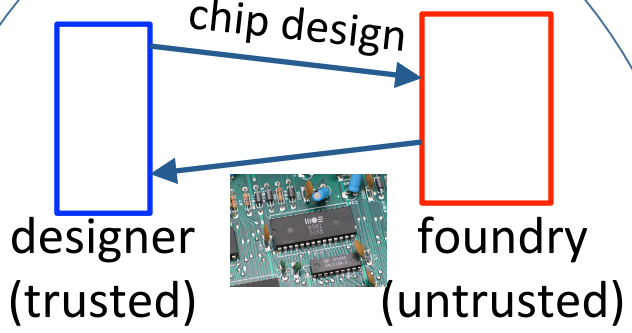
Scaling proof-based verifiable computation

Challenge:

- How can one machine *check* that another machine computed correctly? (applies to untrusted hardware, cloud computing, etc.)

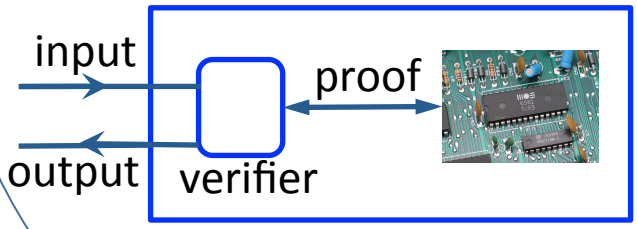
Setting:



Solution:

- Apply and implement theory of probabilistic proofs; also known as *verifiable computation*
- Refine theory and adjust protocols to fit *hardware*

Response:



Scientific Impact:

- Milestone in verifiable computation (first hardware design)
- Broadens verifiable computation to more classes of computations and larger computations

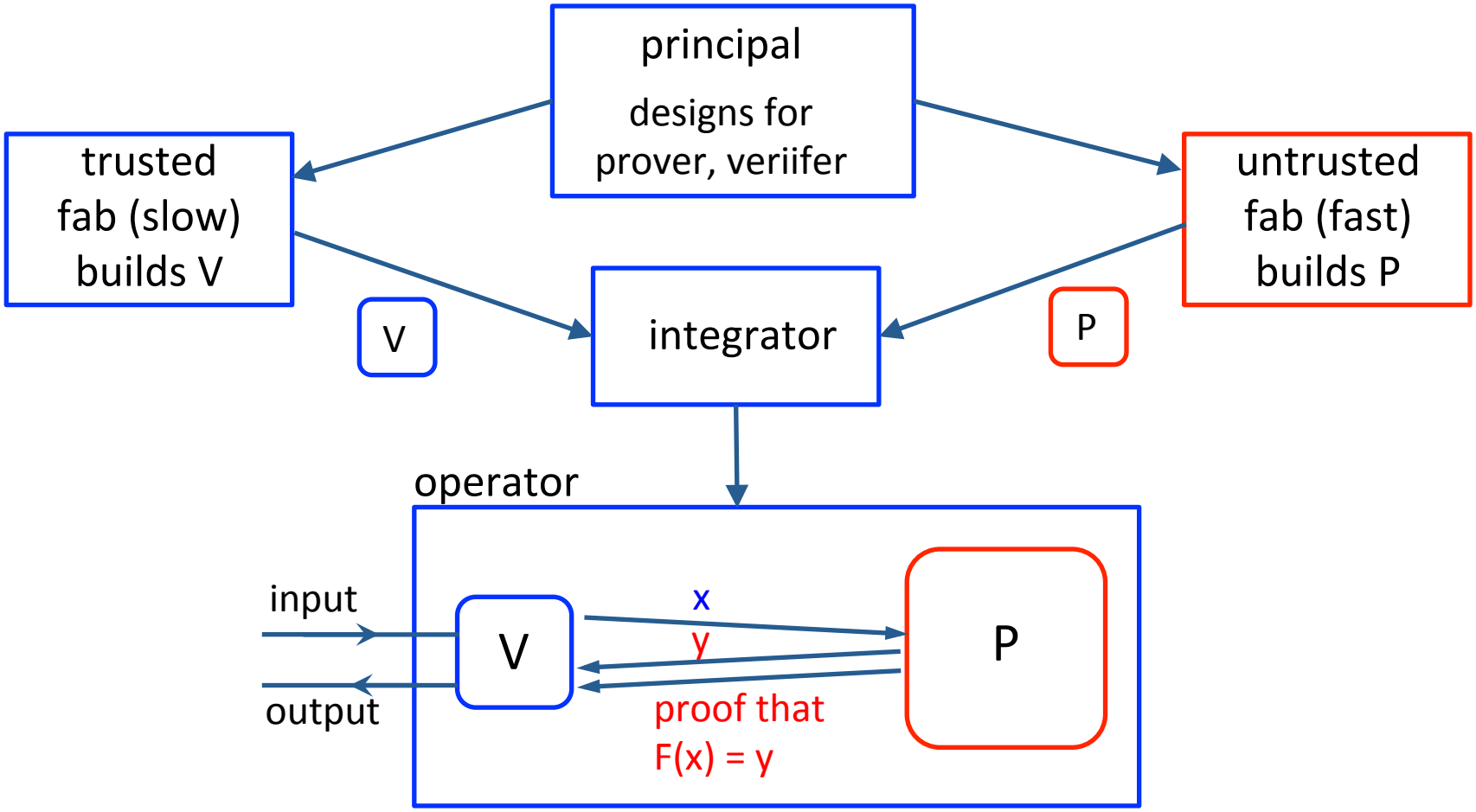
Broader Impact:

- Potential alternative to trusted foundry
- Undergraduate research involvement
- Software released
- Detailed tutorials in verifiable computation

Project number: CNS-1514422, NYU and UT Austin

PIs: Michael Walfish (mwalfish@cs.nyu.edu), Thomas Wies (wies@cs.nyu.edu), Andrew J. Blumberg (blumberg@math.utexas.edu)

Note: This slide depicts one thrust of a broader project.


principal
designs for
prover, veriifer
trusted
fab (slow)
builds V
untrusted
fab (fast)
builds P
V
integrator
P
operator
input
output
V
x
y
proof that
F(x) = y
P