

# 水情遥测系统中快速差错校验的软件方法

Wangljin 收藏.

交流论坛: <http://bbs.cepark.com/>

推荐网站: <http://www.cepark.com>

个人博客: <http://wangljin.cepark.com>

**摘要:** 讨论了在自报式水情无线遥测系统数据通信中进行快速差错校验的必要性, 给出了经过实验检验、可行的软件快速校验方法, 并比较了它们的优劣与适应的场合。

将测站的实时水情数据(水位、闸位、雨量等)准确无误地发送到中心站, 提供水文洪水预报、洪水调度、防洪排涝决策等高一级系统, 是水情遥测系统最基本、最重要的功能之一。水情遥测系统是一个软硬件综合系统。其基本工作流程是: 在测量端(测站)完成水情参数的采集与处理(信源编码、存储记录、信道编码等), 然后将处理过的数据通过无线或有线信道直接或经中继发送至远端的中心站, 由中心站进行接收解码并作进一步处理。图 1 为水情无线遥测系统结构示意图。测站和中继站的主控设备一般采用单片机, 用汇编语言编程; 而中心接收端主机一般采用微型机, 用 C 语言(或其它高级语言)编程。

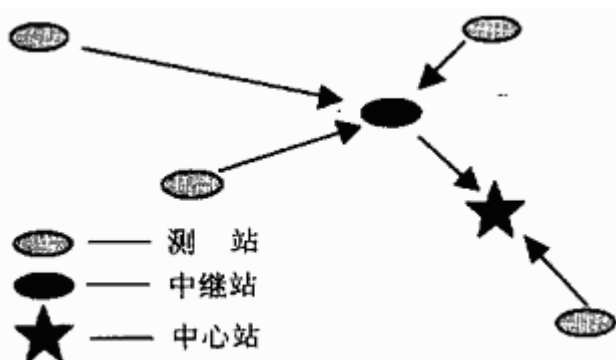


图 1 水情无线遥测系统结构

水情遥测系统的报讯方式一般有三种: 定时自报式、查询-应答多和混合式(自报和查询-应答兼容)。三种报讯方式兼有长短。综合考虑系统功耗、可靠性、复杂性等要素, 定时自报方式在水情遥测系统中仍占主流。其优点是: 功耗极低(值守状态 $<50\mu A@+12V$ , 大多以蓄电池供电), 系统结构简单, 可靠性较高; 缺点是无法实现反馈重发、反传校验等差错控制。显然, 需要选择一种合适的有差错校验方法。

水情遥测系统的数据通信方式可分为超短波通信、微波通信、卫星通信、移动通信、有线通信等。因其遥测站点常建于交通不便、供电及有线通信条件不足的地区, 基于建设成本及运行费用等考虑, 数据通信仍以无线超短波通信为主要方式。

在无线数据通信过程中, 即使信道质量良好, 但由于信号衰减、失真, 特别是某些突发性的干扰(如雷电, 电磁辐射)不可避免地会发生数据误传, 即误码。根据水情遥测系统的相关规范, 超短波数据传输的误码率应小于  $10^{-4}$ , 以及在每个数据收集周期平均应有 90% 以上测站(重点控制站必须包括在内)能准确传送数据至中心站。因此采取适当的差错控制方法, 提高数据传输的可靠性很有必要。常用的差错控制方法分软件和硬件方式。最简单的是由器件直接实现奇偶校验方式, 它占用 10% 的时间, 只检出奇数个位出错。据检测, 在电话网中以 1200 波特率传输数据时, 若采用奇偶校验方式, 仍会有 40% 的错误不能检出, 这对水情遥测显示是不够的。欲对包括中继在内的每一个站实行码校验, 还要求所选校验方式具有高检出率、速度快、编码简单等特点。常见的方式有汉明码、循环冗余校验(CRC)等, 虽然这

个人博客: <http://wangljin.cepark.com>

电子综合站点: <http://www.cepark.com>

些校验方式也可由硬件实现,但人们角倾向于采用简单经济又具灵活性的软件校验。以下结合工作实际给出经验证可行的快速校验方式,并比较了它们的优劣。文中所涉及到的程序算法均以 C 语言的形式给出,而将其转变成单片机的算法也不难。

## 1 CRC 校验

CRC (Cyclical Redundancy Check) 校验,又称循环冗余校验,具有极强的检错能力(不能纠错),算法简单。早期用硬件电路直接搭成,但软件方法成本更低,实现更简单,运算速度也很快。16 位的 CRC 检错率如表 1 所示[1]。

表 1 16 位的 CRC 检错率

单位个位错误	双位错误	奇数个位错误	比 16 位短的突出性错误	恰好 17 位的突发性错误	其他所有突发性错误
100%	100%	100%	100%	99.9969%	99.9984%

常用的 16 位 CRC 多项式有两种:一种是 CRC-CCITT 标准,在微机通信的 XMODEM 协议中得到了应用;另一种是 CRC-16 标准,它实际捕获错误的能力不如 CRC-CCITT,在 IBM 的二进制同步协议(BYSYNC)的数据传送中应用已久。两者采用的多项式如表 2 所示,本文采用前者。

表 2 常用的 16 位 CRC 多项式

生成多项式的值(genpoly)	本原多项式表示	标 准
1021H	$X^{16}+X^{12}+X^5+1$	CRC-CCITT
F005H	$X^{16}+X^{15}+X^2+1$	CRC-16

注: genpoly 为 generator polynomial 的合成调,在程序中用作“生成多项式”寄存器。

### 1.1 直接模 2 除法 CRC 实现方式

对 16 位的 CRC 而言,用信息段作被除数,生成多项式(本文 1021H, CCITT 标准)作除数,进行模 2 除法所产生的余数(2 字节)即为 CRC 校验值,且 CRC 校验只间余数而不管商是多少。发送时将校验值连在信息段的后面一起发送。在接收端,接收方只需把接收到的 CRC 校验值连同信息一,作为新的信息段并对其进行相同的 CRC 运算(只比发送时多 2 字节)。若得到的新余数(校验值)为 0,则表明接收到的信息段和 CRC 都无差错;反之,说明信息段或 CRC 有错,应做相应处理。所以 CRC 的编码和译码并没有本质的区别。程序如下: USHORT crc(USHORT data,USHORT genpoly,USHORT accum)

{// data: 数据,所用信息字的第一个字节; genpoly: CRC 多项式,如 1021H; accum: 累加器的值,第一次赋 0,以后放每次校验结果。

```
data<<=8; //信息字节左移到高字节
```

```
for(int i=8;i>0;i--){
```

```
if((data^accum)&0x800) //如果 (data 异或 accum) 的最高位是 1
```

```

accum=(accum<<1)^genpoly; //移位与 genpoly 异或

else accum<<=1; //否则仅移位

data<<=1; //将信息字的下一位升格

}

return accum; //返回用作下一个信息字校验的累加器值

}
    
```

## 1.2 快速 CRC 实现方式

直接模 2 除法 CRC 方式虽编程简单,但效率不高。采用查表方式,要使用 16 位的多项式及两字节的累加器,对每一信息位 (bit) 累加器都要移位一次,再根据移位结果判断是否作异或;每一字节重复 8 位,运算速度相对较慢,不符合计算机按比特进行计算的规律。但如果采用微机通信中 XMODEM 协议所使用的 CRC 查询方式,则比直接 CRC 模 2 除法方式快 4~10 倍。查询方法实施过程:首先用信息字节与累加器的高字节进行异或,并将其结果作初始累加器为 0 的 CRC;然后与原累加器的低字节再作一次异或。第一步只有 256 个样式,可以构造一个 256 个双字节的查询表,一步实现。这样对每一字节只要作两次操作就可完成。以下是具体步骤。

(1) 构造查询表,运行直接模 2 除法 CRC 函数  $CRC(i, 1021, 0)$ , 用  $i$  从 0~255 代入,将结果按序排列可得到一个 256 个样式的双字节查询表。该表只作一次,可以先用 C 语言微型机上作好,然后再移到单片机上,留作以后查询使用。

(2) 取一个双字节累加器  $accum$ , 赋初值 0, 将信息流的第一个字节赋给另一双字节变量  $data$  ( $accum$  和  $data$  都是双字节变量,以下步骤也是作双字节运算)。

(3) 将  $accum \gg 8$  (也即取原累加器  $accum$  的高字节) 的值与信息字  $data$  相异或, 所得结果 (是一个 <256 的值) 查上述构造好的查询表, 得到一个 16 位的暂存值。

(4) 将  $accum \ll 8$  (即原累加器  $accum$  的低字节左移成高字节, 低位补 0), 与上一步得到的暂存值 (16 位的值) 相异或, 结果作为新的累加器值, 赋值给  $accum$ 。

(5) 取信息流的下一字节赋给  $data$ , 重复进行第 (3) 步和第 (4) 步, 直至所有的信息字节读用完后为止, 最后累加器的值就是余数。

## 2 扩展汉明码

### 2.1 编码方法

CRC 校验只能检错但不能纠错。而 1949 年提出的汉明码是一种能纠正单个错误的线性分组码。其中, 既是线性分组码同时也是循环码的 (7, 4) 码有两种。其生成矩阵和校验矩阵分列如下:

$$G_1 = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{pmatrix} \quad H_1 = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

$$G_2 = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 \end{pmatrix} \quad H_2 = \begin{pmatrix} 1 & 0 & 1 & 1 & 1 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{pmatrix}$$

两者使用效果等价。

汉明码是纠正单个错误的完备码，所有的接收码都可对应到一个信息（多一对应），要么是正确信息，要么是发生单个错误的情形。当有两个错误时，会把它当成另一个码的单个错误加以纠正，导致误码。

扩展汉明码在此基础上引入一个校验和，即在编码在时候增加第 8 位偶校验位，构成（8，4）线性分组码，因而可以纠正一位错误同时检出两位错误。事件上，在发生错误时就是这个偶校验位确定了是错一位还是错两位。若错一位则可以纠正，错两位就只能检出但不能纠正。

编、译码均以扩展汉明码（8，4）线性分组码为例。为了方便单片机的运算，实现快速编码，可采用查询法。因为信息是一个 4 位的矢量，记作 C，共有 16 个可能值。为了构成 8 位发射码矢量，可以建立 16 个一字节的查询作为 8 位的发送码。以生成矩阵 G1 为例，用信息矢量 C 乘以成矩阵 G1 再加上一位偶校验就得到了生成码（发送码）。查询表为：

信息	生成码	信息	生成码	信息	生成三	信息	生成码
0000	00000000	0100	01001110	1000	10001011	1100	11000101
0001	00010111	0101	01011001	1001	10011100	1001	10011100
0010	00101101	0110	01100011	1010	10100110	1110	11101000
0011	00111010	0111	01110100	1011	10110001	1111	11111111

## 2.2 译码方法

用查询法对（8，4）码进行译码，需要建造有 256 个值的查询表。按照译码编写查询表。先定出扩展汉明码的校验矩阵，实际上就是将原校验矩阵 H1 扩展，记为 H1，

$$\tilde{H}_1 = \begin{pmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

对于作一 8 位的接收码矢量 R, 进行 RH1 T 运算, 得到一个 4 位的伴随矢量, 再按如下步骤比较确定原信息。

(1) 如果伴随式矢量是全 0 矢量, 接收码是正确的, 码的前 (低) 4 位就是信息。

(2) 如果伴随矢量的最后一位是 1, 则有一位错, 可纠正。将伴随矢量与矩阵 H1 的每一列相比较, 找出相同的那一列, 记下列号, 再将接收码与该列号相对应的那一位变号 (1 变 0, 0 变 1), 得到的码就是纠正后的原码, 信息取码的前 (低) 4 位。

(3) 否则, 是一位以上的错码且不能纠正。

将一个字节可能出现的所有 0~255 个可能值都按上面的译码步骤做一遍得到查询表, 留作译码用。另外译码和编码还可以对整个信息字节作一字节的垂直校验以增强校验能力。

上述检验方式已在江苏、宁夏、福建等地的实际工作中得到了验证。CRC 校验虽不具备纠错功能但有很高的检错率, 应用面也很广。其中, 直接模 2 除法 CRC 方式因编程简单、占用程序空间少 (不用查询表), 适合于数据通信量不大且程序及内存空间有限的场合, 反之可选用快速 CRC 方式。在对数据完整性要求高的场合, 可根据具体情况考虑使用汉明码呀扩展汉明码。某些要求更高的特殊情况下, 则可选用更复杂一些的校验码, 同时通信条件的好坏也是影响校验方式选用的因素之一。

最下面给大家介绍几个下载资料的地方:

51 学习专区:

<http://51.cepark.com/>

USB 学习专区:

<http://usb.cepark.com/>

CAN 学习专区:

<http://can.cepark.com>

AVR 学习专区:

<http://avr.cepark.com/>

FPGA 学习专区:

<http://fpga.cepark.com/>

STM32 学习专区:

<http://stm32.cepark.com/>

ARM 学习专区:

个人博客: <http://wang1jin.cepark.com>

电子综合站点: <http://www.cepark.com>

<http://arm.cepark.com/>

DSP 学习专区:

<http://eda.cepark.com/>

PIC 学习专区:

<http://pic.cepark.com/>

DIY 电子制作专区:

<http://diy.cepark.com/>

GPS 学习专区:

<http://gps.cepark.com/>

GUI 学习专区:

<http://gui.cepark.com/>

EDA 软件学习专区:

<http://eda.cepark.com/>

电源学习专区:

<http://power.cepark.com/>