



CANopen 协议库接口使用手册

文档所属：长沙六叶树教育科技有限公司

官方网址：www.liuyeshu.cn

软件版本：V1.1

文档版本：V1.1.0

编制日期：2026-05-24

适用平台：纯 C 跨平台、无硬件依赖、支持 CAN2.0/CANFD

保密等级：公开

1. 文档概述

1.1 产品简介

LCO_CANOPEN 是长沙六叶树教育科技有限公司自主研发的轻量化、纯 C 实现 CANopen 协议报文编解码库，完全遵循 CiA301、CiA402 标准协议规范。库体无任何硬件平台依赖，可适配各类 MCU、工控主机、嵌入式 Linux 等软硬件平台，原生支持标准 CAN2.0、CANFD 报文解析与封装，覆盖 NMT、SDO、PDO、SYNC、EMCY、心跳、节点守护、时间戳等全部核心 CANopen 服务，同时集成完整的伺服电机 CiA402 运动控制协议接口。

本库主打轻量化、高适配、易移植、接口简洁的特点，剥离底层硬件驱动逻辑，仅通过用户自定义回调函数对接底层 CAN 收发驱动，大幅降低 CANopen 协议二次开发难度，广泛适用于工业伺服控制、CAN 总线设备组网、自动化设备通信、教学实训设备等场景。

1.2 版本信息



库版本	文档版本	更新日期	更新内容
V1.1	V1.1.0	2026-05-24	初始正式版本，完整支持 CAN2.0/CANFD、 标准 CANopen 协议、CiA402 伺服控制，完善全部对外 API 接口

1.3 核心特性

- **纯 C 无依赖**: 标准 C 语言编写，无第三方库依赖，跨平台可移植性极强
- **双总线支持**: 原生兼容 CAN2.0、CANFD 报文，支持超长帧数据传输
- **全协议覆盖**: 支持 NMT 网络管理、SDO 读写、PDO 过程数据、SYNC 同步、EMCY 故障上报、心跳监测、节点守护、时间戳服务
- **集成 CiA402**: 内置伺服电机专用控制接口，适配位置、速度、转矩、回零、同步控制等主流运动模式
- **解耦硬件层**: 协议层与硬件驱动完全分离，通过回调函数对接底层收发，适配任意 CAN 驱动
- **完善状态机制**: 标准化错误码、状态机管理，便于异常排查与程序容错处理

1.4 适用范围

本文档适用于使用 LCO_CANOPEN 协议库进行二次开发的工程师、技术实训人员、设备调试人员，用于指导用户完成库初始化、接口调用、报文解析、伺服控制、异常处理等全流程开发工作。

2. 开发环境与依赖



2.1 编译环境

- 编译器：标准 C/C++/C#QT 语言编译器
- 语言标准：C89 及以上

2.2 依赖文件

- 核心头文件：lyscanopen.h
- 底层适配头文件：lyscan.h（提供 CAN/CANFD 基础帧结构体定义）

2.3 硬件适配要求

用户无需修改协议库源码，仅需根据自身硬件平台，实现底层 CAN/CANFD 收发回调函数，即可完成硬件适配。

3. 整体调用流程

本库有严格的接口调用时序，必须遵循固定流程开发，否则会出现初始化失败、接口报错、通信异常等问题。

标准调用流程：LCO_Init 初始化库实例 → 配置回调与参数 → 执行 CANopen 业务操作 → LCO_Destroy 销毁实例

3.1 详细流程步骤

1. 实现用户底层回调：完成 CAN 收发、CANFD 收发、报文解析通知回调函数的自定义实现；无 CANFD 场景可将 CANFD 回调置空
2. 初始化参数配置：填充 LCO_INIT_PARAM 初始化结构体，绑定所有有效回调函数与用户私有数据
3. 创建库实例：调用 LCO_Init 获取 CANopen 库操作句柄
4. 参数配置（可选）：设置 SDO 超时时间、心跳周期等全局参数
5. 业务逻辑开发：调用 NMT、SDO、PDO、CiA402 等接口完成设备通信与控制



6. 报文输入解析：底层收到 CAN/CANFD 报文后，调用 `LCO_InputFrame/LCO_InputFdFrame` 送入协议库解析
7. 资源释放：程序退出、设备断电、任务销毁前，调用 `LCO_Destroy` 释放库资源

4. 用户必须实现的回调接口

库本身无底层硬件驱动能力，所有 CAN 报文收发、解析结果通知均依赖用户层回调实现，共 3 个必填接口、2 个保留接口。

4.1 CAN2.0 接收回调（必填）

函数原型：`typedef UINT (*LCO_CAN_REC_CB)(PLCAN_CAN_OBJ pFrame, UINT maxNum, void *priv);`

功能说明：协议库主动调用该接口读取底层接收的 CAN2.0 报文

参数说明：

- `pFrame`：CAN 帧数据缓冲区指针，用于存储读取到的报文
- `maxNum`：缓冲区最大可存储报文数量
- `priv`：用户初始化时传入的私有数据指针

返回值：实际读取到的有效 CAN 报文数量

4.2 CAN2.0 发送回调（必填）

函数原型：`typedef LCO_STATUS (*LCO_CAN_SEND_CB)(PLCAN_CAN_OBJ pFrame, void *priv);`

功能说明：协议库需要发送 CAN2.0 报文时，调用该底层发送接口

参数说明：

- `pFrame`：待发送的 CAN 帧数据指针
- `priv`：用户私有数据指针

返回值：`LCO_OK` 发送成功，其他值为发送失败



4.3 解析结果通知回调（必填）

函数原型: `typedef void (*LCO_NOTIFY_CB)(LCO_RESULT *result, void *priv);`

功能说明: 协议库完成报文解析后, 通过该回调将解析结果推送至用户层

参数说明:

- `result`: CANopen 报文解析结果结构体, 包含报文类型、节点 ID、协议数据
- `priv`: 用户私有数据指针

补充说明: 无需解析功能时, 可将该参数设置为 `NULL`

4.4 CANFD 收发回调（保留）

`LCO_CANFD_REC_CB`、`LCO_CANFD_SEND_CB` 为预留接口, 当前版本无需实现, 初始化时直接设置为 `NULL` 即可。

5. 核心数据结构与枚举说明

5.1 库状态码（`LCO_STATUS`）

所有对外 API 统一返回该状态码, 用于判断接口执行结果, 是异常处理的核心依据。

枚举值	数值	状态说明
<code>LCO_OK</code>	0	接口执行成功
<code>LCO_ERR_PARAM</code>	1	输入参数非法、为空或超出范围
<code>LCO_ERR_TIMEOUT</code>	2	通信超时, 未收到设备响应报文
<code>LCO_ERR_BUSY</code>	3	资源繁忙, 如 SDO 正在进



		行传输
LCO_ERR_ABORT	4	远程节点返回中止报文，通信中断
LCO_ERR_NOT_INIT	5	库未初始化或句柄无效
LCO_ERR_CAN_SEND	6	底层 CAN 发送失败
LCO_ERR_CAN_RECV	7	底层 CAN 接收异常

5.2 NMT 控制命令（LCO_NMT_CMD）

遵循 CiA301 标准，用于控制 CANopen 节点工作状态切换。

枚举值	数值	功能说明
LCO_NMT_OPERATIONAL	0x01	节点切换至运行状态，正常响应所有协议报文
LCO_NMT_STOP	0x02	节点切换至停止状态，仅响应 NMT 命令
LCO_NMT_PRE_OP	0x80	节点切换至预操作状态，支持对象字典配置
LCO_NMT_RESET_NODE	0x81	复位节点（应用层+通信层全部重置）
LCO_NMT_RESET_COMM	0x82	仅复位通信层状态机，保留应用层数据

5.3 CiA402 伺服操作模式（LCO_402_OP_MODE）

适配工业伺服驱动器标准运动控制模式，为伺服控制接口的核心参数。



枚举值	数值	控制模式
LCO_402_MODE_PROFILE_POS	1	轮廓位置模式 (PP)
LCO_402_MODE_PROFILE_VEL	3	轮廓速度模式 (PV)
LCO_402_MODE_TORQUE	4	转矩控制模式 (TQ)
LCO_402_MODE_HOMING	6	电机回零模式 (HM)
LCO_402_MODE_CSP	8	同步位置模式
LCO_402_MODE_CSV	9	同步速度模式
LCO_402_MODE_CST	10	同步转矩模式

5.4 初始化参数结构体 (LCO_INIT_PARAM)

库初始化唯一入参，用于绑定底层驱动回调与用户私有数据。

- canRecvCb: CAN2.0 接收回调 (必填)
- canSendCb: CAN2.0 发送回调 (必填)
- canfdRecvCb: CANFD 接收回调 (置 NULL)
- canfdSendCb: CANFD 发送回调 (置 NULL)
- notifyCb: 报文解析通知回调 (必填)
- priv: 用户私有数据指针，全局透传至所有回调

5.5 解析结果结构体 (LCO_RESULT)



存储所有 CANopen 报文解析结果，通过通知回调推送用户层，包含 NMT、SDO、PDO、EMCY、时间戳等全类型报文数据，同时保留原始 CAN/CANFD 帧数据，满足高级调试需求。

6. 核心 API 接口详细说明

6.1 初始化与销毁接口

6.1.1 LCO_Init

功能：初始化 CANopen 协议库，创建实例句柄

原型：`LCO_HANDLE LCO_Init(const LCO_INIT_PARAM* pParam);`

参数：pParam-初始化参数结构体指针，非空必填

返回值：成功返回有效库实例句柄，失败返回 NULL

注意：所有业务接口调用前，必须先执行本函数

6.1.2 LCO_Destroy

功能：销毁库实例，释放内部资源

原型：`LCO_STATUS LCO_Destroy(LCO_HANDLE handle);`

参数：handle-有效库实例句柄

返回值：LCO_OK 销毁成功，其他为失败

6.1.3 LCO_InputFrame / LCO_InputFdFrame

功能：将底层接收的 CAN/CANFD 报文送入协议库解析

使用场景：底层 CAN 接收中断或轮询任务中，收到报文后主动调用

6.2 NMT 网络管理接口



6.2.1 LCO_NMT_SendCommand

功能：向指定节点发送 NMT 状态控制命令，切换节点工作状态

参数：节点 ID（0 为广播）、NMT 控制命令

6.2.2 LCO_NMT_SetHeartbeatPeriod

功能：通过 SDO 配置从站节点心跳周期，0 为关闭心跳功能

6.2.3 LCO_NMT_SendHeartbeat

功能：主站主动上报自身心跳状态，用于总线节点监测

6.3 SDO 服务接口

6.3.1 LCO_SetSDOTimeout

功能：设置 SDO 通信超时时间，默认 1000ms

6.3.2 LCO_ReadObject

功能：读取远程节点对象字典数据，自动处理单帧/分段传输

核心参数：对象字典索引、子索引、数据接收缓冲区、实际读取长度输出

6.3.3 LCO_WriteObject

功能：向远程节点对象字典写入数据，支持分段传输

6.4 总线辅助功能接口

- **LCO_SendSYNC**：发送总线同步广播报文，用于同步 PDO 数据传输
- **LCO_SendEMCY**：发送设备紧急故障报文，上报设备异常信息
- **LCO_SendTimestamp**：发送时间戳广播报文，实现总线设备时间同步



- **LCO_SendNodeGuardRequest:** 发送节点守护查询帧，监测从站在线状态

6.5 CiA402 伺服专用控制接口

本系列接口为伺服电机运动控制专用封装，直接对接标准 CiA402 对象字典，无需用户手动填写字典索引，接口极简易用。

6.5.1 基础状态控制

- **LCO_402_SetControlWord:** 设置伺服控制字，控制电机使能、运行、停止
- **LCO_402_GetStatusWord:** 读取伺服状态字，判断故障、就绪、运行状态
- **LCO_402_SetOpMode / LCO_402_GetOpMode:** 设置/读取伺服当前控制模式

6.5.2 运动参数控制

- **LCO_402_SetTargetPos / LCO_402_GetActualPos:** 设置目标位置、读取实际位置
- **LCO_402_SetTargetVel / LCO_402_GetActualVel:** 设置目标速度、读取实际速度
- **LCO_402_SetTargetTorque / LCO_402_GetActualTorque:** 设置目标转矩、读取实际转矩

6.5.3 专用功能

- **LCO_402_SetHomingMode:** 配置电机回零模式
- **LCO_402_FaultReset:** 伺服故障状态复位
- **LCO_402_PowerOn:** 一键伺服上电，自动完成 Pre-Op 到 Op 状态切换

7. 快速使用示例

7.1 库初始化基础示例



```
c
// 1. 定义全局句柄与私有数据
LCO_HANDLE g_lcoHandle = NULL;

// 2. 实现底层回调（简化示例）
UINT LCO_CanRecvCallback(PLCAN_CAN_OBJ pFrame, UINT maxNum, void
*priv)
{
    // 用户底层 CAN 接收逻辑，返回有效报文数
    return 0;
}

LCO_STATUS LCO_CanSendCallback(PLCAN_CAN_OBJ pFrame, void *priv)
{
    // 用户底层 CAN 发送逻辑
    return LCO_OK;
}

void LCO_NotifyCallback(LCO_RESULT *result, void *priv)
{
    // 报文解析结果处理逻辑
    if(result == NULL) return;
}

// 3. 初始化库
void LCO_Module_Init(void)
{
    LCO_INIT_PARAM initParam = {0};

    // 绑定必填回调
    initParam.canRecvCb = LCO_CanRecvCallback;
    initParam.canSendCb = LCO_CanSendCallback;
    initParam.notifyCb = LCO_NotifyCallback;
```



```
// 保留接口置空
initParam.canfdRecvCb = NULL;
initParam.canfdSendCb = NULL;
initParam.priv = NULL;

// 创建库实例
g_lcoHandle = LCO_Init(&initParam);
}
```

7.2 伺服电机一键启动示例

```
c
// 伺服节点 ID 为 1，上电启动
LCO_STATUS ret = LCO_402_PowerOn(g_lcoHandle, 1);
if(ret == LCO_OK)
{
    // 伺服启动成功
}
```

8. 异常处理与开发注意事项

8.1 通用注意事项

- 必须严格遵循 **初始化→业务操作→销毁** 调用时序，禁止未初始化直接调用接口
- 所有接口入参必须合法，句柄为空、节点 ID 超出 0-127 范围会直接返回参数错误
- SDO 读写、伺服控制类接口为阻塞式通信，需处理超时异常
- 底层 CAN 收发回调必须保证线程安全、中断安全，避免报文丢失或错乱



8.2 常见异常排查

- **LCO_ERR_NOT_INIT**: 库未初始化、句柄未创建或已销毁
- **LCO_ERR_TIMEOUT**: 从站无响应、总线断线、节点 ID 错误、设备未上线
- **LCO_ERR_CAN_SEND**: 底层 CAN 驱动异常、总线挂载失败
- **LCO_ERR_ABORT**: 从站对象字典权限不足、参数非法、设备状态不允许操作

8.3 资源规范

多任务场景下，支持多线程同时调用 SDO 读写、NMT 控制接口。

9. 版本更新说明

V1.1 版本更新要点:

- 完整支持 CAN2.0 与 CANFD 双协议，扩展 PDO 最大数据长度
- 完善 CiA402 全系列伺服控制接口，覆盖主流运动控制模式
- 优化 SDO 分段传输逻辑，提升大数据读写稳定性
- 标准化错误码与状态机，提升异常适配性
- 完全剥离硬件依赖，实现全平台通用

10. 技术支持

开发厂商: 长沙六叶树教育科技有限公司

官方网站: www.liuyeshu.cn

技术服务: 如需技术答疑、定制化适配、功能拓展，可通过官网联系官方技术团队。