# 上海方堰CAN Bootloader命令及参数说明

## 1，命令格式

所有数据采用扩展数据帧格式传输，CAN ID包含节点地址和命令，CAN DATA包含该命令对应的参数。

|  |  |  |
| --- | --- | --- |
| **保留** | **节点地址** | **命令** |
| ID28..ID20 | ID19..ID4 | ID3..ID0 |

我们程序中定义的命令如下：

|  |  |  |
| --- | --- | --- |
| **命令** | **具体值** | **说明** |
| EraseFlashID | 0x03 | 擦除APP程序存储扇区数据 |
| WriteBlockFlashID | 0x06 | 以多字节形式写数据 |
| BlockWriteInfoID | 0x05 | 设置多字节写数据相关参数（写起始地址，数据量） |
| OnlineCheckID | 0x01 | 监测当前节点是否在线，返回版本信息 |
| ExcuteAppID | 0x09 | 控制节点执行对应的固件 |

## 2，命令详解

### 2.1，EraseFlashID

擦除APP程序一块扇形存储区域的数据。擦除命令携带一个参数，存储擦除块的起始地址与终止地址。

|  |  |  |  |
| --- | --- | --- | --- |
| 起始地址flashStartAddr | | | |
| array[3] | array[2] | array[1] | array[0] |
| flashStarAddr[7..0] | flashStarAddr[15..8] | flashStarAddr[23..16] | flashStarAddr[31..24] |
| 终止地址flashEndAddr | | | |
| array[7] | array[6] | array[5] | array[4] |
| flashEndAddr[7..0] | flashEndAddr[15..8] | flashEndAddr[23..16] | flashEndAddr[31..24] |

### 2.2，WriteBlockFlashID

将数据写入到空闲的缓冲器中。

### 2.3，BlockWriteInfoID

设置写数据的起始地址和数据量大小。命令携带两个参数，分别代表写入数据的起始地址和数据量的大小。

|  |  |  |  |
| --- | --- | --- | --- |
| 数据起始地址flashStartAddr | | | |
| array[3] | array[2] | array[1] | array[0] |
| flashStartAddr[7..0] | flashStartAddr[15..8] | flashStartAddr[23..16] | flashStartAddr[31..24] |
| 数据量的大小len | | | |
| array[7] | array[6] | array[5] | array[4] |
| len[7..0] | len[15..8] | len[23..16] | len[31..24] |

### 2.4，OnlineCheckID

检查节点是否正常工作，同时获取该节点的固件版本信息，从节点接收到该命令后，需要返回获取固件信息的状态命令，若是成功，则需要携带固件版本参数。

|  |  |  |  |
| --- | --- | --- | --- |
| 固件版本Version | | | |
| 次版本号MINOR | | 主版本号MAJOR | |
| array[3] | array [2] | array [1] | array [0] |
| MINOR[7..0] | MINOR [15..8] | MAJOR[7..0] | MAJOR [15..8] |

### 2.5，ExcuteAppID

控制节点，执行对应的固件。命令发送之后，进行跳转，跳转到对应的固件。ExcuteAppID命令携带一个参数，参数代表固件对应的起始地址。

|  |  |  |  |
| --- | --- | --- | --- |
| 固件起始地址flashStartAddr | | | |
| array[3] | array[2] | array[1] | array[0] |
| flashStartAddr[7..0] | flashStartAddr[15..8] | flashStartAddr[23..16] | flashStartAddr[31..24] |

## 3，其他注意事项

1，由于CAN总线是广播传输，所以在实际使用的时候是可以进行多节点同时升级的，比如可以将0地址设置为广播地址，也就是当命令地址为0的时候，每个节点收到命令之后都应该执行该命令，但是由于同一时刻，CAN总线上不能传输多个节点的数据，所以这些从节点再执行0地址命令的时候可以不用返回状态，所以这样做在实际使用的时候，若某个节点的某个命令执行出了问题，主节点缺无法立即知道，因此就需要额外的方式来判断升级是否成功，比如可以通过升级完毕之后，通过获取每个节点的固件信息来判断。

2，为了保证烧写数据的可靠性，可以在发送数据的时候加上CRC校验，然后数据写入内部程序存储器中之后再做一次CRC校验，对比两次的CRC校验数据，可以保证写入芯片中的数据是完整无误的。

3，我们已经将这些命令封装在了USB2XXX的上位机API库中，若想快速实现CAN Bootloader，可以直接使用USB2XXX的上位机API库，直接调用对应的接口函数，或者直接使用我们提供的上位机软件即可。