Infineon

CHES 2005 - Edinburgh

Design of Testable Random Bit Generators

M. Bucci, R. Luzzi
{marco.bucci, raimondo.luzzi}@infineon.com

Infineon technologies

Never stop thinking.

Summary

- Effective entropy of a real RBG device
- Security assumptions, entropy redundancy and compression
- Meaningfulness and limits of statistical tests
- Suppressing pseudo-randomness
- Implementation and implications of pseudo-randomness suppression: testing online and in “certification” mode
- Detecting and contrasting forcing attacks

Do we need testability?
We must design for testability!

Someone would like to test an HW system without schematics and test points?

Someone else would like to test a SW without source code, logs and (possibly) a debugger?

?

On these deterministic systems you will have low probability to find bugs :o)

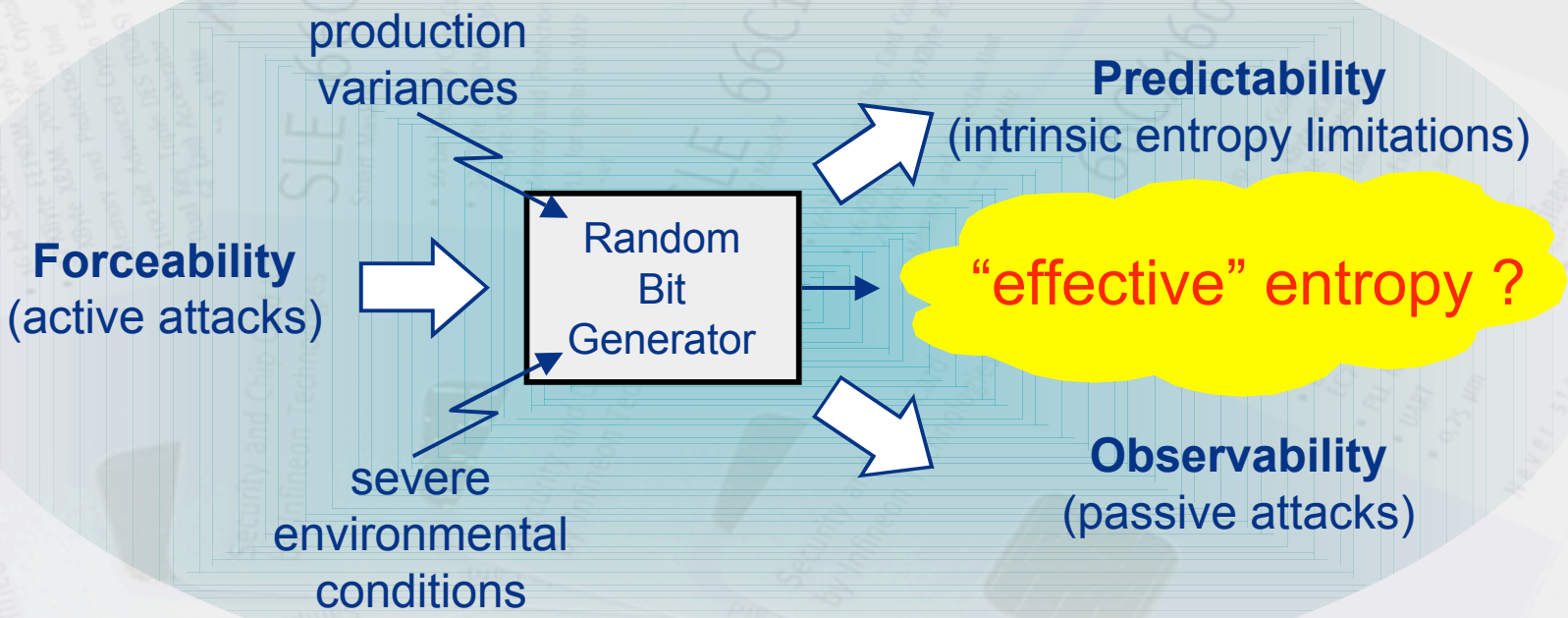
We can not test even a deterministic system as a black box. Why it should work with RBG's?
If we want to find bugs and defects, we must design our system to be testable.

Lets try to make RBG's testable!

!

Effective entropy and robustness of a real RBG device

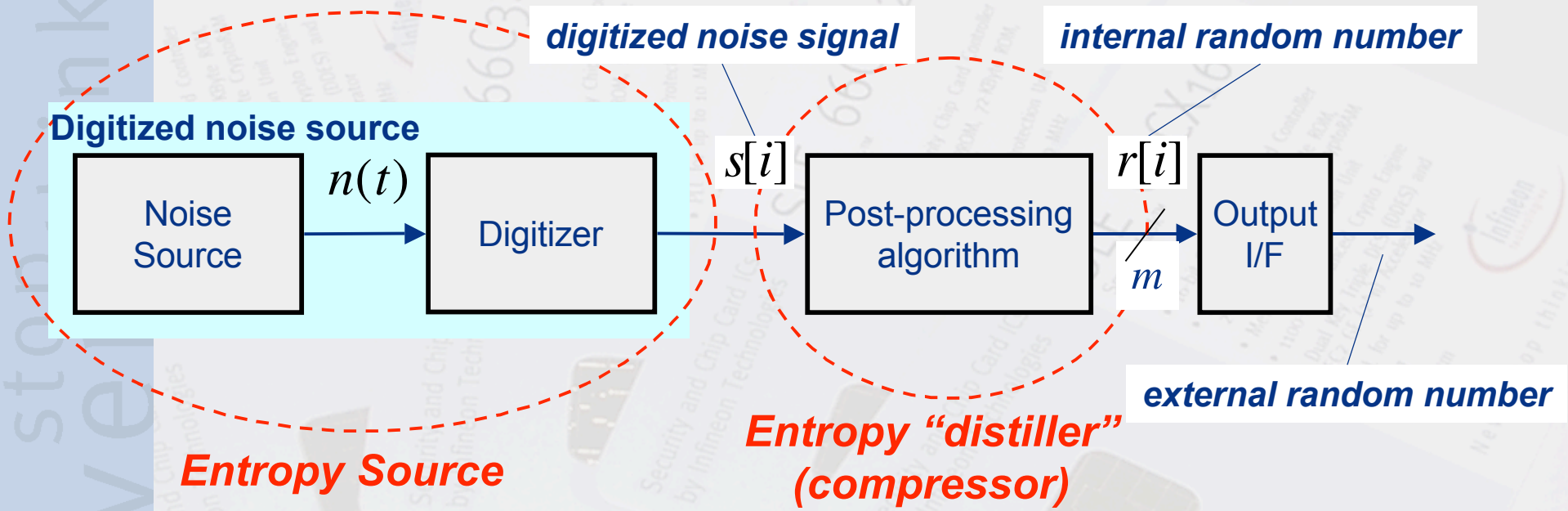
Effective entropy of an RBG results from its intrinsic entropy, but also from its possible **observability** and **forceability**.



Robustness with respect of production variances and severe environmental conditions must be also guaranteed.

Entropy redundancy and compression

Entropy redundancy and compression are always needed to guarantee correct operation under severe conditions.

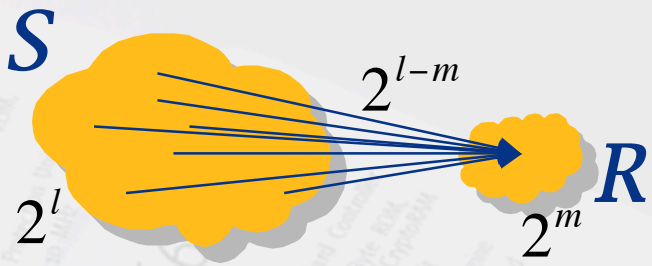


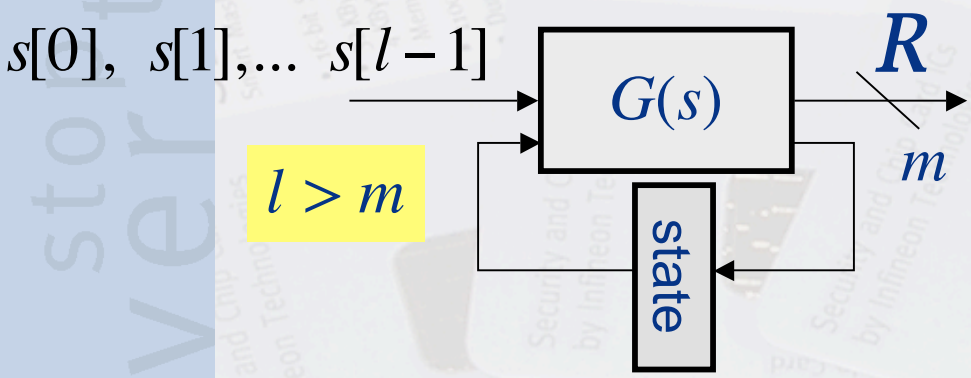
Generic architecture of an RBG (following AIS 31 definitions)

A suitable post-processing compression is the only way to increase the effective entropy of the output.

How post-processing should operates? Can we prove how much is the output entropy?

Post-processing should compress (hashing) the input in order to "flat" (to average) the probability of output symbols.



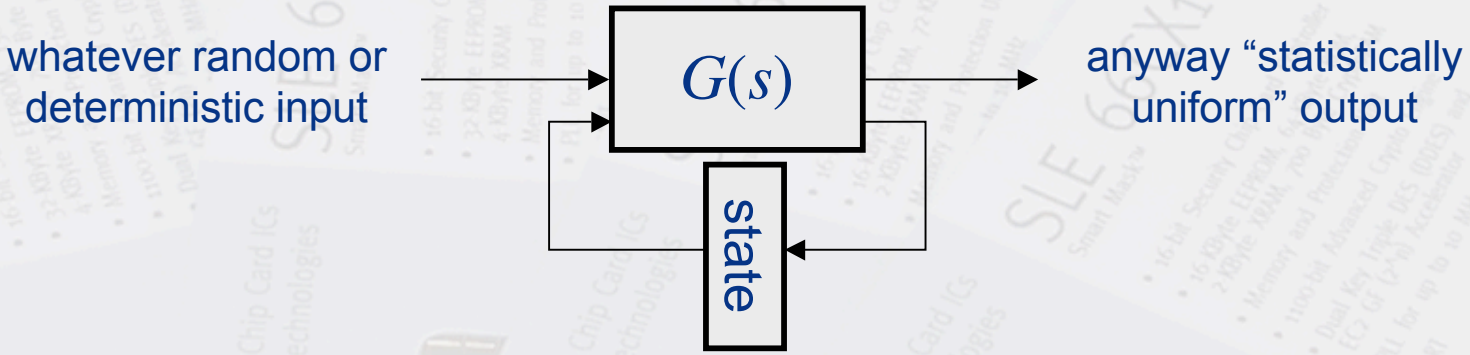


In practical implementations, post-processing is a sequential machine making use of a cryptographic function.

Empirically very powerful, but it is practically impossible to prove how much is the entropy/bit on the output.

Can we (at least) test entropy after post-processing?

In case of lack of source entropy, the post-processor, by construction, acts as a pseudo-random generator: the output will be "statistically uniform" whatever the input from the source is!

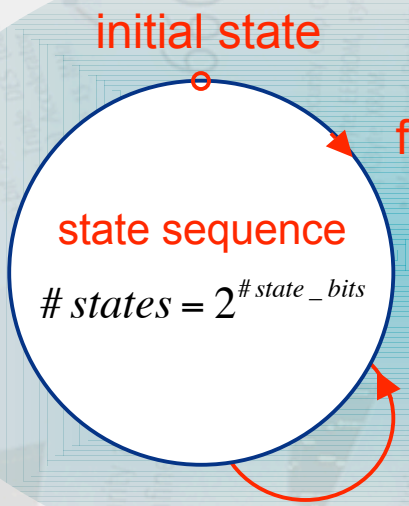


As a consequence, **it seems that, after post-processing, it is impossible to test the amount of entropy even empirically**.

REMARK: the same problem arises on the digitized noise source (i.e. even without compression) if it is intrinsically pseudo-random.

Pseudo-randomness hides the actual amount of entropy! Let suppress pseudo-randomness!

A pseudo-random generator is basically a system whose free state evolution (actually a loop) “looks” random.

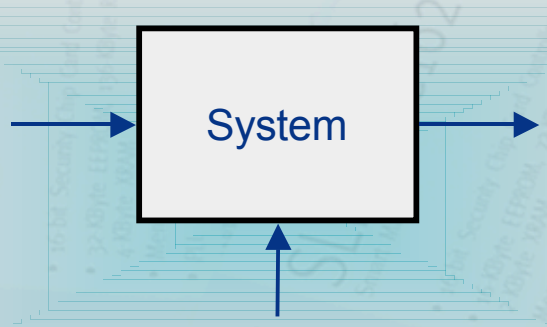


In case an input is supplied (e.g. a compressor), the state is forced to “jump”.

FACT: If the state space degenerates to a point, no pseudo-randomness is possible: **a stateless system cannot behave pseudo-randomly**.

What does “stateless” mean?

A stateless system is, in facts, a system without memory.



Whatever system behaves as stateless if its memory is cleared (i.e. if it is restarted from the same state before generating a new output).

REMARK: the “starting state” must not necessarily be the “zero-state”.

What does “stateless” imply?

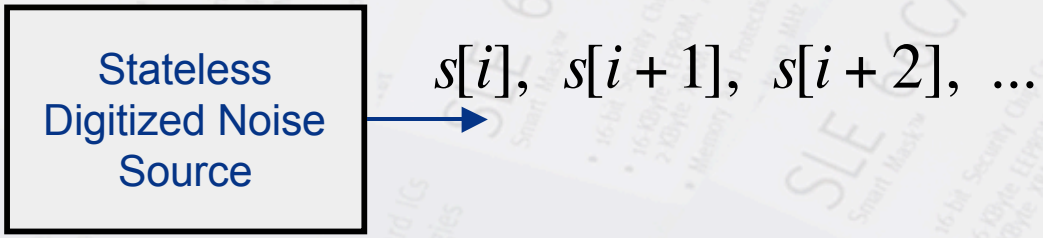
A stateless RBG can produce just independent symbols.

Test complexity depends only on the output cardinality.

Now statistical tests can be simple and significant: a flat histogram really means maximal entropy.

What does “stateless” imply for a digitized noise source?

- Output bits are independent.
- Since the output cardinality is just 2, the source can be easily tested “on the fly” (**online test**).

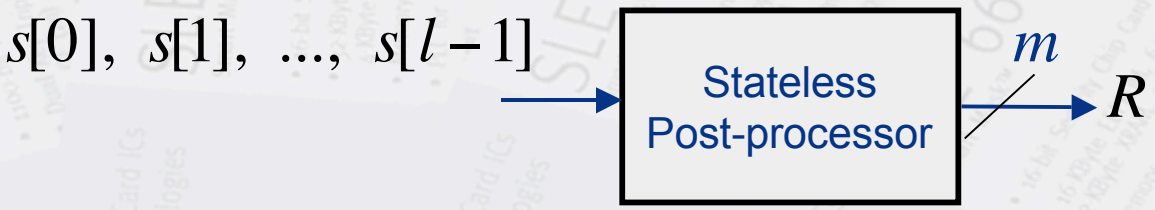


$$N_{trans} = \sum s[i] \oplus s[i-1]$$

Since there is no state evolution, each transition represents an “unexpected” (i.e. a random) event.
A simple transition test is enough: no entropy will result in no transitions.

What does “stateless” imply for the compressor?

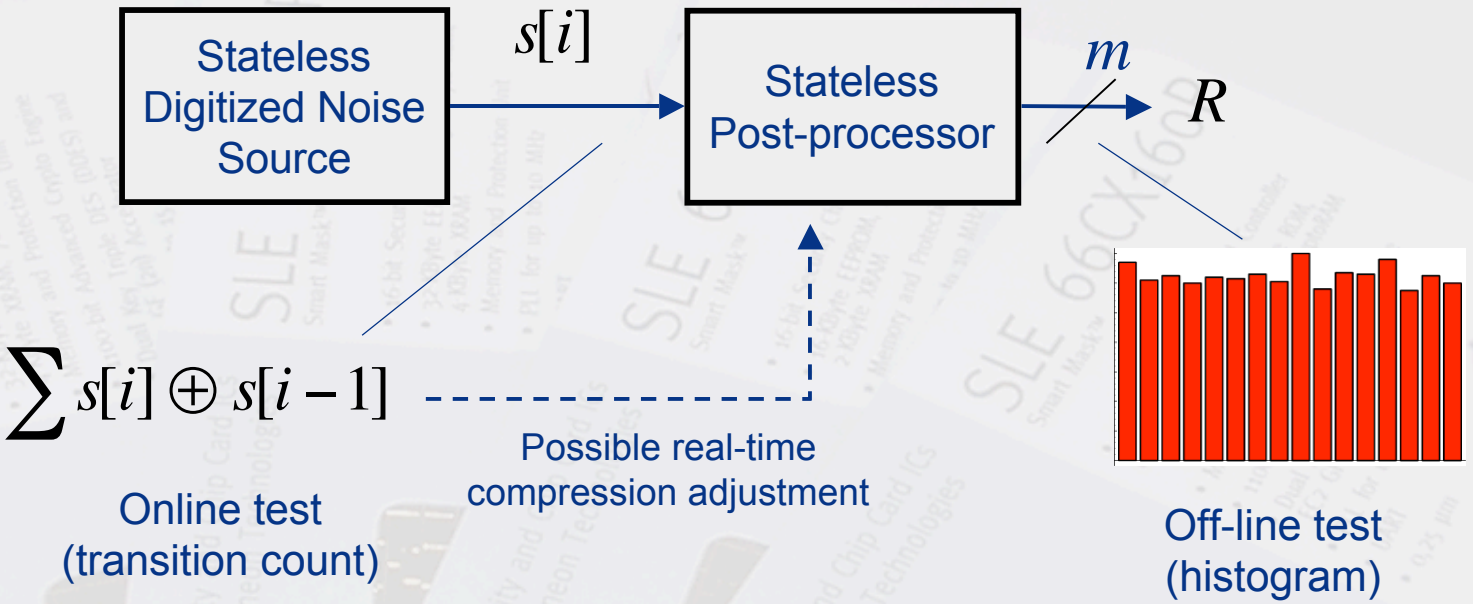
The post-processor actually behaves as a **combinatorial function** (i.e. same input results in the same output).



Therefore, if output symbols R are generated from independent strings $\{s[0], s[1], \ldots, s[l\text{-}1]\}$ then they are independent too.

REMARK: independence amongst $s[i]$ symbols of the same input string is not needed.

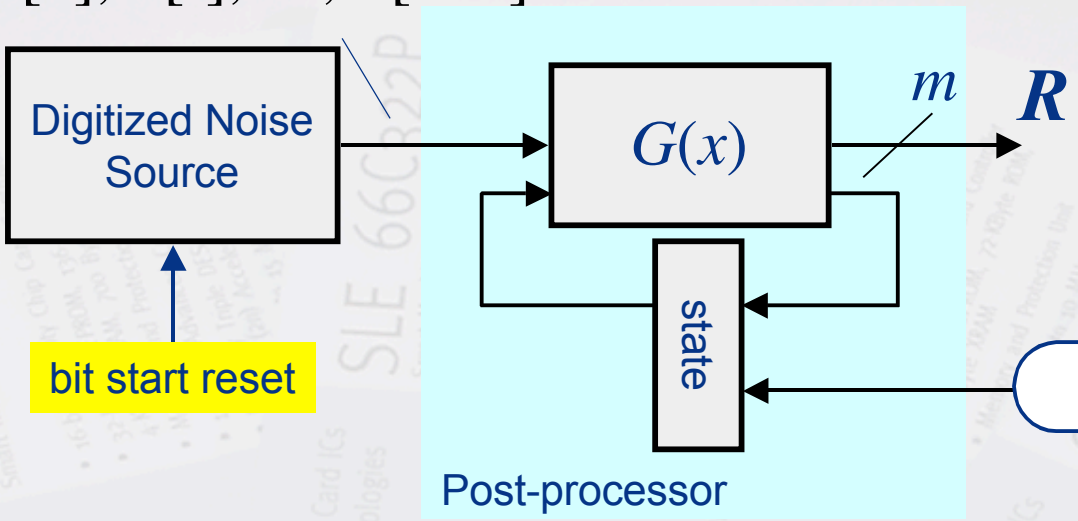
On-line and off-line (e.g. certification) testing



Notice: since it is extremely short, the online test can be used to adjust i**n real time** the compression ratio. In this way, **each** delivered output symbol is guaranteed to have the requested entropy.

Implementation (example)

$s[0],\ s[1],\ \ldots,\ s[l-1]$



Example:

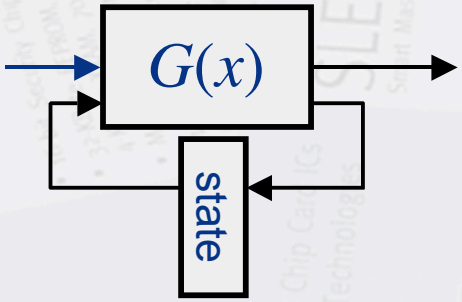
$l = 32;$
$m = 8 \Rightarrow |R| = 256$

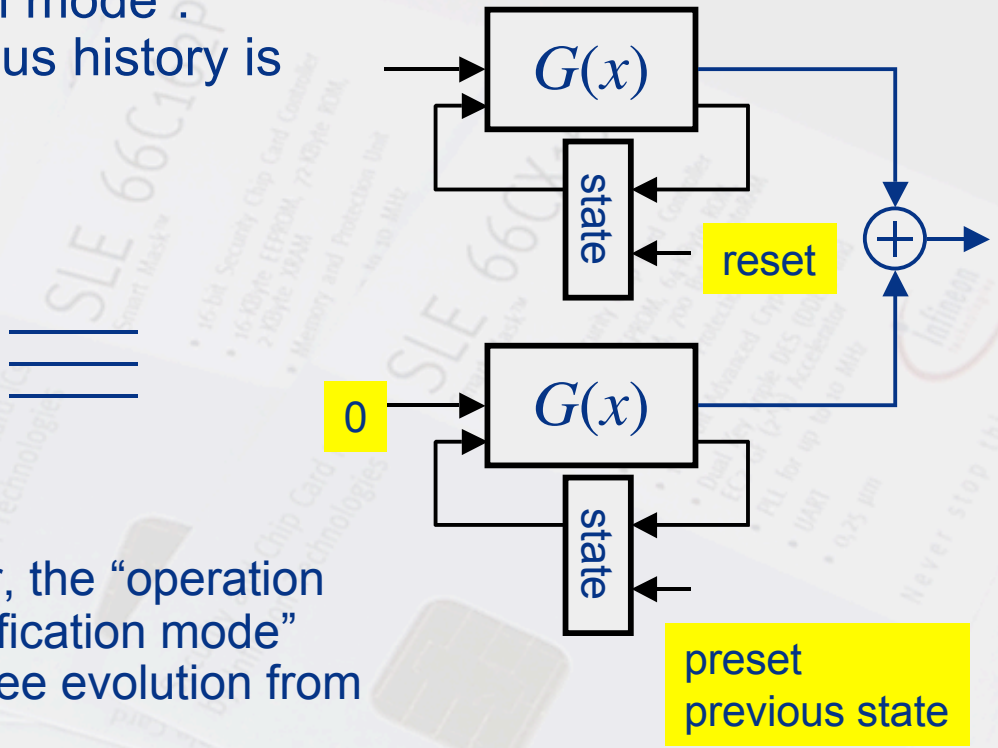
- In principle this schema is applicable to every kind of source and compressor.
- Only requirement is a fast recovery from the reset (starting) state.

REMARK: if the source test is not performed, also the source can be reset just at word start.

How secure is the normal “operation mode” with respect of the “certification mode”?

The “operation mode” is very likely more secure than the “certification mode”: residual entropy from previous history is not lost.



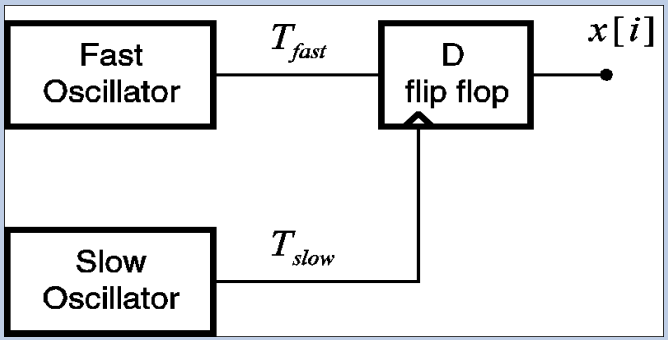


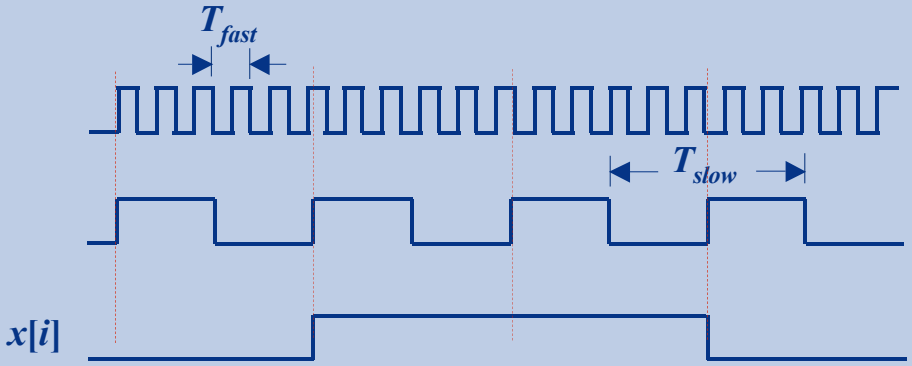
In case the compressor is linear, the “operation mode” is equivalent to the “certification mode” xored with the compressor in free evolution from the previous state.

Therefore the operation mode is proven to be not less secure than the certification one

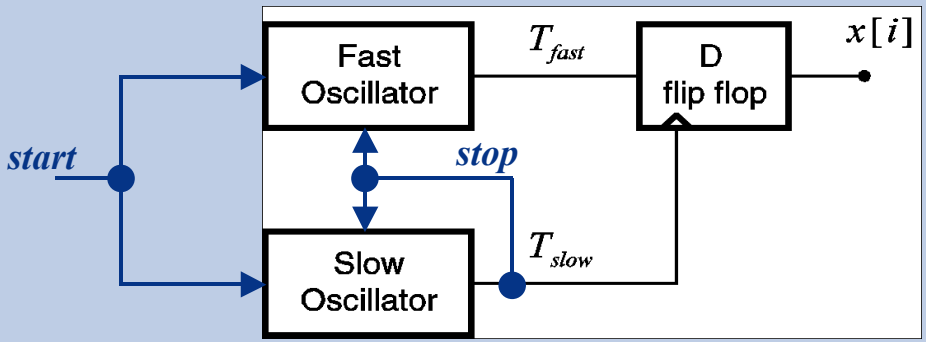
Stateless digitized noise source example:

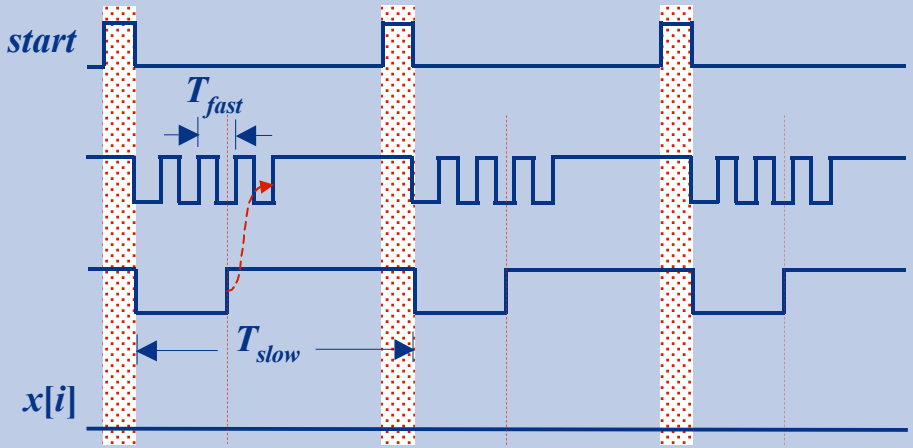
"An offset-compensated oscillator-based random bit source for security applications" (CHES 2004)





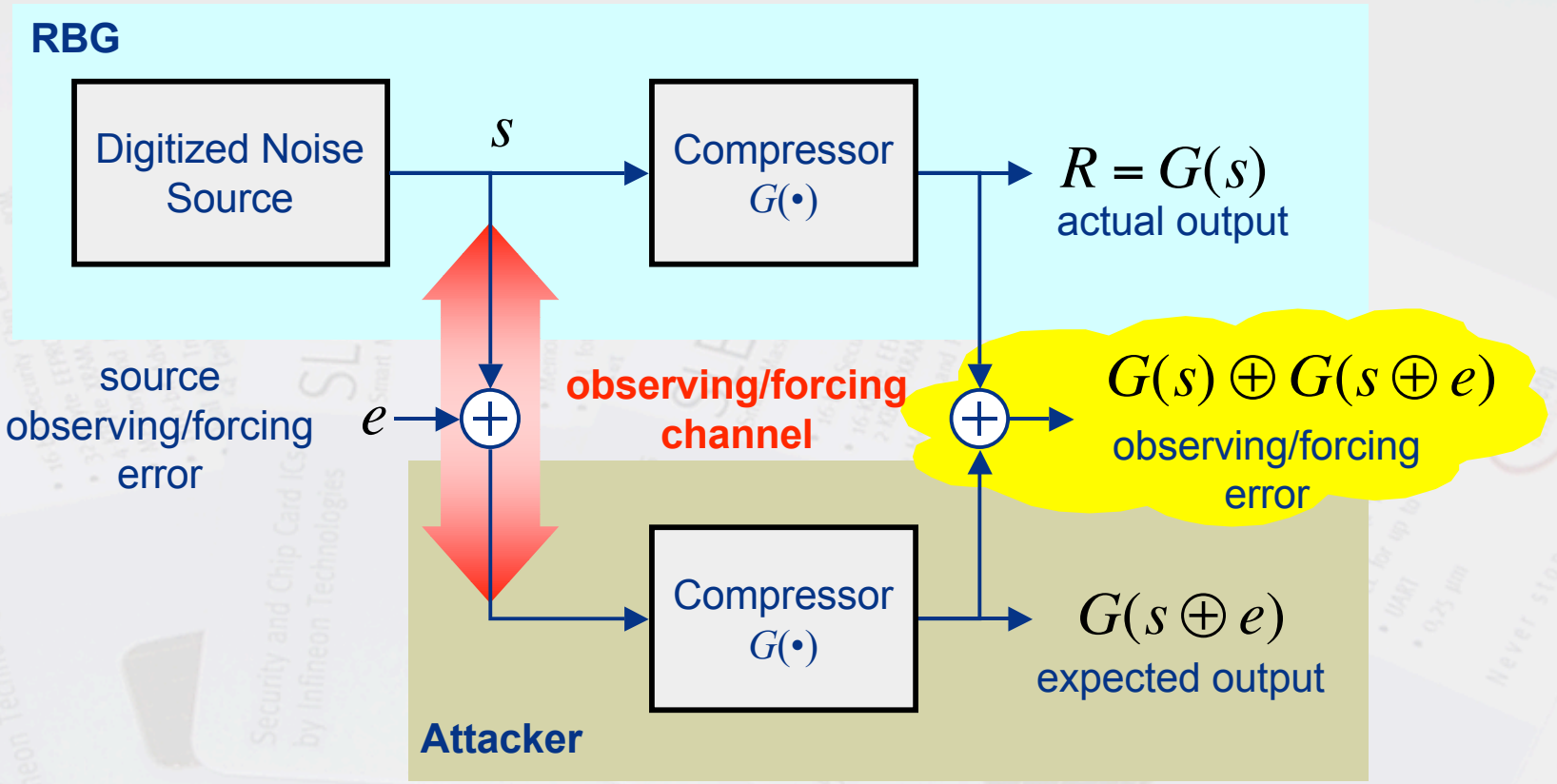
"classic"





stateless

A simplified model for a RBG under observing/forcing attack



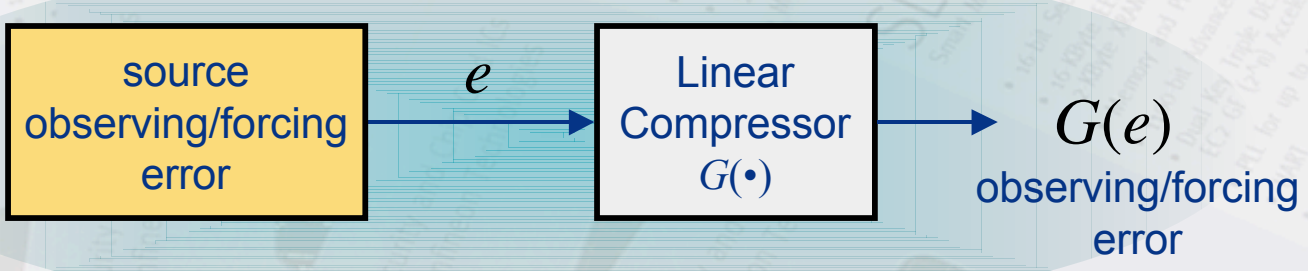
We consider attacks focused on the source which is intrinsically the most vulnerable part. The robustness of the compressor can be approached by the techniques used for digital devices.

Effective entropy of an RBG under observing/forcing attack (case of a linear compression function)

In case the compressor function is linear with respect of the x-or operation:

$$G(s) \oplus G(s \oplus e) = G(e)$$

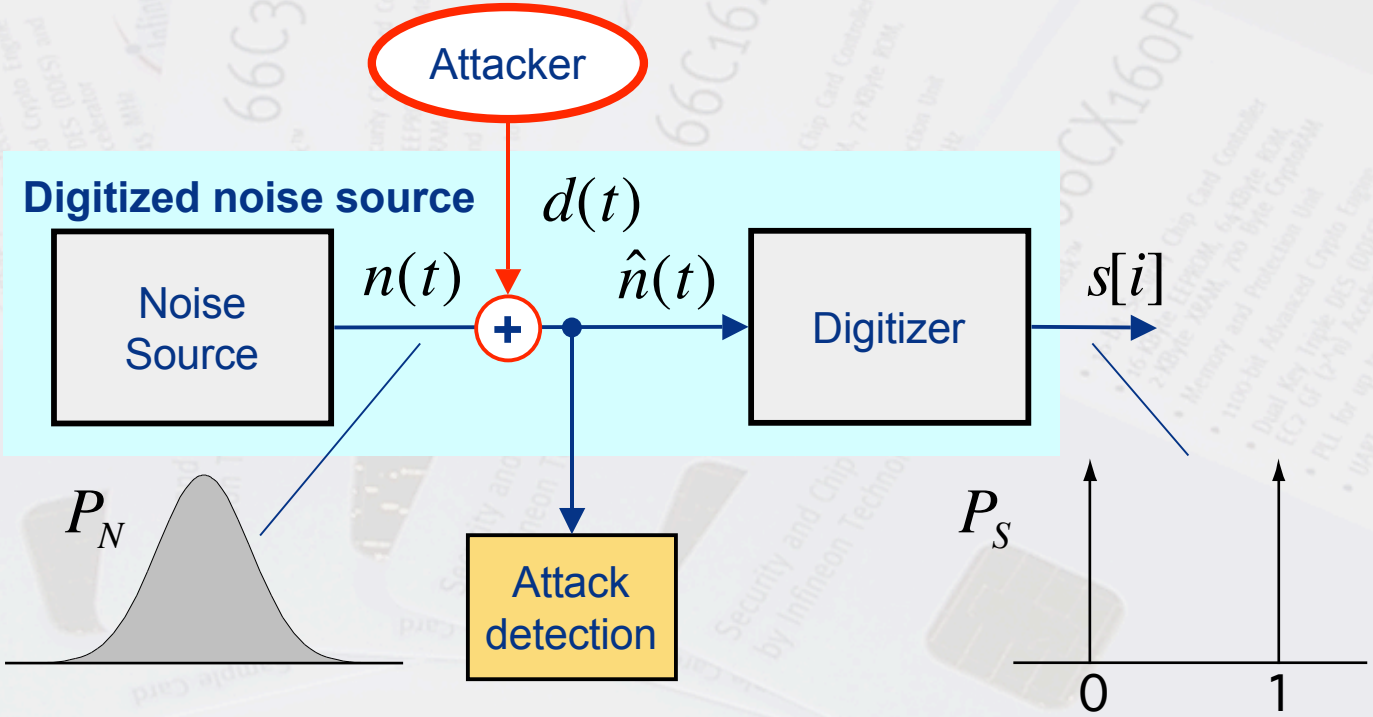
In other words, **the attacker "sees" an RBG having a "virtual source" consisting in his error in observing/forcing the real source**.



Assuming this model, the compression factor must be fixed considering a source consisting in the (estimated) residual error of the attacker.

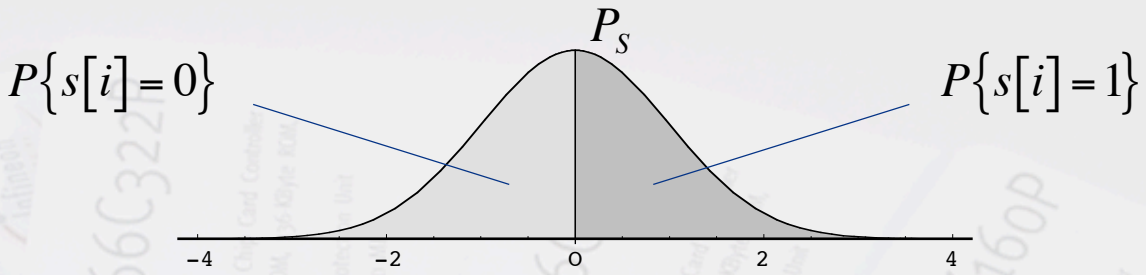
Forcing attacks: scenario

An attacker can try to superimpose an its own “random”, signal $d(\bullet)$ to the entropy source $n(\bullet)$.

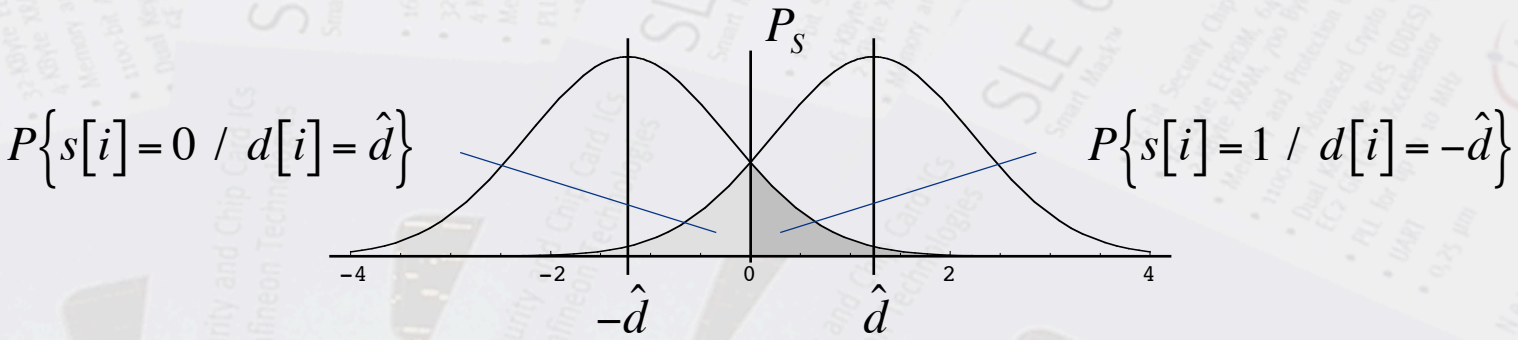


REMARK: No way to detect the attack after quantization (i.e. no statistic anomalies if $d(\bullet)$ is random or pseudo-random).

Source probability distribution: under normal condition and under attack



Source distribution under normal conditions



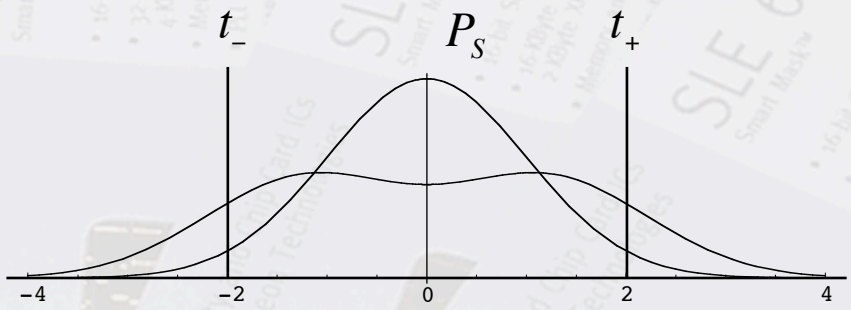
Conditional source distribution under attack and residual error of the attacker

REMARK: in order to actually reduce the entropy, the attacker needs to obtain a small residual error. Hence he needs to inject a signal $d(\bullet)$ having a large amplitude.

Detection of forcing attacks

An attacker cannot force the source without increasing its intensity.

Therefore, a forcing attack can be easily detected testing the source intensity (assuming its typical value is known).



As an example, a simple statistical test can consist in counting the number of samples beyond two prefixed thresholds t_- and t_+.

Conclusions

- The entropy of a stateless RBG can be easily tested after post-processing.

Therefore:

- The main merit figure of an entropy source is its entropy/second (entropy throughput) and not anymore its statistical quality or entropy/bit.
- The design of the entropy source is not anymore constrained by its “quality” and can be more focused on its actual robustness with respect of:
  - Attacks
  - Faults
  - Production defects