# Accelerating time to first trade with FPGAs

Mark Eslinger
Siemens EDA
IC Verification Solutions / DVT



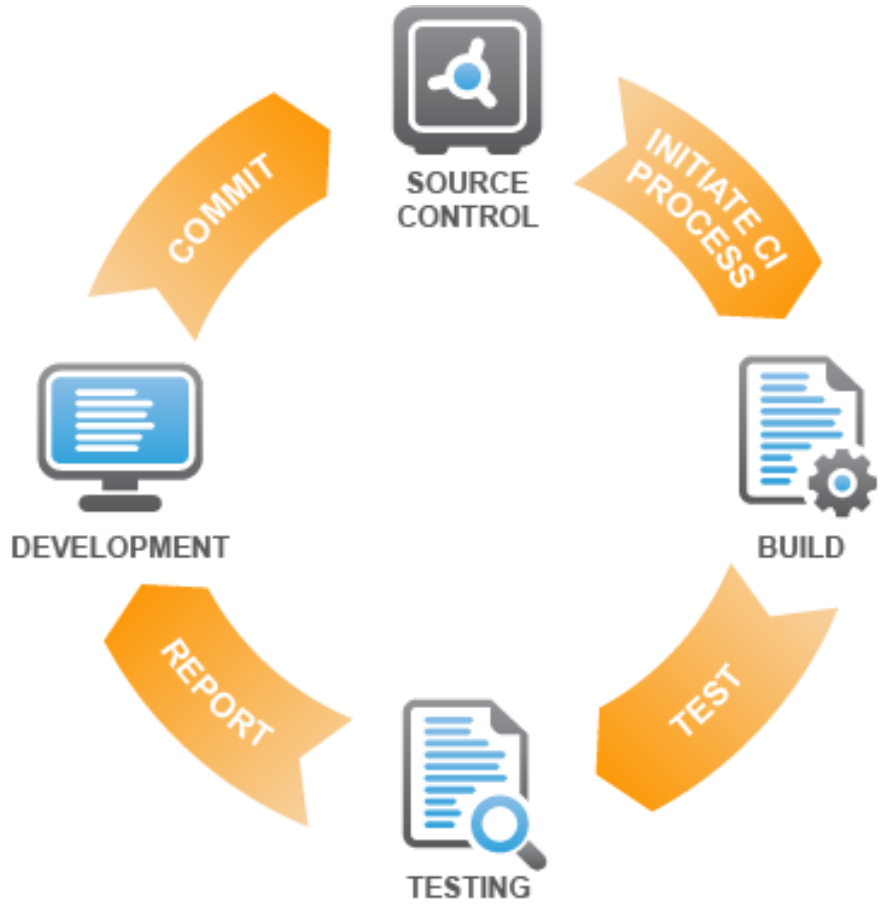
SIEMENS

# Applying agile/CI methods to h/w design with static and formal

Agile

DEVELOPMENT

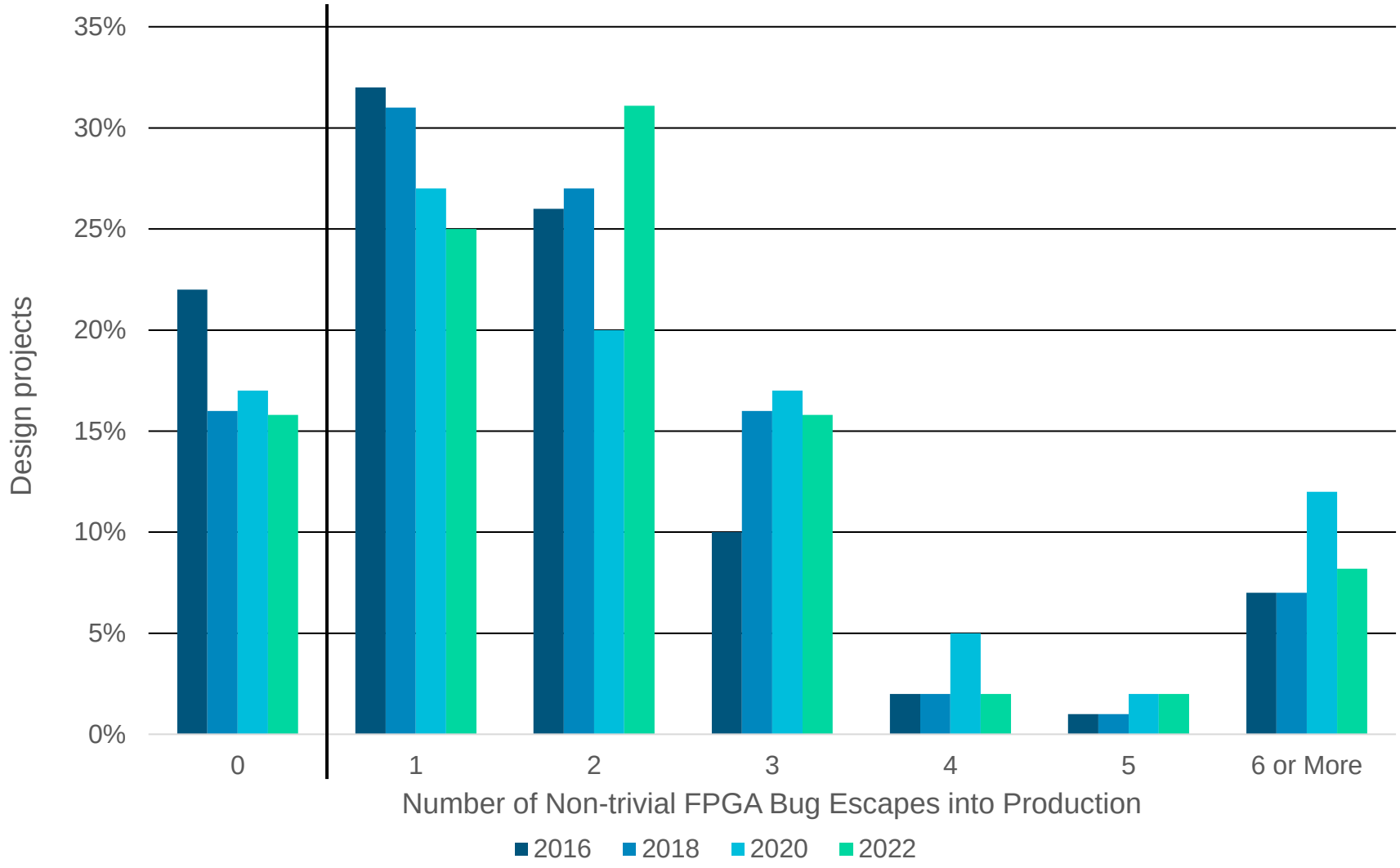
- **Continuous Integration** (CI) is a development practice that requires developers to **integrate** code into a shared repository several times a day. Each check-in is then verified by an automated build, allowing teams to detect problems early.






SIEMENS

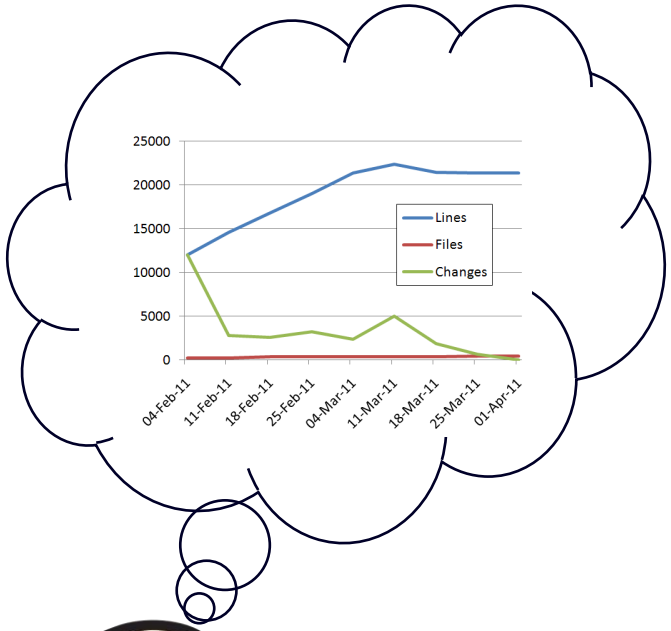
# 84% of FPGA projects have non-trivial bug escapes






SIEMENS

# Complete continuous integration pipeline




Static and Formal Checks

Results (Normalized)

Pass

CI Pipeline

Jenkins

Image from Jenkins Project
https://jenkins.io/

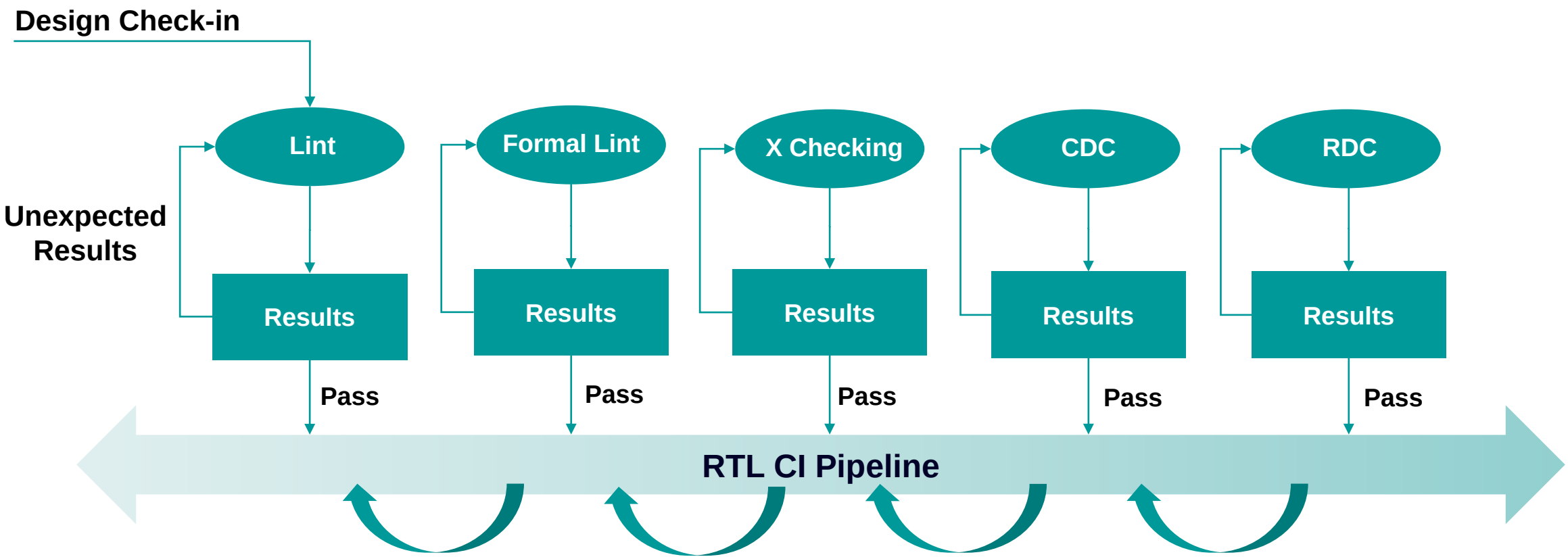
Success

AMD
XILINX
intel
LATTICE SEMICONDUCTOR
MICROCHIP



SIEMENS

# Static and formal design checks build a continuous integration pipeline

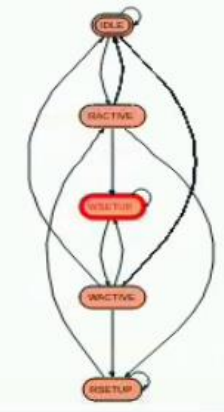


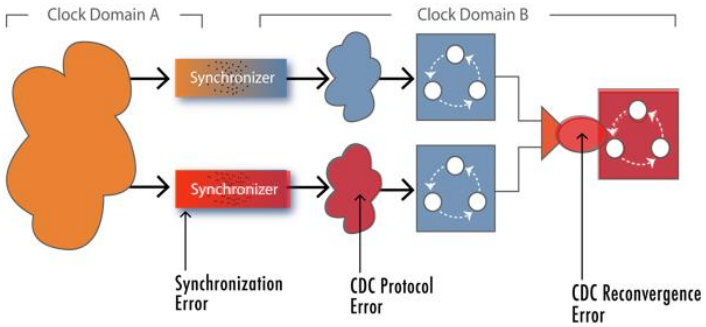


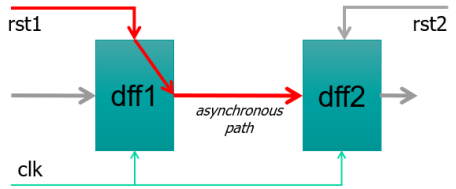

SIEMENS

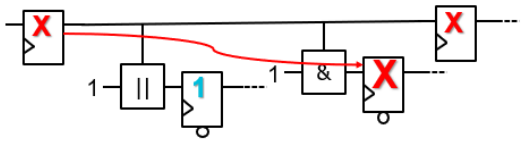
# Team based early functional verification tied to CI flows

combo_loop
undriven_wire
syn_sim_mismatch
…





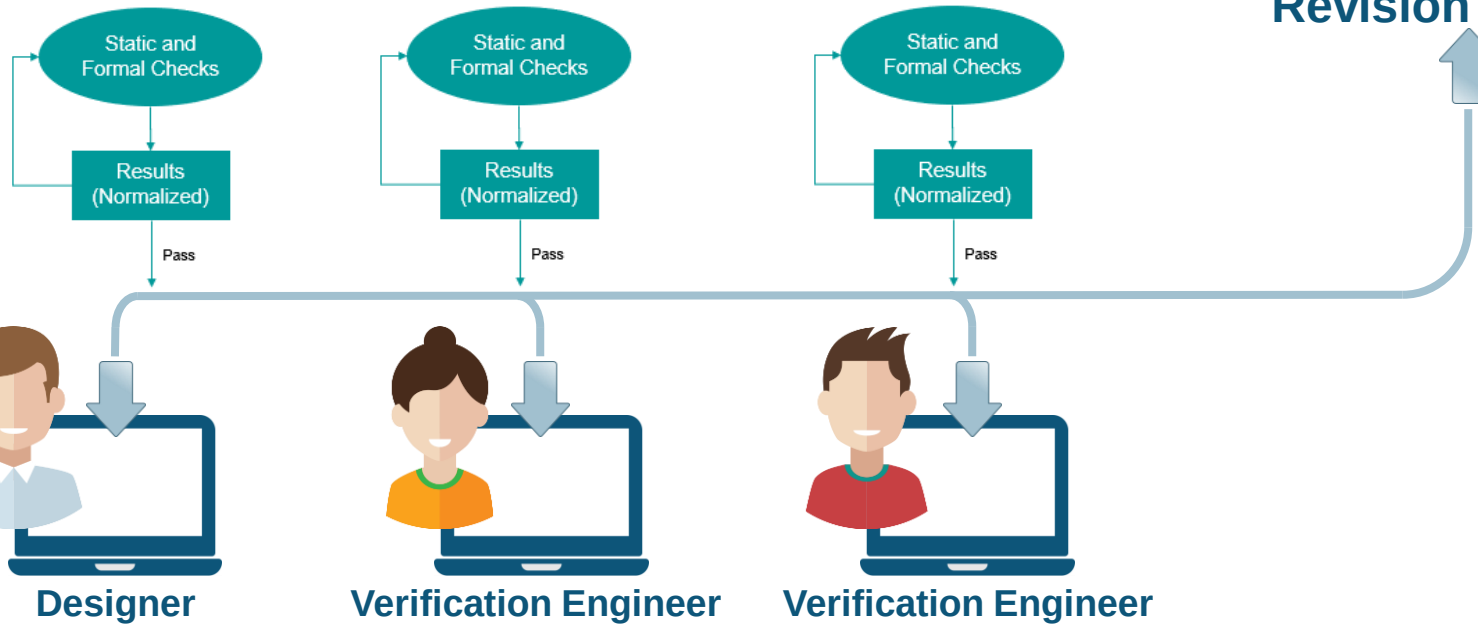




Static and Formal Checks

Revision Control

Manager




Designer

Verification Engineer

Verification Engineer

# Example: Comparing lint and formal lint

## Lint?

**No – this is not Lint**

**Syntax checks: ✓**

**Semantic checks: ✓**

**Structural checks: ✓**

**Stylistic checks: ✓**

```
case (qstate)
3'b001: if (en) dstate = 3'b010;
 else dstate = 3'b001;
3'b010: dstate = 3'b100;
3'b100: if (rtn) dstate = 3'b001;
 else dstate = 3'b100;
default: dstate = 3'b001;
endcase
```

## Formal Lint?

**Yes! It's Formal Lint**

**Sequential checks:**

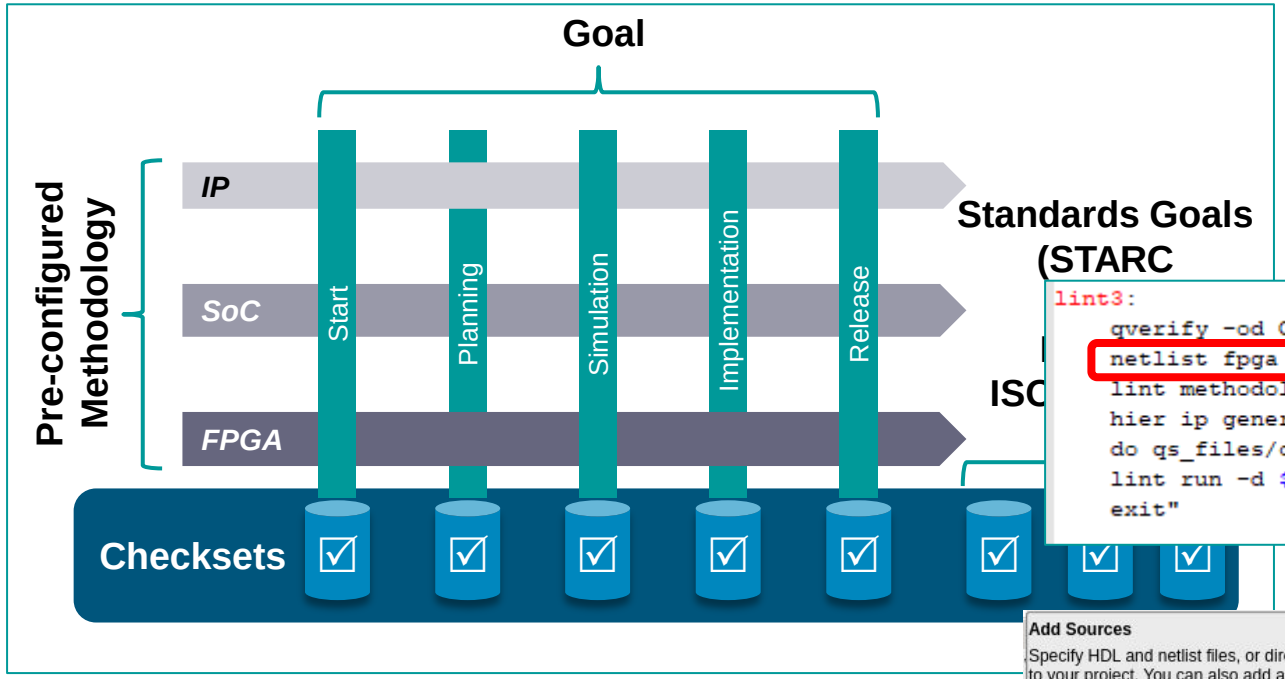
- Understands whether `rtn` can ever be 1.
- If not, the FSM will deadlock at state 3'b100.

**Advanced Linting checks beyond static checks**



SIEMENS

# FPGA methodologies and library support to enhance value of a CI flow




```
lint3:
    qverify -od Output_bbox -c -do " \
    netlist fpga -vendor xilinx -library vivado; \
    lint methodology ip -goal release; \
    hier ip generic_fifo_dc_gray; \
    do qs_files/checks_off.tcl; \
    lint run -d $(DUT); \
    exit"
```

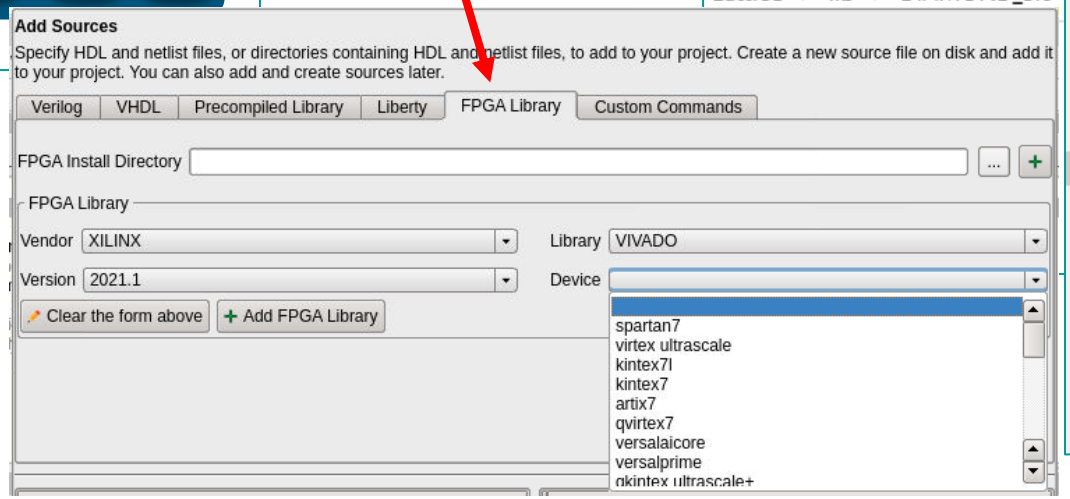
share › fpga_libs › Xilinx › lib

Name
- ISE_13.2
- ISE_13.4
- ISE_14.7
- VIVADO_2019.2
- VIVADO_2020.1
- VIVADO_2020.2
- VIVADO_2021.1

share › fpga_libs › Intel › lib

Name
- QUARTUS_19.4_pro
- QUARTUS_20.1_pro
- QUARTUS_20.1_std
- QUARTUS_20.2_pro
- QUARTUS_21.2_pro

Lattice › lib › DIAMOND_3.6




fpga_libs › Microchip › lib

Name
- LIBERO_11.8
- LIBERO_11.9
- LIBERO_12.0
- LIBERO_12.1
- LIBERO_12.2
- LIBERO_12.3
- LIBERO_12.4
- LIBERO_12.5
- LIBERO_12.6
- LIBERO_2021.1



SIEMENS

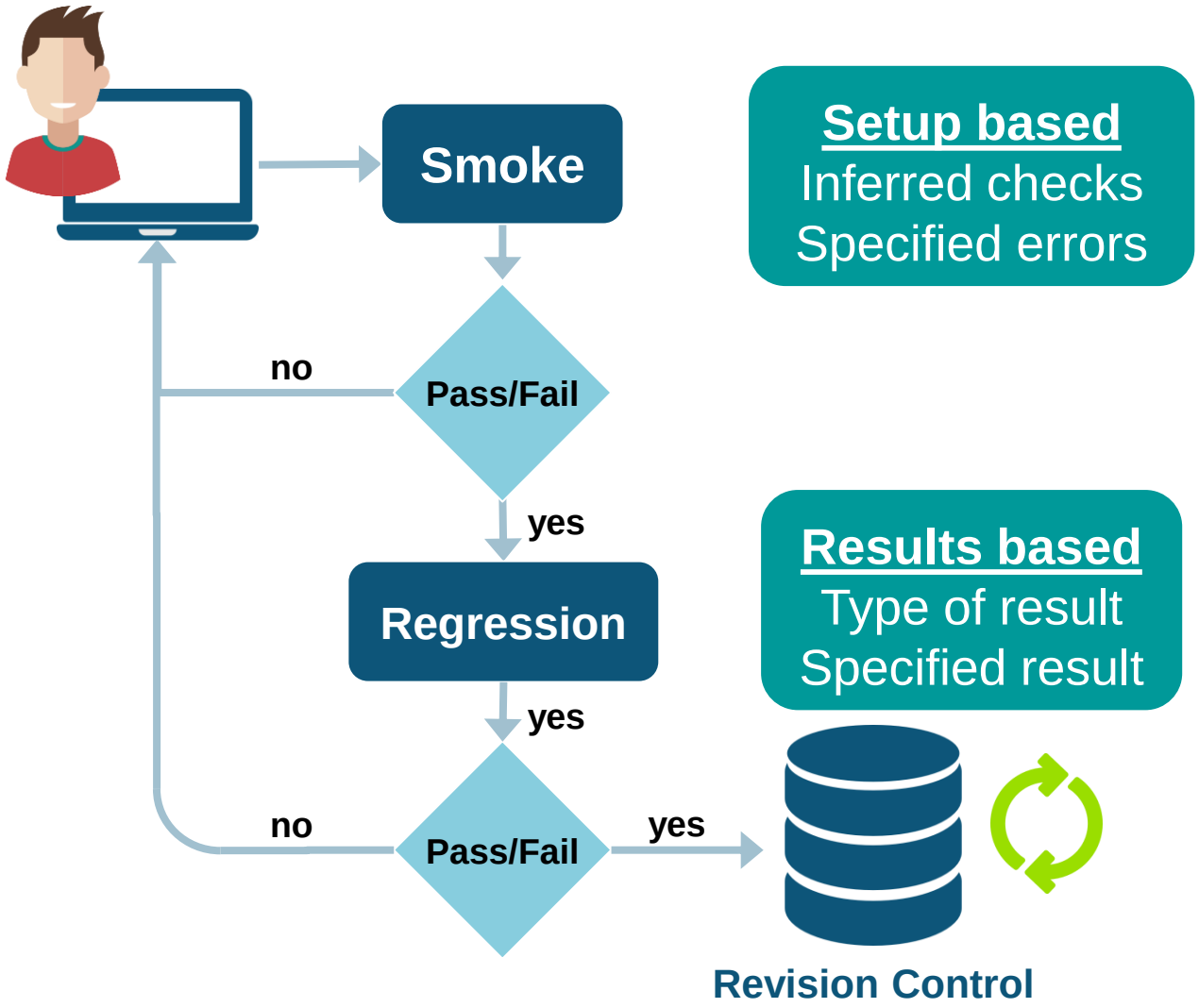
# Build and test using CI

## Trunk check

- Sanity test
- Subset of regression
- Detect simple yet severe problems
- Frequency
  - Ran at least daily on trunk
  - Before every check-in

## Regression

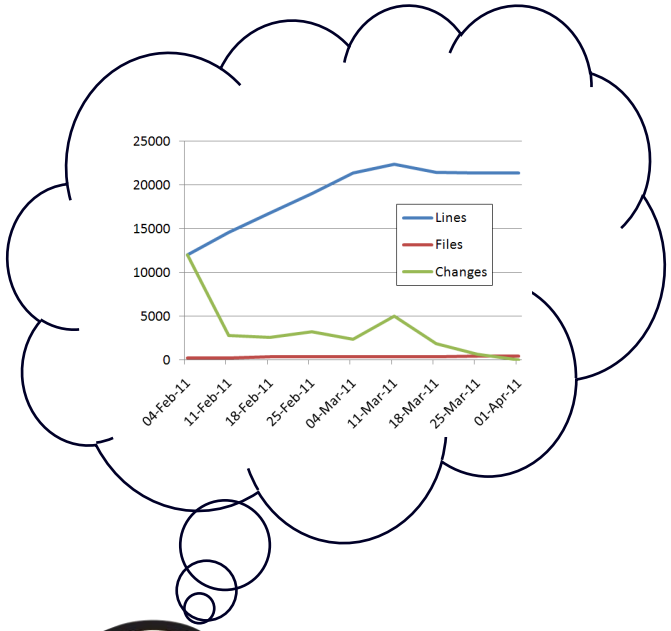
- Full test of functionality
- Verify implemented features
- Adds coverage
- Frequency
  - Ran at most nightly






SIEMENS

# Complete continuous integration pipeline




Static and Formal Checks

Results (Normalized)

Pass

Regression

Results (Normalized)

Pass

CI Pipeline

Jenkins

Image from Jenkins Project
https://jenkins.io/

Success

AMD
XILINX
intel
LATTICE SEMICONDUCTOR
MICROCHIP



SIEMENS

# Automating Verification: Providing productivity and efficient use of resources

## Typical Regression Limitations

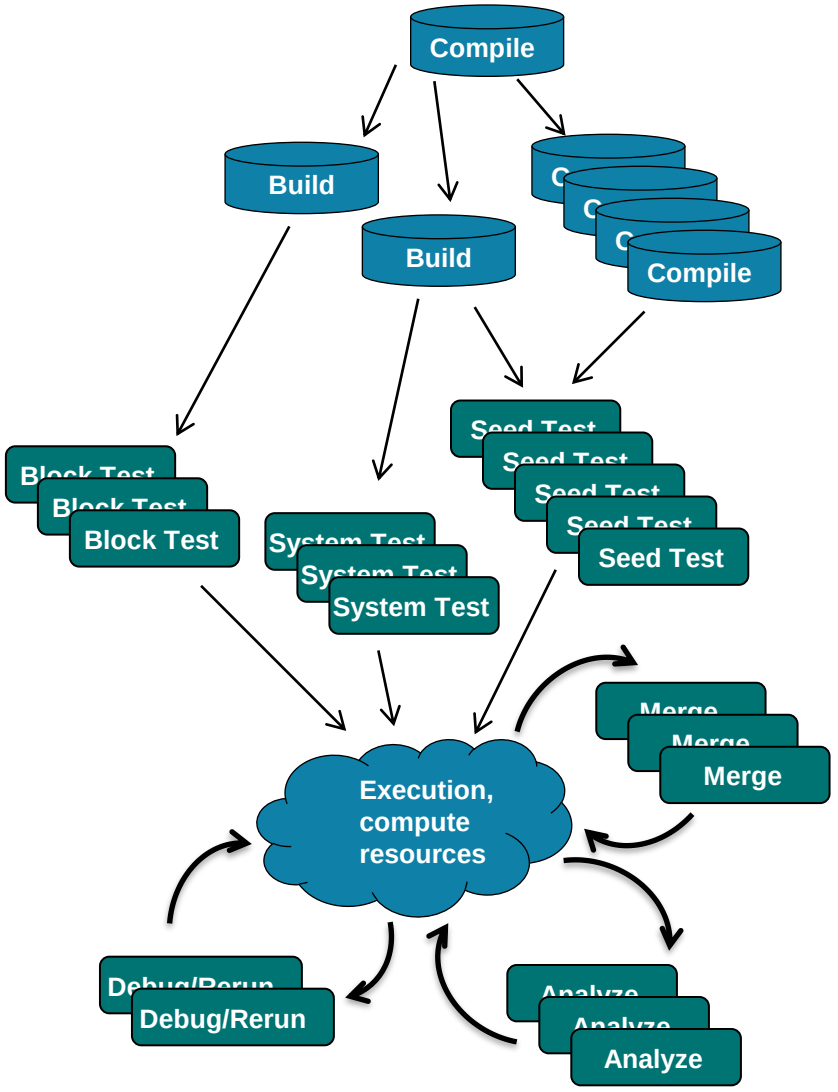
- No alternative but to home grow
- Script based, little separation between configuration and control
- Expensive to create & maintain
- Hidden data & automation opportunities

## Benefits of a pre-built environment

- Focus resources on verification, not infrastructure
- Simplify set-up & Maintenance
- Continual development & support

## Improve throughput and turn-around

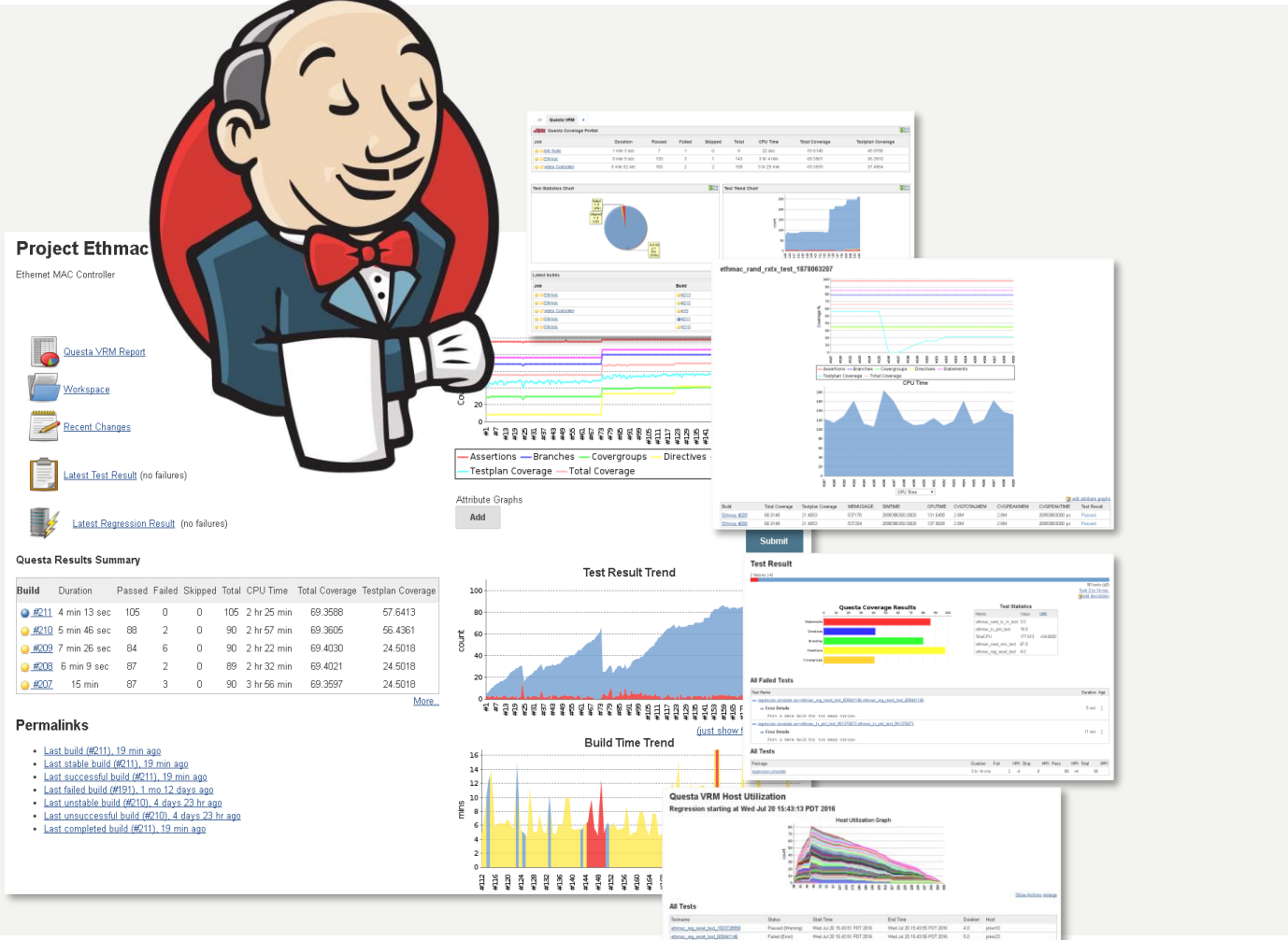
- Faster execution of regression tasks
- Automation and repeatability of process






SIEMENS

# Regression Management and Continuous Integration
## Run Management Plug-in for Jenkins




## Verification Automation

- Continuous Integration
- Run Management

## Improved Quality

- Frequent Testing
- Faster Coverage Closure

## Process Collaboration

- Results Analysis
- Metric Visualization

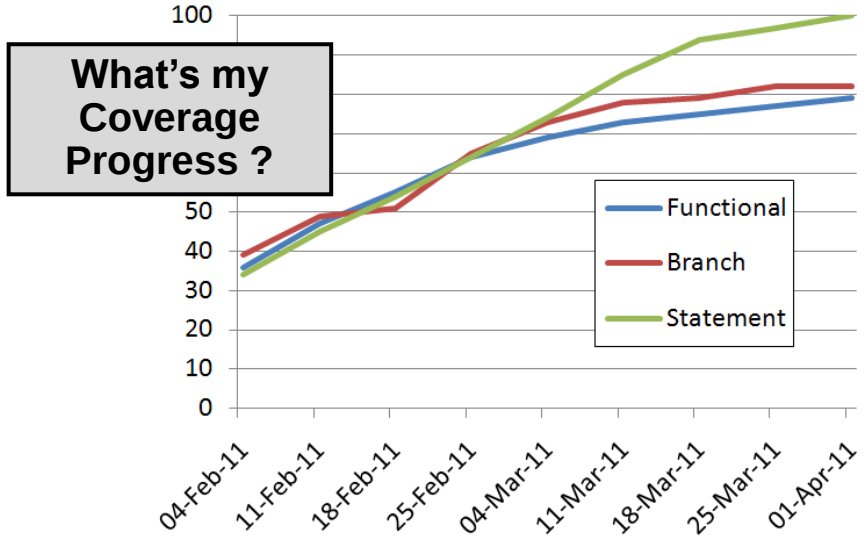


SIEMENS

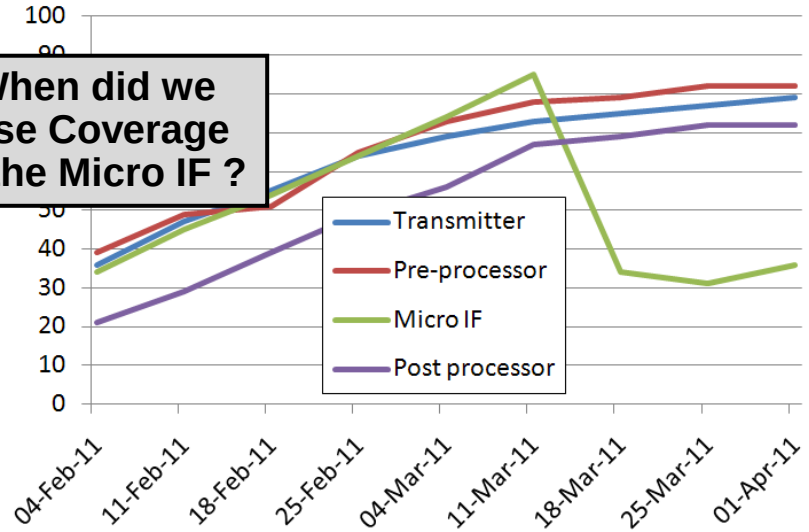
# Powerful analysis

*Who, What, When, Where, Why?*

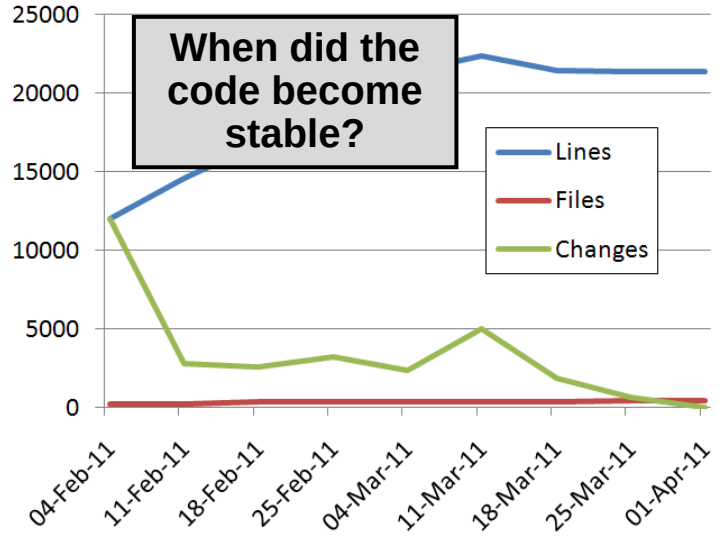
**What's my Coverage Progress ?**



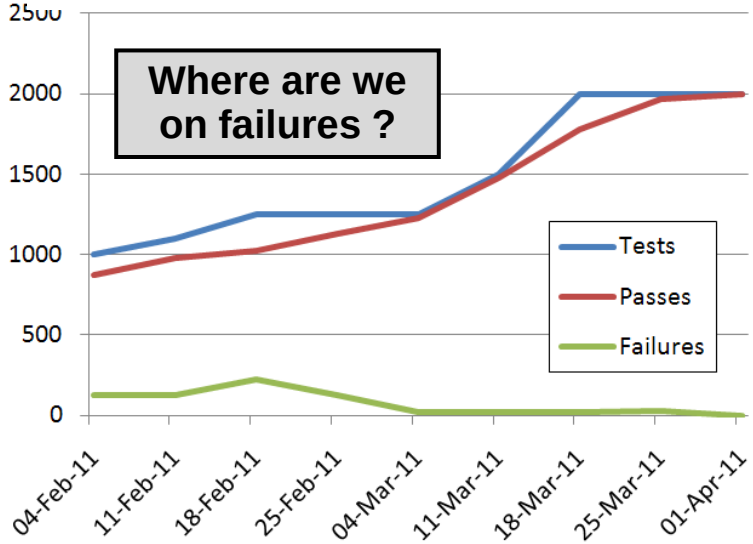

**When did we lose Coverage in the Micro IF ?**




**When did the code become stable?**



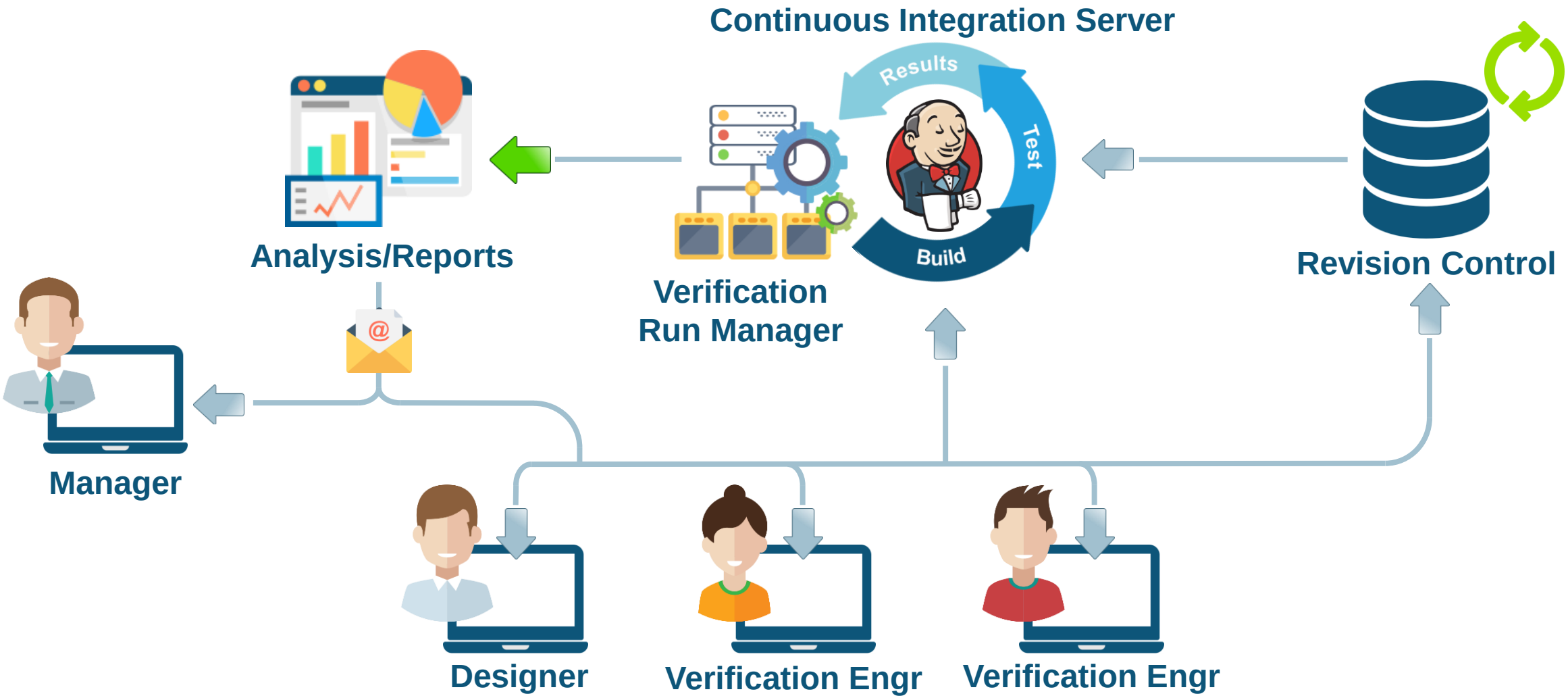

**Where are we on failures ?**






SIEMENS

# Complete regression flow



## Once intent is proven, ensure it is never broken with continuous integration

Run tools early and often to gain time-of-error insight

### Integrate analyses into continuous integration flows

- Protect from faulty check-ins
- Protect from conflict errors

### Build increasing rigor of checks into build stages

- Light high-value checks for check-ins
- Deeper checks prior to daily and weekend regressions
- Deepest checks prior to emulation/prototyping builds

**Continuous Integration**



SIEMENS

# Siemens DVT's Enterprise Verification Platform components

## Verification engines

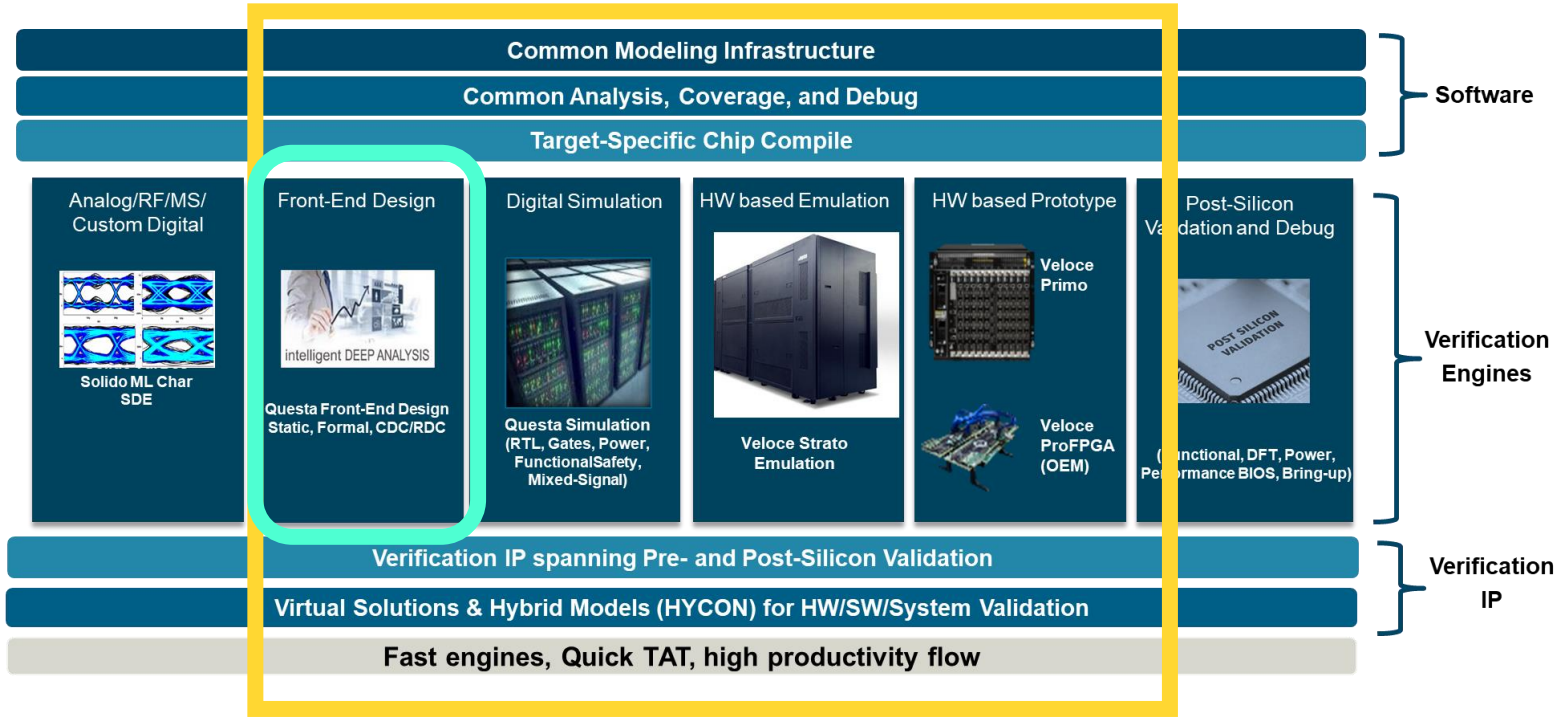
- Questa Simulation
- Questa Formal
- Veloce HW-Assisted Verification

## Platform technologies

- Questa Visualizer Debug
- Questa Verification IQ Coverage
- Portable Stimulus / Verification IP

## Extended verification

- Questa Design Solutions
- Functional safety
- Cloud solutions




**Questa Lint, AutoCheck, X-Check, CDC, RDC**



SIEMENS

# Disclaimer



Subject to changes and errors. The information given in this document only contains general descriptions and/or performance features which may not always specifically reflect those described, or which may undergo modification in the course of further development of the products. The requested performance features are binding only when they are expressly agreed upon in the concluded contract.

# Contact

Published by Siemens 2022

**Mark Eslinger**
Questa Design Solutions Product Manager
DVT (IC Verification Solutions)

E-mail: mark.eslinger@siemens.com



SIEMENS