Intel's Digital Random Number Generator (DRNG)

July, 2011

George Cox, Charles Dike, and DJ Johnston

Legal Disclaimer

- INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL® PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER, AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL® PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT. INTEL PRODUCTS ARE NOT INTENDED FOR USE IN MEDICAL, LIFE SAVING, OR LIFE SUSTAINING APPLICATIONS.

- Intel may make changes to specifications and product descriptions at any time, without notice.

- All products, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice.

- Intel, processors, chipsets, and desktop boards 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.

- Nehalem, Merom, Wolfdale, Harpertown, Tylersburg, Penryn, Westmere, Sandy Bridge and other code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees and other third parties are not authorized by Intel to use code names in advertising, promotion or marketing of any product or services and any such use of Intel's internal code names is at the sole risk of the user

- Performance tests and ratings are measured using specific computer systems and/or components and reflect the approximate performance of Intel products as measured by those tests. Any difference in system hardware or software design or configuration may affect actual performance.

- Intel, Intel Inside, Intel Core, Intel Xeon, Intel Core2, Intel Xeon and the Intel logo are trademarks of Intel Corporation in the United States and other countries.

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

- Copyright © 2011 Intel Corporation

- "Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products. For more information go to http://www.intel.com/performance"

(intel)

DRNG Outline

- DRNG Goals

- DRNG Reusable Block Overview

- DRNG Pipeline

- DRNG Performance

- DFx

- SW Interface – The RdRand Instruction

- Paucity to Abundance

- Solving the Platform Entropy problem

DRNG Goals

Solve the long standing "platform entropy problem"

By delivering consistently high quality/high performance, fully post processed (e.g., DRBGed) entropy in each Intel silicon product

Thus creating a "common brand promise"

By building a DRNG circuit that is reusable across all Intel process, design, and manufacturing environments and

Embedding it in each family of Intel silicon products in the most effective way for that family

(intel)

DRNG Reusable Block Overview
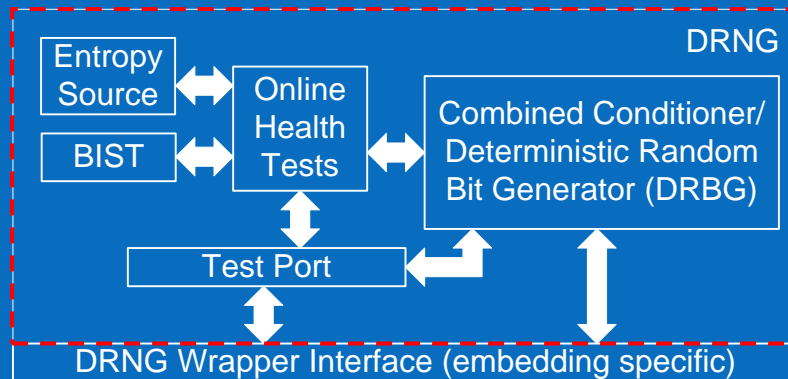
A reusable circuit that provides each embedding with an autonomous/self contained, high quality/high performance, "complete" DRNG

Provides a hardware source of high quality, high performance entropy to be embedded across ALL Intel products. It is composed of

– An all-digital Entropy Source (NRBG), runtime entropy quality measurement via Online Health Test (OHT),

– Conditioning (via AES CBC-MAC mode) and DRBGing (via AES CTR mode) post processing and
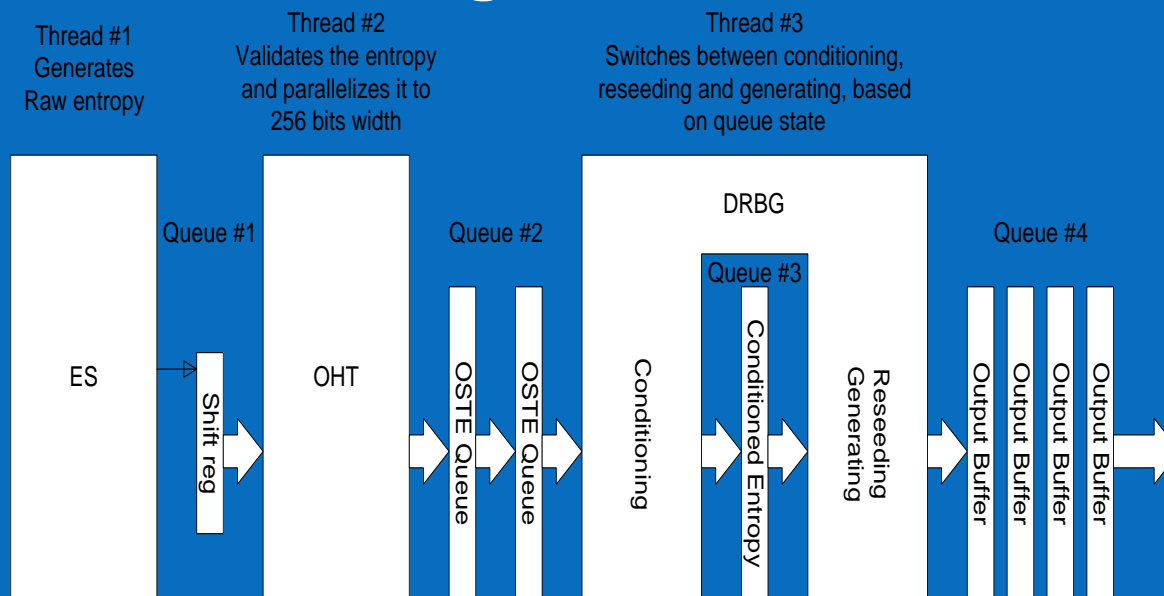
– Built In Self Test (BIST) and Test Port

"Standards" compliant (NIST SP 800-90) and FIPS 140-2/3 Level 2 certifiable as such and

Designed for ease of testability, debug, and validation in HVM and in end user platforms

– Comprehensive Built In Self Test (BIST) and

– Test Port (and associated tools) for full pre/post-silicon debug flexibility

Red dotted line is the DRNG's FIPS boundary

Queue State Driven Pipeline Eliminates External Scheduling

Thread #1
Generates
Raw entropy

Thread #2
Validates the entropy
and parallelizes it to
256 bits width

Thread #3
Switches between conditioning,
reseeding and generating, based
on queue state

DRBG

Queue #1

Queue #2

Queue #3

Queue #4

ES

Shift reg

OHT

OSTE Queue

OSTE Queue

Conditioning

Conditioned Entropy

Reseeding Generating

Output Buffer

Output Buffer

Output Buffer

Output Buffer

Our DRNG is logically a three stage/subunit asynchronous production pipeline (composed of the Entropy Source, Online Health Test, and Conditioner and DRBG)

Each function schedules itself based strictly on the queue state between itself and its neighbors.

The ES runs asynchronously to the downstream logic. Varying demand at the output and asynchronous supply at the input leads to unpredictable internal scheduling, making external prediction of reseed times impossible

(intel)
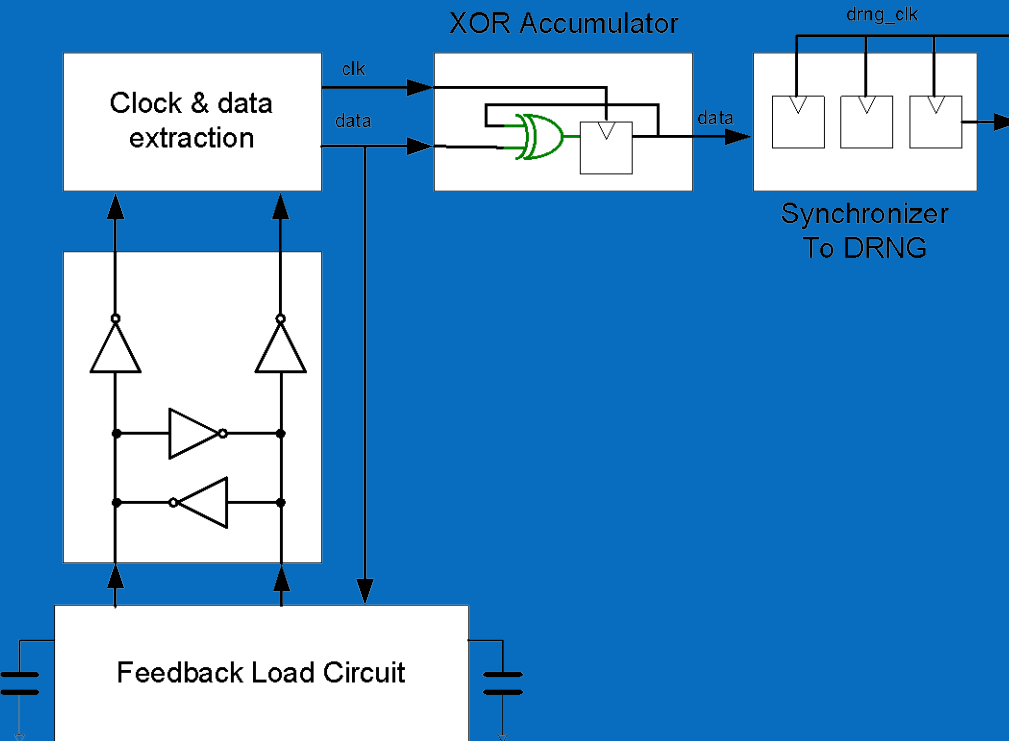
The Entropy Source

The Entropy source is a self timed, digital circuit that pushes a latch into metastability at ~3GHz. The timing of each cycle is determined by the resolution time of the latch. Thus the rate of production is very jittery.

Thermal noise determines the resolution state of each metastable event and this is the fundamental source of entropy in the circuit.

This circuit design enables a reliable, testable entropy source to be built on-chip, using the existing digital power supply network from standard cells while achieving non deterministic output.

Cross process migration is "easy" relative to previous analog elements.
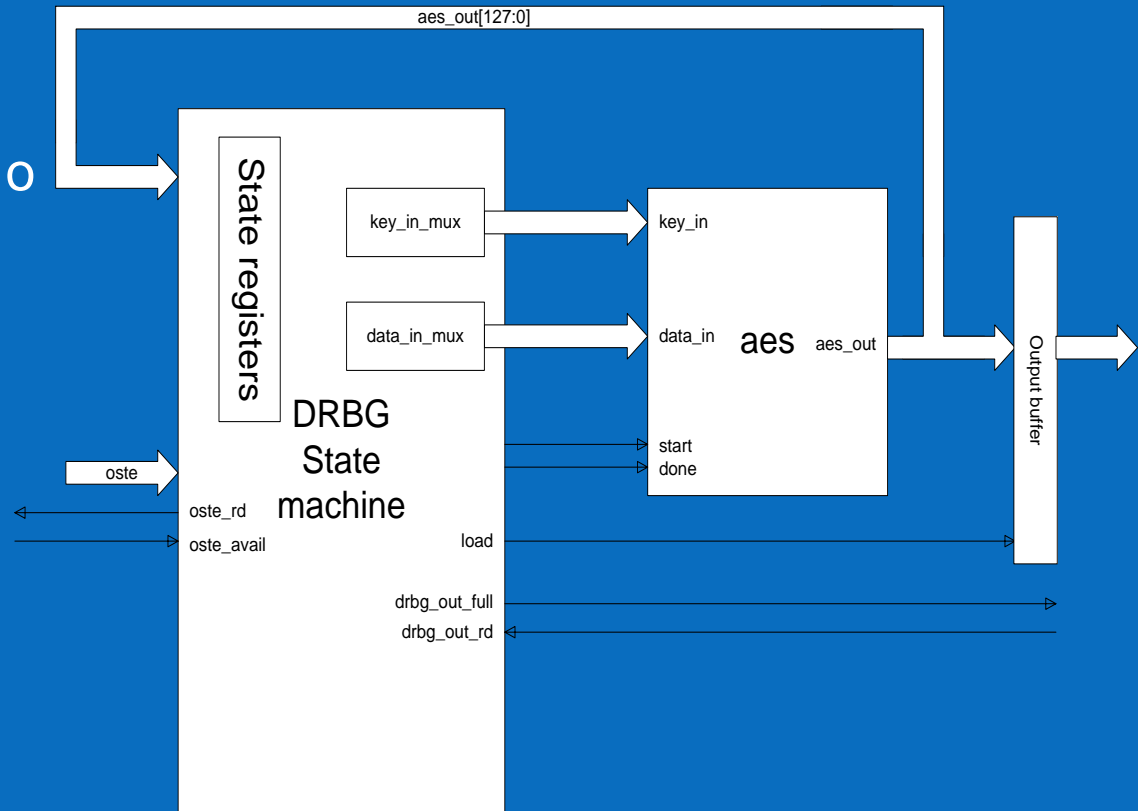
The DRBG

Consists of two pipeline stages

Both use a shared (encrypt only) AES unit to perform

Conditioning/entropy extraction (via AES CBC-MAC mode) and

DRBGing (via AES CTR mode) post processing (NIST SP800-90 compliant and FIPS certifiable as such)

Performance

When our DRNG is embedded in the uncore of a next generation Intel processor on a 22nm process, the performance of each pipeline stage is:

ES

        Self Timed, 2.5 – 3Gbps serial output

        > 80% entropic

DRBG AES-CBC-MAC Conditioner (800Mhz)

        2:1 Compression

        converts 512 input to 256 bits output

DRBG AES-CTR DRBG (800Mhz)

        11 Clocks per 128 bits @ 800Mhz

        Reseeded every 512 bits from the ES

(intel)

DFx

Built In Self Tests (BISTs)

- Comprehensive – ES BIST and KAT BIST

- After every "reset" (as required by FIPS)

- Quick "Pass/Fail" indicator – design through HVM

Online Health Tests (OHTs)

- Dynamic per sample of the ES outputs against expected statistics

- Controls quality and volume of "raw" entropy going into Conditioning

Test Port

- DRNG internal inspection and sequencing mechanism for low level debugging (when not in FIPS mode)

- Proved extremely useful in pre-silicon debugging => largely unneeded in post-silicon debugging => very few post-silicon bugs

SW Interface – The RdRand Instruction

Access to the DRNG is provided to SW through the new RdRand instruction

- Intel Advanced Vector Extensions Programming Reference, Chapter 8 (http://software.intel.com/en-us/avx/)) – pages 8-15 and 8-11 and

- The RdRand Software Implementation Guide (SIG) (http://software.intel.com/en-us/articles/download-the-latest-bull-mountain-software-implementation-guide/);

RdRand retrieves a hardware generated random value from the DRNG and stores it in the destination register given as an argument to the instruction.

RdRand is available to any system or application software running on the platform. That is, there are no hardware ring requirements that restrict access based on process privilege level. As such, RdRand may be invoked as part of an operating system or hypervisor system library, a shared software library, or directly by an application.

(intel)

Entropy Pool Design When Moving from a Paucity to an Abundance of entropy

Existing product Entropy Sources (TRNGs) operate at a rate of a few hundred Kbits/sec to 10-20 Mbits/sec. Platform entropy sources such as keyboard and mouse tend to be much slower.

RNG design in existing operating systems has tended towards large entropy pools to capture all available entropy from slow sources so that it serves as a 'rainy day fund' for when random numbers are consumed (e.g., Linux's* entropy pool is 4K bytes).

The equivalent DRNG's DRBG entropy pool is 256 bits. It is replenished at high rate from a continuous 3Gbps source, so there is no need to buffer larger amounts. Similarly, downstream software consuming or gathering entropy should aim to minimize storage of random numbers and instead use the online supply of fresh random numbers, thus minimizing scope for memory based attacks.

(intel)

Solving The Platform Entropy Problem
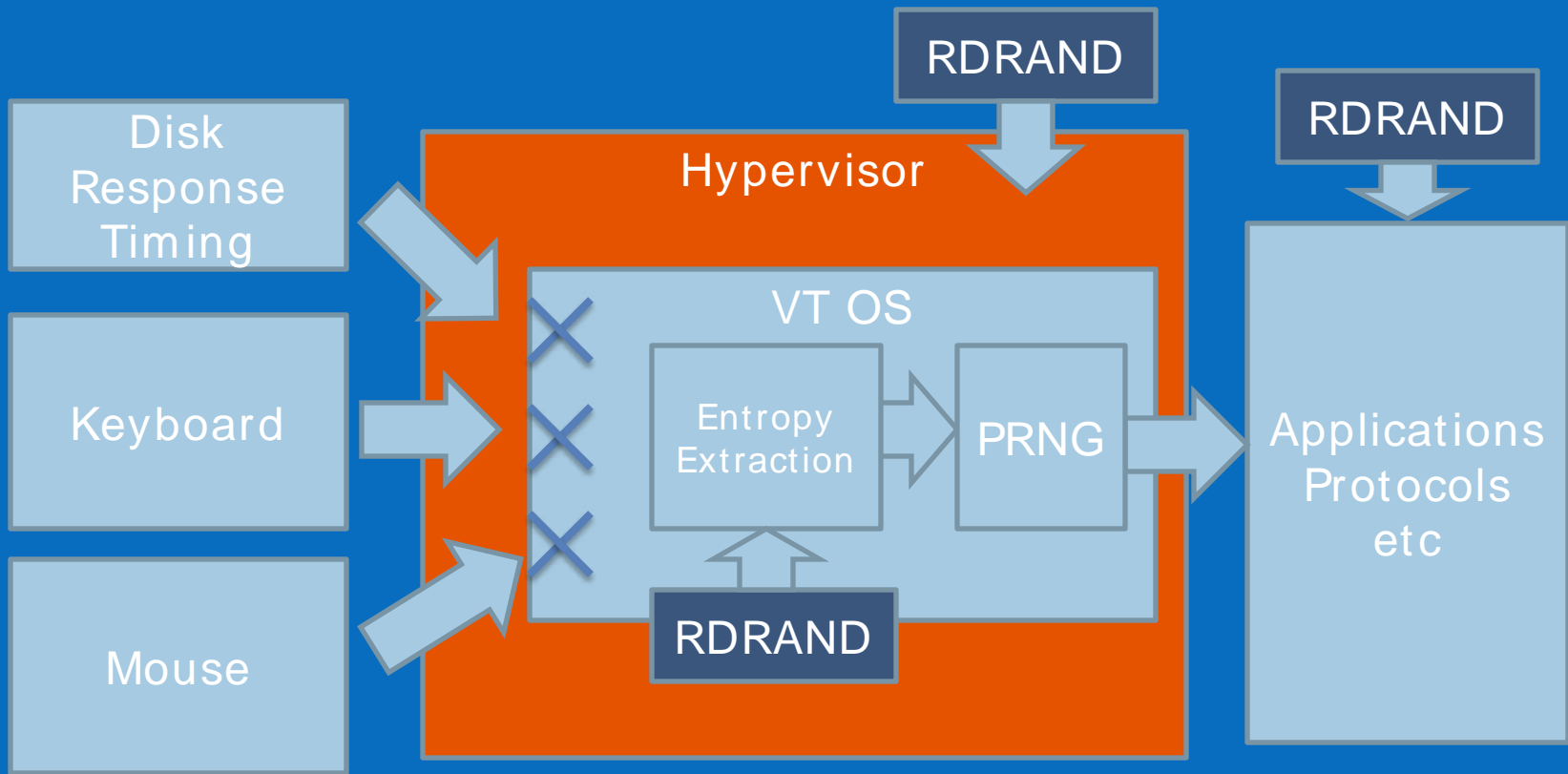
Future VT/Cloud/Server deployments suffer from diminishing entropy sources resulting from virtualization, SSDs, and no local human I/O. RdRand supplies entropy to application, OS and virtual machine/Hypervisor directly from hardware.

(intel)