# 说明

1. 数据包均为8 \* 32位，下面都采用数组的格式进行说明

即：unsigned char data[32];

2. 数组中未填写的部分都为0

# 上位机向下位机传输的数据格式

## 电源设定数据封装

data[0] = 0x09; //数据包开始

data[1] = 0x0A; //电源设定标识

data[2] 到 data[4] //脉冲宽度

data[5] 到data[7] //幅值电压

data[8] 到 data[10] //脉冲频率

data[29] = 0x8D; //临时存放数据结束标志，占位没啥用

data[30] = 0x8E; //临时存放数据结束标志

unsigned int CRCFlag = CRCHandle(29); //产生CRC( [crc代码](#_CRC代码))

data[29] = CRCFlag >> 8; //29位存放CRC的高8位

data[30] = CRCFlag & 0xff; //30位存放CRC的低8位

data[31] = 0x10; //结束位

## 运动设定数据封装

data[0] = 0x09;

data[1] = 0x0B; //运动设定标识符

data[2] 到 data[4] // X的差值（当前值 与 设定值的绝对值）

data[5] 到 data[7] // Y的差值（当前值 与 设定值的绝对值）

data[8] 到 data[10] // Z的差值（当前值 与 设定值的绝对值）

data[11] 到 data[13] // PZT的差值（当前值 与 设定值的绝对值）

data[14] 到 data[16] // X设定速度（前面的都是位置）

data[17] 到 data[19] // Y设定速度

data[20] 到 data[22] // Z设定速度

data[23] // X,Y,Z的移动方向([详解在下面](#第23位详解))

data[29] = 0x8D; //临时存放数据结束标志

data[30] = 0x8E; //临时存放数据结束标志

unsigned int CRCFlag = CRCHandle(29); //产生CRC

data[29] = CRCFlag >> 8; //29位存放CRC的高8位

data[30] = CRCFlag & 0xff; //30位存放CRC的低8位

data[31] = 0x10; //结束位

**data[23]详解：**

最低位，标识X的方向，为0时，X正方向移动，

为1时，负方向移动

次低位，标识Y的方向

倒数第三位，标识Z的方向

例如：data[23] = 0x07，即 0000 0111

表示：X,Y,Z都沿正方向移动

## 点位控制数据封装

直接发送三位数据

+X：0x20 20 10

//data[0] = 0x20 data[1] = x020 data[12] = 0x10 此时代表X+

-X：0x20 21 10

+Y：0x20 22 10

-Y：0x20 23 10

+Z：0x20 24 10

-Z：0x20 25 10

+PZT：0x20 26 10

-PZT：0x20 27 10

# 下位机向上位机传输的数据格式

## 传回运动设定的数据包

data[0] = 0x09

data[1] = [0x1F, 0x2F]

//这个范围内，则在原本的位置开始改变值

//否则，从原点 (0位置) 开始改变值

data[2] 到 data[4] // X的差值（当前值 与 设定值的绝对值）

data[5] 到 data[7] // Y的差值（当前值 与 设定值的绝对值）

data[8] 到 data[10] // Z的差值（当前值 与 设定值的绝对值）

data[11] 到 data[13] // PZT的差值（当前值 与 设定值的绝对值）

data[23] // X,Y,Z的移动方向([详解在上面](#第23位详解))

data[29]; //29位存放CRC的高8位

data[30]; //30位存放CRC的低8位

data[31] = 0x10; //结束位

# 正确标识与错误标识

## 正确标志

data[0] = 0x10;

data[29]和data[30]为CRC校验位

data[30] = 0x10 结束标识位

10 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 14 E9 10

## 错误标志

data[0] = 0x10

data[1] = 0x11

data[29]和data[30]为CRC校验位

data[30] = 0x10 结束标识位

10 11 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 56 7F 10

# CRC代码即参考文献

## CRC代码

unsigned short DataHandle::CRCHandle(unsigned int len)

{

unsigned short CrcFlag = 0xFFFF; //初值与下位机协定好

unsigned int m\_dataIndex = 0;

while(len--)

{

for(unsigned char i=0x80; i != 0; i >>= 1) //8次循环

{

if((CrcFlag & 0x8000) != 0) //上一位存在余式，CRC乘以2再求CRC

{

CrcFlag <<= 1;

CrcFlag ^= 0x1021; //CRC生成多项式 0x1021，欧洲推荐标准

}

else //上一位不存在余式

{

CrcFlag <<= 1;

}

if((m\_data.data[m\_dataIndex]&i) != 0)

CrcFlag ^= 0x1021; //最后加上本位的异或值，产生本位CRC

}

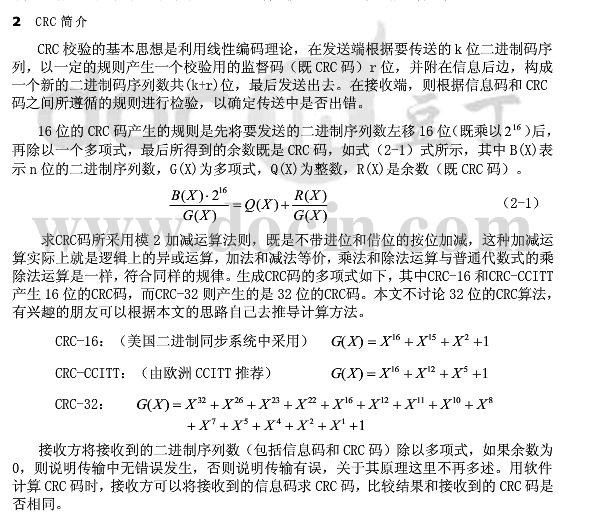
m\_dataIndex++;

}

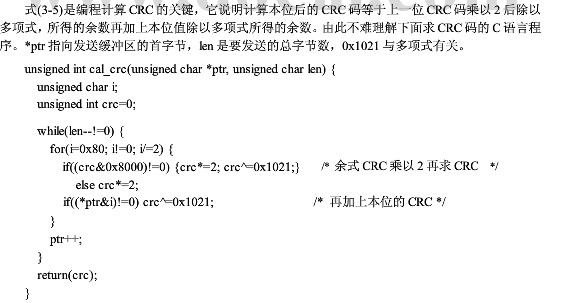
return CrcFlag; //Crc校验码

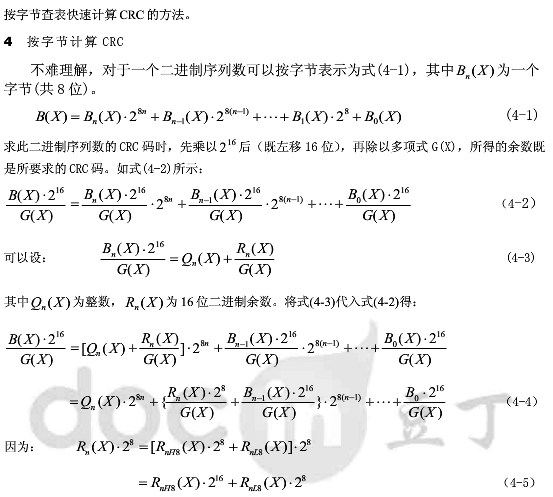
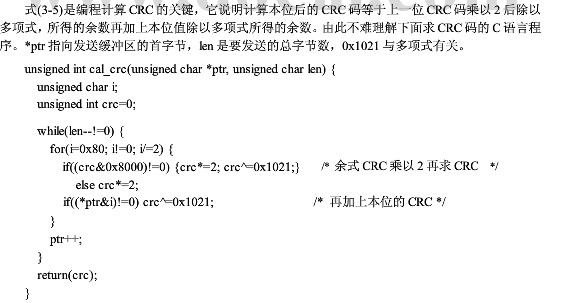
}

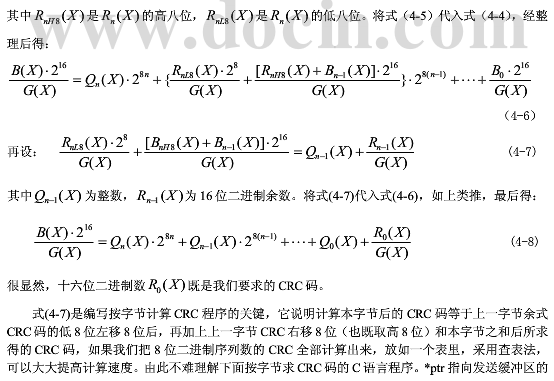
## CRC参考文献











C:\Users\32861\AppData\Local\Temp\enhtmlclip\H(IQXEVDP7(OB0KJSRNN0B9.png