

HK-CAN30B

非智能隔离型 CAN 总线通讯板

目 录

| | |
|----------------------------------|-----------|
| 一、概述..... | 1 |
| 二、性能及技术指标..... | 1 |
| 2.1 性能..... | 1 |
| 2.2 技术指标..... | 2 |
| 2.3 应用..... | 2 |
| 2.4 物理尺寸和工作环境条件..... | 2 |
| 三、工作原理..... | 2 |
| 3.1 工作原理概述..... | 2 |
| 3.2.1 PCI 总线接口部分..... | 2 |
| 3.2.2 CAN 通讯部分..... | 2 |
| 四、主要元件位置图、信号输入/输出插座和跳线选择定义..... | 3 |
| 4.1 主要元件位置图..... | 3 |
| 4.2 信号输入/输出插座定义..... | 3 |
| 4.2.1 HK-CAN30B 信号输入/输出插座定义..... | 3 |
| 4.3.1 通讯口的设置..... | 4 |
| 五、硬件安装、拆除方法..... | 4 |
| 5.1 安装步骤..... | 5 |
| 5.2 拆除步骤..... | 5 |
| 六、安装驱动软件..... | 5 |
| 6.1 Windows98、2000 下的安装..... | 5 |
| 6.2 Window NT 下的安装..... | 6 |
| 七、应用注意事项..... | 6 |
| 7.1 注意事项..... | 6 |
| 7.2HK-CAN30B 板不能通讯..... | 6 |
| 八、驱动软件..... | 6 |
| 8.1 结构体定义..... | 错误！未定义书签。 |
| 8.1.1CAN 帧数据结构：..... | 错误！未定义书签。 |
| 8.1.2 设备控制参数结构：..... | 错误！未定义书签。 |

| | |
|-------------------------------------|-----------|
| 8. 2 驱动函数说明 | 错误！未定义书签。 |
| 8. 2. 1 HKCanOpen () | 错误！未定义书签。 |
| 8. 2. 2 HKCanClose () | 错误！未定义书签。 |
| 8. 2. 3 HKCanGetBDVersion () | 错误！未定义书签。 |
| 8. 2. 4 HKCanInitState () | 错误！未定义书签。 |
| 8. 2. 5 HKCanSendFrame () | 错误！未定义书签。 |
| 8. 2. 6 HKCanReadFrame () | 错误！未定义书签。 |
| 8. 2. 7 HKCanReadFrameEx () | 错误！未定义书签。 |
| 8. 2. 8 HKCanGetLastError () | 错误！未定义书签。 |
| 8. 3 CAN 信息包格式说明 | 错误！未定义书签。 |

HK-CAN30B

PCI 总线 CAN 非智能隔离型通讯板

一、概述

HK-CAN30B PCI 总线 CAN 非智能隔离型通讯板，是一种将 CAN (Control Area Network) 通讯协议与 PC 机 PCI 总线标准相连接的非智能 CAN 插卡，非智能 CAN 卡可以充分满足 CAN 网络高实时性的要求，通过该卡可对工业现场具有 CAN 通讯接口的仪表和控制设备进行监控。

CAN 是一种全数字化的通讯网络，在国外，已经安装了大量具有 CAN 通讯接口的传感器、执行器、马达等工业设备；在国内，许多具有 CAN 接口的工控设备也逐渐得到广大用户的认可。CAN 最大的优点在于它的实时性很高，由于它采用位仲裁方式进行网络分配，因此可以最大限度的保证系统对紧急事件的响应；CAN 另外一个优点是具有很高的可靠性，它有五种方法判断和纠正数据在传输中可能发生的错误，所以 CAN 可以应用在对可靠性要求较高的系统中。

二、性能及技术指标

华控公司 HK-CAN30B 非智能隔离型 CAN 总线通讯板，板上有 DC/DC，CAN 通讯无需外供电，适用于通用场合。

2.1 性能

- 32 位 33M PCI 数据总线, PCI2.1 兼容, 既插既用。
- PCI 总线数据宽度: 32 位
- PCI 总线地址/中断设定: 自动分配
- 符合 CAN 协议 2.0A 规范
- CAN 网络采用 DB-9 针式连接器。
- CAN 网络接口控制器: Philips SJA1000T 或兼容芯片
- CAN 网络收发器: Philips 82C250 或兼容芯片

2.2 技术指标

- CAN 网络通讯最高速率：1Mbit/s
- 隔离耐压：500VRMS
- 驱动程序：适用于 Windows98/NT/2000

2.3 应用

- 各种集散式控制系统
- 要求抗干扰能力强的通讯网络中

2.4 物理尺寸和工作环境条件

- 外形尺寸：117mm×98mm
- 工作温度范围：0℃～+50℃
- 贮藏温度范围：-25℃～+85℃
- 湿度范围：90%（不结露）
- HK-CAN30B 板功耗（典型值）：+5V、1.0A

三、工作原理

3.1 工作原理概述

从功能上，HK-CAN30B 非智能隔离型 CAN 总线通讯板可分为二个部分：PCI 总线接口部分和 CAN 网络通讯部分。

3.2.1 PCI 总线接口部分

PCI 总线接口部分是 HK-CAN30B 板和 PC 机之间交换数据的桥梁，HK-CAN30B 板和 PC 机之间的数据交换是通过 PCI 总线电路实现的。HK-CAN30B 板是一 32 位 33 兆 PCI 总线从设备。兼容 PCI2.1 规范，既插即用。

3.2.2 CAN 通讯部分

该部分实现了 CAN 物理层和数据链路层协议，这部分功能受 HK-CAN30B 板内的控制电路控制。

四、主要元件位置图、信号输入/输出插座和跳线选择定义

4.1 主要元件位置图

图 4.1 为 HK-CAN30B 非智能隔离型 CAN 总线通讯板的主要元件位置图，此元件位置图上的跳线设置为出厂标准设置。

设置为：XF1, XF3 未短接。

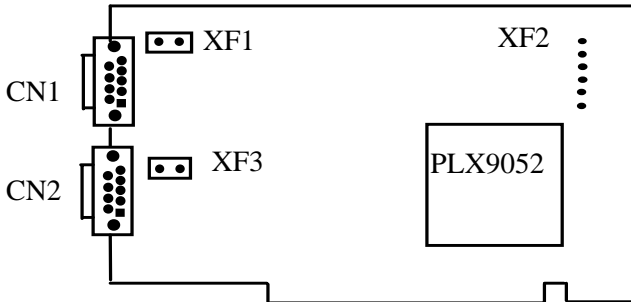


图 4.1 HK-CAN30A/B 板的主要元件位置图

CN1: CAN 通讯端口 1

CN2: CAN 通讯端口 2 (HK-CAN30A 此端口无效)

XF1: 通讯端口 1 匹配电阻的选择跳线

XF3: 通讯端口 2 匹配电阻的选择跳线

XF2: 厂家测试使用，用户无关

4.2 信号输入/输出插座定义

4.2.1 HK-CAN30B 信号输入/输出插座定义

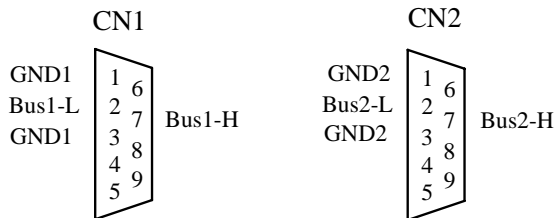


图 4.2.1 为 HK-CAN30B 板信号输入/输出插座定义图。

CN1, CAN2 信号定义说明:

Bus1- H, Bus2- H: CAN 通讯信号高电平

Bus1- L, Bus2- H: CAN 通讯信号低电平

GND1, GND2: 地

4.3.1 通讯口的设置

通讯端口采用 DB-9 针式插座, 用户可采用 DB-9 孔式插头与之配合连接, DB-9 插座的引脚参见 4.2 节

使用 HK-CAN30B 板卡时, 只使用 CAN1 和 CAN2 口的 BUS_H 和 BUS_L。接线方法见, 图 4.2.3。

CAN 是一种半双工通讯协议, 所以用户只需一对导线就可实现数据的发送和接收, 但这一对导线是有极性的, 如果网络上有多台设备连接, 请将它们的 Bus-H 连接在一起, Bus-L 连接在一起 (参见图 4.3.3)。

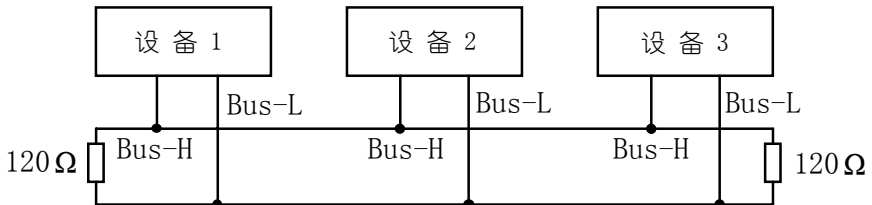
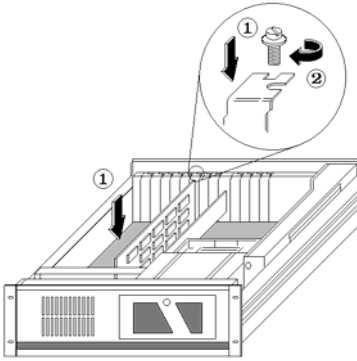


图 4.3.3 多台设备连接示意图

由于 CAN 是一种全数字的通讯, 信号在通讯电缆传送过程中存在反射, 因此, 在配置 CAN 时要加匹配电阻 (参见图 4.3.3), 电阻值为 120Ω 。本卡上已经安装了匹配电阻, 用户可根据现场实际需要, 通过跳线器 XF1, XF3 选择是否要匹配电阻。(跳线器插上为选择要匹配电阻)

五、硬件安装、拆除方法

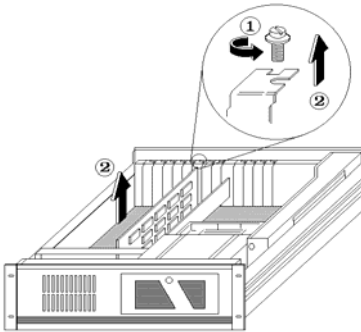
HK-CAN30B 板禁止带电插拔。接插和拔下引线插头都必须在断电的情况下进行。



5.1 安装步骤

① 把模板垂直对准任一扩展槽，以手向下用力将该板插入扩展槽。

② 拧紧螺钉，连接输入/输出电缆。



5.2 拆除步骤

① 断开输入/输出信号电缆，拧下螺钉。

② 双手向上垂直用力，将该板从扩展槽中拔出。

六、安装驱动软件

6.1 Windows98、2000 下的安装

HK-CAN30B 是 PCI 产品，在 Win98、Win2000 下为即插即用安装。当 PCI CAN 卡插入机器启动时，系统会提示找到新硬件，安装时选择 hkcan30.inf 和 hkcan30.sys 文件所在目录即可（光盘 Drivers\hkcan30b\ 与当前操作系统同名目录）。再将 HKcandll.dll (Drivers\dll\目录下) 复制到 windows\System 目录下，或执行程序目录下。

6.2 Window NT 下的安装

WinNT 操作系统不支持即插即用, 所以需要手动安装设备驱动, 运行 Setup.exe (该程序在 Drivers\hkcan30b\WinNT\目录下) 完成安装。

此外安装盘上有一套自测试软件以及用户例程。

安装完成后在 Windows 控制面板的系统设备中可以看 CAN30B 设备。在一台 PC 机中最多可以插四块 HK-CAN30B 板卡。

七、应用注意事项

7.1 注意事项

在公司售出的产品包装中, 用户将会找 HK-CAN30B 通讯板一块, HK-CAN30B 板驱动程序(光盘)及测试程序一套。同时还有产品质保卡。产品质保卡请用户务必妥善保管, 当该产品出现问题需要维修时, 请用户将产品质保卡同产品一起, 寄回本公司, 以便我们能尽快的为您解决问题。在使用 HK-CAN30B 通讯板时, 通讯板正面 IC 芯片不要用手去摸, 防止芯片受到静电的危害。

7.2 HK-CAN30B 板不能通讯

HK-CAN30B 板不能通讯是一个复杂的问题, 这即可能是硬件配置问题, 也可能是与之通讯的设备的设置的问题:

- ① 如果使用 HK-CAN30B, 则必须使用另外一个 CAN 设备与之通讯。这个 CAN 设备可以是 HK-CAN30B 也可以是华控公司的 CAN 系列中的其他产品。在 HK-CAN30B 发送任何数据之前, 该 CAN 设备必须被正确地初始化, 初始化使用的波特率必须和 HK-CAN30B 使用的一致。
- ② 终端电阻是否被正确连接。按照 CAN 总线标准, 总线两端必须各自连接一个 120 欧姆左右的电阻, 否则会降低信噪比。

八、驱动程序

为了让用户方便灵活的使用 HK-CAN30B 板, 我们为用户编制了 Windows 98/NT/2000/XP 下的驱动。以下是用户接口函数的结构定义说明。

8.1 结构体定义

8.1.1 CAN 帧数据结构:

```
typedef struct _HKCANFRAME {  
    UCHAR    nPort;  
    UCHAR    mFrame[10];  
} HKCANFRAME, *PHKCANFRAME;
```

成员含义:

nPort : 端口号

mFrame[10] : 数据帧, 按照 CAN 标准数据帧定义。

8.1.2 设备控制参数结构:

```
typedef struct _HKCANHANDLE {  
    HANDLE    mHandle;  
    DWORD    mGuid;  
    WORD     mClass;  
    WORD     mDev;  
    WORD     mType;  
    WORD     mState;  
} HKCANHANDLE, *PHKCANHANDLE;
```

成员含义:

mHandle : 设备句柄。

mGuid : 设备标志。

mClass : 设备类型。

| | |
|---|--------------|
| 0 | HK-CAN10S 板卡 |
| 1 | HK-CAN20C 板卡 |
| 2 | HK-CAN30B 板卡 |
| 4 | HK-CAN50 模块 |

mDev : 设备数量。

一部机器内, 最多可插 4 块 CAN10 板卡设备序号取值为 0-3

mType : 驱动类型。

驱动类型有内核式、虚拟设备驱动。内核式驱动用于 windows 98/2000 虚拟设备驱动用于 windows 95 和 windows98。

| | | |
|------------|---|--------|
| DRIVER_SYS | 0 | 内核式驱动 |
| DRIVER_VXD | 1 | 虚拟设备驱动 |

mState : 设备状态。

指示该设备驱动是否已正常工作。1 为正常，0 为不正常，可能是硬件参数冲突、端口未初始化、设备未打开等错误。

8.2 驱动函数说明

8.2.1 HKCanOpen ()

函数原型 :

```

BOOL HKCanOpen(
                PHKCANHANDLE mDevHandle,
                char *InDriverName,
                int Type,
                int nDev
                )

```

功能: 打开设备。

返回值:

TRUE 设备打开成功。

FALSE 设备打开错误。

参数:

mDevHandle: 设备状态控制结构体 (OUT)

InDriverName: 设备名称 (IN)

HK-CAN10S 板卡——“HKCAN10”

HK-CAN20C 板卡——“HKCAN20”

HK-CAN30B 板卡——“HKCAN30”

HK-CAN50 模块——“HKCAN50”

nDev : 设备序号 (0 - 3) (IN)

Type : 设备驱动类型 (IN)

| | | |
|------------|---|--------|
| DRIVER_SYS | 0 | 内核式驱动 |
| DRIVER_VXD | 1 | 虚拟设备驱动 |

8.2.2 HKCanClose ()

函数原型:

```
BOOL HKCanClose(  
                PHKCANHANDLE mDevHandle  
                )
```

功能: 关闭设备。

返回值:

TRUE 设备正确关闭。

FALSE 设备关闭错误。

参数:

mDevHandle : 设备状态控制结构体 (IN)

8.2.3 HKCanGetBDVersion ()

函数原型:

```
BOOL HKCanGetBDVersion(  
                PHKCANHANDLE mDevHandle,  
                char          *sVersion,  
                int           buflen  
                )
```

功能: 取得当前驱动版本号。

返回值:

TRUE 函数操作正确

FALSE 函数操作错误

参数:

mDevHandle: 设备状态控制结构体 (IN)

sVersion: 返回设备驱动版本字符串指针 (长度不小于 2 字节) (OUT)

buflen: 字符缓冲区长度 (小于 16 会导致调用错误) (IN)

8.2.4 HKCanInitState ()

函数原型:

```

BOOL HKCanInitState(
                                PHKCANHANDLE mDevHandle,
                                Int           nPort,
                                UINT          CAN_bps,
                                UCHAR        CAN_StationAddress,
                                UCHAR        CAN_Mask,
                                HANDLE        mRxEvent
                                );
    
```

功能: 初始化设备端口

返回值:

TRUE 初始化设备端口成功。

FALSE 初始化设备端口失败。

参数:

mDevHandle: 设备状态控制结构体 (IN)

nPort : 设备端口号 (取值为 0、1) (IN)

CAN_bps : HK-CAN10S 通讯板的波特率。现有五档波特率, 请参见波特率设置表。如果用户欲使用其它波特率, 请参照有关 CAN 总线原理的资料自行计算。

| 波特率 | CAN_bps |
|------|---------|
| 1M | 0xC0A3 |
| 500K | 0xC1A3 |
| 250K | 0xC3A3 |
| 125K | 0xC7A3 |
| 50K | 0xC7AF |

CAN_StationAddress: 为本站的站地址 (IN)

CAN_Mask: 为通讯板的接收屏蔽字, 与 CAN_StationAddress 共同作用决定本站可接收信息包 (IN)。

判定公式如下：

$$ID \mid \text{CAN_MASK} = \text{CAN_MASK} \mid \text{CAN_StationAddress}$$

其中：ID 为信息包标识符的高 8 位。

例如：当 CAN_MASK=0xFF 时，则接收网络上的所有信息包，当 CAN_MASK=0 时，则只接收 ID 与 CAN_StationAddress 相等的信息包。

中断屏蔽字设置实例：

假设 CAN 总线上现共有三个站点，站地址分别设为：10、12、13，其中地址为 10 的站点需要接收标识为 10 和 11 的信息包，地址为 12 的站点只关心标识为 12 的信息包，地址为 13 的站点欲接收所有的信息包。因为 10 和 11 只在最低位不同（00001010 和 00001011），所以站点 10 的中断屏蔽字应设置为 00000001，既 0x01；站点 12 只关心与本身有关的信息包，其中断屏蔽字应为 0x00；而站点 13 则应设置为 0xFF。

mRxEvent：数据帧到达核心对象句柄，一般为事件。如果采用查询方式读取数据，可将此参数设为 NULL（IN）。

8.2.5 HKCanSendFrame（）

函数原型：

```
int HKCanSendFrame(
                                PHKCANHANDLE mDevHandle,
                                PHKCANFRAME pSendFrame
)
```

功能：发送一帧数据

返回值：

- 0 设备正确发送数据
- 5 设备未初始化
- 6 设备发送忙
- 9 事件句柄为空，DLL 中事件初始化失败
- 13 设备发送超时
- 106 设备正在发送中

参数：

mDevHandle: 设备状态控制结构体 (IN)
pSendFrame: 发送数据帧指针 (IN)

8.2.6 HKCanReadFrame ()

函数原型:

```
int HKCanReadFrame(  
    PHKCANHANDLE mDevHandle,  
    Int nPort,  
    PHKCANFRAME pReadFrame  
)
```

功能: 读取一帧数据

返回值:

>=0: 该值为缓冲区中剩余帧数 (等于 0 表示缓冲区已读空)。

<0: 读取操作失败

参数:

mDevHandle: 设备状态控制结构体 (IN)
nPort : CAN 端口号 (IN)
pReadFrame : 接收数据帧指针 (OUT)

8.2.7 HKCanReadFrameEx ()

函数原型:

```
int HKCanReadFrameEx(  
    PHKCANHANDLE mDevHandle,  
    int nPort,  
    PHKCANFRAME pReadFrame,  
    int *pReadnum  
)
```

功能: 读取多个数据帧

返回值:

>=0: 该值为缓冲区中剩余帧数 (等于 0 表示缓冲区已读空)。

<0: 读取操作失败

参数:

mDevHandle: 设备状态控制结构体 (IN)
nPort : CAN 端口号 (IN)

pReadFrame : 接收数据帧指针 (大小应足够容纳指定帧数
sizeof(PHKCANFRAME) * pReadnum) (OUT)

pReadnum : 读取帧数 (IN)

8.2.8 HKCanAbortSend ()

函数原型:

```
BOOL HKCanAbortSend(  
    PHKCANHANDLE mDevHandle,  
    int nPort)
```

功能: 停止发送

返回值:

TRUE 函数执行正确,

FALSE 函数执行错误。

参数:

mDevHandle: 设备状态控制结构体 (IN)

nPort : CAN 端口号 (IN)

8.2.9 HKCanGetLastError ()

函数原型:

```
BOOL HKCanGetLastError(  
    PHKCANHANDLE mDevHandle,  
    PULONG pError  
)
```

功能: 获取当前设备状态

返回值:

TRUE 函数执行正确,

FALSE 函数执行错误。

参数:

mDevHandle: 设备状态控制结构体 (IN)

pError : 返回数据为设备端口状态 (OUT)

d0-d7 : 核心对象状态 (一般为事件)

d8-d15: 设备错误状态

d16-d23: 端口二当前接收数据帧数 (未读取, 在驱动缓冲区内)

d24-d31: 端口一当前接收数据帧数 (未读取, 在驱动缓冲区内)

核心对象状态（d0-d7）：

| | | |
|-------------------|---|---------|
| HKCAN_EVENT_ERROR | 1 | 设备操作失败 |
| HKCAN_EVENT_SEND | 2 | 设备发送完成 |
| HKCAN_EVENT_READ | 4 | 接收到数据帧 |
| HKCAN_EVENT_INIT | 8 | 端口初始化完成 |

设备错误状态（d8-d15）：

| | | |
|----------------------|------|--------------|
| | 0 | 为设备操作正确 |
| INIT_SUCCESS | 0x01 | //初始化成功 |
| INIT_ERROR | 0x02 | //初始化错误 |
| RECEIVE_QUE_OVERFLOW | 0x03 | //接收数据区溢出 |
| SENDFRAME_FAILED | 0x04 | //发送出错（失败） |
| SEND_QUE_OVERFLOW | 0x05 | //发送数据区溢出 |
| DEV_INT_ERROR | 0x06 | //CAN 设备出错中断 |
| UNINIT_ERROR | 0x07 | //CAN 没有初始化 |
| UNEXIST_BOARD_ERROR | 0x08 | //CAN10 板不存在 |
| RECEIVE_ERROR | 0x09 | //接收出错（失败） |
| RECEIVE_QUE_EMPTY | 0x0a | //接收数据区空 |

8.3 CAN 信息包格式说明

CAN 信息包分为两部分：信息部分和数据部分。头两个字节为信息部分，其前十一位为标识符。然后是一位 RTR 位（应设为 0），最后是四位的 DLC（数据长度位，即所发数据的实际长度，单位：字节）。其余八个字节是数据部分，存有实际要发的数据。详见右表。

注：DOS、WINDOWS 是美国 Microsoft 公司的注册商标

| | | | | | | | | |
|------|----------|---|-----|---|-----|---|---|---|
| | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
| 字节1 | 标识符（高八位） | | | | | | | |
| 字节2 | 标识符 | | RTR | | DLC | | | |
| 字节3 | 数 据 | | | | | | | |
| 字节4 | 数 据 | | | | | | | |
| 字节5 | 数 据 | | | | | | | |
| 字节6 | 数 据 | | | | | | | |
| 字节7 | 数 据 | | | | | | | |
| 字节8 | 数 据 | | | | | | | |
| 字节9 | 数 据 | | | | | | | |
| 字节10 | 数 据 | | | | | | | |