XJY-160 仪表采用 RS485 传输标准与计算机通迅，详细资料如下：0X

波特率：1200，2400，4800，9600

起始位：1

数据位：8

校验位：无

停止位：1

一．MODBUS_RTU 帧结构

消息发送至少要以 3.5 个字符时间的停顿间隔作为开始。整个消息帧必须作为连续的数据流来传送，如果整个数据流在传送完成之前发生超过 3.5 个字符时间的传送停顿，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，接收设备将认为它是前一消息的延续。

一帧信息的标准结构如下所示：

开始	地址域	功能域	数据域	CRC 校验	结束
T1-T2-T3-T4	8Bit	8Bit	n 个 8Bit	16Bit	T1-T2-T3-T4

地址域: “主机”是通过把“要联络的从机”的地址放入“消息”中的“地址域”来选择相应的“从设备”的；“从机”的地址范围是 1…15(十进制)。

功能域：有效的编码范围是 1…255（十进制）；“从机”收到的“消息”（发自“主机”）里的“功能代码”将告诉“从机”，“主机”需要它完成的动作。例如：读/写一组寄存器的数据内容等。

数据域：“主机”发给“从机”的“数据域”中，包含了“从机”要完成“功能域”指示的动作时所必须的附加信息。如：寄存器地址等。

CRC 校验：CRC 生成之后，低字节在前，高字节在后。

注：本仪表通讯时帧与帧之间的响应间隔，在通讯速率等于 9600bps 时不大于 5ms。

二．XJY-160 参数集

菜单一设置参数（PASS:0001）						
设置参数	**设置内容**	**地址**	**操作**	**参数范围**	**数据类型**	**小数点位**
YEAR	**系统日期--年**	**0X0906**	**读/写**	**00~99**	**单字节（自动）**	**无**
MON	**系统时间—月**	**0X0904**	**读/写**	**01~12**	**单字节（自动）**	**无**
DATE	**系统时间—日**	**0X0903**	**读/写**	**01~31**	**单字节（自动）**	**无**
HOUR	**系统时间—小时**	**0X0902**	**读/写**	**00~23**	**单字节（自动）**	**无**
MIN	**系统时间—分钟**	**0X0901**	**读/写**	**00~59**	**单字节（自动）**	**无**
SEC	**系统时间—秒**	**0X0900**	**读/写**	**00~59**	**单字节（自动）**	**无**
P-T	**打印时间间隔**	**0X05A0**	**读/写**	**0001~9999(Min)**	**双字节(非易失)**	**无**
PrT	**打印使能**	**0X041F**	**读/写**	**0:无打印功能 1:手动打印 2:自动打印 3:手动+自动**	**单字节(非易失)**	**无**

菜单二设置参数（PASS:0036）						
ID	**通讯地址**	**0X05B0**	**只读**	**1-15**	**单字节（非易失）**	**无**

BAUD	通讯波特率	0X05C0	只读	**0：1200** **1：2400** **2：4800** **3：9600**	单字节（非易失）	无
菜单三设置参数（PASS:0089）						
IMP	选择重要通道	0X0540	读/写	**CH01** **……** **CH16**	单字节（非易失）	无
CT	通道滚动显示切换时间	0X0520	读/写	**1~10(Sec)**	单字节（非易失）	无
FilT	数字滤波系数	0X0530	读/写	**0~10**	单字节（非易失）	无
通道参数地址的编制规则如下：						
CHn	n 通道的相关参数					
EN	使用与否	0X0400+1*(n-1)	读/写	0:OFF 1:ON	单字节（非易失）	无
INTY	输入信号选择	0X0420+1*(n-1)	读/写	0:　K 1:　S 2:　J 3:　T 4:　E 5:　B 6:　R 7:WRE 8:CU50 9:C100 10:P100 11:4-20 12:0-10 13:0-20 14:1-5V 15: 10V 16:0-5V	单字节（非易失）	无
DOT	小数点位置	0X0500+1*(n-1)	读/写	**0~3(注 1)**	单字节（非易失）	无
PVL	显示下限	0X0440+2*(n-1)	读/写	**-1999~9999**	双字节（非易失）	注 3
PVH	显示上限	0X0460+2*(n-1)	读/写	**-1999~9999**	双字节（非易失）	注 3
AL	下限报警设定值	0X0480+2*(n-1)	读/写	**-1999~9999**	双字节（非易失）	注 3
AH	上限报警设定值	0X04A0+2*(n-1)	读/写	**-1999~9999**	双字节（非易失）	注 3
ALC	报警回差	0X0550+2*(n-1)	读/写	**-1999~9999**	双字节（非易失）	注 3
PSD	变送器/传感器零位修正系数	0X04C0+2*(n-1)	读/写	**-1999~9999**	双字节（非易失）	注 3
UNIT	打印单位	0X0580+1*(n-1)	读/写	**00-42(注 2)**	单字节（非易失）	无
如：**CH16**						
CH16	16 通道的相关参数					
EN	使用与否	0X0400+1*E	读/写	**0:OFF** **1:ON**	单字节（非易失）	无

INTY	输入信号选择	0X0420+1*E	读/写	0: K 1: S 2: J 3: T 4: E 5: B 6: R 7:WRE 8:CU50 9:C100 10:P100 11:4-20 12:0-10 13:0-20 14:1-5V 15: 10V 16:0-5V	单字节（非易失）	无
DOT	小数点位置	0X0500+1*E	读/写	0~3(注 1)	单字节（非易失）	无
PVL	显示下限	0X0440+2*E	读/写	-1999~9999	双字节（非易失）	注 3
PVH	显示上限	0X0460+2*E	读/写	-1999~9999	双字节（非易失）	注 3
AL	下限报警设定值	0X0480+2*E	读/写	-1999~9999	双字节（非易失）	注 3
AH	上限报警设定值	0X04A0+2*E	读/写	-1999~9999	双字节（非易失）	注 3
ALC	报警回差	0X0550+2*E	读/写	-1999~9999	双字节（非易失）	注 3
PSD	变送器/传感器零位修正系数	0X04C0+2*E	读/写	-1999~9999	双字节（非易失）	注 3
UNIT	打印单位	0X0580+1*E	读/写	00-42(注 2)	单字节（非易失）	无

报警指示 LED 和继电器状态

参数名	参数内容	地址	操作	备注	数据类型	小数点位
A01-A08	1~8 通道独立报警指示灯状态 MSB LSB A01 A02 A03 A04 A05 A06 A07 A08	0X0A00	只读	注 5	单字节（自动）	无
A09-A16	9~16 通道独立报警指示灯状态 MSB LSB A09 A10 A11 A12 A13 A14 A15 A16	0X0A01	只读	注 5	单字节（自动）	无
AH-AL-IAH-IAL（注）	4 路继电器（继电器指示灯）状态 MSB LSB XX X X X IAL IAH AL AH	0X0A02	只读	注 5	单字节（自动）	无
F01-F08	1~8 通道独立报警指示灯闪烁状态 MSB LSB F01 F02 F03 F04 F05 F06 F07 F08	0X0A03	只读	注 5	单字节（自动）	无
F09-F16	9~16 通道独立报警指示灯闪烁状态 MSB LSB F01 F02 F03 F04 F05 F06 F07 F08	0X0A04	只读	注 5	单字节（自动）	无
AH01	通道 1 的上限报警状态	0X0A10	只读	0：无报警 1：有报警 2：报警已解除	单字节（自动）	无

本参数地址的编制规则如下：						
AHn	通道 n 的上限报警状态	0X0A10+(n-1)	只读	0：无报警 1：有报警 2：报警已解除	单字节（自动）	无
AH16	通道 16 的上限报警状态	0X0A1F	只读	0：无报警 1：有报警 2：报警已解除	单字节（自动）	无
AL01	通道 1 的下限报警状态	0X0A20	只读	0：无报警 1：有报警 2：报警已解除	单字节（自动）	无
本参数地址的编制规则如下：						
ALn	通道 n 的下限报警状态	0X0A20+(n-1)	只读	0：无报警 1：有报警 2：报警已解除	单字节（自动）	无
AL16	通道 16 的下限报警状态	0X0A2F	只读	0：无报警 1：有报警 2：报警已解除	单字节（自动）	无

注：AH、AL：通用的上限和下限报警器；IAH、IAL：重要通道的上限和下限报警器

通道测量值						
参数名	参数内容	地址	操作	备注	数据类型	小数点位
PV-01 …… PV-16	对应通道的测量值	0X0800+通道号-1	只读		浮点（自动）	
Et	环境温度（冷补）	0X0810	只读		浮点（自动）	

产品信息

		地址	操作	内容	数据类型	小数点位
XJY-160		0X0780	只读	0X58（X）	常数	无
		0X0781	只读	0X4A（J）	常数	无
		0X0782	只读	0X59（Y）	常数	无
		0X0783	只读	0X31（1）	常数	无
		0X0784	只读	0X36（6）	常数	无
		0X0785	只读	0X30（0）	常数	无
版本信息	版本： A	0X0786	只读	0X41（A）	常数	无

★ 注 1：小数点位 DOT

—0、 表示小数位消隐，如：0000（真实值）

—1、 表示小数点在第 1 位，如：000.0（真实值）

—2、 表示小数点在第 2 位，如：00.00（真实值）

—3、 表示小数点在第 3 位，如：0.000（真实值）

注意：当本通道输入信号类型为 K ， S ， J ， T ， E ， B ， R ， WRE ， CU50 ， Cu100 ， P100 时 DOT 是默认的无须处理，除此之外的信号类型上位机应根据 DOT 进行小数点的处理！否则上位机与下位机的显示有可能不同（DOT 不为零时）!

★ 注 2：UNIT 的数值所对应的具体单位如下：

0	无	11	T	22	Kg/H	33	mm
1	g/cm^3	12	L	23	m/m	34	KN
2	Pa	13	M^3	24	T/m	35	V
3	KPa	14	Kg	25	L/m	36	A
4	MPa	15	Hz	26	m^3/m	37	mV
5	mmHg	16	KHz	27	Kg/m	38	mA
6	mmH2O	17	rpm	28	m/s	39	W
7	bar	18	m/h	29	T/s	40	KW
8	℃	19	T/h	30	L/s	41	VA
9	%	20	L/h	31	m^3/s	42	KVA
10	m	21	m^3/h	32	Kg/s	43	

★ 注 3：本参数受本通道内的 DOT 控制。

★ 注 4：关于浮点数的格式详述如下：

本仪表的浮点数为符合 IEEE-754 的 32Bit 浮点数，其具体格式如下图所示：

Address	+0	+1	+2	+3
Contents	SEEE EEEE	EMMM MMMM	MMMM MMMM	MMMM MMMM

低位　　高位

S: 符号位 1 时为负，0 时为正。

E: 指数，向上偏置了 127。

M: 24 位尾数（存储在 23 位空间）。

注意：浮点数在发送时是先发 S 符号位。

数转十进制的计算方法：

则按照规定，浮点数的值用十进制表示为：

= (-1)^s * (1 + x) * 2^(e - 127)

对于 49E48E68 来说，

1、其第 31 bit 为 0，即 s = 0

2、第 30～23 bit 依次为 100 1001 1，读成十进制就是 147，即 e = 147。

3、第 22～0 bit 依次为 110 0100 1000 1110 0110 1000，也就是二进制的纯小数 0.110 0100 1000 1110 0110 1000，其十进制形式为(0.110 0100 1000 1110 0110 1000 * 2^23) / (2^23) = (0x49E48E68 & 0x007FFFFF) / (2^23) = (0x648E68) / (2^23) = 0.78559589385986328125，即 x =

0.78559589385986328125。

这样，该浮点数的十进制表示

= (-1)^s * (1 + x) * 2^(e - 127)

= (-1)^0 * (1+ 0.78559589385986328125) * 2^(147-127)

= 1872333

http://download1.csdn.net/down3/20070618

例如读第 1 通道的 PV（浮点数）值：

主 机 发 送							
从机地址	功能码	目标寄存器的地址高位		目标寄存器的地址低位		CRC 低位	CRC 高位
01	68	08		00		87	C4
从 机 应 答							
从机地址	功能码	目标寄存器数据高位	目标寄存器数据次高位	目标寄存器数据次低位	目标寄存器数据低位	CRC 低位	CRC 高位
01	68	40	00	00	00	75	C3

注意： 读其它浮点寄存器中数据均按此格式，寄存器的地址不能超出菜单范围，超出范围将导致读取失败。

★ 注 5：对应位置 1 时使能。

三．标准 MoDBUS 读写命令说明和示例

（一）数据读出

数据读出命令为 03，不同的数据格式安装地址自动由仪表进行判断。地址为连续地址时可以连读。

（1）读出单字节数据： 从 1 号仪表中读出通道 1 的报警状态

主 机 发 送					
从机地址	功能码	目标寄存器的地址	要读出寄存器的个数	CRC 低位	CRC 高位
0X01	0X03	0X0A ， 0X10	0X00 0X01	0X86	0X17
从 机 应 答					
从机地址	功能码	字节数	返回数据	CRC 低位	CRC 高位
0X01	0X03	0X02	0X00，0X01	0X79	0X84

从 1 号仪表中读出 1-8 的通道报警状态

主 机 发 送					
从机地址	功能码	目标寄存器的地址	要读出寄存器的个数	CRC 低位	CRC 高位
0X01	0X03	0X04 ， 0X80	0X00 0X08	0X46	0X11
从 机 应 答					
从机地址	功能码	字节数	返回数据	CRC 低位	CRC 高位
0X01	0X03	0X10	0X00，0x01，0x00，0X01，0X00，0x01，0x00，0X01，0X00，0x01，0x00，0X01，0X00，0x01，0x00，0X00	0X47	0XDE

（2）读出双字节数据： 从 1 号仪表中读出第一通道，第二通道的 AL 值

主 机 发 送					
从机地址	功能码	目标寄存器的地址	要读出寄存器的个数	CRC 低位	CRC 高位
0X01	0X03	0X0A ， 0X10	0X00 0X02	0XC6	0X16
从 机 应 答					
从机地址	功能码	字节数	返回数据	CRC 低位	CRC 高位
0X01	0X03	0X04	0X00，0X50 ， 0x00，0x80	0X89	0X84

(3) 读出浮点数: 从 1 号仪表中读出第一第二通道的值

主 机 发 送					
从机地址	功能码	目标寄存器的地址	要读出寄存器的个数	CRC 低位	CRC 高位
0X01	0X03	0X08，0X00	0X00 0X04	0X46	0X69
从 机 应 答					
从机地址	功能码	字节数	返回数据	CRC 低位	CRC 高位
0X01	0X03	0X08	0X42，0XD6 ，0x51，0xCF，0X42，0XD8 ，0x51，0xCF	0X17	0XE5

注意：（1） 在地址连续的时候，可以一次读取多个数据.双字节数据地址间隔 1，也默认为是连续的。这点在处理的时候需要注意。

（2）MODBUS 协议里面很少有单字节数据，为了和 MODBUS 协议兼容，在高字节自动填充 00.

（3） 每个单字节数据，双字节数据都是各自占用一个寄存器地址，每个浮点数占据两个寄存器地址。在读出寄存器个数上要注意区分。数据返回时字节数是寄存器个数的 2 倍。

（二）数据写入

（1）单字节数据写入 修改通道 1 的输入信号类型

主 机 发 送					
从机地址	功能码	目标寄存器的地址	要写的数据	CRC 低位	CRC 高位
0X01	0X06	0X04 ， 0X20	0X00 0X02	0X08	0Xf1
从 机 应 答					
从机地址	功能码	目标寄存器的地址	要写的数据	CRC 低位	CRC 高位
0X01	0X06	0X04 ， 0X20	0X00 0X02	0X08	0Xf1

(2)双字节写入 修改通道 1 的 PVH 值

主 机 发 送					
从机地址	功能码	目标寄存器的地址	要写的数据	CRC 低位	CRC 高位
0X01	0X06	0X04 ， 0X60	0X13 0X88	0X85	0X 把
从 机 应 答					
从机地址	功能码	目标寄存器的地址	要写的数据	CRC 低位	CRC 高位
0X01	0X06	0X04 ， 0X60	0X13 0X88	0X85	0X 把

注意：（1）单字节数据，高字节总是为 00.

（2）数据在写入的时候，一律忽略小数点。如 500.0；5.00；5.000 实际写入数据都是 0x1388。仪表会自动判断位数。

四．老版本支持的自定义命令读写说明

（一）读单字节寄存器中的数据：

功能代码：0X64

示例格式如下：（读地址为 0X0400 单字节寄存器中的数据）

主 机 发 送					
从机地址	功能码	目标寄存器的地址高位	目标寄存器的地址低位	CRC 低位	CRC 高位
0X01	0X64	0X04	0X00	0X42	0XC7
从 机 应 答					
从机地址	功能码	目标寄存器数据		CRC 低位	CRC 高位
0X01	0X64	0X01		0XCB	0X00

★ 读其它单字节寄存器中数据均按此格式，寄存器的地址不能超出菜单范围，超出范围将导致读取失败。

（二）. 读双字节寄存器中的数据：

功能代码：0X66

示例格式如下：（读地址为 0X0550 双字节寄存器中的数据）

主 机 发 送					
从机地址	功能码	目标寄存器的地址高位	目标寄存器的地址低位	CRC 低位	CRC 高位
0X01	0X66	0X05	0X50	0XE2	0XAB
从 机 应 答					
从机地址	功能码	目标寄存器数据高位	目标寄存器数据低位	CRC 低位	CRC 高位
0X01	0X66	0X00	0X03	0XA1	0XC6

★ 读其它双字节寄存器中数据均按此格式，寄存器的地址不能超出菜单范围，超出范围将导致读取失败。

（三）. 读浮点寄存器中的数据：

功能代码：0X68

示例格式如下：（读地址为 0X0800 浮点寄存器中的数据）

主 机 发 送							
从机地址	功能码	目标寄存器的地址高位		目标寄存器的地址低位		CRC 低位	CRC 高位
0X01	0X68	0X08		0X00		0X87	0XC4
从 机 应 答							
从机地址	功能码	目标寄存器数据高位	目标寄存器数据次高位	目标寄存器数据次低位	目标寄存器数据低位	CRC 低位	CRC 高位
0X01	0X68	0X40	0X00	0X00	0X00	0X75	0XC3

★ 读其它浮点寄存器中数据均按此格式，寄存器的地址不能超出菜单范围，超出范围将导致读取失败。

（四）. 写单字节寄存器中的数据：

功能代码：0X65

示例格式如下：（写地址为 0X0400 单字节寄存器中的数据）

主 机 发 送						
从机地址	功能码	寄存器的地址高位	寄存器的地址高位	预置数据	CRC 低位	CRC 高位
0X 01	0X65	0X04	0X00	0X00	0X46	0XCD
从 机 应 答						
从机地址	功能码	寄存器的高位地址	寄存器的高位地址	预置数据	CRC 低位	CRC 高位
0X 01	0X65	0X04	0X00	0X00	0X46	0XCD

★ 写其它单字节寄存器中数据均按此格式，寄存器的地址不能超出菜单范围，超出范围将导致写入失败。

（五）. 写双字节寄存器中的数据：

功能代码：0X67

示例格式如下：（写地址为 0X0550 双字节寄存器中的数据）

主 机 发 送

从机地址	功能码	寄存器的地址高位	寄存器的地址高位	预置数据高位	预置数据低位	CRC 低位	CRC 高位
0X01	0X67	0X05	0X50	0X00	0X06	0XB4	0XDD
从　机　应　答							
从机地址	功能码	寄存器的高位地址	寄存器的高位地址	预置数据高位	预置数据低位	CRC 低位	CRC 高位
0X01	0X67	0X05	0X50	0X00	0X06	0XB4	0XDD

★　写其它双字节寄存器中数据均按此格式，寄存器的地址不能超出菜单范围，超出范围将导致写入失败。

附1：**关于错误校验码（CRC校验）：**

主机和从机应用校验码来进行接收信息是否正确的判别。由于电子噪声或一些其它干扰，信息在传输过程中有可能会发生错误，循环冗余校验码（CRC）可以检验主机或从机在通讯数据传送过程中的信息是否有误，错误的数据可以放弃（无论是发送还是接收），这样增加了系统的安全和效率。

MODBUS通讯协议的CRC（冗余循环校验码）包含2个字节，即16位二进制数。CRC码由发送设备计算，放置于所发送信息帧的尾部。接收信息的设备再重新计算所接收信息（除CRC之外的部分）的CRC，比较计算得到的CRC是否与接收到的CRC相符，如果两者不相符，则认为数据出错。

CRC码的计算方法：

1、预置1个16位的寄存器为十六进制FFFF（即全为1）；称此寄存器为CRC寄存器；

2、把第一个8位二进制数据（既通讯信息帧的第一个字节）与16位的CRC寄存器的低8位相异或，把结果放于CRC寄存器；

3、把CRC寄存器的内容右移一位（朝低位）用0填补最高位 ，并检查右移后的移出位；

4、如果移出位为0：重复第3步（再次右移一位）； 如果移出位为1：CRC寄存器与多项式A001（1010 0000 0000 0001）进行异或；

5、重复步骤3和4，直到右移8次，这样整个8位数据全部进行了处理；

6、重复步骤2到步骤5，进行通讯信息帧下一个字节的处理；

7、将该通讯信息帧所有字节按上述步骤计算完成后，得到的 16位CRC寄存器的高、低字节进行交换；

8、最后得到的CRC寄存器内容即为：CRC码。

附2：**上位机16bit Modbus CRC编程参考**

```
/******************直接计算 CRC*****************************/
//=============================================================================
unsigned int CRC16(uchar *arr_buff,uchar len)
{ uint crc=0xFFFF;
   uchar i,j,Data;
   for(j=0;j<len;j++)
   { crc=crc ^ *arr_buff++;
      for(i=0;i<8;i++)
      { if((crc&0x0001)>0){crc=crc>>1;crc=crc^0xa001;}
           else crc=crc>>1;
```

```
        }
    }
    return(crc);
}
//==========================================================================

/**************查表法计算 CRC*********************/
//==========================================================================
unsigned int crc_ta[256]=
{ 0x0000,0xC0C1,0xC181,0x0140,0xC301,0x03C0,0x0280,0xC241,
  0xC601,0x06C0,0x0780,0xC741,0x0500,0xC5C1,0xC481,0x0440,
  0xCC01,0x0CC0,0x0D80,0xCD41,0x0F00,0xCFC1,0xCE81,0x0E40,
  0x0A00,0xCAC1,0xCB81,0x0B40,0xC901,0x09C0,0x0880,0xC841,
  0xD801,0x18C0,0x1980,0xD941,0x1B00,0xDBC1,0xDA81,0x1A40,
  0x1E00,0xDEC1,0xDF81,0x1F40,0xDD01,0x1DC0,0x1C80,0xDC41,
  0x1400,0xD4C1,0xD581,0x1540,0xD701,0x17C0,0x1680,0xD641,
  0xD201,0x12C0,0x1380,0xD341,0x1100,0xD1C1,0xD081,0x1040,
  0xF001,0x30C0,0x3180,0xF141,0x3300,0xF3C1,0xF281,0x3240,
  0x3600,0xF6C1,0xF781,0x3740,0xF501,0x35C0,0x3480,0xF441,
  0x3C00,0xFCC1,0xFD81,0x3D40,0xFF01,0x3FC0,0x3E80,0xFE41,
  0xFA01,0x3AC0,0x3B80,0xFB41,0x3900,0xF9C1,0xF881,0x3840,
  0x2800,0xE8C1,0xE981,0x2940,0xEB01,0x2BC0,0x2A80,0xEA41,
  0xEE01,0x2EC0,0x2F80,0xEF41,0x2D00,0xEDC1,0xEC81,0x2C40,
  0xE401,0x24C0,0x2580,0xE541,0x2700,0xE7C1,0xE681,0x2640,
  0x2200,0xE2C1,0xE381,0x2340,0xE101,0x21C0,0x2080,0xE041,
  0xA001,0x60C0,0x6180,0xA141,0x6300,0xA3C1,0xA281,0x6240,
  0x6600,0xA6C1,0xA781,0x6740,0xA501,0x65C0,0x6480,0xA441,
  0x6C00,0xACC1,0xAD81,0x6D40,0xAF01,0x6FC0,0x6E80,0xAE41,
  0xAA01,0x6AC0,0x6B80,0xAB41,0x6900,0xA9C1,0xA881,0x6840,
  0x7800,0xB8C1,0xB981,0x7940,0xBB01,0x7BC0,0x7A80,0xBA41,
  0xBE01,0x7EC0,0x7F80,0xBF41,0x7D00,0xBDC1,0xBC81,0x7C40,
  0xB401,0x74C0,0x7580,0xB541,0x7700,0xB7C1,0xB681,0x7640,
  0x7200,0xB2C1,0xB381,0x7340,0xB101,0x71C0,0x7080,0xB041,
  0x5000,0x90C1,0x9181,0x5140,0x9301,0x53C0,0x5280,0x9241,
  0x9601,0x56C0,0x5780,0x9741,0x5500,0x95C1,0x9481,0x5440,
  0x9C01,0x5CC0,0x5D80,0x9D41,0x5F00,0x9FC1,0x9E81,0x5E40,
  0x5A00,0x9AC1,0x9B81,0x5B40,0x9901,0x59C0,0x5880,0x9841,
  0x8801,0x48C0,0x4980,0x8941,0x4B00,0x8BC1,0x8A81,0x4A40,
  0x4E00,0x8EC1,0x8F81,0x4F40,0x8D01,0x4DC0,0x4C80,0x8C41,
  0x4400,0x84C1,0x8581,0x4540,0x8701,0x47C0,0x4680,0x8641,
  0x8201,0x42C0,0x4380,0x8341,0x4100,0x81C1,0x8081,0x4040
 };

unsigned int swap(unsigned int x)
 { unsigned char H,L;
   unsigned int cx=0;
   H=x/256;L=x%256;
   cx=L;cx=cx<<8;
```

```
    cx=cx+H;
    return(cx);
  }


unsigned int car_crc(unsigned char *pty,unsigned char len)
{ unsigned int crc;
   unsigned char da;
    crc=0xffff;
    while(len--!=0)
      { da=(*pty++^crc)&0x0ff;
        crc=swap(crc)&0xff;
        crc=crc ^ crc_ta[da];
       }
    return(crc);
   }

void main ()
  { unsigned char buf[10];
    unsigned int C_crc;
    buf[0]=0x05;
    buf[1]=0x03;
    buf[2]=0x01;
    buf[3]=0x64;
    buf[4]=0x00;
    buf[5]=0x02;
    while(1)
      { C_crc=car_crc(buf,6); }
  }
//=============================================================================
```

附 2、关于通讯串口读写的编程范例请参考光盘中的《串口测试程序范例源代码》 。