Использование начального состояния ОЗУ для генерирования истинно случайных чисел

Губчик К.В.
Кафедра вычислительных методов и программирования
Белорусский государственный университет информатики и радиоэлектроники
Минск, Республика Беларусь
e-mail: gubchikkv@gmail.com

***Аннотация*—Последовательности случайных чисел (СЧ) являются необходимым инструментом решения многих задач криптографии, имитационного моделирования, защиты авторских прав, осуществления случайного тестирования цифровых устройств и др. В работе рассматривается задача получения последовательностей истинно СЧ с использованием начального состояния ОЗУ.**

Ключевые слова: *генераторы истинно случайных чисел; физически неклонируемая функция (ФНФ)*

I. Введение

В зависимости от способа формирования числовой последовательности (ЧП) существующие генераторы СЧ можно разделить на два основных типа: генераторы псевдослучайных чисел (ГПСЧ) и генераторы истинно случайных чисел (ГИСЧ).

Преимущества ГИСЧ: невоспроизводимость, уникальность и непредсказуемость [1,2]. ГИСЧ могут реализовываться в цифровых, аналоговых и аналогово-цифровых схемах. ГИСЧ, основанные на аналоговых схемах, требуют новой технологии производства, что повышает конечную стоимость продукта и время вывода продукта на рынок. ГИСЧ, основанные на цифровых схемах, лишены этих недостатков, что уменьшает стоимость и увеличивает область применения [3].

II. Применение ФНФ для получения СЧ

В качестве источника энтропии в ГИСЧ предлагается использовать ФНФ на базе статического ОЗУ (СОЗУ), которые основаны на использовании непредсказуемых, невоспроизводимых отклонений в физической структуре интегральной схемы при ее изготовлении [4]. Поэтому реализация одинаковых по функциональности ГИСЧ будет уникальной, неповторимой и неклонируемой, что является преимуществом цифровых генераторов.

В работе [5] был предложен метод реализации ФНФ на базе СОЗУ, который основан на анализе начального состояния памяти при включении питающего напряжения. В силу того, что часть ячеек СОЗУ принимает одно из двух состояний фиксировано, а часть ячеек "плавают" под воздействием шума, последовательность битов, считанных из памяти (физический отпечаток памяти) может использоваться в качестве источника СЧ. Недостаток разработанного метода: большинство ячеек СОЗУ принимают одно из состояний чаще, чем другое.

По результатам эксперимента в [6] получены следующие данные: более 90% исследуемых ячеек устанавливались в состояние 0, менее 10% – устанавливались исключительно в единицу. И только менее 1% ячеек устанавливались равновероятно в состояние 0 или 1. Следовательно, нарушается требование абсолютной непредсказуемости данного физического отпечатка [5]. Физический отпечаток всей памяти нерационально использовать в качестве источника случайности за счет большого объема данных и малого количества энтропии. Чтобы обойти эту проблему, предлагается методика использования сигнатуры памяти вместо физического отпечатка памяти. При формировании сигнатуры происходит сжатие с потерями исходной ЧП, и как следствие уменьшается объем хранимой ЧП.

Рассчитаем вероятность появления одинаковых сигнатур и минимум энтропии, содержащийся в сигнатуре.

Если у нас имеется исходная двоичная последовательность длиной N символов, которая с помощью сигнатурного анализатора преобразуется в последовательность из m символов, то количество одинаковых сигнатур X можно выразить следующей формулой:

$$X = \frac{2^N}{2^m} = 2^{N-m} \quad (1)$$

Так как вероятность появления каждой конкретной сигнатуры одинакова, то при помощи (1) рассчитаем вероятность появления одинаковых сигнатур при различных физических отпечатках памяти.

$$p = \frac{1}{2^{N-m}} \quad (2)$$

Рассчитаем теоретический минимум энтропии, содержащийся в сигнатуре. Пусть имеется сигнатура из m битов.

$$B = \{b_1, b_2, ...b_m\} \quad (3)$$

Вероятность установки i-го бита в конкретное состояние нуля или единицы обозначим (p_{i0}, p_{i1}), где

p_{i0} – вероятность установки *i*-го бита в нулевое состояние, p_{i1} – вероятность установки *i*-го бита в единичное состояние.

Для оценки минимума энтропии требуется рассчитать энтропию наиболее вероятного значения сигнатуры.

Для этого найдем наиболее вероятные состояния всех битов сигнатуры.

$$\max = \{\max(p_{10}, p_{11}),...\max(p_{m0}, p_{m1})\} \quad (4)$$

Рассчитаем количество энтропии, которая содержится в одном символе.

$$H_i = -\sum_{k=0}^{1} p_{ik} \cdot \log_2 p_{ik},$$

где *k* – число возможных состояний, в которое может установиться ячейка ОЗУ.

Преобразуем эту формулу, раскрыв знак суммы.

$$H_i = -(p_{i0} \cdot \log_2 p_{i0} + p_{i1} \cdot \log_2 p_{i1}), \quad (5)$$

где p_{i0} , p_{i1} – вероятность появления нуля и единицы соответственно. $p_{i0} + p_{i1} = 1$

Предположим, что все биты сигнатуры принимают свое значение независимо друг от друга и воспользуемся свойством аддитивности энтропии.

Для целой сигнатуры минимальное количество энтропии будет равно сумме энтропий каждого бита наиболее вероятной сигнатуры.

$$H_s = \sum_{i=1}^{m} H_i = -\sum_{i=1}^{m} (p_{i0} \cdot \log_2 p_{i0} + p_{i1} \cdot \log_2 p_{i1}) \quad (6)$$

В сформированной сигнатуре невозможно разделить стабильную и случайную части. Если известен один или несколько физических отпечатков, то с большой степенью вероятности можно предсказать следующий физический отпечаток. Сигнатуру в отличие от физического отпечатка практически невозможно предсказать.

Это происходит за счет того, что становится невозможным определить, какие именно биты изначально являются случайными, а какие – относительно стабильными, а каждый бит сигнатуры формируется несколькими битами физического отпечатка памяти.

Полученная сигнатура может использоваться в качестве начального состояния генератора СЧ, когда не требуется высокой скорости генерации СЧ.

Предложенный ГИСЧ будет реализовываться на Digilent Nexys-3 с ПЛИС XC6LX16-CS324. Выбор ПЛИС объясняется тем, что ПЛИС выигрывают по сравнению с заказными СБИС, т. к. в специализированной схеме эксплуатационная гибкость достигается только за счет написания нового кода, а в ПЛИС есть возможность конфигурирования аппаратуры под конкретную задачу.

Например, можно будет изменять интервал, в котором требуется генерировать СЧ, возможна реализация ГДСЧ, которая каждый раз при включении, будет задавать различный диапазон генерации СЧ и алгоритм формирования сигнатур.

Для формирования сигнатур для ОЗУ можно использовать LFSR-анализатор, CRC-анализатор, адаптивный сигнатурный анализатор [7, 8]. Поэтому спроектированный ГИСЧ должен отвечать требованиям реконфигурируемости и не требовать дополнительной аппаратуры.

В работе показано, что создание ГИСЧ является актуальной проблемой, т. к. существует много областей, где требуются истинно случайные и невоспроизводимые числа.

В качестве источника случайности предложено использовать сигнатуру начального состояния ОЗУ, что позволит обеспечить высокие требования к качеству ЧП, формируемых при помощи ГИСЧ. Приведен расчет теоретических характеристик ГИСЧ, таких как минимум энтропии и вероятность появления одинаковых сигнатур при различных физических отпечатках памяти.

[1] Ярмолик В. Н. Генерирование и применение псевдослучайных сигналов в системах испытаний и контроля – Наука и техника, Минск, 1986. – 200 с.

[2] Kohlbrenner P., Gaj K. An Embedded True Random Number Generator for FPGAs – 12th international symposium on Field programmable gate arrays, New York, 2004. – p. 71-78.

[3] Vasyltsov I., Hambardzumyan E., Kim Y.-S., Karpinskyy B. Fast Digital TRNG Based on Metastable Ring Oscillator – CHES '08, Berlin, 2008. – p. 164-180.

[4] Иванюк А. А. Применение конфигурируемых генераторов импульсов для идентификации ПЛИС – Информатика №4(32), Минск, октябрь-декабрь 2011. – с. 35-46.

[5] Holcomb D. E., Burleson W.P., Fu K. Power-Up SRAM State as an Identifying Fingerprint and Source of True Random Numbers – IEEE, September 2009. – p. 1198-1210.

[6] Ярмолик В.Н., Вашинко. Физически неклонируемые функции. Информатика. – 2011. - №2. – С. 20-. 30

[7] Иванюк А. А., Петроненко Д. С. Современные неразрушающие методы и алгоритмы диагностирования оперативных запоминающих устройств – Доклады БГУИР, №4, Минск, 2004. – с. 84-92.

[8] AN 357: Error Detection & Recovery Using CRC in Altera FPGA Devices: http:// www.altera.com/literature/an/an357.pdf