

# Application Status and Standard Requirements of RISC-V Chips in the Automotive Field

Zhantong Tu <sup>1</sup>, Yujia Li <sup>1</sup>, Zheng Li <sup>2</sup>, Mingkai Yan <sup>1,\*</sup>, Rui Zhao <sup>1</sup>

<sup>1</sup> CATARC Automotive Chip (Shenzhen) Technology Co., Ltd., Shenzhen, Guangdong, 518100, China

<sup>2</sup> China Automotive Technology and Research Center Co., Ltd., Tianjin, 300300, China

\* Corresponding author: Mingkai Yan

## Abstract

In recent years, with the rapid development of the automotive industry toward intelligence, connectivity, and electrification, automotive chips, as the core carriers for implementing related functions, have seen continuously rising computing power demands, while architecture openness and autonomous controllability have also gained increasing attention. Against this backdrop, RISC-V technology, based on an open instruction set architecture, offers a new development path for the automotive chip industry through its modularity, extensibility, and autonomous controllability. However, when entering the automotive field with stringent functional safety requirements, the RISC-V architecture still faces multiple challenges, including incomplete standards, a weak ecosystem, and insufficient safety and reliability verification. This paper elaborates on the development background and research significance of automotive chip architectures, analyzes the advantages of traditional chip architectures versus the RISC-V architecture, the current application status of RISC-V, and the challenges it faces. Focusing on standards, it analyzes deficiencies in the current standard system, points out the importance and urgency of unified RISC-V standards for automotive deployment, and provides reference guidance for the large-scale, standardized application of RISC-V architecture in the automotive field.

## Keywords

RISC-V Chips; Automotive Field; Functional Safety.

## 1. Introduction

At present, the development focus of the automotive industry is gradually shifting away from traditional mechanical core components such as engines and transmissions toward intelligent in-vehicle architectures, while the pace of electrification and connectivity upgrades continues to accelerate. Automotive chips are a prerequisite for supporting the implementation of these technologies and are also the core hardware driving the transformation and upgrading of the automotive industry. Automotive chips, also known as automotive-grade chips, are semiconductor integrated circuit devices specifically developed and manufactured for automotive electronic systems, and must strictly comply with various stringent automotive-grade industry specifications and technical standards. As real-world application scenarios and functional requirements for vehicles become increasingly complex, the computing load and performance demands on automotive chips continue to rise. At the same time, the openness of chip architectures and the capability for domestic autonomous controllability have become key focus areas in current industrial research and applications. Against this industry backdrop, the RISC-V open-source instruction set architecture, with its modular design, strong extensibility, and high degree of autonomous controllability, has opened up a new technical path for the

innovative development of the domestic automotive chip industry. However, the automotive industry has extremely high functional safety entry barriers, and the industrial-scale deployment of RISC-V architecture in the automotive field still faces many practical bottlenecks, mainly reflected in incomplete supporting industry standards, lagging industrial ecosystem development, and inadequate safety and reliability verification systems.

Addressing the above industry pain points and technical bottlenecks, this paper systematically reviews the evolution and development background of automotive chip architectures, deeply analyzes the technical advantages, existing challenges, and future development trends of applying RISC-V architecture in the automotive field, and further clarifies the necessity of deploying this architecture in the automotive industry. Meanwhile, it briefly summarizes the comprehensive RISC-V standard system framework at the current stage, including instruction set specifications, profile standards, hardware technical specifications, and compatibility testing systems, and systematically summarizes their standard coverage and deployment status. Combined with the specific technical and functional requirements of various automotive chip application scenarios, it identifies gaps and shortcomings in the existing RISC-V standard system. Building on the aforementioned research and analysis, it conducts systematic requirement assessment from the perspectives of basic indicator specifications and general technical requirements, clarifying the necessity and practical application demands for RISC-V chip standardization in automotive scenarios, and providing theoretical reference and directional guidance for the subsequent construction and improvement of a dedicated RISC-V standard system for the automotive field.

## 2. Research Background of Automotive Chip Architecture

Under current industry trends, traditional distributed control architectures can no longer accommodate the continuously growing demands for in-vehicle computing power, data throughput, and software iteration updates. The evolution of automotive electrical/electronic (E/E) architectures has gone through two major stages — discrete distributed architecture and logically centralized domain control architecture — and is gradually iterating toward physically centralized zone architectures and central computing architectures. In the development of automotive electronics, in-vehicle electronic systems have long followed an additive development model of "each new function corresponds to a new Electrical Control Unit (ECU)." This distributed architecture has caused the number of in-vehicle ECUs to rise sharply, not only resulting in highly fragmented computing resources but also causing a series of problems such as redundant wiring harnesses and complex routing. Under the distributed E/E architecture model, point-to-point signal connections not only increase the overall weight of in-vehicle wiring harnesses but also raise vehicle R&D and manufacturing costs. Furthermore, the wiring harness system formed by the distributed architecture has a complex structure and poor adaptability, significantly limiting improvements in vehicle production efficiency. To address the technical drawbacks of the distributed architecture, major automakers have gradually introduced domain-centralized architectures. This architecture logically divides vehicle functions into five core functional domains: powertrain, chassis, body, smart cockpit, and intelligent driving. Under the domain-centralized architecture, a Domain Control Unit (DCU) can integrate various in-vehicle functions for the corresponding functional domain, consolidating control logic originally distributed across multiple low-level ECUs and reducing the number of low-level electronic control units deployed. Although the domain-centralized architecture initially shows a trend of localized computing concentration, it has not completely resolved the wiring harness problem.

Against this backdrop, in-vehicle chip architectures are gradually iterating toward a new form of "central computing + zone control." This architecture system is built around one or more

central computing units, paired with several Zone Control Units (ZCUs). Under this zonal architecture, the vehicle abandons the traditional functional division approach and instead organizes layout according to physical space within the vehicle body, with each physical zone configured with an independent zone controller responsible for overall management. The ZCU connects all sensors and actuators within its physical zone, regardless of their functional domain. Each zone controller aggregates and integrates various in-vehicle signals before transmitting them uniformly to the central computing unit via a high-speed Ethernet backbone network. This design fundamentally changes the wiring harness topology from a "star + bus" structure to a "backbone + branch" structure.

Furthermore, in the iteration and upgrade of automotive E/E architectures, Software-Defined Vehicle (SDV) and Service-Oriented Architecture (SOA) have become the core design philosophies of the new generation of in-vehicle electronic systems, reconstructing the overall technological development system of the automotive industry. SDV achieves higher system programmability by migrating functional logic to the software layer. This transformation has raised new core requirements for the in-vehicle hardware system, primarily manifested in the trend toward hardware standardization and generalization: automotive chips are gradually moving away from the custom development model for single functions, evolving into heterogeneous integrated computing platforms with high performance, high security, and low power consumption.

SOA imposes more stringent determinism and throughput requirements on the processing capabilities of underlying chips. Under the SOA architecture, in-vehicle systems must process high-concurrency asynchronous events and various service requests at high frequency, which places strict demands on processor interrupt response determinism and fast task scheduling and switching capabilities to ensure real-time performance and data consistency during cross-service interactions. At the same time, complex system task scheduling mechanisms require higher performance from automotive chips, necessitating excellent multi-core cooperative scheduling, memory management, and cache coherence maintenance capabilities to effectively resolve resource contention conflicts during parallel multitask operation.

The above functional requirements not only raise higher design standards for in-vehicle software development systems but also present new challenges to the comprehensive performance of underlying chip architectures, fully exposing the limitations of traditional chip systems in architectural flexibility, functional extensibility, and overall cost-effectiveness. Furthermore, the deep evolution of automotive E/E architectures from fragmented layouts to highly integrated forms imposes even more stringent technical standards on underlying automotive chips, requiring them to have high computing density, excellent heterogeneous cooperative performance, and reliable functional safety isolation capabilities. Against this backdrop, the RISC-V architecture, leveraging its unique advantages of modular design, customizable development, and open-source instruction set architecture (ISA), has become an excellent underlying technology foundation for achieving in-vehicle software-hardware decoupling. The modular nature of RISC-V supports developers in conducting differentiated custom development for specific task scenarios while maintaining ISA standardization, which can effectively reduce chip R&D licensing costs and significantly shorten the overall development cycle from chip definition to functional implementation. With these core technical advantages, the RISC-V architecture can break through the technical constraints of traditional chip architectures, providing a solid underlying technical foundation for the innovation and reshaping of the new generation of automotive E/E architectures.

### 3. RISC-V

#### 3.1. Research Background

The Instruction Set Architecture (ISA), as an abstract model of computer architecture, defines the programming interface between software and hardware and constitutes the fundamental specification for CPU design[1]. It specifies the executable base instruction set, register architecture, memory access rules, and exception handling mechanisms of a processor, establishing a standardized interaction interface between the software and hardware of computer systems. Throughout the history of computing, instruction set architectures can be broadly categorized into two major technical paths: Complex Instruction Set Computing (CISC) and Reduced Instruction Set Computing (RISC)[2].

The core of CISC lies in directly encapsulating common high-level language operations through complex hardware-level instructions, enabling a single instruction to perform multiple tasks such as loading, computing, storing, and even loop control, thereby reducing compiler design complexity and improving code density. This design was originally intended to alleviate the bottleneck of extremely scarce memory in early computing, but it introduced problems such as variable instruction execution cycles, lengthy control logic, and high power consumption.

RISC is an instruction set architecture that pursues ultimate efficiency by simplifying hardware logic and strengthening software optimization. Its core principle is to retain only frequently used base instructions with fixed execution cycles (such as load, store, add, subtract, and compare), and requires all arithmetic and logic operations to be performed only between general-purpose registers within the CPU, with memory access handled independently by dedicated "load/store" instructions, thereby eliminating the complex and inefficient compound operations of CISC.

In the current global chip industry, the mainstream instruction set architectures are shown in Table 1, mainly including: x86, MIPS, ARM, and RISC-V.

**Table 1.** Mainstream Instruction Set Architectures

| Architecture | Developer/Organization | Release Year | Key Features                            |
|--------------|------------------------|--------------|-----------------------------------------|
| X86          | Intel Corporation, USA | 1978         | High performance, good compatibility    |
| MIPS         | MIPS Technologies, USA | 1981         | Simple, high energy efficiency          |
| ARM          | ARM Ltd., UK           | 1983         | Low power consumption, mature ecosystem |
| RISC-V       | RISC-V Foundation      | 2011         | Modular, open-source                    |

The instruction set market is dominated by two major architectures: x86 and Arm. x86, as the representative of CISC, dominates the personal computer and server markets; Arm, as a typical RISC architecture, holds an absolute advantage in mobile devices and embedded systems due to its low-power characteristics. However, traditional architectures have limitations: x86 has relatively high power consumption and its licensing is subject to restrictions, while Arm, despite lower power consumption, has expensive licensing fees and its technology is controlled by a single company. The RISC-V architecture, with its openness, has broad application prospects.

#### 3.2. RISC-V Development History

RISC-V was created by the University of California, Berkeley in 2010, aiming to develop a simplified, modular instruction set architecture to address the limitations of CISC architectures

at the time. In 2011, the first version of RISC-V, RV32I, was released as a 32-bit base integer instruction set, primarily focused on education and academic research. In 2015, RISC-V introduced several extension instruction set versions covering basic functions such as multiplication, division, atomic operations, and floating-point operations. In the same year, the RISC-V Foundation began preparations. In 2017, RISC-V gradually entered the commercial sector, with some companies starting to design and manufacture processors and chips based on this architecture. In 2018, the RISC-V ISA continued to evolve, adding vector extensions (RVV), security extensions (RVSEC), and multi-core extensions (RVMT), with the ecosystem growing rapidly. With its free, open, and easily extensible characteristics, RISC-V has quickly risen to become one of the world's three major instruction set architectures.

### 3.3. RISC-V Instruction Set

#### 3.3.1. Overview

The RISC-V instruction set consists of a base instruction set and several optional extension instruction sets, where every RISC-V instruction set architecture must adopt one base integer instruction set.

##### (1) RISC-V Base Instruction Set

RISC-V includes five base integer instruction sets: RVWMO (Weak Memory Ordering instruction set), RV32I (32-bit integer instruction set), RV64I (64-bit integer instruction set), RV32E (32-bit embedded integer instruction set), and RV128I (128-bit integer instruction set). Among these, RV32I, RV64I, and RV128I all use 32 general-purpose registers and one program counter (PC), with the three differing only in register width.

##### (2) RISC-V Extension Instruction Sets

RISC-V extension instruction sets are designed to provide specific functions to the ISA. A designed ISA can include multiple extension instruction sets without conflicts. Currently, there are 24 types of RISC-V extension instruction sets in total, including atomic instruction extension, multiplication/division extension, single-precision floating-point extension, double-precision floating-point extension, quad-precision floating-point extension, compressed instruction extension, counter instruction extension, decimal floating-point extension, bit manipulation extension, dynamic translation language extension, transactional memory extension, combined SIMD extension, vector operation extension, control and status register extension, barrier extension, misaligned atomic operation extension, total store ordering extension, pause hint extension, half-precision floating-point extension, half-precision floating-point minimal extension, integer single-precision floating-point extension, integer double-precision floating-point extension, integer half-precision floating-point extension, and integer half-precision floating-point minimal extension[2].

This modular design allows developers to flexibly combine instruction sets according to application requirements, either minimizing the implementation to reduce hardware cost or leveraging extension instructions to support high-performance computing. For automotive applications, this flexibility enables chip vendors to use a minimal instruction set for body control modules, while configuring vector extensions and AI acceleration instructions for smart cockpit and autonomous driving domain controllers[3].

#### 3.3.2. RISC-V Instruction Set Characteristics

The main characteristics of the RISC-V instruction set can be summarized as follows[4]:

(1) Modularity: The base instruction set is RV32I, and its flexibility derives from extension instruction sets; extension instruction sets serve as supplements to base functions, allowing designers to choose whether to use specific extensions as needed.

(2) Simplicity: Its base instruction set contains only about 40 instructions, which significantly reduces the learning and implementation difficulty compared to traditional architectures with

hundreds of instructions; the simplified instruction decoding and pipeline design also improves processor efficiency and operating frequency.

(3) Openness: RISC-V is a free and open-source instruction set architecture. This openness and freeness can avoid high patent fees, reduce R&D costs, and quickly attract global developers to participate, thereby driving continuous optimization and evolution of the instruction set.

It is precisely these technical characteristics that give RISC-V enormous application potential in emerging fields such as the Internet of Things, artificial intelligence, and automotive electronics, while presenting strategic opportunities for China to break through the "chokepoint" dilemma in the chip industry.

### 3.4. Overview of RISC-V Standards

RISC-V international standardization efforts are primarily led by RISC-V International. Its Technical Steering Committee (TSC) provides guidance for the aforementioned standards and specifications and approves their formal adoption and release. Current RISC-V technical specifications can be divided into four major categories: ISA Specification, Profiles, Non-ISA, and Compatibility Test Framework, as shown in Figure 1[5].

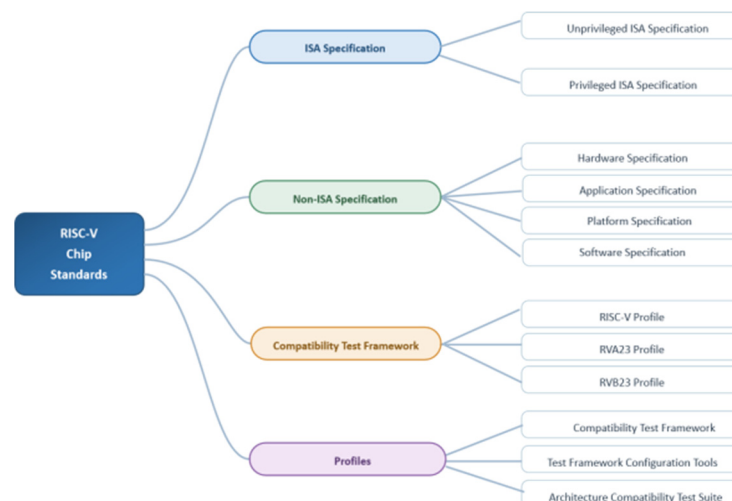


**Figure 1.** RISC-V Chip Standards Overview

#### 3.4.1. Instruction Set Architecture Specification

The RISC-V ISA specification is divided into two volumes: Volume 1 describes the base unprivileged instructions and optional unprivileged extensions, and Volume 2 covers the privileged instruction specification. According to the unprivileged specification, the RISC-V ISA consists of a mandatory base integer instruction set and several optional architecture extensions. Even as various extensions are iterated, the base integer instruction set remains complete and continuously usable as an independent instruction set. The RISC-V privileged architecture is uniformly defined by the privileged instruction specification, which covers all system-level functions beyond the unprivileged instruction set, primarily including the privileged instruction system and various extended functions needed to support operating system scheduling and peripheral data interaction. This specification defines a three-level privilege mode model:

(1) Machine mode (M mode): The highest privilege level, which manages core hardware resources; firmware running in this mode has complete system control.

(2) Supervisor mode (S mode): Primarily hosts the operating system kernel, providing a unified execution environment to upper layers through the standardized Supervisor Binary Interface (SBI).

(3) User mode (U mode): The lowest privilege level, used for executing ordinary application programs. This privileged specification, together with the unprivileged instruction specification, constitutes the complete RISC-V ISA.

### 3.4.2. Profile Specification

The RISC-V architecture supports personalized custom development based on the base instruction set architecture. This capability gives developers the ability to configure specialized instruction sets on demand and finely optimize chip customization solutions, but it also significantly increases the complexity and redundancy of instruction set selection. To address these issues, Profiles establish lightweight, general-purpose ISA selection standards, clearly defining the minimal instruction set collection that a system must be compatible with for operation, effectively standardizing the scope of instruction set configuration and avoiding architecture redundancy. This standardized specification can guide the software ecosystem toward a unified development system, greatly improving the adaptability and portability of operating systems and upper-layer applications across various hardware platforms. RISC-V architecture Profiles define the overall architecture structure and standardize the content of various profiles defined by working groups. Among these, RVI20 is a general-purpose RISC-V unprivileged software profile, RVA20 and RVA22 target application processor architecture scenarios, while the RVA23 profile is used to unify implementation standards for 64-bit RISC-V application processors, leveraging large-scale stable extension functions and a small number of configurable coarse-grained options to improve binary software ecosystem compatibility.

### 3.4.3. Non-ISA Specification

Non-ISA specifications do not add new instructions, do not create or modify opcodes, and do not alter the RISC-V instruction set in any way; their role is to build the RISC-V ecosystem around the RISC-V ISA. These specifications can be divided into four types:

- (1) Hardware specifications: Including I/O Memory Management Unit (IOMMU), Platform-Level Interrupt Controller (PLIC), Advanced Interrupt Architecture (AIA), and Server SoC specifications.
- (2) Software specifications: Including Debug-related specifications, Trace-related specifications, and Reliability, Availability, and Serviceability (RAS) specifications.
- (3) Platform specifications: Including UEFI protocol specifications applicable to RISC-V architecture, Semihosting specifications, Platform Management Interface (RPMI) specifications, and Boot and Runtime Services (BRS) specifications.
- (4) Application specifications: Including Application Binary Interface specifications and Vector C Intrinsics interface specifications.

### 3.4.4. Compatibility Test Framework

The Compatibility Test Framework (RISCOF) is designed to test and verify the compatibility of RISC-V processors. This framework provides a set of commands and tools to execute various tests and compare results against a reference model, thereby verifying processor correctness and compliance. RISCOF aims to build a comprehensive testing framework that, on one hand, ensures RISC-V processors conform to the architecture specification, and on the other hand, guarantees correct execution of their instructions and functions.

## 3.5. Current Application Status of RISC-V

The current development of instruction set architectures presents three major trends: diversification, specialization, and openness. As the pace of Moore's Law iteration gradually slows, various computing application scenarios become increasingly segmented, and the traditional general-purpose architecture design model can no longer accommodate the increasingly diverse and differentiated computing power demands. Particularly in emerging fields such as artificial intelligence, edge computing, and intelligent vehicles, higher

requirements have been placed on specialized optimization of computing architectures[6]. From a technological development perspective, heterogeneous computing has become the mainstream trend in the iteration and evolution of instruction set architectures. Contemporary chip design has abandoned the single general-purpose core architecture model, instead integrating multiple types of computing units within a single chip to build a heterogeneous computing architecture of "general-purpose computing cores + dedicated acceleration cores" tailored to different data types such as scalar, vector, and tensor, thereby achieving efficient parallel computation of various data tasks[7].

Open standards are reshaping the chip industry landscape. The open nature of RISC-V lowers the barrier to chip design, enabling more companies to participate in high-end chip R&D. According to statistics, global shipments of RISC-V-based chips have exceeded 10 billion units, and it is projected that by 2030, its market share in the consumer electronics sector will reach 36%. This openness brings new innovation models to the global chip industry, and especially amid increasing geopolitical uncertainty, RISC-V has become an important path for countries to achieve technological independence.

### 3.6. Challenges Facing RISC-V

Although RISC-V has broad application prospects, its large-scale deployment still faces multiple challenges, mainly reflected as follows[8]:

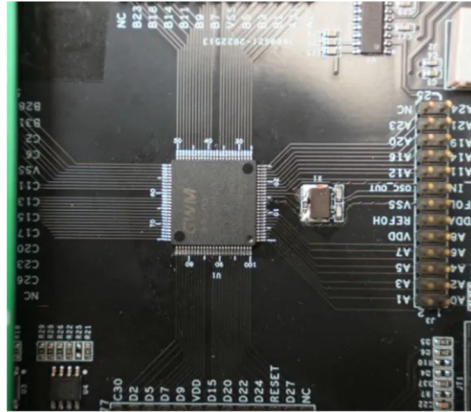
- (1) Technical verification of RISC-V in core automotive control domains is still insufficient, with a lack of long-term reliability data accumulation; more mass production applications are needed to verify its automotive-grade stability.
- (2) Compared to mature architectures such as ARM, the RISC-V software ecosystem is still incomplete. Support for RISC-V in key automotive software such as AUTOSAR, ROS 2, and Android is still in the refinement stage, and further efforts are needed to drive the adaptation of toolchains, middleware, and application software.
- (3) Currently, the number of RISC-V chips and IP cores certified to GB/T 34590 ASIL-D is limited. Automotive applications have extremely high safety requirements, and RISC-V must incorporate more comprehensive safety mechanisms during the hardware design phase, such as hardware encryption, fault detection, and recovery, to ensure reliable system operation.
- (4) Mature RISC-V Tier-1 suppliers and professional engineering teams with RISC-V development experience are still in a growth stage, and supply chain support capabilities need to be strengthened.
- (5) The diversity of RISC-V application scenarios brings the risk of ecosystem fragmentation, and there is an urgent need for unified architecture specifications, safety standards, and testing and certification systems to reduce overall industry costs.

## 4. Application of RISC-V in the Automotive Field

### 4.1. Overview

Currently, several RISC-V automotive chip products have entered the tape-out stage or achieved mass production. In the MCU domain, products such as the Great Wall Zijing M100, Dongfeng DF30, Silergy SA32D, and GoChip CCFC3009PT have entered tape-out or mass production, covering scenarios such as body control, battery management, chassis safety, and zone controllers. Figure 2 shows samples of RISC-V MCU chips. However, for the RISC-V architecture to achieve large-scale commercial deployment in the automotive industry, it must meet a series of hard application requirements for in-vehicle scenarios, including automotive-grade harsh operating environments, standardized functional safety specifications, hard real-time performance constraints, and multi-domain heterogeneous deployment. The following sections analyze RISC-V's technical characteristics, adaptation advantages, and existing

challenges for in-vehicle scenarios from the dimensions of modular instruction set architecture, privilege architecture, low-power technology, interrupt and exception handling, and key virtualization technologies.



### Figure 2. RISC-V Chip Samples

## 4.2. Modular Instruction Set Architecture

Various functional domains of automotive electronic systems have differentiated indicator requirements for processor computing performance, power consumption control, memory resource usage, and real-time response performance. RISC-V adopts a modular architecture design with a base integer instruction set paired with extensible instruction sets, enabling flexible combination and precise tailoring of instruction sets for differentiated in-vehicle business scenarios such as body control, powertrain/chassis, smart cockpit, and autonomous driving. This is the core advantage that allows this architecture to highly adapt to diverse automotive application scenarios. On the premise of ensuring high compatibility of the base software ecosystem, this modular architecture supports chip vendors in developing custom feature enhancements for automotive-grade application scenarios, while effectively avoiding the chip power waste and hardware area overhead caused by instruction redundancy in traditional architectures. The base instruction sets commonly used in the automotive field are mainly RV32I, RV64I, and RV32E, with the corresponding configuration selected depending on the application scenario[9].

Compared with traditional in-vehicle architectures (such as classic automotive-grade architectures like ARMv7-M and ARMv8-M), the RISC-V base instruction set has the following notable differences:

- (1) Highly unified instruction format: Branch, memory access, and arithmetic instructions use a consistent encoding structure, which can effectively reduce hardware design complexity and improve chip timing stability.
- (2) No implicit state or condition flags, avoiding state dependency issues and reducing instruction execution uncertainty.
- (3) Streamlined instructions with low redundancy, achieving better chip power and area control while meeting equivalent functional requirements.
- (4) Core control flow can be achieved without vendor-private instructions, helping to support architecture openness and autonomous controllability.

On the basis of the base instruction set, RISC-V can selectively include various extension instruction sets to targeted enhance core performance such as chip computing power, parallel processing, security encryption, and real-time response. However, excessive extension combinations lead to inconsistent instruction set Profiles across different chips, causing problems such as software fragmentation, redundant toolchain adaptation, and degraded

AUTOSAR compatibility. This is a key issue that automotive RISC-V currently needs to address through standard convergence.

#### 4.3. Privilege Architecture

The RISC-V privilege architecture is the underlying core foundation for implementing hierarchical processor permission control, hardware resource isolation, and secure and stable system operation. It can adapt to two operating modes — lightweight permission control and full-dimension system management — according to different in-vehicle application scenarios, demonstrating strong compatibility and adaptability in complex heterogeneous in-vehicle computing scenarios. Compared with traditional in-vehicle architectures, the RISC-V privilege architecture features simplicity, extensibility, and low overhead. In-vehicle MCU products primarily rely on RISC-V's M mode and U mode for task scheduling, with an overall architecture that is streamlined and efficient, and outstanding real-time performance, fully satisfying the core application requirements of in-vehicle real-time operating system operation and multi-task security isolation. High-performance in-vehicle SoCs can fully support the complete M/S/U/H multi-level privilege operating modes, adapting to complex in-vehicle deployment scenarios with multi-domain fusion and multi-system coexistence, and supporting the diversified operating architectures of high-level in-vehicle systems. This scalable design enables RISC-V to cover the full range of in-vehicle application scenarios from low-power control to high-performance computing. However, in ASIL-D scenarios, unified automotive-grade constraints are still lacking regarding privilege switching latency, permission boundaries, and fault injection protection.

#### 4.4. Low-Power Technology

Chip low-power design directly affects vehicle energy consumption levels, standby duration, and thermal management performance, and has important application value in core in-vehicle scenarios such as ADAS assisted driving, in-vehicle data acquisition sensing, and new energy vehicle energy management. The RISC-V architecture, with its technical characteristics of on-demand instruction set tailoring, reconfigurable microarchitecture design, and flexible core module selection, provides diverse and customizable technical implementation paths for low-power development of automotive-grade chips. Low-power design technologies at the RISC-V ISA level mainly include:

- (1) Modular instruction set tailoring: Based on the RISC-V base instruction set (RV32I), various extension instruction modules can be flexibly selected according to scenario requirements.
- (2) Reconfigurable microarchitecture design: Core microarchitecture parameters such as chip pipeline stages and cache configuration can be customized and optimized according to application scenarios.
- (3) Low-power core selection: Select appropriate RISC-V low-power cores for different automotive application scenarios.

#### 4.5. Interrupts and Exceptions

An interrupt is a response mechanism triggered by external hardware events during processor operation. When the processor receives an external hardware request, it pauses the current task execution, jumps to the corresponding interrupt service routine to complete event handling, and returns to the original task after completion. It is a core mechanism for ensuring the real-time response capability of in-vehicle systems. An exception is a system response behavior triggered by internal errors or special conditions during instruction execution, mainly covering scenarios such as illegal instruction execution, memory access out-of-bounds, division by zero, and system calls. It is actively triggered by instruction execution actions and undertakes the core functions of system fault detection, exception interception, and operational protection. As the underlying core mechanisms for processors to handle external sudden

signals and manage internal operational faults, both directly determine the real-time response level, operational stability, execution determinism, and fault-tolerant self-healing capability of in-vehicle control systems, and are the key underlying foundation supporting the high-reliability, high-real-time operation of in-vehicle RISC-V chips. In automotive electronic control systems, interrupt response latency and jitter characteristics directly affect the execution determinism and operational reliability of core hard real-time tasks such as powertrain control, chassis regulation, and active safety. A comprehensive exception handling system can provide important underlying support for the fault discrimination, error isolation, and system state recovery capabilities required for in-vehicle functional safety, effectively improving the operational reliability and fault tolerance of the vehicle's electronic control system. The RISC-V architecture features a flexibly tailorable and extensible interrupt and exception handling system that can adapt to the full range of in-vehicle application scenarios from entry-level control MCUs to high-end multi-core heterogeneous computing platforms, satisfying both the extreme real-time response requirements of lightweight control systems and the functional safety management and high-reliability operation needs of high-level in-vehicle platforms.

#### 4.6. Virtualization

Hardware virtualization technology enables the abstract encapsulation, secure isolation, and shared reuse of computing resources, and is a core key technology supporting the stable, secure, and efficient operation of in-vehicle multi-domain fusion architectures.

The RISC-V virtualization architecture is centered on the H extension module, which configures an independent Hypervisor privilege level to achieve hardware-level permission isolation between the hypervisor and each guest operating environment, building a standardized virtualization operating system with clear hierarchy and strict boundaries. Based on this virtualization architecture, the hypervisor can leverage native hardware virtualization capabilities to implement core functions such as virtual context management, multi-level address mapping, virtual interrupt routing and distribution, and virtual peripheral mapping, supporting the independent, parallel, and secure scheduling of multiple virtual machines. This architecture achieves abstract isolation and efficient allocation of hardware resources, effectively reducing the development complexity and runtime performance overhead of virtualization software. Leveraging native hardware-assisted virtualization mechanisms, the RISC-V architecture can efficiently adapt to in-vehicle multi-OS mixed deployment architectures, supporting independent deployment, isolated operation, and mutual non-interference of different types of operating systems such as AUTOSAR RTOS, Linux, and QNX within a single chip platform, fully adapting to the system architecture development trend of in-vehicle multitasking and multi-domain fusion.

### 5. Standard Requirements for Automotive RISC-V Chips

In summary, the RISC-V architecture can fully adapt to the core technical requirements of the automotive electronics field. Meanwhile, the industry has established a mature and comprehensive standardization system around multiple dimensions including RISC-V instruction set specifications, hardware configuration standards, non-ISA definitions, and compatibility testing systems, providing the RISC-V architecture with the foundational conditions to support industrial-scale deployment in automotive electronics. However, the RISC-V standard ecosystem in the in-vehicle field is still in a development stage of multi-party collaborative iteration, and the industry has not yet established a unified and comprehensive dedicated in-vehicle standard system. In core areas such as dedicated in-vehicle instruction set specifications and functional safety standard adaptation, there remain significant standard gaps and systemic shortcomings. Among these, incomplete standards have become a key

bottleneck constraining the development of the RISC-V ecosystem, and its fragmentation problem is specifically manifested in:

(1) Fragmented hardware ecosystem: The differentiated custom designs of various vendors have resulted in a large amount of RISC-V software being unable to achieve cross-chip compatible operation. Automakers and hardware vendors are forced to maintain a parallel R&D layout of both ARM and RISC-V architectures, lacking unified standard support for ecosystem adaptation work.

(2) Lagging testing and certification: The functional safety dimension also suffers from inconsistent standards. Existing automotive-grade chips only meet general in-vehicle functional safety level specifications, and a dedicated certification system and evaluation process adapted to RISC-V architecture characteristics have not yet been established.

(3) Instruction set fragmentation: Although the RISC-V base instruction set has been standardized, the specification management and coordinated planning of various extension instruction sets remain chaotic, lacking unified constraint standards. Taking the new server extension instructions added by RISC-V International in 2025 as an example, a unified classification and management mechanism has not yet been established. Against this backdrop, various ISA extension instructions such as AI acceleration and matrix operations are showing explosive growth, significantly increasing the porting and adaptation workload for developers.

In summary, the industry urgently needs to advance the formulation and implementation of unified RISC-V standards for in-vehicle scenarios, providing systematic support for the standardized and robust construction of the in-vehicle RISC-V chip ecosystem.

### 5.1. Basic Requirements

Different chip types have different application scenarios and functional requirements: Automotive MCUs are mainly used in body control, powertrain, chassis control, battery management, and zone controller scenarios, with high requirements for instruction set determinism, code density, and functional safety; SoCs are mainly used in smart cockpit domain controllers and autonomous driving domain controllers, requiring complex operating systems such as Linux and supporting high-performance computing such as multi-virtual machine and AI acceleration; RF chips are mainly used for in-vehicle wireless communications (such as V2X, Bluetooth, Wi-Fi, and UWB keys), with high requirements for low power consumption, stability, and real-time response; ultra-high-speed interface chips are mainly used for in-vehicle Ethernet switching chips, PCIe/SerDes interface chips, and high-speed gateways, with extremely high requirements for data processing performance, real-time performance, and determinism.

In summary, in-vehicle scenarios have clearly demanded standardization of RISC-V chip application configurations: it is necessary to formulate differentiated instruction set configuration specifications (Profiles) combined with different in-vehicle application scenarios and chip categories, clarify the mandatory and optional extension instruction sets, privilege mode configuration guidelines, and memory protection base function system required for various chips, and establish a unified technical specification applicable to the design and selection of in-vehicle RISC-V chips, fundamentally resolving the instruction set fragmentation problem.

### 5.2. General Requirements

At present, the technical implementation schemes, testing standards, and verification methods for RISC-V in-vehicle chips in the areas of functional safety and information security are still incomplete, and a systematic security evaluation system adapted to in-vehicle scenarios has not yet been formed.

In the functional safety dimension, a unified and standardized redundancy technical indicator system needs to be established, clarifying the applicable boundaries and application scenarios of dual-core lockstep, delayed lockstep, coupled and non-coupled redundancy architectures, and unifying the trigger rules and response handling mechanisms of fault comparison logic, to provide standardized support for the compliance verification of core safety indicators such as single-point fault metrics and fault coverage. Regarding the unique custom extension instruction characteristic of the RISC-V architecture, a dedicated evaluation specification needs to be built based on the GB/T 34590 (ISO 26262) in-vehicle functional safety standard. First, all custom instruction extensions must undergo rigorous formal verification processes to prevent the introduction of non-deterministic system behavior and ensure that the execution semantics of the native instruction set remain complete and unaltered. Second, fault injection testing standards adapted to custom extension instructions need to be developed, focusing on verifying their failure characteristics and impact scope in typical hardware fault scenarios such as single-event upsets, and completing quantitative safety assessment. Finally, safety audit specifications for custom extension instructions should be defined, covering core content such as reasonable allocation of instruction encoding space, conflict detection for CSR register usage, and compatibility and interoperability verification with official standard extension instructions, to prevent custom extensions from bypassing hardware safety protection mechanisms and degrading system safety levels. The above specifications can effectively fill the system gap in in-vehicle safety evaluation of RISC-V custom instructions, providing reliable technical basis for standardized industry development and deployment.

In terms of information security, compared to traditional chip architectures, the RISC-V core adds virtualization extensions (H extension), a new privilege level architecture, and control flow integrity protection mechanisms, making its underlying hardware security logic fundamentally different from traditional architectures, and existing information security verification systems are difficult to directly reuse and adapt. With the continuous deepening and popularization of the Software-Defined Vehicle concept, in-vehicle chip architectures are no longer limited to basic computing power supply, and there is an urgent need for native, built-in information security protection capabilities, establishing a comprehensive defense-in-depth security system centered on hardware roots of trust. In this regard, dedicated information security extension specifications need to be developed targeting the new RISC-V architecture characteristics and custom extension instruction system, and combined with existing in-vehicle information security technical standards and evaluation systems, a dedicated information security technical guideline and evaluation scheme for RISC-V in-vehicle chips should be constructed to address the security evaluation gap in this field.

In terms of compatibility, a compatibility testing system addressing the unique fragmentation risk of the RISC-V architecture needs to be established. First, build a RISC-V instruction set compatibility test suite adapted to in-vehicle scenarios. Based on existing RISC-V international standardization achievements, combined with national and industry specifications in the integrated circuit and electronic information fields, targeted test cases for automotive-specific instruction set configuration specifications (Profiles) should be expanded to verify whether the chip's declared instruction set configuration fully implements standard extension instructions and CSR register functions, while also checking for compatibility issues such as opcode conflicts and execution behavior interference between custom extension instructions and standard extension instructions. Second, establish a compatibility testing method oriented toward complete vehicle systems. Select industry mainstream in-vehicle base software to build a standardized benchmark test dataset, verifying the source-code-level portability of RISC-V chips from different vendors under the same in-vehicle instruction set configuration specification (Profile). Ensure that in-vehicle base software can achieve equivalent functionality on different chip platforms without restructuring core algorithm logic, only

through compilation adaptation, effectively improving cross-platform adaptability and generality of software and hardware. Third, establish a coexistence specification for custom extensions and standard extensions. Clarify that custom instructions must not occupy allocated standard extension opcode resources, and require that all custom extension functions must be equipped with compatibility toggle switches to support system degradation and adaptive operation, ensuring overall architecture compatibility and stability.

## 6. Conclusion

This paper discusses the research background, development status, and standard landscape of the RISC-V architecture, examines the background of automotive chip architectures, and reviews the current status of RISC-V chip standards, key technologies, ecosystem, and applications. Combined with current industry development and the application challenges being faced, this paper aims to establish scientific, comprehensive, and systematic standards. As a flexible, extensible, open-source ISA, the RISC-V instruction set architecture has gradually gained commercial application and recognition. The necessity of standardization for RISC-V promotion is mainly reflected in the following aspects: First, unified specifications and standards can ensure compatibility among RISC-V implementations developed by different vendors and developers, reducing the risk of ecosystem fragmentation; second, standardization can improve technical transparency, enhance user and investor confidence, and promote large-scale technology adoption; third, through establishing standards, the maturation of development toolchains, operating systems, and applications can be accelerated, forming a virtuous cycle that continuously drives technological progress and industrial upgrading. Therefore, standardization is a key factor for the success and sustainable development of RISC-V.

## References

- [1] Liu, C., Wu, Y. J., Wu, J. Z., & Zhao, C. (2021). Survey of RISC-V instruction set architecture. *Journal of Software*, 32(12), 3992–4024. (in Chinese)
- [2] Liu, X. L., Lin, H. Y., & Liu, P. (2025). Survey of RISC-V instruction set architecture and its applications. *China Integrated Circuit*, 34(3), 16–20, 49. (in Chinese)
- [3] Yu, Z. H., Fan, J. P., & Bao, Y. G. (2026). Brief analysis of design decisions in the RISC-V instruction set. *Journal of Integration Technology*, 15(4), 306–314. (in Chinese)
- [4] Shao, Y. P., & Huang, L. B. (2025). Research on RISC-V instruction set extensions: Design, implementation, and applications. *Mini-Micro Systems*, 46(10), 2548–2560. (in Chinese)
- [5] Ni, W. J., Huang, R., Yin, H., et al. (2025). Comparative analysis of global RISC-V standard development paths. *Standard Science*, (S1), 206–211. (in Chinese)
- [6] Wang, S. Y. (2026). Breaking through the industrial "chip shortage" dilemma: How domestic RISC-V chips support manufacturing independence? *China Science and Technology Information*, 38(8), 132–134. (in Chinese)
- [7] Chong, D. D. (2021). Current development status and future opportunities of RISC-V-based open-source chip ecosystem. *China Integrated Circuit*, 30(8), 25–30. (in Chinese)
- [8] He, X. Q. (2021). Current status and future of the RISC-V industry. *Microcontrollers & Embedded Systems*, 21(8), 3. (in Chinese)
- [9] Chen, Q. (2025). Accelerating RISC-V automotive application development: How does IAR address challenges and win the future? *Automobile & Parts*, (15), 22–23. (in Chinese)