长安汽车 CHANGAN	编号 Code	BTS-XX.03.02.02-A1-2015
	代替 Instead	
	发布日期 Release date	2015-2-10

FLASH BOOTLOADER 需求规范

FBL Requirement Specification

前　言

本规范起草单位:汽车工程研究总院电装中心软件所。

本规范主要起草人：殷小伟、张志勤、陈杰

本规范与上一版本相比，主要技术变化参见：版本变动与修改记录。

本规范历次发布情况：

——BTS-XX.03.02.02-A1-2015 于 2015 年 2 月 10 日首次发布。

版本变动与修改记录：

版本	更改描述	更改日期	更改人
1.0	初始发布版本	2012-09-25	殷小伟
1.1	1、修订 5.1、5.2 节引用章节序号； 2、修订 7.3.2 节引用表格序号。	2013-01-07	殷小伟
1.2	1、更新逻辑块和软件段解释； 2、修订 7.1 诊断对话管理图示和文字说明； 3、更新 7.2 flash 重编程时序相关要求（超时、诊断仪在线；取消可选步骤，优化编程流程）； 4、增加 7.2.4 子流程说明章节。	2013-04-19	殷小伟、刘志萌、张鹏
1.3	1、修订 7.2.2.4 节例程标识符笔误（0xFF 0x00 改为 0x02 0x02）。	2013-05-02	殷小伟
1.4	1、1.1 节增加目的性描述； 2、增加 1.2 节内容，增加 1.3 节内容； 3、增加 1.6 节“HW”至“VIN”描述； 4、增加 1.7 节内容，增加 1.8 节内容； 5、更新 5.2 节各服务细化描述；	2013-09-28	张志勤
1.5	1、5.1 节中加入“在引导程序中有 20ms 的延迟时间用于判断是否有诊断报文，如有则也可进入下载流程”描述； 2、STEP 1 取消 e、f 流程； 3、增加附录 CRC 算法	2014-01-08	陈杰
1.6	1、3.2 节第一段文字描述修改； 2、5.1 节中，图 3 流程中增加了 20ms 延时判断过程，删除了由编程模式向扩展模式切换的箭头，对第三段的文字描述进行了补充； 3、5.2.1 节中，第一段增加了文字描述，流程图中 a 步骤由功能寻址改为物理寻址，增加了步骤 g，所有的功能寻址请求报文改成肯定响应抑制请求； 4、图 5 中增加了读取 Bootloader 版本信息的步骤； 5、5.2.2.1 节中，将“三个字节种子和密钥”改成“四”，同时修改表 8，增加“security seed [byte4]”；	2014-02-10	陈杰

	6、5.2.2.2 节中，增加了的读 Bootloader 版本号的服务的相关信息； 7、5.2.2.3 节中，完善了指纹信息的定义，增加了表 11、12 来说明写指纹信息服务的请求格式和响应报文格式； 8、5.2.2.4 节中，对 Flash 驱动下载过程进行了补充说明； 9、5.2.2.6 节中，删除了所有请与求数据上传（35）服务相关内容的描述； 10、5.2.2.6.3 节第一段中，“该校验和通过最后一次请求下载服务……”，其中的“最后一次”改为“每次”， 11、对 5.2.2.7 节编程依赖检测重新进行了描述，表 19 中增加了“logical block numbe”两行，增加表 21 用于规范编程依赖检测否定响应格式； 12、5.2.4.1 节中，删除了“安全访问子流程说明”中的第 2、3 点； 13、图 8 中，将流程图中最后一步由“31 $01 $02 $02”改成“31 $01 $FF $01”，同时更改了图下面的说明文字； 14、图9中，步骤“RequestTransferExit $37”和“RoutineControl $31 $01 $02 $02”均提到判断条件前，图后面软件下载子流程说明中，增加了例程0x0202的详细说明。		
1.7	1、5.2.3 节中，图 6 中 ECUReset $11 $01(HardwareReset)；改为 ECUReset $11 $03(SoftwareReset)； 2、5.1中，图3中timeout处箭头指向更改，由指向Programsession改为指向Extendedsession; 3、表24、表25中，上电复位报文内容由“11 01”改为“11 03”，表24中，安全访问“27 11 、27 12”更改为“27 01 、27 02” 4、图 4 中第 i 步描述，由“DTCSettingType = off”改为“DisableRxAndTx”，图6中第d步描述，由“DTCSettingType = off”改为“EnableRxAndTx” 5、5.2.1.2 中内容由“5.2.2.2”改为“5.2.2.1” 6、5.2.1.3 节中，读数据服务的描述均增加“若无应用程序，则 Bootloader 给出肯定响应即可，响应报文填充 X 个字节的“FF”； 7、5.2.1.4 节中，增加“若无定义，则回复肯定响应即可。” 8、5.2.3.2 节中，“0x01”改成“0x03”； 9、表24、25中，通信控制处内容由“28 00 01 、28 03 01”更改为“28 00 03 、28 03 03”； 10、5.2.2.7 增加“启动编程依赖检查例程”相关的描述。	2014-02-14	陈杰

1.8	1、修正表23中不符合规范的否定响应格式，增加了最后一行内容“UDS定义了的否定响码按UDS规范”； 2、增加擦除内存服务的报文格式和肯定响应格式，如表13、表14； 3、表4、6、10中的“0-255”改成“0-9”,表11中第#4、#5、#6三个字节的类型由“ASCII”改成“Dec”； 4、5.2 节中，增加“在重编程时序中，功能寻址报文应重复发送 3 次，以确保所有控制器都收到并响应这些功能寻址请求。”	2014-02-20	陈杰
1.9	1、5.2 节中增加“服务器编程过程中允许被编程 ECU 不接收功能寻址报文” ； 2、图6增加步骤“b”，以及该步骤的文字说明； 3、5.2.2.6.2节中第3段完善对BlockSequenceCounter的描述； 4、图8中增加了循环下载数据过程和数据完整性校验过程；	2014-06-05	陈杰
2.0	1、图 4、图 6 中增加功能寻址描述：SuppressPosRspMsg、Request ID:0x7DF 2、修正了表 24、25 中不正确的地方； 3、表 13 中，删除擦除内存服务报文格式中定义的“dataFormatIdentifier”； 4、图 5 中，步骤 c 增加了读 FLB 规范版本号流程；增加表 12、13 描述 FBL 规范版本号报文格式；	2014-07-10	陈杰
2.1	1、5.2.2.4 文字描述更改。更改例程 0x0202 的定义，及其请求格式（如表 27）； 2、5.2.2.6.2、5.2.2.7 更改部分文字描述； 3、5.2.2.6.3 更改部分文字描述，删除表 19 中#2、#3 两行内容； 4、5.2.4.2 节中，更改图 8 下载循环流程，更改 Routine 0x0202 定义； 5、5.2.4.3 节中，更改图 9 下载循环流程。	2014-09-22	陈杰
2.2	1、3.2 节中第一段和倒数第二段文字描述修改； 2、5.1 第三段中“20ms 延时”改为“不小于 20ms 延时”，增加了关于重编程标志的描述； 2、将表 5 中的小写字母 a\b\c 改成大写； 3、删除 5.2.2.1 节中的内容，将原有内容合并到 5.2.4.1 节并顺序调整表格的序号； 4、表 22、23 中的 Message Direction 原来写反了，更正； 5、表 26 中，文字描述“逻辑块数量完整性错误”改成“应用有效性校验错误”	2015-02-06	陈杰

2.3	1、表 29、30 中，10 02 改为不支持功能寻址； 2、图 3 中 Program session 指到左边的箭头下方文字删除“or 10 01” 3、增加擦除内存请求 0xFF00 和 Routine 0x0202 的否定相应格式，如表 16、表 30. 4、表 12、13 中 copy-past error 更正。 5、图 7 中“Is the ECU already unlocked?”判断条件删除 6、5.2.1.3 节中，对第二段进行了补充说明，增加了第三段读取件号的需求。 7、表 31 中，将写指纹信息（F184）改为仅在 02 模式下支持。 8、图 4、6 及 5.2.1.7 中关于 28 服务的描述增加支持允许\禁止“常规报文”发送和接收。	2015-12-22	
2.4	1、增加 3.4 节中安全启动的要求。 2、增加表 2 引导程序和应用程序支持的编程模式表。 3、增加 5.2 节中 NCR78 的例外处理。 4、增加 5.2 节中对上位机重启次数的限制。 5、修改了 5.2.1.7 节中的文字描述。 6、增加了 5.2.2.7.2 节中 TransferRequestParameter 和 maxNumberOfBlockLength 参数关系的说明。 7、增加了 5.2.2.7.3 节中校验和错误时的措施描述 8、修改 5.2.4.1 节中解锁后再进行安全请求的要求 9、 增加了 5.2.1.3 节中关于硬件件号、硬件版本、软件件号的要求 10、更新 5.3 节中《引导软件诊断服务》和《应用软件诊断服务》 11、修改部分文字描述、调整了表格编号 12、删除了 5.2.1.4 编程预条件检查中“若无定义，则回复肯定响应即可”的描述。	2017-12-21	王宇扬、谢伯林
2.5	1、将 3.4 节中安全启动规范名称由《支持 FBL 刷写控制器的安全启动规范》修改为《控制器安全启动需求规范（支持 FBL 的控制器）》； 2、增加了 3.5 节对重编程 ECU 刷写速度的要求。 3、修改了 5.2.2.7.3 节第一段中对校验和的描述； 4、修改了 5.2.2.8 节中对应用程序有效性检查的描述。	2018-02-12	谢伯林

目　录

1 介绍

1.1 目的/范围

本规范适用于长安乘用车型项目的 CAN 网络控节点的 FBL 功能开发。

本规范规定了基于 UDS 的 Flash 重编程的需求。一个重编程过程包括标准接口和通信时序，需要以此来支持 ECU 开发阶段和使用阶段的软件更新。此规范提供了重编程的需求、架构和下载流程，以支持 ECU 的 Flash 的重编程。

本规范只适用于基于 CAN 的系统。其并不能代替已存在的关于诊断的标准文件，但是相对于这些标准而言，本规范增加了额外的需求和限制。如果存在冲突，本文件的优先级要高于标准文件。

对于所有可重编程和不可重编程的 ECU，此规范的内容都是强制的。任何偏离需要得到长安汽车股份有限公司的批准并须记录在相应的 ECU 诊断规范中。

本规范的目的是对架构工程师、系统设计人员、应用开发人员和供应商提供指导和设计要求。

备注：长安保留对与本规范相关的所有软硬件设计进行一致性验证的权利

1.2 目标

允许任何基于微处理器的ECU能够在长安任意的组装工厂和维修站被配置并且实现完全编程。

ECU 在生产线终端被部分或完全编程的目的是为了缩短在整个生产线流程中的最小时间。

ECU 应保证 Flash 存储器或 EEPROM 均可以进行软件修改。

建议所有存储在 ECU 的应用软硬件信息保存在车辆数据库中，可以通过 VIN 等从数据库中识别。

1.3 文档使用

尽管本规范希望覆盖软件下载实现各方面的需求，但是仍然可能有缺失或忽略的细节。当这些需求有更新后，将对本规范进行升级或更新。

下列引用文档中，与本规范冲突的，需以本规范为准。

1.4 规范性引用文件

下列文件中的条款通过本标准的引用成为本标准的条款。凡是注日期的引用文件，其随后的修改单（不包括勘误的内容）或修订版均不适用于本标准。凡是不注日期的引用文件，其最新版本适用于本标准。

ISO 14229-1 *Road vehicles - Unified diagnostic services (UDS) - Part 1: Specification and requirements*

ISO 15765-4 *Road vehicles - Requirement for emission related systems*

ISO 15765-3 *Road vehicles — Diagnostics on Controller Area Networks (CAN) - Part 3: Implementation of unified diagnostic services (UDS on CAN)*

ISO 15765-2 *Road vehicles - Diagnostics on Controller Area Networks (CAN) - Part 2: Network layer services*

《长安 UDS on CAN 总线诊断规范》

1.5 术语和定义

ISO 14229-1 确定的以及下列术语和定义适用于本标准。

1.5.1 客户端（Tester）

外部诊断工具（诊断仪），用来向 ECU 发送诊断请求。

1.5.2 服务器

响应外部诊断工具发送的诊断服务请求的 ECU。

1.5.3 指纹

诊断仪识别信息，用来识别确定的下载尝试。

1.5.4 逻辑块

软件逻辑块（logical block）是指，具有一定功能的软件集合。逻辑块之间可以存在相互依赖关系，也可以是能独立运行的。logical block 由一个或者多个 Segment 组成，这些 Segment 的地址可以是连续的，也可以是不连续的。

软件逻辑块（logical block）占用的 Flash 空间之间不能有交叉、重叠。在进行软件升级时，按照逻辑块的起始地址和长度进行整个逻辑块的擦除。

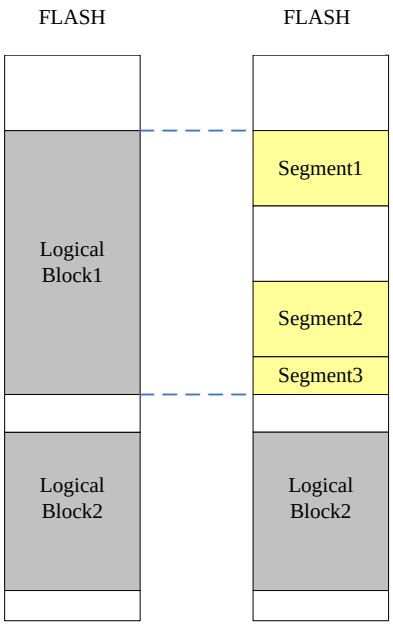

图1 逻辑块说明

1.5.5 软件段（Segment）

软件段（Segment）是指，一段地址连续的软件代码。

1.5.6 睡眠模式

在 ECU 空闲状态减少能耗的模式。

1.5.7 软件互锁

软件互锁是一种保护性封锁机制，通过将关键代码段与其它代码分离实现防止意外的软件执行，例如在错误发生后。

1.6 符号和缩写

ECU	电子控制单元
EEPROM	电可擦除可编程只读存储器
FBL	存储器引导装载器
DTC	诊断故障码
CAN	控制局域网络
CRC	循环冗余码校验
SWIL	软件互锁
LBT	逻辑块表
MCU	微控制器单元
TP	传输层
HW	硬件
M	必选
O	可选
SID	Service identifier，诊断服务标识符
SW	软件
TBD	To be defined，待定
VIN	Vehicle identification number，车辆认证号

1.7 编程ECU分级

对 ECU 可编程级别分为三种不同的架构：

1.7.1 通过Bootloader完全可编程ECU

这是一种典型支持可编程软件下载能力的 ECU。

这种 ECU 应具备完全可编程或支持 Flash 内存可编程。

1.7.2 通过固化的程序可编程ECU（部分可编程）

这是一种非典型的只支持部分可编程软件下载能力的 ECU。这种 ECU 只允许部分固定应用程序可进行编程。例如车载影音系统拥有固定应用程序部分可用于下载影音均衡

器数据。

1.7.3 非可编程ECU

这种 ECU 不具备支持软件下载功能。

1.8 编程电压

当 ECU 执行软件下载功能时，应不需要额外的外部供电。

2 重编程原理

一个可重编程的 ECU 包括两个可执行的软件包：ECU 应用程序和引导程序。在常规操作时，ECU 执行的是应用程序。引导程序仅在应用程序启动下载修改版本的引用程序或系统中没有可用的应用程序时激活。

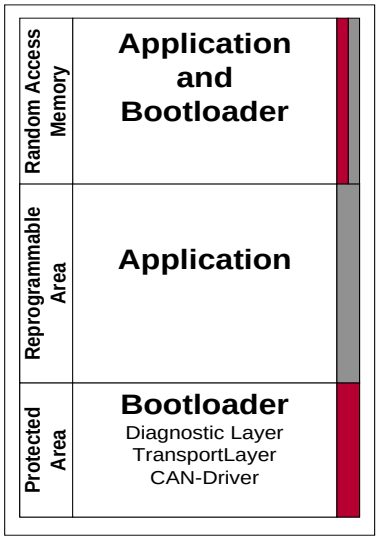

图2 可重编程 ECU 的内存映像

如图 2 所示，应用程序和引导程序占据了 Flash 的一个指定存储区域。当 ECU 执行引导程序或应用程序时，两个软件包都可以完整使用系统的 RAM 存储区。

引导程序使用了 UDS 的诊断服务，将其作为下载通信的协议。因此，引导程序须具有一个由 CAN 驱动、传输层和 UDS 协议层子集构成的通信协议栈。

3 重编程ECU要求

3.1 非可重编程ECU的需求

为使可重编程和非可重编程 ECU 能在一个网络中工作，非可重编程 ECU 应支持 5.3 节中规定的除诊断会话控制-编程会话请求外的诊断服务。

如果一个非可重编程 ECU 接收到诊断会话控制-编程会话（0x10 0x02）请求 ，须发送否定响应码 0x12（子功能不支持）。

3.2 一般需求

应用程序正常运行或应用程序跑飞时，系统应能够进入引导程序进行重编程。

在重编程尝试失败或被中断后，或发生超时或复位后，以及 Flash 被部分擦除或应用程序无效时，系统须是可重编程的。

如果应用程序代码丢失、无效或损坏，系统须执行引导程序。

系统执行引导程序时不允许干扰网络的常规通信。

所有连接到网络上且不处于重编程状态的 ECU，在接收到禁止常规报文发送请求后，须禁止其常规报文的发送（即，所有网络上的 ECU 应支持通信控制诊断服务）。在接收到允许常规报文发送的显式请求、返回至默认会话模式及上电/复位后，所有的 ECU 须允许常规通信。

所有连接到网络上的 ECU，在接收到禁止故障码设置请求后，须禁止其 DTC 设置。在接收到允许故障码设置的显式请求、返回至默认会话模式及上电/复位后，所有的 ECU 须允许故障码设置（即，所有网络上的 ECU 应支持控制 DTC 设置诊断服务）。

在完整的指纹数据被传送至 ECU 前，ECU 不允许启动 Flash 重编程。

一个逻辑块的所有下载数据传送完毕后，应该提供一个确切的校验值用于与 ECU 的计算值进行比较，以确保编程数据的完整性和正确性。系统中所有逻辑块重编程完成后，须进行程序有效性检查，详细的内容参考节 5.2.2.6 程序有效性检查。

3.3 资源需求

因为 ECU 上的可用的 Flash 存储空间由应用程序和引导程序共享，引导程序的 Flash 存储空间占用必须做到最小化。ECU 上可用的 RAM 可被引导程序完全占用，也可被应用程序完全占用。

当 ECU 处于供电模式下时，必须能够响应客户端的诊断服务请求。

3.4 安全需求

在控制器上电或复位启动前，需按照长安《控制器安全启动需求规范（支持 FBL 的控制器）》进行安全启动。如安全启动失败，则禁止刷写。

在重编程过程启动之前须成功通过安全访问服务。

Flash 驱动代码的关键部分不允许存储在 ECU 中，但是须在重编程过程中下载（软件互锁）。这保证了 Flash 烧写程序（擦和写）不会在常规操作模式下被偶然激活。

引导软件应该被存放在已进行保护的存储区域（软件和/或硬件保护）。

3.5 刷写速度

根据待刷写程序 bin 文件大小，控制器刷写速度需满足以下要求：

程序 bin 文件<100K，刷写速度≥100KB/分；

100K≤程序 bin 文件<300K，刷写速度≥150KB/分；

300K≤程序 bin 文件<1M，刷写速度≥250KB/分；

1M≤程序 bin 文件,刷写速度≥500KB/分。

4 网络层参数

4.1 CAN通信参数

寻址方式和 CAN 标识符分配查阅《长安 UDS on CAN 总线诊断规范》。

下载 CAN 报文的 DLC 固定为 8 个字节。

4.2 传输层参数

用于 Flash 下载的传输层的实现须基于《长安 UDS on CAN 总线诊断规范》。

传输层定时参数（N_As 和 N_Br 等）须按参考文档“CAN 诊断规范”中规定进行设置。

在执行程序刷写时，BS 和 STmin 须满足下表要求，如不能满足要求，需获得长安 FBL 部门认可。

表1 网络层参数要求

参数	符号	FBL 模式
持续发送次数	BS	0
最小间隔时间	STmin	0

5 诊断通信

Flash 重编程由诊断通信协议的子集控制。本章指定了诊断通信如会话模式管理，重编程时序和相关诊断服务的需求。

5.1 诊断对话管理

在 Flash 重编程的连接中，使用了三种不同的诊断会话模式：默认会话模式，扩展诊断会话模式和编程会话模式。

表2 引导程序和应用程序支持的编程模式表

	引导程序（Bootloader）	应用程序（Application）
默认会话模式	✓	✓
扩展诊断会话模式	✓	✓
编程会话模式	✓	-

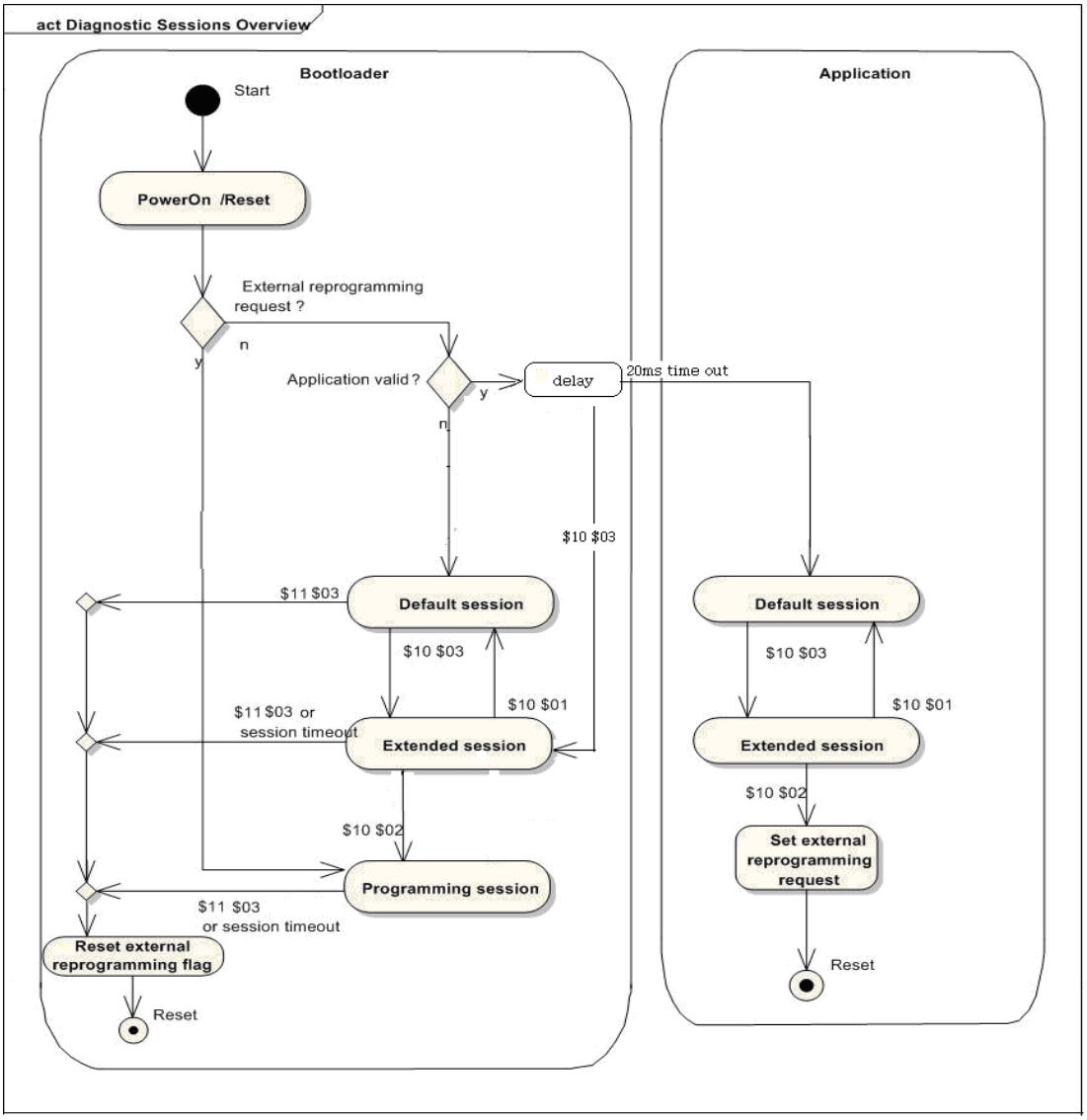

图3 诊断会话概览

如上图，描述了服务器（ECU）中引导程序（Bootloader）和应用程序（Application）会话模式切换的基本过程。

上电/复位发生后，ECU 首先执行引导程序，引导程序执行一些基本的初始化，其中必须完成对 CAN 的初始化，然后检查外部重编程请求标志是否已设置，如果标志已被设置，那么即使应用程序是有效的，引导程序也会继续进一步执行。外部重编程请求标志存放在 RAM 中被保护的固定地址中，当应用程序收到重编程请求时，将编程请求标志（比如：0x5555AAAA）写入该固定地址。

为了防止应用程序存在重大缺陷而不能跳转到 Bootloader 进行重编程，引导程序检测到应用有效后我们开启了一个不小于 20ms 的延时窗口，在此延时窗口期间，如 ECU 收到诊断报文（$10 $03），则亦可进入重编程流程。

如果当前没有重编程请求，则检验应用程序的状态。如果应用程序是有效的，引导

程序则启动应用程序，应用程序在默认会话模式中启动。如果应用程序无效，ECU 在默认会话模式继续执行引导程序并且等待转至扩展会话模式，再至用来 ECU 编程的编程会话模式。

引导程序可以通过默认会话请求离开扩展会话模式，进入引导程序默认会话模式。

服务器（ECU）的应用程序处于扩展模式时，客户端可以通过带有编程会话子参数的诊断会话控制服务请求激活服务器重编程。服务器在收到前述请求后，应用程序须设立外部的重编程请求标志并且执行复位。

编程模式可以通过 ECU 复位、默认模式和编程模式会话超时离开。

5.2 Flash重编程时序

Flash 重编程时序分为三个步骤：预编程步骤，编程步骤和后编程步骤。后面的章节详细描述了这三个步骤。

在重编程时序中，如果目标 ECU 有否定响应，则重启重编程时序。NCR78（请求已接收，等待回复）例外。

在重编程时序中，如果目标 ECU 有服务响应超时（P2CAN_Server 超时），则重启重编程时序。

在重编程时序中，客户端需要在 S3Server 超时前，执行一次功能寻址诊断设备在线服务，保持诊断会话模式。

在重编程时序中，功能寻址报文可能会重复发送多次，以确保网络上所有控制器都收到并响应这些功能寻址请求，被编程服务器应能正确响应客户端的每次请求。

编程步骤中允许被编程 ECU 不处理功能寻址报文。

重编程时序重启次数超过 5 次则上位机认为本次刷写失败。

5.2.1 预编程步骤

预编程步骤用来为编程过程准备网络条件，下位机应支持流程中的所有服务，上位机需按流程发送诊断报文。本规范使用的所有标识符格式参见及数据内容参见 5.3 节中《引导软件诊断服务》和《应用软件诊断服务》。

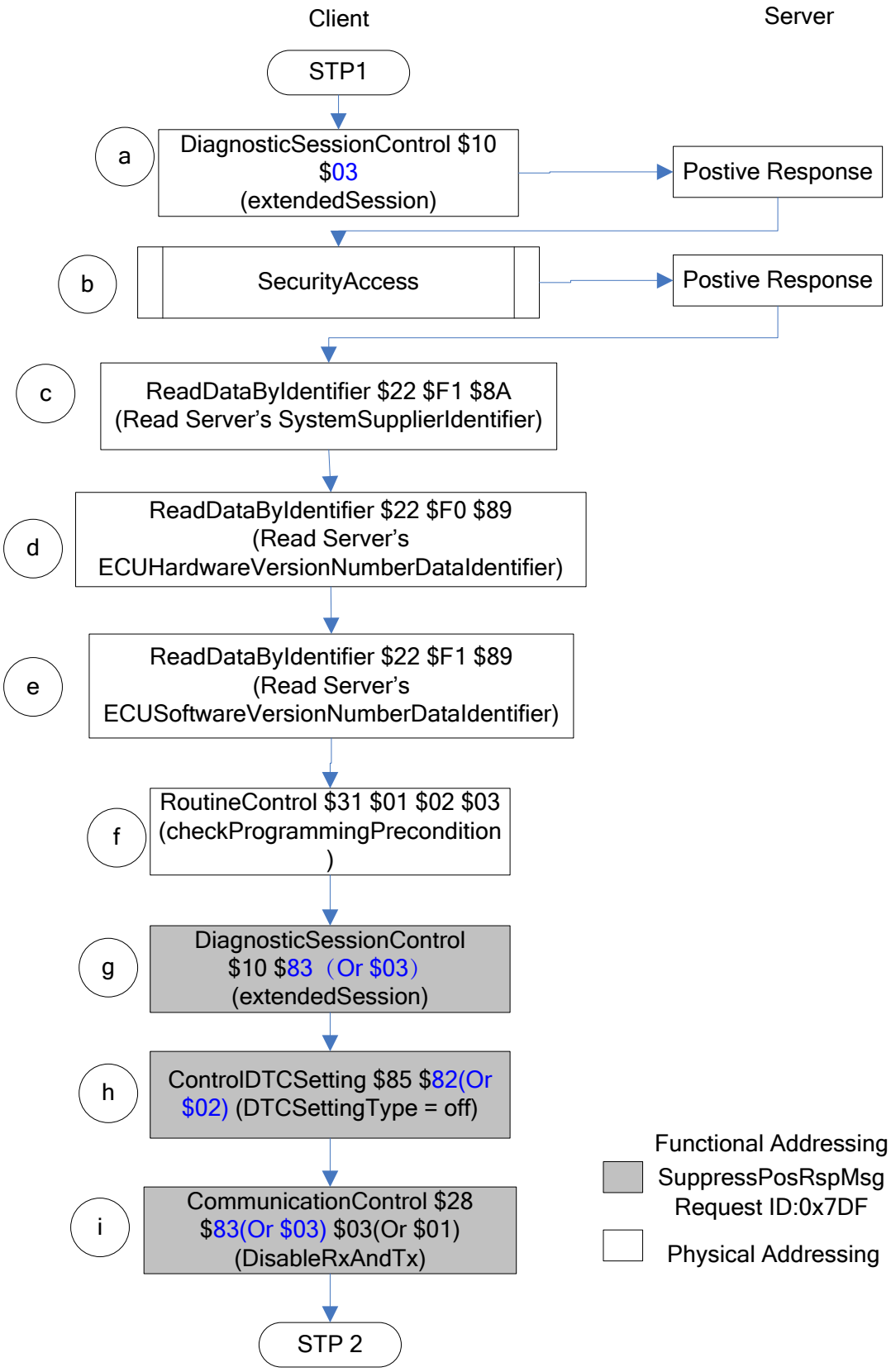
Client
Server
STP1
a
DiagnosticSessionControl $10 $03 (extendedSession)
Postive Response
b
SecurityAccess
Postive Response
c
ReadDataByIdentifier $22 $F1 $8A (Read Server's SystemSupplierIdentifier)
d
ReadDataByIdentifier $22 $F0 $89 (Read Server's ECUHardwareVersionNumberDataIdentifier)
e
ReadDataByIdentifier $22 $F1 $89 (Read Server's ECUSoftwareVersionNumberDataIdentifier)
f
RoutineControl $31 $01 $02 $03 (checkProgrammingPrecondition)
g
DiagnosticSessionControl $10 $83（Or $03） (extendedSession)
h
ControlDTCSetting $85 $82(Or $02) (DTCSettingType = off)
i
CommunicationControl $28 $83(Or $03) $03(Or $01) (DisableRxAndTx)
STP 2
Functional Addressing
SuppressPosRspMsg
Request ID:0x7DF
Physical Addressing

图4 预编程步骤

5.2.1.1 （a）DiagnosticSessionControl(10 hex) service

该服务使被编程 ECU 进入扩展诊断会话模式。

5.2.1.2 （b）SecurityAccess (27 hex) service

具体内容说明参见 5.2.4.1 节

5.2.1.3 ReadDataByIdentifier (22 hex) service

读取数据服务为必选服务，通过标识符读数据，可用来确定 ECU 软硬件等信息。通过标识符读数据服务，每次请求仅支持一个标识符。

本节要求的标识符要求既可以通过应用程序读取（如默认和扩展模式中），又可以在编程模式中读取。而且在 bootloader 引导程序中读取的数据与应用程序中的应保持一致。注意，无论是否有有效的应用程序，本节中要求的所有数据标识符都应该可以被正确的回应（应用程序无效时，部分与其相关的 $22 服务 Positive Response 返回相应字节长度的“0xFF”）。

零部件号（硬件件号）、硬件版本在无应用程序的情况亦可从引导程序中正确读取。

零部件号（硬件件号）F187 hex 和软件件号 F188 hex 虽未在重编程流程中使用。但仍是强制要求的标识符。其长度及格式定义详见长安诊断问卷表《ECU Questionnaires Diagnostic UDSonCAN.xls》，由长安电器部门提供。

（c）dataIdentifier F18A hex (System Supplier Identifier)

该编号为定义的唯一识别 ECU 系统供应商序列号，这个标识符是强制的。读供应商序列号请求报文格式和肯定响应报文格式如下表所示。

表3 ReadDataByIdentifier request message flow

Message direction:	Tester → ECU	
Message Type:	Request	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	ReadDataByIdentifier request SID	22
#2	dataIdentifier [byte 1] (MSB)	F1
#3	dataIdentifier [byte 2] (LSB)	8A

表4 ReadDataByIdentifier positive response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	ReadDataByIdentifier response SID	62
#2	dataIdentifier [byte 1] (MSB)	F1
#3	dataIdentifier [byte 2] (LSB)	8A
#4	dataRecord [data_1]	ASCII
#5	dataRecord [data_2]	ASCII
:		:
:		:
#10	dataRecord [data_7]	ASCII

ECU 系统供应商序列号为 7 字节，数据格式为 ASCII 码，首字符从字节 dataRecord[data_1]开始，所有未使用的字节填充 0x00。

（d）dataIdentifierF089 hex

该编号为长安定义的唯一 ECU 硬件版本号，这个标识符是强制的。

表5 ReadDataByIdentifier request message flow

Message direction:	Tester → ECU	
Message Type:	Request	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	ReadDataByIdentifier request SID	22
#2	dataIdentifier [byte 1] (MSB)	F0
#3	dataIdentifier [byte 2] (LSB)	89

表6 ReadDataByIdentifier positive response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	ReadDataByIdentifier response SID	62
#2	dataIdentifier [byte 1] (MSB)	F0
#3	dataIdentifier [byte 2] (LSB)	89
#4	"H"	ASCII
#5	"W"	ASCII
#6	":"	ASCII
#7	"A\B\C"	ASCII
#8	Dot	ASCII
#9	″0–9″	ASCII
#10	Dot	ASCII
#11	″0–9″	ASCII

ECU 硬件版本号为 8 字节，数据格式为 ASCII 码，首字符从字节 dataRecord[data_1]开始，所有字符填充格式见上表。

（e）**dataIdentifierF189 hex**

该编号为长安定义的唯一 ECU 软件版本号，这个标识符是强制的。

表7 ReadDataByIdentifier request message flow

Message direction:	Tester → ECU	
Message Type:	Request	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	ReadDataByIdentifier request SID	22
#2	dataIdentifier [byte 1] (MSB)	F1
#3	dataIdentifier [byte 2] (LSB)	89

表8 ReadDataByIdentifier positive response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	ReadDataByIdentifier response SID	62
#2	dataIdentifier [byte 1] (MSB)	F1
#3	dataIdentifier [byte 2] (LSB)	89
#4	"S"	ASCII
#5	"W"	ASCII
#6	":"	ASCII
#7	"A\B\C"	ASCII
#8	Dot	ASCII
#9	″0–9″	ASCII
#10	Dot	ASCII
#11	″0–9″	ASCII

ECU 软件版本号为 8 字节，数据格式为 ASCII 码，首字符从字节 dataRecord[data_1] 开始，所有字符填充格式见上表。

5.2.1.4 （f）RoutineControl (31 hex) service

需要编程预条件检查来确保系统处在重编程过程可以实施的安全状态。对 ECU 的重编程预检查条件（31 01 02 03），需要由零部件供应商、零部件负责部门共同确定，并填写入长安统一模板要求的零部件诊断定义文件中。

5.2.1.5 （g）DiagnosticSessionControl(10 hex) service

在 ECU 可以被重编程之前，网络中 DTC 及常规通信的设置须被关闭，这要求所有连接到网络中的 ECU 须以非默认诊断会话模式启动。因此，此请求使用功能寻址方式。此请求服务发送后，外部重编程工具必须以 4s 的周期发送功能寻址诊断在线报文，以使网络中所有 ECU 保持在扩展诊断会话模式。

5.2.1.6 （h）控制DTC设置 0x85 0x82(Or 0x02)

通过 DTC 设置类型设为“关闭”的控制 DTC 设置服务请求，用来在重编程过程中禁止 ECU 检测和存储 DTC。此请求使用功能寻址来关闭网路中所有 ECU 的默认代码设置。

5.2.1.7 （i）通信控制 0x28 0x83(Or 0x03) 0x03(Or 0x01)

通过通信控制服务请求，所有网络中连接的 ECU 禁止非诊断报文的发送。在非诊断报文发送禁止时，ECU 接收并处理诊断请求报文。通过禁止非诊断报文的发送，总线的全部带宽均用来下载，所以下载不会被非诊断报文干扰。

5.2.2 服务器编程步骤

服务器编程步骤用来编程一个或多个逻辑块。本步骤所有服务的请求使用物理寻址，所以有可能平行的编程多个节点。在不终止服务器编程步骤的情况下可以编程多个逻辑块。

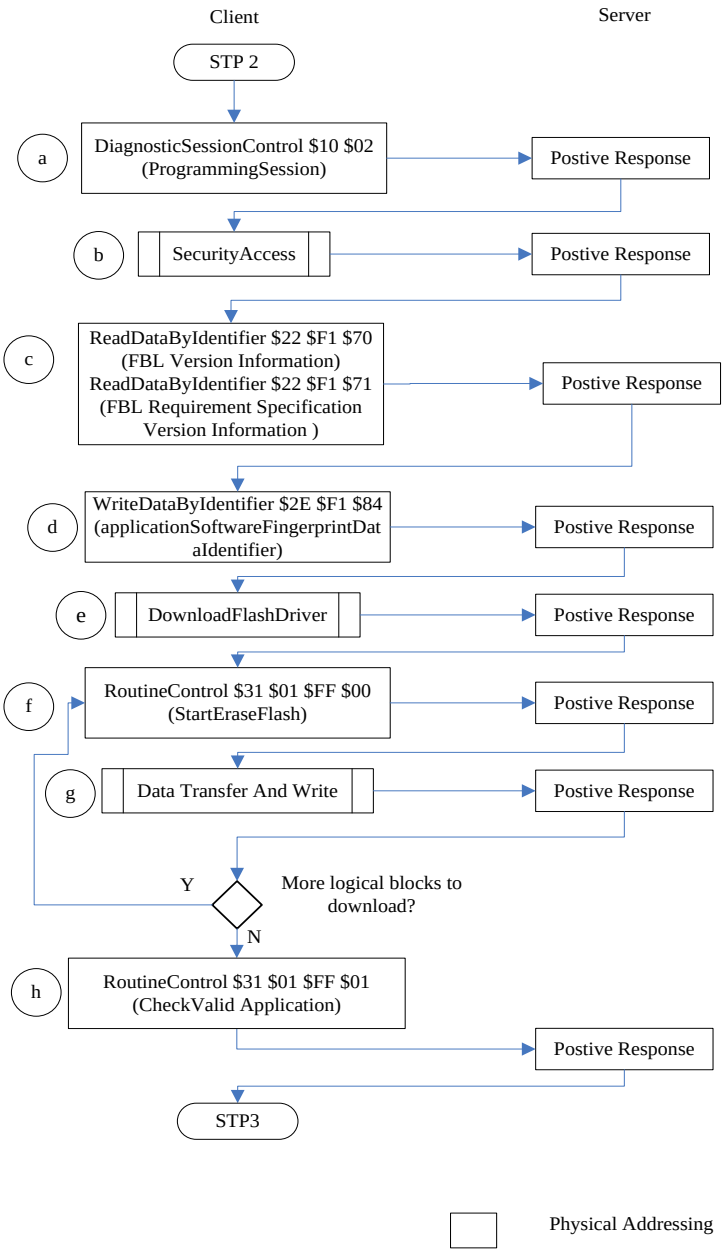

图 5 编程步骤

5.2.2.1 （a）诊断会话控制 0x10 0x02

ECU 中下载过程由子功能参数设为编程会话的诊断会话控制服务请求启动。当 ECU 正在执行操作的应用软件时，此服务请求导致了从应用软件至引导软件的转换。当重编程会话模式激活时，ECU 支持重编程特殊服务请求。

5.2.2.2 （b）SecurityAccess (27 hex) service

具体内容说明参见 5. 2. 4. 1

5.2.2.3 （c）ReadDataByIdentifier (22 hex) service

读取 Bootloader 软件版本号，供应商应在程序中给出相应的 FBL 软件版本号。请求格式和响应报文如下表。

表9 ReadDataByIdentifier request message flow

Message direction:	Tester → ECU	
Message Type:	Request	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	ReadDataByIdentifier request SID	22
#2	dataIdentifier [byte 1] (MSB)	F1
#3	dataIdentifier [byte 2] (LSB)	70

表10 ReadDataByIdentifier positive response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	ReadDataByIdentifier response SID	62
#2	dataIdentifier [byte 1] (MSB)	F1
#3	dataIdentifier [byte 2] (LSB)	70
#4	"V"	ASCII
#5	"0-9"	ASCII
#6	Dot	ASCII
#7	"0-9"	ASCII

读取 FBL 规范版本号，供应商应在程序中给出相应的 FBL 规范的版本号。请求格式和响应报文如下表。

表11 ReadDataByIdentifier request message flow

Message direction:	Tester → ECU	
Message Type:	Request	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	ReadDataByIdentifier request SID	22
#2	dataIdentifier [byte 1] (MSB)	F1
#3	dataIdentifier [byte 2] (LSB)	71

表12　ReadDataByIdentifier positive response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	Description (all values are in hexadecimal)	Byte Value (Hex)
#1	ReadDataByIdentifier response SID	62
#2	dataIdentifier [byte 1] (MSB)	F1
#3	dataIdentifier [byte 2] (LSB)	71
#4	"v"	ASCII
#5	"0-9"	ASCII
#6	Dot	ASCII
#7	"0-9"	ASCII

5.2.2.4 （d）WriteDataByIdentifier (2E hex) service

优先于其它的 Flash 访问，诊断仪的指纹须在 ECU 的非易失性存储器中存储，指纹信息为 7 个字节的内容，前 3 个字节存储刷写程序的时间信息（年月日），后四个字节存储下载设备的设备编号。指纹通过写入数据服务请求转移至 ECU 中。

指纹是重编程过程的强制信息结构，标识了一个特定的下载或下载尝试。指纹是能够追踪一个下载尝试或完成的下载的必需条件。写指纹信息的请求报文格式和肯定响应格式见下表，否定响应按 UDS 规范回复。

表13　WriteDataByIdentifier request message flow

Message direction:	Tester → ECU	
Message Type:	Request	
Data byte	Description (all values are in hexadecimal)	Byte Value
#1	WriteDataByIdentifier Request SID	2E
#2	dataIdentifier [byte 1] (MSB)	F1
#3	dataIdentifier [byte 2] (LSB)	84
#4	year：“0-99”	Dec
#5	month：“1-12”	Dec
#6	day：“1-31”	Dec
#7	device number byte1	ASCII
#8	device number byte2	ASCII
#9	device number byte3	ASCII
#10	device number byte4	ASCII

表14　WriteDataByIdentifier PositiveResponse message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	Description (all values are in hexadecimal)	Byte Value (Hex)
#1	WriteDataByIdentifier response SID	6E
#2	dataIdentifier [byte 1] (MSB)	F1
#3	dataIdentifier [byte 2] (LSB)	84

5.2.2.5 （e）Flash驱动下载（DownloadFlashDriver）

下载 Flash 驱动至 ECU 指定的 RAM 缓存是强制的步骤，如图 8 所示。

驱动下载过程由请求下载，数据传输和请求退出传输 3 个服务请求组成。在驱动下载完成后，需要通过例程控制服务请求使能 Flash 编程。此请求的例程标识符为 0x0202，例程请求格式见 5.2.4.2 节。该例程用来确保在 Flash 驱动没有正确下载到 ECU 的情况，该 ECU 是不能被重编程的。

5.2.2.6 （f）例程控制–启动擦除内存 0x31 0x01 0xFF 0x00

参数为 0x01 0xFF 0x00 的例程控制服务请求可以擦除被请求的逻辑块。在擦除 Flash 驱动的例程被调用前，请求擦除的逻辑块的有效状态位必须设为无效。这样可以在 Flash 过程没有成功结束时阻止意外执行应用程序。另外，此前写入数据服务请求接收到的指纹数据须在逻辑块擦除前存储到非易失性存储器。

擦除内存服务的请求报文格式、肯定响应格式和否定相应格式见下表。

表15　　Memory Erase request message flow

Message direction:	Tester → ECU	
Message Type:	Request	
Data byte	Description (all values are in hexadecimal)	Byte Value (Hex)
#1	Memory Erase Request SID	31
#2	RoutineControlType	01
#3	dataIdentifier [byte 1] (MSB)	FF
#4	dataIdentifier [byte 2] (LSB)	00
#5	memoryAddress[Byte1](MSB)	00-FF
#6	memoryAddress[Byte2]	00-FF
#7	memoryAddress[Byte3]	00-FF
#8	memoryAddress[Byte4] (LSB)	00-FF
#9	memorySize[Byte1] (MSB)	00-FF
#10	memorySize[Byte2]	00-FF
#11	memorySize[Byte3]	00-FF
#12	memorySize[Byte4] (LSB)	00-FF

表16　　Memory Erase positive response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	Description (all values are in hexadecimal)	Byte Value (Hex)
#1	Memory Erase Positive Response SID	71
#2	RoutineControlType	01
#3	dataIdentifier [byte 1] (MSB)	FF
#4	dataIdentifier [byte 2] (LSB)	00

表17　　Memory Erase negative response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	Description (all values are in hexadecimal)	Byte Value (Hex)
#1	Memory Erase Negative Response SID	7F
#2	RoutineControl request SID	31
#3	Negative Response Codes	13、22、33、72、92、93.....

5.2.2.7 （g）软件下载过程（Data Transfer And Write）

5.2.2.7.1 RequestDownload (34 hex) service

请求下载服务用来发起一个从诊断仪到 ECU 的数据传输。请求下载服务的报文格式和肯定响应格式见下表，否定响应按 UDS 规范回复。

表18　　RequestDownload request message flow

Message direction:	Tester → ECU	
Message Type:	Request	
Data byte	Description (all values are in hexadecimal)	Byte Value (Hex)
#1	RequestDownload request SID	34
#2	dataFormatIdentifier	00(非压缩刷新)/10（压缩刷新）
#3	addressAndLengthFormatIdentifier	44
#4	memoryAddress[Byte1] (MSB)	00-FF
#5	memoryAddress[Byte2]	00-FF
#6	memoryAddress[Byte3]	00-FF
#7	memoryAddress[Byte4] (LSB)	00-FF
#8	memorySize[Byte1] (MSB)	00-FF
#9	memorySize[Byte2]	00-FF
#10	memorySize[Byte3]	00-FF
#11	memorySize[Byte4] (LSB)	00-FF

表19　RequestDownload positive response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	RequestDownload response SID	74
#2	lengthFormatIdentifier	20
#3	maxNumberOfBlockLength [byte #1]	00-FF
#4	maxNumberOfBlockLength [byte #2]	00-FF

5.2.2.7.2 **TransferData (36 hex) service**

数据传输服务用来从诊断仪到 ECU 传输数据下载。

如果发起一个请求下载，待传输的数据包含在数据传输服务的请求报文的参数 TransferRequestParameter 中。TransferRequestParameter 数据量 N，根据芯片不同，供应商可自定义，但不能大于 0x34 服务应答报文中的 maxNumberOfBlockLength。

当接收到一个请求下载（0x34）服务，BlockSequenceCounter 应该初始化为 0x01。这意味着在请求下载（0x34）服务后，第一个数据传输（0x36）服务请求的 BlockSequenceCounter 从 1 开始计数，该值依次递增加 1，当达到 0xFF 后，下一个 TransferData 的 BlockSequenceCounter 值以 0x00 开始，依此循环计数。

数据传输服务请求格式，肯定响应格式见下表，否定响应按 UDS 标准回复。

表20　- TransferData request message flow (download)

Message direction:	Tester → ECU(download)	
Message Type:	Request	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	TransferData request SID	36
#2	blockSequenceCounter	00-FF
#3	transferRequestParameterRecord[transferRequestParameter#1] =data#1	00-FF
#4	transferRequestParameterRecord[transferRequestParameter#2] =data#2	00-FF
#5	transferRequestParameterRecord[transferRequestParameter#3] =data#3	00-FF
:	:	:
#n	transferRequestParameterRecord[transferRequestParameter#n] =data#N	00-FF

表21　TransferData positive response message flow (download)

Message direction:	ECU → Tester(download)	
Message Type:	Response	
Data byte	Description (all values are in hexadecimal)	Byte Value (Hex)
#1	TransferData response SID	76
#2	BlockSequenceCounter	00-FF

5.2.2.7.3 RequestTransferExit (37 hex) service

该服务用来退出诊断仪和 ECU 之间的数据传输，其返回参数包含一个两字节校验和（CRC16-CITT），诊断仪需要该校验和来支持数据传输。ECU 将请求下载服务规定的所有数据字节（即每请求下载序列中所有 BlockSquenceCounte 后跟的数据内容）写入内存后，再将其返读出来并计算校验和，具体算法见附录。当校验和错误时，则诊断仪停止刷写。

一个逻辑块的所有数据可通过服务序列请求下载、数传输和请求退出传输 3 个步骤下载至 ECU。一个逻辑块可由多个软件段组成，段的下载由请求下载服务请求启动，请求下载服务通知引导程序内存地址及段长度。一个软件段下载完成后，开启另一个软件段的请求下载，一个逻辑块包含的所有段的数据通过一个或多个的数据传输请求传输。

退出数据传输服务请求格式，肯定响应格式见下表，否定响应按 UDS 标准回复。

表22 RequestTransferExit request message flow

Message direction:	Tester —> ECU	
Message Type:	Request	
Data byte	Description (all values are in hexadecimal)	Byte Value (Hex)
#1	RequestTransferExit response SID	37

表23 RequestTransferExit positive response message flow

Message direction:	ECU —>Tester	
Message Type:	Response	
Data byte	Description (all values are in hexadecimal)	Byte Value (Hex)
#1	ResponseTransferExit request SID	77
#2	Block checksum value byte1(MSB)	00-FF
#3	Block checksum value byte2(LSB)	00-FF

5.2.2.8 （h）例程控制–启动应用程序有效性检查

RoutineIdentifier FF01 hex (Check Valid Application)用来保证重编程的逻辑块的完整和有效性。该例程控制请求在最后一个逻辑块下载完后发送给 ECU，ECU 在检查完应用程序有效标记后，再回复响应报文。逻辑块的完整检查是必选项。例程请求格式，肯定响应格式，否定响应见下表。

表24　　Check Valid Application request message flow

Message direction:	Tester → ECU	
Message Type:	Request	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	Request Check Valid ApplicationSID	31
#2	RoutineControlType	01
#3	dataIdentifier [byte 1] (MSB)	FF
#4	dataIdentifier [byte 2] (LSB)	01

表25　　Check Valid Application positive response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	Positive Response Check Valid Application SID	71
#2	RoutineControlType	01
#3	dataIdentifier [byte 1] (MSB)	FF
#4	dataIdentifier [byte 2] (LSB)	01

表26　　Check Valid Application negative response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	Negative Response check Valid Application SID	7F
#2	RoutineControl request SID	31
#3	应用有效性校验错误	01
	UDS定义了的否定响码按UDS规范	11、12、13、35、……

5.2.3 后编程步骤

后编程步骤的执行用来在重编程 ECU 的编程步骤完成之后结束编程活动。

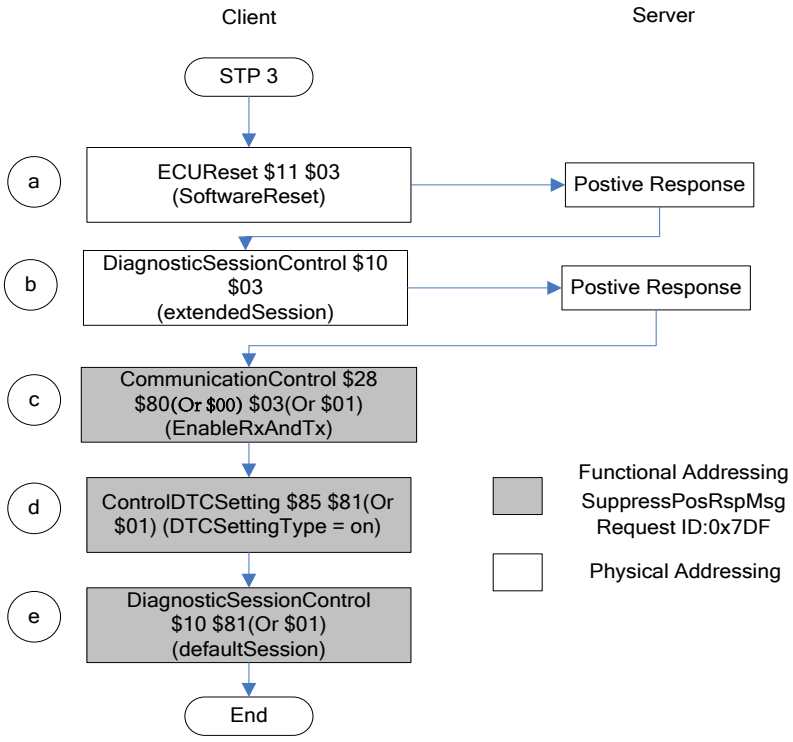

图 6 后编程步骤

5.2.3.1 （a）电控单元复位 0x11 0x03

Flash 重编程过程由 ECU 复位服务请求终止，可使 ECU 返回到正常 ECU 操作。

为了避免代码的意外激活，Flash 驱动代码必须从 RAM 存储器完全擦除 。因为这些代码可能导致擦除或程序操作的终止。

5.2.3.2 （b）诊断模式切换 0x10 0x03

该服务使被编程 ECU 进入扩展诊断会话模式。

5.2.3.3 （c）通信控制0x28 0x80(或 0x00) 0x03(或 0x01)

功能报文请求服务通信控制-允许非诊断通信须被发送至网络连接的所有 ECU 用来返回常规通信。

5.2.3.4 （d）控制DTC设置 0x85 0x81(或 0x01)

在下载执行完毕之后，所有网络中连接的 ECU 须允许 DTC 设置，此设置通过子功能参数 DTC 设置类型设为开启的功能性寻址请求服务控制 DTC 设置实现。

5.2.3.5 （e）诊断会话控制0x10 0x81(或 0x01)

最终，所有网络中的 ECU 都返回至默认会话模式。

5.2.4 子过程说明

5.2.4.1 SecurityAccess (27 hex) service

安全访问服务在对关于安全、排放需要限制访问的 ECU 内使用。安全访问服务用

来解锁 ECU，以便能够进行数据上传和下载。安全访问服务将使用安全访问算法，SecurityAccessLevel 0 算法从长安智能化研究院整车电子电器架构室获取。算法使用四字节种子和密钥，安全访问流程如图 7 所示。

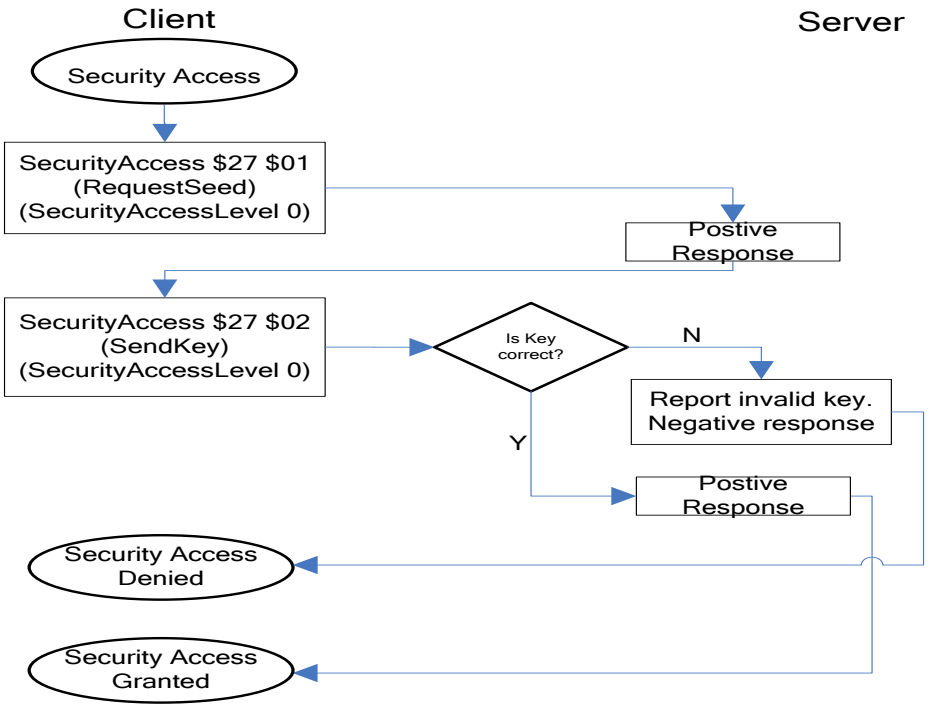

图 7 安全访问子过程

请求格式，肯定响应格式见下表，否定响应按 UDS 规范回复。

表27 SecurityAccess request message flow

Message direction:	Tester → ECU	
Message Type:	Request	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	SecurityAccess request SID	27
#2	securityAccessType = requestSeed, suppressPosRspMsgIndicationBit = FALSE (bit7 = 0)	01

表28 SecurityAccess positive response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	SecurityAccess response SID	67
#2	securityAccessType = requestSeed	01
#3	securitySeed [byte 1] = seed #1	00-FF
#4	securitySeed [byte 2] = seed #2	00-FF
#5	securitySeed [byte 3] = seed #3	00-FF
#6	securitySeed [byte 4] = seed #4	00-FF

ECU 需在 Flash 重编程前进行安全访问。为了解锁 ECU，诊断仪必须先向 ECU 请求种子，然后诊断仪和 ECU 都计算密钥，诊断仪将密匙发送给 ECU。ECU 将它的密钥值与从诊断仪中接收的密钥进行比较，如果二值相等，ECU 准许诊断仪的安全访问。

下述的安全访问流程用来解锁 ECU，并且运行安全访问。一旦接收到有效的安全种子请求，安全种子应该在 ECU 内保持有效或激活，直到下列之一的条件发生：

1、请求不同的安全级别会导致之前已解锁的安全级别重新上锁，请求相同的安全级别不会导致 ECU 重新上锁。

2、ECU 切换到另外一个诊断模式（如，S3server 超时），或 ECU 切换到相同的诊断模式（如，通过诊断模式控制服务）。

当通过安全访问服务 ECU 解锁之前，下述服务需要被禁止：

1）例程控制

2）写入数据

3）FLASH 擦写

4）请求下载：当 ECU 锁定时，传输数据和请求传输退出服务相应的被禁止。在没有激活请求下载或请求上传服务时，如果接收到传输数据或请求退出传输服务请求，ECU 应该响应请求序列错误。

对所请求的安全级别，如果 ECU 已经处于解锁状态，则安全种子各字节应该为 0x00。

5.2.4.2 Flash驱动下载（DownloadFlashDriver）

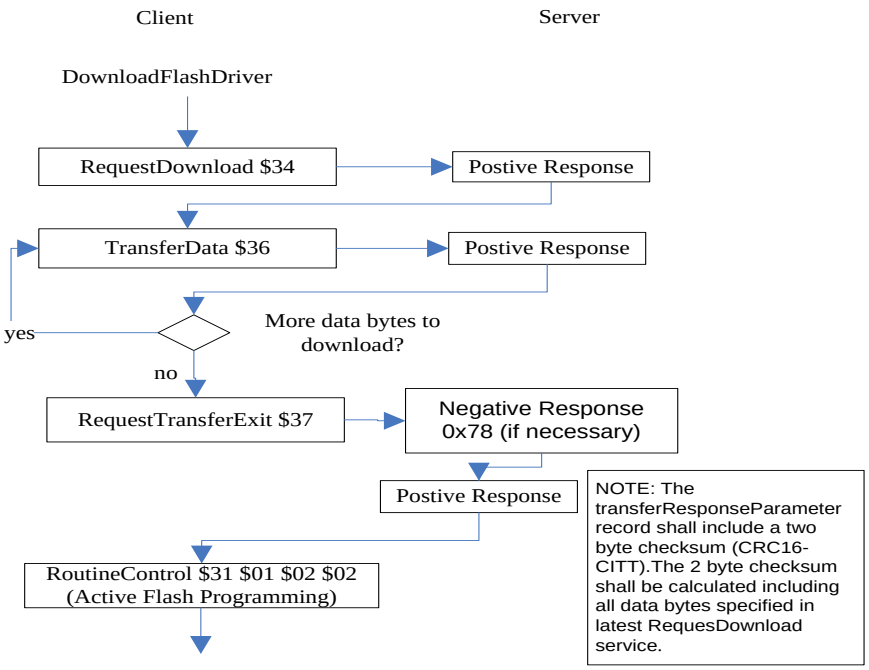

图 8 Flash 驱动下载子过程

Flash 驱动下载子流程说明：

1、FlashDriver 完整下载到 ECU 的 RAM 后，需启动例程检查 0x0202 来使能 Flash 编程,若 FlashDriver 未正确下载或未执行 FlashDriver 下载过程，该例程请求反回错误代码 0x22。若没收到 Routine 0x0202 请求或该请求的回复是否定响应，则拒绝后续的 Erase Flash。0x0202 例程请求格式，肯定响应格式，否定相应格式见下表。

表29　Active Flash Programming process request message flow

Message direction:	Tester → ECU	
Message Type:	Request	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	Request StartCheckMemory SID	31
#2	RoutineControlType	01
#3	routineIdentifier [byte 1] (MSB)	02
#4	routineIdentifier [byte 2] (LSB)	02
#5	Memory start address [byte 1] (MSB)	00-FF
#6	Memory start address [byte 2]	00-FF
#7	Memory start address [byte 3]	00-FF
#8	Memory start address [byte 4] (LSB)	00-FF

表30　Active Flash Programming process positive response message flow

Message direction:	ECU → Tester	
Message Type:	Response	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	Response StartCheckMemory SID	71
#2	RoutineControlType	01
#3	dataIdentifier [byte 1] (MSB)	02
#4	dataIdentifier [byte 2] (LSB)	02

表31　Active Flash Programming process negative response message flow

Message Type:	Response	
Data byte	**Description (all values are in hexadecimal)**	**Byte Value (Hex)**
#1	Active Flash Programming Negative Response SID	7F
#2	RoutineControl request SID	31
#3	Negative Response Codes	01:FLASH DRIVER NOT ACTIVE 13、22、33.....

2、下载数据的校验采用 CRC16-CITT 算法，参考校验算法见附录。

5.2.4.3 软件下载过程（Data Transfer And Write）

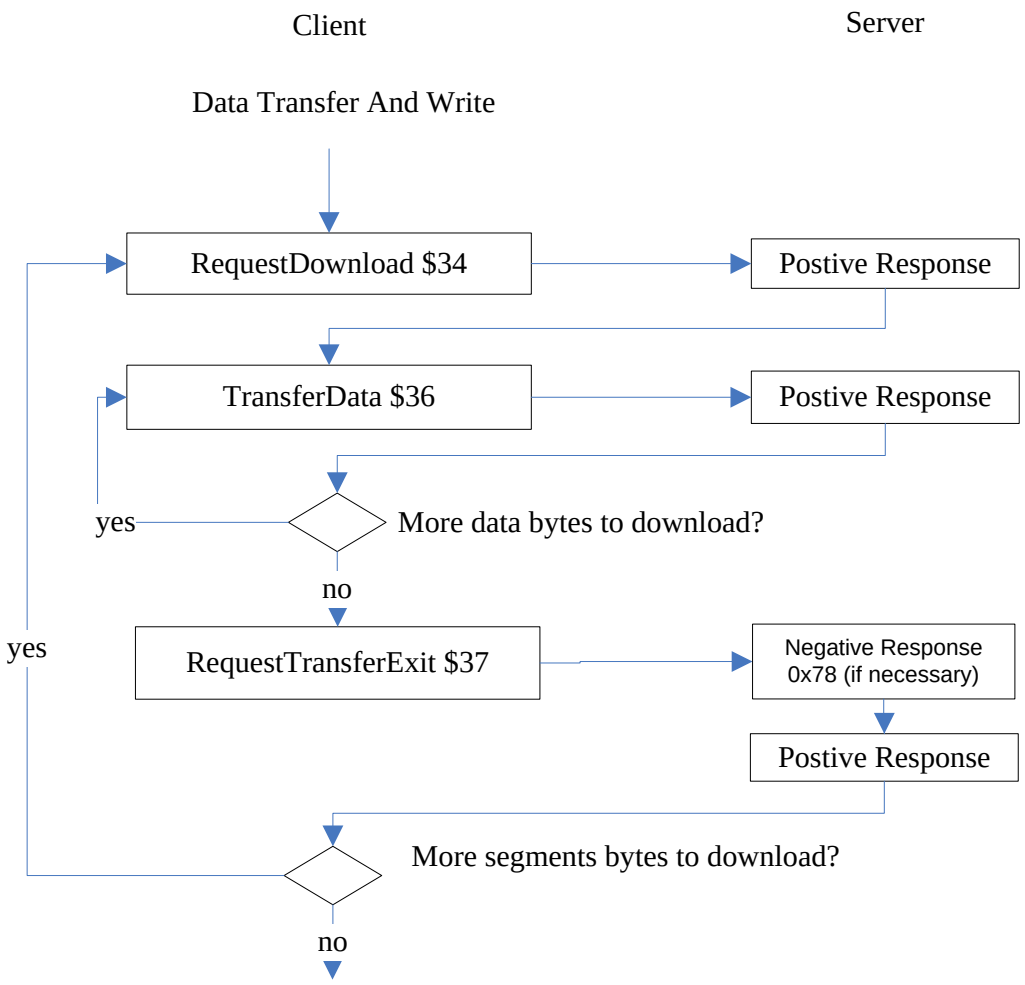

图 9 数据传输和写入子过程

软件下载子流程说明：

1、下载数据的校验采用 CRC16-CITT 算法，参考校验算法见附录。

5.3 诊断服务概览

5.3.1 引导软件诊断服务

本章提供了引导软件需要的 UDS 诊断服务子集的概览。其后的表格列出了需要服务的标识符和支持的子功能参数。此外，表格列出了某服务请求的寻址方式：物理寻址，功能寻址或二者兼有。最后三列指明了该服务是否在默认、编程或扩展会话模式下支持。

表32　　引导软件诊断服务

诊断服务描述 子功能参数	十六进制	物理请求	功能请求	诊断模式支持		
				0x01	0x02	0x03
诊断会话控制	**10**					
默认会话模式	10 01	✓	✓	✓	✓	✓
编程会话模式	10 02	✓	-	-	✓	✓
扩展会话模式	10 03	✓	✓	✓	-	✓
ECU复位	**11**					
软件复位	11 03	✓	✓	✓	✓	✓
读取数据	**22**					
读取系统供应商序列号	22 F1 8A	✓	-	✓	✓	✓
读取ECU硬件版本号	22 F0 89	✓	-	✓	✓	✓
读取ECU软件版本号	22 F1 89	✓	-	✓	✓	✓
读取ECU零部件号（硬件件号）	22 F1 87	✓	-	✓	✓	✓
读取ECU软件件号	22 F1 88	✓	-	✓	✓	✓
读取Bootloader软件版本号	22 F1 70	✓	-	✓	✓	✓
读取FBL规范版本号	22 F1 71	✓	-	✓	✓	✓
安全访问	**27**					
请求种子（为编程）	27 01	✓	-	-	✓	✓
发送密钥（为编程）	27 02	✓	-	-	✓	✓
通信控制	**28**					
禁止读写	28 03 03	✓	✓	-	-	✓
允许读写	28 00 03	✓	✓	-	-	✓
写入数据	**2E**					
引导软件指纹数据标识符	2E F1 84	✓	-	-	✓	-
例程控制	**31**					
检查编程预条件	31 01 02 03	✓	-	-	-	✓
启动擦除内存Start Erase Memory	31 01 FF 00	✓	-	-	✓	-
启动检查例程Start Check Routine	31 01 02 02	✓	-	-	✓	-
程序设计依赖检查	31 01 FF 01	✓	-	-	✓	-
请求下载	**34**					
无子功能参数	-	✓	-	-	✓	-
数据传输	**36**					
块序列计数	36 xx	✓	-	-	✓	-
请求退出传输	**37**					
无子功能参数	-	✓	-	-	✓	-
设备在线	**3E**					
零子功能	0	✓	✓	✓	✓	✓
控制DTC设置	**85**					
DTC设置类型 = on	85 01	✓	✓	-	-	✓
DTC置类型 = off	85 02	✓	✓	-	-	✓

5.3.2 应用软件诊断服务

应用软件中需要一些诊断服务来支持 Flash 重编程。这些服务在下表中列出。

表33　　应用软件诊断服务

诊断服务描述 子功能参数	十六进制	物理请求	功能请求	诊断模式	
				0x01	0x03
诊断会话控制	**10**				
默认会话模式	10 01	✓	✓	✓	✓
编程会话模式	10 02	✓	-	-	✓
扩展会话模式	10 03	✓	✓	✓	✓
ECU复位	**11**				
软件复位	11 03	✓	✓	✓	✓
读取数据	**22**				
读取系统供应商序列号	22 F1 8A	✓	-	✓	✓
读取ECU硬件版本号	22 F0 89	✓	-	✓	✓
读取ECU软件版本号	22 F1 89	✓	-	✓	✓
读取ECU零部件号（硬件件号）	22 F1 87	✓	-	✓	✓
读取ECU软件件号	22 F1 88	✓	-	✓	✓
读取Bootloader软件版本号	22 F1 70	✓	-	✓	✓
读取FBL规范版本号	22 F1 71	✓	-	✓	✓
安全访问	**27**				
请求种子（为编程）	27 01	✓	-	-	✓
发送密钥（为编程）	27 02	✓	-	-	✓
通信控制	**28**				
禁止读写	28 03 03	✓	✓	-	✓
允许读写	28 00 03	✓	✓	-	✓
写入数据	**2E**				
数据标识符	2E xx yy	✓	-	-	✓
例程控制	**31**				
检查编程预条件	31 01 02 03	✓	-	-	✓
设备在线	**3E**				
零子功能	0	✓	✓	✓	✓
控制DTC设置	**85**				
DTC设置类型 = on	85 01	✓	✓	-	✓
DTC置类型 = off	85 02	✓	✓	-	✓

附录

1、CRC16-CITT C-code example 1 (fast)

This example uses a look-up table with pre-calculated CRCs for fast calculation.

\+ fast calculation
– large code size

```
/* function declarations */

unsigned int CalcCRC(unsigned int size,unsigned char *data);


/* test data array */

unsigned char data[256] =

{    0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,

     0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x19,0x1a,0x1b,0x1c,0x1d,0x1e,0x1f,

     0x20,0x21,0x22,0x23,0x24,0x25,0x26,0x27,0x28,0x29,0x2a,0x2b,0x2c,0x2d,0x2e,0x2f,

     0x30,0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x39,0x3a,0x3b,0x3c,0x3d,0x3e,0x3f,

     0x40,0x41,0x42,0x43,0x44,0x45,0x46,0x47,0x48,0x49,0x4a,0x4b,0x4c,0x4d,0x4e,0x4f,

     0x50,0x51,0x52,0x53,0x54,0x55,0x56,0x57,0x58,0x59,0x5a,0x5b,0x5c,0x5d,0x5e,0x5f,

     0x60,0x61,0x62,0x63,0x64,0x65,0x66,0x67,0x68,0x69,0x6a,0x6b,0x6c,0x6d,0x6e,0x6f,

     0x70,0x71,0x72,0x73,0x74,0x75,0x76,0x77,0x78,0x79,0x7a,0x7b,0x7c,0x7d,0x7e,0x7f,

     0x80,0x81,0x82,0x83,0x84,0x85,0x86,0x87,0x88,0x89,0x8a,0x8b,0x8c,0x8d,0x8e,0x8f,

     0x90,0x91,0x92,0x93,0x94,0x95,0x96,0x97,0x98,0x99,0x9a,0x9b,0x9c,0x9d,0x9e,0x9f,

     0xa0,0xa1,0xa2,0xa3,0xa4,0xa5,0xa6,0xa7,0xa8,0xa9,0xaa,0xab,0xac,0xad,0xae,0xaf,

     0xb0,0xb1,0xb2,0xb3,0xb4,0xb5,0xb6,0xb7,0xb8,0xb9,0xba,0xbb,0xbc,0xbd,0xbe,0xbf,

     0xc0,0xc1,0xc2,0xc3,0xc4,0xc5,0xc6,0xc7,0xc8,0xc9,0xca,0xcb,0xcc,0xcd,0xce,0xcf,

     0xd0,0xd1,0xd2,0xd3,0xd4,0xd5,0xd6,0xd7,0xd8,0xd9,0xda,0xdb,0xdc,0xdd,0xde,0xdf,

     0xe0,0xe1,0xe2,0xe3,0xe4,0xe5,0xe6,0xe7,0xe8,0xe9,0xea,0xeb,0xec,0xed,0xee,0xef,

     0xf0,0xf1,0xf2,0xf3,0xf4,0xf5,0xf6,0xf7,0xf8,0xf9,0xfa,0xfb,0xfc,0xfd,0xfe,0xff

};
```

```
unsigned int crctab[256] =

{

    0x0000, 0x1021, 0x2042, 0x3063, 0x4084, 0x50A5, 0x60C6, 0x70E7,

    0x8108, 0x9129, 0xA14A, 0xB16B, 0xC18C, 0xD1AD, 0xE1CE, 0xF1EF,

    0x1231, 0x0210, 0x3273, 0x2252, 0x52B5, 0x4294, 0x72F7, 0x62D6,

    0x9339, 0x8318, 0xB37B, 0xA35A, 0xD3BD, 0xC39C, 0xF3FF, 0xE3DE,

    0x2462, 0x3443, 0x0420, 0x1401, 0x64E6, 0x74C7, 0x44A4, 0x5485,

    0xA56A, 0xB54B, 0x8528, 0x9509, 0xE5EE, 0xF5CF, 0xC5AC, 0xD58D,

    0x3653, 0x2672, 0x1611, 0x0630, 0x76D7, 0x66F6, 0x5695, 0x46B4,

    0xB75B, 0xA77A, 0x9719, 0x8738, 0xF7DF, 0xE7FE, 0xD79D, 0xC7BC,

    0x48C4, 0x58E5, 0x6886, 0x78A7, 0x0840, 0x1861, 0x2802, 0x3823,

    0xC9CC, 0xD9ED, 0xE98E, 0xF9AF, 0x8948, 0x9969, 0xA90A, 0xB92B,

    0x5AF5, 0x4AD4, 0x7AB7, 0x6A96, 0x1A71, 0x0A50, 0x3A33, 0x2A12,

    0xDBFD, 0xCBDC, 0xFBBF, 0xEB9E, 0x9B79, 0x8B58, 0xBB3B, 0xAB1A,

    0x6CA6, 0x7C87, 0x4CE4, 0x5CC5, 0x2C22, 0x3C03, 0x0C60, 0x1C41,

    0xEDAE, 0xFD8F, 0xCDEC, 0xDDCD, 0xAD2A, 0xBD0B, 0x8D68, 0x9D49,

    0x7E97, 0x6EB6, 0x5ED5, 0x4EF4, 0x3E13, 0x2E32, 0x1E51, 0x0E70,

    0xFF9F, 0xEFBE, 0xDFDD, 0xCFFC, 0xBF1B, 0xAF3A, 0x9F59, 0x8F78,

    0x9188, 0x81A9, 0xB1CA, 0xA1EB, 0xD10C, 0xC12D, 0xF14E, 0xE16F,

    0x1080, 0x00A1, 0x30C2, 0x20E3, 0x5004, 0x4025, 0x7046, 0x6067,

    0x83B9, 0x9398, 0xA3FB, 0xB3DA, 0xC33D, 0xD31C, 0xE37F, 0xF35E,

    0x02B1, 0x1290, 0x22F3, 0x32D2, 0x4235, 0x5214, 0x6277, 0x7256,

    0xB5EA, 0xA5CB, 0x95A8, 0x8589, 0xF56E, 0xE54F, 0xD52C, 0xC50D,

    0x34E2, 0x24C3, 0x14A0, 0x0481, 0x7466, 0x6447, 0x5424, 0x4405,

    0xA7DB, 0xB7FA, 0x8799, 0x97B8, 0xE75F, 0xF77E, 0xC71D, 0xD73C,

    0x26D3, 0x36F2, 0x0691, 0x16B0, 0x6657, 0x7676, 0x4615, 0x5634,

    0xD94C, 0xC96D, 0xF90E, 0xE92F, 0x99C8, 0x89E9, 0xB98A, 0xA9AB,

    0x5844, 0x4865, 0x7806, 0x6827, 0x18C0, 0x08E1, 0x3882, 0x28A3,

    0xCB7D, 0xDB5C, 0xEB3F, 0xFB1E, 0x8BF9, 0x  9BD8, 0xABBB, 0xBB9A,

    0x4A75, 0x5A54, 0x6A37, 0x7A16, 0x0AF1, 0x1AD0, 0x2AB3, 0x3A92,

    0xFD2E, 0xED0F, 0xDD6C, 0xCD4D, 0xBDAA, 0xAD8B, 0x9DE8, 0x8DC9,

    0x7C26, 0x6C07, 0x5C64, 0x4C45, 0x3CA2, 0x2C83, 0x1CE0, 0x0CC1,
```

```
    0xEF1F, 0xFF3E, 0xCF5D, 0xDF7C, 0xAF9B, 0xBFBA, 0x8FD9, 0x9FF8,
    0x6E17, 0x7E36, 0x4E55, 0x5E74, 0x2E93, 0x3EB2, 0x0ED1, 0x1EF0
};

/* driver */
void main(void)
{
    unsigned int crc;

    crc=CalcCRC(256,data);

    while(1); /* the result of the above calculation shall be: crc=0x3FBD */
}

/* Calculate CRC */
unsigned int CalcCRC(unsigned int size, unsigned char *data)
{
    unsigned int crc=0xffff; /* initial value */
    int tmp;
    int i;

    for(i=0;i<size;i++) {
      tmp=(crc>>8)^data[i];
      crc=(crc<<8)^crctab[tmp];
    }
    return crc;
}
```

2、CRC16-CITT C-code example 2 (slow)

This example is not using any look-up table, only shift and XOR operations

\+ small code size
– slow calculation

```
#define  MASK 0x1021   /* CRC16-CITT mask */


/* function declarations */

unsigned int CalcCRC(unsigned int size,unsigned char *data);


/* test data array */

unsigned char data[256] =

{    0x00,0x01,0x02,0x03,0x04,0x05,0x06,0x07,0x08,0x09,0x0a,0x0b,0x0c,0x0d,0x0e,0x0f,

     0x10,0x11,0x12,0x13,0x14,0x15,0x16,0x17,0x18,0x19,0x1a,0x1b,0x1c,0x1d,0x1e,0x1f,

     0x20,0x21,0x22,0x23,0x24,0x25,0x26,0x27,0x28,0x29,0x2a,0x2b,0x2c,0x2d,0x2e,0x2f,

     0x30,0x31,0x32,0x33,0x34,0x35,0x36,0x37,0x38,0x39,0x3a,0x3b,0x3c,0x3d,0x3e,0x3f,

     0x40,0x41,0x42,0x43,0x44,0x45,0x46,0x47,0x48,0x49,0x4a,0x4b,0x4c,0x4d,0x4e,0x4f,

     0x50,0x51,0x52,0x53,0x54,0x55,0x56,0x57,0x58,0x59,0x5a,0x5b,0x5c,0x5d,0x5e,0x5f,

     0x60,0x61,0x62,0x63,0x64,0x65,0x66,0x67,0x68,0x69,0x6a,0x6b,0x6c,0x6d,0x6e,0x6f,

     0x70,0x71,0x72,0x73,0x74,0x75,0x76,0x77,0x78,0x79,0x7a,0x7b,0x7c,0x7d,0x7e,0x7f,

     0x80,0x81,0x82,0x83,0x84,0x85,0x86,0x87,0x88,0x89,0x8a,0x8b,0x8c,0x8d,0x8e,0x8f,

     0x90,0x91,0x92,0x93,0x94,0x95,0x96,0x97,0x98,0x99,0x9a,0x9b,0x9c,0x9d,0x9e,0x9f,

     0xa0,0xa1,0xa2,0xa3,0xa4,0xa5,0xa6,0xa7,0xa8,0xa9,0xaa,0xab,0xac,0xad,0xae,0xaf,

     0xb0,0xb1,0xb2,0xb3,0xb4,0xb5,0xb6,0xb7,0xb8,0xb9,0xba,0xbb,0xbc,0xbd,0xbe,0xbf,

     0xc0,0xc1,0xc2,0xc3,0xc4,0xc5,0xc6,0xc7,0xc8,0xc9,0xca,0xcb,0xcc,0xcd,0xce,0xcf,

     0xd0,0xd1,0xd2,0xd3,0xd4,0xd5,0xd6,0xd7,0xd8,0xd9,0xda,0xdb,0xdc,0xdd,0xde,0xdf,

     0xe0,0xe1,0xe2,0xe3,0xe4,0xe5,0xe6,0xe7,0xe8,0xe9,0xea,0xeb,0xec,0xed,0xee,0xef,

     0xf0,0xf1,0xf2,0xf3,0xf4,0xf5,0xf6,0xf7,0xf8,0xf9,0xfa,0xfb,0xfc,0xfd,0xfe,0xff

};
```

```
/* driver */
void main(void)
{
    unsigned int crc;

    crc=CalcCRC(256,data);

    while(1); /* the result of the above calculation shall be: crc=0x3FBD */
}



/* Calculate CRC */
unsigned int CalcCRC(unsigned int size, unsigned char *data)
{
    unsigned int crc=0xffff; /* initial value */
    int tmp;
    int i,j;

    for(i=0;i<size;i++){
      tmp=data[i]<<8;
      for(j=0;j<8;j++){
        if((crc ^ tmp) & 0x8000) crc=(crc<<1)^MASK;
        else crc<<=1;
        tmp<<=1;
      }
    }
    return crc;
}
```