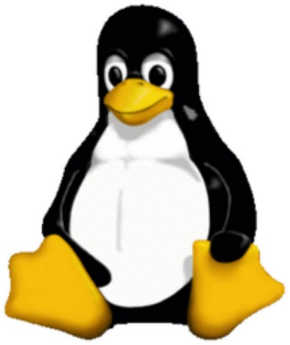
Slides: https://github.com/pdp7/talks/blob/master/nerp-riscv.pdf

RISC-V and open source chip design

NERP meetup at PS1 hackerspace (2020-01-27)



Drew Fustini
OSH Park
drew@oshpark.com
@oshpark / @pdp7

CC BY SA

CC BY SA

- Open Source Hardware designer at OSH Park
  - PCB manufacturing service in the USA
  - drew@oshpark.com / Twitter: @oshpark
- Volunteer Member of Board of Directors of BeagleBoard.org Foundation
  - **drew@beagleboard.org**
- Volunteer Member of the Board of Directors of the Open Source Hardware Association (OSHWA)
  - **drew@pdp7.com**

Linux on Open Source Hardware with Open Source chip design

Drew Fustini

CERN Open Hardware Licence

- Originally written for **CERN** designs hosted in the **Open Hardware Repository**
- Can be used by **any designer** wishing to **share design** information using a **license compliant** with the **OSHW definition criteria**.
- CERN OHL version 1.2
  Contains the license itself and a guide to its usage

Slides: https://github.com/pdp7/talks/blob/master/nerp-riscv.pdf

Section:

RISC-V

the instruction set for everything?

- My column in the latest Hackspace Magazine is an introduction to RISC-V and how it is enabling open source chip design:
  - hackspace.raspberrypi.org/issues/27/


Drew Fustini

COLUMN SPARK

Open-source chips

Breaking free of chip design monopolies with RISC-V

Drew Fustini

@pdp7

Drew Fustini is a hardware designer and embedded Linux developer. He is the Vice President of the Open Source Hardware Association, and a board member of the BeagleBoard.org Foundation. Drew designs circuit boards for OSH Park, a PCB manufacturing service, and maintains the Adafruit BeagleBone Python library.

When we think about what open-source hardware means, we usually think about the board design being freely available. But what about the processor? Is there a way to make hardware that is truly open source? This month's column is dedicated to an exciting – and surprisingly political – development in chip design.

When you write a program in the Arduino IDE, it is compiled into instructions for the microcontroller to execute. How does the compiler know what instructions the chip understands? This is defined by the Instruction Set Architecture. The ISA is a standard, a set of rules that define the tasks the processor can perform.

> **The ISA is a standard, a set of rules that define the tasks the processor can perform**

Chances are that both your laptop and the data centre streaming your favourite movie are using an ISA owned by Intel or AMD. The processor in your smartphone is almost certainly using a proprietary ISA licensed from ARM. Proprietary standards can be overpriced, prevent innovation, or even disappear altogether when companies change strategy.

Enter RISC-V, a free and open ISA created by researchers at UC Berkeley, led by Krste Asanović and David Patterson. "We were always jealous that you could get industrial-strength software that was open," Patterson explained to VentureBeat at the RISC-V Summit back in December. "But when it came to hardware, it was proprietary. Now, with RISC-V, we get the same kind of benefit. It helps education, and it helps competition."

This open standard proved to be useful outside of academia. Nvidia and Western Digital are now shipping millions of devices with RISC-V processors. These companies have the freedom to leverage open-source implementations while avoiding ARM licensing fees – which can really add up when shipping large volumes.

Nations such as India see the importance of being able to create processors that are not under the control of a foreign corporation, who may be forced to build in backdoors for their own government. There is also strong interest from chipmakers in China, especially now that US companies have been banned from doing business with Huawei.

With these financial and political motivations, plus an increasingly mature software ecosystem, including Linux support, it won't be long before you have a device with a RISC-V processor in your home or pocket.

You can learn more about the exciting possibilities that RISC-V unleashes from Dr. Megan Wachs by pointing your web browser to **hsmag.cc/qw4ed1**.

- When you write a program in the Arduino IDE, it is compiled into instructions for the microcontroller to execute.

- How does the compiler know what instructions the chip understands?
  - defined by the **Instruction Set Architecture**
  - The **ISA** is a standard, a set of rules that define the tasks the processor can perform.
  - Examples: x86 (Intel/AMD) and ARM
    - Both are proprietary and need commercial licensing

- **RISC-V: Free and Open RISC Instruction Set Arch**
  - “new instruction set architecture (ISA) that was originally designed to support computer architecture research and education and is now set to become a standard open architecture for industry”

- **RISC-V: Free and Open RISC Instruction Set Arch**
  - Instruction Sets Want To Be Free: A Case for RISC-V
    - David Patterson, UC Berekely – *co-creator of the original RISC!*
    - https://www.youtube.com/watch?v=mD-njD2QKN0
  - RISC-V Summit 2019: State of the Union
    - Krste Asanovic, UC Berkeley
    - https://www.youtube.com/watch?v=jdkFi9_Hw-c

State of the Union

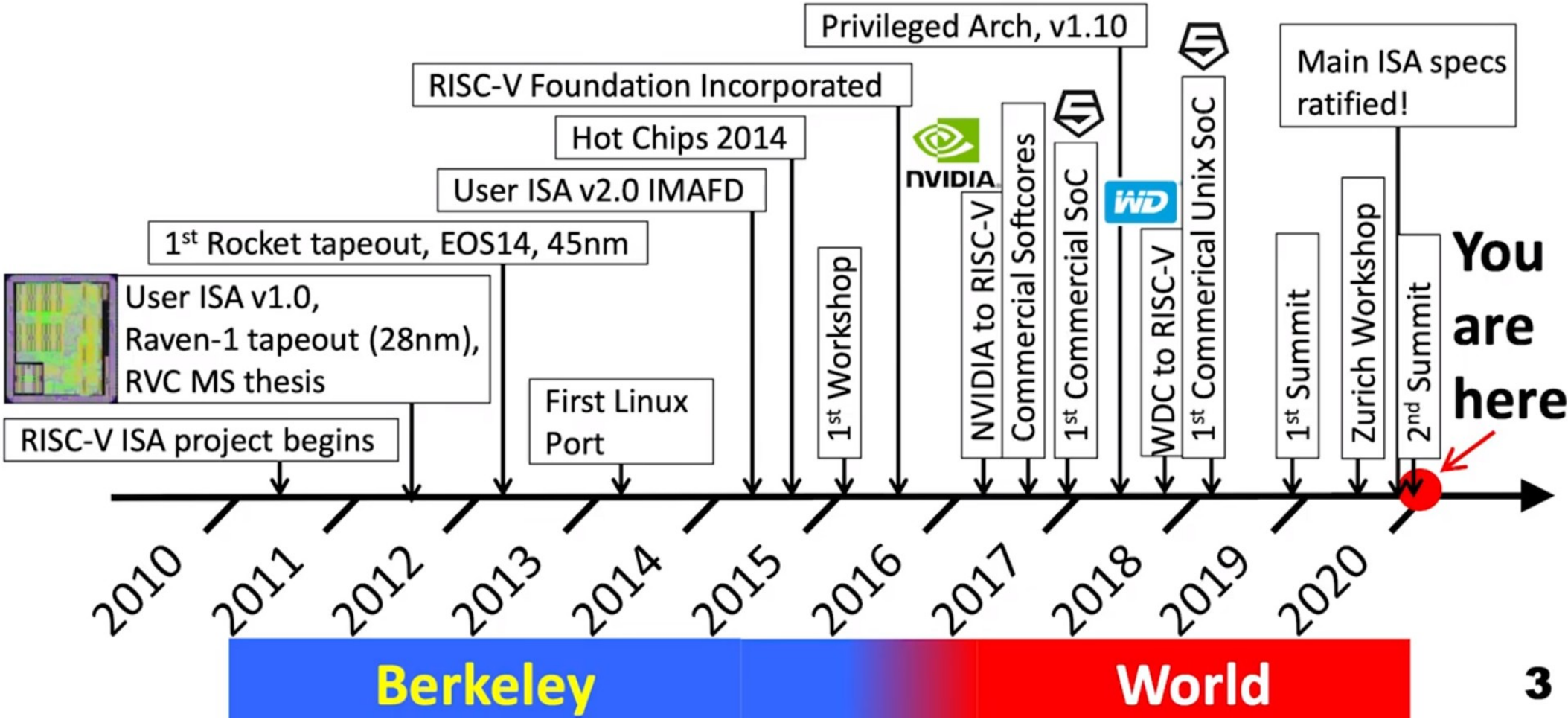
Krste Asanovic
UC Berkeley, RISC-V Foundation, & SiFive Inc.
krste@berkeley.edu

RISC-V Summit
San Jose Convention Center, CA, USA
December 10, 2019

What's Different about RISC-V?

- ***Simple***
  - Far smaller than other commercial ISAs
- ***Clean-slate design***
  - Clear separation between user and privileged ISA
  - Avoids µarchitecture or technology-dependent features
- A ***modular*** ISA designed for ***extensibility/specialization***
  - Small standard base ISA, with multiple standard extensions
  - Sparse and variable-length instruction encoding for vast opcode space
- ***Stable***
  - Base and standard extensions are frozen
  - Additions via optional extensions, not new versions
- ***Community designed***
  - With leading industry/academic experts and software developers

RISC-V Timeline



RISC-V Ecosystem

Software

Open-source software:
Gcc, binutils, glibc, Linux, BSD, LLVM, QEMU, FreeRTOS, ZephyrOS, LiteOS, SylixOS, …

Commercial software:
Lauterbach, Segger, IAR, Micrium, ExpressLogic, Ashling, AntMicro, Imperas, UltraSoC …

RISC-V Foundation

ISA specification | Golden Model | Compliance

Hardware

Open-source cores:
Rocket, BOOM, RI5CY, Ariane, PicoRV32, Piccolo, SCR1, Shakti, Serv, Swerv, Hummingbird, …

Commercial core providers:
Alibaba, Andes, Bluespec, Cloudbear, Codasip, Cortus, InCore, Nuclei, SiFive, Syntacore, …

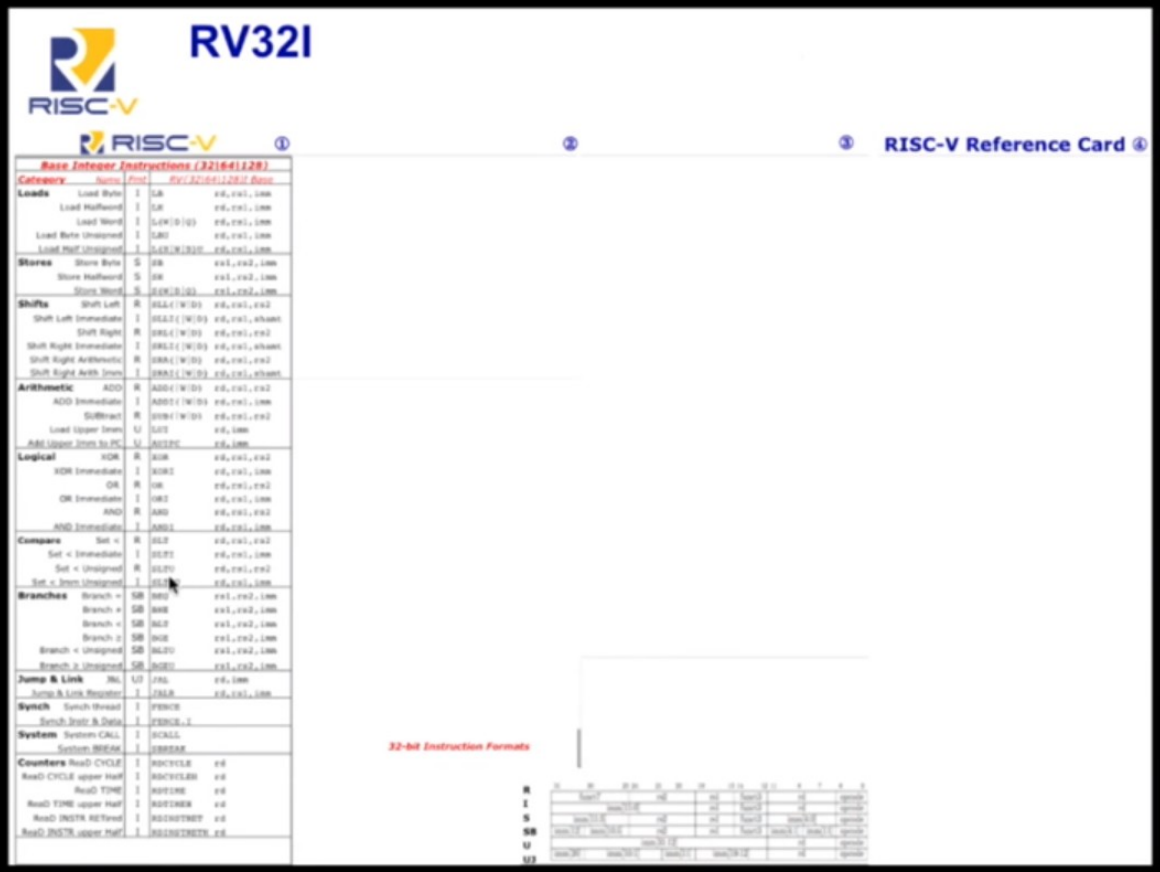
Inhouse cores:
Nvidia, WD, +others

RISC-V Standard Base ISA Details

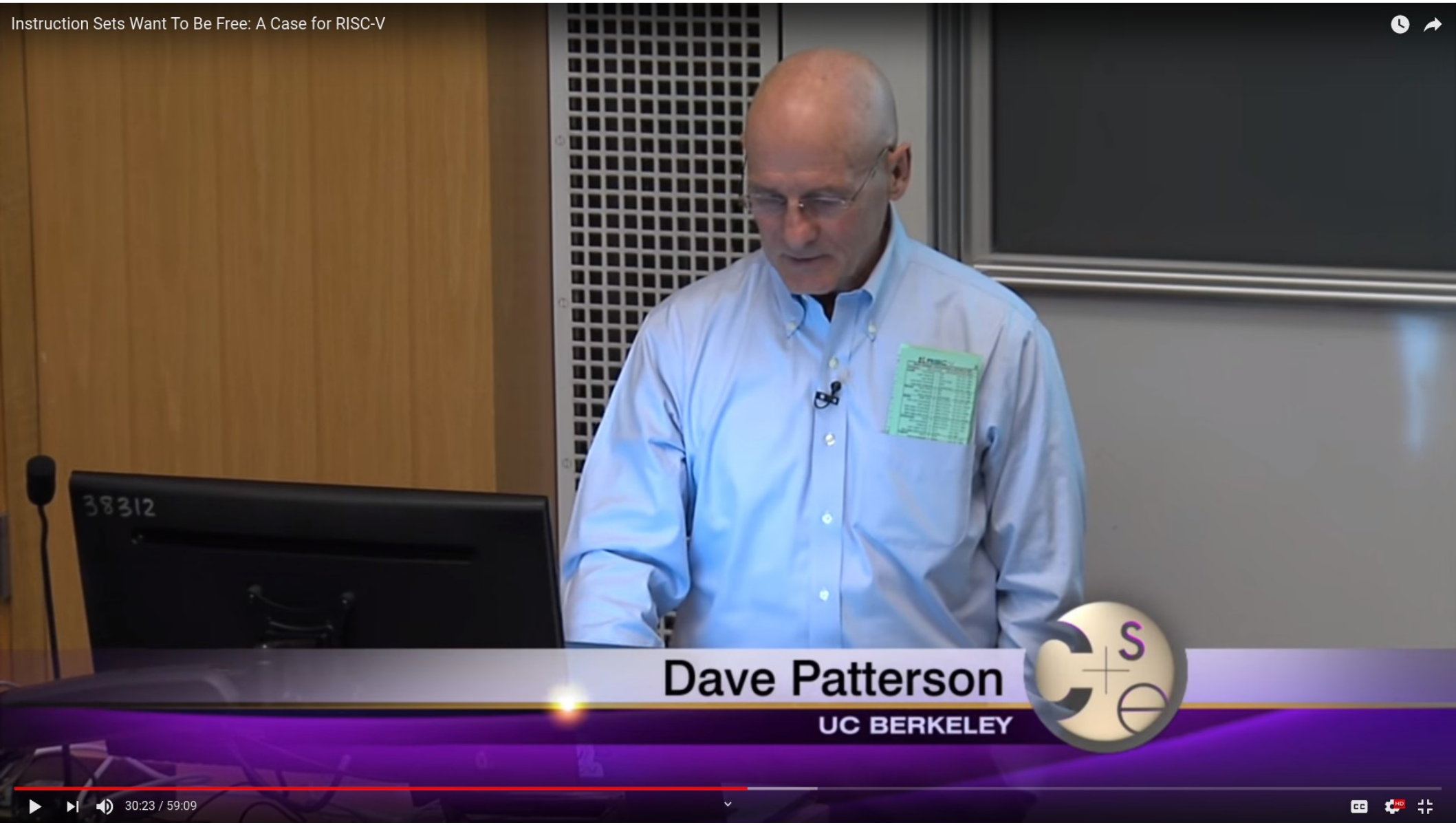
- 32-bit fixed-width, naturally aligned instructions
- 31 integer registers x1-x31, plus x0 zero register
- rd/rs1/rs2 in fixed location, no implicit registers
- Immediate field (instr[31]) always sign-extended
- Floating-point adds f0-f31 registers plus FP CSR, also fused mul-add four-register format
- Designed to support PIC and dynamic linking

43

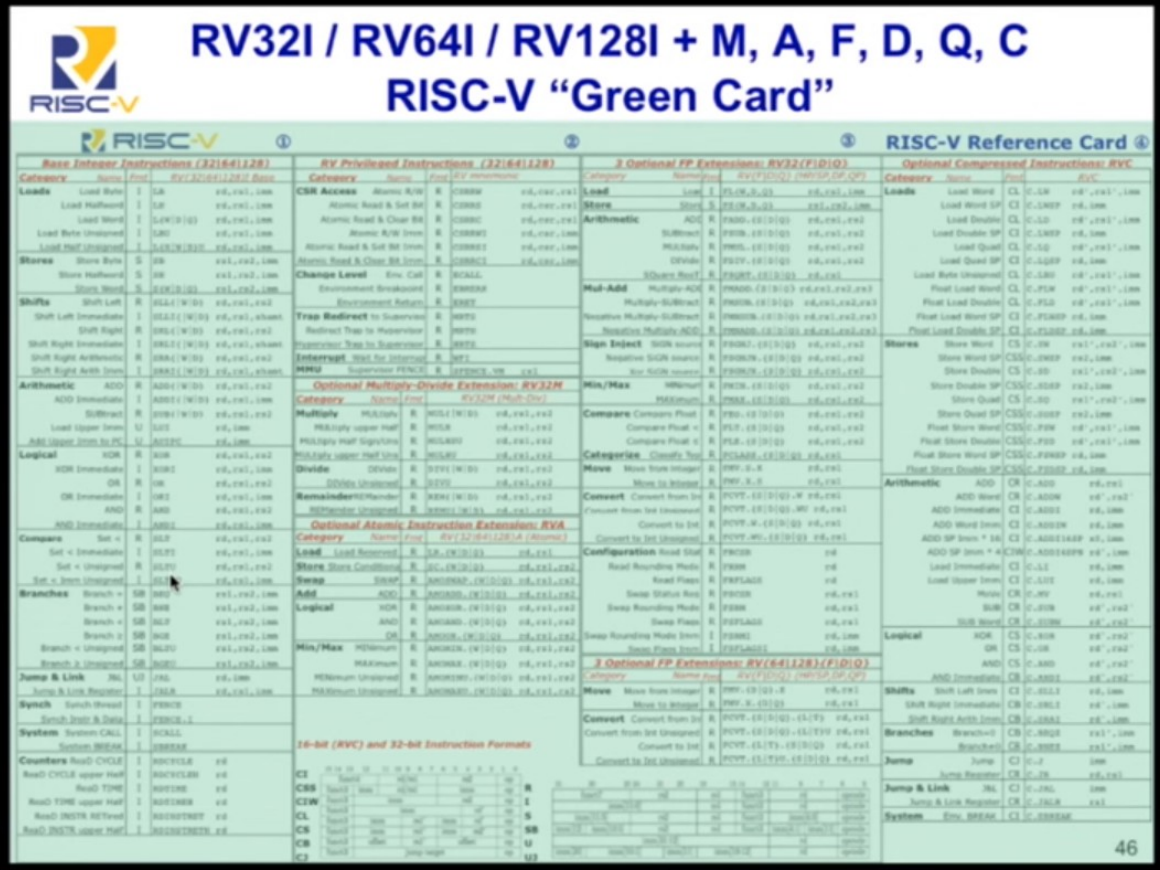
Instruction Sets Want To Be Free: A Case for RISC-V



30:33 / 59:09


Instruction Sets Want To Be Free: A Case for RISC-V
38312
Dave Patterson
UC BERKELEY
30:23 / 59:09

Instruction Sets Want To Be Free: A Case for RISC-V



RISC-V and Industry

- Created at UC Berkeley but useful beyond academia
- Designed to be extensible
  - Microcontroller to supercomputer
- RISC-V Foundation now controls standard: riscv.org
  - Over 400 members: companies, universities and more
  - YouTube channel has hundreds of talks!
    - https://www.youtube.com/channel/UC5gLmcFuvdGbajs4VL-WU3g
- Nvidia and Western Digital are now shipping millions of devices with RISC-V processors
  - freedom to leverage open source implementations
    - BOOM, Rocket, PULP, SweRV, and many more
  - avoiding ARM licensing fees

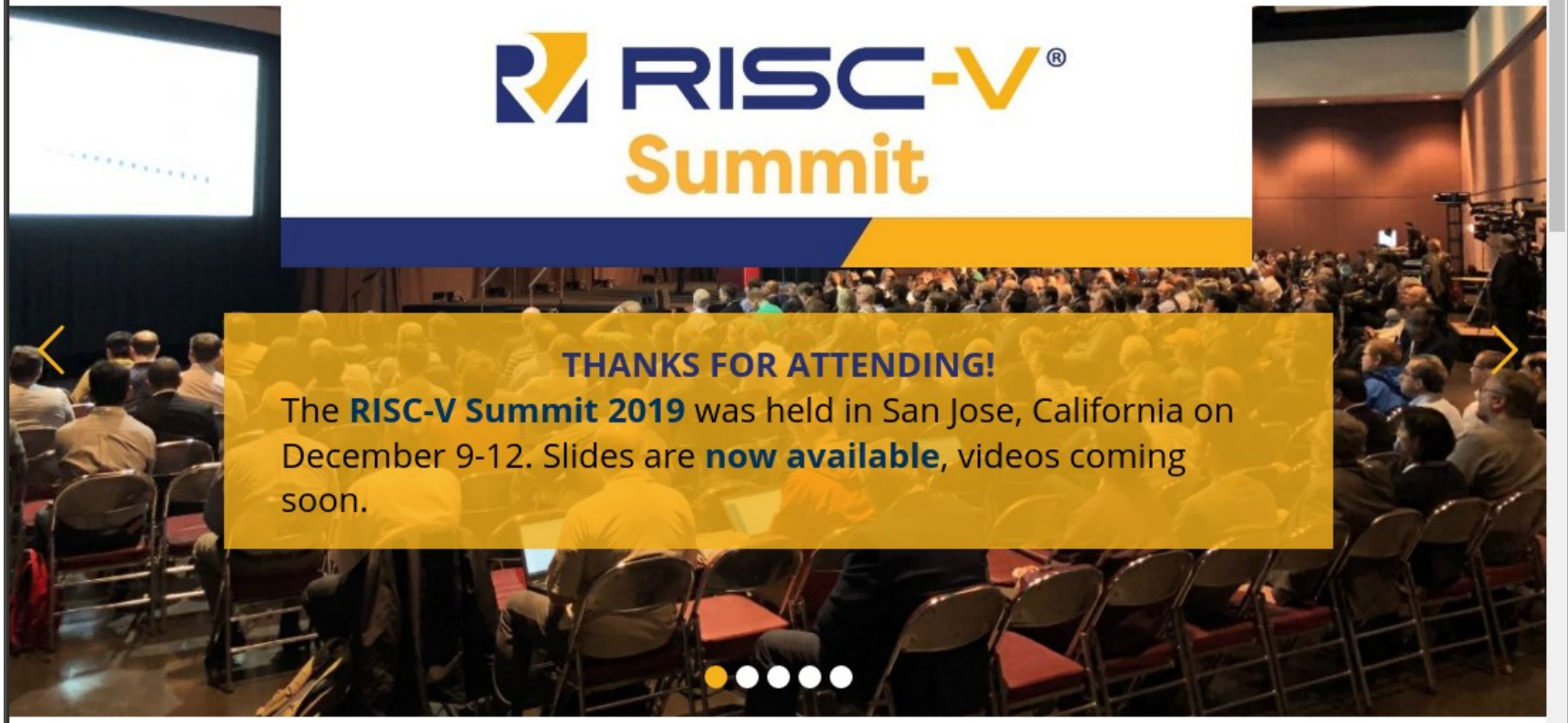
RISC-V Foundation | Instru

riscv.org

Join the Mailing Lists | info@riscv.org | Member Login

RISC-V®

ABOUT | MEMBERSHIP | SPECS & SUPPORT | CORES & TOOLS | NEWS | EVENTS

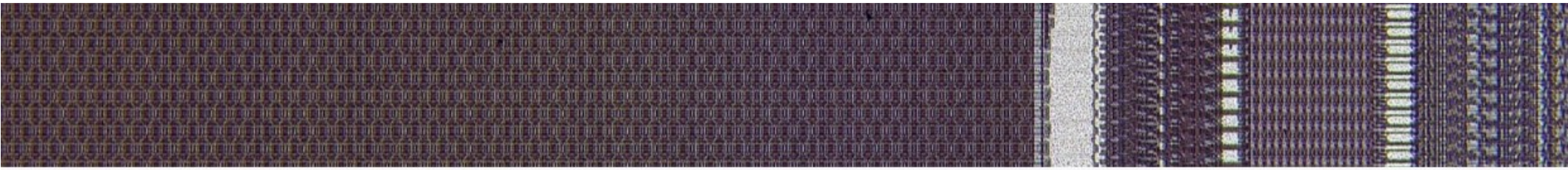


RISC-V: The Free and Open RISC Instruction Set Architecture

RISC-V is a free and open ISA enabling a new era of processor innovation through open standard collaboration. Born in academia and research, RISC-V ISA delivers a new level of free, extensible software and hardware freedom on architecture, paving the way for the next 50 years of computing design and innovation.

NEW TO RISC-V? LEARN MORE

Privacy - Terms



RISC-V

7.86K subscribers

SUBSCRIBED

HOME | VIDEOS | PLAYLISTS | COMMUNITY | CHANNELS | ABOUT

Uploads PLAY ALL

SORT BY

RISC-V Summit 2019: 69
SafeRV Building Blocks for...

150 views • 1 week ago

RISC-V Summit 2019: 61
Andes RISC V Processor...

210 views • 1 week ago

RISC-V Summit 2019: 47
Production ready RISC V...

563 views • 1 week ago

RISC-V Summit 2019: 64 Ara
2 0 64 bit RISC V Vector...

237 views • 1 week ago

RISC-V Summit 2019: 63
Working Towards a Commo...

467 views • 1 week ago

RISC-V Summit 2019: 59
RISC V Processor Verificati...

76 views • 1 week ago

RISC-V Summit 2019: 62
Ruby Sponsor SiFive presen...

36 views • 1 week ago

RISC-V Summit 2019: 66
Rambus presents Challenge...

27 views • 1 week ago

RISC-V Summit 2019: 49
SweRV Cores Roadmap

151 views • 1 week ago

RISC-V Summit 2019: 50
RISC V Enclaves A Clean...

77 views • 1 week ago

RISC-V Summit 2019: 54
RISC V A New Zero Trust...

52 views • 1 week ago

RISC-V Summit 2019: 48
Ruby Sponsor SiFive presen...

122 views • 1 week ago

RISC-V Summit 2019: 51
seL4 on RISC V Verified OS...

108 views • 1 week ago

RISC-V Summit 2019: 60
Headline Sponsor Western...

54 views • 1 week ago

RISC-V Summit 2019: 46
RISC V For Heterogeneous...

54 views • 1 week ago

RISC-V Summit 2019: 58
Innovation in CPU...

180 views • 1 week ago

RISC-V Summit 2019: 53
Integrate RISC V to build...

58 views • 1 week ago

RISC-V Summit 2019: 56
OneSpin presents More tha...

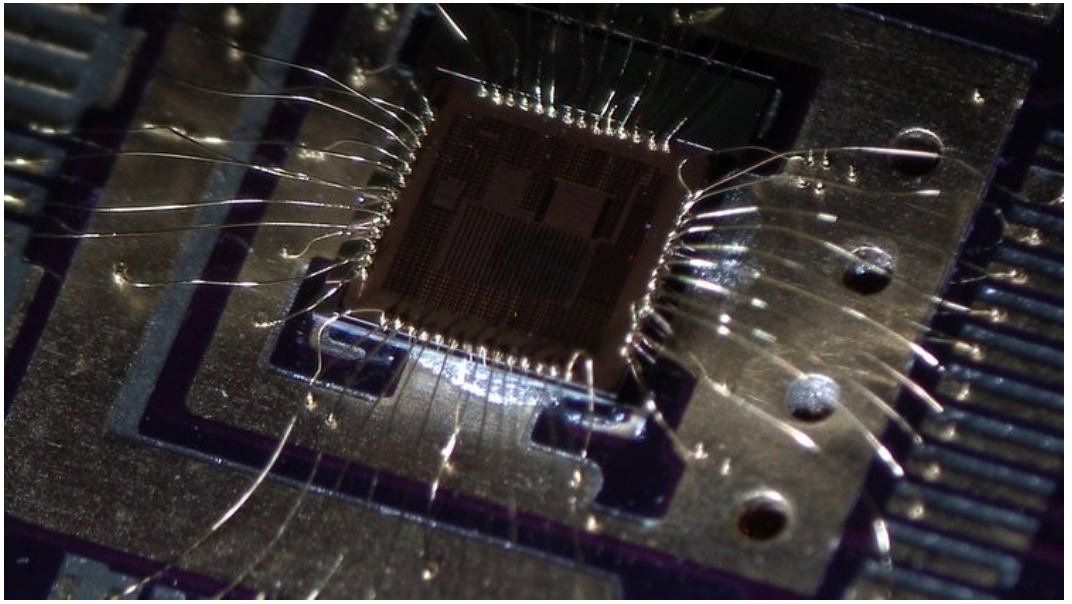
104 views • 1 week ago

RISC-V and the world

- RISC-V Foundation moving from US to Switzerland
- Nations such as India have RISC-V initiatives
  - Desire for sovereign technology and avoid backdoors from other nations
- Strong interest from chipmakers in China
  - U.S. companies have been banned from doing business with Huawei… who's next?
  - ARM deemed UK-origin tech so ok to do business with Huawei, but what will brexit-govt bring?

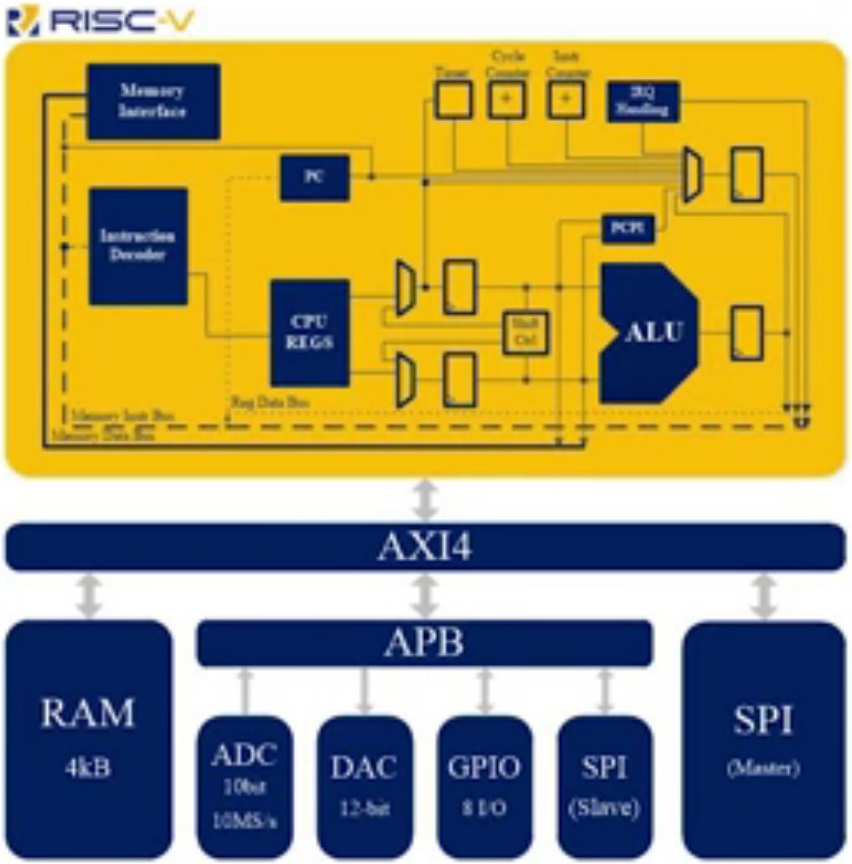
- **OnChip Open-V**

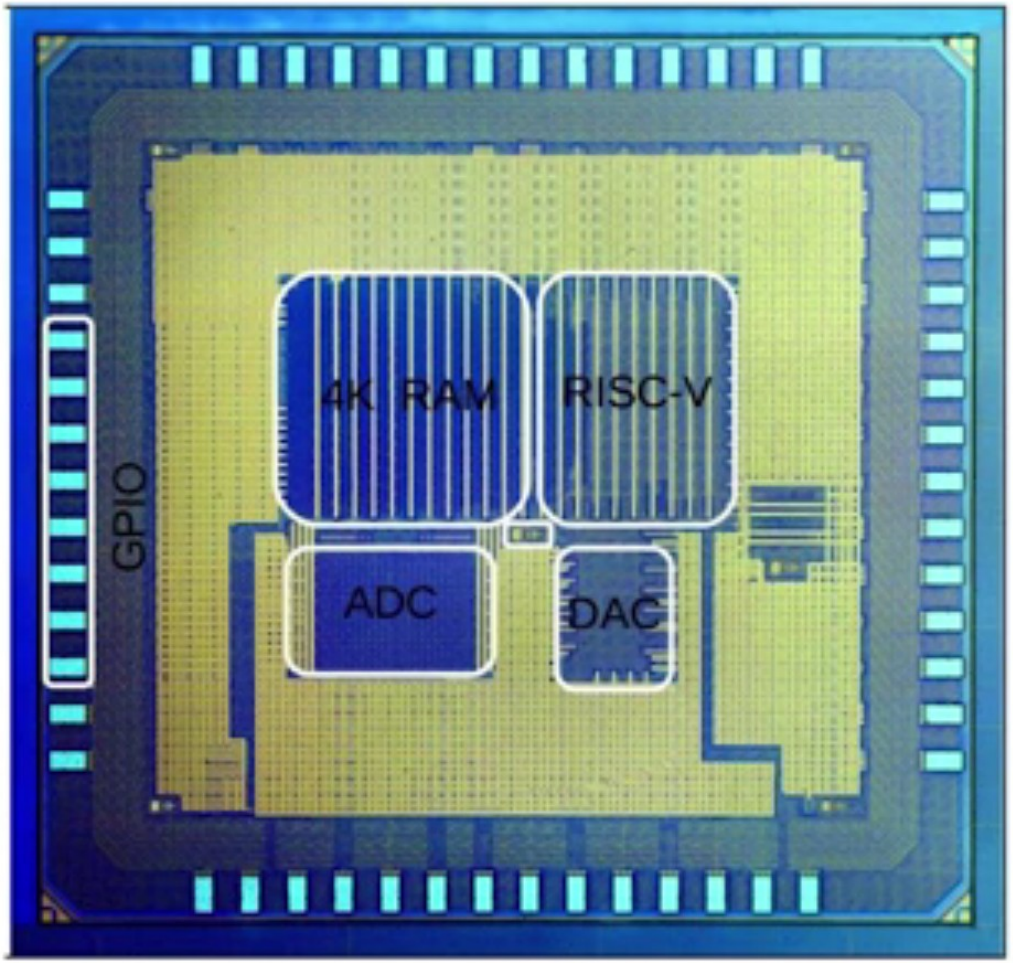
  “completely free (as in freedom) and open source 32-bit microcontroller based on the RISC-V architecture”



OnChip Open-V

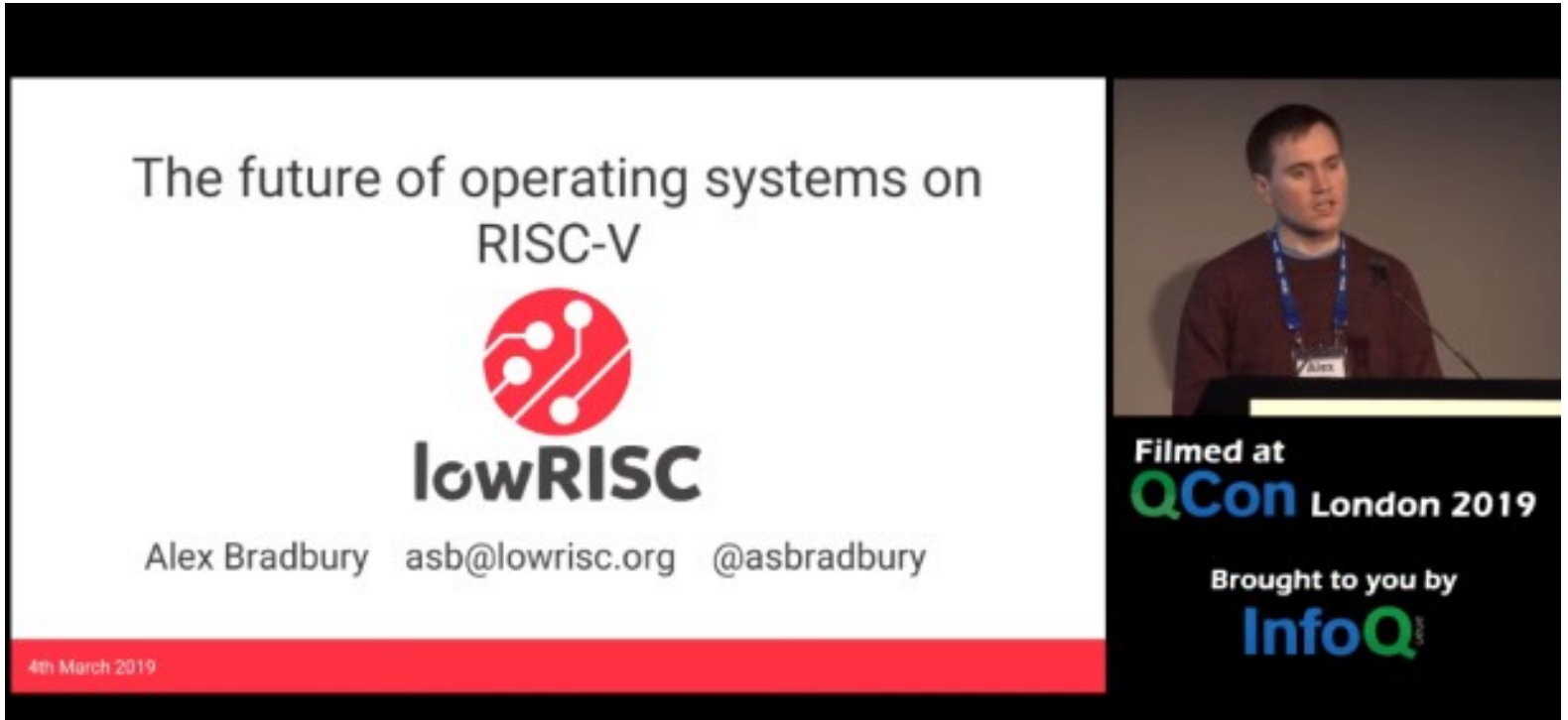
A 32-bit RISC-V based Microcontroller



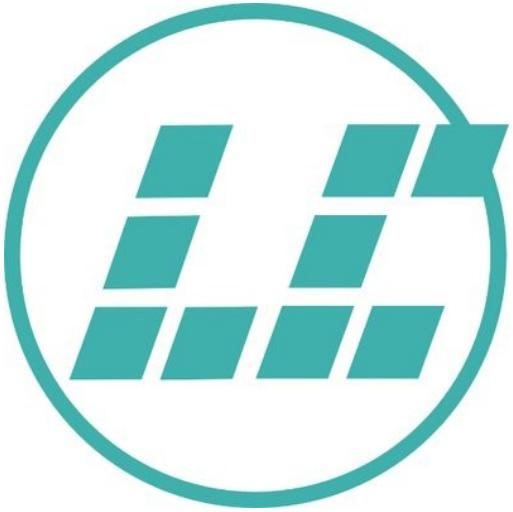


- **lowRISC**: “creating a fully open-sourced, Linux-capable, RISC-V-based SoC, that can be used either directly or as the basis for a custom design”
- Video: Rob Mullins talking about lowRISC
  - (RISC-V & Open Source Silicon Event in Munich on March 23, 2017
- OpenTitan project with Google:
  - Announcing OpenTitan, the First Transparent Silicon Root of Trust

- The Future of Operating Systems on RISC-V
  - Alex Bradbury gives an overview of the status and development of RISC-V as it relates to modern operating systems, highlighting major research strands, controversies, and opportunities to get involved.
  - https://www.youtube.com/watch?v=emnN9p4vhzk

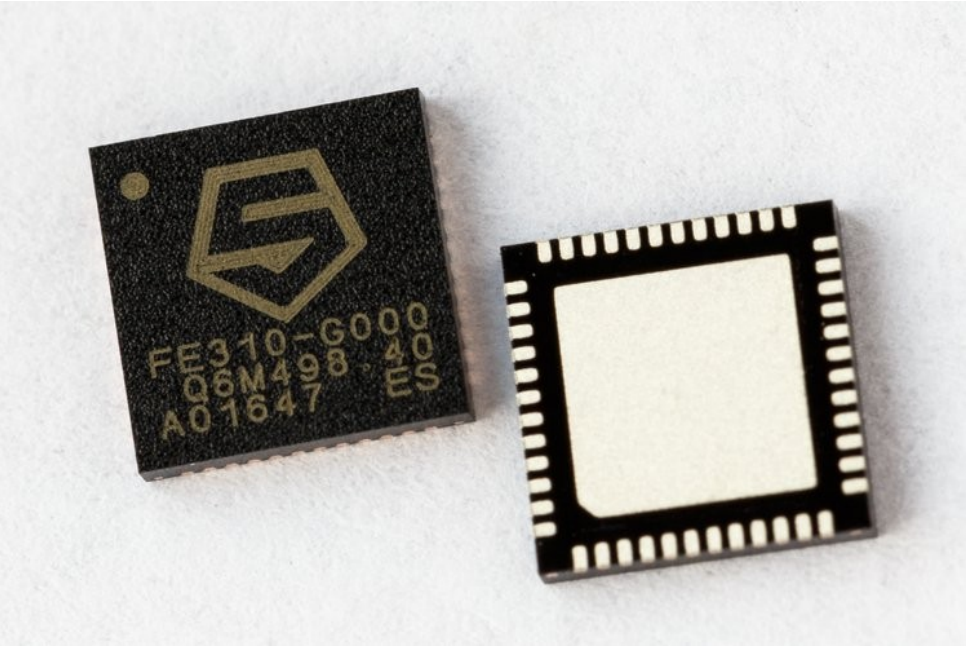


- **FOSSi Foundation**
  - The **F**ree and **O**pen **S**ource **S**ilicon **F**oundation
  - “non-profit foundation with the mission to promote and assist free and open digital hardware designs”
  - Events: ORConf, Latch-up, Week of OSHW
  - **Open Source Silicon Design Ecosystem**
    - Talk by FOSSi co-founder Julius Baxter



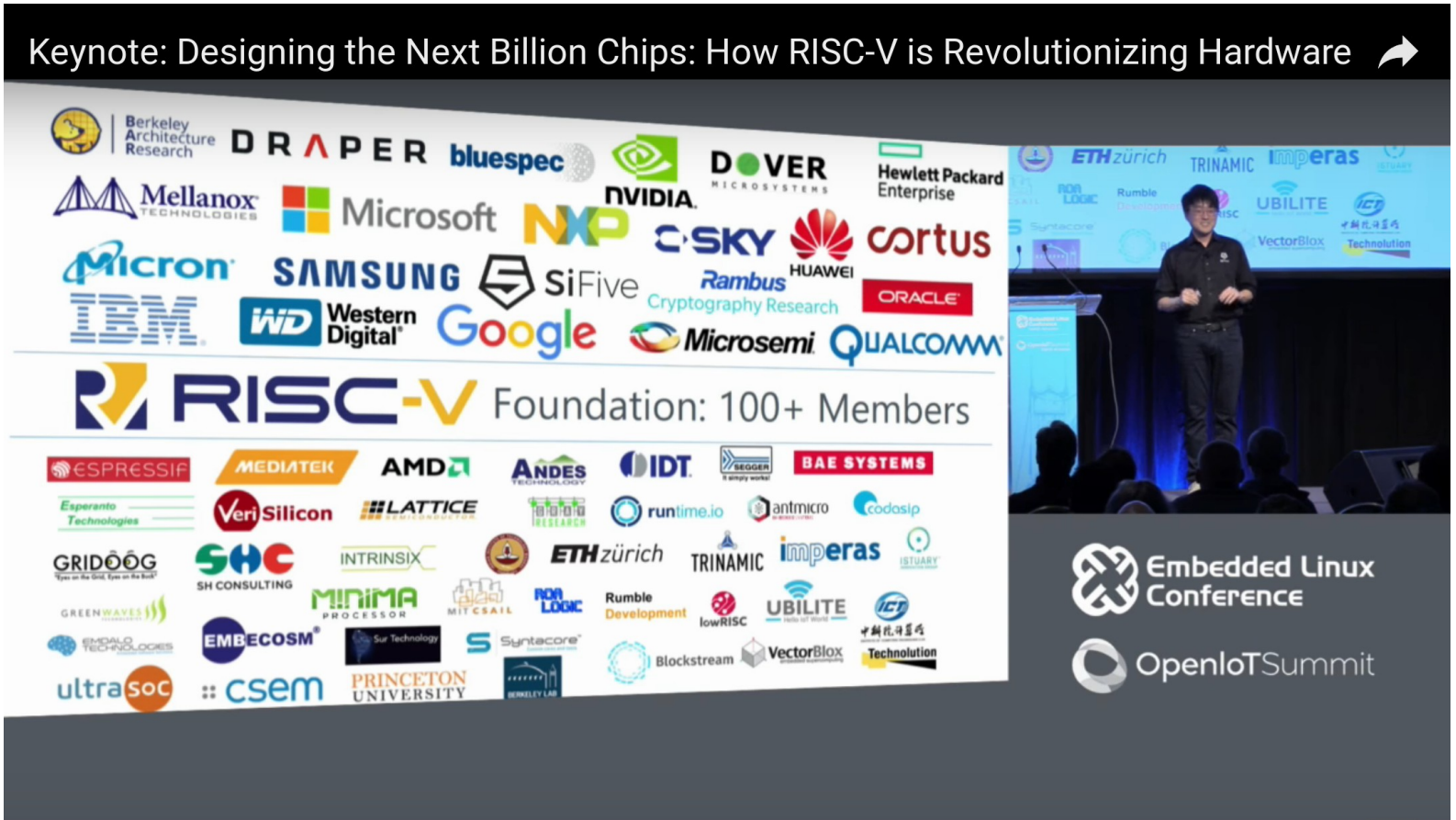
- **LibreCores**
  - Project of the FOSSi Foundation
  - “**gateway to free and open source digital designs** and other components that you can use and **re-use in your digital designs**”
  - “advances the idea of OpenCores.org”

SiFive



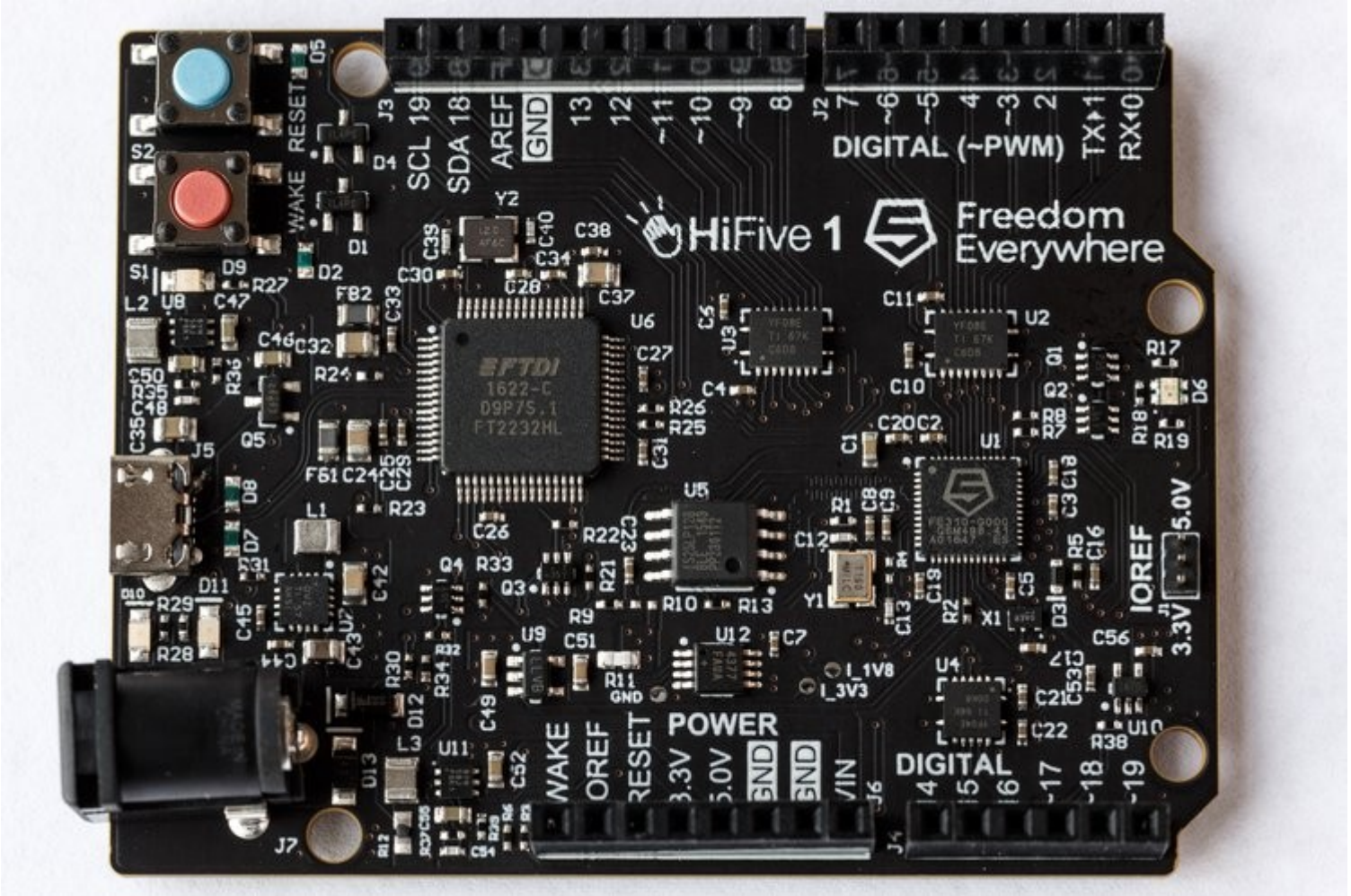
- “founded by the creators of the free and open RISC-V architecture as a reaction to the end of conventional transistor scaling and escalating chip design costs”

- **RISC-V Keynote at Embedded Linux Conf**
  - March 12th, 2018
  - Yunsup Lee, Co-Founder and CTO, SiFive
  - Designing the Next Billion Chips: How RISC-V is Revolutionizing Hardware



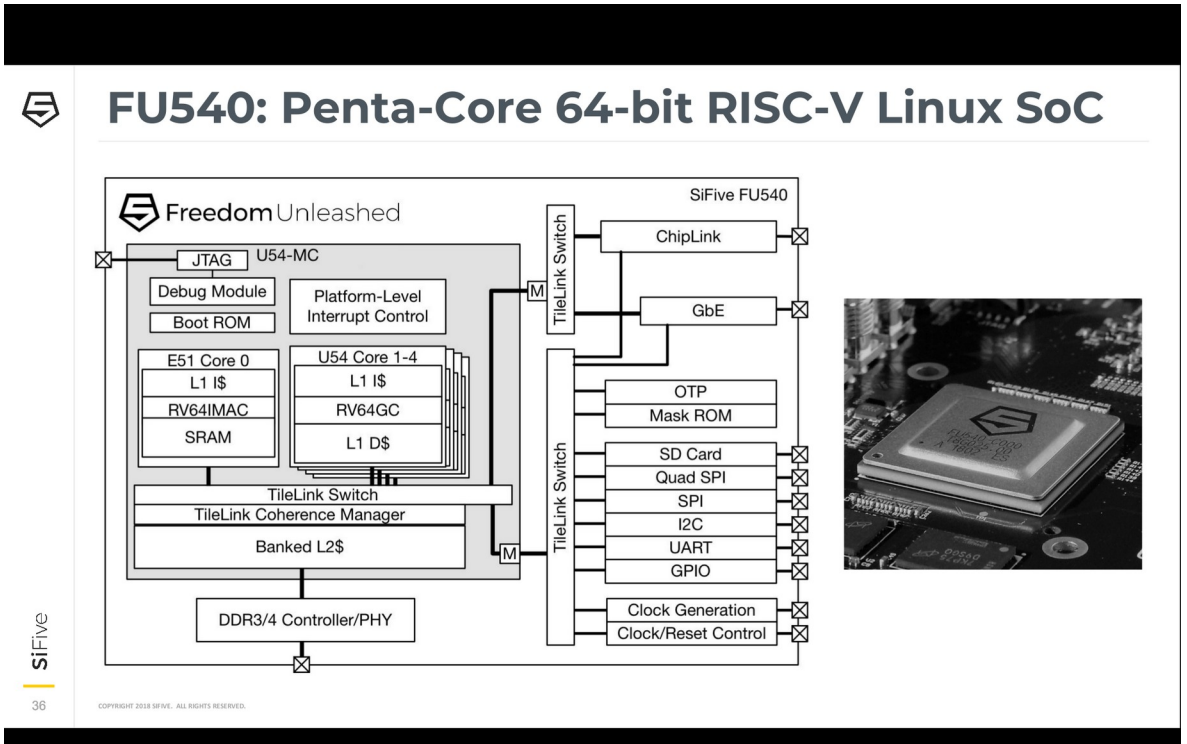
SiFive FE310 microcontroller

- **HiFive1**: Arduino-Compatible RISC-V Dev Kit



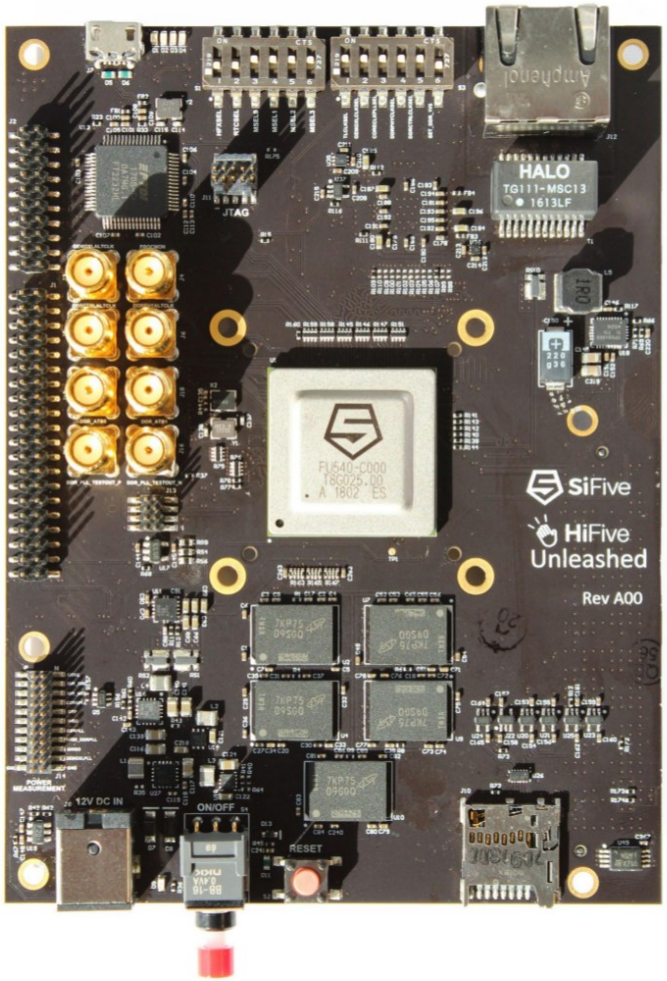
SiFive: Linux on RISC-V

- FOSDEM 2018 talk
  - YouTube: “Igniting the Open Hardware Ecosystem with RISC-V: SiFive's Freedom U500 is the World's First Linux-capable Open Source SoC Platform”
  - Interview with Palmer Dabbelt of SiFive



SiFive: Linux on RISC-V

HiFive Unleashed



- World's First Multi-Core RISC-V Linux Development Board
  - SiFive FU540-C000 (built in 28nm)
    - 4+1 Multi-Core Coherent Configuration, up to 1.5 GHz
    - 4x U54 RV64GC Application Cores with Sv39 Virtual Memory Support
    - 1x E51 RV64IMAC Management Core
    - Coherent 2MB L2 Cache
    - 64-bit DDR4 with ECC
    - 1x Gigabit Ethernet
  - 8 GB 64-bit DDR4 with ECC
  - Gigabit Ethernet Port
  - 32 MB Quad SPI Flash
  - MicroSD card for removable storage
  - FMC connector for future expansion with add-in cards

Western Digital.

RISC-V NOMMU and M-Mode Linux

Damien Le Moal, Western Digital

Linux Plumbers Conference, September 9th, 2019

© 2019 Western Digital Corporation or its affiliates. All rights reserved.

Kendryte K210 SoC + Busybox

Sipeed MAIX Go Board (6+2 MB SRAM)

```
[    0.000000] Linux version 5.1.0-rc5-00314-g375c2321604f (damien@washi) (gcc version 8.2.0 (Buildroot 2018.11-rc2-00003-ga0787e9)) #221 SMP Fri May 10 15:17:17 JST 2019
[    0.000000] earlycon: sbi0 at I/O port 0x0 (options '')
[    0.000000] printk: bootconsole [sbi0] enabled
[    0.000000] initrd not found or empty - disabling initrd
[    0.000000] Zone ranges:
[    0.000000]   DMA32    [mem 0x0000000080000000-0x00000000807fffff]
[    0.000000]   Normal   empty
[    0.000000] Movable zone start for each node
[    0.000000] Early memory node ranges
[    0.000000]   node   0: [mem 0x0000000080000000-0x00000000807fffff]
[    0.000000] Initmem setup node 0 [mem 0x0000000080000000-0x00000000807fffff]
[    0.000000] elf_hwcap is 0x112d
[...]
[    0.000000] Built 1 zonelists, mobility grouping off.  Total pages: 2020
[    0.000000] Kernel command line: console=hvc0 earlycon=sbi init=/bin/bash
[    0.000000] Dentry cache hash table entries: 1024 (order: 1, 8192 bytes)
[    0.000000] Inode-cache hash table entries: 512 (order: 0, 4096 bytes)
[    0.000000] Sorting __ex_table...
[    0.000000] Memory: 6284K/8192K available (920K kernel code, 101K rwdata, 158K rodata, 393K init, 95K bss, 1908K reserved, 0K cma-reserved)
[    0.000000] SLUB: HWalign=64, Order=0-3, MinObjects=0, CPUs=2, Nodes=1
[    0.000000] rcu: Hierarchical RCU implementation.
[    0.000000] rcu: RCU calculated value of scheduler-enlistment delay is 25 jiffies.
[    0.000000] NR_IRQS: 0, nr_irqs: 0, preallocated irqs: 0
[...]
[    0.251433] Freeing unused kernel memory: 392K
[    0.254361] This architecture does not have kernel memory protection.
[    0.259473] Run /bin/bash as init process

BusyBox v1.30.1 (2019-05-10 14:49:46 JST) hush - the humble shell

# mount -t proc none /proc
# cat /proc/cpuinfo
processor : 0
hart : 0
isa : rv64imafdc

processor : 1
hart : 1
isa : rv64imafdc
```

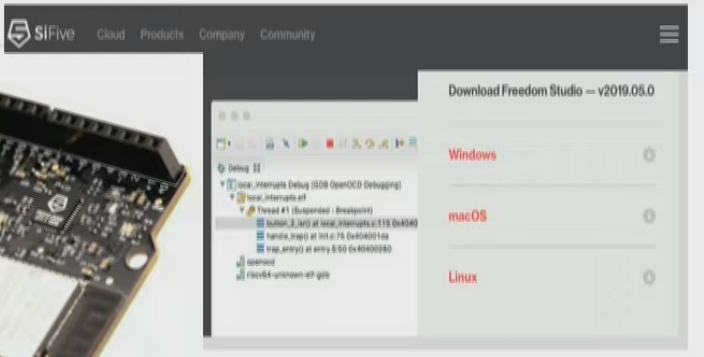

© 2019 Western Digital Corporation or its affiliates. All rights reserved.

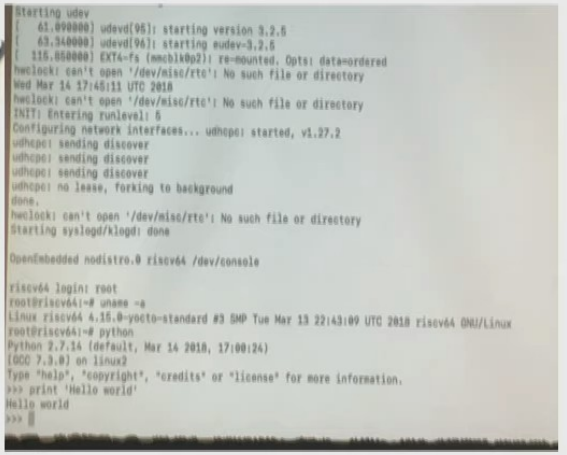
- Experiment to get Linux on the low cost Kendryte K210 RISC-V microcontroller
- PDF: RISC-V NOMMU and M-mode Linux
- https://www.youtube.com/watch?v=ycG592N9EMA&t=10394
- jump to 2h 53m
- Many RISC-V Improvements Ready For Linux 5.5: M-Mode, SECCOMP, Other Features

- Great talk with overview of bootloader, Linux kernel, distro support
  - HOT CHIPS 2019: Linux RISC-V tutorial
  - https://youtu.be/nPXdbm9lc3A?t=6139
  - 1 hour 42 minutes
  - “Overview of RISC-V SW Ecosystem”
    - Bunnaroath Sou, SiFive

Getting Started

- **Freedom Studio from SiFive.com**
  - or Freedom E SDK & Metal
- **RISC-V on Qemu**
  - https://risc-v-getting-started-guide.readthedocs.io/en/latest/index.html
- **sw-dev@groups.riscv.org**
- **#riscv on freenode**





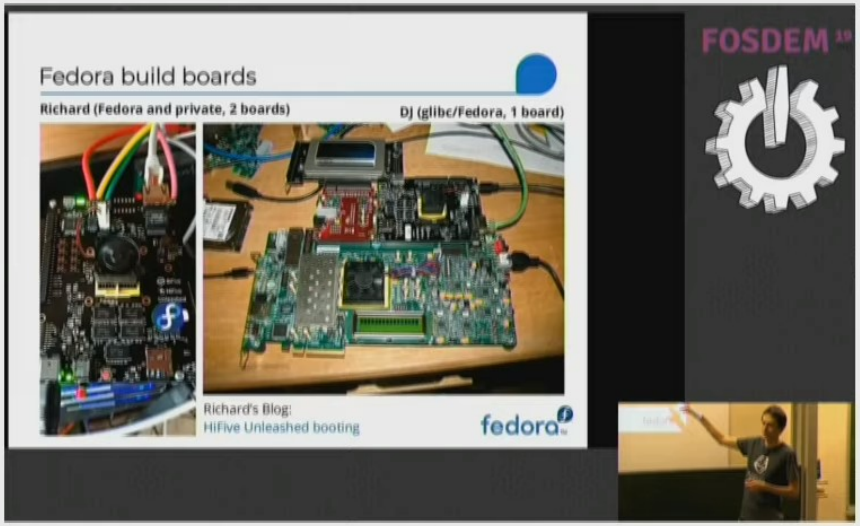
To download and build QEMU from git:

```
git clone https://git.qemu.org/git/qemu.git
cd qemu
git submodule init
./configure --target-list=riscv64-softmmu
make
```

Fedora on RISC-V

Through the works of David Abdurachmanov

- ~20% of Fedora packages built for RISC-V
- Pre-build images available for Qemu and HiFive Unleashed
  - Build farm running, producing nightly images
- No signed RPM yet
- No images for Fedora Workstation/Server



Want to know more/get involve, see

- https://fedoraproject.org/wiki/Architectures/RISC-V

Interest to try it out,

- Self Hosting images available
- https://fedoraproject.org/wiki/Architectures/RISC-V/Installing

1:43:07 / 2:56:38

RISC-V Linux Early Boot

Linux boots expects the system to be in the following state

- a0 contains a unique per-hart id
- a1 contains a pointer to device tree, as a binary flattened device tree (DTB)
- Memory is identity mapped
- The kernel's ELF image has been loaded correctly
- Handle impedance mismatch between RISC-V spec and what Linux expects
- Perform "hart lottery," which is a very short AMO-based sequence that picks the first hart to boot, while the rest spin, until they can proceed

Proceed with a fairly standard Linux early boot process:

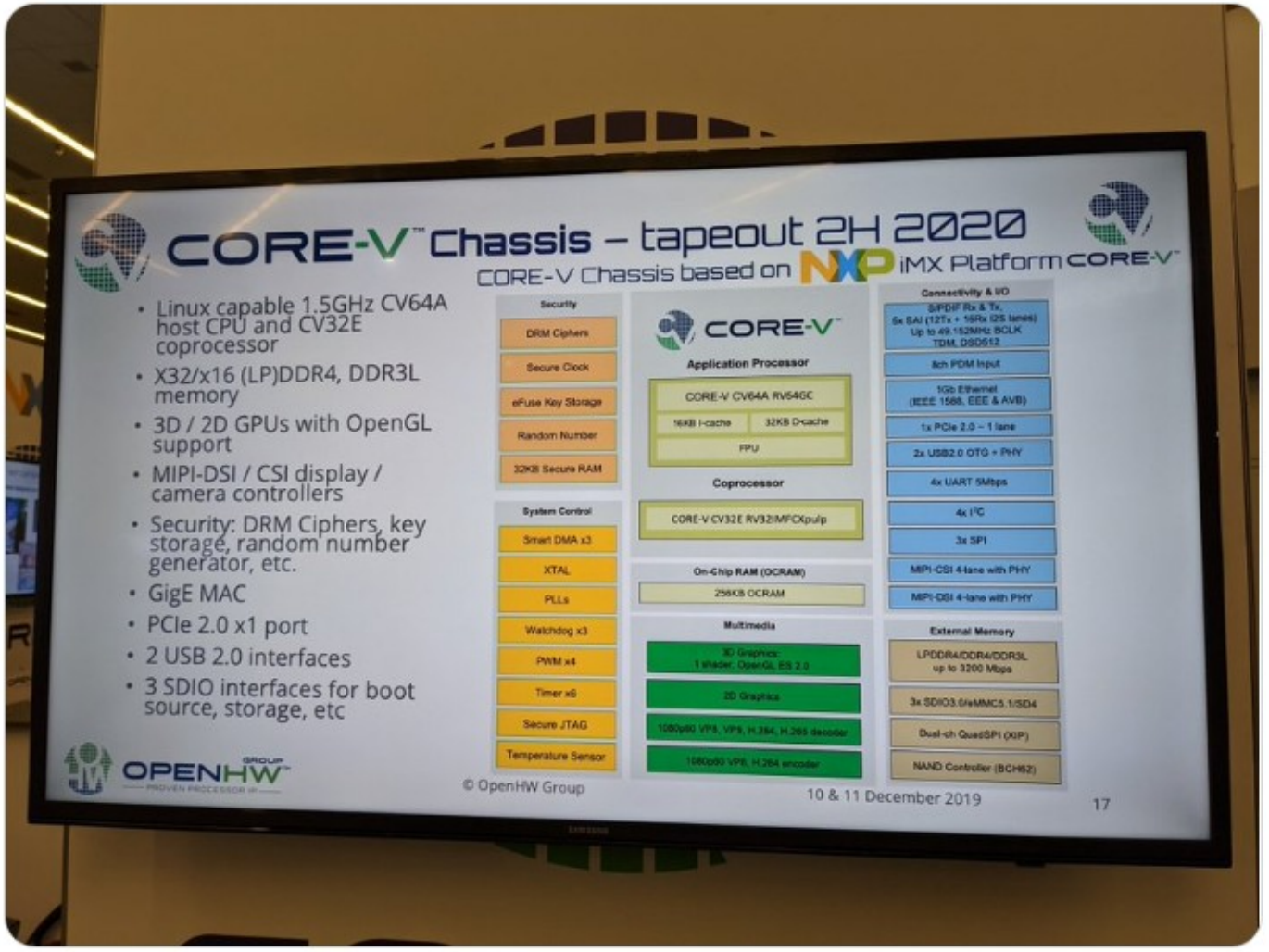
- A linear mapping of all physical memory is set up, with PAGE_OFFSET as the offset
- Paging structures are initialized and then used (BBL boots with paging enabled)
- The C runtime is set up, which includes the stack and global pointers
- A spin-only trap vector is set up that catches any errors early in the boot process
- start_kernel is called to enter the standard Linux boot process

Coming in 2020?

- Andes 27-series CPU
  - “32-bit A27 and 64-bit AX27 and NX27V cores, which will enter production in Q1 2020.”
  - Andes’ RISC-V SoC debuts with AI-ready VPU as Microchip opens access to its PolarFire SoC
- Microchip PolarFire SoC FPGA
  - Hard RISC-V with FPGA fabric… like the Xilinx Zync for ARM
- NXP iMX with RISC-V instead of ARM!
  - “OpenHW Group Unveils CORE-V Chassis SoC Project, Building on PULP Project IP”

Karim Yaghmour
@karimyaghmour

Spotted at RISC V summit: an iMX chip where the ARM core was ripped out and replaced with a RISC V/PULP -- just wow

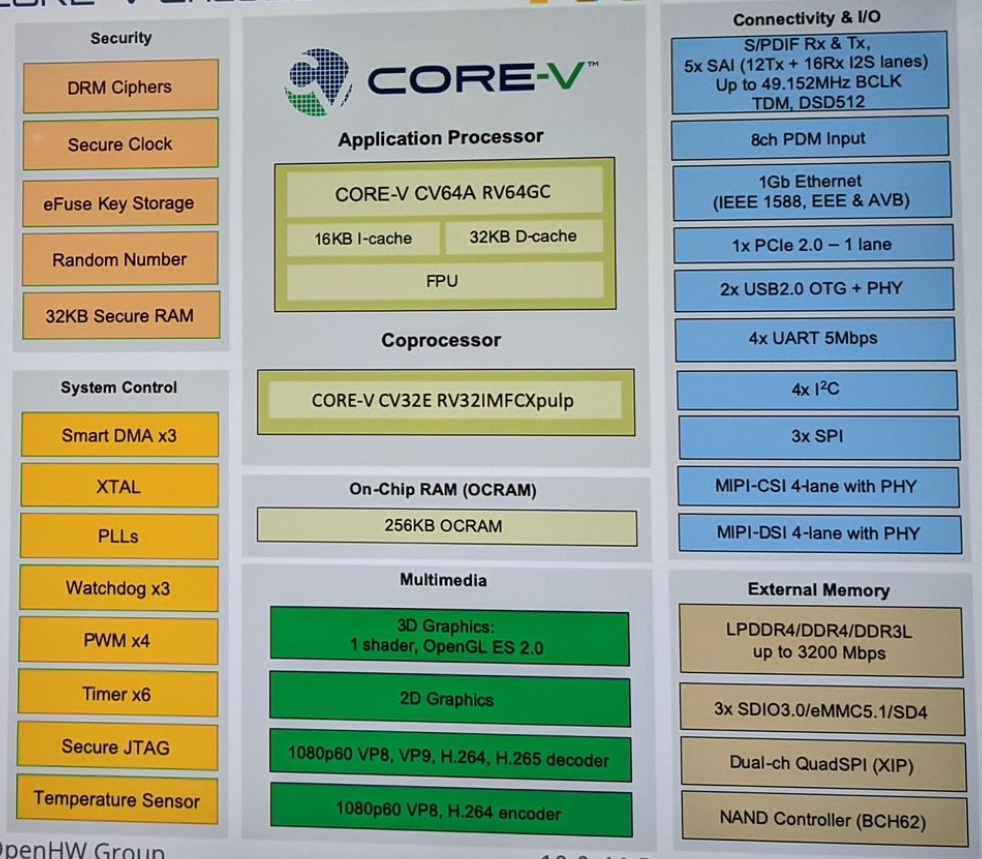


10:53 PM · Dec 10, 2019 · Twitter for Android

CORE-V™ Chassis – tapeout 2H 2020

CORE-V Chassis based on NXP iMX Platform CORE-V™

- Linux capable 1.5GHz CV64A host CPU and CV32E coprocessor
- X32/x16 (LP)DDR4, DDR3L memory
- 3D / 2D GPUs with OpenGL support
- MIPI-DSI / CSI display / camera controllers
- Security: DRM Ciphers, key storage, random number generator, etc.
- GigE MAC
- PCIe 2.0 x1 port
- 2 USB 2.0 interfaces
- 3 SDIO interfaces for boot source, storage, etc



OPENHW GROUP™ PROVEN PROCESSOR IP

© OpenHW Group

OSHW RISC-V Linux board for less than $100?

- **Goal: Sub-$100 Open Source Hardware board that can run Linux on RISC-V**
- Possible by 2021?
- Interested in working together?
  - drew@oshpark.com / Twitter: @pdp7
  - create a mailing list?

Slides: https://github.com/pdp7/talks/blob/master/nerp-riscv.pdf

Section:

Open Source FPGA tools

Open Source and FPGAs

- Hackspace Magazine column about how about open source FPGA tools developed by Claire Wolf (oe1cxw), David Shah and others have made FPGAs more accessible than ever before to makers and hackers:
  - hackspace.raspberrypi.org/issues/26/

The rise of the FPGA

Reconfigure your chips to suit your project

Drew Fustini

@pdp7

Drew Fustini is a hardware designer and embedded Linux developer. He is the Vice President of the Open Source Hardware Association, and a board member of the BeagleBoard.org Foundation. Drew designs circuit boards for OSH Park, a PCB manufacturing service, and maintains the Adafruit BeagleBone Python library.

FPGAs have been the talk of the town at many of this year's hacker conferences. But what exactly is an FPGA, and why are they so hot right now?

FPGA stands for Field Programmable Gate Array, a digital logic chip that can be programmed to reconfigure the internal hardware. An FPGA does not run software – it physically changes the configuration of its gate arrays to adapt to the task at hand, making an FPGA an incredibly versatile tool. Need 25 PWM pins for a project? No problem. Want to replicate the functionality of a vintage CPU? Your FPGA has you covered. Not only is an FPGA versatile, but it is also better at handling timing-critical tasks than a microcontroller. You can filter high-speed sensor data before it's read by your processor, or offload repetitive tasks like debouncing buttons, reducing the burden on your microcontroller.

FPGAs are hot right now, but they're not a new technology – they've been used in industry for decades. However, pricey FPGA development boards, and the massive proprietary software suites needed to develop FPGA designs, means that there has historically been limited adoption in the maker community.

When you write an Arduino program, you're using a language called Wiring, a variation of C++. When you press compile, this high-level code is used to generate a binary file that the processor on the Arduino board executes to achieve the purpose of your program. Similarly, we describe the circuits we want in an FPGA using a high-level, text-based format known as a hardware description language (HDL), such as Verilog. The HDL design is then transformed by a synthesis tool into the basic building blocks that exist in the FPGA.

This process is normally done with proprietary software from the FPGA vendor, but these tools can take ages to download and devour disk space. That is, until Project IceStorm, led by Clifford Wolf, enabled a complete open-source workflow for the Lattice iCE40 FPGA.

FPGAs are hot right now but they're not a new technology – they've been used in industry for decades

This opened the door for low-cost, open hardware boards such as myStorm Blackice, TinyFPGA, iCEBreaker, and Fomu, which are great tools for teaching workshops and building projects.

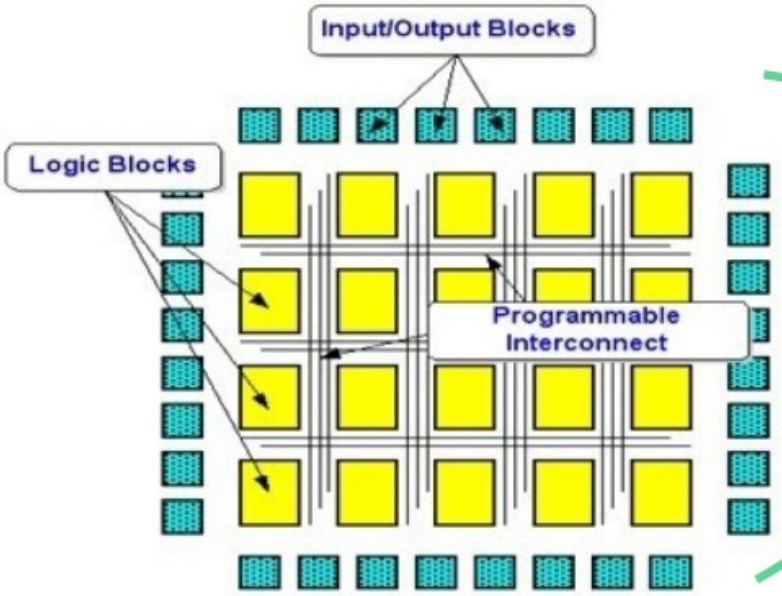
The Lattice ECP5 FPGA is capable of more advanced features than the iCE40, and it now has an open-source workflow too, thanks to Project Trellis led by David Shah. This enabled the ECP5-powered Supercon badge to have cool features like HDMI video, while still being open for anyone to hack on without requiring proprietary tools.

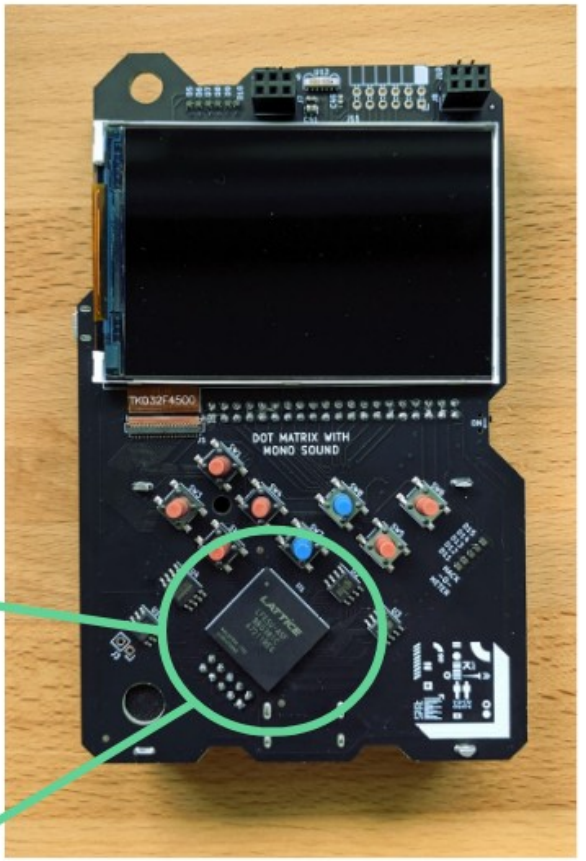
FPGAs are a fascinating technology with lots of awesome applications. If you want to find out more, start off by reading Luke Valenty's *The Hobbyists Guide to FPGAs* on **Hackaday.io** (**hsmag.ce/G0AQwR**), and watch Tim Ansell's Supercon talk to learn about the exciting future of open-source FPGA tools (**hsmag.cc/kyStPD**).

- Keynote at Hackday Supercon 2019 by Dr. Megan Wachs of SiFive
- **“RISC-V and FPGAs: Open Source Hardware Hacking”**
  - https://www.youtube.com/watch?v=vCG5_nxm2G4

Where do FPGAs Come In?

- Field Programmable Gate Array
- Change a chip's HARDWARE in a few minutes
- Make it act like a new chip!





Open Source and FPGAs

- Open Source toolchains for FPGAs!
  - Project IceStorm for Lattice iCE40
    - “A Free and Open Source Verilog-to-Bitstream Flow for iCE40 FPGAs”
      by Claire Wolf (oe1cxw) at 32c3

media.ccc.de

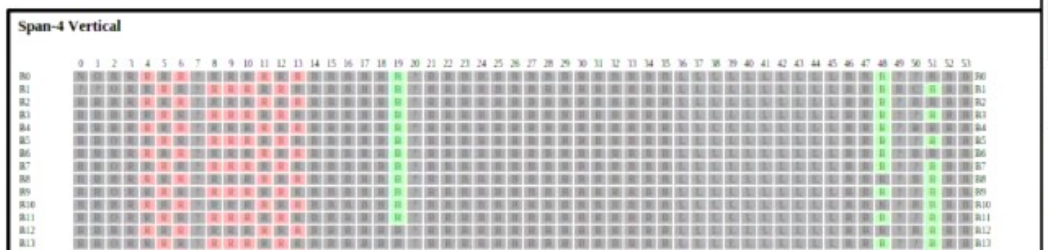
Search...

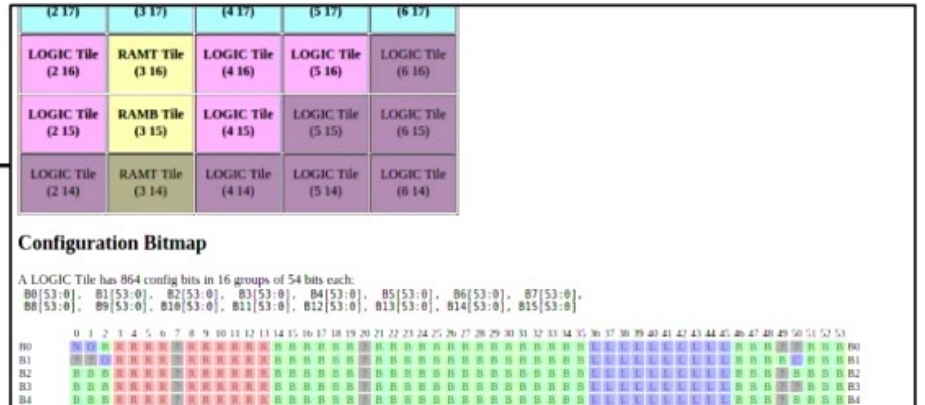
browse › congress › 2015 › event

A Free and Open Source Verilog-to-Bitstream Flow for iCE40 FPGAs

Clifford

Some screenshots from IceStrom Docs:



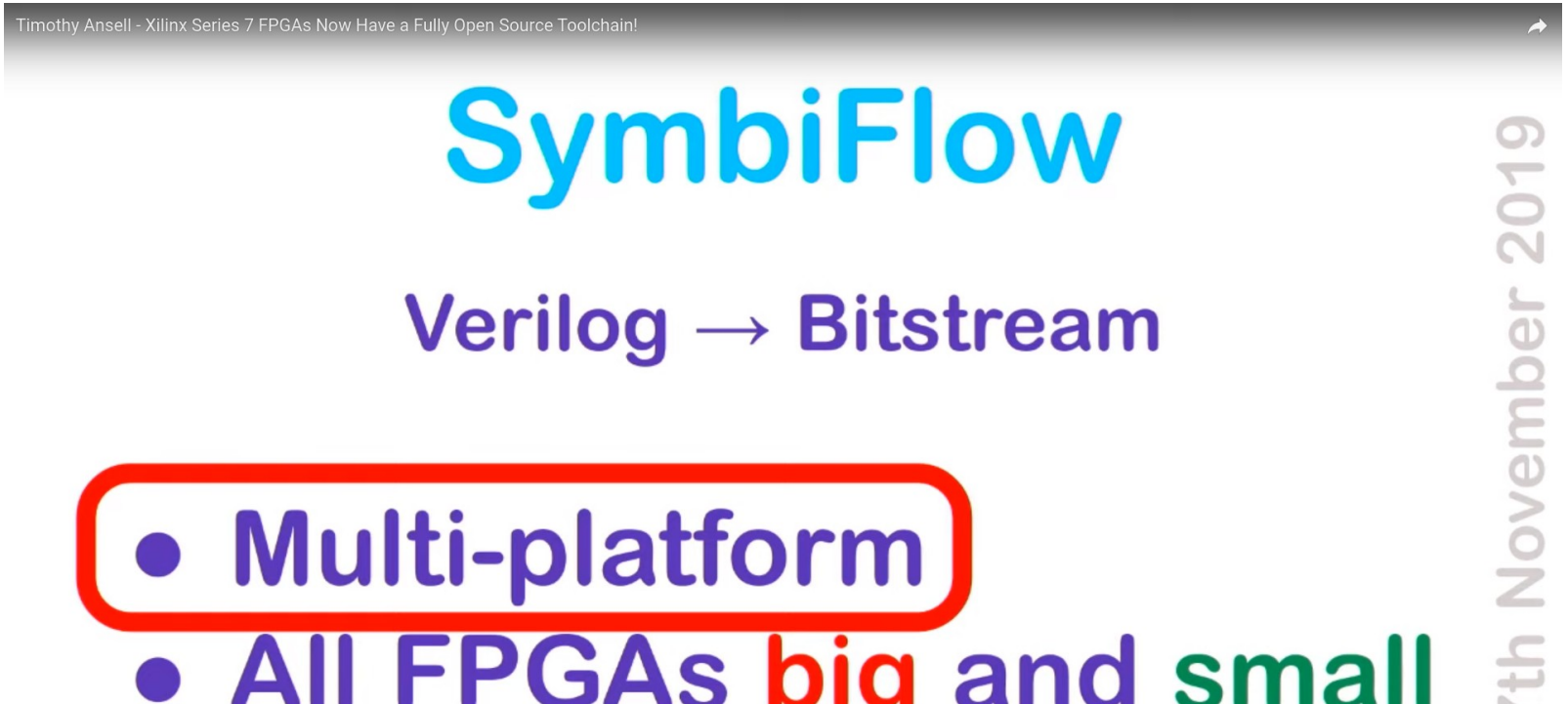


Open Source and FPGAs

- Open Source toolchains for FPGAs!
  - Project Trellis for Lattice ECP5
  - “Project Trellis and nextpnr FOSS FPGA flow for the Lattice ECP5”
    - David Shah (@fpga_dave)
      - youtube.com/watch?v=0se7kNes3EU

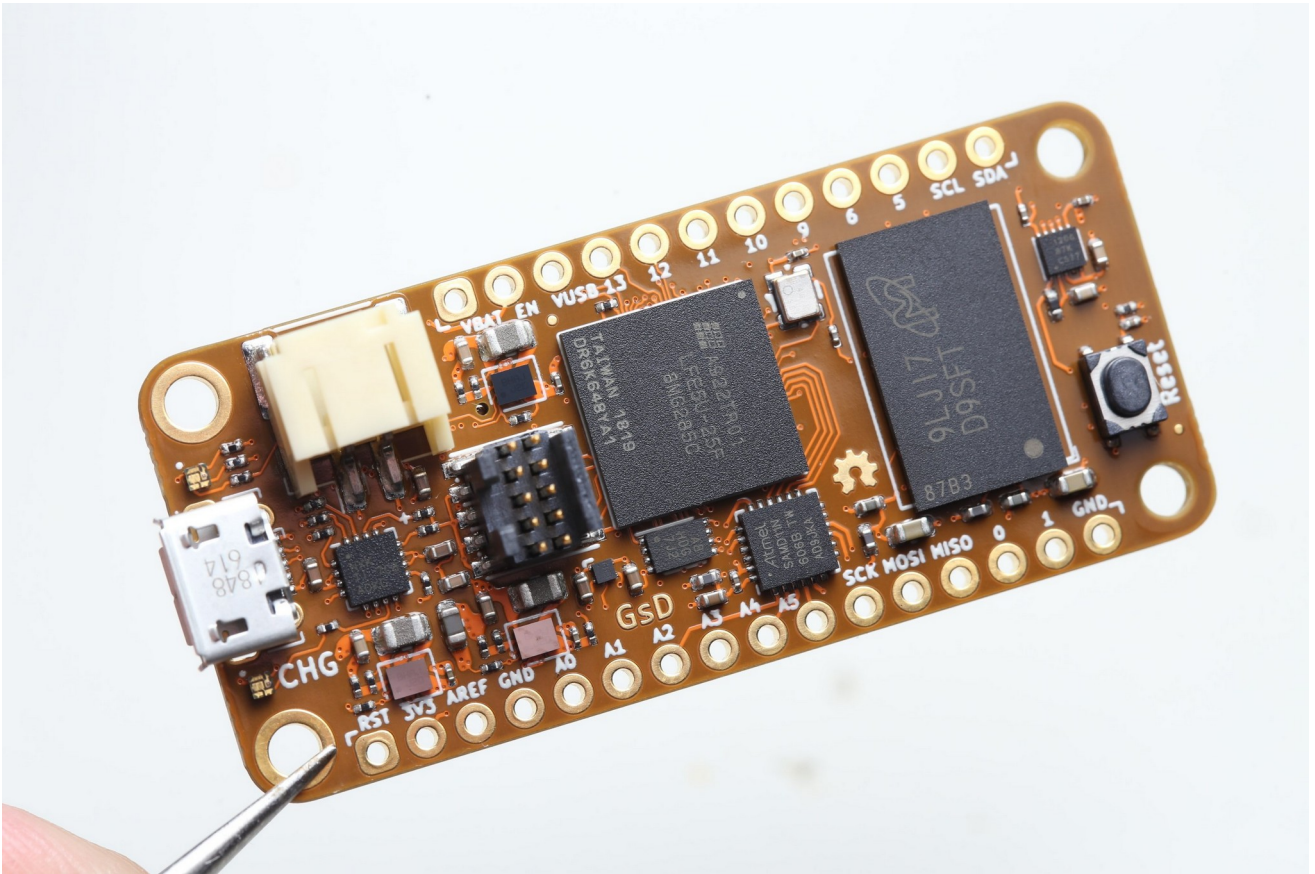
Open Source and FPGAs

- Open Source toolchains for FPGAs!
  - Project X-Ray and SymbiFlow for Xilinix Series 7
  - Timothy ‘mithro’ Ansell: “Xilinx Series 7 FPGAs Now Have a Fully Open Source Toolchain!” *(almost)*
    - youtube.com/watch?v=EHePto95qoE



Open Source and FPGAs

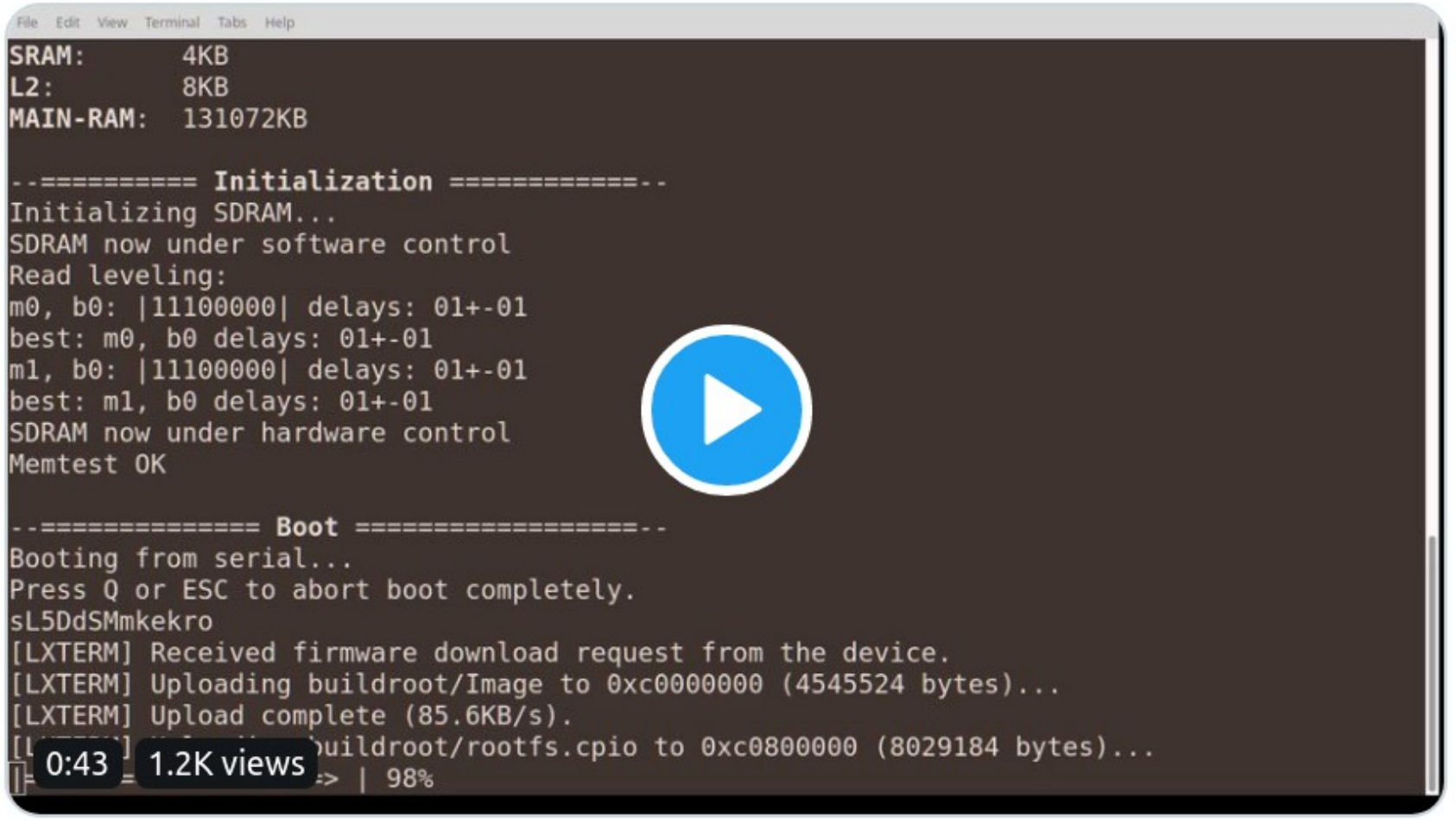
- Open Source Hardware boards with Lattice ECP5 FPGA with open RISC-V “soft” CPU:
  - Orange Crab by Greg Davill
    - https://github.com/gregdavill/OrangeCrab



Greg @ #36c3
@GregDavill

Replying to @mithro @pdp7 and 2 others

Done. 😃



4:55 PM · Dec 28, 2019 · Twitter Web App

Open Source and FPGAs

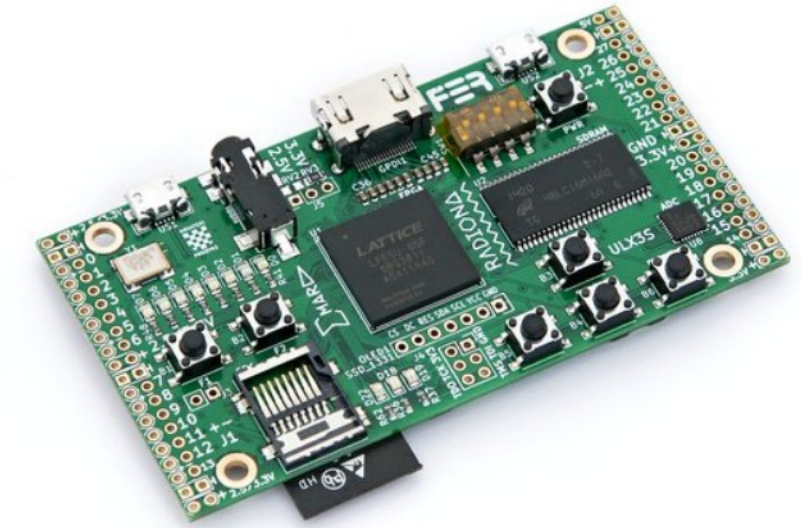
- Radiona.org ULX3S
  - https://www.crowdsupply.com/radiona/ulx3s

ULX3S

A powerful, open hardware ECP5 FPGA dev board

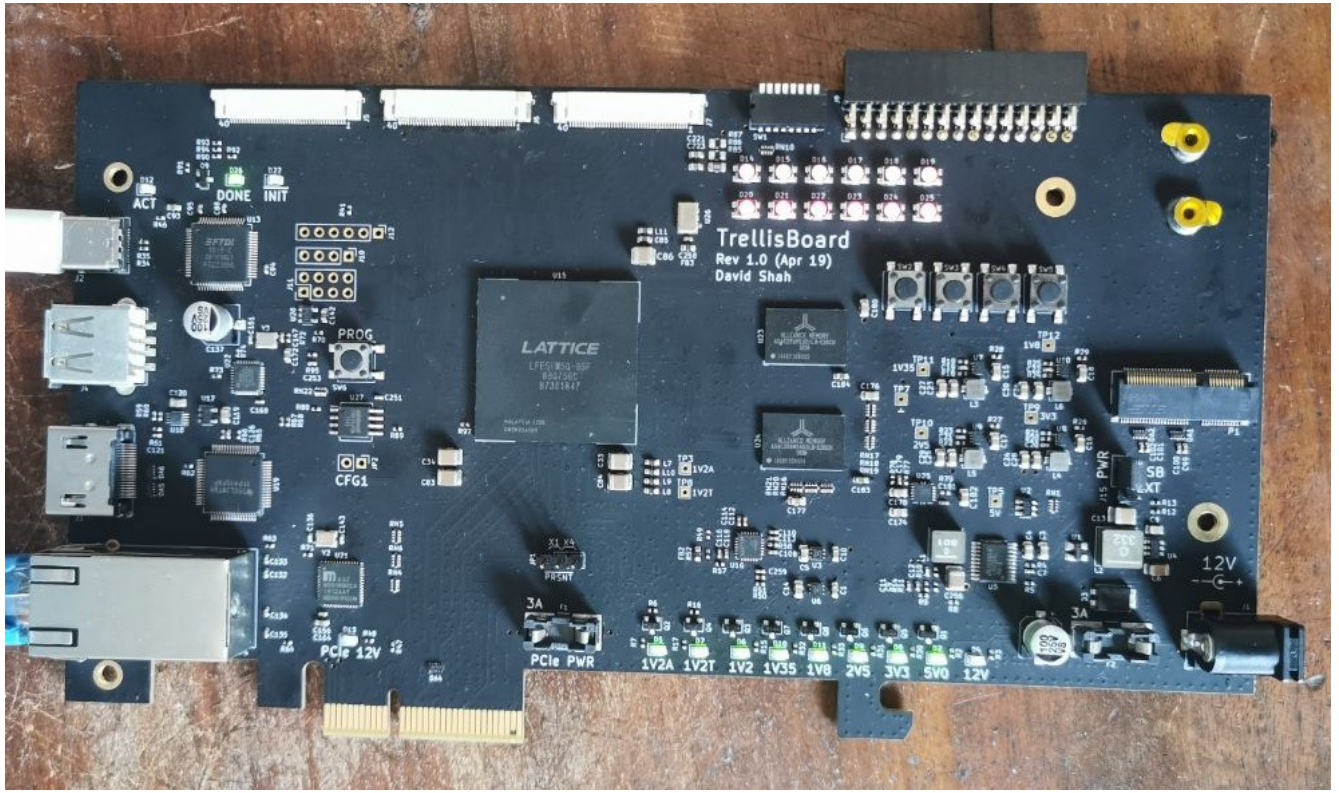
This project is coming soon. Sign up to receive updates and be notified when this project launches.

me@example.com Subscribe



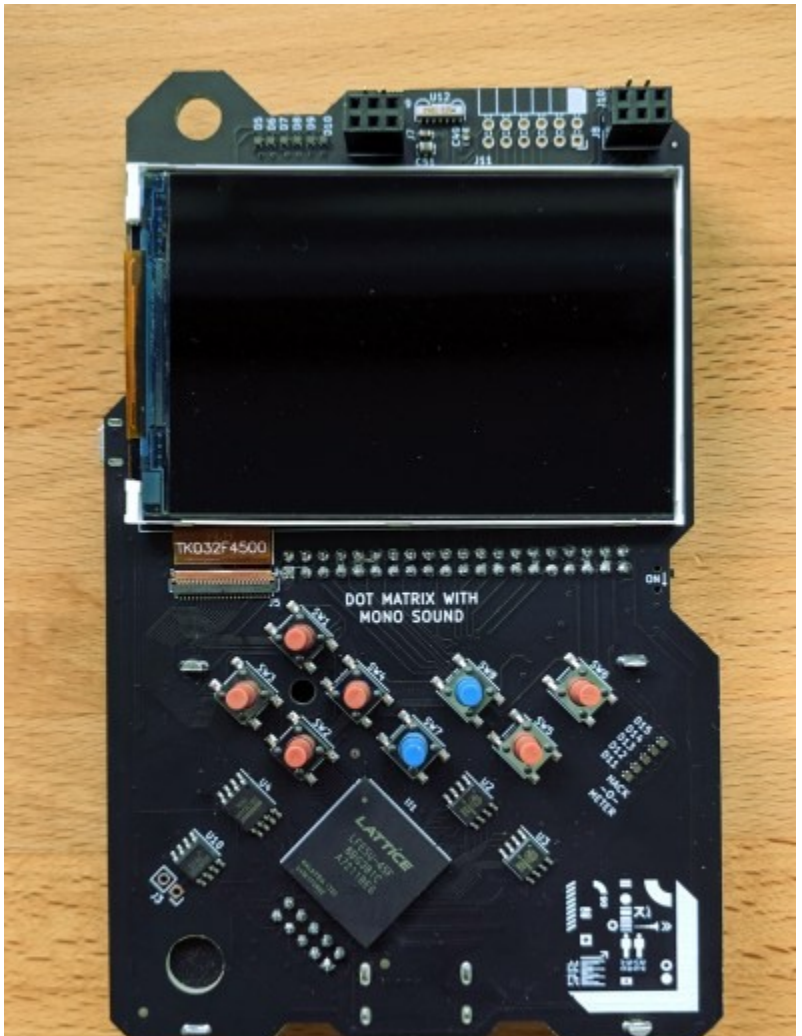
Open Source and FPGAs

- Open Source Hardware boards with Lattice ECP5 FPGA with open RISC-V “soft” CPU:
  - David Shah's Trellis board (Ultimate ECP5 Board)
  - https://github.com/daveshah1/TrellisBoard



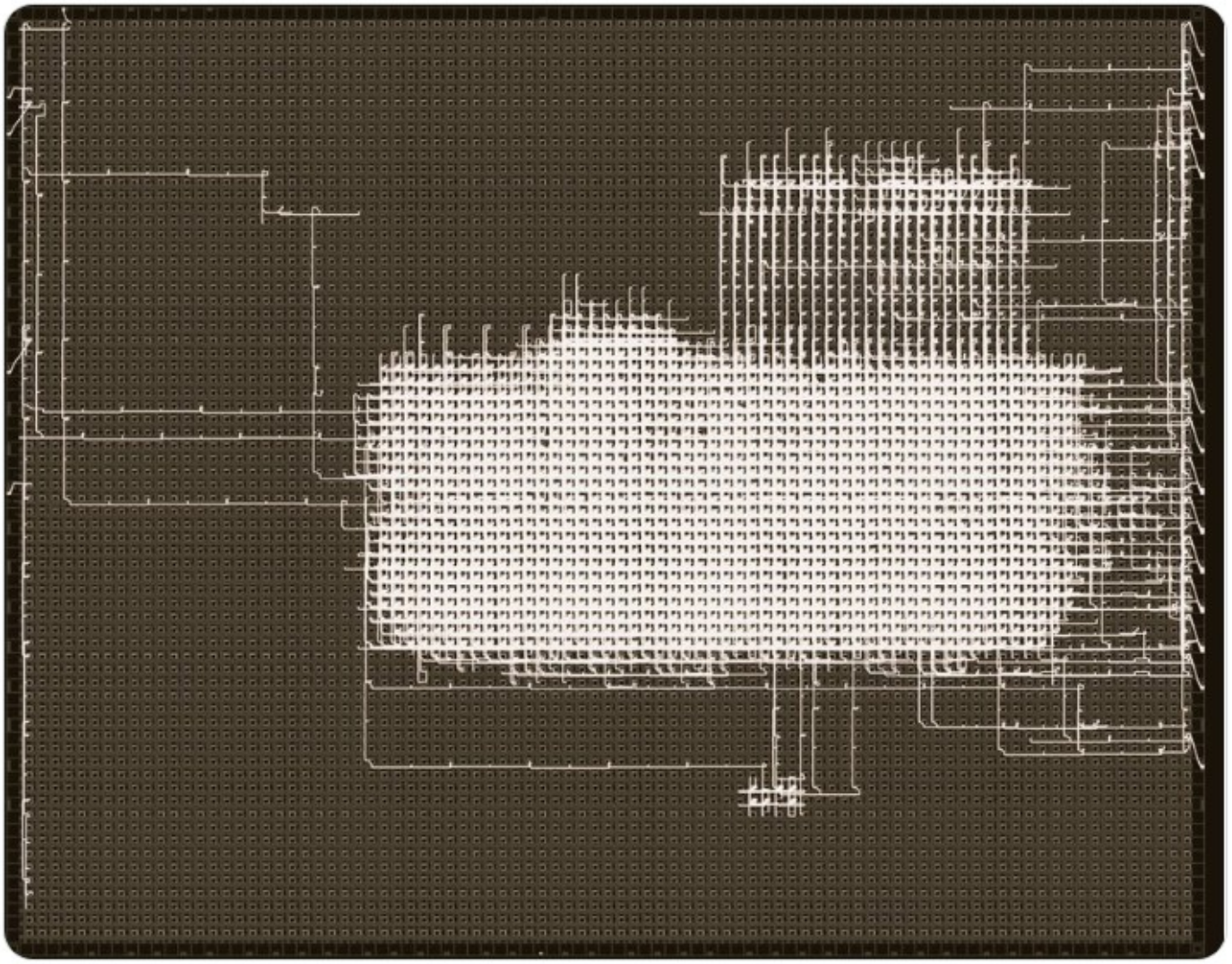
Hackaday 2019 Supercon badge

- RISC-V “soft” core on ECP5 FPGA
- Gigantic FPGA In A Game Boy Form Factor



OSS FPGA & EDA tools
@ico_TC

This is how a Linux capable core looks like on an FPGA.#nextpnratwork



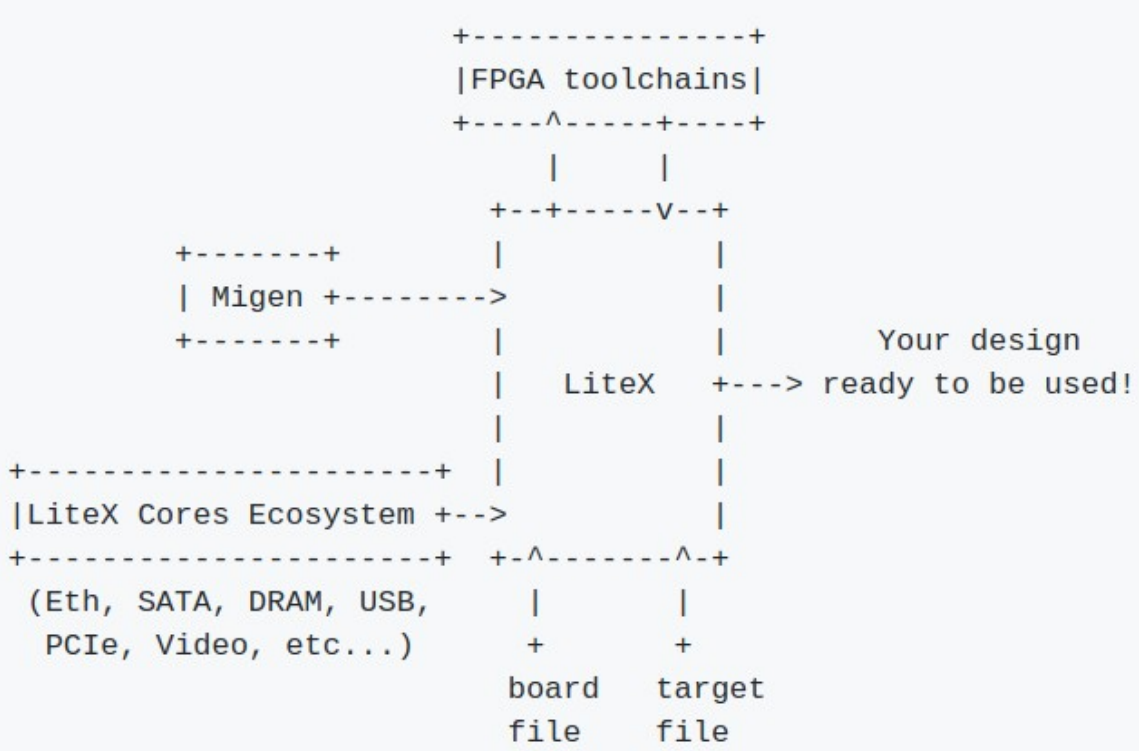
12:42 AM · Dec 15, 2019 · Twitter for Android

- LiteX is a FPGA design/SoC builder that can be used to build cores, create SoCs and full FPGA designs.
- LiteX is based on Migen and provides specific building/debugging tools for a higher level of abstraction and compatibily with the LiteX core ecosystem.
- Think of Migen as a toolbox to create FPGA designs in Python and LiteX as a SoC builder to create/develop/debug FPGA SoCs in Python
- https://github.com/enjoy-digital/litex

- Bunnie:
  LiteX vs. Vivado: First Impressions

  https://www.bunniestudios.com/blog/?p=5018

Typical LiteX design flow:

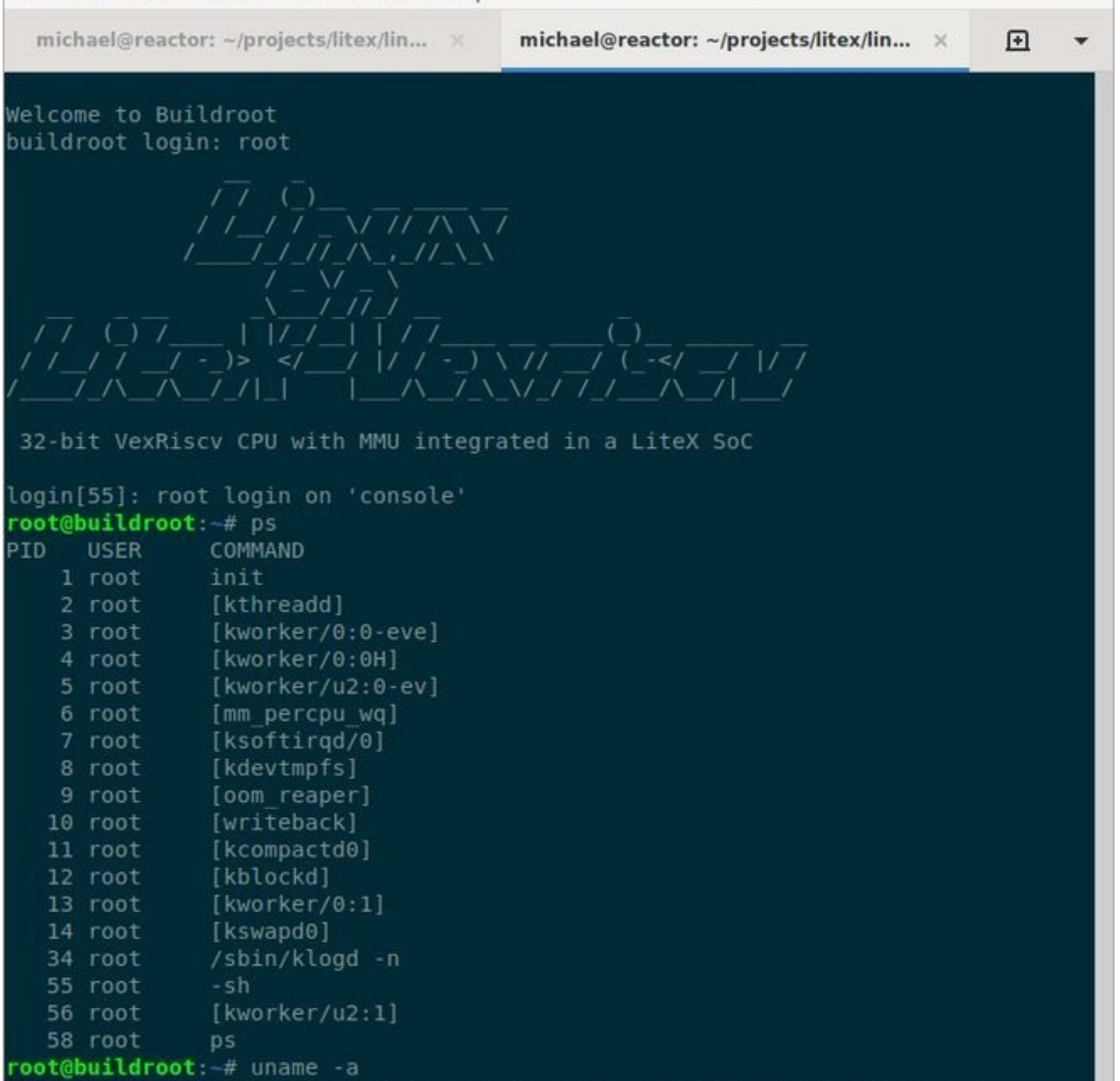


LiteX already supports various softcores CPUs: LM32, Mor1kx, PicoRV32, VexRiscv and is compatible with the LiteX's Cores Ecosystem:

Name	Build Status	Description
LiteDRAM	build passing	DRAM
LiteEth	build passing	Ethernet
LitePCIe	build passing	PCIe
LiteSATA	build passing	SATA

Linux on LiteX-VexRiscv

- Linux with VexRiscv CPU, a 32-bits Linux Capable RISC-V CPU written in Spinal HDL
- SoC around the VexRiscv CPU is created using LiteX as the SoC builder and LiteX's cores written in Migen Python DSL (LiteDRAM, LiteEth, LiteSDCard)
- github.com/litex-hub/linux-on-litex-vexriscv


michael@reactor: ~/projects/litex/lin...
michael@reactor: ~/projects/litex/lin...
Welcome to Buildroot
buildroot login: root
32-bit VexRiscv CPU with MMU integrated in a LiteX SoC
login[55]: root login on 'console'
root@buildroot:~# ps
PID USER COMMAND
1 root init
2 root [kthreadd]
3 root [kworker/0:0-eve]
4 root [kworker/0:0H]
5 root [kworker/u2:0-ev]
6 root [mm_percpu_wq]
7 root [ksoftirqd/0]
8 root [kdevtmpfs]
9 root [oom_reaper]
10 root [writeback]
11 root [kcompactd0]
12 root [kblockd]
13 root [kworker/0:1]
14 root [kswapd0]
34 root /sbin/klogd -n
55 root -sh
56 root [kworker/u2:1]
58 root ps
root@buildroot:~# uname -a

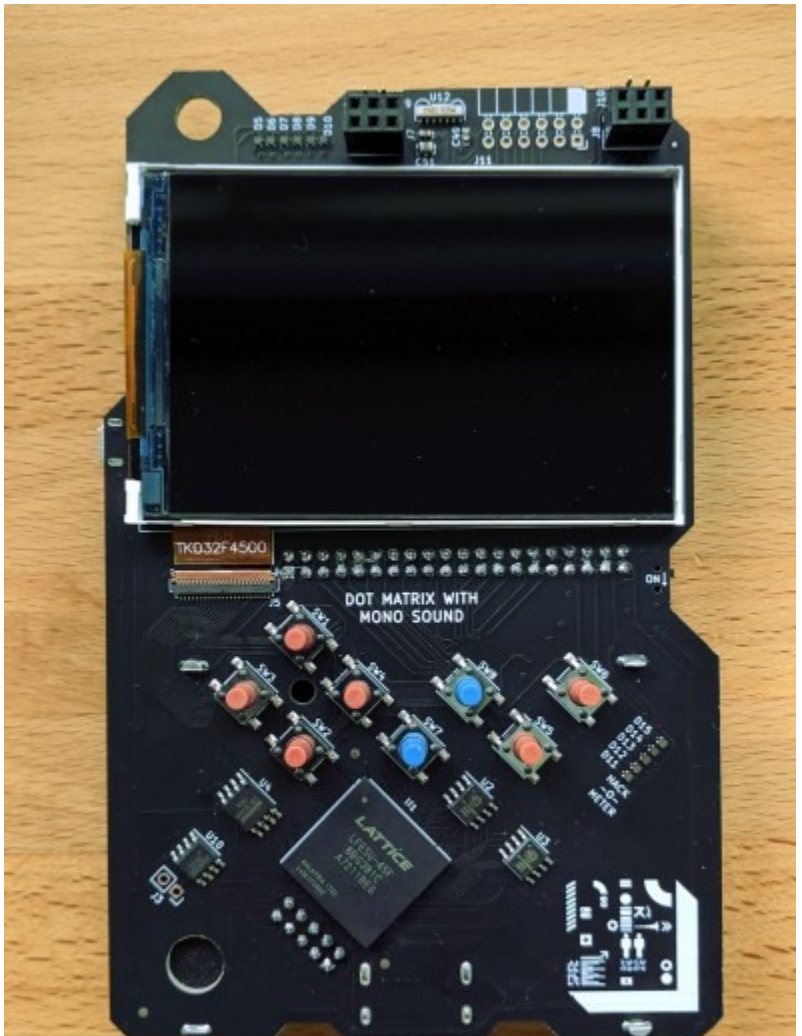
Slides: https://github.com/pdp7/talks/blob/master/nerp-riscv.pdf

Section:

Linux on the Hackaday Badge

Hackaday 2019 Supercon badge

- RISC-V “soft” core on ECP5 FPGA
- Gigantic FPGA In A Game Boy Form Factor



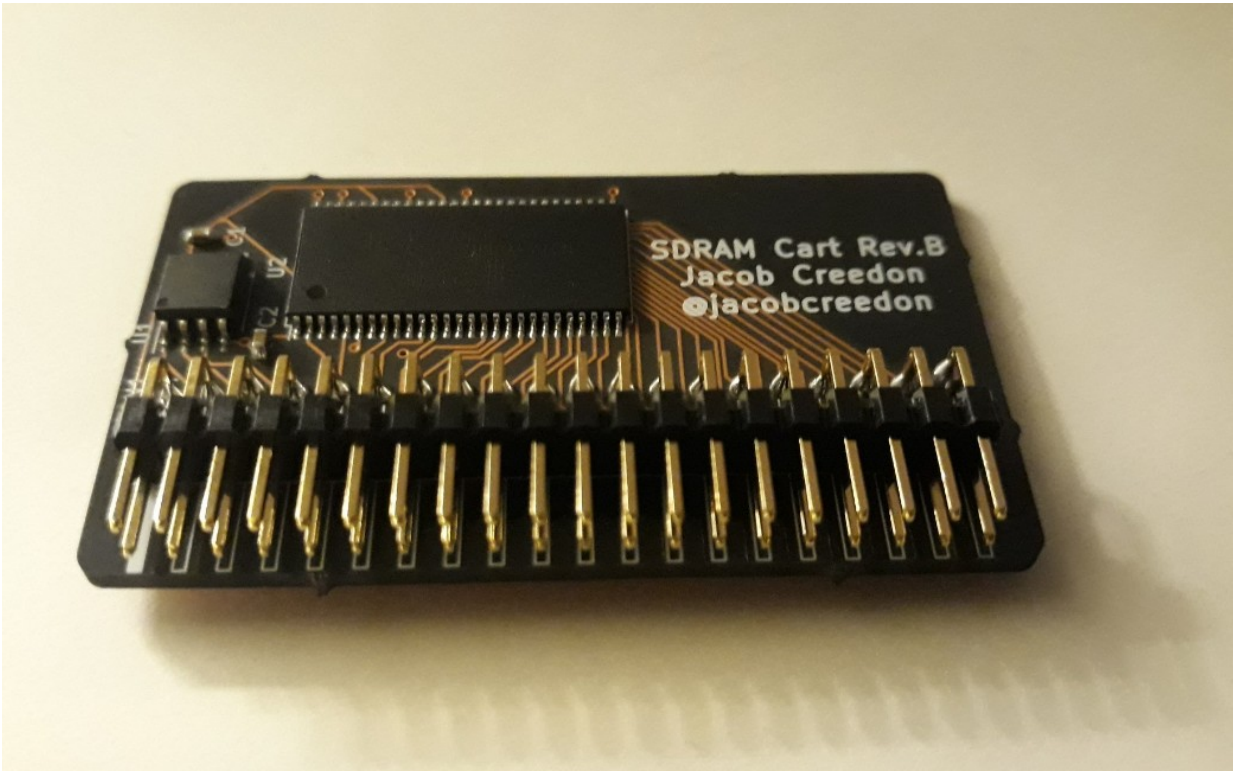
“Team Linux on Badge”

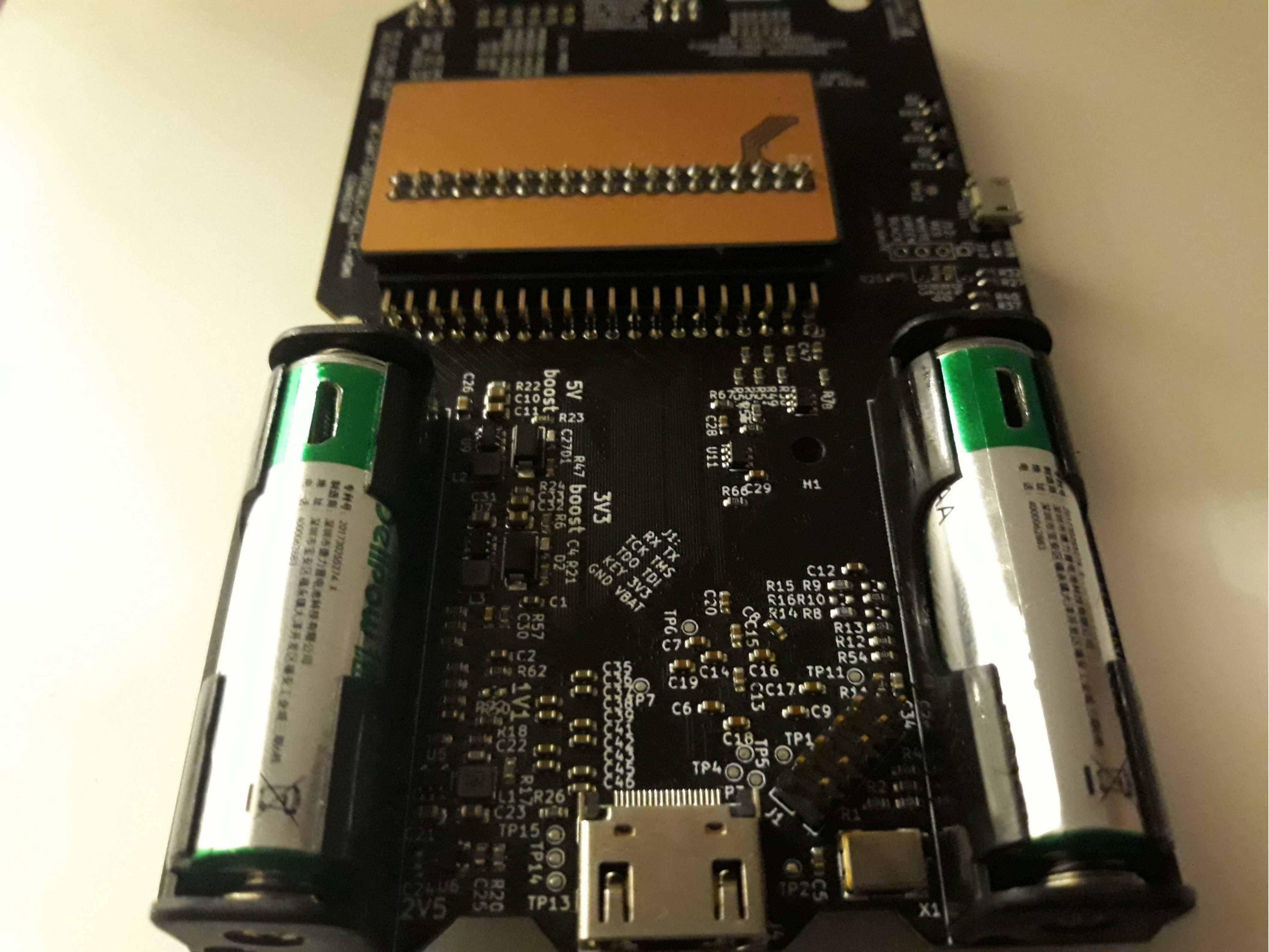
“Team Linux on Badge”

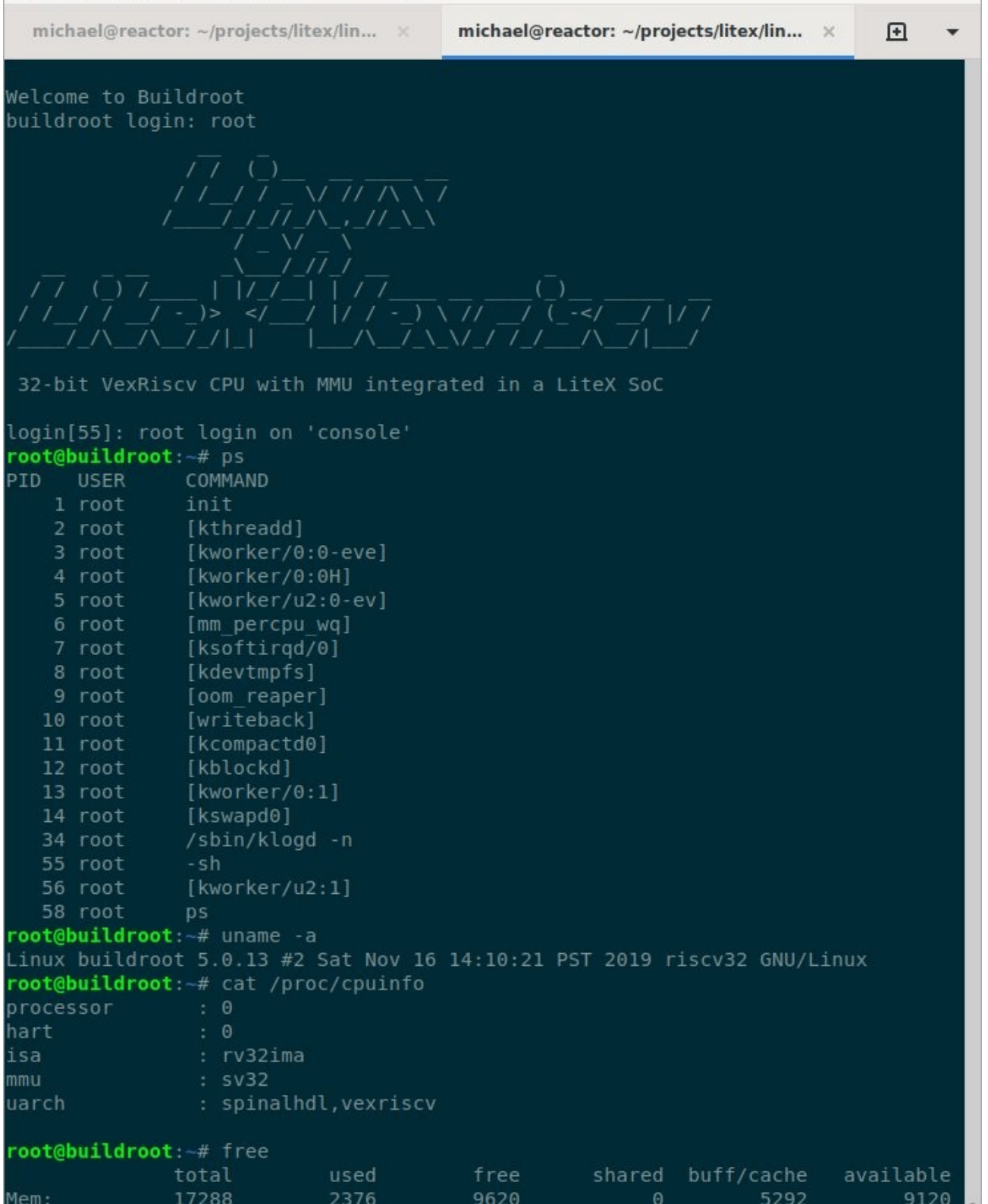
- Blog post: Hackaday Supercon badge boots Linux using SDRAM cartridge
  - https://blog.oshpark.com/2019/12/20/boot-linux-on-this-hackaday-supercon-badge-with-this-sdram-cartridge/
- Michael Welling (@QwertyEmedded), Tim Ansell (@mithro), Sean Cross (@xobs), Jacob Creedon (@jacobcreedon)
- First attempt: use the built-in 16MB SRAM… no luck :(
  - *(though xobs now might have a way to do it)*

“Team Linux on Badge”

- Second attempt:
  - Jacob Creedon designed an a cartridge board that adds 32MB of SDRAM to the Hackaday Supercon badge… before the event!




5V
boost
3V3
boost
J1:
RX TX
TCK TMS
TDO TDI
KEY 3V3
GND VBAT
1V1
2V5
H1
X1
TP1
TP2
TP4
TP5
TP6
TP7
TP11
TP13
TP14
TP15
R22
C10
C11
R23
R47
R24
R6
C4
R21
D2
C1
R57
C30
C2
R62
C22
R17
L1
R26
C23
C24
U6
C25
R20
C26
C31
R67
C28
U11
R66
C29
C12
R15 R9
R16 R10
R14 R8
R13
R12
R54
C20
C7
C15
C14
C16
C19
C13
C17
C6
C9
C18
C35
R1
R2
C5


michael@reactor: ~/projects/litex/lin...
michael@reactor: ~/projects/litex/lin...
Welcome to Buildroot
buildroot login: root
32-bit VexRiscv CPU with MMU integrated in a LiteX SoC
login[55]: root login on 'console'
root@buildroot:~# ps
PID USER COMMAND
1 root init
2 root [kthreadd]
3 root [kworker/0:0-eve]
4 root [kworker/0:0H]
5 root [kworker/u2:0-ev]
6 root [mm_percpu_wq]
7 root [ksoftirqd/0]
8 root [kdevtmpfs]
9 root [oom_reaper]
10 root [writeback]
11 root [kcompactd0]
12 root [kblockd]
13 root [kworker/0:1]
14 root [kswapd0]
34 root /sbin/klogd -n
55 root -sh
56 root [kworker/u2:1]
58 root ps
root@buildroot:~# uname -a
Linux buildroot 5.0.13 #2 Sat Nov 16 14:10:21 PST 2019 riscv32 GNU/Linux
root@buildroot:~# cat /proc/cpuinfo
processor : 0
hart : 0
isa : rv32ima
mmu : sv32
uarch : spinalhdl,vexriscv
root@buildroot:~# free
total used free shared buff/cache available
Mem: 17288 2376 9620 0 5292 9120

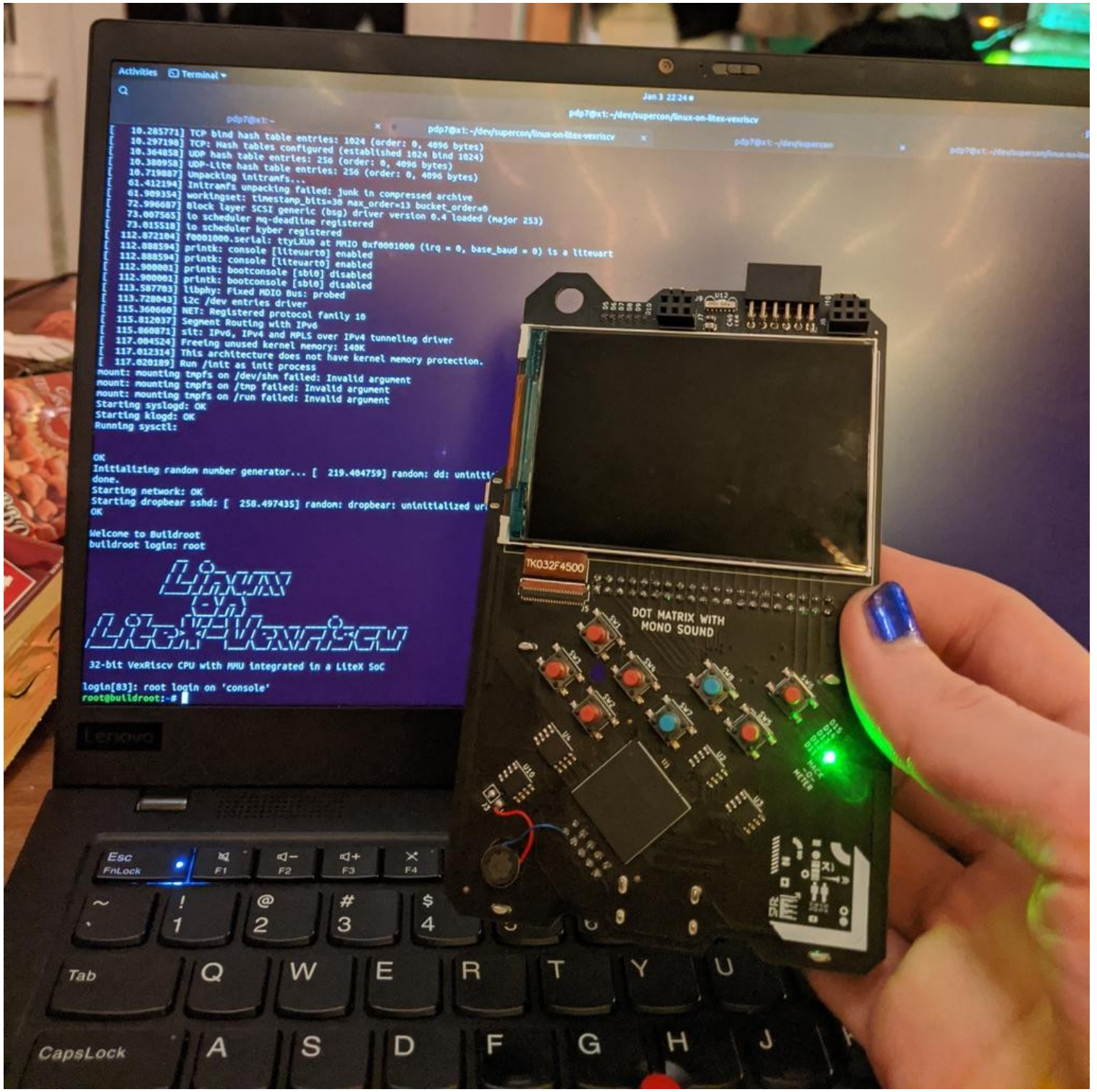
“Team Linux on Badge”

- https://youtu.be/3se_L0tRZeg?t=1055

Watch the demo during the Badge Hacking ceremony (jump to 17m 35s):

Linux on LiteX-VexRiscv

- Linux with VexRiscv CPU, a 32-bits Linux Capable RISC-V CPU written in Spinal HDL
  - github.com/litex-hub/linux-on-litex-vexriscv
- NOW with upstream support for the Hackaday Supercon badge!
  - https://github.com/litex-hub/litex-boards/pull/31


Activities
Terminal
Jan 3 22:24
pdp7@x1: ~/dev/supercon/linux-on-litex-vexriscv
[10.285771] TCP bind hash table entries: 1024 (order: 0, 4096 bytes)
[10.297198] TCP: Hash tables configured (established 1024 bind 1024)
[10.364858] UDP hash table entries: 256 (order: 0, 4096 bytes)
[10.380958] UDP-Lite hash table entries: 256 (order: 0, 4096 bytes)
[10.719887] Unpacking initramfs...
[61.412194] Initramfs unpacking failed: junk in compressed archive
[61.909354] workingset: timestamp_bits=30 max_order=13 bucket_order=0
[72.996687] Block layer SCSI generic (bsg) driver version 0.4 loaded (major 253)
[73.007565] io scheduler mq-deadline registered
[73.015518] io scheduler kyber registered
[112.872104] f0001000.serial: ttyLXU0 at MMIO 0xf0001000 (irq = 0, base_baud = 0) is a liteuart
[112.888594] printk: console [liteuart0] enabled
[112.888594] printk: console [liteuart0] enabled
[112.900001] printk: bootconsole [sbi0] disabled
[112.900001] printk: bootconsole [sbi0] disabled
[113.587703] libphy: Fixed MDIO Bus: probed
[113.728043] i2c /dev entries driver
[115.360660] NET: Registered protocol family 10
[115.812037] Segment Routing with IPv6
[115.860871] sit: IPv6, IPv4 and MPLS over IPv4 tunneling driver
[117.004524] Freeing unused kernel memory: 140K
[117.012314] This architecture does not have kernel memory protection.
[117.020189] Run /init as init process
mount: mounting tmpfs on /dev/shm failed: Invalid argument
mount: mounting tmpfs on /tmp failed: Invalid argument
mount: mounting tmpfs on /run failed: Invalid argument
Starting syslogd: OK
Starting klogd: OK
Running sysctl:
OK
Initializing random number generator... [219.404759] random: dd: uninitialized
done.
Starting network: OK
Starting dropbear sshd: [258.497435] random: dropbear: uninitialized
OK
Welcome to Buildroot
buildroot login: root
Linux on LiteX-VexRiscv
32-bit VexRiscv CPU with MMU integrated in a LiteX SoC
login[83]: root login on 'console'
root@buildroot:~#
TK032F4500
DOT MATRIX WITH
MONO SOUND
Lenovo
Esc
FnLock
F1
F2
F3
F4
Tab
CapsLock

- Opened GitHub issue:
  - optimize performance on Hackaday Badge #35
    - https://github.com/litex-hub/litex-boards/issues/35
- Now 10x faster!
  - https://asciinema.org/a/Pcm3vd1BEdEKY9srYX6MsNfCE
  - Thanks to enjoy-digital

asciinema.org/a/293450

Watch Record Docs Blog About

Asciicast updated.

```
[    7.896093] Initramfs unpacking failed: junk in compressed archive
[    7.953813] workingset: timestamp_bits=30 max_order=13 bucket_order=0
[    9.236463] Block layer SCSI generic (bsg) driver version 0.4 loaded (major 253)
[    9.239004] io scheduler mq-deadline registered
[    9.240102] io scheduler kyber registered
[   13.977920] f0001000.serial: ttyLXU0 at MMIO 0xf0001000 (irq = 0, base_baud = 0) is a liteuart
[   13.980290] printk: console [liteuart0] enabled
[   13.980290] printk: console [liteuart0] enabled
[   13.982965] printk: bootconsole [sbi0] disabled
[   13.982965] printk: bootconsole [sbi0] disabled
[   14.058778] libphy: Fixed MDIO Bus: probed
[   14.074959] i2c /dev entries driver
[   14.247461] NET: Registered protocol family 10
[   14.307974] Segment Routing with IPv6
[   14.315214] sit: IPv6, IPv4 and MPLS over IPv4 tunneling driver
[   14.455698] Freeing unused kernel memory: 140K
[   14.457905] This architecture does not have kernel memory protection.
[   14.459170] Run /init as init process
mount: mounting tmpfs on /dev/shm failed: Invalid argument
mount: mounting tmpfs on /tmp failed: Invalid argument
mount: mounting tmpfs on /run failed: Invalid argument
Starting syslogd: OK
Starting klogd: OK
Running sysctl: OK
Initializing random number generator... [   23.063050] random: dd: uninitialized urandom read (512 bytes read)
done.
Starting network: OK
Starting dropbear sshd: [   27.336210] random: dropbear: uninitialized urandom read (32 bytes read)
OK

Welcome to Buildroot
```

▶ 00:00

Linux boots on Hackaday Supercon FPGA badge [10x faster!]

Slides:
github.com/pdp7/talks/blob/master/lug-riscv.pdf

Drew Fustini
drew@oshpark.com
@pdp7 / @oshpark

This work is licensed under a Creative Commons Attribution-ShareAlike 4.0 International License.

Slides: https://github.com/pdp7/talks/blob/master/lug-riscv.pdf

bonus section:

Open Source Hardware laptop from Berlin

MNT Reform

Open Source DIY Laptop for Hacking, Customization, and Privacy

This project is coming soon. Sign up to receive updates and be notified when this project launches.

me@example.com Subscribe

Introducing the much more personal computer.

Modern laptops have secret schematics, glued-in batteries, and mystery components all over. But Reform is the opposite — it invites both curious makers and privacy aware users to take a look under the hood, customize the documented electronics, and 3D-print their own parts.