一文讲懂 UDS 诊断协议

原创 已于 2025-11-03 13:41:21 修改 · 3.4k 阅读 · 50 · 51 · CC 4.0 BY-SA版权

文章标签： #网络 #服务器 #人工智能

在汽车电子 领域，当 ECU（电子控制单元）出现故障、需要标定参数或读取运行数据时，工程师靠什么与 ECU “对话”？答案是 UDS 诊断协议。作为汽车行业通用的诊断标准，UDS（Unified Diagnostic Services，统一诊断服务）就像 ECU 的“通用语言”，让不同厂商、不同类型的 ECU 都能被统一管理。今天我们从基础到实战，全面拆解 UDS 诊断协议。

一、先搞懂：UDS 是什么？为什么需要它？

在 UDS 出现前，汽车诊断是“各自为战”的局面 —— 不同车企、不同供应商会自定义诊断协议（比如某品牌用“0x01”代表读取故障码，另一品牌可能用“0x0A”），导致诊断仪无法通用，维修和开发效率极低。

UDS 的核心价值，就是**制定一套统一的应用层诊断标准**，解决“协议不兼容”问题。它基于 ISO 14229 系列标准（ISO 14229-1 定义核心服务，ISO 14229-2 定义 CAN 总线实现，ISO 14229-3 定义以太网实现），规定了 ECU 与诊断仪（或上位机）的通信规则：比如“用什么指令读取故障码”“用什么格式反馈数据”“如何保障通信安全”，让诊断仪能跨品牌、跨 ECU 类型通用。

简单说，UDS 的本质是：**汽车诊断领域的“通用 API”** —— 诊断仪发送“标准化指令”，ECU 返回“标准化响应”，无需再为不同 ECU 定制专属诊断逻辑。

二、UDS 的核心：7 个常用诊断服务（必学）

UDS 定义了几十种诊断服务（用“服务 ID”区分，占 1 字节），但实际开发中常用的只有 7 种。掌握这些服务，就能应对 90% 以上的诊断场景。

1. 会话控制服务（0x10）：切换 ECU 的 “工作模式”

ECU 的诊断模式不是固定的，不同场景需要切换“会话”（类似手机的“普通模式”“飞行模式”），**0x10 服务就是用来切换会话的。**

常见会话类型及用途：

会话子功能（1 字节）	会话名称	用途
0x01	默认会话	车辆正常运行时的基础诊断（如读取故障码、读取基本数据），权限最低
0x03	扩展会话	开发 / 维修时的深度诊断（如读取详细运行日志、标定参数），权限高于默认会话
0x02	编程会话	给 ECU 刷写程序（如升级固件），仅在 ECU 未运行核心功能时可用
0x04	安全会话	执行高权限操作前的“过渡会话”（如解除安全锁定）

实例流程：诊断仪要读取 ECU 的详细标定数据（需扩展会话），需先发送：

- 诊断请求：0x10 0x02（0x10 = 会话服务，0x02 = 扩展会话）
- ECU 响应：0x50 0x02（0x50 = 会话服务响应，0x02 = 确认进入扩展会话）

→ 只有切换到扩展会话，后续才能执行高权限操作。

2. 安全访问服务（0x27）：防止 ECU 被 “非法操作”

ECU 的核心功能（如刷写程序、清除故障码）不能随便被访问，**0x27 服务就是“身份验证”机制**，只有通过验证，才能执行高权限操作。

核心逻辑：**种子 - 密钥（S**

上一篇：一文讲懂Autosar网
下一篇：UDS诊断服务-10 服

车载软件开发测试

1. 诊断仪发送 “请求种子” 指令：0x27 0x01（0x01 = 请求种子）；
1. ECU 生成随机 “种子”（如 0x12 0x34），返回给诊断仪：0x67 0x01 0x12 0x34（0x67 = 安全服务响应，0x01 = 确认请求种子）；
1. 诊断仪用预设算法（如异或、AES 加密）将 “种子” 转换成 “密钥”（如 0x12 0x34 → 0x56 0x78），发送 “提交密钥” 指令：0x27 0x02 0x56 0x78（0x02 = 提交密钥）；
1. ECU 验证密钥：正确则返回0x67 0x02（验证通过），错误则返回否定响应（如 0x7F 0x27 0x35，0x35 = 密钥不正确）。

应用场景：刷写 ECU 程序前，必须通过 0x27 服务验证，防止恶意篡改固件。

3. 读取故障码服务（0x19）：ECU “报故障” 的核心方式

当 ECU 检测到异常（如传感器短路、执行器失效），会生成 DTC （Diagnostic Trouble Code，故障码，如 P0301=1 缸失火）并存储。**0x19 服务就是用来读取这些故障码的。**

常用子功能：

- 0x02：读取 “当前存在的故障码”（ECU 当前正在发生的故障）；
- 0x04：读取 “历史故障码”（ECU 曾经发生过、现在已恢复的故障）；
- 0x06：读取 “永久故障码”（无法通过清除指令删除，需修复硬件后自动清除）。

实例流程：读取当前故障码

- 诊断仪请求：0x19 0x02
- ECU 响应：0x59 0x02 0x01 P0301（0x59 = 故障码服务响应，0x02 = 当前故障码，0x01 = 故障码数量，P0301 = 具体故障码）

4. 读取数据服务（0x22）：获取 ECU 的 “实时数据”

ECU 运行时会实时采集很多数据（如发动机转速、水温、车速），这些数据被分配到 “数据标识符（DID，Data Identifier）” 中，**0x22 服务通过 DID 读取指定数据。**

DID 是关键：每个数据项对应唯一的 2 字节 DID，比如：

- 0x0100：发动机转速（单位：rpm）；
- 0x0101：发动机水温（单位：℃）；
- 0x0102：当前车速（单位：km/h）。

实例流程：读取发动机转速（DID=0x0100）

- 诊断仪请求：0x22 0x01 0x00（0x22 = 读取数据服务，0x01 0x00=DID）
- ECU 响应：0x62 0x01 0x00 0x07 0x08（0x62 = 读取数据响应，0x01 0x00=DID，0x07 0x08 = 转速数据，换算：0x0708=1800 → 1800rpm）

5. 写入数据服务（0x2E）：修改 ECU 的 “配置参数”

某些 ECU 参数（如怠速目标值、报警阈值）需要在开发或维修时调整，**0x2E 服务通过 DID 写入指定参数**（注意：需先通过 0x27 安全验证，否则会被拒绝）。

实例流程：修改怠速目标值（DID=0x0200，目标值 = 800rpm，十六进制 = 0x03 0x20）

- 诊断仪请求：0x2E 0x02 0x00 0x03 0x20
- ECU 响应：0x6E 0x02 0x00（0x6E = 写入数据响应，确认写入成功）

6. 清除故障码服务（0x14）：删除 ECU 的 “故障记录”

当故障修复后，需要清除 ECU 中存储的故障码，**0x14 服务就是 “清除指令”**（需先通过 0x27 安全验证，且部分永久故障码无法清除）。

车载软件开发测试

实例流程：清除所有可清除故障码

- 诊断仪请求：0x14 0xFF 0xFF（0xFF 0xFF = 清除所有可清除故障码）
- ECU 响应：0x54（确认清除成功）

7. 控制 DTC 设置服务（0x85）：开启 / 关闭 ECU 的 “故障检测”

某些场景下（如 ECU 标定、测试），需要临时关闭故障检测功能（避免正常测试被误判为故障），**0x85 服务就是用来控制 DTC 的使能 / 禁用。**

实例流程：禁用所有 DTC 检测

- 诊断仪请求：0x85 0x03 0xFF 0xFF（0x03 = 禁用，0xFF 0xFF = 所有 DTC）
- ECU 响应：0xCA 0x03 0xFF 0xFF（确认禁用成功）

三、UDS 通信的 “通用规则”：请求与响应格式

不管是哪种服务，UDS 的通信都遵循固定格式，核心是 “请求帧” 和 “响应帧” 的结构（以 CAN 总线为例，CAN ID 通常为诊断专用 ID，如 0x7E0 = 诊断请求，0x7E8 = 诊断响应）。

1. 诊断请求帧格式（诊断仪→ECU）

字节位置	含义	说明
1	服务 ID（SID）	如 0x10 = 会话控制，0x19 = 读取故障码
2	子功能（SF）	可选，服务的细分功能，如 0x10 服务的 0x02 = 扩展会话
3~8	数据参数（Datas）	可选，服务所需的参数，如 0x22 服务的 DID、0x2E 服务的写入数据

例：读取发动机转速（0x22 服务，DID=0x0100）的请求帧：0x22 0x01 0x00（共 3 字节，无额外参数）。

2. 诊断响应帧格式（ECU→诊断仪）

响应分两种：**肯定响应**（操作成功）和**否定响应**（操作失败）。

（1）肯定响应格式

字节位置	含义	说明
1	肯定响应 ID（PRID）	服务 ID + 0x40，如 0x10→0x50，0x19→0x59，0x22→0x62
2	子功能（SF）	与请求帧的子功能一致（若请求有子功能）
3~8	响应数据（Datas）	服务的返回数据，如 0x19 服务的故障码、0x22 服务的读取数据

例：0x22 服务的肯定响应：0x62 0x01 0x00 0x07 0x08（0x62=0x22+0x40，0x01 0x00=DID，0x07 0x08 = 转速数据）。

（2）否定响应格式（ECU 拒绝请求时）

字节位置	含义	说明
1	否定响应标识	固定为 0x7F
2	被拒绝的服务 ID	如请求 0x27 服务被拒绝，此处为 0x27
3	否定响应码（NRC）	表示拒绝原因，共定义 40 + 种，常见如 0x11 = 服务未支持、0x22 = 参数不正确、0x35 = 密钥错误

例：请求 0x27 服务（安全访问）但密钥错误，ECU 返回：0x7F 0x27 0x35（0x35 = 密钥不正确）。

四、UDS 的 “进阶知识”：关键技术点

1. 否定响应码（NRC）

NRC 是 UDS 调试中最常用

车载软件开发测试

NRC 码	含义	典型场景
0x11	服务未支持	向 ECU 发送它不支持的服务（如 ECU 没有编程功能，却发送 0x34 刷写服务）
0x22	请求参数不正确	读取数据时 DID 错误（如 DID=0x9999，ECU 无此 DID）
0x33	安全访问被拒绝	未通过 0x27 安全验证，就执行高权限操作（如直接发送 0x14 清除故障码）
0x35	密钥不正确	0x27 服务提交的密钥与 ECU 计算的不一致
0x78	资源暂时不可用	ECU 正在执行其他高优先级任务（如刷写程序），无法响应诊断请求

2. 数据长度与传输：应对 “长数据” 的问题

UDS 单帧（CAN 总线）最多只能传输 8 字节数据，但有些场景需要传输长数据（如刷写程序时的固件数据、读取大量故障日志），这时需要**多帧传输（ISO 15765-2 标准）**。

多帧传输分两种：

- **首帧（FF）**：第 1 字节 = 0x10（表示首帧），第 2-3 字节 = 总数据长度，后续字节 = 首段数据；
- **连续帧（CF）**：第 1 字节 = 0x20 + 帧序号（如 0x21 = 第 1 帧连续帧，0x22 = 第 2 帧），后续字节 = 分段数据。

例：传输 100 字节数据，首帧为0x10 0x00 0x64 ...（0x00 0x64=100 字节），后续用连续帧0x21 ...、0x22 ...直到传输完成。

3. 超时管理：防止 “通信卡住”

ECU 收到诊断请求后，需要在规定时间内响应，否则诊断仪会判定 “通信超时”。UDS 规定了默认超时时间：

- 普通服务（如 0x19、0x22）：500ms 内响应；
- 复杂服务（如 0x34 刷写程序）：可延长至 10s 甚至更久（需提前通过 0x83 服务协商超时时间）。

五、UDS 的实际应用：从开发到维修

UDS 不是 “纸上谈兵”，而是贯穿汽车全生命周期的核心技术，主要应用在 3 个场景：

1. 研发阶段：ECU 调试与标定

- 工程师用诊断仪（如 Vector VN1630）通过 UDS 读取 ECU 的实时数据（如传感器信号），验证功能是否正常；
- 通过 0x2E 服务修改标定参数（如喷油脉宽），优化 ECU 性能；
- 通过 0x19 服务监控测试中的故障，快速定位问题。

2. 生产阶段：ECU 刷写与配置

- 汽车下线时，工厂通过 UDS 的 0x34（请求下载）、0x36（传输数据）、0x37（请求退出编程）服务，给 ECU 刷写量产固件；
- 通过 0x2E 服务写入 “车辆配置信息”（如 VIN 码、车型参数），让 ECU 适配具体车辆。

3. 维修阶段：故障诊断与修复

- 4S 店技师用诊断仪（如元征 X431）通过 0x19 服务读取故障码，快速判断故障部位（如 P0301→1 缸失火，检查火花塞或点火线圈）；
- 修复后通过 0x14 服务清除故障码，确认故障已解决；
- 若需升级 ECU 固件，通过 UDS 刷写新程序（如解决软件 BUG）。

六、UDS 的未来：与以太网 、智能化结合

随着汽车向 “电动化、智能化” 升级，UDS 也在不断演进：

1. 从 CAN 总线走向以太

车载软件开发测试

传统 UDS 基于 CAN 总线（速率最高 1Mbps），无法满足智能化汽车的大数据传输需求（如自动驾驶 ECU 的海量日志）。现在 UDS 已支持以太网（ISO 14229-3），传输速率可达 100Mbps 甚至 1Gbps，能高效传输长数据（如固件镜像、高清故障视频）。

2. 与功能安全、网络安全深度融合

- 功能安全（ISO 26262）：UDS 需支持 “安全相关故障码” 的快速读取（如自动驾驶 ECU 的传感器故障），确保故障能及时触发安全机制；
- 网络安全（ISO/SAE 21434）：0x27 安全访问服务的加密算法从简单异或升级为 AES-128，防止黑客破解密钥、篡改 ECU 数据。

3. 支持远程诊断（OTA）

现在车企可通过 OTA（远程升级）技术，远程发起 UDS 诊断：比如车辆出现故障时，云端平台通过 4G/5G 向 ECU 发送 UDS 请求，读取故障码并分析，无需车主到店即可完成初步诊断，甚至远程修复软件故障。

总结

UDS 诊断协议是汽车电子的 “通用语言”，它通过标准化的服务、请求响应格式，解决了不同 ECU 的诊断兼容问题。掌握 UDS，不仅能看懂 ECU 的 “故障报告”，还能灵活调试、配置 ECU，是汽车电子工程师的核心技能之一。

本文从基础概念到实际应用，梳理了 UDS 的核心知识点，但 UDS 还有很多细节（如用户自定义服务、不同总线的实现差异）需要结合实际项目深入学习。如果你在 UDS 调试中遇到具体问题（如否定响应码排查、多帧传输异常），欢迎在评论区交流！

显示推荐内容

显示推荐内容

2 条评论 vhuffffffhuygcvbb 热评 四.2.那的首帧描述有点问题，首帧字节1的高四位固定是1没问题... 写评论

车载软件开发测试