DOI:10.19651/j.cnki.emt.2107859

# 基于 AUTOSAR 的 CAN 通信栈设计*

李超超[1,2] 武 恪[1,2] 方 菱[1]
(1.中国科学院 合肥物质科学研究院 合肥 230031;2.中国科学技术大学 合肥 230026)

**摘 要**:汽车开放系统架构(AUTOSAR)是作为包括汽车制造商、供应商和工具开发商在内的开发合作伙伴成立的,旨在创建一个开放和标准化的汽车架构,解决汽车控制器软件的诸如开发周期长、可重用性差等问题。AUTOSAR CAN 通信栈是 AUTOSAR 的重要协议栈,其是为了解决汽车 CAN 网络通信层软件质量良莠不齐、复用性差等问题。本文旨在依据 AUTOSAR4.0.3 标准在 NXP MPC5748G 平台设计实现 AUTOSAR CAN 通信协议栈底层模块,包括 CAN 控制器状态控制、通道通信控制、发送缓冲机制、发送取消机制、软件接收滤波机制。在实验中对比传统 CAN 通信软件和本文实现的 CAN 通信软件的周期发送延迟,结果显示 AUTOSAR CAN 通信软件能够降低高优先级报文周期发送的 90%左右的平均延时,证明本文实现的 AUTOSAR CAN 通信栈有效地提升了 CAN 通信软件性能。

**关键词**:AUTOSAR;CAN 协议栈;汽车电子
**中图分类号**:TP311.52 **文献标识码**:A **国家标准学科分类代码**:520.4070

# Design of CAN communication stack based on AUTOSAR

Li Chaochao[1,2] Wu Ke[1,2] Fang Ling[1]
(1. Hefei Institutes of Physical Science, Chinese Academy of Sciences, Hefei 230031, China;
2. University of Science and Technology of China, Hefei 230026, China)

**Abstract**: The automotive open system architecture (AUTOSAR) is an open and standardized automotive architecture that enables development partners to integrate, exchange, re-use and transfer functions within a vehicle network and improves their efficiency of development. AUTOSAR CAN communication stack is an important protocol stack of AUTOSAR, which is designed to pave the problems of uneven software quality and poor reusability in the communication layer of the automobile CAN network. The purpose of this paper is to design and implement the AUTOSAR CAN communication stack base module based on the AUTOSAR 4.0.3 standard in the NXP MPC5748G platform, including CAN controller state control, channel communication control, transmit buffer mechanism, transmit cancel mechanism, and software receive filter mechanism. In the experiments comparing the cycle sending delay of traditional CAN communication software and the CAN communication software implemented in this paper, the results show that AUTOSAR CAN communication software can reduce the average delay of high priority message cycle sending by around 90%, proving that the AUTOSAR CAN communication stack implemented in this paper effectively improves the CAN communication software performance.

**Keywords**: AUTOSAR;CAN communication stack;automotive electronics

## 0 引 言

随着汽车技术的发展,汽车电子控制器(electronic control unit,ECU)的数量不断增加,导致开发周期长、开发成本高、软件可复用性差等问题在软件开发中频繁出现[1],因此众多知名汽车企业建立汽车开放系统架构联盟,并合作研发汽车开放软件架构规范(automotive open system architecture,AUTOSAR)。AUTOSAR 架构为实现汽车 ECU 的软硬件解耦和高效集成,采用标准化的层次结构,AUTOSAR 中的每个模块都有标准的数据结构定义、服务接口定义和功能描述[2-3]。AUTOSAR 架构以其规范化的服务接口以及模型定义提高了系统应用软件的复用能力,提高了 ECU 软件系统的集成和开发能力,降低了软件开发的成本[4]。

收稿日期:2021-09-13
* 基金项目:安徽省重点研究与开发计划项目(202004a05020041)资助

车内电子控制器之间的通信普遍采用控制器局域网络(controller area network,CAN),但在 ECU 开发中经常出现由于需求更改、性能不足、硬件修改等问题,需要在开发过程中不断修改和维护通信代码,导致开发周期的延长和开发成本的增加,因此通信软件的复用性十分重要。随着汽车智能化程度的不断加深,车内 ECU 数量不断增加,需要在 CAN 总线上传输的流量增加,但 CAN 总线的通信质量会随着负载率的增加而下降,一般要求总线负载率在 30%以下且最佳总线负载率不超过 25%[5],当超过 30%时总线上的冲突和延迟较为明显。文献[6]中提出一种基于最先截止时间优先(earliest deadline first,EDF)调度算法的分布式 CAN 网络动态调度方法,并通过实验证明该方法可以有效降低最坏发送响应时间,提高了 CAN 网络的带宽利用率。

AUTOSAR CAN 协议栈从属于 AUTOSAR CAN 通信服务栈,只要实现符合 AUTOSAR 标准的 CAN 协议栈接口就能和其他模块以及应用软件集成,提高了 CAN 通信软件的复用性。文献[7]中在 MPSoC 平台实现虚拟 CAN 控制器以及符合 AUTOSAR 标准的接口,实验验证其能够在真实的 CAN 总线中使用并能与符合 AUTOSAR 标准的其他模块集成。文献[8]中在 MC9S12DG128 平台上实现 CAN 底层通信服务,并着重描述实现的 AUTOSAR CAN 通信栈的发送和接收流程。文献[9]中基于提出共享和独享两种发送缓冲机制,在不同的应用场景下选择适合的发送缓冲机制可以有效地减少协议栈的内存占用。文献[10]中实现基于 AUTOSAR 通信栈的网络传输模块,能够将应用层报文在 CAN 总线上分包传输,并使用 TTworkBench 进行功能的一致性测试证明符合 AUTOSAR 标准。文献[11]中为解决总线负载率过高导致的发送延迟,通过配置 CAN 通信栈确保报文的错峰发送,并通过计算预测可以将最大发送延时下降 45%。文献[12]中基于 AUTOSAR CAN 通信协议栈实现统一诊断服务(unified diagnostic services,UDS),并通过实验验证 UDS 的常用诊断功能。

上述文献均将重点放在 AUTOSAR CAN 通信栈规范实现和功能验证上,并未将相应文献中实现的 AUTOSAR 软件与现有 CAN 通信软件进行对比。本文根据 AUTOSAR 4.0.3 标准在 MPC5748G 平台上设计实现 CAN 通信协议栈底层模块,并使用 CANoe 设备在不同负载率下比较直接控制底层周期发送和 AUTOSAR CAN 通信栈的轮询周期发送之间的周期延时差异,证明本文实现的 AUTOSAR CAN 通信栈的性能优于直接控制底层模块。

## 1　AUTOSAR CAN 通信服务栈

AUTOSAR 为提升汽车 ECU 应用软件的复用性,在架构中定义一个运行时环境将应用软件和底层硬件平台隔离,架构自下而上包括微控制器层、基础软件层、运行时环境和应用层。应用层由开发人员实现的应用软件组件构成,应用软件组件之间通过运行时环境相互通信,协同实现 ECU 的应用功能。运行时环境负责应用软件组件之间、应用软件组件和基础软件组件之间的通信。微控制器层指 ECU 实际工作的硬件平台,包括微处理器及其外围电路。基础软件层是 AUTOSAR 架构的基石,其中包含微控制器抽象层、ECU 抽象层、服务层和复杂驱动 4 个软件部分[2]。

如图 1 所示,AUTOSAR CAN 通信服务栈的底层硬件对象包括 CAN 控制器和 CAN 收发器,CAN 控制器内置基础 CAN 通信协议,CAN 收发器负责逻辑电平和差分电信号之间的转换。以 MPC5748G 为例,其有 8 个内置 CAN 控制器,每个 CAN 控制器内部有 96 个邮箱(mail box,MB),每个 MB 都可以存储 CAN 数据报文并配置为发送或者接收。当发送 CAN 报文时,CAN 驱动会将报文存储到指定的控制器 MB,之后控制器会将报文通过 CAN 收发器发送到 CAN 总线,并通知 ECU 发送成功或失败,同理接收 CAN 报文时 CAN 控制器将报文存储在 MB 并通知 ECU 读取。

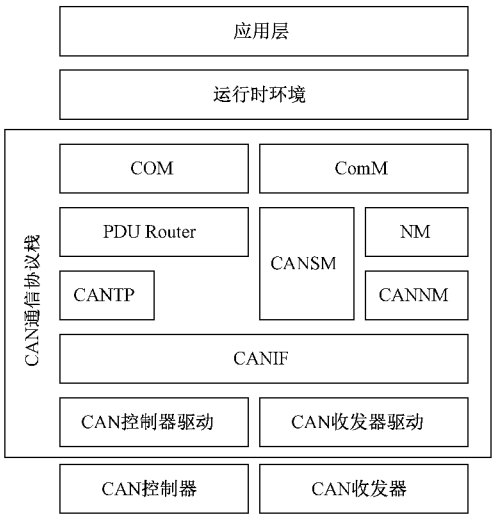


图 1　AUTOSAR CAN 通信服务栈

AUTOSAR CAN 控制器驱动[13]和收发器驱动负责管理和访问 CAN 控制器和收发器。CAN 接口[14](CAN interface,CANIF)一方面负责底层 CAN 驱动的报文和上层模块的协议数据单元(protocol data unit,PDU)之间的路由,另一方面,CANIF 还为上层提供统一的底层驱动访问和控制的服务接口,为底层提供统一事件通知的回调接口。本文主要是讨论 CAN 通信栈的底层服务,CANIF 的上层模块不在本文的讨论范围内,因此不再赘述。

## 2　CAN 通信栈底层设计

### 2.1　控制器模式切换设计

为了通过统一接口管理所有的底层 CAN 控制器,

CANIF 模块要求实现 CanIf_SetControllerMode()接口,该接口可以切换指定控制器的工作状态,本文设计该接口的工作流程如图 2 所示。根据 CAN 控制器规范[13]要求支持 4 种工作模式:1)UNINIT,此时控制器尚未初始化,控制器的所有寄存器都处于复位状态,禁用控制器中断;2)STOPPED,此时控制器不参与 CAN 总线通信,不会应答 CAN 报文和发送错误帧;3)STARTED,控制器参与网络且允许发送和接收 CAN 报文,控制器在该状态下正常工作;4)SLEEP,此时控制器处于休眠模式,不能发送和接收 CAN 报文但可以被唤醒,状态切换如表 1 所示。

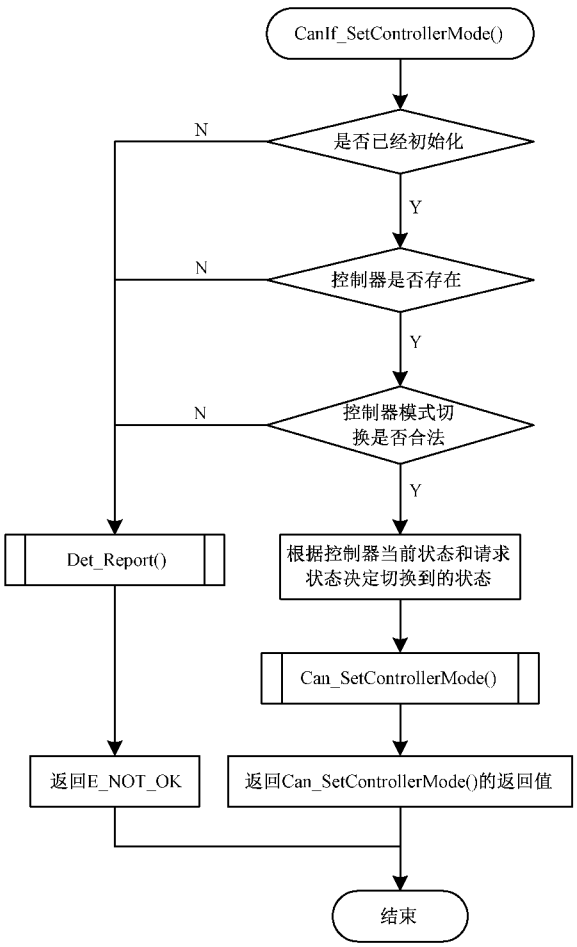


图 2 CAN 控制器状态切换逻辑

**表 1 CAN 控制器状态转移表**

| 当前/请求 | SLEEP | STOPPED | STARTED |
|---|---|---|---|
| SLEEP | SLEEP | STOPPED | — |
| STARTED | — | STOPPED | STARTED |
| STOPPED | SLEEP | STOPPED | STARTED |

### 2.2 通信约束设计

根据 2.1 节的设计 CAN 控制器在 STARTED 模式下可以正常参与网络,但是缺少对通信链路的约束的话会增加上层应用软件的复杂程度,因此根据 CANIF 规范[14]要求设计 6 种通信状态:1)OFFLINE 不允许通道发送和接收;2)OFFLINE_ACTIVE 不允许接收且阻塞发送,但允许发送完成通知;3)OFFLINE_ACTIVE_RX_ONLINE 允许发送通知和接收但阻塞发送;4)ONLINE 允许发送和接收;5)RX_ONLINE 允许接收但不允许发送;6)TX_ONLINE 允许发送但不允许接收。上层应用软件可以调用 CANIF 的 CanIf_SetPduMode()接口切换指定控制器的通信状态,从而约束通道的收发功能,简化上层应用的开发。本文设计的该接口的工作流程和 2.1 节相似,仅须将控制器工作状态切换替换为控制器通信状态的切换。CANIF 在各个服务接口和回调接口中根据目标控制器的当前工作状态和通信状态判断是否接受请求和回调,从而实现对控制器底层和通信链路的控制。

### 2.3 硬件抽象映射设计

在 CAN 驱动模块中,CAN 控制器中配置为发送模式的 MB 用硬件发送标识(hardware transmit handle,HTH)标识,同理 CAN 控制器中配置为接收模式的 MB 用硬件接收标识(hardware receive handle,HRH)标识,HRH 或 HTH 和硬件 MB 是一一对应的关系。在 CANIF 模块中 HTH 和 HRH 映射到接收协议数据单元(receive protocol data unit,RxPdu)和发送协议数据单元(transmit protocol data unit,TxPdu),它们之间存在如图 3 所示的两种映射关系。1)$N$ 个 RxPdu 可以从同一个 HRH 接收 CAN 报文,从而将接收到的报文分发到数个上层模块;2)$M$ 个 TxPdu 可使用一组 $N$ 个 HTH 发送,多个 TxPdu 可以用同一个 HTH 从而节省硬件资源,一个 TxPdu 可以对应一个 HTH 保证单个报文发送的可靠性,一个 TxPdu 可以对应多个 HTH 实现多路发送,保证多帧发送的可靠性。

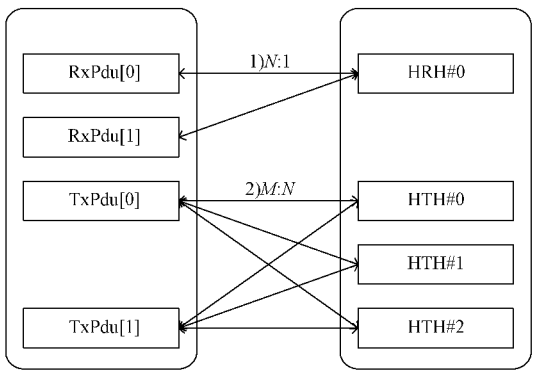


图 3 硬件抽象映射

### 2.4 发送功能设计

CAN 通信栈完整的发送流程包含发送请求和发送确认两个步骤:上层模块调用 CANIF 的 CanIf_Transmit()发起发送请求,然后 CANIF 将调用 CAN 驱动的 Can_Write()将 CAN 报文写入控制器 MB;下层模块发送完成后将回调 CANIF 的 CanIf_TxConfirmation()通知发送完成。

CAN 驱动有两种发送完成通知方式即中断通知和轮

询通知：在 MPC5748G 中当 CAN 控制器发送报文完成时会置位相应的中断标志位，因此将回调通知方法嵌套到中断处理函数中即可实现中断发送通知；轮询发送确认方法是通过周期调用 Can_MainFunction_Write()检查中断标志位来判断是否发送完成，如果发送完成就回调通知上层发送完成。无论是轮询还是中断方式，CAN 通信栈的发送请求和发送完成通知都是保持异步进行，不会相互阻塞，保证了 AUTOSAR CAN 通信栈发送的低延迟。

当发送的硬件资源受限时需要多个 TxPdu 共用一个 HTH，此时发送请求失败的几率增加。在实际开发中可以将发送失败的处理交给上层应用模块，但这会影响上层的应用逻辑，因此可以在通信栈底层设计通用的应对策略来解决发送失败的问题，本文根据 AUTOSAR 规范[14]设计发送缓冲机制作为一种发送失败处理策略。

在本文设计的发送缓冲机制中，如果在发送请求时硬件发送对象繁忙，发送失败的 PDU 报文会存入发送缓冲区，在硬件对象发送完成并通知时从发送缓冲区中取出该 PDU 再次发送，从而避免 PDU 报文的丢失。这里提出两种发送缓冲机制：第 1 种是先进先出缓冲队列，当发送失败时将 PDU 入队，如果发送缓冲区已满就删除最旧的 PDU，当发送完成通知时从队列中取最旧的 PDU 发送；第 2 种是为每个 PDU 在缓冲区中设置一个确定的缓存位置，当发送失败时如果缓冲区中已经存在该 PDU 则覆盖原本的位置，否则存入空闲位置，当发送完成时选择缓冲区中最高优先级 PDU 发送。

第 1 种发送缓冲机制保证一段时间的 PDU 不会丢失，但是也导致控制器的发送整体延迟，在使用时要考虑系统的实时性要求。第 2 种发送缓冲机制总是保证缓冲区中存储的是最新的数据，避免第 1 种方法的实时性问题，但是会丢弃部分 PDU 且不会通知上层这些报文被丢失，适合实时性要求高的系统，另外选择最高优先级 PDU 发送避免了内部优先级反转。因此，第 2 种方法更适合汽车这类时效性要求高的系统。事实上，发送缓冲机制也可以选择两种方法混合的设计，对不同需求的 PDU 选择不同的缓存策略，使得 CAN 通信软件更加灵活。

CANIF 的发送接口 CanIf_Transmit()设计如图 4 所示。1)首先确保 CANIF 模块已经完成初始化；2)要求指定的 CAN 控制器处于 STARTED 工作状态；3)要求 CAN 控制器通信状态处于 ONLINE 或 TX_ONLINE，根据 2.2 节的描述只有这两个状态才能发送报文；4)调用 Can_Write()发送 CAN 报文，在 Can_Write()会检查 MB 是否空闲，如果 MB 空闲就将报文存入并返回 CAN_OK，否则返回 CAN_BUSY；5)如果使能发送缓冲机制，则 Can_Write()返回 CAN_BUSY 的报文将暂存到 CANIF 的发送缓冲区。

### 2.5　发送取消设计

CAN 报文的优先级倒置表现为高优先级报文失去和低优先级报文竞争的机会，出现这种情况会导致高优先级报文的延迟[15]。当多个 TxPdu 共用一个硬件发送对象时

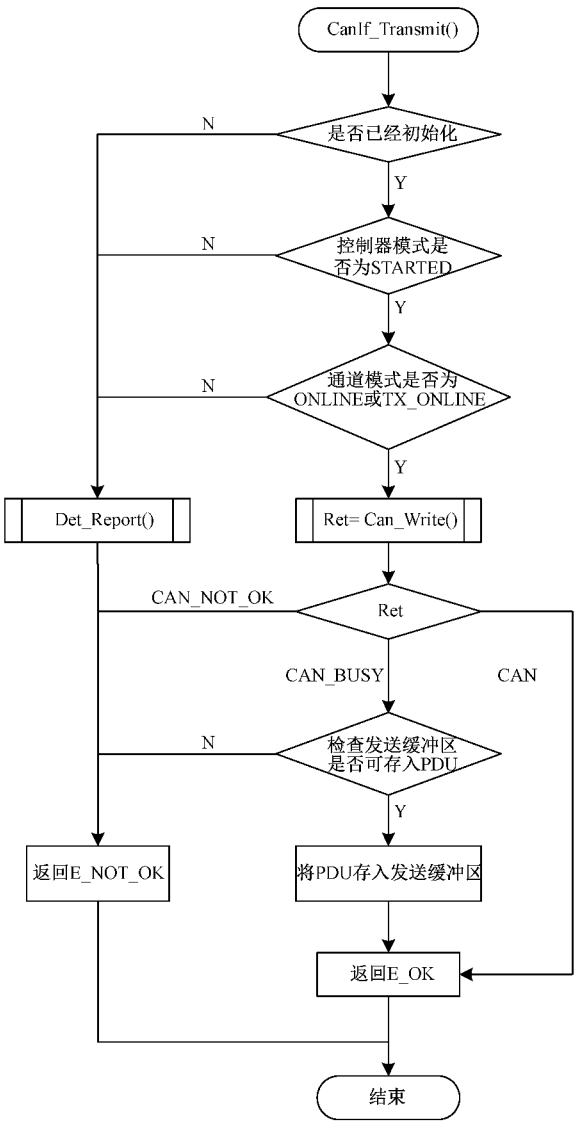


图 4　报文发送逻辑

可能会出现 ECU 内部的优先级反转：低优先级报文先存储到硬件发送对象，但是由于网络中存在高优先级报文导致发送延迟，此时如果同一个 ECU 打算发送高优先级报文，但是硬件发送对象中已经有待发送的低优先级报文，导致高优先级报文被低优先级报文阻塞造成发送失败，从而形成内部优先级反转。优先级反转会导致无法预测的故障，例如紧急的高优先级报文由于 ECU 的内部反转导致发送延迟，导致报文失去时效性。

为避免内部优先级反转，本文根据 AUTOSAR 规范设计发送取消机制如图 5 所示。如果硬件发送对象有尚未发送完成的报文，就比较当前存储的报文和请求发送报文的优先级；如果请求发送报文的优先级更高就取消当前硬件发送对象中的报文并将高优先级报文存储到发送缓存，在下次轮询主函数中发送高优先级报文并将取消发送的报文存储到发送缓冲，从而避免内部优先级反转。本文设计的

发送取消机制需要 CAN 控制器支持硬件发送对象取消功能,即取消尚未发送的硬件发送对象。

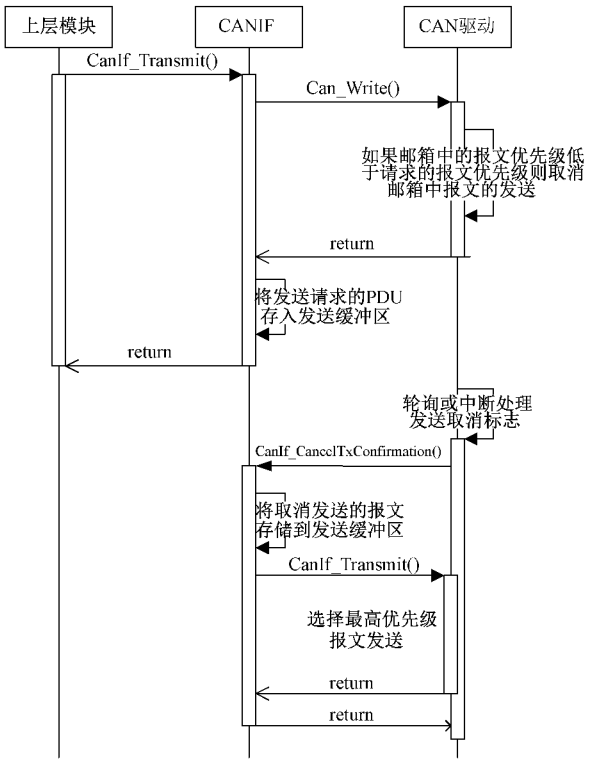


图 5　发送取消流程

### 2.6　接收功能设计

CAN 通信栈接收功能可以通过中断或轮询方式进行:当 CAN 控制器接收到报文并通过 MB 的硬件过滤时会置位相应的接收中断标志,在中断模式下会触发接收中断并在中断处理函数中回调 CanIf_RxIndication()通知 CANIF 接收到报文;在轮询模式下会检查到中断标志位是否置位,如果置位会回调 CanIf_RxIndication()通知 CANIF。

CAN 通信栈 CanIf_RxIndication()的工作流程如图 6 所示,在该回调接口中需要进行报文过滤、数据长度验证和报文分发。完整的报文接收过滤由 CAN 控制器硬件接收过滤、HRH 接收过滤和 RxPdu 接收过滤组成。CAN 控制器硬件可以只接收指定 ID 范围的报文,以 MPC5748G 内置 CAN 控制器为例,接收模式的 MB 可以设置接收掩码和期望的接收报文 ID,当接收到报文时 CAN 控制器可以直接抛弃不符合要求的报文。每个 HRH 可以配置一组 CAN 报文 ID 范围,只要接收到的报文 ID 满足其中任一范围即可接收。每个 RxPdu 可以设置单一报文 ID 或者一组 CAN 报文 ID 范围,只有接收报文的 ID 是指定 ID 或满足一组 CAN 报文 ID 范围时才接收,并将过滤得到的报文路由到上层模块。数据长度验证功能是为了避免接收到长度错误的报文,当接收到的报文的数据长度大于等于配置的接收数据长度时通过验证,该功能可以防止上层应用接收到无法处理的长度的报文。

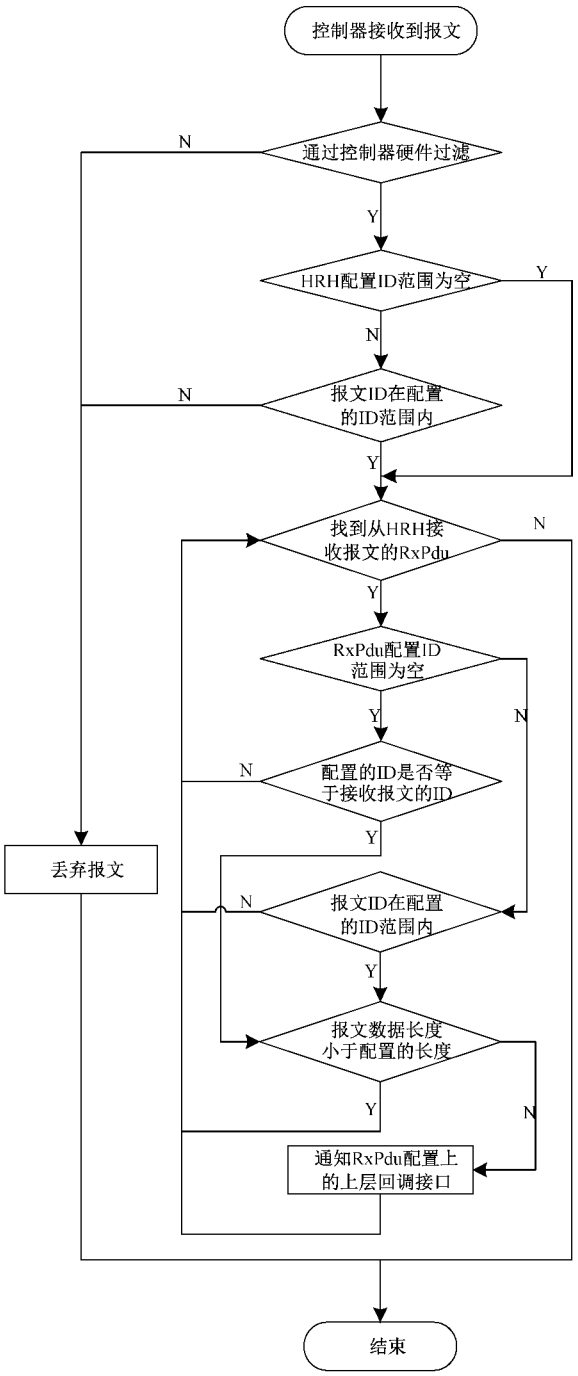


图 6　报文接收逻辑

## 3　实验验证

在本节将通过实验对本文实现的 AUTOSAR CAN 协议栈底层进行测试,实验硬件平台为 MPC5748G 开发板,单片机工作主频为 160 MHz,CAN 总线波特率为 500 kbit/s。实验对照组为直接驱动 CAN 控制器的轮询发送和本文设计实现的 AUTOSAR CAN 通信栈的轮询发送,用上述两种方式以 10 ms 为周期发送报文 ID 分别为 0x10 和 0x60 的 CAN 报文,0x10 为高优先级报文,0x60 为低优先级报

文。本文设计实现的 AUTOSAR CAN 通信栈主函数的轮询周期为 1 ms。

通过 CANoe 上位机的虚拟节点周期发送 CAN 报文(报文 ID 为 0x30)来控制总线负载率,以 30%负载率为例,持续记录 10 min CAN 总线上的报文并通过脚本处理得到图 7 和 8。周期延迟图是由指定 ID 报文与前一个该 ID 报文之间的时间差值减去确定的周期 10 ms 得到的结果,横坐标是报文序号,这些报文从 10 min 内截取的连续报文,纵坐标是周期误差(单位 ms)。

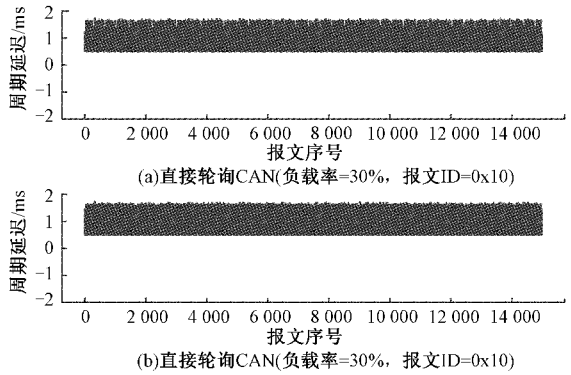


图 7　直接轮询 CAN 发送周期延迟

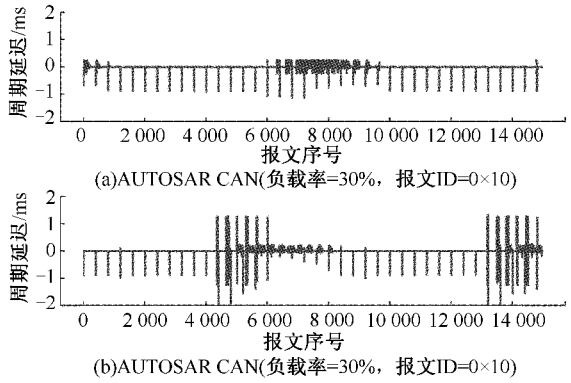


图 8　AUTOSAR CAN 轮询发送周期延迟

如图 7(a)和(b)所示,直接轮询发送的高优先级和低优先级的延迟相近,这是由于高优先级报文和低优先级报文是连续发送的,导致高优先级报文总是会被低优先级报文的发送延迟影响。而如图 8(a)和(b)所示,本文实现的 AUTOSAR CAN 轮询发送不同优先级报文的延迟差异明显,图 8(a)中发送的高优先级报文的发送延迟明显低于图 8(b)中发送的低优先级报文,这是由于发送取消机制避免了内部优先级反转,确保高优先级报文优先获得发送权,从而降低高优先级报文的整体延迟,但是低优先级报文的延迟却会增加。

图 8(a)中报文采用 AUTOSAR CAN 周期发送的平均延时要低于图 7(a)中的直接驱动方式,这是因为直接轮询发送是同步发送和确认,而 AUTOSAR CAN 轮询发送是异步发送和确认,避免了 ECU 内部等待报文发送完成的延迟。

图 8(a)和(b)中的报文提前发送是因为 AUTOSAR CAN 轮询发送是实时操作系统的周期任务执行,任务的周期不会因为报文没有发送完成而暂停,因此上一帧报文的发送延迟会导致为下一帧报文的发送超前。直接轮询发送采用定时器延时,因此发送延迟会直接影响到发送周期,才会出现图 7(a)和(b)的结果。

比较 AUTOSAR CAN 和直接轮询 CAN 的高优先级报文在总线负载率为 30%、50%和 70%的平均延时如表 2 所示,可以看到 AUTOSAR CAN 在本实验用例中可以降低 90%左右的发送延时。比较 AUTOSAR CAN 和直接轮询 CAN 的低优先级报文在总线负载率为 30%、50%和 70%的平均延时如表 3 所示,可以看到 AUTOSAR CAN 在本实验中降低 17%左右的发送延时,证明异步发送和发送确认机制的性能优于直接轮询发送,且发送取消机制可以有效减少高优先级报文的周期发送延迟。

**表 2　不同负载率下高优先级报文的平均延时**

| 负载率/% | AUTOSAR CAN/ms | 直接轮询/ms | 降低比例/% |
|---|---|---|---|
| 30 | 0.253 | 1.722 | 85.31 |
| 50 | 0.253 | 2.760 | 90.83 |
| 70 | 0.283 | 3.886 | 92.72 |

**表 3　不同负载率下低优先级报文的平均延时**

| 负载率/% | AUTOSAR CAN/ms | 直接轮询/ms | 降低比例/% |
|---|---|---|---|
| 30 | 1.336 | 1.719 | 22.28 |
| 50 | 2.274 | 2.758 | 17.55 |
| 70 | 3.277 | 3.886 | 15.67 |

## 4　结　　论

针对汽车电子控制器 CAN 通信软件长期存在代码可复用性差和通信性能低的问题,本文基于 AUTOSAR4.0.3 标准设计并实现代码复用性好和通信性能高的 CAN 通信栈。首先,为提升代码复用性,本文设计了 CAN 控制器模式控制、CAN 通信约束以及邮箱的抽象映射,实现通过统一接口访问和控制多个 CAN 控制器。然后,为提高通信栈性能,本文设计了异步发送和确认机制、发送取消机制和接收过滤功能:异步发送和确认机制在不使用中断时减少因等待发送完成带来的延时;发送取消机制避免报文发送的优先级反转,减少低优先级报文阻塞高优先级报文的可能;接收过滤功能实现报文多级过滤和分发。最后通过实验证明本文实现的 CAN 通信栈的性能高于传统 CAN 通信软件。下一步计划是在本文实现的 CAN 通信栈上实现 CAN 传输协议,从而拓展 CAN 通信栈的适用范围,进一步提高代码的复用性和软件开发效率。

## 参考文献


[1] SANTOS R, NETO R, COSTA R, et al. Automotive network management impacts on vehicle key-off load current [R]. SAE Technical Paper, 2021, DOI: 10.4271/2020-36-0006.

[2] AUTOSAR GBR. AUTOSAR layer software architecture R4.0.3[S]. AUTOSAR GbR, 2013.

[3] 孟天闯,李佳幸,黄晋,等.软件定义汽车技术体系的研究[J].汽车工程,2021,43(4):459-468.

[4] 孙升,宋珂,章桐.AUTOSAR 标准发展及应用现状[J].机电一体化,2014(12):33-38,44.

[5] 杜常清,朱体刚.电动汽车动力系统 CAN 网络设计及测试分析[J].自动化与仪表,2016,31(7):34-39.

[6] 王跃飞,胡京津,韩江洪,等.基于 EDF 的汽车 CAN 网络动态调度机制设计[J].电子测量与仪器学报,2014,28(8):819-826.

[7] WASICEK A, HOFTBERGER O, ELSHUBER M, et al. Virtual can lines in an integrated mpsoc architecture[C]. 2014 IEEE 17th International Symposium on Object/Component/Service-Oriented Real-Time Distributed Computing, 2014:158-165.

[8] 余庆,张晓先,戴柔逸.AUTOSAR 通信模块的设计与实现[J].计算机工程,2011,37(9):51-53,56.

[9] 余其涛.基于 AUTOSAR 标准的 CAN 通信栈设计与实现[D].上海:上海交通大学,2016.

[10] 李璐璐.车载 CAN 网络数据链路层和传输层软件的设计与实现[D].成都:电子科技大学,2013.

[11] 吴震云,崔书浩,余世运.车载控制器网络占用过高导致网络延时问题分析[J].汽车实用技术,2021,46(6):26-28.

[12] 马天才,许建森.基于 AUTOSAR 标准的 UDS 诊断通信[J].机电一体化,2020,26(5):34-40.

[13] AUTOSAR GBR. Specification of CAN driver R4.0.3[S]. AUTOSAR GbR, 2013.

[14] AUTOSAR GBR. Specification of CAN interface R4.0.3[S]. AUTOSAR GbR, 2013.

[15] 杨福宇.CAN 优先级倒置原因与对策[J].单片机与嵌入式系统应用,2012,12(4):1-4.


## 作者简介


**李超超**,硕士研究生,主要研究方向为汽车嵌入式总线协议。

E-mail:lcc135@mail.ustc.edu.cn

**武恪**,硕士研究生,主要研究方向为嵌入式系统信息安全与应用。

E-mail:wke@mail.ustc.edu.cn

**方菱**(通信作者),副研究员,博士,主要研究方向为嵌入式系统、形式化方法验证。

E-mail:fangl@hfcas.as.cn