

50系列通讯接口操作说明 V2.0 1993年12月
日本岛电公司50系列PID过程控制器通讯接口中文学习软件)

本资料 and 开发的学习软件, 作为用户学习50系列通讯接口的参考, 不足之处请给与指正。

日本岛电公司中国技术服务中心 北京市希曼顿自动化研究所

说明: CC5020是先锋开发的日本岛电公司50系列PID过程控制器和SD20型智能过程监视器的中文通讯接口学习软件

用户在购SR50系列和SD20带通讯接口产品时, 将随机提供(仅收工本)该学习软盘。

CC5020的通讯学习软件操作说明书目录

1. 软件清单
2. CC5020 V2.1的使用方法
3. 两种通讯规约以及BASIC程序方法
4. 50系列的通讯命令
5. 在PC计算机上, 采用BASIC语言, 实现数据采集的编程例
6. 附录: A. 通讯串口接线方法
B. RS232通讯口的技术数据
C. RS422/485通讯口的技术数据

1. 软件清单

在CC5020软盘内, 提供了下述的软件和资料

- CC5020.EXE - 新通讯协议中文学习软件
- CC50SRFP.EXE - 与SR25, FP21通讯协议兼容的中文学习软件
- SR50COM.wps - 50通讯学习软件操作说明书(WPS文件)
- SD20COM.wps - 20通讯学习软件操作说明书(WPS文件)
- BASIC.EXE - 高级BASIC语言
- 50SRFP.BAS - BASIC程序的50的"D1"命令数据采集软件(FP21, SR25通用的旧通讯协议学习软件)
- SR50.BAS - BASIC程序的50的读写命令软件(新通讯协议)
- 232T.BAS - BASIC程序的PC机232口及先锋RS422口测试软件
- 485T.BAS - BASIC程序的先锋的RS485口测试软件

用户可用WPS或中文WORD STAR的"N"命令检查或打印SR50COM.wps文本内容。

建议: 在WPS中, 用"X"命令退出后, 设置SPDOS/600, 可增加屏幕显示宽度。

2. CC5020 V2.1的使用方法

2-1. CC5020 V2.1 的工作环境

- IBM PC/XT, AT, 286, 386或兼容机
- EGA(640×350)或EGA/VGA彩色显示器
- 640KB 以上的存储器
- 1.2MB 5.25" 软盘驱动器
- 至少5MB硬盘空间
- 一个RS232串口
- MS-DOS 3.0以上版本的磁盘操作系统
- UCDOS V2.0 汉字操作系统

2-2. CC5020A V2.00的安装

2-2-1. UCDOS 2.0 的安装 (已配置UCDOS的用户跳过此项)

将UCDOS的#2盘装入A驱动器, 键入INSTALL后回车。按屏幕上的提示依次将#3、#4盘装入。

2-2-2. UCDOS 2.0系统的配置

键入命令: C:/S 后, 进入UCDOS系统配置的设置。

建议如下: 字库设置: 64K 用于放入基本内存

显示设置: EGA 方式

打印模式: 西文方式

系统减: 键盘 ON; 联想 ON; 打印 ON

注: 更详细的说明见UCDOS操作说明书。

2-2-3. CC5020的装入

键入 MD回车后, 建立CC5020的子目录。

键入CD后回车, 进入子目录。

将盘插入A驱动器后, 键入 COPY A: *.* 后回车。至此系统安装完成。2-2-4. CC5020 V2.1软件的运行

在UCDOS操作系统下, 键入CD后, 在 C 盘提示符下键入:

C>CC5020 即可启动学习软件, 进入主菜单。

CC5020的操作简单, 均可按屏幕提示进行。

在主菜单时的操作:

按左、右方向键(), 可水平移动主菜单, 当光标指向某一功能项时, 按回车键即可进入主菜单内的子菜单。

在子菜单时的操作:

按上、下方向键(), 可上、下移动菜单, 当光标指向某一子菜单命令项时, 按回车键即可选择此命令项功能。

选择子菜单内命令项时的操作:

CC5020软件已被设计成自动向用户提示命令参数的选择。按左、右方向键(), 可水平移动子菜单命令参数窗口, 当光标指向某一参数时, 按翻页上键、下键 (PgUp, PgDn) 或删除键 (Del) 修改参数。按回车键即可执行该命令。按Esc键后, 将返回子菜单。

其它的画面显示

CC5020软件已被设计成自动向用户显示发接的全文件通讯格式、分项数据参数 格式和定义、通讯的出错信息等较为完善的学习功能。

主菜单和子菜单

进入主菜单后, 将显示七个子菜单, 主要功能:

1) 串口接线

计算机与带RS-232C通讯口的连线

计算机与先锋RS-232C/RS-422A通讯变换器连线

先锋RS-232C/RS-422A通讯变换器与带RS-422A通讯口的连线

D型25针、九针串口接线对照表

2) 通讯协议

3) 参数设置

CC50A对调节器通讯地址和PC机串口设置和选择, PC机的通讯参数设置。

4) SR50通讯学习窗口

5) SD20通讯学习窗口

6) 工具

PC机串口检查。

先锋RS-232C/RS-422A通讯转换器检查。

发送连续脉冲串, 对通讯数据波形测试。

7) 版本信息

2-3. 进入通讯命令学习前的准备工作

2-3-1. 初次连接系统的准备工作 (仪表未连接)

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。

检查通讯系统: 参照屏幕的工具窗口, 以短路线方法, 通过测试软件检查。

PC机232通讯口正常(包括地线、握手信号)。

232接口连线正确。

先锋RS232/RS422接口正确。

先锋RS232/RS422接口到仪表的连线正确。

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或先锋RS232C/RS422A通讯口的分配。

2-3-2. 通常的操作

连接仪表及上电, 确信仪表已进行了后叙有关的通讯参数设定。

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。

执行仪表的通讯测试窗口功能, 画面将自动指示仪表的通讯状态。检测通过后, 程序才能进入下一步的通讯学习窗口。否则用户需先排除软硬件故障。

3. 两种通讯协议以及BASIC程序方法

50系列提供标准(normal) 和与岛电FP21, SR25兼容的(SRFP)两种通讯协议

3-1. S R F P (与岛电SR25和FP21兼容的)通讯协议

3-1-1. 通讯采用了六个专用ASCII码控制符:

字符名称	英文名称	16进制表示	ASCII码	屏幕显示
读写命令的引导符	STX	02H	CHR\$(2)	
读写命令的结束符	ETX	03H	CHR\$(3)	
建立连接命令符	EOT	04H	CHR\$(4)	
关闭连接命令符	ENQ	05H	CHR\$(5)	
正常应答符号	ACK	06H	CHR\$(6)	
不正常应答符号	NAK	15H	CHR\$(&H15)	

BASICA程序例

131 REM 设置六个通讯控制字符

140 STX\$=CHR\$(2):ETX\$=CHR\$(3):EOT\$=CHR\$(4)

150 ENQ\$=CHR\$(5):ACK\$=CHR\$(6):NAK\$=CHR\$(&H15)

3-1-2. 建立通讯的连接命令

上位机的
422
通讯接口

422通讯示意图

发送数据总线 接收数据总线

SR50-(1)

SR50-(2)

SR50(32)

RS422通讯采用差动的两线发送,两线接收的四线制方式。下位调节器的内部接收器的接收高(RDA)和低(RSD)线与上位机RS422A接口的发送数据总线连接,下位调节器内部发送器的发送高(SDA)和低(SDB)线挂在上位机RS422A口的接收数据总线上,通常内部发送器处于高阻关闭态。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。通讯时,上位机必需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,按下图示的顺序首先建立与下位机间的通讯连接。下位调节器在接收地址符合,接收字符格式和校验正确后,将内部发送器开放(变低阻态),作为讲者回送地址和ACK回答符,指示该调节器与上位机的接收数据总线建立了连接,又成为听者正等待上位机的继续通讯命令。不正常时为无响应。

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422方式相同

上位机的
485
通讯接口

485通讯示意图

发送/接收双向数据总线

SR50-(1)

SR50-(2)

SR50(32)

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(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。

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。

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。

上位机 (主)	调节器 (从)
EOT+'地址'+ENQ	
	'地址'+ACK (正常)
	无响应 (不正常)

(上位机和调节器连接命令示意图)

地址:调节器设定的地址号 00~31。多调节器时,设定地址号不能重叠

ASICA程序例:与仪表口地址为"00"的连接

```
200 REM 使用PC COM1口,设置1200波特,偶效验,7位数据,1停止位,禁止联络信号.
210 OPEN "COM1:1200,E,7,1,CD,RS,CS,DS" AS #1:REM 初始化串行口 COM(1)
230 PRINT #1,EOT$:REM 关闭总线上的外设 (GO TO SLEEP)
510 ADR$="00":REM 访问口地址"00"号
530 C$=EOT$+ADR$+ENQ$
540 PRINT #1,C$:REM 建立连接
550 FOR T=0 TO 500:NEXT:REM 延时
560 A$=INPUT$(LOC(1),#1)
570 IF A$=ADR$+ACK$
580 PRINT "连接成功!"
590 REM 转读写
```

3-1-3. 关闭通讯连接:

当主从通讯完成后,上位机重新发送关闭总线的命令.通讯总线上已建立通讯连接的调节器响应后,自动将调节器的内部发送器关闭(转成高阻态),解除了对上位机接收总线的连接。否则将产生总线的竞争,不能进行正常通讯。对RS232接口,由于无三态功能,只能单台点对点间进行通讯。

上位机 (主)	调节器 (从)
EOT	无响应

关闭通讯连接的示意图

关闭通讯连接的BASICA程序例

```
PRINT #1,CHR$(4)
```

3-1-4. 上位机读命令和全文件的组成

读命令是对调节器的控制参数,设置工作内容的读入。

读命令的全文件是由读命令(单字节)读文件,加入引导符,读写命令的结束符(CHR\$(4)),BCC二进制块校验符字符串组成。正常返回的是由引导符,数据文件(读内容),结束符和BCC二进制块校验的字符串。不正常的答复包括对接收全文件字符串格式错误,校验错误并回送的错误号码。

上位机	调节器(下位机)
STX+'文件'+ETX+BCC (数据请求)	
	'错误号'+NAK (不正常答复)

异常

STX+'文件'+ETX+BCC(正常答复)

NAK(请求重发)

(最多三次)

通讯读命令示意图

BASICA程序例

```

10 PRINT #1, CHR$(2)+"读文件"+CHR$(3)+块校验符
20 FOR T=0 to 500:NEXT: REM 延时
30 A$=INPUT$(LOC(1),#1)
40 PRINT A$
50 STOP

```

3-1-5. 上位机写命令和全文件的组成

写命令是对调节器的控制参数,工作参数内容的写入.

写命令的全文件是由写命令以及带或不带写参数的写文件,加上引导符,读写命令的结束符(CHR\$(4)),BCC二进制块校验的字符串组成.

正常返回的是响应码(ACK)的字符.表示写命令成功.

不正常的响应中包括对发送命令格式,校验错误号的回送.

上位机

调节器

STX+'写文件'+ETX+BCC

ACK (正常)

'错误号'+NAK (不正常)

通讯写命令示意图

BASICA程序例

```

PRINT #1, CHR$(2)+"写文件"+CHR$(3)+块校验符

```

3-1-6. 单字节 BCC 块校验码

除字节的偶校验外,传送字符串还采用二进制BCC块校验方式.校验码的产生是将引导符STX后到BCC效验符前的全部数据(含EXT)二进制数求和后得到的单字节码.

例如: STX + "D1" + ETX + BCC

(02H) (44H) (31H) (03H) (78H)

二进制求和 44H+31H+03H=78H (8bits)

转成ASC 码 78H&7FH=78H (7bits) (屏蔽字节第8位)

注:"D1"是读测量值和设定值的命令

BASICA BCC块效验程序例,其中CMD\$为读/写文件,例如:CMD\$="D1"

```

550 CMD$="D1"
560 CMD$=CMD$+ETX$:LEC=LEN(CMD$):BCC=0
570 FOR I=1 TO LEC:S$=MID$(CMD$,I,1)
580 BCC=BCC+ASC(S$)
590 NEXT
600 BCC=BCC MOD 128:REM 屏蔽字节第8位
610 BCC$=CHR$(BCC):REM BCC块校验的ASC码
620 TXD$=STX$+CMD$+BCC$:REM 生成发送读写命令的全文件
630 RETURN

```

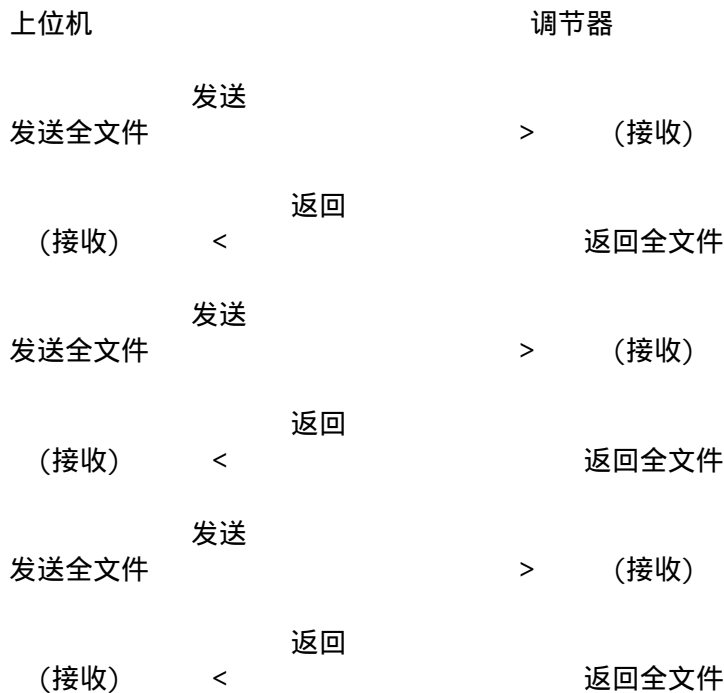
3-1-7. 通讯中的错误,经调节器软件分析后,将自动回送错误类型码(详见错误类型)

3-1-8. 通讯中的定时

A) 上位机发送命令后,3秒内无回答,可视为通讯超时错误.上位机在通讯软件的设计中,必须考虑定时措施,以便能够及时地从超时或通讯故障中退出.

B) 联接命令成功后,5分钟内不读写,仪表将自动关闭通讯连接,再次通讯,需重新建立联接.

3-2. NOMAL 标准通讯协议说明:
 NOMAL标准通讯协议的通迅过程示意图



3-3. 发送全文件和返回全文件的组成, 标准通讯协议和SRFP通讯协议的比较

"@" + "地址" + "文件" + ":" + "二位BCC" + CR

发送全文件和返回全文件的组成图

号码说明:

起始符以 "@" (40H), 表示文件的开始。和SRFP协议的"STX"相当。

地址: 调节器设定的地址号 00 ~ 31。多调节器时, 设定地址号不能重叠

文件: 文件是由单独的读命令或写命令(其后带或不带的参数)组成

文件是以 ASC 码字符 冒号":"(3AH)作为结束。和SRFP协议的"ETX"相当

BCC校验码: 从起始字符"@"后的到文件的结束符":"(含结束符)全部字符的异或运算(XOR), 所得结果的双字节校验码。 SRFP协议的BCC码是单字节校验码。

回车CR符(0DH)表示全文件的结束

BASICA程序例

A) 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)
```

B) 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, 禁止联络信号.
```

```
40 BPS$ = "1200" :ADR$="00" :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
```

C) 双字节BCC块效验("D1"是50的测量值和设定值的读命令)

例如: "@" + "01" + "D1" + ":" + "BCC" + CR
 (40H) (30H)(31H) (44H)(31H) (3AH) (34H)(45H) (0DH)

二进制异或 30H XOR 31H XOR 44H XOR 31H XOR 3AH = 4EH

```
70 CMD$="D1":REM READ PV and SV
```

```
80 BC$ = "00" + "D1" + ":"
```

```
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"
490 RETURN

```

D)仪表口地址为"00"的读

CMD\$="D1" (读命令)

```

20 PPRINT #1,"@"+"00" + "D1"+": "+BCC+CR$: REM 通讯的全文件

```

```

30 FOR T=0 to 500:NEXT: REM 延时

```

```

40 A$=INPUT(LOC(1),#1)

```

```

50 PRINT A$

```

```

60 STOP 读完成

```

C) 仪表口地址为"00"的写(建立通讯工作方式)

```

10 CMD$="@"+"00" + "C1 _COM"+": "+BCC+CR$

```

```

20 PPRINT #1,CMD$

```

```

30 FOR T=0 to 500:NEXT: REM 延时

```

```

40 A$=INPUT(LOC(1),#1)

```

```

50 IF A$=CMD$ :PRINT "通讯工作方式被建立"

```

```

60 STOP 或转读或写命令(参照D)

```

说明: "C1 _LOC" 命令是机内工作方式(仅能读ROM)

D) 仪表口地址为"00"的建立RAM存储工作方式

```

10 CMD$="@"+"00" + "C2 _RAM"+": "+BCC+CR$

```

```

20 PPRINT #1,CMD$

```

```

30 FOR T=0 to 500:NEXT: REM 延时

```

```

40 A$=INPUT(LOC(1),#1)

```

```

50 IF A$=CMD$ :PRINT "RAM存储方式被建立"

```

```

60 STOP 或转读或写命令

```

说明: "C2 _ROM" 命令是ROM(EEPROM)永久存储方式

两种通讯协议的区别

A)两种通讯协议的读写文件格式相同,仅文件的结束符": "或CHR\$(3)有别。

B)在标准通讯协议中,块的地址和数据同时发送,无需像SRFP协议,需首先建立连接 和最后的关闭连接

C) 调节器在接收到全文件后,设备地址符合,对命令进行解释后,作为命令的响应,内部接口发送器(RS422和RS485)对上位机的接收总线开放(变低阻态),并向上位机发送返回的全文件,自动在文件内插入内部设备的地址。完成了对上位机的通讯后,内部接口发送器自动关闭(变高阻态),重新等待新的通讯。此后,上位机也可以继续访问发送数据总线上的其地址号的调节仪表。

如果地址不符合,不作回答,内部接口发送器继续保持关闭,继续等待新的通讯。接收的如果是读命令,返回的是读命令和被读参数等组成的全文件;如果是写命令,返回的是写命令和被后写参数等组成的全文件

SRFP通讯协议,对写命令,返回的仅是回答信号,设备的地址也不插入。

D)调节器在接收过程中,从接收到文件的起始"@"字符后的3秒限定时间内,如果还未接收到有效的文件和发送文件的结束符CR,将重新自动等待下一次的通讯。(下个"@"符)

E) 通讯的定时等待时间

上位机发送命令后,4秒内无回答,可视为通讯超时错误。上位机在通讯软件的设计中,必须考虑定时措施,以便能够及时地从超时或通讯故障中退出。

3-4. 读写命令和读写文件

3-4-1. 命令和文件的说明

通讯命令是由26个大写英文ASC 字符和0~9数字组成,文件可分为单独的读文,单独写命令及带写参数文件,返回的文件。加上引导符、地址,文件的结束符,校验符、结束符后,组成了发送和返回的全文件。

命令的分类

	读命令	
命 令		写命令
	写命令	
地 址		执行键命令

读命令
文件

参数1 参数 2

写命令
文件

参数1 参数2 ; 参数3

返回数据
文件

命令可分为读命令和写命令两类,第一位是英文字符,第二位数字的两位组成。

在本说明中,符号【XXX】仅作为引用的命令表示

读命令是上位机对调节器的设置工作内容的读入。读文件是仅两位的读命令组成。

例如:读PID参数文件【D4】。

写命令是对调节器的控制参数,工作参数内容的写入。写文件是由写命令紧跟着空格(为了便于区分,在后续说明中用字符" "表示空格)和后面的分项参数组成。分项参数以 等表示在参数项的位置,分项参数是由1、4的字符、6位的数值(数字和符号)组成,分项参数间以","逗号作为分隔符。

写文件可采用分项参数的省略格式,亦即局部参数修改格式。分号";"用于省略该分项参数后的参数项,"", "双逗号作为某分项参数的省略符。例如:PID参数的标准写文件是【D4 , , 】,而【D4 ;】或【D4 , ;】或【D4 , , 】,省略格式都是正确的,并仅对局部PID参数进行修改。

写命令文件必需严格按书写格式要求,不得随意增减符号,空格,正负号,改变参数长度和小数点位置。准确记忆和书写命令的文件中参数格式是困难的。规律的是读写命令的返回数据文件正是写命令的标准参考格式,仅需修改文件中的参数回填即可正确。

为便于流程图对照,在本说明中,引用【 】内的数字,来标明命令在流程图的窗口位置。对仅读命令用R字符表示,对仅写命令用W字符表示,对能读能写命令用W/R字符表示。

标准协议返回的全文件中的数据文件是由读写命令紧跟着空格和后面的分项参数组成。数据文件的格式和写文件格式相同,但无省略格式。

SRFP通讯协议,对写命令,返回的仅是回答信号,设备的地址也不插入。

3-4-2. 文件中所使用字符的语法

字符规定:

- 1) 字符...26个大写的英文字母
- 2) 数值的正"÷",负"-"和小数点"."号。
- 3) "?"用于不确定的数据
- 4) "_"字符用于字符参数位的填充。

3-4-3. 数值参数格式说明:

- 1)数值参数包括含正,负号和小数点"."号在内,共计6位固定字长。
- 2)数值的正负号在首位
- 3)数值不够6位以0填充
- 4)第5位必须是数字
- 5)符号位只能是正负号和特殊的英文符号
- 6)+0或-0都表示0.但通讯返回的0以+0表示

例: 1=+00001 -1=-00001

 0.001=+0.001 -0.001=-0.001

 1234=+01234 -1234=-01234

 0=+00000 -0.000=-0.000

- 7) 特殊的数值参数将被插入特殊的英文符号

数值在 +10000 ~ 19999范围内

+12345 U02345

+123.45 U23.45

+10.001 U0.001

即 "U"(55H)=+1000

数值在 -10000 ~ -19999范围内

-12345 D02345

-123.45 D23.45

-10.001 D0.001

即 "D" (44H)=-1000

数值正超量程

"H" (48H) H00000

数值负超量程

"L" (48H) L00000

测温铂电阻异常(断线), 显示为"B___"时

"B" (42H) B00000

测温铂电阻异常(断线), 显示为"C___"时

"C" (43H) C00000

用于不确定的数据

"?" (3FH) ?00000

例如: 程序复位时, [0-0]窗口程序设定值无显示时[----]

3-4-4. 字符参数的格式

1) 4位固定字长。

2) 字符不足4位, 以"_"填充

3) 字符中有空格, 以"_"替换

例: 传感器[rAnG]范围选择

(1 b) (1__b) (4 K1) (4_K1)

(Pt 3) (Pt_b) (0 50) (0_50)

例: __ON __REM

__OFF SPCL

4) 不确定的字符 ?__

3-5-5. 1位字符格式

1) 文件内位参数1位字长

例:

ON =0 OFF=F

YES =Y NO=N

LED灯亮=0 LED灯灭=F

输出 ON=0 输出OFF=F

2) 通讯返回的不确定的位参数以"?"表示

4. SR50通讯命令细则

4-1. 命令的索引:

参照先锋编写的SR50中文操作操作流程图和读写命令索引。SR50的通讯命令可分为流程图的0~7组画面群、键操作命令和LED状态指示灯。共计36组读写命令, 基本包括了全部的操作流程图窗口。在流程图参数窗口的左上脚, 标出了有关的通讯读写命令。命令中标有"R"脚注的, 该命令仅能作为读命令。

4-2. 流程图的有关的通讯参数说明和通讯前的必要设置

选件窗口群

7 - 5 PR键

通讯/机内方式选择

C - m d

L o C

7 - 6 PR键

存储方式选择

A o - L

0

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.

能完成上位机全部的读/写命令. 键设定内部参数被禁止.

rom: 电可擦写的 EEPROM 存储方式

ram: 随机存储方式(断电不保存)

EEPROM 的读写次数在10万次(保证), 为延常使用寿命,

通常的通讯调试中, 建议使用RAM方式.

仅能由上位机控制命令, 完成存储方式的转换.

7 - 7 PR键

通讯的设备号设定 设定范围: 00 ~ 31
仅能由面板键设定

A d d r

0 0

7 - 8 PR键

数据效验停止位设定

d A t A

7 E 1

7E1: 数据7位、偶效验、停止位1位
7E2: 数据7位、偶效验、停止位2位
7n1: 数据7位、无效验、停止位1位
7n2: 数据7位、偶效验、停止位2位
8E1: 数据8位、偶效验、停止位1位
8E2: 数据7位、偶效验、停止位2位
8n1: 数据7位、无效验、停止位1位
8n2: 数据7位、无效验、停止位2位

7 - 9 PR键

通讯的波特率设定

b P S

1 2 0 0

仅能由面板键设定数据格式
1200 bps
2400 bps
4800 bps
9600 bps
仅能由面板键设定通讯的波特率

7 - 10 PR键

通讯协议选择

p r t C

n o m L

PR键
(7 - 0)

1. RS-232C RS-422A接口
nomL: 标准协议
SrFP: 岛电SR25, FP21的准通讯协议
2. RS-485 接口时该窗口变成485的延时时间dELy
设定, 仅一种标准通讯协议
dELy: 从接收到发送间的最小延时时间设定 范围: (0 ~ 255)
延时时间=0.128×设定值
0 设定 =0.128×0 = 0 msec (最小值设定)
80 设定 =0.128×80 = 10.24 msec (初始值设定)
255 设定 =0.128×255 = 32.64 msec (最大值设定)
实际的延时还应加上软件的处理时间. 特别是写命令, 大约有250 msec.
仅能由面板键选择通讯协议

在进行通讯前, 处于机内工作方式时, 必须做以下的手动设置:

- 1) 通讯协议选择NOMAL或SRFP通讯协议
- 2) 选与系统一致的通讯的波特率
- 3) 数据格式选用7E1: 数据7位、偶效验、停止位1位的标准ASCII码
- 4) 设置与通讯链路其它通讯的设备号不发生冲突的设备号

4-3. 读/写命令群的细则:

对命令中符号的说明:

- 【 】 流程图的窗口号
 - () 面板显示窗口字符
 - [[]] 引用的读写文件
 - [] 引用文件内的参数项
- 空格表示

R : [[D3] 读命令文件
W: [[D3 , ,] 写命令文件
返: [[D3 , ,] 返回的文件
 , , 等 文件中分项参数序号

SR50'S 读写命令的关键字索引

0) 基本画面群	[[D1~D6]]	R/W	命令
1) 编程窗口群	[[P3]] R; [[P1、P2、P4]] [[S1~S5]]	R/W	
2) 时间窗口群	[[T1]] R; [[T2]]	R/W	
3) 键锁定窗口群	[[K1]], [[K2]]	R/W	
4) 输入参数窗口群	[[I1]], [[I2]] [[I3]]	R/W	
5) 调节输出参数窗口群	[[O1~O4]]	R/W	
6) 事件窗口群	[[H1]] R; [[V1~V3]] [[H2]]	R/W	
7) 选件窗口群	[[R1]], [[A1]], [[C1]], [[C2]]	R/W	
8) 执行键命令	[[X1~X6]]	W	
9) 面板状态灯指示	[[D8~D9]]	R	
10) 仪表返回的通讯出错信息	[[ER 00~ER 12]]	13种	

0) 基本画面群 [[D1~D6]] R/W 命令
 [[D1]] : 【0 - 0】窗口 R 命令

R : [[D1]]
 返 : [[D1 ,]]
 (PV) : 测量值(数值量)
 (SV) : 设定值(数值量)

 [[D2]] : 【0 - 1】【0 - 2】【0 - 3】窗口 R/W 命令

R : [[D2]]
 W : [[D1 ,]]
 返 : [[D2 ,]]
 (LSV) : 机内设定值(数值量)
 (rSV) : 遥控设定值(数值量)
 (SV-b) : 设定值偏移(数值量)

 [[D3]] : 【0 - 4】【0 - 5】【0 - 6】窗口 R/W 命令

R : [[D3]]
 W : [[D3 ,]]
 返 : [[D3 ,]]
 (EV-1) : 事件1设定和显示(数值量)
 (EV-2) : 事件2设定和显示(数值量)
 (EV-3) : 事件3设定和显示(数值量)

 [[D4]] : 【0 - 7】【0 - 8】【0 - 9】窗口 R/W 命令

R : [[D4]]
 W : [[D4 ,]]
 返 : [[D4 ,]]
 (P) : 比例带P设定(数值量)
 (I) : 积分时间设定(数值量)
 (d) : 微分时间设定(数值量)

 [[D5]] : 【0 - 10】【0 - 11】窗口 R/W 命令

R : [[D5]]
 W : [[D5 ,]]
 返 : [[D5 ,]]
 (mr) : 人工的调节输出补偿(数值量)
 (SF) : 防超抑制系数(数值量)

 [[D6]] : 【0 - 12】窗口 R/W 命令

R : [[D6]]
 W : [[D6]]
 返 : [[D6]]
 (Out) : 调节量指示和自动手动切换(数值量)

1) 编程窗口群

[[P3]] R; [[P1、P2、P4]] [[S1~S5]] R/W

[[P1]]: 【1 - 1】 【1 - 2】窗口 R/W 命令

R: [[P1]]

W: [[P1 ,]]

返: [[P1 ,]]

(StEP): 编程步数设定(数值量)

(S_SV): 设定步的起始设定值(数值量)

[[S1]]: 【1 - 3】窗口 R/W 命令

R: [[S1]]

W: [[S1 , , ,]]

返: [[S1 , , ,]]

(S_01): 第一步的目标值设定(数值量)

(t_01): 第一步运行时间设定(数值量)

(S_02): 第二步的目标值设定(数值量)

(t_02): 第二步运行时间设定(数值量)

[[s2]]: 【1 - 3】窗口 R/W 命令

R: [[S2]]

返: [[S2 , , ,]]

W: [[S2 , , ,]]

(S_03): 第三步的目标值设定(数值量)

(t_03): 第三步运行时间设定(数值量)

(S_04): 第四步的目标值设定(数值量)

(t_04): 第四步运行时间设定(数值量)

[[S3]]: 【1 - 3】窗口 R/W 命令

R: [[S3]]

W: [[S3 , , ,]]

返: [[S3 , , ,]]

(S_05): 第五步的目标值设定(数值量)

(t_05): 第五步运行时间设定(数值量)

(S_06): 第六步的目标值设定(数值量)

(t_06): 第六步运行时间设定(数值量)

[[S4]]: 【1 - 3】窗口 R/W 命令

R: [[S4]]

W: [[S1 , , ,]]

返: [[S4 , , ,]]

(S_07): 第七步的目标值设定(数值量)

(t_07): 第七步运行时间设定(数值量)

(S_08): 第八步的目标值设定(数值量)

(t_08): 第八步运行时间设定(数值量)

[[S5]]: 【1 - 3】窗口 R/W 命令

R: [[S5]]

W: [[S5 , , ,]]

返: [[S5 , , ,]]

(S_09): 第九步的目标值设定(数值量)

(t_09): 第九步运行时间设定(数值量)

(S_10): 第十步的目标值设定(数值量)

(t_10): 第十步运行时间设定(数值量)

[[P2]]: 【1 - 4】窗口 R/W 命令

R: [[P2]]

返: [[P2]]

W: [[P2]]

(rPt): 曲线重复次数的设定(数值量)

[[P3]]: 【1 - 5】 【1 - 6】 【1 - 7】窗口 R/W 命令

R: [[P3]]

W: [[P3 , , ,]]

返：[[P3 , ,]]
(E_ti): 程序步剩余时间显示(数值量)
(E_SP): 当前运行步号指示 (数值量)
(E_rP): 曲线运行重复数指示(数值量)
[[P4]]: 【 1 - 8 】窗口 R/W 命令
R : [[P4]]
W: [[P4 , ,]]
返： [[P4 , ,]]
(Prog): 程序控制方式选择(字符型)
[ON]: 程序控制
[OFF]: 定值控制

2) 时间窗口群 [[T1]] R; [[T2]] R/W
[[T1]]: 【 2 - 1 】窗口 R/W 命令
R : [[T1]]
W: [[T1 , ,]]
返： [[T1 , ,]]
(E_St): 定时起动时间的剩余时间指示(数值量)
(E_Ed): 定时结束时间的剩余时间指示(数值量)
[[T2]]: 【 2 - 2 】 【 2 - 3 】 【 2 - 4 】窗口 R/W 命令
R : [[T2]]
W: [[T2 , ,]]
返： [[T2 , ,]]
(t_St): 定时起动时间设定(数值量)
(t_Ed): 定时结束时间设定(数值量)
(t_md): 时间方式选择 (字符型)
[_OFF]: 取消定时方式
[_EC]: 定时控制方式
[_T1]: 纯时间信号方式
[_PON]: 上电定时方式

3) 键锁定窗口群 [[K1]] [[K2]] R/W
[[K1]]: 【 3 - 1 】 【 3 - 2 】窗口 R/W 命令
R : [[K1]]
W: [[K1 , ,]]
返： [[K1 , ,]]
(SV_L): 设定值下限范围限定(数值量)
(SV_H): 设定值上限范围限定(数值量)
[[K2]]: 【 3 - 8 】 【 3 - 9 】窗口 R/W 命令
R : [[K2]]
W: [[K2 , ,]]
返： [[K2 , ,]]
(di _1): 外部开关1作用选择(字符型)
(di _2): 外部开关2作用选择(字符型)
[_NON]: 取消定义
[_SB]: 设定值偏移
[_AT]: 自整定
[_DA]: 正/反作用选择
[_EC]: 控制运行/解除
[_MAN]: 手动/自动调节输出
[_REM]: 模拟遥控
[_ADV]: 程序运行中的跳步
[_HLD]: 程序运行中的保持

4) 输入参数窗口群 [[I1]] [[I2]] [[I3]] R/W
[[I1]]: 【 4 - 1 】 【 4 - 2 】窗口 R/W 命令
R : [[I1]]
W: [[I1 , ,]]
返： [[I1 , ,]]

(PV_b): 测量值补偿设定 (数值量)

(PV_F): 测量值滤波常数设定(数值量)

[[12]]: 【4 - 3】【4 - 4】【4 - 5】窗口

R/W 命令

R: [[12]]

W: [[12], ,]

返: [[12], ,]

(rAnG): 信号的测量范围选择(字符量子)

热电偶选择表(, 华氏略)

[TC_B]:	B	0 ~ 1800
[TC_R]:	R	0 ~ 1700
[TC_S]:	S	0 ~ 1700
[TCK1]:	K1	-100.0 ~ 400.0
[TCK2]:	K2	0.0 ~ 800.0
[TCK3]:	K3	0 ~ 1200
[TC_E]:	E	0.0 ~ 700.0
[TC_J]:	J	0.0 ~ 600.0
[TC_T]:	T	-199.9 ~ 200.0
[TC_N]:	N	0 ~ 1300
[TCPL]:	PL	0 ~ 1300
[TCWR]:	WRe5-26	0 ~ 2300
[TC_U]:	U	-199.9 ~ 200.0
[TC_L]:	L	0.0 ~ 600.0

铂电阻选择表(, 华氏略)

[PT_1]	- 199.9 ~ 600.0
[PT_2]	-100.0 ~ 100.0
[PT_3]	-100.0 ~ 300.0
[PT_4]	- 50.0 ~ 50.0
[PT_5]	0.00 ~ 50.00
[PT_6]	0.0 ~ 99.99
[PT_7]	0.0 ~ 100.0
[PT_8]	0.0 ~ 200.0
[PT_9]	0.0 ~ 500.0

电流输入(mA)组

[MA_1]	0 ~ 20 mA
[MA_2]	4 ~ 20 mA

电压选择表(mV)组

[MV_1]	-10 ~ 10 mv
[MV_2]	0 ~ 10 mv
[MV_3]	0 ~ 20 mV
[MV_4]	0 ~ 50 mV
[MV_5]	10 ~ 50 mV
[MV_6]	0 ~ 100 mV

电压选择表(V)组

[V_1]	-1 ~ 1 V
[V_2]	0 ~ 1 V
[V_3]	0 ~ 2 V
[V_4]	0 ~ 5 V
[V_5]	1 ~ 5 V
[V_6]	0 ~ 10 V

(unit): 摄氏-华氏选择(字符)

[__C]:

[__F]: F

(type): 铂电阻类型选择(字符)

[__PT]: Pt100标准

[_JPT]: JPt100标准

[[I3]]: [[4 - 6]] [[4 - 7]] [[4 - 8]] [[4 - 9]] 窗口 R/W 命令

R: [[I3]]

W: [[I3 , , ,]]

返: [[I3 , , ,]]

dP : 直流小数点选择(字符)

[__]

[_.]

[. _]

[. _]

(SC_L): 直流输入下限刻度设定(数值)

(SC_H): 直流输入上限刻度设定(数值)

(root): 直流输入开平方选择(字符)

[_ON]: 开平方

[_OFF]: 不开平方

5) 调节输出参数窗口群 [[01~04]] R/W

[[01]]: [[5 - 1]] [[5 - 2]] [[5 - 3]] 窗口 R/W 命令

R: [[01]]

W: [[01 , ,]]

返: [[01 , ,]]

(o_md): 普通-线性调节输出限幅选择(字符)

[NOML]: 普通限幅

[SPCL]: 线性限幅

(o_SL): 特殊输出功率限幅下限(数值)

(o_SH): 特殊输出功率限幅上限(数值)

[[02]]: [[5 - 4]] [[5 - 5]] 窗口 R/W 命令

R: [[02]]

W: [[02 ,]]

返: [[02 ,]]

(o_L): 调节输出下限限幅(数值)

(o_H): 调节输出上限限幅(数值)

[[03]]: [[5 - 6]] [[5 - 7]] [[5 - 8]] 窗口 R/W 命令

R: [[03]]

W: [[03 , ,]]

返: [[03 , ,]]

(o_dF): 位式控制动作灵敏度(数值)

(o_Cy): Y-P类型比例周期设定(数值)

(o_AC): 调节输出正/反作用选择(字符)

[_RA]: 反作用

[_DA]: 正作用

[[04]]: [[5 - 9]] [[5 - 10]] 窗口 R/W 命令

R: [[04]]

W: [[04 ,]]

返: [[04 ,]]

(At_P): AT整定点设定(数值)

(Ctrl): 调节算法选择(字符)

[_PID]: PID控制方式

6) 事件窗口群 [[H1]] R; [[V1~V3]] [[H2]] R/W

[[V1]]: [[6 - 1]] [[6 - 2]] [[6 - 3]] 窗口 R/W 命令

R: [[V1]]

W: [[V1 , ,]]

返：【V1 , , 】

(E1_m):事件1输出种类定义(字符量子)

[_NON]: 无或取消设置

报警类	定时类
[_PHL]: P V 上限绝对值报警	[T_ST]: 定时的起动机标输出
[_PLL]: P V 下限绝对值报警	[T_ED]: 定时的起动机标输出
[_SHL]: S V 上限绝对值报警	[T_SE]: 定时的起动机结束复合时标输出
[_SLL]: S V 下限绝对值报警	
[_DH]: P V 上限偏差值报警	其它类
[_DL]: P V 下限偏差值报警	[_AT]: 自整定过程中吸合指示,完成后断开
[L_H]: P V 上下限偏差值内报警	[_SO]: 测量值超量程时的吸合指示
[_LH_]: P V 上下限偏差值外报警	[_RUN]: 程序运行中吸合指示结束后断开
	[_END]: 程序运行结束时的吸合一秒指示
	[STEP]: 程序步进时和程序结束时的吸合一秒指示

(E1_d):事件1报警回差设定 (数值量)

(E1_s):事件1上电报警态抑制(字符量子)

[_ON]:抑制

[_OFF]:非抑制

【V2】: 【6 - 4】 【6 - 5】 【6 - 6】窗口 R/W 命令

R: 【V2】

W: 【V2 , , 】

返: 【V2 , , 】

(E2_m):事件2输出种类定义 (字符量子)

(E2_d):事件2报警回差设定 (数值量)

(E2_s):事件2上电报警态抑制(字符量子)

【V3】: 【6 - 7】 【6 - 8】 【6 - 9】窗口 R/W 命令

R: 【V3】

W: 【V3 , , 】

返: 【V3 , , 】

(E3_m):事件3输出种类定义 (字符量子)

(E3_d):事件3报警回差设定 (数值量)

(E3_s):事件3上电报警态抑制(字符量子)

【H1】: 【6 - 7】窗口 R/W 命令

R: 【H1】

返: 【H1 , 】

(Hb_A):控制输出ON时,加热回路电流值

(HI_A):控制输出OFF时,加热回路电流值

【H2】: 【6 - 8】 【6 - 9】 【6 - 10】窗口 R/W 命令

R: 【H2】

W: 【H2 , , 】

返: 【H2 , , 】

(Hb_S):断线报警电流值设定 (数值量)

(HL-S):加热环报警电流设定 (数值量)

(Hb-m):报警动作方式设定 (字符量子)

[LOCK]:锁定方式

[REAL]:实时方式

7) 选件窗口群 【R1】 【A1】 【C1】 【C2】 R/W

【R1】: 【7 - 1】 【7 - 2】 【7 - 3】 【7 - 4】窗口 R/W 命令

R: 【R1】

W: 【R1 , , , 】

返: 【R1 , , , 】

(rE_L): 模拟遥控下量程设定(数值量)
 (rE_H): 模拟遥控上量程设定(数值量)
 (rE_b): 模拟遥控设定值偏移 (数值量)
 (rE-F): 模拟遥控滤波常数设定(数值量)
 [[A1]]: 【7 - 5】【7 - 6】【7 - 7】窗口 R/W 命令
 R: [[A1]]
 W: [[A1 ,]]
 返: [[A1 ,]]
 (Ao_m): 模拟发送类型选择(字符型)
 [__PV]: 模拟发送测量值
 [__SV]: 模拟发送设定值
 (Ao_L): 模拟发送下限设定(数值量)
 (Ao_H): 模拟发送上限设定(数值量)
 [[C1]]: 【7 - 5】窗口 R/W 命令
 R: [[C1]]
 W: [[C1]]
 返: [[C1]]
 (C_md): 通讯/机内方式选择(字符型)
 [__LOC]: 机内方式
 [__COM]: 通讯方式
 [[C2]]: 【7 - 6】窗口 R/W 命令
 R: [[C2]]
 W: [[C2]]
 返: [[C2]]
 (m_md): 存储方式选择(字符型)
 [__ROM]: 存入EEPROM
 [__RAM]: 存入RAM
 8) 执行键命令 [[X1 ~ X6]] W
 [[X1]]: W 命令
 W: [[X1]]
 [EXEC]: 控制的执行/解除
 [[X2]]: W 命令
 W: [[X2]]
 [__REM]: 模拟遥控/机内控制
 [[X3]]: W 命令
 W: [[X3]]
 [__MAN]: 手动/自动调节输出
 [[X4]]: W 命令
 W: [[X4]]
 [__AT]: 自整定的起动/解除
 [[X5]]: W 命令
 W: [[X5]]
 [__HLD]: 程序运行中的保持/继续
 [[X6]]: W 命令
 W: [[X6]]
 [__ADV]: 程序运行中的跳步
 9) 面板状态灯指示 [[D8 ~ D9]] R
 [[D8]]: R 命令
 R: [[D8]]
 返: [[D8 ,]]

EV1: 事件1动作指示(字符量)
 EV2: 事件2动作指示(字符量)
 EV3(HB): 事件3动作指示/加热器报警指示(字符量)
 [0]: 动作时ON
 [F]: 不动作时OFF

[[D9]] : R 命令

R : [[D9]]

返 : [[D9 , , , , , , , , ,]]

AT : 自整定的起动/解除(字符量)
 PRG : 程序方式ON/OFF(字符量)
 COM : 通讯方式ON/OFF(字符量)
 REM : 模拟遥控执行/解除(字符量)
 MAN : 手动/自动调节输出(字符量)
 EXEC: 控制的执行/解除(字符量)
 HLD : 程序运行中的保持/继续指示(字符量)
 SB : 设定值偏移/不偏移(字符量)
 [0]: ON时 [F]: OFF时

10) 仪表返回的出错信息

九种通讯的出错信息 [[ER 01]] [[ER 05 ~ ER 12]]

错误号	错误种类	错误内容
[ER 01]:	硬件错误	通讯格式, 接收缓存器数据重迭, 校验位错误
[ER 05]:	BCC效验错误	接收数据的再次BCC校验和接收的BCC码不符
[ER 06]:	错误的命令	写命令用于机内工作方式, 未定义的错误命令
[ER 07]:	文本格式错误	错误地使用"空格, 逗号, 省略冒号, 数据位数
[ER 08]:	数据格式错误	数值数据、字符数据、位数据的格式错误
[ER 09]:	数据错误	数值数据超限、字符错误
[ER 10]:	执行命令错误	因处于某工作方式而不能执行的命令键
[ER 11]:	写命令错误	写命令错误地用于不能修改的数据
[ER 12]:	配置和选件错误	与仪表配置和选件条件不符合的错误命令

说明: 在ER后跟着空格。

5. BASICA的程序通讯软件说明:

5-1. 在用户的DOS3.0以上的操作系统上, 插入#1号软盘起动BASICA后, 可列表打印或执行带有.BAS后缀的源程序.

50SRFP.BAS 和50NOM.BAS分别是于FP21和SR25通用的通讯协议和SD20通用的新通讯协议编写的有关50的"D1"命令连续数据采集程序例。用户可参考测量值和设定值的数据采集, PC机通讯口初始化, 发送接收缓存区的访问, BCC校验, 接收文件的自动分类和错误分类, 通讯定时关系等子程序, 扩展应用程序。

232T.BAS为外设通讯口测试程序(19200BPS), 对COM1通讯口发接和屏幕显示发接字符, 完成:

短路PC机COM1口的2, 3端, 检测PC机通讯口的好坏。

将先锋422转换器口的SDA接RDA, SDB接RDB后, 分别检测四个通讯口。

参照 的短接方式, 在通讯距离内(最远1200米), 利用示波测量发送波形的前沿, 确定通讯线路的品质以及选择合适的通讯波特率。

5-2 在PC计算机上, 采用BASICA语言和SRFP通讯协议, 对50数据采集的编程例 (RS232/RS422接口)

5-2-1 50SRFP.BAS程序清单及流程图

仪表窗口需设置 BPS=1200 地址=00 数据格式=7E1(数据7位、偶效验、1个停止位) 选SRFP 通讯协议
 PC机需设置COM1口 波特率BPS=1200 数据格式=E, 7, 1(数据7位、偶效验、1个停止位) 本程序对PC握手信号无要求

```

130 REM 50的 "D1"命令数据采集软件PC机应用示例
131 REM 设置六个通讯控制字符
140 STX$=CHR$(2): ETX$=CHR$(3): EOT$=CHR$(4)
150 ENQ$=CHR$(5): ACK$=CHR$(6): NAK$=CHR$(8&H15)
160 BPS$="1200" :REM 设置1200通讯波特率
210 OPEN "COM1: "+BPS$+", E, 7, 1, CD, RS, CS, DS" AS #1:REM 初始化串行口, 握手无效
230 PRINT #1, EOT$:REM 关闭总线上的外设 (GO TO SLEEP)

```

```

500 CMD$="D1"
510 ADR$="00": REM 访问口地址"00"号
520 RC=0 : REM 开始通讯,设置重发次数"0"次
530 C$=EOT$+ADR$+ENQ$: GOSUB 1150 : REM 调用联接
540 GOSUB 560: REM 调用发送文件的BCC校验
545 GOSUB 640 : REM 调用数据接收
550 GOTO 500: REM 循环或继续
560 CMD$=CMD$+ETX$: LEC=LEN(CMD$): BCC=0: REM BCC块效验
570 FOR I=1 TO LEC: S$=MID$(CMD$, I, 1)
580 BCC=BCC+ASC(S$)
590 NEXT
600 BCC=BCC MOD 128
610 BCC$=CHR$(BCC)
620 TXD$=STX$+CMD$+BCC$
630 RETURN
640 PRINT #1, TXD$: REM 发送文件
641 PRINT "SENDING DATA = "; TXD$
650 T3=VAL(MID$(TIME$, 7, 2)) : REM 2秒定时
660 IF EOF(1)=0 THEN 700: REM 接收缓存器有效否 ?
670 T4=VAL(MID$(TIME$, 7, 2))
680 IF ABS(T4-T3)<2 THEN 660: REM 超过等待时间否?
690 PRINT"超过2秒,通讯出错!": RETURN
700 D$="" : REM 接收数据
710 A$=INPUT$(1, #1): REM 接一个字符
720 D$=D$+A$: REM 拼字符串
730 IF A$= ACK$ THEN GOTO 1040 : REM 写命令跳转
740 IF A$=ETX$ THEN 780 : REM 读命令跳转
750 IF A$=NAK$ THEN 1060 : REM 错误命令跳转
760 GOTO 710
770 REM 接收数据的BCC校验
780 A$=INPUT$(1, #1)
790 H$=D$: D$=D$+A$
800 LEC=LEN(H$): BCC=0
810 FOR I=2 TO LEC: S$=MID$(H$, I, 1)
820 BCC=BCC+ASC(S$)
830 NEXT
840 BCC=BCC MOD 128
850 BC$=CHR$(BCC): TXD$=NAK$
860 IF A$<>BC$ THEN 1030: REM 接收数据BCC错误,跳转重发
870 LEC=LEN(D$): F$="": K=1: REM 除掉数据中的控制符
880 X$=MID$(D$, 2, 2) : REM STX$, ETX$, BCC$
890 FOR P=5 TO LEC : REM 以", "号为分割,确定接收参数
900 N$=MID$(D$, P, 1) : REM 个数N,用于的字符串分解.
910 IF N$=", " THEN U$(K)=F$: K=K+1: F$="" : GOTO 940
920 IF N$=ETX$ THEN U$(K)=F$: N=K : GOTO 990
930 F$=F$+N$
940 NEXT
990 PRINT"读数据BCC校验成功 ! " TAB(25) "RD = "; D$
1000 FOR N=1 to K
1010 PRINT U$,
1020 NEXT N
1010 PRINT
1020 RETURN: REM 返回
1030 RC=RC+1
1035 IF RC< 4 THEN GOTO 640 ELSE PRINT"DATA BCC ERROR!!": GOTO 1010
1040 IF D$=ADR$+ACK$ THEN RETURN 1200: REM 连结或写命令判断

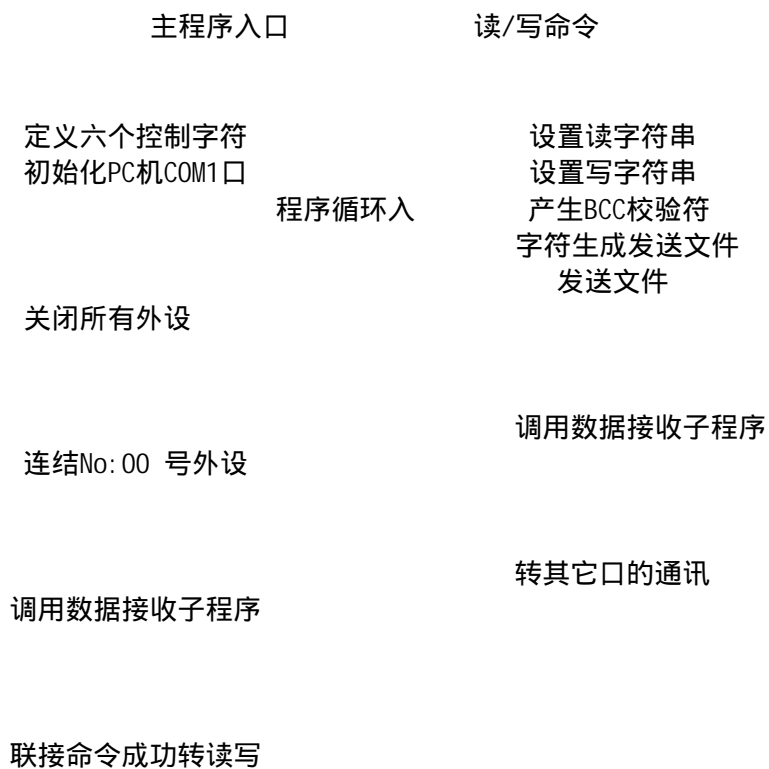
```

```

1050 PRINT"写命令成功! RD = ";:PRINT D$;TAB(32);"TD=";TXD$:GOTO 1020
1060 B$=LEFT$(D$,3):REM 查出文件的错误类型
1140 PRINT "错误号 No. = ";B$:GOTO 1020
1150 TS=0
1161 PRINT
1160 PRINT #1,C$:TS=TS+1
1161 PRINT "联结命令 = ";C$
1170 GOSUB 650
1180 IF TS<3 THEN 1160 ELSE PRINT"无法联结(查错:地址,通讯速度,联机等)"
1190 RETURN
1200 PRINT D$; " 地址连接成功 !!"
1210 RETURN 540

```

5-2-2 50SRFP. BAS软件通讯程序流程图



F P 2 1、S R 2 5、S R 5 0 通用接收发送子程序



主程序入口

读/写命令

5-3: SR50N. BAS程序清单及流程图(NORMAL 通讯协议 ,RS232/RS422接口用)

流程图与SRFP协议相似,但无需建立连接便可执行读写命令。

5-3-1: SR50N. BAS程序清单

10 CLS

20 REM 程序名 SR50N. BAS 仪表需设置 BPS=1200 地址=00 数据格式=7E1(数据7位、偶

22 REM PC机需设置 COM1口 波特率BPS=1200 数据格式=E,7,1(数据7位、偶效验、1个

30 STX\$ = "@": ETX\$ = ": " :CR\$=CHR\$(13) :REM 设置三个控制符

40 BPS\$ = "1200" :ADR\$="00" :REM 设置波特率和仪表通讯地址

45 R=0: S=0 :REM 设置计数

50 OPEN "COM1:" + BPS\$ + ",E,7,1,CD,RS,CS,DS" AS #1: REM 初使化PC机和设数据格

60 REM LINE INPUT "ORDER="; CMD\$ 可单独输入命令的语句

70 CMD\$="D1"

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

90 GOSUB 420 : REM 建立BCC块效验

100 TXD\$ = STX\$ + BC\$ + BCC\$ + CR\$

110 PRINT #1, TXD\$; :REM 发送文件

```

120 PRINT "SENDING DATA = "; TXD$
130 T3 = VAL(MID$(TIME$, 7, 2)): REM 定时4秒
140 IF EOF(1) = 0 THEN 170 :REM 接收数据缓存器空否?
150 T4 = VAL(MID$(TIME$, 7, 2))
160 IF ABS(T4 - T3) < 4 THEN 140 ELSE R=R+1:PRINT "OVER 4S AND COMMUNICATION ERROR!"
165 IF R < 3 THEN 110 ELSE PRINT "YOU' TRIED 3 TIMES! PLEASE CHECK!"
170 D$ = "": REM 接收数据
180 A$ = INPUT$(1, #1): REM 接收1个字符
190 D$ = D$ + A$ :REM 拼字符串
200 IF A$ = CHR$(13) THEN GOTO 220:REM 接收完成(回车符)
210 GOTO 180
220 RBCC$ = LEFT$(RIGHT$(D$, 3), 2): REM 接收数据的BCC码
230 LEC = LEN(D$)
240 BC$ = MID$(D$, 2, LEC - 4)
250 GOSUB 420: REM 读文件的BCC块效验
260 IF RBCC$ = BCC$ THEN 290: REM BCC块效验正确否?
265 ELSE S=S+1
270 IF S < 3 THEN 80
280 ELSE PRINT "BCC ERROR ! YOU'VE TRIED 3 TIMES! PLEASE CHECK!"
290 LEC = LEN(D$): F$ = "": K = 1 :REM 分解接收文件的参数
300 FOR P = 6 TO LEC
310 N$ = MID$(D$, P, 1)
320 IF N$ = ", " THEN U$(K) = F$: K = K + 1: F$ = "": GOTO 350
330 IF N$ = ETX$ THEN U$(K) = F$: N = K: GOTO 360
340 F$ = F$ + N$
350 NEXT
360 PRINT "RECEIVING DATA="; D$
370 FOR N = 1 TO K
380 PRINT U$(N),
390 NEXT
400 PRINT
410 GOTO 60:REM 建立BCC块效验
420 BCC = ASC(LEFT$(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"
490 RETURN

```

说明: 将语句60代替70, 可以对话方式输入其它读命令

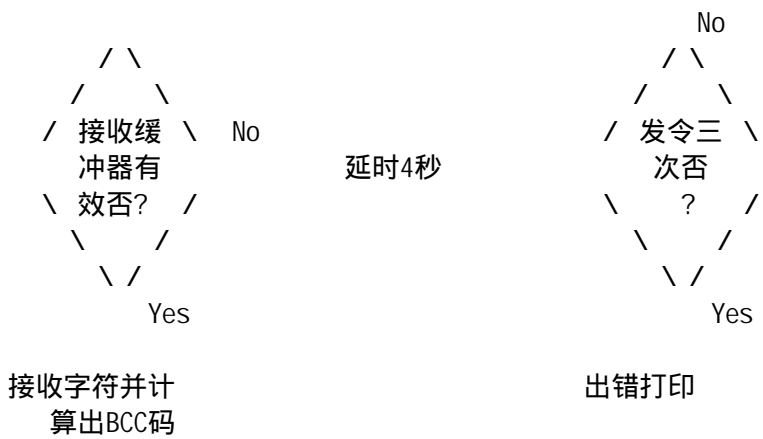
5-3-2. 流程图

初始化通讯口

接受命令并计
算出BCC码

命令重发

发送命令

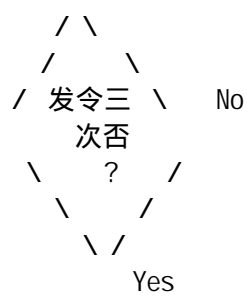


接收字符并计算出BCC码

出错打印



打印出所接收的参数



出错打印

5-4 485通讯接口和BASIC程序方法
5-4-1 RS485通讯示意图

上位机的
485
通讯接口

485通讯示意图

发送/接收双向数据总线

SR50-(1)

SR50-(2)

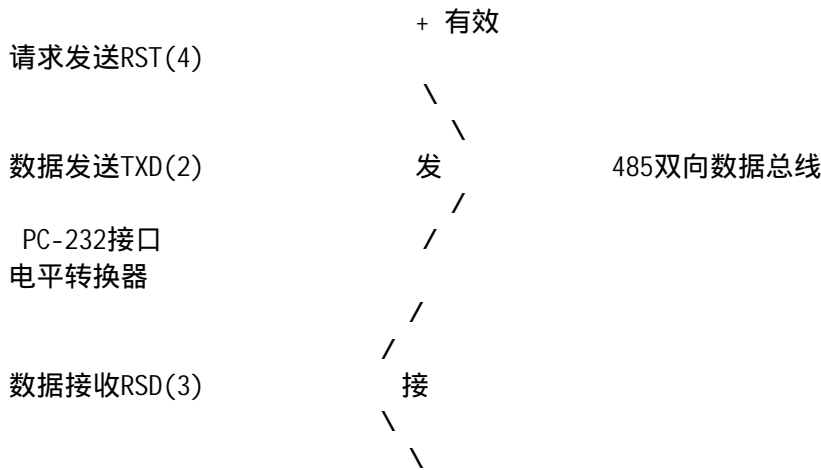
SR50(32)

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(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。

5-4-2 RS485双向数据总线转换硬件示意图



232/485转换硬件示意图

RS485接口要求在发送数据完成后, 立即关闭发送, 否则无法接收其它设备的通讯。而存在着双向数据总线转换冲突和发送数据被自己接收的问题, 在上位机的通讯软件的设计中, 可采用两种方法: UART 的发送寄存器空的位测试命令 完整的接收到自发的数据, 来确认发送数据完成, 以便及时地关闭发送。

上位机的RS232/485转换器通常是利用232口的RST请求发送信号的位置位/位复位信号, 作为发送数据总线的转换控制。在BASIC程序 OPEN "COM 1 , 1200, E, 7, 1, CD, RS, CS, DS" AS #1命令后, 初始化PC机的通讯口, RST信号置零, 使发送驱动器变成高阻输出。发送数据时, "OUT(&H3FC), &H0B"的命令, 使经UART 8250输出的RST信号置高, 令发送驱动器变成低阻输出; 发送数据完成后, 输出"OUT(&H3FC), &H09"命令, 又将RST信号置零, 发送驱动器恢复成高阻输出。

下位仪表, 可在仪表的RS485延时时间窗口, 根据通讯速度, 调整发送数据总线的转换时间。

5-4-3: SR50N48. BAS程序清单及流程图(NORMAL 通讯协议, RS485接口用)

流程图与SR50N. BAS流程相似, 区别在于增加了确认发送数据完成以及总线的转换命令。

6. 附录:

A. 通讯串口接线方法

RS-232C通讯口接线示意图

数据发送	SD 2	RD	
数据接收	RD 3	SD	
请求发送	RTS 4		SR50系列
清除发送	CTS 5		RS-232C
数据设备准备	DSR 6		
载波检测	CD 8		
数据终端准备	DTR20		
信号地	7	SG	

PC机 25 针 RS-232C

仪表9针 RS232 (端子号见使用说明书)

RS-422A通讯口接线示意图

SD 2	3 口一 (6)	RD+ 接受高
RD 3	2 口二 (4)	RD- 接受低
PC机 RS 4	口三 (9)	SD+ 发送高

CS 5	口四 (3)	SD- 发送低
25 针 DS 6	(5)	SG 信号地
RS-232C CD 8	先锋232/422	
DR20	转换器	仪表的422
信号地 7	7	接线端子

说明: 1. 短距离时, 可不使用信号地和屏蔽线.

2. 转换器提供了4个RS422驱动口, 在总线上都是并联的, 使用数量, 具体视现场的通讯距离来分配.

RS-422A通讯口接线示意图

	2	2 口一 (6)	RD+ 接受高
	3	3 口二 (4)	RD- 接受低
PC机	7	口三 (9)	SD+ 发送高
	8	口四 (3)	SD- 发送低
9 针	1	(5)	SG 信号地
RS-232C	4	先锋232/422	
	6	转换器	仪表的422
	5	7	接线端子

PC机 RS-232C 串口25针与9针接线对照表:

9PIN	1	2	3	4	5	6	7	8	9
25PIN	8	3	2	20	7	6	4	5	22
13				1		5		1	
25				14		9		6	

25针连接器接线图

九针准连接器接线图

B. RS232C通讯口的技术数据

信号电平: EIA RS-232C 电平(±12V)
 通讯方式: RS232C 3线半双工
 同步系统: 起始位-停止位, 异步通讯
 通讯距离: RS232C 15 米
 通讯速度: 1200, 2400, 4800, 9600 波特率
 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
 数据块校验: 数据异或(双字节)
 通讯码: ASCII
 握手信号: 未使用
 连接台数: RS-232C 1 台

C. RS422/RS485通讯接口的技术数据

信号电平: EIA RS422A/485 电平 5V差动
 通讯方式: RS422A 4线半双工(多路)/RS485 2线半双工(多路)
 同步系统: 起始-停止位同位, 异步通讯
 通讯距离: 1200 米
 通讯速度: 1200, 2400, 4800, 9600 波特率
 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

数据块校验: 异或(双字节)

通讯码: ASCII

握手信号: 未使用

连接台数: RS-422/485 32 台 1.5公里(配先锋RS232/422接口转换器)

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");
    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++)
printf("%c", receive[j]);
for(j=21; j<24; j++) ccc[j-21]=receive[j];
flag=atoi(ccc);
gotoxy(35, 3+number);
if(flag&0x01) printf("SAFE");
else printf("ACTI");
gotoxy(43, 3+number);
if(flag&0x02) printf("SAFE");
else printf("ACTI");
gotoxy(50, 3+number);
if(flag&0x100)printf("SAFE");
else printf("ACTI");
gotoxy(57, 3+number);
if(flag&0x04) printf("ENAB");
else printf("DISA");
gotoxy(64, 3+number);
if(flag&0x20) printf("MANU");
else printf("AUTO");

```

```

/*Transmitter Function*/

```

```

void transmitter(char *send, int BASE_ADD, int len)

```

```

int j;
outportb(BASE_ADD+4, 0x02); /*enable RTS, enable send*/
for(j=0; j<len; j++)
outportb(BASE_ADD, send[j]); /*send a byte*/
/*check if transmitting shift register is empty*/
while((inportb(BASE_ADD+5)&0x40)==0);

```

```

return;

```

```

/* Receiver Function, success return the number of received byte,
failure return 0 */

```

```

int receiver(char *rec, int BASE_ADD)

```

```

int i=0;

```

```

long j=0;

```

```

outportb(BASE_ADD+4, 0); /*disable RTS, enable receive*/

```

```

while((inportb(BASE_ADD+5)&0x01)!=1) /* receive data available? */

```

```

j++; if(j>20000) return(0);

```

```

rec[0]=inportb(BASE_ADD); /*receive data*/

```

```

do

```

```

while((inportb(BASE_ADD+5)&1)!=1) /*receive data available?*/

```

```

j++; if(j>20000) return(0);

```

```

rec[i]=inportb(BASE_ADD); /*receive data*/

```

```

i++;

```

```
while(rec[i-1]!=0x2a);  
return(i);
```