

K-7120 USBCAN 总线转换模块使用说明书

一：概述：

K-7120 是与 USB 总线相连的 CAN 总线接口模块，通过 USB 总线与 PC 机连接，应用于基于 CANBUS 的现场总线应用领域，集成有一个 CAN 接口，一个 USB 接口，CAN 总线接口为光电隔离。具有体积小，即插即用，自动配置，总线供电等特点，是便携式系统用户的最佳选择。

二：技术指标：

1. CANBUS：

采用 PHILIP CAN 控制器 SJA1000

CAN2.0B.规范（PeliCAN）兼容 CAN2.0A，符合 ISO/ISO11898 规范

传输速率：5K-1M BPS

传输介质：屏蔽或非屏蔽双绞线。

隔离电压：1000V

2. USB：

采用 USB 接口芯片 CH372。

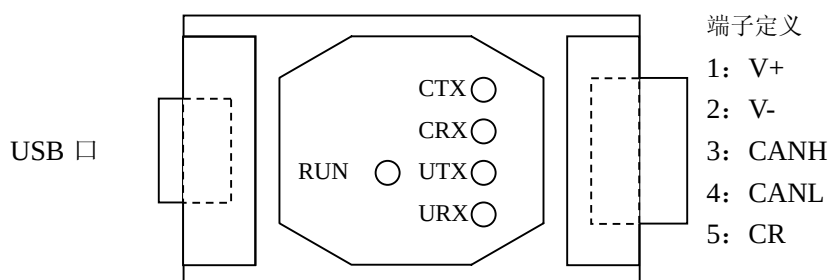
符合 USB1.1 协议规范，通讯电缆长度<1.5m

3. 工作温度：0°C-70°C

4. 外形尺寸：100mm×70mm×20mm，塑料外壳。

5. 供电：USB 总线供电或外部供电（5V/12V/24V，100mA）

三：接口定义：



1. 电源接口 V+, V-:

通过该电源接口，由外部电源为系统供电，外部电源并不是必需的，当系统连接到一个 USB 总线供电的集线器上时，必须连接外部电源。

选择 5V 供电时，应保证模块电源输入电压不低于 5V。

选择 12V 供电时，应保证模块电源输入端电压不低于 12V。

选择 24V 供电时，应保证模块电源输入端电压不低于 24V。

2. USB 接口：

标准 USB 接口,USB-BB

3. CANBUS 接口：

CANH、CANL。

当该模块位于 CAN 总线终端时，将 CR 端与 CANL 端短接。

4. 信号指示灯(壳：

RUN：红色，模块工作指示灯，上电亮。

CTX：黄色，CAN 通讯发送指示灯，发送时闪烁。

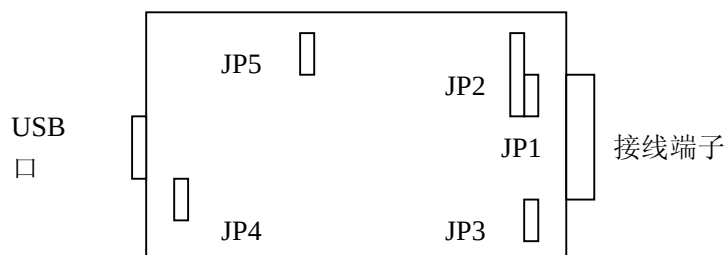
CRX：绿色，CAN 通讯接收指示灯，发送接收时均闪烁。

UTX: 黄色, USB 通讯发送指示灯, 发送时闪烁。

URX: 绿色, USB 通讯接收指示灯, 接收时闪烁。

如果传送数据时接收发送指示灯常亮或常灭, 表示总线工作不正常。

5. 跳线选择:



JP1: 当参数设置选择缺省值时, 将 JP1 短接, 上电后, 按缺省参数运行, 否则断开。

JP2: 供电方式选择: JP2 的 1、2 短接, 3、4 短接, 为 5V 供电, 断开为 12V/24V 供电, (模块外供电方式出厂时选定, 该跳线用户无须设置)

JP3: 当 CAN 总线选择高速率 (800K 以上) 通讯时, 将 JP3 短接, 否则 JP3 断开。

JP4: 选择 USB 总线供电时短接, 选择外供电方式时断开。

JP5: 看门狗有效选择跳线, 插长为有效。

四: 使用说明:

1. CAN2.0B 通讯帧格式:

标准帧格式 (兼容 CAN2.0A):

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
帧结构信息 (SFF)	FF	RTR	X	X	DLC. 3	DLC. 2	DLC. 1	DLC. 0
识别码 1 (ID1)	ID. 28	ID. 27	ID. 26	ID. 25	ID. 24	ID. 23	ID. 22	ID. 21
识别码 2 (ID2)	ID. 20	ID. 19	ID. 18	X (注 1)	X	X	X	X
数据场	DATA1-DATAN (N≤8)							

扩展帧格式:

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
帧结构信息 (EFF)	FF	RTR	X	X	DLC. 3	DLC. 2	DLC. 1	DLC. 0
识别码 1 (ID1)	ID. 28	ID. 27	ID. 26	ID. 25	ID. 24	ID. 23	ID. 22	ID. 21
识别码 2 (ID2)	ID. 20	ID. 19	ID. 18	ID. 17	ID. 16	ID. 15	ID. 14	ID. 13
识别码 3 (ID3)	ID. 12	ID. 11	ID. 10	ID. 9	ID. 8	ID. 7	ID. 6	ID. 5
识别码 4 (ID4)	ID. 4	ID. 3	ID. 2	ID. 1	ID. 0	X (注 1)	X	X
数据场	DATA1-DATAN (N≤8)							

注: 1: FF: 结构格式, “1” 为扩展帧, “0” 为标准帧

2: RTR: “1” 为远程帧, “0” 为数据帧

3: X: 不影响

4: DLC: 数据场的长度

注 1: 在接收帧中, 该位为远程帧位。

发送到 CAN 口的报文, 只有当报文标示符 (IDn.7-IDn.0) 等于验收码 (ACRn.7-ACRn.0)

由验收屏蔽码 (AMRn.7-AMRn.0) 相关屏蔽的那些位时, 报文才予以接收。(n=1-4)

AMR 为 0 的位对应的 ACR 的位为报文标示符相关屏蔽位。

在设置验收码和屏蔽码时，对于标准帧，只用到 ACR1、AMR1 和 ACR2、AMR2 的高 3 位，ACR 的其余位设为 0，AMR 的其余位设为 1。

详细信息请参阅 SJA1000 数据手册。

2. 参数配置说明(十六进制):

PC 机通过 USB 口对 K-7120 进行参数设置，该参数保存在 E2PROM 中，断电保存。

如果按重新配置的参数运行，请在配置完毕，回读正确后退出应用程序并拔出电缆，

重

新插入，并再次回读，以确定配置的正确性。

设置参数包括：CAN 接口波特率出厂缺省值为 20K，验收码（ACR1-4）为 0，屏蔽码（AMR1-4）为 FFH。

CANBUS 速率表:

	5K	10K	20K	50K	125K	250K	500K	800K
BTR0	BF	31	18	09	03	01	0	0
BTR1	FF	1C	1C	1C	1C	1C	1C	16

配置指令格式:

30	AA	BTR0	BTR1	ACR1-4	AMR1-4
----	----	------	------	--------	--------

回读配置状态:

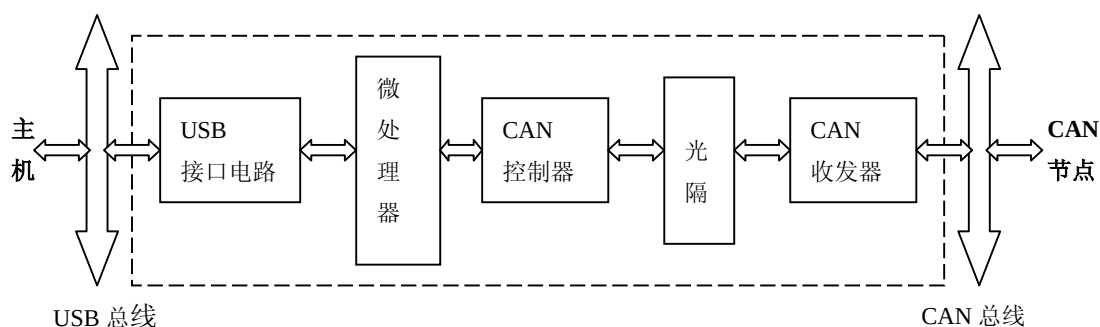
30	55
----	----

返回:

30	FF	BTR0	BTR1	ACR1-4	AMR1-4
----	----	------	------	--------	--------

3. 透明转换工作原理

4.



USB 转 CAN:

PC 机将待发送的数据以 CAN2.0B 帧格式，从 USB 口发出去。K-7120 将从 USB 口接收到的数据从 CAN 接口发出去。

CAN 转 USB

K7120 将从 CAN 口接收到的数据由 USB 口发出去。

5. 动态链接库使用说明

提供标准动态链接库，操作系统环境 win98/ME/2000/XP, 可被任意开发环境调用，如 VB，VC，DELPHI。

1) WINDOWS 下驱动的安装:

当第一次连接 USB 设备到 PC 机上时，WINDOS 提示找到新硬件，会出现一个对话框

提示您安装设备驱动程序,通过对话框导航来安装光盘提供的驱动程序即可。安装好驱动程序后,查看系统的设备硬件列表,外部接口项显示“USB 设备 K7120”。

2) 用户程序接口函数说明。

Ch375dll.dll 提供以下函数。

CH375OpenDevice(): 打开 USB 设备。

函数原型: HANDLE WINAPI CH375OpenDevice(ULONG iIndex // 指定 CH375 设备序号,0 对应第一个设备,以下同); // 打开 CH375 设备,返回句柄,出错则无效

出错返回值为-1: 可能原因:

1: 没有连接硬件。

2: 没有安装驱动程序

CH375CloseDevice(): 关闭 USB 设备。

函数原型: VOID WINAPI CH375CloseDevice(ULONG iIndex // 指定 CH375 设备);

此函数一定要被调用。建议在关闭设备并退出应用程序后再拔出 USB 电缆。

CH375SetTimeout(): 设定读写超时函数

函数原型: VOID WINAPI CH375SetTimeout(ULONG iIndex,
ULONG iwritetimeout,ULONG ireadtimeout)

Iwritetimeout: 写超时,单位毫秒,缺省设置为 0xFFFFFFFF 表示无超时

Ireadtimeout: 读超时,单位毫秒,缺省设置为 0xFFFFFFFF 表示无超时

CH375ReadData(): 读取上传数据块。

函数原型: BOOL WINAPI CH375ReadData (ULONG iIndex, // 指定 USB 设备序号,
下同

PVOID oBuffer, // 指向一个足够大的缓冲区,用于保存读取的数据

PULONG ioLength); // 指向长度单元,输入时为准备读取的长度,返回后为实际读取的长度,最大长度为 64 个字节。

返回值: 操作状态,成功或失败

CH375WriteData(): 下传数据块

函数原型: BOOL WINAPI CH371WriteData (ULONG iIndex,

PVOID iBuffer, // 指向一个缓冲区,放置准备写出的数据

PULONG ioLength); // 指向长度单元,输入时为准备写出的长度,返回后为实际写出的长度,最大长度为 64 个字节。

返回值: 操作状态,成功或失败

typedef VOID (* mPCH375_INT_ROUTINE)(// 中断服务程序

PUCHAR iBuffer); // 指向一个缓冲区,提供当前的中断特征数据

CH375WaitInterrupt(): 等待中断事件

BOOL WINAPI CH375WaitInterrupt(ULONG iIndex,

PVOID oBuffer); // 指向一个足够大的缓冲区,用于保存中断特征数据,为 NULL 则无需返回数据

CH375AbortWaitInt(): 放弃等待中断

BOOL WINAPI CH375AbortWaitInt(ULONG iIndex);

CH371SetIntRoutine(): 设定中断服务程序

BOOL WINAPI CH375SetIntRoutine(ULONG iIndex,

mPCH371_INT_ROUTINE iIntRoutine); // 指定中断服务程序,为 NULL 则取消中断服务,否则在中断时调用该程序

CH375ReadInter ()：读取中断数据

BOOL WINAPI CH375ReadInter(ULONG iIndex,
PVOID oBuffer, // 指向一个足够大的缓冲区,用于保存读取的中断数据
PULONG ioLength); // 指向长度单元,输入时为准备读取的长度,返回后为实际读取的长度

3) VC++编程实例：(VB 编程参见光盘提供的 K7120.vbp)

将 CH375DLL.LIB 拷贝到项目所在目录，在项目的设置中加入 CH375DLL.LIB 链接。

在头文件中加入链入标识。

或用 LoadLibrary () 加载 CH372 的 DLL，

定义一个结构体变量，如 CAN_REG，用于透明转换方式下 CAN 口接收发送

```
typedef struct CAN_REG
```

```
{
    uchar ff;
    uchar id0;
    uchar id1;
    uchar id2;
    uchar id3;
    uchar buffer[8];
} mCAN_REG
```

定义一个结构体变量，如 CAN_CONFIG，用于请求/应答模式的应用层结构

```
typedef struct CAN_CONFIG
```

```
{
    uchar mstx
    uchar mcomd
    uchar mbtr0
    uchar mbtr1
    uchar macr(4)
    uchar mamr(4)
}mcan_config
```

index=0;//设备序号,如果有多个 K7120 则指示序号

【打开设备 0】:

```
mopen=CH375OpenDevice(0)
```

【初始化 CAN 接口】:

```
mcan_config.stx=0x30
```

```
mcan_config.command=0xaa
```

```
mcan_config.mbtr0=btr0
```

```
mcan_config.mbtr1=btr1
```

```
mcan_config.macr[0]=acr1
```

```
mcan_config.macr[1]=acr2
```

```
mcan_config.macr[2]=acr3
```

```
mcan_config.macr[3]=acr4
```

```
mcan_config.mamr[0]=amr1
mcan_config.mamr[1]=amr2
mcan_config.mamr[2]=amr3
mcan_config.mamr[3]=amr4
mlength=12
I=ch372writedata(index,&mcan_config,&mlength);
```

【配置参数回读】:

```
mcan_config.mstx=0x30
mcan_config.command=0x55
mlength=2
I=ch372writedata(index,&mcan_config,&mlength);
mlength=sizeof(mcan_config)
I=ch372readdata(index,&mcan_config,&mlength)
Btr0=mcan_config.mbtr0
Btr1=mcan_config.mbtr1
Acr1=mcan_config.macr[0]
Acr2=mcan_config.macr[1];
Acr3=mcan_config.macr[2];
Acr4=mcan_config.macr[3];
```

```
Amr1=mcan_config.mamr[0]
Amr2=mcan_config.mamr[1];
Amr3=mcan_config.mamr[2];
Amr4=mcan_config.mamr[3];
```

【发送 CAN 通讯包】。

标准数据帧:

```
mcan_reg.ff=cansff
mcan_reg.id0=canid0
mcan_reg.id1=canid1
mcan_reg.id2=data0
mcan_reg.id3=data1
mcan_reg.buffer[0]=data2
mcan_reg.buffer[1]=data3
... ..
mcan_reg.buffer[5]=data7
mlength=11
I=ch372writedata(index,&mcan_reg,&mlength);
```

扩展数据帧:

```
mcan_reg.ff=caneff
mcan_reg.id0=canid0
mcan_reg.id1=canid1
mcan_reg.id2=canid2
```

```
mcan_reg.id3=canid3
```

```
mcan_reg.buffer[0]=data0
```

```
mcan_reg.buffer[1]=data1
```

```
... ..
```

```
mcan_reg.buffer[7]=data7
```

```
mlength=13
```

```
I=ch372writedata(index,&mcan_reg,&mlength);
```

【接收 CAN 通讯包】

中断方式：PC 机在 2 毫秒之内响应中断上传数据。

```
CH375SetIntRoutine( Index, mInterruptEvent );// 设定中断服务程序
```

```
VOID mInterruptEvent( // 中断服务程序
```

```
PUCHAR iBuffer ) // 指向一个缓冲区,提供当前的中断特征数据
```

```
{
```

```
    CH375ReadInter(index, mcan_reg, mlength );
```

```
    PostMessage( mDialogMain, WM_COMMAND, IDC_INT_PRESS, 0 );
```

```
    //mInterruptEvent 线程向主程序发出消息进行处理
```

```
}
```

加载中断服务程序,中断服务程序是在应用层执行的,其线程优先级是

```
THREAD_PRIORITY_TIME_CRITICAL
```

消息处理程序

```
{
```

```
    mlength=sizeof(mcan_reg)
```

```
    I=ch372readdata(index,&mcan_reg,&mlength)
```

```
    Canid0=mcan_reg.id0
```

```
    Canid1=mcan_reg.id1
```

```
    data0=mcan_reg.id2
```

```
    data1=mcan_reg.id3
```

```
    data2=mcan_reg.buffer[0]
```

```
    ... ..
```

```
    data7=mcan_reg.buffer[7]
```

扩展帧

```
    mlength=sizeof(mcan_reg)
```

```
    I=ch372readdata(index,&mcan_reg,&mlength)
```

```
    Canid0=mcan_reg.id0
```

```
    Canid1=mcan_reg.id1
```

```
    Canid2=mcan_reg.id2
```

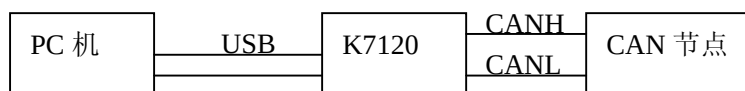
```
    Canid3=mcan_reg.id3
```

```

data0=mcan_reg.buffer[0]
... ..
data7=mcan_reg.buffer[7]
}

```

2. 测试及范例使用。



K7120 连线正确后，将测试程序 K7120。EXE(VB)复制到选择的目录底下，双击：

输入 BTR0、BTR1、ACR1-4, AMR1-4, 按“配置”，即按所选参数配置 CAN 通讯口，如选择 K7120 缺省设置，则可省略此步。除非改变参数，否则省略此步。

设置参数包括：CANBUS 的通信波特率，验收码（ACR），屏蔽码（AMR），CAN 接口波特率出厂缺省值为 BTR0=18H, BTR1=1CH, 验收码 ACR 为 0, 屏蔽码 AMR 为 FFH。按“回读”，读回参数设置值。

发送测试：将待发送的数据填入文本框中，按“发送”，

例如：PC 机向 1 号 CAN 节点发送数据 12345678

1 号 CAN 节点向 PC 机发送数据 12345678，接收到的信息显示在接收信息文本框中。

五：保修：

本产品自售出之日起两年内，凡用户遵守储存，运输及使用要求，而产品质量低于技术指标的，凭保修单免费维修，因违反操作规定和要求而造成损坏的，需交纳器件和维修费。

六：产品成套性：

K-7120 通讯转换模块一块，测试程序及使用说明书一份。

七：常见问题解答：

PC 机和 CAN 节点之间不能通讯。

- 1: 测试 USB 接口：插入 USB 电缆后，RUN 指示灯亮，配置或回读正常，URX, UTX 指示灯闪烁，则 USB 接口工作正常，
- 2: 测试 CAN 接口：发送或接收测试正常，CRX, CTX 闪烁。若 CRX, CTX 常亮，表示 CAN 总线故障，检查 CAN 总线连线及 CAN 节点之间波特率设置是否匹配。K7120 的 CAN 控制器选用 SJA1000，晶振为 16M。

附录：SJA1000 标准波特率

SJA1000 独立 CAN 控制器的通讯波特率由寄存器 BTR0、BTR1、晶振等参数共同决定，下表列出了一组推荐的 BTR0、BTR1 设置值（十六进值），标“*”号的为 CiA 协会推荐的标准值。

K7120CAN 控制器采用 16MHZ 的晶振。

序号	波特率	晶振=16MHZ	晶振=12MHZ
----	-----	----------	----------

	(Kbps)	BTR0	BTR1	BTR0	BTR1
1	5	BF	FF	--	--
2*	10	31	1C	65	1C
3*	20	18	1C	52	1C
4	40	87	FF	--	--
5*	50	09	1C	47	1C
6	80	83	FF	--	--
7*	100	04	1C	43	1C
8*	125	03	1C	42	1C
9	200	81	FA	--	--
10*	250	01	1C	41	1C
11	400	80	FA	--	--
12*	500	00	1C	40	1C
13	666	80	B6	--	--
14*	800	00	16	40	16
15*	1000	00	14	40	14