# Homework 7

## Due on 3/31 midnight. Total available points: 100

1. **Memory Interleaving(30 points)**

A machine has a main memory of 4 KB, organized as 1 channel, 1 rank and N banks (where N > 1). The system does not have virtual memory.

- Data is interleaved using a **cache block interleaving** policy, as described in lecture, where consecutive cache blocks are placed on consecutive banks.
- The size of a cache block is 32 bytes. Size of a row is 128 bytes.
- An open row policy is used, i.e., a row is retained in the row-buffer after an access, until an access to another row is made.
- A row-buffer hit is an access to a row that is present in the row-buffer. A row-buffer miss is an access to a row that is not present in the row-buffer.

a) For a program executing on the machine, accesses to the following bytes miss in the on-chip caches and go to memory.
**0, 32, 320, 480, 4, 36, 324, 484, 8, 40, 328, 488, 12, 44, 332, 492**
The row-buffer hit rate is 0%, i.e., all accesses miss in the row-buffer.
What is the minimum value of N - the number of banks?

b) For the same sequence, if the row-buffer hit rate for the same sequence were 75%, what would be minimum value of N - the number of banks?

c) Could the row-buffer hit rate for the sequence be 100%? Why or why not? Explain. If yes, what is the minimum number of banks required to achieve a row-buffer hit rate of 100%?

## 2. Memory Interleaving II (20 points)

A DRAM main memory system has N banks in one rank on one channel. A row is 256 bytes and a cache block is 64 bytes. Data is **row-interleaved across banks** using the following physical address bit scheme: (Row: Rank: Column: Byte Offset).

An open row policy is used, i.e., a row is retained in the row-buffer after an access until an access to another row is made. Initially, row 1024 is open in all banks in all channels.

a) Determine the total number of banks (across all channels) in the system given that the row-buffer hit rate on the following sequence of cache block numbers is 33.3% (1/3)
Sequence:
0, 4, 8, 16, 32, 64, 128, 256, 128, 64, 32, 16, 8, 4, 0.

b) For what total number of banks in the system will the row-buffer hit rate of this sequence be 7/15?

## 3. **Memory Basics (15 points)**

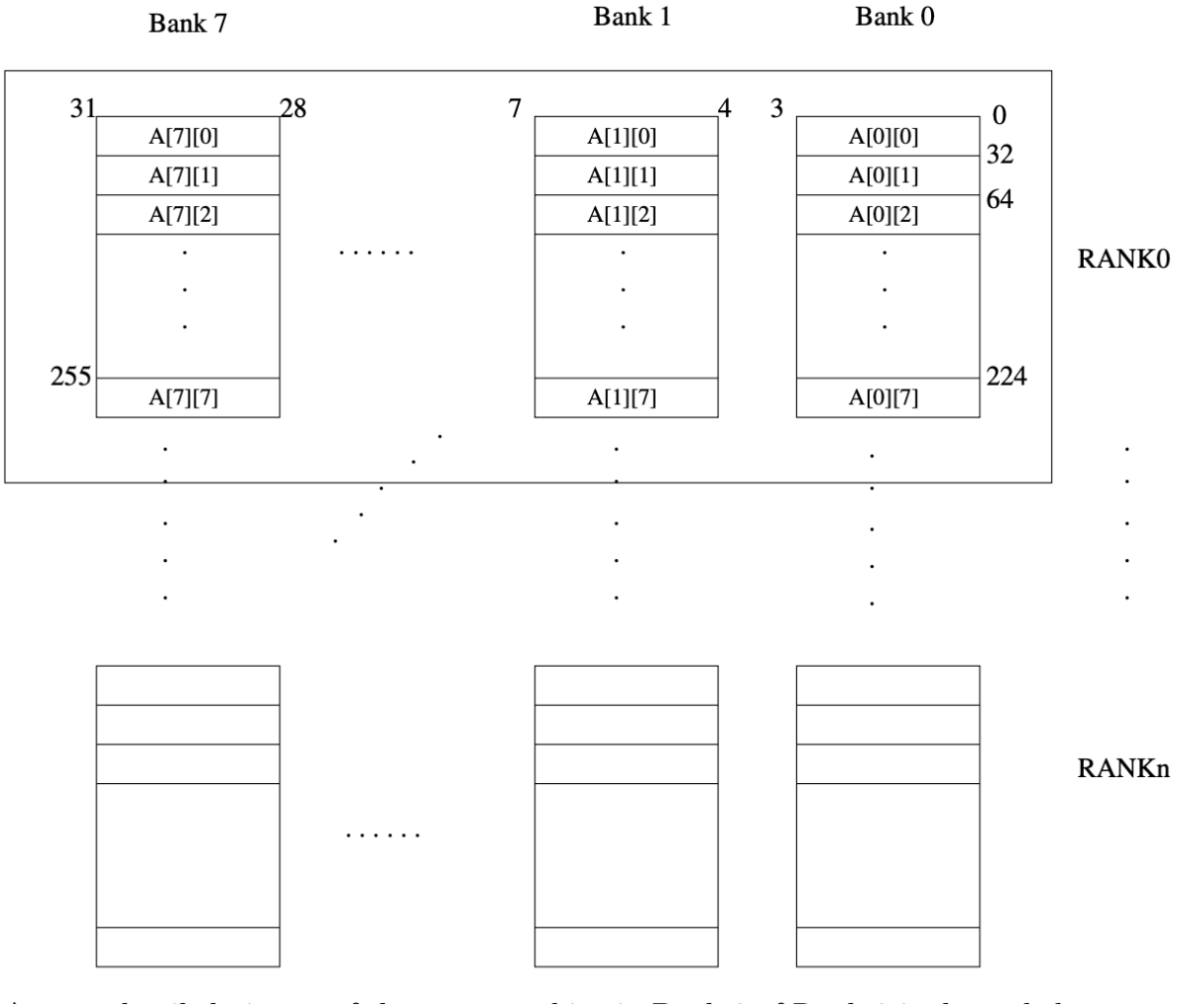
a) Which one needs refresh, SRAM or DRAM? Why?
b) What are the typical page table entry fields and what is the function of each field?
c) Where does page table locate? How to solve the slow page table lookup problem?
d) When accessing the DRAM, does the access time remain unchanged?
e) What does the bank access latency consist of (Hint: multiple cases to discuss)?

## 4. **Memory Organization (35 points)**

Consider the following piece of code:

```
for(i = 0; i < 8; ++i){
        for(j = 0; j < 8; ++j){
                sum = sum + A[i][j];
        }
}
```

The figure below shows an 8-way interleaved, byte-addressable memory. The total size of the memory is 4KB. The elements of the 2-dimensional array, A, are 4-bytes in length and are stored in the memory in column-major order (i.e., columns of A are stored in consecutive memory locations) as shown. The width of the bus is 32 bits(4 bytes), and each memory access takes 12 cycles.



a) Since the address space of the memory is 4KB, 12 bits are needed to uniquely identify each memory location, i.e., Addr[11:0]. Specify which bits of the address will be used for:
**Byte offset: Addr[ 1 : 0 ] (4 byte need 2 bits to differentiate)**
**Row Bits: Addr[ : ]**
**Bank Bits: Addr[ : ]**
**Rank Bits: Addr[ : ]**

b) How many cycles are spent accessing memory during the execution of the above code (Ignore the time spent on the bus)? Compare this with the number of memory access cycles it would take if the memory were not interleaved (i.e., a single 4-byte wide array).

c) Can any change be made to the current interleaving scheme to optimize the number of cycles spent accessing memory? If yes, which bits of the address will be used to specify the byte on bus, interleaving, etc. (use the same format as in part a)? With the new interleaving scheme, how many cycles are spent accessing memory(Still assume the time spent on bus is ignored)? Remember that the elements of A will still be stored in column-major order.

**Byte offset: Addr[ 1 : 0 ] (4 byte need 2 bits to differentiate)**
**Row Bits: Addr[ : ]**
**Bank Bits: Addr[ : ]**
**Rank Bits: Addr[ : ]**

d) Using the original interleaving scheme, what small changes can be made to the piece of code to optimize the number of cycles spent accessing memory? How many cycles are spent accessing memory using the modified code(Still assume the time spent on bus is ignored)?