第36卷第9期 中国科学技术大学学报 Vol. 36, No. 9
2006年9月 JOURNAL OF UNIVERSITY OF SCIENCE AND TECHNOLOGY OF CHINA Sep. 2006

文章编号:0253-2778(2006)09-0990-05

基于 FPGA 的任意分布高速伪随机数发生器*

杜学峰,武 杰

(中国科学技术大学近代物理系快电子学实验室,安徽合肥 230026)

摘要:在 FPGA 上通过并行线性反馈移位寄存器实现高速均匀分布伪随机数,并且采用适合 FPGA 处理的"接受拒绝"的方法使输出满足用户指定的任意分布.伪随机数发生器结构简单,无需 FPGA 内嵌 DSP 模块,适用于各种类型 FPGA 上实现.

关键词:伪随机数发生器;任意分布;线性反馈移位寄存器;高斯白噪声

中图分类号:TP274 **文献标识码**:A

Arbitrary distribution high-speed pseudo-random number generator based on FPGA

DU Xue-feng, WU Jie

(*Fast Electronics Lab, Dept. of Modern Physics, University of Science and Technology of China, Hefei* 230026, *China*)

Abstract: High-speed uniform-distribution pseudo-random number was generated by parallel linear feedback shift register (LFSR), and the output followed a certain distribution by means of "acceptance and rejection" method which fitted FPGA well. The structure of the generator is simple, and it is suitable for all kinds of FPGA even without DSP modules.

Key words: pseudo-random number generator; arbitrary distribution; linear feedback shift register; white Gaussian noise

0 引言

伪随机数在信号处理中特别是信道仿真、系统模拟中广泛应用,如信道仿真中的加性高斯白噪声.伪随机数通常按照一定算法产生,可以使用 DSP[1] 或 FPGA 来实现. DSP 处理速度慢,成本高,随着信号处理的速度越来越快,对伪随机数的产生速度要求越来越高,DSP 已经无法胜任.本文充分利用了 FPGA 并行处理的优势,通过简单的适合 FPGA 处理的算法实现了 250 MHz,32 bit 高速均匀分布伪随机数发生器,并且通过"接受拒绝"的方法实现了满足任意分布的高速伪随机数.

1 原理

产生满足一定分布伪随机数的方案有很多,我们需要的是适合 FPGA 处理的方案. FPGA 具有并行计算的优势,可以胜任逻辑操作、大小比较、多路选择、简单加减法和时序逻辑等.通常 FPGA 采用查找表实现组合逻辑运算,当组合逻辑过于复杂,往往需要多级查找表完成运算,大大降低了速度[2],如乘除法等. 此时虽可以用流水线方式解决速度问题,但使用的 FPGA 资源将成倍增加.因此用FPGA

* **收稿日期**:2005-08-31;**修回日期**:2006-01-09
基金项目:国家自然科学基金(10505020)和安徽省高校"物理电子学"省级重点实验室资助.
作者简介:杜学峰,男,1978年生,博士生.研究方向:数据采集,大规模数字电路设计. E-mail:dxf@mail.ustc.edu.cn
通讯作者:武杰,博士. E-mail:wuj@ustc.edu.cn

实现伪随机数发生器，算法应该尽量简单，并且尽量不要使用乘除法等操作. 虽然有些 FPGA 内嵌 DSP 处理模块可以用于乘法等操作，但这种 DSP 模块资源比较少，往往留给 FFT，FIR 等模块使用. 考虑到 FPGA 并行处理的优势，我们最终选用线性反馈移位寄存器(linear feedback shift register，LFSR)产生均匀分布的伪随机数，然后通过“接受拒绝”的方法产生指定分布的伪随机数.

如图 1 所示，LFSR 有内部反馈(A)和外部反馈(B)两种，反馈系数 $C_i(1\leqslant i\leqslant n-1)$ 等于二进制“1”或“0”，其中的加法操作不考虑进位，用 XOR 门实现；乘法操作实际就是选择操作. 通常内部反馈方式因反馈链分散到各个寄存器之间，避免了外部反馈复杂的长反馈链，因此 LFSR 可以工作在更高的频率上. 本文采用内部反馈 LFSR，如式(1)，其中 D 表示一个时钟的延迟.

$$\begin{bmatrix} X_0(t+D)\\ X_1(t+D)\\ X_2(t+D)\\ \vdots\\ X_{n-3}(t+D)\\ X_{n-2}(t+D)\\ X_{n-1}(t+D)\end{bmatrix}=\begin{bmatrix} 0 & 0 & \cdots & 0 & 0 & 1\\ 1 & 0 & \cdots & 0 & 0 & C_1\\ 0 & 1 & \cdots & 0 & 0 & C_2\\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots\\ 0 & 0 & \cdots & 1 & 0 & C_{n-3}\\ 0 & 0 & \cdots & 0 & 1 & C_{n-2}\\ 0 & 0 & \cdots & 0 & 0 & C_{n-1}\end{bmatrix}\cdot\begin{bmatrix} X_0(t)\\ X_1(t)\\ X_2(t)\\ \vdots\\ X_{n-3}(t)\\ X_{n-2}(t)\\ X_{n-1}(t)\end{bmatrix}, \tag{1}$$

即

$$\boldsymbol{X}(t+D)=\boldsymbol{T}\boldsymbol{X}(t). \tag{2}$$

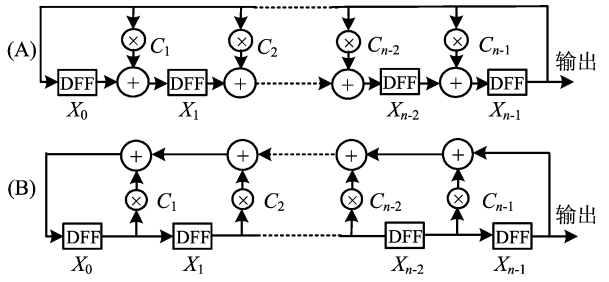

图 1　两类线性反馈移位寄存器

Fig. 1　Two kinds of linear feedback shift register

当选择合适的反馈网络 $\boldsymbol{T}$ 时，LFSR 的输出具有周期性，循环周期是 (2^n-1) 个时钟[3]，并且输出的随机性较好[5]. 通常产生 m bit 的伪随机数需要 m 个时钟周期. 为了提高输出速度，我们把式(2)迭代 m 次($m\leqslant n$)，得

$$\boldsymbol{X}(t+mD)=\boldsymbol{T}^m\boldsymbol{X}(t), \tag{3}$$

则以 $\boldsymbol{T}^m$ 作为反馈网络时，一次可以得到 m bit 伪随机数. 为了保证伪随机数的循环周期足够长，一般 n 要比较大，同时为了提高输出速度，一次并行输出更多位，m 也要足够大，这样导致反馈网络 $\boldsymbol{T}^m$ 很复杂，使得总体工作频率不但无法提高，反而会降低. 在具体的设计中需要精心挑选 n 和 m，使得反馈网络足够简单. 首先应该根据设计需求确定并行输出的位长 m_0，m 应该大于或等于 m_0；其次根据设计的随机性能需求和工作频率确定伪随机数的重复周期 N，n 应该满足 $2^n-1>N$；另外应该满足 $m\leqslant n$. n 和 m 的范围确定后还要挑选满足 LFSR 随机输出的反馈网络 $\boldsymbol{T}$. 首先应该选用系数 $C_i(1\leqslant i\leqslant n-1)$ 中“1”的个数少，同时任意两个等于“1”的系数 C_i 和 C_j，i 和 j 尽量分散的反馈网络，这样，最终的反馈网络 $\boldsymbol{T}^m$ 中，“1”的个数少，而且分散到各个行中，使得没有特别长的反馈链. “1”的个数越少，占用的资源越少；反馈链短，最大工作频率就高. 例如 250 MHz，32 bit 伪随机数发生器重复周期大于 1 d 需要 $n\geqslant 45$，$m\geqslant 32(m\leqslant n)$. m 较小时反馈网络较简单，因此 m 取 32. 当 $n=45\sim 50$，$m=32$ 时反馈网络 $\boldsymbol{T}^m$ 的复杂性对比如表 1 所示. 可见当 $n=49$ 时反馈网络 $\boldsymbol{T}^m$ 不仅“1”的个数最少，而且最长反馈链最短.

表 1　$n=45\sim 50$，$m=32$ 时反馈网络

Tab. 1　Feedback matrixes when $n=45\sim 50$，$m=32$

n	C_i 等于“1”的下标[3]	$\boldsymbol{T}^m$ 中“1”的个数	$\boldsymbol{T}^m$ 最长反馈链长度
45	44,43,41,40	357	18
46	45,44,25,24	534	27
47	46,41	166	8
48	47,46,20,19	589	27
49	48,39	123	5
50	49,48,23,22	589	29

通过 LFSR 产生的伪随机数具有平均分布，并且随机性较好[5]. 若要产生特定分布的伪随机数，有很多方法，如产生高斯白噪声的 CLT，Box-Muller 等方法[4]. 但这些方法只能产生特定分布，并且使用了乘法，在没有内嵌 DSP 处理模块的 FPGA 上很难提高速度. 我们采用最简单的“接受拒绝”的方法，如图 2 所示. 先通过并行 LFSR 产生[0，a]均匀分布的 U_1 和[0，b]均匀分布的 U_2，若 $U_2<f_X(U_1)$，则输出 U_1，否则 U_1 被拒绝，这样输出的 U_1 满足分布 $f_X(x)$. 这种方法仅需要索引和比较大小两种操作，因此很适合用 FPGA 高速实现. 输出可以达到的速度和具体的分布有关，如果伪随机数产生速度是 $M(i)$，图 2 中阴影部分面积为 s，则输出平均速度

$$M(o)=\frac{s}{ab}M(i). \tag{4}$$

因为输出的满足一定分布的伪随机数是不连续的,若需要连续输出,则可以在输出部分加入FIFO,当FIFO大小合适时,可以保证连续输出.

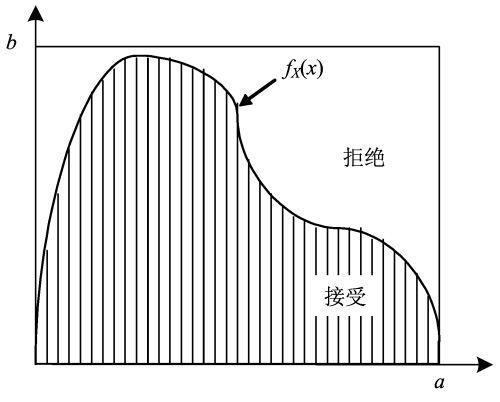

图 2 产生任意分布的"接受拒绝"方法

Fig. 2 "Acceptance and rejection" method for generating arbitrary distribution

2 具体实现

本文以一个 250 MHz,32 bit,平均分布伪随机数产生任意分布的输出为例子. 选用的产生伪随机数的 LFSR:$n=49$,$m=32$,工作频率 250 MHz 时循环周期大概是 1 d,反馈系数 C_{48},C_{39}[3],反馈网络 $\boldsymbol{T}^m$ 中每一行最多 5 个"1";用于产生分布选择的 LFSR:$n=33$,$m=24$,反馈系数 C_{32},C_{19},反馈网络 $\boldsymbol{T}^m$ 中每一行最多 3 个"1". 系统总框图如图 3 所示.

工作流程:LFSR(49,32)产生均匀分布伪随机数,取高 10 bit 作为地址从 RAM 中读取 16 bit 分布信息 b,同时 LFSR(33,24)产生均匀分布伪随机数取高 16 bit 的 a,当 a 小于 b 时接受. 通过控制模块把延迟同步后的 32 bit 伪随机数存入 FIFO. 控制模块保证 FIFO 不会溢出,同时按照一定速度输出,当输出频率小于式(4)中给出的平均速度时,同时 FIFO 长度足够大,可以保证连续输出.

本发生器速度的瓶颈在 RAM,FIFO 和比较器. 在 Altera 公司的 Cyclone 系列[6]和 Stratix 系列[7]两种 FPGA 上测试,前者(EP1C6Q240C8)器件手册中给出的 RAM 速度为 200 MHz,实际整体工作频率可以到 180 MHz 左右;后者(EP1S25F780C5) RAM 给出速度为 290 MHz,实际整体工作频率可以到 250 MHz 左右. 比较器位数越长速度越慢,我们为提高速度只用了 24 bit 中的 16 bit,保留 8 bit,这样的精度已经足够高. 提高输出速度还可以考虑增加 LFSR 数据位宽. 本发生器 LFSR 最长反馈链等于 5,两级寄存器逻辑单元中只需要增加一级 4 输入查找表. 因为用到的上述两种 FPGA 中带有寄存器的逻辑单元可以包含一个 4 输

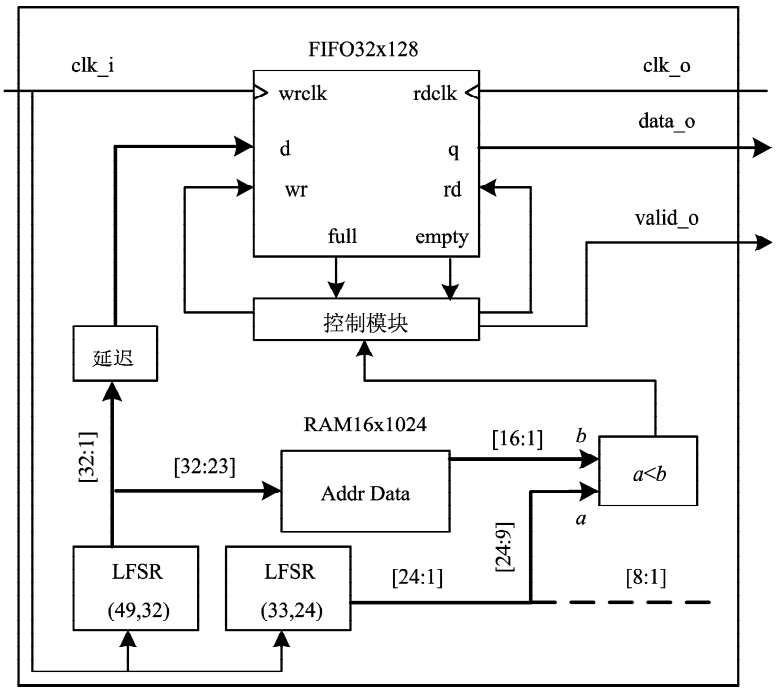

图 3 系统总框图

Fig. 3 Main block diagram of system

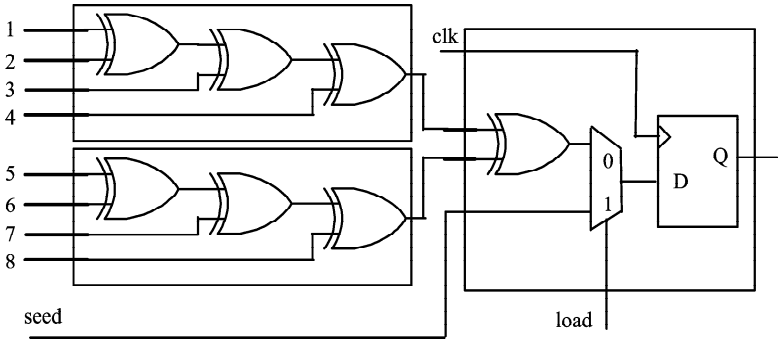

图 4　反馈链长度为 8 时 FPGA 配置结果

Fig. 4　FPGA configurate result when the feedback chain length is 8

入查找表并配置成 XOR 门(设置 LFSR 初始种子需要占用两个查找表输入 load 和 seed),所以两级寄存器之间有一级查找表时可以有 8 个输入. 如图 4 所示. 本发生器最长反馈链等于 5,因此在速度不会降低很多的前提下还可以增加 n 或 m 的大小. 当然这要参考 FPGA 中逻辑单元配置的原理,如某些 FPGA 不支持含有寄存器的逻辑单元包含查找表,必须用两级查找表才能支持 8 个输入. 每增加一级查找表,可以达到的最高工作频率都会有所降低.

本发生器能通过以下三方面保证输出随机性:

(Ⅰ)两个 LFSR 都可以初始化种子,有 $(2^{49}-1)\times(2^{33}-1)$种可能,可以通过物理方法产生真随机数设置;

(Ⅱ)LFSR 本身产生的伪随机数,重复周期可以足够长,随机性较好[5];

(Ⅲ)输入时钟 clk_i 和输出时钟 clk_o 可以不相关,如两个晶体分别产生的时钟,随时间和温度的漂移使两路时钟有一定的随机性. 这样 FIFO 读写过程不定,使得 FIFO 满的时刻不定,FIFO 满时丢掉部分数据,相当于又增加了一级平均分布的"接受拒绝".

3　测试结果

如图 5 是几种分布的 500 000 个样本测试数据,其中,LFSR 的频率 250 MHz,输出 32 bit. 平均分布连续输出速度是 240 Msps,三角分布输出 120 Msps,$\pm 4\sigma$ 的正态分布输出 75 Msps. 从测试结果看,当分布的有效面积比较小时,实际能够得到的输出速度比较低,如图 5(c)中的 $\pm 4\sigma$ 正态分布,只能达到 LFSR 速度的 31.33%. 当然,由于 LFSR 速度很高,因此最终输出速度相对较高,且本设计的精度较高. 几种设计的比较如表 2 所示.

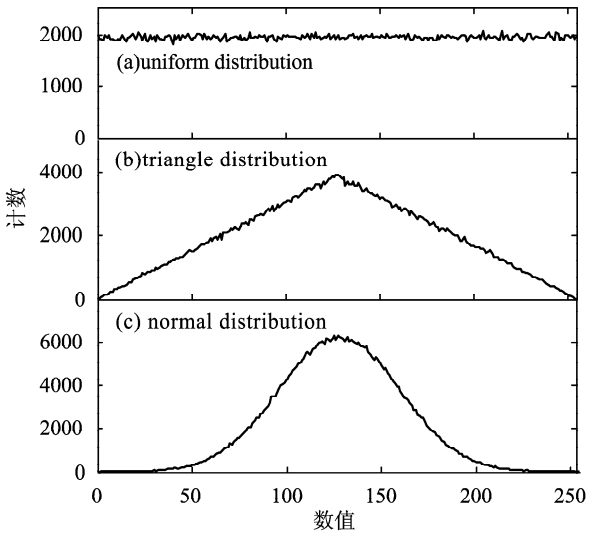

图 5　平均分布、三角分布和正态分布测试结果

Fig. 5　Test results of uniform distribution, triangle distribution and normal distribution

表 2　设计对比($\pm 4\sigma$ 标准正态分布)

Tab. 2　Comparison of designs(standard normal distribution of $\pm 4\sigma$)

	实现方法	最大输出速度/Msps	输出位长/bit	FPGA	占用资源		分布性能 *	
					逻辑单元(LE)	内部嵌入 RAM/bit	平均值(理论值 0)	标准偏差(理论值 1)
设计 A[8]	Box-Muller	24.5	10	FLEX10K100EQC240-1	文献未给出	文献未给出	0.001 009	1.437 347
设计 B[4]	改进的 Box-Muller	73.48	19	EP1M120F484C7	336	5 856	0.008 649	0.930 655
本文设计	接受拒绝	75	32	EP1S25F780C5	319	20 480	0.000 808	0.998 501

* 500 000 个样本测试结果.

4 方案的不足与改进

本方案在分布的面积占总面积比较小时,发生器最终的输出速度无法提高,相比 FIR,FFT 等,处理速度还不够快,如用到的 Stratix 系列的 FPGA,8 bit 16 阶 FIR 可以工作到 150 MHz. 如果想在 FIR 滤波后加入实时高斯白噪声,需要产生 150 Msps 正态分布伪随机数,但从表 2 可看出,本文设计只能达到 75 Msps. 实际上可以使用两个独立发生器交替输出满足要求,同时由于输出的位宽从 32 bit 减少为 8 bit,即使是两个发生器也不会占用更多的 FPGA 资源.

对于 32 bit 并行输出,本发生器中只用了 10 bit 精度作为分布的 x 轴索引,实际上是把连续的分布变为 1 024 个点的直方图分布. 如果某个位置分布变化比较快,往往 1 024 个点的精度无法满足要求,每增加一位精度需要增加一倍的 RAM 需求,因此精度不能太高. 实际上,我们可以通过线性拟合的方法来增加精度. 这个方法比较简单,只需要在 RAM 里保存一下相邻两个点的斜率,不过需要一个乘法器,如果没有内嵌 DSP 模块,发生器速度会有所降低. 如果对分布的精度要求不高还可以减少到 8 bit x 轴索引,这样表 2 中的本文设计的 RAM 资源占用将减少到 8 192 bit,如果输出位长从 32 bit 减少到 8 bit 则占用的资源会更少.

5 结论

我们在 FPGA 上通过并行 LFSR 实现高速均匀分布伪随机数的输出,并且通过适合 FPGA 处理的"接受拒绝"的方法使输出满足用户指定的任意分布. 但本文采用的方法是针对 FPGA 特别优化的,不适合用 DSP 等实现. 该发生器结构简单,通用性好,在基于 FPGA 的数据处理中可以使用,而且如果 FPGA 内嵌 DSP 处理模块,还可以通过线性拟合的方法提高精度. 另外如果对输出速度要求不高,可以选用较复杂的反馈网络,使输出随机性更好.

参考文献(References)

[1] YIN Li, MA Zhong-mei. A new method for generating high-precision digital Gaussian noise [J]. Applied Acoustics, 1996,15(3):23-25.
尹力,马忠梅. 一种快速产生数字式高精度高斯噪声的新方法[J]. 应用声学 , 1996,15(3):23-25.

[2] Gao H,Yang Y,Ma X,Dong G. Analysis of the effect of LUT size on FPGA area and delay using theoretical derivations [R]. Sixth International Symposium on Quality of Electronic Design, 2005:370-374.

[3] Alfke P. Efficient shift registers, LFSR counters, and long pseudo-random sequence generators, Xilinx application note [EB/OL]. [2005-06-10]. http://www.xilinx.com/bvdocs/appnotes/xapp052.pdf.

[4] FAN Yong-quan, Zilic Z. A novel scheme of implementing high speed AWGN communication channel emulators in FPGAs [C]// Proceedings of the 2004 International Symposium on Circuits and Systems. IEEE Press, 2004, 2:877-880.

[5] WANG Xin-cheng, SUN Hong. Research and design on high-performance pseudo-random number generator[J]. Computer Engineering and Applications, 2004,40(11):20-23.
王新成,孙宏. 高速伪随机数发生器的设计与实现[J]. 计算机工程与应用, 2004,40(11):20-23.

[6] Altera. Cyclone FPGA Family Data Sheet, version 1.2 [EB/OL]. [2003-10-10]. http://www.altera.com.cn/literature/hb/cyc/cyc_c5v1_01.pdf.

[7] Altera. Stratix Device Handbook, volume 1, version 3.3 [EB/OL]. [2005-07-10]. http://www.altera.com.cn/literature/hb/stx/stratix_handbook.pdf.

[8] Fung E, Leung K. ASIC implementation of a high speed WGNG for communication channel emulation [C]// IEEE Workshop on Signal Processing Systems, 2004. IEEE Press,2004:304-309.