Генераторы действительно случайных чисел

Губчик К.В.; Иванюк А.А.
Кафедра ВМиП, факультет ИТУ, БГУИР
e-mail: gubchikkv@gmail.com

***Аннотация* — Последовательности случайных чисел (СЧ) являются необходимым инструментом решения многих задач криптографии (например, генерация ключей, протоколы аутентификации), имитационного моделирования и др. В работе рассматривается задача получения последовательностей действительно СЧ.**

Ключевые слова: случайное число (СЧ), генератор псевдослучайных чисел (ГПСЧ), генератор действительно случайных чисел (ГДСЧ)

I. Введение

Генераторы случайных чисел можно разделить на две категории:

1. Генераторы псевдослучайных чисел (Pseudo Random Number Generator): в их основе лежат детерминированные алгоритмы, которые на основе начального значения, обычно берущегося из таймера компьютера, формируют последовательности чисел со статистическими свойствами случайности.
2. Генераторы действительно случайных чисел (True Random Number Generator): данный тип генераторов основывается на процессах, имеющий физическую природу, что делает невозможным воспроизведение точно такой же числовой последовательности в одних и тех же условиях.

Во многих случаях используют комбинацию ГПСЧ и ГДСЧ, когда при помощи ГДСЧ периодически заменяют начальное значение в ГПСЧ для достижения необходимой скорости генерации СЧ [1].

II. Генераторы псевдослучайный чисел

Любая последовательность, сгенерированная по жестко заданному алгоритму, не может считаться истинно случайной, но может аппроксимировать некоторые свойства СЧ. На сегодняшний день псевдослучайные числа используются в самых разных приложениях — от имитационного моделирования до криптографии. При этом от качества используемых ГПСЧ напрямую зависит качество получаемых результатов. Однако у ГПСЧ имеется два существенных недостатка:

1. Эти числа не являются случайными. Случайным может быть только начальное значение. Поэтому при одинаковых условиях, в которых формируется последовательность чисел, получаются абсолютно идентичные результаты.

2. Как бы не был совершенен алгоритм генерации псевдослучайных чисел - рано или поздно наступает момент, когда последовательность чисел начинает повторяться.

На данный момент широкое распространение получили следующие методы генерирования псевдослучайных чисел.

Линейный конгруэнтный метод. Применяется в простых случаях, обычно реализован в качестве стандартной функции во многих библиотеках.

Метод Фибоначчи с запаздываниями (Lagged Fibonacci generator): очень качественный, но ресурсоемкий алгоритм. К достоинствам этого метода можно отнести то, что он не требует операций умножения и все биты случайного числа равнозначны по статистическим свойствам. Недостатками являются требования большого объема памяти и большого массива чисел для запуска. Обычно вместо него используют фибоначчиевые алгоритмы "Subtract-with-borrow Generators" (SWBG) или "Linear feedback shift registers" (LFSR). Наиболее широко распространен алгоритм с использованием LFSR, который порождает псевдослучайные двоичные последовательности или М-последовательности, которые обладают следующими свойствами:

- М-последовательности являются периодическими с периодом $N = 2^n - 1$, где n – число разрядов регистра сдвига;
- количество символов, принимающих значение ‘1’, на длине одного периода М-последовательности на единицу больше, чем количество символов, принимающих значение ‘0’;
- сумма по модулю 2 любой М-последовательности с её произвольным циклическим сдвигом также является М-последовательностью [2].

Также в качестве ГПСЧ используются клеточные автоматы и генетические алгоритмы [3], [4].

III. Генераторы действительно случайных чисел

Наравне с существующей необходимостью генерировать легко воспроизводимые последовательности случайных чисел, также существует необходимость генерировать совершенно непредсказуемые и невоспроизводимые числа.

Чаще всего необходимы последовательности случайных чисел, которые равномерно и равновероятно распределены на некотором отрезке. Большинство природных процессов являются случайными, и, согласно теории математической статистики, они подчиняются различным законам распределения случайных величин. Для порождения случайных чисел, равномерно распределенных на отрезке [0..M], достаточно для некоторого случайного процесса, подчиняющегося закону равномерного распределения, ввести меру случайной величины. А затем последовательно проводить эксперименты, измерять значения случайной величины и после нормирования получать необходимую равномерную случайную последовательность чисел.

Как уже было сказано выше, используя только детерминированные алгоритмы, невозможно сгенерировать действительно случайное число. Этим обусловлено применение сочетания ГПСЧ и ГДСЧ на основе внешнего источника случайности (энтропии). ГДСЧ используется для задания начального значения для ГПСЧ, а задача последнего сводится к обеспечению спектральной и статистической чистоты последовательности [2].

В качестве источника случайности используются разнообразные естественные процессы, имеющие случайную природу. Например, тепловой шум, нажатие клавиши (активность пользователя), регистрация заряженных частиц, отклонение фазы/

флуктуационные помехи, движение расплавленного воска в лавовых лампах и многое другое [5].

IV. Оценка качества СЧ

Так как применения для СЧ могут найтись самые разные, от выборочного контроля качества изделий до криптографии, то и требования, предъявляемые к случайным последовательностям, тоже сильно варьируются. Для оценки качества полученных псевдослучайных чисел существуют специальные тесты:

1) Частотный тест.
2) Тест на последовательность одинаковых битов.
3) Спектральный тест.
4) Автокорреляционный тест.

Данные тесты проверяют сгенерированные случайные последовательности на соответствие распределению действительно случайного сигнала, отсутствие повторяющихся участков и др.

V. Предыдущие разработки

Работы по получению ГДСЧ проводились и раньше и предназначались для специализированных приложений, например, для шифрования телефонного трафика. Одной из разработок ГДСЧ для стандартной ЭВМ является компонент чипсета Firmware Hub, представленный Intel в 1999 г, основанный на тепловом шуме, схема которого представлена на рис. 1. Когда транзисторы 1 и 2 открыты, пара инверторов переводит узлы A и B в состояние «0», цепь замкнута. Когда приходит синхроимпульс, транзисторы, перейдя в состояние «1», закрываются, а выходы двух инверторов переходят в недетерминированное состояние логических «0» или «1». Эта пара битов на выходах узлов A и B является источником случайной последовательности нулей и единиц. Недостатком данной схемы с ГДСЧ является большое потребление энергии [5].

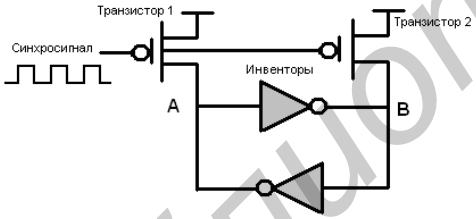

Рис.1. Схема компонента чипсета Firmware Hub

VI. Проблемы реализации ГДСЧ

Проблем, возникающих при генерации действительно случайных чисел, довольно много, что делает реализацию ГДСЧ нетривиальной. К таким недостаткам можно отнести смещение (bias), корреляция, неравномерное распределение, низкие скорости генерации, большое энергопотребление, высокая стоимость. Поэтому после генерации ДСЧ требуется постобработка, которая может производиться с помощью:

- XOR функции: этот метод уменьшает смещение между некоррелированными битами. Недостатком является уменьшение выходного потока случайных битов. Если же биты коррелированны, данный метод нельзя использовать, иначе выходное смещение значительно увеличится.

- корректора Фон Неймана: производит сбалансированное распределение единиц и нулей. Корректор Фон Неймана является очень эффективным и полностью устраняет смещение, но данный метод является весьма ресурсоемким, что уменьшает выходной поток случайных битов [1].

На данный момент есть три главных подхода к реализации ГДСЧ:

1. Кольцевые генераторы: в основном, этот метод использует отклонения в периоде сигнала синхронизации (jitter) в полностью цифровых схемах;
2. Прямое увеличение шума, что присуще аналоговым сигналам: тепловой шум, дробовой шум;
3. ГДСЧ, основанные на хаосе: метод базируется на хорошо определенных детерминистических аналоговых сигналах, в которых проявляется хаос [6].

ГДСЧ, основанные на аналоговых схемах, требуют новой платформы или технологии производства, что повышает конечную стоимость продукта и время вывода продукта на рынок. ГДСЧ, основанные на цифровых схемах, лишены этих недостатков, что уменьшает стоимость и увеличивает область применения [7]. Поэтому намечается тенденция к переходу на полностью цифровые ГДСЧ. В связи с этим, наиболее подходящей платформой для реализации ГДСЧ являются ПЛИС. ПЛИС выигрывают по сравнению с аппаратным решением, 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. Поэтому желательно сделать ГДСЧ реконфигурируемым и не требующим дополнительной аппаратуры.

За счет физических вариаций процесса изготовления между интегральными схемами реализация одинаковых по функциональности ГДСЧ будет уникальной, неповторимой и неклонируемой, что и является преимуществом цифровых генераторов.

VII. Заключение

Создание ГДСЧ является довольно актуальной проблемой, т. к. существует много областей, где требуются случайные и невоспроизводимые числа. Для реализации ГДСЧ является перспективным использование стандартных ПЛИС, что позволит сделать ГДСЧ реконфигурируемыми.

[1] P. Kohlbrenner, K. Gaj, “An Embedded True Random Number Generator for FPGAs”, IEEE, pp. 71–78.
[2] Генерация случайных чисел на микроконтроллерах. [Электронный ресурс]. – Электронные данные. – Режим доступа: http://habrahabr.ru/blogs/controllers/121849.
[3] M. G. Chegini, A. Mehrabi, Intelligent Random Sequence Generating, 2009 5th International Conference on National Computation, pp. 307-310.
[4] K.H. Tsoi, “Compact FPGA-based True and Pseudo Random Number Generators”, IEEE, pp. 1-11.
[5] Behind Intel’s New RND[Электронный ресурс]. – Электронные данные. – Режим доступа: http://spectrum.ieee.org/semiconductors/processors/behind-intels-new-randomnumber-generator/0.
[6] A. Cret, A. Suciu, “Practical Issues in Implementation TRNGs in FPGAs based on the Ring Oscillator Sampling Method”, IEEE, 2008, pp.433-438.
[7] I. Vasyltsov, E. Hambardzumyan, “Fast Digital TRNG Based on Metastable Ring OScillator”.