

PCI/CPCI 1553B 通信板卡

CLV-3051/4051 V2.2 用户手册



成都科洛威尔科技有限公司

地址：成都市高新区双柏路68号23栋

TEL: 1878-0222-336 191-3621-6517

EMAIL: clovertech@163.com 公司官网: www.clvtech.net

目 录

版本历史	4
用户手册内容简介	5
1 CLV3051/CLV4051—基于 PCI/CPCI 总线的 1553B 通信板卡概述	6
2 关于 1553B 总线	8
2.1 总线框架	8
2.2 双冗余	8
2.3 信号特性	9
2.4 RT 地址和子地址	9
2.5 数据格式	9
2.6 方式代码（模式码）	10
3 1553B 总线连接	11
3.1 变压器耦合连接方式	11
3.2 直接耦合连接方式	12
3.3 1553B 通信测试网络连接示例	12
4 板卡硬件安装	13
4.1 板卡安装	13
4.2 连接器定义	13
4.2.1 CLV-3051 V2.0/CLV-4051 前面板连接器定义	13
4.2.2 CLV-4051 后 IO 板连接器定义	14
4.3 驱动程序的安装	15
5 配套资料说明	22
6 实现原理	23
7 功能及应用	24
7.1 软件编程总体流程	24
7.2 设备自检	24
7.2.1 内部自检	24
7.2.2 外部（回环）自检	24
7.3 BC 功能	25
7.3.1 BC 相关函数接口	25
7.3.2 BC 功能初始化流程	25
7.3.3 BC 消息队列	25
7.3.4 BC 消息类型	26
7.3.4.1. 标准 1553B 消息	26
7.3.4.2. 无条件跳转消息	27
7.3.4.3. 条件跳转消息 1	27
7.3.4.4. 条件跳转消息 2	27
7.3.4.5. 空消息	27
7.3.4.6. 条件跳转及空消息应用示例	27
7.3.5 周期性消息与插入消息	31
7.3.6 单次消息	32

7.3.7 消息重试	33
7.3.8 消息数据更新	33
7.4 RT 功能	33
7.5 BM 功能	34
7.6 时间标签（消息时间戳）	34
7.6.1 时间标签计时范围及精度	34
7.6.2 时间标签初值及复位	34
7.6.3 时间标签在编程应用中的处理	35
7.7 消息读取方法	36
7.7.1 中断方式消息读取	36
7.7.2 查询方式消息读取	38
8 测试示例软件使用说明	38
8.1 综述	38
8.2 操作流程	39
8.3 板卡级设置	39
(1) 板卡初始化	39
(2) 耦合方式设置	40
(3) 自测试	40
8.4 多通道支持	41
8.5 BC 功能	41
8.5.1 Bc 参数设置	41
8.5.2 消息设置	42
8.5.3 插入消息发送示例	45
8.6 RT 功能	46
8.6.1 Rt 初始化	46
8.6.2 RT 参数设置	46
8.7 BM 设置	48
8.7.1 BM 初始化	48
8.7.2 BM 过滤设置	48
8.8 BC、RT、BM 功能的启动与停止	49
8.9 数据显示、存盘与查看	49
8.10 退出程序	50
9 产品选型说明	51

版本历史

版本	修订日期	内容
V1.0	2012 年 8 月	初始版本
V1.01	2013 年 10 月	第一次修订
V1.2	2015 年 7 月	添加函数: MC1553B_BC_GetFrameCount
V2.0	2021 年 10 月 19 日	细化了一些章节描述
V2.1	2023 年 2 月 28 日	增加第七章部分描述
V2.2	2024 年 1 月	更新第 8 章测试示例软件使用说明 合并 CLV-3051、CLV-4051 用户手册

用户手册内容简介

本手册分章节：

通信板卡概述

关于 1553B 总线

1553B 总线连接

板卡硬件安装

配套资料说明

实现原理

功能与应用

测试示例软件使用简介

产品选型说明

1 CLV3051/CLV4051—基于 PCI/CPCI 总线的 1553B 通信板卡概述

该系列产品基于标准 PCI/CPCI 总线接口, 可提供 1~4 通道标准 1553B 接口, 性能卓越, 可用于各种航电地面测试系统、外场检测设备、系统仿真等领域的 1553B 总线通信、测试、仿真。

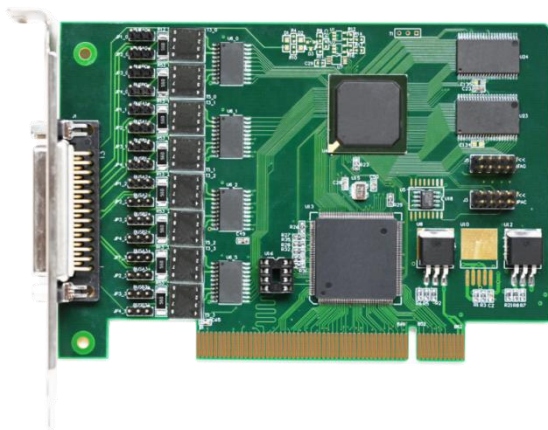


图 1 CLV-3051

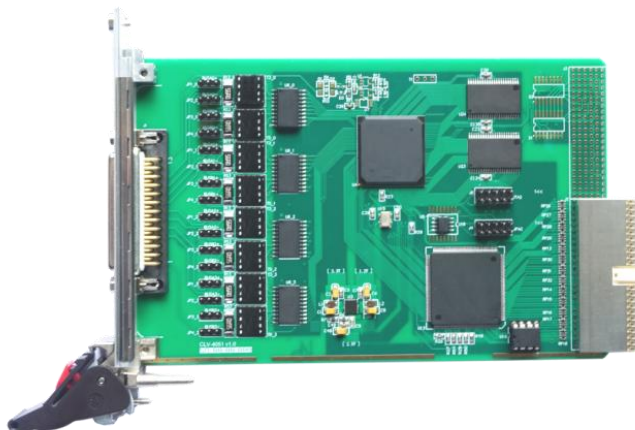


图 2 CLV-4051

板卡特点:

- 遵循 MIL-STD-1553B Notice2 (GJB289A-97) 规范
- 工业和宽温两种级别可选
- 1~4 个 MIL-STD-1553A/B 双冗余通道;
- 支持 Bus Control、Remote Terminal、Bus Monitor 终端模式。

- 变压器耦合、直接耦合可设置
- 支持自检：内部自检、外部（回环）自检
- BC 功能：

帧可编程，消息间隔可编程

突发消息发送

数据双 Buffer

消息重试

BUSA, BUSB 可编程

支持分支跳转消息：固定跳转、条件分支跳转

支持错误模拟注入

- RT 功能

RT 地址可软件设置

单数据 Buffer、双数据 Buffer 可编程

支持错误注入

- BM 功能

100%消息记录

消息过滤

- 32 位时间标签

平台特性：

- CLV3051：PCI Specification Version 2.1
CLV4051：PICMG2.0 Version 2.1
- 32 位/33MHZ
- 5-volt PCI signaling

机械尺寸：

- CLV3051：142mm x 100mm x 10mm
CLV4051：3U/4HP

环境特性：

- 操作温度：-20~+70°C
- 存储温度：-40~85°C

- 相对湿度：5%~95%(无凝结)

软件特性：

- 提供工程级源码示例程序；
- 支持 WINDOWS XP/win7 32/64bit/win10/linux 操作系统；
- 提供用户接口函数库,支持 Microsoft VC++，labview 等多种开发工具。

2 关于 1553B 总线

1553B 总线是一种数字命令/响应式多路数据复用总线，是一种广泛应用的飞机航电系统集成军用总线。

2.1 总线框架

1553B 总线网络中设备按照功能角色分为一下三种：

总线控制器，Bus Controller 简称 BC。BC 的功能是发起数据通信消息，进行总线管理。一个 1553B 网络中只允许有一个被激活的 BC，整个网络的通信内容及时序，如消息帧内容、帧发送速率在 BC 处进行规划和设置。

远程终端，Remote terminal 简称 RT。RT 设备是 1553B 网络中与应用业务相关的端设备，在机载系统中，一个功能单元可以被定义为一个 RT，比如说导航系统。RT 接收来自 BC 的指令，并作出响应，也即接收数据、回送数据并视通信情况返回状态字。

总线监视器，Bus Monitor,简称 BM。BM 不具有通信应答功能，专用于总线数据监视、记录。

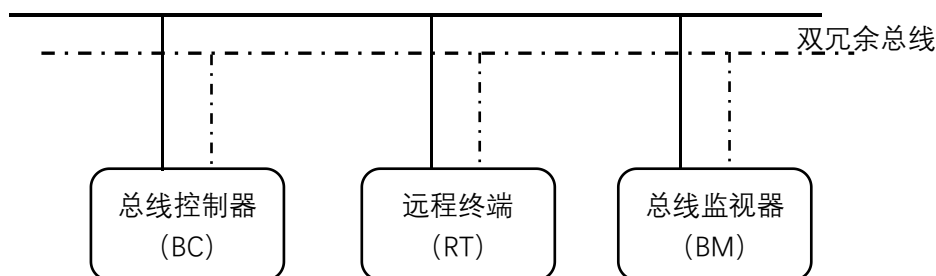


图 3 1553B 网络构成

2.2 双冗余

1553B 网络采用双冗余方式设计，一个 1553B 通信通道，包含 BUSA 和 BUSB 两路互为

备份的总线数据接口。正常状态下，只有一路总线接口（如 Bus A）处于工作状态下，当该路通信出现故障时，可以切换到备份总线上进行数据传输（Bus B），像 CLV3051/4051 这样的标准 1553B 通信接口卡是支持这种备份冗余方式的，这也极大的保证了通信的可靠性。且 CLV3051 的上述基于双冗余的总线的切换传输是可以可编程设置后自动执行的，相关设置方法请参见 CLV3051 软件手册。

2.3 信号特性

常用的 1553B 总线通信速率为 1Mbps，采用曼彻斯特编码方式。

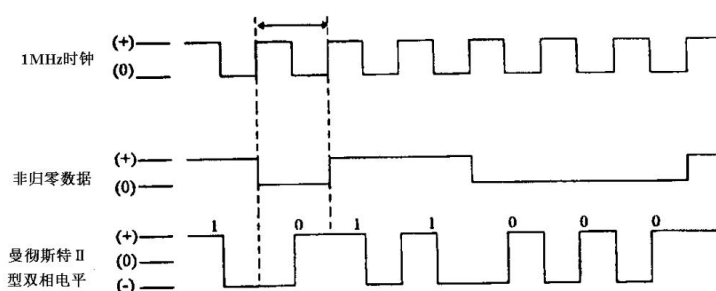


图 4 1553B 信号编码

1553B 总线数据传输时，由字组成消息，再由消息组成帧（见 2.5 数据格式）。每个字的传输波形由 3 个位时的同步头+16 个位时的有效位+1 位时校验位组成。

2.4 RT 地址和子地址

每个 RT 都有 RT 地址，在 1553B 总线规范中，RT 地址范围为 0~31，RT 地址 31 被用作广播地址，所以实际可用的 RT 地址为 0~30，但一般不推荐把 0 作为 RT 地址。RT 设备下，还可以划分 RT 子系统，子系统地址范围为 0~31，其中 0 和 31 号子地址用于模式码，所以实际能用于子系统地址区分的值为 1~30。在实际应用中，广播消息除外，如果 BC 要和一个 RT 设备进行通信，BC 端在编辑发送消息时，消息命令字中的 RT 地址和 RT 子地址必须和该 RT 的地址信息一致。

2.5 数据格式

在 1553B 网络上传输的数据是由一系列消息组成的帧。消息又由命令字、数据字、状态字组成。

命令字有时又被称为指令字，包含：远程终端地址（RT 地址）、发送接收标志、子地址/方式代码操作标识、数据字计数/方式代码。数据字计数指定本条消息发送的数据字个数，一条消息最多可以带 32 个数据字，CLV3051 在应用中约定，数据字计数可设置值为 0~31，数值 0 表示 32 个数据字发送。

数据字是 1553B 通信中需要被传输的数据，每个数据字的宽度为 16 位，也即一条消息最大数据传输容量为 16bit x 32。

状态字是 RT 在收到非广播消息时，根据 RT 本身及消息的接收处理状态，回送给 bC 的状态信息。状态字由远程终端地址字段、消息差错位、测试手段位、服务请求位、备用位、广播指令接收位、忙位、子系统标志位、动态总线控制接受位、终端标志位及奇偶校验位组成，每个标志位的具体说明请参见 1553B 通信协议标准。

对应于协议规范，CLV3051 在 API 应用层将命令字，数据字，状态字封装成结构体，方便用户进行二次开发，这些结构体的详细说明请参见《软件手册》及产品例程。

2.6 方式代码（模式码）

方式代码又称模式码，是 1553B 总线网络中的一类专用特殊指令，用于总线通信维护和管理。1553B 总线通信协议中，对方式代码的格式、内容和含义及用途有明确的规定，1553B 通信支持的方式代码见下表（摘自 GJB-289A）：

发/收位	方式代码	功 能	是否带数据字	是否允许广播指令
1	00000	动态总线控制	否	否
1	00001	同步	否	是
1	00010	发送上一状态字	否	否
1	00011	启动自测试	否	是
1	00100	发送器关闭	否	是
1	00101	取消发送器关闭	否	是
1	00110	禁止终端标志位	否	是
1	00111	取消禁止终端标志位	否	是
1	01000	复位远程终端	否	是
1	01001	备用	否	待定
...	...	...	...	...
1	01111	备用	否	待定
1	10000	发送矢量字	是	否
0	10001	同步	是	是
1	10010	发送上一指令字	是	否
1	10011	发送自检测字	是	否

0	10100	选定的发送器关闭	是	是
0	10101	取消选定的发送器关闭	是	是
1 或 0	10110	备用	是	待定
...	...	...	...	...
1 或 0	11111	备用	是	待定

表 1 方式代码

3 1553B 总线连接

一个 1553B 设备，如本手册所描述的 PCI 1553B 产品，CLV3051，在和其他 1553B 设备连接组网时，有两种连接方式：变压器耦合方式和直接耦合方式。顾名思义，变压器耦合方式需要有专用的耦合器实现设备到总线网络的接入，直接耦合则不需要。在机载环境及其他实际应用场景，直接耦合方式应该尽量避免。

3.1 变压器耦合连接方式

包括 CLV3051 在内的科洛威尔的 1553B 产品可以订购为变压器耦合或直接耦合。当使用更标准的变压器耦合模式时，需要使用耦合器设备。如总线 A，连接方法见下图所示；总线 B 的连接，需要另一个耦合器，CLV3051 的总线 B 通过耦合器连接到其他 1553B 设备的总线 B。这样的连接使得总线 A、B 分别有独立的信号传输路径，形成了 1553B 可靠的双冗余通信网络。

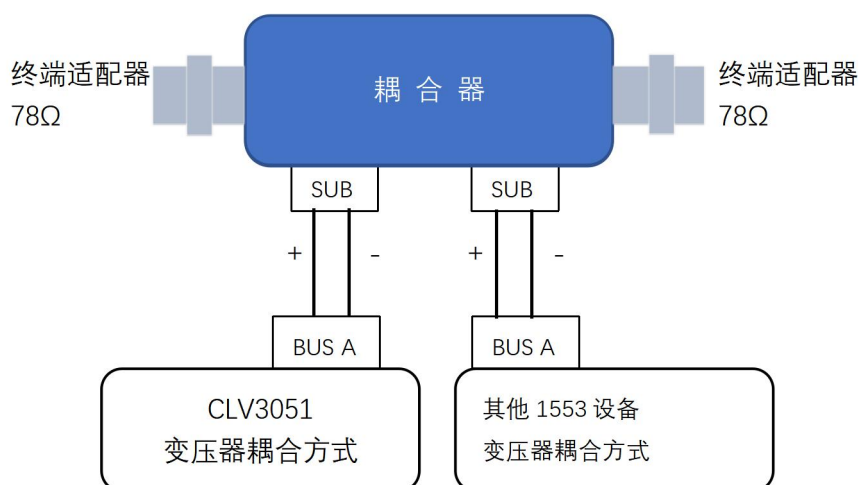


图 5 总线 A 变压器耦合方式连接

3.2 直接耦合连接方式

对于短距离通信，一个 1553B 设备可以直接耦合连接到另一个 1553B 设备。为确保正常通信，连接两个设备的电缆需要正确端接 78 欧姆电阻器。连接方式见图 4，总线 B 也通过同样的方式连接到另一个设备的总线 B。

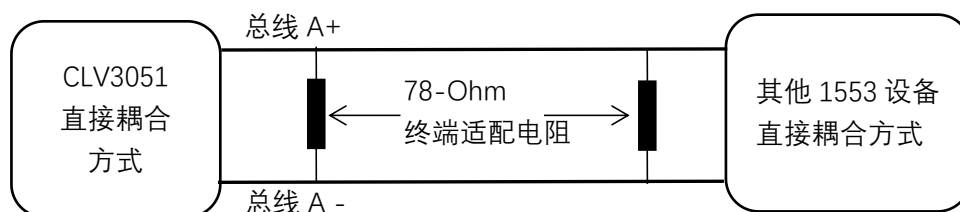


图 6 直接耦合方式设备连接

3.3 1553B 通信测试网络连接示例

利用 CLV3051/4051 板卡，可以快速方便的和被测设备（网络）对接，以进行数据记录、应答激励等测试任务，网络连接图如下：

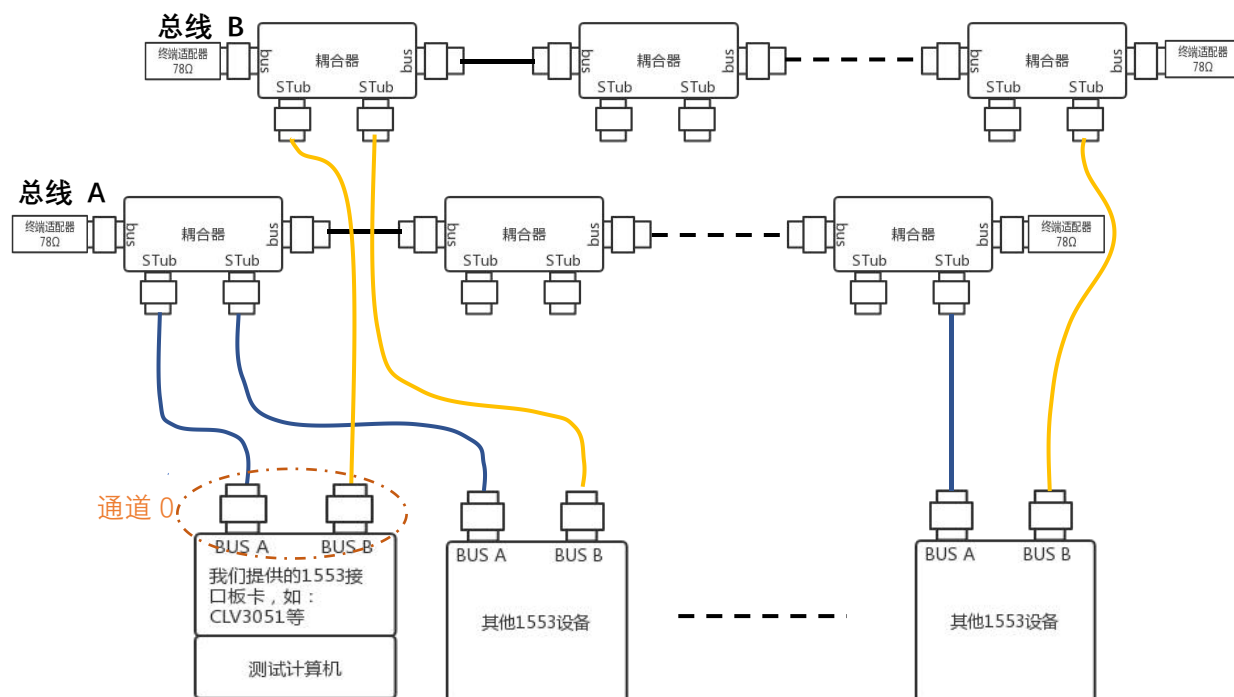


图 7 1553B 总线网络连接示意图

上图中用到的 1553B 组网设备 (配件) 有: CLV-3051/4051 标准板卡、双口盒式耦合器、终端适配器以及 1553B 专用线缆。如果您在搭建 1553B 测试仿真系统或者 1553B 应用组网时, 有任何疑问可联系我们的技术工程师, 为您针对具体应用提供网络连接图和详细配件配置清单。

4 板卡硬件安装

4.1 板卡安装

CLV-3051 适用于带 PCI 插槽的计算机;

CLV-4051 适用于带 3U CPCI 插槽的计算机。

安装前请将计算机系统断电, 然后将板卡插入插槽, 并用螺钉固定好即可。

4.2 连接器定义

4.2.1 CLV-3051 V2.0/CLV-4051 前面板连接器定义

CLV-3051 V2.0/CLV-4051 前面板提供了一个 DB-25 芯的连接器。管脚定义如下:

引脚号	定义	引脚号	定义
25	GND	13	IRIG-BI (保留)
24	GND	12	IRIG_GND(保留)
23	BUSB0+	11	BUSA0+
22	BUSB0-	10	BUSA0-
21	BUSB1+	9	BUSA1+
20	BUSB1-	8	BUSA1-
19	BUSB2+	7	BUSA2+
18	BUSB2-	6	BUSA2-
17	BUSB3+	5	BUSA3+
16	BUSB3-	4	BUSA3-
15	GND	3	NC

14	GND	2	ExTrg_IN1(保留)
		1	ExTrg_IN0(保留)

4.2.2 CLV-4051 后 IO 板连接器定义

CLV-4051 可支持后 IO 走线方式（选配）。CLV-4051 后 IO 板提供一个 DB37（孔）的连接器。信号定义如下：

引脚号	定义	引脚号	定义
14	BUSB0+	35	BUSA0+
33	BUSB0-	16	BUSA0-
3	BUSB1+	13	BUSA1+
21	BUSB1-	32	BUSA1-
25	BUSB2+	2	BUSA2+
6	BUSB2-	4	BUSA2-
29	BUSB3+	27	BUSA3+
10	BUSB3-	8	BUSA3-
34	IRIG-BI（保留）	31	ExTrg_IN1(保留)
17	IRIG_GND(保留)	12	ExTrg_IN0(保留)
1, 18, 19, 20	GND	其他	未定义

说明：其他未定义管脚，请保持悬空。

信号说明：

BUSAn+：通道 n,总线 A 信号正。

BUSAn-：通道 n,总线 A 信号负。

BUSBn+：通道 n,总线 B 信号正。

BUSBn-：通道 n,总线 B 信号负。

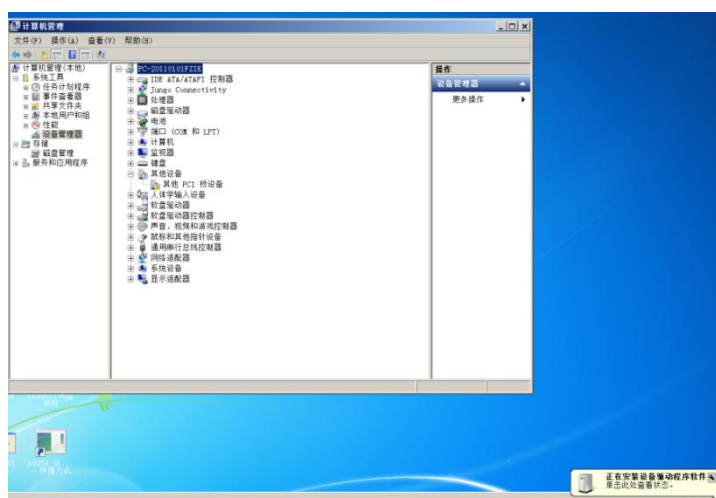
GND：信号地。

*如需配线，请提前联系公司销售人员，确认线缆配置要求。

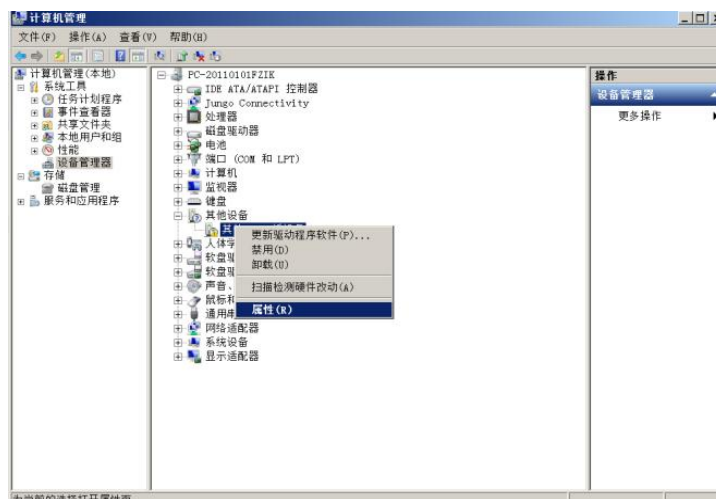
4.3 驱动程序的安装

如果是第一次使用本板卡，操作系统会提示发现新硬件，要求安装驱动。驱动程序在随产品配套的光盘上/drives 目录下, 根据所使用的计算机平台选择 32 位 (/drives/driverx86 目录下) 或者 64 位驱动 (/drives/driverx64 目录下)，按照系统提示选择安装即可。

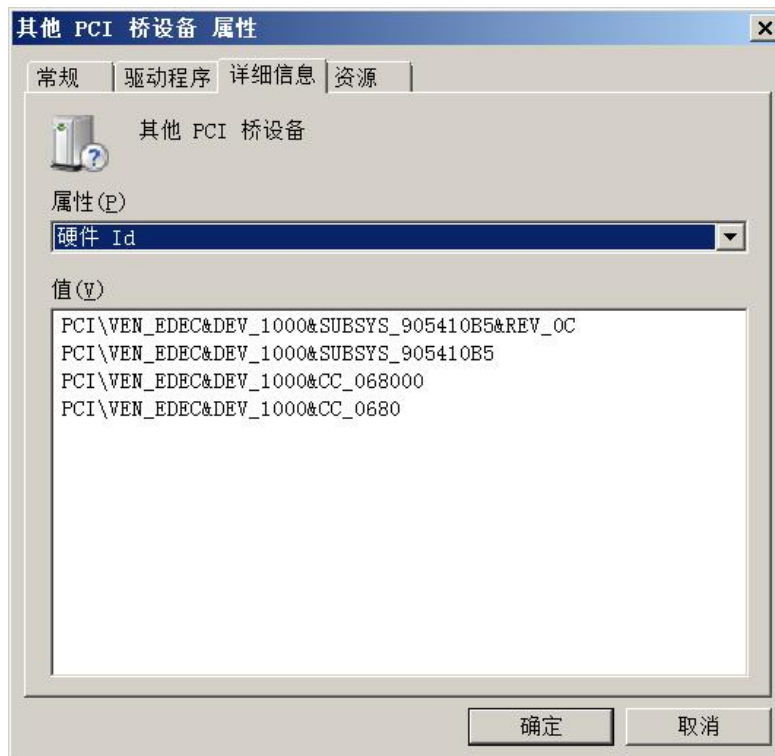
如果没有提示新设备，可以打开设备管理器（“我的电脑”图标，点击右键，管理->设备管理）：



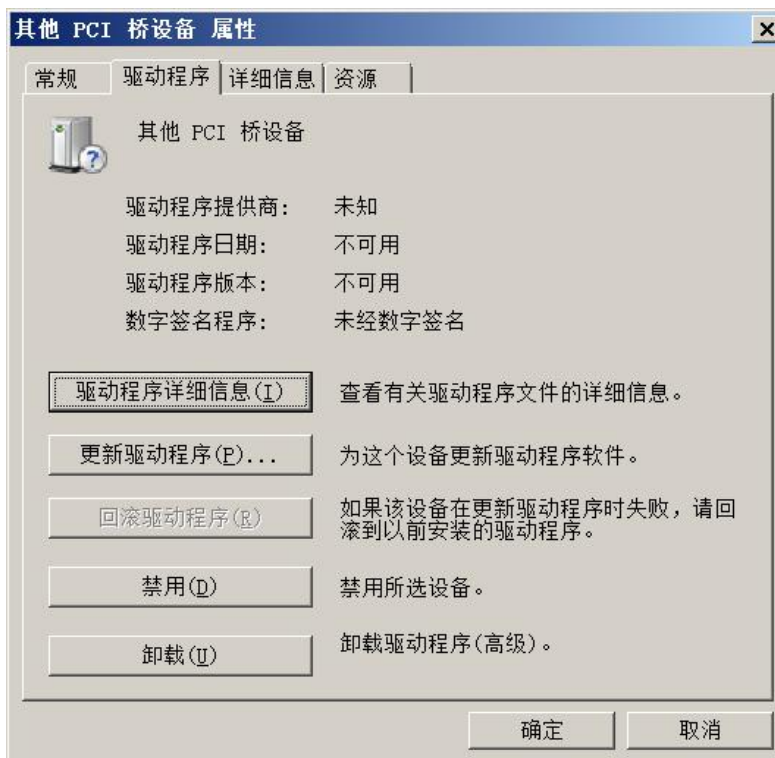
其他设备里有“其他 PCI 桥设备”，在其上点右键，选择“属性”。



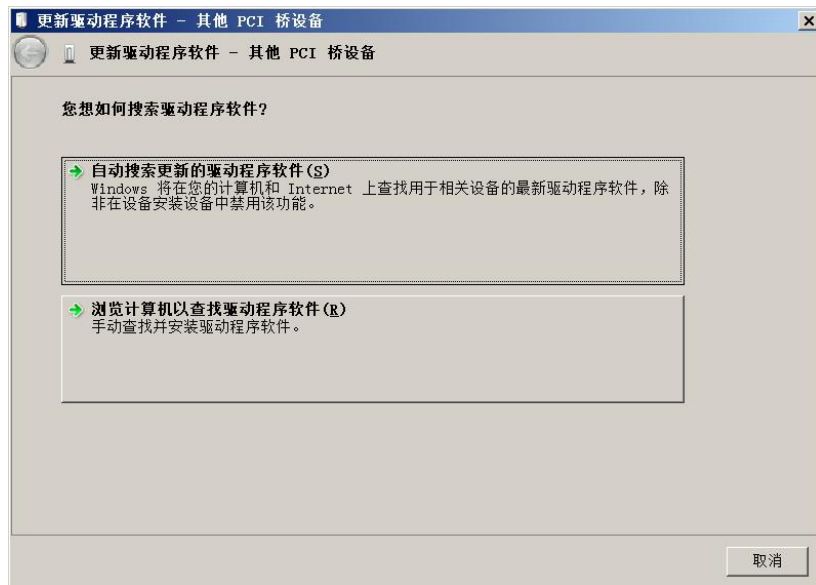
可以在弹出的属性面板，点击“详细信息”，查看板卡的硬件 ID，CLV3051 的硬件 ID 如下：



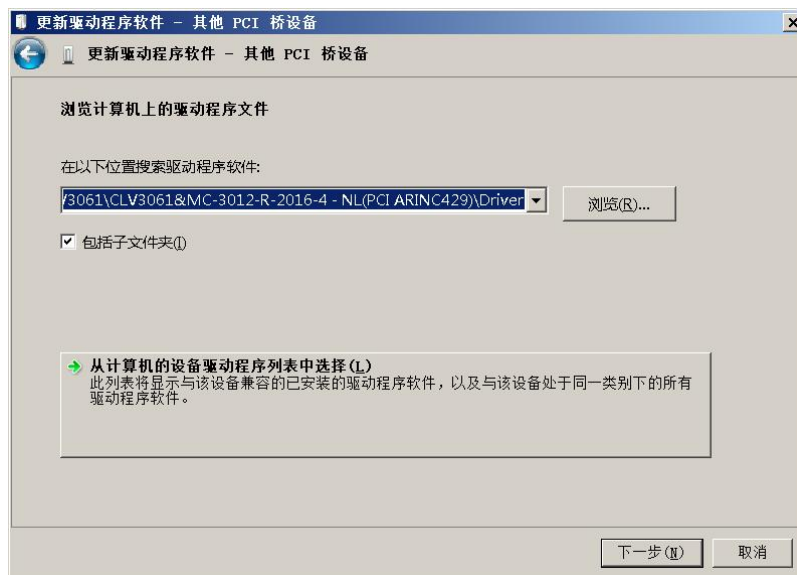
如果硬件 ID 完全正确，说明 CLV3051 与计算机之间的接插正确，可以继续安装驱动。
点击面板上的“驱动程序”，准备开始驱动按照：



点击“更新驱动程序...”,出现如下界面：



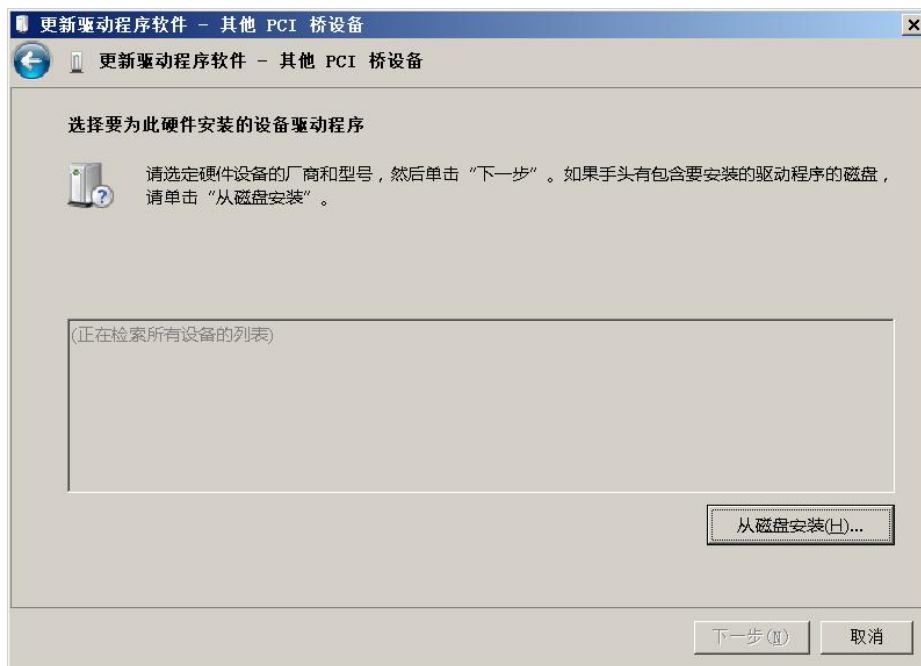
点击“浏览计算机以查找驱动程序软件”，出现如下界面：



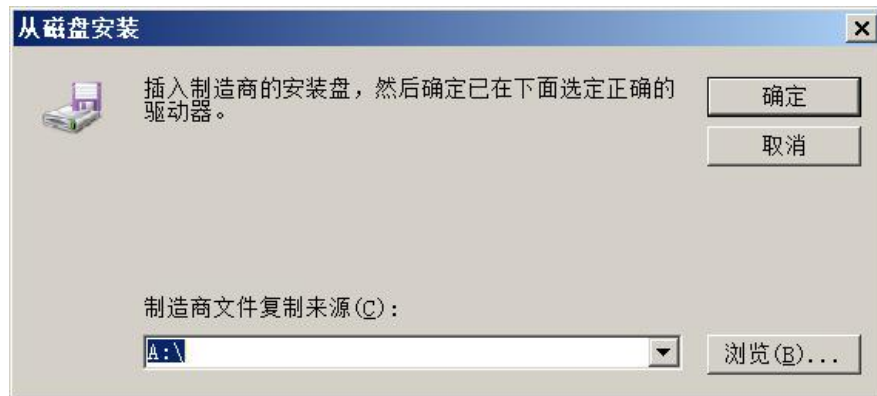
选择“从计算机的设备驱动程序列表中选择”，进入下一个界面：



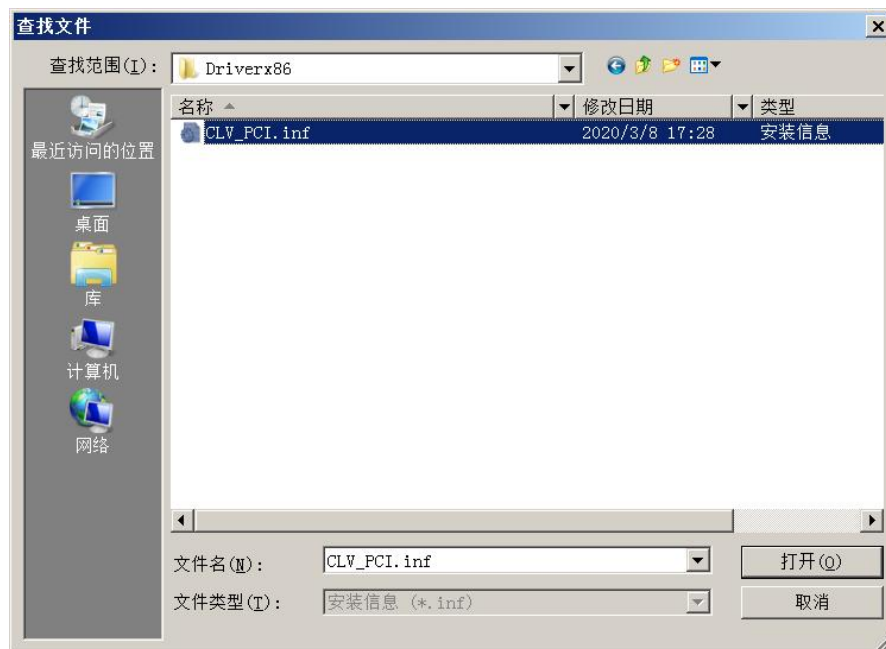
直接点击下一步：



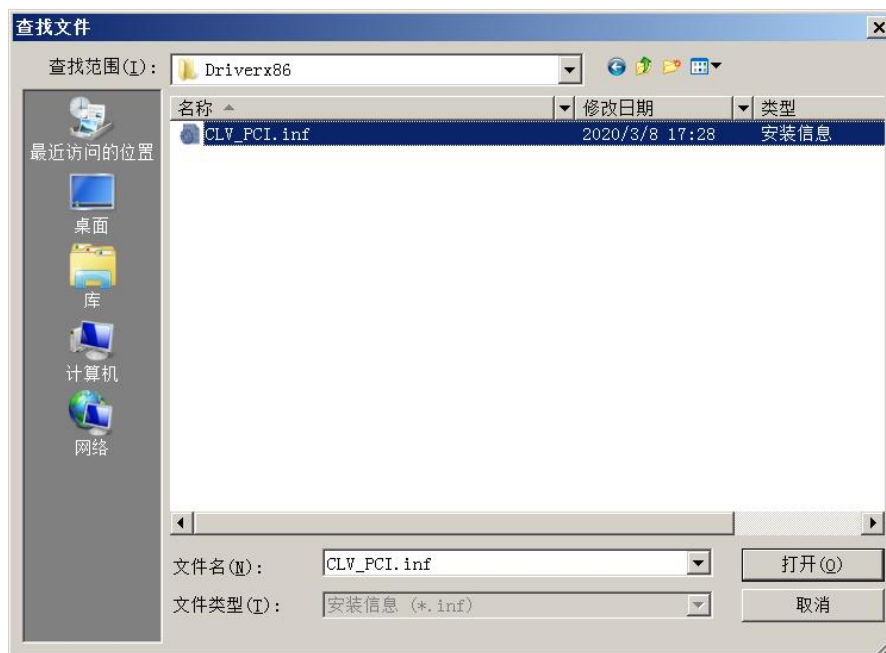
点击从“磁盘安装”按钮：



点击“浏览”，进入下一个界面：



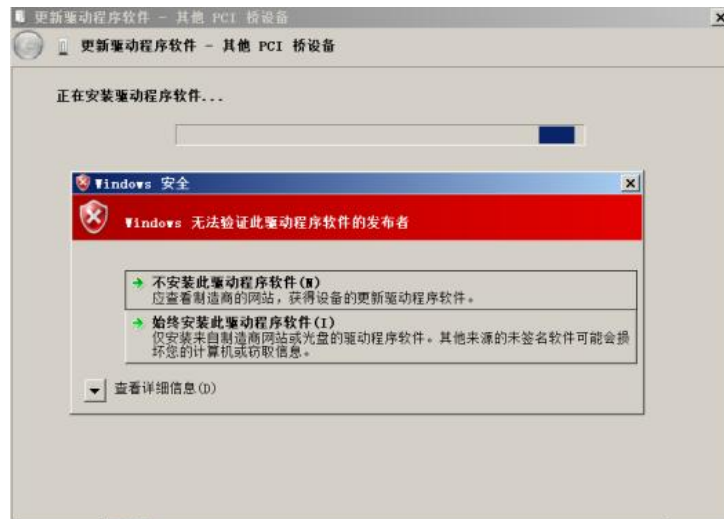
在打开窗口中，定位到配套光盘/Drivers 目录下，再根据当前操作系统，进入相应的文件夹选项，选取 inf 文件打开进行安装。



双击或者选择 inf 文件后点击“打开”，后会出现确定对话框，点击“确定”，出现如下界面：



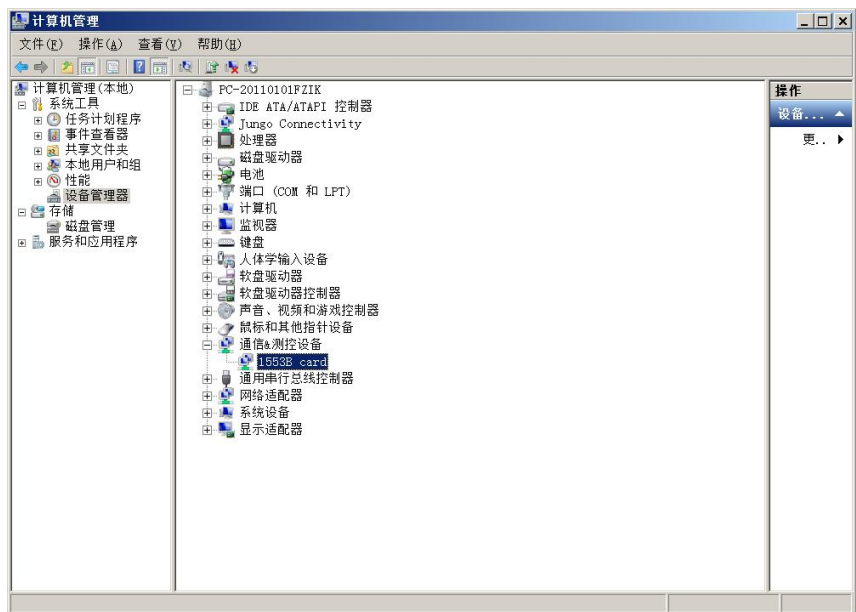
直接点击下一步，出现“驱动数字签名”提示信息，忽略，直接点“始终安装此驱动程序软件”：



系统将安装选定的驱动，安装完成后出现如下界面：



驱动安装完成，关闭窗口即可。此时设备管理器里的设备列表会被自动刷新。如下图所示，CLV3051 会被识别为“1553 card”设备：



5 配套资料说明

产品配有二次开发支持包，驱动程序、API 函数库、示例工程、以及相关文档。二次开发包以电子文档或者光盘的形式提供。

序号	文件目录	说明
1	/api	API 函数库动态库文件
2	/demo	示例程序工程（开发环境 labwindows CVI 2017）
3	/demo_setup	示例程序安装文件
4	/doc	文档手册
5	/drivers	驱动程序

6 实现原理

CLV3051 采用了科洛威尔公司自主研制的 1553B 通信 FPGA IP 核心，该 IP 核心已经在多个领域，客户应用场景中得到广泛应用，性能稳定可靠。整个 CLV3051/4051 板卡的功能框图如下：

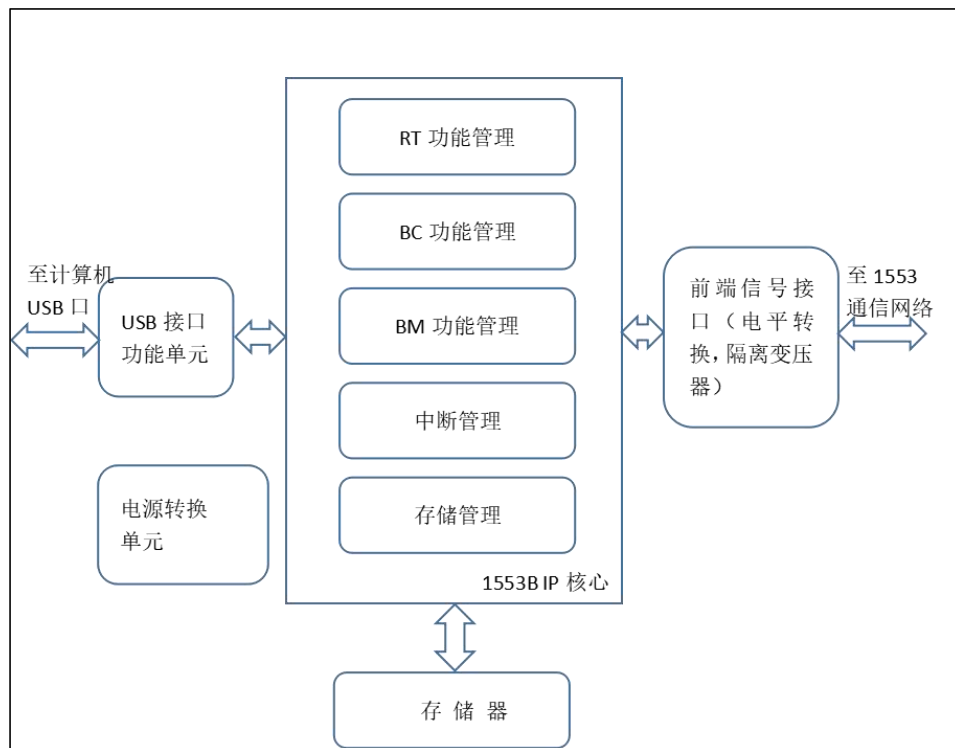


图 8 功能逻辑框图

7 功能及应用

7.1 软件编程总体流程

CLV3051 板卡在软件接口的实现上也分 BC、RT、BM 三个功能块，软件总体流程如下：

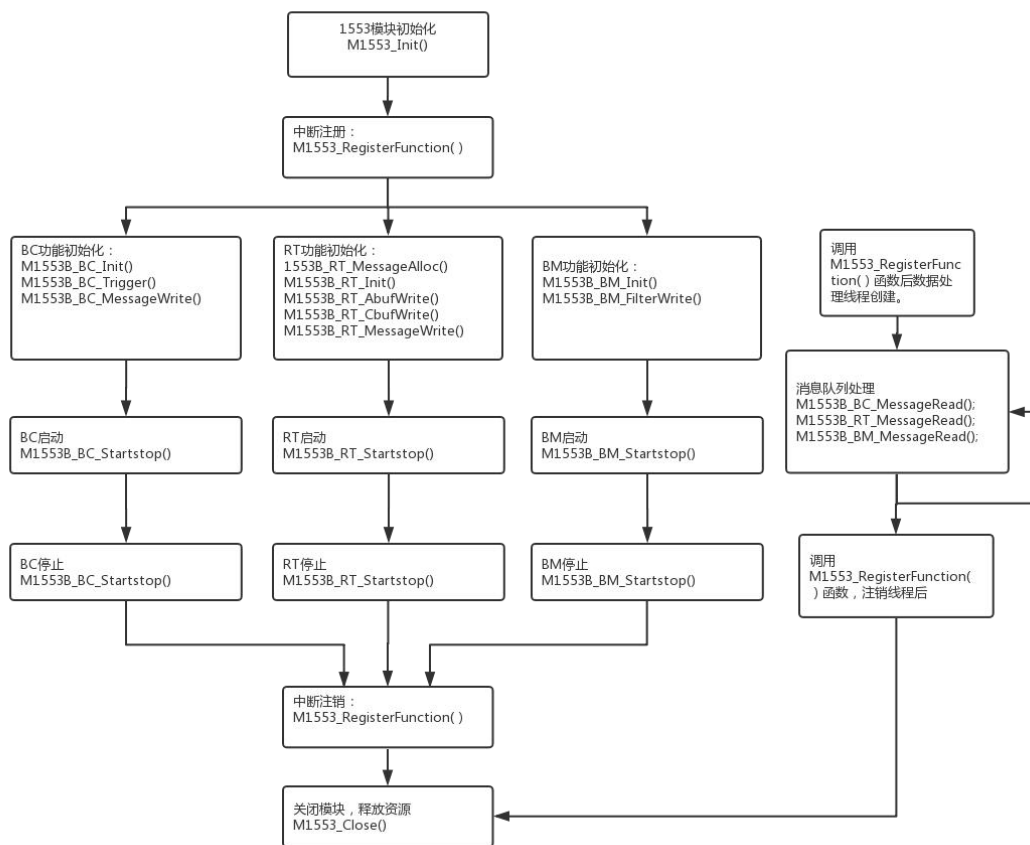


图 9 1553B 通信接口编程流程

7.2 设备自检

板卡支持设备自检功能，可以对板卡自身状况进行检查。自检分内部自检和外部（回环）自检。如果正在执行的其他通信任务，执行自检前需先停止这些任务；自检完成后，板卡处于初始状态，需重新配置，才能启动相应功能。

7.2.1 内部自检

内部自检无需外部接线，可以对板卡上除前端收发器、连接器之外的数据处理及控制部分进行自检。可以调用 M1553B_BIT_InternalBit()函数启动内部自检。

7.2.2 外部（回环）自检

外部（回环）自检，需要将通道的 BUS A 和 BUS B 按直接耦合方式短接，再进行操作。外部（回环）自检，可以对板卡上 1553B 信号所有处理控制回路进行检查。可以调用

M1553B_BIT_CableWrap()函数启动外部自检。

7.3 BC 功能

7.3.1 BC 相关函数接口

BC 功能是 1553B 网络的管理功能单元，是 1553B 网络上所有消息的发起者和管理者。BC 主要实现消息内容编辑与解析（如命令字、数据字）、消息帧组织（如消息间隔、帧周期）、消息状态识别、发送控制（如启动/停止发送、突发消息插入）。

BC 功能相关函数（相关详细说明请参见软件手册）：

M1553B_BC_Init	BC 功能初始化。
M1553B_BC_AperiodicRun	突发消息（非周期消息）发送。
M1553B_BC_RetryInit	重试方式初始化。
M1553B_BC_IsRunning	BC 运行状态测试。
M1553B_BC_MessageNoop	空消息设置。
M1553B_BC_MessageRead	BC 消息读取。
M1553B_BC_MessageReadData	BC 消息数据读取。
M1553B_BC_MessageWrite	写入 BC 消息。
M1553B_BC_OneShotExecute	OneShot 消息发送。
M1553B_BC_OneShotInit	OneShot 模式初始化。
M1553B_BC_Start	启动 BC 功能。
M1553B_BC_StartStop	启动/停止 BC 功能。
M1553B_BC_Trigger	BC 触发方式设置。
M1553B_BC_ReadNextMessage	查询读取 BC 消息。
M1553B_BC_ReadLastMessage	查询读取最新的 BC 消息。
M1553B_BC_ReadLastMessageBlock	查询读取最新的 BC 消息块。
M1553B_BC_GetFrameCount	读取 BC 已发送帧数。

7.3.2 BC 功能初始化流程

- (1) M1553B_BC_Init()初始化 BC 参数：重试、无响应时间、延迟响应时间、帧周期。
- (2) M1553B_BC_Trigger()设置消息触发方式。
- (3) M1553B_BC_MessageWrite()写入发送消息。
- (4) M1553B_BC_StartStop()启动/停止 BC 功能。

7.3.3 BC 消息队列

BC 消息队列，也即在一个 1553B 网络中要发送的消息。一条消息是 1553B 一次命令响应式通信的信息实体；消息队列由 1 到多条消息组成，它们被有序的存储在硬件存储区中。在软件编程层面上，它是一个消息链表，而且是一个双向链表。我们初始化一个消息队列，也就是初始化一条消息，并将这个消息作为一个节点有序的写入，最终形成一个消息链表。

这样的消息节点，用一个结构体 API_BC_MBUF（具体描述可以参见软件手册）来进行完整的表示，并可以调用 M1553B_BC_MessageWrite（）函数来完成消息写入动作。

API_BC_MBUF 结构体中的 messno 项即消息编号，消息编号从 0 开始，并依次递增，消息号不允许重复或者不连续编号。API_BC_MBUF 结构里的另一个参数 messno_next，用于指定下一条消息的编号，也是依次递增的，但当周期性消息结束时，messno_next 应赋值为 0xffff。周期性消息结束后如果有非周期性消息，可以继续写入，messno 继续往后递增。例如，编辑 10 条消息用于通信，其中 0~7 条是周期性消息；8，9 条是非周期消息；消息编号应该如下所示进行赋值：



图 10 1553B 消息队列

7.3.4 BC 消息类型

BC 端的消息从功能上分有以下几种：

- (1) 标准的 1553B 消息：主要用于数据传输。
- (2) 分支跳转消息：用于消息链执行路径控制。
- (3) 条件跳转消息 1：用于消息链执行路径控制。
- (4) 条件跳转消息 2：用于消息链执行路径控制。
- (5) 空消息：用于消息链结构维护。

要定义这些消息，可以在写入消息时，在其消息结构体的控制字 (API_BC_MBUF.control) 里，使能相应标志位。

7.3.4.1. 标准 1553B 消息

定义一个标准 1553B 消息，需要在其控制字设置标志：BC_CONTROL_MESSAGE。如定义一条周期性消息，从 RT 地址 2，子地址 5，读取 10 各数据，代码如下：

```
API_BC_MBUF    bc_msg;
bc_msg.control =BC_CONTROL_MESSAGE;
bc_msg.control |=BC_CONTROL_CHANNELA;
bc_msg.control |=BC_CONTROL_BUFFERA;
bc_msg.control |=BC_CONTROL_MFRAME_BEG;
```

```
bc_msg.control |= BC_CONTROL_MFRAME_END;
bc_msg.errorid = 0;
bc_msg.gap_time = 50;
bc_msg.messno = 0;
bc_msg.messno_next = 0xffff;
bc_msg.mess_command1.rtaddr = 2;
bc_msg.mess_command1.tran_rec = 1;
bc_msg.mess_command1.subaddr = 5;
bc_msg.mess_command1.wcount = 10;
```

```
hr = M1553B_BC_MessageWrite(
    &DevInf1,
    chn,
    bc_msg.messno,
    &bc_msg);
```

7.3.4.2. 无条件跳转消息

无条件跳转消息，控制字标志为：BC_CONTROL_BRANCH。当消息链执行到无条件跳转消息，将直接跳转到指定消息号（由 API_BC_MBUF 消息结构体的 messno_branch 域设置）继续执行。

7.3.4.3. 条件跳转消息 1

条件跳转消息 1，控制字标志为：BC_CONTROL_CONDITION。当消息链执行到条件跳转消息，将与“上一条”消息执行结果进行比较，如果结果为真，跳转到指定消息号（由 API_BC_MBUF 消息结构体的 messno_branch 域设置）继续执行；结果为假，则继续执行下一条消息。

7.3.4.4. 条件跳转消息 2

条件跳转消息 2，控制字标志为：BC_CONTROL_CONDITION3。当消息链执行到条件跳转消息，将与指定消息执行结果进行比较，如果结果为真，跳转到指定消息号（由 API_BC_MBUF 消息结构体的 messno_branch 域设置）继续执行；结果为假，则继续执行下一条消息。

7.3.4.5. 空消息

空消息占用一个消息存储位置，但不发起任何通信动作，也不占用总线时间，可以用于到下一个执行节点（消息）的过度。空消息的控制字标志为：BC_CONTROL_NOP。

7.3.4.6. 条件跳转及空消息应用示例

以下代码初始化写入了 6 条消息。如下图所示，0~3 号消息+4 号空消息构成一帧周期性消息。3 号消息为一条 BC_CONTROL_CONDITION3 类型跳转消息，5 号消息视 3 号消息

跳转条件成立情况执行。

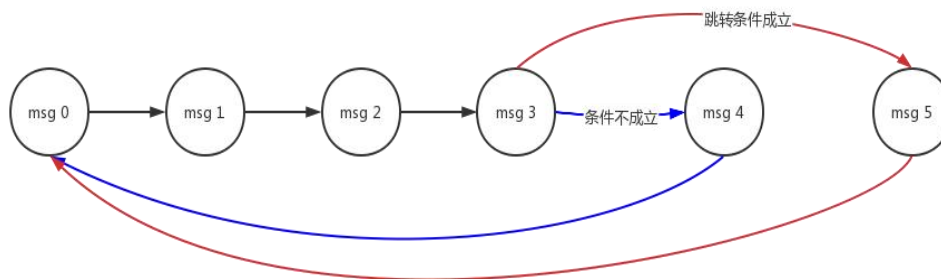


图 11 含跳转消息的消息链

```

int k=0;
API_BC_MBUF    bcMsg; //消息结构体，请参见《软件手册》数据结构章节。
int ch=0;
//-msg 0-----
//查询矢量字
memset(&bcMsg,0,sizeof(API_BC_MBUF));
bcMsg.control =BC_CONTROL_MESSAGE;
bcMsg.control |=BC_CONTROL_CHANNELA;
bcMsg.control |=BC_CONTROL_MFRAME_BEG;
bcMsg.errorid =0 ;
bcMsg.gap_time = 50;
bcMsg.messno =0;
bcMsg.messno_next=1;
bcMsg.mess_command1.rtaddr=2;
bcMsg.mess_command1.tran_rec=1;
bcMsg.mess_command1.subaddr=0;
bcMsg.mess_command1.wcount=16;
M1553B_BC_MessageWrite(
    &DevInf1,
    ch,
    bcMsg.messno,
    &bcMsg);

//-msg 1-----
//rt->BC
memset(&bcMsg,0,sizeof(API_BC_MBUF));
bcMsg.control =BC_CONTROL_MESSAGE;
bcMsg.control |=BC_CONTROL_CHANNELA;
bcMsg.errorid =0 ;
  
```

```
bcMsg.gap_time = 50;
bcMsg.messno =1;
bcMsg.messno_next=2;
bcMsg.mess_command1.rtaddr=2;
bcMsg.mess_command1.tran_rec=1;
bcMsg.mess_command1.subaddr=3;
bcMsg.mess_command1.wcount=0;
M1553B_BC_MessageWrite(
    &DevInf1,
    ch,
    bcMsg.messno,
    &bcMsg);

// -2-----
// BC->RT
memset(&bcMsg,0,sizeof(API_BC_MBUF));
bcMsg.control =BC_CONTROL_MESSAGE;
bcMsg.control |=BC_CONTROL_CHANNELA;
bcMsg.errorid =0 ;
bcMsg.gap_time = 50;
bcMsg.messno =2;
bcMsg.messno_next=3;
bcMsg.mess_command1.rtaddr=2;
bcMsg.mess_command1.tran_rec=0;
bcMsg.mess_command1.subaddr=5;
bcMsg.mess_command1.wcount=0;
for (k=0;k<32;k++)
    bcMsg.data [0][k]=0x0205;

M1553B_BC_MessageWrite(
    &DevInf1,
    ch,
    bcMsg.messno,
    &bcMsg);

// -msg 3-----
// CONDITION3 条件跳转
// 比较结果为真， 跳转到 messno=5； 为假则继续执行 messno=4;
// 判断条件为：
// ( bcMsg.address 所标示字段的值 AND bcMsg.data_mask == bcMsg.data_value )
memset(&bcMsg,0,sizeof(API_BC_MBUF));
bcMsg.control =BC_CONTROL_CONDITION3;// 条件跳转消息
bcMsg.control |=BC_CONTROL_CHANNELA;
bcMsg.errorid =0 ;
bcMsg.gap_time = 50;
```

```

bcMsg.messno =3;
bcMsg.messno_next=4;
bcMsg.mess_command1.rtaddr=2;
bcMsg.mess_command1.tran_rec=0;
bcMsg.mess_command1.subaddr=0;
bcMsg.mess_command1.wcount=0;
bcMsg.data_value=0x8100;    //跳转生效比较值
bcMsg.data_mask=0xffff;    //比较掩码
//比较字段。
//address=4 表示被比较消息执行结果的第一个数据字，这里也即消息 1 的第一个数据字
//data[0][0]。
bcMsg.address=4;
bcMsg.messno_branch=5;      //比较结果为真时，下一条执行的消息号
bcMsg.messno_compare=1;    //用于比较的消息号
M1553B_BC_MessageWrite(
    &DevInf1,
    ch,
    bcMsg.messno,
    &bcMsg);

// -4-----
//空消息结束
//用空消息作个占位，加上帧结束标记：BC_CONTROL_MFRAME_END，消息结束标记：
//bcMsg.messno_next=0xffff; 形成完整的周期消息链。
memset(&bcMsg,0,sizeof(API_BC_MBUF));
bcMsg.control =BC_CONTROL_NOP;      //空消息，不会发送到总线上
bcMsg.control |=BC_CONTROL_CHANNELA;
bcMsg.control |=BC_CONTROL_MFRAME_END;
bcMsg.errorid =0 ;
bcMsg.gap_time = 50;
bcMsg.messno =4;
bcMsg.messno_next=0xffff;
bcMsg.mess_command1.rtaddr=2;
bcMsg.mess_command1.tran_rec=1;
bcMsg.mess_command1.subaddr=2;
bcMsg.mess_command1.wcount=0;
hr=M1553B_BC_MessageWrite(
    &DevInf1,
    ch,
    bcMsg.messno,
    &bcMsg);

// -5-----
//取 RT2 SA2 的数据

```

//该消息在周期性消息序列之外，不会自动发送

```
memset(&bcMsg,0,sizeof(API_BC_MBUF));
bcMsg.control =BC_CONTROL_MESSAGE;
bcMsg.control |=BC_CONTROL_CHANNELA;
bcMsg.errorid =0 ;
bcMsg.gap_time = 50;
bcMsg.messno =5;
bcMsg.messno_next=0xffff;
bcMsg.mess_command1.rtaddr=2;
bcMsg.mess_command1.tran_rec=0;
bcMsg.mess_command1.subaddr=6;
bcMsg.mess_command1.wcount=0;
```

```
for (k=0;k<32;k++)
    bcMsg.data [0][k]=0x6666;
```

```
hr=M1553B_BC_MessageWrite(
    &DevInf1,
    ch,
    bcMsg.messno,
    &bcMsg);
```

7.3.5 周期性消息与插入消息

(1) 周期性消息

周期性消息是指按照设定周期循环发送的消息（消息序列）。周期消息的帧周期由 M1553B_BC_Init（）函数的 frame_period 参数决定，有效的帧周期为 100us 到 6.5535 s，详细说明可以参见“软件手册”章节中 M1553B_BC_Init（）函数的函数的说明。如“7.2.3 BC 消息队列”节所述设置好消息队列后，调用 M1553B_BC_StartStop()函数可以启动/停止周期性消息的发送（如果是外触发模式，还需要有外触发信号输入）。

(2) 插入消息

插入消息是用于突发性数据发送，这类消息可以插入到周期性发送中被执行。插入消息在当前正发送消息结束时被执行，且只执行一次。

如“7.2.3 BC 消息队列”节所述，编辑写入的消息队列里的任意一条消息都可以用于插入发送。需要执行插入发送操作时，即时调用以下代码段完成一次消息发送操作：

```
re=M1553B_BC_AperiodicRun(
    &DevInf1, //设备句柄
    ch,      //通道号
```

```

messno,    //消息号编号
0,         //优先级, 保留
0,         //等待标志, 保留
40);      //等待时间, 保留

```

也可以调用 M1553B_BC_MessageWrite() 函数, 新写入一条消息, 再调用 M1553B_BC_AperiodicRun () 函数将该消息插入发送, 但需要注意写入消息时消息号是否和已有消息队列中消息号重叠, 如果重叠将覆盖原有消息, 甚至改变、破坏原有的消息队列。

7.3.6 单次消息

单次消息模式, 所有的消息只发送一次。发送的消息可以是一条消息, 也可以是多条消息。与循环消息不同, 单次消息有一套专用的函数:

M1553B_BC_OneShotInit (), 初始化板卡处于单次消息发送模式;

M1553B_BC_OneShotExecute (), 执行单次发送消息;

需要注意的是, 单次消息执行完成, 不会触发消息结束中断, 消息执行结果的读取和处理也不在中断回调函数中处理。M1553B_BC_OneShotExecute () 函数的 pMessages 参数是一个消息结构指针, 单次发送的消息从该参数传入, 消息的执行结果也从该参数传出。所以, 如果需要保证通信的先后逻辑, 单次消息模式, 需要应用层软件插入等待时间, 等待 pMessages 内的消息执行结束, 再进行下一步操作。

BC 功能, 要么工作于周期性消息模式下, 要么工作于单次消息模式下, 这两种消息模式可以被软件配置随时切换, 但不能同时存在。单次发送模式相关代码如下:

```

M1553B_BC_OneShotInit(&DevInf1, chnum, channel, timeresp, timelate);

API_BC_MBUF bcmmessage[3];

消息结构初始化

....

WStatus=M1553B_BC_OneShotExecute(&DevInf1,   chnum,   wCount,   bcmmessage,
wWaitFlag, Timeout);

等待消息执行结束;

if(WStatus==STATUS_OK)

{

    处理执行完毕后的 bcmmessage;

}

```

7.3.7 消息重试

消息重试是指，一条消息在总线上发送失败时，再重新尝试发送。1553B 板卡支持消息的可编程自动重试发送。编程步骤如下：

- (1) 在 BC 初始化，M1553B_BC_Init () 函数的 wRetry 参数设置重试条件，重试条件可以是以下三种的任意组合：

BC_RETRY_ME 消息错；

BC_RETRY_NRSP 无响应；

BC_RETRY_BUSY 总线忙；

- (2) 调用 M1553B_BC_RetryInit () 函数，设置重试方式和重试次数：

原总线上重试；

切换总线重试；

结束重试。

- (3) 在写入消息时，定义消息重试使能，M1553B_BC_MessageWrite () 函数写入消息时，消息结构体，控制字中添加 BC_CONTROL_RETRY 宏开关，如：

API_BC_MBUF bcmmessage;

.....

bcmmessage.control |= BC_CONTROL_RETRY;

7.3.8 消息数据更新

1553B 消息除支持 M1553B_BC_MessageWrite() 函数写入外，还可以通过 M1553B_BC_MessageUpdate() 函数进行消息数据更新。M1553B_BC_MessageUpdate() 函数仅对消息的数据字进行更新，不会改写被更新消息的命令字，也不会改变消息队列的结构。

7.4 RT 功能

RT 作为 1553B 系统中消息被动执行与响应机构，需要配置 RT 地址、初始化 RT 数据缓冲区以及其他一些属性配置，功能相关函数如下：（相关详细说明请参见软件手册）

M1553B_RT_AbufRead	读取RT地址缓冲区设置。
M1553B_RT_AbufWrite	写RT地址缓冲区。
M1553B_RT_MessageAlloc	分配RT数据缓冲区。
M1553B_RT_CbufRead	读取RT控制缓冲区设置。
M1553B_RT_CbufWrite	写RT控制缓冲区设置。

M1553B_RT_Init	RT功能初始化。
M1553B_RT_MessageRead	读取RT消息。
M1553B_RT_MessageWrite	写RT发送数据。
M1553B_RT_StartStop	启动/停止RT功能。
M1553B_RT_ReadLastMessage	查询读取最新的RT消息。
M1553B_RT_ReadLastMessageBlock	查询读取最新的RT消息块。
M1553B_RT_ReadNextMessage	查询读取RT消息。

RT 功能按照以下步骤初始化：

- (1) M1553B_RT_MessageAlloc()函数分配 RT 数据缓冲区。
- (2) M1553B_RT_Init()初始化 RT 功能。
- (3) M1553B_RT_AbufWrite()函数设置总线 A/B 使能，动态总线控制接受允许，初始化状态字、命令字、自检字。
- (4) M1553B_RT_CbufWrite () 函数配置 RT 控制缓冲区，包括非法命令字、收/发缓冲区数量。
- (5) M1553B_RT_MessageWrite()函数初始化 RT 发送/接收缓冲区。

7.5 BM 功能

BM 功能按照以下步骤初始化：

- (1) M1553B_BM_Init()函数初始化 BC 功能。
- (2) M1553B_BM_FilterWrite()函数设置 BM 消息监视过滤表（视应用需求调用）。

7.6 时间标签（消息时间戳）

7.6.1 时间标签计时范围及精度

板卡具有消息时间标签功能，每条消息都带有时间标签。时间标签基于一个本地硬件时钟计数器，计数由 BC 或 RT 或 BM 功能启动而触发启动。时钟计数器的计数时间粒度可以设置，范围：2~64us（该功能保留，默认计数时间粒度为 64us）。BC 和 RT 功能的时间标签为 32 位，BM 的时间标签为 48 位。

7.6.2 时间标签初值及复位

板卡上电时，时间标签的初值为 0。修改时间标签的值有两种方式：

- (1) 应用程序直接调用 M1553B_TimeTagWrite () 函数写入。（此功能保留）。
- (2) RT 设备

当接收到同步模式码（00001）时，本地时间标签计数器复位清零。

当接收到同步模式码（10001）时，本地时间标签计数器赋值为模式码所携带的数据字。

7.6.3 时间标签在编程应用中的处理

(1) BC 消息时间标签

在消息读取后，立即调用 M1553B_TimeTagRead（）函数读取时间标签值，再调用 M1553B_TimeGetString（）函数，将其转化为“（x 天）：时：分：秒：微秒”格式。

如果需要，也可以在应用程序的任何位置，调用 M1553B_TimeTagRead（）函数读取当前时间计数器。

(2) RT 消息时间标签

RT 消息时间标签由板卡内核管理，在读取消息时，由消息结构体的 BT1553_TIME 结构一起带出，再调用 M1553B_TimeGetString（）函数进行格式转换。RT 消息时间标签获取处理流程如下：

```
API_RT_MBUF_READ    mbuf;

char databuf[200];

M1553B_RT_MessageRead(&DevInf1, chnum, rtaddr, subaddr,

                      transrec, messno, &mbuf);

M1553B_TimeGetString(&mbuf.time, databuf);
//显示或其他处理
```

(2) BM 消息的时间标签

BM 消息时间标签也是由板卡内核管理，在读取消息时，由消息结构体的 BT1553_TIME 结构一起带出，再调用 M1553B_TimeGetString（）函数进行格式转换。BM 消息时间标签获取处理流程如下：

```
API_BM_MBUF    bmbuf;

char databuf[200];

M1553B_BM_MessageRead(

                      &DevInf1,

                      chnum,

                      messno,

                      &bmbuf);
```

```
M1553B_TimeGetString(&bmbuf.time,databuf);
```

```
//显示或其他处理
```

7.7 消息读取方法

7.7.1 中断方式消息读取

1553B 板卡以队列的方式将待处理的消息进行缓存,并通过中断通知应用层,应用层在中断回调函数(线程)中对消息进行读取处理。注册/注销中断回调函数(线程)由 M1553_RegisterFunction()函数实现。

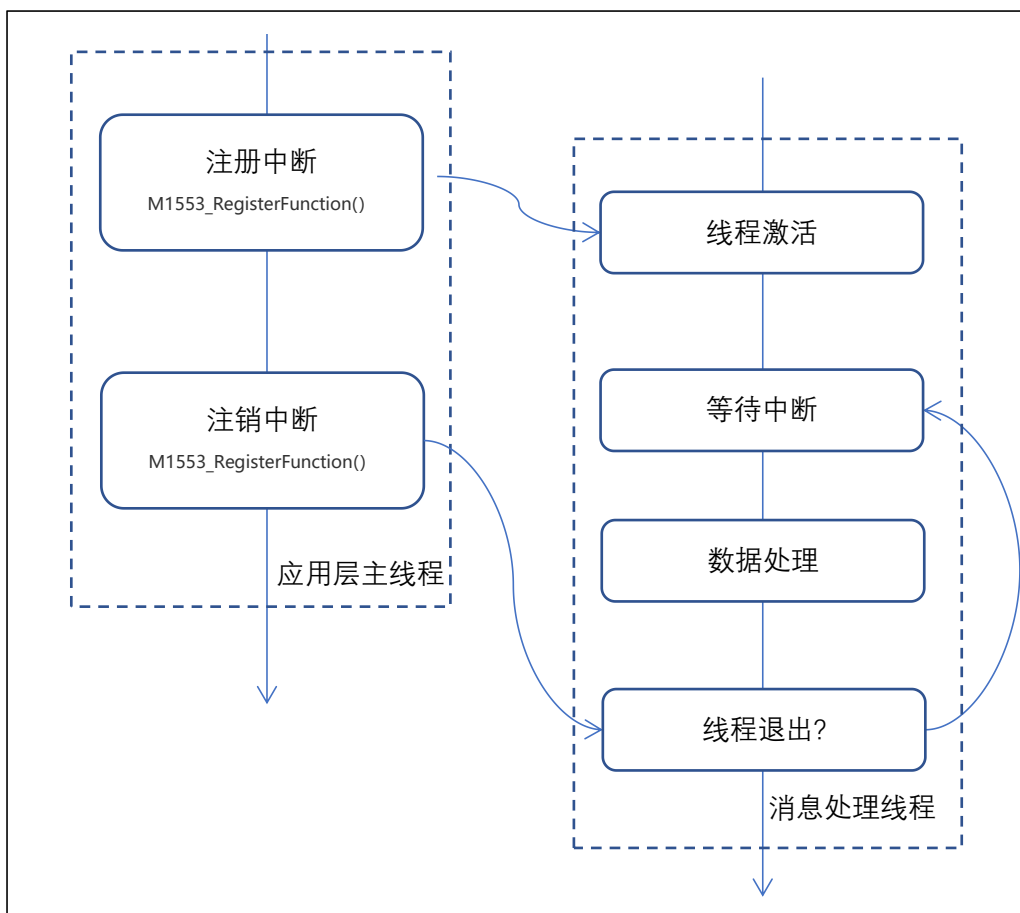


图 12 消息处理线程

每条消息可以按照“事件属性”,分为 BC 消息、RT 消息、BM 消息。这个事件属性跟 1553B 设备在接入的 1553B 通信网络中充当的角色有关,比如本次测试中 1553B 设备作为 BC 使用,触发中断的便是 BC 消息;1553B 设备作为 RT 使用时,触发中断的便是 RT 类型的消息;以此类推,便是 BM 消息。所以,在中断回调函数里,这三类消息在代码上是分为三个分支进行处理的,中断回调函数代码框架被设计如下:

```
INT32_stdcall DataRecProcess(UINT16 chnum,struct api_int_fifo *sIntFIFO)
{
```

```
.....

tail = slntFIFO->tail_index;

while ( tail != slntFIFO->head_index )

{

    messno = slntFIFO->fifo[tail].bufferID;

    if (slntFIFO->fifo[tail].event_type==EVENT_BC_MESSAGE)

    {

        hr=M1553B_BC_MessageRead(

            slntFIFO->hDevData,

            chnum,

            messno,

            &api_message);

        //在此处添加 BC 数据处理代码

        .....

    }

    else if (slntFIFO->fifo[tail].event_type==EVENT_RT_MESSAGE)

    {

        M1553B_RT_MessageRead(slntFIFO->hDevData,chnum,rtaddr,subaddr,

            transrec,messno,&mbuf);

        //此处添加 RT 数据处理代码

        .....

    }

    else if (slntFIFO->fifo[tail].event_type==EVENT_BM_MESSAGE)

    {

        messno = slntFIFO->fifo[tail].bufferID;

        M1553B_BM_MessageRead(

            slntFIFO->hDevData,

            chnum,

            messno,
```

```

        &bmbuf);

        //此处添加 BM 数据处理代码

        .....

    }

    tail++;        // Next entry

    tail &= slntFIFO->mask_index;    // Wrap the index

    slntFIFO->tail_index = tail;        // Save the index

} //while

return API_SUCCESS;

}

```

在随产品的示例程序中，有上述回调函数框架的完整代码，用户在二次开发时，可以直接参考引用。

7.7.2 查询方式消息读取

除中断方式消息读取外，也支持查询方式消息读取。查询方式消息读取，需要应用层代码自行控制读取时机，比如按钮触发式单次读取或者定时器式周期性读取。

BC/RT/BM 三种功能都提供有查询方式读取 API 函数接口：

M1553B_BC_ReadNextMessage	查询读取 BC 消息。
M1553B_BC_ReadLastMessage	查询读取最新的 BC 消息。
M1553B_BC_ReadLastMessageBlock	查询读取最新的 BC 消息块。
M1553B_BM_ReadNextMessage	查询读取 BM 消息。
M1553B_BM_ReadLastMessage	查询读取最新的 BM 消息。
M1553B_BM_ReadLastMessageBlock	查询读取最新的 BM 消息块。
M1553B_RT_ReadLastMessage	查询读取最新的 RT 消息。
M1553B_RT_ReadLastMessageBlock	查询读取最新的 RT 消息块。
M1553B_RT_ReadNextMessage	查询读取 RT 消息。

8 测试示例软件使用说明

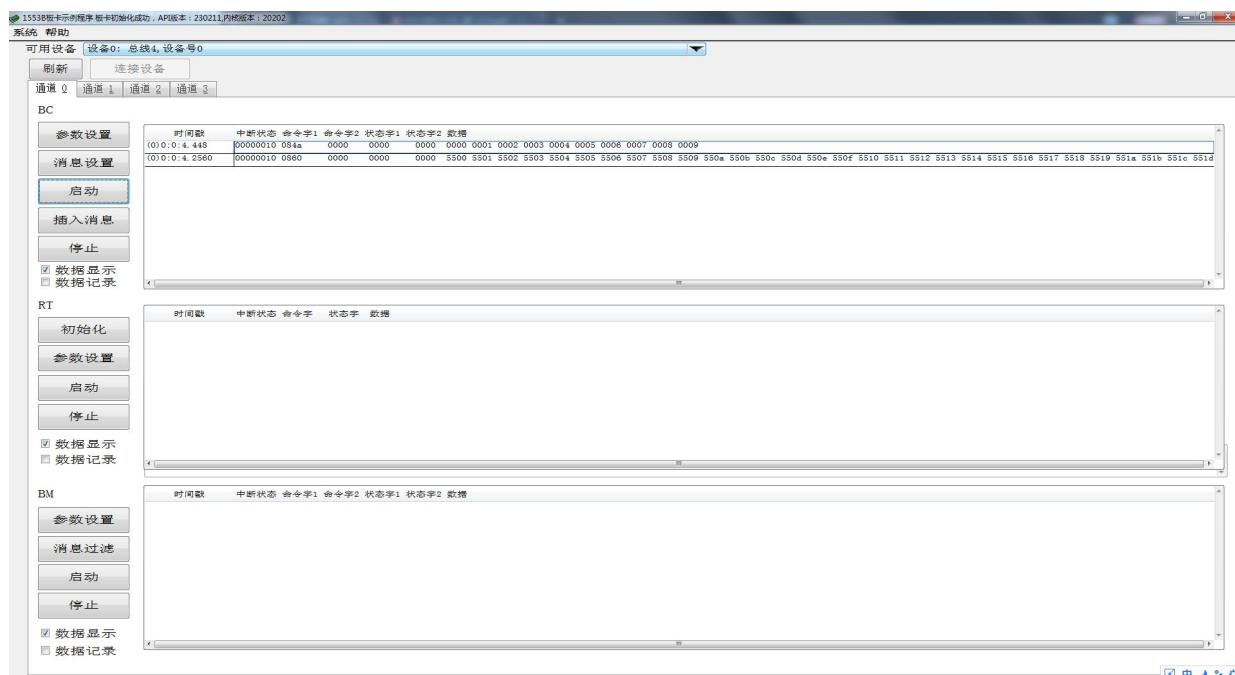
8.1 综述

随板卡配套有示例软件。该软件，位于光盘目录（\demo）下，界面简约，涵盖绝大多数板卡应用基本功能，且免费向客户提供源代码。

特别说明：该示例软件作为产品附属品提供，仅作为一般功能测试和参考示例，科洛威尔不对该程序的功能完备性及应用场景适应性作任何担保；在购买相关硬件产品后，用户可对该程序作任何形式的修改。

示例程序的开发环境为：win7 32bit , labwindows CVI 2017。

软件分为菜单、运行控制、数据显示三大块；1~4 个通道独立控制。示例程序运行主界面如图：



8.2 操作流程

测试软件的具体操作请参见本章 3~8 节，软件的一般操作流程如下：

- (1) 运行程序
- (2) 初始化板卡
- (3) 设备自检（可选）
- (4) BC 功能设置或 RT 功能设置或 BM 功能设置
- (5) 启动 BC 功能或 启动 RT 功能或 启动 BM 功能
- (6) 查看 BC 消息显示，或查看 RT 消息显示，或查看 BM 消息显示
- (7) 测试完成，退出程序。

8.3 板卡级设置

(1) 板卡初始化

运行软件会自动检索系统中可用设备，并在主界面的“可用设备”一栏显示，每个 PCI 设备带有总线号和设备号，这些信息跟设备在系统中安装的槽位有关。选择要使用的设

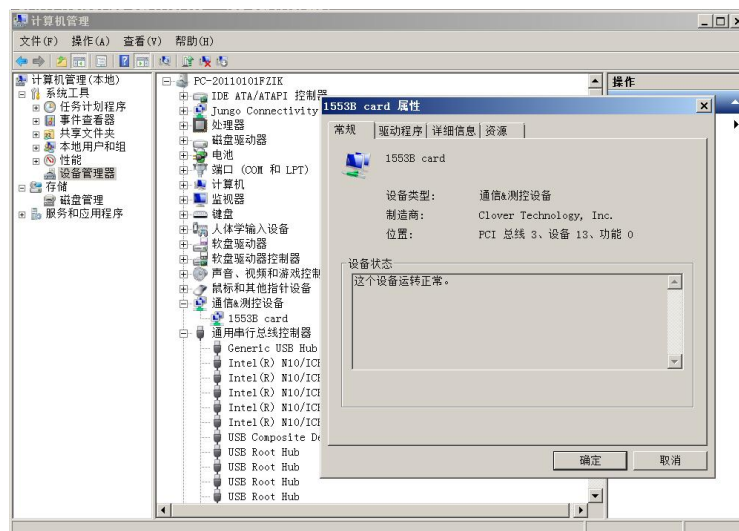
备，点击“连接设备”，对其进行初始化。

如果板卡连接成功，则显示如下提示界面：



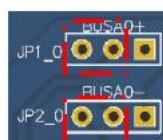
且示例程序的主界面标题栏会显示当前所连接设备的相关信息提示

*系统中，可以在设备管理器中查看板卡的设备号和总线号：

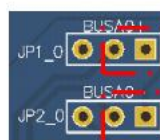


(2) 耦合方式设置

CLV-3051/4051 板卡不支持该设置项。 CLV-3051/4051 的耦合方式采用跳线方式进行设置。每路总线都有对应的两个跳线用于设置该路信号的耦合方式。如通道 1，总线 A 的跳线：



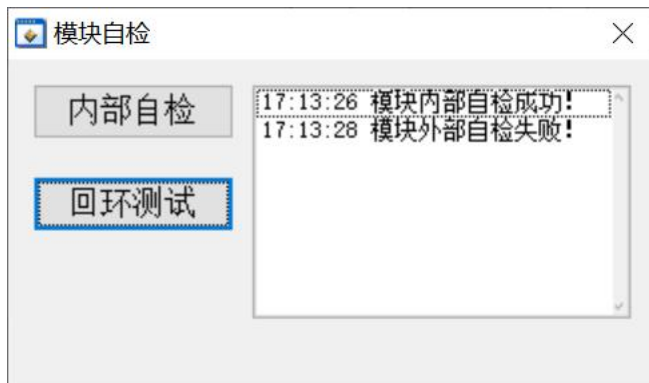
变压器耦合



直接耦合

(3) 自测试

点击菜单“系统\自检”，可以弹出自检对话框，进行板卡自检操作。

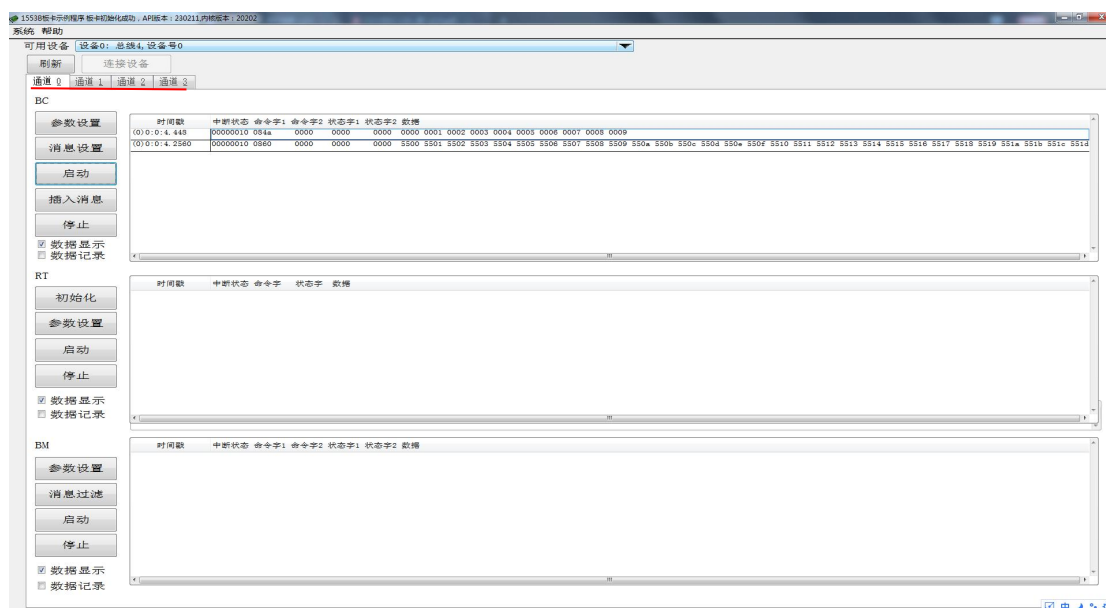


内部自检：存储器级自检。

回环测试：需要将总线 A、B 外部短接，进行外部回路自检。

8.4 多通道支持

配套软件会自动识别所选 1553B 板卡产品的通道数量。对不同通道的操作切换通过软件主界面上的选项页实现。

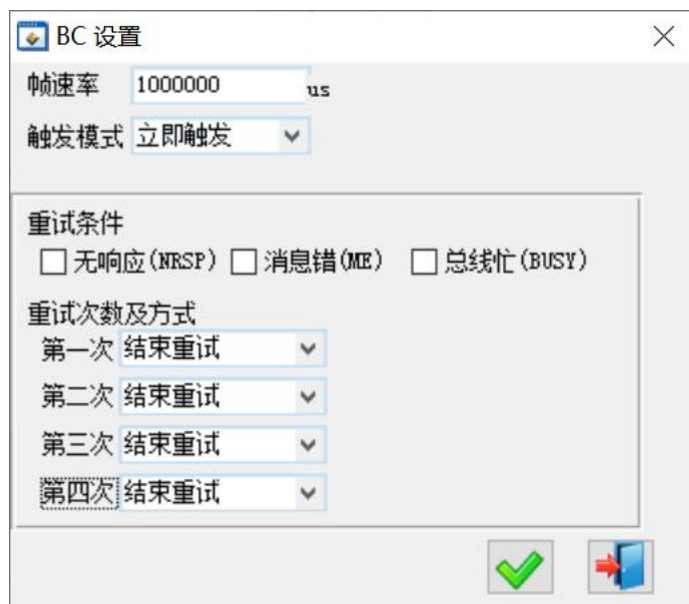


8.5 BC 功能

8.5.1 Bc 参数设置

如果要进行 BC (Bus Control) 功能相关操作，需要进行 BC 设置。

点击“BC 功能\Bc 设置”菜单项，可以进入 BC 设置面板：



帧速率：1553B 消息子帧发送周期。

触发方式：BC 消息发送触发方式。

重试条件：编辑消息重试的条件。

重试次数及方式：消息重试的次数和每次重试的方式选择。

如果是简单测试，可以直接保持软件面板的默认值，直接点击“”确定。

8.5.2 消息设置

点击主界面“BC 消息设置”按钮，进入 BC 消息编辑面板，面板左侧是消息编辑区，中间为操作按钮，右侧是消息队列显示区：



创建消息：

先在左侧消息编辑区，编辑好消息内容，点击“创建消息”按钮，编辑好的消息将被加入 BC 消息队列。

修改消息：

用于对已有消息的修改。操作步骤如下：

- 先在消息队列框中，选中待修改的消息；
- 在左侧消息编辑区，修改消息内容；
- 点击“修改消息”按钮，修改生效；

删除消息：

用于删除消息队列里的一条消息。操作步骤如下：

- 先在消息队列框中，选中待删除的消息；
- 点击“删除消息”按钮，执行删除；

清空消息：

用于删除整个消息队列。操作步骤如下：

- 点击“清空消息”按钮，删除所有消息；

举例：

以下操作，我们将编辑一个含 2 条周期性消息和 1 条非周期性消息的消息链。

第一条消息：

编辑第一条消息内容：

消息类型：BC->RT；

在总线 A 上发送；

帧开始；

RT 地址 1，子地址 2，数据个数 10；

数据 0~9 的递增序列。

点击“创建消息”，可以看到 BC 消息队列中已经生成一条消息“Msg 1 帧开始”。

第二条消息：

消息类型: BC->RT

☒ BUS A ☐ BUS B ☐ 帧开始 ☐ 帧结束 ☐ 消息结束 ☐ 允许重试

命令字1: 2 (Bit15...Bit11)RT Addr, 2 (Bit9...Bit5)Sub addr, 15 (Bit4...Bit0)WordCount

命令字2: 2 (Bit15...Bit11)RT Addr, 1 (Bit9...Bit5)Sub Addr, 10 (Bit4...Bit0)WordCount

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222
2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222	2222

起始值: 2222 数据递增 数据相同

错误注入: EI_NONE

BC 消息队列: Msg 1 帧开始, Msg 2

编辑第二条消息内容：

消息类型：BC->RT；

在总线 A 上发送；

RT 地址 2，子地址 2，数据个数 15；

数据为 0x2222 的相同数据序列。

点击“创建消息”，可以看到 BC 消息队列中已经生成一条消息“Msg 2”。

第三条消息：

消息类型: BC->RT

☒ BUS A ☐ BUS B ☐ 帧开始 ☒ 帧结束 ☒ 消息结束 ☐ 允许重试

命令字1: 3 (Bit15...Bit11)RT Addr, 2 (Bit9...Bit5)Sub addr, 3 (Bit4...Bit0)WordCount

命令字2: 2 (Bit15...Bit11)RT Addr, 1 (Bit9...Bit5)Sub Addr, 10 (Bit4...Bit0)WordCount

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333
3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333	3333

起始值: 3333 数据递增 数据相同

错误注入: EI_NONE

BC 消息队列: Msg 1 帧开始, Msg 2, Msg 3 帧结束 周期消息结束

编辑第二条消息内容：

消息类型：BC->RT；

在总线 A 上发送；

帧结束标志，消息结束标志；

RT 地址 3，子地址 2，数据个数 3；

数据为 0x3333 的相同数据序列。

点击“创建消息”，可以看到 BC 消息队列中已经生成一条消息“Msg 3 帧结束 周期消息结束”。

第四条消息（非周期性消息）：

编辑第二条消息内容：

消息类型：BC->RT；

在总线 A 上发送；

帧开始标志，帧结束标志，消息结束标志；

RT 地址 4，子地址 2，数据个数 4；

数据为 0x4444 的相同数据序列。

点击“创建消息”，可以看到 BC 消息队列中已经生成一条消息“Msg 4 帧结束 帧结束 周期消息结束”。

第 4 条消息不会按周期自动发送，可以使用插入消息功能，进行单次突发性发送。

消息导入与导出：

软件支持将本次编辑的消息导出，并以文件形式保存。如果需要再次使用到这些消息，无需逐条添加，直接点击“消息导入”按钮，选择消息文件，即可实现 BC 消息自动加载。

8.5.3 插入消息发送示例

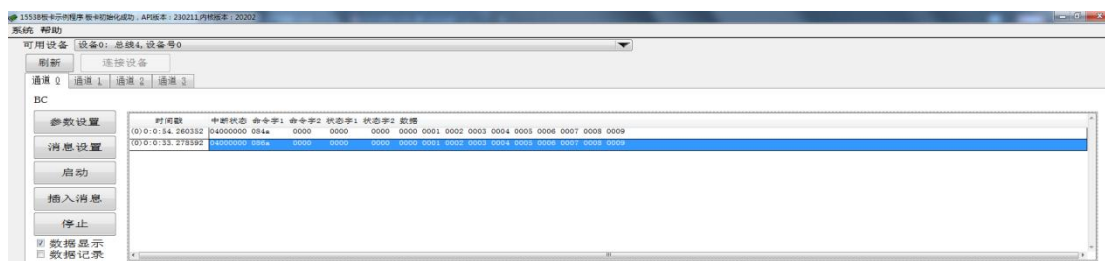
在编辑 BC 消息时，消息结束标志有效后，再创建的消息将不会自动发送。这些消息可以用于突发信息传输。BC 消息编辑时，软件会自动记录这些非周期消息，需要发送时，可以点击菜单项“通道 n/BC 功能/插入消息发送示例”，启动插入消息面板执行单次发送操作。

如我们现在编辑了如下消息：



可见有 Msg1, Msg2 两条周期性消息；Msg2 带有“周期性消息结束标志”，所以其后的 Msg3, Msg4 属于不会自动发送的非周期性消息。

如果要触发这些消息发送，我们可以在启动 BC 后，点击“插入消息”按钮：



将出现插入消息操作界面：



选择需要发送的消息，点击“插入消息”按钮，将触发相应消息执行发送。

8.6 RT 功能

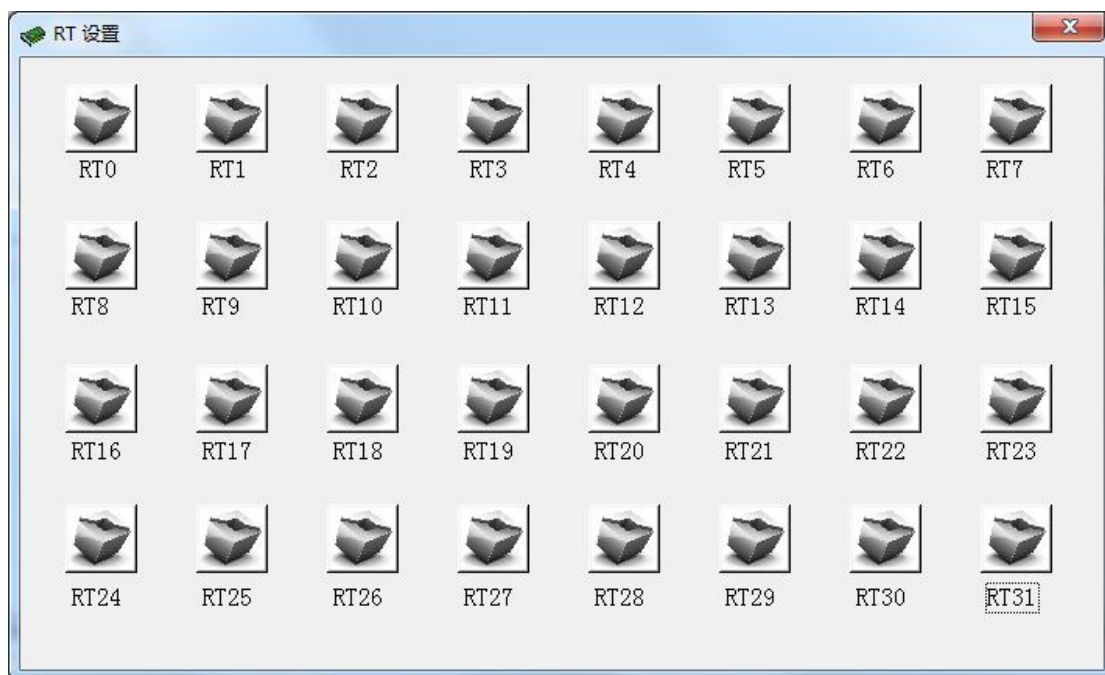
如需进行 RT (Remot terminal) 功能相关操作，需要对 Rt 进行初始化和设置。对 Rt 的设置分三部分：

8.6.1 Rt 初始化

点击“RT 初始化”按钮，初始化 RT 功能

8.6.2 RT 参数设置

点击主界面的 RT 区域“参数设置”按钮，进入 Rt 设置面板：

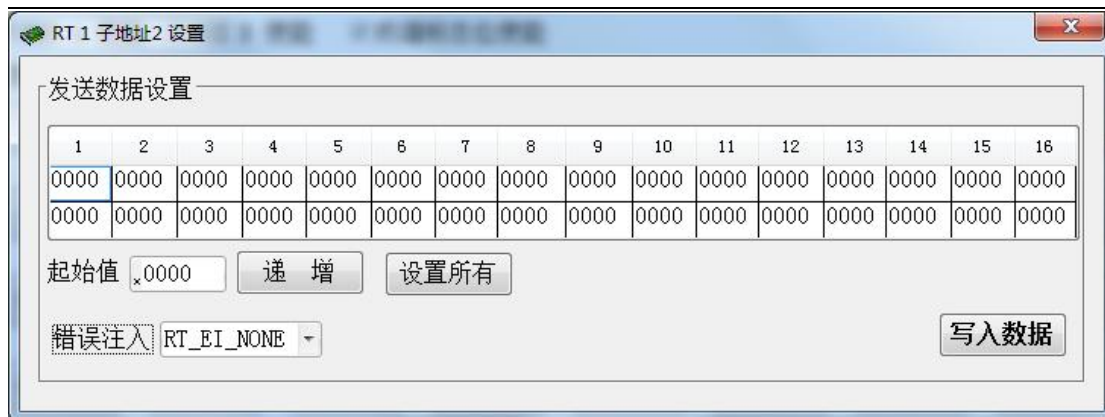


点击相应的 RT 图标，进入 Rt 设置面板，如下图所示：



在该面板上，可以对该 RT 是否响应来自 BUS A/B 的消息、终端标志位是否使能以及各个子地址参数进行设置。**如无特殊要求，请保持 BUS A/B 的消息、终端标志位处于选中使能状态。**

点击面板上子地址按钮，可打开该子地址的设置面板。在子地址设置面板中可对该子地址的（发送）数据缓冲区和错误注入功能进行设置：



按上述界面，编辑完数据，点击“写入数据”按钮，将数据写入。

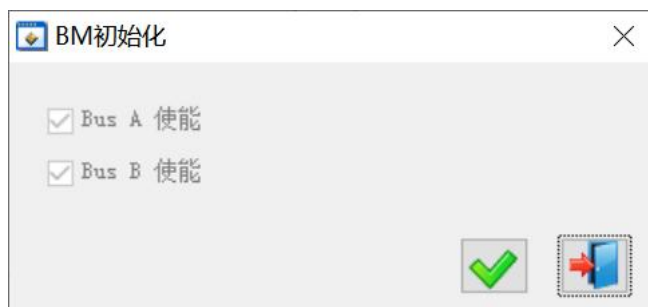
通过 RT 子地址设置面板的“错误注入”选框，可以选择模拟错误方式，进行模拟“错误消息”发送。

8.7 BM 设置

如果要进行 BM (bus monitor) 相关操作，必须进行 BM 初始化和相关设置。

8.7.1 BM 初始化

点击主界面“BM/初始化”按钮，将弹出 BM 初始化面板

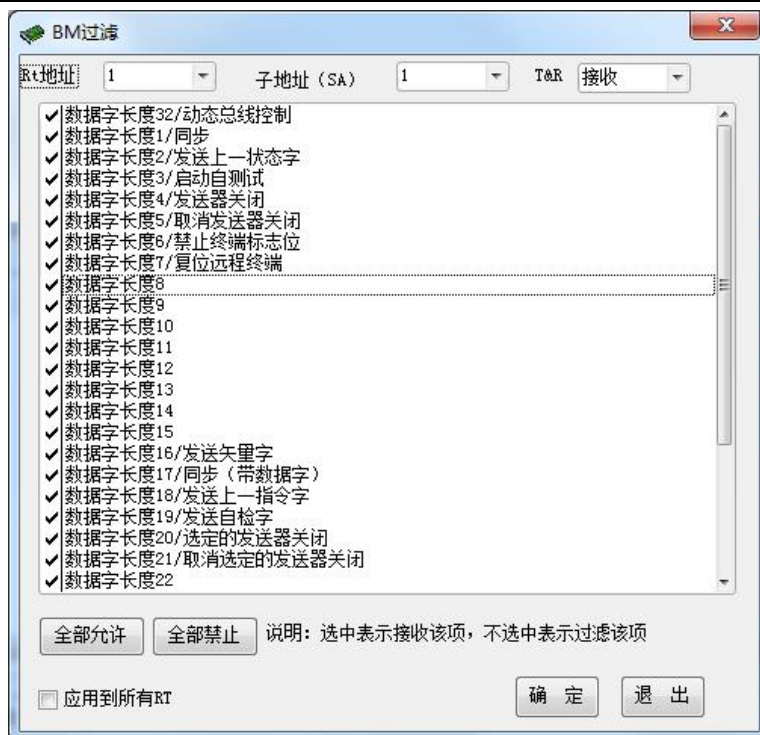


直接点击面板上的确定按钮，可以完成 BM 的初始化操作。

注意：灰色显示的“Bus A 使能”和“Bus B 使能”选项在该版本中不被支持。默认情况下 Bus A 、Bus B 上的消息都被记录。

8.7.2 BM 过滤设置

当需要进行 BM 过滤时，才进行该设置；默认情况下 BM 将记录总线上所有消息。点击“BM/过滤设置”按钮，将弹出 BM 过滤条件设置面板：



BM 消息过滤可以按照 RT 地址、子地址、发送/接收标志进行设置。当勾选上消息时，消息使能，否则该消息将被板卡 BM 功能忽略。

8.8 BC、RT、BM 功能的启动与停止

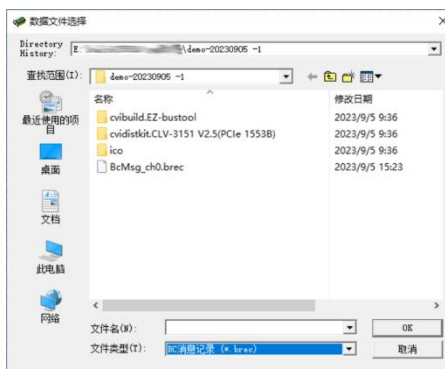
点击程序主界面左侧的相关启动、停止按钮可以启动、停止板卡相关功能。

8.9 数据显示、存盘与查看

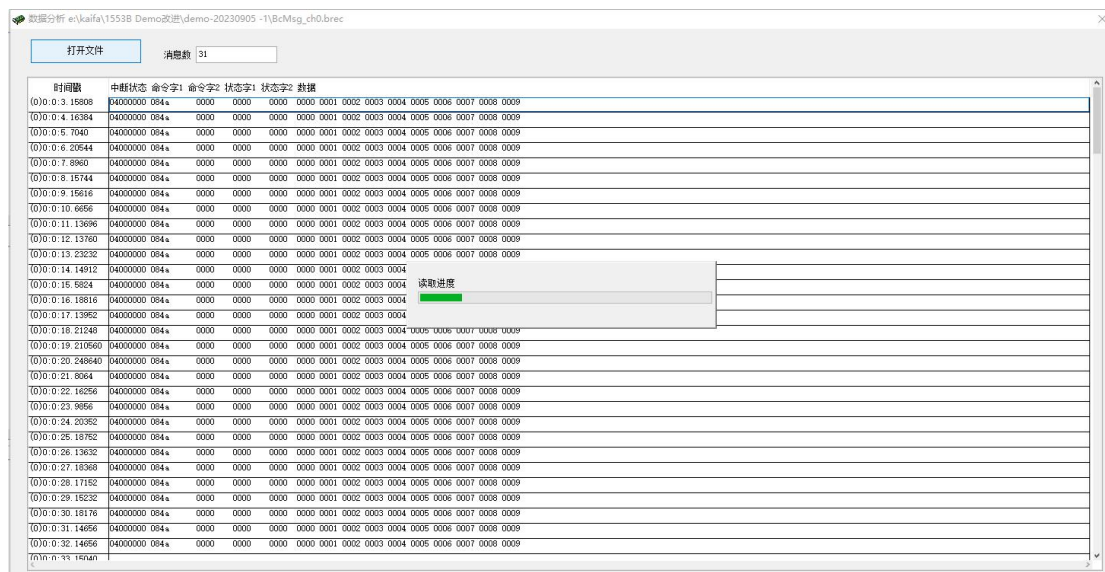
在主界面左侧，BC、RT、BM 对应各有各的显示使能勾选框、数据记录使能勾选框，可以控制当前是否实时显示消息数据、是否继续数据记录存盘。

数据的界面显示会占用一定的处理时间，当通信数据量很大（BC 帧周期短）时，可能会有数据堆积，此时可视情况，选择仅进行数据记录、不实时显示，通信（测试）完成后再进行读取数据进行查看。

记录文件按 BC/RT/BM、通道号命名，存放于软件运行目录下：



点击菜单项“系统/打开记录文件”，可以打开数据分析（查看）界面。点击界面上的“打开文件”按钮，选择记录文件，可以查看记录的数据。

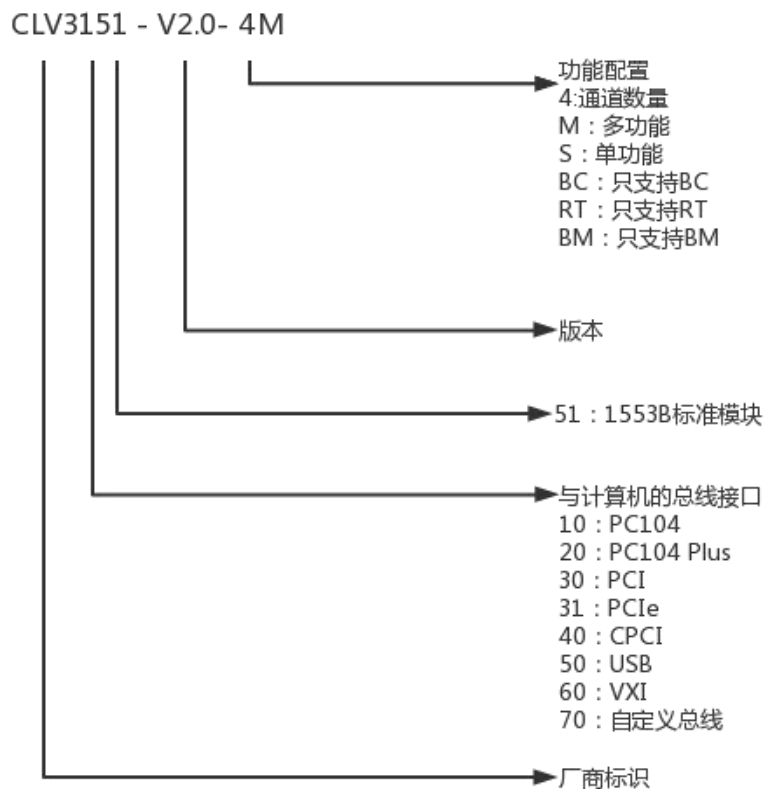


8.10 退出程序

结束操作时，可以点击“系统/退出”菜单项或者直接点击程序关闭按钮，退出程序。

9 产品选型说明

我们提供基于多种总线接口的 1553B 产品，可适应不同应用需求，产品型号编码说明如下：



说明：单功能产品可以软件配置为 BC 模式或者 RT 模式或者 BM 模式。
多功能产品可以同时启动 BC 功能，RT 功能，BM 功能。