

S R 80 调节器通讯指南 ★ V 2 . 0      2001 年 6 月

本资料和开发的学习软件, 作为用户学习 SR80 系列仪表通讯编程的参考, 不足之处请给与指正。



说明:用户在购买 SR80 系列带通讯接口产品时,将随机提供该学习软盘。

\*\*\*\*\*

## ————— 目录 —————

1. 软件清单
2. CC80 V2.0 的使用方法
3. 进入通讯命令学习前的准备工作
4. 通讯协议以及 BASIC 程序方法
5. SR80 系列仪表的通讯命令
6. BASICA 的程序通讯软件说明:
7. 附录:A. 通讯串口接线方法  
    B. 有关 RS422/485 通讯口的技术数据

\*\*\*\*\*

### 1. 软件清单

在软盘内,提供了下述的软件和资料

CC80. BAS - 中文学习软件原程序

CC80. DOT - SR80 调节器通讯指南(WORDS 文件)

BASICA. EXE - 高级 BASIC 语言

SR80. BAS - 通讯测试程序

232T. BAS - BASIC 程序的 PC 机 232 口及先锋 RS422 口测试软件

用户可用中文 WINDOWS 的书写器检查或打印 CC80. DOC 文件内容。



## 2. CC80 V2.0 的使用方法



在 WINDOWS 操作系统下, 打开 WORDS 或文档, 调入 CC80. DOT 文件,

1) 串口接线

①计算机与带 RS-232C 通讯口的连线

②计算机与希曼顿 RS-232C/RS-485A 通讯变换器连线

③RS-232C/RS-485 通讯变换器与仪表 RS-485 通讯口的连线

④D 型 25 针、九针串口接线对照表

2) 通讯协议

3) 参数设置

设置调节器通讯地址和使用的 PC 机串口, 及通讯参数设置。

### 3. 进入通讯命令学习前的准备工作

#### 3-1. 初次连接系统的准备工作 (仪表未连接)

1.) 参照串口接线窗口和附录 A. 通讯串口接线方法, 对系统进行正确的接线。

2.) PC 机 RS232 通讯口正常(包括地线、握手信号), 将 SD, RD 端短接。

3.) RS232 接口至 RS232/RS485 转换器连线是否正确。

**注意:9 针与 25 针串口的定义区别。**



4.) 将 RS232/RS485 转换器的 RS422 输出端发送、接收短接, 测转换器好坏。

5.) RS232/RS485 转换器到仪表通讯口的连线正确。

如果远距离通讯(1200 米), 利用示波测量发送波形的前沿, 确定通讯线路的传输品质, 选择合适的通讯波特率。

**注: 如采用 PC 机内式转换卡, 可省去前两步骤。**

其它的操作:

连接仪表且上电, 确信仪表已进行了有关的通讯参数(地址、波特率)设定。

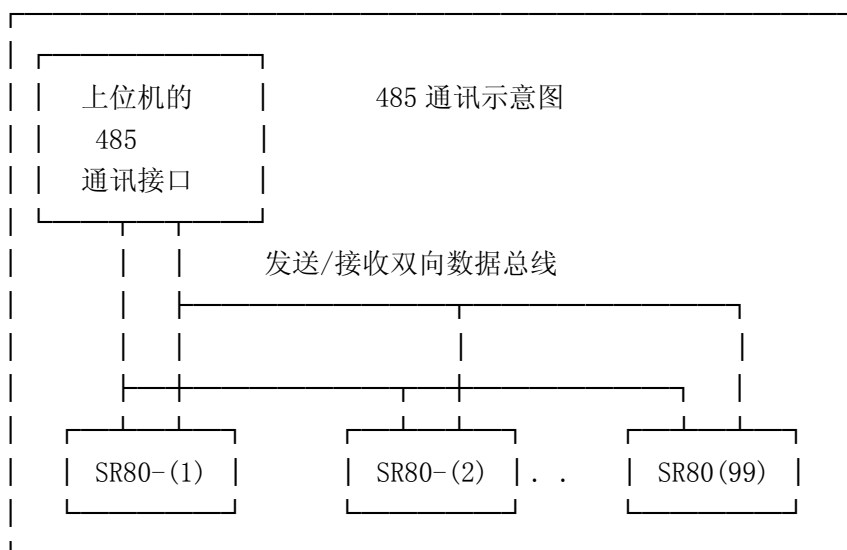
在学习软件中相应的画面应设置与仪表相一致的 PC 机通讯地址和字符参数, 否则将不能正常通讯。**注: 仪表的有关设定请参阅 SR80 操作流程**

## 4. 通讯协议以及 BASIC 程序方法

SR80 提供与岛电 SR253 兼容的通讯协议

### 4-1. 通讯的含意

RS232 接口, 只能单台点对点的通讯, 不能进行总线的并联, 但通讯软件和 485 方式相同



RS485 通讯采用差动的两线发送, 两线接收的双向数据总线两线制方式。上位机和下位调节器的内部接收器的接收高(RD+)和低(RS-)线以及内部发送器的发送高(SD+)和低(SD-)线都挂在数据总线上, 平时内部发送器的发送线处于高阻关闭态。如下图通讯过程示意图所示, 通常上位机是讲者, 下位调节器是听者, 并按主、从方式进行通讯, 多台仪表的通讯靠地址(设备号)的不同来区分。通讯中, 发送方需将发送线置于低阻态。发送完成后, 发送线需重新恢复到高阻关闭态。接收方在接收数据完成后, 又成为发送方。

因此, RS485 接口存在着双向数据总线转换冲突问题。在上位机可由软件调整, 下位可由仪表的 RS485 延时时间窗口调整。

建议: 选用 RS485 接口的仪表时, 可采用研华 5020 型 RS232/RS485 智能通讯转换模块, 编程时无需考虑总线切换的问题。

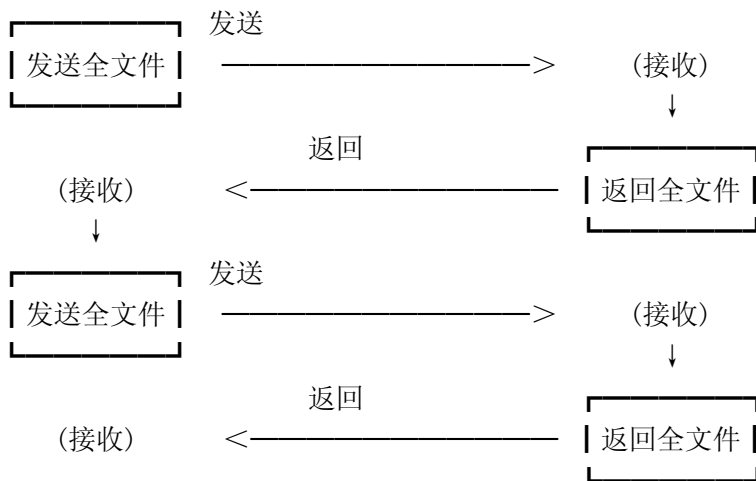
注意: 通讯时, 上位机必须根据调节器设定的地址, 共同约定的数据格式, 波特率等通讯规约, 发送通讯文件, 下位调节器在接收地址符合, 接收字符格式和校验正确后才能进行正常的通讯。

### 4-2. 通讯协议说明:

通讯协议的通信用示意图

上位机

调节器



### 4-3. 发送全文件和返回全文件的组成

4.3.1 通讯控制符的三种格式：1. STX\_ETX\_CR 2. STX\_ETX\_CRLF 3. @:\_:CR

#### 4.3.2 通讯发送格式

a	b		c	d	e				f	g	h	i		j	
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)
STX	0	1	1	R	0	1	0	0	0	----	ETX	D	A	CR	
STX	0	1	1	W	0	1	8	c	0	,****	ETX	7	8	CR	

#### 1. 通讯发送格式的解释

- 通讯的起始符，[(1)一位，STX: (02H) 或 "@"(40H)]
- 通讯下位机地址[(2)、(3)两位]，由 8 位二进制组成。地址范围 1~99(1:0000 0001~0110 0110)，这 8 位二进制码被分成高 4 位和低 4 位，其中高 4 位被送入(2)中，低 4 位被送入(3)，并转换成 ASCII 码。
- 通讯下位机地址的子地址[(4)一位]，这位被固定为 1。
- 通讯命令类型[(5)一位]。“R”(52H)，表明在上位机发送或仪表应答中的读命令。“W”(57H)，表明在上位机发送或仪表应答中的写命令。“B”表明在上位机以广播方式发送命令，但 SR80 不支持广播方式，“B”只作为预留命令。
- 通讯命令代码[(6)、(7)、(8)、(9)四位]。是 16 位二进制代码 (0~65535)，这 16 位被分成四组，并转换成相应的 ASCII 码。命令代码详见命令代码表。
- 通讯命令连续读代码[(10)一位]。表明上位机要连续读取多少个参数。这位取值范围“0”(30H) ~ “9”(39H)，十个数。实际的连续读参数的个数=表明的数值+1。读命令此位固定为“0”(30H)。
- 通讯数据[(11)这位的数据量决定于这位的数据，既这位的数据长度不定]。数据总是以“,”(2CH)，数据项与数据项之间不需要任何分割符。数据的长度主要取决于第(10)的方式。每一个数据项由 16 位二进制代码组成(1 个字)，每 4 位被分成一个数据单元，同时每个数据单元又被转换成 ASCII 数据。当(5)位为“R”读命令时，此位不写。

	第一数据项				第二数据项				.....	第 N 数据项			
	高位			低位	高位			低位		.....	高位		
“, ” 2CH	第一 单元	第二 单元	第三 单元	第四 单元	第一 单元	第二 单元	第三 单元	第四 单元	.....	第一 单元	第二 单元	第三 单元	第四 单元

- 数据发送结束符[(12)一位，ETX(03H) 或 “:”(3AH)]。所有的数据和命令再此位之前都以发送完成，遇到此字符表明结束。
- BCC 块校验 [(13)、(14)两位] 三种 BCC 块校验和无校验。上位机的 BCC 校验应通过软件处理。仪表的 BCC

校验可在[1-34]窗口设置。当 BCC 校验结果有错误时，将没有应答。BCC 校验数据被分成高 4 位和低 4 位，并被转换成 ASCII 码，

(13):高 4 位的 ASCII 码。(14):低 4 位的 ASCII 码。

1). ADD 块校验

例: STX 0 1 1 R 0 1 0 0 9 EXT E 3 CR LF  
 (02H)+(30H)+(31H)+(31H)+(52H)+(30H)+(31H)+(30H)+(30H)+(39H)+(03H)=1E3H  
 BCC 校验结果 (13): "E"=45H (14): "3"=33H

2). ADD\_TWO'S CMP 块校验

例: STX 0 1 1 R 0 1 0 0 9 EXT 1 D CR LF  
 (02H)+(30H)+(31H)+(31H)+(52H)+(30H)+(31H)+(30H)+(30H)+(39H)+(03H)=1E3H  
 BCC 校验结果 (13): "1"=31H (14): "D"=44H

3). XOR 块校验

例: STX 0 1 1 R 0 1 0 0 9 EXT 1 D CR LF  
 (02H) (30H) (31H) (31H) (52H) (30H) (31H) (30H) (30H) (39H) (03H)=59H  
 BCC 校验结果 (13): "5"=35H (14): "9"=39H

j 回车符[(15)、(16)一位或两位 CR (0DH) 或 CRLF] 全文结束符既回车符。CR 或 CR LF 可以选择。

4). None 无校验

4.3.3 通讯应答格式

a	b		c	d	e		g	h	i		j	
(1)	(2)	(3)	(4)	(5)	(6)	(7)	(11)	(12)	(13)	(14)	(15)	(16)
STX	0	1	1	R	0	1	-----	ETX	3	C	CR	
STX	0	1	1	W	0	1	,****	ETX	4	E	CR	

1. 通讯应答格式的解释

- a 通讯的起始符，[(1)一位，STX: (02H) 或 "@" (40H)]
- b 通讯下位机地址[(2)、(3)两位]，由 8 位二进制组成。地址范围 1~99(1:0000 0001~0110 0110)，这 8 位二进制码被分成高 4 位和低 4 位，其中高 4 位被送入(2)中，低 4 位被送入(3)，并转换成 ASCII 码。
- c 通讯下位机地址的子地址[(4)一位]，这位被固定为 1。
- d 通讯命令类型[(5)一位]。"R" (52H), 表明在上位机发送或仪表应答中的读命令。"W" (57H), , 表明在上位机发送或仪表应答中的写命令。"B" 表明在上位机以广播方式发送命令, 但 SR80 不支持广播方式, "B" 只作为预留命令。
- e 应答代码[(6)、(7)两位]。是 8 位二进制代码 (0~255)，这 8 位被分成高 4 位和低 4 位，并转换成相应的 ASCII 码。应答代码详见应答代码表。(6):高 4 位的 ASCII 码。(7):低 4 位的 ASCII 码。
- g 通讯数据[(11)这位的数据量决定于这位的数据，既这位的数据长度不定]。数据总是以", " (2CH), 数据项与数据项之间不需要任何分割符。数据的长度主要取决于第(10)的方式。每一个数据项由 16 位二进制代码组成(1 个字)，每 4 位被分成一个数据单元，同时每个数据单元又被转换成 ASCII 数据。当(5)位为 "W" 写命令时，此位不用。

	第一数据项				第二数据项				.....	第 N 数据项			
	高位			低位	高位			低位	.....	高位			低位
" , "	第一	第二	第三	第四	第一	第二	第三	第四	.....	第一	第二	第三	第四
2CH	单元	单元	单元	单元	单元	单元	单元	单元	.....	单元	单元	单元	单元

h 数据发送结束符[(12)一位，ETX(03H) 或 ":" (3AH)]。所有的数据和命令再此位之前都以发送完成，遇到

此字符表明结束。

i BCC 块校验 [(13)、(14)两位] 三种 BCC 块校验和无校验。上位机的 BCC 校验应通过软件处理。仪表的 BCC 校验可在[1-34]窗口设置。当 BCC 校验结果有错误时，将没有应答。BCC 校验数据被分成高 4 位和低 4 位，并被转换成 ASCII 码，

(13):高 4 位的 ASCII 码。(14):低 4 位的 ASCII 码。

1). ADD 块校验

例: STX 0 1 1 R 0 1 0 0 9 EXT E 3 CR LF  
 (02H)+(30H)+(31H)+(31H)+(52H)+(30H)+(31H)+(30H)+(30H)+(39H)+(03H)=1E3H  
 BCC 校验结果 (13): "E"=45H (14): "3"=33H

2). ADD\_TWO'S\_CMP 块校验

例: STX 0 1 1 R 0 1 0 0 9 EXT 1 D CR LF  
 (02H)+(30H)+(31H)+(31H)+(52H)+(30H)+(31H)+(30H)+(30H)+(39H)+(03H)=1E3H  
 BCC 校验结果 (13): "1"=31H (14): "D"=44H

3). XOR 块校验

例: STX 0 1 1 R 0 1 0 0 9 EXT 1 D CR LF  
 (02H) (30H) (31H) (31H) (52H) (30H) (31H) (30H) (30H) (39H) (03H)=59H  
 BCC 校验结果 (13): "5"=35H (14): "9"=39H

j 回车符[(15)、(16)一位或两位 CR (0DH) 或 CRLF] 全文结束符既回车符。CR 或 CR LF 可以选择。

4). None 无校验

#### 4.3.4 读命令、写命令及应答举例

##### 1. 读命令

d	e				f
(5)	(6)	(7)	(8)	(9)	(10)
R	0	4	0	0	9
52H	30H	34H	30H	30H	39H

d: 这位表明这是一个读命令。

e: 这位表明这个读命令是读 SV1 的控制输出 1 的比例带的参数。

f: 这位表明这读命令要读多少个参数。

具体含义如下:

这位表明这个读命令是读 SV1 的控制输出 1 的比例带 =0400H (十六进制)  
 =0000 0100 0000 0000 (二进制)

这位表明这读命令要读多少个参数。  
 =9H  
 =1001 (二进制)  
 =9 (十进制)

(实际读取参数的个数) =10 (9+1)

##### 2. 正确的应答

D	e		g													
(5)	(6)	(7)	第一数据项				(11)	第二数据项			第五数据项					
R	0	0	,	0	0	6	4	0	0	6	E	.....	0	0	0	0
52H	30H	30H	2CH	30H	30H	36H	34H	30H	30H	36H	45H	.....	30H	30H	30H	30H

- d: 这位表明这是一个应答的读命令。
- e: 这位表明这是一个应答代码<0(30H) 0 (30H) 正确的应答>。(详见应答代码表)
- g: 这位表明这应答的读命令返回的数据项。返回数据项个数与上位机的 f(0)位有关。

### 3. 不正确的读命令应答

D	e	
(5)	(6)	(7)
R	0	7
52H	30H	37H

- d: 这位表明这是一个应答的读命令。
- e: 表明这是一个应答代码<0(30H) 7 (30H) 是数据格式错误的应答>。(详见应答代码表)

### 4. 写命令

d	e				f	g				
(5)	(6)	(7)	(8)	(9)	(10)	(11)				
W	0	4	0	0	0	,	0	0	2	8
57H	30H	34H	30H	30H	30H	2CH	30H	30H	32H	38H

- d: 这位表明这是一个写命令。
- e: 这位表明这个读命令是写 SV1 的控制输出 1 的比例带的参数。
- f: 这位表明这读命令要读多少个参数。
- g 通讯数据 [(11)这位的数据量决定于这位的数据，既这位的数据长度不定]。数据总是以“,” (2CH), 数据项与数据项之间不需要任何分割符。数据的长度主要取决于第(10)的方式。每一个数据项由 16 位二进制代码组成(1 个字)，每 4 位被分成一个数据单元，同时每个数据单元又被转换成 ASCII 数据。

具体含义如下：

这位表明这个写命令是写 SV1 的控制输出 1 的比例带参数	=0400H (十六进制)
	=0000 0100 0000 0000 (二进制)
这位表明这读命令要写多少个参数。	=0H
	=0000 (二进制)
	=0 (十进制)
(实际写参数的个数)	=1 (0+1)
被写入的具体数据	=0400H (十六进制)
	=0000 0000 0010 1000 (二进制)
	=40 (十进制)

### 5. 正确的写命令应答

d	e	
(5)	(6)	(7)
W	0	0
57H	30H	30H

- d: 这位表明这是一个写应答的命令。
- e: 表明这是一个应答代码<0(30H) 0 (30H) 是一个写命令的正确应答>。(详见应答代码表)

### 6. 不正确的写命令应答举例

d	e	
(5)	(6)	(7)

W	0	9
57H	30H	30H

d: 这位表明这是一个写应答的命令。

e: 表明这是一个应答代码<0(30H)9(39H)是一个不正确写命令的应答>。(详见应答代码表)

#### 4-4 应答代码表

应答代码		代码类型	代码类型的详细说明
二进制码	ASCII		
0000 0000	"0", "0":30H, 30H	正确的应答	读、写命令的正确应答
0000 0001	"0", "1":30H, 31H	硬件错误	当发生硬件错误例如帧溢出或奇偶校验错误被检测到时。
0000 0111	"0", "7":30H, 37H	数据格式错误	数据格式部分和设计的固定格式不符。
0000 1000	"0", "8":30H, 38H	命令或数据的数量错误	命令代码或数据的数量和设计的要求不同。
0000 1001	"0", "9":30H, 39H	数据错误	被写入的数据不是有效的可被设定的范围
0000 1010	"0", "A":30H, 41H	执行命令错误	执行命令的接收是在一定条件下的(例如 AT), 否则将不被执行
0000 1011	"0", "B":30H, 42H	写模式错误	一些类型的数据在某一时刻将不能被写入。这种数据写入应在这种数据允许写入的时刻写入。
0000 1100	"0", "C":30H, 43H	其他或操作错误	写命令中的特殊数据或操作, 不能被加入或接收。

小数点的表示方法: 将小数点去掉后, 直接连同小数点后的数转换成十六进制数。小数点的位置与使用的量程有关。这四位十六进制代码(16位二进制码)的使用范围(-32768~32767)。

例: 十六进制十六进制

20.0%            200            00C8

100.0%          1000          03E8

#### 4-5 通讯命令表

命令代码 (十六进制)	参数	参数的详细说明	读/写
0030		公司名称代码 1 (此命令为保留地址)	读
0031		公司名称代码 2 (此命令为保留地址)	读
0032		公司名称代码 3 (此命令为保留地址)	读
0033		公司名称代码 4 (此命令为保留地址)	读
0034		公司名称代码 5 (此命令为保留地址)	读
0035		公司名称代码 6 (此命令为保留地址)	读
0036		公司名称代码 7 (此命令为保留地址)	读
0037		公司名称代码 9 (此命令为保留地址)	读

这些命令由 16 位二进制组成，被分成高 8 位和低 8 位两个单元。不用的地址用“0”填充。

例：SHIMADEN	命令	高 4 位	低 4 位	命令	高 4 位	低 4 位
	0030	"S"	"H"	0034	00H	00H
	0031	"I"	"M"	0035	00H	00H
	0032	"A"	"D"	0036	00H	00H
	0033	"E"	"N"	0037	00H	00H

命令代码 (十六进制)	参数	参数的详细说明	读/写
0040		产品代码 1 (此命令为保留地址)	读
0041		产品代码 2 (此命令为保留地址)	读
0042		产品代码 3 (此命令为保留地址)	读
0043		产品代码 4 (此命令为保留地址)	读

这些命令由 16 位二进制组成，被分成高 8 位和低 8 位两个单元。不用的地址用“0”填充。

例：SR253	命令	高 4 位	低 4 位	高 4 位	低 4 位
	0040	"S"	"R"	53H	52H
	0041	"2"	"5"	32H	35H
	0042	"3"	"0"	33H	30H
	0043	"0"	"0"	30H	30H

命令代码 (十六进制)	参数	参数的详细说明	读/写
0100	PV_W	测量值	读
0101	SV_W	当前执行的设定值	读
0102	OUT1W	控制输出 1 的值	读
0103	OUT2W	控制输出 2 的值 (无输出时=0000H)	读
0104	EXE_FLG	执行标志 (不执行时=0)	读
0105	EV_FLG	事件输出标志 (无事件输出时=0000)	读
0106	SV No.	当前执行的 SV 号 0=SV1, 1=SV2, SB, REM	读
0107	EXE_PID	当前执行的 PID 号 0=PID1, 1=PID2	读
0108	REM_W	模拟遥控输入值 (无输入时=0000H)	读
0109	HB_W	加热器断线报警值 (无报警时=0000H)	读
010A	HL_W	欠流报警值 (无报警时=0000H)	读
010B	DI_FIG	外部 DI 开关输入状态标志 (无输入时=0000H)	读

命令代码 (十六进制)	参数	参数的详细说明	读/写
0111	RANGE	测量范围 (见测量范围代码表)	读
0112	CJ	冷端补偿 0=内部, 1=外部	读
0113	DP	小数点位置 0=无, 1=XXX. X, 2=XX. XX, 3=X. XXX	读
0114	SC_L	测量范围下限值	读
0115	SC_H	测量范围上限值	读

命令代码 (十六进制)	参数	参数的详细说明	读/写
0180	SV_No	设定执行的 SV 号	写
0181	SV_QNo	设定执行的 SV 号 (无斜率)	写
0182	OUT1W	在手动方式下设置输出 1 的值	写
0183	OUT2W	在手动方式下设置输出 2 的值	写
0184	AT	自整定 0=不执行, 1=执行	写
0185	MAN	手动 0=自动, 1=手动	写
0186	STBY	脱机 0=执行, 1=脱机	写
0187	REM	遥控 0=设定值, 1=遥控值	写
0188	SB	偏移 0=关, 1=开	写
0189	Reserved	保留不用	写
018A	Reserved	保留不用	写
018B	STOP	停止 0=运行, 1=停止	写
018B	COM	通讯 0=本机, 1=通讯	写

0300	SV1	设定 SV1 的值	读/写
0301	SV2	设定 SV2 的值	读/写

030A	SV_L	SV 下限值	读/写
030B	SV_H	SV 上限值	读/写
030C	RAMO_UP	斜率上升	读/写
030D	RAMP_DW	斜率下降	读/写
030E	RAMP_UNT	斜率单位 0=秒 1=分钟	读/写
030F	RAMP_RTE	斜率倍率 0= $\times 1$ 0= $\times 0.1$	读/写

0311	SB	设置偏移值	读/写
0312	SV_MD	SB/SV 的模式 0=无, 1=SV, 2=SB	读/写
0313	Reserved	保留不用	读/写
0314	REM_L	模拟遥控输入的 量程下限	读/写
0315	REM_H	模拟遥控输入的 量程上限	读/写
0316	REM_B	模拟遥控输入的 偏移值	读/写
0317	REM_F	模拟遥控输入的 滤波时间系数	读/写
0318	REM_T	模拟遥控输入跟踪 0=不跟踪, 1=跟踪	读/写

031D	REM_P	模拟遥控输入的 切换点值	读/写
031E	REM_D	模拟遥控输入的 切换点回差	读/写

0400	PB1	SV1 的控制输出 1 的比例带	读/写
0401	IT1	SV1 的控制输出 1 的积分时间	读/写
0402	DT1	SV1 的控制输出 1 的微分时间	读/写
0403	MR1	SV1 的人工补偿	读/写
0404	DF1	SV1 的回差	读/写
0405	011_L	SV1 的控制输出 1 下限	读/写

0406	011_H	SV1 的控制输出 1 上限	读/写
0407	SF1	SV1 的控制输出 1 抗超调系数	读/写
0408	PB2	SV2/SB, 遥控输入的控制输出 1 的比例带	读/写
0409	IT2	SV2/SB, 遥控输入的控制输出 1 的积分时间	读/写
040A	DT2	SV2/SB, 遥控输入的控制输出 1 的微分时间	读/写
040B	MR2	SV2/SB, 遥控输入的人工补偿	读/写
040C	DF2	SV2/SB, 遥控输入的回差	读/写
040D	012_L	SV2/SB, 遥控输入的控制输出 1 下限	读/写
040E	012_H	SV2/SB, 遥控输入的控制输出 1 上限	读/写
040F	SF2	SV2/SB, 遥控输入的控制输出 1 的抗超调系数	读/写

0460	PB21	SV1 的控制输出 2 的比例带	读/写
0461	IT21	SV1 的控制输出 2 的积分时间	读/写
0462	DT21	SV1 的控制输出 2 的微分时间	读/写
0463	DB21	SV1 的死区	读/写
0464	DF21	SV1 的回差	读/写
0465	021_L	SV1 的控制输出 2 下限	读/写
0466	021_H	SV1 的控制输出 2 上限	读/写
0467	SF21	SV1 的控制输出 2 的抗超调系数	读/写
0468	PB22	SV2/SB, 遥控输入的控制输出 2 的比例带	读/写
0469	IT22	SV2/SB, 遥控输入的控制输出 2 的积分时间	读/写
046A	DT22	SV2/SB, 遥控输入的控制输出 2 的微分时间	读/写
046B	DB22	SV2/SB, 遥控输入的死区	读/写
046C	DF22	SV2/SB, 遥控输入的回差	读/写
046D	022_L	SV2/SB, 遥控输入的控制输出 2 下限	读/写
046E	022_H	SV2/SB, 遥控输入的控制输出 2 上限	读/写
046F	SF22	SV2/SB, 遥控输入的控制输出 2 的抗超调系数	读/写

0500	EV1_MD	事件报警 1 的模式 (见说明书事件报警)	读/写
0501	EV1_SP	事件报警 1 的设定值 (见说明书事件报警)	读/写
0502	EV1_DF	事件报警 1 的回差	读/写
0503	EV1_STB	事件报警 1 的抑制和非抑制状态 OFF: 无抑制。 1: 初次上电, 报警抑制。 2: 初次上电脱机状态时, 报警抑制。 3: 初次上电脱机状态或改变设定值时, 报警抑制。 4: 脱机状态时抑制, 运行状态时无抑制。	读/写
0504	EV1_TM	事件报警 1 的延迟时间	读/写
0505	Reserved	保留不用	读/写
0506	Reserved	保留不用	读/写
0507	Reserved	保留不用	读/写
0508	EV2_MD	事件报警 2 的模式 (见说明书事件报警)	读/写
0509	EV2_SP	事件报警 2 的设定值 (见说明书事件报警)	读/写
050A	EV2_DF	事件报警 2 的回差	读/写

050B	EV2_STB	事件报警 2 的抑制和非抑制状态 OFF:无抑制。 1: 初次上电, 报警抑制。 2: 初次上电脱机状态时, 报警抑制。 3: 初次上电脱机状态或改变设定值时, 报警抑制。 4: 脱机状态时抑制, 运行状态时无抑制。	读/写
050C	EV2_TM	事件报警 2 的延迟时间	读/写
050D	Reserved	保留不用	读/写
050E	Reserved	保留不用	读/写
050F	Reserved	保留不用	读/写
0510	EV3_MD	事件报警 3 的模式 (见说明书事件报警)	读/写
0511	EV3_SP	事件报警 3 的设定值 (见说明书事件报警)	读/写
0512	EV3_DF	事件报警 3 的回差	读/写
0513	EV3_STB	事件报警 3 的抑制和非抑制状态 OFF:无抑制。 1: 初次上电, 报警抑制。 2: 初次上电脱机状态时, 报警抑制。 3: 初次上电脱机状态或改变设定值时, 报警抑制。 4: 脱机状态时抑制, 运行状态时无抑制。	读/写
0514	EV3_TM	事件报警 3 的延迟时间	读/写

0580	DI1	外部开关 DI1 软指定的功能 0=无, 1=脱机, 2=SV 或 SB 的转换, 3=自整定, 4=手动, 5=正反作用, 6=斜率运行的保持/继续, 7=外给定 RSV/本机 SV 设定	读/写
0581	DI2		读/写

0590	HBS	加热器断线报警设置	读/写
0591	HBL	加热器环路报警设置	读/写
0592	HBM	加热器断线报警锁定模式设置 0=软锁定, 1=硬锁定	读/写

05A0	A01_MD	模拟变送模式 0=测量值, 1=设定值, 2=偏差值, 3=控制输出 1 的值, 3=控制输出 1 的值	读/写
05A1	A01_L	模拟变送下限	读/写
05A2	A01_H	模拟变送上限	读/写

05B0	COM_MEM	通讯的存贮模式 0=EEP, 1=REM, 2=r_E	读/写
------	---------	-----------------------------	-----

0600	ACTMD	输出的特性 0=反作用, 1=整作用	读/写
0601	01_CYC	SV1 的比例周期	读/写
0602	ERRROUT1	SV1 的错误输出	读/写
0603	Reserved	保留不用	读/写
0604	02_CYC	SV2 的比例周期	读/写
0605	ERRROUT2	SV2 的错误输出	读/写

0610	ATP	自整定点	读/写
0611	KLOCK	键盘锁 0=不锁定, 1=除了 SV、AT、MAN 以外锁定 2=除了 SV 以外锁定, 3=全部锁定	读/写

0701	PV_B	PV 值偏移	读/写
0702	PV_F	PV 值滤波	读/写

#### 4-6 ASCII 代码

	b7b6b5	000	001	010	011	100	101	110	111
b4b3b1		0	1	2	3	4	5	6	7
0000	0	NUL	TC7 (DLE)	SP	0	@	P	`	p
0001	1	TC1 (SOH)	DC1	!		A	Q	a	q
0010	2	TC2 (STX)	DC2	”		B	R	b	r
0011	3	TC3 (ETX)	DC3	#		C	S	c	s
0100	4	TC4 (EOT)	DC4	\$		D	T	d	t
0101	5	TC5 (ENQ)	TC8 (NAK)	%		E	U	e	u
0110	6	TC6 (ACK)	TC9 (SYN)	&		F	V	f	v
0111	7	BEL	TC10 (ETB)	'		G	W	g	w
1000	8	FE0 (BS)	CAN	(		H	X	h	x
1001	9	FE1 (HT)	EM	)		I	Y	i	y
1010	A	FE2 (LF)	SUB	*		J	Z	j	z
1011	B	FE3 (VT)	ESC	+		K	[	k	{
1100	C	FE4 (FF)	IS4 (FS)	,		L	\	l	
1101	D	FE5 (CR)	IS3 (GS)	-		M	]	m	}
1110	E	SO	IS2 (RS)	.		N	^	n	~
1111	F	SI	IS1 (US)	/	?	O	_	o	DEL

#### 4-7 BASICA 程序例

4.7.1 设置起始符, 文件结束, 全文件结束的三个控制符

```
10 STX$ = "@": ETX$ = ":": CR$=CHR$(13)
```

```
15 REM 初使化 PC 机口和设数据格式(必需和仪表的设置相同)
```

```
20 REM 使用 PC COM1 口, 设置 1200 波特, 偶效验, 7 位数据, 1 停止位, 屏蔽握手信号。
```

```
40 BPS$ = "1200": ADR$="01": REM 设置波特率和仪表通讯地址
```

```
50 OPEN "COM1:" + BPS$ + ",E,7,1,CD,RS,CS,DS" AS #1
```

```
70 CMD$="0100": REM READ PV
```

```
80 BC$ = ADR$ + CMD$ + ETX$
```

```
90 GOSUB 420
```

```
100 PRINT BCC$
```

```
110 STOP
```

```
420 BCC = ASC(LEFT$(BC$, 1)): REM 发/接的 BCC 块效验程序
```

```
430 L = LEN(BC$)
```

```
440 FOR N = 2 TO L
```

```
450 BCC = BCC XOR ASC(MID$(BC$, N, 1))
```

```
460 NEXT N
```

```
470 BCC$ = HEX$(BCC)
```

```
480 IF LEN(BCC$) = 1 THEN BCC$ = "0" + BCC$: REM 如效验结果为单字节, 需加"0"
```

4.7.1 仪表的通讯设置

1-29 窗口

通讯/机内方式选择	
C O M M	C O M M
L o c	L O C A L

LOC: 机内方式

◎此时, 面板通讯 COM 指示灯灭。

◎仅能由上位机控制命令, 转成通讯方式 (COM)。

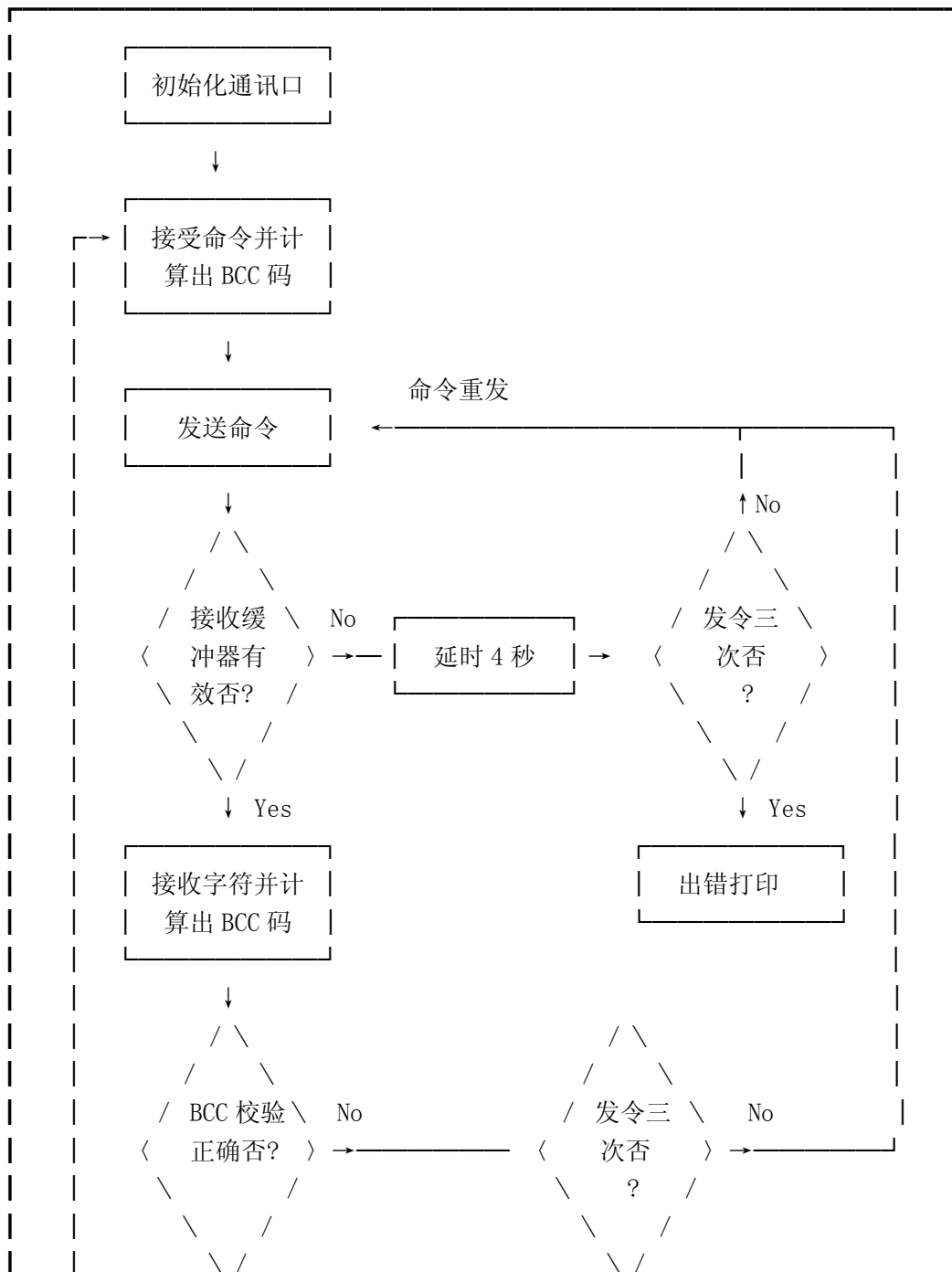
仅能完成上位机的读命令. 可由键设定内部参数。

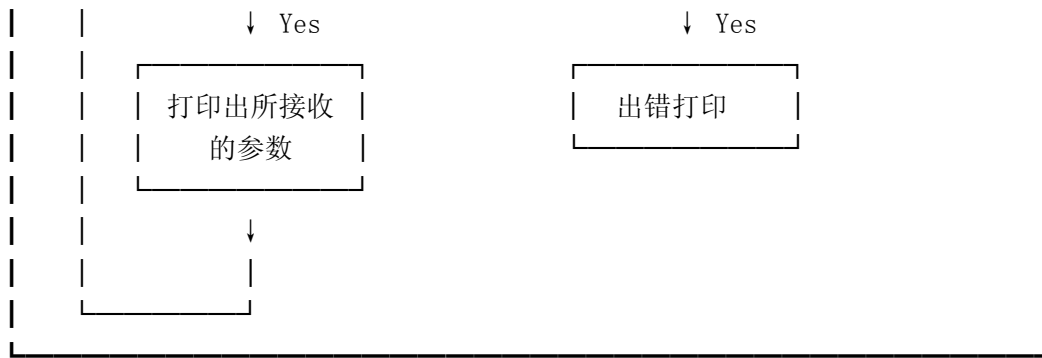
COM: 上位机通讯方式

◎此时, 面板通讯 COM 指示灯亮

◎仅能由面板键设定或上位机控制命令, 转成 LOC 机内方式。

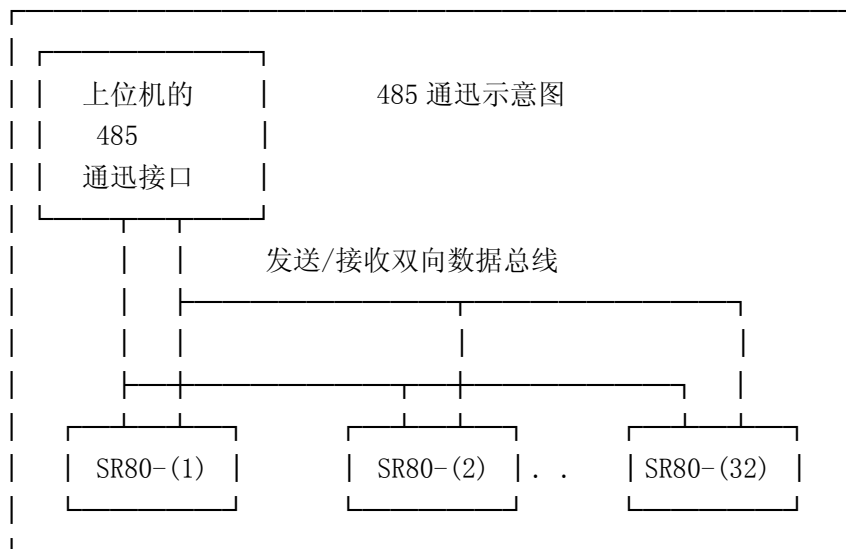
4.7.2 流程图





#### 4.7.3 RS485 通讯接口和 BASIC 程序方法

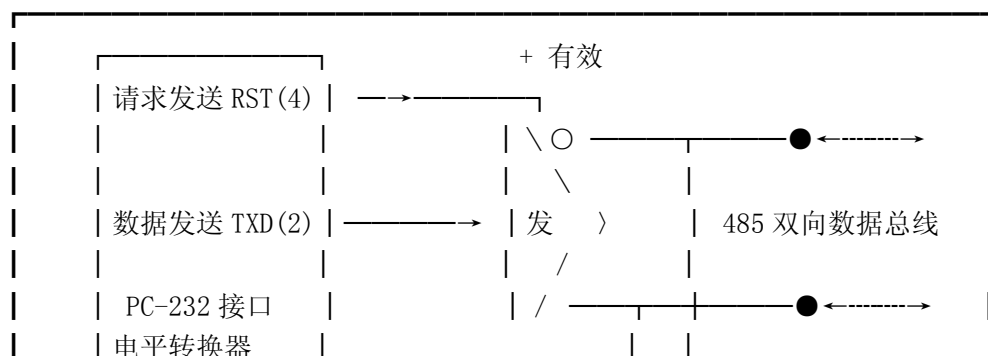
##### 1. RS485 通讯示意图



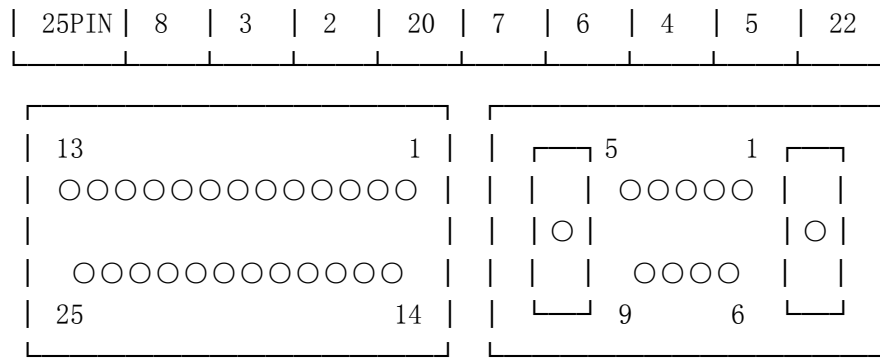
RS485 通讯采用差动的两线发送, 两线接收的双向数据总线两线制方式。上位机和下位仪表的内部接收器的接收高 (RDA) 和低 (RSD) 线以及内部发送器的发送高 (SDA) 和低 (SDB) 线都挂在数据总线上, 平时内部发送器的发送线处于高阻关闭态。如下图通讯过程示意图所示, 通常上位机是讲者, 下位调节器是听者, 并按主、从方式进行通讯, 多台仪表的通讯靠地址 (设备号) 的不同来区分。通讯中, 发送方需将发送线置于低阻态。发送完成后, 发送线需重新恢复到高阻关闭态。接收方在接收数据完成后, 又成为发送方, 以应答方式进行通讯。

通讯时, 上位机必须根据调节器设定的地址, 共同约定的数据格式, 波特率等通讯规约, 发送通讯文件, 下位调节器在接收地址符合, 接收字符格式和校验正确后, 才能进行正常的通讯。

##### 2. RS485 双向数据总线转换硬件示意图







25 针连接器接线图

九针准连接器接线图

B. RS232 通讯口的技术数据

- 1.信号电平: EIA RS-232C 电平(±12V)
- 2.通讯方式: RS232C 3 线半双工
- 3.同步系统: 起始位-停止位, 异步通讯
- 4.通讯距离: RS232C 15 米
- 5.通讯速度: 1200, 2400, 4800, 9600, 19200 波特率
- 6.数据格式: 8 种.  
常用格式: 数据 7 位, 一个偶校验位, 一个停止位
- 7.数据块校验: 数据异或(双字节)
- 8.通讯码: ASCII
- 9.握手信号: 未使用
- 10.连接台数: RS-232C 1 台

C. RS422/RS485 通讯接口的技术数据

- 1.信号电平: EIA RS422A/485 电平 5V 差动
- 2.通讯方式: RS422A 4 线半双工(多路)/RS485 2 线半双工(多路)
- 3.同步系统: 起始-停止位同位, 异步通讯
- 4.通讯距离: 1200 米
- 5.通讯速度: 1200, 2400, 4800, 9600, 19200 波特率
- 6.数据格式: 8 种.  
常用格式: 数据 7 位, 一个偶校验位, 一个停止位
- 7.数据块校验: 异或(双字节)
- 8.通讯码: ASCII
- 9.握手信号: 未使用
- 10.连接台数: RS-422/485 32 台 1.5 公里(配先锋 RS232/422 接口转换器)

## 6. C 语言程序例(仅供参考!)

```
#include<stdio.h>
#include<stdlib.h>
#include<conio.h>
#include<bios.h>
#include<string.h>
#define CH1_PA 0x3f8
void transmitter();
int receiver();
void main()
```

```

{
    int length, i, h, j, m, number, flag, k;
    char receive[100], comm[3], ccc[4];
    char ch;
    char send[]="D1"; /*发送命令*/
    printf("please input number:");
    scanf("%d", &k);
    outportb(CH1_PA+4, 0); /*disable interrupt, RTS, DTR*/
    outportb(CH1_PA+1, 0); /*disable interrupt enable register*/
    bioscom(0, 0xfa, 1); /*9600, 7, e, 1*/
    clrscr();
    gotoxy(26, 1);
    printf("SHIMADEN CONTROLLER");
    gotoxy(10, 2); printf("No");
    gotoxy(16, 2); printf("SV");
    gotoxy(22, 2); printf("PV");
    gotoxy(28, 2); printf("OP");
    gotoxy(35, 2); printf("ALM1");
    gotoxy(43, 2); printf("ALM2");
    gotoxy(50, 2); printf("LBA");
    gotoxy(58, 2); printf("ST");
    gotoxy(64, 2); printf("A/M");
    while(!kbhit())
    {
        for(number=0; number<k; number++) {
            h=number+1;
            gotoxy(10, 3+number); printf("%d", h);
            itoa(h, comm, 10);
            if(h<10) {send[1]='0'; send[2]=comm[0];}
            else {send[1]=comm[0]; send[2]=comm[1];}
            length=strlen(send);
            for(m=0; m<3; m++) /* Retry */
            {
                transmitter(send, CH1_PA, length);
                i=receiver(receive, CH1_PA);
                if(i!=0) break;
            }
            if(i==0)
            { printf("Timeout\n");}
            else{ gotoxy(15, 3+number);
                for(j=5; j<9; j++)
                    printf("%c", receive[j]);
                gotoxy(21, 3+number);
                for(j=10; j<14; j++)
                    printf("%c", receive[j]);
                gotoxy(27, 3+number);
                for(j=15; j<19; j++)

```



```
{j++;if(j>20000) return(0);}
rec[i]=inportb(BASE_ADD);/*receive data*/
i++;
}while(rec[i-1]!=0x2a);
return(i);
}
```