UDS 诊断教程（六）

这篇文章将介绍 InputOutputControlByIdentifier (0x2F) 和 RoutineControl (0x31) 这两个诊断服务的用途和用法。它俩的作用有点类似，都是调用 ECU 内部一些预定义的操作序列，相当于是我们从外部利用诊断手段控制 ECU 的接口。

InputOutputControlByIdentifier (0x2F)

ECU 简单来说就是一个对输入(sensor)进行计算再产生输出（actuator）的系统。2F 这个服务就是对 ECU 的输入和输出进行控制。这个服务在生产线上会需要使用，比如，在总装阶段，工人需要验证车上的各种功能是否正常，例如四个车窗的升降是否正常，如果挨个开关去按，那效率很低，如果通过一个诊断命令就能够观察到车窗升降的情况，效率则高得多。

ECU 就是一个处理输入信息、输出控制的系统

比如，ECU 接收一个输入信号 A，我们就可以利用 2F 给这个 A 赋个我们需要的值；ECU 对某个执行器 B 进行控制，我们就可以利用 2F 服务再配上某些特定的参数来实现对 B 的控制，例如门控对车窗升降、后视镜折叠等的控制。

2F 命令的格式

2F 服务的 request 由 4 部分组成

1. SID
2. dataIdentifier，用于标识被控制的 IO 对象
3. controlOptionRecord，用于标识控制方式，比如是启动、停止控制，还可以有一些自定义的参数来进行更精准的控制，比如让某个执行器的动作持续多长时间。controlOptionRecord 又分为两部分，分别是 1 个 byte 的 inputOutputControlParameter，以及若干 byte 由厂家自定义使用的 controlState。
4. controlEnableMaskRecord，这是一个可选参数，用于标识 controlOptionRecord 中的哪些 parameter 被使用。

UDS 明确定义了四种 inputOutputControlParameter

0x00 returnControlToECU （将控制权还给 ECU，即结束控制）

0x01 resetToDefault （将 dataIdentifier 所引用的输入信号、内部参数、输出信号等设为默认值）

0x02 freezeCurrentState（将 dataIdentifier 所引用的输入信号、内部参数、输出信号等冻结住）

0x03 shortTermAdjustment （将 dataIdentifier 所引用的输入信号、内部参数、输出信号进行设置，其实就相当于开始了对 ECU 的控制）

另外，UDS 定义可以用 22 服务读取 2F 服务中使用的 dataIdentifier，返回值是状态信息，具体的状态信息是什么，则由使用者自定义了。

我们以 14229 中举的一个例子来感受一下 2F 服务：

这个例子是使用 2F 控制 Air Inlet Door Position （进气口门位置），用标识符 0x9B00 来标识进气口门的位置。Air Inlet Door Position [%] = decimal(Hex) * 1 [%] ，即用一个百分比来表示这个位置。

step1:

tester 发送 22 9B 00 读取当前进气口门的位置

ECU 返回 62 9B 00 0A， 0x0A = 10（dec），表示当前位置是 10%

step2:

tester 发送 2F 9B 00 03 3C ，表示要将进气口门的位置调整到 60%，0x3C = 60（dec）

ECU 返回 6F 9B 00 03 0C，表示接受控制，当前进气口门的位置为 12%。因为 ECU 收到请求后是立刻响应的，而门的位置调节需要时间，所以还没有达到 60%。

step3:

过一段时间后 tester 发送 22 9B 00 读取当前进气口门的位置

ECU 返回 62 9B 00 3C， 0x3C = 60（dec），表示当前位置已经到了 60%

step4:

tester 发送 2F 9B 00 00，将控制权交还给 ECU

ECU 返回 6F 9B 00 00 3A，表示接受请求，当前位置为 58%

step5:

tester 发送 2F 9B 00 02，冻结 9B 00 这个 ID 所代表的进气口门位置这个状态

ECU 返回 6F 9B 00 02 32，表示接受请求，当前位置保持在 50%

RoutineControl (0x31)

31 服务是调用 ECU 内置的一些操作序列的接口，这个服务的应用很灵活，因为厂家可以根据自己的需要为 ECU 定义各种各样的内部操作，而要执行这些操作只需要调用 31 服务就好了。典型的用途包括检查边界条件、清除闪存、对数据进行较验、对软硬件依赖性进行校验等，甚至有需要的话可以进行恢复出厂设置的操作，还有很多与 ECU 自身逻辑功能相关的操作也可以定义。

31 命令的格式 2F 命令的格式

31 服务的 request 由 4 部分组成

1. SID
2. sub-function，用于标识要执行什么动作，启动（0x01）、停止（0x02）、查询结果（0x03）？
3. routineIdentifier，用于标识要执行的 routine
4. routineControlOptionRecord，这是一个可选参数，用于标识 routine 执行时所需要的参数，由各家自定义它的内容

举个例子，假设用 0x0809 这个 ID 来代表检查 ECU 是否满足软件刷写条件（比如车速、转速为 0，KL15 接通等）的 routine。

tester 发送 31 01 08 09 来启动 0x0809 这个 routine

如果所有条件都满足，则 ECU 返回 71 01 08 09 作为 echo 即可，如果条件不满足，则 ECU 返回 71 01 08 09 XX YY ZZ，后边的 XX YY ZZ 则表明哪些条件不满足，具体的内容就由厂家自己定义了。