Taming Formal to define the Verification Quality

Surinder Sood, ARM Ltd, Manchester UK, *surinder.sood@arm.com*

Kishan Mushar, ARM Ltd , Manchester UK, *kishan.mushar@arm.com*

arm

SPONSORED BY CEDA IEEE Council on Electronic Design Automation sigda acm

Agenda

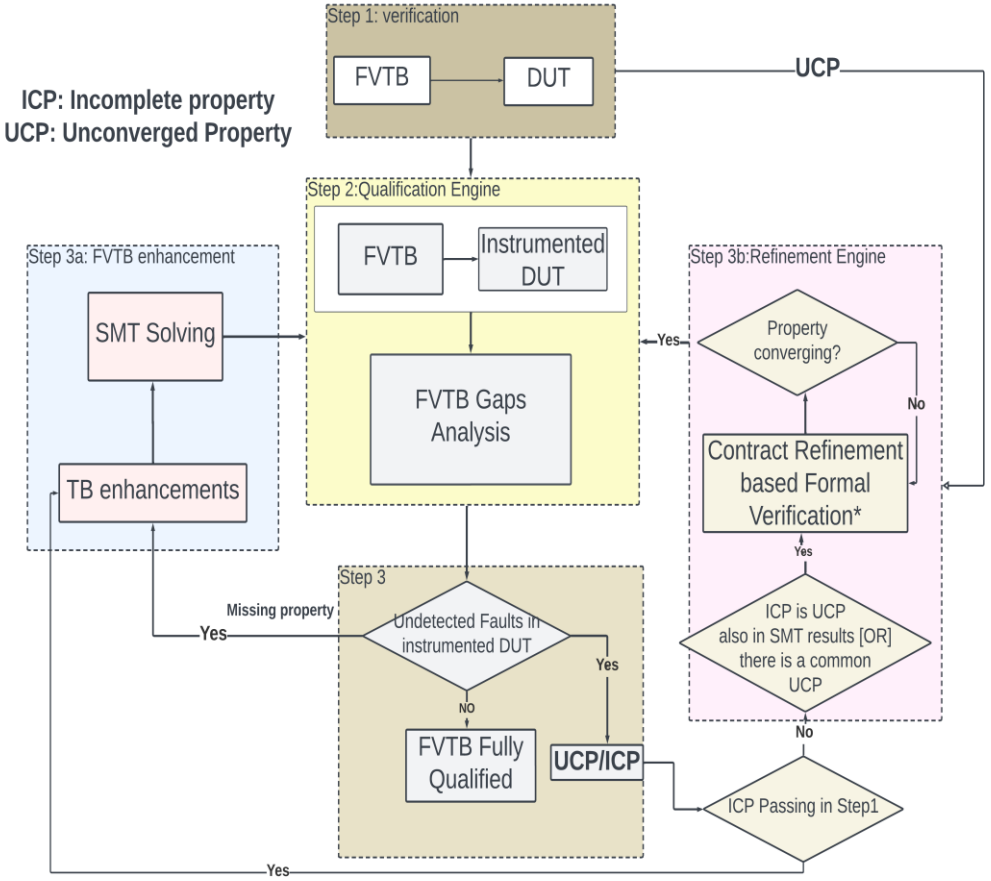
- Motivation and Problem statement
- Main Idea
- Evidence
- Summary

Motivation and Problem statement

- There is no technique available to verify specific system behavior which should guarantee
  - Completeness
  - Correctness
  - Consistency
- Lack of 3Cs in formal environments lead to poor verification quality[1]
- Moreover, Proof convergence for all the properties is difficult to achieve using available state-of-the-art formal techniques.
- As the designs scale, convergence of proofs become more challenging, questioning verification quality.

[1] https://dvcon-proceedings.org/document/lets-be-formal-while-talking-about-verification-quality-a-novel-approach-to-qualify-assertion-based-vips/

Main Idea: Property refinement based formal verification Qualification



Step 1: We verify the DUT using formal verification testbench. Any un-converged properties are passed to the refinement engine

Step 2: We instrument the design and analyze the faults not detected by formal verification testbench

Step 3: If there are no Undetected faults Formal testbench is fully qualified and is able to detect all the issues in the design, else go to Step (3a) or (3b)

Step 3a : If there are any missing properties, we write them and try to converge those

Step 3b: If there is ICP which is not converging in Step 1, [OR] there is UCP in step 1 and step 2We do contract-based decomposition and refinement[1,2]] of the UCP

[1]A. Cimatti and S. Tonetta, "Contracts-refinement proof system for component-based embedded systems," Science of computer programming, vol. 97, pp. 333–348, 2015.
[2]A novel formal verification technique to System verification using contract refinement, Surinder Sood, Nirmal Jose and Scott Meeth, "DAC 2024, San Francisco, USA"

Evidence: Formal verification of a Clock and power block [CPB] in SMMU

- CPB manages clock supply and power modes of (TMU) Translation Management Unit in SMMU.
- The CPB not only interfaces with external power and clock controllers but also interfaces with the TMU's constituent blocks once the CPB has decided to powerup, to tell such blocks to perform initialization activities via the dedicated init_req/done interfaces
- Adjacent table shows few of the issues in the verification testbench which were fixed by applying the proposed methodology

Design Issue	Property	Property Status(UCP/ICP)	Comments
Output signal Stuck at 0	The power down request should be propagated to connected blocks and that can only be done if **qaccept** is high	ICP-Converging	XDMCP* protocol, where the signal **qaccept** was stuck at 0. The **incomplete property** passed and does not check complimentary functionality
Output signal stuck at 0	Power down request can only be received when LPIPD is in QREQUEST and PTI Interface Block fence is open.	ICP-Converging	XDMCP* protocol, where the signal **qdeny** was stuck at 0. The **incomplete property** passed and does not check complimentary functionality. So if fence is open the corresponding interface should accept the incoming traffic, and when fence is closed, qdeny should be high.
Output signal stuck at 0	To enable clock gating in a block, a proper start -done sequence should be followed	ICP-Converging	Start-done handshake protocol, where **start** signal was stuck at 0. **incomplete property** passed and does not check complimentary functionality
Output signal stuck at 1	During power down the RAS interface is masked	ICP & UCP	The property to check the mask behavior for RAS was missing, and when we wrote the property it was not converging. We used Refinement solution to make this property converge, with a guarantee that it covers both the scenarios when mask is set or when mask is cleared

Evidence: Formal verification of Clock and power block [CPB]

Snapshot of Faults in the CPB Block						
Fault Category	Faults in design	Non-activated*	Detected	Non-detected	Dropped	Comments
Top outputs connectivity	33	9	18	1	5	Few of the design faults were dropped as they were trivial.
Internal connectivity	87	15			72	Internal faults were not of much relevance to this block
Combo Logic	5	0	5	0	0	
Synchronous control flow	0	0	0	0	0	

- Functional qualification performs a run by injecting faults in the Design Under Verification and executes testcases to measure the ability of the verification environment to detect these faults.
- Weaknesses in the verification can be discovered by analyzing the Non-Activated, Non-Propagated and Non-Detected faults.
- Adjacent table show the results, where it was able to detect flaws in Formal Verification Test Bench.

Evidence: FV of a Translation Routing Block [TRB]

- TRB receives translation requests from different TBUs [or] PCIE agents
- It is the role of the TRB to arbitrate between all different sources to produce a single interface to the Translation management block

Design Issue	Property	Property Status(UCP/ICP)	Comments
Output signal Stuck at 0	The property check which TBU is being selected according to the value of **tdest**	ICP-Converging	dti_txn_rsp_d_tdest, where the signal **tdest** was stuck at 0. The **incomplete property** passed and does not check complimentary functionality
Output signal stuck at 0	The property checks based on the Payld,TID from input and send the TWB req.tid from the req path for the specific TBU	ICP-Converging	txn_req_s_payld.tid,, where the signal **tid** was stuck at 0. The **incomplete property** passed and does not check complimentary functionality .

Snapshot of faults in TRB

Snapshot of Faults in the TRB Block						
Fault Category	Faults in design	Non-activated*	Detected	Non-detected	Dropped /Not relevant	Comments
Top outputs connectivity	72	45	21	2	4	Few of the design faults were dropped as they were trivial.
Internal connectivity	57	0	39	0	18	Internal faults were not of much relevance to this block
Combo Logic	475	175	189	0	111	
Synchronous control flow	4	0	4	0	0	

- Functional qualification performs a run by injecting faults in the Design Under Verification and executes testcases to measure the ability of the verification environment to detect these faults.
- Weaknesses in the verification can be discovered by analyzing the Non-Activated, Non-Propagated and Non-Detected faults.
- Adjacent table show the results, where it was able to detect flaws in Formal Verification Test Bench.

Summary

1. Design instrumentation is a promising approach to qualify formal testbenches
2. When this approach is integrated with our proof convergence technique (deployed on un-converged properties), the results are very promising as this dual approach not only converge the properties of a bigger design but also qualifies our verification.
3. We are researching further on the following aspects of functional qualification:
    1. Instrumenting the design using custom faults which are meant for specific design blocks (like synchronous/asynchronous/pipelined etc..)
    2. AI Assisted functional qualification.
    3. Security verification qualification: Correlating faults with security threats and checking the quality of a formal TB.

AI

Security

Systems

EDA

Design

62 THE CHIPS TO SYSTEMS CONFERENCE

SPONSORED BY CEDA® IEEE Council on Electronic Design Automation

sigda acm