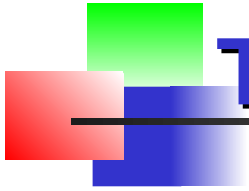


“From Intel x86 to RISC-V”

**“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 – .. now: controllers, SBCs, laptops,

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



Past: Intel **x86** – PCs, servers, ..

■ Why Past

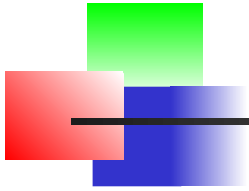
- Generations : 8080 : 8-bit, 8086: 16-bit, 80386: 32-bit, .., I3,5,7,9)
- **CISC** architecture: many formats → CISC to RISC transcoder
- **no low power design** : the designs for stationary use
- **closed design**: no standard (open) HDLs
- **closed production**: mainly produced by the designer company
 - 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

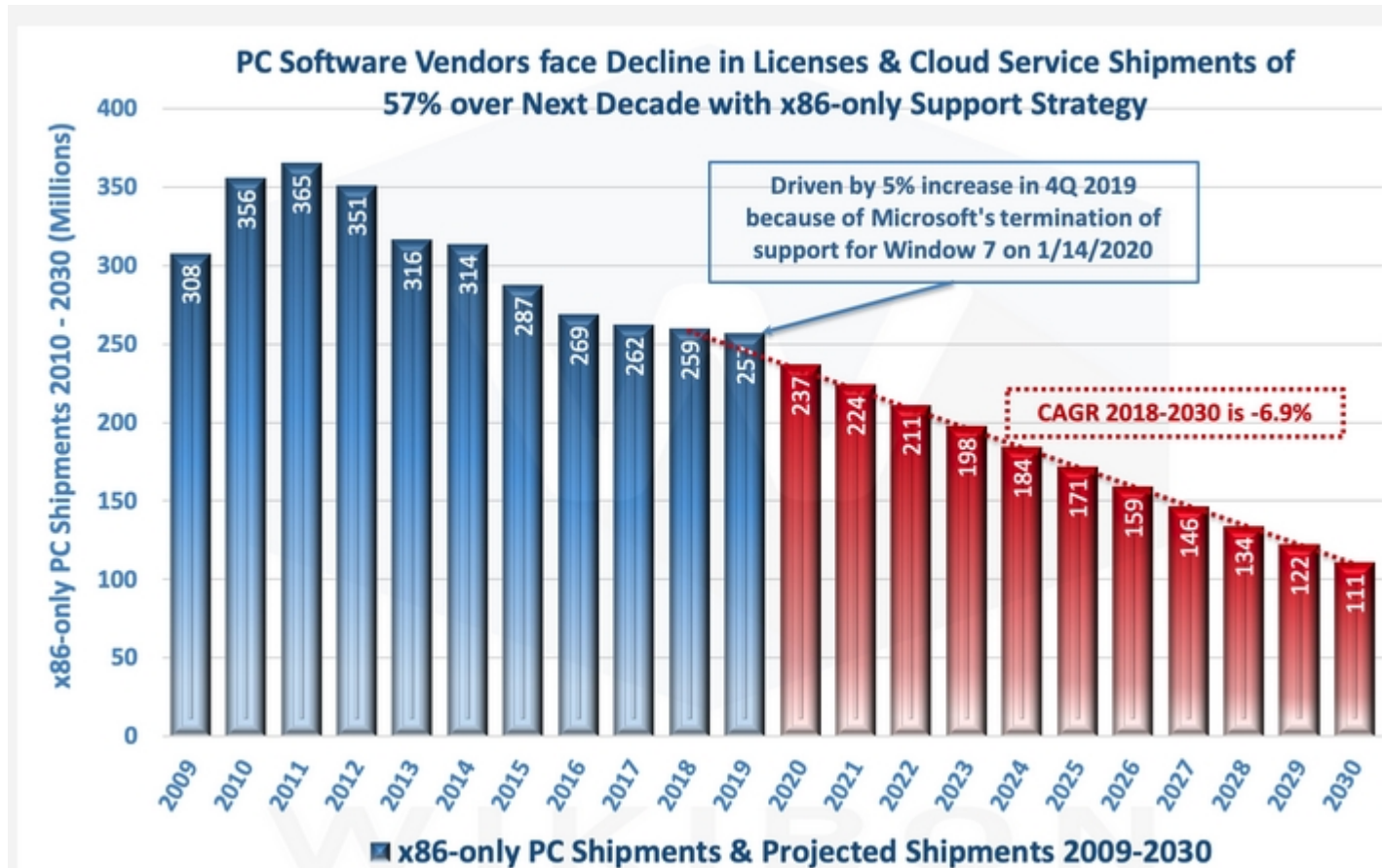
■ Why Future

- 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

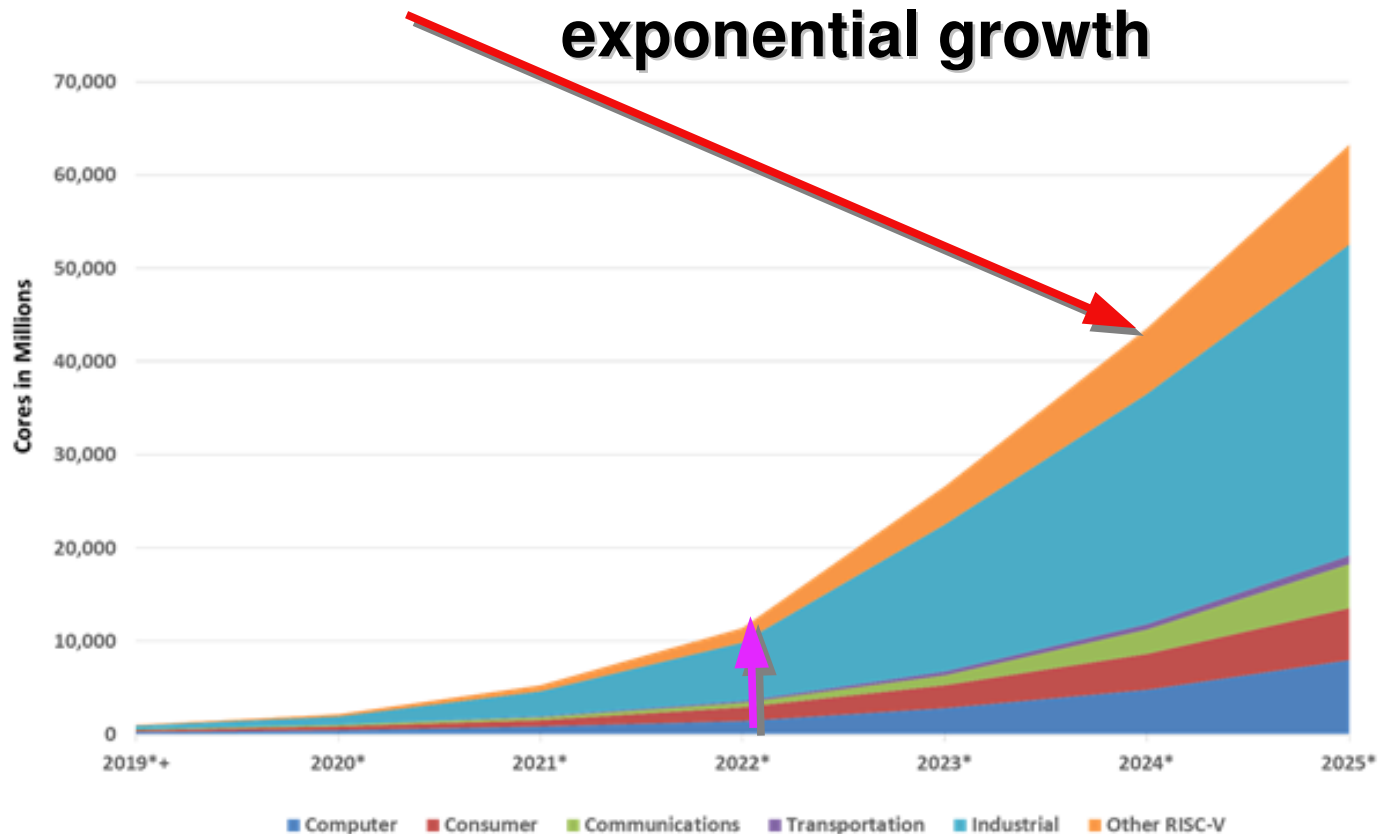
2021: Intel x86 – 400 M



The numbers of produced cores




2022: RISC-V 12 billions

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

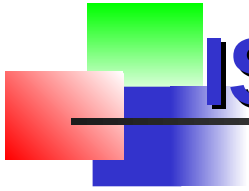




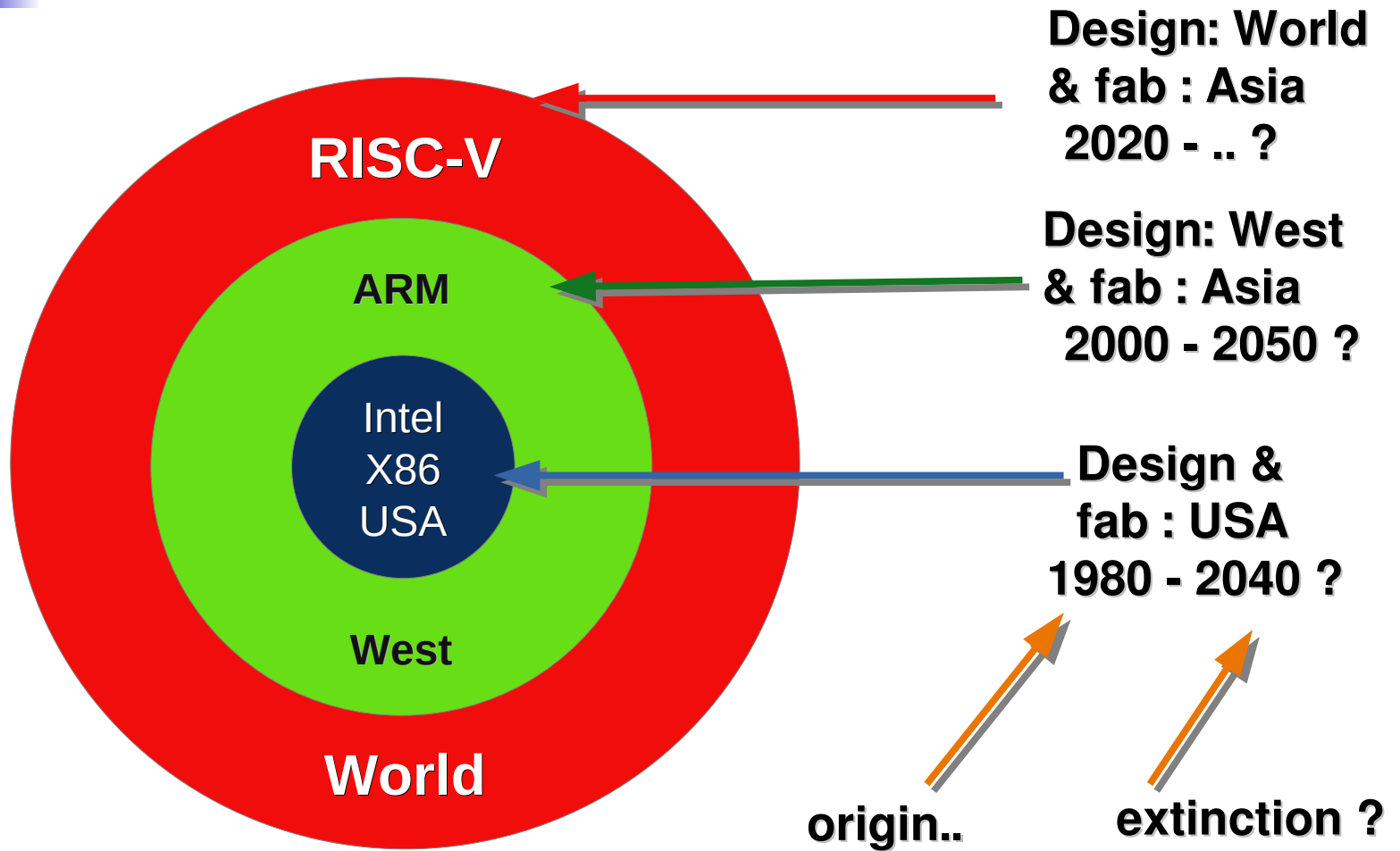
ISA - business models



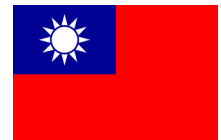
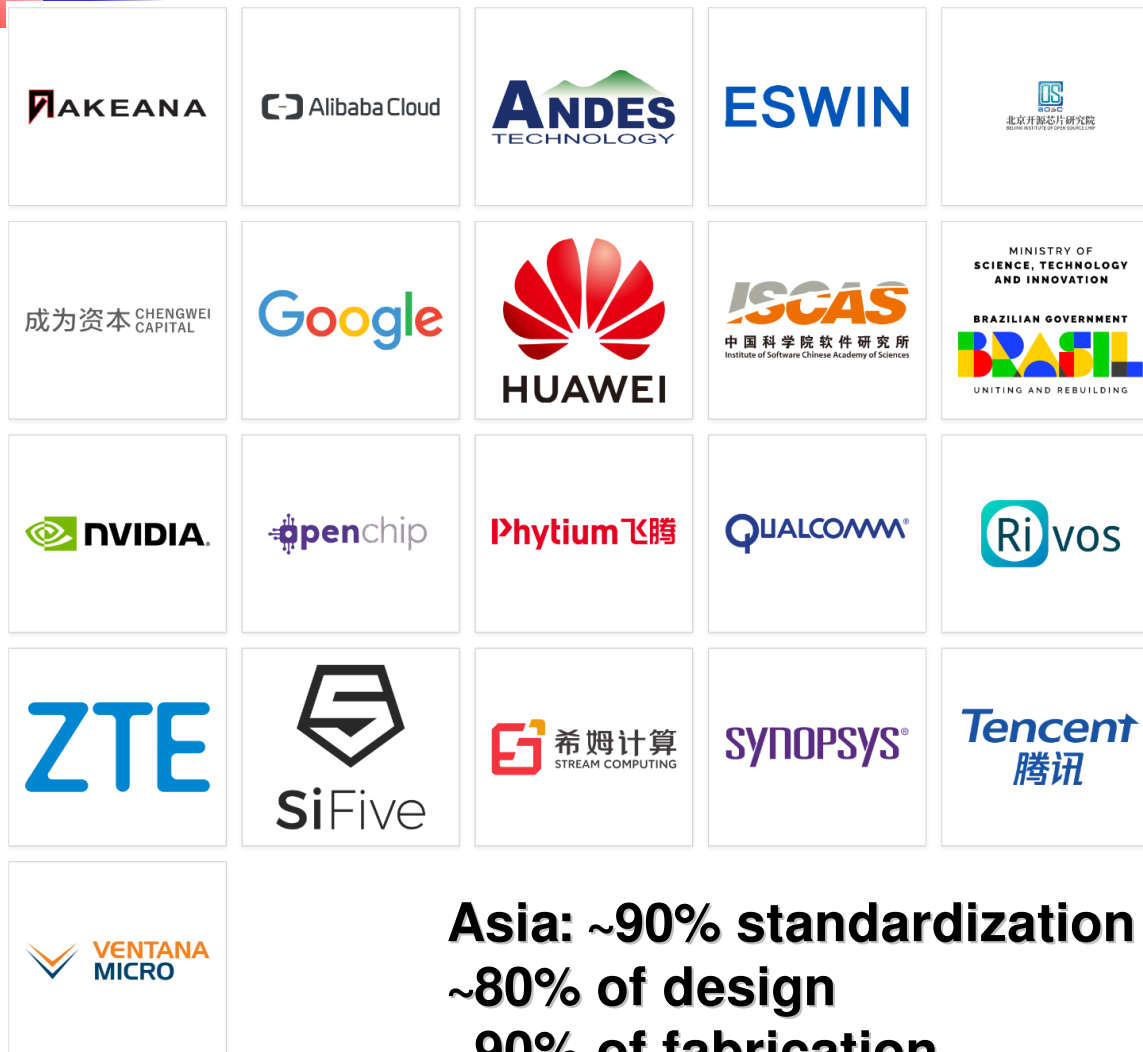
ISA	Chips?	Architecture License?	Commercial Core IP?	Add Own Instructions?	Open-Source Core IP?
x86	Yes, <i>three</i> vendors	No	No	No	No
ARM	Yes, <i>many</i> vendors	Yes, <i>expensive</i>	Yes, <i>one</i> vendor	No (Mostly)	No
RISC-V	Yes, <i>many</i> vendors	Yes, <i>free</i>	Yes, <i>many</i> vendors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



ISA - design & fabrication (CPUs)



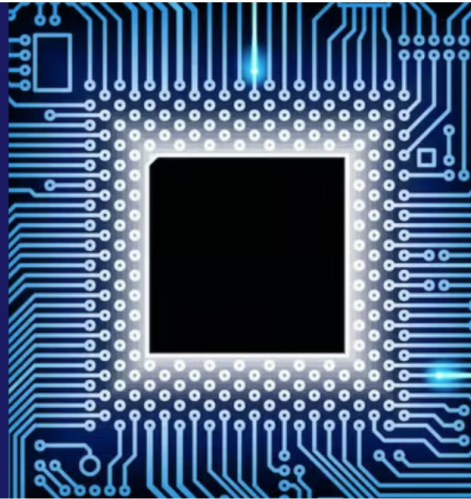
RISC-V : prime members - 2024



Asia: ~90% standardization decision
~80% of design
~90% of fabrication

RISC-V for servers - 2025

SpacemiT
Develops Server
CPU Chip V100
for Next-Gen AI
Applications



SpacemiT Vital Stone V100. The chip represents an advancement in RISC-V technology, providing a complete hardware and software platform designed to meet server specifications.



Ventana's 192-core Veryon V2 surpasses AMD's Genoa and Bergamo. Veryon V2 will be released into the market in 2025



RISC-V for laptops: SpacemiT K1 -V60

Dual-cluster Octa-core processors

RISC-V 64GCVB architecture

- Cluster0

Quad-core

2.0TOPS AI-Power

32K L1-Cache per core

512K L2-Cache

512KB TCM

Vector-256bit

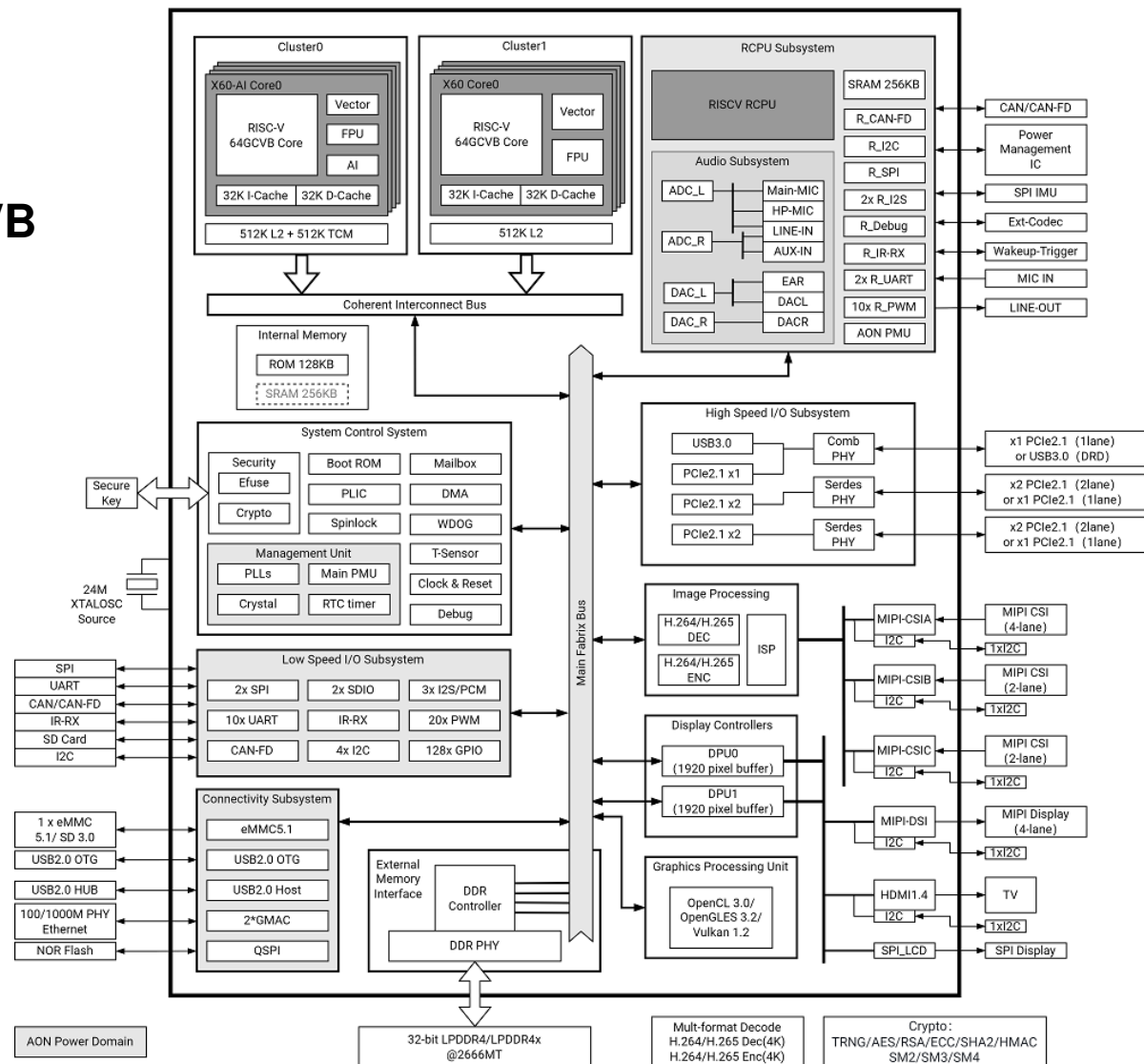
- Cluster1

Quad-core

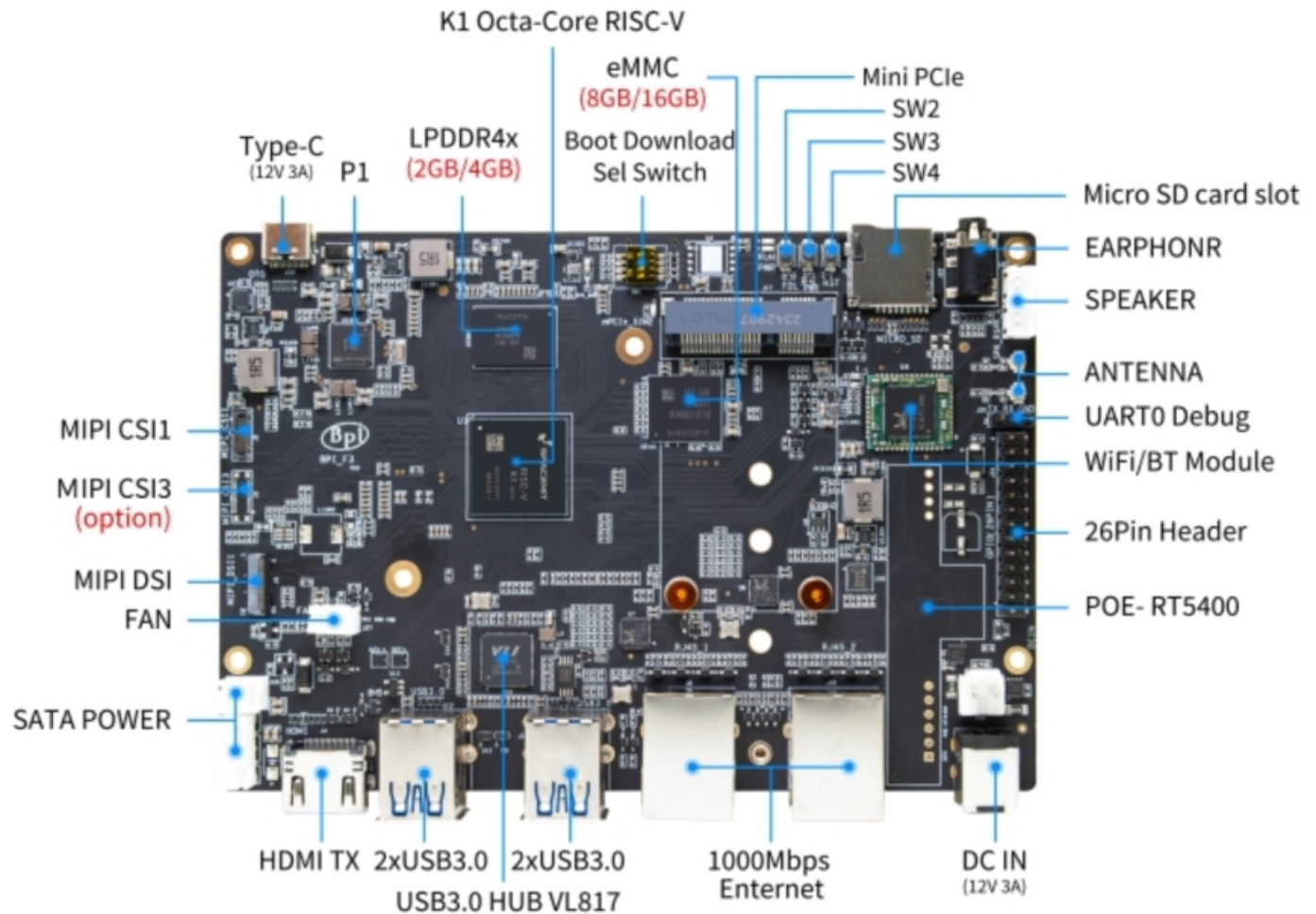
32K L1-Cache per core

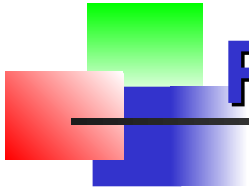
512K L2-Cache

Vector-256bit



SpacemiT K1 - SBC





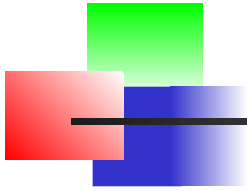
RISC-V in “few” words:

Prompt:

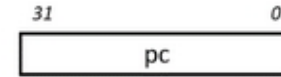
What are essential features of RISC-V architecture. Give the answer in few phrases.

Here are the essential features of the RISC-V architecture in a few phrases:

- ➡ Open-Source ISA: Free and open instruction set architecture (ISA).**
- ➡ Simplicity: Designed for efficiency and ease of implementation.**
- ➡ Modular Design: Base ISA with optional standard extensions.**
- ➡ Scalability: Supports multiple data widths (RV32, RV64, RV128).**
- ➡ Reduced Complexity: Minimal instruction set with a focus on performance.**
- ➡ Extensibility: Allows custom instructions and vendor-specific extensions.**
- ➡ Portability: Suitable for diverse applications (embedded systems to supercomputers).**
- ➡ Endianness: Supports both little-endian and big-endian formats.**
- ➡ Energy Efficiency: Optimized for low-power consumption.**
- ➡ Wide Ecosystem: Growing tools and community support.**



RISC-V ISA - registers (RV32I)



31 0

x0 / zero
x1 / ra
x2 / sp
x3 / gp
x4 / tp
x5 / t0
x6 / t1
x7 / t2
x8 / s0 / fp
x9 / s1
x10 / a0
x11 / a1
x12 / a2
x13 / a3
x14 / a4
x15 / a5

31 0

x16 / a6
x17 / a7
x18 / s2
x19 / s3
x20 / s4
x21 / s5
x22 / s6
x23 / s7
x24 / s8
x25 / s9
x26 / s10
x27 / s11
x28 / t3
x29 / t4
x30 / t5
x31 / t6

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							
<ul style="list-style-type: none">• 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

RV32IMAC

LR.W SC.W AMOAND.W AMOOR.W AMOXOR.W
 AMOADD.W AMOMIN.W AMOMAX.W AMOMINU.W AMOMAXU.W
 AMOSWAP.W

RV32A

Atomic Instruction ISA Extension

MULH DIV MUL REM REMU
 MULHU DIVU
 MULHSU

RV32M

Integer Multiplication and Division ISA Extension

ADD ADDI AND ANDI BEQ
 SLL SRL OR ORI BNE
 SLLI SRLI XOR XORI BGE
 SLT SLTU SRA LUI BGEU
 SLTI SLTIU SRAI AUIPC BLT
 LB LH LW SB BLTU
 LBU LHU SW SH JAL
 CSRRW CSRRS CSRRC ECALL JALR
 CSRRWI CSRRSI CSRRCI EBREAK SUB
 FENCE FENCE.I

RV32I

Base Integer ISA

C.LW C.AND
 C.FLW C.ANDI
 C.FLD C.OR
 C.LWSP C.XOR
 C.FLWSP C.LI
 C.FLDSP C.LUI
 C.SW C.SLLI
 C.FSW C.SRLI
 C.FSD C.SRAI
 C.SWSP C.BEQZ
 C.FSWSP C.BNEZ
 C.FSDSP C.J
 C.ADD C.JR
 C.ADDI C.JAL
 C.ADDI16SP C.JALR
 C.ADDI4SPN C.EBREAK
 C.SUB C.MV

RV32C

Compressed ISA Extension

RISC-V : Software-Hardware

high level programming

assembly level programming

USER - binaries

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

OS

I

M

A

F

D

C

V

B

S

R

I

UI

S

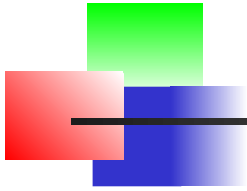
B

J

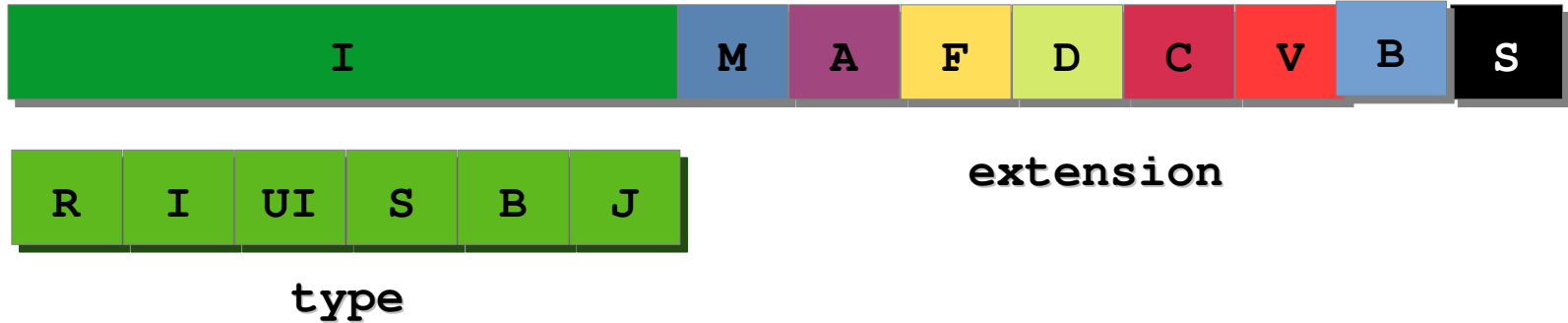
extension

type

design



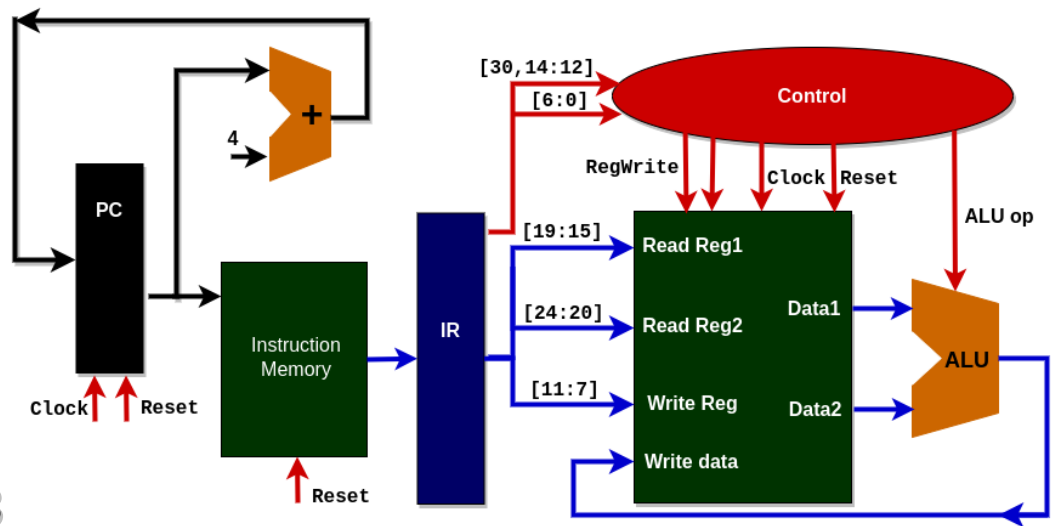
RISC-V : Software-Hardware



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
);
```

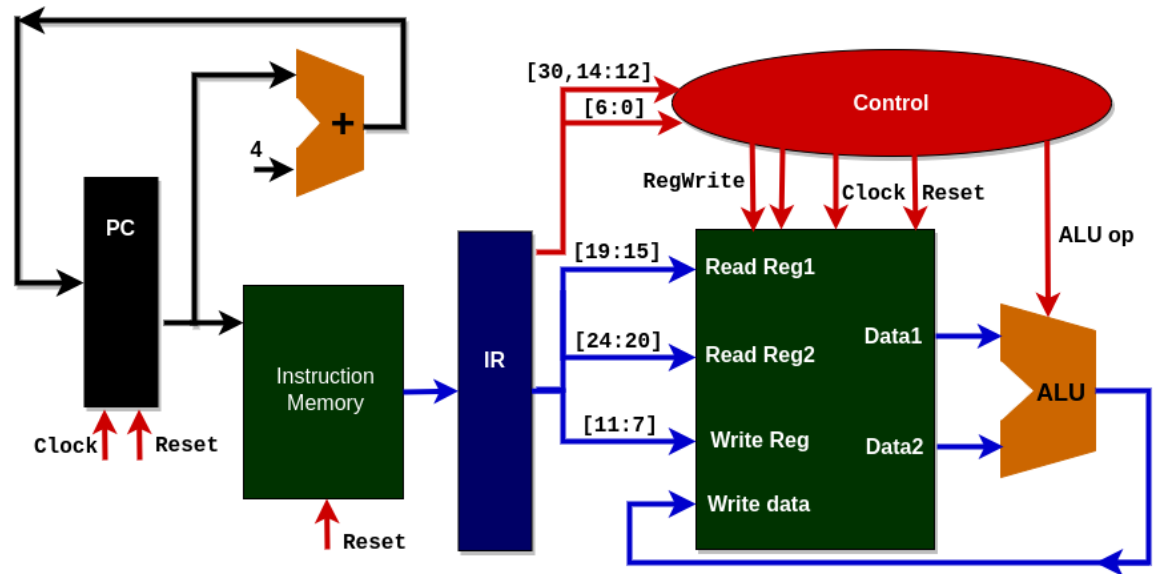
```
wire [31:0] instruction_code;
wire [3:0] alu_control;
wire regwrite;
```

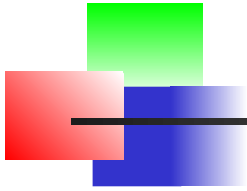
```
IFU IFU_module(clock, reset, instruction_code);
```

```
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);
```

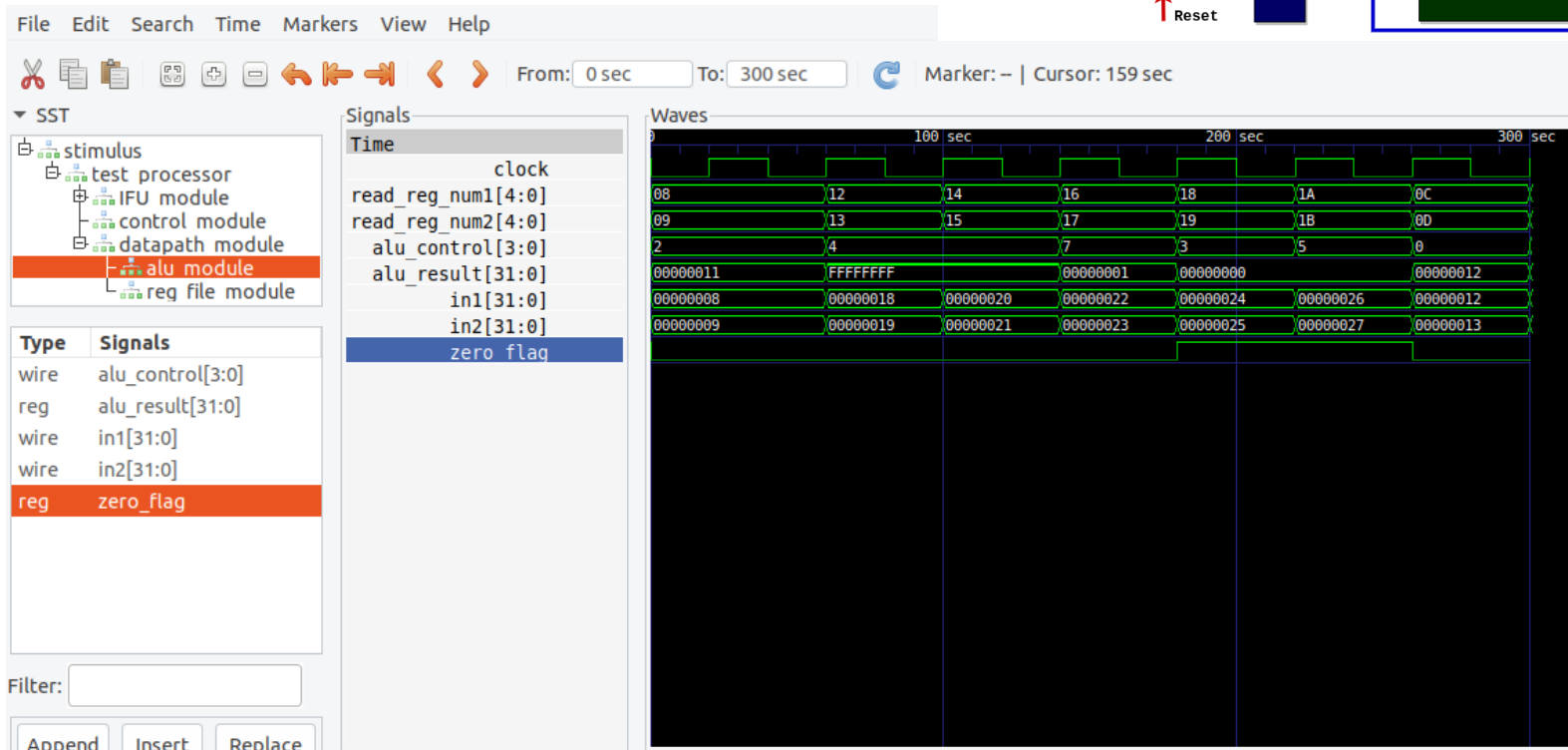
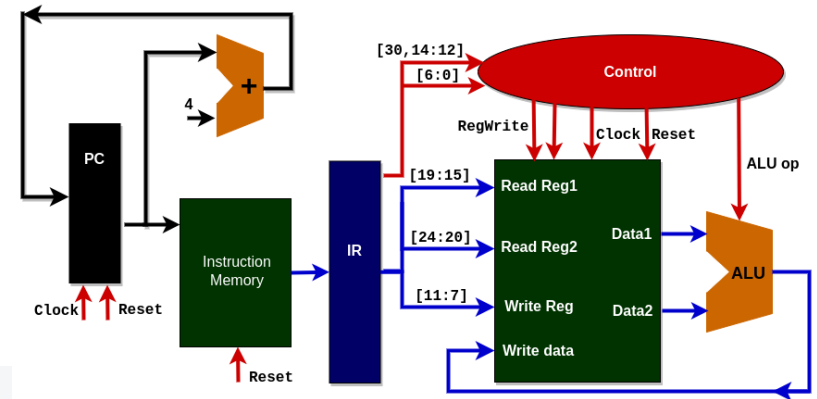
```
endmodule
```





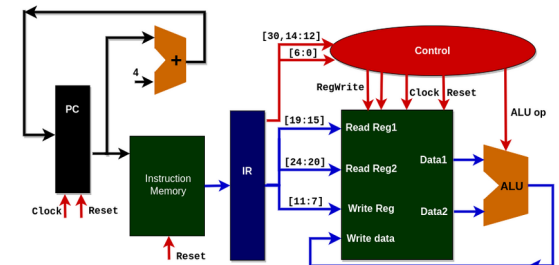
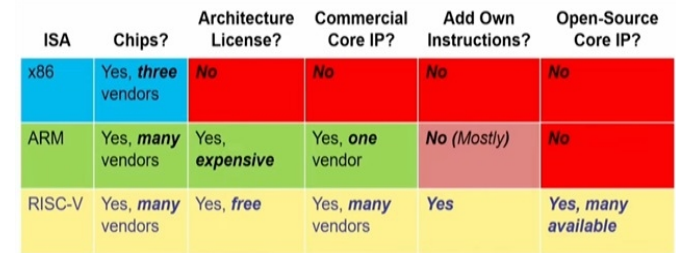
RISC-V : Software-Hardware

GTKWave waveforms



RISC-V open source & business model

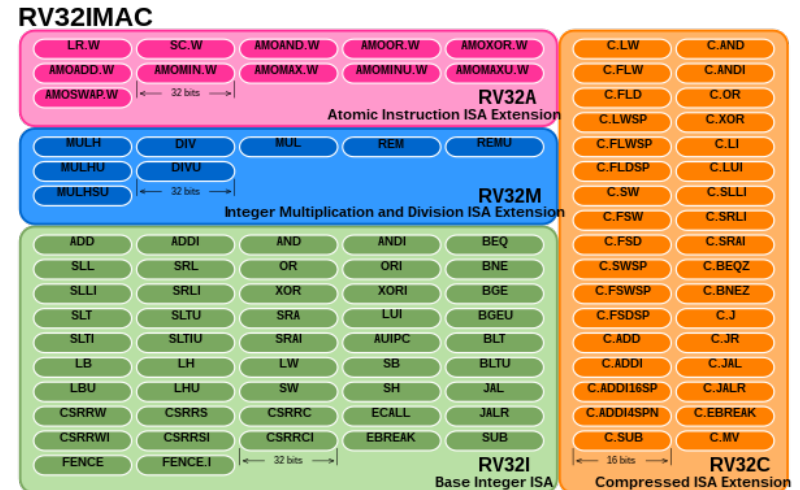
RISC-V architecture & Verilog synthesisable models



RVLabs : Plabs & MLabs

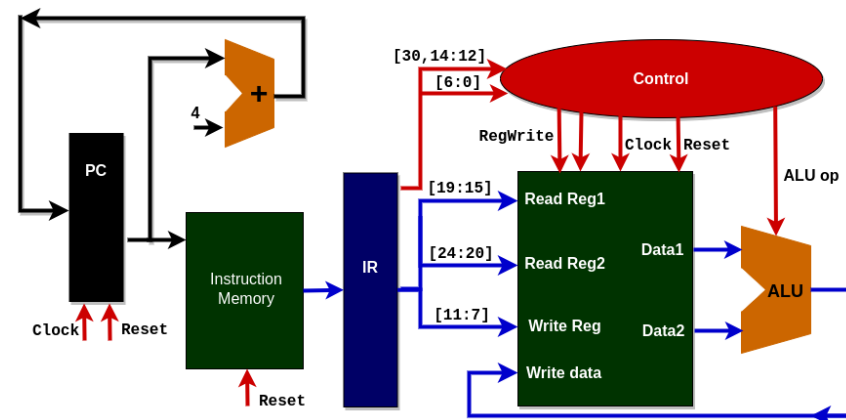
PLabs: Lab1, Lab2, Lab3, .. RISC-V ISA and assembly programming

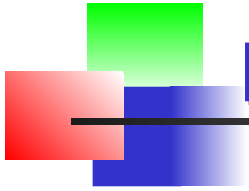
```
add    a2, a2, a3
mulw   a5, a4, a5
addi   s0, sp, 32
jr     ra
```



MLabs: Lab1, Lab2, Lab3, .. RISC-V design: architecture & Verilog models

```
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;
```





RISC-V International – November 2024

