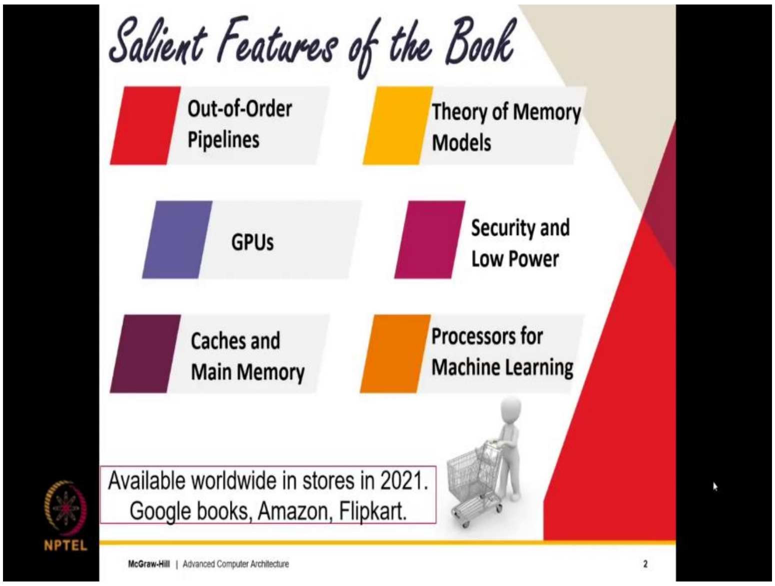
Advanced Computer Architecture
Prof. Smruti R. Sarangi
Department of Computer Science and Engineering
Indian Institute of Technology, Delhi

Lecture - 21
Caches
Part - III

(Refer Slide Time: 00:25)



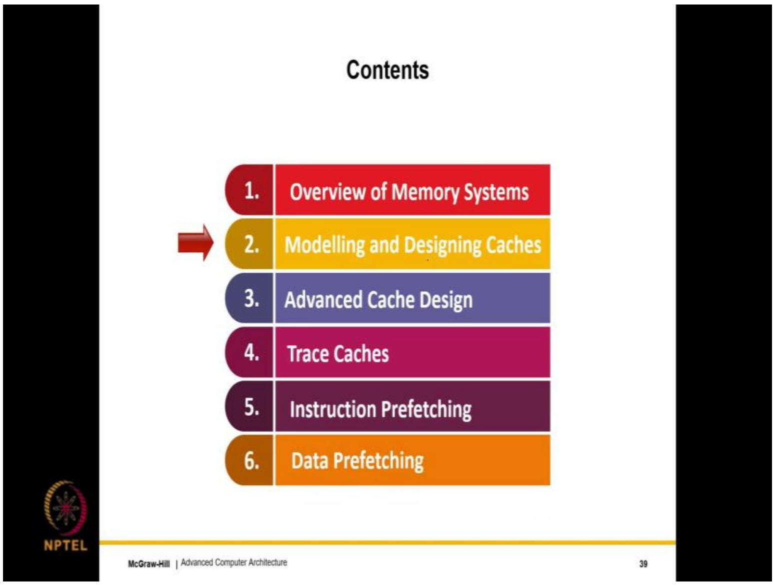
Salient Features of the Book

- Out-of-Order Pipelines
- Theory of Memory Models
- GPUs
- Security and Low Power
- Caches and Main Memory
- Processors for Machine Learning

Available worldwide in stores in 2021.
Google books, Amazon, Flipkart.

McGraw-Hill | Advanced Computer Architecture 2

(Refer Slide Time: 00:37)



Contents

1. Overview of Memory Systems
2. Modelling and Designing Caches
3. Advanced Cache Design
4. Trace Caches
5. Instruction Prefetching
6. Data Prefetching

McGraw-Hill | Advanced Computer Architecture 39

In this lecture, we will discuss Modelling and Designing Caches. So, modelling and designing caches is a very complicated engineering problem, and there are lot of factors that need to be taken into account. So, we will look at many of these factors, and see it is a complex interplay of the area budget, the timing, the power, and the implications on performance.

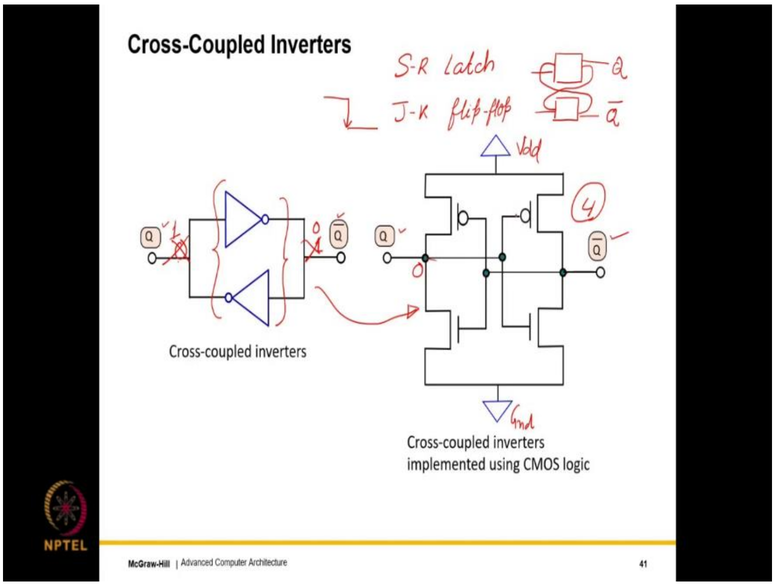
(Refer Slide Time: 01:02)

~~S~~RAM Array

McGraw-Hill | Advanced Computer Architecture 49

So, we will discuss two important technologies in this space, one is an SRAM array, a Static Random Access Memory, SRAM, an SRAM array, and other is the CAM array a Content Addressable Memory Array. So, these are the two basic technologies that are used for creating on chip memories.

(Refer Slide Time: 01:30)



Cross-Coupled Inverters

Cross-coupled inverters

Cross-coupled inverters implemented using CMOS logic

McGraw-Hill | Advanced Computer Architecture 41

So, we have all studied latches in our basic digital electronics courses. So, I will not discuss latches further, but there are some key concepts that a person should know before attempting to go through or understand this section.

So, one is the notion of an SR latch, a Set Reset latch. So, here basically we have a cross coupled pair of NAND gates. So, ultimately, in every latch to store a value, the output of one is the input of other and the output of the other one is the input of this. So, pretty much, and then we have some controlling inputs, and then we have the outputs which are typically a complement of each other.

So, this is typically the way that a latch is constructed, that we have this cross coupled pair. An SR latch is typically a cross coupled pair of NAND gates. So, this is something we want people to know before they proceed.

And the other concept that we want the we were to know is a notion of a J-K flip flop. So, this is also an important concept. Often, the words latches and flip flops are used interchangeably even though they should not be.

Say, flip flop is typically used where we are talking of a negative edge triggered circuit, but I would say that the terms are often misused, abused, any combination thereof. So, for example, in a pipeline when you talk a pipeline latches, what we actually mean is edge triggered latches, where at a negative edge the value get stored. So, the value gets stored and reflects at the output of the latch.

But, one thing that is clear is that a pair of NAND gates have a lot of transistors. Say, I have a lot of transistors we cannot really create a dense or compact memory array just with a pair of NAND gates. So, it will be very very inefficient when it comes to area and it is clearly not desirable. So, we need a new kind of a memory cell which is why instead of using a cross coupled NAND gate what we use is, we use a pair of cross coupled CMOS inverters.

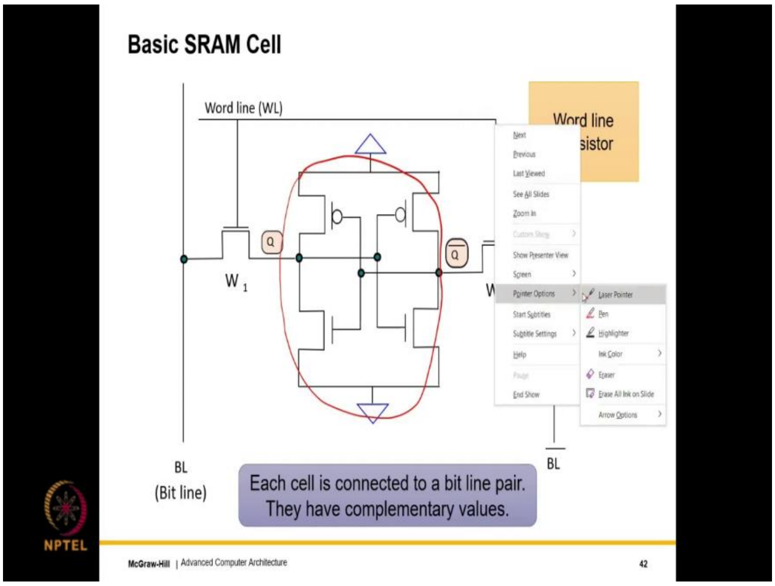
So, once we do that whatever is the value here the complement will come up over here Q and Q bar, but that is not the important point. The important point is that a cross coupled inverter made in this fashion is capable of storing a value. It is still not edge triggered, in the sense that this has nothing to do with a clock. So, it has nothing to do with a clock, neither does it have anything to do with clocks edges. But, any kind of a cross coupled structure because of mutual feedback is capable of storing a bit. That is the important take home point.

So, we will use two triangles. Triangle pointed upwards is the supply V dd and a triangle pointed downwards is ground. So, this circuit is just a realization of this. So, this is just breaking it down into CMOS logic. So, as you can see, the output of one is the input of the other and the output of one is the input of the other. That is all that we have to it.

So, this just has 4 transistors. So, this 4 transistor structure is capable of storing a bit. And of course, it has two terminals, one terminal has Q and other has the complement of Q which is Q bar. So, of course, a lot of analysis can be done, and we can look at the basic transistor equations, and we can see how exactly this structure will charge and discharge.

But, that is anyway not in the scope of our current discussion. It is just that if let us say we have a 0 over here. So, let us say we have a 0 over here, we will have a 1 over here and then a 0. Or the other state can be we have a 1 over here, then we have a 0 over here. So, as you can see it has two states. So, I am not going beyond that in the sake of readability, and also this is not in the scope of the current discussion.

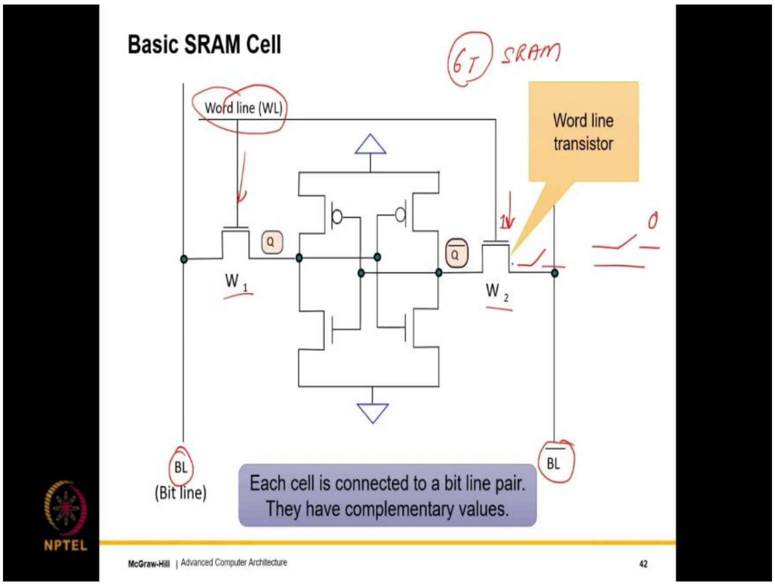
(Refer Slide Time: 06:06)



Basic SRAM Cell

Each cell is connected to a bit line pair. They have complementary values.

(Refer Slide Time: 06:33)



Each cell is connected to a bit line pair. They have complementary values.

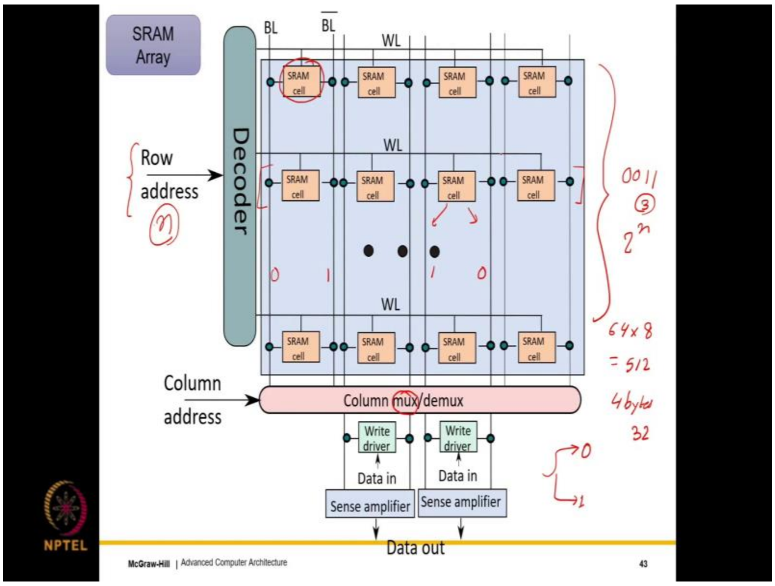
So, we can take the small cell over here which is a SRAM cell and we can actually create we can add two more transistors, they are known as access transistors; and actually create. So, I stand corrected, these 4 transistors are not the SRAM cell, that is only a part of it. So, when we add the two additional transistors, we have a 6 transistor structure and this becomes the SRAM cell.

The two additional transistors are W 1 and W 2, which you can see in the figure. These are called word line transistors. So, recall that a transistor is basically a switch. When the word line signal is the 1, the switch is connected. So, this becomes a closed circuit. Otherwise, if the word line the voltage on the word line which is this same as this, if the voltage here is 0, then the switch is open, which means that it is not connected.

So, in the 6 transistor SRAM cell, these two access transistors control the access of the transistor to the bit line pair. So, bit line are essentially the bit lines are two vertical lines on both sides of the cell. So, one is called BL and the other is called BL bar. So, essentially, they hold complementary values, if the switch is connected. If the switch is disconnected, then of course, the values in the cell remain what they are and they are unaffected by the voltages of the bit lines.

So, the way to access a cell a 6 transistor SRAM cell is basically by setting the word line voltage to 1, and that ensures that this cross coupled inverter pair gets connected to the bit line pair.

(Refer Slide Time: 08:08)



So, now let us create an array out of this. So, what we see is we have the 6 transistor SRAM cell. We create columns of these SRAM cells, and they are connected to both sides the bit line pair BL and $\overline{BL}$.

And so, then we have several such columns, and each row it shares the word line. So, as you can see each row shares the word line. And we have a decoder over here, the input is the row address. Say, the row address is n bits. We will essentially have $(2)^n$ rows. And what is the job of a decoder? The job of a decoder is to take an n bit input. It will have a $(2)^n$ output lines. One of them will be set to 1, depending upon the input.

So, the input is let us say 0 0 1 1, this basically says that 0, 1, 2, 3 it is counting from 0, the third line is set to 1. So, as I said if the row address is n bits, we have $(2)^n$ rows over here. One of the word lines is set to 1, and this enables all the SRAM cells, enables means connects to the bit line, all the SRAM cells in that row.

So, let us look at the case of reading first. So, when I am reading what will happen is that all of these bit lines will get connected. So, let us say this is the row. So, the bit lines will then get charged to their values. So, the bit lines will then get charged to their values. And so, let us say one bit line can be, let us say one bit line pair can be 0 1, other can be 1 0. So, whatever is the value stored in the SRAM cell that will come to the bit lines.

So, it is possible that we might have a very very wide row, in the sense the row might contain let us say 64 bytes, but we might be interested in only reading 4 bytes. So, what we will have is that we will have a multiplexer over here. So, which I am calling a column mux, demux.

So, we will have a multiplexer over here. What the multiplexer will do is that out of all the inputs that are coming, it will essentially choose the set that we are interested in. So, let us say that there are 64 bytes. 64 bytes means 64 * 8 = 512 bits. So, there are 512 cells in the row. Out of that let us say we are interested in only 4 bytes. So, 4 bytes is essentially 32 bits. So, we are interested in the values of 32 cells in the row. So, the mux structure will choose 32 out of 512 and then they will come to these circuits called sense amplifiers.

So, the sense amplifier is a specialized circuit which, what it does is that it senses the difference in voltage between the bit line pair, and then essentially converts it to a logical 0 or a logical 1. So, these are essentially voltages.

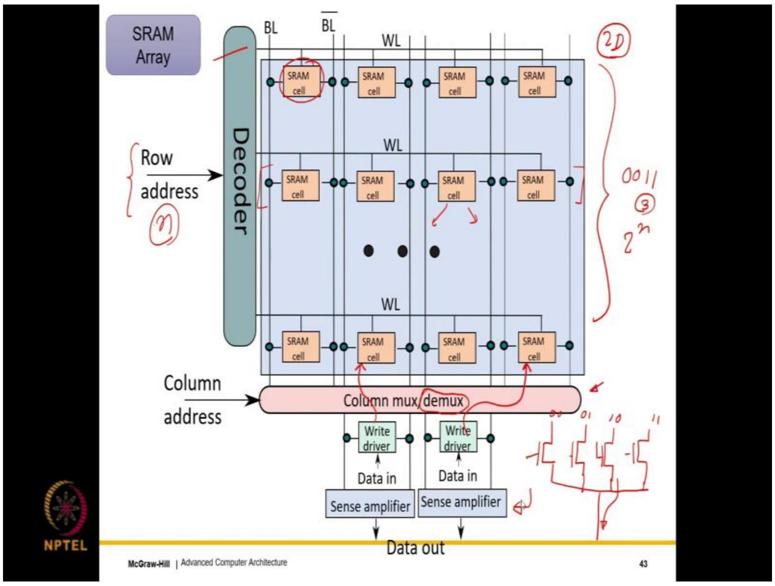
And the SRAM cell is a very weak driver. So, we will see that the sense amplifier is the special circuit, but give me a second to explain that. But, prior to that it is important to understand that the voltage difference in the bit line pair is what the sense amplifier will look at and it will convert that to a logical 0 or 1, such that the values can be sent to the CPU.

So, we have looked at the case of a read. Now, let us look at the case of a write. It is pretty analogous. So, what we do over here in the case of a write is that, first we do not use the sense amplifiers, rather the data that has to be written is sent to these write drivers. So, what is driver circuits do is that they basically set the voltages of the bit line pair.

Then, instead of using a column multiplexer, we will use the column demultiplexer. What the column demultiplexer is going to do is that it is going to do the reverse in the sense that if you want to write 4 bytes or 32 bits, and let us say the row has 512 SRAM cells. So, essentially whatever is being written has to be routed to the correct SRAM cell. So, we are writing 32 bits, but we have a choice of 512 so the demux structure will ensure that whatever we are writing actually goes to the right column. So, this goes to the right column.

So, a multiplexer and demultiplexer these are very standard circuits, but there is little bit of a twist here. So, typically multiplexers and demultiplexers use logic gates to a large extent, but when we are looking at this kind of logic, we typically do not use logic gates in an SRAM array. Instead, what we do is that we use pass transistors. So, a simple way of using pass transistor is like this.

(Refer Slide Time: 14:01)



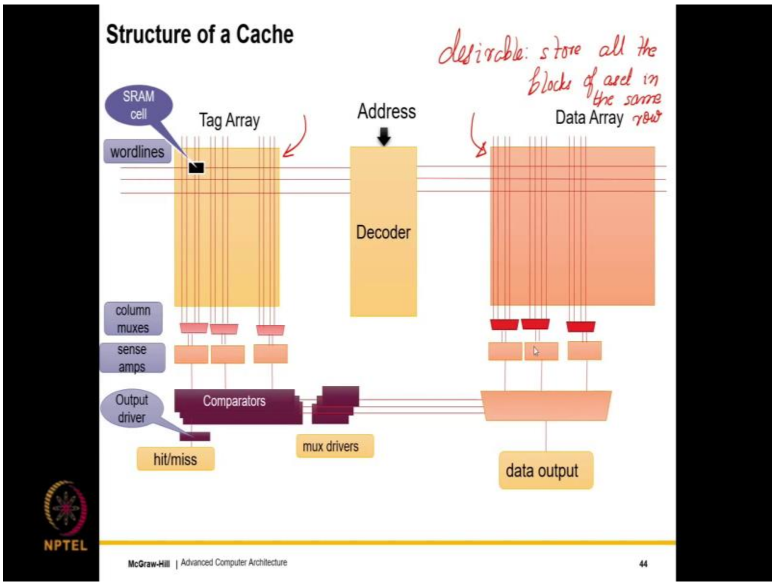
So, let us say that there are 4 inputs and we need to choose one. So, what we do is that for each one of them similar to the word line transistor, we just add a switch. So a transistor is just a switch. It is nothing more than that. So, it is just. And the switch has two states open and closed. So, for digital logic this is all that you need to know. Of course, when the transistors have analog properties as well, but we are not interested in them. So, we are only interested in the digital properties of transistors.

So, what we do is that we have a specialized circuit. Given these 4 inputs, what the specialized circuit would do is that it would enable one of these inputs and then these would just be connected to one line, so which is output. So, what will happen is that let us say if they are numbered 00, 01, 10, and 11, depending upon which input we want the enabling line at the gate of the transistor will be set to 1 and that will ensure that input will flow to the output.

So, this is how you would actually construct a multiplexer using pass transistors. So, a very similar technology is used over here, where it is primarily based on pass transistors. And this is a rather fast and area efficient way of constructing this. On similar lines, we can just reverse this and this will become a demultiplexer where the current flow will be in the reverse direction, but the logic, the controlling logic will still be the same.

So, this is for new b’s. This is a simple SRAM circuit. So, what you need to notice that we have an array of SRAM cells that are arranged in a 2D fashion write a rectangular array. Note the position of the decoder, note the column mux, demux, and of course, the sense amplifies. So, we will talk more about them. So, we are not done yet.

(Refer Slide Time: 16:16)



Structure of a Cache

desirable: store all the blocks of a set in the same row

McGraw-Hill | Advanced Computer Architecture 44

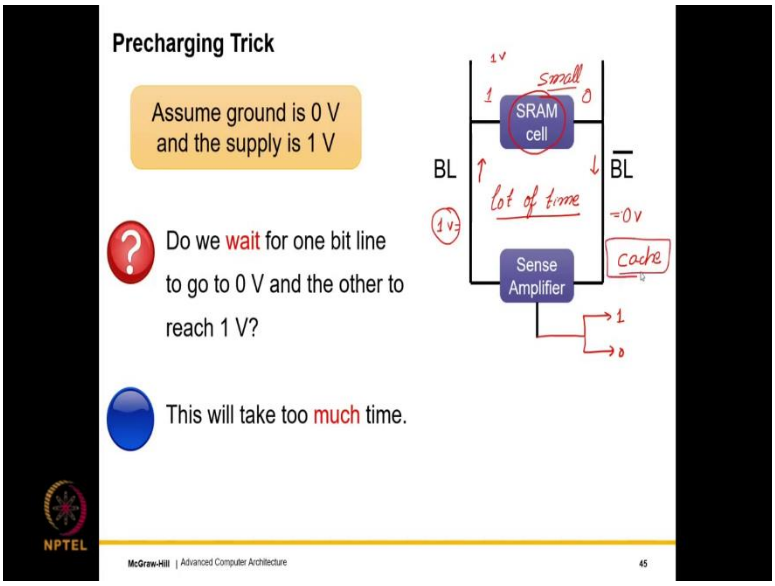
So, let us visit them slightly later, but we are now in a position to actually look at the structure of a cache. So, we have a tag array over here, and we have the data array. So, they can share the decoder, there is no reason to have separate decoders, we can have just one.

So, the decoder will then enable the appropriate word line. Once the word line is enabled, the values will start flowing in into the column muxes, into the sense amplifiers. And of course, if it is a set associative cache, then what we can do is in the tag array the data for all the sets can be stored in the same row of SRAM cells similar for the data array.

And then, all that we need to do is we need to compare, and after performing the comparison the tag comparison. We can then have a multiplexer to choose the data output. So, we are assuming that all the blocks of a set are stored along the same row, and this will clearly give us some advantage. So, what we are doing over here is is that we are connecting a concept that we learnt earlier with the physical organization of a cache.

So, we are saying that a desirable property is that store all the blocks of a set, in the same row of the array. So, then we can read it in one go, and then we can perform the comparison, and choose one of them.

(Refer Slide Time: 18:13)



Precharging Trick

Assume ground is 0 V and the supply is 1 V

Do we wait for one bit line to go to 0 V and the other to reach 1 V?

This will take too much time.

McGraw-Hill | Advanced Computer Architecture 45

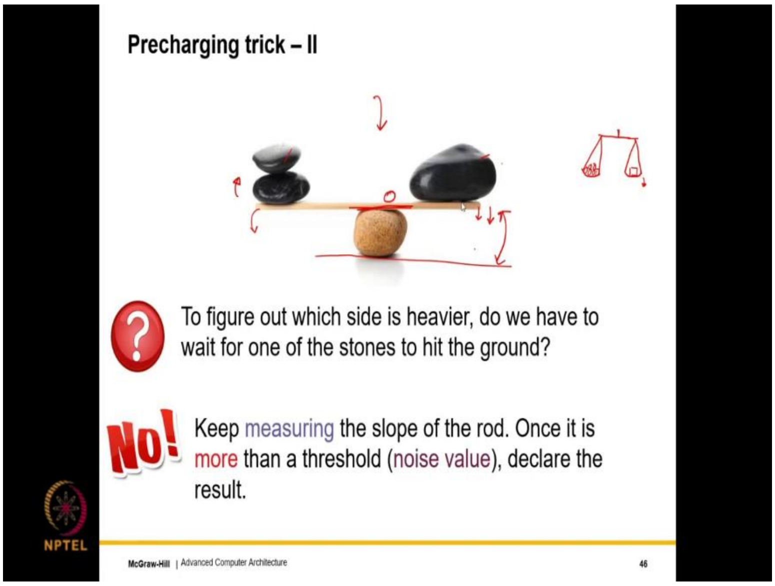
So, now let us revisit the issue of the sense amplifiers. So, what are we doing? So, I have just simplified this circuit for you where we have an SRAM cell, and we have 2 bit lines on both sides.

So, let us say the SRAM cell stores 1 and 0. So, the moment it is connected the voltage here will move towards the logical 1. So, let us assume logical 1 is also 1 volt which is roughly the case and a logical 0 is 0 volts and the voltage here will go down. Until, this becomes equal to 0 volts and this becomes equal to 1 volt. And then of course, the sense amplifier will sense this, and tell you that look the output is either a logical 1 or logical 0. This is simple.

The main disadvantage over here is, that if we actually wait for the voltage to reach 1 volt it will take a lot of time. The reason it will take a lot of time is that for the reasons are efficiency because you want to pack in as much a memory as possible on the chip. This is a very small circuit, the transistors are as small as possible, so smaller the transistor weaker it is. What do you mean by weak transistor? It means it can drive little current.

If it is current driving capacity is low for it to charge the bit line to 1 volt, it will take a lot of time. So, this is clearly slow and we are performance creek. So, we want to run our circuit as fast as possible, our entire processor as fast as possible. And since the SRAM technology is going in creating caches, this is vitally important for us. That is where we do something called a trick.

(Refer Slide Time: 20:12)



Precharging trick – II

To figure out which side is heavier, do we have to wait for one of the stones to hit the ground?

No! Keep measuring the slope of the rod. Once it is more than a threshold (noise value), declare the result.

McGraw-Hill | Advanced Computer Architecture 46

So, take a look at the scale over here. Let us assume that this big stone is slightly heavier than the load on the left, then gradually this the bar over here will tilt towards the bottom and this will tilt towards the top.

So, the question is so, we want to figure out which side is heavier this side or that side. If let us say this stone is heavier, do we wait for this bar to actually hit the ground which is over here? If we wait for it to hit the ground which is which means it needs to cover this much of distance, it will actually take a lot of time. What we can do is we can do something smarter.

So, we can keep measuring the slope of the rod. So, initially let us say it is 0. So, what can happen is because have a little bit of wind, there might be little bit of oscillation and so we need to ignore that. But, once we are sure that the rod has tilted, right, the rod, bar, whatever, but I am calling it the rod.

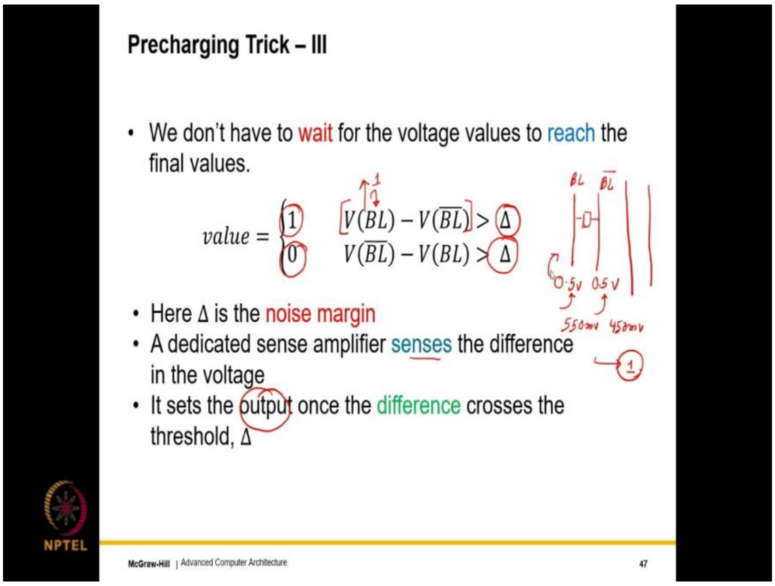
And so, once let us say this rod or slab is tilted, beyond what you would expect, beyond a normal fluctuation because of wind you can confidently say that this side is heavier. And let us see if it tilts the other way, like if tilts this way, you can say that this side is heavier. See, you need not wait for one side to actually hit the ground because just by seeing the direction, it will be I am clear which side is heavier.

So, this is something similar you would have seen these scales that are used so many of you, so I am not sure if this is there or not. But, at least when I was a child, so if the vegetable vendor did not have these weight measuring machines. So, the so nowadays they have this machine. So, you put in let us say potatoes, it tells you it is 650 grams on the basis of that they calculate the price.

But, when I was a child that was not the case, so what you would do is you would put potatoes on one side and the vegetable vendor would put weights on the other side. So, let us say 1 kg weight, half kg weight and so on. And then, what he would do is that he would wait for either, so let us say initially of course, one side will be unbalanced. So, when one side is unbalanced one side will tilt. So, let us say the weights are more. See, he never waited for this side to actually hit the ground. He quickly removed the weights and replaced them, until there was roughly a balance.

So, we can use the same logic over here where we just look at the direction in which it is moving, and after that we essentially declare the result that which side is heavier. So, can we do something with bit lines? Yes, we can. And the technique is called precharging.

(Refer Slide Time: 23:26)



Precharging Trick – III

- We don't have to wait for the voltage values to reach the final values.

$$value = \begin{cases} 1 & V(BL) - V(\overline{BL}) > \Delta \\ 0 & V(\overline{BL}) - V(BL) > \Delta \end{cases}$$

- Here Δ is the noise margin
- A dedicated sense amplifier senses the difference in the voltage
- It sets the output once the difference crosses the threshold, Δ

McGraw-Hill | Advanced Computer Architecture 47

So, the technique is like this that we monitor the voltage of the bit line and the complement of the bit line.

If let us say, this difference exceeds the threshold, whether threshold is essentially the noise margin. What is the noise margin? It is equivalent to the effect of wind on a scale. So, basically what this means is that since we have a long copper wire, it will pick up some amount of electromagnetic interference, right from all sides. In the sense, if you turn on a microwave oven then a changing magnetic field will induce a potential difference, right. It will be minor, but nevertheless there will be something. Also, if there is another line that is close by because the bit lines run in parallel.

Say, there is one current flowing over here, it will induce some amount of a current over here, so which is called crosstalk noise. So, because of several effects of noise, there will be little bit of a noise margin. So, we need to ignore that. But, the moment the difference crosses this threshold, we need not wait for this voltage to actually reach 1. We do not have to wait.

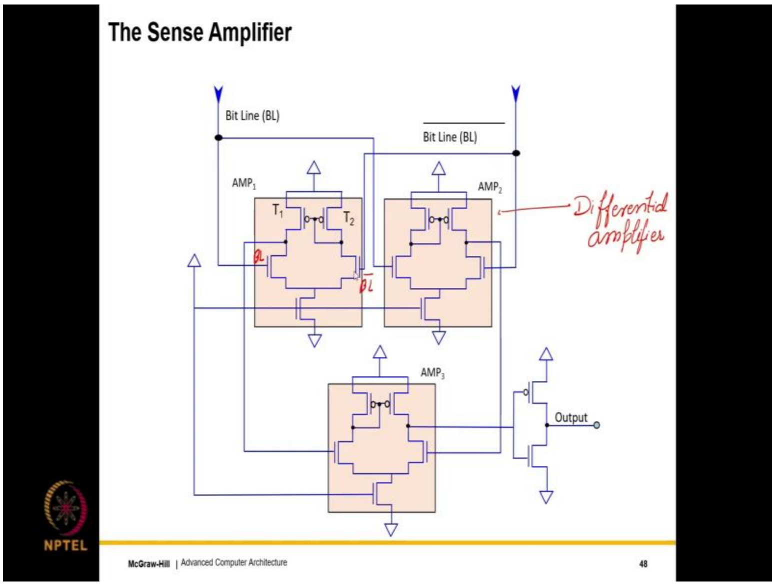
Once it crosses the threshold, we declare victory and say that it is a logical 1. And similarly, in the other direction, once the voltage of the complement of the bit line exceeds the voltage of the bit line by the same noise threshold, we declare a 0. So, this is the brilliant idea. So, this is known as precharging.

So, what we essentially do is we take both the lines, and then both the bit lines, BL and BL bar, we set their voltage to 0.5 volts. And then, we connect the SRAM cell. After that we just sense the difference in the voltage, and it sets the output once the difference crosses the threshold delta.

So, what do we need to do? Well, we first need to precharge it to 0.5 volts. This can be done very quickly because in this case a sense, a small SRAM cell is not driving it, instead we will have this fat precharge driver circuits which can quickly set its voltage to 0.5 volts. And after that the teeny weeny weak SRAM cell is connected, and we just sense the difference. So, if let us say the difference is 100 millivolts. So, the moment that this reaches let us say 550 millivolts, and this let us say reaches 450 millivolts, and we realize that, hey, look, the difference is crossing the threshold we declare a logical 1.

So, this is much faster. In the sense, we do not have to wait for the voltage to actually reach a 1000 millivolts which is 1 volt. Hence, we have we gain performance. And we gain performance significantly. It is not a small gain. So, it can very well become 5 or 10 times faster just by using this trick.

(Refer Slide Time: 26:42)



The Sense Amplifier

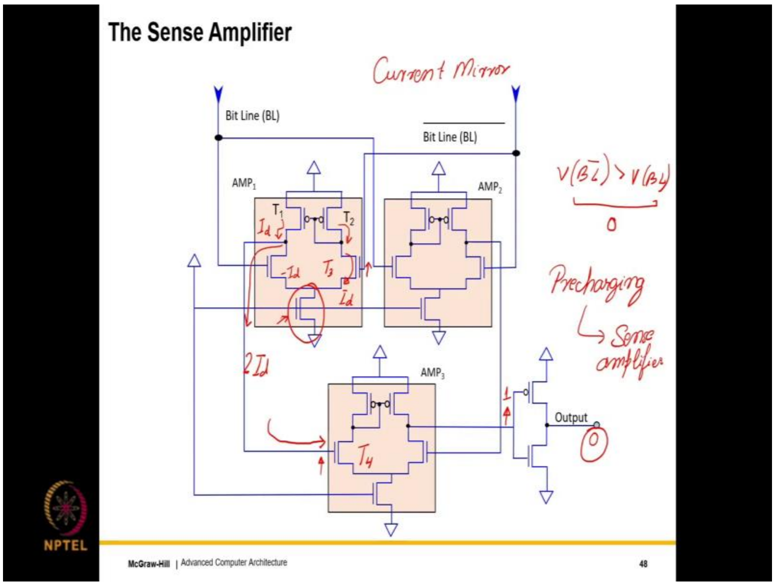
McGraw-Hill | Advanced Computer Architecture 48

So, let us discuss more about the sense amplifier. So, I will not discuss much in the sense that this is out of scope. This is just to give you an idea of the elaborate circuitry that goes into making a sense amplifier.

So, I will explain the broad insights. So, what you see in this figure is that you see three shaded squares each of them is a differential amplifier. So, each of them its job basically is to amplify the difference in the voltages. I will tell you in a second what those voltages are. So, this voltage is the bit line voltage, and then this voltage is the bit line complement voltage.

Let us now see what happen if the voltages on the bit line or the bit line complement deviate from their precharge values.

(Refer Slide Time: 27:46)



The Sense Amplifier

McGraw-Hill | Advanced Computer Architecture 48

So, let us assume that we have a position where V BL bar becomes greater than V BL, right. So, there is an increase in the voltage over here which means that this transistor over here this one becomes more conductive. So, this particular circuit that you see over here, the transistors T 1 and T 2, this is a classic organization is called a current mirror.

So, what a current mirror basically means is that whatever is the current that is flowing through this branch the same current flows through the other branch. Now, given that this transistor over here let us call it T 3, has become more conductive. An additional current will flow, let us call this I d. Because an additional current will flow through this branch, a similar additional current will also flow via this branch.

But, the interesting part is that if you take a look at this transistor over here, this transistor is in saturation. So, what it means is that the current that will flow through this transistor is constant, given that it is constant, and one of the components is increasing I d, the other component has to decrease by I d.

So, there are two things happening first is that because T 3 became more conductive additional I d milliamperes flow through it. And the other thing that is happening is that because of the current mirror effect additional I d milliamperes are flowing through T 1. Additionally, there is a short fall of - I d milliamperes because the sum has to remain constant at this transistor.

The only way that this situation is possible is if I d + I d, 2 I d milliamperes, they flow via this branch over here into the gate of this transistor which let us call as T 4. So, what is happening over here? What is happening is that because the voltage at V BL bar increased, we are seeing an additional 2 I d units of current flow into T 4.

So, of course, this is a transient phenomenon. It is not a steady state phenomenon. But nevertheless, the effect is that the voltage here will increase. If the voltage here will increase so, we can see something similar over here, the voltage here increased.

And then what happened? The voltage at the output increase. So, the voltage increases over here the voltage will increase over here. And given that this is an inverter, ultimately, the voltage will move to 1. So, the output will move towards 0. Which is anyway what we expect because when V $\overline{BL}$ > V BL. In any case, the cell stores a logical 0, which is exactly what we get to see over here that the output gets driven to 0.

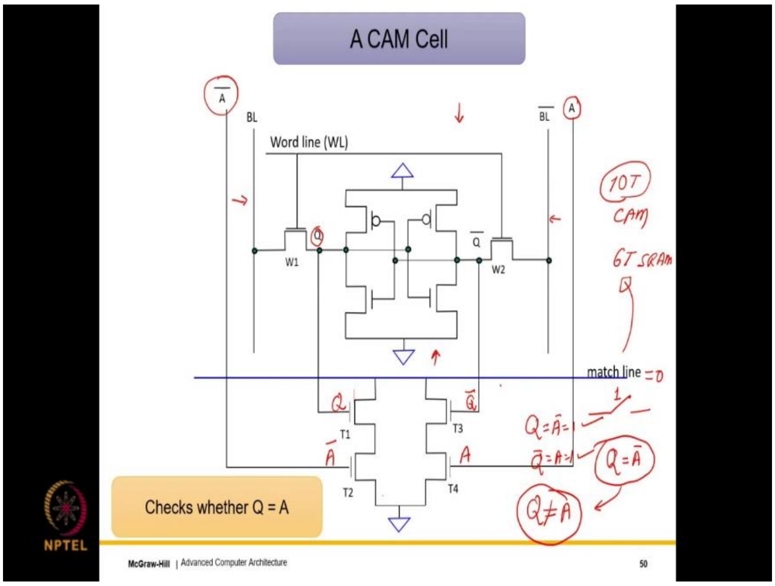
So, we will have a similar effect over here. Of course, a V BL > V $\overline{BL}$ the output will be driven to 1. And again, all three of these differential amplifiers will come into play. But, what we see is that we can adjust the parameters of the circuit. What kind of parameters? Well, the dimensions of the transistors, and the resistances of the wires, and so on, such that if the difference in voltage is slightly greater than the noise threshold, then the output over here will either become a logical 0 or logical 1. And you already saw it becoming logical 0, and a mechanism was explained.

So, sense amplifier as such as a pretty elaborate circuit, but since we do not have very many of them, we can afford to have such an elaborate circuit, such that a small difference in voltage can be amplified and you can have 0 or 1. And we can use a precharging trick.

So, the key thing is that whenever you have a precharging, you require a sense amplifier to essentially amplify the difference.

This is the key, key, key, most important concept in SRAM design. We will also find that this is also very important in DRAM design, Dynamic RAM cell design. And the DRAM design will use sense amplifiers. I will bet their design will be not like this, it will be different. But, the basic concept is still the same. That, you precharge and then you look at deviations from the precharge voltage.

(Refer Slide Time: 32:47)



A CAM Cell

Checks whether Q = A

McGraw-Hill | Advanced Computer Architecture

50

We have discussed the SRAM. So, let us get into the it should be an CAM array. Now, this is what happens when you copy paste slides. So, I will advise all of you not to copy paste slides. If so, do not do it the way I am doing. Use your brain.

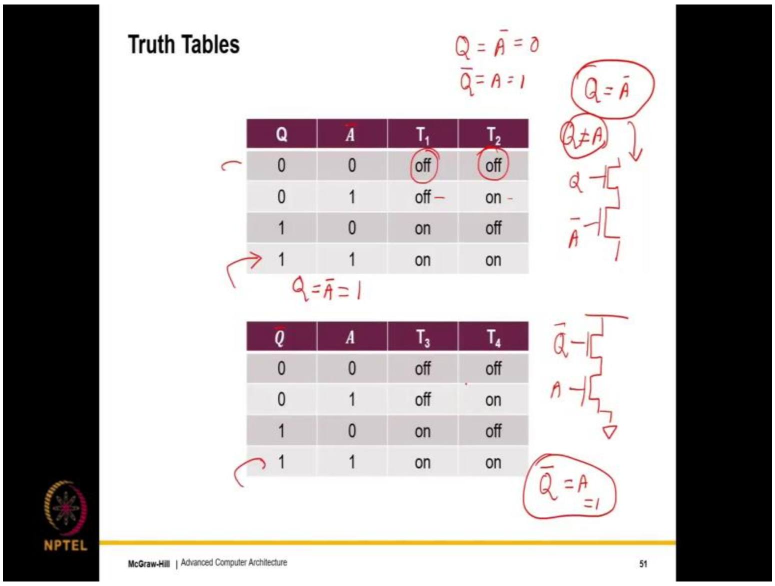
Apologize for this mistake. So, we have a CAM array. So, CAM is a Content Addressable Memory. A Content Addressable Memory at its heart is an SRAM cell plus slightly more. So, we recognize the top part above the match line. It is the regular 6 transistor SRAM cell which by now we totally understand, we are experts in it. And we understand everything that is happening between these two arrows. We understand everything that is happening between these two arrows. See, we understand this completely.

So, I do not want to draw. Let me draw a big circle to clutter up the diagram, and then I what I am going to do is that I am going to erase the big circles as a diagram becomes uncluttered. Now, the important point is that I have an additional bit A, which is an input bit, and I would like to check if the value stored in the SRAM cell which is Q = A or not. If it is = A, I would like to do something. If it is not equal to A, I would like to do something else.

So, what will I do if it is equal to A? Here is what I am going to do. So, the additional A, the value that I would like to compare because recall that a content addressable memory, you compare the contents of the cell. So, one way to compare is we use a traditional comparing circuit which is essentially a XOR gate. We can use a XOR gate, but a XOR gate would require more transistors. And also, a XOR gate is not very efficient when comparing several values. So, we will use a different technology. But, let me tell you what it is.

So, first we input A and A complement both. So, this value over here which is coming as A complement, this is Q, this is Q complement and this is A. So, every transistor as I said is a switch. You send it 1 the switch closest, you send it 0 switch opens. So, the thing is that we need to understand under what conditions is there a conducting path from the match line to the ground, which means the match line will get set to 0 volts.

(Refer Slide Time: 35:37)



Truth Tables

$Q = \bar{A} = 0$
$\bar{Q} = A = 1$

$Q = \bar{A}$

$Q \neq A$

Q	$\bar{A}$	T_1	T_2
0	0	off	off
0	1	off	on
1	0	on	off
1	1	on	on

$Q = \bar{A} = 1$

$\bar{Q}$	A	T_3	T_4
0	0	off	off
0	1	off	on
1	0	on	off
1	1	on	on

$\bar{Q} = A = 1$

Q
$\bar{A}$
$\bar{Q}$
A
NPTEL

McGraw-Hill | Advanced Computer Architecture 51

So, we will keep on coming back to this slide. Let me show you the next slide. So, here I should maybe have a slightly longer one.

So, let us look at these values once again, alright. So, let us take a look at the first row. So, let me assume that both Q and A complement both are 0. See, both of them are 0, both these transistors are off. So, there is clearly no path from here to here.

So, let us say that there is 0 and 1, then also one is off, other is on. So, even if one is off, there since they are in series, so since you know these are in series. So, basically and other one is also in series. What we want is that if there is a conducting path from ground to the match line, both the transistors have to be on and what we see is both of them are on only when either Q = $\bar{A}$ which is this condition where both match or $\bar{Q}$ = A, which is again the same, when both are equal to 1. So, that is when there is a conducting path.

So, which we can clearly see over here, that if $\overline{Q}$ is also 1 and A =1, and both are equal to 1. Then, there is a conducting path or when Q = $\bar{A}$ = 1 there is a conducting path. So, basically, I should write it as this. So, this will kind of clarify things.

So, here is the important point. So, the important point is that if let us see Q = $\bar{A}$ = 0, automatically you can see that $\bar{Q}$ = A = 1. And similarly, you can say the vice versa. So, which basically means that if Q is equal to the complement of A, only then do we have a conducting path which means if Q ≠ A, if Q and A are not the same, then only we have a conducting path, otherwise we do not have.

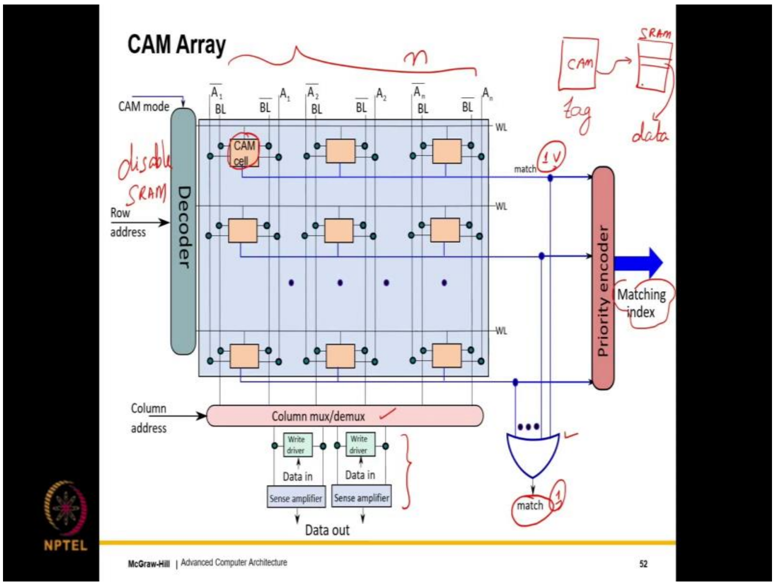
So, this is the most important insight over here that look the way that we have organized this, the conducting path will come only when both of them are equal to 1 which means either Q = $\bar{A}$ = 1 or $\overline{Q}$ = A = 1, any one of them, both cannot be true. But, essentially, if I were to distil both of them, the important point that will come out is that both of them have to be equal. If they are equal, either they are equal to 1, this is satisfied or if both = 0, then this is satisfied.

If they are equal, I can then write this as if Q and A are not equal, given its Boolean way I can write this, if they are not equal only then the match line in this case will be set to 0, otherwise it will retain its previous voltage.

So, let us see, if I precharge the match line to 1 volt it will remain so, unless Q ≠ A. In this case, it will quickly get discharged to 0. So, this is the key idea behind a 10 transistor CAM cell, where we have a 6 transistor SRAM which is a traditional SRAM, nothing new about it, new with it. And in addition, we have this little trick over here with these 4 transistors T 1, T 2, T 3, T 4, which essentially checks if Q and A are the same or not.

If they are the same the match line maintains the same voltage, otherwise it gets set to the ground voltage. So, you keep taking a look at this table. It will help you understand the previous table over here.

(Refer Slide Time: 40:04)



CAM Array

McGraw-Hill | Advanced Computer Architecture

52

But, let me move ahead. So, what we can do is that we can have the same SRAM array, instead we replace an SRAM cell with a CAM cell. And of course, there is some additional circuitry in the sense along with the bit line and bit line bar, we also input the bits, that we would like to compare.

So, if let us say there is a there are n elements over here in a row, we put in n elements. And then what we do is we simply compare. So, our job over here is to simply compare. The way that we do it is that we precharge the match line to 1 volt. Even if there is one mismatch, the match line will get discharged to 0 volts.

So, even if there is one mismatch, the match line over here will get discharged to 0 and you will know that there is no match. So, what we can do is that in our decoder, we can have two modes. One can be the regular SRAM mode, where we do not use the CAM functionality and this can be used for example, to populate data, to read out data, non-CAM functionality.

So, in this case what we do is that we disable these A 1, A 2, A n lines. They do not matter. And we do not care about the match lines. And we just access them as regular SRAM cells. And recall, we had a column mux, demux there also, we have one here also. And we have the right driver in the sense amplifiers. So, all of these remain the same.

What actually varies or differs in this case is that in a CAM mode, What the decoder will do is it will disable all the word lines, so it will essentially disable the SRAM functionality. It will totally disable the SRAM functionality, and what it is going to do is that we will precharge all the match lines.

After we precharge all the match lines we will allow the match lines to be set, and then if let us say we do not have multi; if you do not have multiple copies only one of them will remain 1, at most one of them. So, then what we can do is that we can have an OR gate, and if one of the match lines is 1, then the final match which means whether a match exists or not, that can be set to 1.

Sometimes we might have duplicates, it is possible. See, if you have duplicates does not matter, this circuit will work as it is, but we will need a priority encoder instead of a regular encoder. So, the priority encoder will basically tell us the index of the match line that has a 1 and the one that has the highest priority.

How is the priority decided? The priority is decided based on other heuristics. For example, it can be the earliest match line or based on some other information, some other state. So, that is not really that important. But, this is where a priority encoder would differ from a normal encoder or a regular encoder, in the sense that if multiple match lines are 1, it will choose that match line that has the highest priority. Howsoever, it is determined. And the output will be a matching index. So, that is the important point over here. That, the output will be a matching index a row index that in which row we have a match.

So, then, this can be used to for further processing. For example, in a fully associative cache, the tag array can be a CAM array. So, we can check the tag. Wherever there is a match, we can send it to the SRAM array, which is the data array, which is made of SRAM cells. So, since we have the exact index, we send it over here, and we get the data. So, the data array will clearly be much bigger, because it will contain 64 byte blocks. So, that can be made using SRAMs.

And the tag array can be made using CAMs. So, CAMs are of course, much slower and require much more area as you can clearly see over here, that each cell is bigger, and also, we need may more space for these wires. So, that is why the CAM array is much bigger and slower and inefficient. But, it can be used for the tag part, and tags are in any case small.

And also, CAM arrays will be used when we have fewer entries. So, once that gives us the matching index, if there is a match that is, we can quickly access the data array made of SRAMS. So, this should give you a give you a broad overview of the CAM and SRAM technologies. And so, they form the basis of modern memory systems.

SRAMS as I said are faster, much faster. So, they are used almost all that almost everywhere for the data part for the data arrays. CAM arrays are occasionally used for the tag arrays, particularly when we have few entries. And there is a need for creating a hash table like structure, where we would need fully, where we would need a fully associative structure.

(Refer Slide Time: 45:52)

The Cacti Tool

Now, the next part is the design of the Cacti Tool. The cacti tool is an automatic cache design space exploration tool which we will discuss next.