# XC-CLR CAN/LIN 转换器使用说明

成都讯川科技有限公司（Chengdu Xunchuan Technology Co., Ltd.） www.xcmotion.com
support@xcmotion.com

配套上位机：CAN-LIN-Config-Tool V1.2.0 适用设备固件：1.0.28 及以上
文档日期：2026-11-10

## 1. 产品简介

XC-CLR 跨在 CAN 和 LIN 两条总线之间。CAN 上来的报文，按规则翻译成 LIN 报文发出去；LIN 上来的报文，按规则翻译成 CAN 报文发出去。翻译规则不写在设备程序里，而是存成一张规则表放在设备内部，用配套上位机软件修改，改完掉电不丢。设备上电后自己按表干活，电脑不需要在场。

规则表里每条规则回答三个问题：

1. 什么样的报文算命中——源 ID、掩码、帧类型；
2. 命中之后发出什么——目标 ID、八个字节怎么填、长度、校验和类型；
3. 什么时候做——命中立即做、按周期重发、超时了怎么办。

一台设备同一时刻要么当 LIN 主机、要么当 LIN 从机，由机壳上的 LIN 主从拨码开关决定，软件不能切换。同一条规则在两种模式下的动作完全不同，这是配置时最容易出错的地方，第 4 章专门讲。

# 2. 快速上手

## 2.1 连接设备

1. 用串口线（485 转串口）把设备接到电脑；
2. 打开上位机，在顶部选择串口号和串口波特率，填上设备站号（出厂默认 1）；
3. 点**打开连接**。连接成功后按钮变为**断开连接**，顶部右侧会亮出设备的**主机/从机**状态徽标。

连接成功后软件会自动读取设备当前的拨码状态和收发数据上报通道，无需额外操作。

## 2.2 顶部一栏

顶部从左到右依次是：串口号、串口波特率、站号、打开连接，最右侧是主从机状态徽标。徽标绿色为**主机**、橙色为**从机**、灰色表示尚未读到（刚连接时可点一下"转换规则"页触发刷新）。设备拨码拨到哪一档，徽标就显示哪种模式；改拨码后需要给设备重新上电才生效。

## 2.3 配好参数再干活

站号、CAN 波特率、LIN 波特率都在**参数配置**页修改（见第 7 章），点**写入到设备**立即生效并掉电保存。首次使用建议流程：

1. 参数配置页确认站号和波特率；
2. 转换规则页配置转换规则并下载（见第 3、5、6 章）；
3. CAN 收发 / LIN 收发页验证总线数据（见第 9 章）。

# 3. 三种转换规则

规则按数据流向分三种类型。配置界面里每种类型一张卡片，下面逐一说明，每条规则都配了动画演示（网页版可播放，PDF 中为示意图）。

## 3.1 类型一：CAN→LIN（写）

CAN 总线上来一帧报文，设备把它翻译成一条完整的 LIN 报文（报文头 + 数据 + 校验和）发到 LIN 总线。这是最常用的一种，比如车身控制器用 CAN 发灯光指令，通过本设备转给 LIN 灯具。



**主机模式**

把整帧发上 LIN 总线：报文头（Break + Sync + PID）→ 8 字节数据 → 校验和 0x49。

**从机模式**

只更新 0x10 的应答缓存（单值覆盖，只留最新值），**不上 LIN 总线**。等真实主机来读时再返回。

要点：

- 写规则不需要配置"回传 CAN ID"。它只往 LIN 方向发，不会发任何 CAN 帧，界面上这个输入框对写规则是隐藏的。
- 校验和类型选"增强"或"经典"。拿不准就选增强（LIN 2.x 的主流）；对接老设备时若收不到，再换经典试。
- 目标 LIN ID 只能填 0x00~0x3F。填大了软件会直接拒绝下载，设备端也会拦。

## 3.2 类型二：CAN→LIN（读）

CAN 总线上来一帧报文，设备把它当作"读命令"：只在 LIN 总线上发出报文头（不带数据），等 LIN 从机应答，再把应答数据**原样**装进一帧 CAN，用规则里指定的"回传 CAN ID"发出去。常用于把 LIN 传感器采集的数据搬到 CAN 网络上。

### CAN→LIN（读）

**CAN 总线**

收到一帧报文：0x110，被一条读规则命中

设备把它当作“读命令”，不去 LIN 总线发数据

**LIN 总线（主机动作）**

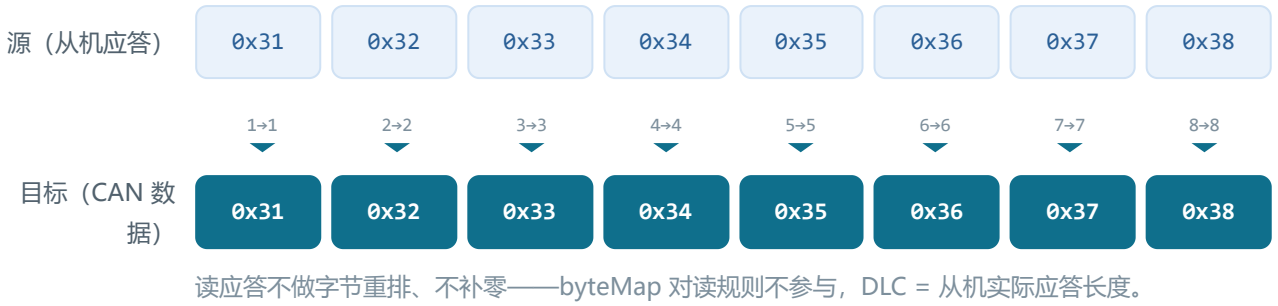
只发报文头：Break → Sync → PID **0x11**（ID 0x11）

**不带数据**——数据要由从机给

**LIN 总线（从机动作）**

从机对 PID 应答 8 字节：

0x31 0x32 0x33 0x34 0x35 0x36 0x37 0x38（校验和自动识别增强/经典）




读应答不做字节重排、不补零——byteMap 对读规则不参与，DLC = 从机实际应答长度。

**主机模式**

用**回传 CAN ID 0x201** 发一帧标准/扩展 CAN（由标志位决定），数据就是刚收到的应答。

**从机模式**

整条规则忽略：从机发不出报文头，也不能和总线上的真实主机抢调度。

要点：

- **只在主机模式下有效。**从机模式发不出报文头（报文头只能主机发），这类规则会被忽略，界面上也有灰色"从机不生效"的标记。
- "回传 CAN ID"必填且不能为 0，填 0 时下载会被拒绝。
- 回传数据不做字节重排、不补零，长度等于从机实际应答长度。字节映射对读规则不参与，界面上映射区同样隐藏。
- 读规则可以配周期（周期读），多条读规则并发不串台。
- 回传上来的数据全 0，先别怀疑设备：大概率是从机的应答本身就是 0。如果从机是另一台本设备（从机模式），需要先给它配一条写规则把数据写进缓存。

## 3.3 类型三：LIN→CAN

LIN 总线上出现一帧带数据的报文，设备按规则翻译成 CAN 帧发出去。这是**从机模式下唯一的 CAN 上报通路**：设备作为从机收到主机写来的数据、或应答完主机读之后，能不能把数据转给 CAN，完全看有没有配这条规则，没配就不发。

### LIN→CAN

**LIN 总线**

收到带数据的报文：ID **0x30**（PID 0xF0），长度 8

数据：0x11 0x22 0x33 0x44 0x55 0x66 0x77 0x88

**从机缓存**

先把数据写进 0x30 的缓存（64 槽，按 ID 索引，单值覆盖）

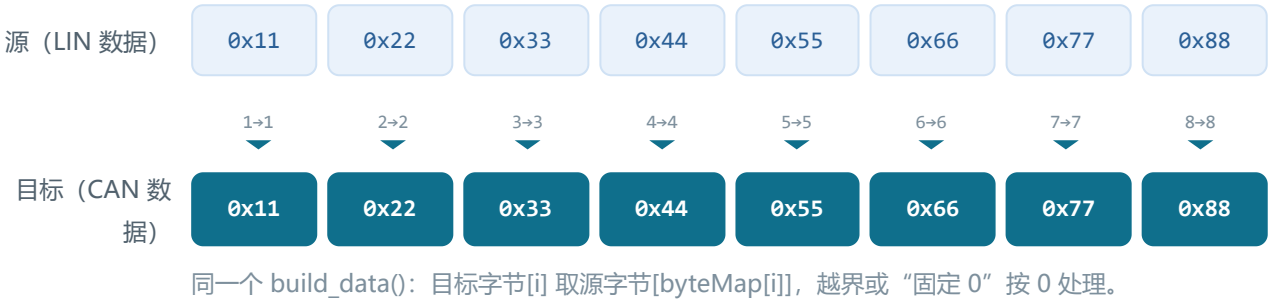
不管有没有 LIN→CAN 规则，缓存都会更新

**查规则表**

规则：源 0x30，掩码 0（精确匹配）

比对：( 0x30 & 0x3F ) = 0x30 == 0x30 **命中**

动作：生成 CAN 帧，目标 ID 0x003B0（29 位扩展帧）




同一个 build_data()：目标字节[i] 取源字节[byteMap[i]]，越界或“固定 0”按 0 处理。

**主机模式**

不参与：主机模式下读应答统一由“CAN→LIN（读）”规则的回传 CAN ID 上报，LIN→CAN 规则避免重复发送。

**从机模式**

把 CAN 帧发出去：ID 0x003B0，DLC 8。 勾了“数据不变不重发”，内容没变就不发。

要点：

- 只在从机模式下有效。主机模式有独立的回传通路（读规则的"回传 CAN ID"），LIN→CAN 规则不参与，避免同一份数据发两遍。
- "数据不变不重发"勾上后，内容与上一帧相同就不再发 CAN，用来抑制周期数据刷屏。
- "接收超时发全 0"用于信号失效场景：LIN 侧超过设定时间没来数据，主动发一帧全 0 的 CAN，告诉整车这个信号已经失效。
- 目标 CAN ID 可标准帧也可扩展帧（29 位），由"回传帧用 29 位扩展帧"勾选项决定。

## 3.4 示例规则逐条演示

配套的示例规则文件（canlin_test_rules.json）里有 11 条测试规则，覆盖了上面三种类型和几个容易误解的配置。网页版可以逐条点开播放动画，对照自己的场景：

**测试规则库：点击任意一条播放它的完整动作**

| # | 类型 | 源 ID | 掩码 | 目标 | 说明 |
|---|---|---|---|---|---|
| 1 | CAN→LIN 写 | 0x101 | — | 0x10 | 0x101 写 → LIN 0x10，增强校验，100ms 周期 |
| 2 | CAN→LIN 写 | 0x102 | — | 0x11 | 0x102 写 → LIN 0x11，事件触发 |
| 3 | CAN→LIN 读 | 0x110 | — | 0x201（回传） | 0x110 读 LIN 0x11，应答回传 CAN 0x201 |
| 4 | CAN→LIN 写 | 0x103 | — | 0x12 | 0x103 写 → LIN 0x12，经典校验，4 字节 |
| 5 | CAN→LIN 写 | 0x104 | — | 0x13 | 0x104 写 → LIN 0x13，字节交换映射 |
| 6 | CAN→LIN 写 | 0x105 | — | 0x14 | 0x105 写 → LIN 0x14，只取前 2 字节其余补 0 |
| 7 | CAN→LIN 写 | 0x19000000 | — | 0x15 | 0x19000000 扩展帧写 → LIN 0x15，200ms 周期 |
| 8 | CAN→LIN 写 | 0x1A0 | 0x1F0 | 0x16 | 0x1A0/0x1F0 掩码范围匹配（0x1A0~0x1AF）→ LIN 0x16 |
| 9 | LIN→CAN | 0x30 | — | 0x3B0 | LIN 0x30 → CAN 0x3B0 扩展帧，变化抑制 |
| 10 | LIN→CAN | 0x21 | — | 0x19000100 | LIN 0x21 → CAN 0x19000100 扩展，变化抑制 + 500ms 超时全 0 |
| 11 | LIN→CAN | 0x10 | — | 0x3A1 | LIN 0x10 → CAN 0x3A1，仅上报读应答 |

### 规则 1：0x101 写 → LIN 0x10，增强校验，100ms 周期

**CAN 总线**

收到一帧报文：**0x101**（标准帧），DLC 8

数据：0x11 0x22 0x33 0x44 0x55 0x66 0x77 0x88

**查规则表**

规则：源 0x101，掩码 0（精确匹配）

比对：( 0x101 & 0x1FFFFFFF ) = 0x101 == 0x101 **命中**

动作：翻译为 LIN，目标 ID 0x10

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 源（CAN 数据） | 0x11 | 0x22 | 0x33 | 0x44 | 0x55 | 0x66 | 0x77 | 0x88 |
| | 1→1 | 2→2 | 3→3 | 4→4 | 5→5 | 6→6 | 7→7 | 8→8 |
| 目标（LIN 数据） | 0x11 | 0x22 | 0x33 | 0x44 | 0x55 | 0x66 | 0x77 | 0x88 |

目标字节[i] 取 源字节[byteMap[i]]，与固件 build_data() 逐字节一致。

**LIN 帧**

ID 0x10 → PID **0x50**（校验位 P0/P1 自动计算）

长度 8，增强校验 → 校验和 **0x49**

周期 100 ms 重发

**主机模式**

把整帧发上 LIN 总线：报文头（Break + Sync + PID）→ 8 字节数据 → 校验和 0x49。

**从机模式**

只更新 0x10 的应答缓存（单值覆盖，只留最新值），**不上 LIN 总线**。等真实主机来读时再返回。

其中第 8 条（源 0x1A0、掩码 0x1F0）是掩码匹配的典型例子：0x1A0~0x1AF 范围内的 CAN ID 都会命中并转发到同一个 LIN ID，第 5.1 节有推导过程。

# 4. 主机与从机

## 4.1 模式从哪来

LIN 主/从由机壳上的 **LIN 主从拨码开关**决定，上电时读入，运行中不改：

| 拨码开关 | 模式 |
| --- | --- |
| 主机档（OFF） | 主机：报文头由本设备调度 |
| 从机档（ON） | 从机：报文头由总线上真实主机调度 |

当前模式不用自己猜：上位机连上设备后会自动读取，顶部徽标和规则卡片上的"主机/从机"徽标都按它显示，实心底并带"（当前）"字样的那枚是设备现在的模式。改拨码后必须重新上电才生效。

## 4.2 行为矩阵

同一条规则在两种模式下的动作对照。建议配置前先看这张表，确认每条规则在目标模式下是否真的有用：

| 总线事件 | 主机模式 | 从机模式 |
| --- | --- | --- |
| CAN 命中写规则 | 发 LIN 报文头 + 数据 + 校验 | 只写该 LIN ID 的缓存（单值覆盖），不上总线 |
| CAN 命中读规则 | 发报文头，应答按"回传 CAN ID"发 CAN | 忽略 |
| 收到 LIN 数据帧 | 只认本设备读命令的应答，按回传 ID 发 CAN | 先写缓存，再按 LIN→CAN 规则发 CAN，无规则不发 |
| 收到 LIN 报文头（主机读） | 不适用（本机就是主机） | 用缓存应答；有 LIN→CAN 规则则把应答数据发 CAN |
| 未配置规则的 LIN ID | 不上报 | 不应答、不上报 |

两张表合起来看：主机模式用得上"写 + 读"两类规则，从机模式用得上"写（缓存）+ LIN→CAN"两类。一张表配下来通常只对其中一种模式全部有效，另一种模式下会有一部分规则闲置，这是设计如此，不是配置错误。

## 4.3 发布与监听

LIN 协议规定一个 ID 在总线上只能有一个发布者。设备从机模式下对这个 ID 应答、还是监听这个 ID，由规则表自动推导：**配了 CAN→LIN 写规则、且源掩码为 0 的 LIN ID，就是本设备发布的 ID**（主机会来读它）；其余 ID 一律按监听处理（收到写就缓存，是否转 CAN 看 LIN→CAN 规则）。

由此有两个边界要知道：

- 没配写规则的 ID 被主机读时，设备**不应答**。这不是故障，是避免回一帧无意义的空数据。
- 带掩码的写规则无法静态判定目标 ID，这类 ID 的方向退化成"是否预置过应答数据"来判定。新配置尽量别用掩码写规则。

# 5. 规则字段逐项说明

## 5.1 源 ID 与源掩码

匹配条件是：

```
(帧ID & 掩码) == (源ID & 掩码)
```

掩码填 0 表示精确匹配：CAN 源按 29 位全比、LIN 源按 6 位全比。需要范围匹配时才填非 0 掩码，例如源 ID 0x1A0、掩码 0x1F0，则 0x1A0~0x1AF 都命中（0x1A0 & 0x1F0 = 0x1A0，而 0x1A3 & 0x1F0 同样等于 0x1A0）。掩码位为 1 的位参与比较，为 0 的位忽略。

还有一条隐含规则：帧类型必须一致。规则勾了"源 ID 是扩展帧"就只匹配 29 位扩展帧，否则只匹配标准帧。标准帧 0x123 和扩展帧 0x00000123 是两个东西，不会互相误匹配。

## 5.2 目标 ID

CAN→LIN 两种类型里，目标 ID 是 LIN ID（0x00~0x3F）；LIN→CAN 里是 CAN ID（标准 11 位，勾扩展帧则 29 位）。

## 5.3 回传 CAN ID

只有 CAN→LIN（读）规则有这个字段：从机应答回来后，用哪个 CAN ID 发回去。必填，且不能为 0。

## 5.4 校验和与长度

- 校验和类型：增强（LIN 2.x）或经典（LIN 1.x）。读规则不用选，设备应答时自动识别两种。
- 长度：0 表示沿用源报文长度，1~8 表示固定长度。写规则固定长度时，源数据不足的字节按映射规则补 0。

## 5.5 字节映射

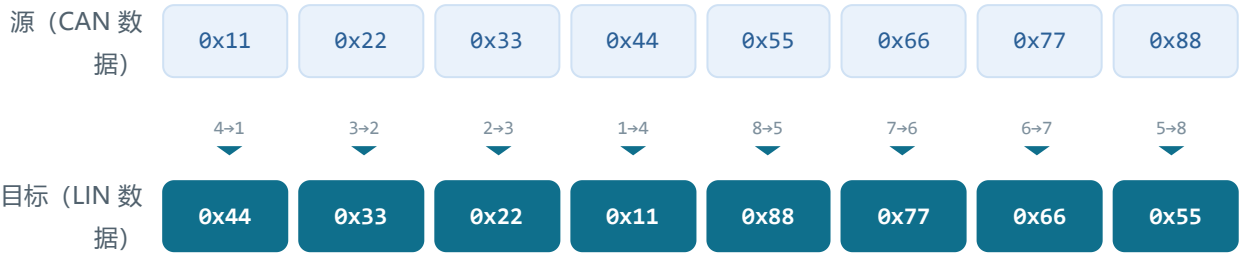
字节映射决定"目标的 8 个字节分别从哪来"。上位机里是 8 个下拉框，第 1~8 个字节各自选"取源 1~取源 8"或"固定 0"：

```
目标字节[i] = 源字节[byteMap[i]]        （byteMap[i] < 源长度时）
目标字节[i] = 0x00                      （byteMap[i] 为"固定 0"，或超出源长度时）
```

三个典型例子（动画里可以逐字节看对应关系）：

**字节映射：目标 8 字节分别从哪来**

正序直通 | 字节交换 | 部分固定 0




两端字节序相反时使用，4 字节一组倒序。

JSON 对应字段：`"byteMap": "03,02,01,00,07,06,05,04"`

正序直通（0,1,2,3,4,5,6,7）最常见；字节交换（3,2,1,0,7,6,5,4）用于两端字节序相反的场合；"固定 0"用于只搬运部分字节、其余补零。注意映射引用的源字节超出本次实际收到的长度时，该字节同样按 0 处理。

## 5.6 周期与超时

- 周期 ms：大于 0 时，规则按此周期主动重发（写规则重发上次的 LIN 数据；读规则周期性发读头）。0 表示只由 CAN 报文触发。
- 超时 ms：只对 LIN→CAN 有效。LIN 侧超过这个时间没来数据，且勾了"接收超时发全 0"，就发一帧全 0 的 CAN。

## 5.7 勾选项

| 勾选项 | 含义 | 适用类型 |
| --- | --- | --- |
| 源 ID 是扩展帧 | 源 ID 按 29 位扩展帧匹配 | CAN→LIN 两种 |
| 回传帧用 29 位扩展帧 | 发出的 CAN 帧用 29 位扩展帧 | 读、LIN→CAN |
| 数据不变不重发 | 数据与上次相同则不发 | LIN→CAN |
| 接收超时发全 0 | 超时后发一帧全 0 | LIN→CAN |

写规则不涉及扩展回传帧（它不发 CAN 帧），界面上对应项对写规则显示为"不适用"。

# 6. 配置的保存与下载

## 6.1 转换规则页怎么用

每条规则一张卡片，标题栏有序号、类型下拉和自动摘要。常用的操作都在页顶工具栏：

- **读取设备配置**：把设备里的规则全部读回来显示，改动前先读一遍是习惯；
- **下载到控制器**：软件先自动校验全部规则，通过后写入设备，写完自动回读逐条核对，全部一致才提示成功；
- **清空设备规则**：向设备下发 0 条空表，等于关闭全部转换；
- **导入 / 导出 JSON**：规则表可以存成文件备份或发给同事；导入时软件先整体校验，有错的文件不会被写入一半。

编辑规则时注意三处易错点：读规则的回传 CAN ID 不能空；LIN ID 别超 0x3F；写规则不需要也不显示回传帧配置。卡片右侧的"复制"可以快速建相似规则，上下箭头调整顺序（匹配按顺序进行，命中即执行，多条规则可同时命中）。规则表最多 200 条，加满后计数会变红。

## 6.2 掉电安全

规则在下载过程中先写在临时区，全部校验通过后才一次性写入存储并回读比对。所以下载中途拔线、掉电，最坏结果是这次配置没写成，设备里还是旧配置，不会出现半新半旧；设备运行中掉电也不伤已有配置。

下载写入的瞬间（一般一两秒）设备总线收发会有短暂停顿，产线配置没有影响，请避免在整车通信关键时段执行下载。

## 6.3 下发被拒的常见原因

| 提示 | 处理 |
| --- | --- |
| 回传 CAN ID 空 / 为 0 | 读规则必须填一个有效的回传 CAN ID |
| LIN ID 超出 0x3F | 改成 0x00~0x3F 范围内的值 |
| 规则数超上限 | 一台设备最多 200 条，删掉不用的再下载 |
| 与设备协议版本不符 | 设备固件太旧，先按第 8 章升级固件 |

# 7. 参数配置

**参数配置**页集中管理设备基础参数。页面上**从设备读取**、**写入到设备**两个按钮：改前先读取确认当前值，改完写入立即生效并掉电保存。

| 参数 | 说明 |
| --- | --- |
| 站号 | 设备通信地址，1~247。改完站号软件会自动按新站号重连 |
| CAN 波特率 | 选 0 表示由**CAN 波特率拨码开关**决定（拨码 ON → 250 kbps，OFF → 125 kbps）；选非 0 值（20~1000 kbps）则以软件设置为准。改回 0 后需给设备重新上电，波特率才交回拨码控制 |
| LIN 波特率 | 单位 bps（常用 19200）。填 0 表示不修改 |

改完站号后按新站号重连。打开连接后软件会自动读一次拨码状态，规则页的"主机/从机"徽标由此而来；如果徽标显示"未知"，点一下规则页任意位置触发重读。

设备 SN 序列号在**软件升级**页只读显示，不允许在配置软件里修改，避免现场误改。

# 8. 固件升级

设备功能更新通过**软件升级**页刷写固件完成，页面分三部分：

1. **设备信息**：SKU 型号、SN 序列号、当前固件版本，连接后自动显示；
2. **固件升级**：点固件路径右侧按钮选择固件文件（.bin），再点**开始升级**，进度条走完即升级成功。升级过程约十几秒，期间**切勿断开串口或给设备断电**；
3. **串口收发日志**：底部窗口实时显示通信字节，绿色是设备发来的数据，蓝色是本机发出的命令，方便排查升级异常。

升级完成后给设备重新上电，软件升级页会显示新版本号；版本差异见上位机"版本信息"页。

# 9. CAN / LIN 收发调试

顶部**CAN 收发**、**LIN 收发**两个调试页不走规则表，直接操作总线，用于现场验证。

## 9.1 LIN 收发

LIN 页三种模式：

- **主机发送**：发一条完整 LIN 帧（报文头 + 数据 + 校验和）；
- **主机读取**：只发报文头，等从机应答；
- **从机应答**：把数据写进设备应答表，等总线上真实主机来读。从机应答固定使用增强校验。

模式和设备拨码不符时只弹气泡提示不拦截，方便在旧固件上使用。设备自动应答后会以"自动应答"方向显示在接收表里，能看到答了什么。接收表对每条收到的报文标注校验结果：**增强校验通过**或**经典校验通过**；若报文标注是增强校验、但数据实际按经典校验才对得上，会显示"经典校验通过（设备标注增强）"，出现这种提示说明发送端校验和类型与标注不一致，值得排查。

## 9.2 CAN 收发

CAN 页支持标准帧 / 扩展帧、数据帧 / 远程帧，可设周期发送（10~1000 ms）。断开串口后周期发送自动停止。

# 10. 版本信息与运行日志

## 10.1 版本信息页

上位机最后一个页签是**版本信息**，列出了软件各版本的主要功能变化，以及公司联系方式。升级上位机前可以先看这页确认需要哪些新功能。

## 10.2 运行日志

软件会把重要操作自动记录到日志，方便后期排查问题：

- 日志保存在软件所在目录的 **log** 文件夹里，按日期命名；
- 单个日志文件有大小上限，超出后自动分新文件，旧文件自动清理；
- 日志只记录操作和结果（如连接、读写配置、固件升级、报错），**不记录任何报文内容和个人隐私信息。**

报修时把当天日期的日志文件发给我们即可。

# 11. 常见问题

| 现象 | 原因与处理 |
| --- | --- |
| 打开串口失败 | 串口被其它软件占用，关掉占用程序重试 |
| 设备模式一直"未知" | 设备固件过旧（低于 1.0.23），先按第 8 章升级固件；或串口被其它软件占用 |
| 读取 / 写入超时 | 站号不符，或设备不在位；确认站号与连接 |
| 写规则下发被拒 | 读规则回传 CAN ID 空/为 0；LIN ID 超 0x3F；规则数超 200（见 6.3） |
| 从机模式读规则不动作 | 设计如此，读规则只在主机模式生效（见 4.2） |
| 回传上来全是 0 | 从机应答本身就是 0，先查从机侧缓存（见 3.2） |
| 下载提示协议版本不匹配 | 设备固件太旧，先升级固件 |
| 拨码改了没反应 | 改拨码后必须给设备重新上电才生效 |

## 12. 技术支持

**成都讯川科技有限公司**Chengdu Xunchuan Technology Co., Ltd.

- 官网：www.xcmotion.com
- 邮箱：support@xcmotion.com

反馈问题时请附上：软件版本（版本信息页可见）、设备固件版本（软件升级页可见）、当天的运行日志文件，以及一句现象描述。我们会尽快回复。