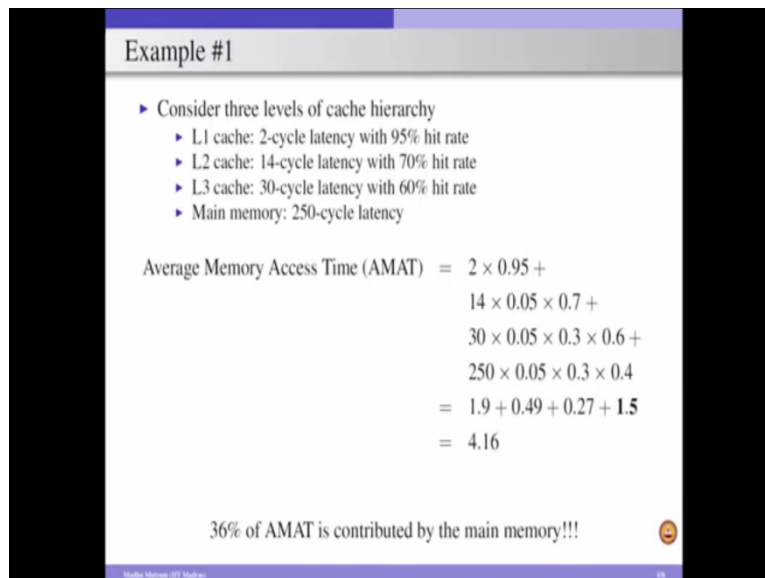


**Computer Architecture**  
**Prof. Madhu Mutyam**  
**Department of Computer Science And Engineering**  
**Indian Institute of Technology, Madras.**

**Module – 04**  
**Lecture - 10**  
**Memory Hierarchy Design (Part 5)**

So, as part of memory hierarchy design, last week we discussed cache memory design and optimization techniques and this week we are going to look at DRAM based main memory organization. So, first we will discuss a motivating example that shows the importance of the memory.

(Refer Slide Time: 00:35)



**Example #1**

- ▶ Consider three levels of cache hierarchy
  - ▶ L1 cache: 2-cycle latency with 95% hit rate
  - ▶ L2 cache: 14-cycle latency with 70% hit rate
  - ▶ L3 cache: 30-cycle latency with 60% hit rate
  - ▶ Main memory: 250-cycle latency

Average Memory Access Time (AMAT) =  $2 \times 0.95 +$   
 $14 \times 0.05 \times 0.7 +$   
 $30 \times 0.05 \times 0.3 \times 0.6 +$   
 $250 \times 0.05 \times 0.3 \times 0.4$   
 $= 1.9 + 0.49 + 0.27 + 1.5$   
 $= 4.16$

36% of AMAT is contributed by the main memory!!!

So, consider a three level of cache hierarchy in a system, where L1 cache takes 2 cycle latency with 95% hit rate. L2 cache is bigger than L1 and which takes 14 cycles latency with 70% hit rate and the last level cache LLC or L3 cache which takes 30 cycle latency with 60% hit rate. In addition to these three levels of cache hierarchy we will also consider the main memory and which takes 250 cycle latency.

And also we assume that the main memory is perfect. That means like if data is not there in any of the L1, L2 or L3 caches, then the data will be there in the memory. So, there would not be any misses in the main memory that is the assumption we consider. And also we assume a couple of other things here when I say 14 cycle latency for L2, we go to L2 only when we have a miss in L1. So these fourteen cycles also includes the latency of accessing L1.

Similarly, when I say thirty cycle latency for L3, these thirty cycles also include the latency associated with accessing L1 and L2.

And another thing is the hit rates are given in this problem statement are local hit rates. So, once we have these three levels of cache hierarchy plus main memory. So, we will compute the average memory access time by using our standard formula that consists of,

$$\text{Memory access time}_{\text{average}} = \text{Hit latency} * \text{Hit rate of cache} + \text{Miss latency} * \text{Miss rate of cache}$$

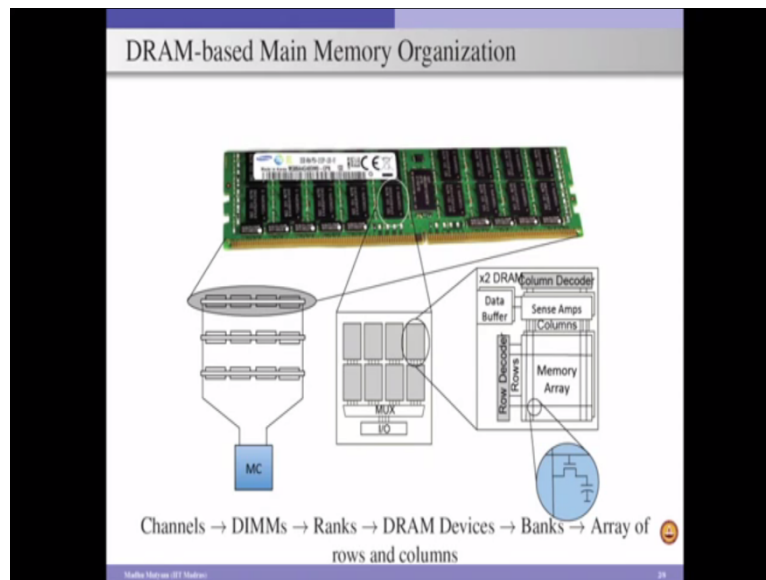
And there also we again multiply 'the hit latency of that particular cache' times 'the global hit rate' plus to the next level and so on. So, once we compute all these things which is like -

$$\begin{aligned} \text{Memory access time}_{\text{Average}} &= (2 * 0.95) + (14 * 0.05 * 0.7) + (30 * 0.5 * 0.3 * 0.6) + (250 * 0.05 * 0.3 * 0.4) \\ \text{Memory access time}_{\text{Average}} &= 4.16 \end{aligned}$$

because we know that is 95% hit rate at L1 and 2 cycle latency is at L1 plus 14 into 5% of the time we will go to L2 from L1. So, 0.05 times 0.7 because 70% hit rate in L2 plus 30 into 0.5 into 0.3 where .three is the miss rate from L2. 30% of the time we will come from L2 to L3 and then there is a hit rate of 60% in L3. So, this is 0.6 plus 250 cycles we incur at memory and the total number of times we come to main memory is nothing but the miss rate at L1 times miss rate at L2 times miss rate at L3, which is 0.05, 0.3 and 0.4 respectively. Once we do all this math so we get 4.16 is the average memory access time for this particular problem statement.

And in that we know that 1.5 is the total time taken for accessing the data from the memory. So, which is in other words 36% of the AMAT is contributed by the main memory. So, as a result for improving the overall performance we have to come up with efficient memory mechanisms, but to come out with any optimizations at the memory and so on. First of all we have to know the internal organization of the memory. So, unless we are clear about the organization of the memory, we cannot propose any new optimization techniques. So, that is the motivation for us to study the internals of DRAM based main memory organization. So, all of us know that in our computers laptops we find this component.

(Refer Slide Time: 04:44)



This is a dual inline memory module DIMM. And this is the DRAM based memory that is sitting in our mother board. And you can see here this DIMM consists of lots of the blocks, rectangular boxes, which are called as DRAM devices or chips. There are plenty of the DRAM chips placed on this DIMM. And this particular DIMM is having a capacity of 32 Giga bits. This is a recently developed the DIMM from Samsung. In order to overcome the memory wall problem we know that we have to use multiple levels of cache hierarchy as well as large capacity memories.

So, with that point in view so these memory vendors are supplying the DIMMs with huge capacities and for instance this particular DIMM is supporting 32 Gigabits of capacity. So, this is the physical organization memory which is organized in terms of a DIMM and DIMM consists of collection of DRAM devices. Now, we want to see how this is connected to the other components and then what are the logical organizations of the DRAM based memory and so on we will look at now. So, in our mother board typically we can have one or more DIMMs connected to a memory controller.

And effectively all these the DIMMs are controlled by a single channel and there is a memory controller which can handle multiple channels. So, there is a memory controller which is connected to multiple DIMMs or which is controlling multiple DIMMs and each DIMM has multiple devices on it. And if we look at each of this individual DRAM devices which in turn consists of collection of rectangular boxes and these boxes we call it as sub-bank. So, in this

particular example we consider 8 sub-banks within a single device. So, we have collection of DRAM devices on a DIMM and each device in turn consists of collection of sub-banks and each of these sub-banks if we look at further, then it consists of collection of the DRAM memory arrays.

In this particular example, we consider each of these sub-banks as x2 device and x2 stands for, there are 2 memory arrays associated with that particular sub-bank. By the way I mentioned the sub-bank here not the bank. So, this is like. So, in the physical organization of the DRAM based memory consists of the DIMM devices. And each device has collection of sub-banks and each sub-bank has collection of DRAM memory arrays, but if we look at this from the logical organization point of view.

A DIMM can have one or more ranks associated with that. And each rank consists of a set of DRAM devices and all the devices within a rank work in unison. So, given a signal this signal will go to all the DRAM devices associated with that particular rank. And this rank can be further divided into collection of banks. But previously I mentioned the DRAM device is divided into collection of sub-banks, but when I said from the logical organization point of view, a rank is organized as a collection of banks.

So, effectively a bank is a physically distributed across multiple devices associated with that particular rank. So, in this particular example, if I consider a rank consists of 4 DRAM devices and each DRAM device has 8 sub-banks. Effectively my rank has 8 banks and one fourth of the bank is there in one device, another one fourth will be there in the second device, third one fourth will be in the third device and the last one-fourth of the bank is there in the final device.

So, effectively each quarter of the DRAM bank is associated with one of the 4 the DRAM devices. So, as a result when we give a command from the memory controller to read data from a particular bank, this command will go to the corresponding rank and within that rank it will go to all the devices associated with that rank. And within that it will go the appropriate bank and then it will get the data. So, this type of organization is required mainly because we want to provide higher bandwidth to the processor. And we already discussed that each of this sub-banks is physically organized as collection of DRAM arrays.

A DRAM array is nothing but it is a collection of DRAM cells organized as the word lines and bit lines. And whenever a request is sent from the memory controller, it will go to the

corresponding RAM, to the corresponding device, to the corresponding bank and finally, once it goes to the corresponding bank, as we know the bank is distributed across multiple devices, the command will go to the corresponding sub-banks in each of these devices and it will go to the row decoder part.

And part of the address is given to the row decoder and which selects a particular word line in that memory array. And the second part of the, remaining part of the, address will be given to the column decoder and which is going to select the required bits from the selected word line. The moment you apply the address to the row decoder it is going to activate a word line. And this, whatever the word line you activated the data from that word line will be read into sense amplifiers and from the sense amplifiers wants to apply the address to the column decoder, it selects the required bits and that bits will be transferred to the data buffer.

And remember in this particular example we consider, it is a x2 device. So, there are 2 DRAM memory arrays. So, the command whatever the request we send will go to 2 DRAM arrays simultaneously and both will perform the operation parallelly and they will send the data to the corresponding data buffers. This way we can parallelize the things.

And if you want to increase our data read per unit time, then we can go for x4, x8 and x16 devices and so on. When I consider x16, there are 16 DRAM arrays working in unison, given a command all these 16 arrays will read one bit of data at a time and then supply that to the data buffers. So, with that like once we have more number of the DRAM arrays packed together all these can work simultaneously and supply the data. By the way the intersection of this word line and bit line will be the DRAM cell and DRAM cell is nothing but it consists of a transistor and a capacitor.

Unlike any SRAM cell where to store each bit of information we require 4 to 10 transistors, but whereas, in the case of the DRAM cell we require only one transistor, but one capacitor also is required. Because of this low number of transistors required to store one bit of information, the density wise DRAM based technology is going to provide higher density. So, that we can keep more and more information, but at the same time because the capacity is also associated with the DRAM cell and the data is represented in the DRAM cell in terms of the charge in the capacitor.

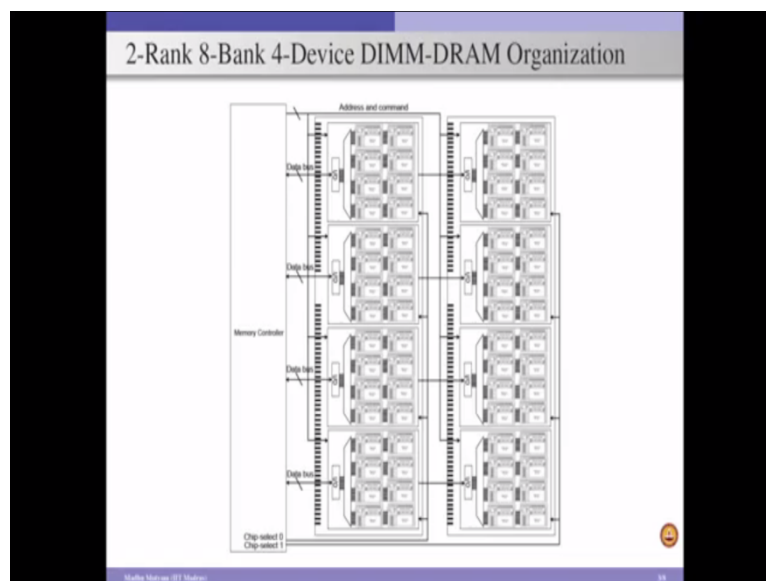
So, as a result we have side effects associated with this the capacitor which in turn increases our overall access time also. So, overall our the entire memory is organized like set of

channels, each channel can consists of multiple DIMMs and each DIMM again consists of multiple ranks and each rank can consists of multiple DRAM devices and each DRAM device will have multiple sub-banks.

So, as a result from the rank point of view, a rank consists of multiple banks and all the banks in within a rank can work independently. And each bank consists of array of the rows and columns are also can be represented as a collection of DRAM memory arrays. So, this is the internal organization of any DRAM based main memory. And to perform a single read operation or a write operation.

So, we have to do lots of preprocessing and that actually results in significant amount of time. So, that is also one of the reasons why the DRAM access latency is significantly higher compared to the SRAM based the caches.

(Refer Slide Time: 15:12)



We just consider an example here. So, in this particular example, we have a DIMM, consists of 2 ranks and each rank has 8 banks, these 8 banks are distributed across 4 devices. So, effectively when we see a physical organization we have a DIMM, within a DIMM we have 4 devices, within a device we have 8 the sub-banks. And we consider in this particular example each of this sub-bank is x4 device. Since we have 2 ranks, when a command is send from the memory controller at any point of time we can perform operation with one rank because there are 2 ranks in this particular thing in the address generated by the memory controller.

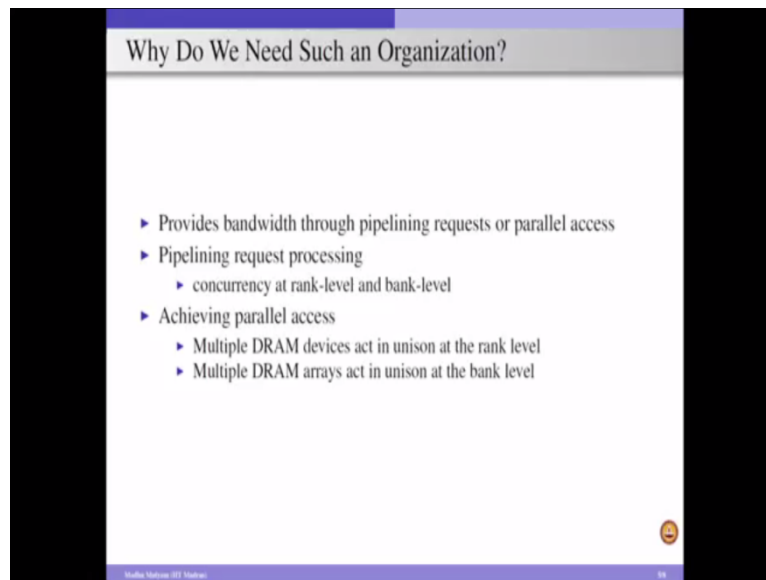
There is one bit dedicated for this rank select, we select a particular rank based on that particular bit. And then once we select that particular rank based on the bank id because here there are 8 banks. So, we dedicate three bits to indicate which bank we want to access we access that particular bank. And then based on our row address, which is nothing but, because we know that each of the sub-banks are collection of the DRAM memory arrays and each memory array has collection of word lines.

And each word line we call it as a row or a hardware page in the DRAM memory terminology. So, we select a particular row within that particular DRAM array and then read the data from that particular row to the sense amplifiers and using the column address we select the required bits and transfer that to the output buffers and which in turn will transfer the data to the processor or to the L3 cache or what. So, in this particular example, to access a DRAM row or a page so we selected the rank 0 within that we selected 0 bank and within that we selected a particular row because we know that bank is distributed across all these 4 devices.

So, our hardware page or DRAM row is also distributed across all these 4. So, the total size of a DRAM row is nothing but the size, sum of the sizes, of the DRAM rows in each of these sub-arrays. So, why do we need such a complex organization for our DRAM based memory? As I said previously, we want to provide higher bandwidth, we want to support higher bandwidth from the memory because the processor performance is significantly increasing, but the memory performance is not increasing at that rate. So, as a result we have the performance gap widening over a period of time and that is called as the memory wall problem.

In order to tackle the memory wall problem in addition to the multi-levels of cache hierarchy, we also have to have an efficient DRAM memory organization and that is the reason why we are considering all these the ranks, banks, collection of DRAM arrays and so on.

(Refer Slide Time: 18:28)



So, in one line answer for such a complex organization is providing higher bandwidth through pipelining of our requests and parallel access of our requests. So, pipelining can happen through a concurrency at the rank level and the bank level. So, multiple requests, memory requests, can be processed in pipeline fashion with the support of this concurrency at the rank level and the bank level. And similarly, to provide the parallel access because we have multiple DRAM devices associated with each rank and all these devices within a rank will act in unison.

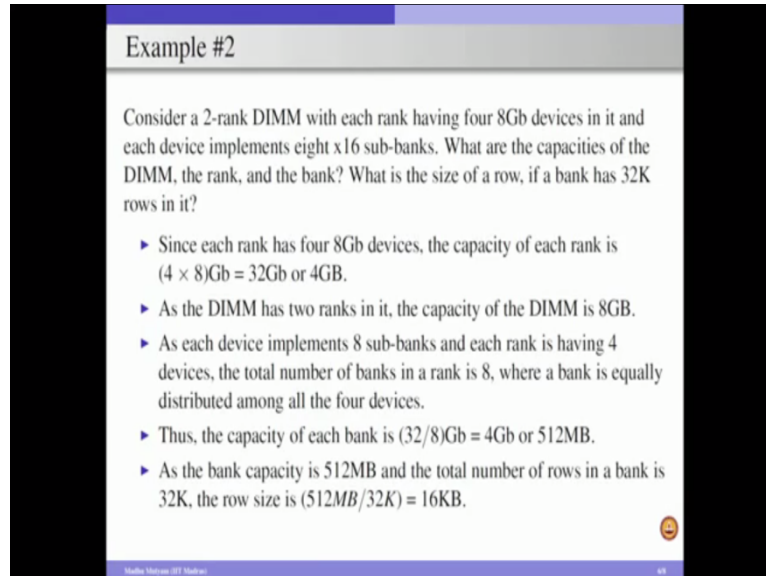
So, that indicates that given a command all these devices will act simultaneously. So, that, when we want to read the data, we can read data from 4 devices if the rank is having 4 devices simultaneously. So, that way like we can improve our overall, the bandwidth and similarly, within a bank we know that there are collection of DRAM arrays such as x2 device or x4, x8 and x16, given a request all these arrays will work simultaneously and supply multiple data bits simultaneously. So, that is where again like we can exploit parallelism within a bank so that the overall bandwidth can be improved.

In summary, we can exploit parallelization within a bank by accessing all these DRAM arrays simultaneously. And we can exploit parallelization within a rank by sending a request to all the devices associated with that rank. So, that the data can be sent the overall bandwidth can be improved. And we can exploit the concurrency at the rank level and the bank level. So



that, we can process the DRAM request in a pipelined fashion and bandwidth can be improved

(Refer Slide Time: 20:32)



**Example #2**

Consider a 2-rank DIMM with each rank having four 8Gb devices in it and each device implements eight x16 sub-banks. What are the capacities of the DIMM, the rank, and the bank? What is the size of a row, if a bank has 32K rows in it?

- ▶ Since each rank has four 8Gb devices, the capacity of each rank is  $(4 \times 8)\text{Gb} = 32\text{Gb}$  or 4GB.
- ▶ As the DIMM has two ranks in it, the capacity of the DIMM is 8GB.
- ▶ As each device implements 8 sub-banks and each rank is having 4 devices, the total number of banks in a rank is 8, where a bank is equally distributed among all the four devices.
- ▶ Thus, the capacity of each bank is  $(32/8)\text{Gb} = 4\text{Gb}$  or 512MB.
- ▶ As the bank capacity is 512MB and the total number of rows in a bank is 32K, the row size is  $(512\text{MB}/32\text{K}) = 16\text{KB}$ .

So, we just consider an example to see how much we understood this DRAM terminology and rank, bank, the device, the row and other things. So consider a 2 rank DIMM with each rank having four - 8 gigabit devices in it and each device implements 8 x16 sub-bands. So, what are the capacities of the DIMM, the rank and the bank? Also what is your size of a row, if a bank has 32k rows in it? So, here k stands for 1024. So, we need to find the size of the complete DIMM, size of a rank, size of a bank.

Also we need to find the size of a row. So, we know that the DIMM consists of 2 ranks and each rank has four 8 Gigabit devices. So,

$$\begin{aligned} \text{The capacity of each Rank} &= 4 * 8 \text{ Gb} \\ \text{The capacity of each Rank} &= 32 \text{ Gbits} \\ \text{The capacity of each Rank} &= 4 \text{ Gbytes} \end{aligned}$$

and because a DIMM consists of 2 ranks. So,

$$\begin{aligned} \text{Capacity of DIMM} &= 2 * \text{Capacity of a Rank} \\ \text{Capacity of DIMM} &= 8 \text{ GB} \end{aligned}$$

And we also mention that each device implements 8, x16 banks.

So, effectively a logical bank is physically distributed across multiple devices associated with a given rank and in this particular example our rank consists of 4 devices. So, as a result our logical bank is actually distributed across all these 4 devices. So, the total number of banks within a rank is 8. It is a simple thing, you just look at how many number of sub-banks are there within a device associated with a rank. And that is the total number of banks associated with that rank. And once we know that number of banks within a rank.

$$\text{The capacity of each Bank} = \frac{\text{Capacity of the Rank}}{\text{Total number of Banks within that Rank}}$$

$$\text{The capacity of each Bank} = 4 \text{ Gigabits}$$

$$\text{The capacity of each Bank} = 512 \text{ Megabytes}$$

And finally, we know that,

$$\text{The Bank capacity} = 512 \text{ megabytes}$$

$$\text{Total number of rows in a bank} = 32 \text{ k}$$

$$\text{So, The Row size} = 16 \text{ kB}$$

The typical size of a row considered in the current DRAM memories are 1 kB, 2 kB and it can be up to 8 kB. So, having discussed this example now we want to concentrate on the basic DRAM operations.

(Refer Slide Time: 23:46)

**Basic DRAM Operations**

- ▶ Address lines are multiplexed to reduce the number of pins
  - ▶ Send row address when *Row Address Strobe (RAS)* is asserted
  - ▶ Send column address when *Column Address Strobe (CAS)* is asserted
- ▶ To access data
  - ▶ *Precharge (PRE)* – places the bank in a voltage state that is suitable for activation
  - ▶ *Activate (ACT)* – reads an entire row of bits into the sense amplifiers
  - ▶ *Column Access (Rd/Wr)* – transfers data to/from the memory controller
- ▶ To retain data
  - ▶ *Restore* – DRAM reads are self-destructive
  - ▶ *Refresh (REF)* – DRAM cells lose charge due to leakage
    - ▶ Periodic charge/refill is needed
    - ▶ Retention: 64ms for < 85°C temperature and 32ms for 85°C – 95°C temperature

19

So remember that the DIMM is connected to the DIMM socket associated with the mother board and the DIMM has limited number of pin bandwidth. So, that is the reason why we apply address multiplexing. So, the same address lines are used to perform multiple operations. One is performing operations to access a particular row and we perform, we use the same address lines to send the column address. So, effectively we consider the address into 2 parts. First part is specifying the row address the second part is specifying the column address.

Once we have that we select a particular row and that is transferred to the sense amplifiers and using the second part of the address we select a required portion from the sense amplifiers. This is the way in which we send the address from the memory controller to the DRAM device. To access data from our bank we need to perform three basic operations. The first operation is pre-charge.

A pre-charge is an operation which ensures that the bank that we are going to access will be put into a voltage level, so that our subsequent operations can be performed without any issue. And this PRE is required before any access to that particular bank. So, then we will issue an ACT or activate command which actually selects a particular row based on the data available, in data is given to the, based on the data that is given to the row decoder. So, row decoder gets a data and that activates a particular word line. So, we read that word activating the word line reading the data from the word line to the sense amplifier is the part of this activate operation.

So, once the data is read to sense amplifiers, then we actually perform the final operation which is column access. It can be for a read operation or a write operation. So, it transfers the data to the memory controller or it transfers the data from the memory controller. So, you may get a doubt that why we have to activate a row when we want to perform a write operation, writing some data from the memory controller to the memory to a particular bank. When we are performing a read or write operation from the memory controller, first thing we have to do is, we have to get the corresponding row and the hardware page on to the sense amplifiers because we know that the operations we perform at a bank level is at a granularity of a row.

So, even when processor wants to write 32 bits of data to the memory. So, we have to first load the entire row to the sense amplifiers and then get this particular data from the memory

controller and then through the write buffers, input output buffers, we will write it to the sense amplifiers and from the sense amplifiers we take this entire data and put it back into the DRAM bank. So, as a result whether you perform read or a write, activate is required. And before any act operation we have to ensure that the entire bank is in a stable voltage level a proper voltage level for that we require a pre-operation.

So, in summary, the three key operations we perform for any access to a DRAM bank are the pre-charge, activate and column access. In addition to these three operations in order to retain data in the DRAM bank we need to perform 2 more operations. One is a restore operation, the other one is a refresh operation. When we perform an access to a particular row in the bank, we have to restore the entire data of that row by writing this data from sense amplifiers back to the DRAM array.

So, this is called as a restore operation. This is required as all the DRAM accesses are self-destructive, whenever we read from a DRAM bank the data will be lost. So, you have to restore it back. And note that whether you perform a read operation or a write operation, DRAM read is required from the corresponding bank. And the last operation is the refresh. We know that the DRAM cell consists of a capacitor and the data in the DRAM cell is represented in terms of charge in the capacitor, but the DRAM cells will lose the charge over a period of time due to leakage.

So, once it loses the charge beyond certain value, then the DRAM cell is going to represent a wrong value. In order to retain the data, so we have to periodically refresh each of these DRAM cells. So, periodic refill or a refresh is nothing but you read data from a row to the sense amplifiers and write it back. The moment you perform an read and a writing back, again the DRAM cells associated with that DRAM row will be maintaining its full charge and it can retain the data for some more time. Typically the retention time in DRAMs depends on the temperature of the DRAM. Because the charge is leaked based on the leakage and the leakage is dependent on the temperature. Higher the temperature the DRAM cells leak quickly. If the temperature is low then it takes more time to leak.

So, in other words if the temperature is low the retention time is higher, if the temperature is high the retention time is low. If the temperature is below 85 degrees centigrade the retention time is 64 milliseconds and if the temperature is above 85 degrees centigrade then the retention time is 32 milliseconds. Refresh operation consists of reading a row from the bank

and writing the data back to the bank, while we are performing this read and write. So, this entire bank is performing this operation. So, it cannot accept anymore request from the memory controller.

So, that is the reason why the DRAM bank is not available for servicing processor requests while a refresh is going on. And so, that is the reason why there are several techniques proposed for optimizing this refresh overheads and so on. An any way that is beyond the scope of this course, but those who are interested can find lots of interesting material on the internet or IEEE explore or ACM digital library and so on. So, with that I am concluding the basic discussion about the DRAM organization and in the next module we are going to look at DRAM memory controller design and DRAM page management policies and DRAM address mapping techniques.

Thank you.