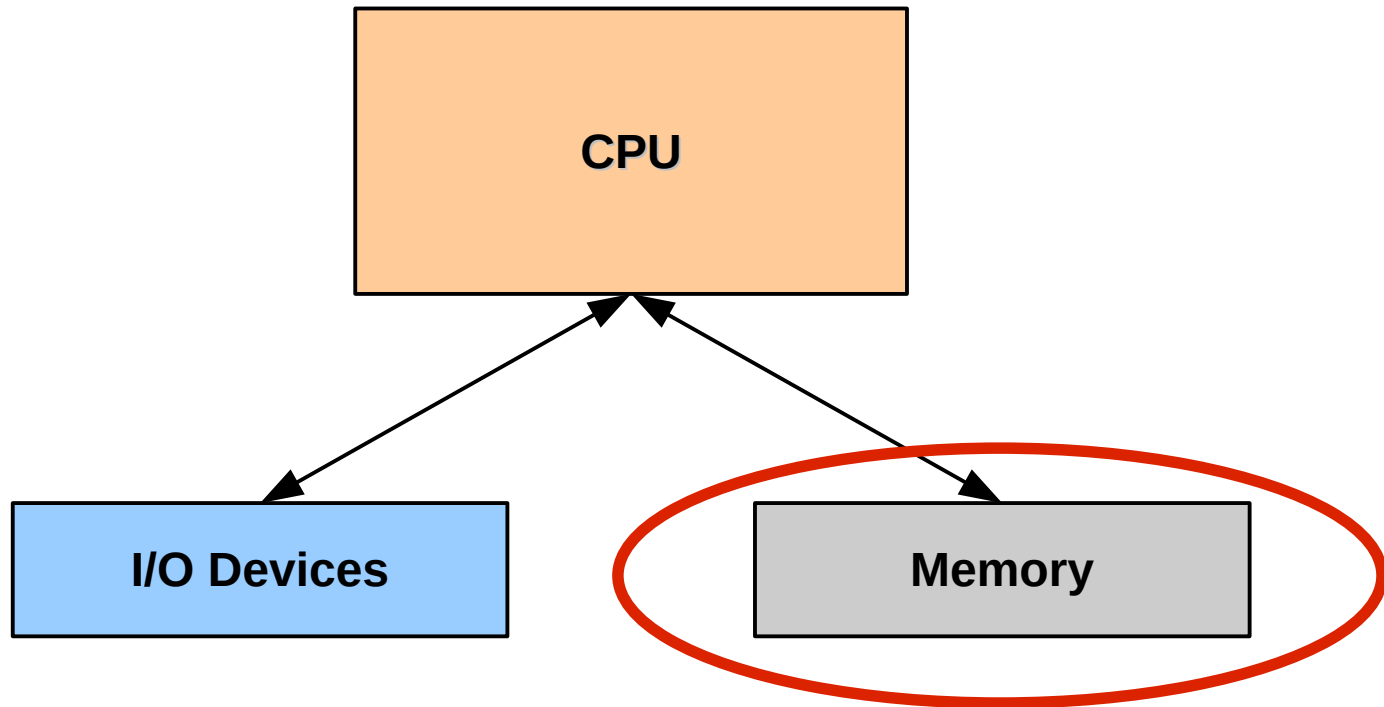# COMPUTER ARCHITECTURES

Random Access Memory Technologies

Prepared by: **Gábor Horváth**, ghorvath@hit.bme.hu

Presented by: **Gábor Lencse**, lencse@hit.bme.hu

Budapest,
2025. 03. 04.

MŰEGYETEM 1782

- Storing single bits:
  - With SRAM
  - With DRAM

- Storing 1 bit: with two inverters
  - A bi-stable flip-flop!
  - **1 bit → 6 transistors**: 6T
    - 2 access transistors:
      connect the flip-flop to BL and $\overline{\text{BL}}$
      (BL = bit line)
    - 4 transistors realizing the flip-flop
- **Reading**:
  - Setting WL (word line) to logical „H"
  - Set to BL → the bit itself, to $\overline{\text{BL}}$ → its inverse
  - Sense amplifiers: monitor the difference between BL and $\overline{\text{BL}}$
- **Writing**:
  - Force BL and $\overline{\text{BL}}$ to the desired value
  - Set logical „H" to WL
  - The driving force is stronger than the inverters

www.hit.bme.hu

DEPARTMENT OF NETWORKED SYSTEMS AND SERVICES
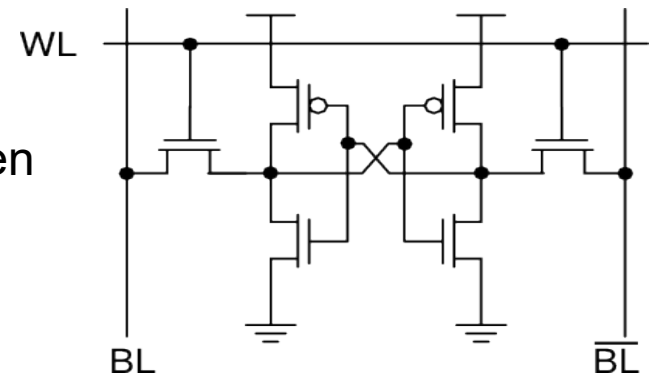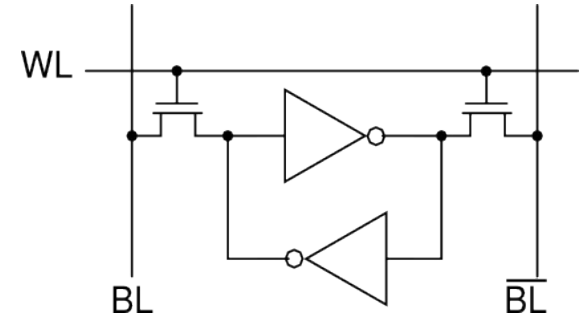
- Storing 1 bit: with a capacitor
  - Charged: bit=1, empty: bit=0
  - + 1 access transistor
  - **1 bit → 1 tranzistor + 1 capacitor**: 1T1C
- **Reading**:
  - Setting BL halfway between 0 and 1 (precharging)
  - Setting logical „H" to WL
  - Sensing amplifiers monitor the level of BL:
    - Increasing: bit=1
    - Decreasing: bit=0
  - The charge escaped during the readout!
    - → Reading a DRAM is desctructive
    - → The original value has to be written back after reading it out
- **Writing**:
  - Setting logical „H" to WL
  - Either charging the capacitor (bit=1) or emptying it (bit=0) through the BL
- **Refreshing**:
  - The charge escapes by itself, too!
  - Periodic refresh is necessary (around every 10 ms) → reading out + writing back everything

access transistor

WL

capacitor

BL
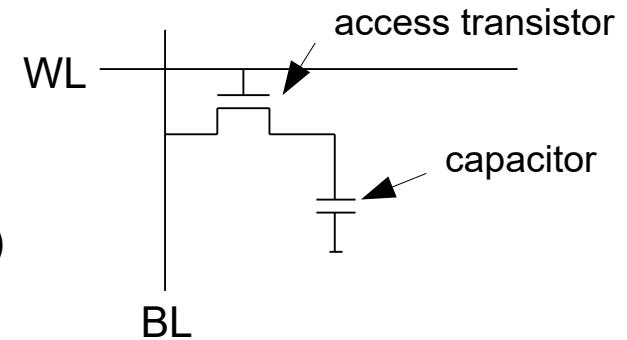
- Which is *faster*?
  - SRAM: the bit value is driven to the bit line by active transistors
  - DRAM: the charge leaving the capacitors (passively) modify the bit lines
  - … and DRAMs need refreshing, too!
  - **→ Faster: SRAM**
- Which has *higher data density*?
  - SRAM: 6T
  - **DRAM:** 1T1C → **higher data density**
- What are they used for?
  - SRAM: cache
  - DRAM: system memory
- Which be can integrated with the CPU?
  - SRAM: easily. It consists of transistors only, like the CPU.
  - DRAM: more difficult. Capacitors have to be created (CPUs don't need them)
  - eDRAM (embedded DRAM): DRAM integrated with CPU
    - 2009, POWER7, 32MB L3 cache made from eDRAM
    - 2013, Intel Haswell GT3e, 128MB L4 cache made from eDRAM
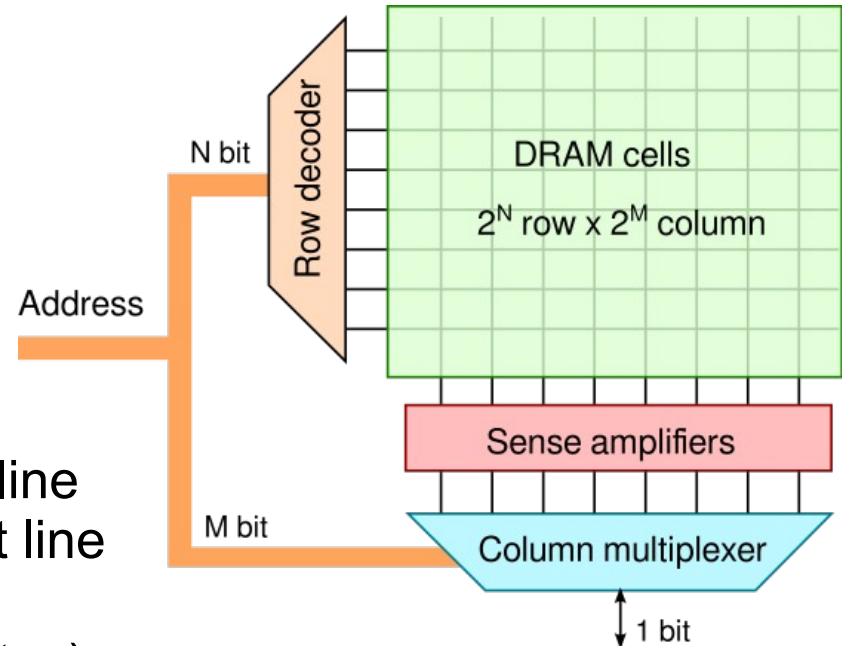   - Game consoles: PlayStation 2, PlayStation Portable, Nintendo Wii U, etc.

DRAM based memory systems

- ## How to create a memory system for the computer
  - ### It must be: cheap, large, low latency, high throughput
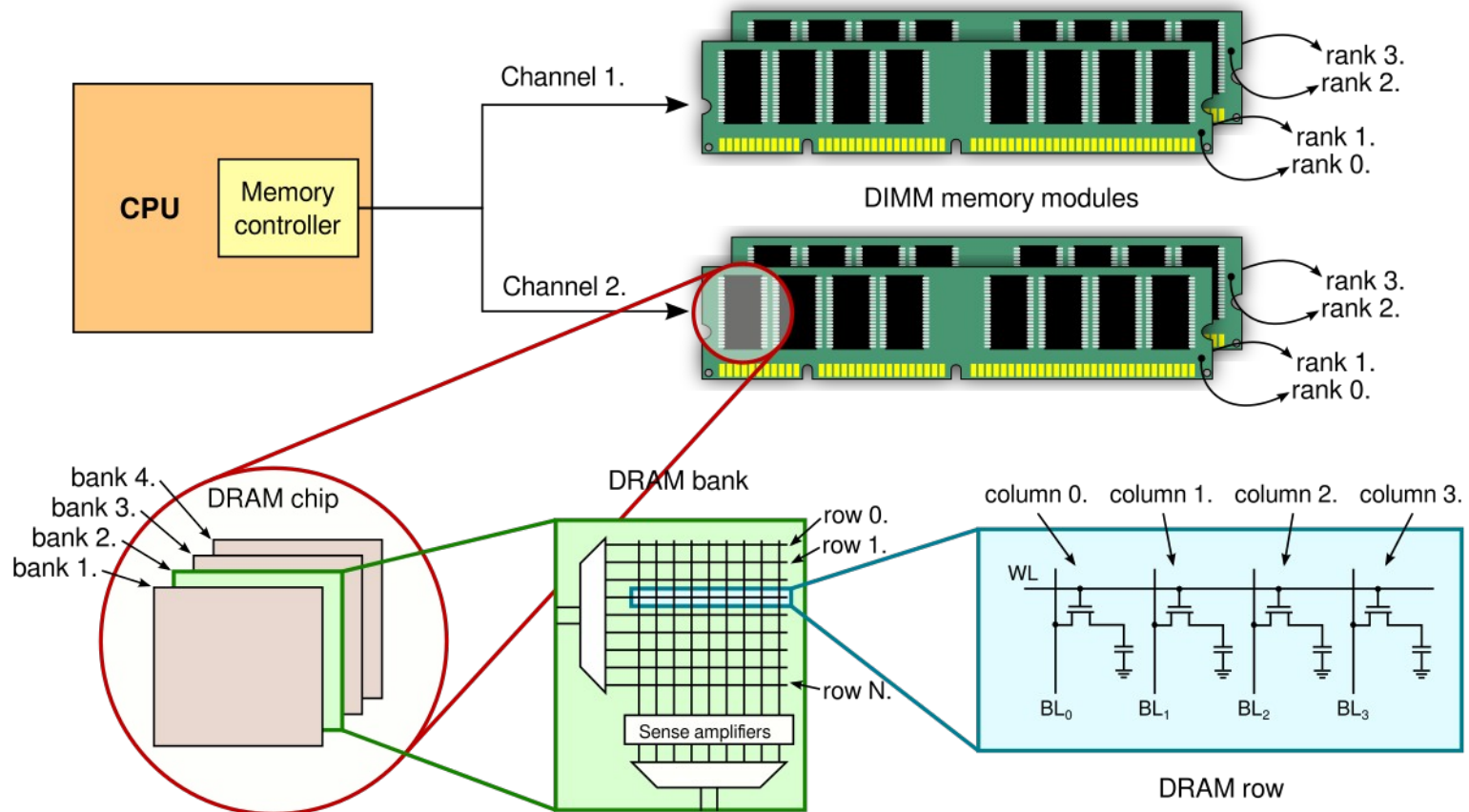
- Structure:

- DRAM cells in a 2D grid
  - Each row shares the same word line
  - Each column shares the same bit line
- Reading:
  - The **row decoder** selects (activates) a row
  - The **sense amplifiers** detect and store the bits of the row
  - The **column multiplexer** selects the desired column from the row
- Two-phase operations:
  - To reduce the width of the address bus
  - Address bus: row address → wait → address bus: column address → data bus: the desired data

Row decoder

N bit

Address

M bit

DRAM cells

2^N row x 2^M column

Sense amplifiers

Column multiplexer

1 bit

- How to obtain the row and column address from the linear address?
- Example: in decimal system (!)
  - Memory capacity: 1 million (address: 0 … 999999)
  - DRAM cell grid: 1000x1000

- Linear address: 123456
- Row and column address: **123|456** → row 123, column 456
- No division is needed, only to split the address!

- Binary system: the same with bits

- The 5 most important commands:
  - **ACTIVATE**
    - Opens a row (data → moved to the sense amplifiers)
  - **READ**
    - Reads a column from the open row
    - It reads from the sense amplifiers
  - **WRITE**
    - Writes a data to the open row
    - It writes to the sense amplifiers
  - **PRECHARGE**
    - Closes the open row
    - Precharges the bit lines to make the next row activation fast
  - **REFRESH**
    - Refreshes the content of a row
    - Almost an activate + precharge
      - But does not need row address. It is auto-incremented each time.

- Example read requests:

    ```
    (row 3, column 8)
    (row 3, column 14)
    (row 1, column 3)
    (row 1, column 4)
    ```

- Commands (assume the DRAM is precharged initially):

    ```
    ACTIVATE 3
    READ 8
    READ 14
    PRECHARGE
    ACTIVATE 1
    READ 3
    READ 4
    ```

- The execution time of the commands
- The 4 most important ones:
  - T_{RCD}: The time needed to open a row (Row-to-Column command Delay)
  - T_{CAS} (CL): The delay between receiving the column address and the appearance of the data (Column Access Strobe time *or* CAS Latancy)
  - T_{RP}: The delay of the PRECHARGE command (Row Precharge)
  - T_{RAS}: The minimal time a row must stay open (Row Active Time)
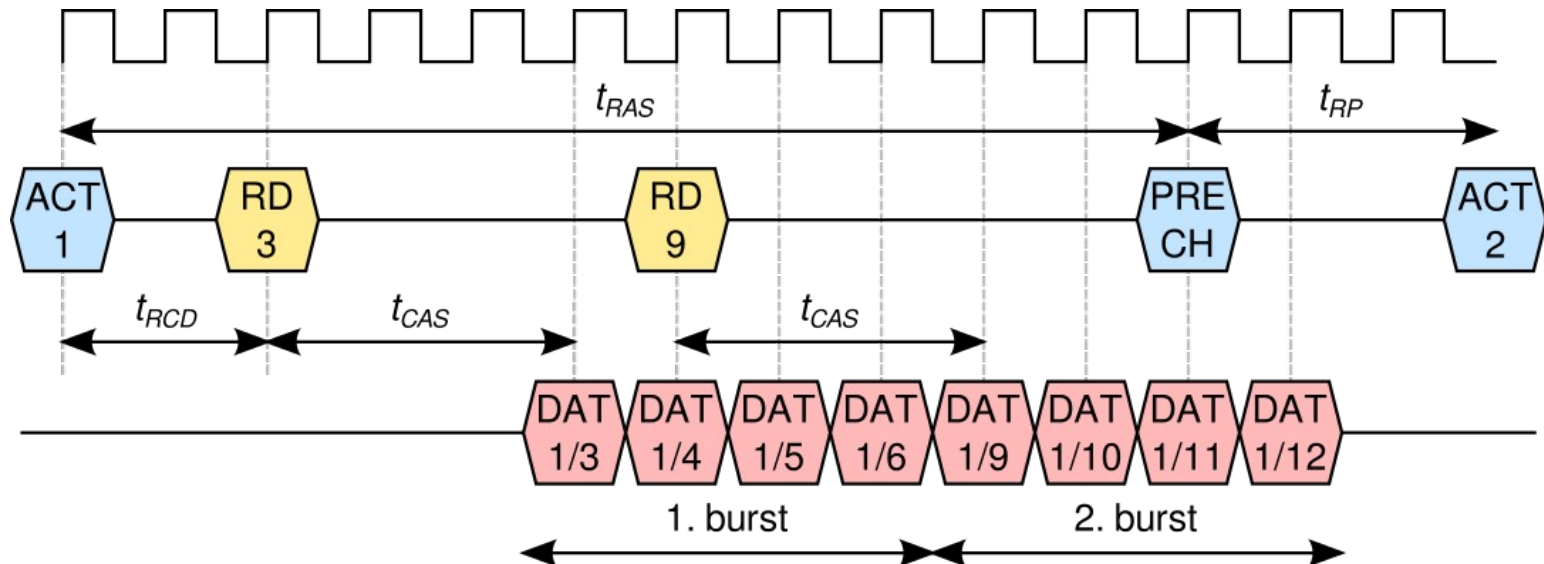
- And there are many of these…
- Unit: clocks (synchronous DRAM), ns (asynchronous)
- A DRAM module parameterized by 8-9-10-11 means:
  - $T_{CAS}=8$, $T_{RCD}=9$, $T_{RP}=10$, $T_{RAS}=11$
- If only „CL7" is provided: $T_{CAS}=7$
- For those, who would like to learn more:

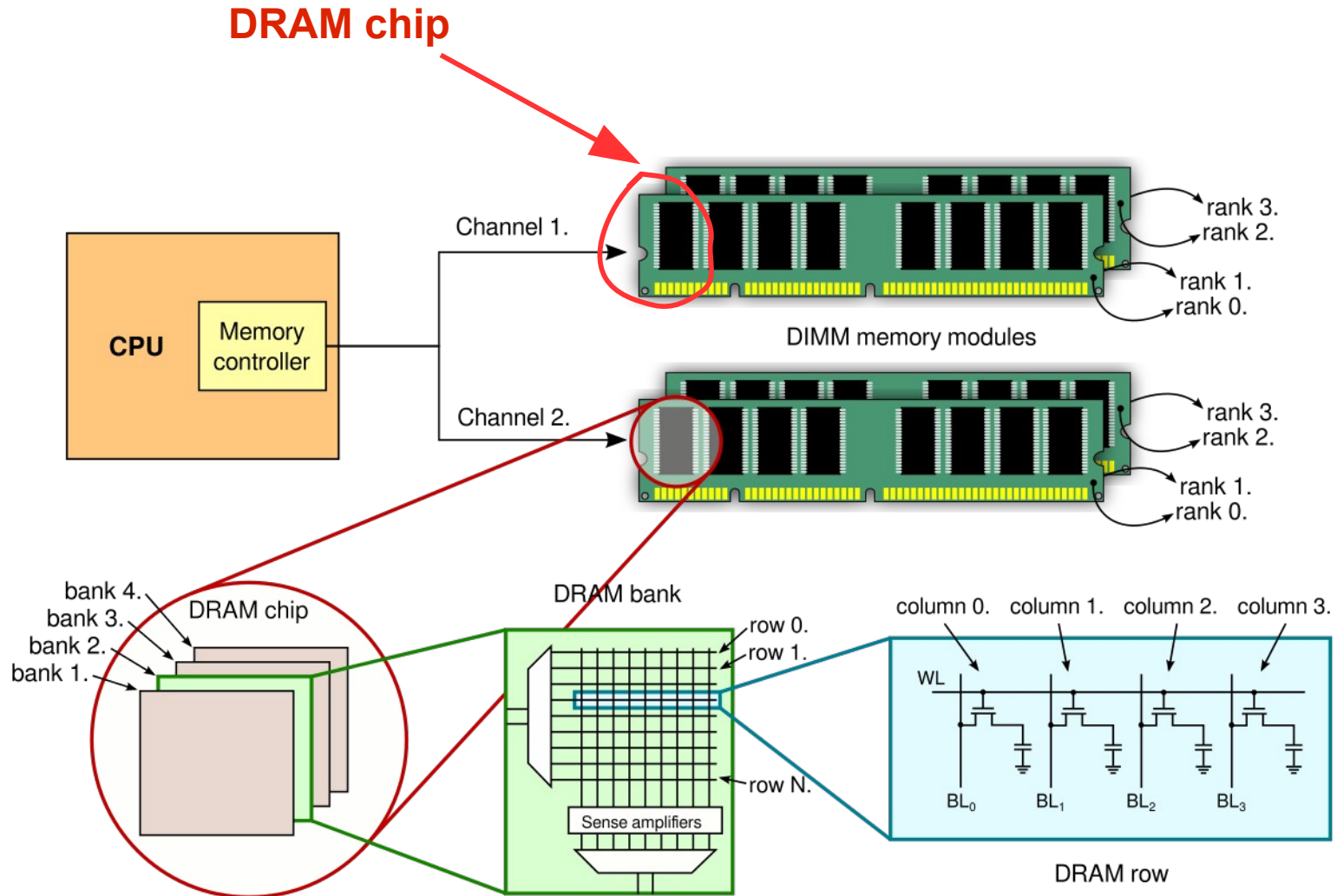   https://www.hardwaresecrets.com/understanding-ram-timings/

- How does the memory controller know the timing values?
  - It asks the memory modules
  - Components of the memory modules:
    - DRAM chips
    - ...and an **SPD** (serial presence detect) chip! It stores the timing parameters (among others)
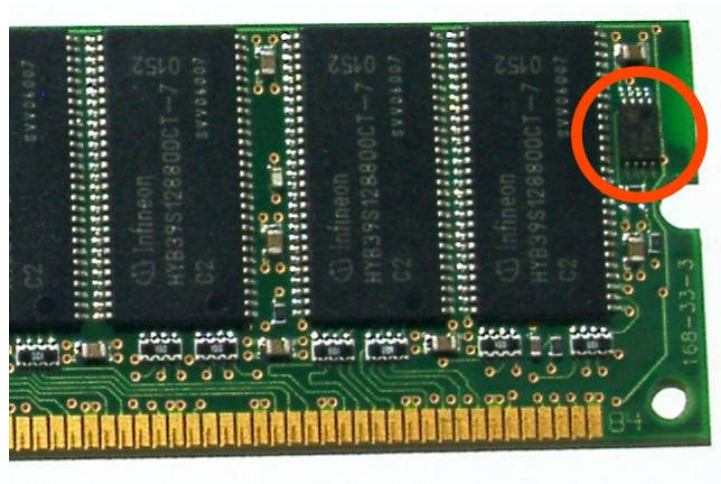
- For every column read operation targetting the same row:
    Column address → T_{CAS} → appearance of the data
    - Waste of time!
- **Burst mode**
    - Column address is given to the DRAM
        … the response is not a single column, but a burst (a series of columns)!
    - Burst length: configuration parameter
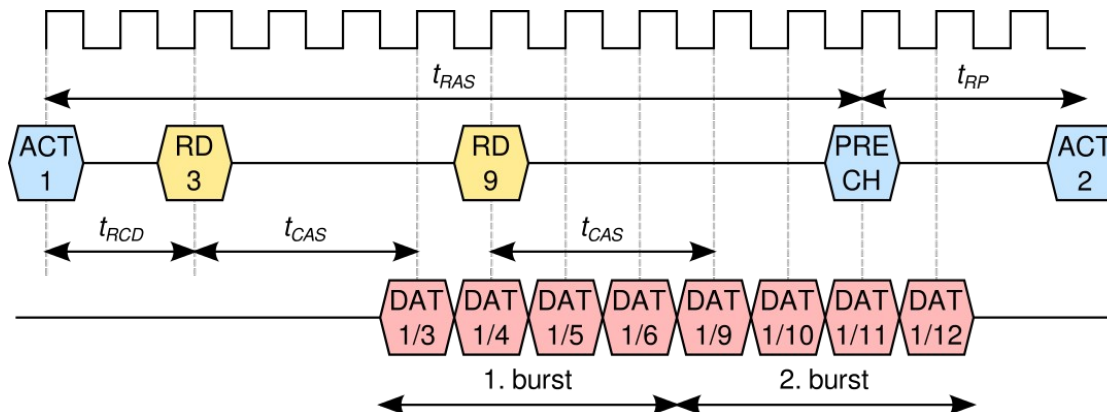
- **The commands and the data can be overlapped**
    - The next command does not have to wait till the current one finishes

DRAM chip

Channel 1.

CPU

Memory controller

Channel 2.

rank 3.
rank 2.
rank 1.
rank 0.

DIMM memory modules

rank 3.
rank 2.
rank 1.
rank 0.

bank 4.
bank 3.
bank 2.
bank 1.

DRAM chip

DRAM bank

row 0.
row 1.

row N.

Sense amplifiers

column 0. column 1. column 2. column 3.

WL

BL_0 BL_1 BL_2 BL_3

DRAM row

- A DRAM chip consists of banks
- Bank:
  - Independent DRAM cell grids
  - Each has its own row decoders, column multiplexers, sense amplifiers
  - Each of them may have a row open
    - → There can be more open rows in a DRAM chip (one in each bank)
    - → **The latency is reduced (less row activations are needed)!**

- Banks do not store individual bits
  - One column: 4 bit, 8 bit, 16 bit (x4, x8, x16)
- Interface:
  - **Command lines**
    - What the chip has to do (ACTIVATE, READ, WRITE, etc.)
  - **Bank selection lines**
    - The bank the command is given to
  - **Address lines**
    - ACTIVATE: row address
    - READ/WRITE: column address
  - **Data lines**
    - 4, 8, or 16 bit
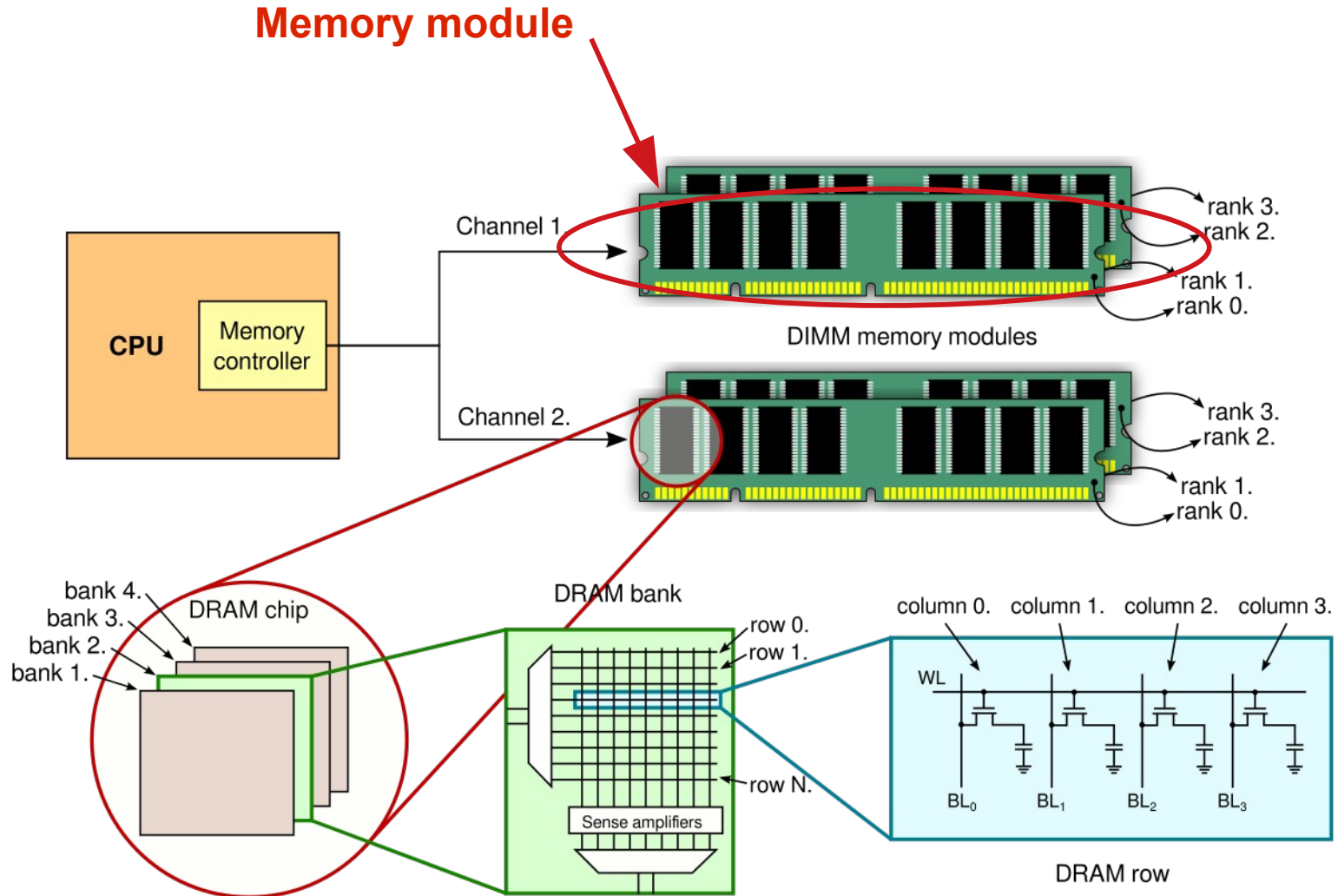
Memory module

- A memory module consists of DRAM chips
- Command lines, bank selection lines, address lines: shared
- Data lines: concatenated

x8 DRAM chips

Command, bank-
selection and addr.
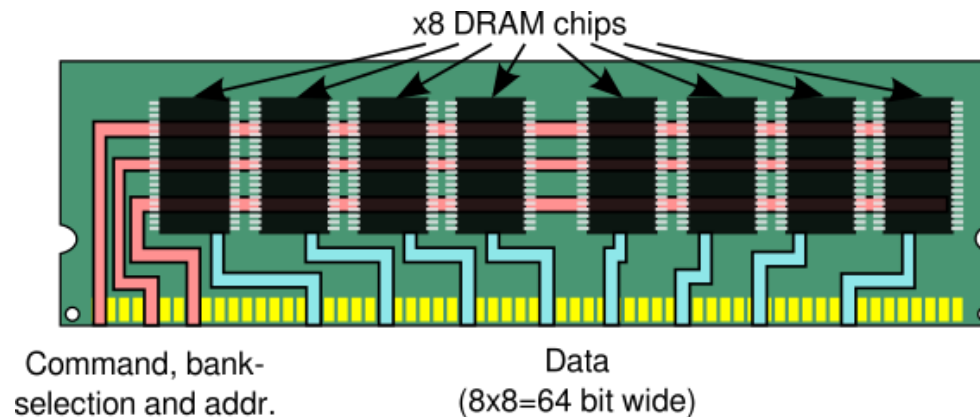
Data
(8x8=64 bit wide)

- Each chip receives all commands
- Effect:
  - Throughput increases 8x
  - Delay: the same

DRAM ranks
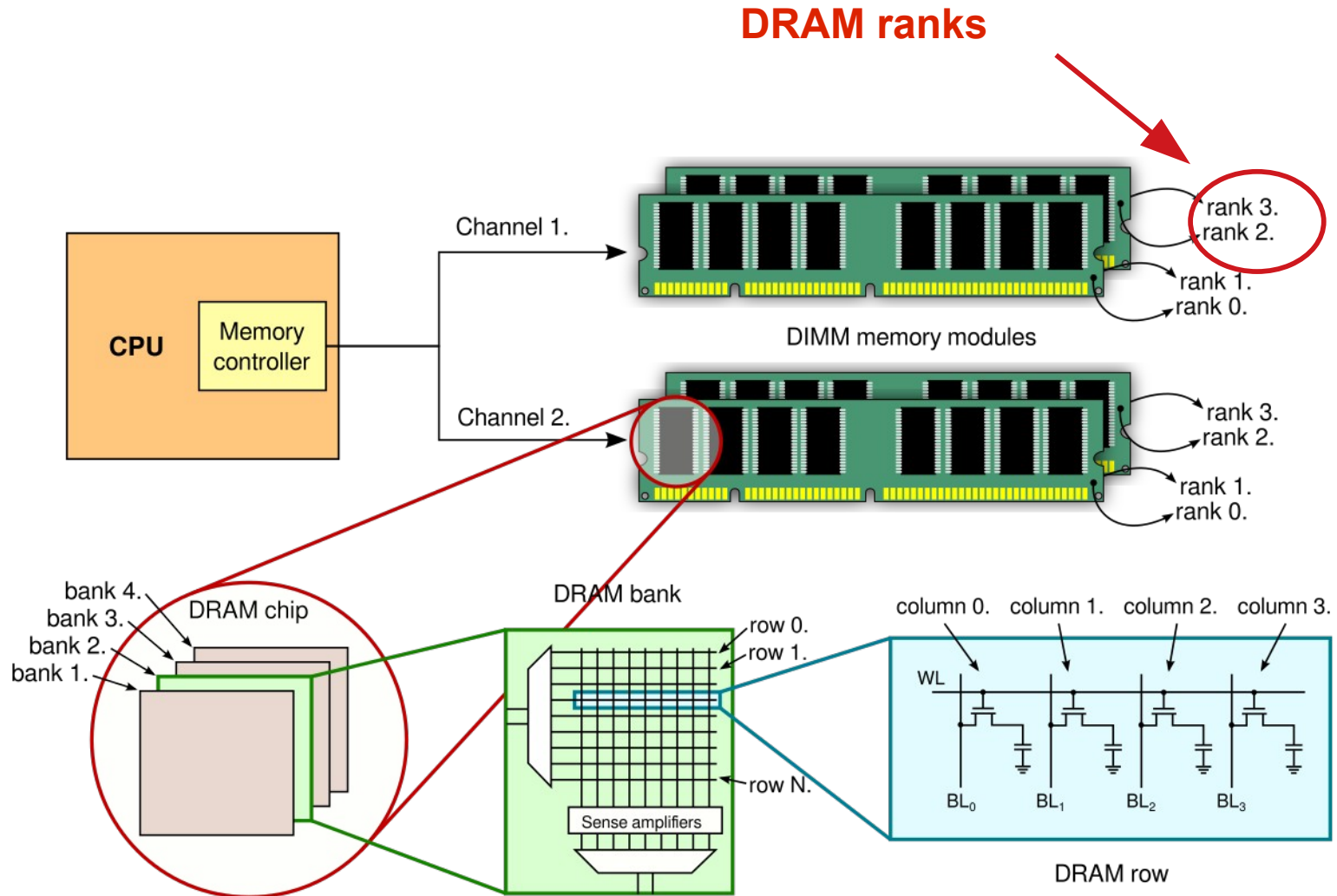
- To increase storage capacity
- Independent memory devices
- All lines are shared
  - ...but only a single rank can be enabled at a time → rank select lines
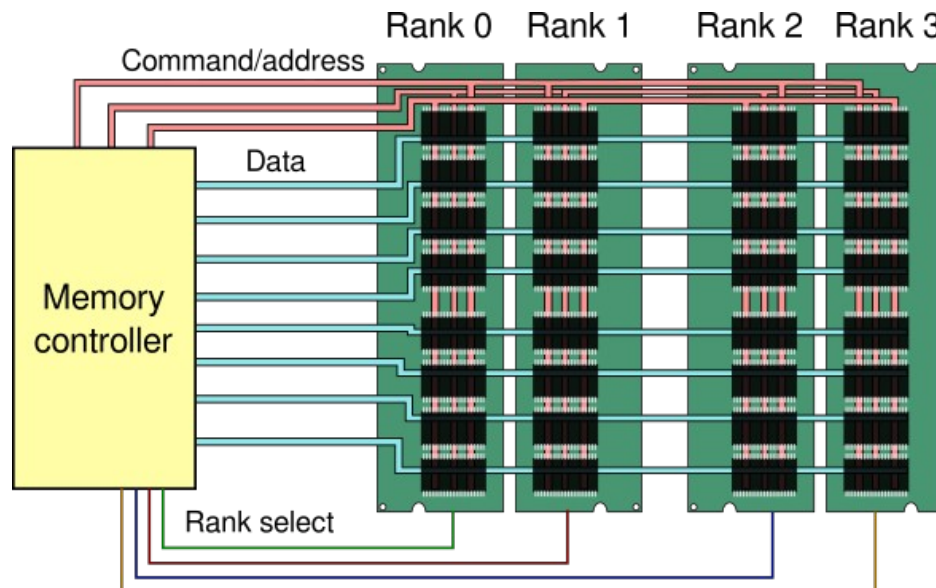
- Effect:
  - Throughput is the same as without multiple ranks
  - Delay: better (more banks, more open rows)

www.hit.bme.hu

DEPARTMENT OF
NETWORKED SYSTEMS
AND SERVICES

Memory channels

CPU

Memory controller

Channel 1.

Channel 2.

DIMM memory modules

rank 3.
rank 2.
rank 1.
rank 0.

rank 3.
rank 2.
rank 1.
rank 0.

bank 4.
bank 3.
bank 2.
bank 1.

DRAM chip

DRAM bank

row 0.
row 1.

row N.

Sense amplifiers

column 0. column 1. column 2. column 3.

WL

BL_0 BL_1 BL_2 BL_3

DRAM row
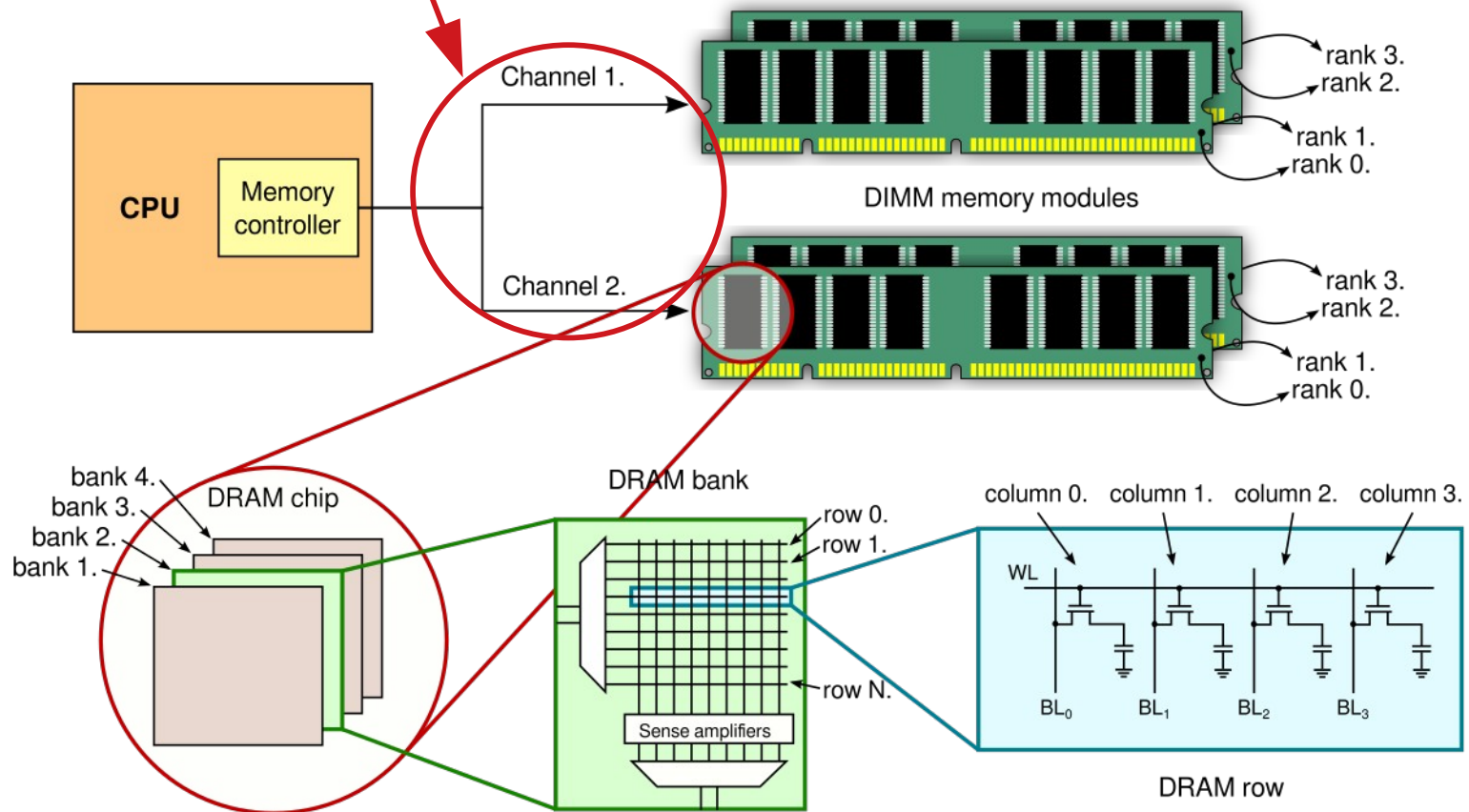
- The memory controller may have multiple channels
- **Synchronized** case:
  - If all modules are identical (size, timing, etc. are the same)
  - Operate in lockstep
    - 2x 64 bit wide channels → 1x 128 bit wide channel
- **Independent** channels
  - Modules don't have to be identical in different channels
  - Every channel has its own memory controller

www.hit.bme.hu

DEPARTMENT OF
NETWORKED SYSTEMS
AND SERVICES

Memory controller

CPU

Memory controller

Channel 1.

Channel 2.

rank 3.
rank 2.
rank 1.
rank 0.

DIMM memory modules

rank 3.
rank 2.
rank 1.
rank 0.

bank 4.
bank 3.
bank 2.
bank 1.

DRAM chip

DRAM bank

row 0.
row 1.

row N.

Sense amplifiers

column 0. column 1. column 2. column 3.

WL

BL_0 BL_1 BL_2 BL_3

DRAM row

www.hit.bme.hu

- Purpose:
  - It serves memory read/write requests (coming from the CPU and the I/O devices)
- Main tasks:
  - It translates the memory addresses to channel/rank/bank/row/column coordinates
  - Re-orders memory read/write requests
  - Open row management
  - Scheduling DRAM refresh commands

- Optimizing the order of read/write requests:
  - To minimize the slow row activation and precharge operations
    - First-Come-First-Serve scheduling (no optimization)
    - First-Response-First-Come-First-Serve scheduling (optimized)

FCFS

FR-FCFS

Requests:
```
(row 3, column 8)          (row 3, column 8)
(row 1, column 3)          (row 3, column 14)
(row 3, column 14)         (row 1, column 3)
```

Commands:
```
ACTIVATE 3                 ACTIVATE 3
READ 8                     READ 8
PRECHARGE                  READ 14
ACTIVATE 1                 PRECHARGE
READ 3                     ACTIVATE 1
PRECHARGE                  READ 3
ACTIVATE 3
READ 14
```

- Open row management
- All read/requests are served. What to do with the active row?
  - **Don't close it**:
    - If the next request falls to the same row, we don't have to open it (no extra delay)
    - If the next request falls to a different row, we have to close the current one before opening the next one (extra delay)
  - **Close it**:
    - If the next requests falls to the same row, we have to open it again (extra delay)
    - If the next request falls to a different row, we don't have to close the current one before opening the next one (no extra delay)
  - **Adaptive**:
    - Speculates
    - APM (Active Page Management)
      Core i7: „Adaptive Page Closing" option in the BIOS

Evolution of the DRAM technologies
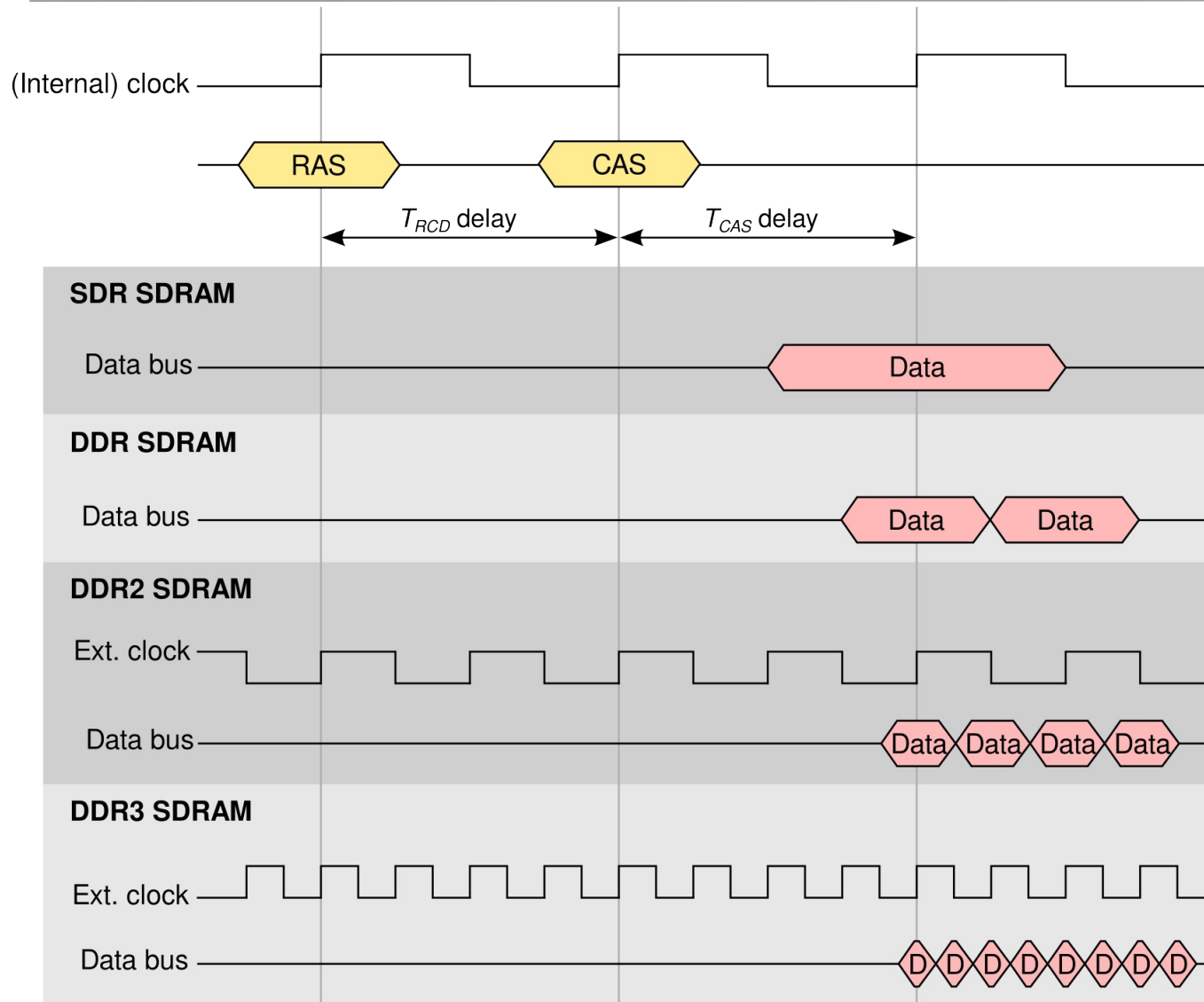
- Standard notation:
  - With the equivalent SDR clock frequency:
    - DDR-400, DDR2-800, DDR3-1600 clock freq.: 200 MHz for all!
    - **The latencies of the command are the same!!!**
    - … but the data transmission speed (throughput) is faster
  - With the throughput:
    - e.g., DDR2-800: data transmission at 800 MHz (data unit: 8 byte)
      → PC2-6400 (800x8 = 6400)
    - e.g., DDR3-1600: data transmission at 1600 MHz (data unit: 8 byte)
      → PC3-12800 (1600x8 = 12800)

- Warning! This is just an example. The clock frequency is not always 200 MHz!

- **The ratio between the external ↔ internal clock (burst length) has not been increased further**
- Increasing the performance:
  - **The internal clock frequency is higher.** To make it possible:
    - Voltage is lower (1.2V)
      → lower power consumption, lower heat dissipation
    - Bus frequency is higher, crosstalk is getting worse, signal shifts are getting worse
      → CRC (for the data) / parity bit (for the address and the command) protection needed
        calibration is necessary
    - There are 16 banks instead of 8 (greater level of parallelism)
    - Bank groups were introduced, more complicated timing
  - The memory modules have higher capacity. To make it possible::
    - At most 4 ranks / module
    - Banks are not square shaped any more
      → more row bits than columns bits → command signals are multiplexed with the higher address bits (to get fewer number of pins)
  - Four channel memory controllers are supported

- Announced: 2020
- **Burst length increases to 16**
- Imrpoved performance:
  - Lower voltage (1.1V)
  - Number of banks: 2x
  - **2 channels for each module** (64 bit → 32+32 bit data unit)
  - Twice the burst length with half the width → 64 byte/burst
    - → Throughput is about the same
    - → Better latency

- Conclusion
  - There are serious latency problems
    - Internal clock rate is almost the same in the last 10-15 years

      → **latency is the same**

      (latency: delay between the address and the corresponding data)
    - Not critical for GPUs
    - Getting critical for CPUs
  - Data units transferred / clock cycle increased significantly

    → **throughput is improving**

    (Throughput: amount of data transmitted / second)