汽车诊断服务（UDS）基本原理

一、 UDS 诊断方法

现代汽车内部普遍使用了数十个以上的控制器（Electronic Control Unit，ECU），用于控制发动机、变速器、底盘、仪表、空调、灯光、车窗座椅、车锁等部件，从而使得车辆具有良好的动力性、安全性和舒适性。但是大量电子设备的使用也造成了车辆电气故障排除的复杂性，因此大多数 ECU 都装有自诊断系统（On-Board Diagnostics, OBD），能够及时发现传感器、执行器、ECU 和通信网络产生的电气故障，以大幅提高故障诊断的效率。当 OBD 检测到故障时，会生成相应的故障码信息，并点亮故障灯。此时，把诊断仪插入 OBD 接口，即可以和 OBD 系统进行通信，从而获取故障码和相关信号流等数据，实现快速定位故障源和故障类型。

由于全球有大量的汽车和诊断仪生产商，所以诊断仪和 OBD 系统之间的通信都遵循统一的汽车诊断服务协议（Unified Diagnostic Services，UDS），以实现诊断仪和 OBD 系统之间诊断数据的有效传输，如图 1 所示：

图 1 UDS 功能示意图

UDS 把所有的 OBD 诊断功能规范成 26 种诊断服务，归为 6 大类，每种服务用 SID 进行编号，如表 1 所示：

表 1 UDS 诊断服务类型

服务类别	SID（0x）	诊断服务名	服务 Session Control
诊断和通信管理	10	诊断会话控制	Diagnose Session Control
	11	ECU 复位	ECU Reset

功能单元	27	安全访问	Secure Access
	28	通讯控制	Communication Control
	3E	待机握手	Tester Present
	83	访问时间参数	Access Timing Parameter
	84	安全数据传输	Secured Data Transmission
	85	控制 DTC 的设置	Control DTC Setting
	86	事件响应	Response On Event
	87	链路控制	Link Control
数据传输功能单元	22	通过 ID 读数据	Read Data By Identifier
	23	通过地址读取内存	Read Memory By Address
	24	通过 ID 读比例数据	Read Scaling Data By Identifier
	2A	通过周期 ID 读取数据	Read Data By Periodic Identifier
	2C	动态定义标识符	Dynamically Define Data Identifier
	2E	通过 ID 写数据	Write Data By Identifier
	3D	通过地址写内存	Write Memory By Address
存储数据传输功能单元	14	清除诊断信息	Clear Diagnostic Information
	19	读取故障码信息	Read DTC Information
输入输出控制功能单元	2F	通过 ID 控制输入输出	Input Output Control By Identifier
例行程序功能	31	例行程序控制	Routine Control
上传下载功能单元	34	请求下载	Request Download
	35	请求上传	Request Upload
	36	数据传输	Transfer Data
	37	请求退出传输	Request Transfer Exit
	38	请求文件传输	Request File Transfer

从表中可知，这些服务涵盖了 ECU 数据流的读取、ECU 故障码读取与清除、ECU 程序更新等 OBD 诊断功能。

为了便于描述，把通信的双方分别定义为客户端和服务端。诊断仪一般充当客户端角色，ECU 一般充当服务端角色。客户端和服务端采用“请求服务—响应服务”的方式完成一次故障诊断服务，既首先由客户端向服务端发出诊断服务请求消息，服务端接收到请求消息后再给

出诊断服务响应消息。服务响应消息又分为正响应和负响应两种，分别对应服务端接收服务请求和拒绝服务请求两种情况。

UDS 服务请求消息和服务响应消息都以报文的形式呈现。UDS 明确定义了服务请求报文的基本格式为“SID+其他参数“，正响应报文的基本格式为“[SID+0x40]+其他参数“，负响应报文的格式为“0x7F+其他参数“。图 2 所示的是一个读取油门开度信息的例子（其报文内容只作为解说用，不具有真实意义）。

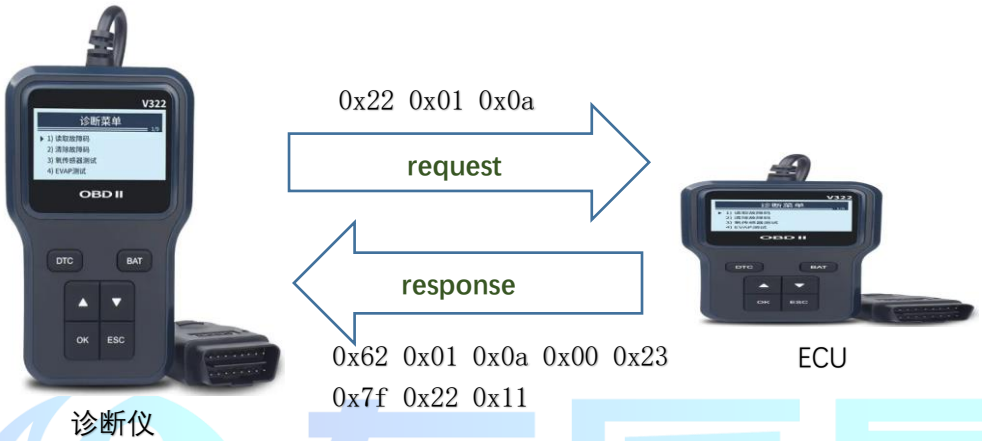


图 2 读取油门开度服务过程示意图

在上图中，诊断仪向 ECU 发出了请求报文“0x22 0x01 0x0a”，该报文中的第一个数据“0x22”表示当前请求的服务为 SID 为 22 的“通过 ID 读数据”服务，该报文中的“0x01 0x0a”表示要读的数据的 ID 为“010a”，其含义为油门开度，由汽车生产商指定。如果 OBD 给出了正响应，则发送报文“0x62 0x01 0x0a 0x00 0x23”，该报文的第一个数据值”0x62”是 0x22 加上 0x40 的结果，后面第二、三个数据为请求报文中指定的 ID 值“0x01 0x0a”，第四、五个数据 “0x00 0x23”为当前油门开度值“0x23（十进制为 35）”，表示当前发动机油门开度为 35%。如果 OBD 给出了负响应，则发送报文“0x7f 0x22 0x11”。该报文的第一个数据值为“0x7f”，表示拒绝服务，后面第二个数据为被拒绝的服务的 SID 值“0x22”，第三个数据为拒绝服务的原因，该值由 UDS 定义，“0x11”表示当前服务不支持。

UDS 协议定义了所有服务的报文格式和参数含义（见 ISO 14229-1），诊断仪和 OBD 都必须遵循该协议，才能正确完成各种诊断服务。参照 ISO 网络通信体系，该部分协议称为 UDS 应用层协议。

二、 UDS 传输方法

诊断仪和 OBD 生成的 UDS 报文需要通过通信网络进行传输。目前汽车 ECU 普遍使用 CAN 总线进行数据通信，因此 UDS 报文必须加载到 CAN 帧中才能发送。但是每个 CAN 帧的最大传输数据量只有 8 个字节，而 UDS 报文的数据量是根据服务内容进行变化的。最小的 UDS 报文有 2 个字节，大的 UDS 报文可以达到几千个字节以上。这意味着超过 8 个字节的 UDS 报文需要多个 CAN 帧才能完成传输。为了保证报文传输的有效性和可靠性，需要定义 UDS 报文在 CAN 总线上的传输协议（ISO-15765-2），其一般称为 UDS 的网络层或传输层协议。

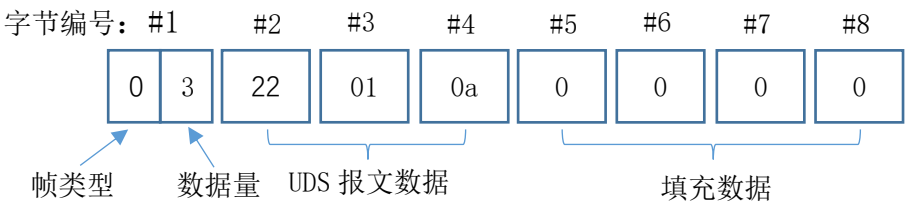
UDS 定义了四种类型的 CAN 帧来传输 UDS 报文，分别是单帧（Single Frame, SF）、首帧（First Frame, FF）、 连续帧（Consecutive Frame, CF）和流控帧（Flow Control, FC）。以上四种 CAN 帧的内容格式如表 2 所示：

表 2 UDS CAN 帧格式

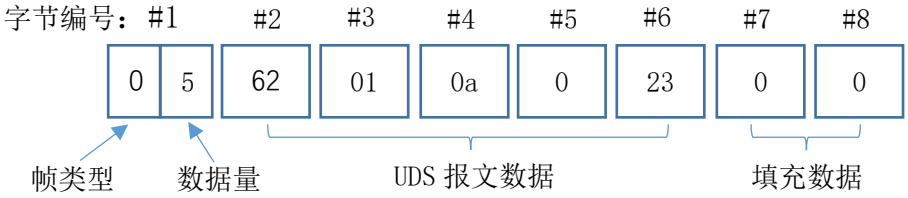
N_ PDU name	N_ PCI bytes			
	Byte #1		Byte #2	Byte #3
	Bits 7-4	Bits 3-0		
SingleFrame (SF)	N_ PCItype = 0	SF_ DL	N/A	N/A
FirstFrame (FF)	N_ PCItype = 1	FF_ _DL		N/A
ConsecutiveFrame (CF)	N_ PCItype = 2	SN	N/A	N/A
FlowControl (FC)	N_ PCItype = 3	FS	BS	STmin

从表可知，每种 CAN 帧都占用第一个字节的高 4 位来存放帧类型数据（单帧是 0、首帧是 1、连续帧是 2、流控帧是 3）。

单帧的第一个字节的低 4 位存放了该帧传输的报文数据量，剩余字节存放 UDS 报文数据。以图 2 中传输的请求报文 “0x22 0x01 0x0a”和正响应报文“0x62 0x01 0xa 0x0 0x23”为例，其单帧数据区的内容如图 3 所示：



（a）请求报文的单帧



（b）正响应报文的单帧

图 3 单帧数据区内容（16 进制数）示意图

可以看到，单帧最多可以传输数据量为 7 个字节的 UDS 报文，不足 7 个字节则用约定的数据进行填充；当要发送的报文的数据量大于 7 个字节时，就需要使用首帧、连续帧和流控帧了，其运用机制如图 4 所示：

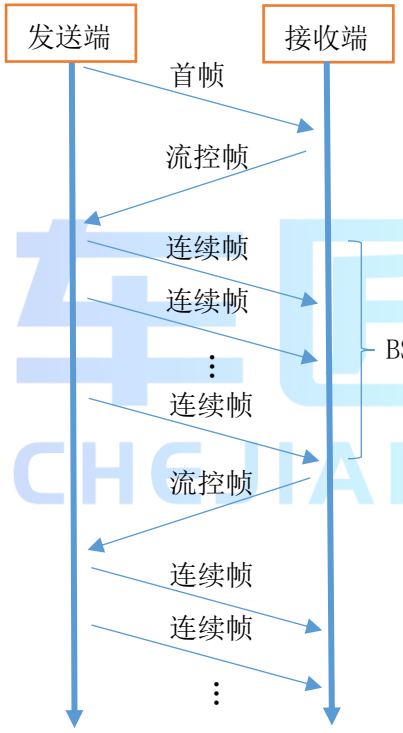


图 4 首帧、连续帧和流控帧运行机制示意图

从图中可知，发送端首先会把 UDS 报文数据填入首帧，填满后发出；接收端接收到首帧后回复流控帧，其作用主要是控制后续报文数据的发送速率；发送端接收到流控帧后会把 UDS 报文剩余的数据依次填入若干（BS 值）个连续帧，并按序发出，以此重复直到 UDS 报文数据全部发送完。

首帧的第一个字节的低 4 位和第二个字节存放了需要传输的 UDS 报文总数据量，剩余字节存放 UDS 报文前 6 个数据。

流控帧的第一个字节的低 4 位存放 FlowStatus(FS)值，该值如为 0 表示可以继续发送后续连续帧；如为 1 表示暂停发送后续连续帧；如为 2 表示接收端已经溢出。流控帧的第二个

字节存放 BlockSize(BS)值，该值如为 0 表示接收端后续将不会再发送流控帧，发送端直接发送余下的所有连续帧；如为 01-0xff 之间的某个值，表示发送端在连续发送了该值数量的连续帧个数后，需要等待接收方再次发送流控帧。流控帧的第三个字节存放 Stmin 值，该值为两个连续帧传输的时间间隔，单位 ms。流控帧的剩余字节存放填充数据。

连续帧的第一个字节的低 4 位存放连续帧的序号 SN 值。第一个连续帧的序号为 1，后续连续帧的序号依次加 1，直到 0xf 后，后续连续帧的 SN 值从 0 开始依次加 1。

以传输 17 位 VIN 码为例：

（1） 诊断仪发出“0x22 0xf1 0x90”的 UDS 报文，请求读取车辆 VIN 码；

（2） ECU 收到请求后回复正响应报文“0x62 0xf1 0x90 0x57 0x30 0x4c 0x30 0x30 0x30 0x30 0x34 0x33 0x4d 0x42 0x35 0x34 0x31 0x33 0x32 0x36”，共 20 个字节的数据量。因此 ECU 先发出首帧，其数据区内容为如图 5 所示：

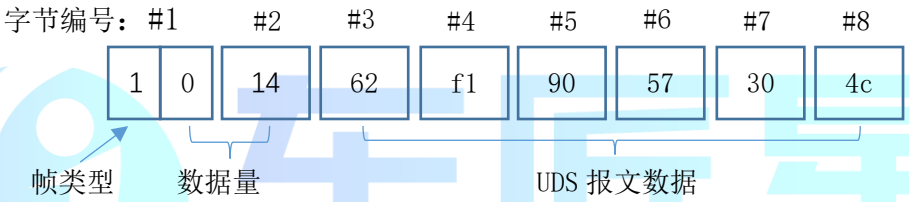


图 5 首帧数据区内容（16 进制数）示意图

（3） 诊断仪收到首帧后，向 ECU 回复流控帧，其数据区内容为如图 6 所示：

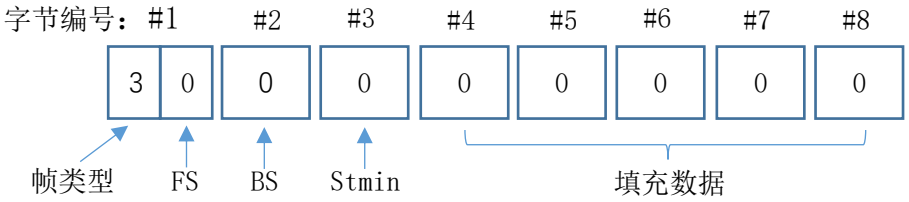
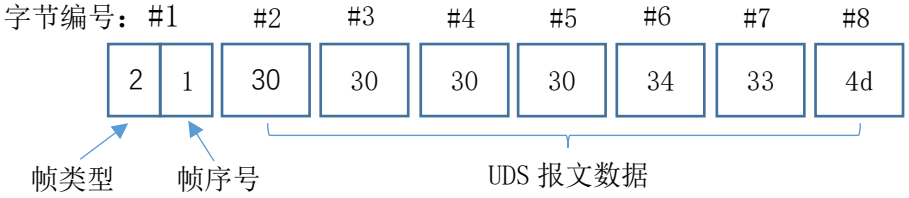
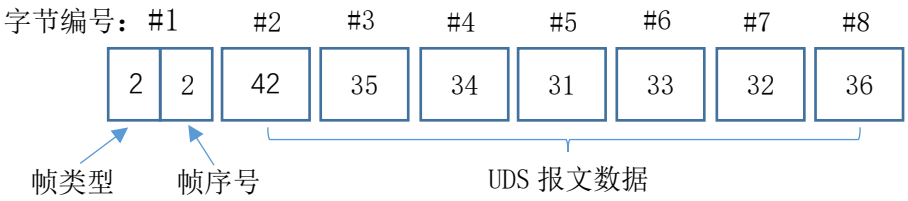


图 6 流控帧数据区内容（16 进制数）示意图

（4） ECU 收到流控帧后，会依次发送剩余 2 个连续帧，其数据区内容为如图 7 所示：



（a）第一个连续帧



（b）第二个连续帧

图 7 连续帧数据区内容（16 进制数）示意图

三、UDS 时间管理

UDS 在传输的过程中，可能会遇到因网络故障或节点故障导致的通信延迟或中断的情况，因此 UDS 定义了通信时间管理机制来保证其工作的时效性，并基于设定的时间参数来实现。以下列出几个基本的时间参数:

（1）UDS 应用层时间参数

- P2 Client：诊断仪成功发送请求报文之后，接收到 ECU 响应报文的时间间隔;
- P2 Server：ECU 接收到诊断报文请求后，发出回复响应报文的时间间隔;

如图 8 所示:

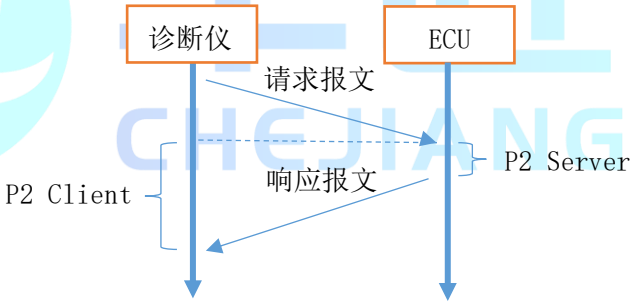


图 8 P2 Client、P2 Server 时间参数示意图

（2）UDS 网络层时间参数

- N_As：发送端发送一帧所需的时间，若超时表明发送端没有成功发出帧;
- N_Ar：接收端发送一帧所需的时间，若超时表明接收端没有成功发出帧;
- N_Bs：发送端等待成功接收流控帧的时间间隔，若超时表明发送端没有成功接收流控帧;
- N_Br：接收端等待发送流控帧的时间间隔，若超时表明接收端没有成功发送流控帧;
- N_Cs：发送端等待发送一连续帧的时间间隔，既 stmin;
- N_Cr：接收端等待成功接收一连续帧的时间间隔，若超时表明接收端没有成功接收连续帧;

如图 9 所示:

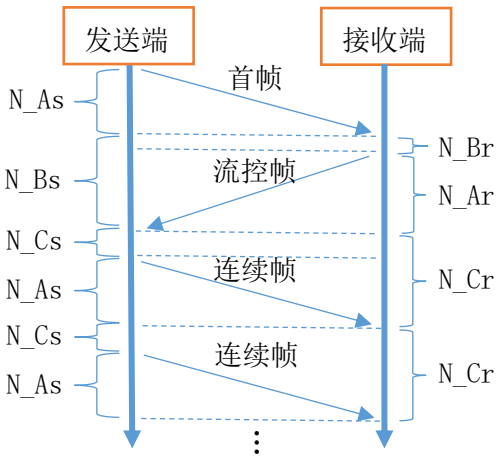


图 9 网络层时间参数示意图

UDS 协议明确给出了各个时间参数的取值范围，具体取值可根据网络环境和诊断应用需求进行调整。