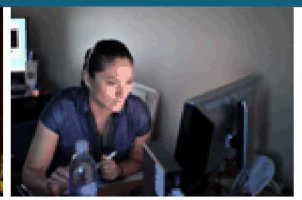
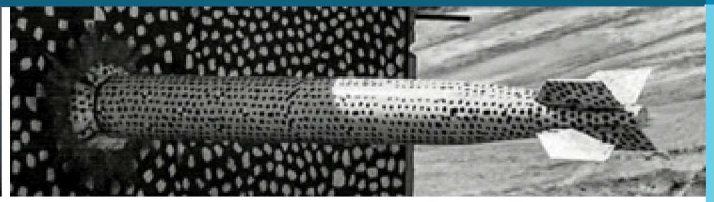
SAND2020-2052PE

Evaluating Physical Unclonable Functions

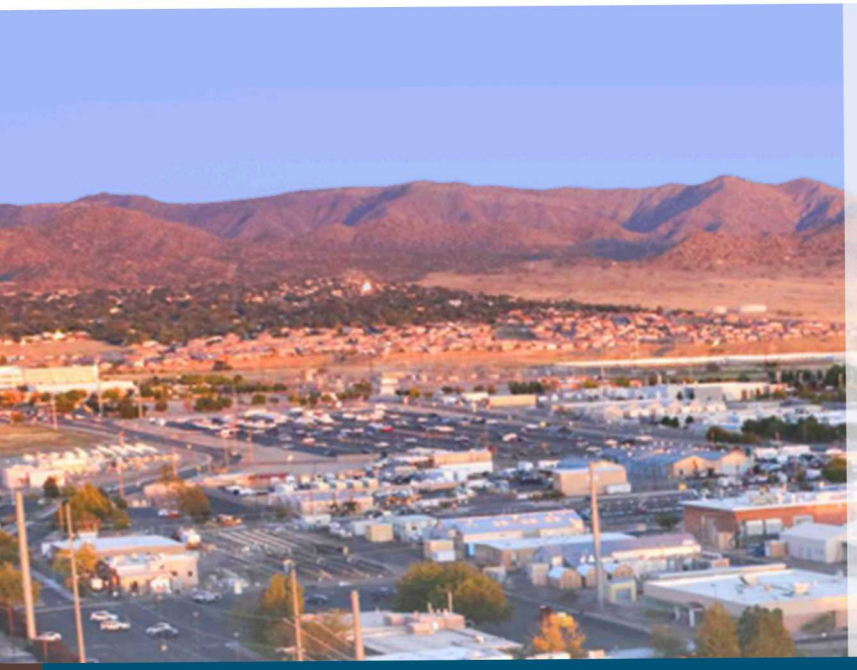




Presented by

Ryan Helinski

Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.



What is a PUF?

Physical Unclonable Function

Physical Unclonable Functions (PUFs)

A function f which takes an m-bit challenge C and produces an n-bit response R

$$f(C) = R$$

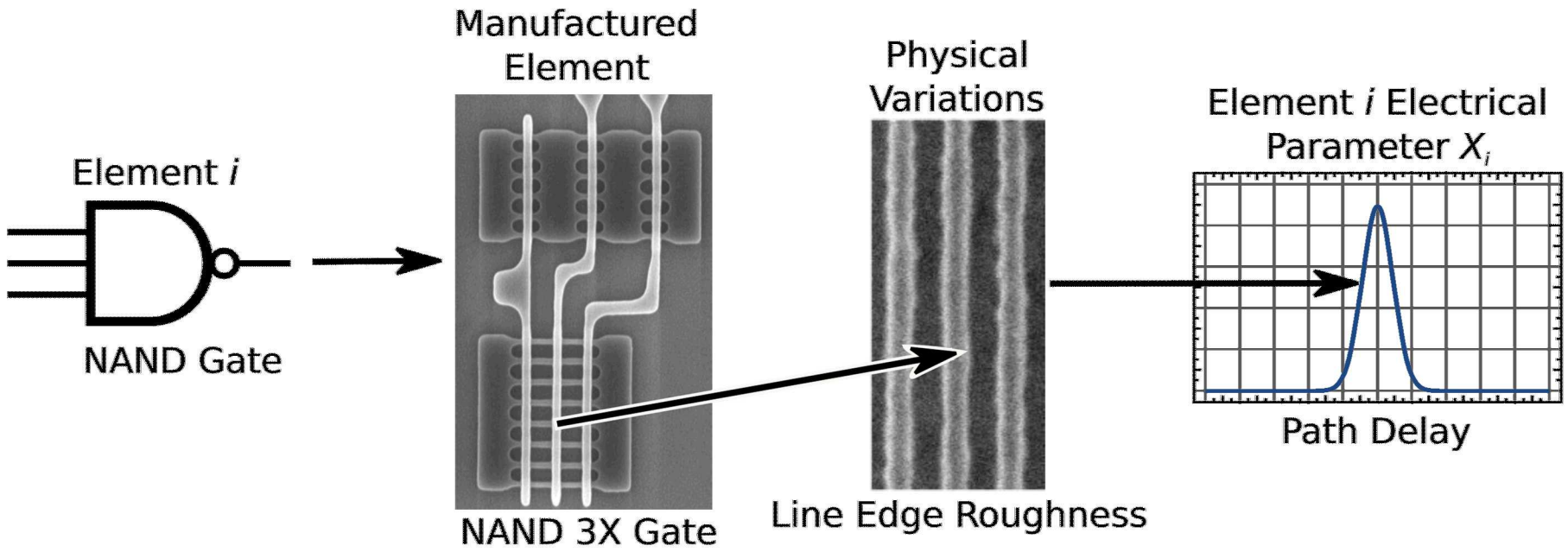
$$\text{where } C \in \{0,1\}^m, \quad R \in \{0,1\}^n$$

Often, $m = n$. For example, $m = n = 1024$ bits.

More often, the PUF has a null challenge with $m = 0$ and $n > 0$. For example, $m = 0,\ n = 256$.

Most PUFs claim to derive their randomness from manufacturing variation in some particular physical aspect of these elements. Examples of this include interconnect edge roughness, transistor length variation, variation in semiconductor doping concentrations and dielectric material impurities.

Extracting Manufacturing Process Variations

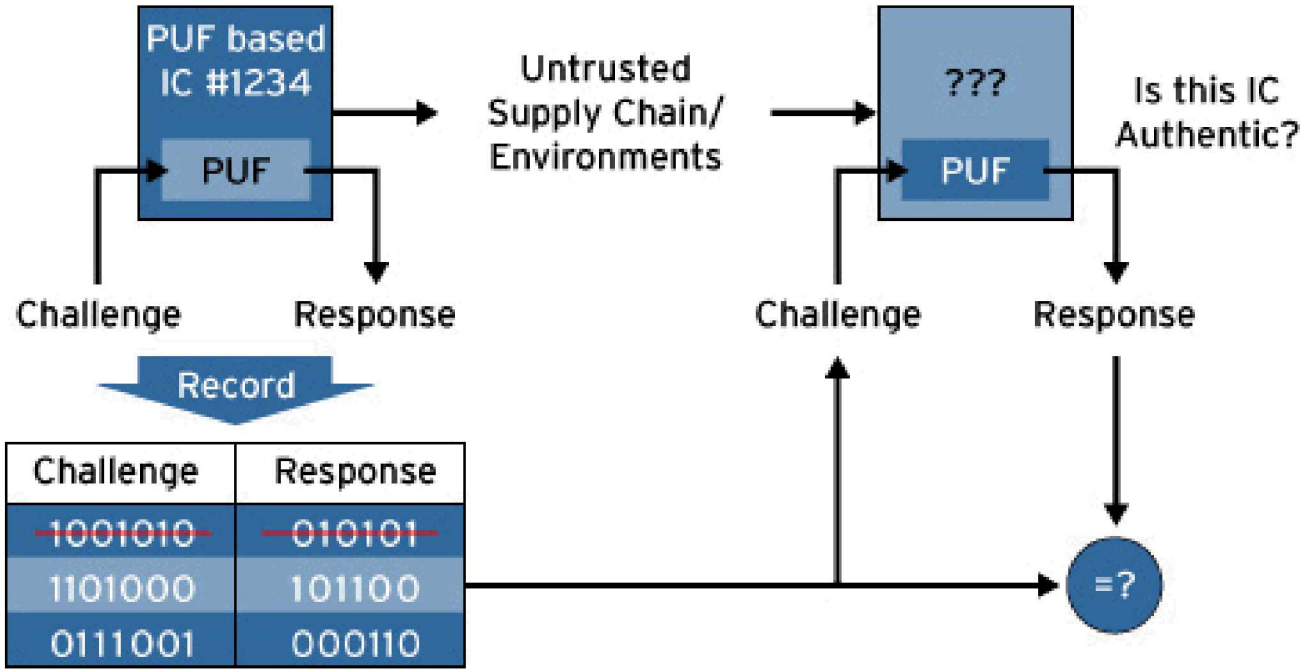


Most PUFs compare two circuits against each other (e.g., the metastable state)

- Two transistors
- Two ring oscillators
- Two multi-staged delay paths

Some PUFs compare elements against a reference value

The Classic PUF Protocol



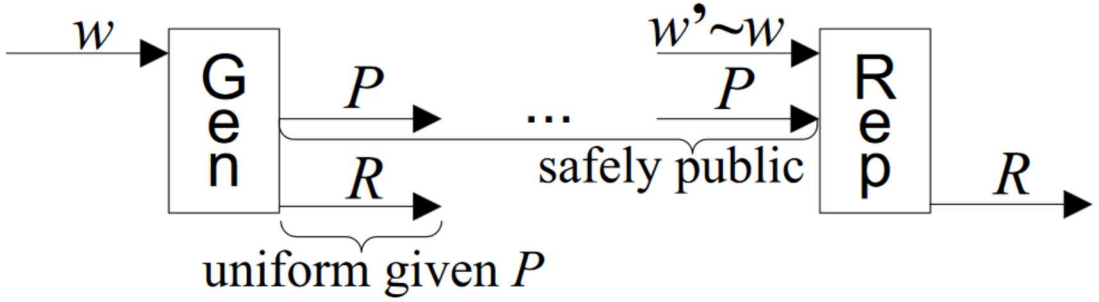
During enrollment, a large number of challenge/response pairs are stored for each PUF

During deployment, a new challenge can be issued each time and the Hamming distance of the new response and the original response is tested to see if it is "close enough"

Each challenge/response pair should be used only once

Image credit: Authentication with challenge/responses pairs, from Verayo, Inc.

Fuzzy Extraction



Typically, some bits (or sites, or locations, etc.) are very error-prone but the application requires the same response R to be regenerated each time.

One approach is to compute some "helper data" P so that the initial response R can be regenerated at a later time. This can be done using an initial response w taken during enrollment and then use this data P to correct the bits that have changed in a later w' to regenerate R exactly.

This helper data is considered public data. However, if $P \in \{0,1\}^q$ it is assumed that P reveals up to q bits about the response R. Therefore, q is chosen such that $q < n$.

Image credit: Fuzzy extractor consisting of generation and reproduction steps, from Dodis et al., 2004

Weak or strong?

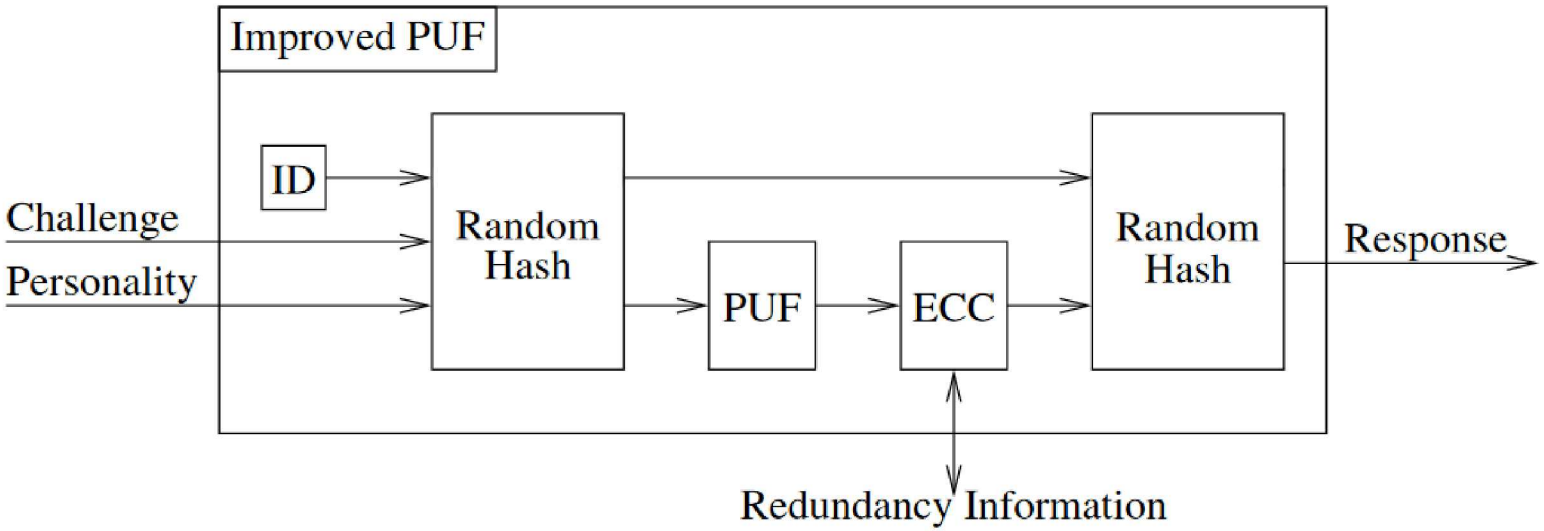
There is a distinction in the literature between "weak" and "strong" PUFs

- A weak PUF : small challenge space, e.g., $m < 32$
- A strong PUF : "cryptographically strong" challenge space, e.g., $m \geq 128$

However, there have been a large number of papers presenting attacks on so-called "strong PUFs"

- Model-building

Improving the Challenge/Response Space



ID: for example, stored in non-volatile memory (NVM)

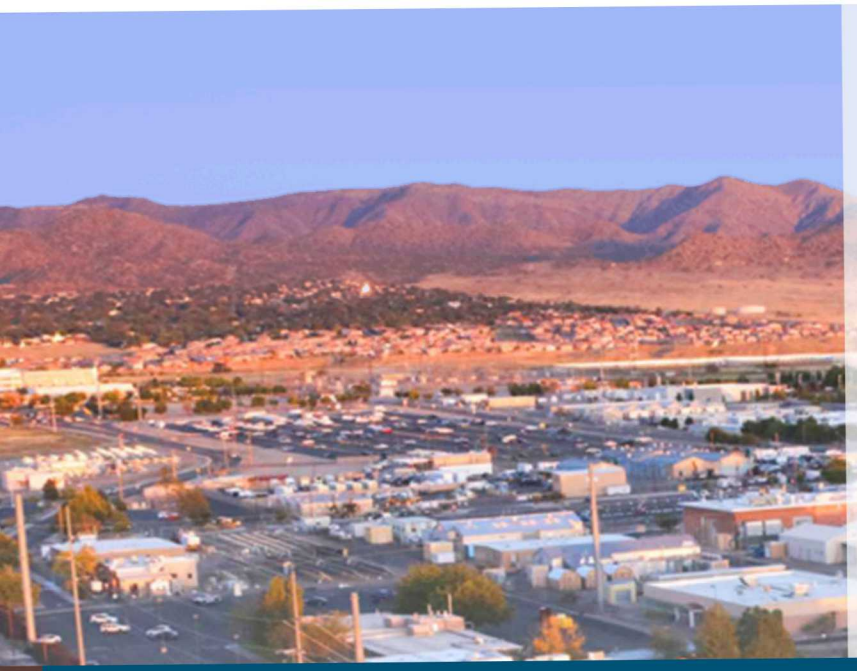
Personality: can be changed to re-enroll the PUF, can be stored in NVM

Redundancy Information: helper data, can also be stored in NVM

If the PUF does not have a challenge ($m = 0$) or if the challenge/response space is otherwise diminished, then the approach shown in the figure could be used to bring the PUF up to ideal performance.

However, this adds a significant amount of computational overhead.

Image credit: PUF control circuitry from Gassend et al., 2002



How can we characterize PUFs?

It's all about performance

Common PUF Metrics and Ideal Values

Property	Characteristic	Identifier	Ideal Value
Mean value	Bias, randomness	μ or $P(b)$	$0.5, \quad \sigma = \frac{\sqrt{n}}{2}$
Error rate or fraction of noisy bits	Reliability	$\mathrm{HD}_{\mathrm{noise}}$ (or $\mathrm{HD}_{\mathrm{intra}}$ if $m = 0$)	$\mu = 0$
Autocorrelation between response bits	Randomness	R_{xx}	$\mu = 0, \quad \sigma = \frac{1}{2\sqrt{n}}$
Hamming distance between responses to different challenges	Randomness	$\mathrm{HD}_{\mathrm{intra}}$	$\mu = 0.5, \quad \sigma = \frac{\sqrt{n}}{2}$
Hamming distance of responses between devices	Uniqueness	$\mathrm{HD}_{\mathrm{inter}}$	$\mu = 0.5, \quad \sigma = \frac{\sqrt{n}}{2}$
Statistical test value	Randomness	v or P-value	$\mu = 1 - \alpha$

$\alpha :=$ confidence level (e.g., 0.05)

What is "random"?

A random event (e.g., a bit value) is an event that has a non-deterministic outcome. The means that any information gathered about the system's current or previous states does not allow the next outcome of the next event to be predicted with absolute certainty.

Idealized example from NIST: An ideal random bit sequence could be generated by flipping an unbiased (fair) coin with sides that are labeled "0" and "1"

- Unbiased: each flip has a probability of exactly ½ of producing a "0" or "1"
- Flips are independent of each other: the result of any previous coin flip does not affect future coin flips

What if you already have bit strings and want to know if they might have come from an ideal source of randomness (like our coin flips)?

- Look for signs of patterns using a standard set of tests (NIST 800-22, Diehard tests, etc.)
- If a pattern is found in too many of the bit strings, then the source is not random

Statistical Hypothesis Testing

Null hypothesis: the data is unbiased and independent and identically distributed (IID), i.e., it came from an ideal source

Reference distribution: the distribution of test values for truly random inputs; CDF gives P-values

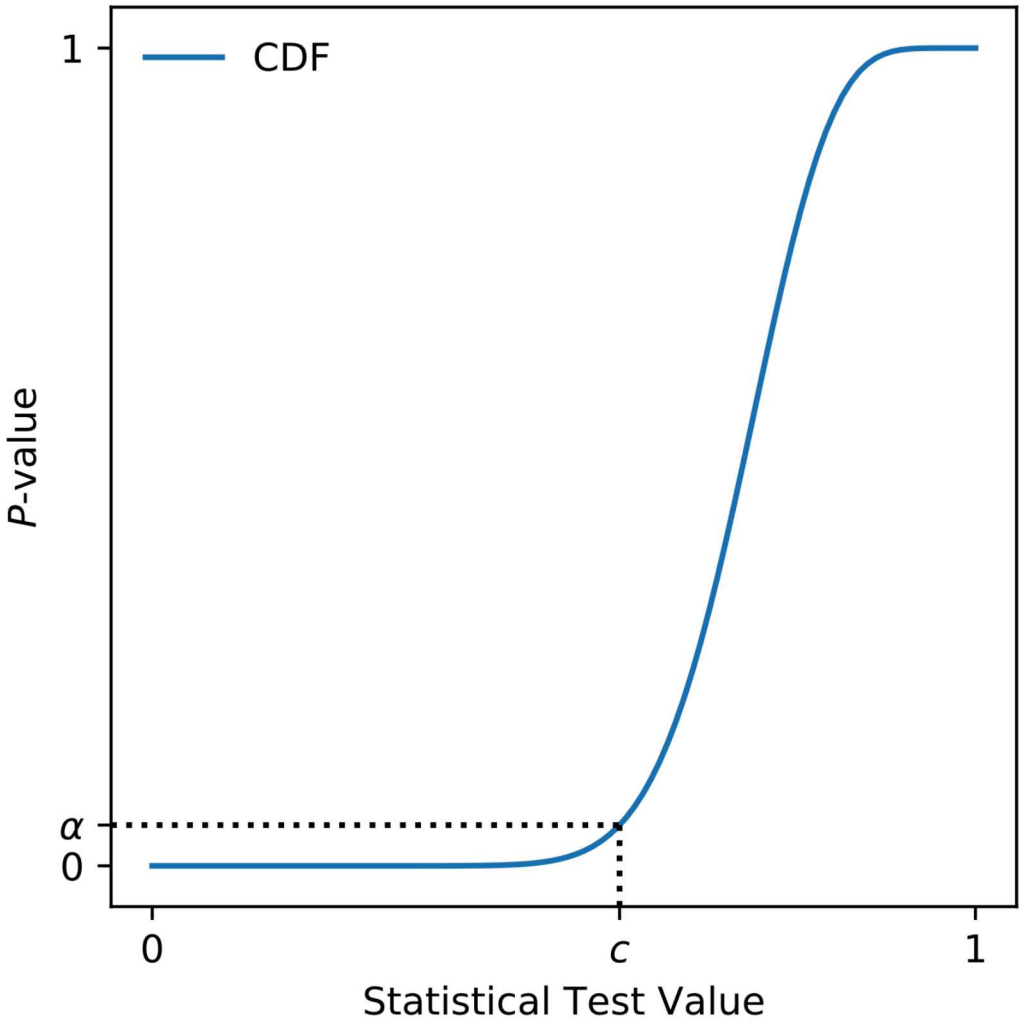
Significance level α: fraction of statistical test values that are below c for the reference distribution, e.g., $\alpha = 0.05$

Critical value c: $1 - \alpha$ of the statistical test values lie above this threshold

Decision rule: if P-value is greater than significance level α, then accept the null hypothesis

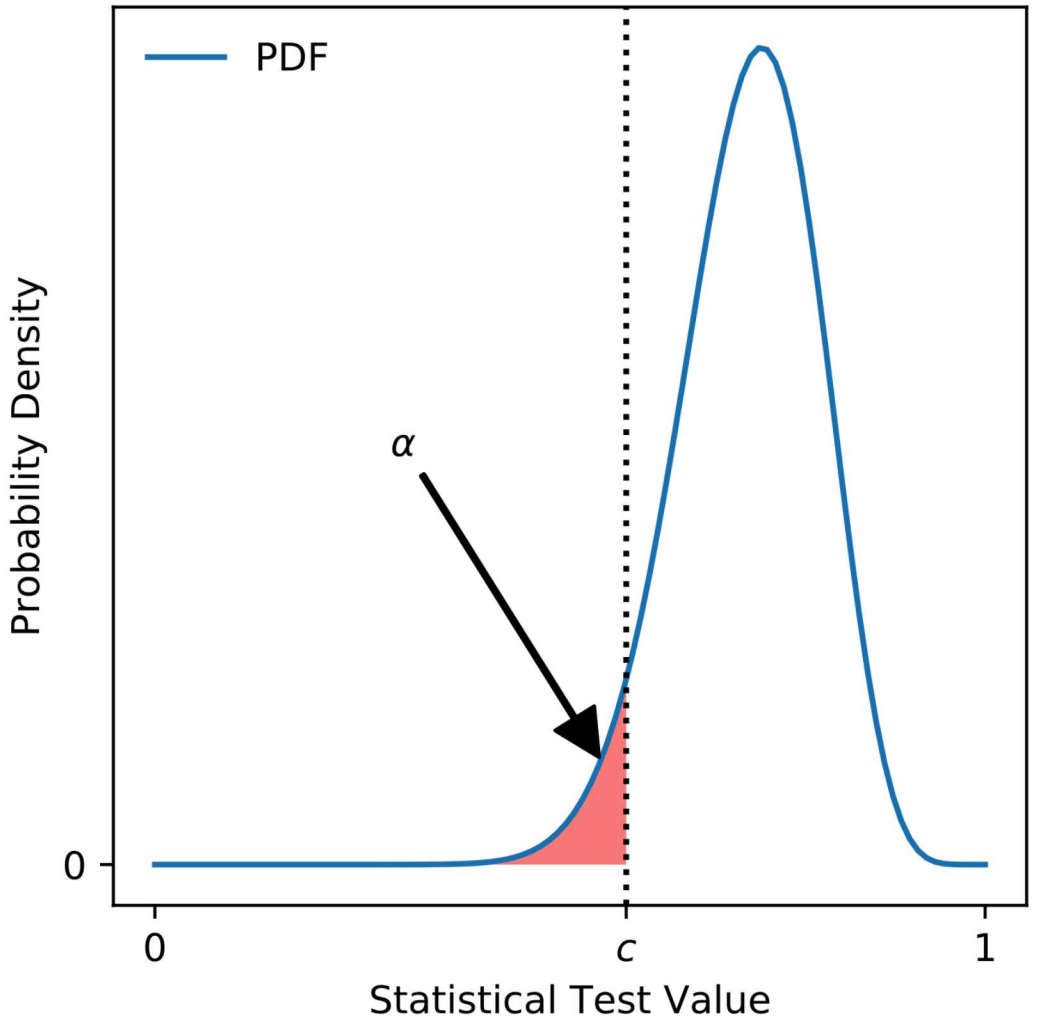
Important: The average fraction of the time that the null hypothesis should be rejected for a truly random source is α

- Your random source should pass a test on average $1 - \alpha$ of the time, for example 95% of the time if $\alpha = 0.05$



Statistical Hypothesis Testing

Corresponding probability distribution function (PDF)



Statistical Hypothesis Testing – NIST SP 800-22

1. The Frequency (Monobit) Test
2. Frequency Test within a Block
3. The Runs Test
4. Tests for the Longest-Run-of-Ones in a Block
5. The Binary Matrix Rank Test
6. The Discrete Fourier Transform (Spectral) Test
7. The Non-overlapping Template Matching Test
8. The Overlapping Template Matching Test
9. Maurer's "Universal Statistical" Test
10. The Linear Complexity Test
11. The Serial Test
12. The Approximate Entropy Test
13. The Cumulative Sums (Cusums) Test
14. The Random Excursions Test
15. The Random Excursions Variant Test.

Some tests require inputs of 1,000,000 bits or longer to achieve the desired statistical significance

What if your random source is not ideal?

It is *typical* for random sources to deviate from an ideal random source

There are established means of dealing with low entropy, e.g., "conditioning" defined in SP 800-90B

- The conditioning component is deterministic
- The entropy of the output is at most the entropy of the input
- Ideally, the number of output bits is reduced but the entropy becomes ideal (1 random bit/1 output bit)

NIST SP 800-90B recommends several vetted algorithms including

- Keyed functions HMAC (FIPS 198), CMAC (SP 800-38B), CBC-MAC (SP 800-90B, Appendix F)
- Unkeyed functions like SHA-1 through SHA-512 (FIPS 180), SHA-3 (FIPS 202)

Probability of Response Aliasing [Helinski et al. 2009]

For a chosen confidence level, what is the likelihood that two devices within a sample will have responses for a given challenge that are too close?

Can be estimated empirically from measurements of inter-device and noise variation

- Inter-device Hamming distances: between responses of different PUFs
- Noise (sometimes intra-device) Hamming distances: between repeated measurements

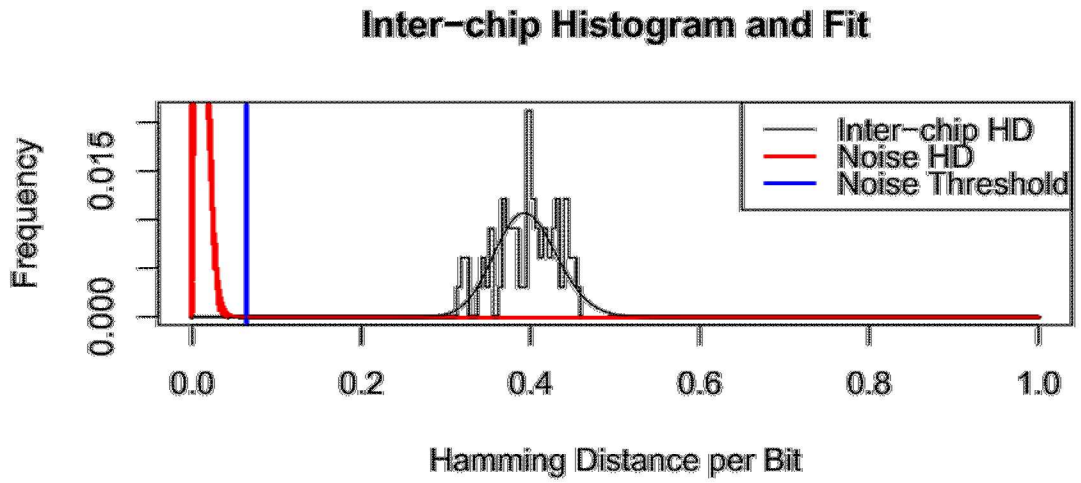
Inter-device and Noise Probability Distribution Functions

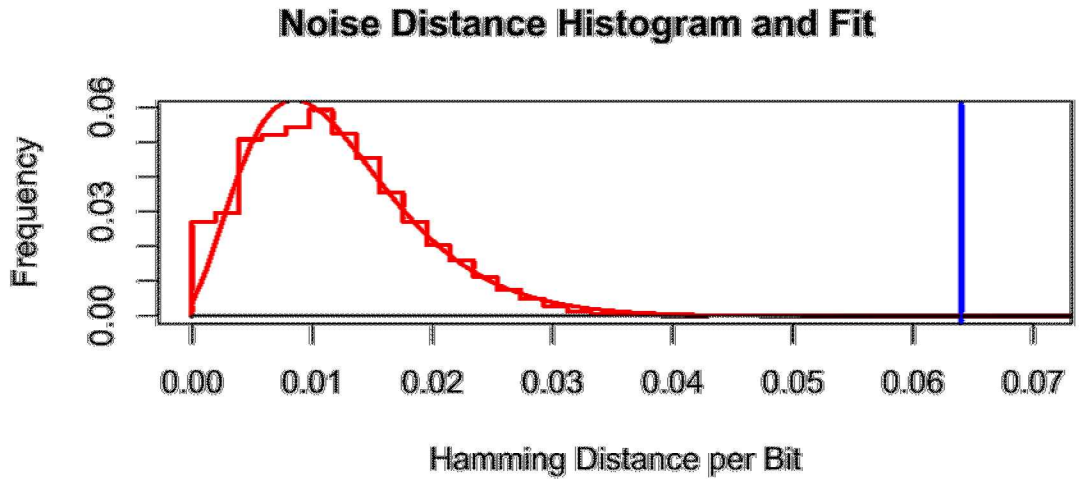
Compute $\binom{N}{2}$ inter-device Hamming distances

Compute $\binom{R}{2}$ noise Hamming distances, where R is the number of repeated measurements

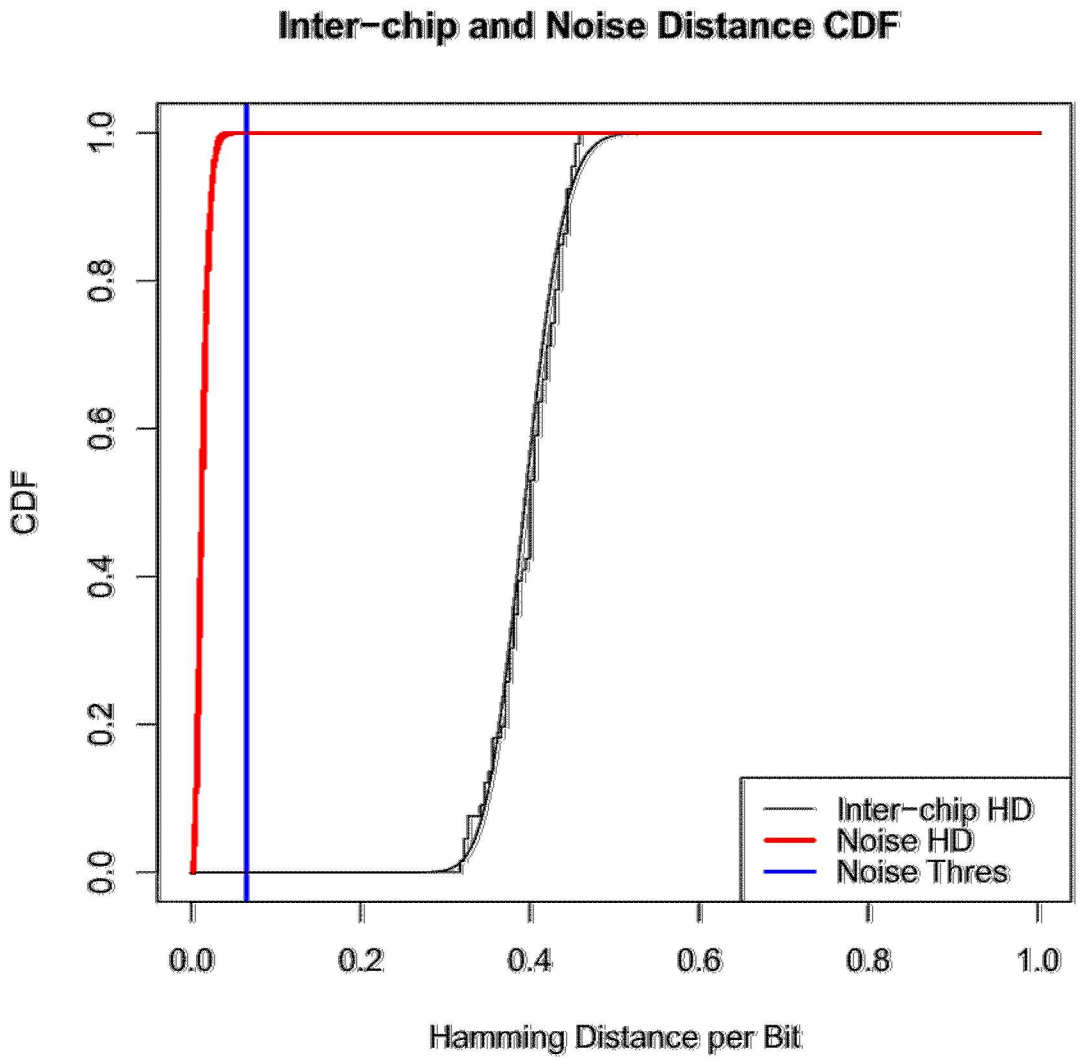
Fit both sets of distances to (discrete) negative binomial distributions

This provides estimates of the cumulative distribution functions $F_{\text{inter-device}}(x)$, $F_{\text{noise}}(x)$





Inter-device and Noise Cumulative Distribution Functions



Choose desired confidence level; e.g., 99.7%, 99.99%, 99.9999%, etc.

Find corresponding Hamming distance from $F_{\text{noise}}(x)$, this is the noise threshold

Plug this distance threshold into $F_{\text{inter-device}}(x)$ to obtain the estimated fraction of devices that will have a response distance less than this noise threshold

Probability of Aliasing Caveats

Concisely, *the expected fraction of inter-device distances that are smaller than the expected largest noise distance*

Exponentially sensitive to the noise distances (e.g., if temperature cannot be controlled)

Reasonable values are greater than 2^{-n} and orders of magnitude less than $1/N$
where n is the number of response bits and N is the number of devices that may ever be produced.

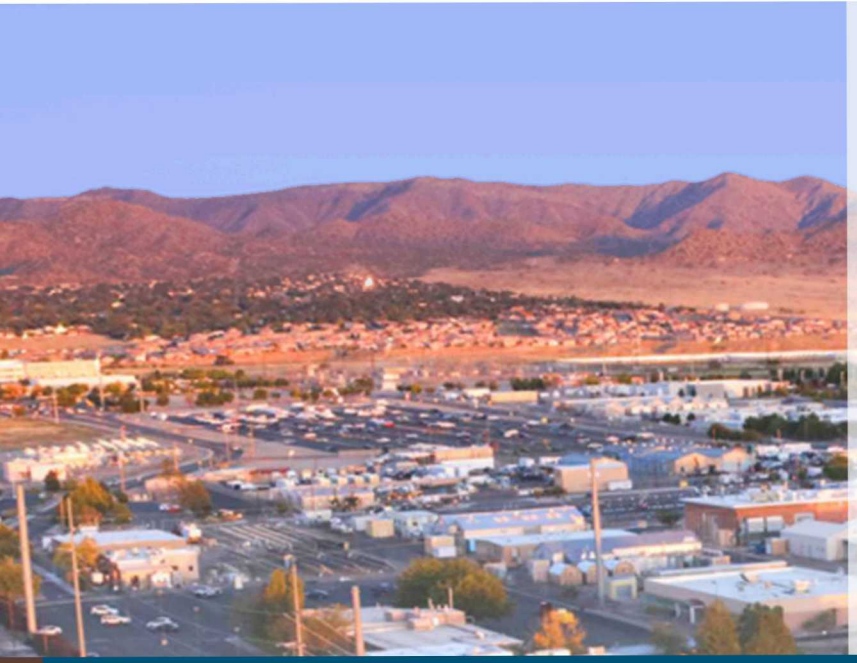
- e.g., $2^{-1024} < P(\text{alias}) < 1/10^{10}$ for $n = 1024$ and $N = 100{,}000{,}000 = 1 \times 10^{8}$
  i.e., $5.56 \times 10^{-309} < P(\text{alias}) < 1 \times 10^{-10}$

n	2^n
128	1×10^{38}
256	1×10^{77}
512	1×10^{154}
1024	1×10^{308}

Probability may increase after error correction

- E.g., correcting 128 bits reduces the total 1024-bit space to 896
- Shifts the inter-chip HD distribution toward 0

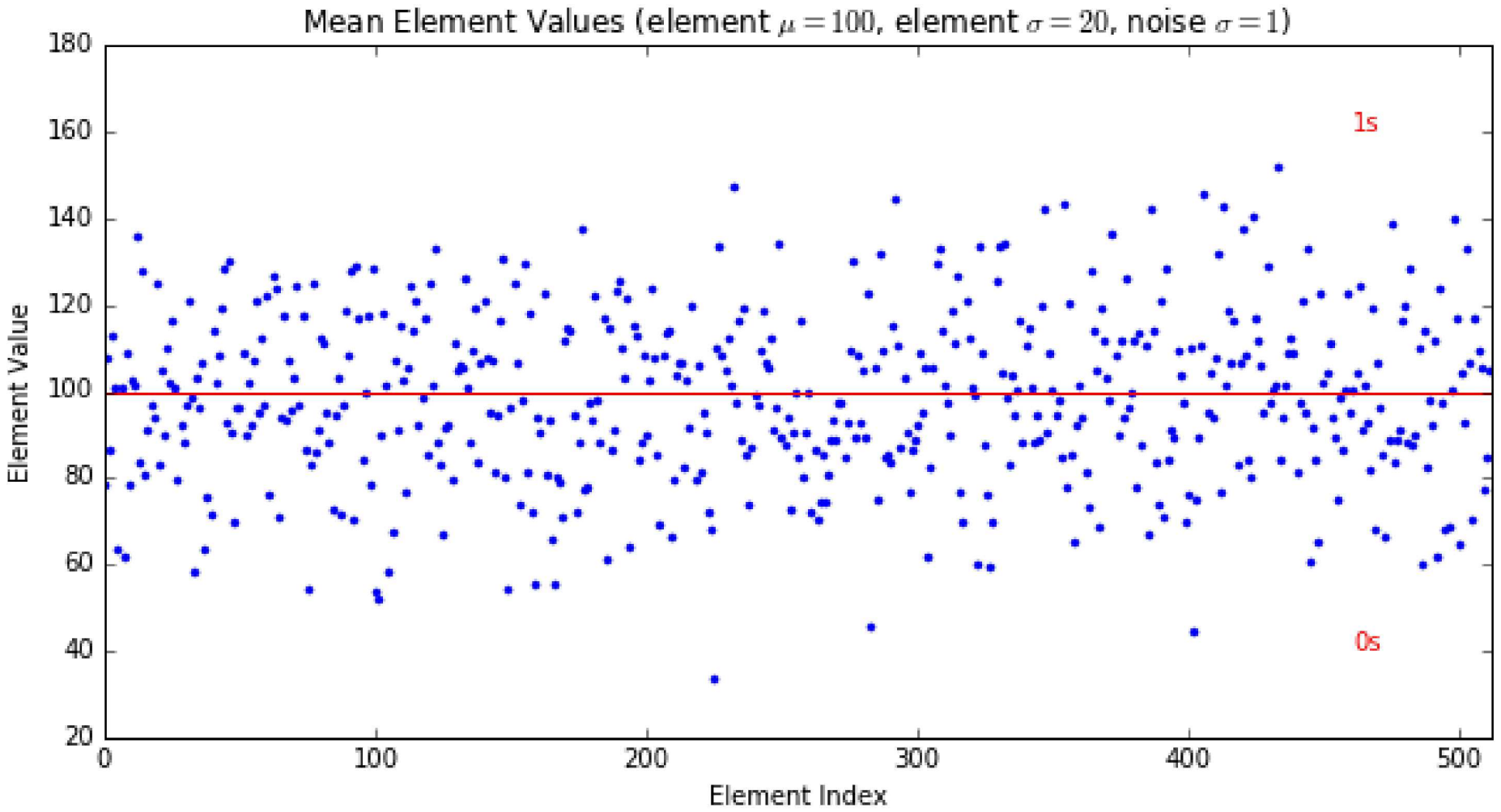
On the other hand, the probability of aliasing could be estimated a priori
to guide choosing how much error correction to use



Noisy Bits and How to Deal with Them without Error Correction

Keep out zones, helper data, etc.

Example measured (or averaged) element values



Histogram of averaged element values

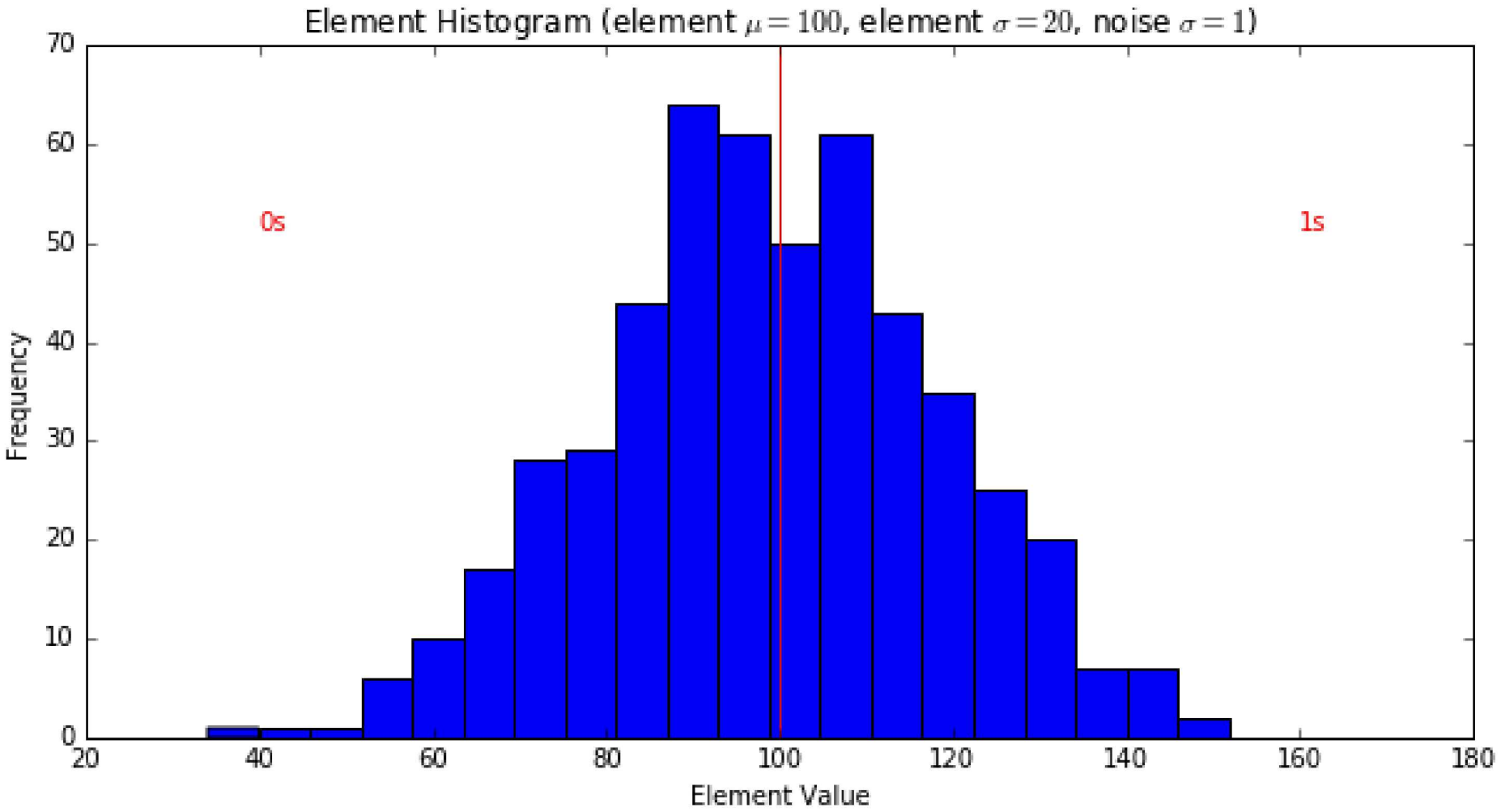
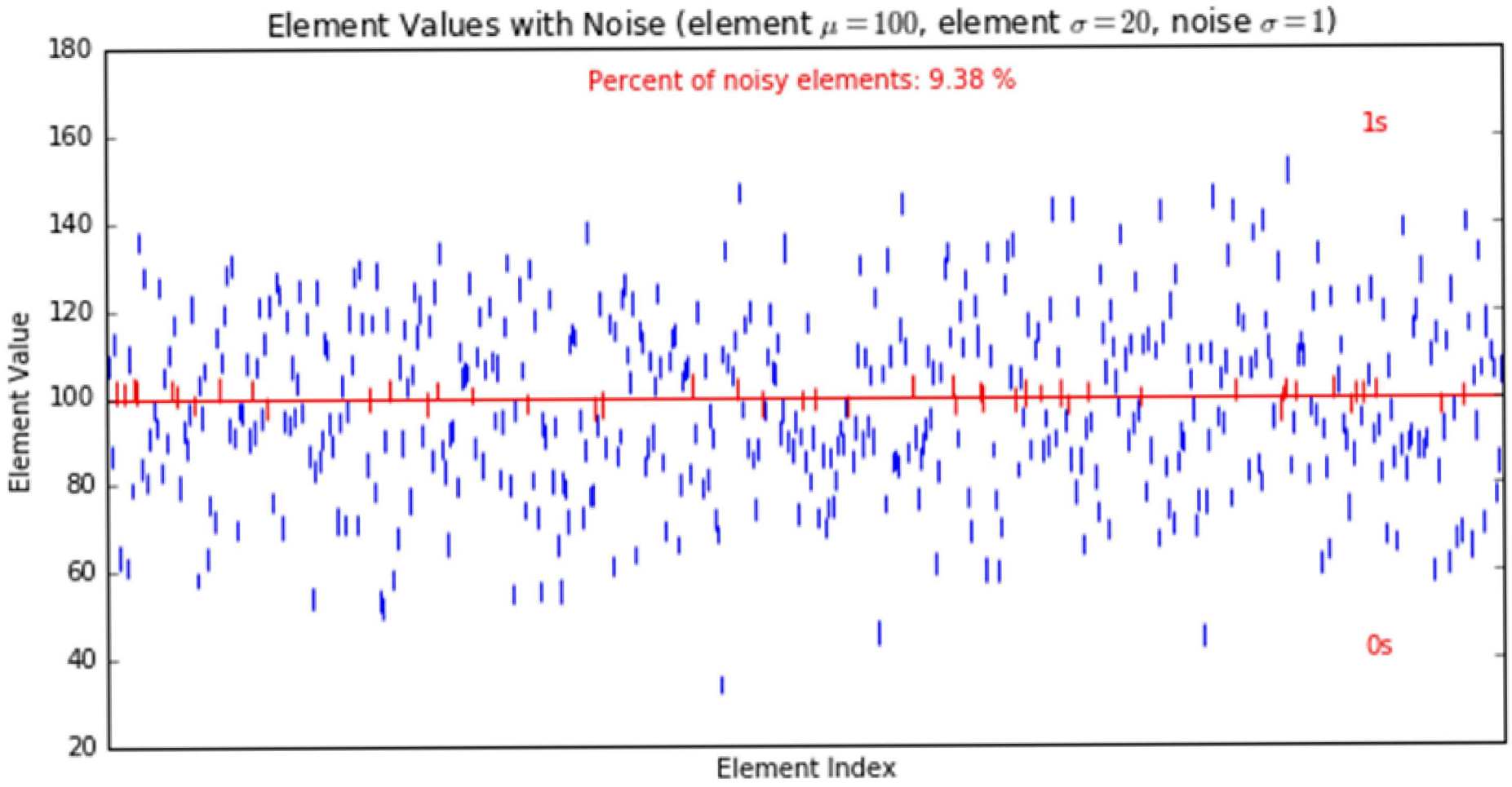


Illustration of element value noise



How can we deal with noisy bits?

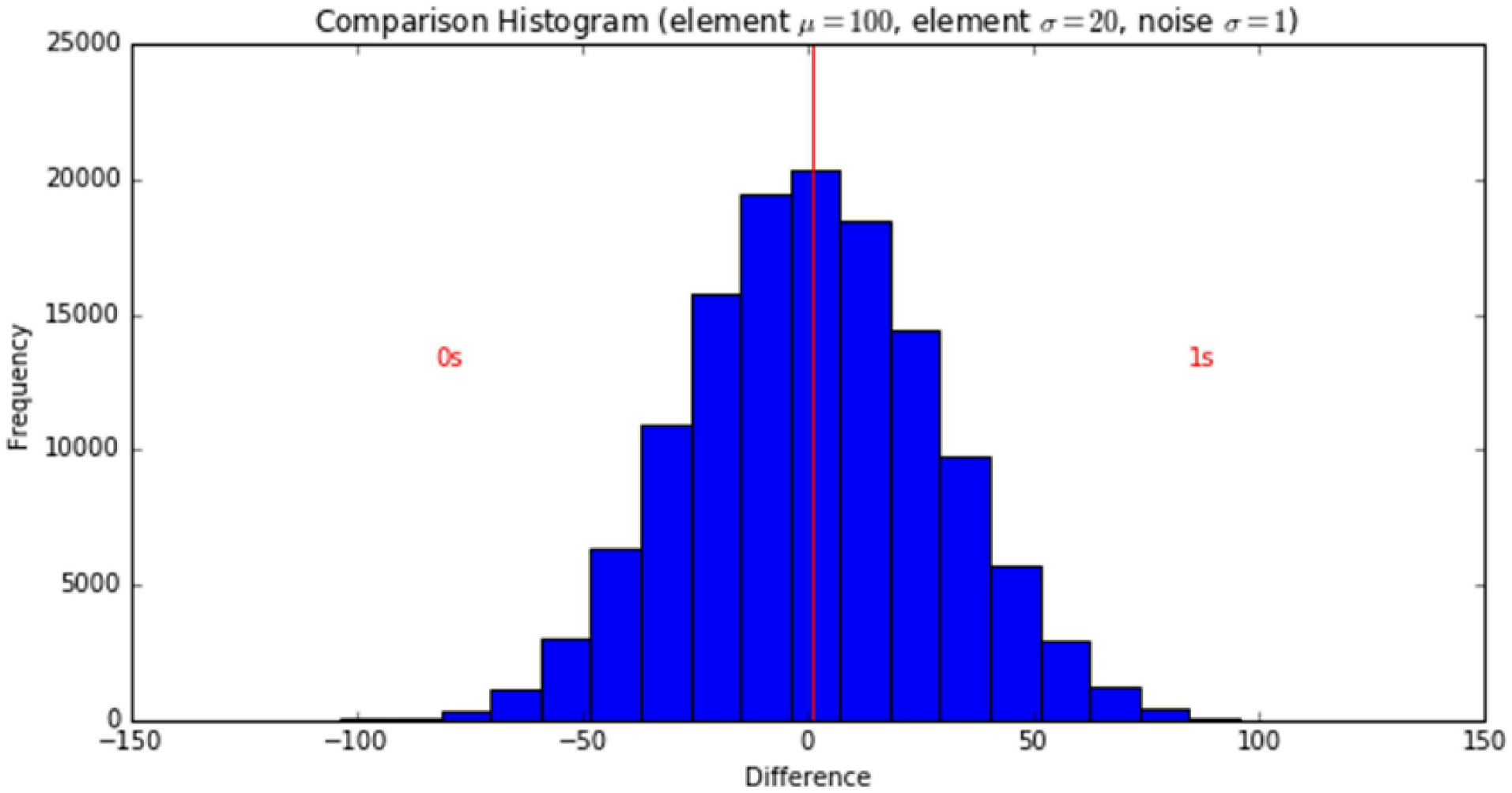
These sites that are noisy may be excluded based on

- How close they are to the reference value
- If they flip among several subsequent measurements

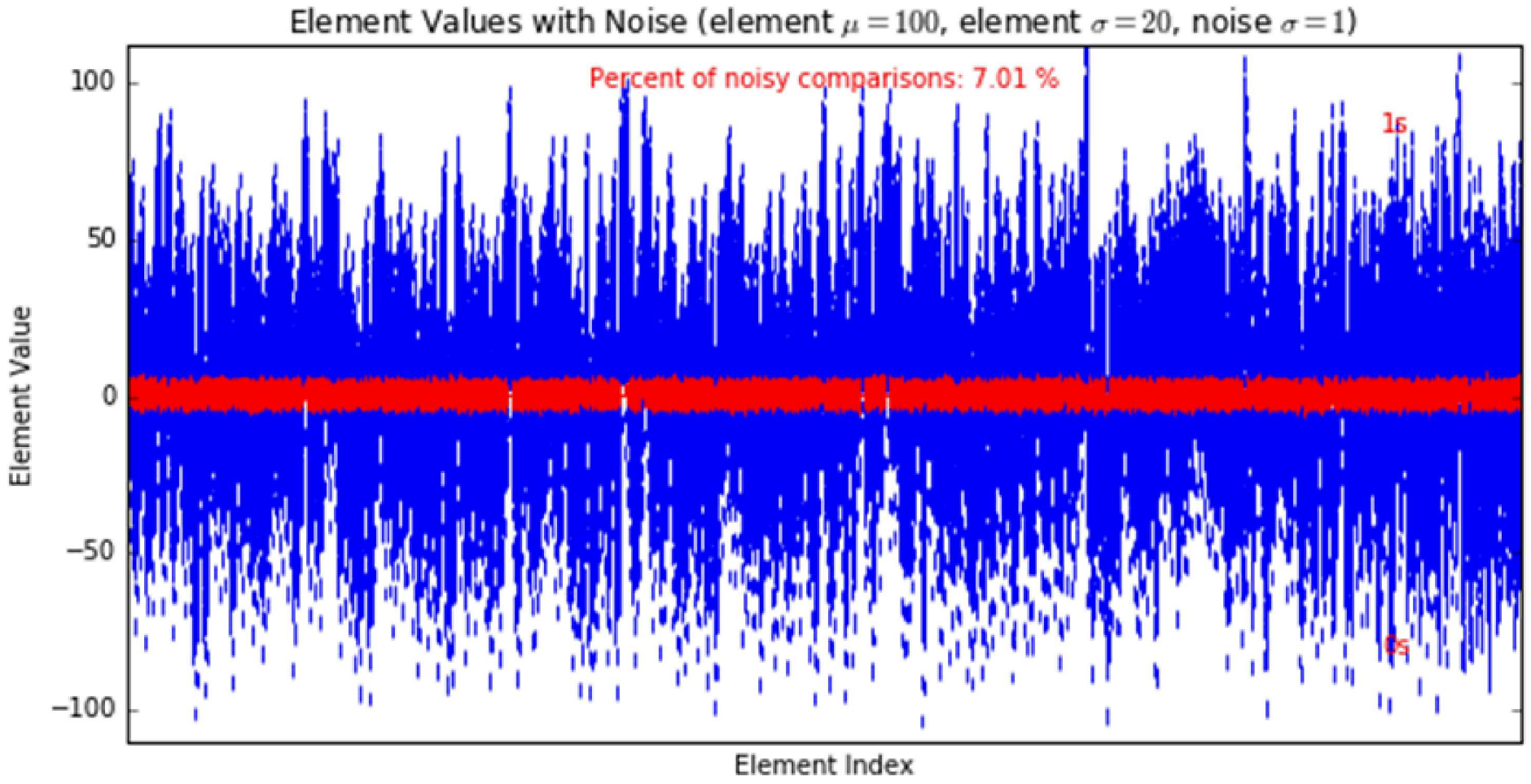
Comparing elements against each other is a similar problem

- Most values are normally distributed
- Then, most elements are near the mean, and therefore are near each other

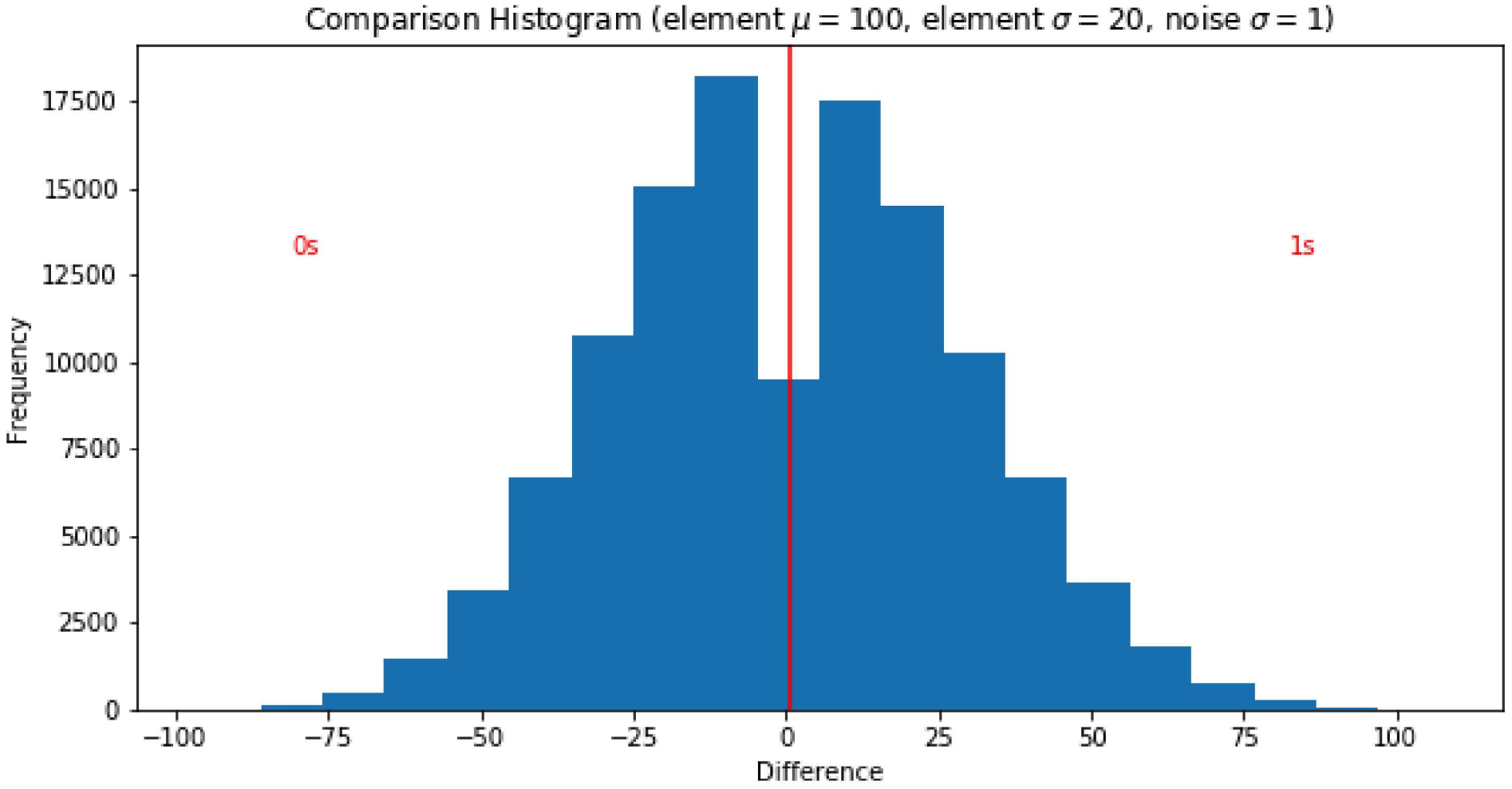
Distribution of $\binom{n}{2}$ comparisons of elements



$\binom{n}{2}$ Comparison Values with Noise for an example PUF



Distribution of $\binom{n}{2}$ comparisons, unstable ones removed

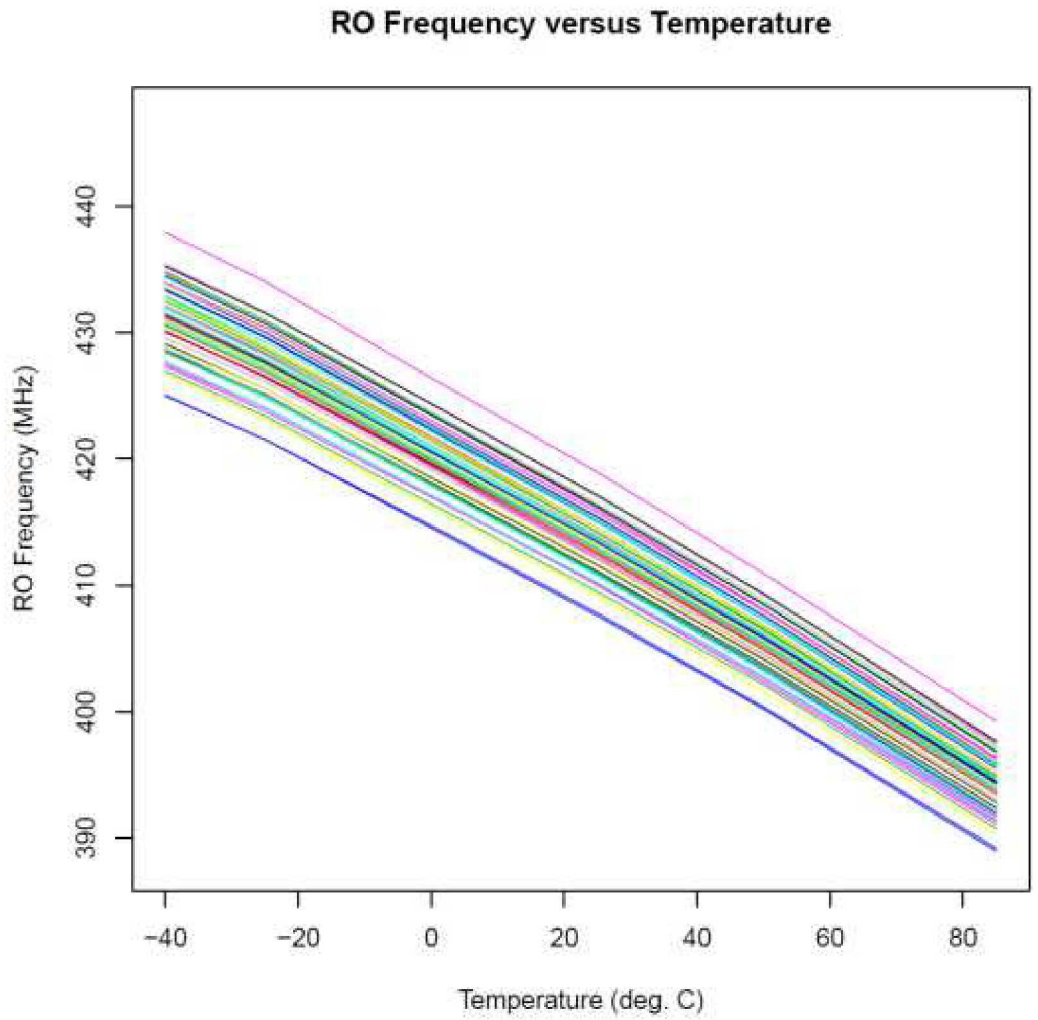


What about environmental variations?

Temperature Effects

Circuit performance tends to decrease as temperature increases

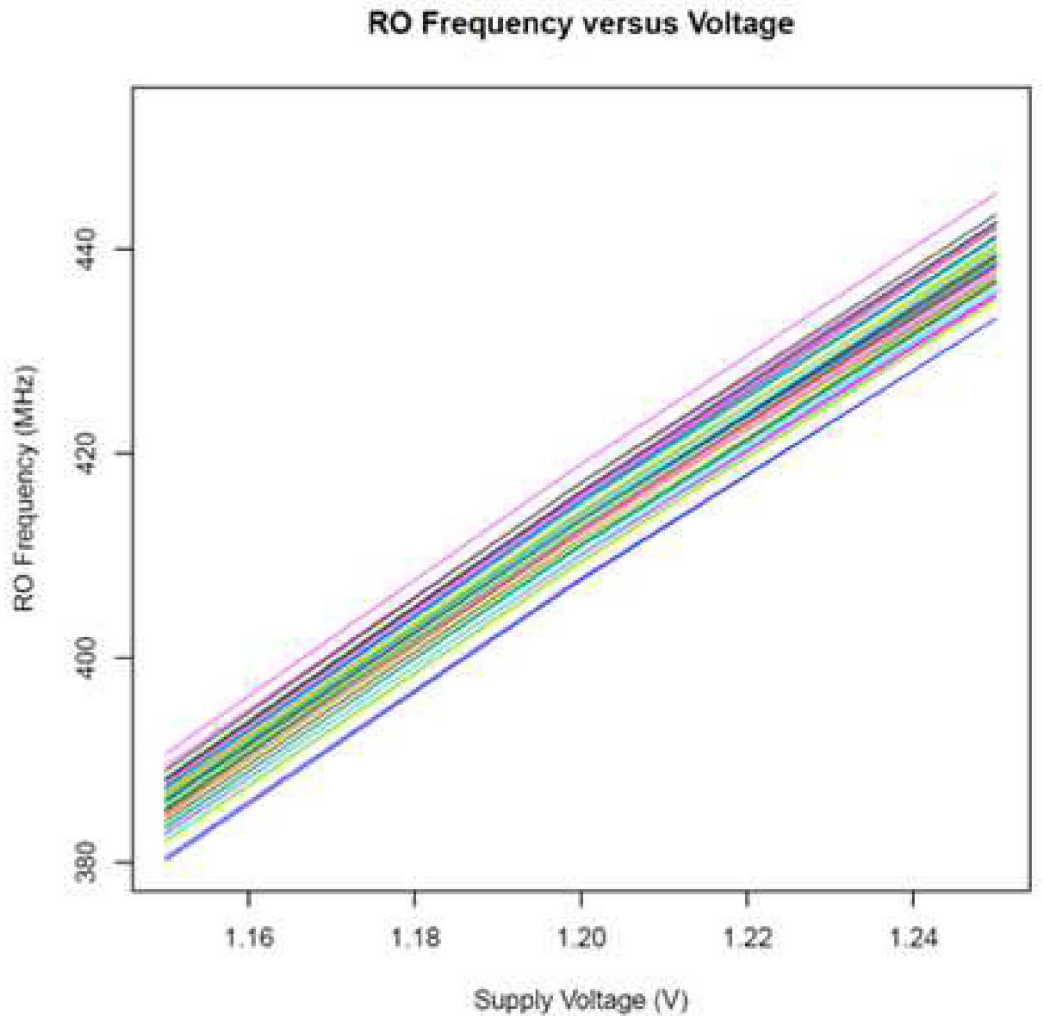
For example, ring oscillator frequency decreases as the temperature increases



Voltage Effects

Circuit performance tends to decrease when the supply voltage is reduced

For example, ring oscillator frequency decreases as the supply voltage decreases



Common-mode Variations

PUF designs usually compare measured values

- to a reference value
- to each other

This has the benefit of cancelling common-mode variation

- As we saw, temperature and voltage tend to have the same effect on all the elements (carrier mobility, threshold voltage)

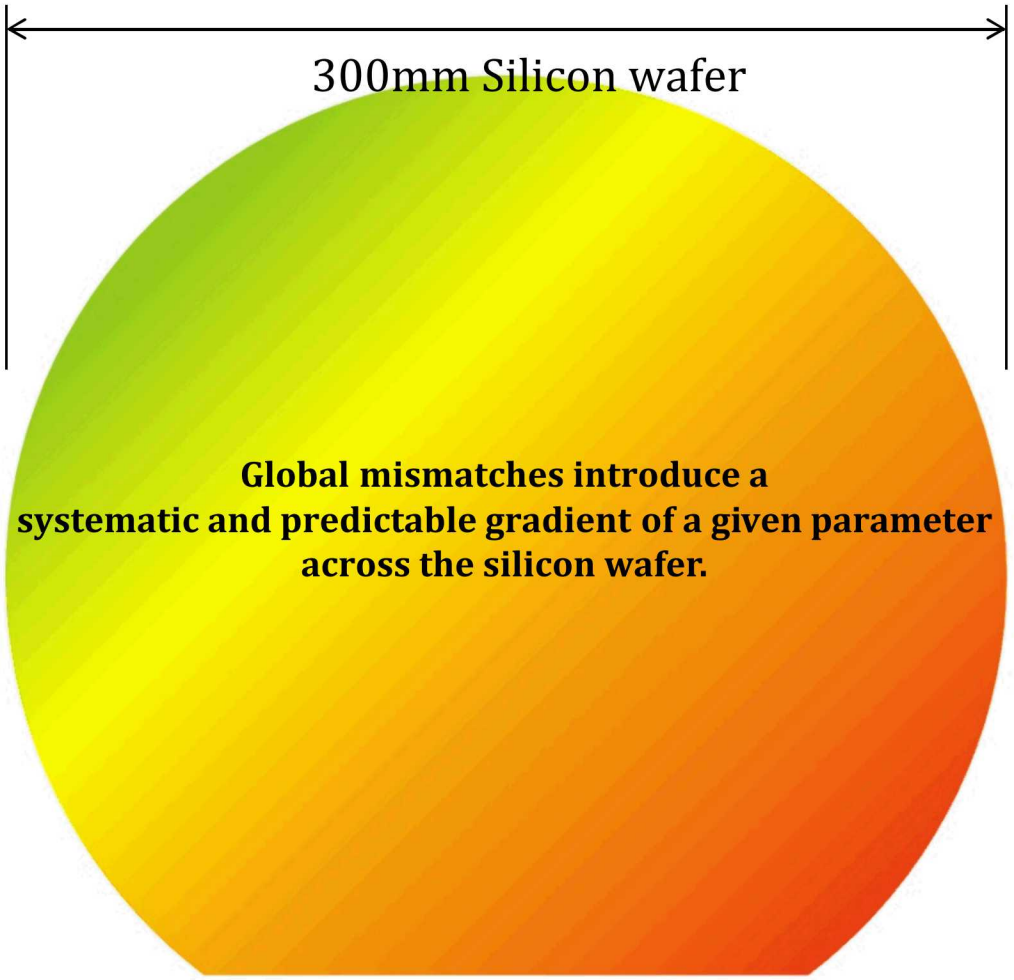
However, there are still *minor* variations in the temperature coefficient of the measured value due to manufacturing variations

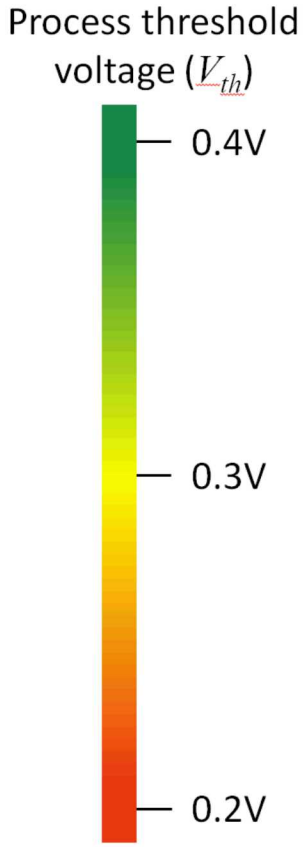
- Occasionally, the relative values will still reverse their order
- Causes additional unstable bits when temperature cannot be controlled

We can model the RO frequency using $F \approx -aT + c$

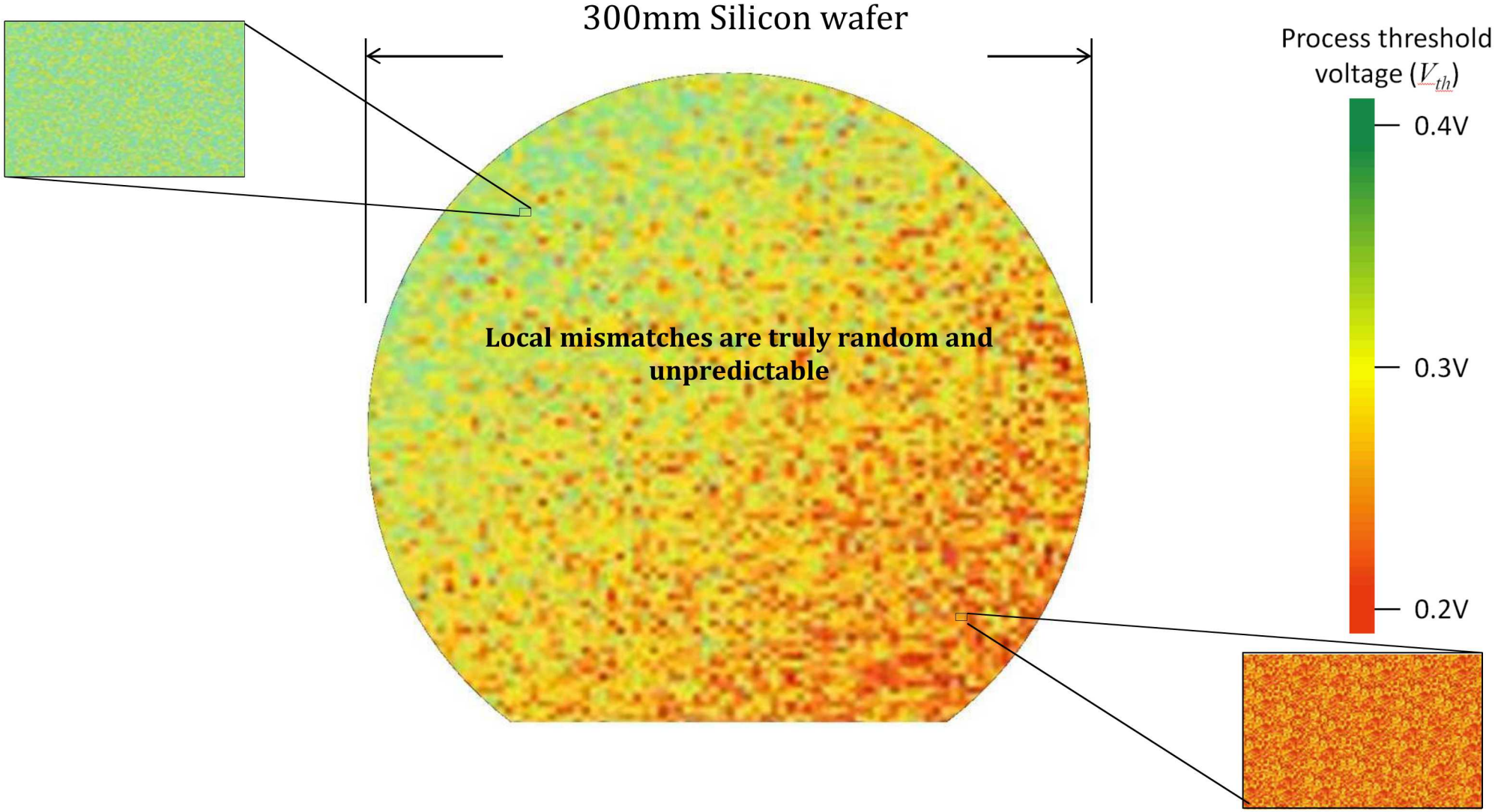
The coefficient of temperature a and the offset c can be broken into average and individual parts: $F \approx -(a_{\text{avg}} + a_{\text{ind}})T + (c_{\text{avg}} + c_{\text{ind}})$ where $a_{\text{ind}} < a_{\text{avg}}$, $c_{\text{ind}} < c_{\text{avg}}$

Influence of global variation and mismatch





Influence of local variation and mismatch



How long do PUFs last?

Accelerated Life Testing

Typically, microelectronic wear-out mechanisms increase with elevated temperature and voltage

NIST/SEMATECH e-Handbook of Statistical Methods

- shows an excellent method for estimating the acceleration factor and the mean time to failure (MTTF)
- as a function of temperature and voltage
- requires **many** chips split into a number of samples

$$t_f = AV^{\beta} \exp\left(\frac{\Delta H}{kT}\right)$$

To completely solve the parameters of this equation, many samples are tested at distinct conditions (see Table on right)

Since failure rates are lower at lower stress levels, *even more* samples are needed at these conditions

The "backwards L" design is commonly used for acceleration modeling experiments covering the full range of both stresses, but still hopefully having enough acceleration to produce failures

Example accelerated life testing conditions and number of samples tested at each condition

VDD \ Temp	85 °C	105 °C	125 °C
6V	10,000	1,000	**30**
8V	1,000	100	**30**
12V	**30**	**30**	**30**

Alternative MTTF Approximation based on [Seymour 1993] and [Vigrass 2010]

If we assume a common failure mechanism, we can compute the acceleration factor (AF) using

- $\text{AF} = \exp\left(\frac{E_a}{k}\left(\frac{1}{T\text{use}} - \frac{1}{T\text{stress}}\right)\right)$, E_a is the thermal activation energy (in eV), k is Boltzmann's constant
- E_a can be assumed to be 1.0 eV

If there are zero failures during the test,

- $\lambda = \frac{M}{\text{TDH}\times\text{AF}}$ [failures per hour]
- TDH := Total Device Hours = number of units × hours under stress
- $M := \frac{\chi^2_{\alpha,2r+2}}{2}$
- χ^2 := the Chi square factor (or probability) for $2r + 2$ degrees of freedom
- r := the total number of failures
- α := the complement of the confidence level, e.g., 0.05

Then, we can compute MTTF using

- $\text{MTTF} = \frac{1}{\lambda} = \frac{2\times\text{TDH}\times\text{AF}}{\chi^2_{\alpha,2r+2}}$ [hours]

Citations

References

Gassend, Blaise, et al. "Silicon physical random functions." *Proceedings of the 9th ACM conference on Computer and communications security*. 2002.

Dodis, Yevgeniy, Leonid Reyzin, and Adam Smith. "Fuzzy extractors: How to generate strong keys from biometrics and other noisy data." *International conference on the theory and applications of cryptographic techniques*. Springer, Berlin, Heidelberg, 2004.

R. Helinski, D. Acharyya and J. Plusquellic, "A physical unclonable function defined using power distribution system equivalent resistance variations," *2009 46th ACM/IEEE Design Automation Conference*, San Francisco, CA, 2009, pp. 676-681.

NIST Engineering Statistics Handbook, Accelerated life tests
https://www.itl.nist.gov/div898/handbook/apr/section3/apr314.htm

Bob Seymour. Application note ab-059: Mttf, failrate, reliability and life testing. 1993.

William J Vigrass. Calculation of semiconductor failure rates.Harris Semiconductor, 2010