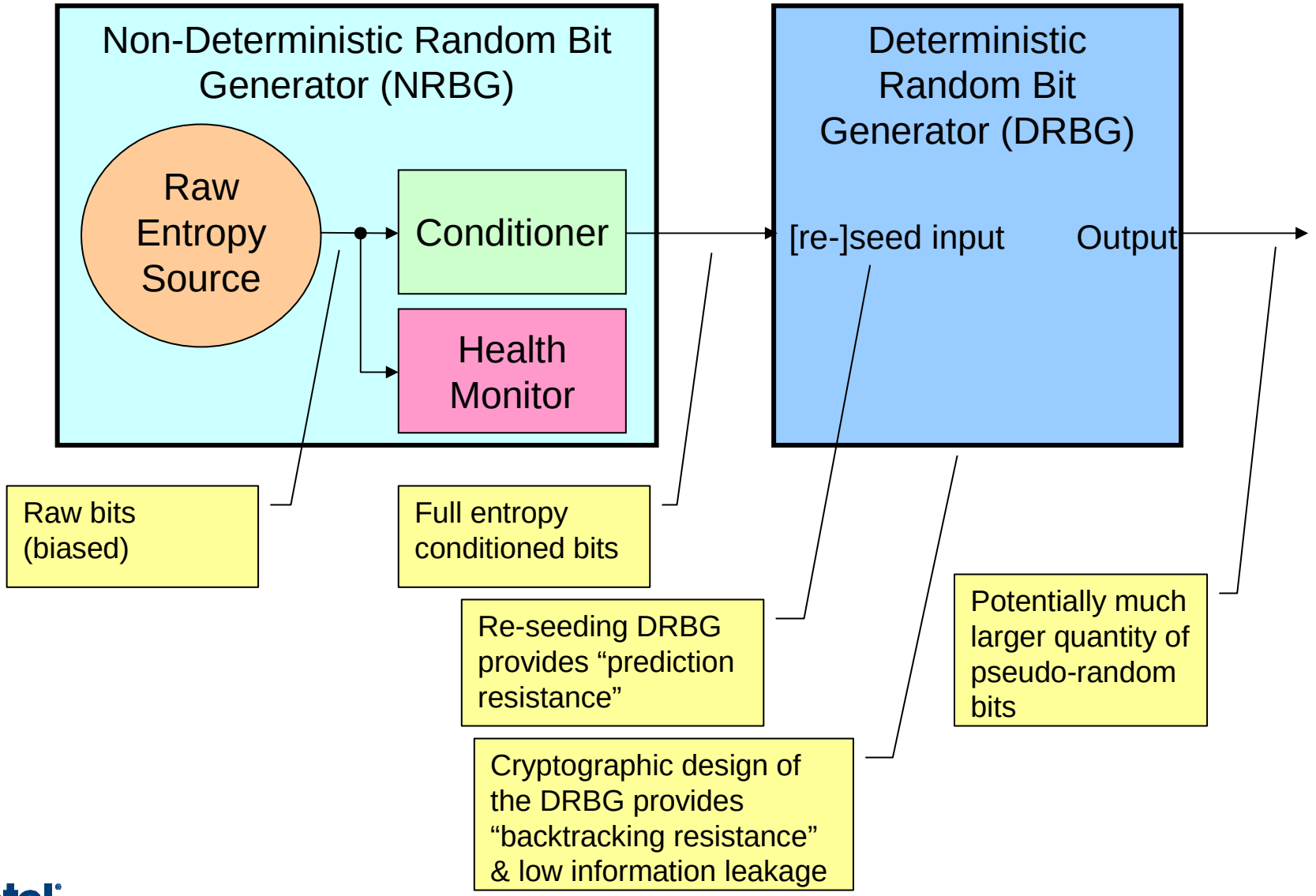
Conditioner and Health Monitor
for Use in Conjunction with a Non-Deterministic Random Bit Generator

G. Richard Newell
Senior Principal Product Architect
Actel Corporation

for CryptArchi 2009

Hardware testing performed by
James Vorgert
Actel Area Technical Manager

Generic Random Bit Generator Architecture

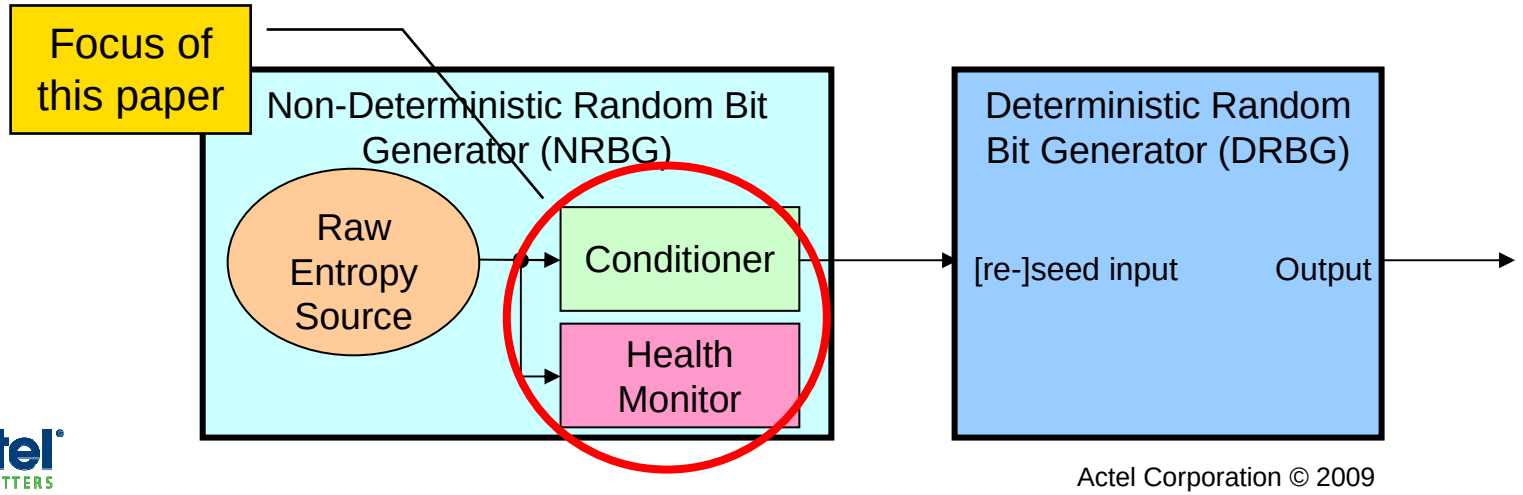


NRBG blocks

- Raw Entropy Source
  - Uses random physical process to generate raw bits
  - Raw bits likely have biases and correlations
    - Plus, some susceptibility to environment such as supply voltages & temperature
    - Due to finite gains and bandwidths, mis-matching of components, materials, etc.
  - Thus, raw output bits contains less than 1 bit of entropy each
  - But, more than some design minimum
- Health Monitor
  - Ensures raw entropy source is performing to specification
    - Entropy generation hasn't dropped too close to zero
  - Environmental monitors (if available) ensure valid operational conditions
- Conditioner
  - Compresses entropy of raw bits so output bits have full entropy
  - Removes any biases and correlations (referred to as "deskewing")
  - Output bits should pass statistical tests for randomness
    - e.g., NIST Statistical Test Suite (SP800-22)

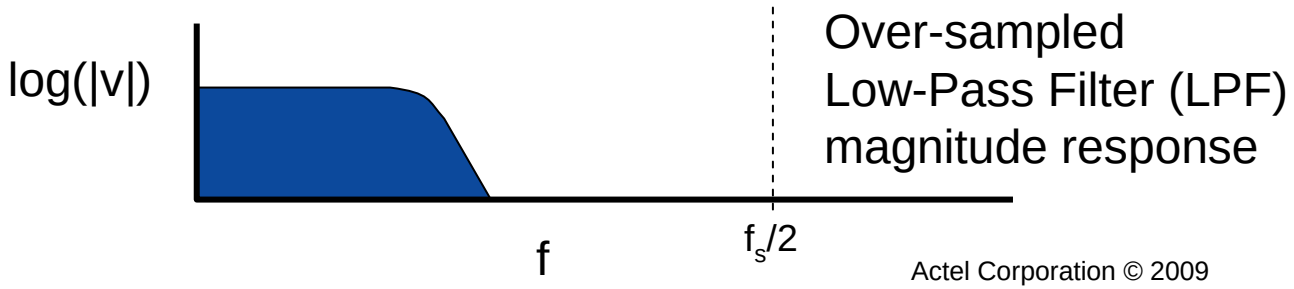
FPGA-friendly Health Monitors and Conditioners

- This paper presents FPGA-friendly Health Monitors and Conditioners
  - Partly survey
  - Plus some observations and a few new practical proposals
- It does not address:
  - The physical raw bit source
    - Many other Cryptarchi papers on this subject
  - The cryptographically strong Deterministic RBG which follows the conditioner
    - See NIST SP800-90 for approved DRBG algorithms



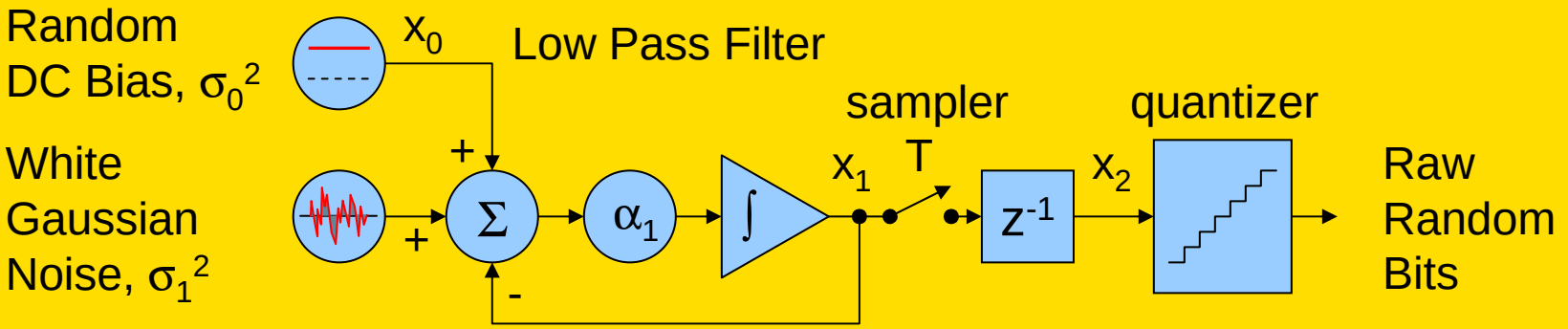
Common Defects in Raw Bits

- DC Biases
  - The probability of a bit being “1” is above or below 50%
  - For example, due to electrical offsets, the raw bit generator returns 49% ones and 51% zeroes for large sample sizes
- Time Correlations
  - The expected value of a bit depends upon the prior state of the system
    - The current output bit is [anti-]correlated with past outputs
    - Possibly characterized as a first- or higher-order Markov chain
  - For example:
    - Due to bandwidth limitations (high frequencies under-represented), the probability of a bit matching the preceding bit is greater than 50%
      - Could be caused by over-sampling relative to noise bandwidth
      - For once, aliasing is good!



Model of RBG Raw Entropy Source

RBG Model with Bias and 1st-order Correlation defects



- Markov model approximation:

state vector propagation

$$x_{n+1} = A \cdot x_n + B \cdot u_n$$

covariance matrix propagation

$$P_{n+1} = A \cdot P_n \cdot A^T + Q_n$$

$$A = \begin{bmatrix} 1 & 0 & 0 \\ \alpha_1 \cdot T & 1-\alpha_1 \cdot T & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

Transition Matrix

$$B = \begin{bmatrix} 1 & 0 \\ 0 & \alpha_1 \cdot T \\ 0 & 0 \end{bmatrix}$$

Input Mapping Matrix

- With stationary input noise, the variance and the covariance of adjacent samples are, respectively:

$$\lim_{n \to \infty}: \quad \mathrm{Var}(x_1) = \mathrm{Var}(x_2) = r_{xx}[0] = \sigma_1^2 \cdot \frac{1}{2 - \alpha_1 \cdot T} + \sigma_0^2$$

$$Q_n = \begin{bmatrix} \sigma_0^2 & 0 & 0 \\ 0 & \alpha_1 \cdot T \cdot \sigma_1^2 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

for n=0 only, otherwise 0

Covariance of input vector

$$\mathrm{Cov}(x_1,x_2) = r_{xx}[1] = \sigma_1^2 \cdot \frac{1 - \alpha_1 \cdot T}{2 - \alpha_1 \cdot T} + \sigma_0^2$$

where $r_{xx}[k]$ is the autocorrelation for shift = k

Ideal FPGA RBG Health Monitor Characteristics

- Low gate-count and SRAM requirements
- Operates on the same clock as the raw-bit sampler
  - Parallelizable; low algorithmic complexity; high speed
- Detects low entropy without fail (low false negatives)
  - Both bias and correlation type defects should be reliably detected
- Not too many false alarms (low false positives)
  - However, every sequence of random bits has a finite probability
    - Including those that look like generator failures
  - As such, some false alarms may be inevitable
    - More frequent if small test-block sizes are used, and if tighter limits are chosen to minimize false negatives
      - Beware that discarding too many failed blocks may also bias the results
- Availability of environmental monitors
  - Mixed-signal FPGAs (e.g., Actel's Fusion series) have on-chip voltage and temperature monitors and an on-chip R-C oscillator that can be used for environmental monitoring

RBG Health Monitor Options (slide 1/2)

- Block Test
  - NIST FIPS 140-3 (paragraph 4.9.2) requires continuously comparing adjacent blocks of bits[1] and, if identical, rejecting the 2nd block
  - FIPS 140-3 also requires an “Entropy Source Test”
- Frequency Test (a.k.a.: Monobit Test)[2,3]
  - Count “ones” as a percentage of total bits in a block
  - Variations:
    - Sliding window vs. non-overlapping blocks (FIR filter)
    - Exponentially decaying response window (IIR filter)
  - Measures DC bias directly
- Serial Test (a.k.a. Two-bit Test)[3]
  - Checks the frequency of two-bit overlapping sequences
  - Tests for bias and some correlations

RBG Health Monitor Options (slide 2/2)

- Poker Test[2,3]
  - Checks the frequency of m-bit non-overlapping sequences
    - m is a parameter; for FIPS 140-1 m = 4
- Runs Test[2,3]
  - Checks the frequency of runs of ones or zeroes of various lengths
    - The length of the runs is a parameter; for FIPS 140-1, $1 \leq i \leq 6$, with runs greater than 6 counted same as 6; zeroes and ones counted separately
- Long Run Test[2]
  - Reports the longest run of ones or zeroes in the sample block
- Autocorrelation Test[3]
  - Compares bits stream to shifted version of itself
    - Number of shifts is a parameter
  - Detects bias and most correlations, if several shifts are considered
    - Can also detect sine-wave interference

Autocorrelation from Transfer Functions

- For Linear Time-Invariant (LTI) signals, the Wiener–Khinchin theorem relates the power spectrum S_{xx} to the autocorrelation r_{xx} using the Discrete-time Fourier transform (DTFT) pair,

$$S_{xx}(f) = \sum_{k=-\infty}^{\infty} r_{xx}[k] e^{-j2\pi kf}$$

$$r_{xx}[k] = T \int_{-\frac{1}{2T}}^{\frac{1}{2T}} S_{xx_T}(f) \cdot e^{j2\pi fkT}\, df$$

where the autocorrelation is defined in terms of the expectation:

$$r_{xx}[k] = E[\, x[n] x^*[n-k]]$$

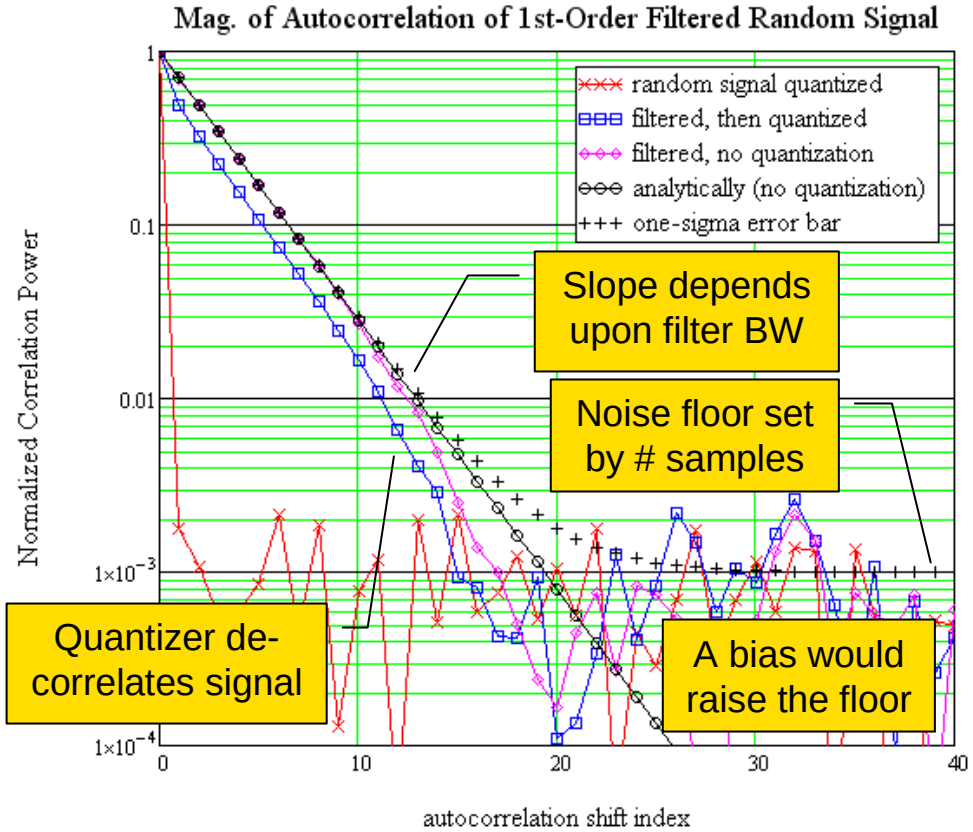
- Since in our model the input is white, its power spectrum $W_{xx}(f)$ is constant over frequency. The output spectrum can be calculated from the filter transfer function & its conjugate:

$$S_{xx}(f) = W_{xx} \cdot T(f) \cdot T^*(f)$$

- So, now we have another way to compute the autocorrelation from our stationary noise, using filter transfer functions

Autocorrelation Analysis Results

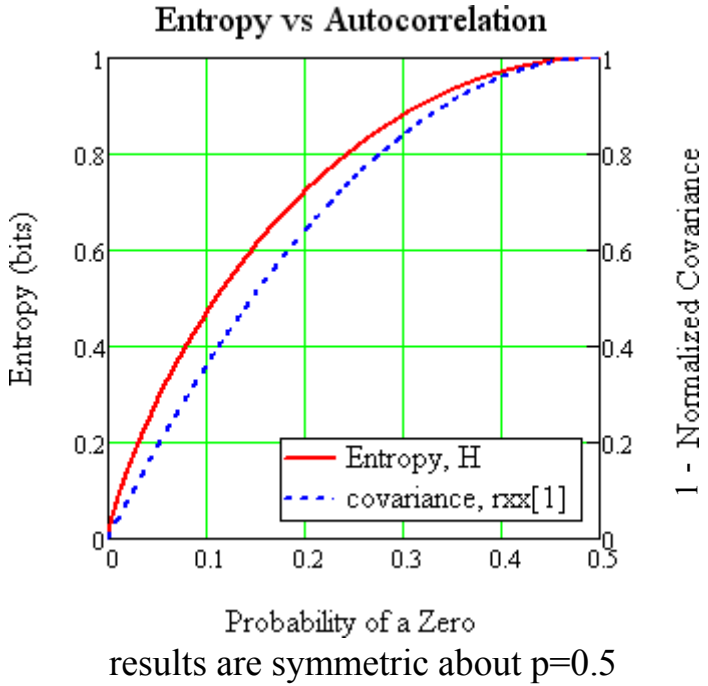
- Autocorrelation provides a useful measure of data flaws
  - An upper-bound on entropy can be estimated by comparing the power represented in $r_{xx}[0]$ vs. that in $|r_{xx}[k\neq0]|$
  - For a correlation defect:



• Single LPF pole at z=0.7
• Single-bit quantizer

• No bias added
• 1M samples

  - For a bias defect:

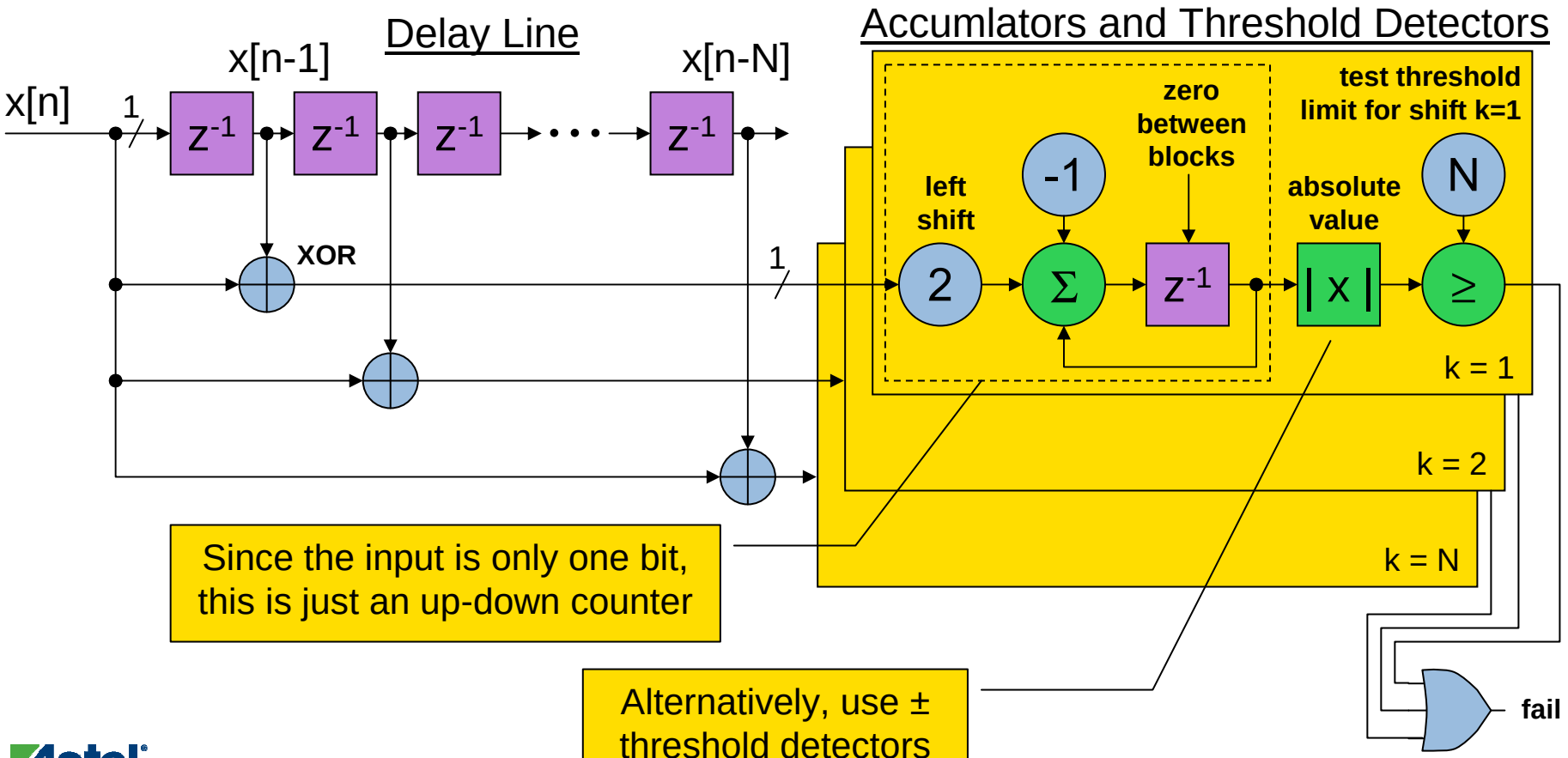


results are symmetric about p=0.5

  - The entropy is close to the value of 1 - $r_{xx}[k\neq0]/r_{xx}[0]$ for all values of bias and shift

FPGA Implementation of Autocorrelation Test

- The implementation is relatively efficient
  - Just need an accumulator for each shift value desired
    - Don't need too many; the set should include k=1 plus a few others
    - For most RBG bit rates, circuits (except storage) can be time-shared



Health Monitor Design Considerations

Test Name	Bias	1st-order	Other	Size	Comments
Block			x	med	Req'd. Detects total failure
Frequency	x			low	Special case of poker test
Serial	x	x		low	Similar to autocorrelation shift=1
Poker	x	x	x	med	Statistic uses multiplier^4
Runs			x	low	
Long Run			x	low	
Autocorrelation	x	x	x	med	Best overall entropy estimator

Add'l Considerations	Response	Comments
Data Window Function		
Non-overlapping Blocks	Slowest	Most tests already structured this way Statistical formulae available Easy to identify, quarantine, & discard bits
Sliding Window (rectangular)	Faster	Higher storage requirements
Sliding Window (exponentially decaying)	Fastest	Often more efficient to implement Bits never completely removed from results (infinite impulse response)
Block Size		
Small	Faster	Less memory, but more false alarms

Ideal FPGA RBG Conditioner Characteristics

- Low gate-count and SRAM requirements
- Operates on the same clock as the raw-bit sampler
  - Parallelizable; low algorithmic complexity; high speed
- Removes all DC biases; removes all correlations
  - Output bits statistically indistinguishable from random bits
    - Should pass the Statistical Test Suite in NIST SP800-22
- Entropy compression
  - Therefore, there are fewer output bits than input bits,
    - the max. ratio of outputs to inputs depending upon the entropy of the input bits
    - but, without throwing away too many bits (i.e. we want close to the max.)
    - A constant compression ratio is preferred for fixed-rate applications
  - Good mixing of the weak-entropy input bits with the internal conditioner state
  - Output bits should “contain” *very nearly* 100% entropy
    - According to Santha and Vazirani it is impossible to deterministically extract even one true random bit from a single weak random source[5] (per information theory)
      - However, in practice, we can get computationally acceptable results
- Cryptographically strong
  - Low information leakage; hard to determine internal conditioner state

Von Neumann Extractor - for Removing Bias

- Removes the bias from a stationary and otherwise uncorrelated bit stream (i.e., bits are independent), using the mapping:

$$00 \rightarrow \Lambda, \quad 01 \rightarrow 0, \quad 10 \rightarrow 1, \quad 11 \rightarrow \Lambda$$

where Λ indicates that no bit is output for that input pair

- The efficiency is defined as the number of output bits per input bit
  - The efficiency is fairly low: $\leq 1/4 \approx p_1 * p_0$ (where p_1 is the probability of a one and p_0 is the probability of a zero)
- It can be used as a test by measuring the actual ratio of output to input bits (i.e., measure the "efficiency")
  - It approaches the upper limit of 1/4 for input sequences with no bias
  - The frequency test seems simpler and more direct for measuring bias

Improved Efficiency Bias Removal

- Elias[6] generalized the Von Neumann Extractor for input n-tuples with $n \geq 2$, with improved efficiency for large n
  - As n approaches ∞, the efficiency approaches the entropy limit
  - for example, a 6-tuple version, with an efficiency of ≤11/24 vs. ≤ ¼ (= 6/24) for the Von Neumann 2-tuple version:

000000	Λ	001000	01	010000	10	011000	010	100000	11	101000	110	110000	111	111000	1111
000001	0	001001	00	010001	11	011001	0011	100001	011	101001	1001	110001	1100	111001	101
000010	1	001010	01	010010	000	011010	0100	100010	100	101010	1010	110010	1101	111010	110
000011	Λ	001011	01	010011	0000	011011	1	100011	0110	101011	11	110011	010	111011	01
000100	00	001100	10	010100	001	011100	0101	100100	101	101100	1011	110100	1110	111100	111
000101	0	001101	10	010101	0001	011101	00	100101	0111	101101	000	110101	011	111101	10
000110	1	001110	11	010110	0010	011110	01	100110	1000	101110	001	110110	100	111110	11
000111	00	001111	Λ	010111	0	011111	0	100111	10	101111	1	110111	00	111111	Λ

where Λ means no bits are output for that input 6-tuple

- Peres[7] showed that a similar efficiency can be obtained by iterating the Von Neumann Extractor on the bits it discards

Improving the Bias

- Exclusive-OR[8]
  - XOR'ing independent bits with an already small bias results in a bit with a substantially lower bias. (E ≡ Expectation):

If $E(X_1)=E(X_2)=E(X_n)=\mu$,

Then $E(X_1 \oplus X_2 \oplus \ldots X_n) = \frac{1}{2} + (-2)^{n-1} \cdot (\mu - \frac{1}{2})^n$

  - Good building block for compressors
  - Not that good at removing correlation
    - If significant correlation is present in the input bits due to a Markov process, then there may still remain a significant bias in output
  - Very inexpensive implementations

Removing Correlation

- Samuelson-Pratt[6] mapping
  - Removes the correlation from a first-order stationary, ergodic, two-state Markov process using the mapping:

$$00 \rightarrow \Lambda, \quad 01 \rightarrow \Lambda, \quad 10 \rightarrow 0, \quad 11 \rightarrow 1$$

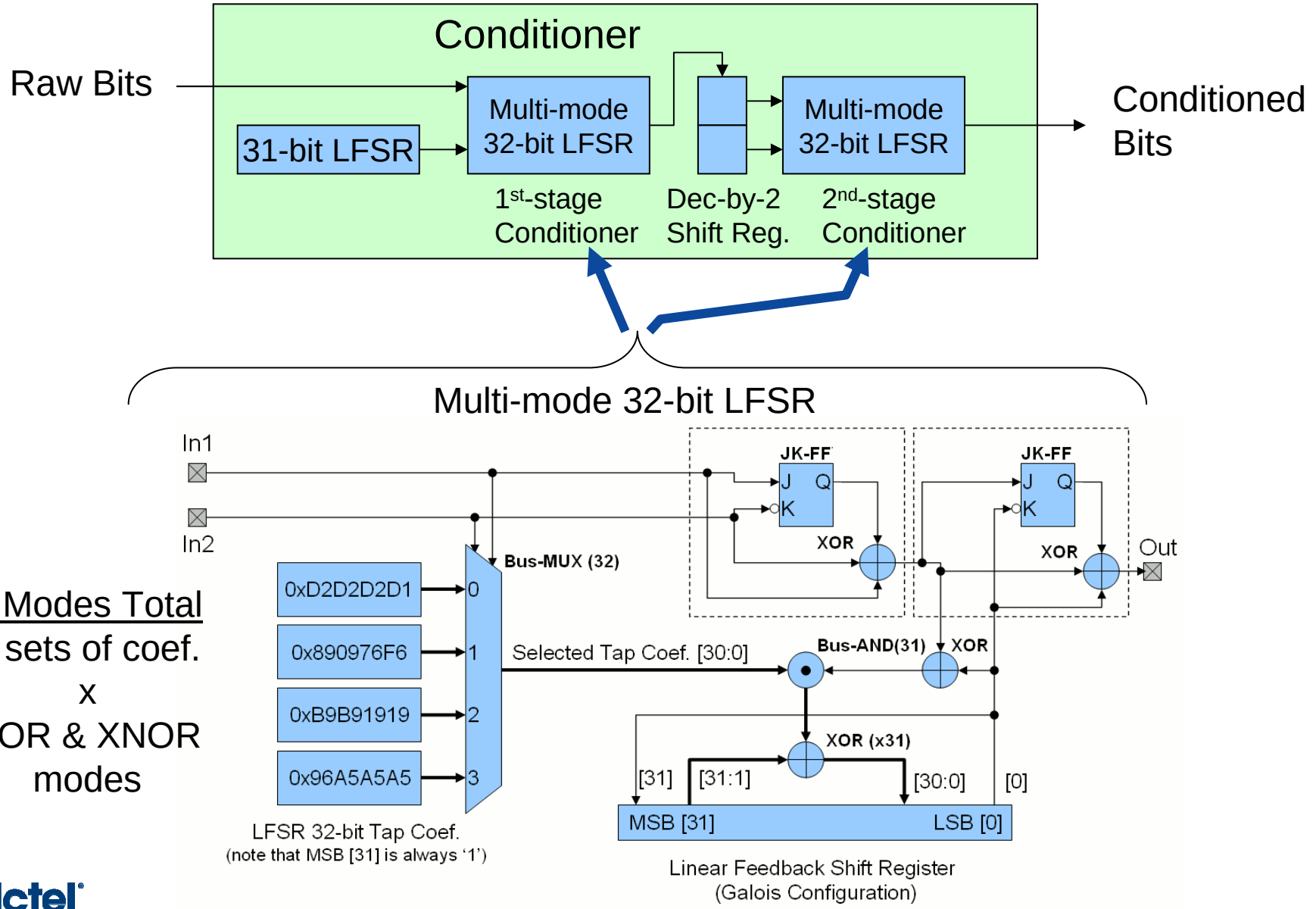
where Λ indicates that no bit is output for that input pair

  - One can remove both first-order correlations and biases by applying the Samuelson-Pratt mapping followed by the Von Neumann Extractor
    - But, the efficiency is low: $\leq 1/8$ output bit per input bit
  - Can be used as a test by measuring the ratio of output to input bits
- Elias[6] generalized the Samuelson-Pratt mapping for larger n-tuples and for higher-order Markov processes, with improved efficiency
- Blum[9] also considered higher-order Markov processes
- More recent advances in information theory have connected randomness extractors to psuedo-random bit generators and error-correcting codes[10,11]

Ad Hoc Conditioner Approaches

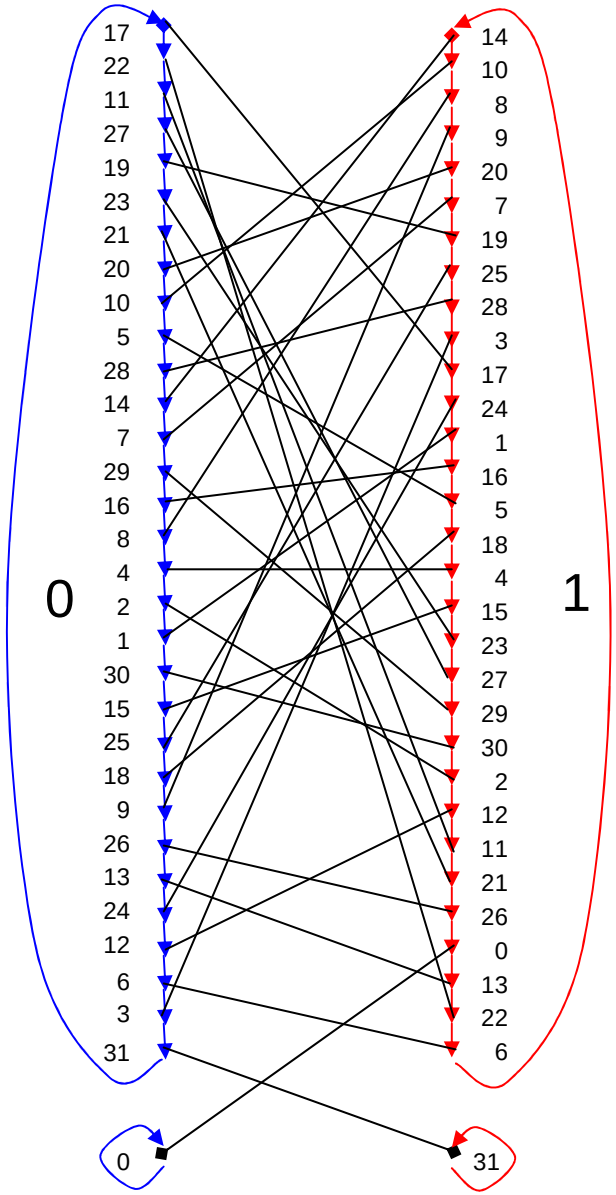
Type	Statistical Quality	Block/ Stream	Good Mixing	Compression	Crypto. Strength	FPGA Complexity	Comments
Exclusive-OR (XOR)	poor	stream	possible	yes	very low	very low	flexible
Linear Feedback Shift Register (LFSR)	fair	stream	yes	yes	low	low	Several may be chained
Cyclic Redundancy Check (CRC)	fair	block	yes	yes	low	low	
Message Digest	excellent	block	yes	yes	high	very high	
Deterministic Random Bit Generator	excellent	?	yes	yes	high	very high	For good quality DRBGs (e.g., Blum-Blum-Shub)
Block Cipher	excellent	block	yes	yes	high	very high	Can be configured as a hash
Stream Cipher	excellent	stream	no	no	good	fairly low	additive cipher

A Proposal for a New Ad Hoc Conditioner



8 Modes Total
4 sets of coef.
x
XOR & XNOR
modes

Simplified Example State Trajectory



- **Example shown is for a 5-bit LFSR (31 + 1 states), with two modes**
  - The Output is the LSB of the current state
  - Each LFSR state sequence is fairly random
  - Connections between them are well mixed
- **Now Imagine…**
  - For a 32-bit Multi-mode LFSR with 2 input bits and 8 LFSR modes
  - A full mesh of 8 such sequences
  - Each with over 4 billion states

Conditioner Results

- MATLAB simulations
  - Passes all NIST Statistical Test Suite tests[12] (post conditioner)
  - Even for a raw bit stream with fairly severe defects
    - 1st-order filter with pole at z=-0.8 and zero at z=-1 (3dB BW $\cong 0.93 \cdot f_s/2$)
    - DC Bias (pre quantizer) of $0.2 \cdot \sigma^2$, where σ^2 is the std. dev. of raw source
    - Combined with a Sine wave interferer with amplitude $\pm 0.3 \cdot \sigma^2$ and frequency of 10 sample periods
- Hardware tests – Actel Fusion FPGA
  - Harvested randomness from Fusion FPGA analog front end
    - Estimated input entropy greater than 0.9 bits per bit, 160M bit sample
  - Passes all NIST Statistical Test Suite tests[12] (post conditioner)
- The conditioner uses only about 300 tiles (approx. 120 LE's) and can operate at 90MHz (with 45Mbps at the output)

Type	Statistical Quality	Block/ Stream	Good Mixing	Compression	Crypto. Strength	FPGA Complexity	Comments
Multi-Mode LFSR	excellent	stream	yes	yes	good (?)	low	Best of: Stream Cipher, LFSR, & XOR

Conclusion

- Several candidates for a source entropy test were considered for FPGA implementation
  - Standard bit-level tests plus autocorrelation test
- An error model and several analysis methods were presented
  - Recursive filter model, with results for a 1st order Markov process
  - Stationary model approach using the Weiner-Khinchin theorem
    - With simulation results compared to analytical model
- A brief survey of classical randomness extractors
  - Von Neumann, Elias, and etc. (with footnotes)
- A summary of common ad hoc conditioners
  - LFSR, CRC, hash algorithms, stream ciphers, and etc.
  - Including an assessment of performance and cost
- A novel ad hoc conditioner was proposed
  - Simulation and actual FPGA hardware results presented

Footnotes

1. with a block size greater than 63 for FIPS 140-3; note that this was 15 in FIPS 140-1
2. NIST FIPS 140-1 specified that the monobit, poker, runs, and long run tests be performed on a 20,000-bit block of [conditioned] data upon demand for security levels 3 and 4, and at power-up (level 4). It provided test parameters and limits.
3. These five tests are described, along with the statistic used for each, in Handbook of Applied Cryptography, by A. Menezes, P. van Oorschot, and S. Vanstone, CRC Press, 1996, section 5.4.4 (http://www.cacr.math.uwaterloo.ca/hac/)
4. The statistic relies upon the sum of the squares of the frequencies. (Limits for the other tests can be precomputed.) The use of a multiplier can be avoided by keeping track of the square instead of the raw count. This can be done efficiently by incrementing by increasing odd numbers instead by ones, noting that: $n^2 + (2 \cdot n + 1) = (n + 1)^2$
5. under fairly reasonable assumptions of the source characteristics. See M. Santha and U. V. Vazirani. Generating quasi-random sequences from semi-random sources. *Journal of Computer and System Sciences*, 33:75–87, 1986.
6. Peter Elias, The Efficient Construction of an Unbiased Random Sequence, *Annals of Mathematical Statistics*, 1972, Vol. 43, No. 3, pages 865-870
7. Yuval Peres, Iterating Von Neumann's Procedure for Extracting Bits, *Annals of Statistics* 1992, Vol. 29, No. 1, pages 590-597
8. Robert B. Davies, Exclusive OR (XOR) and hardware random number generators (2002)
9. M. Blum. Independent unbiased coin flips from a correlated biased source: a finite Markov chain. *Combinatorica*, 6(2):97–108, 1986
10. R. Shaltiel. Recent developments in explicit constructions of extractors, *Bull. Europ. Assoc. for Theor. Comput. Sci.*, vol. 77, pp. 67–95, June 2002
11. L. Trevisan, "Extractors and pseudorandom generators," *J. ACM*, pp. 860–879, 2001.
12. 100% of the NIST STS tests pass using the default parameters. For the MATLAB simulations 400 blocks of 2Mbits were used; for the hardware tests, 300 blocks of 500Kbits. The FFT test showed P-values skewed towards the high side in both cases, with all other tests having flat P-value distributions as expected. Note that the STS built-in Blum-Blum-Shub DRBG gives similar skewed results on the FFT test.