利用无损压缩降低循环冗余校验的错误漏检率及其电路实现

董 刚[①②] 杨海钢[①]
[①](中国科学院电子学研究所 北京 100190)
[②](中国科学院研究生院 北京 100049)

摘 要：循环冗余校验(CRC)算法在很多领域都有广泛的应用。对于确定格式的 CRC 校验码生成多项式，其错误漏检率基本为确定值。因此待检数据的长度越大，出现错误而不会被检测到的机会也就越多。为了解决这方面存在的问题，该文利用无损压缩霍夫曼算法缩短待测数据的长度，从而降低了数据出错之后不能被检测到的概率。并设计出相应的可靠性校验电路。与单纯使用 CRC 校验的方法相比，该文提出的方法可以将出错的几率下降为原来的万分之一以下。设计得到的电路模块可以作为 VLSI 中的可靠性电路模块(IP)加以利用。

关键词：可靠性电路；霍夫曼编码；CRC 编码；无损数据压缩
中图分类号：TN406 **文献标识码**：A **文章编号**：1009-5896(2010)03-0705-05
DOI:10.3724/SP.J.1146.2009.00255

Reducing Undetected Error Rate of CRC Coding Based on Lossless Compression and Implement in Circuits

Dong Gang[①②] Yang Hai-gang[①]
[①](*Institute of Electronics*, *Chinese Academy of Sciences*, 100190)
[②](*Graduate University of the Chinese Academy of Sciences*, *Beijing* 100049, *China*)

Abstract: The Cyclic Redundancy Checking (CRC) is widely used in many applications. For a certain format of the code, the error undetected probability is nearly a constant. But more bits of the information take more chances to burst errors. This paper proposes to reduce the error burst probability with lossless compression method, i.e., Huffman coding. The probability can be less then one ten-thousandth. Consequently, a new reliability circuit for VLSI with combined Huffman and CRC coding has been designed in this paper.

Key words: Reliability circuit; Huffman coding; CRC coding; Lossless compress

1 引言

随着集成电路的制造工艺逐渐向深亚微米和纳米级发展，空间中的游离粒子对电路的影响日益突出。特别是在某些重要场合，电路的可靠性成了至关重要的因素，因此集成电路的可靠性验证成为一项十分重要的任务。对于二进制数据传输中的可靠性验证，现多采用 CRC(Cyclic Redundancy Check，循环冗余校验)编码技术，对数据内容进行预编码，并保存在待传输数据之后，以便用于校验。但是由于 CRC 编码是冗余编码，因此将产生一定的成本开销。特别对于较大数量的信息，其存储开销和运算时间将不可轻视[1,2]。实际操作时，往往对全部或部分数据进行单次 CRC 计算。根据 CRC 算法的原理，这将降低校验码的可靠度，随着数据量的增长，可靠性也随之下降，因此需要采用新的手段和方法。

为了解决成本(包括时间和资源等)与可靠性的矛盾，各方面做了大量的研究工作。Minnesota 大学的研究人员 Chao Cheng 等人提出了并行 CRC 电路结构，利用数学变换手段，将大容量的数据校验工作分割成若干数目的并行操作，并使用高速电路缩短消耗的时间[3]。而 Moon Jaekyun 等研究者则从算法的生成多项式角度下手，提高了编码算法对错误样本的检测能力[4]。伊斯坦布尔 Bogazici 大学的研究者 Hocanin 利用二维的 CRC 校验算法，提高了高速通信信道数据传输的可靠性[5]。这些方法都把注意力集中在了 CRC 算法本身及其实现手段上，而本文则是把解决问题的突破点放在了 CRC 算法所处理的数据上。

本文的基本思想是通过降低 CRC 码无法检测到的错误编码(也就是错误漏检)的出现概率，来提高校验的可靠性。基于这个思路，本文在 CRC 算法中引入了霍夫曼(Huffman)编码技术。利用无损压缩算法对校验数据进行压缩，降低数据量之后，再进行 CRC 校验计算，可以很大程度上提高可靠性电路的准确性，降低错误漏检的几率，且其改善程度随着压缩率的提高呈指数增长[6,7]。

2009-03-03 收到，2009-07-16 改回
通信作者：杨海钢 yanghg@mail.ie.ac.cn

2 可靠性电路的整体结构设计

(1)压缩算法选择 通常 CRC 校验电路是由移位寄存器和异或门电路组成的，其结构和电路都已发展得较为成熟[6]。但对于大量的数据，错误漏检发生的几率还是可观的。对于某一特定的模式如 CRC32 标准，其错误漏检率为 $1/2^{32}$。当传输的数据长度为 1 Mbit 时，其无法检测出的错误共有 $2^{1024\times1024}/2^{32}$ 种，也就是漏检错误图样共有 $2^{1024\times1024}/2^{32}$ 种。因此可以认为当待检数据的长度降低时，其漏检错误图样的数目将呈指数规律下降。

而降低待测数据长度，就需要采用数据压缩算法。由于待测数据在传输或存储过程中单比特或者小数量比特的错误图样出现几率是最大的，所以必须采用无损压缩才能保证不会因压缩而遗漏所出现的错误图样。

无损压缩编码算法有很多，常见的有字典算法、RLE 算法、LZ77 算法、霍夫曼算法等。表 1 中将各种算法的特点进行了归纳和整理，由于各个算法的特点及适用性不同，因此很难找到一个适合所有算法的衡量标准。本文只对算法硬件实现的复杂程度及成本开销进行分析比较。

表 1 常见压缩算法

算法名称	压缩率	特点及适用范畴	复杂程度	硬件开销
字典算法	较高	是把文本中出现频率比较多的单词或词汇组合做成一个对应的字典列表，并用特殊代码来表示这个单词或词汇。适用于长度不同的数据串组成的数据流。	编码复杂，解码简单。	最大
RLE 算法	较低	用重复字节和重复次数的简单描述来代替重复的字节，适用于数据较为规律的数据流。	编码简单，解码复杂。	较小
LZ77 算法	较低	利用动态窗口和预读缓存器进行数据的查找压缩编码，编码规则与动态窗口内的值有关。适用于数据量很大的数据流。	编码和解码的操作都很复杂	较大
霍夫曼算法	较高	需要事先得到数据出现的概率统计，之后根据结果进行编码。适用于数据量很大的数据流。	编码简单，解码复杂。	较小

我们所要处理的数据是数量规模较大的连续二进制代码，而对于数据的处理只进行压缩，不考虑解压的操作。因此根据几种算法的特点，经比较，选择霍夫曼算法进行编码压缩。

结合霍夫曼和 CRC 两种编码方法，先对原始数据依据霍夫曼编码进行无损压缩，再对压缩后的数据用 CRC 校验方法进行计算，得出一组长度为 32 比特的校验码。

设原始数据的长度为 m，压缩后的数据长度为 n。则本方法将数据出错而漏检的情况降低了 $2^{m-32}/2^{n-32}=2^{m-n}$。

其数值要根据压缩比 n:m，以及数据长度而变化。霍夫曼编码部分需事先进行统计，得到各组数据的出现概率，然后按照概率大小排列编码，确定一个固定的编码方案。以后就可按照该方案，对后续任意数据进行编码。

本系统还采用了一个队列 FIFO(First in First out)模块，以协调霍夫曼模块与 CRC 模块之间的数据缓存问题。系统原理框图如图 1 所示。

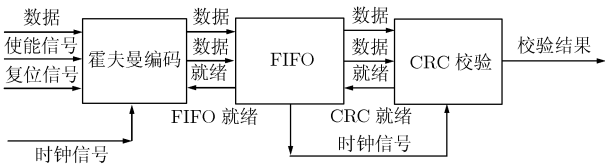


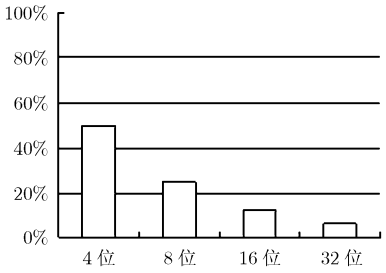
图 1 系统原理图

(2)霍夫曼编码模块设计 霍夫曼编码是一种可变长编码方式，由美国数学家 David Huffman 创立，是二叉树的一种特殊转化形式。编码原理是：将使用次数多的代码转换成长度较短的代码，而使用次数少的可使用较长的编码，并且保持编码唯一可解性。霍夫曼算法最根本的原则是：累计的和(字符的统计数字×字符的编码长度)为最小，也就是权值(字符的统计数字×字符的编码长度)的和最小。

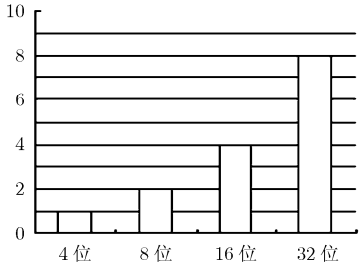
图 2 给出了不同长度霍夫曼编码方案的数据压缩率及电路开销。图中的数值是根据所要压缩的数据统计得到的。图 2(a)中横轴为编码长度，纵轴为压缩比例。图 2(b)横轴为编码长度，纵轴为硬件开销的比例坐标。

从图 2 中可以看到 16 位和 32 位编码的电路硬件开销很大。而 4 位编码的数据压缩比只有 50%左右，压缩率比较低。因此本文选择了 8 位编码方式的硬件电路与 CRC 电路进行组合使用。

利用霍夫曼编码，对较大信息量的一组数据进行编码。这些数据是由 3.3×10^5 个“0”或“1”组成的



(a) 霍夫曼电路压缩率示意图



(b) 霍夫曼电路硬件开销示意图

图 2

一串代码，按照 8 个字符 1 组，分成不同的组。经软件统计，不同组合的字符按照出现概率从大到小的顺序进行排列。该组数据中 8 bit 全零的代码出现的几率最大，占 98%以上。根据霍夫曼编码的规则，将用两比特的代码替换这种代码，因此长度将缩短为原有的$1/4$。从而将全部的代码长度大大缩短，约为原来的$1/4$ [8]。

本文的霍夫曼编码实现电路采用了一个具有 256 条分支的查找表。每条分支对应于一种“0/1”组合。实现编码的硬件描述语言可以用简单的查找表结构加以实现。因此整体面积的增加并不十分明显，在后面的测试数据中也可以看到这点。

原始数据将串行写入数据寄存器中，每个时钟周期写入一位。当一个完整的字节输入之后，控制逻辑模块将在查找表中进行查找，将原始数据翻译成新的霍夫曼编码。根据编码规则，大部分的编码长度小于原始数据，较少部分编码长度大于原始数据[9−11]。

(3)CRC 编码模块设计　CRC 编码的错误漏检率是与校验位长度密切相关的。对于$(n,\ k)$的二进制编码，令$l=n-k$，则 CRC 校验码的长度为 l。理论上不考虑其编码方式时，出现错误漏检的概率为：2^{-l}。

针对常用的 CRC32 编码方式，错误漏检率约为$1/2^{32}$。当 k 的长度小于等于 32 的时候，数据中的错误在 CRC 校验码无误的情况下，是可以百分之百检测出来的。当 k 值逐渐增加的时候，将会出现错误漏检。但 k 值并不影响错误漏检率，而是影响错误发生的情况的多少。错误发生的情况(或者称错误数据样本)数目随着位数的提高呈指数增长：$N=2^{k-32}$。当待校验数据长度很长的时候，将会存在很多种错误情况。而这些情况不会被 CRC 编码所检测出。(无法检测到的错误情况或称错误数据样本是指特定的一串 0、1 数字组合，其必定是生成多项式的整数倍，因此长度将大于等于 32。在一组代码中，依据一定几率出现。而一般运行情况下出现的几率较小。但在特殊场合对可靠性具有很高要求的情况下，是必须考虑的。)[12,13]

CRC 校验模块需要将预先算好的 CRC 校验码和待校验数据一起输入校验模块中。同时还需要一个标志信号，指明待测数据输入结束。当这个标志信号为高电平时，CRC 校验模块将会计算出新的 CRC 编码，并将其与外部输入的 CRC 编码进行比较。如果相同，会输出一个低电平，否则将输出高电平[14]。

(4)FIFO 电路设计　从前面的模块设计中可以看到，霍夫曼编码得到的码长是随机变化的，而且得到的数据是并行输出的。而在同步时钟信号的控制下，CRC 模块的输入信号是串行输入的。由并行数据转变成串行数据，需要加入一个额外的模块对数据进行缓存，以便及时地将输入的数据全部处理完毕。

本文设计了一个带有握手协议的异步队列电路(FIFO)。当霍夫曼模块计算得到编码并发送给 FIFO 的同时，还将向 FIFO 发送一个“data ready”的握手信号以及编码长度的信号。FIFO 电路将接收数据，并将存储的数据串行移位。当 FIFO 内部的存储空间充足，FIFO 将继续接收数据，同时向 CRC 模块发送数据和时钟信号。当 FIFO 内部的存储空间不足时，FIFO 将向霍夫曼模块发送指示信号，“FIFO ready”将由高电平变为低电平。霍夫曼模块的操作将暂停。

FIFO 主要由逻辑控制模块和移位寄存器组成，其面积比较小，并不会给整体电路带来过多的硬件资源消耗。

3　可靠性电路仿真分析

电路是利用 Verilog 硬件描述语言进行设计，并在 Mentor 公司的 ModelSim 软件下进行仿真[12]。用于测试的数据是一个长度为 340 kbit 的二进制码组。当全部数据中的最后一个比特输入的同时，将输入一个高电平信号，标示着全部待测数据的输入完毕。电路将进行 CRC 校验运算。仿真的波形如图 3 所示。图中横轴为仿真时间，纵向的几个波形从

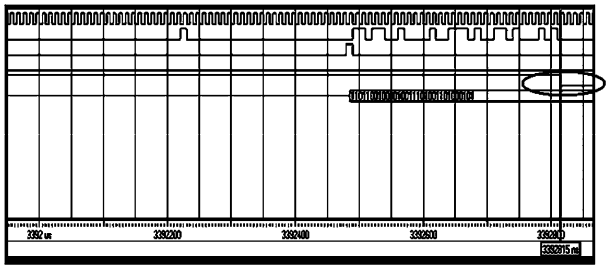


图 3 电路仿真波形图

上至下分别是时钟信号、待校验的 CRC 编码、两个控制信号和校验结果输出信号、冗余码计算结果。

仿真中的数据输入频率为 10 MHz，时钟信号频率为 100 MHz。当输入的待测数据与预先计算得到的 CRC 校验码相符时，将输出一个低电平信号。图 3 中用圆圈标出的位置即为电平变低的时刻，说明 CRC 校验结果正确。

当输入的待测数据与预先计算的 CRC 编码不符合时，输出的信号为高电平。基于前面提到的 CRC 校验原理，当出现长度短于 CRC 生成多项式的错误信息时，该错误都将被检测出来。

4 电路设计

CRC 电路通常可用来提高系统的可靠性，而附加了霍夫曼电路后，可提供更高的可靠性。假设原始数据长度为 M，压缩后的数据长度为 N，这里 N 要小于 M。CRC 计算是每个比特计算一次，则 CRC 的总计算时间约等于输入到 CRC 电路的数据比特数与时钟周期的乘积(输入预先计算的 CRC 编码时间固定为 32 个时钟周期，远小于待测数据输入的时间)。原始数据的 CRC 计算时间为 M 个时钟周期。那么不能被检测到的错误数目，在加入压缩算法与未加入压缩算法前后的比值如下式所示。

$$\frac{2^N/2^{32}}{2^M/2^{32}}=2^{N-M}=\frac{1}{2^{M-N}} \tag{1}$$

压缩后的运算时间为 $N+M/8$ 个时钟周期。这里霍夫曼电路数据输入是 8 bit 并行输入，而 CRC 校验电路数据输入方式则是单比特串行输入的。假设 $M=\alpha N$。则式(1)可以化简为 $1/2^{(\alpha-1)N}$。运算时间的比值为 $\alpha+8/(8\alpha)$。

整体设计在 Altera 公司的 Cyclone 系列芯片 EP1C3T144C6 上进行了仿真测试。在表 2 中列出了测试中得到的数据。CRC 模块电路需要消耗 90 个逻辑单元(LE)，最高运行速度为 189 MHz。而加入了压缩编码模块之后，需要消耗 103 个逻辑单元，最高运行速度为 166 MHz。校验模块的运算速度一方面与算法结构有关，另一方面也和所使用的 FPGA 芯片性能有关。若是采用 ASIC 设计，速度还会有所提升。文中所设计的校验模块，不论是单纯的 CRC 模块或是带有数据压缩的校验模块，都与文献资料中的运算时间相当，相差小于 10%[13,14]。但从错误漏检率来说，由于采用无损压缩编码，带有数据压缩的校验模块其漏检率会低得多。

该电路设计的目的是用于提高FPGA配置过程中数据传输的可靠性，要求利用硬件电路对配置数据进行校验，而且所处理的数据也具有一定的特点。根据资料统计，FPGA 配置数据内容的 85%是“0”。本文中的霍夫曼编码规则是根据配置数据内容特点而得到的，因此在处理随机数据时压缩效率不是很理想。对于表 2 中的部分随机数据，压缩率较低。目前业内已经提出自适应霍夫曼编码方法[9]，但是多应用于通信领域，主要采用软件编码的方法。对于使用固化的硬件电路进行编码的办法而言，很难实现编码规则的自适应功能。一旦确定了某种编码规则，其对应的硬件电路便无法改动。因此文献中提出的一些编码方法用硬件加以实现尚存在一定难度。

表 2 测试数据校验时间(μs)

	CRC 校验模块	带有压缩编码的校验模块
340 k 配置数据	3.4	1.26
1 M 配置数据	10	3.12
1 M 随机数据	10	7.832
1 M 随机数据(“1”占 20%，“0”占 80%)	10	4.23
1 M 随机数据(“1”占 50%，“0”占 50%)	10	7.685
1 M 随机数据(“1”占 80%，“0”占 20%)	10	8.931

配置数据长度压缩后约为原来的 1/4。不能检测到的错误数目与待测数据长度相关，减少到原有的 1/10000 以下。虽然整体电路运算速度与单纯的 CRC 校验电路相比略有下降，但整体的运算时间得到了明显的缩短。而消耗的硬件资源只增加了 10%左右。如采用其他压缩率更大的压缩算法，即 α 值更小，则可以得到更为明显的改善。但随着压缩算法的复杂度提高，其编码的硬件开销将会更大。考虑到不同的硬件环境，本文中的电路可稳定工作在 100 MHz，最高的工作频率为 140 MHz。如果需要更高速度或待测数据量更为庞大，电路的结构需要做进一步优化和改进。

参 考 文 献

[1] Godard B, Daga J M, Torres L, and Sassatelli G. Hierarchical code correction and reliability management in embedded nor flash memories [C]. European Test Symposium, Verbania, Italy, 25–29 May 2008: 84–90.

[2] Park Jihoon and Moon Jaekyun. Error-pattern-correcting cyclic codes tailored to a prescribed set of error cluster patterns [J]. *IEEE Transactions on Information Theory*, 2009, 55(4): 1747–1765.

[3] Cheng Chao and Parhi K K. High-speed parallel CRC implementation based on unfolding, pipelining, and retiming [J]. *IEEE Transactions on Circuits and Systems*, 2006, 53(10): 1017–1021.

[4] Moon Jaekyun, Park Jihoon, and Lee Jun. Cyclic redundancy check code based high-rate error-detection code for perpendicular recording [J]. *IEEE Transactions on Magnetics*, 2006, 42(5): 1626–1628.

[5] Hocanin A, Delic H, and Sarin S V. Two-dimensional CRC for efficient transmission of ATM cells over CDMA [J]. *IEEE Communications Letters*, 2000, 4(4): 131–133.

[6] Ray J and Koopman P. Efficient high hamming distance CRCs for embedded networks [C]. Dependable Systems and Networks 2006, Philadelphia, USA, 25–28 June 2006: 3–12.

[7] Kløve T, Oprisan P, and Bella B. The probability of undetected error for a class of asymmetric error detecting codes [J]. *IEEE Transactions on Information Theory*, 2005, 51(3): 1202–1205.

[8] Hosany M A and Soyjaudah K M S. A complexity study of joint and separate huffman with array codes [C]. International Conference on Networking and the International Conference on Systems, Morne, Mauritius, 23-29 Apr. 2006: 159.

[9] Lipyeow L, Wang M, and Vitter J S. SASH: A self-adaptive histogram set for dynamically changing workloads [C]. Proceedings of the 29th VLDB Conference, Berlin, Germany, 9-12 Sept. 2003, Vol. 29: 369–380.

[10] Bao Ergude, Li Wei-sheng, Fan Dong-rui, and Ma Xiao-yu. A study and implementation of the huffman algorithm based on condensed huffman table [C]. Computer Science and Software Engineering, Wuhan, China, 12–14 Dec. 2008: 42–45.

[11] Karras K and Manolakos E S. An embedded dynamically self-reconfigurable master-slaves MPSoC architecture [C]. Field Programmable Logic and Applications 2008, Heidelberg, Germany, 8–10 Sept. 2008: 431–434.

[12] Keirn Z A, Krachkovsky V Y, Haratsch E F, and Burger H. Use of redundant bits for magnetic recording: Single-parity codes and reed-solomon error-correcting code [J]. *IEEE Transactions on Magnetics*, 2004, 40(1): 225–230.

[13] Moon J, Park J, and Lee J. Cyclic redundancy check code based high-rate error-detection code for perpendicular recording [J]. *IEEE Transactions on Magnetics*, 2006, 42(5): 1626–1628.

[14] Ellingson R A. 32 bit cyclic redundancy check source code for C++. http://www.createwindow.com/programming/crc32/index.htm, 2009, May.

董　刚：　男，1979 年生，博士生，研究方向为 FPGA 芯片内部配置系统相关数字集成电路设计.

杨海钢：　男，1960 年生，研究员，博士生导师，从事数模混合信号 SoC 设计和大规模集成电路设计等方面的研究工作.