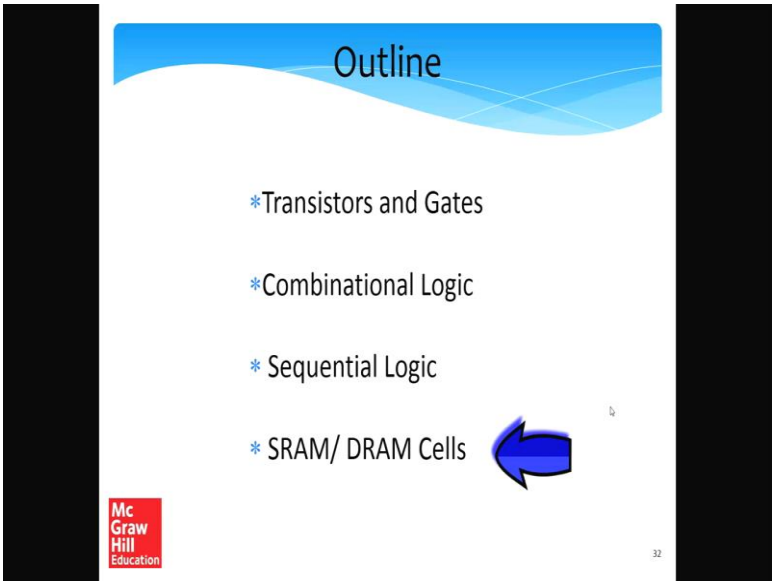
Computer Architecture
Prof. Smruti Ranjan Sarangi
Department of Computer Science and Engineering
Indian Institute of Technology, Delhi

Lecture - 16
A Primer on Digital Logic Part-III

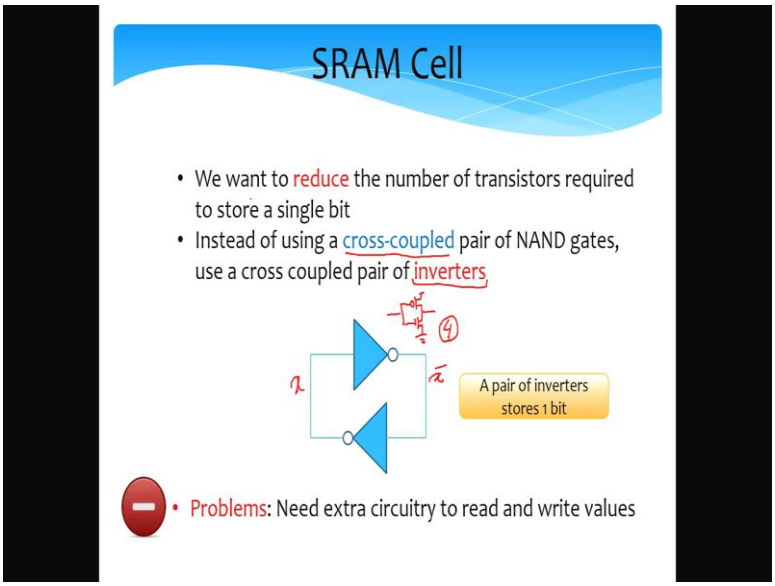
Now let us look at memory cells.

(Refer Slide Time: 00:25)



So, memory is a far denser storage technology than flip flops, and so that is a reason it is used in structures inside the processor to actually store a lot of data, not just 1 bit or several bit is, but 1000s or millions of bits. So, the two kinds of memory technologies that are normally used are SRAM static ram cells and DRAM dynamic ram cells.

(Refer Slide Time: 00:56)

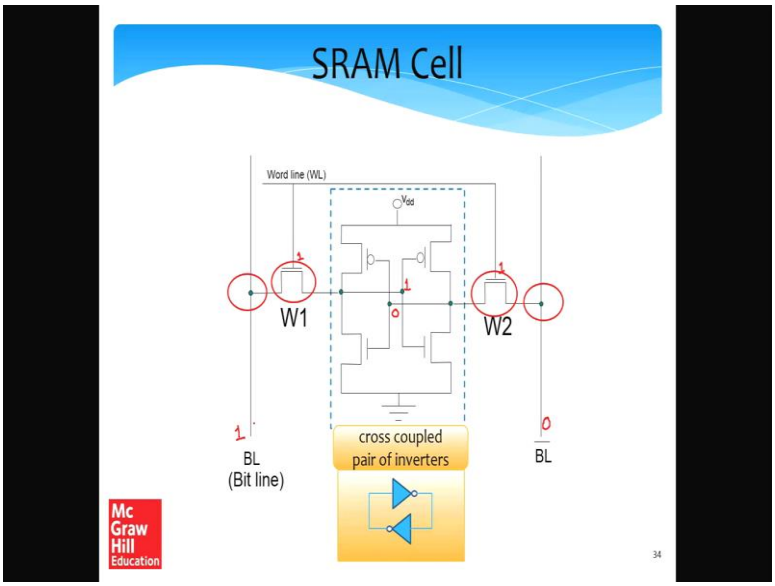


So, in the SRAM cell works like this that. So, what is the main philosophy here? The main philosophy is, that we want to reduce the number of transistors that are required to store a single bit, because we will be saving millions of bit is, possibly billions of bit is. So, that is the reason we need to keep the number of transistors as low as possible. And we also want to provide a memory technology; that is as fast as possible. So, what we do, is that to reduce the number of transistors, instead of using cross coupled NAND gates, as was the case in flip flops right. We use a cross coupled pair of inverters, which does the same thing, but we need some additional circuitry as we shall see.

So, in this case this is an example, of a cross coupled pair of inverters. So, the output of one inverter draw, is connected to the input of the other, and similarly the output of the lower inverter is connected to the input of the upper inverter. So, let us say value x is stored here, then the complement of x is stored on the other side. So, this structure by itself is potent enough to store the value of one bit, and how many transistors does this requires, if you would recall the structure of an inverter was, that it required two transistors; a PMOS transistor and an NMOS transistor.

Since, one inverter requires two transistors we need a total of 4 transistors in this structure, but however this much of circuitry is not enough, to read and write values into the structure. So, what we need to do is that we need to have something else, to actually read and write into the structure.

(Refer Slide Time: 03:01)



So, the something else is like this. So, we let us first consider or cross coupled pair of inverters, which is here at the bottom and. So, here as shown in the previous diagram we have 4 transistors one 2 3 and 4, and, but let us sort of isolate this as a black box. So, we will need two more transistors. So, what do the transistors do? So, we have one transistor w 1, and one transistor w 2. These are simple NMOS transistors. So, these are called word line transistors. So, when the word line is set to 0, at that point the NMOS transistors are off.

So, it is like that this memory cell is disconnected from the rest of the world. So, it is not possible to read it is values, neither is it possible to write it is values. So, the value that is stored inside the memory cell will be maintained. On the other hand, if the value of the word line transistors is equal to one, then this transistor is on. So, this line gets connected, and pretty much what we see is, that this 4 transistor cell is connected to the outside world, it is possible to both read and write it is values

So, let us consider writing first. So, since in a we have two inverters whatever value is saved here, the complement of the value will be saved on the other side. So, it is a symmetric structure. So, one copper line connected to one side is called a bit line, and the other one is also a bit line, but it stores the complement of the value. So, both of these copper wires are called bit lines. So, one stores a value, and the other stores the complement or the value stored inside the cell. So, if you to want to write, let us say one

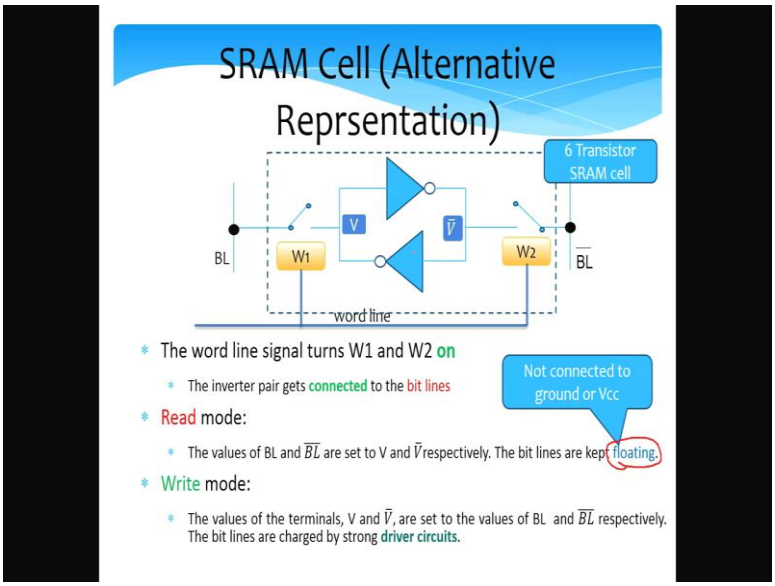
into the cell, we set this bit line to a logical one, which is the supply voltage, and we set this bit line to a logical 0 which is ground. So, in this case, if the voltage drivers are strong enough, then the 4 transistor cross coupled pair of inverters will be programmed; such that the value saved over here is one, and the value saved at this point which is also the same as a value store in this points right is 0 right, and here it is one.

Similarly, if you want to save 0 in the 4 transistor SRAM cell, I am sorry in the 4 transistor cross coupled pair of inverters, what we need to do is that we need to set; of course, we need to set the word lines to one, to enable the SRAM cell. Then we need to store is 0 over here, in a set drive the bit line 2 a 0 BL to 0, and BL bar we need to drive it to a one. So, that will set the values, the voltages at these points to 0, and the voltages at these points to one. So, writing is easy, reading is also easy.

So, in this case what we do, for the purpose of reading, is that we enable the transistor by setting the word lines to one, and we do not use any circuitry to explicitly drive the voltage of BL and BL bar, instead we allow their voltage to become the same, as the voltages that are stored inside the cell. For example, if this voltage is a logical one and this is a logical 0. So, overtime the voltage of BL will become a logical one, and the voltage of BL bar will become a logical 0, and thus we can infer that one is stored inside the cell; that is how we will read. So, what are the important points that need to be taken care of?

So, our basic cross coupled pair of inverters had 4 transistors. Now we have two word line transistors, 6 transistors are there now in each cell, and then these are connected to two bit lines.

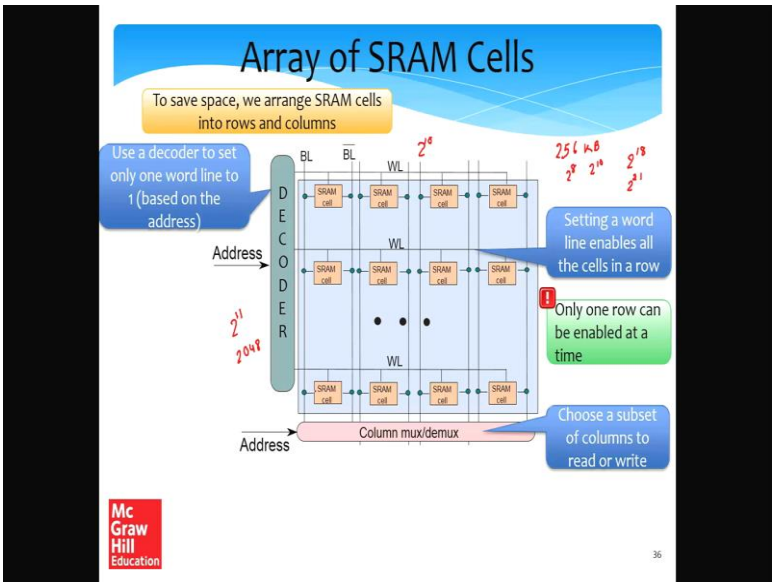
(Refer Slide Time: 07:46)



So, alternative representation of an SRAM cell looks something like this, where we have a pair of inverters, and the voltage at one side is V and the voltage at the other side is a compliment of V. So, then the word line transistors can be treated as switches, and this entire 6 transistor cell called an SRAM cell. It is a static, it is called a static ram cell we will see why, when we discuss dynamic rams. So, they drive the voltage of BL and BL bar. In the read mode the bit lines are gets floating.

So, what is floating mean? Floating means that they are not explicitly connected to either the supply voltage or the ground voltage, so we allow the voltages V and V bar to propagate, to set the values of the bit lines BL and BL bar. So, that is how we can read the contents. For writing what we do is that we again enable the word lines, and then we explicitly set the voltages of BL and BL bar; such that this 4 transistor SRAM cell gets appropriately programmed

(Refer Slide Time: 09:10)



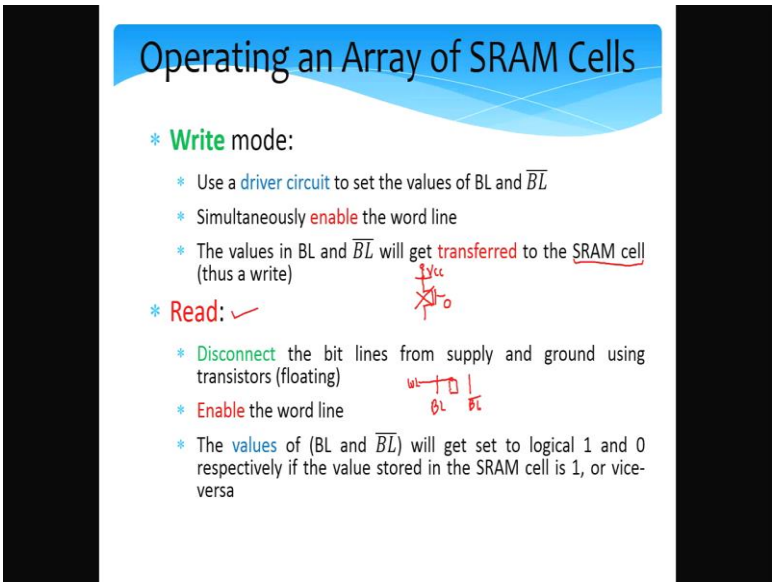
Now let us create an array of SRAM cells. So, to create an array of SRAM cells, let us see what needs to be done. So, what we do is, that we create one SRAM cell, and we arrange them in a matrix form. So, let all the SRAM cells on a single row, be connected via the same word line. And let the rest of the SRAM cells along a column be connected to the same pair of bit lines. So, the first thing that we do, number one; you know this is like step number one. So, let us assume that we have 128 rows. So, we need to enable one of the rows. So, how do we do that, we use a decoder to set only one word line to one, which is true, based on the address and the rest are 0.

So, let us assume that each row can store one integer which is 4 bytes. So, since 4 bytes is 32 bit is, we will need 32 SRAM cells, and let us assume that there are hundred 20 8 such rows. So, the address of 128 is 2 to the power 7. So, what can be done is that we can send a 7 bit address. The 7 bit address can be used in the decoder, to set one of the word lines to one, and the rest will remain 0. So, let us assume that this word line get set to one, which means that this entire row gets enabled, and the rest of the SRAM cells in other rows are disabled.

So, now what we do is that this particular row after it is enabled. So, mind you only one row can be enabled at a time. So, after doing that, all the SRAM cells are enabled, and they will be setting the values of BL and BL bar. Subsequently what we can do is, either we can consider the values in all the SRAM cells, or let us say we want to read only two

bytes. If you want to use to bytes we can use pair of multiplexers; such that for this example if 4 bytes are coming in, we can choose the two bytes that we need to read, and then this can go out. And same is the case for writing as well, if you want to write to 2 bytes, then we will have driver circuit is, that can help us, that can essentially help us set the voltages, as we shall see of BL and BL bar, and then the multiplexer circuit over here, will enable certain SRAM cells; such that we can read or write them.

(Refer Slide Time: 12:30)

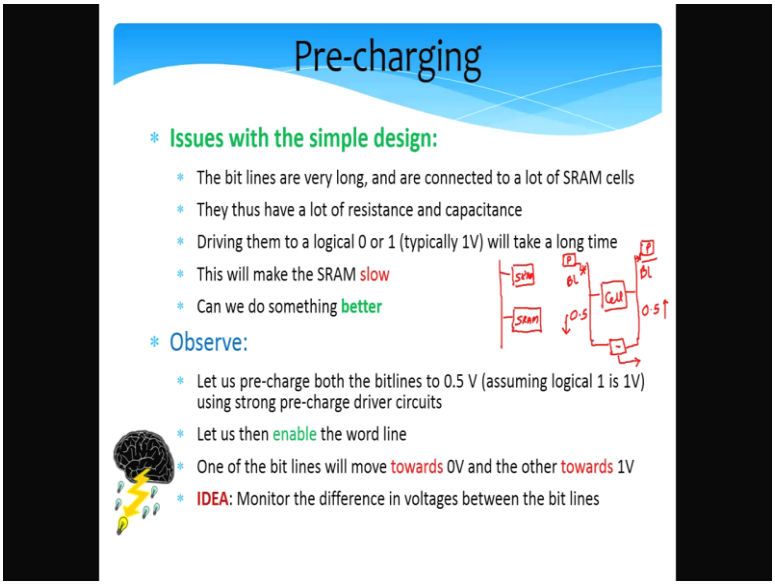


For example if you just want to enable this and this, then the multiplexer will only ensure that the outputs of these cells actually flow out of it. So, how do we operate an array of SRAM cells? So, an array of SRAM cells will have two modes; a write mode and a read mode. In the right mode we will use a driver circuit, to set the values of the pair of bit lines BL and BL bar. Simultaneously we need to enable the word line from the decoder. The values in BL and BL bar will get transferred to the SRAM cell, thus affecting a write. In the case of a read what we need to do, is the first step, disconnect the bit lines from supply, from the supply voltage and from ground, using transistors.

So, basically what we can do is, we can have a circuit of this type, where this is the supply voltage right, this is V c c the supply voltage, and we will have a transistor. So, here if we set the gate voltage of the transistor to 0, essentially this line is disconnected from the supply voltage. So, we say that it is floating right. If a certain copper wire is disconnected from supply, as well as the ground we say that the wire is floating, then we

enable the word line. Once we enable the word line. So, there is pair of bit lines, then there is the SRAM cell here, in the middle, and then there is a word line. So, once we enable the word line the SRAM cell gets enabled, and the values of BL and BL bar get set to logical one and 0, depending upon the value stored in a SRAM cell.

(Refer Slide Time: 14:17)



So, there are some issues with this is, you know very simple design the first is. So, let us go back, let us actually go back to this particular slide one again, remove all the annotations and look at it once again. So, we have an array of SRAM cells. So, mind you this array can be very big. So, the array can have literally 1000s of rows. So, if you want me to give a typical example. So, let us assume that I have a 256 kilobyte SRAM array. So, 1 kilobyte is 2 to the power 10, and 256 is 2 to the power 8; so total there are 2 to the power 18 bytes. So, the number of bit is would be 2 to the power 18 times 8, which is 2 to the power 21.

So, even if you try to make a nice structure, which is sort of looks like a square, and we have 2 to 2 to the power 11 rows here, and 2 to the power 10 columns here. We will still have 248 rows. So, which means that one SRAM array is a very big structure, and to each bit line which is a thin copper wire, 2048 SRAM cells are connected. If that is the case, this long wire will have a lot of resistance as well as capacitance, and the RC delay. See we recall what you are taught in first year, if a wire is very long and a wire is connected to a lot of other elements, it is resistance and capacitance will be high. So,

driving the voltage of the wire either towards the ground or towards the supply voltage will take a lot of time right.

So, basically why is the wire long, the wire is long, basically because a lot of SRAM cells, you know literally 1000s of SRAM cells are connected to it. So, this will make the SRAM very slow, and the question is can we do something better. We definitely can do something better, but that start with this idea. So, let us do one thing, let us consider the simple SRAM cell, which is 6 transistors, which is connected to two very long bit lines, they are called BL and BL bar. So, it takes a long time to charge both of these bit lines to either 0 or 1.

So, let us do one thing, let us pre charge both the bit lines, let say a 0.5 volts, where the assumption is that a logical 1 is 1 volt, using very strong pre charge driver circuit is. So, what we do is, that before we begin reading. So, mind your reading is slow, writing is not slow. The reason writing is not slow, because we can connect very strong in a right driver circuit is, to the BL and BL bar bit lines right. So, d is for a driver; such that you know you can quickly drive them to a logical 0 or logical 1, but that is not the case when you are reading. The reason that is not the case when you are reading is because this small little SRAM cell has to drive the voltages of the bit lines to either 0 or 1.

So, what we do is that before we start to read, we have this pre charge circuit is right, p is for pre charging, which are connected to the BL and BL bar lines, and we quickly drive them up to 0.5 volts, but the assumption is that a logical 1 is 1 volt. So, we drive them to the midpoint. Once the voltage has reached 0.5 volts, then using a transistor, a programming a transistor we disconnect both of these pre charge circuit is, and the lines are now floating, and we enable the word line. So, what will happen is that; so initially t equal to 0, the voltage at both the bit lines is 0.5 volts. Subsequently one of the bit lines will move towards 0 volt.

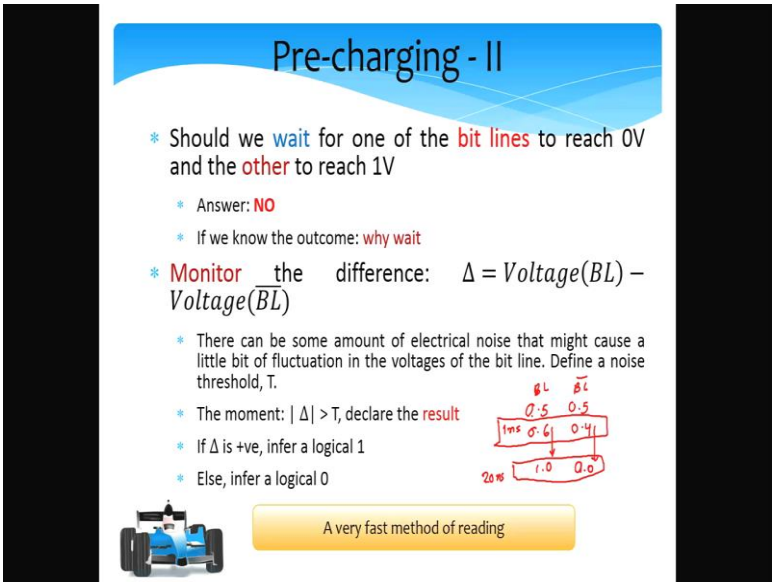
So, let us assume this bit line is moving towards 0 volts, and the other bit line will move towards 1 volt. So, now, the question is that if we know that ultimately one bit line will reach 1 volt and the other one will ultimately reach 0 volts, the moment we have made this determination. We do not really have to wait for the bit line to reach 1 volt and the other one to reach 0 volts right. So, it is similar to the fact that assume a student is not doing well, and a courses let us say 5 exams, and after the first two exams this student

has found out, that you know there is no hope of passing the course. It is not necessary for the student to sort of wait till all the 5 exams are over.

The student can at that point withdraw from the course. Say similar in a in a game of chess, you actually do not wait for the king to be killed; long before the king is killed typically one side realizes the game is up and they resign. So, the case is similar here, we start by pre charging both BL and BL bar to 0.5 volts. Subsequently one of the lines will move towards 0 volts, and the other line will move towards 1 volt. So, let us monitor the difference, in voltages between the bit lines right. So, let us have a simple circuit over here, whose only job is to subtract the voltages of V BL and V BL bar right. So, we are monitoring the difference. And in this case you know.

So, say again you know this sort of a repeat of what I said, should be wait for one of the bit line it is 0 volts and 1 volt, the answer is no. So, this is what we do.

(Refer Slide Time: 20:23)



Let us monitor the difference delta which is voltage of BL minus voltage of BL bar. So, there can be a little bit of electrical noise in a system, why is this the case you would have typically seen, that if it turn on some electrical appliance; like a microwave oven or a washing machine, there is little bit of noise in your phone. So, that is basically because a little bit of inductance happening. So, it is possible that inside an SRAM also we have a little bit of electrical noise. So, let us define a noise threshold, and let us call it t. So, this is the amount of random noise that can come on a copper wire. So, the moment the

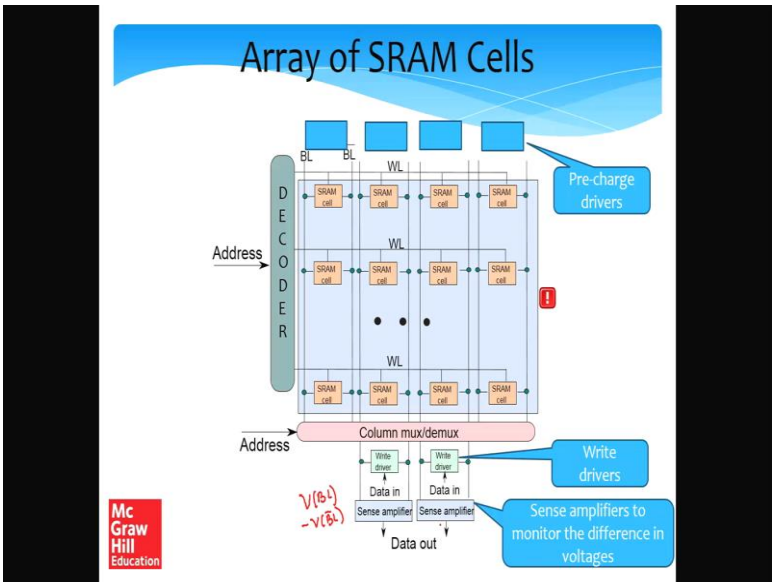
absolute value of the difference in voltages, is beyond that noise threshold, which means say this is a real difference is not a random difference.

We can declare the result right. Say the voltage of BL. So, assume that we start at 0.5 and 0.5, after let us say 1 nanosecond. Let us see that the voltage of BL becomes 0.6 volts and voltage of BL bar becomes 0.4 volts. So, at this point we will be very sure that 0.6 will ultimately go till 1. Similarly at this point we will be very sure that 0.4 will ultimately go till 1. So, is also the same as declaring the results of an election? So, let us assume that there are two candidates, and one candidate is just racing ahead. So, see at the beginning, we will not be able to say, because the candidates might be going neck and neck, and there might be some noise in the counting process. So, maybe you know the votes for one candidate are sort of disproportionately mixed up, but gradually as a time passes, clearly one candidate will emerge to be the winner. And once we are sure we can pretty much declare one candidate of the winner, and in a sense not wait for all the votes to be counted. So, it is very similar over here.

So, let us assume that the amount of noise that the system will introduce is very small. So, maybe from 0.5 when it reaches 0.6 and the other one reaches 0.4. We can be sure that in this case BL will reach logical 1 ultimately, we do not have to wait and BL bar will reach, I am sorry this should be logical 0, this should be logical 0, it will reach 0 volts ultimately, and we do not have to wait. So, what we can do is that at this very point that 1 nanosecond we can declare the result.

So, this is a very fast way, as a very standard approach in hardware, where what we do, is that we instead of waiting for the bit lines, one bit line to get fully discharge, and one bit line to get fully charged, we monitor the difference, once we are sure we declare the result. In this case is the difference is positive in for a logical 1, otherwise we infer a logical 0 is a very fast method of reading, because assume that for one line to reach logical 1 and the other to reach logical 0, it would have taken like 20 nanoseconds. In this case in one nanosecond we have the result.

(Refer Slide Time: 24:15)



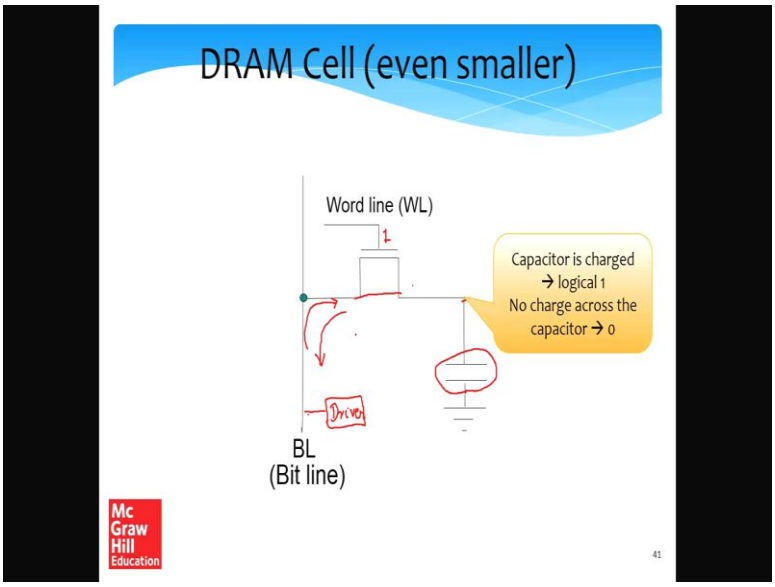
So, what does an array of SRAM cells now look like, we have our previous diagram. The only additions are as follows. So, we have a set of pre charge drivers, as you can see on the top. So, what this pre charge drivers do, is that they help in pre charging the bit lines before you read subsequently after the column multiplexer. So, let me just repeat what a column multiplexer does, is that it helps you choose a subset of columns that you want to read or write, because typically the array is very large, we do not want to be, we do not want the array to be like one long strip, we wanted to be nicely organized as a square, for certain circuit design considerations.

So, that makes a necessary to choose a subset of SRAM cells within each row right, a subset of columns within each row. So, that is the reason we have a multiplexer over here, and this is connected the pair of bit lines comes out. So, for the case of a write we can use these write drivers which are very powerful circuit is, to set the values of the bit lines to 0 or 1. For the case of a read we have this simple circuit called a sense amplifier. So, the role of the cell sense amplifier is basically to monitor the difference between the voltages of the one bit line minus the voltage of the other bit line, so it just monitors the difference. Once the difference increases in acceptable threshold, it declares the result, either a logical 0 or a logical 1.

So, what are the only additions in this most important circuit? Well, we started out with an array of SRAM cells, so the decoder to choose the right row, enabling the word line.

We also have the column multiplexer de multiplexer to basically choose a set of columns. The additions are the set up pre charge drivers that help pre charge the two bit lines; a set of write drivers in this diagram to basically set the voltages of the bit lines, and some very specialized circuit is call sense amplifiers, that help monitor the difference in voltages between the bit lines; such that we can declare a logical 0 or a logical 1, depending upon which way the difference is going.

(Refer Slide Time: 26:57)



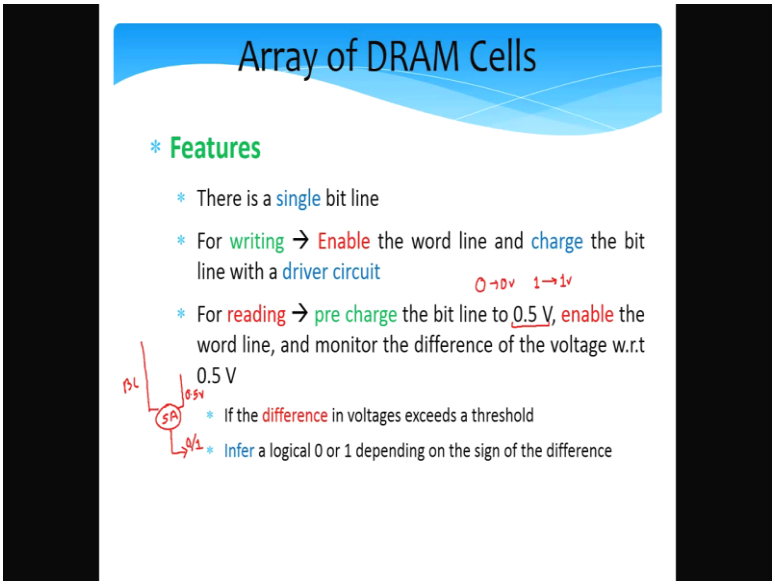
Now, let us considered a DRAM cell which is even smaller than an SRAM cell; so smaller is the cell more compact it is. So, for the same area, more cells can be packed. So, the memory can be more dense, but also smaller is the cell, the lesser is this power to actually drive the bit line to 0 or 1. So, in a sense is also slower. So, that also needs to be kept in mind, that latches and flip flops are very fast, because they are more transistors, and as a result it is easier for them, to program them you do not have bit lines. SRAMs are still fast enough, but DRAMs are so slow and we shall see why.

So, the reason for this is that instead of having a cross coupled pair of inverters, the main circuit in a DRAM is a capacitor. So, charge is stored across the capacitor. So, if the voltage across the capacitor is close to the supply voltage, we can infer a logical 1, but if the voltage across the capacitor is 0, which means at this point here is close to a logical 0, the potential is closed to 0 volts, we can infer a logical 0. So that is the only structure, there is no cross coupled pair of inverters.

So, we shall see that there is, you know there are good aspects to this and there are bad aspects to this, but this is pretty much the only structure inside the DRAM cell a single capacitor, and the way that we control access to it is that we have a single transistor here, which is controlled by the word line. So, the word line is set to one, we can assume that the switch is connected, and the capacitor is directly connected to the bit line, and here also there is a single bit line, instead of two bit lines there is a single bit line.

So, we infer the value of the voltage stored across the capacitor by taking a look at the voltage of the bit line. Also for writing what is done is that, when we write we connect a driver circuit to the bit line, write the connection is again done with the help of a transistor switch, and the driver circuit can do one of two things. So, if it pumps in charge into the capacitor, and it is potential increases, then we are trying to write a logical 1. Otherwise if it drains out charge from the capacitor, then we are writing a logical 0. And once the word line transistor is disabled, disabled basically means that we set it is voltage to 0, and then the capacitor is disconnected. You can sort of thing that the switch is disconnected. So, whatever is the voltage across the capacitor will remain

(Refer Slide Time: 30:09)



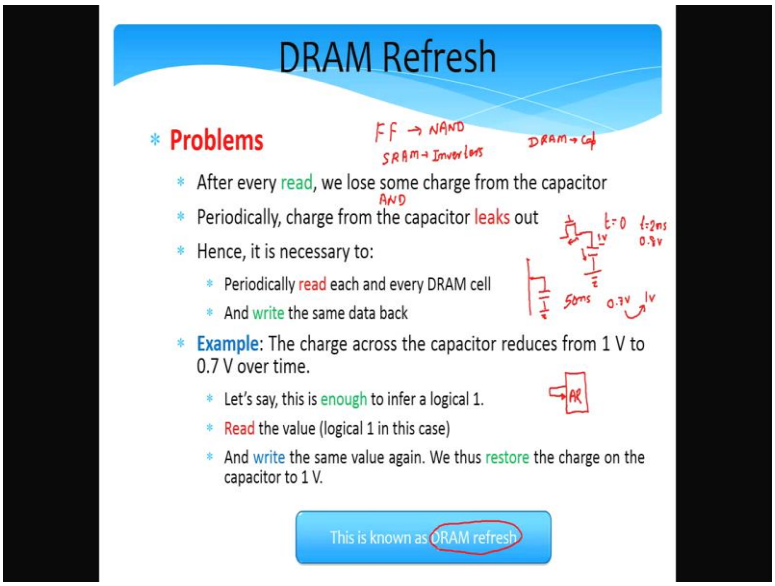
So, now let us consider an array of DRAM cells. So, this is very similar to an array of SRAM cells as we had seen. There is a single bit line right. So, that is one difference, in an SRAM array they were two bit lines is the single bit line. For writing we enable the word line, and charge the bit line of the driver circuit, and then the bit line transfers the

charge to the capacitor. For reading we do something very similar, to what we are done in the case of SRAMs. We pre charge the bit line to 0.5 volts.

So, fair again the assumption is that, a logical 0 is 0 volts, so which this assumption is most of the time correct actually. Now it is most of the supply voltages are closed to 1 volt. So, we pre charge the bit line to half of the supplied voltage, which in this case is 0.5 volts. We enable the word line, and we monitor the difference of the voltage with respect to 0.5 volts.

So, basically this is the bit line. So, we have a simple, I am sorry I should not have added the BL bar, just a bad habit from the SRAM (Refer Time: 31:24) 1 second. So, if this is the bit line, we will have a sense amplifier, which is essentially a comparator, and the other signal that is going into it, is a fixed signal in this case of 0.5 volts. So, we essentially monitor the difference, and the idea is the same. Once the difference exceeds the threshold, we declare the result, whether it is a logical 0 or a logical 1. And what is the threshold, the threshold is expected to be a small value of the order of several millivolts. So, that will be the amount of noise that is introduced in the wire, because of external electromagnetic fields that is all. So, that is the only difference that instead of comparing BL and BL bar, we compare a single bit line with another fixed voltage, which is 0.5 volts, and we see whether the voltage is going up shooting above 0.5 are going below, and based on the direction we infer the value stored in the DRAM cell.

(Refer Slide Time: 32:38)



So, there is a problem. So, basically you know we initially had a cross coupled pair on nand gates. So, in a flip flop, if you would recall in a flip flop, we are a cross coupled pair on nand gates. So, since there are more transistors it was possible to drive in more current, there are more paths from the ground and V c c. And also it was easier to program these devices in comparison in a SRAM cell is a cross coupled pair of inverters, and you have bit lines and so on which make it slow. And even slower or DRAM cells which have a single capacitor, but there is a problem with it. So, the problem is that after every read operation. So, let us assume the capacitor stores a logical 1; we lose some charge from the capacitor.

So, this mean also right; so this should be we lose some charge definitely. So, charge is shared between the capacitor in the bit line, and on a regular basis you know charge from the capacitor leaks out right on a regular basis. Hence it is necessary to. So, what will happen? So, what will happen is that let see at t equal to 0 if we store, let us say it in a time equal to 0 we store 1 volt across the capacitor. What will happen is that may be at t equals 2 nanoseconds. A little bit of, see no dielectric is perfect. So, you would hope that you know between the two parallel plates of a capacitor is no leakage, but there is always some amount of leakage wire the capacitor, little bit of current, and even the transistor when it is turned off, is still a little bit of leakage.

So, a little bit of current that flows out. So, maybe after t equals 2 nanoseconds the voltage can be 0.8 volts, and maybe after like 20 nanoseconds or 50 nanoseconds, or you know 1 microsecond you will find that the entire voltage across the capacitor is disappeared. So, this will cause an error; that is the reason it is necessary to periodically read each and every DRAM cell, and read the data back, and I am sorry write the same data back.

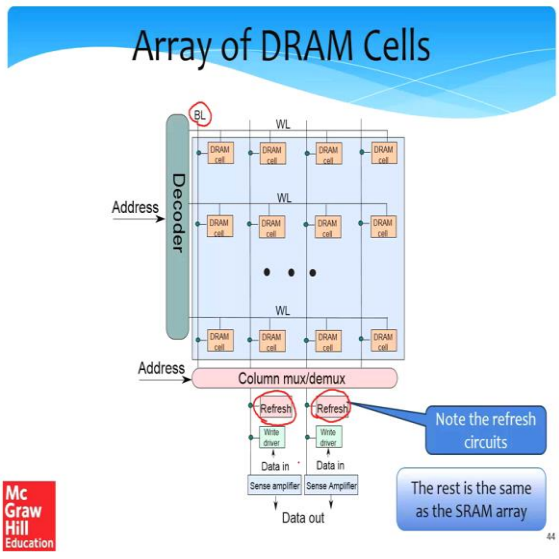
For example if you write 1 to a DRAM cell, and let us say that after 50 nanoseconds we read it, and find that the voltage across the capacitor is 0.7 volts; that is fine we can still infer a logical 1, but we will also be aware that the voltage has decreased. So, what we will do is, we will write a logical 1 back again, and in this case the voltage across the capacitor will again become 1 volt. So, this sort of you know periodically compensates, for the charge that is leaking out of a capacitor, because dielectrics are not perfect ,and also if there is a read operation then basically this charge will get shared, between the

long bit line and the capacitor right, and since a charge get shared the voltage across the capacitor will decrease.

So, that is the reason it is necessary that after every read, we need to write the data that we write back again, and also periodically we need to read each, and every DRAM cell and write the same data back. So, this process is called restoring the value of a single DRAM cell, and the overall process where we read the entire DRAM array. So, where we read? So, given an entire DRAM array, where we read the contents of the entire DRAM array, and refresh all the contents, is known as a DRAM refresh, is known as the DRAM refresh cycle, where all the contents are red and again they are written back.

So, it is not that we read all of it and then write it back. We read let us say 1 byte and then we write back the same byte, or we read say 64 bytes and we write back the same 64 bytes. So, it is definitely not done at the entire array level at one go, but the aim is that periodically we read some lines, and then again write them back to sort of restore their voltages across the capacitors to the maximum levels, to the highest levels.

(Refer Slide Time: 37:18)

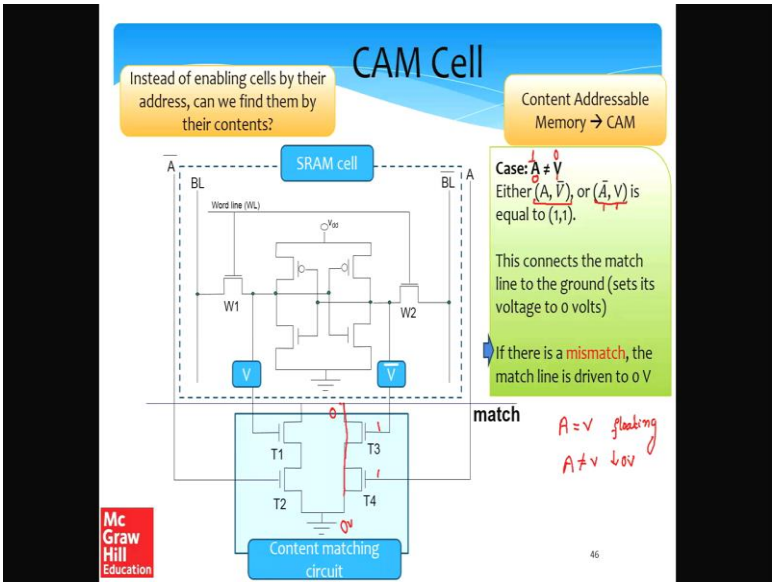


So, here is what an array of DRAM cells look like. It is very similar to an array of SRAM cells. The only differences are that there is a single bit line instead of multiple bit lines. The address and the decoder part are all there, and that the cells are organized as a matrix. The column multiplexer de multiplexer is also there, and addition is the refresh

circuitry right. So, the refresh circuitry what they do, is it periodically read values from each row and write them back right.

So to restore the original voltage levels for writing we have write drivers which are fast circuit is to set the values of the bit lines, and for reading we have sense amplifiers, similar to the case of SRAMs, where we read the values of a DRAM cell out. So, rest everything is same as an SRAM array. The only difference is we have a single bit line, and we have refresh circuit is.

(Refer Slide Time: 38:24)



Now, let us consider one more type of cell called a cam cell. So, the idea is that instead of enabling cells by their address, can we find them by their contents. So, let us go back to the previous slide. So, let us assume that there are only 4 cells in each row. So, let us assume that the problem is, that we want to find the row, that contains 1 0 1 1, and let us for the sake of simplicity assume, that only one row contains this particular value. The question is that can you find out which row it is. So, one simple solution is that we just like a for loop, we read the contents of each row and we compare, we read and compare, we read and compare, till we find the row that is possible, but what is the problem the problem is that we need to read the contents of the entire SRAM array or the DRAM array.

So, which is not a good idea, we can do something better right. So, this is called content base addressing, and a cam cell is called a content addressable memory. So, you are

trying to find it bytes content not bytes address. So, my mind if the address is coming, the decoder is setting one of the word lines to one, and then that row is getting enabled, but in this case, we want to see if there is any row, whose contents are same as the contents that we want.

So, we design a cam cell which is nothing, but a simple SRAM cell plus more. So, this part that you see in this dotted rectangle is a simple SRAM cell. Subsequently what we have, is that we have these additional connections over here, where we get in the values that we want to call them V and V bar, V is the value and V bar is it is compliment, and we get in the values that we want to compare this again. So, this is a single bit, we will compare it with a single bit. So, this value V will go into transistor t 1, and we want to compare value V with another bit value called a. So, A bar, the compliment of A is send to transistor t 2 over here. So, this is basically a compliment of a, and this is the value of v. similarly this is the value V bar complement of V and this is the value A and. So, these transistors are connected in series. So, one side of them is connected to the ground, and the other side is connected to this blue wire called the match line.

So, let us see what it is. So, let us consider the pair A and V bar right. So, transistors t 3 and t 4, and let us consider the case when A is equal to v. So, one of the, if A is equal to V 1 of the values has to be 0 right. So, let us assume that A is 1, and then V bar has to be 0. if A is 0 V bar has to be 1. So, one of them has to be 0. So, either t 3 or t 4 is off, one of them is off. So, essentially this path over here is pretty much disconnected from the ground right, because of any one of these transistors t 3 or t 4, one of them is off. Similarly in the pair A bar and V 1 of the values has to be 0.

So, either t 1 or t 2 1 of them is off. So, you can find out, you can use values. So, if A is 1 and A bar is 0, if A is 0 then since A is equal to V, V is 0. So, one of these values is 0. So, which means one of t 1 or t 2 is off. So, the even this path is turned off, which basically means that for this particular cam cell, which has 1 2 3 4 5 6 7 8 9 10, which is a 10 transistor cam cell, the match line is not connected to the ground right. So, it is in a sense floating, it is not connected to the ground.

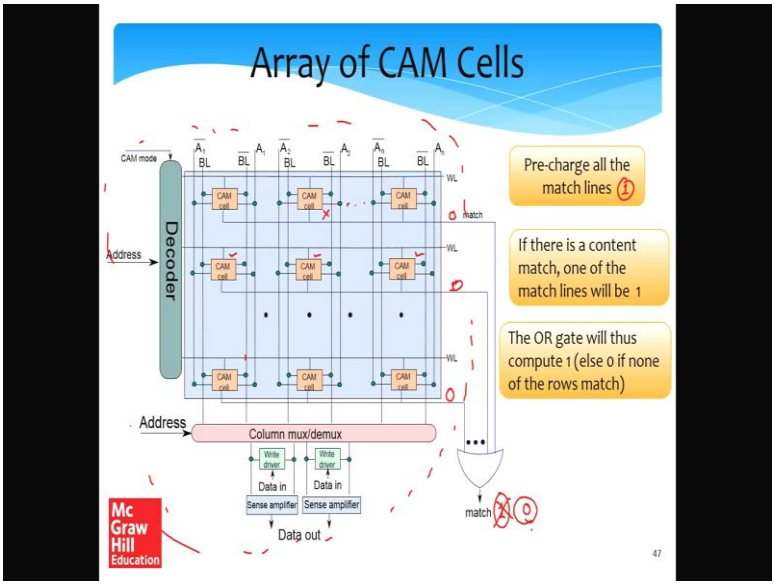
So, this is what we get to find just to summaries that when A is equal to v, the match line will be disconnected from the ground terminal fine. So, let us see what happens when A is not equal to v. So, when A is not equal to v, let us see what is the case. So we say, let

us assume what we claim, is either a comma V bar or A bar comma V is equal to 1 comma 1. So, let us see why. So, let us assume A is 1 and V is 0. So, in this case A is 1 and V compliment is 1. So, this pair is equal to 1 and 1, assume that A is 0 and V is 1. So, in this case A bar is 1 and V is 1.

So, basically irrespective of the value of a 1 of the pairs a V compliment or a complement and V 1 of the pairs is equal to 1 comma 1. So, with no loss of generality, let us assume that t 3 input is 1 and t 4's input is 1, which means both of them both of these transistors are turned on. So, we basically have a direct connection from the match line to the ground, and this means that if A is not equal to V there is a mismatch, and in this case the match line will be driven to 0 volts, because it is connected to the ground. So, what do we understand? We understand that if A is equal to V then the match line is kept floating, which means it is not connected to the ground, and if A is not equal to V then the match line is given to 0 volts right. So, that is the important take away point, from this particular slide, that you know if there is equality between A and V, then we are fine the match line is still get floating.

Otherwise it is given to 0 volts. So, given this we can design a very simple circuit. We can design an array of cam cells, which looks very similar to an SRAM array absolutely no difference.

(Refer Slide Time: 45:02)



So, let us consider a regular SRAM array. So, essentially all of this part that you see right, all of this part is a regular SRAM array. The only difference, is that we have this additional cam mode, where pretty much all the. So, basically it is all the word lines in this case can be turned off, it is not a problem, and we the only difference is that instead of an SRAM cell we have a cam cell over here, and it has it is regular bit lines as the inputs, because a cam can work as an SRAM as well and you can see write drivers and sense amplifiers and everything else, but the only extra part other than the cam cell, is connections to these values. So, for the first caller we compare it with the first bit a 1. So, one side we connect to a 1 complement and other to a 1, as was shown in in the previous figure.

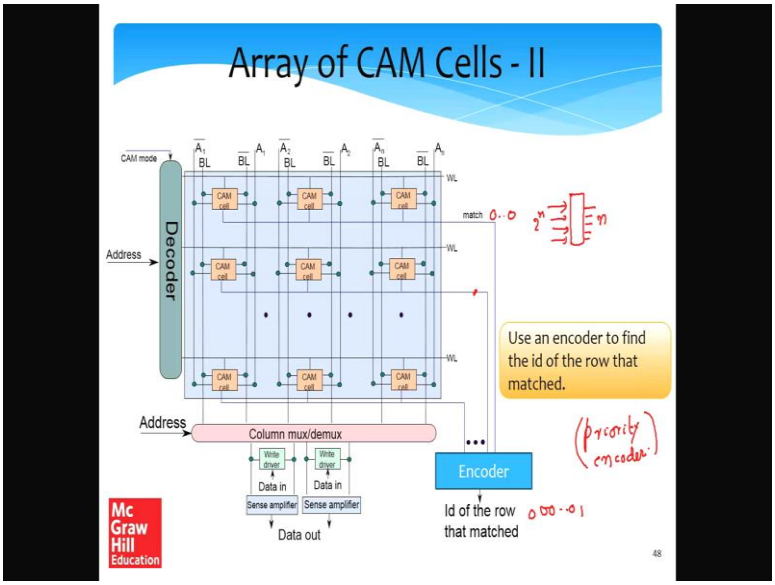
Similarly, the second column we connected to a 2 compliment and a 2, and similarly you know the nth column we compare we compare it with the nth bit, and all of the cam cells are connected to a single match line. So, here is what we do. We pre charge all the match lines to let us say a logical 1. Subsequently let us consider one of the match lines. So, so let us consider maybe the first one. So, let us assume in this case there is equality in the sense the value stored in the cam cell, is not equal to a 2. So, what do you know from the previous slide, when it is not equal to, the match line is driven to 0 volts

Say immediately the value on the match line will be 0. So, what did we do? We pre charge all of them to one, then we disconnected the pre charging circuit and we started comparing. So, whenever there is a mismatch even in a single bit all of these match lines will get driven to 0 volts, but let us assume for this row, all the bit is match right; that is the complete hundred percent match. Then the match line will not be driven, this match line will not be driven to 0 volts, it will maintain the value that it was pre charge with, right because this is not connected to ground. So, it will maintain the value that it was pre charge with which was a logical 1 right, and. So, what do we get to see? Now when we put an all of these match lines to a or gate, all the rows that do not match will have a 0, and all the rows that do match will have a 1.

So, let us assume with no loss of you know, let us assume for the simple case where only one line matches. So, what we will see is that, you know one or. Let us first assume that one line matches, say one matches and the or of all of these functions is a 1 and a next match is equal to 1, which means there is a match. Now let us assume that no line matches, if no line matches all the values are 0. If all the values are 0 the or gate at this

point will compute a 0. So, what we can pretty much do, is that the o-ring all the match lines we can find out, if we found a match in the cam array or not. Subsequently what we can do is we can use an encoder, to find the ID of the row that matched. So, if we would recall what is an encoder. So, an encoder pretty much, it takes 2 to the power n inputs where one of the inputs is 1, the rest are 0.

(Refer Slide Time: 48:43)



Subsequently you have a n bit output which gives the I d of the input that matched, this is exactly what we need over here that we can use an encoder to find out. So, for example, if this is the second one then, the output will pretty much b 0 0 0 1 right. So, where the first line with assuming is all 0s. So, this is the match in a second row, it will be 0 0 0 1. Now assume that multiple rows match. So, we can always have some sort of a priority scheme of choosing one row over the other, and user priority encoder.

So, that is relatively step forward, but if you want to do further reading, you can take a look at the term call the priority encoder on Google, and you will see a circuit, where if there are multiple match lines, that are true then how do you choose one over the other right, but in a simple case if we assume that there is a single match, then the encoder will tell you the I d of the row that matched.

(Refer Slide Time: 50:09)

So, this finishes the end of this chapter a very short quick 3 hour lecture on digital logic. So, what did we cover? We covered basics of transistors, they can write down we covered basics of transistors, we looked at logic gates, and our logic gates we will find something called Karnaugh maps in the books are not covered that, but that will find in the book then we covered latches and flip flops, where our definition is that a flip flop, is basically a large with a clock and then we looked at memory cells right, and in memory cells we looked at 3 kinds of cells.

We looked at SRAM cells which is a cross coupled pair of inverters, connected with bit lines. And a main idea there was that we it is true that we connect an SRAM cell with two bit lines right, this is the SRAM cell BL and BL complement. And the reason is such that we can get take the difference of the voltages between the two bit lines. The advantage of taking the differences that we do not have to wait till a bit line goes to logical 0 or 1, from the direction in which the voltages are moving, we can infer the value stored in a SRAM cell.

The other kind of cell that we looked at is a DRAM cell. So, a dynamic ram cell, the data is stored in a capacitor. So, here there is some problem. So, the capacitor can leak and so on; so necessary to periodically read the value and write it back again. So, there is a voltage can be restored to normal levels. So, this is called the process of refreshing. And

finally, we looked at cam cells content addressable cells, where we have a 10 transistor cam cell and a match line.

So, in the next chapter we will take a look at computer arithmetic which is methods to do addition, subtraction, multiplication and hardware.