Artur CHORĄŻYCZEWSKI, Maciej GĘBALA

POLITECHNIKA WROCŁAWSKA, Wyb. Wyspiańskiego 27, 50-370 Wrocław

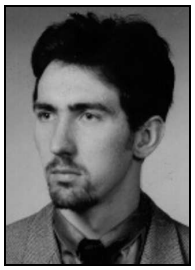
Implementacja FPGA szybkiego generatora liczb pseudolosowych

Dr inż. Artur CHORĄŻYCZEWSKI

ukończył w 1997 roku Wydział Elektroniki PWr, a w 2001 uzyskał tytuł dra nauk technicznych w dziedzinie Automatyka i Robotyka. Od 2001 pracuje na stanowisku adiunkta w Instytucie Informatyki, Automatyki i Robotyki PWr. W latach 2002-2012 realizował liczne projekty w firmie inżynierskiej bazujące na układach FPGA i technice mikroprocesorowej. Zakres zainteresowań obejmuje m.in. szybkie implementacje algorytmów oparte na układach programowalnych.

e-mail: artur.chorazyczewski@pwr.wroc.pl

Dr inż. Maciej GĘBALA



ukończył w 1995 roku studia z informatyki a następnie w 2002 uzyskał tytuł doktora nauk matematycznych w dziedzinie informatyki na Wydziale Matematyki i Informatyki Uniwersytetu Wrocławskiego. Od 2001 pracuje na stanowisku adiunkta w Instytucie Matematyki i Informatyki na Wydziale Podstawowych Problemów Techniki Politechniki Wrocławskiej. Jego zainteresowania badawcze obejmują konstruowanie i analizę algorytmów.

e-mail: maciej.gebala@pwr.wroc.pl

Streszczenie

W artykule zaprezentowano rezultaty prac nad realizacją modułu generowania ciągów pseudolosowych dla systemu kryptograficznego korzystającego z techniki mieszania pakietów. Rozważano różne implementacje uwzględniając wymaganą szybkość generowania bitów i dostępne zasoby sprzętowe. Przedstawiono rezultaty implementacji proponowanych generatorów w strukturze FPGA.

Słowa kluczowe: generator liczb pseudolosowych, szyfry strumieniowe, FPGA.

A FPGA Implementation of Fast Pseudo-Random Generator

Abstract

The paper presents results of a research on fast pseudo-random generator. The task was a part of a bigger cryptographic system developed for high speed (10Gbps) networks. The system utilized pseudo-random bits for initial increase of randomness of incoming data and for mixing bits from different packets. In the system whole process required 13 times more bits then data itself leading to 130G of pseudo-random bits per second. To achieve the expected performance FPGA technology was selected. Direct VHDL implementation of three different structures of generators was proposed, namely AES in counter mode, shrinking generator and cellular automata generator. Each of them had some shortcomings but some compromise of throughput and cryptographic strength was possible to achieve. Implementations of the last two generators were successfully used and tested in the final system. The description of generators is presented in Chapter 2. Basic performance results and FPGA structure utilization one can find in Chapter 3.

Keywords: pseudorandom generator, stream cipher, FPGA.

1. Wprowadzenie

W systemach kryptograficznych z szyframi strumieniowymi istnieje potrzeba generowania dużej ilości bitów losowych. Ilość ta jest proporcjonalna do liczby przesyłanych bitów danych, a stała proporcji zależy od algorytmu szyfrowania. W ramach realizacji projektu [1] w czasie operacji mieszania pakietów zużywanych było 13 bitów pseudolosowych na każdy bit danych. Przy zakładanych przepustowościach rzędu 1Gbps i 10Gbps oznaczało to konieczność generowania odpowiednio 13G i 130G bitów losowych. W związku z tym wybrano jako platformę implementacyjną architekturę FPGA opartą o układy Virtex 5 (XC5VLX110T) firmy Xilinx [2]. Sprzętowa realizacja obejmowała płytę z dwoma układami FPGA oraz wsparciem dla interfejsów 1Gb Ethernet i 10Gb optycznego. Rozważano trzy realizacje generatorów liczb pseudolosowych, których opisy zamieszczone są w rozdziale 2.

2. Generatory liczb pseudolosowych

Dla potrzeb jednej rundy algorytmu mieszającego blok zawierający 64 bity trzeba było wygenerować 832 bity pseudo-losowe, co zostało uwzględnione w opisach poniżej.

2.1. AES w trybie licznikowym

Ze względu na strukturę blokową implementowanego algorytmu mieszania naturalnym podejściem wydawało się zastosowanie blokowego generatora liczb pseudolosowych. Jedna z możliwych realizacji to zastosowanie AESa w trybie licznikowym [3]. Rozwiązanie to ma znane i korzystne własności losowości, jednak jego implementacja (przykładowo [4]) wymaga sporych zasobów wewnątrz układu FPGA. Dla osiągnięcia wymaganej przepustowości 1Gbps wymagana jest implementacja 13 jednostek AES, co z uwzględnieniem dupleksowej transmisji przekracza rozmiar dostępnych zasobów zastosowanych układów Virtex 5.

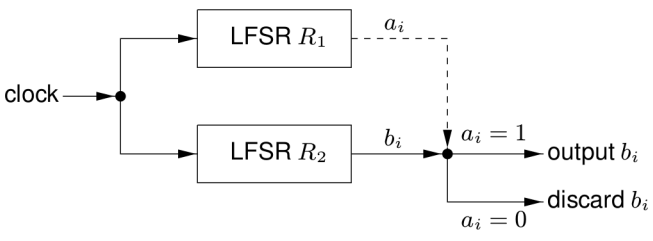
2.2. Shrinking generator

Do implementacji wykorzystano układ shrinking generator [3] z pierwszym rejestrem 65 bitowym i drugim 63 bitowym (Rys. 1), z nierozkładalnymi wielomianami generującymi postaci:

$$R_1=x^{65}+x^{18}+1 \quad (1)$$

$$R_2=x^{63}+x+1 \quad (2)$$

Shrinking generator został dokładnie przebadany zarówno pod kątem dobrych właściwości losowych jak i kryptograficznych. Stąd jego wybór wydaje się optymalny. Jedyną jego wadą jest pewna asynchroniczność w pojawianiu się bitów losowych jak i szeregowe działanie, co nie sprzyja szybkim algorytmom równoległym. Wymagało to zastosowanie układu buforowania bitów. W implementacji na FPGA zastosowano 13 shrinking generatorów inicjowanych ciągiem pseudolosowym generowanym przez jeden z nich na podstawie klucza. Dzięki temu można było osiągnąć większą wydajność generując równolegle 13 bloków po 64 bity zamiast 1 bloku zawierającego 832 bity.



Rys. 1. Struktura shrinking generatora
Fig. 1. A structure of shrinking generator

2.3. Równoległy generator komórkowy

Z uwagi na zakładaną szybkość przetwarzania (10Gbps) konieczne jest zastosowanie generatora pracującego całkowicie równolegle w stałej liczbie taktów. Temat równoległych generatorów bitów pseudolosowych jest często rozważany w kontekście wykorzystania automatów komórkowych (patrz [5-7]). Dla potrzeb rozważanego algorytmu zaprojektowano generator działający na podstawie następujących wzorów:

$$Z_i' \leftarrow Z_i \text{ xor } (Z_i \text{ or } Z_{(i+1) \bmod 832}),\ i \in \{0,\ldots,831\} \quad (3)$$

$$S_i' \leftarrow S_{(i+1) \bmod 832} \text{ xor } (S_i \text{ or } S_{(i-1) \bmod 832}),\ i \in \{0,\ldots,831\} \quad (4)$$

$$X_{i,j}' \leftarrow S_{64i+j} \text{ xor } (Z_{64i+j} \text{ and } X_{i,j}) \text{ xor } X_{(i-1) \bmod 13,j} \text{ xor } \text{xor } X_{i,(j-1) \bmod 64} \text{ xor } X_{(i+1) \bmod 13,j} \text{ xor } X_{i,(j+1) \bmod 64},\ i \in \{0,\ldots,12\},\ j \in \{0,\ldots,63\} \quad (5)$$

gdzie prim oznacza wartość w następnym takcie. Do generowania S i Z użyto reguły 30 dla jednowymiarowych automatów komórkowych, a dla X użyto reguł 64, 105, 150, 165 dla dwuwymiarowych automatów komórkowych, wyznaczanych przez odpowiednie wartości S i Z. Wynikiem jest 832 bity zawarte w X.

3. Implementacja

Przedstawione generatory zostały zaimplementowane w języku VHDL i dołączone do całego projektu zrealizowanego z wykorzystaniem środowiska *Xlilinx ISE Design Suite 10.1*. Ze względu na pracę w systemach transmisji 10Gbps należało dostosować prędkość pracy bloku do parametrów tranceivera. Jego interfejs po stronie układu FPGA ma organizację 64 bitowej szyny danych pracującej z prędkością 156.25 MHz. Takie też ograniczenie prędkościowe przyjęto dla układów w całym systemie.

Zajętość struktury FPGA przedstawiono w Tabeli 1. Jest to zajętość dla przypadku jednokierunkowej implementacji. Dla Realizacji pełnej transmisji wymagane są dwa moduły generowania bitów pseudolosowych. Dla wersji z AES przedstawiono estymację na podstawie danych katalogowych. Procent w tabeli przedstawia ilość zużytych zasobów struktury. Dane dla shrinking generatora obejmują również niezbędne struktury FIFO.

Wszystkie układy generatorów po etapie *Place&Route* syntezy układu FPGA spełniały podstawowe wymaganie prędkościowe. Zajętość struktur wskazuje jednak na możliwość zastosowania jedynie dużych matryc programowalnych.

Tab. 1. Utylizacja zasobów struktury FPGA dla poszczególnych generatorów liczb pseudolosowych. Dla AES przedstawiono estymację na podstawie danych katalogowych.
Tab. 1. A utilization of FPGA resources for each of pseudo-random generator. For an AES case estimation based on datasheet is presented.

Generator	FFs	LUTs	SLICEs
AES*		15249 – 22%	4862 – 28%
shrinking	3996 – 5.8%	3496 – 5.1%	3999 – 23%
komórkowy	2639 – 3.8%	3354 – 4.9%	2640 – 15%

4. Podsumowanie

W pracy przedstawiono wyniki z implementacji FPGA układów trzech generatorów bitów pseudolosowych. Ich zastosowanie w systemie mieszania kryptogramów nakładało pewne ograniczenia co do siły kryptograficznej, wydajności dostarczania bitów i zajętości zasobów struktury FPGA. Otrzymane wyniki wskazują na możliwość dokonywania kompromisów pomiędzy szybkością i jakością ciągu pseudolosowego. Implementacja generatorów na poziomie języka VHDL pozwoliła osiągnąć zakładaną szybkość pracy, a w docelowym rozwiązaniu ułatwiła realizację synchronizacji poszczególnych elementów składowych systemu.

5. Literatura

[1] *Implementacja nowych technologii zabezpieczenia danych transmitowanych kanałami obsługującymi dużą liczbę tuneli kryptograficznych przed wyciekami typu kleptograficznego poprzez pseudolosowe mieszanie kryptogramów i opracowanie prototypu urządzenia*, Nr projektu rozwojowego OR00013611

[2] *Virtex-5 Family Overview LX, LXT, and SXT Platforms,* DS100, www.xilinx.com

[3] A.J. Menezes, P.C. van Oorschot, S.A. Vanstone: *Handbook of Applied Cryptography*. CRC Press, ISBN: 0-8493-8523-7.

[4] [AES] AES128-C - Optimized AES Encryption - Decryption, Fast version 128-bit Data Path, CAST, Inc.

[5] B.Applebaum, Y.Ishai, E. Kushilevitz: *Cryptography by Cellular Automata or How Fast Can Complexity Emerge in Nature?*

[6] M.Tomassini, M.Perrenoud: *Nonuniform Cellular Automata for Cryptography*. Complex Systems, 12 (2000) 71-81

[7] D.H.K.Hoe, J.M.Comer, J.C.Cerda, C.D.Martinez, M.V.Shirvaikar: *Cellular Automata-Based Parallel Random Number Generators Using FPGAs*, Research Article. International Journal of Reconfigurable Computing, Volume 2012 (2012).