

Slides: <https://github.com/pdp7/talks/blob/master/rv-israel.pdf>

Linux on RISC-V

with open source hardware and open source FPGA tools

Embedded Israel meetup (April 6th, 2020)



Drew Fustini
drew@oshpark.com
Twitter: [@pdp7](https://twitter.com/pdp7)



- Open Source Hardware designer at OSH Park
 - PCB manufacturing service in the USA
 - drew@oshpark.com / Twitter: [@oshpark](https://twitter.com/@oshpark)
- Volunteer Member of Board of Directors of BeagleBoard.org Foundation
 - small open source Linux boards
 - drew@beagleboard.org
- Volunteer Member of the Board of Directors of the Open Source Hardware Association (OSHWA)
 - Open Source Hardware Certification Program: <https://certification.oshwa.org/>



Statement of Principles:

Hardware whose **design** is made **publicly available** so that anyone can **study**, **modify**, **distribute**, **make**, and **sell** the design or hardware based on that design



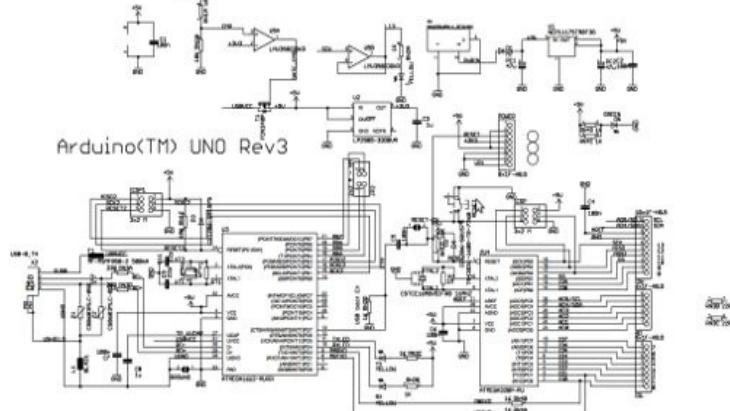
Open Source Hardware



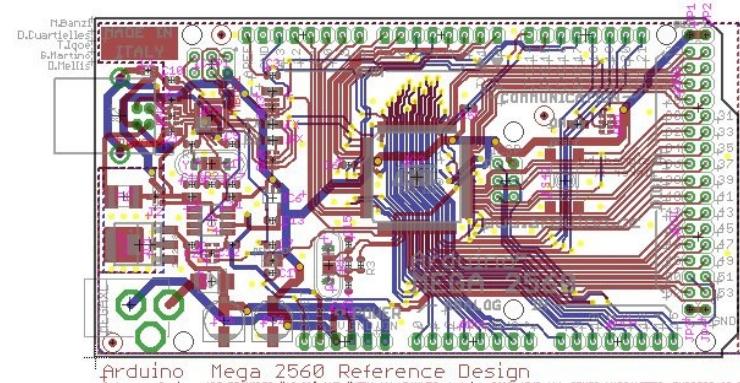
Documentation required for electronics:



Schematics



Board Layout



Editable source files for CAD software such as KiCad or EAGLE



Bill of Materials (BoM)

Not strict requirement, but best practice is for all components available from distributors in **low quantity**

36c3 talk: Linux on Open Source Hardware with Open Source chip design



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/rv-israel.pdf>

Section:
RISC-V

the instruction set for everything?

- When you write a C or C++ program, it is compiled into instructions for the microprocessor (CPU) to execute.
- How does the compiler know what instructions the CPU 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 Berkeley – *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

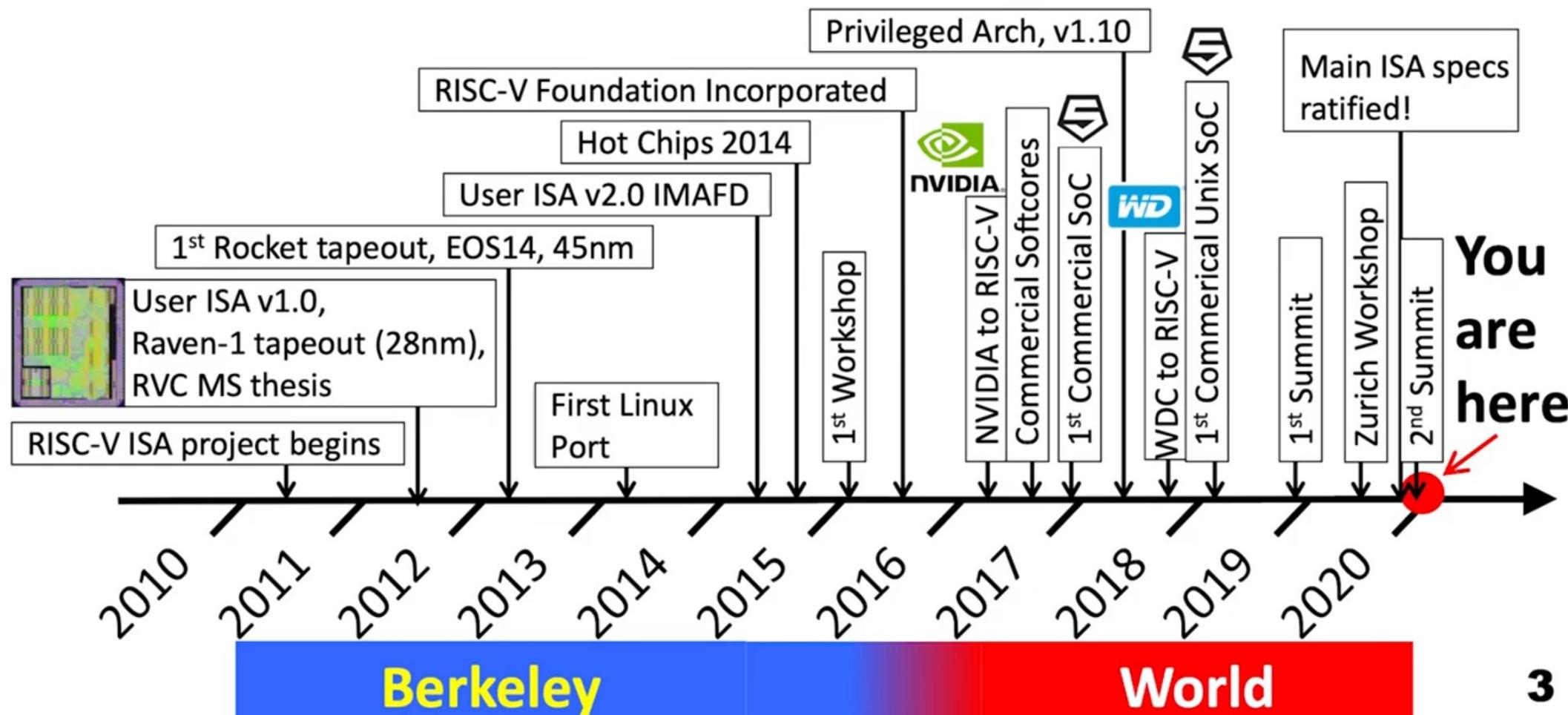


What's Different about RISC-V?

- ***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

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 ...

Software



ISA specification

Golden Model

Compliance

Hardware

Open-source cores:

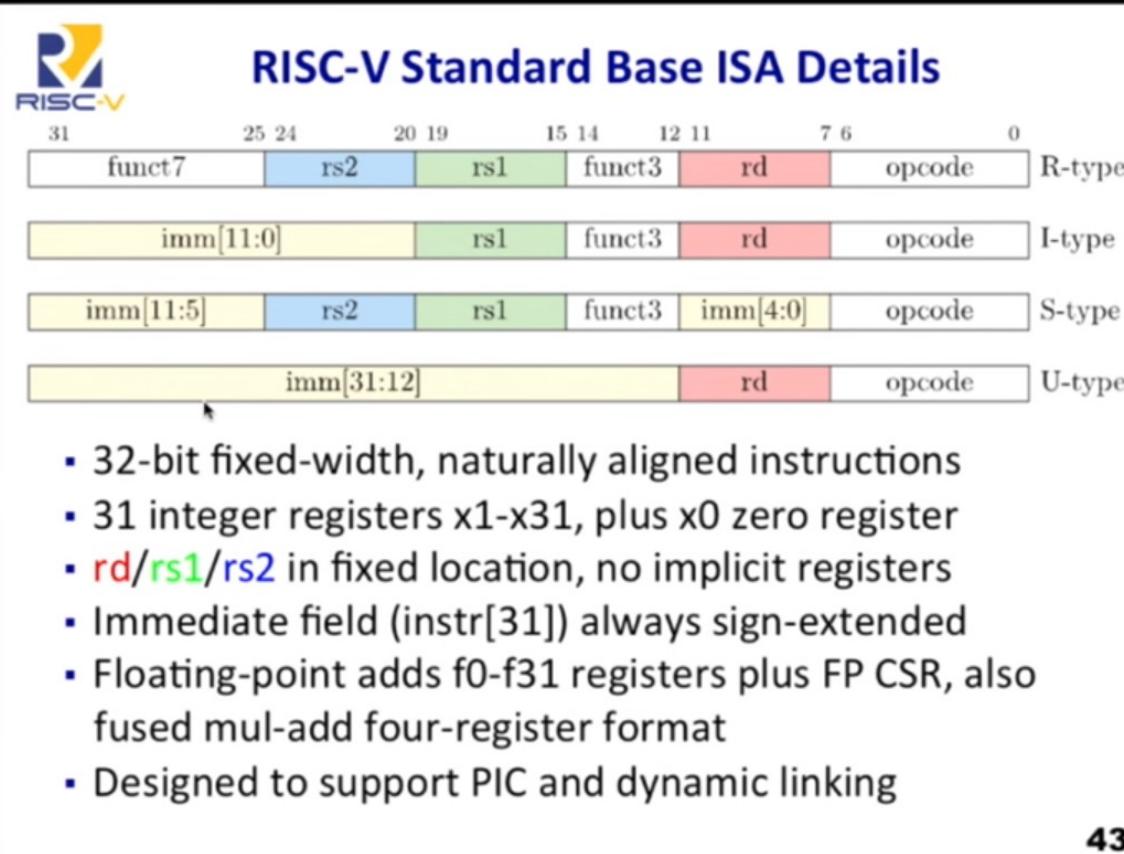
Rocket, BOOM, RI5CY, Ariane, PicoRV32, Piccolo, SCR1, Shakti, Serv, Swerv, Hummingbird, ...

Commercial core providers:

Alibaba, Andes, Bluespec, Cloudbear, Codasip, Cortus, InCore, Nuclei, SiFive, Syntacore, ...

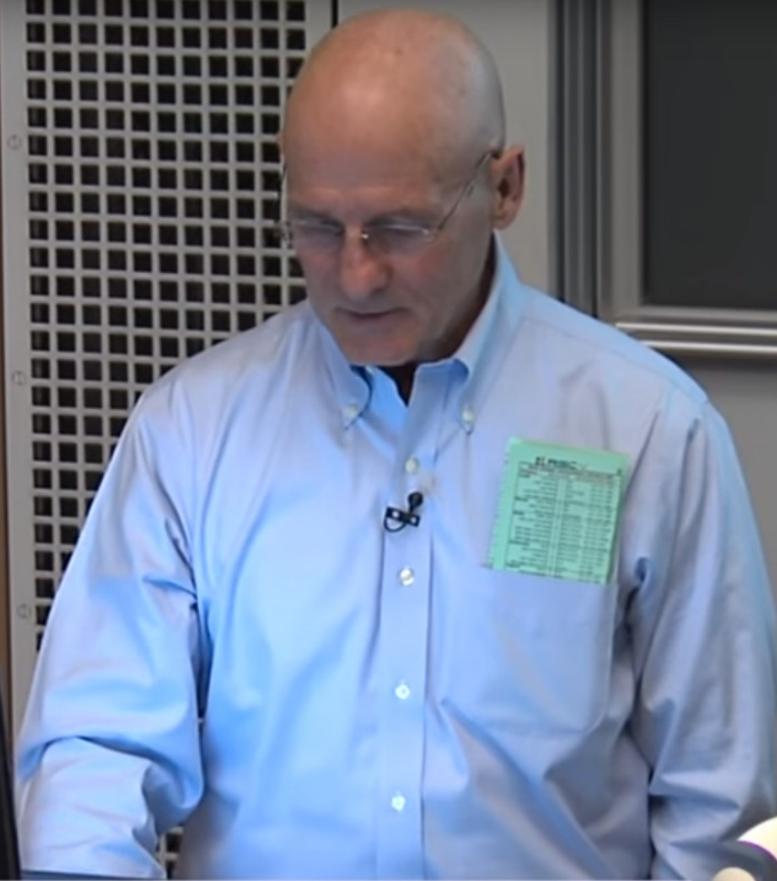
Inhouse cores:

Nvidia, WD, +others





RV32I					
RISC-V		① RISC-V Reference Card			
Base Integer Instructions (32-bit(64)/128-bit Base)					
Category Name Format RV32I/32-bit/128-bit Base					
Loads	Load Byte	I	LB rd,rs1,rs2		
	Load Halfword	I	LH rd,rs1,rs2		
	Load Word	I	LW rd,rs1,rs2		
	Load Byte Unsigned	I	LBU rd,rs1,rs2		
	Load Half Unsigned	I	LHU rd,rs1,rs2		
Stores	Store Byte	S	SB rs1,rd,rs2		
	Store Halfword	S	SH rs1,rd,rs2		
	Store Word	S	SW rd,rs1,rs2		
Shifts	Shift Left	R	SLT(W) rd,rs1,rs2		
	Shift Left Immediate	I	SLTI(W) rd,rs1,rs2		
	Shift Right	R	SLTU(W) rd,rs1,rs2		
	Shift Right Immediate	I	SLTUI(W) rd,rs1,rs2		
	Shift Right Arithmetic	R	SRAL(W) rd,rs1,rs2		
	Shift Right Arithmetic Immediate	I	SRALI(W) rd,rs1,rs2		
Arithmetic	ADD	R	ADD(W) rd,rs1,rs2		
	ADD Immediate	I	ADDI(W) rd,rs1,rs2		
	Subtract	R	DSUB(W) rd,rs1,rs2		
	Load Upper Imm.	U	LUI rd,imm		
	Load Upper Imm. to PC	U	LDUPC rd,imm		
Logical	XOR	R	XOR rd,rs1,rs2		
	XOR Immediate	I	XORI rd,rs1,rs2		
	OR	R	OR rd,rs1,rs2		
	OR Immediate	I	ORI rd,rs1,rs2		
	AND	R	AND rd,rs1,rs2		
	AND Immediate	I	ANDI rd,rs1,rs2		
Compare	Set <	R	SLT rd,rs1,rs2		
	Set < Immediate	I	SLTI rd,rs1,rs2		
	Set < Unsigned	R	SLTU rd,rs1,rs2		
	Set < Unsigned Immediate	I	SLTUI rd,rs1,rs2		
Branches	Branch =	SB	BEQ rd,rs1,rs2		
	Branch =	SB	BEQZ rd,rs1,rs2		
	Branch =	SB	BNE rd,rs1,rs2		
	Branch =	SB	BNEZ rd,rs1,rs2		
	Branch <	SB	BLT rd,rs1,rs2		
	Branch < Unsigned	SB	BLTU rd,rs1,rs2		
	Branch > Unsigned	SB	BLTZ rd,rs1,rs2		
Jump & Link	Jump	UJ	jal rd,imm		
	Jump & Link Register	UJ	jalr rd,rs1,rs2		
Synch	Sync Thread	I	FSYNC rd		
	Sync Inst & Data	I	FSYNC_I rd		
System	System CALL	I	SYSCALL rd		
	System BREAK	I	SYSEXIT rd		
Counters	Read CYCLE	R	RSYCL rd		
	Read CYCLE upper Half	R	RSYCLH rd		
	Read TIME	I	RSYCTIME rd		
	Read TIME upper Half	I	RSYCTHME rd		
	Read INSTR RETired	I	RSYCTINRET rd		
	Read INSTR RETired upper Half	I	RSYCTINRETH rd		



Dave Patterson
UC BERKELEY



**RV32I / RV64I / RV128I + M, A, F, D, Q, C
RISC-V “Green Card”**



Base Integer Instructions (32/64/128)

Category	Name	Format	RV32/64/128 Basic
Loads	Load Byte	I	LB
	Load Halfword	I	LH
	Load Word	I	LD
	Load Halfword Unaligned	I	LH.U
	Load Word Unaligned	I	LD.U
Stores	Store Byte	S	SB
	Store Halfword	S	SH
	Store Word	S	SD
Shifts	Shift Left	R	SL
	Shift Left Immediate	I	SLI
	Shift Right	R	SR
	Shift Right Immediate	I	SRI
	Shift Right Arithmetic	R	SA
	Shift Right Arithmetic Immediate	I	SRI.A
Arithmetic	Add	R	ADD
	Add Immediate	I	ADDI
	Subtract	R	SUB
	Subtract Immediate	I	SUBI
	Mul	R	MUL
	Mul Immediate	I	MULI
	Divide	R	DIV
	Divide Immediate	I	DIVI
Logical	Not	R	NOT
	Not Immediate	I	NOTI
	Or	R	OR
	Or Immediate	I	ORI
	And	R	AND
	And Immediate	I	ANDI
Compare	Set =	R	SSET
	Set < Immediate	I	SSETI
	Set < Unsigned	I	SSETU
Branches	Branch	I	BEQ
	Branch +	I	BNE
	Branch -	I	BGE
	Branch -	I	BGT
	Branch < Unsigned	I	BGEU
	Branch < Unsigned	I	BGTU
Jump & Link	Jump	I	JAL
	Jump & Link Register	I	JALR
Synch	Synch Branch	I	FSYNC
	Synch Branch & Status	I	FSYNC_I
System	System CALL	I	SYSCALL
	System RETRET	I	SYCRETRET
Counters	Read CYCLE	I	RSVCYC
	Read CYCLE upper Half	I	RSVCYCH
	Read TIME	I	RSVTIME
	Read TIME upper Half	I	RSVTIMEH
	Read INSTR RETVAL	I	RSVINSTRRET
	Read INSTR RETVAL half	I	RSVINSTRRETH

RV Privileged Instructions (32/64/128)

Category	Name	Format	RV32/64/128
C/SW Access	Atomic R/W	R	CSWRW
	Atomic Read & Set R/W	R	CSSRW
	Atomic Read & Clear R/W	R	CSCRW
	Atomic R/W Imm	R	CWRIM
	Atomic Read & Set R/W Imm	R	CSSRIM
	Atomic Read & Clear R/W Imm	R	CSCRIM
Change Level	Env. Call	R	RCALL
	Environment Breakpoint	R	RCB
Trap Redirect	To Supervisor	R	RTS
	Redirect Trap to Supervisor	R	RTSTS
	Processor Trap to Supervisor	R	PTSTS
Interrupt	Wait for Interrupt	R	RIWT
	Processor PENDING	R	RPND
	Processor PENDING, 16-bit	R	RPND16
Optional Multiplication-Division Extension: RV32M			
Multiply	Multiply	R	MULL
	Multiply upper half	R	MULLH
	Multiply half signs/ints	R	MULLHSI
	Multiply upper half ints	R	MULLHSI
Divide	Divide	R	DIVU
	Divide Unsigned	R	DIVU
	Divide	R	DIV
	Divide Unsigned	R	DIV
Optional Atomic Instruction Extension: RVA			
Load	Load Reserved	R	LR
	More Store Condition	R	LSRC
	Swap	R	ASWAP
And	And	R	AMOD
Logical	Not	R	ANMOD
	And	R	ANMOD
	Or	R	ANMOD
Mix/Max	Miximum	R	AMIXMAX
	Miximum	R	AMIXMAX
	Miximum Unsigned	R	AMIXMAXU
	Miximum Unsigned	R	AMIXMAXU
3 Optional FP Extensions: RV64/128B (F/D/Q)			
Moves	Move from Integer	R	FMOV.F
	Move to Register	R	FMOV.X
Convert	Convert from Int	R	FPOV.F
	Convert from Int Unsigned	R	FPOV.M
	Convert to Int	R	FPOV.E
	Convert to Int Unsigned	R	FPOV.U
32-bit (RVC) and 32-bit Instruction Formats			
CE	CE		
C8B	C8B		
C1W	C1W		
CL	CL		
CS	CS		
CB	CB		
CJ	CJ		
	Jump target	U	
	U32		

3 Optional FP Extensions: RV64/128B (F/D/Q)

Category	Name	Format	RV64/128B (F/D/Q)
Load	Load	R	FDQW
	Store	S	FDQW
Arithmetic	ADC	R	FADDS
	Subtract	R	FADDS
	Mul	R	FMULS
	Divide	R	FDIVS
	SQRT	R	FSQRTS
Mul-Add	Mul-Add	R	FMADDS
	Multiply-Subtract	R	FMADDS
	Negative Multiply-Subtract	R	FMADDS
	Negative Multiply-Add	R	FMADDS
Sign Inject	Sign Invert	R	FSIGNS
	Sign Negate	R	FSIGNS
	Sign Negate, 16-bit	R	FSIGNS16
Mix/Max	Mix	R	FMIXS
	Max	R	FMIXS
Compare	Compare Float	R	FPOV.F
	Compare Float <	R	FPOV.L
	Compare Float =	R	FPOV.E
Categories	Cloud Top	R	FPOV.U
Moves	Move from Integer	R	FMOV.F
	Move to Register	R	FMOV.X
Convert	Convert from Int	R	FPOV.F
	Convert from Int Unsigned	R	FPOV.M
	Convert to Int	R	FPOV.E
	Convert to Int Unsigned	R	FPOV.U
Configuration	Read Global	R	FPREGS
	Read Registers	R	FPREGS
	Read Flags	R	FPFLGDS
	Save Status Reg	R	FPREGS
	Save Flags	R	FPFLGDS
	Save Round Mode Int	R	FPREGS
	Save Flags	R	FPFLGDS
	Save Flags	R	FPFLGDS
3 Optional FP Extensions: RV64/128B (F/D/Q)			
Moves	Move from Integer	R	FDQW
	Move to Register	R	FDQW
Convert	Convert from Int	R	FPOV.F
	Convert from Int Unsigned	R	FPOV.M
	Convert to Int	R	FPOV.E
	Convert to Int Unsigned	R	FPOV.U
Arithmetic	ADD	R	FDQADDS
	ADD Word	R	FDQADDS
	ADD Immediate	R	FDQADDS
	ADD Imm	R	FDQADDS
	ADD Double	R	FDQADDS
	ADD SP Imm	R	FDQADDS
	ADD SP Imm < 4	R	FDQADDS
	ADD SP Imm > 4	R	FDQADDS
	ADD Immediate	R	FDQADDS
	Load Immediate	R	FDQADDS
	Load User Imm	R	FDQADDS
	Move	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS
	Sub User Imm	R	FDQADDS
	Mov	R	FDQADDS
	Sub	R	FDQADDS
	Sub Word	R	FDQADDS
	Sub SP Imm	R	FDQADDS
	Sub SP Imm < 4	R	FDQADDS
	Sub SP Imm > 4	R	FDQADDS
	Sub Immediate	R	FDQADDS</td

RISC-V and Industry

- 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>
- Companies like Nvidia and Western Digital will ship millions of devices with RISC-V
- Avoid ARM licensing fees
- Freedom to leverage open source implementations
 - BOOM, Rocket, PULP, SweRV, and many more

[Join the Mailing Lists](#)info@riscv.org [YouTube](#) [Twitter](#) [LinkedIn](#) [GitHub](#) [YouTube](#) | [Member Login](#)[ABOUT](#) [MEMBERSHIP](#) [SPECS & SUPPORT](#) [CORES & TOOLS](#) [NEWS](#) [EVENTS](#)

RISC-V® Summit

THANKS FOR ATTENDING!

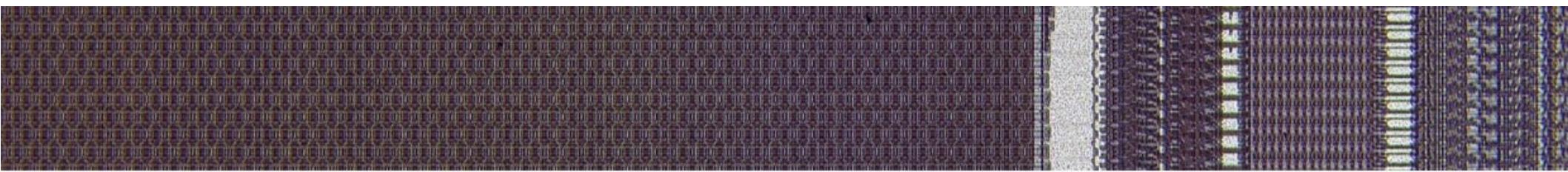
The **RISC-V Summit 2019** was held in San Jose, California on December 9-12. Slides are [now available](#), videos coming soon.



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?

- My column in the 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

Open-source chips

Breaking free of chip design monopolies with RISC-V



Drew Fustini

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 existing – 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.

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.

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

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.

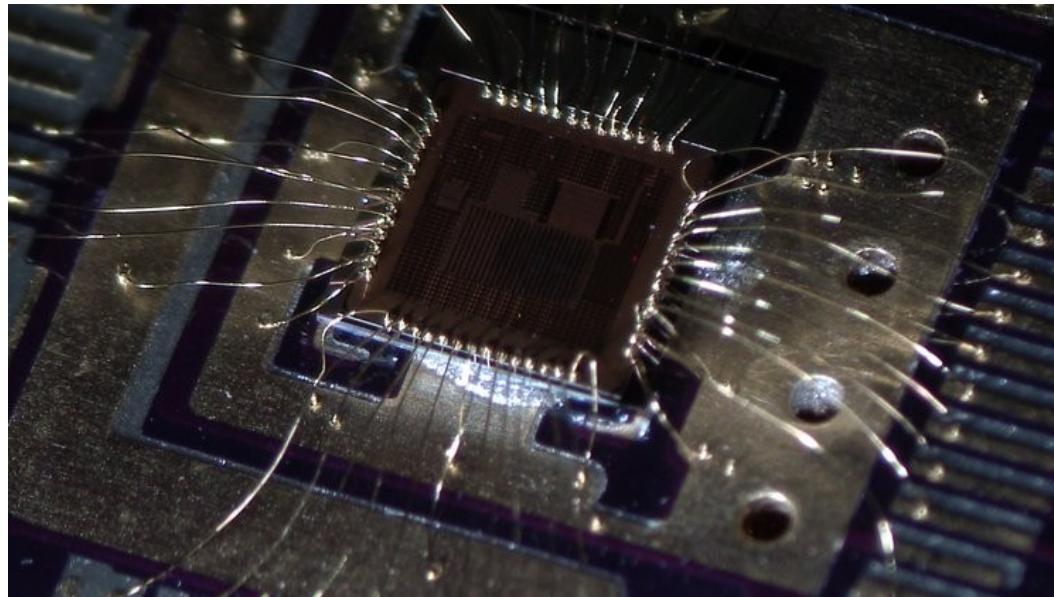
With these financial and political motivations, plus an increasingly mature software ecosystem, including Linux support, it won't be long before you have a device with a RISC-V processor in your home or pocket.

You can learn more about the exciting possibilities that RISC-V unleashes from Dr. Megan Wachs by pointing your web browser to hsmag.cc/qwted1.

- OnChip Open-V

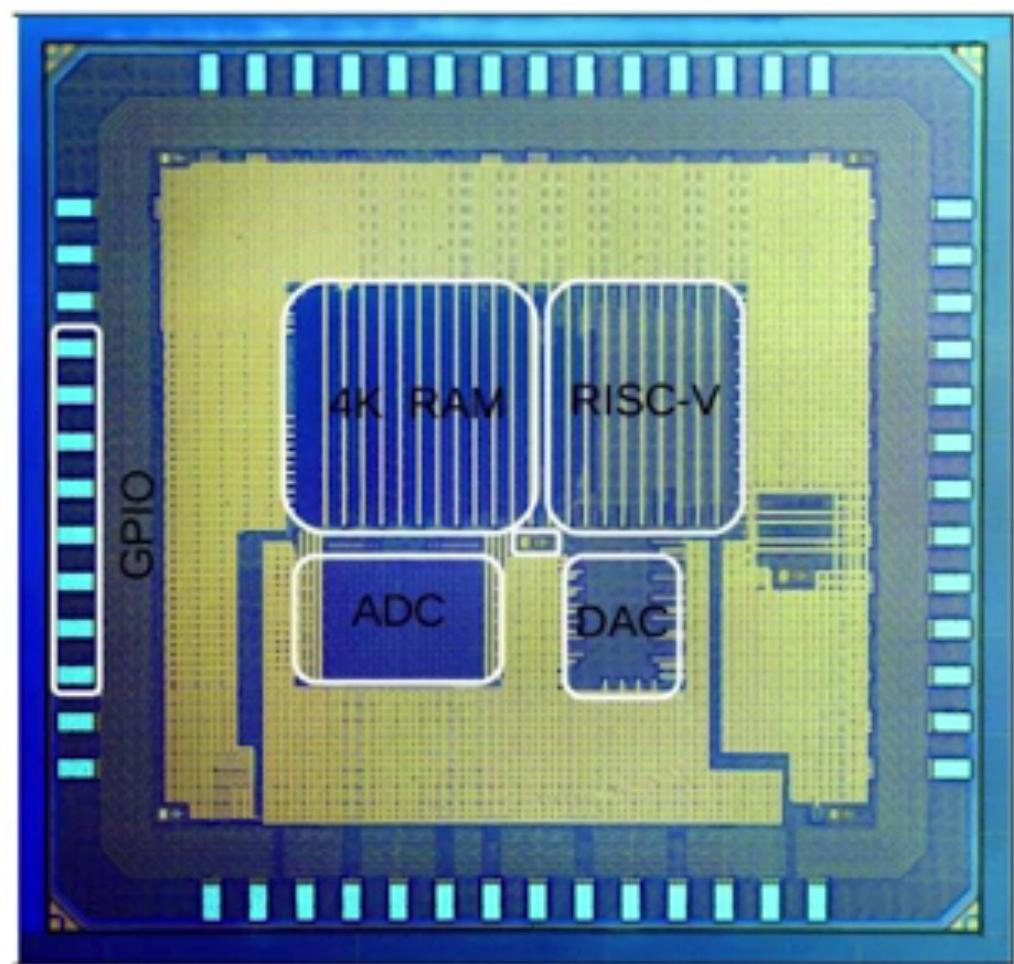
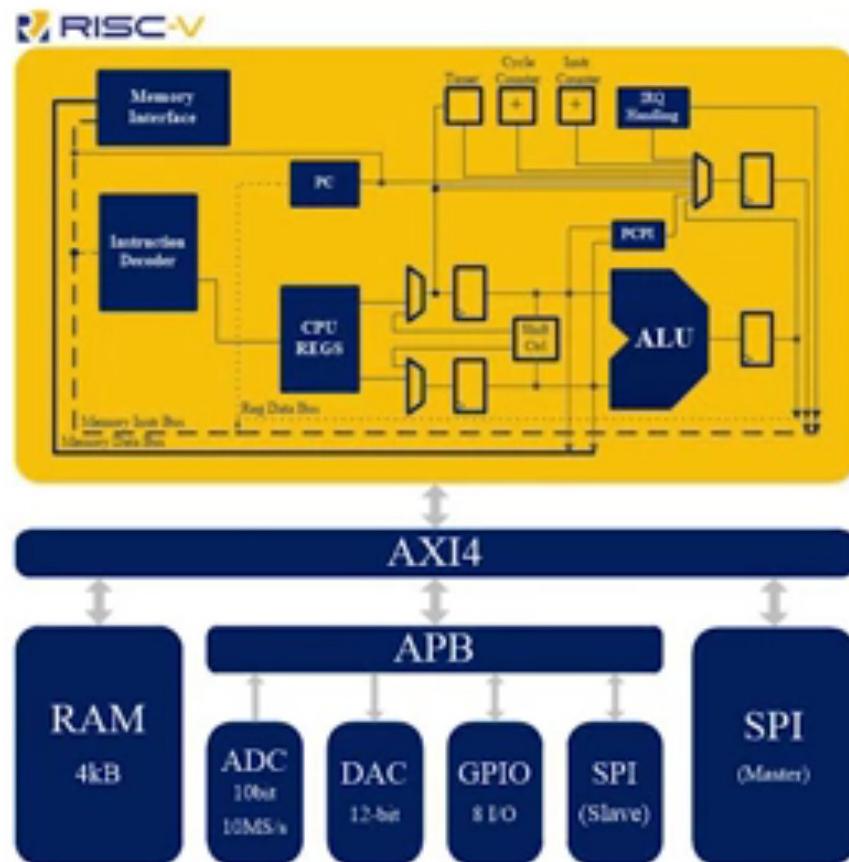


“completely free (as in freedom) and open source 32-bit microcontroller based on the RISC-V architecture”



OnChip Open-V

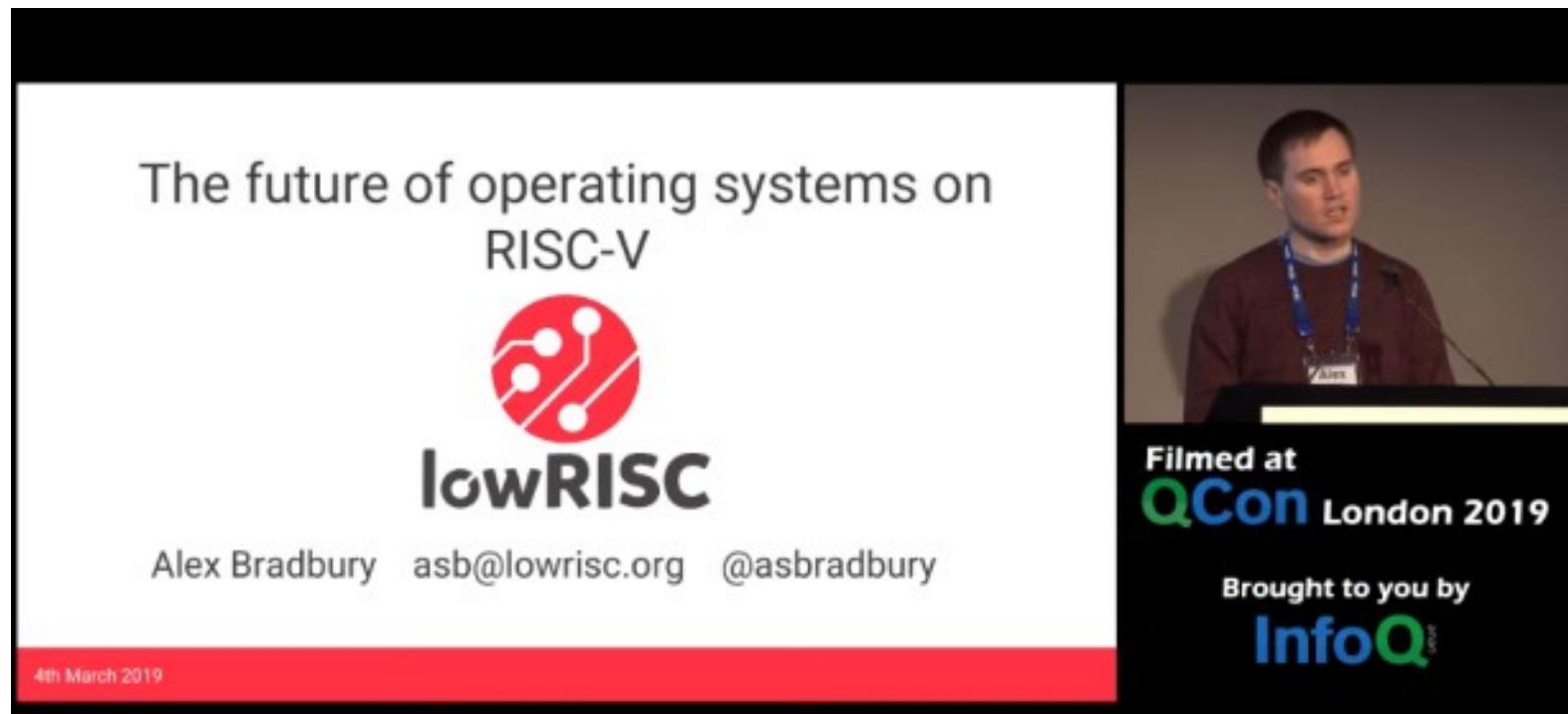
A 32-bit RISC-V based Microcontroller





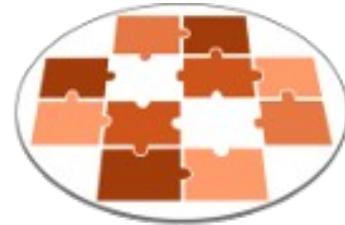
- **lowRISC** is a not-for-profit organisation whose goal is to produce a fully open source System-on-Chip (SoC) in volume
 - “We will produce a SoC design to populate a low-cost community development board and to act as an ideal starting point for derivative open-source and commercial designs”
- OpenTitan 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>



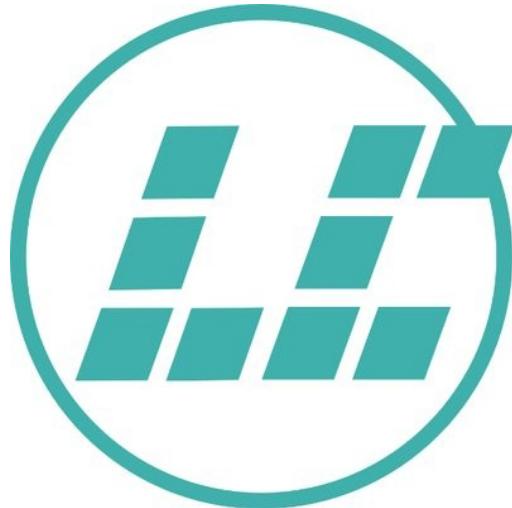


- Tutorial for the v0.7 lowRISC release
 - By Jonathan Kimmitt (lead developer), and Alex Bradbury (lead reviewer)
 - <https://www.cl.cam.ac.uk/~jrrk2/docs/ariane-v0.7/tutorial/>
 - Digilent Nexys A7-100T: \$265
 - This tutorial adds further functionality towards the final SoC design:
 - Graphical Colour Console with X-windows support incorporating mouse and keyboard events.
 - Choice of SD-Card, Quad-SPI or Ethernet TFTP boot-loader with DHCP support.
 - Linux 5.3.8 RISCV kernel and updated Debian userland with advanced package tool.
 - Choice of RV64-GC Rocket (Chisel) or Ariane (SystemVerilog) CPU



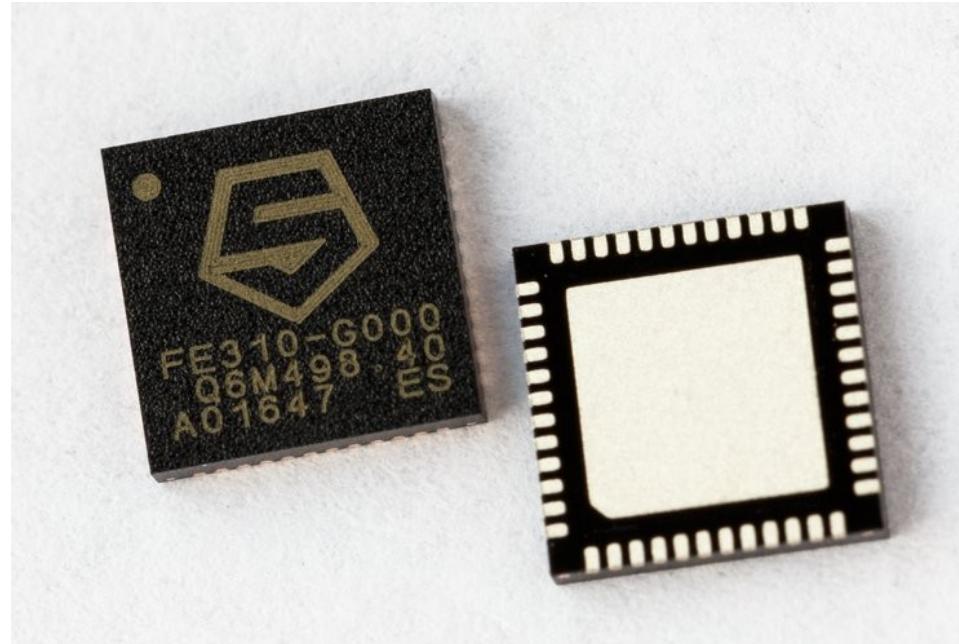
FOSSi
Foundation

- **FOSSi Foundation**
 - The Free and Open Source Silicon Foundation
 - “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”

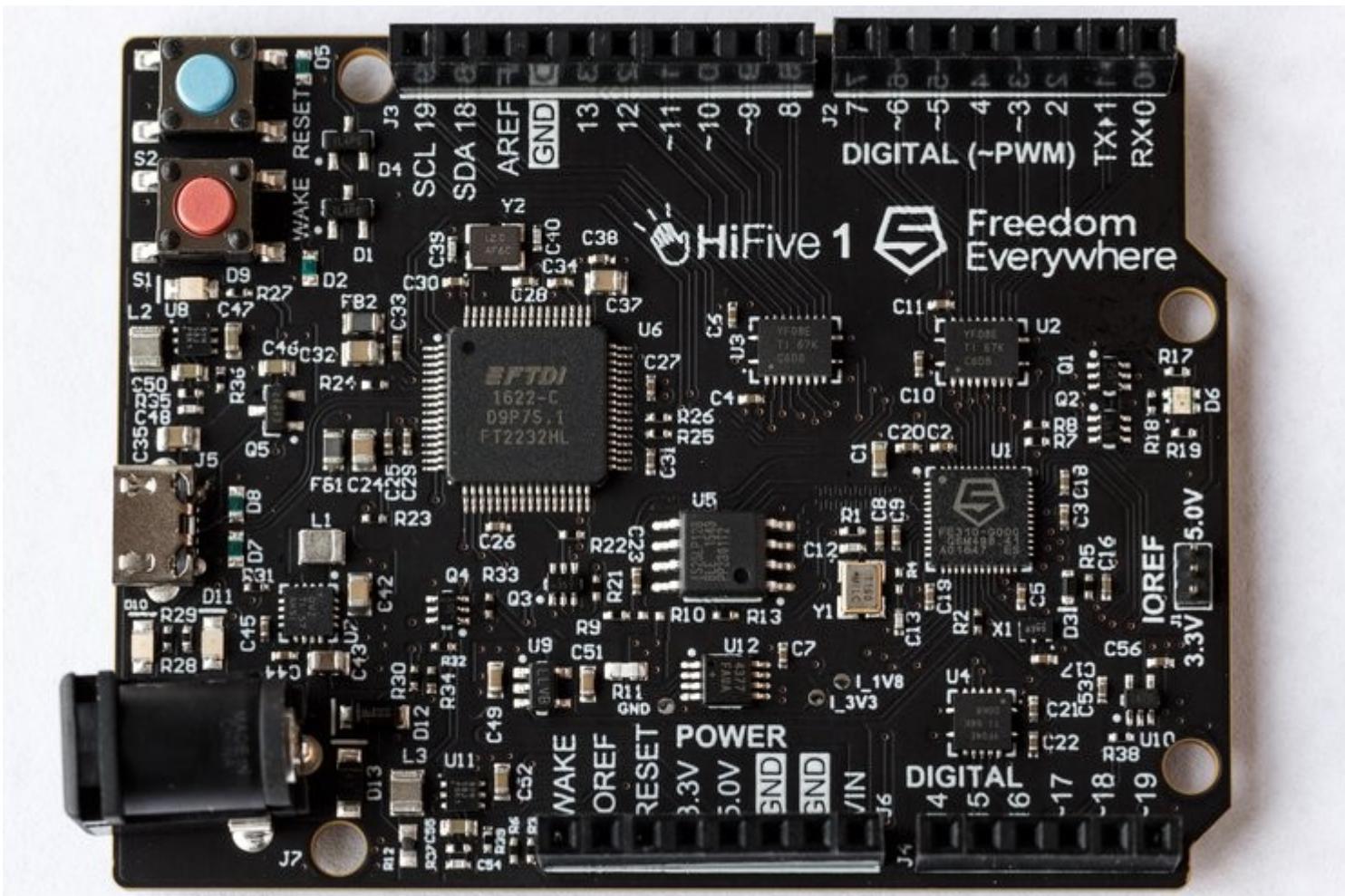
SiFive



- “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”

SiFive FE310 microcontroller

- HiFive1: Arduino-Compatible RISC-V Dev Kit



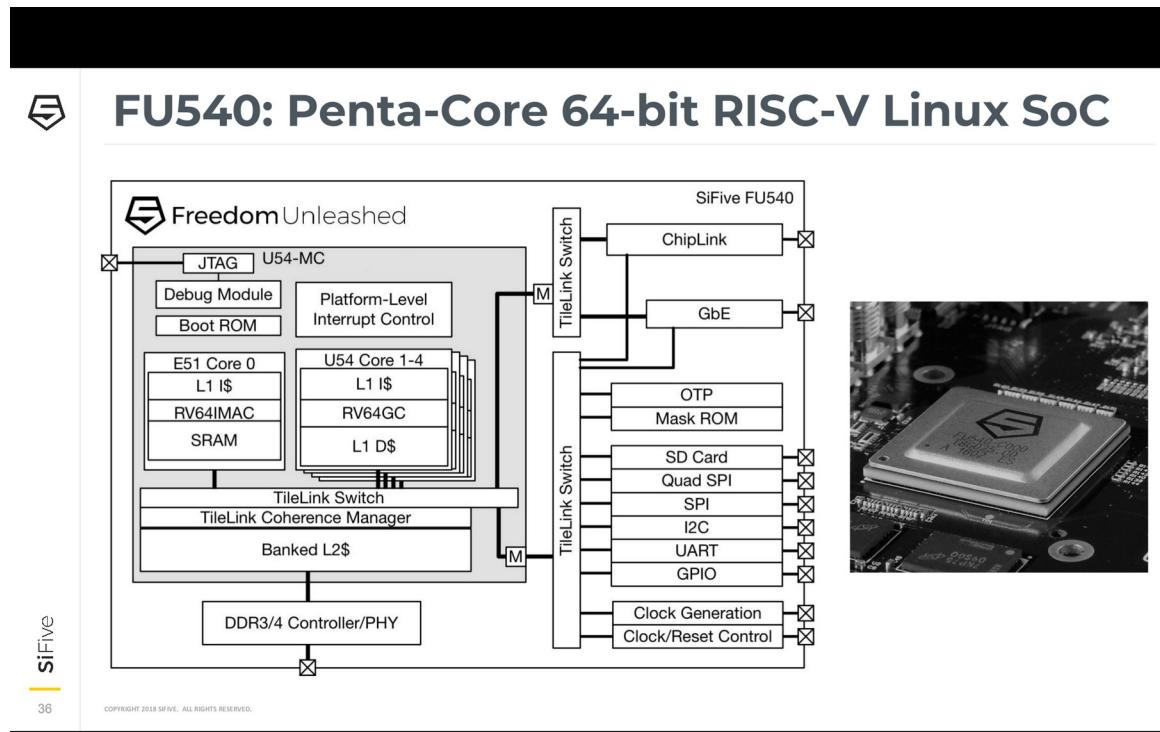
Slides: <https://github.com/pdp7/talks/blob/master/rv-israel.pdf>

Section:
Linux-capable RISC-V chips

SiFive: Linux on RISC-V

- FOSDEM 2018 talk

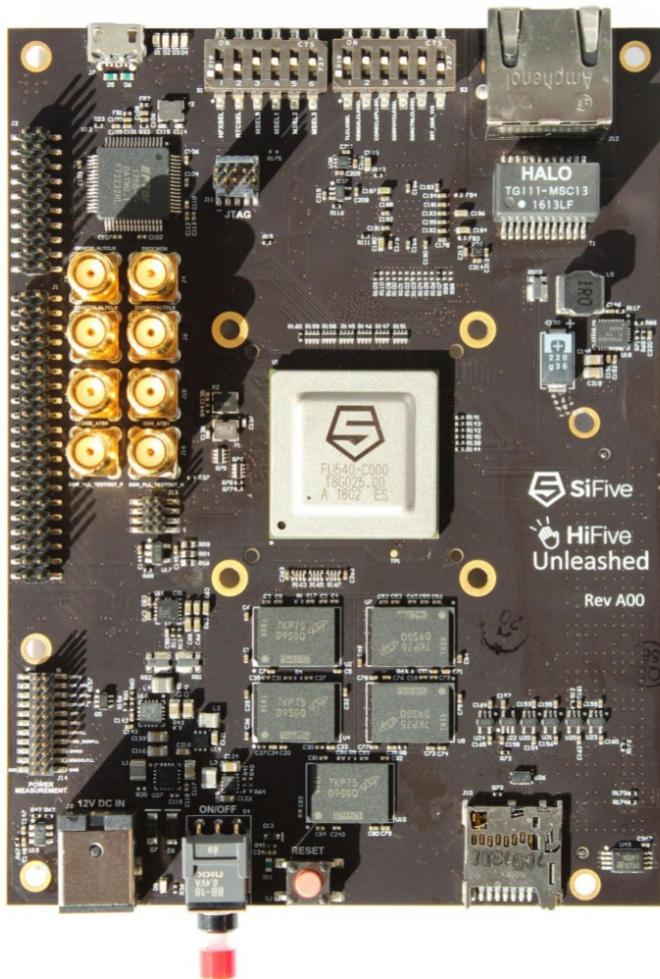
- YouTube: “Igniting the Open Hardware Ecosystem with RISC-V: SiFive's Freedom U500 is the World's First Linux-capable Open Source SoC Platform”
- Interview with Palmer Dabbelt of SiFive



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

RISC-V Summit 2019: Linux on RISC V Fedora and Firmware Status Update

- <https://www.youtube.com/watch?v=WC6e3g8uWdk>
- Wei Fu – Software Engineer, Red Hat

The screenshot shows a YouTube video player interface. At the top, the title is "RISC-V Summit 2019: 10 Linux on RISC V Fedora and Firmware Status Update". Below the title, the channel name "RISC-V" is visible with "7.91K subscribers". The video progress bar shows "9:56 / 18:37". The main content area displays a slide with the following text and graphics:

Fedora on RISC-V

Rich Software EcoSystem →

RISC-V
Instruction Sets Want to be Free!

Rich Hardware EcoSystem →

From www.codasip.com

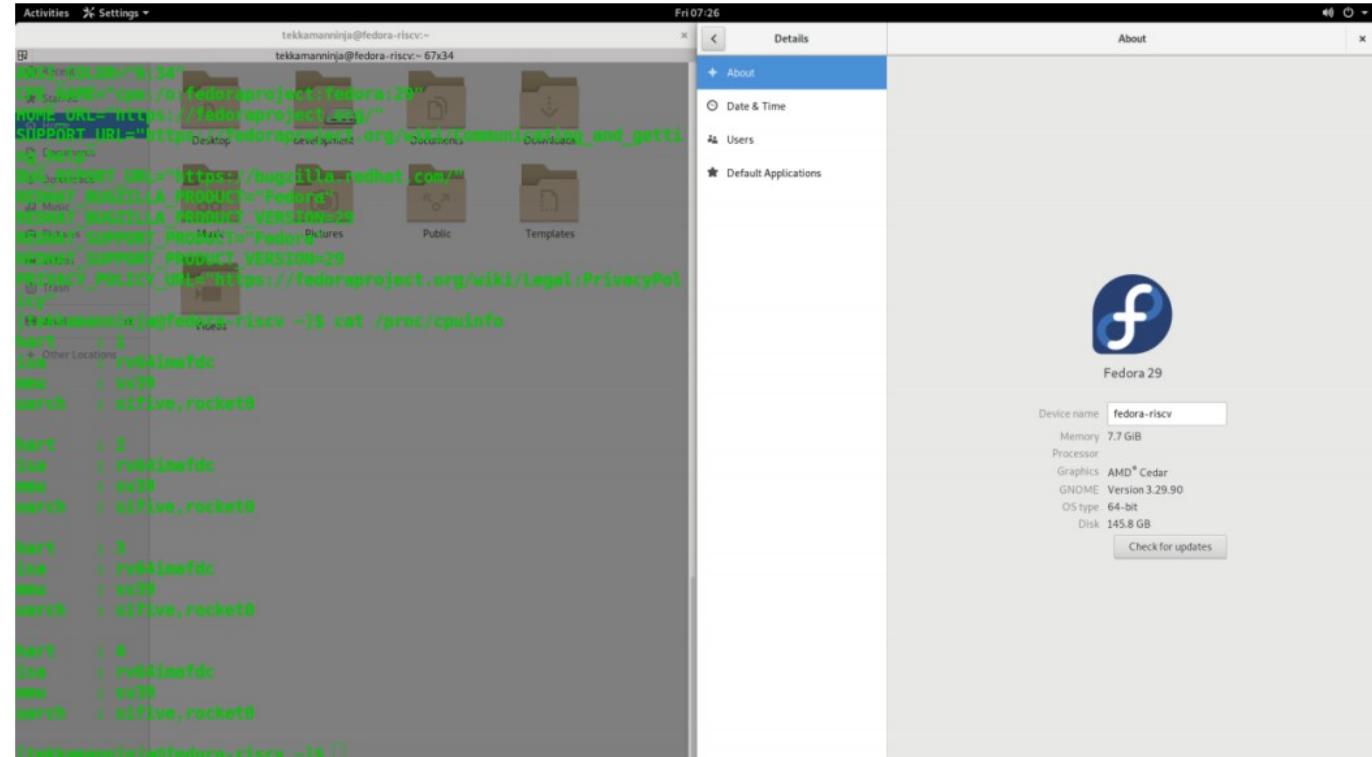
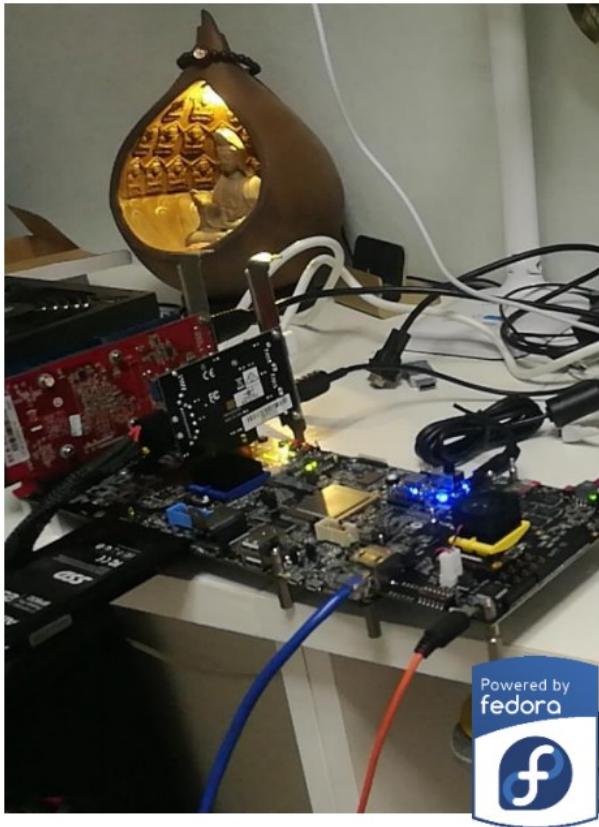
We would like to support more targets based on standard RISC-V Spec.

#RISCVSUMMIT | tmt.knect365.com/risc-v-summit/

Red Hat logo

Below the video player, the video description reads: "RISC-V Summit 2019: 10 Linux on RISC V Fedora and Firmware Status Update" and "183 views • Jan 16, 2020". The video stats show "1 like, 6 dislikes, 1 share, 1 save, and 3 comments". The "SUBSCRIBED" button is present at the bottom right.

Fedora GNOME Image on SiFive Unleashed



#RISCVSUMMIT | tmt.knect365.com/risc-v-summit/

Targets

Supported



Virtual: libvirt + QEMU

with graphics parameters (Spice).



Real Hardware: SiFive Unleashed

with Expansion Board, PCI-E graphic Card & SATA SSD

Tested



QEMU for AndeStar V5 && ADP-XC7KFF676

Andes QEMU and AndeShape **FPGA** board



中国科学院计算技术研究所
INSTITUTE OF COMPUTING TECHNOLOGY, CHINESE ACADEMY OF SCIENCES



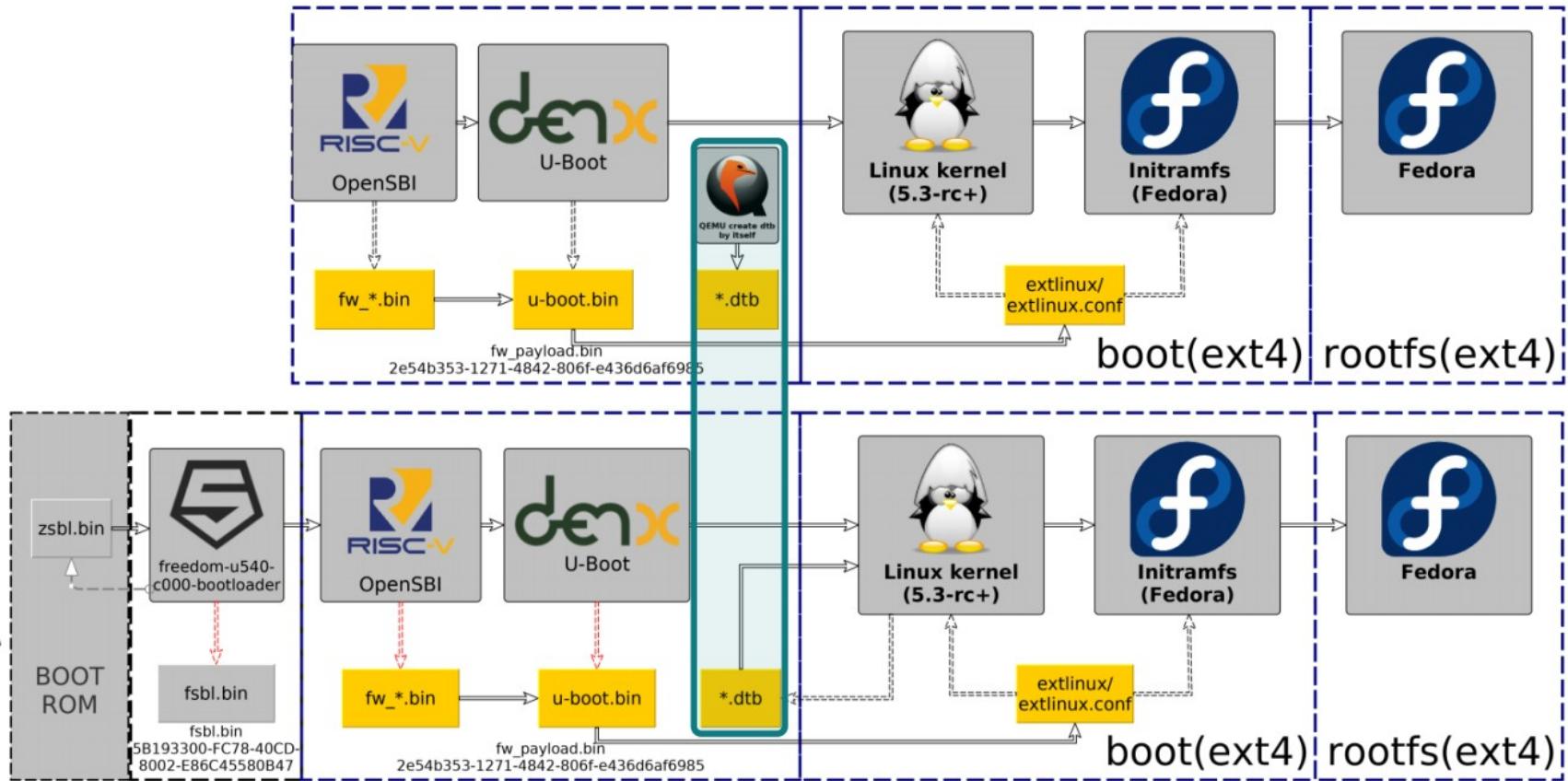
ICT SERVE Platform: FlameCluster

FPGA Cloud development platform (with PCI-E SSD and graphic Card)



#RISCVSUMMIT | tmt.knect365.com/risc-v-summit/

The current boot flow for Fedora on RISC-V





RISC-V

This page contains details about a port of Debian for the RISC-V architecture called **riscv64**.

Contents

[1. In a nutshell](#)

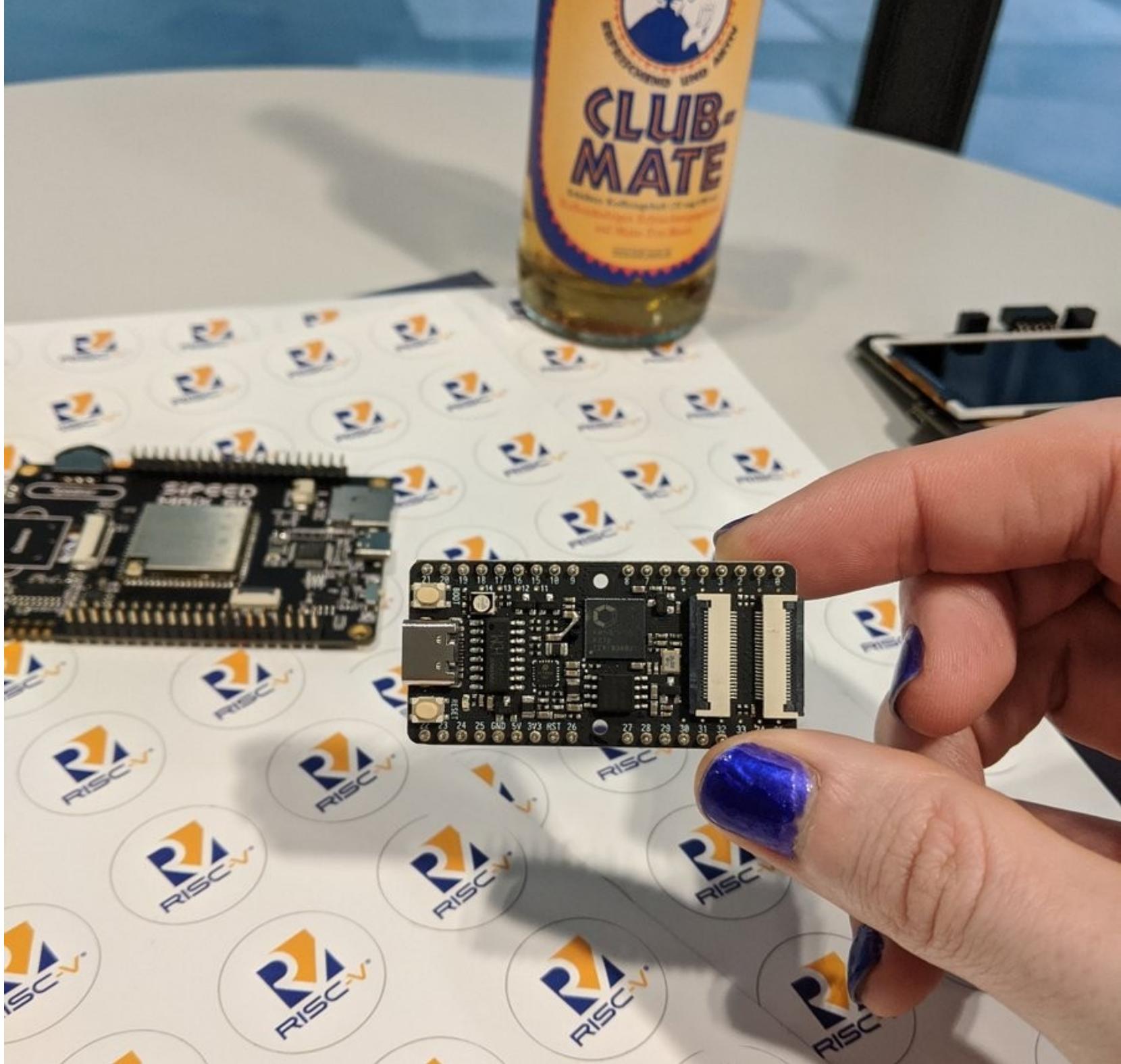
- [1. What is RISC-V?](#)
- [2. What is a Debian port?](#)
- [3. What are the goals of this project in particular?](#)
- [4. Progress](#)

[2. Upstream project / Architecture](#)

- [1. Upstream project / Community](#)
- [2. Architecture details](#)
- [3. Toolchain upstreaming status](#)

[3. Hardware](#)

- [1. ASIC implementations, i.e. "real" CPU chips](#)
 - [1. SiFive "Freedom U540" SoC \(quad-core RV64GC\) / "HiFive Unleashed"](#)
 - [2. Planned](#)
- [2. FPGA implementations](#)



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: sb10 at I/O port 0x0 (options '')
[ 0.000000] printk: bootconsole [sb10] 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 rwdta, 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

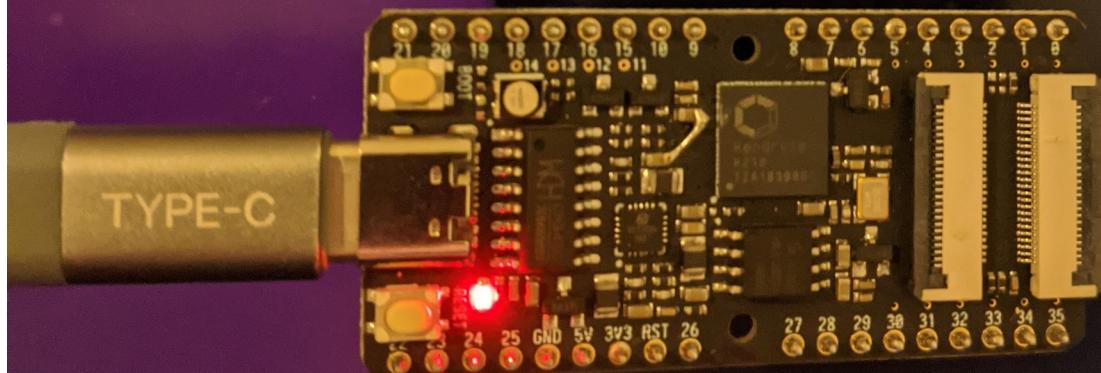
BusyBox v1.30.1 (2019-05-10 14:49:46 JST) hush - the humble shell

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

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

- **Linux now runs the low cost Kendryte K210 RISC-V processor**
 - dual core 64-bit RISC-V at 400MHz with 8MB SRAM
 - Sipeed MAix BiT for RISC-V is only \$13!
- Damien Le Moal at Linux Plumbers Conf:
RISC-V NOMMU and M-mode Linux
 - 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
- How to Build & Run Linux on Kendryte K210 RISC-V NOMMU Processor

```
File Edit Log Configuration Control signals View Help  
/ # uname -a  
Linux k210 5.6.0-rc1vowstar #1 SMP Mon Feb 17 23:  
/ # cat /proc/meminfo |head  
MemTotal:           6656 kB  
MemFree:            2496 kB  
MemAvailable:       2080 kB  
Buffers:             0 kB  
Cached:              1916 kB  
SwapCached:          0 kB  
Active:              0 kB  
Inactive:            0 kB  
Active(anon):        0 kB  
Inactive(anon):      0 kB  
/ # cat /proc/cpuinfo  
processor      : 0  
hart          : 0  
isa           : rv64imafdc  
  
processor      : 1  
hart          : 1  
isa           : rv64imafdc  
  
/ # tcc -run -nostdlib hello.c  
hello.c:2: warning: implicit declaration of function  
hello show 'n tell  
/ #
```



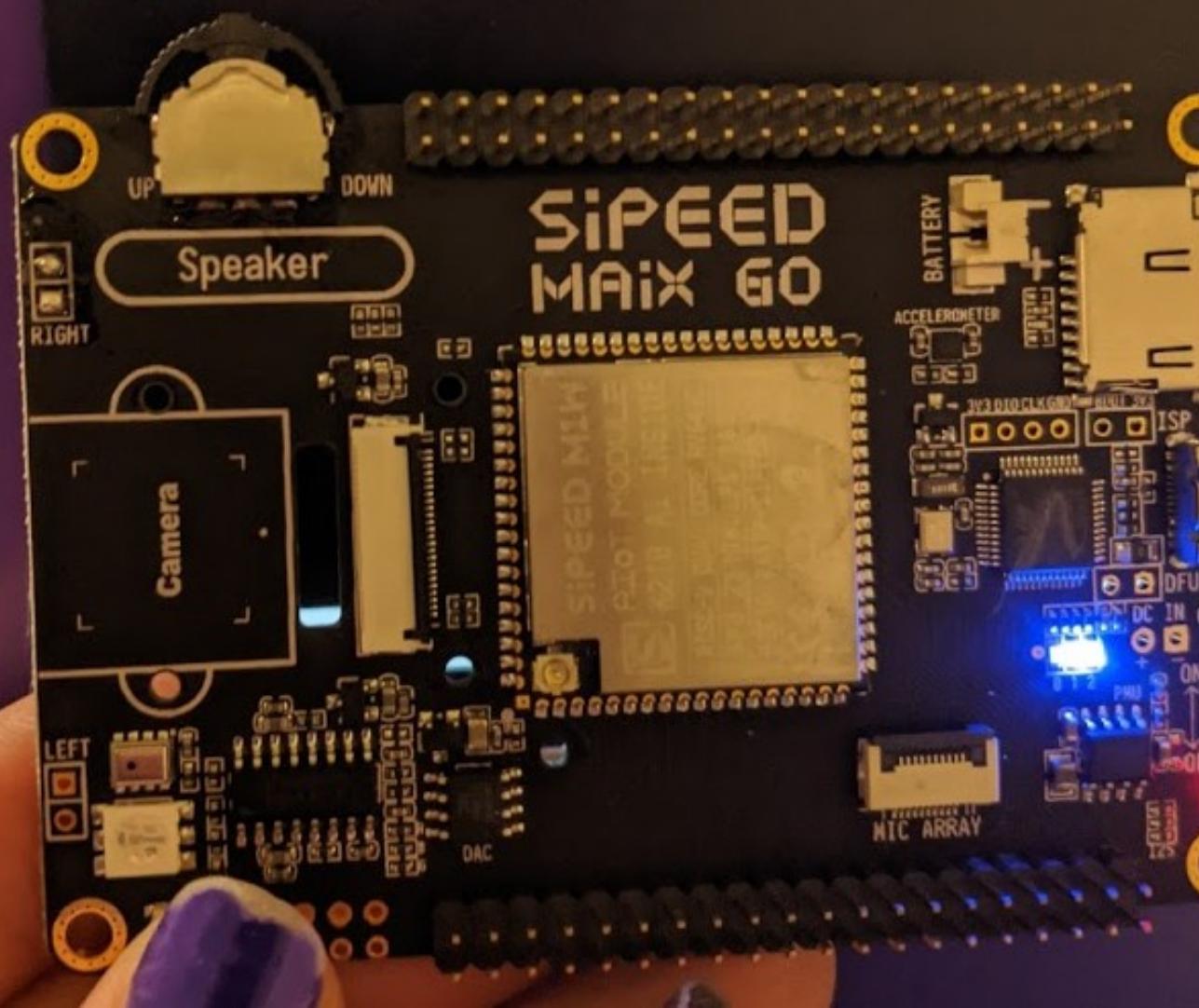
pdp

GtkTerm - /dev/ttyUSB1 115200-8-N-1

pdp7@: File Edit Log Configuration Control signals View Help

pdp7@:

```
/ #  
/ #  
/ #  
/ # uname -a  
Linux k210 5.6.0-rc1vowstar #1 SMP Mon Feb 17 23:36:34 CET 202  
U/Linux  
/ #
```



- Sipeed now has prebuilt Linux 5.6

- <https://twitter.com/SipeedIO/status/1228594799675990016/> ← Tweet



Sipeed

@SipeedIO

The prebuild linux-5.6.0-rc1 image for Maix Boards,
Just burn and try it~
dl.sipeed.com/MAIX/MaixLinux...

```
version 5.6.0-rc1-gdbcd412b (vorstar@yzen) (gcc version 9.2.0 (Buildroot 2020.02-git-g2ceb6f4a3f
n: sifive0 at MMIO 0x0000000038000000 (options '')
bootconsole [sifive0] enabled
not found or empty - disabling initrd
names:
    [mem 0x0000000080000000-0x00000000807fffff]
1 empty
zone start for each node
empty node ranges
    0: [mem 0x0000000080000000-0x00000000807fffff]
setup_node 0 [mem 0x0000000080000000-0x00000000807fffff]
ap is 0x112d
max_distance=0x18000 too large for vmalloc space 0x0
Embedded 12 pages/cpu s18272 r0 d30880 u49152
zonelists, mobility grouping off. Total pages: 2020
command line: earlycon console=ttySIF0
cache hash table entries: 1024 (order: 1, 8192 bytes, linear)
ache hash table entries: 512 (order: 0, 4096 bytes, linear)
    ex_table...
einit: stack off, heap alloc off, heap free:off
6024K/8192K available (918K kernel code, 110K rdata, 166K rodata, 629K init, 91K bss, 2168K res
archical ECU implementation
calculated value of scheduler-enlistment delay is 25 jiffies.
: 0, nr_irqs: 0, preallocated irq: 0
apped 65 interrupts with 2 handlers for 4 contexts.
inner_init_dt: Registering clocksource cpuid[0] hartid [0]
ure: riscv_clocksource: mask: 0xffffffffffff max_cycles: 0x3990be68, max_idle_ns: 881590404 in
lock: 64 bits at 7MHz, resolution 128ns, wraps every 4398046511054ns
: colour dummy device 80x25
ting delay loop (skipped), value calculated using timer frequency.. 15.60 BogoMIPS (lpj=31200)
: default: 4096 minimum: 301
ache hash table entries: 512 (order: 0, 4096 bytes, linear)
int-cache hash table entries: 512 (order: 0, 4096 bytes, linear)
archical SRCU implementation.
ing up secondary CPUs ...
ought up 1 node, 2 CPUs
s: initialized
ure: jiffies: mask: 0xffffffff max_cycles: 0xffffffff, max_idle_ns: 7645041785100000 ns
ash table entries: 16 (order: -2, 1024 bytes, linear)
e K210 SoC sysctl
ure: Switched to clocksource riscv_clocksource
set: timestamp_bits=62 max_order=11 bucket_order=0
0 serial: ttysIF0 at MMIO 0x38000000 (irq = 1, base_baud = 0) is a SiFive UART v0
console: [ttysIF0] enabled
console: [ttysIF0] enabled
bootconsole [sifive0] disabled
bootconsole [sifive0] disabled
get_random_bytes called from 0x00000000800a2128 with erag_init=0
s: mounted
unused kernel memory: 628K
chitecture does not have kernel memory protection.
in/init as init process
e/init as init process
v/init as init process
```

/proc
shell

v1.32.0.git (2020-02-15 12:47:51 CST) hush - the help for a list of built-in commands.

e -a
one) 5.6.0-rc1-g9dbcd412b #10 SMP Sat Feb 15 15:30:

v	etc	proc	root	sbin	sys	tmp	usr
chown	grep	ls	printenv	set:			
cp	hush	mkdir	ps	sh			
date	init	mknod	pwd	sleep			
dmesg	kill	more	rm	stty			
echo	link	mount	rmdir	tou			
false	ln	mv	sed	true			

usr/bin

env	id	seq	tty	wall
expr	last	tail	uniq	wc
find	printf	tcc	unlink	who
head	readlink	tee	users	yes
hostid	realpath	test	w	

Coming in 2020

- Microchip PolarFire SoC FPGA
 - Hard RISC-V with FPGA fabric... like the Xilinx Zync for ARM
- OpenHW Core-V SoC which is similar to NXP iMX with RISC-V instead of ARM!
 - [“OpenHW Group Unveils CORE-V Chassis SoC Project, Building on PULP Project IP”](#)

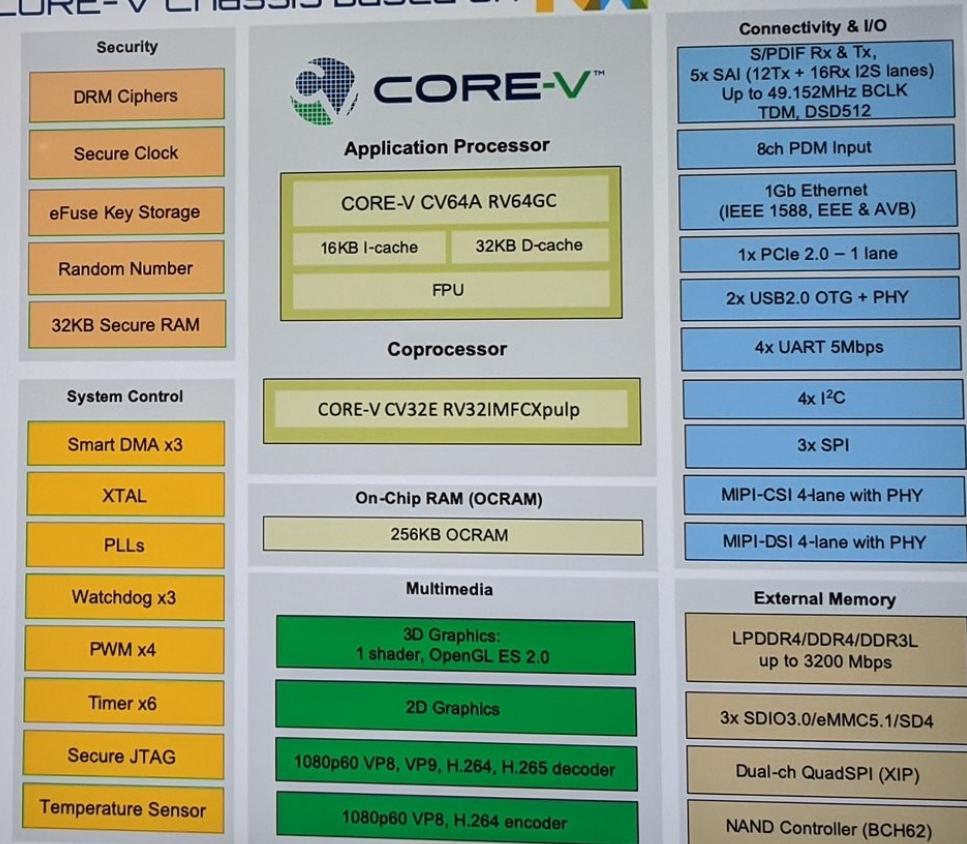


CORE-V™ chassis – tapeout 2H 2020



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

10 & 11 December 2019

17

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 FOSS North 2021?
- Interested in working together?
 - drew@oshpark.com / Twitter: [@pdp7](https://twitter.com/pdp7)
 - create a mailing list?

Slides: <https://github.com/pdp7/talks/blob/master/rv-israel.pdf>

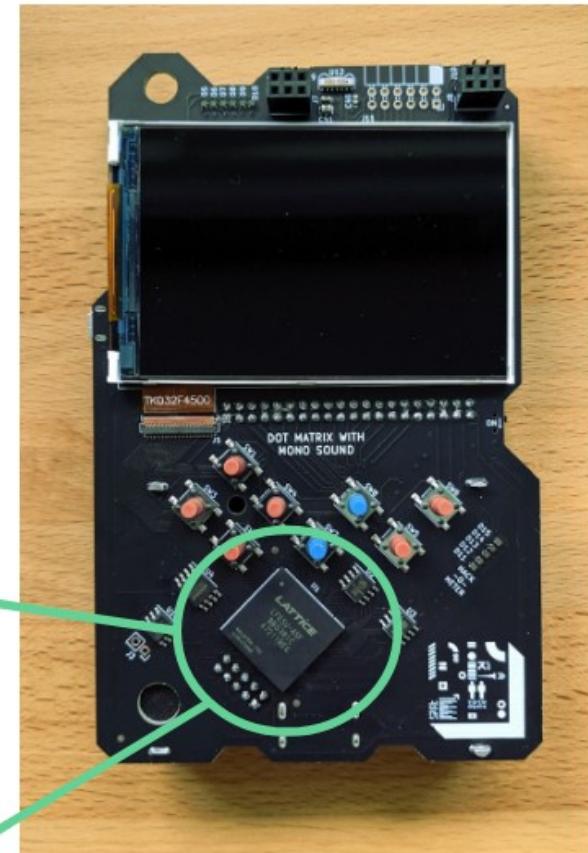
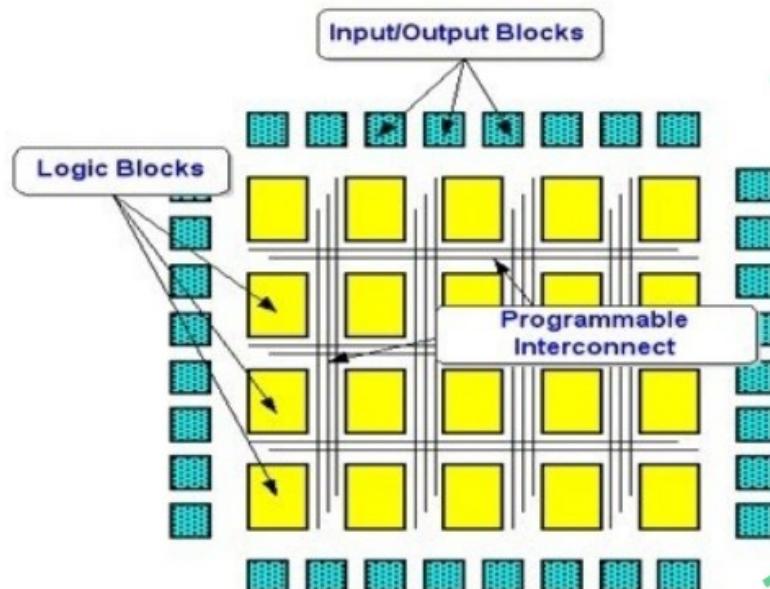
Section:
Open Source FPGA tools



- 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 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



browse > congress > 2015 > event

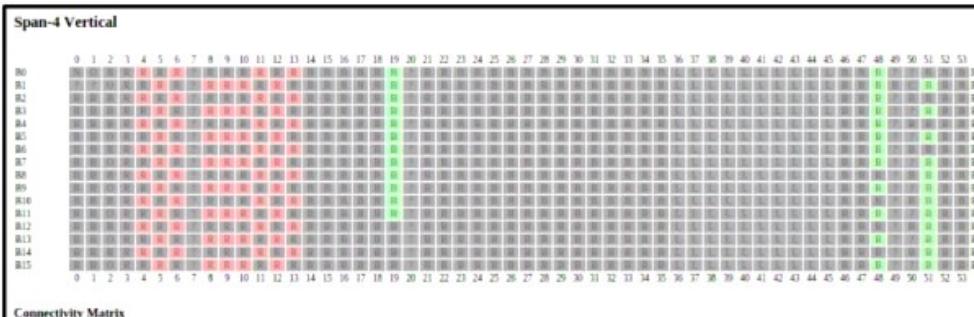
A Free and Open Source Verilog-to-Bitstream Flow for iCE40 FPGAs



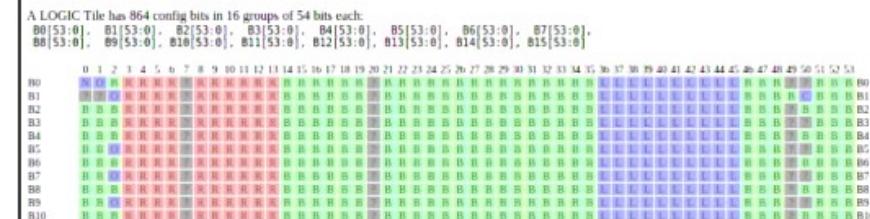
by [Clifford](#)

(2 17)	(3 17)	(4 17)	(5 17)	(6 17)
LOGIC Tile (2 16)	RAMT Tile (3 16)	LOGIC Tile (4 16)	LOGIC Tile (5 16)	LOGIC Tile (6 16)
LOGIC Tile (2 15)	RAMT Tile (3 15)	LOGIC Tile (4 15)	LOGIC Tile (5 15)	LOGIC Tile (6 15)
LOGIC Tile (2 14)	RAMT Tile (3 14)	LOGIC Tile (4 14)	LOGIC Tile (5 14)	LOGIC Tile (6 14)

Some screenshots from IceStrom Docs:



Configuration Bitmap



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

Project Trellis and nextpnr FOSS FPGA flow for the Lattice ECP5

Project Trellis & nextpnr

FOSS Tools for ECP5 FPGAs

David Shah
@fpga_dave
Symbiotic EDA || Imperial College London

FOSDEM 19
org



Open Source toolchains for FPGAs

- Project X-Ray & SymbiFlow for Xilinx Series 7
 - Timothy 'mithro' Ansell: "Xilinx Series 7 FPGAs Now Have a Fully Open Source Toolchain!" (*almost*)
 - youtube.com/watch?v=EHePto95qoE



- Multi-platform
- All FPGAs **big and small**

17th November 2019

Open Source and FPGAs

- Hackspace Magazine column about how 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/

MAKE | BUILD | HACK | CREATE **132 PAGES** OF MAKING

HackSpace

TECHNOLOGY IN YOUR HANDS

hsmag.cc | January 2020 | Issue #26

WHAT 3D PRINTER?

Find the ultimate replicator for 2020

CIRCUIT PYTHON

SOLDERING WITH GAS

PICKING AN IMPACT DRIVER

SEWING MACHINES

Building a kiln

Melting glass with a Raspberry Pi

Drew Fustini

@pdpf

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 Foundation. Drew designs circuit boards for OSH Park, a PCB manufacturing service, and maintains the Adafruit BeagleBone Python library.

FPGA

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. Is 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 from 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

The Lattice ECP5 FPGA is capable of more advanced features than the iCE40, and it's easier to get started with 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.cc/GOAQnR), and watch Tim Ansell's Supercon talk to learn about the exciting future of open-source FPGA tools (hsmag.cc/kY5IPD). □

The rise of the FPGA

Reconfigure your chips to suit your project

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's easier to get started with 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.cc/GOAQnR), and watch Tim Ansell's Supercon talk to learn about the exciting future of open-source FPGA tools (hsmag.cc/kY5IPD). □

HackSpace

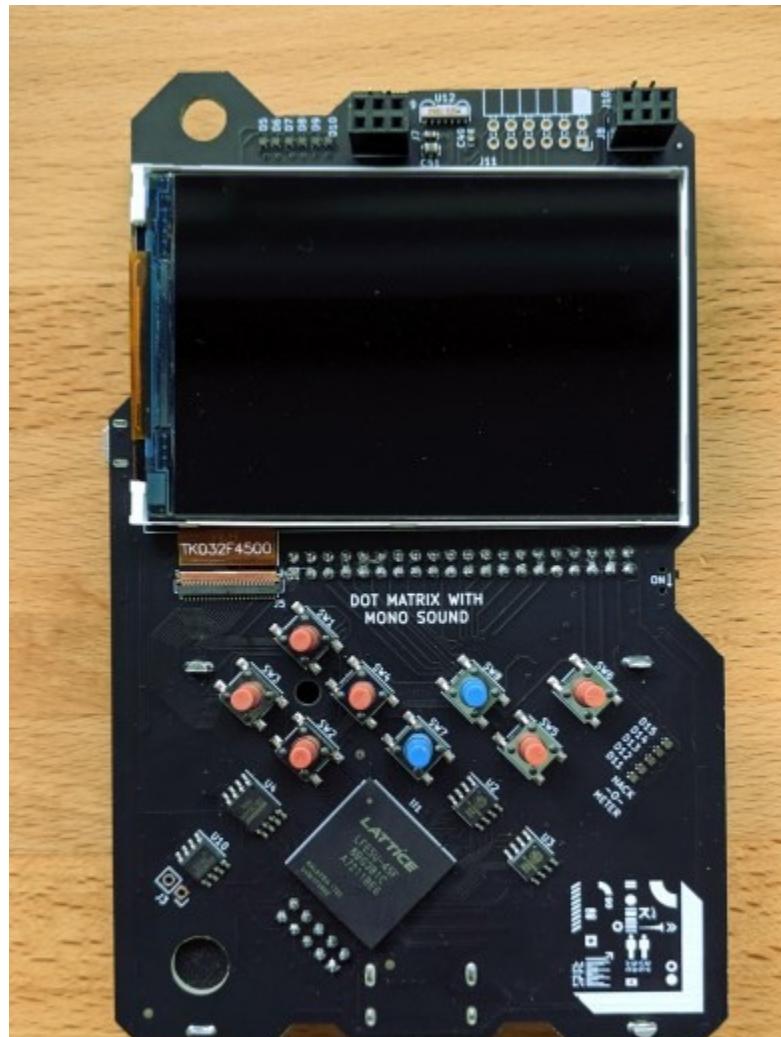
23

Slides: <https://github.com/pdp7/talks/blob/master/rv-israel.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”



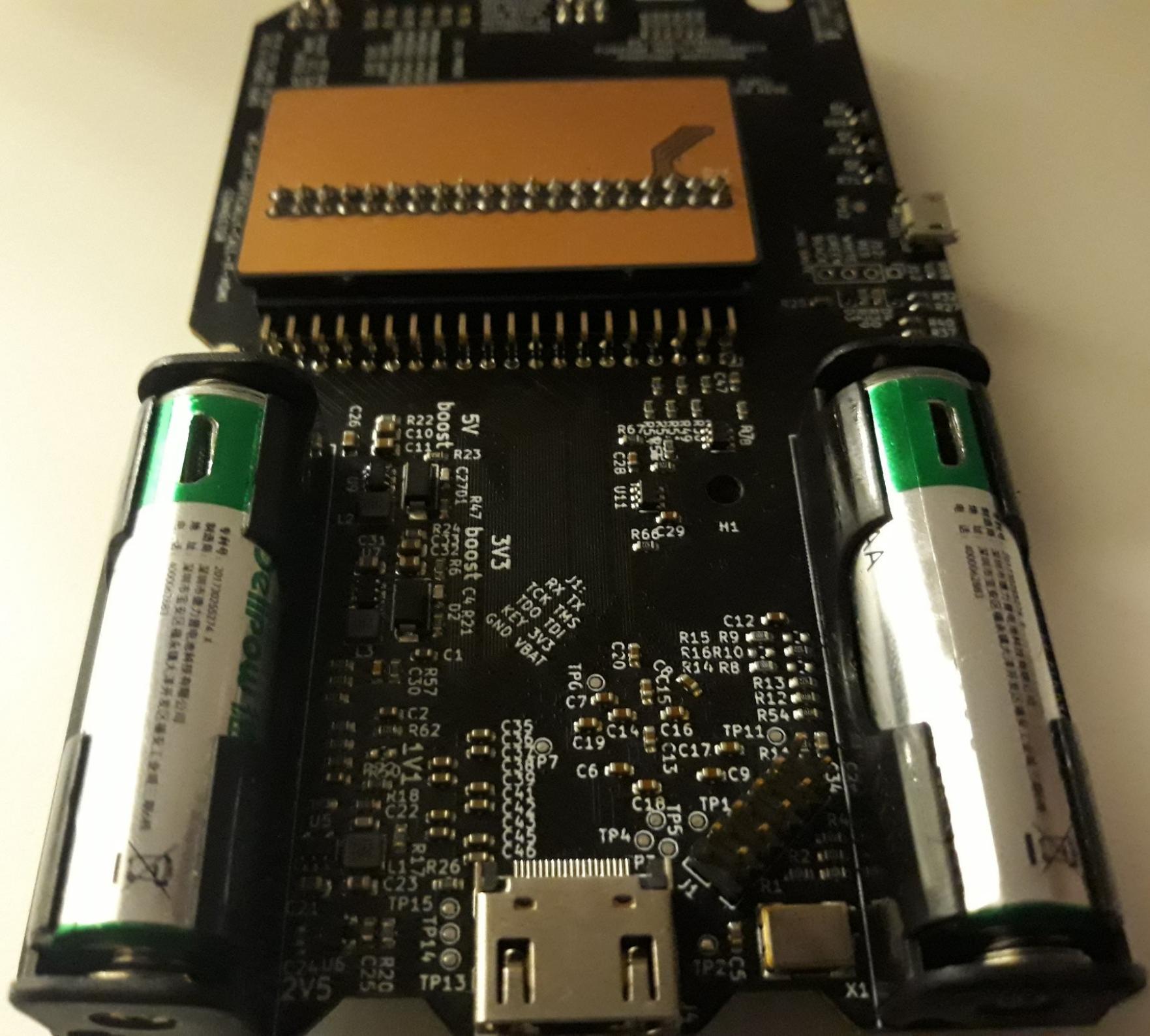
“Team Linux on Badge”

- 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 (@QwertyEmdedded), 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!



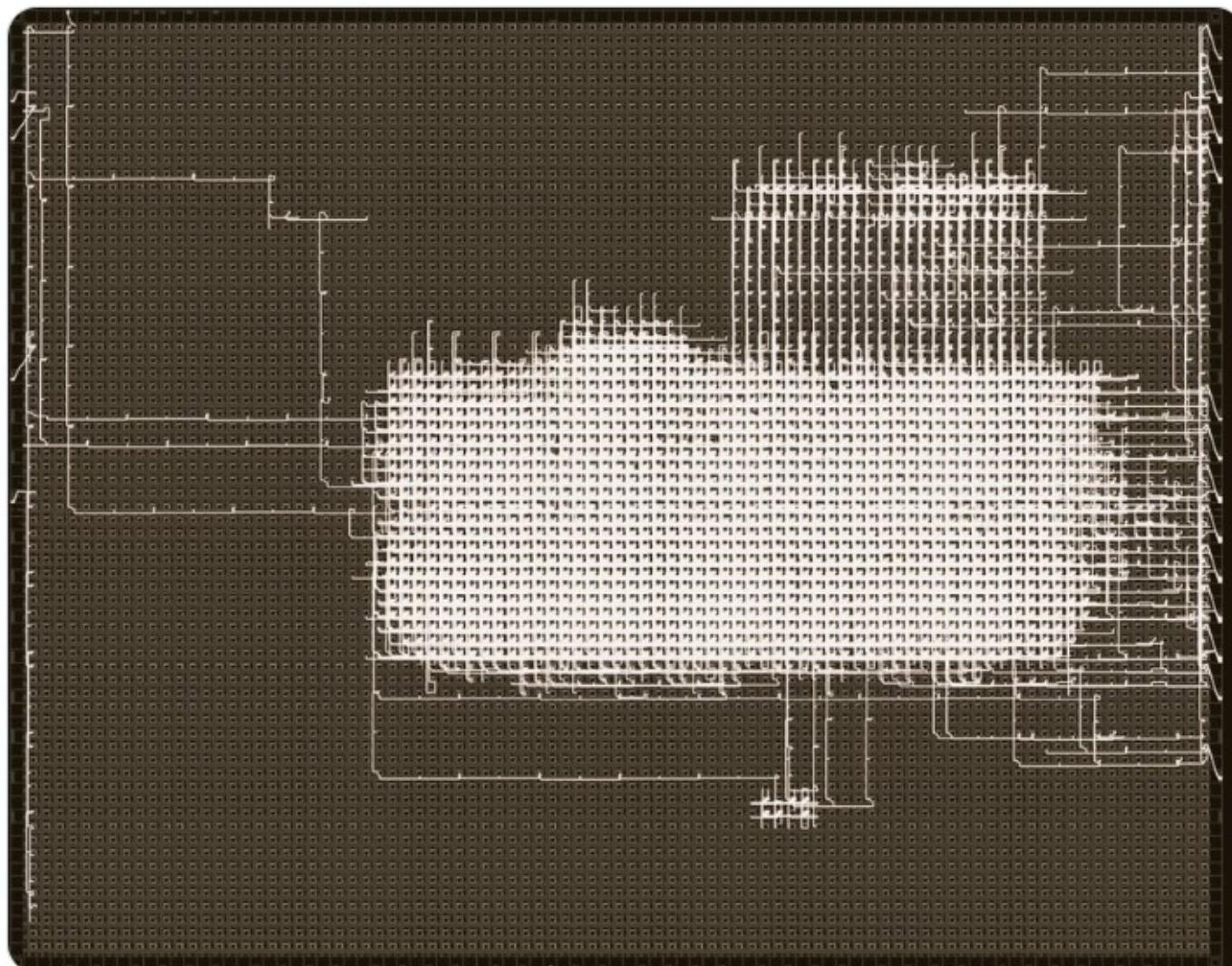




OSS FPGA & EDA tools
@ico_TC



This is how a Linux capable core looks like on an
FPGA. [#nextpnratwork](#)



Designing Hardware in Python?

- Yes!
- “Using Python for creating hardware to record FOSS conferences!”
- Tim “mithro” Ansell
- youtube.com/watch?v=MkVX_mh5dOU

Why is this powerful?

- Python is a **powerful** language
- Python is a **productive** language
- Can generate exact Verilog as needed

hdmi2usb.tv

@mithro

j.mp/pyhw-lca2017



Using Python for creating hardware to record FOSS conferences!

1,041 views • Jan 19, 2017

16 0 SHARE SAVE ...

Up next



AUTOPLAY

Python HDL?

You can think of Migen as
*"an easy way
to generate Verilog"*

Blink Led - Migen

```
class Blinker(Module):
    def __init__(self, led, cycles=15000000):
        counter = Signal(max=cycles)
        self.sync += counter.eq(counter - 1)
        self.comb += led.eq(counter[-1])
```

Most significant bit
connected to LED signals

What is Migen?

```
-- Libraries imports
library ieee;
use ieee.std_logic_1164.all;

-- Module interface description
entity my_module is
    port(
        clk : in std_logic;
        o   : out std_logic
    );
end entity;

-- Module architecture description
architecture rtl of my_module is
    signal d : std_logic;
    signal q : std_logic;
begin
    -- Combinatorial logic
    o <= q;
    d <= not q;

    -- Synchronous logic
    process(clk)
    begin
        if rising_edge(clk) then
            d <= q
        end if;
    end process
end rtl;
```

VHDL

*An alternative HDL
based on Python*



BASICS

```
from migen import *

class MyModule(Module):
    def __init__(self):
        self.o = Signal()

    # ## #

    d = Signal()
    q = Signal()

    # combinatorial logic
    self.comb += [
        self.o.eq(q),
        d.eq(~q)
    ]

    # synchronous logic
    self.sync += d.eq(q)
```

Migen

Enjoy Digital
Do

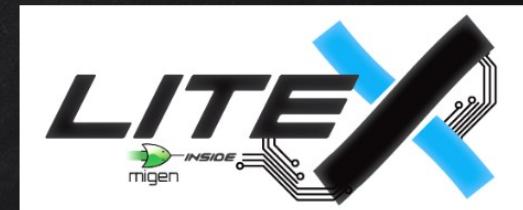
- *SoC can be create with VHDL or Verilog or System Verilog. That's what is generally used in the industry but very verbose and error prone. (Not a problem if you have the money and the number of developers...).*
- *We already that Migen can be use to create digital designs. LiteX provides a higher level of abstraction.*



FPGAs FOR CUSTOM SYSTEM ON CHIP



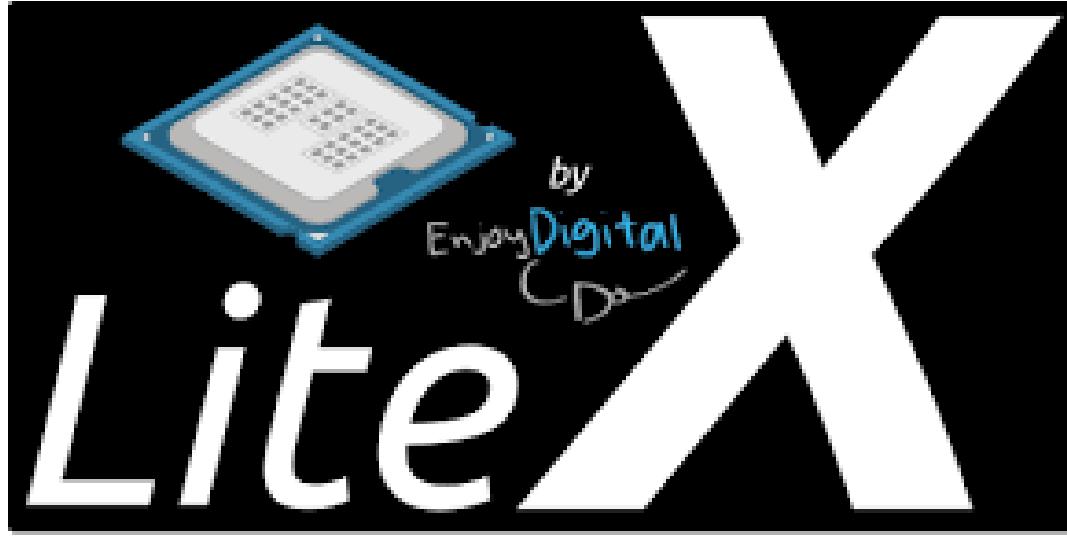
+



=

Build your hardware easily!





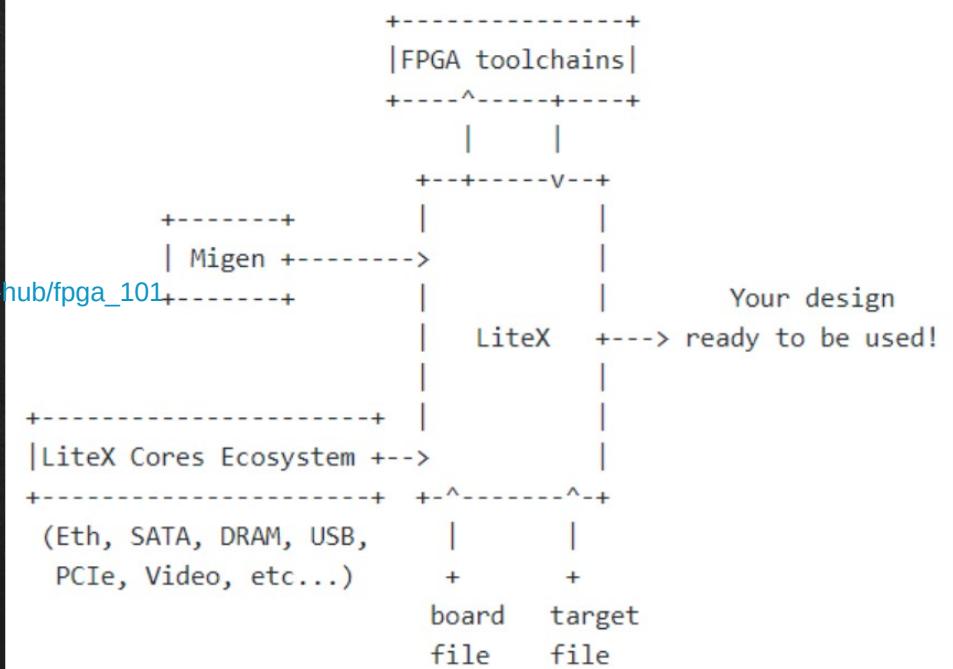
Build your hardware, easily!

- LiteX used to build cores, create SoCs and full FPGA designs.
- LiteX is based on Migen
- Migen lets you do FPGA design in Python!
- <https://github.com/enjoy-digital/litex>

- *LiteX automates parts of the SoC design (buses, registers, software) and allow being more efficient.*
- *It provides most of the base elements required in a modern SoC (buses, streams, fifos, arbiters, muxes, etc...)*
- *It is compatible with all the LiteX core ecosystem (DRAM, Ethernet, PCIe, SATA, USB controllers...)*
- *Can be found at:*
github.com/enjoy-digital/litex

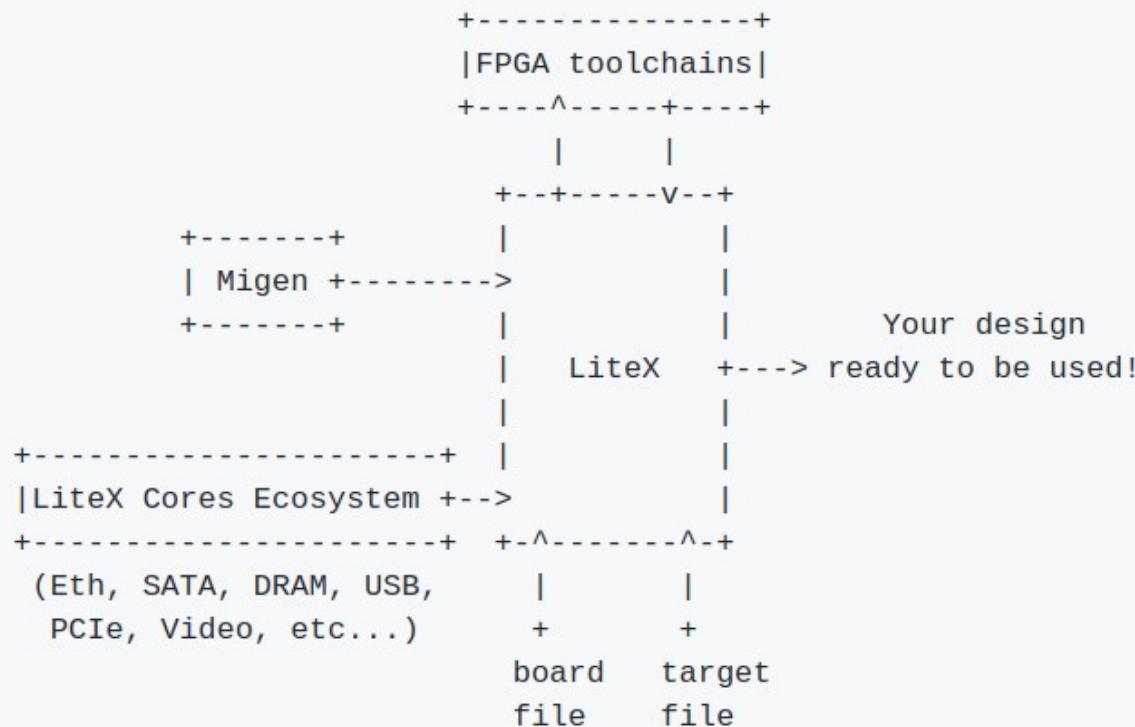


FPGAs FOR CUSTOM SYSTEM ON CHIP



Enjoy Digital
Do

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

- VexRiscv: 32-bit Linux Capable RISC-V CPU
 - SoC built using VexRiscv core and LiteX modules like LiteDRAM, LiteEth, LiteSDCard, ...
 - github.com/litex-hub/linux-on-litex-vexriscv



- upstream support for Hackaday Supercon badge:
 - <https://github.com/litex-hub/litex-boards/pull/31>

[litex-hub / litex-boards](#)

Unwatch 7 Unstar 17 Fork 24

Code Issues 2 Pull requests 1 Actions Projects 0 Wiki Security Insights

add the Hackaday Supercon ECP5 badge #31

Merged enjoy-digital merged 1 commit into `litex-hub:master` from `pdp7:master` 21 days ago

Conversation 18 Commits 1 Checks 1 Files changed 2 +461 -0

 pdp7 commented 22 days ago • edited

Contributor + ...

Add the [Hackaday Supercon 2019 badge](#) which has an ECP5 FPGA.

These changes are from a [fork](#) by Michael Welling (@mwelling)

During Supercon, we tried two approaches:

- use the built-in 16MB QSPI SRAM
- use add-on cartridge with 32MB SDRAM by Jacob Creedon

We were not able to get the QSPI SRAM working so I've removed those changes, and I have just added the changes that are needed to boot Linux with the 32MB SDRAM.

In addition to @mwelling, thank you to Jacob Creedon (@jcreedon), @gregdavill, Tim Ansell (@mithro), and Sean Cross (@xobs) who all helped get Linux working on this badge.

Reviewers
No reviews

Assignees
No one assigned

Labels
None yet

Projects
None yet

Milestone

- upstream support for Hackaday Supercon badge:
 - <https://github.com/litex-hub/litex-boards/pull/31>

Merged add the Hackaday Supercon ECP5 badge #31 Changes from all commits ▾ File filter... ▾ Jump to... ▾ ⚙ Review changes ▾

215 litex_boards/partner/platforms/hadbadge.py

```

@@ -0,0 +1,215 @@
+ from litex.build.generic_platform import *
+ from litex.build.lattice import LatticePlatform
+
+ # IOs -----
+
+ _io = [
+     ("clk8", 0, Pins("U18"), IOStandard("LVCMOS33")),
+     ("programn", 0, Pins("R1"), IOStandard("LVCMOS33")),
+     ("serial", 0,
+         Subsignal("rx", Pins("U2"), IOStandard("LVCMOS33"), Misc("PULLMODE=UP")),
+         Subsignal("tx", Pins("U1"), IOStandard("LVCMOS33")),
+     ),
+     ("led", 0, Pins("E3 D3 C3 C4 C2 B1 B20 B19 A18 K20 K19"), IOStandard("LVCMOS33")), # Anodes
+     ("led", 1, Pins("P19 L18 K18"), IOStandard("LVCMOS33")), # Cathodes via FET
+     ("usb", 0,
+         Subsignal("d_p", Pins("F3")),
+         Subsignal("d_n", Pins("G3")),
+         Subsignal("pullup", Pins("E4")),
+         Subsignal("vbusdet", Pins("F4")),
+         IOStandard("LVCMOS33")
+     ),
+     ("keypad", 0,
+         Subsignal("left", Pins("G2"), Misc("PULLMODE=UP"))
+     )
+ ]

```

- upstream support for Hackaday Supercon badge:
 - <https://github.com/litex-hub/litex-boards/pull/31>

Merged add the Hackaday Supercon ECP5 badge #31 Changes from all commits ▾ File filter... ▾ Jump to... ▾ ⚙ ▾ Review changes ▾

246 litex_boards/partner/targets/hadbadge.py

Viewed ...

```
@@ -0,0 +1,246 @@
+#!/usr/bin/env python3
+
+ # This file is Copyright (c) 2018-2019 Florent Kermarrec <florent@enjoy-digital.fr>
+ # This file is Copyright (c) 2018 David Shah <dave@ds0.me>
+ # License: BSD
+
+ import argparse
+ import sys
+
+ from migen import *
+ from migen.genlib.resetsync import AsyncResetSynchronizer
+
+ from litex_boards.platforms import hadbadge
+
+ from litex.soc.cores.clock import *
+ from litex.soc.integration.soc_sdram import *
+ #from litex.soc.integration.soc_core import *
+ from litex.soc.integration.builder import *
+
+ #from .spi_ram_dual import SpiRamDualQuad
+
+ from litedram import modules as litedram_modules
+ from litedram.phy import GENSDRPHY
```

[Code](#)[Issues 17](#)[Pull requests 2](#)[Actions](#)[Security](#)[Insights](#)

add the Hackaday Supercon ECP5 badge #68

Mergedenjoy-digital merged 1 commit into [litex-hub:master](#) from [pdp7:master](#)  21 days ago[Conversation 2](#)[Commits 1](#)[Checks 0](#)[Files changed 1](#)

pdp7 commented 22 days ago

Contributor

+  ...

Add the [Hackaday Supercon 2019 badge](#) which has an ECP5 FPGA.

These changes are from [a fork](#) by Michael Welling (@**mwellings**)

During Supercon, we tried two approaches:

- use the built-in 16MB QSPI SRAM
- use add-on cartridge with 32MB SDRAM by Jacob Creedon

We were not able to get the QSPI SRAM working so I've removed those changes, and I have just added the changes that are needed to boot Linux with the 32MB SDRAM.

In addition to @**mwellings**, thank you to Jacob Creedon (@**jcreedon**), @**gregdavill**, Tim Ansell

Merged

add the Hackaday Supercon ECP5 badge #68

Changes from all commits ▾ File filter... ▾ Jump to... ▾ ⚙

▼ 13 [make.py] ↗

```
@@ -160,6 +160,16 @@ def __init__(self):  
160      160          def load(self):  
161      161              os.system("ujprog build/ulx3s/gateware/top.svf")  
162      162  
163 + # HADBadge support -----  
164 +  
165 + class HADBadge(Board):  
166 +     def __init__(self):  
167 +         from litex_boards.targets import hadbadge  
168 +         Board.__init__(self, hadbadge.BaseSoC, {"serial"})  
169 +  
170 +     def load(self):  
171 +         os.system("dfu-util --alt 2 --download build/hadbadge/gateware/top.bit --reset")  
172 +  
163      173      # OrangeCrab support -----  
164      174  
165      175      class OrangeCrab(Board):  
@@ -209,6 +219,7 @@ def load(self):  
209      219          # Lattice
```

[Code](#)[Issues 21](#)[Pull requests 3](#)[Actions](#)[Projects 0](#)[Wiki](#)[Security](#)[In](#)

add 32MB SDRAM for hadbadge #97

Mergedenjoy-digital merged 1 commit into [enjoy-digital:master](#) from [pdp7:master](#)  21 days ago[Conversation 2](#)[Commits 1](#)[Checks 0](#)[Files changed 1](#)

pdp7 commented 22 days ago

Contributor



...

Add AS4C32M8SA-7TCN 32MB SDRAM used on cartridge PCB by Jacob Creedon (@jcreedon) for the [Hackaday Supercon 2019 badge](#) which has an ECP5 FPGA.

These changes are from [a fork](#) by Michael Welling (@mwelling)

In addition to @mwelling, thank you to Jacob Creedon (@jcreedon), @gregdavill, Tim Ansell (@mithro), and Sean Cross (@xobs) who all helped get Linux working on this badge.

KiCad design files by @jcreedon for the SDRAM cartridge are [available on GitHub](#).

There is also a [shared project](#) to order the SDRAM cartridge PCB.

Refer to [my blog post](#) for more information.

Merged

add 32MB SDRAM for hadbadge #97

Changes from all commits ▾

File filter... ▾

Jump to... ▾



9 litedram/modules.py



```
@@ -190,6 +190,15 @@ class AS4C32M16(SDRAMModule):
190      technology_timings = _TechnologyTimings(tREFI=64e6/8192, tWTR=(2, None), tCCD=(1,
191      speedgrade_timings = {"default": _SpeedgradeTimings(tRP=18, tRCD=18, tWR=12, tRFC=
192
193 +     class AS4C32M8(SDRAMModule):
194 +         memtype = "SDR"
195 +         # geometry
196 +         nbanks = 4
197 +         nrows  = 8192
198 +         ncols   = 1024
199 +         # timings
200 +         technology_timings = _TechnologyTimings(tREFI=64e6/8192, tWTR=(2, None), tCCD=(1,
201 +         speedgrade_timings = {"default": _SpeedgradeTimings(tRP=20, tRCD=20, tWR=15, tRFC=
193
194     # DDR -----
195
```



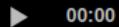
- Opened GitHub issue:
 - optimize performance on Hackaday Badge #35
 - <https://github.com/litex-hub/litex-boards/issues/35>
- Now 10x faster!
 - <https://asciinema.org/a/Pcm3vd1BEdEKY9srYX6MsNfCE>
 - Thanks to [enjoy-digital](#)



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: 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!]

[Code](#)[Issues 2](#)[Pull requests 1](#)[Actions](#)[Projects 0](#)[Wiki](#)[Security](#)[Insights](#)

optimize performance on Hackaday Badge #35

[Edit](#)[New issue](#)[Open](#)

pdp7 opened this issue 17 days ago · 7 comments



pdp7 commented 17 days ago

Contributor



...

Assignees

No one assigned

Labels

None yet

Projects

None yet

Milestone

No milestone

Notifications

Customize

[Unsubscribe](#)

You're receiving notifications because
you're watching this repository.

[Related to comments in PR #31 \(comment\)](#)

The performance of the ECP5 Hackaday Badge with 32MB SDRAM is "painfully" slow.

@mithro suggested there could be some issue with the configuration.

@enjoy-digital has attempted some optimizations:

[#31 \(comment\)](#)

With [enjoy-digital/litedram@ 34e6c24](#) and [enjoy-digital/litex@ fa22d6a](#) we have a ~10% boot time speedup on designs using SDRAM:

- De0Nano: 94.6.s to 84.6s
- ULX3S: 75.9s to 68.2s

On Arty with DDR3 the gain is effect more limited: 8.7s to 8.4s. That would be interesting to test this on the badge.

I will measure if the boot time improve

Open

optimize performance on Hackaday Badge #35

pdp7 opened this issue 17 days ago · 7 comments



pdp7 commented 15 days ago • edited ▾

Contributor

Author

+ 😊 ...

@enjoy-digital WOW! much faster! It gets to login in 28 seconds (previous version was 258 seconds).

Recording:

<https://asciinema.org/a/Pcm3vd1BEdEKY9srYX6MsNfCE>

Text:

```
pdp7@x1:~/dev/enjoy/linux-on-litex-vexriscv$ lxterm --images=images.json /dev  
[LXTERM] Starting....  
lBIOS CRC passed (561ab1e2)
```

```
Migen git sha1: 063188e  
LiteX git sha1: -----
```

```
-===== SoC =====  
CPU: VexRiscv @ 48MHz  
ROM: 32KB  
SRAM: 4KB  
Lo:
```



enjoy-digital committed 15 days ago

1 parent 2317519

commit 39ce39a298f5



Showing **1 changed file** with **5 additions** and **3 deletions**.

8 litex/soc/integration/soc_sdram.py

@@ -26,12 +26,13 @@ class SoCSDRAM(SoCCore):

26 26 }

```
27      27      csr_map.update(SoCCore.csr_map)
```

28

```
-      def __init__(self, platform, clk_freq, l2_size=8192, **kwargs):
```

```
29 +     def __init__(self, platform, clk_freq, l2_size=8192, l2_data_width=128, **kwargs):
```

```
30      30          SoCCore.__init__(self, platform, clk_freq, **kwargs)
```

```
31      31          if not self.integrated_main_ram_size:
```

32 32 if self.cpu_type is not None and self.csr_data_width > 32:

```
raise NotImplementedException("BIOS supports SDRAM initialization only for c
```

34 - self.l2_size = l2_size

34 + self.l2_size = l2_size

35 + self.l2_data_width = l2_data_width

35 36

36 37 self, s dram phy = []

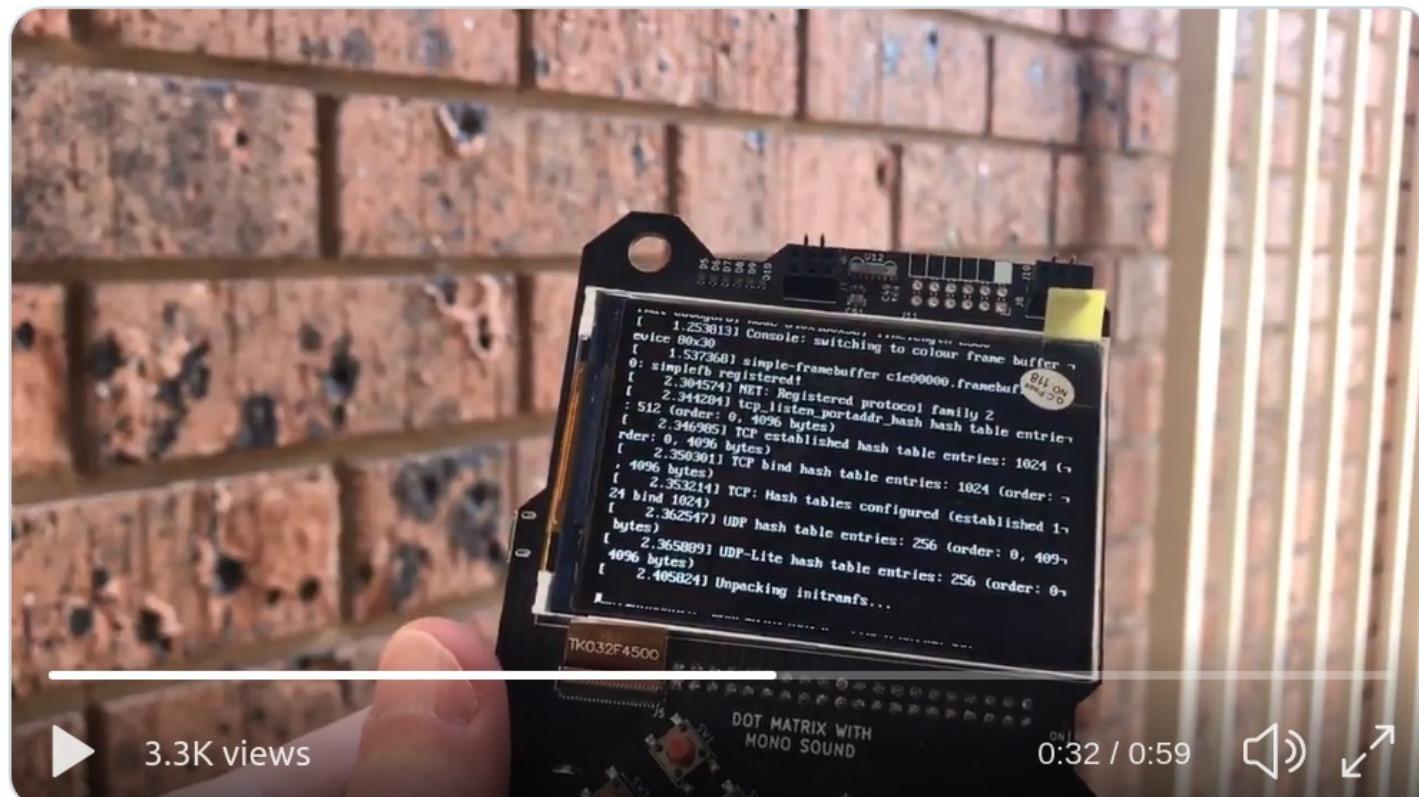
```
37      38          self.wb_sdram_ifs = []
```

- Greg Davill got the screen working with LiteVideo!
 - twitter.com/GregDavill/status/1231082623633543168



A screenshot of a Twitter post from user @GregDavill. The profile picture shows a man with short hair. The tweet text reads: "Now you can enjoy watching Linux boot while outside!! 😊". The timestamp is 8:05 AM · Mar 10, 2020.

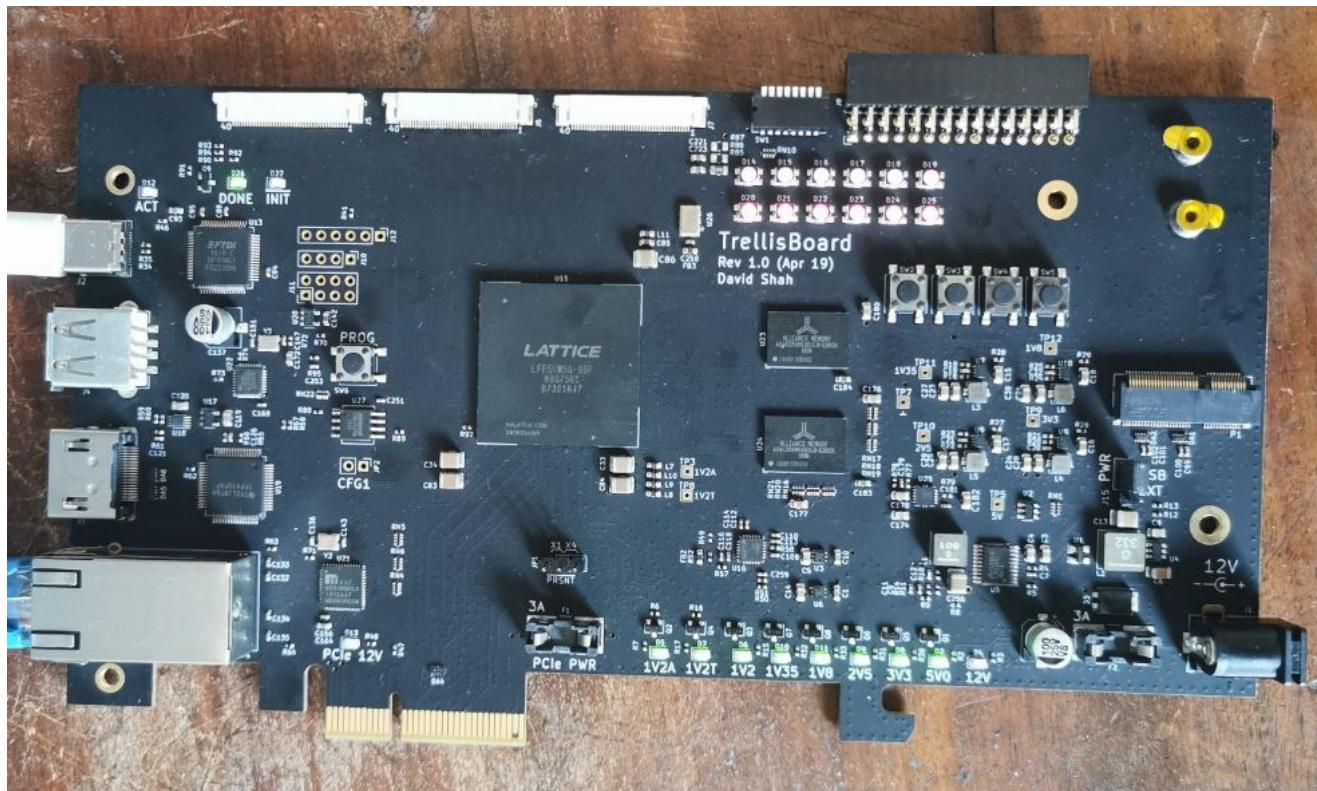
No PC tether required.



Slides: <https://github.com/pdp7/talks/blob/master/rv-israel.pdf>

Open Source boards with ECP5 FPGA (can run Linux)

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



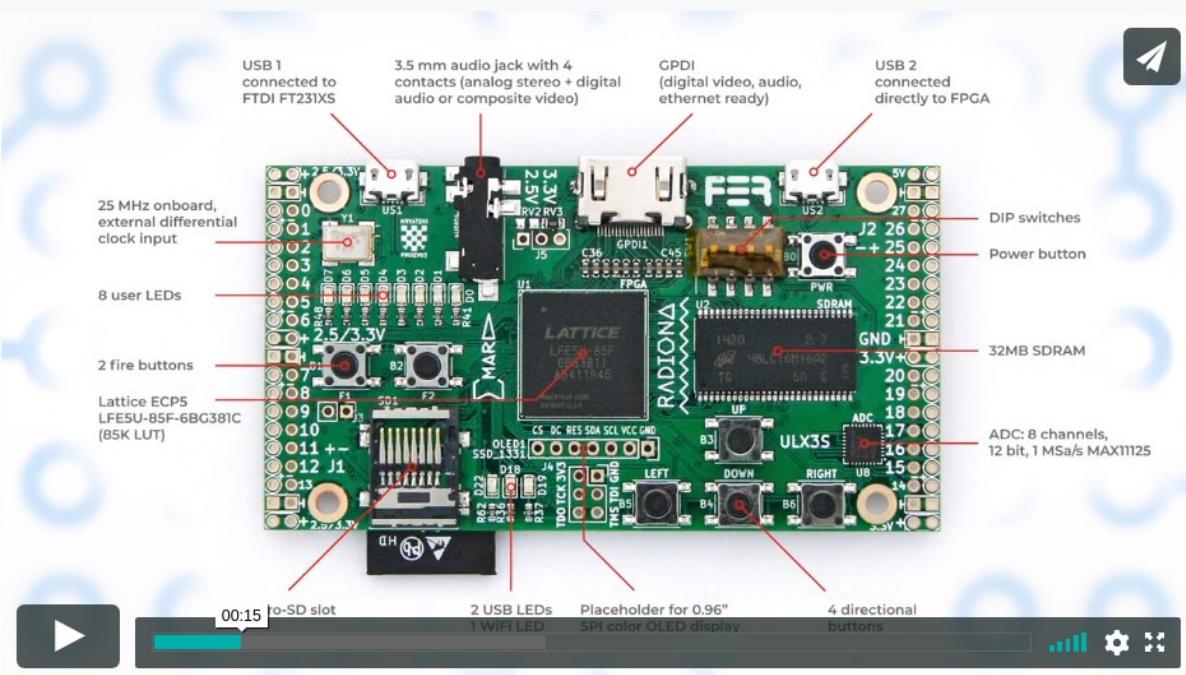
Open Source ECP5 boards

- Radiona.org ULX3S

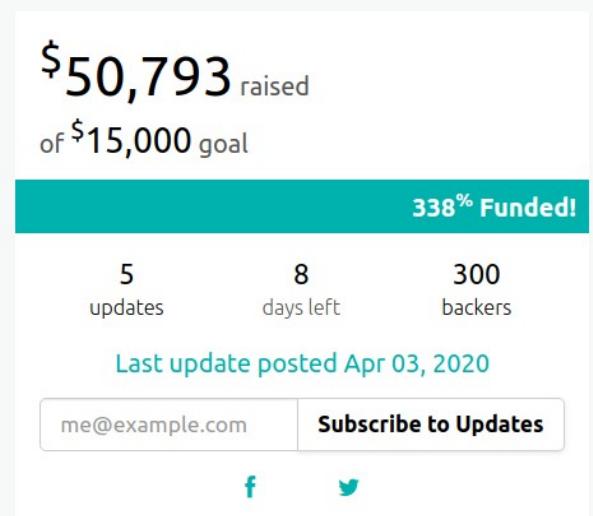
- <https://www.crowdsupply.com/radiona/ulx3s>

crowdsupply.com/radiona/ulx3s

CROWD SUPPLY BROWSE LAUNCH ABOUT US Search 

A detailed diagram of the ULX3S board showing various components and their connections. Labels include:

- USB 1 connected to FTDI FT231XS
- 3.5 mm audio jack with 4 contacts (analog stereo + digital audio or composite video)
- GPDI (digital video, audio, ethernet ready)
- USB 2 connected directly to FPGA
- 25 MHz onboard, external differential clock input
- 8 user LEDs
- 2 fire buttons
- Lattice ECP5 LFE5U-85F-6BG381C (85K LUT)
- 00:15 SD slot
- 2 USB LEDs
- Placeholder for 0.96" SPI color OLED display
- 4 directional buttons
- DIP switches
- Power button
- 32MB SDRAM
- ADC: 8 channels, 12 bit, 1 MSa/s MAX11125

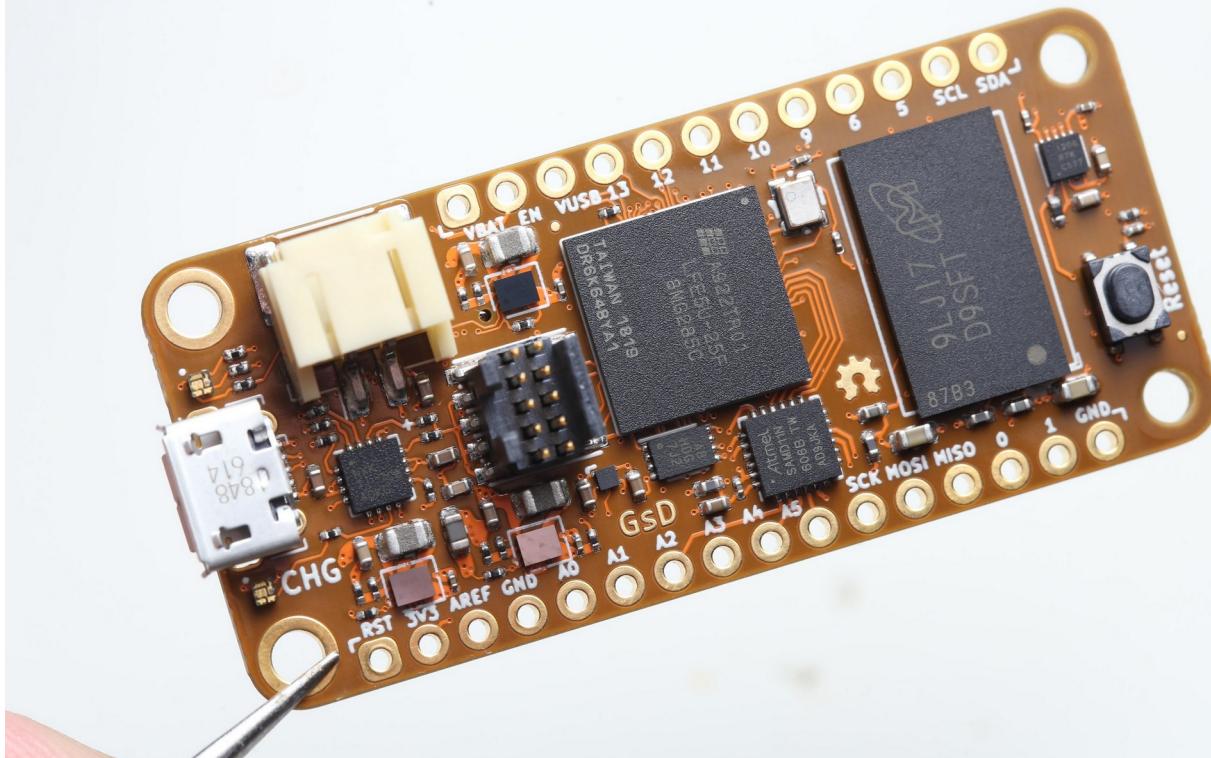
**\$50,793 raised** of **\$15,000 goal**
338% Funded!
5 updates 8 days left 300 backers
Last update posted Apr 03, 2020
me@example.com **Subscribe to Updates**
 

PMOD Set \$36
A set of three PMODs for ULX3X: one dual-USB board with support for IR send and receive, one Digital Video board for HDMI input or as a

Recent Updates View all 5 updates.

Open Source ECP5 boards

- Lattice ECP5 FPGA in Adafruit Feather form factor and 128MB DDR RAM:
 - Orange Crab by Greg Davill
 - <https://github.com/gregdavill/OrangeCrab>
 - <https://groupgets.com/campaigns/710-orangecrab>





Greg @ #36c3
@GregDavill



Replying to @mithro @pdp7 and 2 others

Done. 😊

```
File Edit View Terminal Tabs Help
SRAM:      4KB
L2:        8KB
MAIN-RAM: 131072KB

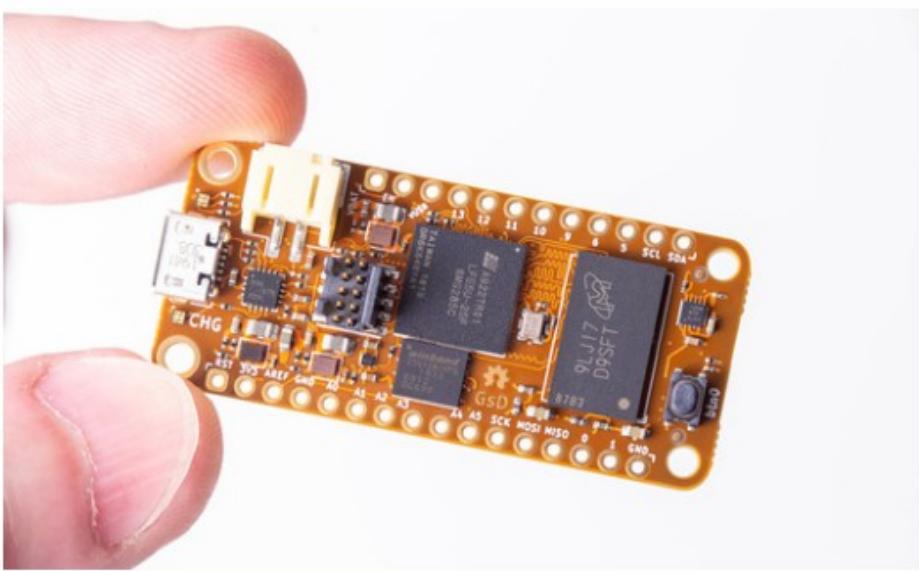
----- Initialization -----
Initializing SDRAM...
SDRAM now under software control
Read leveling:
m0, b0: |11100000| delays: 01+-01
best: m0, b0 delays: 01+-01
m1, b0: |11100000| delays: 01+-01
best: m1, b0 delays: 01+-01
SDRAM now under hardware control
Memtest OK

----- Boot -----
Booting from serial...
Press Q or ESC to abort boot completely.
sL5DdSMmkekro
[LXTERM] Received firmware download request from the device.
[LXTERM] Uploading buildroot/Image to 0xc0000000 (4545524 bytes)...
[LXTERM] Upload complete (85.6KB/s).
[LXTERM] Uploading buildroot/rootfs.cpio to 0xc0800000 (8029184 bytes)...
[ 0:43 ] 1.2K views => | 98%
```



Open Source ECP5 boards

- Orange Crab by Greg Davill
 - <https://groupgets.com/campaigns/710-orangecrab>



A thumbnail image showing a close-up of the OrangeCrab board being held between two fingers. The board is orange and populated with various electronic components, including a central microcontroller and several surface-mount chips.

OrangeCrab

Single-unit price: **\$99.00** + shipping

Campaign scheduled end date: Mon, 13 Apr 2020 16:19:00 PDT

Backers: 174

Target: 200

Funded: 286

Join this Buy

A row of social media sharing icons: Facebook, Twitter, Pinterest, and Email.

Want to learn FPGAs? Try Fomu!

- workshop.fomu.im
- crowdsupply.com/sutajio-kosagi/fomu

- Fits in USB port

- RGB LED

- Learn:

- MicroPython
- Verilog
- LiteX

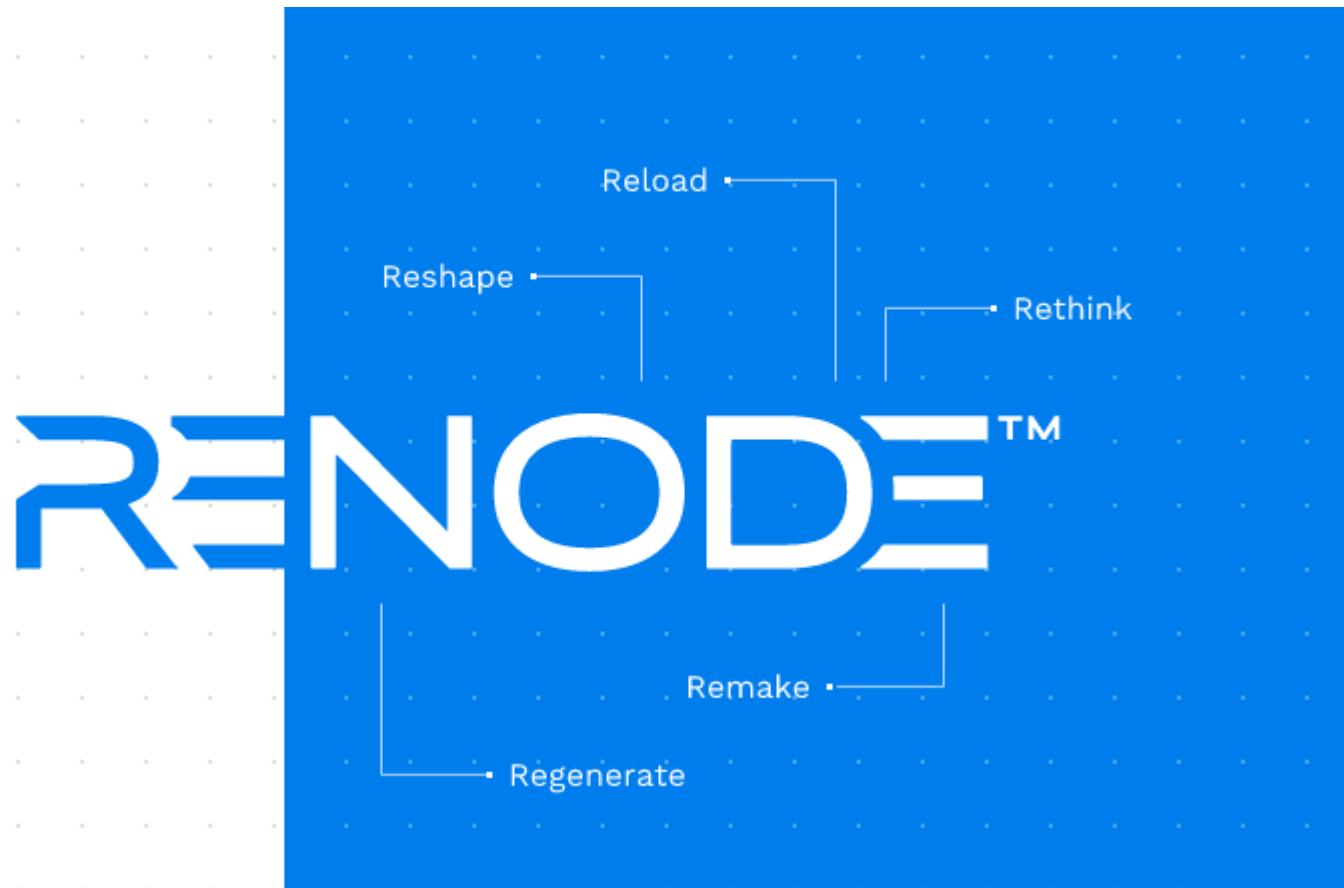
Fomu by Sutajio Kosagi

An FPGA board that fits inside your USB port



No hardware? Try Renode!

- <https://renode.io/>



No hardware? Try Renode!

- <https://renode.io/>

What is Renode?

- Renode is a development framework which accelerates IoT and embedded systems development by letting you simulate physical hardware systems - including both the CPU, peripherals, sensors, environment and wired or wireless medium between nodes.
- **It lets you run, debug and test unmodified embedded software on your PC - from bare System-on-Chips, through complete devices to multi-node systems.**



SiFive HiFive1

single-node/sifive_fe310.resc



SiFive HiFive Unleashed

single-node/hifive_unleashed.resc



Microchip PolarFire SoC
Hardware Development
Platform

single-node/polarfire-soc.resc



Toradex Colibri T30

single-node/tegra3.resc



OpenISA VEGAboard

single-node/vegaboard_ri5cy.resc

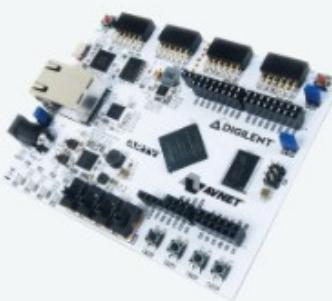


Intel Quark SE
Microcontroller Evaluation
Kit C1000

single-node/quark_c1000.resc



Fomu



LiteX/VexRiscv on Digilent
Arty



Xilinx ZedBoard

Activitie

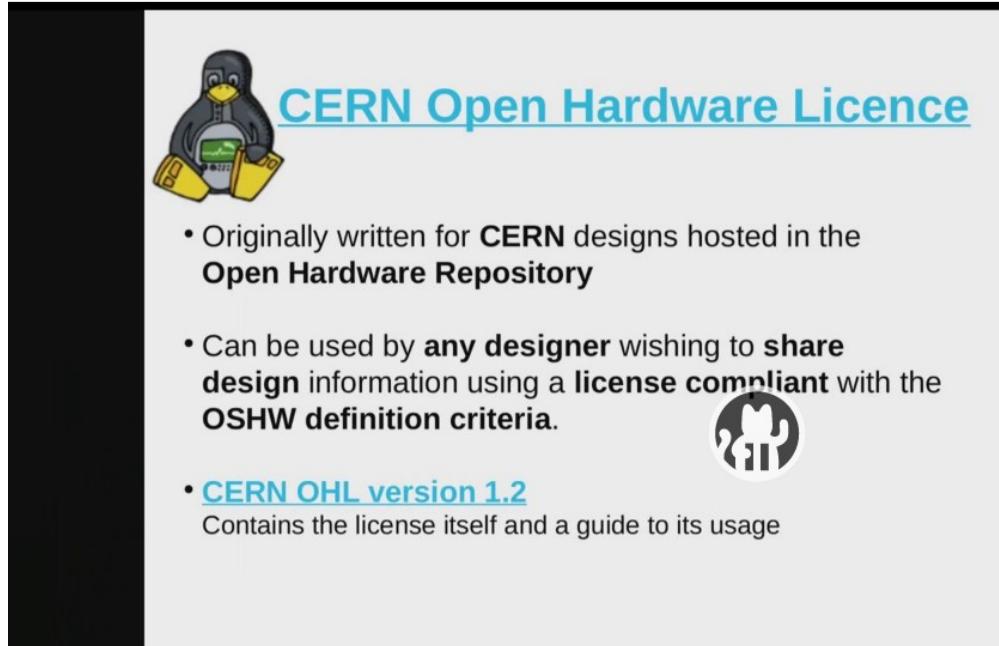
RE Renode ▾

```
SiFive RISC-V Coreplex
[0.000000] OF: fdt: Ignoring memory range 0x80000000 - 0x80200000
[0.000000] Linux version 4.15.0-80044-g2b0aa1de45f6 (houen@bakura) (gcc version 7.2.0 (GCC)) #5 SMP Wed
[0.000000] bootconsole [early0] enabled
[0.000000] Initial ramdisk at: 0x (ptrval) (9593856 bytes)
[0.000000] Zone ranges:
[0.000000]   DMA32    [mem 0x0000000080200000-0x000000008ffffffff]
[0.000000]   Normal   [mem 0x0000000900000000-0x000008ffffffffffff]
[0.000000] Movable zone start for each node
[0.000000] Early memory node ranges
[0.000000]   node 0: [mem 0x0000000000000000-0x0000000000000000]
[0.000000]   node 1: [mem 0x0000000000000000-0x0000000000000000]
```

Renode

```
(hifive-unleashed) $bin?=@http://antmicro.com/proj
.elf-s_17219640-c7e1b920bf81be4062f467d9ecf689dbf7f
(hifive-unleashed) $fdt?=@http://antmicro.com/proje
icetree.dtb-s_10532-70cd4fc9f3b4df929eba6e6f22d02e6
(hifive-unleashed) $vmlinux?=@http://antmicro.com/p
-vmlinux.elf-s_80421976-46788813c50dc7eb1a1a33c1738
(hifive-unleashed)
(hifive-unleashed) macro reset
> """
>     sysbus LoadELF $bin
>     sysbus LoadFdt $fdt 0x81000000 "earlyconsole"
>
>     # Load the Linux kernel symbols, as they are
>     sysbus LoadSymbolsFrom $vmlinux
>
>     # Device tree address is passed as an argument
>     e51 SetRegisterUnsafe 11 0x81000000
> """
(hifive-unleashed) runMacro $reset
(hifive-unleashed) start
Starting emulation...
(hifive-unleashed)
```

- Slides:
github.com/pdp7/talks/blob/master/rv-israel.pdf
- Contact: @pdp7 || drew@oshpark.com
- 36c3 talk



The screenshot shows a presentation slide titled "Linux on Open Source Hardware with Open Source chip design" by Drew Fustini. The slide features a penguin icon and the text "CERN Open Hardware Licence". It lists three bullet points about the 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

On the right side of the slide, there is a large "36c3 RESOURCE EXHAUSTION" logo. Below the slide, a video player interface shows a man speaking at a podium, and the bottom right corner of the slide also features the "36c3 RESOURCE EXHAUSTION" logo.