CrypTech

Building a More Assured Hardware Security Module

Randy Bush <randy@psg.com>

HSMs Are Used For

- Principally, Lock-box for Private Keys for
  - DNSsec
  - RPKI
  - PGP
  - Tor
  - Corporate Authentication
- Also,
  - Encryption / Decryption
  - VPNs
  - Source of Randomness

The Need

- Every week a new horror about Crypto/Privacy
- *der Spiegel's* revelation of the "SpyMall Catalogue"
- Compromises of and trojans in most network devices, servers, firewalls, ...
- **We are relying on HSMs designed and made by 42-eyes government contractors**
- Many people are not comfortable with this

Origins

- This effort was started at the suggestion of Russ Housley, Jari Arkko, and Stephen Farrell of the IETF, to **meet the assurance needs of supporting IETF protocols in an open and transparent manner**.

- **But this is NOT an IETF, ISOC, ... project**, though both contribute. As the saying goes, "We work for the Internet."

Goals

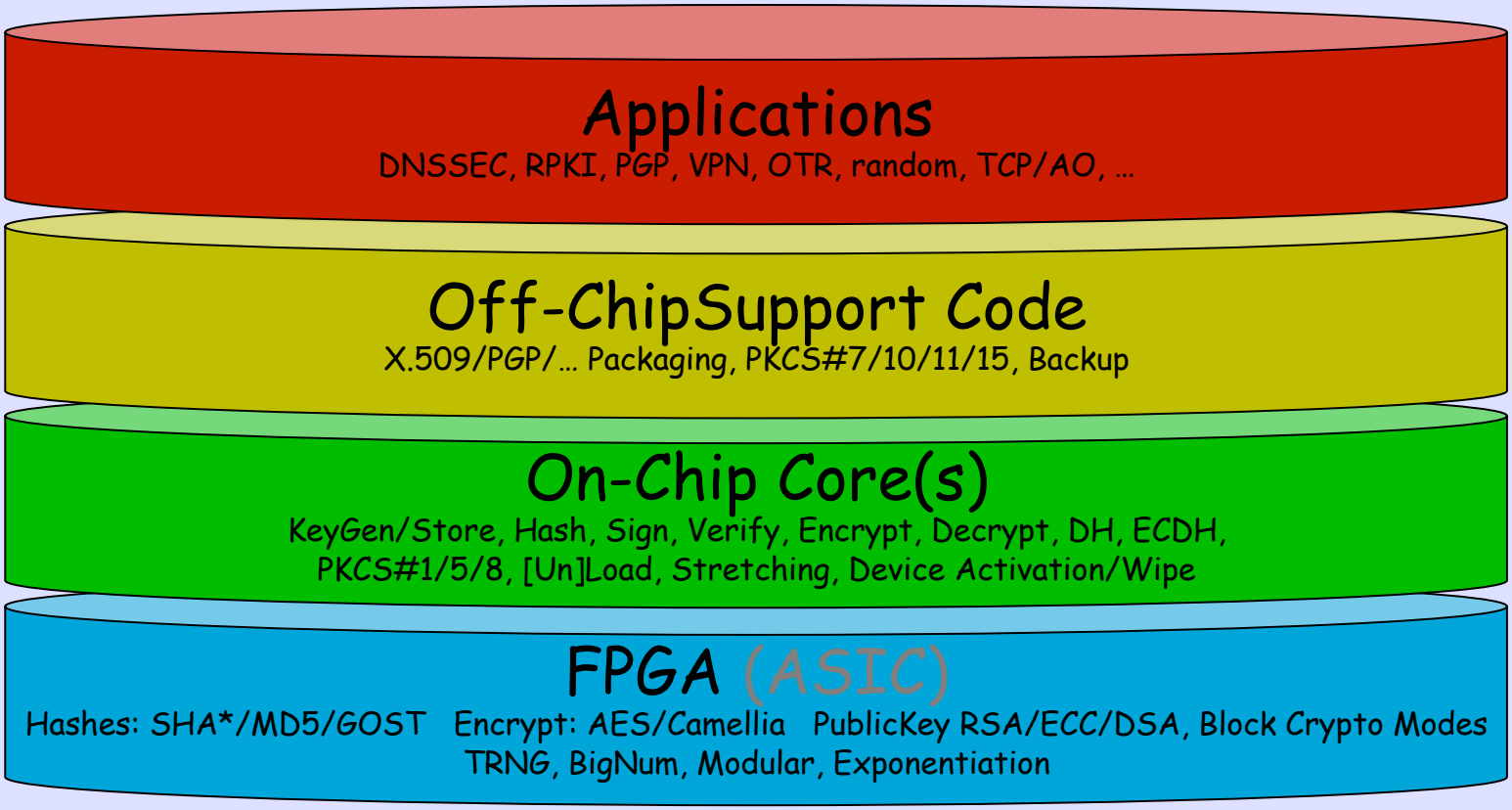
- **An open-source reference design for HSMs**
- Scalable, first cut in an FPGA and CPU, later allow higher speed options
- Composable, e.g. "Give me a key store and signer suitable for DNSsec"
- Reasonable assurance by being open, diverse design team, and an increasingly assured tool-chain

Funding (so far) From

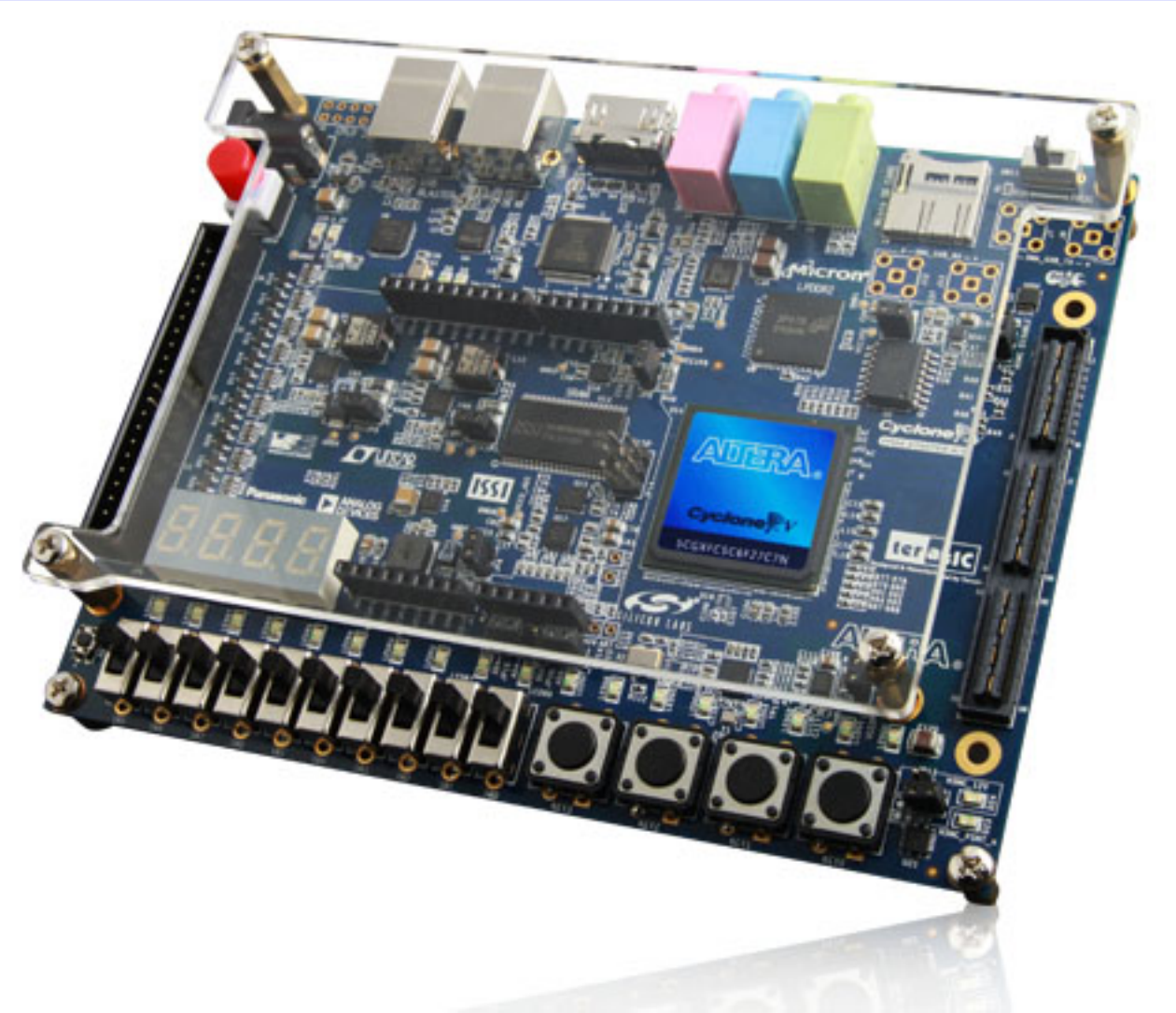
A Few Private Donations

Your Logo Goes Here

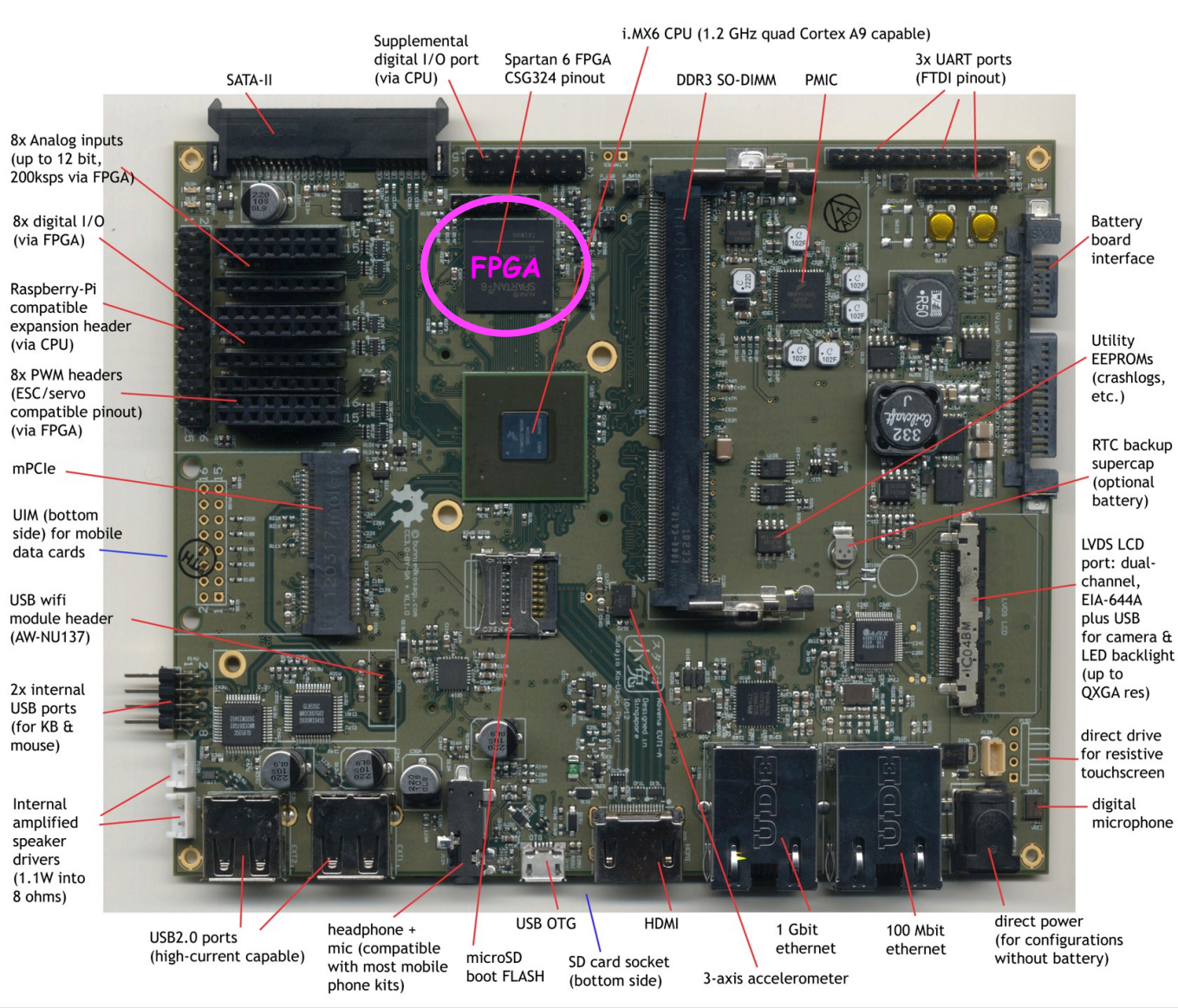
Layer Cake Model



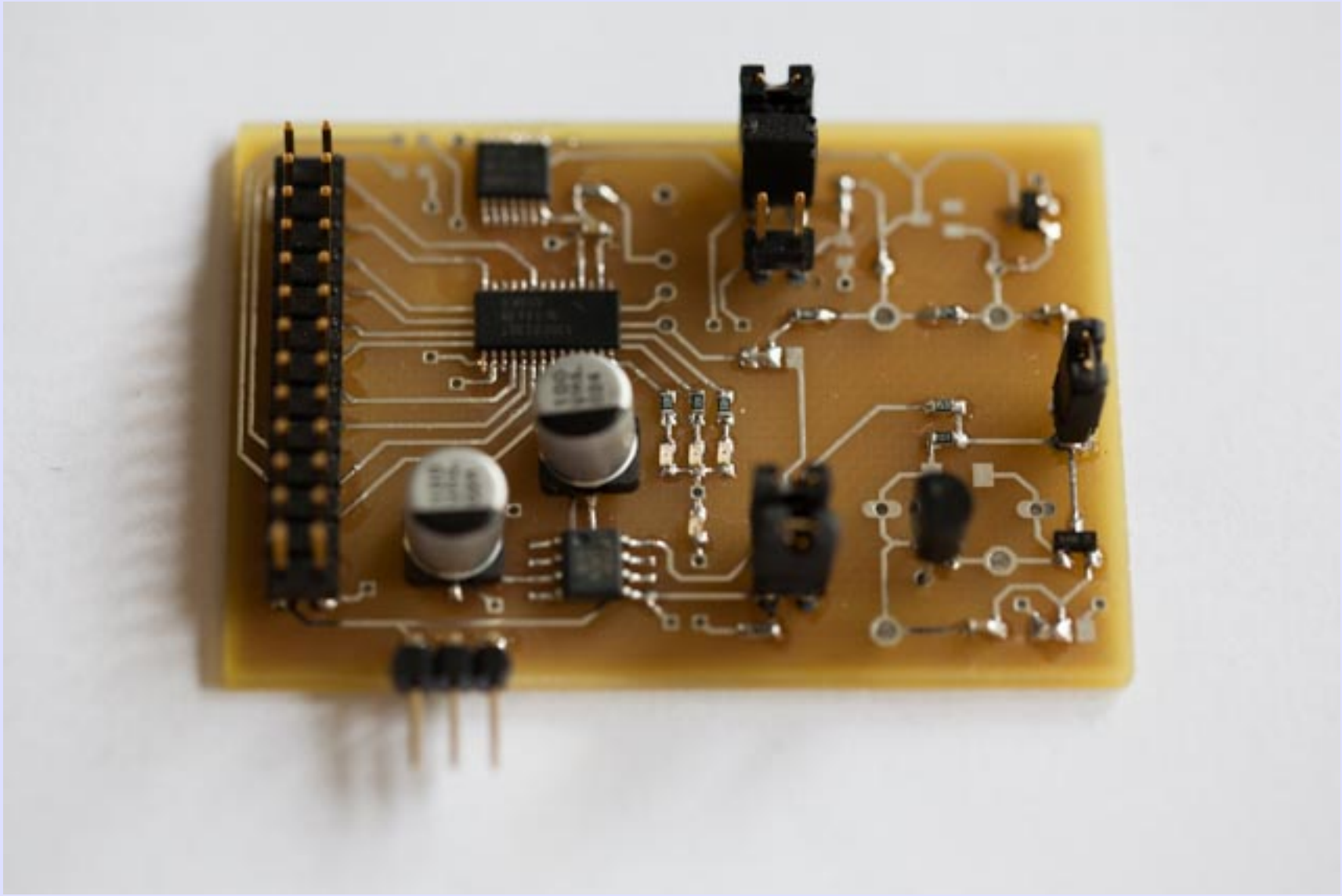
A Prototyping Board



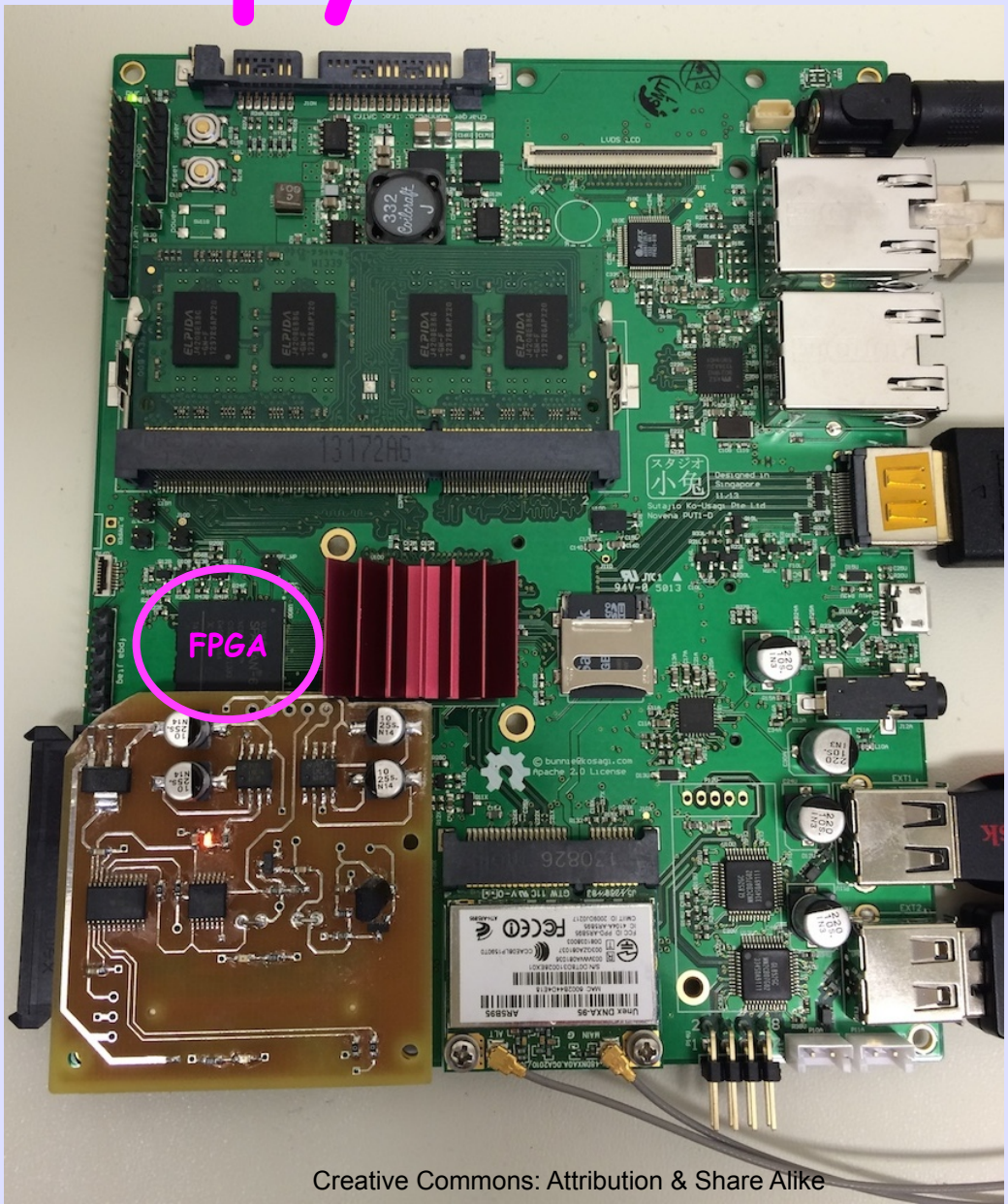
Novena Spartan 'Laptop'



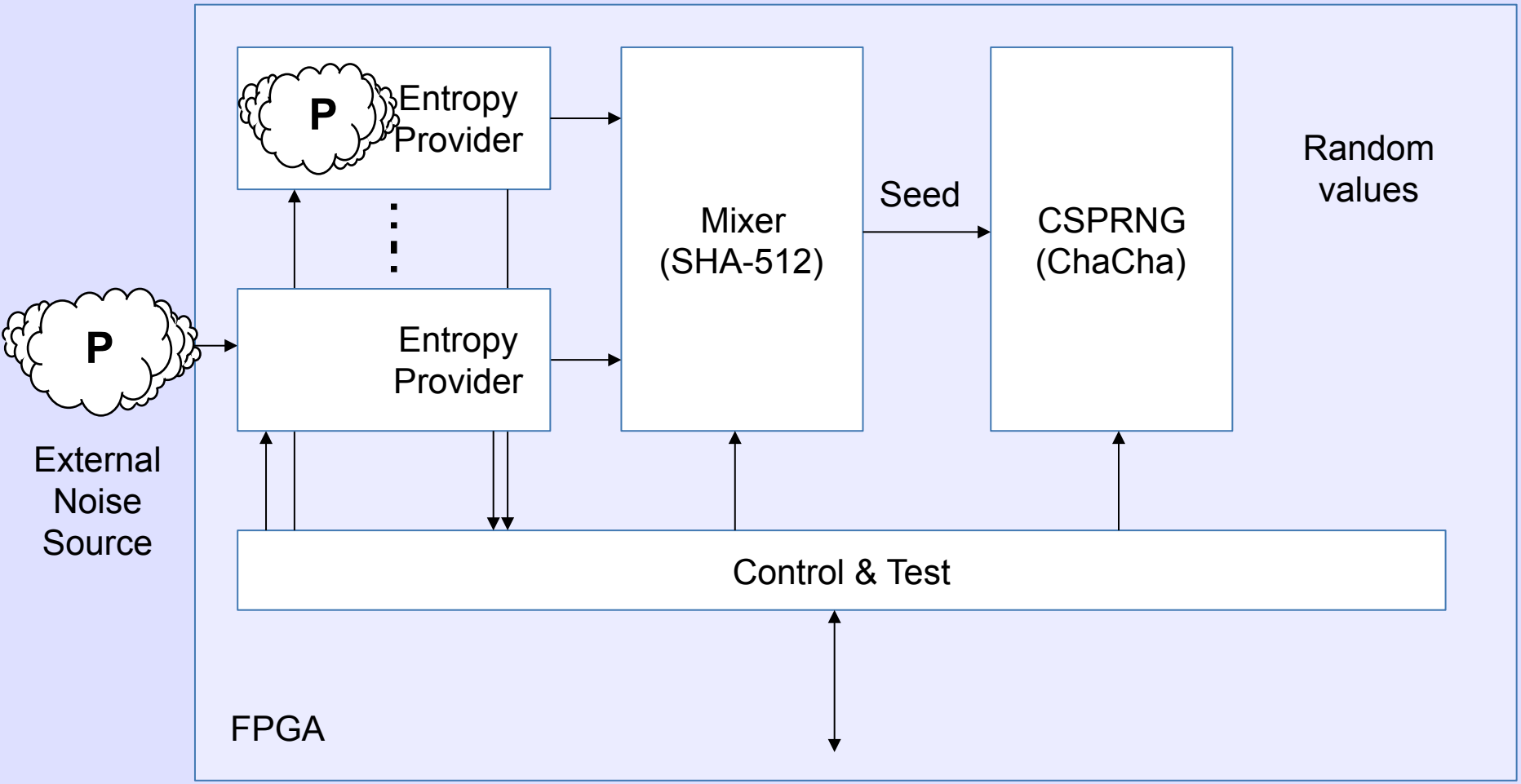
Entropy with Pi Pin-Out



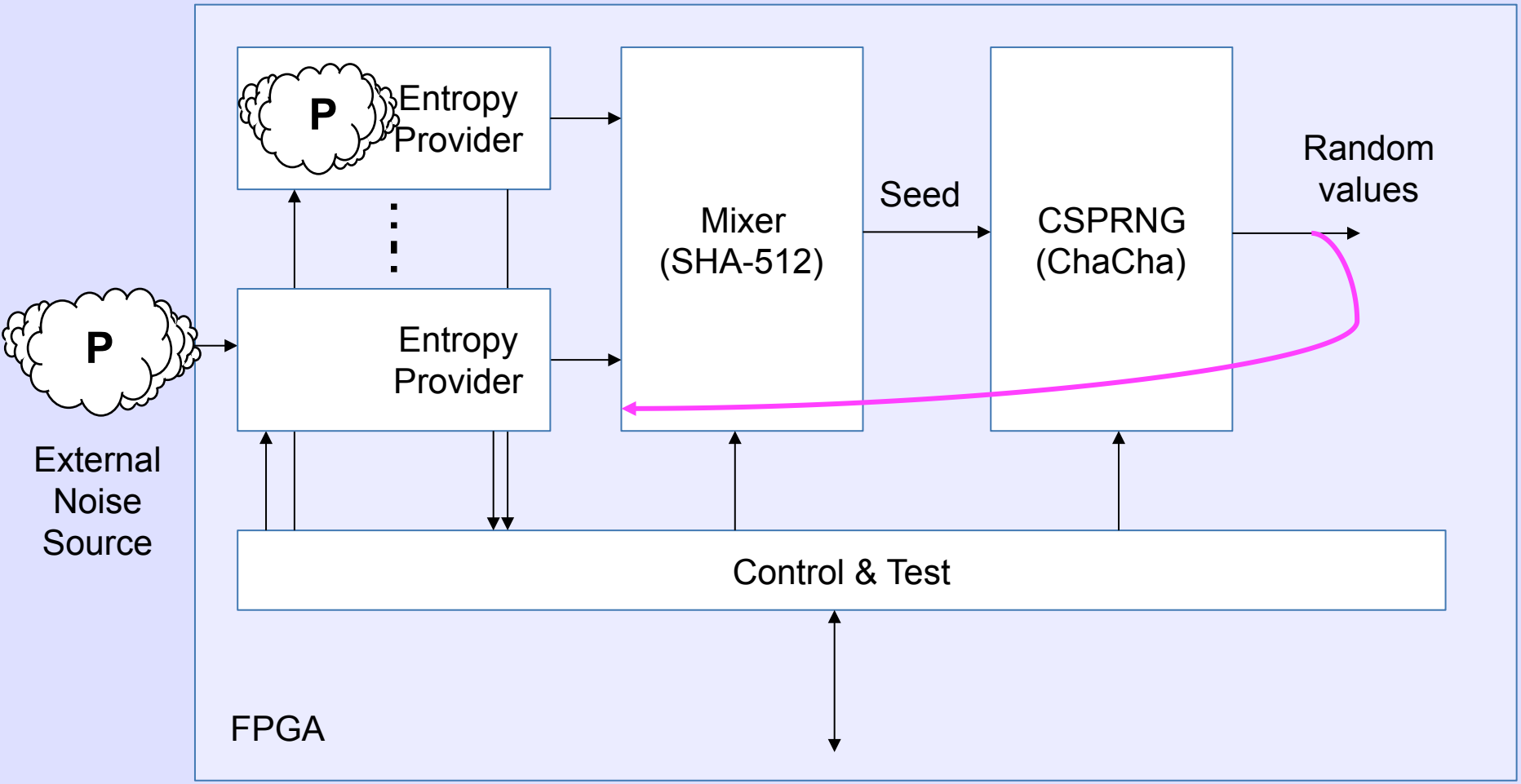
Entropy on Novena



The TRNG Architecture



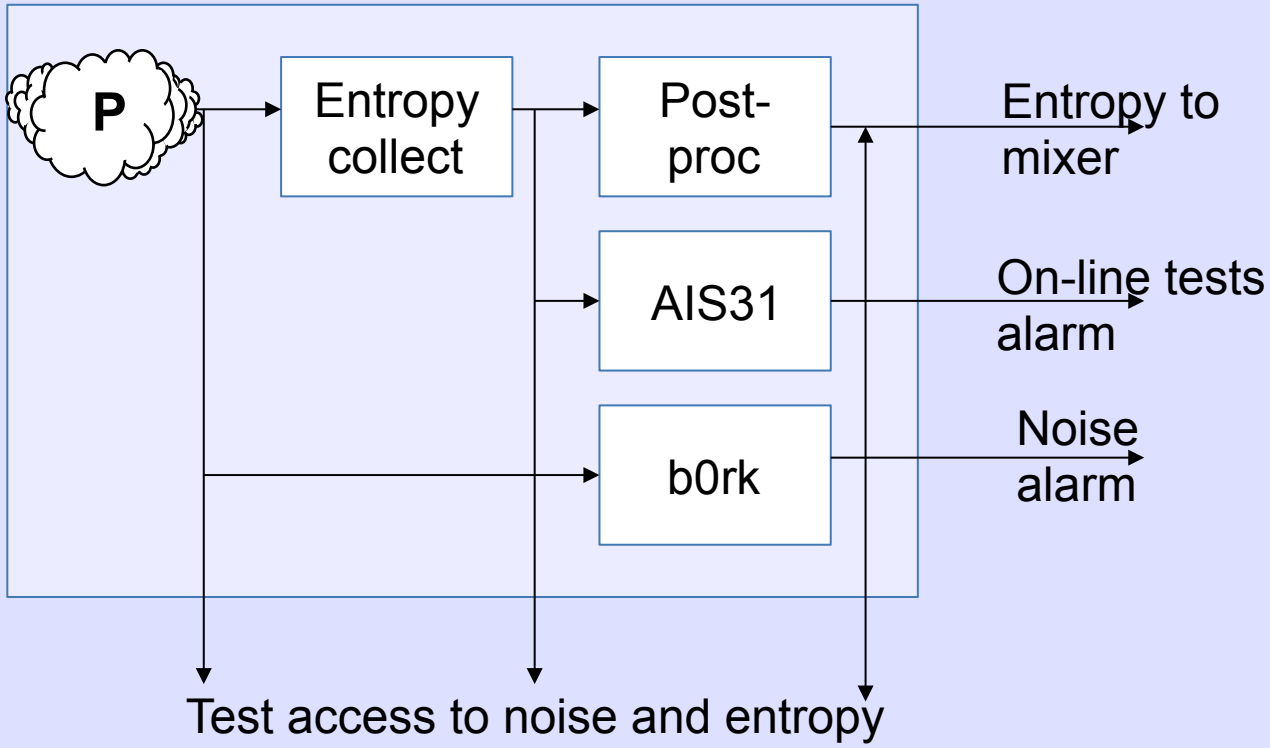
Or Maybe



Test and Observability

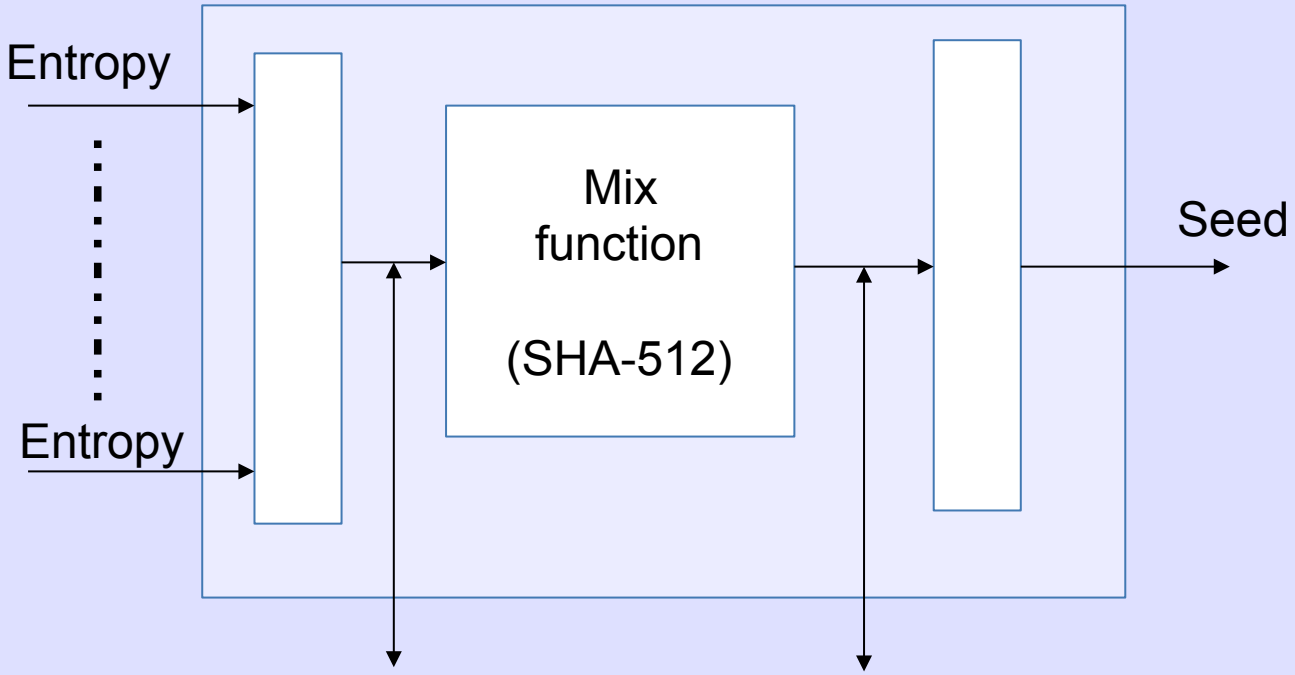
- Two modes
  - Production Mode (PM) and Test Mode (TM)
- Observability of entropy sources in PM
- Continuous on-line testing in PM
- Injection in stages and complete chain in TM
- Generation of a small number of values in TM
- Allows test of all digital functionality including continuous tests.
- Full restart when going between TM and PM

Observability & Test of Entropy Sources



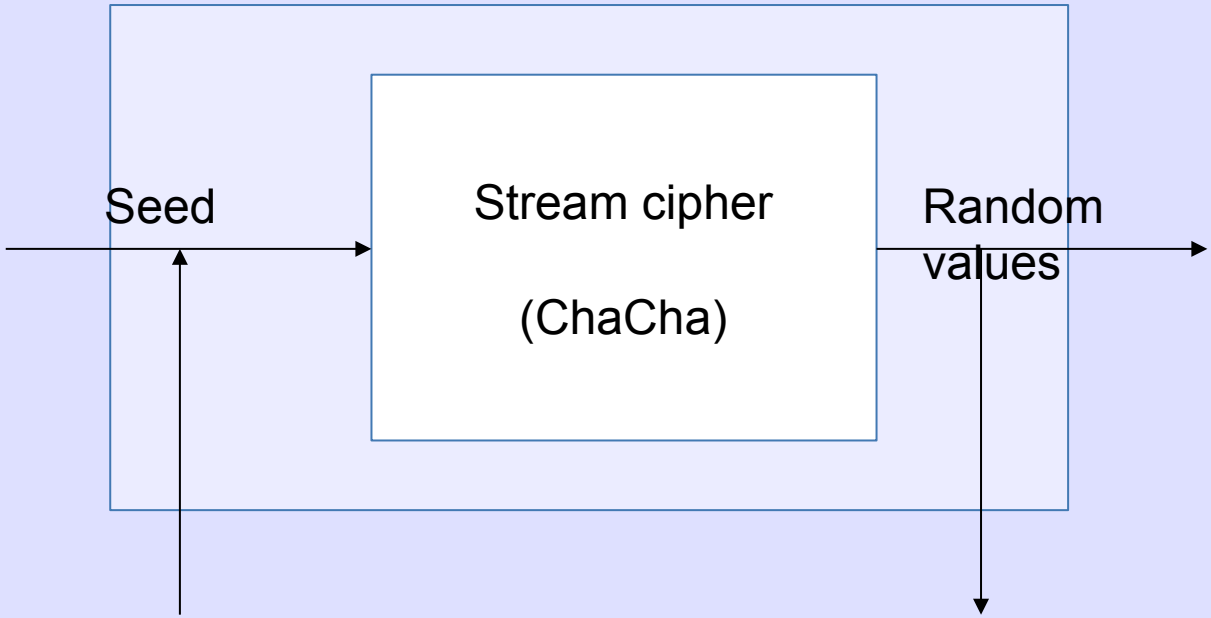
- Extract for off-line comprehensive testing
- Inject for functional testing in test mode

Observability & Test of Mixer



Inject for functional testing in test mode

Observability & Test of CSPRNG



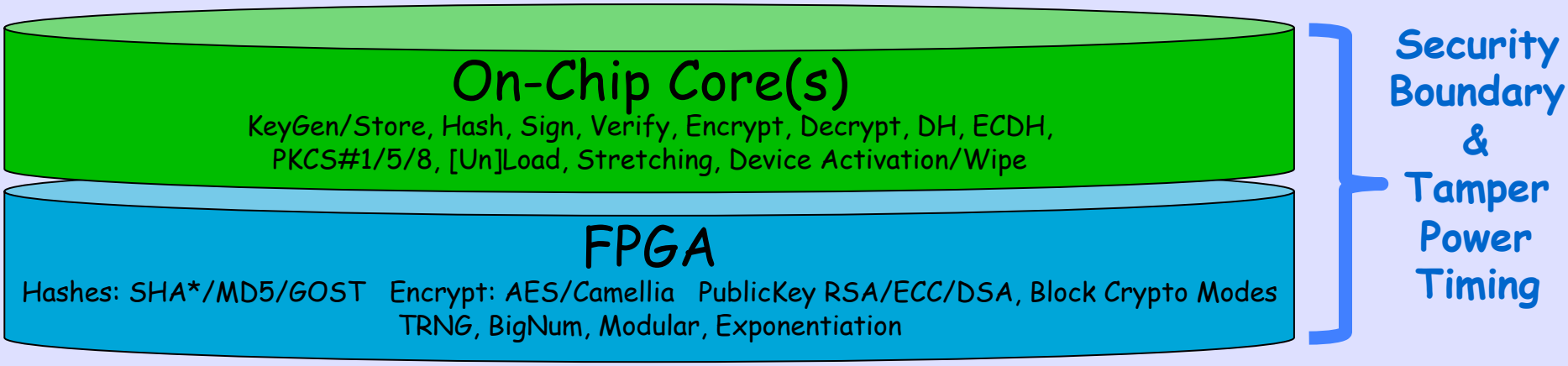
Inject for functional testing in test mode

Side Channel & Tampering

- Exponentiation circuit **timing leaks** are exploitable remotely
- **Power leakage** is exploitable locally
- **Physical attack** detection critical
- Wipe key store if tampering detected
- Side-Channel attacks are the subject of entire conferences

Potting Boundary

- The FPGA/ASIC and accompanying Core(s) (ARM, whatever) are within the physically protected boundary of the chip carrier potting.
- On-board battery/capacitor to buy the time to wipe all data if unplugged from power
- We worry about tampering, what if the chip is opened and attacked? So the potting includes tampering sensors and code to wipe all keys if tampering is detected.



Some Phases

- First Year: Tool-chain, Basic Design, not all cyphers, not all protocols, prototype implementations on FPGAs and boards
- Second Year: Better Tool-chain, all needed cyphers, hashes, crypting, … and integration with some apps, DNSsec, RPKI, TLS, PGP, Tor
- Third Year: Solid packaging, ability to compose designs for use cases, etc.