GY8505 NET-CAN100/GY8506 NET-CAN200
以太网转 CAN 总线适配器使用说明书

说明书版本：V2.1

日期	手册版本	
2007.7,19	v1.0	Beta 版
2008.8.5	v2.0	接口引脚重新规划，函数库架构更新
2011.6.4	v2.1	更换默认主机端口为 4060，设备端口为 4001，产品此固件参数同时修改。

目　　录

第一章 产品简介

1.1 概述

GY8505/GY8506 NET-CAN 以太网转 CAN 总线接口适配器是带有 1 路或 2 路 CAN 接口和一路 RJ45 以太网接口的智能型 CAN 总线接口适配器，可进行双向传送。采用该接口适配器，PC(或其他以太网设备)可以通过 RJ45 接口连接一个标准 CAN 网络，构建现场总线测试实验室、工业控制、智能楼宇、汽车电子等领域中数据处理、数据采集、数据通讯网络的 CAN 核心控制单元。

NET-CAN 接口适配器可以被作为一个标准的 CAN 节点，是 CAN 总线产品开发、CAN 总线设备测试、数据分析的强大工具；同时，NET-CAN 接口适配器具有体积小、方便安装等特点，也是便携式系统用户的最佳选择。

NET-CAN 接口适配器设备中，CAN 总线电路采用独立的 DC-DC 电源模块，进行光电隔离，使该接口适配器具有很强的抗干扰能力，大大提高了系统在恶劣环境中使用的可靠性。

用户产品说明书给出了通讯编程规约，用户可以方便的开发出 CAN 系统应用软件产品。

1.2 性能与技术指标

- 以太网与 CAN 总线的协议转换；
- GY8505 NET-CAN100 具备 1 路 CAN 接口，GY8506 NET-CAN200 具备 2 路独立 CAN 接口；
- 以太网通讯采用 UDP 协议，透明转换；
- 支持 CAN2.0A 和 CAN2.0B 协议，支持标准帧和扩展帧；
- 支持双向传输，CAN 发送、CAN 接收；
- 支持数据帧，远程帧格式；
- CAN 控制器波特率在 5Kbps-1Mbps 之间可选，可以软件配置；
- CAN 总线接口采用光电隔离、DC-DC 电源隔离；
- 最大帧转换流量为每秒钟 1200 帧 CAN 总线数据；
- CAN 接收缓冲区容量达到 100 帧，共 1300 字节；*(注 GY8506 每通道各 50 帧)*
- CANTools 测试软件可自动检索此设备的网络参数。
- 外部直流工作电源：建议使用 7-24V 输入。
- 隔离模块绝缘电压：1000Vrms；
- 工作温度：-20～85℃；
- 外壳尺寸：100mm*70mm，支持 DIN 导轨安装方式。

1.3 典型应用

- 通过 PC 或笔记本的以太网 RJ45 接口实现对 CAN 总线网络的发送和接收;
- 快速 CAN 网络数据采集、数据分析；
- CAN 总线－以太网网关；
- RJ45 以太网接口转 CAN 网络接口；
- 延长 CAN 总线的网络通讯长度；
- 工业现场 CAN 网络数据监控。

1.4 产品销售清单

1）NET-CAN 以太网转 CAN 总线接口适配器。

3）光盘 1 张。（CANTools 配置软件 NetCanConfig，通信测试软件，以及 Visual C++的 CAN 测试软件的源代码，用户手册，CAN 总线相关资料等）；

1.5 技术支持与服务

一年内免费维修或更换；终身维修服务。

技术支持及购买信息请查阅 www.glinker.cn

第二章　外形与接口描述

2.1 硬件接口描述

产品实物外观如下：

图 1：GY8505 NET-CAN100 适配器

图 2：GY8506 NET-CAN200 适配器

NET-CAN 接口适配器共有 3 组对外接口。

一个标准的 RJ45 以太网接口：LAN 以太网端口；

一个 7pin 的接线端子：电源输入接口；

一个 10pin 的接线端子：CAN 总线信号接口；

红色 LED-PWR 指示电源；

绿色 LED-LAN 指示以太网的连接状态，亮表示已连接，灭表示没有连接；

绿色 LED-CAN 指示 CAN 总线的活动状态，闪烁表示有发送或接收工作；

注 1：GY8506 NET-CAN200 分别对每个 CAN 通道有一个 LED 指示。

注 2：本文中，关于 CAN1 通道的描述只针对 GY8506 NET-CAN200 设备。

端子 1 信号描述

序号		
1		空
2		空
3	VIN	电源输入，**直流 7V-24V**
4	0V	电源地
5	CFG	当 CFG 与 GND 短接后，通电，则恢复出厂配置
6	GND	信号地
7		空

端子 2 信号描述

序号		
1	R1+	当 R1+与 R1-短接，则内部的 120 欧将会被使用
2	R1-	
3	CAN1L	CAN 通道 1：CAN 总线 L 信号
4	CAN1H	CAN 通道 1：CAN 总线 H 信号
5		空
6		空
7	R0+	当 R0+与 R0-短接，则内部的 120 欧将会被使用
8	R0-	
9	CAN0L	CAN 通道 0：CAN 总线 L 信号
10	CAN0H	CAN 通道 0：CAN 总线 H 信号

2.2 硬件恢复出厂配置

恢复出厂配置的方法：将 CFG 与 GND 短接，适配器上电越 3 秒钟，则适配器内部的参数会被恢复为出厂配置。

此情况一般用于当客户遗忘了适配器的 IP 地址的时候，因为当不知道设备 IP 地址时，是无法进行设置参数的。

适配器出厂后的内部默认网络参数如下：

主机 IP: 192.168.0.55，端口 4060

设备 IP: 192.168.0.101，端口 4001

关于恢复出厂配置后的详细参数值，请参见章节 3.2 配置软件的图示描述。

第三章 参数配置

3.1 PC 机网络 IP 设置

用户在使用软件进行配置前，需要保证用户的 PC 机内有以太网卡，而且其配置的 PC 机与 NET-CAN 适配器同在一个网段内。NET-CAN 适配器在出厂时设定了一个默认的 IP 地址（192.168.0.101） 和网络掩码（255.255.255.0），用户可以计算一下看是否和 NET-CAN 适配器在同一网段，公式为：用户 PC 机 IP 地址 与上 用户 PC 机网络掩码，如果结果等于 NET-CAN 适配器的 IP 地址与上 NET-CAN 适配器的网络掩码（按出厂设定的值计算为 192.168.0.0）， 那恭喜你，以下下关于 PC 机网络设置的内容你就不必看了。如果不相等，那以下 PC 机网络设置的内容对你来说就非常重要了。

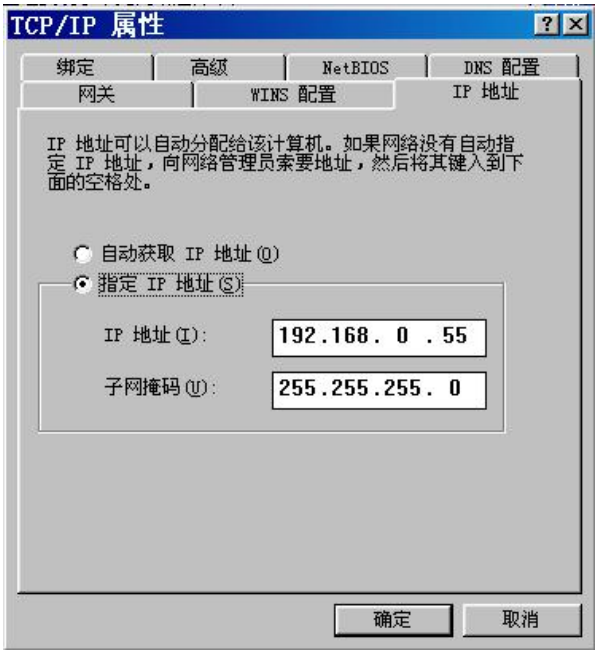
以下的内容是说明：如何使用户的 PC 机与 NET-CAN 适配器处于同一网段。

注：以下仅说明在 Windows 操作系统的网络 IP 设置方法。在其他系统如 Linux 中的方法，请用户自行参考相关资料。

3.1.1 Windows98/Me 网络设置

如果用户使用的操作系统是 Windows 98/ME， 用户首先进入操纵系统，然后使用鼠标点击任务栏的“开始”—>“设置”—>“控制面板”， 双击“网络”图标，您会看到如下界面。

请选择“配置”页面的“TCP/IP” 的属性，可能您会看到不止一个“TCP/IP”,请选择连接NET-CAN适配器的网卡的“TCP/IP”属性界面如下：



请依图所示，在“IP 地址” 页选择“指定IP 地址”， 并填入IP 地址192.168.0.55， 子网掩码255.255.255.0 。点击该页面的“确定”，依提示重启PC 机。

3.1.2 Windows2000/XP 网络设置

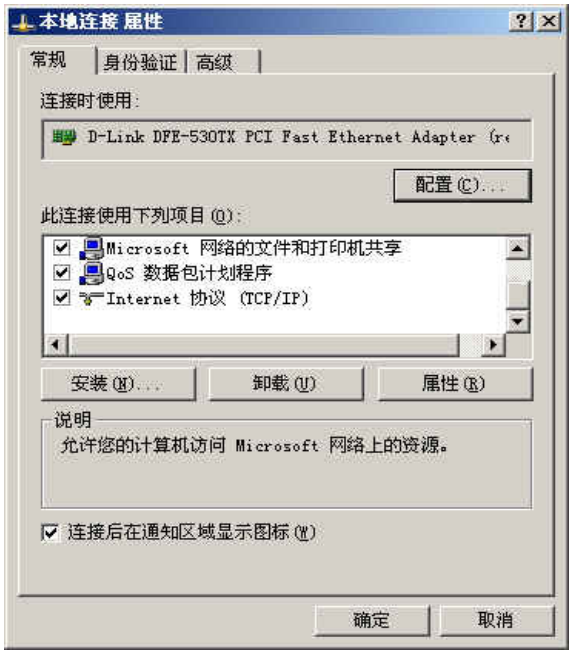
如果用户使用的操作系统是 Windows 2000/XP ，那就有两种方法：一种是增加本机 IP 地址，另一种是修改本机 IP 地址。

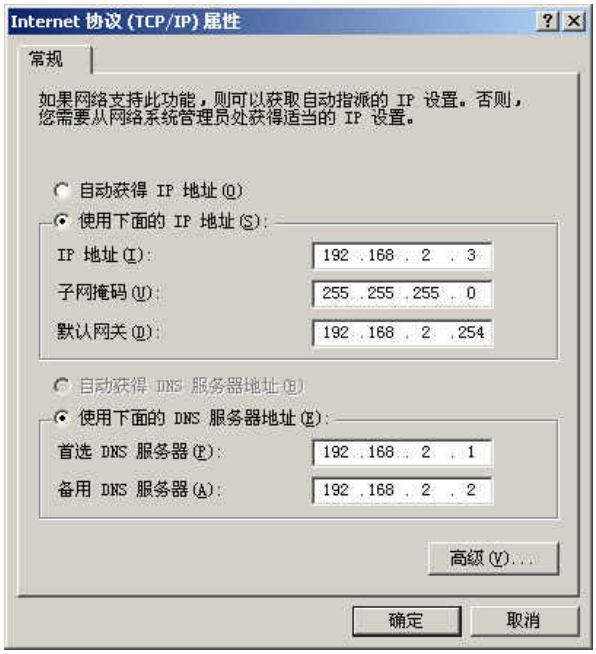
其中，第一种方法一般不会影响您正常上网工作，所以建议采用第一种。

方法1：增加本机IP 地址

假定用户的 PC 机的 IP 地址时 192.168.2.3 ，而模块的 IP 地址时默认 IP 192.168.0.101 。用户进入操作系统后，然后右击网上邻居—>属性。这时网络连接窗口被打开，然后选择本地连接图标（注意，该连接是连接模块网络的连接，如果用户是多网卡的，可能会有多个本地连接，请注意选择），再右击本地连接—>属性。这时弹出下面窗口“本地连接 属性”。

我们选择“常规”页面下的“此连接使用下列项目(D):”的“Internet 协议（TCP/IP）” 项。点击属性弹出以下窗口“Internet 协议(TCP/IP)属性”。





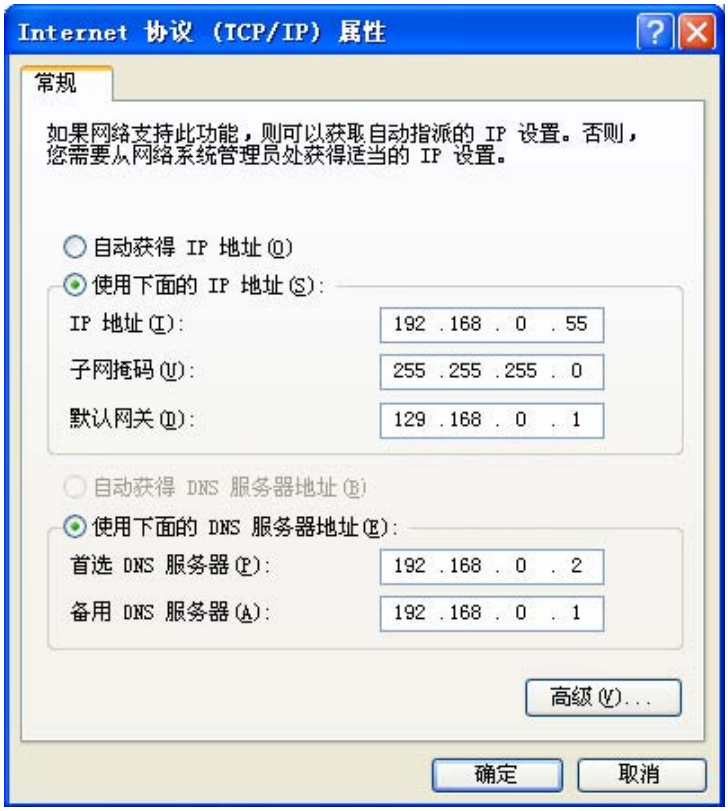
点击该窗口的“高级（Y）…”按钮，这时又会弹出下面窗口“高级TCP/IP设置”。在该窗口的“IP 设置”页面“IP 地址（R）”栏点击添加按钮。这时又弹出以下窗口“TCP/IP地址”。

然后按上内容填入，按添加按钮即可。在退出时请按确定。现在，您就可以设置 NET-CAN 适配器了！

方法 2：修改本机 IP 地址

第二种方法，修改本机 IP 地址。

用户首先进入操纵系统，然后使用鼠标点击任务栏的“开始”—>“设置”—>“控制面板”（或在“我的电脑”里面直接打开“控制面板”），双击“网络和拨号连接”（ 或“网络连接”） 图标，然后单击选择连接 NET-CAN 适配器的网卡对应的“本地连接”，单击右键选择“属性”在弹出的“常规”页面选择“internet 协议（TCP/IP）”,查看其“属性”，您会看到如下页面，请按图所示，选择“使用下面的 IP 地址”,并填入 IP 地址 192.168.0.55，子网掩码 255.255.255.0，默认网关 192.168.0.1（DNS 部分可以不填）。点击该页面的“确定”及“本地连接属性”页面的确定，等待系统配置完毕。



现在，您就可以设置和连接 NET-CAN 适配器了！

3.2 CANTools 测试软件

CANTools 软件可以用于设置参数，以及通讯测试。

安装软件 CANTools_setup.exe，根据提示安装完成即可。

如果需要终端电阻，请将 R+与 R-用导线短接。

注：自测模式下，需要连接终端电阻。如果总线上没有其他设备提供终端电阻，请使用本适配器的内置电阻。

通过交叉网线将本设备与 PC 的以太网接口相连；

运行工具软件 CANTools.exe 测试程序，如下图 4 所示。

3.2.1 软件操作

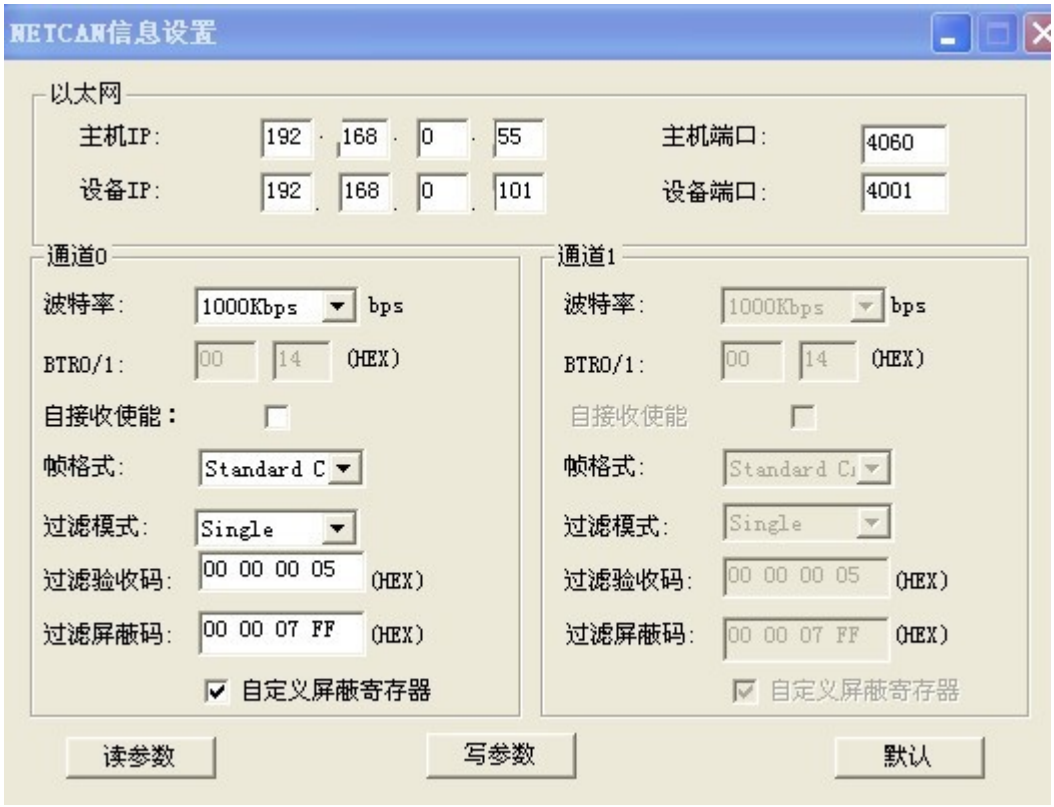
菜单“设备型号”，选择 NET-CAN100,或者 NET-CAN200。

菜单“设备操作”->“启动设备”,会在弹出的窗口中将网络中的 NET-CAN 设备列表显示。显示信息包括该设备的主机 IP，主机端口，设备 IP 和设备端口。

设备出厂默认的设备 IP 地址为“192.168.0.101”，通讯的设备端口和主机端口分别为 4001,4060。

点击“确定”，如果设备工作正常，并且您的电脑 IP 配置正常，将会连接成功。如果不正常，将弹出打开失败的提示窗口。

运行菜单“设置”->“获取 NET-CAN 信息”,将可以得到当前 NET-CAN 设备中的参数，并可以进行修改设置。如下图：



注：上述参数就是出厂配置的默认参数。

2) 按钮“读参数”：会将当前内部的参数都读出并显示。

3)按钮“写参数”：将界面上的参数写入到设备中。这些参数会被保存到 EEPROM 中，会在每次上电时提取。

4) 按钮“默认”：将界面上的参数自动修改为出厂默认的参数。需要点写入参数，才可以被修改到设备中。

3.2.2 设备参数描述

1) 主机 IP：用于通讯转发的主机 IP 地址。

2) 主机端口：用于通讯转发的主机 IP 端口。

适配器只有接收到从这个 IP 地址和端口发来的数据，才能转发到 CAN 总线；同时，CAN 总线端过来的数据，将通过以太网 UDP 方式发到这个 IP 地址和端口。

3) 设备 IP：适配器的 IP 地址。

4) 设备端口：适配器用于 UDP 通讯的端口号。

主机必须通过 UDP 协议发送数据到这个 IP 地址和端口，设备才能接收到。

该端口号为 CAN0 通道的端口号。如果是双通道产品，CAN1 通道的端口号为该值加 1。

如：设置值为 4001，则 CAN0 通道的端口号为 4001，CAN1 通道的端口号为 4002。

注：因以太网 TCP/UDP 协议中，很多端口号已经被占用，所以这些端口号不能被使用。建议使用 4000 以后的端口号。

5) 波特率：CAN 总线的波特率，支持 5k－1Mbps 的 15 中常规速率可选。如果客户需要的波特率不在这个下拉选项中，则可选择自定义速率。BTR0/BTR1 就是适配器中 CAN 总线的波特率位定时器。

6）自接收使能：在使能方式下，设备往 CAN 总线上发送的消息都可以被收回来。可用于设备自测试。

7) 帧类型：标准帧还是扩展帧，仅用于滤波器设置参考。此参数对发送和接收无影响。无论如何设置，这两种类型的 CAN 帧都能接收和发送。

8) 过滤模式：转换器接收 CAN 总线数据时，CAN 总线侧报文的滤波方式。单滤波或双滤波可选。

9) 过滤验收码(ACR):。

接收CAN“帧标识”时的比较值，和“过滤屏蔽码”按照位的关系相对应。在“过滤屏蔽码”设定为相关时，只有接收的“帧标识（帧ID）” 和“过滤验收码”相同时才会将该帧数据收到接收缓冲区中，否则不接收。

填充数据格式为 16 进制形式，每个 8 位的字节间用“空格符”隔开。字节从左到右分别命名为 ACR0、ACR1、ACR2 和 ACR3，位序列为高位到低位。和接收的 CAN 帧 ID 进行对应滤波。最低位位于 ACR3 的 Bit0。

10) 验收屏蔽码 (AMR): 。

当 Mask Code 为 07 FF FF FF 的时候，可以接收所有 CAN 节点的报文。

用来管理“ 过滤验收码”，按位相应管理。当“ 过滤屏蔽码” 的位值为0 时（意为相关），相应位的接收“帧标识”只有和相应位的“过滤验收码”相同才会将该帧数据收到接收缓冲区中；当“过滤屏蔽码”的位值为1 时（意为无关）， 相应位的接收“帧标识”为任何值都可以将该帧数据收到接收缓冲区中。最低位位于ACR3的Bit0。

填充数据格式为 16 进制形式，每个 8 位的字节间用“空格符”隔开。字节从左到右分别命名为 AMR0、AMR1、AMR2 和 AMR3，位序列为高位到最低位。

11) 自定义屏蔽寄存器:

选中该项，则用户可以自己定义CAN 控制器的滤波器，以提供丰富的滤波方式（参见附录A.3 CAN报文滤波器设置）； 不选中该项，则由配置软件设置滤波器为单滤波方式，并且设置滤波器的屏蔽码为只接收ID为设定的“过滤验收码值”的CAN帧。

注：建议：1、若需要接收所有CAN 节点的报文，那么应当选中该项，并且确定“过滤屏蔽码”值为“1F FF FF FF”或者“00 00 07 FF”。

2、若仅需要接收固定标识的信息，那么应不选中该项，只填充实际的验收代码值到“过滤验收码”。如只收帧标识（帧ID）为6 的CAN 报文，那么设置为：不选中该项，“过滤屏蔽码”值为“00 00 00 00”，“过滤验收码”值为“00 00 00 06”。

关于滤波器的设置，下文专门举例介绍。

3.3 举例介绍验收滤波的设置

3.3.1 非自定义屏蔽码

由于是“非自定义屏蔽码”情况，所以用户只需要填充“过滤验收码”的实际值，“过滤屏蔽码”则由配置软件自动设置，全部字节均为0x00 （相关）。

如果要了解整个滤波器原理请参考附录“CAN 报文滤波器的设置”。

图 3.3 非自定义屏蔽码标准帧情况下设置

图 3.3 表示在帧类型为“标准帧”情况下，单滤波和双滤波的设置。

由于是“标准帧”，那么帧ID 只有11 位，所以“过滤验收码”最大值为“0x07FF”， 超过的部分无效，软件只取低11 位。

“单滤波”使用一组滤波器，只能设置一组“过滤验收码”值；“双滤波”则可以设置两个“过滤验收码”，接收的帧ID 如果符合其中任意一个，该帧可以被成功接收。如果帧ID 用来表示地址的话，那么这个节点就有两个地址。

图 3.4 非自定义屏蔽码扩展帧情况下设置

图 3.4 表示在帧类型为“扩展帧”情况下，单滤波和双滤波的设置。由于是“扩展帧”， 那么帧ID 有29 位， 所以“过滤验收码”最大值为“0x1F FF FF FF”， 超过的部分无效，软件只取低29 位。

“单滤波”使用一组滤波器，只能设置一组“过滤验收码”值。

“双滤波”（特殊说明：由于CAN 控制器的特性，“双滤波”的两组“过滤验收码”仅针对最高位的两个帧ID 字节，ID.28～ID.13） 可以设置两个“过滤验收码”，接收的帧ID 如果符合其中任意一个，该帧可以被成功接收。如果帧ID 用来表示地址的话，那么这个节点就有两个地址。

3.3.2 自定义屏蔽码

图 3.5 表示在帧类型为标准帧”情况下，单滤波和双滤波的设置。ID 只有低 11 位有效，所以滤波器的最大值为 07 FF。

图 3.5 自定义屏蔽码标准帧情况下设置

图 3.6 表示在帧类型为“扩展帧”情况下，单滤波和双滤波的设置。

图 3.6 自定义屏蔽码扩展帧情况下设置

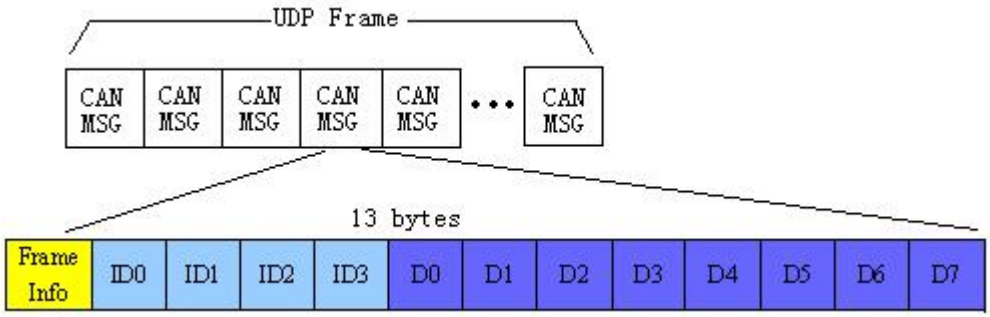
由于是“扩展帧”， 那么帧ID 有29 位， 所以“过滤验收码”最大值为“0x1F FF FF FF”，超过的部分无效，软件只取低29 位。

“单滤波”使用一组滤波器，只能设置一组“过滤验收码”值。

“双滤波”（特殊说明：由于CAN 控制器的特性，“双滤波”的两组“过滤验收码”仅针对最高位的两个帧ID 字节，ID.28～ID.13） 可以设置两个“过滤验收码”，接收的帧ID 如果符合其中任意一个，该帧可以被成功接收。如果帧ID 用来表示地址的话，那么这个节点就有两个地址。

第四章 通信转换规约

4.1 转换格式规约



UDP-->CAN：发送出去的 UDP 包，最多可以含有 40 个 CAN 消息帧；

CAN-->UDP：接收到的 UDP 包，最多含有 51 个 CAN 消息帧；

注：在 GY8506 中，如果希望一个 UDP 包同时被 2 个 CAN 通道进行发送，请使用端口号 3001。

1) 帧信息

帧信息 Frame Info： 一个字节，该字节的 bit 定义如下

Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
FF	RTR	保留	保留	DLC3	DLC2	DLC1	DLC0

FF: 标准帧和扩展帧的标识， 0 为标准帧，1 为扩展帧。

RTR: 远程帧和数据帧的标识， 0 为数据帧，1 为远程帧。除非特殊应用，一般客户都是用数据帧，填 0 即可。

DLC3～DLC0 ： 标识该 CAN 消息帧中的有效数据长度，最多 8 个。

2）ID 域

ID0	ID1	ID2	ID3

CAN 消息帧的 ID 填充域，共 4 个字节。

当为标准帧的时候，占用后 2 个字节。只有 ID0，ID1，以及 ID2 的高 5 位无效，补 0。

举例：当 ID＝0x03FF 的时候，按如下方式填充。

00h	00h	03h	FFh

当为扩展帧的时候，占用 4 个字节。ID0 的高 3 位无效，补 0。

举例：当 ID＝0x12345678 的时候，按如下方式填充

12h	34h	56h	78h

3）数据域

D0	D1	D2	D3	D4	D5	D6	D7

根据 CAN 消息的定义，一个 CAN 帧中，最多可以包含 8 个字节的数据。

当该 CAN 帧不需要 8 个字节的时候，余下的字节补 0。

注意：需要在 FrameInfo 字节中指明有效数据个数。

举例：FrameInfo 中的 DLC3-0 ＝8，表明有 8 个数据有效时，按如下表示

11h	22h	33h	44h	55h	66h	77h	88h

FrameInfo 中的 DLC3-0 ＝8，表明有 8 个数据有效时，按如下表示

11h	22h	33h	44h	55h	66h	00h	00h

4）CAN 消息帧举例

以下例子是一个扩展格式的数据帧，ID 为 0x12345678，包含 8 个数据字节，数据为（11h,22h,33h,44h,55h,66h,77h,88h ）的 CAN 帧的表示方式

88h	12h	34h	56h	78h	11h	22h	33h	44h	55h	66h	77h	88h

以下例子是一个标准数据帧，I D 为 0x3ff， 包含 6 个数据字节，数据为（11h,22h,33h,44h,55h,66h）的 CAN 帧的表示方式

06h	00h	00h	03h	ffh	11h	22h	33h	44h	55h	66h	00h	00h

5）注意事项

该转换器的最大 CAN 帧流量为 1200 帧/S，注意，此为 CAN 波特率为 1Mbps 情况下测得的。

请进行 UDP 发送的时候，UDP 帧 与 UDP 帧之间的间隔。否则有可能出现上一个 UDP 包中的 CAN 帧还没转发完成，新的 UDP 又来了，这样就可能会造成丢帧。举例，CAN 速率 1Mbis 情况下，当发送完一个含有 40 个 CAN 帧的 UDP 包以后，应延时大约 40ms，再发送下一个 UDP 包。当 CAN 总线速率越低，延时等待时间将越长。具体应根据客户实际帧流量情况进行延时处理。

帧流量定义：每秒需要用转换器进行发送或者接收的 CAN 消息帧个数。根据厂家经验，大部分客户的应用都低于 200 帧/S，一般不会超过 500 帧/S。

4.2 软件编程示例

说明：下文以 Windows 操作系统中 Visual C++编程的部分源代码示例，仅供参考。

初始化网络套接字时，先进行如下定义。

```
SOCKET socket1,socket2;
SOCKADDR_IN sockDest, sockDest2, sockDest3;
SOCKADDR_IN sockFrom;
SOCKADDR_IN sockSrc;
CString strDevIpAddress;
int SockAddrlen=sizeof(SOCKADDR);
```

```
int StopFlag=0;
int num=0;
//绑定主机 IP 和端口
socket1 = socket(AF_INET, SOCK_DGRAM, 0);
SOCKADDR_IN sockSrc;
sockSrc.sin_family = AF_INET;
sockSrc.sin_port = htons(4060);
sockSrc.sin_addr.S_un.S_addr = htonl(INADDR_ANY);
bind(socket1, (SOCKADDR *)&sockSrc, sizeof(SOCKADDR));

//生成 SockDest 用于连接测试
sockDest.sin_family = AF_INET;
sockDest.sin_port = htons(3000);
sockDest.sin_addr.S_un.S_addr = inet_addr("192.168.0.101");

//生成 SockDest2 通讯用 CAN0 通道
sockDest2.sin_family = AF_INET;
sockDest2.sin_port = htons(4001);
sockDest2.sin_addr.S_un.S_addr = inet_addr("192.168.0.101");

//生成 SockDest3 通讯用  CAN1 通道
sockDest3.sin_family = AF_INET;
sockDest3.sin_port = htons(4002);
sockDest3.sin_addr.S_un.S_addr = inet_addr("192.168.0.101");
```

4.2.1 设备连接测试

为便于用户进行通信前测试，NET-CAN 适配器提供了一组连接测试的命令字 CDh,0Dh。当适配器收到这个命令以后，将回送当前的设备中的 DevIP 和 HostIP 等网络参数。

注：连接测试在编程时不是必须的，可以略过此步骤

编程示例如下：

```
void CNETCANTestDlg::OnButtonConnect()
{
    char SendData[15];
    char rbuf[100];
    BYTE rbuf2[100];
    int i;
    SendData[0]=0xCD;//连接测试命令字
    SendData[1]=0x0D; //连接测试命令字
```

```
    // 发送连接测试请求
    if(sendto(socket1, SendData, 2, 0, (SOCKADDR*)&sockDest,
sizeof(SOCKADDR))==SOCKET_ERROR)
    {
        MessageBox("udp 发送失败");
        return ;
    }
    Sleep(10);
    CString strFromIP,str;
    //接收数据
    int datalen=recvfrom(socket1, rbuf, 1024, 0, (SOCKADDR*)&sockFrom, &SockAddrlen);
    if(datalen!=13)//正常应返回 13 个字节
        return ;
    for(i=0;i<datalen;i++)//格式转换，char to byte
        rbuf2[i]=rbuf[i];
    for(i=0;i<4;i++)
        DevIP[i]=rbuf2[1+i];//设备 IP 地址
    for(i=0;i<4;i++)
        HostIP[i]=rbuf2[5+i];//主机 IP 地址
    DevUdpPort = rbuf2[9]*256+rbuf2[10];//设备 UDP 端口号
    HostUdpPort=rbuf2[11]*256+rbuf2[12]; //主机 UDP 端口号
    MessageBox("连接成功");
}
```

4.2.2 CAN 发送

编程示例如下：

```
void CNETCANTestDlg::OnButtonSend()
{
    char SendData[1500];
    char rbuf[1000];
    BYTE rbuf2[1000];
    int i;
    for(i=0;i<40;i++)
    {
        SendData[i*13]=0x08;
        SendData[i*13+1]=0x00;
        SendData[i*13+2]=0x00;
        SendData[i*13+3]=0x02;
```

```
            SendData[i*13+4]=0x15;
            SendData[i*13+5]=i*8;
            SendData[i*13+6]=i*8+1;
            SendData[i*13+7]=i*8+2;
            SendData[i*13+8]=i*8+3;
            SendData[i*13+9]=i*8+4;
            SendData[i*13+10]=i*8+5;
            SendData[i*13+11]=i*8+6;
            SendData[i*13+12]=i*8+7;
            SendData[i*13+13]=i*8+8;
        }
        // 发送 UDP 包
        if(sendto(socket1, SendData, 40*13, 0, (SOCKADDR*)&sockDest2,
sizeof(SOCKADDR))==SOCKET_ERROR)
        {
            MessageBox("udp 发送失败");
            return ;
        }
        Sleep(45);
}
```

4.2.3 CAN 接收

编程示例如下：

```
void CNETCANTestDlg::OnButtonReceiveThread()
{
    AfxBeginThread(ReceiveThread,0);//用线程来处理 UDP 接收
}
UINT CNETCANTestDlg::ReceiveThread(LPVOID v)//UDP 接收线程
{
    char rbuf[1000];
    BYTE rbuf2[1000];
    CString str1;
    CNETCANTestDlg *dlg=(CNETCANTestDlg*) AfxGetApp()->GetMainWnd();
    int i=0,temp=0;
    int datalen;
    while(1)
    {
        //UDP 接收函数
```

```
        datalen=recvfrom(socket1, rbuf, 1024, 0, (SOCKADDR*)&sockFrom, &SockAddrlen);
        temp =datalen/13;//当前收到的 CAN 帧数量
        if((datalen%13)==0)//如果该字节数是 13 的整数倍，表示 UDP 包字节完整。
        {
            num+=temp;//累计收到的 CAN 帧数量
            str1.Format("%d",num);
            dlg->GetDlgItem(IDC_EDIT_INFO)->SetWindowText(str1);//显示收到的 CAN 帧总数量
            //CAN 消息的内容都在 rbuf[datalen]数组中
            //处理或显示收到的 CAN 消息，省略……

        }
        if(StopFlag==1)//线程退出
        {
            StopFlag=0;
            return 0;
        }
    }
}
```

第五章 附录

5.1 CAN2.0B 协议帧格式

5.1.1 CAN2.0B 标准帧

CAN 标准帧帧信息是11 个字节，包括帧的信息和帧数据两部分。前3 字节为帧的信息部分。格式如下表：

		7	6	5	4	3	2	1	0
字节1	帧信息	FF	RTR	x	x	DLC （数据长度）			
字节2	帧ID1	x	x	x	x	x	ID.10	ID.9	ID.8
字节3	帧ID2	ID.7～ID.0							
字节4	数据1	数据							
字节5	数据2	数据							
字节6	数据3	数据							
字节7	数据4	数据							
字节8	数据5	数据							
字节9	数据6	数据							
字节10	数据7	数据							
字节11	数据8	数据							

字节1 为帧信息，第7 位（FF）表示帧格式，在标准帧中FF= 0；第6 位（RTR）表示帧的类型，RTR=0 表示为数据帧，RTR=1 表示为远程帧。

DLC表示在数据帧时实际的数据长度。

字节2~3 为报文识别码，其低11 位有效，高5位无效。

字节4~11 为数据帧的实际数据，远程帧时无效。

5.1.2 CAN2.0B 扩展帧

CAN 标准帧帧信息是13 个字节，包括帧信息和帧数据两部分。前5 字节为CAN帧的帧信息部分。

		7	6	5	4	3	2	1	0
字节1	帧信息	FF	RTR	x	x	DLC （数据长度）			
字节2	帧ID1	x	x	x	ID.28－ID.24				
字节3	帧ID2	ID.23－ID.16							
字节4	帧ID3	ID.15－ID.8							
字节5	帧ID4	ID.7－ID.0							
字节6	数据1	数据							
字节7	数据2	数据							

字节8	数据3	数据
字节9	数据4	数据
字节10	数据5	数据
字节11	数据6	数据
字节12	数据7	数据
字节13	数据8	数据

字节1 为帧信息，第7 位（FF）表示帧格式，在扩展帧中FF= 0；第6 位（RTR）表示帧的类型，RTR=0 表示为数据帧，RTR=1 表示为远程帧。

DLC表示在数据帧时实际的数据长度。

字节2~5 为报文识别码，其低29位有效，高3位无效。

字节6~13 为数据帧的实际数据，远程帧时无效。

5.2 SJA1000 标准波特率

GY850X 协议转换器内部采用的是最典型的CAN控制器芯片SJA1000。SJA1000 CAN 控制器的CAN 通讯波特率由寄存器BTR0、BTR1 晶振等参数共同决定。下表A.1列出了一组推荐的BTR0、BTR1 设置值。标注*符号的值是由国际CiA 协会推荐的标准值。

表 5.2　SJA1000 标准波特率

波特率序号	波特率值（Kbps）	晶体振荡器频率 16MHz	
		BTR0（Hex）	BTR1（Hex）
1	5	BF	FF
2*	**10**	**31**	**1C**
3*	**20**	**18**	**1C**
4	40	87	FF
5*	**50**	**09**	**1C**
6	80	83	FF
7*	**100**	**04**	**1C**
8*	**125**	**03**	**1C**
9	200	81	FA
10*	**250**	**01**	**1C**
11	400	80	FA
12*	**500**	**00**	**1C**
13	666	80	B6

14*	**800**	**00**	**16**
15*	**1000**	**00**	**14**

GY850X转换器中的CAN控制器SJA1000 采用16MHz 晶体振荡器。用户也可以自行定义CAN通讯波特率，然后根据SJA1000 的数据手册计算出寄存器BTR0、BTR1值进行设定。

5.3 CAN 报文滤波器设置

转换器的CAN 报文滤波器是基于PHILIPS 公司CAN 控制器SJA1000 的PeliCAN 模式来进行设计的。SJA1000 的滤波器由4 组（4 字节）验收代码寄存器（ACR）和4 组（4 字节）验收屏蔽寄存器（AMR）构成。ACR 的值是预设的验收代码值，AMR 值是用来表征相对应的ACR 值是否用作验收滤波。

但是在SJA1000 的某些模式下，滤波器的某些寄存器没有用到，为了使用方便，所以在配置软件中使用的是直接ID号进行滤波设置和屏蔽，摒弃一些无关的内容。所以本手册滤波器和**SJA1000**的滤波器设置一致，但不相同。

滤波的一般规则是：每一位验收屏蔽分别对应每一位验收代码，当该位验收屏蔽位为1 的时候（即设为无关），接收的相应帧ID 位无论是否和相应的验收代码位相同均会表示为接收；但是当验收屏蔽位为0 的时候（即设为相关），只有相应的帧ID 和相应的验收代码位值相同的情况才会表示为接收。并且只有在所有的位都表示为接收的时候，CAN 控制器才会接收该帧报文。

滤波的方式上又分“单滤波”和“ 双滤波”两种。并且在标准帧和扩展帧情况下滤波又略有不同。在配置软件的“自定过滤屏蔽码”的情况下开放滤波器所有功能。现阐述如下：

1． 单滤波配置

这种滤波器配置方式可以定义成一个长滤波器。滤波器字节和信息字节之间位的对应关系取决于当前接收帧格式。

标准帧： 在帧格式为标准帧时，在验收滤波中仅使用ACR前两个字节（ACR3和ACR4） 中的部分数据（低11位）来存放过滤验收码。同样，过滤屏蔽码也只采用AMR3和AMR4的低11位。

在AMR的位为0时（意为相关）， 当ACR的相对应位（ 如ACR1.0 对应AMR1.0，同时也和ID.00相对应）和接收帧标识的对应位值相同时，表现为“可接收”（逻辑1）； 当两者不等时表现为“不接收”（逻辑0）。或者当AMR的位为1时，无论ACR的相对应位和接收帧标识的对应位值是否相同，均表现为“可接收”（逻辑1）。

对于一个成功接收的信息所有单个位的比较后都必须发出接收信号。如图 5.1 所示。

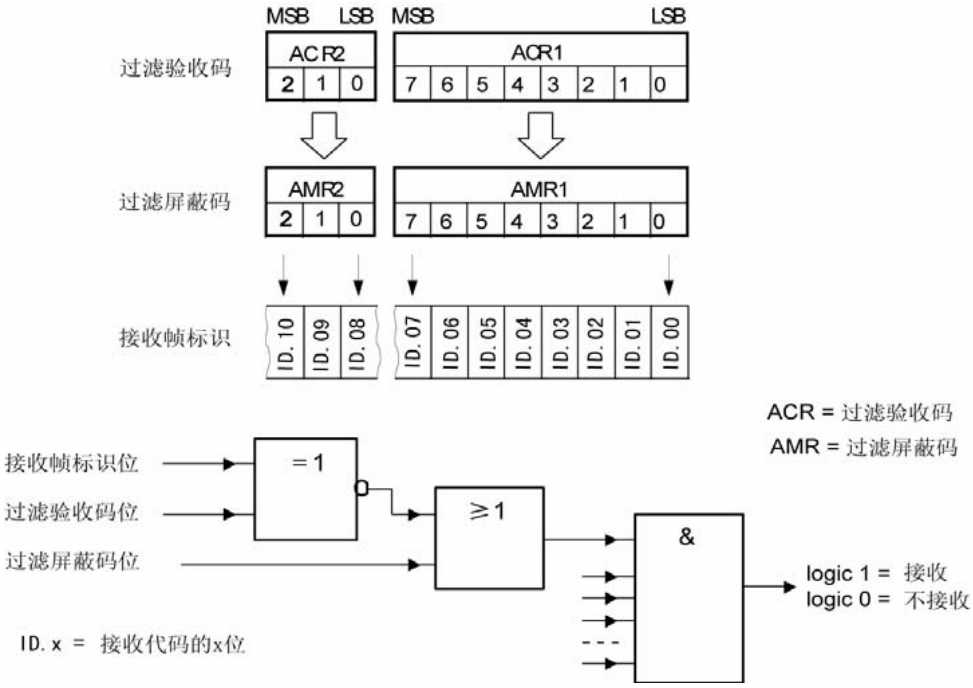


图 5.1 标准帧单滤波示意图

扩展帧：在帧格式为扩展帧时，由于帧标识是29 位，所以在验收滤波中使用ACR 的四个字节中的部分数据（低29 位）来存放过滤验收码。同样，过滤屏蔽码也只采用AMR 的低29 位。

接收逻辑关系和标准帧相同，逻辑表示如图 5.2 所示。

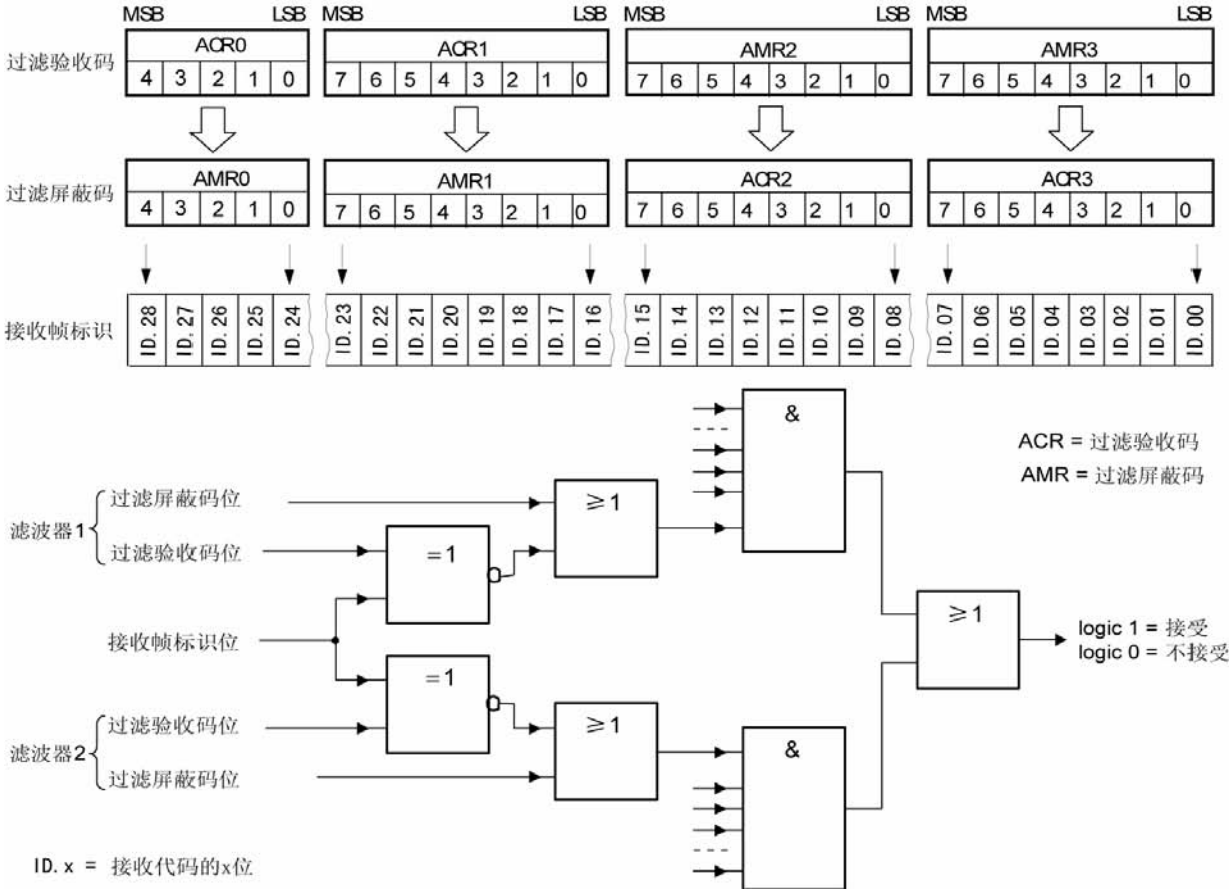


图 5.2 扩展帧单滤波示意图

2. 双滤波配置

这种配置可以定义两个短滤波器。一条接收的信息要和两个滤波器比较来决定是否放入接收缓冲器中。至少有一个滤波器发出接受信号，接收的信息才有效。滤波器字节和信息字节之间位的对应关系取决于当前接收的帧格式。

标准帧：对于标准帧，那么则相当于有两个单滤波情况下的滤波器对接收帧标识进行滤波。接收逻辑如图 5.3 所示。为了能成功接收信息，一组滤波器的单个位的比较时均要表示为接收。 两组滤波器至少有一组表示接收，该帧才会被接收。

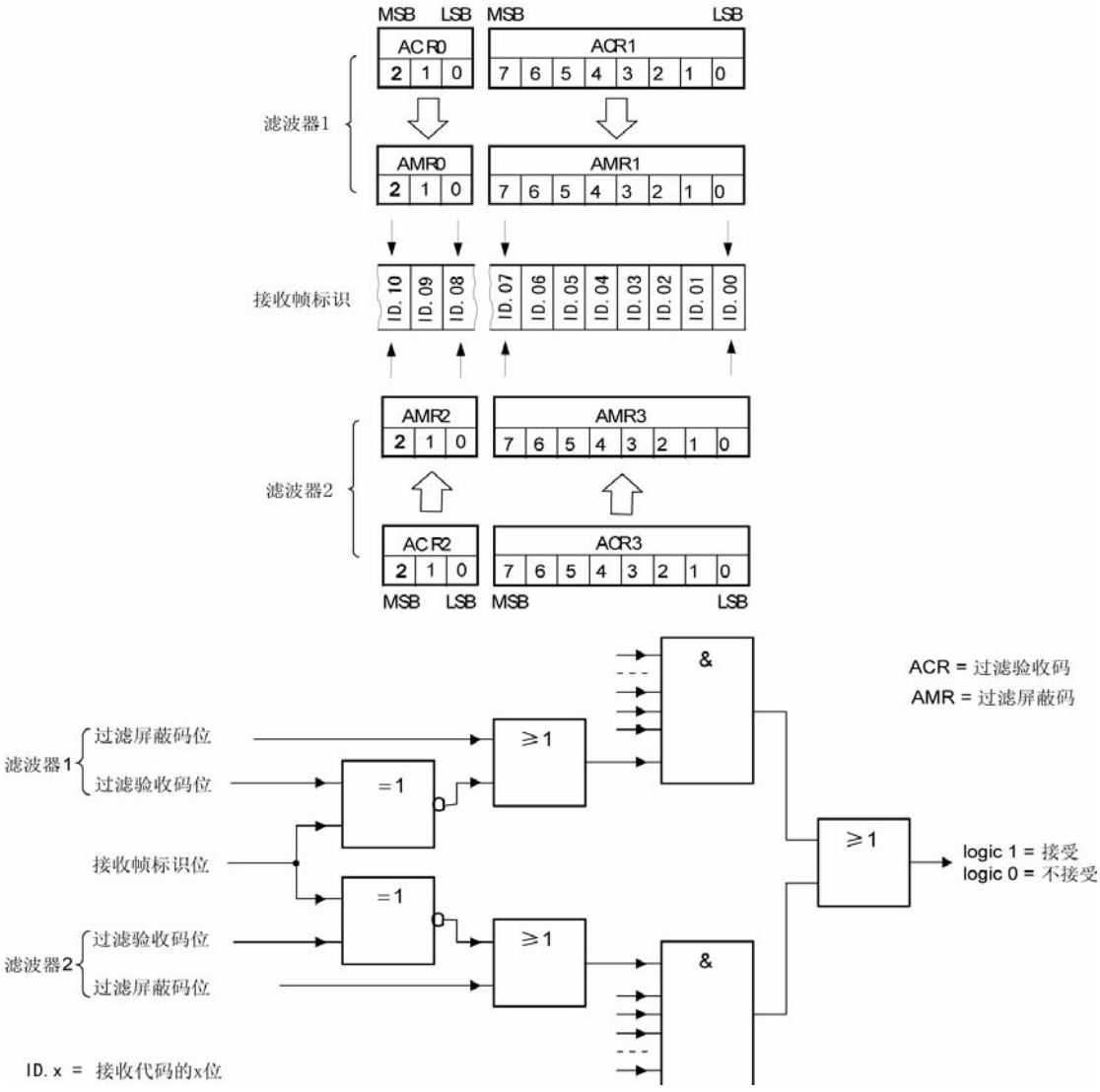


图 5.3 标准帧双滤波示意图

扩展帧：

对于扩展帧，定义的两个滤波器是相同的。两个滤波器都只比较扩展识别码的前两个字节——ID.28到ID.13，而不是全部的29位标识。如图 5.4 所示。

为了能成功接收信息，一组滤波器的单个位的比较时均要表示为接收。

两组滤波器至少有一组表示接收，该帧才会被接收。

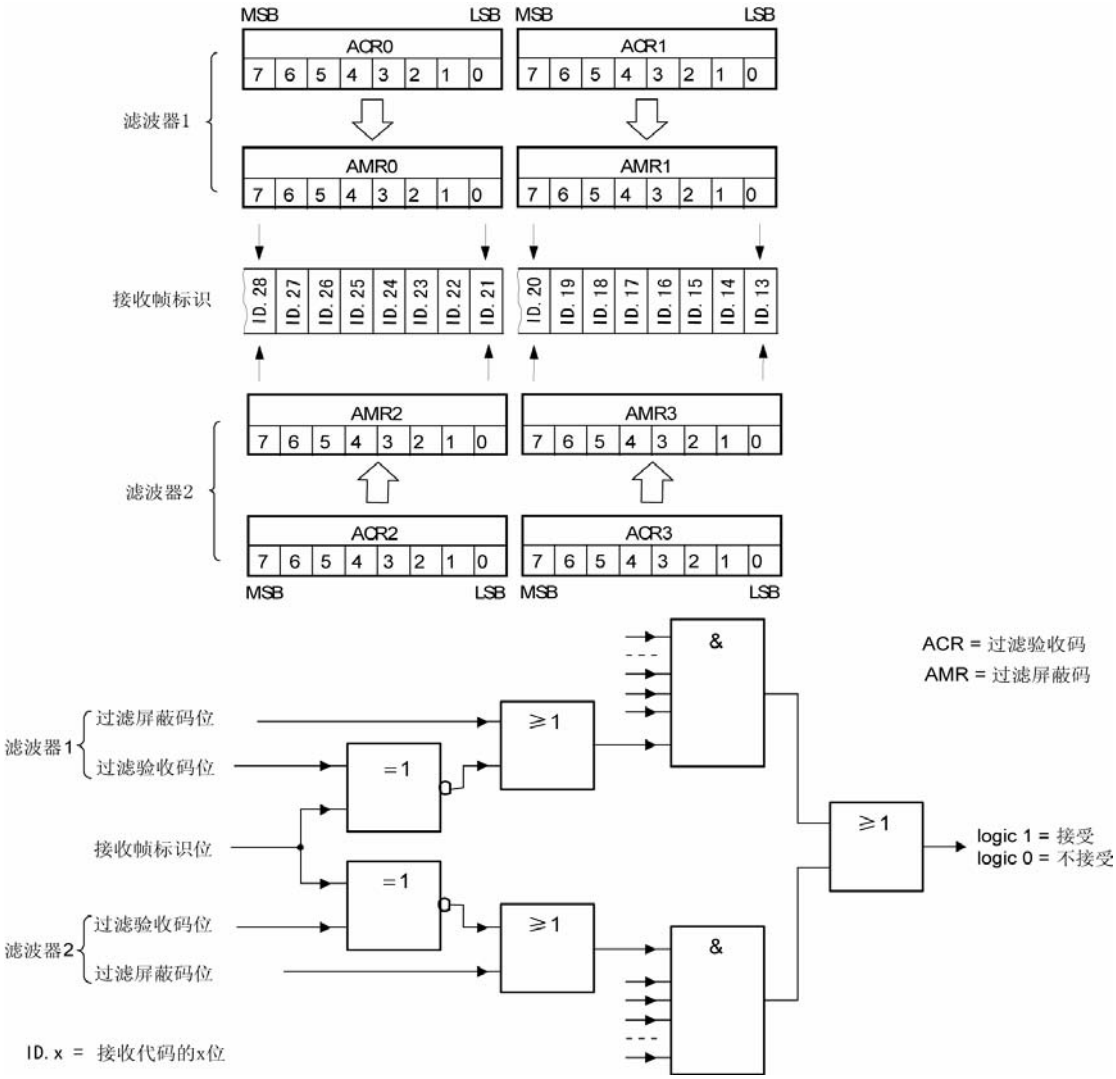


图 5.4 扩展帧双滤波示意图