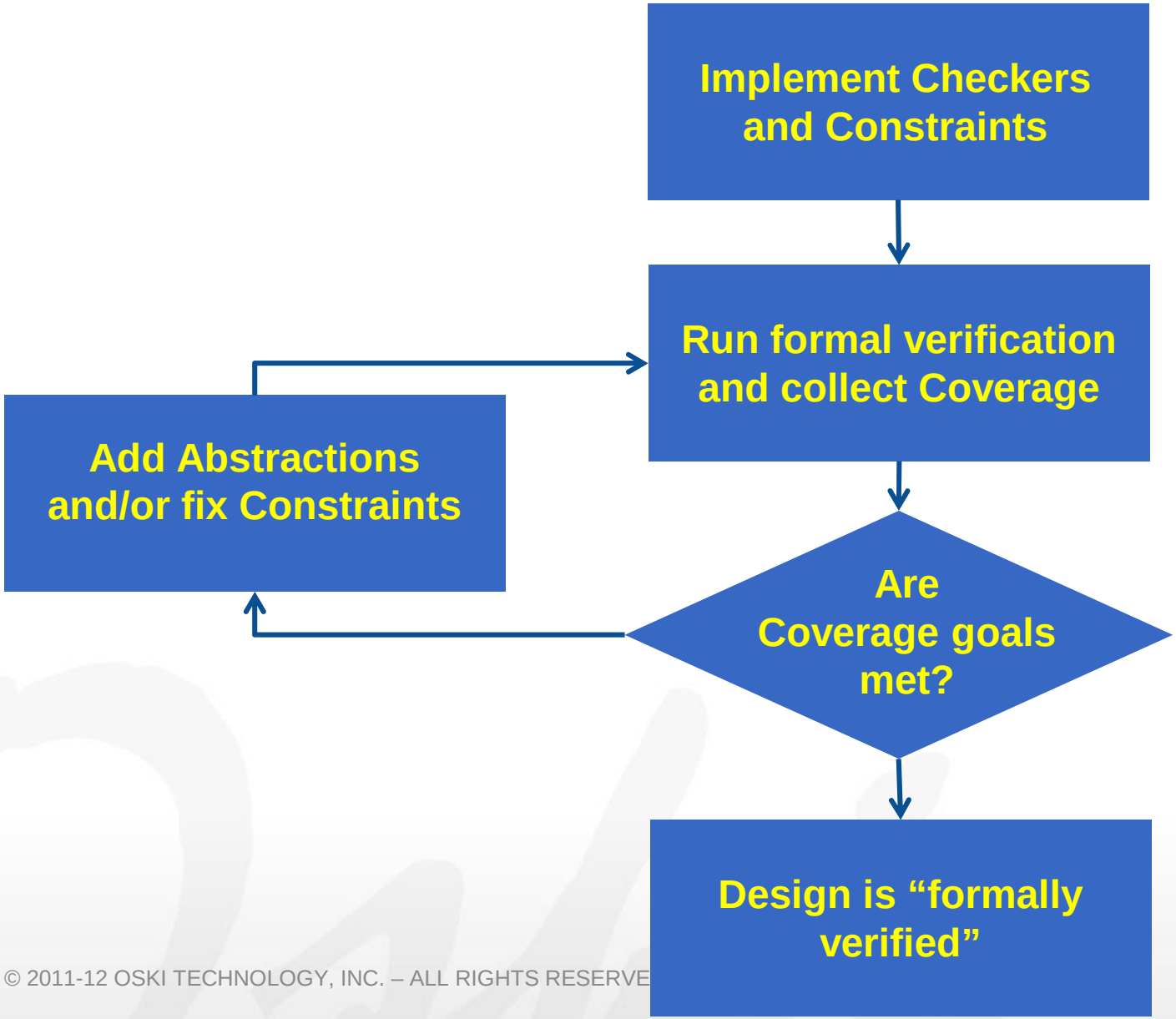




Runtime status



Model checking with coverage



Cloud can help in later stages

- Early stages (user intensive, not parallelizable)
  - Building constraints
  - Build abstractions
  - Debugging first checker failures
  - Building multiple checkers
- Later stages (machine intensive, parallelizable)
  - Running daily/weekly regressions
  - Formal code coverage
    - Thousands to hundreds of thousands of targets
  - Hybrid formal: search from tons of user-specified far states
    - Validate proof depths are good enough

Non-technical challenges with cloud

- "Perceived" IP risk
  - VP Engineering more conservative than CFO or VP Sales
  - People use SalesForce, CRMs, in same companies
- Legal responsibility (vendor, cloud host, customer?)
- Licensing model
  - Time-based-licensing or Pay-per-use
- First solve the most capital-intensive problems
  - Emulators, costing $1M++
- Vendor solutions exist
  - Synopsys VCS in Amazon cloud
- Private vs public cloud

Opportunities with the cloud

- Access to design and verification environment from anywhere in the world
- Vendors and customers monitor usage, and build business efficient pay-per-use models
- Manage peak usage
- Possible to have flexible architecture – plug-in any engines
  - Exploit latest engine advances
- Lower barrier for proof engine performance feedback back to EDA developers

- Cloud will happen, don't know when… (after emulation?)