2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Formal Verification of Floating-Point Hardware with Assertion-Based VIP

Ravi Ram, Adam Elkins, Adnan Pratama – Xilinx Inc.

Sasa Stamenkovic, Sven Beyer, Sergio Marchese – OneSpin Solutions

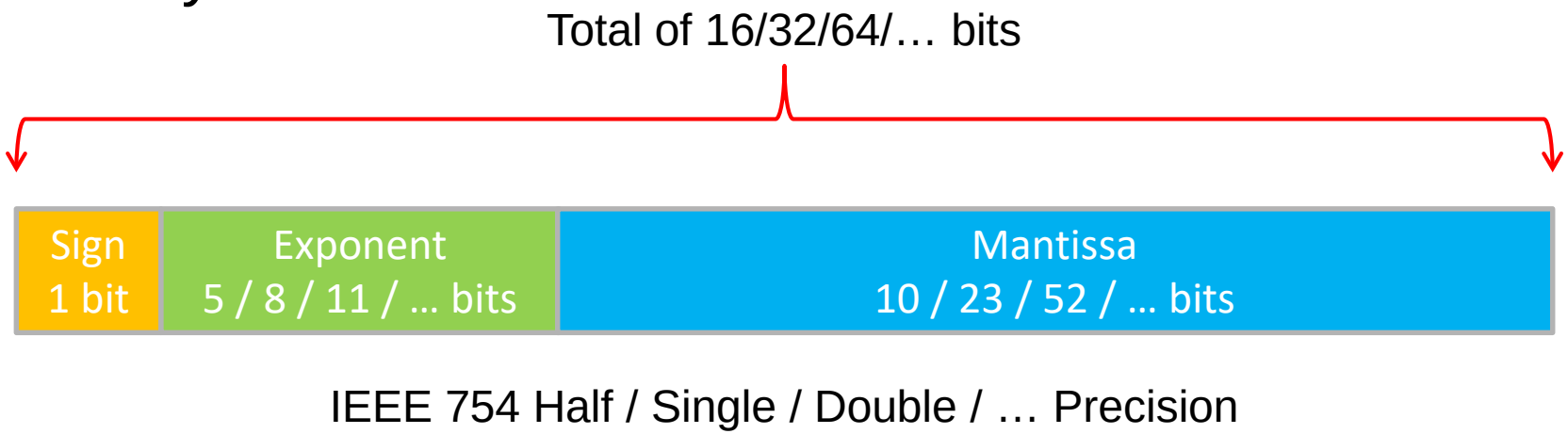
2018 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION UNITED STATES

Agenda

- Floating-point (FP) arithmetic
- Functional verification of FP hardware
- Formal verification with assertion-based VIP
- Results
- Conclusions

Floating-Point Arithmetic

- Compared to fixed-point arithmetic
  - Covers wider range of values
  - No loss of precision, higher accuracy
  - More complex hardware
  - Notoriously hard to verify



Functional Verification

- Simulation misses bugs
  - Exhaustive verification is not feasible
    - One and a half engineer-years to cover all scenarios
  - Many implementation-dependent corner cases
- Sequential Equivalence Checking (SEC) of RTL against reference model
  - Requires detailed understanding of both implementations
  - Reference model (C++/SystemC) needs adaptation
  - High effort, little reusability
- Formal Assertion-Based VIP (ABVIP
  - Xilinx tried but ran into usability and convergence issues
  - Xilinx worked with OneSpin to develop a new solution

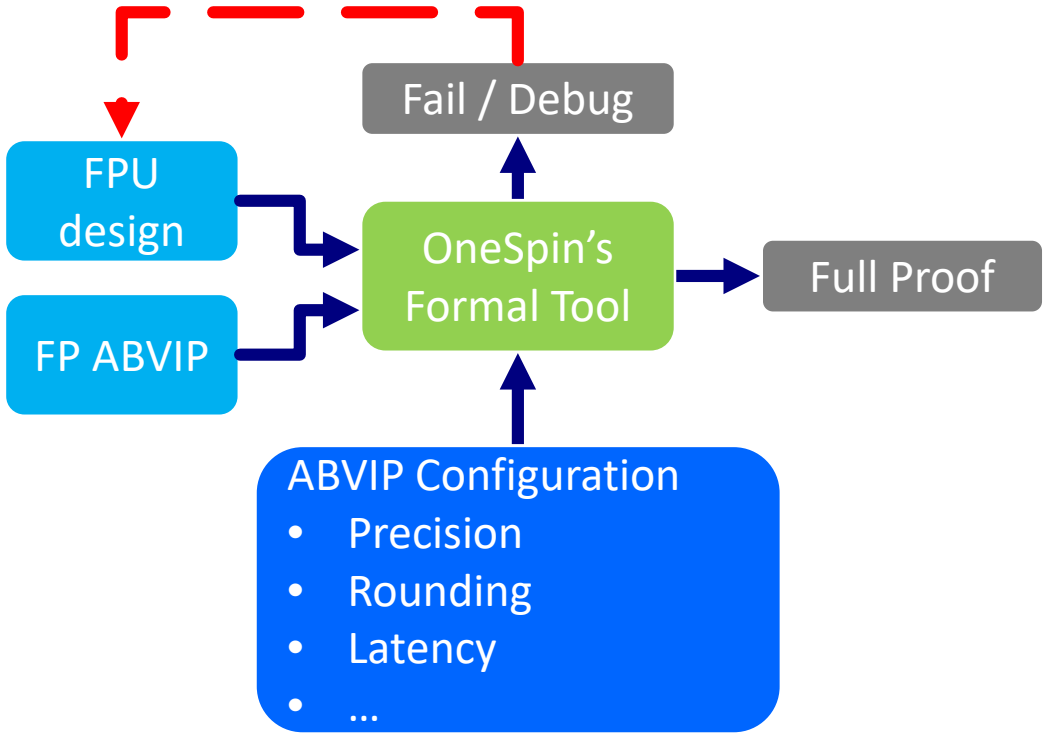
Floating-Point ABVIP

FP ABVIP

- SystemVerilog package
- *ieee_*... functions and data types
- No reference model required
- Minimal design knowledge by end user

OneSpin Tool

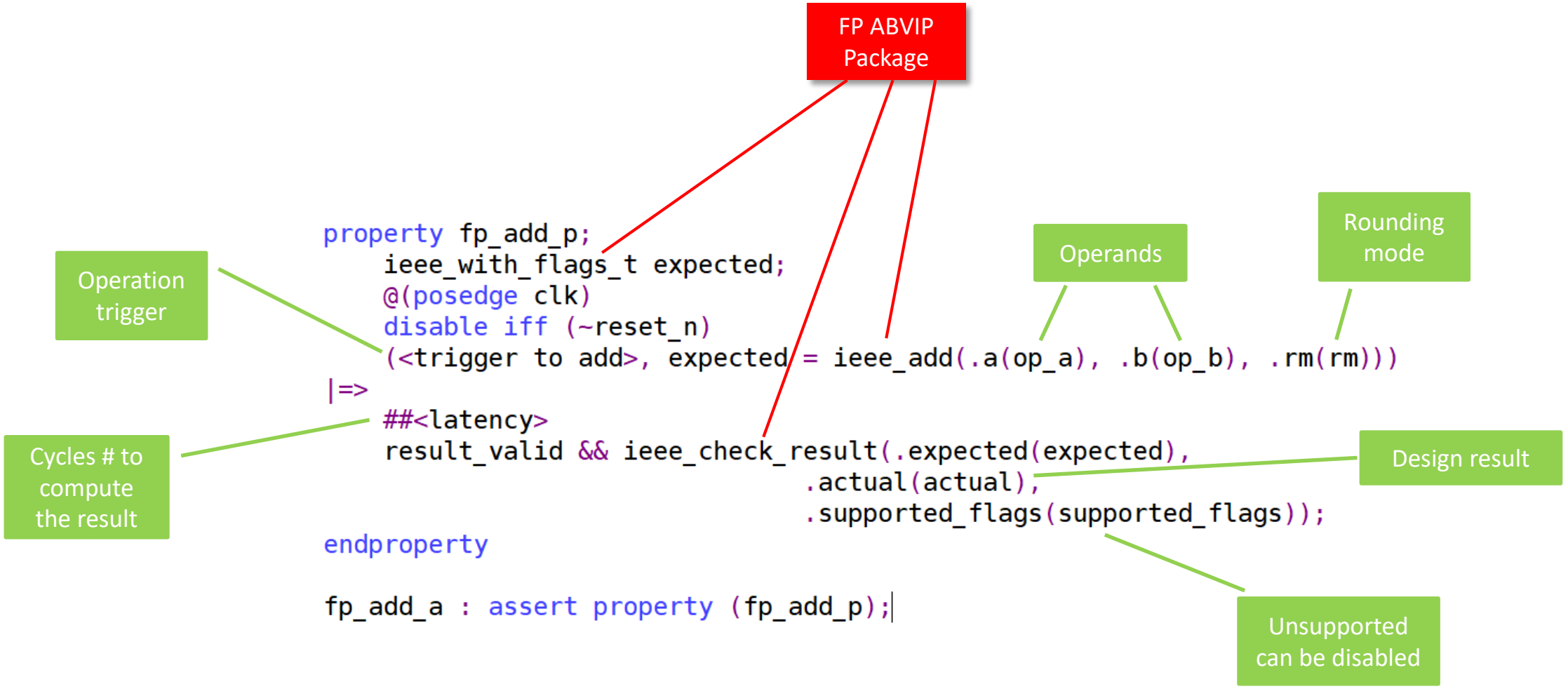
- FP ABVIP available
- Proof engines/strategies for arithmetic
- Debug with FP data types



Floating-Point ABVIP

- Compliant to IEEE-754
- Supports
  - Half, single and double precision formats
  - All the rounding modes and the exception flags
  - Tininess before or after rounding
  - Add, sub, mult, absolute value, negation, and all comparison operations
  - Conversion functions also included
- Customizable
  - Custom precision
  - Intended deviations from standard

FP ABVIP Property Template



Xilinx FPU

- Supports addition, subtraction and multiplication
- Tool found a previously undiscovered bug in the module interface constraints
- General and specific scenarios assertions created
  - e.g. operations with signalling or quiet NaN
- Design bugs previously found in simulation and emulation
  - FP ABVIP found them within seconds

Results

- Open Cores

Operation	# bugs	Setup effort	Runtime	Result
FADD	0	30 minutes	**52 seconds**	Full proof
FSUB	1	5 minutes	**1 minute to find a bug**	Fail
FMUL	2	5 minutes	**1 second to find a bug**	Fail

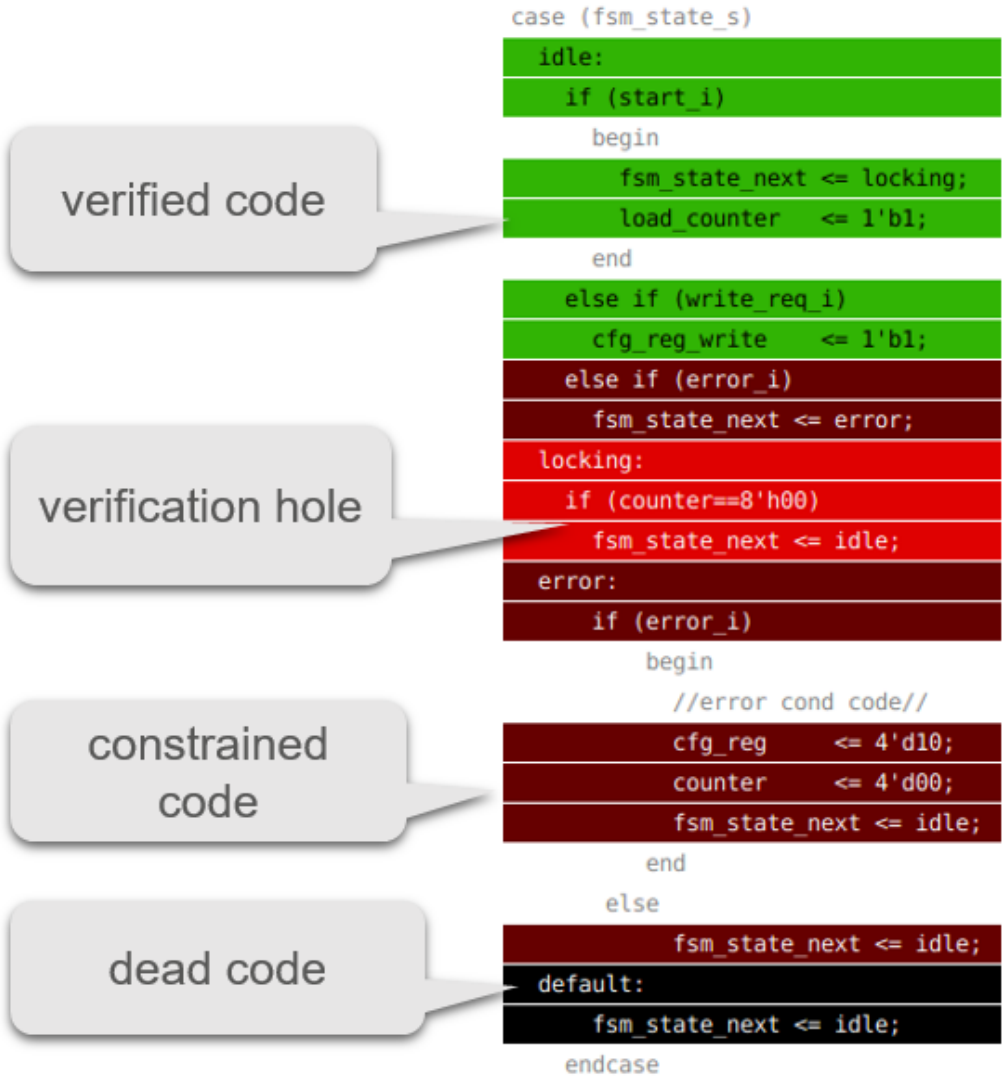
- Xilinx FPU

Operation	# bugs	Setup effort	Runtime	Result
FADD	0	4 days	**3 minutes**	Full proof
FSUB	0	3 days	**1 minute**	Full proof
FMUL	1	15 days	**4 minutes**	Full proof

Tool and FPU App familiarization and constraints setup

Formal Coverage

- Metric-driven verification
  - OneSpin Quantify
- Answers
  - How much has been verified?
  - What is the next assertion to write?
  - Is my design over-constrained?
- User written assertions and covers
- Overall coverage ~90%
- Holes point to logic not contributing floating-point operations



Formal coverage results

Verified with assertions

Structural Coverage Overview

Status		Statements		Branches	
1	covered	185	99.46%	15	51.72%
R	reached	0	0.00%	0	0.00%
U	unknown	0	0.00%	2	6.90%
0R	unobserved	1	0.54%	0	0.00%
0	uncovered	0	0.00%	0	0.00%
0C	constrained	0	0.00%	0	0.00%
0D	dead	0	0.00%	12	41.38%
Sum	**quantify targets**	**186**		**29**	

Excluded Code Overview

Code Status		Statements		Branches	
Xu	excluded by user	697	75.84%	286	89.38%
Xr	excluded redundant code	36	3.92%	5	1.56%
Xv	excluded verification code	0	0.00%	0	0.00%
0/1/U	quantify targets	186	20.24%	29	9.06%
Sum	**total code**	**919**		**320**	

Conclusions

- FP ABVIP is compliant with the IEEE 754 standard
- Solution is easy to set up and use
- Excellent experience for Xilinx FPU project
  - Low effort
  - Uncovered corner-case bugs within seconds
  - Exhaustive verification with full proofs, within minutes
  - High coverage
- Current limitations
  - No support for iterative operations (division, square root)
  - Additional effort may be required to achieve full proofs