D-RaNGe

Using Commodity DRAM Devices to Generate True Random Numbers with Low Latency and High Throughput

HPCA 2019

presented by David Schumacher

Jeremie S. Kim Minesh Patel Hasan Hassan Lois Orosa

Onur Mutlu

Outline

Background

Motivation

Goal

Main Idea

Methodology

D-RaNGe – A DRAM based TRNG

Evaluation

Strengths and Weaknesses

Discussion

Outline

Background

Motivation

Goal

Main Idea

Methodology

D-RaNGe – A DRAM based TRNG

Evaluation

Strengths and Weaknesses

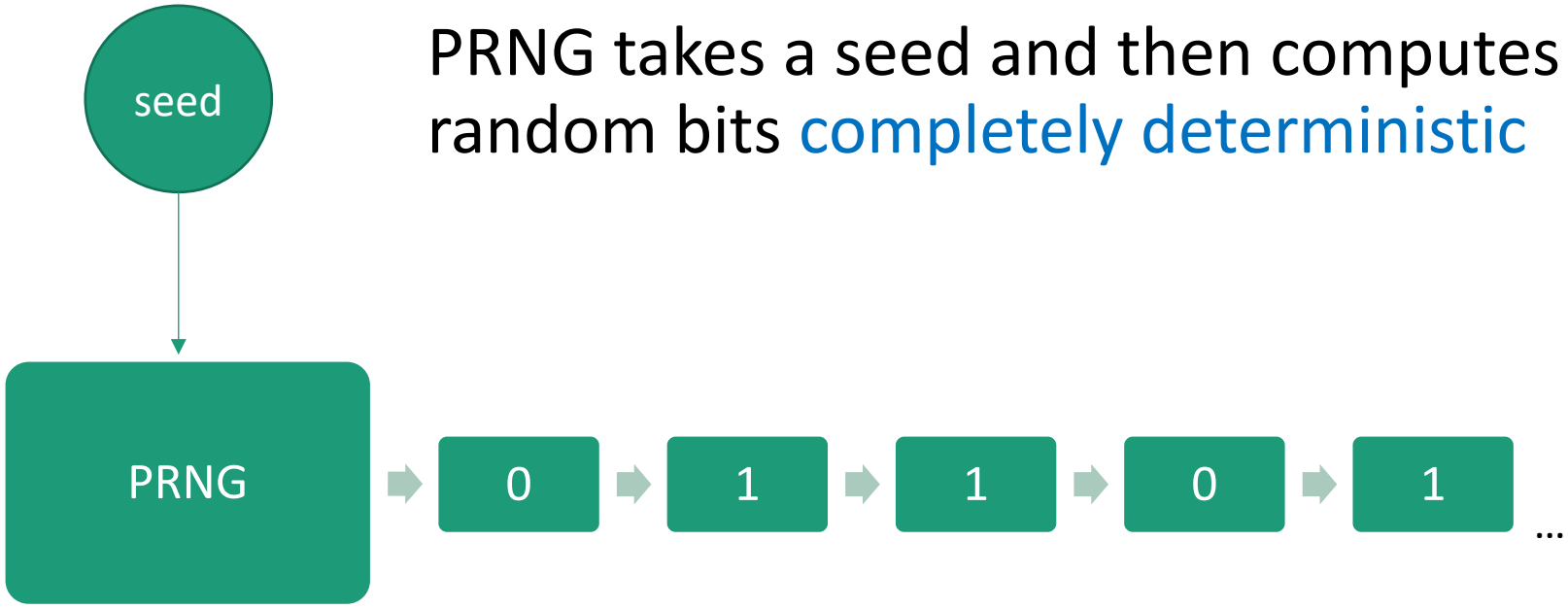
Discussion

Background

Pseudorandom number generator (PRNG)

True random number generator (TRNG)

Pseudorandom Number generator



PRNG takes a seed and then computes the random bits completely deterministic

True Random Number Generator

True random number generators sample truly random physical processes, called an entropy source.

They are fully non-deterministic, even with total knowledge about the underlying process. No initial seed is needed.

They may need a post-processing step to fix unwanted biases and correlation.

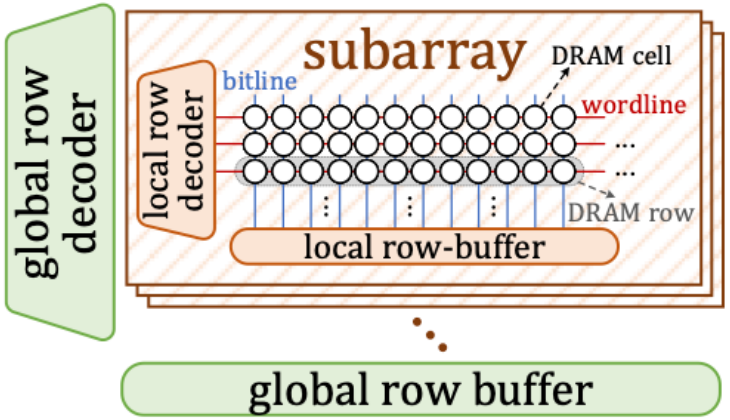
They are limited in throughput by the physical process.

DRAM Bank hierarchy

A DRAM bank is structured into several subarrays.

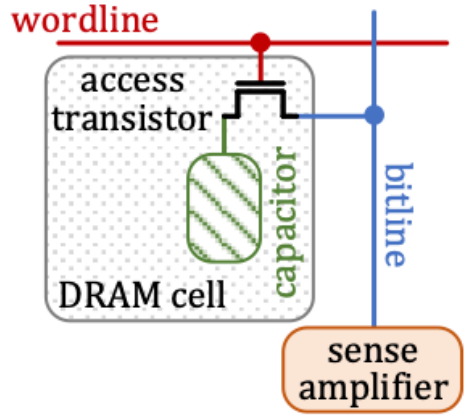
Columns of cells have the same local bitline.

Rows share the same local wordline.



Reading data from a DRAM cell

To read data from a cell, the memory controller first sends an activation command to the DRAM row.



This activates the wordline, and thus the access transistor.

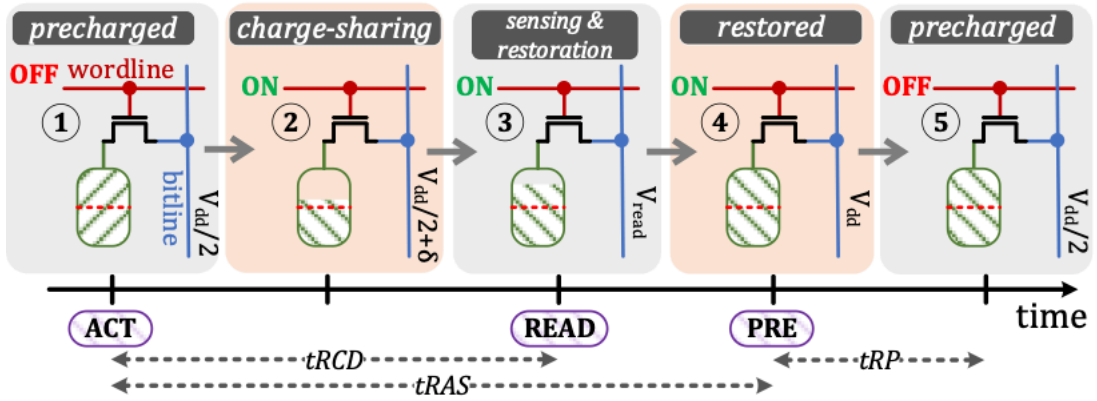
In the next phase, the voltage of the bitline gets perturbed from Vdd/2 towards Vdd/2 + δ. This is called the charge sharing process.

Reading data from a DRAM cell

The signal then travels to the sense amplifier at the end of the bitline which puts the voltage to either 0 or Vdd, depending on the signal. Also the charge inside the capacitor gets restored in this phase.

The tRCD limit is the minimum time, the memory controller has to wait before the bit can be safely read.

If we want to read from another row, the memory controller sends a precharge command to close this row.



Outline

Background

Motivation

Goal

Main Idea

Methodology

D-RaNGe – A DRAM based TRNG

Evaluation

Strengths and Weaknesses

Discussion

Motivation

Why do we need random numbers?

Motivation

One of the applications is Cryptography.

Any deterministic encryption is not semantically secure, so that‘s why cryptography has to use random numbers.

(How secure is deterministic encryption?)

semantic security: Only negligible information about the plaintext can be feasible extracted from the cyphertext.

The advantages of a TRNG in DRAM

It doesn‘t require special hardware, which most high throughput TRNGs need.

Most devices have DRAM, so a TRNG in DRAM can be used almost anywhere.

Outline

Background

Motivation

Goal

Main Idea

Methodology

D-RaNGe – A DRAM based TRNG

Evaluation

Strengths and Weaknesses

Discussion

Goal

The goal is to design a TRNG that

- is implementable on commodity DRAM devices today
- is fully non-deterministic
- provides continous high-throughput random values at low latency
- provides random values while minimally affecting concurrently-running applications.

Outline

Background

Motivation

Goal

Main Idea

Methodology

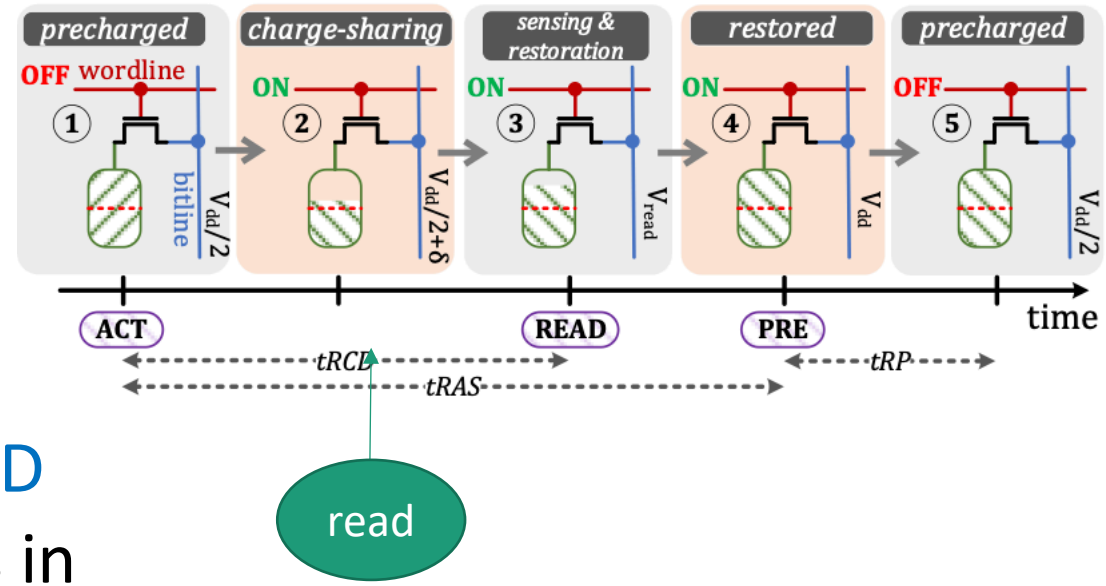
D-RaNGe – A DRAM based TRNG

Evaluation

Strengths and Weaknesses

Discussion

Main Idea



The main idea is to read data before the tRCD limit is over. If we get the opposite of what's in the cell, that is called an activation failure.

We want to access cells with ~50% failure probability(RNG cells), when accessed with reduced tRCD limit.

Is this a good entropy source?

Do RNG cells provide good enough quality of random numbers?

Are there enough of them to ensure high throughput?

Outline

Background

Motivation

Goal

Main Idea

Methodology

D-RaNGe – A DRAM based TRNG

Evaluation

Strengths and Weaknesses

Discussion

Methodology

The tests were done with 282y-nm LPDDR4 DRAM chips from 3 different manufacturers.

They did the tests in a thermally controlled chamber with an ambient temperature of 40-55℃ with an accuracy of 0.25 ℃

They had complete control over DRAM commands and timing parameters.

They used a different infrastructure for DDR4 chips based on open-source SoftMC.

Outline

Background

Motivation

Goal

Main Idea

Methodology

D-RaNGe – A DRAM based TRNG

Evaluation

Strengths and Weaknesses

Discussion

D-RaNGe – A DRAM based TRNG

First, the RNG cells have to get pre-identified and stored.

So there is a slight overhead in storing all those locations.

It was showed in the tests that entropy variation is not a problem, so RNG cells stay RNG cells for a long time.

D-RaNGe – A DRAM based TRNG

Key idea: Access RNG cells with reduced tRCD parameter.

For more throughput, it generates random bits in parallel across multiple DRAM banks.

Within a bank, it chooses two words with the highest density of RNG cells, because

- ▶each access generates more random bits that way
- ▶after every access, the DRAM cell has to be closed -> alternate accesses

D-RaNGe gets exclusive access to those DRAM words (and physically adjacent rows)

Outline

Background

Motivation

Goal

Main Idea

Methodology

D-RaNGe – A DRAM based TRNG

Evaluation

Strengths and Weaknesses

Discussion

Evaluation

Quality of randomness

Throughput

Latency

Energy consumption

Quality of Randomness

Goal: We want to make sure these random bits satisfy these statistical properties.

Methodology:

They generated 1Mb random bitstreams per RNG cell.

- The bitstreams have to pass the NIST statistical test.

- They passed the NIST test with good p-values, so the quality of randomness is high.

Throughput

The throughput depends on

- the density of RNG cells and
- the number of available DRAM banks.

Methodology:

They measured the throughput for an example memory system with 4 DRAM channels.

The peak throughput was 717.4Mb/s, the average throughput was 435.7Mb/s.

Latency

- The latency is related with the latency of a normal DRAM access.

Methodology:

They have computed an upper bound for the latency of a 64 bit word, 960ns, by assuming that every used DRAM word only has a single RNG cell.

- The best empirical latency for a 64 bit word was 100ns.

- So the average latency will be between 100-960ns.

Energy consumption

Methodology:

Subtract the energy needed of an idle system from one where D-RaNGe generates random bitstreams. Then divide this number by the number of random bits that were generated.

The average energy consumption was 4.4nJ/bit.

Comparison to related works

Data retention:

Dynamic memory-based physically unclonable function for the generation of unique identifiers and true random numbers

Key idea: DRAM cells are leaky and lose charge over time. That is called data retention. The entropy comes from the variation of how fast cells lose charge.

Peak throughput: 0.05Mb/s

Latency: 40s per 64 bit word

Comparison to related works

Startup values:

Robust hardware true random number generators using DRAM remanence effects

Key idea: When starting up the DRAM system, the bitlines have a voltage of Vdd/2, that can be amplified to either 0 or Vdd. That‘s the entropy source.

Disadvantages: It is very unpractical to keep rebooting the DRAM system when generating random values.

Outline

Background

Motivation

Goal

Main Idea

Methodology

D-RaNGe – A DRAM based TRNG

Evaluation

Strengths and Weaknesses

Discussion

Strengths

- The method improves throughput and latency of TRNGs in DRAM by orders of magnitude, and it‘s scalable.

- It can be implemented on any device with commodity DRAM today.

- It‘s possible to choose a tradeoff between system interference and D-RaNGe throughput.

Weaknesses

- D-RaNGe runs in the memory controller. A mistake in the implementation could result in large damage.

- They tested DRAM activation failures on chips from 3 different manufacturers, but didn‘t name them. That violates a core tenet of science, repeatability.

- The overhead of storing the DRAM rows was claimed to be 0.018%, which is a low percentage, but can still be a high absolute value.

Discussion

Can we improve throughput or latency by using a similar approach in Cache?

Discussion

Can we improve throughput or latency by using a similar approach in Cache?

I think the problem here is that Cache is much smaller in size, and thus the system interference gets too high by giving exclusive access to a lot of cache lines.

Discussion

The technique is promising. What are potential problems in the implementation?

Discussion

Is this method as widely applicable as they claim? What issues do you see?

Questions?

Testing

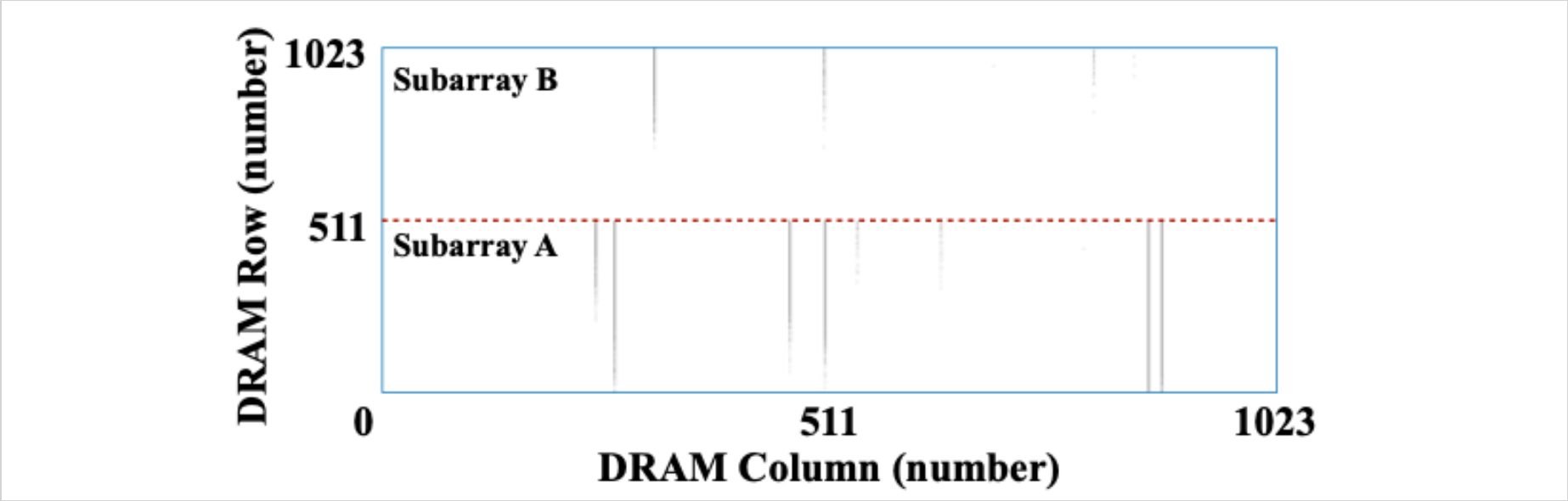
Tests were made for

- 3 different manufacturers
- importance of temperature
- different data patterns in the cells

and a lot of repetitions to trigger as much activation failures as possible

Example

- Activation failures happen only in few columns, but there almost the entire column
- Activation failures more likely in higher numbered rows in subarrays



Data pattern dependence

Goal:

Find data pattern which can find the most cells that are vulnerable to activation failure.

Data pattern dependence - Results

- different data patterns lead to different vulnerable cells found
- some data patterns are systematically better than others

Another interesting result:

When we repeat the experiment, the total number of vulnerable cells found increased, for all data patterns.

Temperature

A change in temperature can have an impact on the failure probability.

On average, the failure probability increases with higher temperature

Large differences between manufacturers in variance of temperature effects

Entropy variation over time

Is entropy variation over time a problem?

Short answer: No.

In a window of 15 days, no significant change.

But maybe long-term?

D-RaNGe: A DRAM-based TRNG

- key idea: violate tRCD limit to trigger activation failures
- subset of cells fail non-deterministically with 50% probability with high entropy. We call them RNG cells from now.
- RNG cells give unbiased output, so no postprocessing step necessary

RNG cells

- Identify RNG cells through a lot of testing
- store them in the memory controller

What about the temperature problem we identified?

-> have to store a mapping for temperature to RNG cells

D-RaNGe

Algorithm 2: D-RaNGe: A DRAM-based TRNG

```
1  D-RaNGe(num_bits):   // num_bits: number of random bits requested
2     DP: a known data pattern that results in high entropy
3     select 2 DRAM words with RNG cells in distinct rows in each bank
4     write DP to chosen DRAM words and their neighboring cells
5     get exclusive access to rows of chosen DRAM words and nearby cells
6     set low t_RCD for DRAM ranks containing chosen DRAM words
7     for each bank:
8        read data in DRAM word 1 (DW_1)  // induce activation failure
9        write the read value of DW_1's RNG cells to bitstream
10       write original data value back into DW_1
11       memory barrier            // ensure completion of write to DW_1
12       read data in DRAM word 2 (DW_2)  // induce activation failure
13       write the read value of DW_2's RNG cells to bitstream
14       write original data value back into DW_2
15       memory barrier            // ensure completion of write to DW_2
16       if bitstream_size ≥ num_bits:
17          break
18    set default t_RCD for DRAM ranks of the chosen DRAM words
19    release exclusive access to rows of chosen words and nearby cells
```

Evaluation

Quality of randomness: good

Throughput: peak 717.4Mb/s, average 435.7Mb/s

Latency: 100ns – 960ns

Energy Consumption: 4.4nJ/bit

Comparison to prior TRNGs

Proposal	Year	Entropy Source	True Random	Streaming Capable	64-bit TRNG Latency	Energy Consumption	Peak Throughput
Pyo+ [116]	2009	Command Schedule	✗	✓	$18\mu s$	N/A	$3.40Mb/s$
Keller+ [65]	2014	Data Retention	✓	✓	$40s$	$6.8m\mathcal{J}/bit$	$0.05Mb/s$
Tehranipoor+ [144]	2016	Startup Values	✓	✗	$> 60ns$ (optimistic)	$> 245.9p\mathcal{J}/bit$ (optimistic)	N/A
Sutar+ [141]	2018	Data Retention	✓	✓	$40s$	$6.8m\mathcal{J}/bit$	$0.05Mb/s$
D-RaNGe	2018	Activation Failures	✓	✓	$100ns < x < 960ns$	$4.4n\mathcal{J}/bit$	$717.4Mb/s$

Table 2: Comparison to previous DRAM-based TRNG proposals.

Strengths

- Wide application range, because it runs on DRAM
- Great throughput improvement
- Low latency
- Low system interference
- Important problem

- (easy to read)

Weaknesses

- Can‘t compete with special hardware TRNGs
- (long term effect of repeatedly accessing DRAM cells with reduced tRCD limit)

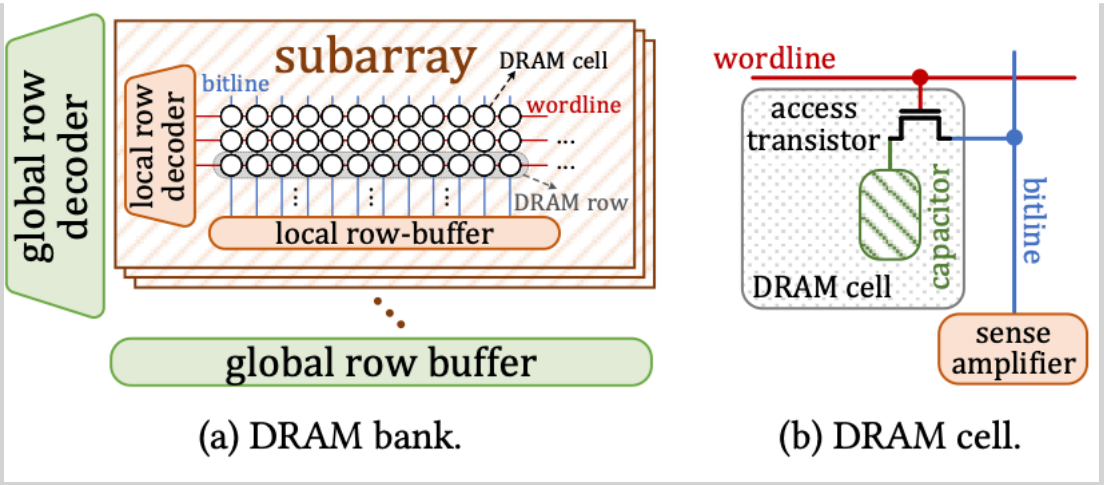
Discussion

Do you think D-RaNGe will get used by high throughput cryptographic applications, or that they rather have specialized hardware?

Will there be high enough density RNG cells in DRAM in the future to guarantee high throughput?

Why not use D-RaNGe to generate a random seed and then use a Pseudo Random Number Generator?

DRAM



(a) DRAM bank. (b) DRAM cell.

Throughput

The throughput of D-RaNGe depends on

- density of RNG cells per DRAM word
- bandwidth of DRAM words when using this methodology

Observation:

- There are a lot of RNG cells in every bank
- We can use access parallelism for more throughput

Evaluation

- Low implementation cost -> simple software that is able to access DRAM below manufacturers limit
- Fully non-deterministic
- High throughput of random data(717.4Mb/s peak)
- Low Latency(100-960ns)
- Low System Interference
- Low Energy consumption(4.4nJ/bit)

Strengths

- A lot higher throughput in random data than any prior method which uses DRAM as an entropy source
- Easy to implement on lots of different machines
- good quality of random data
- important problem, as more and more data has to be encrypted

Weaknesses

- Not much actual data shown