Jakub OLSZYNA, Wiesław WINIECKI
POLITECHNIKA WARSZAWSKA, INSTYTUT RADIOELEKTRONIKI,
ul. Nowowiejska 15/19, 00-665 Warszawa

Realizacja generatora liczb losowych dla bezprzewodowych sieci czujnikowych

Mgr inż. Jakub OLSZYNA

Absolwent Wydziału Elektroniki i Technik Informacyjnych PW, specjalności Radiokomunikacja i Techniki Multimedialne. Obecnie doktorant w Instytucie Radioelektroniki, członek IEEE. Autor lub współautor 15 publikacji naukowych. Aktualne obszary zainteresowań: rozproszone systemy pomiarowo-sterujące, sieci czujnikowe, arytmetyka modularna w kryptografii klucza publicznego.

e-mail: J.Olszyna@ire.pw.edu.pl

Prof. dr hab. inż. Wiesław WINIECKI

Prof. nzw. na Wydziale Elektroniki i Technik Informacyjnych PW. Kierownik zespołu Komputerowej Techniki Pomiarowej. Członek Komitetu Metrologii i Aparatury Naukowej PAN, wiceprezes POLSPAR, członek IEEE. Autor lub współautor 4 książek i ponad 170 publikacji naukowych. Obszary zainteresowań: systemy pomiarowe, przyrządy wirtualne, nowoczesne technologie komunikacyjne i programowe w skupionych i rozproszonych systemach pomiarowo-kontrolnych.

e-mail: W.Winiecki@ire.pw.edu.pl

Streszczenie

W artykule przedstawiono układ generatora liczb pseudolosowych dostosowany do specyfiki autonomicznych bezprzewodowych sieci czujnikowych. Realizacja podstawowych usług kryptograficznych wymaga dostarczenia liczb losowych, jednak ze względu na asymetrię zasobów (ograniczona moc zasilania i zasoby po stronie czujnikowej) konieczny jest dobór algorytmów i optymalizacja implementacji sprzętowej według kryterium mocy rozpraszanej.

Słowa kluczowe: autonomiczne bezprzewodowe sieci czujnikowe, generatory liczb losowych, arytmetyka modularna.

Realization of a random number generator for autonomous wireless sensor networks

Abstract

The paper presents a pseudo-random number generator circuit tailored to the specific properties of autonomous wireless sensor networks [1, 2]. Implementation of essential cryptographic services, like zero-knowledge proof entity authentication [3], requires delivery of random numbers. The concept of autonomous wireless sensor networks involves energy consumption from the environment, as well as efficient management of system resources. Due to the asymmetry of resources (insufficient power and computing resources on the sensor side) careful selection of the algorithm and low-power implementation of the random number generator are required. Therefore we chose to implement the BBS algorithm (Blum-Blum-Shub generator) whose security is based on the integer factorization problem and whose operation is based on modular multiplication. In order to reduce power dissipation, we decided to implement the Montgomery modular multiplication algorithm in a bit-serial fashion. Due to the proposed modifications on algorithm and architecture level, the generator is suitable for use in constrained environments like autonomous wireless sensor networks. The power consumption is only 141 µW for an Actel Igloo low-power FPGA AGLN250V2 device operating at 100 kHz (1024 bit operands).

Keywords: autonomous wireless sensor networks, random number generators, modular arithmetic.

1. Wprowadzenie

Bezprzewodowe sieci czujnikowe (ang. *Wireless Sensor Networks*) znajdują zastosowanie w wielu obszarach współczesnej metrologii [1, 2]. Systemy te charakteryzują się asymetrią zasobów tj. ograniczoną mocą obliczeniową i mocą zasilania po stronie sensorowej. Niewielkie wymiary węzłów ograniczają wybór i pojemność stosowanych baterii. Ponadto elementy sieci mogą być rozmieszczone w sposób *ad hoc*. Integracja bezprzewodowych sieci czujnikowych z istniejącą infrastrukturą komunikacyjną i wykorzystanie niezabezpieczonych fizycznie kanałów transmisji czynią kluczowym problem zapewnienia bezpieczeństwa przesyłanej informacji [3]. Kryptografia udostępnia narzędzia do rozwiązania tego problemu, jednak ograniczenia zasobów determinują możliwe do zastosowania metody zabezpieczania takich systemów (algorytmy, protokoły, układy). Realizacja podstawowych usług kryptograficznych wymaga dostarczenia liczb losowych [4]. Niestety większość implementacji generatorów liczb pseudolosowych wymaga dużych zasobów sprzętowych oraz odpowiedniej mocy zasilania. Konieczny jest zatem dobór algorytmów i optymalizacja implementacji sprzętowej według kryterium mocy rozpraszanej.

Koncepcja autonomicznych bezprzewodowych sieci czujnikowych (ang. *Autonomous Wireless Sensor Networks*) [2] zakłada pozyskiwania energii z otoczenia, za pośrednictwem dedykowanych układów zbierających energię (ang. *energy harvesting*). Autonomiczne bezprzewodowe sieci czujnikowe charakteryzują się uproszczoną konfiguracją, zredukowanymi kosztami utrzymania i zwiększoną skalowalnością (zachowaniem wydajności systemu przy zwiększaniu liczby jego elementów). Przyjęcie powyższych założeń prowadzi do zmniejszenia zużycia energii w systemie, co skutkuje niższymi kosztami jego instalacji, konfiguracji i utrzymania. Obszarami zastosowań autonomicznych bezprzewodowych sieci czujnikowych są m.in. ochrona i monitoring zdrowia, przemysł motoryzacyjny, inteligentne budynki oraz testowanie i utrzymanie budowli (mostów, budynków, tuneli).

Średnie zapotrzebowanie na energię typowego modułu sieci czujnikowej (pomiar-transmisja-tryb uśpienia) zawiera się w granicach 20 – 100 µW [2]. Techniki pozyskiwania energii z otoczenia można zgodnie z rodzajem wykorzystywanego zjawiska fizycznego przyporządkować do jednej z grup: ruch i wibracje [5], różnica temperatur [6], promieniowanie świetlne [7] lub fale radiowe [8]. Źródłem energii może być sam sygnał pomiarowy, co umożliwia integrację układu poboru mocy z układem czujnika. W zależności od wykorzystywanego zjawiska fizycznego możliwe jest uzyskanie od 0,1 µW/cm^2 do 10 mW/cm^2. Naszym celem jest zaproponowanie układu generatora liczb losowych wymagającego jak najmniejszej mocy zasilania (przy zachowaniu efektywności działania i dobrego poziomu bezpieczeństwa) tak, by nadawał się do stosowania w autonomicznych bezprzewodowych sieciach czujnikowych.

2. Generatory liczb losowych

Generator liczb losowych może mieć postać programu komputerowego lub urządzenia elektronicznego, przeznaczonego do generowania liczb losowych. Ze względu na sposób działania można wyróżnić dwa typy generatorów: generatory liczb prawdziwie losowych i generatory liczb pseudolosowych [9]. Generatory liczb prawdziwie losowych TRNG (ang. *True Random Number Generators*) działają na zasadzie ciągłego pomiaru parametrów fizycznego procesu stochastycznego. Działanie generatorów liczb pseudolosowych PRNG (ang. *Pseudo Random Number Generators*) może być opisane za pomocą deterministycznych wzorów matematycznych. Największą zaletą generatorów PRNG

jest jednak szybkość działania. Istotną ich cechą jest także bezpieczeństwo kryptograficzne generowanej sekwencji, które wiąże się z niemożnością przewidywania kolejnych generowanych wartości na podstawie analizy tych już wygenerowanych [9].

Zastosowania generatorów liczb losowych w sieciach czujnikowych obejmują:

- generację par kluczy w kryptosystemach z kluczem publicznym,
- generację kluczy w kryptosystemach z kluczem jednorazowym,
- negocjację i wybór klucza w systemach z predystrybucją paczek kluczy (zestaw kluczy zapisany na stałe w pamięci, przesyłany jest tylko indeks)
- generację liczb w protokołach identyfikacji z wiedzą zerową.

Szczególnie ostatnie z wymienionych zastosowań jest istotne, gdyż zakłada generowanie liczb losowych w każdym węźle. Ze względu na wymienioną wcześniej asymetrię zasobów nakłada to dodatkowe ograniczenia, które muszą być uwzględnione w procesie implementacji sprzętowej.

Generator Bluma-Bluma-Shuba (BBS) generuje sekwencję l słów h bitowych $Z=[z_1, z_2, \ldots, z_l]_h$ z liczby naturalnej a zgodnie z algorytmem [10]:

Wejście: n, l, s, h, gdzie: $n = pq$, $p = 3k_p + 4$, $q = 3k_q + 4$, $1 \le s \le n-1$, $nwd(s,n) = 1$

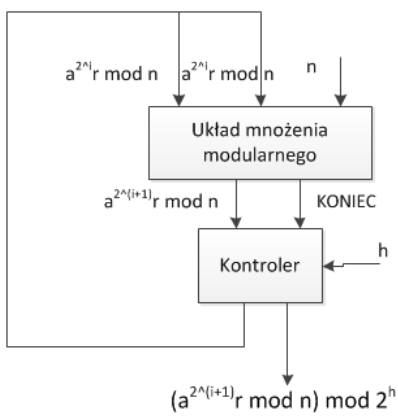
Wyjście: $Z = [z_1, z_2, ..., z_l]_h$

Krok 1. $a = s^2 \bmod n$

Krok 2. dla i=0 do l-1

a. $z_{i+1} = a^{2^i} \bmod h$

Liczba a wyliczana w kroku 1 jest kwadratem pewnej liczby naturalnej s względnie pierwszej z n. Liczba s jest nazywana ziarnem generatora - w idealnym przypadku powinna pochodzić z generatora liczb prawdziwie losowych. Generacja sekwencji na podstawie kwadratu ziarna zapewnia dobre właściwości statystyczne.



Rys. 1. Schemat blokowy generatora BBS
Fig. 1. Block diagram of the BBS generator

W oryginalnej wersji algorytmu w h wynosi 1 co oznacza, że podczas każdej iteracji generowany jest jeden losowy bit. Dobór wartości parametru h warunkuje szybkość działania generatora oraz bezpieczeństwo kryptograficzne generowanej sekwencji. Jak podano w literaturze [10], górna granica liczby nieskorelowanych bitów w każdej iteracji wynosi

$$h = \log_2(\log_2 n). \tag{1}$$

Dostatecznie długi okres sekwencji jest zapewniony, gdy p i q są liczbami silnie pierwszymi (tzn. p, q, k_p, k_q, $2k_p+1$, $2k_q+1$ są liczbami pierwszymi).

3. Implementacja generatora liczb pseudolosowych

Efektywna implementacja generatora BBS zależy od implementacji mnożenia modularnego, które jest kluczową i najbardziej kosztowną operacją w każdej iteracji algorytmu. Schemat blokowy układu generatora przedstawiono na rysunku 1. Zachowanie bezpieczeństwa kryptograficznego i odpowiednio długiego okresu generatora wymaga wykonywania operacji na długich liczbach (rzędu kilkuset bitów). Algorytm Montgomery'ego pozwala na realizację mnożenia modularnego liczb całkowitych bez konieczności wykonywania dzielenia, stosując jedynie serię sumowań i przesunięć bitowych operandów. Dla danych elementów a, b, n algorytm wyznacza

$$\bar{z} = M_{n,r}(a,b) = abr^{-1} \bmod n, \tag{2}$$

gdzie r jest pewną ustaloną liczbą naturalną. Aby uzyskać wynik mnożenia modularnego należy jeszcze wyeliminować czynnik r^{-1}. W przypadku potęgowania modularnego korzystniej jest jednak przechowywać wyniki pośrednie w reprezentacji Montgomery'ego:

$$\bar{z} = M_{n,r}(a^i r \bmod n, a^i r \bmod n) = a^{i+1} r \bmod n. \tag{3}$$

Konwersja wyniku musi zostać przeprowadzona tylko raz, po zakończeniu potęgowania

$$z = M_{n,r}(\bar{z}, 1) = a^{i+1} \bmod n. \tag{4}$$

W ogólności algorytm może być stosowany do elementów ciał liczbowych [11]. W realizacji sprzętowej, gdy operujemy liczbami binarnymi, wygodnie jest przyjąć, że $r=2^k$ gdzie k oznacza liczbę bitów w zapisie binarnym liczby n. Przy takim założeniu algorytm Montgomery'ego może zostać przedstawiony w postaci binarnej, obejmującej k iteracji [12]:

Wejście: a, b, n gdzie $0 \le a < n$, $0 \le b < n$, $2^{k-1} < n < 2^k$, $r = 2^k$, $n' = -n^{-1} \bmod 2$

Wyjście: $u = abr^{-1} \bmod n$

Krok 1. $u = 0$

Krok 2. dla i=0 do k-1

a. $u = u + a_i b + n'_0 a_i b_0 n$

b. $u = u div 2$

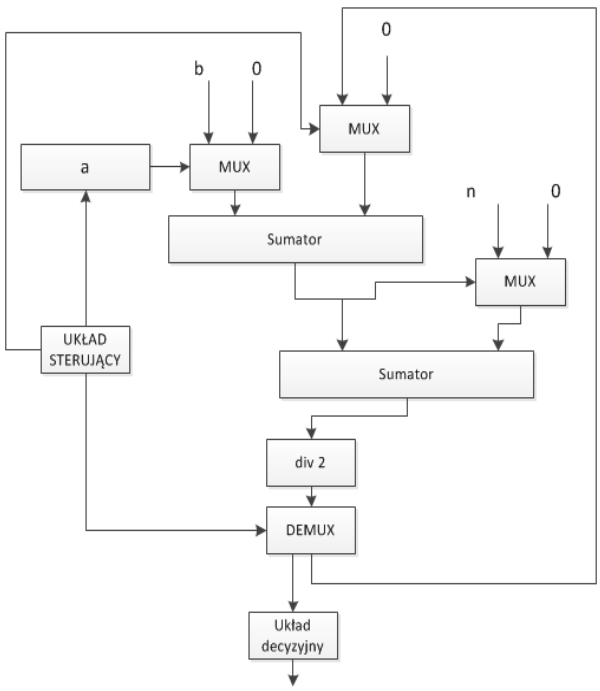
Krok 3. jeżeli $u \ge n$

a. $u = u - n$

W każdej iteracji wartość u jest mniejsza od $2n$. Jeżeli końcowa wartość u jest większa niż n konieczne jest pomocnicze odejmowanie w kroku 3a. Algorytm przetwarza dane bit po bicie, dzięki czemu nadaje się do implementacji szeregowej. Mnożenia binarne mogą zostać zastąpione przez ciąg operacji dodawania z przesunięciem. Dzięki odpowiedniemu doborowi r oraz faktowi, że n jest liczbą nieparzystą wiemy, że najmniej znaczący bit n' jest zawsze równy 1, co upraszcza krok 2a. Krok 3 wymaga porównania dwóch liczb k bitowych, co zwiększa zapotrzebowanie na moc, a także zmniejsza w sposób znaczący odporność układu na ataki czasowe (ang. *timing attacks*). W przypadku iteracyjnego podnoszenia do kwadratu krok 3 może być pominięty [12].

Przedstawiony algorytm wraz z proponowanymi modyfikacjami stanowi kluczowy komponent w realizacji generatora BBS. Niedogodnością jest fakt, że w każdej iteracji wynik mnożenia (krok 2b) wymaga transformacji pozwalającej pozbyć się czynnika r^{-1}. Optymalizacja może w tym przypadku polegać na przechowywaniu wszystkich wyników w reprezentacji Montgomery'ego (3).

Rozkład tak powstałej sekwencji liczb jest normalny, a dzięki rezygnacji z transformacji wyników uzyskuje się uproszczony układ o niższym poborze mocy.



Rys. 2. Schemat blokowy układu mnożenia modularnego alg. Montgomery'ego
Fig. 2. Block diagram of the Montgomery multiplication circuit

Do realizacji zaproponowanego układu potrzeba 2 sumatorów RCA, 2 multiplekserów, 2 rejestrów przesuwnych i licznika. Zarówno operandy, jak i moduł są każdorazowo podawane na wejście układu. Dzięki przyjętej architekturze niepotrzebne są żadne obliczenia wstępne. Moduł kontrolera uwidoczniony na rysunku 1 zawiera automat, która pozwala na wyłączanie nieużywanych fragmentów układu poprzez bramkowanie sygnału zegara. Liczba bitów dodawanych do sekwencji pseudolosowej w każdej z iteracji algorytmu jest wybierana zgodnie z wartością parametru *h*.

4. Weryfikacja

W celu wstępnego oszacowania mocy układ zaimplementowano w postaci modelu GEZEL [13], co pozwoliło oszacować moc dynamiczną, która zależy od aktywności bramek (liczby przełączeń pomiędzy wartościami '0' i '1'). Następnie dokonano konwersji modelu do postaci VHDL oraz zaimplementowano go w układzie FPGA AGLN250V2 z niskomocowej rodziny Igloo firmy Actel. Testy funkcjonalne były przeprowadzane w środowisku ModelSim, a oszacowanie rozpraszanej mocy uzyskano przy użyciu aplikacji SmartPower. Jakość generowanej sekwencji liczb pseudolosowych została dodatkowo zbadana przy użyciu testów z pakietu DieHarder [14].

Tab. 1. Szacowanie mocy układu mnożenia Montgomery'ego (k=1024)
Tab. 1. Power estimation of the Montgomery multiplication circuit (k=1024)

Liczba przełączeń	Moc @100 kHz [µW]	Moc @ 500 kHz [µW]
17 207 125	141	594

5. Podsumowanie

W artykule przedstawiono realizację generatora BBS, którego działanie jest oparte na mnożeniu modularnym. Ograniczenie mocy rozpraszanej osiągnięto dzięki zastosowaniu algorytmu Montgomery'ego wraz z zaproponowanymi modyfikacjami na poziomie algorytmu (rezygnacja z końcowego odejmowania, przeprowadzanie wszystkich obliczeń w reprezentacji Montgomery'ego, generacja *h* bitów w każdej iteracji zamiast jednego) oraz architektury układu (realizacja układu w architekturze szeregowej, redukcja liczby przełączeń, bramkowanie sygnałów zegarowych).

Model VHDL został zsyntetyzowany do wykorzystania w module FPGA z rodziny Igloo firmy Actel. Przy pracy na 1024 bitowych liczbach przy częstotliwości 100 kHz zapotrzebowanie układu na moc nie przekracza 141 µW. Dzięki proponowanym modyfikacjom zrealizowany układ może być potencjalnie zasilany z wykorzystaniem energii pobranej z otoczenia, nadaje się więc do zastosowań w autonomicznych bezprzewodowych sieciach czujnikowych o ograniczonych zasobach.

Praca została częściowo sfinansowana ze środków Samorządu Województwa Mazowieckiego w ramach stypendium 36/ES/ZS-II/W-2151.1/11 oraz przez Fundację Wspierania Rozwoju Radiokomunikacji i Technik Multimedialnych.

6. Literatura

[1] Akyildiz F., Su W., Sankarasubramaniam Y., Cayirci E.: A survey on sensor networks, Communications Magazine, t. 40 (8), s. 102– 114, 2002.
[2] Vullers R. J. M., Schaijk R. V., Visser H. J., Penders J., Hoof C. V.: Energy Harvesting for Autonomous Wireless Sensor Networks, Solid-State Circuits Magazine, t. 2 (2), s. 29–38, 2010.
[3] Winiecki W., Adamski T., Bobinski P., Łukaszewski R.: Bezpieczeństwo rozproszonych systemów pomiarowo-sterujacych (RSPS), Przegląd Elektrotechniczny, t. 84 (5), s. 220–227, 2008.
[4] Adamski T., Winiecki W., Olszyna J.: Algorithms and Circuits for Low Power Secured Sensor Networks with Asymmetric Computational Resources, Proc. XIX IMEKO World Congress, s. 627–631, 2009.
[5] Bridgelall H. R., Zoghi B.: Vibration Energy Harvesting for Disaster Asset Monitoring Using Active RFID Tags, Proceedings of the IEEE, t. 98 (9), s. 1620–1628, 2010.
[6] Bottner H., Nurnus J., Schubert A., Volkert F.: New high density micro structured thermogenerators for standalone sensor systems, Proc. ICT 2007, s. 306–309, 2007.
[7] Nasiri A., Mandic G.: Indoor Power Harvesting Using Photovoltaic Cells for Low-Power Applications, IEEE Transactions on Industrial Electronics, t. 56 (11), s. 4502–4509, 2009.
[8] Sample A., Smith J. R.: Experimental results with two wireless power transfer systems, Proc. IEEE RWS '09, s. 16–18, 2009.
[9] Czernik P., Olszyna J.: Cryptographic random number generators for low-power distributed measurement system, Proc. SPIE, t. 7502, s. 75022A, 2009.
[10] van Tilborg H. C. A, Jajodia S.: Encyclopedia of cryptography and security, Springer, 2011.
[11] Olszyna J., Winiecki W.: Niskomocowa realizacja algorytmu mnożenia modularnego Montgomery'ego dla rozproszonych systemów pomiarowo-sterujących, Przegląd Elektrotechniczny, t. 2011 (09a), s. 69–71, 2011.
[12] Walter D.: Montgomery exponentiation needs no final subtractions, Electronics Letters, t. 35 (21), s. 1831–1832, 1999.
[13] Schaumont P.: A practical introduction to hardware/software codesign, Springer, 2010.
[14] Olszyna J., Czernik P., Winiecki W.: Methods for testing random number generators in low-power distributed measurement systems, PAK, t. 56 (11), s. 1339–1341, 2010.

otrzymano / received: 28.06.2012
przyjęto do druku / accepted: 02.08.2012 *artykuł recenzowany / revised paper*