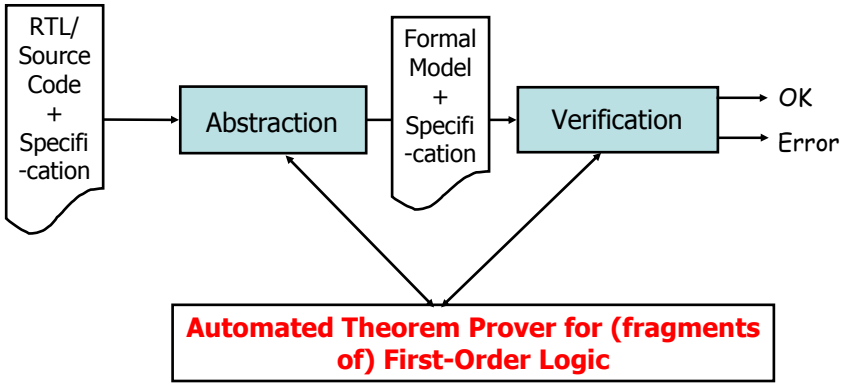
EECS 219C: Computer-Aided Verification

Automated Theorem Proving for First-Order Logic

Sanjit A. Seshia
EECS, UC Berkeley

Automated Theorem Proving in Formal Verification

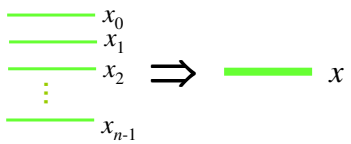


Applications in verification of: Microprocessor designs, cache coherence protocols, software verification, …

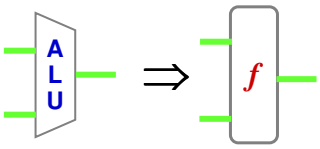
First-Order Logic (Predicate Calculus)

- Propositional/Boolean logic is zeroth-order
- In general, propositions can involve non-propositional variables/terms
  - Propositions: x = y, f(x) > z, 2x + 3y = 5, p(x), etc.
  - Have x, y, f(x), z, 2x, 3y, 5, etc. as terms
  - The propositions themselves are often called "predicates" or "atomic formulas"
  - Also have "functions" like "f"
- Quantifiers occur only over primitive types (Boolean or term/"individual" variables)
  - Not over function or predicate types
- First order formula is a Boolean combination of propositions

Data and Function Abstraction in Hardware



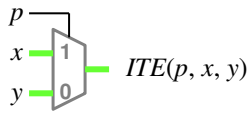
Bit-vectors to (unbounded) Integers



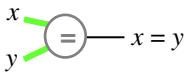
Functional units to Uninterpreted Functions

$$a = x \wedge b = y \Rightarrow f(a,b) = f(x,y)$$

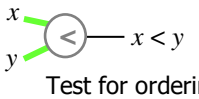
Common Operations



If-then-else



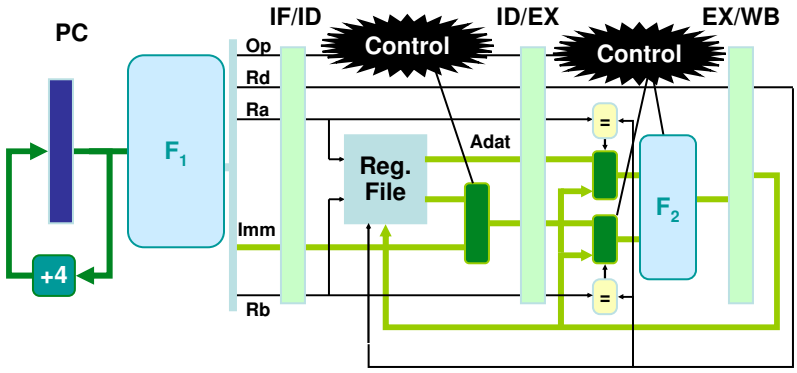
Test for equality



Test for ordering

Counters

Abstraction Via Uninterpreted Functions



- For any Block that Transforms or Evaluates Data:
  - Replace with generic, unspecified function
  - Also view instruction memory as function

Focus of this Lecture and the Next

- SAT-based automated theorem proving for fragments of first-order logic
- This lecture: The “eager encoding” approach
  - as in the UCLID decision procedure
- Next lecture: The “lazy” encoding approach