PMC-D726X装置
Modbus通信协议
（V1.0）

深圳市中电电力技术股份有限公司
Ceiec Electric Technology Inc.

目 录

1 简介

本规约详细地描述了PMC-D726X产品在MODBUS通信模式下的输入和输出命令、信息和数据，为使用PMC-D726X产品的Modbus通信规约提供参考。

1.1 通信协议的目的

通信协议的作用是使信息和数据在上位机主站和PMC从站之间有效地传递，它包括：

（1）允许主站访问从站装置的测量值信息，开关量状态，测控信息；

（2）允许访问和设置从站装置的参数；

（3）允许主站对从站装置进行遥控操作；

（4）允许主站访问和设置从站装置的相关装置信息；

（5）允许主站访问从站装置的SOE。

1.2 通信协议的版本

该通信协议适用于PMC-D726X产品，以后若有改动会特别说明。

2 MODBUS串行通信协议详细说明

2.1 协议基本规则

以下规则确定在RS485或者RS232C回路控制器和其他RS485串行通信回路中设备的通信规则：

（1）所有RS485回路通信应遵照主/从方式。在这种方式下，信息和数据在单个主站和最多128个从站监控设备之间传递；

（2）主站将初始化和控制所有在RS485通信回路上传递的信息；

（3）只有主站向从站发送通信命令后，从站才能进行数据通信；

（4）RS485环路以数据“打包”方式进行通信，一个数据包就是一个简单的字符串(每个字符8位)，一个数据包中最多可含255个字节。组成这个数据包的字节构成标准异步串行数据，并按预先设置的传送模式进行传递。串行数据流由类似于RS232C中使用的设备产生；

（5）主站发送数据包称为请求，从站发送数据包称为响应；

（6）任一时刻仅一个从站响应主站的请求。

2.2 传送模式

MODBUS协议可以采用ASCII或者RTU模式传送数据，PMC装置支持RTU模式。

通信波特率： 1200bps、2400bps、4800bps、9600bps、19200bps，默认设置9600bps；

校验方式：支持偶校验、奇校验、无校验，默认设置偶校验；数据按照1位启动位、8位数据位、1位校验位、1位或2位停止位传送。

2.3 数据包结构描述

每个MODBUS 数据包都由以下几个部分组成：

（1）地址域

（2）功能码域

（3）数据域

（4）校验域

2.3.1 地址域

MODBUS的从站地址域包含数据包传送的从站地址，长度为一个字节，有效的从站地址范围为1~247。地址0为广播地址，仅应用于广播对时。如果从站接收到数据包中的地址信息与本地地址相同，则执行数据包中所包含的命令。从站所响应的数据包中，该地址域为从站本地地址。

2.3.2 功能码域

MODBUS数据包中功能域长度为一个字节，用以通知从站应当执行何操作。从站响应数据包中应当包含主站所请求操作的相同功能域字节。有关PMC装置的功能码参照下表：

功能码	含义	功能
0x03	读取寄存器	实时数据、电能、状态量、统计量、SOE、定时记录、波形记录等数据；
0x10	设置一个或多个寄存器	参数设置、对时、清除等操作；
0x05	设置线圈	遥控操作

2.3.3 数据域

MODBUS 数据域长度不定，依据其具体功能而定。MODBUS数据域采用“BIG INDIAN”模式，即高位字节在前低位字节在后。例如：

1个16位寄存器包含数值为0x12AB 寄存器数值发送顺序为：

高位字节= 0x12

低位字节= 0xAB

本规约上传的数据类型有char(字符型)、uint16(无符号短整形16位)、int16(有符号短整形16位)、uint32(无符号长整形32位)、int32(有符号长整形32位)、float(浮点数32位)、状态字，具体的上传数据格式说明如下：

uint16和int16：16位的短整形由两个字节组成byte0（bit0~bit7）、byte1(bit8~bit15),寄存器数据发送顺序为:byte1、byte0。

uint32和int32:32位的长整形由四个字节组成byte0(bit0~bit7)、byte1(bit8~bit15)、byte2(bit16~bit23)、byte3(bit24~bit31),寄存器数据发送顺序为：byte3、byte2、byte1、Byte0。

float：浮点数上传的是IEEE754 表示格式4字节，占两个寄存器。例如：2.66为0x71、0x3D 0x2A、0x40，用两个寄存器上传这四字节的数据,上传的顺序为0x40、0x2A、0x3D、0x71。

int64:64位的长整型由八个字节组成byte0(bit0~bit7)、byte1(bit8~bit15)、byte2(bit16~bit23)、byte3(bit24~bit31)、byte4(bit32~bit39)、byte5(bit40~bit47)、byte6(bit48~bit55)、byte7(bit56~bit63),寄存器数据发送顺序为：byte7、byte6、byte5、byte4、byte3、byte2、byte1、byte0。

2.3.4 校验域

MODBUS RTU模式采用16位CRC校验。发送设备应当对数据包中的每一个数据都进行CRC16计算，最后结果存放入检验域中。接收设备也应当对数据包中的每一个数据(除校验域以外)进行CRC16计算，将结果域校验域进行比较。只有相同的数据包才可以被接受。具体的CRC校验算法参照附录A。

2.4 网络时间考虑

在RS485 网络上传送数据包需要遵循以下有关时间的规定：

（1）9600bps下，主站请求数据包结束到从站响应数据包开始之间的时间最大为250毫秒，典型值为60毫秒；

（2）从站响应数据包结束后，主站必须经过至少3.5个字符的传输延迟后才能发送下一个数据请求；

（3）主站发送广播命令后，必须经过至少100ms的延迟才能发送下一个数据请求。

2.5 异常响应

如果主站发送了一个非法的数据包给PMC装置或者是主站请求一个无效的数据寄存器时，异常的数据响应就会产生（如果接收到的数据CRC校验错误，则直接丢弃）。 这个异常响应数据包括从站地址、功能码、故障码和校验域。当功能码的高比特位置为1时，说明此数据包为异常响应。下表说明故障码的含义：

故障码名称	功能码	说 明
0x01(非法功能码)	0x80+原功能码	表示从站接收到不支持的功能码(除上面列举的功能码)。
0x02(非法数据地址)	0x80+原功能码	说明 PMC 装置接收到无效的数据地址或者是请求寄存器不在有效的寄存器范围内(例如对只读寄存器进行写操作，再比如请求寄存器偏移+长度无效)
0x03(非法数据值)	0x80+原功能码	读写数据时寄存器数量超出允许范围或字节数!=寄存器数*2; 写入的数据超出范围

		必须连续写的寄存器写入不完整
0x04(操作寄存器失败)	0x80+原功能码	遥控操作失败，或异常码 02、03 规定之外的情况。

2.6 广播命令

广播命令的地址域为 0x00，仅在使用 0x10 功能码时有效。主站发送广播命令时，从站只接收数据包，不返回响应数据包，以防止网络内所有从站同时响应，造成网络堵塞。

3 MODBUS串行通信数据帧

标准的MODBUS协议仅支持16位数据模式，可传输最大为65535的数据，本装置支持浮点数传送。

本装置 MODBUS支持多个功能码，一个寄存器表示16位数据，对于超过一个寄存器的数据，则分成多个寄存器读写。比如，一个32位数据，高字占一个寄存器，低字占一个寄存器。64位数据则用4个寄存器表示。16位数据模式中，数据都是使用两个8位寄存器表示。

3.1 设置继电器输出状态（0x05）

设定系统参数区中的“遥控预置”参数，如果该参数设定为“投入”，则通过modbus协议设置装置的继电器步骤是：第一步，发送预置命令 ，第二步，在指定15s的时间内，发送执行命令；如果该参数设定为“退出”，则直接通过执行寄存器地址执行命令，无需预置命令。“遥控预置”参数默认为“投入”。数据域的内容指定继电器的动作，0xFF00有效，发送其他的数据不会影响继电器的状态。

从站响应数据中，对正确设置的数据，返回请求帧。如果设置值无效，响应异常数据值。

例如：设置装置的遥合继电器(预置寄存器地址为9100，执行寄存器地址为9101)合闸，则通信请求和响应命令如下：

预置命令：

请求格式（主站→PMC 装置）		响应格式（PMC 装置→主站）	
从站地址	0x11	从站地址	0x11
功能码	0x05	功能码	0x05
预置地址高	0X23	预置地址高	0X23
预置地址低	0X8C	预置地址低	0X8C
设置数据高	0XFF	设置数据高	0XFF
设置数据低	0x00	设置数据低	0x00
CRC 校验码低	44	CRC 校验码低	44
CRC 校验码高	C5	CRC 校验码高	C5

执行命令：

请求格式（主站→PMC 装置）		响应格式（PMC 装置→主站）	
从站地址	0x11	从站地址	0x11
功能码	0x05	功能码	0x05
执行地址高	0X23	执行地址高	0X23
执行地址低	0X8D	执行地址低	0X8D
设置数据高	0XFF	设置数据高	0XFF
设置数据低	0x00	设置数据低	0x00
CRC 校验码低	15	CRC 校验码低	15
CRC 校验码高	05	CRC 校验码高	05

3.2 读寄存器（0x03）

由主站机发送的数据包请求，本装置响应所有有效的寄存器(在起始寄存器和终止寄存器之间)。命令格式如下：

请求格式(主站→PMC 装置)		响应格式(PMC 装置→主站)	
从站地址	1 字节	从站地址	1 字节
功能码	1 字节	功能码	1 字节
寄存器起始地址高位	1 字节	字节数 n	1 字节
寄存器起始地址低位	1 字节	Data1 高位	1 字节
寄存器数量高位	1 字节	Data1 低位	1 字节
寄存器数量低位	1 字节	……	
		Datan/2 高位	1 字节
		Datan/2 低位	1 字节
CRC 校验码低位	1 字节	CRC 低位	1 字节
CRC 校验码高位	1 字节	CRC 高位	1 字节

备注：一个数据包最少读 1 个寄存器，最多可读 125 个寄存器。

3.3 写寄存器（0x10）

该命令允许主站设置本装置工作参数，命令格式如下：

请求格式(主站→PMC 装置)	响应格式(PMC 装置→主站)

从站地址	1 字节	从站地址	1 字节
功能码	1 字节	功能码	1 字节
寄存器起始地址高位	1 字节	寄存器起始地址高位	1 字节
寄存器起始地址低位	1 字节	寄存器起始地址低位	1 字节
寄存器数量高位	1 字节	寄存器数量高位	1 字节
寄存器数量低位	1 字节	寄存器数量低位	1 字节
字节数(n)	1 字节	CRC 校验码低位	1 字节
Data1 高位	1 字节	CRC 校验码高位	1 字节
Data1 低位	1 字节		
……			
Data(n/2) 高位	1 字节		
Data(n/2) 低位	1 字节		
CRC 校验码低位	1 字节		
CRC 校验码高位	1 字节		

备注：一个数据包最少写 1 个寄存器，最多可写 123 个寄存器。

4 装置寄存器说明

本规约规定，请求本装置中一个地址为 xx 寄存器的数据时，主站实际读取寄存器地址 xx 的数据。例如：请求本装置中寄存器地址为 10 的数据，主站传送实际寄存器地址为 10 的数据。详细的寄存器说明如下所示。

4.1 实时测量数据寄存器

更新周期：1s

地址	类型	描述	数据格式	单位/系数	备注	适用装置		
						D726I	D726V	D726M
0	RO	A相电压	UINT32	×100，V	【注 1】 【注 2】		√	√
2	RO	B相电压	UINT32	×100，V			√	√
4	RO	C相电压	UINT32	×100，V			√	√
6	RO	平均相电压	UINT32	×100，V			√	√
8	RO	AB线电压	UINT32	×100，V			√	√
10	RO	BC线电压	UINT32	×100，V			√	√

12	RO	CA线电压	UINT32	×100，V			√	√
14	RO	平均线电压	UINT32	×100，V			√	√
16	RO	A相电流	UINT32	×1000，A		√		√
18	RO	B相电流	UINT32	×1000，A		√		√
20	RO	C相电流	UINT32	×1000，A		√		√
22	RO	平均相电流	UINT32	×1000，A		√		√
24	RO	A相有功功率	INT32	×1000，kW				√
26	RO	B相有功功率	INT32	×1000，kW				√
28	RO	C相有功功率	INT32	×1000，kW				√
30	RO	三相有功功率	INT32	×1000，kW				√
32	RO	A相无功功率	INT32	×1000，kvar				√
34	RO	B相无功功率	INT32	×1000，kvar				√
36	RO	C相无功功率	INT32	×1000，kvar				√
38	RO	三相无功功率	INT32	×1000，kvar				√
40	RO	A相视在功率	INT32	×1000，kVA				√
42	RO	B相视在功率	INT32	×1000，kVA				√
44	RO	C相视在功率	INT32	×1000，kVA				√
46	RO	三相视在功率	INT32	×1000，kVA				√
48	RO	A相功率因数	INT16	×1000				√
49	RO	B相功率因数	INT16	×1000				√
50	RO	C相功率	INT16	×1000				√

		因数						
51	RO	总功率因数	INT16	×1000				√
52	RO	频率	INT16	×100 Hz				√
…		保留				√	√	√
96	RO	DO的状态	INT16		0：开 1：合	√	√	√
97	RO	DI的状态	INT16		【注 3】 【注 5】	√	√	√
98	RO	SOE总指针	UINT32		【注 4】	√	√	√

【注1】：×100，V表示通讯上传的数据比实际数据放大了100倍，数据单位是V。
比如读电压寄存器的数据为000055F3H，55F3H = 22003,表示实际电压是220.03V。
【注2】：当接线方式为角型接线时，各相相电压/有功功率/无功功率/视在功率/功率因数无意义，为保留寄存器。对于726I选型，电压和功率寄存器仍然可读，但读出值为无效值，对于726V选型，电流和功率寄存器仍然可读，但读出值为无效值。
【注3】：DI状态寄存器：B0～B1分别表示DI1～DI2的状态。1代表闭合，0代表打开。
【注 4】：每产生一条事件，SOE 指针加 1，范围 0～0xFFFFFFFF。当进行 SOE 清除时，指针先清零，由于又产生清除的 SOE 事件，因此指针变为 1。
【注 5】：对于非 DO 配置的选型，DO 状态寄存器可读，但读出值无效，对与非 DI 配置的选型，DI 状态寄存器可读，但是读出值无效。

4.2 计量数据寄存器

更新周期：1s
电能设置寄存器对于 726I/726V/726M 都可读可写，但 726I/726V 读出为无效值，写入任何值也无效。

地址		类型	描述	数据格式	单位/系数	备注
1000	100	RW	正向有功电能 千瓦时	UINT32	0.1kWh	【注 1】
1002	102	RW	反向有功电能 千瓦时	UINT32	0.1kWh	【注 1】
1004	104	RW	保留			
1006	106	RW	正向无功电能 千乏时	UINT32	0.1kvarh	【注 1】
1008	108	RW	反向无功电能 千乏时	UINT32	0.1kvarh	【注 1】
1010	110	RW	保留			
1012	112	RW	视在电能 千伏安时	UINT32	0.1kVAh	【注 1】

【注1】正、反相电能寄存器的范围为0～999,999,999，溢出后自动翻转为0。

4.3 装置参数寄存器

地址	类型	描述	数据格式	单位/范围	备注	适用装置		
						D726I	D726V	D726M
6000	RW	保留				√	√	√
……	……	保留				√	√	√
6010	RW	CT 变比	INT16	5A 配置：1～6000 1A 配置：1~30000	【注 1】	√		√
6011	RW	接线方式	INT16	0=星型 1=角型 2=演示	默认值 0，星型		√	√
6012	RW	PT 变比	INT16	1～5000	【注 1】		√	√
6013	RW	ID 地址	INT16	1~247	默认值：100	√	√	√
6014	RW	波特率	INT16	0~4	0:1200 1:2400 2:4800 3:9600 4:19200 默认值 3，对应 9600	√	√	√
6015	RW	奇偶校验	INT16	0~5	0：8N2 1：8O1 2：8E1 3：8N1: 4：8O2: 5：8E2； 默认值：2，对应：8E1	√	√	√
6016	RW	选择与AO输出成比例的电量	INT16		【注2】 【注3】	√	√	√
6017	RW	AO 输出为0时的相关参数值	INT32	-999,999~999,999	【注3】	√	√	√

6019	RW	AO 输出为满量程时的相关参数值	INT32	-999,999~999,999	【注3】	√	√	√
6021	RW	AO 输出选型	INT16	0~1	0: 4~20mA, 1: 0~20mA 默认值：0 【注3】	√	√	√
6022	RW	A 相电流CT方向设置	INT16	0~1	0：正方向 1：负方向 默认值：0	√		√
6023	RW	B相电流CT方向设置	INT16	0~1	0：正方向 1：负方向 默认值：0	√		√
6024	RW	C相电流CT方向设置	INT16	0~1	0：正方向 1：负方向 默认值：0	√		√
6025	RW	功率因数符号选择位	INT16	0~2	B1B0: 00: IEC 01:IEEE 10:-IEEE 默认值：0			√
6026	RW	计算视在功率方法选择	INT16	0~1	0：V（矢量法） 1：S（标量法） 默认值：0			√
6027	RW	电能脉冲	INT16	0~1	0：不使用电能脉冲 1：使用电能脉冲 默认值：1			√
6028	RW	电能脉冲常数	INT16	0~4	0：1000 1：3200 2：5000 3：6400 4：12800			√

					默认值：0 【注 4】			
6029	RW	DO遥控预置	INT16	0~1	1：DO遥控预置投入 0：DO遥控预置退出 默认值：1 【注5】			
6030	WO	清SOE事件缓冲区	INT16	0xFF00	0xFF00 进行清除，清除后，总指针先变0，然后产生清除SOE 事件，总指针+1。	√	√	√
6031	WO	清电度	INT16	0xFF00	0xFF00 进行清除			√

【注 1】：PT 与 CT 的乘积限制： PT×CT×额定电流≤2000000。

【注 2】：选择与 AO 输出成比例的变量：

Value	变量		
	D726I	D726V	D726M
0	IA	VAB	IA
1	IB	VBC	IB
2	IC	VCA	IC
3	I avg	VLL avg	I avg
4			VAB
5			VBC
6			VCA
7			VLL avg
8			PA
9			PB
10			PC
11			ΣP
12			ΣPF
13			FREQ

其中，频率放大 100 倍写入，功率因数放大 1000 倍写入，功率单位 kw。其余参数大小与实际值一致。

【注 3】：对于无 AO 的选型，此寄存器可读可写，但读出值无意义，写任何值均不生效。

【注 4】：电压电流规格对应的脉冲常数：

电压规格 V	电流规格 A	可选脉冲常数(imp/kWh)
3×57.74/100，3×100	1	1000/3200/5000/6400/12800

	5	1000/3200/5000/6400/12800
3×220/380，3×380	1	1000/3200/5000/6400/12800
	5	1000/3200/5000

【注 5】：对于无 DO 的选型，此寄存器可读可写，但读出值无意义，写任何值均不生效。

4.4 时间寄存器

地址		类型	描述	数据格式	范围/备注
9000	60000	RW	Hi：年（-2000） Lo：月	uint16	年：0～37（-2000） 月：1～12
9001	60001	RW	Hi：日 Lo：时	uint16	日：1～31 时：0～23
9002	60002	RW	Hi：分 Lo：秒	uint16	分：0～59 秒：0～59
9003	60003	RW	毫秒	uint16	毫秒：0～999
9004~ 9005	60004~ 60005	RW	UNIX 秒（UTC 时间）	uint32	0x386D4380～0x 7FE8177F【注 1】

【注 1】UNIX 时间和年月日时间要分开设置，其中 UNIX 时间 0x386D4380～0x 7FE8177F 对应年月日时间是 2000.01.01 00:00:00~2037.12.31 23:59:59（GMT 0:00 时区）；

4.5 操作寄存器

地址	类型	描述	数据格式	范围/备注
9100	WO	DO 遥合预置	uint16	0xFF00【注 1】
9101	WO	DO 遥合执行	uint16	0xFF00
9102	WO	DO 遥分预置	uint16	0xFF00
9103	WO	DO 遥分执行	uint16	0xFF00

【注1】：遥控分为预置和非预置两种模式，通过“遥控预置使能”寄存器设置；“非预置”模式，操作执行寄存器直接生效；“预置”模式对于装置的继电器操作，分两步执行，先遥控预置寄存器成功，再遥控执行寄存器成功，此时继电器才会真正的执行动作。预置之后15秒没有执行操作，则预置命令无效。

4.6 装置信息寄存器

地址	类型	描述	数据格式	备注
9800～9819	RO	设备类型	uint16	【注1】
9820	RO	程序版本号	uint16	如10000，表示版本 V1.00.00
9821	RO	MODBUS规约版本	uint16	如10表示规约版本

				1.0
9822	RO	程序版本更新日期（年）	uint16	如：20140110表示版本日期为2014年1月10日
9823	RO	程序版本更新日期（月）	uint16	
9824	RO	程序版本更新日期（日）	uint16	
9825	RO	装置序列号	uint32	如：1401030100表示2014年1月3日生产的第100台装置
9827	RO	备用		
9828	RO	726I/726V/726M选型	uint16	1：D726I 2：D726V 3：D726M
9829	RO	装置选型配置	uint16	Bit0：Bit1： DI/DO/AO 配置 00：NC 01 ： 2 路 DI 10 ： 1 路 DO 11 ： 1 路 AO Bit2：系统频率 0:50hz 1:60hz Bit3：是否选配 RS-485 功能 1:配置 RS485 通讯 0:不配置RS485通讯
9830	RO	电流输入额定值	uint16	
9831	RO	电压输入额定值	uint16	

【注 1】： 设备类型寄存器，共含 20 个寄存器，每个寄存器地址存放一个字符的 ASCII 码，多余的寄存器填 0x0020，用于将来扩展。比如“PMC-D726X”，则读 9800～9810 寄存器的值分别是：0x50、0x4d、0x43、0x2d、0x44、0x37、0x32、0x36、0x58、0x20、0x20，读 9811～9819 寄存器的值均为 0x20。

4.7 SOE 事件记录寄存器

地址	寄存器类型	描述	备注
10000～10007	RO	事件1	SOE LOG 【注1】
10008～10015	RO	事件2	SOE LOG 【注1】
10016～10023	RO	事件3	SOE LOG 【注1】
10024～10031	RO	事件4	SOE LOG 【注1】

10032～10039	RO	事件5	SOE LOG 【注1】
10040～10047	RO	事件6	SOE LOG 【注1】
10048～10055	RO	事件7	SOE LOG 【注1】
10056～10063	RO	事件8	SOE LOG 【注1】
10064～10071	RO	事件9	SOE LOG 【注1】
10072～10079	RO	事件10	SOE LOG 【注1】
10080～10087	RO	事件11	SOE LOG 【注1】
10088～10095	RO	事件12	SOE LOG 【注1】
10096～10103	RO	事件13	SOE LOG 【注1】
10104～10111	RO	事件14	SOE LOG 【注1】
10112～10119	RO	事件15	SOE LOG 【注1】
10120～10127	RO	事件16	SOE LOG 【注1】

【注 1】：SOE LOG 定义如下，每条事件占用8个寄存器地址：

+0RO 在SOE缓冲区的存储位置（0～15）

+1(Hi) RO 类**（均为0）**

+1(Lo) RO 事件代码（子类）**请见下表。**

+2(Hi) RO 年（-2000）

+2(Lo) RO 月

+3(Hi) RO 日

+3(Lo) RO 时

+4(Hi) RO 分

+4(Lo) RO 秒

+5 RO 毫秒

+6 RO 记录值高字

+7 RO 记录值低字

子类值	1	2	3	11	12	21	22	31	32
含义	DI1变位	DI2变位	DO变位	面板清除电能	通信清除电能	通信或面板清除SOE	装置掉电	面板修改参数	通信修改参数
记录值	1:闭合 0:打开	1:闭合 0:打开	1:闭合 0:打开	0	0	0	0	0	0

附录 A、CRC-16 校验算法

使用RTU模式，消息包括了一基于CRC方法的错误检测域。CRC域检测了整个消息的内容。

CRC域是两个字节，包含一16位的二进制值。它由传输设备计算后加入到消息中。接收设备重新计算收到消息的CRC，并与接收到的CRC域中的值比较，如果两值不同，则有误。

CRC是先调入一值是全“1”的16位寄存器，然后调用一过程将消息中连续的8位字节各当前寄存器中的值进行处理。仅每个字符中的8Bit数据对CRC有效，起始位和停止位以及奇偶校验位均无效。

CRC产生过程中，每个8位字符都单独和寄存器内容相或(OR)，结果向最低有效位方向移动，最高有效位以0填充。LSB被提取出来检测，如果LSB为1，寄存器单独和预置的值或一下，如果LSB为0，则不进行。整个过程要重复8次。在最后一位(第8位)完成后，下一个8位字节又单独和寄存器的当前值相或。最终寄存器中的值，是消息中所有的字节都执行之后的CRC值。

CRC添加到消息中时，低字节先加入，然后高字节。

CRC简单函数如下：

```
unsigned short CRC16(puchMsg, usDataLen)
unsigned char *puchMsg ；  /* 要进行CRC校验的消息 */
unsigned short usDataLen ；  /* 消息中字节数 */
{
    unsigned char uchCRCHi = 0xFF ；  /* 高CRC字节初始化 */
    unsigned char uchCRCLo = 0xFF ；  /* 低CRC 字节初始化 */
    unsigned uIndex ；  /* CRC循环中的索引 */
    while (usDataLen--) /* 传输消息缓冲区 */
    {
        uIndex = uchCRCHi ^ *puchMsgg++ ；  /* 计算CRC */
        uchCRCHi = uchCRCLo ^ auchCRCHi[uIndex]；
        uchCRCLo = auchCRCLo[uIndex] ；
```

```
    }

    return (uchCRCHi << 8 | uchCRCLo) ；

}

/* CRC 高位字节值表 */

static unsigned char auchCRCHi[] = {

    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
    0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
    0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
    0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
    0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
    0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40, 0x00, 0xC1, 0x81, 0x40,
    0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0, 0x80, 0x41, 0x00, 0xC1,
    0x81, 0x40, 0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41,
    0x00, 0xC1, 0x81, 0x40, 0x01, 0xC0, 0x80, 0x41, 0x01, 0xC0,
    0x80, 0x41, 0x00, 0xC1, 0x81, 0x40

    } ；

    /* CRC低位字节值表*/

    static char auchCRCLo[] = {

    0x00, 0xC0, 0xC1, 0x01, 0xC3, 0x03, 0x02, 0xC2, 0xC6, 0x06,
    0x07, 0xC7, 0x05, 0xC5, 0xC4, 0x04, 0xCC, 0x0C, 0x0D, 0xCD,
    0x0F, 0xCF, 0xCE, 0x0E, 0x0A, 0xCA, 0xCB, 0x0B, 0xC9, 0x09,
    0x08, 0xC8, 0xD8, 0x18, 0x19, 0xD9, 0x1B, 0xDB, 0xDA, 0x1A,
```

```
0x1E, 0xDE, 0xDF, 0x1F, 0xDD, 0x1D, 0x1C, 0xDC, 0x14, 0xD4,
0xD5, 0x15, 0xD7, 0x17, 0x16, 0xD6, 0xD2, 0x12, 0x13, 0xD3,
0x11, 0xD1, 0xD0, 0x10, 0xF0, 0x30, 0x31, 0xF1, 0x33, 0xF3,
0xF2, 0x32, 0x36, 0xF6, 0xF7, 0x37, 0xF5, 0x35, 0x34, 0xF4,
0x3C, 0xFC, 0xFD, 0x3D, 0xFF, 0x3F, 0x3E, 0xFE, 0xFA, 0x3A,
0x3B, 0xFB, 0x39, 0xF9, 0xF8, 0x38, 0x28, 0xE8, 0xE9, 0x29,
0xEB, 0x2B, 0x2A, 0xEA, 0xEE, 0x2E, 0x2F, 0xEF, 0x2D, 0xED,
0xEC, 0x2C, 0xE4, 0x24, 0x25, 0xE5, 0x27, 0xE7, 0xE6, 0x26,
0x22, 0xE2, 0xE3, 0x23, 0xE1, 0x21, 0x20, 0xE0, 0xA0, 0x60,
0x61, 0xA1, 0x63, 0xA3, 0xA2, 0x62, 0x66, 0xA6, 0xA7, 0x67,
0xA5, 0x65, 0x64, 0xA4, 0x6C, 0xAC, 0xAD, 0x6D, 0xAF, 0x6F,
0x6E, 0xAE, 0xAA, 0x6A, 0x6B, 0xAB, 0x69, 0xA9, 0xA8, 0x68,
0x78, 0xB8, 0xB9, 0x79, 0xBB, 0x7B, 0x7A, 0xBA, 0xBE, 0x7E,
0x7F, 0xBF, 0x7D, 0xBD, 0xBC, 0x7C, 0xB4, 0x74, 0x75, 0xB5,
0x77, 0xB7, 0xB6, 0x76, 0x72, 0xB2, 0xB3, 0x73, 0xB1, 0x71,
0x70, 0xB0, 0x50, 0x90, 0x91, 0x51, 0x93, 0x53, 0x52, 0x92,
0x96, 0x56, 0x57, 0x97, 0x55, 0x95, 0x94, 0x54, 0x9C, 0x5C,
0x5D, 0x9D, 0x5F, 0x9F, 0x9E, 0x5E, 0x5A, 0x9A, 0x9B, 0x5B,
0x99, 0x59, 0x58, 0x98, 0x88, 0x48, 0x49, 0x89, 0x4B, 0x8B,
0x8A, 0x4A, 0x4E, 0x8E, 0x8F, 0x4F, 0x8D, 0x4D, 0x4C, 0x8C,
0x44, 0x84, 0x85, 0x45, 0x87, 0x47, 0x46, 0x86, 0x82, 0x42,
0x43, 0x83, 0x41, 0x81, 0x80, 0x40

} ;
```

修订记录

序号	版本号	修改日期	修改摘要	修改人
1	V1.0	2014-09-24	第一版	张红卫
2	V1.0	2014/11/19	在第一版基础上修正部分说明。	张红卫