Ogni programma di grandi dimensioni è indistinguibile dalla magia

Alessandro Barenghi

Politecnico Open unix Lab

4 aprile 2023

Un sondaggio veloce

Cosa succede ...

- dall'invio alla ricezione di un messaggio via Whatsapp/Telegram/Signal?
- quando scorrete in verso il basso un feed Twitter/Facebook?
- quando create, modificate, un documento con MS Word
- lo stesso con Google Docs/Office 365?

Un sondaggio veloce

Cosa succede ...

- dall'invio alla ricezione di un messaggio via Whatsapp/Telegram/Signal?
- quando scorrete in verso il basso un feed Twitter/Facebook?
- quando create, modificate, un documento con MS Word
- lo stesso con Google Docs/Office 365?

Magia!

- ... oppure, uno o più computer eseguono uno o più programmi

Calcolatori, calcolatori ovunque

Quanti computer ci sono in questa stanza?

- Qualcuno? Qualche decina? Qualche centinaio?

Ogni tecnologia sufficientemente avanzata... è un computer

- Desktop, laptop, server
- Smartphone, a.k.a., il vero personal computer
- Smart*qualcosa*, con qualcosa $\in \{$tv, microfono, lavatrice$\}$

Ok,ma cos'è un computer?

Church-Turing Thesis

What's in a name? That which we call a general computer, by any other shape would compute anything all the same.

– *A.M. Turing, A. Church, W. Shakespeare*

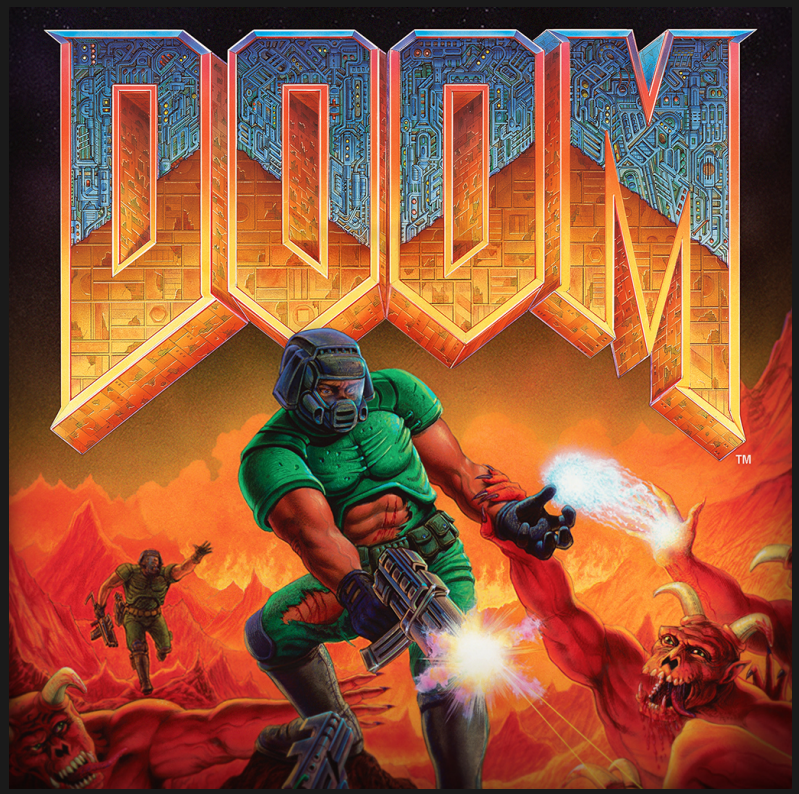
Perchè sono ovunque?

- Per cambiare la funzione svolta da un dispositivo dedicato ad uno scopo serve riprogettarlo e cambiarlo
- Per cambiare la funzione svolta da un computer basta riprogrammarlo

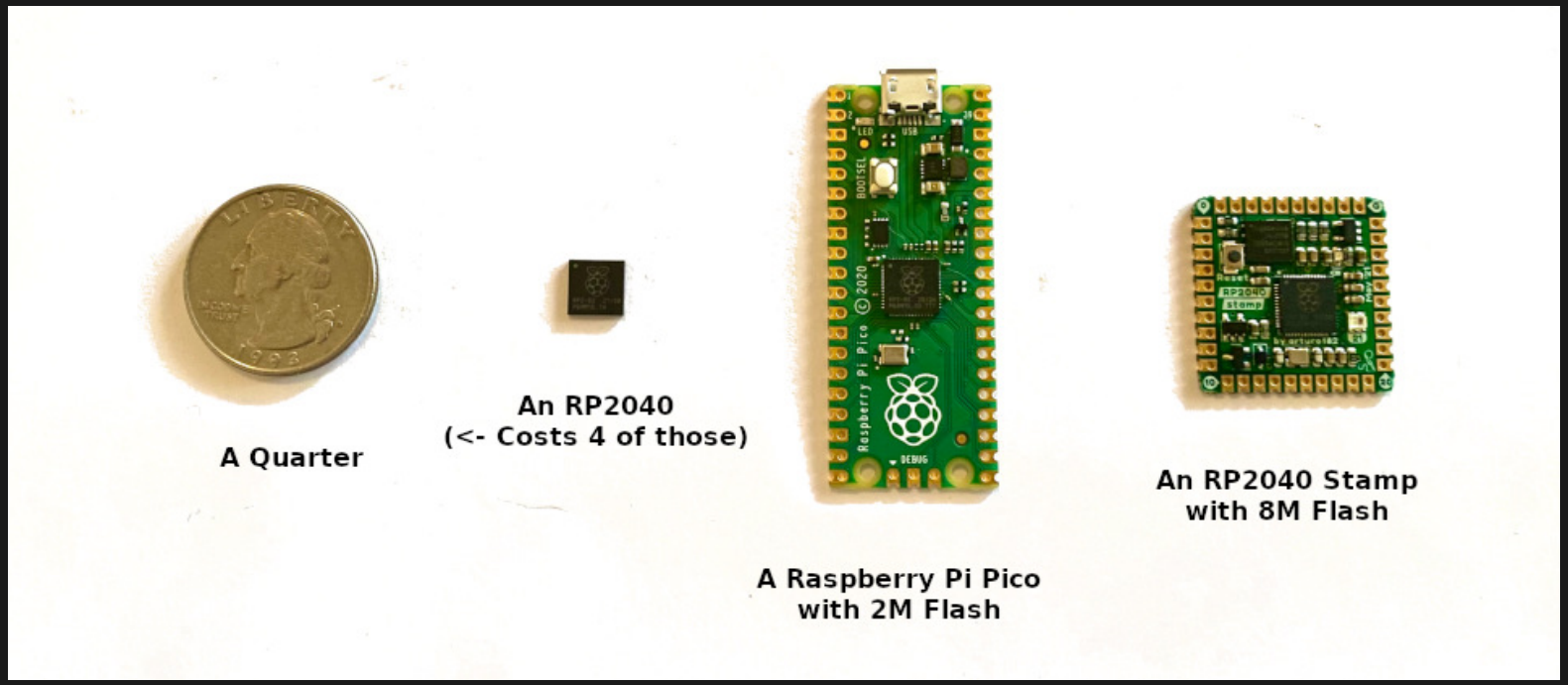
Does it run Doom?

Quindi posso eseguire ovunque un programma qualsiasi?

- DooM (1993): raccomandati 486 @66 MHz, 8 MiB RAM



Does it run Doom?



Cortex-M0+ 133 MHz, 264 kiB RAM e, sì, ci gira Doom [6]

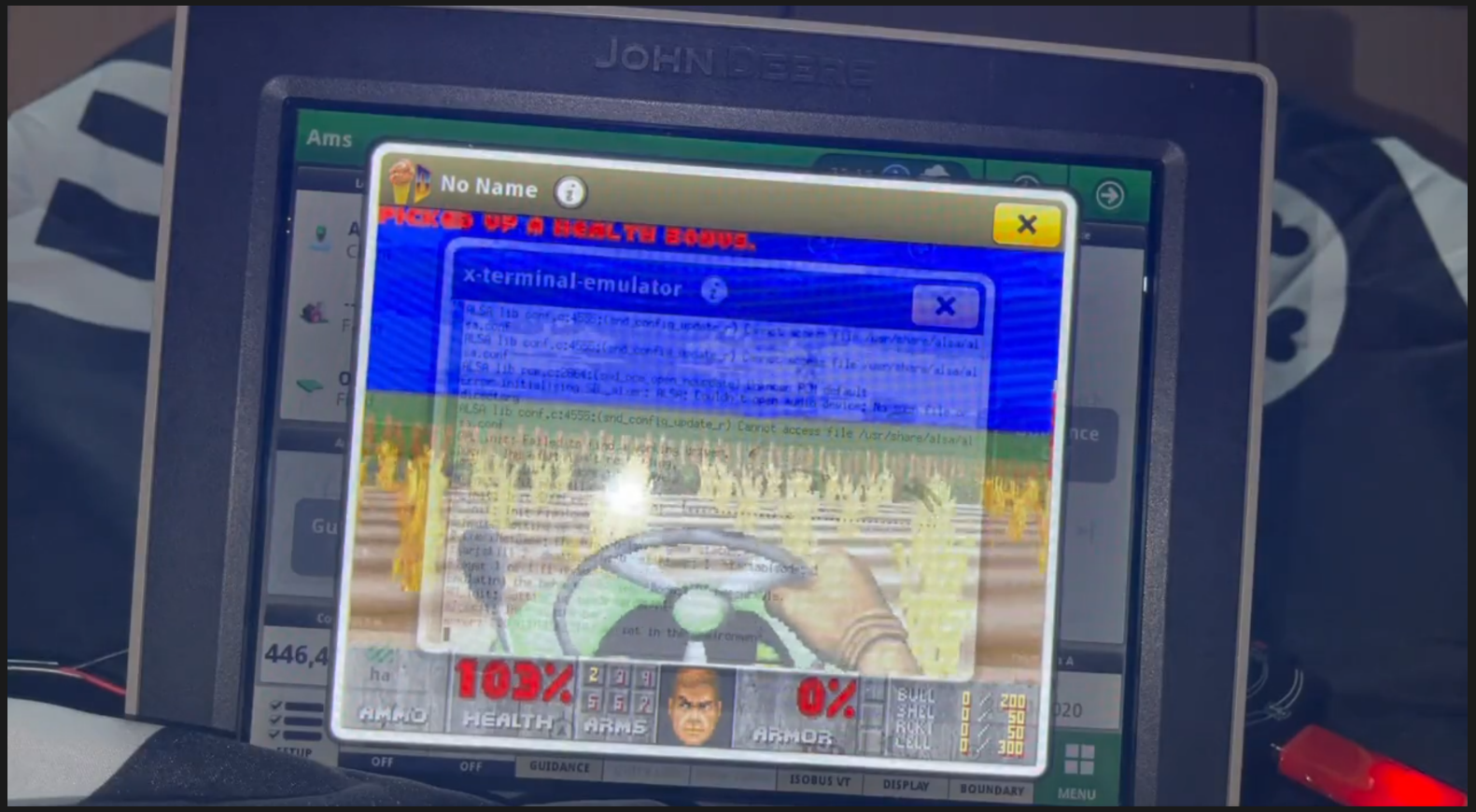
Does it run Doom?

Posso impacchettare un computer in un mattoncino Lego e...
Sì, ci gira Doom [1]

Does it run Doom?

John Deere 6105E, 4500 cm^3, 106 HP

Does it run Doom?



... con un computer di bordo, su cui gira Doom [3]

La programmazione determina lo scopo ....

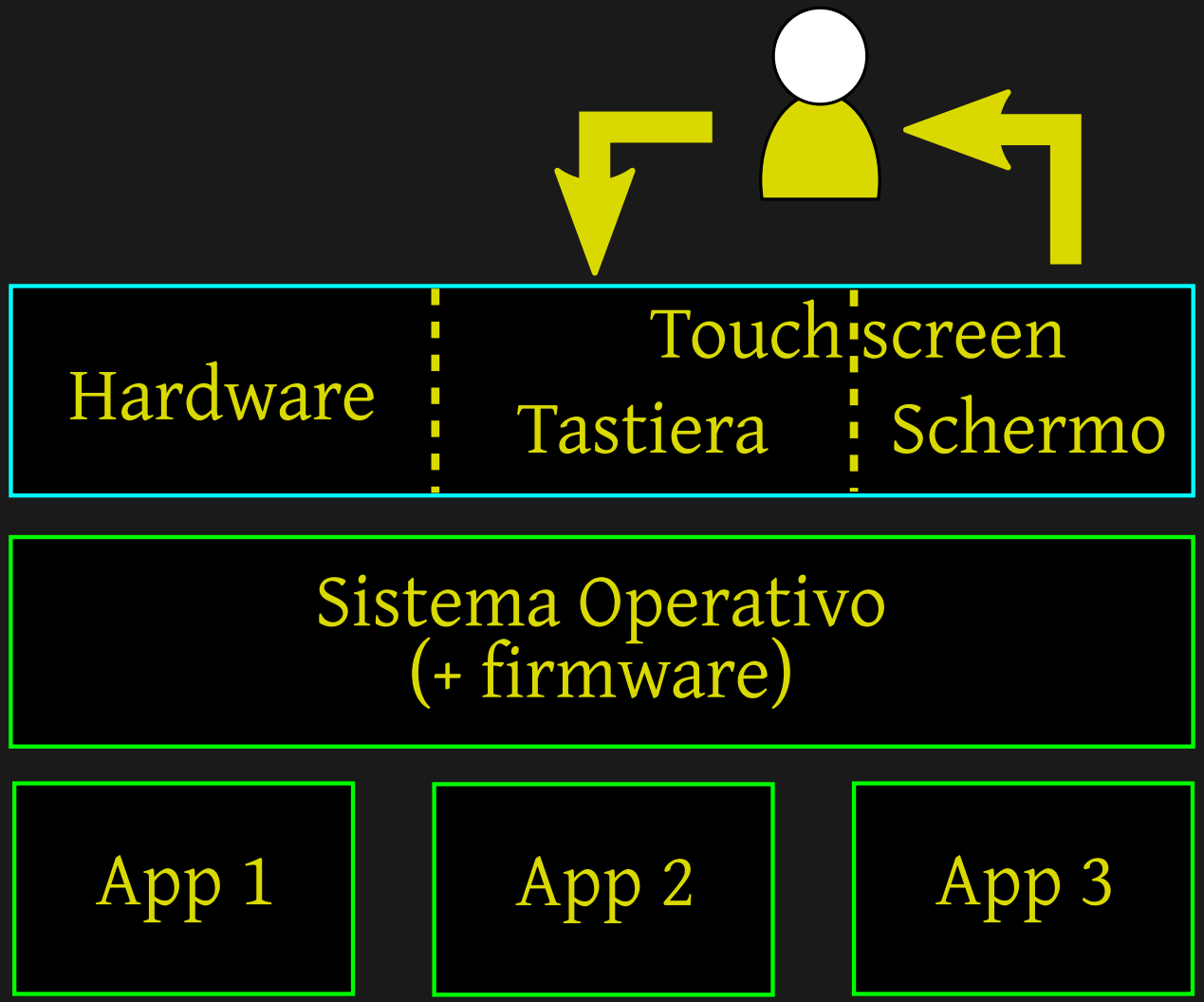
Computer con accessori

- Smartphone: computer con mic, telecamera, sensore di movimento, ricevitore GPS, scheda di rete, modem
- Un laptop: computer con mic, telecamera, scheda di rete
- Un trattore: computer con ricevitore GPS, modem, aratro

Una questione di fiducia

Usare un computer equivale a fidarsi che il software faccia tutto e solamente ciò che vogliamo che faccia

Di quanti elementi ci stiamo fidando?



Una scelta per ogni elemento

Ci fidiamo

- Più efficiente
- Non richiede competenze specifiche
- Serve qualcuno di cui fidarsi

Verifichiamo

- Meno efficiente
- Richiede competenze specifiche
- Può essere divertente[1]

[1] oppure orribilmente frustrante

Applicazioni

Fidato perchè prodotto da sorgente fidata

- Ci si fida di chi produce il software
  - tipicamente perchè ha una buona reputazione
- Serve essere certi che l'applicazione che si usa sia effettivamente quella prodotta dal produttore

Repositories (stores)

- Apple - App store (2008)
- Microsoft - Microsoft Store (2012)
- Distro Linux - repositories (Debian - 1995/1998)

Applicazioni

Come ispezionare applicazioni?

- Ispezionare il binario che uso
- Ispezionare il sorgente da cui si ottiene il binario

Un problema di scala

- Intera opera di Shakespeare: 182 krighe
- Intera serie “La ruota del tempo”: 297 krighe
- Visualizzatore di PDF che sto usando: 83 krighe
- Libreoffice 7.5: 2,14 Mrighe

Open source e Free software

Open source software

- Consente a tutti di ispezionare il codice dell'applicazione
- Mitiga il problema di scala: più occhi sullo stesso codice

open source initiative®

Software Libero

- Impone l'ispezionabilità del sorgente (mitiga il problema di scala)
- Dà diritto anche a eseguire modificare e ridistribuire l'applicazione liberamente

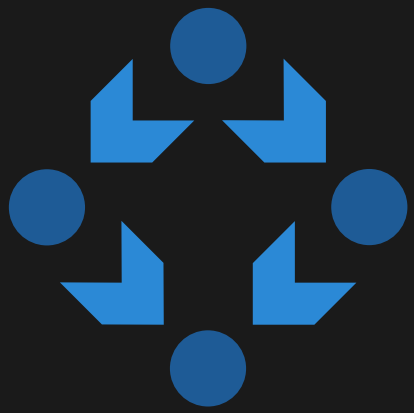
Applicazioni - Il binario è ottenuto dal sorgente?

Compilatori e vulnerabilità

- Abbiamo assunto che il compilatore traduca fedelmente
- Modifiche a XCode per impiantare trojan in sorgenti sani: [7]

Ricostruisco...

- il sorgente dal binario che uso $\rightarrow$ il prossimo intervento
- il binario che uso dal sorgente che ho letto $\rightarrow$ non immediato



Reproducible Builds

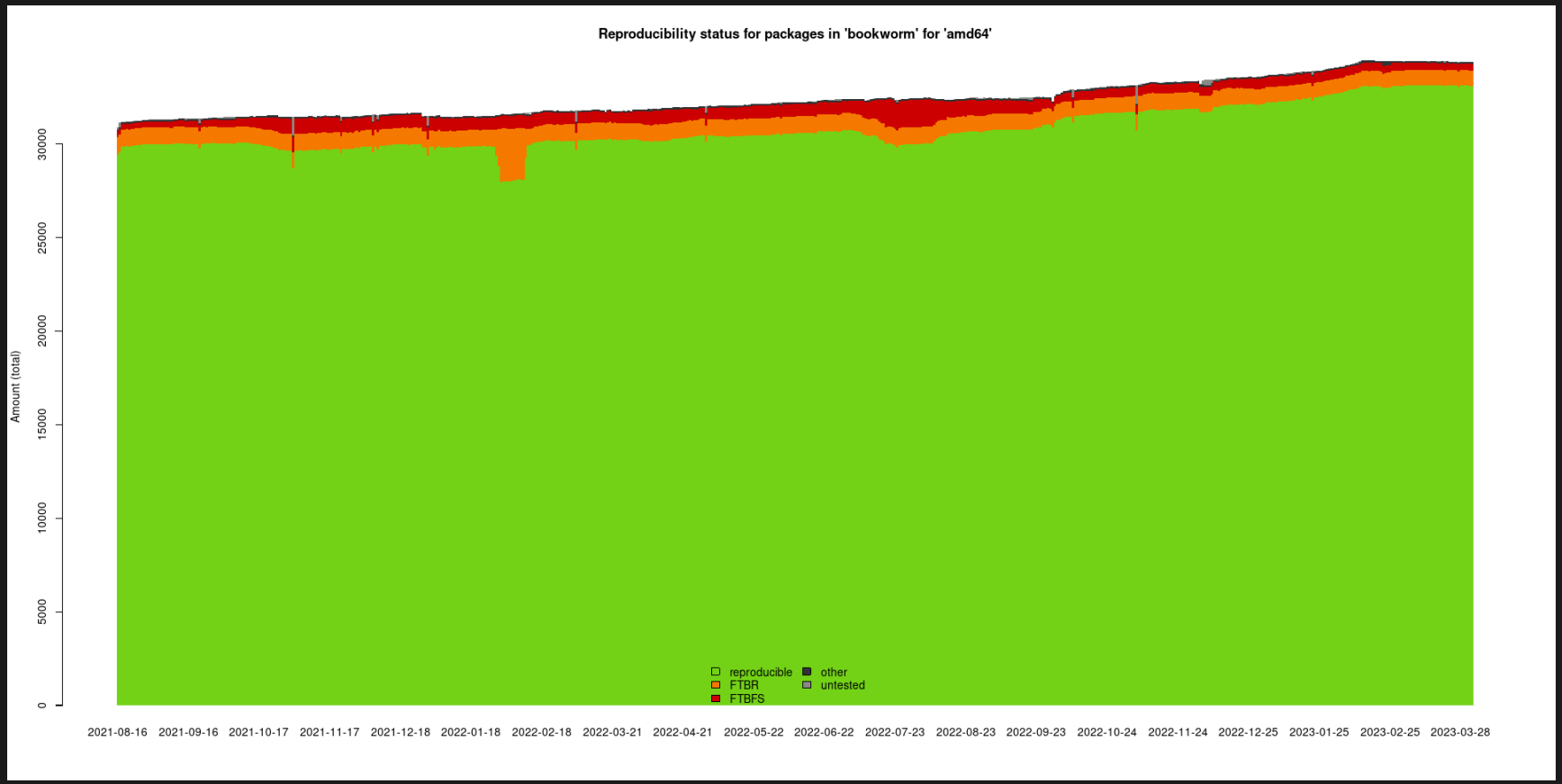
Ostacoli tipici

- Ordine di emissione del codice: fornendo sorgenti in ordine diverso a compilatore/collegatore cambia il binario
- Timestamps: inclusi da molti strumenti di compilazione/pacchettizzazione

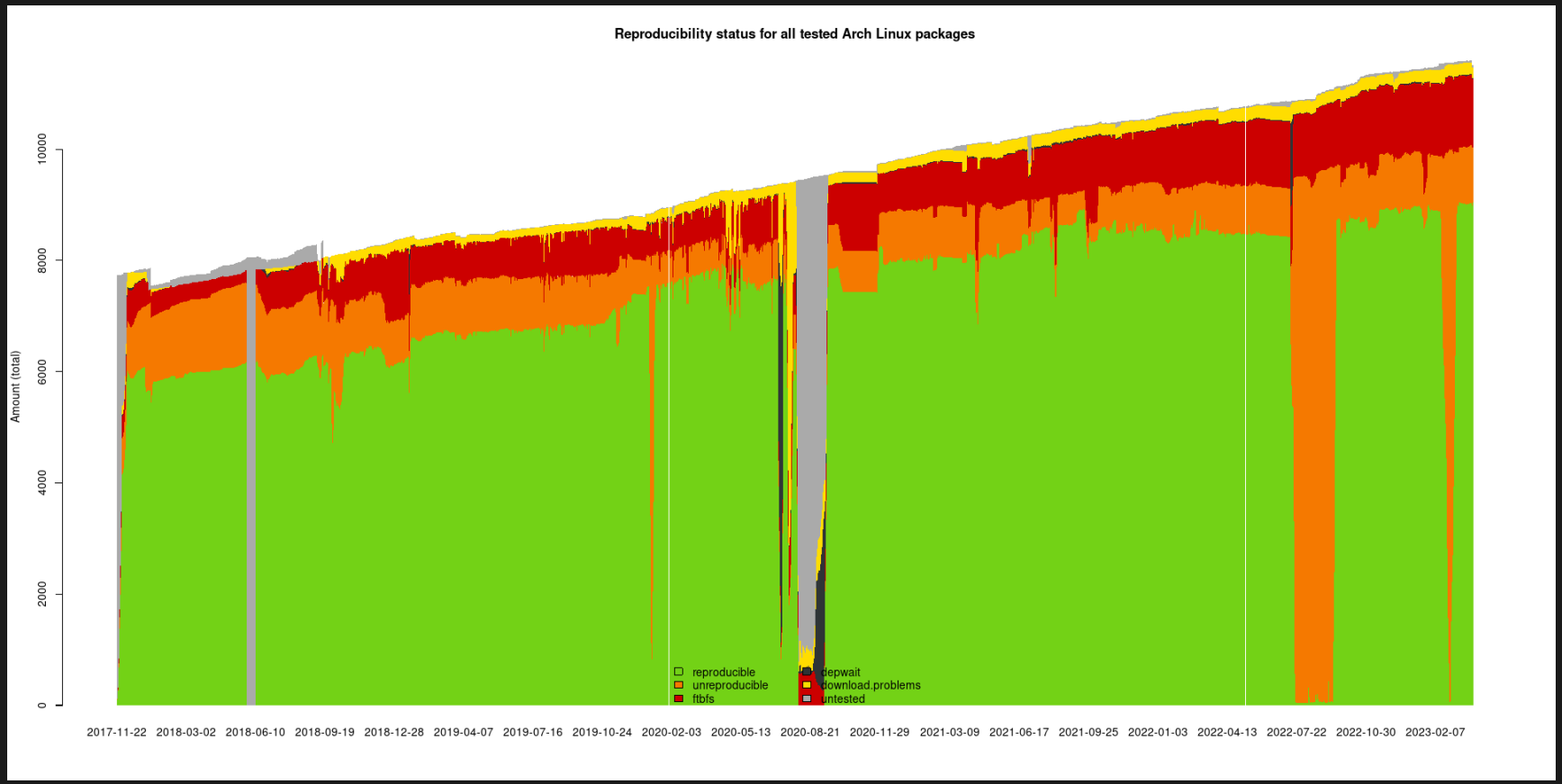
Esempi

- Debian (started 2013): includere plot
- NixOS: pensata per ottenere 100% riproducibilità

Reproducible builds: Debian bookworm amd64



Reproducible builds: Arch amd64



Compilare compilatori

Come ottengo un compilatore (binario eseguibile)?

- Per compilare un compilatore serve un compilatore compilato
- Il primo compilatore era scritto in binario, a mano

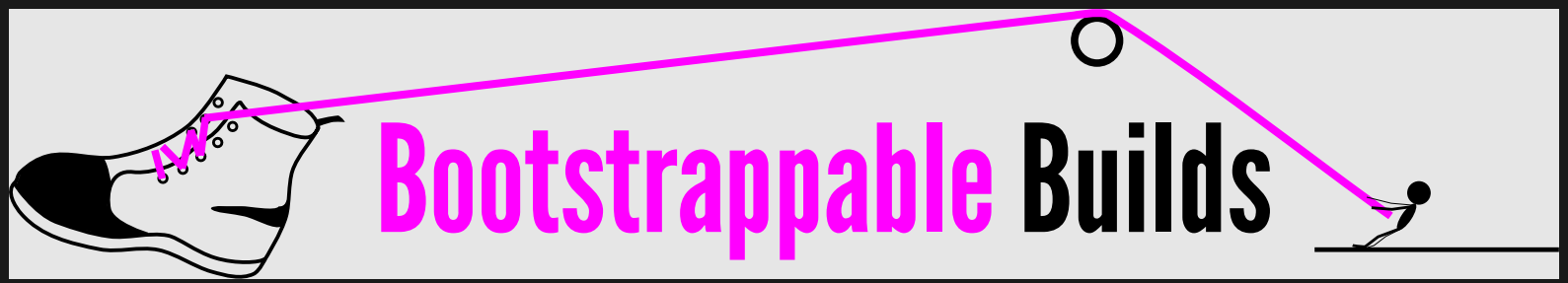
Fidarsi del compilatore

- Ken Thompson - Reflections on trusting trust [9]
- Mitigazione [10]: compilo il compilatore con sè stesso e con un compilatore fidato, "confronto" i risultati
- Mi serve almeno un compilatore di cui mi fido
  - oppure di cui, se voglio, posso leggere il binario

Un compilatore fidato

Bootstrappable builds[4]

- Progetto per ridurre al minimo il binario iniziale da leggere
- Bootstrap: hex0 $\rightarrow$ Mes $\rightarrow$ MesCC $\rightarrow$ gcc 2.95 $\rightarrow$ gcc 4.7.4
- Seed originale (hex0) ridotto a 357B [5]
- Obiettivo: Bootstrap di un'intera distribuzione (GNU Guix)
- Bootstrapping Arch: https://wiki.archlinux.org/title/Bootstrappable



Sistema operativo

Dopotutto, è solo un programma complesso

- Forte concorrenza tra aziende che sviluppano OS
  - Un danno di reputazione ha ripercussioni evidenti
- Relativamente semplice ottenerlo in maniera fidata
  - Solitamente, già a bordo del computer

Possibili problemi di fiducia

- OS interagisce con l'HW direttamente, può usarlo
  - mic, telecamera, scheda di rete controllate da esso
- Usato per forzare aggiornamenti e rimozione software

Sistema operativo

Dopotutto, è “solo” un programma complesso

- ... quindi, se ho il sorgente posso analizzarlo tutto
  - ... con molto tempo. Linux 6.2.9: 23,6 Mrighe
- un OS che sia software libero aiuta molto :)

Supporto a verificabilità

- OS per computer critici (e.g., smartcards) hanno dimensioni più ridotte

Firmware?

Computer dentro altri computer

- Una periferica svolge compiti complessi $\rightarrow$ viene realizzata con un(o o più) computer generale(/i)
- Firmware: OS+applicazioni eseguite dalla periferica

Esempi

- Schede di rete Wi-Fi
- Schede video con GPU "ricca"
- Modem 3G/4G/5G
- Dischi fissi (sia elettromeccanici che SSD)

Approcci

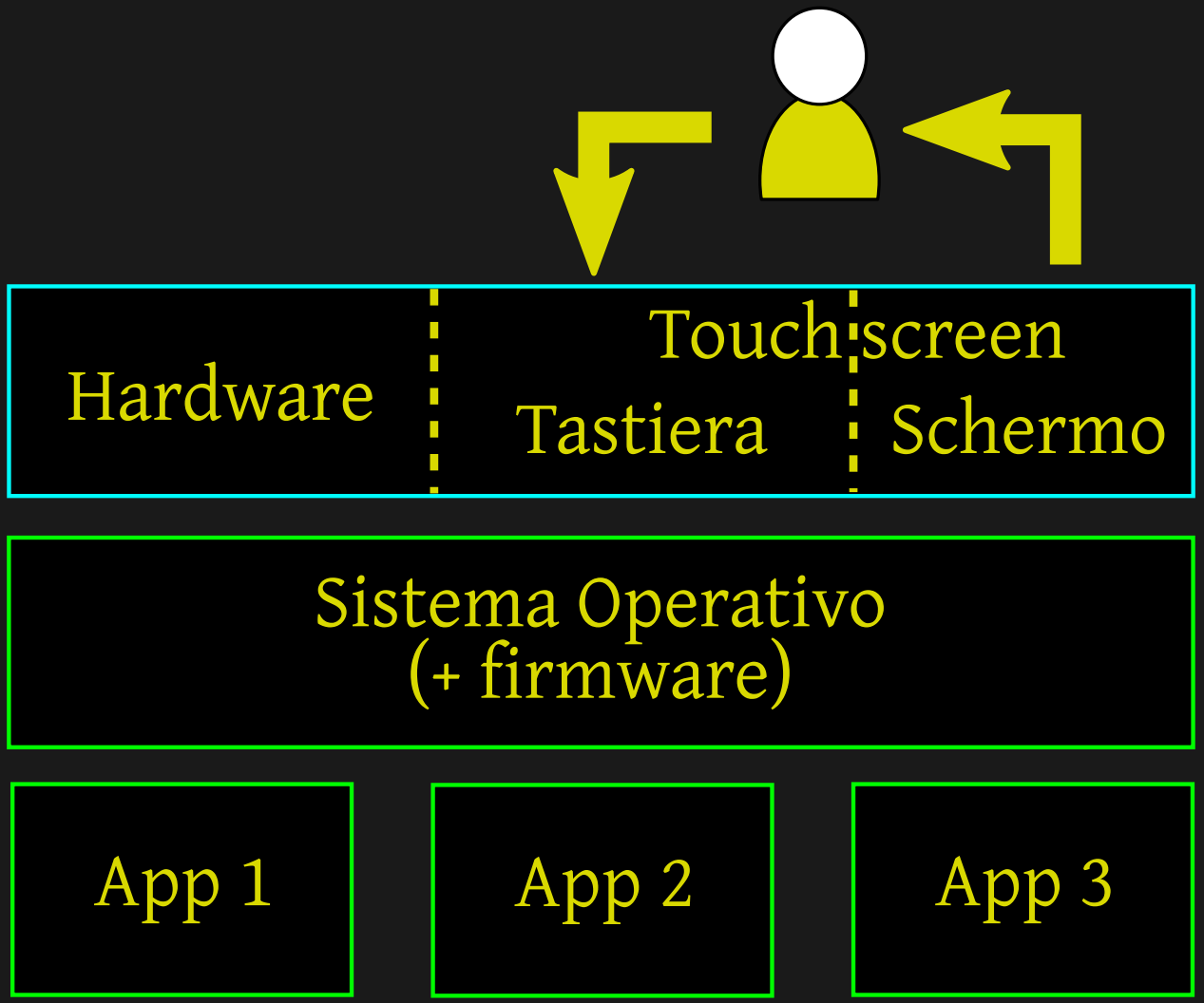
Firmware

- Il produttore di HW fornisce il firmware insieme ai driver, aggiornandolo periodicamente

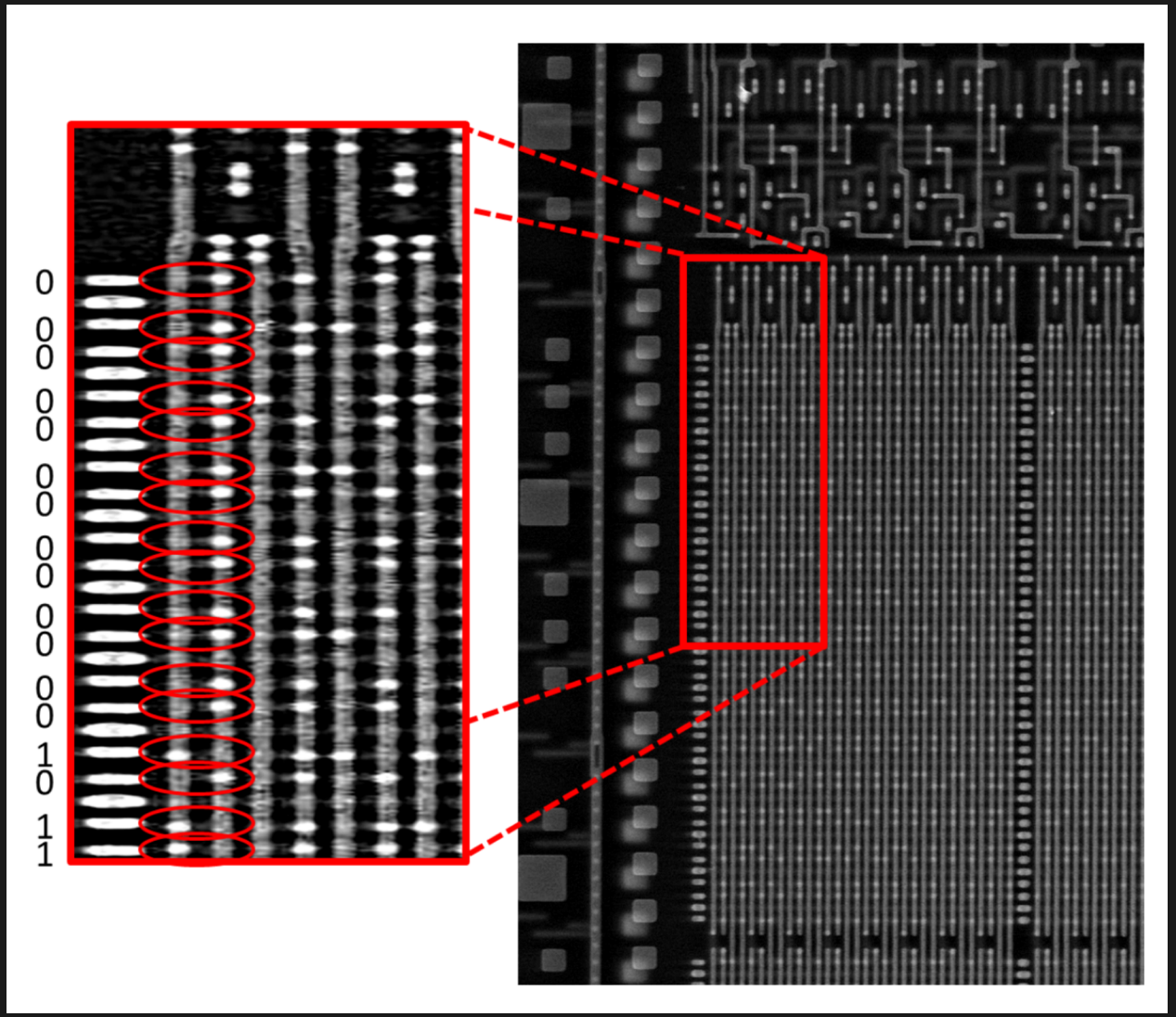
Firmware

- Problema 1: non sono quasi mai disponibili i sorgenti
- Problema 2: a volte non è nota l'ISA della CPU nella periferica
- Prerequisito per soluzione: scelta cosciente dell'hardware
  - Casi senza alternative: GPU e modem 3G/4G/5G compatti
- Riconoscimento di FSF per HW con firmware libero: `https://ryf.fsf.org/`

Hardware?


Hardware
Touch screen
Tastiera
Schermo
Sistema Operativo
(+ firmware)
App 1
App 2
App 3

Hardware



Hardware ispezionabile/"riproducibile"

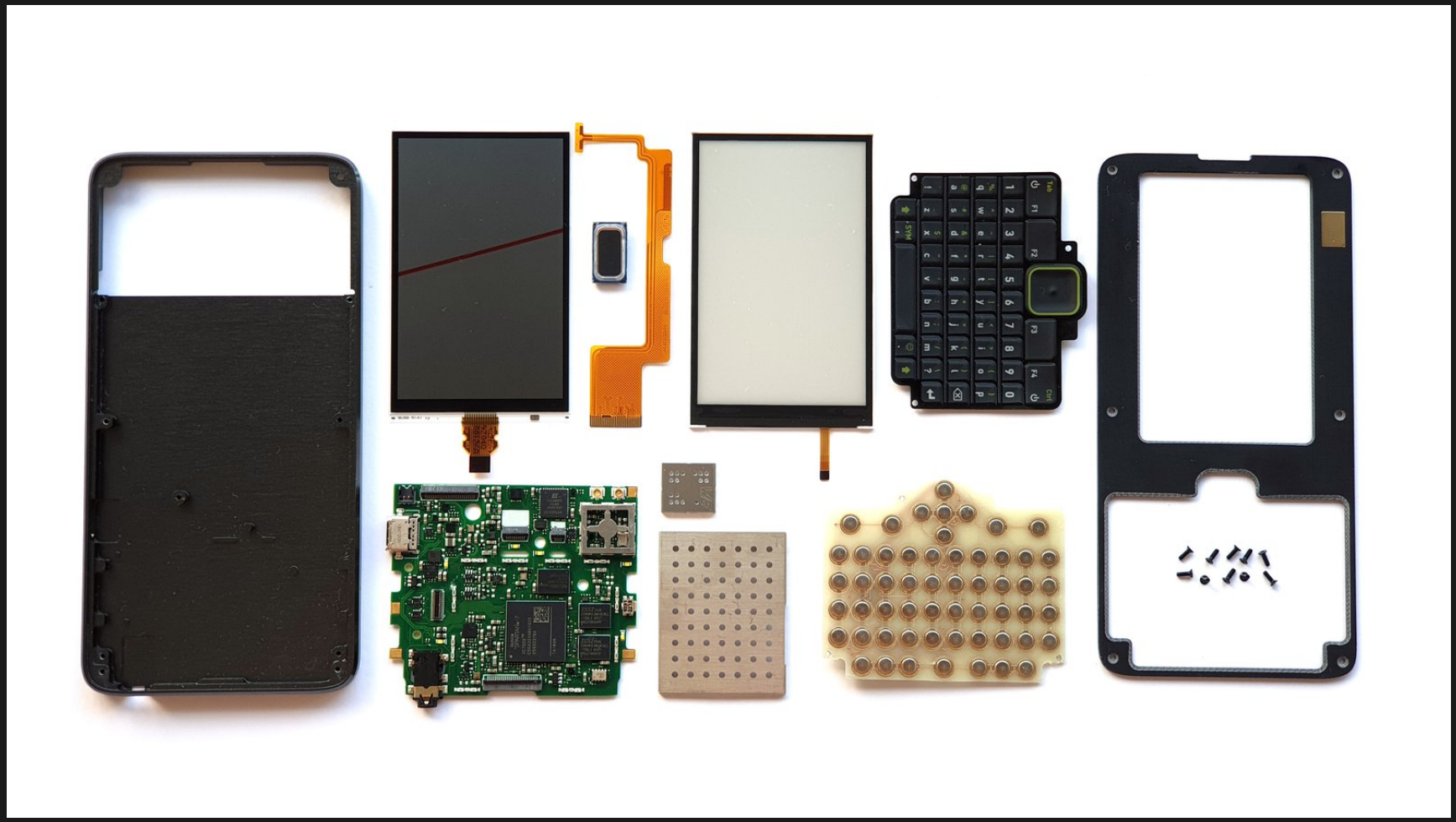
Hardware programmabile

- Per inserire una modifica malevola, l'attaccante deve conoscere il mio progetto HW
- Idea: usare hardware programmabile (FPGA), fornire il progetto in modo verificabile
  - L'utente programma il processore sul dispositivo al primo uso

Periferiche

- Caso ideale: le periferiche sono componenti dedicati, ispezionabili

Precursor [2]



SoC ospitato su una Xilinx Spartan XC7S50

Ancora il problema del compilatore

Hardware Description Languages

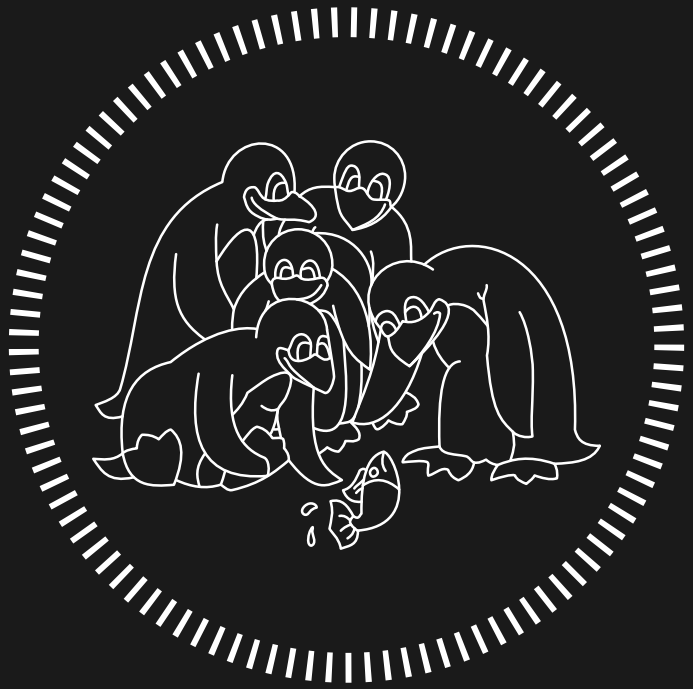
- Precursor offre la descrizione del SoC in HDL
- L'FPGA viene programmata con un bitstream ottenuto da essa
- Come essere sicuri della traduzione?

Toolchain di sintesi HDL open source

- Yosys e nextpnr sono una toolchain di sintesi,place and route per FPGA open source [8]
- Supporto per FPGA di Lattice, lavori in corso per Xilinx

Una questione di scelte

Grazie per l'attenzione!



Domande?

Ask Me Anything

Bibliography I

James "Ancient" Brown.
Doom in un mattoncino lego.
https://www.youtube.com/watch?v=o76U0JPrMFk.

Andrew "bunnie" Huang.
Precursor.
https://www.bunniestudios.com/blog/?p=5921.

Sick Codes.
Hacking the farm: Breaking badly into agricultural devices.
https://forum.defcon.org/node/241833.

janneke.
Bootstrappable builds.
https://bootstrappable.org/.

janneke.
Gnu mes - full source bootstrap.
FOSDEM 21https:
//raw.githubusercontent.com/oriansj/talk-notes/master/fosdem_2021/gnu_mes_fosdem21.pdf.

Graham Sanderson.
Rp2040 doom.
https://kilograham.github.io/rp2040-doom/.

Jeremy Scahill and Josh Begley.
The cia campaign to steal apple's secrets.
The Intercept - https://theintercept.com/2015/03/10/ispy-cia-campaign-steal-apples-secrets.

Bibliography II

David Shah, Eddie Hung, Clifford Wolf, Serge Bazanski, Dan Gisselquist, and Miodrag Milanovic.

Yosys+nextpnr: An open source framework from verilog to bitstream for commercial fpgas.
In *27th IEEE Annual International Symposium on Field-Programmable Custom Computing Machines, FCCM 2019, San Diego, CA, USA, April 28 - May 1, 2019*, pages 1–4. IEEE, 2019.

Ken Thompson.

Reflections on trusting trust.
Commun. ACM, 27(8):761–763, 1984.

David A. Wheeler.

Countering trusting trust through diverse double-compiling.
In *21st Annual Computer Security Applications Conference (ACSAC 2005), 5-9 December 2005, Tucson, AZ, USA*, pages 33–48. IEEE Computer Society, 2005.