Leveraging ML/AI in OneSpin: Use cases and opportunities

Yassine Eben Aimine

DV Club 23 Nov 2021

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.

Agenda

Introduction

ML design verification using OneSpin:

- SystemC formal verification
- FPU app

ML in OneSpin

Conclusion

SIEMENS

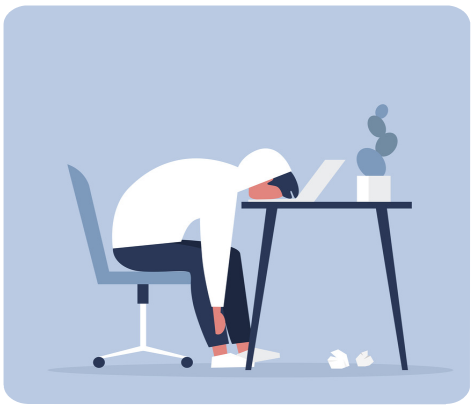
Introduction

Artificial Intelligence: Threat or Opportunity

- **What is Artificial Intelligence:**
  - Ability of non-organic life to perform tasks often attributed to human intelligence
  - Typically by silicon chips

- **Great opportunity**
  - Let the machine do the work

- **Threat:**
  - No need for engineers



- **Bigger Threat:**
  - AI to control human life

SIEMENS

What is Machine Learning?

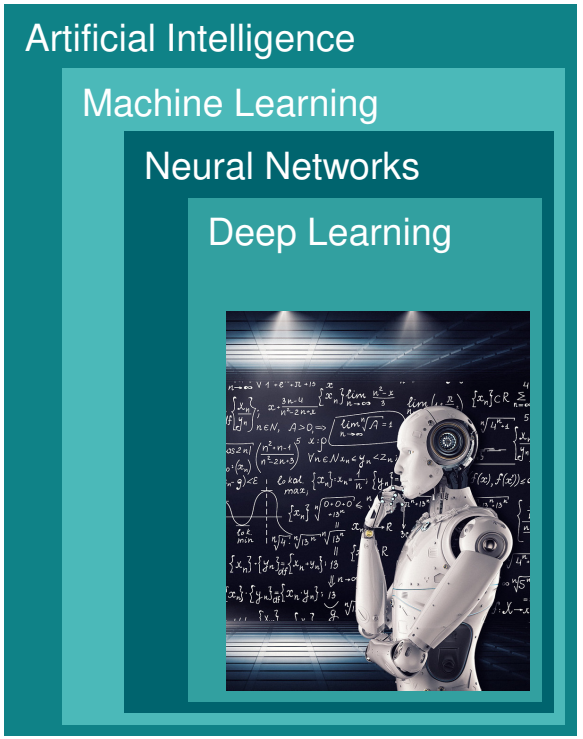
Process by which systems can learn from data, to recognise patterns.

Two-step Machine Learning…

- Use a Deep Neural Network model, configured to find patterns
- Score each outcome and adjust the model parameters to improve accuracy
  → ***This is called TRAINING***
- Test the model with similar data sets to measure accuracy
- Deploy and use the model on real data with the expectation of similar accuracy
  → ***This is called INFERENCE***

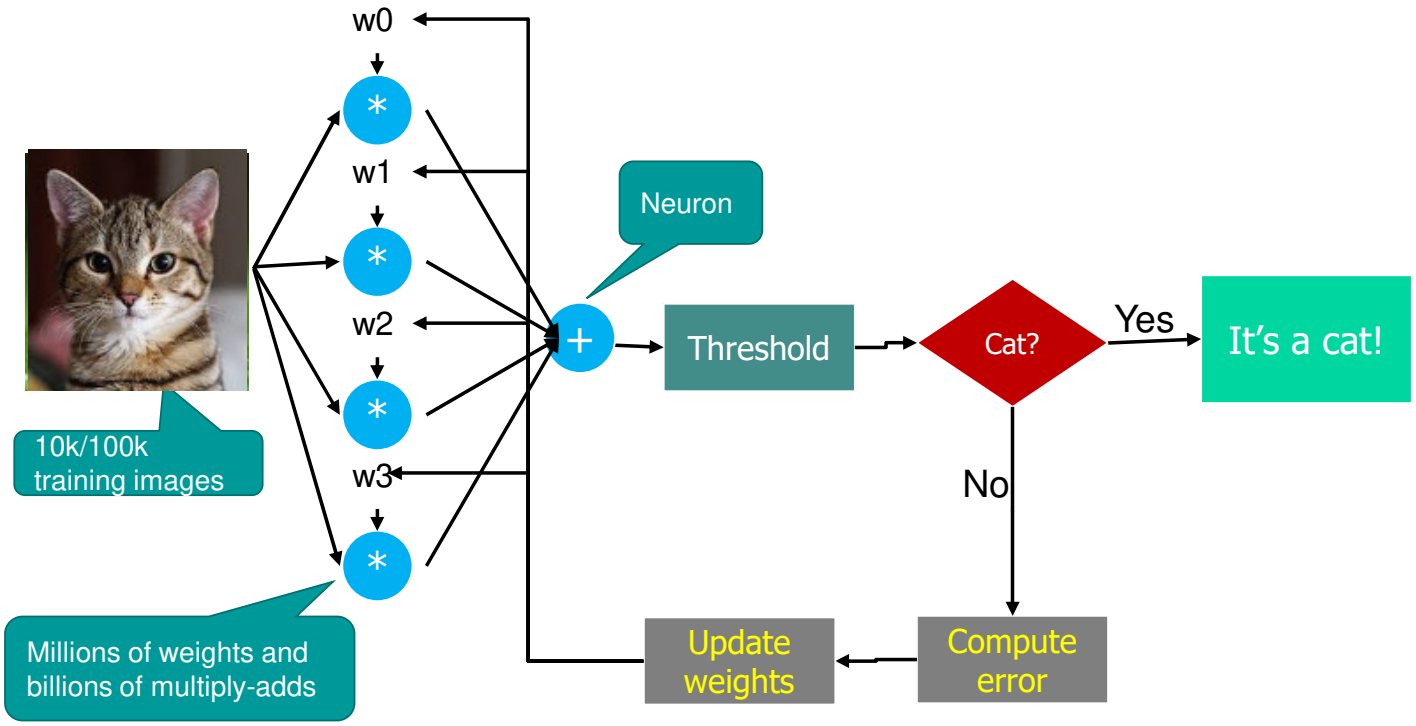
Many use-cases where learning patterns from data can be useful

- Image classification
- Object detection, tracking or alignment
- Face or person detection or recognition
- Speech recognition
- EDA!

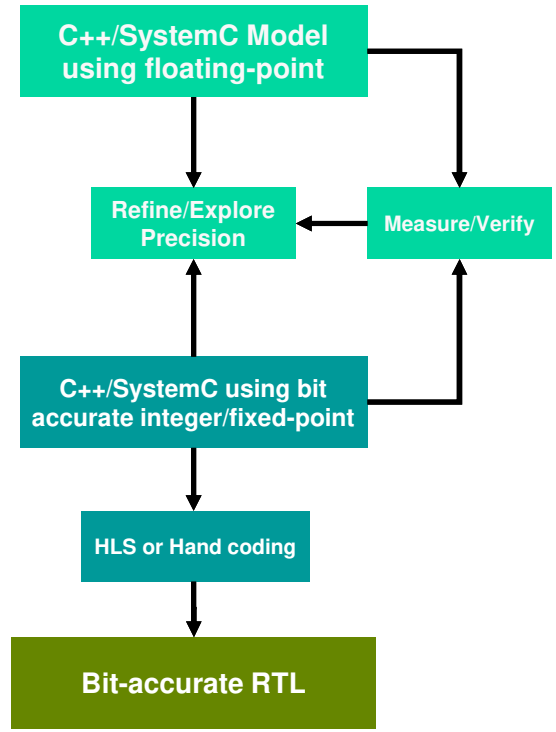


SIEMENS

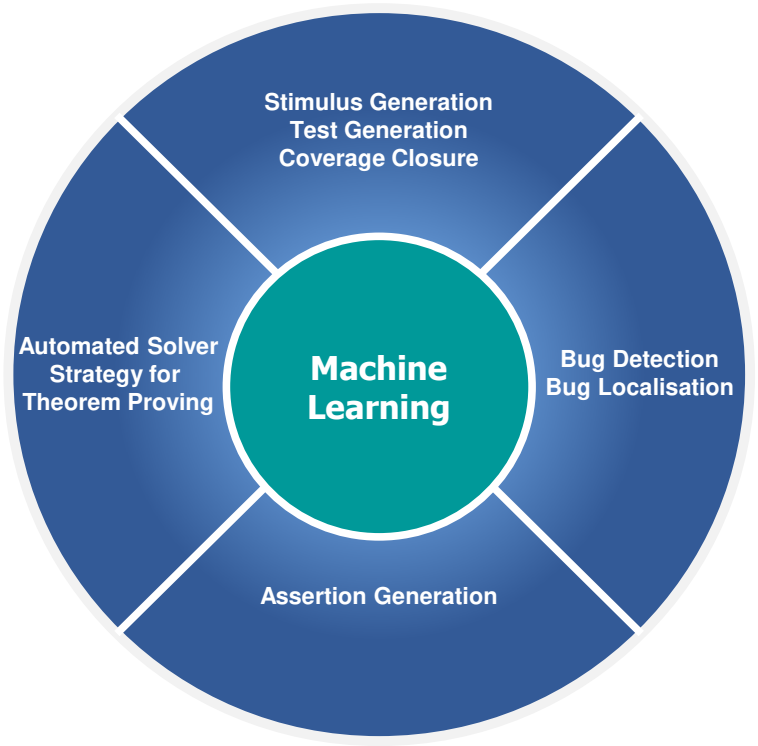
Basic ML Neural Network:



Convolutional Neural Network (CNN) HW design

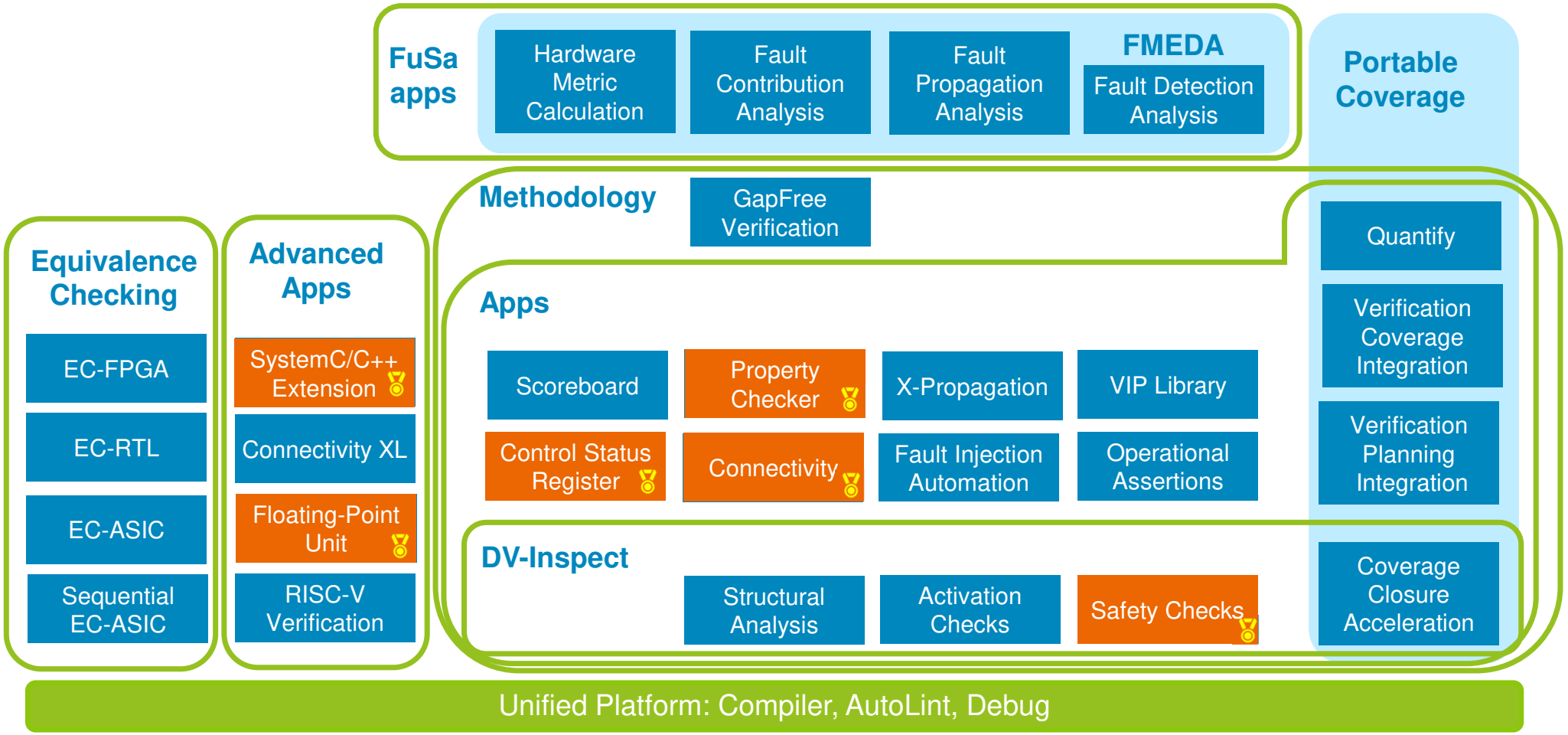


ML applications in EDA - Survey



- Stimulus and Test Generation / Specification and constraint mining:
  - Suggest constraints and assertions to decrease the time of verification cycle.
  - Coverage Closure: Identify holes and tweak stimulus
- Bug Detection and Localisation / Root cause analysis:
  - Learn from previous database of failures and their root causes
  - automate the process of tracing waveforms to find the root cause of a bug
- Automated solver strategy for theorem proving:
  - Beyond prover orchestration
  - Automate process of discovering relationship between logic and successful algorithm for proofs
- Assertions Generation

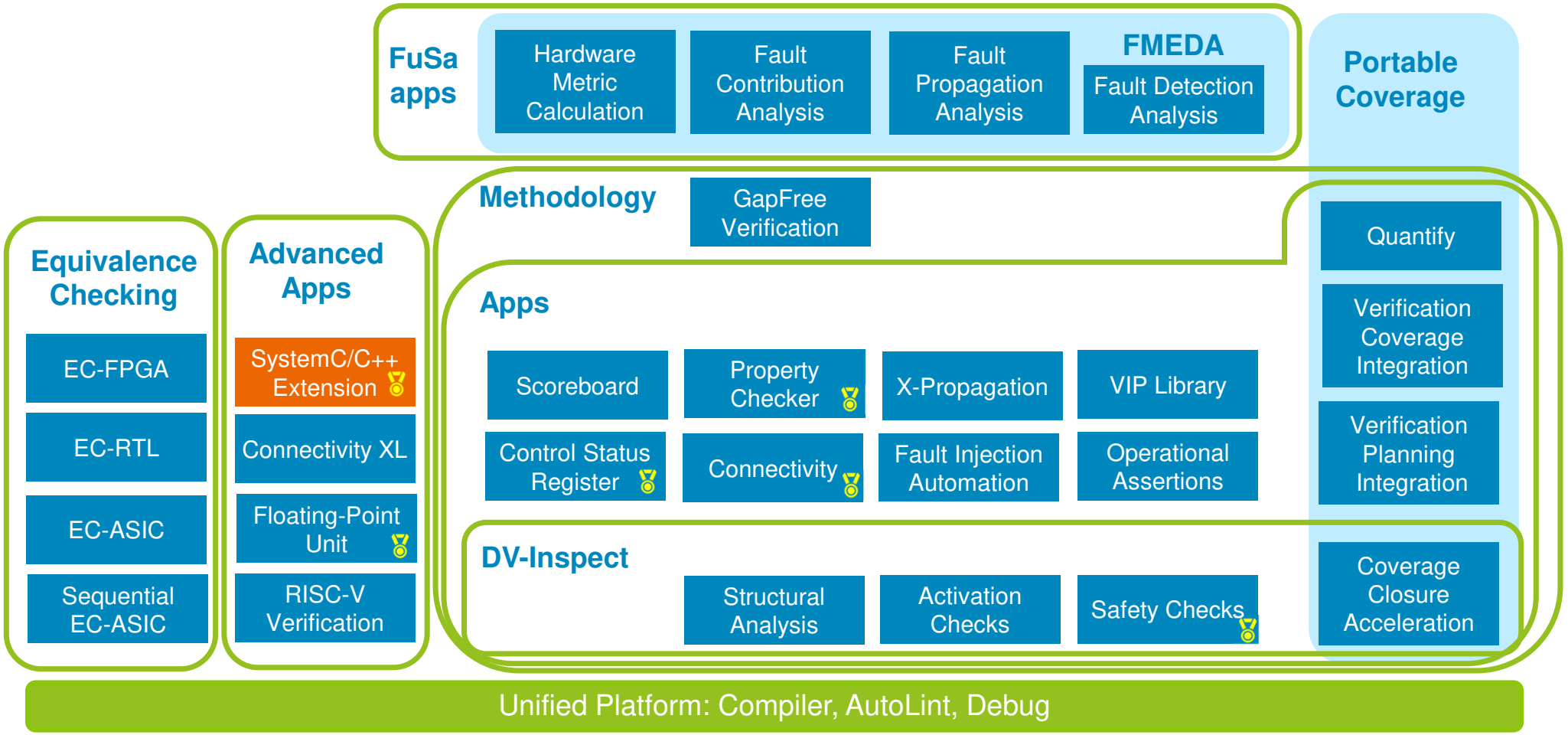
OneSpin 360 Products: Apps for ML, and ML in apps



OneSpin for ML designs

SIEMENS

Apps for ML designs



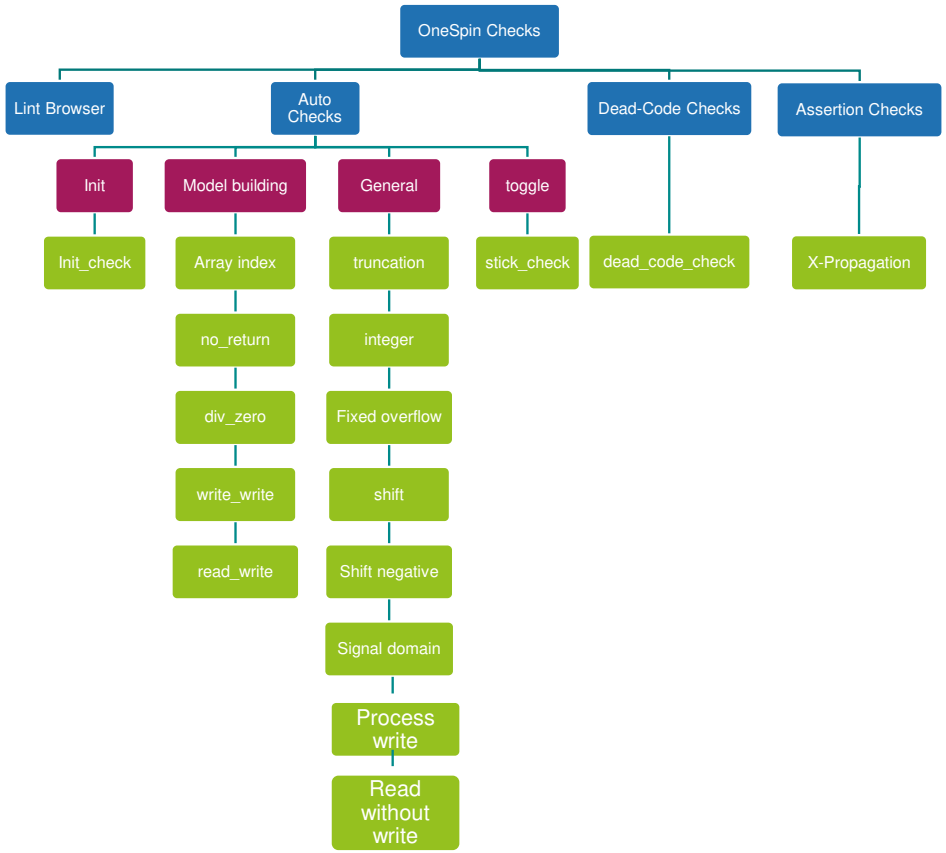
Design Verification for C++/SystemC HLS Code

Run automatic checks on your ML algorithm

Benefits

- Eliminate design bugs before HLS synthesis
- Push-button flow (no need for stimulus, neither testbench)
- Start formal verification much earlier in the process
- Reduce simulation effort in SystemC and RTL
  - Functional simulation in SystemC/RTL for functional coverage
  - Formal verification for completeness
- Optimize HLS input code before synthesis

OneSpin 360 DV-Inspect for SystemC

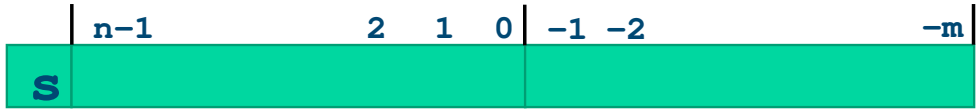


SIEMENS

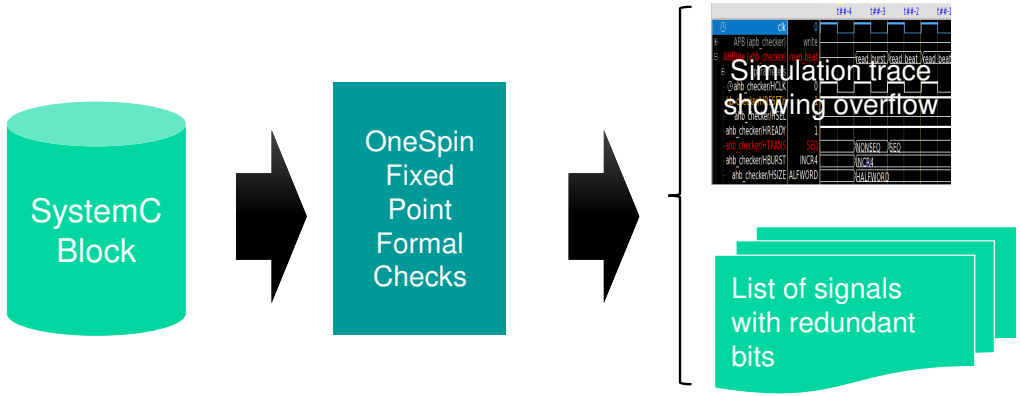
Fixed Point Precision Verification

- Challenge: find “right” bit widths
  - Too many bits: unnecessary complexity
  - Too few bits: overflows and functional errors
- Hard to verify bit widths using simulation
  - Too many possible combinations

sc_(u)fixed data types



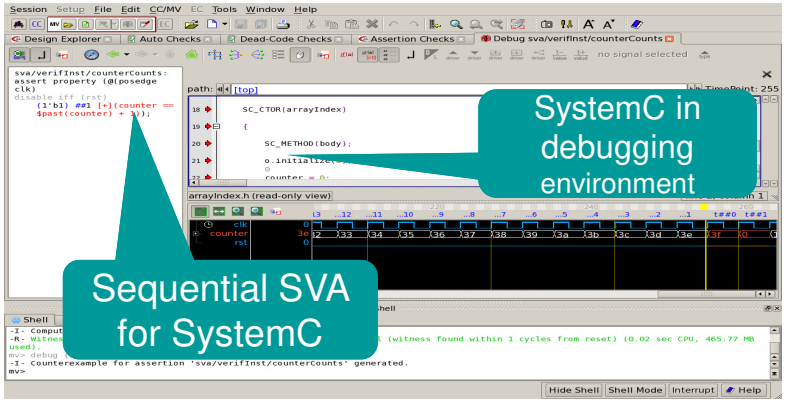
- Check for overflow
  - Check all operations for signed/unsigned overflow
  - Overflow is functional failure
  - Full automation, no need for stimulus
  - Prove absence of overflows
  - Show traces of overflow scenarios
- Check for redundant bits
  - Checks uppermost bits for redundancy
  - Automated, no need for stimulus

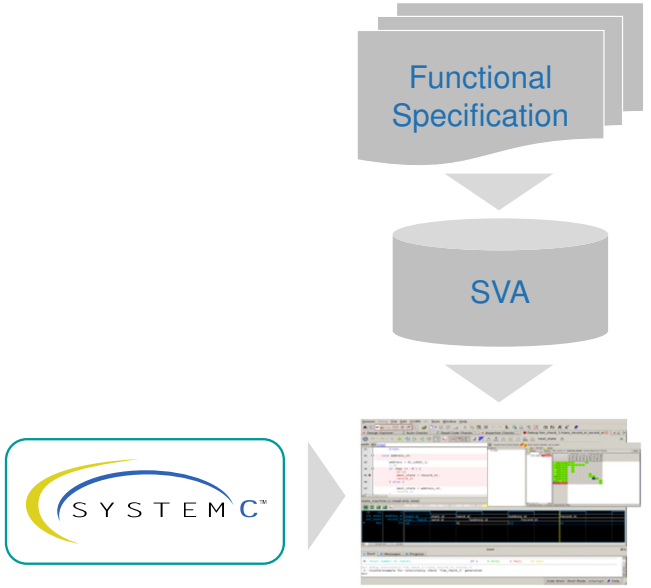


SIEMENS

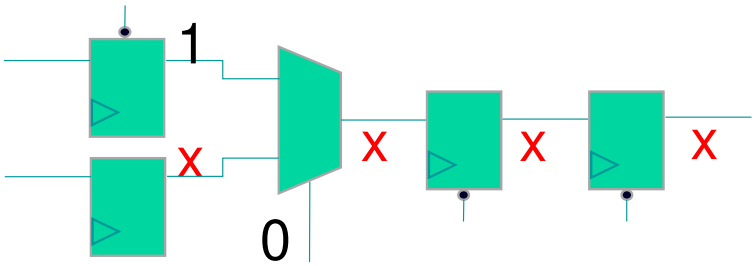
SystemC Property Checking / X propagation

- Leverage SVA and C-asserts in SystemC/C++ designs
- Consistent SystemVerilog assertions pre- and post-synthesis



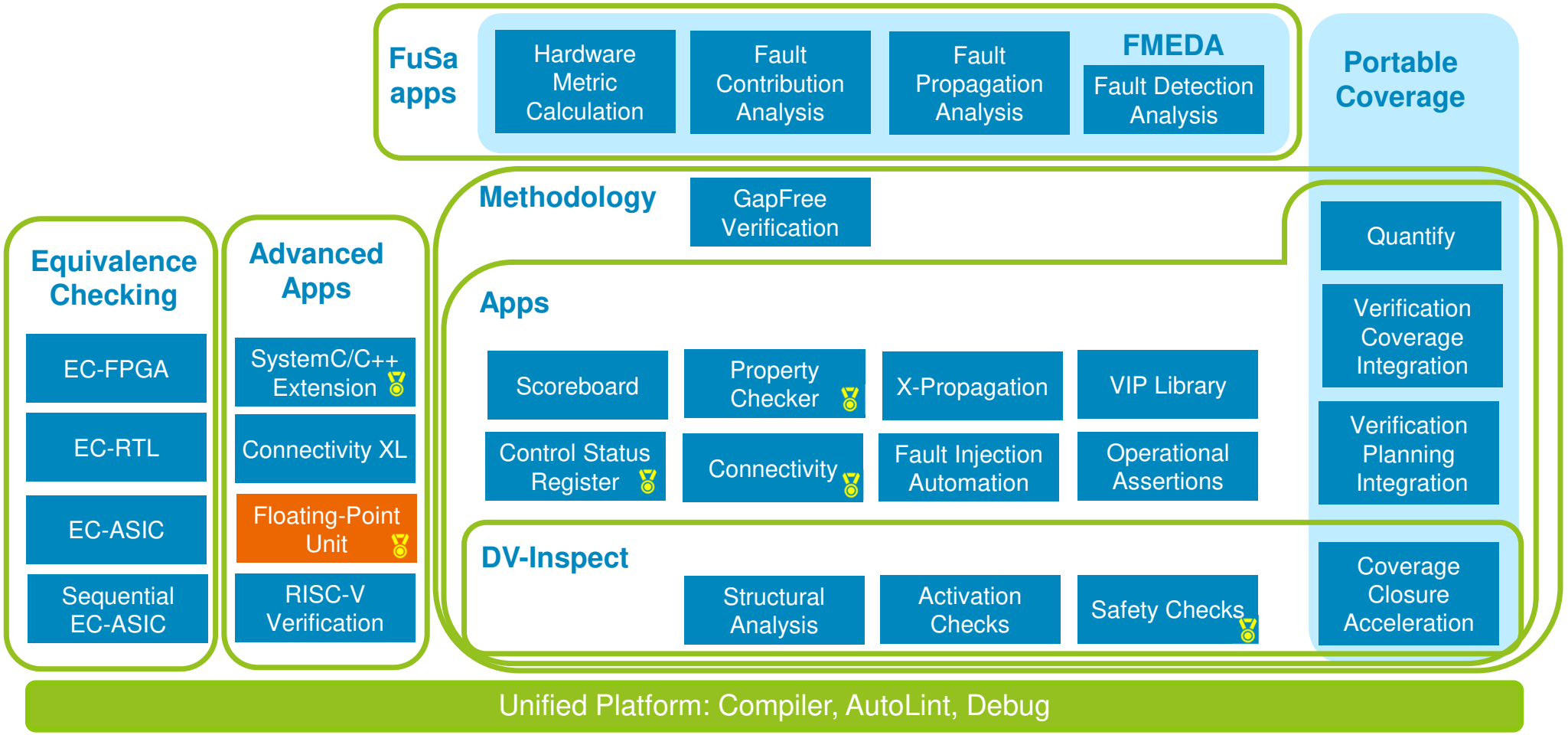


- SystemC is not immune to Xs:
  - Uninitialized registers
  - Undefined operations
  - Uninitialized RAM
- SystemC has no notion of undefined values/RTL semantics
- OneSpin provide formal x propagation for SystemC



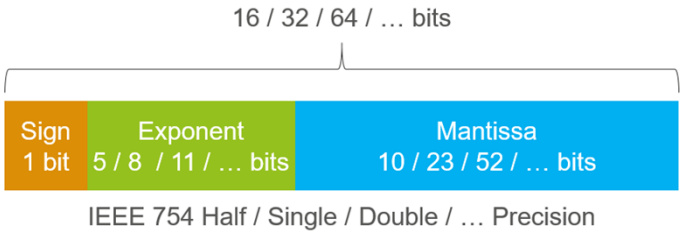
SIEMENS

Apps for ML designs



OneSpin FPU App

- Floating-point computations increasingly important for AI applications
- IEEE 754 FP specification very complex
  - Arithmetic and comparison operations
  - Half, single, and double precision operands
  - Four rounding modes, five exception flags
  - Many special cases (denormals, NaN, ±∞, ±0, etc.)
- OneSpin FPU app can formally prove correctness
  - No need to develop reference models, testbench, or test cases
- Supports all operands, rounding modes, and exception flags
- Includes conversion and comparison functions
- Supports half/single/double precision, bfloat16 (used in machine learning)
  - Custom precision easy to handle by configuring bit widths



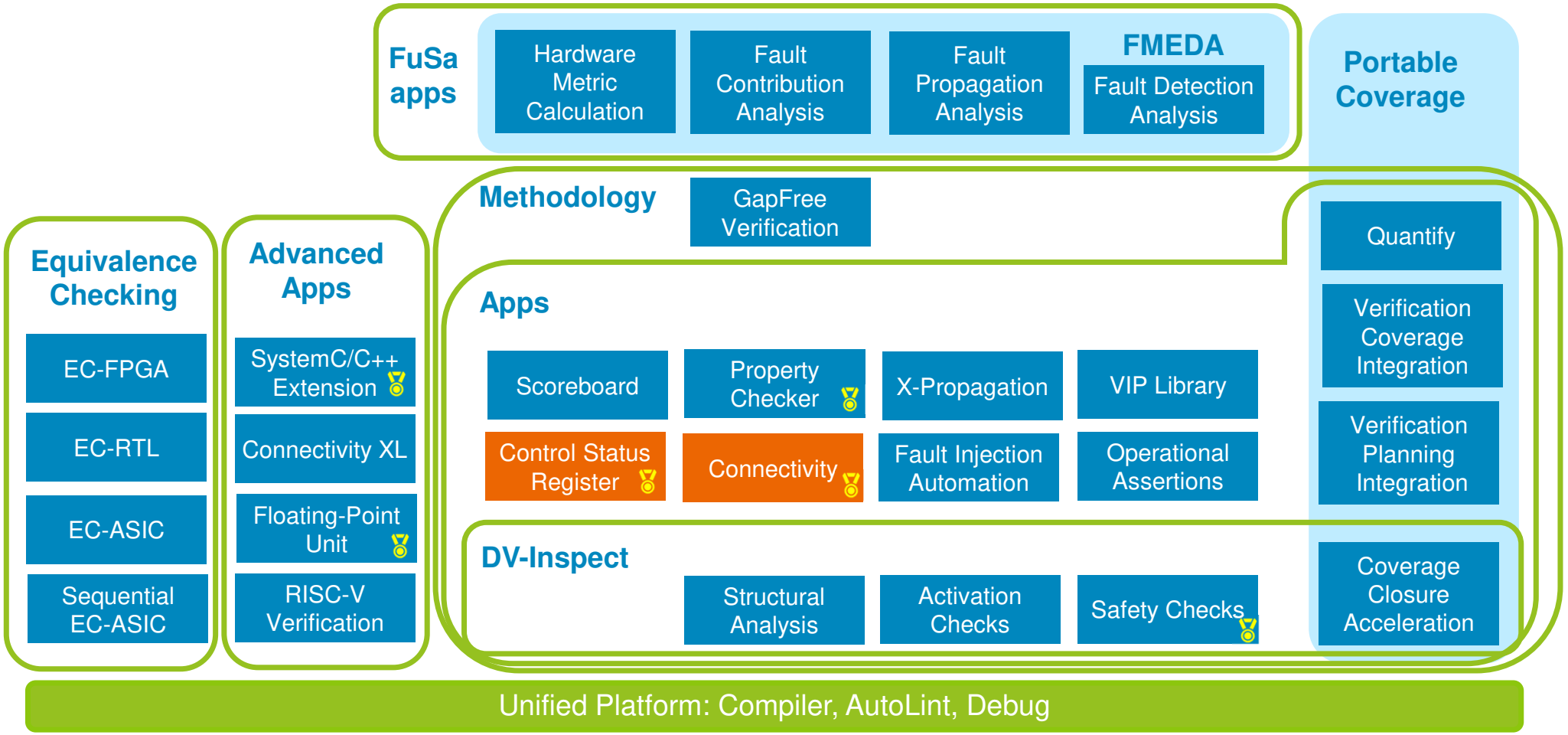
Opcode	# bugs	Runtime	Result
FADD	0	3 minutes	Full proof
FSUB	0	1 minute	Full proof
FMUL	1	4 minutes	Full proof

events.dvcon.org/2018/proceedings/papers/11_3.pdf

ML in OneSpin

SIEMENS

ML in the OneSpin 360 Products



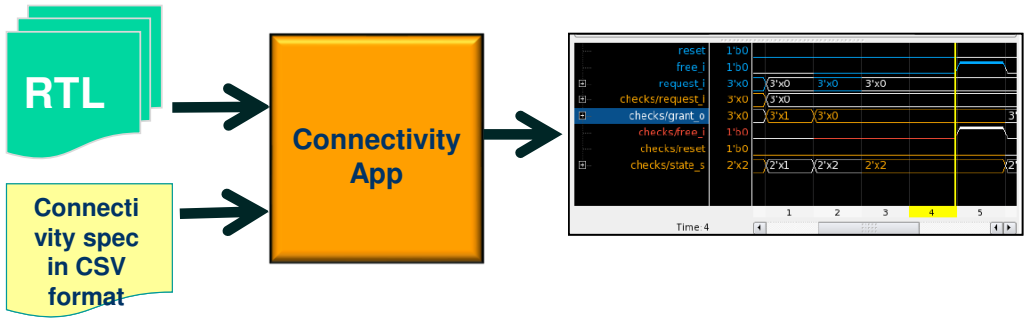
OneSpin Connectivity and Register Apps

No need to write testbenches

Connectivity definition in a spreadsheet (CSV format).

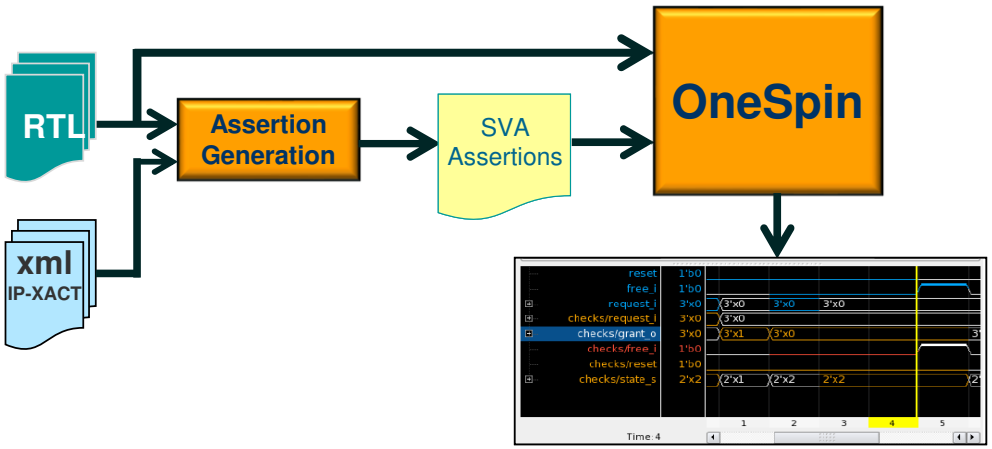
- Connectivity spec (CSV format) can be manually defined
- or generated by other tools

Automatically translates connectivity spec to assert properties



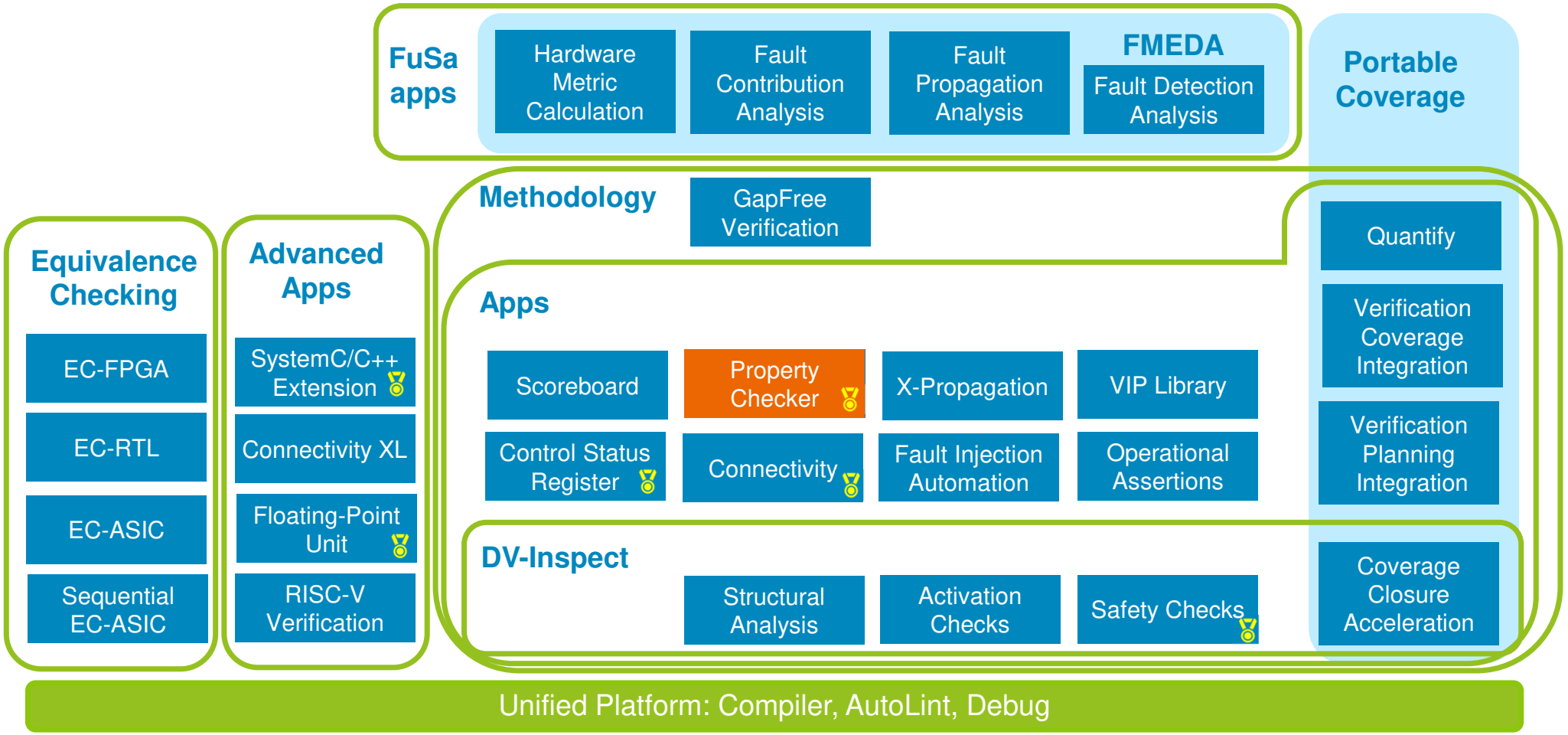
Read the RTL and Register description file

Creates assertions for Memory Mapped Register verification



SIEMENS

ML in the OneSpin 360 Products



Machine Learning in OneSpin

Machine Learning paradigms:

- Unsupervised Learning
- Supervised Learning
- Reinforcement Learning

ML algorithms in the property checker include:

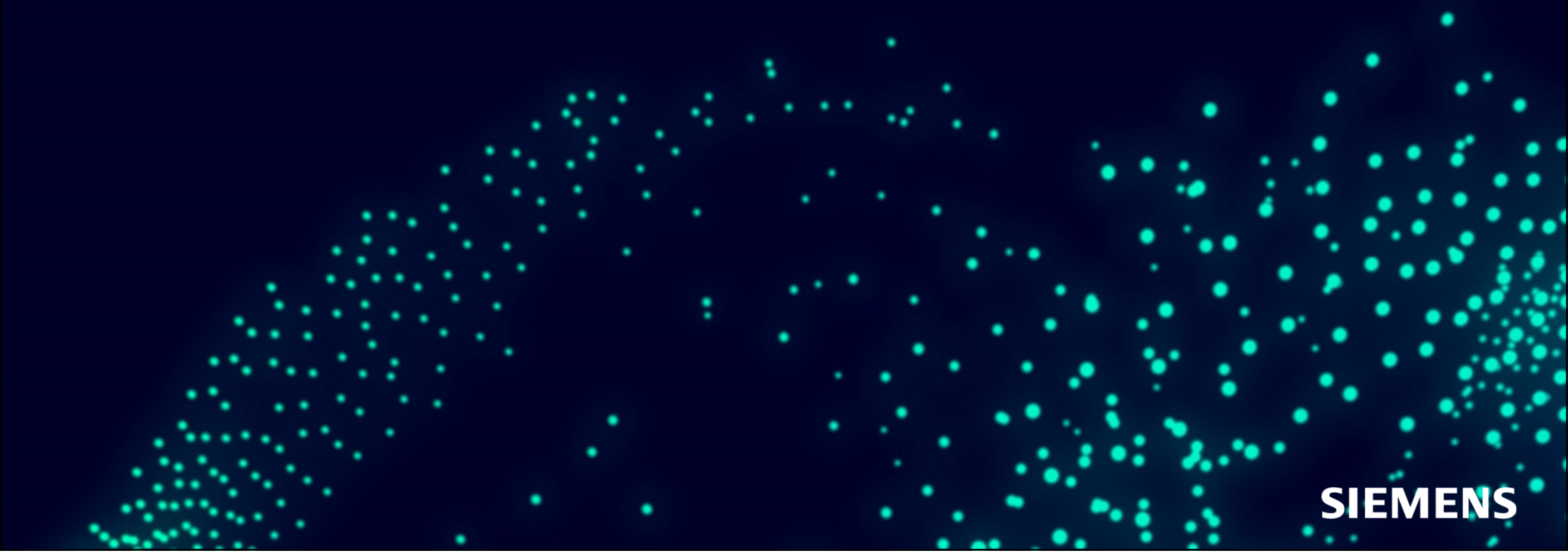
- Case-splitting heuristics in SAT solvers
- Proof caching across solvers/properties
- Invariant generation and generalization
- Abstraction-refinement algorithms

Regression Mode

- Significantly reduce re-run time:
  - With limited changes in design or properties
  - Learn essential engine info and re-use

- Improve results with learning:
  - retry_proof
  - Leverage learnt design structure and proof results

Conclusion



SIEMENS

ML Use cases and opportunities

Contact

Published by Siemens EDA

Yassine Eben Aimine
E-mail yassine.eben_aimine@siemens.com

SIEMENS