



NKCAN SDK (Windows)

二次开发用户手册

NODKA

1.0.2

2025-12-5

发布列表:

名称/ 组织	公司
EA	NODKA

评审/批准:

	姓名 / 职能 / 公司	签名
作者:	EA	
评审:		

目录

1 修改历史	3
2 概述	4
2.1 关于文档	4
3 SDK 安装包说明	5
4 函数库 API 说明	6
4.1 NKCAN API	6
4.1.1 NKCAN API 函数列表	6
4.1.2 CAN 设备类型定义	7
4.1.3 错误码定义	7
4.1.4 NKCAN API 数据结构详解	8
4.1.5 NKCAN API 使用流程	12
4.1.6 NKCAN API 功能详解	13
5 FAQ	20

1 修改历史

版本	修改日期	修改人	描述
1.0.0	2025-03-21	EA	初始版本
1.0.1	2025-04-17	EA	添加对于错误结构体的说明和发送的注意事项说明。
1.0.2	2024-12-05	EA	修复了初始化卡住超时的问题，修改了NKCAN_Receive和NKCAN_GetReceiveCount的函数说明。

2 概述

2.1 关于文档

本手册旨在帮助用户对 MINIPCIE_CAN 卡的 Windows 下的软件库函数的调用，参数的设置等。
当前 SDK 版本仅适用于硬件版本 0x1 及以上的。

3 SDK 安装包说明

SDK 包可以通过安装“NKCAN_SDK_Driver_Setup_x86.exe”文件来解压。默认的安装路径是“C:\NODKA”，在驱动程序安装成功后，将找到一个名为 NKCAN_SDK 的文件夹。下面是 NKCAN_SDK 文件夹中的内容列表。

 Demo

 Include

 Lib

 Manual

 vc_redist

 Histroy

- Demo: 二次开发示例 demo。
- Include: 包含 API 声明的头文件。
- Lib: 库文件。
- Manual: 手册。
- vc_redist: VC++ 运行时库，如果你的电脑还没有安装，你必须手动安装它们。

4 函数库 API 说明

此章节用于对函数库的 API 的功能说明以及使用方法。

4.1 NKCAN API

4.1.1 NKCAN API 函数列表

函数名称	函数功能
库资源函数	
NKCAN_LibraryInit	初始化 NKCAN 库中 CAN 设备资源
NKCAN_LibraryDeinit	释放 NKCAN 库中 CAN 设备资源
设备函数	
NKCAN_OpenDevice	打开 CAN 设备
NKCAN_CloseDevice	关闭 CAN 设备
NKCAN_ReadBoardInfo	读取 CAN 的设备信息
设置函数	
NKCAN_InitCAN	初始化某路 CAN 通道
NKCAN_OpenCAN	打开某路 CAN 通道
NKCAN_CloseCAN	关闭某路 CAN 通道
收发函数	
NKCAN_Transmit	发送某路 CAN 通道的数据帧
NKCAN_GetReceiveCount	读取某路 CAN 接收通道中缓存数据帧的个数
NKCAN_Receive	接收某路 CAN 通道的数据帧
辅助函数	
NKCAN_ClearBuffer	清除某路 CAN 通道的缓存数据帧
NKCAN_ReadErrorInfo	获取某路 CAN 通道的错误信息

4.1.2 CAN 设备类型定义

CAN 设备类型	库中宏定义	数值	说明
MINPCIE_CAN	MINIPICE_MCU_USB	1	该设备有两路 CAN 通道，设备号从 0 开始，通道号从 0 开始。

4.1.3 错误码定义

错误代号	数值	错误说明
通用错误码		
NKCAN_NOERROR	0x0000	无错误
NKCAN_EINVAL	0x0016	非法参数
设备		
NKCAN_DEVICE_OPENED	0x0001	设备已经打开
NKCAN_DEVICE_OPENERR	0x0002	打开设备错误
NKCAN_DEVICE_INIT_ERROR	0x0003	初始化设备失败
NKCAN_DEVICE_NOTOPEN	0x0004	设备没有打开
NKCAN_DEVICE_NOTEXIST	0x0005	设备不存在
NKCAN_INTERFACE_NOTEXIST	0x0006	接口不存在
NKCAN_DEVICE_CLOSED	0x0007	设备已经关闭
NKCAN_DEVIDE_TIMEOUT	0x0008	设备超时
库		
NKCAN_LIBRARY_INITERR	0x1001	库初始化失败
NKCAN_LIBRARY_NOTINIT	0x1002	库未初始化
NKCAN_LIBRARY_INITED	0x1003	库已初始化
NKCAN_LIBRARY_DEINITED	0x1004	库已释放资源
NKCAN_LIBRARY_THREADERR	0x1005	线程创建失败
CAN		
NKCAN_CAN_SAMPLEPOINT_ILLEGAL	0x2001	采样率不存在
NKCAN_CAN_STARTED	0x2002	CAN 通道已打开
NKCAN_CAN_NOTSTARTED	0x2003	CAN 通道未打开
NKCAN_CAN_BUFFEREMPTY	0x2004	缓存区为空

4.1.4 NKCAN API 数据结构详解

4.1.4.1 NKCAN_BOARD_INFO

■ 功能描述

NKCAN_BOARD_INFO 结构体包含了 CAN 卡的设备信息，结构体将在 NKCAN_ReadBoardInfo 函数中被填充。

■ 结构体原型

```
typedef struct _NKCAN_BOARD_INFO {  
    unsigned short hw_Version;  
    unsigned short fw_Version;  
    unsigned short dr_Version;  
    unsigned short in_Version;  
    unsigned short irq_Num;  
    unsigned char can_Num;  
    char str_Serial_Num[20];  
    char str_hw_Type[40];  
    unsigned short Reserved[4];  
} NKCAN_BOARD_INFO, *PNKCAN_BOARD_INFO;
```

■ 成员变量说明

- hw_Version: 硬件版本号，用 16 进制表示。
- fw_Version: 固件版本号，用 16 进制表示。
- dr_Version: 驱动版本号，用 16 进制表示。
- in_Version: 接口库版本号，用 16 进制表示。
- irq_Num: 中断号。
- can_Num: CAN 通道数。
- str_Serial_Num: 序列号，当前只用 0 和 1 元素，后面保留。
- str_hw_Type: 硬件类型，当前只用 0 和 1 元素，后面保留。

4.1.4.2 NKCAN_INIT_CONFIG

■ 功能描述

NKCAN_INIT_CONFIG 结构体定义了 CAN 通道初始化的参数配置值，结构体将通过 NKCAN_InitCAN 函数来初始化 CAN 通道。

■ 结构体原型

```
typedef struct _NKCAN_INIT_CONFIG {
    unsigned int AccCode;
    unsigned int AccMask;
    unsigned int BaudRate;
    unsigned char Filter;
    unsigned char ExtFrame;
    double SamplePoint;
    unsigned char reserved[2];
}NKCAN_INIT_CONFIG, *PNKCAN_INIT_CONFIG;
```

■ 成员变量说明

- AccCode：对经过屏蔽码 AccMask0 过滤为“有关位”进行匹配，全部匹配，则接收，否则不接收。
- AccMask：对接收的 CAN 帧的 ID 进行过滤，对应位为 1 的是“有关位”，对应位为 0 的是“无关位”。屏蔽码设置为 0，则表示全部接收。
- BaudRate：波特率，波特率和采样点的推荐设置见下表。
- Filter：滤波方式。0：未滤波，1：单滤波。
- ExtFrame:0：标准帧，1：扩展帧。
- SamplePoint：采样点，波特率和采样点的推荐设置见下表。
- reserved：保留

CAN 波特率	采样点
10Kbps	0.875
20Kbps	
50Kbps	
100Kbps	
125Kbps	
250Kbps	
500Kbps	
800Kbps	0.8
1000Kbps	0.75

4.1.4.3 NKCAN_CAN_OBJ

■ 功能描述

NKCAN_CAN_OBJ 结构体是 CAN 数据帧的结构体，一个结构体变量表示一帧数据，在发送函数 NKCAN_Transmit 用来发送和接收函数 NKCAN_Receive 中用来接收。

■ 结构体原型

```
typedef struct _NKCAN_CAN_OBJ {  
    unsigned int TimeStamp[2];    //  
    unsigned char RemoteFlag;    //  
    unsigned char ExternFlag;    //是否是扩展帧,用于接收和发送  
    unsigned char LoopFlag;      //是否是回环帧，用于发送。  
    unsigned char DataLen;       //数据长度,用于接收和发送  
    unsigned int ID;             //  
    unsigned char Data[8];       //数据,用于接收和发送  
}NKCAN_CAN_OBJ, *PNKCAN_CAN_OBJ;
```

■ 成员变量说明

- TimeStamp：时间戳，用于接收。
- RemoteFlag：远程帧标识，0：数据帧，1：远程帧，用于接收和发送
- ExternFlag：扩展帧标识，0：标准帧（11 位 ID），1：扩展帧（29 位 ID），用于接收和发送
- LoopFlag：0：数据帧 1：回环帧，用于发送。
- DataLen：数据长度，即 CAN 帧 Data 中有效字节的长度，最大为 8。
- ID：普通帧 11 位，扩展帧 29 位,用于接收和发送。
- Data[8]：CAN 帧的数据。

4.1.4.4 NKCAN_ERR_INFO

■ 功能描述

NKCAN_ERR_INFO 结构体包含了 CAN 卡的错误信息，结构体将在 NKCAN_ReadErrorInfo 函数中被填充。

■ 结构体原型

```
typedef struct _NKCAN_ERR_INFO {
    unsigned int ErrCode;
    unsigned char Passive_ErrData[3];
    unsigned char ArLost_ErrData;
} NKCAN_ERR_INFO, *PNKCAN_ERR_INFO;
```

■ 成员变量说明

➤ ErrCode: 暂未使用。

➤ Passive_ErrData: 0 元素: 见下表, 1 元素: 发送错误计数, 2 元素: 接收错误计数。

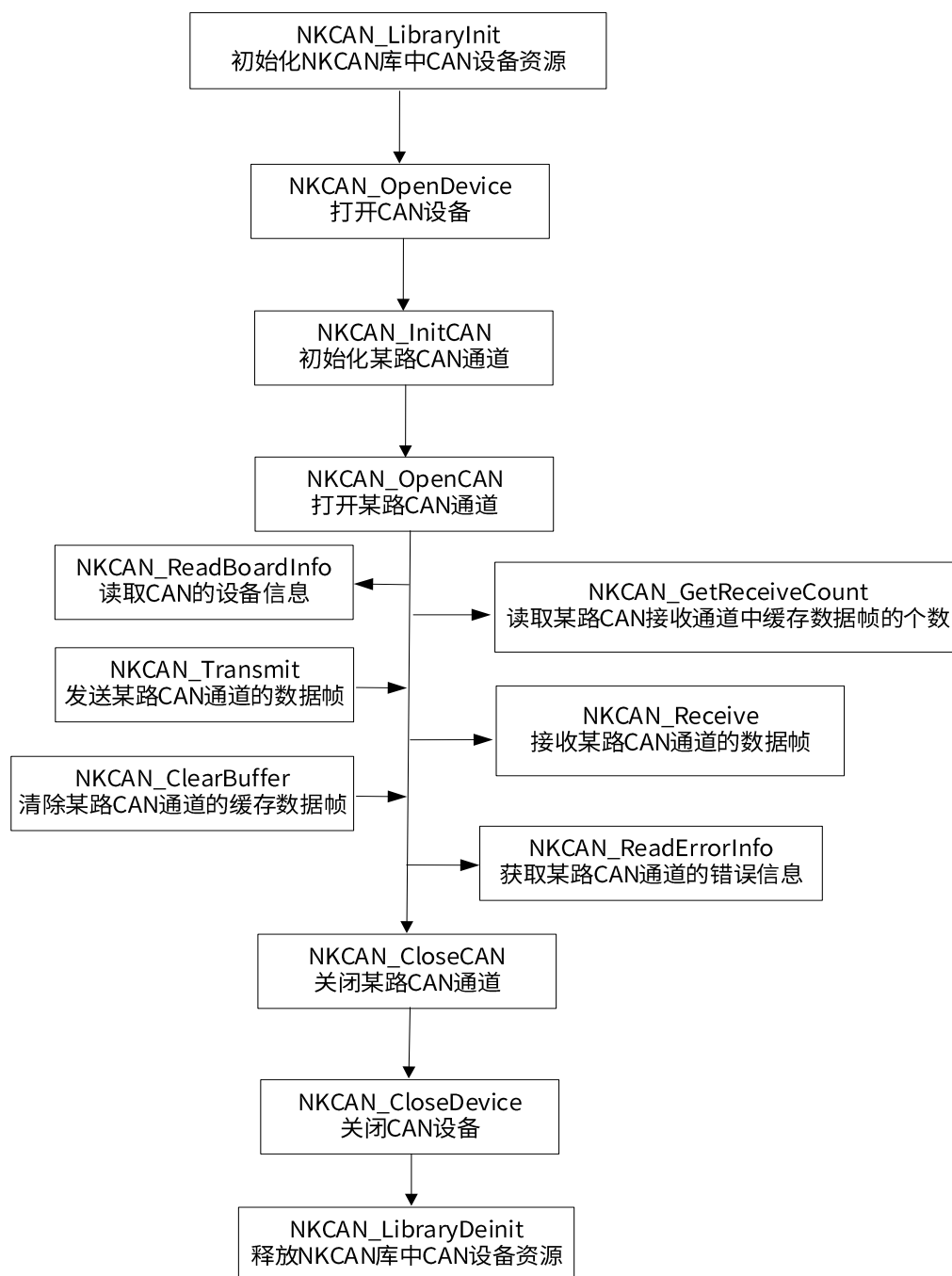
对于发送错误计数, 建议用在**判断**是否可以发送, 如果读取的数据一直是大于十进制 128 数值, 那么就不能发送数据, 如果是读取的数值在减小或者是 0 或是 1, 那么就可以发送。如果在读取 128 的时候一直发送, 那么有可能在下次打开设备时卡住很久或者报错。

➤ ArLost_ErrData: 暂未使用。

x 表示未使用。

Bit 位	7	6	5	4	3	2	1	0	说明
	x	x	x	x	x	0	0	0	没有错误
	x	x	x	x	x	0	0	1	位填充错误
	x	x	x	x	x	0	1	0	格式(FORM)错误
	x	x	x	x	x	0	1	1	确认(ACK)错误
	x	x	x	x	x	1	0	0	隐性错误
	x	x	x	x	x	1	0	1	显性错误
	x	x	x	x	x	1	1	0	CRC 错误
	x	x	x	x	x	1	1	1	由软件设置, 暂未设置

4.1.5 NKCAN API 使用流程



4.1.6 NKCAN API 功能详解

4.1.6.1 库资源函数

4.1.6.1.1 NKCAN_LibraryInit

■ 功能描述

初始化 NKCAN 库中 CAN 设备资源。

■ 函数名称

```
int NKCAN_LibraryInit(unsigned int DeviceType);
```

■ 参数说明

➤ 输入参数

DeviceType: CAN 设备类型, 见设备类型定义。

➤ 输出参数

无。

■ 返回值

成功返回 0, 其他返回错误代号。

■ 备注说明

在调用其他的函数之前需调用此函数, 而且此函数仅需调用一次。

4.1.6.1.2 NKCAN_LibraryDeinit

■ 功能描述

释放 NKCAN 库中 CAN 设备资源。

■ 函数名称

```
int NKCAN_LibraryDeinit(unsigned int DeviceType)
```

■ 参数说明

➤ 输入参数

DeviceType: CAN 设备类型, 见设备类型定义。

➤ 输出参数

无

■ 返回值

成功返回 0, 其他返回错误代号。

■ 备注说明

关闭应用程序时, 调用该函数释放库资源。

4.1.6.2 设备函数

4.1.6.2.1 NKCAN_OpenDevice

■ 功能描述

打开 CAN 设备。

■ 函数名称

int NKCAN_OpenDevice(unsigned int DeviceType, unsigned int DeviceInd, unsigned int Reserved)

■ 参数说明

➤ 输入参数

DeviceType: CAN 设备类型, 见设备类型定义。

DeviceInd: CAN 设备号, 从 0 开始。

Reserved: 保留, 默认传入数值 0。

➤ 输出参数

无。

■ 返回值

成功返回 0, 其他返回错误代号。

■ 备注说明

此函数必须在初始化之后才可以使用。

4.1.6.2.2 NKCAN_CloseDevice

■ 功能描述

关闭 CAN 设备。

■ 函数名称

int NKCAN_CloseDevice(unsigned int DeviceType, unsigned int DeviceInd)

■ 参数说明

➤ 输入参数

DeviceType: CAN 设备类型, 见设备类型定义。

DeviceInd: CAN 设备号, 从 0 开始。

➤ 输出参数

无。

■ 返回值

成功返回 0, 其他返回错误代号。

■ 备注说明

此函数必须在初始化之后才可以使用。

4.1.6.2.3 NKCAN_ReadBoardInfo

■ 功能描述

读取 CAN 的设备信息。

■ 函数名称

int NKCAN_ReadBoardInfo(unsigned int DeviceType, unsigned int DeviceInd,
PNKCAN_BOARD_INFO pInfo)

■ 参数说明

➤ 输入参数

DeviceType: CAN 设备类型, 见设备类型定义。

DeviceInd: CAN 设备号, 从 0 开始。

pInfo: 指向 NKCAN_BOARD_INFO 的指针, 详见 NKCAN_BOARD_INFO 说明。

➤ 输出参数

无。

■ 返回值

成功返回 0, 其他返回错误代号。

■ 备注说明

此函数必须在初始化和打开 CAN 设备之后才可以使用。

4.1.6.3 设置函数

4.1.6.3.1 NKCAN_InitCAN

■ 功能描述

初始化某路 CAN 通道。

■ 函数名称

int NKCAN_InitCAN(unsigned int DeviceType, unsigned int DeviceInd, unsigned int CANInd,
PNKCAN_INIT_CONFIG pInitConfig)

■ 参数说明

➤ 输入参数

DeviceType: CAN 设备类型, 见设备类型定义。

DeviceInd: CAN 设备号, 从 0 开始。

CANInd: CAN 通道号, 从 0 开始, 具体通道数查看设备类型定义。

pInitConfig: 指向 NKCAN_INIT_CONFIG 的指针, 详见 NKCAN_INIT_CONFIG 说明。

➤ 输出参数

无。

■ 返回值

成功返回 0, 其他返回错误代号。

■ 备注说明

此函数必须在初始化和打开 CAN 设备之后才可以使用。

4.1.6.3.2 NKCAN_OpenCAN

- **功能描述**

打开某路 CAN 通道。

- **函数名称**

int NKCAN_OpenCAN(unsigned int DeviceType, unsigned int DeviceInd, unsigned int CANInd)

- **参数说明**

- **输入参数**

DeviceType: CAN 设备类型, 见设备类型定义。

DeviceInd: CAN 设备号, 从 0 开始。

CANInd: CAN 通道号, 从 0 开始, 具体通道数查看设备类型定义。

- **输出参数**

无。

- **返回值**

成功返回 0, 其他返回错误代号。

- **备注说明**

此函数必须在初始化和打开 CAN 设备之后才可以使用。

4.1.6.3.3 NKCAN_CloseCAN

- **功能描述**

关闭某路 CAN 通道。

- **函数名称**

int NKCAN_CloseCAN(unsigned int DeviceType, unsigned int DeviceInd, unsigned int CANInd)

- **参数说明**

- **输入参数**

DeviceType: CAN 设备类型, 见设备类型定义。

DeviceInd: CAN 设备号, 从 0 开始。

CANInd: CAN 通道号, 从 0 开始, 具体通道数查看设备类型定义。

- **输出参数**

无。

- **返回值**

成功返回 0, 其他返回错误代号。

- **备注说明**

此函数必须在初始化和打开 CAN 设备之后才可以使用。

4.1.6.4 收发函数

4.1.6.4.1 NKCAN_Transmit

■ 功能描述

发送某路 CAN 通道的数据帧。

■ 函数名称

int NKCAN_Transmit(unsigned int DeviceType, unsigned int DeviceInd, unsigned int CANInd, PNKCAN_CAN_OBJ pSend, unsigned int sendnum, unsigned int *realnum);

■ 参数说明

➤ 输入参数

DeviceType: CAN 设备类型, 见设备类型定义。

DeviceInd: CAN 设备号, 从 0 开始。

CANInd: CAN 通道号, 从 0 开始, 具体通道数查看设备类型定义。

pSend: 指向 NKCAN_CAN_OBJ 的指针, 详见 NKCAN_CAN_OBJ 说明。

Sendnum: 要发送的帧结构体数组的长度 (发送的帧数量)。此最大值为 10。

➤ 输出参数

realnum: 实际发送数量。

■ 返回值

成功返回 0, 其他返回错误代号。

■ 备注说明

此函数必须在初始化和打开 CAN 设备和通道之后才可以使用。

4.1.6.4.2 NKCAN_Receive

■ 功能描述

接收某路 CAN 通道的数据帧。

■ 函数名称

int NKCAN_Receive(unsigned int DeviceType, unsigned int DeviceInd, unsigned int CANInd, PNKCAN_CAN_OBJ pReceive, unsigned int receivenum, unsigned int *realnum);

■ 参数说明

➤ 输入参数

DeviceType: CAN 设备类型, 见设备类型定义。

DeviceInd: CAN 设备号, 从 0 开始。

CANInd: CAN 通道号, 从 0 开始, 具体通道数查看设备类型定义。

pReceive: 指向 NKCAN_CAN_OBJ 的指针, 详见 NKCAN_CAN_OBJ 说明。

receivenum: 要接收的帧数量。建议设置为 1。

➤ 输出参数

realnum: 实际接收数量。

■ 返回值

成功返回 0，其他返回错误代号。

■ 备注说明

此函数必须在初始化和打开 CAN 设备和通道之后才可以使用。

4.1.6.5 辅助函数

4.1.6.5.1 NKCAN_GetReceiveCount

■ 功能描述

读取某路 CAN 接收通道中缓存数据帧的个数。

■ 函数名称

int NKCAN_GetReceiveCount(unsigned int DeviceType, unsigned int DeviceInd, unsigned int CANInd, unsigned int *pReceviceCount)

■ 参数说明

➤ 输入参数

DeviceType: CAN 设备类型，见设备类型定义。

DeviceInd: CAN 设备号，从 0 开始。

CANInd: CAN 通道号，从 0 开始，具体通道数查看设备类型定义。

➤ 输出参数

pReceviceCount: CAN 通道中缓存的数据帧。

■ 返回值

成功返回 0，其他返回错误代号。

■ 备注说明

此函数必须在初始化和打开 CAN 设备和通道之后才可以使用。

4.1.6.5.2 NKCAN_ClearBuffer

■ 功能描述

清除某路 CAN 通道的缓存数据帧。

■ 函数名称

int NKCAN_ClearBuffer(unsigned int DeviceType, unsigned int DeviceInd, unsigned int CANInd)

■ 参数说明

➤ 输入参数

DeviceType: CAN 设备类型，见设备类型定义。

DeviceInd: CAN 设备号，从 0 开始。

CANInd: CAN 通道号，从 0 开始，具体通道数查看设备类型定义。

➤ 输出参数

无。

■ 返回值

成功返回 0，其他返回错误代号。

■ 备注说明

此函数必须在初始化和打开 CAN 设备和通道之后才可以使用。

4.1.6.5.3 NKCAN_ReadErrorInfo

■ 功能描述

获取某路 CAN 通道的错误信息。

■ 函数名称

int NKCAN_ReadErrorInfo(unsigned int DeviceType, unsigned int DeviceInd, unsigned int CANInd, PNKCAN_ERR_INFO pErrInfo)

■ 参数说明

➤ 输入参数

DeviceType: CAN 设备类型，见设备类型定义。

DeviceInd: CAN 设备号，从 0 开始。

CANInd: CAN 通道号，从 0 开始，具体通道数查看设备类型定义。

pErrInfo: 指向 NKCAN_ERR_INFO 的指针，详见 NKCAN_ERR_INFO 说明

➤ 输出参数

无。

■ 返回值

成功返回 0，其他返回错误代号。

■ 备注说明

此函数必须在初始化和打开 CAN 设备和通道之后才可以使用。

5 FAQ