Formal Hardware Verification: Theory Meets Practice

Carl Seger

Feb. 3, 2017

Short Bio

- Civilingenjör Teknisk Fysik, Chalmers, 1985
- M.Math. University of Waterloo (Canada), 1986
- Ph.D. University of Waterloo, 1988
- Post-doc Carnegie Mellon University (USA), 1988-1990
- Assistant & Tenured Associate professor, University of British Columbia (Canada), 1990-1995
- Prinicipal & Sr. Principal Engineer, Intel (USA). 1996-2016
- Visiting Fellow at Balliol College Oxford (UK), 2006-2007
- Now visiting Chalmers until end of June…

Quiz – Large Numbers

Order the following in order of size (largest first)

Number of light bulbs in the world

Number of transistors in a 2014 cell phone

Number of atoms in the Empire State Building

Number of patterns needed to simulate all possible inputs to one AVX instruction (two 256-bit inputs)

Quiz – Large Numbers

Order the following in order of size (largest first)

$\sim 10^{10}$

Number of light bulbs in the world

$\sim 10^{11}$

Number of transistors in a 2014 cell phone

$\sim 10^{31}$

Number of atoms in the Empire State Building

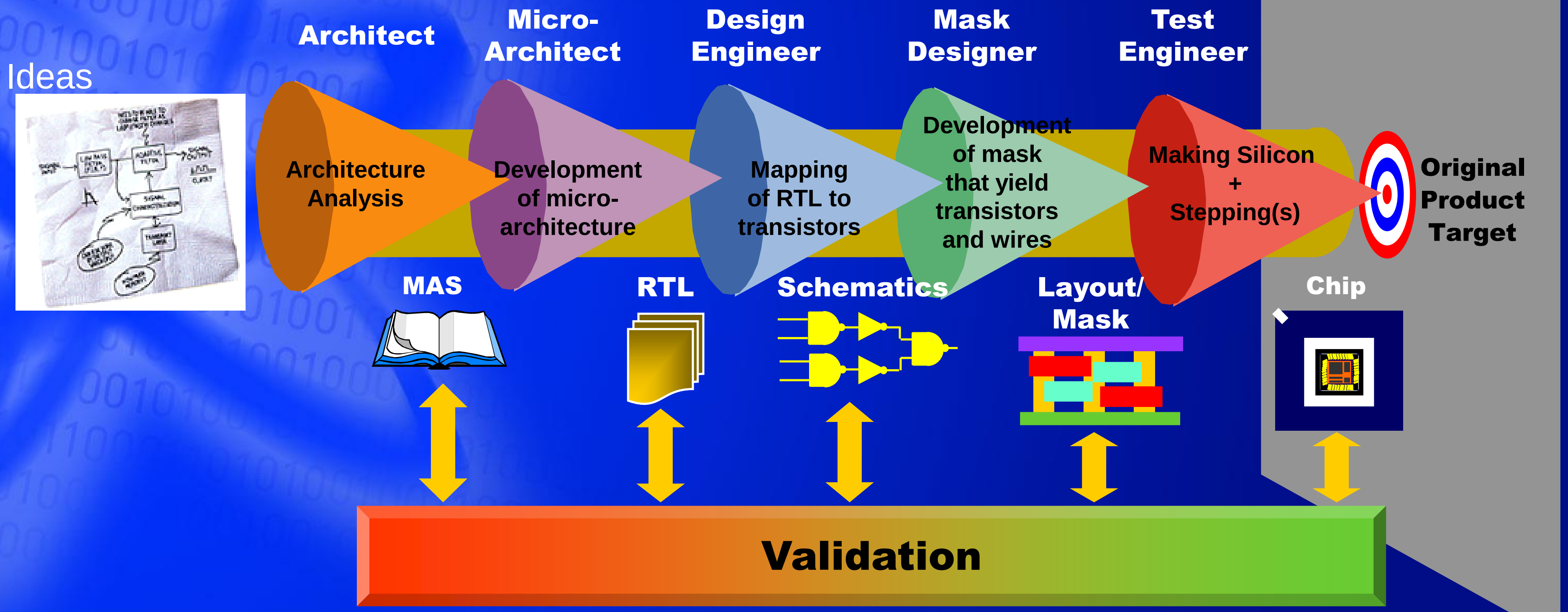
$\sim 10^{154}$

Number of patterns needed to simulate all possible inputs to one AVX instruction (two 256-bit inputs)

Outline

- Background
- Formal Verification in Theory
- Formal Verification in Practice
- The Return-On-Investment (ROI) of FV
- Open Problems

The Design Process at 10,000 m

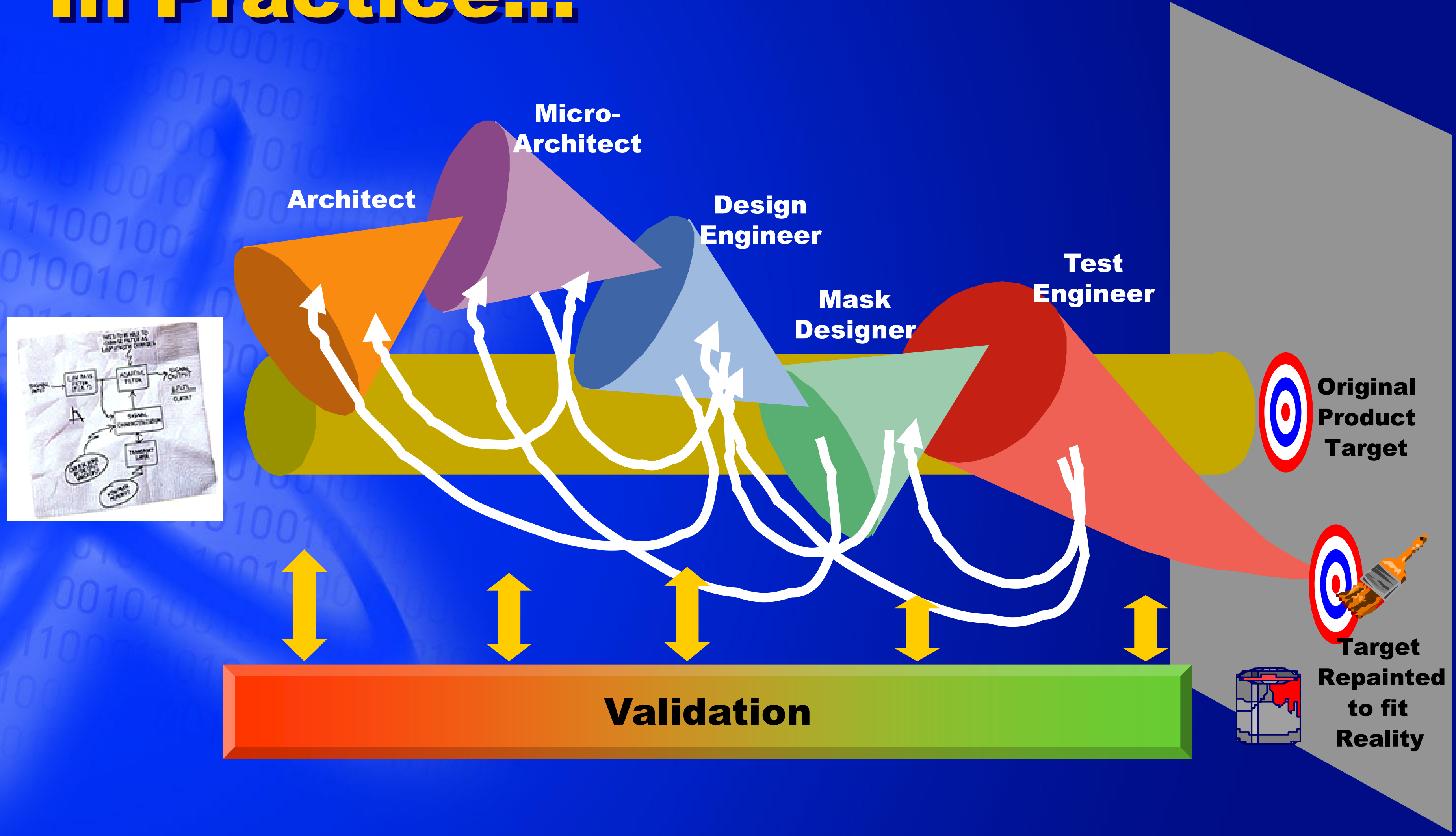


MAS: Micro-Architecture Specification

RTL: Register-Transfer Language

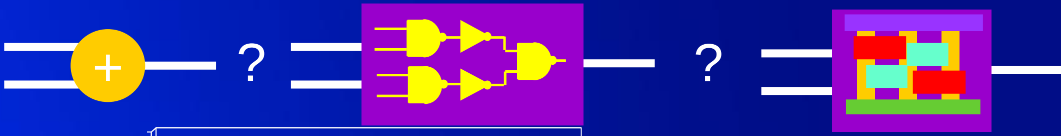
This is the theory...

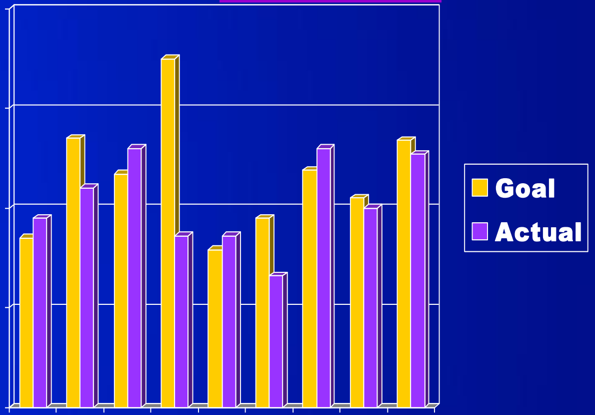
In Practice...

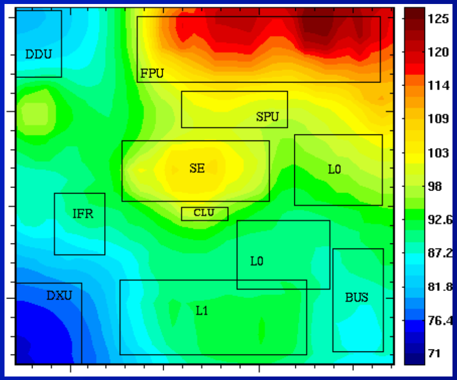

Micro-Architect
Architect
Design Engineer
Mask Designer
Test Engineer
Original Product Target
Validation
Target Repainted to fit Reality

What Needs to be Validated?

- Functionality
- Performance
- Power & Thermal
- Physical form
- Documentation
- Reliability
- Testing procedure
- …

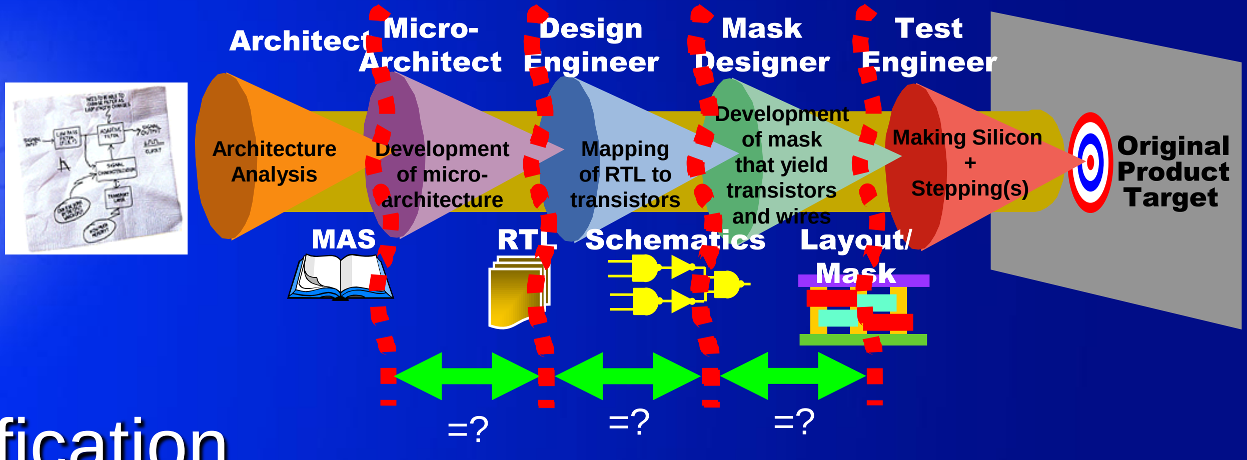




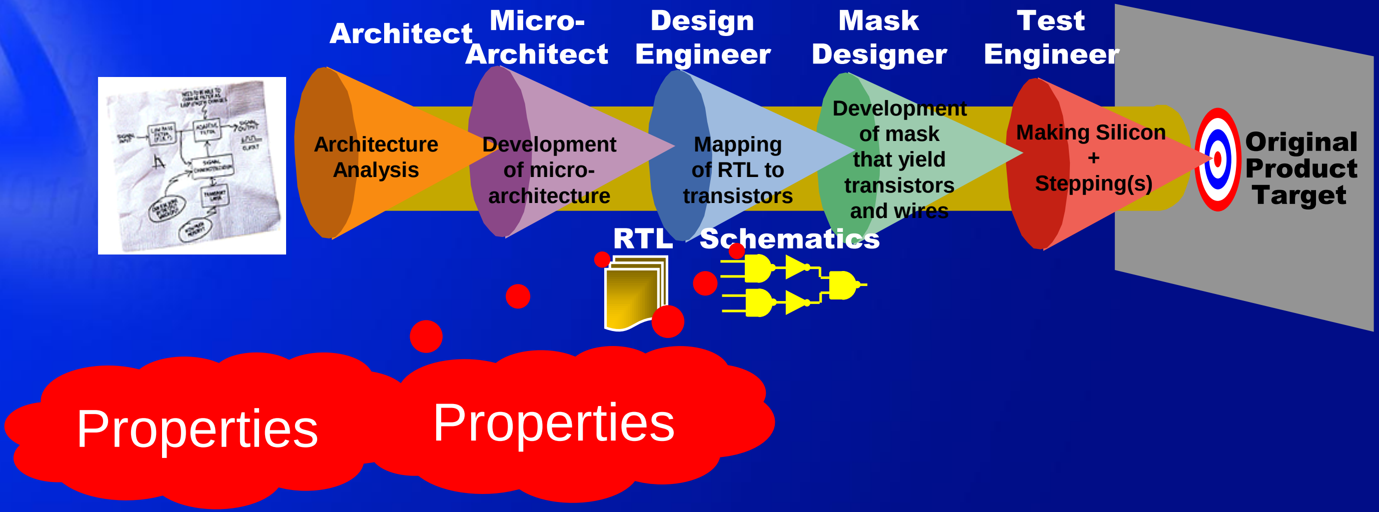


Types of Functional Verification

- Equivalence Verification



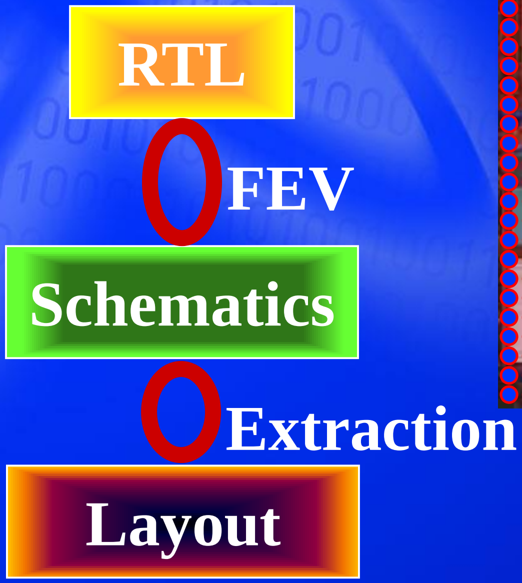
- Property Verification

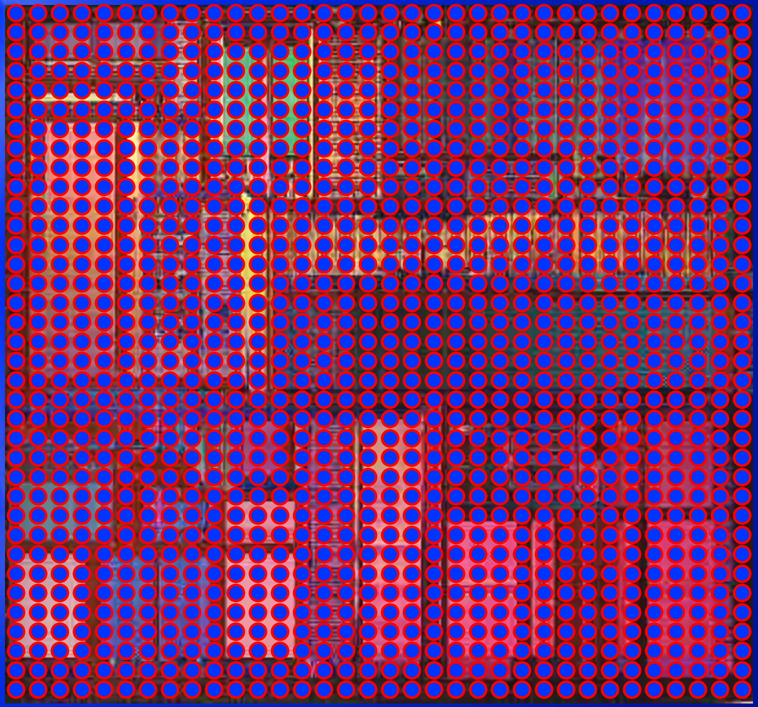


Architect
Micro-Architect
Design Engineer
Mask Designer
Test Engineer
Architecture Analysis
Development of micro-architecture
Mapping of RTL to transistors
Development of mask that yield transistors and wires
Making Silicon + Stepping(s)
Original Product Target
RTL
Schematics
Properties
Properties

Formal Equivalence Verification

- Use of symbolic/algebraic methods to completely verify that a circuit implements a specification



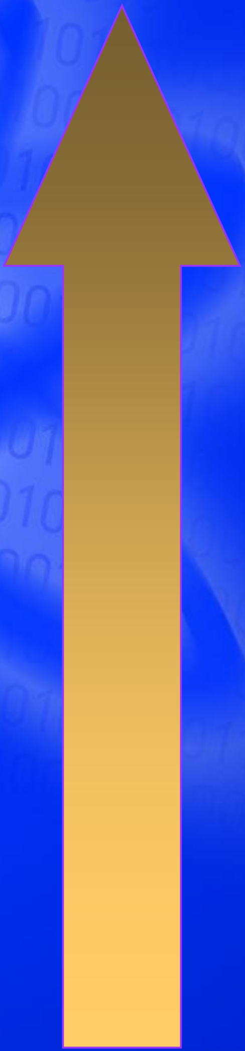


Today: 100% of a design is run through FEV before tape-out

Extremely successful application of formal verification in practical engineering!

Usability high enough that every design engineer is able to run the verification.

Property Verification Approaches



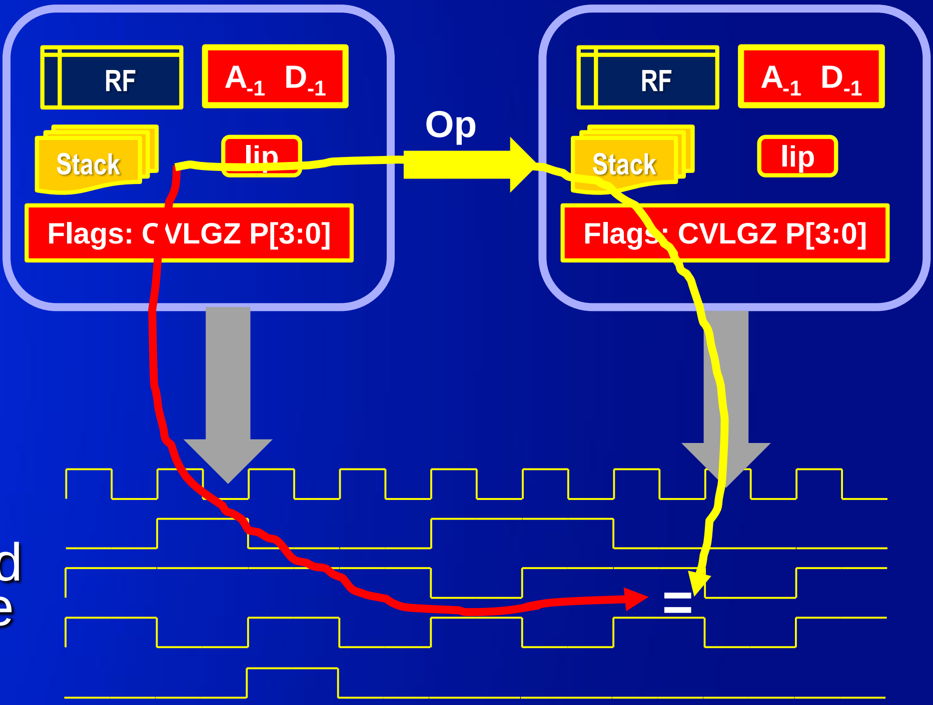
	Pro	Con
Formal Verification	•100% coverage •Proves absence of bugs	•Requires special skills •Constrained by complexity
Directed Random Tests	•Targets areas most likely to be of concern •Greatly reduces cycle requirements •Develops strong uArch knowledge	•Requires strong uArch knowledge
Generic Random Tests	•After generator created, easy to write •Requires little uArch knowledge •Can create things no one would ever think of	•Requires almost ∞ cycles / time •Difficult / impossible to avoid broken features
Directed Tests	•Easy to write •Easy to understand •Easy to reuse	•Requires almost ∞ number of tests •Difficult to hit uArch conditions

Formal Verification Approaches

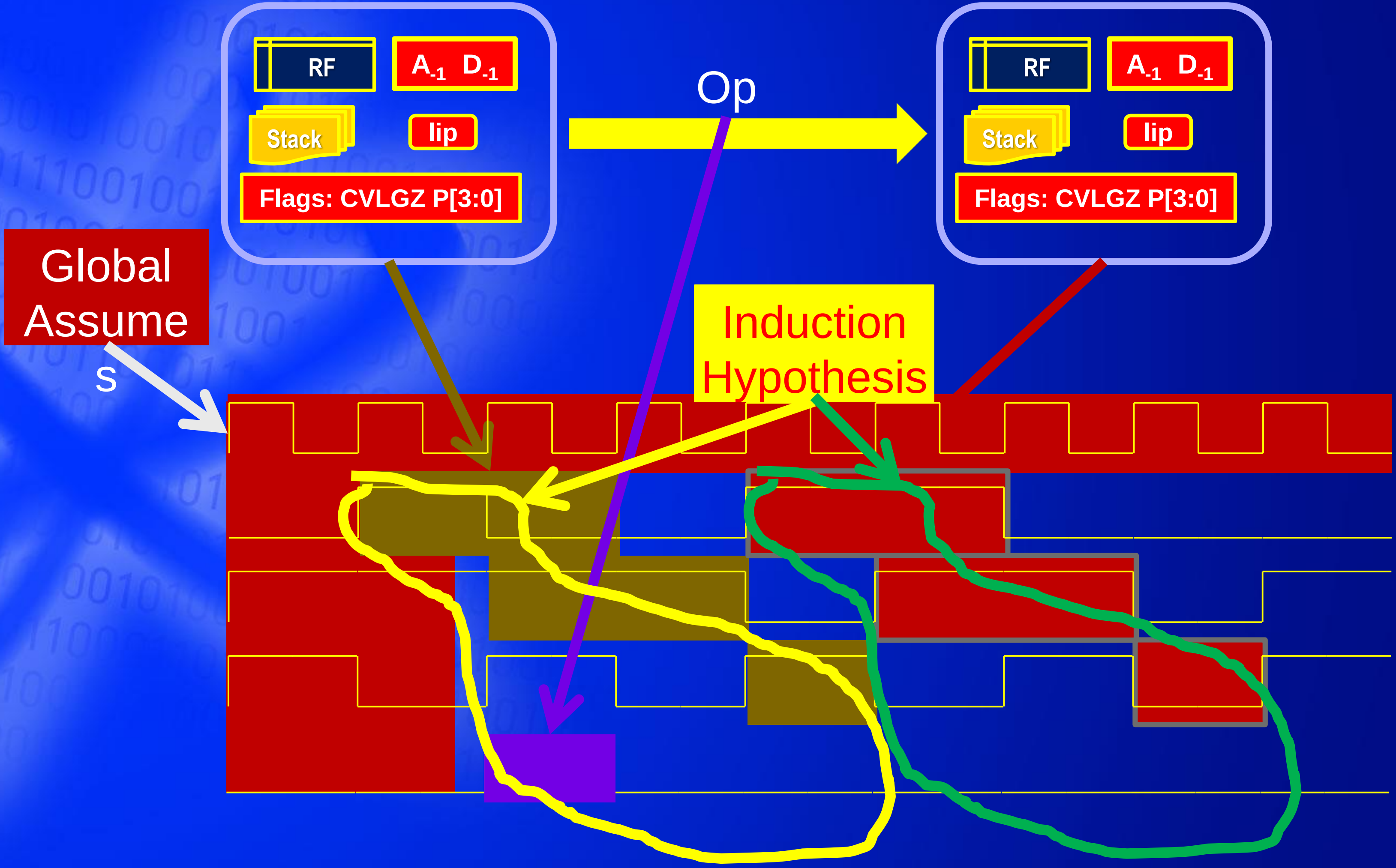
- Symbolic Trajectory Evaluation (STE), a form of symbolic simulation, are today used to formally verify very large computation units/blocks
  - Complete formal property verification of all (>3,000) uops in the execution cluster of Intel processors is now routinely done
- Symbolic model checking is seeing more wide spread use
  - Early architecture exploration/validation
  - Control intensive designs
- Theorem Proving
  - Combining STE with theorem proving increases the quality of specification
  - Floating point spec is mathematical statement of IEEE standard

STE from 10,000 m

- Reference model verification:
  - Create an abstract state representation
  - Define an abstract next-state function
  - Define a mapping from abstract state to circuit state
  - Verify commutativity:
    For every instance in time and for every possible value in the machine, if the signal values match the mapped "before" state, then they will also match the "after" state.



STE from 1,000 m



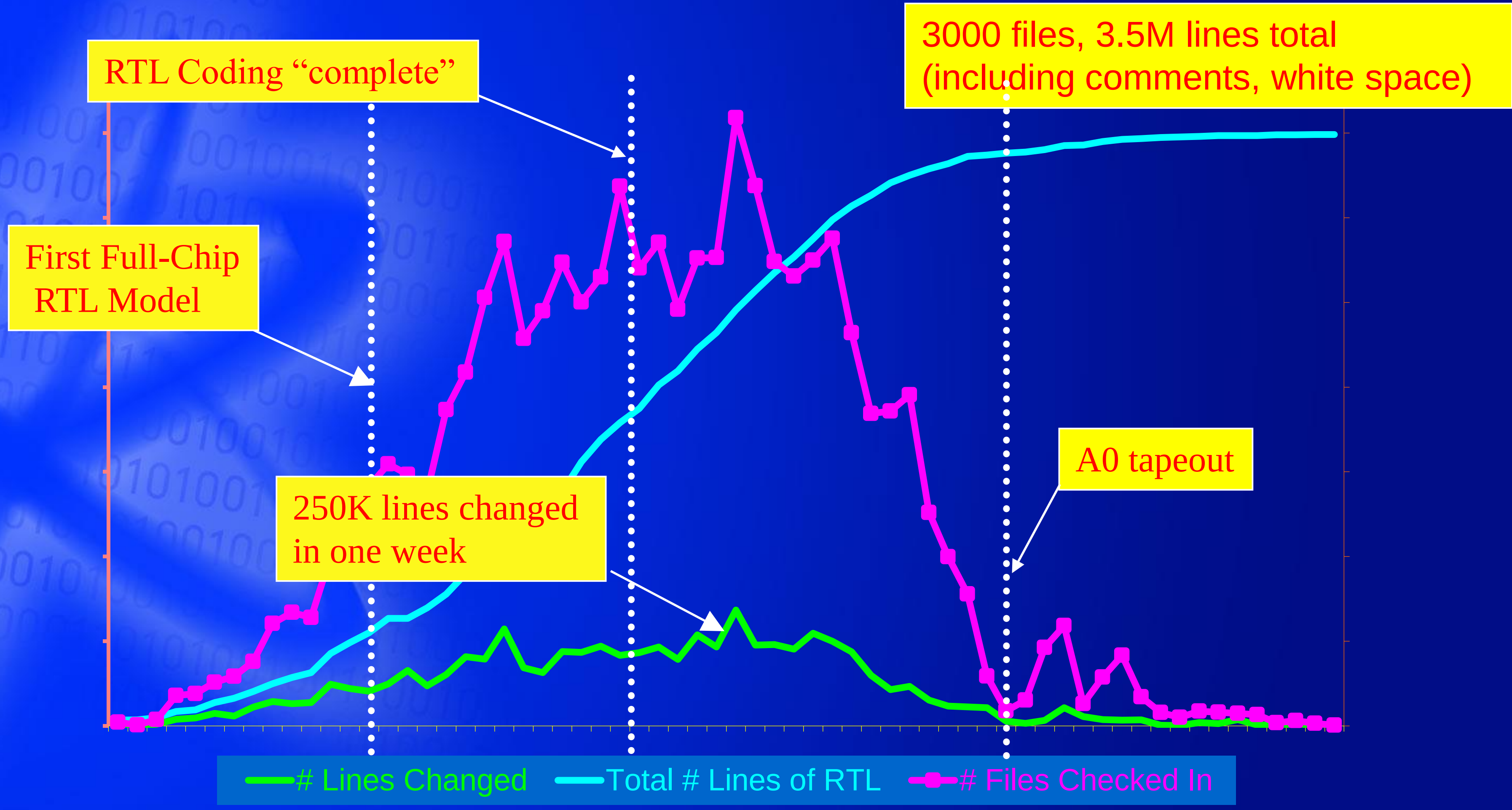
STE at Ground Level

- Model build
- Wiggling
- Specification writing
- Simple verification
- Debugging failures
- Complete verification
- Debugging failures
- Assumption checking
- Debugging failures

In Summary:

- A lot of rather tedious work.
- Significant amount of reverse engineering.
  - Both FV and micro-architectural knowledge needed
- You get it wrong way more than you get it right.
  - If the proof succeeds on first attempt, it's probably wrong!
- Debugging and exploration critical!
  - Fast turn-around more important than FV capacity!
- The FV **activity** is the real value!

RTL Changes Constantly

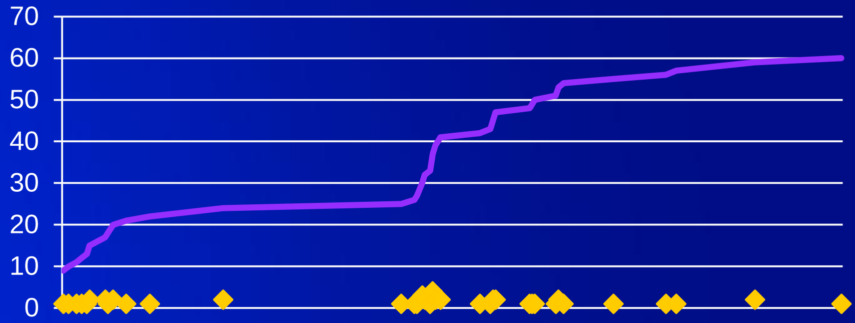


In Summary: FV in Practice

- A lot of rather tedious work.
- Significant amount of reverse engineering.
- You get it wrong way more than you get it right.
  - If the proof succeeds on first attempt, it’s probably wrong!
- Debugging and exploration critical!
  - Fast turn-around more important than FV capacity!
- The FV **activity** is the real value!
- Regression is critical (and time consuming!)

The Return On All This Work

Example STE FV of a Unit



- ~60 tickets filed:
  - 34 bugs in spec. (C++)
  - 37 bugs in RTL
  - 8 bugs in EAS (English document)
  - At least one ticket caused a change in all three models!
  - 2-3 bugs were already present in existing (and shipping) HW!
  - Most bugs related to setting of flags and other “corner cases”.
  - Most complex bug required a program with 71 instruction and carefully selected program layout to split cache lines + suitable cache misses. Also known as a “Friday the 13th bug”
  - All bugs were fixed, even though many required several spins.
- Total effort: ~1 man year.

FV Over Time

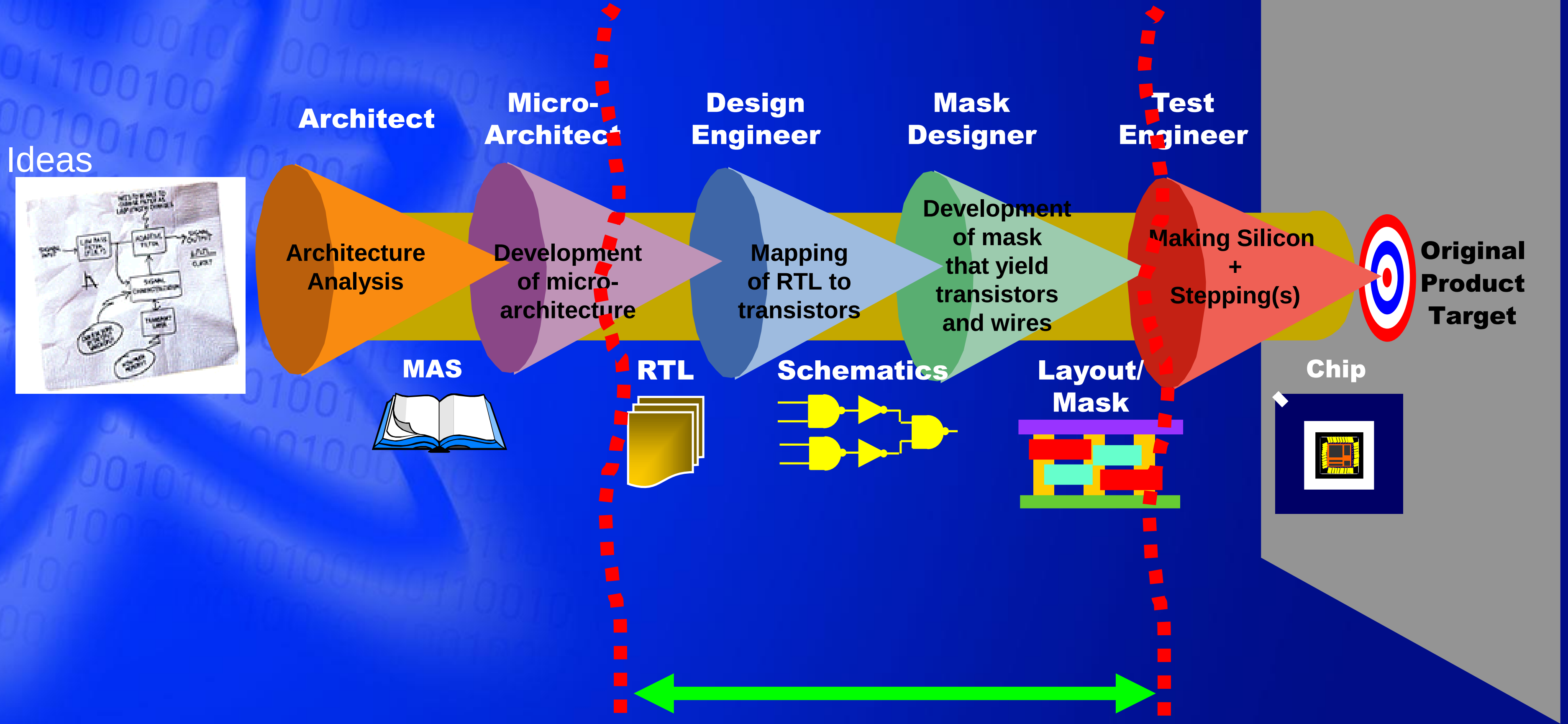
- STE FV deployed for more than 15 years for the execution cluster.
- Initially only data path verified
- Eventually all data path and control were verified.
- Today FV has replaced simulation entirely.
- Headcount today lower with FV than what DV would require.
- For at least one project, we achieved zero post-Si bugs, for others, much cleaner Si.

Good News / Bad News

- Good news:
  - Formal verification can guarantee the correctness of extremely large and complex hardware
  - The verification programs allow continuous regression runs, thus preventing bugs from re-appearing
  - The verification specifications and verification scripts can often be re-used for new designs
- Bad news:
  - Difficult to capture control aspect accurately & robustly
  - Knowledge intensive activity to create initial specs and verification scripts
  - FV capacity not growing as fast as design size/complexity.
  - Structural verification decompositions are fragile

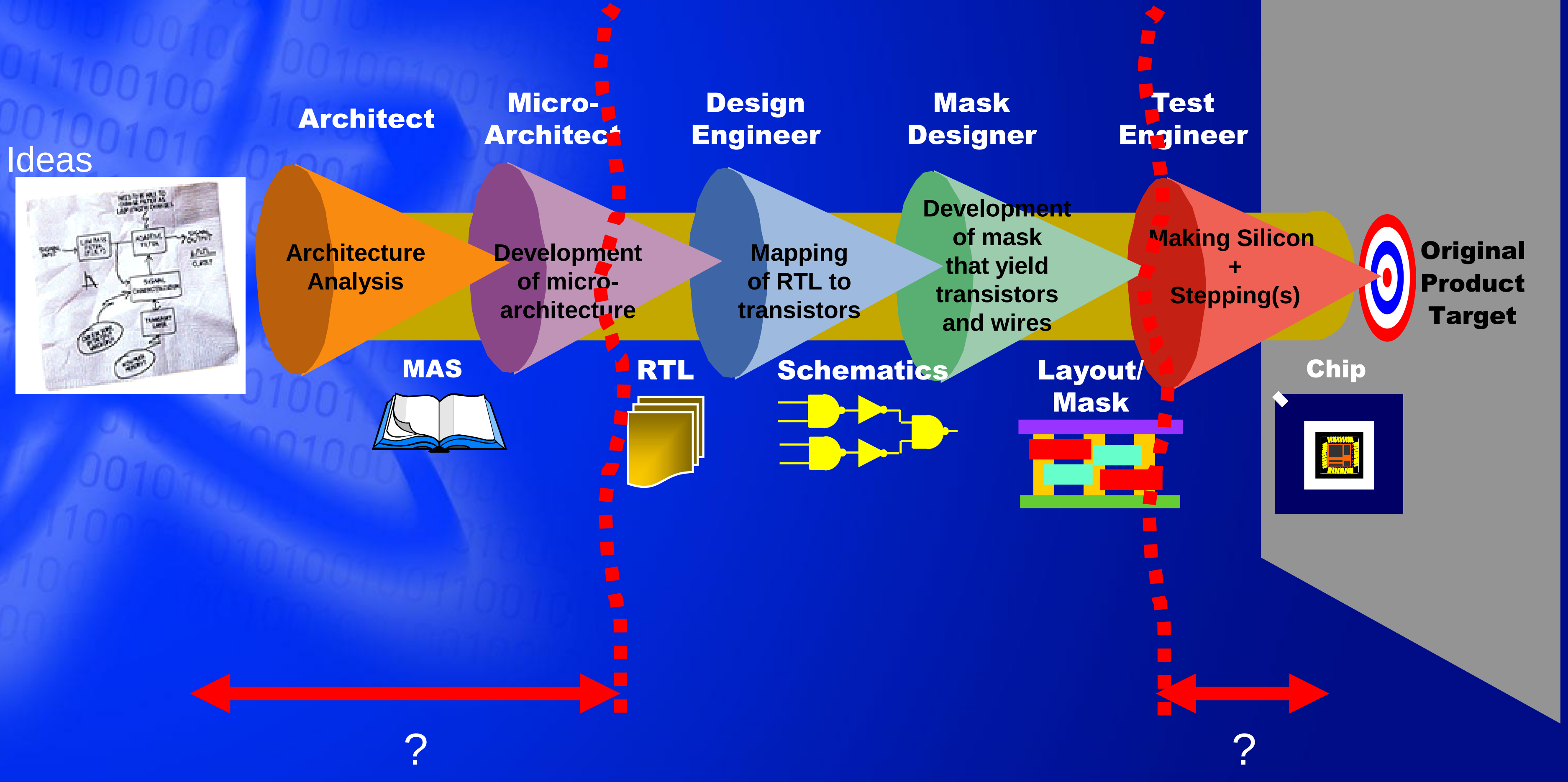
Future Directions and Research Problems

Solid Formal Link with Good Return of the Investment



FPV+FEV + Extraction+DRC

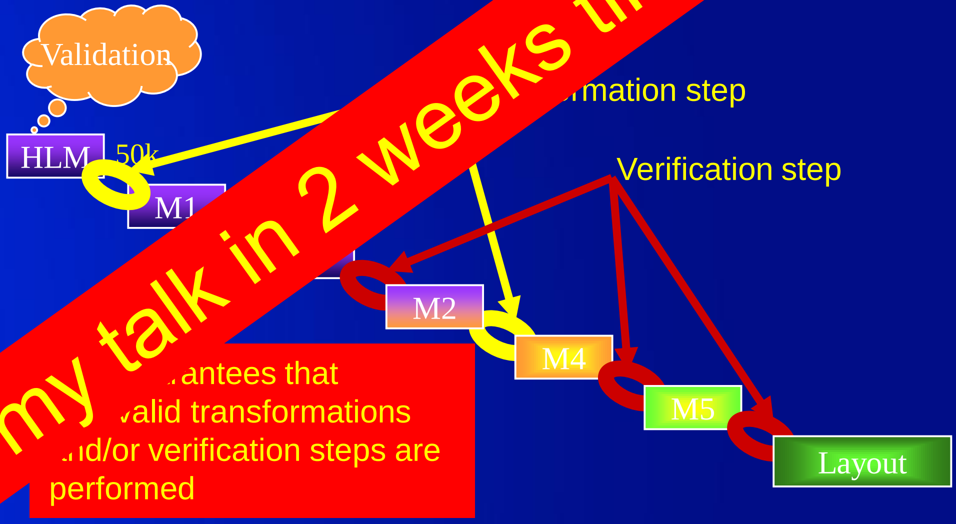
Mind the Gap(s)…



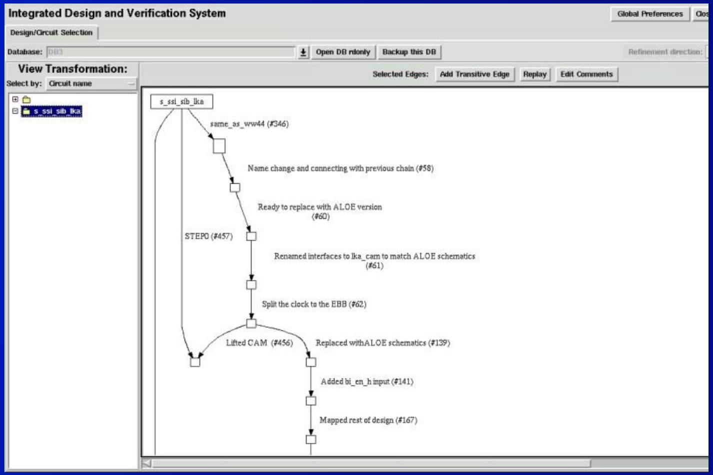
Integrate Verification and De gn

- All validation work is reactive; the design gets created somehow and now we need to figure out if it is correct
- Rather than trying to do post-design verification, verify each step along the way.
  - Can mix “correct-by-constructi and “trust-but-verify” parts.
  - Can use different verifi engines at different abstraction
  - Imposes a r odest overhead sign process for a big n
  - A n be built to track the of a design from ness point of view.

For more details, come to my talk in 2 weeks time!



antees that valid transformations nd/or verification steps are performed


Integrated Design and Verification System
Global Preferences
Clos
Design/Circuit Selection
Database:
DB3
Open DB rdonly
Backup this DB
Refinement direction:
View Transformation:
Select by:
Circuit name
s_ssi_sib_lka
Selected Edges:
Add Transitive Edge
Replay
Edit Comments
s_ssi_sib_lka
same_as_ww44 (#346)
Name change and connecting with previous chain (#58)
Ready to replace with ALOE version (#60)
STEP0 (#457)
Renamed interfaces to lka_cam to match ALOE schematics (#61)
Split the clock to the EBB (#62)
Lifted CAM (#456)
Replaced withALOE schematics (#139)
Added bi_en_h input (#141)
Mapped rest of design (#167)

IDV prototype system for abstract RTL to layout with complete verification

Finding a Needle in a Haystack vs Finding a HW bug

vs

Fi ... le pair of values for ... precision floating point ... ub/mult/div) op that fails.

Formal Techniques are Essential
& They are Practical!

For probability to be the same, how big should the haystack be? (Assume half-sphere haystack)

Answer: Radius ~550 light years!

Thank you!

Questions?