A Self-timed Ring Based True Random Number Generator

Abdelkarim Cherkaoui, Viktor Fischer, Alain Aubert, Laurent Fesquet

A Self-timed Ring Based True Random Number Generator

Abdelkarim Cherkaoui, Viktor Fischer, Alain Aubert
Hubert Curien Laboratory
Saint-Etienne, FRANCE
Email: (abdelkarim.cherkaoui, fischer, alain.aubert)
@univ-st-etienne.fr

Laurent Fesquet
TIMA Laboratory
Grenoble, FRANCE
Email: laurent.fesquet@imag.fr

***Abstract*—Self-timed rings are oscillators in which several events can evolve evenly-spaced in time thanks to analog effects inherent to the ring stage structure. One of their interesting features is that they provide precise high-speed multiphase signals. This paper presents a true random number generator that exploits the jitter of events propagating in a self-timed ring with a high entropy. Designs implemented in Altera Cyclone III and Xilinx Virtex 5 devices provide high quality random bit sequences passing FIPS 140-1 and NIST SP 800-22 statistical tests at a high bit rate.**

I. Introduction

Random Number Generators (RNG) are basic blocks of cryptographic systems. They are used in many cryptographic primitives to generate confidential keys, challenges, padding values, to authentify protocols and even in coutermeasures against attacks. They need therefore to fulfill very strict security requirements because a weak RNG can jeopardize the whole cryptographic system security. Ideal RNGs are mathematical constructs that generate independent and uniformly distributed random numbers. Real-world RNGs are classified into Deterministic (DRNG) and True Random Numbers Generators (TRNG). DRNGs are based on complex deterministic algorithms and cryptographic functions such that their output cannot be predicted easily in a reasonable amount of time. DRNGs usually provide high bit rate data sequences that pass the standard statistical tests. However, they only guarantee practical security and do not allow a proved assessment of security. If its algorithm is known, the output of a DRNG can be theoretically predicted. Even when the algorithm is not known, but some of the generator output sequences have been recorded, its behavior during the recorded sequence can be used in future attacks. On the other hand, TRNGs usually rely on physical random processes to generate random bit streams. Their bit rate is limited by the spectrum of the underlying physical process and by the entropy extraction technique. But contrarily to DRNGs, TRNGs allow a mathematical assessment of the security: a precise modeling of the random process and the entropy extraction enable to compute the lower bound of entropy per output bit. If the minimal entropy per output bit approaches 1, then the TRNG is not manipulable and it can be assimilated to an ideal RNG.

TRNGs use physical random processes to generate random bits. Although physical random processes such as radioactive decay and thermal agitation are not exploitable directly in electronic devices, one of their consequences is the random noise present in all electronic signals. Due to this noise, the significant instants of a digital signal vary from their ideal position in time. This phenomenon, called the jitter, can be exploited to generate random numbers thanks to its random properties. *RAND Corp.* exploited the random jitter in the fifties for generating the well-known random numbers tables used worldwide by the cryptographic community back then [1]. Then, Fairfield *et. al.* were the first to propose a TRNG embedded in dedicated hardware [2]. In their design, often refered to as "Coupled-oscillators TRNG", a low frequency jittery signal samples a high frequency jittery signal to generate random numbers. Since then, the principle of sampling a jittery signal to generate random numbers have been widely reused and enhanced by the cryptographic community. Authors of [3] use the tracking jitter of a PLL (Phase Locked Loop) to generate provably random bits at a high bit rate. In [4], Golic *et. al.* propose a hybrid RNG where a jittery oscillator randomly samples the output of Galois and Fibonacci rings. Sunar *et. al.* proposed a global approach for ring oscillator based TRNGs [5]. Their design combines the jitter issued from several inverter ring oscillators to enhance the entropy harvesting.

This paper presents a novel TRNG using the jitter of events propagating in a self-timed ring to generate random numbers at a high bit-rate. First, Section 2 defines the jitter and explains how to generate random numbers using it. Then Section 3 presents the self-timed ring based true random number generator (STRNG), its architecture and working principle. Sections 4 and 5 detail the self-timed ring architecture and temporal behavior. Section 6 describes two designs implemented in Xilinx Virtex 5 and Altera Cyclone III FPGAs and provides jitter measurement results for each implementation. Section 7 evaluates the STRNG using the FIPS and NIST standard statistical tests. Finally, Section 8 concludes the paper.

II. From Jitter to Random Numbers

Jitter is a phenomenon proper to any electronic circuit involving a switching digital signal. It refers to the short-

term variations of a digital signal's significant instants from their ideal positions in time. Jitter is a consequence of several phenomena: thermal noise, shot noise, power supply noise, environmental fluctuations, etc. Jitter can be used to generate random numbers as long as its source exhibits sufficiently random properties. In fact, for randomness generation purpose, two types of noise sources are the most important: local Gaussian sources and global deterministic sources. Local Gaussian noise sources generate a random noise at the transistor level and are not influenced by external perturbations. They lead to Gaussian timing distributions (propagation delay of an inverter, oscillation period of a ring oscillator etc). This is actually a direct consequence of the central limit theorem: the distribution function of the sum of mutually independent random variables is well-approximated by a normal density function. A frequency analysis of the local random noise underlines various sub-classifications with respect to the noise frequency. The flat-band white noise represents the random unbiased uncorrelated noise source that is the most suitable for randomness generation. It comes mostly from thermal noise, i.e. the random movements of the current carriers, for example across a PN junction or at the collector or drain of a transistor. Another kind of noise, the $\frac{1}{F}$ noise (also known as the Flicker noise) is exploitable, but correlated due to its frequency dependence. This frequency dependence is much more notable for the $\frac{1}{F^2}$ Brownian noise, making it hardly usable for randomness generation. Global deterministic noise sources refer to the non-random noise sources which affect equally each component of a circuit, as for example: power supply noise, environmental fluctuations (temperature, electromagnetic emanations ...). These noise sources are dangerous and unwanted in TRNG design for many reasons. They can be predicted and manipulated providing a back door for cryptographic attacks. They can also dominate the local random sources making their measurement difficult.

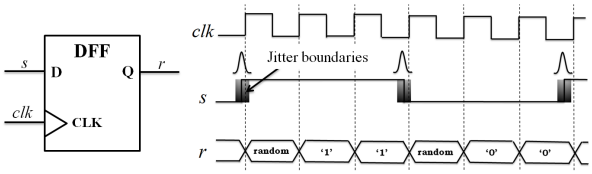


Fig. 1. Entropy harvesting using a jittery signal and a reference clock scheme

One way to generate random bits using a jittery clock is to sample this clock at its edges. Considering a signal edge arrival time as a random variable, let's define the jitter boundaries as the time interval around the mean arrival time of a signal edge that bounds 99% of this random variable draws. If the sampling happens between the jitter boundaries as shown in Fig. 1, the obtained sample has a random value. While the principle is relatively simple, its realization is not straightforward because these jitter boundaries are often very small compared to the oscillation period (usually less than 1%). The design has to meet precise timing requirements (picosecond order) so that the sampling happens between the jitter boundaries. It also needs to synchronize the sampling clock and the jittery clock to prevent the sampling signal from drifting out of the jitter boundaries.

To circumvent these issues, Sunar proposed to combine the outputs of several identical inverter ring oscillators using a XOR function [5]. The corresponding architecture is shown in Fig. 2. His objective was to fill an oscillation period of the resulting signal ψ with events. If the time lapse between two successive events is sufficiently short compared to the jitter boundaries, then sampling this signal ψ at any time would provide a random bit. Even if the relative phases of the oscillators (and thus the elapsed time between successive events) are not controlled, the idea is that, statistically, if enough ring oscillators are used, acceptable filling rates can be achieved with high probabilities. From a model standing point, the oscillation period is divided into N equally matched time intervals called urns equal to or shorter than the jitter boundaries of one ring output. The mean timings of events are supposed to be selected randomly because no assumption is made on the initial relative phases of the ring oscillators. The number of needed oscillators to fill each of the N urns with at least one event is computed using a probabilistic model: the number of uniformly random selections of N urns such that all urns are selected at least once can be approximated by $N\log(N)$.

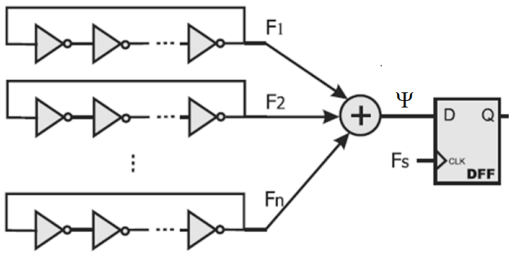


Fig. 2. Architecture of the core of the TRNG presented in [5]

The idea proposed by Sunar is certainly a major contribution to the jitter-based TRNGs theory. Nonetheless, the proposed design have been also one of the most commented by the cryptographic community. Authors of [6] remark that the generator relies mostly on the pseudo-randomness which is introduced by the phase drift between the oscillators. They demonstrate this fact using digital simulations: they show that the combined signal issued from 18 ring oscillators having slight frequency differences passes FIPS and NIST statistical tests without incorporating jitter in their simulations. Another important issue concerns the mutual dependence of rings. In fact, a full independence of rings implemented in the same single logic device is a theoretical construct that is not generally achieved in real devices. Authors of [6] show that up to 25% of a set of 118 ring oscillators can be mutually locked in an Altera Cyclone III device. If the rings are not fully independent, their mutual phases cannot be uniformly distributed, which causes a lack of entropy at the output of the TRNG.

In the following, we describe an easy way to uniformly fill the time domain with events using one self-timed ring oscillator where several events evolve evenly-spaced in time. The proposed method allows to precisely control the relative phase of the events. Locking phenomena cannot take effect because only one ring oscillator is used.

III. Self-timed Ring Based True Random Number Generator

Self-timed rings are oscillators where several events can evolve without colliding thanks to a handshake request and acknowledgment protocol. Under certain conditions, they can provide events which are evenly-spaced in time and distributed over half an oscillation period of one ring stage output. A self-timed ring provides L jittery synchronized signals $(C_i)_{1\leq i\leq L}$ having the same period T and a constant mean phase difference between them $\triangle\varphi = T/2L$ as shown in Fig. 3. A clock signal clk samples each ring stage output using a flip-flop. $(s_i)_{1\leq i\leq L}$, the obtained signals, are then combined using a XOR function. ψ is the resulting combined signal.

$$\psi = s_1 \oplus s_2 \oplus ... \oplus s_L$$

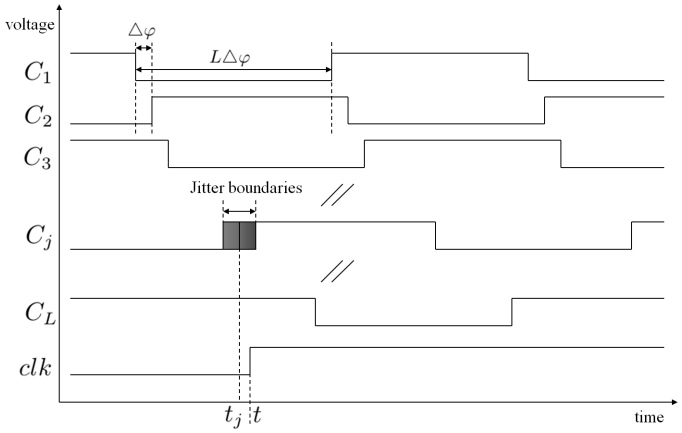


Fig. 3. Chronogram of the self-timed ring outputs

The entropy extraction principle is illustrated in Fig. 3. Since each signal C_i is sampled with the same reference clock clk, for any sampling instant t, there exists j such that $|t - t_j| \leq \frac{\triangle\varphi}{2}$, where t_j is the switching time of the signal C_j. If the jitter boundaries are larger than the phase difference $\triangle\varphi$, the signal C_j is sampled in its jitter boundaries as shown in Fig. 3. The resulting sample s_j has then a random value, and hence the output of the XOR gate is also random. The entropy of the output bit at the signal ψ is at least equal to the entropy of the sample s_j. The higher is the jitter magnitude and the lower is the phase difference $\triangle\varphi$, the higher is the entropy of the sample s_j and at the output of the TRNG.

Figure 4 shows the architecture of the STRNG (Self-timed ring based True Random Number Generator). The self-timed ring provides the jittery signals which are evenly-spaced in time. The output signals are re-indexed according to their mean arrival time (C_i and C_{i-1} are not adjacent stages). Sections 4 and 5 detail the architecture and behavior of the self-timed ring. The entropy extractor consists of two elements: the flip-flops which sample the output signals of the self-timed ring using a reference clock clk, and the XOR tree which realizes the XOR operation between the sampled outputs of the self-timed ring.

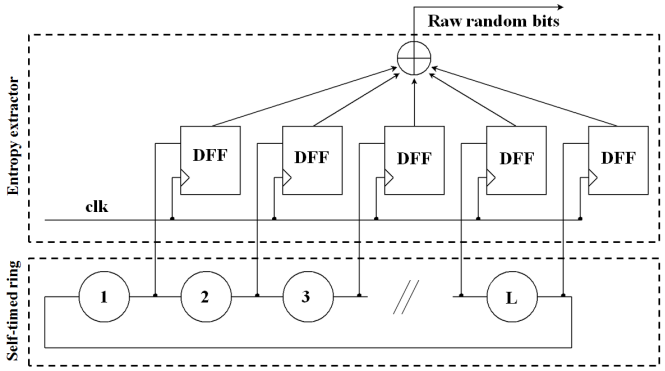


Fig. 4. STRNG core architecture

Although the theoretical concept proposed here does not require a jittery sampling clock, in practical designs, the jitter of the sampling clock enhances the entropy at the output of the TRNG. However, we do not take it into account while sizing the design (i.e. choosing the phase resolution of the self-timed ring according to its jitter magnitude). This way no assumption or constraint is made on the sampling clock (worst case scenario).

The proposed architecture suggests the potential for metastability in the flip-flop outputs. The higher is the sampling clock frequency, the higher is the probability that a flip-flop output does not resolve to a fully driven value between two successive samplings. Although this phenomenon is not discussed in the paper, previous works suggest that it can be exploited to harvest more entropy from noise ([8]), especially if these not fully driven outputs are resampled at the output of the XOR tree. However, this matter should be deeply investigated in our upcoming works.

Finally, it can be noted that the design presented in [5] relies on a probabilistic assumption: if enough ring oscillators are used, it is possible to achieve acceptable filling rates of the combined signal. On the contrary, in this design, the self-timed ring allows to precisely adjust the mean elapsed time between successive events. This time lapse can be set as short as needed, it can be thus adjusted to the jitter magnitude of a self-timed ring stage.

IV. Self-timed Ring Oscillator

Self-Timed Rings (STR) are ripple FIFOs (First In First Out memories) that have been closed to form a ring. These ripple FIFOs feature an asynchronous handshaking protocol to organize the data transfer across the structure. When closed, the FIFO retains the handshaking mechanism that ensures data ordering, but exhibits properties which are interesting for providing high precision timing signals.

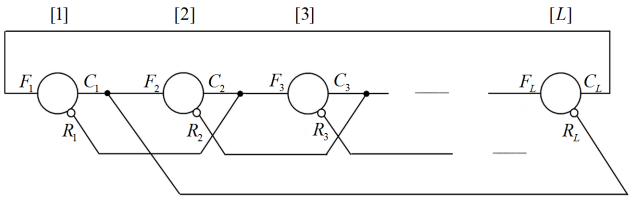
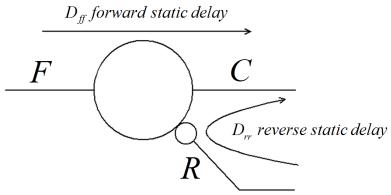


Fig. 5. Architecture of a self-timed ring

The self-timed ring structure is depicted in Fig. 5. It corresponds to a ripple FIFO as proposed by I. E. Sutherland in [7], which has been closed to form a ring of L stages. Each stage is composed of a Muller gate and an inverter. For the stage i, F_i is the forward input, R_i the reverse input, and C_i is the output. Fig. 6 shows the ring stage structure and its truth table. The forward input value is written to the output if the forward and reverse input values are different, otherwise previous output is maintained. The micropipeline stages communicate using a two-phase handshake protocol as described in [7]. In the first phase, a ring stage i sends a request signal via its output C_i, signaling to the next stage $i+1$ that data to be processed is available. In the second phase, the stage $i+1$ latches the data and sends an acknowledge signal to the previous stage i via its output C_{i+1} signifying that the data have been consumed. The same event at C_{i+1} serves also as a request signal from the stage $i+1$ to the stage $i+2$. Each 2-phase request and acknowledgment signifies an event transfer between interconnected stages. This way, data can propagate in the ring without colliding thanks to the handshake protocol. The tokens and bubbles concept is derived from the 2-phase communication protocol described bellow:

- Stage i contains a bubble if its output C_i is equal to the output of the previous stage C_{i-1}: $C_i = C_{i-1}$
- Stage i contains a token if its output C_i is different from the output of the previous stage C_{i-1}: $C_i \neq C_{i-1}$



F	R	C
0	0	C^{-1}
0	1	0
1	0	1
1	1	C^{-1}

Fig. 6. Ring stage structure and truth table

A stage that contains a token is a stage which is currently processing an event, while a stage that contains a bubble is a free stage that is ready to process an event. Knowing the stage truth table and the token and bubbles concept described above, a token propagates from the stage i to the stage $i+1$ if and only if the next stage $i+1$ contains a bubble. In the same time, a bubble propagates from the stage $i+1$ to the previous stage i if and only if the previous stage i contains a token. The condition for a token to propagate from stage i to stage $i+1$ is expressed as follows:

$$C_i \neq C_{i-1} \; and \; C_i = C_{i+1}$$

This way, tokens (or events) propagate in the ring as long as there is at least one bubble and an even number of tokens in the ring. In practice, the number of tokens N_T and the number of bubbles N_B are chosen during the ring initialization by setting the initial values of the ring stage outputs.

Independently of the initial disposition of tokens and bubbles in the ring, experiments on self-timed rings show that after a transient state, events in the ring reorganize themselves in time as the ring achieves a steady state. This steady state exhibits two oscillation modes depicted in Fig. 7: an evenly-spaced and a burst propagation mode. The evenly-spaced mode occurs when the events spread evenly all-around the ring and propagate with a constant spacing. The burst mode occurs when the events get together to form a cluster that propagates around the ring. Both these oscillation modes are stable and depend on the static parameters of the self-timed ring (e.g. the propagation delays of each ring stage). It can be noted that digital simulations do not predict these two behaviors: events propagate in a disordered manner depending on the static propagation delays and their initial disposition without necessarily clustering or spreading around the ring. In fact, digital simulations do not take into account two analog phenomena that determine the temporal behavior of the self-timed ring: the Charlie and drafting effects. Temporal behavior of self-timed rings have been widely studied in the past ([9], [10] and [11]), the following section briefly describes it and presents the features which are exploited in the STRNG principle.

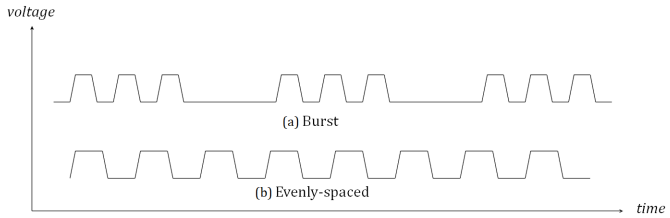


Fig. 7. Burst and evenly-spaced propagation modes in a self-timed ring

V. Temporal Behavior of Self-timed Rings

One of the major issues with ripple FIFOs was to study the loss of performances due to data clustering behaviors. Ebergen *et. al.* were the first to use Charlie diagrams as a tool for a better understanding of data movement in ripple FIFOs [12]. Charlie diagrams are used to predict the timing behavior of a Muller gate as a function of the separation time between the events that drive this gate. Winstanley *et. al.* carry on the study by closing the FIFO [9] and introducing another analog effect that helps understanding the movement of tokens (called the drafting effect).

A. The Charlie and Drafting Effects

The Charlie and drafting effect are both analog phenomena which are inherent to the self-timed ring stage structure. The

Charlie effect describes the impact of the separation time between input events on a Muller gate delay: the closer are the arrival times of inputs, the longer is this propagation delay. The drafting effect describes the impact of the elapsed time from the last output commutation on the stage propagation delay: the shorter is this time, the shorter is the stage propagation delay. In [9], Winstanley *et. al.* implemented an experimental circuit where the Charlie and drafting effects can be controlled within the design. They highlighted how those effects affect the event propagation across the ring: the Charlie effect favorizes the evenly-spaced propagation mode while the drafting effect favorizes the burst propagation mode. Intuitively, the Charlie effect causes two close events to push away from each other due to the increased delay experienced by a ring stage when driven by two events separated with a short time lapse. The evenly-spaced propagation happens when the events keep pushing from each other until they spread-out evenly across the ring. The time lapses that separate successive events converge to one final value corresponding to the working point of a steady regime. Contrarily, the drafting effect causes two close events to gather together because of the reduced propagation delay of a ring stage when it switches at a higher rate. The final state of the self-timed ring (evenly-spaced or burst) and moreover, the working point of the evenly-spaced mode depend on the following parameters of the design:

- The Charlie and drafting effects which depend on the ring stage implementation and the used technology
- The propagation delays of a ring stage D_{ff} and D_{rr} as represented in Fig. 6, or more precisely, their ratio $\frac{D_{ff}}{D_{rr}}$
- The ring occupancy, i.e. the ratio $\frac{N_T}{N_B}$ where N_T is the number of tokens and N_B the number of bubbles

Thus, for a fixed design, the final state of a self-timed ring depends only on its initial occupancy. The evenly-spaced mode is achieved for a range of values of $\frac{N_T}{N_B}$ around $\frac{D_{ff}}{D_{rr}}$. It is automatically achieved when [11]:

$$\frac{N_T}{N_B} \simeq \frac{D_{ff}}{D_{rr}} \tag{1}$$

In practice, the higher is the Charlie effect magnitude, the larger is the interval of $\frac{N_T}{N_B}$ around $\frac{D_{ff}}{D_{rr}}$ where the ring achieves the evenly-spaced propagation mode. For example, a 64-stage self-timed ring in Altera Cyclone III with $\frac{D_{ff}}{D_{rr}} \simeq 1$ exhibits the evenly-spaced mode for N_T varying between 22 and 42 ($N_B = 64 - N_T$). While the same 64-stage configuration in Xilinx Virtex 5 achieves the evenly-spaced mode for N_T between 28 and 38 (which suggests a stronger Charlie effect in the Cyclone III implementation).

B. Frequency

The frequency of a self-timed ring is a function of its occupancy. Figure 8 shows the frequency curve of an L-stage self-timed ring as a function of the number of initialized events. N_{min} and N_{max} correspond to the limit where the events start to bunch. N_0 corresponds to the point described by Equation 1. This point also separates the curve into two regions (not necessarily symmetrical depending on D_{ff} and D_{rr}): the token-limited region where $\frac{N_T}{N_B} < \frac{D_{ff}}{D_{rr}}$ and the bubble-limited region where $\frac{N_T}{N_B} > \frac{D_{ff}}{D_{rr}}$. In the token-limited region, the frequency increases with the number of events as it could be expected intuitively. The maximal frequency is achieved when Equation 1 is satisfied. Then the frequency starts dropping with the number of events in the bubble-limited region. This is due to the fact that events have to wait for the acknowledgment signals since most stages are already processing events. This reduces the operating frequency of the ring.

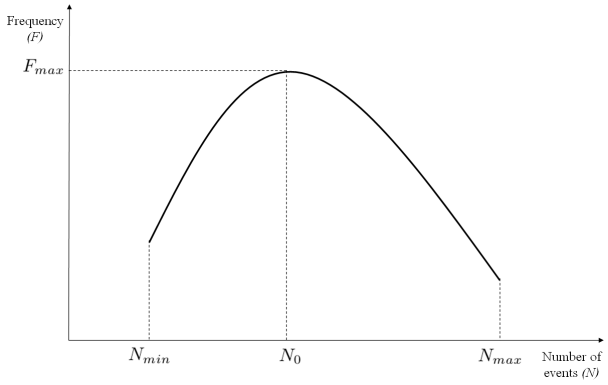


Fig. 8. Theoretical frequency curve of an L-stage self-timed ring as a function of the ring occupancy

C. Phase Distribution

In [10], Fairbanks describes design methods for using self-timed rings as generators of signals with high timing precision. Contrary to inverter ring oscillators, self-timed rings allow phase resolutions which are fractions of the propagation delay of one ring stage because several events evolve simultaneously in the ring. Each event propagating in a self-timed ring stage inverts its output. If N events spread evenly in the ring (across its structure), then each stage exhibits a $\frac{N}{L} \times 180^o$ phase difference with its predecessor. Therefore, the phase difference between 2 stages which are n stages apart is [10]:

$$\varphi_n = n \times \frac{N}{L} \times 180^o \tag{2}$$

According to Equation (2), if the number of stages is a multiple of the number of events, some stages may exhibit the same absolute phase. But if the number of events and the number of stages are co-prime, the self-timed ring exhibits as many different equidistant phases as the number of stages. For example, a 9-stage self-timed ring with 4 tokens and 5 bubbles exhibits 9 different equidistant phases. An 18-stage self-timed ring with 8 tokens and 10 bubbles also exhibits 9 different equidistant phases since the ratio $\frac{8}{10}$ can be reduced to $\frac{4}{5}$. A self-timed ring where $N_T = N_B$ and $D_{ff} = D_{rr}$ exhibits 4 different equidistant phases no matter the number of stages. If T is the oscillation period of a self-timed ring where the number of events and the number of stages are co-prime, its phase resolution expressed in the time domain is as follows:

$$\triangle\varphi = \frac{T}{2L} \tag{3}$$

On the other hand, the oscillation period in a self-timed ring does not depend directly on the number of stages as it does in inverter ring oscillators, but it is a function of the ring occupancy (the ratio of the number of events to the number of stages $\frac{N}{L}$). This means that it is possible to enhance the phase resolution $\triangle\varphi$ without modifying the ring frequency by increasing L while keeping the same ratio $\frac{N}{L}$. Subsequently, the phase resolution of a self-timed ring can theoretically be set as finely as needed. Elissati *et. al.* demonstrate the efficiency of the method in [13] by implementing several designs where they measure phase resolutions of the order of picoseconds.

D. Jitter

As explained before, self-timed rings are able to auto-regulate timings between the events when locked in the steady regime. One major consequence of this feature is that they deliver a low jitter issued mainly from the local Gaussian noise sources. To understand this fact, let's consider an inverter ring oscillator where one event propagates freely around the ring. This event experiences variations in its significant timings due to noise each time it crosses a ring stage. Since the event propagation is unconstrained, this timing variation is transmitted to the next stage and accumulates as the event propagates in the ring. Jitter measurements show that the period jitter of an inverter ring oscillators (which refers to the standard deviation of a population of measured oscillation periods) increases with the number of ring stages [14]. On the contrary, an event propagating in a self-timed ring progressively loses the carried jitter timing variation: its significant timings are self-regulated as the separation time between successive events converges to the working point of the steady regime and its value does not depend on dynamic parameters such as noise in the circuit. Jitter measurements show that the period jitter of a self-timed ring does not increase with the number of stages [14]. Moreover, the jitter measured at the output of a self-timed ring is of the same order of magnitude than the measured jitter of one single ring stage. This suggests that the jitter generated locally in each ring stage does not propagate to the other stages as it does in inverter ring oscillators. On the other hand, global deterministic jitter affects each event in the same manner. Subsequently, these deterministic timing variations are strongly attenuated when we consider the separation time between successive events. In conclusion, the jitter measured at the output of a self-timed ring stage is mostly composed of the random local jitter that originates from the concerned ring stage.

VI. STRNG Design and Measurements in Virtex Xilinx 5 and Altera Cyclone III FPGAs

The STRNG principle relies on the setup of the self-timed ring phase resolution according to the measured jitter magnitude. This section details the STRNG implementation and design constraints. It also provides jitter and phase resolution measurements for different self-timed ring configurations in Altera Cyclone III and Xilinx Virtex 5 FPGAs.

A. STRNG Implementation and design constraints

Muller gates are basic elements in asynchronous circuit design. They can be implemented in FPGAs using Look-Up-Tables (LUTs). Each self-timed ring stage (Muller gate + inverter) can be implemented using one LUT. At least 4 inputs are required: 2 inputs are used for the forward and reverse inputs, 1 input is used to SET or RESET the stage, and one input serves as the feedback-loop which is necessary for maintaining the state value. SET and RESET allow to set the initial number of tokens in the ring. We implemented two kinds of self-timed ring stages: some cells with SET and other cells with RESET. When connecting the stages and closing the loop, one common INIT signal initializes the ring with the correct number of tokens. The internal logic function of a self-timed ring stage is realized using one LUT function generator as shown in Table I (both Altera Cyclone III and Xilinx Virtex 5 feature 4-input 1-output function generators). I0 corresponds to the INIT input, I1 and I2 are the forward and reverse inputs (*F* and *R* in Fig. 6) and I3 is the feedback input connected to the output S (which refers to the output *C* of the self-timed ring).

I0	I1	I2	I3	S
0	0	0	0	0
0	0	0	1	1
0	0	1	0	0
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	0
0	1	1	1	1
1	-	-	-	0

TABLE I
SELF-TIMED RING STAGE WITH RESET DESCRIPTION USING A 4-INPUT 1-OUTPUT FUNCTION GENERATOR (LUT)

Although the self-timed ring can support propagation delay disparities between its stages, if one delay is much longer than the others, the ring can exhibit a bottleneck: events gather at the input of this ring stage and cannot be distributed evenly in time. The self-timed ring frequency is then limited by this long propagation delay. To avoid bottleneck effects, it is recommended to choose a placement topology that guarantees balanced delays between the ring stages. In particular, a topology with a long feedback loop should be avoided. Fig. 9 illustrates an example of a ring topology that limits bottleneck effects in Altera Cyclone III.

For a given number of stages and according to Equation 3, the lower is the oscillation period of the self-timed ring, the better is its phase resolution. As shown in Fig. 8, the maximum frequency of the self-timed ring is achieved when the ratio $\frac{N_T}{N_B}$ is as near as possible to $\frac{D_{ff}}{D_{rr}}$. The ratio $\frac{N_T}{N_B}$ needs also to be irreducible to obtain as many different phases as the number of ring stages. Considering the placement topologies in both Virtex 5 and Cyclone III, the majority of ring stages have equivalent forward and reverse propagation delays:

$$\frac{D_{ff_{mean}}}{D_{rr_{mean}}} \simeq 1 \tag{4}$$

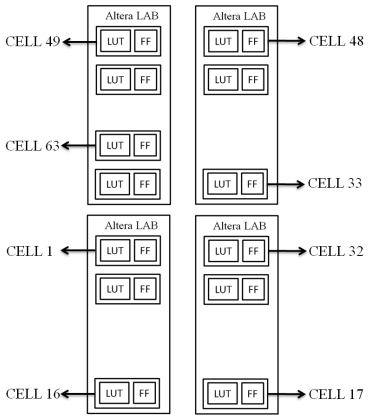


Fig. 9. Example of a self-timed ring (STR) topology for a 63-stage STR in Altera Cyclone III device (each Altera Logic Array Block -LAB- contains 16 logic cells consisting of a LUT and a flip-flop)

This is also verified by the fact that the interval of tokens that achieve the evenly-spaced mode is centered around $N_T \simeq \frac{L}{2}$.

In this case, one simple strategy is to choose first L odd, then $N_T = \frac{L+1}{2}$ and $N_B = \frac{L-1}{2} = N_T - 1$. Thus, if $L > 2$, $\frac{L+1}{2}$ and $\frac{L-1}{2}$ are relatively prime. Moreover, if L is high enough, $\frac{L+1}{L-1} \simeq 1$, this way the frequency is maintained at its maximum while increasing the number of ring stages and enhancing the phase resolution of the self-timed ring.

In order to guarantee a correct phase distribution at the flip-flop inputs, connections between all self-timed ring stages and corresponding flip-flops must remain the same. Fortunately, the majority of recent FPGAs (including Xilinx Virtex 5 and Altera Cyclone III) feature a hard-wired connexion between LUTs and flip-flops that can be successfully used in the STRNG design.

Finally, the XOR tree is implemented using 4-input LUTs in Altera Cyclone III and 6-input LUTs in Xilinx Virtex 5.

B. Phase resolution and jitter measurements

1) Measuring the jitter: One main assumption of the STRNG principle is the presence of an uncorrelated, unbiased random jitter issued from the white noise. This can be obtained in practice in digital circuits because of the unavoidable thermal noise (due to random movements of the current carriers). One major issue however is to precisely measure this jitter magnitude independently from additional noise sources in order to correctly set the phase resolution of the self-timed ring. Fortunately, due to the high frequencies of self-timed rings ($\simeq 400$ MHz in FPGAs, a few GHz in ASICs), Flicker and Brownian noises can be neglected. Nonetheless, a few precautions must be taken when realizing this measurement:

- The surrounding logic (i.e. the flip-flops and XOR tree) should not operate during measurements in order to limit any deterministic effects that could result from its operation. This allows to measure the minimal (unavoidable) jitter present at the output of the self-timed ring
- Linear voltage regulators should be used in order to reduce the power supply noise
- Low Voltage Differential Signaling (LVDS) outputs in conjunction with differential oscilloscope probes should be used in order to reduce the impact of slow input/output circuitry and parasitic effects of the output

The period standard deviation (σ_{period}) is obtained by acquiring a significant number of successive oscillation periods. According to [14], the jitter magnitude of a self-timed ring stage can be estimated using the following equation:

$$\sigma \simeq \frac{\sigma_{period}}{\sqrt{2}} \tag{5}$$

2) Estimating the phase resolution: The phase resolution is computed by measuring the mean oscillation period and using Equation 3. However, contrarily to the jitter measurement, the whole design must be operating in this case. In fact, the flip-flops and XOR tree connected to the ring stages cause its frequency to drop (the phase resolution can therefore be worse than expected). It is thus more cautious to measure the phase resolution in this case.

3) Results: Frequency and jitter were measured using a wide band digital oscilloscope LeCroy Wavepro 735 ZI. We used the LVDS (Low Voltage Differential Signaling) interface of the device and an active differential probe with a 4 GHz bandwidth. First, we implemented different configurations of the self-timed ring in both devices and measured their oscillation period and period jitter. The standard deviation of the propagation delay of one self-timed ring stage is computed using Equation 5 and the phase resolution of the self-timed ring is computed using Equation 3. Figure 10 shows the period distribution of a 127-stage self-timed ring with 64 tokens in both Altera Cyclone III and Xilinx Virtex 5. As it can be seen, both configurations exhibit a Gaussian jitter profile. Table II gives the measured oscillation period (T), the computed phase resolution using Equation 3 ($\triangle\varphi$) and the measured jitter magnitude (σ). L is the number of ring stages, N is the number of initialized events. Measured jitter magnitude values vary around $2ps$ in Altera Cyclone III and $2.5ps$ in Xilinx Virtex 5.

Device	L	N	T	$\triangle\varphi$	σ
Altera Cyclone 3	63	32	2.07 ns	16.4 ps	2.1 ps
	127	64	2.07 ns	8.2 ps	1.7 ps
	255	128	2.08 ns	4.0 ps	1.7 ps
	511	256	2.46 ns	2.4 ps	1.9 ps
	1023	512	2.63 ns	1.3 ps	1.8 ps
Xilinx Virtex 5	63	32	3.44 ns	26.9 ps	2.7 ps
	127	64	3.42 ns	13.5 ps	2.6 ps
	255	128	3.72 ns	7.3 ps	2.8 ps
	511	256	3.95 ns	3.9 ps	2.4 ps
	1023	512	4.12 ns	2.0 ps	2.5 ps

TABLE II
OSCILLATION PERIOD (T), PHASE RESOLUTION ($\triangle\varphi$) AND JITTER PER RING STAGE (σ) FOR DIFFERENT SELF-TIMED RING CONFIGURATIONS IN ALTERA CYCLONE III AND XILINX VIRTEX 5

VII. STRNG STATISTICAL EVALUATION

For each self-timed ring configuration, we acquired 1.2 GBytes of data. In theory, the STRNG principle allows sampling frequencies up to the self-timed ring frequency (a few

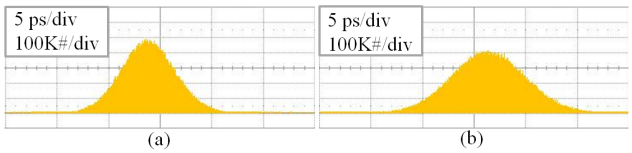


Fig. 10. Period distribution histogram of a 127-stage STR with 64 tokens: (a) Altera Cyclone III (b) Xilinx Virtex 5

hundred Mhz). However, our USB transfer protocol limited the output bit rate to 16 Mbit/s. In order to correctly evaluate the entropy issued from the self-timed ring jitter, we used a low-jitter 16 MHz quartz as a sampling clock. We applied FIPS 140-1 statistical tests on 1000 sequences of 20000 bits and NIST SP 800-22 statistical tests on 1000 sequences of 10^6 bits with a 0.01 confidence level. As it can be seen in Table III, the TRNG output passes the FIPS and NIST tests starting from $L = 511$ in Cyclone III (corresponding to $\triangle\varphi \simeq 0.7\sigma$). The design passes the FIPS tests in Virtex 5 starting from $L = 511$ ($\triangle\varphi \simeq 0.8\sigma$), but does not pass the NIST tests due to a higher bias (the majority of tests pass except the frequency test). This table also shows that the principle does not work when the number of events and the number of stages are not co-prime as expected from Equation (2) (when $L = 128$ and $N = 64$ in the table).

Parity filter	STR configuration		Cyclone III		Virtex 5	
	L	N	FIPS	NIST	FIPS	NIST
none	63	32	55%	FAIL	14%	FAIL
	127	64	98%	FAIL	1%	FAIL
	128	64	0%	FAIL	0%	FAIL
	255	128	100%	FAIL	99%	FAIL
	511	256	100%	PASS	100%	FAIL
8^{th} order	63	32	100%	FAIL	100%	FAIL
	127	64	100%	PASS	100%	FAIL
	128	64	0%	FAIL	0%	FAIL
	255	128	100%	PASS	100%	PASS
	511	256	100%	PASS	100%	PASS

TABLE III
STATISTICAL EVALUATION OF THE STRNG USING NIST SP 800-22 AND FIPS 140-1 STANDARD TEST SUITS

If necessary, the designer can reduce the TRNG core area at the cost of a reduced bit rate by compressing successive bits using a parity filter. An n^{th} order parity filter regroups n successive bits using a XOR function to provide one bit at the filter output, thus enhancing the entropy per bit (and reducing the bias), but dividing the bit rate by n. We used an 8^{th} order parity filter and increased the sampling clock frequency to 120 MHz in order to maintain the output bit-rate at 16 Mbit/s. As shown in Table III, the parity filter corrects most bias problems in Virtex 5 and enhances the tests passing rates in both devices. Using the 8^{th} order parity filter, only 127 stages are needed to pass the statistical tests in Cyclone III (255 stages in Virtex 5).

VIII. CONCLUSION

This paper presented a novel TRNG design that uses the jitter of events propagating in a self-timed ring to generate random bits at a high bit rate. The self-timed ring allows to adjust the time lapse between two successive events as short as needed by simply increasing its number of stages and adjusting its number of events. This time lapse can be thus adapted to the jitter magnitude which depends on the selected technology and device. Therefore, the design allows to extract entropy from the jitter even if its magnitude is extremely low. Moreover, the designer can precisely tune the architecture based on his security, throughput, cost and power consumption requirements with a very low design effort. Future works will include a detailed stochastic model of the entropy extraction allowing to compute the lower bound of entropy per output bit. This model will be used along with experimental measurements for selecting the number of ring stages in order to achieve a sufficient entropy per output bit of the TRNG.

ACKNOWLEDGMENT

The presented research is funded by the Rhone-Alpes region (France) in the frame of the SEMBA project.

REFERENCES

[1] RAND Corp, "A Million Random Digits with 100000 Normal Deviates", *Free press NY*, 1955.

[2] R.C. Fairfield, R.L. Mortenson and K.B. Coulthart, "An LSI Random Number Generator (RNG)", *in the proceedings of CRYPTO 84 on Advances in cryptology*, pages 203-230, NY USA, 1985.

[3] V. Fischer and M. Drutarovsky, "True Random Number Generator Embedded in Reconfigurable Hardware", *Lecture notes in computer science*, pages 415-430, 2003.

[4] M. Dichtl and J.D. Golic, "High-speed True Random Number Generation with Logic Gates Only", *Lecture notes in computer science*, 4727:45, 2007.

[5] B. Sunar, W.J. Martin and D.R. Stinson, "A Provably Secure True Random Number Generator with Built-in Tolerance to Active Attacks", *in IEEE Transactions on Computers*, 56(1):109, 2007.

[6] N. Bochard, F. Bernard, V. Fischer and B. Valtchanov, "True-Randomness and Pseudo-Randomness in Ring Oscillator-Based True Random Number Generators", *International Journal of Reconfigurable Computing*, volume 2010, article ID 879281.

[7] I. E. Sutherland, "Micropipelines", *in Communications of the ACM (Association of Computing Machinery)*, Vol/Issue:32/6, pages 720738, 1989.

[8] J.L. Danger, S. Guille and P. Hoogvorst, "Fast True Random Generator in FPGAs", *IEEE Northeast Workshop on Circuits and Systems*, NEWCAS 2007, pages 506-509.

[9] A. Winstanley and M. R. Greenstreet, "Temporal Properties of Self-Timed Rings", *in Proceedings of the 11th Advanced Research Working Conference on Correct Hardware Design and Verification Methods, CHARM01*, London, UK: Springer-Verlag, pages 140-154, 2001.

[10] S. Fairbanks, "High Precision Timing using Self-timed Circuits", *Technical report no. UCAM-CL-TR-738*, University of Cambridge, Computer Laboratory, January 2009, url: http://www.cl.cam.ac.uk/techreports/UCAM-CL-TR-738.pdf

[11] J. Hamon, L. Fesquet, B. Miscopein and M. Renaudin, "High-Level Time-Accurate Model for the Design of Self-Timed Ring Oscillators", *in Proceedings on the 14th International Symposium on Asynchronous Circuits and Systems, ASYNC08*, pages 29-38, April 2008.

[12] J. C. Ebergen, S. Fairbanks and I. E. Sutherland, "Predicting Performance of Micropipelines Using Charlie Diagrams", *in Proceedings of the Fourth International Symposium on Advanced Research in Asynchronous Circuits and Systems, ASYNC98*, pages 238-246, 1998.

[13] O. Elissati, E. Yahya, S. Rieubon and L. Fesquet, "A Novel High-speed Multi-phase Oscillator using self-timed Rings", *in the International conference of Microelectronics, ICM 2010*, pages 204-207, 2010.

[14] A. Cherkaoui, V. Fischer, A. Aubert and L. Fesquet, "Comparison of Self-timed and Inverter Ring Oscillators as Entropy Sources in FPGAs", *in Design, Automation and Test in Europe conference, DATE12*, pages 1325-1330, March 2012.