

基于 Handel-C 的伪随机数发生器的设计与实现

杨 益, 方潜生

(安徽建筑工业学院 计算机与信息工程系, 安徽 合肥 230022)

摘 要:伪随机数发生器在硬件进化、通信、信息加密甚至在其它信号处理如噪声的产生和测试数据等方面都有着非常重要的应用。结合 Handel-C 语言和 CA (Cellular Automata) 的特点, 按一定的规则数, 利用混合 CA90 和 CA150 算法规则来设计伪随机数发生器, 并用 Handel-C 语言对其进行描述。从仿真运行的结果来看, 产生的随机序列的周期非常之长, 且随机特性好, 最后通过 FTU2 下载工具最终在 FPGA 上实现了硬件电路功能, 为实现产生高速随机序列提供了一种实用的设计方法。

关键词:伪随机数发生器; 原胞自动机; Handel-C 语言; Celoxica DK2; 现场可编程门阵列

中图分类号: TP319

文献标识码: A

文章编号: 1673-629X(2006)12-0124-03

Design and Implementation of Pseudo-Random Number Generator Based on Handel-C

YANG Yi, FANG Qian-sheng

(Dept. of Computer and Info. Eng., Anhui Inst. of Architecture and Industry, Hefei 230022, China)

Abstract: Pseudo-random number generators have a lot of very important applications, such as evolvable hardware, communication, information encrypting and especially noise generating and testing datum at signal processing. Combining the characters of Handel-C and CA, this paper is according to a kind of rule numbers with the CA90/CA150 hybrid CA to design pseudo-random number generator. The pseudo-random number generator was described by Handel-C. Pseudo-random numbers has a very long period and good random characters from the simulation result. The pseudo-random number generator was implemented through Xilinx FTU2 on FPGA at last. The design method can design a high-performance pseudo-random number generator.

Key words: pseudo-random number generator; cellular automata; Handel-C language; Celoxica DK2; FPGA

0 引言

通常随机数发生器是通过一定的算法规则来设计实现的, 由于算法规则确定, 所以称为伪随机数发生器 (Pseudo-Random Number Generator, PRNG)。伪随机数发生器生成的不是绝对随机数, 是相对的, 称为伪随机数。伪随机数发生器在硬件进化、通信、信息加密甚至在其它信号处理如噪声的产生和测试数据等方面都有着非常重要的应用。

近几年, 随着计算机技术和微电子技术的迅猛发展, 可编程逻辑芯片 (FPGA) 朝着高密度、低压、低功耗方向挺进, 在计算机应用领域有着广泛的应用。这些都为伪随机数发生器在 FPGA 上直接实现奠定了坚实的基础。用传统的算法规则来设计伪随机数发生器^[1], 大多涉及大数乘

法、除法和模余等运算, 这样如果要在 FPGA 上硬件实现, 不但限制了时钟频率的提高, 而且需要消耗大量的 FPGA 逻辑资源。文中结合 Handel-C 语言和原胞自动机 (Cellular Automata, CA) 的结构特点, 用 Handel-C 描述伪随机数发生器电路, 最终在 FPGA 上实现伪随机数发生器的电路功能。

1 Handel-C 语言简介

Handel-C 是一种起源于 ISO/ANSI-C 的高级程序设计语言^[2,3], 为了支持硬件设计, Handel-C 加入了可预知的定时和确定的并发功能。通过发挥传统的软件编译技术和逻辑优化技术的优点, Handel-C 的综合器能把 Handel-C 的源代码编译成直接针对 FPGA 目标的网表 (Netlist), 这个过程无需 VHDL/Verilog 作为中间步骤, 最后用 FPGA 布线工具处理并最终下载到 FPGA 上。它允许系统开发人员和软件工程师通过某些特殊的表达式来指定某些算法是否并行, 并把它直接映射到硬件上从而加速算法的设计。在一些领域如宽带网、3G 基站和因特网骨干路由器, 这种技术是非常有用的, 因为在这些领域需要高速运行的复杂算法, 如: 加解密和压缩算法等, 并且大

收稿日期: 2006-03-22

基金项目: 安徽建筑工业学院重点青年基金项目 (200510303); 安徽建筑工业学院博士基金项目 (2003-001)

作者简介: 杨 益 (1978-), 男, 安徽安庆人, 硕士, 研究方向为 EDA 技术、计算智能; 方潜生, 教授, 博士, 研究方向为 EDA 技术、硬件进化、计算智能。

多数算法都是由 C 语言描述的。当然 Celoxica DK2 编译器也可以生成 VHDL, Verilog 等文件。Celoxica DK2 设计工具的优势在于能使复杂运算在硬件内设计实现^[4]。它包含内置设计输入、模拟与综合。DK2 可有效地扩展 FPGA 的应用,或作为 ASIC 开发至关重要的中间阶段。

Xilinx 公司的 Platform FPGAs 和 Celoxica 公司的 DK 设计工具两者结合在一起,为构建各种前沿应用所需的嵌入式系统级解决方案提供了一个很重要的新方法,这些前沿应用包括诸如基站、网络压缩、加密和流量控制以及数字电视等。

2 Cellular Automata(CA)

一个原胞自动机(CA)是一种有限状态机,有 N 个离散的单元,每个单元以离散的节拍进化,其中每个单元的下一个状态值由该单元的当前状态值和其相邻的当前状态值通过查找表(Look-Up Table, LUT)(或等价的逻辑函数)来决定,而相邻单元的状态值是独立的。 S_i^t 是 t 代第 i ($i = 0, 1, \dots, N-1$) 单元的状态值^[5,6],这样可以用 $S_0^t, S_1^t, \dots, S_{N-1}^t$ 来描述 t 代 CA 的状态值 S^t ,即 $S^t = S_0^t, S_1^t, \dots, S_{N-1}^t$ 。 η_i^t 是 t 代第 i 单元的 CA 相邻单元状态值, Φ (查找表或等价的逻辑函数)为 CA 的转换规则,可以用 $S_i^{t+1} = \Phi(\eta_i^t)$ 来更新 $t+1$ 代 CA 中每个单元的状态值,即 $S^{t+1} = \Phi(S^t)$ 。一维 CA 的相邻单元是由该单元和其相邻每个边的 r 个单元组成。 k 和 r 是 CA 的参数,其中 k 为每个单元状态的总和, r 为相邻单元的半径。对于一维 256 种状态的 CA 规则,当 $(k, r) = (2, 1)$ 时称为基本的 CA, CA 有三个相邻单元。CA30, CA90 和 CA150 都属于一维基本的 CA。

利用 CA 设计的伪随机数发生器产生的伪随机数过程如下:第一步任意写出一个规定长度的初始数(一般取经验值),即通常所说的伪随机数“种子”。第二步按照相应的规则查找表(或等价的逻辑函数)产生下一代伪随机数。第三步把产生的伪随机数作为下一代伪随机数“种子”(即作为父代),重复第二步。

3 伪随机数发生器的设计及实现

文献[1]给出在 FPGA 内利用线性反馈移位寄存器(Linear Feedback Shift Registers, LFSR)结构实现伪随机数发生器的方法。这种方法不仅结构简单,易于实现,而目所产生的伪随机序列具有周期长、随机特性好的特点。但文献[7]研究发现利用 CA30 算法规则来设计实现伪随机数发生器,它能比 LFSR 产生更好统计特性的伪随机序列,并且混合利用 CA90 和 CA150 算法规则来设计实现伪随机数发生器比 CA30 更有优势。

3.1 CA90 和 CA150

文中采用 CA90 和 CA150 混合算法规则来设计实现伪随机数发生器。CA90 和 CA150(即 CA01011010B 和

CA10010110B)算法规则的规则查找表(Φ)分别如表 1, 2 所示^[8,9],其中相邻单元的值为 η ,输出值为 $S = \Phi(\eta)$ 。

表 1 CA90 规则查找表

η	000	001	010	011	100	101	110	111
S	0	1	0	1	1	0	1	0

表 2 CA150 规则查找表

η	000	001	010	011	100	101	110	111
S	0	1	1	0	1	0	0	1

按 CA90 算法规则产生伪随机数过程如图 1 所示(伪随机数“种子”长度为 9)。

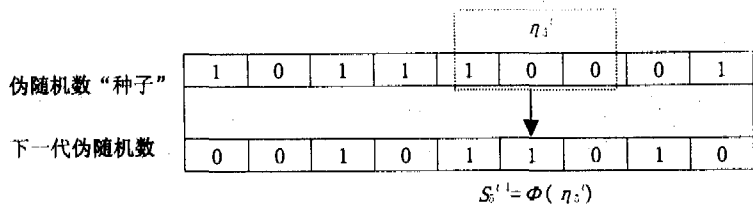


图 1 CA90 算法规则产生伪随机数过程图

CA90 算法规则等价的逻辑函数是由当前单元的相邻左右单元的状态值相异或,而 CA150 算法规则等价的逻辑函数是由相邻当前单元和左右单元的状态值相异或。下面分别给出它们的逻辑函数表达式(其中 t 表示代数, S_i^{t+1} 和 S_i^t 分别表示 CA 中第 i 个单元 $t+1$ 代和 t 代的状态值, N 表示 CA 单元的个数即随机数的长度)。

CA90 算法规则是:

当 $i = 0$ 时, $S_0^{t+1} = S_1^t$;

当 $0 < i < N-1$ 时, $S_i^{t+1} = S_{i-1}^t \oplus S_{i+1}^t$;

当 $i = N-1$ 时, $S_{N-1}^{t+1} = S_{N-2}^t$ 。

CA150 算法规则是:

当 $i = 0$ 时, $S_0^{t+1} = S_0^t \oplus S_1^t$;

当 $0 < i < N-1$ 时, $S_i^{t+1} = S_{i-1}^t \oplus S_i^t \oplus S_{i+1}^t$;

当 $i = N-1$ 时, $S_{N-1}^{t+1} = S_{N-2}^t \oplus S_{N-1}^t$ 。

利用 CA90 和 CA150 混合算法规则来设计实现伪随机数发生器,一般需要按照一定的规则数来交叉使用它们^[10],因为这样所设计伪随机数发生器产生随机序列,具有良好的统计特性和随机特性,且周期性长。文中采用了文献[10]中的规则数,如表 3 所示(这里只列出实验用到的一部分规则数),用这样的规则数可以使 CA 所对应的多项式权值最小。

表 3 中构造部分的‘1’指的是利用 CA150 算法规则,‘0’指的是利用 CA90 算法规则。例如实现一个长度为 4 的伪随机数,使用 CA90 和 CA150 算法规则情况是: CA90, CA150, CA90, CA150, 即 $S_0^{t+1} = S_1^t$, $S_1^{t+1} = S_0^t \oplus S_1^t \oplus S_2^t$, $S_2^{t+1} = S_1^t \oplus S_3^t$, $S_3^{t+1} = S_2^t \oplus S_3^t$ 。

3.2 伪随机数发生器的实现

按表 3 的规则数来交叉使用 CA90 和 CA150 算法规则设计 256 位伪随机数发生器,用 Handel-C 语言描述伪随机数发生器电路,其中 Handel-C 核心代码源程序如下:

表 3 规则数

长度	构造
4	0101
16	0001111001001000
32	00001100010001110000110000000110
64	100111010100110111101101100110010011100110110111 011001010111001
128	01001000100010000010111110111101100111000000 11000110100111110100111101001111001110000001110 011010111101111101000001000100010010
256	0001110011100100101001100011001111000111101001101 1100101010010111110010100010001101100001110000000 1011001100010000111101011111011101111010111100001 000110011010000000111000011011000100010100111101 0010101001110110010111100011110011000110010100100 11100111000

/* 定义 i 为 8 位无符号数, 初始化为零; rand_old_data[256] 为存储 256 位规则数的存储器; rand_old_data[256] 和 rand_new_data[256] 分别为存放伪随机数“种子”和下一代伪随机数的一位数组变量, 其中初始化伪随机数“种子”rand_old_data[0] = 1, 其它均为零, 初始化伪随机数“种子”为经验值 */

Do // Handel-C 描述的 256 位伪随机数发生器核心代码源程序

if(rand_old_data[i] == 0) // 选择 CA90 算法规则

if(i == 0)

rand_new_data[0] = rand_old_data[1];

else if(i == 255)

rand_new_data[255] = rand_old_data[254];

else

rand_new_data[i] = rand_old_data[i-1] ^ rand_old_data[i+1];

};

else // 选择 CA150 算法规则

if(i == 0)

rand_new_data[0] = rand_old_data[0] ^ rand_old_data[1];

else if(i == 255)

rand_new_data[255] = rand_old_data[254] ^ rand_old_data[255];

else

rand_new_data[i] = rand_old_data[i-1] ^ rand_old_data[i+1];

rand_old_data[i+1];

i++; while(i != 0);

用 Handel-C 语言描述的 256 位伪随机数发生器电路源程序在 Celoxica DK2 开发环境中调试运行, 从仿真运行的结果 (由于伪随机序列周期太长, 文中没有一一列出) 来看, 其随机序列的随机特性好, 且周期非常之长, 并最终

将 256 位伪随机数发生器在 FPGA (Xilinx 公司 Virtex-II XC2V1000 芯片) 上实现其电路功能。FPGA 硬件实现的流程如图 2 所示 (PRNG 为文件名)。

4 结束语

随着大规模集成电路工艺水平的不断提高, 在 FPGA 上实现伪随机数发生器电路功能已成为现实。文中以一定的规则数来交叉利用 CA90 和 CA150 算法规则设计伪随机数发生器电路, 并用 Handel-C 语言对电路进行描述, 最终在 FPGA 上实现了硬件电路功能, 这为实现产生高速随机数提供了一种实用的设计方法。

参考文献:

- [1] 束礼宝, 宋克柱, 王砚方. 伪随机数发生器的 FPGA 实现与研究[J]. 电路与系统学报, 2003, 8(3): 121-124.
- [2] 杨益, 方潜生, 汪力君. 基于 Handel-C 的硬件优化设计[J]. 安徽建筑工业学院学报, 2005, 13(6): 56-58.
- [3] Handel-C Language Reference Manual[S]. [s.l.]: Celoxica, 2003.
- [4] DK Design Suite User Manual[S]. [s.l.]: Celoxica, 2003.
- [5] Mitchell M, Crutchfield J P, Hraber P. Evolving Cellular Automata to Perform Computations: Mechanisms and Impediments[EB/OL]. 2005-08. <http://web.cecs.pdx.edu/mm/mech-imped.pdf>.
- [6] Crutchfield J P, Mitchell M, Das R. The evolutionary design of collective computation in cellular automata[C]// In Crutchfield J P, Schuster P K. Evolutionary Dynamics—Exploring the Interplay of Selection, Neutrality, Accident, and Function. New York: Oxford University Press, 2002: 361-411.
- [7] Shackleford B, Tanaka M, Carter R J, et al. FPGA implementation of neighborhood-of-four cellular automata random number generators[C]// Proceedings of the 2002 ACM/SIGDA tenth international symposium on Field-programmable gate arrays, February 24-26, 2002 Monterey, California, USA: [s.n.], 2002.
- [8] Cattell K, Zhang A, Sun X, et al. One-Dimensional Linear Hybrid Cellular Automata: Their Synthesis, Properties, and Applications in VLSI Testing[EB/OL]. 2006-01. <http://www.cs.uvic.ca/~mserra/CATurnain.pdf>.
- [9] Kaminsky A. Cellular Automata Based Stream Ciphers Lecture Notes[EB/OL]. 2005-12. <http://www.cs.rut.edu/ark/lectures/casc01/casc01.pdf>.
- [10] Cattell K, Muzio J. Tables of linear cellular automata for minimal weight primitive polynomials of degrees up to 300 [EB/OL]. 2006-01. <http://citeseer.ist.psu.edu/correct/605661>.

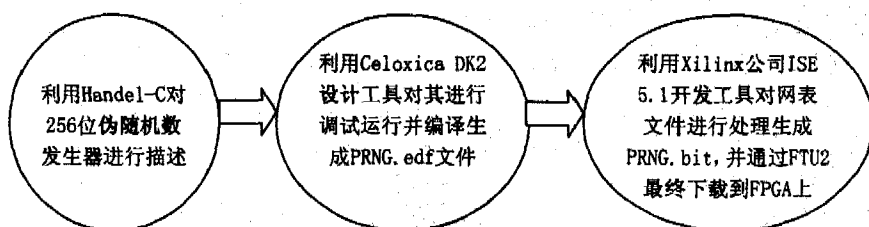


图 2 FPGA 硬件实现的流程