

Embedded Technology and Intelligent Systems

2025年8月2卷4期



Editorial Board

编委名单

ISSN: 3065-1220

<https://www.hanspub.org/journal/etis>

主编

何立民教授 北京航空航天大学

Editor-in-Chief

Prof. Limin He Beihang University

副主编

何小庆秘书长 嵌入式系统联谊会

Associate Editors

Allan He Secretary General of the Embedded Systems Association

吴薇特聘教授 杭州电子科技大学

Distinguished Prof. Wei Wu Hangzhou Dianzi University

名誉编委

王田苗教授 北京航空航天大学
严义教授 PLCopen China主席/杭州电子科技大学
邵贝贝教授 清华大学工程物理系

Honorary Chief Editor

Prof. Tianmiao Wang Beihang University
Prof. Yi Yan Hangzhou Dianzi University
Prof. Beibei Shao Department of Engineering Physics, Tsinghua University

编委会

马忠梅副教授 北京理工大学计算机学院
王朋朋系统工程 恩智浦（中国）管理有限公司
高级总监
牛建伟教授 北京航空航天大学
陈渝长特聘副教授 清华大学计算机系
张永进总经理 深圳拓普微科技开发公司

沈建华副教授 华东师范大学计算机学院
周立功创始人/ 广州致远电子股份公司
董事长
桑楠教授 电子科技大学信息与软件工程学院

袁涛副教授 清华大学自动化系
常晓明教授 太原理工大学
韩德强高级工程师 北京工业大学计算机学院
魏洪兴教授 北航机械工程及自动化学院
林金龙教授 北京大学软件与微电子学院
刘洪涛研发副总裁 华清远见教育科技集团
/研发中心总经理

Editorial Board

Prof. Zhongmei Ma Beijing Institute of Technology
Lucy Wang Senior Engineering Director of NXP China Management Ltd.

Prof. Jianwei Niu Beihang University
Prof. Yu Chen Tsinghua University
Yongjin Zhang General Manager of Shenzhen Topway Technology Ltd.

Prof. Jianhua Shen East China Normal University
Ligong Zhou Founder of Zhiyuan Electronics Ltd.

Prof. Nan Sang University of Electronic Science and Technology of China

Prof. Tao Yuan Tsinghua University
Prof. Xiaoming Chang Taiyuan University of Technology
Prof. Deqiang Han Beijing University of Technology
Prof. Hongxin Wei Beihang University
Prof. Jinlong Lin Peking University
Hongtao Liu Vice President of R&D of HQYJ Education Technology Group

嵌入式系统发展趋势报告 2025~2026

目 录

卷首语——嵌入式系统发展趋势报告 2025~2026 总述	
中国计算机学会嵌入式系统专业委员会.....	197
嵌入式系统体系架构研究现状与发展趋势	
Current Research Status and Development Trends of Embedded System Architecture	
赵庆玲, 周江南.....	199
嵌入式处理器技术的发展现状与趋势	
Development Status and Trends of Embedded Processor Technology	
吴武飞.....	207
嵌入式操作系统技术发展趋势	
Development Trends in Embedded Operating System Technology	
张润, 夏晨, 陈咸彰.....	213
嵌入式软件语言及开发新范式	
New Paradigms for Embedded Software Languages and Development	
陈铭松, 焦跃军, 杨永湜.....	223
嵌入式网络技术现状与发展趋势	
The Current Status and Development Trends of Embedded Network Technology	
冯智伟, 邓庆绪.....	232
嵌入式系统中间件和软总线综述	
An Overview of Middleware and Soft Bus in Embedded Systems	
杨鹏飞, 郑天洋, 郭恒, 王泉.....	240
异构智能嵌入式系统 AI 模型推理与部署优化——从模型轻量化到系统级加速的综述与展望	
AI Model Inference and Deployment Optimization for Heterogeneous Intelligent Embedded Systems—A Survey and Perspective from Model Lightweighting to System-Level Acceleration	
庞为光.....	255
云边端协同计算与智能	
Cloud-Edge-End Collaborative Computing and Intelligence	
周俊龙, 侯祥鹏, 兰兰, 徐妍.....	261
嵌入式系统安全与可信技术发展趋势	
Development Trend of Security and Trusted Technologies for Embedded Systems	
张凯龙, 马萌旭, 何周灿, 樊弈辰, 周世钰, 闫琳珊.....	268

嵌入式系统产业现状与发展趋势

Current Situation and Development Trends of the Embedded Systems Industry

何小庆	276
-----------	-----

卷后语——CCF 嵌入式系统专委发展规划

郭兵	283
----------	-----

期刊信息

期刊中文名称:《嵌入式技术与智能系统》

期刊英文名称: **Embedded Technology and Intelligent Systems**

期刊缩写: **ETIS**

出刊周期: 双月刊

语 种: 中文

出版机构: 汉斯出版社(Hans Publishers, <https://www.hanspub.org/>)

编辑单位:《嵌入式技术与智能系统》编辑部

主 编: 何立民, 北京航空航天大学教授

网 址: <https://www.hanspub.org/journal/etis>

订阅信息

订阅邮箱: sub@hanspub.org

订阅价格: 180 美元每年

广告服务

联系邮箱: adv@hanspub.org

版权所有: 汉斯出版社(Hans Publishers)

Copyright©2025 Hans Publishers, Inc.

版权声明

文章版权和重复使用权说明

本期刊版权由汉斯出版社所有。

本刊文章已获得知识共享署名国际组织(Creative Commons Attribution International License)的认证许可。

<https://creativecommons.org/licenses/by/4.0/>

单篇文章版权说明

文章版权由文章作者与汉斯出版社所有。

单篇文章重复使用权说明

注: 著作权者准许任选 CC BY 或 CC BY-NC 作为文章的重复使用权, 请慎重考虑。

权责声明

期刊所刊载的评论、意见、观点等均出自文章作者个人立场, 不代表本出版社的观点或看法。对于文章任何部分及文内引用材料给任何个人、机构、及其财产所带来的任何损失及伤害, 本出版社均不承担任何责任。我们郑重声明, 本出版社的出版业务, 不构成对任何产品商业性能的保证, 也不表示本社业已承认本社出版物中所述内容适用于某特定用途。如有疑问, 请寻找专业人士协助。

卷首语——嵌入式系统发展趋势报告 2025~2026总述

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

嵌入式(计算机)系统是计算机技术演化中形成的一个重要分支,是赋予“万物”与环境进行感知、交互、互联与协同能力的技术基石。其技术体系高度综合,不仅具有与通用计算技术同样完整的技术体系和范式,还呈现出内涵持续演化、软硬件形态多元、技术路径丰富、应用领域广泛等诸多特征。从知识与技术体系看,嵌入式计算与系统覆盖了计算机科学与工程、软件工程、自动化与电气工程等多个信息学科,同时又与航空航天、工业制造、交通运输、电力电网、智能家居、医疗电子等广泛应用领域紧密交叉融合。由此,嵌入式系统已经成为融通信息与物理世界,构建新型具身智能机器人等新型智能体与虚实融合空间的重要载体和底座,是新数智化时代全球科技创新、产业变革发展的强大驱动力。

近年来,嵌入式计算与系统的技术体系及其软/硬件随着新一代 ICT 技术的迭代高速演化,同时与人工智能/大模型、云计算、大数据、物联网、移动计算等新兴计算技术的深度融合,呈现出诸多新的发展内涵与趋势,如:

- (1) 异构化, (多) CPU + DSP/ISP/NPU 等异构加速协处理器的混合计算架构具有突出的计算效能优势,已成为当前机器人、消费电子、航空航天、智能驾驶等诸多领域的主流计算体系;
- (2) 网联化,越来越多的嵌入式设备的多态泛在网络与中间件等互联互通,彼此实现数据、资源的流动、交互与赋能,互联网 + 物联网的融合形态愈发丰富、体系日益庞大;
- (3) 智能化,硬件架构演进、算力资源提升、云边端协同以及轻量化技术大幅促进深度神经网络、大模型等代表性人工智能技术与嵌入式设备的结合与融合;
- (4) 协同化,个体与群体融合的云边端协同计算、群体智能协同任务等新计算模式已在行业中广泛应用,诸如协作式智能交通系统中的车-路-云协同、无人机航行器集群中的行为与任务协同等;
- (5) 开放化,微服务、网络接入、软件定义、群体协同等呈现高度动态开放特征的理念与技术广泛融入在新一代嵌入式系统的体系中,使得系统架构的弹性、韧性更强,当然,这种开放也带来了被入侵、攻击的风险及安全可信的挑战。

除此之外,嵌入式系统软硬件的发展还具有复杂化、集成化、模块化等相关特征,这里不再阐述。鉴于此,经中国计算机学会(CCF)嵌入式系统专委会于 2025 年初将本趋势报告的研究与撰写确立为专委会年度的重点工作任务,旨在系统梳理国内外前沿进展,研判未来的关键趋势,为学界、业界相关专业人员与机构提供信息与决策参考。撰写组由专委会主任郭兵教授担任组长,副主任张凯龙教授、陈铭松教授担任副组长,以及东北大学邓庆绪教授、冯智伟副教授,南昌大学吴武飞副教授、重庆大学陈咸彰教授,

西安电子科技大学杨鹏飞教授，南京理工大学周俊龙教授、赵庆玲副教授，齐鲁工业大学庞为光博士，麦克泰软件公司创始人何小庆教授等 13 名委员组成撰写组。期间，以调研选题、分组撰写、研讨修订为主线，在西安、上海等地多次专题讨论、反复打磨报告内容，最终形成本报告。

本报告以“主题化”组织内容，聚焦嵌入式系统的新发展、新趋势，覆盖 10 个重点方面，包括：1) 嵌入式系统体系架构；2) 嵌入式处理器技术；3) 嵌入式操作系统；4) 嵌入式软件语言及开发新范式；5) 嵌入式网络；6) 嵌入式中间件与软总线；7) 嵌入式智能与系统；8) 云-边-端协同计算与智能；9) 嵌入式系统安全与可信；10) 国内外产业发展综评等。每个主题均从“国外现状分析-国内现状分析-发展趋势研判”三部分展开，系统梳理理论、技术、产业层面的代表性进展与典型实践，并给出趋势要点、研判和建议，便于学界、业界专业人员与机构的快速检索与对照参考。

最后，本报告的形成离不开各位编写成员的专业付出，也得益于相关高校、研究机构与诸多领域专家的宝贵建议与大力支持，在此一并致谢！

报告疏漏与不尽之处难免，敬请批评指正！

中国计算机学会嵌入式系统专业委员会

嵌入式系统体系架构研究现状与发展趋势

赵庆玲, 周江南

南京理工大学计算机科学与工程学院, 江苏 南京

收稿日期: 2025年10月28日; 录用日期: 2025年11月20日; 发布日期: 2025年12月1日

摘要

嵌入式系统作为信息技术的神经末梢, 经历了从1971年微处理器出现到半个世纪的演进, 形成了硬件中心、RTOS支撑、SoC集成、AI边缘等多阶段发展。文章系统阐述了嵌入式系统体系架构的研究现状与发展趋势, 回顾了国外从Apollo制导计算机到现代AI异构平台的演进历程, 梳理了国内自80年代引进模仿到近年来鸿蒙微内核、RISC-V和边缘AI的自主创新路径, 对比了中外在技术起步、生态成熟度和创新模式上的差异, 最后展望了异构计算、微内核模块化、边缘智能和分布式协同等未来发展方向, 指出嵌入式架构将向高性能、安全、可扩展和智能协同演进。

关键词

嵌入式系统, 体系架构, 实时操作系统, SoC, 异构计算

Current Research Status and Development Trends of Embedded System Architecture

Qingling Zhao, Jiangnan Zhou

School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing Jiangsu

Received: October 28, 2025; accepted: November 20, 2025; published: December 1, 2025

Abstract

Embedded systems serve as the neural endpoints of information technology. Since the introduction of microprocessors in 1971, they have evolved over half a century through stages including hardware-centric design, real-time operating system (RTOS) support, System-on-Chip (SoC) integration, and edge AI applications. This paper systematizes the research status and development trends of embedded system architectures. It reviews the international evolution from the Apollo Guidance Computer to contemporary AI-driven heterogeneous platforms, and outlines China's developmental trajectory—from technology introduction in the 1980s to recent indigenous innovations such as

the HarmonyOS microkernel, RISC-V architecture, and edge intelligence. A comparative analysis is conducted between China and other countries regarding technological origins, ecosystem maturity, and innovation models. Finally, the paper prospects future trends—including heterogeneous computing, modular microkernel design, edge intelligence, and distributed coordination—highlighting the ongoing evolution of embedded architectures toward higher performance, security, scalability, and intelligent collaboration.

Keywords

Embedded Systems, System Architecture, Real-Time Operating System, SoC, Heterogeneous Computing

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

嵌入式系统作为现代信息技术的“神经末梢”，是嵌入到物理设备中的专用计算系统，通过软硬件协同实现对设备功能、性能及交互的智能控制。从 1971 年 11 月第一款微处理器的推出[1]，经过半个世纪的发展，嵌入式系统体系架构已经囊括虚拟化、微服务、异构并行计算和边缘计算等多个领域。

2. 国外发展现状

图 1 为美国 NASA 为阿波罗登月任务设计的 Apollo Guidance Computer，这是最早的嵌入式设备之一，具有定制微控制器和只读程序存储器，用于实现特定的飞行控制功能[2]。此阶段嵌入式架构以硬件为中心，系统功能固定，控制方式以轮询和简单中断为主，尚未形成操作系统支持的体系结构。



Figure 1. Apollo guidance computer

图 1. 阿波罗制导计算机

进入 70 年代末至 80 年代，随着微处理器的快速发展，嵌入式系统开始普及。Intel 推出的 8080、8051 以及 Motorola 6800 (如图 2)等处理器成为工业自动化、医疗电子等领域嵌入式系统的主力平台[3]。系统软件方面，实时操作系统(RTOS)的出现使得嵌入式系统首次具备了多任务调度、中断优先级、时间管理

等功能[4], 体系结构开始向分层设计演进。

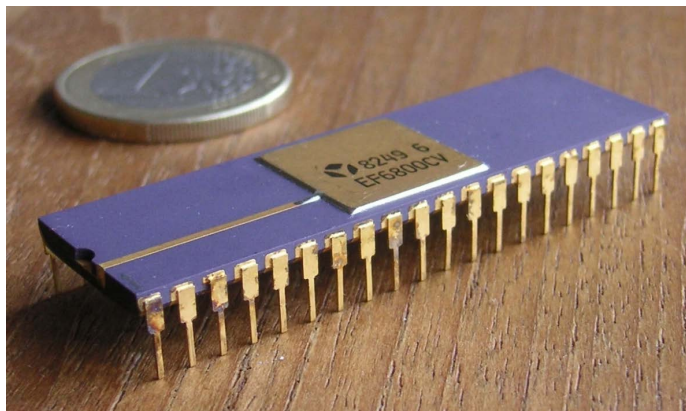


Figure 2. Motorola 6800 chip
图 2. Motorola 6800 芯片

1990 年代是嵌入式架构发展的关键时期。System-on-Chip (SoC)设计理念兴起, 处理器厂商开始将 CPU 核、DMA、内存控制器等功能集成于单芯片, 大幅提升了性能与集成度[2]。同时, ARM 架构以其低功耗、高性能的优势迅速崛起, 成为嵌入式处理器的主流。

21 世纪初, 嵌入式系统迈向网络化与智能化阶段, 体系架构呈现出以硬件抽象层、RTOS 内核、中间件、应用层组成的分层结构, 并广泛应用轻量级协议(如 MQTT)与远程交互机制, 适应物联网的需求[4][5]。此阶段的嵌入式系统已不再局限于本地控制, 而是成为边缘智能节点, 参与更大范围的协同计算体系, 基于物联网与边缘计算的设计被广泛提出, 如图 3 就是一款基于边缘计算的工业机器人系统设计架构。

近年来, 人工智能和边缘计算进一步推动嵌入式体系结构升级。以 NVIDIA Jetson、Google Edge TPU 等平台为代表的新一代嵌入式系统, 采用异构多核架构(CPU + GPU + NPU), 具备强大的 AI 推理能力[5]。操作系统则向微内核、分布式和虚拟化方向发展(如 L4ReC)(见图 4), 这是一款基于微内核设计的操作系统, 旨在为嵌入式系统和虚拟化提供支持。操作系统强调实时性、安全性与可扩展性[6], 系统结构设计更注重模块解耦、资源隔离与 AI 能力的深度融合, 使嵌入式系统在自动驾驶、工业视觉、智能终端等领域发挥关键作用。

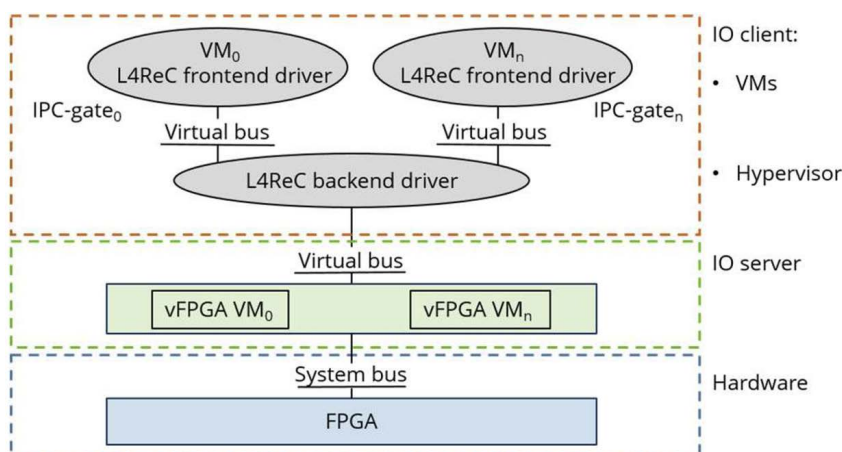


Figure 3. L4ReC Embedded reconfigurable system
图 3. L4ReC 嵌入式可重构系统

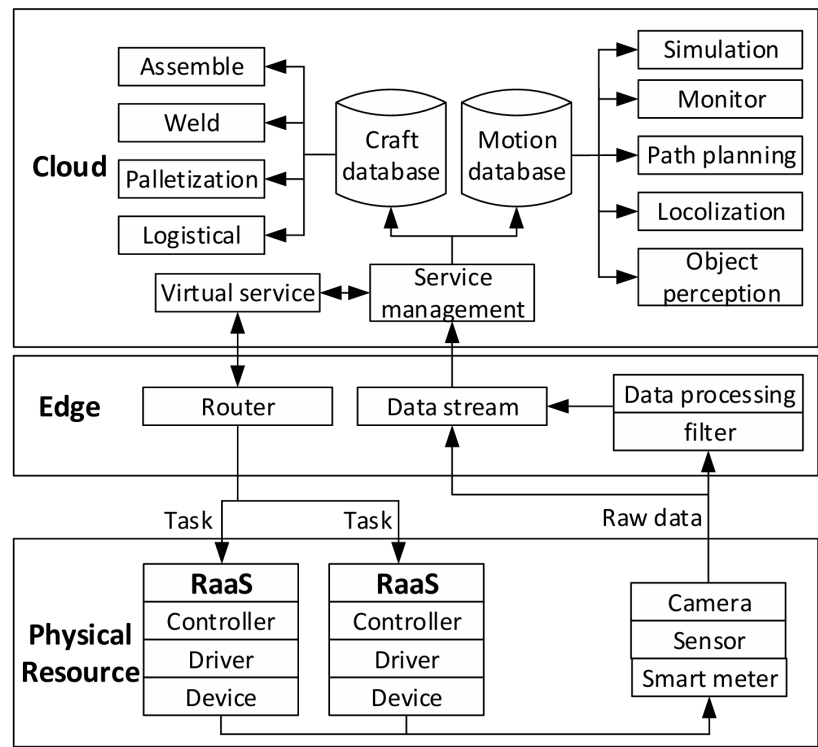


Figure 4. Edge-computing-based industrial robot system architecture diagram
图 4. 基于边缘计算的工业机器人系统架构设计图

总体来看, 国外嵌入式系统体系架构的发展历程体现了从专用向通用、从简单向智能、从单核向异构的演进趋势。随着 AI、IoT 与通信技术融合不断深入, 嵌入式架构将持续在性能、安全、可扩展性与智能协同等方面推进创新, 继续支撑下一代智能系统的发展。

3. 国内发展现状

中国嵌入式系统体系架构的发展历程始于 20 世纪 80 年代, 早期嵌入式系统主要依赖引进国外平台和器件[7], 这一阶段的嵌入式架构尚未形成完整体系, 主要为高校和研究所基于单片机进行的开发实验与科研探索。

进入 90 年代中后期, 随着电子信息技术在国内的快速发展, 嵌入式系统逐步进入产业应用阶段。高校和企业开始引入国外嵌入式处理器和开发平台, 如 ARM7、ARM9 等, 同时接触并应用一些主流实时操作系统(RTOS), 包括 uC/OS-II、VxWorks [8]等。

进入 2000 年代, 嵌入式系统在国内的应用领域迅速扩大, 涵盖工业控制、家电控制、通信设备、消费电子等。体系架构进一步发展, 开始采用基于 SoC 的设计模式, 集成度和复杂性显著提升。国产处理器如龙芯、飞腾、君正等逐步推向市场[9], RT-Thread、MiniGUI、嵌入式 Linux 等国产操作系统及图形界面库也在不断完善[4]。

2010 年以后, 随着物联网和智能硬件兴起, 嵌入式系统体系结构进一步向网络化、平台化和智能化演化。国产芯片厂商如兆易创新、全志科技、瑞芯微等崛起, 推出面向物联网和边缘计算的 MCU、MPU 平台[10], 嵌入式处理器架构向 ARM Cortex-M (如图 5)和 Cortex-A 系列靠拢, 支持多核并发、外设接口丰富, 具备较高实时性与低功耗特性。操作系统方面, RT-Thread 发展出 Nano、Smart 等不同形态, 华为 LiteOS、阿里 AliOS Things 也加入国产 RTOS 阵营, 支持 BLE、Wi-Fi、NB-IoT 等通信协议栈, 形成以

RTOS 为核心、多层中间件协同的完整体系结构[11]。

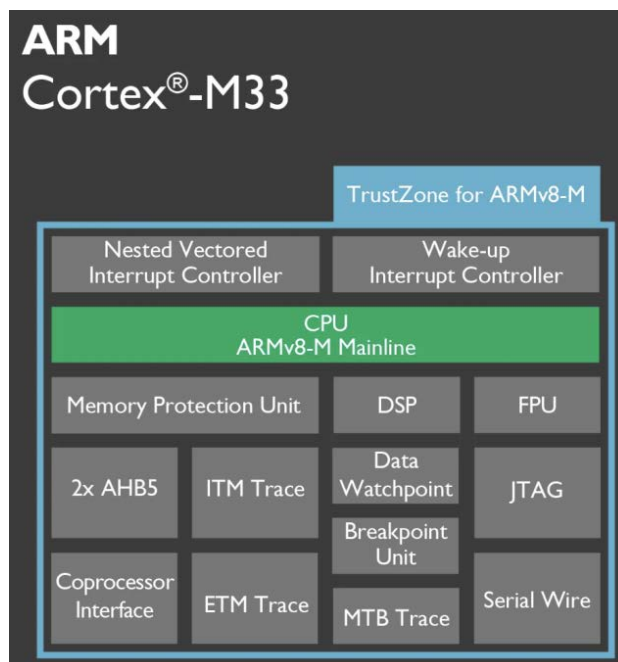


Figure 5. ARM Cortex-M architecture

图 5. ARM Cortex-M 架构

近年来,随着鸿蒙操作系统的发布以及 RISC-V 开源指令集在国内的推广,中国嵌入式体系架构正进入融合创新的新阶段[12]。鸿蒙 OS 以微内核与分布式架构为核心理念,针对多设备协同运行、端云协同构建和高可靠实时控制场景进行了架构设计,推动国产嵌入式系统向泛终端、泛网络协同方向演进。同时,赛昉、芯来、全志等公司纷纷推出 RISC-V 架构芯片,结合轻量化操作系统与推理框架(如 Tengine [13], 见图 6),在智能终端、边缘 AI 等场景中形成新的体系结构路径。翼辉信息自主研发的 ECS 嵌入式安全容器通过 μs 级实时响应、MB 级轻量化资源隔离和全架构兼容特性,为航空航天、智能制造等关键领域提供自主可控的容器化基础设施,填补了传统容器在实时嵌入式系统的技术空白,推动任务关键系统向云原生架构安全演进。翼辉 VSOA 微服务框架凭借全双工通信、内置服务发现和 SIL4 安全认证等核心能力,为嵌入式实时系统提供高可靠、低延迟的分布式服务架构,全面满足汽车电子和工业控制等关键领域对实时性、安全性和灵活性的严苛要求,加速嵌入式系统向云原生架构的智能化升级。总体来看,中国嵌入式系统体系架构的发展实现了从引进模仿到自主构建的转变,未来将在异构计算、低功耗 AI、安全可信计算等领域持续创新,构建以国产软硬件为核心的新型嵌入式技术生态。

在嵌入式系统体系结构中,混合关键性系统代表了一种从传统的“一功能一硬件”的联邦式架构向基于资源共享的集成式架构演进的核心范式。其根本目标是在单个高性能硬件平台(如多核处理器或片上系统 SoC)上,整合运行具有不同安全关键性等级(如安全关键、任务关键、非关键)的软件组件,从而显著降低系统的尺寸、重量、功耗和成本。这种架构的核心设计原则是保证时空隔离性。与传统架构依赖物理隔离来实现安全不同,混合关键性系统通过硬件与系统软件层面的协同设计,在共享资源上构建逻辑隔离。在时间维度,采用时间分区调度或增强型实时调度算法,确保高关键性任务的计算时间预算不被低关键性任务抢占;在空间维度,利用内存管理单元(MMU)或内存保护单元(MPU)为每个分区或任务提供独立受保护的地址空间,防止错误的内存访问造成跨任务数据污染。因此,在嵌入式系统架构中,

混合关键性系统是实现“软件定义”功能集中化(如汽车域控制器、航空综合模块化航电)的核心基石,它通过在集成化与安全性之间取得精密平衡,推动了嵌入式系统向更高效、更灵活、更经济的方向发展。国内在工业(如 OpenEuler Embedded [14])和学术界[15]-[18]都有大量的关于混合关键性系统的工作。

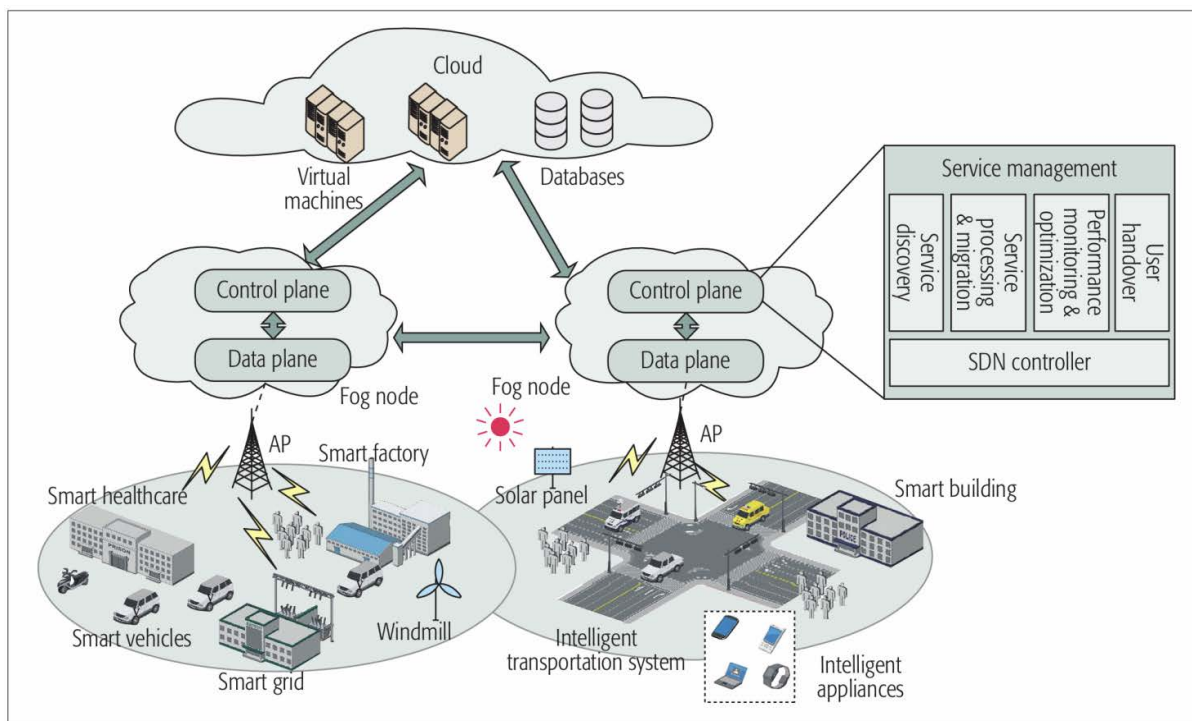


Figure 6. ECIoT architecture

图 6. ECIoT 架构

4. 国内外发展对比

嵌入式系统作为信息技术与工业应用融合的关键技术,其体系架构的发展在国内外呈现出不同的路径与节奏。国外嵌入式体系架构的发展起步早,以美国 NASA 的 Apollo 导航计算机为代表,随着微处理器的出现,嵌入式系统逐渐形成以微处理器+实时操作系统为核心的架构形态。进入 90 年代,国外体系结构迅速迈入 SoC 集成阶段,ARM、MIPS 等处理器架构与嵌入式 Linux、VxWorks、QNX 等操作系统紧密结合,形成软硬件高度耦合、支持多任务和网络通信的成熟平台,广泛应用于通信、汽车、消费电子等领域。

相比之下,中国的嵌入式系统体系架构起步于 20 世纪 80 年代,初期以引进国外芯片与开发板为主,系统功能主要集中在工业控制、仪器仪表等场景。90 年代后,随着高校教学与科研的推广,ARM 架构逐渐在国内普及,嵌入式软件架构逐步从裸机控制向分层设计演进。进入新世纪后,中国在政策支持和产业驱动下,大力推动自主嵌入式体系的发展,体系结构日趋规范化与平台化。

当前,国外嵌入式体系架构已广泛实现多核异构、AI 协同、虚拟化等先进特性,嵌入式设备具备强大边缘计算能力,并通过微内核架构与分布式机制实现设备间协同控制。而国内在保持稳定控制能力的基础上,正加速向智能化、网络化方向迈进,鸿蒙 OS 的提出标志着国内嵌入式体系在分布式架构方面的重大突破,同时 RISC-V 指令集的兴起也为中国构建开放、自主的体系结构提供了新路径。总体而言,国外嵌入式体系以长期积累和技术成熟为优势,形成了完整生态和标准体系,而中国则通过政策引导和技

术追赶, 逐步建立起以自主可控为目标的本土嵌入式架构体系。

5. 发展趋势与展望

随着新一代信息技术的迅速发展, 嵌入式系统体系架构正面临前所未有的机遇与挑战, 从传统的单芯片控制逐步走向异构融合、智能协同和边缘自治, 呈现出多个显著的发展趋势。

首先, 嵌入式体系架构正朝着异构计算方向快速发展。传统嵌入式系统以单一 CPU 或 MCU 为核心, 难以满足当前对计算密度、低功耗和多任务并行处理的需求。为了应对人工智能、图像识别、实时决策等高复杂度应用, 嵌入式平台逐步集成多种异构计算单元, 如 GPU、DSP、FPGA、NPU 等, 形成 CPU+X 的多核协同架构[2]。

其次, 操作系统和软件体系正向微内核、模块化、组件化方向演进。嵌入式操作系统正在逐步摒弃传统的单体内核设计, 采用更具安全性和可扩展性的微内核架构, 能够支持动态加载、服务隔离、权限分离等机制, 提高系统的可管理性和容错能力。同时, 中间件技术和轻量级虚拟化工具也被广泛引入, 以便于跨平台部署和设备间协同。操作系统还不断向下适配资源受限设备, 向上对接物联网平台和云服务, 成为连接设备、网络和服务的核心软件支撑。

随着人工智能技术的普及, 嵌入式系统正逐步具备边缘智能的能力。过去需要通过云端完成的图像识别、语音处理、目标检测等任务, 现今越来越多地在终端设备本地完成, 降低了延迟, 减少了带宽占用, 同时提升了系统的隐私保护能力。这一趋势推动了边缘 AI 芯片、TinyML 算法和小模型加速库的快速发展, 也对架构设计的灵活性与可裁剪性提出了更高要求[19]。

未来, 嵌入式系统体系架构还将继续向分布式与协同计算演化。在车联网、工业物联网、智慧城市等复杂应用场景中, 多个嵌入式节点需要以高可靠、低延迟的方式协同工作。这要求体系架构不仅关注单一设备的功能实现, 还需考虑网络通信协议、分布式资源调度、实时同步与容灾机制等系统性问题。

综上所述, 嵌入式系统体系架构的发展正从功能单一、结构固定的传统模式, 迈向智能化、异构化、安全化与开放化的新阶段。未来的嵌入式系统将不仅仅是“设备的大脑”, 更将成为边缘智能协同网络中的关键节点, 其体系架构设计也将持续融合多领域技术, 成为推动智能社会发展的核心支柱。

参考文献

- [1] Lenhardt, D., O'Malley, M., Payne, C., et al. (2004) A Look at Intel Processors from the 4004 to the Pentium Pro. <https://docsbay.net/doc/562505/a-look-at-intel-processors-from-the-4004-to-the-pentium-pro>
- [2] 何小庆, 何灵渊. 嵌入式软件开发的三个趋势[J]. 嵌入式技术与智能系统, 2024, 1(1): 1-10.
- [3] Wray, W.C. and Greenfield, J.D. (1994) Using Microprocessors and Microcomputers the Motorola Family. Prentice-Hall Inc.
- [4] 何小庆. 国产嵌入式操作系统发展思考[J]. 单片机与嵌入式系统应用, 2019, 19(12): 4-5+10.
- [5] Capogrosso, L., Cunico, F., Cheng, D.S., Fummi, F. and Cristani, M. (2024) A Machine Learning-Oriented Survey on Tiny Machine Learning. *IEEE Access*, 12, 23406-23426. <https://doi.org/10.1109/access.2024.3365349>
- [6] Wu, M., Duan, Z. and Li, J. (2024) Performance Optimization of Microkernel-Based Embedded I/O Virtualization Techniques. *International Conference on Artificial Intelligence, Automation and High Performance Computing*, Zhuhai, 19-21 July 2024, 134-141. <https://doi.org/10.1145/3690931.3690955>
- [7] 王福刚, 杨文君, 葛良全. 嵌入式系统的发展与展望[J]. 计算机测量与控制, 2014, 22(12): 3843-3847.
- [8] Behnam, M., Nolte, T., Shin, I., et al. (2008) Towards Hierarchical Scheduling in VxWorks. *Proceedings of the 4th International Workshop on Operating Systems Platforms for Embedded Real-Time Applications*, Prague, 1 July 2008, 63-72.
- [9] 平安证券. 国产 CPU 引领者, 构建自主可控生态体系[EB/OL]. <https://www.vzoo.com/document/20221213c77b3a451721616c054a00d7.html>, 2022-12-12.
- [10] 华福证券. 全志科技首次覆盖: AIoT 创新迭出, 智能硬件 SoC 厚积薄发[EB/OL].

- https://m.hibor.com.cn/wap_detail.aspx?id=a0681ac3f7ef31601edbb5bcd3a27b08, 2024-08-07.
- [11] 何小庆. 3 种物联网操作系统分析与比较[J]. 微纳电子与智能制造杂志, 2020, 2(1): 65-72.
 - [12] 刘畅, 武延军, 吴敬征, 等. RISC-V 指令集架构研究综述[J]. 软件学报, 2021, 32(12): 3992-4024.
 - [13] Sakamura, K. and Koshizuka, N. (2003) T-Engine: The Open, Real-Time Embedded-Systems Platform. *IEEE Micro*, **22**, 48-57. <https://doi.org/10.1109/mm.2002.1134343>
 - [14] openEuler 在 RISC-V 生态论坛发布 AI 边缘计算解决方案[J]. 自动化博览, 2024, 41(2): 5.
 - [15] Zhao, Q.L., Gu, Z.H., Zeng, H.B. and Zheng, N.G. (2018) Schedulability Analysis and Stack Size Minimization with Preemption Thresholds and Mixed-Criticality Scheduling. *Journal of Systems Architecture*, **83**, 57-74. <https://doi.org/10.1016/j.sysarc.2017.03.007>
 - [16] Zhao, Q.L., Gu, Z.H. and Zeng, H.B. (2017) Design Optimization for AUTOSAR Models with Preemption Thresholds and Mixed-Criticality Scheduling. *Journal of Systems Architecture*, **72**, 61-68. <https://doi.org/10.1016/j.sysarc.2016.08.003>
 - [17] Zhao, Q.L., Al-Bayati, Z., Gu, Z.H. and Zeng, H.B. (2016) Optimized Implementation of Multirate Mixed-Criticality Synchronous Reactive Models. *ACM Transactions on Design Automation of Electronic Systems*, **22**, 1-25. <https://doi.org/10.1145/2968445>
 - [18] Zhao, Q.L., Gu, Z.H. and Zeng, H.B. (2015) Resource Synchronization and Preemption Thresholds within Mixed-Criticality Scheduling. *ACM Transactions on Embedded Computing Systems*, **14**, 1-25. <https://doi.org/10.1145/2783440>
 - [19] Dutta, D.L. and Bharali, S. (2021) Tinyml Meets IoT: A Comprehensive Survey. *Internet of Things*, **16**, Article 100461. <https://doi.org/10.1016/j.iot.2021.100461>

嵌入式处理器技术的发展现状与趋势

吴武飞^{1,2}

¹南昌大学人工智能学院, 江西 南昌

²南昌大学先进信号处理与智能通信江西省重点实验室, 江西 南昌

收稿日期: 2025年10月28日; 录用日期: 2025年11月20日; 发布日期: 2025年12月1日

摘要

嵌入式系统作为专用计算机的重要分支, 具有广泛而多样的应用场景, 其核心嵌入式处理器长期面临体积、可靠性、功耗与成本等方面的严苛要求。随着物联网(IoT)与人工智能(AI)的深度融合, 嵌入式设备数量呈爆发式增长, 通用计算与嵌入式计算的界限日益模糊, 使嵌入式处理器技术的重要性进一步凸显。文章系统回顾了嵌入式处理器技术的历史演进、关键进展与设计目标, 分析了国内外研究现状及差异, 并从五个方面展望其未来发展趋势。文章的研究为嵌入式处理器技术的系统认知与创新发展提供了结构化的分析框架与参考依据。

关键词

嵌入式系统, 物联网, 嵌入式处理器技术, 芯粒(Chiplet), 边缘AI

Development Status and Trends of Embedded Processor Technology

Wufei Wu^{1,2}

¹School of Artificial Intelligence, Nanchang University, Nanchang Jiangxi

²Jiangxi Provincial Key Laboratory of Advanced Signal Processing and Intelligent Communications, Nanchang University, Nanchang Jiangxi

Received: October 28, 2025; accepted: November 20, 2025; published: December 1, 2025

Abstract

Embedded systems, as a key branch of special-purpose computers, have a wide and diverse range of applications. Their core—the embedded processor—has long faced stringent requirements in terms of size, reliability, power consumption, and cost. With the deep integration of the Internet of Things (IoT) and artificial intelligence (AI), the number of embedded devices has exploded, blurring

the lines between general-purpose and embedded computing, further highlighting the importance of embedded processor technology. This article systematically reviews the historical evolution, key advances, and design goals of embedded processor technology, analyzes the current status and differences in domestic and international research, and provides an outlook on its future development trends from five perspectives. This research provides a structured analytical framework and reference basis for the systematic understanding and innovative development of embedded processor technology.

Keywords

Embedded Systems, Internet of Things, Embedded Processor Technology, Chiplets, Edge AI

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

1961 年应用于阿波罗登月计划的麻省理工 Apollo 制导计算机(Apollo Guidance Computer, AGC)被认为是最早的嵌入式计算机。大众汽车 Beetle 在 1968 年使用首个电子控制单元(Electronic Control Unit, ECU)调节燃油喷射。嵌入式处理器的发展历程始于 20 世纪 70 年代微处理器的问世。1971 年, 英特尔推出了首款商用微处理器 4004, 标志着嵌入式系统的开端[1]。随后, 8 位和 16 位微处理器相继面世, 被广泛应用于仪器仪表、医疗设备等领域。80 年代, 随着微电子工艺的提升, 微控制器(Microcontroller Unit, MCU)将处理器、存储器和 I/O 接口集成于单一芯片, 推动了嵌入式系统的小型化和高效化。90 年代, 数字信号处理器(Digital Signal Processor, DSP)和片上系统(System on Chip, SoC)技术迅速发展, ARM 等嵌入式 CPU 崭露头角。进入 21 世纪, ARM 架构处理器凭借低功耗、高性能的优势, 成为嵌入式领域的主流选择[2]。2015 年后, 第五代精简指令集计算机(Reduced Instruction Set Computer, RISC-V) [3]等开源指令集架构兴起, 为嵌入式处理器创新注入新动力。嵌入式处理器的发展经历了从简单微处理器到高度集成的 SoC, 目前正朝着多样化、开源化的演进过程。现场可编程阵列(Field Programmable Gate Array, FPGA)凭借其高度灵活的硬件可重构性, 成为嵌入式系统中加速计算和定制处理任务的理想选择[4]。随着异构计算、RISC-V、芯粒(Chiplet) [5]等技术的发展, FPGA 将在嵌入式系统中扮演更加关键的角色, 为未来智能计算提供高效解决方案。Chiplet 作为一种构建 SoC 的全新方法, 可提高良率并将成本降低 45%以上 [6], 如图 1 所示。Chiplet 将芯片划分为多个离散元件, 并通过标准化接口连接它们, 使设计人员能够应对 5/6G、AI 和 VR 时代的性能、效率、功耗、尺寸和成本挑战。

在市场占有率方面, 全球出货量最大的嵌入式处理器 MCU 被普遍认为是 STMicroelectronics 公司的 STM32 系列[7]。目前, 中国 MCU 市场呈现出多元化、高性能、国产化的发展趋势。2023 年中国嵌入式系统市场规模约为 180 亿美元, 占全球市场的 16.4%。预计到 2028 年, 中国市场的规模将达到 300 亿美元, 年复合增长率约为 10.5%, 高于全球平均水平[8]。

我国在嵌入式处理器领域面临“双循环”战略下的独特需求: 一方面, 工业自动化、新能源车等新兴领域亟需高可靠、低功耗的定制化处理器(如 RISC-V 在工控芯片渗透率达 38%), 但 ARM 架构授权受限倒逼自主可控; 另一方面, AIoT 设备年出货量超 30 亿台催生边缘算力需求, 而通用处理器在能效比(国产嵌入式芯片达 5TOPS/W)和实时性上难以匹配。中国计算机互联技术联盟(CCITA)表示, 我国可使用成熟的 28 纳米制程生产芯片, 并通过小芯片封装成高性能芯片, 借助 UCIe 互联技术, 可实现与 16 甚至

7 纳米芯片相当的性能[9]。当前国际竞争聚焦 Chiplet 异构集成与存算一体架构,我国亟需精准跟踪技术路线,关注其关键科学挑战和工艺问题,避免错失万亿级智能终端市场主导权。掌握嵌入式处理器发展动态,是突破高性能芯片“工艺墙”制约、构建自主产业生态的关键突围路径。因此,接下来我们将围绕嵌入式处理器国内外研究现状展开详细讨论。

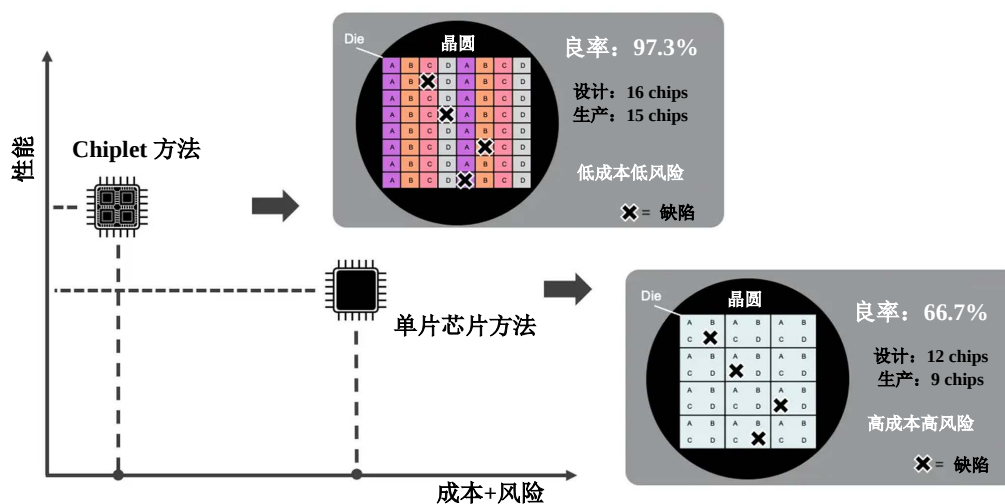


Figure 1. Performance and cost comparison of Chiplets and monolithic chip approaches [6]

图 1. Chiplet 与单片芯片方法的性能和成本比较[6]

2. 国内外研究现状

2.1. 国内研究现状

近年来,国内嵌入式处理器领域呈现出异构集成、高性能与国产化相结合的发展趋势。复旦微推出的 FMQL20SM ARM + FPGA SoC 平台[9],基于四核 ARM Cortex-A7 处理器与 85K 可编程逻辑资源,实现 PS 端与 PL 端协同工作,满足机载、工业等多场景应用,并由创龙科技基于该平台设计,实现国产化率 100%。同时,芯动科技于 2022 年发布的 Innolink™ Chiplet 技术[10],通过跨工艺、跨封装的异构集成,兼容 UCIe 标准,填补了国内在这一领域的空白。另一方面,云天励飞推出的 IPU-X6000 加速卡,基于国产 14nm Chiplet 工艺打造,具备 256T 算力和高速 C2C Mesh 互联技术,支持大模型推理,进一步推动了 AI 应用在嵌入式领域的发展[11]。

国产 RISC-V 架构的兴起为嵌入式 FPGA 提供更多自主可控的应用场景,如 AIoT、自动驾驶等。国产 RISC-V IP 核(芯来科技 N300 系列)适配国产 FPGA (安路科技 PH1A 系列),形成全自主技术链[12]。关键领域应用方面,航天控制系统采用 RISC-V + FPGA 方案,抗辐照指标达 300krad(Si)[13],华为、海思、长江存储等公司为代表,也相继推出了包括 FPGA 与 ARM 架构融合的异构计算平台,满足了对高性能计算密集型应用的需求。

2.2. 国外研究现状

ARM 公司 M 系列在全球嵌入式处理器市场中保持领先地位,并持续推动智能硬件的创新与普及,当前最新的 M 系列采用的是 Armv8-M,为了给嵌入式设备提供所有神经网络处理所需的动力,2023 年,Arm 针对物联网应用领域发布了 Cortex-A320 [14],这是 ARM 推出的首款针对边缘工作负载的 64 位 Armv9 CPU 内核。

近年来, Chiplet 技术[5]被广泛应用于高性能计算、服务器、人工智能加速器等领域。同时, Intel 也在其 Foveros 等先进封装技术中采用了 Chiplet 理念, 推动了 3D 封装和高带宽互连的发展[15]。ARM 在 2025 年推出了首个公开规范的 Chiplet 系统架构(CSA), 通过模块化设计和标准化接口, 推动了 Chiplet 技术的普及和生态建设[16]。低功耗一直是嵌入式领域永恒的主题之一, 高通在 2024 年推出 QCC730 Wi-Fi 解决方案[17], 与此同时, 功耗相比前代降低 88%, 适用于电池供电的工业、商业和消费级应用。值得注意的是, UCIe 2.0 已于 2024 年 8 月发布, 相比 1.0 版本, 2.0 标准在带宽密度和功耗表现上更优, 同时增强对 3D 封装、系统可管理性和兼容性的支持[18]。在 EDA 工具方面, Keysight 推出“Chiplet PHY Designer 2025”, 以支持 UCIe 2.0 和 OCP BoW 标准, 可用于 AI 和数据中心级高速 Chiplet 设计[19]。

2.3. 国内外研究对比

当前, 国内外在嵌入式处理器技术方面均正处于高速发展状态且各具特色, 其中国外在嵌入式技术的生态系统建设方面较为成熟, 如 NVIDIA 的 CUDA 和 TensorRT 工具链, 为开发者提供了强大的支持。国内在 RISC-V 生态系统的建设上取得了显著进展, 如 Chisel 3.6 开源硬件设计语言在自动化 Chiplet 设计中的应用。值得注意的是, 由于国外嵌入式技术起步早, 积累厚, 因此其在高端市场占据主导地位。例如, 美国、欧洲和日本等地区的企业在汽车电子、航空航天、工业自动化等领域拥有巨大的市场份额并正引领 Chiplet 技术的发展及生态构建; 值得注意的是, 近年来国内相关研究和从业者紧密拥抱开源、RISC-V 等新型技术生态, 加快了嵌入式处理器技术的发展和應用。

3. 嵌入式处理器技术发展趋势及研判

嵌入式处理器技术正呈现新的发展趋势, 正朝着低功耗、高性能、异构多核、开源等方向发展。随着 AIoT、边缘计算、5G、自动驾驶等应用的兴起, 未来嵌入式处理器将进一步强化 AI 加速、低功耗、安全内建、MCU + 无线一体化等方向发展。主要体现在以下五个方面:

(1) 边缘 AI 计算能力增强: 推动嵌入式处理器集成 AI 加速单元, 英伟达 Jetson、谷歌 Edge TPU, 以及 ARM Ethos 等都面向 AI 推理优化, 支持边缘设备的智能计算需求。NXP 正在将嵌入式处理器演进为高性能异构平台, 如为软件定义汽车而生 S32 系列统一架构覆盖整车各类控制域(动力、ADAS、车身、网关等), 近年来 NXP 越来越多地将 AI/ML 功能下放到嵌入式 SoC 上, 尤其是边缘设备, 例如面向边缘视觉应用推出的 i.MX 8M Plus 内置神经网络加速器。

趋势: 在不依赖云的情况下实现本地智能(安全、实时、低延迟、长生命周期学习), 让 AI 技术无处不在。

(2) 安全性内建成为标配: 随着 SoC 复杂度提升(集成 CPU、GPU、NPU、ISP、无线、加密单元等), 导致 SoC 攻击面增大, 因此所有新一代 SoC 均强在芯片设计的最初阶段就系统性地融入安全机制即“Security by Design, 简称 SbD”。尤其在汽车领域, 硬件级安全机制成为刚需, 例如 NXP 提供 EdgeLock 安全架构, 涵盖安全启动、密钥存储、加密加速、物理防护等, 以支持 PSA Certified、Arm TrustZone、SE050 安全芯片等标准; 于此同时, ARM TrustZone、RISC-V PMP (物理内存保护)、Intel SGX 等技术也加强了数据加密、安全引导等能力, 以满足车规级、工业控制等应用的安全需求。Chiplet 的出现标志着半导体技术在性能、可扩展性和灵活性方面的重大转变。然而, Chiplet 也带来了独特的安全挑战, 尤其是在功能安全关键的嵌入式领域的应用, 其安全问题是处理器完整性和可靠性的基础。

趋势: 安全性从可选项变成默认配置, 成为设计起点和基础规范。

(3) 面向低功耗物联网优化: 在物联网(IoT)方面, RM Cortex-M、Cortex-A 系列不断迭代, 采用更先进的制程工艺(如台积电 5 nm、三星 3 nm), 集成低功耗特性, 如 big.LITTLE 架构、高能效计算等, 以满

足物联网(IoT)和边缘计算对低功耗的需求。NXP 近年来推出轻量、低功耗的嵌入式处理器 LPC 系列,其特点主要有超低功耗、快速启动,适合电池供电场景。

趋势: 将 MCU + 无线一体化(如支持蓝牙、Zigbee、Thread、Matter 等协议),降低 IoT 产品 BOM 成本和开发复杂度。

(4) RISC-V 生态正快速崛起: 作为开源指令集架构(ISA),RISC-V 自 2015 年起快速发展,全球众多企业(如 SiFive、华为、阿里平头哥等)纷纷推出基于 RISC-V 的嵌入式芯片。其开放性降低了芯片设计门槛,推动产业生态加速成熟。作为 ARM 授权费用增长的替代选项,RISC-V 也成为了 NXP、ST、TI 及 Microchip 等主流嵌入式处理器厂商的重要备选方案。随着 RISC-V 架构的发展,国产 MCU 正加速替代进口产品,逐步提升市场份额。香山作为全球性能领先的一款开源 RISC-V 处理器核,支持 RV64GCBVH,目前受到国内外的广泛关注。

趋势: 开源凝聚共识,契合市场规律,加速 RISC-V 技术推广。

(5) Chiplet 技术发展及应用: Chiplet 技术为快速实现定制的高性能嵌入式处理器硬件提供了新的技术路线。与此同时,也存在一些新的亟待解决的问题。例如,面向新一代嵌入式与实时系统在片上多芯粒(Chiplets)集成中的关键互联需求。(1) 需要具有高带宽、低延迟、低功耗、可扩展且具备高可靠性的互联网络协议及控制 IP (网络片上互连(Network-on-Chip, NoC) IP); (2) 针对异构工艺和多厂商芯粒协同带来的兼容性挑战,需要统一的互联协议框架与行业标准。

趋势: Chiplet 技术背景下,高效可靠的网络片上互连(Network-on-Chip, NoC) IP 方案及其行业标准的制定需进一步突破。

4. 结语

综上所述,嵌入式处理器技术在物联网与人工智能深度融合的背景下,展现出持续发展的重要性与广阔前景。本文系统回顾了嵌入式处理器的历史演进、关键技术进展及设计目标,并对国内外研究现状进行了对比分析,进一步梳理了该领域的发展脉络与差异。在此基础上,从边缘 AI 计算能力增强、安全性内建、低功耗物联网优化、RISC-V 生态发展以及 Chiplet 技术应用五个方面对未来发展趋势进行了研判。嵌入式处理器正朝着低功耗、高性能、异构多核及开源化方向发展,同时强化 AI 加速、硬件安全、无线集成和模块化设计,以满足智能终端、边缘计算、车规级应用等多样化需求。未来,随着开源架构的推广、异构多核与 Chiplet 技术的广泛应用,嵌入式处理器的设计将更加灵活、高效与安全,推动整个产业生态的优化与创新。总体而言,本文的综述不仅为理解嵌入式处理器技术的演进规律提供了系统性认知,也为相关研究与工程实践提供了理论参考与战略指导。

参考文献

- [1] Faggin, F., Shima, M., Mazor, S., et al. (1996) The History of the 4004 Microprocessor: The First CPU on a Chip. *IEEE Micro*, **16**, 10-20.
- [2] ARM Ltd (2015) The Evolution of the ARM Architecture and Its Role in Low-Power Embedded Systems. UK: ARM Holdings.
- [3] Waterman, A., Lee, Y., Patterson, D., et al. (2014) The RISC-V Instruction Set Manual, Volume I: User-Level ISA, Version 2.0. EECS Department, University of California.
- [4] Doe, J. (2021) FPGA-Enhanced Embedded Systems for Real-Time Processing. *IEEE Transactions on Industrial Informatics*, **17**, 2003-2012.
- [5] Kothari, G.N. (2024) Chiplet Based High-Performance Processing Systems. State University of New York at Binghamton.
- [6] Keysight Technologies (2024) What is a Chiplet, and Why Should You Care? <https://www.keysight.com/blogs/en/tech/sim-des/what-is-a-chiplet-and-why-should-you-care>

-
- [7] STMicroelectronics (2025) STM32 32-Bit Arm® Cortex®-M Microcontrollers. <https://www.st.com/en/microcontrollers-microprocessors/stm32-32-bit-arm-cortex-mcus.html>
 - [8] IDC (2023) Worldwide Embedded Systems Market Forecast 2023-2028, IDC.
 - [9] 复旦微电子. FMQL20S400M ARM + FPGA SoC 国产平台简介[EB/OL]. <https://www.fmsh.com/68a87800-b264-b6d3-30a9-d2e5fdb6e1b4/>, 2025-04-10.
 - [10] 芯动科技. Innolink™ Chiplet 技术介绍[EB/OL]. <https://www.innosilicon.com.cn/ip-solution/ucie>, 2025-04-10.
 - [11] 中国日报. 国产推理算力革命: 云天励飞 IPU-X6000 加速卡[EB/OL]. <https://tech.chinadaily.com.cn/a/202407/24/WS66a0ac2aa3107cd55d26c654.html>, 2024-07-24.
 - [12] 芯来科技. 芯来助力中移芯昇发布两款 RISC-V 内核通信芯片[EB/OL]. 2023-08-15. <https://www.nucleisys.com/newsdetail.php?id=258>, 2024-07-20.
 - [13] 张伟, 李航. 基于 RISC-V 与国产 FPGA 的星载控制系统设计[J]. 宇航学报, 2023, 44(7): 1023-1031.
 - [14] Arm (2025) Arm® Cortex®-A320: Introducing Cortex-A320: Ultra-efficient Armv9 CPU Optimized for IoT. <https://newsroom.arm.com/blog/introducing-arm-cortex-a320-cpu>
 - [15] Intel Corporation (2024) Intel Foveros 3D Packaging Technology White Paper. <https://cdrdv2-public.intel.com/826148/Volume%206%20Intel%20Advanced%20Packaging%20Technology.pdf>
 - [16] Chen, S., Zhang, H., Ling, Z., *et al.* (2024) The Survey of Chiplet-Based Integrated Architecture: An EDA Perspective. <https://arxiv.org/abs/2411.04410>
 - [17] Qualcomm Technologies (2024) QCC730 Wi-Fi Solution Product Brief. <https://www.qualcomm.com/wi-fi/products/qcc730>
 - [18] UCIE Consortium (2024) UCIE Consortium Releases 2.0 Specification Supporting Manageability System Architecture and 3D Packaging. UCIE Consortium. https://files.futurememorystorage.com/proceedings/2024/20240808_SPOS-301-1_UCIE_Consortium_Das_Sharma.pdf
 - [19] Keysight Technologies (2024) Keysight Expands Chiplet Interconnect Standards Support in Chiplet PHY Designer 2025. <https://www.keysight.com/gb/en/about/newsroom/news-releases/2025/0121-pr25-026-keysight-expands-chiplet-interconnect-standards-support-in-chiplet-phy-designer-2025.html>

嵌入式操作系统技术发展趋势

张 润, 夏 晨, 陈咸彰

重庆大学计算机学院, 重庆

收稿日期: 2025年10月28日; 录用日期: 2025年11月20日; 发布日期: 2025年12月1日

摘 要

随着信息技术的迅速发展, 嵌入式系统已由传统的单一功能控制器演进为支撑物联网、工业互联网、智能汽车和消费电子等领域的核心计算平台。嵌入式操作系统(Embedded Operating System, EOS)作为嵌入式设备的基础软件, 在资源调度、安全隔离和智能协同方面发挥着关键作用, 其性能、可靠性与安全性直接影响系统的实时性、能效与可信度。文章系统梳理了嵌入式操作系统的研究现状与发展趋势。从安全架构、漏洞防御、模糊测试、形式化验证等方面总结了国内外在嵌入式系统安全技术方面的主要进展; 从调度优化、能效管理、内存与缓存协同设计等角度分析了性能优化的关键技术; 在文件系统层面, 归纳了面向资源受限设备的高效文件管理与闪存优化方案; 最后, 从智能化、泛在化及基于Rust语言的安全可扩展方向, 探讨了嵌入式操作系统未来的发展趋势。研究表明, 嵌入式操作系统正由传统的“轻量与实时”逐步迈向“安全、智能与泛在”的新阶段, 成为万物互联时代的关键基础软件。

关键词

嵌入式操作系统, 物联网, 工业自动化

Development Trends in Embedded Operating System Technology

Run Zhang, Chen Xia, Xianzhang Chen

College of Computer Science, Chongqing University, Chongqing

Received: October 28, 2025; accepted: November 20, 2025; published: December 1, 2025

Abstract

With the rapid advancement of information technology, embedded systems have evolved from traditional single-function controllers into the core computing platforms that support the Internet of Things (IoT), industrial Internet, smart vehicles, and consumer electronics. As the fundamental software of embedded devices, the Embedded Operating System (EOS) plays a crucial role in resource

scheduling, security isolation, and intelligent collaboration. Its performance, reliability, and security directly determine the system's real-time responsiveness, energy efficiency, and trustworthiness. This paper systematically reviews the current research status and future trends of embedded operating systems. It summarizes recent progress in embedded system security technologies—including system architecture design, vulnerability defense, fuzz testing, and formal verification. From the perspective of performance, it analyzes key techniques such as scheduling optimization, energy-efficient management, and cache-algorithm co-design. In the domain of file systems, it explores efficient file management and flash memory optimization strategies for resource-constrained devices. Finally, the paper discusses future development directions toward intelligent, ubiquitous, and Rust-based secure embedded operating systems. The study indicates that embedded operating systems are transitioning from the traditional focus on “lightweight and real-time” features to a new era characterized by “security, intelligence, and ubiquity,” becoming a cornerstone of the Internet of Everything.

Keywords

Embedded Operating System, Internet of Things, Industrial Automation

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

随着信息技术的不断进步，嵌入式系统已逐渐从早期的单一功能控制器，发展成为支撑物联网、工业互联网、智能汽车与消费电子的核心计算平台。在这一转型过程中，嵌入式操作系统(Embedded Operating System, EOS)作为嵌入式设备的基础软件，扮演着资源调度、安全隔离和智能协同的关键角色。其性能与可靠性不仅直接决定设备的实时性和能效表现，更在安全可信与智能化应用中发挥着基础性作用。

近年来，嵌入式操作系统发展出了安全关键嵌入式操作系统、物联网嵌入式操作系统、智能嵌入式操作系统和泛在嵌入式操作系统等类型。图 1 展示了嵌入式操作系统的四大分类及其代表性系统的发展时间线，其中，安全关键/分区嵌入式操作系统(如 INTEGRITY、VxWorks 653)专注于高可靠性和任务隔离，适用于航空航天和工业控制等领域。物联网嵌入式操作系统(如 FreeRTOS、Zephyr)以轻量化和多协议支持为核心，广泛应用于智能家居和工业物联网。智能嵌入式操作系统(如 LiteOS、Google Coral)集成了 AI 本地化推理能力，支持设备端智能决策。泛在嵌入式操作系统(如 HarmonyOS、XIUOS)则强调跨设备协同和生态统一性，推动全场景智能互联。随着技术的融合与发展，嵌入式操作系统正逐步向智能化、泛在化演进，成为万物互联时代的核心基础软件。本文将介绍支撑四类嵌入式操作系统发展的关键技术进展。

2. 安全技术

嵌入式操作系统广泛应用于航空航天、工业控制、智能交通等关键领域，其安全性直接关系到系统的稳定运行和数据安全。随着物联网和人工智能技术的发展，嵌入式系统面临的安全威胁日益复杂多样，如漏洞利用、数据泄露、恶意攻击等。因此，保障嵌入式操作系统的信息安全成为当前研究的热点和难点。本节从系统安全架构与机制、安全管理机制、漏洞模糊测试、安全检测与验证等方面，对近年来国

内外关于嵌入式操作系统信息安全的研究进行了分类总结，如图 2 所示。



Figure 1. Timeline of embedded operating system development
图 1. 嵌入式操作系统发展时间线

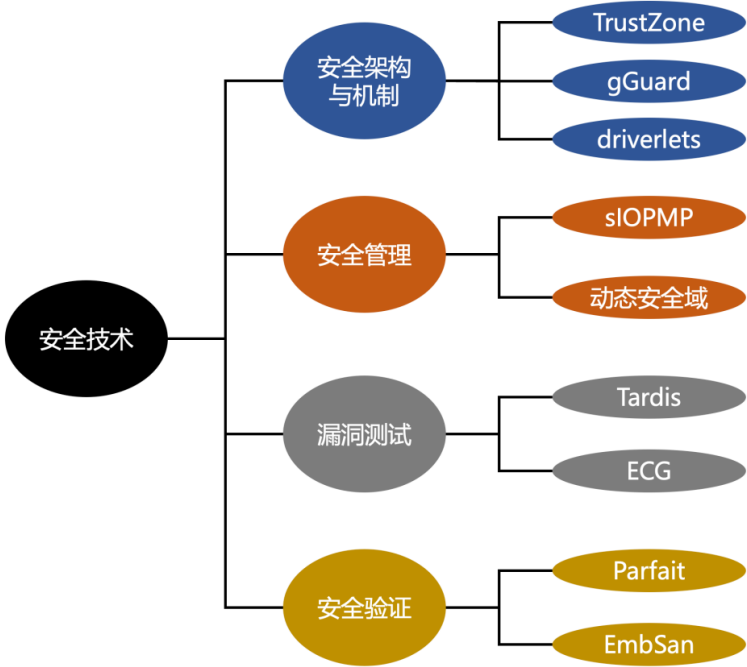


Figure 2. Classification of security technologies for embedded operating systems
图 2. 嵌入式操作系统安全技术分类

(1) 安全架构与机制

嵌入式操作系统的安全机制与架构设计是防止漏洞被利用的关键。一个健壮的安全架构可以有效隔离潜在的 attack 面，限制恶意行为的扩散。当前，ARM Cortex-M TrustZone 等技术被广泛应用于嵌入式系

统中, 以实现安全区域的划分和保护。然而, 这些技术在实际应用中仍存在一些安全隐患和性能瓶颈。例如, 快速状态切换机制可能导致安全状态和非安全状态之间的权限控制出现漏洞, 从而被攻击者利用。因此, 研究更精细、高效的安全机制与架构, 对于提升嵌入式系统的整体安全性具有重要意义。

现有实时系统越来越多地使用现成的组件, 并且经常连接到互联网, 从而暴露于各种攻击之下。Hasan 等人[1]提出了“攻击者负担”(attacker's burden)这一度量指标, 用于评估实时系统中基于调度器的安全机制的有效性。该指标基于“工作量因子”(work factor)的概念, 通过计算攻击者在不同安全机制下可利用的时间来衡量安全性的强弱。具体来说, 攻击者负担越高, 意味着攻击者可利用的时间越少, 系统的安全性越高。Yadlapall 等人[2]提出的 gGuard 通过在编译器、库和操作系统层面实现高效的数据粉碎技术, 防止信息泄露。具体来说, gGuard 在编译时分析应用程序的数据访问模式和依赖关系, 将原始程序转换为“自粉碎”版本, 在执行过程中自动清理不再使用的内存对象。对于无法在执行期间清理的内存对象, gGuard 采用异步清理方法, 减少清理操作对性能的影响。ARM Cortex-M TrustZone 技术是 ARM 公司推出的硬件安全扩展, 旨在为嵌入式系统提供基于硬件的安全隔离机制。其核心思想是通过将处理器的执行环境划分为安全世界(Secure World)和非安全世界(Non-Secure World), 实现对密钥、用户数据、安全固件等敏感资源的隔离保护。Ma 等人[3]针对 ARM Cortex-M TrustZone 架构中快速状态切换机制的安全隐患, 提出了一种名为 return-to-non-secure 的攻击技术, 并设计了两种地址清理机制进行防御, 实现了在不影响系统性能的前提下, 有效防止特权升级和任意代码执行的效果。Park 等人[4]针对在移动设备的 TrustZone 可信执行环境中运行敏感 GPU 计算时, 如何安全且高效地部署 GPU 软件栈的问题, 提出了 GR-T 架构, 通过移动设备与无 GPU 的云服务协作进行 GPU 记录与回放, 实现了安全、高效的 GPU 计算, 同时显著降低了客户端的时间和能量消耗。Kang 等人[5]针对嵌入式系统中广泛使用的内存不安全 C/C++代码导致的堆内存安全问题, 提出了基于 ARMv8-M 架构的微胖指针方案, 实现了一种高效的边界检查机制, 确保嵌入式系统的堆内存安全, 同时性能开销较低。Guo 等人[6]针对 ARM TrustZone 缺乏现代 I/O 设备驱动程序的问题, 提出了一种称为“driverlets”的新型方法。通过记录和回放全功能驱动程序与设备的交互过程, 生成最小可行的驱动程序, 解决了 TrustZone 中安全 I/O 的关键挑战。

(2) 安全管理机制

高沙沙等人[7]针对现有基于 MILS 架构的嵌入式操作系统在任务故障后无法实现功能重构和实时动态加载的问题, 提出了面向任务的多级安全域动态管理架构。通过安全域管理、故障监控和功能重构模块, 实现任务在特定安全域内的动态迁移与功能恢复。确保了任务在故障后的持续运行, 提升了系统的可靠性和安全性。Feng 等人[8]针对现有 I/O 隔离机制在 TEE 系统中存在性能和可扩展性问题, 提出了 sIOPMP 机制。通过多级树形检查器、可挂载 IOPMP 和 IOPMP 重映射等设计, 实现了高效且可扩展的 I/O 保护。多级树形检查器支持超过 1000 个硬件区域, 可挂载 IOPMP 支持无限数量的设备, 而 IOPMP 重映射机制则允许设备在热和冷状态之间动态切换, 以适应不同的 I/O 工作负载。Guo 等人[9]针对大规模关键任务 IoT 系统的可扩展性、安全性和实时性挑战, 提出跨层解决方案, 涵盖数据驱动建模、动态资源管理、形式化漏洞检测和自适应传感器攻击防御。通过层次化控制架构和分布式资源管理, 提升了系统的可扩展性、安全性及对物理攻击的实时响应能力。

(3) 漏洞模糊测试

嵌入式操作系统的漏洞可能导致系统崩溃、数据泄露甚至恶意攻击, 严重威胁关键基础设施的安全。传统的测试方法难以全面、高效地发现这些漏洞, 因此, 漏洞检测与模糊测试技术成为保障嵌入式系统安全的重要手段[10]。早期的黑盒测试方法虽然简单易行, 但效率低下, 难以覆盖复杂的代码路径。随着技术发展, 覆盖引导模糊测试等方法逐渐兴起, 但仍存在对特定架构依赖强、生成测试用例质量不高等问题。Shen 等人[11]提出的 Tardis 是一种覆盖引导的嵌入式操作系统模糊测试工具。它通过在编译时对

目标操作系统进行插桩,实现与操作系统无关的代码覆盖收集和分析。具体来说,Tardis 使用基于位图的存储方式记录每个执行的代码分支,并通过共享内存缓冲区将覆盖信息传递给主机端的模糊测试引擎。主机端的分析模块利用简单的位运算快速判断测试用例是否触发了新的代码路径,从而指导后续的测试用例生成和变异。Zhang 等人[12]提出的 ECG 利用大型语言模型(LLM)来增强嵌入式操作系统的模糊测试能力。ECG 首先通过静态分析提取目标系统调用的参数类型和约束信息,并结合文档生成实际的 C 代码示例。然后,利用 strace 捕获执行跟踪信息,将其转换为系统调用规范。在模糊测试过程中,ECG 使用 LLM 生成测试用例,并根据执行反馈动态调整生成策略,以提高测试用例的质量和覆盖范围。

(4) 安全检测与验证

如何有效检测并修复嵌入式系统中的潜在漏洞,以及如何从形式上验证其安全性,仍然是嵌入式系统领域的热点问题。Liu 等人[13]针对嵌入式操作系统固件缺乏有效漏洞检测手段的问题,提出 EmbSan 框架,利用动态插桩和解耦的主机运行时库实现对多种嵌入式操作系统固件的漏洞检测,成功检测出实际漏洞,且性能开销与现有内核检测工具相当,提升了检测效率与适应性。Athalye 等人[14]针对硬件安全模块(HSM)从应用规范到电路实现的安全性和无泄漏验证问题,提出 Parfait 框架,通过信息保持精化(IPR)的形式化方法,利用中间抽象层次和传递性信息保持精化实现模块化验证,成功验证了四个 HSM 实例的安全性,确保其无正确性、安全性和漏洞泄漏,且具有良好的可扩展性和适应性。Pasquier 等人[15]针对嵌入式操作系统安全分析工具选择困难的问题,提出通过构建设备驱动案例研究并应用多种分析技术(如模型检查、代码分析等)来评估工具适用性,实现了对嵌入式操作系统模型和代码分析工具的部分互补性分析,为工具选择提供依据。

3. 调度优化

嵌入式系统由于资源受限、能耗敏感与实时性要求高,其调度机制需在性能、功耗与任务时限之间实现最优平衡。针对这一需求,研究者提出了多种资源与任务协同优化策略,如图 3 所示。

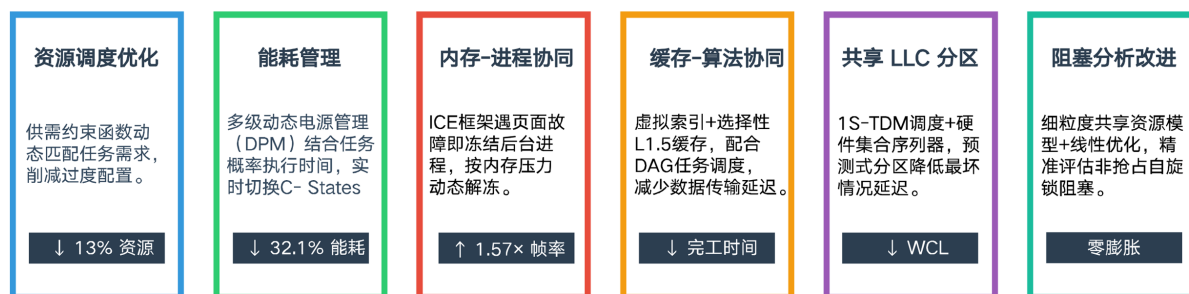


Figure 3. Scheduling optimization in embedded operating systems

图 3. 嵌入式操作系统中的调度优化

Mamata 等人[16]提出了一种面向资源优化的调度模型,以最小化实时嵌入式系统中的资源过度配置。该方法利用供需约束函数优化资源分配,在保证任务时限的同时,平均减少了 13%的资源使用量。Reghenzani 等人[17]提出了一种多级动态电源管理(DPM)策略,以优化异构处理器上实时有向无环图任务的能耗。该方法结合任务概率执行时间分析,动态调整处理器 C-States,在保证硬实时约束的前提下,实现了最高 32.1%的能耗降低,并通过仿真和实体验证了优化效果。Li 等人[18]提出了一种协作式内存和进程管理框架 ICE。ICE 通过在内存页面故障时冻结后台进程,并基于内存压力动态解冻,从而有效减少后台应用引发的内存故障,提升了 1.57 倍的帧率和减少了交互警告。Jiang 等人[19]提出了一种面向并行实时系统的缓存与算法协同设计方案。该研究引入了一种虚拟索引、物理标记、选择性包含的 L1.5 缓存,

并提出了针对有向无环图任务的调度方法,以减少数据传输延迟并降低任务完工时间。Wu 等人[20]提出了一种预测性共享最后级缓存(LLC)分区的方法,方法采用了 1S-TDM 调度策略并结合硬件扩展集合序列表,显著降低了最坏情况延迟(WCL),提高了共享缓存分区的利用效率。Chen 等人[21]提出了一种基于细粒度共享资源模型的改进阻塞分析方法,解决并行实时任务中非抢占式自旋锁的阻塞问题。方法通过线性优化技术实现了无膨胀分析,避免了传统分析中的过度表示。

4. 文件系统

嵌入式设备具有存储资源受限、计算能力有限、成本敏感等特点,导致嵌入式系统在文件系统设计方面面临重大挑战。现有研究从空间管理粒度与元数据精简、数据结构与页分配优化、压缩/缓存以抑制读写放大、以及高效垃圾回收策略等方向展开,如图 4 所示。



Figure 4. File system optimization techniques in embedded operating systems
图 4. 嵌入式操作系统中的文件系统优化技术

为此, Zhang 等人[22]提出了一种新型嵌入式文件系统 LOFFS,以解决嵌入式文件系统由于内存消耗过多和启动功能差而无法有效管理大容量 NAND 闪存的问题。LOFFS 重新定义了空间管理粒度并简化元数据以加快挂载程序,为不同类型的文件提出了混合文件结构,以解决性能和内存占用之间的权衡,从而以最小的资源占用实现高性能。Jiang 等人[23]为减少数据传输延迟及内存占用,采用 B+树来管理连续物理空间,令每个叶节点指向一个段列表,引用一组物理上连续的页面。确保在低内存占用的情况下实现确定性的文件数据定位。为了提升写操作性能,MIFS 提出了包括顺序按需分配以及单页分配的双模式页分配策略,将根据不同的情况采用不同的模式以减少执行时间。Gao 等人[24]针对嵌入式系统的有限存储空间和有限运行时内存,分析了现有压缩只读文件系统 I/O 放大和冗余计算的问题,认为传统压缩只读文件系统的问题在于采用了固定大小的输入压缩策略。因此,EROFS 引入了固定大小的输出压缩策略,并设计了高效记忆的解压方案,以缓解读放大率问题并减少不必要的计算。

闪存采用页和块的粒度对物理空间进行管理,大于现有嵌入式文件系统的逻辑管理单位。因此,将嵌入式文件系统移植到闪存介质之上将导致逻辑 I/O 粒度和物理 I/O 粒度不匹配的问题,手动扩大文件系统的逻辑 I/O 粒度将带来大量的移植工作,也将产生写放大问题,导致额外的空间消耗和性能降低。Zhang 等人[25]提出了存储中间件 NV-middle,保留了嵌入式文件系统原有的操作粒度,定义了空间状态图来标识属于同一物理块的逻辑块的状态,并根据逻辑/物理页面与块之间的关系以及数据访问的局部性,优化了内存使用策略和页面缓存机制。此外, Zhang 等人[26]提出了一种新型嵌入式文件系统 ELOFS,重新定义了空间管理粒度并简化了元数据以提升性能。同时根据目录和数据文件不同的访问模式设计了混合文件结构,以日志方式组织频繁更新的目录文件,将数据页分散在闪存上,并通过 D-index

进行索引。Sun 等人[27]发现闪存垃圾回收的开销占总 I/O 时间的 85%以上,降低了嵌入式文件系统的性能,并缩短了闪存芯片的耐用性,而删除不同热度的数据块的影响也有所差异。针对上述问题,该文献提出了低复杂度、高效率的 GC 方案 LightGC,基于页面更新间隔来衡量页面的热度,并使用 hot-delay 策略衡量擦除块在未来一段时间内的影响,以选择垃圾回收的数据块。

5. 发展趋势

人机物融合智能泛在操作系统持续发展:梅宏院士等[28]探讨了面向人机物融合泛在计算的泛在操作系统,认为泛在操作系统是操作系统发展的新蓝海。在此概念下,北京大学发布了一款面向工业物联网场景的泛在操作系统——XiUOS 矽琇工业物联网操作系统[29]。Syswonder 社区正推动在不同应用场景中的泛在操作系统的构建及技术应用,开发了基于 Rust 语言开发的 Unikernel 操作系统以及基于硬件分区的 Hypervisor [30]。

随着人、机、物的深度互联融合,泛在操作系统正由“场景适配”向“场景自适应”方向演进。未来的泛在操作系统将进一步实现跨平台、跨终端的智能协同,支持多态计算与实时资源编排。同时,可信计算与边缘智能将成为推动泛在操作系统落地的关键力量。未来,泛在操作系统有望在工业互联网、智慧城市和车联网等场景中形成生态级应用集群,构建以操作系统为核心的智能底座。

AI 原生嵌入式操作系统:学术界着重研究高效、轻量化的智能嵌入式操作系统。Luo 等人[31]提出了首个由深至浅的可转换神经架构搜索(NAS)范式——Double-Win NAS (DW-NAS),探索兼具深层网络高精度和浅层网络高效硬件性能的网络架构。Pasricha 等人[32]探讨了在物联网平台上部署机器学习软件的挑战,提出了简化部署的策略。Van Delm 等人[33]提出了 HTVM 编译器,最大限度地提高异构加速器的利用率并减少数据移动。Han 等人[34]提出了 DTMM 库,旨在解决在物联网设备上高效部署和执行机器学习模型的挑战。Xu 等人[35]提出了一种超轻量级神经网络,专为资源受限的低功耗设备设计。

AI 原生嵌入式操作系统正逐步由上述的融合 AI 模式迈向 AI 内生模式。未来系统将配备以 AI 推理引擎、模型/智能体调度器等构成的 AIOS 内核,形成 AIOS 内核与传统内核的深度共生架构。AI 原生 OS 不仅将赋能边缘端智能感知、决策与自优化能力,还将推动软硬件协同设计的新范式。随着 TinyML、边缘大模型与自适应能耗管理技术的发展,AI 原生操作系统有望成为物联网时代智能计算的关键支撑层。

Rust 成为热门内核开发语言:Rust 语言凭借其内存安全和类型安全的特性,逐渐成为嵌入式操作系统开发的热点工具。Li 等人[36]对 Rust-for-Linux 项目进行了首次实证研究,分析了 Rust 如何与 Linux 内核融合以及其在安全性和性能方面的表现。Hu 等人[37]提出一款基于 Rust 的高性能嵌入式单核内核,利用 Rust 的内存安全特性减少错误并确保可靠的资源管理。Narayanan 等人[38]开发了一个基于 Rust 开发的新型操作系统,通过语言级别的类型和内存安全实现轻量级细粒度隔离。Ma 等人[39]探讨了在基于 Rust 的软实时嵌入式操作系统中实现堆栈展开和恐慌恢复的可行性,针对资源受限环境提出了几种新颖的优化方法。中关村实验室联合发布的“星绽 OS”采用 Rust 语言构建框内核架构,最大限度减少非安全代码的比例,解决了传统 C 语言在内存安全方面的潜在问题。此外,Rust 在嵌入式操作系统领域的发展也存在一些问题,如 Sharma 等人[40]系统性研究了 Rust 在嵌入式系统中的应用现状和挑战,研究表明 Rust 需要进一步改进工具支持、增强与现有代码库的互操作性,并为开发人员提供更好的资源和支持,以推动 Rust 在嵌入式领域的普及。Ayers 等人[41]研究了 Rust 在嵌入式系统中二进制文件过大的问题,提出了减少二进制文件大小的编程原则。

随着 Rust-for-Linux 项目成熟及产业生态完善,Rust 有望成为下一代操作系统内核的主流开发语言。未来的发展重点将聚焦于工具链完备性、实时性优化及与 C/C++生态的无缝互操作。同时,Rust 在安全物联网、车载控制系统及国防级实时系统中的应用将进一步扩展,形成安全可信且高性能的系统开发新范式。

6. 结语

随着物联网、人工智能、工业自动化等技术的迅猛发展，嵌入式操作系统正从传统的“轻量、实时”向“安全、智能、泛在”方向演进，成为支撑未来智能互联世界的关键基础软件。本文系统梳理了嵌入式操作系统在安全架构、调度优化、文件系统等方面的研究进展，并探讨了智能化、泛在化及基于 Rust 语言的安全开发趋势。

未来的嵌入式操作系统将面临更加复杂的应用场景与更高的性能要求，不仅要保障系统的实时性与稳定性，还需应对多样化的安全威胁与资源受限的挑战。因此，发展具备自适应能力、智能协同机制与高可信保障的新型嵌入式操作系统，将是学术界与产业界共同努力的方向。

嵌入式操作系统的发展不仅是技术演进的体现，更是推动数字社会、智能产业和绿色计算的重要力量。随着新架构、新语言和新模型的不断涌现，嵌入式操作系统将在万物互联时代扮演更加核心的角色，助力构建更加安全、智能与可持续发展的计算生态。

基金项目

国家自然科学基金面上项目(62372073)，面向新型可计算存储的高效任务管理及调度研究，2024.1~2027.12。

参考文献

- [1] Hasan, M., Kashinath, A., Chen, C. and Mohan, S. (2024) Sok: Security in Real-Time Systems. *ACM Computing Surveys*, **56**, 1-31. <https://doi.org/10.1145/3649499>
- [2] Yadlapalli, Y., Zhou, H., Zhang, Y. and Liu, C. (2021) gGuard: Enabling Leakage-Resilient Memory Isolation in GPU-Accelerated Autonomous Embedded Systems. *2021 58th ACM/IEEE Design Automation Conference (DAC)*, San Francisco, 5-9 December 2021, 817-822. <https://doi.org/10.1109/dac18074.2021.9586244>
- [3] Ma, Z., Tan, X., Ziarek, L., Zhang, N., Hu, H. and Zhao, Z. (2023) Return-to-Non-Secure Vulnerabilities on ARM Cortex-M TrustZone: Attack and Defense. *2023 60th ACM/IEEE Design Automation Conference (DAC)*, San Francisco, 9-13 July 2023, 1-6. <https://doi.org/10.1109/dac56929.2023.10247972>
- [4] Park, H. and Lin, F.X. (2023) Safe and Practical GPU Computation in TrustZone. *Proceedings of the Eighteenth European Conference on Computer Systems*, Rome, 8-12 May 2023, 505-520. <https://doi.org/10.1145/3552326.3567483>
- [5] Kang, J., Park, J., Seo, J. and Kwon, D. (2024) Look before You Access: Efficient Heap Memory Safety for Embedded Systems on ARMv8-M. *Proceedings of the 61st ACM/IEEE Design Automation Conference*, San Francisco, 23-27 June 2024, 1-6. <https://doi.org/10.1145/3649329.3655949>
- [6] Guo, L. and Lin, F.X. (2022) Minimum Viable Device Drivers for ARM TrustZone. *Proceedings of the Seventeenth European Conference on Computer Systems*, Rennes, 5-8 April 2022, 300-316. <https://doi.org/10.1145/3492321.3519565>
- [7] 高沙沙, 王中华. 基于 MILS 架构的嵌入式操作系统多级安全域动态管理技术[J]. *计算机科学*, 2019, 46(S2): 460-463.
- [8] Feng, E., Feng, D., Du, D., Xia, Y., Zheng, W., Zhao, S., et al. (2024) SIOPMP: Scalable and Efficient I/O Protection for Tees. *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems*, Volume 2, La Jolla, 27 April 2024-1 May 2024, 1061-1076. <https://doi.org/10.1145/3620665.3640378>
- [9] Guo, X., Han, S., Hu, X.S., Jiao, X., Jin, Y., Kong, F., et al. (2021) Towards Scalable, Secure, and Smart Mission-Critical IoT Systems. *Proceedings of the 2021 International Conference on Embedded Software*, Virtual, 8-15, 2021 October 1-10. <https://doi.org/10.1145/3477244.3477624>
- [10] Yun, J., Rustamov, F., Kim, J. and Shin, Y. (2022) Fuzzing of Embedded Systems: A Survey. *ACM Computing Surveys*, **55**, 1-33. <https://doi.org/10.1145/3538644>
- [11] Shen, Y., Xu, Y., Sun, H., Liu, J., Xu, Z., Cui, A., et al. (2022) Tardis: Coverage-Guided Embedded Operating System Fuzzing. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **41**, 4563-4574. <https://doi.org/10.1109/tcad.2022.3198910>
- [12] Zhang, Q., Shen, Y., Liu, J., Xu, Y., Shi, H., Jiang, Y., et al. (2024) ECG: Augmenting Embedded Operating System

- Fuzzing via LLM-Based Corpus Generation. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **43**, 4238-4249. <https://doi.org/10.1109/tcad.2024.3447220>
- [13] Liu, J., Shen, Y., Xu, Y., Sun, H., Shi, H. and Jiang, Y. (2024) Effectively Sanitizing Embedded Operating Systems. *Proceedings of the 61st ACM/IEEE Design Automation Conference*, San Francisco, 23-27 June 2024, 1-6. <https://doi.org/10.1145/3649329.3658490>
- [14] Athalye, A., Corrigan-Gibbs, H., Kaashoek, F., Tassarotti, J. and Zeldovich, N. (2024) Modular Verification of Secure and Leakage-Free Systems: From Application Specification to Circuit-Level Implementation. *Proceedings of the ACM SIGOPS 30th Symposium on Operating Systems Principles*, Austin, 4-6 November 2024, 655-672. <https://doi.org/10.1145/3694715.3695956>
- [15] Pasquier, M., Jouault, F., Brun, M. and Pérochon, J. (2020) Evaluating Tool Support for Embedded Operating System Security. *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*, Virtual, 16-23 October 2020, 1-10. <https://doi.org/10.1145/3417990.3420048>
- [16] Mamata, R. and Azim, A. (2022) Work-in-Progress: A Resource-Aware Optimization Model for Real-Time Systems Analysis and Design. 2022 *International Conference on Embedded Software (EMSOFT)*, Shanghai, 7-14 October 2022, 9-10. <https://doi.org/10.1109/emsoft55006.2022.00012>
- [17] Reghenzani, F., Bhuiyan, A., Fornaciari, W. and Guo, Z. (2021) A Multi-Level DPM Approach for Real-Time DAG Tasks in Heterogeneous Processors. 2021 *IEEE Real-Time Systems Symposium (RTSS)*, Dortmund, 7 December 2021, 14-26. <https://doi.org/10.1109/rtss52674.2021.00014>
- [18] Li, C., Liang, Y., Ausavarungrun, R., Zhu, Z., Shi, L. and Xue, C.J. (2023) ICE: Collaborating Memory and Process Management for User Experience on Resource-Limited Mobile Devices. *Proceedings of the Eighteenth European Conference on Computer Systems*, Rome, 8-12 May 2023, 79-93. <https://doi.org/10.1145/3552326.3567497>
- [19] Jiang, Z., Zhao, S., Wei, R., Gao, Y. and Li, J. (2024) A Cache/Algorithm Co-Design for Parallel Real-Time Systems with Data Dependency on Multi/Many-Core System-on-Chips. *Proceedings of the 61st ACM/IEEE Design Automation Conference*, San Francisco, 23-27 June 2024, 1-6. <https://doi.org/10.1145/3649329.3657372>
- [20] Wu, Z. and Patel, H. (2022) Predictable Sharing of Last-Level Cache Partitions for Multi-Core Safety-Critical Systems. *Proceedings of the 59th ACM/IEEE Design Automation Conference*, San Francisco, 10-14 July 2022, 1273-1278. <https://doi.org/10.1145/3489517.3530614>
- [21] Chen, Z., Lei, H., Yang, M., Liao, Y. and Qiao, L. (2021) A Finer-Grained Blocking Analysis for Parallel Real-Time Tasks with Spin-Locks. 2021 *58th ACM/IEEE Design Automation Conference (DAC)*, San Francisco, 5-9 December 2021, 1177-1182. <https://doi.org/10.1109/dac18074.2021.9586270>
- [22] Zhang, R., Liu, D., Chen, X., She, X., Yang, C., Tan, Y., et al. (2020) LOFFS: A Low-Overhead File System for Large Flash Memory on Embedded Devices. 2020 *57th ACM/IEEE Design Automation Conference (DAC)*, San Francisco, 20-24 July 2020, 1-6. <https://doi.org/10.1109/dac18072.2020.9218635>
- [23] Jiang, J., Yang, M., Qiao, L., Wang, T. and Chen, X. (2025) MIFS: A Low Overhead and Efficient Mixture File Index Management Method in Flash File System. *Journal of Systems Architecture*, **162**, Article 103387. <https://doi.org/10.1016/j.sysarc.2025.103387>
- [24] Gao, X., Dong, M., Miao, X., et al. (2019) EROFS: A Compression-Friendly Readonly File System for Resource-Scarce Devices. 2019 *USENIX Annual Technical Conference*, Washington, 10-12 July 2019, 149-162.
- [25] Zhang, R., Liu, D., Shen, Z., She, X., Yang, C., Chen, X., et al. (2021) Bridging Mismatched Granularity between Embedded File Systems and Flash Memory. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **40**, 2024-2035. <https://doi.org/10.1109/tcad.2020.3036814>
- [26] Zhang, R., Liu, D., Chen, X., She, X., Yang, C., Tan, Y., et al. (2022) ELOFS: An Extensible Low-Overhead Flash File System for Resource-Scarce Embedded Devices. *IEEE Transactions on Computers*, **71**, 2327-2340. <https://doi.org/10.1109/tc.2022.3152079>
- [27] Sun, D., Song, Y., Chai, Y., Peng, B., Lu, F. and Deng, X. (2022) Light-GC. *Proceedings of the 23rd ACM/IFIP International Middleware Conference*, New York, 7-11 November 2022, 216-227. <https://doi.org/10.1145/3528535.3565246>
- [28] 梅宏, 曹东刚, 谢涛. 泛在操作系统: 面向人机物融合泛在计算的新蓝海[J]. 中国科学院院刊, 2022, 37(1): 30-37.
- [29] Cao, D., Xue, D., Ma, Z. and Mei, H. (2022) XiUOS: An Open-Source Ubiquitous Operating System for Industrial Internet of Things. *Science China Information Sciences*, **65**, Article 117101. <https://doi.org/10.1007/s11432-021-3294-y>
- [30] RuxOS 手册[Z/OL]. <https://ruxos.syswonder.org/>, 2025-09-15.
- [31] Luo, X., Liu, D., Kong, H., Huai, S. and Liu, W. (2024) Double-Win NAS: Towards Deep-to-Shallow Transformable Neural Architecture Search for Intelligent Embedded Systems. *Proceedings of the 61st ACM/IEEE Design Automation Conference*, San Francisco, 23-27 June 2024, 1-6. <https://doi.org/10.1145/3649329.3656247>
- [32] Pasricha, S. (2023) Lightning Talk: Efficient Embedded Machine Learning Deployment on Edge and IoT Devices. 2023

-
- 60th ACM/IEEE Design Automation Conference (DAC), San Francisco, 9-13 July 2023, 1-2. <https://doi.org/10.1109/dac56929.2023.10247845>
- [33] Van Delm, J., Vandersteegen, M., Burrello, A., Sarda, G.M., Conti, F., Pagliari, D.J., *et al.* (2023) HTVM: Efficient Neural Network Deployment on Heterogeneous TinyML Platforms. 2023 60th ACM/IEEE Design Automation Conference (DAC), San Francisco, 9-13 July 2023, 1-6. <https://doi.org/10.1109/dac56929.2023.10247664>
- [34] Han, L., Xiao, Z. and Li, Z. (2024) DTMM: Deploying TinyML Models on Extremely Weak IoT Devices with Pruning. *IEEE INFOCOM 2024-IEEE Conference on Computer Communications*, Vancouver, 20-23 May 2024, 1999-2008. <https://doi.org/10.1109/infocom52122.2024.10621387>
- [35] Xu, K., Li, Y., Zhang, H., Lai, R. and Gu, L. (2022) EtinyNet: Extremely Tiny Network for TinyML. *Proceedings of the AAAI Conference on Artificial Intelligence*, **36**, 4628-4636. <https://doi.org/10.1609/aaai.v36i4.20387>
- [36] Li, H., Guo, L., Yang, Y., *et al.* (2024) An Empirical Study of Rust-for-Linux: The Success, Dissatisfaction, and Compromise. 2024 *USENIX Annual Technical Conference (ATC)*, California, 10-12 July 2024, 425-443.
- [37] Hu, K., Wang, L., Mo, C. and Jiang, B. (2023) Work-in-Progress: Unishyper, a Reliable Rust-Based Unikernel for Embedded Scenarios. *Proceedings of the International Conference on Embedded Software*, Hamburg, 17-22 September 2023, 13-14. <https://doi.org/10.1145/3607890.3608459>
- [38] Narayanan, V., Huang, T., Detweiler, D., *et al.* (2020) RedLeaf: Isolation and Communication in a Safe Operating System. 14th *USENIX Symposium on Operating Systems Design and Implementation (OSDI)*, Virtual Event, 4-6 November 2020, 21-39.
- [39] Ma, Z., Chen, G. and Zhong, L. (2023) Panic Recovery in Rust-Based Embedded Systems. *Proceedings of the 12th Workshop on Programming Languages and Operating Systems*, Koblenz, 23 October 2023, 66-73. <https://doi.org/10.1145/3623759.3624549>
- [40] Sharma, A., Sharma, S., Tanksalkar, S.R., Torres-Arias, S. and Machiry, A. (2024) Rust for Embedded Systems: Current State and Open Problems. *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, Salt Lake, 14-18 October 2024, 2296-2310. <https://doi.org/10.1145/3658644.3690275>
- [41] Ayers, H., Laufer, E., Mure, P., Park, J., Rodelo, E., Rossman, T., *et al.* (2022) Tighten Rust's Belt: Shrinking Embedded Rust Binaries. *Proceedings of the 23rd ACM SIGPLAN/SIGBED International Conference on Languages, Compilers, and Tools for Embedded Systems*, San Diego, 14 June 2022, 121-132. <https://doi.org/10.1145/3519941.3535075>

嵌入式软件语言及开发新范式

陈铭松, 焦跃军, 杨永淦

华东师范大学软硬件协同设计技术与应用教育部工程研究中心, 上海

收稿日期: 2025年10月28日; 录用日期: 2025年11月20日; 发布日期: 2025年12月1日

摘要

嵌入式系统作为针对各类特定用途的计算平台, 已被广泛应用到航空航天、轨道交通、汽车电子以及智慧城市等领域。随着嵌入式系统复杂度增加, 传统嵌入式软件语言及开发面临性能瓶颈、资源受限、安全要求严苛及开发效率低下等多重挑战。传统嵌入式软件语言内存管理机制复杂不仅提高了学习门槛, 也增加了维护难度。嵌入式软件开发层次多、复杂性高、软硬件耦合紧密导致上市时间长难以快速迭代。近年来兴起的新型嵌入式开发语言、低代码平台、智能开发辅助工具以及虚拟仿真平台有效提升了嵌入式开发效率。文章将围绕这些新兴的嵌入式软件语言和开发范式, 综述国内外研究现状与发展趋势, 为嵌入式软件开发提供参考。

关键词

嵌入式软件, 软件开发语言, 低代码, 虚拟仿真

New Paradigms for Embedded Software Languages and Development

Mingsong Chen, Yuejun Jiao, Yonghao Yang

MoE Engineering Research Center of Software/Hardware Co-Design Technology and Application, East China Normal University, Shanghai

Received: October 28, 2025; accepted: November 20, 2025; published: December 1, 2025

Abstract

Embedded systems are application-specific computing platforms that have been widely used in various safety-critical domains, including aerospace, rail transit, automotive electronics, and smart cities. As the complexity of embedded systems increases, traditional embedded software languages and development processes face multiple challenges, including performance bottlenecks, limited resources, strict security requirements, and low development efficiency. The complex memory

management mechanisms of traditional embedded languages not only raise the learning threshold but also increase maintenance difficulty. The multiple layers, high complexity, and tight hardware-software coupling of embedded software development lead to long time-to-market and difficulty in rapid iteration. In recent years, emerging embedded development languages, low-code platforms, intelligent development tools, and virtual simulation platforms have been reshaping the paradigms for embedded software languages and development. This survey focuses on these emerging embedded software languages and development paradigms, reviews the current research status and development trends at home and abroad, and provides references for embedded software development.

Keywords

Embedded Software, Software Development Language, Low-Code, Virtual Simulation

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

嵌入式系统作为面向特定应用场景的计算平台，已广泛应用于航空航天、轨道交通、汽车电子以及智慧安防等领域。随着系统功能的持续扩展与集成度提升，嵌入式软件的开发语言与方法面临新的挑战。传统嵌入式开发语言虽然在执行性能和资源占用方面具有显著优势，但其内存管理机制复杂、语法门槛高，难以适应嵌入式项目的快速迭代需求[1]-[3]。同时，嵌入式系统的开发流程层次多、实现复杂度高，这导致一致性维护与软件验证成为开发效率瓶颈。此外，嵌入式开发中软硬件紧密耦合，从前期的系统规约、软硬件划分到具体的软硬件实现，持续对软件开发效率和优化迭代造成影响，从而限制了应用场景的扩展并延长了项目研发周期。

为了应对传统方法在嵌入式软件开发过程中展现的弊端，围绕新型嵌入式开发语言、低代码平台、智能开发辅助工具以及虚拟仿真平台的研究正成为业界与学术界关注的重点。本文将系统剖析当前嵌入式开发所面临的关键技术挑战，探讨这些嵌入式软件语言与开发范式在嵌入式领域的核心机制、成功案例与工业实践，为嵌入式软件开发提供参考。

2. 嵌入式系统开发挑战

嵌入式系统开发面临在有限时间内实现高性能与高可靠性的双重压力[4]。随着市场竞争推动上市时间[5] (Time-to-Market) 要求不断缩短，传统的嵌入式软件开发方法逐渐暴露出在复杂系统中难以兼顾开发效率与产品质量的局限性。本章将分析嵌入式开发过程中面临的挑战，从开发语言的设计局限与开发流程的层次复杂两个方面展开探讨。

2.1. 嵌入式开发语言设计局限多

嵌入式系统开发领域存在语言种类繁多、学习周期长等问题。由于不同应用场景往往采用特定的 DSL (Domain-Specific Language, 领域特定语言) 或平台相关框架，开发者不仅需要掌握底层硬件接口与实时操作系统机制，还必须熟悉多种语言特性与工具链，从而显著延长了学习与开发周期。在实际工程中，传统通用语言(如 C/C++)虽然在性能优化与硬件控制方面具备优势，但其语法允许程序直接操作内存地址，缺乏访问约束与内存安全机制，极易引发缓冲区越界与指针错误，对系统可靠性构成长期威胁[6] [7]。

此外，随着嵌入式软件代码规模普遍达到百万行级别，这些问题的检测与修复难度急剧增加。传统事后审查的排错方式需要根据故障现象排查漏洞。由于嵌入式系统运行环境复杂，运行时暴露的问题往往难以稳定复现，使故障定位与修复过程更加困难。

2.2. 嵌入式开发流程复杂度高

嵌入式系统开发是一个贯穿方案设计、系统实现到产品维护的多阶段复杂工程，其流程层次多、环节复杂导致一致性维护难度大。嵌入式系统开发的典型流程如图 1 所示。在方案设计阶段，系统架构师与软件工程师往往因专业背景差异产生认知偏差，导致设计意图与技术实现间存在落差[8] [9]。在项目实现阶段，开发团队成员能力与习惯差异可能降低代码质量并引入潜在漏洞[10]。同时，受限于硬件资源与实现复杂度，开发目标常被迫调整为简化方案，从而削弱系统性能优化空间[11]。在维护阶段，复杂的技术文档使问题定位与修复过程高度依赖开发人员，延长了纠错周期。

此外，嵌入式软件与硬件之间的强耦合关系极大延长了设计周期和开发周期。在开发前期，若目标板卡或外设尚未就绪，驱动程序、BSP (Board Support Package, 板级支持包)及底层固件等模块无法在真实环境中进行验证与调优，团队只能依赖功能受限的模拟环境或延迟实现[12] [13]。而在后期集成阶段，软件测试往往需等待硬件原型完成后才能开展，且硬件验证(从印制电路板制造到原型调试)周期长、成本高，进一步压缩了软件测试与优化的空间。这种软硬件紧密耦合的串行开发模式导致开发流程缺乏灵活性与自动化支持，使嵌入式系统难以实现快速迭代与持续优化。

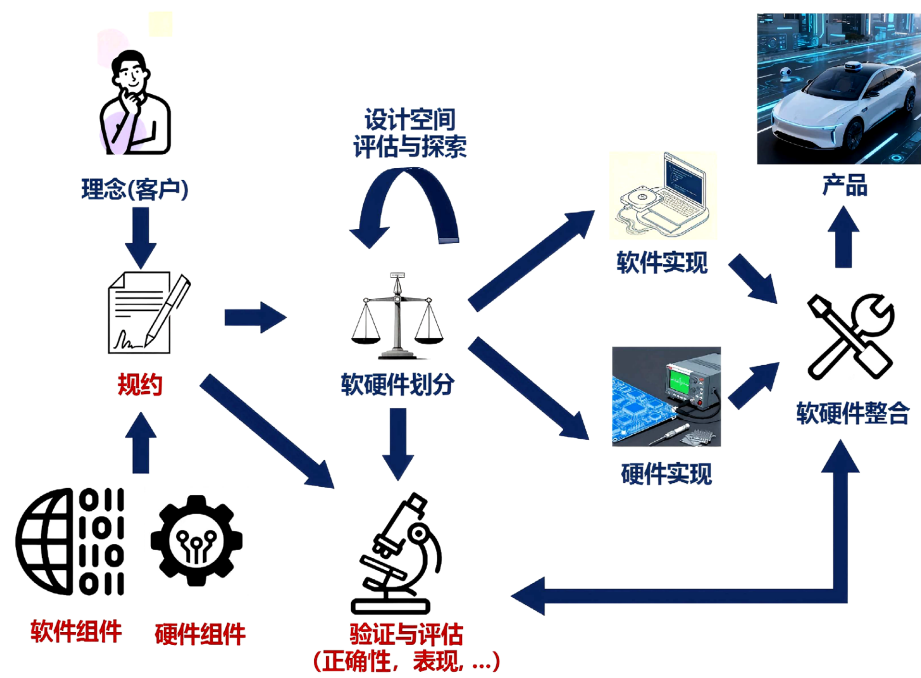


Figure 1. General development process of embedded systems
图 1. 嵌入式系统开发基本流程

3. 嵌入式软件语言创新趋势

以 C/C++为代表的传统嵌入式软件语言在其性能与泛用性上已经得到了充分验证，但基于传统软件语言的工业项目在长期运行中暴露出了安全隐患与漏洞。此类项目的代码修复需要从运行中暴露出的问

题出发,通过问题复现与事后代码分析来进行问题定位。不适应当前“高精度,高实时,高可信”的嵌入式开发要求。目前新型嵌入式软件语言的特征主要集中在以下几个方面。其一是运行前消除,在编译过程中通过合理安全规则检出代码潜在漏洞。其二是运行时安全,通过运行时异常处理策略来避免漏洞损失。其三是保证代码的运行效率,在保证系统安全的基础上尽力贴近 C/C++的运行效率。本章将列举分析 Ada、Go 与 Rust 等典型新型嵌入式语言的特性与进一步发展方向,并结合实际案例,说明其如何在实际应用中构建高可信的嵌入式开发项目。

3.1. 新型嵌入式语言

新型嵌入式软件语言的核心设计理念是将传统由开发者负责的内存安全保障,转变为由语言规范与编译器强制执行的约束条件。其中一种设计是利用先进的类型系统、所有权模型和生命周期分析等技术,在编译阶段识别潜在的内存错误,从源头上消除安全隐患。另一种解决方案是在运行时采用垃圾回收或智能指针等技术,自动化管理内存分配与释放。在嵌入式系统开发中,不同编程语言在内存安全与执行效率的平衡上呈现出差异化特征。目前工业项目中较常见的新型嵌入式软件语言包括 Ada, Go 及 Rust。

Ada 语言在嵌入式领域积累了深厚的应用基础。它的强类型系统、契约编程(含前置与后置条件)及并发任务模型,使其在航空航天、国防等对可靠性要求极高的场景中被广泛采用。通过 SPARK 子集及配套工具(如 SPARK Examiner、CodePeer),Ada [14] [15]可实现静态分析,对缓冲区越界、空/悬挂指针等潜在运行时错误进行验证,显著增强系统的可靠性与容错能力。

Go 语言则通过垃圾回收机制保障内存安全,并借助其 Go routine 轻量线程与 channel 工具简化并发实现,在快速开发、系统原型验证及非硬实时系统中具有优势。然而,Go 语言的垃圾回收机制引入的不可预测暂停使其难以适配硬实时任务[16]。已有 TinyGo 项目[17]裁剪 Go 运行时检测机制进行优化,目前已应用于驱动开发、IO 控制、OTA 客户端等轻量级任务,在部分场景中满足了基本需求。但在严格硬实时的安全关键系统中,Go 语言垃圾回收机制导致的延迟与内存开销仍是未完全解决的限制[18]。



Figure 2. Memory error cases and their Rust solutions
图 2. 内存错误案例及其 Rust 解决方案

Rust 语言低运行成本的安全管理使其在嵌入式领域的关注度显著提升。Rust 核心优势在于通过独特

的所有权系统与借用检查机制，在编译阶段时保障了内存安全与无数据竞争的并发执行，因此在无需依赖垃圾回收机制的情况下，实现了安全性与性能间的更优平衡[19]。相较于传统 C/C++，Rust 在保留底层硬件控制能力的同时，从语言层面消除了多数悬挂指针、缓冲区越界等内存错误。图 2 中展示了 Rust 编译器可检出的常见内存漏洞。多项研究[20]-[22]结果显示，在资源密集型基准测试中，Rust 程序的执行时间相比于同功能 C++程序仅增加个位数百分比，且内存峰值消耗更低。这种在内存安全与效率上的双重优势，使得 Rust 目前已在嵌入式系统开发中获得较为广泛的应用。

总体而言，当前新型嵌入式编程语言在嵌入式系统的安全与性能平衡方面已取得了优异表现，但 Ada 的复杂语法，Go 的运行时开销，Rust 的复杂编译等问题仍限制了更广泛的应用。新兴开发语言正试图从以下方面取得突破，一是通过语义简化与模块化设计以提升编译效率与开发体验，二是在静态安全性与可用性之间寻求平衡，以降低入门门槛，三是在垃圾回收机制上实现可配置或轻量化设计，以适应边缘计算与云端快速迭代需求。当前嵌入式开发语言的研究趋向于在安全性、可扩展性与性能表现之间实现动态平衡，以满足更复杂的嵌入式系统需求。

3.2. 工业应用案例

已有大量新型嵌入式开发语言应用于开源项目或大型工业项目。如表 1 所示，新型嵌入式开发语言在嵌入式软件开发中广泛应用。例如 Ada 语言在航空航天、轨道交通等安全关键型系统中依旧保持稳健地位。代表性实现包括 SPARK/Ada [23]及 Ravenscar Profile [24]，通过形式化验证与任务调度约束，为波音飞控系统等高安全标准项目提供了长期可靠的技术支撑。Go 语言在嵌入式领域的探索更多集中于边缘计算与云端协同类场景。典型项目如 TinyGo [17]与 Periph.io [25]，前者通过轻量化编译链实现对 ARM Cortex-M、RISC-V 等平台的原生支持，已被应用于物联网终端、工业监测节点等场景。后者则通过统一的硬件抽象层(HAL)接口，简化了基于 Go 的设备驱动与通信协议开发。

与此同时，Rust 语言因其高效且安全的特性，对嵌入式操作系统有较大应用价值。TockOS [26]是由 Rust 实现的嵌入式实时操作系统，保证必要的不安全操作(如直接硬件寄存器访问)被明确标记并封装于受信任的驱动模块中，缩小了潜在危险代码范围，其具体内核安全区分如图 3 所示。清华大学团队以 Rust 为基础语言研发的 ArceOS [27]可以快速定制适配不同场景的内核。ArceOS 仅需新增几千行代码，即可实现功能完善的宏内核 StarryOS [28]，而替换插件为 vCPU 管理模块后，又能快速形成 Hypervisor AxVisor [29]，其应用 Rust 进行的开发设计兼顾了内核的性能、安全性与可移植性。

总体来看，Go 与 Ada 在其特长领域表现稳定，而 Rust 凭借其编译期内存安全机制与较低运行时损耗，正在成为当前嵌入式开发语言中最受重视、增长最迅速的方向。

Table 1. Large-scale industrial projects/open source projects developed using memory-safe languages
表 1. 使用内存安全型语言开发的大型工业项目/开源项目

使用语言	项目名称	核心特征
GO	TinyGo/Embedded Go	TinyGo 允许将 Go 代码编译用于资源受限的嵌入式设备。
	Mender OTA 客户端	Mender 团队使用 Go 语言编写嵌入式客户端应用。
Ada	Boeing 777 飞控系统	飞机飞控系统采用 Ada 编写，确保高可靠性与容错能力。
Rust	Tock OS	面向微控制器的 Rust 安全嵌入式实时操作系统。
	Ariel OS	支持多核抢占式调度的 Rust 嵌入式操作系统。
	蓝河 OS	国内行业首个全栈 Rust 语言编写的操作系统。

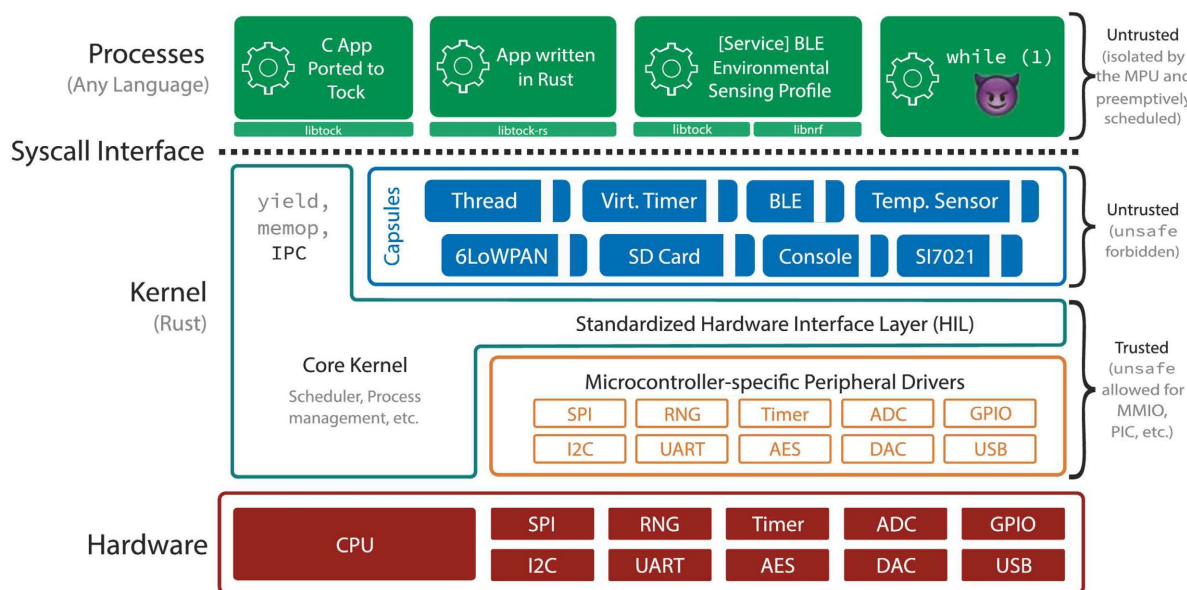


Figure 3. The architecture of TockOS [26]

图 3. TockOS 的基本架构[26]

4. 嵌入式软件开发演进趋势

随着嵌入式系统功能复杂度的不断提升,传统开发模式在性能优化与开发效率之间的矛盾愈发突出。嵌入式软件开发虽然允许底层代码对硬件进行精细控制,从而充分发挥有限资源的潜力、满足实时性与可靠性要求,但其学习门槛高、开发层次多、自动化工具少及软硬件耦合紧密等问题已成为制约创新效率的瓶颈。为应对上述挑战,在嵌入式软件开发领域已涌现出低代码平台、智能开发辅助工具以及虚拟仿真平台的应用案例。低代码平台通过可视化编程显著降低开发门槛,促进跨领域协作。智能辅助开发工具利用大模型与智能体技术,在软件实现与测试评估环节提供持续决策支持。虚拟仿真平台则通过解耦软硬件开发流程,使软件得以在硬件未就绪阶段完成基本功能验证与性能优化,从根本上突破了传统串行开发的限制。本章将系统性地阐述这三类技术的特征与优势,分析它们如何共同推动嵌入式系统开发流程向智能化、自动化与协同化方向演进。

4.1. 嵌入式低代码平台

嵌入式低代码平台通过可视化编程显著降低了开发门槛,使多领域专家能够直接参与应用逻辑的构建过程。嵌入式低代码平台核心技术优势主要体现在三个方面。一是通过拖拽式操作定义功能模块与业务流程,构建可视化开发环境,使开发者能够专注于业务逻辑而非实现细节。二是将实际编码量大幅降低,提升开发效率。三是通过可视化的业务流清晰表达设计意图,有效减少了与专业软件开发团队之间的理解偏差,显著降低了沟通成本。在架构设计上,嵌入式低代码平台通常以节点编程为核心,将应用逻辑转化为由功能节点及其连接关系所构成的数据流图来进行系统定义与实现。每个功能节点代表独立的处理模块。开发者通过连接线定义节点间的数据流向与控制逻辑。

在嵌入式低代码开发领域,已涌现出一批具有代表性的技术平台。国际新兴平台(如 Node-RED、XOD、Visuino 等)普遍侧重于通过节点式可视化编程实现快速的产品原型构建。例如 Node-RED [30]采用基于流(Flow)的可视化编程模型,将应用组织为由输入、逻辑处理与输出节点构成的流程结构。数据在节点之间沿既定路径依次传递与处理,使复杂的底层实现被封装于节点内部。开发者只需通过节点组合与流程编

排即可完成功能构建,从而显著提升开发的直观性与效率。国内相关平台在具体业务应用上有较多实践,其通常与某一类专用硬件设备或某类具体任务紧密结合。例如中科创达 ModelFarm 平台基于其公司设备,专注图像识别任务的低代码开发。

4.2. 嵌入式智能开发辅助工具

嵌入式智能开发辅助工具正逐步成为提升嵌入式系统开发效率与质量的重要手段。这类工具不同于一般的代码补全与生成工具,它们通过集成大模型推理与多智能体协作机制,能够从代码开发到运维管理进行长周期的决策与执行支持。

在开发阶段,基于大模型的工具(如 GitHub Copilot、Tabnine 与 OpenAI Codex)能够理解嵌入式代码上下文,并在特定语义层面生成 RTOS 调度逻辑、硬件驱动模板及中断服务框架。其中, Codex [31] 不仅能进行上下文感知代码生成,还可自动构造测试场景以提高覆盖率与系统稳定性。进一步的研究如 AutoDev [32] 框架展示了“人工智能代理 + 沙箱环境”的模式。智能体可在受控环境中自动执行编译、调试及优化任务,从而减少人工循环迭代成本。在运维阶段,智能体的作用从开发支持延伸至系统生命周期管理。联通数科已实现设备管理智能体,它可以进行基于自然语言的设备状态查询、故障定位与策略调整,使嵌入式系统的运维由被动监控转向主动诊断与自优化。总体而言,这类嵌入式智能辅助开发工具显著提升了嵌入式系统的开发效率、代码质量与可维护性。

4.3. 虚拟仿真平台

虚拟仿真平台是解决嵌入式系统开发中硬件资源依赖问题的关键技术。其核心思想是通过在软件开发阶段构建虚拟硬件环境,实现软硬件开发流程的解耦,使软件模块能够在硬件尚未到位时独立完成编码与测试。该技术的应用极大缩短了系统集成周期,并减少了硬件迭代带来的开发延误与成本开销。

嵌入式系统的仿真平台经历了从 RTL (Register Transfer Level, 寄存器传输级) 仿真到 ESL (Electronic System Level, 电子系统级) 仿真的演化过程。RTL 仿真面向硬件逻辑验证,能够准确反映信号变化,但由于仿真粒度细和效率低的问题,导致其难以支撑大规模系统的快速验证。ESL 建模的提出标志着从硬件行为模拟到系统行为模拟的层次提升,它通过 TLM [33] (Transaction Level Modeling, 事务级建模) 将硬件通信抽象为高层交互事件,在保证功能一致性的同时显著提升仿真效率,允许软件开发脱离硬件设计进度进行前期软件开发、系统集成验证与虚拟原型(Virtual Prototype)构建。

现代虚拟仿真平台普遍采用接口抽象与模块化设计,以提升系统的可扩展性与移植性。例如, VCML [34] (Virtual Component Modeling Library, 虚拟组件模型库) 基于 SystemC/TLM 框架构建,提供标准化的组件接口与时序控制机制。图 4 演示了使用 VCML 组件进行虚拟平台构建。在汽车电子领域, SystemC/TLM 技术支持开发者对多 ECU (Electronic Control Unit, 电子控制单元) 单元与车辆动力学模型进行并发仿真[35],实现并行测试与快速迭代开发。AIoTML 框架[36]通过平台无关建模与模块化编译器架构,实现了面向 AIoT 系统的虚拟化仿真与控制模型统一抽象。Renode [37] 则通过可插拔接口支持多架构 CPU 与外设的快速集成,加速嵌入式系统验证。深圳航天科技创新研究院[38]通过高精度元器件模型与实时数据接口,实现软硬件闭环验证。这些平台通过统一的事务接口和协同仿真机制,使多核处理器、通信总线、外设模型等组件能够在同一环境中高效协作。

总体而言,虚拟仿真平台的发展通过实现软硬件解耦,使嵌入式系统开发模式从依赖硬件的串行流程向软硬件并行协同演进,从而显著提升了开发效率。其高抽象层次的系统建模方法,不仅为复杂嵌入式系统的前期快速验证提供了可行途径,也为智能化开发工具和自动化验证体系的构建奠定了技术基础。

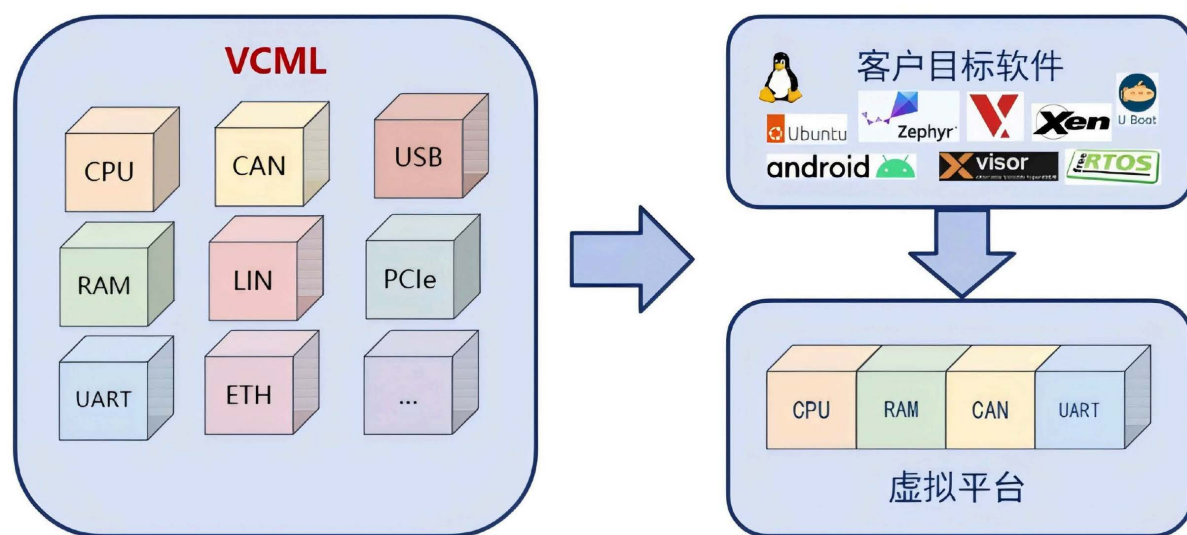


Figure 4. Illustration of virtual platform construction with VCML [34]

图 4. 基于 VCML 的虚拟平台构建示意图[34]

5. 结束语

随着嵌入式系统向“人-机-物”系统演化，嵌入式系统开发语言及开发新范式正以“安全、泛在、高效”为目标加速演进。各类创新技术的协同融合持续推动嵌入式开发向更高效、更可信的阶段跨越。具体来说，新型嵌入式软件语言通过运行前消除机制与运行时安全策略，从根源防范内存泄漏、数据竞争等底层风险。低代码平台降低了嵌入式开发的入门门槛，吸引更多跨领域群体参与嵌入式创新。智能开发辅助工具为专业开发者提供支持，帮助他们突破复杂场景的开发限制。虚拟仿真平台通过软硬件解耦，打破了软件开发对物理硬件依赖的效率瓶颈。随着技术融合的持续深化，嵌入式系统将更广泛地适配特定应用领域的需求。无论是关键行业的核心控制场景，还是民生领域的智能应用场景，都将依托更成熟的开发技术实现创新突破。嵌入式软件语言及开发新范式将加快推动嵌入式系统开发完成从技术迭代到价值落地的全面跃升，为各行业数字化转型提供更坚实的底层支撑。

参考文献

- [1] CWE-119: Improper Restriction of Operations within the Bounds of a Memory Buffer. <https://cwe.mitre.org/data/definitions/119.html>
- [2] Rules for Developing Safe, Reliable, and Secure Systems (2016 Edition). <https://www.sei.cmu.edu/library/sei-cert-c-coding-standard-rules-for-developing-safe-reliable-and-secure-systems-2016-edition>
- [3] Secure by Design Alert: Eliminating Buffer Overflow Vulnerabilities. <https://www.cisa.gov/resources-tools/resources/secure-design-alert-eliminating-buffer-overflow-vulnerabilities>
- [4] Sas, D. and Avgeriou, P. (2020) Quality Attribute Trade-Offs in the Embedded Systems Industry: An Exploratory Case Study. *Software Quality Journal*, **28**, 505-534. <https://doi.org/10.1007/s11219-019-09478-x>
- [5] van der Spek, P. and Verhoef, C. (2014) Balancing Time-to-Market and Quality in Embedded Systems. *Systems Engineering*, **17**, 166-192. <https://doi.org/10.1002/sys.21261>
- [6] Cimpanu, C. (2019) Microsoft: 70 Percent of All Security Bugs Are Memory Safety Issues. <https://www.zdnet.com/article/microsoft-70-percent-of-all-security-bugs-are-memory-safety-issues>
- [7] Eliminating Memory Safety Vulnerabilities at the Source. <https://sechub.in/view/2949364>
- [8] Mohanani, R., Salman, I., Turhan, B., Rodriguez, P. and Ralph, P. (2018) Cognitive Biases in Software Engineering: A Systematic Mapping Study. *IEEE Transactions on Software Engineering*, **46**, 1318-1339.

- <https://doi.org/10.1109/tse.2018.2877759>
- [9] McDermott, T.A., Folds, D.J. and Hallo, L. (2020) Addressing Cognitive Bias in Systems Engineering Teams. *INCOSE International Symposium*, **30**, 257-271. <https://doi.org/10.1002/j.2334-5837.2020.00721.x>
 - [10] Tornhill, A. and Borg, M. (2022) Code Red: The Business Impact of Code Quality—A Quantitative Study of 39 Proprietary Production Codebases. *Proceedings of the International Conference on Technical Debt*, Pittsburgh, 16-18 May 2022, 11-20. <https://doi.org/10.1145/3524843.3528091>
 - [11] Kaisti, M., Rantala, V., Mujunen, T., Hyrynsalmi, S., Könnölä, K., Mäkilä, T., *et al.* (2013) Agile Methods for Embedded Systems Development—A Literature Review and a Mapping Study. *EURASIP Journal on Embedded Systems*, **2013**, Article No. 15. <https://doi.org/10.1186/1687-3963-2013-15>
 - [12] Herdt, V. and Drechsler, R. (2022) Advanced Virtual Prototyping for Cyber-Physical Systems Using RISC-V: Implementation, Verification and Challenges. *Science China Information Sciences*, **65**, Article 110201. <https://doi.org/10.1007/s11432-020-3308-4>
 - [13] Liukkunen, K., Eteläperä, M., Oivo, M., Soininen, J. and Pellikka, M. (2008) Virtual Prototypes in Developing Mobile Software Applications and Devices. In: Jedlitschka, A. and Salo, O., Eds., *International Conference on Product Focused Software Process Improvement*, Springer, 174-188. https://doi.org/10.1007/978-3-540-69566-0_16
 - [14] Jhaloyan, G.A., Moy, Y. and Paskevich, A. (2018) Borrowing Safe Pointers from Rust in Spark. arXiv:1805.05576.
 - [15] GNAT Pro Assurance: Product for Safety-Critical Systems. <https://www.adacore.com/gnatpro/assurance>
 - [16] Adapting Go for Embedded/Real-Time Use. <https://groups.google.com/g/golang-nuts/c/6MPAzFgoS7M>
 - [17] Microcontroller Targets, TinyGo Documentation. <https://tinygo.org/docs/concepts/compiler-internals/microcontrollers>
 - [18] TinyGo: Driver Design and Development. <https://tinygo.org/docs/guides/driver-design>
 - [19] Sharma, A., Sharma, S., Tanksalkar, S.R., Torres-Arias, S. and Machiry, A. (2024) Rust for Embedded Systems: Current State and Open Problems. *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security*, Salt Lake, 14-18 October 2024, 2296-2310. <https://doi.org/10.1145/3658644.3690275>
 - [20] Rust vs C Speed and Zero-Cost Abstractions. <https://kornel.ski/rust-c-speed>
 - [21] Martins, E.M., Faé, L.G., Hoffmann, R.B., *et al.* (2025) NPB-Rust: NAS Parallel Benchmarks in Rust. arXiv:2502.15536.
 - [22] Benchmarks: Rust vs C/C++ Performance Comparisons. <https://benchmarksgame-team.pages.debian.net/benchmarksgame>
 - [23] SPARK/Ada: Formal Verification for High-Integrity Software. <https://www.adacore.com/sparkpro>
 - [24] Ravenscar Profile (Ada Real-Time Subset). https://en.wikipedia.org/wiki/Ravenscar_profile
 - [25] Peripheral I/O in Go. <https://periph.io>
 - [26] Levy, A., Campbell, B., Ghena, B., Giffin, D.B., Pannuto, P., Dutta, P., *et al.* (2017) Multiprogramming a 64kb Computer Safely and Efficiently. *Proceedings of the 26th Symposium on Operating Systems Principles*, Shanghai, 28 October 2017, 234-251. <https://doi.org/10.1145/3132747.3132786>
 - [27] ArceOS: An Experimental Modular OS Written in Rust. <https://github.com/arceos-org/arceos>
 - [28] Starry Tutorial Book. <https://azure-stars.github.io/Starry-Tutorial-Book>
 - [29] AxVisor Tutorial Book. <https://arceos-hypervisor.github.io/doc/architecture/axvisor.html>
 - [30] Low-Code Programming for Event-Driven Applications. <https://nodered.org>
 - [31] Chen, M., Tworek, J., Jun, H., *et al.* (2021) Evaluating Large Language Models Trained on Code. arXiv:2107.03374.
 - [32] Tufano, M., Agarwal, A., Jang, J., *et al.* (2024) AutoDev: Automated AI-Driven Development. arXiv:2403.08299.
 - [33] Mahmoudi, A., Nešković, A., Thermann, C., Sehm, R., Hübner, C., Plattenteich, T., *et al.* (2025) A Systematic Mapping Study on Systemc/TLM Modeling Capabilities in New Research Domains. *ACM Transactions on Design Automation of Electronic Systems*, **30**, 1-41. <https://doi.org/10.1145/3735641>
 - [34] Virtual Components Modeling Library. <https://github.com/machineware-gmbh/vcm1>
 - [35] Kim, H., Kwak, J. and Cho, J. (2024) Autosar-Compatible Level-4 Virtual ECU for the Verification of the Target Binary for Cloud-Native Development. *Electronics*, **13**, Article 3704. <https://doi.org/10.3390/electronics13183704>
 - [36] Hu, M., Cao, E., Huang, H., Zhang, M., Chen, X. and Chen, M. (2023) AIOtML: A Unified Modeling Language for AIOt-Based Cyber-Physical Systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **42**, 3545-3558. <https://doi.org/10.1109/tcad.2023.3264786>
 - [37] Antmicro Platform Renode. <https://antmicro.com/platforms/renode>
 - [38] 深圳航天科技创新研究院[EB/OL]. <https://app.puliedu.com/#/portals>, 2025-10-10.

嵌入式网络技术现状与发展趋势

冯智伟, 邓庆绪

东北大学计算机科学与工程学院, 辽宁 沈阳

收稿日期: 2025年10月28日; 录用日期: 2025年11月20日; 发布日期: 2025年12月1日

摘要

嵌入式网络正处于由“万物互联”向“万物智联”演进的关键阶段, 对低时延、高可靠与异构融合提出了更高要求。最近, 国际研究聚焦于体系架构创新与范式重塑, 如太赫兹通信及时间敏感网络调度优化; 国内研究则更强调工程可行性与系统集成, 例如5G-A与TSN异构融合、自主短距协议星闪、WiFi智能感知及RFID精确定位等方面取得进展。未来, 嵌入式网络将沿着“通信-感知-计算-智能”一体化等方向演进, 为工业、车载与智慧城市等场景提供确定性与智能化的网络支撑。

关键词

嵌入式网络, 5G/6G, 时间敏感网络(TSN), 低功耗广域网(LPWAN), 星闪(SparkLink), WiFi智能感知, RFID

The Current Status and Development Trends of Embedded Network Technology

Zhiwei Feng, Qingxu Deng

School of Computer Science and Engineering, Northeastern University, Shenyang Liaoning

Received: October 28, 2025; accepted: November 20, 2025; published: December 1, 2025

Abstract

Embedded networks are at a pivotal stage in their transition from “connecting everything” to “intelligently connecting everything”, imposing stricter requirements for ultra-low latency, high reliability, and heterogeneous integration. Recent international research emphasizes architectural innovation and paradigm shifts—such as terahertz communications, and optimization of Time-Sensitive Networking (TSN) scheduling—whereas domestic efforts prioritize engineering feasibility and

system integration, with advances in 5G-Advanced (5G-A)-TSN heterogeneous convergence, the indigenous short-range protocol NearLink, Wi-Fi-based intelligent sensing, and RFID-based precise localization. Looking ahead, embedded networks will evolve toward integrated “communication-sensing-computing-intelligence”, providing deterministic and intelligent networking support for industrial, vehicular, and smart-city scenarios.

Keywords

Embedded Networks, 5G/6G, Time-Sensitive Networking (TSN), Low Power Wide Area Network (LPWAN), SparkLink, WiFi Sensing, RFID

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

在智能化浪潮加速推进的背景下，嵌入式网络正处于从“万物互联”向“万物智联”转型的关键阶段，其演进路径如图 1 所示。作为人-机-物深度融合的基础设施，嵌入式网络不仅需要在通信层面满足低时延、高可靠、高并发、高带宽等多场景差异化需求，更要满足嵌入式网络通信异构互联需求。从技术演进角度看，嵌入式网络通信技术经历了从早期的串口(UART)、CAN 总线、Wi-Fi、蓝牙、ZigBee、RFID 等，到近年来发展出了 5G-Advanced (5G-A, 5G 演进)、第六代移动通信(6G)、时间敏感网络(TSN)、星闪(SparkLink)以及低功耗广域网(如 LoRaWAN)等。这些技术推动通信、感知与计算的一体化融合，加速嵌入式网络从“连接为主”向“感传智一体化”的范式跃迁。面对新一代工业互联网、车载协同系统、智能终端及智慧城市等复杂场景需求，嵌入式网络正日益成为承载智能服务与实时交互的关键支撑平台。本章将围绕上述演进趋势，系统梳理国内外在嵌入式网络领域的主要研究成果与技术路径，重点对比不同通信技术的适用性，并进一步探讨未来嵌入式网络在架构、协议与智能能力等方面的发展方向与关键挑战。

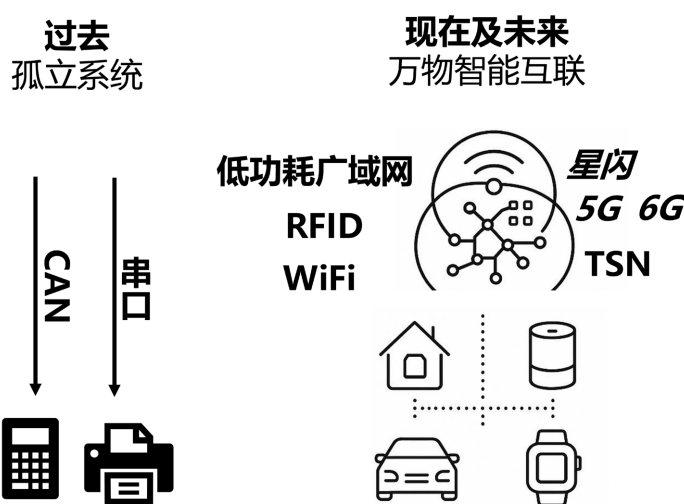


Figure 1. From isolated systems to the intelligent interconnection of all things

图 1. 从孤立系统到万物智能互联

2. 国外研究现状

随着移动通信技术代际更替，通信系统不仅在数据速率、连接密度和时延控制等物理层性能指标上持续提升，其服务形态亦从基础语音与短信逐步演化为支持高清视频、增强现实(AR)、沉浸式虚拟现实等多样化形式，如图 2 所示。

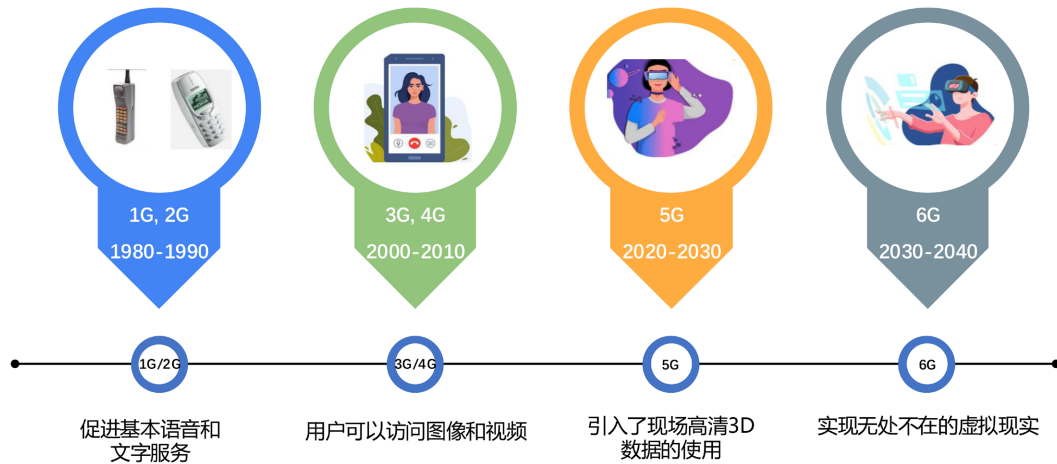


Figure 2. From voice to virtual reality: the mobile communication revolution from 1G to 6G
图 2. 从语音到虚拟现实：1G 至 6G 的移动通信革命

5G-A 是基于 5G 网络在功能上和覆盖上的演进和增强，是支撑互联网产业 3D 化、云化、万物互联智能化、通信感知一体化和智能制造柔性化等产业数字化升级的关键信息化技术。在 5G 与 6G 的相关研究中，国际研究聚焦于网络架构的重构与跨域融合能力的构建，深入探索面向未来网络的基本能力体系。例如，Siddiky 等人提出了一套结合非 IP 协议与信息中心网络的新型架构，旨在系统性地提升 5G 网络效率与灵活性[1]。为实现 6G 所构想的太比特级(Tbps)传输速率，El Mattar 等人将研究焦点集中于太赫兹 (THz)与光无线通信等超高频段，并将其视为突破传统频谱瓶颈的关键路径[2]。相较于 5G，6G 不仅在峰值速率、端到端时延、可靠性及连接数密度等关键性能指标上实现量级提升(如表 1 所示)，还将与人工智能技术深度融合，构建能够连接物理世界与虚拟世界的智能化网络，最终实现人、机、物与虚拟空间的全面互联互通。

Table 1. Performance comparison of 5G and 6G
表 1. 5G 与 6G 性能对比表

性能	5G	6G
峰值数据速率	20 Gbit/s	1~10 Tbit/s
用户体验数据速率	0.1 Gbit/s	1 Gbit/s
时延	>1 ms	<0.3 ms
移动速度	<500 km/h	1000 km/h
全球覆盖率	大约 70%	>99.9%
移动速度	500 km/h	>800 km/h
区域流量密度	0.01 Gbit/(s·m ²)	1 Gbit/(s·m ²)
网络能源效率	比 4G 高 10~100 倍	比 5G 高 10~100 倍
频谱效率	比 4G 高 3~5 倍	比 5G 高 3~5 倍

时间敏感网络是 IEEE 802.1 工作小组中的 TSN 工作小组发展的系列标准, 用于定义以太网上时间敏感传输的机制。其核心方向包括时钟同步、调度机制、队列管理与可靠性增强。近年来, 围绕 IEEE 802.1Q 系列展开多个标准修正案(如 ASds [3]、ASeb [4]、Qdq [5])的修订, 涵盖对同步精度、整形器配置与带宽分配等方面。TSN 标准架构[6]如图 3 所示。

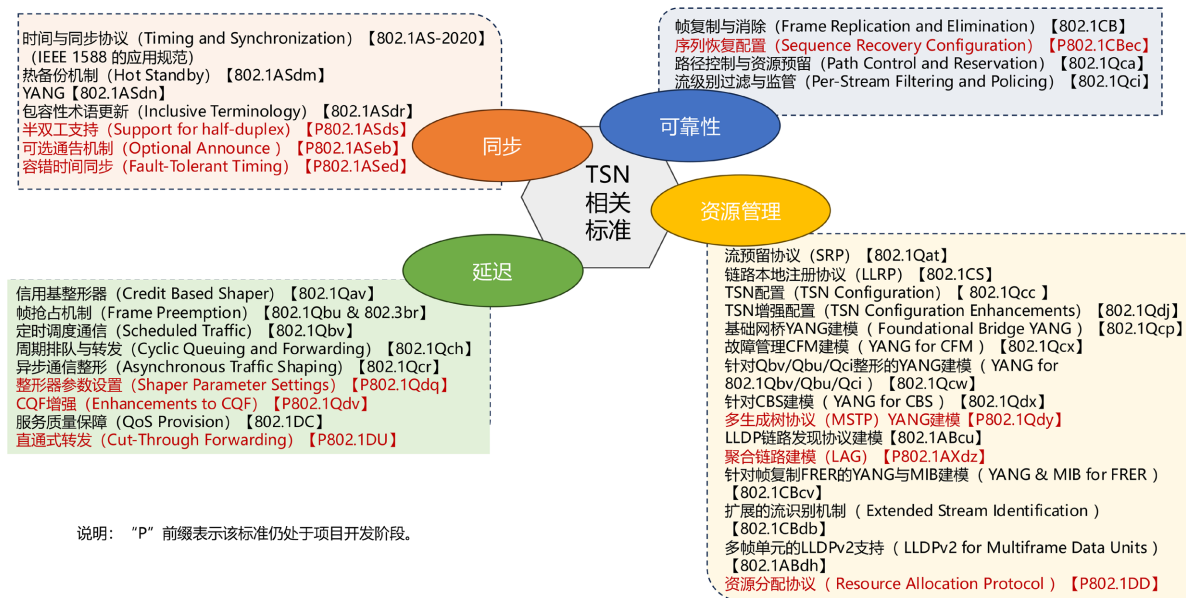


Figure 3. TSN standard architecture

图 3. TSN 标准架构图

此外, 针对 TSN 的研究, 为提升协议层面的系统性理解与工程适用性, Xue 等对 IEEE 802.1Qbv (时间感知整形器) 调度机制进行了全面综述与实验评估, 分析了其在多类工业场景中的可行性与性能差异, 为调度算法的实际部署提供了实证基础[7]。在调度机制优化方面, Muñoz 等提出了一种联合调度模型, 能够协同调度时间同步协议报文与关键工业流, 提升了链路资源利用效率与端到端传输的紧凑性[8]。Lopes 等则从系统架构层面探讨了 TSN 在智慧城市通信骨干网中的部署可行性, 面向自动驾驶、智能电网等应用场景提出了低时延、高可靠性的确定性通信保障机制[9] [10]。

低功耗广域网(Low Power Wide Area Network, LPWAN)是针对物联网中远距离、低功耗通信需求设计的网络层技术体系, 通过超长电池寿命(可达 10 年)和广域覆盖(城市超 2 公里)特性解决传统无线通信在物联网部署中的局限。该技术分为授权频谱(NB-IoT, LTE-M)与非授权频谱(LoRa, SigFox)两方面。其研究方面, Maurya 等系统性地梳理了机器学习与深度学习在资源分配、信道选择、拥塞控制与能效优化中的应用, 为 LPWAN 架构优化提供了思路[11]。Chinchilla 等通过构建混合整数规划(Mixed Integer Programming, MIP)模型, 对 LoRaWAN 网络的参数配置进行优化, 在保持服务公平性的前提下显著提升系统吞吐能力[12]。Al-Sammak 等则针对智能电表等终端设备的能耗问题, 提出了一种基于网络状态感知的自适应传输策略, 实现了动态发射控制与电池寿命延长的双重目标[13]。

在 WiFi 方面, 近期研究主要围绕如何提升基于 WiFi 的感知能力展开。Hoang 等通过引入去噪自动编码器(Denoising Autoencoder)提升了 WiFi 信号下的姿态识别鲁棒性, 显著增强了系统在复杂环境中的感知能力[14]。为解决 WiFi 感知应用中高质量训练样本稀缺的问题, Wang 等引入生成式人工智能(AIGC)方法, 构建合成样本以扩展数据集, 有效支撑模型泛化能力提升[15]。

在 RFID 与反向散射通信领域, 相关研究正不断拓展其面向智能化应用的能力。Nolan 等设计了“KeyStub”无源交互设备, 借助反向散射技术实现了低功耗人机交互, 为新型感知界面提供了可行路径[16]。Mattar 等进一步探索了以 WiFi 和 5G 信号替代专用激励源的技术路径, 提升了反向散射通信系统的可部署性与扩展性, 为广域智能感知提供新支撑[2]。

3. 国内研究现状

在 5G-A 与 6G 系统研究方面, 北京邮电大学的高伟东团队提出一种优化的下行链路寻呼机制, 在兼顾时延控制的同时实现了最高达 38.89%的节能效果[17]。北京邮电大学冯志勇团队专注于通信感知一体化设计, 在多参考点场景下显著提升了系统的感知精度[18]。此外, 中国诺基亚贝尔实验室的 Chao 等系统梳理了 5G-A 支持的环境物联网(Ambient IoT)相关国际标准制定进程[19]。

国内关于 TSN 的研究以应用为导向, 湖南大学谢国琪团队提出一种两阶段网关调度机制, 有效提升了 CAN 报文在 TSN 骨干网上的接收率(提升幅度为 4.3%~8.2%), 解决了异构总线融合的核心问题[20]。重庆邮电大学张磊团队设计并实现了支持国产 AUTBUS 协议的 TSN 转换网关, 具备 1.455 毫秒的低延迟转换能力, 为国产工业总线标准融入 TSN 生态提供关键支撑[21]。东北大学冯智伟等通过流选择、路由选择和队列分配等步骤实现高效的 TSN 循环队列转发整形器调度[22]。

在新一代国产短距通信协议方面, 南京理工大学冯李航团队将星闪协议部署于智能汽车避障测试平台, 实验证明其低延迟特性能够满足动态路径规划要求[23]。

在 LPWAN 领域, 北京大学刘云淮团队提出了 LoBee 跨协议通信机制, 打通了 LoRa 与 ZigBee 之间的物理层通信障碍, 实现远距离通信与高自组网能力的结合, 为多协议物联网协同提供了新范式[24]。

在提升 WiFi 智能感知方向, 国内研究在系统级方案与基础模型层面均取得了显著进展。北京大学张大庆团队引入物理层衍射模型, 提升了复杂多径环境下的识别精度[25]。此外, 张大庆团队还提出了利用波束成形反馈矩阵进行 WiFi 感知, 该方法通过充分利用 WiFi 芯片中现有的波束成形反馈信息, 实现了高精度的感知功能[26]。在人工智能与感知系统结合的研究中, 苏州大学的邓滔团队提出了面向 WiFi 的 ConSense 持续学习框架, 有效解决了 WiFi 感知模型在动态变化场景下面临的灾难性遗忘问题[27]。而在 WiFi 感知系统的隐私保护方面, 浙江大学卢立团队则揭示了基于 WiFi 感知系统的新型窃听漏洞, 为系统部署提供了隐私风险评估依据[28]。

在 RFID 智能感知应用方面, 天津大学刘秀龙团队提出了移动 RFID 机器人定位方法, 结合信号解缠绕与多天线优化算法, 提升了移动机器人定位精度并降低了因速度变化引起的测量误差[29]。

4. 国内外研究对比

在新一代信息通信技术发展进程中, 国内外研究在重点方向与技术路径上呈现出一定的差异性与互补性。

在 5G/6G 网络研究方向, 国际研究普遍聚焦于底层体系结构的变革与技术范式的重塑, 重点探索非 IP 通信协议、太赫兹通信与信息中心网络等前沿关键技术, 力图推动网络架构的深度重构。以支撑未来高性能、泛在智能网络的构建。相比之下, 国内研究则更加强调技术的工程化落地与系统集成能力提升, 聚焦于网络能效优化、通信与感知一体化设计等方向, 注重应用可行性与部署实效。

在时间敏感网络研究方向, 国际研究以协议机制优化与系统性能评估为主要突破点, 围绕时间同步流量调度、TSN 在 5G/6G 无线通信与智慧城市中的适配性开展研究, 具有较强的前瞻性与技术引领性。国内研究则聚焦工业场景中的异构网络融合与 TSN 部署可行性问题, 重点推进协议转换、调度策略设计及系统集成等技术, 强调落地部署与本地化适配。

在低功耗广域网研究方向, 国际研究侧重于引入人工智能方法对资源调度、能效管理与带宽利用进行深度优化; 而国内研究则在跨协议互通机制方面提出解决方案, 例如通过物理层跨技术通信手段实现 LoRa 与 ZigBee 间的双向直接通信, 提升了系统互操作能力与部署灵活性。

在短距离通信技术方向, 国际研究多基于对现有协议栈的优化扩展, 而国内则围绕“星闪”协议构建自主可控的短距通信体系, 形成与国际标准演进路径不同的发展路线, 重点服务于高可靠、低延迟需求下的工业制造与智能交通等应用场景。

在 WiFi 与 RFID 方面, 国内外研究内容都扩展至智能感知应用方面, 但国际研究更侧重于 AI 与新型应用范式结合的探索, 涵盖如基于 AIGC 的合成训练数据生成、无源交互式设备构建等方向; 国内研究则更加注重系统级功能完备性与实用性提升, 在 WiFi 感知建模优化、RFID 定位精度提升与隐私保护机制方面进行深入研究, 形成了较为系统的研究体系。

综上所述, 国际研究在架构重塑与未来网络范式创新方面具备引领优势, 国内研究则以工程实践为导向, 在融合应用与自主体系构建方面形成独特路径, 二者互为补充, 共同推动通信与感知系统的协同演进。

5. 发展趋势和展望

未来, 嵌入式网络将朝着低时延、高可靠、高带宽、高并发与异构融合等方向持续演进。

在低时延与高可靠方面, 未来嵌入式网络架构将深度嵌入轻量化 AI 模型, 以提升边缘节点的自适应处理能力与协同决策效率, 实现毫秒级响应与稳定性保障。与此同时, 为满足大规模部署下的能耗控制与资源调度需求, 嵌入式网络需在有限算力条件下实现动态优化, 确保端到端通信链路的稳定与敏捷。

面向高带宽与高并发场景, 6G 将以“通信 - 感知 - 计算”深度融合为核心, 突破太赫兹通信频段、非正交波形等物理层瓶颈, 显著提升系统容量与连接密度。未来 6G 不仅是带宽提升的技术飞跃, 更将构建支撑智能服务的综合平台, 实现从数据传输向智能处理的演进。

在异构通信方面, TSN 技术将向无线接入、城市级泛在覆盖等方向拓展, 支撑多制式、多协议设备之间的确定性通信需求。其应用正由工业控制与车载系统扩展至智慧交通、电网调度、航空航天等复杂场景, 构建具有端到端时延与抖动保障的网络服务能力。LPWAN 将强化与 ZigBee、5G、6G 等网络系统的互联互通, 构建统一的异构通信架构, 提升系统整体互操作性与扩展性。

在面对海量接入设备时, LPWAN 进一步引入机器学习与强化学习算法, 实现链路选择与能效控制的优化, 以支撑大规模无源节点的接入。针对高并发场景与低时延场景, 星闪作为新一代国产短距通信协议, 具备统一物理层架构与原生并发调度机制, 能够有效满足智能制造、自动驾驶、人机协作等对通信确定性与安全性要求极高的关键业务需求。其标准化进程与生态建设正稳步推进, 成为支撑本地高密度集部署的重要基础。

WiFi 与 RFID 在 AI 能力加持下, 其智能化应用正实现由传统识别向高精度、智能化方向的跃升。除基本身份识别功能外, 其应用已拓展至厘米级定位、行为分析、主动交互与安全防护, 在构建智能空间与智慧城市中发挥着关键作用, 助力实现通信 - 感知 - 计算的一体化融合。

基金项目

本研究得到了国家自然科学基金项目(No. 62202086), 辽宁省联合基金项目面上资助计划项目(No. 2023-MSBA-072)与中央高校基本科研业务费项目(N2416010)的支持。

参考文献

- [1] Siddiky, M.N.A., Rahman, M.E. and Uzzal, M.S. (2024) Beyond 5G: A Comprehensive Exploration of 6G Wireless

- Communication Technologies.
- [2] El Mattar, S. and Baghdad, A. (2025) Beyond Traditional Rfid: Unveiling the Potential of Wifi, 5G, Bluetooth, and Zigbee for Backscatter Systems. *Transactions on Emerging Telecommunications Technologies*, **36**, e70062. <https://doi.org/10.1002/ett.70062>
 - [3] P802.1ASds—Support for the IEEE Std 802.3 Clause 4 Media Access Control (MAC) Operating in Half-Duplex. <https://1.ieee802.org/tsn/802-1asds/>
 - [4] P802.1ASeb—Optional Use of Announce. <http://1.ieee802.org/tsn/802-1aseb/>
 - [5] P802.1Qdq—Shaper Parameter Settings for Bursty Traffic Requiring Bounded Latency. <https://1.ieee802.org/tsn/802-1qdq/>
 - [6] Time-Sensitive Networking (TSN) Profiles. <https://www.ieee802.org/1/files/public/docs2025/admin-tsn-summary-0325-v01.pdf>
 - [7] Xue, C., Zhang, T., Zhou, Y., Nixon, M., Loveless, A. and Han, S. (2025) A Survey and Experimental Study of Real-Time Scheduling Methods for 802.1qbv TSN Networks. *ACM Computing Surveys*, **58**, Article ID: 3736715. <https://doi.org/10.1145/3736715>
 - [8] Muñoz, P., Rodríguez-Martín, P., Caleyá-Sánchez, J., Prados-Garzón, J., Adamuz-Hinojosa, Ó. and Pablo Ameigeiras, a. (2025) Joint Scheduling of IEEE 802.1AS gPTP and Industrial Data Traffic in TSN-6G Networks. *IEEE Access*, **13**, 99842-99862. <https://doi.org/10.1109/access.2025.3577021>
 - [9] Lopes, R.E., Raposo, D., Martins, R. and Sargento, S. (2025) Time-Sensitive Smart Cities: Addressing Smart City Challenges with TSN. *IEEE Access*, **13**, 10459-10480. <https://doi.org/10.1109/access.2025.3528257>
 - [10] Egger, S., Dürr, F., Varga, B., De Andrade, M., Sharma, G.P., Sachs, J., et al. (2025) Wireless-Aware TSN Engineering: Implications for 5G and Upcoming 6G Networks. *IEEE Network*, **39**, 99-107. <https://doi.org/10.1109/mnet.2025.3556002>
 - [11] Maurya, P., Hazra, A., Kumari, P., Sørensen, T.B. and Das, S.K. (2025) A Comprehensive Survey of Data-Driven Solutions for LoRaWAN: Challenges and Future Directions. *ACM Transactions on Internet of Things*, **6**, 1-36. <https://doi.org/10.1145/3711953>
 - [12] Chinchilla-Romero, N., Prados-Garzon, J., Delgado-Ferro, F. and Navarro-Ortiz, J. (2025) An Optimization Model for Resource Allocation in Multitenant LoRaWAN Scenarios. *IEEE Internet of Things Journal*, **12**, 29713-29728. <https://doi.org/10.1109/jiot.2025.3569060>
 - [13] Al-Sammak, K.A., Al-Gburi, S.H., Marghescu, I., Drăgulescu, A.C., Marghescu, C., Martian, A., et al. (2025) Optimizing IoT Energy Efficiency: Real-Time Adaptive Algorithms for Smart Meters with LoRaWAN and NB-IoT. *Energies*, **18**, Article No. 987. <https://doi.org/10.3390/en18040987>
 - [14] Hoang Nguyen, X., Nguyen, V., Luu, Q., Dinh Gian, T. and Shin, O. (2025) Robust Wifi Sensing-Based Human Pose Estimation Using Denoising Autoencoder and CNN with Dynamic Subcarrier Attention. *IEEE Internet of Things Journal*, **12**, 17066-17079. <https://doi.org/10.1109/jiot.2025.3535156>
 - [15] Wang, Z. and Mao, S. (2025) AIGC for Wireless Sensing: Diffusion-Empowered Human Activity Sensing. *IEEE Transactions on Cognitive Communications and Networking*, **11**, 657-671. <https://doi.org/10.1109/tccn.2025.3525588>
 - [16] Nolan, J., Qian, K. and Zhang, X. (2023) KeyStub: A Passive RFID-Based Keypad Interface Using Resonant Stubs. *Proceedings of the ACM on Interactive, Mobile, Wearable and Ubiquitous Technologies*, **7**, 1-23. <https://doi.org/10.1145/3631442>
 - [17] Ma, W., Gao, W., Liu, J., Zhang, K., Zhao, X., Cui, B., et al. (2025) Research on Paging Enhancements for 5G-A Downlink Transmission Energy Saving. *Digital Communications and Networks*, **11**, 818-828. <https://doi.org/10.1016/j.dcan.2024.07.005>
 - [18] Wei, Z., Li, F., Liu, H., Chen, X., Wu, H., Han, K., et al. (2024) Multiple Reference Signals Collaborative Sensing for Integrated Sensing and Communication System towards 5G-A and 6g. *IEEE Transactions on Vehicular Technology*, **73**, 15185-15199. <https://doi.org/10.1109/tvt.2024.3410352>
 - [19] Chao, H., Tao, T., Wang, Y., Liu, J., Meng, Y., Liu, Y., et al. (2025) Architecture Evolution for Ambient Power-Enabled IoT in 5G-A and Future Cellular Networks. *IEEE Internet of Things Magazine*, **8**, 48-54. <https://doi.org/10.1109/iotm.001.2400128>
 - [20] Yan, W., Fu, B., Huang, J., Lu, R., Li, R. and Xie, G. (2024) A Conflict-Free CAN-to-TSN Scheduler for CAN-TSN Gateway. *Journal of Systems Architecture*, **153**, Article ID: 103188. <https://doi.org/10.1016/j.sysarc.2024.103188>
 - [21] Rong, Z., Wei, M., Dan, J. and Zhang, L. (2025) Design and Implementation of Industrial Gateway for AUTBUS and TSN Protocol Conversion. 2025 *International Conference on Information Networking (ICOIN)*, Chiang Mai, 15-17 January 2025, 517-522. <https://doi.org/10.1109/icoi63865.2025.10993179>
 - [22] Feng, Z., Dong, W., Gao, S., Lin, Y., Jin, X. and Deng, Q. (2025) An Efficient Heuristic CQF Scheduling in Time-

- Sensitive Networking. *IEEE Transactions on Industrial Informatics*, **21**, 5213-5223. <https://doi.org/10.1109/tii.2025.3552701>
- [23] Yi, H., Lin, R., Wang, H., Wang, Y., Ying, C., Wang, D., *et al.* (2025) Dynamic Obstacle Avoidance with Enhanced Social Force Model and DWA Algorithm Using Sparklink. *Sensors*, **25**, Article No. 992. <https://doi.org/10.3390/s25040992>
- [24] Gao, D., Wang, H., Chen, Y., Ye, Q., Wang, W., Guo, X., *et al.* (2025) Lobee: Bidirectional Communication between LoRa and ZigBee Based on Physical-Layer CTC. *IEEE Transactions on Wireless Communications*, **24**, 6857-6868. <https://doi.org/10.1109/twc.2025.3556552>
- [25] Wang, X., Yu, A., Niu, K., Shi, W., Wang, J., Yao, Z., *et al.* (2024) Understanding the Diffraction Model in Static Multipath-Rich Environments for Wifi Sensing System Design. *IEEE Transactions on Mobile Computing*, **23**, 10393-10410. <https://doi.org/10.1109/tmc.2024.3377708>
- [26] Yi, E., Wu, D., Xiong, J., *et al.* (2024) BFMSense: WiFi Sensing Using Beamforming Feedback Matrix. 21st *USENIX Symposium on Networked Systems Design and Implementation (NSDI 24)*, Santa Clara, 16-18 April 2024, 1697-1712.
- [27] Li, R., Deng, T., Feng, S., Sun, M. and Jia, J. (2025) Consense: Continually Sensing Human Activity with Wifi via Growing and Picking. *Proceedings of the AAAI Conference on Artificial Intelligence*, **39**, 14292-14300. <https://doi.org/10.1609/aaai.v39i13.33565>
- [28] Lu, L., Chen, M., Yu, J., Ba, Z., Lin, F., Han, J., *et al.* (2024) An Imperceptible Eavesdropping Attack on Wifi Sensing Systems. *IEEE/ACM Transactions on Networking*, **32**, 4009-4024. <https://doi.org/10.1109/tnet.2024.3403839>
- [29] Zhang, J., Liu, X., Chen, S., Tong, X., Deng, Z., Gu, T., *et al.* (2024) Toward Robust RFID Localization via Mobile Robot. *IEEE/ACM Transactions on Networking*, **32**, 2904-2919. <https://doi.org/10.1109/tnet.2024.3373770>

嵌入式系统中间件和软总线综述

杨鹏飞, 郑天洋, 郭 恒, 王 泉

西安电子科技大学计算机科学与技术学院, 陕西 西安

收稿日期: 2025年10月28日; 录用日期: 2025年11月20日; 发布日期: 2025年12月1日

摘 要

嵌入式中间件与软总线作为现代分布式系统的核心基础设施, 对于降低系统开发复杂度、实现异构环境互操作至关重要。文章系统梳理了应用服务器、远程过程调用(RPC)、消息中间件、容器编排平台等主流中间件以及新兴软总线技术的发展脉络。通过从系统完整性、环境适配性、对分布式架构与大模型等新兴技术的支撑性三个维度进行深入对比, 揭示了国内外技术方案的差异化格局。研究发现, 国际中间件凭借成熟的生态与标准化设计在系统完整性上具备优势, 而国内中间件在国产化浪潮驱动下, 依托云原生架构实现了跨越式发展, 尤其在服务治理、本土软硬件生态适配及新兴场景应用方面形成了独特竞争力。展望未来, 嵌入式中间件与软总线技术正朝着系统完整性更高、适配性更强, 并与云原生、人工智能等前沿技术深度融合的方向演进, 将成为构筑智能制造、智慧城市等未来应用场景的泛在连接与智能协同的核心技术底座。

关键词

嵌入式软件, 软件中间件, 软总线, 分布式系统

An Overview of Middleware and Soft Bus in Embedded Systems

Pengfei Yang, Tianyang Zheng, Heng Guo, Quan Wang

School of Computer Science and Technology, Xidian University, Xi'an Shaanxi

Received: October 28, 2025; accepted: November 20, 2025; published: December 1, 2025

Abstract

As the core infrastructure of modern distributed systems, embedded middleware and soft bus play a crucial role in reducing system development complexity and enabling interoperability across heterogeneous environments. This paper systematically reviews the development of mainstream middleware technologies—including application servers, remote procedure calls (RPC), message-

ented middleware, and container orchestration platforms—as well as emerging soft bus technologies. From three perspectives—system integrity, environmental adaptability, and support for emerging technologies such as distributed architectures and large-scale models—this study conducts an in-depth comparative analysis to reveal the differentiated patterns between domestic and international solutions. The findings indicate that international middleware demonstrates advantages in system integrity due to its mature ecosystems and standardized design, whereas domestic middleware, driven by the wave of localization, has achieved leapfrog progress based on cloud-native architectures. In particular, it exhibits unique competitiveness in service governance, compatibility with local software and hardware ecosystems, and application to emerging scenarios. Looking ahead, embedded middleware and soft bus technologies are evolving toward higher system integrity, stronger adaptability, and deeper integration with frontier technologies such as cloud-native computing and artificial intelligence. They are expected to become the foundational enablers of ubiquitous connectivity and intelligent collaboration for future applications, including intelligent manufacturing and smart cities.

Keywords

Embedded Software, Software Middleware, Soft Bus, Distributed Systems

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

软件中间件是部署于操作系统与应用软件之间的独立软件实体，通过协议抽象与接口封装机制，实现跨平台通信服务的透明化供给，支持事务处理、消息路由、资源调度等关键功能，从而确保分布式应用在异构环境中的互操作性与协同效率。作为现代分布式系统的核心基础设施，其本质功能在于构建底层异构计算环境与上层应用逻辑的标准化连接纽带，降低复杂系统的开发复杂度。软件中间件是云计算、大数据等新兴技术栈的重要支撑底座。

随着边缘计算、物联网等技术的普及，中间件范畴也从早期的消息队列(如 Rabbit Message Queue RabbitMQ 实现的亿级消息吞吐量)、企业服务总线(Enterprise Service Bus ESB)等传统形态，拓展至 API 网关、服务网格、数据集成工具等融合架构。当设备互联规模突破物理总线连接瓶颈，异构协议的动态适配需求催生了中间件技术创新形态——软总线技术。软总线技术通过协议抽象层将蓝牙、Wi-Fi、NFC 等异构通信协议映射为统一的编程接口，实现跨设备数据传输的透明化与高效化。软总线通过软件定义的通信机制重构了设备交互模型，反映了分布式系统从分层架构向扁平化通信架构的范式转换。

同时，嵌入式中间件与软总线的技术融合也标志着分布式系统架构的深度进化，在继承标准化接口与资源调度核心能力的基础上，通过软件定义的连接机制突破物理边界限制，为数字孪生、元宇宙等新兴应用场景提供了泛在连接的技术底座。

2. 国内外研究现状

目前，国际上关于中间件和软总线的研究主要集中在工程应用领域，开发针对特定应用场景的中间件产品，如应用服务器、远程过程调用(RPC)框架以及容器编排平台等，或基于某一种中间件在特定场景中的定制化需求，优化其性能和适用性。部分研究也探讨了中间件的设计原则、关键技术及其面临的挑战[1]。

2.1. 中间件国内外研究现状

中间件是位于操作系统和应用程序之间的软件层，负责协调不同系统、服务或组件之间的通信和数据交换。它屏蔽底层技术细节(如网络协议、硬件差异等)，使开发者能专注于业务逻辑，而无需处理复杂的底层交互，嵌入式中间件技术的常见架构如图1所示，各部分功能及国内外研究现状详细对比如下。

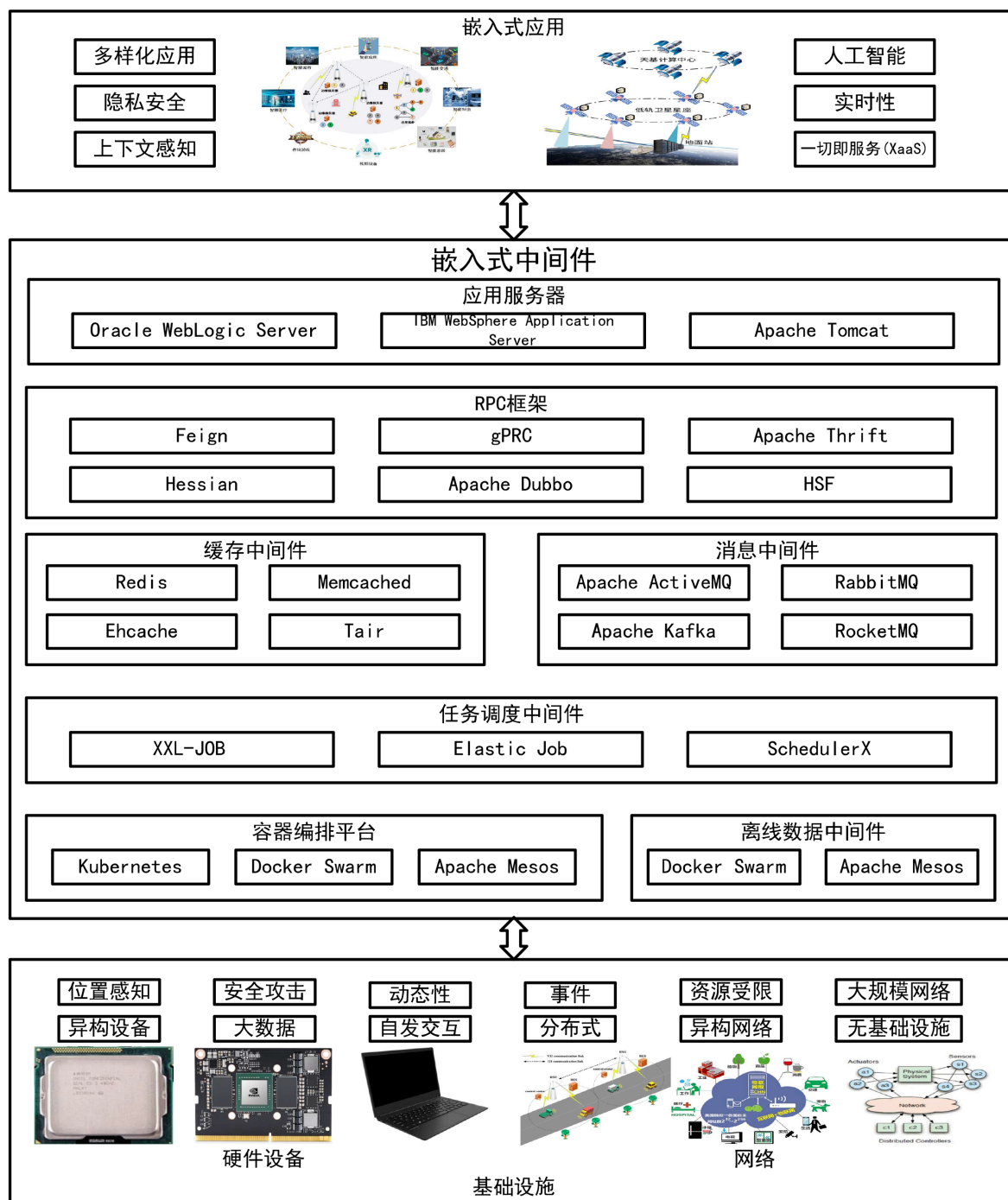


Figure 1. Middleware architecture diagram

图 1. 中间件架构图

2.1.1. 应用服务器

应用服务器是一种关键的软件框架，旨在为企业级应用程序的开发、部署、运行与管理提供统一的运行环境和系统支持。其主要功能不仅限于支持应用程序的基本执行逻辑，更在此基础上集成了诸如安全机制、事务处理、资源调度与管理、负载均衡、会话维护以及连接池管理等多项高级服务。这些服务的整合显著提高了企业应用的可扩展性、可靠性与安全性，从而有效支撑复杂业务逻辑的稳定运行[2]。

当前主流的 Java 企业级应用服务器主要包括 Oracle WebLogic Server、IBM WebSphere Application Server、Apache Tomcat 以及 WildFly 等。这些服务器在企业应用开发、部署与管理中起到了核心支撑作用。Oracle WebLogic Server 作为一种强健、成熟且可扩展的平台，实现了对 Java EE 与 Jakarta EE 标准的全面支持，广泛应用于本地与云环境下的大型分布式系统开发，具备良好的互操作性与安全性。IBM WebSphere Application Server 则依托消息队列机制实现跨平台与跨网络的高效通信，能够屏蔽底层操作系统差异，保障了企业级应用在异构环境下的稳定运行[2]。

此外，Apache Tomcat 作为轻量级的开源 Web 容器，广泛支持 Jakarta 相关规范，因其稳定性与灵活性，在中小型 Web 应用开发与调试中得到广泛应用，成为开发者常用的本地部署与测试工具。WildFly (原 JBoss AS)则作为一款基于 SOA (Society of Actuaries)架构的开源企业级中间件，支持复杂 Web 应用与服务的构建与集成。总体而言，这些主流 Java 应用服务器在架构能力、平台兼容性与部署灵活性方面各具优势，构成了当前 Java 企业应用支撑平台的多元生态。

在国内信息化建设持续推进的背景下，国产中间件在核心系统中的应用逐渐增多。中国移动在其“One OSS 2.0”战略框架下建设的“综合网络资源管理系统”一期工程中，采用东方通中间件公司的 TongWeb 应用服务器作为底层支撑平台。TongWeb 以其高可靠性、可扩展的集群能力以及对标准 API 的支持，为系统提供了包括应用开发、部署与管理在内的全面支撑服务，实现了对全专业网络资源的精细化管理与共享[3]。

2.1.2. 远程过程调用(RPC)

远程过程调用(Remote Procedure Call, RPC)是一种基于请求-响应模型的进程间通信(IPC)技术，使分布式系统中不同节点的程序能够通过网络调用彼此提供的函数或过程[4] [5]。RPC 技术通过封装网络传输、序列化、路由等底层实现细节，为开发者提供了一种类似于本地过程调用的抽象接口，从而实现跨进程或跨节点之间高效且透明的服务调用。一个完整的 RPC 调用流程通常包含三类主体角色：服务提供者、服务消费者以及服务注册中心。其中，服务提供者负责具体业务逻辑的实现及远程接口的暴露；服务消费者则发起远程调用请求；而服务注册中心则起到协调服务注册与发现的作用。

在国外框架中，Open (基于 Netflix Feign)面向 Spring Cloud 微服务间的通信，依托注解驱动实现声明式服务调用，并通过 Ribbon 实现客户端负载均衡[5]，广泛应用于电商及金融领域的微服务拆分，支撑高并发业务模块的协同运作；gRPC 采用 Protobuf 编码与 HTTP/2 协议，在多语言、高并发与跨数据中心场景下表现出低延迟、高吞吐的优势，同时还支持负载均衡、健康检查等可插拔功能[6]；Apache Thrift 提供了统一的接口定义语言和多语言代码生成能力，适合构建跨语言的分布式系统[7]，被 Facebook、Uber 等企业用于构建跨语言服务网格；Hessian 则以二进制协议为核心，通过紧凑数据结构实现高效传输，适用于对网络带宽和解析性能要求较高的应用[7]，常用于移动应用后端及物联网设备的数据交互场景。

在国内框架中，阿里巴巴的 HSF 着重系统高性能与稳定性，优化网络传输与请求处理效率，支撑企业级大规模服务调用[8]，支撑了淘宝、天猫等平台历年“双十一”的平稳运行，已被整合至 Dubbo 3 中；SOFARPC 由蚂蚁金服开源，集成链路追踪、流量调度与故障剔除等能力，增强系统在复杂分布式架构下的调度与治理效率，具有良好的可伸缩性与可靠性[9]，用于蚂蚁集团支付核心及风控系统的事务处理。

综合对比, Open (Feign)类框架面向 Spring Cloud 生态, 适合快速构建 Java 微服务系统, 但通信效率较低且不支持多语言; gRPC 具备跨语言、高性能优势, 适用于多语言分布式系统, 但缺乏内建服务治理功能, 接口变更需重新定义与编译[10][11]。相比之下, 国内框架更侧重服务治理与系统集成, 如 SOFARPC 支持服务注册、负载均衡与容错控制, 适配复杂业务场景, HSF 则以高性能和稳定性支撑大规模企业系统。

2.1.3. 消息中间件

消息中间件(分布式消息系统)是分布式系统之间进行消息传递的重要组件, 它利用高效、可靠的消息传递机制进行平台无关的数据交流, 并基于数据通信进行分布式系统的集成。目前, 消息中间件已经在企业中广泛应用, 有些企业还自主研发更加适合自身应用场景的消息中间件。

国外主流消息中间件中, Apache ActiveMQ (Apache 开源多协议 Java 消息代理)凭借对 STOMP/AMQP 等协议的支持及多语言客户端兼容性(JavaScript/C++/Python 等), 成为传统企业异构系统集成的核心工具[11]; RabbitMQ 以插件化架构和高可靠性著称, 广泛应用于中小规模异步通信场景(如电商订单队列)[12]; Apache Kafka 通过高吞吐(百万级 TPS)和持久化日志设计, 主导实时数据管道与流分析领域(如用户行为追踪)[13]。

国内消息中间件领域, RocketMQ 以低延迟、高并发(双十一峰值 1.5 万亿/天)及原生事务消息为核心优势, 深度整合阿里云生态, 支撑电商、支付等核心业务场景, 成为金融级实时数据处理的首选方案。

国内外消息中间件技术广泛应用于地震监测、电力安全、金融交易等领域, 通过支持多协议通信(如 ActiveMQ)、高可靠异步调度(如 RabbitMQ)、海量实时流处理(如 Kafka)及低延迟事务消息(如 RocketMQ), 有效解决了异构系统集成、分布式数据协同、高并发业务容灾等核心问题。典型场景包括地震台网跨模块通信、煤矿供电实时监测、电商峰值交易保障等, 技术趋势聚焦协议扩展性、吞吐量优化及云原生深度集成, 为工业数字化转型与核心业务高可用提供底层支撑。

2.1.4. 缓存中间件

缓存中间件是一种在应用程序和数据库之间的组件, 用于存储和提供经常使用的数据, 以加快数据访问速度。它可以有效地减轻数据库的负载, 提高应用程序的性能和响应速度。

国外主流缓存服务中, Redis 基于超大规模架构与现代化缓存技术, 提供实时数据处理能力, 支持复杂数据结构(如哈希/流计算), 广泛应用于社交网络实时推荐、金融交易缓存加速等场景, 其多模块扩展能力(Redis Modules)助力企业实现全栈缓存控制[14]; Memcached 以极简设计为核心, 专注于快速存取小块数据(如数据库查询结果、API 响应), 凭借无持久化特性与多线程架构, 成为高并发 Web 应用(如电商页面缓存)中缓解数据库压力的首选方案。

在特定技术生态领域, Ehcache 通过进程内/进程外混合部署模式, 实现从单机到 TB 级分布式缓存的平滑扩展, 深度整合 Spring、Hibernate 等 Java 主流框架, 为企业级应用提供透明化缓存集成(如数据库查询结果复用), 在电信计费系统、医疗影像处理等场景中显著降低系统延迟[15]。

国内阿里云的云数据库 Tair 是一种全托管、兼容 Redis 协议且具备超高性能的数据库服务, 能够保证亚毫秒级的稳定时延, 为应用程序起到加速作用。其中, Redis 开源版内核基于开源代码进行了强化, 而 Tair 则在此基础上进一步增加了大量的企业级特性, 能够覆盖 Redis 开源版难以应对的场景, 提供稳定可靠的服务。

2.1.5. 任务调度中间件

任务调度中间件作为分布式环境中管理和协调任务执行的关键技术, 通过提供定时触发、任务管理和自动化作业处理能力, 显著提升了系统的效率和可靠性。

以轻量级设计著称的 XXL-JOB [16]是国内颇具影响力的分布式任务调度平台，其以开发迅速、学习简单、开箱即用为核心优势，已被多家企业成功应用于生产环境。而 ElasticJob [17]则专注于互联网场景，通过弹性调度、资源管控和作业治理功能，构建了支持多元化作业生态的分布式解决方案，其统一的作业 API 设计实现了“一次开发，随处部署”的便利性。

国内行业实践中，阿里巴巴自主研发的 SchedulerX 展现了较强的技术整合能力。该平台基于 Akka 架构，不仅兼容 XXL-JOB、ElasticJob 等主流开源框架，还支持 K8s Job 和 Spring Schedule，提供从 Cron 定时任务到分布式数据处理的全场景支持，其高可用性、可视化运维和低延时特性满足了企业级复杂需求。

当前，任务调度中间件正持续演进以适应日益复杂的企业应用场景，在提升系统自动化水平、增强分布式任务处理能力以及支持大规模系统部署等方面发挥着关键作用。各技术方案通过差异化定位和功能创新，共同推动着分布式任务调度领域的快速发展。

2.1.6. 离线数据中间件

离线数据中间件是用于处理大规模数据集的框架和技术集合，它支持通过简单的编程模型在计算机集群上分布式处理大型数据集，旨在提高数据处理效率、增强系统的容错能力，并简化大数据应用的开发与维护。

在开源生态中，Apache Hadoop [18]是最具代表性的分布式处理框架之一。其采用 MapReduce 编程模型，能够在由数千台普通服务器组成的集群上实现数据的并行处理。Hadoop 的核心设计理念是通过软件层面的容错机制来应对硬件不可靠性，使得整个系统能够在节点故障时仍保持服务可用性。

随着实时计算需求的增长，Apache Spark [19]作为新一代统一分析引擎应运而生。与 Hadoop 的批处理模式不同，Spark 通过内存计算和优化的执行引擎显著提升了处理性能。它提供了多语言支持 (Java/Scala/Python/R) 和丰富的生态组件，包括 Spark SQL 用于结构化数据处理、MLlib 支持机器学习任务、GraphX 处理图计算，以及结构化流 (Structured Streaming) 实现实时分析能力。这些特性使 Spark 成为当前大数据处理领域的主流选择。

离线数据中间件的持续演进正推动着大数据处理模式从批处理向流批一体化发展，同时不断提升计算效率、资源利用率和开发便捷性，以适应日益复杂的业务分析场景。

2.1.7. 容器编排平台

容器编排平台是用于自动化管理、调度和协调容器化应用的技术工具，通过整合资源调度、服务发现、弹性伸缩、故障恢复等功能实现大规模分布式系统的高效运维，其核心价值在于将复杂的容器集群管理抽象为标准化流程，支持微服务架构的快速迭代与跨环境部署，显著提升资源利用率和应用稳定性 [20]。

国外主流容器编排平台中，Kubernetes (K8s) 作为 Google 开源、CNCF 维护的核心工具，具备自动化部署、服务发现、负载均衡及自愈能力，支持跨云环境弹性扩展，广泛应用于云原生应用开发，适用于大规模微服务架构与持续交付场景；Docker Swarm 作为 Docker 官方解决方案，与 Docker 生态深度集成，以简单易用和高可用性特点，成为中小型企业快速搭建容器集群的首选 [20]；Apache Mesos 作为开源集群管理平台，通过资源抽象层支持多框架调度，适用于大数据处理、机器学习等复杂场景 [20]；Nomad 由 HashiCorp 开发，支持多容器格式与灵活调度策略，满足跨数据中心部署的简单可靠需求 [20] [21]；OpenShift 基于 Kubernetes 构建企业级平台，集成 CI/CD、微服务治理等功能，适配复杂企业架构 [21] [22]；Rancher 作为开源管理平台，通过直观界面与插件生态实现多 Kubernetes 集群统一纳管，降低中小型企业运维成本 [22]。

国内容器编排平台以阿里云容器服务(ACK)和腾讯云容器服务(TKE)为代表,前者基于 Kubernetes 增强,整合阿里云基础设施,提供高性能网络与存储方案,广泛应用于电商、金融等领域,支撑大规模分布式应用部署[22];后者同样基于 Kubernetes,支持多存储与网络解决方案,适用于游戏、视频等行业的云原生应用快速开发[22]。

从应用实践看, Kubernetes 在金融行业的天弘基金亿级用户交易系统中实现每秒数千笔并发交易,清算时间缩短至 1 小时内,保障高并发场景下的稳定性[23];阿里云 ACK 助力西门子部署物联网操作系统 MindSphere,通过微服务架构与 DevOps 自动化提升全球设备数据实时分析效率[23];腾讯云 TKE 在视频云离线转码服务中提升 CPU 峰值利用率至 80%,动态扩缩容使任务处理效率提升 30% [24]。

国内外容器编排平台的应用覆盖金融、电商、工业、视频等多领域,有效解决弹性伸缩、高可用性、复杂网络管理等问题。国外平台中, Kubernetes 主导云原生生态, Docker Swarm 和 Rancher 适配中小场景, Mesos 与 Nomad 聚焦大数据;国内 ACK 与 TKE 结合本土基础设施优势,在企业级容器化转型中发挥关键作用。技术趋势上,混合云管理与 Serverless 化成为方向,如 Rancher 与 ACK 支持跨云调度, XTransfer 基于 Knative 实现算法模型按需扩容以节省资源成本[23] [24]。

2.1.8. 其他中间件

除了上述常见的分类之外,数据库中间件、API 网关中间件、身份认证与授权中间件以及分布式文件系统中间件也属于重要的中间件类别。

在分布式计算与云计算架构快速发展的背景下,中间件技术在保障系统性能、可扩展性和安全性方面发挥着关键作用。数据库中间件为应用程序提供统一的数据库访问接口,支持多数据库连接管理、负载均衡、读写分离和事务控制,广泛应用于电商、金融等高并发场景。阿里巴巴推出的 TDDL 中间件被成功应用于其电商平台[25],有效提升了数据库的性能与容错能力。API 网关中间件则在微服务架构中扮演着通信枢纽的角色,负责 API 请求的统一管理,支持认证、流量控制和负载均衡,简化了服务间的交互流程。Amazon 的 API Gateway 广泛应用于 AWS 平台[25],帮助开发者高效管理分布式系统中的 API 流量。身份认证与授权中间件提供用户身份验证与权限管理能力,采用 OAuth、JWT 等标准协议保障系统安全,腾讯云身份认证(CAM)为腾讯云平台提供了灵活可靠的访问控制方案[26]。分布式文件系统中间件用于跨节点高效存储和管理数据,提升数据可靠性与可用性,典型如 Facebook 的 Hadoop HDFS 系统[27],为社交平台的海量数据存储提供了坚实基础。各类中间件技术的广泛应用,不仅解决了分布式环境下的诸多技术难题,也为企业系统的高效运行与持续扩展提供了强有力的支撑。

2.2. 软总线技术现状

软总线(Soft Bus)是一种通过软件协议和虚拟化技术实现的分布式通信框架,旨在连接异构设备、系统或服务,实现跨平台、跨网络的数据传输与资源共享,软总线架构图如图 2 所示。其核心在于通过标准化的通信协议(如 MQTT、HTTP)、消息队列和异步通信机制,模拟硬件总线的功能,屏蔽底层硬件差异,支持设备间的自动发现、组网与高效协同[28] [29]。例如,在嵌入式开发环境中,基于异步通信的软总线框架能够集成远程调试工具、动态加载模块和系统监控工具,显著提升开发工具链的协作效率[29]。

国内软总线技术主要围绕物联网和分布式系统展开。华为的鸿蒙分布式软总线是其操作系统的核心组件,支持手机、平板、智能穿戴等设备通过“碰一碰”实现无感互联与跨设备协作,如文件传输和屏幕投射。此外, TCL 开发的分布式软总线服务调用框架基于安卓系统扩展,通过跨进程通信模块实现远程设备服务调用,提升资源共享能力。在工业领域,嵌入式 Linux 软总线解决方案通过虚拟化总线管理器和 TCP/IP 协议,支持电力配网系统中的多节点通信与冗余管理,优化工业自动化效率。国内软总线技术的典型应用包括智能家居、工业自动化与企业级服务。华为鸿蒙的分布式软总线已深入消费电子领域,

支持智能家居设备间的无缝协作(如会议投屏、车机互联)[30]。在工业场景中,嵌入式软总线框架被集成到电力配网系统中,通过实时通信与冗余管理提升电网监控效率[31]。此外,基于软总线的文件管理系统在多设备协同办公场景中发挥作用,例如跨设备文件共享与缓存管理,显著降低企业内部的协作复杂度。

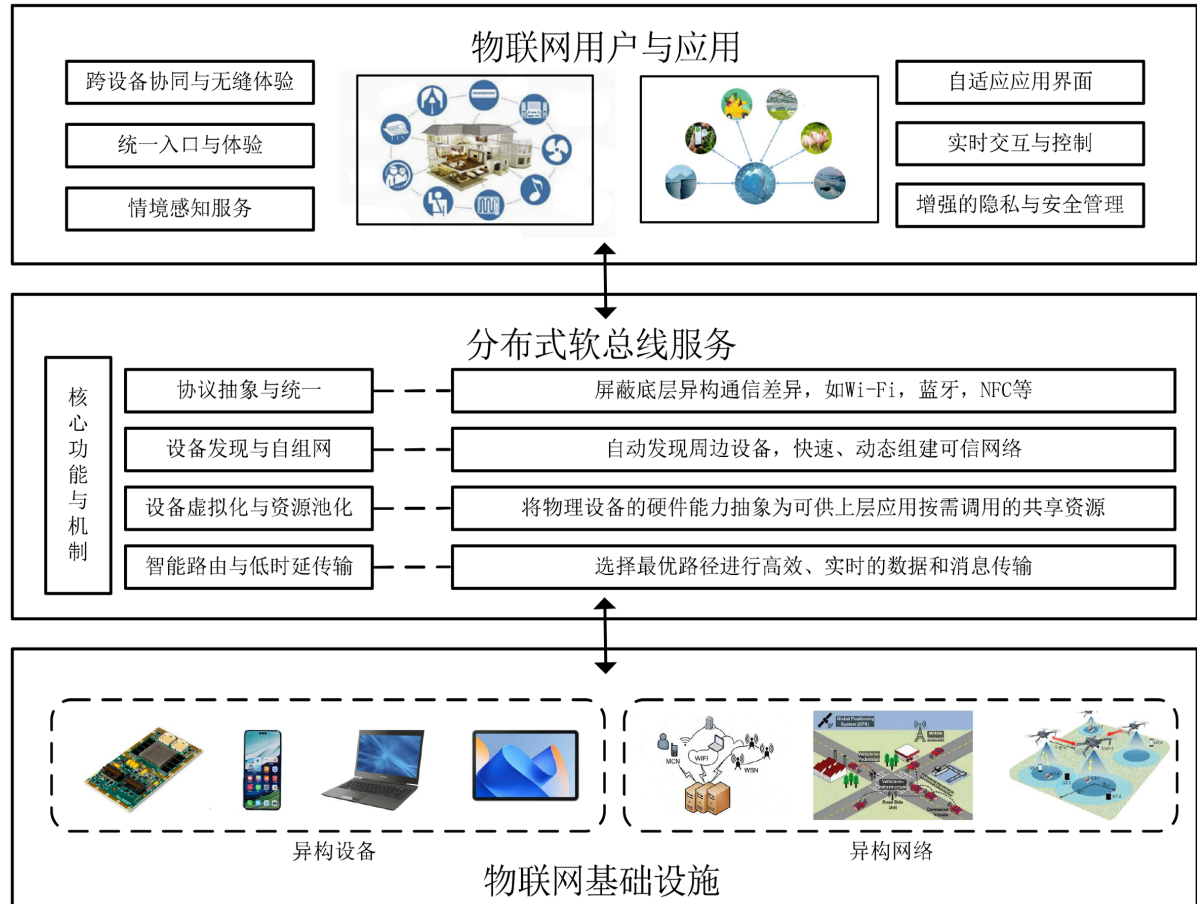


Figure 2. Diagram of soft bus architecture

图 2. 软总线架构图

在分布式计算与跨平台通信领域, Meersman 等人[30]系统探讨了分布式计算中的软件总线技术, 提出了面向语义互联网系统的多维度架构模型, 涵盖上下文感知移动系统(CAMS)、社区信息学(COMINF)和本体内容管理(OnToContent)等关键技术, 为跨平台通信提供了理论框架。Sijtema 等学者[32]通过 Neopost 公司的工业案例验证了形式化工程方法的有效性, 他们采用 mCRL2 建模语言构建软件总线协议, 结合 JTorX 工具实现自动化测试, 证明形式化方法可将开发周期缩短 17%且显著提升错误检测效率, 特别是对时序敏感型缺陷的识别能力优于传统测试方法。

在架构设计方面, Liu [33]基于 CORBA 标准提出分布式计算机软件总线模型, 通过标准化接口定义语言(IDL)实现跨语言组件通信, 其分层架构包含可移植对象适配器(POA)和 IIOP 协议, 实验显示该模型在 Java 与 C++混合编程环境下实现高效数据交互。Purtilo [34]开发的 POLYLITH 系统创新性地引入模块互连语言(MIL), 通过软件总线抽象解耦功能实现与接口需求, 支持异构环境中 LISP、Ada 与 C 组件的无缝集成, 其核心突破在于将通信协议选择与功能组件分离, 使系统重构成本降低 42%。

面向未来技术演进, Cheng [35]提出软系统总线(SSB)作为可重构反应式系统的核心架构, 该设计采用环形总线结构集成自测量(Me)、自监控(Mo)等永久控制组件, 通过数据/指令保存机制实现非停机维护, 理论验证表明 SSB 可使系统可用性提升至 99.999%。Xu 与 Shen [36]基于组件技术对比传统开发模式, 构建的软件总线开发框架将代码复用率提高 68%, 其敏捷开发流程通过标准化插件接口实现功能模块的“即插即用”, 实验数据显示目标处理能力达到 100 批次/秒, 响应时间控制在 8 ms 以内。

国外软总线技术主要通过企业服务总线(ESB)产品体现。MuleSoft Anypoint Platform 以其灵活的集成框架和可视化工具著称, 支持多种传输协议。其基于 Java 的轻量级 ESB 和集成平台允许开发者快速连接应用程序并实现数据交换。其他产品如 IBM Integration Bus 和 Oracle Service Bus 在大型企业中广泛用于异构系统集成, 提供强大的协议转换和消息路由能力。国外软总线技术主要应用于工业物联网与机器人系统。例如, OPC UA 与 PROFIBUS 协议在工厂自动化中集成 PLC、传感器与云端系统, 支持实时数据采集与生产流程优化。ROS 2 在自动驾驶和工业机器人中协调多传感器与执行器, 确保实时决策的可靠性。在云计算领域, Apache Kafka 作为数据总线支持智能城市中的交通监控系统, 实现云边协同的大规模数据流处理。此外, 软件定义广域网(SD-WAN)技术通过虚拟化网络管理, 优化了跨区域企业的资源调度与服务效率。

2.3. 性能测试与评价

在中间件性能评估与测试方面, Sachs 等人提出了基于供应链管理场景的 SPECjms2007 基准测试框架[37], 用于评估消息导向中间件(MOM)的吞吐量与可扩展性, 并通过 BEA WebLogic 案例验证了其在点对点(P2P)和发布/订阅(Pub/Sub)模式下的性能差异。Gokhale 和 Schmidt 对比了 CORBA、RPC 与底层 Socket 在 ATM 网络中的性能[38], 发现底层通信机制(如 C/C++)的吞吐量比 CORBA 高 25%~67%, 揭示了序列化与内存管理对高速网络的性能瓶颈。

在物联网中间件架构与评估方面, da Cruz 等人系统评估了物联网中间件性能指标[39], 提出安全性(如设备认证)与协议支持(MQTT/CoAP)的定性标准, 并验证 Sitewhere 在吞吐量上的优势。Patro 等人对比了消息中间件(如 RabbitMQ、YAMI4)在工业控制场景的性能[40], 指出 YAMI4 因低延迟(<10 ms)和高吞吐量(10 k msg/s)成为监控系统的优选方案。Cruz 等人进一步提出物联网中间件通用模型[39], 通过统一 API 支持多协议自适应选择, 解决了异构设备互操作性问题。

在服务导向与自适应中间件设计方面, Al-Jaroodi 和 Mohamed 综述了服务导向中间件(SOM) [41], 强调其在跨平台服务组合中的作用, 但指出动态 QoS 保障仍是挑战。Zhang 和 Jacobsen 通过面向方面编程(AOP)重构中间件架构[42], 将横切关注点(如日志、安全)模块化, 使代码复杂度降低 30%且性能开销可忽略。García Valls 等人提出实时中间件 iLAND 与 DREQUIEMI [43], 支持分布式任务动态重配置, 在毫秒级延迟约束下实现服务切换与资源调度。

在中间件在实时系统优化方面, Zhang 等人开发了 ControlWare 中间件[44], 基于控制理论实现反馈机制, 在 Web 服务器负载波动时将响应时间误差控制在 5%以内, 验证了控制理论在软件性能管理中的有效性。

3. 国内外研究对比

通过上面的分析可以看出, 在嵌入式软件中间件与软总线领域, 产业驱动的功能实现与集成实践发展迅速。国际厂商凭借长期技术积累, 在标准化与模块化设计上占据优势, 而国内企业则通过国产化替代与自主创新, 在特定领域实现突破性进展, 形成差异化竞争格局, 尤其在系统完整性、适配性及对分布式与大模型的支撑能力上, 具体对比如表 1 所示。

Table 1. Comparison of domestic and international middleware soft bus
表 1. 国内外中间件软总线对比

中间件		系统完整性	适配性	支撑性
应用服务器	国内	发展中，在核心系统应用增多	良好，适配国内特定行业需求	增长，国产化趋势下支持增强
	国外	成熟，功能全面支持 Java EE 标准	广泛，已支持多种企业级应用场景	强大，生态成熟且社区活跃
RPC 框架	国内	完善，侧重服务治理与稳定性	较好，适配国内复杂业务与高并发	强大，国内大厂主导且生态成熟
	国外	多样，性能与稳定性较为突出	广泛，覆盖微服务系统等	活跃，开源社区与大厂支持
消息中间件	国内	领先，低延迟高并发与事务突出	优秀，深度整合云生态，核心业务	强大，头部云厂商主导支持
	国外	成熟，各有侧重满足不同需求	广泛，支持异构系统、异步通信等	强大，开源社区活跃且应用广泛
缓存中间件	国内	发展中，云服务形态提供扩展性	良好，兼容 Redis 并优化云上体验	依赖云厂商，提供专业服务支持
	国外	成熟，功能丰富且性能优异	广泛，适用于多种场景和技术栈	强大，开源社区活跃，生态完善
任务调度中间件	国内	完善，功能丰富且注重分布式特性	优秀，适配国内生态和复杂场景	活跃，开源社区和企业支持良好
	国外	成熟，多为框架集成或云服务形式	广泛，支持定时、工作流等模式	良好，社区或云厂商可提供支持
离线数据中间件	国内	依赖开源，基于 Hadoop/Spark	良好，结合云服务提供整合方案	依赖云厂商，提供整合后的服务
	国外	成熟，Spark 性能相对较好	广泛，支持批处理、机器学习等	强大，开源社区庞大，生态完善
容器编排平台	国内	基于 K8s，深度整合云基础设施	良好，优化云上部署与特定行业	云厂商可提供全栈式服务支持
	国外	成熟，K8s 主导，选择多样化	极高，支持微服务、跨云等场景	强大，CNCF 及等力推，生态繁荣
其他中间件	国内	发展中，特定领域已突破性成果	良好，可满足大规模云平台需求	依赖云厂商/企业，提供针对性支持
	国外	成熟多样，商业服务与开源并存	广泛，可解决数据库、安全等问题	强大，大型云厂商和开源社区支持
软总线	国内	创新中，侧重设备互联与工业应用	发展中，在消费、工业领域突出	稳步增长，主要由大型公司推动
	国外	现有产品主要位于工业物联网领域	主要支持工业控制、机器人等	一般，商业产品与开源项目并存

3.1. 完整性

中间件的系统完整性是指基于模块化架构设计原则构建的全栈式技术体系，其通过标准化接口与服务治理机制实现对分布式系统核心功能的完整性支撑，涵盖数据持久化、事务协调、消息通信、服务调

度等关键场景的技术闭环。该属性要求中间件在功能完整性层面，需提供覆盖数据访问层至应用逻辑层的全链路技术栈，包括事务管理器、消息队列、远程过程调用(RPC)框架等核心组件，确保跨系统交互时的事务原子性、消息可靠传输及服务调用一致性；在架构完整性层面，采用分层解耦的模块化设计策略，通过可插拔组件实现功能扩展与场景适配。

在系统完整性方面，国际主流中间件产品如 IBM、Oracle 等凭借其标准化和模块化设计理念，构建了覆盖数据访问、事务处理等核心场景的完整技术栈，其解决方案在一致性和可靠性方面具有显著优势。

而随着国产化替代进程加速，国内中间件厂商在系统完整性领域取得长足进步，逐步形成功能完备的自主技术体系。从市场应用来看，国内企业更青睐能与现有技术生态深度整合的轻量化中间件，例如阿里巴巴 Dubbo 和蚂蚁金服 SOFARPC 这类高性能 RPC 框架，以及 RocketMQ 消息中间件，它们凭借开箱即用的特性和敏捷开发支持赢得了广泛采用。与此同时，以阿里云 SchedulerX 为代表的云原生任务调度中间件，正在通过弹性扩展和智能化运维能力重塑企业级中间件应用范式，展现出国产中间件在软总线架构下的创新活力。

3.2. 适配性

中间件的适配性，是指中间件产品在异构计算环境中实现跨平台互操作、多技术栈集成及动态扩展的综合能力体系。该特性不仅要求中间件具备跨平台兼容性以支持包括 Windows 与 Linux 在内的异构操作系统环境，更需要实现对各类主流数据库管理系统(如 Oracle、MySQL)及编程语言生态(涵盖 Java、Python 等)的无缝集成，典型代表如 Apache Kafka 通过构建统一的分布式消息协议，有效解决了复杂技术栈下的系统间通信难题。

在适配性方面，国际主流中间件产品通常强调跨平台兼容性，能够无缝支持 Windows、Linux 等多种操作系统，并适配各类主流数据库管理系统及编程语言环境，例如 Apache Kafka 凭借其出色的跨平台能力成为全球广泛采用的分布式消息系统。

与此同时，随着国内自主可控需求的持续增长，本土中间件产品的适配策略呈现出差异化特征——除需兼容国际主流技术生态外，更需深度适配国产化技术栈，包括华为 OpenHarmony 操作系统、达梦数据库等自主基础软件，这种双重适配能力正成为国产中间件在金融、政务等关键领域落地的重要竞争力。当前，领先厂商已通过分层架构设计和标准化接口，逐步构建起兼顾国际通用性与本土化特色的弹性适配体系。

3.3. 对分布式、大模型的支撑

中间件对分布式架构与大模型的支撑性体现为基于云原生技术构建的弹性服务治理体系与智能计算基础设施的深度融合。在分布式架构层面，通过深度整合 Kubernetes 等容器编排技术，实现微服务架构的动态调度与资源优化，支持多节点协同计算与弹性伸缩能力，特别是在边缘计算场景下形成差异化技术优势。针对大模型支撑，依托训练推理一体化架构构建 AI 中间件平台，通过内存计算优化、分布式任务调度和异构硬件适配，有效提升计算机视觉、自然语言处理等领域的模型训练效率与推理性能，同时实现生成式 AI 场景下的算力资源动态分配。技术实现路径既包含对国产软硬件生态的深度适配，又强调通过开源策略促进 PaaS 平台融合创新，形成覆盖基础资源调度、智能算法优化、服务治理监控的全栈式支撑体系，最终构建起兼具高并发处理能力与智能计算特性的新型中间件技术范式。

在分布式架构支持方面，国际主流中间件已深度整合 Kubernetes 等云原生技术，通过容器编排能力有力推动了全球微服务架构的普及，同时积极拥抱 AI 与大数据浪潮——如 Apache Spark 凭借内存计算优化成为大数据处理领域的事实标准。

国内中间件生态则呈现出“双轨并行”的发展态势：一方面，阿里云、华为云等厂商基于自身云计算优势，推出支持弹性伸缩的分布式中间件解决方案，在边缘计算等新兴场景形成特色竞争力；另一方面，以百度飞桨为代表的 AI 中间件平台正加速突破，不仅在计算机视觉、自然语言处理等传统优势领域持续深耕，更通过大模型训练推理一体化架构，推动国产中间件在生成式 AI 时代实现技术跃迁。这种在基础架构与智能应用两条技术路线上的同步突破，正在重塑全球中间件产业的技术格局。

经过对比可以看出，国内中间件行业在国产化替代和技术创新方面取得了显著进展，通过政策支持和企业自主研发，打破了国外厂商的垄断，特别是在金融、电信和政府等关键领域实现了广泛应用。国内厂商积极拥抱云计算和微服务架构，推动中间件与 PaaS 平台的深度融合，并通过开源化策略构建了繁荣的技术生态，降低了研发成本并提升了产品可靠性。此外，国内中间件特别注重对国产软硬件的支持及在人工智能、大数据分析等新兴领域的优化，展现了强劲的发展势头和独特优势。

4. 发展趋势和展望

4.1. 系统完整性增强

在系统完整性方面，嵌入式中间件正朝着更高标准化和模块化设计的方向演进。国际厂商如 IBM、Oracle 等凭借成熟的技术积累，其解决方案在数据访问、事务处理等关键环节展现出较强的完整性和一致性，能够满足企业级应用的复杂需求。

在国内信创产业加速发展的背景下，本土中间件厂商正通过技术创新和生态整合，持续提升产品的系统完整性。尤其在金融、电信、政务等关键行业，国产中间件已逐步实现对国际产品的替代，并在高可用性、安全合规等方面形成差异化优势。未来，随着行业标准的完善和开源生态的成熟，国内外中间件在系统完整性上的差距有望进一步缩小，推动全球中间件市场向更加开放、兼容的方向发展。

4.2. 适配性提高

当前中间件的发展正呈现出显著的适配性升级趋势。国际主流产品持续强化跨平台能力，在 Windows、Linux 等操作系统、多种数据库管理系统及开发语言环境之间实现无缝兼容，例如 Apache Kafka 凭借其卓越的跨系统适配性成为分布式架构的关键组件。

在自主可控战略驱动下，国内中间件厂商正构建“双轨适配”体系——既要确保对国际主流技术栈的完整支持，又要深度适配国产化生态。这一趋势在金融、能源等关键领域表现尤为突出，国产中间件已逐步实现对华为 OpenHarmony、欧拉操作系统、达梦数据库等自主基础软硬件的优化支持。未来，随着信创产业的深入发展，中间件的适配能力将从简单的兼容性向性能调优、安全增强等深层次协同演进，推动形成更加开放且自主可控的技术生态。

4.3. 分布式系统及新兴技术支持

当前中间件技术正经历着分布式架构与智能计算的双重变革。国际领先的中间件产品已深度整合 Kubernetes 容器编排能力，为微服务架构提供强大支撑，同时 Apache Spark 等工具通过内存计算优化持续引领大数据处理领域。我国中间件产业在这一轮技术演进中展现出强劲的追赶态势：一方面，阿里云、华为云等平台基于自身云计算优势，在容器化部署、边缘计算节点管理等分布式场景实现技术突破；另一方面，在 AI 赋能方面，以百度飞桨平台为代表的 AI 中间件通过提供从模型训练到推理部署的全流程支持，正在快速缩小与国际领先水平的差距。阿里云 PaaS 平台则通过整合大模型能力，为开发者提供更高效率的 AI 应用开发环境。这种在基础架构与智能应用两个维度的协同发展，正在重塑全球中间件产业的技术格局。

国内中间件行业通过政策支持和企业自主研发，打破了国外厂商的垄断局面，并在金融、电信等关键领域实现了广泛应用。未来，通过积极拥抱开源化策略，将进一步降低研发成本，提升产品可靠性，构建更加繁荣的技术生态。随着技术的进步，嵌入式中间件与软总线将更深入地融入人工智能、大数据分析等新兴领域，推动各行业的数字化转型。尤其是在智能制造、智慧城市等应用场景中，它们将成为连接各类设备和服务的核心纽带，实现信息的有效传递和智能决策。

基金项目

本研究得到了陕西省重点研发计划项目——多源信息融合的驾驶员/飞行员状态监测和预警技术及应用(编号: 2024GX-ZDCYL-02-15)和陕西省杰出青年科学基金项目——天地一体化遥感智能协同解译方法研究(编号: 2025JC-JCQN-079)的资助。

参考文献

- [1] Razzaque, M.A., Milojevic-Jevric, M., Palade, A. and Clarke, S. (2016) Middleware for Internet of Things: A Survey. *IEEE Internet of Things Journal*, 3, 70-95. <https://doi.org/10.1109/jiot.2015.2498900>
- [2] Kistijantoro, A.I., Morgan, G., Shrivastava, S.K. and Little, M.C. (2008) Enhancing an Application Server to Support Available Components. *IEEE Transactions on Software Engineering*, 34, 531-545. <https://doi.org/10.1109/tse.2008.38>
- [3] Markiewicz, T. (2011) Using MATLAB Software with Tomcat Server and Java Platform for Remote Image Analysis in Pathology. *Diagnostic Pathology*, 6, 1-7. <https://doi.org/10.1186/1746-1596-6-s1-s18>
- [4] Birrell, A.D. and Nelson, B.J. (1984) Implementing Remote Procedure Calls. *ACM Transactions on Computer Systems*, 2, 39-59. <https://doi.org/10.1145/2080.357392>
- [5] Brock, B.A., Chen, Y., Yan, J., Owens, J., Buluc, A. and Yelick, K. (2019) RDMA vs. RPC for Implementing Distributed Data Structures. 2019 *IEEE/ACM 9th Workshop on Irregular Applications: Architectures and Algorithms (IA3)*, Denver, 18 November 2019, 17-22. <https://doi.org/10.1109/ia349570.2019.00009>
- [6] Wang, X., Zhao, H. and Zhu, J. (1993) GRPC: A Communication Cooperation Mechanism in Distributed Systems. *ACM SIGOPS Operating Systems Review*, 27, 75-86. <https://doi.org/10.1145/155870.155881>
- [7] Slee, M., Agarwal, A. and Kwiatkowski, M. (2007) Thrift: Scalable Cross-Language Services Implementation. *Facebook White Paper*, 5, 127.
- [8] Kiraly, S. and Szekely, S. (2018) Analysing RPC and Testing the Performance of Solutions. *Informatica*, 42, 555-561. <https://doi.org/10.31449/inf.v42i4.1510>
- [9] Kraft, H. and Johansson, R. (2020) Evaluating RPC for Cloud-Native 5G Mobile Network Applications. Department of Computer Science and Engineering, Chalmers University of Technology.
- [10] Hamo, N. and Saberian, S. (2023) Evaluating the Performance and Usability of HTTP vs gRPC in Communication between Micro-Services Faculty of Computing, Blekinge Institute of Technology.
- [11] Pamadi, V.N., Chaurasia, A.K. and Singh, T. (2020) Comparative Analysis of GRPC vs. ZeroMQ for Fast Communication. *International Journal of Emerging Technologies and Innovative Research*, 7, 937-951.
- [12] Wood, I. (2004) Distributed Message Transmission System and Method. WO, EP1477034.
- [13] Ge, Y., Liang, X.X., Pan, Z., et al. (2018) Message Parsing in a Distributed Stream Processing System. U.S. Patent Application 15/258,629, 2018-03-08.
- [14] Magnoni, L. (2015) Modern Messaging for Distributed Systems. *Journal of Physics: Conference Series*, 608, Article ID: 012038. <https://doi.org/10.1088/1742-6596/608/1/012038>
- [15] Anthony, A. and Rao, Y.N.M. (2022) Memcached, Redis, and Aerospike Key-Value Stores Empirical Comparison. University of Waterloo.
- [16] Chen, B., Zhang, L., Huang, X., Cao, Y., Lian, K., Zhang, Y., et al. (2024) Efficient Detection of Java Deserialization Gadget Chains via Bottom-Up Gadget Search and Dataflow-Aided Payload Construction. 2024 *IEEE Symposium on Security and Privacy (SP)*, San Francisco, 19-23 May 2024, 3961-3978. <https://doi.org/10.1109/sp54263.2024.00150>
- [17] Liu, F. and Weissman, J.B. (2015) Elastic Job Bundling: An Adaptive Resource Request Strategy for Large-Scale Parallel Applications. *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, Austin, 15-20 November 2015, 1-12. <https://doi.org/10.1145/2807591.2807610>
- [18] Shvachko, K., Kuang, H., Radia, S. and Chansler, R. (2010) The Hadoop Distributed File System. 2010 *IEEE 26th*

- Symposium on Mass Storage Systems and Technologies (MSST)*, Incline Village, 3-7 May 2010, 1-10. <https://doi.org/10.1109/msst.2010.5496972>
- [19] Salloum, S., Dautov, R., Chen, X., Peng, P.X. and Huang, J.Z. (2016) Big Data Analytics on Apache Spark. *International Journal of Data Science and Analytics*, **1**, 145-164. <https://doi.org/10.1007/s41060-016-0027-9>
 - [20] Pan, Y., Chen, I., Brasileiro, F., Jayaputera, G. and Sinnott, R. (2019) A Performance Comparison of Cloud-Based Container Orchestration Tools. 2019 *IEEE International Conference on Big Knowledge (ICBK)*, Beijing, 10-11 November 2019, 191-198. <https://doi.org/10.1109/icbk.2019.00033>
 - [21] Bernstein, D. (2014) Containers and Cloud: From LXC to Docker to Kubernetes. *IEEE Cloud Computing*, **1**, 81-84. <https://doi.org/10.1109/mcc.2014.51>
 - [22] Carrión, C. (2022) Kubernetes Scheduling: Taxonomy, Ongoing Issues and Challenges. *ACM Computing Surveys*, **55**, 1-37. <https://doi.org/10.1145/3539606>
 - [23] Casalicchio, E. (2018) Container Orchestration: A Survey. In: Puliafito, A. and Trivedi, K.S., Eds., *Systems Modeling: Methodologies and Tools*, Springer International Publishing, 221-235. https://doi.org/10.1007/978-3-319-92378-9_14
 - [24] Hua, L., Tang, T., Wu, H., Wu, Y., Liu, H., Xu, Y., et al. (2020) A Framework to Support Multi-Cloud Collaboration. 2020 *IEEE World Congress on Services (SERVICES)*, Beijing, 18-23 October 2020, 110-116. <https://doi.org/10.1109/services48979.2020.00036>
 - [25] Han, M., Zhang, J., Wang, Y., Yan, R. and Wu, H. (2024) Microservices Architecture: Application and Outlook. In: Chinese Institute of Command and Control, Ed., *Proceedings of 2024 12th China Conference on Command and Control*, Springer, 1-10. https://doi.org/10.1007/978-981-97-7774-7_1
 - [26] Jawarneh, I.M.A., Bellavista, P., Bosi, F., Foschini, L., Martuscelli, G., Montanari, R., et al. (2019) Container Orchestration Engines: A Thorough Functional and Performance Comparison. 2019 *IEEE International Conference on Communications (ICC)*, Shanghai, 20-24 May 2019, 1-6. <https://doi.org/10.1109/icc.2019.8762053>
 - [27] Karun, A.K. and Chitharanjan, K. (2013) A Review on Hadoop-HDFS Infrastructure Extensions. 2013 *IEEE Conference on Information & Communication Technologies*, Thuckalay, 11-12 April 2013, 132-137.
 - [28] Hall, D.E., Greiman, W.H., Johnston, W.F., Merola, A.X., Loken, S.C. and Robertson, D.W. (1989) The Software Bus: A Vision for Scientific Software Development. *Computer Physics Communications*, **57**, 211-216. [https://doi.org/10.1016/0010-4655\(89\)90214-2](https://doi.org/10.1016/0010-4655(89)90214-2)
 - [29] Niemelä, E., Perunka, H. and Korpipää, T. (1998) A Software Bus as a Platform for a Family of Distributed Embedded System Products. In: van der Linden, F., Ed., *Development and Evolution of Software Architectures for Product Families*, Springer, 14-23. https://doi.org/10.1007/3-540-68383-6_4
 - [30] Selim, M.R., Endo, T., Goto, Y. and Cheng, J. (2006) A Comparative Study between Soft System Bus and Traditional Middlewares. In: Meersman, R., Tari, Z. and Herrero, P., Eds., *On the Move to Meaningful Internet Systems 2006: OTM 2006 Workshops*, Springer, 1264-1273. https://doi.org/10.1007/11915072_30
 - [31] Eles, P., Doboli, A., Pop, P. and Peng, Z. (2000) Scheduling with Bus Access Optimization for Distributed Embedded Systems. *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, **8**, 472-491. <https://doi.org/10.1109/92.894152>
 - [32] Sijtema, M., Belinfante, A., Stoelinga, M.I.A. and Marinelli, L. (2014) Experiences with Formal Engineering: Model-Based Specification, Implementation and Testing of a Software Bus at Neopost. *Science of Computer Programming*, **80**, 188-209. <https://doi.org/10.1016/j.scico.2013.04.009>
 - [33] Liu, F. (2018) Analysis on the Distributed Computer Software Bus Architecture. In: *Proceedings of the 2018 3rd International Workshop on Materials Engineering and Computer Sciences (IWMECS 2018)*, Atlantis Press, 114-117. <https://doi.org/10.2991/iwmeecs-18.2018.25>
 - [34] Purtilo, J.M. (1994) The POLYLITH Software Bus. *ACM Transactions on Programming Languages and Systems*, **16**, 151-174. <https://doi.org/10.1145/174625.174629>
 - [35] Cheng, J. (2004) Soft System Bus as a Future Software Technology. *Systems Engineering*, **7**, 8.
 - [36] Xu, K. and Shen, W. (2020) Software Development Method Based on Software Bus. 2020 *International Conference on Advance in Ambient Computing and Intelligence (ICAACI)*, Ottawa, 12-13 September 2020, 147-150. <https://doi.org/10.1109/icaaci50733.2020.00037>
 - [37] Sachs, K., Kounev, S., Bacon, J. and Buchmann, A. (2009) Performance Evaluation of Message-Oriented Middleware Using the Specjms2007 Benchmark. *Performance Evaluation*, **66**, 410-434. <https://doi.org/10.1016/j.peva.2009.01.003>
 - [38] Gokhale, A. and Schmidt, D.C. (1996) Measuring the Performance of Communication Middleware on High-Speed Networks. *ACM SIGCOMM Computer Communication Review*, **26**, 306-317. <https://doi.org/10.1145/248157.248183>
 - [39] da Cruz, M.A.A., Rodrigues, J.J.P.C., Sangaiah, A.K., Al-Muhtadi, J. and Korotayev, V. (2018) Performance Evaluation of IoT Middleware. *Journal of Network and Computer Applications*, **109**, 53-65.

-
- <https://doi.org/10.1016/j.jnca.2018.02.013>
- [40] Patro, S., Potey, M. and Golhani, A. (2017) Comparative Study of Middleware Solutions for Control and Monitoring Systems. 2017 *2nd International Conference on Electrical, Computer and Communication Technologies (ICECCT)*, Coimbatore, 22-24 February 2017, 1-10. <https://doi.org/10.1109/icecct.2017.8117808>
 - [41] Al-Jaroodi, J. and Mohamed, N. (2012) Service-Oriented Middleware: A Survey. *Journal of Network and Computer Applications*, **35**, 211-220. <https://doi.org/10.1016/j.jnca.2011.07.013>
 - [42] Zhang, C. and Jacobsen, H. (2003) Quantifying Aspects in Middleware Platforms. *Proceedings of the 2nd international Conference on Aspect-Oriented Software Development*, Boston, 17-21 March 2003, 130-139. <https://doi.org/10.1145/643603.643617>
 - [43] García Valls, M. and Basanta Val, P. (2014) Comparative Analysis of Two Different Middleware Approaches for Re-configuration of Distributed Real-Time Systems. *Journal of Systems Architecture*, **60**, 221-233. <https://doi.org/10.1016/j.sysarc.2013.08.010>
 - [44] Zhang, R., et al. (2002) ControlWare: A Middleware Architecture for Feedback Control of Software Performance. *Proceedings 22nd International Conference on Distributed Computing Systems*, Vienna, 2-5 July 2002, 301-310.

异构智能嵌入式系统AI模型推理与部署优化

——从模型轻量化到系统级加速的综述与展望

庞为光

齐鲁工业大学(山东省科学院), 山东省计算中心(国家超级计算济南中心), 山东 济南

收稿日期: 2025年10月28日; 录用日期: 2025年11月20日; 发布日期: 2025年12月1日

摘要

随着人工智能技术与嵌入式硬件的快速发展, 嵌入式人工智能系统(如移动机器人、自动驾驶汽车和星载无人机)在工业自动化、交通运输和航空航天等关键领域变得越来越重要。作为集成CPU、GPU、NPU等多种异构处理器单元的智能实时系统, 其核心任务是通过计算密集型的深度神经网络(DNN)实现环境感知、决策控制等复杂功能, 同时面临严格的时间约束与资源瓶颈。文章从网络模型在嵌入式系统加速推理优化的角度, 将围绕DNN模型轻量化、推理加速优化与动态任务调度三个方面, 详细分析嵌入式智能系统的国内外研究现状。

关键词

异构嵌入式系统, 深度神经网络, 推理加速, 实时调度

AI Model Inference and Deployment Optimization for Heterogeneous Intelligent Embedded Systems

—A Survey and Perspective from Model Lightweighting to System-Level Acceleration

Weiguang Pang

Shandong Computer Science Center (National Supercomputing Center in Jinan), Qilu University of Technology (Shandong Academy of Sciences), Jinan Shandong

Received: October 28, 2025; accepted: November 20, 2025; published: December 1, 2025

Abstract

With the rapid advancement of artificial intelligence (AI) technologies and embedded hardware, embedded AI systems—such as mobile robots, autonomous vehicles, and spaceborne unmanned aerial vehicles—are becoming increasingly significant in key domains including industrial automation, transportation, and aerospace. As intelligent real-time systems integrating heterogeneous processing units such as CPUs, GPUs, and NPUs, their core mission is to execute computationally intensive deep neural networks (DNNs) to achieve complex functions such as environmental perception and decision control, all under stringent temporal constraints and resource limitations. From the perspective of accelerating and optimizing neural network inference on embedded systems, this paper provides a comprehensive analysis of the current research progress at home and abroad, focusing on three major aspects: DNN model lightweighting, inference acceleration optimization, and dynamic task scheduling.

Keywords

Heterogeneous Embedded Systems, Deep Neural Networks, Inference Acceleration, Real-Time Scheduling

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

深度神经网络模型轻量化技术是突破人工智能应用在嵌入式系统部署瓶颈的关键路径，其通过算法重构与参数压缩的双重优化，在可接受网络精度损失范围内构建高效推理模型。在算法层面，知识蒸馏技术实现复杂模型向轻量化架构的能力迁移，配合模块化网络设计降低结构冗余；参数剪枝(结构化/非结构化)、量化(二值化/混合精度)及低秩分解等方法系统性地减少模型计算量[1]。硬件适配层面则通过稀疏矩阵加速器、多分支网络架构等定制化设计，提升轻量化模型在嵌入式异构平台的能效表现，形成算法-硬件协同优化方法[2]。

面向大语言模型的嵌入式部署需求，轻量化技术呈现细粒度创新趋势：一方面，根据量化所应用的不同阶段，可以将量化方法分为三类：量化感知训练(QAT, Quantization-Aware Training)、量化感知微调(QAF, Quantization-Aware Fine-tuning)及训练后量化(PTQ, Post-Training Quantization) [3]。QAT 在模型的训练过程中采用量化，QAF 在预训练模型的微调阶段应用量化，PTQ 在模型完成训练后对其进行量化，并结合硬件特性开发出极限低比特的整型(如 INT4、INT8)压缩方案；另一方面，混合专家模型等异构架构革新了模型部署范式，通过大小模型动态协作实现推理效率的阶跃式提升[4]。此类技术使百亿参数级模型在嵌入式设备端的实时推理成为可能，推动嵌入式系统向智能认知层级跨越。

2. 嵌入式智能系统推理优化加速技术发展现状

当前嵌入式智能系统的网络模型部署主要集中在推理加速优化，其技术策略在保持模型精度的前提下提升运行效率。核心优化方向包括网络模型编译优化、异构资源调度以及存储计算优化。英伟达的 TensorRT 推理框架通过算子融合与内存优化技术有效提升了推理速度[5]。关于网络模型推理任务在异构计算单元上的分配方法，当前研究工作采用模型并行、数据并行和流水线并行等模型的推理加速方法，

进一步提升了嵌入式系统上的模型推理性能[6]。

在大语言模型在嵌入式系统上优化部署方面,伊利诺伊大学针对大模型输出长度不确定导致的端到端推理时间不可预测问题,提出了一种推测性最短作业优先调度器。该方案利用轻量级代理模型预测大模型输出序列长度,有效解决了传统先到先服务调度的队首阻塞问题[7]。英伟达开发了动态内存压缩技术,通过在推理过程中在线压缩键值缓存,成功缓解因输入序列长度与批处理规模线性增长引发的缓存膨胀问题[8]。首尔大学提出的细粒度调度机制实现了迭代级连续批处理,可通过动态整合多个大模型请求显著提升推理效率[9]。针对 Transformer 架构的计算特性,学界提出了 KV 缓存复用、FlashAttention 以及 PageAttention 等加速方法[10],并结合投机采样与混合专家模型技术,在保证模型精度的前提下实现推理效率突破。

国内研究团队在模型推理加速领域取得显著进展。北京邮电大学在片上神经处理单元实现高效设备端大模型预填充加速的系统,该系统通过在提示供工程、张量和模型三个层次上优化了大模型在端侧设备上的推理,从而显著减少了推理延迟[11]。东北大学在边端系统推理加速方面积累了较多的系统部署优化基础,其中 GPU 并行加速方面研究了 GPU 内部异构计算核心的并行策略,提升了系统整理利用率和任务吞吐量[12]。国内人工智能团队 DeepSeek 通过创新的多头隐式注意力(MLA)设计,突破了现有优化方案的瓶颈,使得模型在存储和计算效率上达到了新的高度[13]。

3. 动态智能任务实时调度方法发展现状

动态神经网络通过运行时自适应调整模型结构或参数,成为实时系统应对计算资源约束的关键技术。其核心优势在于能够根据输入特征(如图像尺寸、批处理规模)及系统约束(如截止期限、资源限制),如图 1 所示,动态神经网络通过灵活调整网络压缩率、分支路径或输出节点,实现负载的动态适配[14]-[16]。例如,通过动态调节输入图像分辨率或网络分支选择,模型可在保证模型精度的同时显著降低推理延迟,满足工业物联网、自动驾驶等场景的实时性需求。

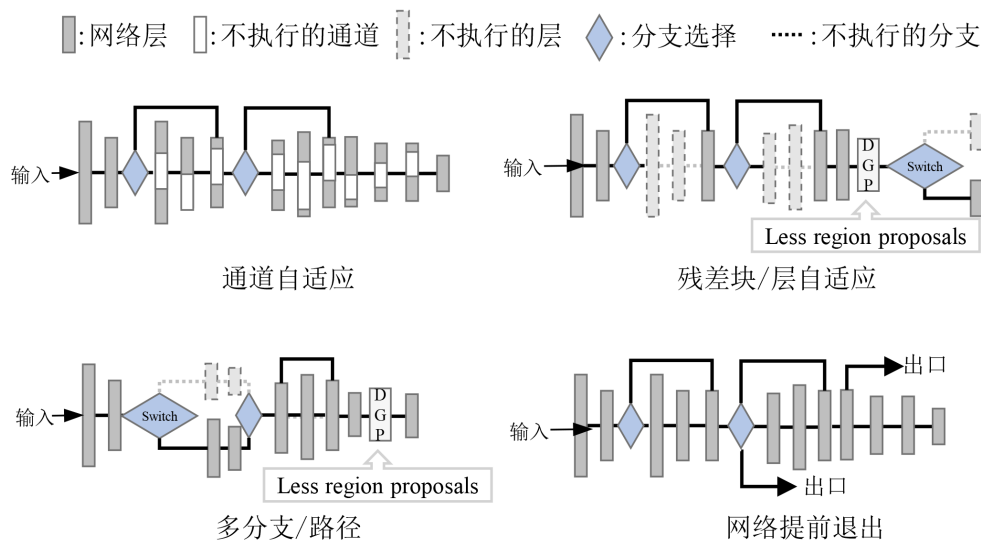


Figure 1. Dynamic neural network-based workload adjustment method

图 1. 动态神经网络调节计算负载方法

在动态推理 DNN 任务方面,学术界提出了多维度的系统调度方法。美国得克萨斯大学所提出的近似网络,量化了计算负载缩减与精度/时延的关联模型,支持运行时动态负载调整[17][18]。韩国庆熙大学研

研究者结合 GPU 最坏执行时间分析与自适应图像缩放技术,设计了动态路径切换机制,在任务截止期约束下将精度损失降至最低[19][20]。工业界则聚焦轻量化动态架构创新,如三星公司提出的分支条件神经网络(BPNet)实现了系统化的时间与精度权衡[21]。苹果公司开发的 UPSCALE 通道剪枝策略通过权重重排序技术,实现了无显著时延代价的动态网络裁剪[22]。微软提出基于全局的大批量 LLM 推理优化前缀共享和面向吞吐量的令牌批处理方法,通过全局前缀识别与请求调度重组、内存中心的分批处理及水平融合注意力核优化,实现共享前缀的 KV 上下文高效复用、预填充与解码阶段的 GPU 负载均衡,显著提升工业场景下大批量 LLM 推理效率[23]。北卡罗莱纳大学提出的 SubFlow 框架从模型结构层面出发,利用动态诱导子图策略在运行时根据任务截止期自适应选择子网络路径,实现了可变时间预算下的低时延高精度推理,为网络任务动态推理提供了新思路[24]。韩国汉阳大学提出的 Exegpt 系统则从系统层面出发,引入约束感知资源调度机制,通过联合优化批量大小与 GPU 分配,在延迟约束下实现高吞吐并发推理,体现了动态推理在资源调度与 QoS 保障方面的潜力[25]。

国内学者在动态自适应负载建模与部署优化方面取得显著进展。清华大学团队系统阐述了动态神经网络的理论框架[16]。上海交通大学通过扩展深度学习编译器实现了动态网络的高效推理支持[26]。上海科技大学进一步提出带时间约束的自适应任务模型,构建了兼顾服务质量与实时性的调度优化框架[1]。西北工业大学则聚焦环境自适应技术,通过动态调整模型参数降低资源消耗,为智能物联网系统提供高效解决方案[27]。香港中文大学利用深度学习编译技术在 GPU 上实现多 DNN 推理任务调度,在不损失网络精度的情况下,通过神经网络图和内核优化,提高 GPU 并行性,减少多任务之间的资源争用[28]。东北大学在异构 CPU-GPU 平台上的多 DNN 调度方面[29],采用有效的 CUDA 流优先级管理方法实现了不同优先级多 DNN 任务在共享 GPU 上的实时调度策略。

4. 发展趋势与展望

随着大模型逐步渗透至边缘端,主流技术的发展推动了模型轻量化和压缩技术的突破。通过模型压缩、量化和知识蒸馏等手段,使得模型在资源受限的嵌入式设备(如手机和机器人)上实现高效推理和实时响应,同时配合实时调度技术,确保动态任务处理能力。2025 年被视为“具身智能元年”,嵌入式系统借助轻量化和压缩技术,助力人形机器人在工业、医疗、家庭和自动驾驶等场景中完成复杂操作与实时决策,体现了主流技术在物理交互领域的应用优势和调度能力。原生多模态大模型整合视觉、音频、文本及 3D 数据,通过端到端训练实现数据对齐,并借助低功耗 AI 芯片和边缘计算平台降低推理延迟。此过程中,模型轻量化与实时调度技术是实现综合感知与实时处理的关键支撑。未来嵌入式智能系统将向垂直领域定制化发展,例如医疗诊断、农业机器人和消费电子。主流技术的发展促使模型更轻量、压缩更高效,同时借助实时调度实现自主智能体的动态任务管理,推动“All-in-One”超级应用的崛起,实现多场景智能服务。总之,嵌入式智能系统的发展正依托主流技术的模型轻量化、压缩技术及实时调度能力,实现高效推理、多模态融合和精细化物理交互。未来,这些技术将在垂类应用与自主智能体领域发挥核心作用。

基金项目

本文受山东省自然科学基金资助项目 ZR2024QF052。

参考文献

- [1] Wang, W., Chen, W., Luo, Y., Long, Y., Lin, Z., Zhang, L., *et al.* (2024) Model Compression and Efficient Inference for Large Language Models: A Survey. arXiv: 2402.09748.
- [2] Liu, D., Kong, H., Luo, X., Liu, W. and Subramaniam, R. (2022) Bringing AI to Edge: From Deep Learning's Perspective.

- Neurocomputing*, **485**, 297-320. <https://doi.org/10.1016/j.neucom.2021.04.141>
- [3] Zhou, Z., Ning, X., Hong, K., *et al.* (2024) A Survey on Efficient Inference for Large Language Models. <https://arxiv.org/abs/2404.14294>
 - [4] Dai, D., Deng, C., Zhao, C., Xu, R.X., Gao, H., Chen, D., *et al.* (2024) DeepSeekMoE: Towards Ultimate Expert Specialization in Mixture-Of-Experts Language Models. *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Bangkok, 11-16 August 2024, 1280-1297. <https://doi.org/10.18653/v1/2024.acl-long.70>
 - [5] NVIDIA. (2024). TensorRT-LLM [Computer Software]. GitHub. <https://github.com/NVIDIA/TensorRT-LLM>
 - [6] Ascend. (2024). AscendSpeed [Computer Software]. GitHub. <https://github.com/Ascend/AscendSpeed>
 - [7] Qiu, H., Mao, W., Patke, A., *et al.* (2024) Efficient Interactive LLM Serving with Proxy Model-Based Sequence Length Prediction. arXiv: 2404.08509.
 - [8] Nawrot, P., Łańcucki, A., Chochowski, M., *et al.* (2024) Dynamic Memory Compression: Retrofitting LLMs for Accelerated Inference. arXiv: 2403.09636.
 - [9] Yu, G.I., Jeong, J.S., Kim, G.W., *et al.* (2022) Orca: A Distributed Serving System for {Transformer-Based} Generative Models. *16th USENIX Symposium on Operating Systems Design and Implementation (OSDI 22)*, 521-538.
 - [10] Kwon, W., Li, Z., Zhuang, S., Sheng, Y., Zheng, L., Yu, C.H., *et al.* (2023) Efficient Memory Management for Large Language Model Serving with PagedAttention. *Proceedings of the 29th Symposium on Operating Systems Principles*, Koblenz, 23-26 October 2023, 611-626. <https://doi.org/10.1145/3600006.3613165>
 - [11] Xu, D., Zhang, H., Yang, L., *et al.* (2024) Empowering 1000 Tokens/Second On-Device LLM Prefilling with MLLM-NPU. arXiv: 2407.05858v1.
 - [12] Pang, W., Jiang, X., Liu, S., Qiao, L., Fu, K., Gao, L., *et al.* (2024) Control Flow Divergence Optimization by Exploiting Tensor Cores. *Proceedings of the 61st ACM/IEEE Design Automation Conference*, San Francisco, 23-27 June 2024, 1-6. <https://doi.org/10.1145/3649329.3658462>
 - [13] Meng, F., Yao, Z. and Zhang, M. (2025) TransMLA: Multi-Head Latent Attention Is All You Need. arXiv: 2502.07864.
 - [14] 王子曦, 邵培南, 邓畅. 异构并行平台的 Caffe 推理速度提升方法[J]. 计算机系统应用, 2022, 31(2): 220-226.
 - [15] 尚绍法, 蒋林, 李远成, 等. 异构平台下卷积神经网络推理模型自适应划分和调度方法[J]. 计算机应用, 2023, 43(9): 2828-2835.
 - [16] Han, Y., Huang, G., Song, S., Yang, L., Wang, H. and Wang, Y. (2022) Dynamic Neural Networks: A Survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **44**, 7436-7456. <https://doi.org/10.1109/tpami.2021.3117837>
 - [17] Bo, Z., Guo, C., Leng, C., Qiao, Y. and Wang, H. (2024) RTDeepEnsemble: Real-Time DNN Ensemble Method for Machine Perception Systems. *2024 IEEE 42nd International Conference on Computer Design (ICCD)*, Milan, 18-20 November 2024, 191-198. <https://doi.org/10.1109/iccd63220.2024.00037>
 - [18] Han, Y., Liu, Z., Yuan, Z., Pu, Y., Wang, C., Song, S., *et al.* (2024) Latency-Aware Unified Dynamic Networks for Efficient Image Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **46**, 7760-7774. <https://doi.org/10.1109/tpami.2024.3393530>
 - [19] Heo, S., Jeong, S. and Kim, H. (2022) RTScale: Sensitivity-Aware Adaptive Image Scaling for Real-Time Object Detection. *34th Euro-Micro Conference on Real-Time Systems*, Modena, 5-8 July 2022, 1-22.
 - [20] Heo, S., Cho, S., Kim, Y. and Kim, H. (2020) Real-Time Object Detection System with Multi-Path Neural Networks. *2020 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, Sydney, 21-24 April 2020, 174-187. <https://doi.org/10.1109/rtas48715.2020.000-8>
 - [21] Park, K., Oh, C. and Yi, Y. (2020) BPNet: Branch-Pruned Conditional Neural Network for Systematic Time-Accuracy Tradeoff. *2020 57th ACM/IEEE Design Automation Conference (DAC)*, San Francisco, 20-24 July 2020, 1-6. <https://doi.org/10.1109/dac18072.2020.9218545>
 - [22] Wan, A., Hao, H., Patnaik, K., *et al.* (2023) UPSCALE: Unconstrained Channel Pruning. arXiv: 2307.08771.
 - [23] Zheng, Z., Ji, X., Fang, T., Zhou, F., Liu, C. and Peng, G. (2024) BatchLLM: Optimizing Large Batched LLM Inference with Global Prefix Sharing and Throughput-Oriented Token Batching. arXiv: 2412.03594.
 - [24] Lee, S. and Nirjon, S. (2020) SubFlow: A Dynamic Induced-Subgraph Strategy toward Real-Time DNN Inference and Training. *2020 IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS)*, Sydney, 21-24 April 2020, 15-29. <https://doi.org/10.1109/rtas48715.2020.00-20>
 - [25] Oh, H., Kim, K., Kim, J., Kim, S., Lee, J., Chang, D., *et al.* (2024) ExeGPT: Constraint-Aware Resource Scheduling for LLM Inference. *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, La Jolla, 27 April-1 May 2024, 369-384. <https://doi.org/10.1145/3620665.3640383>

-
- [26] Cui, W., Han, Z., Ouyang, L., *et al.* (2023) Optimizing Dynamic Neural Networks with Brainstorm. *17th USENIX Symposium on Operating Systems Design and Implementation (OSDI 23)*, Boston, 10-12 July 2023, 797-815.
 - [27] Wang, H., Zhou, X., Yu, Z., Liu, S., Guo, B., Wu, Y., *et al.* (2020) Context-aware Adaptation of Deep Learning Models for IoT Devices. *Scientia Sinica Informationis*, **50**, 1629-1644. <https://doi.org/10.1360/ssi-2020-0067>
 - [28] Zhao, Z., Ling, N., Guan, N. and Xing, G. (2022) Aaron: Compile-Time Kernel Adaptation for Multi-DNN Inference Acceleration on Edge GPU. *Proceedings of the 20th ACM Conference on Embedded Networked Sensor Systems*, Boston, 6-9 November 2022, 802-803. <https://doi.org/10.1145/3560905.3568050>
 - [29] Pang, W., Luo, X., Chen, K., Ji, D., Qiao, L. and Yi, W. (2023) Efficient CUDA Stream Management for Multi-DNN Real-Time Inference on Embedded GPUs. *Journal of Systems Architecture*, **139**, Article ID: 102888. <https://doi.org/10.1016/j.sysarc.2023.102888>

云边端协同计算与智能

周俊龙, 侯祥鹏, 兰 兰, 徐 妍

南京理工大学计算机科学与工程学院, 江苏 南京

收稿日期: 2025年10月28日; 录用日期: 2025年11月20日; 发布日期: 2025年12月1日

摘 要

云边端协同计算作为应对数据爆炸与实时智能需求的关键架构, 通过将计算任务合理分布于终端、边缘与云端, 有效缓解了传统云计算在高时延、带宽压力和隐私安全等方面的瓶颈。文章系统梳理了云边端协同计算与智能的国内外研究进展, 重点从云边端协同计算框架构建与分布式智能实现两个维度展开分析, 总结了包括任务卸载、资源调度、多模态感知、联邦学习等关键技术路径。文章进一步展望了未来发展趋势, 指出算力网络化、边缘智能化、绿色计算与内生安全将成为推动云边端协同走向纵深的核心方向, 为我国构建高效、可信、可持续的协同智能基础设施提供理论参考。

关键词

边缘智能, 云边端协同计算, 分布式智能

Cloud-Edge-End Collaborative Computing and Intelligence

Junlong Zhou, Xiangpeng Hou, Lan Lan, Yan Xu

School of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing Jiangsu

Received: October 28, 2025; accepted: November 20, 2025; published: December 1, 2025

Abstract

Cloud-Edge-End collaborative computing has emerged as a critical architecture to address the challenges of data explosion and the demand for real-time intelligence. By efficiently distributing computational tasks across end devices, edge nodes, and the cloud, this paradigm effectively mitigates bottlenecks inherent in traditional cloud computing, such as high latency, bandwidth pressure, and privacy concerns. This paper systematically reviews the state-of-the-art in Cloud-Edge-End collaborative computing and intelligence, analyzing key research from two primary dimensions: the construction of collaborative computing frameworks and the implementation of distributed intelligence.

It summarizes pivotal technical approaches, including task offloading, resource scheduling, multi-modal perception, and federated learning. Furthermore, the paper outlines future development trends, identifying computing power networking, edge intelligence, green computing, and intrinsic security as core directions for advancing collaborative intelligent infrastructure, providing a theoretical reference for building efficient, trustworthy, and sustainable systems.

Keywords

Edge Intelligence, Cloud-Edge-End Collaborative Computing, Distributed Intelligence

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

随着物联网设备数量的持续攀升、传感器应用的日益广泛以及生成式 AI 的蓬勃发展,全球数据规模正经历指数级扩张。国际数据公司(IDC)预测显示[1],至 2028 年全球数据总量将突破 384.6 ZB,较 2024 年实现倍增。在此背景下,传统云计算架构面临实时性不足、网络时延高以及分布式数据隐私风险等多重核心挑战。为应对上述挑战,云边端协同计算应运而生,通过网络连接多个节点(端设备、边缘服务器或云数据中心)协同工作,实现计算能力的扩展与并行化,有效降低核心网带宽压力[2],如图 1 所示。智能化技术的融入,使云边端协同计算具备环境自适应与协同决策能力[3]。人工智能模型被部署于从云到端的各个层级,使系统能够动态感知和适应复杂的外部环境,并进行协同决策。更重要的是,边缘智能激活了海量终端设备的数据价值,将其从被动的数据采集单元,转变为能够主动感知、推理和决策的智能体,为构建泛在智能服务奠定了坚实基础。国家层面高度重视并积极引导云边端协同与边缘智能的发展。《“十四五”数字经济发展规划》强调推进云网协同和算网融合发展,打造智能算力、通用算法和开发平台一体化的新型智能基础设施[4]。工业和信息化部等六部门联合发布的《算力基础设施高质量发展行动计划》提出要促进边缘算力协同部署,推动“云边端”算力泛在分布、协同发展[5]。这些政策文件共同构成了推动云边端协同计算与智能发展的顶层设计,为相关技术创新和产业应用指明了方向,注入了强大动力。

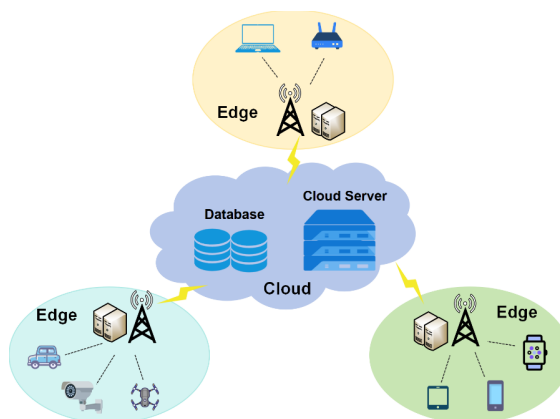


Figure 1. Cloud-edge-end collaborative computing architecture diagram

图 1. 云边端协同计算架构图

2. 国内外研究现状

在物联网与人工智能技术的推动下，云边端协同架构已成为学术界的研究前沿。当前的研究路径清晰地分化为两大相辅相成的领域。其一为云边端协同计算，该领域致力于构建一个普适、高效、可靠的分布式计算底座，将任务与资源进行抽象化处理，追求系统层面的全局最优。其二为分布式智能，该领域则从 AI 任务的内生需求出发，研究如何设计专有的协同架构与算法，以实现复杂的分布式感知、学习与决策，追求智能任务本身的效果最大化。

2.1. 云边端协同计算：普适性计算框架的构建与优化

本领域的核心目标是打造一个通用的分布式计算平台，它视上层应用为待处理的抽象任务，致力于解决在异构、动态环境下普适性的调度与管理难题。研究重点在于优化系统级的关键性能指标，如时延、能耗、成本和可靠性，为所有上层应用提供一个稳定高效的运行环境，如下表 1。

在计算卸载与资源分配这一核心问题上，学术界已发展出较为成熟的优化理论。传统方法侧重于运筹学模型，例如，华东师范大学吴连涛团队[6]通过构建多小区协作框架来最小化用户总成本；陈英团队[7]则从博弈论视角切入，构建了用户体验驱动的任务卸载模型。随着环境动态性的增强，基于 AI 的决策方法成为主流。郝昊团队[8]利用深度强化学习(DRL)赋能边缘节点进行动态协同决策；北京邮电大学范文浩团队[9]-[11]更是系统性地构建了“服务放置 - 任务调度 - 资源分配”三维协同框架，通过混合 DRL 突破了长期服务的稳定性瓶颈。南京理工大学周俊龙团队[12]聚焦智能交通场景，提出多智能体强化学习模型，实现车路协同场景下任务卸载和计算资源分配的最优决策。

在异构与跨域协同方面，研究者致力于打破资源壁垒，实现更广泛的算力共享。中南大学的王伟等人[13]提出的混合模式资源协作框架，实现了边缘与云算力的弹性适配。陈星延团队[14]创新性提出面向“云 - 边 - 端”算力系统的计算和传输联合优化方法，采用梯度投影法突破算力与网络耦合瓶颈。电子科技大学的张林团队[15]与英国萨里大学 Chong Huang 团队[16]的研究则将协同的范畴从地面网络拓展至“空间 - 空中 - 地面”一体化的立体网络，探索了该复杂场景下的能效与任务联合优化。

此外，为解决异构环境下的部署难题，张文柱等人[17]引入 Docker 容器化技术，为构建通用的多接入边缘计算环境提供了实践方案。

Table 1. Research status of cloud-edge-end collaborative computing
表 1. 云边端协同计算研究现状

研究方向	参考文献	核心技术/方法	主要贡献
计算卸载与资源分配	[6]	运筹学模型	构建多小区协作框架，以最小化用户总成本
	[7]	博弈论	构建用户体验驱动的任务卸载博弈模型
	[8]	深度强化学习	赋能边缘节点进行动态协同决策，适应环境变化
	[9]-[11]	混合深度强化学习	构建“服务放置 - 任务调度 - 资源分配”三维协同框架
	[12]	多智能体强化学习	优化车路协同场景下的任务卸载与资源分配
异构与跨域协同	[13]	混合模式协作框架	实现边缘算力与云算力的弹性按需适配
	[14]	梯度投影法	突破算力与网络资源的耦合瓶颈，实现联合优化
	[15] [16]	多智能体强化学习	将协同范畴拓展至“空间 - 空中 - 地面”一体化网络
通用部署	[17]	Docker 容器化技术	解决异构环境下多接入边缘计算的通用部署难题

2.2. 分布式智能：AI 赋能下的协同感知、决策与生成

与协同计算的普适性视角不同，分布式智能领域的研究出发点是 AI 任务本身，如表 2 所示。在分布式智能中，资源调度和协同策略不再是通用目的，而是被深度定制，以服务于特定的智能目标，如最大化模型精度、加快学习收敛速度或提升推理实时性。

在分布式协同感知方面，研究旨在融合多源、多模态数据以形成超越单点的场景理解。哈尔滨工业大学崔双双等人[18]通过构建多维特征张量，实现了云边端跨层异构数据的语义对齐，解决了多模态数据的融合难题。刘林峰团队[19]则将图注意力机制与时空卷积网络结合，这种为时空数据“定制”的融合框架，是实现精准城市交通预测的关键。来自 Uber 和多伦多大学的 Liang 等人[20]探索了如何利用车对车通信来提升自动驾驶汽车的感知和运动预测性能。他们通过智能整合从附近多辆车接收到的信息，能够从不同视角观察同一场景，穿透遮挡物，在观测数据极为稀疏或根本不存在的条件下，远距离检测目标。

在面向 AI 任务的实时协同方面，协同机制的设计完全服务于 AI 模型的实时推理需求。例如，赵林东团队[21]开发的数字孪生驱动架构，其核心是通过虚实映射机制，将工业预测性维护 AI 模型的决策延迟压缩至毫秒级。华中科技大学郝义学团队[22]提出的轻量化实时计数框架，其采用的可微分神经架构搜索和视频流动态裁剪技术，本质上是一种 AI 驱动的协同策略——系统根据 AI 模型的需求来智能决定传输哪些数据，从而在保证计数精度的前提下极大降低带宽消耗。同样，北京航空航天大学曹志强团队[23]设计的自适应判别器协作检测系统，其“边筛选、云精判”的模式，是为优化困难样本检测这一特定 AI 任务而设计的协同流程。

在分布式协同学习与生成方面，研究已进入更复杂的协同智能阶段。例如，联邦学习(Federated Learning)是分布式智能的标志性范式，其客户端选择、模型聚合等协同策略，完全是为了在保护数据隐私的前提下，高效、准确地完成分布式模型训练这一核心智能任务。例如，由来自 Google 的研究人员 McMahan 等人提出的开创性工作[24]，其设计的联邦平均(FedAvg)算法奠定了该领域的基础，使得在不共享原始数据的情况下，通过聚合本地模型更新来训练全局模型成为可能。然而，真实场景中各节点的数据异构性(Non-IID)会严重影响模型收敛，为解决这一挑战，Li 等人提出的 FedProx 算法[25]通过引入近端项来约束本地训练过程，显著提升了异构环境下的模型稳定性和性能。同样，在生成式 AI 的浪潮下，如何通过边云协同来运行大型语言模型(LLM)成为前沿，研究者们探索将模型进行切分，或通过检索增强生成(RAG)让边云协同完成推理，这些协同方式的设计初衷均是为了实现高质量的内容生成。如基于向量数据库的智能缓存机制 VELO 框架[26]。该方法在边缘侧缓存 LLM 的问答对，并通过多智能体强化学习训练一个决策智能体。当新请求到达时，智能体能判断该请求与缓存中问题的相似度，从而决定是直接返回“语义相似”的答案，还是向云端发起新的推理请求。这种将 LLM 视为“黑盒”并优化其服务流程的思路，为降低 LLM 使用成本和延迟提供了极具实用价值的方案。

Table 2. Research status of distributed intelligence
表 2. 分布式智能研究现状

研究方向	参考文献	核心技术/方法	主要贡献
分布式协同感知	[18]	多维特征张量	解决云边端跨层异构数据(视频、点云、文本)的语义对齐与融合难题
	[19]	图注意力机制和时空卷积网络	构建为时空数据“定制”的融合框架，实现精准城市交通预测
	[20]	空间感知图神经网络	融合多车辆视角信息，突破单车感知遮挡，提升自动驾驶感知性能

续表

面向 AI 任务的实时协同	[21]	数字孪生驱动架构	通过虚实双向映射机制，将 AI 模型决策延迟压缩至毫秒级
	[22]	可微分神经架构搜索	AI 驱动的动态视频流裁剪，根据模型需求智能传输数据，保证精度并降低带宽
	[23]	自适应判别器和知识蒸馏	设计“边筛选、云精判”模式，为优化困难样本检测这一特定 AI 任务而协同
分布式协同学习与生成	[24]	联邦平均算法	奠定联邦学习基础，在保护隐私前提下实现分布式模型训练
	[25]	FedProx 算法	解决数据异构性(Non-IID)问题，提升联邦学习的稳定性
	[26]	向量数据库和 MARL	将 LLM 视为“黑盒”，通过智能缓存优化服务流程，降低延迟与成本

3. 发展趋势和展望

随着云边端协同计算与智能技术的深度融合，其发展将呈现以下核心趋势：

(1) 从资源孤岛到分布式算力网络

面对 AI 模型、数字孪生等应用带来的爆炸式算力需求，传统的、孤立的边、云数据中心正在向一个统一、协同的“算力网络”演进。其核心目标是让算力像电力和网络一样，成为一种可按需取用、即取即用的社会级服务。一方面，从“任务卸载”到“算力路由”，系统将具备全局视野，能够实时感知从云到端所有可用算力资源(CPU、GPU、NPU 等)的状态、成本和能耗。如同网络中的路由协议，智能调度算法将为每一个计算任务动态规划出最优的“算力路由”，实现跨地域、跨运营商、跨空天地的算力最优分配。另一方面，从“应用适配算力”到“算力主动服务应用”：未来的算力网络将是“意图驱动”的。开发者只需声明其应用对时延、成本、精度等方面的需求，算力网络便能自动化地完成服务部署、资源配置、弹性伸缩和故障自愈的全生命周期管理，从而极大降低复杂应用的开发和运维门槛。

(2) 终端与边缘成为智能创新的核心

AI 能力下沉至边缘，已成为应对实时性需求与隐私保护双重压力的必然趋势。例如，机器视觉作为智能制造的典型场景，需要上行带宽超过 50 Mbps 甚至 200 Mbps，端到端通信时延小于 10 ms，可靠性要求大于 99.9999% [27]。这一趋势正从根本上颠覆长期以来“云重边轻”的模式，推动终端与边缘从被动的“数据采集器”转变为主动的“智能体”。这场变革的实现，得益于硬件算力的突破与协同算法的创新。硬件层面，以华为海思 Ascend310 芯片为例，其能以约 25 W 的低功耗提供高达 22TOPS 的 AI 算力，为在边缘部署复杂 AI 模型提供了坚实的物理基础[28]。算法层面，智能化的协同范式不断涌现，为不同 AI 任务在边缘的落地提供了可行路径。以联邦学习为代表的技术使边缘节点群能在不共享原始数据的前提下，协同训练强大的全局模型，并可进一步演化为最适应本地需求的个性化模型，实现“群体智能”与“个体智能”的共生。对于生成式 AI，业界正积极探索“小模型在边缘、大模型在云端”的高效协同模式。其核心是利用边缘侧的轻量化模型和智能语义缓存处理海量高频请求，仅在需要深度创作或复杂推理时才调用云端大模型，从而在保证服务质量的同时大幅降低 AIGC 的应用成本，为其在各行业的普及铺平道路。

(3) 绿色计算成为协同架构的内生目标

随着全球数据总量和计算需求的激增，数据中心和通信网络的能耗问题日益凸出，全球数据中心年

耗电量已达 1000 TWh [27] [29]。云边端协同架构通过将计算推向数据源头,天然具备了节能潜力。未来,“绿色计算”将从一个附加选项,变为协同架构设计与调度的核心约束和优化目标。绿色化路径呈现出从宏观能源供给到微观芯片设计,再到系统级智能调度的多层次、立体化特征:一是供电模式的绿色革新,例如腾讯云正大规模开发建设数据中心分布式新能源项目,预计 2030 年实现 100%绿色电力,自身运营及供应链的全面碳中和[30];二是硬件能效提升,从追求性能到专注“算力功耗比”。例如,3D 堆叠、存算一体、新型半导体材料以及专用计算架构等技术在加速发展,其共同目标是在有限的功耗预算内,为边缘设备提供出最大的有效算力。三是架构与算法的协同节能,将计算推向数据源头,最大限度减少非必要的数据传输。架构级节能通过在边缘侧完成数据预处理、清洗和特征提取,仅将高价值、小体积的结果数据上传至云端,从物理上减少了广域网的流量和能耗。算法级节能中轻量化的模型压缩与量化技术,则直接降低了在资源受限的边缘设备上运行算法所需的计算能耗。这种从传输“原始数据”到传输“有效知识”的转变,是实现全链路绿色化的关键。

(4) 安全与隐私保护加强

云边端协同在带来灵活性和高效性的同时,也因其开放和分布式的特性,引入了更复杂的安全挑战,据统计 2024 年物联网恶意软件攻击增长了 45%,其中路由器是攻击目标最多的设备类型,超过 66%的攻击针对这些设备[31]。安全能力必须“内生”于协同体系的每一环,并构建于零信任架构与隐私计算范式两大基石之上。首先是遵循“永不信任,永远验证”原则的零信任架构:在传输层,国密 SM9 等高性能加密算法[32]可为 5G 边缘网络提供高速安全信道;在计算层,基于可信执行环境的安全节点(如蚂蚁链“摩斯”)能实现高效的机密计算与联邦学习;在存储层,“分片加密-边缘存储-云端审计”等机制,通过将数据分散加密存储,能有效防范因单点故障或攻击导致的数据泄露与篡改风险。其次是隐私计算范式的普及深化。以联邦学习为代表的“数据不动模型动”技术正成为处理敏感数据的标准范式,从源头保障用户隐私。在此基础上,业界正积极探索效率更高的多方安全计算、同态加密等前沿技术,以适应更广泛的协同计算场景。

参考文献

- [1] International Data Corporation (2025) IDC: Global Data Generation to Reach 159.2 ZB in 2024, More Than Doubling by 2028. <https://www.idc.com/getdoc.jsp?containerId=US52076424>
- [2] Chen, R. and Wang, X. (2023) Maximization of Value of Service for Mobile Collaborative Computing through Situation-Aware Task Offloading. *IEEE Transactions on Mobile Computing*, 22, 1049-1065. <https://doi.org/10.1109/tmc.2021.3086687>
- [3] 张晓东, 张朝昆, 赵继军. 边缘智能研究进展[J]. 计算机研究与发展, 2023, 60(12): 2749-2764.
- [4] 国家发展和改革委员会. “十四五”数字经济发展规划[EB/OL]. 2022-01-12, https://www.ndrc.gov.cn/fggz/fzzlgh/gijzxgh/202203/t20220325_1320207.html, 2025-02-24.
- [5] 工业和信息化部, 中央网络安全和信息化委员会办公室, 教育部, 等. 关于印发《算力基础设施高质量发展行动计划》的通知[EB/OL]. 2023-10-09. https://www.gov.cn/zhengce/zhengceku/202310/content_6907900.htm, 2025-02-24.
- [6] Wu, L., Sun, P., Wang, Z., Li, Y. and Yang, Y. (2024) Computation Offloading in Multi-Cell Networks with Collaborative Edge-Cloud Computing: A Game Theoretic Approach. *IEEE Transactions on Mobile Computing*, 23, 2093-2106. <https://doi.org/10.1109/tmc.2023.3246462>
- [7] Chen, Y., Zhao, J., Wu, Y., Huang, J. and Shen, X. (2024) QoE-Aware Decentralized Task Offloading and Resource Allocation for End-Edge-Cloud Systems: A Game-Theoretical Approach. *IEEE Transactions on Mobile Computing*, 23, 769-784. <https://doi.org/10.1109/tmc.2022.3223119>
- [8] 郝昊, 杨树杰, 张伟. 算网融合下时间连续的计算任务卸载机制[J]. 计算机研究与发展, 2023, 60(4): 735-749.
- [9] Fan, W., Zhao, L., Liu, X., Su, Y., Li, S., Wu, F., et al. (2024) Collaborative Service Placement, Task Scheduling, and Resource Allocation for Task Offloading with Edge-Cloud Cooperation. *IEEE Transactions on Mobile Computing*, 23, 238-256. <https://doi.org/10.1109/tmc.2022.3219261>

- [10] Fan, W. (2024) Hybrid Deep Reinforcement Learning-Based Task Offloading for D2d-Assisted Cloud-Edge-Device Collaborative Networks. *IEEE Transactions on Mobile Computing*, **23**, 13455-13471. <https://doi.org/10.1109/tmc.2024.3425723>
- [11] Fan, W., Liu, X., Yuan, H., Li, N. and Liu, Y. (2024) Time-Slotted Task Offloading and Resource Allocation for Cloud-Edge-End Cooperative Computing Networks. *IEEE Transactions on Mobile Computing*, **23**, 8225-8241. <https://doi.org/10.1109/tmc.2024.3349551>
- [12] 郑莹莹, 周俊龙, 申钰凡, 等. 时间和能量敏感的端-边-云车路协同系统资源调度优化方法[J]. 计算机研究与发展, 2023, 60(5): 1037-1052.
- [13] Wang, W., Zhang, Y., Huang, R., Ren, J., Lyu, F. and Zhang, Y. (2024) Efficient Resource Management and Expansion Scheme for Collaborative Edge-Cloud Computing. *IEEE Transactions on Mobile Computing*, **23**, 2731-2747. <https://doi.org/10.1109/tmc.2023.3267497>
- [14] 陈星延, 张雪松, 谢志龙, 等. 面向“云-边-端”算力系统的计算和传输联合优化方法[J]. 计算机研究与发展, 2023, 60(4): 719-734.
- [15] Wang, Z., Zhang, L., Feng, D., Wu, G. and Yang, L. (2024) Intelligent Cloud-Edge Collaborations for Energy-Efficient User Association and Power Allocation in Space-Air-Ground Integrated Networks. *IEEE Journal on Selected Areas in Communications*, **42**, 3659-3673. <https://doi.org/10.1109/jsac.2024.3459089>
- [16] Huang, C., Chen, G., Xiao, P., Xiao, Y., Han, Z. and Chambers, J.A. (2024) Joint Offloading and Resource Allocation for Hybrid Cloud and Edge Computing in SAGINs: A Decision Assisted Hybrid Action Space Deep Reinforcement Learning Approach. *IEEE Journal on Selected Areas in Communications*, **42**, 1029-1043. <https://doi.org/10.1109/jsac.2024.3365899>
- [17] 张文柱, 余静华. 移动边缘计算中基于云边端协同的任务卸载策略[J]. 计算机研究与发展, 2023, 60(2): 371-385.
- [18] 崔双双, 吴限, 王宏志, 等. 面向云边端协同的多模态数据建模技术及其应用[J]. 软件学报, 2023, 35(3): 1154-1172.
- [19] Liu, L., Zhou, Y. and Xu, J. (2024) A Cloud-Edge-End Collaboration Framework for Cruising Route Recommendation of Vacant Taxis. *IEEE Transactions on Mobile Computing*, **23**, 4678-4693. <https://doi.org/10.1109/tmc.2023.3294898>
- [20] Wang, T., Manivasagam, S., Liang, M., Yang, B., Zeng, W. and Urtasun, R. (2020) V2VNet: Vehicle-To-Vehicle Communication for Joint Perception and Prediction. In: *Lecture Notes in Computer Science*, Springer, 605-621. https://doi.org/10.1007/978-3-030-58536-5_36
- [21] Zhao, L., Ni, S., Wu, D. and Zhou, L. (2023) Cloud-Edge-Client Collaborative Learning in Digital Twin Empowered Mobile Networks. *IEEE Journal on Selected Areas in Communications*, **41**, 3491-3503. <https://doi.org/10.1109/jsac.2023.3310060>
- [22] Wang, R., Hao, Y., Miao, Y., Hu, L. and Chen, M. (2024) RT3C: Real-Time Crowd Counting in Multi-Scene Video Streams via Cloud-Edge-Device Collaboration. *IEEE Transactions on Services Computing*, **17**, 1739-1752. <https://doi.org/10.1109/tsc.2024.3377156>
- [23] Cao, Z., Cheng, Y., Zhou, Z., Chen, Y., Hu, Y., Lu, A., et al. (2025) Edge-Cloud Collaborated Object Detection via Bandwidth Adaptive Difficult-Case Discriminator. *IEEE Transactions on Mobile Computing*, **24**, 1181-1196. <https://doi.org/10.1109/tmc.2024.3474743>
- [24] McMahan, B., Moore, E., Ramage, D., et al. (2017) Communication-Efficient Learning of Deep Networks from Decentralized Data. *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, Fort Lauderdale, 20-22 April 2017, 1273-1282.
- [25] Li, T., Sahu, A., Zaheer, M., et al. (2020) Federated Optimization in Heterogeneous Networks. *Proceedings of Machine Learning and Systems*, Austin, 2-4 March 2020, 429-450.
- [26] Yao, Z., Tang, Z., Yang, W. and Jia, W. (2025) Enhancing LLM QoS through Cloud-Edge Collaboration: A Diffusion-Based Multi-Agent Reinforcement Learning Approach. *IEEE Transactions on Services Computing*, **18**, 1412-1427. <https://doi.org/10.1109/tsc.2025.3562362>
- [27] 华为. 5G 确定性网络产业白皮书[R]. 深圳: 华为技术有限公司, 2019.
- [28] Ascend (2025) Intelligent Edge. <https://www.hiascend.com/en/hardware/intelligent-edge>
- [29] IEA (2024) Electricity 2024. <https://www.iea.org/reports/electricity-2024>
- [30] 中国信息通信研究院. 中国绿色算力发展研究报告[R]. 北京: 中国信息通信研究院, 2024.
- [31] Zscaler (2025) New ThreatLabz Report. <https://www.zscaler.com/blogs/security-research/new-threatlabz-report-mobile-remains-top-threat-vector-111-spyware-growth>
- [32] 国家密码管理局. SM9 标识密码算法行业标准[S]. 北京: 国家密码管理局, 2022.

嵌入式系统安全与可信技术发展趋势

张凯龙¹, 马萌旭¹, 何周灿¹, 樊弈辰¹, 周世钰², 闫琳珊¹

¹西北工业大学软件学院, 陕西 西安

²西北工业大学国家卓越工程师学院, 陕西 西安

收稿日期: 2025年10月28日; 录用日期: 2025年11月20日; 发布日期: 2025年12月1日

摘要

作为信息世界与物理世界的桥梁, 嵌入式系统发展呈现出网络化、智能化、协同化发展特征与趋势, 其安全、可信问题日益突出。首先分析、讨论了面向嵌入式系统的安全、可信计算体系与技术。进而, 重点从国内与国外、学术与技术、可信本体与计算架构等多个维度, 对增强、提升嵌入式系统安全、可信性的相关理论与技术进展进行了梳理、分析与对照。

关键词

嵌入式系统, 安全, 可信计算, 体系, 技术, 趋势

Development Trend of Security and Trusted Technologies for Embedded Systems

Kailong Zhang¹, Mengxu Ma¹, Zhoucan He¹, Yichen Fan¹, Shiyu Zhou², Linshan Yan¹

¹School of Software, Northwestern Polytechnical University, Xi'an Shaanxi

²National Institute of Excellence in Engineering, Northwestern Polytechnical University, Xi'an Shaanxi

Received: October 28, 2025; accepted: November 20, 2025; published: December 1, 2025

Abstract

As a bridge between the information world and the physical world, embedded systems are evolving toward greater connectivity, intelligence, and coordination, and their security and trustworthiness issues have become increasingly prominent. This paper first analyzes and discusses security and trusted computing frameworks and technologies tailored for embedded systems. It then systematically reviews, compares, and critically analyzes theoretical and technological advances for enhanc-

文章引用: 张凯龙, 马萌旭, 何周灿, 樊弈辰, 周世钰, 闫琳珊. 嵌入式系统安全与可信技术发展趋势[J]. 嵌入式技术与智能系统, 2025, 2(4): 268-275. DOI: 10.12677/etis.2025.24025

ing the security and trustworthiness of embedded systems from multiple perspectives—domestic and international, academic and industrial, and with respect to trust ontologies and computing architectures.

Keywords

Embedded Systems, Security, Trusted Computing, Architecture, Technologies, Trends

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

如我们所知, 嵌入式系统日益广泛应用于工业控制、智能家居、医疗设备、航空航天、智能交通等诸多领域。同时, 随着与物联网、人工智能等新兴技术的快速融合, 嵌入式系统应用场景和规模不断扩大且日益呈现出突出的网络化、智能化、分布化、复杂化发展趋势, 其安全性问题随之而来也日益突出。鉴于嵌入式系统连接信息、物理/社会的跨域融合特性, 一旦在信息域遭受攻击或是计算系统出现故障, 其信息域中产生的安全问题更会通过装备(如车辆、电力等实体)传递、放大到物理世界, 产生危及生命财产安全乃至国家社会稳定的灾难性后果。

近年来, 其已成为学术界和工业界关注的焦点[1]。据 WISE 统计, 2023 年全球嵌入式安全市场规模为 49.7 亿美元, 预计到 2032 年将达到 94 亿美元, 2025~2032 年的复合年增长率为 7.32% [2]。根据 QYResearch 预测, 2030 年全球可信计算芯片市场销售额预计将达到 22.9 亿元, 年复合增长率为 5.2% (2024~2030) [3]。

2. 嵌入式系统可信体系分析

图 1 所示为面向嵌入式系统的可信技术体系。该体系构建了一个从底层硬件根基到上层应用安全的垂直信任链条。其基础始于最下层的可信根, 该层内置了加密引擎和白名单管理等核心安全功能; 在此之上构建可信硬件层, 集成了量子安全引擎或 PUF 等物理级安全防护机制; 向上延伸至虚拟化与分区隔离层, 通过可信的虚拟机监视器(VMM)启动和管理机制, 确保不同计算环境之间的严格隔离; 进一步加固可信系统软件层, 包含安全的 BootLoader 引导程序等组件, 为上层运行提供可信软件基础; 最终抵达顶层的应用安全层, 实施如应用白名单等机制, 保障具体应用程序的可信执行。在整个可信技术体系中, 可信开发部署工具涵盖从可信根到软件层面, 提供安全的开发框架与工具链, 确保所有组件在开发阶段即内嵌安全能力, 并在部署时按预期运行。可信升级依赖可信根存储版本元数据, 并在升级后验证新组件的完整性, 保障系统可持续演进, 避免升级成为安全突破口。而可信启动则是自下而上逐步构建的, 可确保系统从启动伊始即处于已知可信状态, 抵御 Bootkit 等底层攻击。可信运行是在系统启动完成后, 通过技术手段保障系统在持续运行过程中始终处于安全可信状态的动态防护机制。

基于 TPM 芯片内置加密引擎实现关键数据加密防护, 通过系统/应用软件哈希值校验及权限管理机制, 在启动或运行前完成软件完整性验证, 例如操作系统内核哈希比对机制, 以此构建计算环境初始信任源, 确保仅合法未篡改软件及操作被允许执行, 为系统可信运行奠定基础[4] [5]。可信硬件借助量子密钥分发与 PUF 安全引擎, 前者实现通信加密, 后者通过设备唯一身份识别抵御物理层攻击与仿冒, 同时依托手机启动时系统镜像验证等安全引导机制, 保障硬件启动过程中加载软件的完整性, 有效解决硬件

层面加密通信、身份认证及启动安全问题，提升物理层抗攻击能力。虚拟化与分区隔离通过 VMware ESXi 等 VMM 可信启动机制，完成虚拟机监控程序完整性验证以保障启动安全，并实施云计算平台租户虚拟机资源隔离等可信管理措施，阻断恶意攻击与数据泄露的跨虚拟机传播路径，在多业务多用户场景下实现资源隔离与环境可信，确保虚拟化实体的独立性与安全性。可信系统软件层通过应用分区技术，实现功能模块安全隔离以阻断故障与攻击扩散链，同时依托安卓 SELinux 强制访问控制等安全内核机制，限制应用对系统资源的越权访问行为，全面强化应用隔离与内核安全，提升系统整体稳定性与防御能力。应用层安全体系采用企业 MDM 办公 APP 白名单审核机制，从源头限制非法应用运行，结合 TLS 加密交易数据传输等安全通信技术，构建覆盖访问控制、通信安全与新兴技术防御的立体防护网，直接保障用户业务操作的可信性与数据安全。

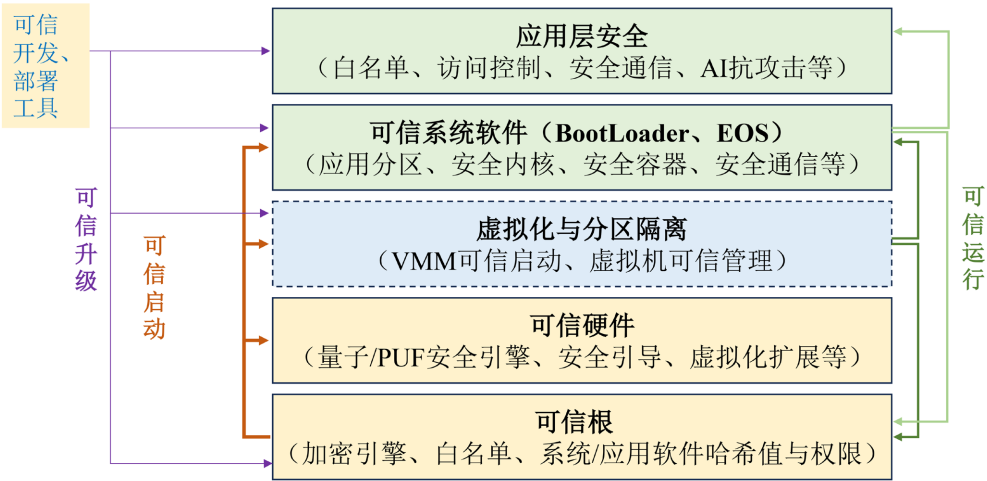


Figure 1. Architecture of trusted technology for embedded systems
图 1. 面向嵌入式系统的可信技术体系

以上策略为嵌入式系统安全提供了较为全面的保障，然而，现有的嵌入式系统安全解决方案仍面临诸多挑战。(1) 硬件安全机制虽然能够提供底层保障，但在面对复杂的攻击手段时，仍存在被绕过的风险。(2) 软件层面的安全措施如实时操作系统和微服务架构虽然增强了系统的灵活性和可靠性，但其复杂性也带来了新的安全漏洞和管理难度。(3) 工具链虽然在开发效率和代码质量方面发挥了重要作用，但在安全漏洞检测和修复方面仍存在不足。(4) AI 时代，复杂神经网络模型的不可解释性为嵌入式设备带来新的安全隐患，如代码篡改难以追溯等。(5) 5G、边缘计算等新兴技术与嵌入式系统的融合，可能带来新的安全需求，如终端设备数量与攻击面的扩大、数据隐私保护、边缘计算中的数据完整性和一致性等等。为此，在可信 3.0 的基础上，面向嵌入式系统日益呈现的云-边-端协同形态，进一步融合多端协同、零信任和保密计算的新一代计算架构进化出新的可信 4.0 计算体系。其特质在于，强调在多云-边-端融合环境中，通过硬件隔离、持续认证、动态策略和去中心化信任等手段，实现跨域、全生命周期、分级渐进的安全可信保障。

3. 国内研究现状

近年来，随着相关政策、项目的不断推进落地，对国内嵌入式系统安全发展起到了不可忽略的积极作用。由北京航空航天大学牵头，联合复旦大学、中山大学等多家单位协同攻关，旨在突破国内嵌入式智能计算软件关键技术，推动国产化智能芯片软件栈的持续完善，支持国产智能芯片生态建设[6]。由北

京和利时系统工程有限公司、中国科学院沈阳自动化研究所联合承担的国家“十二五”863计划课题，针对工业测控系统的信息安全防护需求，突破可编程嵌入式电子设备开发与运行阶段的信息安全防护关键技术[7]。由华东师范大学牵头，与中国航发商发、中航工业615所、航天五院502所等单位合作，开发高安全嵌入式控制软件开发方法和支撑工具，成功应用于航空发动机控制软件研制、机载航空电子系统研制以及“嫦娥五号”探测器软件开发等重要任务[8]。此外，工业界也有一批嵌入式可信相关的军用、民用产品落地。2024年华北工控研发的国产化可信嵌入式主板，集成系统级安全机制(密码加速引擎、抗物理攻击)，适配统信UOS、银河麒麟等国产操作系统亮相军博会[9]。同年，国民技术研发并推出第四代可信计算芯片NS350系列，支持SM2/SM3/SM4国密算法，兼容TPM 2.0，通过商用密码认证，适用于工业计算和嵌入式系统[10]。

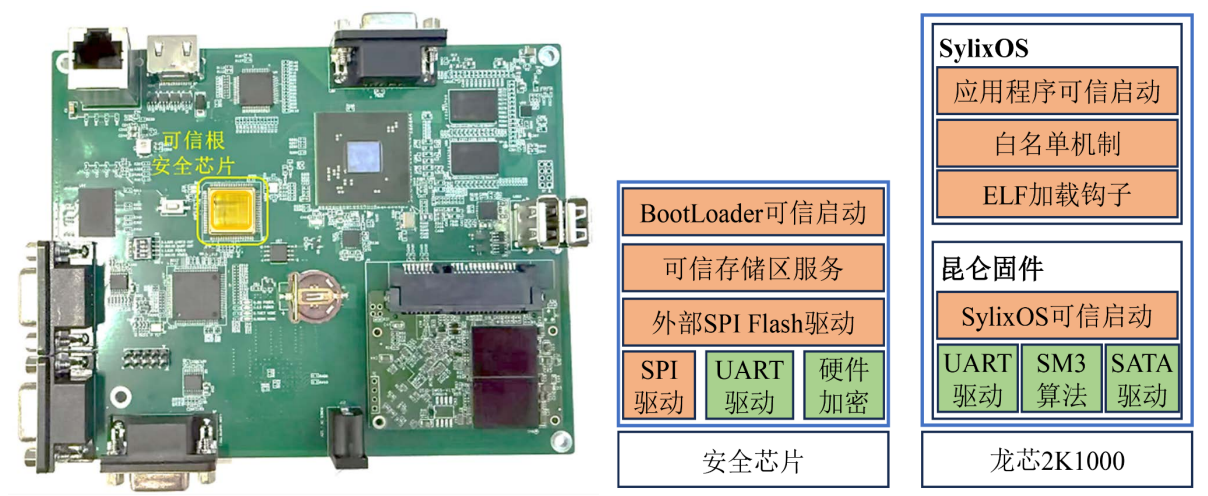


Figure 2. A trusted embedded platform developed by the authors’ team on a domestically-developed software-hardware stack
图 2. 作者团队基于国产软硬件技术栈研制的可信嵌入式平台

针对嵌入式设备的安全启动需求，Zhao 等人[11]设计了一种轻量级的基于安全模块的可信启动方法，通过在嵌入式设备启动前验证预设信息，确保设备按照预定方式启动，并利用安全模块对启动过程中的数据完整性进行度量，从而实现可信嵌入式系统。为保证电力系统嵌入式装置能够安全可信启动，张翔等人[12]提出了一种基于可信平台控制模块(TPCM)的轻量型可信启动方法。Liu 等人[13]提出了一种将区块链的链上信任扩展到链下物理世界的框架，以可信疫苗追溯为例进行了实现，包含基于可信执行环境(TEE)的可信环境监测系统和一致性协议两部分。Ma 等人[14]从高可信嵌入式系统、人工智能嵌入式模型、智能芯片嵌入式操作系统三个方面展开，通过将区块链共识机制与特定方法结合起来提升系统性能，设计异构计算平台和混合嵌入式训练系统，利用智能芯片增强功能并完成相关硬件设计。在基于量子计算的加密方法上，Yu 等人[15]研究了部分可信中继量子密钥分发(QKD)网络中的密钥分发路由问题，提出了一种基于协作路由的秘密密钥分发(SKP-CR)算法。Wang 等人[16]提出一种面向无 TPM 嵌入式智能设备的可信启动模型，以解决此类设备的安全启动及系统执行环境可信性问题。在航空航天领域，为了提升机载嵌入式计算机的安全防护能力，杨子怡等人[17]在将可信计算平台引入到机载计算机的基础上，设计并实现了基于机载嵌入式可信计算平台的安全配置管理系统。Yang 等人[18]提出基于区块链的 BC-UTSON 机制，通过 U-PBFT 共识、BMWSL 信任评估和 TPDR 可信路由技术，构建无人机群可信自组织网络以抵御内部恶意攻击并提升安全性。此外，基于可信 3.0 思想，作者团队面向特种装备可信需求，基于“自主可信根 + 龙芯 + 昆仑固件 + SylixOS”的全自主技术栈，研究并设计了一种基于双芯片架构的

可信执行环境,实现了可信启动、可信加载、可信运行、可信升级等核心能力,如图2所示。针对嵌入式系统混合部署与功能安全需求,何瑞琦等人[19]提出 DHR-OS 架构,在多核 CPU 上以 Linux 为主系统、动态部署 RTOS 从系统,利用 OpenAMP 实现通信及驱动复用、RPC 调用、中断路由等协同机制;设计调度-分发-裁决一体的安全执行机制,通过 Linux 池化 RTOS 核心、加权投票共识算法裁决任务结果。基于飞腾 D2000 的测试表明,该架构在差模/多模攻击下可靠性高,为嵌入式系统提供了兼具灵活性与抗攻击性的新方案。吉晨等人[20]提出基于轻量级虚拟化环境的可信多级安全容器机制,划分系统安全域并制定多级安全策略规则,通过形式化方法证明其安全性,利用联合文件系统技术和以 Docker 为代表的容器技术说明技术可行性,从来源和运行两方面保证可信性,可改善传统多级安全机制实用性差的问题。

4. 国外研究现状

STMicroelectronics 于 2023 年 9 月推出了一款新的安全微控制器,具备高级加密和认证功能,可有效保护物联网设备的安全[21]。一年后,该公司基于零信任安全模型,推出 STM32L5 系列 MCU,支持安全启动、硬件加密,功耗低至 33 nA [22]。Infineon Technologies AG 在 2023 年 2 月推出了 SECORA Connect 产品组合,包含小型芯片,适用于多种嵌入式设备[23]。此外,英特尔的 Gaudi 3 AI 加速器、至强 6 处理器液冷方案、酷睿 Ultra 系列均支持硬件级隔离(鲲鹏 TEE)、机密计算[24]。

Bognar 等人[25]以 SancusV 和 VRASED 系统为例,讨论了在嵌入式可信执行架构中,形式化方法证明的安全性与实际系统安全性之间的差距。Aaraj 等人[26]研究了在资源受限的嵌入式系统中实现 TPM 的硬件/软件协同设计,提出了一种基于软件的 TPM (SW-TPM)实现方案,通过在嵌入式处理器上执行受保护的代码域来实现 TPM 功能。Fedorov 等人[27]提出了一种基于隐藏软件代理和隐写术的方法,用于构建可信环境并保护信息系统免受内部攻击。在医疗设备的嵌入式系统可信认证方面,Gebreab 等人[28]提出了一种基于同质化代币(NFT)的解决方案,用于确保翻新医疗器械的可信追溯和认证。该方案利用动态可组合的 NFT 作为医疗器械及其翻新过程的数字表示,通过将替换部件和认证文件嵌入到父子 NFT 层次结构中,并通过动态令牌的演变记录翻新步骤,从而实现对翻新医疗器械的认证、追踪和所有权管理。在新一代量子计算方面,Trochatos 等人[29]提出了一种量子计算机可信执行环境(QC-TEE)的硬件架构,旨在保护用户量子电路和数据免受诚实但好奇的云服务提供商的窥探。该架构通过在用户端软件添加诱饵控制脉冲混淆真实量子门操作,在量子计算机端的稀释制冷机内使用简单的 RF 开关衰减诱饵脉冲,并由硬件安全管理器控制开关。Phalak 等人[30]提出两种量子物理不可克隆函数(QuPUF),以解决嵌入式设备量子计算中基于云平台的量子硬件安全与信任问题。面对 AI 时代对嵌入式系统安全冲击,Seng 等人[31]围绕嵌入式智能(EI)展开全面研究,指出其在安全、隐私和信任方面存在显著挑战,EI 服务器加速器和边缘设备在安全需求上可能存在差异。基于人工智能物联网(AIoT)的概念,Alkhoori 等人[32]讨论了 AI 决策与物联网设备结合的安全性挑战,被篡改的深度学习算法可能会操控 AIoT 的运行,进而威胁到整个系统的安全态势。此外,由于设备间持续进行通信和数据共享,保护这些信息免受泄露、篡改或中断至关重要。Raja 等人[33]提出 Secured UAV 模型,以 UAV 位置为输入、借助集中控制器形成无线网状网络,利用 A*搜索算法实现高效通信,并运用高级加密标准和 Blowfish 等加密技术及安全认证机制,保障多无人机通信的可信性与安全性,有效应对各类安全攻击。Nawshin 等人[34]提出 DP-RFECV-FNN 方法,将差分隐私与前馈神经网络结合,用于安卓恶意软件检测,遵循零信任安全模型对应用严格验证,在保障用户数据隐私的同时实现对已知和新型恶意软件的准确检测。Holmes 等人[35]提出了 SEVeriFast,一种针对 AMD SEV 微虚拟机的新型引导方案,其通过引入最小引导验证器、利用内核压缩减少测量开销、优化预加密等方式,在保证通过硬件强制信任根建立信任、借助测量直接启动和远程认证确保 VM 初始化完整性的前提下,将 SEV VM 冷启动性能提升 86%~93%。

5. 国内外研究现状对比

通过对国内外研究现状的系统性梳理可知,国内研究以国家重大科技专项为牵引,聚焦自主可控技术体系构建,重点开展国产操作系统与国密算法的适配应用研究。在研究模式上,强调高校与军工、航天等单位的产学研协同,围绕高安全等级嵌入式软件开发、可信启动机制、双芯片异构架构等关键技术展开攻关,研究成果直接服务于国防安全与关键信息基础设施建设。国外研究则呈现企业主导的技术创新特征,侧重集成硬件级安全模块的系统性解决方案研发,致力于构建零信任安全模型。其研究热点集中于形式化方法验证、量子计算可信执行环境架构设计、区块链技术与物联网融合应用等领域,注重技术创新与产业应用的深度耦合。

综合调研显示,当前国内外研究均在量子安全技术体系、人工智能与嵌入式系统安全融合、硬件可信根技术以及零信任安全架构等方向加速战略布局,体现出技术交叉融合与场景化应用的显著趋势。

6. 发展趋势与展望

从国内外研究动态来看,嵌入式可信安全领域正呈现技术深度融合与场景化需求驱动的发展趋势。在技术层面,量子安全技术(如量子密钥分发、量子可信执行环境)与传统可信计算体系加速融合,推动国密算法体系向量子抗毁方向升级;AI与区块链技术赋能嵌入式可信架构,通过联邦学习实现设备间安全协同与未知攻击检测,基于区块链的链上链下信任传递机制(如可信疫苗追溯、无人机群自组织网络认证)逐步落地;硬件级可信根(国产可信计算芯片、集成安全模块的MCU)与轻量级软件可信链深度协同,结合形式化验证方法构建资源受限环境下的全栈可信执行环境,成为高安全嵌入式系统(如航空发动机控制软件、工业测控设备)的核心支撑。在应用层面,边缘计算与物联网的泛在化催生轻量化可信认证需求,无人机群、智能医疗设备等场景推动分布式信任架构研发;国防军工、航天装备领域则加速自主可控可信技术栈的生态构建,聚焦异构架构、可信启动机制等关键技术的工程化应用。

展望未来,嵌入式可信安全研究一方面需突破国产软硬件协同优化瓶颈,建立从芯片级可信根到应用级可信开发部署工具的全流程体系,完善适应高安全场景的可信技术标准;在另一方面,针对量子计算威胁、资源受限设备安全效率平衡等挑战,需加强跨学科交叉创新,推动零信任模型与动态信任评估技术的轻量化改造,构建覆盖设备启动、运行、交互全生命周期的嵌入式可信安全体系,为国防安全、关键基础设施等领域提供坚实的可信保障。

参考文献

- [1] 赵波,倪明涛,石源,等. 嵌入式系统安全综述[J]. 武汉大学学报(理学版), 2018, 64(2): 95-108.
- [2] WISE GUY. 全球嵌入式安全市场研究报告[EB/OL]. <https://www.wiseguyreports.com/cn/reports/embed-ded-security-market>, 2023-09-01.
- [3] QYResearch. 可信计算芯片行业总体规模、市场占有率排名报告 2025 [EB/OL]. <https://www.gelonghui.com/p/1576665>, 2025-01-08.
- [4] Morris, T. (2024) Trusted Platform Module. In: Jajodia, S., Samarati, P. and Yung, M., Eds., *Encyclopedia of Cryptography, Security and Privacy*, Springer, 1-5. https://doi.org/10.1007/978-3-642-27739-9_796-2
- [5] Perez, R., Sailer, R. and van Doorn, L. (2006) vTPM: Virtualizing the Trusted Platform Module. *Proceedings of the 15th conference on USENIX Security Symposium*, San Jose, 31 July-4 August 2006, 305-320.
- [6] 北京航空航天大学. 北航牵头的国家重点研发计划“高安全强实时嵌入式智能软件系统”项目启动暨实施方案论证会顺利召开[EB/OL]. <https://scse.buaa.edu.cn/info/1092/11559.htm>, 2024-03-29.
- [7] 中国科学院沈阳自动化研究所. 国家 863 课题“可编程嵌入式电子设备安全防护技术”通过技术验收[EB/OL]. https://sia.cas.cn/xwzx/kydt/201810/t20181031_5152011.html, 2024-10-24.
- [8] 华东师大软件工程学院. 喜报! 华东师大牵头研发项目获上海市技术发明一等奖保障关键设备安全可信

- [EB/OL]. <https://sei.ecnu.edu.cn/da/cf/c33170a645839/page.htm>, 2024-10-23.
- [9] 深圳华北工控股份有限公司. 第十届中国(北京)军事智能技术装备博览[EB/OL]. <https://m.gkong.com/news/121541.html>, 2025-05-12.
- [10] Nations 加油站. 国民技术第四代可信计算芯片 NS350 正式投入量产[EB/OL]. <https://mcu.eetrend.com/content/2024/100580225.html>, 2024-10-05.
- [11] Zhao, H., Xu, C. and Zhou, F. (2021) Research on Embedded Startup Method of Trusted Module. 2021 *IEEE 5th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, Xi'an, 15-17 October 2021, 953-957. <https://doi.org/10.1109/itnec52019.2021.9587096>
- [12] 张翔, 王元强, 聂云杰, 等. TPCM 的轻量型安全可信启动方法[J]. 单片机与嵌入式系统应用, 2023, 23(6): 36-39, 44.
- [13] Liu, C., Guo, H., Xu, M., Wang, S., Yu, D., Yu, J., et al. (2022) Extending On-Chain Trust to Off-Chain—Trustworthy Blockchain Data Collection Using Trusted Execution Environment (TEE). *IEEE Transactions on Computers*, **71**, 3268-3280. <https://doi.org/10.1109/tc.2022.3148379>
- [14] Ma, Q. (2022) Design of High-Confidence Embedded Operating System Based on Artificial Intelligence and Smart Chips. 2022 *Second International Conference on Artificial Intelligence and Smart Energy (ICAIS)*, Coimbatore, 23-25 February 2022, 58-62. <https://doi.org/10.1109/icaais53314.2022.9742917>
- [15] Yu, X., Liu, Y., Zou, X., Cao, Y., Zhao, Y., Nag, A., et al. (2022) Secret-Key Provisioning with Collaborative Routing in Partially-Trusted-Relay-Based Quantum-Key-Distribution-Secured Optical Networks. *Journal of Lightwave Technology*, **40**, 3530-3545. <https://doi.org/10.1109/jlt.2022.3153992>
- [16] Wang, R. and Yan, Y. (2022) A Novel Trusted Boot Model for Embedded Smart Device without TPM. 2022 *24th International Conference on Advanced Communication Technology (ICACT)*, PyeongChang Kwangwoon Do, 13-16 February 2022, 228-233. <https://doi.org/10.23919/icaict53585.2022.9728958>
- [17] 杨子怡, 李亚晖, 王中华, 等. 基于机载嵌入式可信计算平台的安全配置管理系统[J]. 航空计算技术, 2023, 53(2): 123-126.
- [18] Yang, J., Liu, X., Jiang, X., Zhang, Y., Chen, S. and He, H. (2023) Toward Trusted Unmanned Aerial Vehicle Swarm Networks: A Blockchain-Based Approach. *IEEE Vehicular Technology Magazine*, **18**, 98-108. <https://doi.org/10.1109/mvt.2023.3242834>
- [19] 何瑞琦, 张凯龙, 吴金飞, 等. 基于多核异构操作系统的动态冗余可靠机制研究[J]. 计算机科学, 2025, 52(4): 33-39.
- [20] 吉晨, 石勇, 戴明, 等. 基于轻量级虚拟化环境的可信多级安全容器机制[J]. 计算机应用研究, 2017, 34(6): 1770-1773.
- [21] Marketsandmarkets. (2023) Embedded Security Market Size & Trends. <https://www.grandviewresearch.com/industry-analysis/embedded-security-market-report>
- [22] STMicroelectronics (2025) STM32L5 Series. <https://www.st.com/en/microcontrollers-microprocessors/stm32l5-series.html>
- [23] Business & Financial Press (2023) Infineon's SECORA™ Connect Can Make Anything a Wallet; New Technologies Will Make Contactless Payment Easier in the Future. <https://www.infineon.com/press-release/2023/infxx202302-068>
- [24] Intel (2025) Intel Newsroom: Corporate. <https://www.intel.cn/content/www/cn/zh/newsroom/corporate.html>
- [25] Boggar, M., Van Bulck, J. and Piessens, F. (2022) Mind the Gap: Studying the Insecurity of Provably Secure Embedded Trusted Execution Architectures. 2022 *IEEE Symposium on Security and Privacy (SP)*, San Francisco, 22-26 May 2022, 1638-1655. <https://doi.org/10.1109/sp46214.2022.9833735>
- [26] Aaraj, N., Raghunathan, A. and Jha, N.K. (2008) Analysis and Design of a Hardware/Software Trusted Platform Module for Embedded Systems. *ACM Transactions on Embedded Computing Systems*, **8**, 1-31. <https://doi.org/10.1145/1457246.1457254>
- [27] Fedorov, V.K., Balenko, E.G., Shterenberg, S.I. and Krasov, A.V. (2021) Development of a Method for Building a Trusted Environment by Using Hidden Software Agent Steganography. *Journal of Physics: Conference Series*, **2096**, Article ID: 012047. <https://doi.org/10.1088/1742-6596/2096/1/012047>
- [28] Gebreab, S.A., Salah, K., Jayaraman, R. and Zemerly, J. (2023) Trusted Traceability and Certification of Refurbished Medical Devices Using Dynamic Composable NFTs. *IEEE Access*, **11**, 30373-30389. <https://doi.org/10.1109/access.2023.3261555>
- [29] Trochatos, T., Xu, C., Deshpande, S., et al. (2023) Hardware Architecture for a Quantum Computer Trusted Execution Environment. arXiv: 2308.03897.
- [30] Phalak, K., Saki, A.A., Alam, M., Topaloglu, R.O. and Ghosh, S. (2021) Quantum PUF for Security and Trust in Quan-

-
- tum Computing. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems*, **11**, 333-342. <https://doi.org/10.1109/jetcas.2021.3077024>
- [31] Seng, K.P. and Ang, L. (2022) Embedded Intelligence: State-Of-The-Art and Research Challenges. *IEEE Access*, **10**, 59236-59258. <https://doi.org/10.1109/access.2022.3175574>
- [32] Alkhoori, A., Alkhoori, A., Alkhoori, A. and Ahmed, O. (2024) Security and Reliability Concerns of AI on Critical Embedded Systems. In: Rasheed, J., Abu-Mahfouz, A.M. and Fahim, M., Eds., *Forthcoming Networks and Sustainability in the AIoT Era*, Springer, 32-45. https://doi.org/10.1007/978-3-031-62871-9_4
- [33] Raja, G., Anbalagan, S., Ganapathisubramaniyan, A., Selvakumar, M.S., Bashir, A.K. and Mumtaz, S. (2021) Efficient and Secured Swarm Pattern Multi-UAV Communication. *IEEE Transactions on Vehicular Technology*, **70**, 7050-7058. <https://doi.org/10.1109/tvt.2021.3082308>
- [34] Nawshin, F., Unal, D., Hammoudeh, M. and Suganthan, P.N. (2024) AI-Powered Malware Detection with Differential Privacy for Zero Trust Security in Internet of Things Networks. *Ad Hoc Networks*, **161**, Article ID: 103523. <https://doi.org/10.1016/j.adhoc.2024.103523>
- [35] Holmes, B., Waterman, J. and Williams, D. (2024) SEVeriFast: Minimizing the Root of Trust for Fast Startup of SEV MicroVMs. *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2*, La Jolla, 27 April-1 May 2024, 1045-1060. <https://doi.org/10.1145/3620665.3640424>

嵌入式系统产业现状与发展趋势

何小庆^{1,2}

¹嵌入式系统联谊会, 北京

²麦克泰软件技术公司, 北京

收稿日期: 2025年10月28日; 录用日期: 2025年11月20日; 发布日期: 2025年12月1日

摘要

嵌入式系统从过去孤立、功能单一的“专用计算机”, 演变为智能化、网络化、集成化复杂的自治计算系统。嵌入式系统成为物联网、工业4.0、人工智能、自动驾驶等所有前沿科技的基石, 作为物理世界与数字世界的接口, 嵌入式系统的重要性将与日俱增。文章系统梳理了国外和中国嵌入式系统产业几大核心板块的现状和发展趋势, 包括嵌入式处理器、嵌入式操作系统、编程语言以及产业生态。文章对中外嵌入式系统产业特点进行深入分析, 最后介绍最新嵌入式领域相关的产业政策, 包括具身智能, 论文的目标是给产业界在嵌入式系统发展方向和研发方法上带来一些启迪。

关键词

嵌入式系统, 嵌入式实时操作系统, SoC, MCU, 物联网, 人工智能

Current Situation and Development Trends of the Embedded Systems Industry

Xiaoqing He^{1,2}

¹Embedded System Association, Beijing

²BMR Technology, Beijing

Received: October 28, 2025; accepted: November 20, 2025; published: December 1, 2025

Abstract

Embedded systems have evolved from isolated “dedicated computers” with single functions in the past into complex autonomous computing systems featuring intelligence, networking, and integration. As the cornerstone of all cutting-edge technologies such as the Internet of Things (IoT), Industry 4.0, artificial intelligence (AI), and autonomous driving, embedded systems serve as the interface between the physical world and the digital world, and their importance will grow with each

passing day. This article systematically sorts out the current status and development trends of several core segments in the embedded systems industry, both in China and abroad, including embedded processors, embedded operating systems, programming languages, and the industrial ecosystem. It conducts an in-depth analysis of various characteristics of the embedded systems industry in China and abroad, and finally introduces the latest industrial policies related to the embedded field, including those concerning embodied intelligence. The objective of this paper is to provide the industry with insights into the development direction and R&D methods of embedded systems.

Keywords

Embedded System, Embedded Real-Time Operating System, SoC, MCU, Internet of Things, Artificial Intelligence

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

1. 引言

单片机(MCU)和微处理器自 70 年代在欧美开始流行, 80 年代初进入中国。80 年代北京工业大学电子厂开发了 TP801, 上海和江苏等地高校和企业开发了 MCS-48/MCS-51 的单片机开发系统, 一场“单片机”的热潮席卷大地, 这股浪潮随后引发了全国的智能电子大变革的时代的到来。1986 年 10 月在上海复旦大学举行了首届全国单片机学术交流会, 它标志了中国单片机产业的开始。2000 年 10 月首届嵌入式系统研讨会(后改名为 CCF 嵌入式系统大会)在北京召开, 标志着中国单片机应用发展到嵌入式系统新的时代。以 ARM 架构为核心的嵌入式处理器开始了广泛的应用, 高校纷纷开设 ARM + Linux (或者 ARM + $\mu\text{C}/\text{OS}$)嵌入式系统课程, 以北航出版社为代表, 大量嵌入式系统图书出版发行, 嵌入式技术也成为国家重视的关键技术之一[1]。

物联网(Internet of Things, IoT)这一概念最早由海外学者提出, 在中国, 2009 年 8 月, 时任国务院总理温家宝视察中科院无锡微纳传感网工程技术研发中心时指出: “在传感网发展中, 要早一点谋划未来, 早一点攻破核心技术, 尽快建立‘感知中国’中心。” 2009 年被后人称为“中国物联网元年”, 之后中国物联网产业走上了蓬勃发展之路, 嵌入式系统从孤立部件走向感知、计算和通信为一体的智能硬件[2]。

工业物联网(IIoT)是指在制造业、能源管理等工业环境中, 由相互连接的传感器、机器和系统构成的网络。它能够采集、监控并分析生产运营中的数据, 目标在于提升效率、降低成本并增强安全性。工业物联网备受关注, 但尚处于发展初期。根据行业报告和专家分析, 全面 IIoT 化系统只有 10%~15%, 现代化工业设备需要具备远程监控、预测性维护、云边缘数据分析及实时控制能力, 实时和安全的嵌入式系统在这个领域将大有作为[3]。

工业 4.0, 常被称为第四次工业革命, 代表了当前制造业和工业流程数字化转型的新阶段, 其特征是将工业物联网、人工智能(Artificial Intelligence, AI)和自动化等技术深度融合。智能工厂是应用工业 4.0 技术的具体案例, 旨在打造高度互联且自动化的生产环境。工业 4.0 通过信息物理系统(CPS)实现设备、生产线与网络的互联, 而嵌入式系统是 CPS 的底层技术支撑。支持复杂算法和大系统集成, 满足工业 4.0 对数据处理和智能计算的新需求, 嵌入式边缘计算系统在工业 4.0 时代正在发展重要作用。

随着人工智能(Artificial Intelligence, AI), 尤其是大语言模型(Large Language Model, LLM)应用的持续深入, 在物联网和边缘计算广泛普及的加持下, 嵌入式系统将更有效地整合 AI 技术。包括在嵌入式设备

上实现复杂的 AI 算法和模型推理，以及提供更智能、更加自适应的系统功能和服务，例如自动驾驶汽车、智能机器人和智慧医疗等智能应用。在智能制造领域，AI 应用与传统机械设备结合可替代人工值守，降低工业制造的成本，例如故障检测等典型场景。AI 应用在现代战争中的地位变得举足轻重，例如无人机和无人飞艇等场景。

嵌入式系统范围广泛，包括芯片、组件、软件、工具、服务、系统和生态等方方面面，核心产业包括嵌入式处理器芯片、嵌入式操作系统与中间件、编程语言以及产业生态等若干方面。以下我们围绕产业现状、国内外对比和趋势展望三个部分展开讨论。

2. 国外嵌入式系统产业现状

今天，嵌入式处理器形成以 SoC 应用处理器和 MCU 为主流[4]，其他处理器类型，比如 FPGA/DSP 为辅助，GPU/xPU AI 训练和推理处理器为支撑的新格局。在处理器微架构上异构多核处理器渐成主导力量，Arm 依然占据嵌入式处理器指令集架构的核心地位，RISC-V 在物联网领域专用架构(DSA)市场有一定斩获。市场研究机构 Counterpoint Research 预测：2030 年 RISC-V 芯片出货量将突破 160 亿颗；RISC-V 在 IoT 市场的出货量将增至 108 亿颗，全球汽车 RISC-V 芯片预计出货量将突破 25 亿颗[5]。

近年，边缘和端侧 AI 处理器芯片是嵌入式系统研究和产品开发的热点，市场上有代表性的是 NVIDIA Jetson 和 Qualcomm Snapdragon 系列，已有大量应用成果。在端侧 MCU 领域，以意法半导体 STM32N8，瑞萨 RA8 和 NXP MCXN947 为代表的 AI MCU 具有市场影响力，但未见有知名品牌量产应用。

嵌入式操作系统是嵌入式系统的基石，是工业软件的基础平台。在市场占有率上，Eclipse 基金会 2024 年物联网开发者调查表明，资源受限设备上的开发人员嵌入式操作系统首选排在前三位分别是：Linux 49%、FreeRTOS 29%和 Zephyr 21%，其中 Zephyr 从 2023 年的 13%增长到了 21%，进步显著[6]，见下图 1。

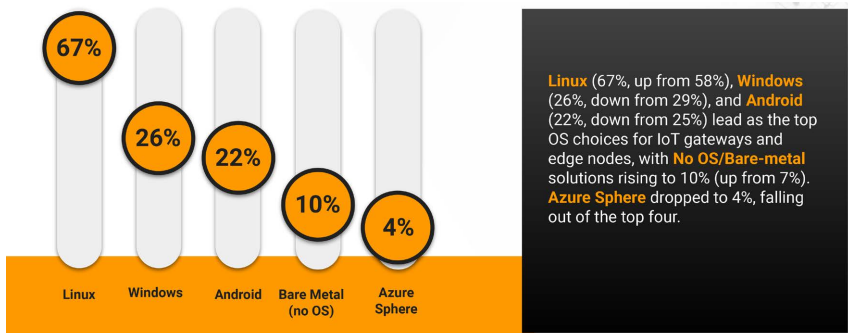


Figure 1. 2024 IoT & embedded developer survey report (source: Eclipse Foundation)

图 1. 2024 年物联网与嵌入式开发者调查报告(来自：Eclipse 基金会)

通用人工智能(Artificial General Intelligence, AGI)兴起，推动嵌入式系统智能化快速发展的进程，智能嵌入式系统面临急需提升基础算力、构建更加复杂应用系统，同时满足智能化及实时性挑战。应对挑战，嵌入式处理器正在广泛采用异构多核高算力芯片，通过多内核混合部署实现大小脑双系统，支撑包括具身智能机器人的智能系统应用。混合关键系统部署方案，底座采用虚拟化技术。虚拟化技术可以支撑多个操作系统运行在同一个物理处理器上，例如可以在同一个处理器芯片上同时运行高实时性操作系统(如 FreeRTOS)和通用的操作系统(如 Linux)。虚拟化技术是今天多核芯片采用的全新解决方案，它平衡了通用性与可靠性两方面需求(实现处理器核和 OS 绑定后的隔离)。我们常见的嵌入式虚拟化的典型应用可以在智能汽车电子系统中找到，比如在一个智能座舱系统中同时运行高实时性操作系统和人机交互操作系统(比如 Android)，来自加拿大黑莓旗下的 QNX 虚拟化软件在汽车电子中应用已经许多年了，具有

非常坚实的市场地位[7]。

嵌入式开发工具正从传统的 IDE 朝着云原生和 AI 增强型平台方向发展,例如, CMake 和 Ninja 的组合逐渐取代传统 Make 工具;云原生开发解决方案在嵌入式领域逐渐普及,推动低代码编译流程。嵌入式开发团队正在快速采用 DevOps 实践,以实现自动化构建、测试和部署。开发工具需要支持 CI/CD 流程,提高开发效率和产品质量。市面上已经出现了整合云和 CI/CD 的嵌入式开发平台,如 Wind River Studio,这是一个云原生工具集,用于开发、部署、运营和服务关键任务智能系统的工具。在 AI 开发工具方面,基于大型语言模型(LLM)的智能编程助手如 Copilot、Cursor、Codex 在软件开发(包括嵌入式)中广泛应用。

3. 中国嵌入式系统产业现状

通用 MCU 领域,以 Arm Cortex-M 为代表的 MCU 芯片是市场主流的芯片架构,国产 MCU 在性价比、解决方案和厂商技术支持方面有优势。国产 MCU 核心技术来自 Arm 授权,工具依赖第三方和开源软件,同质化严重。国产 MCU 在走过了国产替代的高峰期之后,再次步入行业竞争加剧的阶段。在嵌入式处理器架构种类方面,国内有一定的多样性,国产龙芯架构的处理器芯片在嵌入式领域有一定的市场份额。Arm 和 x86 依然主导国内微处理器市场和产业生态,中美科技竞争加速中国芯片企业采用 RISC-V 替代 Arm 架构。凭借开源开放、精简灵活的商业模式和技术特点上的优势,RISC-V 的崛起在中国是大势所趋。

边缘和端侧 AI 处理器芯片,国产瑞芯微 RK3568/RK3588、华为昇腾 310 和 910 在国内亦有一定市场份额,全志异构多核芯片 T113/T157 将 Arm, RISC-V 和 DSP 三个核融合在一颗芯片里面,技术上颇有特色,受到开发者关注。国产端侧 AI 处理器芯片在中低端的价格和生态上有优势。在端侧 MCU 领域,国内乐鑫科技 ESP32-S3 在开发者社区和高校学生间有很高的知名度,乐鑫最新推出了基于 RISC-V 架构的双核微控制器芯片 ESP32-P4,该芯片是乐鑫首款不带无线功能,支持 AI 指令扩展的边缘 AI 芯片。在通用 MCU 领域,国内企业在端侧 AI 方面未见很大的动作,仍处于观望阶段。以知存科技为代表的存算一体的芯片研究取得许多成果,生态建设持续发展,有望在可穿戴和 AI 服务器市场落地应用。

嵌入式操作系统领域,国内以 openEuler 和 OpenHarmony 为代表的开源操作系统蓬勃发展,广泛应用于各类嵌入式系统之中,传统的开源嵌入式实时操作系统,比如 RT-Thread 在物联网领域颇具生态优势。openEuler Embedded 旨在推进面向多核的 SoC 混合关键系统部署,通过嵌入式虚拟化技术实现多 OS 之间的隔离与保护,借助使用 Jailhouse 和 ZVM 虚拟机可支持部署多个操作系统,该方案聚焦智能工业领域,但成果尚待验证[8][9],见下图 2。



Figure 2. Mixed critical system architecture based on openEuler Embedded

图 2. 基于 openEuler Embedded 混合关键性系统架构

在嵌入式系统编程语言方面，2023 年嵌入式开发状态市场调查显示，C/C++占比 70%，依然是绝对的优势[10]，见下图 3，国内情况与全球主流市场基本一致。Rust 正在受到广泛的关注，相对于传统嵌入式开发语言(比如 C/C++)Rust 有其优势，包括内存安全、并发支持、性能、代码可读性，和一个成长中的生态系统。Rust 在嵌入式开发中的使用率正在逐渐增加，特别是安全和可靠性优先的项目。清华大学和北航团队使用 Rust 开发 rCore OS 和虚拟机 Rust-Shyper，Rust 在 Linux 内核和系统开发中的应用正在逐步推进，尽管面临一些挑战，但其内存安全和并发处理的优势使其在嵌入式系统编程领域具有广阔的应用前景，手机公司比如 VIVO 在积极推进基于 Rust 的内核——BlueOS 的研发与应用，并将其开源给开发者社区[11]。

Software development requires more cycle time

“C” dominates other languages for embedded software programming

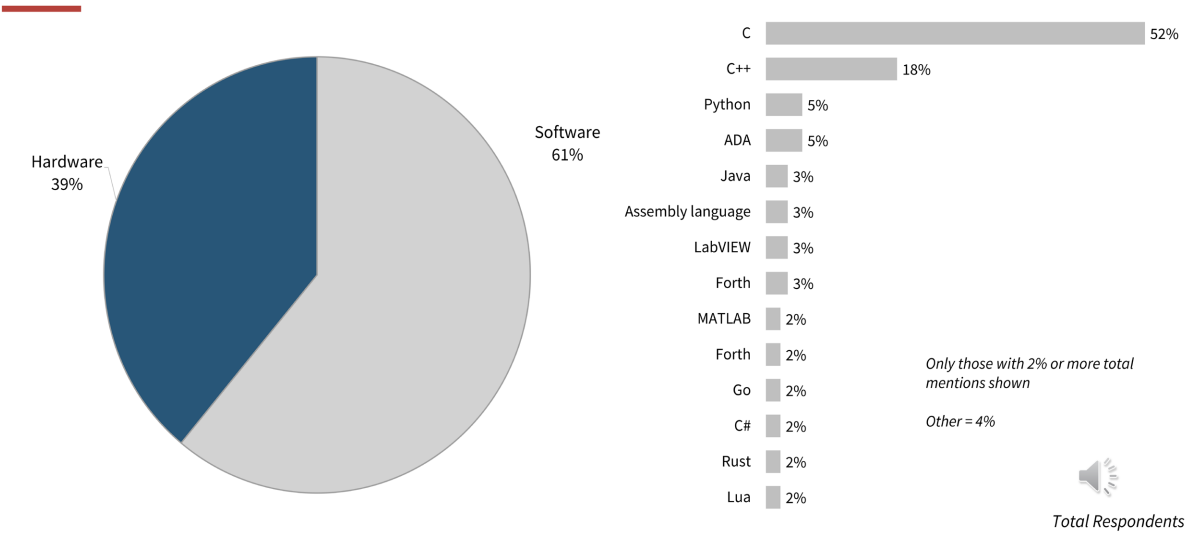


Figure 3. Market survey on the state of embedded development in 2023
图 3. 2023 年嵌入式开发状态市场调查

近年中国嵌入式系统产业的蓬勃发展得益于大疆、比亚迪、小米和宇树科技这样“新三样”产业领袖，它们在传统产业向新型产业转型中快速崛起，这里嵌入式技术起到重要推动作用，反过来新型产业也在倒逼嵌入式技术创新与发展。

4. 中外嵌入式系统产业分析和比较

嵌入式操作系统分为开源和商业两大阵营，以 Linux 和 FreeRTOS 为代表的开源阵营，以 QNX 和 Wind River 为代表的商业闭源阵营。还有一些企业既有开源版本也有商业版本，比如 QT 和 RT-Thread，不同阵营的企业在其特定市场上均有优势。以 SafeRTOS 为代表的预认证的安全操作系统发展迅速，国内锐华(ReWorks)、OneOS (中移物联网)和 RT-Thread 积极跟进。嵌入式操作系统依然处在百花齐放市场阶段，无法形成 PC 和手机 Windows 和 Android 各自一家独大的格局。多家商业巨头曾在物联网操作系统研发和推广中投入巨资，比如微软 Azure RTOS、Arm mbedOS、阿里(AliOS Things)和腾讯(TencentOS tiny)，最后许多是无果而终[12]。

国外的嵌入式操作系统企业经过 30 多年发展，已经形成成熟的产品定位和商业模式，比如 QNX 在汽车电子、INTEGRITY RTOS 在军工系统，VXWORKS 在通信和航空航天，国产嵌入式操作系统企业的

商业运作仍处在探索期，天脉(西安航空计算所)、翼辉和睿赛德等公司的产品初步形成自己的特点和垂直产业，开源鸿蒙正在形成行业应用，比如针对电网系统的电鸿操作系统，开源鸿蒙在政策的扶植下发展势头正旺。

国外在嵌入式开发工具方面处在领先地位，国内嵌入式企业积极跟进。比如翼辉新型开发环境 RealEvo-Stream，主要针对应用开发者，其功能偏向上层业务开发、应用适配以及生态软件移植。RealEvo-Stream 可支持多种常用开源生态工具链，可以让应用开发者高效地完成软件适配，提高应用开发效率[13]。

在处理器指令集架构的创新上，国外优势明显。无论是 Arm、Intel 以及 MIPS，还是今天的开放指令集架构 RISC-V，指令集架构核心技术均来自北美和欧洲。在 FPGA/DSP 以及 GPU/xPU AI 训练处理器方面，欧美具有先发的优势。在处理器微架构和 SoC 芯片设计方面，近年中国企业显现出积极发展势头。中国芯片企业依托制造产业优势，在面向行业应用的芯片设计上表现出色，比如地瓜机器人 SoC 算控一体化机器人芯片 RDK S100，该芯片整合 CPU + BPU + MCU，包括 6 核 Arm Cortex-A78AE CPU、80 TOPS BPU Nash 架构 AI 算力和 4 核 Arm Cortex-R52 MCU，支持端云一体开发，芯片市场定位在具身智能机器人领域。富瀚微电子的超低功耗的摄像机芯片 FH8852 和 AI 芯片 FH889X 在消费电子市场上有相当的影响力。

国外芯片公司在通用计算，GPU 和手机领域有压倒性优势和强大的市场占有率，嵌入式和物联网领域，STM32 MCU 以其丰富的功能、众多的封装以及完善的生态占据主导地位。截至 2024 年 12 月，STM32 的出货量已超过 110 亿片，服务的客户超过 20 万家[14]。凭借长期在 PC 市场积累的生态优势，x86 在工控控制器市场占有率依然很高。高通的智能手机芯片通过绑定 Android 生态，在嵌入式终端市场的占有率长期居高不下。

国产嵌入式应用处理器和 MCU 芯片在中低端消费终端应用和物联网 IoT 模组上，以性价比高的优势有着极高的市场占有率，比如，瑞芯微 RK3568 和 ESP32-S3。以华为昇腾 910 系列为代表的国产高性能 AI 训练和推理芯片，因为美国断供，英伟达、AMD 最高端的芯片买不到，国内很多厂商硬着头皮用国产芯片，经过一段时间的迭代，昇腾已经是国产芯片中的佼佼者，比如，宝德计算机 2024 年营收为 100 亿元，其中昇腾系列服务器业务贡献显著。作为华为昇腾领先级整机硬件伙伴，宝德 2024 年中标 50 亿元 AI 服务器的订单，占其总营收的 50%。综合下来，昇腾服务器在宝德计算机的业务占比为 40% [15]。以升腾、海思、海光和飞腾为代表的国产服务器芯片和以全志、瑞芯微和芯驰微为代表的端侧 SoC 芯片，它们在电网和工业机器人等国产大国重器项目中正在发展重要的作用[16]。

嵌入式世界是 RISC-V 的主战场，百万级应用已经落地，生态日臻成熟：从物联网、工业控制到智能汽车，RISC-V MCU 无处不在[17]。国内芯片企业在 RISC-V 投入与日俱增，更高性能的处理器持续涌现，高算力嵌入式 AI 应用蓄势待发，每年的 RISC-V 中国峰会已经成为产业界一道亮丽的风景线。

5. 总结与展望

嵌入式系统的发展需要产业政策的支持和助力。近年来，中国政府出台了一系列政策，积极推动边缘人工智能产业发展。在《“十四五”智能制造发展规划》中，将工业软件作为实现智能制造的关键技术之一，纳入重点支持范围，提出加快推动制造业向智能化、高端化、绿色化方向发展。地方政府的人工智能和物联网产业政策持续加码，比如《广州市推动物联网产业高质量发展行动计划(2024~2028 年)》，《湖南省“智赋万企”行动方案(2023~2025 年)》提出建设面向特定场景的边缘计算设施，推动边缘计算与云计算协同部署。《广东省关于人工智能赋能千行百业的若干措施》(2024 年 5 月)强调建立人工智能芯片生态体系，面向家电家居、安防监控、医疗设备等，加大高性能、低功耗的端侧芯片开发生产。2025 年《北京具身智能科技创新与产业培育行动计划(2025~2027 年)》中明确，到 2027 年，围绕具身大小脑

模型、具身智能芯片、全身运动控制等方面突破不少于 100 项关键技术, 产出不少于 10 项国际领先的软硬件产品[18]-[21]。

这些政策表明, 中国政府高度重视边缘人工智能产业的发展, 通过技术创新、基础设施建设、产业生态构建等多方面措施, 为嵌入式边缘智能系统产业的高质量发展提供了有力支持。

回顾四十年发展历程, 嵌入式系统从过去孤立、功能单一的“专用计算机”, 演变为智能化、网络化、集成化复杂的自治计算系统。嵌入式系统成为物联网、工业 4.0、人工智能、自动驾驶等所有前沿科技的基石, 作为物理世界与数字世界的接口, 嵌入式系统的重要性将与日俱增。

参考文献

- [1] 陈章龙. 中国单片机 20 年[J]. 单片机与嵌入式系统应用, 2007(12): 89-91, 94.
- [2] 万物赋能, 智领未来——世界物联网日大咖分享会[EB/OL].
https://jsnews.jschina.com.cn/zt2018/2018wiot/wlxf/201808/t20180816_1841802.shtml, 2018-08-16.
- [3] Maxfield, C.M. (2025) The IIoT and Industry 4.0—Connectivity and Intelligence Are Transforming Industry. *Designing Electronic Engineering North America*, 4, 30.
- [4] 何小庆, 何灵渊. 嵌入式软件开发的三个趋势[J]. 嵌入式技术与智能系统, 2024, 1(1): 1-10.
<https://doi.org/10.12677/etis.2024.11001>
- [5] 芯智讯到 2030 年全球 RISC-V 芯片出货量将超 160 亿颗! [EB/OL]. <https://www.icsmart.cn/69543/>, 2023-11-09.
- [6] Eclipse Foundation (2025) 2024 IoT & Embedded Developer Survey Report.
<https://outreach.eclipse.foundation/iot-embedded-developer-survey-2024-thank-you>
- [7] QNX (2025) Hypervisor 2.0.
<https://blackberry.qnx.com/content/dam/qnx/products/hypervisor/hypervisorGEM-ProductBrief.pdf>
- [8] 谢国琪, 熊程来. ZVM 开源文档[EB/OL]. <https://gitee.com/openeuler/zvm>, 2025-02-25.
- [9] OpenEuler 社区. MICA: 面向复杂嵌入式系统的混合关键性部署框架[EB/OL].
<https://blog.csdn.net/openEuler/article/details/128631236>, 2025-02-25.
- [10] Aspecore (2025) The Current State of Embedded Development.
<https://www.embedded.com/wp-content/uploads/2023/05/Embedded-Market-Study-For-Webinar-Recording-April-2023.pdf>
- [11] VIVO (2025) Blue OS. <https://github.com/vivoblueos/kernel>
- [12] 何小庆. 嵌入式实时操作系统的昨天、今天和明天[J]. 中国计算机学会通讯, 2023(2): 84-89.
- [13] 翼辉信息 RealEvo-Stream 助力软件高效迁移至 SylixOS [EB/OL].
<https://mp.weixin.qq.com/s/lgec-MMAG2sNuiXUW2UDOA>, 2025-07-01.
- [14] 意法半导体 MCU 双供应链策略, 打消中国客户后顾之忧[EB/OL].
<https://www.eeworld.com.cn/emp/STMChina/a392312.aspx>, 2025-01-16.
- [15] 华为昇腾营收占比 50% + 信创市占率第一[EB/OL]. <https://xueqiu.com/8607068205/336309830>, 2025-05-25.
- [16] 南方电网公司受邀与工信部共同见证鸿蒙 5.0 发布“电鸿”物联操控系统亮相备受关注[EB/OL].
<https://baijiahao.baidu.com/s?id=1821193072453076769&wfr=spider&for=pc>, 2024-12-30.
- [17] (2024) RISC-V Market Report: Application Forecasts in a Heterogeneous World.
<https://theshdgroup.com/wp-content/uploads/2024/01/RISC-V-Market-Analysis-2024-Abridged-Report-2.pdf>
- [18] 工业和信息化部. “十四五”智能制造发展规划[EB/OL].
https://www.gov.cn/zhengce/zhengceku/2021-12/28/content_5664996.htm, 2021-12-21.
- [19] 广州市工业和信息化局. 广州市推动物联网产业高质量发展行动计划(2024-2028 年) [EB/OL].
https://gxi.gz.gov.cn/yw/zchb/zcwj/cyzc/content/mpost_9817815.html, 2024-08-16.
- [20] 广东省人民政府办公厅. 广东省关于人工智能赋能千行百业的若干措施[EB/OL].
https://www.gd.gov.cn/zwgk/wjk/qbwj/ybh/content/post_4436455.html, 2024-06-06.
- [21] 北京市科学技术委员会. 北京具身智能科技创新与产业培育行动计划(2025-2027 年) [EB/OL].
https://www.beijing.gov.cn/zhengce/zhengcefaui/202503/t20250304_4024579.html, 2025-02-28.

卷后语——CCF嵌入式系统专委会发展规划

郭 兵^{1,2}

¹中国计算机学会嵌入式系统专业, 北京

²四川大学计算机学院, 四川 成都

Copyright © 2025 by author(s) and Hans Publishers Inc.

This work is licensed under the Creative Commons Attribution International License (CC BY 4.0).

<http://creativecommons.org/licenses/by/4.0/>



Open Access

CCF 嵌入式系统专业委员会成立于 1979 年, 作为 CCF 旗下立足于嵌入式系统技术领域的专业平台, 专委会以“聚焦价值、追求卓越、服务社会”为宗旨, 致力于服务嵌入式计算与系统技术的推动发展和领域专业人才的培养, 期望为国内外嵌入式系统及相关领域的专家、学者、工程师、技术管理者和研究生提供学术研讨和技术交流平台, 推动“政产学研用”合作, 引领技术进步、支撑产业发展。相关技术领域全面覆盖嵌入式微处理器、系统芯片及智能微系统、嵌入式系统设计、嵌入式软件工程、嵌入式系统应用技术等专业方向, 以及可信嵌入式计算、开源嵌入式软硬件、物联网技术、信息物理融合系统等前沿领域。专委会发展先后经历了学组、分会、专委会三个阶段, 其前身为微机专业委员会, 发展历程如图 1 所示。

1. 专委会沿革

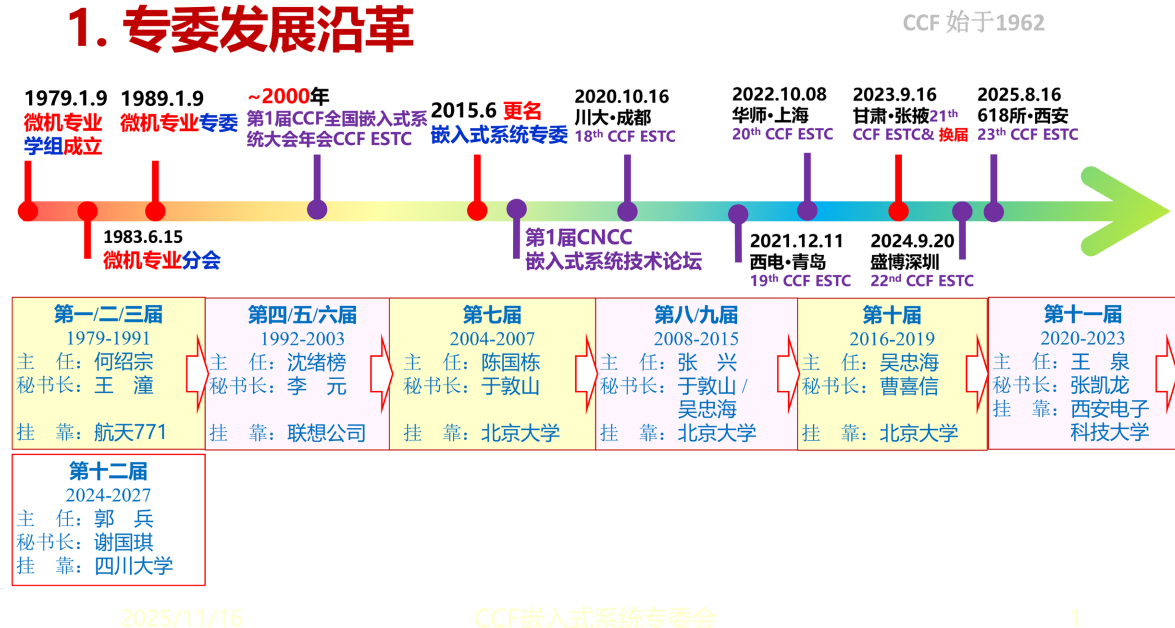


Figure 1. The evolution of the CCF technical committee on embedded systems

图 1. CCF 嵌入式系统专委会沿革

文章引用: 郭兵. 卷后语——CCF 嵌入式系统专委会发展规划[J]. 嵌入式技术与智能系统, 2025, 2(4): 283-284.

DOI: 10.12677/etis.2025.24027

目前,专委有执行委员 260 余人,其中高校 208 人、院所 30 人、企业 22 人。第 12 届专委主任由 CCF 理事、四川大学郭兵教授担任,副主任为东北大学邓庆绪教授、华东师范大学陈铭松教授、西北工业大学张凯龙教授,秘书长为湖南大学谢国琪教授。常务委员共 20 人,由专委领导班子和陈良银(副秘书长、四川大学)、杜军朝(西安电子科技大学)、江维(电子科技大学)、江建慧(同济大学)、卢冶(南开大学)、彭鑫(湖南理工大学)、谢勇(南京理工大学)、杨鹏飞(副秘书长、西安电子科技大学)、周全(华中科技大学)、周俊龙(副秘书长、南京理工大学)、崔西宁(中航 631 所)、何小庆(麦克泰软件)、芦潇静(北航出版社)、吴姣(中航 631 所)、赵勇(深圳盛博嵌入式计算机有限公司) 15 位组成。其中学术界 15 名、院所/企业 5 名。

专委围绕嵌入式系统领域开展学术、技术、走进高校等系列活动。其中,作为主体发起的学术会议——CCF 全国嵌入式系统大会已成功举办 20 届,形成本专委的品牌学术会议,主题涵盖“智能、安全、交叉、融合”、“芯智新时代、嵌入新机遇”等,并与软件学报、计算机学报、计算机研究与发展、JSA、IEEE TDC、JCSC 等国内外一流期刊展开专刊合作。2025 年度会议将由中航工业西安飞行自动控制研究所、西北工业大学、西安电子科技大学联合承办,于西安举办。作为主体,在中国计算机大会(CNCC)发起的嵌入式系统技术已举办多届,主题涵盖“智能嵌入式系统”、“交叉赋能:嵌入式遇上云原生”、“嵌入式赋能端云互融”等,邀请领域的顶级学者、青年新秀等进行主题报告。同时,专委积极组织 CCF 嵌入式系统专委走进高校、院所等活动,以专业的方式服务社会需求,受到广泛好评。

在未来,专委将进一步加强专业平台的服务、辐射作用,邀请国际、国内的领域同行参与专委建设并对外输出专委的智力资源与服务。同时,将进一步加强平台建设,发挥好合作舞台的作用,深度促进产学研用模式下的联合科研攻关、人才培养、交叉转化等合作,为国家、行业的技术创新、产业发展做出贡献。

欢迎加入 CCF 嵌入式系统专业委员会!



Call for Papers

Embedded Technology and Intelligent Systems

嵌入式技术与智能系统

国际中文期刊征文启事

<https://www.hanspub.org/journal/etis>

ISSN: 3065-1220

《嵌入式技术与智能系统》是一本开放获取、关注集成传统嵌入式技术与新兴智能系统的前沿研究最新进展的国际中文期刊，期刊特别注重软件算法、芯片设计与硬件实施的协同进展，以及理论研究与工程实践的紧密结合，面向学术界学者、产业界专家与工程师、学生及技术爱好者，关注中国领先产业集群的广阔发展潜力。本期刊强调发表原创性、创新性及具有实用价值的研究成果。该期刊由汉斯出版社出版，全球发行，现诚邀相关领域的学者投稿。

主编

何立民，北京航空航天大学教授

副主编

何小庆，嵌入式系统联谊会秘书长

吴薇，杭州电子科技大学特聘教授

投稿领域：

人工智能技术-边缘计算-端侧智能和大模型嵌入式应用
GPT-行业GPT以及GPT在嵌入式及智能系统研发中的应用
信息物理融合系统(CPS)-物联网技术-感知计算和无线传感网-泛在电力物联网-智能电表-储能技术-智能输变电
嵌入式系统结构-嵌入式操作系统与中间件-Linux、安卓和开源鸿蒙应用
实时操作系统-虚拟化和容器技术-混合关键系统
嵌入式软件形式化建模-软件测试和仿真-功能安全技术
嵌入式软件云原生技术-CI/CD和DevOpt-微服务
软硬件协同设计-开源指令集和开源芯片-RISC-V产业生态
嵌入式SoC技术--MCU 创新与生态-FPGA/DSP技术和应用
AI芯片和算法-存储技术-GPU技术-视觉芯片及嵌入式显控应用
CAN和工业总线技术-时间敏感系统-电机控制-PLC和工业PC
无线通信技术-WiFi/蓝牙/Mesh/蜂窝/5G网络-物联网安全-低功耗设计
嵌入式系统课程改革-物联网和AI教学研究-职业教育-企业人才培养
嵌入式智能系统应用（智能家居、可穿戴设备、机器人、医疗电子、汽车电子和航空航天等）

征文要求及注意事项：

1. 稿件务求主题新颖、论点明确、论据可靠、数字准确、文字精炼、逻辑严谨、文字通顺，具有科学性、先进性和实用性；
2. 稿件必须为中文，且须加有英文标题、作者信息、摘要、关键词和规范的参考文献列表；
3. 稿件请采用WORD排版，包括所有的文字、表格、图表、附注及参考文献；
4. 从稿件成功投递之日起，在2个月内请勿重复投递至其他刊物。本刊不发表已公开发表过的论文。文章严禁抄袭，否则后果自负；
5. 本刊采用同行评审的方式，审稿周期一般为5~14日。

欲了解更多信息请登录 <https://www.hanspub.org/journal/etis>

联系邮箱：etis@hanspub.org



嵌入式技术与智能系统

主编：何立民 北京航空航天大学教授
主办：汉斯出版社 珠海吴谷电子科技有限公司
编辑：《嵌入式技术与智能系统》编委会

网址：<https://www.hanspub.org/journal/etis>
电子邮箱：etis@hanspub.org