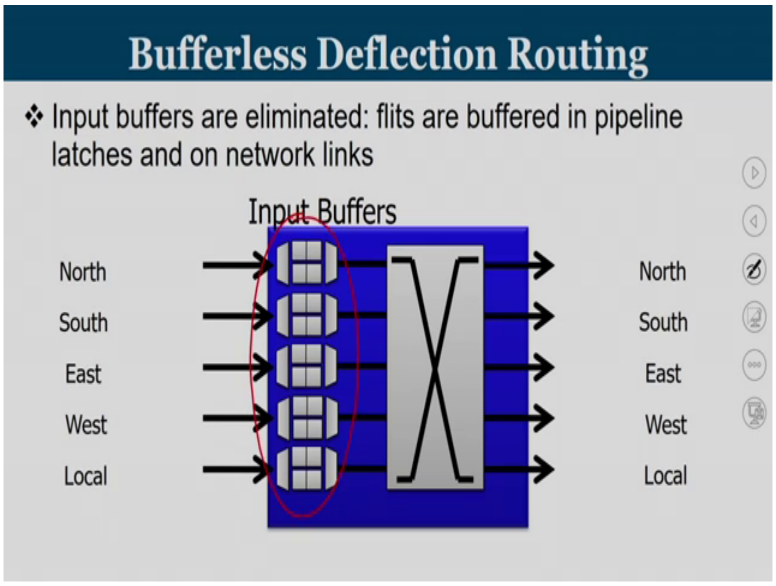
Multi - core Computer Architecture – Storage and Interconnects

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

Lecture – 16
Sidebuffered Deflection Routers

Welcome to the 16th lecture of the course. Today our discussion is on yet another category of energy efficient routers, it is known as Sidebuffered Deflection Routers. In our last lecture, we have seen to make routers more energy efficient, the virtual channels or buffers in the input port are been removed. And we are trying to use a concept called deflection routing, where all the flits are assigned to some ports at the end of the assigned time period, some of them may be productively assigned, and some maybe deflecting away from the destination. Let us, try to revisit some of the concepts what we have seen on the last day.

(Refer Slide Time: 01:16)

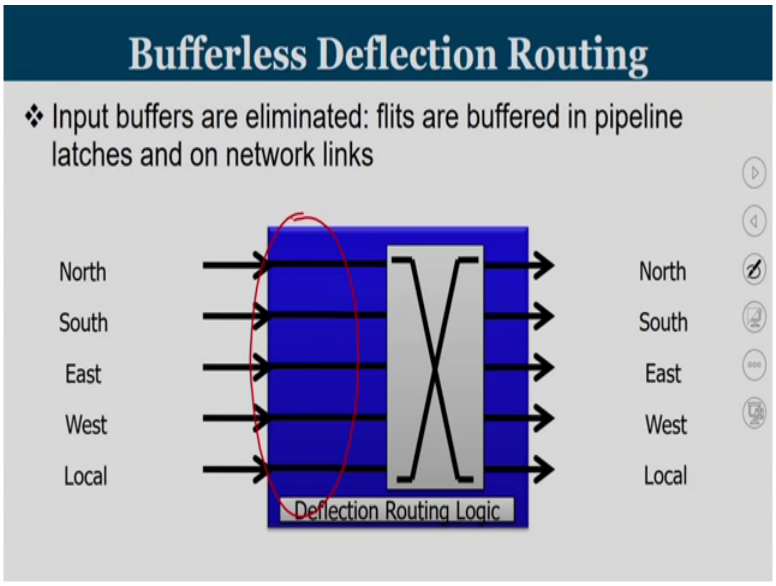


Bufferless Deflection Routing

❖ Input buffers are eliminated: flits are buffered in pipeline latches and on network links

So, input buffers are eliminated and flits are buffered in pipeline latches and network links. So, whatever buffers that you are having in the input side, these were the buffers and that is buffers are been replaced with deflection routing logic.

(Refer Slide Time: 01:28)



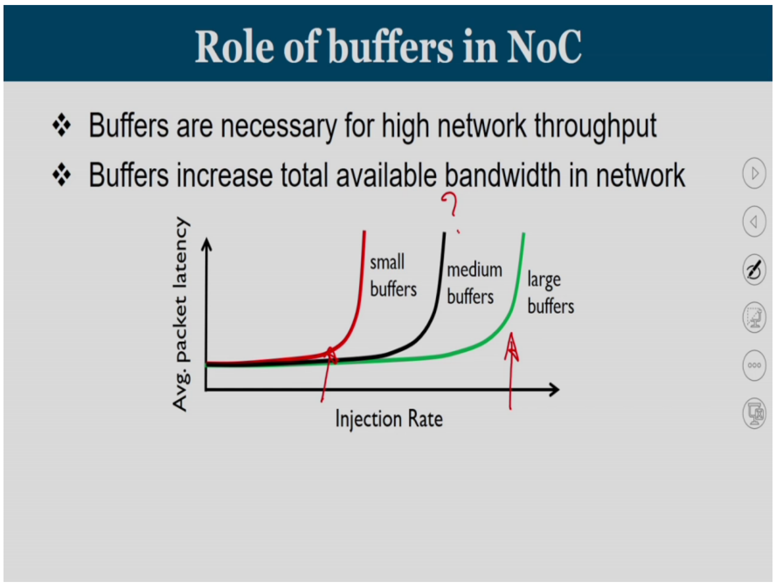
Bufferless Deflection Routing

❖ Input buffers are eliminated: flits are buffered in pipeline latches and on network links

North
South
East
West
Local
Deflection Routing Logic
North
South
East
West
Local

So, all flits coming through the various directions are been assigned some ports and their moving outs. So, input buffers are removed and it will improve your energy efficiency.

(Refer Slide Time: 01:42)



Role of buffers in NoC

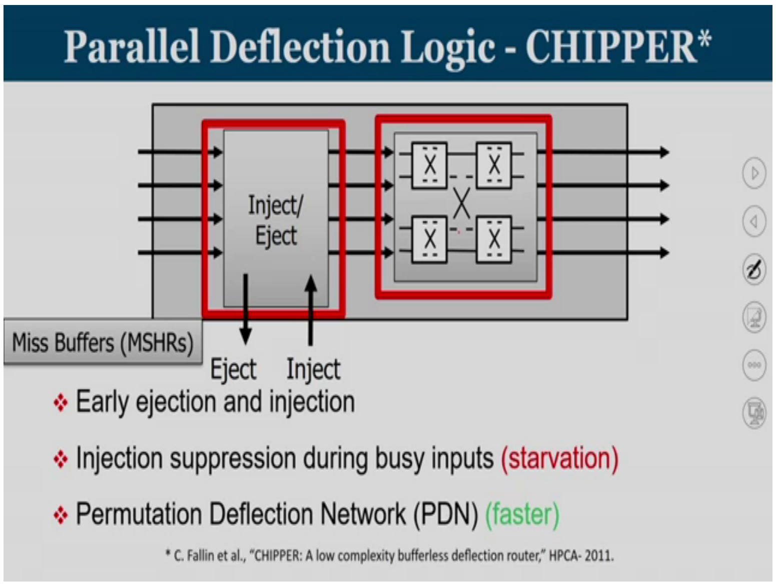
❖ Buffers are necessary for high network throughput

❖ Buffers increase total available bandwidth in network

Avg. packet latency
small buffers
medium buffers
large buffers
Injection Rate

Now, you have seen what was the role of buffers, if you have more number of buffers, then the saturation point is stretched; if you do not have buffers then we are going to saturate early, but you are going to get lot of savings in terms of power and energy dissipation. So, the question is can we have some buffers, let us say we would not keep all the buffers in all the ports, some other kind of mechanism and that is what is known as the concept of side buffering or it is also known as central buffering.

(Refer Slide Time: 02:09)



Parallel Deflection Logic - CHIPPER*

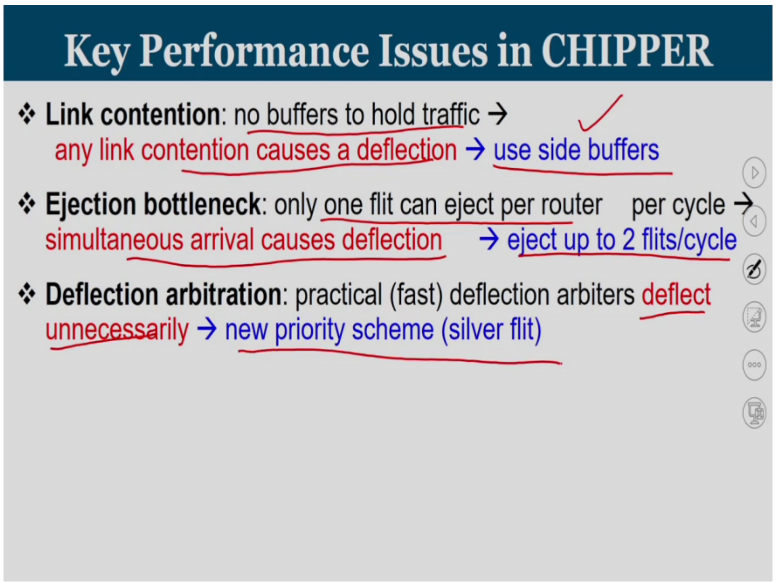
- Early ejection and injection
- Injection suppression during busy inputs (starvation)
- Permutation Deflection Network (PDN) (faster)

* C. Fallin et al., "CHIPPER: A low complexity bufferless deflection router," HPCA- 2011.

Buffers are not available in the input ports, rather we have buffers will be kept at a central location, and after port allocation we see whether should be employ or use this buffers or not. So, this is the parallel deflection logic called chipper what we have seen. Chipper has an early inject and eject stage, early ejection and then whenever there is at least one of the input port is free, we are going to inject.

And then you have a faster parallel port allocation unit which is known as permutation deflection network. So, this chipper was a work there is published in high performance computer architecture conference in 2011 by Chris Fallin and the resource group.

(Refer Slide Time: 02:54)



Key Performance Issues in CHIPPER

- **Link contention**: no buffers to hold traffic → any link contention causes a deflection → use side buffers
- **Ejection bottleneck**: only one flit can eject per router per cycle → simultaneous arrival causes deflection → eject up to 2 flits/cycle
- **Deflection arbitration**: practical (fast) deflection arbiters deflect unnecessarily → new priority scheme (silver flit)

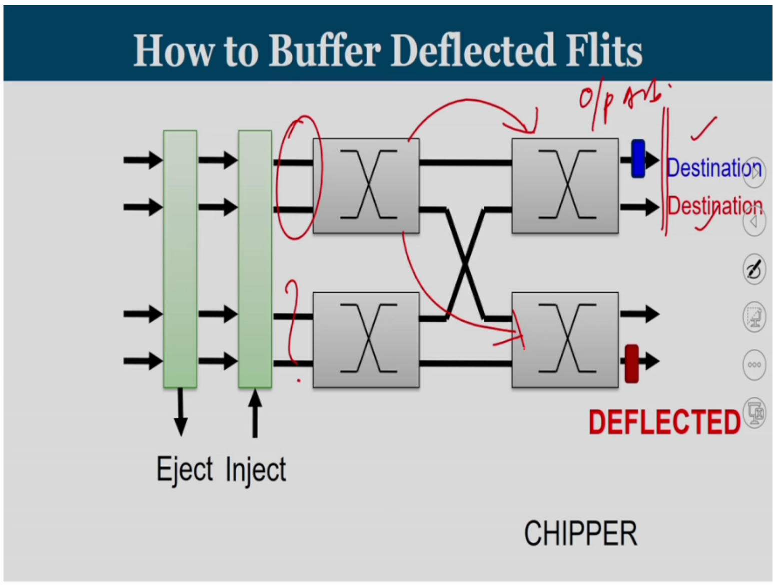
Let us try to see, what are the key performance issues in chipper. First one is link contention, since there are no buffers to hold the traffic whenever there is contention when two packets are looking for the same output port then it may create one of this a scenario, in which one of this packet is deflected away from the destination. So, it is going to get a non-productive port. So, how can that be handled the concept is known as side buffers, all the flits that up to be deflected may be kept in side buffers.

Second one is ejection bottleneck. You can see that in the case of a chipper design, only one flit can be ejected per router per cycle. So, when you have simultaneous arrival of flits that are to be ejected, then it will called to lead to more deflections. The scenario typically happens, when in a given clock cycle you have more than one flits to be ejected. So, since you have only one ejection port and that is an early ejection port. So, can we slightly improve the bandwidth of ejection, eject up to 2 flits per cycle.

And third one deflection arbitration. So, we are using fast deflection arbiter. So, and they are going to deflect unnecessarily, because you have the golden flit concept and only one packet at any given point of time is been termed as golden, all others are not golden. So, can you bring in one more level of priority. So, a new priority mechanism is being brought that is called the silver flit.

So, on top of the conventional chipper architecture, we are trying to introduce three more concepts. The first concept is called rather than permitting all the flits to go in deflection direction, can you save at least a couple of flits or maybe at least one flit per cycle to be accommodated in a special buffer which is known as side buffer. And the second one is called multiple ejection circuitry, and the third one is called introduction of one more level of priorities scheme that is known as the silver flit priority.

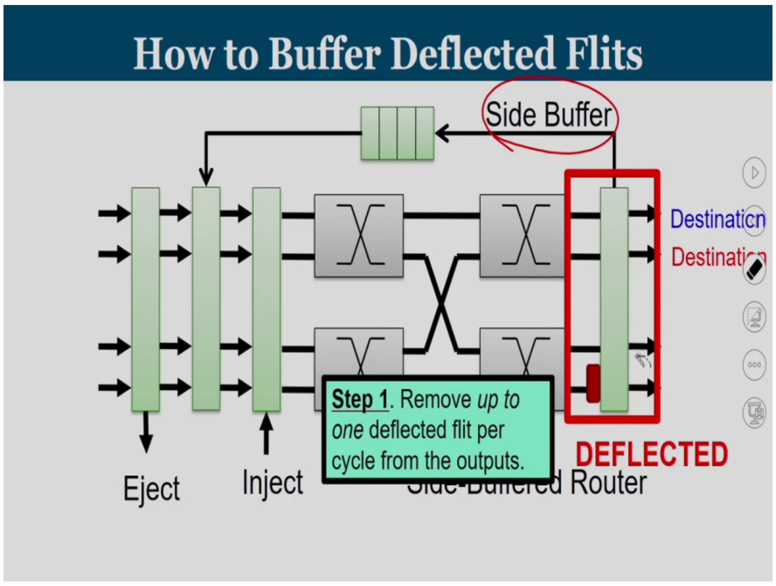
(Refer Slide Time: 05:01)



How to Buffer Deflected Flits

Now, how to buffer deflected flits? So, consider the diagram where in you are having the chipper architecture that is there. Let us say, there are two flits a red flit and the blue flit; the destinations of red and blue is already been shown. Now, assume that the red flit got deflected in the first arbiter and the blue is going to get. So, in this case since both of them wondered on the same output port, we are not able to satisfy them. So, this is the scenario, you have two flits there is no flits in the other two ports. And the destination of this blue and red are connected to the same output arbitrer, this is called as output arbitrer, but because of the structural connection only one of them will be permitted to this arbitrer, and the second one has to be given to the other arbiter. So, this lead to an unnecessary deflection let us say the red is coming down here and the blue is going down there, and unnecessarily the red flit is going to be deflected. So, can you get rid of this scenario?

(Refer Slide Time: 06:10)

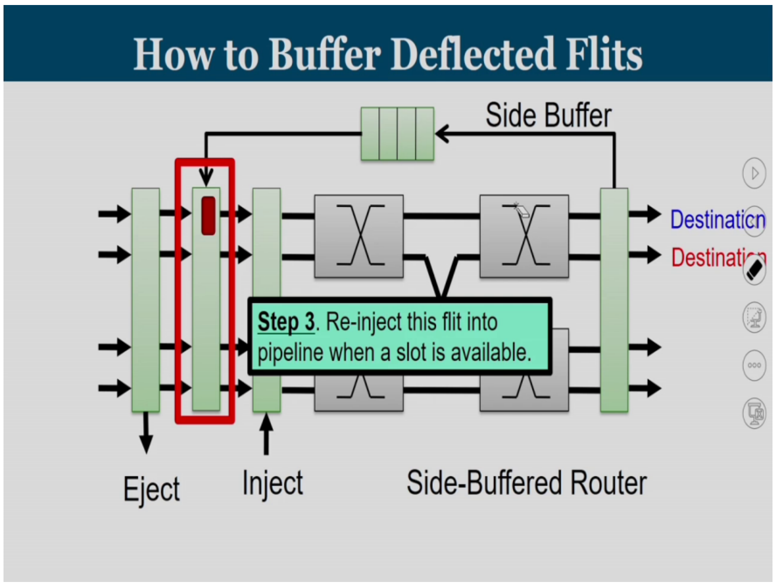


How to Buffer Deflected Flits

Step 1. Remove *up to one* deflected flit per cycle from the outputs.

So, this is something known as the concept of side buffers. There is a unit that is working after the parallel port allocation unit. Now, consider this scenario you have the same two flits the red and blue that is going to come. The red got deflected out, and then you have the blue that is going to get the productive port. So, the blue is moving out; so what happens to the red? The red is deflected. Now, what you do? Remove up to one deflected flit per cycle from the output port. So, this red flit which is already there, that is going to be removed in to the side buffer that is what we are going to see.

(Refer Slide Time: 06:55)



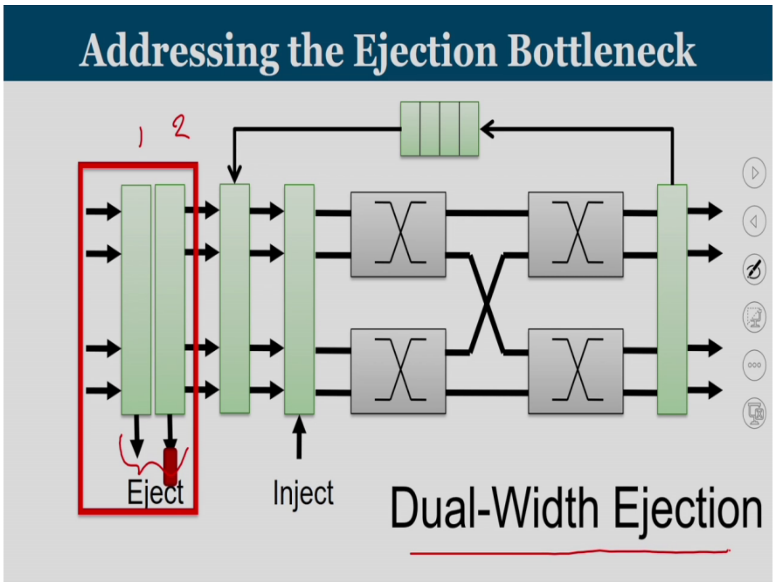
How to Buffer Deflected Flits

Step 3. Re-inject this flit into pipeline when a slot is available.

The red flit is going to be entered into side buffer. So, buffer this flit in a small FIFO like structure which is called side buffer. Then the third step is re-inject this flit into the pipeline when a slot is available. So, the flit will coming to the pipeline in the next cycle, and followed by that we are going to have the flit that is going to move and that may get a productive port at that point of time.

So, this way we are trying to improve upon the performance by reducing the deflection rate. So, the concept of side buffer is at most one flit per cycle is been placed in the side buffer, and these are that flits that did not get the productive output port. So identify one of the deflected flits, place them in the side buffer and permit them to be re-injected into the router pipeline in the adjacent cycles.

(Refer Slide Time: 07:47)



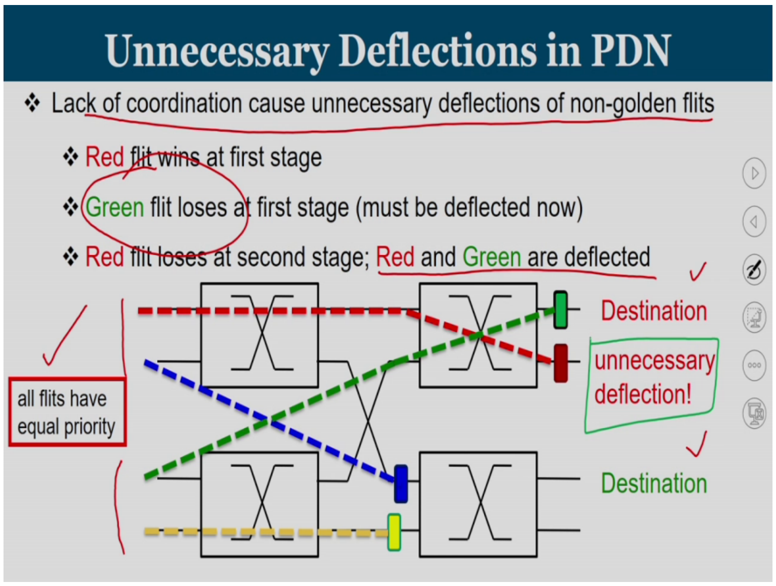
Addressing the Ejection Bottleneck

Now, let us see how are you going to address the ejection bottleneck, the second problem that is associated with the chipper. So, we have already added the side buffer and this is the re-inject unit from the side buffer. Now, consider that let us say you have two flits, both are to be ejected. So, one is going to be ejected like in the normal channel, the other one there is no other way it has to go all the way and it gets deflected.

So, this can be eliminated by a mechanism by which lets say if that particular flit can come and get out in ejection only in the subsequent cycles. So, it has to go around the network come back to the same router, and then get itself ejected out. Can you improve upon the scenario and that is what we are going to see here, we are going to have double ejection bandwidth; whatever is the ejection unit previously we had, you are going to have one more unit after that.

So, the first flit the blue flit gets ejected through the first eject channel, and the red flit get ejected through the second eject channel and both happens in the same clock cycle. So, you are going to have double ejection bandwidth channel. So, addressing the ejection bottleneck is by dual width ejection that is the scenario that we are going to work on.

(Refer Slide Time: 09:01)



Unnecessary Deflections in PDN

- Lack of coordination cause unnecessary deflections of non-golden flits
    - Red flit wins at first stage
    - Green flit loses at first stage (must be deflected now)
    - Red flit loses at second stage; Red and Green are deflected

Now, the third problem that we see in the conventional chipper router architecture is, it there is a lack of coordination that cause unnecessary deflection of non-golden flits. So, we have seen that in the case of chipper architecture, flits are been prioritized by giving a golden status. One of the flits the highly old flits is been given a status known as golden flit, and the golden flit always gets the productive port.

So, if you consider a large NOC setup with multiple routers, only one of the router is going to handle with the golden flit and all the other routers are going to deal with non-golden flits. And regarding the prioritization of non-golden flit, only one flit will be chosen by random. So, in the arbitress that you get one of the flit is chosen by random and that make it a priority.

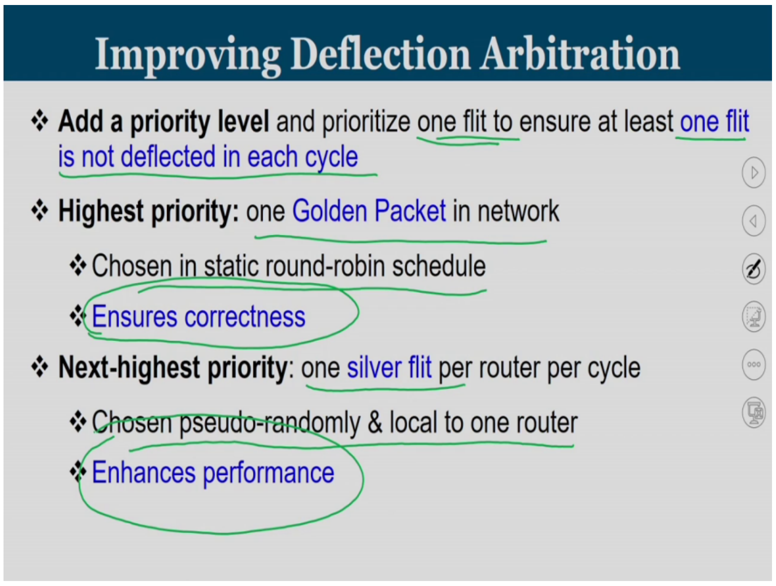
So, the issue is majority of the routers are handling with non-golden flits. So, they need to be some kind of an improvement that can be done on handling with this non-golden flits. So, consider this case let us say you have 4 flits, out of which this green color is the destination for two of the these two which are looking for this destination, and these two flits are looking for the top destination.

Now, let us say the red flit is going to win at the arbitress, because they all are non-golden flits that means, they are of equal priority that is what is been mentioned here. Now, assume that the red flit is going to win at stage one. So, the red flit will proceed into this arbiter automatically the blue has to come down. So, the red flit will go straight and the blue flit has to come down that is what is happening with respect to the first one.

Now, we see what happens in the next two. In this case you assume green flit loses at the first stage. So, green actually wanted to come straight that is going to lose. So, green is going up and then the yellow is going to come straight. Now, you assume that the red flit is going to lose at the second stage, a green which was a loser in the first stage; when it came to the second stage it is been winner, because when you compare to red both green and red are of equal priority.

Let us say by random priority now green is going to get preference over red. So, this will naturally the result in a scenario where, green is been given some port. Now, it so happened that green got the first port and the red got the second one. So, in the case what happens so green is already deflected, and now the red is also getting deflected. And this is been quantified here as an unnecessary deflection, because had red got preference in the second stage also, red would have got its destination; anyway green is deflected.

(Refer Slide Time: 11:56)



Improving Deflection Arbitration

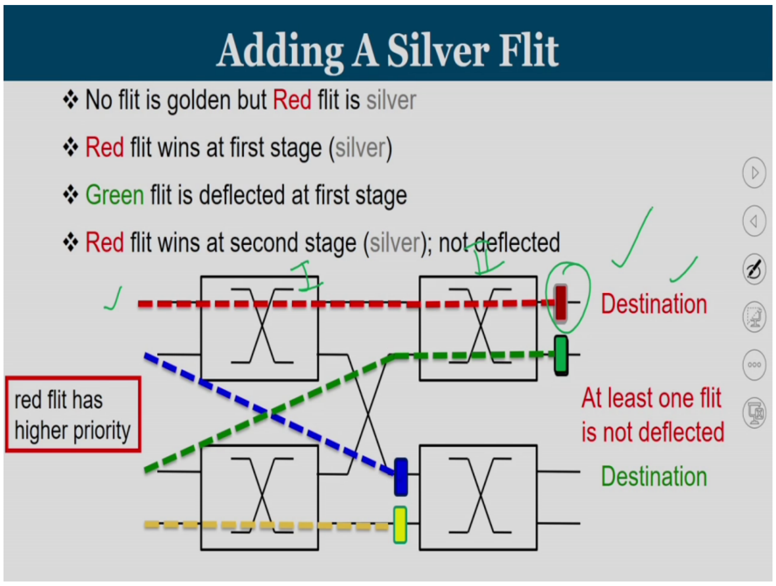
- **Add a priority level** and prioritize one flit to ensure at least one flit is not deflected in each cycle
- **Highest priority:** one Golden Packet in network
  - Chosen in static round-robin schedule
  - Ensures correctness
- **Next-highest priority**: one silver flit per router per cycle
  - Chosen pseudo-randomly & local to one router
  - Enhances performance

Now, how can you improve this kind of a scenario, it is called deflection arbitration. Your adding one more priority level and priorities one flit to ensure that at least one flit is not deflected in each cycle that is going to be the mechanism. So, the highest priority it still holds that is the golden flit packet scheme in the network and that is chosen based upon a static round-robin schedule, one of the flit will be golden, and that ensures the correctness of the algorithm.

The next-highest level is called silver flit. So, what you have you are going to have a pseudo-randomly chosen flit in every router. So, every router we have one of the flit that is been chosen as silver. So, there exist a golden flit in the network, and there exist a silver flit in each of the router. And the silver flit we make sure that whenever you have a scenario where none of the flits are golden flits. Then the silver flit is always going to have the productive port assignment.

So, by this we make sure that the silver flit in that router gets a productive port. So, three important things, first one we are going to add up a new priority level. This golden flits status still holds, but there exists only one golden flit in the entire network. So, majority of the routers will be dealing with non-golden flits. So, when you have the second level like when you have all the flits non golden, then you are going to consider one of them a silver flit and that is the way how you are going to improve the performance. So, golden flits make sure that it is free from live lock. Every flit will become one time golden they are not getting ejected out. And second one to silver flit is going to enhance your performance.

(Refer Slide Time: 13:48)



Adding A Silver Flit

- ❖ No flit is golden but Red flit is silver
- ❖ Red flit wins at first stage (silver)
- ❖ Green flit is deflected at first stage
- ❖ Red flit wins at second stage (silver); not deflected

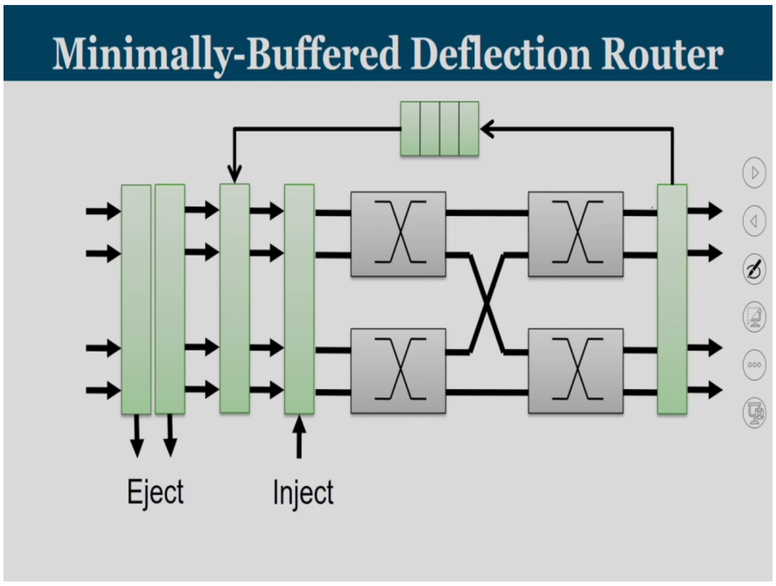
red flit has higher priority
Destination
At least one flit is not deflected
Destination
I
II

So, how are you going to add up a silver flit? So, consider that you are going to make one of the flit as silver flit. The red flit is being considered a silver. So, in this scenario none of the flits are golden. The red flit is been given a higher priority and that is a local thing whenever this flits reaches here one of the flit is been chosen by random. In this case the red flit is chosen by random, the red flit is known as the silver flit. Now, the red flit wants this of the destination. So, the red flit wins at the first stage, because it is silver, so the red will go straight the blue will come down.

Let us say the green flit is deflected at the very first stage just like what we have seen in the previous example. So, the green is deflected away giving way the yellow is given productive port, and the green is going to come out. Now, the third stage you know since the red flit is silver, it is going to have a highest priority in both the arbitress stages. Both in stage one and in stage two the silver flit is going to win.

So, the red flit at the second stage also it is going to get winner, and thereby it makes sure that the red flit is winning. So, the idea of silver flit is in both the stages of the arbiter the silver flit will gets highest priority thereby you make sure that the silver flits gets a productive port it will never be deflected.

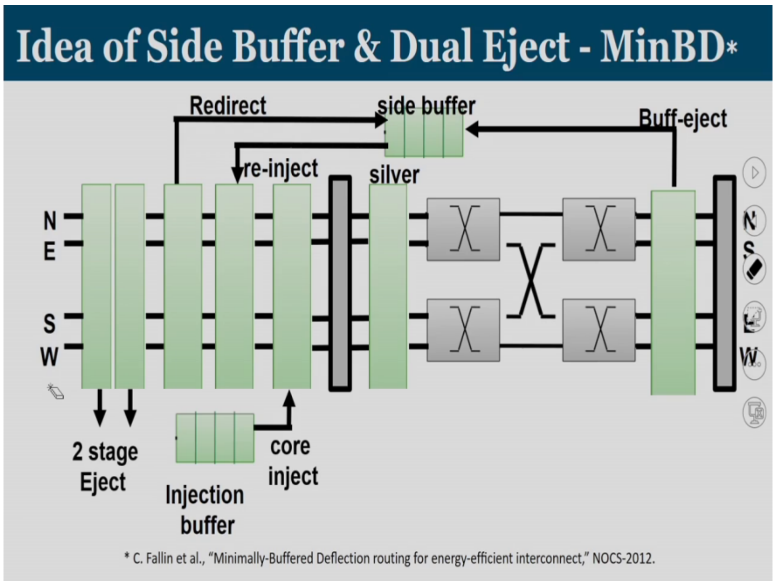
(Refer Slide Time: 15:09)



Minimally-Buffered Deflection Router

Now, this is called the concept of minimally buffered deflection router. So, whenever you have link contention then the solution is side buffer. Second one the ejection bottleneck is handled by 2 ejection stages. Third one we are going to have a two-level priority scheme by introducing a silver flit concept. And silver flit always get a productive port in the current router.

(Refer Slide Time: 15:36)



Idea of Side Buffer & Dual Eject - MinBD*

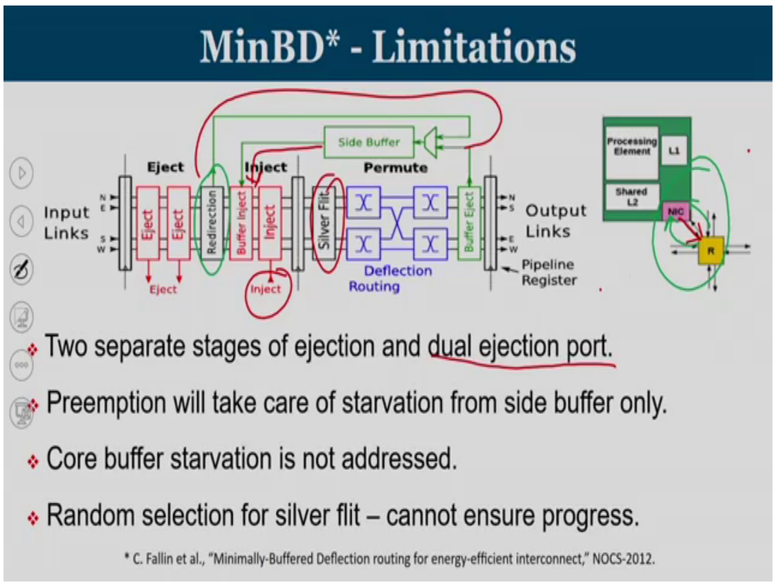
* C. Fallin et al., "Minimally-Buffered Deflection routing for energy-efficient interconnect," NOCS-2012.

So, the idea of side buffer and dual eject min B D that is published by Chris Fallin, and his colleagues, it was published in the NOCS network on chip symposium conference in 2012.

So, this is the whole idea of minimally buffered deflection router. So, we can see that it has 2 stages of eject. It has a silver flit module, it has a side buffer, and to take flit into the side buffer. We have one unit which is called the buffer eject unit. And we have one unit which will take care of the redirection of flits into the router pipeline.

So, this is the way how the entire min BD router is going to works. On top of the conventional chipper router there are few more architectural enhancements that are added. Basically, the first one is you are going to have dual ejection unit, second one is you are having a side buffer, and you need to feed data into side buffer, and you need to take data from the side buffer. And the third one is the silver flit priority mechanism.

(Refer Slide Time: 16:38)



MinBD* - Limitations

- Two separate stages of ejection and dual ejection port.
- Preemption will take care of starvation from side buffer only.
- Core buffer starvation is not addressed.
- Random selection for silver flit – cannot ensure progress.

* C. Fallin et al., "Minimally-Buffered Deflection routing for energy-efficient interconnect," NOCS-2012.

Now, let us try to see what are the limitations of this Min BD architecture. This is the conventional diagram of Min BD. So, here you have two separate stages of ejection. So, what do you mean by two separate stages of ejection and that will lead to dual ejection port. What you see on the right side slide is the way how a router is interface into a tile. So, dual ejection port means we have two ports to connect from the router, and one port to connect from the tile to the router.

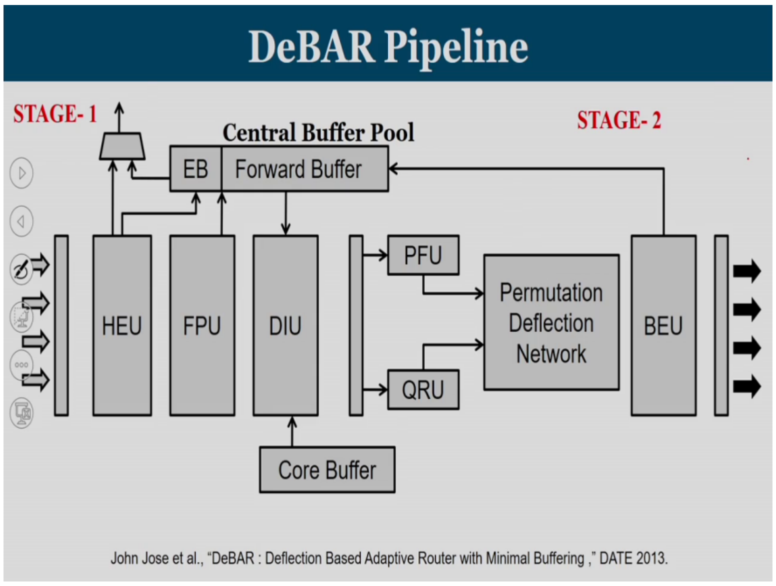
So, how are we going to draw it let us say we have one port through which the injection happens, and we have two ports by which the router is going to give back data into this. So, this is going to increase the bandwidth of ejection a natural, it is going to be a little bit costly, and space, and power consumption.

Second one the concept of preemption will take care of starvation from the side buffer only. So, we are having a circuit, which is taking care of the redirection aspect. So, this redirection aspect is defined as whenever flits are there in the side buffer waiting in the side buffer, but if all the input channels are busy, then that will lead to starvation.

So, this is the process by which one of the flit is been taken, and that flit is been added into the side buffer. And this process is known as redirection. So, this preemption process will take care of starvation from the side buffer only. The flits that are waiting in the core buffer that is newly created flits there starvation is not addressed. And the last problem with respect to Min BD is silver flit is a strictly local selection.

So, a flit that is silver in the current router may get a productive port, but once it reaches the adjacent router then it may not be a silver you would be getting a silver flit there. So, there is no guarantee that a flit which is silver in one router will get the silver status in the adjacent router. So, in the current router it may move closer to destination. In the adjacent router it may not be silver leading to a deflection, then it may again go back so that will lead to kind of exact moment it cannot guarantee progress. So, these limitations where addressed by the next router in the series that is known as debar deflection based adaptive router.

(Refer Slide Time: 19:15)



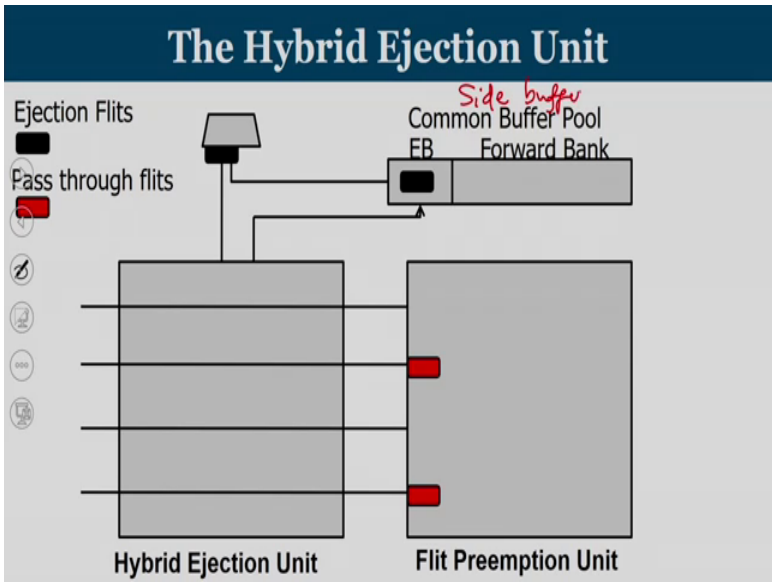
DeBAR Pipeline

John Jose et al., "DeBAR : Deflection Based Adaptive Router with Minimal Buffering ," DATE 2013.

This is also a two stage router that stage one consisting of hybrid ejection unit, flit preemption unit, and dual injection unit. Three units are there in the stage 1. And stage 2 there is a priority fixer, there is a quadrant routing unit, and as usual our permutation deflection network, which is a parallel port allocation unit and the buffer ejection unit. So, these are the two stages, and the various functional units of the new router DeBAR.

Now, see stage 1 and stage 2 we will try to see, so this work was published by our research group published in the date 2013 conference. Now, we will try to see what are the functionalities of this one of the main issue with respect to Min BD was since it is having dual ejection port. It will take care of multiple flits that are to be ejected, but how many of such kind of scenario we get multiple ejections. So, it was found that roughly only 8 percent of the cases we get multiple flits to be ejected and that to at high traffic.

(Refer Slide Time: 20:32)

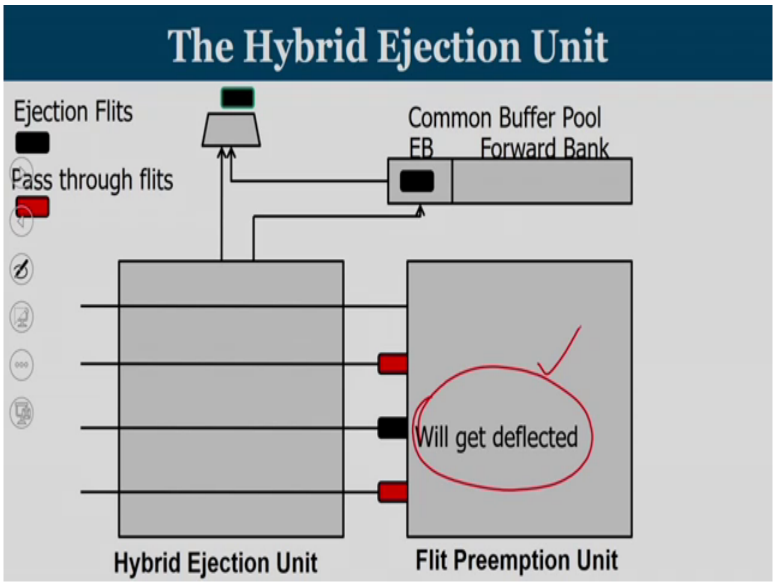


So, this proposal is going to talk about a hybrid ejection unit. So, this is the zoomed version of a hybrid ejection unit and the flit preemption unit. So, these two units what you see on the left hand side of the pipeline let us say hybrid ejection unit and the flit preemption unit have to be discussed in detail. So, the next slide will show hybrid ejection unit towards your left side, and the flit preemption unit towards your right side.

So, the duty of hybrid ejection unit is to analyze whether a flit is to be ejected or not whether they are destined to the local core or not. Let us say how the hybrid ejection unit works, consider the scenario where the color coding tells that the black flits are the ejection flits, and the red flits will pass through the router to its neighbors. So, this is been shown with the help of a animation.

We can just listen to this animation. When the flits come to the hybrid ejection unit, the combination circuit there the intelligence that is embedded in the hybrid ejection unit, we will check whether the flit is to be removed to the local port or not. So, when the flits will come, it is been removed to the output port, so the flits will move out through this port. Now, consider a scenario we get two flits to be ejected, we can see that there are two flits coming from the input port both are to be ejected, because of the black color coding. In our convention this black color indicates they are flits to be ejected. And there are two flits they are the red flits.

(Refer Slide Time: 22:54)



Now, you can see that one of the flit will move out through the ejection port and the other flit get occupied in the in the buffer pool. This is also known as the side buffer. So, this common buffer pool can also be defined as side buffer. Now when you have a scenario like this, where already you have a flit, which came in the last clock cycle is waiting in the ejection bank of the side buffer. And now in this cycle you are getting one more flit. And let us say this flit is to be ejected, so that flit will go to the ejection bank, and the one that was previously there that will go out through the ejection port.

Now, you have one more scenario, where you have a flit that is already there in the ejection port, and you get two flits to be ejected. So, in this case one of them will move to ejection port and the other one we will surely be get deflected. So, this is will happen we cannot do anything. And we have found that such a kind of a scenario will happen very rarely less than

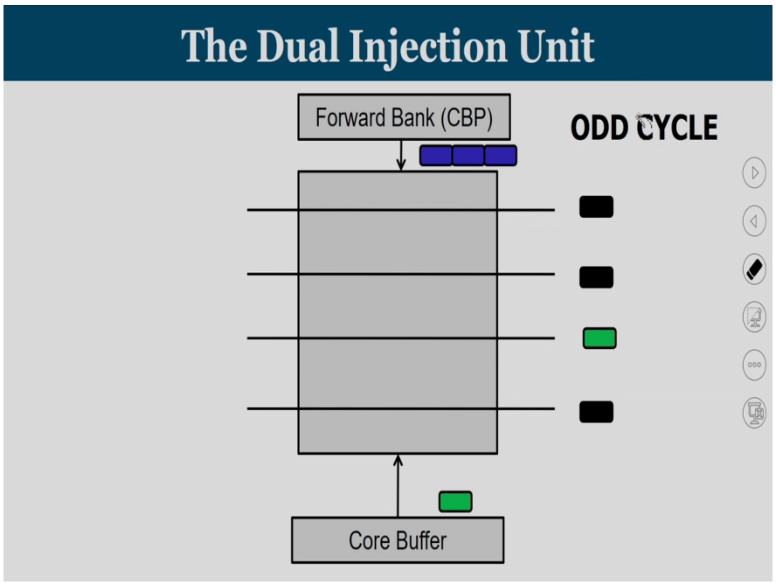
0.025 cases, because in two consecutive cycles we should get flits that are destined for the ejection core.

So, this is how the hybrid ejection unit works we have only one ejection port, but it could potentially handle two flit removes on an average to the ports, but with one port we are actually managing it. Now, let us try to understand, what is the role of dual injection unit. One of the problem with respect to Min BD design was when the flit that is there in the side buffer is waiting to enter into the router pipeline, it may suffer from the problem called starvation.

So, what is essentially starvation, starvation happens when you have a flit it is waiting in the front of the side buffer to enter into the router pipeline, but all the inputs of the router are carrying valid flits. So, the input ports are not idle. So, there is no vacancy inside the internal channels in the router. We have seen that one of the main principle for injection in the case of a deflection router is, injection is possible only if one of the internal flit channel is empty, so starvation occur when all the flits channels are busy with flits coming from the neighbor.

So, we have introduced like in the case of a Min BD they have introduced a redirection circuit, where a flit is forcefully removed to the side buffer giving way that is which will create a slot in the routers internal channel, but that will take care of starvation only from the side buffer. Starvation of the core buffer was not address, because core buffer was coming later in the pipeline in the case of a Min BD design. So, in this case in the case of DeBAR both the side buffer entry as well as the core buffer entry, are happening in the same unit.

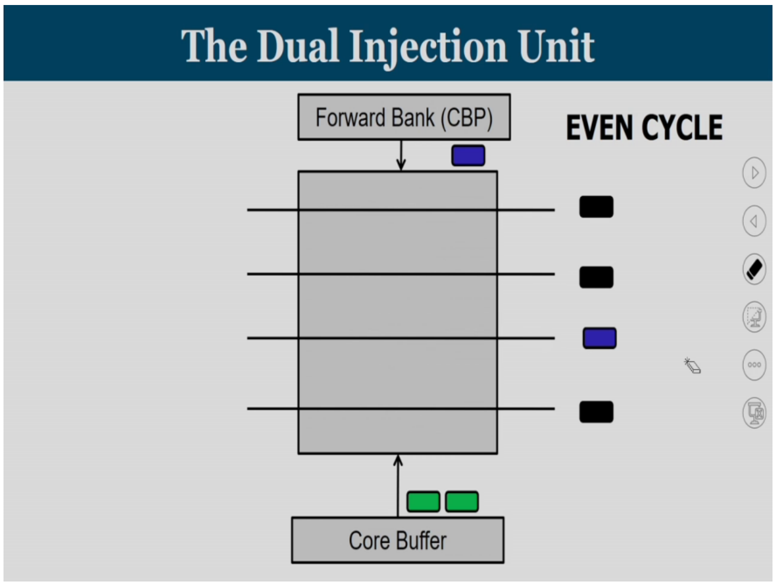
(Refer Slide Time: 25:08)



And they are working on an odd even based priority. So, flits from the side buffer that is a forward bank common buffer the central buffer pool, and from the core buffer. They are going to entered them based upon an odd even priority. So, we can see that in odd clock cycles preference is given to the flits from core buffer.

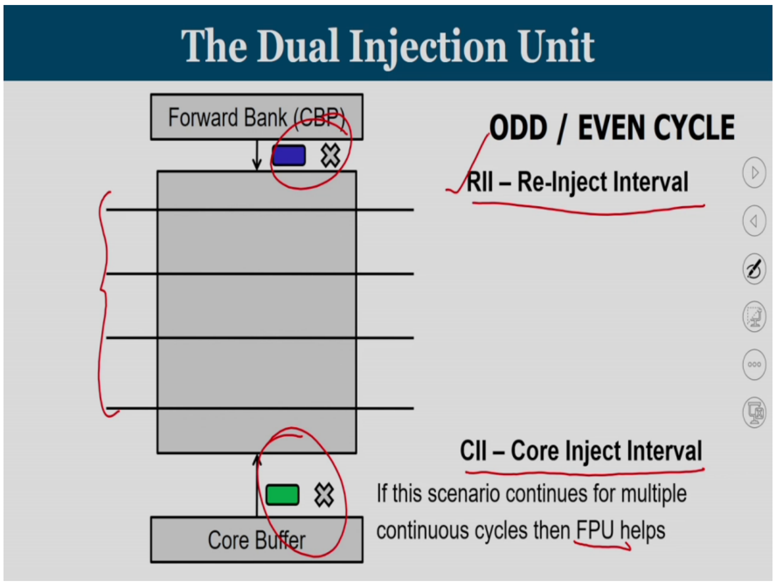
You can see that in all odd cycles can you look at the animations once again. You have a one empty slot. Since, there is an empty slot a flit can be injected. So, if it is an odd cycle, flit will be injected from the core buffer. So, the green flit that is there in the core buffer will enter, and it will move on.

(Refer Slide Time: 25:58)


The Dual Injection Unit

Now, we will try to see what happens in an even cycle. Let us say if it is an even cycle, then the flit from the central buffer pool that is the blue flit it will occupy the vacancy. So, the blue fit will enter and then it pass.

(Refer Slide Time: 26:11)


The Dual Injection Unit

Forward Bank (CBP)
ODD / EVEN CYCLE
RII – Re-Inject Interval
CII – Core Inject Interval
If this scenario continues for multiple continuous cycles then FPU helps
Core Buffer

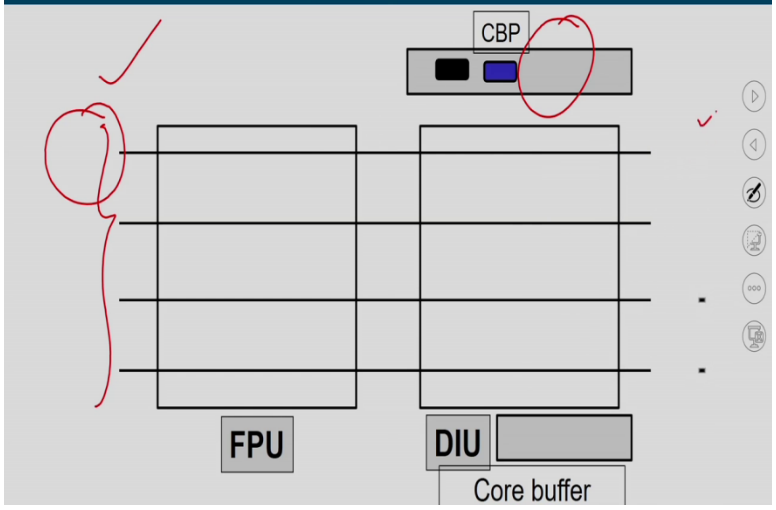
Now, if there are two vacancies, now in this case whether it is an odd cycle or an even cycle, if there are two empty slots, then I can inject from both of the circuits that is how it is going to work. So, this is called the principal of dual injection unit. Now, there is one problem. Let us say odd four are busy odd four internal flit channels are busy.

Then the flits that are waiting in the central buffer pool as well as in the core buffer cannot inject. So, they cannot inject that the scenario. This will lead to starvation, but this scenario is going to continue for multiple clock cycles. Then the flits that is waiting in the core buffer as well as in the side buffer is going to face the problem of starvation. And this can be addressed by the unit which is called the flit preemption unit.

So, we defined 2 components one is called the re-inject interval, other one is called core inject interval. Re-inject interval is defined as the number of cycles of at most starvation can happen. So, the re-inject interval is equal to 2 that means a flit in the side buffer will wait at most 2 cycles. If for two cycles if it is not getting, an entry into the router pipeline, then preemption happens leading to away for this out of flits. Similarly, we have something called core inject interval. Core inject interval is a scenario, where how long or it is defined as a threshold for how long a flit in the core buffer will wait to enter into the router pipeline so that is all about this.

(Refer Slide Time: 27:54)

Flit Preemption Unit



And now we will say about the flit preemption unit. So, what you can see that when a flit is moving, and that this is going to create starvation. So, the next time when a flit is moving, it is been removed and then you are able to get. So, in this scenario, the second scenario we can see that we have four flits that is coming, since the channel is busy and the flits that is waiting in the central buffer pool is starving.

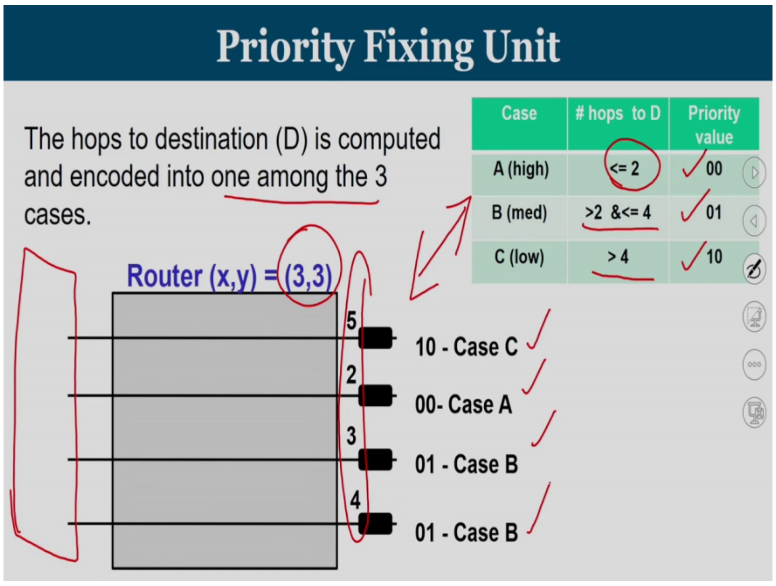
So, one of the flit will enter into the router pipeline whereas the other is going to be removed. So, this flit is going to be preempted so that flit is going to be preempted out into the central buffer pool in the next cycle that is what we are going to see that the flit is been removed. And the red one will enter, and they are going to work like that.

Now, the next one we will see about the stage two of the DeBAR pipeline. So, in the stage two we have a priority fixing unit, a quadrant routing unit plus buffer ejects unit and permutation deflection network. So, first we try to understand what is our priority fixing unit. So, in the case of a chipper the priority was based upon age we call it as a golden packet.

In the case of min b d one more level of priority was introduced. So, we have a golden packet as well as a silver flit. Silver flit selection was completely random. So, a flit that is becoming silver in one router may not become silver in the adjacent router. This lead a scenario that a flit will move closer to destination in one router, and due to adverse priority mechanism it may move away from the destination leading to a zigzag moment or it is just having experiencing some kind of a live lock.

So, a new kind of a priority mechanism is introduced the heart of the priority mechanism is when a flit has almost reached close to destination, let us give a priority to it in such a way that it is not deflecting away. So, this can be implemented by looking into the hops to destination. Every flits that is reaching the router. They are been graded or compartment aliased or partitioned based upon the hops to destination.

(Refer Slide Time: 30:05)



Priority Fixing Unit

The hops to destination (D) is computed and encoded into one among the 3 cases.

Case	# hops to D	Priority value
A (high)	<= 2	00
B (med)	>2 &<= 4	01
C (low)	> 4	10

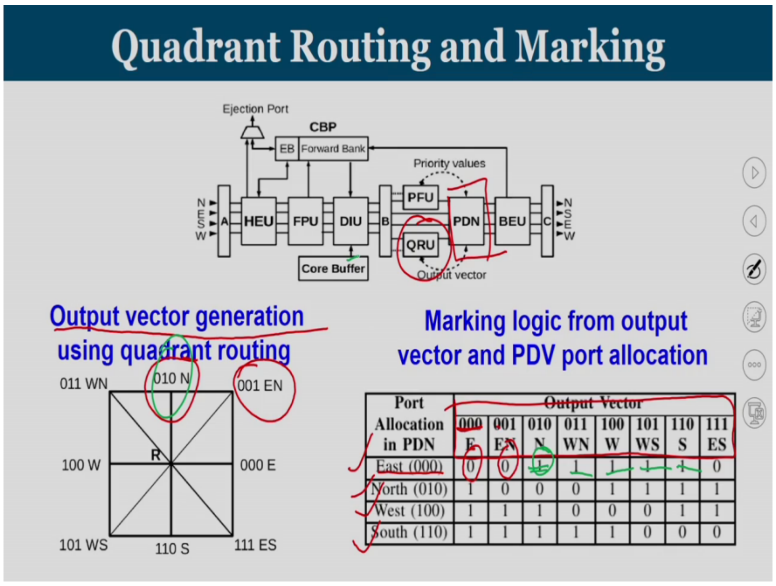
So, the hops D to destination is computed an encoded in 3 different cases. So, consider that you are working on an 8 by 8 mesh n o c, where you have 64 routers. And each router is been given a naming X and Y. So, consider the case you are talking about a router 3, 3, 3rd column and 3rd row in the network. Consider there are four flits what is marked on the flit is the destination number. So, the first flit wanted to go to 4, 7 second one wanted to go to 4,4 third one wanted to go for 0,3 and the fourth one wanted to go to 1,1.

So, these four flits are going to reach a router whose coordinate is 3, 3. So, the hops to destination is computed by Euclidean distance [vocalized -noise]. So, 4, 7 where exist 1 difference. So, from 3, 3 to reach 4, 7 you have to move 1 jump 1 hop in the X dimensions. And 3, 4 hops in the Y dimensions. So, totally it is 5 hops. Similarly, 4, 4 will have total of 1 plus 1, 2 hops to make 0, 3 has to make 3 more hops 1,1 has to make four more hops.

So, based upon the number of hops to destination it your flit is been given a priority value into one among these three cases. If the number of hops is less than or equal to 2, the priority is given as 00. If the number of hops is greater than 2, but less than or equal to 4, the priority is given as 01. If it is larger than 4, the priority is given as 10. So, for every incoming flit we are going to look at what is the priority value, and this priority value is been used as the key in order to give preference in PDN. So, you have four flits that is coming these four flits upon passing through the router.

You are going to find out the number of hops. So, what you see here is a number of hops. And upon referring into the table these flits are categorized as category C A B and all. So, case A is the highest priority, case B is the medium priority, and case C is the lowest priority. So, whenever the flits are coming into the priority fixing unit from the destination address that is stored on the flit. The flits number the destination number is extracted, and hops to destination is computed, and the priority value to be priority value is assigned to them.

(Refer Slide Time: 32:44)



Quadrant Routing and Marking

Output vector generation using quadrant routing

Marking logic from output vector and PDV port allocation

Port Allocation in PDN	Output Vector							
	000 E	001 EN	010 N	011 WN	100 W	101 WS	110 S	111 ES
East (000)	0	0	1	1	1	1	1	0
North (010)	1	0	0	0	1	1	1	1
West (100)	1	1	1	0	0	0	1	1
South (110)	1	1	1	1	1	0	0	0

The next concept is known as quadrant routing and marking. So, we have one more unit which is called Q R U quadrant routing unit. So, the idea of quadrant routing unit is we will find out a vector a direction for a flit and that direction has been assigned with a code. And this binary code will tell; which are the possible directions, which will take the flit closer to destination and which are the possible direction, which will take the flit away from the destination. So, consider the left hand side what it is been given, let us say this is the current router that is see center of the diagram. Let us say the packet is here.

The destination can be in any of these 8 possible cases. It can be in the east north side of the current router it can be strictly on the North side of the current router. So, strictly on the North side means the destination is exactly on the same column as that of the current router. It can be strictly on the South side that also the destination is exactly on the same column, but it is towards south direction.

Destination can be strictly on the East or West side in that case there in the same row. So, we have 8 different possibilities for an incoming packet. So, let us say in the current router you got a packet. The packet will belong to 8 of these quadrants. It can be East, East North, North, North West, West, then South West, South and then in South East. These are the directions potential directions that have shown here.

So, based upon the relative position of the destination router with respect to current router a 3 bit output vector is generated. So, what this 3 bit that you see is the output vector, and this routing is known as quadrant routing. So, the output vector is generated. Now, there is a marking logic. This output vector is been given to the permutation deflection network. And this PDN is going to have a tabled structure inside that and from the table a lookup is made.

So, PDN can assign one of the four ports that is East North West and South. These are the four possible ports that a flit can get by passing through the permutation deflection network. Now, what you see here is the output vector of a flit. If the output vector is 000, and the port that the PDN gave is also 000 then the marking is 0 that means the flit is in the productive direction. I will take one more case if the output vector is 001 output vector is 001 means, the flit is relatively in East North direction with respect to current router.

Let us say if East is support allocated, so a flit got East direction the destination is at relatively East North direction with respect to current router. The flit is in the productive path, because East North a flit can flit, which got East output port is still travelling in the East North quadrant that is way we can see that marking bit is 0 means the flit is in the productive port.

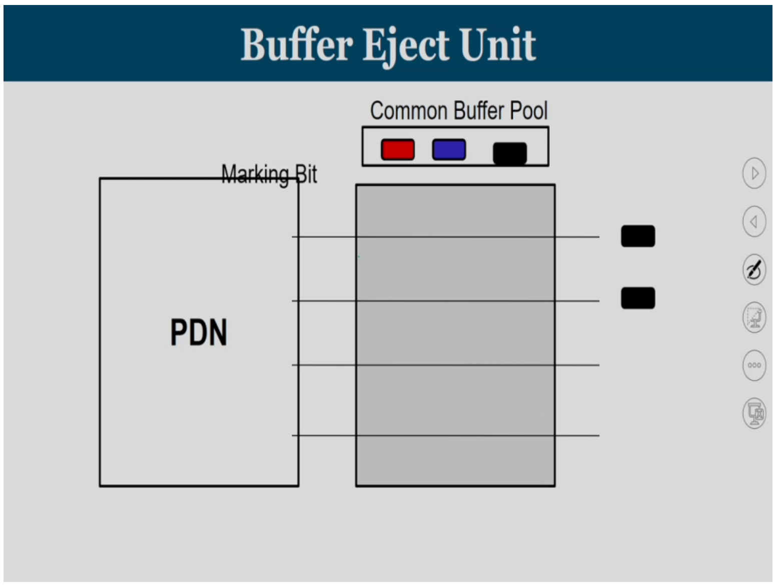
Now, the third scenario let us assume a flits output vector is 010010 means this is the output vector. When a flits output vector is 010 that means destination is exactly in the North of the flit. Now, the PDN is allocating 000. So, when the output port assigned to a flit is East, but its quadrant vector is 010 that means it needed exactly North 010 means exactly North is what I am looking for, but what I got is East so in that case the flit is not in productive direction that is why it is marked as 1.

So, any 1 in this table indicates of there are many 1 you can just find it out from this table. What are all the different combinations where we should get 1 and 0? So, any 1 that you get then, that means that the flit is going in a known productive direction. So, this will in turn help the next unit to take up a good decision. So, the idea of marking is basically to tag a flit whether it is marked or non-marked. Marked means if a flit is marked means, it is going in a nonproductive direction.

So, it is a potential candidate to be side buffer. So, the quadrant routing and the marking together will help the buffer eject unit to take up a quad. So, quadrant vector is nothing but it is a 3-bit value that indicates what is the relative position of the destination, with respect to the current router. The destination can be either on the axis exactly on the East, West, North or South or it can be in a quadrant where there are 2 possible productive ports to reach that router.

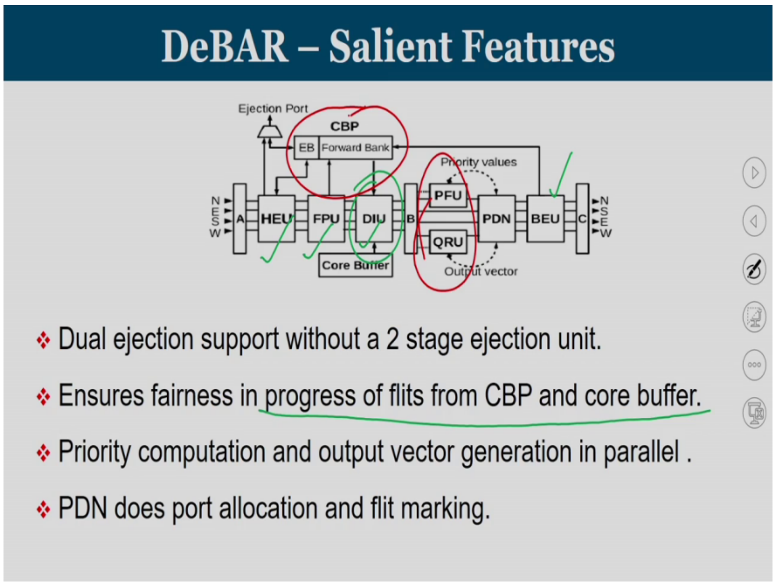
And based upon the output vector, and the output that is allocated there is a marking scheme, where 1 indicates a flit is marked with 1 that means flit is not productively assigned. It is deflecting away. If it is marked with 0 means nothing should be done even though, the flit is going in same direction; it is still one hop closer to the destination.

(Refer Slide Time: 38:44)



Now, let us consider the last unit that is buffer eject unit. Now in this buffer eject unit the PDN s output is going to the buffer eject unit, which is interconnected to the common buffer pool. The PDN is going to perform marking that we have seen just previously. If the marking is there, then the buffer eject unit job is easy the flit is removed to the destination. So, the marking bit is 0 the flit will directly pass through, the marking bit is 1 the flit is been taken to the common buffer pool that is a way how the whole idea of buffer eject unit will work.

(Refer Slide Time: 39:16)


Ejection Port
CBP
EB
Forward Bank
Priority values
PFU
N
E
S
W
A
HEU
FPU
DIU
B
QRU
PDN
BEU
C
N
S
E
W
Core Buffer
Output vector

DeBAR – Salient Features

- Dual ejection support without a 2 stage ejection unit.
- Ensures fairness in progress of flits from CBP and core buffer.
- Priority computation and output vector generation in parallel .
- PDN does port allocation and flit marking.

So, that completes the various stages of the DeBAR. It had a hybrid ejection unit which will take care of 2 ejections without having 2 ejection ports. You have a flit preemption unit that takes care of the starvation of both side buffer and the core buffer. And then we have a dual injection unit where flits can enter from both the side buffer and core buffer with odd even priority based mechanism.

And then you have a priority mechanism is just not working based upon random type silver or golden based on age. Priority is strictly based on hops to destination whichever flit is nearing the destination they may get highest priority. So, you may initially deflect when you are away from destination as you become closer to the destination your priority is going to improve. And then you have a quadrant routing unit, which will tell you what are the productive directions and a PDN as usual followed by a buffer eject unit.

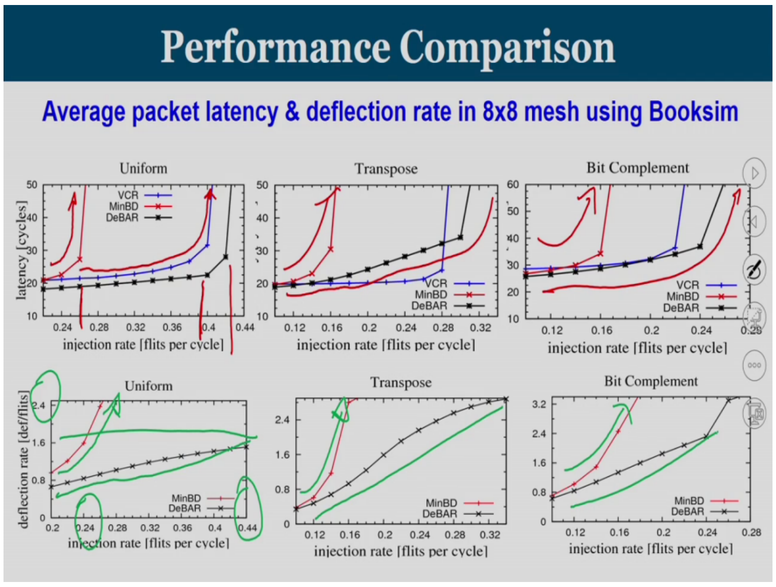
So, the dual injects dual ejection support without two stages of ejection. Still we can manage to ejection with the help of a side buffer and to ensure fairness in the progress of flits from the central buffer pool and the core buffer. So, since we are having a dual injection unit that will take care of progress of flits from both side buffer as well the core buffer.

And then the priority computation and the output vector computation. So, these two things are happening parallely since these two things happened parallely it is going to save little bit of time. And the PDN port allocation is done with the help of a flit marking scheme and that is going to help this overall port allocation.

One more salient feature of dEBAR was when you consider a mesh NOC, the central routers typically carry more traffic whereas; the edge and corner routers are going to carry less amount of traffic. So, the number of flits buffer space in the so called side buffer whatever side buffer that you see here.

It is typically defined as four flits buffer space in the case of min B D. In the DeBAR they are using a non uniform side buffer allocation. The central routers are having four flit side buffer the edges are having 3 flit side buffer. And the corners are having 2 flits side buffer that also is going to save little bit of your space and energy budget.

(Refer Slide Time: 41:46)



Performance Comparison

Average packet latency & deflection rate in 8x8 mesh using Booksim

Now, let us try to see the performance comparison. There are two important parameters that is being used here, one is the average packet latency other one is deflection rate. And these are been taken from a simulator open source simulator known as Booksim. So, there are various synthetic traffic patterns, which is been defined in the graph at uniform traffic, transpose traffic, and bit complement traffic. These are all artificially traffic generation models, where certain nodes are going to create packet to certain other nodes.

What you see on all these graph is on the x axis, it is called packet injection rate. So, if packet injection rate is equal to one that means, at every cycle, every router is injecting a new packet. So, if packet injection rate is equal to 0.5, then at every cycle only 50 percent of the routers are going to inject packets or a router is going to inject packet once in two cycles.

So, across various traffic you can see that the packet injection rate is increasing. As the packet injection rate is increasing, you can see the graph the y axis is the average packet latency. How much is the approximate number of cycles, a packet is going to take in the network. So, the packets are generated in the network, we are going to travel through the network, and how much time approximately a packet will take. So, it is typically known as load versus latency graph or injection rate to versus latency graph.

Now, little bit of the result analysis that is been given here, it is a comparison between different techniques. You take this first graph on the left side, this red color is called MinBD. So, as the load increases the latency of packets, which are using MinBD traffic is on the increasing consult latency increases to high value somewhere around this point. And the blue one is a normal input buffer router. Input buffer router even if you increase the load still it is low and it goes to high latency roughly around the stage. And the black one is DeBAR routers. So, DeBAR can accommodate more number of packets in the network still it is not saturating.

So, this is something like a load latency graph, we have seen. If you have less number of buffers, if there is not deflection, then it is saturated early that is what you can see in the MinBD. So, the trend is there MinBD saturate early in transpose as well as bit complement traffic. And we can see that this DeBAR that is the black one is showing, slightly better performance in the other two traffic. Yet another parameter that is to be used for analysis of deflection routers is a deflection rate. So, when a flit is not getting productive port, the flit gets deflected away. It is going to other unwanted ports, so that will increase the number of hops of the flit. So, the more number of hops its travel latency of the flit is also going to increase.

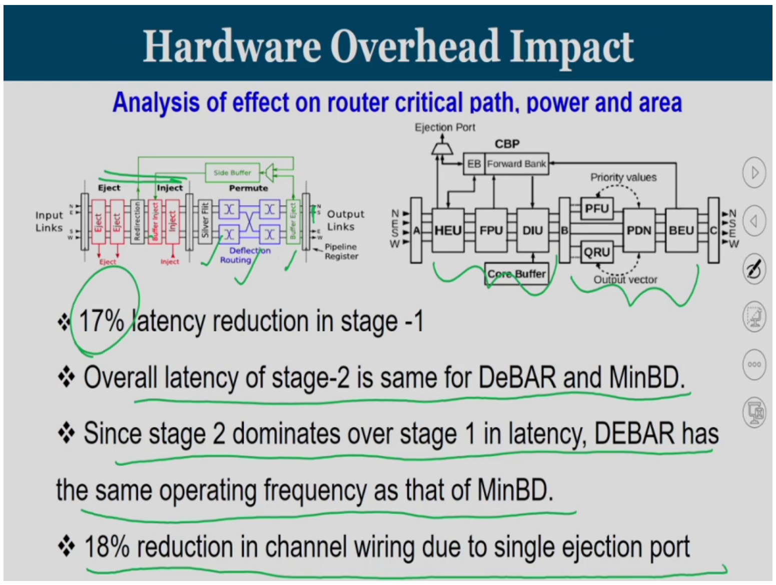
So, this is a comparison of deflection rate of MinBD and DeBAR. MinBD you can take the average deflections per flit is on the higher side. When it reaches 0.24 packet injection rate, you can see that you know every flit is roughly getting at most for at least two hops extra. For deflection rate of DeBAR is also increasing, even at slightly higher load DeBAR packets are experiencing roughly 1.5 deflections per flit. So, this graph tells that when the load is increasing, the conflict is increasing, and more packets will get deflected away. But, a number of flit deflecting away is drastically reduce in the case of DeBAR because of a better priority mechanism.

You can see the same trend in the case of transpose traffic, where MinBD has heavy deflections, whereas DeBAR has a slightly lower number of deflections. So, this shows that why virtue of the priority mechanisms, the changes in the router micro architecture are able to reduce the number of deflections, thereby reducing the average package latency.

So, the whole concept of moving towards buffer less deflection routing, which was mentioned in the chipper was to reduce the area as well as power consumption, because we are getting rid of buffers. But, once we do not have buffers at higher traffic rate, it will lead to heavy deflections. And that was been mitigated by introducing the concept of side buffers, first proposed by Chris Fallin and at all in the knocks paper leading to the MinBD design.

And some of the micro architectural features proposed by MinBD was been revisited by the authors of the DeBAR paper. And they were trying to put up a different kind of a priority mechanism, trying to get rid of the dual ejection ports and some kind of a dual injection unit, where fairness is given to both packets inside buffer as well as in the core buffer, so that we give into the performance analysis.

(Refer Slide Time: 47:25)



Hardware Overhead Impact

Analysis of effect on router critical path, power and area

Side Buffer
Eject
Inject
Permute
Input Links
Output Links
Deflection Routing
Pipeline Register
Ejection Port
CBP
EB
Forward Bank
Priority values
PFU
HEU
FPU
DIU
PDN
BEU
QRU
Core Buffer
Output vector

- 17% latency reduction in stage -1
- Overall latency of stage-2 is same for DeBAR and MinBD.
- Since stage 2 dominates over stage 1 in latency, DEBAR has the same operating frequency as that of MinBD.
- 18% reduction in channel wiring due to single ejection port

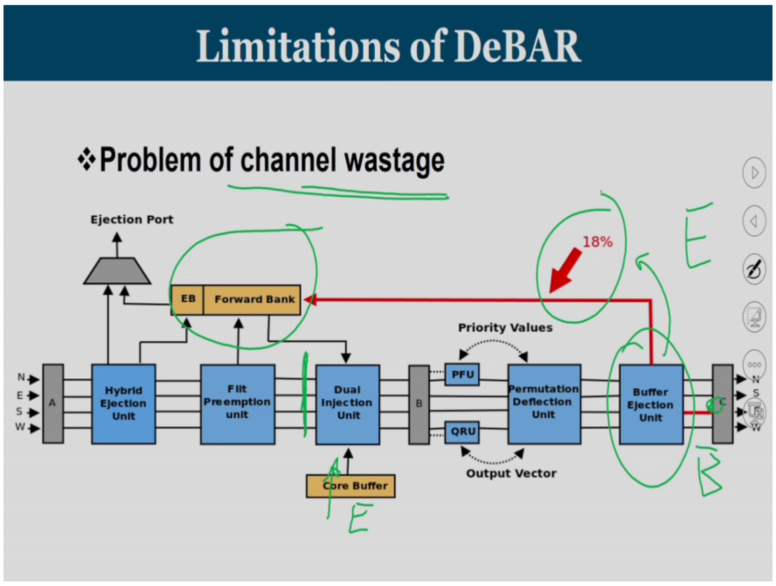
Now, coming to the hardware overhead. So, when's you have a design in hand computer architects, normally try to describe this hardware in hardware description language like VHDL and Verilog. So, once you model your micro architectural feature using this hardware description language. Then we are going to synthesis them using the appropriate IDE tools and a find out the synthesis report in terms of area and power consumption.

So, both MinBD and DeBAR routers, one implementing on this verilog design. It was shown that the stage one of DeBAR is 17 percent less time consuming than that of stage one of MinBD, because in MinBD you have dual ejection unit, you have a redirection, you have a buffer inject. So, there were many units on stage one of MinBD, but when compared to that the number of combinational block needed in implementing stage one of DeBAR is rather less which is so the maximum propagation delay for the circuits for DeBAR is 17 percent less, when compared to that of MinBD.

But, the overall latency of stage-2 is same, so because stage two is more or less same, we have a silver flit you have a permutation deflection network and buffer eject. Here also we have a routing mechanism that takes care of the priority plus PDN plus buffer eject. So, stage-2 is basically same. Since, both MinBD and DeBAR are pipeline the routers. We know that in the case of a pipeline of multiple stages, whichever stage is taking more time, and that is going to govern your pipeline timing.

So, since even though we say little bit in DeBAR in the stage-1. Since, stage-2 both for DeBAR and MinBD it is same, then that means that since stage-2 dominates over stage-1 in latency. DeBAR can operate in the same frequency as that of MinBD. So, DeBAR routers will also operate at the same frequency. So, the critical path of DeBAR is same as that of critical path of MinBD. Since, the dual ejection bandwidth is been reduced. We could achieve 18 percent of reduction in channel wiring due to the single ejection port. So, these are the performance that we gain with respect to the DeBAR routers in comparison with MinBD.

(Refer Slide Time: 49:22)



Limitations of DeBAR

❖ Problem of channel wastage

Now, let us try to understand, what are the problems associated with DeBAR router. So, we have to see the sequence the concept of chipper router itself was evolved from the limitations of bless the conventional bless router. And the limitations of chipper was addressed by MinBD router, limitations of MinBD router was addressed by DeBAR router. Now, we will try to see, what are the limitations of this DeBAR router.

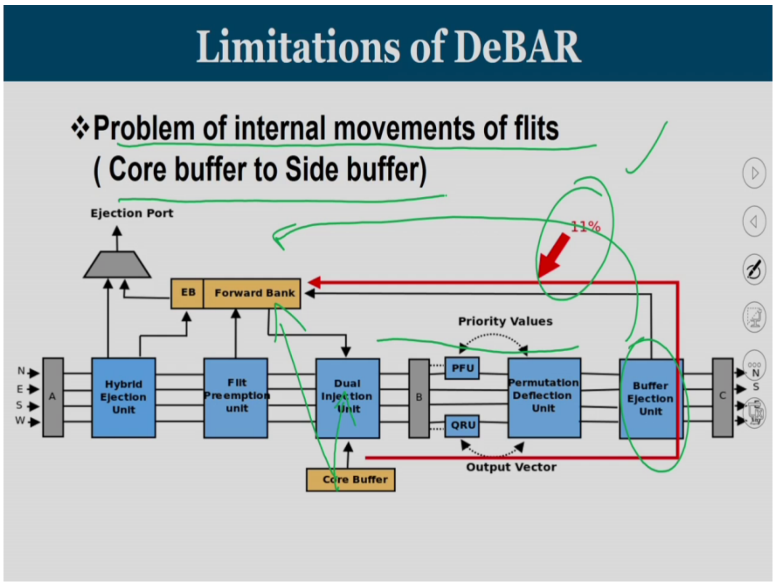
So, the first problem with respect to DeBAR is channel wastage. This is the router pipeline of DeBAR. Now, consider the case that you have a flit that is waiting in the core buffer. And injection from core buffer into the router pipeline can happen only if, one of the link in the router pipeline is idle. So, when there was a flit that is waiting to enter into the router pipeline, it was denied permission due to the fact that all the channels where having valid flits. But, it so happened that after the port allocation stage one of the flit got a non-productive port, and that flit was moved about from the side buffer.

Essentially leading to one of the output port going idle, this scenario is called channel wastage. So, I will rephrase the same scenario, we have a flit that is waiting in the core buffer to enter into the router pipeline. At the time entry into the router pipeline was denied, because none of the internal flit channel was idle. So, the flit will wait there. But, it so happened that all the flits that is that was there in the internal pipeline channel. After the port allocation, some of the flits got non-productive port.

And we see that the buffer eject unit that is going to remove the flit has found that one flit is been removed. For example, you assumed that was the flit that was assigned with east output port. So, the flit that got east output port was not happy. So, that flit was being taken to the side buffer, because it is a deflected flit. Now, you have a scenario when east port is idle, but we have a flit in the core buffer that is looking for east output port.

So, to summarize a flit looking for a specific output port is waiting in the core buffer, because it cannot inject. At the point of injection all the channels were busy, but that specific output port for which the flit is waiting that particular port is idle. This is called channel waste; because we have we are wasting a channel, when there is a valid flit looking for the same channel. This roughly happens around 18 percent of cases, when the implemented DeBAR router.

(Refer Slide Time: 52:19)



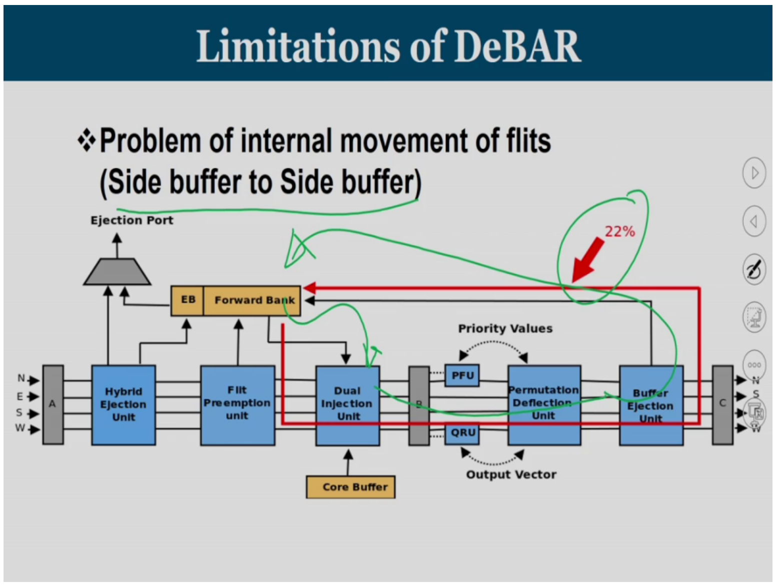
Limitations of DeBAR

- Problem of internal movements of flits (Core buffer to Side buffer)

The second one is called problem of internal movement of flits. So, consider the case that you have a flit that is entering from core buffer that is going into the dual injection unit. And it goes all the way into the permutation deflection network. It so happened that the flit that got recently added, the flit is going to have in going into the permutation deflection network. It got a non-productive port. Once you get a non-productive port, the buffer eject unit is going to take you into the side buffer.

So, it so happened that in 11 percent of cases, a flit that recently entered from the core buffer, reached in the permutation deflection network, got a non- productive port leading to movement into the side buffer. So, essentially it is a movement from core buffer to side buffer. This is called problem of internal movement of flits. There is no productive movement for the flit, the flit is just moving from array inside a router to some array other array inside the same router. So, this is called core buffer to side buffer movement, it happened roughly around 11 percent of cases.

(Refer Slide Time: 53:25)



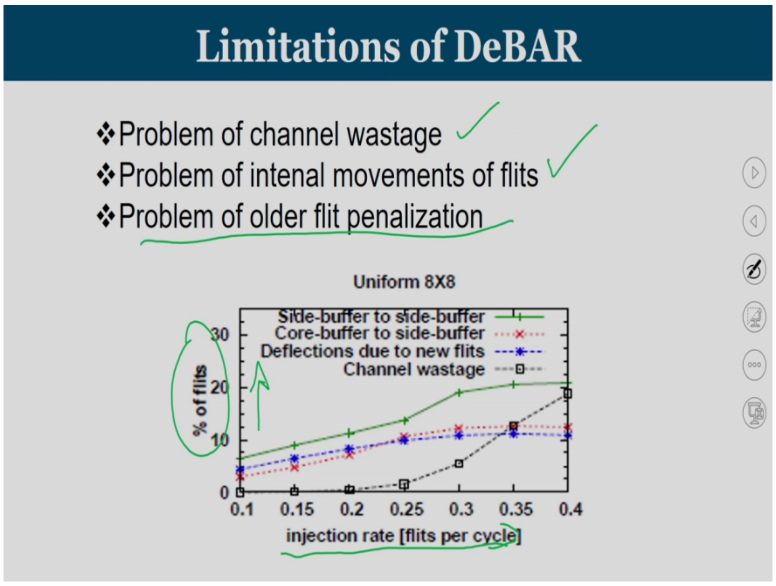
Limitations of DeBAR

- Problem of internal movement of flits (Side buffer to Side buffer)

The third limitation of DeBAR is problem of internal flit movement from side buffer to side buffer. So, the concept of side buffer to side buffer is there was a flit which reaches side buffer, because it was assigned a non-productive port in the previous cycle. So, we have a flit in side buffer, which tries to reenter the router pipeline, and in this cycle also upon reaching permutation deflection network. It got again a non-productive port, so that flit is going back to side buffer again this happened in 22 percentage of the cases.

So, a flit which was there in side buffer, enter the router, go for port allocation again come back to side buffer, because the port allocation was not in favorable for the flit. Why this is happening, when a flit is re injecting into the router pipeline, it is priority is not changing, because in DeBAR the priority is based on hops to destination. Since, the priorities still the same, the flit may again compete with the other flits, it may not get the productive port. So, here also we have side buffer to side buffer issue.

(Refer Slide Time: 54:28)



Limitations of DeBAR

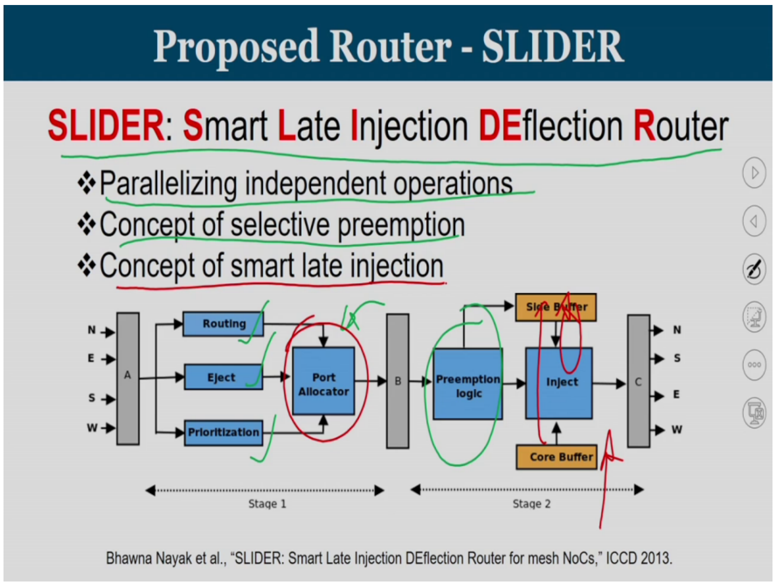
- ❖Problem of channel wastage ✓
- ❖Problem of intenal movements of flits ✓
- ❖Problem of older flit penalization

Uniform 8X8
Side-buffer to side-buffer
Core-buffer to side-buffer
Deflections due to new flits
Channel wastage
% of flits
0
10
20
30
0.1
0.15
0.2
0.25
0.3
0.35
0.4
injection rate [flits per cycle]

So, if you look at, the problem of channel wastage, the problem of internal movement of flits, and there is one more problem that is called the older flit penalization. In DeBAR, the priority mechanism used was holders like hops to destination. So, when you have a really old flit is not even nearer to the destination, those flits may lose out in arbitration to a newly generated flit. So, consider the case that you have a newly generated flit travelling to a nearby destination. You have an old flit that is travelled all the way, but still it has to go further. And these two when they compete, since the priority is not based upon age your round robin, the priority is based on hops to destination.

The newly generated flits whose destination is nearby will penalize an old flit. So, let us called the problem of older flit penalization, you can see that how much percentage of flits are being affected by problem of channel wastage, problem of internal movement of flits, and problem of older flit penalization. We can see that as injection rate increases, all these problems are going to affect more number of flits; you can see that all this graph are tilting towards the y axis. So, the limitations of DeBAR are more exposed, when you go for higher traffic injection rates.

(Refer Slide Time: 55:58)



Proposed Router - SLIDER

SLIDER: Smart Late Injection DEflection Router

- ❖Parallelizing independent operations
- ❖Concept of selective preemption
- ❖Concept of smart late injection

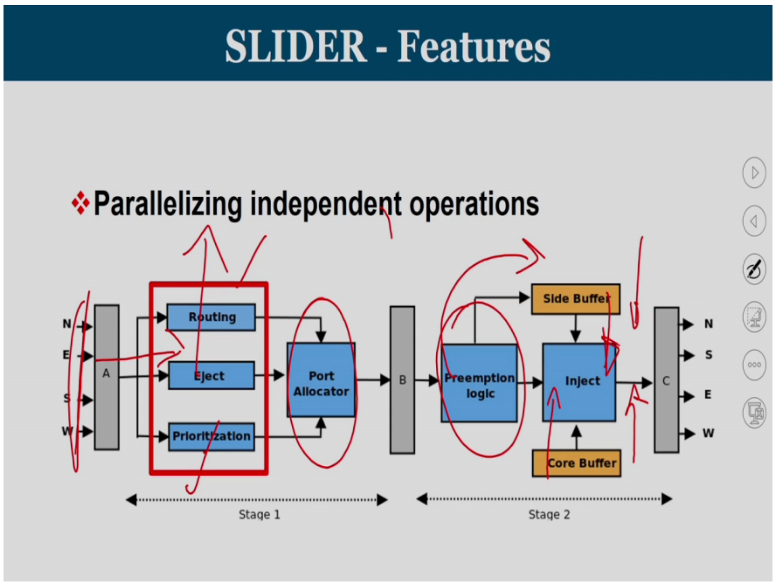
Bhawna Nayak et al., "SLIDER: Smart Late Injection DEflection Router for mesh NoCs," ICCD 2013.

So, here was at another proposal by Bhawna Nayak et al, it is published in conference on computer design in 2013. They were proposing a new router called SLIDER-Smart Late Injection Deflection Router, where this is the router pipeline, it just reorganizing the various units in DeBAR itself. So, three main features is there are three units the routing unit, the eject unit, and prioritization unit.

So, parallelizing independent operations, then we have a concept of selective preemption. So, we have a preemption logic that comes over here. So, the PDN will move to stage-1 this is port allocator is basically your PDN. So, PDN is moving to stage-1 this stage-1, and the injection happens late in the pipeline rather than having an injection unit early.

The injection from side buffer as well as core buffer is happening late in the pipeline. And there is no movement from core buffer to side buffer, and side buffer back to side buffer. Packets will enter into the inject stage, only if they are going to get a productive movement. So, the concept of smart late injection is inject the packets of late of possible in the pipeline such that you look into what are the ports that are available. And if your productive port is available, then inject there. So, smart late injection deflection router. So, this is called a concept of slider.

(Refer Slide Time: 57:23)



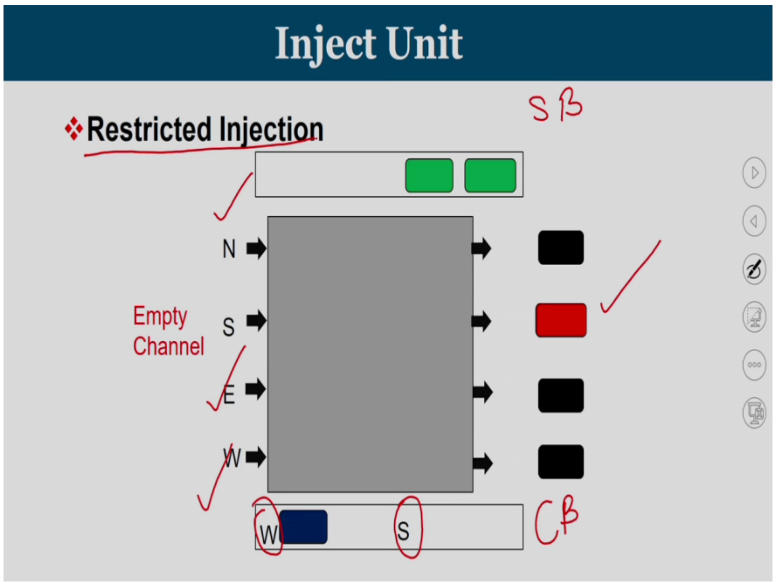
SLIDER - Features

- Parallelizing independent operations

We have three parallel operation that can be done. The routing unit only one to destination address. The prioritization unit by hops to destination priority that also want the destination address. So, they just extract the destination address field from the incoming flits. And the flit will directly go to the eject stage, and they are going to eject the flit. And then, we have the port allocator. So, the port allocator is basically your PDN. And the eject unit will take care of the flit to be ejected. And remember SLIDER comes back to a single ejection bandwidth. And port allocator is normal PDN. After, that since the priorities already computed and stored in the flit.

The stage-2 of the router pipeline has the preemption logic. So, if the flits in the side buffer and it core buffer are really starving, then preemption happens. If they are not starving, then we have the inject unit. They have basically dual inject unit based on odd even cycles, whether flit will be injected from core buffer or flit will be injected from the side buffer. So, the concept of late injection is a bit tricky. So, the concept of smart late injection we will just try to understand what it is.

(Refer Slide Time: 58:37)

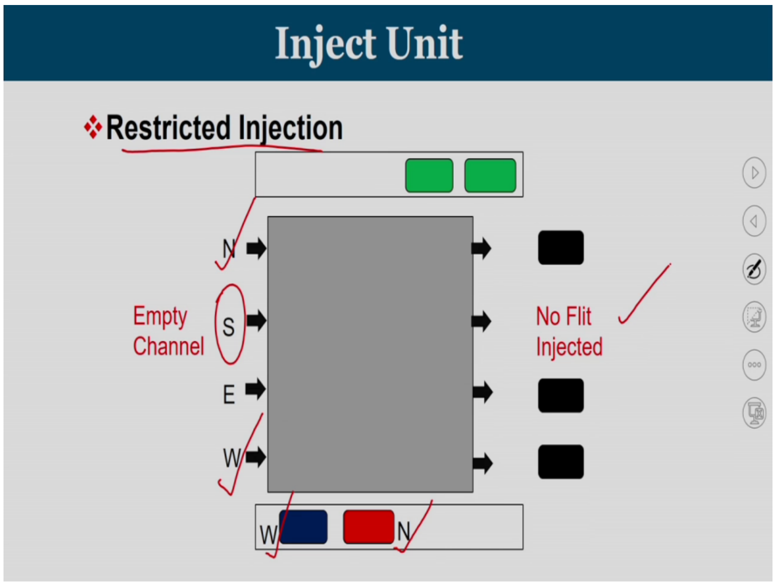


This is the inject unit, what you see there down is the core buffer, and what you see is there up is called the side buffer. So, there are two green flits waiting in the side buffer. And there are two flits in the core buffer; the blue flit is waiting for west output port. So, its reproductive port is west, and the red flit is looking for the south output port.

So, consider a scenario that in by the time this flits reach the injection stage, the south channel is empty. So, the south output port there is no flit. If you have a flit, going to north direction, you have a flit going to east direction, and then you have a flit moving to the west direction, the south channel is empty. You can see that in core buffer, you have the red flit waiting for south.

So, this red flit will enter from core buffer, and then it is going to travel. So, the flit got its productive port that is what the take away is. So, this is called a restricted injection. Restricted injection means a flit will be injected into the router only if the direction, which takes the flit in a productive way, which takes a flit closer to destination or the productive port is available for the flit. In this case, we have seen that the red flit is looking for south output port, and exactly south output port is empty. So, flit will be injected into the router, only if the productive port is idle.

(Refer Slide Time: 60:05)

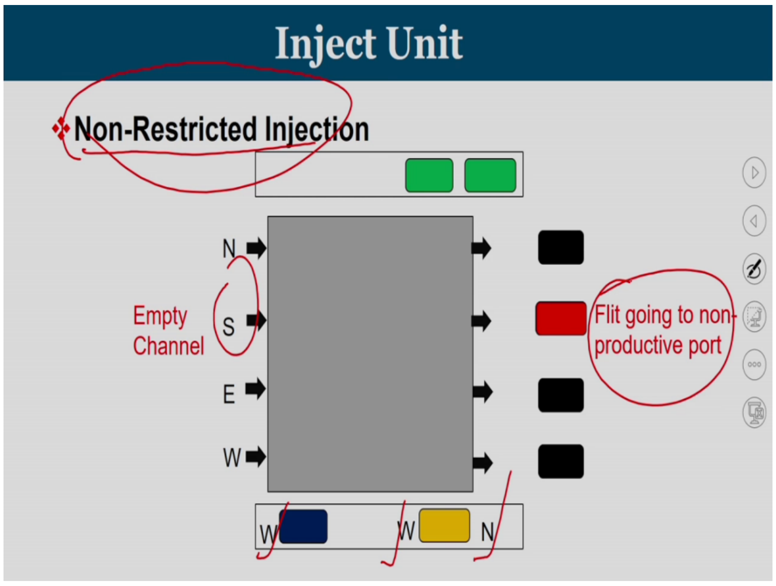


Inject Unit

- Restricted Injection

Now, consider another scenario. We have two flits, one is waiting for west, other one is waiting for north are there in the core buffer. But, only the south channel is empty the flit. Already the north output port has a flit, and the west output port has a flit, but there is an empty south channel. Since, it is a restricted injection the flit would not get injected into this particular empty slot. So, there is no flit that is injected.

(Refer Slide Time: 60:33)



Inject Unit

- Non-Restricted Injection

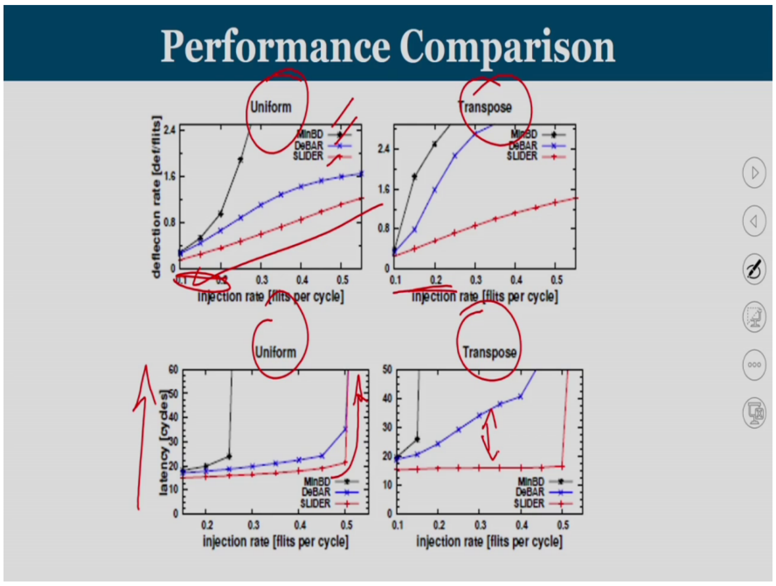
And then we moved to the non-restricted injection. So, if you are very much choosy about injecting a flit in the case of a slider, then your core buffer and side buffer will get accumulated with flits. You may not be able to inject flits, because your preferred port may not be freely available. Some other port may be freely available, but since your preferred port is not there, you refrain from injecting the flits.

Is this scenario continuous for multiple clock cycle, then slowly your core buffer and side buffer, we will get accumulate with more number of flits leading to overflowing of them or throttling of applications. This is prevented by two mode, you have two different modes for slider. One is called restricted injection and the other one is called non-restricted injection.

In the case of restricted injection flits will be injected only to their productive output ports and non-restricted injection at the routers which into non-restricted injection, when the buffer occupancy level of core buffer is more than 50 percent of its capacity. So, you can see that a core buffer can accommodate maximum of four flits, already three flits are there. So, it is going to be near full. Once it is near full, then there is no point in moving for restricted injection, you are moving into non-restricted injection.

So, in this case the south is the empty channel. One flit wants west, one flit wants west, and the third flit wants north. But, I am not going for restricted injection, because now my buffer occupancy is very high, I cannot offer this. So, whatever be the channel that is available the flit gets added into it. So, you can see that one of the flit is added. And that is not the direction, but the flit was looking for. So, the flit is going to a non-productive port. So, when the buffer occupancy of a core buffer or side buffer is nearing to full, then the router operates in non-restricted injection. So, flit will be injected into whatever output port that is freely available.

(Refer Slide Time: 62:33)



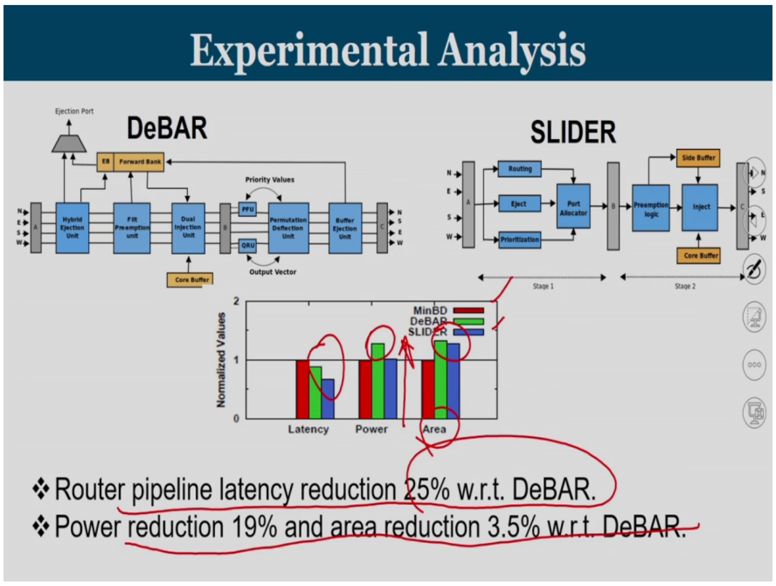
Performance Comparison

Let us look at the performance characteristics, this is called deflection rate. There is a comparison between three deflection routers we learned so far MinBD, DeBAR, and SLIDER. MinBD is shown by black color, DeBAR is shown by blue color, and SLIDER is shown by the red color you can see that. Deflection rate of the SLIDER is much lower than that of MinBD and DeBAR; this is because a flit will deflect in SLIDER, only if it is in the non- restricted injection. So, at very low packet injection rate, DeBAR has a very less deflection rate. This is for transpose traffic, and the previous one is for uniform traffic.

Once the deflection is low, surely it is going to deflect in the latency also that is what we can see for uniform as well as transpose traffic in the load versus latency graph. X axis is packet injection rate, which is an direct representation of number of packets injected into the network. And y axis is the latency, as we can see that DeBAR offers a lower packet latency.

The red graph is lower to that of blue, blue is DeBAR. And SLIDER is the red one SLIDER has a lower packet latency than DeBAR. And SLIDER saturate that the point of saturation of SLIDER is also late. So, SLIDER can accommodate more traffic, it can manage congestion better than that of DeBAR. Similarly, this trend is visible it has a lower packet latency than that of DeBAR.

(Refer Slide Time: 64:01)



❖ Router pipeline latency reduction 25% w.r.t. DeBAR.
❖ Power reduction 19% and area reduction 3.5% w.r.t. DeBAR.

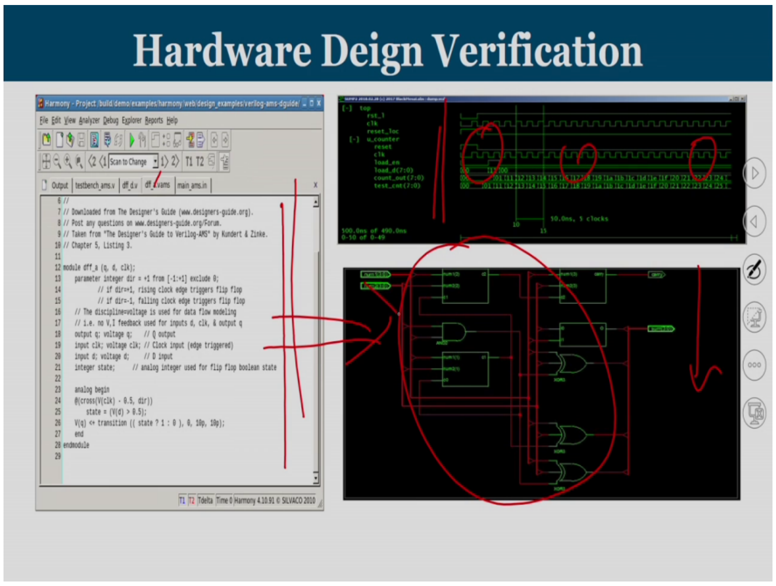
So, the experimental analysis comparison of DeBAR and SLIDER. So, when the, if you look at the latency of a single router, this was done by the verilog implementation. Once we implement all the units of the DeBAR pipeline as well as the SLIDER pipeline, and then synthesize it. It was found that the the router pipeline latency reduction by 25 percent with respect to DeBAR. The SLIDER is the blue one; SLIDER achieves 25 percent latency reduction with respect to the DeBAR.

This latency is not average packet latency, it is known as router pipeline latency. How big or how long is the combinational circuit, that implement DeBAR. The combinational circuit that implement SLIDER is 25 percent smaller in terms of critical path length than that of DeBAR. Similarly, there is a power reduction of 19 percent and area reduction of 3.5 percent. So, DeBAR is slightly complex circuit, it is going to consume more power, but SLIDER we are able to cut down the logic significantly less.

So, SLIDERs and MinBD more or less the power consumption of the router is same or when you look at the area, since the combinational logic is been effectively managed. The area you are able to get some (Refer Time: 65:23) to 3.5 percent. So, all these new routers for chip multicore processors any new idea that if a researcher has, the idea has to be first modeled in a network on chip simulator like Booksim, Nocsim, garnet many simulators are there, you model them, you get the latency, average latency, average deflection rate many parameters.

Similarly, you wanted to know how bulky is this circuit. And that is being done by a hardware implementation verilog synthesis of it.

(Refer Slide Time: 65:53)



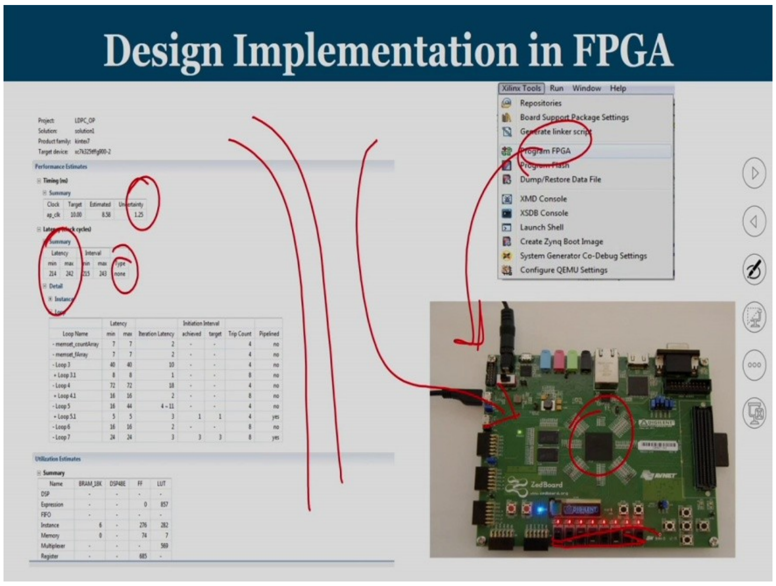
Hardware Deign Verification

```
//
// Downloaded from The Designer's Guide (www.designers-guide.org).
// Post any questions on www.designers-guide.org/Forum.
// Taken from "The Designer's Guide to Verilog-AMS" by Kundert & Zinke.
// Chapter 5, Listing 3.

module dff_a (q, d, clk);
    parameter integer dir = +1 from [-1:+1] exclude 0;
            // if dir=+1, rising clock edge triggers flip flop
            // if dir=-1, falling clock edge triggers flip flop
    // The discipline=voltage is used for data flow modeling
    // i.e. no V,I feedback used for inputs d, clk, & output q
    output q; voltage q;    // Q output
    input clk; voltage clk; // Clock input (edge triggered)
    input d; voltage d;     // D input
    integer state;      // analog integer used for flip flop boolean state

    analog begin
    @(cross(V(clk) - 0.5, dir))
        state = (V(d) > 0.5);
    V(q) <+ transition (( state ? 1 : 0 ), 0, 10p, 10p);
    end
endmodule
```

Harmony - Project /build/demo/examples/harmony/web/design_examples/verilog-ams-dguide/
File Edit View Analyzer Debug Explorer Reports Help
Scan to Change
Output
testbench_ams.v
dff_d.v
dff_a.vams
main_ams.in
T1 T2 Tdelta Time 0 Harmony 4.10.91 © SILVACO 2010

So, how will you typically do, hardware design verification. You have to write your code the verilog code inside some of the ISI tools. And then you synthesize and verify whether all the output signals that you are having is happening in the proper timing cycles. Generate the gate list, so this is the equal and gate level circuit of the code that we have.

(Refer Slide Time: 66:16)



Design Implementation in FPGA

Project: LDPC_OP
Solution: solution1
Product family: kintex7
Target device: xc7k325tffg900-2

Performance Estimates

Timing (ns)

Summary

Clock	Target	Estimated	Uncertainty
ap_clk	10.00	8.58	1.25

Latency (clock cycles)

Summary

Latency		Interval		
min	max	min	max	Type
214	242	215	243	none

Detail

Instance

Loop

	Latency			Initiation Interval			
Loop Name	min	max	Iteration Latency	achieved	target	Trip Count	Pipelined
- memset_countArray	7	7	2	-	-	4	no
- memset_fArray	7	7	2	-	-	4	no
- Loop 3	40	40	10	-	-	4	no
+ Loop 3.1	8	8	1	-	-	8	no
- Loop 4	72	72	18	-	-	4	no
+ Loop 4.1	16	16	2	-	-	8	no
- Loop 5	16	44	4 ~ 11	-	-	4	no
+ Loop 5.1	5	5	3	1	1	4	yes
- Loop 6	16	16	2	-	-	8	no
- Loop 7	24	24	3	3	3	8	yes

Utilization Estimates

Summary

Name	BRAM_18K	DSP48E	FF	LUT
DSP	-	-	-	-
Expression	-	-	0	857
FIFO	-	-	-	-
Instance	6	-	276	282
Memory	0	-	74	7
Multiplexer	-	-	-	569
Register	-	-	685	-

And then you get the synthesis report, how much is the latency, how much is the area, how much is the type, the number of various sub unit. So, you this is what is known as the synthesis report, and that will tell you what are the hardware components used. And then you generate the bit stream. Once you get the bit stream, you how to program it onto the FPGA.

And you can use the FPGA board, where the design that you developed inside your computer is transferred into the FPGA board and the board, now models let us say your DeBAR router or MinBD router or chipper router. And there are pins and knobs that is available, through this pins and LEDs you can give some input, maybe a packet number or destination number. And verify whether they are giving the appropriate results or not.

So, today we had a discussion on minimally buffered deflection routers. Basically, it is a side buffer deflection router we saw what are the limitations of chipper, and came up with MinBD router. The MinBD router was basically focusing on introducing one more level of priority called silver flit mechanism. And then we have a buffer eject unit, which will accommodate flits in side buffer. So, the concept of side buffer was introduced first by the MinBD router design, and then you have the dual ejection bandwidth.

Now, finding out some of the limitations of MinBD router, the DeBAR router was proposed, which has a hybrid ejection unit. And as a dual injection unit which will give equal preference to injections from side buffer as well as the core buffer. And the priority structure was totally like (Refer Time: 67:49) with a new proposal like rather than going for golden and silver flit mechanisms. It was better it was shown that if you go by hops per hops to destination that is going to be a better of priority mechanism. And the quadrant routing mechanism was also been discussed in the case of DeBAR.

Now, some of the problems with respect to DeBAR was channel wastage, and the problem of internal flit movements, and the problem of penalizing older flits by the newer flits. And this was addressed by yet another proposal course SLIDER - Smart Late Injection Deflection Router, where the inject unit is kept late in the pipeline, and injection happens either in two modes, it restricted mode and non-restricted mode.

If it is a restricted mode, packet injection happens only to the productive port. If it is non-restricted happens, packet injection will happen to whatever port that is available. So, in the last two lectures, we are trying to understand can we get rid of input buffer, yes, we can. We can remove input buffers, but that has to be compensated by an equally good intelligent system that will take care of the flits. And in order to reduce the number of deflection, side buffers are introduced manipulation of data or management of flits to the side buffer and from the side buffer has to taken care of by appropriate priority mechanisms.

So, in this way we have proposing, we have learned to find out how buffers can be removed, and there are cost effective replacement for buffers known as minimally buffer or side buffer deflection routers. So, with this will conclude about the router micro architecture. So, we have covered what is a basic structure of NOC, the routers different types of routers, flow control everything. Over the next few lectures, we will try to see a bigger picture where storages and interconnects are coming together and trying to improve the quality of tiled chip multicore systems.

Thank you.