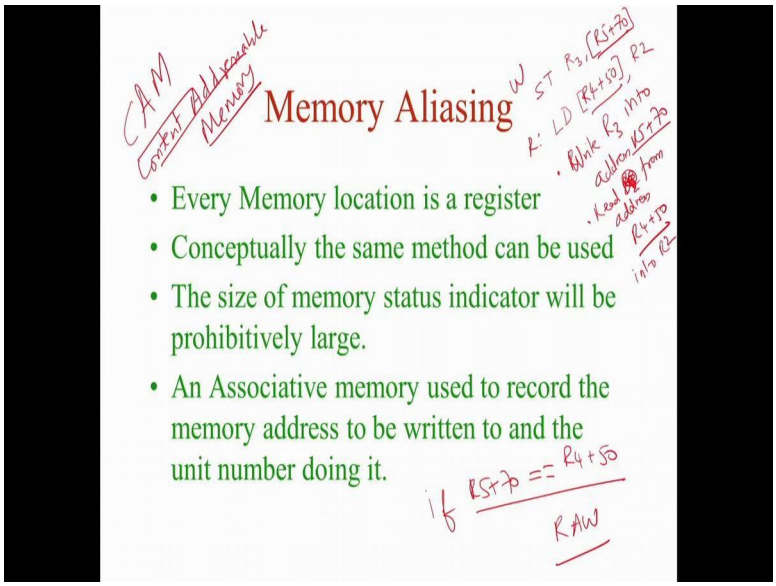
**Computer Organization and Architecture**
**Prof. V. Kamakoti**
**Department of Computer Science and Engineering**
**Indian Institute of Technology, Madras**

**Lecture – 20**
**Dynamic Instruction Scheduling (Contd..) Part- II**

Fine, now come to your question right what is memory aliasing we have already seen this memory aliasing, right.

(Refer Slide Time: 00:23)



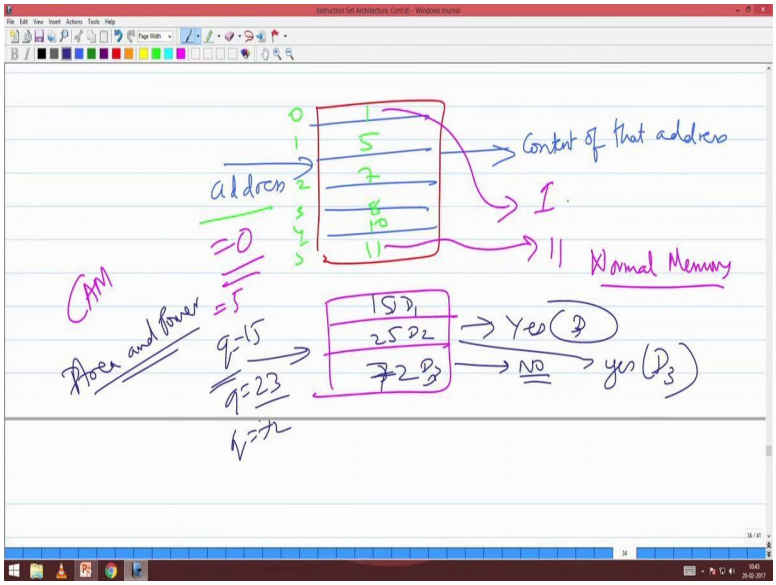

Now I have load R 4 plus 50 comma R 2 store R 3 2 R 5 plus 70; that means, this is write R 3 into address R 5 plus 70 read R 2 from address R 4 plus 50 right now if R 5 plus 70 is equal to R 4 plus 50 then we have a raw hazard because this is reading from R 4 plus 50 read from address R 4 plus 50 into R 2 this is reading from address R 4 plus 50 while this is writing into address R 5 plus 70. So, this becomes this is a read on the memory this is a write on the memory. So, this becomes read after write.

This becomes read after write. So, how do we handle this for every if I treat every memory location if I treat every memory location as a register then this is done right because when I am doing the load I know that this is I know the address. So, if I will go

and tell for this address for every address if I have a location then what will happen for this address latest value is not there in the address, but is currently computed by an execution unit. So, whatever I have done for register I can do it for this if for every address if I am going to have indicator like register status indicator I will have memory address status indicator. So, if I am writing into 5 then I will have an indicator for 5; 5 will say that this is the latest value of 5 is not in the memory 5, but it is currently computed by some unit 8. So, when I am reading from 5, I will not get from 5, but I will wait for this 8 I am happy because it will come very fast rather than going in to the memory you are getting this are able to follow yes or no right, but I cannot have if I have today 32 GB memory I have can I have 32 GB register status indicator or is it necessary right.

So, what we the way we solve is we use the concept of what we call as cam content addressable memory content addressable memory. So, I will talk about content addressable memory in the do you all follow this no I will go to content addressable memory.

(Refer Slide Time: 04:23)



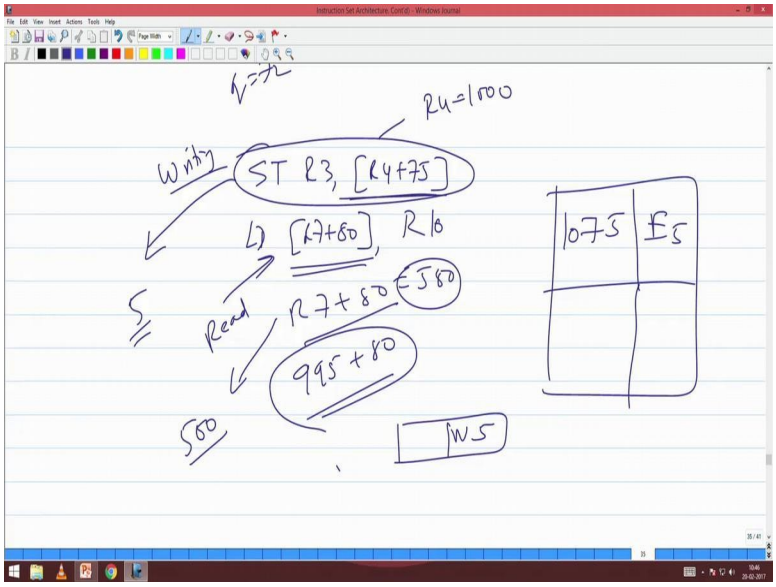

So, I will what is content addressable what is normal memory normal memory I given

address right and I get the content of that address. So, let us say this is one 5 7 8 ten

eleven the address is 0 1 2 3 4 5. So, when I give 0 I get one because this is one when I give say 5 I get eleven. So, I give the address and get the answer. So, this is a normal memory, but a content addressable memory which you as cam say it has some 3 locations let me say I am storing some 15, 25, 72, I go and ask a query array is fifteen there then it will say yes and it will also give you some data associated with sixteen let me call it D 1 D 2 D 3 it will also give you the data.

If I go and ask query is fifteen it will say yes suppose I go and ask query is twenty 3 it will say no suppose I ask query is 70 2 it will say yes and it will also give me the data associated with it which is D 3. So, I am I am addressing the content not the location I am addressing whether this content belongs there. So, there is a difference between how I am addressing the top memory and how I normal memory and how I am addressing the content addressable memory. So, when I in the content memory basically I will send a query and if that query element belongs to anywhere in the content addressable memory this fellow will say yes and it will also sends some data associated with it if the query element is not there in the content addressable memory then it will say no. So, this is the. So, please note that the content addressable memory will consume several times more data and the more area and power and that is why I cannot have large content addressable memories right it will consume lot of area and power in comparison to the normal memory that is why I cannot realise large content addressable memories in the system now basically it is like I have a set and I am asking for a membership query for the set.

(Refer Slide Time: 07:37)




Set 2 3 4 5 is 2 a member is 4 a member right, it is a query for the set understood clear yes no, now, how will I use content addressable memory for this how do I use a content addressable memory for this I use a content addressable memory for this suppose last time we had no store R 3 to R 4 plus 75 load R 7 plus 80 into R 10 something like that. So, the moment I getting in the content addressable memory moment I am getting R 4 let R 4 be thousand in the content addressable memory I will put 1075 and suppose this is this instruction is allotted to execution unit say 5 I will put E 5 also here. So, in the content addressable memory I will store 1075 and I will put E 5 this essentially says that if somebody wants to read from 1075 that 1075 is not there in 1075 in the memory it is currently being computed the latest value is currently being computed in the execution unit E 5 are you able to follow now when this fellow wants to read he wants to read from memory he wants to read from memory this fellow is writing into memory correct.

I will calculate R 7 plus 80, now suppose this is 995 R 70 is 995 plus 80 then suppose R 7 is also say some 500. So, the answer would be 580, I will go the moment I get 580 I will go to this content addressable memory and ask hey (Refer Time: 09:24) 580 is there that fellow will say no; that means, there is no problem for me right, but our luck suppose R 7 is 995 995 plus 80 is 1, 1075 before loading I will go and ask hey it is there

this fellow will say yeah, yeah, yeah, it is here. So, immediately what you will do you

will go to the reservation station and wait for this execution unit 5 and that answer you take and load it into actually you will be happy because you need not go all way to cache and to memory and. So, many things we are going to talk about are you able to follow this. So, so your read after write hazard will get handled. So, essentially what is going to happen if I am going to have a content addressable memory like this then what will happen I can handle memory aliasing problem also like the way I am going to handle the register issue all the WAR, WAW and RAW can be handled. So, like this, correct.

And the operand forwarding also happens your remaining can also happen renaming happens with the unit number now before coming to your question, what should be the size of this content addressable memory at the most how many loads can happen it is equal to the number of units. So, there will be at most 4 load store units. So, the content address memory I will just need 4 locations right. So, what was my crib about content addressable memory in contrast to normal memory the content addressable memory?

Takes lot of area and time correct content addressable memory takes lot of area and power now if I am going to have one GIGA content addressable memory that is all the chip will be content addressable memory not c p you will spent more time on that it will it will consume one mega watt substation of nuclear power for doing this right that will be 2 costly now, but I do not need that much I just need 4 and by this I can handle the real thorn in the flesh, what is it memory aliasing problem dynamically I can do it understood clear, yes.

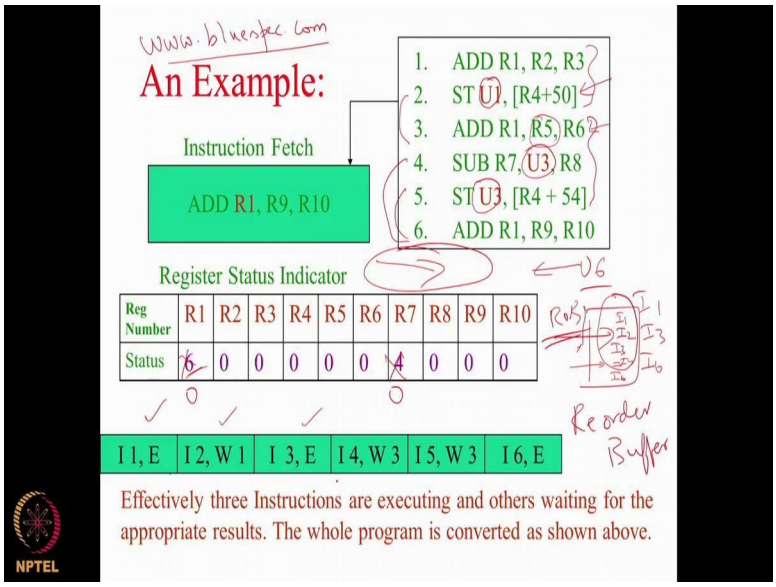
Student: (Refer Time: 11:47).

Making to a same location that is so, the last location I have to store. So, so suppose now this is used means say store R 3 R 4 7 5 again store R 6 R 8 plus 60 while R 8 turns out to be 1000, sorry, correct, there is a store R 3 R 4 plus 75 followed by store R 6 R 8 plus 60 where I have R 8 is 1015 then what you do you just mean suppose this store is currently allocated to execution unit 8. So, you make this E 8 that is all we handle no 2 fellows waiting in to the R 1 also. So, all the subsequent instructions beyond that should do this, but when you are writing back the reorder buffer will see that it is coming the latest value. So, what we have done we now have a piece of hardware which can handle

raw write and this dynamically your computer as such again I repeat your computer can be as idiot as me the hardware is as intelligent as you can do that fine.

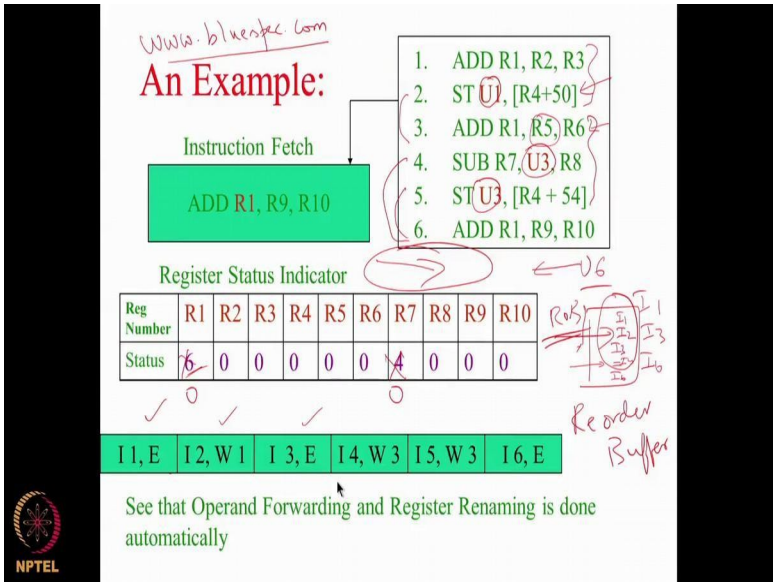
So, the point is it can be we are doing this entirely in the hardware it is done dynamically that is why we call it as dynamic instruction scheduling.

(Refer Slide Time: 13:11)



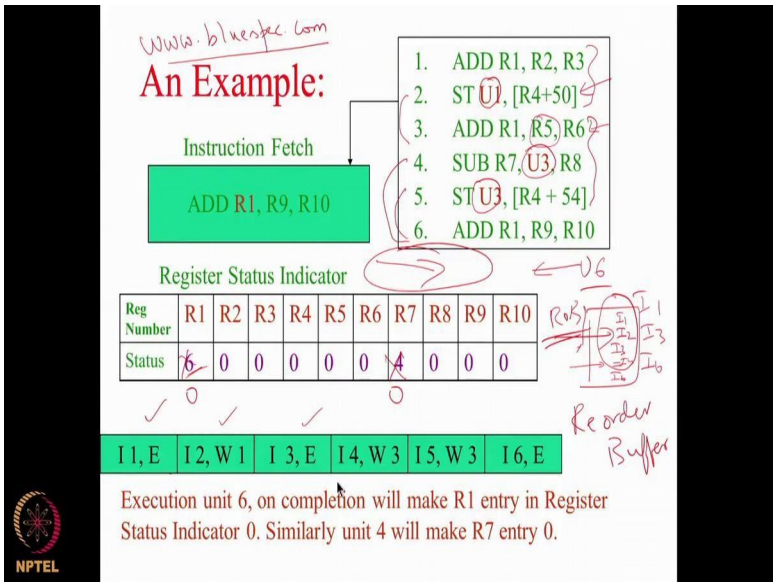

Because dynamically I go and schedule I go and say dynamically I have gone and said. So, this instruction this instruction will go to one and start executing this instruction will go to you know this instruction will go to 2 and wait 3 and all these things I have done who has done this hardware has done this and it has scheduled every instruction to this unit and say execute or wait. So, this is basically scheduling and this is done when the program is in execution. So, that is why we call it as dynamic instruction scheduling got this fine. So, the next we will go into the ok.

(Refer Slide Time: 13:49)




So, these small-small things see that operand forwarding and register naming is done automatically here.

(Refer Slide Time: 13:54)



And then when the execution unit 6 on completion will make R 1 entry to 0 if still it is 6 where there might be a other instruction which would have changed it and similarly unit 4 will make R 7 entry right and then we have seen the memory aliasing problem.