Multi-core Computer Architecture – Storage and Interconnects

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

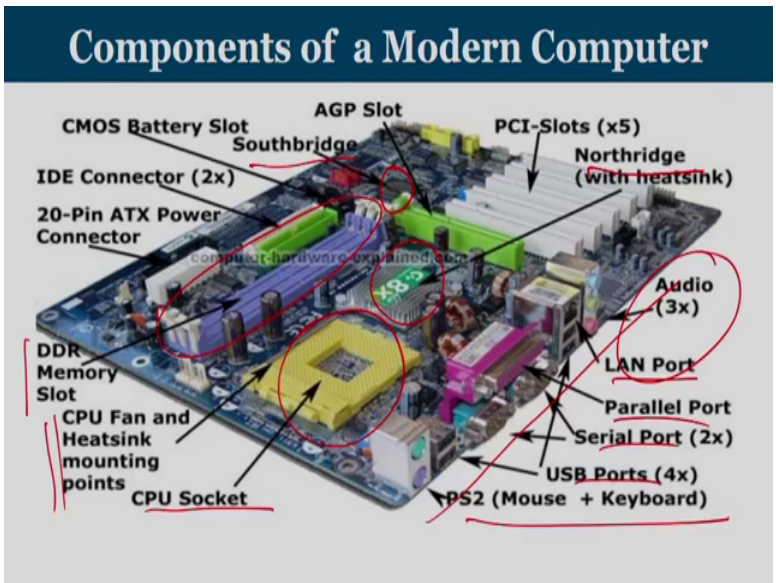
Lecture – 09
Introduction to DRAM System

Welcome to the 9th lecture. Today, we will be focusing our attention into the main memory systems. Typically, we call main memory systems as DRAM as well. We have seen how a cache memory works and what are the various techniques, which are used to improve the performance of cache memory, basically abbreviated as cache memory optimization techniques?

Now, we have to understand since cache memory is very small. The entire program and data of a given application cannot be located fully inside cache memory. And, we have learned that our memory is implemented as a hierarchy, where we have the registers inside the processor as the fastest memory, then we have multiple levels of cache L 1 L 2. And, then we move onto the next level of memory, which is known as the main memory or it is also known as the DRAM system.

We will try to understand, what is organization of DRAM and how the, what is the working principle? And, what are the controlled circuitry that is associated with the DRAM system known as the DRAM controller.

(Refer Slide Time: 01:45)

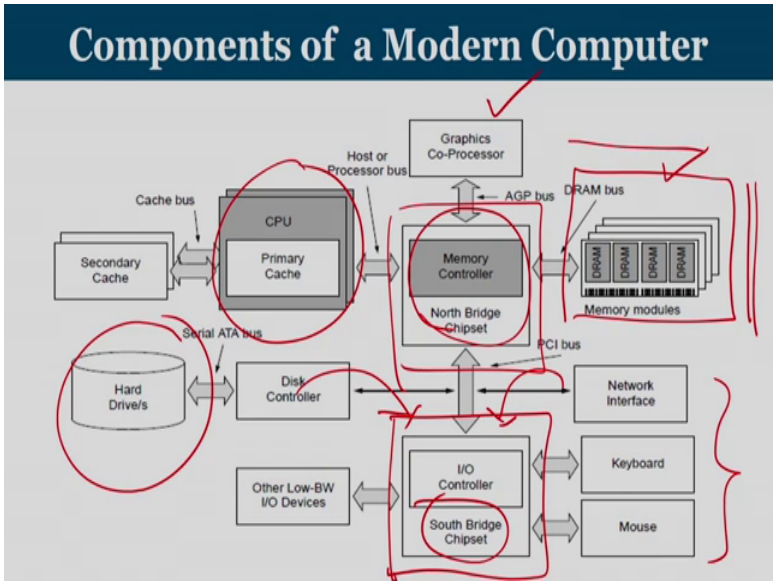


This diagram shows you the structure of a motherboard and how various units are going to be connected into this motherboard. This is an area where the CPU is kept also known as CPU socket and on top of this we will have the CPU fans.

We have various ports that are used for connecting the peripheral devices including the network port, then Parallel and Serial Port, USB Ports, Mouse and Keyboard audio jackets all other things. And, we have this is the area in which the DRAM system is connected or it is also known as DDR memory slots. We can see that there are 2 bridges; one is known as the north bridge and the other one is known as South Bridge.

So, North Bridge and south bridge are the 2 connection points in which high end devices and low end devices are appropriately connected in the system. So, in short your motherboard is going to house your processor chip and on top of that, the heat sink fans are being connected. And, then we have slots for connecting the peripheral devices including the hard disk and then we have separate slot, wherein the DRAM slots are being kept the DRAM DIMMs are being kept and there are 2 bridges which are appropriately connecting high end and low end devices to processors.

(Refer Slide Time: 03:14)



Let us try to understand a more logical picture of this is the way how CPU and the first level caches and second level caches are hosted inside the CPU chip and then these are the DRAM modules.

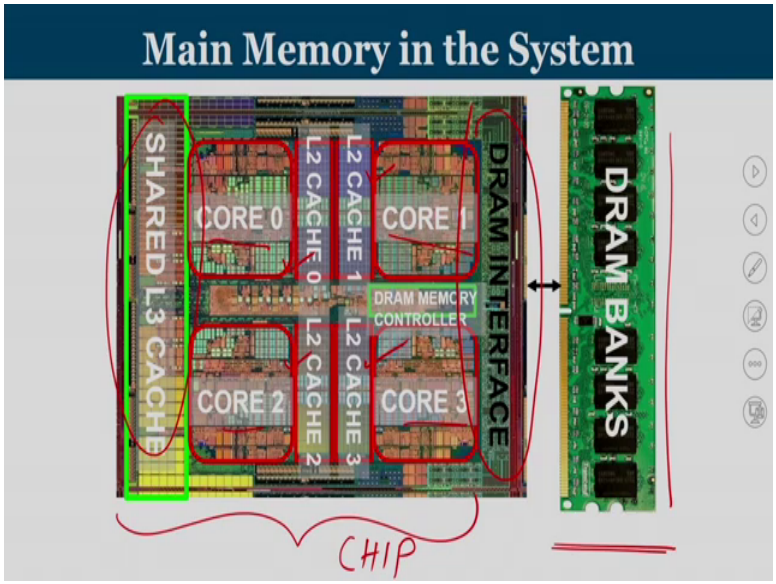
So, typically DRAM is kept outside the chip and then we have your hard disk and peripheral devices plus the network phase is there. And, south bridge is the interface point at which hard disk and other peripheral devices are been connected and north bridge. So, south bridge is connecting to north bridge and relatively high end devices like main memory graphics coprocessor or are all connected to the north bridge chipset.

So, what we try to learn from this block diagram perspective is CPU is operating and relatively very high speed. Compared to the speed of CPU the peripheral devices and hard disk, which are basically electromechanical in nature will be operating at the milliseconds level at least a couple of order of magnitude difference is there in it is operation speed.

So, to interface easily with the high end CPU, the peripheral devices are connected through south bridge. And, then relatively faster devices like DRAM system, which are semiconductor memories they are connected through the north bridge.

So, south bridge is connecting to north bridge all peripheral devices and hard disk are connected to south bridge. And all the semiconductor memories the co-processors graphics co processors all are connected to north bridge and then north bridge is the primary interface point to the processor. Now, our today's lecture is going to focus about how this DRAM system is going to work.

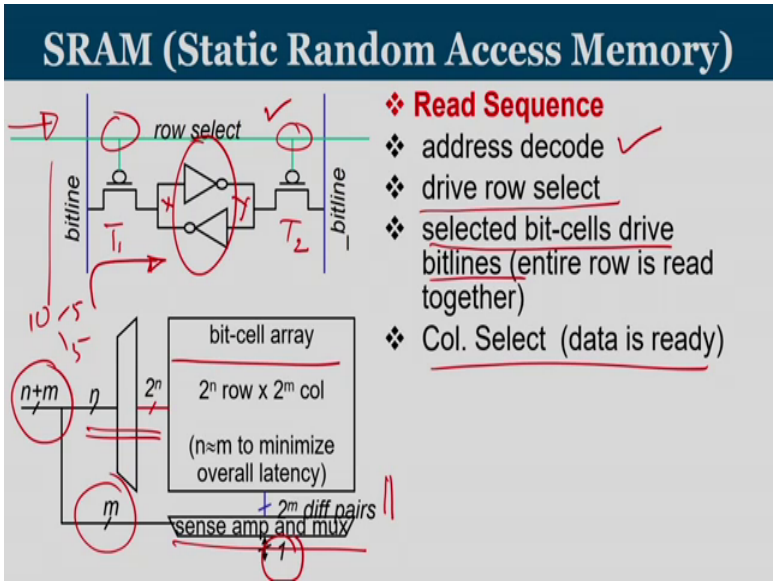
(Refer Slide Time: 05:05)



This is how typically a DRAM unit looks like it is also known as a DIMM and this the layout of the recent multi core processor by Intel.

So, what do you see here is you have many cores that is inside a single chip. So, what do you see here is basically a processor chip and within this chip we have where this cores this example we have 4 cores core, 0 core, 1 core, 2 and core 3 the primary cache or the L 1 cache is inside the core, the L 2 caches are kept near to the core that is the L 2 cache for core 1 L 2 cache for core 0 2 and 3. And, then you have a shared L 3 cache. Now, what do you see here is the DRAM banks you have the DRAM system and this is the DRAM interface. So, processor is going to control the DRAM through the DRAM interface or it is also known as the DRAM controller.

(Refer Slide Time: 06:01)



Now, let us try to understand for your main memory system there were 2 choices either you have to use an SRAM, which is known as Static RAM. And the other one is known as Dynamic RAM, DRAM we will briefly touch upon the properties and characteristics of a SRAM. And, DRAM and we will try to conclude why we prefer DRAM over SRAM in implementation of the main memory system.

This is how an SRAM cell looks like? This cell can store 1 bit of information and we have a row line that is what you see in green color and you have a bitline, what you see in this blue color. And, you have a pair of bit lines for storage of every bit, one is bitline and one is bitline compliment. Bitline compliment is always or bitline bar is always

carrying the negative value, which is stored in this bitline. We have 2 cross coupled NOT gates connected by 2 transistor. So, we have 2 transistors T 1 and T 2, which are connected to the row select. So, once this particular row is selected both T 1 and T 2 is going to be in on position.

So, whatever value is there in the bitline it flows into the not gates. And, here you have 2 points X and Y. X will store the value that is there in the bitline. So, just to summarize whenever we wanted to write a binary value 1 into an SRAM, the row is selected thereby 2 transistors will be in on mode the transistors will conduct whatever is the value in the bitline into the node gates.

So, the value in the bitline will pass through pass through the transistor and it will reach point X and Y is always complement of X. So, through this node gate this value keep on so, circulating X will store your value and Y will store the complement of this value. So, for a very short time this transistor is activated and then this value get stored or stuck inside these 2 cross coupled NOT gates. Similarly, if you wanted to perform a read operation again the word line is activated. Once the word line is activated the value that is stored in point X will move to this bitline.

So, it is like with one direction that is in the case of a store operation or a right operation the value in the bitline passes through the transistor and reach the input of the NOT gate or point X. If, it is a read operation the value that is stored in the X will transfer to the bit line through the transistor so, transistor, basically act as a switch in this context.

Now, how will you implement larger SRAMs? We have to order, whatever binary cell you have seen this binary cell consists of 2 transistors and 2 NOT gates. And, one NOT gate is implemented using 2 transistors. So, altogether we have 2 transistors, which act as switching element and we have 4 transistors that are used to implement the not gate. So, this b cells which each of the b cell consists of 6 transistors, these are organized as the rows and columns such that you get a organized structure for your main memory. And, you have to generate an address the address consists of 2 components, that is n bits which will represent the row and m bits which will represent the column.

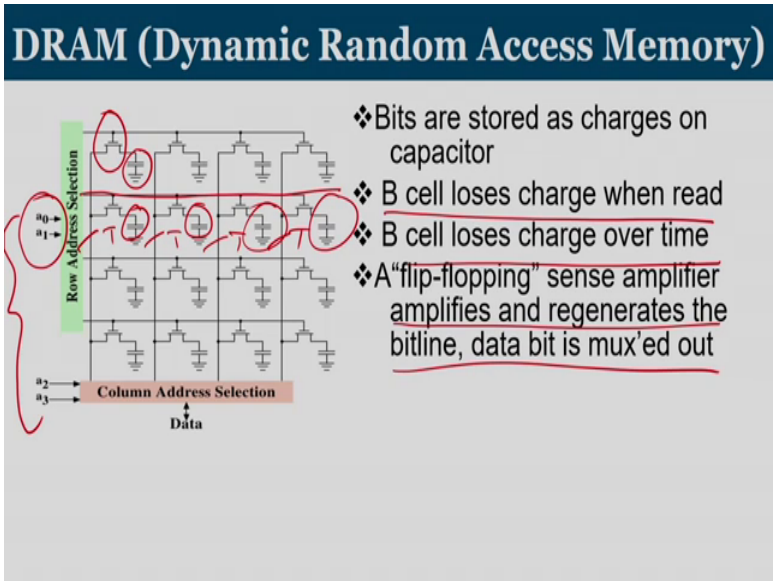
So, you when you have an address say example 10 bits, you can divided into 5 and 5 5 bits for row and 5 bits for column, the most significant 5 bits will uniquely choose one of the row. So, you are using an n 2 2 power n decoder. And, once that row is selected the

entire contents of this row gets transferred to a sense amplifier and then you apply your column bits. So, one of the bit is going to be extracted. So, you get 2 power m values is going to come and when you apply m 1 of the value will come out. This is how you organize a binary cell and from the binary cell we are going to extract the values.

So, in the case of a reading operation you have to first decode the address. So, address is been splited up into row and columns. And, then you using the row decoder the appropriate row is selected and selected bit cells drive the bitlines, the entire row value is read together and then you apply the column select so, that you will get this value.

So, to summarize what is the basic operation of an SRAM or how SRAM is built, SRAM is built with the help of 2 cross coupled NOT gates and 2 transistors. And, we have a word line and a bitline. Once, the word line is activated the transistor will act like a short circuit or an on position. And, then the values will flow from bitline into the not gate and your values are stored there. And, multiple such bits are been organized or combined together in order to form a b cell array.

(Refer Slide Time: 11:02)



Now, let us try to see how DRAM is implemented? The implementation of DRAM is slightly different the number of components used to store a 1 bit value is very less here. We are making use of 1 transistor and 1 capacitor in order to store a single binary cell value. So, bits are basically stored as charges on the capacitor and a B cell lose charge when you read.

So, essentially you have a capacitor, when you are going to charge the capacitor or when they are exist a potential difference between the parallel plates of a capacitor, we call it as logic 1. And, when the potential difference between the 2 parallel plates of a capacitor is less than a threshold value, then we call it as logic 0.

So, if the capacitor is charged; that means, a value 1 is stored there, if the capacitor is not charged then the value 0 is being stored there, but the property of this capacitor is when you are you trying to read a data, then capacitor discharges. So, if there sufficient amount of potential that get discharge, then the sense amplifier knows that the value that was stored in the capacitor was 1.

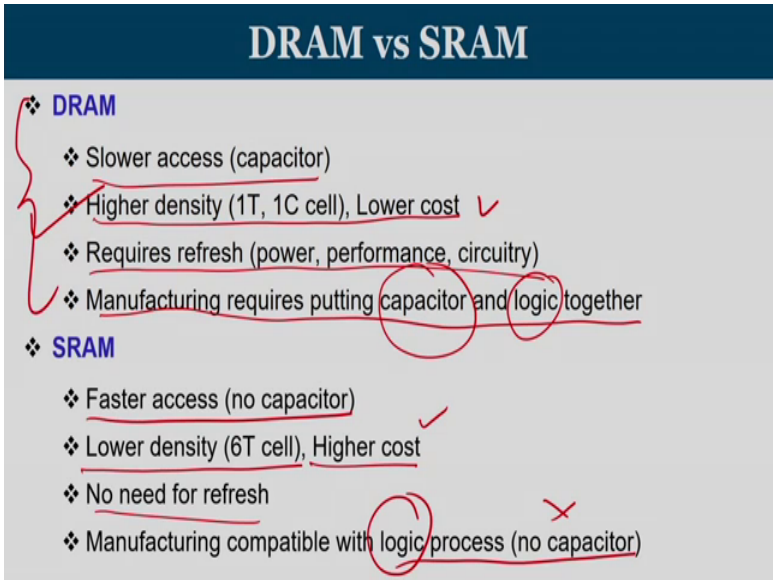
Since, the charge is already lost a reading the B cell will lose charge when you read, because of the leakage property of the capacitor, whenever you are not doing anything also over a period of time the capacitor loses the charge.

So, since we are going to implement your 1's and 0's charges over parallel plates of a capacitor. In reading we will discharge the capacitor and capacitor discharges over time due to D K property as well. Now, you have a flip flopping sense amplifier, that amplifies and regenerates the bitline and data bit is multiplexed out of it. You can see from this diagram that you are going to use a 4 bit address out of which 2 bits are used uniquely select a row.

So, consider a k is that this particular row is selected. Once, that row is selected that transistor that you see there all the transistors will be on position; that means; they will permit whatever is the value in the corresponding bitline. These are the bitlines, the corresponding bit lines will flow and the charges get stores in the capacitor, there is a process of storing something.

Now, when you wanted to read what you have to do is again activate the appropriate word line the capacitor discharges. So, if the capacitor already has a high potential between it is parallel plate, the sense amplifier will get voltage thereby it recognizes that the value that was stored on the B cell was 1. If, there is no discharge that is happening; that means, already there is no charge between the parallel plates meaning that the value that is stored there is 0.

(Refer Slide Time: 14:05)



So, a brief comparison between DRAM and SRAM, DRAM is relatively slower because in order to know what is the value that is stores across the capacitor, we have to permit the capacitor to discharge. So, the discharging time of capacitor is not under electronic timing. So, it is slightly a slower access, but it is having high density that makes it suitable for large memory systems like a present a main memory. Only 1 transistor and 1 capacitors cell is require thereby making it is implementation a lower cost affair. Because of the leakage property of the capacitor you requires a refresh circuitry, that is power hungry and it will affect the performance as well.

So, and the manufacturing requires a bit of extra overhead, because you are going to put a logic that is a semiconductor logic and a capacitor going to be put together. Moving onto SRAM, since there is there is no capacitor that is involved all the components are electronic components, because we have only transistors accessing from an SRAM is relatively faster. And, it serving low density because to store 1 bit require 6 transistors, thereby making it of higher cost affair.

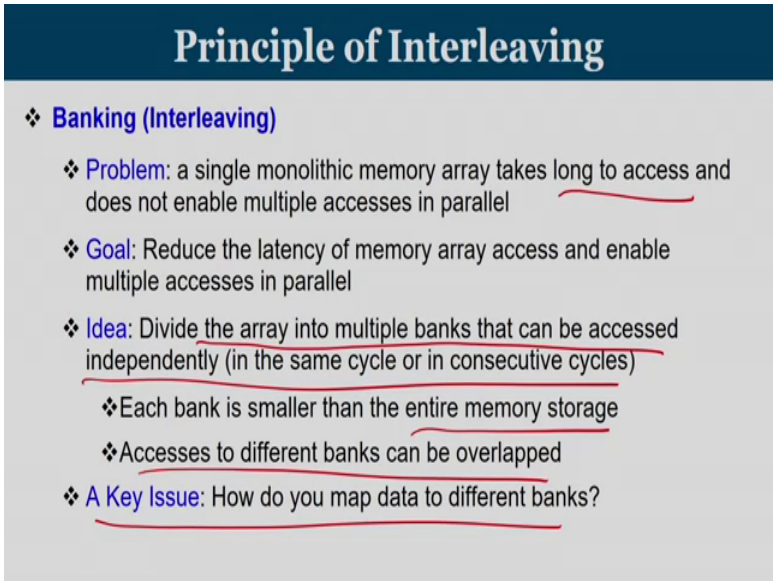
Since, there is no capacitor that is involved there is no component that is having a leakage property, no refreshing is required and manufacturing is compatible because all the units are basically your semiconductor logic, there is no capacitor that is involved.

This is the basic difference, because of the higher density. We are going to make use because your main memory should have really large size of the order of gigabytes and the chip should be small enough so, that I can accommodate it in the motherboard. So,

the high density make our main memory system to be built with capacitors and transistors. So, DRAM system is basically used in order to implement the primary memory of current day processors.

Let us try to see before going into the implementation structure of DRAM. We will try to understand, what is the principle of interleaving? And, what is the role of interleaving? Or, why we should understand the concept of interleaving, before designing a DRAM system.

(Refer Slide Time: 16:20)



The interleaving is also known as banking. The problem that we face is when you have a single monolithic memory array, it takes lot of time to access them because you have a bulk of memory and then you have 1 decoder. Let us say you are going to use a 4 GB of memory, which is organized as a single monolithic unit.

This 4 GB memory to address it you require 32 bits. So, we have a 32 to 2 power 32 decoder, that uniquely identify one of these bits memory locations. So, that will take a relatively longer access. So, in order and it is not possible to have 2 parallel access together because you are only 1 decoder.

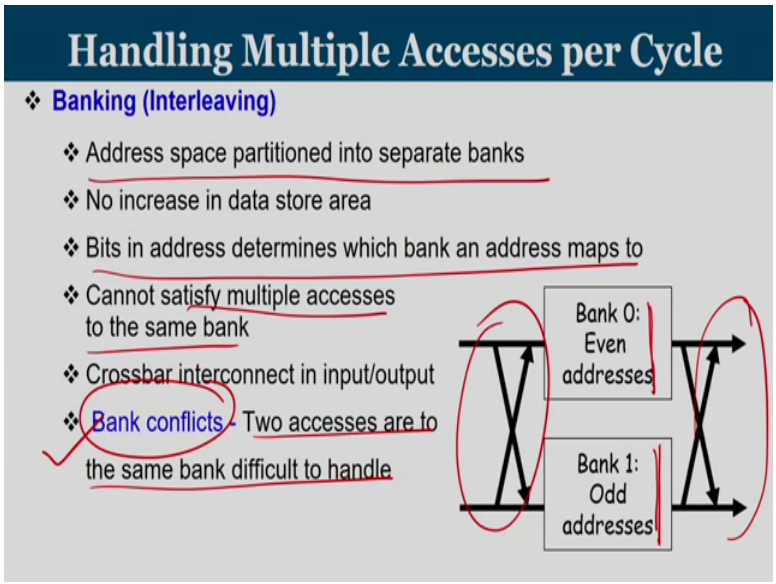
So, the goal while going for banking or interleaving means, reducing the latency of memory array access. And, enable multiple access in parallel. How it is done? Divide the

array into multiple banks, rather than keeping your 4 GB as a single monolithic unit I can divide them into 1 GB into 4.

So, the entire 4 GB of your main memory is divided into 1 GB structure. Similarly, we have 4 such 1 GB structures and these 1 GB structures can be accessed to parellely. So, divide the entire memory into multiple banks that can be accessed independency either in the same cycle or in consecutive cycle. So, each bank is smaller than the entire memory storage and access to different banks.

So, whenever we are accessing 1 bank, we can access or we can make necessary facilities such that I can access the other bank immediately after the data transfer of the current bank is over, but the key design issue is how do you map data to different banks? This is the principle by with interleaving works.

(Refer Slide Time: 18:13)



Now, how will you handle multiple access per cycle? So, this DRAM is going to be acting as a primary memory structure for all processes and cache. We are already in multi core environment, where multiple processes works parallely with their own private and shared caches. And, there is high possibility that these processors will have their own cache misses reaching to DRAM may be at the same time. Or in short how will the DRAM manage, when multiple memory request are going to come to DRAM.

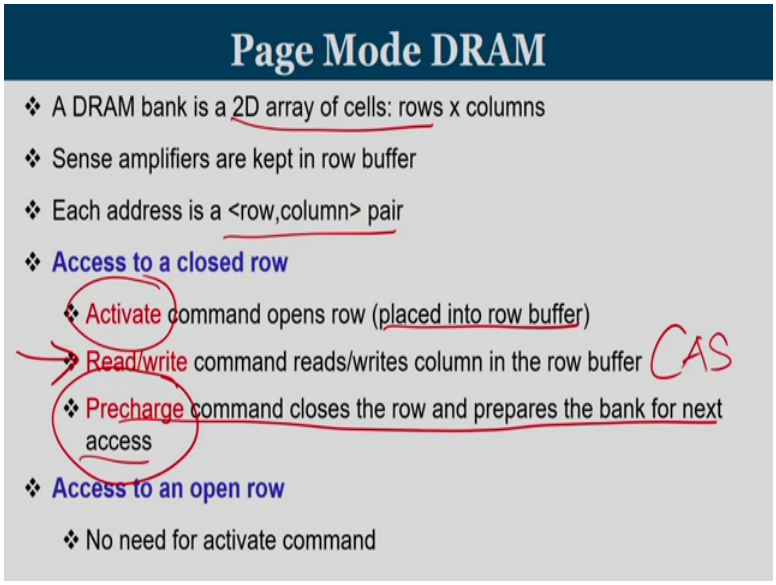
So, what it is done in the case of the interleaving means address space is partitioned into separate banks. So, whatever is a existing storage that is been partitioned into separate banks. And, bits in the address let us say it is a most significant bit or the least significant bits or somewhere in middle bits, these bits will tell you whether you belong to bank 1, bank 2, bank 3 like that. One simple way of implementing banking is what is given in this diagram. All the even addresses can be considered as mapped to bank 0 and all the odd addresses can be considered as mapped to bank 1.

So, when you have 2 request that is coming. Let us have one of the request is for an even address, that is ending with 0 then the data is physically located in your bank 0 in this case. Let us say, the other request that is coming at this point is an odd address that is the least significant bit of the address is 1 and it will be located in bank 1.

So, if you have 2 read write ports and the addresses of our DRAM access are belonging to do different banks independently I can service them, but we cannot satisfy multiple access that are to the same bank. So, we will use a crossbar kind of a structure depending on address, whether it is go to bank 0 or bank 1 both in the output and in the input structure. Two access that are to the same bank are been termed as bank conflicts.

So, whenever there exist a bank conflict between 2 address the meaning is 2 addresses are mapped to the same bank. So, we cannot handle such cases. Moving further DRAM consists of a page mode structure.

(Refer Slide Time: 20:39)



Like DRAM bank is a 2 D array of cells which consists of rows and columns and we keep a sense amplifier at the row buffers and each address is basically a row and column pair.

Now, there are 2 terminologies that will (Refer Time: 20:53). So, one is known as closed row, the other one is known as opened row. So, we have learned that DRAM consists of rows and columns. There can be case like when you give an address the row is no longer existing in the row buffer. So, first we have to take the entire contents of a row into the row buffer then give the column bits such that you can extract the data.

A scenario where a given row is not already kept in the row buffer is known as a closed row scenario. So, whenever you have a closed row scenario. First, you have to give an activate command such that the appropriate row is placed on to the row buffer. And, then you have a read or write command such that they appropriate column from the row buffer is chosen and this precharge command closes their row and prepare the bank for next access.

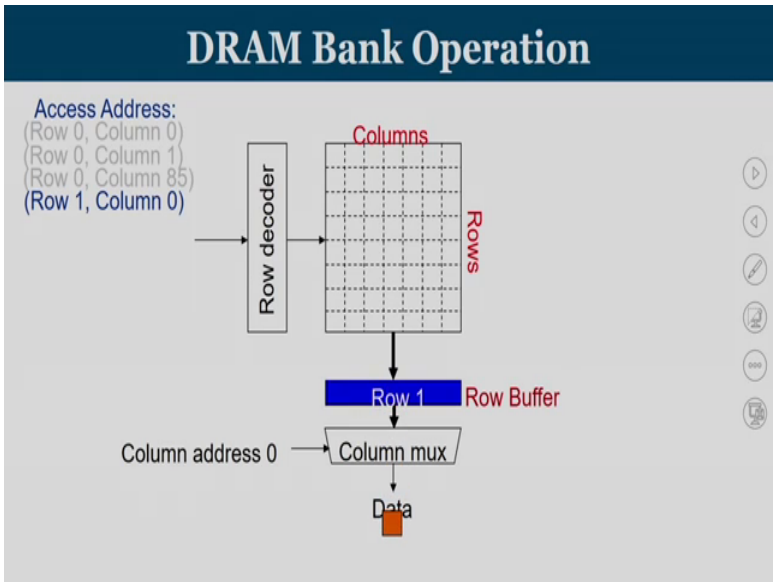
Since, we have learnt that in the case of a DRAM structure. When, you give a read operation the contents of the capacitor had been discharging through the bitline and reaching row buffer. So, that now this capacitor are no longer charged. If, this is called a destructive read any read operation that you carry out on a capacitor, will discharge the charges that exist over the capacitor plates leading to a 0 potential layer.

So, any reading will exactly delete the value that is stored the logic that is stored there so, and the values are now there in the row buffer. If, I wanted to activated new row buffer, the existing value that is there in the row buffer should be stored back such that my corresponding row capacitors are now storing the value again. The operation of storing the condense in the row buffer back to the appropriate row is known as precharge.

So, there are 3 basic operation that is involved. The first one is known as activate where the contents of row are moved to the row buffer. The second one is known as precharge where the condense of row buffer are going back to the appropriate row. And, then we have a read or write command which is also known as column address strobe, where your data is already available in row buffer give the corresponding column address. And, that will tell you what data is to be extracted.

Now, ones you have an address which is already opened; that means, the corresponding row is already open, you need not give an activate command only a read or a write command is needed.

(Refer Slide Time: 23:40)



Let us try to understand a DRAM bank operation. We have seen that, the DRAM consist of rows and columns and we have a row buffer. Now depending on the addresses that are coming; we have to see that how this row buffer is getting populated. Initially the row buffer is empty as you see from the diagram.

Let us see CPU is going to give an address row 0 and column 0. So, first we give row 0 there to the contents of row 0 are activated such that the contents of row 0 will reach the row buffer. So, now, the row buffer is currently holding row 0 give the column number.

So, the sequence is first you give the row number; the contents of the entire row will reach the row buffer, give the column number. So, here column address is 0. So, the 0th address is being taken off. So, the data will move from the row buffer. So, once you give the column address 0. The 0th columns value that particular data will get transferred row buffer over the column multiplexer to the data bus. Let us say the very next request that CPU gives is a row 0 column 1. Here, the raw is already there in the row buffer and we call this scenario as a row hit. Once, you have a row hit then there is no need to give an activate command, because already the raw is there give the column address.

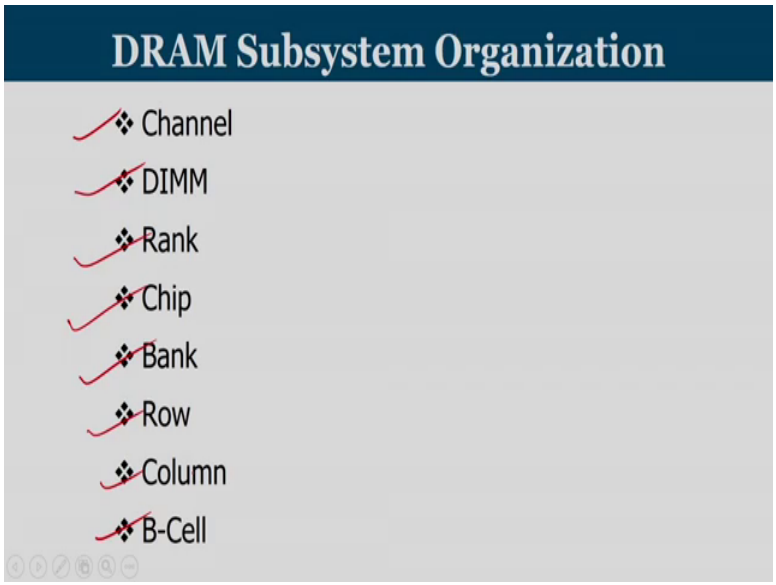
So, here column address number is 1. So, the contents of one are been getting transferred. Let us now consider the third address that is coming to CPU from the CPU into the DRAM. Row 0 and column 85, this is also a row hit, because Row 0 is already there inside the row buffer. So, you only need to give the column address, column address 85, the contents of the 85th column is been transferred.

Now, we get a scenario where CPU is going to give Row 1 and column 0. So, whatever is a Row that is kept open in the row buffer is different from the row address of the incoming address. This scenario is known as conflict, it is known as row buffer conflict because the requested row is different from the existing row in the row buffer.

So, in this case the value from the row buffer has to be stored back and that process is known as precharge. So, the contents of Row 0, which was already there in row buffer is been transferred back into the corresponding Row. So, at the time of activation of row 0, now the capacitors no longer hold the data, only the Row buffer has it.

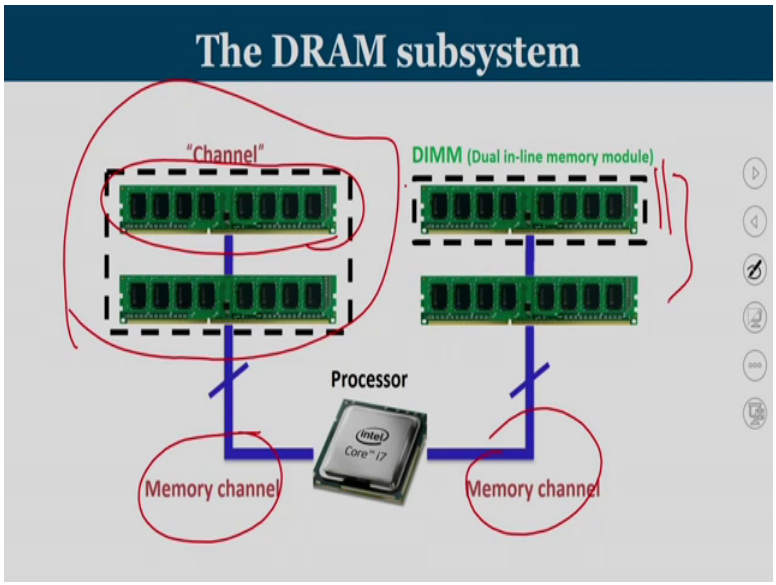
So, before the row buffer is going to get loaded with a new row existing values have to be stored back. So, now, I am going to give activate command for Row address 1. So, the row 1 will come apply the column address. So, column address 0 will get transferred into this.

(Refer Slide Time: 26:45)



We will now look into the organization of the DRAM. DRAM, itself is consisting of multiple hierarchy consisting of channels, DIMM, rank, ship, bank, row columns, and B-cells. We will try to see what each one of them is.

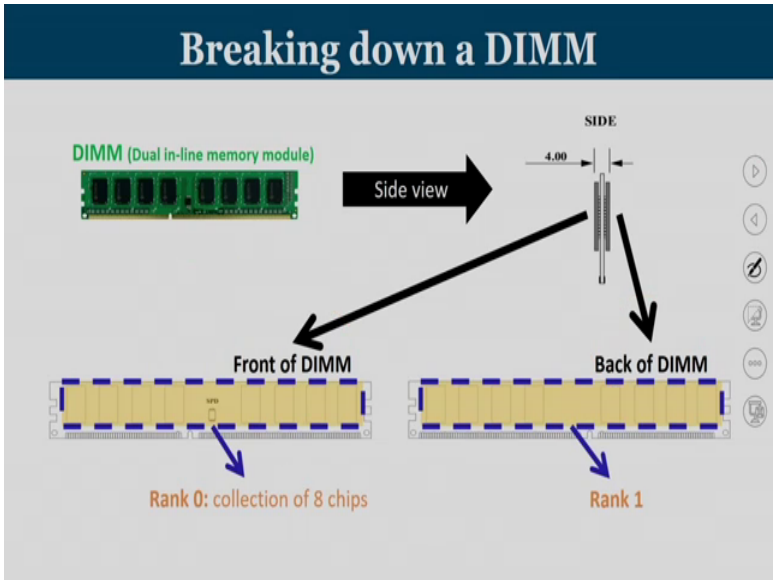
(Refer Slide Time: 27:06)



So, when you have a processor? The processor may have multiple channels. Means, it may have multiple controllers and multiple channels is also known as multiple address buses or data buses. So, if a processor has N set of buses, then typically it has N channels. So, this is one channel and other one, this is one channel and other one is going to be the second channel.

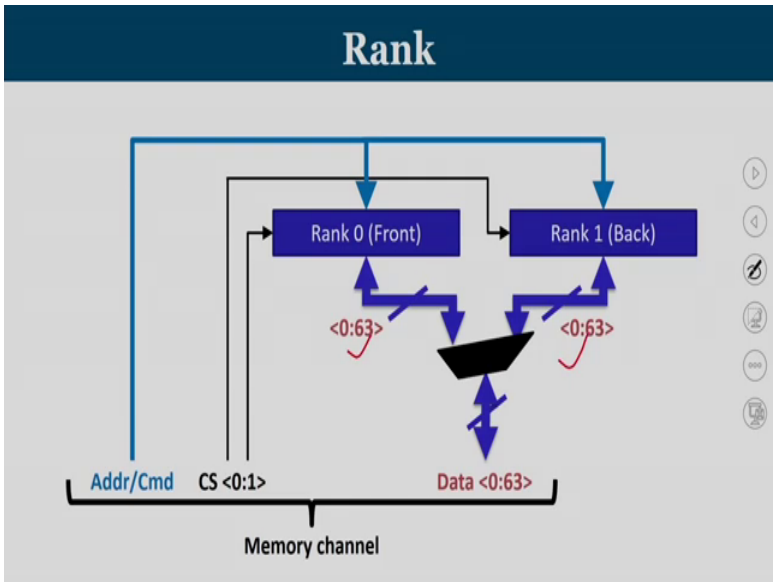
So, this is memory channel number 1 and channel number 1 and this is channel number 2. So, 2 channels are there and each of this channel consists of multiple DIMMs. So, what do you see is a structure here that is called a DIMM, the entire structure is known as a channel. So, here I have 2 channels and each channels has stood in. So, typical structure what you see as main memory is known as multiple DIMM is known as a single DIMM, this is what is a DIMM. I can create multiple such DIMMs and put together in one DRAM slot.

(Refer Slide Time: 28:15)



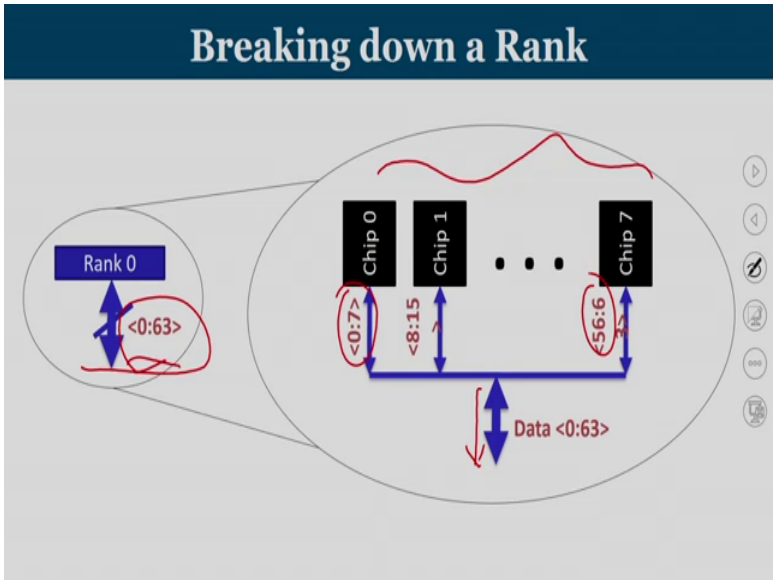
Now, once you reach DIMM, the DIMM has a front side and it has a back side. So, front side and back side is there, this is our DIMMs looks like the front side is known as rank 0 and back side is known as rank 1.

(Refer Slide Time: 28:34)



Now, you take a rank, when you give it as a rank you are going to give a common address to a rank, and either rank 0 or rank 1 will give the corresponding data that you need.

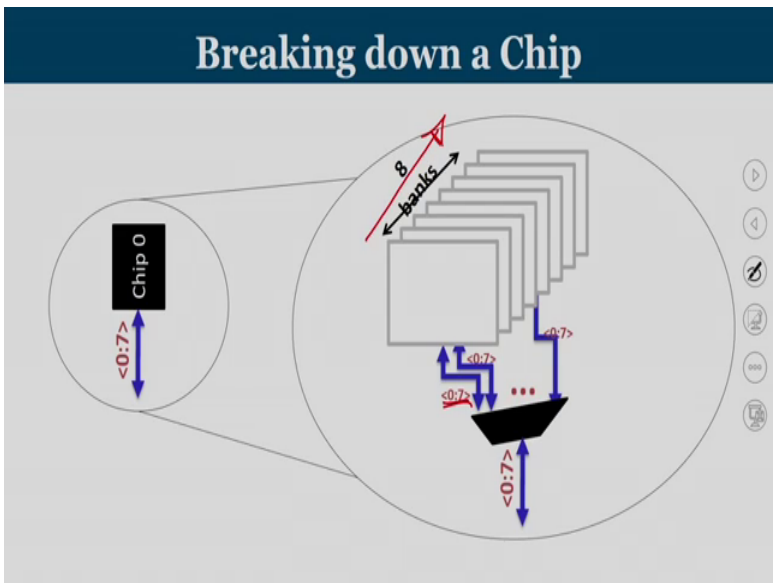
(Refer Slide Time: 28:51)



Now, each of the rank that you take let us say these rank is going to supply to a 64 bits of data to me. We can consider that this rank consists of 8 chips with numbering 0 to 7. Each chip is going to give a small fragment of the entire 64 bit, I have total 64 bit of data that has to move through the data bus out of which chip 0 will give me bit 0 to bit 7, chip 1 will give me bit 8 to bit 15. Like that chip 7 will retrieve me it 56 to bit 63.

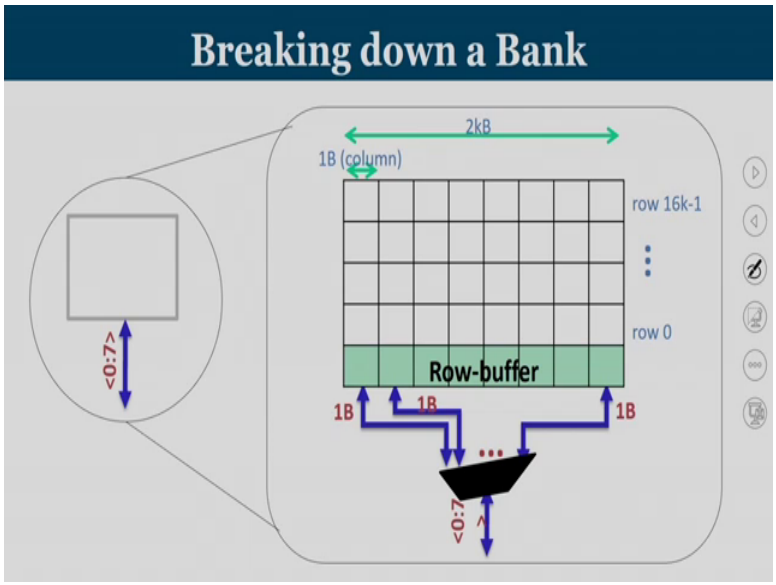
So, in short your rank is going to give total of 64 bits and these 60 bits are not generated from a single chip, there are 8 chips that is being kept, each chip will give a sub component of the entire 64 bit data. So, the data that is given by 8 chips together will form 64 bit of data.

(Refer Slide Time: 29:58)



Now, if you take each chip, let us say this is the first chief each chip is going to retrieve me 7 bits, but I have a 3 D structure now and that is known as banks. So, each chip consists of multiple layer of such kind of rows and columns and together the 3 D structure is known as bank.

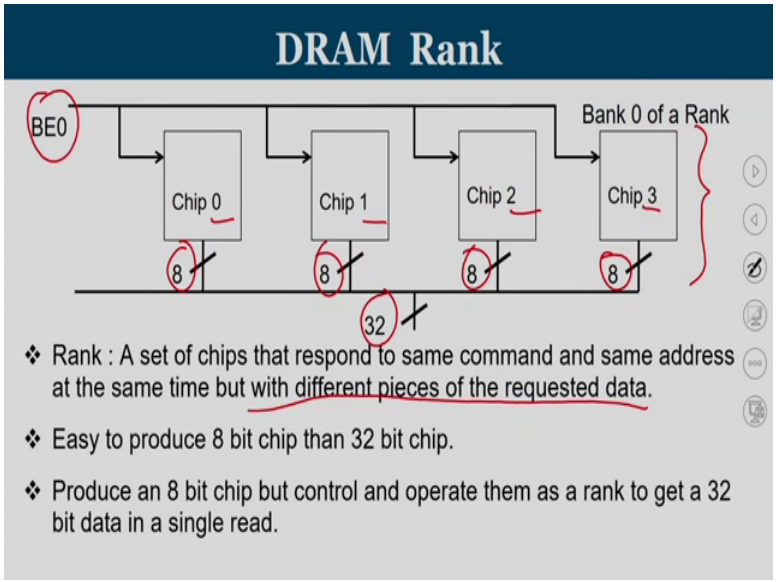
(Refer Slide Time: 30:21)



Now, breaking down a bank each bank consist of rows as well as columns. So, what we have seen is the given DRAM address this address should mention to which channel it belongs to which DIMM is belongs to. So, channel consist of multiple DIMMs a DIMM consist of multiple ranks, typically 2 ranks rank 0 that is a front side of the DIMM and rank 1, that is back side of the DIMM. 2 ranks and each rank consist of multiple chips and each of the chip will contribute a fraction of the big data that is going to be transmitted through the data bus. And each of this chip further has a 3 dimensional layering on it and that is known as banks.

So, if you have 8 layers of storage inside each chip we call it as an 8 bank DRAM system. Now, looking into each of this bank you can see rows and columns and the meeting point of rows and columns will give you 1 bit of information.

(Refer Slide Time: 21:23)

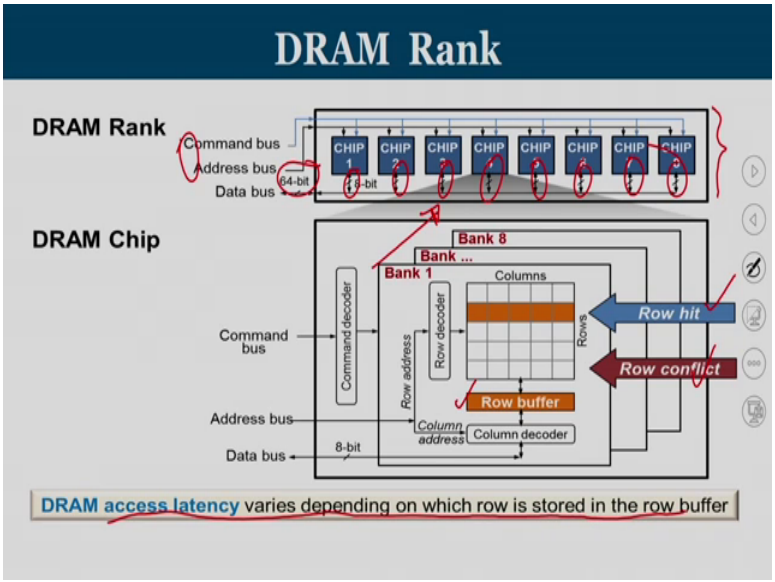


So, what is typically a DRAM rank? Rank is a set of chips that respond to the same command and same address at the same time, but with different pieces of requested data. Like what we can see you are giving a common command here and this entire structure is known as the rank. This rank consist of 4 chips with chips numbered as chip 0 1 2 and 3. Once you give a command each of the chip will return say 8 bit of data and together adding up all this 8 bits we will be getting 32 bits of data.

So, in short rank consist of a set of chips that respond to this same command and same address at the same time, but with different pieces of requested data. So, the peculiarity of the chip is each chip is going to give a different piece of data for the given address. So, it is an easy to produce 8 bit chip rather than 32 bit chip. So, each chip will generate only 8 bits and the organization is in such a way that at the end you get the 32 bit data together.

So, produce an 8 bit chip, but control and operate them as a rank together to get a 32 bit data in a single read.

(Refer Slide Time: 32:43)



This is how the organization is going to be, you can see that this is what you see it as a rank. And, the rank consist of 8 chips, that is what we see here 8 chips are there and whatever is the command and address you give the same command and address will goes to all the 8 chips. Each of the 8 bit is going each of the chip is going to return with a 8 bit data. And, together we will be getting a 64 bit data out.

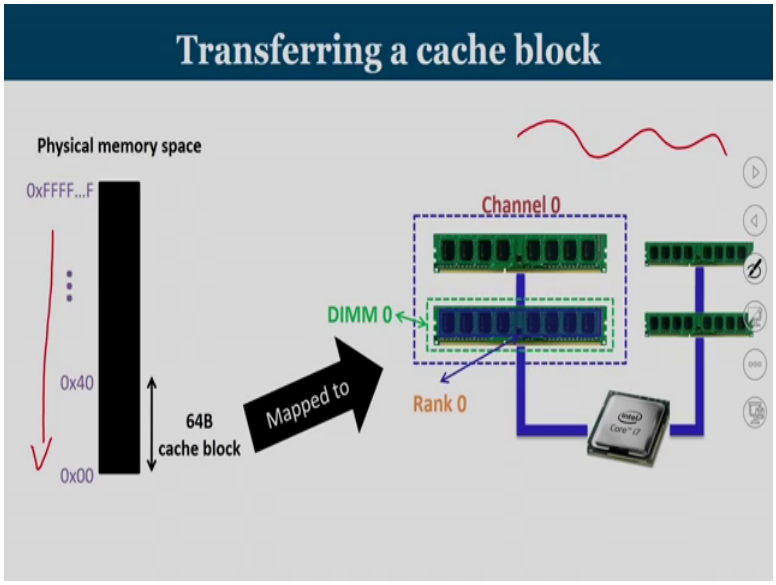
And looking deeper into each of the chip we can see that there are 8 banks, it is a 3 D structure and the meeting point of banks within this banks we have rows. And, columns and then you have row buffer each bank has it is own row buffer.

So, depending on the addresses we will be able to extract the data from the appropriate row buffer. So, there can be a row hits as well as row conflicts. And, DRAM access latency varies depending on which row is stored in row buffer. Now, we try to correlate with the principal of interleaving that we have already learnt. In the concept of interleaving we are telling if 2 addresses are coming to different banks and if you have multiple retrieve ports, it can be easily handled, means I could have achieve paralysm.

Similarly, if 2 request that is coming to the DRAM system is belonging to 2 different banks. So, that I can activate 2 banks parallelly I can record this data by appropriate activate command the data of the corresponding row will reach the row buffer. And, from the row buffer you give the column address. And, once the column address is ready I can extract the data, but if you only one data channel only one data bus that is available, I cannot give the read or write signal together, but I can always keep my row buffer ready

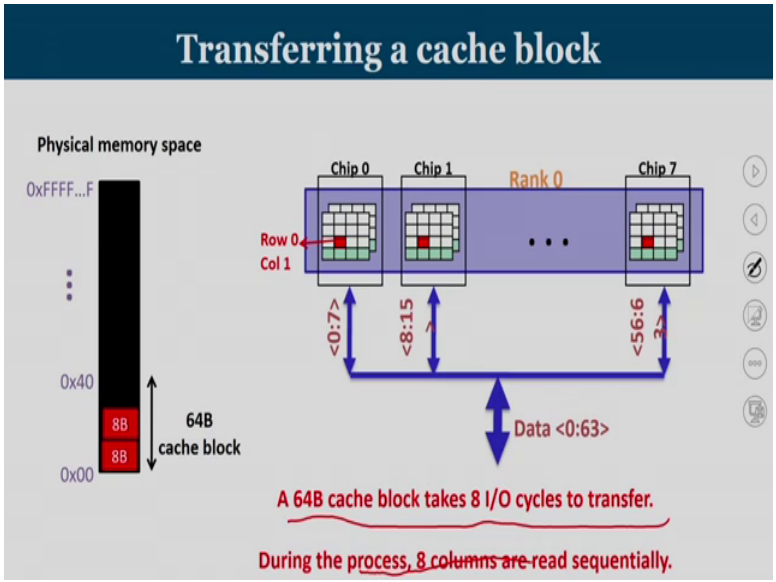
with the appropriate data. Such that once data transfer is over in bank 1, data transfer operations are ready in bank 2. So, that bus is continuously occupied.

(Refer Slide Time: 34:52)



Now, we will try to see how a transferring that happens inside a cache block? So, when you have a physical memory this is the logical view of the physical memory whatever we have seen. And, this is how the physical organization of the DRAM structure. Consider that from this main memory you are going to have to transfer 64 byte cache block if; that means, any data that is moving from main memory into cache memory is of the size 64 byte, my cache block size 64 bytes. So, how I am I going to extract these 64 bytes 64 bytes cannot be taken together, where is 64 byte located inside my main memory.

(Refer Slide Time: 35:37)



So, consider this case you have a cache memory block size of 64 bytes. So, when cache memory request for a word is typically a block of size 64 bytes is getting transferred and this is how the data comes? First I give row 0 and columns 0. So, across all the chips row 0 column 0 is selected and this is going to give me 1 byte of information that is 8 bit each one is going to give me 8 bit of information or exactly 1 byte of information.

So, at the end I am going to get 8 bytes of information chip 0 will return 8 bit chip 1 will return 8 bit. Similarly, chip 7 is also going to return 8 bits each of them I say byte. So, altogether I have 8 chips 8 bytes of data. So, I got 8 bytes of data. Next, I will appropriately increment the columns. So, the row buffer of bank 0 of chip 0 will carry this first 8 bits. Similarly, the contents of chip 1 that is row 0 of chip 1 bank 1 will store the value in the corresponding row buffer.

So, every row buffer across bank 0 of all the chips will carry the corresponding row and then you vary across the column number. So, once you put column number 1 you get 8 bits each and these 8 bits together will constitute, my next 8 bytes making it 16 bytes. Similarly, I have to carry out this operation. So, a 64 byte cache block takes 8 I O cycles to transfer. So, during this process 8 columns are read sequentially.

So, even though we consider main memory as a linear unit today we have seen that the main memory is organized as channels, DIMMs, ranks, chips and banks and then followed by rows and columns. So, when you wanted to transfer a block of data to cache memory, it is not coming from a single point. First, we give a row and column that you

give you 8 bytes of data. These 8 bytes of data are transferred together, because 8 bytes means 64 bits. Let us say your data bus width is 64 bit; that means, from main memory to cache memory only 64 bits can flow at a time; 64 bits will travel together and then that is what you get.

Once you get your 64 bit then your first set of data is ready that is 8 byte is ready, but for the cache memory to continue it is operation on entire block has to come only 8 byte out of the 64 byte block has reached. So, I have to bring another 7 more I O cycles. So, increment the column number. So, the row is already there if all are row hits increment a column number transfer the next 8 bytes, increment the column bit my 1 more transfer the next 8 bytes write that you require 8 such I O cycles in order to complete the operation.

So, with this we come to the end of today's this lecture, where we have learned about, what is basic structure of a DRAM? The difference between SRAM and DRAM and how basically DRAM is organized into multiple hierarchy consisting from channels up to the rows and columns. And, basically with the help of an example you saw how cache memory block copy happens from the DRAM. The next day we will try to see that how a DRAM controller is working and how address mappings are done on the DRAM structure. Feel free to post if you have any queries.

Thank you very much for the day.