一种并行CRC 算法的实现方法

陈玉泉
(厦门大学 福建 厦门 361005)

摘 要: 简要分析了CRC 算法的基本原理。在传统串行CRC 的实现基础上, 介绍了一种快速的CRC 并行算法, 导出了32 位并行CRC 码的逻辑关系, 推导过程简单。与查表法比较, 此并行算法不需要存储大量的余数表, 可以减少延迟。同时, 这种并行处理方法也适合于其他位宽并行CRC 码。最后, 利用ISE 开发平台和Verilog HDL 硬件描述语言进行设计, 实现了基于此并行算法的32 位并行CRC-32 码的编码器, 并给出了仿真和综合结果。设计出来的CRC 编码器, 已经成功应用于以太网的接入系统中。

关键词: CRC; LFSR; 并行实现; FPGA
中图分类号: TN 914.3 **文献标识码**: B **文章编号**: 1004-373X (2005) 22-021-03

Implementation of a Parallel CRC Calculation

CHEN Yuquan
(Xiamen University, Xiamen, 361005, China)

Abstract: After having analyzed the principle of CRC calculation, this article introduces a fast parallel CRC calculation based on serial realization. This parallel calculation could easy deduce the logical relationship for 32 b parallel CRC. Comparing with Table Lookup Algorithm, it needn't store the large remainder table, and decrease the delay. This parallel processing method is also fit for other bit-wide CRC. Finally, using ISE platform and Verilog HDL, we have designed the 32 b CRC-32 encoder based on this parallel calculation, its simulation and synthesis results are provided. The designed encode have been successfully used to the Ethernet system.

Keywords: CRC; LFSR; parallel implementation; FPGA

1 引 言

在数字通信的传输过程中, 由于信道存在的噪声、线路间的串扰等各种因素的影响, 造成所传输的信号失真。为了提高通信的可靠性和减少误码率, 通常采用信道编码技术来进行差错控制。循环冗余校验 (Cyclic Redundancy Code, CRC) 由于其误码检测能力强, 抗干扰性能优异, 在众多的信道编码方法中得到广泛的应用。

传统的CRC 编译码硬件的实现是采用串行的实现方法, 电路结构简单, 实现容易, 可以工作在较高的时钟频率下, 但速度较慢。随着现代通信的发展, 高速率的数据传输要求加快CRC 的计算速度, 串行的计算方法已经不能满足高速通信领域的要求, 因此越来越多地使用并行的计算方法。本文将介绍一种快速的CRC 并行算法[1], 并借助EDA 工具和硬件描述语言的设计方法对这种算法进行验证和实现。

2 基本原理

CRC 码的结构如图1 所示。图中, $m(x)$ 的 k 个系数对应 k 位信息, $r(x)$ 的 $n-k$ 个系数对应 $n-k$ 个校验位数。在信道编码角度上, 整个 n 位帧就是一个码字, 习惯仅把 $n-k$ 位校验位部分称为CRC 码。

CRC 校验的基本算法是按位操作的, 其原理为首先, 将待传送的位串看作是一个系数为0 或1 的多项式 $m(x)$ 的系数序列, 并定义一个合适的 $(n-k)$ 阶生成多项式 $g(x)$。然后将 $x^{n-k}m(x)$ 作为被除数, $g(x)$ 作为除数, 进行行模二多项式除法, 模二除法所得的余数 $r(x)$ 就是所谓的CRC 校验码。通常将CRC 码加到位串的末端构成 $C(x)$ 一起发送, 即在发送端: $C(x)=x^{n-k}m(x)+r(x)$。

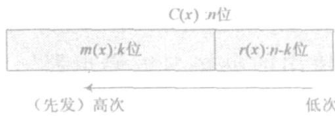

图1 CRC 码结构

接收方在收到信息码后, 如果没有误码, 应有接收码 $R(x)$ 等于发送码 $C(x)$。这时, 接收码 $R(x)$ 应该能被生成多项式 $g(x)$ 模二整除。反之, 如果不能整除, 必是传输中出现了误码。

从以上分析可以看出, 生成多项式在CRC 校验中的重要作用。目前国际上常用的CRC 码及其生成多项式有:

CRC-12: $g(x)=x^{12}+x^{11}+x^3+x^2+1$

收稿日期: 2005-06-30

CRC－ITU－T: $g(x) = x^{16} + x^{12} + x^5 + 1$

CRC－16: $g(x) = x^{16} + x^{15} + x^2 + 1$

CRC－32: $g(x) = x^{32} + x^{26} + x^{23} + x^{22} + x^{16} + x^{12} + x^{11} + x^{10} + x^8 + x^7 + x^5 + x^4 + x^2 + x + 1$

在上述的CRC 码中, CRC－12 码用在字符长度是6 b 的情景下; 国际电联标准CRC－ITU－T 用于HDLC, SDLC, X25, 七号信令、ISDN 等处; CRC－16 用于美国二进制同步系统; CRC－32 用于以太网及ATM AAL5 适配层的CRC。

3　串行实现

通常CRC 的串行实现是通过线性反馈移位寄存器(LFSRs) 来实现的。移位寄存器LFSRs 由m (设生成多项式的最高阶是m) 个D 触发器, 加上一些异或门和一条反馈回路组成。线性反馈移位寄存器(LFSRs) 有2 种可能结构, 分别如图2、图3 所示。本文中把图2 所示的结构称为LFSR, 而把图3 的结构称为LFSR2。

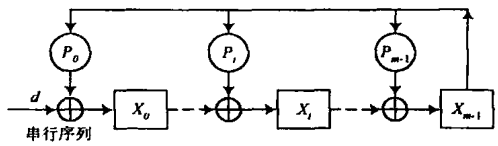

图2　LFSR 结构

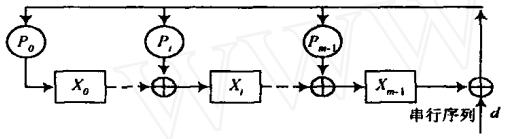

图3　LFSR2 结构

串行编码实现如图2 所示, 开始时, 寄存器初始化为全0 (或全1), 数据流码字按高位到低位 (或反之) 的顺序, 依次通过LFSRs。当数据的一位码字从右边移出寄存器时, 就通过反馈回路并根据生成多项式相应位的值 (1 表示通路, 0 表示断路), 和后续输入数据进行异或运算后, 从左边反馈进入移位寄存器。当所有位数据移入寄存器后, 移位寄存器中所形成的结果即为CRC 校验码。需要指出的是, 利用LFSR 进行CRC 编码时, 应在信息码后面加上m 位的0 序列, 而利用LFSR2 进行编码则不需要。

串行译码使用的电路与编码电路一样, 通常利用LFSR 作为译码电路。与编码不同的是, 利用LFSR 进行译码时, 不需要在信息码后面添加零序列。

4　并行实现

串行实现方法电路结构简单, 但是一个时钟周期只能计算1 位数据, 效率低下, 因此需要引入并行的计算方法, 以提高CRC 的计算速度。本文采用的是文献[1] 提出的并行算法, 这种算法比文献[2] 提出算法的推导过程要简便得多。

将从图2 的LFSR 的结构推导出并行的CRC 码。设CRC 的码长是m, 如果用$X = [x_{m-1}, \cdots, x_1, x_0]^T$ 表示LFSR 各寄存器的第j 个状态值, 用$X' = [x_{m-1}', \cdots, x_1', x_0']^T$ 表示第$(j+1)$ 个状态值; 把生成多项式写成$P = \{p_m, p_{m-1}, p_0\}$; d 表示串行进入LFSR 的1 比特码字; 同时用$\oplus$ 表示异或运算, 用$\otimes$ 表示位与运算, 两者均是模二运算; 则从图2 可以得到下列方程组:

$$\begin{cases} x_{m-1}' = (p_{m-1} \otimes x_{m-1}) \oplus x_{m-2} \\ x_{m-2}' = (p_{m-1} \otimes x_{m-1}) \oplus x_{m-3} \\ \cdots \\ x_1' = (p_1 \otimes x_{m-1}) \oplus x_0 \\ x_0' = (p_0 \otimes x_{m-1}) \oplus d \end{cases} \tag{1}$$

图2 中输入端的输入向量d 为$[d_{i-1}, d_{i-2}, \cdots, d_1, d_0]^T$, 其中$d_{i-1}$ 是码字的高位, 是第1 位进入LFSR 的码字; 而d_0 是低位, 是第i 位进入寄存器组的。实际上图2 所描述的系统是一个离散非时变的线性系统, 所以方程组(1) 是一个类似离散域的系统状态方程, 可以求出他的解[1]:

$$\begin{aligned} X(i) &= [F^i \otimes X(0)] \oplus ([F^{i-1} \otimes G, \cdots F \otimes G, F] \\ &\quad \otimes [d_{i-1}, d_{i-2}, \cdots, d_1, d_0]^T) \\ &= F^i \otimes X(0) \oplus D \end{aligned} \tag{2}$$

其中$i \leq m$, $G = [0, 0, \cdots, 0, 1]^T$, 向量$D = [0, 0, \cdots, 0, d_{i-1}, d_{i-2}, \cdots, d_1, d_0]^T$,

$$F = \begin{bmatrix} p_{m-1} & 1 & 0 & \cdots & 0 \\ p_{m-2} & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \cdots & \vdots \\ p_1 & 0 & 0 & \cdots & 1 \\ p_0 & 0 & 0 & \cdots & 0 \end{bmatrix}$$

如果采用图3 中LFSR2 的电路结构, 在这种情况下$G = [p_{m-1}, p_{m-2}, \cdots, p_1, p_0]^T$, $D = [d_{i-1}, d_{i-2}, \cdots, d_1, d_0, 0, 0, \cdots, 0]^T$, 有:

$$X(i) = F^i \otimes [X(0) \oplus D] \tag{3}$$

从式(2), (3) 可以看出, 采用这种并行算法, 最重要的是要求出矩阵F^i 的值, 对此可以利用Matlab 计算。下面给出本文要用到CRC－32 码的矩阵F^{32}的值, 为了书写方面, 把矩阵的每一行用8 位十六进制数表示, 共有32 项。

$F^{32}_{CRC-32} =$

$$\begin{bmatrix} \text{FB808B20} & \text{7DC04590} & \text{BEE022C8} & \text{5F701164} \\ \text{2FB808B2} & \text{97DC0459} & \text{B06E890C} & \text{58374486} \\ \text{AC1BA243} & \text{AD8D5A01} & \text{AD462620} & \text{56A31310} \\ \text{2B518988} & \text{95A8C4C4} & \text{CAD46262} & \text{656A3131} \\ \text{493593B8} & \text{249AC9DC} & \text{924D64EE} & \text{C926B277} \\ \text{9F13D21B} & \text{B409622D} & \text{21843A36} & \text{90C21D1B} \\ \text{33E185AD} & \text{627049F6} & \text{313824FB} & \text{E31C995D} \\ \text{8A0EC78E} & \text{C50763C7} & \text{19033AC3} & \text{F7011641} \end{bmatrix}^T$$

设32 b 并行CRC－32 电路的初始状态是$(R_{31}, R_{30}, \cdots,$

R_0), 输入序列是(d_{31}, d_{30}, …, d_0) (其中 d_{31} 是输入的第 1 比特码字), 输出的 CRC-32 码是(C_{31}, C_{30}, …, C_0)。利用图 3 中的 LFSR2 结构作为编码电路, 根据式(3) 和 F^{32} 的值, 可以得出 32 比特并行 CRC-32 编码器的组合逻辑表达式。设 $S_j = R_j \oplus d_j$, 从 F^{32} 的值可以知道 F(31) = 0XF7011641, 将这个值展开成二进制的表示形式, 则可以很容易得到 CRC-32 码中 C_0 的值, 结果如下:

$$C_0 = S_{31} \oplus S_{30} \oplus S_{29} \oplus S_{28} \oplus S_{26} \oplus S_{25} \oplus S_{24} \oplus S_{16} \oplus S_{12} \oplus S_{10} \oplus S_9 \oplus S_6 \oplus S_0$$

CRC-32 其余位的逻辑表达式, 可以依次类推, 限于篇幅关系, 这里不全部写出。

5 CRC 编码器的验证和实现

5.1 CRC-32 编码器的验证仿真

本文利用 Xilinx 公司 Virtex2 系列的 XC2V40-6FG256 的 FPGA 开发板对 CRC-32 码的编码和译码进行设计, 并对结果进行了功能验证。采用 ISE 开发平台和 Verilog HDL 进行设计, 验证系统的总体结构如图 4 所示, 系统的仿真波形如图 5 所示。

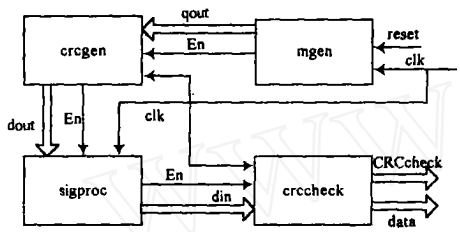

图 4 验证系统的总设计图

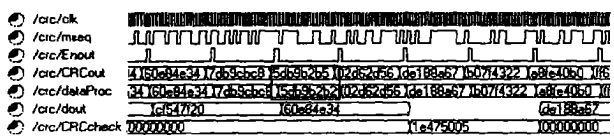

图 5 系统仿真波形

系统结构中的模块说明如下:

模块 mgen: m 序列信号发生器, 并将产生的信号做串/并转换, 输出 qout 是 32 位宽的数据流, 作为信号源;

模块 crcgen: CRC-32 编码器, 使用本文介绍的并行算法并采用图 3 中的 LFSR2 电路结构进行编码计算, 输出 dout 是已经添加了 32 位 CRC 码的数据;

模块 sigproc: 传输控制模块, 人为地引入误码, 以达到验证算法的目的;

模块 crccheck: 接收译码检错部分, 译码采用图 2 中的 LFSR 电路结构, 输出 data 是去除 CRC 码的数据, 而输出 CRCcheck 是 CRC 译码伴随式。

仿真波形图中的主要信号说明如下:

sdin: 输入到 CRC-32 编码器的序列;

CRCout: CRC-32 编码器的编码输出, 包括 CRC 码和信息码字;

dataProc: 传输控制模块的信号输出, 包含人为引入的误码;

dout: CRC-32 译码后的数据输出, 不含有 CRC 码;

CRCcheck: CRC 译码产生的伴随式。

从图 5 中可以看出, 当输入码字为 0x60E84E34H, 他的 CRC 码输出是 0x7DB9CBC8H, 这与用串行方法实现得出的结果是一致的。再来看接收译码检错部分, 在传输没有误码的情况下, CRC 的译码伴随式 CRCcheck 的输出为 0, 表示信息码输出 dout 是正确的; 如果有误码产生, 如图 4 中的方框部分所示传输出现了错误, 这时伴随式将不再等于 0, 系统可以根据这个非零值进行近一步的处理, 对错误处理本文没有涉及。至此, 从实践的观点证明了本文介绍的并行算法的正确性, 并且可以对此方便地通过 FPGA 设计加以实现。

5.2 符合 IEEE802.3 要求的 CRC-32 编码器的设计实现

为了符合 IEEE802.3 的协议要求, 对 CRC 编码器部分稍微进行改动。IEEE802.3 规定, 以太网的帧校验序列 FCS 以 CRC-32 为基础, 在编码时首先对数据的前 32 位取反, 再编码计算出 CRC-32 码后对结果取反, 最后的结果才是需要的 FCS。同时 IEEE802.3 还给设计者提供了一个测试样本, 将以下 12 B 的序列: 0xBED7_2347_6B8F_B314_5EFB_3559 循环 126 次后得到的序列作为输入数据, 经过 CRC-32 编码后得到的 FCS 是 0x94D2_54AC。

最后将稍微改动后的 CRC-32 编码器进行综合布线, 所用的 FPGA 器件就是上面提到的 XC2V40-6FG256, 再利用 IEEE802.3 提供的序列进行后仿真验证, 得到了正确的结果。从综合后的结果知道, 输入输出的延迟是 4.366 ns, 允许的最高时钟频率超过 200 MHz, 所耗费的资源如表 1 所示。

表 1 CRC-32 编码器的综合结果

Number of Slices	114
Number of Slice Flip Flops	80
Number of 4 input LUTs	213
Number of bonded IOBs	65
Number of GCLKs	1
Maximum Frequency	229.069 MHz

6 结 语

本文介绍的 CRC 并行算法, 在分析了 CRC 的串行实现电路结构的基础上, 通过系统状态方程的求解和矩阵计算, 推导出并行 CRC 码。采用这种方法可以容易地实现各种 CRC 生成多项式在不同并行度下的计算, 相对于其他并行算法更具优越性。当然同其他并行实现方法一样, 并行计算需要通过多级组合逻辑的反馈, 产生较大的门时延, 应该通过优化组合电路结构, 在最大程度上降低延迟, 使电路适用于较高的时钟频率。

(下转第 26 页)

第5～6位: 为车主在小区内的楼号;

第7位: 为车主在小区内住宅楼的单元号;

第8～9位: 为车主房间的楼层号;

第10位: 为车主在该楼层的房屋号。

因此, 如果一辆汽车的射频识别编码为0101332156, 那么该编号的含义如表1所示。

表1　车辆射频识别码举例说明

编号	代码含义
0101	车辆在小区内的档案号为0101
33	小区内的33号楼
2	2单元
15	15层
6	6号房间

3　全自动车辆管理系统在不同停车场的应用举例

3.1　收费制停车场

收费制停车场的全自动车辆管理系统的工作原理: 当车辆驶入停车场入口时, 入口处车辆感应器感应汽车到来, 利用射频装置对汽车进行识别, 待驾驶员缴费后, 自动路闸打开, 车辆通行; 在停车场出口, 当车辆进入出口时, 车辆感应器感应车辆到来, 自动打开路闸, 放行车辆。视频监控器实时监控。其工作流程图如图2所示。

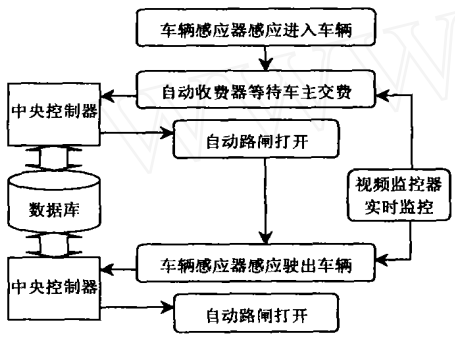

图2　收费制停车场的全自动车辆管理系统的工作流程图

3.2　非收费制停车场

非收费制停车场全自动车辆管理系统的工作原理: 当车辆驶入停车场入口时, 入口处车辆感应器感应汽车到来, 利用射频装置对汽车进行识别, 将数据传入主控中心的数据库, 由数据库进行判断, 符合条件的车辆可以通行, 并打开路闸, 不符合条件的车辆不予放行, 不打开路闸。在停车场出口, 当车辆进入出口时, 车辆感应器感应车辆到来, 自动打开路闸, 放行车辆。视频监控器进行实时监控。其工作流程图如图3所示。

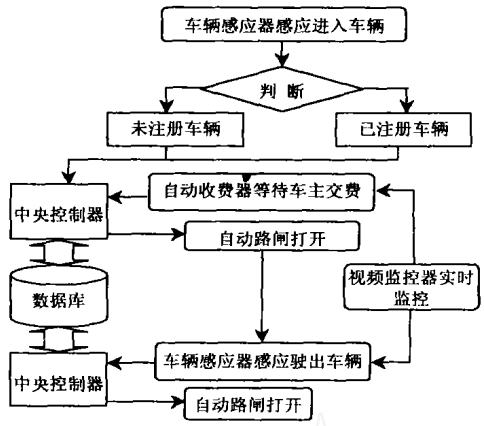

图3　非收费制停车场全自动车辆管理系统的工作流程图

4　结　语

全自动车辆管理系统涉及面广、技术性强, 对于不同的用户又有不同的要求。本文提出了一个总体设计方案, 在实践中还应根据不同用户的不同需求进行设计研发, 以满足各类用户的需求。

参　考　文　献

[1] 张成海, 张铎. 现代自动识别技术与应用 [M]. 北京: 清华大学出版社, 2003.

[2] 黄叔武, 杨一平. 计算机网络工程教程 [M]. 北京: 清华大学出版社, 1999.

[3] 赵红怡. DSP技术与应用实例 [M]. 北京: 电子工业出版社, 2003.

[4] 薛钧义, 张彦斌. MCS-51/96系列单片微型计算机及其应用 [M]. 西安: 西安交通大学出版社, 1997.

(上接第23页)

参　考　文　献

[1] Campobello G, Patane G, Russo M. Parrallel CRC Realization [J]. IEEE Transactions on Computers. 2003, 52 (10): 1312-1319.

[2] 朱荣华. 一种CRC并行计算原理及实现方法 [J]. 电子学报, 1999, (4): 4-6.

[3] 刘新宁, 王超, 胡晨, 等. 一种快速CRC算法的硬件实现方法 [J]. 电子器件, 2003, 3, 26 (1): 88-91.

[4] 张宗橙. 纠错编码原理和应用 [M]. 北京: 电子工业出版社, 2003.

[5] 田晓燕, 陈雷. 一种差错控制编译码算法的FPGA实现 [J]. 现代电子技术, 2004, 27 (19): 49-50M 52.

作者简介　陈玉泉　男, 1979年出生, 硕士研究生。研究方向为无线通信。