FPGA Based True Random Number Generators for Embedded Cryptographic Applications

Michal Varchola

CERG meeting,
March 5, 2009
Quotes

- “Any one who considers arithmetical methods of producing random digits is, of course, in a state of sin. There is no such thing as a random number there are only methods to produce random numbers”

  John Von Neumann

- “God does not play dice with the Universe.”

  Albert Einstein

- “Remember that all models are wrong; the practical question is how wrong do they have to be to not be useful.”

  George E. P. Box
Outline

- TRNG Research Motivation
- Brief History of TRNG
- TRNG Introduction
- Sources of Entropy in FPGAs
- Correctors
- Testing of TRNGs
- Practical TRNGs for FPGAs
- My research
TRNG Research Motivation

- Basic Cryptographic Primitive
  - Session keys, signature parameters, ephemeral keys, challenges, zero-knowledge protocols
- No NIST validated standard
- FPGA – Flexible reconfiguration
  - (of weak or obsolete crypto-protocols)
- Several Known Attacks
  - SSL in Early Netscape Browser (1996)
  - Windows RNG (2007)
Brief History of TRNGs

- **1946**: AT&T issued US patent 2406031
  - Large container with Black & White balls
  - Encryption of Teletype traffic

- **1955**: RAND Corp.
  - A Million Random Digits with 100,000 Normal Deviates

- **1984**: First LSI RNG
  - Fairfield, Mortenson, Coulthard
  - 67 byte per 20 seconds

- **1999**: Intel RNG
  - Jun, Kocher

- **2002**: First Designs of FPGA based TRNG
  - Fischer & Drutarovsky; Tkacik
Mathematical model of Ideal RNG

- Random numbers should be uniformly distributed in their range and independent.

- Random numbers should remain unpredictable, even if an adversary knows a large number of other random numbers (predecessors or successors of the random numbers of interest).
Practical Hardware TRNG

- Should employ the natural random process
- Should be testable and provably secure
- Has to be evaluated on how the tested sequence shares the statistical properties of ideal random sequence
- Generated sequences are completely different even though they are produced under the same operating conditions
- Knowledge of hardware structure does not help to predict random sequence
- Stochastic model should be provided
What is Random?

- Natural Phenomena Supposed to be Random:
  - Nuclear Decay, Quantum Mechanics
  - Thermal Noise, other types of Noise
  - Frequency instability of the free running oscillators

Dilbert

Tour of Accounting

Over here we have our random number generator.

Nine nine nine nine nine.

Are you sure that’s randomness?

That’s the problem with randomness, you can never be sure.
Classification of RNGs

DRNG: constant entropy given by entropy of the seed
TRNG: entropy is increasing by each generated random bit
Concept of TRNG
Block Diagram of the TRNG

Source of digital noise with its main components

Basic block diagram of the TRNG
Sources of Entropy in FPGA

- Frequency instability – Jitter
  - Clock signals (PLL, DLL)
  - Free running oscillators (RC-Actel)
- Delay Instability of Logic Elements
  - Ring Oscillators
  - Dedicated Delay Lines
- Metastability
  - Analog Behavior of Logic Gates
- Chaos (for FPAA)
Sources of Randomness - Jitter

Jitter is defined as the short-term variations of a digital signal's significant instants from their ideal positions in time.

\[
\phi = t_n - nT_0 \quad \text{...Phase jitter}
\]

\[
\phi_2' = \phi_2 - \phi_1 \quad \text{...Period jitter}
\]

\[
\phi_3'' = \phi_3' - \phi_2' \quad \text{...Cycle-to-cycle jitter}
\]
Composition of the Jitter

- Total Jitter
  - Deterministic Jitter
    - Power supply interference
  - Random Jitter
    - Gates switching interference
According Fischer et al. in FPL 2008, random jitter of Gaussian distribution accumulates slower: \( \sqrt{N_{\text{periods}}} \) while deterministic jitter accumulates faster, linearly with number of periods.
Where is origin of the jitter?

The Central Limit Theorem

Noise Model of MOS Transistor

Spectrum of Resonator
Sources of Randomness - Metastability

- Three Uncertainties:
  - If the circuit enters stable state or not
  - The state which circuit will settle into
  - Time while circuit is metastable

Output glitches.

Output temporarily remains at a metastable state and may return to either stable state, incurring additional delays.
Correctors
(or how to turn loaded dice into a fair coin)

- Von Neumann
- XOR
- Dichtl’s Good ways of post-processing
- Linear Feedback Shift Register (LFSR)
- Resilient Functions
- Hash Function, Encryption Algorithm
Basic Correctors

- **Von Neumann**
  
  \[
  00 \rightarrow \Lambda, 01 \rightarrow 0, 10 \rightarrow 1, 11 \rightarrow \Lambda
  \]
  
  - \(\Lambda\) – not used
  - Preserves correlation
  - Efficiency is \(\frac{1}{4}\) at \(p(0) = p(1) = \frac{1}{2}\)

- **XOR**
  
  - XOR operation between \(n\) input bits
  - Efficiency is \(\frac{1}{n}\)
  - Preserves correlation

- **Resilient Function**
  
  - Used in coding theory and cryptography
  - Produce one bit per \(n\)-bit input
Dichtl’s H functions

- Entropy per bit is higher in comparison to basic XOR corrector

\[
H(a_1,a_2) = a_1 \underbrace{RL(a_1,1)}_{1} \ a_2 \\
H_2(a_1,a_2) = a_1 \underbrace{RL(a_1,1)}_{1} \underbrace{RL(a_1,2)}_{2} \ a_2 \\
H_4(a_1,a_2) = a_1 \underbrace{RL(a_1,1)}_{1} \underbrace{RL(a_1,2)}_{2} \underbrace{RL(a_1,4)}_{4} \ a_2
\]
Testing of TRNGs

- Menezes et al. Handbook of Applied Cryptography (5 basic tests; DRNG)
- Diehard (DRNG)
- Tests and requirements of the NIST
  - FIPS-140 (DRNG)
  - NIST 800-22 (DRNG)
- Tests and requirements of the BSI
  - AIS 20 (DRNG)
  - AIS 31 (TRNG – entropy estimation)
DIEHARD

- Developed in 1996 by prof. Marsaglia
- Supposed to give better way of analysis in comparison to FIPS-140
- Consisted of 15 different statistical tests
- Length of sequence: 80Mbits
- 2002: Marsaglia and Tsang proposed reduced set (only 3) of DIEHARD tests. Possible to use shorter sequence
FIPS-140

- Four basic tests
  - Monobit Test, Poker Test, Runs Test, Long Run Test
- Introduced in 1994 (FIPS-140-1)
- Revision in 2001 (FIPS-140-2)
  - Changed thresholds in statistical tests
  - Statistical tests were removed
- Draft Revision in 2007 (FIPS-140-3)
  - In development now
NIST 800-22

- Introduced in 2001
- First proposal consisted of 16 tests
- Two of them were wrong
  - Kim, Umeno, Hasegawa: Corrections of the NIST statistical test suite for randomness, 2004
  - DFT Spectral Test, Lempel-Ziv Compression test
- NIST 800-22 consists of 15 tests now (without Lempel-Ziv Compression test)
- Recommended length of sequence: 1Gbit
AIS 31

- Introduced in 2001
- AIS 31 consists of 9 tests (T0-T8)
- T1-T4 tests were taken from FIPS-140
- T0-T5 tests evaluate external random numbers (after post-processing)
- T6-T8 tests evaluate digitized noise signal (before post-processing)
- T8 test: Entropy test!
- Added paradigm: Mathematical (stochastic) model should be provided
Practical TRNGs for FPGAs

- Fischer, Drutarovsky (CHES 2002)
- Tkacik (CHES 2002)
- Kohlbrenner, Gaj (FPGA 2004)
- Golic (IEEE TC 2006)
- Bucci et al. (ISCAS 2006)
- Danger et al. (NEWCAS 2007)
- Sunar et al. (IEEE TC 2007)
- Vasyltsov et al. (CHES 2008)
Authors ensure extracting at least one random bit over TQ period by proper CLJ and CLK frequency selection by mathematical model.

PLLs are usually manufactured apart from the rest logic, that provides better electro-magnetic interference immunity.

First TRNG targeted for FPGAs.

Speed: 70kbps

NIST 800-22 passed (1Gb of samples)
Practical TRNGs for FPGAs
Tkacik

- 2002: Tkacik’s proposal
  - He claimed, Motorola use this TRNG in their chips
  - DIEHARD & FIPS-140 Passed

- 2003: Dichtl’s attack
  - Not enough entropy provided by ROs

- 2003: Schindler:
  - To obtain sufficient amount of entropy, output has to be sampled 60,000 times slower as was proposed by Tkacik

- 2004: Motorola → Freescale
  - Semiconductor Division

- 2008: MCF51JM128 MCU
  - 32-bit Coldfire V1 MCU with Crypto support
  - Caution in Reference Manual regarding using RNG (by description similar to Tkacik’s) → recommendation to use another randomness sources as well
Source of entropy: RO optimized for Xilinx CLB in Virtex

Verified by NIST 800-22 (1Gbit samples)

Output rate: 600kbps
Practical TRNGs for FPGAs
Golic

- Based on so-called Fibonacci and Galois Ring Oscillators
- In depth-mathematical analysis of principle
- Capable to produce higher entropy rates than the single RO

80ns trials
Stateless RNG principle introduced
- Generation process of each single bit begins from the same conditions
- No memory
Practical TRNGs for FPGAs
Danger et al.

- Principle based on an open loop
- Entropy source: wire delays, (potentially LUT based Latches, they have longer resolving time)
- Speed: 100Mbps
- NIST 800-22 passed (1Gbit sample)
Practical TRNGs for FPGAs
Sunar et al.

- Simple and clear principle
- Deep mathematical study
- Too huge (~1000 LUTs)
- Speed: 50-80Mbps
- NIST 800-22, DIEHARD passed
- Dichtl attacked this principle: proof is irrelevant – since assumptions can never be satisfied (independent ROs, 70 GHz at the XOR input)
- Sunar proposed to sample less RO each by separated DFF, but as the result, when assuming slightly different frequencies of ROs, after XOR we will obtain statistically good bits even without jitter
Novel Metastable Ring Oscillator Principle
- Each inverter is reset into metastable state before connecting to RO chain
- Implemented for CMOS circuits, evaluated by FPGA platform
- Evaluated by AIS 31 and NIST 800-22 (1Gbit sample)
- Not sufficient results for FPGA Xilinx Virtex 2 (fluctuation of statistical parameters)
- Probably because of temperature and power supply fluctuations
- Throughput significantly increased in comparison to traditional RO
Practical TRNGs for FPGAs

Conclusion

- Fischer & Drutarovsky
  - First TRNG for FPGA, slow, manual P&R not necessary, good mathematical description, PLL needed

- Tkacik
  - Attacked by Dichtl, used by Motorola, only logic

- Kohlbrenner & Gaj
  - Optimized for Xilinx, only logic

- Golic
  - FIGARO design, only logic

- Bucci et al.
  - Stateless Principle

- Danger et al.
  - Wire Delay, very fast, only logic, manual P&R

- Sunar et al.
  - Sunar vs. Dichtl, fast, highly discussed, only logic

- Vasyltsov et al.
  - Metastable Ring Oscillator, not satisfactory results for FPGA yet, only logic
My research – past and future

- Actel ARM Platform
  - ARM7TDMI Crypto-Library porting
  - PLL based TRNG
  - Coupling of ROs
- Ring Oscillator Attack
- New Method for Entropy Extraction from logic cells (potential)
- PLL based TRNG study
- Universal Testing Board
Actel Fusion CoreMP7 Platform (past)
PLL based TRNG with RC 100MHz oscillator as clock input
- NIST 800-22 (500 Mbit sample)
- 40kbps – passes
- 1Mbps – not passed

ARM7TDMI Crypto Library (ASM optimized)

<table>
<thead>
<tr>
<th></th>
<th>Throughput [kB/s]</th>
<th>ECB mode (without key expansion)</th>
<th>Throughput [kB/s]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>Flash</td>
<td>RAM</td>
</tr>
<tr>
<td>SHA-1</td>
<td>108</td>
<td></td>
<td>119</td>
</tr>
<tr>
<td>SHA-256</td>
<td>98</td>
<td></td>
<td>100</td>
</tr>
<tr>
<td>SHA-512</td>
<td>73</td>
<td></td>
<td>86</td>
</tr>
</tbody>
</table>

Performance of SHA and AES in KB/s (25MHz Clock)
Ring Oscillator (RO) 
Principle of Operation (past)

Testing Circuit for observation of influence between two ROs depending on their mutual position inside the FPGA

Circular and Linear Layout of Two ROs
RO Experimental Results
Synchronized and Unsynchronized ROs
Ring Oscillator Attack
(past)

RO’s Noise signal sampling by DFF

Simple TRNG Based on ROs, XOR and DFF
Ring Oscillator Attack

Waveforms of the Principle of the RO Attack

Block Diagram of RO Attack Experiment
New Method of Entropy Extraction (current)
New Method of Entropy Extraction (current)
New Method of Entropy Extraction (current)
New Method of Entropy Extraction (current)
New Method of Entropy Extraction (current)
New Method of Entropy Extraction (current)
PLL based TRNG DFF analysis (future)

Observe DFF behavior more deeply, by observing its output
To be able to evaluate various TRNG principles in various FPGAs in the same operating conditions.
Thank You for Your Attention