Digital VLSI Testing

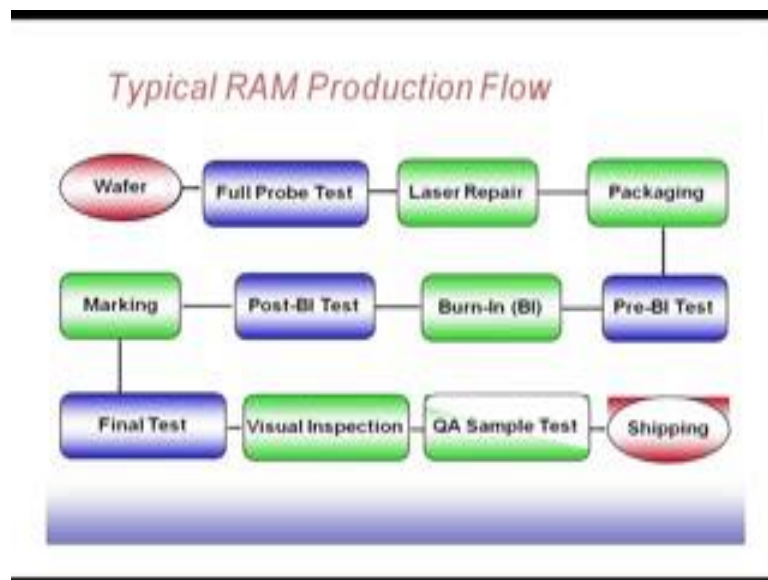
Prof. Santanu Chattopadhyay
Department of Electronics and EC Engineering
Indian Institute of Technology, Kharagpur

Lecture – 57
Memory Testing

So, in the next part of the course, we will look into another important testing challenge which is known as the testing of memory. So, testing of memory is significantly different from the testing of logic circuits consisting of say this logic gates, because memory has got some regular pattern and the structure is regular in nature, it has got a particular architecture consisting of the decoder then sense amplifiers and all that, so that is one thing and that is one side. And the other side is that it has got a number of cells; the data are stored in the cells and the cells, they show different types of behavior. Like it may be the case that when the cells being close proximity of each other, when we try to put a particular cell to some value, some neighboring cell also changes their value, so that was not the case in case of in the logic circuit.

We have always assumed that antenna analysis there is a bridging fault, so these lines can be changed without any problem. But that may not be the case in case of in the memory and also the fault models are totally different when we are looking into this memory testing, so that way it makes a very interesting part of this testing VLSI testing process.

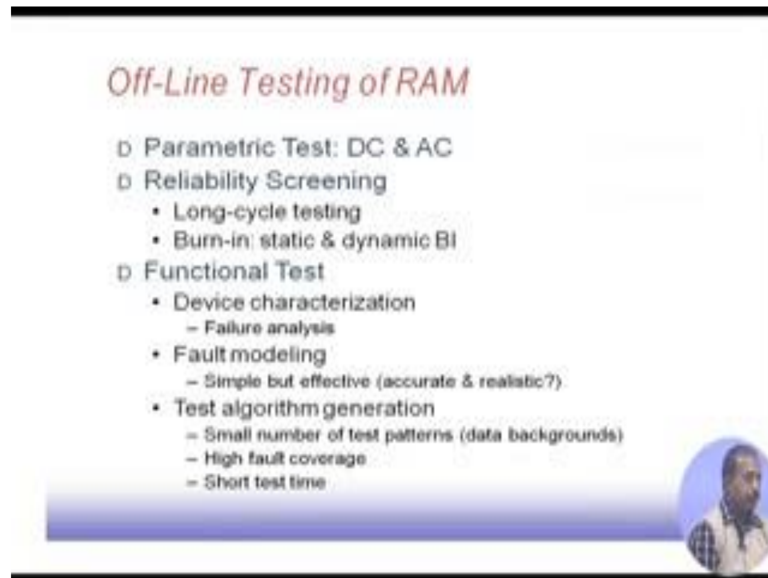
(Refer Slide Time: 01:42)



So, typical memory production flow or the RAM production flow is like this. First we start with the wafer; and the wafer level we do a full probe test and then if there is some defects are found, we do some laser repairing something that is corrected locally and then it goes to packaging. So, in the packaging, after packaging, so we do some tests again which is known as pre burn-in test pre-BI test. So, basically some patterns will be written and it will be read back and it you see that whether they are been done properly or not.

They need goes in to the burn-in phase where it does it has subjected to a long test sequence, and doing for maybe different types of patterns and all that. So, some chips they will survive in the burn-in phase, they will go to the post burn-in test. So, again the after the burn-in phase, some more test will be done. So, then the marking will be done like which one is good, which one is bad in the final test will be done, visual inspection, quality assurance sample test and then shipping. So, this is the total RAM production flow.

(Refer Slide Time: 02:58)



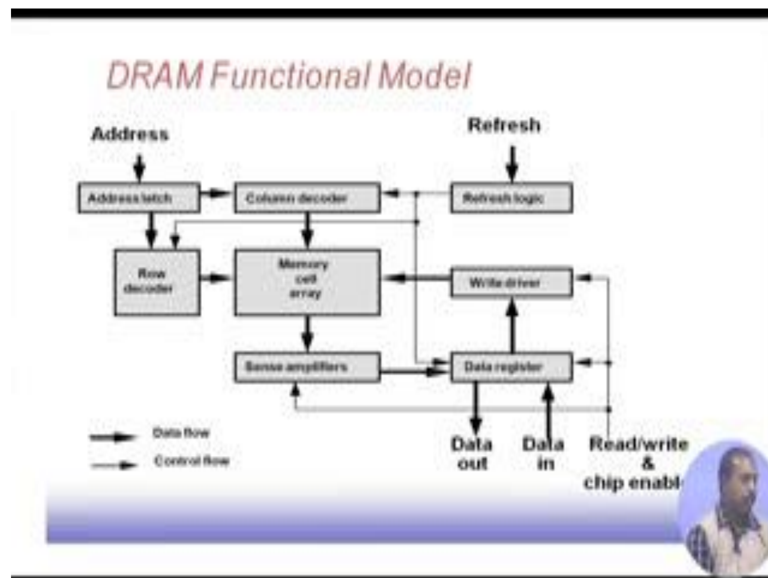
Off-line testing of ram, so it is done in this way that is parametric testing, DC and AC. So, we can measure some of the voltage values, some of the current values that are there, so that way you can do some parametric testing of this various points, then it goes into reliability screening. So this is another offline testing; so long-cycle testing, so we made say large number of read-write operation, and see whether the values have been written properly or read properly or not. Then burn-in, so this can be static or it can be dynamic bur in. So, under it may be dynamic means it is put in to system operation; and this static means it is it is statically testing, before it is putting into this functional test it is we do some testing, so that we get some confidence about its operation.

Then the functional testing, so device characterization we try to see what sort of failure are occurring. And if the failures are found then we try to model like why this failure has occurred and that way we can do some device characterization. Some fault modeling, so whether it is we can model some of the faults like there may be actual defect maybe something different, but it will may be modeled as some other fault, so this that is true for logic fault also, so that same thing here so we have to do some sort of fault modeling.

And then there are some test generation algorithm, so that are run for the functional test purpose. So, we start with the device characterization then we do some fault modeling. And then finally, some test algorithm is run to generate test sets. So, this test sets some small number of test patterns are generated which act as background. So, you see that

there are, there is an array of cells in a memory, so they need to be put to some initial values for testing, so that is actually the background pattern. So, under a background, so we can go for applying some more test pattern, so that we can get some high fault coverage and the test time is also less. So, this background initialization becomes an important part of the testing RAM testing part.

(Refer Slide Time: 05:23)



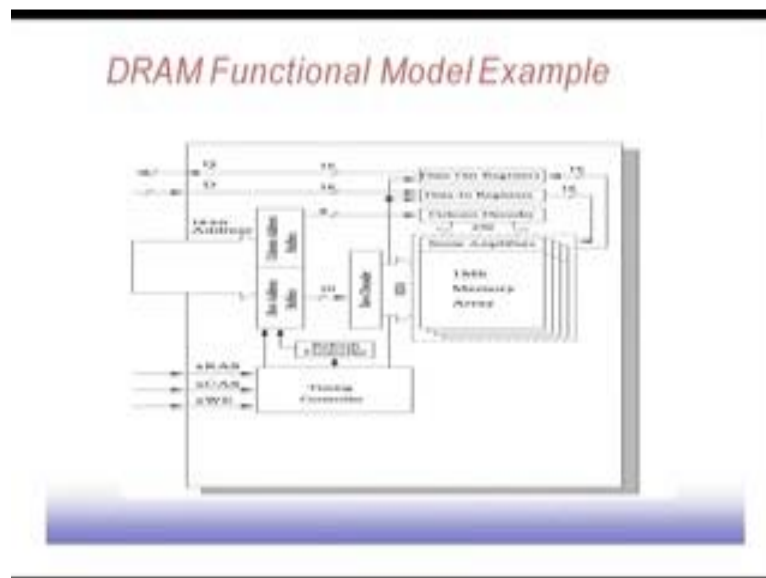
So, DRAM functional models of how does the DRAM some memory looks like. So, you see that we have got a memory cell array, so this is the memory cell array of memory cells for actually the data are going to be stored. Now, there is so the address part that comes, so address part is divided into two parts, so the row address and column address. So, address part the address that is coming, so it gets latch onto this the address latch and then this address is divided into row part and column part, they go in to row decoder and column decoder. And this row decoder will select a particular row, column decoder selects a particular column and that way the content of that cell is read.

And what happens is that this in the DRAM cell, so this is basically is nothing but a single capacitor and that single capacitor gets charged when we store a particular value there. And when we try to read the content of that cell, every there is there are two lines one is known as bit line, another is known as bit bar line. So, the logic that we get at bit line and bit bar line, so their difference will tell us what is the value that was stored in the cell. So, there is a sense amplifier that will identify that will amplify the difference

between this bit and bit bar lines and this difference gets amplified and stored into the data register. So, if for the read operation, so this difference between this bit and bit bar line is coming to the data register and that is coming as data out from the memory.

On the other hand, if it is a write operation, then this data that is there in the data register, so that is put into this write driver. Again this address location and address decoders the row and column address decoder, they select a particular cell and accordingly the value gets written onto that cell by again through that the bit and bit bar lines. And also since this is a DRAM, so we need to have some refresh logic, so refresh logic it selects it goes to the column decoder and this column decoder will decode that particular column, and for each entry in that column, so it will get rewritten. So, this value comes to the data register, data register from data register it goes to the write driver, and from write driver the cell is again rewritten, so that refresh cycle is also a part of this DRAM functional input. So, we have got this data flow and control flow controlling the whole operation of this DRAM.

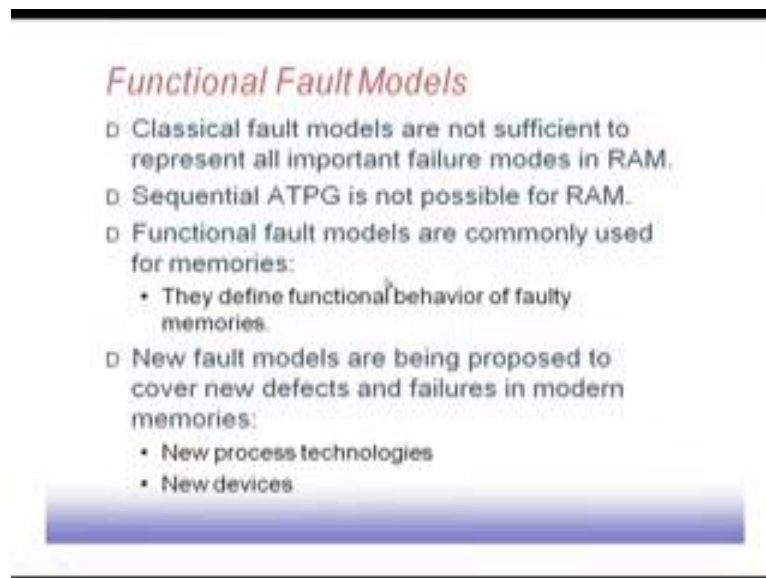
(Refer Slide Time: 08:12)



So, this is the functional; so this maybe a typical example, so we have got this total eighteen bit address out of that this row bit may be 10 and this column address maybe 8. So, they go to two different decoder, so the memory cell is the memory array is 1 mega bit memory it is arranged as 1024 by 256 bits. So, each row has got 1024 bits and each column having 256, so there the 1024 rows and each of them having 256 bits there. And

we have got sense amplifiers for at every column, so that the bit and bit bar lines are amplified difference between them. And that way these values will be stored in the data register for read operation in the data out register and for data in register, so they will be for writing onto it the value will be written data in register, and from there they will go into this memory cell. Now, there are timing controls, so that controls this read access then this refresh control and all these are controlled by this time, so that will activate different modules within this DRAM.

(Refer Slide Time: 09:29)

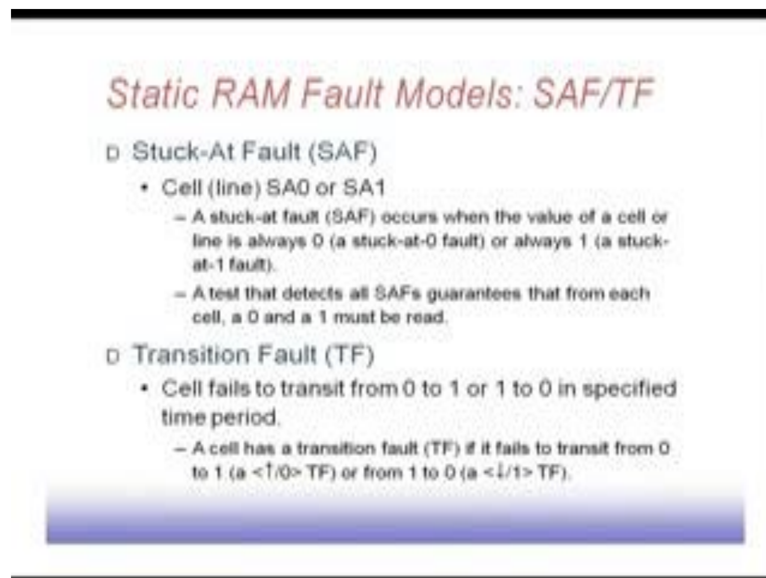


Now, the functional fault models that we have for RAM, so that is significantly different from the classical fault models that we have for logic circuit. So, classical fault models are not sufficient to represent all important failure modes in RAM. Sequential ATPG is not possible for RAM because it is a huge number of cells, so if you try to do sequential ATPG, so if you modeled every cell as a flip-flop or latch, so that way it is a huge number of flip-flops, so it is not possible.

Functional fault models are commonly used for memories this, so normally we go for a functional fault model, so they define the behavior of the faulty memory. So, faulty memory like two cells getting same value coupled to each other, then some cell value some cell reaching a particular continually holding a particular logic level, so like that stuck-at-fault. Stuck-at-fault is some sort of stuck-at-fault will be there, but there are some other type of coupling faults that will come into picture, so will see them.

So, new fault models are being proposed to cover new defects and failures in modern memory. So, with that new strategies for new technologies coming up for designing this memories, so that defect possibilities are also changing, as a result the testing process the fault models for them that they are also changing. So, anyway, so this is a functional fault models are going on.

(Refer Slide Time: 11:00)



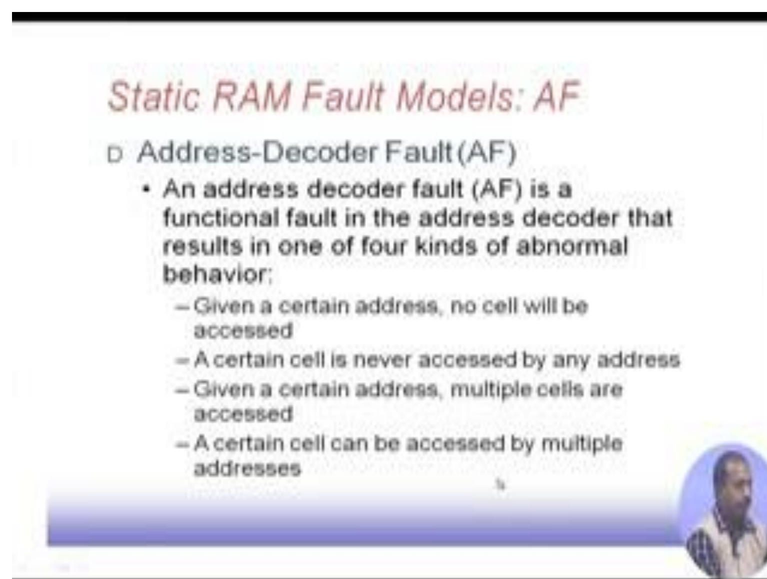
So, we will first consider the static RAM fault model which is known we will look into two such fault model one is stuck-at-fault model, another is transition fault model. So, stuck-at-fault model means a particular cell is stuck at 0 or stuck at 1, so this is similar to the stuck-at-fault that we have in logic circuits. So, stuck-at-fault occurs in the value of a cell or a line is always 0 that is a stuck at 0 fault or always 1 is stuck at 1 fault. A test that detects all stuck-at-fault guarantees that from each cell is 0 and a 1 must be read. So, if we are generating a set of test pattern that can detect all stuck-at-faults then for every cell, it must be reading a 0 and must be reading a 1, so that test pattern must be ensuring that.

Then there is a transition fault, so transition fault the cell fails to transit from 0 to 1 or 1 to 0 in specified time period. So, we apply a transition on the cell, and see whether that transition has occurred in that cell the cell content is changed or not. So, maybe we read a we write a 0 value then we try to see whether the after sometime whether the 0 value

has been written there or not. Similarly, previously the cell content has to be ensure to 1 of course.

Then similarly we are going to check for a transition from 0 to 1, so initially we write is 0 there and then try to write a 1, and then try to see whether this value has 1 has really come onto the cell or not, so that way we can have transition fault. So, cell fails to transit 0 to 1 or 1 to 0 in specified time period. A cell has a transition fault, if it fails to transit from 0 to 1, so this is written as rising stuck at 0, so rising and 0, and 1 to 0 falling and previously the pattern was 1. So, previously the bit was 0, now it is time to rise; and here it means the previously the bit was 1 and now it is trying to fall.

(Refer Slide Time: 13:09)



Another major source of failure of RAM is the address decoder. So, you see that in this RAM we have got this address decoder, so if this address decoders fail then we are not going to check the particular memory cell, but something else. So, the address decoders failed then also we have got severe problem with a operation of this memory. So, this address decoder fault, so they formulate one class of faults for this RAM. And address decoder fault is a functional fault in the address decoder that results in one of the four kinds of abnormal behavior. First of all given a certain address no cell will be accessed, so we put a address, but none of the cells get selected, so that is a problem with a address decoder.

A certain cell is never accessed by any address. So, we try to select one particular cell, but whatever address we put, so it is never selected, so that is there is a decoding problem. So, the difficulties that you see from the address decoder, so every cell has to be selected, so that way the address decoder assumes a very important part a very important role in the operation of the memory. The third type of fault that can come is given a certain address multiple cells are accessed. So, several cells are getting selected simultaneously, so that is also an address decoder fault. A certain cell can be accessed by multiple addresses, multiple addresses they select same address location same cell location, so that is also type of fault, so the address decoder we can have this four types of faults.

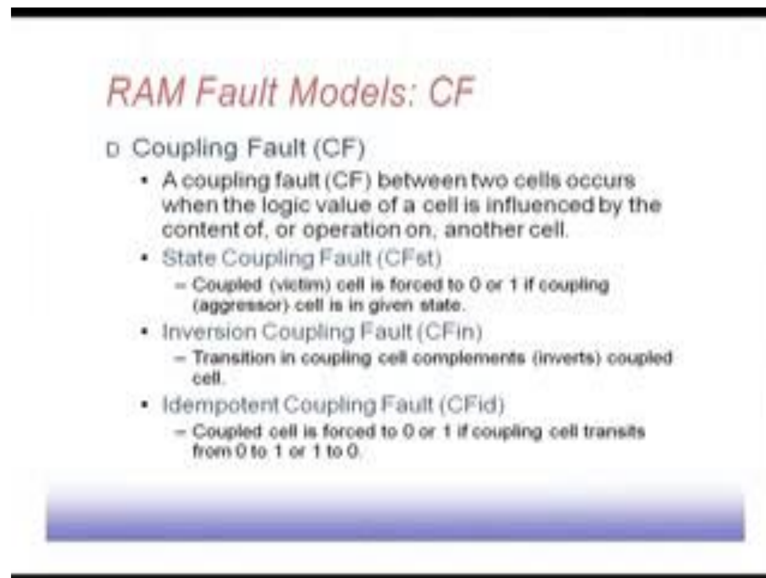
(Refer Slide Time: 14:46)

Static RAM Fault Models: SOF

- Stuck-Open Fault(SOF)
  - A stuck-open fault (SOF) occurs when the cell cannot be accessed due to, e.g., a broken word line.
  - A read to this cell will produce the previously read value.

Then we can have something called a stuck-open fault, so it occurs when is the cell cannot be accessed due to a broken word line. So, the cell has become un accessible, so we cannot change the content of it, so that is the word line failure is there. A read to this cell will produce the previously read value, so whatever was previously read, so it will again give the same value though it try to change it and then read it, so this change will not get reflected into the read value.

(Refer Slide Time: 15:20)



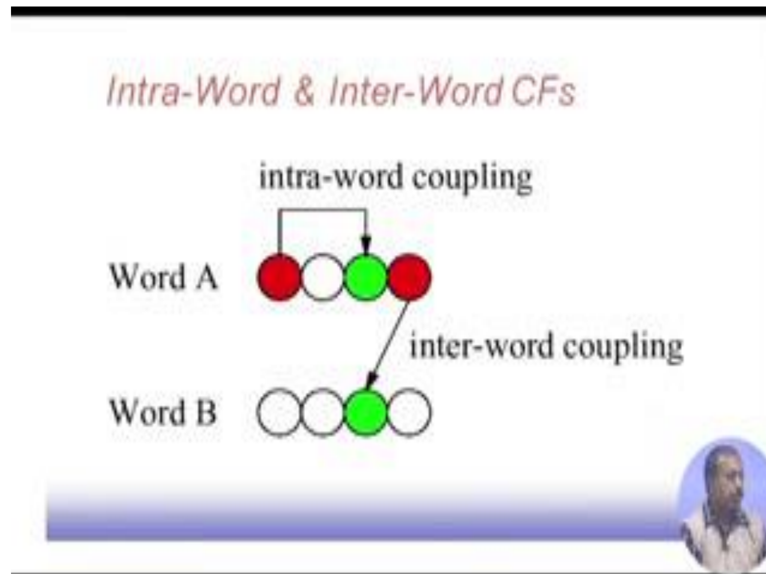
Then more interestingly in RAM, we have got the coupling fault. So, these are specific for this memories. So, a coupling fault between two cells occurs when the logic value of its cell is influenced by the content of or operation on another cell. So, you see that the content of a cell is 0, so that effects content of one of its neighbor, so that is one type of coupling. Similarly, I want to make a transition on a particular cell that effects one of its neighbor, so that is also a coupling fault. So, we have got this state coupling fault, so coupled cell is force to 0 or 1, if coupling cell is in the given state.

So, we coupled cell is the victim cell and the coupling cell is the aggressor cell. So, we are trying to put a 0 in the aggressor and the victim is also getting a 0; or we are trying to put a one in the aggressor we it is victim is also getting 0, so this is called state coupling fault or CFST. Then there is inversion coupling fault, it is a transition coupling cell compliments the coupled cell, so there is a transition though coupling cell makes a transition from 0 to 1 or 1 to 0 and as a result the victim cell the victim cell gets inverted, so that is a transition fault the inversion coupling fault or CFIN.

And we have got idempotent coupling fault, so a coupled cell is forced to 0 or 1 by coupling cell if the coupling cell transits from 0 to 1 or 1 to 0. So, whenever this coupled cell is going from to it goes to 0, if the coupling cell changes from 0 to 1, it goes to 0. Similarly, if it goes to 1, if the coupling cell changes from 1 to 0, so this thing happen we

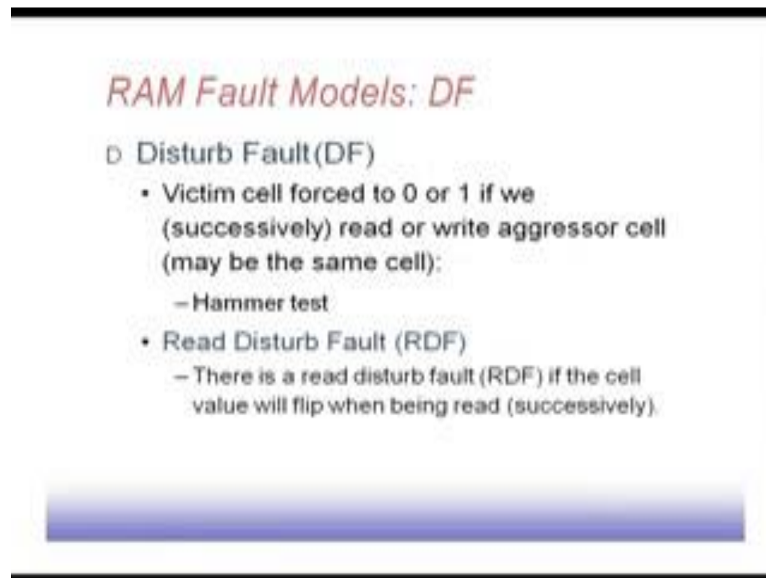
call it an idempotent coupling fault or CFID, so these are the different types of coupling fault that have been observed in the RAM cells.

(Refer Slide Time: 17:25)



We can have intra-word fault and inter-word this coupling fault. So, intra-word coupling fault means within a particular row we have got coupling. So, this cell and this cell, so they are coupled, so if this cell content goes to change, so this cell is also getting modified. And we have got inter-word coupling, maybe we have got two successive words and this word when it is this particular bit when it is changing effects this particular this cell of the next word, so we have got inter-word coupling. So, we have got intra-word coupling as well as inter-word coupling.

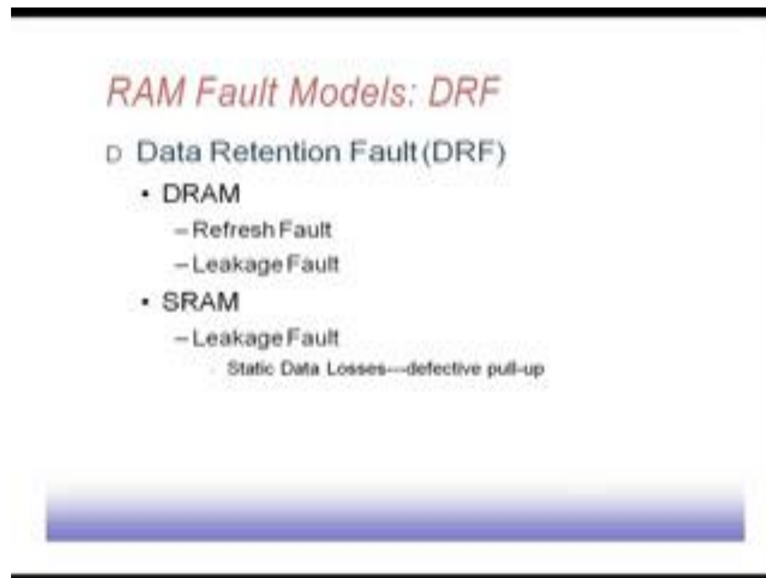
(Refer Slide Time: 18:07)



More fault like that disturb faults - the DF. So, DF the victim cell is forced to 0 or 1, if we read or write aggressor cell and maybe say the aggressor thus say we do it successively we do it continually. So, if we read or write an aggressor cell continually then it permanently it the victim cell is becoming 0 or victim cell is becoming 1. So, this victim cell an aggressor cell may be the same cell also, but what is happening is for a particular cell we have continually doing a read or write operation as a result that cell or some other particular cell, so that becomes 0 or 1, so that is say disturb fault that is marked as disturb fault.

And it is detected by one test which is known as hammer test, so we will not going to the detail because that is quite long. And then this read disturb fault, so there is a read disturb fault if the cell value will flip when being rate successively. So, read disturb, so we are just trying to read the content of the cell, so when you read the content its call read the read a cell, so its contents should not be disturbed, its content should not be modified. So, this read disturb fault says that if I am reading it continually then the content of the cell gets flipped, so that is the read disturb fault.

(Refer Slide Time: 19:28)



Then there is data retention fault. So, data retention fault basically for SRAM, so we have got leakage because s RAM there is no refreshing, so there is a back-to-back inverters are connected to from the SRAM, cell now this or maybe there are some other designs are also available. But essentially what happens is that ultimately this bit that we are going to store is stored by means of some feedback mechanism between transistors. So, they are actually holding the value. So, now due to leakage, so this transistors may lose the logic level, so that is the static data loss, so that is the leakage fault for SRAM.

For this DRAM, so we have got refresh fault and leakage fault. The refresh circuit is trying to do the refreshing of this RAM cells continually, while getting reading the value from the cell and writing it back onto there. But if there is some problem in that refresh circuitry itself then this refreshing will not work, so that gives rise to the refresh fault, the content of several cells may be lost. Similarly, we have got the leakage fault for DRAM cell as well, so that is the leakage fault for DRAM.

(Refer Slide Time: 20:50)

Test Time Complexity (100MHz)

Size	N	10N	NlogN	$N^{1.5}$	N^2
1M	0.01s	0.1s	0.2s	11s	3h
16M	0.16s	1.6s	3.9s	11m	33d
64M	0.66s	6.6s	17s	1.5h	1.43y
256M	2.62s	26s	1.23m	12h	23y
1G	10.5s	1.8m	5.3m	4d	366y
4G	42s	7m	22.4m	32d	57c
16G	2.8m	28m	1.6h	255d	915c

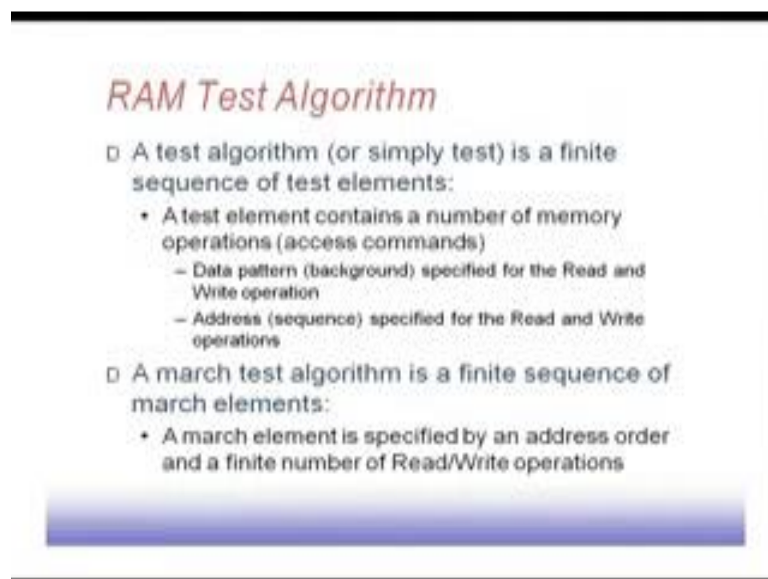
So, if you look into the complexities of this test pattern, so if the size of the memory is 1 million, 1 mega, 16 mega, 64 mega up to say 16 gigabit like that, and if I have got a test policy that that can test the memory in linear time. Then and the test runs at 100 megahertz frequency, and it can do the testing in linear time then for 1 megabits size memory, so it takes 0.01 second, whereas for 16 gigabits size memory it takes 2.8 minutes. On the other hand, if the complexity of this testing process is high, so it requires a 10N time to apply the test pattern and see the responses then this is 0.1 second and this is 28 minute, so that is you can understand now the complexity has started growing.

If the complexity of this test application process is N log N, then where N being the number of cells that we have in the memory it is N log N then this 16 GB memory, so it will require 1.6 hours to test. And if the complexity is even higher N power 1.5, then it requires 255 days to test one 16 GB memory 16 gigabit memory. And if the complexity is N square, then it will require 915 centuries to test this 16 gigabit memory. So, there actually is the complexity, so the normally when we are designing any policy or any algorithm for some problem, so we are normally happy if we does not going to the exponential domain. So, if it is remaining in the polynomial domain N, N square N log N N cube, so like that we happy because we know that it is not going to affect much in terms of time.

However, here you see you cannot think of anything any algorithm, which is beyond the linear time. Because if you go beyond linear time, then this memory capacities are increasing every day, so this new technologies are coming, so this memory capacities are increasing every day. So, this way that distinct time will of those memory chips that will become more and more high, so we cannot I do not think we can afford say 1.6 hours to test one 16 gigabit memory for its current occurrence.

So any of this computers now they have 16 gigabit, 16 gigabyte or in that range, so that way the amount of memory available on this normal systems are also very high, so if we are going to test them we cannot go for this high highly complex test pattern sets highly complex test application procedural. So, the testing process for memory should be simple, it should be fast; at the same time we want that all the fault models that we have discussed, so all those fault models will be covered by the test that we are applying for the memory.

(Refer Slide Time: 24:07)



So, a test algorithm or simply a test, it is a finite sequence of test elements. So, how do you test we have to apply a sequence of test element. A test element it contains a number of memory operation. So, what is a memory operation we may like to read the content of the memory cell, we may like to write some value onto the memory cell, maybe we write some value. And then after sometime, we try to check the content, so that way we can see we can try to see where that this operation was successful or not. So, a test algorithm

it is a finite sequence of test elements and a test element it contains a number of memory operations or the access comments, so read-write access comments.

So, there maybe we may perform some data pattern background data pattern of background specified further read or write operation. So, previous some first we have to set the memory to some background and then we try to do a different type of write. Then also we need to tell the address sequence in which the read write operations will be done. So, first we write at a location then at the next location then at is it in an increasing order starting from the lowest possible address to the highest possible address or it is in the decreasing order. So, maybe we first write it in the increasing order and then while reading we read it in the decreasing order, so that we get certain types of faults detected, so that way we can will see that this type of read write address sequence that also has got some definite role to play in detecting faults.

So, this main memory test algorithms, so they are known as march test algorithm because they define some sort of marching pattern through the memory. So, this march test algorithm that is also a finite sequence of march elements just like any test algorithm is a sequence of test elements, so march test algorithm is also a sequence of march element. Again a march element is specified by an address order and a finite number of read-write operation. So, each element will tell me in which order you are going to apply the going to do the read-write operations; and the second it will tell what are the read-write operation that we are talking about.

(Refer Slide Time: 26:27)

March Test Notation

- ⇑: address sequence is in the ascending order
- ⇓: address changes in the descending order
- ⇕: address sequence is either ⇑ or ⇓
- r: the Read operation
  - Reading an expected 0 from a cell (r0); reading an expected 1 from a cell (r1)
- w: the Write operation
  - Writing a 0 into a cell (w0); writing a 1 into a cell (w1)
- Example (MATS+): {⊂(w0); ⇑ (r0, w1); ⇓ (r1, w0)}

While talking about say march test, so this is the thing that is we can apply this the symbol. So, this is the address sequence is in ascending order that is starting with a lowest possible address, so it will go upward. Then there is a decrease, this symbol is given, so that is changing in the descending order. So, this symbol tells that you can put the address in either of this two order ascending or descending. Then as per as the operations are concerned, so there are two operation r and w, read operation and write operation. So, when you are trying to read some value, so we have some expectation, so maybe previously I have written a 0 there, so I would expect that I will read a zero if the cell content is remaining intact.

So, while talking about the read operation we tell that I want to do a read, and what is the expected value that I am getting that we that I like to have from the cell. So, r 0 is that I am expecting to read a 0 value. Similarly, r 1 means we have expecting to read a one value from the cell. Similarly, the write operation w, so it means that we want to writing 0, so that is w 0, so we want to write a 0 value into the cell; and w 1 means we want to write a 1 value into the cell.

An example of a march sequence may be like this, so sorry this is not the c actually this is the problem of this power point. So, this particular symbol should be there. So, it tells that first we apply this write 0 in any order, so upward, downward does not matter, so this is the first march element. So, for all the cells we are writing 0. So, may be in an if

we follow an upward addressing, so it is for all the addresses, for all the locations it has written a 0. Now, we would like to check whether this zero has been written properly in all the cells or not. So, we do this operations. For all the cells, we first read the content of the cell and we are expecting to get a 0 there and then after that we are writing a 1. So, we check the value rate, we check the value rate from the cell and then we write the value 1 at that point, so we are changing we are complimenting the content of that cell.

Then in the descending order, we are doing this read operation, so since we are written a 1 here, so we are expecting to get a 1, while we are reading, so this is the descending order reading and then we are writing a 0 there. So, if the memory is flawless then what will happen initially the memory content will be all 0 then I will be reading the 0 and writing 1 there. Then at the end, I will be reading the then on the reverse order I will be reading this 1 and writing the 0 is there, so that way I will be getting the again the memory back to all zero content, but it will be testing a number of faults that will that will see in the successive lectures.