iisys
Institute for
Information Systems
at Hof University

Reverse-Engineering Bank Addressing Functions on AMD CPUs

Presentation for DRAMSec

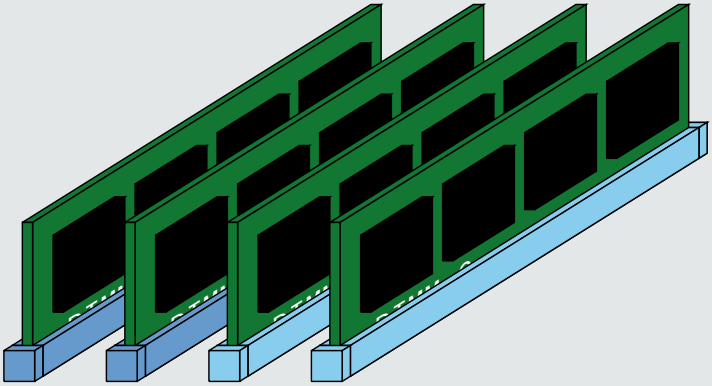
Martin Heckel, Florian Adamsky

June 17, 2023

Introduction

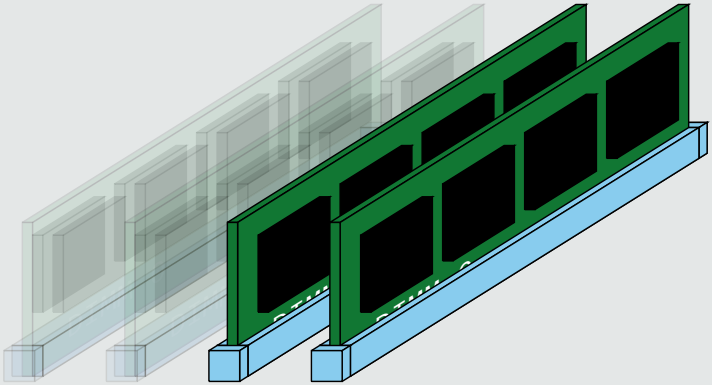
Physical DRAM architecture

System DRAM



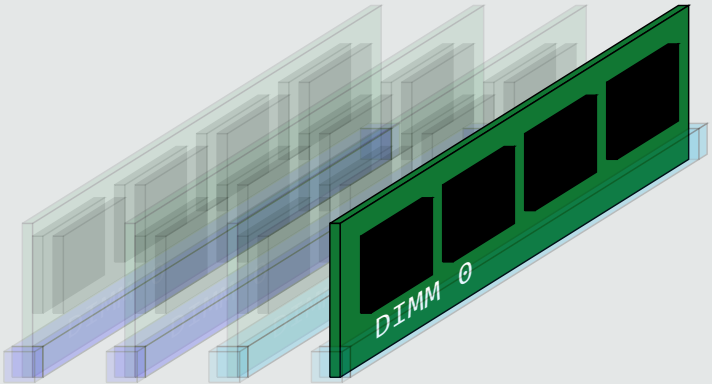
Physical DRAM architecture

Channel



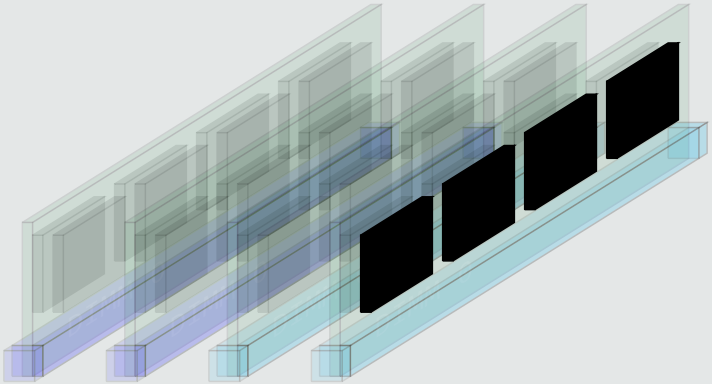
Physical DRAM architecture

DIMM



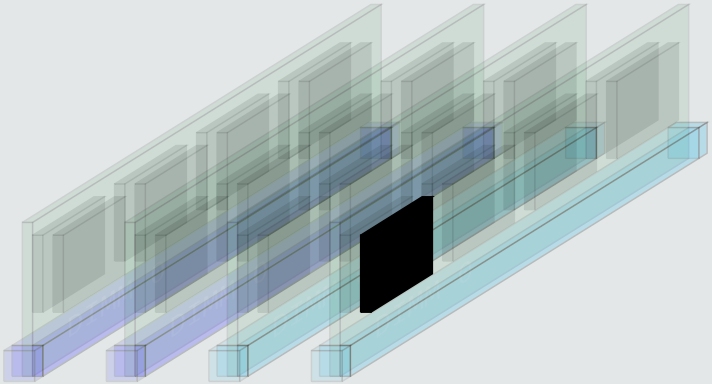
Physical DRAM architecture

Rank



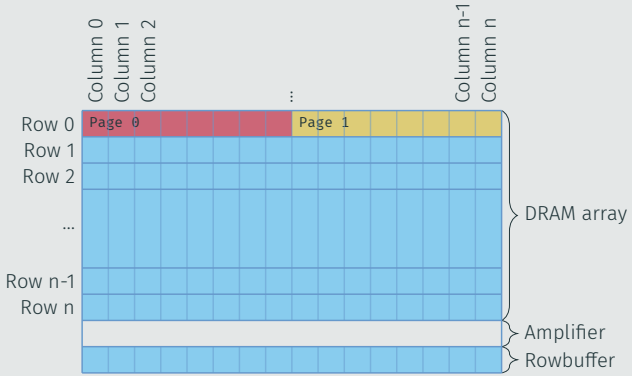
Physical DRAM architecture

Bank



Physical DRAM architecture

Within a bank



DRAM Bank Addressing Functions

- Physical address space is distributed over all DRAM banks

DRAM Bank Addressing Functions

- Physical address space is distributed over all DRAM banks
- DRAM addressing functions implemented in the memory controller determine the bank a physical address is stored on

DRAM Bank Addressing Functions

- Physical address space is distributed over all DRAM banks
- DRAM addressing functions implemented in the memory controller determine the bank a physical address is stored on
- AMD had published these addressing functions in the BKDG until micro architecture 16h, Intel did not publish them

Example for DRAM Bank Addressing Functions

f_1 : `0000001000000000000`

f_2 : `0100100000000000000`

f_3 : `0010010000000000000`

f_4 : `1001000000000000000`

Example for DRAM Bank Addressing Functions

Physical Address: 0x00000

$A = 00000000000000000000$

f_1 : 00000010000000000000

f_2 : 01001000000000000000

f_3 : 00100100000000000000

f_4 : 10010000000000000000

Example for DRAM Bank Addressing Functions

Physical Address: 0x00000

$A =$ 00000000000000000000

$f_1:$ 00000010000000000000

$\oplus(f_1 \wedge A) = \oplus(00000000000000000000) = 0$

$f_2:$ 01001000000000000000

$f_3:$ 00100100000000000000

$f_4:$ 10010000000000000000

Example for DRAM Bank Addressing Functions

Physical Address: 0x00000

$A =$ 00000000000000000000

f_1 : 00000010000000000000

$\oplus(f_1 \wedge A) = \oplus(00000000000000000000) = 0$

f_2 : 01001000000000000000

$\oplus(f_2 \wedge A) = \oplus(00000000000000000000) = 0$

f_3 : 00100100000000000000

f_4 : 10010000000000000000

Example for DRAM Bank Addressing Functions

Physical Address: 0x00000

$A = 00000000000000000000$

f_1 : 00000010000000000000

$\oplus(f_1 \wedge A) = \oplus(00000000000000000000) = 0$

f_2 : 01001000000000000000

$\oplus(f_2 \wedge A) = \oplus(00000000000000000000) = 0$

f_3 : 00100100000000000000

$\oplus(f_3 \wedge A) = \oplus(00000000000000000000) = 0$

f_4 : 10010000000000000000

Example for DRAM Bank Addressing Functions

Physical Address: 0x00000

$A = 00000000000000000000$

f_1 : 00000010000000000000

$\oplus(f_1 \wedge A) = \oplus(00000000000000000000) = 0$

f_2 : 01001000000000000000

$\oplus(f_2 \wedge A) = \oplus(00000000000000000000) = 0$

f_3 : 00100100000000000000

$\oplus(f_3 \wedge A) = \oplus(00000000000000000000) = 0$

f_4 : 10010000000000000000

$\oplus(f_4 \wedge A) = \oplus(00000000000000000000) = 0$

Example for DRAM Bank Addressing Functions

Physical Address: 0x00000

$A = 00000000000000000000$

f_1 : 00000010000000000000

$\oplus(f_1 \wedge A) = \oplus(00000000000000000000) = 0$

f_2 : 01001000000000000000

$\oplus(f_2 \wedge A) = \oplus(00000000000000000000) = 0$

f_3 : 00100100000000000000

$\oplus(f_3 \wedge A) = \oplus(00000000000000000000) = 0$

f_4 : 10010000000000000000

$\oplus(f_4 \wedge A) = \oplus(00000000000000000000) = 0$

0000 $\leftrightarrow$ 0

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x01000

$A =$ 00000001000000000000

f_1 : 00000010000000000000

f_2 : 01001000000000000000

f_3 : 00100100000000000000

f_4 : 10010000000000000000

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x01000

$A =$ 0000000100000000000

$f_1:$ 0000010000000000000

$\oplus(f_1 \wedge A) = \oplus(0000000000000000000) =$ 0

$f_2:$ 0100100000000000000

$f_3:$ 0010010000000000000

$f_4:$ 1001000000000000000

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x01000

$A =$ 0000000100000000000

$f_1:$ 0000010000000000000

$\oplus(f_1 \wedge A) = \oplus(0000000000000000000) =$ 0

$f_2:$ 0100100000000000000

$\oplus(f_2 \wedge A) = \oplus(0000000000000000000) =$ 0

$f_3:$ 0010010000000000000

$f_4:$ 1001000000000000000

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x01000

$A =$ 0000000100000000000

f_1 : 00000010000000000000

$\oplus(f_1 \wedge A) = \oplus(00000000000000000000) = 0$

f_2 : 01001000000000000000

$\oplus(f_2 \wedge A) = \oplus(00000000000000000000) = 0$

f_3 : 00100100000000000000

$\oplus(f_3 \wedge A) = \oplus(00000000000000000000) = 0$

f_4 : 10010000000000000000

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x01000

$A =$ 0000000100000000000

$$f_1: \quad 0000001000000000000$$
$$\oplus(f_1 \wedge A) = \oplus(0000000000000000000) = 0$$

$$f_2: \quad 0100100000000000000$$
$$\oplus(f_2 \wedge A) = \oplus(0000000000000000000) = 0$$

$$f_3: \quad 0010010000000000000$$
$$\oplus(f_3 \wedge A) = \oplus(0000000000000000000) = 0$$

$$f_4: \quad 1001000000000000000$$
$$\oplus(f_4 \wedge A) = \oplus(0000000000000000000) = 0$$

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x01000

$A =$ 0000000100000000000

f_1 : 00000010000000000000

$\oplus(f_1 \wedge A) = \oplus(00000000000000000000) = 0$

f_2 : 01001000000000000000

$\oplus(f_2 \wedge A) = \oplus(00000000000000000000) = 0$

f_3 : 00100100000000000000

$\oplus(f_3 \wedge A) = \oplus(00000000000000000000) = 0$

f_4 : 10010000000000000000

$\oplus(f_4 \wedge A) = \oplus(00000000000000000000) = 0$

0000 $\leftrightarrow$ 0

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Physical Address 0x01000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x02000

$A =$ 0000001000000000000

f_1 : 0000001000000000000

f_2 : 0100100000000000000

f_3 : 0010010000000000000

f_4 : 1001000000000000000

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Physical Address 0x01000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x02000

$A =$ 0000001000000000000

$f_1:$ 0000001000000000000

$\oplus(f_1 \wedge A) = \oplus(0000001000000000000) = 1$

$f_2:$ 0100100000000000000

$f_3:$ 0010010000000000000

$f_4:$ 1001000000000000000

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Physical Address 0x01000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x02000

$A =$ 0000001000000000000

f_1 : 0000001000000000000

$\oplus(f_1 \wedge A) = \oplus(0000001000000000000) = 1$

f_2 : 0100100000000000000

$\oplus(f_2 \wedge A) = \oplus(0000000000000000000) = 0$

f_3 : 0010010000000000000

f_4 : 1001000000000000000

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Physical Address 0x01000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x02000

$A =$ 0000001000000000000

f_1 : 0000001000000000000

$\oplus(f_1 \wedge A) = \oplus(0000001000000000000) = 1$

f_2 : 0100100000000000000

$\oplus(f_2 \wedge A) = \oplus(0000000000000000000) = 0$

f_3 : 0010010000000000000

$\oplus(f_3 \wedge A) = \oplus(0000000000000000000) = 0$

f_4 : 1001000000000000000

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Physical Address 0x01000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x02000

$A =$ 0000001000000000000

f_1 : 0000001000000000000

$\oplus(f_1 \wedge A) = \oplus(0000001000000000000) = 1$

f_2 : 0100100000000000000

$\oplus(f_2 \wedge A) = \oplus(0000000000000000000) = 0$

f_3 : 0010010000000000000

$\oplus(f_3 \wedge A) = \oplus(0000000000000000000) = 0$

f_4 : 1001000000000000000

$\oplus(f_4 \wedge A) = \oplus(0000000000000000000) = 0$

Physical Address 0x00000 $\leftrightarrow$ Bank 0

Physical Address 0x01000 $\leftrightarrow$ Bank 0

Example for DRAM Bank Addressing Functions

Physical Address: 0x02000

$A =$ 0000001000000000000

f_1 : 0000001000000000000

$\oplus(f_1 \wedge A) = \oplus(0000001000000000000) = 1$

f_2 : 0100100000000000000

$\oplus(f_2 \wedge A) = \oplus(0000000000000000000) = 0$

f_3 : 0010010000000000000

$\oplus(f_3 \wedge A) = \oplus(0000000000000000000) = 0$

f_4 : 1001000000000000000

$\oplus(f_4 \wedge A) = \oplus(0000000000000000000) = 0$

1000 $\leftrightarrow$ 8

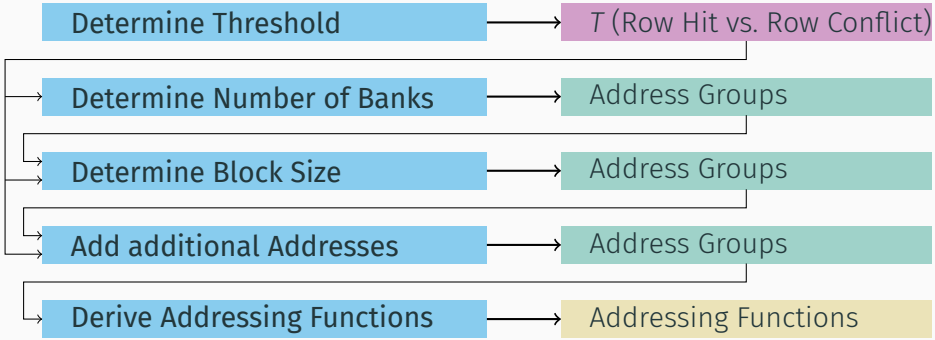
Physical Address 0x00000 $\leftrightarrow$ Bank 0

Physical Address 0x01000 $\leftrightarrow$ Bank 0

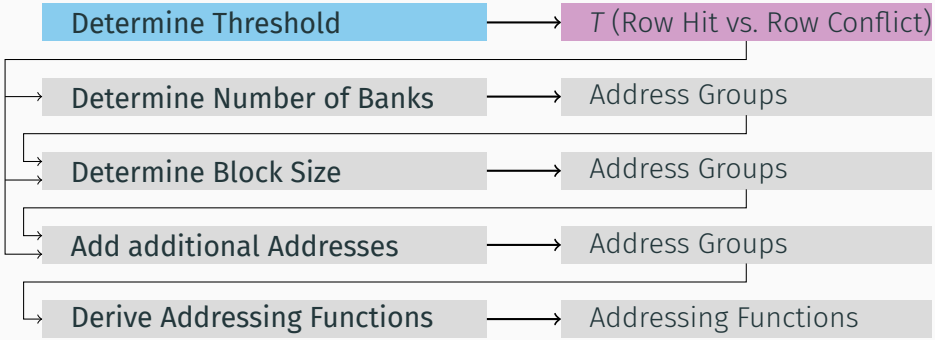
Physical Address 0x02000 $\leftrightarrow$ Bank 8

Addressing Function Reverse-Engineering on AMD CPUs

Overview of the Reverse-Engineering Process



Determine Threshold between Row Hit and Row Conflict



Determine Threshold between Row Hit and Row Conflict

- Allocate a 2 MiB transparent hugepage

Determine Threshold between Row Hit and Row Conflict

- Allocate a 2 MiB transparent hugepage
- Access the first and the n^{th} address 200 times alternatingly (n is a multiple of 4096) and measure the access time

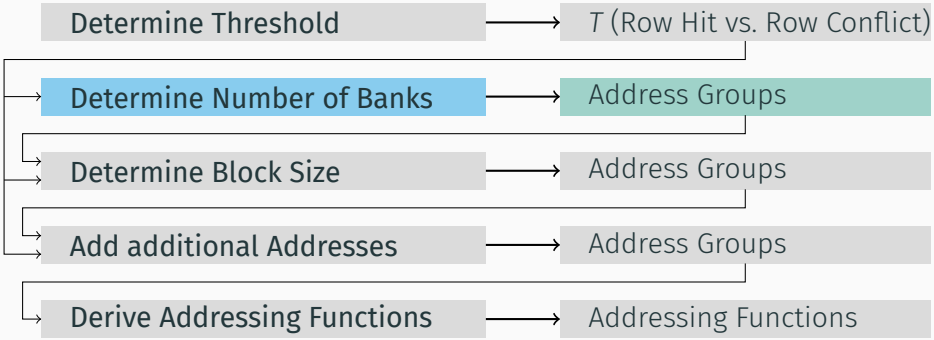
Determine Threshold between Row Hit and Row Conflict

- Allocate a 2 MiB transparent hugepage
- Access the first and the n^{th} address 200 times alternatingly (n is a multiple of 4096) and measure the access time
- Generate a histogram of the access times and search for a gap between row hits and row conflicts

Determine Threshold between Row Hit and Row Conflict

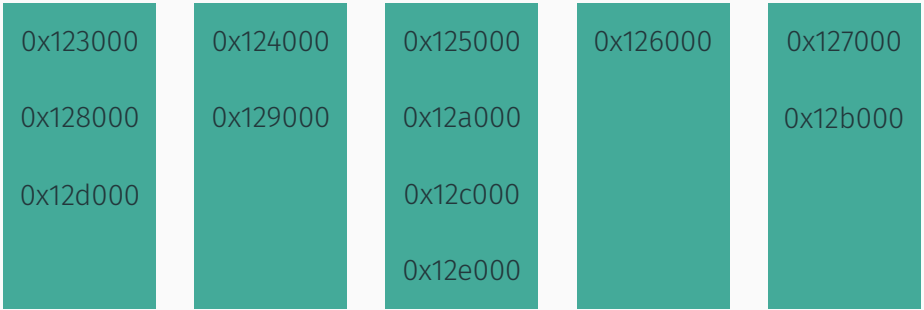
- Allocate a 2 MiB transparent hugepage
- Access the first and the n^{th} address 200 times alternatingly (n is a multiple of 4096) and measure the access time
- Generate a histogram of the access times and search for a gap between row hits and row conflicts
- Use the middle of the gap as threshold

Determine Number of Banks



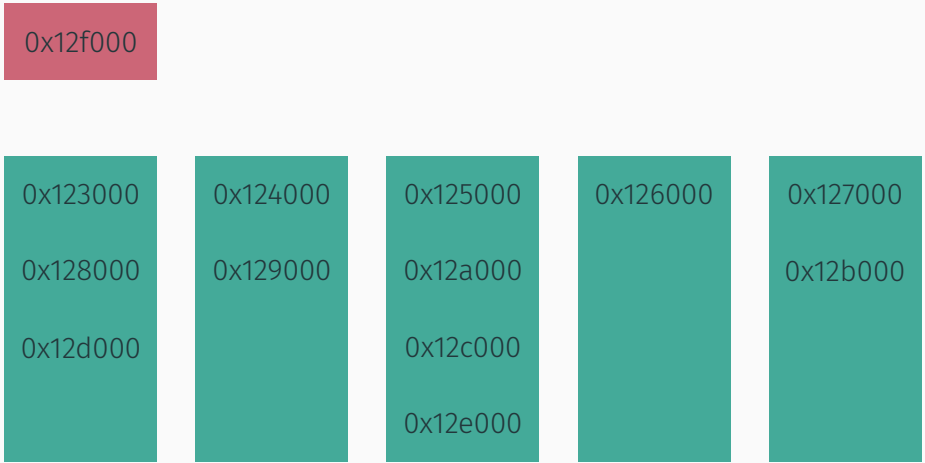
Determine Number of Banks

$t_h = 550$



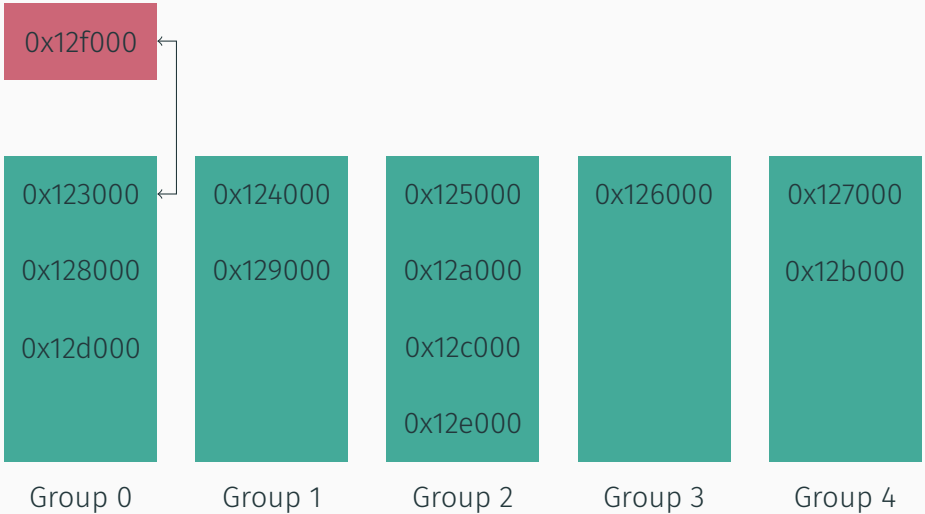
Determine Number of Banks

$t_h = 550$



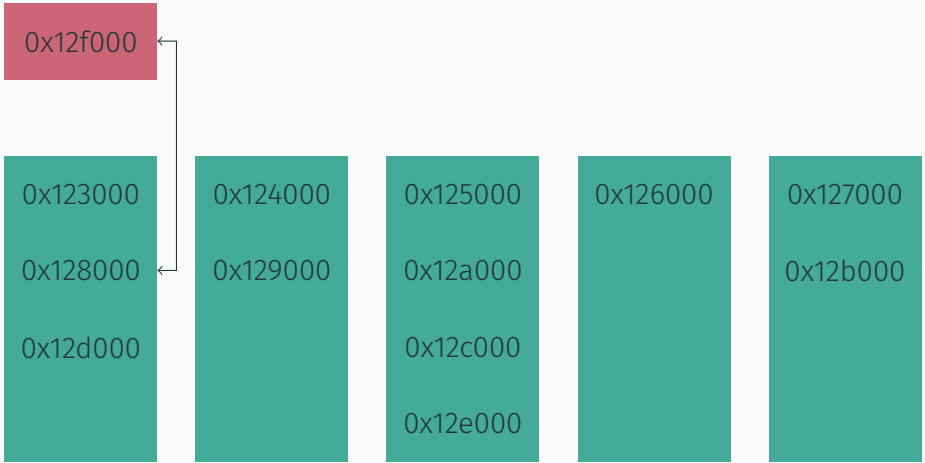
Determine Number of Banks

$t_h = 550$



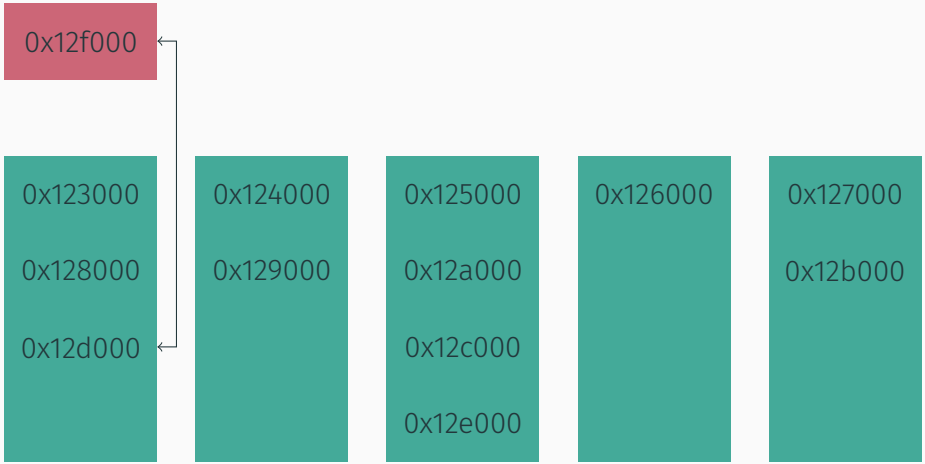
Determine Number of Banks

$t_h = 550$

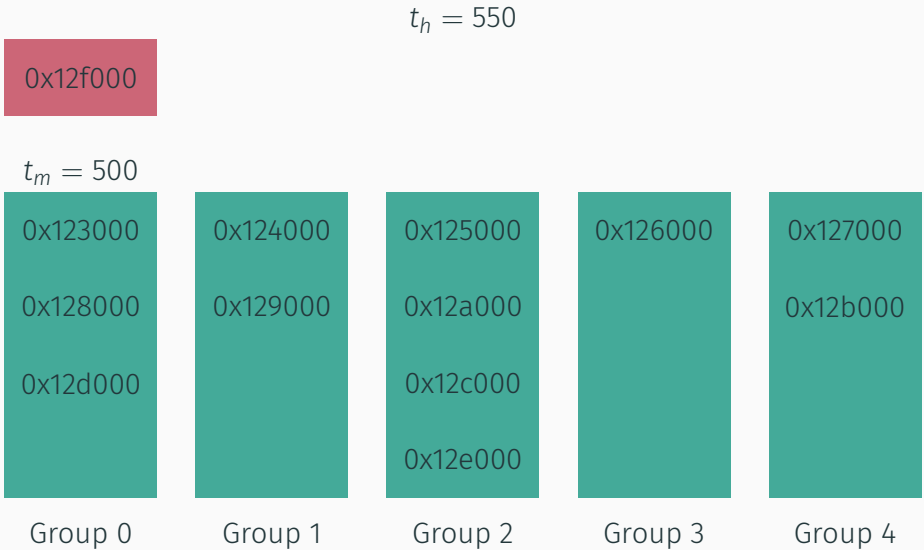


Determine Number of Banks

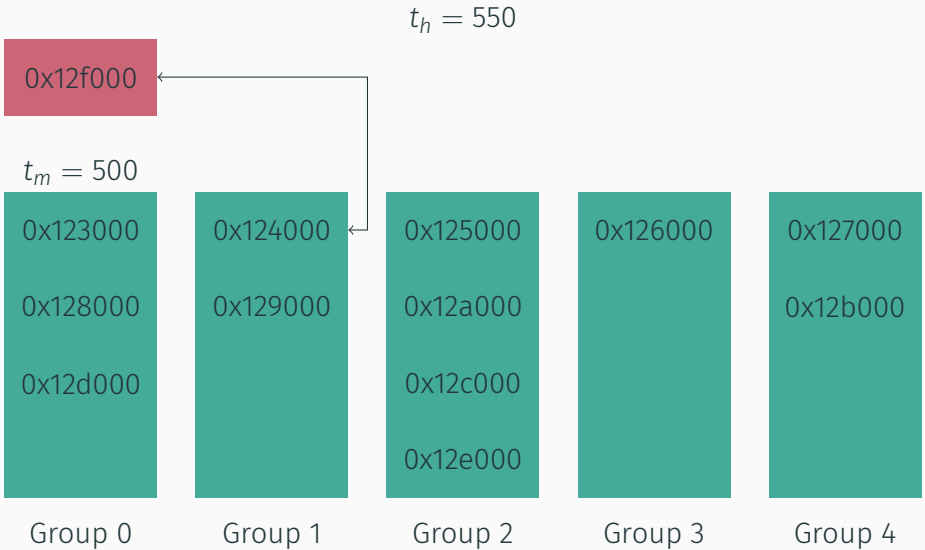
$t_h = 550$



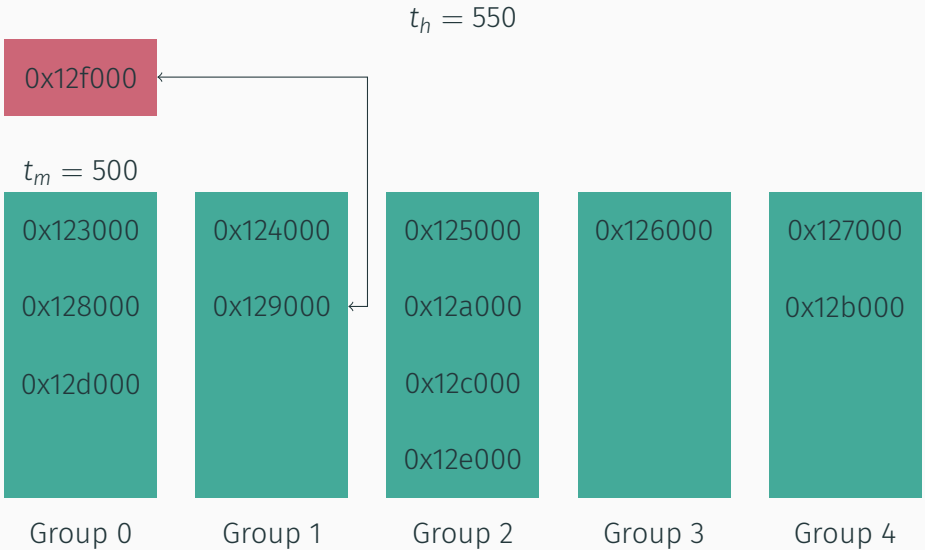
Determine Number of Banks



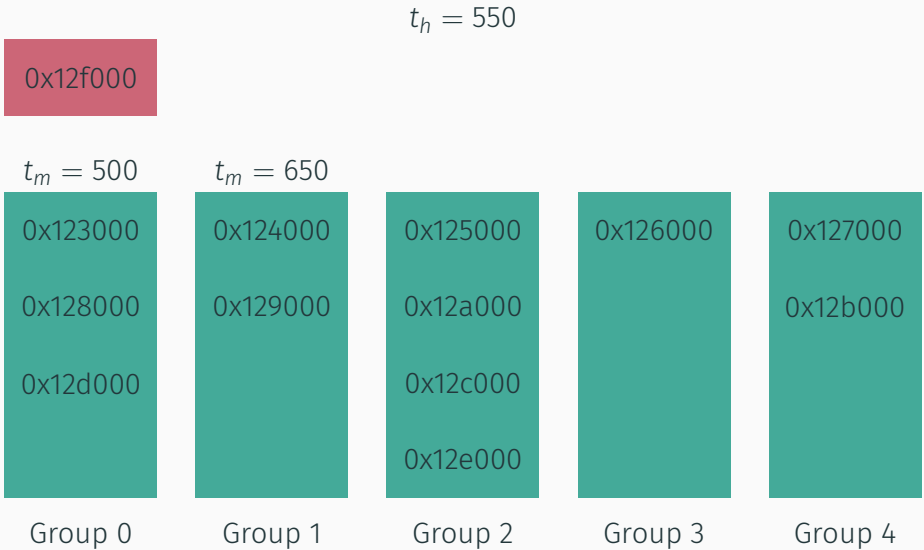
Determine Number of Banks



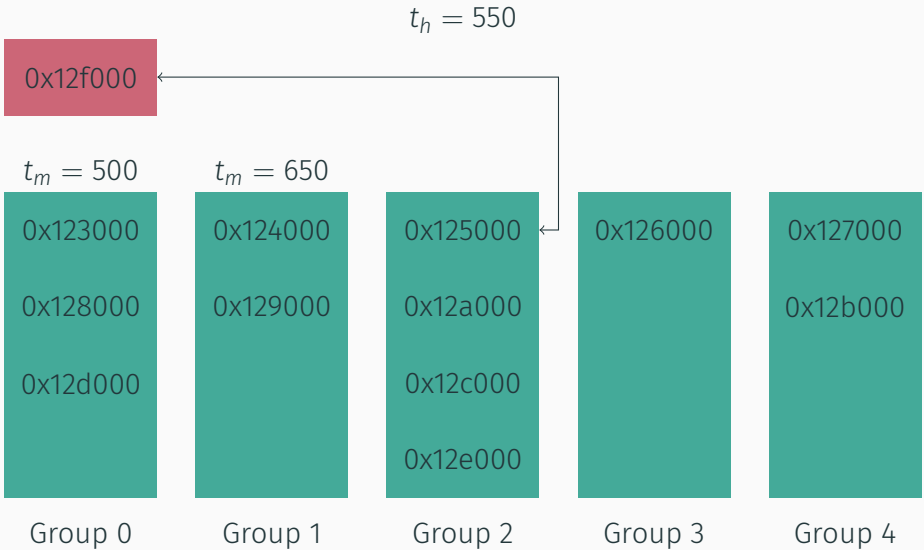
Determine Number of Banks



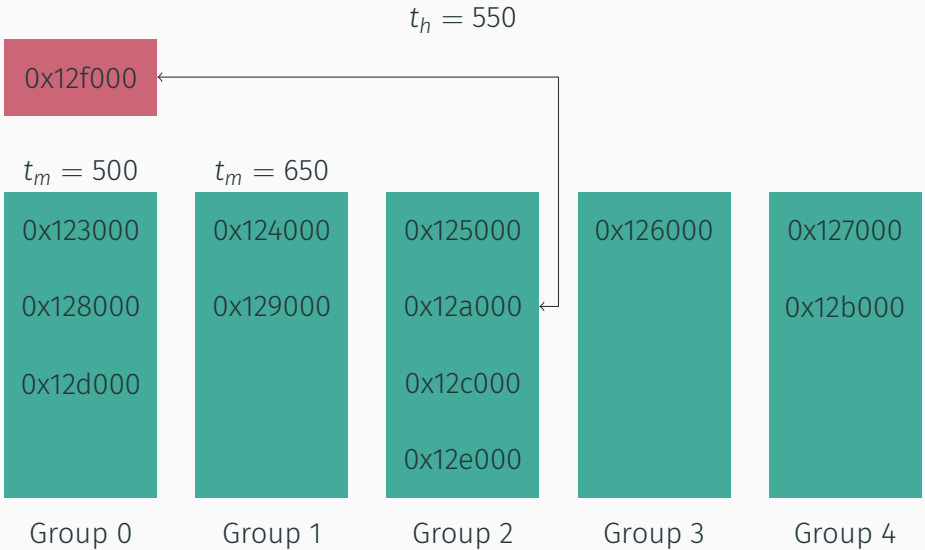
Determine Number of Banks



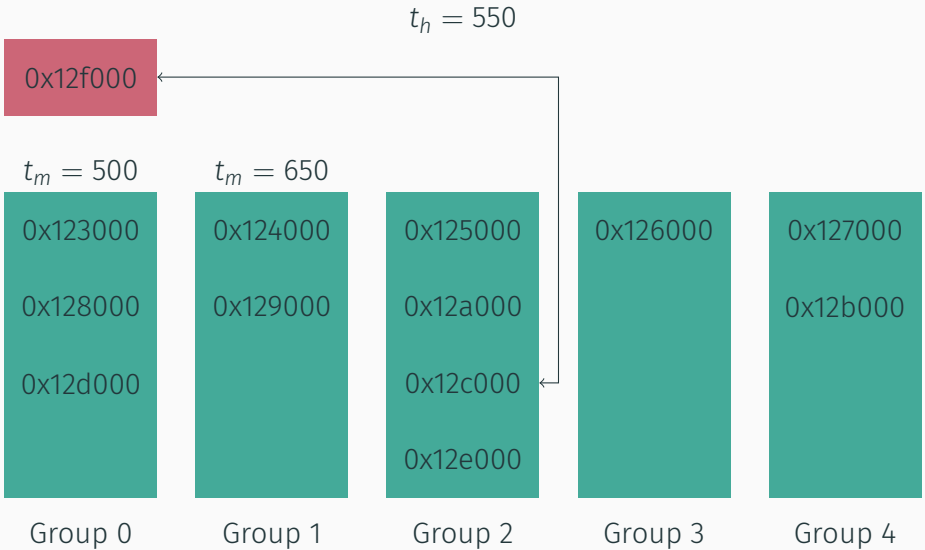
Determine Number of Banks



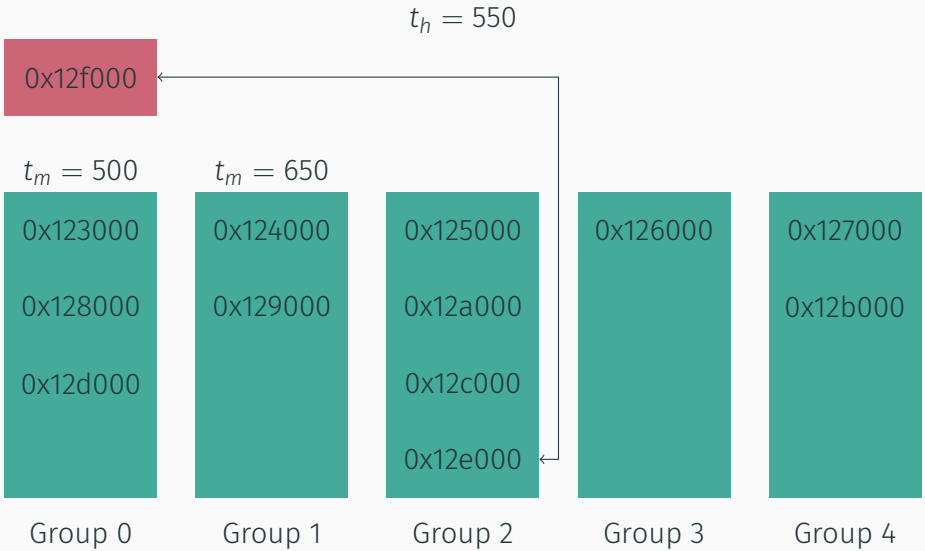
Determine Number of Banks



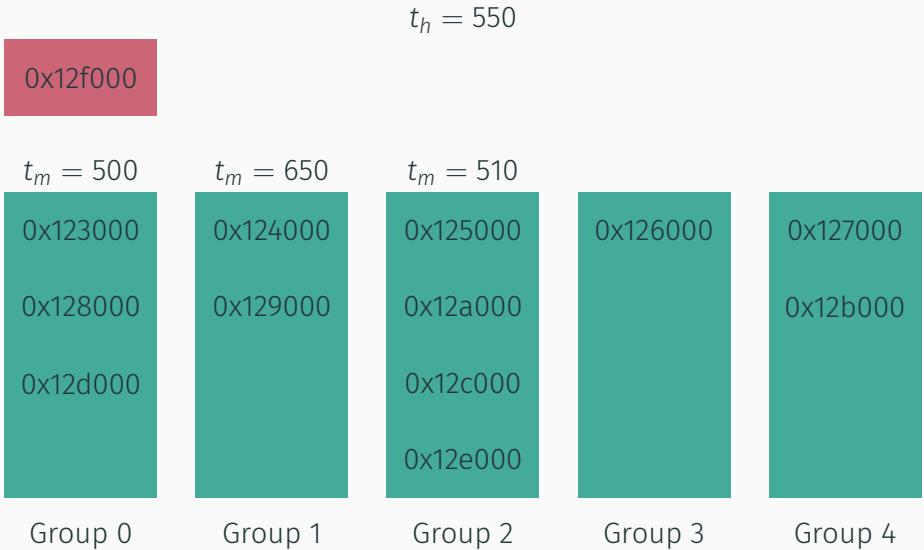
Determine Number of Banks



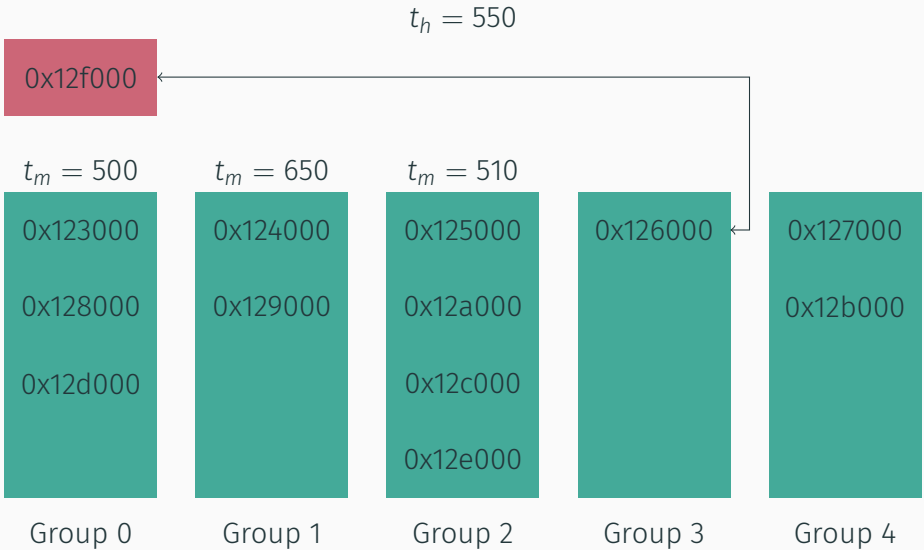
Determine Number of Banks



Determine Number of Banks

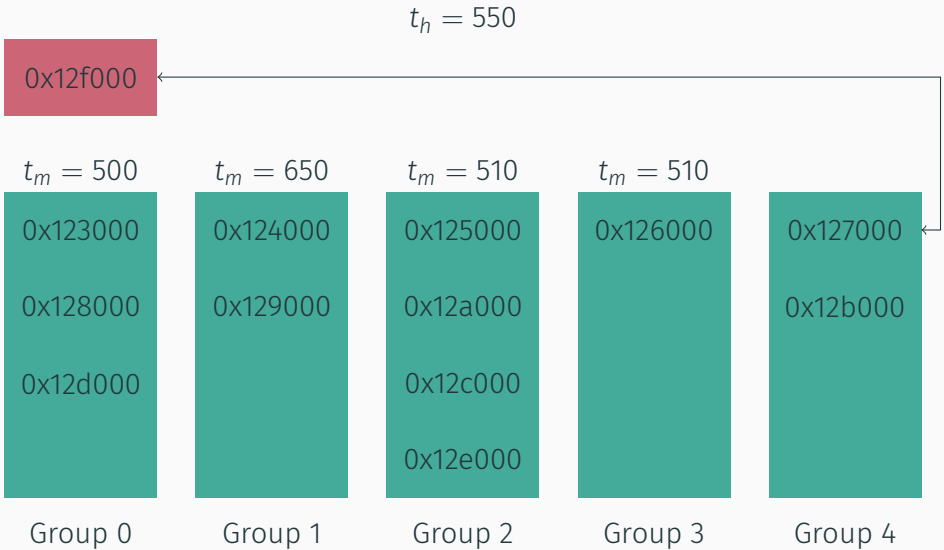


Determine Number of Banks

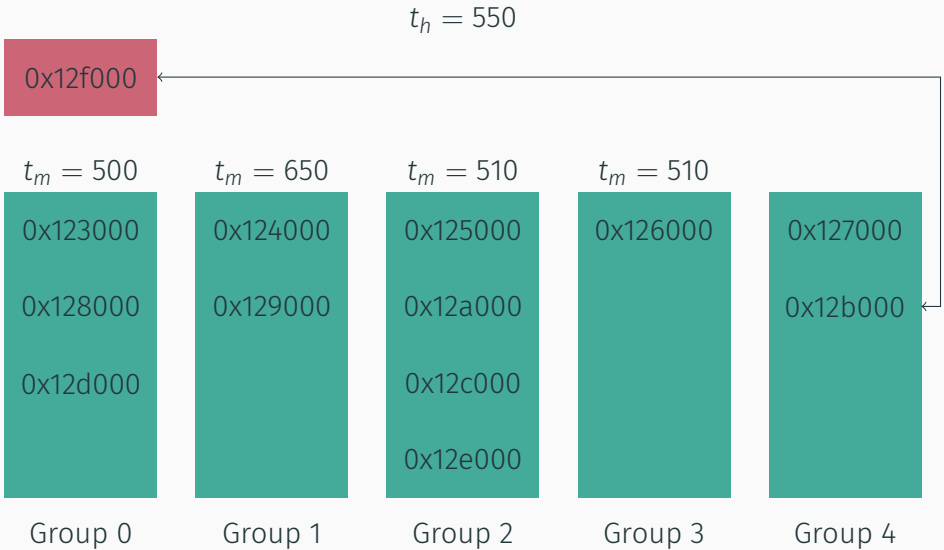


Determine Number of Banks

Determine Number of Banks



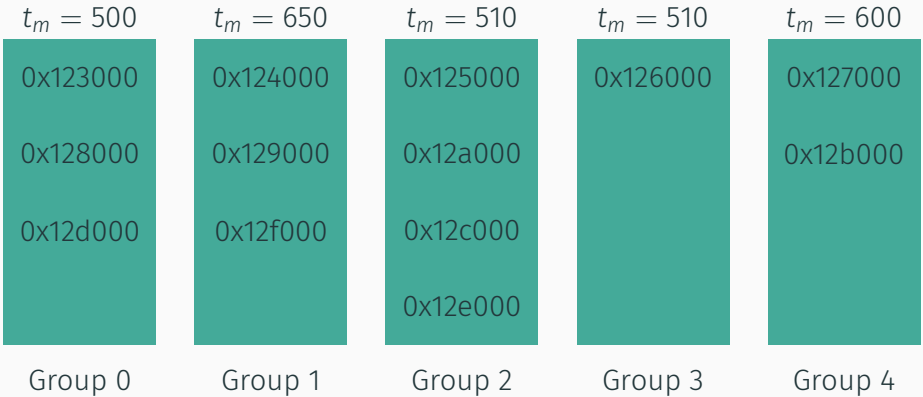
Determine Number of Banks



Determine Number of Banks

Determine Number of Banks

$t_h = 550$



Determine Number of Banks (regroup)

- There can be multiple pages in the same DRAM row, which results in a row hit if they are on the same bank

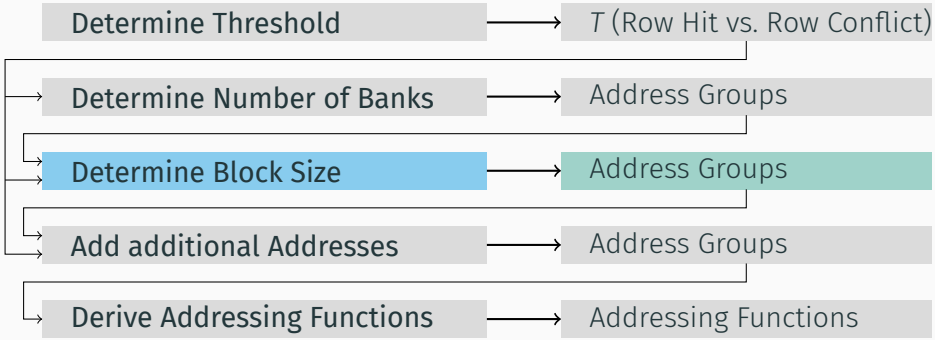
Determine Number of Banks (regroup)

- There can be multiple pages in the same DRAM row, which results in a row hit if they are on the same bank
- The groups generated before are removed one after another and each address of the removed group is grouped again (using all other groups)

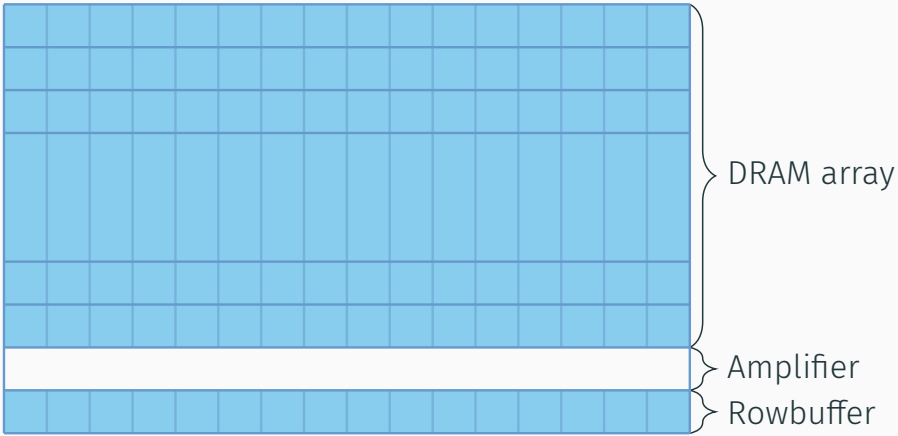
Determine Number of Banks (regroup)

- There can be multiple pages in the same DRAM row, which results in a row hit if they are on the same bank
- The groups generated before are removed one after another and each address of the removed group is grouped again (using all other groups)
- This removes groups that were created due to row hits on the same bank in the step before (since there were fewer addresses in the groups before)

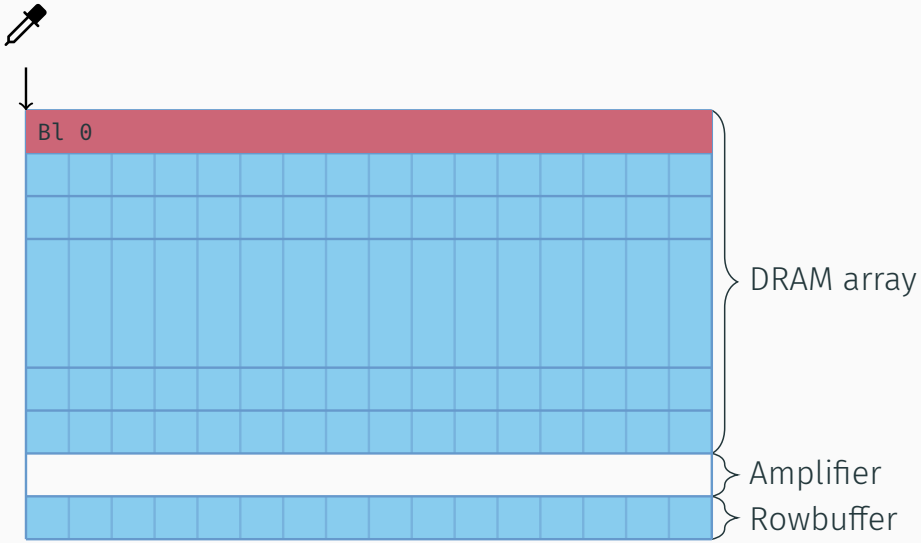
Determine Block Size



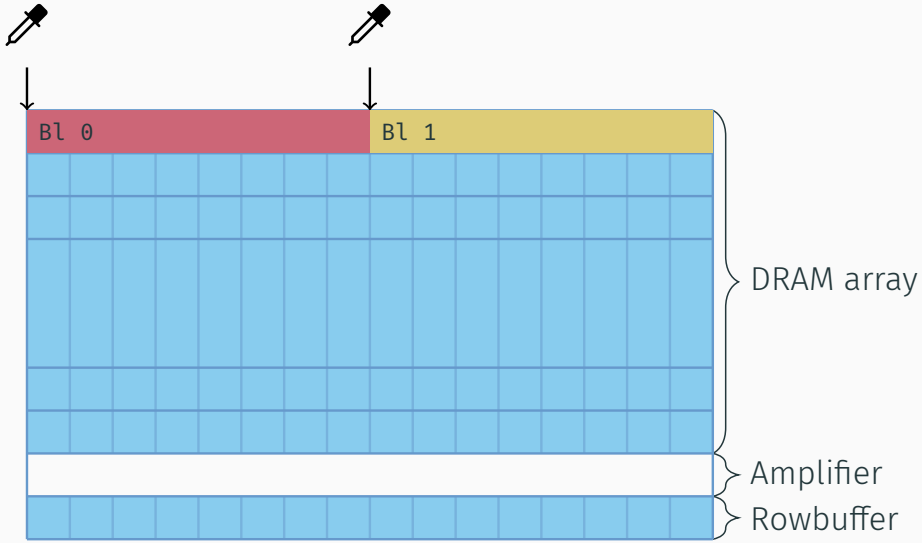
Determine Block Size



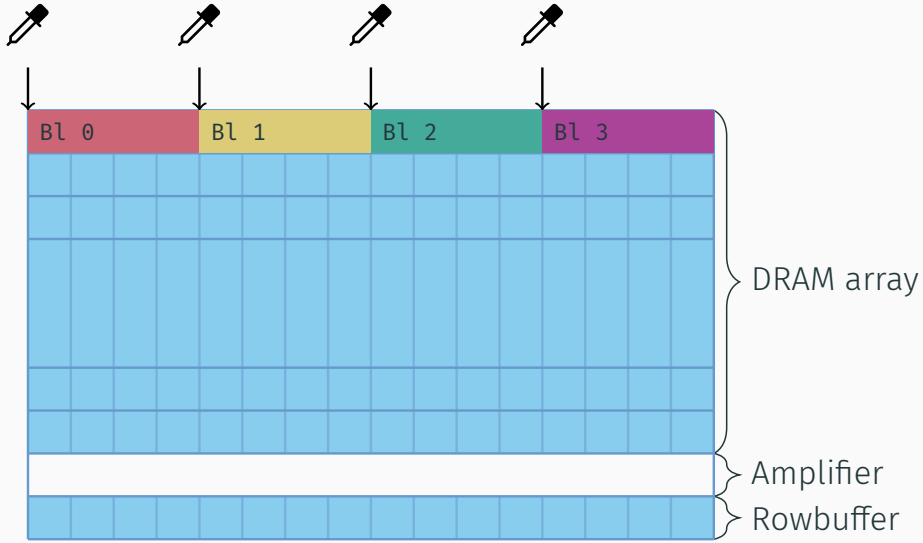
Determine Block Size



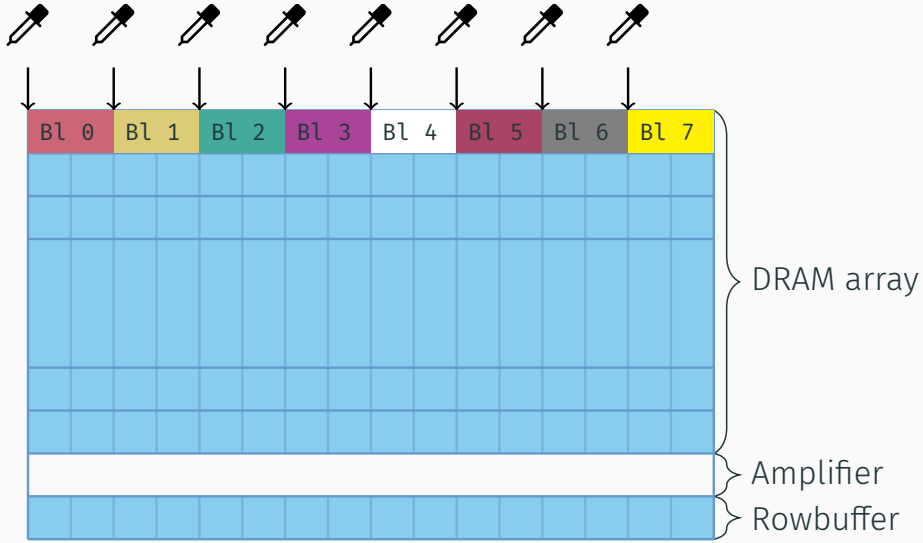
Determine Block Size



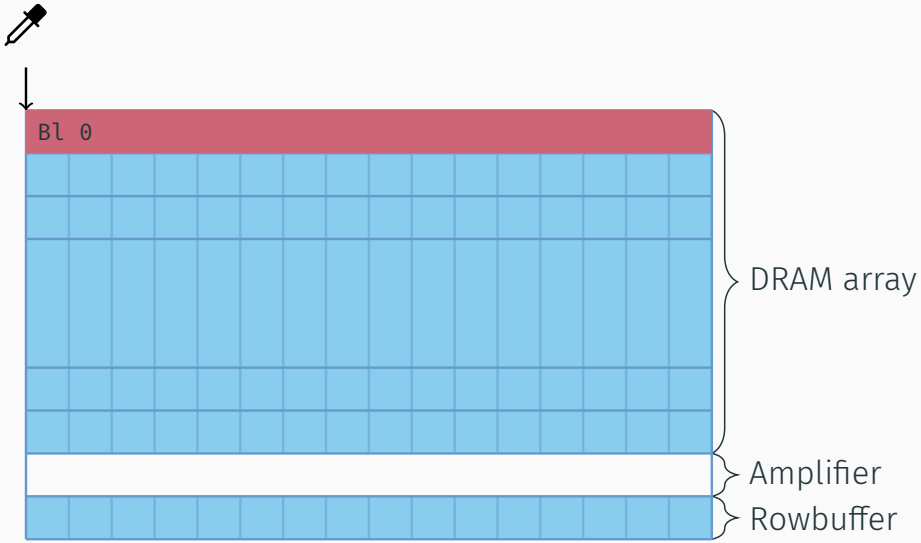
Determine Block Size



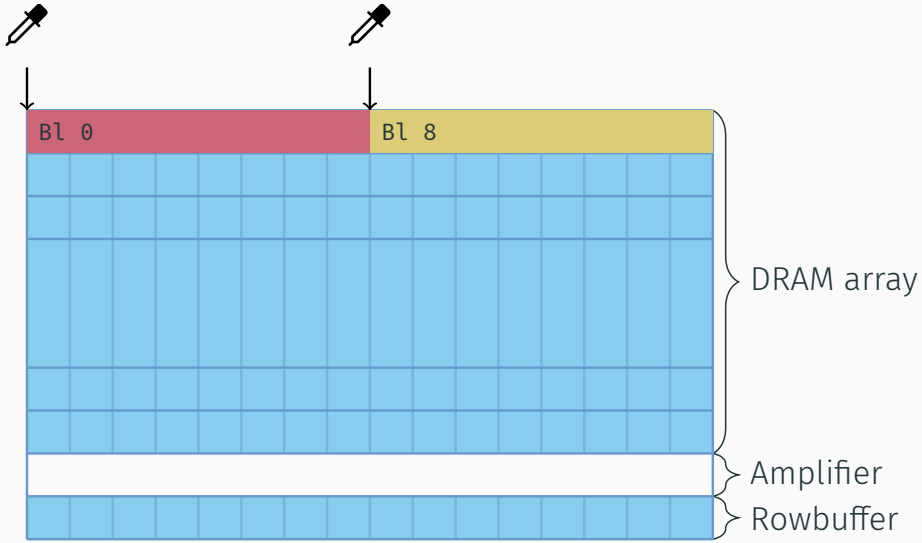
Determine Block Size



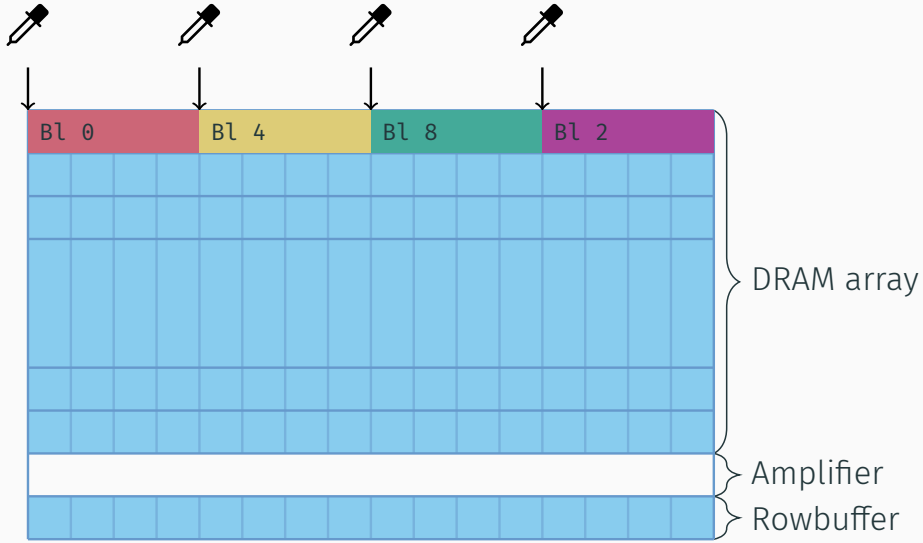
Determine Block Size



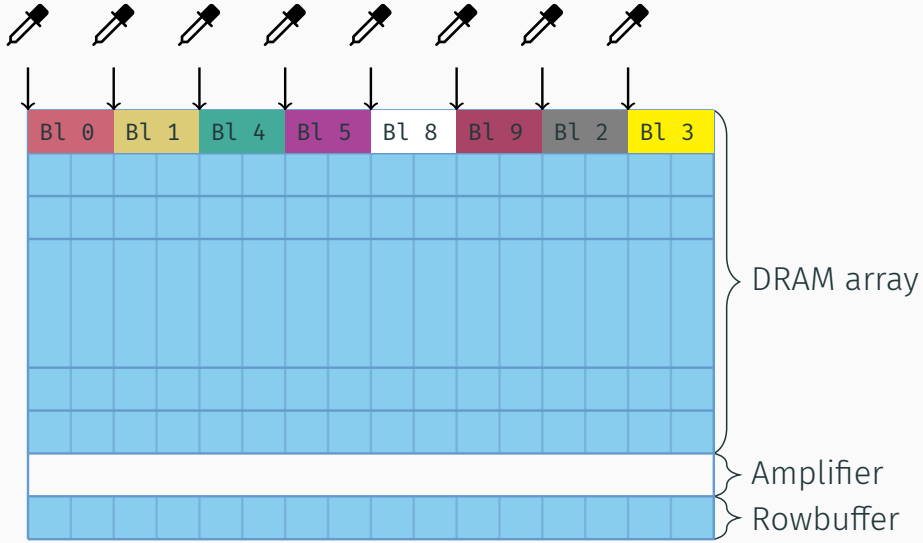
Determine Block Size



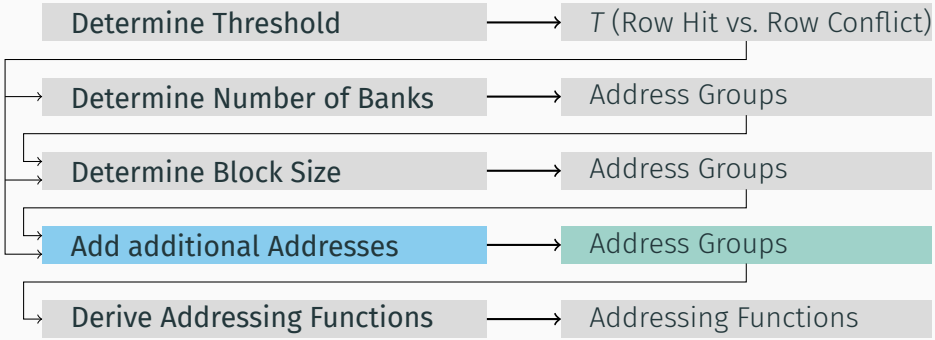
Determine Block Size



Determine Block Size



Add additional Addresses



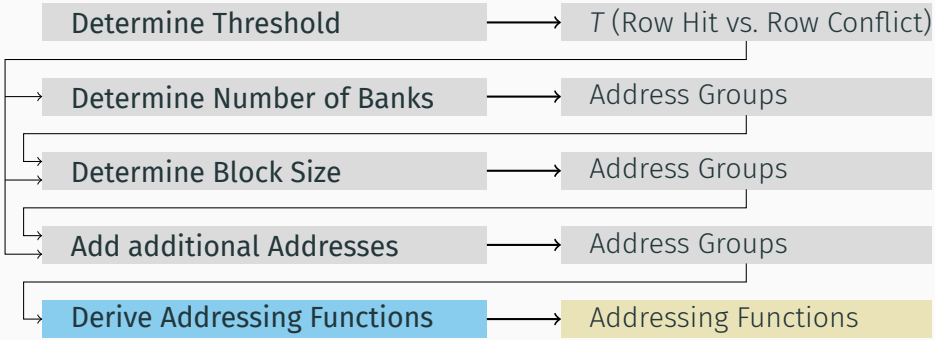
Add additional Addresses

- A number of transparent hugepages is allocated

Add additional Addresses

- A number of transparent hugepages is allocated
- The addresses of the transparent hugepages are added to the groups based on the block size determined before

Derive Addressing Functions



Derive Addressing Functions

- For each possible addressing function, the following requirements have to be met in order to add the function as valid:

Derive Addressing Functions

- For each possible addressing function, the following requirements have to be met in order to add the function as valid:
  - Yields the same results for all addresses within the same group

Derive Addressing Functions

- For each possible addressing function, the following requirements have to be met in order to add the function as valid:
  - Yields the same results for all addresses within the same group
  - Has a 1 as result for 50 % of the groups (0 for the other 50 %)

Derive Addressing Functions

- For each possible addressing function, the following requirements have to be met in order to add the function as valid:
  - Yields the same results for all addresses within the same group
  - Has a 1 as result for 50 % of the groups (0 for the other 50 %)
  - All bits are relevant (there is no addressing function with fewer bits that yields the same result)

Derive Addressing Functions

- For each possible addressing function, the following requirements have to be met in order to add the function as valid:
  - Yields the same results for all addresses within the same group
  - Has a 1 as result for 50 % of the groups (0 for the other 50 %)
  - All bits are relevant (there is no addressing function with fewer bits that yields the same result)
- Simplify addressing functions by removing linear combinations

Sanity Checks

- There are several sanity checks to avoid invalid results:

Sanity Checks

- There are several sanity checks to avoid invalid results:
  - The number of banks has to be a power of two (the approach does only support linear addressing functions)

Sanity Checks

- There are several sanity checks to avoid invalid results:
  - The number of banks has to be a power of two (the approach does only support linear addressing functions)
  - The number of addressing functions should be $\log_2(n_{Banks})$

Sanity Checks

- There are several sanity checks to avoid invalid results:
  - The number of banks has to be a power of two (the approach does only support linear addressing functions)
  - The number of addressing functions should be $\log_2(n_{Banks})$
  - The addressing functions should be orthogonal

Conclusion

Conclusion

- This approach yields good results on the tested Intel and AMD CPUs
  - AMD Ryzen 9 5950X, AMD Ryzen 9 3900X, Intel Core i9-10900K, Intel Core i7-4800MQ

Conclusion

- This approach yields good results on the tested Intel and AMD CPUs
  - AMD Ryzen 9 5950X, AMD Ryzen 9 3900X, Intel Core i9-10900K, Intel Core i7-4800MQ
- The results are more constant than the ones of Drama at the cost of being slower

Conclusion

- This approach yields good results on the tested Intel and AMD CPUs
  - AMD Ryzen 9 5950X, AMD Ryzen 9 3900X, Intel Core i9-10900K, Intel Core i7-4800MQ
- The results are more constant than the ones of Drama at the cost of being slower
- The tool is open-source, feel free to try it (https://anonymous.4open.science/r/amdre-poc-A085/)

Questions, Discussion