Improved Approaches to Online Health Testing in SP800-90 RNGs

David Johnston

dj.Johnston@intel.com

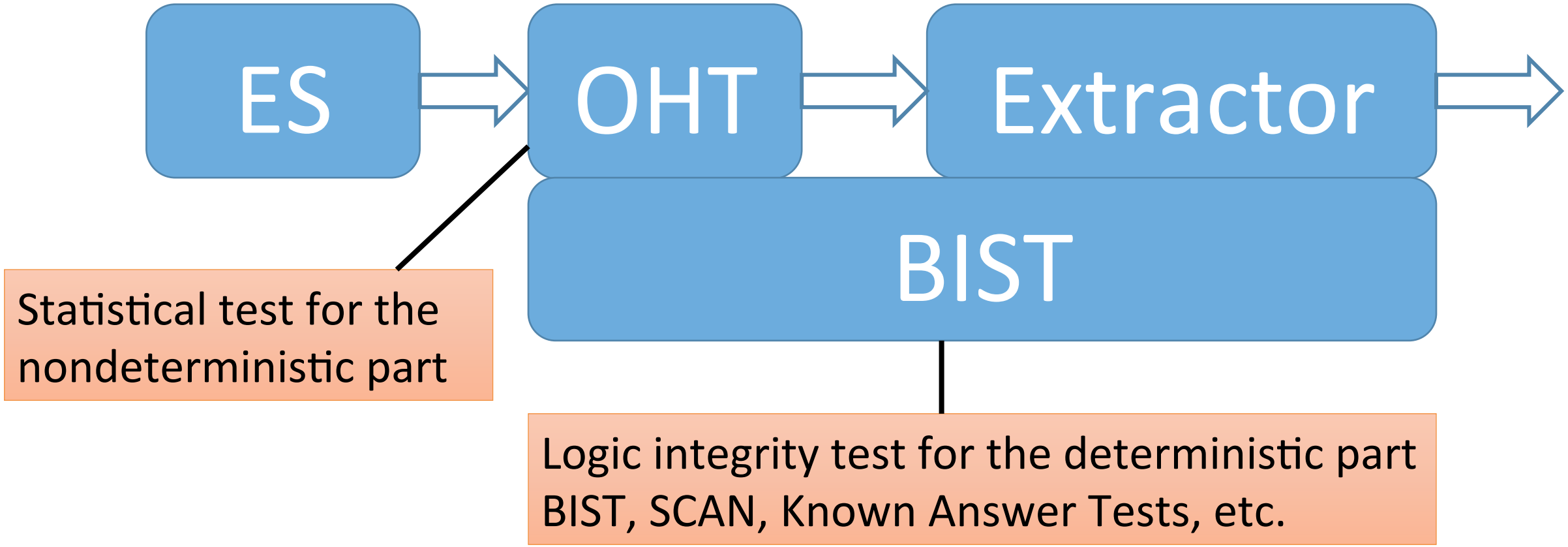
Find this file in PDF form at http://www.deadhat.com/slides/ICMC2015-dj.pdf

Random Number Generator, Basic Models

BIG and FAST

SMALL and SLOW

But You Also Need Online Testing



The OHT: But You Can't Test for Random!

- Min entropy tests are too slow and data hungry to do online
- All patterns are equally likely.
  - $P(X_i=00000000) = P(X_i=10011100) = P(X_i=10101010) = P(X_i=11111111)$
- Some patterns are characteristic of a broken state
  - $X_i=00000000$, $X_i=11111111$, $X_i=10101010$, $X_i=01010101$
  - E.G. Strong bias or strong serial correlation coefficient (SCC)

So you can only test for "Broken – Maybe"

A Nice Test For “Broken – Maybe”

Pattern	Min Freq	Max Freq	Model
1	96	159	127.5 ± 31.5
01	44	87	65.5 ± 21.5
010	9	58	33.5 ± 25.5
0110	4	35	19.5 ± 15.5
101	9	58	33.5 ± 25.5
1001	4	35	19.5 ± 15.5

- Note the binomial distribution of short and long patterns in a number of fully random bits.
- Set bounds for each
- Measure each over sample and check they are all within the bounds. If outside, tag as unhealthy
- The bounds determine the false positive rate

- It’s Cheap – A shift register, 6 comparators and 6 counters
- Spots all repeating patterns up to 6 bits in length and detects bias and correlation
- Highly bimodal with stationary data of some bias and auto correlation
- Intel CPUs do this over 256 bit samples and aims for 1% false positive

What Does SP800-90 Say?

- SP800-90C [4]: *When a failure is detected in an RBG component and reported to the RBG-as-a whole, the RBG **shall** enter an error state.*
  - For an Entropy Source OHT, what's an error, when all patterns are equally likely?
- SP800-90B [3]: *These tests are run continuously on all digitized samples obtained from the noise source, and so must have a very low probability of yielding a false positive.*

- But there is a tradeoff. A lower false positive rate implies a high false negative rate. So a very low probability of yielding a false positive means letting low entropy data pass to the conditioner without being caught

- We need a better scheme that accommodates a low false negative rate, with a high false positive rate of "Broken – Maybe" tagging of data while not entering error states on every false positive.

Entropy Pools Enable Adaptive Response

- If you discard the unhealthy tagged samples, you reduce the entropy
- If you accepts unhealthy tagged samples, you risk false negatives

"Never Throw Away Entropy" – Margaret Salter

- Extract with Output = MAC (last_output || MAC(X_i || … || X_{i+r})
  - Where n is the number of samples and r is the number of samples that contain the necessary number of healthy tagged samples. Also mixed in are the unhealthy samples that aren't counted.
  - Suspicious of MACing over variable field length? [7] Proves it is ok.

The Long Term Error Response

- Over many samples, remember just the 1 bit tag per samples – Allows a test over lots of data without huge amounts of memory
- Count the N Healthy:Unhealthy ratio in the last M samples
  - Intel CPUs have M=256 ➔ So the history statistic is over 64Kibit of data
- Good ES:
  HHHHHHHHHHUHHHHHHHHHHHHHHHHHHHHUHHHHHUHHHHHHH
- Marginal ES:
  UUUUUUUUUUUUUHUUUUUUUUUUUUUUUHUUUUUUUHUUUUUUUU
- A Transition from not-broken to broken:
  HHHHHHUHHHHHHHHHHHHHH<break>UUUUUUUUUUUUUUUUUU

What Makes Pool Feedback Good?

- E.G. Intel's CPUs demand 768 bits of healthy entropy, MACed to 256 bits of full entropy, but all intervening unhealthy samples are mixed in, so no entropy is thrown away and occasional false positives don't raise an error response.
- If the entropy quality reduces as an attack is mounted or the circuit starts to fail in some way – the unhealthy sample rate increases so more entropy is mixed into each seed.
- If the entropy source breaks, no more seeds are produced and the error response happens after ≤ 128 unhealthy samples are measured

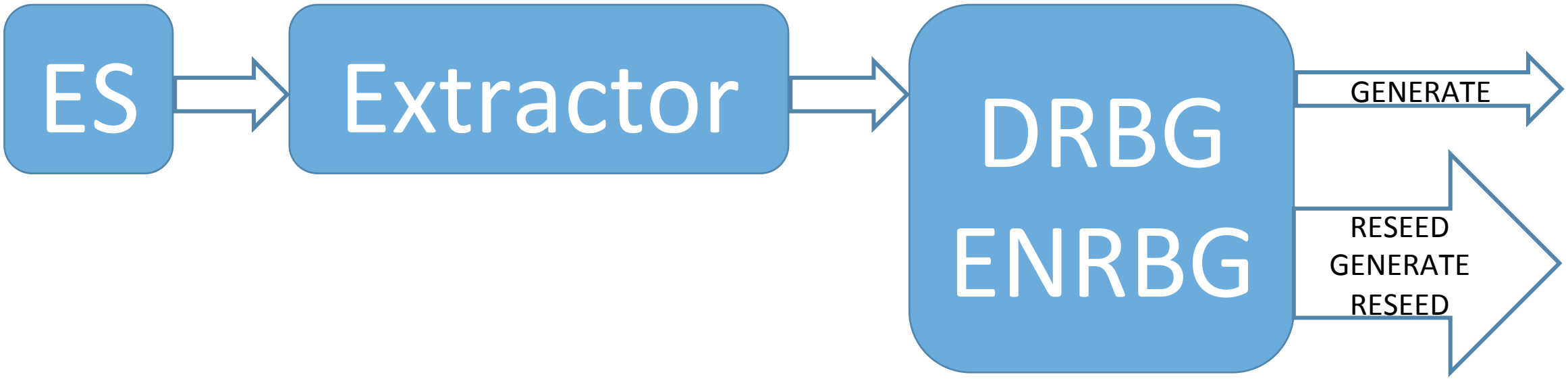
Instantaneous response to unhealthy samples ➔ increase the extraction ratio.

An error response if the ratio of Healthy:Unhealthy < 50% over 256 samples

What's Wrong with SP800-90C

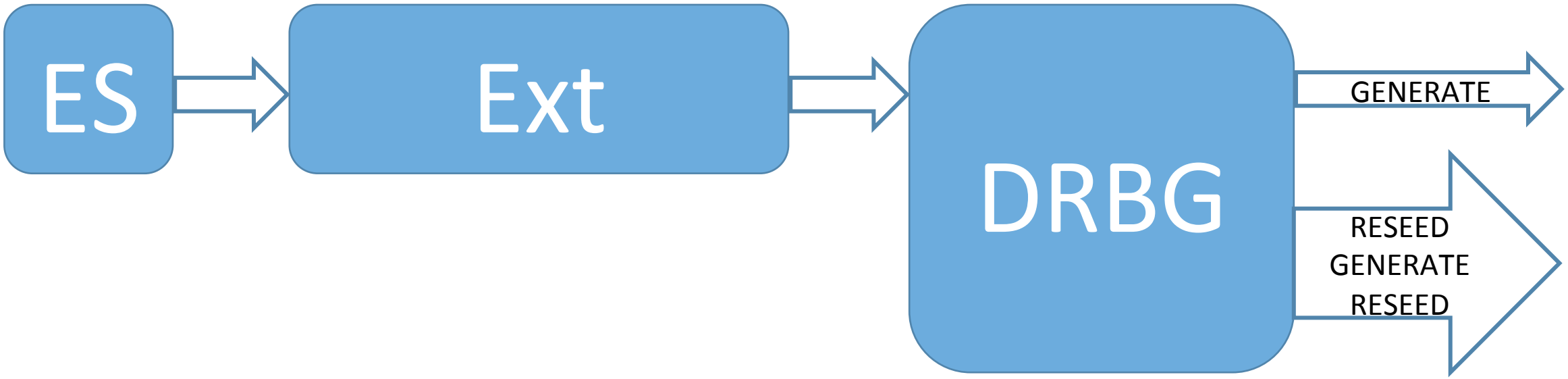
- The ENRBG output offers a superset of the DRBG's cryptographic properties, Full entropy vs. prediction computational complexity respectively. So a general purpose RNG needs both
  - DRBG for performance
  - ENRBG for arbitrary strength keys and seeding
    - (E.G. Intel's RdRand and RdSeed instructions = DRBG and ENRBG respectively)
- The ENRBG demands a DRBG also:
  - XOR Construction: Output: DRBG_output XOR Extractor Output
  - Oversampling Construction: Reseed, Generate, Reseed
- Both ES and Extractor+DRBG need testing. In silicon, the failure rate is proportional to the surface area.

Oversampling Construction



- Kills performance of DRBG output by forcing intervening reseeds
- Unless you put in two DRBGs, doubling the area, doubling the failure rate

XOR Construction



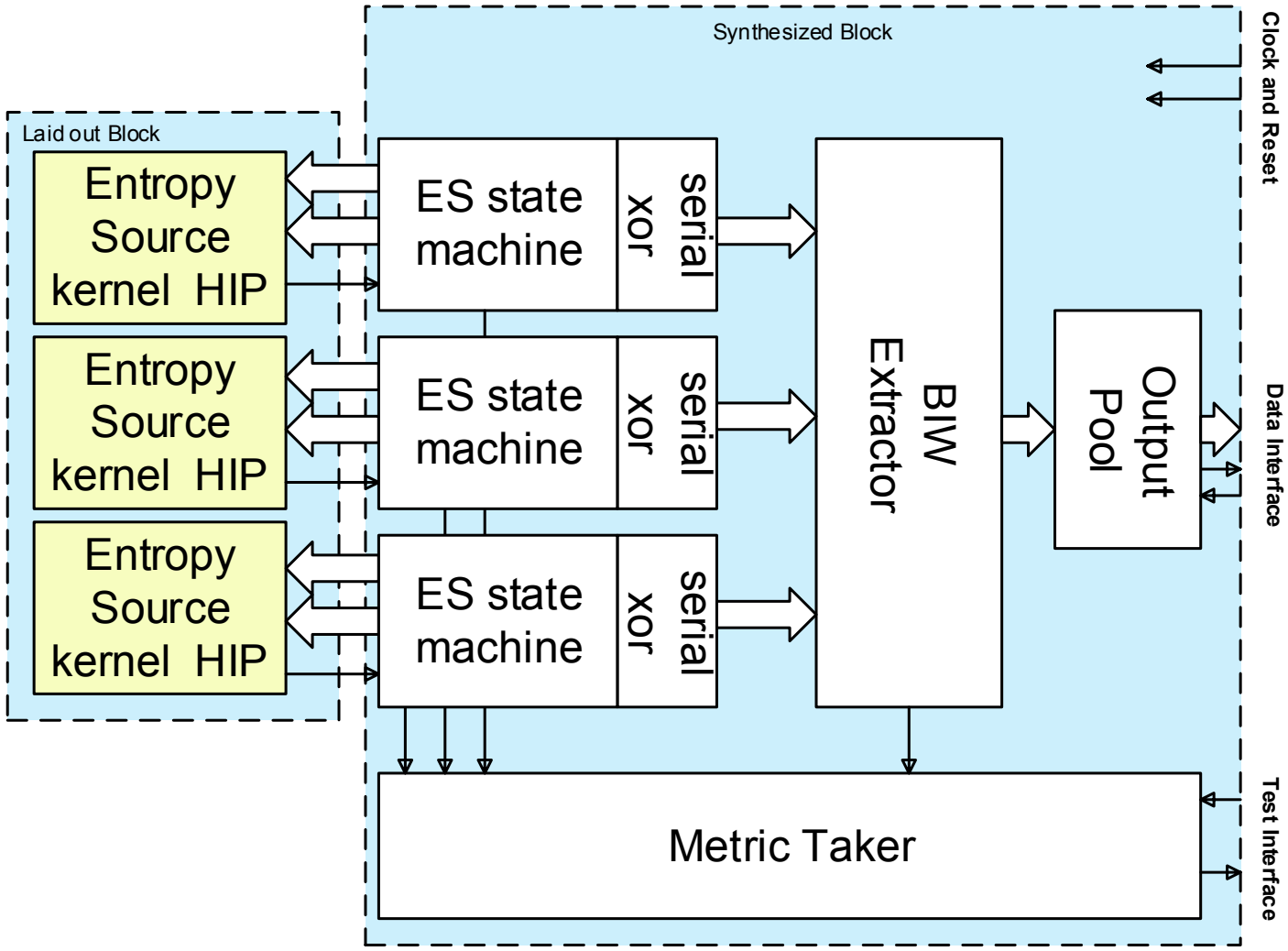
- Kills performance of DRBG output by forcing intervening reseeds
- Unless you put in two DRBGs, doubling the area, doubling the failure rate

Let’s Look at Surface Area

ES+Extractor
Full Entropy NRBG

BIW Extractor,
3 Entropy Sources
OHT

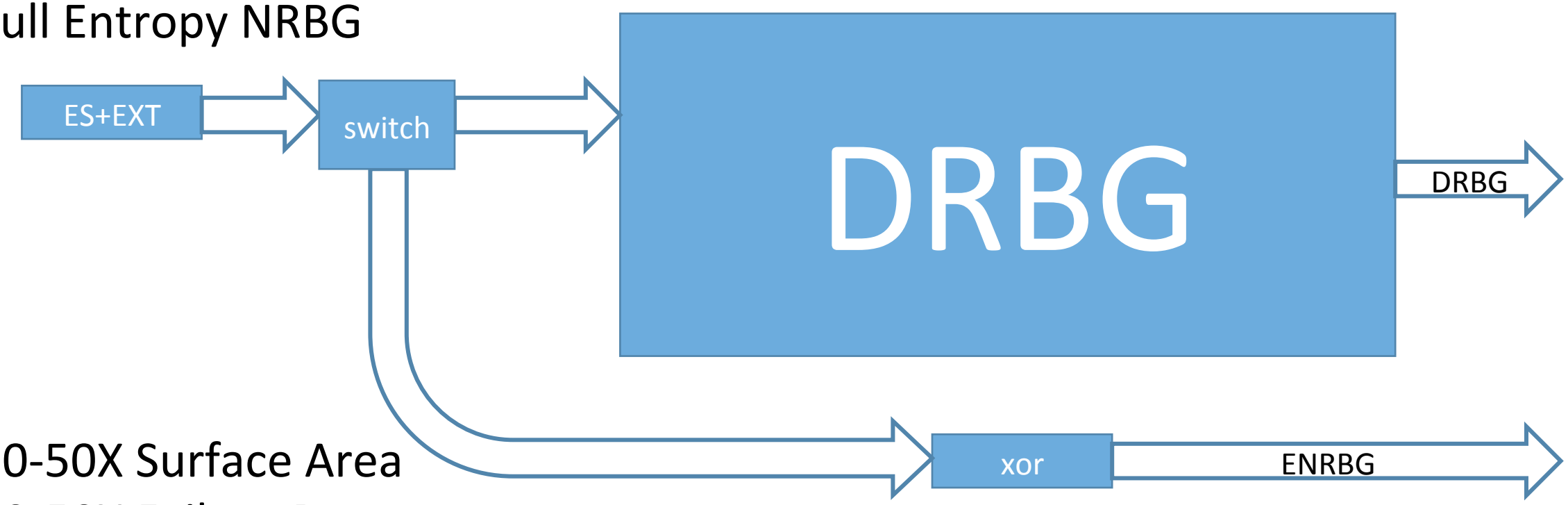
1000µm² 14nm [1]



Add a DRBG For XOR Construction

AES-CTR-DRBG
10 clock AES block cipher
~30K-100K gates (200Mhz – 2.5GHz)

ES+Extractor
Full Entropy NRBG



20-50X Surface Area
20-50X Failure Rate

Add 2 DRBGs For Oversampling Construction

40-100X Surface Area

40-100X Failure Rate

Better Reliability and Performance without DRBG

- A modern full entropy ES+Extractor has higher bits/clock/μm^2 than an AES-CTR-DRBG
  - ES+EXT+DRBG: Best Case of 128/10 clocks/31K gate equivalents = 4.13E-4 bits per clock per gate (Asymptotic as the gen:update ratio → 1.0)
    - Worst case 3X less efficient (1 AES to generate, 2 to update)
  - ES+EXT: Actual Case = (1/12)/1/800 gate equivalents = 1.04E-4
- So without the DRBG, it is 4X more efficient and 30X more reliable.
- Get the same performance by taking ¼ of the surface area with parallel ES+EXT blocks.
- Get greater reliability by merging the outputs of multiple ES+EXT blocks.

And Then Comes FIPS 140-2

- Section 4.9.2 "*If each call to a RNG produces blocks of n bits (where n > 15), the first n-bit block generated after power-up, initialization, or reset shall not be used, but shall be saved for comparison with the next n-bit block to be generated. Each subsequent generation of an n-bit block shall be compared with the previously generated block. The test shall fail if any two compared n-bit blocks are equal*."
- This yields a random stream trivially distinguishable from Random
  - No 16 bit equal pairs in 1MByte data = Definitely not random, 16 are expected.
- It creates algebraic invariates $X_i \neq X_{i+1}$ For all output values, reducing entropy and helping algebraic attacks
- Intel refused to put this in its silicon because it may be a back door
  - The risk to Intel of having a back door is greater than the cost of not being FIPS compliant
- ISO 19790-2012 [6] removed this test – Let's hurry up with FIPS 140-3
- But still resource constrained devices can't have hardware FIPS compliant RNGs because of the DRBG requirement.

Summary

- Pool structures that use health tagging allow appropriate adaptive responses to entropy source failure and degradation behavior and instantaneous response to instantaneous ES failure.
- The DRBG requirements of SP800-90C lead to a reduction in reliability and/or efficiency of RNGs and prevent SP800-90 compliant full entropy hardware RNGs in resource constrained situations
- FIPS 140-2 [5] makes it worse by reducing the security of the RNG output

References

[1] µRNG: A 300-950mV 323Gbps/W All-Digital Full-Entropy True Random Number Generator in 14nm FinFET CMOS. Sanu Mathew, David Johnston†, Paul Newman*, Sudhir Satpathy, Vikram Suresh, Mark Anders, Himanshu Kaul, Gregory Chen, Amit Agarwal, Steven Hsu, Ram Krishnamurthy (ESSCIRC-2015)

[2] SP800-90A Rev 1

[3] SP800-90B - Most Recent Draft (June 2012)

[4] SP800-90C – Most Recent Draft (June 2012)

[5] FIPS 140-2 (12-03-2002)

[6] ISO 19790-2012

[7] A Provable-Security Analysis of Intel's Secure Key RNG, Thomas Shrimpton(B) and R. Seth Terashima, Department of Computer Science, Portland State University, Portland, USA, {teshrim,seth}@cs.pdx.edu (IACR 2015)