# 1 CCP协议基础知识

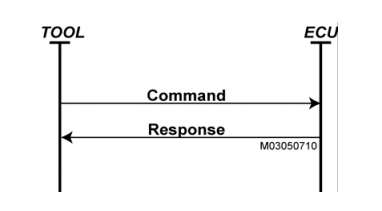
## 1.1 CCP

CCP(CAN Calibration Protocol)，中文名为CAN标定协议,是基于CAN总线的一种应用层协议。

CCP可以读取RAM、PORTS、ROM、FLASH，写RAM、PORTS、FLASH。

CCP协议采用主从通信方式，其流程可以简化为选择从机->选择控制方式->接收数据->断开连接。

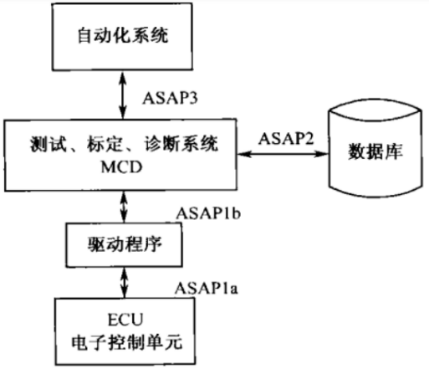
CCP报文有两种：一种是CRO(Command Receive Object),即命令接收对象；一种是DTO(Data Transmission Object),即数据传输对象。结合CCP的主从通信模式，可以说CRO为指令报文，DTO为应答报文。



CCP工作模式，Polling（查询模式）：一问一答；DAQ（DATA Acquistion模式）托管模式。

## 1.2 ASAM

ASMA标准：



ASAM1a：定义ECU与校准/测量系统间的逻辑与物理接口（协议）。

ASAM1b：定义应用程序与校准/测量系统间的接口（软件接口、MCS驱动）。

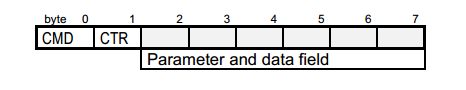
ASAM2：定义数据库，包括标定数据库（A2L数据）和诊断数据库（Open Diagnostic data eXchange）,ASAM2中记录了ECU中各项参数详细信息，如标定参数和测量变量ECU中存储地址、存储结构、数据类型和转换公式。CANape中，每个标定参数和测量数据都会有一个变量名，如发动机温度、冷却水温度。当CANape需要访问某个变量时，找到ASAP2描述文件中变量名,找到该变量在ECU中的存储地址、数据长度等信息。为了对ASAP2文件进行维护和修改，CANape集成了一个ASAP2 Editor的数据库编辑器,生成和修改ASAP2控制器描述文件。所有信息都能对话框形式进行设置和修改。该数据库编辑器还能工作在独立模式下，以生成一个A2L格式的控制器描述文件。 当ECU底层程序修改后,一些标定参数和测量数据内存址可能发生变动，CANape支持linker map文件自动更新ASAP2文件里的信息。Map文件是ECU底层程序编译时由编译器生成一种映射文件，Map文件可以自动更新ASAP2文件，保证了测量与标定变量时地址的一致性。

ASAM3：操作和控制标定系统的远程接口。

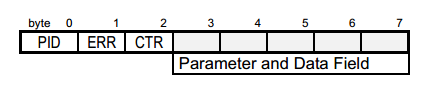
# 2 CCP报文帧

## 2.1 报文帧基本格式

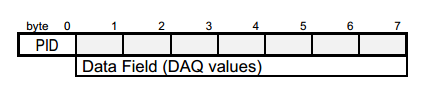
CRO报文：CCP报文帧格式为CMD + CTR + DATA，即Command、Counter、Data，其中Command是一些列的命令标号，为人为规定，Counter计数用，Counter根据发送消息的先后顺序进行赋值，例如第一条发送的CRO其CTR为01，相对应的，从机对其进行应答的相应报文的CTR值也应该为01，第二条的CTR值为02，以此类推。CCP协议共规定了28条命令。



DTO报文：分为三种类型报文，分别是CRM-DTO（命令返回消息）,EVENT-DTO（事件消息）,DAQ-DTO。



CRM-DTO和EVENT-DTO共用上一种报文结构，第一个字节PID（data packet ID）用于判断报文属于那一类型，PID = 0xFF为CRM，PID = 0xFE为EVENT，其它的都是DAQ，第二个字节ERR（error code），第三个字节为代码指令计数器CTR，其它字节为指令代码相关的返回参数和数据



DAQ-DTO（数据采集）报文除了第一个字节为PID外，其余字节都是返回数据。

## 2.2 报文帧格式解析

|  |  |
| --- | --- |
| PID | 定 义 |
| 0xFE | DTO是CRM-DTO |
| 0xFF | DTO是EVENT-DTO |
| 0~0xFD | DTO是DAQ-DTO |

Error:对于CRM-DTO来说，Error反映了CRO所请求命令的执行情况，例如返回了0x00，表明CRO命令正确执行命令。对于事件消息EVENT-DTO，Error的数值表示ECU内部发生了那种错误

## 2.3 报文帧含义

CRM-DTO发生在Polling模式下，是针对主设备发送的CRO消息的应答。

EVENT-DTO事件消息发生在ECU内部错误时由ECU主动发送，这类DTO不需要事先收到主设备的CRO，一旦ECU发生内部错误，会主动向主设备发生这一消息，报告内部情况。

# 3 DAQ下的通信

DAQ模式下，设备从机可以脱离主设备的命令控制，自动按照一定周期向主设备上传数据，此模式中，需要主设备先发送一条请求DAQ命令，从设备接收到之后会按照命令中参数自行配置并上传数据，然后按一定周期自主向主设备上传数据，这种方法工作效率高，但实现较为复杂，上传数据量偏大，会大量占用ECU空间。

DAQ通信需要依靠DAQ列表，ODT列表，DAQ-ODT来实现

ODT（Obiect Descriptor Table）：元素（变量）列表，用于组织数据采集

## 3.1 DAQ列表

按照不同的上传周期，ECU内部需要多个DAQ列表。例如有三个信号，上传周期分别为10ms、20ms、30ms，那么就需要3个DAQ列表，而同一上传周期的数据/变量（OTD）都被归类在同一个DAQ列表内，这些ODT被称作ODT列表，一个DAQ列表内部最多可以包含7个ODT列表。

## 3.2 ODT列表

七个ODT组成一个ODT列表，每一个ODT的最大元素数目为7，可以存放7个单字节数据变量的信息，其内容包括数据变量的存放地址，数据长度及其偏移地址。ODT列表需要通过DAQ-DTO方式才能向主设备发送，由于每一个ODT都有唯一的绝对编号，该编号即为DAQ-DTO的PID，排在最前面的ODT相对编号为0。绝对编号是对于所有ODT来说的，每一个ODT的绝对编号都不相同。相对编号是相对于当前DAQ列表而言，不同DAQ列表的ODT，其相对编号可能会相同。正是因为单帧有八字节数据，而ODT编号需要占用一字节，因此在DAQ-DTO中只剩下7字节来用于ODT的传输，而又因为CCP无法进行跨帧的数据/命令传输，所以一个ODT最大只能存储7字节数据。又因为PID的长度为8位，因此ODT的总数不能超过254个（0~0xFD）（通过修改PID长度可以增加ODT总数，但会减少单个ODT的长度）。

## 3.3 DAQ通信设置

在使用DAQ通信前，主设备需要对DAQ列表及ODT列表进行配置：

获取ECU内部DAQ列表数量和ODT数目，DAQ列表数量由上传的数据周期决定，每个不同周期对应一个DAQ列表。ODT列表数则由每个周期下上传的数据个数及长度决定。

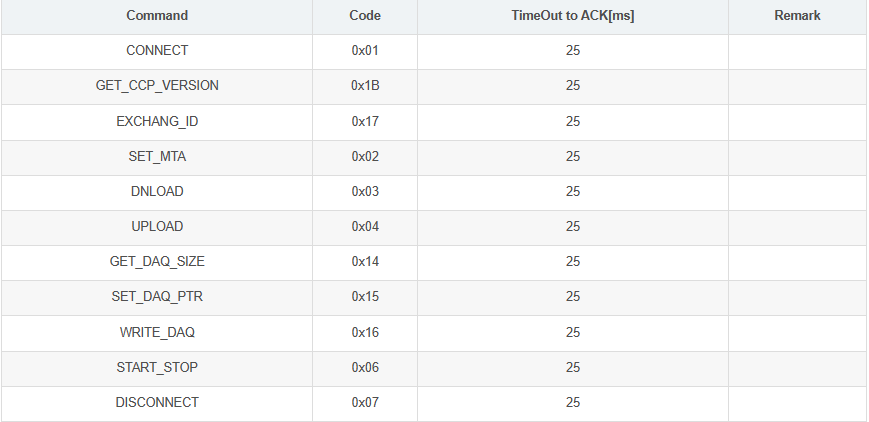
向相应的ODT填入数字，及数据的存储地址和数据长度，随后根据类型和上传周期被写在不同DAQ的不同ODT中。

上传周期由事件通道（Event Channel）和预分频值(Prescaler)决定.事件周期可以理解为ECU每多少ms触发一次，若为10，则说明ECU每10ms触发一次该事件。预分频值可以理解为：以事件通道的时间为基准进行分频，我们以此来复用事件通道。若预分频值设定为2，则将某一事件以20ms的周期进行触发，但使用的仍然是同一个DAQ列表。

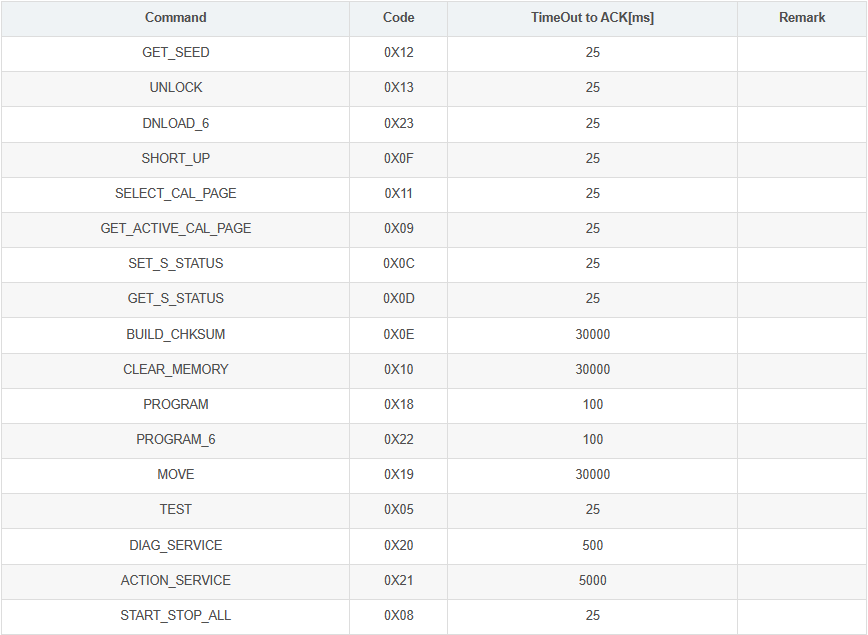
启动DAQ，进行数据采集。

# 4 CCP命令

## 4.1 基本指令11个



## 4.2 可选指令17个



## 4.3 命令返回码和错误类别



错误等级：

C0：警告

C1：伪错误

C2：可修复错误

C3：不可修复

# 5 CCP命令详解

## 5.1 基本指令

### 5.1.1 连接（CONNECT 0x01）

此命令用于连接ECU，若在同一网络上使用此命令连接一个新的ECU，则将会断开与原来ECU的连接，如果连接已连接的ECU，则会返回一个确认信息。CCP虽然支持Intel或Motorola的数据格式，但是在CONNECT中仅支持Intel格式（LSB）。

CONNECT的CRO结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x01 |
| 1 | 字节 | 命令符号 = CTR |
| 2 | 字节 | ECU地址（Intel格式，低字节在前） |
| 3~7 | 字节 | 无效 |

针对CONNECT的命令反馈DTO结构如下:

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID = 0xFF |
| 1 | 字节 | 命令返回代码 = ERR（一般是0） |
| 2 | 字节 | 命令符号 = CTR（与发送的CRO的CTR一致） |
| 3~7 | 字节 | 无效 |

### 5.1.2 获取CCP协议版本（GET\_CCP\_VERSION 0x1B）

该命令用于统一主、从设备所使用的CCP协议版本，该命令应在EXCHANGE\_ID命令之前执行，GET\_CCP\_VERSION命令的CRO数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x1B |
| 1 | 字节 | 命令序号 = CTR |
| 2 | 字节 | 协议主版本号（期望值） |
| 3 | 字节 | 协议副版本号（期望值） |
| 4~7 | 字节 | 无效 |

针对其返回的DTO数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = EER |
| 2 | 字节 | 命令序号 = CTR |
| 3 | 字节 | 从设备所使用的协议主版本号 |
| 4 | 字节 | 从设备所使用的协议副版本号 |
| 5~7 | 字节 | 无效 |

注：若协议版本为2.1则主版本号为2，副版本号为1

### 5.1.3 交换站标识符（EXCHANGE\_ID 0x17）

在ASAP标准中，MCD（主设备）与ECU的通信需要ASAP2文件的支持，通过这个命令，自动化系统可以由DTO返回的ID标识符自动为ECU分配一ASAP文件

EXCHANGE\_ID的CRO结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x17 |
| 1 | 字节 | 命令符号 = CTR |
| 2 | 字节 | ECU地址（Intel格式，低字节在前） |
| 3~7 | 字节 | 无效 |

EXCHANGE\_ID的DTO结构如下

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x17 |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令符号 = CTR |
| 3 | 字节 | 从设备ID标识符的长度（字节数） |
| 4 | 字节 | 从设备ID数据类型（可选字节，视实际应用而定） |
| 5 | 字节 | 资源可用状态字节 |
| 6 | 字节 | 资源保护状态字节 |
| 7 | 字节 | 无效 |

为了防止可能发生的对ECU的误操作，会对某些功能通过密钥进行保护。EXCHANGE\_ID的第5、第6字节反映了这些特殊功能是否正处于保护状态。其字节结构定义如下：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| —— | X | PGM | X | X | X | X | DAQ | CAL |

CAL:标定 DAQ：DAQ通信模式 PGM：非易失性内存烧写 X：暂时无效

对于资源可用状态字节，某位置1则说明当前该项功能未受保护，可以启用。对于资源保护状态字节，某位置1则说明当前该项功能受密钥保护，需要权限才能对其进行访问。

### 5.1.4 设置MTA地址（SET\_MTA 0x02）

SET\_MTA命令用于设置一个初始地址（32位基地址+地址偏移），后续对内存的读取操作都由该起始地址开始。

CCP协议一共规定了两个MTA地址:MTA0与MTA1，分别针对不同的命令。DNLOAD、UPLOAD、DNLOAD\_6、SELECT\_CAL\_PAGE、CLEAR\_MEMORY、PROGRAM及PROGRAM\_6命令使用MTA0，MOVE命令使用MTA1。

SET\_MTA命令CRO的数据常结构如下:

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x02(SET\_MTA) |
| 1 | 字节 | 命令序号 = CTR |
| 2 | 字节 | MTA序号（0或1） |
| 3 | 字节 | 地址偏移 |
| 4~7 | 无符号长整型 | 地址 |

针对SET\_MTA命令返回DTO的数据场结构，如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID：0xFF |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3~7 | 字节 | 无效 |

例：主设备向从设备发送SET\_MTA命令，当前CTR为0x23，MTA序号为0，地址偏移为0x02，基地址为0x34002000则该SET\_MTA命令应为：

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| —— | 0x02 | 0x23 | 0x00 | 0x02 | 0x34 | 0x00 | 0x20 | 0x00 |

从设备返回DTO，包括确认代码ERR，CTR。

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| bit | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| —— | 0xFF | 0x00 | 0x23 | —— | —— | —— | —— | —— |

### 5.1.5 数据下载（DNLOAD 0x03）

DNLOAD指令负责将CRO中的数据下载到ECU中，起始地址为先前设定的MTA0，下载完毕后MTA0指针自增，自增的字节数为下载的字节数。DNLOAD命令的CRO数据场结构如下:

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x03 |
| 1 | 字节 | 命令符号 = CTR |
| 2 | 字节 | 下载数据大小 |
| 3~7 | 字节 | 下载数据（最多为5个字节） |

针对DNLOAD命令返回DTO的数据场结构：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID：0xFF |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3 | 字节 | MTA0偏移量（自增后） |
| 4~7 | 无符号长整型 | MTA0地址（自增后） |

### 5.1.6 数据上传（UPLOAD 0x04）

此命令请求设备以MTA0为起始地址，将命令中规定字节数的数据上传，随后MTA0指针自动增加相应的字节数。UPLOAD命令的数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x04 |
| 1 | 字节 | 命令符号 = CTR |
| 2 | 字节 | 请求上传的数据大小（字节数） |
| 3~7 | 字节 | 无效 |

其命令返回DTO的数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3~7 | 字节 | 所请求的数据 |

### 5.1.7 获取DAQ列表大小（GET\_DAQ\_SIZE 0x14）

此命令用于获取某特定DAQ列表的大小，即其中ODT列表的个数，并清空当前DAQ列表内的数据，为下次通讯做准备，如果GET\_DAQ\_SIZE命令中选定的DAQ列表不存在或者不可用，，从设备返回的ODT列表个数为0。该命令的CRO数据场如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x14 |
| 1 | 字节 | 命令符号 = CTR |
| 2 | 字节 | DAQ列表序号 |
| 3 | 字节 | 无效 |
| 4~7 | 无符号长整型 | 读DAQ列表，其所对应的DTO的CAN ID标识符 |

下面是该命令返回的DTO的数据场结构：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3 | 字节 | DAQ列表大小（ODT列表数） |
| 4 | 字节 | DAQ列表的第一个PID号 |
| 5~7 | 字节 | 无效 |

DAQ中的某高官ODT的PID号可以通过计算得来，其值为DAQ列表中的第一个PID号+ODT数目。

### 5.1.8 设置DAQ列表指针（SET\_DAQ\_PTR 0x15）

在进行DAQ通讯前，必须将DAQ列表进行配置，将数据写入到相应的DAQ列表里的ODT元素中。SET\_DAQ\_PTR用来为写入DAQ列表数据设置入口地址指针。其CRO数据场结构如下所示。

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x15 |
| 1 | 字节 | 命令符号 = CTR |
| 2 | 字节 | DAQ列表序号 |
| 3 | 字节 | ODT学号 |
| 4 | 字节 | 该ODT中的第几个元素 |
| 5~7 | 无符号长整型 | 无效 |

SET\_DAQ\_PTR命令返回DTO的数据场结构，如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3~7 | 字节 | 无效 |

### 5.1.9 写入DAQ列表（WRITE\_DAQ 0x16）

在DAQ通信前，需要对DAQ列表进行配置，将需要上传的数据先写到DAQ列表所在的ODT列表中，先前由SET\_DAQ\_PTR命令所定义的地址即为该命令的数据写入地址，在此命令中，一次写入的数据被成为一个DAQ元素，其字节可分为1字节、2字节、4字节。下面为该命令的CRO数据场结构：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x16 |
| 1 | 字节 | 命令序号 = CTR |
| 2 | 字节 | DAQ元素的大小 |
| 3 | 字节 | DAQ元素的地址偏移 |
| 4~7 | 无符号长整型 | DAQ元素的地址 |

针对WRITE\_DAQ命令返回DTO的数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3~7 | 字节 | 无效 |

### 5.1.10 开始/终止数据传输（START\_STOP 0x06）

该条命令用于DAQ通信模式，其作用是开始或终止某个DAQ列表的数据上传，其数据场结构如下表所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x06 |
| 1 | 字节 | 命令序号 = CTR |
| 2 | 字节 | 模式：开始/终止/准备数据传输 |
| 3 | 字节 | DAQ列表符号 |
| 4 | 字节 | 最后一个ODT符号 |
| 5 | 字节 | 事件通道号 |
| 6~7 | 字 | 传输速率预分频值 |

对于第3个字节：

0x00代表终止某DAQ列表的数据传输

0X02代表为开始同步传输做准备

0x01代表开始某DAQ列表的数据传输

命令中的事件通道号对应DAQ列表上传的周期，通过预分频值可延长上传周期，预分频值必须大于或等于1。

针对START\_STOP命令返回DTO的数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3~7 | 字节 | 无效 |

### 5.1.11 断开（DISCONNECT 0x07）

断开分为两种模式，一种离线模式，一种扯断断开与ECU的通信，彻底断开通信时，ECU将会被自动初始化。暂时断开即离线模式不会终止与当前ECU的DAQ通信，也不会影响先前的各项设置。在该命令中的ECU地址采用Intel格式，低位在前。其CRO数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x07 |
| 1 | 字节 | 命令序号 = CTR |
| 2 | 字节 | 命令参数：0x00：暂时断开；0x01：终止 |
| 3 | 字节 | 无效 |
| 4、5 | 字 | ECU地址（Intel格式，低位在前） |
| 6、7 | 字节 | 无效 |

针对该命令返回的DTO数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3~7 | 字节 | 无效 |

## 5.2 可选指令

### 5.2.1 申请密钥（GET\_SEED 0x12）

若某项功能处于保护状态，主设备需要通过GET\_SEED命令向ECU申请解开该功能的密钥。GET\_SEED每次只能解锁一项功能，如果请求开放的资源多于一项，需要多次并用GET\_SEED与UNLOCK命令。从设备通过DTO返回密钥数据，主设备通过密钥解算出某个功能的钥匙，使主设备有权限访问某项功能。

GET\_SEED的CRO数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x12 |
| 1 | 字节 | 命令符号 = CTR |
| 2 | 字节 | 请求设备开放的功能 |
| 3~7 | 字节 | 无效 |

针对GET\_SEED命令的DTO数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3 | 字节 | 请求功能当前受保护状态（真或者假） |
| 4~7 | 字节 | 密钥数据 |

### 5.2.2 解除保护（UNLOCK 0x13）

UNLOCK的CRO数据常结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x13 |
| 1 | 字节 | 命令符号 = CTR |
| 2~7 | 字节 | 钥匙 |

其返回的DTO数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3 | 字节 | 各项功能当前状态 |
| 4~7 | 字节 | 无效 |

### 5.2.3 数据下载6字节（DNLOAD\_6 0x23）

此命令与DNLOAD功能相同，但此命令可以一次性下载6字节数据，其CRO结构除第一个字节的命令代码不同外，DNLOAD中第三个字节也应为数据位。

### 5.2.4 数据短上传（SHORT\_UP 0x0F）

该命令功能等同于UPLOAD命令，区别在于UPLOAD命令上传数据的起始地址是MTA0，SHORT\_UP的起始地址由命令本身指定，从设备按该起始地址上传数据。上传后MTA0指针保持不变。SHORT\_UP命令的CRO数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x0F |
| 1 | 字节 | 命令符号 = CTR |
| 2 | 字节 | 请求上传的数据大小（字节数） |
| 3 | 字节 | 地址偏移量 |
| 4~7 | 无符号长整型 | 地址 |

其返回DTO的数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3~7 | 字节 | 所请求的数据 |

### 5.2.5 选择标定数据页（SELECT\_CAL\_PAGE 0x11）

该命令的功能取决于ECU的内部实现。执行该调命令后，先前设置的MTA0地址将会自动指向由该命令激活的标定页。其CRO数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x11 |
| 1 | 字节 | 命令序号 = CTR |
| 2~7 | 字节 | 无效 |

其DTO如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3~7 | 字节 | 无效 |

### 5.2.6 获取处于激活状态下的标定页（GET\_ACTIVE\_CAL\_PAGE 0x09）

该命令的功能是返回当前处于激活状态下的首地址：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x09 |
| 1 | 字节 | 命令序号 = CTR |
| 2~7 | 字 | 无效 |

针对GET\_ACTIVE\_CAL\_PAGE命令返回DTO的数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3 | 字节 | 地址偏移 |
| 4~7 | 无符号长整型 | 首地址 |

### 5.2.7 设置当前通信状态（SET\_S\_STATUS 0x0C）

从设备通过该条命令设置当前主从设备间的通信状态，其CRO数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x0C |
| 1 | 字节 | 命令序号 = CTR |
| 2 | 字节 | 状态字节 |
| 3~7 | 字节 | 无效 |

状态字节的定义表如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 名称 | 描述 |
| 0 | CAL | 标定数据初始化完成 |
| 1 | DAQ | DAQ列表初始化完成 |
| 2 | RESUME | 请求ECU断电时自动保存DAQ列表设置，在下次启动时再自动启动DAQ列表 |
| 6 | STORE | 请求ECU断电时保存标定数据 |
| 7 | RUN | 正处于运行阶段 |
| 3~5 | 保留 | 保留 |

该命令返回DTO的数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 指令返回代码 |
| 2 | 字节 | CTR |
| 3~7 | 字节 | 不关心 |

### 5.2.8 获取当前通信状态（GET\_S\_STATUS 0x0D）

主设备通过该条命令请求从设备提供当前通信状态，其CRO数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x0D |
| 1 | 字节 | 命令序号 = CTR |
| 2~7 | 字节 | 无效 |

针对GET\_S\_STATUS命令返回DTO的数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 名称 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3 | 字节 | 状态字节 |
| 4 | 字节 | 其它状态信息限定 |
| 5~7 | 字节 | 其他状态信息（可选） |

### 5.2.9 建立checksum表（BUILD\_CHKSUM 0x0E）

checksum是CCP提供的一个可选功能，为了提高效率，在每次对ECU进行标定前，CCP支持先对需要进行标定的内存区域进行checksum算法，如果checknum结果与下载数据不一致，表明内存中的数据与标定数据不同。

主设备用该命令请求从设备对指定内存区域（起始地址为MTA0，大小由命令中的block大小决定）进行checksum计算，并返回计算结果。checknum算法由制造商根据实际应用决定，不属于CCP协议范畴。该命令的CRO数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x0E |
| 1 | 字节 | 命令序号 = CTR |
| 2~5 | 无符号长整形 | block大小（以字节数标识） |
| 6、7 | 字节 | 无效 |

针对BUILD\_CHKSUM命令返回DTO的数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3 | 字节 | checksum数据长度 |
| 4~7 | 字节 | checksum数据（根据实际应用而定） |

### 5.2.10 清空内存（CLEAR\_MEMORYU 0x10）

此条命令可以用于在标定前清空FLASH EPROM中的数据，被清空区域的起始地址即是MTA0。CLEAR\_MEMOY命令的CRO数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x10 |
| 1 | 字节 | 命令序号 = CTR |
| 2~5 | 长整形 | 内存区域大小 |
| 3，6、7 | 字节 | 无效 |

针对CLEAR\_MEMORY命令返回DTO的数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 3~7 | 字节 | 无效 |

### 5.2.11 编程（PROGRAM 0x18）

不同于DNLOAD类命令，PROFRAM命令用以将一块大小确定的数据烧写到ECU的非易失性内存中，起始地址为MTA0，烧写结束后，MTA0按照烧写的字节数自增。其CRO数据域结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x18 |
| 1 | 字节 | 命令序号 = CTR |
| 2 | 字节 | 数据大小 |
| 3~7 | 字节 | 数据（最多5个字节） |

该命令返回的DTO数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3 | 字节 | MTA0偏移（自增后） |
| 4 | 无符号长整型 | MTA0地址（自增后） |

### 5.2.12 编程6字节（PROGRAM\_6 0x22）

将编程（PROGRAM）中用于标注数据大小的字节用于装订数据，其他不变。

### 5.2.13 内存转移（MOVE 0x19）

MOVE指令可将固定长度的一块内存区域中的内容由MTA0起始地址转移到MTA1起始地址处。MOVE命令的CRO数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 指令返回代码 |
| 2~5 | 长整型 | CTR |
| 6、7 | 字节 | 不关心 |

其DTO的数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 指令返回代码 |
| 2 | 字节 | CTR |
| 3~7 | 字节 | 不关心 |

### 5.2.14 连接状态测试（TEST 0x05）

测试连接使用该命令，命令消息中的ECU地址使用Intel格式，其CRO数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x05 |
| 1 | 字节 | 命令序号 = CTR |
| 2、3 | 字 | ECU地址（Intel格式，低字节在前） |
| 4~7 | 字节 | 无效 |

其DTO数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0xFF |
| 1 | 字节 | 命令序号 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3~7 | 字节 | 无效 |

### 5.2.15 诊断服务（DIAG\_SERVICE 0x20）

该条命令使从设备自动指向主设备请求的诊断服务，MTA0会自动重新定位，主设备从新的MTA0起始地址出获取诊断服务的反馈信息。其数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x20 |
| 1 | 字节 | 命令序号 = CTR |
| 2、3 | 字 | 诊断服务号 |
| 4~7 | 字节 | 附加参数（如果有） |

针对DIAG\_SERVICE返回的DTO数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3 | 字节 | 反馈信息长度（字节数） |
| 4 | 字节 | 反馈信息数据类型限定 |
| 5~7 | 字节 | 无效 |

### 5.2.16 操作服务（ACTION\_SERVICE 0x21）

同DIAG\_SERVICE命令，主设备通过该命令请求从设备自动指向某项操作，MTA0将自动重新定位，主设备可从新的MTA0起始地址处获取所请求操作服务的反馈信息。

其CRO结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x21 |
| 1 | 字节 | 命令序号 = CTR |
| 2、3 | 字 | 请求操作服务函数 |
| 4~7 | 字节 | 附加参数（如果有） |

其DTO结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字节 | 命令序号 = CTR |
| 3 | 字节 | 反馈信息长度（字节数） |
| 4 | 字节 | 反馈信息数据类型限定 |
| 5~7 | 字节 | 无效 |

### 5.2.17 开始/停止同步数据传输（START\_STOP\_ALL 0x08）

在先前的START\_STOP命令中，如果模式命令值为0x02，则对DAQ列表进行标识，为同步数据传输做准备。其CRO数据场结构如下：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | 命令代码 = 0x08 |
| 1 | 字节 | 命令序号 = CTR |
| 2 | 字 | 0x00停止数据传输；0x01开始数据传输 |
| 3~7 | 字节 | 无效 |

其返回DTO的数据场结构如下所示：

|  |  |  |
| --- | --- | --- |
| 位置 | 类型 | 描述 |
| 0 | 字节 | PID |
| 1 | 字节 | 命令返回代码 = ERR |
| 2 | 字 | 命令序号 = CTR |
| 3~7 | 字节 | 无效 |