# Computer architecture

Hardware security (addon slides) – RowHammer, Spectre, Meltdown

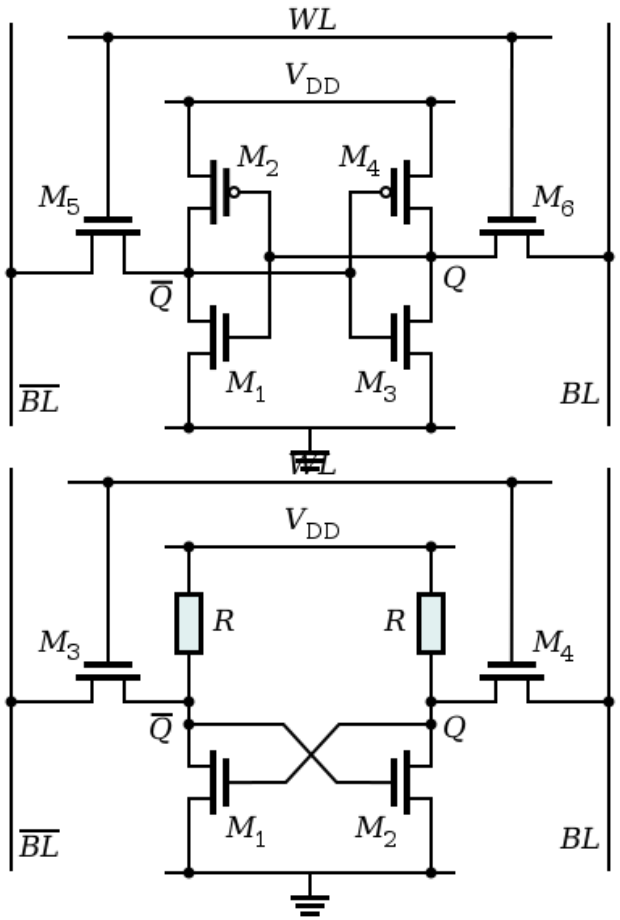
# First, let’s talk about general hardware security (The semiconductor security war)

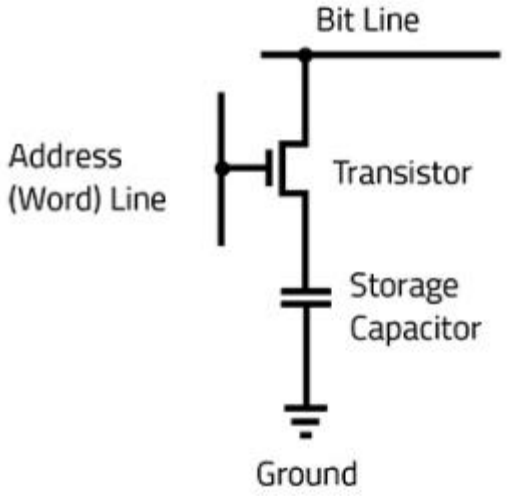
# RowHammer attacks

# The usual talking point

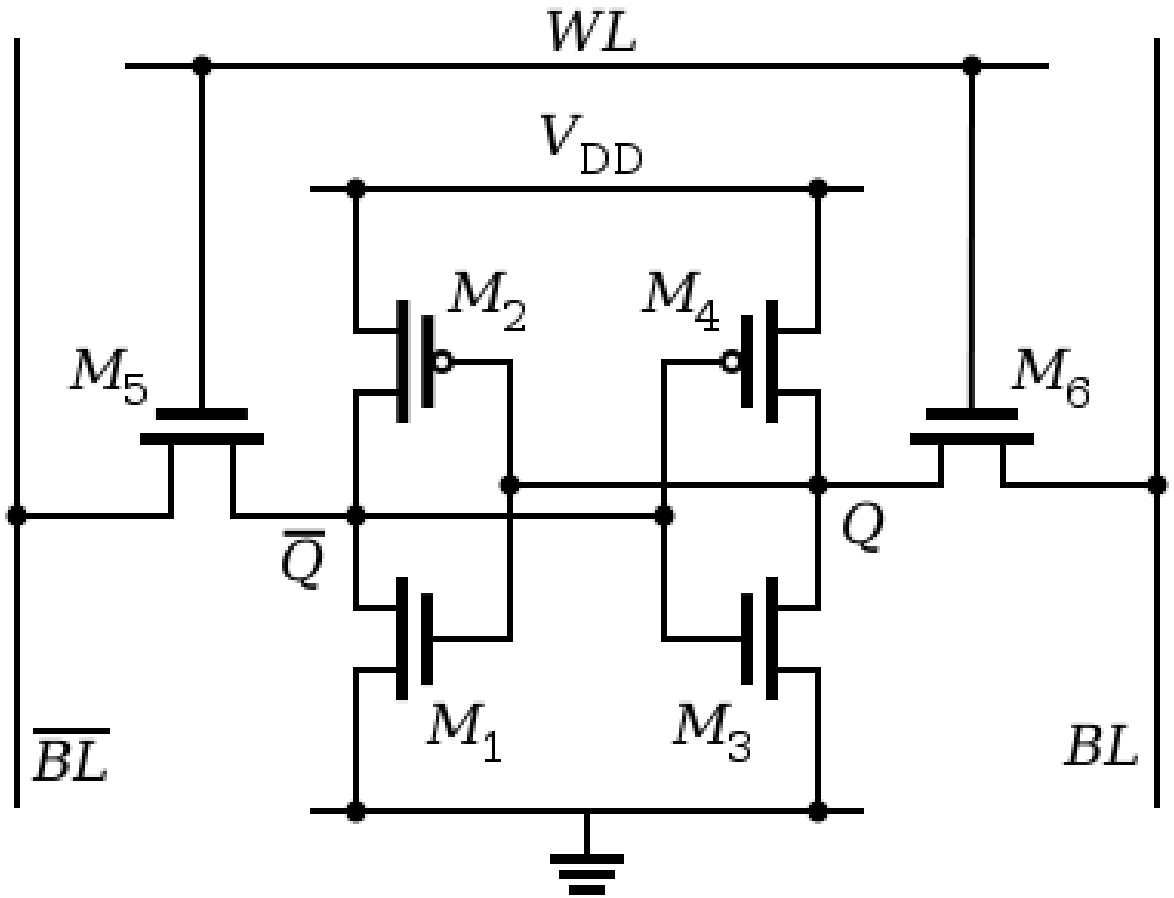
- Why are we over-complicating things – let's just solve the memory bandwidth problem by doing away with the "slow" DRAM memory – let's just have everything in the fast, SRAM memory and caches
- Let's discuss why this is both impractical and impossible to do so

# SRAM vs DRAM

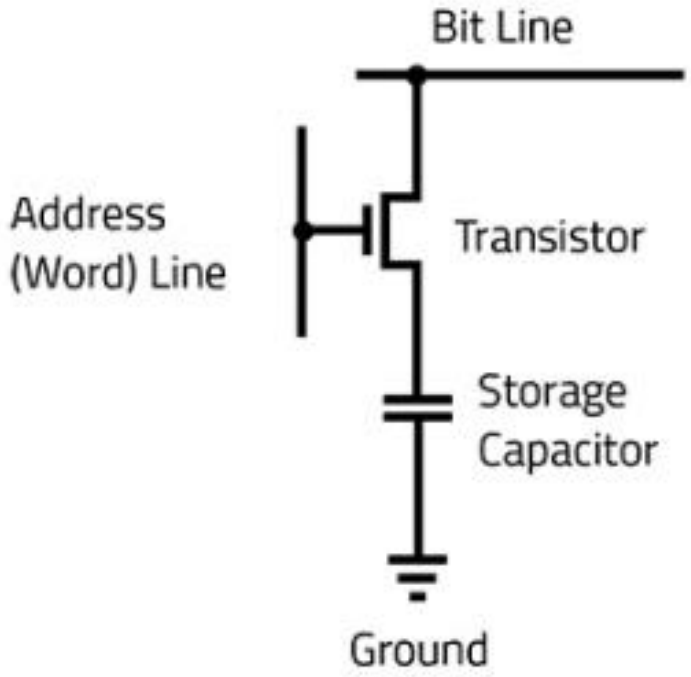

WL
$V_{DD}$
$M_2$
$M_4$
$M_5$
$M_6$
$\overline{Q}$
$Q$
$M_1$
$M_3$
$\overline{BL}$
$BL$
WL
$V_{DD}$
R
R
$M_3$
$M_4$
$\overline{Q}$
$Q$
$M_1$
$M_2$
$\overline{BL}$
$BL$


Bit Line
Address
(Word) Line
Transistor
Storage
Capacitor
Ground

# SRAM



# DRAM


Bit Line
Address
(Word) Line
Transistor
Storage
Capacitor
Ground

# An example (Sandy Bridge architecture)

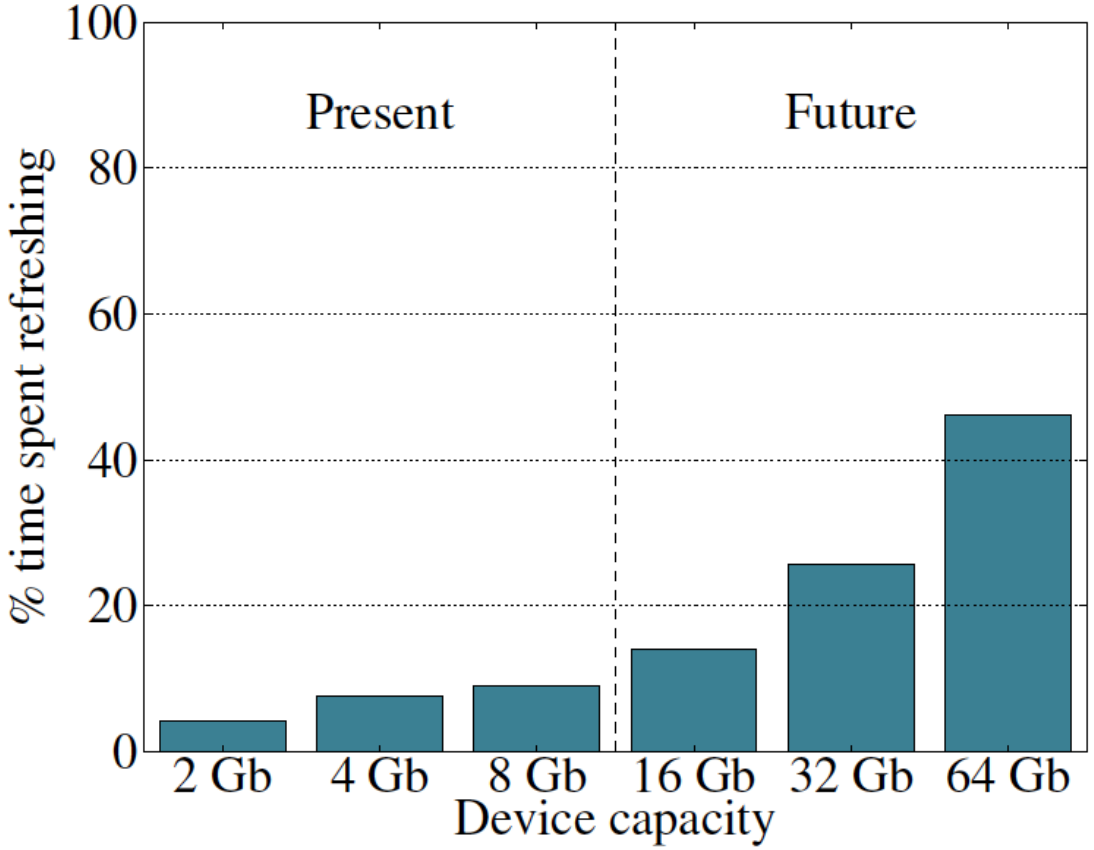
- L1 cache size 32KB cca 3 cycles
- L2 cache size 256KB cca 8 cycles
- L3 cache size 10-20MB cca 35 cycles
- Main memory – whatever GBs are supported/we put in, cca 250 cycles

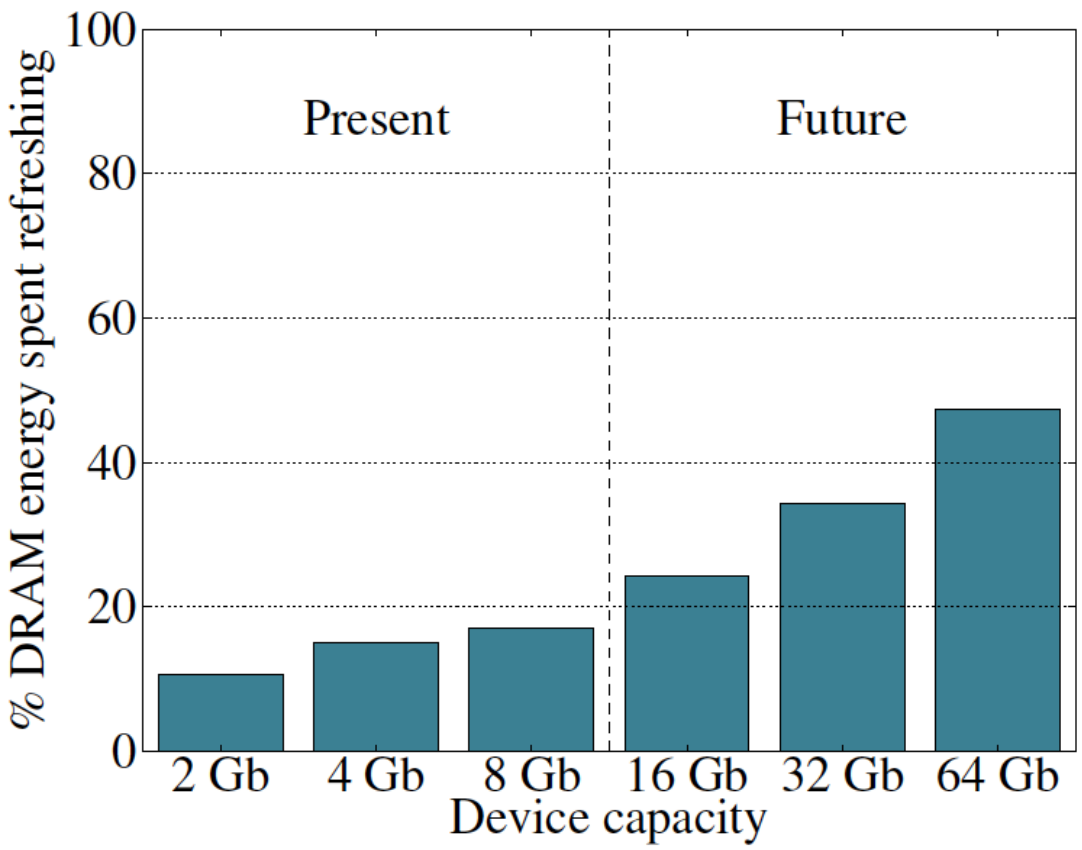
# Why do we care?

# Simple explanation

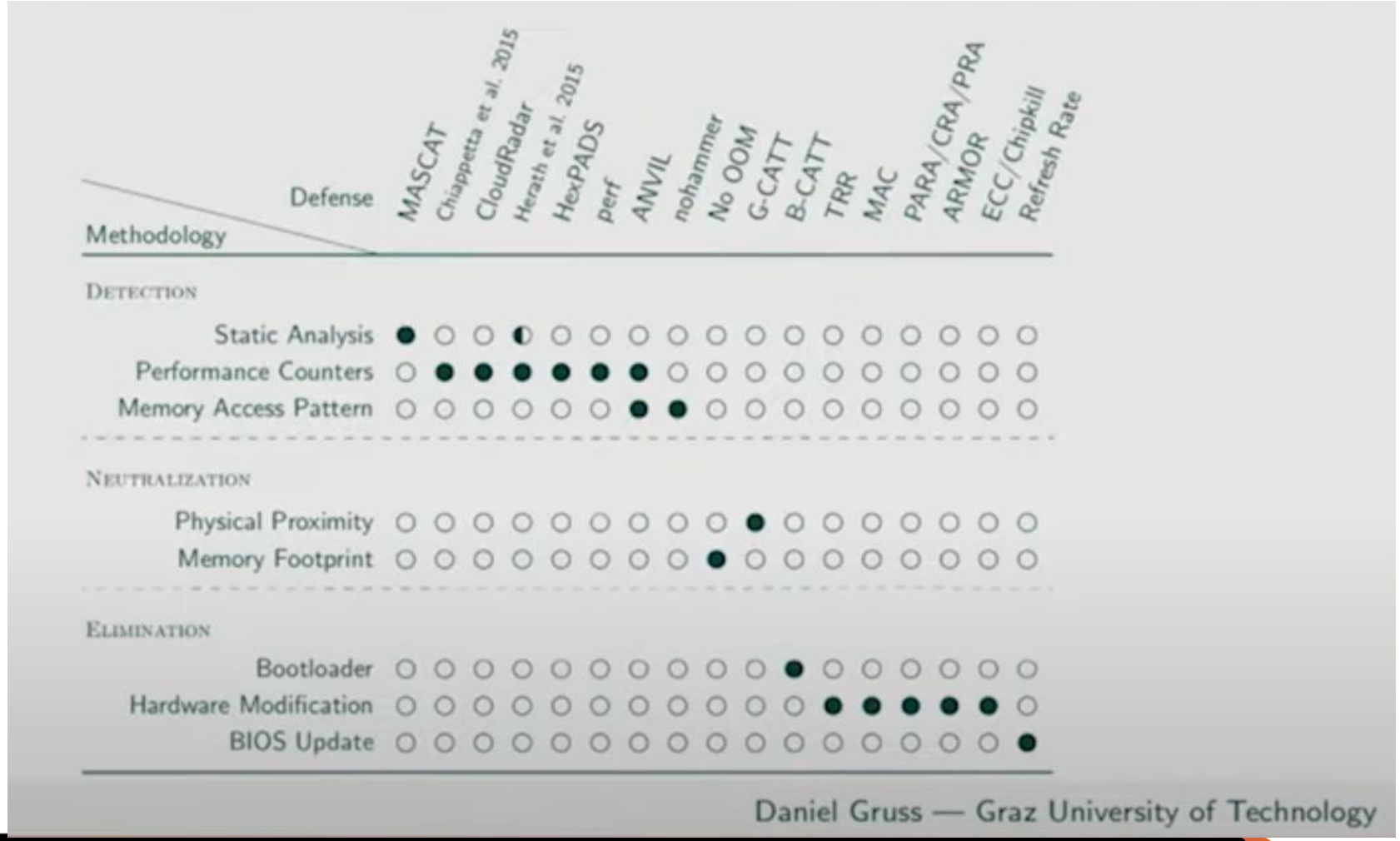
- As we previously noted, memory is the bottleneck
- Memory development in terms of latency hasn’t been all that great in the past 3 decades
- Whatever we could gain in terms of bandwidth, lowering the cost and expanding the address range (32, 64-bit) – it was worthwhile
- The downside of that is that it’s really, really coming to haunt us now
- As memory technology gets developed, cells are getting closer and closer together, and that has direct effect on performance
- It would be cool if we could go around basic physics, but just because we want that really, really badly, it doesn’t mean it’s possible ☺
- Competing design goals and market needs have put us into this position where Spectres, Meltdowns and RowHammers of the world are “a thing”
- RowHammer is “a thing” with all memory tech, especially DDR3+ - the newer the chip is, the worse it usually is

# Refresh problems – performance and energy



# Some detection, neutralization and elimination mechanisms for RowHammer



# Spectre and Meltdown

# What are these attacks?

# Meltdown and Spectre Attacks

- Someone can steal secret data from the system even though
  - your program and data are perfectly correct and
  - your hardware behaves according to the specification and
  - there are no software vulnerabilities/bugs

# Meltdown and Spectre

- Hardware security vulnerabilities that essentially effect almost all computer chips that were manufactured in the past two decades
- They exploit “speculative execution”
  - A technique employed in modern processors for high performance
- Speculative execution: Doing something before you know that it is needed
  - We do it all the time in life, to save time
    - Guess what will happen and act based on that guess
  - Processors do it, too, to run programs fast
    - They guess and execute code before they know it should be executed

# Speculative Execution

- Modern processors “speculatively execute” code to improve performance:

```
if (account-balance <= 0) {
    // do something
} else if (account-balance < 1M) {
    // do something else
} else {
    // do something else
}
```

Guess what code will be executed and execute it speculatively
- Improves performance, if it takes a long time to access account-balance

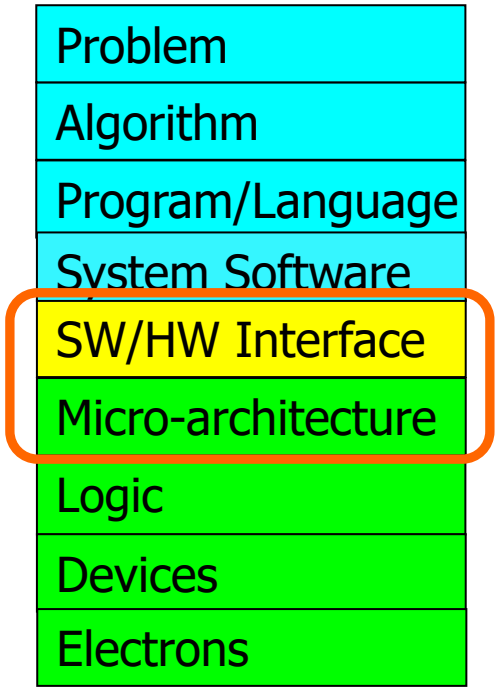
If the guess was wrong, flush the wrong instructions and execute the correct code

# Speculative Execution is Invisible to the User

ISA
(Instruction Set Architecture)

Interface/contract between SW and HW.

What the programmer assumes hardware will satisfy.




Microarchitecture
An implementation of the ISA

Programmer assumes their code will be executed in sequential order

Microarchitecture executes instructions in a different order, speculatively – but reports the results as expected by the programmer

# Meltdown and Spectre

- Someone can steal secret data from the system even though
  - your program and data are perfectly correct and
  - your hardware behaves according to the specification and
  - there are no software vulnerabilities/bugs

- Why?
  - Speculative execution leaves traces of secret data in the processor's cache (internal storage)
    - It brings data that is not supposed to be brought/accessed if there was no speculative execution
  - A malicious program can inspect the contents of the cache to "infer" secret data that it is not supposed to access
  - A malicious program can actually force another program to speculatively execute code that leaves traces of secret data

# Processor Cache as a Side Channel

- Speculative execution leaves traces of data in processor cache
  - **Architecturally correct behavior in accordance to design specification**
  - However, this leads to a side channel: a channel through which someone sophisticated can extract information
- Processor cache leaks information by storing speculatively-accessed data

# More on Meltdown/Spectre Side Channels

Project Zero

News and updates from the Project Zero team at Google

Wednesday, January 3, 2018

## Reading privileged memory with a side-channel

Posted by Jann Horn, Project Zero

We have discovered that CPU data cache timing can be abused to efficiently leak information out of mis-speculated execution, leading to (at worst) arbitrary virtual memory read vulnerabilities across local security boundaries in various contexts.



https://googleprojectzero.blogspot.ch/2018/01/reading-privileged-memory-with-side.html

# Thank you for your attention!