中图分类号:TP331　　文献标识码:A　　文章编号:1009 - 2552(2007)04 - 0071 - 04

32 位 CRC 校验码的并行算法及硬件实现

俞　迅

(同济大学电子与信息工程学院, 上海 200092)

摘　要: 通过对 CRC 校验码原理的分析, 研究了一种并行 32 位 CRC 算法。该算法采用递推的方法, 直接得出计算多位数据后的 CRC 余数与计算前余数之间的逻辑关系。相对于一般的按位串行计算或者查表并行计算的方法来说, 该方法运算速度快且不需要额外的空间存储余数表, 十分有利于硬件实现。

关键词: CRC; 模 2 运算; 并行 CRC 算法

The 32 - bit cyclic redundancy check parallel algorithm and hardware implementation

YU Xun

(**College of Electronic and Information Engineering, Tongji University, Shanghai** 200092, **China**)

Abstract: Based on the theory of the cyclic redundancy check, a parallel algorithm is studied in the paper. This algorithm uses a recursive method to calculate the logic relationship of the checksum. Differing from general serial algorithm or the parallel algorithm based on list - checking, it is faster and doesn't need the extra memory space to store the remainder list. It is very easy to be implemented by hardware.

Key words: Cyclic redundancy check; modulo 2 arithmetic; CRC parallel algorithm

计算机系统中的数据,在进行读、写或者传输时可能产生错误,为了减少和避免错误的产生,一方面可以通过对特定电路的精心设计,提高电路的稳定性和可靠性;另一方面则是对数据采用某种编码,通过少量的附加电路,使之能发现某些错误,甚至能确定出错位置,进而实现自动改错的功能。CRC(循环冗余码)就是一种常用的错误检测码,它可以发现并纠正数据存储或传输过程中连续出现的多位错误,因此在介质存储和网络通信方面得到了广泛的应用。随着技术的发展,数据存储和传输速度大大提高,在一些高速的场合如 usb2.0 或者快速以太网中,传统的串行 CRC 算法已不能满足速度上的要求,而必须采用速度更快的并行算法。

1　CRC 校验码原理简介

CRC 校验的基本思路是利用线性码原理,对需要进行传输的原始 k 位二进制数据按照一定的规则处理,产生一个 r 位的校验码并附加在原始数据后面,形成一个 $k + r$ 位的二进制数据,最后一起发送出去。

首先,可将待编码的 k 位数据表示成多项式 $M(X)$:

$$M(X) = C_{k-1}X^{k-1} + C_{k-2}X^{k-2} + \cdots + C_iX^i + \cdots + C_1X + C_0$$

其中 C_i 为 0 或者 1。

对于 r 位 CRC 来说,校验码产生的过程为:

将 $M(X)$ 左移 r 位,然后除以一个被称为生成多项式的 $G(X)$,所得余数就是 CRC 校验码。这里,生成多项式 $G(X)$ 是一个 $r + 1$ 位的多项式。

用公式表示如下:

$$\frac{M(X) \cdot X^r}{G(X)} = Q(X) + \frac{R(X)}{G(X)}$$

其中 $Q(X)$ 为商,在 CRC 的计算过程中不需要关注,$R(X)$ 为余数,就是需要的 CRC 码。CRC 的计算使

收稿日期: 2006 - 11 - 20

作者简介: 俞迅(1982 -),男,同济大学微电子与固体电子学在读硕士研究生,研究方向为集成电路前端设计及仿真。

用的是模 2 运算,即不带进位和借位的按位加减,这在逻辑上等同于异或运算。

2 串行 32 位 CRC 算法

设 $c_{31}^{j}x^{31}+c_{30}^{j}x^{30}+\cdots+c_{1}^{j}x+c_{0}^{j}=x^{32}(d_{k-1}x^{k-1}+d_{k-2}x^{k-2}+\cdots+d_0)\bmod G(x)$

为计算前的 CRC 多项式,g_i 为生成多项式 $G(x)$ 的第 i 位系数。

则新读入一位数据 d 后,

$$x^{32}(d_{k-1}x^{k-1}+d_{k-2}x^{k-2}+\cdots+d_ix^i+d_0x+d')\bmod G(x)=x^{33}(d_{k-1}x^{k-1}+d_{k-2}x^{k-2}+\cdots+d_ix^i+d_0)\bmod G(x)+x^{32}d'\bmod G(x)=(c_{30}^{j}+g_{31}c_{31}^{j})x^{31}+(c_{29}^{j}+g_{30}c_{31}^{j})x^{30}+\cdots+(c_0^{j}+g_1c_{31}^{j})x+g_0c_{31}^{j}+g_{31}d'x^{31}+g_{30}d'x^{30}+\cdots+g_1d'x+g_0d'=g_0(c_{31}^{j}+d')+\sum_{i}^{31}[c_{i-1}^{j}+g_i(c_{31}^{j}+d')]x^i$$

若记

$$c_{31}^{j+1}x^{31}+c_{30}^{j+1}x^{30}+\cdots+c_1^{j+1}x+c_0^{j+1}=x^{32}(d_{k-1}x^{k-1}+d_{k-2}x^{k-2}+\cdots+d_ix^i+d_0x+d')\bmod G(x)$$

为计算一位数据 d 后的 CRC 多项式,则可得:

$$c_0^{j+1}=g_0(c_{31}^{i}+d')\quad c_i^{j+1}=c_{i-1}^{j}+g_i(c_{31}^{j}+d')\quad i\in[1,31]\tag{1}$$

式(1)给出了读入一位数据后新的 CRC 码 c_i^{j+1} 与原来的 c_i^{j} 之间的逻辑关系。

采用这种算法,当原始数据依次输入完毕后,32 位寄存器内的值就是最后的 CRC 码。这种算法无需在原始数据后添加 32 位 0 进行计算。

3 并行 32 位 CRC 算法

串行算法一个时钟周期内只能处理一位数据,在某些高速场合只能靠提高时钟频率来达到所需的速度要求,这增加了开发的难度和成本,因此提出了 CRC 的并行算法。并行 CRC 算法一次读入多位数据,通过当前余数与读入数据的运算,得出新的余数。显然,一个时钟周期处理 n 位数据的并行算法在计算结果上与串行算法处理 n 个时钟周期所得的余数应该相同。基于这一点,采用递推的方法可由串行算法导出并行算法计算前后余数之间的逻辑关系。下面以 4 位 CRC - 32 并行算法为例,说明推导的过程。

CRC - 32 的成生多项式 $G(x)$ 为:

$$x^{32}+x^{26}+x^{23}+x^{22}+x^{16}+x^{12}+x^{11}+x^{10}+x^8+x^7+x^5+x^4+x^2+x+1$$

转换成二进制序列就是

100000100110000010001110110110111

为了便于表达,记为:

$$g_{32},g_{31},\cdots,g_2,g_1,g_0\tag{2}$$

其中,g_i 对应于生成多项式的系数,取 0 或者 1。

定义 c_i^{j} 为计算了第 j 位数据后所得 CRC 值的第 i 位,d_3,d_2,d_1,d_0 为读入的数据顺序,最初时的 CRC 值为:$c_{31}^{0},c_{30}^{0},c_{29}^{0},\cdots,c_2^{0},c_1^{0},c_0^{0}$。基本思想就是连续套用式(1)给出的串行公式 4 次,以期得出处理 4 位数据后 c_i^{4} 与 c_i^{0} 和 d_3,d_2,d_1,d_0 之间的逻辑关系。推导过程如下:

$$c_0^{4}=c_{31}^{3}+d_0$$

$$c_1^{4}=c_0^{3}+g_1(c_{31}^{3}+d_0)=c_{31}^{2}+d_1+g_1(c_{31}^{3}+d_0)$$

$$c_2^{4}=c_1^{3}+g_2(c_{31}^{3}+d_0)=c_0^{2}+g_1(c_{31}^{2}+d_1)+g_2(c_{31}^{3}+d_0)=c_{31}^{1}+d_2+g_1(c_{31}^{2}+d_1)+g_2(c_{31}^{3}+d_0)$$

$$c_3^{4}=c_2^{3}+g_3(c_{31}^{3}+d_0)=c_1^{2}+g_2(c_{31}^{2}+d_1)+g_3(c_{31}^{3}+d_0)=c_0^{1}+g_1(c_{31}^{1}+d_2)+g_2(c_{31}^{2}+d_1)+g_3(c_{31}^{3}+d_0)=c_{31}^{0}+d_3+g_1(c_{31}^{1}+d_2)+g_2(c_{31}^{2}+d_1)+g_3(c_{31}^{3}+d_0)$$

当 $i\in[4,31]$ 时,

$$c_i^{4}=c_{i-1}^{3}+g_i(c_{31}^{3}+d_0)=c_{i-2}^{2}+g_{i-1}(c_{31}^{2}+d_1)+g_i(c_{31}^{3}+d_0)=c_{i-3}^{1}+g_{i-2}(c_{31}^{1}+d_2)+g_{i-1}(c_{31}^{2}+d_1)+g_i(c_{31}^{3}+d_0)$$

$\because$

$$c_{31}^{1}=c_{30}^{0}+g_{31}(c_{31}^{0}+d_3)=c_{30}^{0}$$

$$c_{31}^{2}=c_{30}^{1}+g_{31}(c_{31}^{1}+d_2)=c_{30}^{1}=c_{29}^{0}+g_{30}(c_{31}^{0}+d_3)=c_{29}^{0}$$

$$c_{31}^{3}=c_{30}^{2}+g_{31}(c_{31}^{2}+d_1)=c_{30}^{2}=c_{29}^{1}+g_{30}(c_{31}^{1}+d_2)=c_{29}^{1}=c_{28}^{0}+g_{29}(c_{31}^{0}+d_3)=c_{28}^{0}$$

$\therefore$

$$c_0^{4}=c_{28}^{0}+d_0$$

$$c_1^{4}=c_{29}^{0}+c_{28}^{0}+d_1+d_0$$

$$c_2^{4}=c_{30}^{0}+c_{29}^{0}+c_{28}^{0}+d_2+d_1+d_0$$

$$c_3^{4}=c_{31}^{0}+c_{30}^{0}+c_{29}^{0}+d_3+d_2+d_1$$

当 $i\in[4,31]$ 时,

$$c_i^{4}=c_{i-4}^{0}+g_{i-3}(c_{31}^{0}+d_3)+g_{i-2}(c_{30}^{0}+d_2)+g_{i-1}(c_{29}^{0}+d_1)+g_i(c_{28}^{0}+d_0)$$

至此得到了计算 4 位数据前后 CRC 值之间的

逻辑关系。其中 4 位数据按照高位到低位顺序读入。同样,采用该算法原始数据之后不必添加 32 位 0。以上推导结果表明,计算 4 位数据后的 CRC 值可由当前 CRC 值与输入数据的异或组合来表示,这为并行 CRC 的实现提供了理论基础。

4 4 位并行 CRC - 32 算法的一些改进及模块的设计思路

4.1 CRC - 32 的生成与校验中的一些问题以及算法的改进

当接收端收到包含 CRC 校验的数据包时,用与发送端同样的生成多项式再次计算 CRC 值,如果数据在传输过程中没有发生错误,则计算的结果应该为 0。

这在通常情况下是正确的。现在考虑两种特殊的情况,一种是当数据包的开头有多余 0 出现(或者以连续 0 开头的数据丢失了开头的几位 0),另一种是数据包的结尾有多余的 0 出现(或者以连续 0 结尾的数据丢失了结尾的几位 0)。在这两种情况下,当接收端计算 CRC 的值为 0 时,不能确定没有错误发生,这是因为如果一个数据串的 CRC 计算结果为 0,那么在串头或者串尾增加或减少几位 0,计算的结果仍然是 0,然而数据已经被改变了。

针对这两种情况,CRC 算法作了相应的改动。对于解决串开头 0 的问题,可以把 32 位 CRC 寄存器的值预设为全 1,相当于把原始数据的高 32 位取反来计算 CRC。这样处理后,如果由于传输错误在数据开头出现 0 的增加或减少,经过取反后的数据就会与原来的不同,故而达到了错误检测的目的。由于在数值上这样等于给被除数加上了个常量 F,而在 CRC 的生成和校验阶段都做这样的处理,所以在算法上对最终的校验结果没有影响。

解决串结尾 0 的问题稍微麻烦点,具体做法是生成 CRC 码时将正常计算得出的结果按位取反,作为最终的 CRC 码附加在数据后。校验时在接收到的数据包(包括原始数据和校验码)之后添加 32 个 0,计算除以生成多项式的余数。这时,当数据传输没有发生错误时,CRC 校验的结果不再为 0,而是一个常数,通常被称为魔数(magic number)。同样,当出现数据结尾有 0 的增减错误时,计算所得的余数会发生变化。所有校验结果不等于这个常数的数据包就被认为是出现了错误。

下面给出 32 位 CRC 校验中这个常量的计算方法:

设 M 为原始数据串,G 为生成多项式,R 为取反前的余数,所用的加法均为模 2 加法可知

$$(Mx^{32} + R) \bmod G = 0 \tag{3}$$

现在将 R 取反,计为 $\overline{R}$

$$x^{32}(Mx^{32} + \overline{R}) \bmod G = x^{32}[Mx^{32} + (R + x^{31} + x^{30} + \cdots + x + 1)] \bmod G = x^{32}[(Mx^{32} + R) + x^{31} + x^{30} + \cdots + x + 1] \bmod G = [x^{32}(Mx^{32} + R) + x^{32}(x^{31} + x^{30} + \cdots + x + 1)] \bmod G$$

由式(3) 可知:$x^{32}(Mx^{32} + R) \bmod G = 0$

$$\therefore\ x^{32}(Mx^{32} + \overline{R}) \bmod G = x^{32}(x^{31} + x^{30} + \cdots + x + 1) \bmod G$$

对于 32 位 CRC 校验的生成多项式来说,这个结果为 0xC704DD78。

4.2 CRC - 32 校验码生成和校验模块的设计

生成模块的功能示意图如图 1 所示。

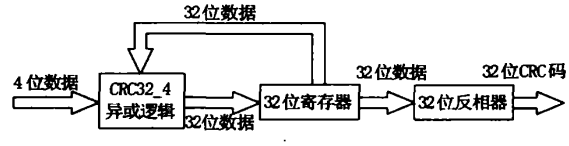

图 1 CRC 生成示意图

32 位寄存器初始化为全 1,将原始数据每个时钟周期读入 4 位数据,与当前寄存器的内容按照第 3 节的公式进行计算,结果存入寄存器中。当所有数据全部读入完毕后,经由反相器输出最终 CRC 校验码。

校验模块的功能示意图如图 2 所示。

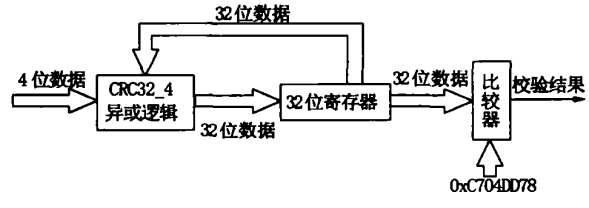

图 2 CRC 校验示意图

同 CRC 生成模块,将 32 位寄存器初始化为全 1,然后把接收到的数据每时钟周期输入 4 位,与当前寄存器中的数据运算,结果存入 32 位寄存器作为新的值,当所有数据输入完毕后,与 0xC704DD78 比较,得出校验结果。

5 电路仿真与结果分析

电路仿真采用的是 ModuleSim SE 6.1f 仿真工具。

取输入数据为 0x0123 4567 7d7b 67c9,每时钟周期读入 4 位,读入顺序从高位到低位。CRC 生成模块的仿真波形图如图 3 所示,当输入结束时,CRC 寄存器被设置为 0xffffffff。

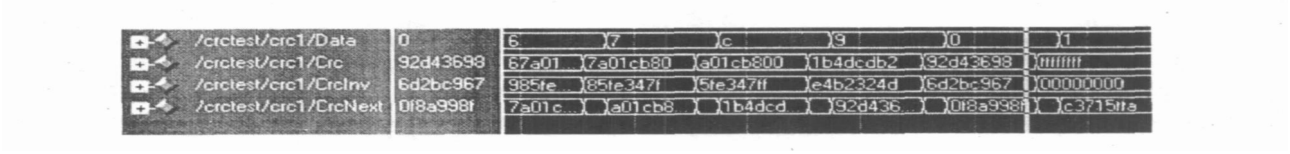

图 3　CRC 生成仿真结果

由图中所示,计算结果为 0x6d2b c967,与理论结果相同。

现在把 0x6d2b c967 作为校验码添加到输入数据之后,作为校验模块的输入,其值为 0x0123 4567 7d7b 67c9 6d2b c967,校验模块的仿真波形如图 4 所示。

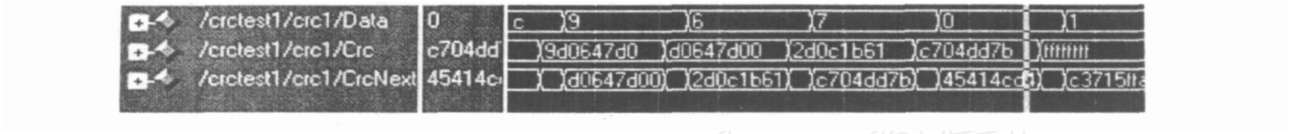

图 4　CRC 校验仿真结果

由图中可以看出,经过校验模块的计算结果为 0xc704 dd7b,与 4.1 中的计算结果相符。

现假设在传输过程中发生了错误,使得校验模块接收到的数据在开头多出了 4 个比特的 0,校验结果如图 5 所示。

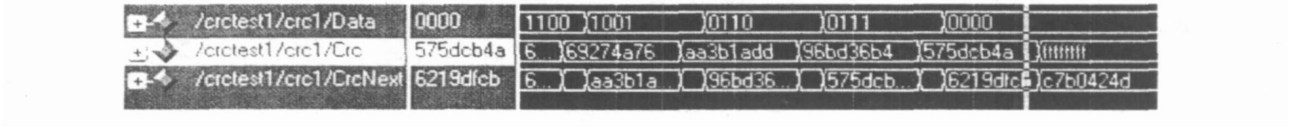

图 5　CRC 校验出错的仿真结果

校验结果为 0x 575d cb4a,与 0xc704 dd7b 不等,所以可以确定传输出现了错误。

6　结束语

本文在对串行 CRC 算法的分析的基础上,利用递推的方法得出了 4 位并行 CRC 校验的逻辑关系公式。算法中 CRC 码的计算和校验完全由一系列的组合逻辑实现,十分有利于转化为硬件电路。同时利用本文中的推导方法也可以方便地得出不同的生成多项式在不同的并行度下的 CRC 计算公式,可以应用到许多高速传输或存储场合的数据校验中。

参 考 文 献:

[1] 李永忠. 通用并行 CRC 实现原理及其硬件实现方法[J]. 西北民族学院学报:自然科学版,2002(3).

[2] Giuseppe Campobello, Giuseppe Patan, Marco Russo. Parallel CRC Realization[J]. IEEE Transactions on Computers. 2003, 52(10): 1312 - 1319.

[3] Andrew S Tanenbaum. Computer Networks[M]. 4th ed. Prentice Hall PTR, 31 Oct 2002.

责任编辑:肖滨

信息天地

信息产业部明确今年信息化思路

“农村信息化”成重点

从信息产业部日前在广州市召开的 2007 年信息化推进工作座谈会上获悉,推进农村信息化等被明确为今年信息化推进工作的重点;同时,如何加强信息化法律法规标准的建设成为讨论的重点。

信息产业部副部长苟仲文在讲话中指出,要加强信息技术在农村、经济、社会、文化、政府管理等领域的应用。2007 年信息产业部推进信息化工作的重点是:推进农村信息化;加强企业信息化;推进城市、社区信息化;加强信息化基础工作。

“加强信息化基础工作,这个‘基础’不是指 IT 网络基础设施,而是指与信息化配套的法律法规、标准规范。”信息产业部信息化推进司司长陈伟说,“要把信息化纳入法制化建设的框架下。”

为此,信息产业部信息化推进司已经并将继续推动电子认证、中小企业信息化、社区信息化、信息系统工程监理等多个领域的政策、法规和标准的起草制订和贯彻落实工作。陈伟司长呼吁各地信息化主管部门积极参与相关法规和标准的制订工作。

关于农村信息化工作,陈伟司长介绍,首批农村信息化试点工作即将正式启动。基本思路是围绕农村信息化的三大瓶颈展开:第一,解决信息进村入户问题。要组织 IT 业为农民提供合用的终端和信息服务,重点扶持农业大户。其次,让信息化手段真正为农民所用。重点是解决好内容问题,让计算机成为农民的一个生产工具。第三,解决机制问题。政府推动、企业参与,在让农民得到实惠的前提下,最终实现市场化运作。