基于商用车中央域控制器的辅助驾驶系统设计

孙 炜, 王 全, 程翔宇, 文健峰, 杨杰君
(中车时代电动汽车股份有限公司, 湖南 株洲 412007)

摘 要:针对商用车分布式辅助驾驶系统存在的算力协同困难与实时控制问题,本文提出一种基于中央域控制器的辅助驾驶系统设计方案。该方案在硬件上采用 SoC 与 MCU 双芯协同架构,在软件上运用分层解耦、服务动态加载与功能安全整合等技术。通过这种软硬件协同设计,实现了系统资源的高效利用与通信瓶颈的突破,为解决分布式系统中的资源浪费与实时性约束问题提供了新的技术路径。

关键词:商用车; 辅助驾驶; 中央域控制器; 功能安全; 中间件
中图分类号:TP273^{+}.5; U469.1 **文献标志码**:A **DOI**:10.15917/j.cnki.1006-3331.2026.01.005

Commercial Vehicle Assisted Driving System Design Based on a Central Domain Controller

SUN Wei, WANG Quan, CHENG Xiangyu, WEN Jianfeng, YANG Jiejun
(CRRC Electric Vehicle Co., Ltd., Zhuzhou 412007, China)

Abstract: To address the challenges of coordinating computing power and implementing real-time control in distributed driver assistance systems for commercial vehicles, this paper proposes a design solution based on a central domain controller. At the hardware level, the solution adopts a dual-chip collaborative architecture consisting of SoC and MCU. At the software level, it employs technologies such as hierarchical decoupling, dynamic service loading, and functional safety integration. Through this hardware-software co-design approach, the system achieves efficient resource utilization and overcomes communication bottlenecks, thereby providing a new technical pathway to address resource wastage and real-time constraints in distributed systems.

Key words: commercial vehicle; assisted driving; central domain controller; functional safety; middleware

随着新型电子电气架构、人工智能和集中域化技术的发展与电动汽车新四化(电动化、网联化、智能化、共享化)[1]需求的普及,整车中央域控技术与辅助驾驶技术已成为国内外商用车行业的两大研究热点。

国内研究主要聚焦于利用分布式控制与 CAN 总线技术以提升辅助驾驶的稳定性与实时性,并针对商用车特殊工况探索了多传感器融合与功能安全方案[2]。但这些研究仍基于传统的分布式架构,其固有的算力分散、通信延迟高与扩展性不足等问题,限制了系统的整体效能。国外研究则更侧重于域集中化、深度学习及硬件协同调度等前沿方向。其主流方案基于 AP AUTOSAR 平台,结合高算力的片上系统(System on Chip,SoC)与高性能实时控制策略,虽能有效处理复杂场景,却因系统复杂性与成本过高而难以适用于商用车领域。为在性能与成本、复杂度之间寻求更优平衡,本文提出了基于中央域控制器的辅助驾驶系统设计方案。

1 系统设计方案概述

本文方案的核心在于通过以下三方面的协同创新,实现成本、性能与安全的平衡。

1)硬件平台创新。硬件平台采用国产 SoC 与微控制器(Micro Control Unit, MCU)协同的架构设计,在严格控制成本的前提下,同时满足高性能计算与高实时控制的双重需求。测试表明,在系统总成本相近的前提下,相比同类型的传统单芯分布式方案,本方案在系统算力、感知性能与通信效率等核心性能上的提升幅度均超过 5%。

2) 软件架构创新。软件架构采用分层解耦与服

收稿日期:2025-03-18。

第一作者:孙 炜(1995—),男,工程师,主要从事整车智能控制相关工作。E-mail:750183947@qq.com。

务动态加载的设计范式，并融合面向服务+中间件管理的核心风格，同时将功能安全与信息安全设计贯穿全局，从而显著降低了系统耦合度与通信负载。

3）技术路径创新。在系统架构层面，创新性地融合了基于规则的算法与数据驱动的算法两种技术路线。该融合策略在确保系统基础安全的前提下，有效提升了其对复杂场景的适应能力，为商用车高阶辅助驾驶提供了新的技术思路。

2　系统平台与架构设计

2.1　基于分布式的辅助驾驶系统分析

1）算力无法协同调度。在传统分布式架构中，各功能控制器相互独立，其算力资源固定且无法跨域调配，导致系统算力利用率低下。以商用车为例，车辆通常配备两类独立的控制器：一类专门处理低速场景，如倒车辅助与 360 环视；另一类则负责高速行驶中的关键功能，包括自适应巡航控制（Adaptive Cruise Control，ACC）、自动紧急制动（Automatic Emergency Braking，AEB）、车道保持辅助（Lane Keeping Assist，LKA）。当车辆处于高速行驶状态时，无法调用闲置的低速控制器算力来增强 ACC 或 AEB；与之类似，当车辆低速行驶或泊车时，也无法调用闲置的高速控制器算力来提升环视、融合泊车及远程泊车等功能。这种“算力孤岛”现象造成系统资源的固化与错配，不仅限制了整体性能拓展，还增加了硬件成本。

2）通信延迟与总线负载压力。在传统分布式架构中，各控制器依赖总线进行数据交换，因此通信延迟普遍较高。同时，总线负载压力随着传感器数据的增加而显著上升。以 1 Mbps 的高速 CAN 总线为例，单个毫米波雷达每周期若传输包含 30 个目标的数据帧，其通信负载可占据总线总容量的约 50%。而过高的负载率不仅限制了其他关键数据的实时传输，还会进一步加剧系统整体的通信延迟。

3）架构不一致与功能堆叠困难。在传统分布式架构中，各控制器内部的辅助驾驶功能通常基于不同的软件架构实现，缺乏统一的设计规范，导致系统整体架构松散且异构性强。功能间仅能通过离散的信号进行通信，并高度依赖上层域控制器进行协调与调度。随着辅助驾驶功能不断叠加，这种架构上的不一致性会显著增加系统的复杂性与离散程度，最终导致系统难以有效整合、扩展和维护。

4）基于功能的设计难以适应高阶场景调度。传统分布式架构以独立功能（如 ACC、LKA、AEB 等）为单位进行设计与部署，导致系统缺乏面向整体驾驶场景的调度能力。在 L3 及以上级别自动驾驶中，系统需根据动态场景协调多模块协同工作，而功能孤岛化的设计使得场景化调度难以实现。随着功能数量不断增多，功能间的交互与边界日趋模糊；同时，由于系统缺乏统一架构来管理并行任务，系统整体复杂度急剧上升，最终制约了高阶自动驾驶的实现与演进。

5）功能模块与感知结果无法共享。在传统分布式架构下，各控制器基于独立的软件栈运行，缺乏统一的中间件或平台支持，导致通用功能模块无法跨控制器复用。这不仅造成相同功能的重复开发与验证，浪费大量开发资源；还使得某一控制器已完成的感知结果无法被其他控制器直接调用，导致相同数据在不同控制器中被重复计算，造成算力资源的浪费。这种模块与数据无法共享的局面，最终导致了开发成本与硬件成本的双重上升。

为应对上述分布式架构的局限性，智能驾驶系统正呈现向域集中式架构演进的发展趋势，其核心在于将原本分散的多个控制器逐步集成至高算力的中央域控制器中。在此架构下，即便系统内仍保留部分专用控制器，其功能也趋于简化，通常仅负责执行特定类型的感知任务并上传原始数据，而核心的感知融合、决策规划与控制等功能均由中央域控制器统一处理。这种高度集成化的设计，要求域控制器的软件架构必须具有强大的实时数据处理、高效的任务调度与可靠的系统服务能力。

2.2　中央域控平台与架构设计

由于商用车智能驾驶系统亟需实时处理多源异构数据并部署高算力算法。因此，采用中央域控制架构已成为明确的技术方向。基于此，本系统采用中央域架构进行设计，并遵循“轻量化、高效率、低延时、低耦合、高内聚”的核心设计理念。

图 1 中的中央域控制器硬件平台支持 9～32 V 宽电压电源输入，可通过以太网、CAN 及整车一挡电信号进行唤醒。基于集中式电子电气架构，该平台能够与整车智能座舱域、底盘域及车身域进行实时跨域数据交互。

系统核心芯片模块采用“性能域”AI-SoC 与“实时域”Safety-MCU 的协同架构。理论上，部分应用场景可采用单 SoC 方案，并依赖其内置安全岛与锁步核执行 MCU 相关任务。然而，相较于传统成熟的外挂 MCU 方案，该集成式设计存在显著局限：内置 MCU 内存资源有限，制约了规划控制算法与系统软件的部

署;同时,其在功能安全与信息安全的保障能力目前也难以全面满足车规级要求。因此,短期内通过 SoC 与 MCU 解耦实现功能隔离与资源互补,仍是更为可靠的技术路径。长期来看,随着 SoC 集成度提升、安全机制完善及软件架构演进,单一芯片支撑全栈功能将成为可能。

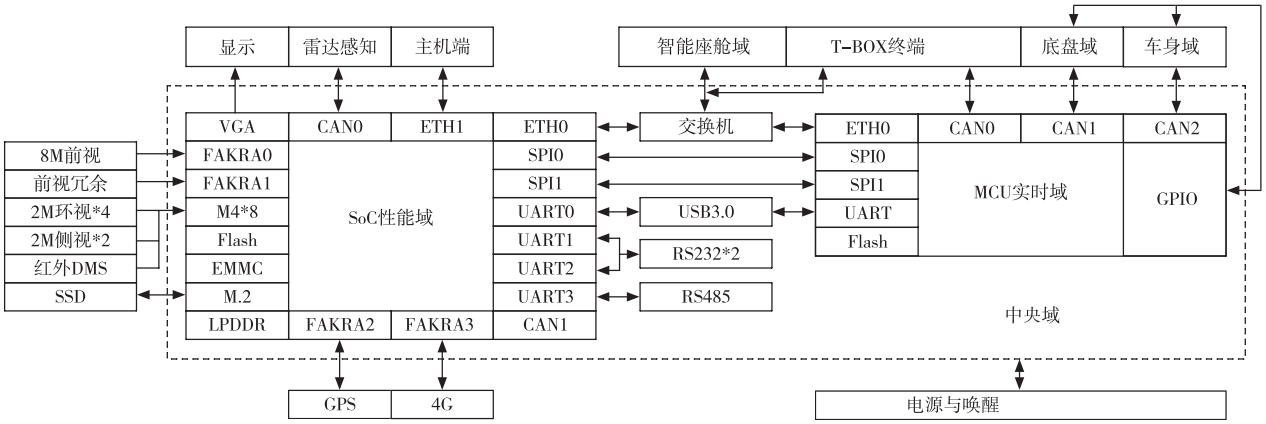

图 1 中央域控平台与整体架构

平台采用“性能域+实时域”双核架构,其核心由 AI-SoC 与 Safety-MCU 共同构成。在部分场景中,单 SoC 方案理论上可通过其内置安全岛执行部分 MCU 功能,但其实现仍存在显著不足:一方面,内置 MCU 内存资源受限,难以支持复杂的规划控制算法及系统软件部署;另一方面,当前单 SoC 方案在功能安全与信息安全方面尚无法完全满足车规级要求。因此,现阶段采用 SoC 与 MCU 功能解耦、协同工作的架构,在资源分配、安全合规与系统扩展性等方面更具可行性与可靠性。

1) SoC 承担高算力任务。SoC 主要负责多传感器感知处理、异构信息融合、轨迹预测、环境建模及高速车载以太网通信等高负载计算。平台选用的黑芝麻华山 A1000 芯片,集成了 8 核 ARM CortexA55 CPU(主频 1.5 GHz,通用算力达 32.4 K DMIPS)与 ARM Mali450 GPU;其专用计算单元可提供高达 58 TOPS(INT8)的 AI 算力与 128 GFLOPs(FP32/FP16)的计算机视觉算力,并满足 ASIL-B 等级(ISO 26262-3:2018[3])的功能安全要求。相比 NVIDIA Orin 平台,该芯片在满足同等系统算力需求的前提下,不仅综合成本和功耗更低,而且供应链更为稳定可靠,因而更契合商用车对高性价比与可持续供应的严格要求。

2) MCU 保障实时控制与功能安全。MCU 主要承担对实时性与可靠性要求极高的控制任务,其核心功能包括底盘域、车身域和能源域等安全关键系统的实时控制与状态监控。平台选用 STM32F407VGT6 芯片,其功能安全等级达到 ASILD(ISO 26262[4]),并基于 Classic AUTOSAR 架构进行开发,能够满足 AEB、LKA 和 CMBS 等辅助驾驶功能中对时序确定性和功能安全的严苛要求。该芯片已通过长期验证,具备成熟的工具链与稳定的生态,足以满足商用车在复杂恶劣环境下长期运行的严苛要求。当前阶段,该方案在满足同等功能安全与实时性要求的前提下,仍在综合成本方面具有一定优势。

平台采用基于(8+1)V-3R 的多模态感知方案,并预留了传感器扩展能力。在视觉感知层面,摄像头严格依据高级驾驶辅助系统的功能需求进行类别划分与布置。各摄像头的原始图像数据,通过串行器-解串器链路集中上传至 SoC 处理。在传输链路设计上,系统综合权衡了信号质量、传输距离、延迟与成本,最终选用吉比特多媒体串行链路技术,以确保数据传输的高速性与稳定性。在雷达感知层面,毫米波雷达当前通过 CAN 总线向 SoC 输出目标级特征数据,满足基础感知需求。未来若需升级至前融合感知架构,可将其输出模式升级为原始点云数据,以支持更深入的跨模态协同处理。

内部通信选用串行外设接口(Serial Peripheral Interface,SPI)实现 SoC 与 MCU 之间的板间通信。这一决策基于对通信对象、数据参数及安全等级的综合评估。在单一主设备、短距离通信的典型板间应用场景下,SPI 接口不仅具备高速、灵活的数据传输特性,其简洁的软硬件架构也有利于实现稳定可靠的互联。同时,SPI 固有的单主模式与短距传输特性,有效避免了多节点干扰与远距离信号衰减问题,从而能够充分满足 SoC 与 MCU 之间对高实时性、高确定性协同通信的设计要求。

外部通信采用基于功能隔离的 CAN 与车载以太网双路架构。车载以太网集中部署于 SoC 侧，凭借其高带宽与低延迟特性[5]，专用于实现与座舱域等内部功能域的高速数据交换；CAN 总线则集中于 MCU 侧，基于其高可靠性与强实时性，负责与底盘、车身等执行域之间的确定性控制信号传输。

此外，SoC 端配置了嵌入式多媒体卡、Nor Flash 及内存等多种存储单元，以支持不同数据格式与内容的存储需求。

3　辅助驾驶系统软件架构设计

3.1　环绕层和核心层的功能与设计

在智能驾驶领域，辅助驾驶系统本质上是一种新型的“人工智能体”。其软件架构主要依托于中央域控制平台进行设计与实现。研究表明，一个真正意义上的智能体需持续维护一个内在的“世界模型”，并通过不断缩小内部模型与外部环境之间的“差异”来实现并维持系统的有序状态（即熵减）。在此过程中，智能体通常包含两大部分：一是功能实现系统，负责执行日常行为决策；二是安全系统，其通过监测系统内部状态与外部环境之间的差异来识别潜在风险，并主动采取规避措施，以避免系统因错误行为而导致自身或环境受损[6]。基于这一理论，本文将辅助驾驶软件架构在业务层面划分为核心层与环绕层（见图 2）。

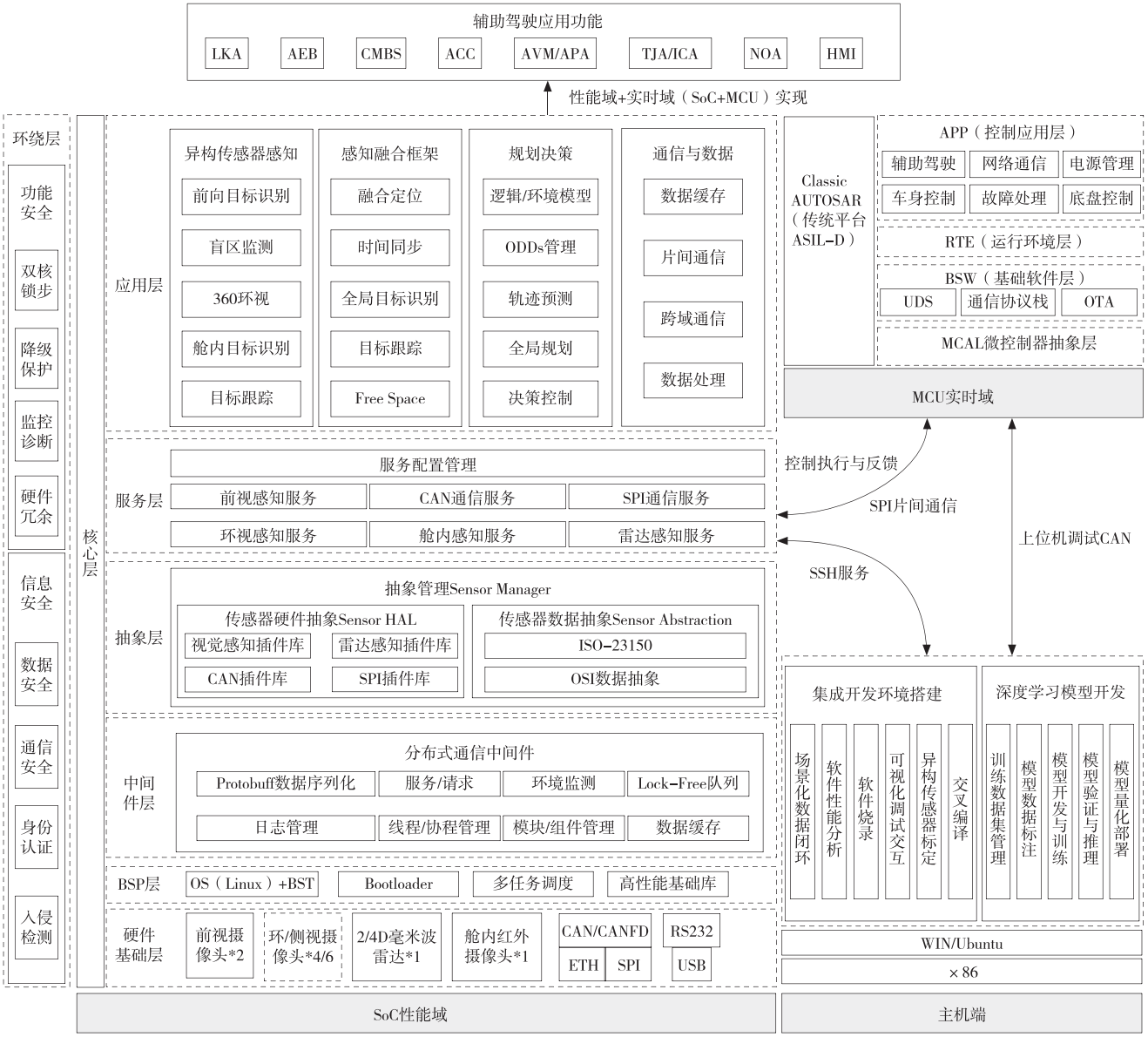

图 2　辅助驾驶系统软件架构

环绕层是在架构设计中将功能安全与信息安全两大关键模块从核心功能逻辑中解耦,并重新整合所形成的独立层次。这种分离并非因其次要,而是为了使其从具体的业务执行中抽离,从而能够在系统层面进行更全局、更统一的框架统筹与实现。功能安全与信息安全模块设计具体说明如下:

1) 功能安全。该模块包含双核锁步、降级保护、硬件冗余与监控诊断。受分布式域控制器系统软件框架的限制,每个独立控制器均需配置冗余硬件(如双 MCU),这导致系统成本与功耗增加。对于 L2 级及以上的高阶主动安全应用而言,其功能安全难以达到 ASIL-D 等级;而基于环绕层的功能安全整合了多个功能模块(如智驾、底盘、车身控制),便于实施统一的安全策略(如 ISO 26262-3:2018[3]中 ASIL-D 等级),且支持全局状态监控与快速故障定位,可及时进行全局安全响应与动态降级保护。

2) 信息安全。该模块包含数据安全、通信安全、身份认证与入侵检测。在分布式架构中,多个独立控制器需通过总线(如 CAN)通信,增加了跨节点安全协同难度,且各节点独立实施安全措施(如不同加密协议),导致整体防护不一致,易受中间人攻击(如伪造 ECU 指令)。因此,保障数据流安全的一致性与可靠性,始终是该架构面临的一项长期挑战。而基于环绕层的设计则能够通过集中式安全加密与统一的访问控制机制,系统化地保障数据流安全并实现高效入侵防护。

核心层负责实现算法执行、感知与规划控制等核心业务功能。其架构特点是越接近控制末端,对实时性的要求越高,尤其在规划与控制环节需满足硬实时约束,即在确定时间内完成响应与执行。这需要底层硬件与操作系统共同提供可靠的实时保障,采用微控制器搭载实时操作系统是一种典型的实现方式。然而,单 MCU 的计算能力有限,难以支撑基于深度学习的大规模感知算法,因此需引入集成高性能中央处理器与专用人工智能算力的片上系统来满足高性能计算需求。基于实时性、系统复杂度及算力需求的不同,核心层可分为两大功能域:①实时域:基于 MCU 及其软硬件系统,确保高实时、高可靠的控制执行;②性能域:其计算平台由 SoC 及黑芝麻智能操作系统(Black Sesame Technologies-Operating System, BST-OS)构成,负责高性能感知、融合与预测等计算密集型任务。

3.2 SoC 性能域架构设计

SoC 性能域的设计核心在于满足高算力、高性能与多进程并发智能应用需求的同时,尽可能达成实时性目标。尽管面向高性能计算的 AP AUTOSAR 架构能够有效实现软硬件解耦,但其在实际应用中仍面临三大技术局限:①开发体系复杂,学习与实施门槛高;②对特定工具链依赖性较强,导致技术选型范围受限;③在支持高实时性任务时,其调度与通信机制难以完全满足确定性响应需求。这些因素使得该架构的研发与集成成本大幅增加。

3.2.1 自主软件架构设计理念与核心机制

基于对 AP AUTOSAR 架构局限性的考量,本系统在其服务发现机制的基础上,构建了一套自主演进的软件架构。该架构运用分层设计,搭载兼容可移植操作系统接口的操作系统,并采用“面向服务+中间件管理”的核心设计范式。面向服务的架构继承了面向对象与构件化设计的核心原则,如封装、复用、松耦合与自包含等,其关键特征可归纳如下:

1) 服务与抽象接口。本架构包含服务的具体实现与其抽象接口两个关键部分,二者在设计上相互分离。服务实现专注于内部数据的处理、计算与存储等核心逻辑;而抽象接口则明确定义了服务提供者与服务消费者之间的交互契约。这种分离使服务实现的内部迭代独立于接口的稳定调用,从而在服务与接口之间实现了真正的解耦。

2) 服务动态加载。服务以动态库的形式提供,其路径集成于环境变量配置中,以此支撑软件架构的分布式更新迭代;应用层功能通过动态配置与加载,将不同服务载入内存随机存取存储器中运行。

3) 基于中间件的消息发布与订阅。各组件间的通信采用基于消息的发布-订阅机制。在该机制中,消息发布者在发出数据后不再保留其控制权,而感兴趣的订阅方则以异步方式接收并处理这些消息。“发布/订阅”模式显著降低了系统各组件间的耦合度,并为组件的可扩展性与可重用性提供了良好支持。

3.2.2 SoC 软件架构实现方案

辅助驾驶系统涉及异构传感器融合、复杂场景规划决策与高实时控制等多项复杂任务。为提升系统的可维护性与可迭代性,需将这些任务拆解为多个独立的服务子系统。每个子系统应支持独立开发、部署与更新,且通过架构设计确保服务间数据流隔离,避免相互干扰。上述需求与架构核心设计风格高度适配。在此风格基础上,结合环绕层中的功能安全与信

息安全机制，最终形成完整的 SoC 软件架构。该架构自下而上可分为：硬件基础层、板级支持包层、中间件层、抽象层、服务层和应用层。

1）硬件基础层。该层作为系统软件架构的数据流基础，其硬件选型遵循整车辅助驾驶功能需求、性能冗余与可拓展性三大设计原则。基于上述原则，系统最终选用黑芝麻 A1000 高算力 SoC 芯片、视觉与毫米波雷达相结合的感知融合方案，以及支持多模态交互的内外通信接口。

2）板级支持包层。该层位于硬件基础层与中间件层之间，主要功能是实现操作系统内核（如 Linux/QNX）与底层硬件的适配与优化。其核心工作是基于 BSP 板级支持包，进行面向特定硬件平台的专属系统定制（即 BST-OS），以提升内核与硬件的匹配度与运行效率。BST-OS 与 BSP 均支持持续的二次开发与迭代优化。此外，该层还集成了 Bootloader、多任务调度器及关键的高性能基础库，用以支撑上层应用的稳定、高效运行。

3）中间件层。该层作为实现 SoC 内部分布式计算的核心架构，通过构建统一的通信框架，对软件服务与组件间的交互方式进行抽象与标准化。其能高效管理包括底层硬件数据访问、上层应用服务调用、任务节点协同，以及进程内、进程间乃至跨物理节点的通信与数据流转，从而为复杂功能提供可靠、解耦且可扩展的集成基础。

通信框架需面向服务与高阶辅助驾驶场景进行设计，并且要针对轻量化、高并发、低延迟以及高吞吐量通信需求进行特定优化。

在上述需求驱动下，系统通信框架以分布式实时消息中间件 CyberRT 与 Protobuf 通信序列化为基础，并针对车载场景进行了系列优化与扩展，其核心逻辑与顺序如下：①通过移除非必要依赖优化系统存储空间；②针对图像等大数据通信场景，采用基于 TZC 的零拷贝传输方案以提升传输效率；③面向高并发数据流，通过线程/协程管理与模块化设计实现通信优先级灵活配置；④为确保通信的可靠性与安全性，引入无锁对象池、端到端环境监测及数据缓存等多重保障机制。

4）抽象层。该层主要面向不同整车项目中因辅助驾驶功能差异导致的传感器、执行器与驱动接口多样化问题。若直接将多样化的硬件接口暴露给上层应用，将导致应用软件与底层硬件深度耦合，增加开发复杂度，并违背分布式架构的解耦原则。为此，本层通过将各类底层硬件功能进行统一抽象与标准化封装，为上层应用软件提供稳定的服务接口。其核心目的在于解耦上层应用软件与底层硬件的具体实现细节。通过引入此抽象接口层，硬件之间的差异被封装于本层之内，使得上层应用不再依赖特定硬件的型号、接口或驱动。基于上述设计，系统最终实现了跨硬件平台的应用软件可移植性、可维护性与功能模块的高效复用。

抽象层包含三个核心部分：①传感器驱动抽象层。该层通过为各类传感器开发遵循统一接口标准的动态插件，封装不同厂家、接口与通信协议的硬件差异。②数据模型抽象层。该层通过定义遵循 ASAM OSI 及 ISO 23150：2023 标准[7]的统一数据结构，实现上层应用与具体硬件数据格式的解耦。③抽象资源管理层。该层负责驱动插件的动态加载与卸载、传感器的注册与生命周期管理、错误状态监控及数据的统一分发。

5）服务层。该层是面向服务架构理念的核心体现，集成了各类关键功能服务，包括前视、环视、舱内及雷达等感知服务，以及 CAN、SPI 等域内外通信服务。这些服务基于高阶辅助驾驶的共性需求进行通用化设计。服务层在运行时，其内部各功能服务通过中间件从底层抽象层按需获取数据流，并完成数据筛选、融合与格式转换等预处理。最终通过统一的接口为上层应用提供规整、可靠的数据与服务支持。

6）应用层。该层作为 SoC 性能域软件架构的顶层，直接承载并实现了辅助驾驶的各项核心智能化功能。依据功能逻辑，该层横向划分为四大解耦的应用模块：异构传感器感知、感知融合[8]、规划决策及域内外数据通信。各模块间通过定义良好的接口进行协作，其运行所需的数据流统一由底层的服务层提供。

本文将辅助驾驶系统视为新兴智能体模型与传统整车功能模型的有机结合。该结合源于二者在技术特性上的互补关系：传统模型基于规则算法，以工程师为主导进行开发，具有结构清晰、可解释性强与功能安全严谨的优势，但其算法固化，面对复杂长尾场景时，迭代潜力与适应能力均显不足；而智能体模型采用数据驱动模式，以训练数据为主导，在场景适应性、算法潜力与迭代能力上优势显著，但其决策过程可解释性弱，且系统的鲁棒性与稳定性高度依赖训练数据的规模与质量。因此，二者融合的目的是兼顾系统的可解释性、安全性与场景适应能力。

综合考虑当前商用车的技术能力、总体成本、训

练数据质量、辅助驾驶应用场景及法规要求等多重因素,最终选定“基于规则的整体软件架构+基于数据驱动的应用层深度学习框架”这一复合式软件架构。该架构能够在确保系统整体安全可靠的基础上,有效支持应用层中感知、规划、决策等多模态、多场景、高算力、高性能核心算法的软件实现需求。

3.3 MCU 实时域软件结构设计

MCU 的核心功能是实现高实时性、高可靠性的整车控制与辅助驾驶执行端。因此,其软件架构严格依据 Classic AUTOSAR 进行设计,以满足 ISO 26262-3:2018[3]标准中最高等级 ASIL-D 的功能安全要求;同时,其设计采用分层式模块化理念(见图 2),构建了包含微控制器抽象层、基础软件层、运行环境层及控制应用层的四层架构体系。该体系的具体设计如下:

1)微控制器抽象层。该层作为 MCU 软件架构的基础,深度集成了 STM32 标准外设库,提供了包括 SPI/CAN 总线驱动、模数转换、脉冲宽度调制等核心硬件接口的统一封装。为确保信号可靠性,该层对关键传感器信号采用双冗余采集机制,并基于循环冗余校验与信号合理性判断进行数据完整性验证。同时,通过配置直接内存访问通道,支持 MCU 在 SPI 从机模式下与 SoC 之间实现高带宽、低延迟的数据交换。

2)基础软件层。该层作为 MCU 功能实现的软件核心,主要提供两大基础服务:一是构建了标准化的 CAN/SPI 通信协议栈,实现与性能域、底盘域及车身域等其他功能域之间的跨域数据交互;二是集成了符合 ISO 14229-1:2020[9]规范的统一诊断服务协议栈,可基于统一诊断服务,实现车端诊断与远程升级,从而增强了系统的可维护性与诊断通信可靠性。

3)运行环境层。该层以实现高实时性控制为目标,采用基于高精度定时器的时间触发架构。其设计通过定时中断机制保证响应的确定性,并借由抢占式调度策略确保调度的可靠性,从而全面保障通信处理、主控制逻辑、传感器数据同步等关键任务的实时性能。

4)控制应用层。该层作为 MCU 软件功能的最终载体,采用“全模型分模块”的自动化代码生成方式进行开发。其核心内容包括:车道保持控制、基于决策树模型的主动制动[10]、网络通信、电源管理、车身与底盘控制及故障处理等关键功能模块;同时,设计了多层次故障安全策略以保障系统降级与安全状态管理。此外,该层支持基于 CAN 标定协议或通用测量与标定协议的在线程序烧录与多变量同步标定功能,为系统调试与参数优化提供标准化接口。

3.4 主机端结构设计

SoC 与 MCU 作为嵌入式软件部署终端,两者的资源、算力与加速优化重心均放在各自核心功能的实现[11-12]上。因此需要主机端作为开发者与嵌入式平台对外交互窗口,实现集成开发环境搭建与深度学习模型开发两大核心功能。

集成开发环境搭建包括以下 4 个子功能:

1)交叉编译环境搭建。交叉编译环境基于“SoC-主机端”架构进行搭建。首先,将 SoC 性能域所需的运行时环境、系统依赖及基础库打包为可移植的软件开发工具包镜像;随后,在主机端使用该镜像创建 Docker 容器,并将此容器作为统一的编译与调试环境。该容器内已集成 GNU 编译工具链、CMake 构建系统、Python、Protobuf 等必要开发工具,并配置安全外壳协议服务,从而实现从主机端到 SoC 目标平台的高效、一致的交叉编译流程。

2)多模态传感器标定。考虑到标定任务本身具有非实时性与计算密集的特点,为降低对嵌入式平台资源的占用并实现标定过程与结果的可视化,系统将标定功能部署于主机端。首先,主机端的可视化上位机软件对视觉、雷达等传感器的原始数据进行处理与参数计算,并生成标定结果及结构化的 XML 配置文件。随后,该配置文件通过安全外壳协议安全地传输至车端的 SoC/MCU。最终,车端依据该文件完成参数的在线写入与同步。

3)可视化调试交互。为支持高效的调试与分析,系统在主机端基于 Qt 框架开发了专用的人机交互界面。该界面能够同步接收来自高级驾驶辅助系统的多源数据,包括激光雷达点云、毫米波雷达点云和摄像头视频流等传感器原始数据,以及横摆角速度、方向盘转角、整备质量和轮速脉冲等车辆动力学数据,并实现数据的实时可视化显示与对比分析功能。

4)性能分析。为对 SoC 平台进行全面的性能剖析,主机端采用 Linux Perf 工具链对 CPU 占用率、DDR 内存带宽利用率及 GPU/NPU 负载等关键硬件指标进行周期性采样与定量分析;同时集成 Valgrind 动态分析工具,对软件运行时的内存使用、线程同步及资源泄漏等问题进行深度诊断,为系统优化提供数据支撑。

主机端深度学习模型开发步骤如下:

步骤 1：训练数据集管理。主机端对采集的调试数据及训练数据集进行同步复用，设置基于边缘场景的数据采集触发条件，在确保数据覆盖典型辅助驾驶场景的同时通过自动化脚本过滤无效数据。

步骤 2：模型数据标注。主机端针对商用车辅助驾驶功能需求构建基础标注层、场景语义层和决层行为层等多层级标注体系，并采用半自动标注工具提高标注效率。

步骤 3：模型开发与训练。为平衡模型精度、资源算力及嵌入式部署的实时性要求，主机端采用通道剪枝、压缩模型参数量、多任务联合训练等方式实现“轻量化网络+全覆盖数据集”的模型开发方案。

步骤 4：模型验证与推理。针对深度学习模型可解释性不足的问题，主机端基于场景数据集搭建了模块化测试框架，对 AEB、LKA、ACC、BSM 等应用功能进行实车场景推理测试并输出推理性能报告。

步骤 5：模型量化与部署。主机端以降低量化精度损失为目标，针对 SoC 中 NPU 所支持的模型输出维度、激活函数、量化参数及定制化算子进行硬件适配，最终通过黑芝麻模型量化部署工具链完成量化优化与嵌入式部署。

4　测试与应用

本文测试基于 10 m 纯电动客车平台，同步部署了中央域式辅助驾驶系统（简称中央式系统）与分布式辅助驾驶系统（简称分布式系统）。测试在天气晴朗、能见度高、道路干燥平整的典型城市场景下进行，以对比评估两套系统的性能。其中，中央式系统的前向感知效果如图 3 所示；分布式系统则采用摩视 MagicDrive 1.0 一体机方案，其 MCU 搭载英飞凌 TC234 芯片。

图 3　中央域控平台辅助驾驶系统前向感知效果

在感知性能方面，中央式系统在典型场景下的目标检测准确率约为 96.6%，相比分布式系统（约 91%）高出约 6.2%；同时，中央式系统平均感知帧率约为 29.8 FPS，相比分布式系统（约 21.4 FPS）高出约 39.2%。

在通信性能方面，中央式系统采用 SPI 总线，其速率范围为 2～4 Mbps；而分布式系统基于 CAN 总线，速率范围为 250～500 Kbps。相比之下，中央式系统的总线速率至少是分布式系统的 4 倍。此外，在同等功能配置下，以 8 字节/帧进行统计，中央式系统完成交互仅需 58 帧，而分布式系统则需 73 帧以上；经计算，前者的通信负载率较后者降低约 20.5%。

中央域辅助驾驶系统在上述各项核心技术指标上均表现出显著优势，目前，该中央域辅助驾驶系统已覆盖多款商用车型，并历经多次量产批次验证，累计装车量超过 150 套。

5　结束语

本文设计了一种基于中央域控平台的辅助驾驶系统，其核心创新在于两方面：一是通过“国产 SoC+MCU”双芯协同的软硬件融合架构，实现了 ASIL-D 级功能安全与高算力的统一，解决了分布式系统的固有瓶颈；二是通过“面向服务+中间件管理”的软件范式与“规则与数据驱动”融合的算法路线，在提升系统灵活性、可扩展性与场景适应性的同时，降低了通信负载。

实车验证表明，本系统在算力利用率、响应效率及安全冗余方面均显著优于传统分布式方案。未来工作将聚焦于多模态感知融合优化与车路协同扩展，以推动商用车向高阶智能化持续演进。

参考文献：

[1] 林澍，陈宇锋，黄丽军. 电动汽车域控制器发展分析与未来展望[J]. 汽车实用技术，2023，48(11)：207-212.

[2] 郑燚，张世民，陆云龙. 基于 AUTOSAR 的域控制器以太网通信系统设计实现[J]. 汽车工程，2023，45(6)：965-973.

[3] International Organization for Standardization. Road vehicles—functional safety Part 3：concept phase：ISO 26262-3：2018[S]. 2nd ed. Geneva：ISO，2018：14-20.

[4] 张婷，潘定海，曹明星. 一种纯电动汽车的自动驾驶系统设计[J]. 汽车文摘，2022(6)：41-48.

（下转第 53 页）

璃左右圆弧处的近壁风速。因此，本次改进预计能达到类似效果，从而改善目标车型该区域的除霜性能。

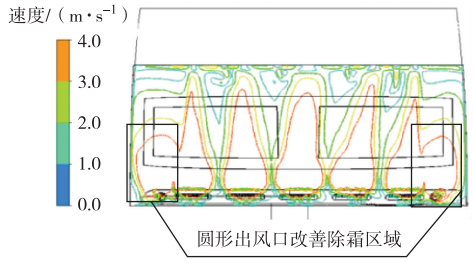

图 8　某同类车型近壁风速分布

4　结束语

城市客车除霜系统的设计是一项综合性工程，涵盖除霜器选型与安装、连接管路与仪表台风腔设计，以及前风挡玻璃外形结构等多个相互关联的方面。在新车型设计初期，面对复杂的系统耦合关系，仅凭经验往往难以保证设计方案的合理性。因此，在开发早期引入 CFD 仿真分析与理论验算进行正向设计至关重要。本文正是通过系统性地应用这一方法，对除霜系统进行了设计与优化。实践表明，应用这一正向设计流程，可在提升设计质量与可靠性的同时，有效控制成本并缩短开发周期，进而为提升产品市场竞争力提供关键技术保障。

参考文献：

[1] 莫荣博. 基于 CFD 的某商用车 HVAC 除雾性能分析及优化[J]. 企业科技与发展，2019(5)：56-58.

[2] 中华人民共和国交通运输部. 客车空调系统技术条件：JT/T 216—2020[S]. 北京：人民交通出版社，2020：15-17.

[3] 郭旭光，汤波，罗杨，等. 汽车风窗玻璃除霜影响因素分析[J]. 轻型汽车技术，2016(11)：22-24.

[4] 刘宁，李龙勇，闫鑫，等. 基于 CFD 的某商用车除霜系统仿真优化[C]//中国汽车工程学会. 2023 中国汽车工程学会年会论文集(4)，2023：552-560.

[5] 王东. 汽车风挡玻璃除霜效果 CFD 分析[J]. 汽车实用技术，2021，46(10)：71-72.

[6] 于翰林. 基于 Star-CCM+的轻型卡车空调除霜性能数值模拟及试验探究[J]. 机械设计，2021，38(S1)：115-119.

[7] 孙城，李文超. 出风道性能优化对挡风玻璃除霜性能影响分析[J]. 机械设计与制造，2026(1)：77-83.

[8] 吴宾，郭鹏，辛俐，等. 汽车除霜格栅角度对前风窗玻璃的除霜性能影响[C]//第十四届全国水动力学学术会议暨第二十八届全国水动力学研讨会文集(上册)，北京：海洋出版社，2017：824.

[9] 崔行振. 电动汽车空调除霜风道系统流场分析及优化研究[D]. 青岛：山东科技大学，2019.

(上接第 33 页)

[5] 杨东. 智能车辆自动驾驶域控制器设计与实现[D]. 重庆：重庆邮电大学，2020.

[6] 殷玮. 智能驾驶系统的混合架构设计[J]. 机电产品开发与创新，2023，36(1)：37-40.

[7] International Organization for Standardization. Road vehicles—data communication between sensors and data fusion unit for automated driving functions—logical interface：ISO 23150：2023[S]. 2nd ed. Geneva：ISO，2023：19-25.

[8] 周文鹏，路林，王建明. 多传感器信息融合在无人驾驶中的研究综述[J]. 汽车文摘，2022(1)：45-51.

[9] International Organization for Standardization. Road vehicles—unified diagnostic services(UDS)—Part 1：application layer：ISO 14229-1：2020[S]. 3th ed. Geneva：ISO，2020：3-25.

[10] 韩玉，马晓磊，陶砚盟，等. 面向驾驶需求的多任务集成式决策控制方法[J]. 中国公路学报，2024，37(10)：249-266.

[11] 徐洋. 先进驾驶员辅助驾驶系统关键技术研究[D]. 重庆：重庆大学，2017.

[12] PALLIERER R，SCHMELZ B. 适用于高性能车载计算平台的自适应 AUTOSAR[J]. 汽车与配件，2019(3)：43-45.