Toward Trusted 3rd-Party Microprocessor Cores: A Proof Carrying Code Approach

PIs: Dr. Yier Jin, Department of Electrical and Computer Engineering, University of Central Florida, yier.jin@eecs.ucf.edu
Dr. Yiorgos Makris, Department of Electrical Engineering, The University of Texas at Dallas, yiorgos.makris@utdallas.edu

The objective of this project is to eliminate the security concern due to the prevailing usage of hardware Intellectual Property (IP) cores from third-parties (untrusted) vendors.

- Existing approaches rarely secure the entire computer system because of the semantic gap between the hardware and the software.
- Existing formal verification frameworks for IP trustworthiness evaluation are often not scalable to microprocessors and SoC designs.

ATTACK MODEL

- Malicious logic/Design faults may be inserted by an untrusted agent at the design stage in the third party IP design house.
- Payloads: sensitive information leakage, design functionality modification, and/or denial-of-service to the hardware.

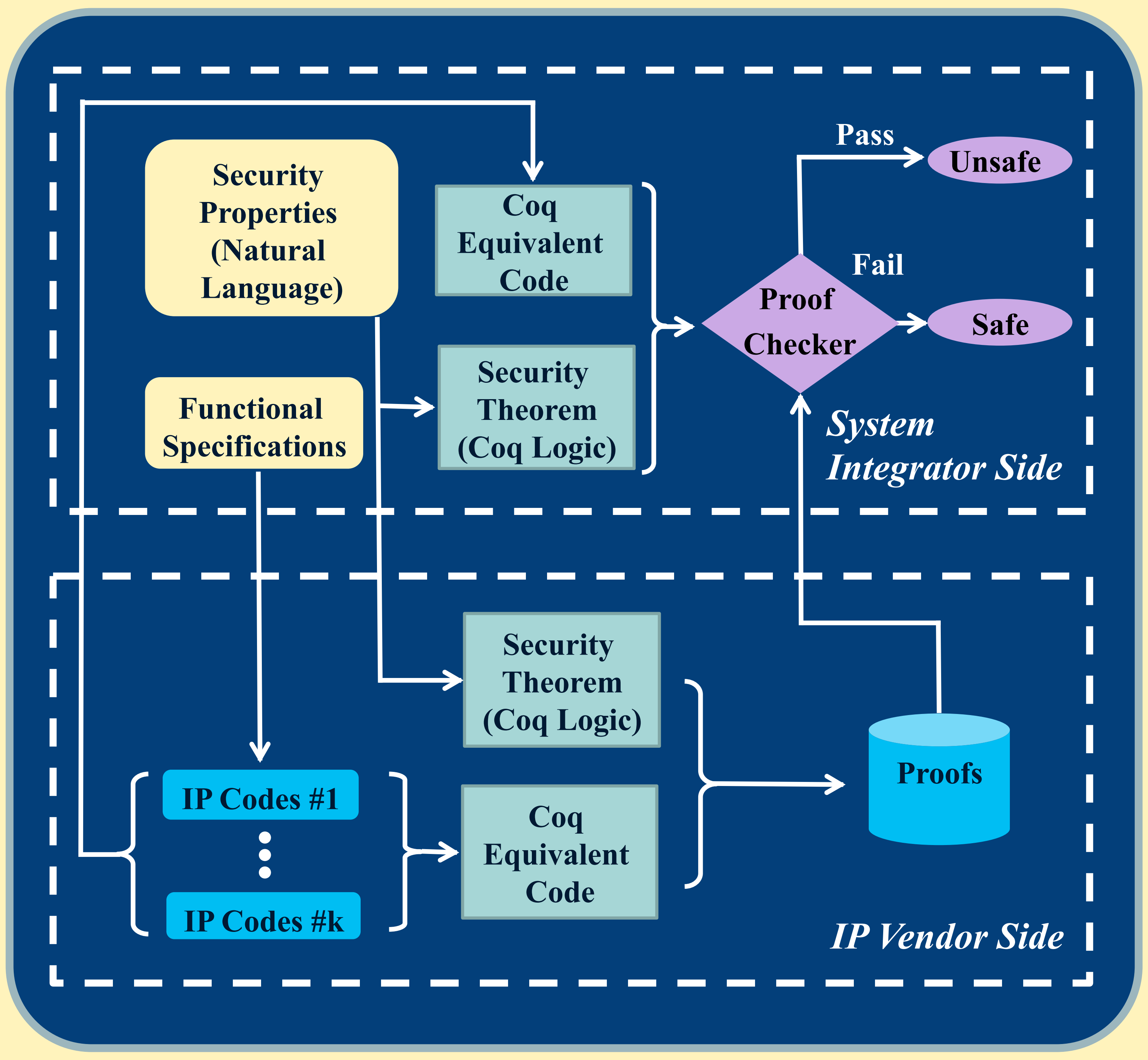


Figure 1. Proof-Carrying Hardware IP Working Procedure.

Approach

Proof-Carrying Hardware (PCH)

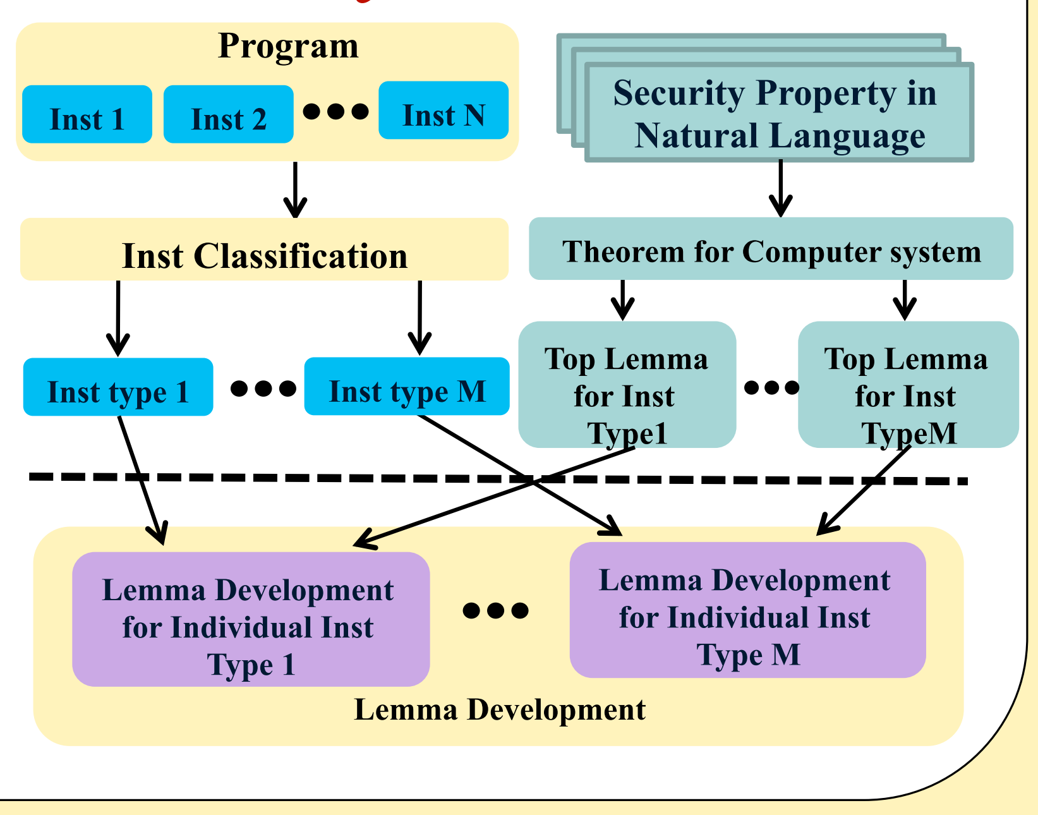
The PCH framework certifies that soft IPs are trustworthy if certain carefully specified security properties hold [1].

A Theorem Prover - Coq

Coq proof assistant is used to represent security properties, hardware designs, and formal proofs.

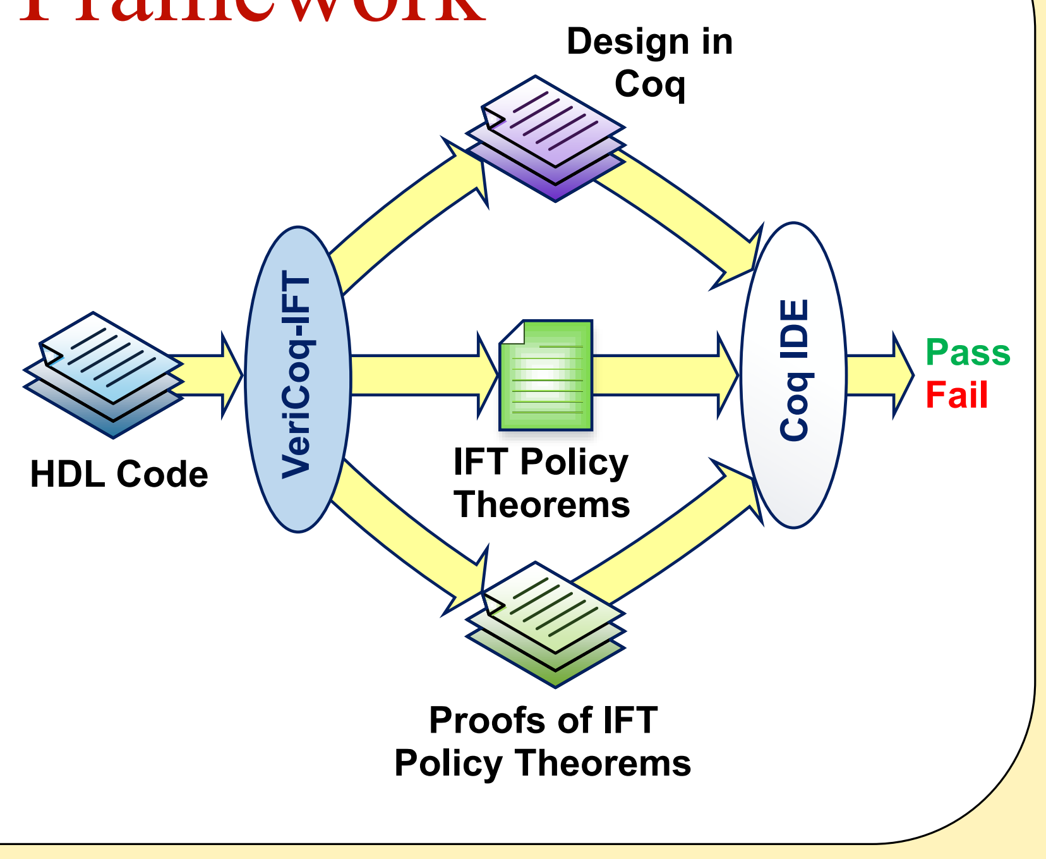
Hardware-Software Boundary Elimination

Bridges the semantic gap by converting the whole computer system into the same formal platform. [2]


Program
Inst 1
Inst 2
Inst N
Security Property in Natural Language
Inst Classification
Theorem for Computer system
Inst type 1
Inst type M
Top Lemma for Inst Type1
Top Lemma for Inst TypeM
Lemma Development for Individual Inst Type 1
Lemma Development for Individual Inst Type M
Lemma Development

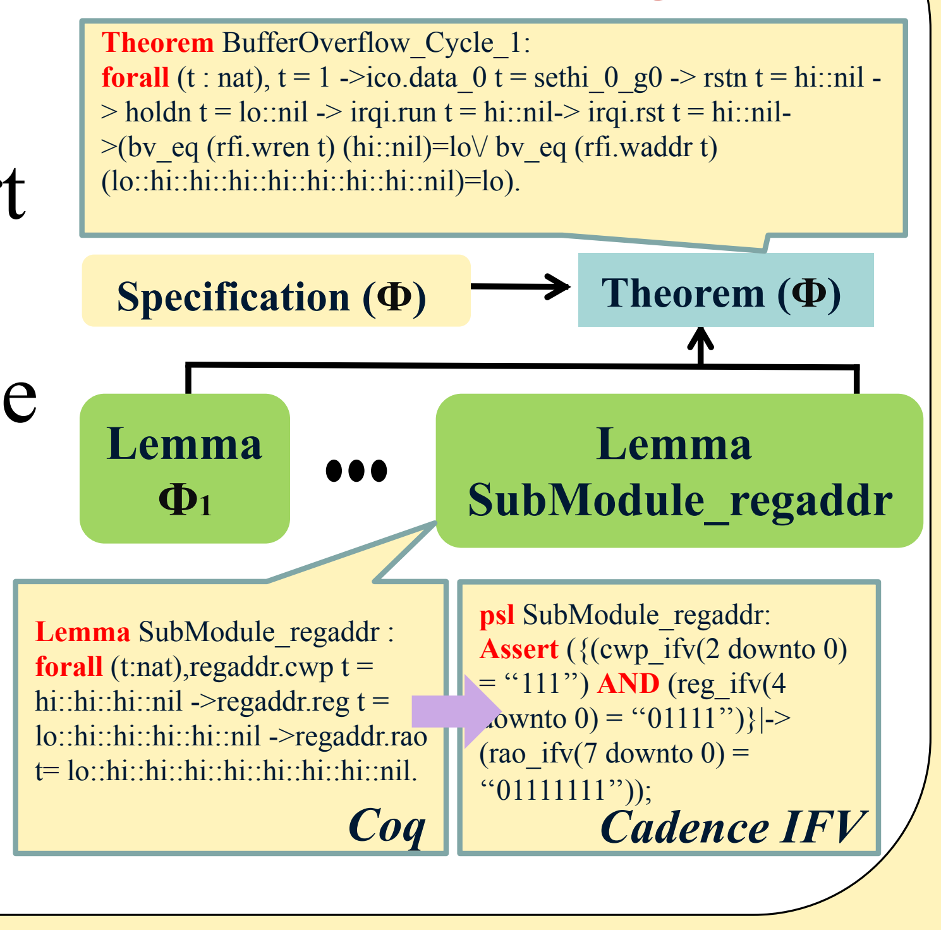
Automation of PCH Framework

VeriCoq-IFT automates all the tasks involve in the PCH framework for enforcing information flow policies on the hardware design. [3]



Theorem Proving and Model Checking Integration

Reduce the amount of effort required for translating the HDL design and proving the security theorem through combining theorem prover (Coq) and model checker (IFV) together. [4]



```
Theorem BufferOverflow_Cycle_1:
forall (t : nat), t = 1 ->ico.data_0 t = sethi_0_g0 -> rstn t = hi::nil -> holdn t = lo::nil -> irqi.run t = hi::nil-> irqi.rst t = hi::nil->(bv_eq (rfi.wren t) (hi::nil)=lo\/ bv_eq (rfi.waddr t) (lo::hi::hi::hi::hi::hi::hi::hi::nil)=lo).
```

```
Lemma SubModule_regaddr :
forall (t:nat),regaddr.cwp t = hi::hi::hi::nil ->regaddr.reg t = lo::hi::hi::hi::hi::nil ->regaddr.rao t= lo::hi::hi::hi::hi::hi::hi::hi::nil.
```

```
psl SubModule_regaddr:
Assert ({(cwp_ifv(2 downto 0) = "111") AND (reg_ifv(4 downto 0) = "01111")}|-> (rao_ifv(7 downto 0) = "01111111"));
```

REFERENCES

[1] X. Guo, R. Dutta, Y. Jin, F. Farahmandi, and P. Mishra, "Pre-silicon security verification and validation: A formal perspective," in Design Automation Conference (DAC), 52nd ACM/EDAC/IEEE, June 2015, pp.1–6.

[2] X. Guo, R. Dutta, and Y. Jin, "Eliminating the Hardware-Software Boundary: A Proof-Carrying Approach for Trust Evaluation on Computer Systems," IEEE Transactions on Information Forensics and Security (TIFS) (to appear).

[3] M.-M. Bidmeshki, and Y. Makris, "Toward automatic proof generation for information flow policies in third-party hardware IP," in IEEE Symposium on Hardware-Oriented Security and Trust (HOST), 2015, pp. 163–168.

[4] X. Guo, R. Dutta, P. Mishra, and Y. Jin, "Scalable SoC Trust Verification using Integrated Theorem Proving and Model Checking," IEEE Symposium on Hardware Oriented Security and Trust (HOST), 2016, pp. 124-129.

Interested in meeting the PIs? Attach post-it note below!

National Science Foundation
WHERE DISCOVERIES BEGIN

NSF Secure and Trustworthy Cyberspace 3rd Biennial Principal Investigator Meeting
Jan. 9th - 11th 2017
Arlington, VA

UCF
UT DALLAS