SL-CAN 多功能通讯转换器使用说明书

用户手册

（V0.9 版本）

2006 年 9 月

一、SL-CAN 产品简介

1．1　概述

SL-CAN 多功能通讯转换器（以下简称 SL-CAN 转换器）是用于 CAN 总线和 RS－232、RS－485、USB 总线之间的数据交换协议的转换器。

SL-CAN 转换器集成有 1 个 CAN 通道、一个 RS232 通道、一个 RS485 通道、一个 USB 通道。实现了 RS232 到 CAN、RS485 到 CAN、USB 到 CAN 的 3 种转换方式。可以方便的实现用户开发，或者将该模块直接嵌入到用户的实际应用中。

其中 USB 接口通过 USB 转串口芯片驱动成串口工作方式，USB 驱动芯片采用 PL2303HX 接口芯片。使用 USB 接口可以很方便的与没有串口的电脑进行连接来调试。用户不需要购买专门的 USB 转 RS－232 设备。RS－485 也采用串口的工作方式，RS－485 驱动芯片采用 TI 公司的 SN65LBC184，最多可以 128 接个从设备。

转换器的设置可以通过一个标准的串口软件（如串口调试助手，超级终端）进行设置，设置界面简单方便，以导航的方式进行操作。包括串口的设置，CAN 的设置，ID 的设置，工作模式的设置，还可以查看设置的参数，并且可以还原到工厂初始状态。关于软件的设置请参见第三章。

CAN 和 RS－232 支持多种通讯的波特率，RS－232 通讯波特率范围从 600bps－115200bps 可选。CAN 通讯速率范围从 10Kbps－1000Kbps 可选。

该转换器支持透明数据转换、透明＋标识的数据转换、MODBUS RTU 和 ASCII 四种工作模式。其中透明数据转换适用于串口和 CAN 透明数据流的传输；透明＋标识的数据转换适用于用户自定义协议的串行数据；MODBUS RTU 和 ASCII 协议转换适用于采用标准 MODBUS 协议的设备之间进行数据转换。四种转换方式可以通过软件设置。

SL-CAN 转换器可以采用 2 种方式供电，外部 DC 供电和 USB 供电。USB 供电可以更方便用户进行调试。

该转换器采用表面安装工艺，CAN 通道带光电隔离模块，电源隔离，大大提高了在恶劣环境中的使用可靠性。

1．2　性能指标：

- CAN 端支持 CAN2.0A、CAN2.0B 协议，符合 ISO/DIS11898 规范；
- 一路 CAN 接口，波特率在 10Kbps－1000Kbps 可选；
- 一路 RS232 接口，波特率在 600bps－115200bps 可选；
- 一路 RS485 接口，波特率在 600bps－115200bps 可选；
- RS－485 符合 TIA/EIA－485 和 ISO/IEC8482：1993（E）标准；
- 一路 USB 接口，波特率在 600bps－115200bps 可选；
- USB 兼容 USB1.1 规范，支持 Windows98/SE、ME　2000、XP 等操作系统；
- 支持透明，透明带标识，标准 MODBUS RTU 和 ASCII 协议转换模式共 4 种；
- CAN 接口采用光电隔离，1000Vrms DC/DC 电源隔离；
- 最高帧流量：xxx 帧/秒；
- 两种电源供电模式，外部 DC 输入和 USB 电源输入；
- 工作电源：＋8V～＋16VDC 或 USB 5V 电源；

- 工作温度：－40～＋85 度；
- 产品尺寸：122×82×32 mm。

1、3　工作原理

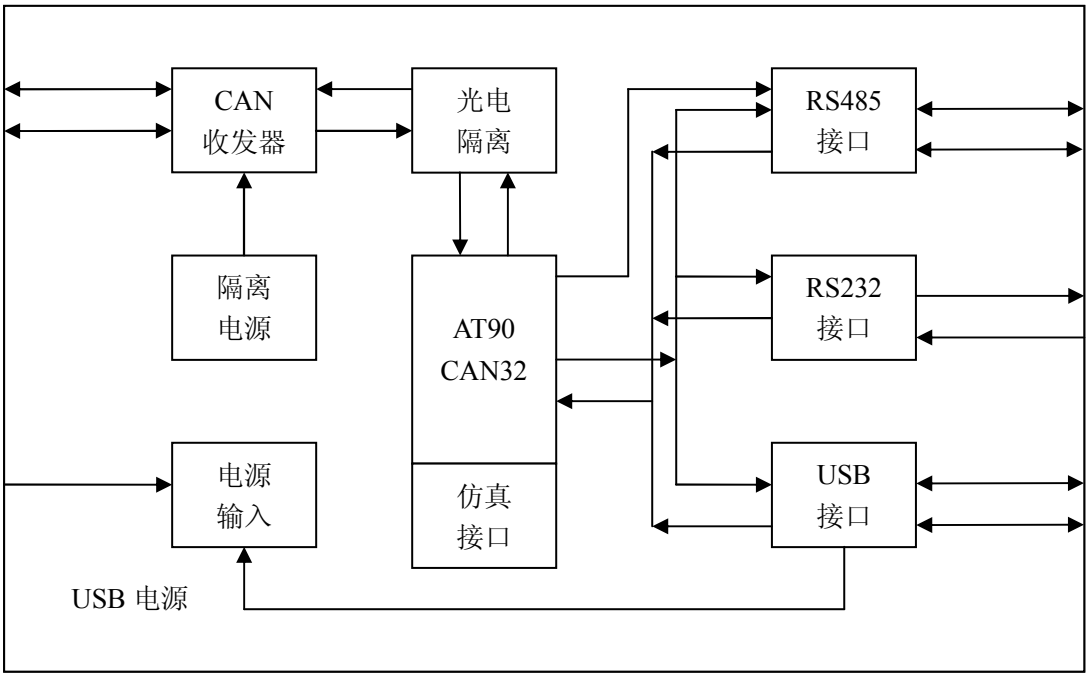


图 1.1　系统构成

SL-CAN 转换器工作原理如图 1.1 所示。RS－485，RS－232，USB 接口通过接口芯片与单片机串口连接，单片机的 CAN 接口通过光电隔离与 CAN 收发器芯片连接。CAN 收发器供电采用隔离电源。

SL-CAN 转换器采用 Atmel 最新的单片机 AT90CAN32-16AU（板子上使用 12M 晶振）。AT90CAN32 自带 CAN 接口，含有 15 个 Mobs，CAN 收发支持 2.0A 和 2.0B 工作模式，CAN 传输速度高。

单片机内部含有的 EEPROM 用来保存用户设置的数据。

单片机内部含有的看门狗用来保障长期运行的稳定。

RS－485，RS－232，USB 接口最终统一转换成串口和单片机连接。3 个接口方式不可以同时使用，同一时刻只能使用其中之一。否则将造成数据传输的错误。在其中一个接口传输或配置的同时，其他 2 个接口可以作为监听，而不能发送数据。

该转换器有 2 种方式供电：1：采用外部 DC 输入供电；2：USB 直接供电。即使同时供电也不会出现问题。

1．4　产品清单：

CAN 多功能转换器	1 台
CAN 多功能转换器用户手册（电子版）	1 份
配套串口接口电缆	1 条
配套 USB 接口电缆	1 条
软件光盘	1 张

1．5　技术支持：

主页：www.sl.com.cn　　www.avr.com.cn
邮箱：bjsl@sl.com.cn
电话：010－82623550、82623551、62653785、62642419

二、硬件描述

2．1　　产品外观

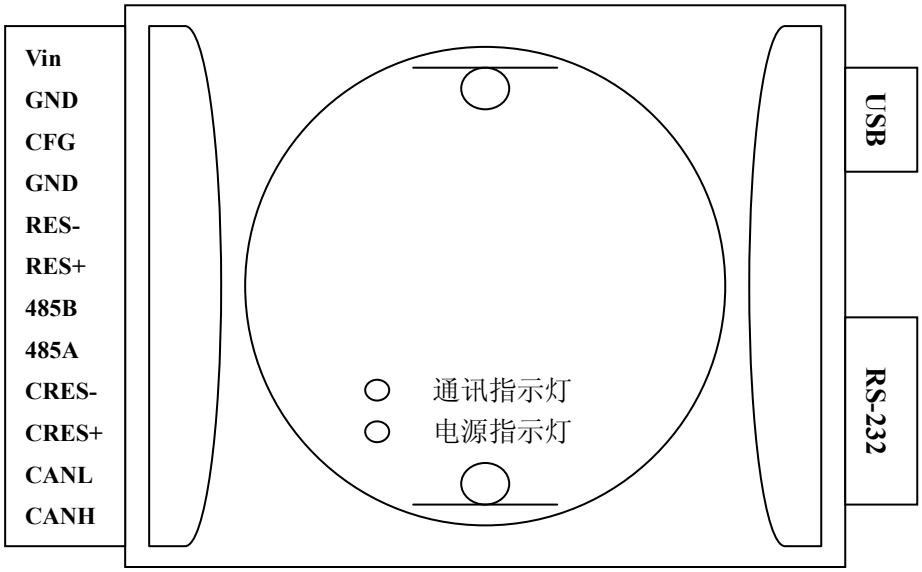


图 2.1　产品外观

2．2　　接口描述

SL-CAN 转换器具有多功能用户接口，如图 2.1 所示。

1、CAN 接口和 RS－485 端接口

2、RS232 接口

3、USB 接口

接口一共分成 3 部分，CAN 和 RS－485 端接口；RS－232 接口；USB 接口。下面分别对端口进行详细的描述。

2．2．1　　CAN 和 RS485 端接口描述

CAN 和 RS－485 端接口定义如下：

引脚号	引脚名称	引脚含义
1	Vin	电源正
2	GND	电源地（0V）
3	CFG	配置输入
4	GND	电源地（0V）
5	RES-	RS－485 网络电阻
6	RES+	RS－485 网络电阻
7	485B	RS－485 信号连接端 B
8	485A	RS－485 信号连接端 A
9	CRES-	CAN 网络电阻
10	CRES+	CAN 网络电阻
11	CANL	CAN 信号连接端 L
12	CANH	CAN 信号连接端 H

表　2.1

CAN 和 RS－485 端接口定义如表 2.1 所示：
引脚 1 标识“Vin”接外部＋8 －＋16V 直流电源；
引脚 2 标识“GND”是外部电源地；
引脚 3 标识“CFG”是转换器的配置引脚。该引脚悬空表示上电后进入正常转换模式，改引脚接 GND，上电后则进入配置模式；
引脚 4 标识“GND”是外部电源地；
引脚 5，6 是“Res ”和“Res+”，是 RS－485 网络的终端电阻，当该 RS－485 做为网络终端时，需要连接该匹配电阻，注意，转换器内部已经集成该电阻，只需将 5，6 短路即可；
引脚 7，8 是“485B”和“485A”，分别接 RS－485 接口的 B 端和 A 端；
引脚 9，10 是“CRES-”和“CRES+”，是 CAN 网络的终端电阻，当该 CAN 做为网络终端时，需要连接该匹配电阻，注意，转换器内部已经集成该电阻，只需将 9，10 短路即可；
引脚 11，12 是“CANL”和“CANH”，分别接 CAN 接口的 CANL 端和 CANH 端。

2．2．2　RS232 端接口描述

RS232 端接口定义如下：

引脚号	引脚名称	引脚含义
1	N.C.	未连接
2	TXD	数据发送端
3	RXD	数据接收端
4	N.C.	未连接
5	GND	地
6	N.C.	未连接
7	N.C.	未连接
8	N.C.	未连接
9	N.C.	未连接

表 2.2

RS232 接口定义如表 2.2，RS232 采用标准的 DB9 孔接口，引脚定义符合 RS232 规范，这里只使用 3 线连接方式。

2．2．3　USB 端接口描述

USB 端接口定义如下：

引脚号	引脚名称	引脚含义
1	USB VCC	USB 电源正
2	D-	USB 数据
3	D+	USB 数据
4	GND.	USB 地

表 2.3

USB 接口定义如表 2.3，USB 采用标准 USB 接口，引脚定义符合 USB 设备端规范。

2．3　指示灯说明

指示灯定义如表 2.4：

指示灯	颜色	说明
电源灯	绿色	上电工作正常 LED 点亮
通讯灯	橙色	配置或数据传输时 LED 闪烁，没有传输 LED 熄灭

表 2.4

注：通讯灯在配置时慢速闪烁，在数据传输时快速闪烁。

2．4　　CAN 总线连接

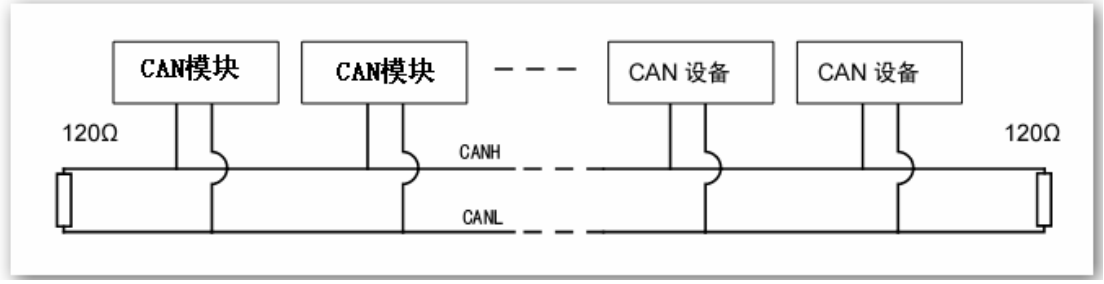


图 2.2　CAN 总线连接

SL-CAN 转换器和 CAN 总线连接时 CANH 接 CANH 口，CANL 接 CANL 口，如图 2.2 所示。

按照 ISO11898 规范，为了增加 CAN 通讯的可靠性，总线网络的 2 个终端要加入网络匹配电阻（120 欧姆），如图 2.2 所示。

SL-CAN 转换器本身自带 120 欧姆 CAN 终端电阻，当做为网络终端使用时，用户可以将接口端的 9 脚和 10 脚的“CRES-”和“CRES＋”用短路线连接起来即可。

2．5　　RS－485 总线连接

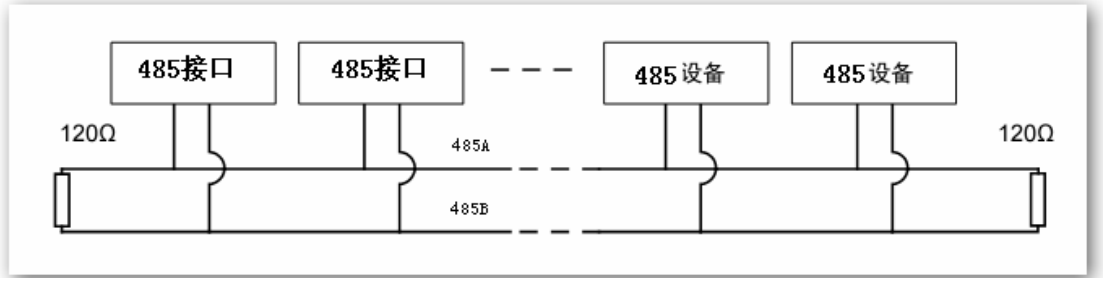


图 2.3　RS－485 总线连接

SL-CAN 转换器和 RS－485 总线连接时 485A 接 485A 口，485B 接 485B 口，如图 2.3 所示。

按照 RS－485 规范，为了增加 RS－485 通讯的可靠性，总线网络的 2 个终端要加入网络匹配电阻（120 欧姆），如图 2.3 所示。

SL-CAN 转换器本身自带 120 欧姆 RS485 终端电阻，当做为网络终端使用时，用户可以将接口端的 5 脚和 6 脚的“RES-”和“RES＋”用短路线连接起来即可。

三、配置说明

3．1　配置说明：

通过一个标准的串口通讯软件（如串口调试助手，超级终端等）可以对 SL-CAN 转换器的所有功能进行设置，下面对设置进行一个详细的介绍。

为了进入设置界面，要把 CAN 接口端的 CFG 管脚 3，和 GND 管脚 4 接起来，当 CFG 被接地后，上电则进入配置模式，CFG 管脚悬空时，上电则进入正常工作模式。

进入配置模式：

1、将 CFG 管脚和 GND 连接起来。

2、将串口或 USB 口和计算机连接起来。

3、运行串口通讯软件。

4、选择好相应的串口并设置为 9600-8-N-1 模式，ASCII 显示和发送。

5、开机上电。

6、通过串口的显示进行设置。

3．2　软件界面：

3．2．1　　串口调试助手

设置软件如果采用的是串口调试助手 V2.2 版本，界面如图 3.1 所示：

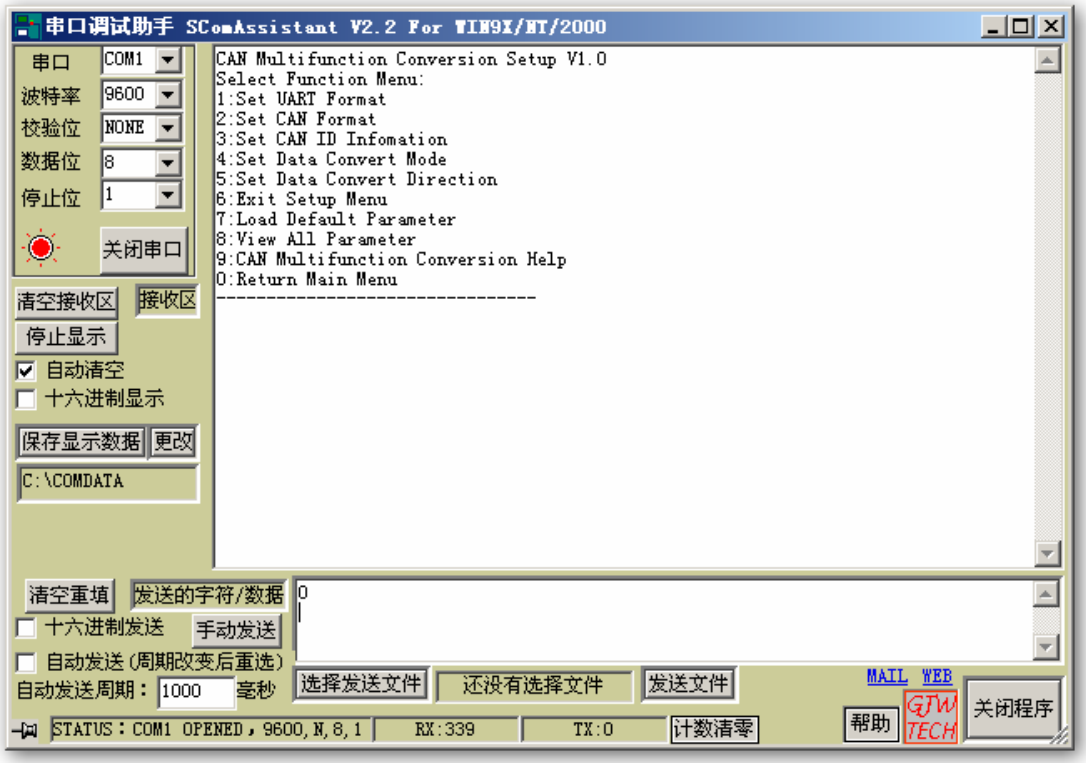


图 3.1

上面的区域是显示接收到的数据，下面的区域是要发送的数据。

注意：每个要发送的数据必须有换行和回车。以换行和回车做为输入的结尾。

3．2．2　　Windows 的超级终端

设置软件如果用 Windows 的超级终端，显示界面如图 3.2：

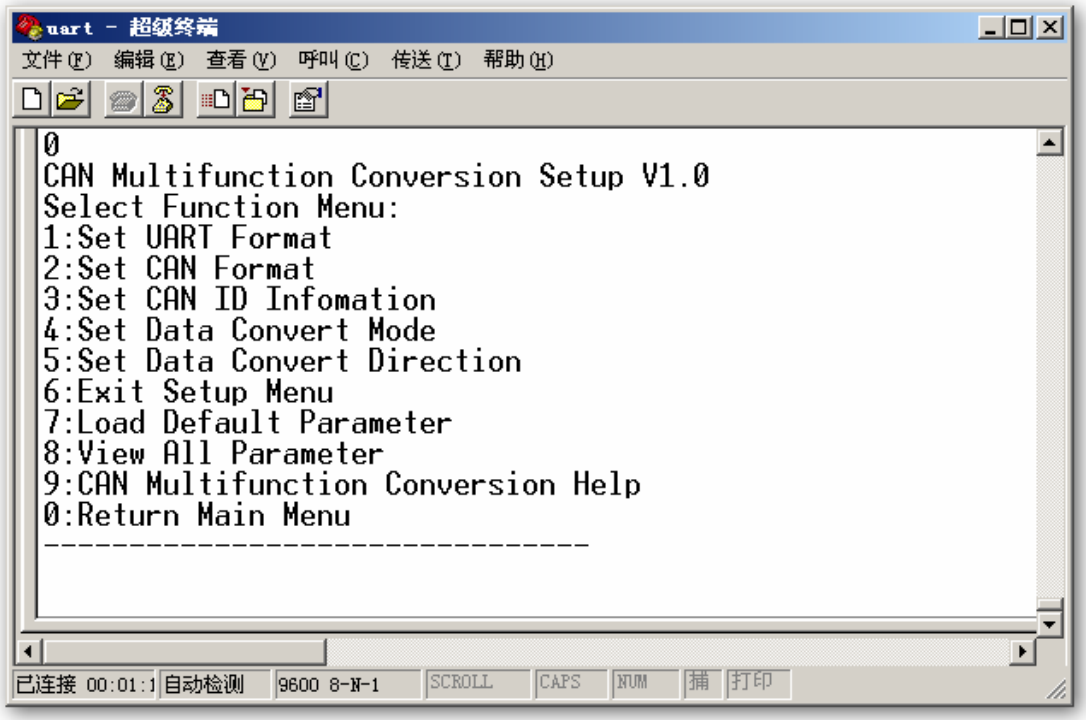


图 3.2

注意：超级终端的“属性”－“设置”－“ASCII 码设置”里的设置如图 3.3。

图 3.3

注意：每个要发送的数据必须换行和回车。以换行和回车做为输入的结尾。

3．3　设置界面介绍

设置好配置管脚后，开机，串口将显示主菜单界面。

要选择相应的选项，请在串口软件的发送区域输入相应的数字，然后按“手动发送”即可。

如输入的参数正确，则显示：

Correct,Parameter is Saved.

如输入的参数错误，则显示：

Error,Please Select Correct Number.

此时可输入“0”再调出主菜单，然后重新选择。

所有设置的参数将被保存，掉电不丢失，可发送“8”查看当前设置。

下面详细的介绍设置菜单和相应的选项。

3．3．1 主菜单：

CAN Multifunction Conversion Setup V1.0
Select Function Menu:
1:Set UART Format
2:Set CAN Format
3:Set CAN ID Infomation
4:Set Data Convert Mode
5:Set Data Convert Direction
6:Exit Setup Menu
7:Load Default Parameter
8:View All Parameter
9:CAN Multifunction Conversion Help
0:Return Main Menu

主菜单一共有 10 个选项，如下：

CAN 多功能转换器设置软件 V1.0
1、设置串口工作模式
2、设置 CAN 工作模式
3、设置 CAN ID 工作模式
4、设置数据转换模式
5、设置数据转换方向
6、退出设置菜单（进入正常工作模式）
7、调入缺省的设置
8、查看当前的设置
9、CAN 多功能转换器帮助
0、返回主菜单

输入 1－9，选择相应的菜单，输入 0 则返回主菜单。下面详细的介绍每个菜单的具体功能。

3．3．2 串口设置菜单：

在主菜单下，当输入 1 后，进入串口工作模式设置界面：

Present UART Format is:9600-8-N-1

Please Select UART Function:

1:Set UART Baud Rate

2:Set UART Data Bit

3:Set UART Parity Bit

4:Set UART Stop Bit

0:Return Main Menu

第一行显示状态为当前串口的工作模式：

状态定义如下：波特率－数据位－校验位－停止位。

如：9600－8－N－1

校验位：N：没有校验；E：偶校验；O：奇校验。

串口工作模式菜单一共有 4 个选项，解释如下：

1、设置串口波特率

2、设置串口数据位

3、设置串口校验位

4、设置串口停止位

0、返回主菜单

输入 1－4 选择相应的串口设置菜单，输入 0 则返回主菜单。其他输入无效。

*注意：串口设置的修改只在 **CAN** 通讯时有效，在配置时则仍保持 **9600-8-N-1** 格式不变。*

3．3．2．1 串口波特率设置菜单：

在串口设置菜单下，当输入 1 后，进入串口波特率设置界面：

Please Set New UART Baud Rate(bps):

1:600

2:1200

3:2400

4:4800

5:9600

6:19200

7:38400

8:57600

9:115200

0:Return Main Menu

输入 1－9 设置相应的串口波特率，输入 0 则返回主菜单。其他输入无效。

3．3．2．2 串口数据位设置菜单：

在串口设置菜单下，当输入 2 后，进入串口波特率设置界面：

Please Set New UART Data Bit(5,6,7,8):

5:Data is 5 bit
6:Data is 6 bit
7:Data is 7 bit
8:Data is 8 bit
0:Return Main Menu

输入 5－8 设置相应的串口数据位，输入 0 则返回主菜单。其他输入无效。

3．3．2．3 串口校验位设置菜单：

在串口设置菜单下，当输入 3 后，进入串口校验位设置界面：
Please Set New UART Parity Bit:
1:No Parity
2:Even Parity
3:Odd Parity
0:Return Main Menu

1、没有校验。
2、偶校验
3、奇校验
0、返回主菜单
输入 1－3 设置相应的串口校验位，输入 0 则返回主菜单。其他输入无效。

3．3．2．4 串口停止位设置菜单：

在串口设置菜单下，当输入 4 后，进入串口停止位设置界面：
Please Set New UART Stop bit:
1:1 Stop bit
2:2 Stop bit
0:Return Main Menu

1、1 个停止位
2、2 个停止位
0、返回主菜单
输入 1－2 设置相应的串口停止位，输入 0 则返回主菜单。其他输入无效。

3．3．3 CAN 设置菜单：

在主菜单下，当输入 2 后，进入 CAN 工作模式设置界面：
Present CAN Format is: Baud Rate=500kbps; Mode=V2.0B(29 ID)
Please Select CAN Function:
1:Set CAN Fixed Baud Rate
2:Set CAN Custom Baud Rate
3:Set CAN Mode
0:Return Main Menu

第一行为 CAN 当前的状态，包括工作波特率，和工作模式。

如：Baud Rate=500kbps; Mode=V2.0B(29 ID)

CAN 工作模式菜单一共有 3 个选项，解释如下：
1、设置 CAN 固定波特率，有 9 个常用的波特率可以选择；
2、设置用户自定义 CAN 波特率
3、设置 CAN 的工作模式
0、返回主菜单
输入 1－3 选择相应的 CAN 设置菜单，输入 0 则返回主菜单。其他输入无效。

3．3．3．1 CAN 固定波特率设置菜单：
在 CAN 设置菜单下，当输入 1 后，进入 CAN 固定波特率设置界面：
Please Set New CAN Baud Rate (kbps):
1:10
2:25
3:50
4:100
5:125
6:200
7:250
8:500
9:1000
0:Return Main Menu

输入 1－9 设置相应的 CAN 波特率，输入 0 则返回主菜单。其他输入无效。

3．3．3．2 CAN 自定义波特率设置菜单：
在 CAN 设置菜单下，当输入 2 后，进入 CAN 自定义波特率设置界面：
Please Input New CAN Baud Rate:(Example:125k)
Note1:10<=CAN Baud Rate <=1000
Note2:CAN Baud Rate Must be divide exactly by 12000
0:Return Main Menu

请输入要设置的 CAN 波特率，输入格式如下：　　xxxxk

注意：输入从 10－1000 的数字，后面以“k”或“K”结尾，并有换行回车。
注意：输入的频率必须能被晶振（12M）频率整除，如 500，10，20 等。否则将提示输入的参数错误。目前支持的参数见附表。
输入 0 则返回主菜单。其他输入无效。

3．3．3．3 CAN 工作模式设置菜单：
在 CAN 设置菜单下，当输入 3 后，进入 CAN 工作模式设置界面：
Please Set New CAN Mode:
1:V2.0A Mode(11 ID)
2:V2.0B Mode(29 ID)

0:Return Main Menu

1、V2.0A 工作模式，11 位 ID

2、V2.0B 工作模式，29 位 ID

0、返回主菜单

输入 1－2 设置相应的 CAN 工作模式，输入 0 则返回主菜单。其他输入无效。

3．3．4 CAN ID 设置菜单：

在主菜单下，当输入 3 后，进入 CAN ID 设置界面：

Please Set New ID Infomation:

1:Set TxID

2:Set RxID

3:Set RxMASK

0:Return Main Menu

1、设置 CAN 发送 ID

2、设置 CAN 接收 ID

3、设置 CAN 接收屏蔽 MASK ID

0、返回主菜单

输入 1－3 选择相应的 CAN ID 设置菜单，输入 0 则返回主菜单。其他输入无效。

3．3．4．1 CAN 发送 ID 设置菜单：

在 CAN ID 设置菜单下，当输入 1 后，进入 CAN 发送 ID 设置界面：

*注意：在 **CAN** 两种模式下显示的界面有所不同。*

当为 V2.0B 模式时显示如下：

Present CAN Tx ID is:0.0.0.0

Please Set New CAN Tx ID(Example:192.168.0.8):

Note1:ID1.ID2.ID3.ID4

Note2:ID4 low 3 bit invalid

0:Return Main Menu

当为 V2.0A 模式时显示如下：

Present CAN Tx ID is:0.0

Please Set New CAN Tx ID(Example:192.128):

Note1:ID1.ID2

Note2:ID2 low 5 bit invalid

0:Return Main Menu

第一行显示都当前设置的发送 ID 信息。

在 V2.0B 模式下，显示：0.0.0.0

在 V2.0A 模式下，显示：0.0

然后提示输入的 ID 信息格式

在 V2.0B 模式下：如要输入的发送 ID 为：192 168 0 8

则输入格式为：192.168.0.8
在 V2.0B 模式下 ID1，ID2，ID3，ID4 都有效。不过 ID4 的低 3 位是无效的。
ID1＝ID28..ID21；
ID2＝ID20..ID13；
ID3＝ID12..ID5；
ID4＝ID4..ID0，低 3 位无效；
在 V2.0A 模式下：如要输入的发送 ID 为：192 128
则输入格式为：192.128
在 V2.0A 模式下仅 ID1，ID2 有效。不过 ID2 的低 5 位是无效的。
ID1＝ID10..ID3；
ID2＝ID2..ID0，低 5 位无效；
如果输入格式错误，则提示错误，并返回。
关于更详细的 ID 信息请看 CAN128 的手册介绍或者附页介绍。
每个 ID 范围从 0－255
0、返回主菜单，其他输入无效。

3．3．4．2 CAN 接收 ID 设置菜单：

在 CAN ID 设置菜单下，当输入 2 后，进入 CAN 接收 ID 设置界面：
注意：在 CAN 两种模式下显示的界面有所不同。
当为 V2.0B 模式时显示如下：
Present CAN Rx ID is:0.0.0.0
Please Set New CAN Rx ID(Example:192.168.0.8):
Note1:ID1.ID2.ID3.ID4
Note2:ID4 low 3 bit invalid
0:Return Main Menu

当为 V2.0A 模式时显示如下：
Present CAN Rx ID is:0.0
Please Set New CAN Rx ID(Example:192.128):
Note1:ID1.ID2
Note2:ID2 low 5 bit invalid
0:Return Main Menu

具体设置方法参见“*3．3．4．1 CAN 发送 ID 设置菜单*”

3．3．4．3 CAN 接收屏蔽设置菜单：

在 CAN ID 设置菜单下，当输入 3 后，进入 CAN 接收屏蔽 ID 设置界面：
注意：在 CAN 两种模式下显示的界面有所不同。
当为 V2.0B 模式时显示如下：
Present CAN Rx Mask is:0.0.0.0
Please Set New CAN Rx Mask ID(Example:192.168.0.8):
Note1:ID1.ID2.ID3.ID4
Note2:ID4 low 3 bit invalid
0:Return Main Menu

当为 V2.0A 模式时显示如下：
Present CAN Rx Mask ID is:0.0
Please Set New CAN Rx Mask ID(Example:192.128):
Note1:ID1.ID2
Note2:ID2 low 5 bit invalid
0:Return Main Menu

具体设置方法参见“*3．3．4．1 CAN 发送 ID 设置菜单*”

3．3．5 数据转换模式设置菜单：

在主菜单下，当输入 4 后，进入数据转换模式设置界面：
Present Data Convert Mode is:Normal
Please Set New Data Convert Mode:
1:Normal Mode
2:Normal With RxID Mode
3:Normal With TxID Mode
4:Normal With RxID and TxID Mode
5:MODBUS RTU Mode
6:MODBUS ASCII Mode
0:Return Main Menu

第一行显示当前设置模式。
1、透明模式
2、透明模式带接收 ID
3、透明模式带发送 ID
4、透明模式带接收 ID 和发送 ID
5、MODBUS RTU 转换模式
6、MODBUS ASCII 转换模式
0、返回主菜单
透明模式即串口和 CAN 的数据流相互透明的传输；
透明模式带接收 ID 即当接收到 CAN 端数据后，将发送当前配置的 ID 信息和透明的数据流到 UART，反方向则透明传输；
透明模式带发送 ID 即当接收到 UART 端数据后，将 UART 输入的前 2 个或 4 个字节（根据 CAN 的模式不同）做为发送 CAN 的 ID，其后则为数据；而反方向则透明传输
透明模式带接收 ID 和发送 ID 则是模式 2 和 3 的混合。
MODBUS RTUH 和 ASCII 转换模式。
具体请参见第四章的详细说明。
输入 1－6 设置相应的数据工作模式，输入 0 则返回主菜单。其他输入无效。

3．3．6 数据转换方向设置菜单：

在主菜单下，当输入 5 后，进入转换方向设置界面：
Present Data Convert Direction is:Bidirection UART<-->CAN

Please Select New Data Convert Direction:
1:Bidirection UART<-->CAN
2:Only UART--->CAN
3:Only CAN--->UART
0:Return Main Menu

第一行显示当前设置模式。
1、双向转换，转换器将 CAN 端的数据发送到 UART，并把 UART 端的数据发送到 CAN 端。
2、单向转换，转换器只将 UART 端的数据发送到 CAN 端。
3、单向转换，转换器只将 CAN 端的数据发送到 UART 端。
0、返回主菜单
输入 1－3 设置相应的数据转换方向，输入 0 则返回主菜单。其他输入无效。

3．3．7 退出设置菜单：

当输入 6 后，退出设置菜单，进入 CAN 转换功能，显示：
Config is OK,Exit setup Menu
注意：此时进入转换功能，串口设置将不再工作。除非重新上电。

3．3．8 调入缺省的设置菜单：

在主菜单下，当输入 7 后，将调入缺省的设置菜单，显示：
Please input 'y' confirm or any key to return Menu

如果确认调入缺省值，则输入“y”或“Y”，否则请按其他按键返回主菜单。
当选择确认后，程序将调入出厂的设置模式：
Default Parameter is Load:
Present UART Format is:9600-8-N-1
Present CAN Format is:Baud Rate=500kbps; Mode=V2.0B(29 ID)
Present CAN Tx ID is:0.0.0.0
Present CAN Rx ID is:0.0.0.0
Present CAN Rx Mask is:0.0.0.0
Present Data Convert Mode is:Normal
Present Data Convert Direction is:Bidirection UART<-->CAN

出厂的工作模式如上。

3．3．9 查看当前设置菜单：

在主菜单下，当输入 8 后，查看当前设置菜单，显示当前的设置：
Default Parameter is Load:
Present UART Format is:9600-8-N-1
Present CAN Format is:Baud Rate=500kbps; Mode=V2.0B(29 ID)
Present CAN Tx ID is:0.0.0.0
Present CAN Rx ID is:0.0.0.0
Present CAN Rx Mask is:0.0.0.0

Present Data Convert Mode is:Normal
Present Data Convert Direction is:Bidirection UART<-->CAN

3．3．10　CAN 多功能转换器帮助：

当输入 9 后，查看 CAN 多功能转换器联系方式：
CAN Multifunction Conversion:
Hardware Ver:1.00
Software Ver:1.00
Double Dragon Co.Ltd.
Copyright (c) 2006.
Tel:010-82623550,3551

四、转换格式

SL-CAN 转换器有 4 种工作方式，包括透明转换；透明带标识转换，MODBUS RTU 和 MODBUS ASCII 转换。下面详细的对这 4 种转换模式进行介绍：

4．1　透明转换

“透明转换”的含义就是将一种格式的总线数据原样转换成另一种总线的数据格式，不附加对数据进行修改，即内容没有任何变化。对于总线两端来看数据如同透明的一样。

这种方式的传输结构如下所示：

UART 数据 －－－》 CAN 数据

UART 数据 《－－－ CAN 数据

4．1．1　串行帧转 CAN 报文

串行帧的全部数据依次填充到 CAN 报文帧的数据域里，转换器接收到串行数据上有数据后立即接收并进行转换。如图 4.1，图 4.2 所示。

本转换器按照如下方式进行发送，当串口有连续不断的数据发送时，当接收数据超过 8 个则开始发送，持续接收串口数据并发送。如果接收到串口的数据小于 8 个，则等待 3 个字节的串口传输时间（参见图 4.5），如果仍没有数据接收到，则将发送该数据。

转换成的 CAN 报文帧信息（帧类型部分）和帧 ID 来自用户配置，并且在转换过程中帧类型和帧 ID 一直保持不变。数据转换对应格式如图所示。

如果串行帧的字节数小于等于 8 字节，依次序将字符 1 到 n（n 为串行帧长度）填充到 CAN 报文的数据域的 1 到 n 个字节位置。

如果串行帧的字节数大于 8，那么单片机从串行帧首个字符开始，第一次取 8 个字符依次填充到 CAN 报文的数据域。将数据发送到 CAN 总线后，在转换余下的串行帧数据填充到 CAN 报文的数据域，直到其数据被转换完。

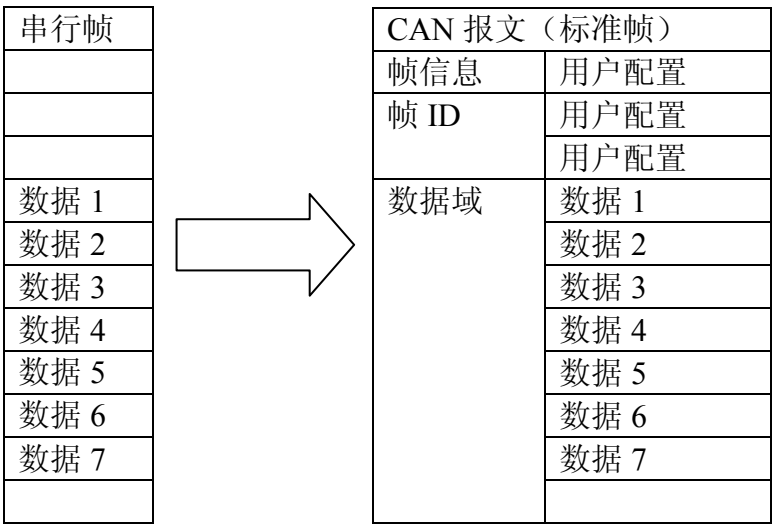
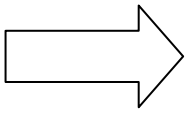


图 4.1　串口转 CAN 报文，（数据小于 8）

如果数据大于 8 个，则转换如下：假设帧 ID 为：00 90

串行帧
数据 1
数据 2
数据 3
数据 4
数据 5
数据 6
数据 7
数据 8
数据 9
数据 10
数据 11
数据 12



	CAN 报文 1	CAN 报文 2
帧信息	08	04
帧 ID	00	00
	90	90
数据域	数据 1	数据 9
	数据 2	数据 10
	数据 3	数据 11
	数据 4	数据 12
	数据 5	
	数据 6	
	数据 7	
	数据 8	

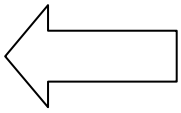
图 4.2　串口转 CAN 报文（数据大于 8）

*注：如果为扩展帧，则帧 **ID** 为 **4** 个字节*

4．1．2　　CAN 报文转串行帧

对于 CAN 总线的报文收到一帧就立即发送一帧。数据对应的格式如图 4.3 所示。

串行帧
数据 1
数据 2
数据 3
数据 4
数据 5
数据 6
数据 7

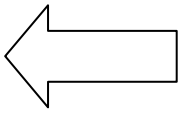


CAN 报文（标准帧）	
帧信息	用户配置
帧 ID	用户配置
	用户配置
数据域	数据 1
	数据 2
	数据 3
	数据 4
	数据 5
	数据 6
	数据 7

图 4.3　CAN 转串口报文（数据小于 8）

如果数据大于 8 个，则转换如下：假设帧 ID 为：00 90

串行帧
数据 1
数据 2
数据 3
数据 4
数据 5
数据 6
数据 7
数据 8
数据 9
数据 10
数据 11
数据 12



	CAN 报文 1	CAN 报文 2
帧信息	08	04
帧 ID	00	00
	90	90
数据域	数据 1	数据 9
	数据 2	数据 10
	数据 3	数据 11
	数据 4	数据 12
	数据 5	
	数据 6	
	数据 7	
	数据 8	

图 4.4　CAN 转串口报文（数据大于 8）

注：如果为扩展帧，则帧 ***ID*** *为* ***4*** *个字节*

4．2　透明带标识转换

“透明带标识转换”就是在转换的时候根据设置的不同在某一个方向增加标识的转换模式。

透明带接收 ID 传输结构如下所示：

UART 数据　　－－－》 CAN 数据

UART 数据　　《－－－ CAN 数据＋帧 ID（配置）

透明带发送 ID 传输结构如下所示：

UART 数据（地址） －－－》 CAN 数据

UART 数据　　《－－－ CAN 数据

透明带发送和接收 ID 传输结构如下所示：

UART 数据（地址） －－－》 CAN 数据

UART 数据　　《－－－ CAN 数据＋帧 ID（配置）

其中“帧 ID”发送到串口开始的 2 个字节或 4 个字节（根据当前设置的 CAN 的工作模式有关），然后再发送数据。

而 UART 数据的“地址”信息转换到 CAN 报文的标识域中。地址信息为串行数据的起始 2 个字节或 4 个字节（根据当前设置的 CAN 的工作模式有关），

4．2．1　UART 数据帧的结构

由于在串行数据中必须取得完整的串行数据帧，转换器以两帧之间的时间间隔做为帧的划分。目前设置为 3 个串口发送的时间和。串行帧最大的长度为缓冲区的长度：255 个字节。

转换器在串行总线空闲状态下初始化接收，如果此时有串口数据接收则做为一帧的开始，传输中改帧内的字符之间的时间间隔必须小于或等于 3 个字符的时间。

如果转换器在接收到一个字符后的 3 个字符时间内没有继续收到字符，则转换器认为改帧接收结束。帧格式如图 4.5 所示：

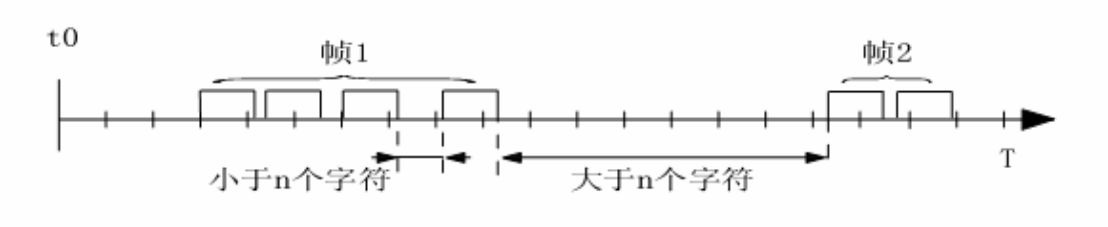


图 4.5　串行帧时间格式

其中图中 n＝3。

例如在 9600－8－N－1 串口设置状态下：1 个字符的传输时间为：

1/9600bit/s×10bit =0.00104 2s = 1042us

3 个字符的传输时间则为：3×1042＝3126us。

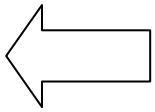
4．2．2　透明带接收 ID 转换格式

“透明带接收 ID”是将 CAN 数据的“帧 ID”信息发送到串口开始的 2 个字

节或 4 个字节（根据当前设置的 CAN 的工作模式有关），然后再发送数据。

如果为标准帧，则帧 ID 为 2 个字节，如图 4.6 所示：

串行帧
帧 ID 1
帧 ID 2
数据 1
数据 2
数据 3
数据 4
数据 5
数据 6
数据 7
数据 8
数据 9
数据 10

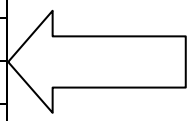


	CAN 报文 1	CAN 报文 2
帧信息	08	02
帧 ID	帧 ID 1	帧 ID 1
	帧 ID 2	帧 ID 2
数据域	数据 1	数据 9
	数据 2	数据 10
	数据 3	
	数据 4	
	数据 5	
	数据 6	
	数据 7	
	数据 8	

图 4.6　串口转 CAN 报文（透明带接收标识）

如果为扩展帧，则帧 ID 为 4 个字节，如图 4.7 所示：

串行帧
帧 ID 1
帧 ID 2
帧 ID 3
帧 ID 4
数据 1
数据 2
数据 3
数据 4
数据 5
数据 6
数据 7
数据 8
数据 9
数据 10



	CAN 报文 1	CAN 报文 2
帧信息	08	02
帧 ID	帧 ID 1	帧 ID 1
	帧 ID 2	帧 ID 2
	帧 ID 3	帧 ID 3
	帧 ID 4	帧 ID 4
数据域	数据 1	数据 9
	数据 2	数据 10
	数据 3	
	数据 4	
	数据 5	
	数据 6	
	数据 7	
	数据 8	

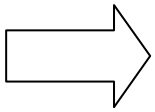
图 4.7　串口转 CAN 报文（透明带接收标识）

而串口发送到 CAN 的数据则是透明传输：
具体请参见“*4．1．1　串行帧转 CAN 报文*”

4．2．3　透明带发送 ID 转换格式

“透明带发送 ID”是将串行数据的“地址”信息转换到 CAN 报文的标识域中。地址信息为串行数据的起始 2 个字节或 4 个字节（根据当前设置的 CAN 的工作模式有关），如图 4.8 所示：

串行帧
帧 ID 1
帧 ID 2
数据 1
数据 2
数据 3
数据 4
数据 5
数据 6
数据 7
数据 8
数据 9
数据 10

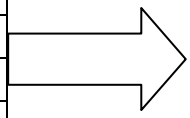


	CAN 报文 1	CAN 报文 2
帧信息	08	02
帧 ID	帧 ID 1	帧 ID 1
	帧 ID 2	帧 ID 2
数据域	数据 1	数据 9
	数据 2	数据 10
	数据 3	
	数据 4	
	数据 5	
	数据 6	
	数据 7	
	数据 8	

图 4.8　串口转 CAN 报文（透明带发送标识）

注意：帧 ID2 只有高 3 位有效，低 5 位将不予处理。
如果为扩展帧，则帧 ID 为 4 个字节，如图 4.9 所示：

串行帧
帧 ID 1
帧 ID 2
帧 ID 3
帧 ID 4
数据 1
数据 2
数据 3
数据 4
数据 5
数据 6
数据 7
数据 8
数据 9
数据 10



	CAN 报文 1	CAN 报文 2
帧信息	08	02
帧 ID	帧 ID 1	帧 ID 1
	帧 ID 2	帧 ID 2
	帧 ID 3	帧 ID 3
	帧 ID 4	帧 ID 4
数据域	数据 1	数据 9
	数据 2	数据 10
	数据 3	
	数据 4	
	数据 5	
	数据 6	
	数据 7	
	数据 8	

图 4.9　串口转 CAN 报文（透明带发送标识）

注意：帧 ID4 只有高 5 位有效，低 3 位将不予处理。

而 CAN 发送到串口的数据则是透明传输：
具体请参见“*4．1．2　CAN 报文转串行帧*”

4．2．4　透明带发送 ID 和接收 ID 转换格式

“透明带发送 ID 和接收 ID”模式等同于“透明带发送 ID”和“透明带接收 ID”。即双向都进行 ID 的发送和接收。

对于 CAN 端，是将 CAN 数据的“帧 ID”信息发送到串口开始的 2 个字节或 4 个字节（根据当前设置的 CAN 的工作模式有关），然后再发送数据。
具体参见*“4．2。2 透明带接收 ID 转换格式”*

对于串口端，是将串行数据的“地址”信息转换到 CAN 报文的标识域中。地址信息为串行数据的起始 2 个字节或 4 个字节（根据当前设置的 CAN 的工作模式有关）。具体参见*“4．2。3 透明带发送 ID 转换格式”*

4．3　MODBUS RTU 转换

“MODBUS RTU 协议转换”是为了支持标准的 MODBUS 协议而建立的，在串口侧使用的是标准的 MODBUS RTU 协议，可以和其他标准的 MODBUS RTU 设备接口。

在 CAN 总线侧使用的是一个简单易用的分段协议来传输 MODBUS 协议。这样就能轻松的在串行网络和 CAN 网络之间来实现 MODBUS 协议的通讯。

4．3．1 串行口帧格式

串行接口采用的是标准的 MODBUS RTU 协议，所以用户帧符合此协议即可（参见附录：MODBUS RTU 协议）。如果传输的帧不符合 MODBUS RTU 格式，那么转换器会将接收到的帧丢弃，而不予转换。

转换器采用的 MODBUS RTU 传输格式是 1 起始位、8 数据位和 1 停止位。请设置正确的串口格式。MODBUS RTU 帧长度最大为缓冲区长度：255 字节。

4．3．2 CAN 帧格式

CAN 侧的设备要采用 MODBUS 协议则需要为之定义一种可靠的传输格式，这里采用一种分段协议实现，其定义了一个长度大于 8 字节的信息进行分段以及重组的方法。分段传送协议的制定参考了 DeviceNet 中分段报文的传送协议。分段报文格式如下表（以扩展帧为例，标准帧只是帧 ID 的长度不同而已，其他格式相同），传输的 MODBUS 协议内容即可从“数据 2”字节开始，如果协议内容大于 7 个字节，那么将剩下的协议内容按照这种分段格式继续转换，直到转换完成。

CAN 总线帧格式说明如图 4.10 所示：

		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	（报文识别码）ID.28..ID.21							
字节 3	帧 ID2	ID.20..ID.13							
字节 4	帧 ID3	ID.12..ID.5							
字节 5	帧 ID4	ID.4..ID.0					X	X	X
字节 6	数据 1	分段标记	分段类型		分段计数器				
字节 7	数据 2	字符 1							
字节 8	数据 3	字符 2							
字节 9	数据 4	字符 3							
字节 10	数据 5	字符 4							
字节 11	数据 6	字符 5							
字节 12	数据 7	字符 6							
字节 13	数据 8	字符 7							

图 4.10

- 分段报文标记：表明该报文是否是分段报文。该位为 0 示单独报文，为 1 表

示属于被分段报文中的一帧。

- 分段类型：表明是第一段，中间段，还是最后段。如图 4.11 所示：

位值	含义	说明
0	第一个分段	如果分段计数器包含值 0，那么这是分段系列中的第一段
1	中间分段	表明这是一个中间分段
2	最后分段	标志最后一个分段

图 4.11

- 分段计数器：每一个段的标志，该段在整个报文中的序号，如果是第几个段那么计数器的值就是几。这样在接收时就能够验证是否有分段被遗失。

4．3．3　转换方式

在串口侧向 CAN 侧转换的过程中，转换器只会在接收到一完整正确的 MODBUS RTU 帧才会进行转换，否则无动作。

消息发送至少要以 3.5 个字符时间的停顿间隔开始。在网络波特率下多样的字符时间，这是最容易实现的(如下图的 T1-T2-T3-T4 所示)。传输的第一个域是设备地址。可以使用的传输字符是十六进制的 0...9,A...F。在最后一个传输字符之后，一个至少 3.5 个字符时间的停顿标定了消息的结束。一个新的消息可在此停顿后开始。一典型的消息帧如图 4.12 所示：

起始位	设备地址	功能代码	数据	CRC 校验	结束符
T1-T2-T3-T4	8bit	8bit	n 个 8bit	16bit	T1-T2-T3-T4

图 4.12　　RTU 消息帧

MODBUS RTU 协议的地址域转换成 CAN 报文中帧 ID 的高字节（无论是标准帧还是扩展帧都是帧 ID1——ID.28-ID.21（扩展帧）或 ID.10-ID3（标准帧），在转换该帧的过程中标识不变。 如图 4.13 所示：

MODBUS RTU 帧
地址域
功能码
数据域

	CAN 报文 1	CAN 报文 X
帧信息	帧信息	帧信息
帧 ID	地址域	地址域
	0	0
	0	0
	0	0
数据域	分段协议	分段协议
	功能码	数据域
	数据域	

图 4.13

而 CRC 校验字节不转换到 CAN 报文中 CAN 的报文中也不必带有串行帧的校验字节，因为 CAN 总线本身就有较好的校验机制。

转换的是 MODBUS RTU 的协议内容——功能码和数据域，转换时将它们依次转换在 CAN 报文帧的数据域（从第二个数据字节开始，第一个数据字节为分段协议使用）里，由于 MODBUS RTU 帧的长度根据功能码的不同而不同。而 CAN 报文一帧只能传送 7 个数据，所以转换器会将较长的 MODBUS RTU 帧分段转换成 CAN 的报文后用上述的 CAN 分段协议发出。用户在 CAN 的节点上接收时取功能码和数据域处理即可。

对于 CAN 总线的 MODBUS 协议数据，无需做循环冗余校验（CRC16），转换器按照分段协议接收，接收完一帧解析后自动加上循环冗余校验（CRC16），转换成 MODBUS RTU 帧发送至串行总线。

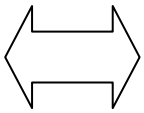
如果接收到的数据不符合分段协议，则将该组数据丢弃不予转换。

4．3．4　转换示例

在配置成扩展帧情况下，如图 4.14 所示，在 MODBUS RTU 帧转换成 CAN 报文时，将地址 0x08 直接填充到帧 ID1，其他帧 ID 填 0x00，在转换该帧的过程中保持此帧 ID 不变。

当一帧 CAN 报文处理不完一帧 MODBUS 报文时，CAN 报文采用分段协议。每个 CAN 报文的“数据 1”都用来填充分段信息（0x81，0xC2），该信息不转换到 MODBUS RTU 帧当中，仅作为帧格式用来确认帧的信息。功能码和数据域的值则依次填入 CAN 报文的数据 2～8 中。

MODBUS RTU 帧	
地址域	08
功能码	11
数据域	00
	01
	00
	02
	04
	00
	0A
	01
	02
CRC 域	ED
	69



	CAN 报文 1	CAN 报文 X
帧信息	08	04
帧 ID	08	08
	0	0
	0	0
	0	0
数据 1	81	C2
数据 2	11	0A
数据 3	00	01
数据 4	01	02
数据 5	00	
数据 6	02	
数据 7	04	
数据 8	00	

图 4.14

4．4　MODBUS ASCII 转换

“MODBUS ASCII 协议转换”是为了支持标准的 MODBUS 协议而建立的，在串口侧使用的是标准的 MODBUS ASCII 协议，可以和其他标准的 MODBUS ASCII 设备接口。

4．4．1 串行口帧格式

串行接口采用的是标准的 MODBUS ASCII 协议，所以用户帧符合此协议即可（参见附录：MODBUS ASCII 协议）。如果传输的帧不符合 MODBUS ASCII 格式，那么转换器会将接收到的帧丢弃，而不予转换。

转换器采用的 MODBUS ASCII 传输格式是 1 起始位、7 数据位和 1 停止位。请设置正确的串口格式。MODBUS ASCII 帧长度最大为缓冲区长度：255 字节。

4．4．2 CAN 帧格式

CAN 侧的设备要采用 MODBUSASCII 协议则和 MODBUS RTU 一样的协议，具体参见：***4．3．2 CAN 帧格式。***

4．4．3　转换方式

在串口侧向 CAN 侧转换的过程中，转换器只会在接收到一完整正确的 MODBUS ASCII 帧才会进行转换，否则无动作。

MODBUS ASCII 协议消息帧，消息以冒号（:）字符（ASCII 码 3AH）开始，以回车换行符结束（ASCII 码 0DH,0AH）。其它域可以使用的传输字符是十六进制的 0...9,A...F。网络上的设备不断侦测“:”字符，当有一个冒号接收到时，每个设备都解码下个域（地址域）来判断是否发给自己的。

消息中字符间发送的时间间隔最长不能超过 1 秒，否则接收的设备将认为传输错误。一个典型消息帧如图 4.15 所示：

起始位	设备地址	功能代码	数据	LRC 校验	结束符
1 个字符	2 个字符	2 个字符	n 个字符	2 个字符	2 个字符

图 4.15

MODBUS ASCII 协议的起始位“：”1 个字符和结束符换行回车 2 个字符不转换到 CAN 报文中 ，因为 CAN 总线本身可以自动进行帧处理。接收端在接收到一帧解析后自动加上起始位“：”和结束符换行回车位发送至串行总线。

在转换过程中，接收到的地址域，功能码和数据域的 ASCII 字符进行重组，即每 2 个 ASCII 字符组成一个 8 位 HEX 字节。这样在传输中可以大大提高 CAN 总线的利用率。而 CAN 接收到一帧后也对一个 HEX 字节恢复成 2 个 ASCII 字符，的转换方法如图 4.16 所示：

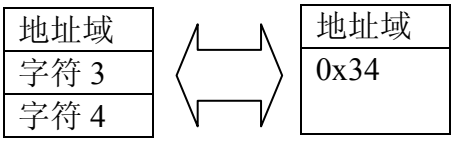


图 4.16

MODBUS ASCII 协议的地址域转换成 CAN 报文中帧 ID 的高字节（无论是标准帧还是扩展帧都是帧 ID1——ID.28-ID.21（扩展帧）或 ID.10-ID3（标准帧），在转换该帧的过程中标识不变。 如图 4.17 所示：

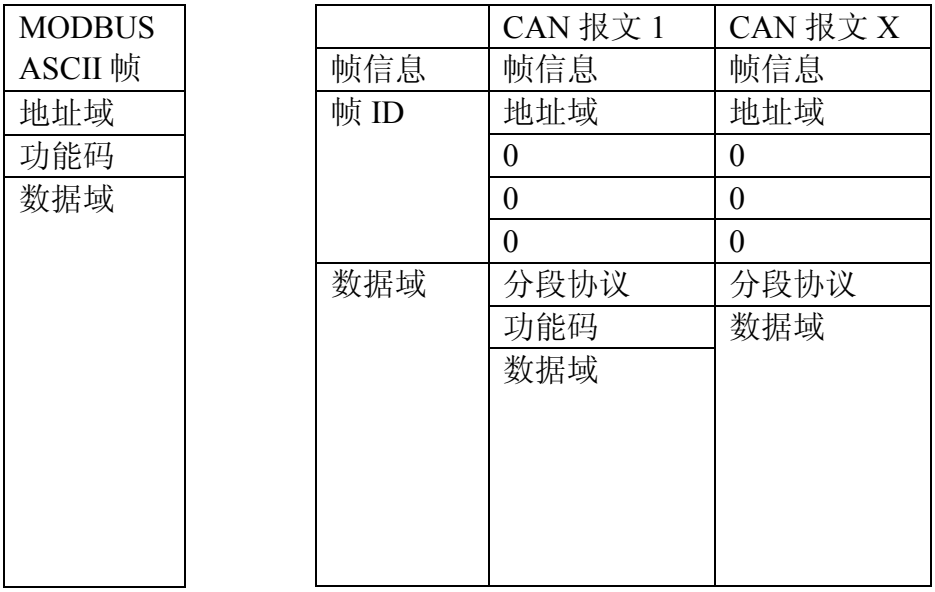


图 4.17

而 LRC 校验字节不转换到 CAN 报文中 CAN 的报文中也不必带有串行帧的校验字节，因为 CAN 总线本身就有较好的校验机制。

转换的是 MODBUS ASCII 的协议内容——功能码和数据域，转换时将它们依次转换在 CAN 报文帧的数据域（从第二个数据字节开始，第一个数据字节为分段协议使用）里，由于 MODBUS ASCII 帧的长度根据功能码的不同而不同。而 CAN 报文一帧只能传送 7 个数据，所以转换器会将较长的 MODBUS ASCII 帧分段转换成 CAN 的报文后用上述的 CAN 分段协议发出。用户在 CAN 的节点上接收时取功能码和数据域处理即可。

对于 CAN 总线的 MODBUS ASCII 协议数据，无需做纵向冗余错误校验（LRC16），转换器按照分段协议接收，接收完一帧解析后自动加上纵向冗余错误校验（LRC16），转换成 MODBUS ASCII 帧发送至串行总线。

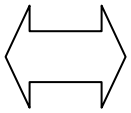
如果接收到的数据不符合分段协议，则将该组数据丢弃不予转换。

4．3．4　转换示例

在配置成扩展帧情况下，如图 4.18 所示，在 MODBUS ASCII 帧转换成 CAN 报文时，将地址 0x08 直接填充到帧 ID1，其他帧 ID 填 0x00，在转换该帧的过程中保持此帧 ID 不变。

当一帧 CAN 报文处理不完一帧 MODBUS ASCII 报文时，CAN 报文采用分段协议。每个 CAN 报文的“数据 1”都用来填充分段信息（0x81，0xC2），该信息不转换到 MODBUS ASCII 帧当中，仅作为帧格式用来确认帧的信息。功能码和数据域的值则依次填入 CAN 报文的数据 2～8 中。

MODBUS ASCII 帧	
地址域	08
功能码	11
数据域	00
	01
	00
	02
	04
	00
	0A
	01
	02
LRC 域	D3



	CAN 报文 1	CAN 报文 X
帧信息	08	04
帧 ID	08	08
	0	0
	0	0
	0	0
数据 1	81	C2
数据 2	11	0A
数据 3	00	01
数据 4	01	02
数据 5	00	
数据 6	02	
数据 7	04	
数据 8	00	

图 4.18

五、附录

5．1　CAN 协议帧格式

5．1．1 CAN2.0B 标准帧格式

CAN 标准帧信息是 11 个字节，包括帧的信息和帧数据两部分。前 3 个字节为帧的信息部分，格式如附图 5.1：

		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	（报文识别码）ID.10..ID3							
字节 3	帧 ID2	ID.2..ID.0			X	X	X	X	X
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							

附图 5.1

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）表示数据帧，RTR＝0 表示数据帧，RTR＝1 表示远程帧；
第 3－0 位表示数据的长度；
字节 2，3 为报文识别码，高 11 位有效；
字节 4，11 为数据帧的实际数据，远程帧时无效。

5．1．2 CAN2.0B 扩展帧格式

CAN 标准帧信息是 13 个字节，包括帧的信息和帧数据两部分。前 5 个字节为帧的信息部分，格式如下附图 5.2：

		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	（报文识别码）ID.28..ID.21							
字节 3	帧 ID2	ID.20..ID.13							
字节 4	帧 ID3	ID.12..ID.5							
字节 5	帧 ID4	ID.4..ID.0					X	X	X
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							

附图 5.2

字节 1 为帧信息，第 7 位（FF）表示帧格式，在扩展帧中 FF＝1；
第 6 位（RTR）表示数据帧，RTR＝0 表示数据帧，RTR＝1 表示远程帧；
第 3－0 位表示数据的长度；
字节 2，5 为报文识别码，高 29 位有效；
字节 6，13 为数据帧的实际数据，远程帧时无效。

5．2　CAN 波特率

CAN 波特率由单片机的 CANBT1,CANBT2,CANBT3 共同设置。下表列出了其中的推荐的值，见附表三。标注*符号的值是由国际 CiA 协会推荐的标准值。注：CAN 晶体振荡频率采用 12M。

序号	分频比	波特率（**kbps**）	序号	分频比	波特率（**kbps**）
1	12*	1000	17*	120	100
2	15*	800	18	125	96
3	16	750	19	150	80
4	20	600	20	160	75
5	24*	500	21	200	60
6	25	480	22*	240	50
7	30	400	23	250	48
8	32	375	24	300	40
9	40	300	25	375	32
10	48*	250	26	400	30
11	50	240	27	480	25
12	60	200	28	500	24
13	75	160	29*	600	20
14	80	150	30	800	15
15	96*	125	31	1000	12
16	100	120	32*	1200	10

附图 5.3

当用户输入自定义波特率时，必须符合上表的波特率，否则将认为输入参数错误。

注意：输入的波特率必须能被 12000 整除。并且波特率大等于 10kbps，小等于 1000kbps。

5．3　CAN ID 设置

SL-CAN 转换器 ID 的详细设置请参加 AT90CAN32 单片机。现部分摘抄如下：
在标准帧（V2.0A）下，ID 屏蔽寄存器如下：

V2.0 part A

Bit	15/7	14/6	13/5	12/4	11/3	10/2	9/1	8/0	
	-	-	-	-	-	RTRMSK	-	IDEMSK	CANIDM4
	-	-	-	-	-	-	-	-	CANIDM3
	IDMSK2	IDMSK1	IDMSK0	-	-	-	-	-	CANIDM2
	IDMSK10	IDMSK9	IDMSK8	IDMSK7	IDMSK6	IDMSK5	IDMSK4	IDMSK3	CANIDM1
Bit	31/23	30/22	29/21	28/20	27/19	26/18	25/17	24/16	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	-	-	-	-	-	-	-	-	

附图 5.4

在扩展帧（V2.0B）下，ID 屏蔽寄存器如下：

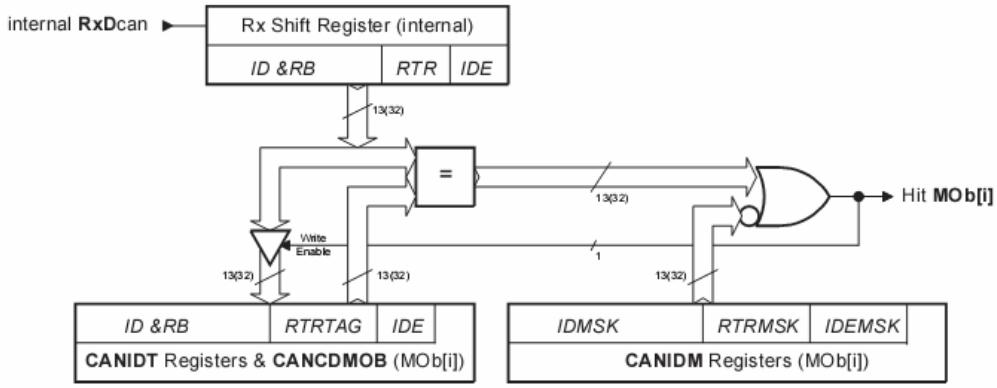
V2.0 part B

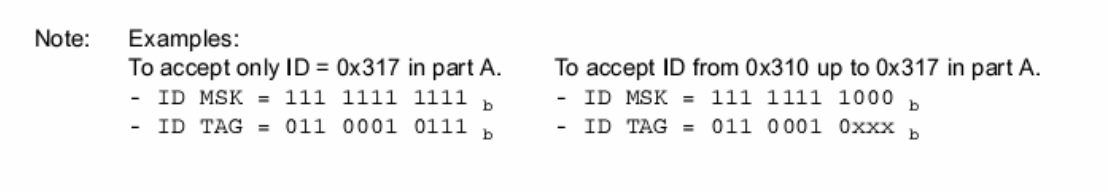
Bit	15/7	14/6	13/5	12/4	11/3	10/2	9/1	8/0	
	IDMSK4	IDMSK3	IDMSK2	IDMSK1	IDMSK0	RTRMSK	-	IDEMSK	CANIDM4
	IDMSK12	IDMSK11	IDMSK10	IDMSK9	IDMSK8	IDMSK7	IDMSK6	IDMSK5	CANIDM3
	IDMSK20	IDMSK19	IDMSK18	IDMSK17	IDMSK16	IDMSK15	IDMSK14	IDMSK13	CANIDM2
	IDMSK28	IDMSK27	IDMSK26	IDMSK25	IDMSK24	IDMSK23	IDMSK22	IDMSK21	CANIDM1
Bit	31/23	30/22	29/21	28/20	27/19	26/18	25/17	24/16	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	
Initial Value	-	-	-	-	-	-	-	-	

附图 5.5

屏蔽寄存器工作如下：

Figure 20-10. Acceptance Filter Block Diagram





附图 5.6

如要接收所有的 ID 信息，则 CANIDM ＝ 0。
即配置模式下输入：-0-0-0-0-，参加 3.3.4.3 章介绍。

如果要接收 ID ＝ 8 的信息，则 CANIDM ＝ 8。
即配置模式下输入：-0-8-0-0-，参加 3.3.4.3 章介绍。

5．4　MODBUS 协议简介

Modbus 协议是应用于电子控制器上的一种通用语言。通过此协议，控制器相互之间、控制器经由网络（例如以太网）和其它设备之间可以通信。它已经成为一通用工业标准。有了它，不同厂商生产的控制设备可以连成工业网络，进行集中监控。

此协议定义了一个控制器能认识使用的消息结构,而不管它们是经过何种网络进行通信的。它描述了一控制器请求访问其它设备的过程，如果回应来自其它设备的请求，以及怎样侦测错误并记录。它制定了消息域格局和内容的公共格式。

当在一 Modbus 网络上通信时，此协议决定了每个控制器须要知道它们的设备地址，识别按地址发来的消息，决定要产生何种行动。如果需要回应，控制器将生成反馈信息并用 Modbus 协议发出。在其它网络上，包含了 Modbus 协议的消息转换为在此网络上使用的帧或包结构。这种转换也扩展了根据具体的网络解决节地址、路由路径及错误检测的方法。

标准的 Modbus 口是使用一 RS-232C 兼容串行接口，它定义了连接口的针脚、电缆、信号位、传输波特率、奇偶校验。控制器能直接或经由 Modem 组网。

控制器通信使用主—从技术，即仅一设备（主设备）能初始化传输（查询）。其它设备（从设备）根据主设备查询提供的数据作出相应反应。典型的主设备：主机和可编程仪表。典型的从设备：可编程控制器。

主设备可单独和从设备通信，也能以广播方式和所有从设备通信。如果单独通信，从设备返回一消息作为回应，如果是以广播方式查询的，则不作任何回应。Modbus 协议建立了主设备查询的格式：设备（或广播）地址、功能代码、所有要发送的数据、一错误检测域。

从设备回应消息也由 Modbus 协议构成，包括确认要行动的域、任何要返回的数据、和一错误检测域。如果在消息接收过程中发生一错误，或从设备不能执行其命令，从设备将建立一错误消息并把它作为回应发送出去。

控制器能设置为两种传输模式（ASCII 或 RTU）中的任何一种在标准的 Modbus 网络通信。用户选择想要的模式，包括串口通信参数（波特率、校验方式等），在配置每个控制器的时候，在一个 Modbus 网络上的所有设备都必须选择相同的传输模式和串口参数。

下面对 MODBUS 的 RTU 和 ASCII 协议进行一个简单的介绍，关于 MODBUS 的更多更详细信息请参考 MODBUS 协议的完整资料（可以从互联网上获得）。

5．4．1　　MODBUS RTU 协议

当控制器设为在 Modbus 网络上以 RTU（远程终端单元）模式通信，在消息中的每个 8Bit 字节包含两个 4Bit 的十六进制字符。这种方式的主要优点是：在同样的波特率下，可比 ASCII 方式传送更多的数据。

- **代码系统**

  8 位二进制，十六进制数 0...9，A...F

  消息中的每个 8 位域都是一个两个十六进制字符组成

- **每个字节的位**

  1 个起始位

  8 个数据位，最小的有效位先发送

  1 个奇偶校验位，无校验则无

  1 个停止位（有校验时），2 个 Bit（无校验时）

- **错误检测域**

  CRC(循环冗长检测)

使用 RTU 模式，消息发送至少要以 3.5 个字符时间的停顿间隔开始。在网络波特率下多样的字符时间，这是最容易实现的(如下图的 T1-T2-T3-T4 所示)。传输的第一个域是设备地址。可以使用的传输字符是十六进制的 0...9,A...F。网络设备不断侦测网络总线，包括停顿间隔时间内。当第一个域（地址域）接收到，每个设备都进行解码以判断是否发往自己的。在最后一个传输字符之后，一个至少 3.5 个字符时间的停顿标定了消息的结束。一个新的消息可在此停顿后开始。

整个消息帧必须作为一连续的流转输。如果在帧完成之前有超过 1.5 个字符时间的停顿时间，接收设备将刷新不完整的消息并假定下一字节是一个新消息的地址域。同样地，如果一个新消息在小于 3.5 个字符时间内接着前个消息开始，接收的设备将认为它是前一消息的延续。这将导致一个错误，因为在最后的 CRC 域的值不可能是正确的。一典型的消息帧如附图 5.7 所示：

起始位	设备地址	功能代码	数据	CRC 校验	结束符
T1-T2-T3-T4	8bit	8bit	n 个 8bit	16bit	T1-T2-T3-T4

附图 5.7　　RTU 消息帧

ModBus RTU 的数据校验方式：**CRC-16（循环冗余错误校验）**

CRC-16 错误校验程序如下：报文（此处只涉及数据位，不指起始位、停止位和任选的奇偶校验位）被看作是一个连续的二进制，其最高有效位（MSB）首选发送。报文先与 X↑16 相乘（左移 16 位），然后看 X↑16+X↑15+X↑2+1 除，X↑16+X↑15+X↑2+1 可以表示为二进制数 11000000000000101。整数商位忽略不记，16 位余数加入该报文（MSB 先发送），成为 2 个 CRC 校验字节。余数中的 1 全部初始化，以免所有的零成为一条报文被接收。经上述处理而含有 CRC 字节的报文，若无错误，到接收设备后再被同一多项式（X↑16+X↑15+X↑2+1）除，会得到一个零余数（接收设备核验这个 CRC 字节，并将其与被传送的 CRC 比较）。全部运算以 2 为模（无进位）。

5．4．2　　MODBUS ASCII 协议

当控制器设为在 Modbus 网络上以 ASCII（美国标准信息交换代码）模式通信，在消息中的每个 8Bit 字节都作为两个 ASCII 字符发送。这种方式的主要优点是字符发送的时间间隔可达到 1 秒而不产生错误。

- **代码系统**

  十六进制，ASCII 字符 0...9，A...F

  消息中的每个 ASCII 字符都是一个十六进制字符组成

- **每个字节的位**

  1 个起始位

  7 个数据位，最小的有效位先发送

  1 个奇偶校验位，无校验则无

  1 个停止位（有校验时），2 个 Bit（无校验时）

- **错误检测域**

  LRC(纵向冗长检测)

MODBUS ASCII 协议消息帧，消息以冒号（:）字符（ASCII 码 3AH）开始，以回车换行符结束（ASCII 码 0DH,0AH）。其它域可以使用的传输字符是十六进制的 0...9,A...F。网络上的设备不断侦测“:”字符，当有一个冒号接收到时，每个设备都解码下个域（地址域）来判断是否发给自己的。

消息中字符间发送的时间间隔最长不能超过 1 秒，否则接收的设备将认为传输错误。一个典型消息帧如附图 5.8 所示：

起始位	设备地址	功能代码	数据	LRC 校验	结束符
1 个字符	2 个字符	2 个字符	n 个字符	2 个字符	2 个字符

附图 5.8　　ASCII 消息帧

ModBus ASCII 的数据校验方式：**LRC（纵向冗余错误校验）**

LRC 错误校验用于 ASCII 模式。这个错误校验是一个 8 位二进制数，可作为 2 个 ASCII 十六进制字节传送。把十六进制字符转换成二进制，加上无循环进位的二进制字符和二进制补码结果生成 LRC 错误校验（参见图）。这个 LRC 在接收设备进行核验，并与被传送的 LRC 进行比较，冒号（：）、回车符号（CR）、换行字符（LF）和置入的其他任何非 ASCII 十六进制字符在运算时忽略不计。