FIRMWARE MANUAL

# 单路 FOC 驱动板 出厂固件说明书

FOC_Single · 出厂固件 · 使用说明书

| | |
|---|---|
| **品牌** | Daliang Auto |
| **产品名称** | FOC_Single 单路 FOC 电机驱动板 |
| **固件名称** | FOC_single |
| **固件版本** | 0.1.0（构建标识见随包 `MANIFEST.txt`） |
| **文档版本** | Rev. 1.0 |
| **发布日期** | 2026-09-09 |
| **控制台** | USB Type-C 虚拟串口，115200 8N1，无流控 |
| **对外接口** | 控制台命令行 · `=tlm` 遥测 · 整车 CAN · 遥控接收机 |

> **本书与规格书的分工**
>
> - **《FOC_Single 产品规格书》描述硬件**：电压平台、功率、电流、接口、引脚、 连接器、保护器件、机械尺寸。硬件不改版，规格书就不改。
> - **本书描述出厂固件**：命令、标定流程、控制逻辑、保护动作、通信协议、升级。 **固件每发布一版，本书跟着走一版**，两者互不牵动。
> - 两本书都出现的量（母线电压范围、电流量程），**以规格书为准** —— 那是硬件事实，本书只说固件怎么使用它。
>
> **板子上跑的是哪一版，用 `ver` 问它，不要从出厂日期推。** 而且 `ver` 报的 `cmdhash` 必须和随包 `COMMANDS.json` 里的相等 —— 那是唯一一条把这份包和手上这块板绑在一起的证据，版本号不是。
>
> ⚠⚠ **这块板和 DlaSteer-100 转向执行器用的是同一块 PCB。** 两个固件烧进去都能跑、都能过校验、都会在串口上说话。 认固件只认发布包 `MANIFEST.txt` 里的 `name: FOC_single`， 不要按目录名、也不要按"哪一版最新"挑。

---

## 1. 这套固件是什么

出厂固件把这块板变成一台**可以直接使用的单路无刷电机驱动器**： 接上电机、霍尔（或编码器）和电池，用一根 USB 线做一次标定， 之后油门、电位器、遥控器或整车 CAN 都能驱动它。客户不需要写一行代码。

### 1.1 功能地图

| 层 | 固件提供什么 |
|---|---|
| **电机控制** | 三相 FOC：电流（转矩）、速度、开环电压、I-f 起动四种模式 |
| **自学习标定** | 一条命令测出这台电机的接线、相电阻电感、霍尔换相表与电流环增益，存在板子自己的 Flash 里 |
| **位置反馈** | 霍尔（标配）+ 编码器（Rev 2.0 起，SPI 绝对 / ABZ 增量）。⚠ 编码器不可用会**自动退回霍尔** |
| **指令来源** | 油门、电位器、遥控接收机（SBUS / CRSF / IBUS，可自动识别）、整车 CAN、控制台 |
| **保护** | 过压、欠压、过流、栅极驱动器故障、过温、NTC 失效、编码器失步、遥控失联、电流采样不可信 —— 九个故障位，切桥一律锁存 |
| **停机方式** | 自由滑行 或 **主动短路**（高速时按反电动势自动选） |
| **看门狗** | 独立看门狗 + 五个 CPU 故障向量，任何一个都先关栅极再复位并记录（0.1.0 起） |
| **对外接口** | 人用的命令行、程序用的 `=tlm` 快照、整车用的 CAN |

### 1.2 三条对外通路

| 通路 | 对面是谁 | 物理层 | 本书章节 |
|---|---|---|---|
| 控制台命令行 | 人，或上位机软件 | USB Type-C（USART1），115200 8N1 | §2 – §6 |
| `=tlm` 快照 | 程序：上位机、测试脚本 | 同上，与命令行共用一条串口 | §8.1 |
| 整车 CAN | 上位控制板 | 经典 CAN，1 Mbps | §7 |

---

## 2. 接上控制台

### 2.1 接线与串口参数

| 项 | 值 |
|---|---|
| 物理口 | USB Type-C（板载 USB-UART 桥，接 USART1） |
| 波特率 | 115200，8 数据位，无校验，1 停止位，无流控 |

| 项 | 值 |
| --- | --- |
| 换行 | 板子回 `CR LF`；发过去 `CR` 或 `LF` 都认 |

敲 `help` 会列出全部 77 条命令。

### 2.2 每条命令都标了它会不会让机器动

随包 `COMMANDS.json` 给每条命令标了一个 `danger`：

| 标记 | 含义 | 例子 |
| --- | --- | --- |
| `none` | 只读，回答一个问题 | `ver` `tlm` `prot` `hall` `enc` `drv` |
| `gates` | 改状态或改一个设置，但**转不动任何东西** | `save` `clear` `stop` `pi` `ibus` |
| `motion` | **可能让电机转起来** | `drive` `cl` `spd` `learn` `iqmax` |

⚠ **一条命令按它最危险的那种形式分类。**

### 2.3 任何时候都可以打断的一条

```
stop
```

`stop` 立刻关掉栅极总使能并把栅极驱动器置于安全态。

⚠ `stop` 之后电机是**自由滑行**的 —— 一个转起来的轮子会滑行很久。

### 2.4 最先该敲的三条

```
ver          这是哪一版固件、哪块板、cmdhash 是多少
tlm          此刻的完整状态，一屏（给程序看的那一份）
prot         保护状态和门限
```

`ver` 的回答里有一行是给程序读的：

```
=tlm 1 build debug=0 major=0 minor=1 patch=0 board=foc_single scm=<commit> hw=2.0 cmdhash=<数>
```

⚠ `debug=1` 表示这是 **Debug 构建，不要拿它驱动真实负载。**

---

## 3. 标定

### 3.1 ⚠⚠ 先把轮子架空

标定会让电机**转起来并且滑行**。这是全套命令里唯一一个"松手之后还在动"的动作。

### 3.2 一条命令

```
learn
```

它按顺序做四件事：

1. **电流零点** —— 桥关着，量三相采样的零位
2. **霍尔映射** —— 转一圈，记下六个霍尔状态各自对应的电角度
3. **相电阻 / 电感** —— 开环定电压激励，量 R 和 L
4. **电流环增益** —— 从 R、L 和母线电压自动整定

跑完之后：

```
save
```

⚠ **不存就白做了**：标定结果在 RAM 里，掉电即失。

### 3.3 板子测不出来的那几个

| 参数 | 为什么只能人填 |
|---|---|
| **极对数** `pp` | ⚠⚠ 厂家规格书上的 `Poles` 是**总极数**，极对数 = Poles ÷ 2。直接抄那一行是这个项目付过账的错 |
| **轮子/负载相关的限值** | `iqmax`（转矩上限）、`ibus`（母线电流折返限）按电机铭牌和电池能力填 |
| **装在哪一侧** `side` | ⚠ 不只是标签：它同时决定同一条指令把轮子往哪边转，也决定这块板在 CAN 上占哪一组 ID |
| **编码器型号与线数** | 接了编码器才需要（`encchip` / `enccpr`），然后 `enccal` |

⚠ **极对数怎么核对**：`tlm` 的 `mot` 那一组里有 `flux_uwb`（磁链估计）。 把它和电机铭牌的电压常数反算出来的值比一比 —— 那是一条**不经过极对数**的 独立证据。两者差一倍，就是极对数错了一倍。

### 3.4 ⚠ 标定之前板子拒绝闭环驱动

没标定过时 `tlm` 的 `hall_calibrated` 是 0，闭环命令会被拒。这是有意的： 没有换相表，闭环就是在猜。

---

## 4. 运行模式与指令来源

```
mode              查询
mode idle         空转，不驱动
drive on|off      总使能
```

指令来源（谁说了算）：

| 来源 | 怎么进来 | 命令 |
|---|---|---|
| 控制台 | 你敲的 | `cl <mA>` 转矩 / `spd <eHz>` 转速 |
| 油门 | 模拟口 | 先 `thrmin` / `thrmax` 标定 |
| 电位器 | 模拟口 | 先 `potmin` / `potmax` 标定 |
| 遥控接收机 | 串行协议 | `rc` 看状态， `rcch` 选通道 |
| 整车 CAN | `0x210` 广播帧 | 见 §7 |

⭐ **换模式本身产不出转矩**：换模式会先重置电流调节器并把两个参考清零， 必须有别的东西随后写一个非零参考。⇒ 换模式是"要问一次"级别， 而**给桥上电（`drive on`）是最高一档** —— 上电会施加此刻已经立在那里的 参考值，那个值可能是上一条命令留下的。

### 4.1 转矩还是转速

```
srcmap torque     指令 = 转矩（出厂）
srcmap speed      指令 = 转速目标
```

⚠ 改成转速之后，松开指令车会**主动减速到零**而不是滑行。 这是两种完全不同的驾驶感受，不是一个调参。

---

## 5. 停机：滑行还是短路

停机有两种，板子按**反电动势**自动选：

| 方式 | 什么时候 | 结果 |
|---|---|---|
| 自由滑行 | 转速低、反电动势小 | 桥全开，电机自己转到停 |
| **主动短路（ASC）** | 反电动势高到母线的一定比例 | 三相低边全开，电机被电磁力矩制动 |

⚠⚠ **为什么高速时不能只是"断开"**：高速滑行时电机是一台发电机， 反电动势会把母线顶上去。短路把能量消耗在电机自己的电阻里。

⚠ 那两个门限（`ascpu`）设错了会**在高速时短接三相**。设它们的命令因此在 命令表里被归为 `motion` 而不是 `gates` —— 它不动电机，但它决定电机怎么被停住。

---

## 6. 保护与故障

### 6.1 九个故障位（`prot` / `tlm` 的 `fault=`）

| 位 | 名字 | 含义 |
|---|---|---|
| 0 | `OV` | 过压 |
| 1 | `UV` | 欠压 |
| 2 | `OC` | 软件过流（20 kHz 中断里判） |
| 3 | `NFAULT` | 栅极驱动器报故障（经中断立即切桥） |
| 4 | `OT` | 功率管过温（NTC） |
| 5 | `NTC` | ⚠ NTC 开路或短路 —— **温度读数不可信** |
| 6 | `ENC` | 编码器失步或不响应 |
| 7 | `RC` | 遥控链路失联（失效保护） |
| 8 | `IMEAS` | 三相电流残差不合理 —— 采样链本身有问题 |

- 过压/欠压门限**随上电检测到的电池节数自动缩放**，另有绝对上下限兜底。
- ⚠ **切桥一律锁存**，要 `clear` 才解除，而且原因还在的时候清不掉。

### 6.2 ⚠⚠ 温度那两位要一起读

**开路的 NTC 读出来是偏冷的。** 一个合理的温度数字旁边如果 `ntc_ok=0`， 那个数字不能用 —— 而它看起来完全正常。⇒ `tlm` 的 `prot` 那一组把这两个 并排放，就是为了这件事。

### 6.3 ⚠ 换完电池、或者主电断过之后，敲一次 `clear`

栅极驱动器的供电取自**主电**。主电断了而逻辑电还在（台架插着 USB、 换电池就是这个），它会复位回**出厂寄存器** —— 电流采样增益从 10 V/V 变成 20 V/V，而固件按 10 算。

⇒ 之后**每个电流读数都少一半**，安静地：环欠驱动、跳闸提前、 `learn` 存下错的 R/L。**而 `drv` 打出来的那七个寄存器在这种状态下看起来 完全正常。**

本版做了两件事：

- `drv` 命令最后会给出**判断**：「这是我们的配置」或者「不是我们的」
- `clear` 和每一次使能之前都会检查并重写它

### 6.4 看门狗（0.1.0 起）

在这之前这块板**一个看门狗都没有**，五个 CPU 故障向量全是空死循环 —— MCU 挂死意味着桥按它死掉那一刻的占空比继续开关。

现在：

- **独立看门狗**，主循环喂。停止喂食就复位。
- **五个 CPU 故障向量**（HardFault / NMI / MemManage / BusFault / UsageFault） 一进去就先关栅极，然后把故障现场记进后备寄存器，再复位。
- 下一次开机，控制台会说出上一次是怎么死的：

```
!! the previous run stopped feeding the watchdog
!! previous run died in a CPU fault: id=1 pc=... lr=... cfsr=... hfsr=... at ... ms
```

⚠ 这条记录**只说一次**（读完就清），所以看到了就记下来。

### 6.5 固件做不到的那一件

⚠⚠ 板上**没有一条绕过 CPU 的硬件关断通路。**所有保护都经过 MCU。 载人或高风险场合必须另配独立的物理断路器，不经过这块板。

---

## 7. 整车 CAN

经典帧，1 Mbps。这块板在整车上是**执行端**：控制板广播行驶请求，它执行。

### 7.1 行驶请求 `0x210`（控制板 → 本板，广播给两块单路板）

| 字节 | 内容 |
|---|---|
| 0:1 | 油门千分比 u16 LE，0..1000 |
| 2:3 | 转向（本板忽略） |
| 4 | 标志：bit0 允许驱动 / bit1 刹车 |
| 5 | 挡位：0=N 1=F 2=R |
| 6 | 序号 |
| 7 | 指令超时 ×10 ms（0 = 用默认） |

⚠ **这一帧永不停发**，"安全"靠内容表达。收不到帧的话本板要等自己的超时才停 —— **发一帧"停"比不发快一个超时窗口。**

### 7.2 状态与遥测（本板 → 控制板）

两组 ID，按 `side` 参数选：**左轮** `0x211` / `0x212`，**右轮** `0x213` / `0x214`。

状态帧：

| 字节 | 内容 |
|---|---|
| 0 | 健康位（本侧的：运行中 / 霍尔边沿正常 / 栅极驱动无故障 / 指令新鲜） |
| 1 | 序号 |
| 2:3 | 左轮转速 s16 LE |
| 4:5 | 右轮转速 s16 LE |
| 6 | 回显收到的指令标志字节 |

| 字节 | 内容 |
|---|---|
| 7 | bit7 = 本板认为指令有效；bit0-3 = 解出的挡位 |

遥测帧：

| 字节 | 内容 |
|---|---|
| 0:1 | 母线电压 mV u16 LE |
| 2:3 | 母线电流 cA s16 LE（10 mA/LSB） |
| 4 | 序号 |

⚠⚠ **健康字节里每一位都是「1 = 好」**，包括那些硬件上低有效的信号 —— 约定不是"照抄引脚电平"，是统一归一到 1 = 好。

⚠ `side` 填错的后果：两块单路板配一辆车时，两块板会抢同一组 ID， 或者一侧反转 —— 而**两块板都"工作正常"**。

---

## 8. 给程序用的接口

### 8.1 `tlm` —— 一次机器可读的状态快照

```
tlm
```

回一帧多行记录，收在 `=tlm end`：

```
=tlm 1 sys   ms=... vbus_mv=... cells=... fault=... warn=... state=... moe=... wdog=... cpufault=...
=tlm 1 prot  uv_mv=... ov_mv=... oc_cnt=... iqmax_ma=... iqcont_ma=... iqlim_ma=... therm_m=... temp_mc=... ntc_ok=... ntc_r
=tlm 1 mot   mode=... op=... drive=... side=... hall=... edges=... dir=... omega_m=... iq_ma=... iqref_ma=... ibus_ma=... re
=tlm 1 enc   chip=... cpr=... pp=... dir=... src=... ok=... pos=... spi_raw=... omega_m=... slips=... spi_err=...
=tlm 1 can   tx=... rx=... err=...
=tlm 1 build debug=... major=... minor=... patch=... board=foc_single scm=... hw=... cmdhash=...
=tlm end
```

规矩：

- **只有整数，单位在名字里**（`*_ma` 毫安、`*_mc` 毫摄氏度、`*_m` 千分之一）
- `=tlm end` **不是装饰**：没有它，读的一方只能等线路安静，而那分不出 "读完了"和"被截断了" —— 一份短答复的每一行都仍然格式完美
- ⚠⚠ **没有任何一个字段是由两个字段算出来的。**快照不是原子的 （20 kHz 中断会在两行之间跑），拿一行的电压乘另一行的电流就是两个不同时刻 相乘。读的一方也不要这么做。

⚠ `mon` 是**另一回事**：那是给人看的一条 4 Hz 文字流，程序读不了。

### 8.2 ⚠ `angle_src` 和 `angle_active` 不是一回事

- `angle_src` 是**配置**：你让它用编码器还是霍尔
- `angle_active` 是**现状**：它此刻真正在用哪一个

编码器不可用时板子会**自己退回霍尔**，而 `angle_src` 不变。 ⇒ 一块配了编码器却已经悄悄退回霍尔的板子，跑的角度和它被整定时的不是同一个。 这两个字段并排放，就是为了让这件事看得见。

### 8.3 `param` —— 参数表

```
param list                  整张表：id / 名字 / 单位 / 范围 / 默认 / 权限 / 当前值
param get <名字>
param set <名字> <值>
param save
```

⚠ **不要把范围和默认值抄进别的文档。** 它们跟着固件走，抄本第二天就是错的，而且抄本看起来完全正常。

⚠ `param list` 的回复带 `begin` / `count=` / `end`。**数出来不等于 `count` 就把整份丢掉** —— 一张被打断的表，每一行仍然格式完美。

### 8.4 命令表 `COMMANDS.json`

发布包里带一份：每条命令的名字、参数形状、前置条件和危险等级。

⚠⚠ **用它之前先比 `cmdhash`**：板子 `ver` 报的和文件里的必须相等。 不等就整份不要用。

---

## 9. 参数

### 9.1 板子自己是参数表的真源

64 行。⚠ 每一行的**默认值都是从固件里真正执行它的那个常量推导出来的**， 不是手抄的 —— 2026-09-09 之前有六行是手抄的，其中两行（转矩电流上限、 母线电流折返限）和板子开机真正用的值对不上。

### 9.2 三条使用规矩

1. **桥活着的时候写不进去。** 标定和几何那些行在电机使能时直接拒绝。
2. `param set` **回的是板子事后真正持有的值**，不是 OK，也不是把你发的抄回来。
3. **改完要存**：`save`。⚠ 有一行故意不进 flash（见下）。

### 9.3 两类特殊的行

| 类 | 例子 | 为什么 |
|---|---|---|
| **只读的测量结果** | 编码器偏置、编码器方向 | 它们是一次测量的结果。一个能填的框会让人去调答案，而不是重做测量 |
| **故意不持久化** | `ibus_limit_ma` | 它描述**这次台架的电源**，不描述这块板。掉电回到出厂值 —— 会被人忘掉的那一个是临时的那一个 |
| **活值和默认值本来就不同** | `ov_fault_mv` | 开机取绝对上限，随后按检测到的电池节数换算。⇒ `default` 那一栏是检测之前的底，不是板子正在用的数 |

### 9.4 ⚠ 错了没有任何症状的那一组

`param list` 会在这些行上带 `silent=1`：相电阻电感、换相角偏置、极对数、 死区与延迟补偿、霍尔观测器那三项、编码器线数，以及**输入标定那四行。**

⚠⚠ 输入标定尤其要紧：范围标反或者没标全，**松开的踏板会读成踩满**， 而板子上没有任何东西会不同意。

---

## 10. 固件升级

### 10.1 两条路

| 路 | 怎么走 |
|---|---|
| **串口 ISP** | USB Type-C，芯片自带的系统 Bootloader。板上有一键下载电路，免按键 |

| 路 | 怎么走 |
|---|---|
| **SWD** | ST-Link，产线和研发用 |

⚠ ST-Link 烧录或复位时，不要开着占用同一个串口的工具。

### 10.2 ⚠ 升级之后重新标定

标定页的位置在 0.1.0 里挪过（从 `0x0803F800` 到 `0x0801F800`）—— 旧地址在这颗芯片**声明的容量之外**，读得回来只是因为裸片比标称的大。 旧固件写下的标定不保证能被新固件正确读出。

⇒ 升级后第一件事：`tlm` 看 `hall_calibrated`，需要就重跑 `learn`。

### 10.3 ⚠ 默认构建

发布包里的是 Release。**判据不是文件名 —— 是板子 `ver` 报的 `debug=` 那一位。**

---

## 11. 版本历史

### 11.1 固件

| 版本 | 日期 | 变化 |
|---|---|---|
| 0.1.0 | 2026-09-09 | **独立看门狗 + 五个 CPU 故障向量**（在这之前一个都没有）； `tlm` 机器可读快照；栅极驱动器掉配置的检查与重写（使能前、 `clear` 时、 `drv` 给判断）； 参数表默认值改为从常量推导（六行原本和开机值对不上，两行是限值）； 标定页挪进芯片声明的 flash；删掉一份写着错误容量的旧链接脚本 |

### 11.2 本书

| 版本 | 日期 | 变化 |
|---|---|---|
| Rev. 1.0 | 2026-09-09 | 首版。此前这个产品**没有出厂固件说明书** |

---

## 12. 已知限制（固件侧）

| # | 限制 | 影响 |
|---|---|---|
| 1 | **看门狗和 CPU 故障处理没有上板验过** | 真验收是「标定跑完不复位」和「故意打一个 CPU 故障，看下一次开机报不报」 |
| 2 | **没有硬件关断通路** | 所有保护都经过 CPU，见 §6.5 |
| 3 | **没有 C-5 那套结构化标定向导** | 标定进度只能读 `learn` 自己打的字 |
| 4 | **Debug 构建的 CPU 占用没有在这块板上量过** | 不要引用驱动板的那个数 |
| 5 | **对外型号还没定** | 规格书写的 `PF1-400-A` 属于双路板已作废的那套命名 |

---

## 13. 附录：常用的十条

| 命令 | 做什么 |
|---|---|
| `ver` | 哪一版固件、哪块板、 `cmdhash` 多少 |
| `tlm` | 此刻的完整状态，一屏（给程序） |
| `prot` | 保护状态和门限 |
| `drv` | 栅极驱动器寄存器 + **配置还是不是我们的** |

| 命令 | 做什么 |
| --- | --- |
| `learn` | 电机自标定（⚠ 先架空轮子） |
| `save` | 标定存进 flash |
| `pp <n>` | 填极对数（⚠ 是极对数不是总极数） |
| `side left\|right` | 装在哪一侧（⚠ 同时决定转向和 CAN ID） |
| `clear` | 清一条锁存的故障，并重写栅极驱动器配置 |
| `stop` | 立刻关栅极 |

---

*Daliang Auto · FOC_Single 出厂固件说明书 Rev. 1.0 · 2026-09-09*