Design and Implementation of a True Random Number Generator Based on Digital Circuit Artifacts

Michael Epstein[1], Laszlo Hars[2], Raymond Krasinski[1], Martin Rosner[3],
Hao Zheng[4]

[1] Philips Electronics, Philips Intellectual Property and Standards, 345 Scarborough Road,
Briarcliff Manor, NY 10510
{Michael.Epstein, Raymond.Krasinski}@philips.com
[2] Seagate Technology, 1251 Waterfront Place,
Pittsburgh, PA 15222
Laszlo.Hars@Seagate.com
[3] Philips Electronics, Philips Research, 345 Scarborough Road,
Briarcliff Manor, NY 10510
Martin.Rosner@philips.com
[4] ActiveEye, 286 Broadway,
Pleasantville, NY 10570
Hao.Zheng@activeye.com

Abstract. There are many applications for true, unpredictable random numbers. For example the strength of numerous cryptographic operations is often dependent on a source of truly random numbers. Sources of random information are available in nature but are often hard to access in integrated circuits. In some specialized applications, analog noise sources are used in digital circuits at great cost in silicon area and power consumption. These analog circuits are often influenced by periodic signal sources that are in close proximity to the random number generator. We present a random number generator comprised entirely of digital circuits, which utilizes electronic noise. Unlike earlier work [11], only standard digital gates without regard to precise layout were used.

1 Introduction

True random-number generators are often desirable in many applications ranging from statistical system analysis to information security protocols and algorithms. Currently available true random number generators utilize circuitry that often consumes significant resources on integrated circuits and/or require incompatible analog and digital elements. Other, more primitive generators do not provide a convenient interface to electronic devices. In this paper we describe the design of a new type of true random number generator that is based solely on digital components (i.e. it is inexpensive to build), consumes little power, provides high throughput, and passes the DIEHARD [15] suite of tests for randomness. It is envisioned that this design of a random number generator will provide an inexpensive alternative to generators that are currently embedded in many systems such as microprocessors and smart cards.

This paper introduces the concepts behind a simple true random number generator. The focus of this paper is on digital circuits that exhibit metastability and those that function as unstable oscillators. Others have used metastability as a source of randomness and have found some success, but only after trimming devices with a laser to achieve a precisely balanced circuit. (See, e.g. [10], [11], [12]).

We designed and implemented a wide array of this type of true random number generators. The prototype chip consists of nine distinct designs. The prototype chip was mounted on an acquisition breadboard for testing. The extracted results were analyzed to determine which designs yield the best results. The results show that even without de-biasing the resulting sequences, some of the designs provide random datasets that pass the DIEHARD suite of tests. This paper details the methodology for design, implementation and testing of a true random number generator based on digital artifacts. We conclude this paper by providing results, and outlining the necessary steps to create practical versions of these promising designs.

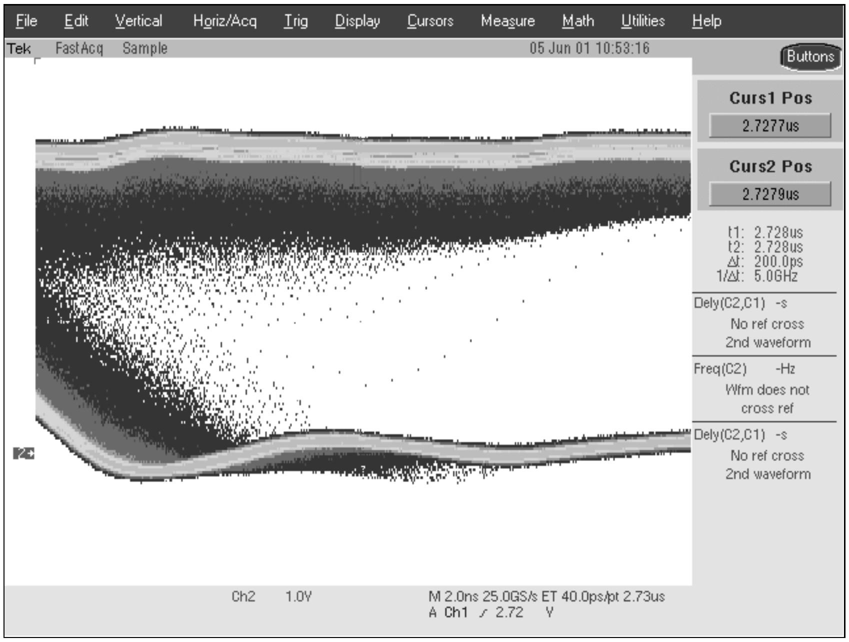
2 Description of Digital Artifacts

The design described in this paper yields random results by means of digital circuit artifacts [13], [14]. The design utilizes a pair of oscillators that are permitted to free-run. At some point, the free-running oscillators are coerced to matching states via a bi-stable device. While we believe that the circuit utilizes the metastability artifact, this conjecture has not been proven. However, we have experimental results from a similar circuit composed of discrete components, which does show metastability. A discussion of the two possible causes of randomness, metastability and oscillator drift and jitter, is presented below.

2.1 Metastability

Digital circuits, by their nature, are designed to be predictable. If the same logic levels in the same order are presented to digital circuit, the same result should always occur. However if the rules governing the inputs of digital circuits are violated then the results can become unpredictable. In particular, the metastability phenomenon may occur in a digital flip-flop or in a latch. A flip-flop, which is a memory element that can store one bit, is one type of bi-stable device. At the heart of every flip-flop is a pair of logic gates that are fed back to each other. This electrical feedback is what preserves the logical bit stored in the circuit. If the setup and hold conditions of the flip-flop are violated [1] then the pair of gates will behave unpredictably or even oscillate about some intermediate voltage [2]. During the oscillatory or metastable state, the output is neither a logical zero nor a logical one. After some time the oscillations will die out and the flip-flop will settle into a logical state of a zero or a one as shown in figure 1.

There are three uncertainties at work here. The first uncertainty is if the circuit will behave normally i.e. attain a logical state after the usual delay for the flip-flop, or will the circuit enter the metastable state [3]. The second uncertainty is the state that the flip-flop will settle into. The third uncertainty is the length of time that the circuit will remain metastable [1]. Some of these uncertainties have been measured [4] and modeled [5] for various kinds of circuitry and environmental conditions. It has also been shown that this phenomenon cannot be avoided in any flip-flop [4], [5]. Thus, the metastability phenomenon provides a source of randomness that can be used to construct a true random number generator without the need for specialized analog circuits.



Tek FastAcq Sample 05 Jun 01 10:53:16

Curs1 Pos 2.7277us

Curs2 Pos 2.7279us

t1: 2.728us
t2: 2.728us
Δt: 200.0ps
1/Δt: 5.0GHz

Dely(C2,C1) -s
No ref cross
2nd waveform

Freq(C2) -Hz
Wfm does not
cross ref

Dely(C2,C1) -s
No ref cross
2nd waveform

Ch2 1.0V

M 2.0ns 25.0GS/s ET 40.0ps/pt 2.73us
A Ch1 ⸝ 2.72 V

Fig. 1. Recovery from metastable statedisplayed on a TDS7254 Tektronix oscilloscope using the P7240 active probe. The signal resolving from a metastable state was captured using infinite persistence in the Fast Acquisition mode [y-axis: 1volt/division with 8 divisions shown; x-axis: 2ns/division with 10 divisions shown]. Lower intensity (lighter traces) shows more frequent signal traces. Note the single trace where the signal oscillates toward zero but eventually stabilizes at a one

2.2 Oscillator Drift and Jitter

A clock period is never a precise constant, even in highly regulated clock oscillators, such as those that utilize a crystal. Careful observation shows that the oscillating signal has slight changes of phase. Such perturbations of oscillator period are called jitter. Some components of jitter are random [6]. Precise measurement of jitter is more of an art that a science [7].

The design uses two free-running (without a crystal or similar reference) oscillators, which are allowed to drift away from each other. Such oscillators are known to exhibit a great deal of variability even over short periods of time. Thus, the different internal noise of the two similar oscillators (causing phase jitter, accumulated as phase drift) can be utilized as a source of randomness. After some time the instantaneous voltage is latched by a digital bi-stable circuit, capturing the random state.

3 Integrated Circuit Design

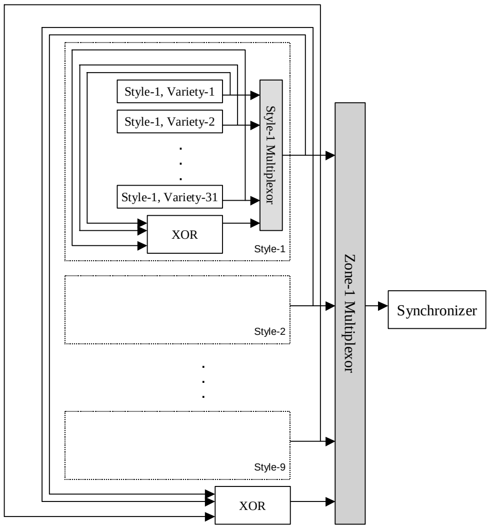


Fig. 2. Zone circuit; Selection of various RNG designs was designated through multiple levels of multiplexers

In order to validate these concepts, an integrated circuit containing nine distinct types (or styles) of random number generators was constructed. Each design utilized a bi-stable device, which in most cases contained a pair of gates to form the memory element. Given the obvious dependence on circuit delays each style of random generator was replicated in 15 to 31 different varieties for a total of 247 distinct random number generators. The varieties used different gate sizes, and thus different circuit delays, in pairs of matched or sometimes unmatched gates in an effort to explore the entire problem space for each of the nine styles. All of the gates were drawn from a standard 0.18 micron CMOS library and laid out automatically. No effort was made to minimize or match wiring delays, although given the small size of the circuits it is likely that some varieties are very similar.

A multiplexer was utilized to allow for the selection of a particular variety for that style of generator. Every variety from a particular style was also connected to a network of XOR gates thus combining all of the varieties of a style into a single output. This XOR output became one of the inputs to the style multiplexer as well. Any single variety or the XOR of the varieties in the style could be selected via a control word as shown in figure 2.

Each of the styles was then fed to a zone multiplexer. As with the varieties, all of the styles were then XOR’ed together to create another input for the zone multiplexer. A control word was configured to choose any particular style or the XOR combination of the all nine styles (see figure 2). Ultimately 288 variations could be individually selected:

- A specific variety of a specific style of the random number generator
- The XOR value of all varieties of a specific style
- The XOR value of specific varieties combining all styles
- The XOR value of all varieties of all styles.

By including a wide variety of designs, numerous combinations of digital circuit artifacts were tested to determine if a real world random number generator could be created via a single circuit design or a combination of designs.

Once a bi-stable device becomes metastable its non-logic voltages can propagate throughout a circuit and cause other flip-flops in the circuit to become metastable as well. While it is desirable that the random number generator has an unpredictable output the same cannot be said for the typical circuit that requires the random bits. Figure 2, includes a synchronizer circuit that effectively isolates the random number generator from the rest of the circuit by capturing the bits in a series of three flip-flops. Typical flip-flops will not enter a metastable state easily. However, even if the first of these flip-flops were to enter the metastable state, the clock period is sufficiently long so that it is likely that the flip-flop will have left the metastable state and resolved to a zero or one prior to the needed setup time for the second flip-flop. Similarly, the second flip-flop protects the third one. This method is a well-known technique [2] for reducing chances of failure due to metastable behavior. The chances of metastable behavior propagating through the synchronization circuit can be measured in tens or hundreds of years.

In the interest of creating the greatest possible set of variations, the entire zone circuit that is shown in figure 2 was replicated eight times. By purposely replicating the design numerous times, further variations in the circuit layout may have been introduced. The outputs of all zones were directed off chip for analysis.

4 Circuit Description of a Random Number Generator

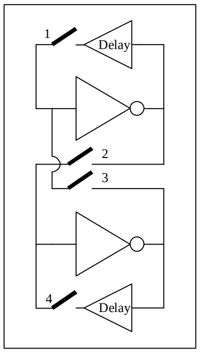


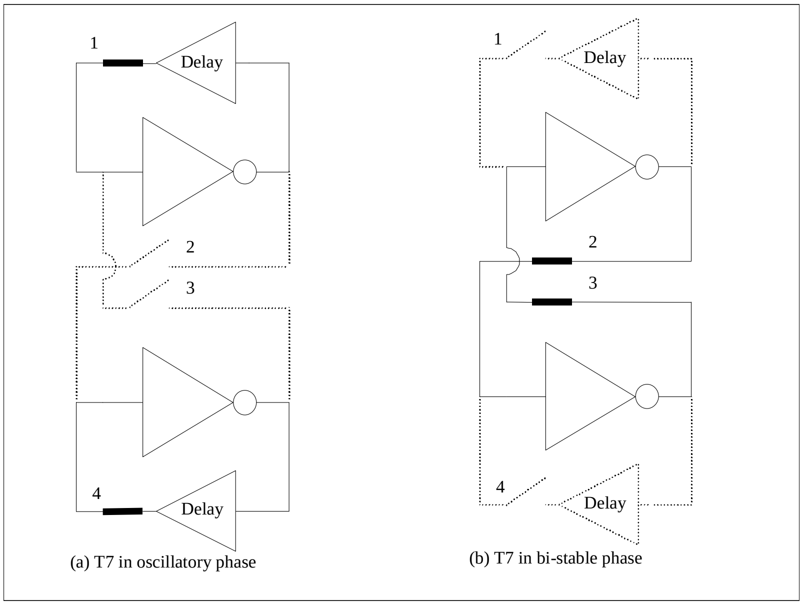
Fig. 3. The T7 concept

Nine different styles or types of the random number generator were implemented in the test chip. Results shown in Appendix A, Table 1 found that six of the styles, designated T1-T6, of the random number generator failed completely by giving only a single binary value of one or zero. These six styles attempted, in different ways, to cause a library flip-flop to become metastable by violating the setup and hold requirements. The failure of these styles to generate truly random sequences can be partially attributed to the fact that modern flip-flops are designed to suppress the metastable artifact. Of the remaining styles, one (T9) gave results that failed to pass all of the DIEHARD Tests. Two styles (T7 and T8) produced results that were random when all of the varieties were

XOR'ed together and sometimes from a single random number generator circuit. Since the T7 and T8 designs were quite similar, the focus of the remaining testing effort was on style T7. Figure 3 shows the basic principle behind the design of T7. Later research may explore the usefulness of designs T8 and T9 and discover why T1-T6 failed to produce random results (see appendix A).

4.1 Theory of Operation

The basic concept behind style T7 is relatively simple. The design consists of two inverters and four switches, numbered 1 through 4, as shown in figure 4.



(a) T7 in oscillatory phase

(b) T7 in bi-stable phase

Fig. 4. Operating phases of T7

The switches can be implemented as transition gates or as multiplexers. The actual design was implemented as multiplexers as shown in figure 5. In this case the multiplexers served as the delay elements as well as the switching elements.

Returning to the switch based design in figure 4a, if switches 1 and 4 are closed while switches 2 and 3 are open a configuration is created where the inverters form two independent, free-running ring oscillators, caused by the delay in the negative feedback loop. The inverters can be supplemented by delay elements implemented as a series of buffer gates or an even number of inverters. Ideally each of the oscillators should be sufficiently different so that if the circuit encounters a strong external signal there is little chance that both oscillators would synchronize to it.

If switches 1 and 4 are opened while switches 2 and 3 are closed the connected inverters form a bi-stable memory device as shown in figure 4b. Because of positive feedback the outputs of the inverters eventually resolve, by clipping, to a consistent logic state. This final logic state creates one random bit. The randomness of the bit is derived from the conditions created when the two free-running oscillators are stopped (the oscillator feedback loops are opened and the two inverters get cross-connected). At that point the relative and absolute values of the instantaneous output voltages and the internal noise determine the eventual logic state the circuit will settle to, sometimes even via the artifact of metastability. Thus, the randomness of the circuit exploits two different mechanisms.

4.2 Drift and Jitter

The two oscillators drift apart in different ways because when the oscillators are switched on they don't start immediately, they "hesitate" for a short period of time then the voltage goes either up or down, creating an uncertain starting point. The loop gain of the oscillators should be small otherwise they will start instantly. As the two oscillators continue to oscillate random circuit noise affects the unregulated oscillators so that their clock periods are inconsistent from cycle to cycle. The combined effects ensure that the two oscillators will find themselves in different states each time they are stopped.

4.3 Metastability

When the oscillators are stopped, the gates are cross-connected forming a bi-stable memory element. At this point one or both output voltages may be between logic levels. The bi-stable device, which is formed by the inverters, must settle to a logic

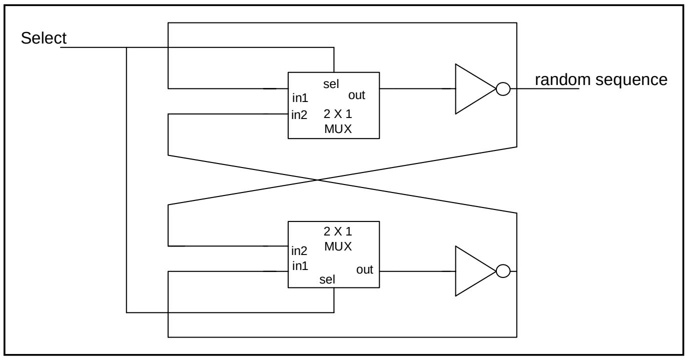


Fig. 5. Actual T7 design based on bi-stable memory element. The random sequence generator is a pair of cross-connected inverters where the multiplexers are used to switch between oscillatory and bi-stable phases

state over time. However, sometimes the bi-stable device will initially find itself in a conflicted state as the two oscillators do not agree or even arrive at a consistent state of disagreement. The bi-stable device will therefore oscillate for a period of time until a final, stable state can be achieved. Thus the final state of the metastable device produces a random bit.

4.5 Circuit Details

There were several varieties, of the design laid out for the test chip. The types of inverters were varied to achieve different delays. One option not explored was to replace a single inverter with a chain of inverters for larger delays. Short (and different) delays are preferred because the ring oscillators will produce smaller amplitude, sinusoid like signals, which should provoke metastability more often.

Very short delays might prevent oscillation if the gain of the circuit is too small at the fundamental frequency of the feedback loop. However, the circuit should work even in this case. The large negative feedback forces the output of the inverters to an intermediate voltage, close to halfway between logic levels. Flipping the switches to activate the bi-stable configuration at an intermediate voltage often forces metastability [10], [11]. The final state achieved under these conditions is shown to be unpredictable if the initial conditions of the bi-stable circuit are at intermediate voltages.

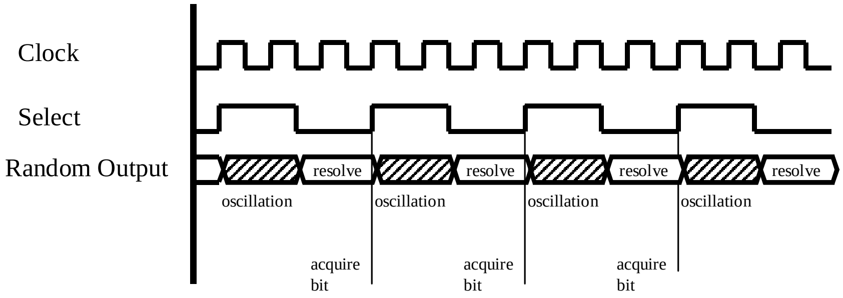


Fig. 6. Random bit acquisition; The ‘Select’ signal is used to drive the multiplexers that choose between acquisition and oscillation phase of the random number generator.

In the actual design a synchronous circuit was used to collect random numbers. Figure 6 shows how a divided version of the clock was used to switch the multiplexers via the select signal. When the select signal is high the circuit is in the oscillation state and each inverter operates independently. At that point the output is not in any logic state as is indicated in by the diagonal crosshatches. Sufficient time is allowed for the oscillators to diverge in the oscillation state. When the select line goes low the circuit is in the bi-stable configuration and resolves to a single value, either a 1 or 0, via metastable oscillations. The resolution time can be short or long but sufficient time must be allowed in order for the value to be resolved on the vast majority of occasions.

Should the random bit be unresolved the synchronization circuit described in section 3 will resolve it. The rising edge of the select signal can be used to acquire the random bit as shown in figure 6.

5 Improving Randomness

Variations within the manufacturing tolerances and unpredictable environmental changes (temperature, supply voltage etc.) will alter the behavior of the randomness circuit. The circuit may fail in an obvious way such as producing all 0 or all 1 output or possibly a heavily biased sequence. The simplest solution to this problem is to lay out many, slightly different versions of the circuit, on the chip. Under different environmental conditions some of the versions will work randomly while others will produce a biased output. If the differences in the circuits are small and there are a large number of varieties of the randomness circuit, with high probability at least one variety will always produce random results.

The randomness of all of the different varieties is collected by simply XOR'ing all of the outputs of the random circuits on the chip [8]. If there is at least one truly random output sequence the final result will still be random, as shown by Matsui [9]. Since each random source is small, only a few hundred standard logic gates will provide very high quality random numbers in real world conditions.

6 Data Gathering and Analysis

For each of the random number generators tested a data sequence of 80 megabits was collected. Each sequence was submitted to the DIEHARD Tests for evaluation.

The DIEHARD Tests is a collection of 16 individual tests that altogether produce 215 results (called Pvalues) Each Pvalue is obtained by applying a function [Pvalue = $F_i(X)$ (i = 1-215)], where the function F_i seeks to establish a distribution function of the sample random variable X as uniform between 0 and 1. In addition all of the functions F_i are an asymptotic approximation, for which the fit will be worst in the tails. Thus only rarely will one find Pvalues near 0 or 1, such as 0.0012 or 0.9983, if the sequence is random. When a sequence is decidedly non-random numerous Pvalues of 0 or 1 to six or more decimal places (i.e. 1.000000) will be present. Thus for a random sequence, only a small number of Pvalues should have a value near zero or one, although the presence of occasional Pvalues of 1.000000 or 0.000000 is insufficient to suggest that a sequence is non-random.

Accordingly, we established a scoring system for the DIEHARD results, where sequences were declared random provided that:

1. There were no more than a single "hard failure" (Pvalue = 1.000000 or 0.000000) for the entire sequence of Pvalues. Two failures were permitted if, and only if, both failures occurred in different varieties of a single DIEHARD test.

2. There were fewer than 5 “near” failure values (where a “near” failure is a Pvalue $<$ 0.000099 or Pvalue $>$ 0.999900)

If a sequence exhibited a small number of “hard failures” or more than 6 “near” values the results were considered as “indeterminate” and a retest was performed with a new dataset. If the retested data produced acceptable results, the generator was considered acceptable. Just to be certain, we also looked for “almost” values (Pvalue $<$ 0.099999 or $>$ 0.900000) and clusters of similar values. In general failed sequences had many “hard failures” to the point where few “near” or “almost” values exist in the entire sequence. Passed sequences almost never had a “hard failure” and very few “near” values.

Ten different chips were received from the CMOS manufacturing facility. Four distinct tests were performed to confirm the results. The first chip was tested twice for each XOR result at each voltage. Additionally, a second chip was tested using another prototype board and the results were compared and verified. While some DIEHARD results produced “hard failures”, there were never more than two of such failures in any given sequence. Occasionally, two “hard failures” occurred in the same test of the DIEHARD suite of tests. No repetition of “hard failures”, or “near failures” was found among the four test runs. The results obtained from the four distinct test runs show that the same design tested in identical configurations produced different and random results.

7 Results and Interpretation

As explained previously, six of the nine proposed designs failed to produce any nontrivial bits (the output remained at constant 1 or 0 value). One design (T9) produced results that failed to pass DIEHARD Tests. Two designs (T7 and T8) produced results that were random when all of the varieties were XOR’ed together. In addition, T7 produced random results from a single generator circuit at the nominal operating voltage of 1.8 volts (see Appendix A, Table 1). Since designs T7 and T8 were closely related, testing and analysis focused only on the T7 design.

In order to simulate real world conditions where gate delays can vary due to voltage, temperature and processing differences, the circuit was tested at various voltages as is shown in Appendix A, Table 2. In all cases the XOR of the 15 varieties of design T7 produced random sequences. Intuitively this seems to indicate that as some varieties became more biased, other varieties became less biased at different voltage levels. However all varieties produce independent, random results. The XOR appears to represent a collection of the entropy of all varieties, which can be explained as follows.

We define the bias b of a random binary variable X as

$$b = \left|\text{Prob}(X = 1) - \tfrac{1}{2}\right| = \left|\text{Prob}(X = 0) - \tfrac{1}{2}\right|$$

The T7 design uses the XOR of 15 binary sequences, each of which is believed to be random, as the output. According to the “Piling-up lemma” [9], if these input sequences are independent, the bias of the output sequence is

$$b = 2^{n-1}\prod_{i=1}^{n} b_i$$

Here n is the number of input sequences and b_i $(1 \le i \le n)$ is the bias of each of the input sequences. It can be easily shown that $b \le b_i$ $(1 \le i \le n)$ and the equality holds if and only if $b_i = 0$ or $b_i = ½$ for all $j \neq i$. In general, the XOR will greatly reduce the collective bias of the independent input sequences. For example, assume $n = 15$ and $b_i = ¼$ for all i, then $b = (½)^{16} \approx 0.000015$, which is negligible. As a practical matter the experiments show that even when none of the individual sequences passes the DIEHARD test, the XOR of the results is still measured as random. Since the cost of the circuit is quite small, a further reduction in bias can be achieved via XOR'ing two of these circuits together.

Further off-line processing of longer sequences from circuits that did not pass the DIEHARD tests provides an indication of the randomness of those circuits. Specifically, when a simple Von Neumann corrector was applied to long sequences that failed, the resulting shorter sequences passed the DIEHARD suite of tests. The Von Neumann corrector is used in many applications to remove bias [16], [17] at the expense of lower throughput. This shows that aside from bias, the results gathered from different varieties of T7 will provide good sources of random bits when properly debiased. This also seems to suggest that the effects of voltage variations on different varieties of T7 are most pronounced in the bias within the sequence. By XOR'ing different varieties of T7, the effects of bias are reduced to tolerable levels as shown in Appendix A, Table 3. In essence the XOR function has allowed us to trade off area, as more varieties of T7 are needed, for speed since the XOR'ed result does not need to be debiased and higher throughput is possible.

The total gate count for a random number generator based on the T7 design is as follows. Two AND gates, one inverter and one OR gate are used to implement each multiplexer. Thus a single randomness circuit requires four AND gates, four inverters and two OR gates. Since the design incorporates 15 instances of T7 and 14 XOR gates to collect the bits, the total number of gates is 60 AND gates, 60 inverters, 30 OR gates and 14 XOR gates. The small number of gates required to realize this random number generator makes the design suitable for applications that mandate strict power and area constraints.

8 Conclusions

In this paper, we have demonstrated that a practical random number generator can be built using standard digital gates and standard layout tools. The generator is stable even over large changes in operating voltage. This is a strong indication that such a generator will have good characteristics in extreme temperature conditions, such as may be found when a Smartcard is used at an outdoor Automatic Teller Machine (ATM) as well as resistance to attack by variation of voltage or temperature (side channel attacks).

The generator produces random bits by exploiting analog circuit artifacts found in common digital circuits. These artifacts are utilized by creating circuits that are noise sensitive, allowing naturally occurring semiconductor noise to determine the final output. Even though an individual circuit may not be perfectly unbiased, a relatively small number of similar circuits could be XOR'ed to produce a usable random result.

Additionally, we used different varieties of gates (all of which are standard library components) to immunize the overall design against expected environmental and process variations.

8.1 Recommendations for Future Work

Future work will involve the testing of the circuit over temperature extremes. Likewise, the circuit should be tested at greater switching frequencies to determine the maximum usable bit rate. Varying the duty cycle of the switch signal so that different “oscillation” and “resolution” times are applied to the circuit would also be interesting for all of the designs. Combining changes in temperature, voltage, and frequency may also yield interesting results.

Further work using de-biased versions of each generator would help, but not prove the independence of the different varieties. It is possible that even a simple von-Neumann corrector applied to a single variety would produce acceptable results. However, such a design may not have some of voltage immunity a combined design seems to have.

References

1. T.J. Chaney. Measured flip-flop responses to marginal triggering. IEEE Transactions on Computers, Vol.32 n.12, December 1983, pp. 1207-1209
2. G.R. Couranz and D.F. Wann. Theoretical and experimental behavior of synchronizers operating in the metastable region. IEEE Transactions on Computers, Vol.24 n.6, June1975, pp. 604-616
3. S.W. Golomb. Shift register sequences. 1967. Reprinted by Aegean Park Press in 1982
4. L. Kleeman, A. Cantoni. "Metastable Behavior in Digital Systems", IEEE Design and Test of Computers, vol. 4. Dec. 1987, pp. 4-19
5. L.R. Marino. “General theory of metastable operation”, IEEE Transactions on Computers, Vol.30 n.2, February 1981, pp. 107-115
6. H. Johnson. Random and deterministic jitter, EDN Magazine, June 27, 2002, pp. 24
7. M. Rowe, Jitter Discrepancies: not explained, EDN Magazine, February 6, 2003, pp. 48
8. B. Schneier. Applied Cryptography. Wiley & Sons, 2nd edition, 1995, pp. 425-426
9. M. Matsui, Linear Cryptanalysis Method for DES cipher, Advances in Cryptology - Eurocrypt'93, Lecture Notes in Computer Science, Springer-Verlag 765**,** 1993, pp. 386-397
10. M.J. Bellido, A.J. Acosta, et al., A simple binary random number generator: new approaches for CMOS VLSI. 35th MIDWEST Symposium on Circuits and Systems. August 1992
11. M. J. Bellido, A. J. Acosta, M. Valencia, A. Barriga,and J. L. Huertas,"Simple Binary Random Number Generator", Electronics Letters, Vol. 28, No. 7, March 1992**,** pp. 617-618

12. S. Walker and S. Foo, Evaluating metastability in electronic circuits for random number generation. IEEE Computer Society Workshop on VLSI, April 2001, pp. 99-102
13. L. Kleeman. The jitter model for metastability and its application to redundant synchronizers. IEEE Transactions on Computers, Vol.39. n.7, July 1990, pp. 930-942
14. L.M. Reyneri, L.M. del Corso, B. Sacco. "Oscillatory Metastability in Homogeneous and Inhomogeneous Flip-flops". IEEE J. of Solid-State Circ. Vol.25. n.1. Feb. 1990, pp. 254-264
15. Marsaglia, G. (1996) "DIEHARD: A Battery of Tests of Randomness," http://stat.fsu.edu/~geo
16. Robert Davies: Hardware random number generators, Statistics Research Associates Limited. http://robertnz.net/hwrng.htm
17. Benjamin Jun and Paul Kocher: The Intel® random number generator, Cryptography research, inc. April 22, 1999 http://www.connectedpc.com/design/chipsets/rng/CRIwp.pdf

Appendix A: Tabulated DIEHARD Results

Table 1. DIEHARD results for T1-T9 at 1.8 volts and room temperature

		Zone 1	Zone 2	Zone 3	Zone 4	Zone 5	Zone 6	Zone 7	Zone 8
T1		Stable result, non-random sequence							
T2									
T3									
T4									
T5									
T6									
T7	V1	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V2	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V3	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V4	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V5	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V6	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V7	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V8	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	FAIL	**PASS**
	V9	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V10	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V11	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V12	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V13	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V14	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	V15	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	XOR	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**
T8		Results similar to T7							
T9		Stable result, non-random sequence							

Table 2. DIEHARD results for T7 at room temperature and varying supply voltage

Variety	Supply Voltage	Zone 1	Zone 2	Zone 3	Zone 4	Zone 5	Zone 6	Zone 7	Zone 8
V7	1.2 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	1.4 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	1.6 volts	FAIL	FAIL	FAIL	**PASS**	FAIL	FAIL	FAIL	FAIL
	1.8 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	2.0 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
V8	1.2 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	1.4 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	1.6 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	1.8 volts	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	FAIL	**PASS**
	2.0 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
V9	1.2 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	1.4 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	1.6 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	1.8 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	2.0 volts	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
XOR	1.2 volts	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**
	1.4 volts	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**
	1.6 volts	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**
	1.8 volts	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**
	2.0 volts	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**

Table 3. Von Neumann correction of variety 7 and 9 for T7 at 1.8 volts. Table shows the DIEHARD test results for biased and debiased sequence, and the reduction in the size of the debiased sequence after passing the baised sequence through the Von Neumann corrector

		Zone 1	Zone 2	Zone 3	Zone 4	Zone 5	Zone 6	Zone 7	Zone 8
V7	Biased	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	Debiased	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	FAIL	**PASS**	**PASS**
	Data Reduction	14.87%	11.85%	12.23%	10.64%	24.77%	23.79%	24.39%	10.46%
V9	Biased	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL	FAIL
	Debiased	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**	**PASS**
	Data Reduction	22.35%	24.56%	19.12%	24.85%	25.10%	25.08%	21.98%	22.16%