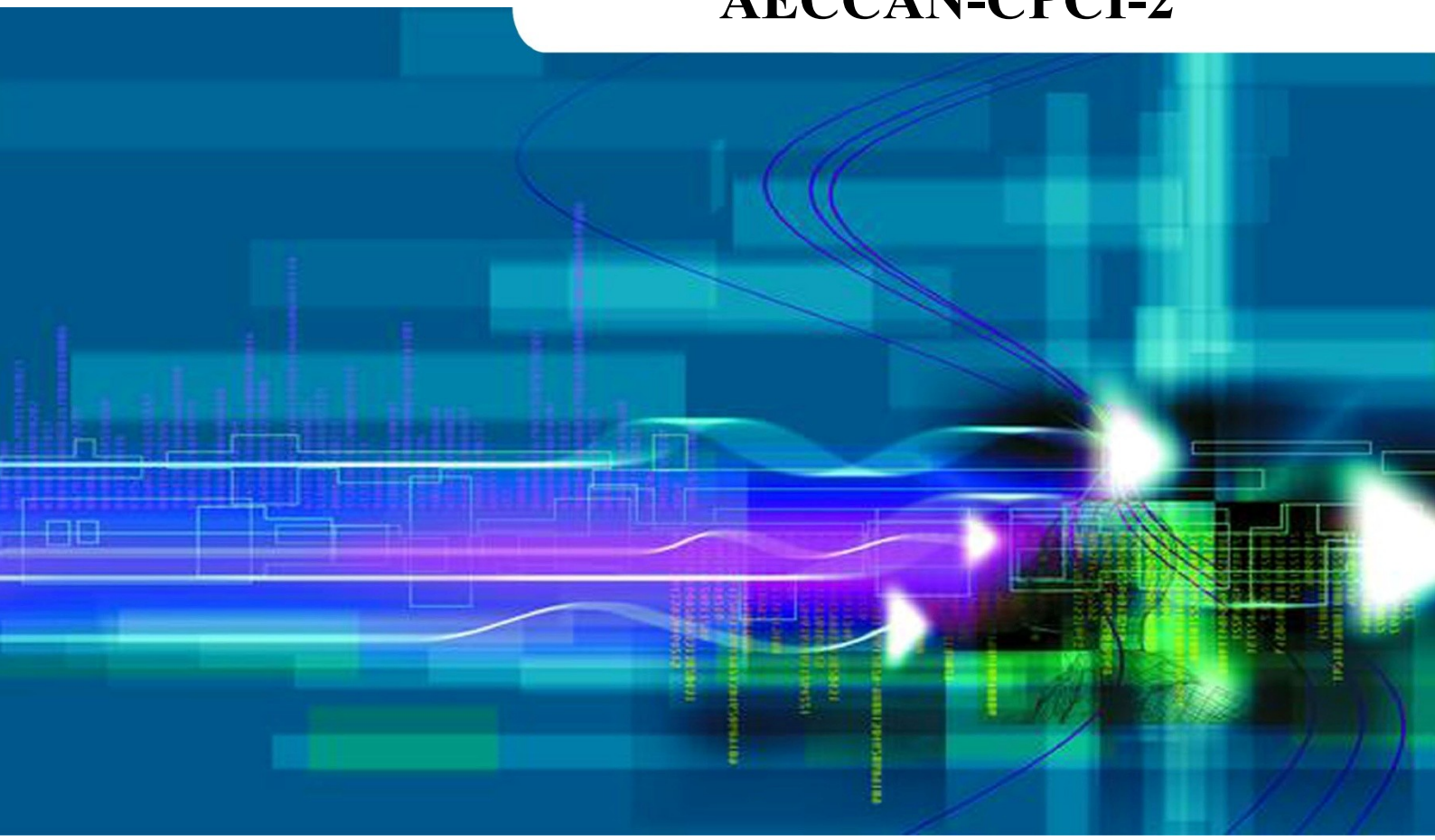
AECCAN-CPCI-2



用户手册

北京神州飞航科技有限责任公司

www.aectech.cn

声　明

AECCAN-CPCI-2 用户手册
文档版本：V1.00
发布日期：2009-01-12

北京神州飞航科技有限责任公司
地址：北京市海淀区西三环北路 21 号北控久凌大厦北楼 1009 室
邮编：100089
电话：010-68403305，68403306，68403307，68403308
传真：010-68403309
E-mail：sales@aectech.cn
网址：www.aectech.cn

目 录

第一章　概述

1.1 关于本手册

本手册适用于下列产品型号：

- **AECCAN-CPCI-2** 双通道 CAN 总线通讯卡

本手册是关于上述产品的完全使用指南。以下各章节提供了关于该产品更详细的信息，包括产品的功能特性、安装使用、硬件和软件说明等内容。

本手册的电子版本，您可以在购买产品的配套光盘中获得，也可以通过访问北京神州飞航有限责任公司网站（http://www.aectech.cn）下载。

注意

在使用该产品之前，请您详细阅读本手册各章节的内容。

1.2 产品描述

AECCAN-CPCI-2 是一款 CAN 总线通讯板卡，其强大的功能能够满足不同用户的工业测量和自动化控制需求，良好的兼容性适用于各类系统配置。

1.2.1 特性

- 两路 CAN 总线通讯
- 波特率最高为 1Mbps
- 支持 CAN2.0B 规范
- 终端电阻可选
- 16 路双向 TTL 电平数字 I/O
- CPCI 总线

1.2.2 详细描述

- CAN 通讯
  - 通道数量：2 通道 CAN 通讯
  - 输入（Input）和输出（Output）

Symbol	Parameter	Min	Max
V_{IH}	Input High Level	0.7VCC	VCC+0.3V
I_{IH}	Input High Current	－200	＋30uA
V_{IL}	Input Low Level	－0.3V	0.3VCC
I_{IL}	Input Low Current	－100	－600 uA
V_{OH}	Output High Level	2.75V	4.5V
V_{OL}	Output Low Level	0.5V	2.25V

- TTL 电平数字 I/O
  - 通道数量：16 路 TTL 电平数字 I/O
  - 输入（Input）和输出（Output）

Symbol	Parameter	Min	Max
V_{IH}	Input High Level	2V	-

I_{OH}	High lever output Current	-	－24mA
V_{IL}	Input Low Level	-	0.8V
I_{OL}	Low lever output Current	-	24mA
V_{IB}	Input Voltage	-	5V
V_{OB}	Output Voltage	-	5V
Δ t/Δ v	Input transition fall or rise time	-	10ns/V

1.2.3 一般规格

- 物理尺寸：160mm×100mm
- 连接器： DB37 座孔连接器
- 工作电源：5V
- 相对湿度：5～95%，无凝结

1.3 产品安装

1.3.1 安装之前的准备

1. 在您安装产品之前请检查包装是否完好，以确定产品在运输的过程中没有遭到损坏。如果包装发现有破损，请您马上与运输商联系。
2. 在打开包装后请检查产品以及配件的完整性。打开产品外包装后，您应该发现如下产品

   - AECCAN-CPCI-2 CAN 总线通讯卡
   - 产品合格证
   - 产品配套光盘
   - 标配连接器

   如有规格不符，请您立刻联系我们，我们将负责维修或者更换。
3. 如果有可能，请您准备防静电工作台并佩戴防静电腕带。如果不具备以上静电防护装备，请您接触计算机设备的导地部分，例如机箱壳金属部分，以释放身体上的静电。

现在您可以准备安装 AECCAN-CPCI-2 CAN 总线通讯卡了。

1.3.2 硬件安装

第一步： 打开板卡的防静电包装袋，取出板卡。

注意

手持板卡时，请您尽量只接触板卡的边缘。在板卡安装到您的计算机设备之前，请将板卡平放置于防静电包装袋中，这样有利于保护板卡不受静电损伤。取出板卡后，请您保留产品的防静电和防震包装，以便在您不使用时产品可以妥善存放。

图 1-1 AECCAN-CPCI-2 产品图片

第二步： 关闭计算机设备的电源，将板卡安装到您的计算机机箱内。

第三步： 将配套的连接器或连接电缆插到板卡的连接器接口上。

关于连接电缆的制作请参照 2.2.1 节的内容。

开启计算机，然后安装产品的驱动。

1.3.3 驱动安装

在产品配套光盘的“驱动安装”目录中，您可以找到 AECCAN-CPCI-2 板卡的驱动，您也可以通过访问北京神州飞航科技有限责任公司网站（http://www.aectech.cn）下载获得 AECCAN-CPCI-2 板卡的驱动。请您按如下步骤安装产品的驱动：

1、 将板卡插入工控机 CPCI 插槽

2、 系统提示找到新硬件，需要安装它的软件

3、 点击硬件安装的下一步，确定驱动程序所在的目录

- Windows 2000/98 的操作系统
  请指向光盘中“驱动安装程序\98-2000”目录
- Windows xp 操作系统
  请指向光盘中“驱动安装程序\XP”目录
- Windows server 2003 操作系统
  请指向光盘中“驱动安装程序\2003” 目录

4、 点击下一步，直到驱动程序安装成功

在完成 AECCAN-CPCI-2 板卡驱动安装后，您可以通过计算机系统的“设备管理器”来确认板卡驱动是否正确安装。访问“设备管理器”可以通过“控制面板”/“系统”/“设备管理器”。如果板卡驱动正确安装，您可以在“设备管理器”的设备列表中看到 AECCAN-CPCI-2 板卡设备项，如图 1-2 所示。

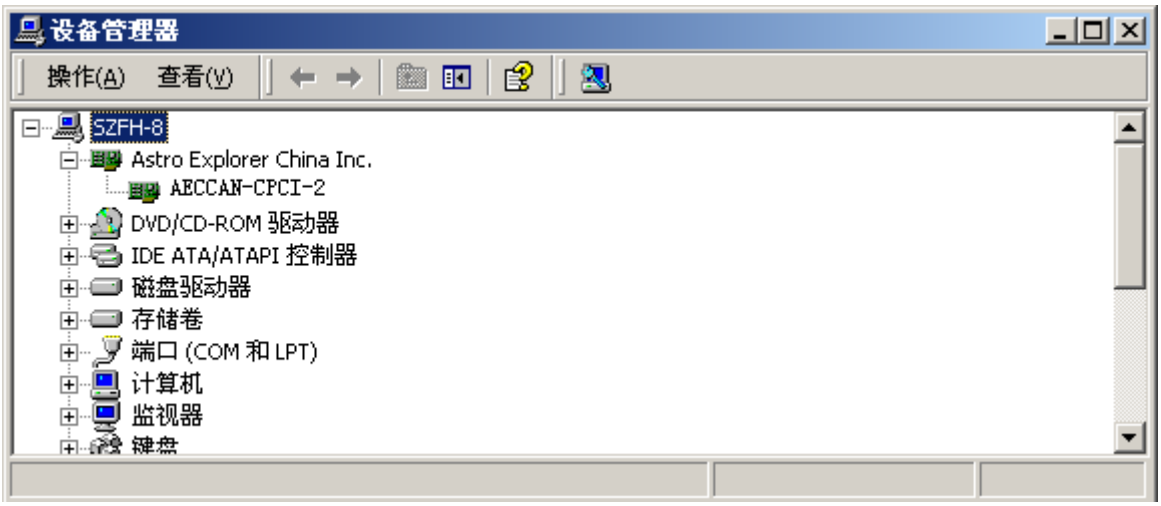


图 1-2　AECCAN-CPCI-2 板卡在设备管理器中的设备项

1.3.4 演示应用软件安装

在安装完 AECCAN-CPCI-2 板卡驱动之后，您可以安装配套光盘中附带的功能演示软件。执行安装程序，按系统提示，即可以完成演示软件的安装。

演示软件可以满足基本的产品测试和演示功能。具体使用方法，您可以参考第 4 章节的内容。

第二章　硬件说明

本章描述了 AECCAN-CPCI-2 CAN 总线通讯卡硬件信息，包括硬件设置、I/O 连接器和信号定义等。

2.1 功能结构图

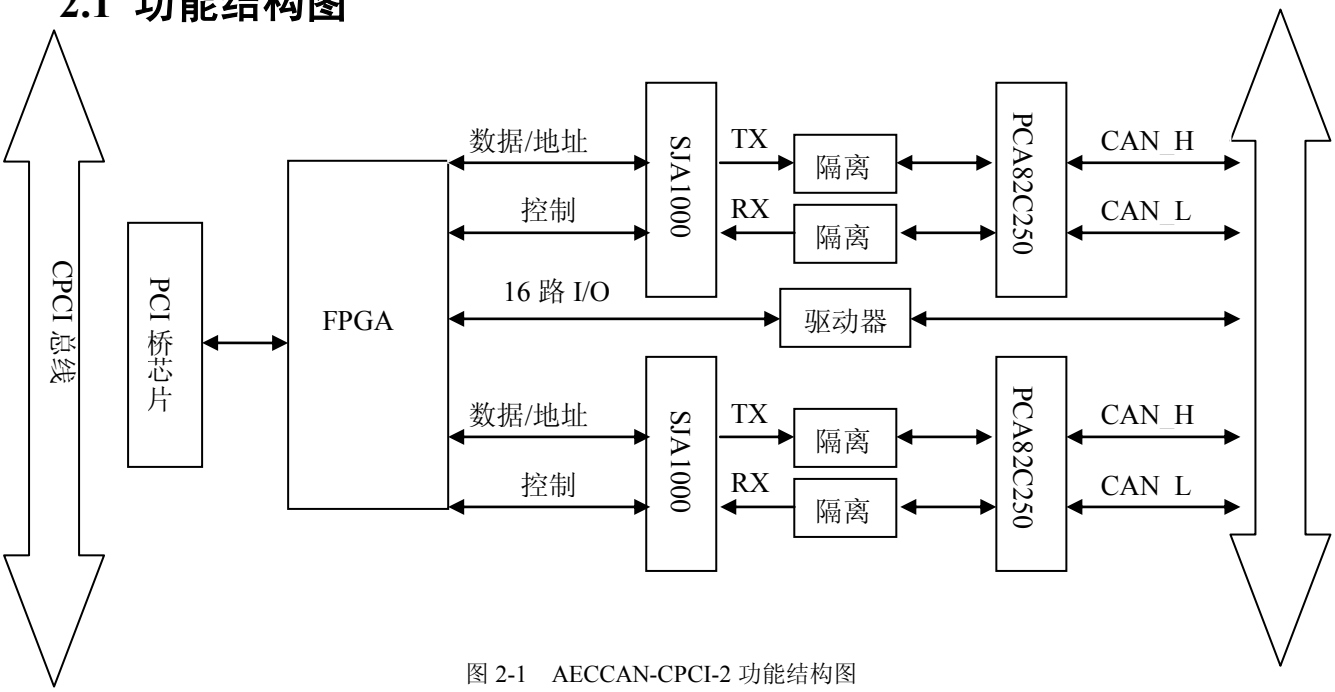


图 2-1　AECCAN-CPCI-2 功能结构图

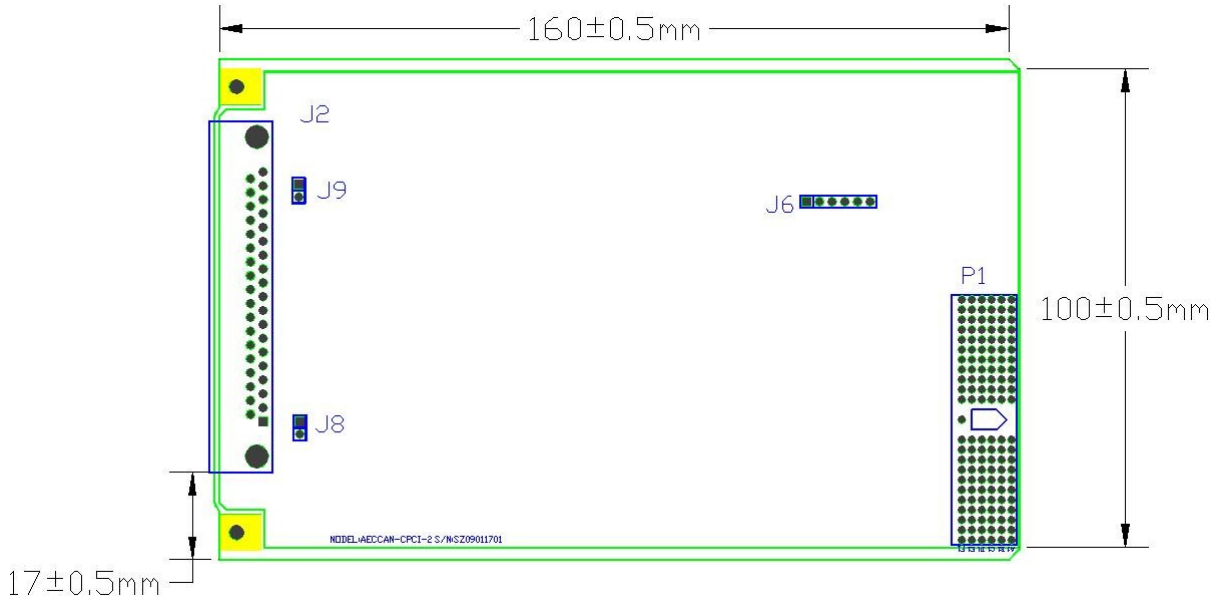


图 2-2　AECCAN-CPCI-2 印制板示意图

表 2-1 AECCAN-CPCI-2 接头说明

接头	说明
J6	JTAG 编程接头，出厂配置使用
J2	信号输入输出连接器
J8，J9	终端电阻选择跳线

2.2 连接器和信号定义

2.2.1 CAN 通讯信号连接器定义：J2

CAN_H0	1	20	GD0
CAN_L0	2	21	VD0
NC	3	22	NC
NC	4	23	NC
NC	5	24	GND
GND	6	25	DIO15
DIO14	7	26	DIO13
DIO12	8	27	DIO11
DIO10	9	28	DIO9
DIO8	10	29	DIO7
DIO6	11	30	DIO5
DIO4	12	31	DIO3
DIO2	13	32	DIO1
DIO0	14	33	GND
GND	15	34	NC
NC	16	35	NC
NC	17	36	CAN_H1
GD1	18	37	CAN_L1
VD1	19		

图 2-3 描述了 CAN 信号连接器的点号定义

表 2-2 连接信号描述

信号	参考	方向	描述
CAN_H0	GD0	输入/输出	第 0 路 CAN 通讯的 CAN_H 信号
CAN_L0	GD0	输入/输出	第 0 路 CAN 通讯的 CAN_L 信号
GD0	—	—	第 0 路 CAN 通讯的参考地
VD0	GD0	—	第 0 路 CAN 通讯的输出电源
CAN_H1	GD1	输入/输出	第 1 路 CAN 通讯的 CAN_H 信号
CAN_L1	GD1	输入/输出	第 1 路 CAN 通讯的 CAN_L 信号
GD1	—	—	第 1 路 CAN 通讯的参考地

VD1	GD1	—	第 1 路 CAN 通讯的输出电源
DIO [0…15]	GND	输入/输出	**数字量输入输出通道** 第 0 通道至第 15 通道
GND	—	—	系统地
N/C	—	—	**无定义** 请悬空处理，不要连接任何信号和地

2.2.2 终端电阻选择跳线：J8 、J9

通过 J8 和 J9 选择是否接入终端电阻，当跳上 J8/J9 时，第 0/1 路 CAN 总线接入终端电阻，当不设置跳线时，不接入终端电阻。

第三章　驱动程序编程接口

本章主要讲述了如何使用 AECCAN-CPCI-2 板卡的驱动程序接口，为用户编程提供参考。AECCAN-CPCI-2 驱动程序提供了丰富的接口函数，能满足用户对板卡的操作需求；具有良好的兼容性，能适用于多种编程环境；操作简单方便，可以大大缩短用户的开发周期。

3.1 动态库 DLL

AECCAN-CPCI-2 驱动程序接口函数按 ANSI C 标准编写，以动态链接库 DLL 形式提供给用户。您可以在 AECCAN-CPCI-2 板卡配套光盘中获取。

3.1.1 适用编程工具

运行环境：Windows 98/2000/2003/xp 操作系统

开发工具：

- Visual C++
- Visual Basic
- C++ Builder
- Delphi
- Labview
- Labwindows/CVI

3.1.2 需要引用的文件

当您进行程序开发时，需要引用下列文件：

- 库文件：AECCAND2.dll 和 AECCAND2.lib
- 函数库头文件：AECCAND2_lib.h

3.1.3 引用结构说明

3.1.3.1 中断屏蔽寄存器设置结构

```
typedef struct
{
    BOOL RX_INT;
    BOOL ERR_WARNING_INT;
    BOOL OV_INT;
    BOOL WAKEUP_INT;
    BOOL ERR_PASSIVE_INT;
    BOOL ARBIT_LOST_INT;
    BOOL BUSERR_INT;
}INT_MASK_REGISTER_STRUCT, *pINT_MASK_REGISTER_STRUCT;
```

结构参数说明：

RX_INT：接收中断使能

TRUE：使能，当接收到一帧数据时，板卡将会触发一个中断

FALSE：禁止

ERR_WARNING_INT：错误报警中断使能

TRUE：使能，如果错误（至少一个错误计数器满或超过了由错误报警限制寄存器定的报警限制）或总线状态改变（总线开启/关闭），驱动将会触发一个中断事件

FALSE：禁止

OV_INT：数据溢出中断使能

TRUE：使能，若接收 FIFO 溢出，驱动将会触发一个中断事件

FALSE：禁止

WAKEUP_INT：唤醒中断使能

TRUE：使能，如果睡眠模式下的 CAN 控制器被唤醒，则请求相应的中断

FALSE：禁止

ERR_PASSIVE_INT：错误消极中断使能

TRUE：若 CAN 控制器的错误状态改变（从消极到活动或反之），则请求相应的中断

FALSE：禁止

ARBIT_LOST_INT：仲裁丢失中断使能

TRUE：使能，如果 CAN 控制器已丢失了仲裁，则请求相应的中断

FALSE：禁止

BUSERR_INT：总线错误中断使能

TRUE：使能，如果检测到总线错误，则请求相应的中断

FALSE：禁止

3.1.3.2 CAN 总线参数设置结构

```
typedef struct
{
    BYTE BTR0;
    BYTE BTR1;
    BYTE AcceptFilterMode;
    BYTE ACR[4];
    BYTE AMR[4];
    BYTE EMLR;
}PELICAN_PARAM_STRUCT, *pPELICAN_PARAM_STRUCT;
```

结构变量说明:

EMLR：错误报警限制设置

BTR0：总线时间参数寄存器 0

BTR1：总线时间参数寄存器 1，与 BTR0 共同决定 CAN 总线通讯的波特率

表 3-1 波特率与时间参数寄存器

波特率（Kbps）	BTR0	BTR1
10	31	1C
20	18	1C
50	09	1C
100	04	1C
125	03	1C
250	01	1C
500	00	1C

800	00	16
1000	00	14

AcceptFilterMode：验收滤波器模式设置

1：单个验收滤波器

0：两个验收滤波器

ACR：验收代码设置

AMR：验收屏蔽寄存器设置

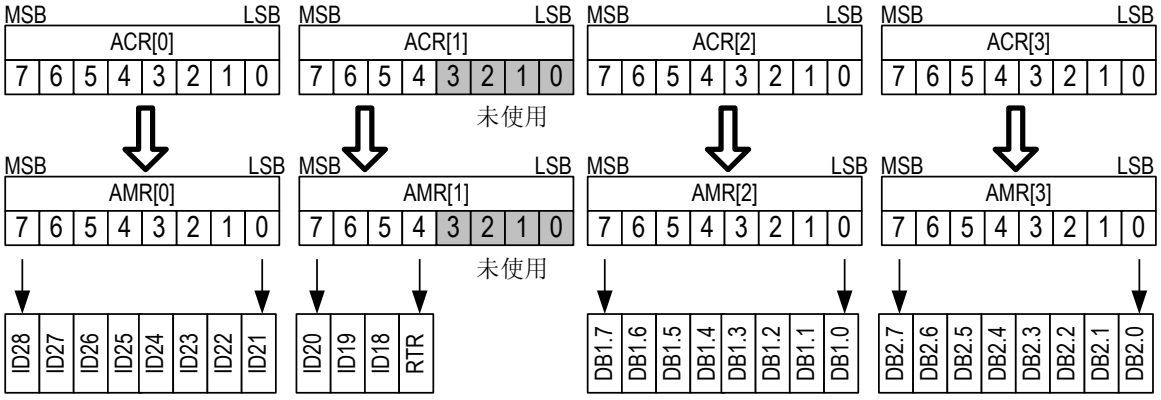


图 3-1 单滤波器标准帧验收机制

从图 3-2 可以看出，单滤波器标准帧的验收位包括 11 位标识符、RTR 位及 2 个数据字节。对于不需要经过验收滤波的位，验收滤波器需要将对应的位置‘1’。但如果数据帧为远程帧导致没有数据字节，或因为设置了相应的数据长度代码而没有或只有 1 个数据字节，此种情况数据也会被接收的。

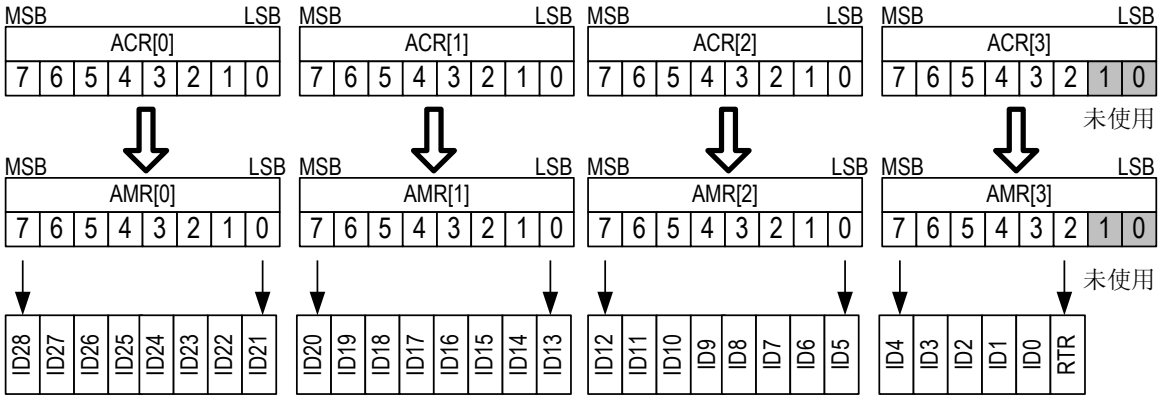


图 3-2 单滤波器扩展帧验收机制

单滤波器扩展帧的验收位包括 29 位标识符、RTR 位。对于不需要经过验收滤波的位，验收滤波器需要将对应的位置‘1’。

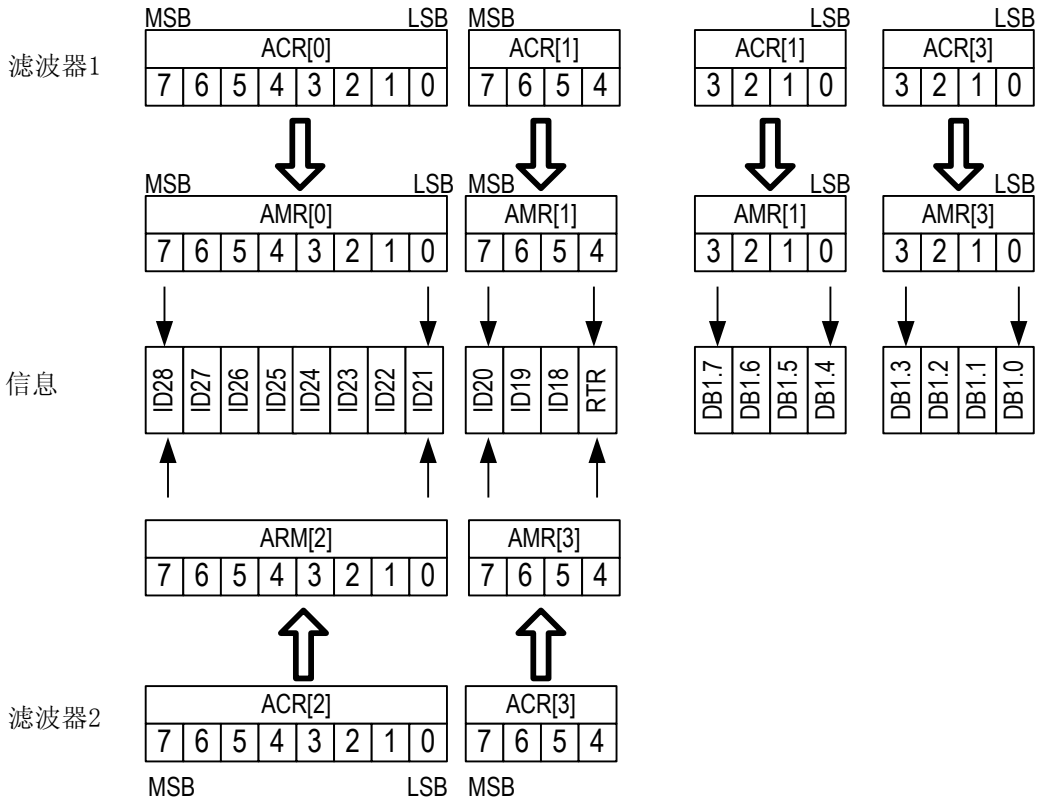


图 3-3 双滤波器标准帧验收机制

从图 3-3 可以看出，双滤波器标准帧的验收位包括 11 位标识符、RTR 位及 1 个数据字节。对于不需要经过验收滤波的位，验收滤波器需要将对应的位置‘1’。但如果数据帧为远程帧导致没有数据字节，或因为设置了相应的数据长度代码而没有数据字节，此种情况数据也会被接收的。

信息只需要通过任何一个滤波器，帧就将会被接收。

从图 3-4 可以看出，双滤波器扩展帧的验收位包括 16 位标识符。对于不需要经过验收滤波的位，验收滤波器需要将对应的位置‘1’。

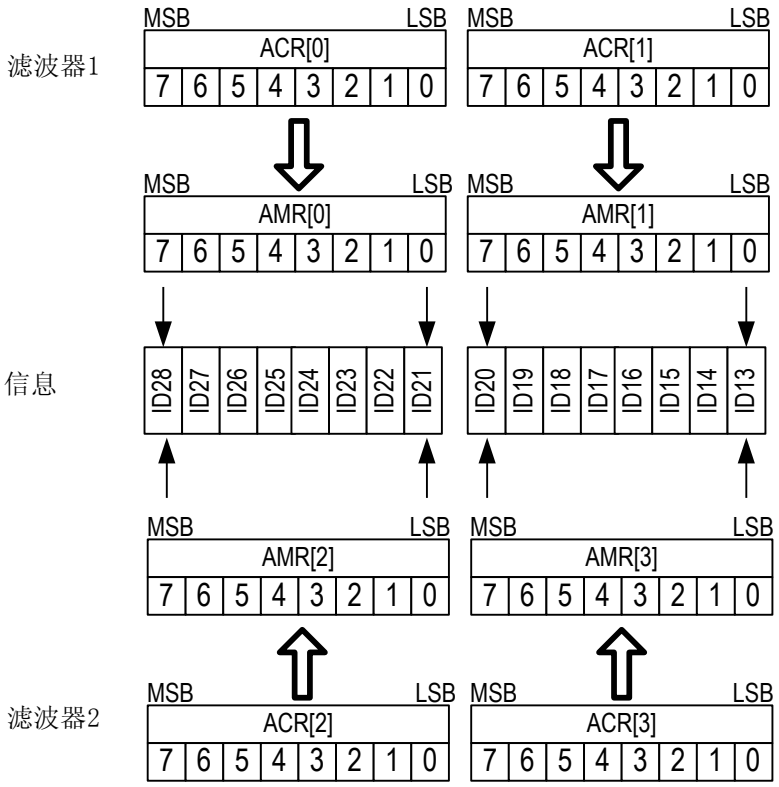


图 3-4 双滤波器扩展帧验收机制

3.1.3.3 帧结构

```
typedef struct
{
    BOOL Extended_Id_Format;
    BOOL Remote_Transmit_Request;
    DWORD Id;
    BYTE DataCnt;
    BYTE DataBuf[16];
}FRAME_STRUCT, *pFRAME_STRUCT;
```

结构参数说明:

Extended_Id_Format：帧标志符的长度

TRUE：扩展帧，帧标识符为 29 位

FALSE：标准帧，帧标识符为 11 位

Remote_Transmit_Request：帧类型

TRUE：远程帧

FALSE：数据帧

Id：帧标识符

DataCnt：帧数据字节的个数，取之范围 0～8

DataBuf：存放帧的数据字节

表 3-2 标准帧的帧标志符 Id 格式

D31	D30	D29	D28	D27	D26	D25	D24
ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21

D23	D22	D21	D20	D19	D18	D17	D16
ID.20	ID.19	ID.18	RTR	0	0	0	0

D15	D14	D13	D12	D11	D10	D9	D8
0	0	0	0	0	0	0	0

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	0	0	0	0	0

表 3-3 扩展帧的帧标志符 Id 格式

D31	D30	D29	D28	D27	D26	D25	D24
ID.28	ID.27	ID.26	ID.25	ID.24	ID.23	ID.22	ID.21

D23	D22	D21	D20	D19	D18	D17	D16
ID.20	ID.19	ID.18	ID.17	ID.16	ID.15	ID.14	ID.13

D15	D14	D13	D12	D11	D10	D9	D8
ID.12	ID.11	ID.10	ID.9	ID.8	ID.7	ID.6	ID.5

D7	D6	D5	D4	D3	D2	D1	D0
ID.4	ID.3	ID.2	ID.1	ID.0	RTR	0	0

3.1.3.4 板卡句柄结构

```
typedef struct
{
    HDEVICE hCard;
    BYTE CardId;
} ST_DEVDSC, *HDEVICE;
```

3.2 驱动程序函数功能

实现 2 路 CAN 总线的数据的收发和 16 路数字量的输入、输出

3.2.1 CAN 总线的初始化

- 通讯波特率可设，最高速率可到 1000Kbps
- 帧过滤：可设置验收代码和验收屏蔽寄存器
- 单、双验收滤波器可选
- 错误报警限制可设
- 多达 7 种中断：接收中断、错误报警中断、数据溢出中断、唤醒中断、错误消极中断、仲裁丢失中断及总线错误中断

3.2.2 CAN 总线数据的接收

- 两种数据接收方式：查询和中断
- 查询方式接收时，提供接收 FIFO 的空满状态，接收 FIFO 的大小为 64 个字节
- 中断方式接收时，提供接收 FIFO 里帧数量，此时接收 FIFO 最多能容纳 4000 帧数据
- 驱动自动解帧

3.2.3 CAN 总线数据的发送

- 以查询的方式发送数据
- 软件提供发送缓存的空满状态
- 数据按帧发送

3.2.4 CAN 总线错误界定

- 仲裁丢失捕捉
- 错误代码捕捉
- 错误报警状态
- 发送/接收错误计数

3.2.5 数字量 I/O

- 16 路单向数字量 I/O
- 可设置 I/O 通道的方向
- 可随时获取和设置 I/O 通道的状态

3.3 驱动程序函数接口说明

本节内容详细描述了 API 函数的调用原型，函数功能、参数说明和返回值。

3.3.1 AECCAND2_Open

函数原型：BOOL __stdcall ***AECCAND2_Open*** (HDEVICE *phAECCAND2, BYTE CardId);

函数功能：打开板卡，并分配板卡资源

参数说明：phAECCAND2：指针变量，用来存放板卡的句柄

CardId：板卡编号，取值范围为 0～7（若工控机中同时插 8 块 AECCAN-CPCI-2 板卡，板卡的编号按板卡所在的插槽离 CPU 的距离由远到近依次编号 0，1，…， 7；若 PC 机中同时只插一块板，板卡编号为 0）

返回值：　若板卡打开成功，函数返回值为真；否则为假

3.3.2 AECCAND2_Close

函数原型：BOOL __stdcall ***AECCAND2_Close*** (HDEVICE hAECCAND2);

函数功能：关闭板卡，释放板卡资源

参数说明：hAECCAND2：板卡的句柄

返回值：　若板卡关闭成功，函数返回值为真；否则为假

3.3.3 AECCAND2_Reset

函数原型：BOOL __stdcall ***AECCAND2_Reset*** (HDEVICE hAECCAND2);

函数功能：板卡复位函数

参数说明：hAECCAND2：板卡的句柄

返回值： 若板卡复位成功，函数返回值为真；否则为假

板卡复位执行的操作为：

1) 验收滤波器的工作模式为双滤波器
2) 接收和发送 FIFO 的状态为空
3) 中断使能寄存器无影响，中断信号清除
4) 波特率设置无影响
5) 接收和发送错误计数器清 0
6) 验收代码寄存器和验收屏蔽寄存器设置无影响
7) 错误报警限制寄存器默认值为 96
8) 仲裁丢失捕捉寄存器清 0
9) 错误代码捕捉寄存器清 0
10) 总线状态为开启状态

3.3.4 AECCAND2_Init

函数原型：BOOL __stdcall ***AECCAND2_Init*** (HDEVICE hAECCAND2, BYTE ChanNo,
pPELICAN_PARAM_STRUCT pPeliCanParam);

函数功能：CAN 初始化

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

pPeliCanParam：CAN 总线参数结构指针变量

返回值： 若 CAN 初始化成功，函数返回值为真；否则为假

3.3.5 AECCAND2_SetIntMaskReg

函数原型：BOOL __stdcall ***AECCAND2_SetIntMaskReg*** (HDEVICE hAECCAND2,
BYTE ChanNo, pINT_MASK_REGISTER_STRUCT pIntMaskReg);

函数功能：设置中断屏蔽寄存器

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

pIntMaskReg：中断屏蔽寄存器结构指针变量

返回值： 若中断屏蔽寄存器设置成功，函数返回值为真；否则为假

3.3.6 AECCAND2_SetEvent

函数原型：BOOL __stdcall ***AECCAND2_SetEvent*** (HDEVICE hAECCAND2, BYTE ChanNo, HANDLE hEvent);

函数功能：设置中断触发事件，当某路产生中断时，驱动将会向应用程序产生一个中断触发事件

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

hEvent： 错误报警中断、数据溢出中断、唤醒中断、错误消极中断、仲裁丢失中断、总线错误中断触发事件。由 CreateEvent 函数创建

返回值： 若中断触发事件设置成功，函数返回值为真；否则为假

3.3.7 AECCAND2_RxEmpty

函数原型：BOOL __stdcall ***AECCAND2_RxEmpty*** (HDEVICE hAECCAND2, BYTE ChanNo);

函数功能：查询方式下（将中断屏蔽寄存器中的 RX_INT 位置为 FALSE），获取接收 FIFO 的状态

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

返回值： 若接收 FIFO 为空，函数返回值为真；否则为假

3.3.8 AECCAND2_GetFrameNum

函数原型：WORD __stdcall ***AECCAND2_GetFrameNum*** (HDEVICE hAECCAND2, BYTE ChanNo);

函数功能：中断方式下（将中断屏蔽寄存器中的 RX_INT 位置为 TRUE），获取接收 FIFO 的帧数量

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

返回值： 函数返回接收 FIFO 的帧数量，最多为 4000 帧

3.3.9 AECCAND2_RxRead

函数原型：BOOL __stdcall ***AECCAND2_RxRead*** (HDEVICE hAECCAND2, BYTE ChanNo,

pFRAME_STRUCT pRxFrame);

函数功能：接收帧

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

PRxFrame：指针变量，存放接收到的帧

返回值： 若接收到了一帧数据，函数返回值为真；否则为假

3.3.10 AECCAND2_TxEmpty

函数原型：BOOL __stdcall ***AECCAND2_TxEmpty*** (HDEVICE hAECCAND2, BYTE ChanNo);

函数功能：获取发送缓冲区的状态

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

返回值： 若发送缓冲区为空，函数返回值为真；否则为假

3.3.11 AECCAND2_TxWrite

函数原型：BOOL __stdcall ***AECCAND2_TxWrite*** (HDEVICE hAECCAND2, BYTE ChanNo,

pFRAME_STRUCT pTxFrame);

函数功能：发送帧

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

PTxFrame：指针变量，存放待发送的帧

返回值： 若帧可以发送，函数返回值为真；否则为假

3.3.12 AECCAND2_TxAbort

函数原型：void __stdcall ***AECCAND2_TxAbort*** (HDEVICE hAECCAND2, BYTE ChanNo);

函数功能：取消等待发送的帧

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

返回值： 空

3.3.13 AECCAND2_Get_IntReg_Status

函数原型：BYTE __stdcall ***AECCAND2_Get_IntReg_Status*** (HDEVICE hAECCAND2,

BYTE ChanNo);

函数功能：获取中断状态寄存器的状态

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

返回值：　函数返回中断状态寄存器的状态，具体值见表 3-4

表 3-4 中断状态寄存器

D7	D6	D5	D4	D3	D2	D1	D0
BEI	ALI	EPI	WUI	DOI	EI	0	0

EI：错误报警中断状态

‘1’：中断请求，错误状态位或总线状态改变，调用 AECCAND2_Get_Error_Warning_Status 函数可获取中断状况

‘0’：无中断产生

DOI：数据溢出中断状态

‘1’：中断请求，接收 FIFO 溢出

‘0’：无中断产生

WUI：唤醒中断状态

‘1’：中断请求，睡眠模式下的 CAN 控制器监测到总线有活动

‘0’：无中断产生

EPI：错误消极中断状态

‘1’：中断请求，CAN 控制器的错误状态改变（从消极到活动或反之）

‘0’：无中断产生

ALI：仲裁丢失中断状态

‘1’：中断请求，CAN 控制器丢失仲裁，变为接收器。仲裁丢失可通过函数 AECCAND2_Arbitration_Lost_Capture 捕捉

‘0’：无中断产生

BEI：总线错误中断状态

‘1’：中断请求，CAN 控制器检测到总线有错误。调用函数 AECCAND2_Bus_Error_Capture 可捕捉错误代码

‘0’：无中断产生

3.3.14 AECCAND2_Arbitration_Lost_Capture

函数原型：BYTE __stdcall ***AECCAND2_Arbitration_Lost_Capture*** (HDEVICE hAECCAND2,

BYTE ChanNo);

函数功能：仲裁丢失捕捉

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

返回值： 仲裁丢失捕捉寄存器的状态，见表 3-5

表 3-5 仲裁丢失捕捉寄存器

D7	D6	D5	D4	D3	D2	D1	D0
0	0	0	ALC4	ALC3	ALC2	ALC1	ALC0

仲裁丢失时，会产生相应的仲裁丢失中断。同时，位流处理器的当前位位置被捕捉送入仲裁丢失捕捉寄存器。一直到用户通过软件读这个值，寄存器中的内容都不会变。随后，捕捉机制又被激活了。

ALC4～ALC0 的取值见表 3-6

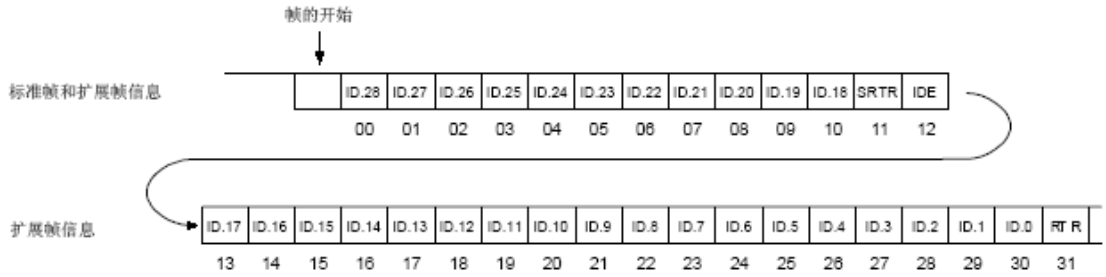


表 3-6 仲裁丢失位定义

位					十进制	功能
ALC4	ALC3	ALC2	ALC1	ALC0		
0	0	0	0	0	0	仲裁丢失在识别码的 bit1；注 1
0	0	0	0	1	1	仲裁丢失在识别码的 bit2
0	0	0	1	0	2	仲裁丢失在识别码的 bit3
0	0	0	1	1	3	仲裁丢失在识别码的 bit4
0	0	1	0	0	4	仲裁丢失在识别码的 bit5
0	0	1	0	1	5	仲裁丢失在识别码的 bit6
0	0	1	1	0	6	仲裁丢失在识别码的 bit7
0	0	1	1	1	7	仲裁丢失在识别码的 bit8
0	1	0	0	0	8	仲裁丢失在识别码的 bit9
0	1	0	0	1	9	仲裁丢失在识别码的 bit10
0	1	0	1	0	10	仲裁丢失在识别码的 bit11
0	1	0	1	1	11	仲裁丢失在 SRR 位;注 2
0	1	1	0	0	12	仲裁丢失在 IDE 位
0	1	1	0	1	13	仲裁丢失在识别码的 bit12;注 3
0	1	1	1	0	14	仲裁丢失在识别码的 bit13;注 3
0	1	1	1	1	15	仲裁丢失在识别码的 bit14;注 3

1	0	0	0	0	16	仲裁丢失在识别码的 bit15;注 3
1	0	0	0	1	17	仲裁丢失在识别码的 bit16;注 3
1	0	0	1	0	18	仲裁丢失在识别码的 bit17;注 3
1	0	0	1	1	19	仲裁丢失在识别码的 bit18;注 3
1	0	1	0	0	20	仲裁丢失在识别码的 bit19;注 3
1	0	1	0	1	21	仲裁丢失在识别码的 bit20;注 3
1	0	1	1	0	22	仲裁丢失在识别码的 bit21;注 3
1	0	1	1	1	23	仲裁丢失在识别码的 bit22;注 3
1	1	0	0	0	24	仲裁丢失在识别码的 bit23;注 3
1	1	0	0	1	25	仲裁丢失在识别码的 bit24;注 3
1	1	0	1	0	26	仲裁丢失在识别码的 bit25;注 3
1	1	0	1	1	27	仲裁丢失在识别码的 bit26;注 3
1	1	1	0	0	28	仲裁丢失在识别码的 bit27;注 3
1	1	1	0	1	29	仲裁丢失在识别码的 bit28;注 3
1	1	1	1	0	30	仲裁丢失在识别码的 bit29;注 3
1	1	1	1	1	31	仲裁丢失在 RTR 位;注 3

注 1：仲裁丢失的二进制编码结构位的号码

注 2：标准帧信息的 RTR 位

注 3：只使用于扩展帧信息

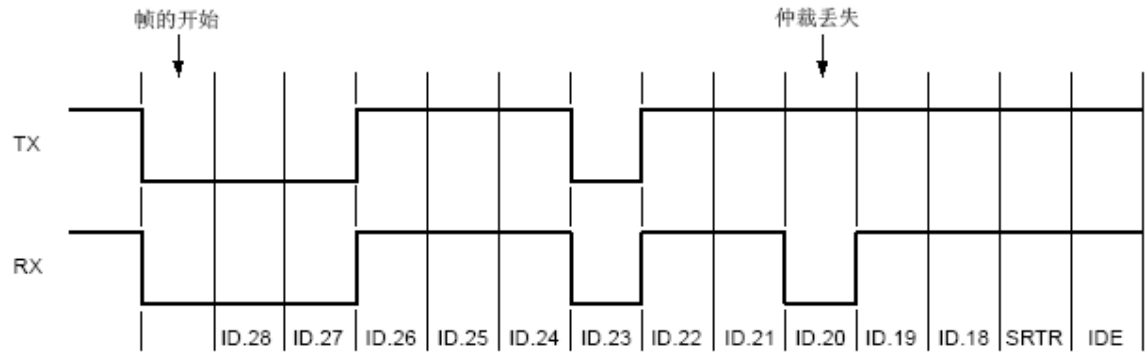


图 3-5 仲裁丢失举例

此时仲裁丢失寄存器的值为 8

3.3.15 AECCAND2_Bus_Error_Capture

函数原型：BYTE __stdcall ***AECCAND2_Bus_Error_Capture*** (HDEVICE hAECCAND2

BYTE ChanNo);

函数功能：错误代码捕捉

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

返回值：　错误代码捕捉寄存器，见表 3-7

表 3-7 错误代码捕捉寄存器

D7	D6	D5	D4	D3	D2	D1	D0
ECC7	ECC6	ECC5	ECC4	ECC3	ECC2	ECC1	ECC0

ECC7～6：定义见表 3-8

ECC5：方向位

‘1’：接收时发生的错误

‘0’：发送时发生的错误

ECC4～0：定义见表 3-9

表 3-8 ECC7 ~6 定义

ECC7	ECC6	定义
0	0	位错
0	1	格式错
1	0	位填充错
1	1	其它错误

表 3-9 ECC4 ~ 0 定义

ECC4	ECC3	ECC2	ECC1	ECC0	定义
0	0	0	1	1	帧开始
0	0	0	1	0	ID.28-ID.21
0	0	1	1	0	ID.20-ID.18
0	0	1	0	0	SRTR位
0	0	1	0	1	IDE位
0	0	1	1	1	ID.17-ID.13
0	1	1	1	1	ID.12-ID.5
0	1	1	1	0	ID.4-ID.0
0	1	1	0	0	RTR位
0	1	1	0	1	保留位1
0	1	0	0	1	保留位0
0	1	0	1	1	数据长度代码
0	1	0	1	0	数据区
0	1	0	0	0	CRC序列
1	1	0	0	0	CRC定义符
1	1	0	0	1	应答通道
1	1	0	1	1	应答定义符
1	1	0	1	0	帧结束
1	0	0	1	0	中止
1	0	0	0	1	活动错误标志

1	0	1	1	0	消极错误标志
1	0	0	1	1	支配控制位误差
1	0	1	1	1	错误定义符
1	1	1	0	0	溢出标志

总线发生错误时被迫产生相应的错误中断（中断使能时）。同时，位流处理器的当前位置被捕捉送入错误代码捕捉寄存器。其内容直到用户通过软件读出时都是不变的。读出后，捕捉机制又被激活了。

3.3.16 AECCAND2_Get_Error_Warning_Status

函数原型：BYTE __stdcall ***AECCAND2_Get_Error_Warning_Status*** (HDEVICE hAECCAND2,

BYTE ChanNo);

函数功能：获取错误报警状态

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

返回值：　错误报警状态

D7	D6	D5	D4	D3	D2	D1	D0
BS	ES	0	0	0	0	0	0

BS：总线状态

‘1’：总线关闭，CAN 控制器不参与总线活动

‘0’：总线开启，CAN 控制器参与总线活动

ES：出错状态

‘1’：出错，至少一个错误计数器满或超过了报警限制寄存器的值

‘0’：正常，两个错误计数器的值都在报警限制以下

3.3.17 AECCAND2_Get_Error_Counter

函数原型：void __stdcall ***AECCAND2_Get_Error_Counter*** (HDEVICE hAECCAND2,

BYTE ChanNo, BYTE *RxErrCnt, BYTE *TxErrCnt);

函数功能：获取错误计数器计数值

参数说明：hAECCAND2：板卡的句柄

ChanNo：通道号，取值范围为 0～1

RxErrCnt：接收错误计数器

TxErrCnt：发送错误计数器

返回值：　空

3.3.18 AECCAND2_GetSerialNum

函数原型：void __stdcall ***AECCAND2_GetSerialNum*** (HDEVICE hAECCAND2, DWORD *SN);

函数功能：获取板卡的序列号

参数说明：hAECCAND2：板卡的句柄

SN：用来存放板卡的序列号

返回值： 空

3.3.19 AECCAND2_IO_SetDir

函数原型：BOOL __stdcall ***AECCAND2_IO_SetDir*** (HDEVICE hAECCAND2, BOOL Input);

函数功能：设置 I/O 通道的方向

参数说明：hAECCAND2：板卡的句柄

Input：I/O 通道方向设置位

TRUE：I/O 通道的方向为输入

FALSE：I/O 通道的方向为输出

返回值： 若 I/O 通道的方向设置成功，函数返回值为真；否则为假

3.3.20 AECCAND2_IO_Get_Input_Status

函数原型：WORD __stdcall ***AECCAND2_IO_Get_Input_Status*** (HDEVICE hAECCAND2);

函数功能：获取 I/O 通道的状态

参数说明：hAECCAND2：板卡的句柄

返回值： I/O 各通道的状态，定义如下表

D15	D14	D13	D12	D11	D10	D9	D8
I/O_15	I/O_14	I/O_13	I/O_12	I/O_11	I/O_10	I/O_9	I/O_8
D7	D6	D5	D4	D3	D2	D1	D0
I/O_7	I/O_6	I/O_5	I/O_4	I/O_3	I/O_2	I/O_1	I/O_0

I/O_0 ~ I/O_15 为第 0 ~ 15 路 I/O 通道的状态，对应的位为 1，表明 I/O 通道的状态为高；为 0，表明 I/O 通道的状态为低

3.3.21 AECCAND2_IO_Set_Output_Status

函数原型：void __stdcall ***AECCAND2_IO_Set_Output_Status*** (HDEVICE hAECCAND2,

WORD POSR);

函数功能：设置 I/O 通道的输出状态（仅在 I/O 通道的方向配置为输出有效）

参数说明：hAECCAND2：板卡的句柄

POSR：各 I/O 输出通道的状态设置。D0～D15 位分别为第 0～15 路 I/O 通道的输出状态，对应的位为 1，I/O 通道输出高；为 0，I/O 通道输出低。

返回值： 空

3.4 驱动接口函数调用步骤

3.4.1 打开板卡

调用函数 AECCAND2_Open 来找板卡，并分配板卡资源

3.4.2 复位板卡

调用函数 AECCAND2_Reset 来复位板卡到一个初始状态

3.4.3 CAN 总线的初始化

调用函数 AECCAND2_Init 来初始化 CAN 总线

3.4.4 CAN 总线数据的接收

3.4.4.1 查询方式下帧的接收

1、 禁止接收中断，调用函数 AECCAND2_SetIntMaskReg 将中断屏蔽寄存器中的 RX_INT 置为 FALSE

2、 获取接收 FIFO 的状态（AECCAND2_RxEmpty）

3、 若接收 FIFO 不空，则接收帧（AECCAND2_RxRead）

4、 重复 2～3 步

3.4.4.2 中断方式下帧的接收

1、 使能接收中断，调用函数 AECCAND2_SetIntMaskReg 将中断屏蔽寄存器中的 RX_INT 置为 TRUE

2、 获取接收 FIFO 的帧数量（AECCAND2_GetFrameNum）

3、 若接收 FIFO 帧数量不为 0，则接收帧（AECCAND2_RxRead）

4、 重复 2～3 步

3.4.5 CAN 总线数据的发送

1、 获取发送缓冲区的状态（AECCAND2_TxEmpty）

2、 若发送缓冲区为空，则发送数据（AECCAND2_TxWrite）

3、 重复以上 2 步

3.4.6 CAN 总线错误界定

1、 设置中断屏蔽寄存器（AECCAND2_SetIntMaskReg），使能相应的中断（ERR_WARNING_INT、OV_INT、WAKEUP_INT、ERR_PASSIVE_INT、ARBIT_LOST_INT、

BUSERR_INT）

2、 用户创建中断事件（CreateEvent，标准 API 函数）

3、 设置中断触发事件（AECCAND2_SetEvent）

4、 判断是否产生中断（WaitForSingleObjects，标准 API 函数），若产生中断，则调用函数 AECCAND2_Get_IntReg_Status 判断中断源，并进行相应的中断处理

3.4.7 数字量 I/O

3.4.7.1 数字量输入

1、 设置 I/O 通道的方向为输入（将函数 AECCAND2_IO_SetDir 的入口参数 Input 置为 TRUE）

2、 获取 I/O 通道的状态（AECCAND2_IO_Get_Input_Status）

3、 重复步骤 2

3.4.7.2 数字量输出

1、 设置 I/O 通道的方向为输出（将函数 AECCAND2_IO_SetDir 的入口参数 Input 置为 FALSE）

2、 设置 I/O 通道的状态（AECCAND2_IO_Set_Output_Status）

3、 重复步骤 2

3.4.8 获取板卡序列号

调用函数 AECCAND2_GetSerialNum 可获取板卡序列号

3.4.9 关闭板卡

应用程序退出时，释放中断触发事件（CloseHandle），复位板卡（AECCAND2_Reset），并关闭板卡（AECCAND2_Close）

第四章　功能演示软件

AECCAN-CPCI-2 主要分为初始化窗口、板卡选择窗口、主窗口、Can 总线设置窗口，IO 操作窗口。

4.1 使用环境

4.1.1 硬件

- 内存：至少 128M 内存
- 总线：支持 PCI 总线
- 分辨率：建议至少 1024*728
- 板卡：AECCAN-CPCI-2 通讯板

4.1.2 操作系统

Windows 98/2000/XP/2003 操作系统

4.2 开发环境

Microsoft Visual C++6.0

4.3 使用说明

4.3.1 板卡号选择

图 4-1 板卡号选择窗口

选择要操作的板卡号：范围为 0~7。

4.3.2 主窗口

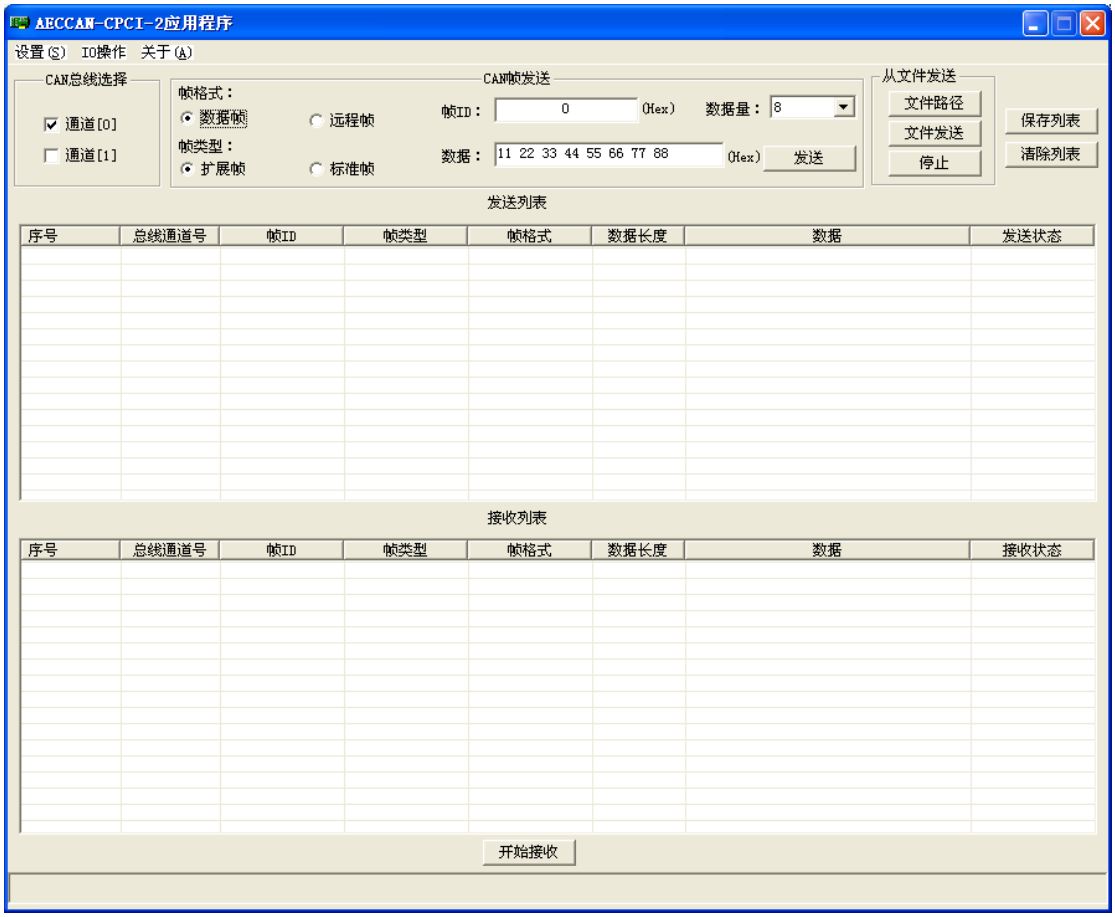


图 4-2 主窗口

- **接收：**

开始接收：开始接收数据后，若先前接收过数据，则将列表清空，按钮变成“停止接收”，并将数据显示在接收列表中

停止接收：停止接收数据。

- **发送：**

CAN 总线选择：通道[0]、通道[1]供选择，可以两路均选择，此时将会发送两路的数据

帧格式：数据帧/远程帧供选择

帧类型：扩展帧/标准帧供选择

帧 ID：一个 32 位数据，此处为 16 进制输入

数据：将要发送的数据填写在该文本框内，数据之间用空格分隔开，每个数据为 2 位的 16 进制。

数据量：0~8 可供选择

发送：发送一帧配置好的数据

文件路径：选择从文件发送的文件路径

文件发送：从文件发送若干帧数据

停止：停止正在进行的发送

保存列表：保存发送列表中的数据到文件，这样，以便下一次发送时可以从文件直接发送若干帧

清除列表：清除发送列表中的数据

4.3.3 板卡设置窗口

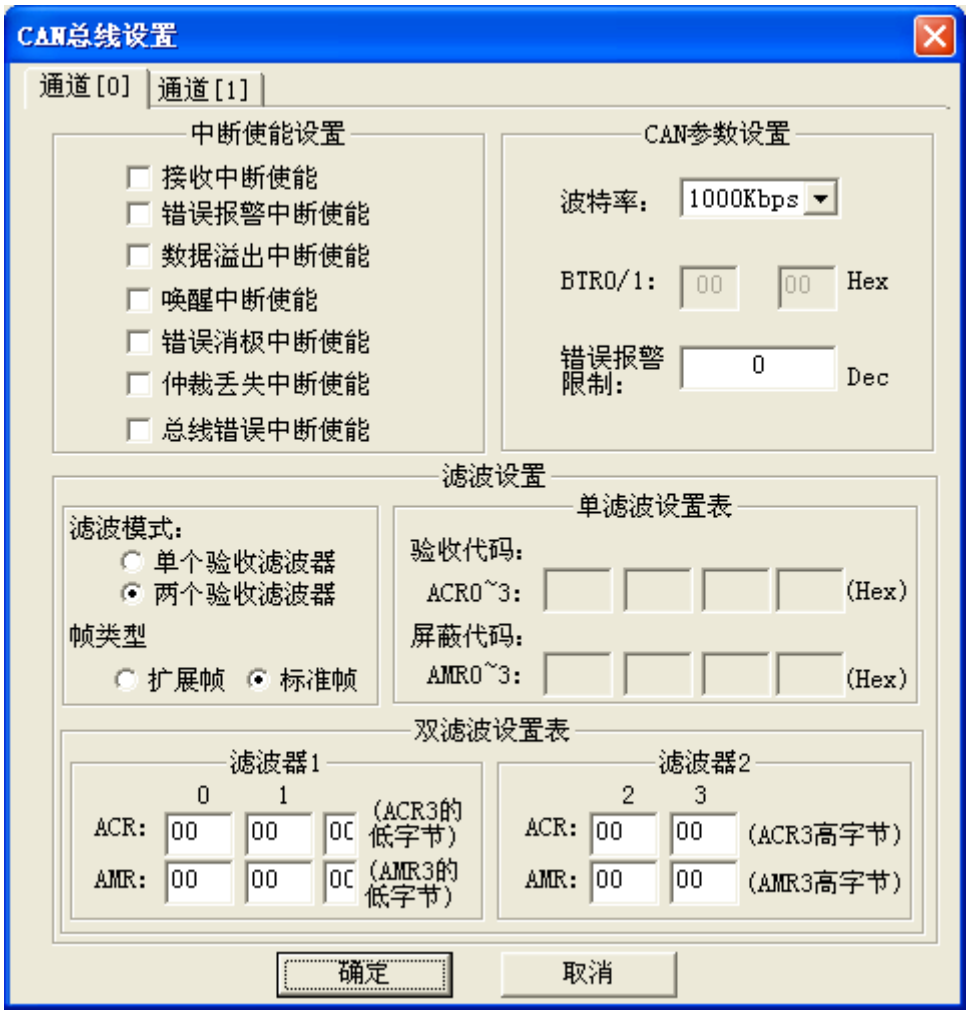


图 4-3 板卡设置窗口

- **中断使能设置：**

  中断大致分为两种：接收中断、报警类中断 (错误报警 / 数据溢出 / 唤醒中断 / 错误消极中断 / 仲裁丢失 / 总线错误)。

- **CAN 参数设置：**

  波特率：波特率有 10k、20k、50k、100k、125k、250k、500k、800k、1000k、自定义供选择，自定义通过设置 BTR0/1 来设置波特率。

  错误报警限制：错误报警限制在 0~255，二进制输入。

- **滤波设置：**

  滤波模式： 单个验收滤波器 / 两个验收滤波器供选择。

  帧类型：扩展帧 / 标准帧供选择。

  滤波模式与帧类型的选择共同决定滤波方式。

  滤波器设置均采用 16 进制

4.3.4 IO 操作窗口

图 4-4　IO 操作窗口

输入：

方向设置为输入，点击“输入”按钮。

输出：

方向设置为输出，设置各路的状态，默认为低电平，点击“输出”按钮。

附件 A　产品配件

为了满足您更精确的测量和测试要求，北京神州飞航科技有限责任公司为您提供如下产品配件供您选配：

- **连接电缆**　信号线采用优良的屏蔽双绞设计，将各种干扰降至最低，提高了信号质量，使您获得更精确的输入输出信号。
- **接线端子**　与连接电缆配套使用，方便灵活的接线方式，倍感轻松的同时更提高了工作效率。

具体订货信息如下：

序号	产品编号	产品描述
1	AECLJ-37	DB37 接线端子
2	AECLJ-62	DB62 接线端子
3	AECLJ-68	SCSI68 接线端子
4	AECLJ-100	SCSI100 接线端子
5	AECCB-3701	DB37 电缆，1 米
6	AECCB-3702	DB37 电缆，2 米
7	AECCB-6201	DB62 电缆，1 米
8	AECCB-6202	DB62 电缆，2 米
9	AECCB-6801	SCSI68 电缆，1 米
10	AECCB-6802	SCSI68 电缆，2 米
11	AECCB-10001	SCSI100 电缆，1 米
12	AECCB-10002	SCSI100 电缆，2 米

附件 B　公司介绍

北京神州飞航科技有限责任公司感谢您选用 AEC 系列产品！

北京神州飞航科技有限责任公司是专业从事嵌入式计算机领域相关产品研发、生产和服务的高科技企业。公司采用电子、计算机软硬件技术，为国内军工企业和科研院所提供电子产品以及相关的技术支持和服务。公司经过多年的电子产品研发，积累了丰富的经验，为航空、航天、兵器、核工业、电子、部队、科研院所等行业提供了不同应用功能的产品，并得到了客户的肯定。

经营理念

自主研发,专业服务,客户至上

主营业务

神州飞航公司产品线包括 ARINC-429、MIL-STD-1553、RS232/422/485、同步器/旋变、可逆计数器、定制产品等六大类。在产品的研发设计上，采用公司自主设计的核心芯片，使产品灵活性大大加强，也为今后的技术服务打下了很好的基础。基于专业化公司的经营理念，神州飞航公司致力于向客户提供全面的通用总线接口解决方案，所以我们的 AEC 系列产品硬件平台包括了几乎所有的通用计算机总线和接口： ISA，PCI，CPCI/PXI，PCMCIA，PC104，PMC 等，并且可支持通用的操作系统和开发语言平台，方便客户的后续开发工作。

质量体系

2005 年 12 月，公司通过 GJB9001A-2001 质量管理体系认证。

企业文化

求真务实：技术第一，踏踏实实地做好本职工作，态度严谨，产品适用、实用。

开拓创新：积极进取、勇于开拓，追求卓越，不断地学习，与时俱进。

真诚服务：建立以客户为中心的服务理念，不断提高企业对客户的服务意识。

公司联系信息

地址：北京市海淀区西三环北路 21 号北控久凌大厦北楼 1009 室

邮编：100089

电话：010-68403305，68403306，68403307，68403308

传真：010-68403309

E-mail：sales@aectech.cn

网址：www.aectech.cn