Can Truly Dependable Systems Be Affordable?

Gernot Heiser
NICTA and UNSW

Australian Government
Department of Broadband, Communications and the Digital Economy
Australian Research Council

NICTA Funding and Supporting Members and Partners

Present Systems are *NOT* Trustworthy!

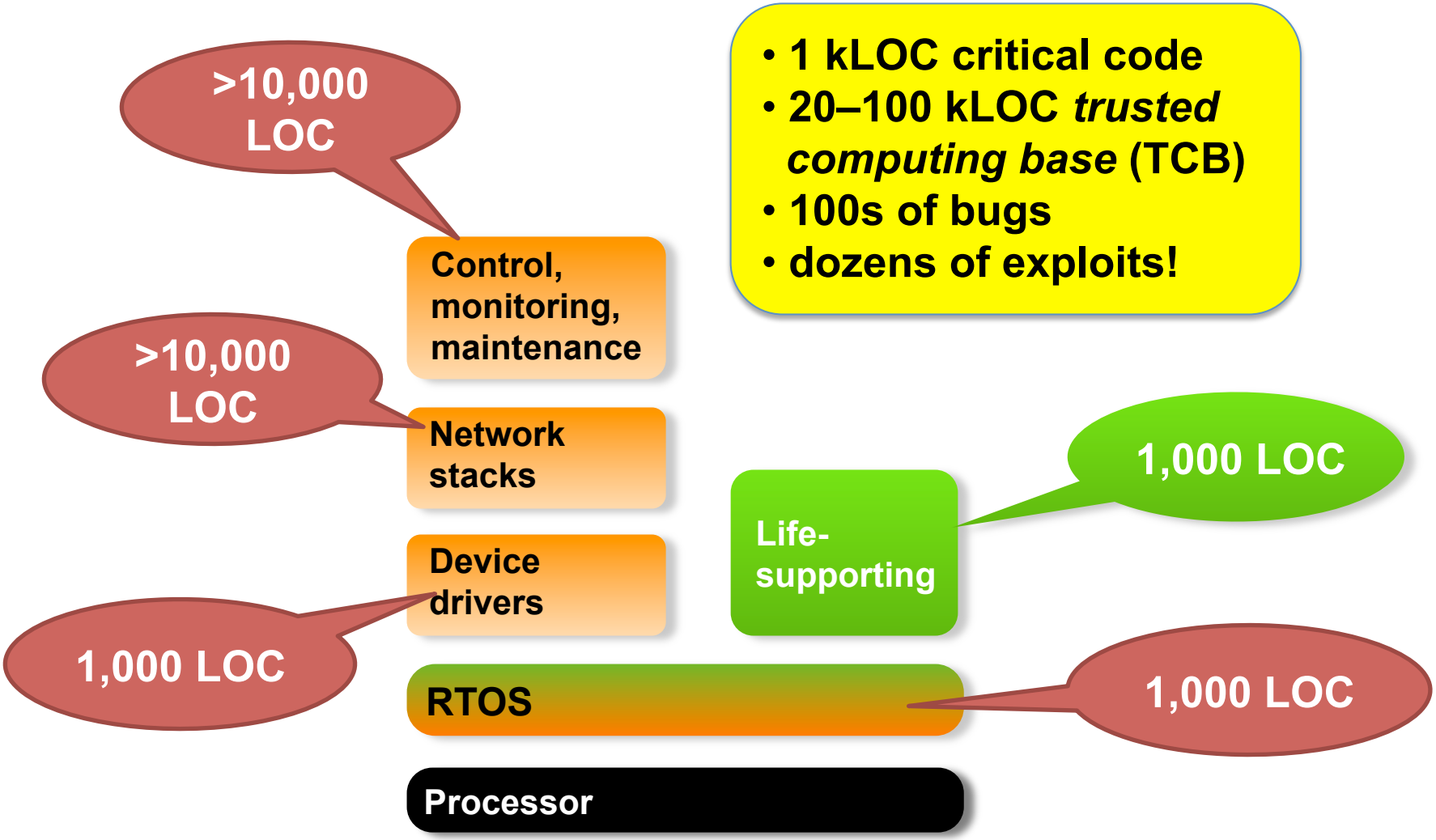
NICTA

Yet they are expensive:

- $1,000 per line of code for "high-assurance" software!

Fundamental issue: large stacks, need isolation

E.g. medical implant

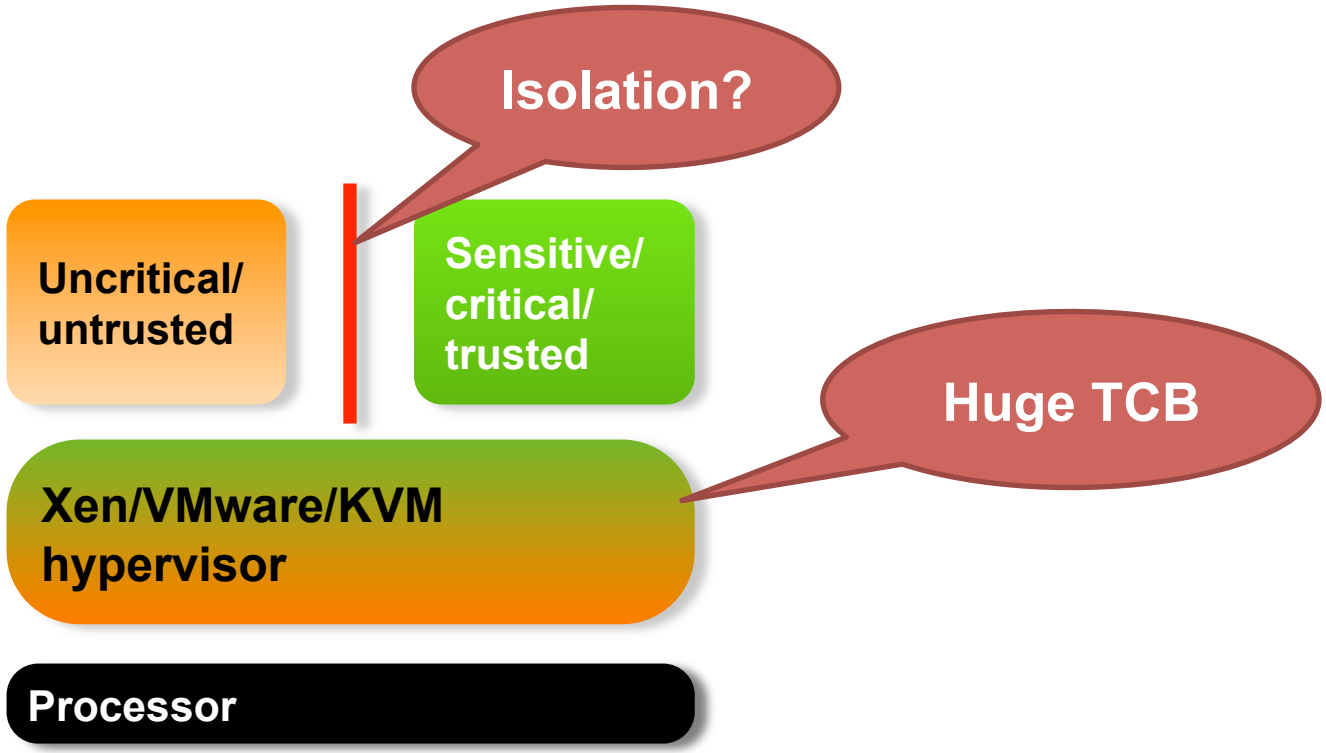


High Assurance *Bad* Practice

NICTA

- TCB of millions of LOC
- Expect 1000s of bugs
- Expect 100s of vulnerabilities

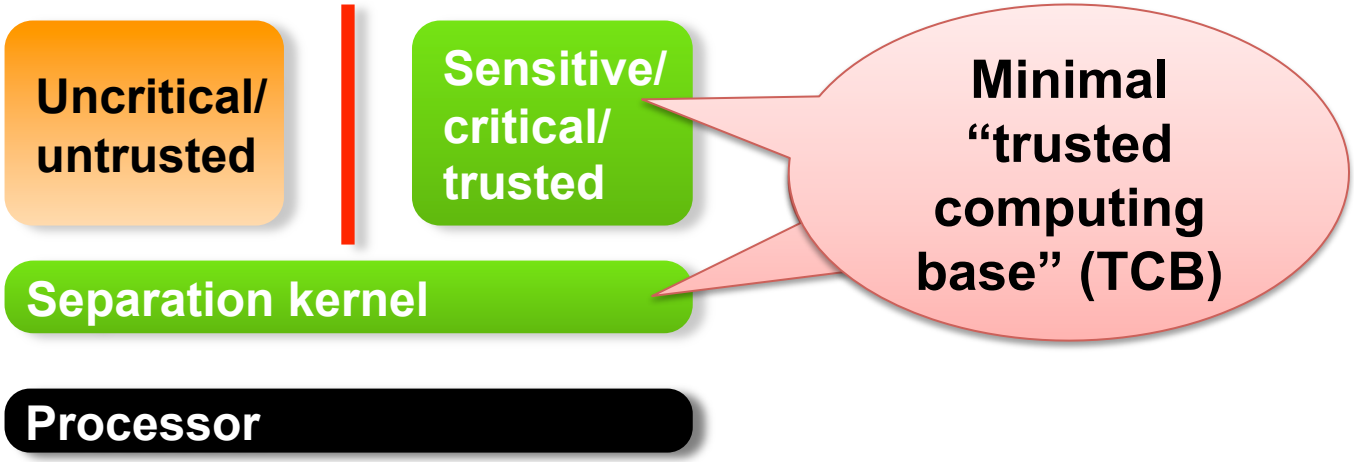
Hacker's delight!



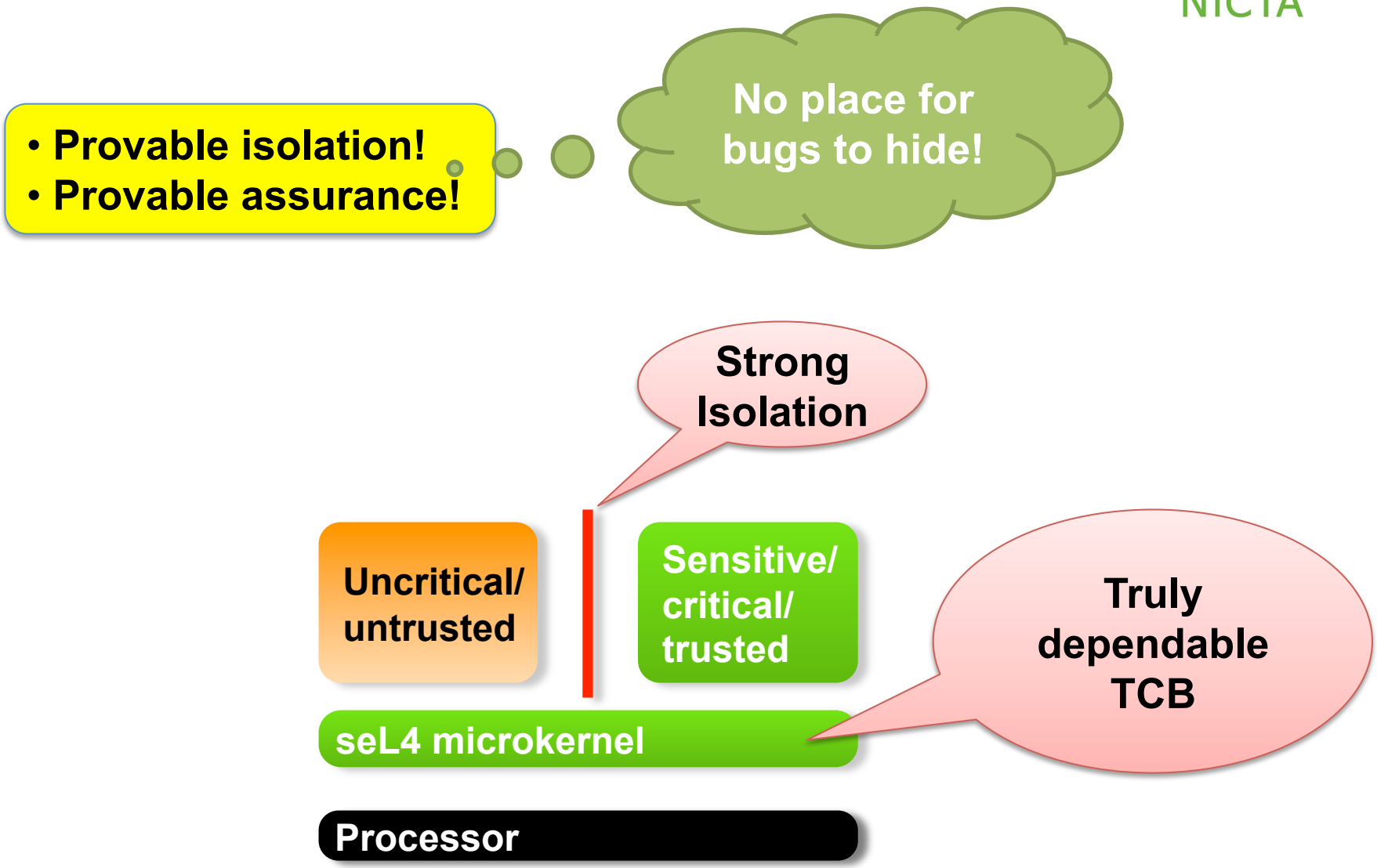
High Assurance Best Practice

- Isolate
- Minimise the TCB
- Assure TCB by
  - testing
  - code inspection
  - bug-finding tools

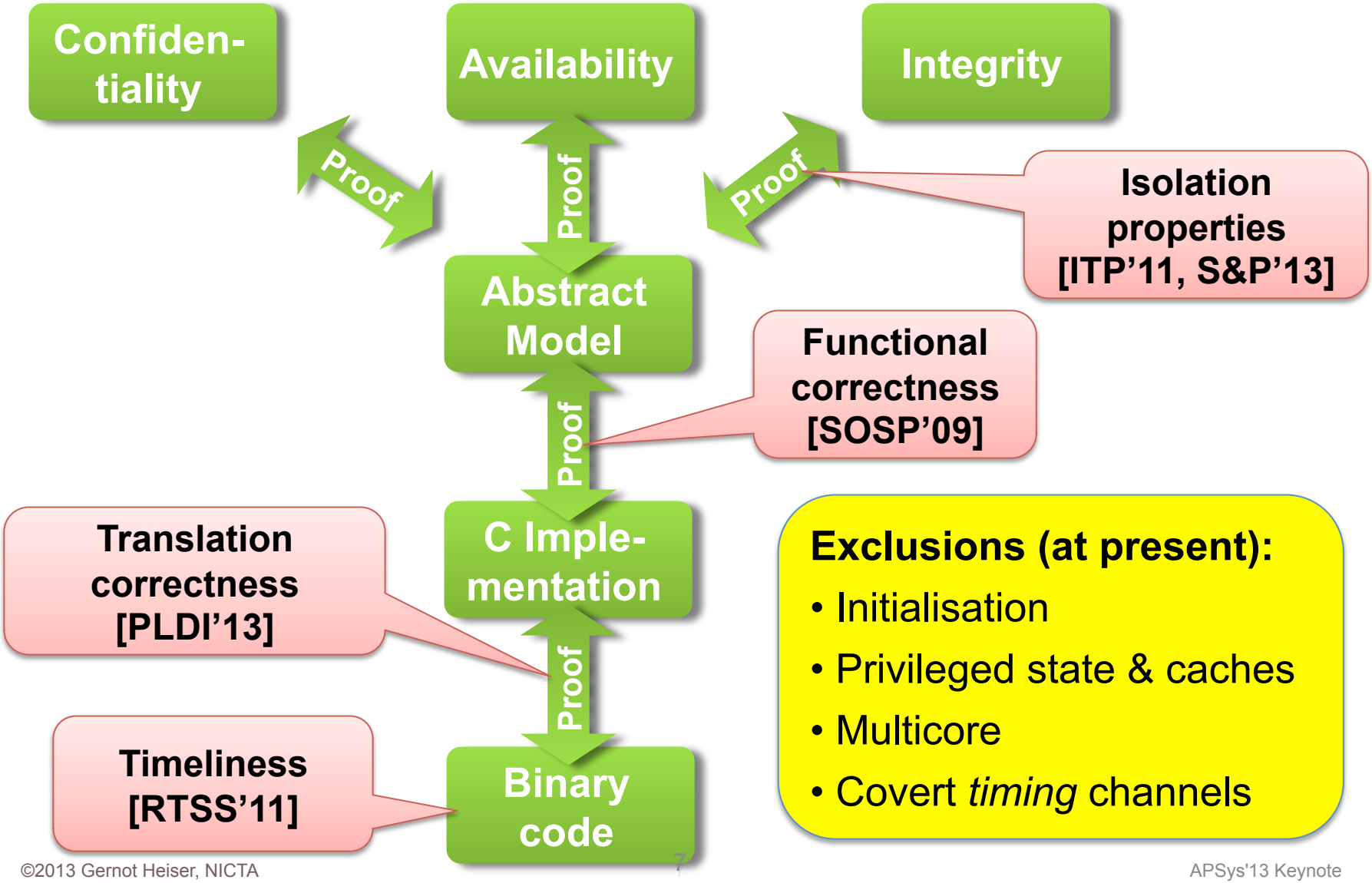
Always incomplete!



State of the Art: NICTA's seL4 Microkernel



NICTA's seL4: Mathematical *Proof* of Isolation



NICTA's seL4 Microkernel: Unique Assurance

First and only operating-system with functional-correctness proof: operation is always according to specification

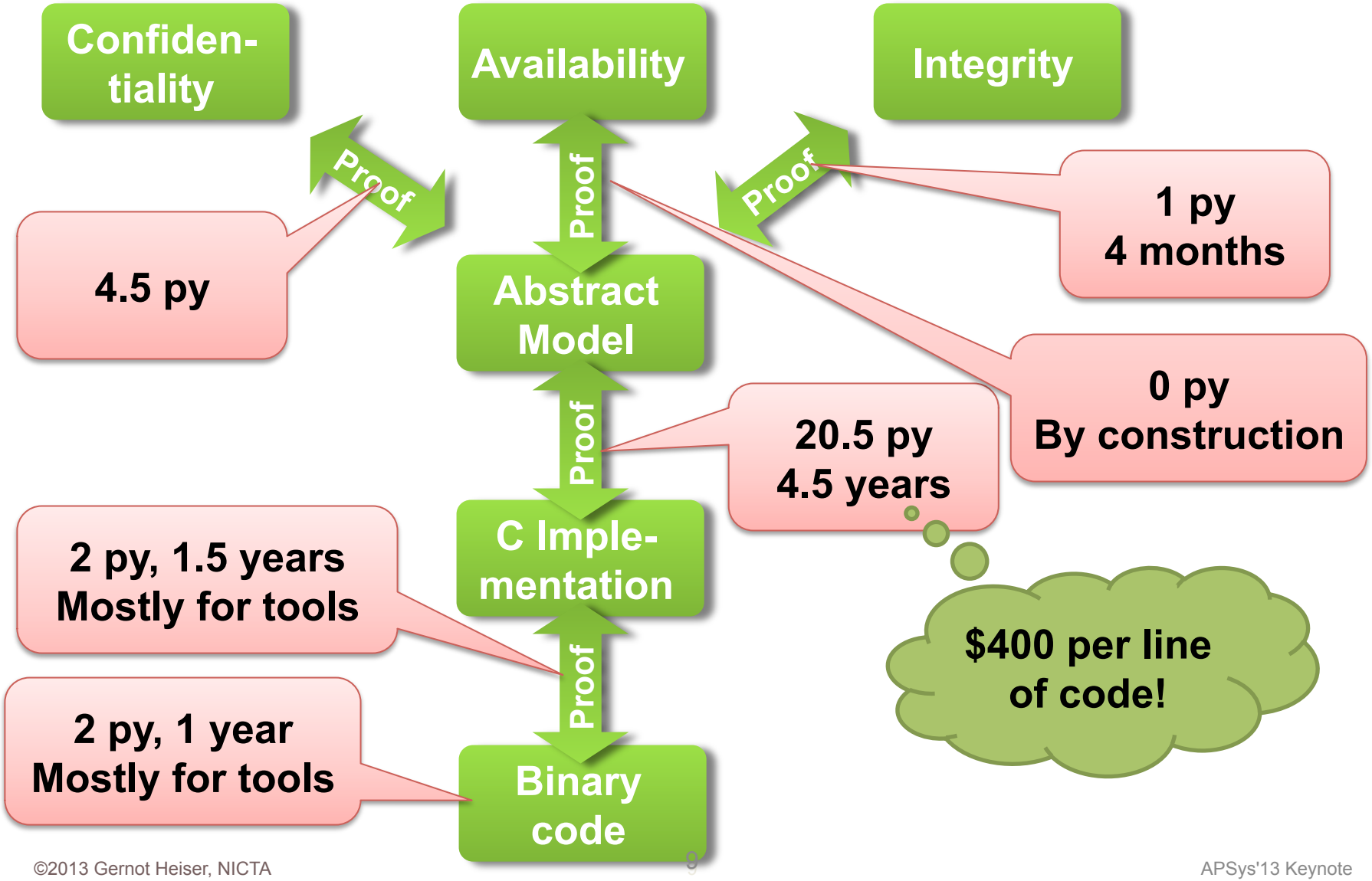
Predecessor deployed on 2 billion devices

First and only operating-system with *proof* of integrity and confidentiality enforcement – at the level of binary code!

World's fastest microkernel on ARM architecture

First and only protected-mode operating-system with complete and sound timing analysis

seL4: Cost of Assurance



Cost of Assurance

Industry Best Practice:

- "High assurance": $1,000/LOC, no guarantees, *unoptimised*
- Low assurance: $100–200/LOC, 1–5 faults/kLOC, *optimised*

State of the Art – seL4:

  - $400/LOC, 0 faults/kLOC
- Estimate repeat would cost half
  - that's about the development cost of the predecessor Pistachio!
- Aggressive optimisation [APSys'12]
  - much faster than traditional high-assurance kernels
  - as fast as best-performing low-assurance kernels

What Have We Learnt?

Formal verification *probably* didn't produce a more *secure* kernel

- In reality, traditional separation kernels are *probably* secure

But:

- We now have certainty
- We did it *probably* at less cost

Real achievement:

- Cost-competitive at a scale where traditional approaches still work
- Foundation for scaling beyond: **2 × cheaper, 10 × bigger!**

How?

- Combine theorem proving with
  - synthesis
  - domain–specific languages (DSLs)

Next Step: Full System Assurance

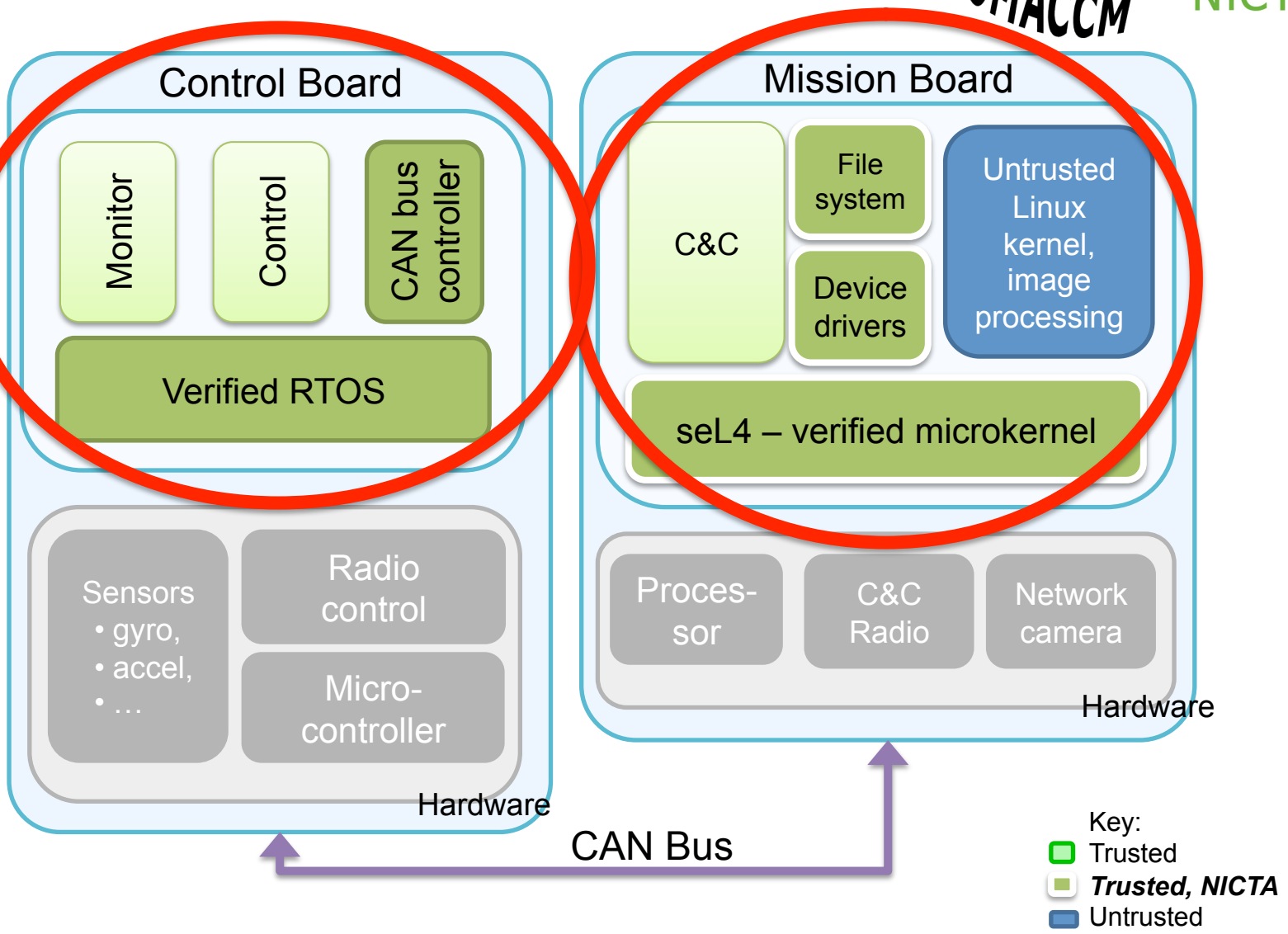
DARPA HACMS Program:

- Provable vehicle safety
- "Red Team" must not be able to divert vehicle

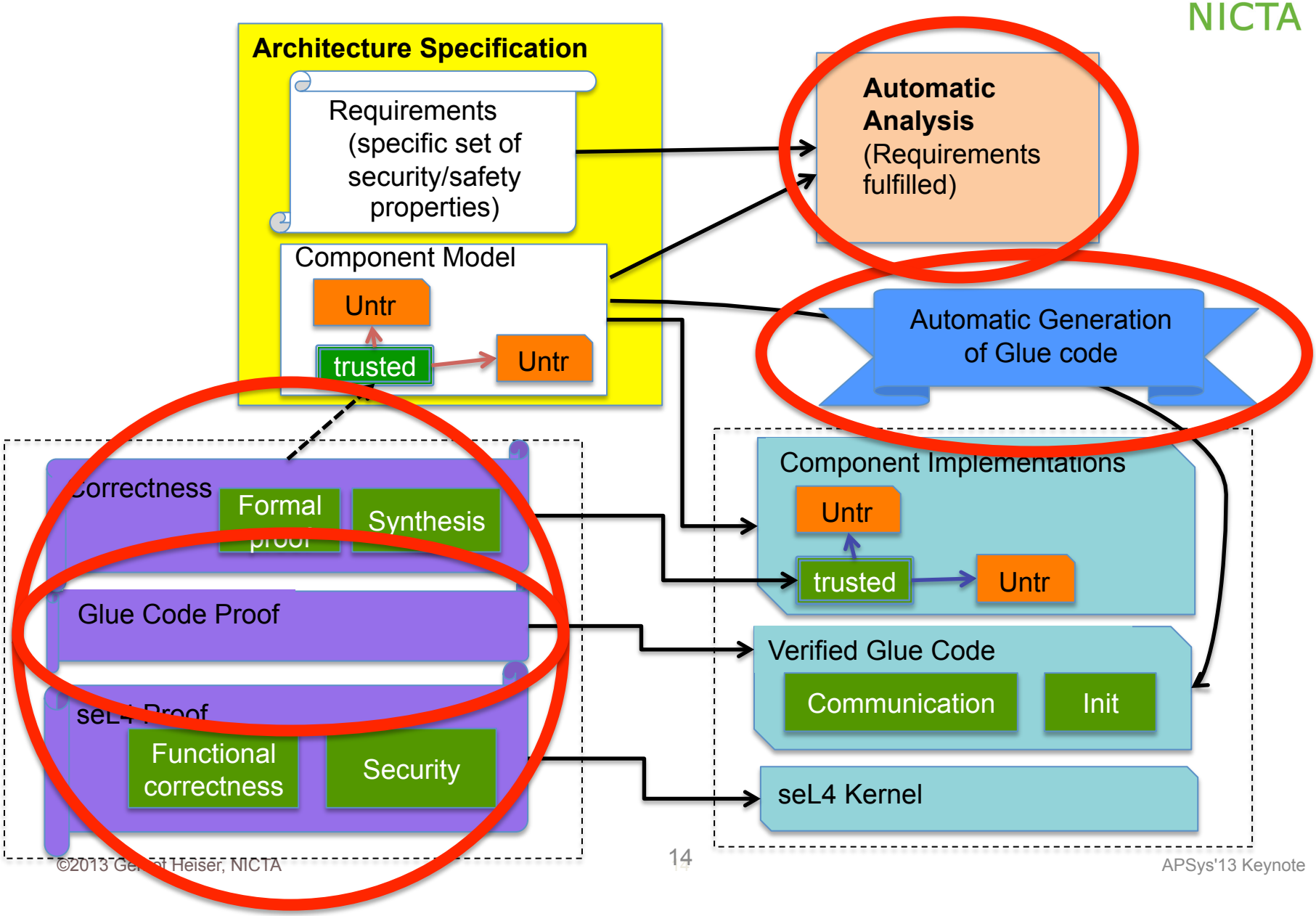
SMACCMcopter
Research Vehicle

Boeing Unmanned
Little Bird (AH-6)
Deployment Vehicle

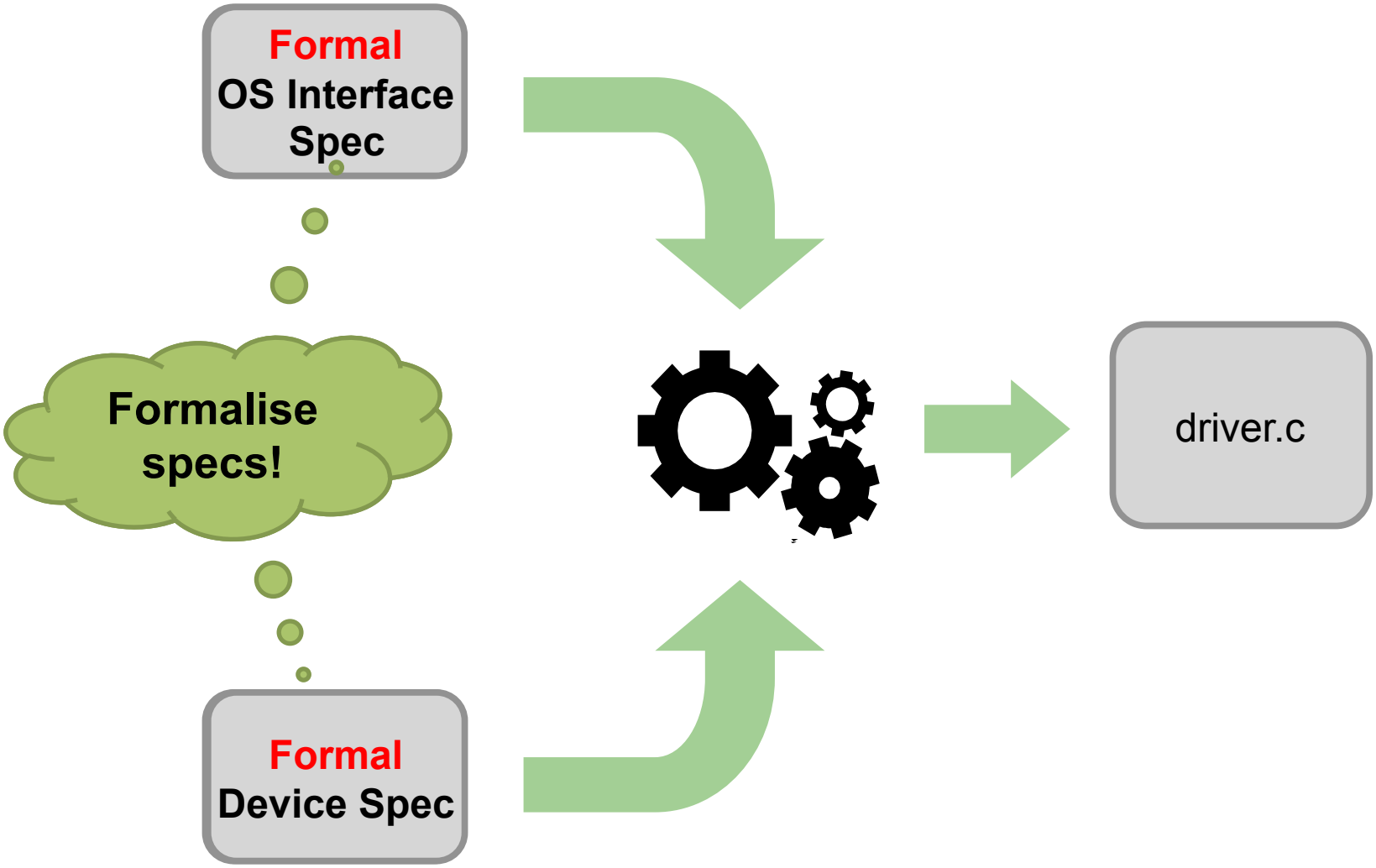
System Structure



Architecting System-Level Security/Safety



Synthesis: Device Drivers [SOSP'09]



Actually works! (On Linux & seL4)

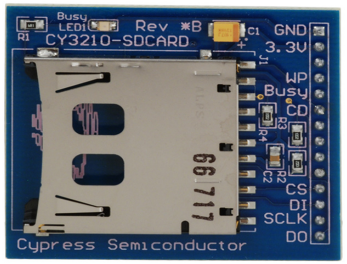
IDE disk controller

W5100 Eth shield

Intel PRO/1000 Ethernet

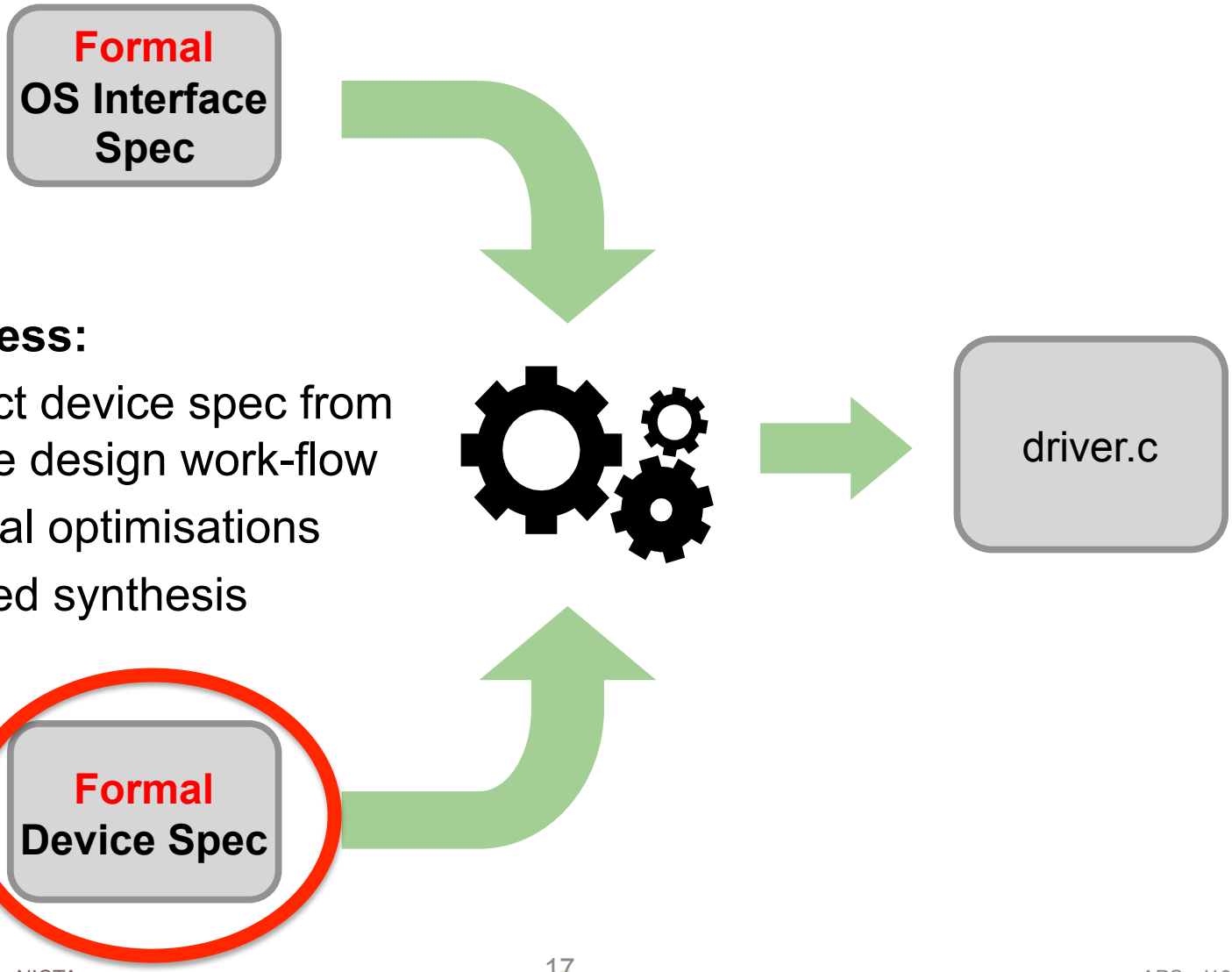
UART controller

Asix AX88772 USB-to-Eth adapter

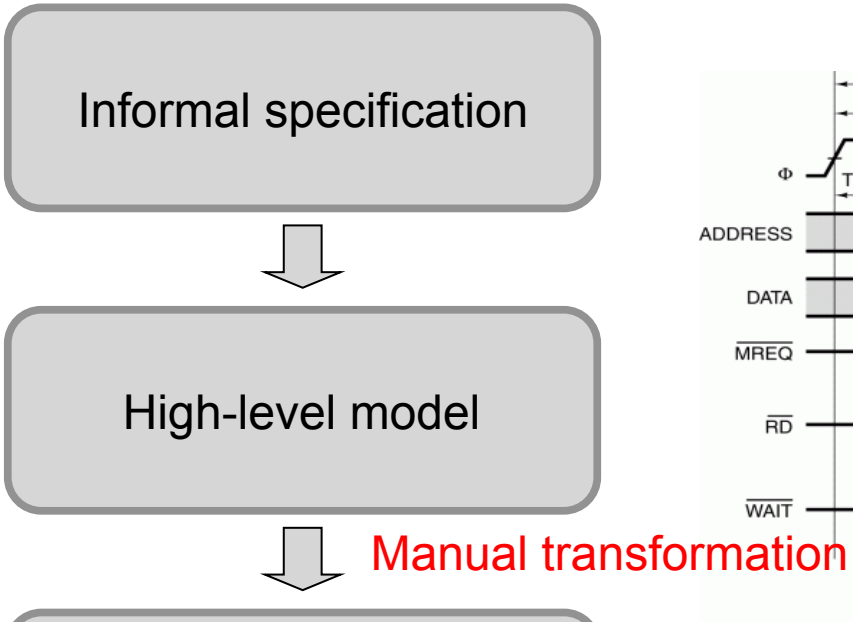


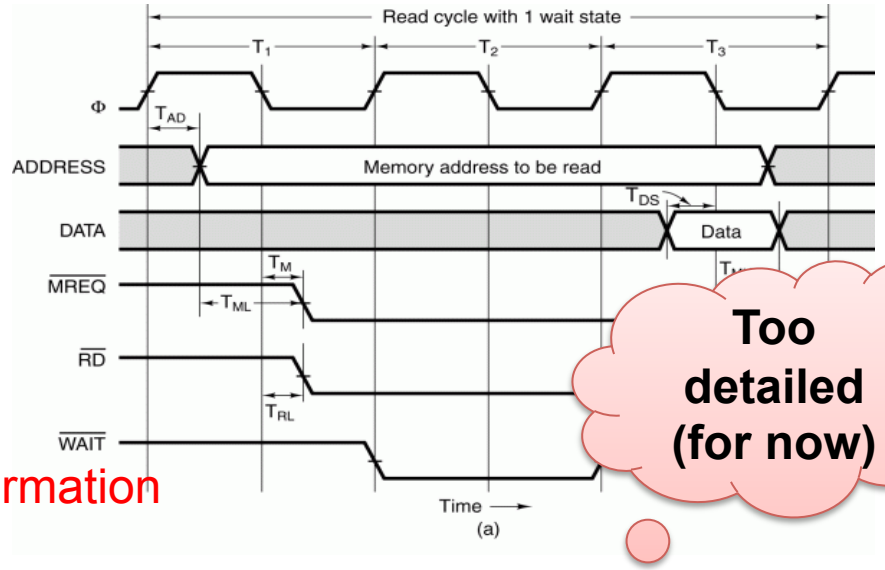
SD host controller

Synthesis: Device Drivers



Hardware Design Workflow

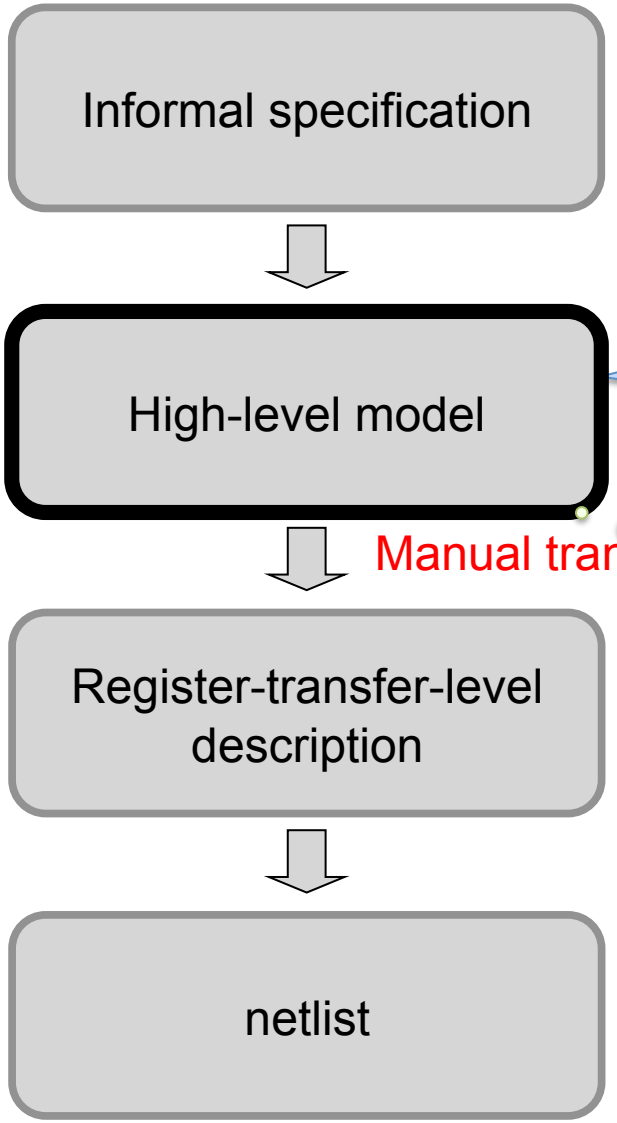




Too detailed (for now)

- Low-level description: registers, gates, wires.
- Cycle-accurate
- Precisely models internal device architecture and interfaces
- “Gold reference”

Hardware Design Workflow

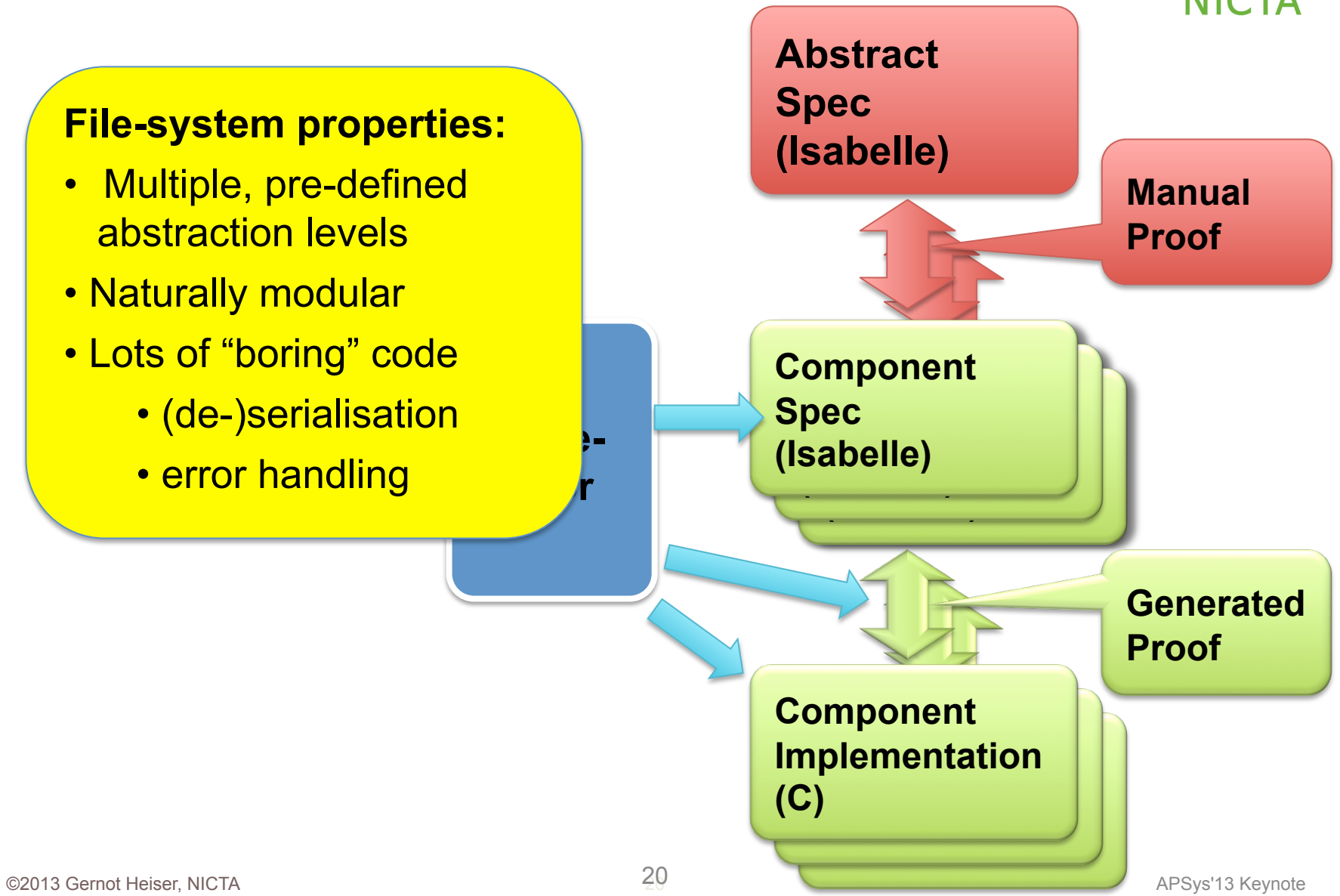


- Captures external behaviour
- Abstracts away structure and timing
- Abstracts away the low-level interface

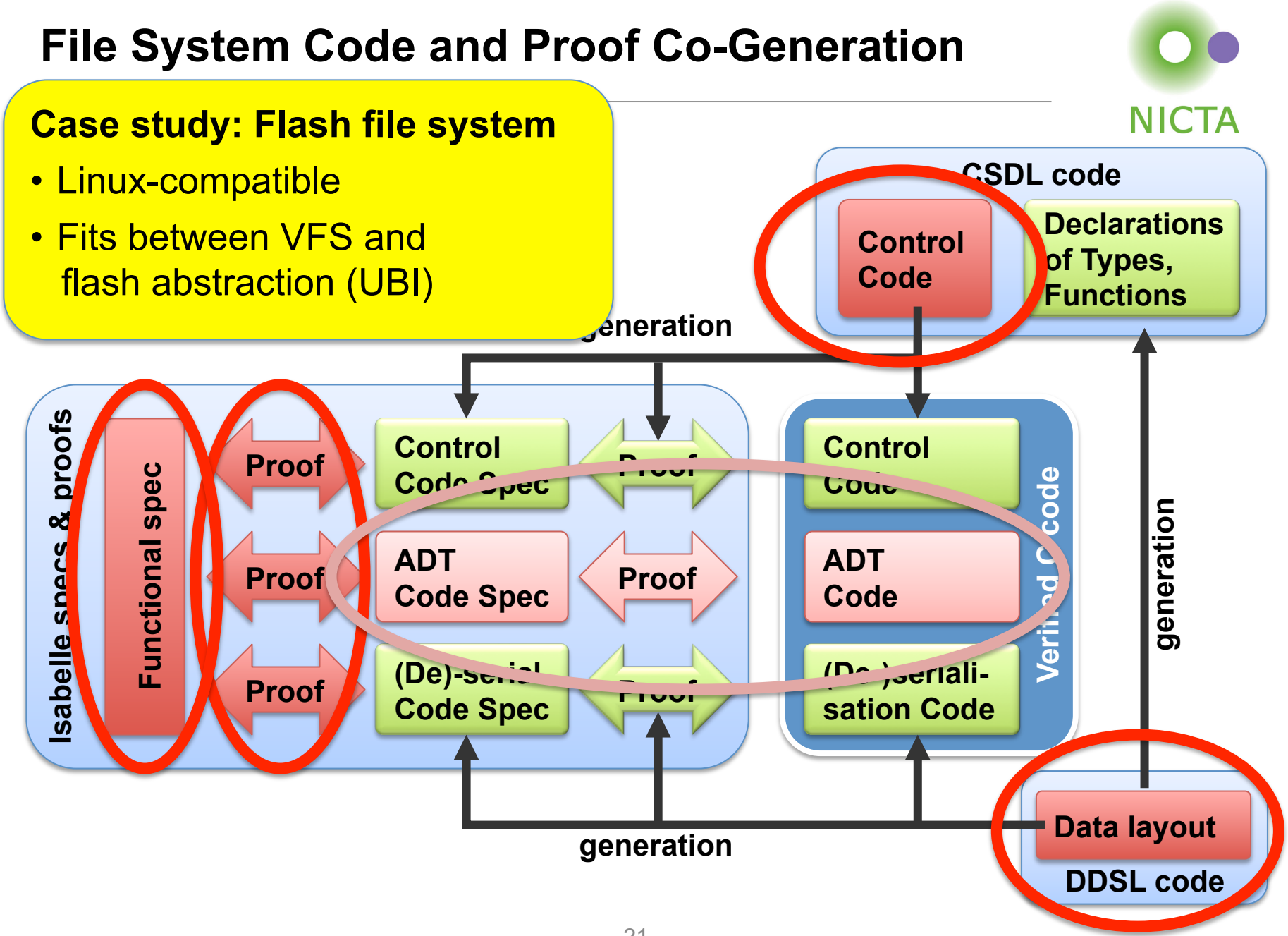
Use for now

```
bus_write(u32 addr, u32 val)
{
  ...
}
```

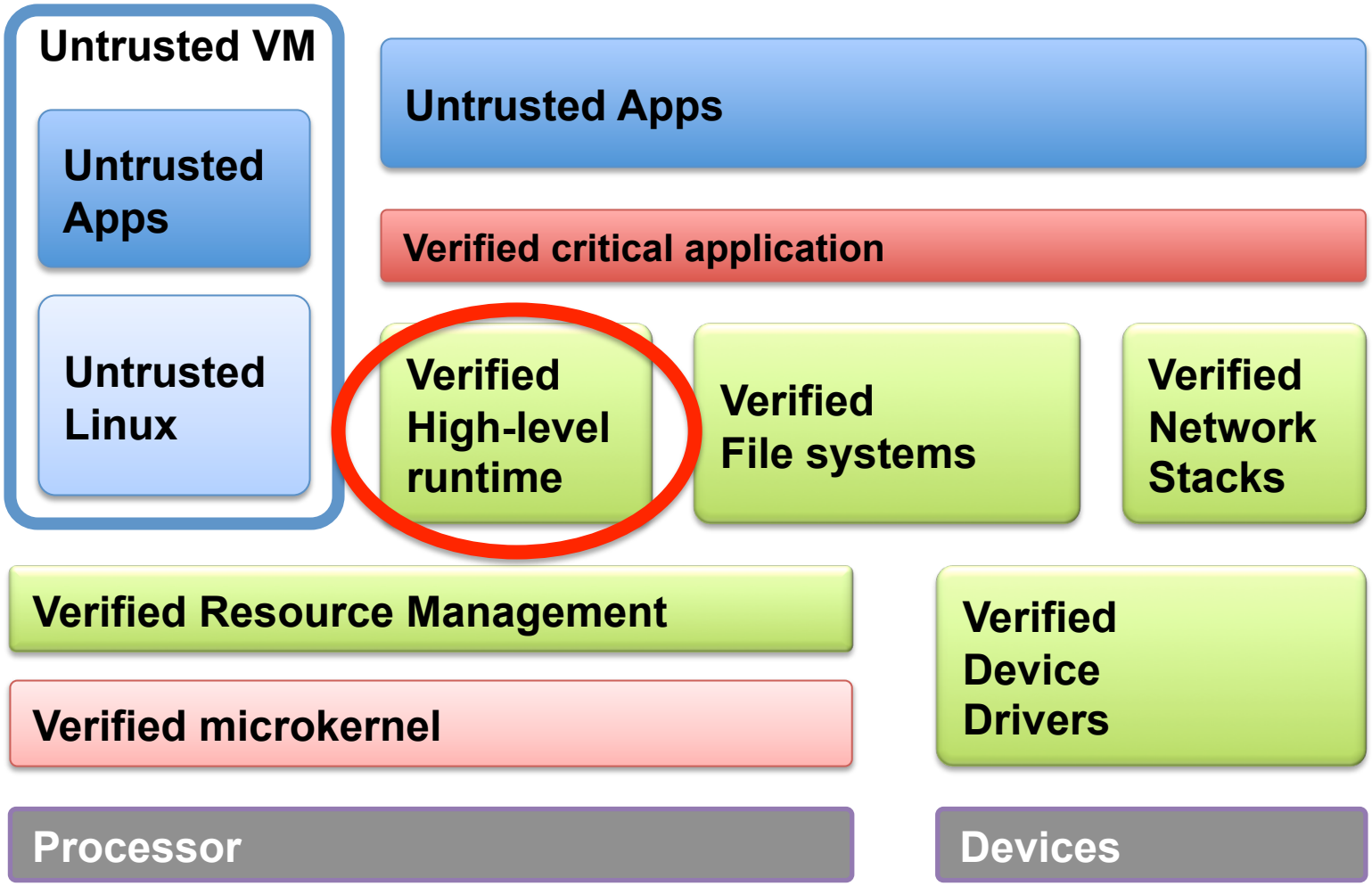
DSLs: File System



File System Code and Proof Co-Generation



Vision: Trustworthy System



Lessons Learnt So Far

Formal methods are expensive?

- Cost-effective for high assurance on small to moderate scale
- $200-400/LOC for 10kLOC

We think we can scale bigger and cheaper:

- Componentisation
  - verify components in isolation – enabled by seL4 guarantees
  - cost – performance tradeoff
- Synthesis
- Abstraction: DSLs, HLLs increase productivity

Big challenge: Proof composition

The next few years will be exciting!