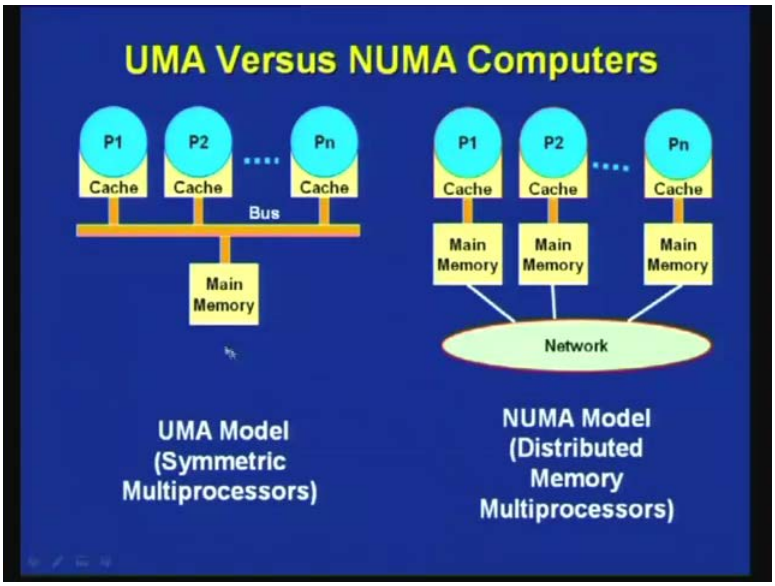
High Performance Computer Architecture
Prof. Ajit Pal
Department of Computer Science and Engineering
Indian Institute of Technology, Kharagpur

Lecture - 40
Distributed Memory Multiprocessors

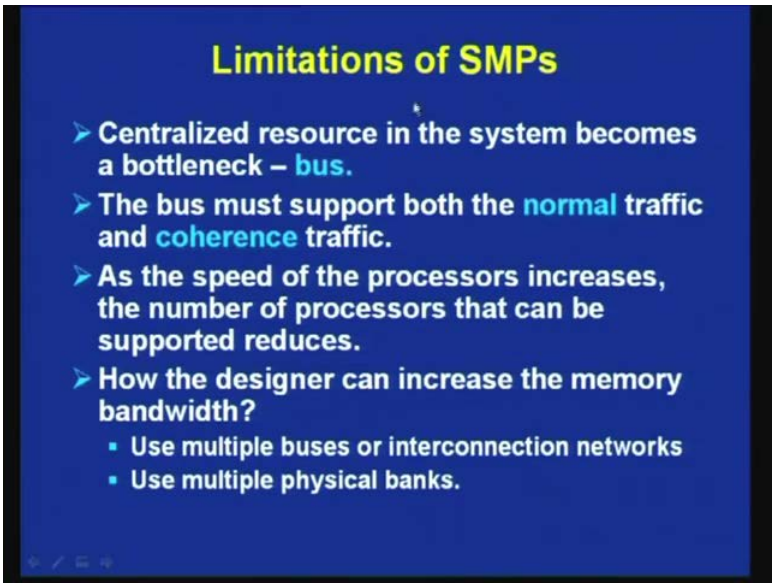
(Refer Slide Time: 00:43)



Hello and welcome to today's lecture on Distributed Memory Multiprocessors. You may recall that we have two basic models; one is known as uniform memory access or this is also used for symmetric multiprocessor, where there is a shared memory connected to the bus, and each processor has equal axis time. Now, on the other hand there is another model is known as NUMA model, NUMA stands for Non Uniform Memory Access, and this NUMA model you can see that you have got separate processors and the memory is not shared it is not common.

So, it is being distributed, so in this case memory is distributed that is the reason why it is called distributed memory multiprocessor. And each processor is having a part of the main memory, with it associated with it connected through a bus, and then each of them is connected to the network.

(Refer Slide Time: 01:41)



So, we have seen the limitations of symmetric multiprocessors, in my last class we have seen there is a centralize source in the system becomes bottle neck. So, whenever there is a centralized resource and which is used by all the processors, then it becomes a bottle neck. So, in the in case of your symmetric multi processor, we have seen that shared bus is a bottle neck, because as the speed of the processors is increased, or if the number of processors is increase, than the traffic on the bus increases.

And, as a consequences it puts a limit on the number of processors that can be connected to this bus, so and the traffic there are two types of traffic, we have seen. The bus must support both normal traffic; that means, when you are reading memory I mean when there is a cache miss, you have to transfer data from the main memory to the cache that is your normal traffic you can say.

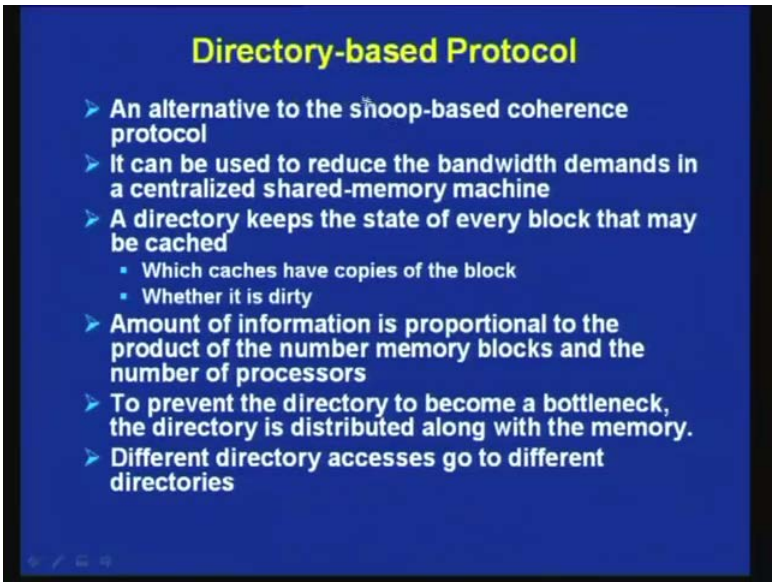
On the other hand there will be other traffic, because they maintain coherence, so those are known as coherence traffic. And as the speed of processors increases the number of processors that can be supported reduces, as I have already mentioned. because the bus has a limited bandwidth. So, with the limited bandwidth the number of processors that can be supported is restricted; now question arises how the designer can increase the memory bandwidth.

So, the memory bandwidth can if the memory bandwidth can be increased, then the number of processors in the system can be increased how can it be done, one possible

approach is to use multiple buses. So, whenever you use multiple buses then; obviously, the traffic on the bus, on each bus reduces and that may help you to increase a number of processors.

Another, alternative is to use multiple physical banks, so whenever we have already discussed the use of multiple physical banks, and whenever you have got multiple physical banks. Then; obviously, the reading from different, I mean from different banks can be done separately, and that will also help you to increase the number of processors.

(Refer Slide Time: 04:09)



So, this is these are the limitations, so in search of better alternative, we have another new technique, as we merged that is known as directory based protocol, where you have got distributed memory. So, in a distributed memory system, we shall use a special type of protocol, which is which is known as directory base protocol, so we have seen in symmetric multi processor.

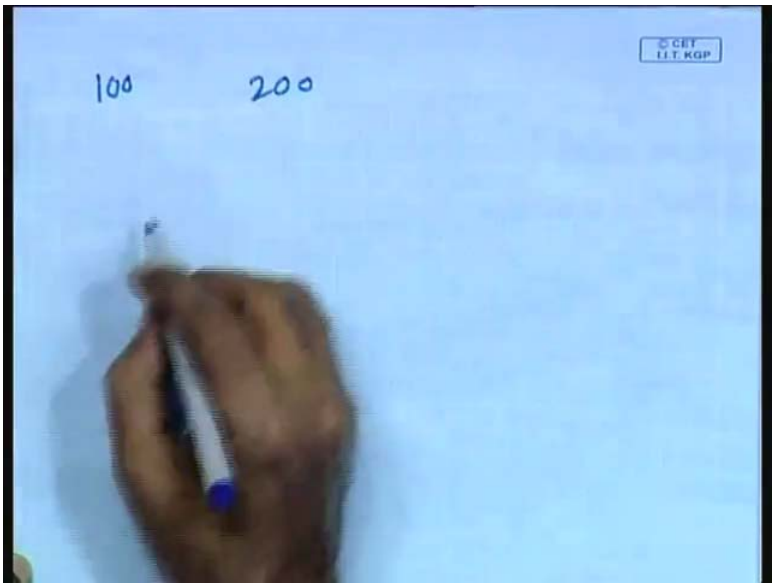
Since, it is a sharing a bus all the protocols are bus based, so by monitoring the bus the bit any transaction that has taken place, can be identified a processor can recognize that and appropriate action is taken. On the other hand we shall see the, in this directory base protocol it is somewhat different, and this is an alternative to this snoop based coherence protocol.

So, in any case we have to maintain that the cache coherence and with using this protocol we shall be able to do that, and basic motivation behind this directory base protocol is that it can be used to reduce the bandwidth demands in a centralize share memory machine. So, a directory keeps the state of every block that may be cached; that means, I am we are telling about directory based, what do you really mean by directory is a kind of database, which that is there which keeps the state of every block, that may be cached, that may be transferred from the main memory to the cache.

And information like which cache have copies of the block, whether a particular cache is dirty and so on or whether it is shared by more than one processor. So, all this information have to be stored in the directory for proper management of the, and to offer at cache coherence, so the amount of information is to proportional to the product of the number of memory blocks, and the number of processors. So, here we have seen that memory unit will have large number of blocks, and for each block we have to maintain some information.

And similarly, for each block corresponding to each processor we have to maintain some database, that is why the amount of information will be proportional to the product of the number of blocks, in to the number of processors in the system. And to be went the directory to become a bottle neck, the directory is distributed along the memory. Now, as the size of the memory increases or in other words number of blocks increases, and as the number of processors increases particularly in a distributed memory architecture the number of processors can be hundreds.

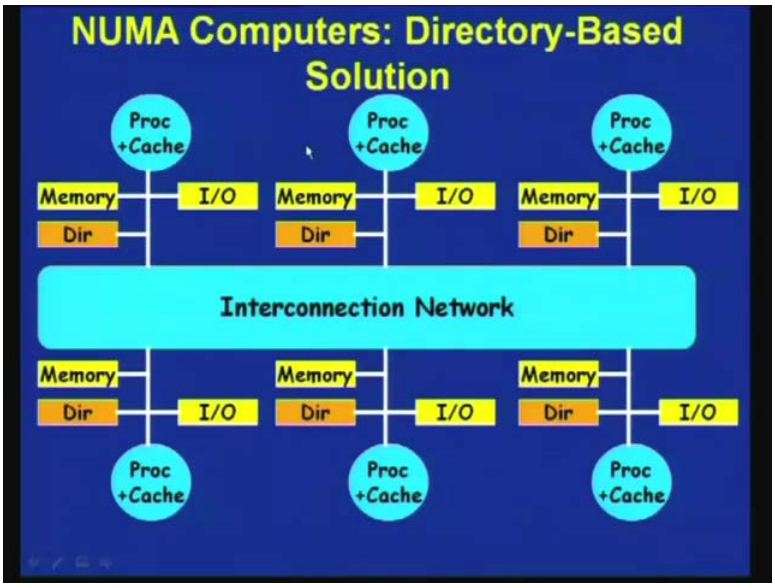
(Refer Slide Time: 07:05)



It may be 100, 200 and more, so it will be large number of processors, contrary to symmetric memory multi processors square, you can have at most few dozens of processors. Here, it will be few hundreds of processors and as a consequence the size of the directory is quite big pretty week, so if you where do you store that directory. So, what can be done just like we have distributed the memory, we can also distribute the directory, and that is what is being done different directory.

If to prevent directory to become a bottle neck the directory is distributed along with the memory, and different it has another advantage different directory access to go to different, it has goes to different directories. That means, the since it is ha divine distributed, so reference to a particular directory, will go to a particular place and when there is another reference it will go to another place, so it will help you reduce the traffic on a particular bus.

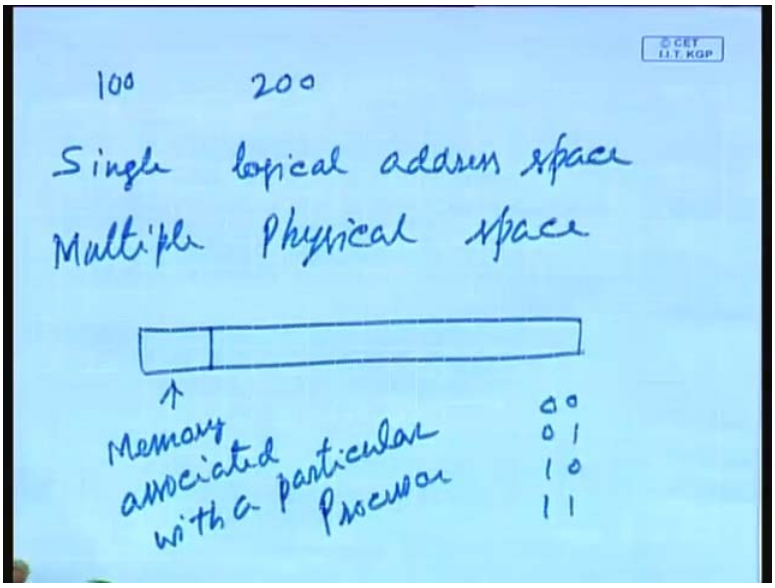
(Refer Slide Time: 08:28)



So, shall this see this in detail as we discuss the protocol, so this is the basic model you can say that NUMA computer, Non Uniform Memory Access computer, and where you can use directory base solution. So, we can see each processor along with the cache memory, and IO each of them is a autonomous system, by autonomous I mean each of them can function independently, each of them may be having it is own operating system that is one advantage in this.

So, and then we have a directory, so we have a directory and then that directory is maintaining information about the different things that I have mention, and each of them is connected through a inter connection network. So, through the inter connection network, they are connected, and then this memory is being shared and by that I mean there is a single logical at the space.

(Refer Slide Time: 09:31)



And of course, there is multiple physical space, so as you have seen here the memory there are this is memory is been available in provided to each of the processors. So, what can be done say for example, this is the address, the higher order bits can be use to signify the memory associated with a particular processor, so memory associated with a particular processor.

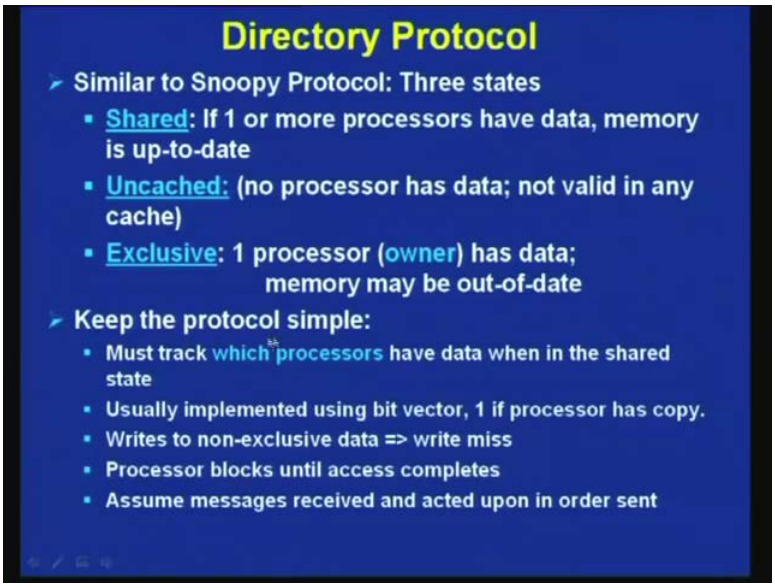
So, if you have got 4 processors two bits will be sufficient 0 0 0 1 1 0 1 1 or if you have got more than in case of this NUMA model we have seen the number can be 100. So, the number of bits that will be required will be dependent on the number of processors, and in the remaining bits can be used to reference the memory with in that particular memory block in a associated with a processor.

(Refer Slide Time: 11:08)

That is how the memory is distributed in this particular case, now in NUMA computers the messages have long latency, because we have it has been information has to go through the inter connection network. And broadcast is inefficient because all messages have explicit responses, so and main memory controller, you has to keep track of it which processors are having cached copies of which memory locations and so on.

So, and on a right only need to inform the users not everyone; that means, where the writing is taking place, and who are the users of that particular block that they are to be inform, but not everyone. And similarly, whenever on a dirty read, whenever a read is occurring and that is already modified by processor is that information has to be forwarded to the owner; that means, owner means quiz reading it.

(Refer Slide Time: 12:17)



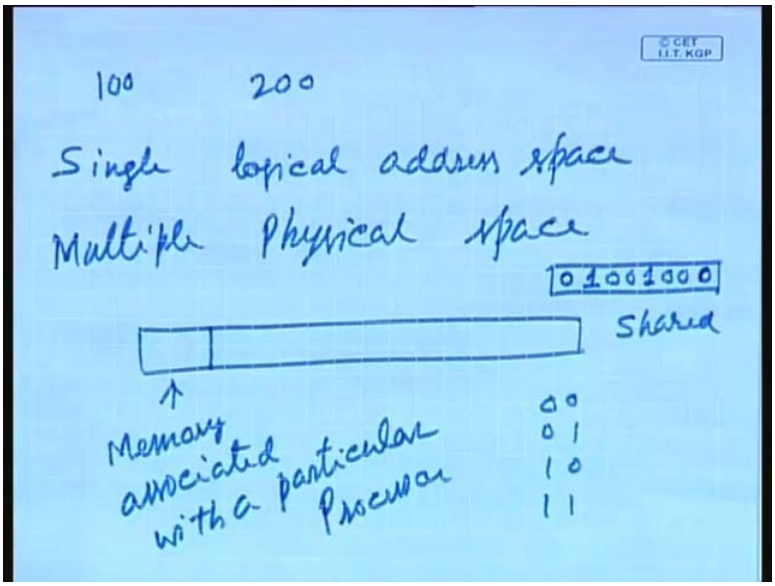
So, I shall going to the details of that particularly, this protocol that will be use this directory this protocol is very similar to snoopy protocol, just like we have seen there are 3 states with the help of each, we can explain the protocol. The way the reading, writing and other things are access taking place, in this case the three states are shared, uncached and exclusive.

So, whenever a particular cache block will be in the shared state, if one or more processors have data and memory is up to date; that means that a more than one processor is has have copied it and memory is also up to date. That means, the same copy not only its present in a memory, but also in the cache of more than one processor, then we shall say it is shared and also it is updated; that means, all the copies are identical.

Then un cached no processor has data not valid in cache; that means, in case of un cached the data is only present in the main memory, and not no processor has cached it. And as a consequence I mean whenever a processor has to read it, it has to read from the main memory, then the third state is exclusive where one processor has the data. And in such a case what happens the memory from the main memory, it is transferred to the cache of a particular processor and whenever, it is transferred to a particular processor and whenever, it modifies it the that processor becomes the exclusive user of that particular block of data.

So, exclusive state corresponds to one processor is the owner of data and memory may be out of date, out of date means in a main memory the modification that has taken place in the cache memory has not yet been return back. So, to keep the protocol simple, but it much track each processors have data, when in the shared state usually implemented using bit vector.

(Refer Slide Time: 14:55)



That means, suppose you have got 8th processors corresponding to 8th processors, you can have 8 bits, so if a particular say this corresponds to 0th processor this correspond to 8th processor. So, whenever a particular processor copies the data that bit is modified to one, so in this way if there is one present in more than one, bit that signifies that the same copy of data is processed by more than one processor.

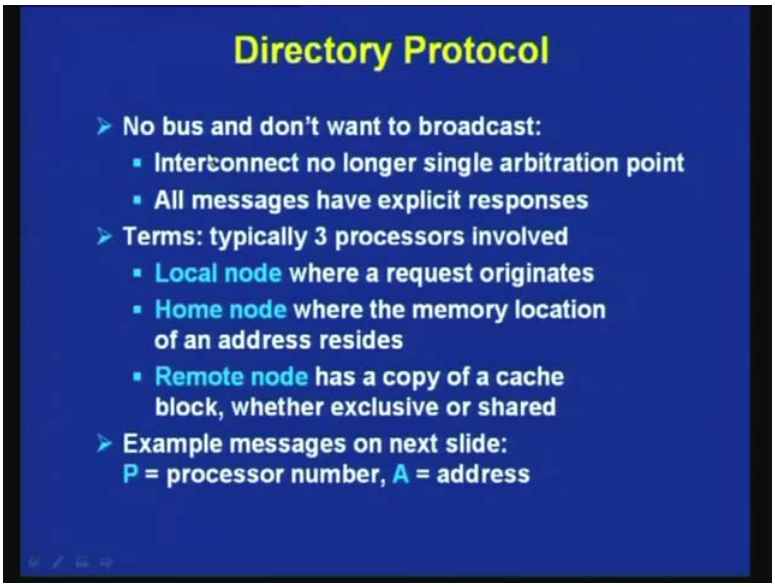
And as a consequence it will be in the shared state, and this bit vector signifies which processor will be having the copy, so writes to non exclusive data; that means, whenever a particular processor tries to write a data, which is non exclusive state. Then write means will occur and it will change the state from exclusive to I mean a write miss will occur, then it will may be become exclusive, because after the modification is done, then state will change to exclusive.

So, from non exclusive to it will change to exclusive, non exclusive means it can be shared or in un cached state, and if it tries to write will be miss the write miss will be generated on the bus. And then this state will change to exclusive and processor blocks

until access conflicts, so here some kind of you know it is not a shared bus. So, some kind of serialization is necessary, because one operation has to be completed, so that is why the processor blocks until access completes; that means, allow other processors to access it.

And assume messages received and acted up on in order sent so; that means, when a particular message is sent, then that appropriate action for that message is performed only then other messages will be exit. And appropriate action will be taken, so this is the kind of serialization is necessary and this is important because bus is not shared and latency is more.

(Refer Slide Time: 17:23)



So, no bus and do not want to broadcast so; obviously, since there is no shared bus there is no possibility of broadcast inter connect no longer single arbitration point, because since it is not a bus or bus arbitration cannot be done with the help of this inter connect network. It can be a point to point network or it can be a inter connection network having multiple paths, so this single arbitration point does not exist and all messages have a explicit process; that means, in case of bus base system.

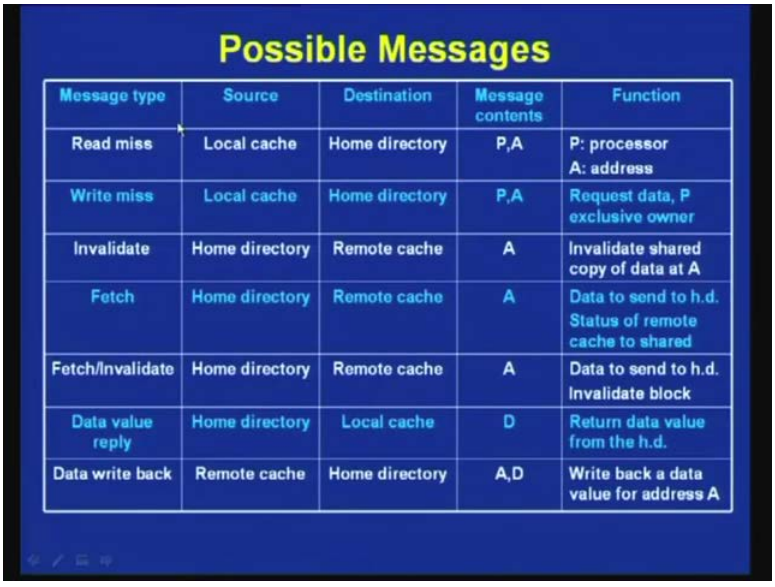
So, it is a kind of broadcast on the bus and as a consequence all the processors and will be will listens it that, but in this particular case that is not sure it has to be explicit I mean for all message each and every message will have explicit responses. Now, in this

directory base protocol will find that they are exists three different types of nodes, one is local node where a request originates.

So, local node will generate an request for read write, on the other hand home node where the memory location and the address resides; that means, a home node is where the memory location and address is reside and from, where data has to be read. And another node is there, that is remote node has a copy of the cache block where whether exclusive or shared.

So; that means, remote node is the place, where the copy in the copy of the cache is already present, and it may be either in the exclusive state or in the shared state. And you can have a number of processors, so P specifies the processor number, and whenever the transaction will takes place the address of the memory has to be specified, and the whenever, we will see then in the messages the P and A processor number and address will be mentioned for different purposes.

(Refer Slide Time: 19:50)



And this is the least of possible messages, so for example, read miss, read, miss means that data is not present in the cache memory, and the processor is trying to read and source can be I mean the source it is generated by the local memory cache. And destination is home directory and message will content the processor number and address. Then write miss again the source is source of the message is local cache, and

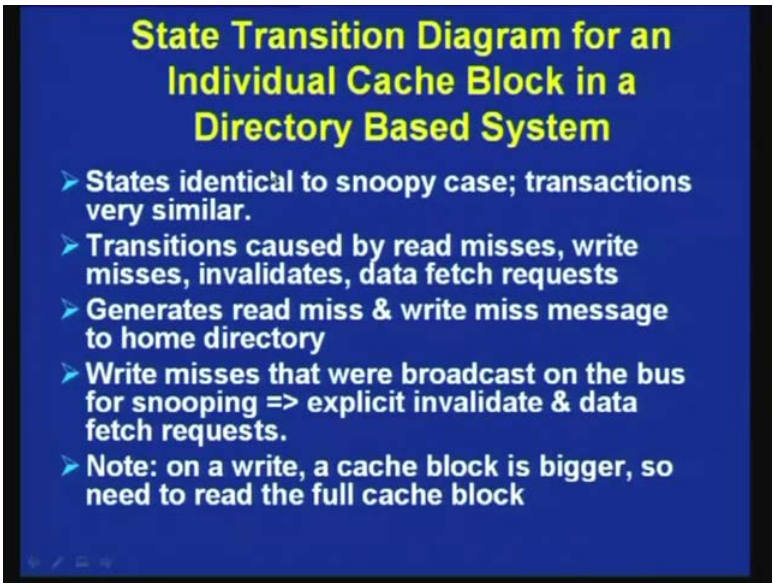
destination is home directory means we have seen the diagram, so this is the directory and is a processor.

So, this is the home directory of this processor and other directories are remote directories of bit processor, then write miss the it is a source is again local cache generated by in invalidate it is generated by the home directory and destination is remote cache. So, whenever data is modified and it was having a copy then it has to be invalidated, and the message content is address invalidate at the shared copy of data at A. So, in that address that data has to be invalidated, then fetch again the source is home directory, because data is not present in the memory.

So, it has to be fetched and from the remote cache it has to come and this will be generated by a data is sent to home directory, and status to remote cache will be modified to shared. And then again fetch or invalidate again the source is home directory, and destination will be remote cache and message contents will be address and data will be sent to home directory.

And it will invalid action will be invalidate the block, then data value copy in the in the local cache, so in this case the source is the home directory and destination is local cache and data will be message, will content the data which will be copy, which will be written in to the local cache, so return data value from the home directory. And data write back again the source is remote cache in this particular case and destination is the home directory, and message will content the address and data and the function will be write back a data value for address A. So, these are the possible messages that will be generated, in this directory base protocol, and let us see and what will be the states, what are the different states that will mention.

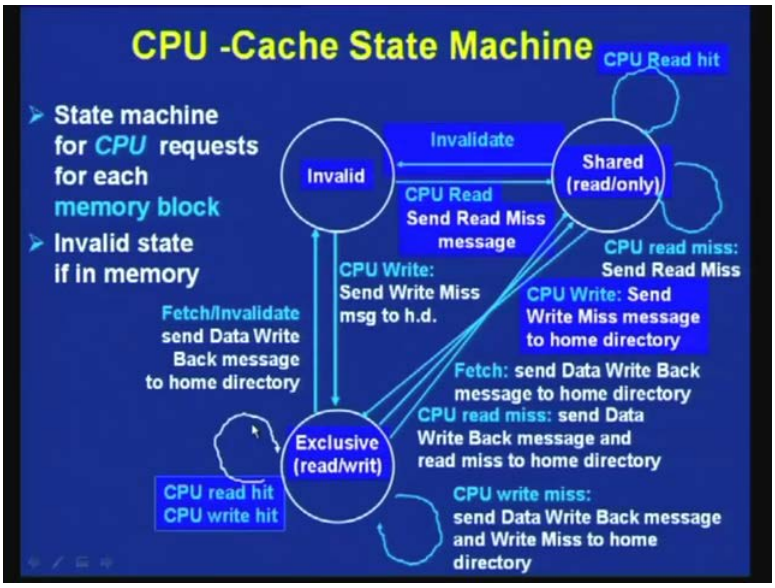
(Refer Slide Time: 22:50)



So, states identical to snoopy case transactions are very similar, we have already discussed about the snoopy base case protocol, we shall see the states are identical and transactions are similar. However, they are little different transactions cause caused by read message write message invalidates data request, so transitions will occur from one state to another, because of this message.

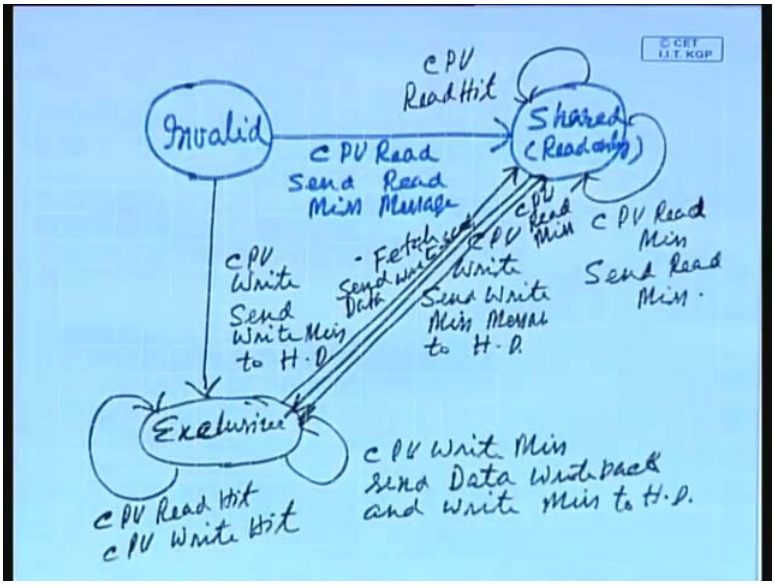
And it will generate read miss and write miss message to home directory, and write message that where broadcast on the bus for snooping. So, in this case that will not happened explicit invalidate and data fetch request are to be used in case of directory base protocol note that on a write a cache block is bigger, so need to be read the full cache block. So, that means, whenever you are performing a write it will although you will be writing in to on a word, but a block consists of several words, so entire block has to be transferred all the words and you have read it then modification can be done.

(Refer Slide Time: 24:13)



So, this is the state transition diagram, state machine for CPU request for each memory block, so this is the state machine for CPU request for each memory block and as you mean that initially invalid state it is in the memory. So, if it has gradually build up and see how it exactly happens.

(Refer Slide Time: 24:47)



So, initially we have got it is in the invalid state, invalid state means that data is not present in the cache, and it is also not in shared state, so CPU can I mean the because the CPU can generate a generate a message that is known as CPU read. So, whenever CPU

read is done generated, it will read to it will send read means message to the directory; that means, that in response to this the reading of data will take place, and obviously, it will change state from invalid to shared and in this case it is read only.

Now, whenever it is in the shared state, in a let us first complete invalid state, from the invalid state it can also perform CPU write. So, whenever tries to CPU write, then it will send write miss to home directory, and the state of the cache memory will change to exclusive because after it has perform the writing, so it will become exclusive. So, from the invalid state there are two transactions, I mean two transactions one because of CPU read, another because of CPU write, and whenever CPU read occurs it goes to shared state, whenever CPU write occurs it goes to exclusive state.

Now, when it is in the shared state, then if you perform continuous reading; that means, CPU read or write whatever it occurs CPU read hit. So, CPU read hit means it will remain in the same state there will be no change after the data is taken to the cache memory, it can continue to read without changing the status. And similarly, whenever it perform CPU read miss, then it will send read miss and it will remain in the same state, because it will read it, and then it will perform it will remain it will continue to I mean se send read miss and it will remain in this particular state.

Now, from this state, it will go to I think CPU read means and whenever it perform CPU write what happens, then it does CPU write, so when CPU write occurs again it goes to exclusive state. So, it will send write miss a message to the home directory and it will change to exclusive state, so from this shared state these are the possible transactions CPU read, CPU read hit, CPU read miss and CPU write. Now, whenever it is in the exclusive state; that means, the cache is present I mean only in that particular processor and it is in the exclusive state, so in this case it can perform CPU read CPU read hit.

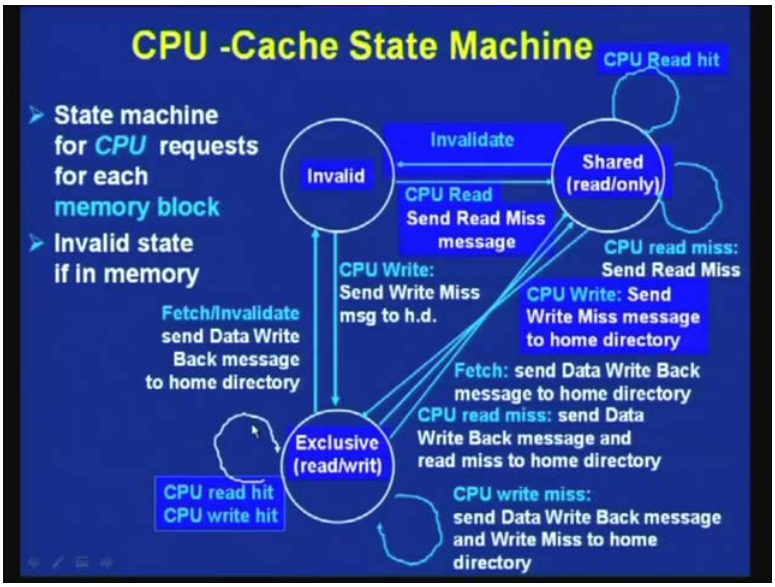
So, it will remain in the same state it will continue to read or CPU write hit it will be no change of state, because it is already in the exclusive state, this CPU is free to read the data or write in to the data, when it is in the exclusive state. Now, what can happen the CPU write miss occurs and it is in the exclusive state then; obviously, it will send data write back and write miss to home directory, and it will remain in this state.

So, CPU write miss that data is not here, but it was in the exclusive state, so it will remain in that exclusive state; however, data write back has to occur. And it will send the

write miss to home directory and accordingly the data will be transferred, and to the at cache and a modification will be done by the processor and it will remain in the this state, so it will it will not change.

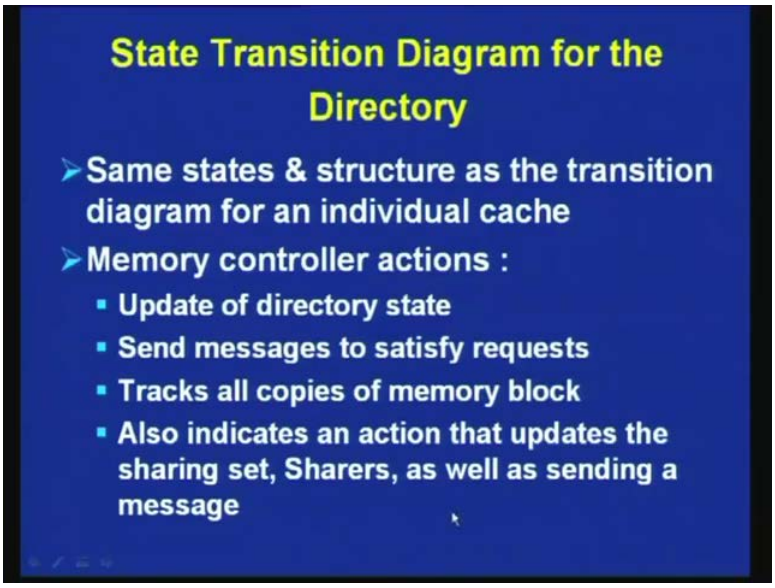
now, comes to two more states from exclusive to, it will go to say it performs CPU write, CPU write is there CPU, so it say in case of fetch, it will send data write back to the home directory, so and then it will go to shared state. And last is last one is CPU read miss, so whenever CPU read miss occurs, then it is not in the, it will go from exclusive state to shared state, and it will generate the write back message to the read miss to home directory, and will switch from exclusive to the shared state.

(Refer Slide Time: 31:48)



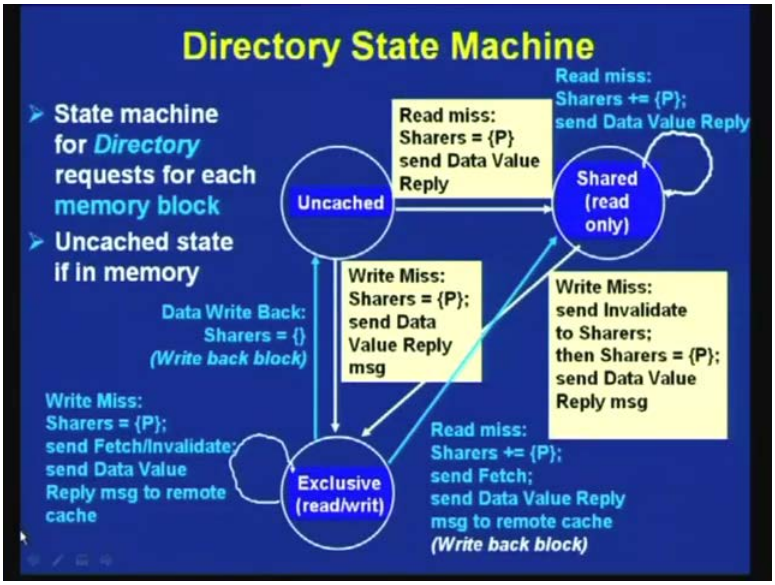
So, these are the different transactions that you will occur, which is shown in this particular diagram and this is the situation, where state machine from CPU request for each memory block that is occurring. And whenever it is in different states, and how this state machine occurs that I have discussed one after the other.

(Refer Slide Time: 32:09)



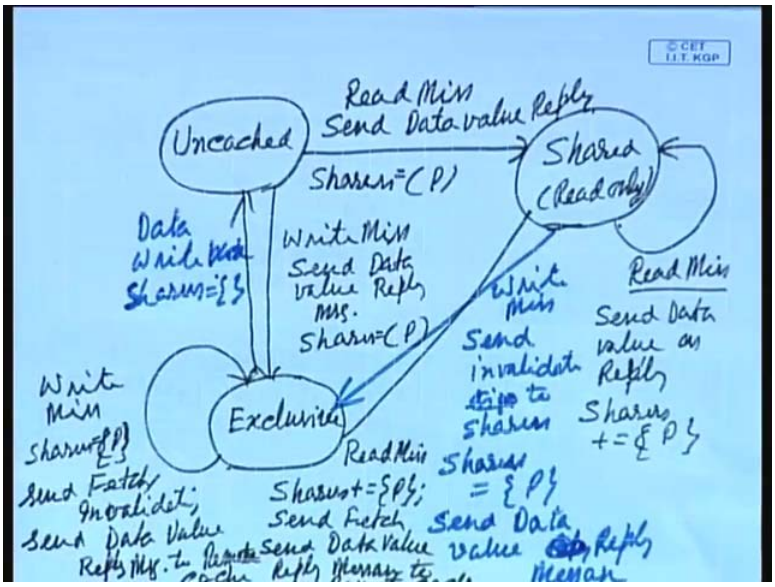
Now, let us see, so I have discussed the same state and structure as the now consider the diagram of the directory; that means, directory also will also maintain a finite state machine. And what the this state's of that and the transitions for individual cache will be same in case of the directory also, and the memory controller will perform the following actions update of the directory state. Send messages to satisfy request track all copies of memory block and all indicates an action also indicates, an action that updates the sharing state sharers as well as sending a message.

(Refer Slide Time: 32:53)



So, let us see the various situations, now in this case you can see the directory can receive three messages, what are the three messages, number one message is read miss, write miss or data write back. So, these are the three messages that will be received by the by the directory the state machine, and directory state machine again will remain in three different states.

(Refer Slide Time: 33:38)



Let us see the three different states are one is un cached, data is not yet un cached or it can be in shared read only and third state is exclusive, so these are the three states. And now whenever it is in the un cached state; that means, a particular block has not yet been transferred to the cache memory and there is a read miss. A read miss occurs and whenever a read miss occurs it will send a message a send a data value reply, so data is sent and it switches to share as shared state.

So, in such a case the you have to maintained a least where sharers will include the this processors, which has to which it has been send that processor name has to be will go to the will be maintained in a sharers list. So, these sharers will be equal to P and it will go to the shared state, and when it is in the shared state from the un cache state it can also go to the exclusive state by write miss.

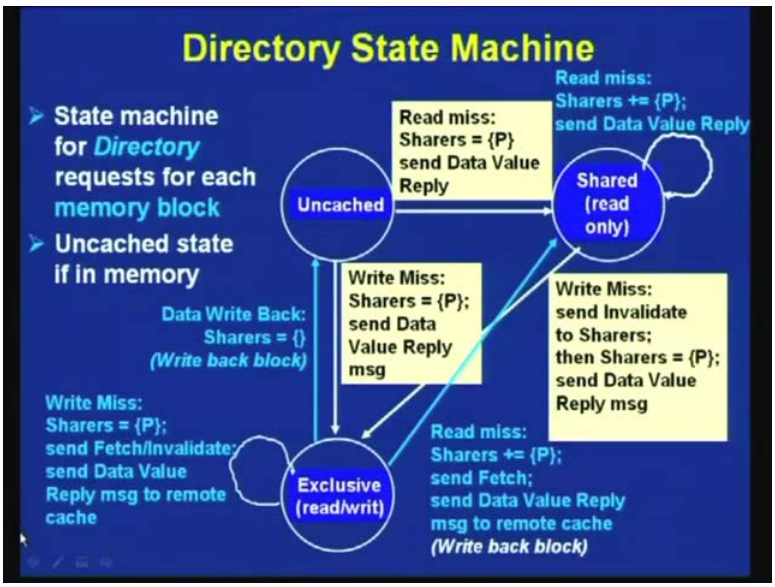
And in this case also it will send data value reply; that means, the directory will send the data value reply message and sharers will be again P that processors will be entered in the sharers list. So, these are the two transitions that will occur whenever it is in the un

cache state, then let us see when it is in the shared state the directory can receive a message. Like read miss, when a read miss occurs then it will send data value reply and also the sharers list will be updated, it will plus it will update P, so this is what will happen in case of a read miss.

Now, let us consider when there is write miss, then it will send invalidate signal in read a message to sharers to all the processors, it has to go and then sharers list will be having only P, because it is in the exclusive controller p that is why it will have the only value, and it will send the invalidate message. And it will send and data value reply message, then we have got another I mean, whenever it is in the exclusive state it can go from exclusive to shared state, whenever there is a read miss and in that case that sharer list will had the P, which is trying to read.

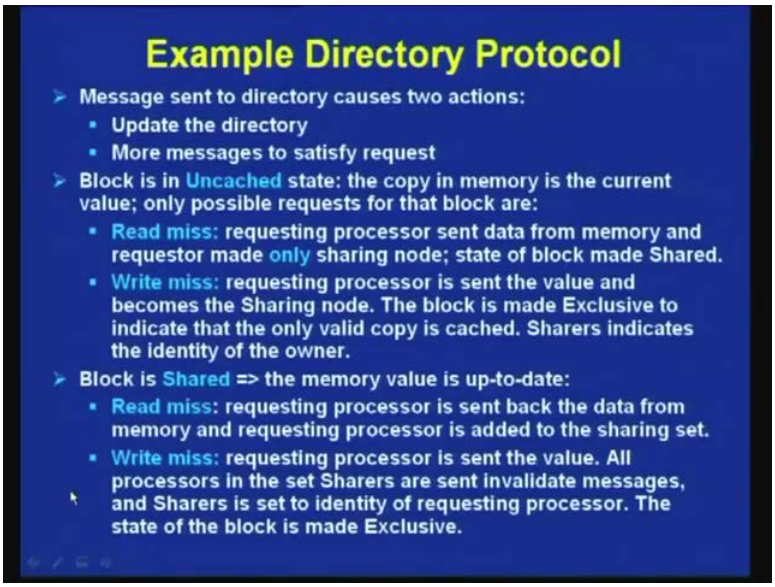
So, this is the list of processors and it will send fetch the data and it will the directory will send data value reply message to remote cache, so this is the case whenever it you are using write back policy. And the last one is in this case write miss, when write miss occurs it remains in the exclusive state, so the sharers list will be having P and it will send fetch or invalidate and also send data value reply message to remote cache. So, these are the various transactions, so and finally there is another one, when it is in the exclusive state, when the there is a data write back, so in this case also this share has sharer list will be empty.

(Refer Slide Time: 40:16)



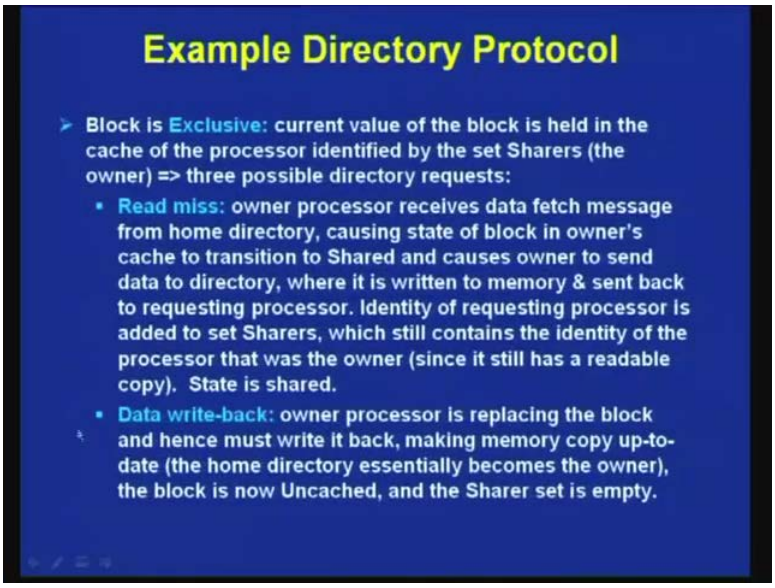
And it will perform write back block will be transferred it will be un cache, so this is these are the areas transitions that will occur in case of the directory state machine, in response two different require messages like read miss write miss and data write back coming from the processors.

(Refer Slide Time: 40:32)



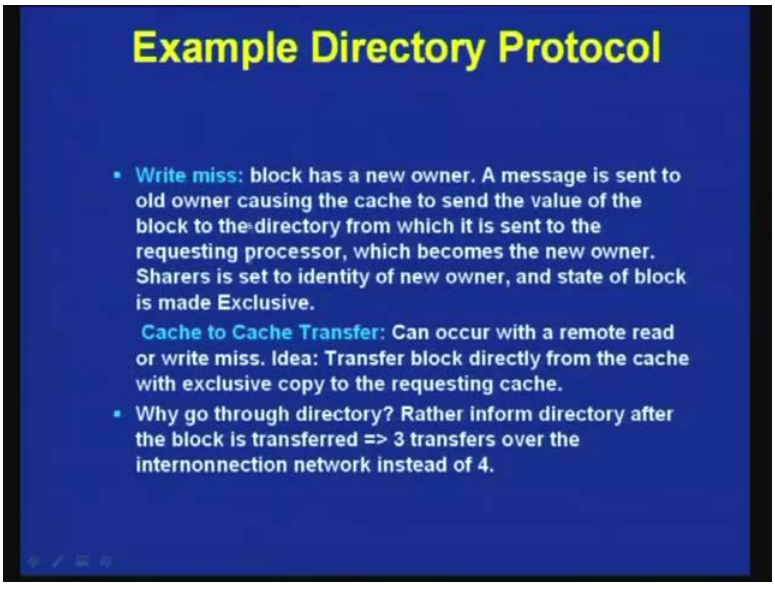
Now let us consider an example, messages sent to directory causes two actions update the directory, so we have already discussed about these things when read miss occurs, write miss occurs, when it is in the shared state block is in the shared state we have discussed in detail.

(Refer Slide Time: 40:55)



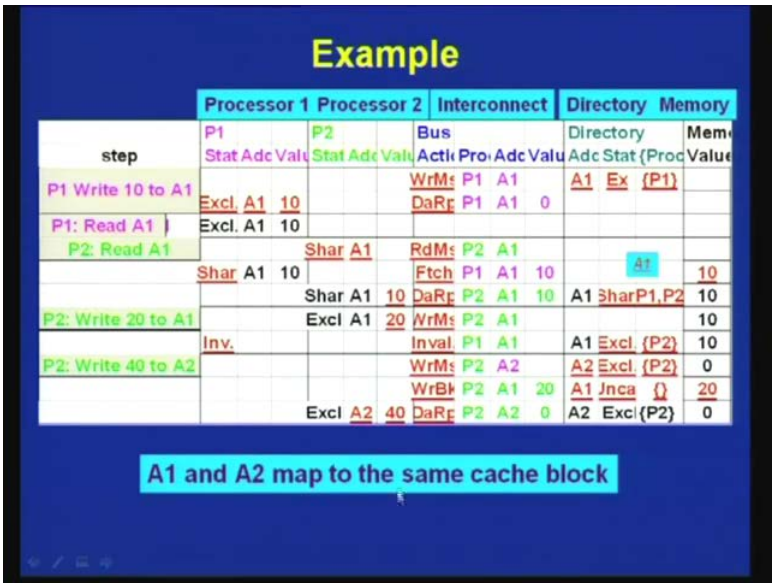
When it is in exclusive state when read miss occurs, data write back occurs we have discussed those things.

(Refer Slide Time: 41:04)



And in response to write miss what will be done, these are again we have discussed in detail.

(Refer Slide Time: 41:09)



Now, let us consider an example, where different actions are taken in response to that what is done by different processors, what is done by the different inter connect, that is your inter connect bus, what is done in the directory, and what is done in the memory. So, different things that happens is mentioned here, so P 1 write tend to address A 1, so a processor P 1 is writing some data in at the address corresponding addresses is A 1.

And so for as the processor P 1 is concerned the state is exclusive, because it is written data and in that address A 1 the value 10 will be depend, and whenever this is being done. Then the corresponding bus sections will be, since it is writing write miss will be on the technique pass and the message that has been sent is by P 1 and at the corresponding address A 1, and this the then the corresponding data read that occurs, data write back occurs is a corresponding to P 1.

And A 1 and the directory protocol and the directory will have the message corresponding a state will be address A 1, the state will be exclusive and processor corresponding is P 1. So, this is the sequence of P 1 that will take place is response to processor P 1 writing ten in to A 1. Now, whenever the processor same processor reads the data of A 1 then what happens, so in this particular case as you can see there is no change in state, because it was already in the exclusive state it will simply read it.

As, you have already seen it there is no change of state, and for the processor will generate the address and it will read it from the cache memory. Then P 2 whenever

another processor tries to read from the same address, then what will happened let us see, so in this case it was earlier in the exclusive state of processor P 1. Now, it will switch from exclusive to shared state, because the processor is reading it and it will be also present in the cache memory of processor P 2.

So, this state will change to shared correspondingly the processor P 1 also will change the state from exclusive to shared, and the value that will be present here is 10. Same thing for P 1 and P 2 address is same and on the inter connect the read the read message, that is being sent is P 2 corresponding to P 2 address is A 1. And it will be fetched from the processor P 1 and address A 1, and the value is ten and the data reply that occurs is P 2 A 1 and 10. So, these are the by inter connect or bus that will be at the directory will response in this way.

And the directory will be responding with A 1, and it will be in the shared state and P 1 and P 2 are having copy, it has to maintain the list of sharers, so list of sharers are here P 1 and P 2 and value is ten now P 2 is writing in the same address some other value. So, in this case what will happened the P 2 will write, and in this case it will become the exclusive under the exclusive control of P 2, and the value is changed. And they will be corresponding changes on the I mean messages on the bus inter connect, and on the directory will also will reach corresponding will change the state A 1.

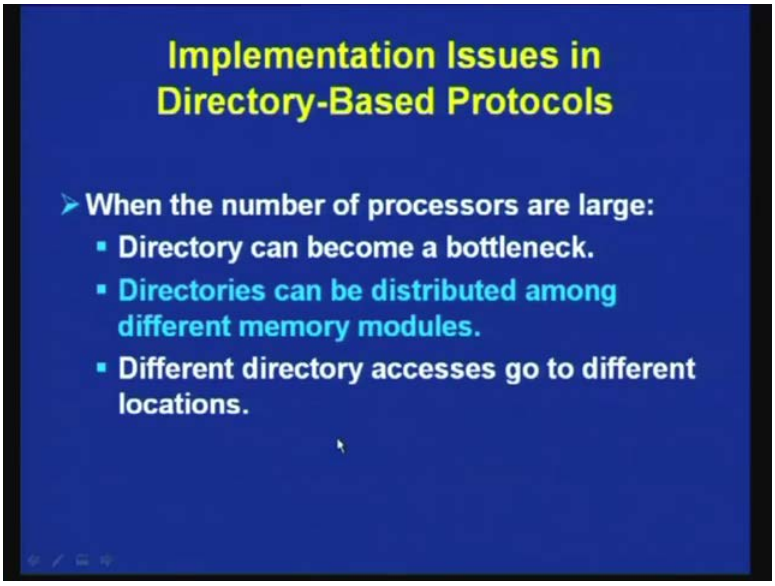
Earlier, it was A 1 address was A 1 it was in the shared state, and P 1 and P 2 where the list of sharers, but in this case state is a exclusive and only it is present in P 2. And value and memory of value remains the same 10, because it is not yet been a return in to the memory, this only present in the cache memory of P 2. And it will be invalidated in processor P 1, so processor P 1 will have this state of invalidate, and for P 2 it is exclusive and so on.

Now, P 2 writes 42 8, so another writing is taking place, where P 2 let us see what will happened in this case, so it will invalidate that inter connect will generated in validate P 1 A 1. Then write message P 2 A 2 and write back P 2 A 1 20, and address is and in this case the exclusive it will go to the exclusive state offers are P 2, because it is written it was already in the exclusive state, but the model value has been modified.

So, it will go to the a remain in the exclusive state, either the address and data values are changing, and in the directory also the same thing is happening A 2, it will remain in the

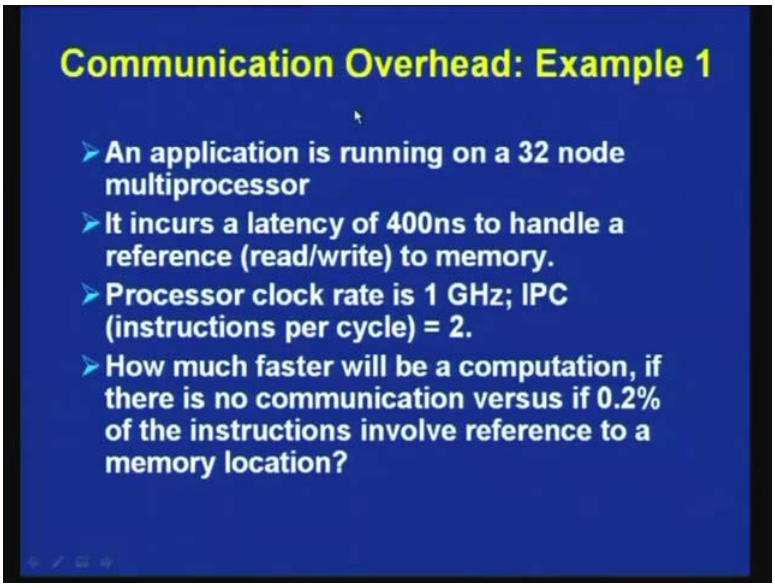
exclusive state and processor which is will be in the sharer list is P 2 and data value is 0. So, this is how it has been assume that A 1 and A 2 map to the same cache block, if it is in the different cache block then of course, they will be a cache miss. That has not happened here, because they belong to the we have assume that 10 20 and 40 all belong to is A 1 and A 2. They belong to the same cache block and as a consequence there was no cache miss, and only the state of the a processor changing from exclusive to sharing share shared or and like that.

(Refer Slide Time: 47:55)



Now, we shall discuss some of the implementation is choose in directory base protocols, so when the number of processors are lodge, directory becomes a bottle neck. As, I have told, because the size of the directory is proportional to the number of blocks in to the number of processors. So, the directory is can be distributed among different memory modules, so earlier we have seen it was in the associated with the same memory, but now what can be done if you can divided into separate memory massive modules. And different directory axis go to different location, because since they are in different modules, then different directory axis will go to different locations leading to lesser traffic on the on the bus corresponding to the memory.

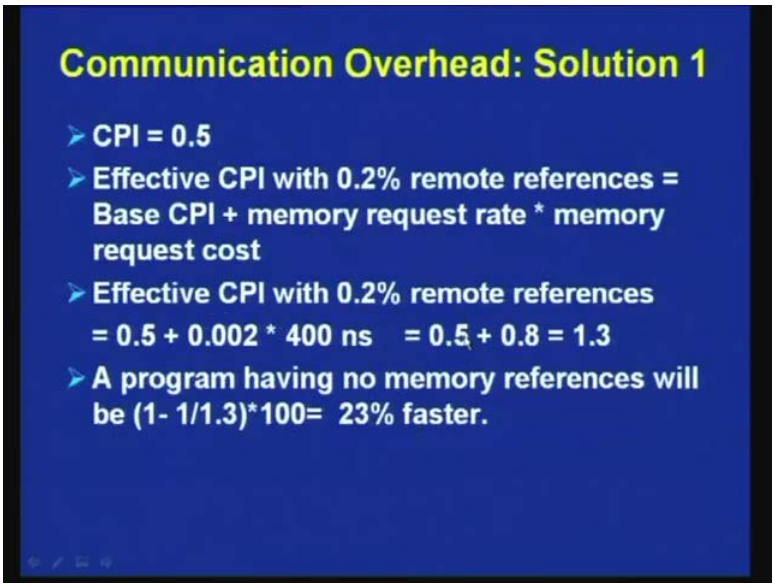
(Refer Slide Time: 48:51)



Here, at two examples, illustrating what kind of communication overhead that occurs, so let us assume that an application is running at 32 node multi processor, and it incurs a latency of 400 nano seconds to handle a reference read write to memory. That means, the 40 nano second is necessary to read from the memory, and processor clock rate is one giga hertz and instruction per cycle is 2; that means, 2 nano seconds is needed.

Whenever it is in the cache, but whenever you have to read it from the main memory, then the time needed is 400 nano seconds, so it is much longer. Now, how much faster will be the computation, if there is no communication, communication means there is no communication with the memory versus the point two percent of instructions involve reference to memory locations.

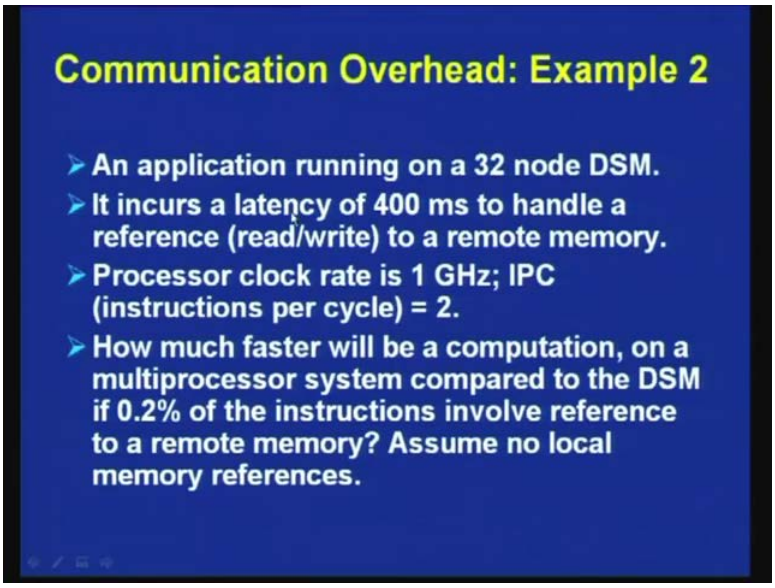
(Refer Slide Time: 50:07)



So, you have to compare the computation time, for the two different situation and compare it, so in this case CPI is 0.2, because you know instruction per cycle is 0.2, so CPI is 2. So, effective CPI with 0.2 percent remote references, so whenever you are performing reading from the memory there will be remote references. So, the CPI when change to effective CPI will become best CPI plus memory request rate in to memory request fast, and you have already seen that effective CPI with 0.2 percent reference will be 0.5 is 0.5 plus 0.002, because it was 2 percent reference and 400 nano second is the axis time of the main memory.

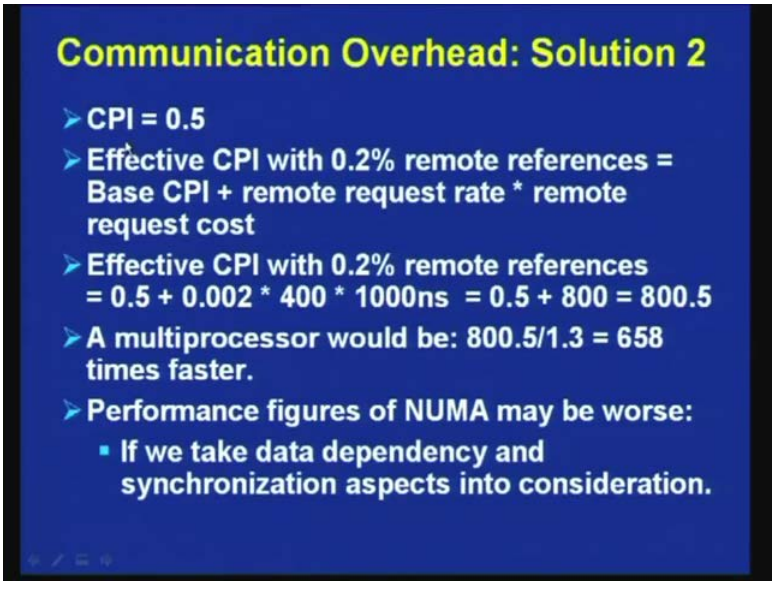
So, this will give you 0.5 in to this will be 400 into 0.002 it will come 0.8, so 1.3, so a program having no memory references will be 1 minus 1 by 1.3 into 100 percent; that means, 23 percent faster. So, whenever A 2.2 percent memory access is taking place it is becoming pretty slow, as you can see it is becoming 23 percent, first I mean slower you can share or whenever there is no memory reference it is 23 percent faster. So, the CPI is changing from 1 to 1.3 from 0.5 to 1.3, so this is the situation.

(Refer Slide Time: 51:54)



Now, consider another example, again it is an application is running on a 32 node distributed share memory, in this case it is a distributed shared memory situation. It incurs a latency of 400 nano second to handled a reference to a remote memory, and processor clock rate is 1 giga hertz IPC is 2. Now, how much faster will be the computation on a multi processor system compare to the distributed shared memory of 0.2 percent of the instructions involve reference to a remote memory, as you local memory references only.

(Refer Slide Time: 52:34)



So, the in this case effective CPI with 0.2 percent remote memory remote reference is equal to basic CPI plus remote request rate in to remote request fast, so that 100 nano second, 1000 nano second is not mentioned here. So, in this case effective CPI will become equal to 0.5 plus 0.0002 into 400 and into 400 or 1000 nano second, because this is the remote request fast. So, remote request fast in substantially more compare to when it a whenever it is read from the local memory, as I was in the situation in the previous case, so it will become 0.5 plus 800 is equal to 800.8.

And so multiprocessor would be having 800.5 by 1.3 is equal to 658 times faster, so in the previous case we have seen the CPI was 1.3, whenever it is read from the local memory a, but in this case it is from the remote memory leading to CPI of 800.5. So, this is the previous one is much faster, and performance figures of NUMA may be worse, if we take data dependency and synchronization aspects into consideration, without taking this data dependency on synchronization aspects into one consideration this is a situation.

And if you take them in to account it will be more, so with this we have come to the end of today's lecture, so it in today's lecture we have discussed about the distributed shared memory architecture, and we have discussed the directory base protocol for overcoming the cache coherence problem. That arises whenever a shared memory is used in a distributed memory distributed memory environment.

Thank you.