

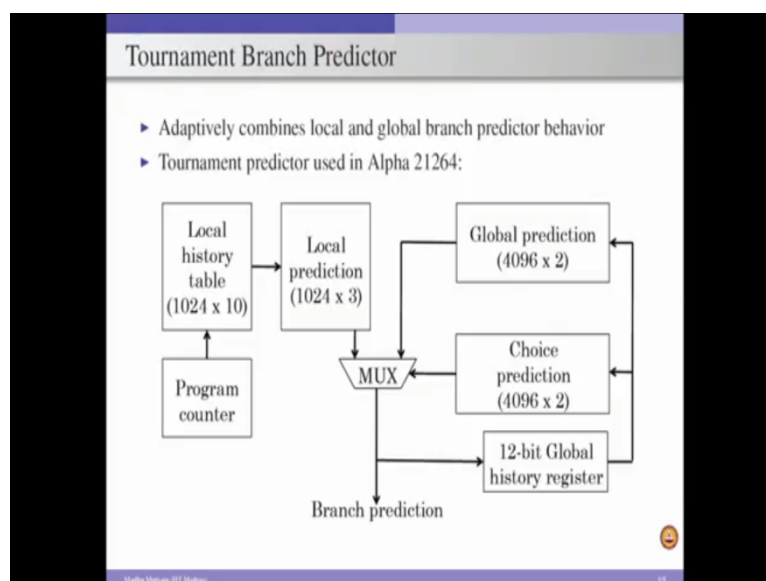
Computer Architecture
Prof. Madhu Mutyam
Department of Computer Science and Engineering
Indian Institute of Technology, Madras

Module – 06
Lecture - 22
Advanced Branch Prediction Techniques (Part – 2)

So in the last module, we discussed correlated branch predictors which are example for two level branch prediction. And now in this module we are going to discuss tournament branch predictor, which is also another example for two level branch predictors. In the case of correlated branch predictors, we consider a local history table and a local prediction table. Each entry in the local history table is going to keep track of the outcomes of all the branches, which are mapped to that particular entry. This correlated branch predictor cannot exploit any information about other branches, which are mapped to a different entry in the local history table.

So, in order to come up with a mechanism which uses the outcomes of other branches to predict the outcome of different branch, we can use this tournament branch predictor. So, here in this tournament branch predictor, we actually consider the local history table, as well as the global history. This local history table always keeps track of the outcomes of all the branches which are mapped to a single entry. But the global history is going to keep track of the outcomes of all the branches that are executed so far.

(Refer Slide Time: 01:35)



So, effectively it combines both the local and the global branch predictor behavior to come up with better accuracy prediction mechanism. This is implemented in 'Alpha 21264' processor, and after that there are several other processors also have used the variant of this tournament branch predictor mechanism. So, the arbitration of this tournament branch predictor is something like this, it consists of a local history table, the local prediction table, the global prediction, the choice prediction, and there is a global history register. So, this global history register is going to keep track of the outcomes of the last few branches, which happen in the program and these branches can be same branches or different branches.

But whereas, in the case of local history table it is going to keep track of the last few outcomes of the same branch. Here, we are considering this design, whatever is implemented in alpha 21264, which has 1024 entry in the local history table, and each entry is having 10 bit information. So because a local history table has 1024 entries, so we use ten bits from our branch instruction, and index into this table to get one particular entry and which is going to give another 10 bit information. And this 10 bit information we use to index into this local prediction, this is going to give a 3 bit outcome.

And using a 3 bit outcome, we will make a local decision, because whatever the decision, whatever the prediction, we are going to make from this local prediction is because of the behavior whatever we observed for the similar branches, which are mapped to the same entry. In other words, this local history table is going to store the last 10 outcomes of the similar branches. When I say branches are similar, these branches have their LSB 10 bits same. So, once LSB 10 bits of branch address is same, they are mapped to the same entry in the local history table. So, as a result, all the similar branches are going to map to the same entry in the local history table, and so this entry is going to give the last ten outcomes of all these similar branches.

And using this, we are going to index into the local prediction table, and whatever the prediction we are going to make from this table is because of the outcomes of the similar branches. So, as a result, we call it as a local prediction. So, this local prediction table is going to give us 3 bit information based on the ten bit outcome, whatever we get from this local history table. And once we have that, we give that value to this MUX, but we are not going to make the decision immediately, and we are going to consider the decision coming out of this global prediction table also, and based on that we are going to take a decision.

And here this 12 bit global history is going to keep track of last 12 outcomes of the all the branches, whether the branches are similar branches or different branches, it is going to keep track of the latest 12 branch instructions, executed by the program. And using this 12 bit, we are going to index into this global prediction table, because this global prediction table has 4 k entries. So, using twelve bit we index into this and we map to one particular entry and that entry is going to give us a 2 bit information. And this 2 bit information is again is going to give us the outcome in a global sense. Each entry in the local prediction table is mapped to 3 bit dynamic branch predictor, so accordingly based on this value we are going to say whether the branch is going to take place or not.

If the value is less than 4 then we say the branch is not going to take place and if the value is 4 or more, then we are going to predict the branch is going to take place here. Whereas, in this case each entry in this global prediction table is mapped to a 2 bit dynamic branch prediction and if the value is either 0 or 1, we predict that the branch is not going to take place and if the value is 2 or 3, we are going to predict the branch is going to take place.

So, based on this 12 bit global history register value, we are getting one entry from this global prediction table, which is going to give us 2 bit information. And now if these two entries are giving the same prediction, then we will just proceed further and then we will consider that as the final branch prediction.

So, that means like for example, if this is going to give a value which is more than 3, so that means like we are going to predict the branch is going to take place, according to this local prediction. And similarly, if this is also going to give us a value either 2 or 3 then global prediction mechanism also going to predict that the branch is going to take place, because both are predicting that the branch is going to take place, then we will consider that as our branch outcome and proceed further.

On the other hand if this is going to predict that the branch is not going to take place and similarly, this also is going to predict that the branch is not going to take place, then in that case also both are agreed in that decision. So, we just consider that as our final decision and proceed further, stating that the branch is not going to take place for this particular branch. But sometimes the local prediction unit says that the branch is going to take place, and global prediction table says that the branch is not going to take place, or the other way. If this is

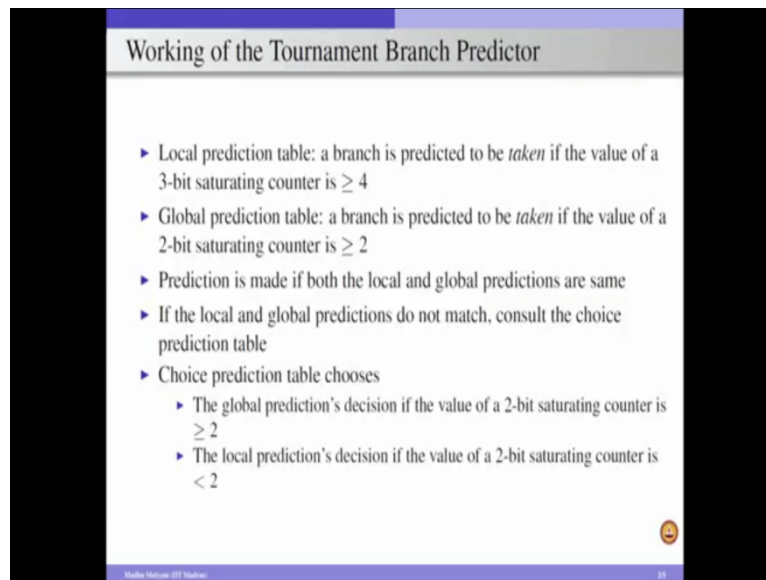
going to say the branch is not going to take place and this says the branch is going to take place.

So, whenever these two tables are going to give a different predictions, then we have to consult the choice predictor. The choice predictor also consists of 4 k entries and each entry is having the two bit information and if the value of this choice predictor is 0 or 1 then we will go by the local prediction decision. If the value is 2 or 3, we will go by the global prediction decision. So, using that we will make a prediction and every time based on our branch address, we take the last 10 bits of the branch address and we index into this local history table, we get the last 10 outcomes of the similar branches. And we index into this local prediction table and this is going to give us the prediction outcome.

And similarly, we use the last 12 branch outcomes of all the branches executed till this point, and we use this to index into this global prediction table and which is going to give us a two bit branch outcome, and based on that we are going to make a branch prediction. And after the branch is resolved then we will update the local history table by pointing to the appropriate location in this, and then we shift that value into the corresponding entry. And also based on the outcome, we are going to change the values of these local prediction entries.

And similarly, the recent outcome whatever we got we insert into this global history table also, effectively the LSB bit will be pushed out, LSB bit in the global history register will be pushed out and the new outcome is inserted at the MSB position in this twelve bit register. And also we are going to update our global predictions and similarly, the choice prediction, whenever there is a choice predictor is involved in taking the decisions. So, this is the overall operation, we do with the tournament branch predictor. So, in summary, in order to work with this the tournament branch predictor.

(Refer Slide Time: 09:53)



The slide is titled "Working of the Tournament Branch Predictor" and contains the following bulleted list:

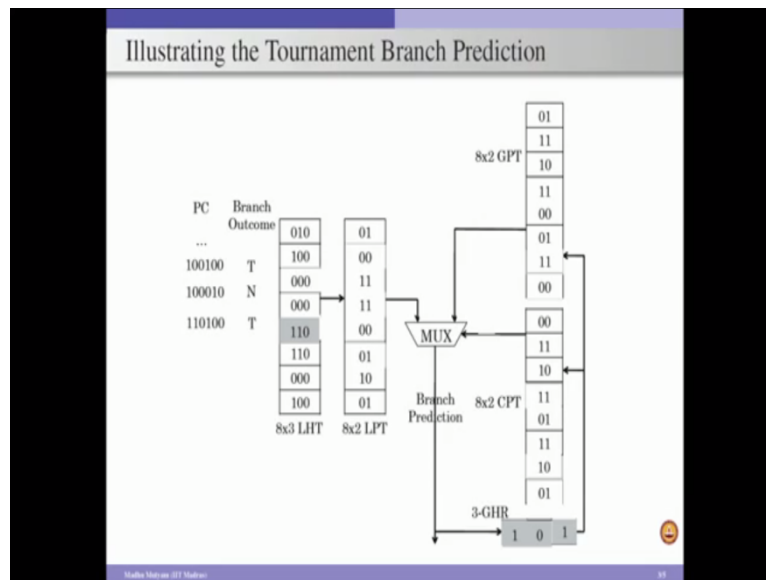
- ▶ Local prediction table: a branch is predicted to be *taken* if the value of a 3-bit saturating counter is ≥ 4
- ▶ Global prediction table: a branch is predicted to be *taken* if the value of a 2-bit saturating counter is ≥ 2
- ▶ Prediction is made if both the local and global predictions are same
- ▶ If the local and global predictions do not match, consult the choice prediction table
- ▶ Choice prediction table chooses
 - ▶ The global prediction's decision if the value of a 2-bit saturating counter is ≥ 2
 - ▶ The local prediction's decision if the value of a 2-bit saturating counter is < 2

At the bottom right of the slide, there is a small orange smiley face icon and the number "19".

We will consider set of tables, one is a local prediction table and branch is predicted to be taken, if the value of a 3 bit saturation counter is ≥ 4 . And the global prediction table also is used in this tournament branch predictor and a branch is predicted to be taken if the value of 2 bit saturation counter is ≥ 2 .

And prediction is made if both the local and the global predictions are the same. If there is a conflict then we are going to take the help of choice predictor, and the choice predictor is going to choose the decision from the global prediction table, if the value of a 2 bit saturation counter is ≥ 2 otherwise it is going to take the decision from the local prediction table. Having discussed this tournament branch prediction mechanism, now we are going to consider an example to illustrate the effectiveness of this mechanism.

(Refer Slide Time: 10:52)



So, here for this example, we are considering eight entry local history table and each entry in the local history table is having 3 bit information. And we also consider the eight entry local prediction table, but each entry in the local prediction table is having 2 bits. And we have a global prediction table which also has eight entries and each entry is having 2 bits. And there is a choice predictor which also has eight entries and entry is having 2 bits, and we consider 3 bit global history register.

So, we are going to make the prediction for a given branch based on the decisions provided by the local prediction table, and the global prediction table and also taking the feedback from choice predictor table. In order to illustrate the tournament branch predictor, we consider 2 branch instructions whose LSB 3 bits are not same. Effectively these two branches are pointing to two different locations in our local history table.

By the way, so we are not resetting these values, so we are showing each entry is having different values and so on. This indicates that the tournament branch predictor is already executed few branches, so that is what is indicated by this. Now we start from a particular time, where already these counters, these tables, are filled with the different values.

Now we will consider a new branch instruction, whose LSB 3 bits are 100, when we consider this branch instruction, now we are indexing into this local history table and this has 011, so that indicates that we have to go to local prediction table to the third location. So this is 0th location, first, second and third location and this third location is having a value 10. So

according to our 2 bit dynamic branch prediction mechanism, if the value is 2 or 3 we are going to predict that the branch is going to take place. So, this is going to predict that the branch is going to take place.

And similarly, our current, the 3 bit GHR has a state 110, so effectively this is pointing to the sixth entry in our eight entry tables. So, we are going to go to the sixth entry in the global prediction table and which also has the value 10. So, this also again a 2 bit dynamic branch predictor, so effectively we are going to predict, the branch is going to take place according to this GPT also. Both are going to say that the branch is going to take place, as a result there is a match in their decisions, so we are going to predict that the branch is going to take place and we proceed further. If we assume in reality that the branch is taken, so as a result, we are going to now update this contents, because the branch is taken and these two predictors are predicted that the branch is going to take place so we have to update these values.

Now, we change the value previously it was 10, now we change that to 11 and similarly, here also its 10, we change that value to 11. And also, we are incrementing the choice predictor value also to incline towards the global prediction. And after that, now we have to insert this latest outcome into the corresponding the history tables, one is the local history table; the other one is the global history register. So, here we have 011, now after inserting this 1 it will become 101. And similarly, it was previously 110 now after inserting the latest outcome it will become 111. So, this is the state of different tables at the end of the branch is completed.

Now, we will go to the next branch, which is actually pointing to a different location in our LHT, because the LSB 3 bits here are 010, which is different from the previous instructions LSB 3 bits, that was 100. So, once this is 010, so we are indexing into the second entry in our local history table, which has the value 001. And once this 001 is there, so we will go to the first location in our LPT, and this has the value 01. So, 01 indicates that branch is predicted to be not taken. And similarly, the global history register has the value 111, which is 7, so we are pointing to the seventh entry in our global prediction table which has the value 01.

So, this also predicts that the branch is not going to take place, so both are 00. So, they are matching in the decision, so we are talking the decision as it is. So, we are predicting that the branch is not going to take place here. And now, we will see, what is the actual outcome of this branch? so the actual outcome is not taken. So, this is effectively in line with our prediction, now once the branch outcome is there, now again we have to do the necessary

updates in our tables. So, previously it was 01, now we are decrementing that to 00 and similar here 01 is decremented to 00.

And so we are updating this CPT's to make it tend to the local predictions, and finally we have to insert this value into our LHT and GHR. So, after that it will become 000 here and it will be 011. So after this, now there is the next branch, this branch is same as the branch we already executed and so as a result it is now pointing to the fourth entry in the LHT, and which has the value 101 now.

So, once it has 101 we are pointing to the fifth entry in our LPT and which has the value 0, so that indicates that according to our local prediction table the branch is not going to take place. We give that value to the MUX. But our global history table has the value 011 which is 3, so we index into the third entry in our global prediction table, and the third entry in the global prediction table has the value 11. That indicates that, this is going to be taken. So, effectively according to global prediction table the branch is going to be taken place.

So, this has given a value 0 and this has given a value 1. Now which one we have to select? Whether we have to go by the local decision or by the global decision. So for that we are going to take the help of this the CPT, and the choice predictor is now indexed using this GHR value, that is 3. So, we take the third entry in this CPT and the third entry in the CPT is showing the value 2, so which is more than 1. So for any value which is 2 or 3, for any entry in the CPT if the value is either 2 or 3 we have to go by the global prediction decision. If it is 0 or 1, we have to by the decision of the local prediction.

Since, if the value is 2, so we will take the decision based on the global prediction. So and this is what we are going to predict, so effectively we predict that the branch is going to take place for this branch instruction. And now see, what is the actual outcome of this branch? and this is also taken. So, effectively our prediction is correct according to the branch outcome. Now, we have to update the contents accordingly, so once it is taken, so we have to change this value. And so the previously the value was 00 now it has changed to 01, and here the previous value is 11, but so 11 is because we are considering a saturation counter, so there is no change in the value here.

Since, our decision is correct and also the decision is taken based on the global prediction, so we are updating the choice predictor entry to incline towards the global prediction. So, previously it was 10 now we increment that to 11 and so after that, we insert this prediction

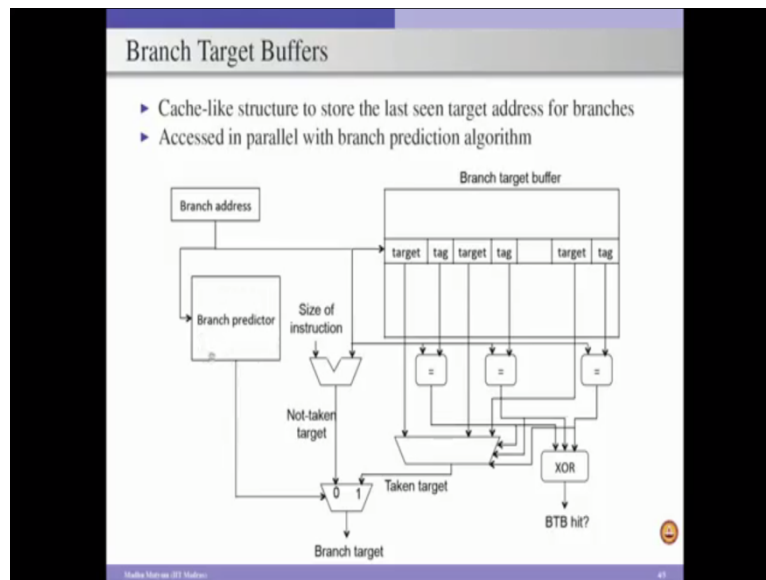
outcome into the corresponding entries in LHT and GHR. So previously it was 101 and now it becomes 110 and similarly, here previously it was 011 and now it becomes 101.

So, this way, when we have different branches, when I say different branches, the branches which are not mapped to the same entry in our local history table. So, we can map to different locations in our LHT, and also we can exploit the global outcome of all the branches. And using this global history register as well as the global prediction table, combined with our local history as well as the local prediction tables, we can make proper decision to come up with high accuracy predictions. So, this is about the tournament branch predictor.

And there are plenty of branch predictors proposed in computer architecture domain, mainly because of the importance of this branch instruction, with respect to the system's performance. So, having discussed this couple of the advanced branch prediction mechanisms, now we are going to look at the branch target buffer. So, as I mentioned earlier, in order to deal with the conditional branches, we need two components, one is the branch prediction mechanism, the other one is the target address identification. So, first part to deal with the branch prediction mechanism, we already discussed various branch prediction mechanisms and to get the predicted target address, we have to go for the branch target buffer.

So, the branch target buffers are going to be implemented using a cache like structure. So, the role of branch target buffer is going to supply the address for a predicted branch. If a branch prediction mechanism says that the branch is going to take place, now the branch target buffer is going to supply the predicted address for that particular branch. So, using the predicted address we fetch the instruction and then proceed further.

(Refer Slide Time: 21:53)



So, as I mentioned earlier, it is a cache like structure to store the last seen target address for a given branch. And this branch target buffer is accessed along with accessing the branch prediction unit. We give the branch address as an input to the branch prediction logic. At the same time, we give this as an input to the cache like structure, associated with the branch target buffer, so that we compute these two things simultaneously and we overlap the computation time. And finally, at the end when the branch prediction mechanism makes a prediction that the branch is going to take place, then we will get the target address from the branch target buffer.

So, the organization of the branch target buffer will be something like this. So this is the branch target buffer and it can be organized as a set associative cache and each set consists of a multiple target addresses. By using this branch target buffer, we get set of tags for a selected set, by using this address the branch address and after that each of these tags are compared with the tag of this the branch address. Once there is a match, we are going to consider that as the corresponding target address as the target address for the branch, provided the branch prediction logic is going to predict that this particular branch is going to take place.

The output from the adder is going to give the next address to this branch and the output from this logic is going to give the target address. And the select input for this MUX is the outcome from this branch predictor, if the branch predictor is going to say that the branch is going to take place then the value will be 1. So effectively we are going to take the target

address here. If the outcome from the branch predictor logic is 0, then we are going to take the next instruction in the sequence. So, that will be supplied as the target address and we proceed further with that.

And note that, because this is working like a set associative cache, our branch address is also considered as the set index field, the tag field and so on. So we apply the set index field as an input to the set decoder to identify one particular set and the tag field in this address can be compared with the tags of the selected set. So that we can know whether there is a hit for this particular branch. If there is no match in this particular thing, then we are not going to get the target address, if the branch is predicted to be taken place. In that scenario, we have to wait for the target address to be calculated.

By the way, once a branch outcome is resolved and the target address is computed, we will come back to this branch target buffer and we update the entry accordingly. Because, we know that this branch target buffer is actually storing the last seen target addresses for the branches. So that the next time if the same branch occurs, then we can use this target address if the branch is going to take place. So for that reason, we have to update this branch target buffer after the branch is resolved and the target address is computed.

So, with that I am concluding this discussion on advanced branch prediction mechanisms, as well as our discussion on branch target buffers. And with that we are concluding this module and in the next module, we are going to discuss the dynamic scheduling used to exploit the instruction level parallelism in Superscalar processor designs.

Thank you