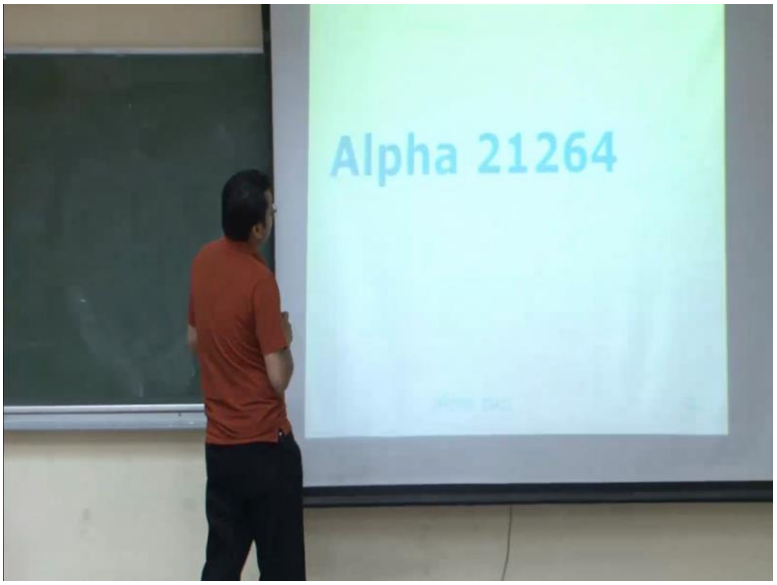
Computer Architecture
Prof. Mainak Chaudhuri
Department of Computer Science & Engineering
Indian Institute of Technology, Kanpur

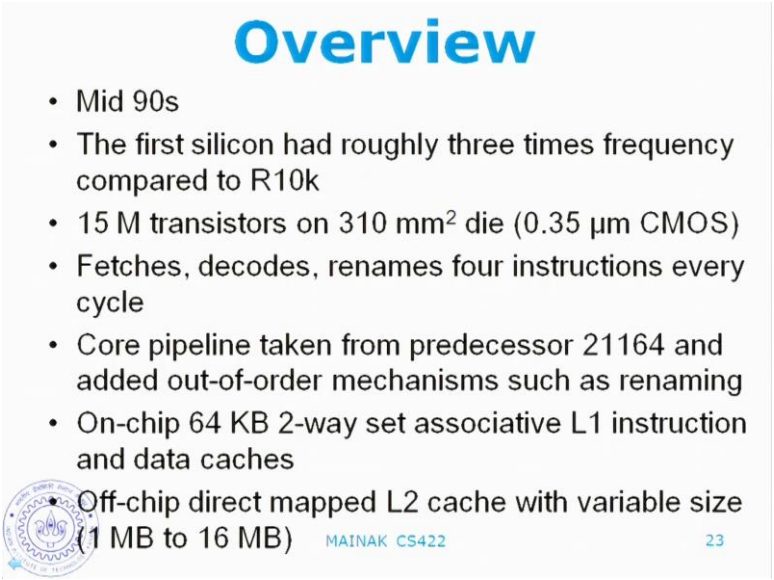
Lecture - 33
Case study: Alpha 21264

(Refer Slide Time: 00:38)



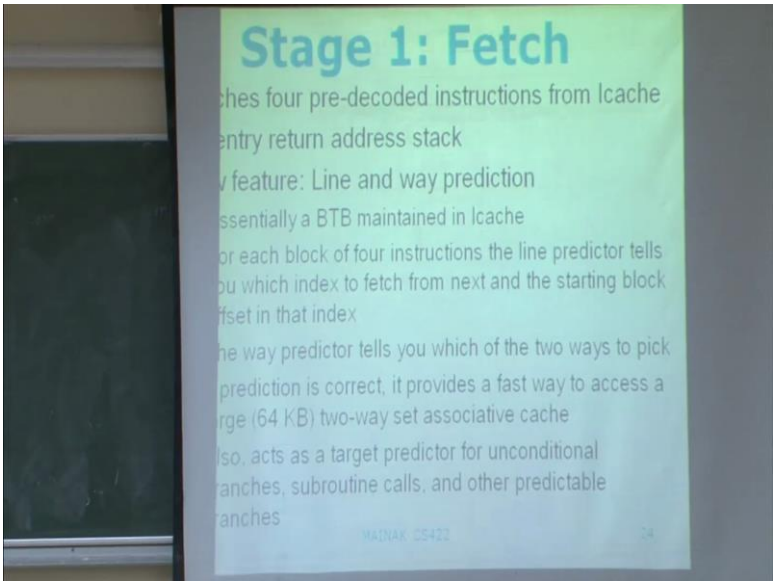
So, today we will talk about one of the last two processors that compacter, so last one was 21364. So, this one was before this ((Refer Time: 00:28)) 21464 was in designed an totally we will talking little bit about 464 also if we get time.

(Refer Slide Time: 00:47)



So, again just ((Refer Time: 00:50)) like its fix it, so anyway, so again just like R10k and the first silicon. So, it have the 15 million transistors for a 310 millimeter ((Refer Time: 01:20)). So, if you go back look at a mids or it have a die sized above 290 will display number of transistors was about 6 million, you can see that this also reflex the of the designers it would. So, many which at the same transistors size just like which is because for instructions every designed to the pipeline was taken from predecessor 2164. And essentially the out of order mechanisms such as remaining, etcetera. The basic pipeline verses borrowed from the producer which around 664 kilobyte two ways instruction data caches and off chip direct mapped L2 cache with variable size. So, again vary from 1 megabyte to 16 mega byte.

(Refer Slide Time: 02:35)



So, if stages a 4 pre-decoded instructions from instruction cache. So, to entry return address stack is a, so this processor introduced new feature that is Line and way prediction. So, is essentially proper use will be instruction cache, so work is a block for each block a four instruction the line predictors tells you which index to fetch from next and the starting block of set in that index.

(Refer Slide Time: 03:15)

Thus essentially let us say this is my instruction cache, which as a two ways curriculum on fetching from this curriculum cache in discuss in this next this way alright. And there a as the line predictor were are next points alright say essentially you just a predictor, which predicts well you say that well say in index next for instruction. So, that is what we should most of the time, however if one of these instruction is a branch instruction it would tell you the targets.

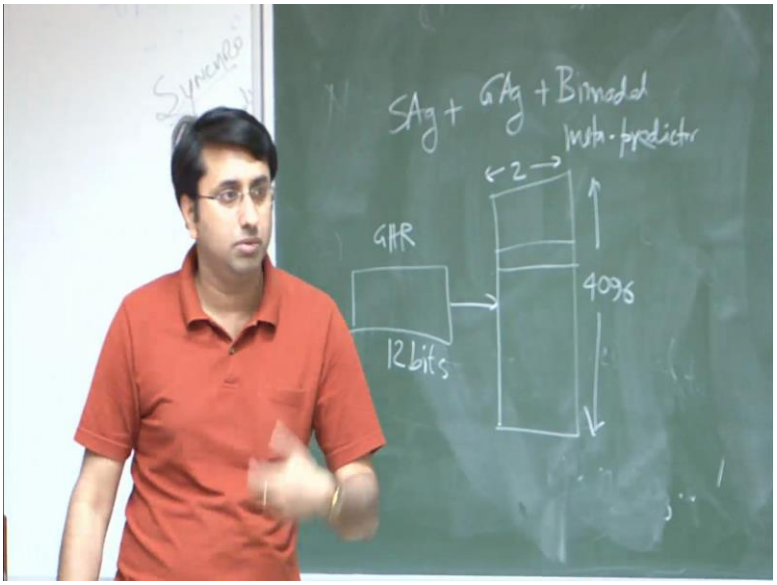
This is the next line different to line to, so that is a new feature the way predictor tells you which of the two ways to pick. So, at this index you have two possibilities right, so therefore this one or this one. So, the weight rate is of this the, so prediction is correct if provides a fast way to access a large to way set associative cache. So, otherwise the found that it was pretty much will possible to access for 64 kilobyte two way cache and forever frequency the process was targeting.

It has essentially here if you we will have avoid the time to effects completely to avoid the among time to you comparison of target with the we just ask the which line. But, off course if background we have to continue feedback other think we look of the index pre decoded index look of both the tax comparison find out the which way in the correct form. And if you predicted correctly then foreign otherwise the cancel whatever in fetch next in the predictor and in.

So, in also in act as a target predictor for unconditional branches subroutine calls and other predictable branches and say is a essentially in predict in scheduled instruction is a concept sophisticated branch predictor from. So, the 2 and 64 pipeline at the 7 cycle branch penalty is the minimum time between prediction and branch execution. So, a this is not larger then you will which at a most of the time.

So, this is a hybrid branch predictor called tournament the basic idea is to have two different types of predictors. That are good at predicting different classes of branches have a third meta predictor also called chooser a. That decides which of the two predictor to pick for each branch based on dynamic history if all this predictor class with if already advanced. So, this pick the predictor that is go back predicting a particular branch class.

(Refer Slide Time: 06:35)

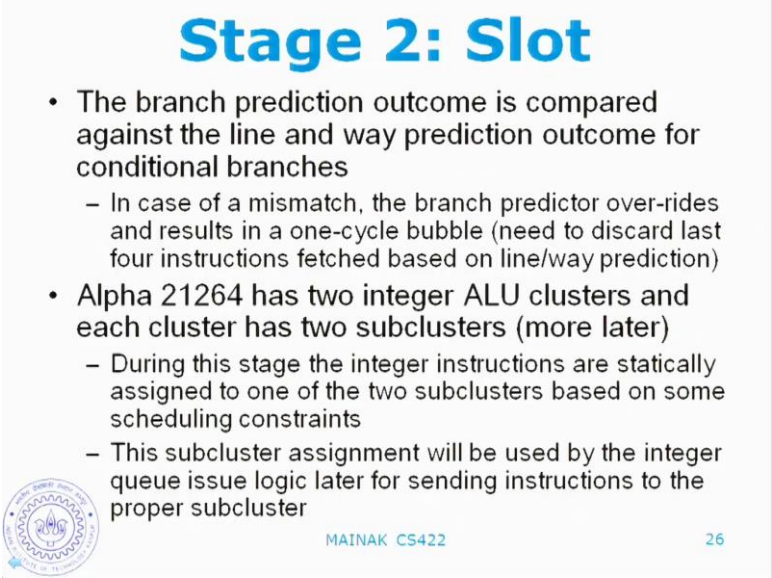


So, what is the what is the two components, so which actually use the SA g and the GA g with the Binadel linear predictor if you a go back in SA g and g share. So, one interesting thing that that act that 216 a deep was technique after the. So, if a recall a you know GA g predictors has a global history register. So, it 216 bits is a index is a table of 4096 entries 2 bits which. So, a so they always get ((Refer Time: 07:31)) and the this thousand of the GA g. So, a if which is look at the GA g isolation the problem is that the second cycle vacancy need prediction to the correct of the. So, it is create that your global history in lack in behind the correct because take a branch to statistic if between take a branches again that actually access her stage nature took that. So, one way to avoid this act is that you believable on which ever that forever the creditors we have the in that is in my history.

That is on speculating of credit of the global intricate and a what it by you SA g GHR is most of can one put way to by in the credit ion actual see that took otherwise off course quality is GHR. But, off course possible your predict the will make its when a process how many fixed design product. So, already sack as you on all mentioned for any way for go in back register map you will make a check when your my secures. So, you make these 12 bits of part of the check when ever in take the branch explanation to.

So, a in case of correct prediction there is no bubble and 216 for supports 20 in flight branches. So, after that in for in case of, so this means in the higher prediction and because it say that these are we have how that we say that we can have we can have allow in one half in the second that you are prediction of a index. If you planning how can you if a we will come out 0.97 0.98 our language, because similarly to way sells of this curriculum in. Otherwise we solve this soon direct me the directly that most like to if we called below that.

(Refer Slide Time: 10:12)

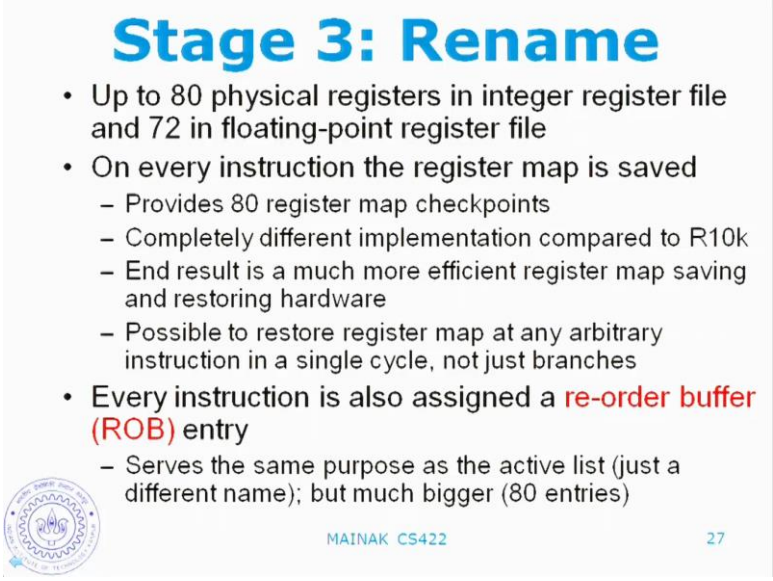


The second stage called slot, so the branch prediction outcome is compared against the line and way prediction outcome for conditional branches and in case of a mismatch the branch predictor over rides. And results the line way prediction and results the one cycle bubble. So, in which particular fetched believe the, so that is that is that is the first thing that you do this in the stage. Now, the 21264 has two integer ALU clusters and each cluster has two sub clusters looking about details about sub clusters we assume. The point is that during this particular stage basically instructions are statistically assigned to one of the two sub clusters based on ((Refer Time: 11:05)).

So, essentially you are saying that a statically by an instruction to particulars of a in that the what happens is that if you did diagonal index one of the such, finally the statistic

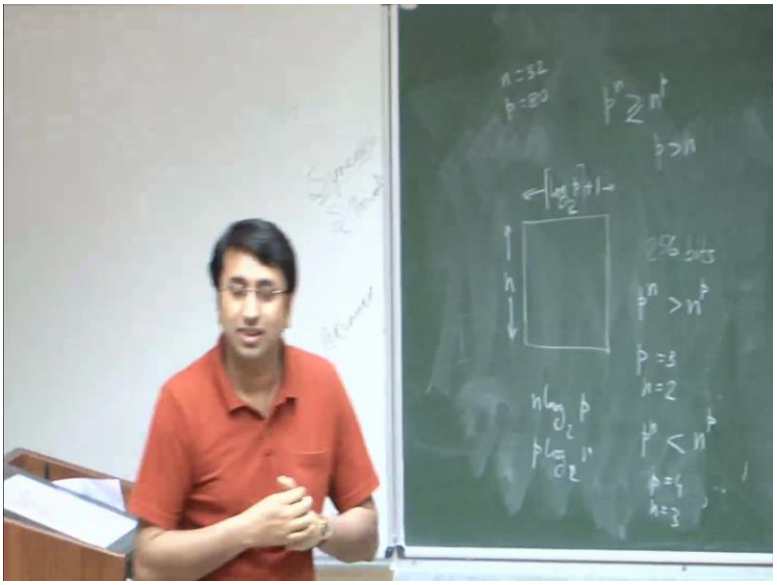
assigned of the actually be see the some ways. And other ways of the actually be and the statically designed the sub clusters and say the. So, statically designed that is way the designed main for of this stage this static what it instructions which that it is ok. So, this sub clusters assignment will be used by the integer issue logic later for set you structure for the a proper purpose.

(Refer Slide Time: 11:56)



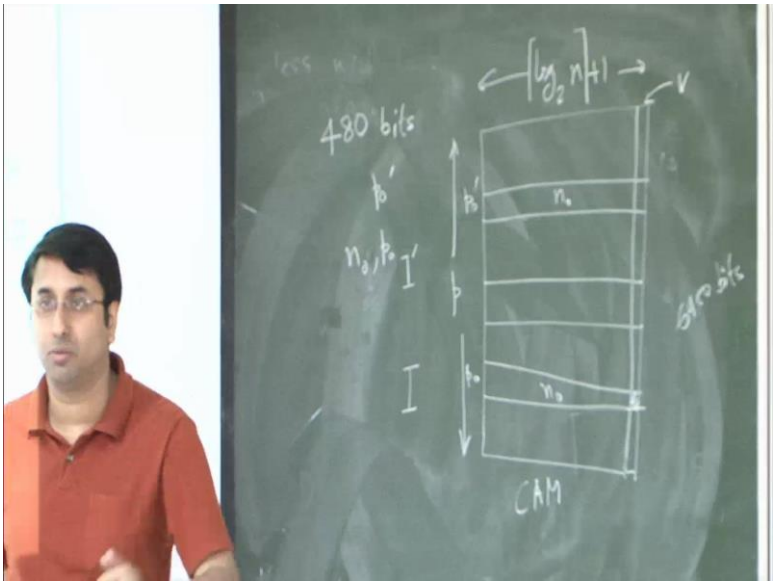
Third stage in the rename, so physically interesting quit with rename is fundamentally different on the way in have talk about in the class and see within your part and teach about. So, suppose up to 80 physical registers in integer register file and 72 floating point register file on every instruction the register map is saved. So, this provides different flexibility to because map is save at for every instruction of facilities more that if subject point of you register map. But, this also if that of you registers the point should not to be heavy should be the because essentially a. So, provides 80 register map check point essentially around the 80 instructions. So, that, so to same 80 points we should on required not in stage. So, completely different implementation compared to R10k. So, next on the look at that will be actually more actually in board.

(Refer Slide Time: 13:07)



So, recall that whatever we have discussed till now and also what in that we have n logical registers we have a log table which is at a increased by and we have if you of p physical registers each at would be log 2 p plus 1. So, this extra this total alright, so if you will calculate for 80 registers check points. So, by which before 80 byte as usual and b, so will very activate. So, in this case how much this in check the 80 alright you will able to. So, want check point is to 52 bits we are the every check points, so that is allots a data to need.

(Refer Slide Time: 14:36)



So, instead of this what we will do is, so they have a p entry table a p physical registers and each entry is log n plus 1 this point look at alright. Say again this particular begins of the have in, so this is line then ((Refer Time: 15:04)). So, how ((Refer Time: 15:09)) whatever you get logical register you make a full register compatible alright have exactly have a inter back and that when see whatever at the p that is the role number of that is to be the physical registered.

So, which one is better, so roughly computing n log p with p log n which was better we can ((Refer Time: 15:49)) we can actual that p is bigger than n. Then solve this and a established in a what is correct which was clear one example for each p n is bigger than n p to if you take p equal to 3 n equal to 2 correct, p equal to 4 n equal to 3 say that is a. But, any way for this designed point you can establish an electron one of the should be, so in this case calculate. But, I think this is in a large for make in that establish what it make what is equal result how will checks for you this in the dynamic table in the bits forever this, so we can calculated that is n equal 32.

So, the continue on 480 bits for in exists to right at p because each entry is equal to p 6 bits I do comparison associative comparison with all entries it is a CAM not a DAM does not is the is 80 bits the bits see it have a large a active. So, for 80 check point if you

decide to check point is countable. So, in which a check points say this way of check point the this also way for how to why how to not will be check point is that easy have to at a taken it is enough to say needs that is all. So, thing of a instruction right correctly short of things of mat how get will the to logical registers cannot have.

So, I behind article register cannot make multiple existing maps right, so that would be only n. Then the that it alright see that look at this particular column as final there exactly n the remaining with that a. So, currently whatever is batch, so we have logical registers there map is a 10 physical registers alright the until this instruction become is if at pick up any this currently allocated physical registers it will see have this same designed an which. So, look at extraordinary which is a alright.

So, this particular some logical instruct here as a is a correspondence physical in this map why does this map get feet on the same logical instruct get is need defined alright defined that the instruction in the step this map. So, instruction the new map actually alright, so new map does not exists that this point alright it is in the feature. So, of the pick point the current instruction when it graduates the claim that all this map 6 bits energies. Now, the physical registers in a bit if you at look at the any of the map physical registered that will only for the next producer branch that is an as idea features that then why did why 6 particular logical register here we should map.

So, currently an looking at only the most recent producer of each at the logical register in this map there that is a that is alright and that will talk what will change that and in this particular instruction graduates and will not to be change and the particular change. So, a change that if a check point to very big it is a lot see back taken our which on introduction which on is check point map following two step copy the. That is it for the balancing column is a exactly 10 bits form which ever activated essentially take ever would map instructions. What will would from a main logical registers a main logical registers that is right it will map to physical registers.

So, that is wants to say, so possible instruction which a predictor this is a stage currently it has some way this point if the column. So, was I saying that is in features at any point in time before this instruction gadgets if you want all take the register map back to back

instruction in the is adopt just copy the column what it was at that change right.

So, talking about remaining current instruction time the talking about instruction before the this I line which property which is already which is already remaining and whatever give of the map in time producing already take. So, without what happens I 12 to this more instructions. So, is right I 1, I 5 produced registered who says some logical registered in what the previous back was p one series right. So, I unless suppose by in between there was more instruction produced by. So, I is details we not say will not right.

So, the point this you can just point this what there are you make if an a became biggest size fully for you instructed for a check point become where it take subject point form when will to commit of I how could the checks it is there are instructions. Now, of the registers which map a this point cannot be speech they will remaining take it a that is what I am exploiting to argue that which is want to go back to the check point this particulars stage you this copy the dynamic copy on is a that.

The enthusiasm is that for a story 80 checkpoints in we meet 6400 weeks 80. So, after the larger than the you are beat the certain check found because attitude was happening only two checks founds which is equal to how it is perfect. But, if it did not having 80 checks found in huge any questions ask here. So, actually that go recycling construction right recycling application like I would said example I choice the which is logical register in law to be assigned in the.

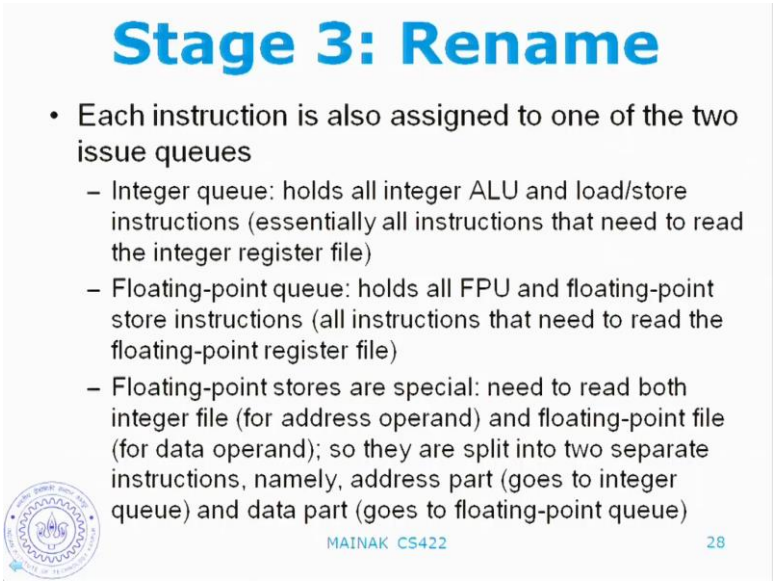
So, if I go the row of p which is like a this one basically this is p 1 if you say not here and b to be not right yes. So, previous we did not find this is like a, so q not pie you loses did not. But, this will be loosed yes you are first, so I cannot commits to recycle phenomenon could not acting that. So, considered for it is a considered I, so before you make this change to a check point good form right. So, if I want to roll that this will and actually, so that copy the. So, that check point is it because of only after age because its new plan, so valid beat follow up exactly following through for of course.

It will say that after I has committed after that for a I, now want to look that to has not

because register back to, now have been change you got that is the not. So, much more efficient register map saving and restoring hardware.

Sir because saving that mono of complaining that amplification what we happen is the on your pipe line depending on how system is on service fast. So, this will the love into the actually support any extension for each instruction if you exhaust for. So, it is arrives into now technician extension in process seven it will happen in a letter you can notified save natural. Although you can check point all this I do not allow into check from branches you only. So, possible to restore register map at any arbitrary instruction in a single cycle not just branches, so a if there is stage every instruction also assigned a reorder buffer entry. And this serves the purpose same purpose as the active list just a different name, but not bigger. So, active list solve to agrees is eighty entries, so the arise if you have 80 kind instructions.

(Refer Slide Time: 31:23)

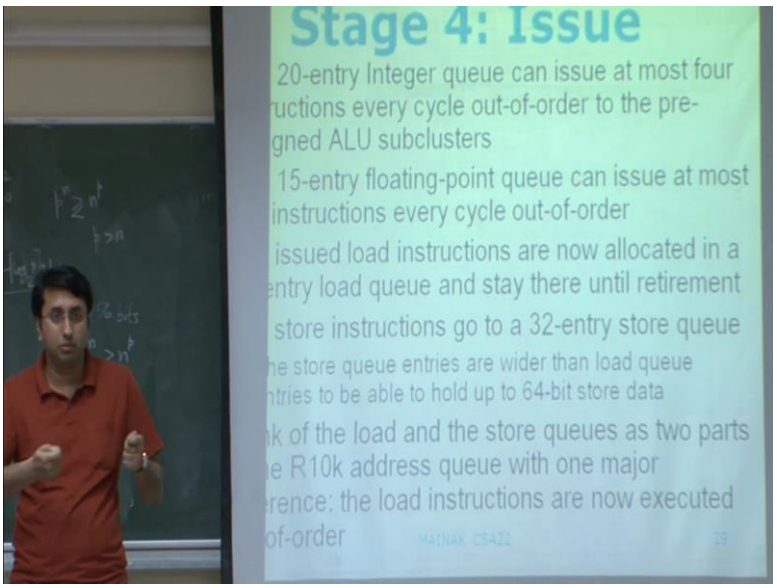


Each instruction is also assigned to one of the two issue queues. So, there are essential two queues one is integer queues another one is cubic point queue. Integer queues holds all ALU and load store instructions essentially all instructions that need to read the integer register file. So, load is the different queues for load second one floating point queues holds all FPU and floating point store instructions. So, essentially these are all

instructions that need to read the floating point register file.

So, there is a slight problem with floating point four which has they we to reach extra file floating for critical. So, say that to be instruction in the floating point queues, but for address. So, floating point are special need to read both integer file for operand and floating point file or data operand. So, they are split into two separate instructions namely address part goes to integer queue loads to instruction queues for loads. So, there here queues the defining task for each floats.

(Refer Slide Time: 32:58)

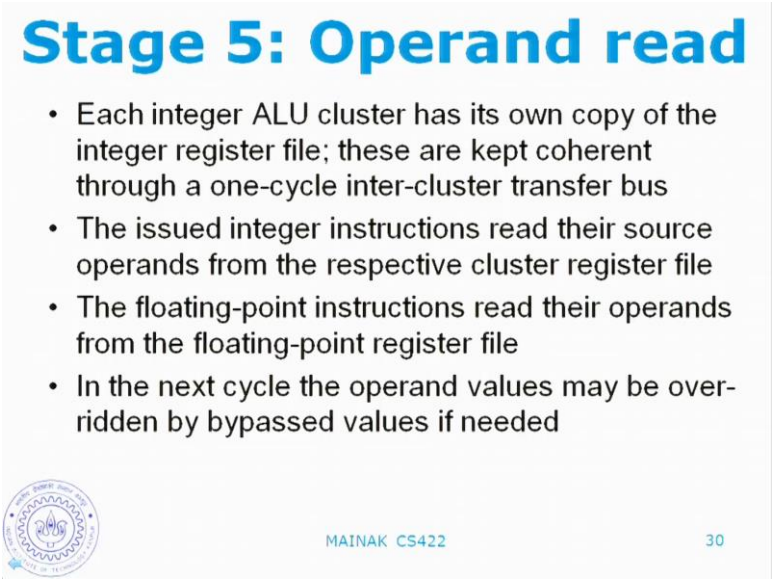


However, issue that 20-entry integer queue can issue at most 4 instructions every cycle out of order to the pre assigned ALU sub clusters, so we have already talked that at the start part 15 entry floating point queue issue at most 2 instructions every cycle out of order. So, I think that issue the sequence, because one requirement is that whatever is present in the queue there must be maintained according to the image that is very important. Because that would guaranty is essentially in the cycle if there are multiple instructions are become ready the older instruction.

The issued load instructions are now allocated in a 32 entry load queue and stay until retirement. So, most they you have a issued from the integer queue they stay another the

store instruction go to a 32 entry store queue. The store queue entries are wider than load queue entries to be able to hold up to 64 bit of store data if for that data base store to cash in that. So, the thick of the load and the store queues as two parts of the R10k address queue with one major difference the load instructions are now executed out of order which was not is integer queue to said does not instructions.

(Refer Slide Time: 34:54)



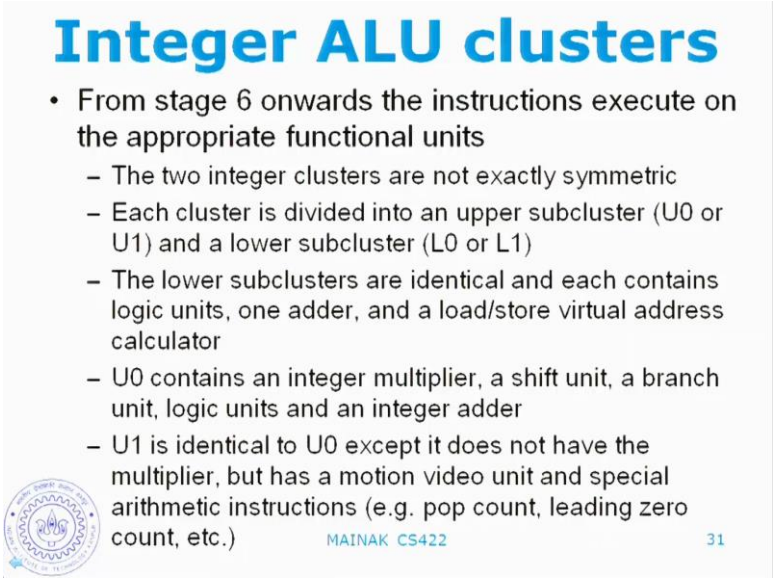
Stage 5 started operand read the integer ALU cluster has its own copy of the register file with I a register file these are kept coherent you a one cycle inter cluster transfer bus. So, if look at there are two way to clusters at these stage each ALU clusters has its own copy of the integer part. So, that right it has what cluster if have exam, so a one cycle inter cluster transfer bus if a single cycle if an actually break it for him which actually has simplication. Because know what it each is that suppose say in the instruction producer register 10 with cluster1 want and is the instruction which consumes register 10 to cluster 2. So, this cluster 2 instruction give before this what side to make your cluster.

So, a the was what enough actually to puts instructions belong to the same depend the stage the same the instructed which an out as per one cycle because it happen what we happen is that one bus may be become very crowd. So, a scheduled the there scheduled the takes care mis, so they issued integer instructions read their source operands from the

respective cluster registers file. The floating point instructions read their operands the floating point register file the next cycle. The operand values may be over by the bypassed values if needed. So, this is the traditional bypassed want yeah very good yes very good question why do why do the like.

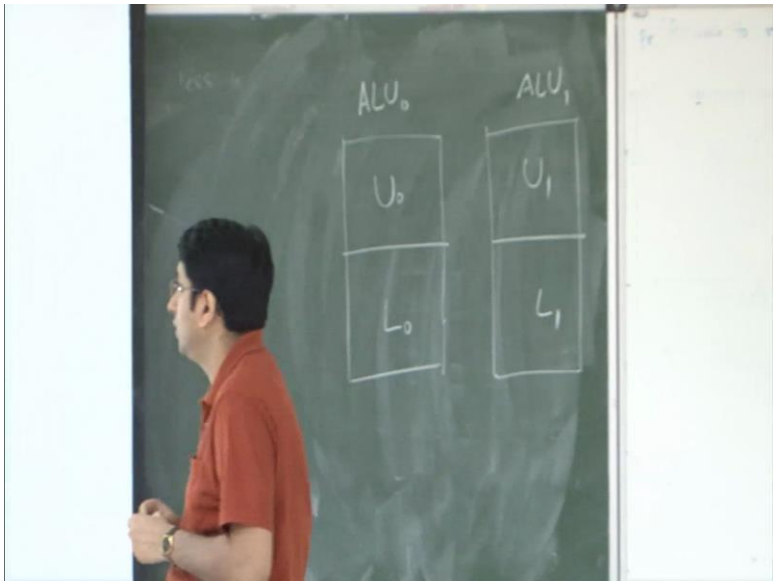
So, in this case stage does not matter, so say think aborted that to a suppose that the clusters. So, then if a look at each of these registers point right they we have some number of points right which a we find the number of added right that is so on. So, what a form to as that is what side to penalty was formatted the registered they essential that the scheduled your required this particulars one cycle transfer two people make sure that the value the instructions depend the instructions get related to...

(Refer Slide Time: 37:50)



So, from stage six onwards.

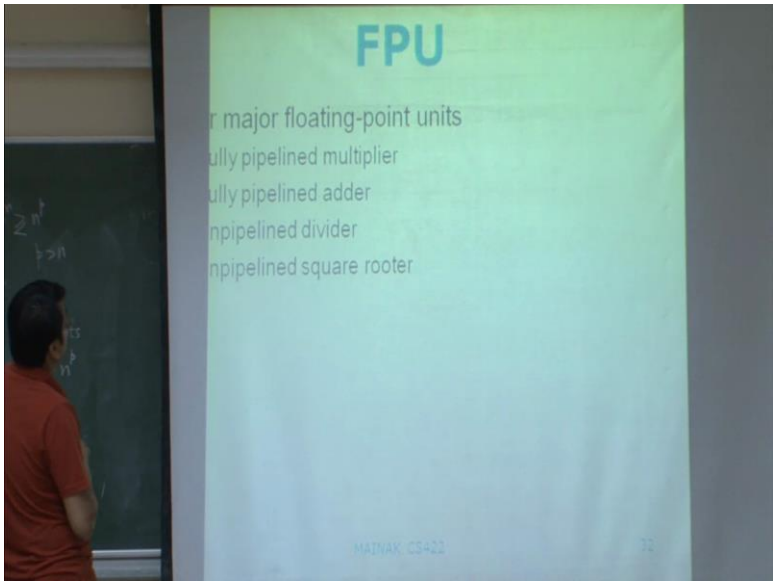
(Refer Slide Time: 37:56)



The execute on the appropriate functional units the two integer clusters are not exactly symmetric each cluster is divided into an upper sub cluster. And that is L 0 U 0 in the reverse of clusters L 0 L 1 essentially the alright ALU 0 is ALU 1 the lower sub cluster are identical. And each contains logic units one adder and a load store virtual address calculator, so that is your... So, your logic we have added you see of the contains an integer multiplier a shift unit a branch unit logic units and an integer adder. So, that is a your use an you want is identical to U 0 except it does not have the multiplier. But, has a motion video unit and special arithmetic instructions to like population count may be use an count etc.

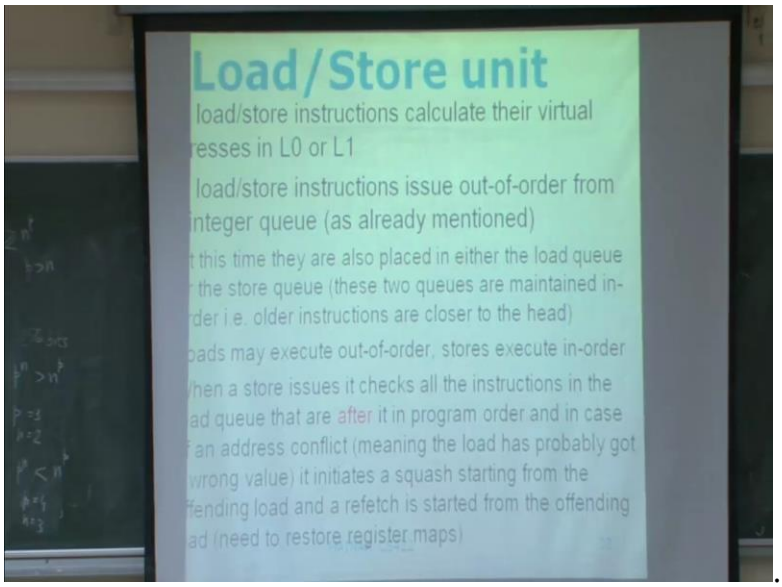
So, this particular instructions pop count counts a number of wants a particular alright and basic 0 count is essentially does not how many 0 a there on the. So, all the see over all what you get is one multiplier there are four logic units because all sub clusters have more which there are three adders sorry four adders and two branch units two shifters and two upper sub clusters.

(Refer Slide Time: 40:25)



The floating point unit, so there are four in number whatever things a fully pipelined multiplier a fully pipelined adder an unpipelined divider an unpipelined square rooter. So, by a more this that there is only to an divider the actually you this one.

(Refer Slide Time: 40:52)

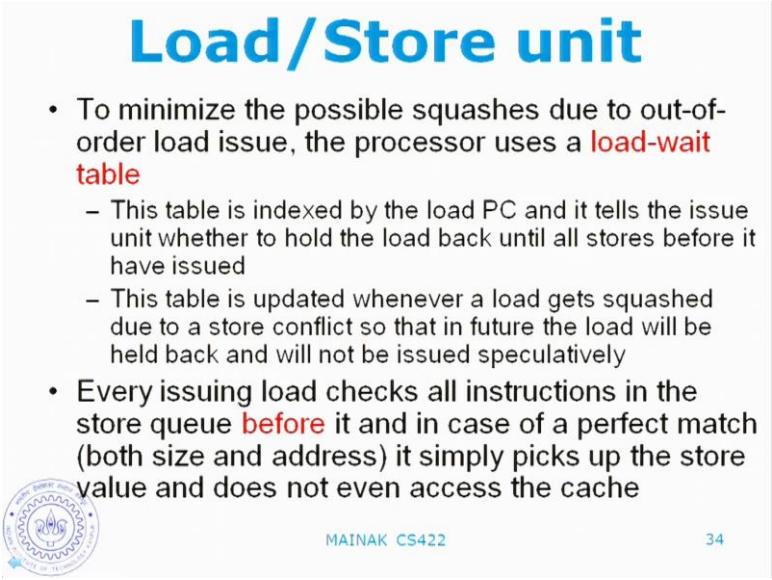


The load store unit load store instructions calculate their virtual in L 0 or L 1 the issue

out of order from integer queue as already mentioned at this time they are also placed in either the load queue. The store queue these two queues are maintained in order I e older instructions are closer to the head. So, such a essentiated what it means that was you puts sub thing a load out of order they can will put in the load.

So, loads may execute out of order stores execute in order it is a store issues it checks all the instructions in the load queue that are after it in program in the program order and in case of an address conflict meaning. The load has probably got a wrong value it initiates a squash starting from the offending load and are fetch is started from the offending load. So, need to restore register maps, so this all the discuss for away you right to that check point is register map.

(Refer Slide Time: 42:23)

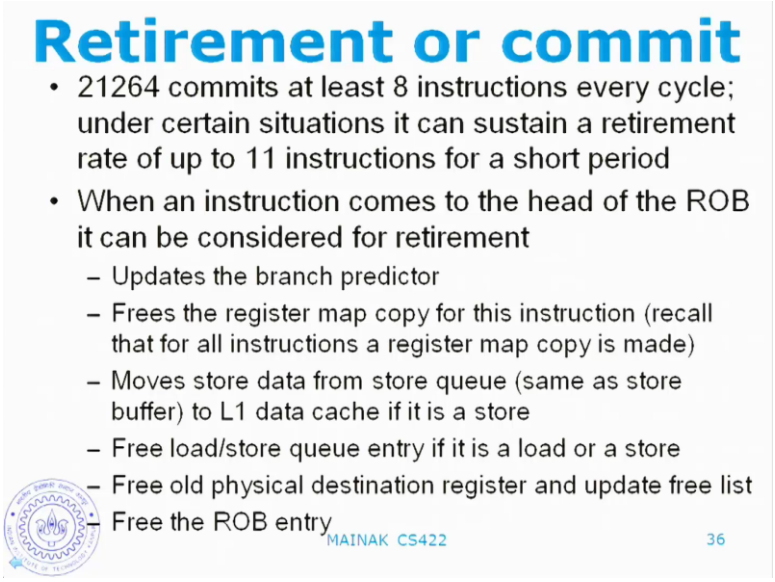


So, minimize the possible squashes due to out of order load issue the processor uses a load wait in. So, load this table is indexed by the load program and it tells the issue unit whether to hold the load back unit all stores before it have issued and how to a decide this table is updated whenever a load gets squashed due to a store conflict. So, that in future the load will be held back and will not be issued speculatively. So, it is very simple taking is identify load instructions that a they would be complicit load will be held back and will not be speculatively.

So, every issuing load checks all instructions in the store queue before it and in case of perfect match it simply picks up the store value. And does not even access the cache its do not for an this in the load all instructions in the store queue before it in R10k we talked about load hit speculation in connection with issuing load dependents that is right. So, say that will be issue of dependent upon the speculated without only with 10k always optimistically speculates that the load will hit is right always the speculation. And if it misses we will have to squash the dependent and re issue it later when the load value is available.

We could have issued an independent instruction in that issue cycle if we knew that this load would miss. So, if could have essentially one issue cycle is wasted if the load misses. So, a 21264 uses a load hit miss predictor when a load issues it predicts using a table based on the history whether it is going to hit or miss this time just like branch predictor tells you taken or not taken. So, it is the final predict hit on the prediction is hit it is a predictions take them only the issues of the make held issued different of its predictors.

(Refer Slide Time: 44:57)



So, a last stage of the pipeline retirement and commit 21264 commits at least 8 instructions every cycle under certain situations it can sustain a retirement rate of up to

11 instructions for a short period. So, in this while is my retirement rate both than my cycle and saying the title sustain of the 11 instructions there which cycle they always at those not sure at the. So, will you have brought they exists lots of other resources, so these are essentially you know copy for instructions the just are pollutes. So, the soon are you late the, so which instructions comes were the each can considered for if update the branch creditor which pick the registers copy for instructions.

So, remember that every instructions has a register map check point it is because you will not together, if store data from store queue to add on data cache is store free the load store queue. And if free the total stores if free the load physical destination and updates free list free the, if make desired. So, next time go to the topic of Intel Pentium.