Intel Digital Random Number Generator SP800-90B Non-Proprietary Public Use Document

Intel Entropy Source

May 2023

Revision 0.3.4

Intel Confidential

Document Number: 618769

Notice: This document contains information on products in the design phase of development. The information here is subject to change without notice. Do not finalize a design with this information.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software, or service activation. Learn more at intel.com, or from the OEM or retailer.

No computer system can be absolutely secure. Intel does not assume any liability for lost or stolen data or systems or any damages resulting from such losses.

You may not use or facilitate the use of this document in connection with any infringement or other legal analysis concerning Intel products described herein. You agree to grant Intel a non-exclusive, royalty-free license to any patent claim thereafter drafted which includes subject matter disclosed herein.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document. The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest Intel product specifications and roadmaps.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit http://www.intel.com/performance.

Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.

Results have been estimated or simulated using internal Intel analysis or architecture simulation or modeling and provided to you for informational purposes. Any differences in your system hardware, software or configuration may affect your actual performance.

Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

Intel is a sponsor and member of the Benchmark XPRT Development Community and was the major developer of the XPRT family of benchmarks. Principled Technologies is the publisher of the XPRT family of benchmarks. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.

Intel, the Intel logo, Pentium, and Xeon are trademarks of Intel Corporation or its subsidiaries..

*Other names and brands may be claimed as the property of others

Contents

Figures

Tables

Revision History

Revision Number	Description	Date
0.3.4	• Updates related to Federal Information Processing Standard (FIPS) 140-3 IG	May 2023
0.3.3	• Title update, CHIP name update.	April 2023
0.3.2	• Initial release of the document.	March 2023

1 Description

The Intel Entropy Source is a physical (P) entropy source.

The circuit component is RNG_ES_BDW_1.0

The synthesizable RTL component is BDW-COOP-RTL1.0

All components of the entropy source are hardware, consisting of logical and electronic components bounded in a rectangle of silicon. There is no firmware or software within the entropy source.

The entropic data from the entropy source is Non-IID.

This PUD is applicable to the following products:

Table 1-1. List of Applicable Devices

Processor Name	Processor Name
Pentium® Processor D1507	Intel® Xeon® Processor D-1528
Pentium® Processor D1508	Intel® Xeon® Processor D-1529
Pentium® Processor D1509	Intel® Xeon® Processor D-1530
Pentium® Processor D1517	Intel® Xeon® Processor D-1531
Pentium® Processor D1519	Intel® Xeon® Processor D-1533
Intel® Xeon® Processor D-1518	Intel® Xeon® Processor D-1537
Intel® Xeon® Processor D-1520	Intel® Xeon® Processor D-1539
Intel® Xeon® Processor D-1521	Intel® Xeon® Processor D-1540
Intel® Xeon® Processor D-1526	Intel® Xeon® Processor D-1541
Intel® Xeon® Processor D-1527	Intel® Xeon® Processor D-1548

2 Security Boundary

The entropy source security boundary surrounds the set of components referred to as the Digital Random Number Generator (DRNG) core, that includes the noise source, digitizer, Continuous Health Tests (CHT), AES-CBC-MAC Vetted Conditioning Component, Deterministic Random Bit Generator (DRBG), Non-deterministic Random Bit Generator (NRBG) and register interface.

For use as a full entropy source, the NRBG output of the DRNG is the necessary output. This is accessed through the Intel CPU instruction "RdSeed".

The security boundary is a sub boundary within the DRNG. Components outside the security boundary but within the DRNG are to attach to the local bus, clock and power systems on the chip.

The DRNG security boundary is not a FIPS140 security boundary. It is designed to be usable within a larger FIPS140 security boundary through compliance to SP800-90A, B and draft C along with relevant requirements in ISO/IEC 19790-2012 and FIPS140-3.

The bold outline in Figure 2-1 shows the security boundary of the entropy source and other RNG components.

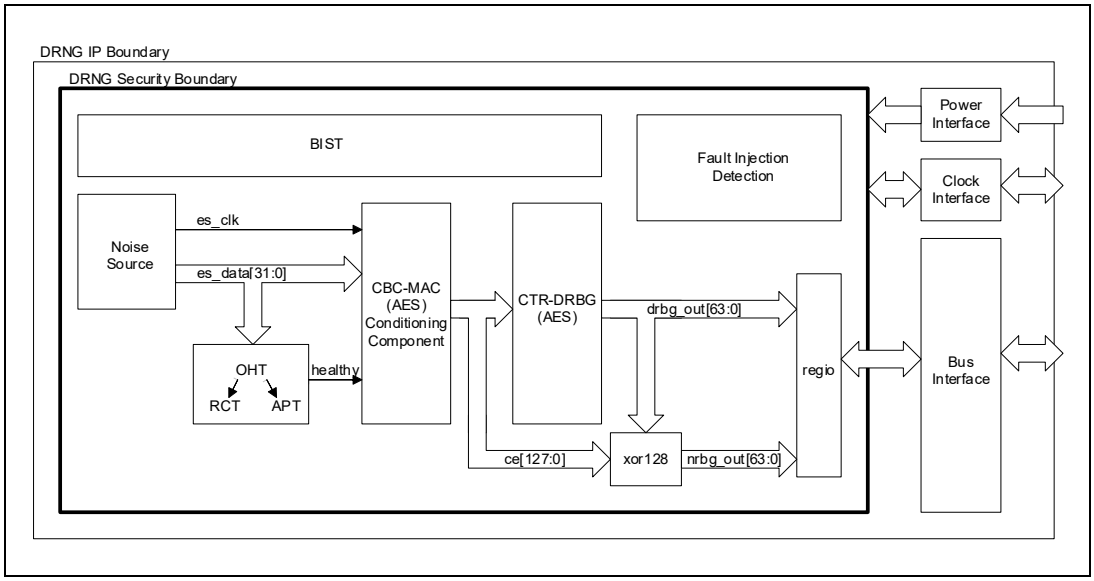
The NRBG output path to the region block is the output of the full entropy source, as an RBG3 construction.

The DRBG output path to the region block is the output of the DRBG, as an RBG2 construction.

The NRBG construction is the XOR (RBG3) construction NRBG from Draft SP800-90C. This XORs the output of the conditioner with an output from the DRBG to form the NRBG output. The strength of the XOR is entirely dependent on the full entropy of the conditioner output. The DRBG's input into the XOR, as well as the XOR operation itself are not considered part of the conditioning chain.

The width of these operations is 128 bits, driven by the output size of Advanced Encryption Standard (AES). The registers sizes are 64 bits and in the logic, this width transforming is performed using a FIFO that is 64 bits wide and takes in 128 bits as two 64-bit entries and outputs 64 bits to match the register width.

Figure 2-1. DRBG Security Boundary



In the context of an Intel CPU, the DRNG register that presents the NRBG output to the local bus is called `egetdata`. This register is read by the CPU whenever the `RdSeed` instruction is executed. The result is passed to the target register of the instruction and the success or failure signaled in the carry flag.

This report is applicable to the devices in Table 1-1. The silicon manufacturing process, IC design and DRNG design is identical for each of these devices. Differences between these devices are the set of enabled features, which has no influence on the RNG behavior.

3 Operating Conditions

The entropy source is guaranteed to operate within the designated operating envelope in the data sheet of the chip. In **Intel® Xeon® Processor D-1520** the operating envelope is as follows:

Table 3-1. Operating Conditions

Parameter	Minimum	Maximum
Temperature	0°C	80°C

These conditions are taken from the Intel Automated Relational Knowledge-Base (ARK) specifications:
https://www.intel.com/content/www/us/en/products/sku/87038/intel-xeon-processor-d1520-6m-cache-2-20-ghz/specifications.html

4 Configuration Settings

There are no configuration settings accessible at any privilege level to software running in the CPU.

5 Physical Security Mechanisms

The RNG contains a security boundary described in Section 2. In the operating mode of the RNG, there is hardware enforcement of the security of the boundary.

Specifically:

- Diagnostic output of raw data from the boundary is disabled.
- Debugging port access to internal state of the RNG is disabled at the boundary.
- Configuration input written to registers is ignored and the default configuration is enforced.
- Algorithms are implemented to detect stuck output failures.
- Register write privileges are enforced in hardware.

When an alarm is triggered, the RNG resets itself and re-runs BIST. It is not possible to distinguish between a failure due to attack or environmental bounds violation or a rare false positive error. The re-running of BIST will lead to the RNG failing if the BIST fails. In the case of a transitory error, the RNG will recover when BIST is re-run.

The packaging of the chip is a tamper evident enclosure.

6 Conceptual Interfaces

For consuming applications running in software on the CPU, `GetEntropy(n)` is implemented by the `RdSeed` instruction, where *n* can be one of 16, 32 or 64, depending on the size of the target register.

Internally to the security boundary, for feeding conditioner output data to the DRBG, `Get_entropy()` is implemented in digital logic between the conditioner and DRBG in the form of a 128-bit wide FIFO.

7 Min Entropy Rate

The assessed entropy from the noise source, as per Section 3.1.4.2 of the SP800-90B, is *min(Hr, Hc, HI) = 0.6 bits* of entropy per bit of data.

H_{out} from the CBC-MAC conditioning component, assessed according to Section 3.1.5 of the SP800-90B, is *1-ε where ε << 2-32*, meeting the draft requirement for full entropy in draft SP800-90C.

The assessment of the conditioner output entropy is computed with the lower bound of the conditioner input data set above the CHT pass threshold of 0.6, the conditioner entropy output assessed using the equations in SP800-90B, Section 3.1.5.1.2, are:

```
H_out = Output_Entropy(n_in, n_out, nw, h_in) where
 n_in   = 512
 n_out  = 128
 nw     = 128
 h_in = 0.6 * 512 = 307.2
```

When using a conditioning component listed in Section 3.1.5.1.1 (given the assurance of correct implementation by CAVP testing), the entropy of the output is estimated as:

$$h_{out} = Output\_Entropy(n_{in}, n_{out}, nw, h_{in})$$

Where *Output_Entropy(n_{in},n_{out},nw,h_{in})* is described as follows:

1. Let $P_{high} = 2^{-h_{in}}$ and $P_{low} = \frac{(1-P_{high})}{2^{n_{in}-1}}$.
2. N = min(n_{out},nw).
3. $\Psi = 2^{n_{in-n}} P_{low} + P_{high}$
4. $U = 2^{n_{in-n}} + \sqrt{2n(2^{n_{in}-n})In(2)}$
5. $\Omega = U \times P_{low}$
6. Return – $\log_2(\max(\psi,\omega))$

This was evaluated using 1,000 binary digit floating point arithmetic.

```
#!/usr/bin/env python3

from __future__ import print_function

import UsrIntel.R1 # Intel specific import to get the standard libraries
import math
from mpmath import *

PRECISION = 1000 # 1,000  of precision.
```

```
def output_entropy(n_in,n_out,nw, h_in):
    n = min(n_out,nw)
    ttninmn = mpf(2.0)**(n_in-n)
    p_high = mpf(2.0)**(-h_in)
    p_low = (mpf(1.0)-p_high)/((mpf(2.0)**n_in)-mpf(1.0))
    phi = ttninmn*p_low + p_high
    U = ttninmn + sqrt(mpf(2.0) * n * ttninmn * log(mpf(2.0)))
    w = U * p_low
    return -log(max(phi,w),2)

if __name__=="__main__":
    mp.prec = PRECISION
    entropy_rate = 0.6
    n_in = 512
    h_in = n_in*entropy_rate

    entropy = output_entropy(mpf(n_in), mpf(128.0), mpf(128.0), mpf(h_in))
    epsilon = mpf(128.0)-entropy

    # compute human readable epsilon
    for i in range(1,1024):
        power = mpf(i)
        ne = epsilon*(2**power)
        if ne > 1:
            print("epsilon = %s*2^-%d" % (nstr(ne,8),i))
            break

> ./pud_entropy.py
epsilon = 1.255939*2^-179
```

The result shows the lower bound of epsilon for entropy rates above the pass threshold of the CHT is, to four digits of precision $1.255*2^{-179}$, which meets the requirement for epsilon the be less than 2^{-32}.

8 Health Tests

The DRNG includes:

- Continuous Health Tests (CHT).
- Startup Noise Source Health Tests.
- Startup Logic Integrity BIST.

The CHT is composed of a short-term test yielding a pass/fail over individual 256-bit blocks of noise source data, and a long-term evaluation of the pass/fail history of the past 256 blocks (totaling 65536 bits) to infer an entropy source failure. The failure condition is invoked when the pass/fail rate drops below 50%.

The logic integrity test is one of the startup tests. This performs a test of the digital logic by running deterministic random sequences through all the logic and testing the resulting output against the expected result.

The Startup Noise Source Health Test involves running the CHT for a probationary period of 65536 bits from the noise source. When the test is complete, assuming the test passes, the RNG enters the operational state. This happens during CPU startup and completes before the first CPU instructions can execute. Thus, `RdSeed` is available from initial instruction execution time.

The startup test is invoked at power-on or exiting the reset state. Thus, the startup tests can be invoked by power cycling the CPU or resetting the CPU.

A failure any of the startup tests (logic integrity or noise source health test) will be reflected as a BIST failure in the internal status register which leads to an MCHECK failure of the CPU.

Following the startup tests, the CHT continue to run. Should failure condition of the CHT test be encountered after the startup tests have completes, this may be either the result of a soft error or a hard error. The test will continue to run and should the entropy quality return, it will exit the failure state. In the failure state, no more random numbers are issued, and the failure state is reflected in the internal state registers, which is visible at the instruction interface by the repeated failure to deliver random numbers, through the returned carry flag of the `RdRand` and `RdSeed` instructions being 0.

A sequence of 1000 back-to-back 0 carry flags on `RdSeed` can be inferred to be an error. Should some condition exist (for example, out of specification operating condition), cessation of that condition will lead to resumption of the supply of random numbers.

A less computationally expensive test of failure is the response from the `RdRand` instruction (not a noise source output, but an RBG2 output), which due to its higher output rate has a 0% chance of underflows arising from high consumption demand by other threads, and so 10 back-to-back failures are specified as a runtime error.

9 Maintenance

There are no maintenance action requirements.

10 Required Testing

We performed raw noise testing using the non-IID lower bound entropy tests of the SP800-90B Entropy Assessment software tool, showing the entropy from the noise source to exceed the minimum input threshold of 0.299 of the vetted conditioning components.

We performed restart testing using the SP800-90B Entropy Assessment software tool, showing the startup min entropy of *Hr* and *Hc* to be greater than 50% of *HI*.