A 3 μW CMOS True Random Number Generator With Adaptive Floating-Gate Offset Cancellation

Jeremy Holleman, *Student Member, IEEE*, Seth Bridges, *Member, IEEE*, Brian P. Otis, *Member, IEEE*, and Chris Diorio, *Member, IEEE*

***Abstract*—This paper presents two novel hardware random number generators (RNGs) based on latch metastability. We designed the first, the DC-nulling RNG, for extremely low power operation. The second, the FIR-based RNG, uses a predictive whitening filter to remove nonrandom components from the generated bit sequence. In both designs, the use of floating-gate memory cells allows us to predict and compensate for DC offsets and other nonrandom influences while minimizing power consumption. We also present an efficient digital post-processing technique for improving randomness. We fabricated both RNGs in a standard 0.35 μm CMOS process. The DC-nulling RNG occupied .031 mm^2 of die area, while the FIR-based RNG occupied 1.49 mm^2.**

***Index Terms*—Cryptography, floating gates, noise, random number generator (RNG), wireless sensor networks.**

I. Introduction

RANDOM number generators (RNGs) are used in many applications, including wireless sensor nodes [1], radio-frequency identification (RFID) [2], and cryptography [3]. Pseudo-random number generators (PRNGs) can have excellent statistical properties and are easily implemented in digital logic. However, they require a seed from a true-random number generator (TRNG), which derives its randomness from a physical noise source, such as thermal noise [4]. Additionally, knowledge of a PRNG's internal state and structure enables perfect prediction of the bit sequence, so for cryptographic security, a PRNG should be periodically reseeded from a TRNG [3]. For these reasons, TRNGs are critical building blocks for communication systems.

Given the increasing drive towards systems-on-a-chip (SOCs), it is desirable to find TRNGs that can be completely integrated onto a CMOS IC. For many applications, particularly wireless sensors and RFID, it is also critical that power consumption be kept to a minimum. Bit rate requirements are typically low. Therefore, we designed our TRNG to operate with very low power and at relatively low frequency.

Manuscript received May 22, 2007; revised January 2, 2008. This work was supported by the National Science Foundation (CCR-0086032), the Packard Foundation (2000-01763), and the Office of Naval Research (N00014-01-1-0676).

J. Holleman and B. P. Otis are with the Department of Electrical Engineering, University of Washington, Seattle, WA 98195 USA (e-mail: hollemj@ee.washington.edu).

S. Bridges was with the Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195 USA, and is now with V2Green, Inc., Seattle, WA 98109 USA.

C. Diorio is with the Department of Computer Science and Engineering, University of Washington, Seattle, WA 98195 USA.

Digital Object Identifier 10.1109/JSSC.2008.920327

In this paper, we describe two TRNGs, first introduced in [5], which use a latch to sample thermal noise. Both circuits use floating-gate memory cells [6] to calculate and store parameters of the offset and noise distribution in order to compensate for nonrandom effects and improve the quality of the resulting bit sequence. The first design, the DC-nulling RNG, applies an adaptive bias to the latch to compensate for constant (DC) and slowly-varying biases and ensure a uniform bit distribution. The second, the FIR-based RNG, uses a novel implementation of a mixed-signal finite-impulse-response (FIR) filter to remove correlated noise sources before quantization in order to maximize the entropy of the resulting bit sequence. We also present an efficient digital post-processing algorithm to further improve the statistical properties of the bit sequence.

In Section II, we present an overview of existing methods for true-random number generation. We describe our proposed designs in Section III. The post-processor is described and analyzed in Section IV. Test results are presented in Section V and in Section VI we give our conclusions.

II. Random Number Generator Background

Random number generators can be broadly divided into two categories. Pseudo-random number generators (PRNGs) are deterministic processes designed to generate sequences that appear random. True random number generators (TRNGs) generate sequences from physical phenomena, such as nuclear decay, thermal noise, cosmic radiation, or keyboard strokes. PRNGs are widely used because they exhibit excellent statistical properties and allow easy digital implementations, but require a TRNG as a source of entropy.

Most integrated TRNGs use one of three basic methods for generating random numbers from a thermal noise source. In this section we will describe the most common techniques for extracting entropy from a thermal noise source. An overview of different types of TRNGs is also presented in [7] and [8].

A. Direct Amplification

Direct amplification is the most intuitive technique, wherein the output of a noise source is amplified, sampled, and quantized. In the classical implementation, the terminal voltages of a resistor drive an amplifier, and the amplified noise is then sampled and quantized using a discrete-time comparator. Such an implementation requires an amplifier with high gain, low $1/f$ noise, low offset, and low sensitivity to conditions such as temperature and supply noise, which results in high power dissipation. While there have been integrated implementations [9], the direct amplification technique is most often used in stand-alone RNGs, such as [10].

B. Latch-Based TRNGs

In a variation of the direct amplification technique, a latch is used to sample the noise. During the reset phase, both outputs of the latch are forced to the same voltage. In the evaluation phase, positive feedback is enabled, which drives one output high and the other low. The latch performs the amplification, sampling, and quantization functions in one step, allowing a compact low-power design.

The latch can be reset by shorting the two outputs to each other [11], [12] or by connecting both outputs to a supply, as in [13] and this work. If the two outputs are connected to each other, the latch is placed at its maximum-current bias point, resulting in high power dissipation; however, such excessive power dissipation can be avoided by collapsing the supply rails in the reset phase [14]. In either case, device mismatch can bias the latch evaluation, resulting in a nonuniform bit sequence. Therefore, most latch-based TRNGs incorporate some means of adjusting the relative strengths of the two halves of the latch, either by injecting a differential current onto the two output nodes [12] or by applying a differential current-limiting voltage, as in [13], [14], and this work.

C. Oscillator Sampling

TRNGs using the oscillator sampling technique sample a fast clock with a much slower clock. Phase noise in either of the clocks provides uncertainty of the sampled value. To achieve adequate randomness in the output sequence, the ratio of the standard deviation of clock jitter to mean fast clock period must be sufficiently high, which typically requires that amplified transistor noise be applied to the control terminal of a voltage-controlled oscillator [15], [16].

D. Discrete-Time Chaos

Discrete time chaos (DTC) RNGs use a pipelined structure where each stage performs a non-linear mapping and adds noise to the signal. The system is very similar to a pipelined analog-digital converter, with the output of the last stage connected back to the input of the first stage. The input to each stage is quantized to one bit, $b = \pm 1$, shifted up or down based on b, and multiplied by the stage gain, typically 2. The mapping operation can be implemented with either a switched-capacitor circuit [17] or a switched-current circuit [18], [19].

The strategy for our proposed TRNG is to use a compact, power-efficient, latch-based RNG to sample noise. We use floating-gate circuits to compensate for nonrandom influences such as mismatch and correlated noise. A post-processor then removes residual correlations to further improve the statistical quality of the final sequence.

III. System Design

Thermal noise is a good source of randomness because of its uniform spectrum. Unfortunately, due to its relatively low amplitude, it is easily corrupted by nonrandom influences, including supply noise, $1/f$ noise, interference, and device mismatch. Our RNG, like all other TRNGs, samples the sum of both the predictable and unpredictable noise sources. To remove the predictable component, both of our designs apply a predictive signal to the evaluation circuit as a compensating voltage, leaving only the unpredictable white noise component to generate the bit sequence. The DC-nulling RNG uses a running estimate of the median total noise as the prediction. The FIR-based RNG uses a linear-prediction filter to remove correlations between bits separated by short time intervals.

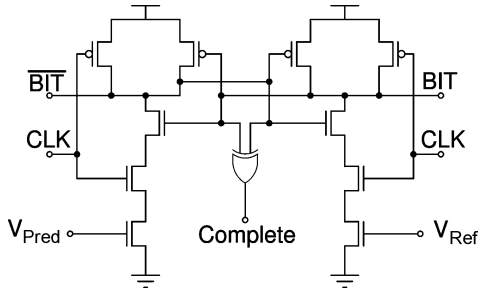

Fig. 1. Schematic of the bit-generating latch.

A. Evaluation Latch

At the core of both RNGs is the evaluation latch, shown in Fig. 1, comprising a pair of cross-coupled, current-starved NAND gates. The latch functions as a discrete-time comparator, with a prediction signal $\mathrm{V_{Pred}}$ and a reference signal $\mathrm{V_{Ref}}$ as the positive and negative inputs, respectively. During the reset phase, when CLK is low, both latch outputs, BIT and $\overline{\mathrm{BIT}}$, reset to $\mathrm{V_{DD}}$. During the evaluation phase, CLK is high and positive feedback will drive one output to 0 and the other to 1. The predictive bias signal $\mathrm{V_{Pred}}$ is applied to one of the current-limiting nFETs, and a reference voltage is applied to the other. By applying different current-limiting voltage inputs to the two halves of the latch, the latch offset can be adjusted to achieve the desired distribution of output bits.

An XOR gate compares the two latch outputs to verify that initial metastability has been resolved. Due to the relatively slow clock frequency ($\leq$ 100 kHz), the probability of a metastable state lasting throughout a clock period is exceedingly low. We examined the XOR output for 100 million bits at 100 kHz and found no unresolved metastable events.

B. Memory Cell

Because the two TRNGs use feedback to calibrate latch offset and remove low-frequency artifacts based on previous bits, we need a mechanism to calculate and store statistics of the bit sequence. For this purpose we use a floating-gate memory cell [6]. The memory cell, shown in Fig. 2, provides non-volatile analog storage. Values are stored as charge on a floating gate, and adjusted using Fowler–Nordheim tunneling [20] and hot-electron injection [21].

Decrements to $\mathrm{V_{Out}}$ are accomplished by tunneling through the tunneling junction $\mathrm{M_{Tun}}$. The charge pump provides the high voltage (about $2\mathrm{V_{DD}}$) necessary to cause tunneling. Increments to $\mathrm{V_{Out}}$ are accomplished by conducting current through $\mathrm{M_{Inj}}$, while bringing its drain to ground, to cause electrons to be injected from the channel onto the floating gate $\mathrm{FG_{Mem}}$.

Because the adjustment mechanism is part of the feedback path that maintains the correct bit distribution, the increment and decrement magnitudes must be balanced. Suppose the differential input to a comparator is adjusted each clock cycle based on

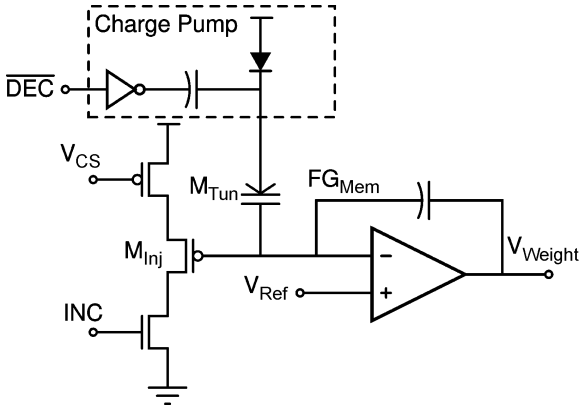

Fig. 2. The floating-gate memory cell.

the bit output. The input is incremented by an amount μ^+ or μ^- when a 0 or 1 is output, respectively. If p represents the probability that an individual bit is a 1, then the average change per clock cycle in the comparator input is

$$\Delta V_{\text{in}} = (1-p)\mu^+ - p\mu^-. \tag{1}$$

V_{in} converges to the value where the expectation of ΔV_{in} is zero, which corresponds to a bit distribution of

$$p = \frac{\mu^+}{\mu^+ + \mu^-}. \tag{2}$$

Mismatch in components or paths can cause the increment and decrement magnitudes to be unequal and the bit distribution to deviate from its ideal value.

To ensure that the updates are symmetric, the tunneling and injection rates can be independently adjusted. The voltages $\mathrm{V_{Ref}}$ and $\mathrm{V_{CS}}$ determine the tunneling and injection rates, and are stored on separate floating gates. Negative feedback keeps the floating gate voltage equal to $\mathrm{V_{Ref}}$, so that the magnitude of updates is independent of the output voltage. Using a 3-bit shift register, each memory cell can be configured to adapt based on the RNG output, store a pre-programmed voltage, programmed externally to a desired voltage, or disabled to save power.

C. DC-Nulling RNG

In the DC-nulling RNG, shown in Fig. 3, $\mathrm{V_{Ref}}$ is adjusted to the desired common-mode level before operation. $\mathrm{V_{Pred}}$ is incremented or decremented by a fixed amount μ whenever a 0 or 1 is generated, respectively. Because a higher value for $\mathrm{V_{Pred}}$ increases the likelihood of a 1, $\mathrm{V_{Pred}}$ converges to the value where the bits are uniformly distributed. Previous latch-based RNGs (e.g., [12]), have used switched-capacitor circuits to cancel DC biases. The floating-gate memory cell is more power efficient, because it avoids the slew-rate and bandwidth requirements of switched-capacitor circuits. It also enables extremely low adaptation rates; faster adaptation can filter out too much low-frequency noise, spectrally coloring the bit sequence.

The adaptation signals Dec and Inc are gated by the clock, as shown in Fig. 3, so adjustments to the predictive signal take place during the evaluation phase. The reset phase is available for the associated transients to settle. If adaptation begins before

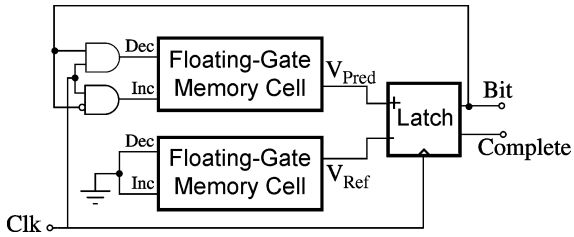

Fig. 3. The DC-nulling RNG.

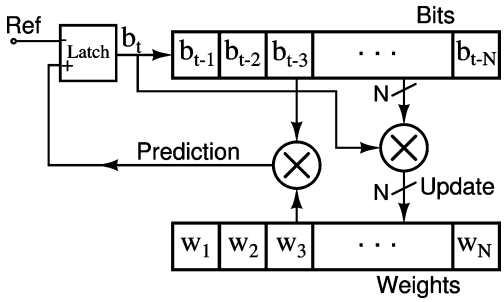

Fig. 4. Symbolic model of the FIR-based RNG.

the latch completes the evaluation, it is possible that coupling from the adaptation circuit could affect the evaluation. To protect against this and other interference, the outputs of the latch are buffered by two inverters in series.

In Fig. 3, the positive input $\mathrm{V_{Pred}}$ is configured to function adaptively and the negative input $\mathrm{V_{Ref}}$ is fixed. Using the configuration shift register described above, the DC-nulling RNG can be configured to operate with either, neither, or both memory cells adapting. There is no common-mode feedback, so if both inputs adapt, any asymmetry in the adaptation rates will cause the two signals, $\mathrm{V_{Pred}}$ and $\mathrm{V_{Ref}}$, to drift and eventually saturate.

The DC-nulling RNG could also be used to implement automated calibration of the floating-gate memory cell update rates. Because the distribution of bits is related to the tunneling and injection rates through (2), the bit distribution can be used to monitor and periodically adjust the calibration. Automated calibration would enable the RNG to autonomously adjust to changing environmental conditions, such as supply and temperature.

D. FIR-Based RNG

The DC-nulling RNG corrects for latch offset to maintain the correct distribution of ones and zeros. We now focus on a technique that removes correlations in the bit sequence, to maximize independence of subsequent bits. The FIR-based RNG, shown in Fig. 4, uses a linear-prediction filter to remove correlated components of the noise before sampling. The total noise is predicted based on previous bits and knowledge of the noise correlation structure. Previous bits are stored in a shift register such that at time t, the RNG output from time $t-k$ is stored in tap k. We use floating-gate memory cells to calculate and store the filter tap weights. The weights can be adjusted at every clock cycle according to the instantaneous autocorrelations at the corresponding lags:

$$w_0(t+1) = w_0(t) + b(t)\mu \tag{3}$$

$$w_k(t+1) = w_k(t) + b(t-k)b(t)\mu \tag{4}$$

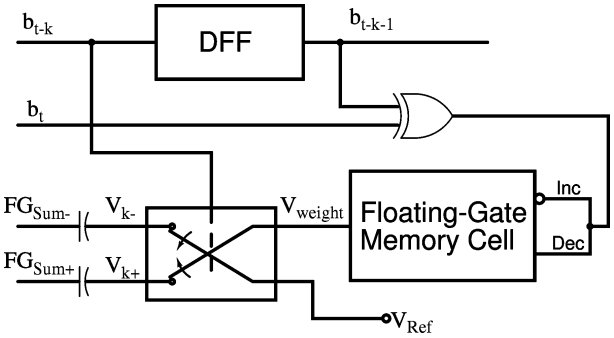

Fig. 5. A single tap of the FIR-based RNG.

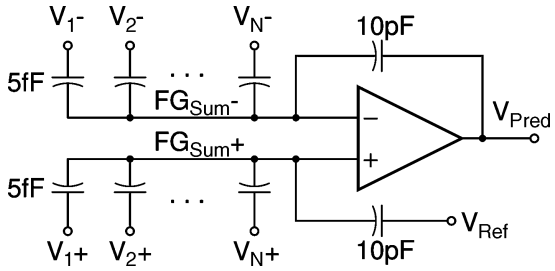

Fig. 6. Capacitively-coupled differential summation circuit.

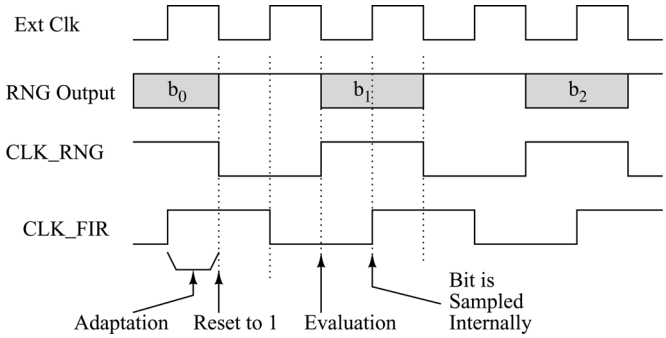

Fig. 7. Timing of the FIR-based RNG.

where $w_k(t)$ is the k^{th} tap weight at time t, and $b(t)$ is the RNG output at time t. One tap weight (w_0) is adjusted based on the current bit alone to account for DC bias. The schematic of a single tap with the update circuitry is shown in Fig. 5. As long as the adaptation is symmetric (i.e., the increment and decrement magnitudes are equal), the weight for a given tap will converge to the value that causes the autocorrelation at the corresponding lag to be zero. The bits are interpreted as ± 1, allowing the 1-bit multiplication $b(t-k)b(t)$ to be performed by an XNOR gate. The output of the XNOR gate determines whether the memory cell should be incremented or decremented. Tap weights w_{1-N} are multiplied by ± 1 using a cross-bar switch controlled by the bit stored in the respective tap, and the outputs of all taps are summed to form the prediction signal. The result is that the prediction signal is the inner product of the N tap weights with the past N bits, plus a term to remove constant (or very slowly changing) bias

$$\mathrm{V_{Pred}}(t) = w_0 + \sum_{k=1}^{N} w_k(t) b(t-k). \tag{5}$$

The prediction signal is applied to the latch input $\mathrm{V_{Pred}}$ to remove correlated noise components. It should be noted that the FIR filter is not a post-processor, such as the one described in Section IV, but an integral part of the TRNG. A post-processor operates on the output of a TRNG, so the resulting sequence can contain no more entropy than the unprocessed sequence. In contrast, the FIR filter operates on the noise before it is quantized, and can therefore increase the entropy in the bit sequence.

We implemented the summation operation using the circuit shown in Fig. 6, which is a well-known summation circuit, modified to use capacitors. Both inputs of the op-amp are floating-gate pFETs, with a DC level set by tunneling and injection. The use of floating gates on the input nodes allows the summing circuit to operate at arbitrarily low frequencies while using purely capacitive feedback. The feedback capacitors are 10 pF double-poly capacitors. The input capacitors connecting each tap weight to $\mathrm{FG_{Sum\pm}}$ are 5 fF pMOS capacitors. The capacitor ratio was chosen to allow the full range of tap outputs without saturating the amplifier output.

The adaptation can compensate for any error in the size of the input capacitor. If the capacitor is too small, the tap weight will adapt to a larger voltage in order to couple the appropriate amount of charge onto the floating gates. Likewise, linearity of the input capacitors is not a concern because the voltages of the two floating nodes $\mathrm{FG_{Sum\pm}}$ are nearly constant. The op-amp keeps the two voltages close to equal, and the high gain of the latch means that the required voltage range on the floating gates is small. For example, to change the RNG from generating 75% 0's to 75% 1's only requires a change in $\mathrm{V_{Pred}}$ of about 6 mV.

At the input, the external clock is split into two clocks, separated by 90°. As shown in Fig. 7, one clock controls evaluation of the RNG, while the other clocks the shift register for the FIR, an arrangement which prevents digital noise from the shift register from affecting the evaluation of the RNG. The FIR clock also samples the output of the RNG and initiates adaptation. Because adaptation does not begin until the latch output is sampled (one fourth of a clock period after evaluation begins) the adaptation circuits do not affect the evaluation. Each tap can be configured or disabled using an externally programmed shift register.

Compared to a previously presented mixed-signal FIR filter utilizing floating gates [22], the filter presented here allows for reduced auxiliary hardware by using single-bit updates. The capacitive summation circuit also allows our filter to provide a voltage output rather than a current output. While the finite gain of the summation amplifier may limit linearity, the small signal swings needed to shift the bit distribution of the RNG reduce the linearity requirement for the filter.

IV. The Post-Processor

Device mismatch, $1/f$ noise, and interference typically cause TRNGs to produce sequences with insufficient randomness for cryptographic applications [12], [13], [17]. Most commonly, the bit distribution will be slightly biased, and bits that are separated by short time intervals will be correlated with each other.

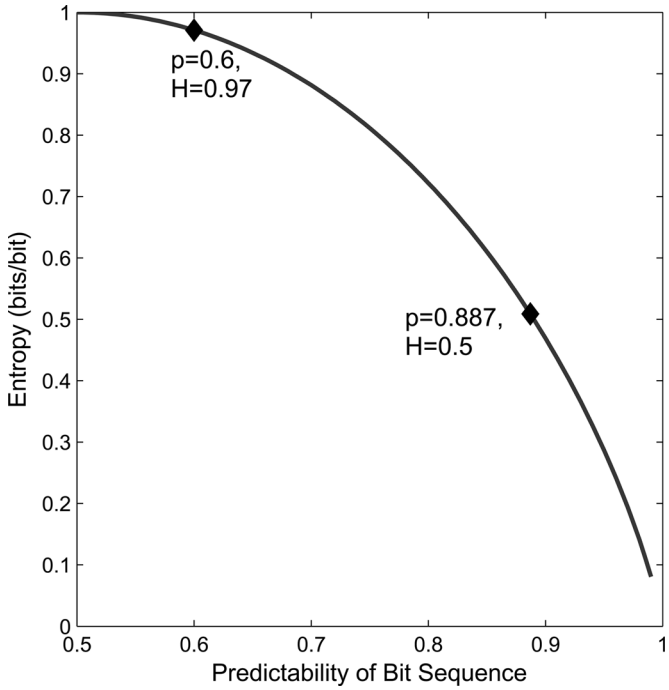

Fig. 8. As a sequence becomes more predictable, its entropy decreases, but even very predictable sequences have well over 1/2 bit of entropy for each bit in the sequence.

We have developed a post-processor which uses a pair of nonlinear-feedback shift registers (NFSRs) to remove residual nonrandomness in the output stream at the expense of reducing the output rate by a factor of two. The post-processor is implemented off-chip, but an estimate of its power consumption based on Spectre simulations is included in the power consumption figures reported in Section V.

A. Information Theory

While many post-processors reduce the output rate by a factor of 6 or more, information theory suggests that perfectly random bit sequences can be constructed from low-quality sequences with decimation ratios as low as 2. The entropy in a bit is defined as

$$H = -\left(p\log_2(p) + (1-p)\log_2(1-p)\right) \tag{6}$$

where p is the probability that the bit takes on a specific value, 0 or 1 [23]. If we assume that an attacker can predict the RNG output with an accuracy p, then (6) relates the entropy of a bit sequence to its predictability. The relationship between predictability and sequence entropy, plotted in Fig. 8, implies that even when an unprocessed bit sequence can be predicted with 88% accuracy, there is still 1 bit of entropy for every two bits in the original sequence. For an RNG with 60% predictability, the sequence contains 0.97 bits of entropy for each unprocessed bit. This result implies that even weakly random sequences can theoretically be processed to yield perfectly random sequences. It also suggests that TRNG designers facing severe power and area constraints might focus on developing post-processing methods to efficiently extract entropy from low-quality bit sequences, rather than expend die area and power to improve the quality of unprocessed sequences.

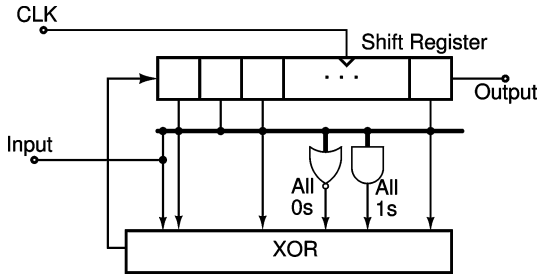

Fig. 9. The modified NFSR.

B. Other Post-Processors

Post-processing techniques presented in the past include parity/XOR, the von Neumann corrector, and hash functions. Parity post-processors, which simply take the parity of non-overlapping blocks of bits, are very compact and simple, but may require large decimation ratios to effectively remove bias [12]. The von Neumann corrector [24], discards pairs of equal bits and keeps the first bit of unequal pairs. In [16] the von Neumann corrector reduced the output rate by an average factor of 6, but the output rate is dependent on the input. Post-processors using hash functions [4] can have arbitrary decimation ratios, but the computational requirements are relatively high.

Previous shift-register-based post-processors include work by Acosta, *et al.* [25] and Poli, *et al.* [26]. Acosta, *et al.* integrated one TRNG into each stage of an eight-stage linear-feedback shift register (LFSR), resulting in a decimation ratio of eight (i.e., one processed bit is generated for every eight evaluations of the TRNG). The post-processor used by Poli, *et al.* uses three shift registers and feedback logic based on the elementary operations of the secure hash algorithm (SHA). The decimation ratio is one, which improves the output rate for a given power dissipation, but if the post-processor function is invertible, then the input can be reconstructed from the output, compromising the security of the sequence. We chose the topology described here for its combination of simplicity, robust security and low decimation ratio.

C. Nonlinear-Feedback Shift Register

Our post-processor is based on a modified nonlinear-feedback shift register (NFSR), shown in Fig. 9. It is similar to a standard linear-feedback shift register (LFSR) with extra terms added to the feedback equation to accept an input and to provide for the generation of pseudo-random bits even in the case a malfunctioning TRNG. The feedback bit is given by

$$s_{1,t+1} = \mathrm{mod}2\left(\sum_{k=1}^{N} c_k s_{k,t} + x_t \prod_{k=1}^{N-1} s_{k,t} + \overline{x_t} \prod_{k=1}^{N-1} \overline{s_{k,t}} + x_t\right) \tag{7}$$

where $s_{i,t}$ denotes the contents of the ith register at time t, c_i is the ith term of the feedback polynomial, and x_t is the output of the latch at time t. The two product terms prevent the NFSR from becoming "stuck" in an all ones or all zeroes state

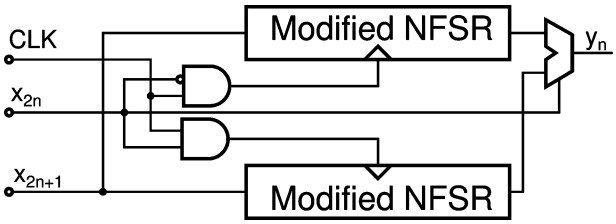

Fig. 10. The alternating-step post-processor.

in the case of TRNG tampering or malfunction. See [4] for an overview of feedback shift registers.

Simulations show that the NFSR can remove large biases and correlations in the bit sequence. However, an observer who knows the feedback polynomial will be able to reconstruct the random input sequence from the output, allowing future outputs to be predicted with the same accuaracy as future inputs. Because it is impossible to add entropy to the sequence, the output bit rate must be smaller than the input bit rate in order to improve security.

Our post-processor is constructed of two modified NFSRs arranged in an alternating-step configuration [4], wherein only one shift register is clocked at each time step. At each clock cycle, a pair of bits from the latch-based RNG is taken and used to generate a single bit from the post-processor. The first of each pair is used to determine which NFSR is clocked and muxes the output of that NFSR to the final output. The second bit of the pair is fed into the clocked NFSR to perturb its state, as described above. Because the two NFSRs are being randomly clocked, it is not possible to directly infer the input sequence (x_{2n+1} in Fig. 10) from the output sequence. It would be possible to double the bit efficiency of the post-processor by using a sliding 2-bit window to use each unprocessed bit once in each of the roles just described—clock control and state perturbation. While statistical measurements of the bit sequence, such as mean and autocorrelation function, would not suffer, the entropy in the processed sequence could be no greater than in the raw sequence, compromising the security of the sequence.

D. Security

The approach taken by most known attacks on FSR-based generators [27]–[29] is to reduce the set of possible states for some starting time t_0 based on the sequence observed since t_0, eliminating initial states that could not have produced the observed sequence. Then, with knowledge of the state at time t_0 and the observed sequence since time t_0, the attacker can track the evolution of the state and predict the output at any time $t > t_0$. Such attacks rely on the fact that the initial state of an LFSR determines its output completely. Our post-processor is resistant to such attacks, because the NFSR's state is constantly perturbed by bits from the TRNG. Because two bits from the latch are used to generate each processed bit, an observer's knowledge of the post-processor's internal state will decrease over time, even if the initial state is known completely (see Appendix). Prediction of the processed bits relies directly on knowledge of the internal state, so an observer's ability to predict the bit sequence will decrease as the post-processor continues to run.

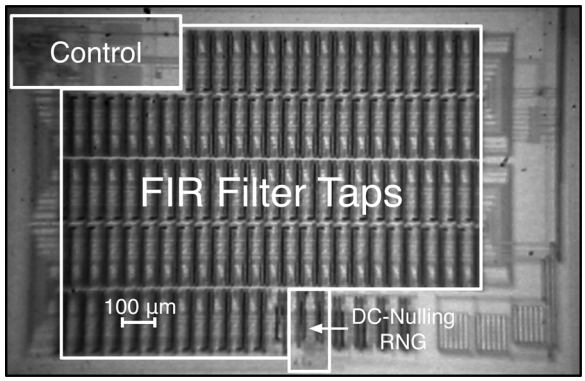

Fig. 11. Microphotograph of the RNG. The filter taps for the FIR-based RNG occupy the majority of the area. The area labeled "control" includes the summing amplifier and feedback capacitors for the FIR filter summation circuit as well as control logic.

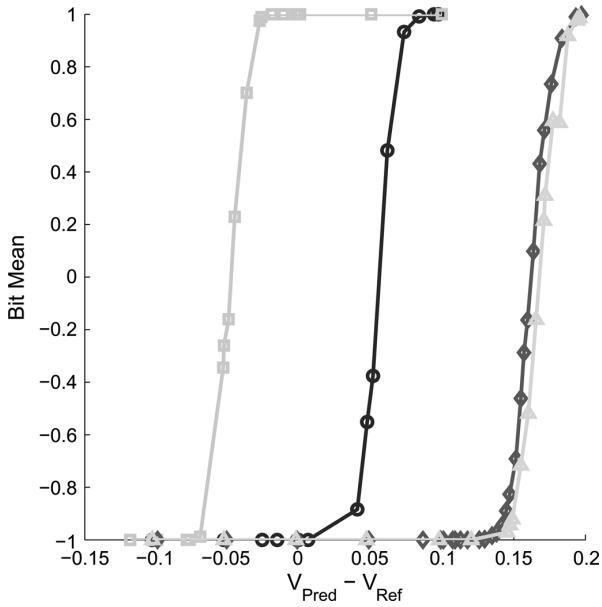

Fig. 12. With $\mathrm{V_{Ref}}$ held at 3.25 V, $\mathrm{V_{Pred}}$ is swept to measure the bit distribution as a function of differential input to the latch for each of four different chips. The bits are counted as ± 1, so a mean of zero corresponds to an ideal distribution.

V. Results

We fabricated both RNGs in a 0.35 μm, four-metal, double-poly process available through MOSIS. A microphotograph of the chip is shown in Fig. 11. The DC-nulling RNG consumes 0.031 mm^2 of die area and can be fabricated in a standard digital process, while the FIR-based RNG occupies 1.69 mm^2, and uses double-poly capacitors in the differential summing circuit. Both RNGs run on a single 5 V supply.

A. DC-Nulling RNG

The transfer function for the latch is shown for several different chips in Fig. 12. $\mathrm{V_{Ref}}$ was held constant at 3.25 V and $\mathrm{V_{Pred}}$ was swept to find the threshold, the point where the output bits are evenly distributed.

To test the convergence of the DC-nulling RNG, we programmed $\mathrm{V_{Ref}}$ to 3.2 V to set the common-mode level. In the two trials of this experiment, we programmed $\mathrm{V_{Pred}}$ to 3.7 V

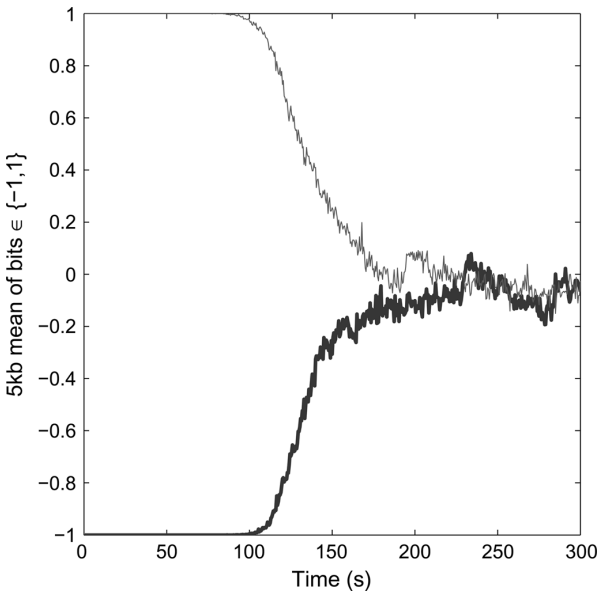

Fig. 13. The DC-nulling RNG converging to a near-uniform distribution after being programmed to output all 1's or all 0's.

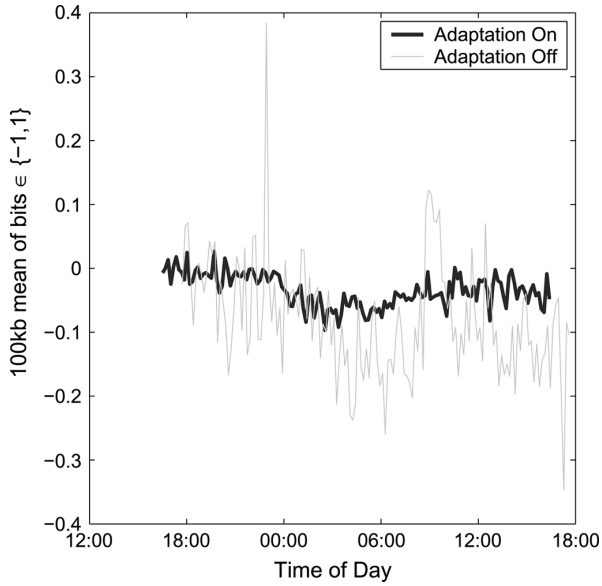

Fig. 14. When the DC-nulling RNG runs for 24 hours with adaptation disabled, the bit distribution varies greatly. When adaptation is enabled, the distribution is kept much closer to zero.

and 2.7 V, so the RNG would output all ones and all zeros, respectively. We enabled adaptation and allowed the RNG to converge. The results are displayed in Fig. 13. In both cases, the mean of the bit sequences converges to close to zero. The mean still varies from an ideal of zero because of $1/f$ noise and, perhaps, other slowly-varying interference sources, which change too quickly for the DC-nulling RNG to compensate.

The advantage of the adaptation used in the DC-nulling RNG is that the RNG can adapt to changing conditions and keep the bit distribution even. To test this feature, we first programmed V_+ externally to a value that produced a zero-mean sequence and ran the RNG for 24 hours with adaptation disabled. We then repeated the experiment with adaptation enabled. Fig. 14 shows the means of 100 kb blocks, taken every 10 minutes in both trials. With adaptation enabled, the block means stay much closer to zero. The residual variation in the distribution is due to the temperature dependence of the tunneling and injection processes.

B. FIR-Based RNG

Fig. 15 illustrates the effectiveness of the FIR-based RNG at removing correlations. The black trace shows the correlations present in bits from the DC-nulling RNG. In the dark gray trace, the FIR filter, with 25 taps enabled, has removed auto-correlations for lags up to 25. There is a small positive correlation at lags of 42 and 90 due to a layout artifact where the floating polysilicon node of the summation circuit crosses under a wire holding the current bit. The light gray trace shows that the 100-tap FIR has compensated for all three effects. The data in this experiment were taken with the tap weights preprogrammed to the correct values. The RNG was run with adaptation disabled because asymmetry in the weight updates prevented the adaptive filter from converging to the correct weights.

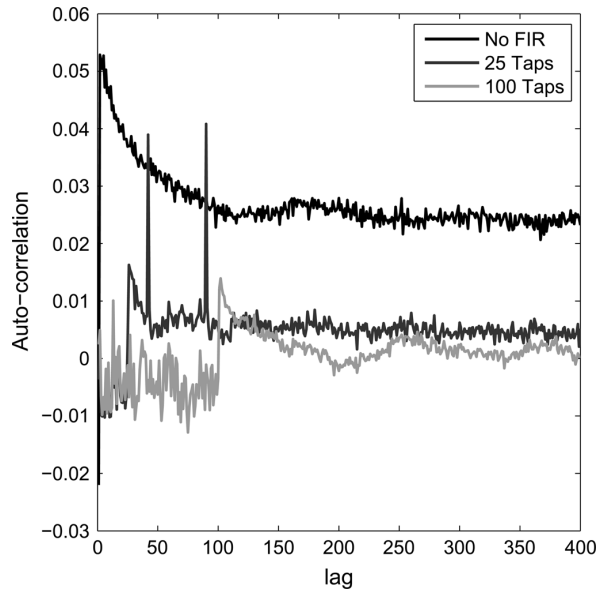

Fig. 15. Auto-correlation sequences for bits taken with the DC-nulling RNG, and the FIR-based RNG with 25 and 100 taps enabled.

C. Temperature and Supply Rejection

While our TRNGs have demonstrated good statistical performance, they are susceptible to environmental variations or long-term device degradation, which can introduce errors in the calibration of the floating-gate memory cells' update rates. This will cause the RNG to converge to an incorrect bit distribution, as described in (2), for the DC-nulling RNG. In the same way, miscalibration in the FIR-based RNG leads to non-zero autocorrelations.

Fig. 16(b) shows the effect of supply voltage on the mean of the bit sequence for the DC-nulling RNG. Decrements to the memory cell are accomplished through Fowler-Norheim tunneling, resulting in a gate current which is exponentially related to the voltage across the gate. The tunneling voltage is derived

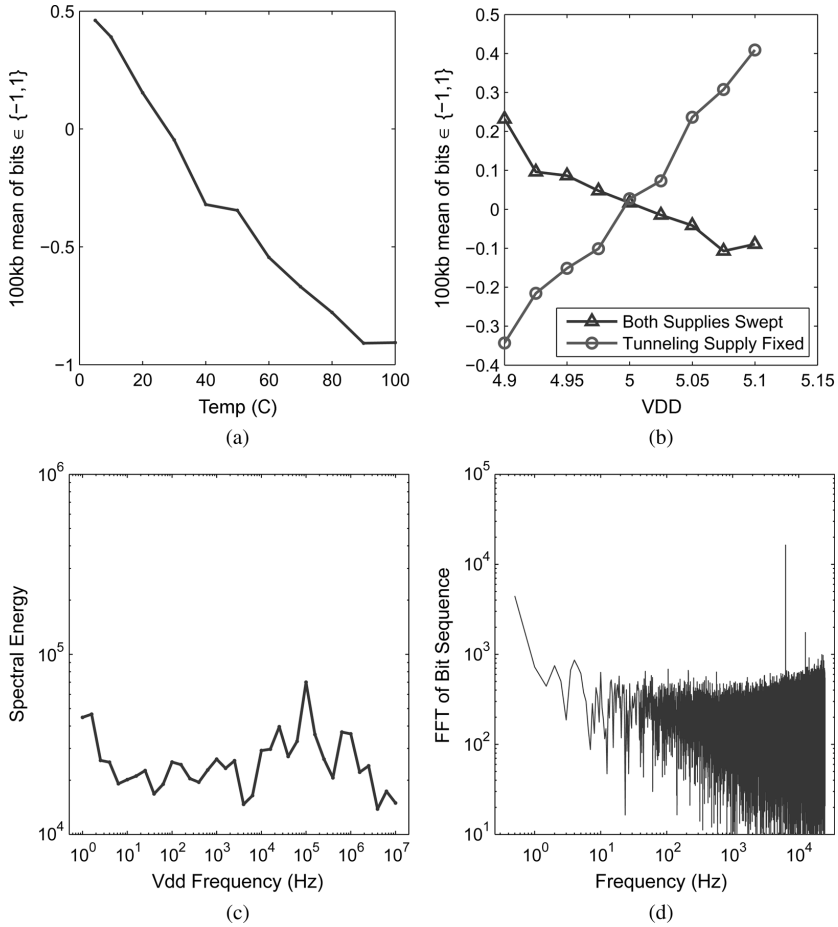

Fig. 16. (a) Mean of the output sequence as a function of temperature. (b) Sequence mean as a function of the supply voltage, with the tunneling supply $\mathrm{V_{DD}}$. (c) Frequency response of the bit sequence with respect to a 200 m $\mathrm{V_{P-P}}$ sine wave on $\mathrm{V_{DD}}$. The combined energy of the fundamental and the second harmonic is plotted against the frequency at which $\mathrm{V_{DD}}$ is varied. (d) Spectrum of the bit sequence when the supply is modulated with a 6.3 kHz, 200 m $\mathrm{V_{P-P}}$ sine wave.

from the supply by the charge pump shown in Fig. 2. Therefore, as the supply to the tunneling circuit increases, the strength of the decrements (μ^-) will increase rapidly, resulting in more zeros and a lower sequence mean, as can be seen in Fig. 16(b). Increments to the memory cells are accomplished through hot-electron injection, which is roughly quadratic with respect to the drain current in the transistor where injection occurs [21]. The injection current is supply-referenced, so an increase in the supply voltage will result in stronger increments (μ^+ increases), and the sequence mean will increase. When both supplies are varied, the increased tunneling rate overwhelms the increased injection rate. However, when the tunneling supply is fixed, the sequence mean increases with supply voltage, as shown in Fig. 16(b). To measure the effect of high-frequency supply noise on the RNG, we modulated $\mathrm{V_{DD}}$ with a 200 $\mathrm{mV_{P\text{-}P}}$ sine wave, with the frequency swept from 1 Hz to 10 MHz. The supply disturbance is evident primarily at the frequency of supply modulation and its second harmonic, as can be seen in Fig. 16(d), which shows the Fourier transform of a bit sequence with a supply frequency of 6.3 kHz. The combined energy of the spectrum of the bit sequence at the fundamental frequency and its second harmonic is plotted against the frequency of supply modulation in Fig. 16(c). As with other non-idealities, the post-processor removes artifacts resulting from supply noise.

Temperature variations affect the sequence mean primarily through the drain current of the injection device. As the temperature increases, drain current and injection rate (μ^+) decrease, causing the sequence mean to decrease. This relationship is shown in Fig. 16(a). The use of a regulated current source for injection, and a stable reference voltage for the tunneling supply will significantly improve temperature and voltage stability in future implementations. While the unprocessed bit sequences deviate significantly from the desired mean, the entropy in the sequence is sufficient to pass the NIST statistical tests after post-processing for supply voltages between 4.9 V and 5.1 V, and for temperatures between 0 °C and 80 °C.

D. Statistical Evaluation

For random sequences generated by hardware RNGs, the most relevant tests are the mean and the autocorrelation function of a bit sequence. To aggregate the mean and autocorrelation function into one figure of merit, we use a measure of total linear predictability, defined as the fraction of bits that are correctly predicted by a linear regression of the preceding M bits. As a longer history of previous bits is used, the prediction becomes more accurate. Fig. 17(a) shows how the measured predictability increases as a longer history is used

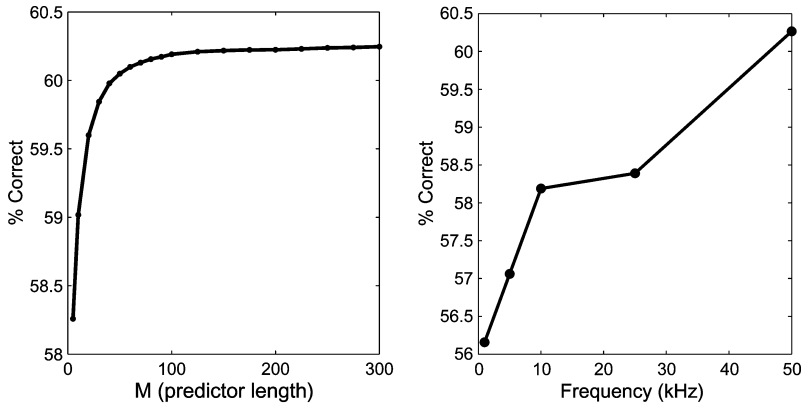

Fig. 17. A single sequence, taken with $\mathrm{Fclk} = 50$ kHz, is evaluated with different predictor lengths. As a longer history is used for the prediction, the predictability increases and eventually saturates. (b) As the clock frequency increases, bit predictability increases. Predictor length $\mathrm{M} = 500$ for all sequences.

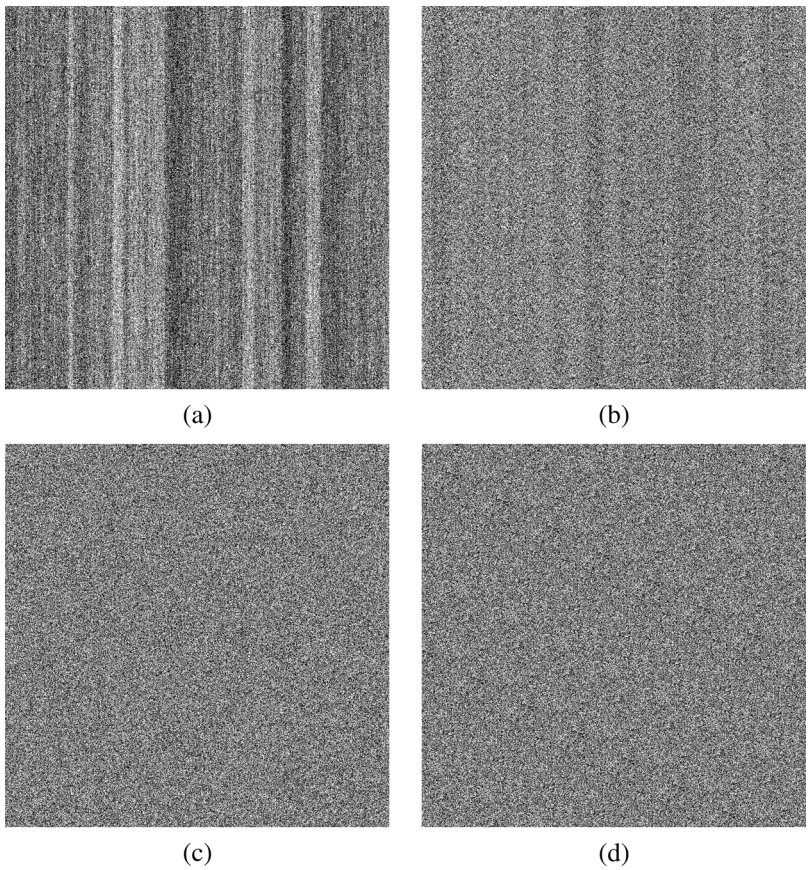

Fig. 18. Bits are arranged in a 1000 × 1000 array such that each column contains 1000 consecutive bits. (a) Unprocessed bits from the DC-nulling RNG, taken at 10 kHz. (b) Unprocessed bits taken from the FIR-based RNG at 10 kHz, with 25 taps enabled. (c) The same bits as in (a), after being post-processed. (d) Bits from a 16-tap LFSR, with a period of 65 535.

in the prediction. Fig. 17(b) shows prediction accuracy for a set of sequences generated by the DC-nulling RNG at several different frequencies, evaluated with $M = 500$. As expected, predictability increases with increasing sampling rate, due to the influence of low-frequency noise. A sequence with 60% predictability still has about .97 bits of entropy per unprocessed bit, as was discussed in IV-A, which means that the sequences analyzed for Fig. 17(b) still have more than enough entropy to yield perfectly random bits with a down-sampling rate of two.

Segments of four sequences are shown in Fig. 18. The sequences are one million bits long, arranged in arrays going first top to bottom, then left to right. The vertical bands in Fig. 18(a) indicate low-frequency noise in the unprocessed output of the DC-nulling RNG. The bands in Fig. 18(b) are much fainter, demonstrating the suppression of low-frequency noise by the FIR filter. The processed bits in Fig. 18(c) show no perceptible patterns, because of the randomizing effect of the post-processor. A sequence from a 16-bit LFSR is also shown for com-

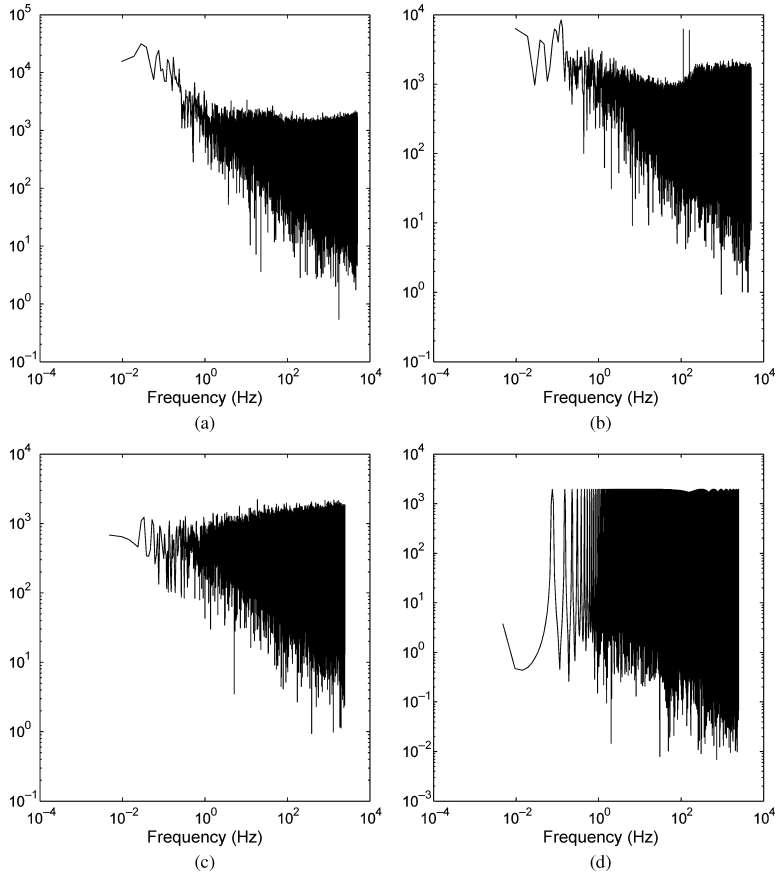

Fig. 19. Fourier transforms of the same bit sequences shown in Fig. 18. The sampling frequency of the LFSR sequence in (d) is set to 5 kHz for consistency with the processed sequence in (c).

parison. The LFSR sequence repeats every 65535 bits, although the pattern is difficult to discern visually. Fourier transforms of the same sequences are shown in Fig. 19. The tones visible in Fig. 19(b) correspond to the peaks in the autocorrelation function visible in Fig. 15.

We tested the processed bits using the NIST test suite [30] and the proportion of sequences passing each test is shown in Table I. Every set of sequences passed every test at a rate exceeding the minimum pass rate, shown in the far right column. The bit rate listed in the table refers to bits produced per second after post-processing, which is half of the original sampling rate, since the post-processor reduces the bit rate by a factor of two. All tests were performed on ten one-million bit sequences. We have included the NIST test results for a pseudo-random number generator for comparison. The bits were acquired in Matlab by taking the sign of a normally distributed random vector, generated using Marsaglia's Subtract-with-Borrow and Ziggurat algorithms [31], [32]. Bits from the PRNG were not post-processed.

Table II compares our TRNGs to other integrated TRNGs. The DC-nulling RNG demonstrates the lowest power consumption of any fully-integrated TRNG reported to date. It is also among the most efficient of TRNGs reporting NIST test suite results.

VI. Conclusion

We have shown two true random number generators with statistically validated randomness based on a novel bias compensation scheme using analog floating-gate memory cells. The low power consumption makes our RNGs attractive for embedded applications with severe power constraints and modest bit rate requirements. We have also presented a simple post-processing scheme suitable for compact implementation in CMOS logic. Statistical tests validated the randomness of the post-processed bits, even when the unprocessed bits showed significant bias. Additionally, the circuit techniques we have proposed could be used for other applications, such as background calibration of comparator offset.

The FIR-based RNG incurred a significant area and power penalty relative to the DC-nulling RNG. The FIR filter improved the randomness of unprocessed bits significantly, and enabled higher frequency operation. However, for energy-constrained applications, the DC-nulling RNG presented the most efficient operation.

The results we have presented suggest that post-processing can improve the quality of random sequences more effectively and efficiently than analog circuit techniques. The relationship between entropy and sequence predictability shows that a sequence need not be of very high quality to contain a useful quantity of randomness. We have introduced low-power techniques to maintain the necessary amount of randomness in a sequence.

Appendix

In this appendix, we examine the effect on the entropy of post-processor state of alternate clocking and state perturbation by the unprocessed bits. We will show that, given knowledge of

TABLE I
RESULTS OF NIST TESTS

RNG Type	FIR	DC	DC	DC	Matlab	
Bit Rate [1]	50k	0.5k	2.5k	5k	—	Min.
Adaptation	Off	On	On	On	—	Pass
Taps Enabled	25	—	—	—	—	Rate
Unprocessed Bits						
Mean	0.0172	0.0283	0.0204	0.0140	-5.5e-3	—
Max. Corr.	-0.097	-0.077	0.0475	0.0579	7.3e-4	—
Predictability	0.5844	0.5616	0.5706	0.5819	.5029	—
Processed Bits						
Test Name	Pass Rates					
Frequency	1.0000	1.0000	1.0000	1.0000	0.9000	0.896
Block-Frequency	1.0000	1.0000	1.0000	1.0000	1.0000	0.896
Cusum	1.0000	1.0000	1.0000	1.0000	0.9000	0.923
Runs	1.0000	0.9000	0.9000	0.9000	1.0000	0.896
Long-Run	1.0000	0.9000	1.0000	1.0000	1.0000	0.896
Rank	1.0000	1.0000	1.0000	1.0000	1.0000	0.896
FFT	1.0000	1.0000	1.0000	1.0000	1.0000	0.896
Aperiodic-Template	0.9892	0.9878	0.9878	0.9939	0.9905	0.982
Periodic-Template	1.0000	1.0000	1.0000	1.0000	0.9000	0.896
Universal	1.0000	0.9000	0.9000	1.0000	1.0000	0.896
Apen	1.0000	1.0000	1.0000	1.0000	0.9000	0.896
Random-Excursion	1.0000	1.0000	1.0000	1.0000	0.9844	0.929
Random-Excursion-V	1.0000	1.0000	1.0000	0.9778	1.0000	0.949
Serial	1.0000	1.0000	1.0000	1.0000	1.0000	0.923
Lempel-Ziv	1.0000	1.0000	1.0000	0.9000	1.0000	0.896
Linear-Complexity	1.0000	1.0000	1.0000	1.0000	1.0000	0.896
$I_{DD,RNG}$ (μA)	35.87	0.58	1.16	1.86	—	—
$I_{DD,Post}$ (nA)	197	1.97	9.85	19.7	—	—
Power(μW)	180.3	2.92	5.87	9.39	—	—
nJ/bit	3.61	5.85	2.35	1.88	—	—

[1] Output rate of processed bits, or half the clock frequency of the latch.

TABLE II
COMPARISON OF TRNGS

RNG	Bit Rate	Power	Area (mm^2)	Process	Passed all NIST Tests	Joules/bit
Petrie [7]	1.4Mb/s	3.9mW	1.5	2μm	Yes	2.78n
Bucci [15]	10Mb/s	2.3mW	.0016	.19μm	(FIPS) [1]	.23n
Brederlow [13]	200kb/s	50μW	.009	.12μm	No	.25n
Pareschi [17]	40Mb/s	29mW	.518	.35μm	Yes	.73n
Tokunaga [14]	200kb/s	1mW	.036	.13μm	Yes	5n
This work (DC)	500 b/s	2.92μW	.031	.35μm	Yes	5.85n
This work (DC)	5kb/s	9.39μW	.031	.35μm	Yes	1.88n
This work (FIR)	50kb/s	180.μW	1.49	.35μm	Yes	3.61n

[1] Presented results from FIPS test suite, but not NIST test suite.

the state at some time t_0 and of the subsequent processed bit sequence, the conditional entropy of the post-processor's state increases as subsequent bits are generated. That is, an attacker's knowledge of the state decreases.

Assume the generator depicted in Fig. 10 is constructed of two modified NFSRs R_1 and R_2, with lengths L_1 and L_2, and states S^1 and S^2, with combined state U_t at time t. The bit in the k^{th} position of R_1 at time t is $s^1_{k,t}$. Also assume that the unprocessed bits can be predicted with probability p in the range [0.5, 1], so the entropy h_1 of a single unprocessed bit is defined by (6). At any time $t > t_0$, an observer may have compiled a distribution of N possible states and their associated probabilities. We will refer to the i^{th} possibility for the state at time t as $U_{t,i}$, so the entropy in the post-processor state is

$$H_{State} = \sum_{i=1}^{N} \Pr(U_t = U_{t,i}) \log \Pr(U_t = U_{t,i}). \quad (8)$$

If we assume that the two possible output bits $s^1_{L_1,t}$, $s^2_{L_2,t}$ are distributed independently and uniformly in the set of possible states, then we may evenly divide the possible states into four classes, based on the observed bit b_t. Class C_1 will comprise, on average, one fourth of the states where $s^1_{L_1,t} = s^2_{L_2,t} \neq b_t$. They can be eliminated, because they could not have produced the observed bit. For classes C_2 and C_3, $s^1_{L_1,t} \neq s^2_{L_2,t}$, so states of these classes could have yielded either output bit, but we can infer from the output bit which NFSR was clocked, and thus one of the two input bits. States of class C_2 would have generated the observed bit b_t with probability p, states of class C_3 with probability $(1-p)$. If the current state $U_t \in C_2 \cup C_3$, there are two possibilities for next state U_{t+1}, since it is known which NFSR was clocked, but it could have been perturbed with a 1 or a 0. The remaining fourth of the states are assigned to class C_4, and $s^1_{L_1,t} = s^2_{L_2,t} = b_t$, so no knowledge of the unprocessed bits is gained. For any state $U_t \in C_4$ there are four possibilities for U_{t+1}. Since there are two possibilities for the clock control

bit that can yield the observation b_t from states in C_4, while only one possibility for states in C_2 or C_3, the conditional probability of observing bit b_t given a hypothetical state $U_{t,i}$ is

$$\Pr(b_t|U_t = U_{t,i}) = \begin{cases} 0 & \text{if } U_{t,i} \in C_1; \\ p & \text{if } U_{t,i} \in C_2; \\ 1-p & \text{if } U_{t,i} \in C_3; \\ 1 & \text{if } U_{t,i} \in C_4. \end{cases} \quad (9)$$

Applying Bayes' rule [33] to the posterior state probability gives

$$\Pr(U_t = U_{t,i}|b_t) = \frac{\Pr(b_t|U_t = U_{t,i})\Pr(U_t = U_{t,i})}{\Pr(b_t)}. \quad (10)$$

Experimental results support an assumption that the bits taken from the output of the NFSR are equally distributed, so $\Pr(b_t) = 0.5$. By substituting (9) into (10) and including knowledge of the initial state and the bit sequence up to time t, the posterior probability that a hypothetical state $U_{t,i}$ is the actual state U_t becomes

$$\Pr(U_t = U_{t,i}|U_0, b_{0:t}) = \begin{cases} 0 & \text{if } U_{t,i} \in C_1; \\ 2p \cdot \Pr(U_t = U_{t,i}|U_0, b_{0:t-1}) & \text{if } U_{t,i} \in C_2; \\ 2(1-p)\Pr(U_t = U_{t,i}|U_0, b_{0:t-1}) & \text{if } U_{t,i} \in C_3; \\ 2\Pr(U_t = U_{t,i}|U_0, b_{0:t-1}) & \text{if } U_{t,i} \in C_4. \end{cases} \quad (11)$$

Assuming for the moment that the prior probabilities were equal for all four classes, and that the input bits are perfectly unbiased ($\mathrm{p} = 0.5$), the posterior probabilities reduce to 0, 0.25, 0.25 and 0.5 times the prior probability for classes 1 through 4, respectively. Then the average number of possibilities for state U_{t+1} generated from each possible state $U_{t,i}$ becomes $0.25 \cdot 2 + 0.25 \cdot 2 + 0.25 \cdot 4 = 2$. States of class C_4 have twice the posterior probability as states of classes $C_{2,3}$, but generate twice as many possible subsequent states. Therefore, if the input bits are unbiased, and the prior probabilities of the states are uniform, then the posterior probabilities of the new set of states are also uniform. At each time step, the number of possible states doubles and the entropy increases by one bit. Eventually, the entropy begins to saturate as multiple paths through the state space converge to the same state, since the total number of possible states can never exceed $2^{L_1+L_2}$.

In the case where the input bits are biased, the candidate states are not equally probable. However, simulations and FSR theory [4] support the assumption that the input sequence and the outputs of the two NFSRs are all mutually uncorrelated, and that therefore the candidate states are equally distributed among the four classes. Under this assumption, the number of possible states will still double at every time step. The set of candidate states can be divided into subsets of 4 equally probable states, with one state of each class. Then the average increase in entropy per time step can be derived as if there were exactly four equally probable candidate states.

Suppose that there are four possible states x_1, x_2, x_3, x_4, each with a probability of 0.25. After a bit is observed, states x_{1-4} are assigned to classes C_{1-4} respectively. State x_1 can then be eliminated. States x_2, x_3 and x_4 will yield states y_{2a} or y_{2b}; y_{3a} or y_{3b}; and y_{4a}, y_{4b}, y_{4c}, or y_{4d}. Based on the analysis above, the state probabilities will be $P_{State} = \{0.25p, 0.25(1-p), 0.25p, 0.25(1-p), .5p^2, .5p(1-p), .5(1-p)p, .5(1-p)^2\}$, respectively. We substitute the list of state probabilities into (8), and after some manipulation we find

$$H_{State} = -\frac{3}{2}(p\log p + (1-p)\log(1-p) - 1) \quad (12)$$

$$= \frac{3}{2}(h_1 + 1). \quad (13)$$

Before generation of the observed bit, we had four equally probable states, corresponding to a state entropy of 2 bits. Therefore, the increase in state entropy is

$$\Delta H = \frac{3}{2}h_1 - 0.5. \quad (14)$$

The entropy of the post-processor state will increase as long as $H_0 \geq 1/3$, which corresponds to a bit sequence that can be predicted with 93% accuracy. We therefore conclude that in the absence of gross malfunction, the entropy of the post-processor state will increase.

REFERENCES

[1] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "Energy-efficient communication protocols for wireless microsensor networks," in *Proc. Hawaiian Int. Conf. Syst. Sci.*, Maui, HI, Jan. 2000, vol. 2, pp. 1–10.

[2] B. Glover and H. Bhatt, *RFID Essentials*. Munich: O'Reilly Vlg. GmbH & Co., 2006.

[3] B. Schneier, *Applied Cryptography*. New York: Wiley, 1996.

[4] A. Menezes, P. van Oorschot, and S. Vanstone, *Handbook of Applied Cryptography*. Boca Raton, FL: CRC Press, 1997.

[5] J. Holleman, B. Otis, S. Bridges, A. Mitros, and C. Diorio, "A 2.92-μW hardware random number generator," in *Proc. 32nd Eur. Solid-State Circuits Conf. (ESSCIRC 2006)*, Montreux, Sep. 2006, pp. 134–137.

[6] M. Figueroa, S. Bridges, and C. Diorio, , L. K. Saul, Y. Weiss, and L. Bottou, Eds., "On-chip compensation of device-mismatch effects in analog VLSI neural networks," in *Advances in Neural Information Processing Systems 17*. Cambridge, MA: MIT Press, 2005, pp. 441–448.

[7] C. S. Petrie and J. A. Connelly, "A noise-based IC random number generator for applications in cryptography," *IEEE Trans. Circuits Syst. I*, vol. 47, no. 5, pp. 615–621, May 2000.

[8] K. Tsoi, K. Leung, and P. Leong, "Compact fpga-based true and pseudo random number generators," in *Proc. 11th Annu. IEEE Symp. Field-Programmable Custom Comput. Machines (FCCM 2003)*, Napa, CA, 2003, pp. 51–61.

[9] W. T. Holman, J. A. Connelly, and A. B. Dowlatabadi, "An integrated analog/digital random noise source," *IEEE Trans. Circuits Syst. I*, vol. 44, no. 6, pp. 521–528, Jun. 1997.

[10] *"Data Security Components Manual"* T. S. Corp., Aug. 2004 [Online]. Available: http://www.tundra.com/Products/8000Series/index.cfm

[11] M. Bellido, A. Acosta, M. Valencia, A. Barriga, and J. Huertas, "A simple binary random number generator: New approaches for CMOS VLSI," in *Proc. 35th Midwest Symp. Circuits Syst.*, Aug. 1992, pp. 127–129.

[12] D. Kinniment and E. Chester, "Design of an on-chip random number generator using metastability," in *Proc. 28th Eur. Solid-State Circuits Conf. (ESSCIRC 2002)*, Florence, Italy, 2002, pp. 595–598.

[13] R. Brederlow, R. Prakash, C. Paulus, and R. Thewes, "A low-power true random number generator using random telegraph noise of single oxide-traps," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, 2006, pp. 422–423.

[14] C. Tokunaga, D. Blaauw, and T. Mudge, "True random number generator with a metastability-based quality control," in *IEEE Int. Solid-State Circuits Conf. Dig. Tech. Papers*, San Francisco, CA, Feb. 2007, pp. 404–405, 611.

[15] M. Bucci, L. Germani, R. Luzzi, A. Trifiletti, and M. Varanonuovo, "A high-speed oscillator-based truly random number source for cryptographic applications on a smart card IC," *IEEE Trans. Comput.*, vol. 52, no. 4, pp. 403–409, Apr. 2003.

[16] B. Jun and P. Kocher, "The Intel random number generator," Intel Corp., Tech. Rep., 1999 [Online]. Available: http://www.cryptography.com/resources/whitepapers/IntelRNG.pdf

[17] F. Pareschi, G. Setti, and R. Rovatti, "A fast chaos-based true random number generator for cryptographic applications," in *Proc. 32nd Eur. Solid-State Circuits Conf. (ESSCIRC 2006)*, Montreux, Sep. 2006, pp. 130–133.
[18] C.-C. Wang, Y.-L. Tseng, H.-C. Cheng, and R. Hu, "Switched-current 3-bit CMOS wideband random signal generator," in *Proc. 2003 Southwest Symp. Mixed-Signal Design*, Las Vegas, NV, Feb. 2003, pp. 186–189.
[19] T. Zhou, M. Yu, and Y. Ye, "A pipelined switched-current chaotic system for the high-speed truly random number generation in crypto processor," in *Proc. 19th Int. Conf. VLSI Design*, Hyderabad, India, 2006, pp. 216–221.
[20] M. Lenzlinger and E. H. Snow, "Fowler-Nordheim tunneling into thermally grown SiO2," *J. Appl. Phys.*, vol. 40, no. 1, pp. 278–283, Jan. 1969.
[21] J. J. Sanchez and T. A. DeMassa, "Review of carrier injection in the silicon/silicon-dioxide system," *IEE Proc. G–Circuits, Devices Systems,* , vol. 138, no. 3, pp. 377–389, Jun. 1991.
[22] M. Figueroa, S. Bridges, D. Hsu, and C. Diorio, "A 19.2 GOPS mixed-signal filter with floating-gate adaptation," *IEEE J. Solid-State Circuits*, vol. 39, no. 7, pp. 1196–1201, Jul. 2004.
[23] T. M. Cover and J. A. Thomas*, Elements of Information Theory*. New York: Wiley, 1991.
[24] J. von Neumann*, Collected Works*. New York: Pergamon, 1961, vol. V, ch. 23, pp. 768–770.
[25] A. J. Acosta, M. J. Bellido, M. Valencia, A. Barriga, and J. L. Huertas, "Fully digital redundant random number generator in CMOS technology," in *Proc. 19th Eur. Solid-State Circuits Conf. (ESSCIRC 1993)*, 1993, pp. 198–201.
[26] S. Poli, S. Callegari, R. Rovatti, and G. Setti, "Post-processing of data generated by a chaotic pipelined ADC for the robust generation of perfectly random bitstreams," in *Proc. 2004 Int. Symp. Circuits and Systems (ISCAS'04)*, May 2004, vol. 4, pp. IV-585–588.
[27] J. D. Golić and R. Menicocci, "Edit distance correlation attack on the alternating step generator," in *Advances in Cryptology—Crypto '97 (Lecture Notes in Computer Science Vol. 1294)*. Berlin, Germany: Springer-Verlag, Jan. 1997, pp. 499–512.
[28] T. Johansson, "Reduced complexity correlation attacks on two clock-controlled generators," in *Advances in Cryptology—ASIACRYPT '98 (Lecture Notes in Computer Science Vol. 1514)*. Berlin, Germany: Springer-Verlag, Jan. 1998, pp. 342–357.
[29] M. Živković, "An algorithm for the initial state reconstruction of the clock-controlled shift register," *IEEE Trans. Inf. Theory*, vol. 37, no. 5, pp. 1488–1490, Sep. 1991.
[30] A. Ruhkin *et al.*, "A statistical test suite for random and pseudorandom number generators for cryptographic applications," NIST, Tech. Rep. 800-22, 2001.
[31] G. Marsaglia and A. Zaman, "A new class of random number generators," *Ann. Appl. Probab.*, vol. 1, no. 3, pp. 462–480, Aug. 1991.
[32] G. Marsaglia and W. Tsang, "The Ziggurat method for generating random variables," *J. Statist. Softw.*, vol. 5, no. 8, pp. 1–7, 2000.
[33] A. Leon-Garcia*, Probability and Random Processes For Electrical Engineering*. Reading, MA: Addison-Wesley, 1994.

Jeremy Holleman (S'03) received the B.S. degree in electrical engineering from the Georgia Institute of Technology, Atlanta, in 1997. In 2006, he received the M.S. degree in electrical engineering from the University of Washington, Seattle, where he is currently pursuing the Ph.D. degree.

From 1999 to 2001 he worked as an Embedded Systems Engineer at Data I/O. His research interests include low-power analog signal processing, biologically inspired computation, and circuits for biomedical applications.

Seth Bridges (S'98–M'07) received the B.S. degree in computer engineering from the University of Pittsburgh, Pittsburgh, PA, in 2000, and the M.S. and Ph.D. degrees in computer science from the University of Washington, Seattle, in 2002 and 2006, respectively. While at the University of Washington, he researched low-power analog VLSI systems for computer vision and pattern recognition.

He is presently the Principal Engineer of V2Green, Inc, a company he co-founded in 2006, and he has worked as an embedded systems engineer at Impinj, Inc. His current research focus includes the management of distributed, energy resources for grid-support applications.

While at the University of Washington, Dr. Bridges received both the Intel Master's and Ph.D. Fellowships in 2000 and 2005, as well as the ARCS Fellowship in 2000.

Brian P. Otis (S'96–M'05) received the B.S. degree in electrical engineering from the University of Washington, Seattle, in 1999, and the M.S. and Ph.D. degrees from the University of California at Berkeley in 2002 and 2005, respectively.

He joined the faculty of the University of Washington as Assistant Professor of electrical engineering in August 2005, where he directs the Wireless Sensing Laboratory. His research interests include ultra-low-power analog, digital, and RF circuits for enabling ubiquitous sensing and communication. He is the coauthor of five book chapters and one book on ultra-low-power RF transceiver design. He has previously held positions at Intel Corporation and Agilent Technologies.

Dr. Otis is an Associate Editor of the IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS PART II. He was the recipient of the 2003 U.C. Berkeley Seven Rosen Funds award for innovation and was co-recipient of the 2002 ISSCC Jack Raper Award for Outstanding Technology Directions Paper.

Chris Diorio (M'88) received the Ph.D. degree from the California Institute of Technology (Caltech), Pasadena, in 1997, and

He is an Associate Professor of computer science and engineering at the University of Washington, Seattle, and is also Chairman and CTO of Impinj, Inc. He has over 20 years of experience in semiconductor and radio engineering. He has published more than 70 journal and conference articles and has 40 issued patents.

Dr. Diorio has received numerous awards including the 2007 WSA Technology Innovator of the Year award, the UW's 2001 Distinguished Teaching Award, and the 1996 IEEE EDS Paul Rappaport Award. He has also received Packard, Sloan, Presidential, and ONR Young Investigator fellowships. He has worked as a Senior Staff Engineer at TRW, Inc., as a Senior Staff Scientist at American Systems Corporation, and as a Technical Consultant at The Analytic Sciences Corporation.