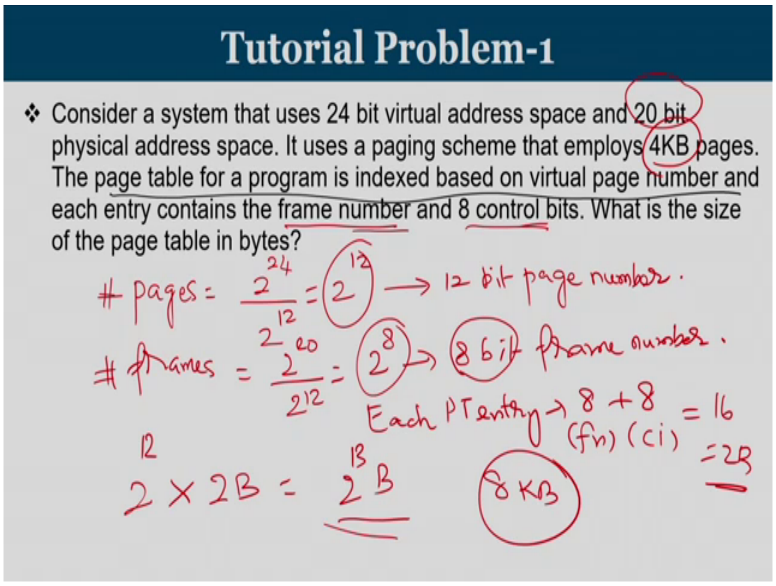
Multicore Computer Architecture – Storage and Interconnects

Dr. John Jose
Department of Computer Science and Engineering
Indian Institute of Technology, Guwahati

Lecture – 19
Tutorial 4
Main Memory Concepts

Welcome to the 4th tutorial. In this tutorial our focus of attention is on concept in main memory and address translation techniques. We have already covered in our lecture what is organization of main memory, and we have seen that main memory organization consist of channels, beams, ranks, banks, rows and columns. Today we will have a couple of problems which will deal with about the address mapping techniques in main memory.

(Refer Slide Time: 01:00)



Tutorial Problem-1

❖ Consider a system that uses 24 bit virtual address space and 20 bit physical address space. It uses a paging scheme that employs 4KB pages. The page table for a program is indexed based on virtual page number and each entry contains the frame number and 8 control bits. What is the size of the page table in bytes?

$\# \text{pages} = \frac{2^{24}}{2^{12}} = 2^{12} \rightarrow$ 12 bit page number.

$\# \text{frames} = \frac{2^{20}}{2^{12}} = 2^{8} \rightarrow$ 8 bit frame number.

Each PT entry $\rightarrow \underset{(fn)}{8} + \underset{(ci)}{8} = 16 = 2B$

$2^{12} \times 2B = 2^{13} B$ 8KB

So, look the first problem today consider a system that used 24 bit virtual address space and 20 bit physical address space. It uses a paging scheme that employs 4 kilobyte pages. The page table for a program is indexed based on virtual page number and each page table entry consists of the frame number and 8 control bytes. What is the size of the page table in bytes?

So, we have to find out what is the size of the page table. There are two things what we have to understand in this context, when you talk about the page table there are the number of entries in the page table that is depending on the number of pages in the program, number of pages is dependent on the virtual address space. So, first you find out number of rows in the page table and each of the row how much of information is being saved. So, here it is mentioned that the page table for the program is indexed based on virtual page number and each of the entry contains a frame number it is not the address it is a frame number and 8 control bits. So, first we have to find out number of rows and in each of the row we have to store a frame number and we have to store 8 other control information which may be page replacement bits, other valid bit, 30 bit etcetera.

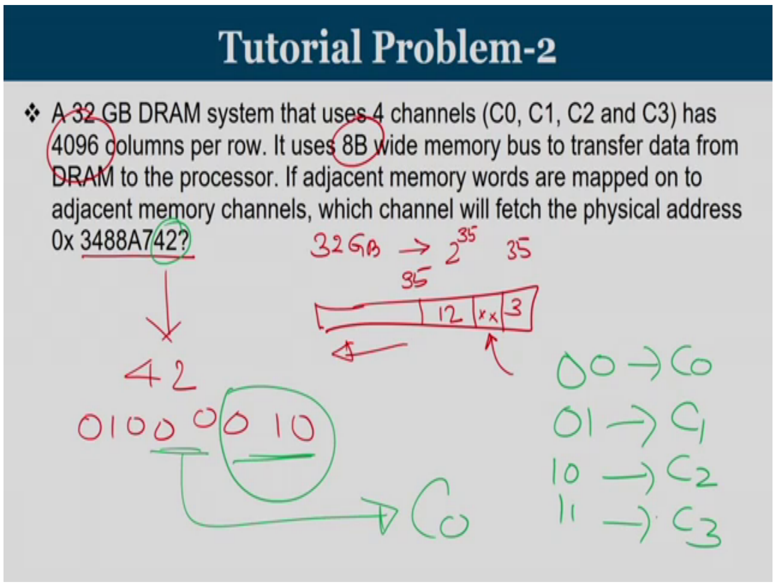
So, let us try to find out what are the number of bits required for representing a virtual page number. So, number of pages that is defined as, you have 24 bit virtual address. So, 2 power 24 is the total virtual memory divided by it is divided into pages each of capacity 4 KB, so 4 KB is 2 power 12. So, you have 2 power 12 meaning 12 bit page number. Now, number of frames, number of main memory frames with depending on the physical address we have a 20 bit physical address. So, it is 2 power 20 is the main memory capacity divided by number of page size is 2 power 12, so that you will get it as 2 power 8. So, meaning I have an 8 bit frame number.

Now, if you look at your page table, the page table has n entry where n equal to number of pages. So, I have 2 power 12 pages. Now, it is an 8 bit frame number will be stored in each of the entry plus 8 control bits. So, each of the entry, each page table entry has an 8 bit frame number plus 8 bit control. So, this is frame number and this is control information. So, together I have 16 bits or also known as 2 bits. So, I have total of 2 power 12 pages into each of this page is going to consume, each of this entry is going to consume 2 bites. So, I have total of 2 power 13 bits is needed for implementing the page table it is also known as 8 kilobytes. So, the final answer is 8 kilo bits.

Let us try to revisit what we have done. We have to find out the number of pages number of pages is obtained by total virtual memory page that is 2 power 24 divided by what is a page size, that is 2 power 12. So, you get total to power 12 as the number of pages. So, its 12 bit page number now we have to find out the number of frames in a similar fashion what is the physical address physical address is 2 power 20 divided by page size that is 2 power 12.

So, we get 2 power 8 as the number of frames. So, to represent a frame number we require 8 bit. So, each of the page table entry has an 8 bit frame number plus an 8 bit control information together 16 bits, that is 2 bit per page table entry we have total of 2 power 12 entries in the page table. So, 2 power 12 into 2 bytes that is 2 power 30 its 8 kilobytes.

(Refer Slide Time: 05:37)



Tutorial Problem-2

- A 32 GB DRAM system that uses 4 channels (C0, C1, C2 and C3) has 4096 columns per row. It uses 8B wide memory bus to transfer data from DRAM to the processor. If adjacent memory words are mapped on to adjacent memory channels, which channel will fetch the physical address 0x 3488A742?

32GB → 2^35 35
35
12
xx
3
42
0100 00 10
C0
00 → C0
01 → C1
10 → C2
11 → C3

Moving into the next problem; a 32 GB DRAM system uses for channels C 0, C 1, C 2 and C 3 and its has total of 4096 columns per row. It uses an 8 byte wide memory bus to transfer data from DRAM to the processor. If adjacent words are mapped to adjacent memory channels which channel will fetch the physical address?

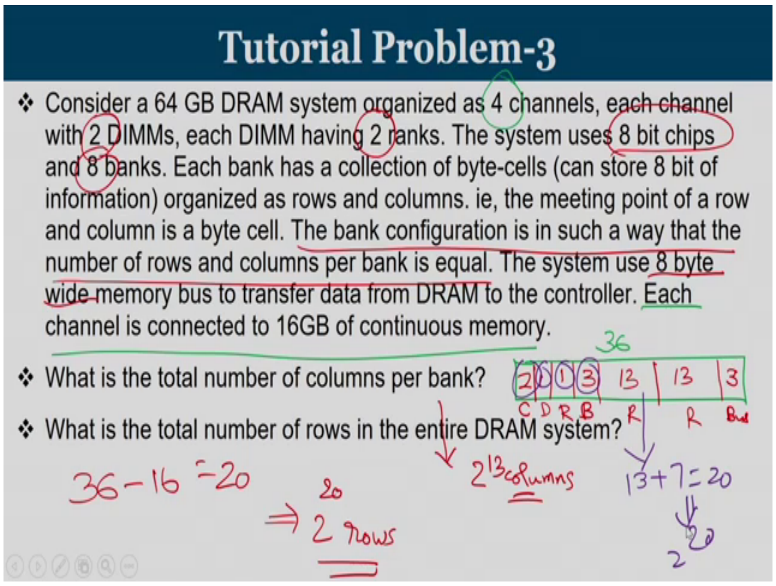
So, let us try to divide the splitting its using a 32 GB DRAM. So, if it is a 32 GB DRAM then it is basically 2 power 35. So, what I need is I have 35 bits in the address. So, given the address it is 35 bits and 8 byte wide, so the last 3 bits will be byte within the bus. Now, this is what is called a word. So, 1 word consist of 8 bytes. So, once you transfer a word then 8 bytes are getting transferred from the memory into the processor.

Now, here it is mentioned that adjacent words are mapped into adjacent channels. So, once we once we transfer a word 8 bytes, the next 8 bytes are located in the next channel. So, we have 4 channels. So, the channel bits will be coming here, these are two bits for channels and then I have 12 bits that is total of 4096 12 bits for the column and then followed by the row bank and other things. So, these are the remaining bits. So, for me the channel bits comes just after the byte within the bus. So, the last 3 bits, last 3 bits the most least significant 3 bits will be used for bus values. So, what we do is we are going to take the next to bits. So, these 2 bits will tell you what is going to be the channel number. So, consider the address. So, out of which I will take only the last 4 two that is only our interest 4 stands for 0 1 0 0, and 2 stands for 0 0 1 0.

Now, this much value is gone for byte within this bus and these two will tell the channel number, so 0 indicates channel 0. So, this particular address is coming from channel 0. So, I will try to re-summarize what we have done. Given a DRAM system consisting of 4 channels and it is mentioned that we have adjacent words mapped into adjacent channels, so 1 word corresponding to 8 byte.

So, the last 3 bit of the address is actually bytes within a word and the next bits in the address looking from the least significant side there should be given to the channel then only adjacent words will be going into adjacent channels. So, when you take up the last 8 bits, here it is the value is 42, once you write that the last 3 bits will go 2 bit within this bus and the next 2 bits is used for channel, since the 2 bit value is 0 0 then it is corresponding to channel 0. If this 2 bit value could have been 0 1 then it is channel 1, if it is 1 0 then it is channel 2, and if it is 1 1 then it is channel 3.

(Refer Slide Time: 09:17)



Tutorial Problem-3

- Consider a 64 GB DRAM system organized as 4 channels, ~~each channel~~ with 2 DIMMs, each DIMM having 2 ranks. The system uses 8 bit chips and 8 banks. Each bank has a collection of byte-cells (can store 8 bit of information) organized as rows and columns. ie, the meeting point of a row and column is a byte cell. The bank configuration is in such a way that the number of rows and columns per bank is equal. The system use 8 byte wide memory bus to transfer data from DRAM to the controller. Each channel is connected to 16GB of continuous memory.
- What is the total number of columns per bank?
- What is the total number of rows in the entire DRAM system?

Now, we move on to the third problem. So, consider a 64 GB DRAM system organized the 4 channels each channel with 2 DIMMS, each DIMM has 2 ranks. The system uses 8 bit chips and 8 banks, each bank has a collection of byte cells and can store 8 bit of information organized as rows and columns, that is a meeting point of a row and column is a byte cell. The bank configuration is in such a way that the number of rows and columns per bank is equal, the system used 8 byte wide memory bus to transfer data from DRAM to the controller. Each channel is channel is connected to 16 gigabytes of continuous memory.

So, let us try to look into the last line each channel is connected to 16 gigabyte of continues memory, I have total of 64 GB and that 64 GB is protocol 4 channels. Now, when you have 4 channels and peculiarity is that 16 GB of memory is continuous inside one channel. So, first 16 GB is channel 0, next 16 GB is channel 1, next 16 GB is channel 2 and the last 16 GB is channel 3.

For this to realize we have to dedicate the most significant 2 bits for channels. So, let us try to divide these split of the address. So, once you take the address we have total of 64 GB DRAM. So, 64 Giga means 2 power 36 that means, I have total of 36 bits in the address. Now, 2 bits is gone for channels that it is a most significant 2 bits are gone for the channel bits. And then we have 1 bit for the DIMM and then we have 1 bit for the rank. So, it is 1 bit DIMM and 1 bit rank because we have 2 DIMMS and 2 ranks. So, 1 bit is sufficient enough and that is why we have now 3 banks. So, 3 bit bank. So, it is 8 banks are there. So, 3 bits are dedicated for banks and we have to understand that the number of chips that we use is no longer relevants as far as the splitting of address is concerned.

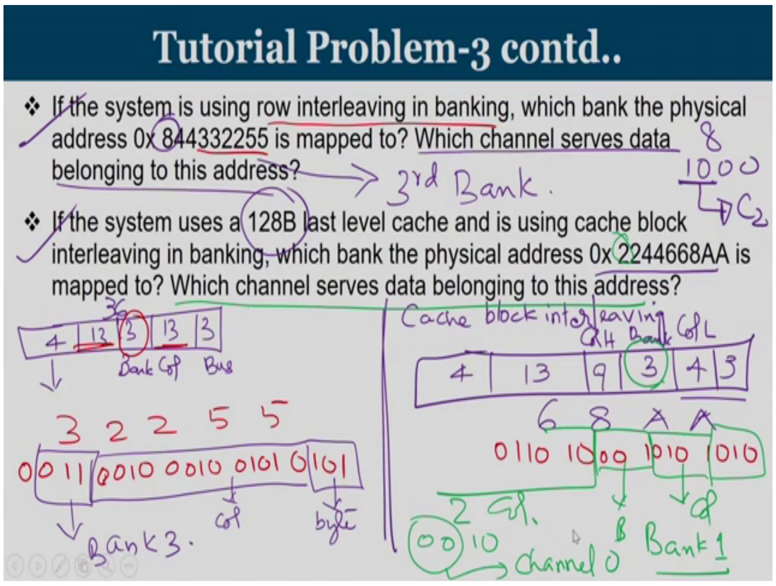
Since, we use 8 byte wide memory bus the last 3 bits that is each of the word is corresponding to 80 bytes. So, the 3 bits are used to find out these 8 bytes. So, now, if you look at the total is 36 out of which now 10 bits are used, you have 3 bits for bus, you have 3 bits for bank, this is for bank and then you have 2 bits for channel 1 bits for DIMM and rank each. So, its 2 plus 2, 4 plus 3, 7 plus 3, 10; out of 36 bits 10 bits are already being gone, now it is split up of row and columns.

So, how will you find out the number of rows and columns? The bank configuration is in such a way that the number of rows and columns per bank is equal. So, you have now 26 bits left, this is going to be divided across rows and columns and if you wanted in the equal split up then it is going to be 13 and 13. So, I have 13 bit for row and they have 13 bit for column per bank there is a way how you divide. So, what is the total number of columns per bank? That is your first question. So, I have total of 2 power 13 columns are there.

Now, what is the total number of rows in the entire DRAM system? So, if you look into the entire DRAM system 13 bits are for column and 3 bits is for byte within the bus. So, total of 16 bits is been removed. So, I have total of 36 bits out of which 16 bits are already taken by byte in the bus and the column. So, I have remaining 20 bits and that many columns are there. Meaning I have 2 power 20 rows that is also equal to if you take up the row bits so for an EC conversion, if you take up the row bits that is 13 and I have to add these 3 the next one one each. So, it is 4 plus 3, 7, so 13 plus 7 total of 20; meaning I have 2 power 20 rows.

So, there are two ways in which I can use first you remove from the total address you remove the column bits and the byte within the bus bits those are removed then we have balance of 20 or in a bank I have 2 power 13 rows I have 8 banks. So, 2 power 13 into 2 power 3 that is 2 power 16 then I have 1 bit for ranks. So, I have 2 ranks. So, into 2 I have 2 DIMMS again into 2, I have 4 channels again into 4, that will give you total of 2 power 20.

(Refer Slide Time: 14:19)



Tutorial Problem-3 contd..

- If the system is using row interleaving in banking, which bank the physical address 0x 844332255 is mapped to? Which channel serves data belonging to this address?
- If the system uses a 128B last level cache and is using cache block interleaving in banking, which bank the physical address 0x 2244668AA is mapped to? Which channel serves data belonging to this address?

Now, continuation of this question, if the system is using row interleaving in banking which bank the physical address 0 x 844332255 is mapped. So, we have to understand what do you mean by row interleaving. Row interleaving means adjacent rows are mapped onto adjacent banks. So, once all the column bits are over then comes the bank bits. So, let us try to find out the scheme of the address, you have the last 3 bits that is going for the bite within this bus and then this is the bank bit.

So, row interleaving means once the column beat is over then you have to use and this is what is called the remaining 4 bits. So, we have 13 plus 3 that is 26, 26 plus 6, 32 and this makes 36. This 4 corresponds to 2 bits for your channel, 1 bit for the DIMM and 1 bit for the rank. So, this is the split up. So, the main point in this context is once the column bits is over between the row and the column the bank bits will come that is what is known as row interleaving. So, row interleaving is a concept where adjacent rows are mapped onto adjacent banks.

So, once all the possible bit values are given in the column then next column number should be mapped into the next bank. So, that is way how its work. So, we have to extract these bits. So, now, let us try to take care the values. So, if the last, so 3 2 2 5 5, these are the last 20 bits out of which 3 stands for 0 0 1 1, 0 0 1 0, again 0 0 1 0 and this 5 is 0 1 0 1, 0 1 0 1.

Now, this last 3 bits this is going for the byte within this bus and then I have 13 bits. So, this much is this 13 bits that is the column. So, this corresponds to the column number this corresponds to the byte number and these 3 bits are going to tell you which is the bank. So, the value is 0 1 1 that means, bank 3. So, this particular address correspond to third bank out of the 8 bank ranging from 0 1 2 3 etcetera up to 7 this corresponds to bank number 3.

Now, the second portion of the question we look into the second portion of the question. If the system uses 128 byte last level cache and is using cache block interleaving in banking, which bank the physical address is mapped two which channel serves the data? So, the previous question also which channel serves the data we know that the most significant 2 bits is the channel. So, here the most significant 2 bits is that is 8, 8 correspond to 1 0 0 0 0. So, the most significant 2 bit is 1 0 1 0 means 2. So, it is corresponding to channel 2 that was the first case.

Now, in this case we will see what is cache block interleaving, now once you come to cache block interleaving adjacent cache blocks are mapped onto adjacent banks. So, this is the address last 3 bits. Now, since I am using 128 byte last level cache 2 power 7 bytes will fit into my cache already in a single word 8 bytes have gone. So, 128 bites has to fill in a cache. So, every 128 bit should be located in adjacent banks.

So, I have 128 bytes of data that comes from main memory, the next 128 bite should be kept in another bank then only it is going to help us. So, if you wanted to get 128 bytes then it is 2 power 7. So, the least significant 7 bits should constitute one cache block and this is called the lower column address. Once this 7 bits is over then we should have our bank bits.

I have total of 13 bits required in the column. So, the column bits 13 bits of column is being divided into lower column bits this is what we have seen that is column lower and then we have column higher. So, how much will be there for column higher? Already I have 13 bits out of which 4 bits is already taken in the column lower and the remaining is 9. So, the idea is total of 13 bits is going to be divided into lower portion column and upper proportion column and in between the bank bits are inserted.

So, I will summarize again. We have cache block interleaving and in cache block interleaving adjacent cache block data are mapped onto adjacent banks. In this case we have 128 byte is the block size of the cache. So, from main memory always 128 bytes of data are moving to the cache, because access to main memory comes whenever there is last level cache miss. So, once 128 byte is gone the next 128 byte is the next demand. So, rather than keeping adjacent cache blocks in the same bank cache block interleaving means every cache block every adjacent cache block is mapped onto adjacent banks. So, once 128 byte is been transferred the next 128 bite should be located in a different bank for that to realize to get 128 bytes we require 7 bits.

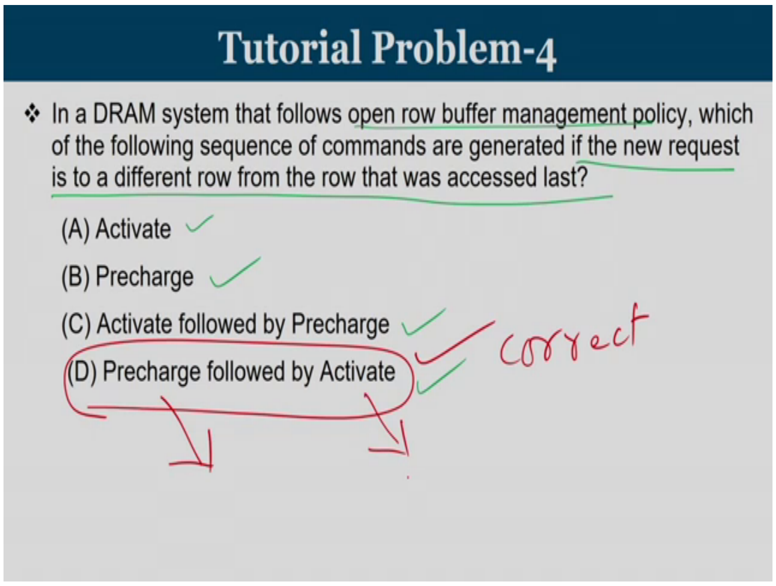
So, consecutive 2 power 7 locations are been transferred that is a last 7 bit and then comes the bank bit. So, my column address is not fully over. So, what we do is the entire column bits of 13 we have total of 2 power 13 columns, the 13 bits will represent a column number that is going to be divided into lower portion of the column and the upper portion of the column between that the bank bit is introduced, and then we have this 13 bit of row and the 4 bit that constitute the channel DIMM and the rank.

Now, if you write the address its 6 8 A A. So, if you write the 6 8 A A into the decimal equivalent form, 6 correspond to 0 1 1 0, 8 correspond to 1 0 0 0, this A is 1 0 1 0, again

1 0 1 0. Now, out of which the last 3 bits are gone for bank or it is basically your bus values that means, bite within this bus. Next 4 bit is the lower order column bits. So, this will be column bits and then we have 3 bits for the bank. So, this is essentially. So, this side is for my columns balance. So, these 3 bit value is 0 0 1 that means, it is bank column. So, I have to find out these 3 bits in the address. So, remove the last 7 bits and take the next 3 bits and that is corresponding to your bank. So, in this context this address maps to bank 1.

Now, there is one more portion the question, which channel serve this and channel is the most significant 2 bits. So, if you take 1, 0 0 1 0 and the most significant 2 bits will tell it is channel 0 that is way it is been connected. So, this is how you solve the problem given an address given the spilt up into ranks banks rows and columns you find out the mapping of the address which all bits constitute ranks and banks and based upon the type of interleaving. If it is row interleaving after the column bits the bank bits will come, if it is cache block interleaving depending on the size of the cache block that many bits has to be kept from the least significant side and then you insert the bank bits. So, that completes your third problem for this tutorial.

(Refer Slide Time: 23:13)



Tutorial Problem-4

- In a DRAM system that follows open row buffer management policy, which of the following sequence of commands are generated if the new request is to a different row from the row that was accessed last?
  - (A) Activate ✓
  - (B) Precharge ✓
  - (C) Activate followed by Precharge ✓
  - [✓ (D) Precharge followed by Activate] ✓ correct

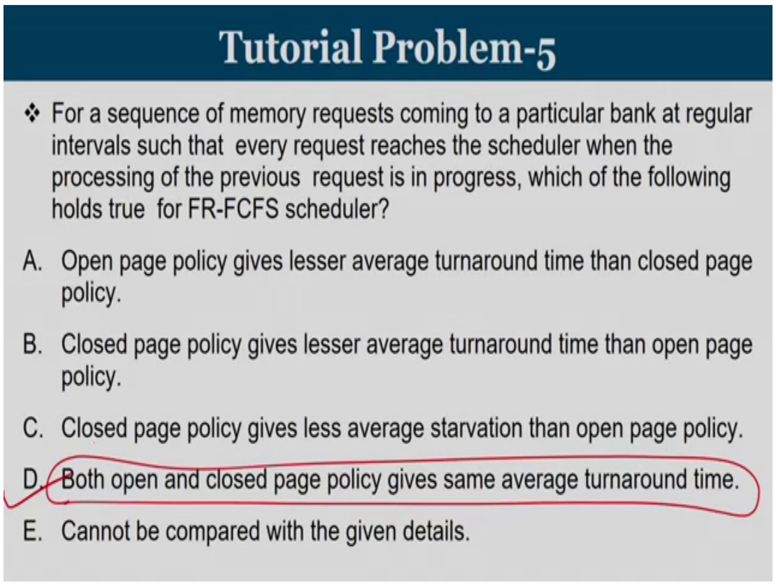
Now, this is a multiple choice question. In a DRAM system that follows open row buffer management policy which of the following sequence of commands are generated if the new request is to a different row from the row that was accessed last? What is the idea?

We have a row buffer for every bank and you transfer the contents from the row into the row buffer and you apply column number and you extract the corresponding value and that is been send on.

Now, while doing this when the next request is to the same row then only column numbers are given because it is an open row buffer management policy. If it is an open row buffer management policy then after every access we keep the row buffer as it is assuming that the next access is to an already opened row. Now, the question tells that in a DRAM system that follows open row buffer management policy, open row buffer management policy means I leave the row buffer as it is such that. If the new request is to same row there is no need to take the raw again already the row is open just give the column address and take the address, if the new request is to a different row from the row that was accessed last, what are the commands that we have to give you have to give activate command pre-charge command activate followed by pre-charge or pre-charge followed by activate.

Activate is a command which is used whenever a new row has to be brought into the row buffer pre-charge command is used whenever the contents of a row buffer has to be return back to the row. So, in this case there is something in the row buffer that is not the row what I want for the new request. So, first I have to write it back. So, I have to do a pre-charging first and then the new row has to be brought into the row buffer. So, if the correct answer is it is actually an pre-charge followed by activate is the current option. So, pre-charge will write the row back into the corresponding row write the contents of the row buffer and then activate means a new is taken into the row buffer.

(Refer Slide Time: 25:37)



Tutorial Problem-5

❖ For a sequence of memory requests coming to a particular bank at regular intervals such that every request reaches the scheduler when the processing of the previous request is in progress, which of the following holds true for FR-FCFS scheduler?

A. Open page policy gives lesser average turnaround time than closed page policy.

B. Closed page policy gives lesser average turnaround time than open page policy.

C. Closed page policy gives less average starvation than open page policy.

[✓ D. Both open and closed page policy gives same average turnaround time.]

E. Cannot be compared with the given details.

Now, let us goes to another problem. For a sequence of memory request coming to a particular bank at regular intervals such that every request reaches the scheduler, when the processing of the previous request is in progress. So, already I am progressing, I am servicing the request. Now while the service is in progress new request come to the queue. The meaning is once where one request is completely serviced the queue has an element if the queue has the same row which is already opened then there is no need for any pre-charge or activate I have to just enable column address strobe. If it is to a different row then I have to pre-charge it and then I have to again activate.

Now, the option that is given open page policy gives lesser average turnaround time then closed page policy. What is the difference between open page policy and closed page policy? In an open page policy our assumption is the next request the next incoming request will be there to the same row what is currently opened. So, I want to close it I want to pre-charge it I leave it open. And what is the closed page policy? Once I have completed servicing of a current request, if the queue is empty then I close the row I do the pre-charge, but if the queue is not empty we come to know whether already open row can I service and incoming request.

So, open page policy gives lesser turnaround time then close page policy that we cannot say because in this case you are getting request at regular intervals. Closed page policy gives lesser turnaround time that also we cannot say. The peculiarity is an closed row policy will differ from opened row policy if an only if by the time I complete a memory request service my queue should be empty. If the queue is empty means there are no more pending request then closed row policy will close the page by using a pre-charge operation opened row policy will leave it open. If the queue is not empty then either it may be to the same row or it may be to different row. So, in that case open row policy and closed row policy is never going to be different.

So, both open row policy and closed row policy will give the same average turnaround time that is the answer. Because there is no difference at all by the time you complete the request the next request is already there, whether it is open row or closed row either you have to pre-charge and activate or you have to give only the column address when there is a row hit or when there is a row conflict. So, this gives you a better idea about the difference between open row policy and closed row policy.

So, with this we come to the end of today's tutorial session. We first worked on a problem with respect to virtual memory finding out the size of the page table and then we have 2-3 problems that were dealing with the address mapping in DRAM, finding out channels and bank numbers from a given address and a given constraint. And then we learn little bit about the significance of activate pre-charge and column address strobe commands. So, I think this will give a fair background in working with address mapping techniques in DRAM. So, with this we complete this tutorial.

Thank you.