The Challenges of Verifying an Arm CPU

Scott Kennedy

Senior Director of Verification, CPU Group @ Arm

scott.kennedy@arm.com

Arm’s challenge is the proliferation of market opportunities

- Arm CPU’s traditional Smartphone/consumer devices market is well understood
  - Quality levels achieved for this market are extremely high… But we also know when to stop
- Time To Market at the correct Quality expectation is just as critical in these newer markets
- Add in the age-old marketing versus engineering “debate” on feature set, schedule, PPA and we have a harder job to do today than we did 10, 5 or even 2 years ago
- But what do those new markets/features really mean for the Verification team?

Arm's challenge is the proliferation of market opportunities

Varying verification expectations compared to our historic Mobile markets

- Infrastructure: Server market needs around ECC/RAS, MMU etc. are markedly different
  - Previously minor errata in Mobile markets are [illegible] Server space… we need to find those bugs before they escape!
- Automotive: Working brakes are app[illegible]portant!
  - FuSa, lock-step, ISO audits all add [illegible]ation team workload… especially requirements tracing for ISO26262
- IoT: Longer battery life for wearab[illegible] always-on domestic appliances
  - Continued focus on PPA, more efficie[illegible]e = more logic complexity = more verification
- Hackers! Security is much more to the for[illegible] across all markets
  - Threat model analysis & a proactive pl[illegible]own and potential threats

More Compute!

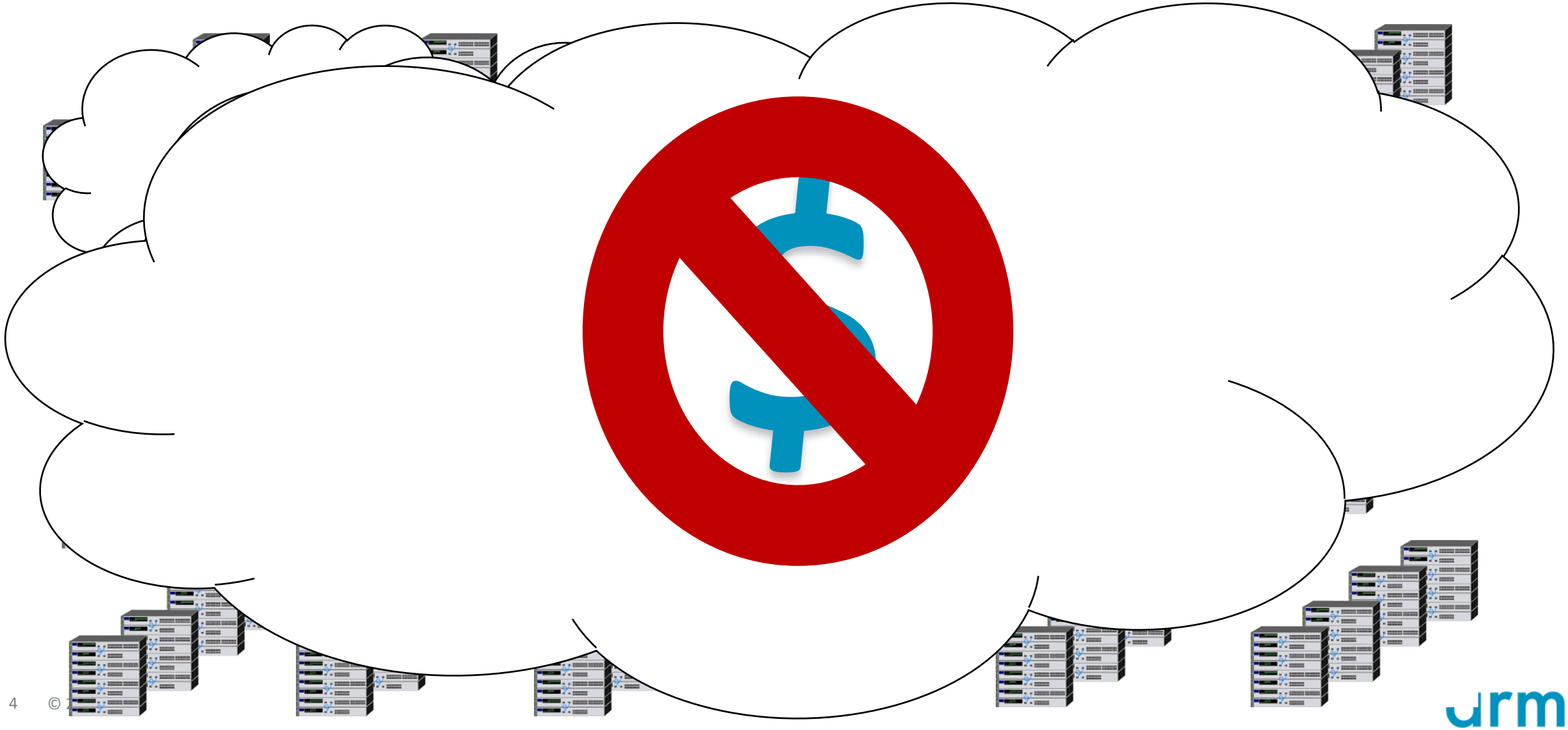
More Compute!

More Compute!

More Compute!

arm

Compute can only go so far in solving this problem



… So our single biggest challenge: How to use Compute efficiently and effectively

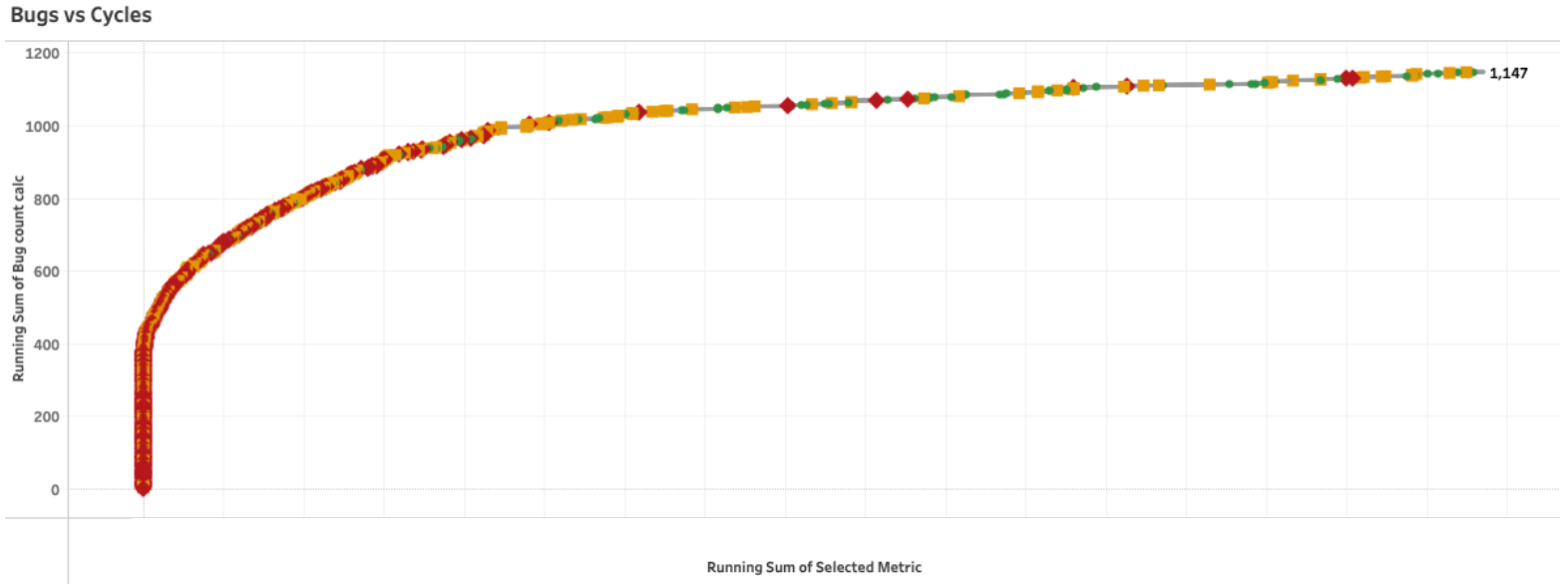
Many simple steps we can take

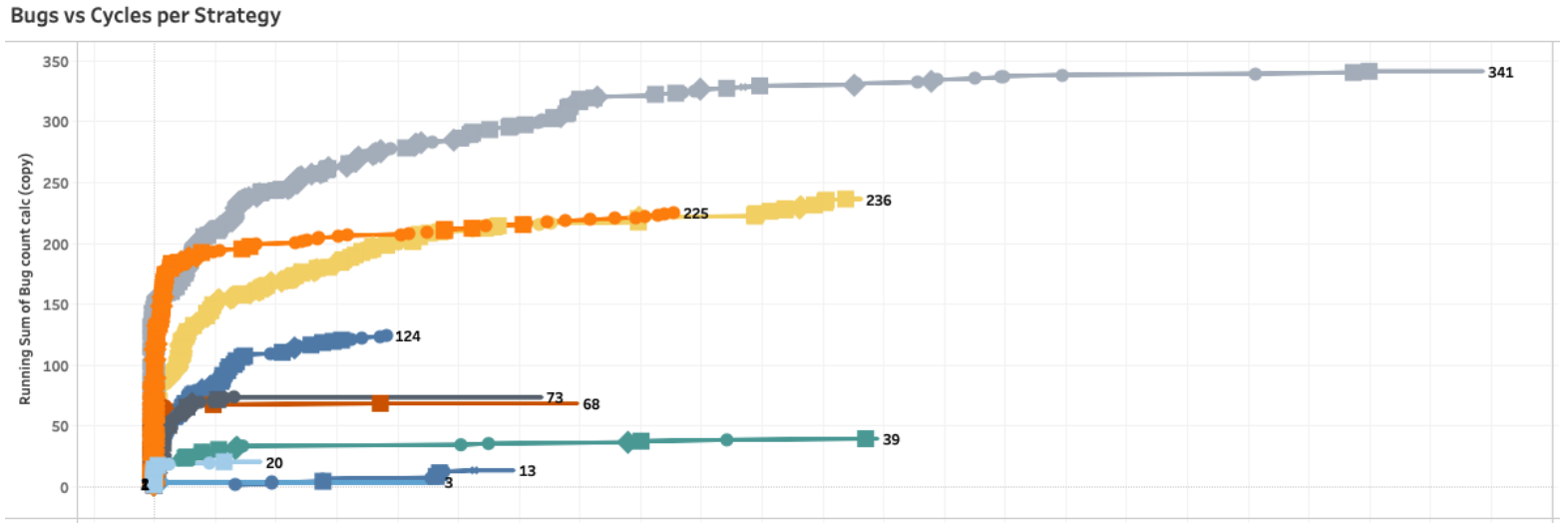
Look at the architecture of the machines we are working on (x86 vs. A64 throughput)	Testbench performance: Work with EDA vendors to resolve unexpected hotspots from profiling	Retire redundant tests: Are we running compute jobs that no longer yield benefit?
Kill regressions when the failure signature count is too high	Find the “correct” balance between simulation and formal (easier said than done)	Only run what can be debugged before the next scheduled regression
Employ the principle of SW Unit testing to the TB code early to minimize the age old “is it the DUT or the TB that’s wrong?” debug cycle	Use multiple simulators… they solve constraints differently so search the state space differently… which means they find different bugs!	Offset longer payloads to Emulation or FPGA: Use the correct platform for the workload… at the correct time!

How to use Compute efficiently and effectively

Be cognizant of data

- If a technique or stimulus source is not finding bugs any more why are you still using compute?
- Use data beyond historic FCOV/CCOV/no fails in last run analysis

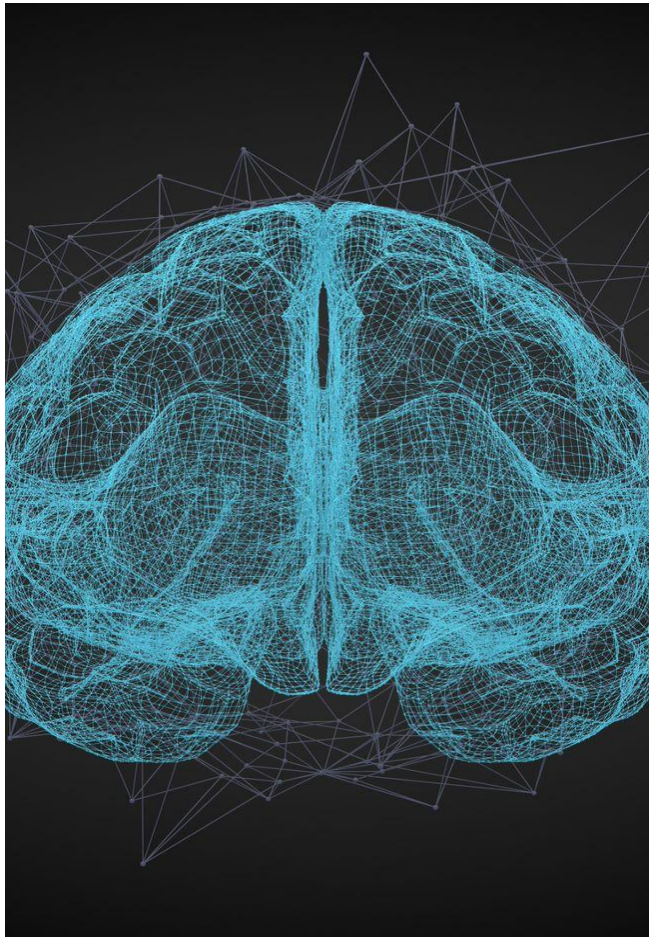




arm

“What about ML? Isn’t it all the rage right now?”

- Arm is using ML and data science techniques to build new verification tools that make verification dramatically more efficient
- We are also working with the EDA companies to better leverage the data they have inside their tools
- Data scientists and ML engineers are now part of our wider verification story, working on how to analyse properties of testbenches and verification data in order to make smarter decisions on what tests we should run



It’s not all about automation and ML… People sharing knowledge is key!

- We have made a concerted effort to move from per-site silos
- Historically each site owned similar projects...

- “Every team for themselves” mentality did not stimulate broader improvement

- But there was minimal knowledge sharing other than on flow-related activities

Silos gone… Sharing is The new normal!

- Share knowledge
- Shares scenarios & concerns even if code can't be shared
- Learn from bugs in similar projects
- Constant dialogue between teams
  - Not retrospectively!

In Summary

- Learn the key differences between your existing markets and emerging opportunities
  - Understand the verification impact
- Leverage a wider knowledge base: Break down the silos. Collaborate on shared verification collateral if possible… if not, then share ideas, issues and solutions
- Compute efficiency is key
  - Only run what you can debug… there is no point in accumulating cycles if you can't debug all the failures before the next scheduled run
  - Run valuable cycles… it's all about training!
  - Train regressions using ML… to target where there is a likelihood bugs
  - Train engineers to make sensible decisions… to target where there is a likelihood bugs
  - Train scripting to stop optimize compute resource… if your project can't use it another team will!
- And finally…

We are always keen to grow the CPU team!

- We have many opening across CPU and our other engineering divisions for all experience levels
- https://careers.arm.com

glassdoor

2021 BEST PLACES TO WORK

arm

arm

Thank You
Danke
Gracias
Grazie
谢谢
ありがとう
Asante
Merci
감사합니다
धन्यवाद
Kiitos
شكرًا
ধন্যবাদ
תודה