

The background features two large, nested triangles. The triangle on the left points upwards and the one on the right points downwards. Both are filled with a vibrant galaxy image, showing swirling blue and purple nebulae and numerous bright stars. The triangles are outlined with multiple concentric lines in a gradient of dark blue, teal, and green.

# RISC-V ISA

## Chapter 1

### Unprivileged ISA Introduction

# CONTENTS



01

## Introduction

CISC and RISC

02

## Features

RISC-V features

03

## Hardware Platform

Definitions of hardware

04

## Software Execution Environment

Definition of SEE

05

## ISA Overview

Standard, custom, ISA and reserved

# CONTENTS



06

## Memory

Memory space

07

## Base Instruction-length Encoding

Instruction encodings

08

## Exceptions, Interrupts, Traps

Definitions of traps

## References

# 1.1 Introduction

## What is the difference between CISC and RISC ?

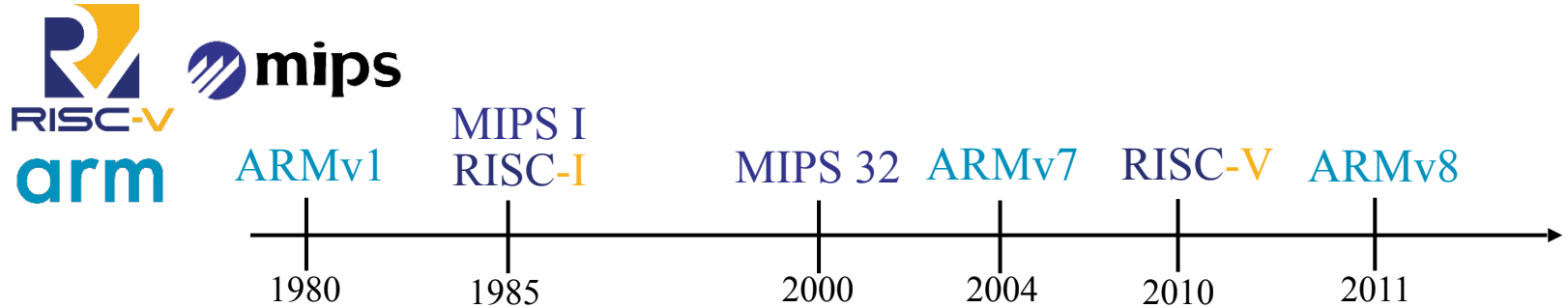
- Complex Instruction Set Computer (CISC):

A single complex instruction can **take several clock cycles to perform more complex tasks**. CISC CPU is hardware-centric design which needs **much complex hardware implementation** to supply the large instruction set.

- Reduced Instruction Set Computer (RISC):

RISC processor **performs single instruction per clock cycle**. Reducing the complexity of instructions **makes hardware implementation much easier**, and makes pipeline more streaming. RISC processors are efficient but take effort to design software program.

# 1.1 Introduction



## 1.2 Features

- Completely open and is freely available to academia and industry.  
(e.g. lowRISC: <https://lowrisc.org/>)
- Separate into **various small bases ISA**. Each base can be used for customised accelerator or educational purpose, or can be used for optional standard extensions.
- Both 32-bit and 64-bit address space variants for applications, kernels and hardware implementations.
- Support highly-parallel multicore including heterogeneous implementations.  
(e.g. SiFive U74 Core has 8-stage pipeline, SiFive U8 has 12-stage pipeline)

## 1.3 Hardware Platform

- A RISC-V *hardware platform* can contain **more than one RISC-V compatible cores and non-RISC-V compatible cores**. Also, it can contain **accelerators**, various **physical memory** structures, **I/O devices** and an **interconnect**.
- A term of *core* is referred to the component contained instruction fetch unit. It can **support multiple threads**, also called *harts*, to implement multithreading.
- A term of *accelerator* is referred to either non-programmable units or a core that can autonomously operate for certain tasks, e.g. the I/O accelerators can offload I/O processing tasks from the main cores.





# 1.4 Software Execution Environment

RISC-V **execution environment interface (EEI)** defines the initial state of the **program**. It can **support multiple threads**, also called ***harts***, to implement multithreading.

Some example of execution environments are listed as follows:

- **Bare Metal** is a pure hardware platform where the harts are implemented by physical processor thread.



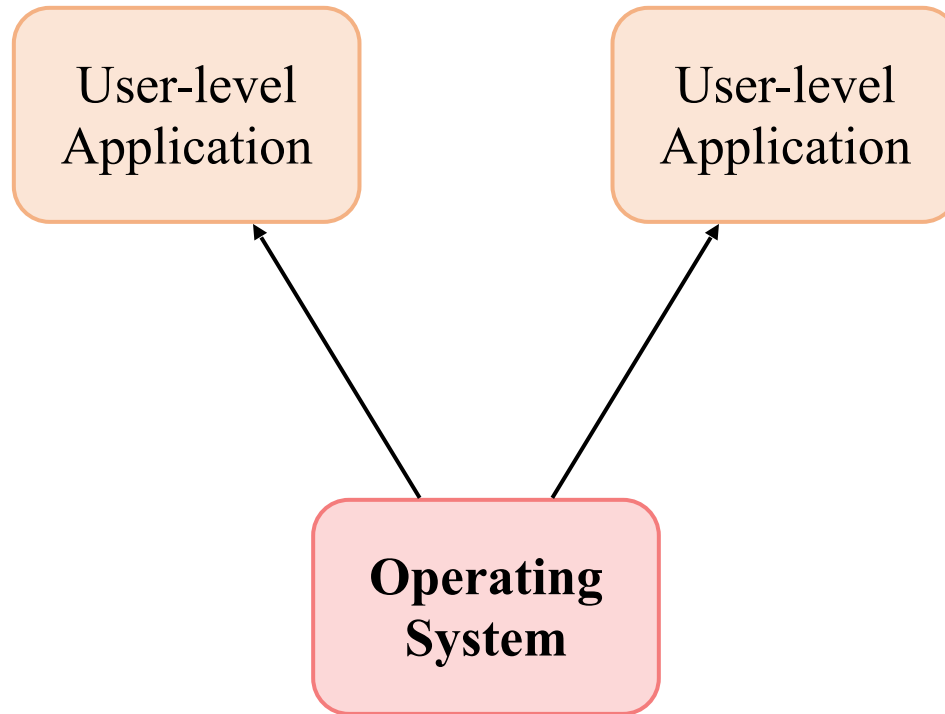
**Bare Metal**

(Physical processor thread)



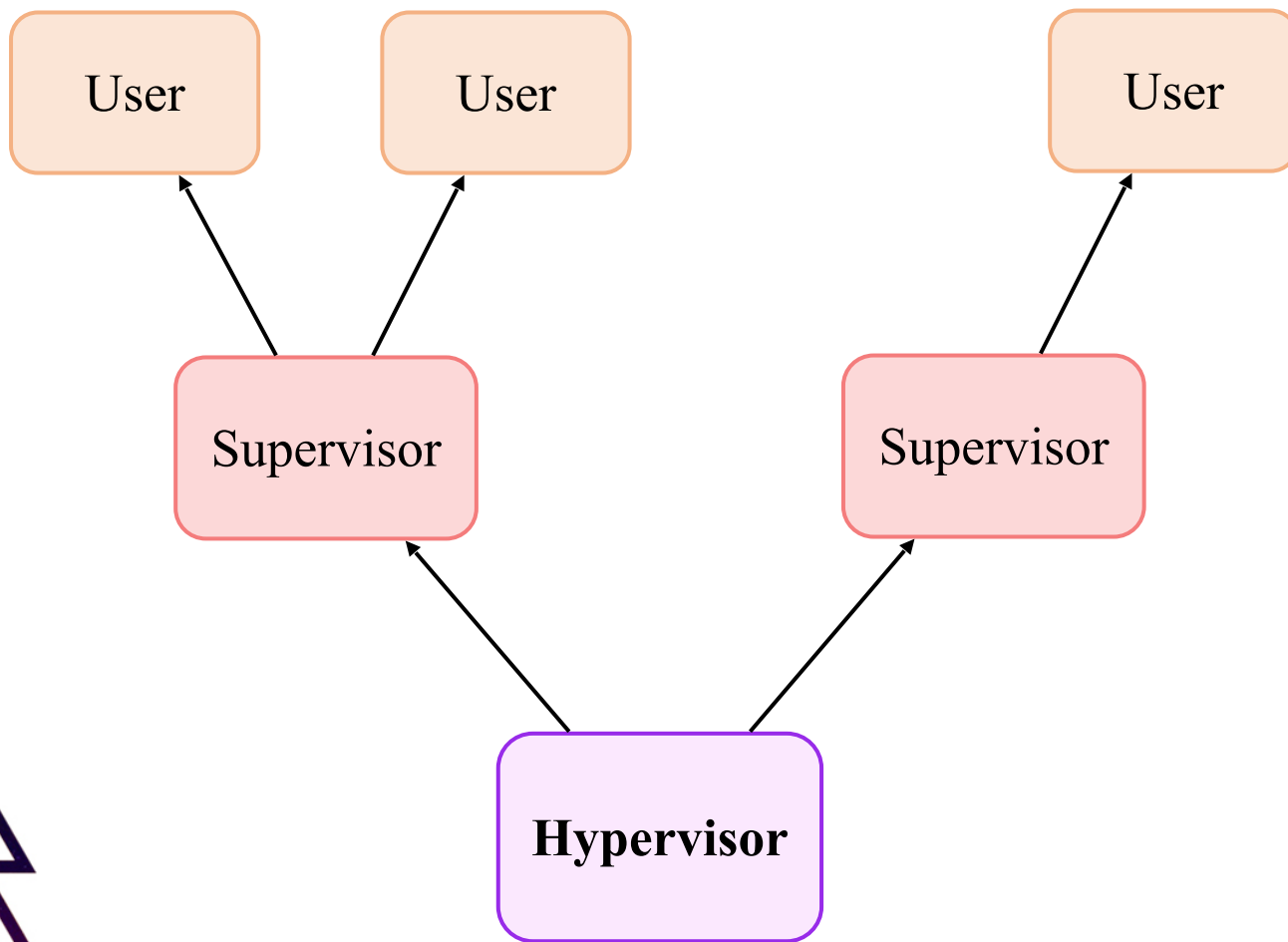
## 1.4 Software Execution Environment

- RISC-V **operating system** that provide multiple user-level execution environments.



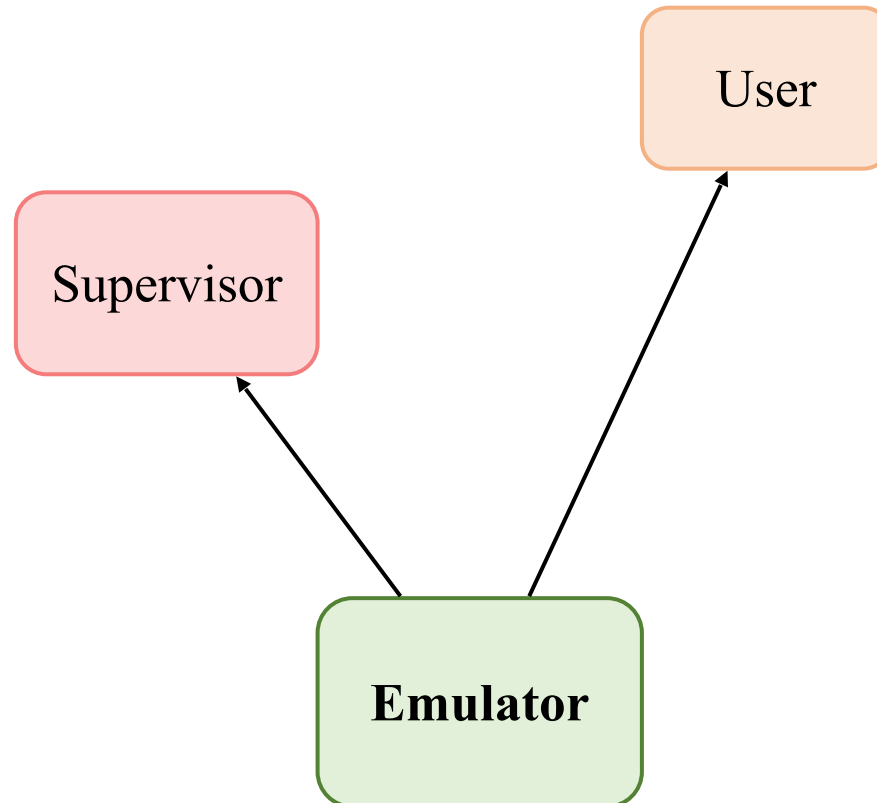
## 1.4 Software Execution Environment

- RISC-V **hypervisor** provides multiple supervisor-level execution environments for guest operating systems.



## 1.4 Software Execution Environment

- RISC-V **emulator** that emulate RISC-V harts on an underlying x86 system, it can provide either user-level or supervisor-level execution environment.



## 1.5 ISA Overview

A RISC-V ISA is defined as a **base integer ISA**, a base is restricted to be a minimal set of instructions sufficient to provide reasonable targets including compilers, assemblers, linkers and operating systems.

Use *XLEN* value in status register to refer to the width of an integer registers. Base integer can be extended with the optional **instruction-set extension**, which categorised as follows:

- **Standard:**  
Defined by the Foundation, only use standard encodings.
- **Reserved:**  
Not defined by the Foundation currently, but it is saved for future extensions.
- **Custom:**  
Only used for vendor-specific non-standard extensions.

## 1.6 Memory

A RISC-V hart has a single address space of  $2^{XLEN}$  bytes. The memory address space is circular, i.e. address at  $2^{XLEN}-1$  is near address at 0. And RISC-V computation ignores overflow, but wrap around instead.

Different memory address space determines the mapping of hardware resources as vacancy, main memory, and I/O devices.

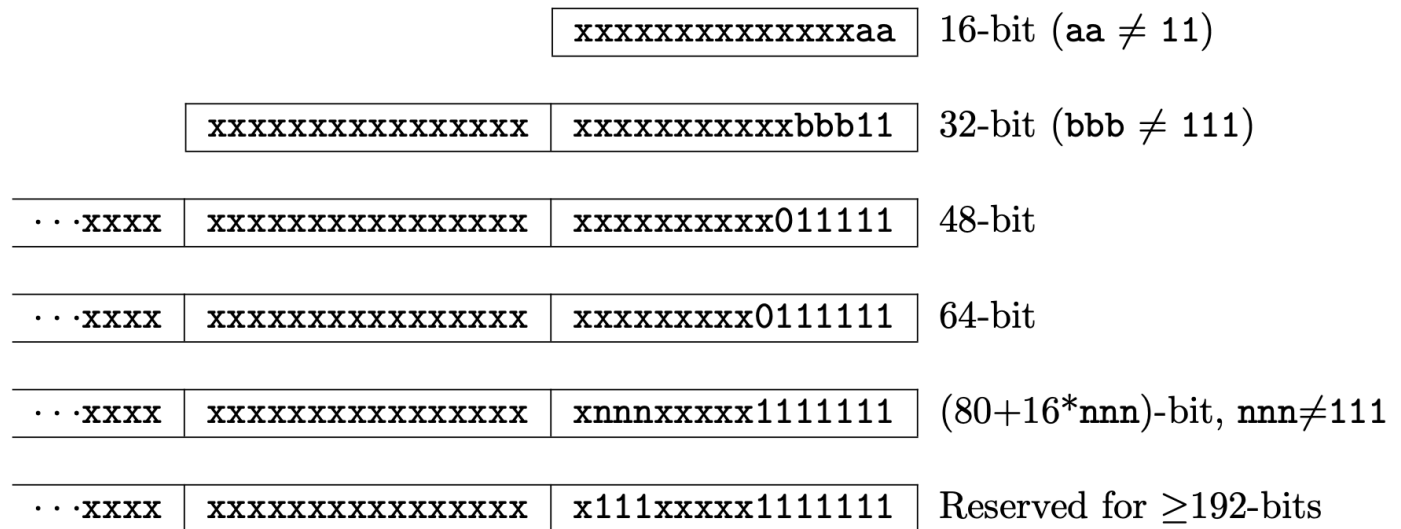
When a RISC-V platform has more than one harts, the address space of the harts can be entirely the same, or completely different but sharing some resources.



## 1.7 Base Instruction-length Encoding

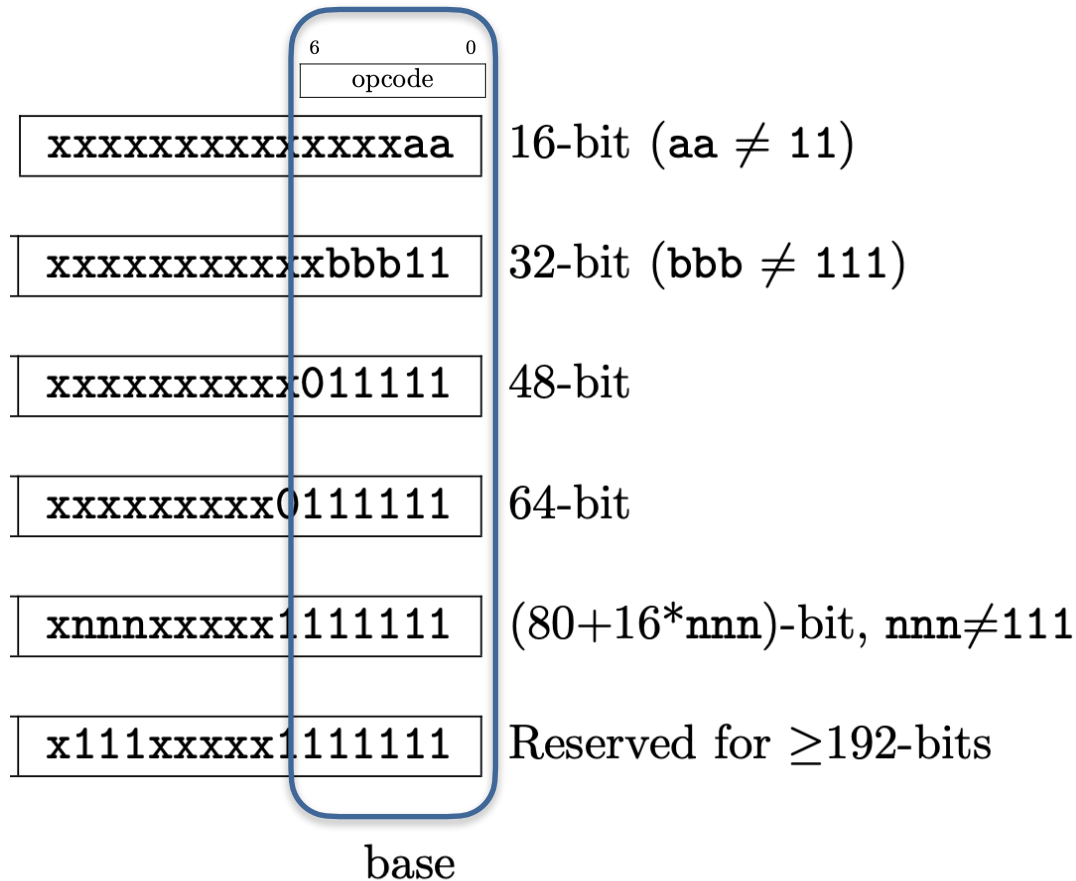
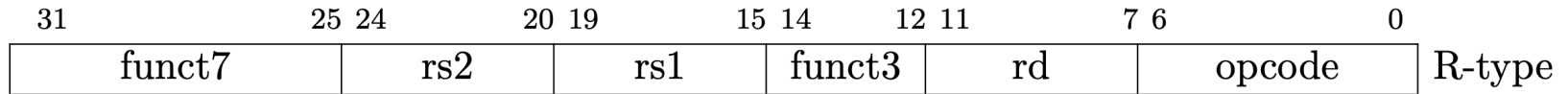
The base RISC-V ISA has fixed-length instructions that **must be aligned the boundaries**. *IALIGN* refers to the instruction alignment constraint, in addition, RISC-V ISA supports many **fixed-length base-integer instructions** as well as the **variable-length extensions**.

*ILEN* refers to the maximum instruction length supported by an implementation.



Byte Address:      base+4                      base+2                      base

# 1.7 Base Instruction-length Encoding





# 1.8 Exceptions, Interrupts, Traps

- **Exception:**

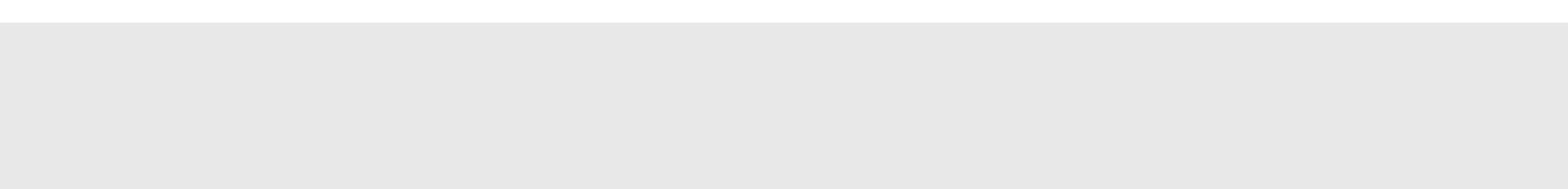
Refer to an **unusual condition** at run time in the current hart.

- **Interrupt:**

Refer to an external **asynchronous event** that causes RISC-V hart to execute interrupt service routine (ISR).

Interrupt	Exception Code	Description
1	0	User software interrupt
1	1	Supervisor software interrupt
1	2	<i>Reserved for future standard use</i>
1	3	Machine software interrupt
1	4	User timer interrupt
1	5	Supervisor timer interrupt
1	6	<i>Reserved for future standard use</i>
1	7	Machine timer interrupt
1	8	User external interrupt
1	9	Supervisor external interrupt
1	10	<i>Reserved for future standard use</i>
1	11	Machine external interrupt
1	12–15	<i>Reserved for future standard use</i>
1	≥16	<i>Reserved for platform use</i>

0	0	Instruction address misaligned
0	1	Instruction access fault
0	2	Illegal instruction
0	3	Breakpoint
0	4	Load address misaligned
0	5	Load access fault
0	6	Store/AMO address misaligned
0	7	Store/AMO access fault
0	8	Environment call from U-mode
0	9	Environment call from S-mode
0	10	<i>Reserved</i>
0	11	Environment call from M-mode
0	12	Instruction page fault
0	13	Load page fault
0	14	<i>Reserved for future standard use</i>
0	15	Store/AMO page fault
0	16–23	<i>Reserved for future standard use</i>
0	24–31	<i>Reserved for custom use</i>
0	32–47	<i>Reserved for future standard use</i>
0	48–63	<i>Reserved for custom use</i>
0	≥64	<i>Reserved for future standard use</i>



# 1.8 Exceptions, Interrupts, Traps

**Trap** is referred to **transfer of control**, the trap handler may be caused by either exception or interrupt. As for execution environment, traps can have following effects:

- **Contained Trap:**

Refer to an **unusual condition** at run time in the current hart.

- **Requested Trap:**

The trap is **synchronous exception requesting an action** on behalf of software inside the execution environment, e.g. system call.



# 1.8 Exceptions, Interrupts, Traps

**Trap** is referred to **transfer of control**, the trap handler may be caused by either exception or interrupt. As for execution environment, traps can have following effects:

- **Invisible Trap:**

The trap is **handled invisibly** by the execution environment, then the execution resumes after the trap handled, e.g. page fault.

- **Fatal Trap:**

The Trap represents a **fatal failure and causes the execution environment terminated**, e.g. page-protection failure or watchdog timer expired.



# References

- [1] (2020) The RISC-V Instruction Set Manual, Volume I - Unprivileged ISA.
- [2] (2020) The RISC-V Instruction Set Manual, Volume II - Privileged ISA.
- [3] (2018) RISC-V - An Overview of the Instruction Set Architecture.

**THANKS**  
for your  
**ATTENTION**

