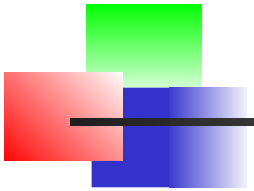


“From Intel x86 to RISC-V”

“Rise of RISC-V: The computer chip design you need to know about”

Three dominant IS Architectures

Past:

Intel x86 – controllers, PCs, servers, ..

Now:

ARM – controllers, smart-phones, laptops, PCs, servers

Future:

RISC-V – controllers, SBCs, laptops,

- **.. soon PCs, smart-phones, servers**

Past: Intel x86 – PCs, servers, ..

- Why Past
- Generations : 8080 : 8-bit, 8086: 16-bit, 80386: 32-bit, .., I3,5,7,9)
- CISC architecture: many formats → CISC to RISC transcoder
- no low power design : the designs for stationary use
- closed design: no standard (open) HDLs
- closed production: mainly produced by the designer company
  - Intel IDM: integrated device manufacturer
  - AMD : fabless + TSMC silicon foundry

Now: ARM – MCU, phones, PCs, servers

- **Why Now**
- Generations: ARM1-11, ARM-Cortex, M,R,A,X
- RISC architecture versions: Cortex-A v6,v7,v8,v9
- Low power: mobile phones, IoT, ..
- Design IPs with licenses : Verilog (synthesis)
  - Problem:
    - high license – model cost : $100K to $10M or more
    - royalties : 1.2% for each chip

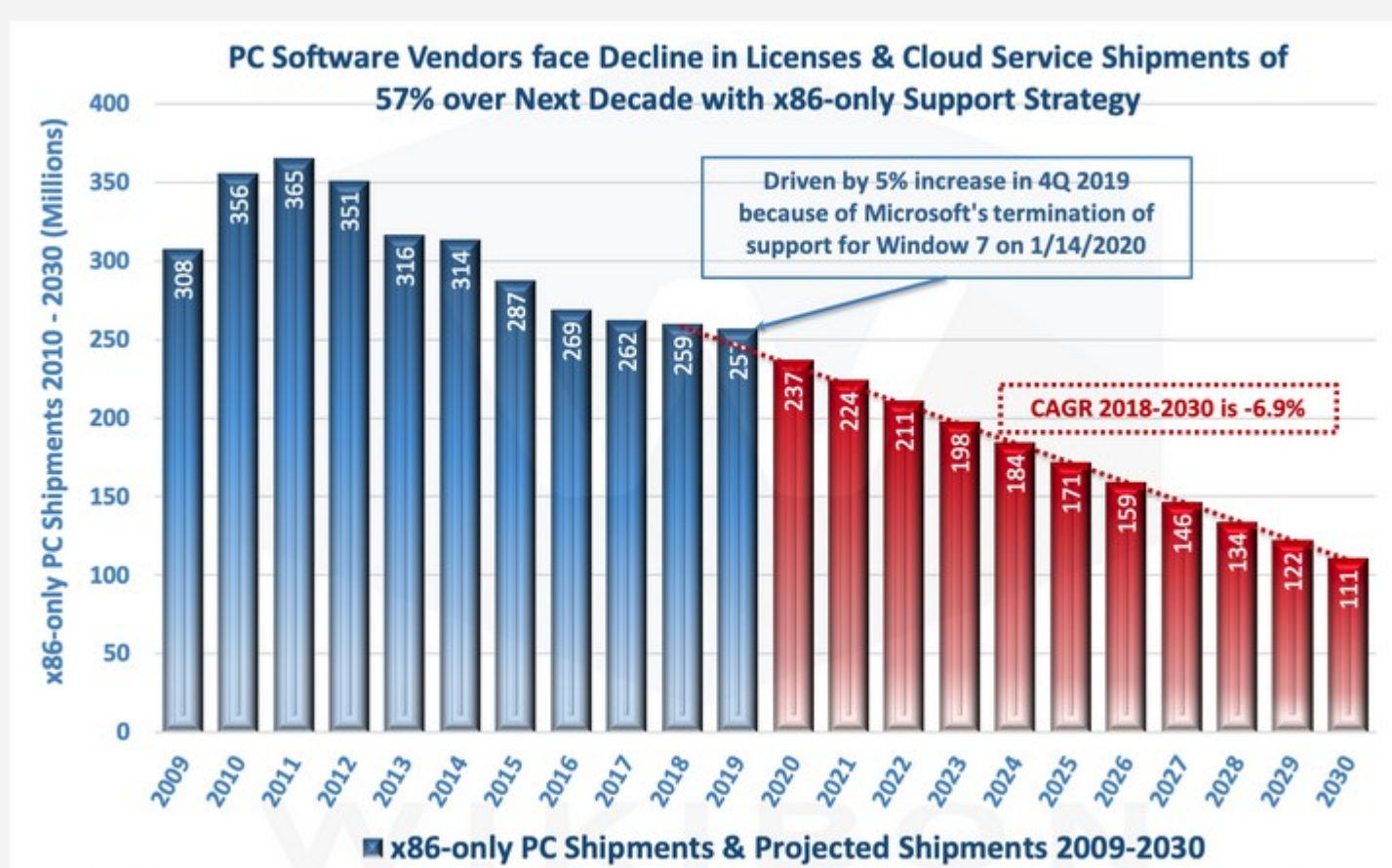
Future: RISC-V all devices

- **Why Future**
- Standardized ISA formats: 32, 64, and 128-bit
- Optimized (for performance) RISC architecture
- No ISA use licenses – "open source"
- Many open source (mainly Verilog) and licensed designs
- Designers free to implement internal architectures
- Designers free to add specific ISA extensions (IA, IoT,..)

Great competition : hundreds/thousands designs

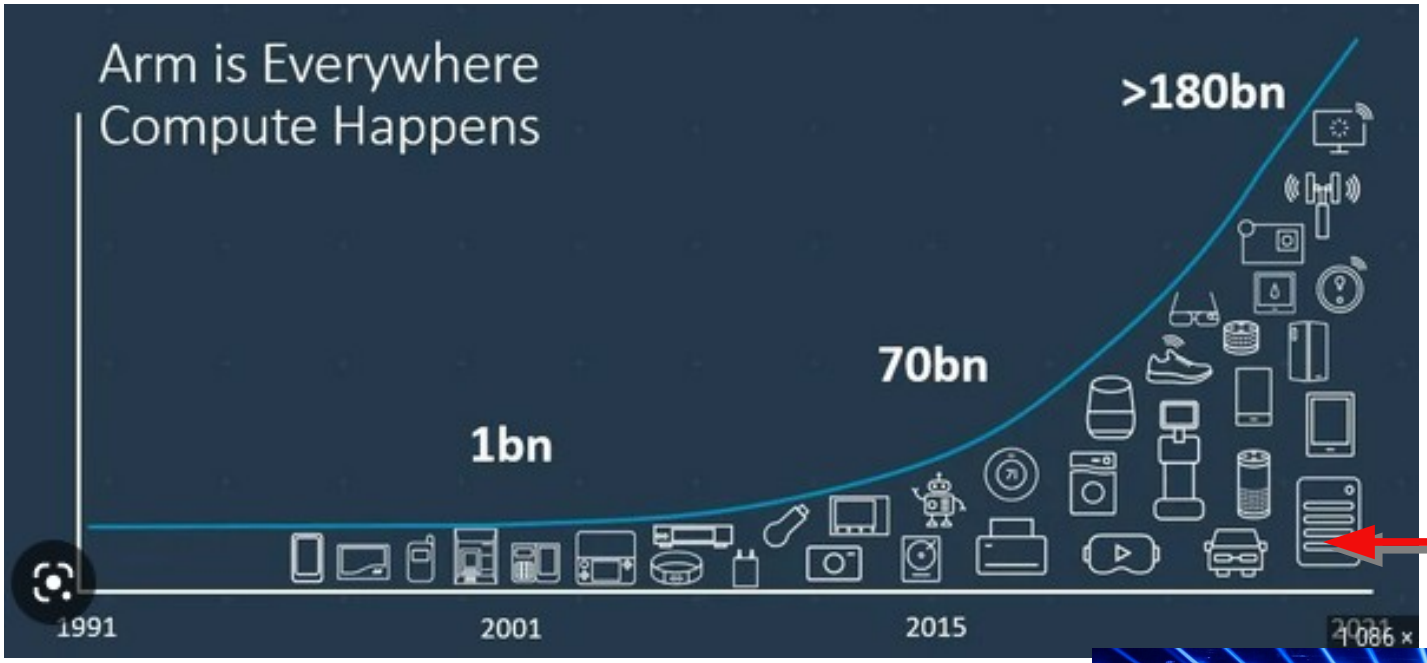
The numbers of produced units

2021: Intel x86 – 400 M



The numbers of produced units

2021: ARM 22 billions cumulative number



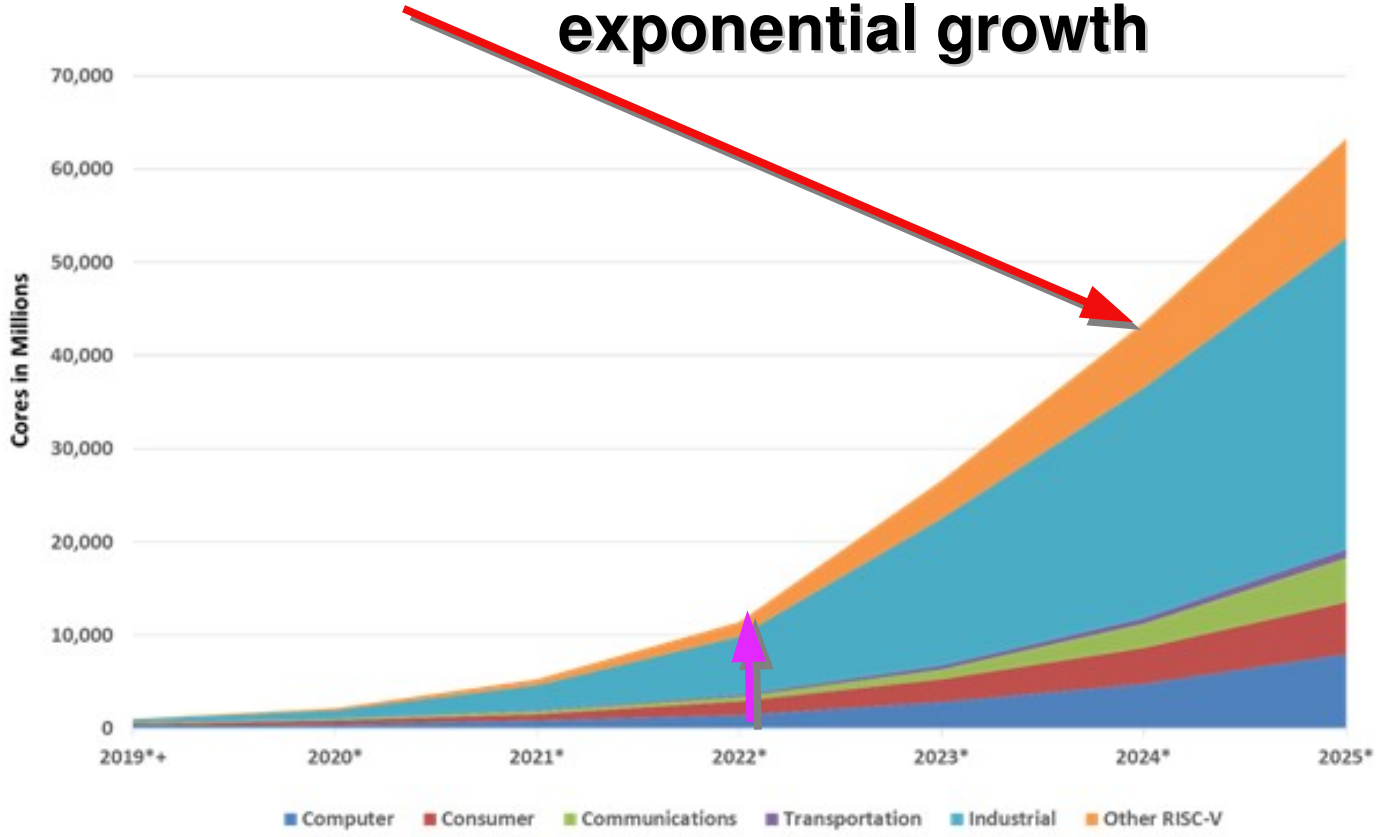
Yitian 710

The numbers of produced cores

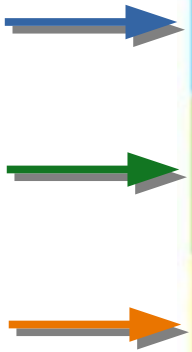
2022: RISC-V 12 billions

2024: RISC-V 45 billions (5 RISC-V cores/head)

exponential growth



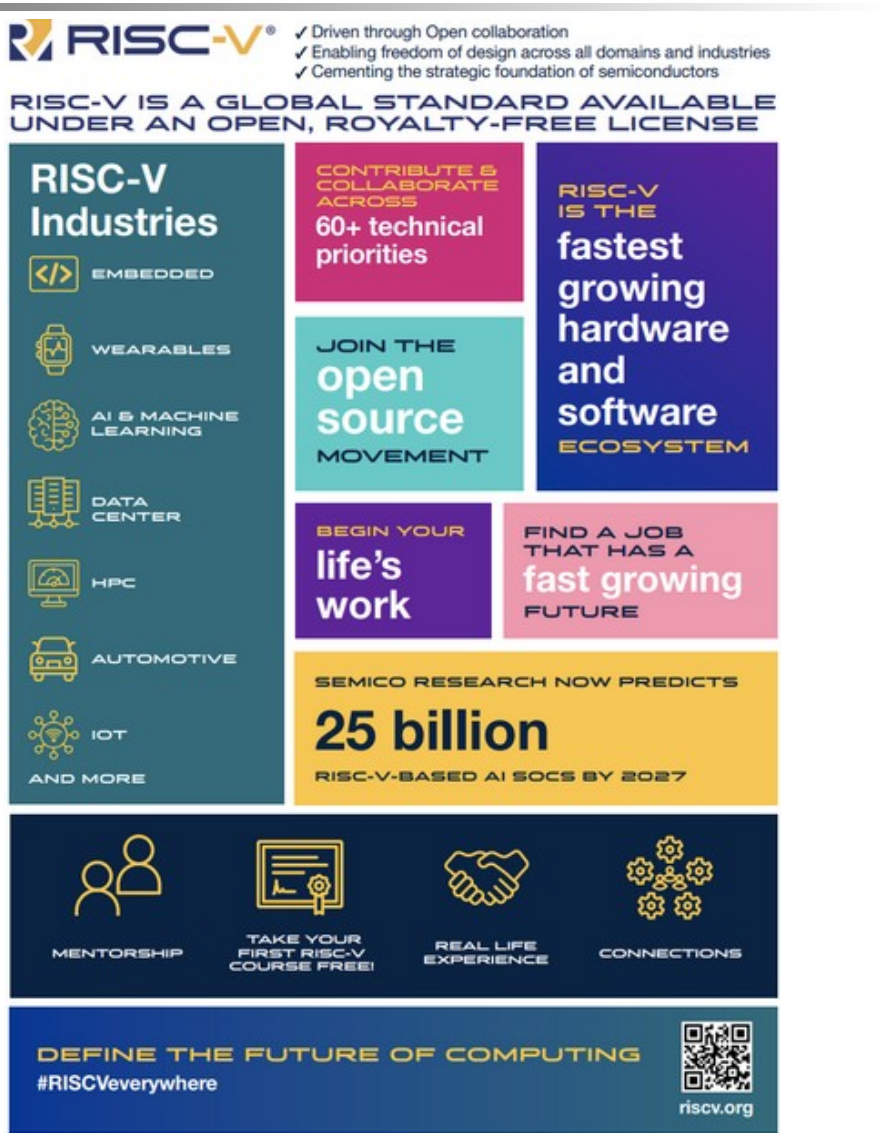
ISA - business models



ISA	Chips?	Architecture License?	Commercial Core IP?	Add Own Instructions?	Open-Source Core IP?
x86	Yes, ***three*** vendors	***No***	***No***	***No***	***No***
ARM	Yes, ***many*** vendors	Yes, ***expensive***	Yes, ***one*** vendor	***No*** *(Mostly)*	***No***
RISC-V	Yes, ***many*** vendors	Yes, ***free***	Yes, ***many*** vendors	***Yes***	***Yes, many available***

RISC-V ecosystem

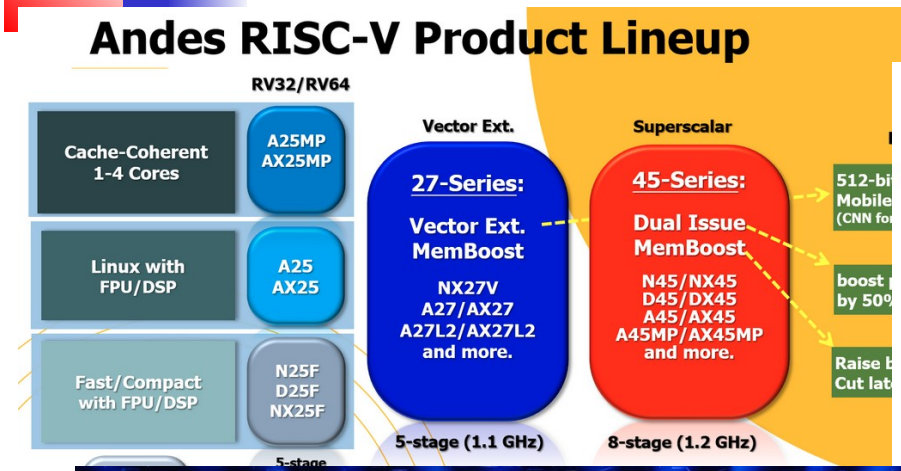
Fastest growing Hardware & Software ecosystem

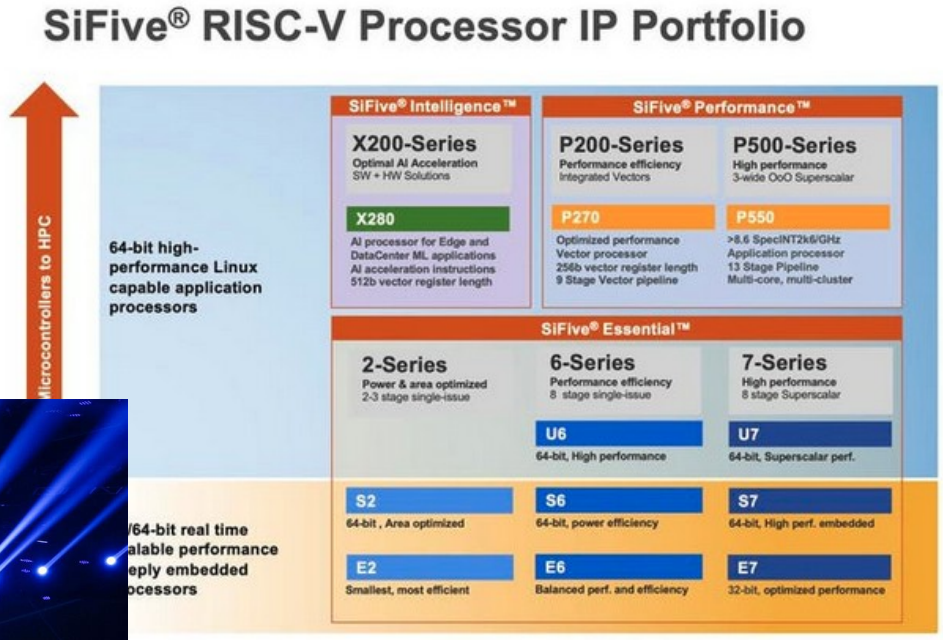


RISC-V conferences, R&D

Fastest growing Hardware & Software research/design activities

RISC-V industrial players





Alibaba : T-HEAD XuanTie – 2.5 billion cores (2022)

RISC-V: T-HEAD XuanTie processors

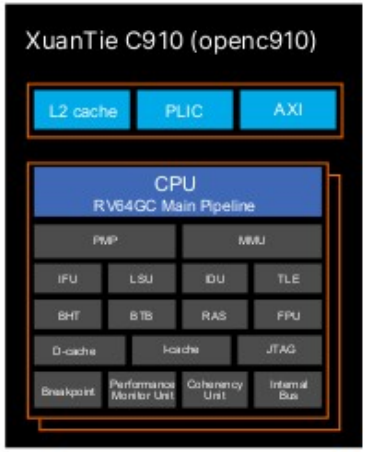
In 2021 T-HEAD released open source IPs in Verilog for: E902, E906, C902, C906 and C910 processor cores

RISC-V: T-HEAD XuanTie C910

Overview

ISA: RV64GC - RV64IAMFDC

C910 is a RISC-V compatible 64-bit high performance processor developed by T-Head Semiconductor Co., Ltd. It delivers industry-leading performance in control flow, computing and frequency through architecture and micro-architecture innovations. The C910 processor is based on the RV64GC instruction set and implements the XIE (XuanTie Instruction Extension) technology. C910 adopts a state of the art 12 stages out-of-order multiple issue superscalar pipeline with high frequency, IPC, and power efficiency. C910 supports hardware cache coherency. Each cluster contains 2 cores. The C910 supports the AXI4 bus interface. The C910 uses the Sv39 virtual address system with XMAE (XuanTie Memory Attributes Extension) technology. In addition, C910 includes the standard CLINT and PLIC interrupt controllers and performance monitors. C910 implements the T-Head debug protocol.



12-stages, out-of-order, super-scalar, Sv39

RISC-V: SG2044 from SOPHGO

SG2044 : first 64-core CPU based on C910

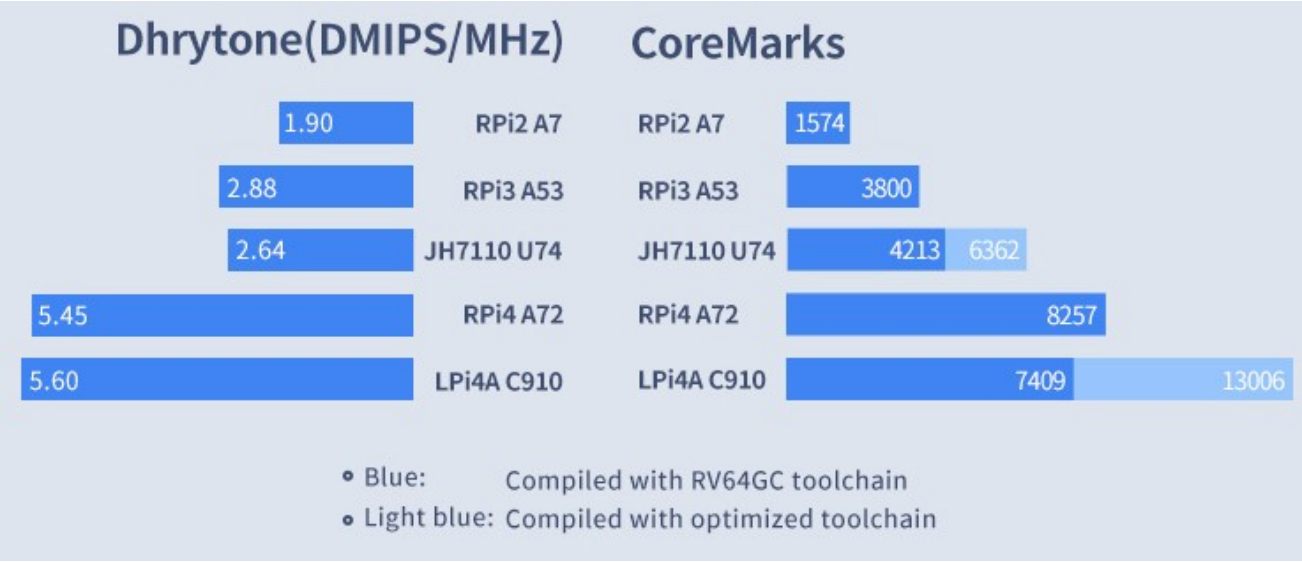
SOPHGO has research and development centers in more than 10 cities in China such as Beijing, Shanghai, Shenzhen, Qingdao, Xiamen, as well as in countries such as the United States and Singapore.

128G DDR4/ 2TB SSD
$2,499 Free US Shipping / $18 Worldwide

SmartComputerLab

RISC-V: T-HEAD XuanTie C910

The **LicheePi4A** - RISC-V SBC based on **TH1520 SoC** (4xC910@1.85GHz),
-4/8/16 GB RAM, and up to 128GB eMMC
HDMI+MIPI dual 4K display, 4K camera input, dual Gigabit Ethernet interfaces (one of these supports POE) and 4 USB3.0 interfaces,- audio processing by C906 core.



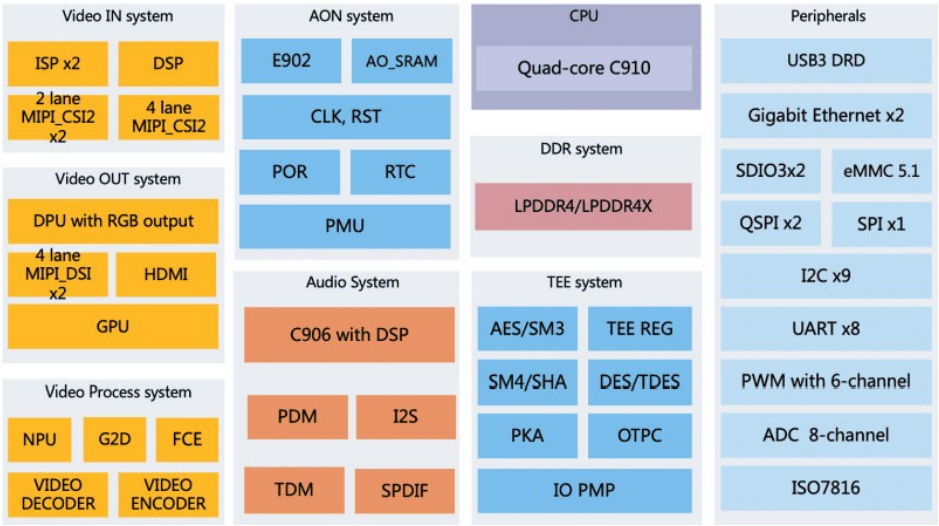
RISC-V: TH1520 SoC

CPU:

RISC-V 64GCV C910*4@1.85GHz

· Each core contains 64KB I cache and 64KB D Cache

· 1MB of Shared L2 Cache



GPU:

· OpenCL 1.1/1.2/2.0

· OpenGL ES 3.0/3.1/3.2

· Vulkan 1.1/1.2

· Android NN HAL

NPU:

Support 4TOPS@INT8, up to 1GHz

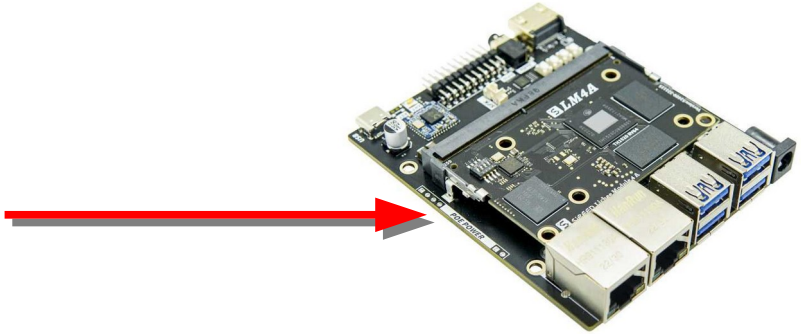
· Support TensorFlow、 ONNX、 Caffe

· Support CNN、 RNN、 DNN

RISC-V: T-HEAD XuanTie C910

You can login with your first name:

ssh nicolas@86.217.17.226



password: **smartcomputerlab**

```
ubuntu@bako:~$ ssh sipeed@86.217.17.226
sipeed@86.217.17.226's password:
Linux lpi4a 5.10.113-g387b6863253c-dirty #21 SMP PREEMPT Fri Jul 7 03:29:22 SAST 2023 riscv64

The programs included with the Debian GNU/Linux system are free software;
the exact distribution terms for each program are described in the
individual files in /usr/share/doc/*/copyright.

Debian GNU/Linux comes with ABSOLUTELY NO WARRANTY, to the extent
permitted by applicable law.
Last login: Thu Nov  9 13:48:40 2023 from 86.236.127.206
sipeed@lpi4a:~$ ls
Design  Desktop  Programming  gst
sipeed@lpi4a:~$ ls Programming/lab1
CountToTen              HelloWorldArgNumNoStack     MultHundredScan.s
CountToTen.c            HelloWorldArgNumNoStack.s   MultiplyByTen
CountToTen.s            HelloWorldArgScan           MultiplyByTen.c
HelloWorld              HelloWorldArgScan.c         MultiplyByTen.s
HelloWorld.c            HelloWorldArgScan.s         MultiplyByTenNoStack
HelloWorld.s            HelloWorldArgScanNoStack    MultiplyByTenNoStack.s
```

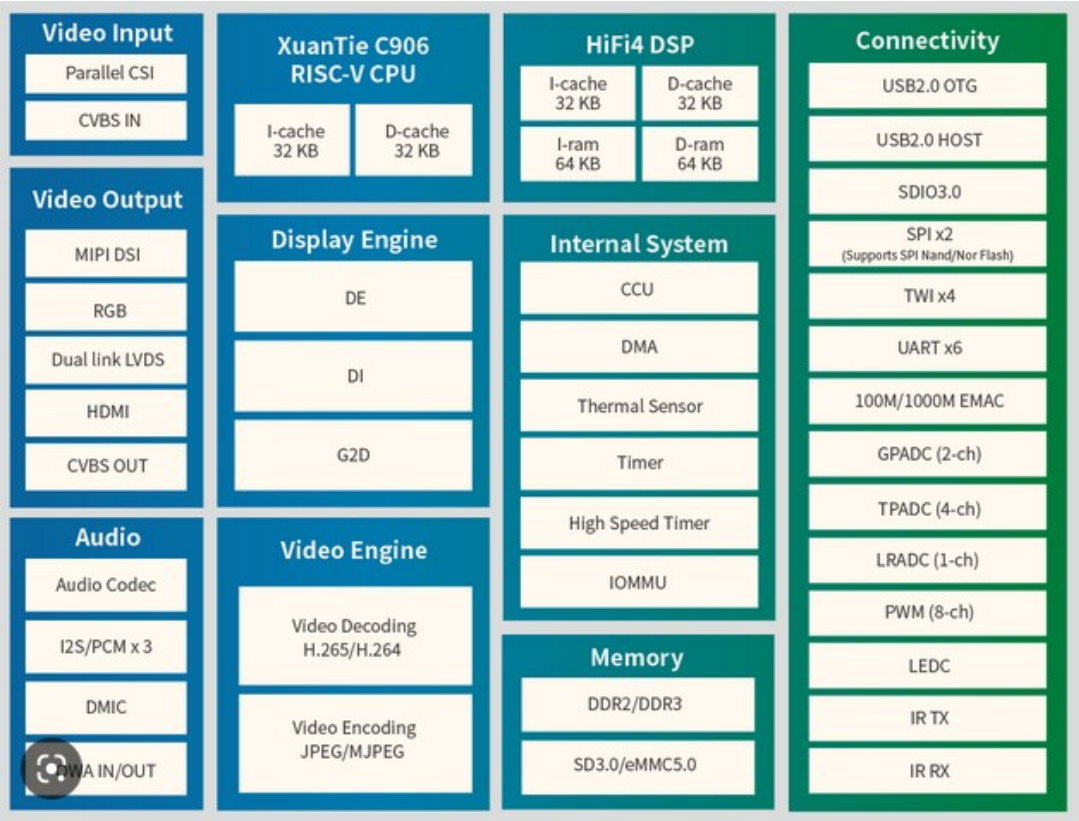
RISC-V: Allwinner – D1 SoC with C906

XuanTie C906 with the RISC-V Vector Extension (**V0.7**)

XuanTie C906 is a **64-bit processor** based on a 64-bit RISC-V architecture.

This processor is designed with a **five to eight stage integer pipeline**. It is also equipped with **128-bit vector operation units**.

Data formats, including **`int8`, `int16`, `int32`, `int64`, `bf16`, `fp16`, `fp32`**, and **`fp64`**, are supported.



RV64IAMFDCV

Lichee RV Dock (RV64)

Lichee RV Dock is a **RISC-V Linux development kit**.

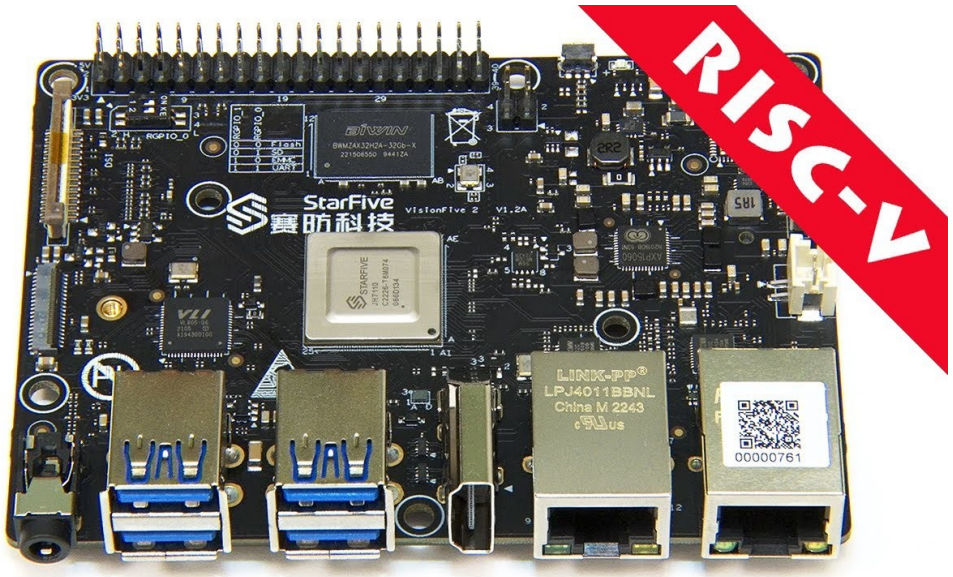
- an **HDMI port** with support for up to 4K@30fps output, a 40-pin header with **GPIO** - speakers, microphones, and more.

- **RGB and MIPI screen** interfaces with its screen convert board.

- an onboard **2.4G WIFI+BT module**, a 2.4G Patch antenna, an IPEX connector, and a **USB Type-A** host.

StarFive VisionFive 2 SBC

VisionFive 2 is high-performance RISC-V single board computer (SBC) with an integrated GPU.



VisionFive 2 is used for the RISC-V laboratories including assembly programming (**Lab1/Lab2**) and design (**Lab3**)

StarFive VisionFive 2 SBC

VisionFive 2 :

- **JH7110 SoC**

and

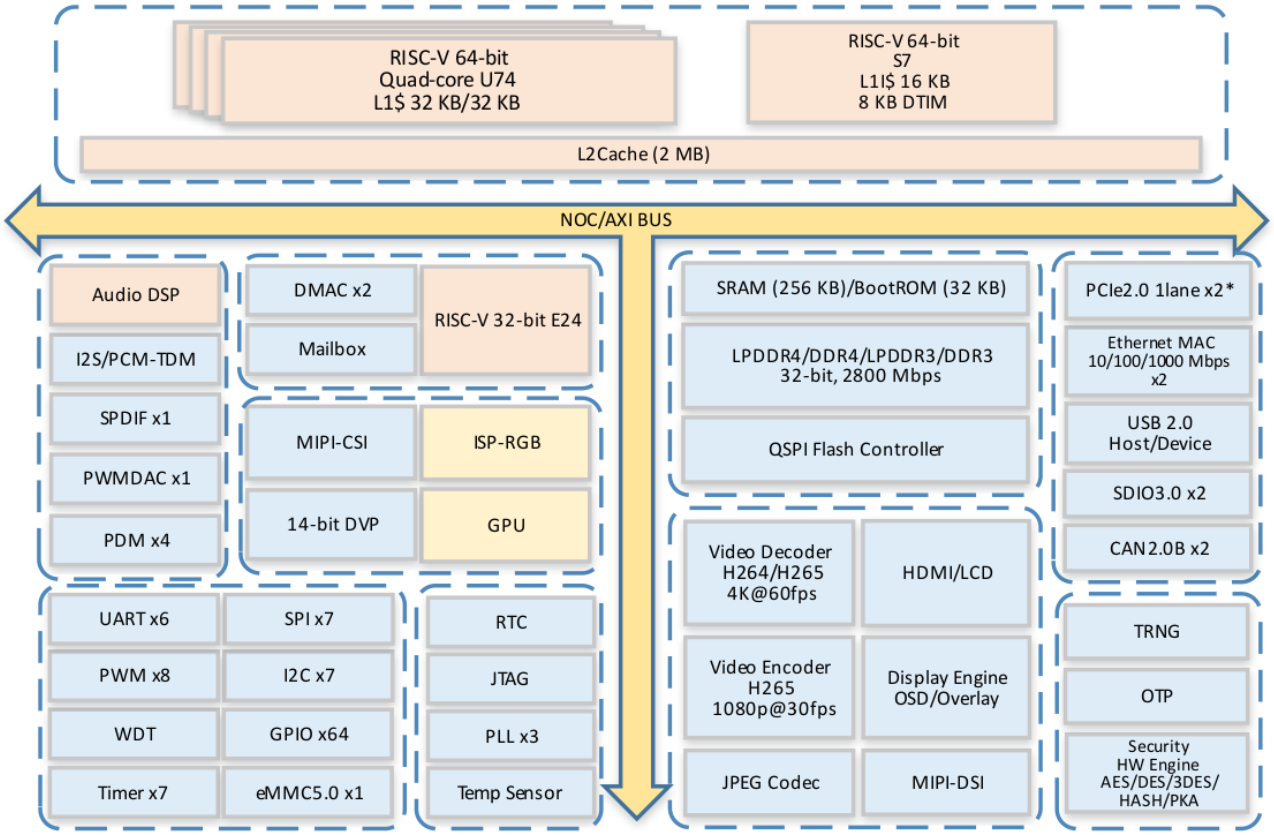
- board

Item	Description
Processor/SOC	StarFive JH7110 64bit SoC with RV64GC, up to 1.5GHz
Memory	LPDDR4, 2GB/4GB/8GB
Storage	TF card slot
	Flash for Uboot
Video output	HDMI 2.0
	MIPI-DSI
Multimedia	Camera with MIPI CSI, up to 4k@30fps
	H.264 & H.265 4k@60fps Decoding H.265 1080p@30fps Encoding JPEG encoder/decoder
	1/ 4-pole stereo audio 2. HDMI
Connectivity	2x RJ45 Gigabit Ethernet
	2x USB2.0 + 2x USB 3.0
	M.2 M-Key
Power	USB-C port, 5V DC via USB-C with PD, up to 30W
	GPIO Power in, 5V DC via GPIO header (minimum 3A+)
	PoE
GPIO	40 pin GPIO header
Dimensions	100 x 72mm
Compliance	RoHS, FCC, CE
Button	Reset Button
Other	Debug Pin Headers

StarFive VisionFive 2 SBC

A strong partnership with **SiFive**, **StarFive** started as an exclusive distributor of **SiFive** RISC-V core IP products in the greater China region

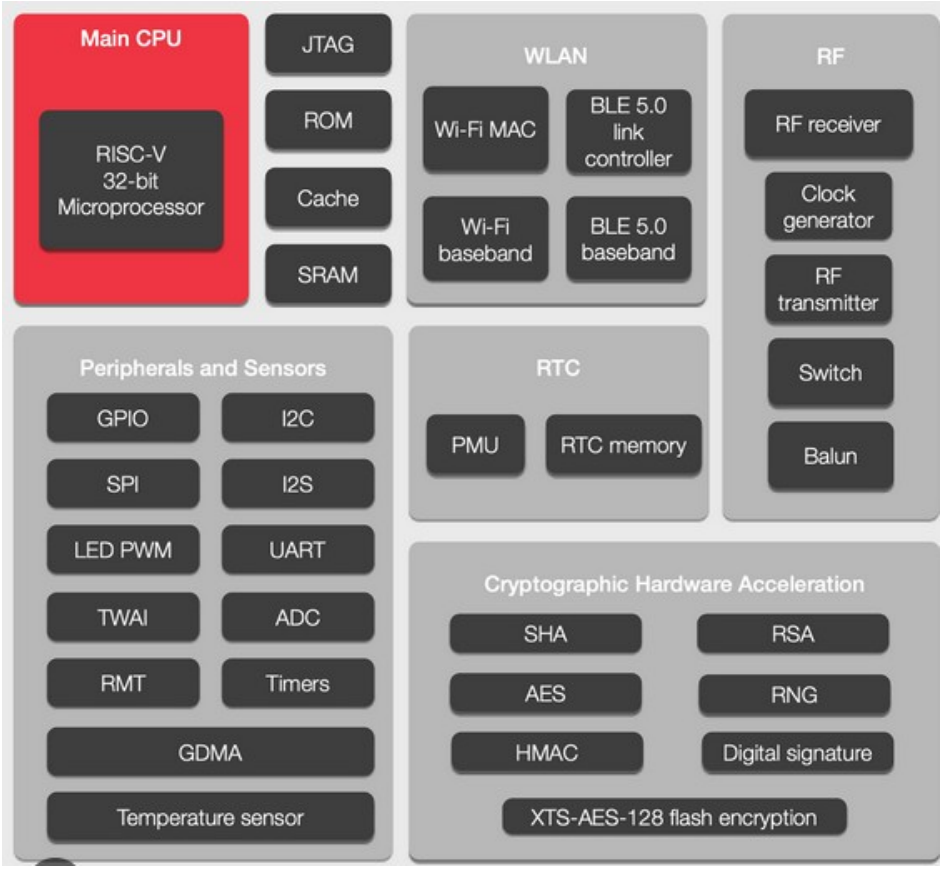
JH7110 SoC



RISC-V MCU – RV32 - ESP32-C3

The **ESP32-C3 SoC** from **Espressif** uses the free and open **RISC-V instruction set architecture.**

It **competes with ARM-based controllers** widely used for **IoT applications**.



RISC-V: ISA (Instruction Set Architecture)

RISC-V is an open standard instruction set architecture (ISA) based on established reduced instruction set computer (RISC) principles.

Unlike most other ISA designs, RISC-V is provided under royalty-free open-source licenses.

RISC-V ISA is a load–store architecture.

Its floating-point instructions use IEEE 754 floating-point.

Notable features of the RISC-V ISA include:

- instruction bit field locations chosen to simplify the use of multiplexers in a CPU.

A design that is architecturally neutral, and a fixed location for the sign bit of immediate values to speed up sign extension.

RISC-V: ISA (Instruction Set Architecture)

The designers' primary assertion is that the instruction set is the key interface in a computer as it is situated at the interface between the hardware and the software.

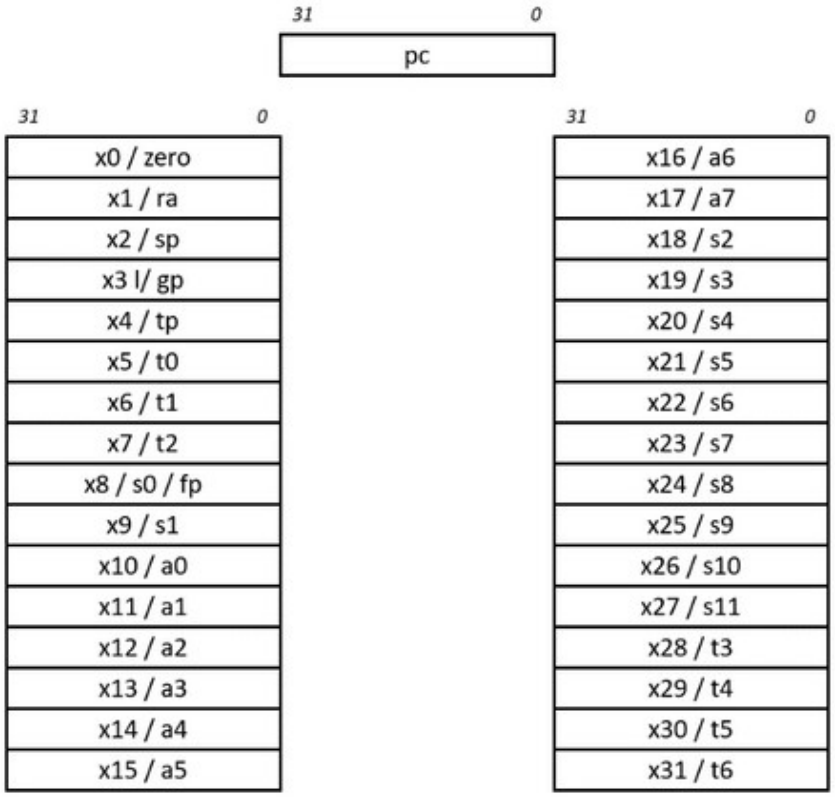
If a good instruction set were open and available for use by all, then it can dramatically reduce the cost of software by enabling far more reuse.

The base instruction set has a fixed length of 32-bit naturally aligned instructions, and the ISA supports variable length extensions where each instruction can be any number of 16-bit parcels in length.

Subsets support small embedded systems, personal computers, supercomputers with vector processors.

The instruction set specification defines 32-bit (examples: E902/E906) and 64-bit (examples: C906/C910) address space variants.

RISC-V ISA - registers (RV32I)



ra: Function return address.
sp: Stack pointer.
gp: Global data pointer.
tp: Thread-local data pointer.
t0–t6: Temporary storage.
fp: Frame pointer for function-local stack data (this usage is optional).
s0–s11: Saved registers (if the frame pointer is not in use, x8 becomes s0).
a0–a7: Arguments passed to functions. Any additional arguments are passed on the stack. Function return values are passed in a0 and a1.

RISC-V ISA - formats

32-bit RISC-V Instruction Formats

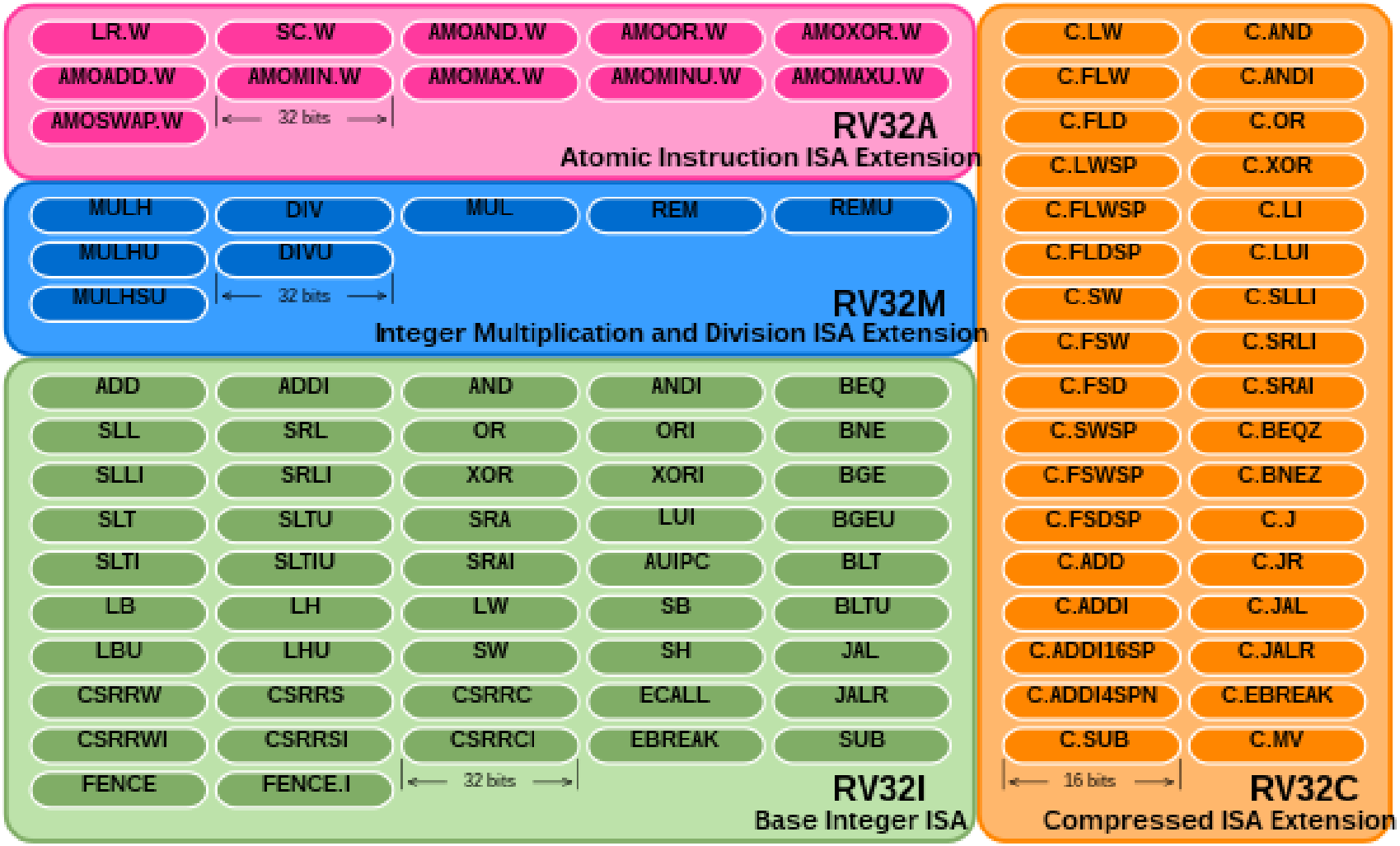
Instruction Formats	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Register/register	funct7							rs2					rs1					funct3			rd					opcode						
Immediate	imm[11:0]												rs1					funct3			rd					opcode						
Upper Immediate	imm[31:12]																				rd					opcode						
Store	imm[11:5]							rs2					rs1					funct3			imm[4:0]					opcode						
Branch	[12]	imm[10:5]						rs2					rs1					funct3			imm[4:1]				[11]	opcode						
Jump	[20]	imm[10:1]										[11]	imm[19:12]								rd					opcode						

- **opcode (7 bit):** partially specifies which of the 6 types of *instruction formats*
- **funct7 + funct3 (10 bit):** combined with **opcode**, these two fields describe what operation to perform
- **rs1 (5 bit):** specifies register containing first operand
- **rs2 (5 bit):** specifies second register operand
- **rd (5 bit):**: Destination register specifies register which will receive result of computation

ISA *computational instructions* use a three-operand format, in which the first operand is the destination register, the second operand is a source register, and the third operand is either a second source register or an immediate value. This is an example three-operand instruction:

```
add x1, x2, x3
```

ISA - instructions/mnemonics



RISC-V : Software-Hardware

high level programming

assembly level programming

```
add     a2,a2,a3
mulw    a5,a4,a5
addi    s0,sp,32
jr      ra
```

USER – binaries

OS

extension

type

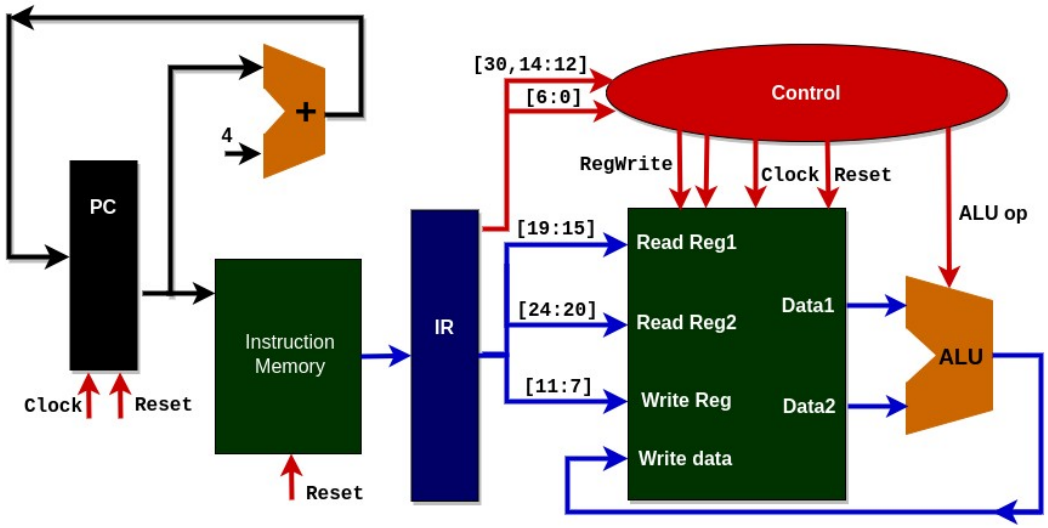
design

RISC-V : Software-Hardware

extension

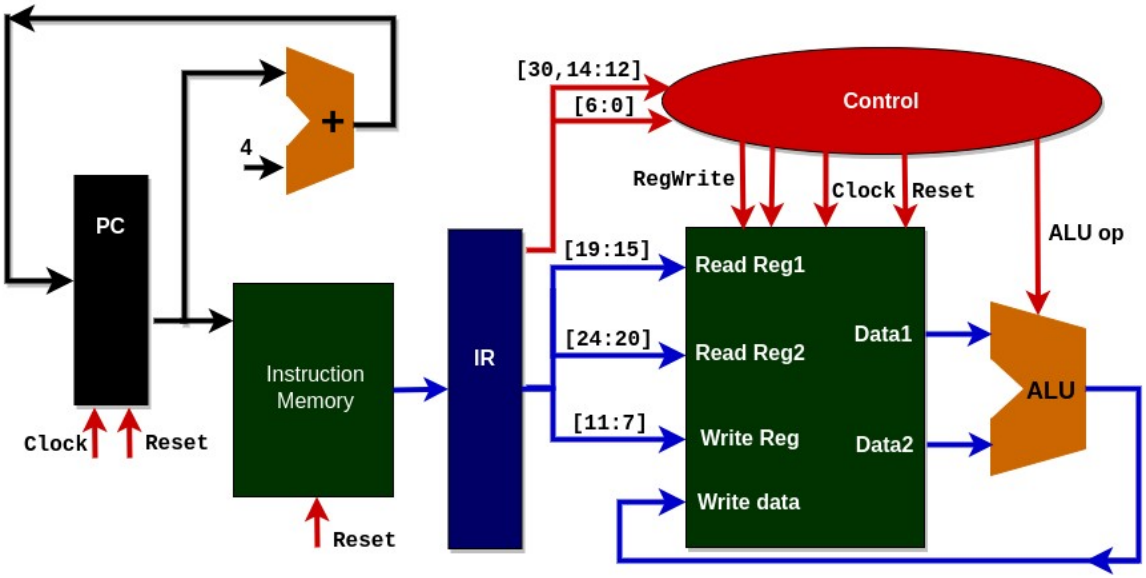
type

design




```
add     a2,a2,a3
```

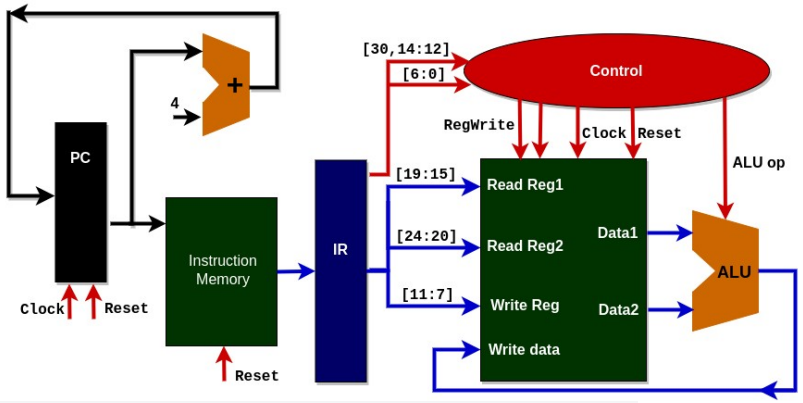
RISC-V : Software-Hardware

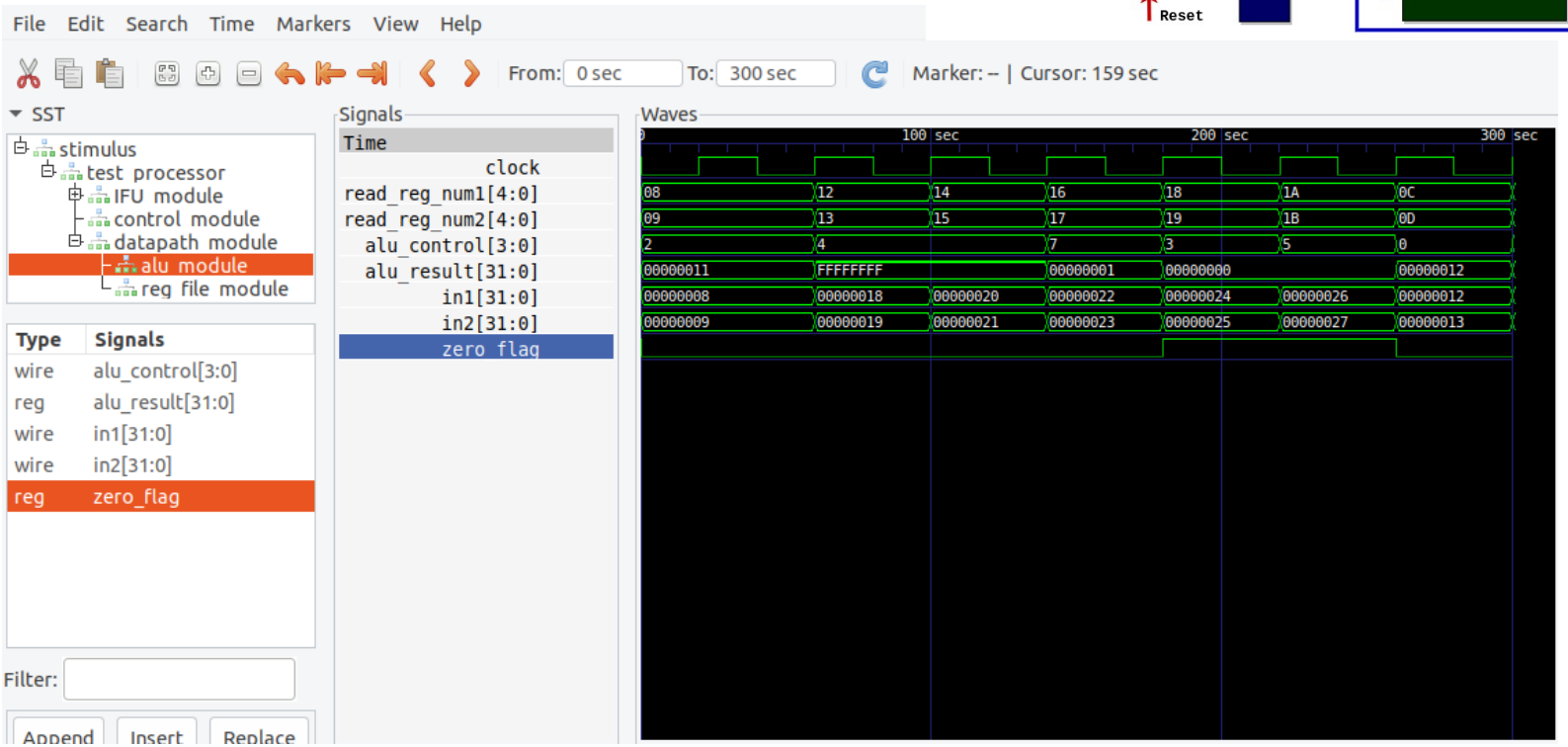



```
`include "CONTROL.v"
`include "DATAPATH.v"
`include "IFU.v"

module PROCESSOR(
    input clock,
    input reset,
    output zero
);

wire [31:0] instruction_code;
wire [3:0] alu_control;
wire regwrite;

IFU IFU_module(clock, reset, instruction_code);

CONTROL control_module(instruction_code[31:25],instruction_code[14:12],
                       instruction_code[6:0],alu_control, regwrite);

DATAPATH datapath_module(instruction_code[19:15], instruction_code[24:20],
                  instruction_code[11:7], alu_control, regwrite, clock, reset, zero);

endmodule
```

RISC-V : Software-Hardware

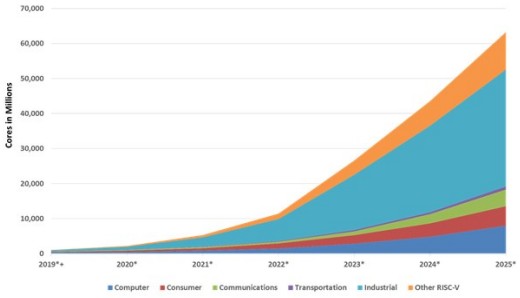
GTKWave waveforms





Summary

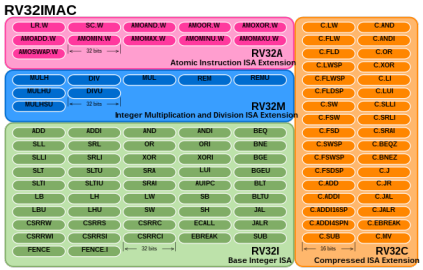
The rise of RISC-V open architecture exponential growth of cores



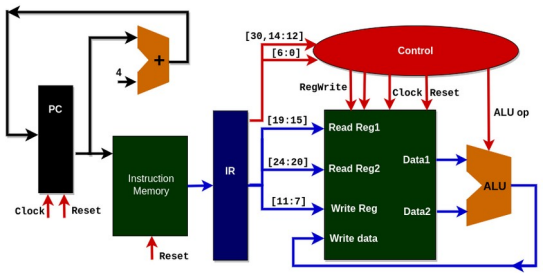
RISC-V open source & business model

ISA	Chips?	Architecture License?	Commercial Core IP?	Add Own Instructions?	Open-Source Core IP?
x86	Yes, *three* vendors	*No*	*No*	*No*	*No*
ARM	Yes, *many* vendors	Yes, *expensive*	Yes, *one* vendor	*No (Mostly)*	*No*
RISC-V	Yes, *many* vendors	Yes, *free*	Yes, *many* vendors	*Yes*	*Yes, many available*

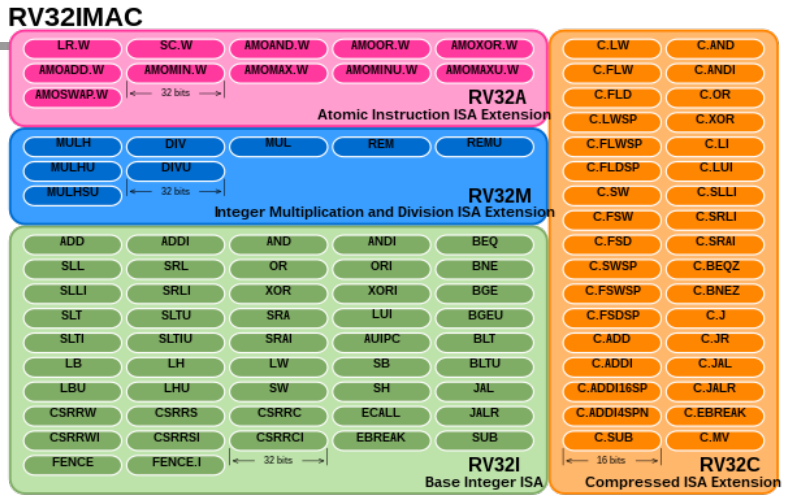
RISC-V optimized & extensible ISA



RISC-V architecture & Verilog synthetisable models



RISC-V labs



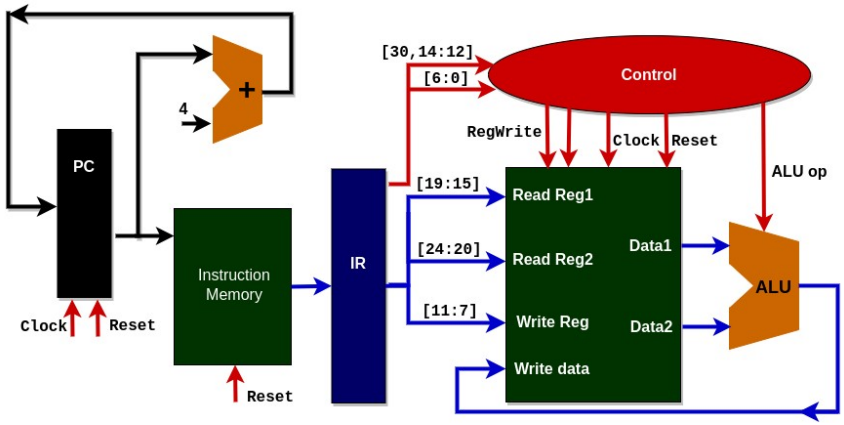
Lab1 & Lab2

RISC-V ISA and assembly programming

```
add     a2,a2,a3
mulw    a5,a4,a5
addi    s0,sp,32
jr      ra
```

Lab3

RISC-V design: architecture & simple Verilog model




```
case(alu_control)
4'b0000: alu_result = in1&in2;
4'b0001: alu_result = in1|in2;
4'b0010: alu_result = in1+in2;
4'b0100: alu_result = in1-in2;
```