

汽车统一诊断服务诊断协议栈网络层测试方法

罗峰^{1,2}, 郁静华^{1,2}

(1. 同济大学汽车学院, 上海 201804; 2. 同济大学新能源汽车工程中心, 上海 201804)

摘要: 提出了一种基于 CAN 总线的汽车统一诊断服务(UDS)诊断协议栈开发过程中的网络层功能测试方法. 利用 AutoCAN 总线设计工具, 将位于 PC 机的上层测试仪与搭载有被测协议栈的下层测试仪连接, 形成实际总线测试网络. 通过参数分层法, 结合实际被测协议栈通信要求, 建立测试用例集, 并据此编写测试脚本和仿真运行. 通过监控每一条测试用例的报文流记录, 判断其是否符合测试要求. 该方法能够实现 UDS 诊断协议栈的网络层测试, 验证网络层是否符合国际标准.

关键词: 统一诊断服务; 网络层; 功能测试; 参数分层方法
中图分类号: TP1 **文献标志码:** A

Approach for Network Layer Test in Vehicle Unified Diagnostic Service Diagnostic Protocol Stack

LUO Feng^{1,2}, YU Jinghua^{1,2}

(1. College of Automotive Studies, Tongji University, Shanghai 201804, China; 2. New Energy Automotive Engineering Center, Tongji University, Shanghai 201804, China)

Abstract: The paper presents an approach for network layer test, which is important during the development of unified diagnostic service(UDS) diagnostic stack. The upper tester in PC was connected to the lower tester, which carried the protocol stack under test, by a network-design tool called AutoCAN to build a test network. After analyzing the characteristics of network parameters and the demands of the device under test, a test case set was designed with the parameters classified method. Then the scripts were programmed and AutoCAN simulated the progress of all the designed communications. By monitoring every record of CAN frames transmission, it is clear to decide if this test meets the requirements of the specification. This method can accomplish the network-layer functional tests and find whether it works correctly based on the relevant international standards.

Key words: unified diagnostic service; network layer; functional test; parameters classified method

国际标准 ISO14229^[1], 即统一诊断服务(unified diagnostic service, UDS), 将各种基于不同总线网络的诊断服务进行了统一的规定, 从而使得在不同物理介质下, 对车辆的相关诊断服务的开发和应用变得更为便捷.

汽车 UDS 诊断协议栈是指符合统一诊断服务(UDS)标准, 能够实现车辆标准化诊断通信的系统. 对于每一个完成开发的 UDS 协议栈而言, 对其进行专业而规范的测试必不可少. 这能保证所开发的协议栈符合国际标准, 满足通信设计需求. 因此, UDS 协议栈的测试技术也是在汽车 UDS 开发过程中较为重要的一个方面.

目前, 国内外的一些研究人员及公司都对该技术有着一定的研究. 德国 Vector Informatik 公司对于该诊断协议栈, 有着一条完整的开发工具链. 使用测试工具 CANoe Option DiVa, 可根据导入的数据库文件, 自动生成测试用例后对整个协议栈进行自动化测试^[2]. 使用 CANdelaStudio, 则可建立诊断数据库文件^[3]. 以上方法均采用 Vector Informatik 公司的诊断工具链中的专业工具对协议栈进行测试, 专业便捷, 且使用较为广泛. 但其工具价格较为昂贵, 使用成本较高. 并且以上方法均为对整个 UDS 协议栈的测试, 而没有对协议栈中的单独网络层测试.

除上述方法外, 还使用 LabVIEW 工具, 设计了一个虚拟测试仪^[4], 它能够对协议栈的应用功能进行验证. 或使用 VC6.0 环境开发了一个测试上位机, 并且通过 USBCAN 接口卡, 将 PC 机连入总线网络进行测试^[5]. 以上方法所使用的工具成本较低, 易于获得, 但其对协议栈所做的验证工作, 并不全面

和规范. 只是针对协议栈所要求的应用功能做了简单验证, 并没有全面专业地将规范中一些关键处进行测试.

本文设计了一种 UDS 诊断协议栈的网络层功能测试方法. 该方法使用 AutoCAN 总线设计工具, 搭建实际测试网络. 通过参数分层法, 结合实际被测协议栈通信要求, 建立测试用例集并据此编写脚本, 仿真通信过程. 最后通过分析实际通信时的报文记录, 从而判断各条件下的单独网络层工作是否通过测试.

该方法无需高昂的专业软件费用, 并且可以单独对网络层进行测试. 因此在协议栈自下而上的开发过程中, 只要完成了网络层的部分功能, 即可马上测试已完成部分的功能, 而无需等待整个协议栈完成后再进行测试, 保证了开发的效率. 此外, 使用该参数分层方法进行用例设计, 使得一个协议栈的测试更加完整通用, 解决了上述部分文献中用例测试过于零散、不全面的问题.

1 测试体系架构

1.1 测试环境

本测试环境框如图 1 所示.

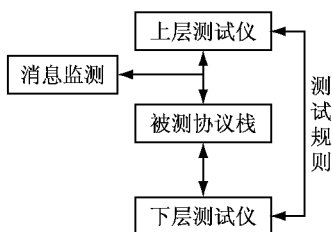


图 1 测试环境框图

Fig.1 Block of test environment

其中, 被测协议栈(protocol stack under test, PSUT)是搭载在下层测试仪(lower tester, LT)来进行测试实现的. 根据测试规范, LT 将特定测试用例的参数传给被测协议栈, 并给当前用例创造所需的运行条件. 上层测试仪(upper test, UT)则用来模拟 PSUT 使用者的行为, 测试中按照测试要求与 PSUT 进行通信, 从而验证 PSUT 是否正常工作. 消息监测则记录整个通信过程中的 CAN 报文流, 从而对照测试规范来判断本次测试是否通过. UT 与 LT 之间则通过测试规范来进行统一协调.

1.2 测试对象

本文测试对象为基于 CAN 总线的汽车 UDS 诊断协议栈网络层部分. 参考 ISO15765-2 标准中的描

述^[6], 可以将整个 UDS 协议栈大致分为 3 个层次, 结构如图 2 所示.



图 2 基于 CAN 总线的 UDS 诊断协议栈结构

Fig.2 Structure of UDS diagnostic stack based on CAN bus

其中, 最底层为 CAN 数据链路层和物理层, 能实现基本 CAN 报文的收发^[7]; 中间层是网络层, 能对上下层之间传输的数据进行打包或解包; 最上层是应用层, 能根据接收的数据, 执行相应的服务. 每一个层次都有对应的国际标准来规范使用.

网络层在整个 UDS 协议栈中起到了一个重要的桥梁作用. 因此, 在整个协议栈的开发中, 必须保证网络层能够按照标准中的要求正常工作, 这样才能有效地进行下一步开发.

一个正确的网络层包含以下几个方面: 上下层间服务原语工作正常; 单帧及多帧的传输操作正常, 数据有效; 上下层间数据映射正确; 定时参数要求满足; 能正确进行异常处理. 因此, 对于网络层的测试必须完整包含这几个方面, 才能确保测试结果的正确性.

2 测试组织设计

2.1 通用步骤

一个完整的测试实际是由很多个小的测试用例组成, 每一个测试用例针对被测对象的性能要求, 组织不同的测试环境, 对某一目标功能进行验证. 虽然测试用例内容不同, 但都遵循着一个通用的测试步骤, 从而有序规范地进行测试. 本文的每一个测试用例都遵循以下 3 个通用步骤: ①测试前, 建立当前测试用例的测试环境, 即根据本条测试规则, 修改下层测试仪对应参数值, 并载入对应用例的上层测试仪测试脚本. ②测试时, 打开报文记录仪, 运行测试工具仿真. ③测试后, 根据所记录的报文流, 对照本条用例测试规范, 判断是否通过, 并在报告记录结果及必要的分析. 至此则完成了一个有效的用例测试. 依照此通用步骤依次进行, 则可完成整个测试.

2.2 用例设计

测试用例的设计是整个测试规范制定当中较为

重要的一部分. 好的用例可以较为完整地反映被测对象的性能, 判断是否符合设计规范要求, 发现可能的潜在错误并及时修正.

在本测试中, 因被测对象为 UDS 网络层协议栈, 本身可配置参数较多, 功能要求也较为复杂, 因此, 此处采用一种分层结构, 将各种功能或主要参数合并分类, 理清相互之间的关系, 从而能够较为有效地设计出合理的测试用例.

参数分层结构如图 3 所示.

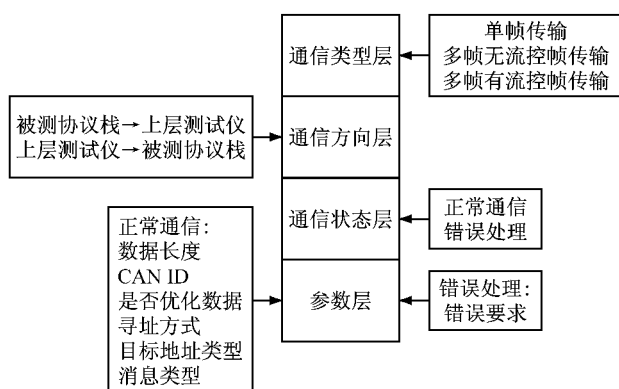


图3 参数分层结构

Fig.3 Structure of parameter group

如图3所示, 第1层为通信类型层, 分为单帧传输, 多帧无流控传输和多帧有流控传输. 单帧传输即为数据在单个 CAN 报文中就能够传输完成. 网络层能支持最高达 4 095 个字节的数据传输, 而 CAN 报文一次只能收发 8 个字节. 因此, 网络层通过多帧传输的方式, 也就是将数据拆分或组装来实现. 多帧传输可使用或不使用流控帧来完成通信. 第2层为通

信方向层, 分为被测协议栈至上层测试仪和上层测试仪至被测协议栈. 该分类保证了网络层在两个方向上都能够正常工作. 第3层为通信状态层, 分为正常通信和错误通信. 正常通信用于验证不同参数下网络层是否能够按照规范正常工作, 而错误通信则用来检查网络层能否进行正确的错误处理. 第4层为参数层, 其中可包含有剩余的各种不同参数.

当第3层为正常通信时, 第4层参数包括, 数据长度, CANID, 是否优化数据, 寻址方式, 目标地址类型和消息类型. 这些参数根据上面各层的参数选择, 按需要设置. 例如, 在特定的网络层要求下, 遍历本层某个关键的参数值而保持其他参数不变, 来检查网络层在此关键参数的每一种情况下都是否能正常工作.

当第3层参数为错误通信时, 第4层参数则为错误要求. 此参数根据上面各层的参数选择, 按需要进行设置. 例如, 在有流控情况下 LT 传输多帧时, 可设置 UT 在接收到首帧时不响应, 使得 LT 端 N_{Bs} 参数超时, 来验证此错误能否被正确检测到并处理. 或者, 多帧传输中, 设置后续帧的帧编号不连续, 从而检测接收方在收到错误序列帧时能否报告帧序列错误的情况.

根据以上的分层方法, 排列组合前3层参数, 并根据组合出来的每一种情况, 按实际被测网络层要求设置第4层参数, 最后可得到一张完整的网络层功能测试用例表, 如表1所示. 表中 OK 表示正确测试; ERR 表示错误测试.

表1 网络层功能测试用例表格式

Tab.1 Table of network functional test case

单帧传输				多帧无流控传输				多帧有流控传输			
UT->LT		LT-> UT		UT->LT		LT-> UT		UT->LT		LT-> UT	
OK	ERR	OK	ERR	OK	ERR	OK	ERR	OK	ERR	OK	ERR
.....											

需要说明的是, 并不是所有的变量都必须遍历. 在实际测试中, 可根据被测协议栈的具体要求, 只取关键变量进行测试即可, 而忽略无关变量. 例如在本文中, 目标网络层的消息类型均为诊断消息, 且寻址方式为混合寻址, 因此, 在形成测试用例时直接将这两个变量排除, 减小测试用例数量.

3 测试验证

3.1 测试环境建立

要完成整个实物功能测试, 首先必须建立一个

实际的测试环境. 使用飞思卡尔 XDP512 芯片作为测试系统中的下层测试仪, 上面搭载被测协议栈, 用 PC 机作为上层测试仪, 模拟协议栈使用者行为. 两者接入同一 CAN 网络, 建立通信.

测试环境实物连接图如图4所示.

其中, AutoCAN 为德国益驰公司的一款 CAN 总线设计测试工具, 它能够直接接入 CAN 网络, 用上位机向网络中发送或者接收报文, 对网络层进行监控及操作. 它还支持脚本操作, 通过编写相应脚本代码, 仿真被测协议栈用户的行为, 来验证协议栈工作是否正常. 测试环境中的消息监测工作也将通过

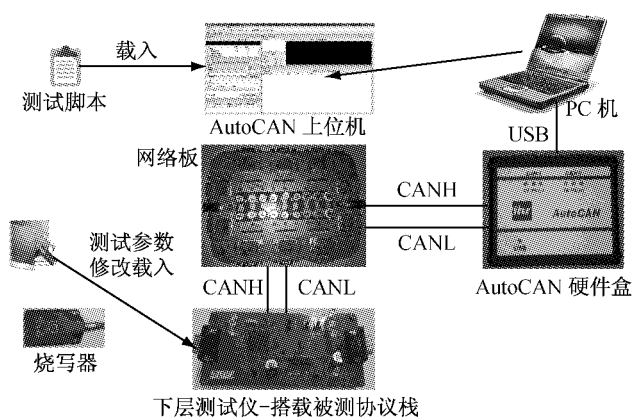


图4 测试环境实物连接图

Fig.4 Connection diagram for real test environment

AutoCAN本身的报文记录仪进行记录。

下层测试仪通过烧写器将测试参数的修改载入到下层测试仪中。而在PC端,则通过调用对应的AutoCAN测试脚本,来实现特定测试行为的配置。CANH和CANL为CAN总线的通信中的两种信号,CANH为CAN高,CANL为CANAL。

3.2 测试实现

以测试用例表中的某一用例为例,详细说明一个网络层测试的全过程。

3.2.1 规范描述

该测试用例用于验证:被测协议栈能否在有流控帧的情况下,正确向使用者发送多帧数据。

(1) 测试准备

测试环境正确建立。下层测试仪:配置协议栈,写命令令其上电3s后自行发送30个字节的数据。上层协议栈载入“有流控多帧接收”脚本。打开AutoCAN报文记录仪,等待开始。

(2) 测试进行

运行仿真后,给下层测试仪上电。3s后下层测试仪开始发送数据,通信开始,直至完成。

(3) 结果判断

根据报文流记录,判断上层端是否接收到完整的数据,且在接收到首帧后,上端向总线发出一个流控帧。各报文数据格式正确。

3.2.2 测试准备

首先对下层测试仪端做相关设置,使其发送的数据满足要求。主要参数修改如表2所示。

然后在PC端使用AutoCAN脚本仿真功能,来响应接收到的首帧。

在AutoCAN界面上选择“仿真”选项卡,并右键在通道1中“插入仿真节点”,完成后界面如图5所示。

表2 下层测试仪端变量值修改

Tab.2 Parameter value in lower test device

变量	值
message->msgLength	30
message->pMsysBuffer[0]	0
message->pMsysBuffer[1]	1
message->pMsysBuffer[2]	2
.....
message->pMsysBuffer[29]	29

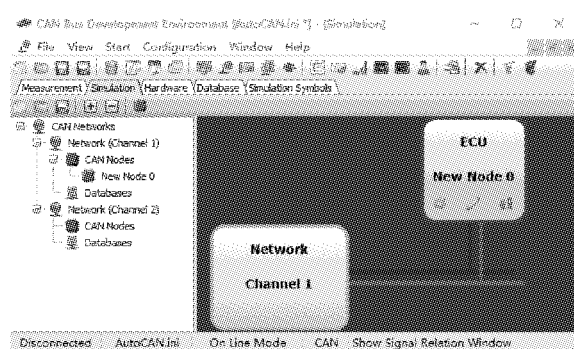


图5 仿真网络配置界面

Fig.5 Configuration interface for network simulation

接着对该仿真节点进行配置,主要是选择一个脚本文件,用于规定该节点在仿真过程中的一些执行动作。而后在脚本文件中编写相关代码,实现所需处理。

为了实现上述测试用例中的功能,在脚本文件中,需要定义一个CAN报文类型的变量,并将其配置成流控帧的形式。其函数如下所示。

Configure_message (&msg1, 0x654, 1, 1, 8, 0x03, 0x08, 0x04, 0xff, 0xff, 0xff, 0xff, 0xff); (1)

其中configure_message函数的参数依次表示所配置消息变量的地址,报文ID,是否为标准帧,是否为数据帧,数据场长度以及8个数据场数据。

然后在CAN报文事件中规定:一旦接受到CAN ID号为0x123的报文时,立即启动一个5ms

的定时器,当该定时器计数溢出后,则向网络上发送事先配置好的流控帧.当完成脚本,并编译通过后,即可等待测试.

3.2.3 测试进行

测试过程较为简单,只需点击 AutoCAN 上位机仿真按钮,并给下层测试板上电后,待其自行完成即可.

3.2.4 结果判断

报文完成发送后,即停止运行,并根据报文流记

录,来查看整个通信过程.

图 6 为仿真测试的报文流记录.从中可以看到,当 AutoCAN 接收到了中央控制单元(electronic control unit, ECU)端发来的首帧,并等待了一个时间间隔后,其向总线上发出一个流控帧.当 ECU 接收到了该流控帧后,则开始发送续帧直至 30 个字节的数据全部发送完成.最后的 4 个 FF 字节即为选择数据自动填充后,网络层对 CAN 报文数据场的自动填写数据.

452.54123	1	0123	数据帧	RX	8	10 1E 00 01 02 03 04 05
452.54869	1	0654	数据帧	TX	8	30 08 14 FF FF FF FF FF
452.55622	1	0123	数据帧	RX	8	21 06 07 08 09 0A 0B 0C
452.56122	1	0123	数据帧	RX	8	22 00 05 0F 10 11 12 13
452.606213	1	0123	数据帧	RX	8	23 14 15 16 17 18 19 1A
452.631221	1	0123	数据帧	RX	8	24 1B 1C 1D 1E FF FF FF FF

图 6 仿真测试的报文流记录

Fig.6 Message record for test simulation

整个过程符合本条测试规范的要求,从而得出,本条测试用例通过.

在某些测试中,得到的测试结果可能不能满足要求,则表示测试不通过.这种情况下,需要根据下层测试仪端变量的变化及报文流的记录,分析并定位协议栈的问题,修改后重新按照测试步骤执行,直至全部通过.

3.2.5 测试总结

根据上述步骤,可以对测试用例表中的用例一一测试.使用上文描述的参数分层法,测试用例表中共有 28 个测试用例,其涵盖了对网络层协议栈的主要功能,如单帧与多帧的收发,流控机制,不同数据流方向,是否具有填充数据等测试.并根据国际标准中规定的错误处理情况,人为模拟出不同的故障情况,进行错误测试.

通过 AutoCAN 工具及仿真脚本,能方便地控制网络中的数据行为,实现了单独网络层的功能测试.通过观察 AutoCAN 的报文记录以及在调试模式下查看下层测试仪中相关变量的改变情况,从而能够判断该条测试用例是否正确实施了,最后得出完整的网络层测试情况报告.

4 结论

设计了一种对汽车 UDS 诊断协议栈网络层进行功能测试的方法.首先给出了测试的通用架构及步骤,以及分层方法下测试用例集的设计.根据用例规范要求,编写测试脚本并使用 AutoCAN 工具进行实际网络测试,从而实现网络层功能的测试.

该方法具有较好的通用性,对测试用例的设计

可以根据具体的被测协议栈的要求进行变化,以达到最高的测试效率.对于总线工具的使用,只需选择支持脚本编辑的总线工具即可,无需使用特定的昂贵工具.从而提高了开发效率,也降低了开发成本.

参考文献:

- [1] International Organization for Standardization. ISO14229-1 Road vehicles-unified diagnostic services (UDS)-Part 1: Specification and requirements[S]. Geneva: ISO, 2013.
- [2] Peti Philipp, Timmerberg Armin, Pfeffer Thomas, *et al.* A quantitative study on automatic validation of the diagnostic services of Electronic Control Units [C]//IEEE International Conference on Emerging Technologies and Factory Automation, Hamburg: IEEE, 2008: 799-808.
- [3] 马莎,奚英泽,耿慧,等.汽油机 ECU 基于 ODX 的 UDS 数据库开发[C]//2013 中国汽车工程学会年会论文集.北京:中国汽车工程学会,2013:413-415.
MA Sha, XI Yingze, JI hui, *et al.* UDS database development of gasoline ECU based on ODX[C]//SAE-China, Annual Meeting Proceedings of SAE-China. Iasi: SAE, 2013: 413-415.
- [4] Salcianu M, Fosalau C. A new CAN diagnostic fault simulator based on UDS protocol [C]//2012 International Conference and Exposition on Electrical and Power Engineering (EPE). Iasi: IEEE, 2012: 820-824.
- [5] 韩鑫,鲍可进. CAN 总线网络层协议栈开发测试[J]. 计算机工程, 2011, 37(15): 232.
HAN Xin, BAO Kejin. Development and test of CAN bus network layer protocol stack[J]. Computer Engineering, 2011, 37(15): 232.
- [6] International Standard Organization (ISO). ISO15765-2 Road vehicles—diagnostics on controller area networks (CAN)—Part 2: network layer services [S]. Geneva: ISO, 2004.
- [7] International Standard Organization (ISO). ISO11898-1 Road vehicles—controller area network (CAN)—part 1: data link layer and physical signalling [S]. Geneva: ISO, 2003.