# GuoZiCAN私有通讯协议

## 

## CAN包结构：

基于避开CANOpen以及最大化利用CAN单包字节数的考虑，将采用Extended ID(29 bit) 的CAN 包格式来进行通讯。通过在收发两端进行Can ID过滤的方式，将数据指令，读写方向以及数据偏移量等均整合在Extended ID中，从而将CAN包的8个Byte数据位充分利用，提高通讯效率。上位机下发数据和目标硬件上传数据均遵循该格式。

### Extended ID 具体bit位：

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  |  |  | Bit29 | Bit28 | Bit27 | Bit26 | Bit25 |
|  |  |  | Msg Cmd value | | | | |
| Bit24 | Bit23 | Bit22 | Bit21 | Bit20 | Bit19 | Bit18 | Bit17 |
| Index（高8位） | | | | | | | |
| Bit16 | Bit15 | Bit14 | Bit13 | Bit12 | Bit11 | Bit10 | Bit9 |
| Index（低8位） | | | | | | | |
| Bit8 | Bit7 | Bit6 | Bit5 | Bit4 | Bit3 | Bit2 | Bit1 |
| 成功/失败 | 预留 | 预留 | 预留 | NodeID(1~31) 0无效 | | | |

Bit 29~Bit25位：Message Command Value数据包指令

通过Message Command，来区分当前数据包的用途：启动升级，获取软件版本，获取固件类型等。(详见Appendix.1)

Bit24~9: Data Package Index

在硬件升级中，Data Package Index用于标明数据包相对于Flash初始地址的偏移量。在上传和下载中，均从1开始计数，每包数据Index增加1。

在寄存器读写时，Bit24~17 为Register Index，如果所读寄存器为Array，Bit16~9 为Sub-Index。

Bit8: Message Read / Write Successful / Failure

通过Bit8标志位，应答命令帧使用，用于表示命令是否执行成功

Successful : 1

Failure : 0

Bit4~1: Node-ID 节点号（1~31） 0为无效节点

Node-ID 为目标在CAN网络中的节点号，由目标硬件当前的拨码确定。（CAN Node ID）

### 数据位格式:

由于数据类型等通讯协议相关的变量均包含在Extended ID中，CAN包的8 Bytes 数据位可全用于实际的数据存储。参照canOpen协议，以Byte为单位采用小端储存的方式,每个Byte采用大端储存。

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Byte 8 | Byte 7 | Byte 6 | Byte 5 | Byte 4 | Byte 3 | Byte 2 | Byte 1 |
| Data（LS Byte） | Data | Data | Data | Data | Data | Data | Data（MS Byte） |

## Appendix. 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| CMD类型 | Command Value | 用途 | Index | Data(8Bytes) |
| Software Version | 0x01 | 读取目标硬件的当前软件版本 | 0 | 0 |
| SV Return | 0x11 | 软件版本回传 | 0 | VX.X.X.X (X为数字)(ASCII） |
| Hardware Type | 0x02 | 读取目标硬件的类型 | 0 | 0 |
| Hardware Type Return | 0x12 | 硬件类型回传 | 0 | PSD600(ASCII） |
| Initial Rom Update | 0x03 | 开始代码下载 | 0 | 升级区块标识（2 Bytes）+index数量（2 Bytes） |
| Preparation Ready | 0x13 | 确认可以开始下载 | 0 | 0 |
| Data Download | 0x04 | 代码下载 | 当前Data Index | 8 Bytes HEX数据 |
| Download Receive Ack | 0x14 | 下载接收应答 | 当前Data Index | 8 Bytes HEX数据 |
| Data Upload Ack | 0x08 | 代码上传命令 | 当前Data Index | 0 |
| Data Upload | 0x18 | 代码上传应答 | 当前Data Index | 8 Bytes HEX数据 |
| Update Successful | 0xA | 确定升级成功，并重启硬件 | 0 | SUCCESS (ASCII) |
| Update Successful Ack | 0x1A | 通过效验确认 | 0 | 0 |
| Update Abort | 0xF | 更新放弃 | 0 | FAILURE (ASCII） |
| Update Abort Ack | 0x1F | 更新放弃确认 | 0 | 0 |
| Read Register | 0xC | 对指定寄存器（Index）进行读取 | 指定寄存器的Index | 0 |
| Register Read Reply | 0x1C | 返回寄存器数值 | 寄存器数据 |
| Write Register | 0xD | 对指定寄存器（Index）进行写入 | 写入数据 |
| Register Write Reply | 0x1D | 确认寄存器写入，并返回写入值 | 寄存器数据 |