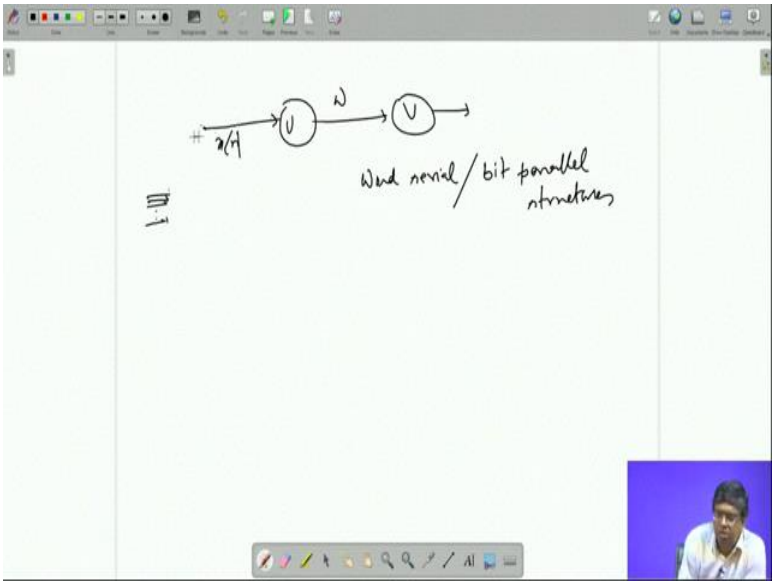
VLSI Signal Processing
Professor Mrityunjoy Chakraborty
Department of Electronics and Electrical Communication Engineering
Indian Institute of Technology, Kharagpur
Lecture 18
Bit Serial, Digit Serial and Word Serial Structures

Okay, now we are in unfolding chapter but then we will do little bit of different things now, things which are which will initially appear to be someone different scenes.

(Refer Slide Time: 00:37)

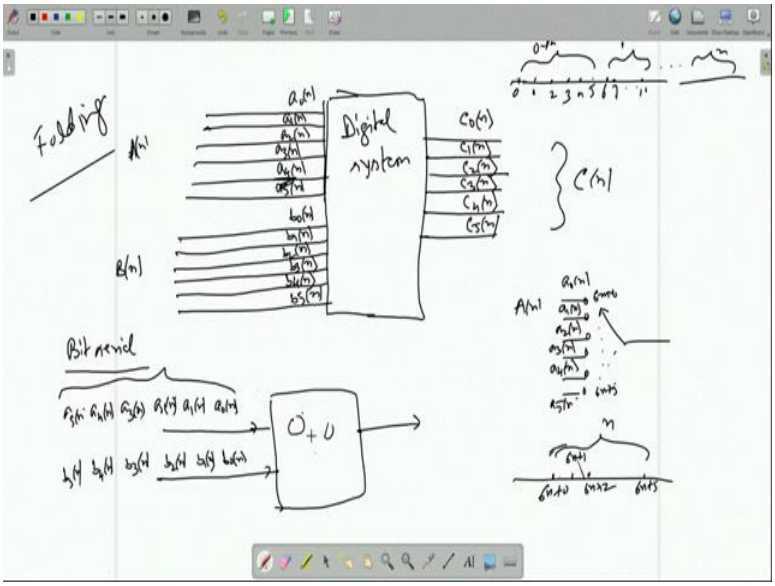


So, long we have been like you know drawing lines like this going into a node U like this V, maybe some delay W. Here I see xn, this xn actually is in line, but it digital hardware it is a bus it we have 16 bits in parallel. For 32 bits in parallel, so it is a bus, there may be parallel lines, in general.

So, if it is a 16 bit word? All the 16 bits will come parallelly. So, one word with the whole data xn. Next is another word. Next is another word, they come through the bus. These kind of structures where the bit are parallel, bits are parallel okay? They are called bit parallel or word serial structure? Word serial and bit parallel structures.

That is all the bits, the 16 bits they are coming, entering the system parallelly for every word. Word is coming cycle after cycle? In a cycle word set of 16 bits go in, N plus 1 th cycle, another set of 16 bits go in. So, bits are parallel or whole word is serial. This is the thing we have been assuming that is why, we should have showing all the lines, bus lines parallelly we just saw one line which understood and basically talking about a bus. So, this is called word serial structured.

(Refer Slide Time: 02:16)



To continue suppose I have got a digital system, it takes two input A data and B data. A data maybe like this, 6 lines. So this is A, so LSB a0, you can write as a function of n in an n th clock, n th data clock. I have got input data An, whose 16, 6 bit depended are like this a0 th, which is a binary bit, either 1 or 0.

Then a1n, a2n, a3n, a4n, a5n. So, s0 to s5 at the nth clock they go in. So, all the 6 bits because data here I am (())(03:25) to be 6 bit, So all the 6 bits of the data they are going parallelly, so bit parallel or word serial structure I mean for A.

Similarly it can take another one b as an example only, this Bn, so we have b0n, b1n, b2n and output could be the n th clock output will be again its LSB Least Significant Bit will be these. All right? This is the perfect bit parallel word serial system. There is one extreme opposite of it called bit serial. Actually this digital system, whatever it is doing you know, maybe it will take a bit of a0. Maybe bit of a, a0n, b0n it will do a processing on that give you c0.

Same processing it will do what a1 and b1, it will give you c1, same processing it will do on a2 and b3. I will show an example a2 and b2. So, there will be various processes are just parallel, one giving c0 one giving c1, one giving output c2, one giving output c3 parallel. And parallel that is why it take more power to keep more hardware, all right. But it is fast that is I mean I am not wasting any cycle. There is another extreme of it when I go to that this will be this, it will be further clearer, this called bit serial.

In bit serial what is happening, if you are n th clock of the input. Here, this over the entered period, in the first diagram I had a0n, either binary 1 or binary 0 standing, in this slide for this

entrepreneur. Simultaneously, a1n which is either binary 1 or binary 0 was standing what this entire period on this line, a2n which is again a binary bit, either one or 0 was standing on this line for this entire period is n th clock. So, on and so forth everywhere.

What I will do here now the n th clock I will divided into 6. So, 6 just a minute, 6 sub clocks. There is a clock is around it, another clock around it, another around it, another around it, another around it, another around it, there is a indices. So, this clock or you can write this way. So, this clock, this water clock, this clock speed is faster, how many time? 6 times because period has gone down.

So, basically I will employ a 6 times faster clock, why 6? 6 is coming from the word length, 6 bit word length from that the finger 6 is coming. I will employed now a 6 times faster clock and there will be single line through which I will first send in a0n, then a1n, then a2n, then a3n, then a4n, then a5n. So, same period n, so first this time a0n will go, then this time a1n will go in, here a2n will go in dot dot dot.

So, I will be using only one line and data will be posted at a 6 times faster rate, so the system will work at a faster clock. But what it will do? Similarly, for B so it is called bit serial. It is taking bits serially, not word the whole word serially, one word after another word not that bits of each word they are coming serially, not altogether parallelly a0, a1, a2, a3, a4, a5 for n th parallelly.

Similarly, alright, you can even for clarity, you can even view it like this. Just for interpretation as though the An and same for Bn the lines are there and there is a switch or say a commutator, it will first task this, then this, then this, then this, then this. So, you have got a0n that will first go in, then a1n, then a2n. So, remember if this was the time period, this was the n th clock, n th will have first one, 6n plus 0.

The next one will be called 6n plus 1, this in terms of time there is a clock around it. I am just marking the indices 6 n plus 0 is a index of the first bit clock under n th, sees n th word clock. 6n plus 1 is a next, 6n plus 2. So, at 6n plus 0, a0n goes in, if 6n plus 1, a1n goes in and 6n plus 2.

So, actually this 6n plus 0 or so on dot, dot, dot, dot 6n plus 5 dot, dot, dot, dot that is what goes in, So, last one is 6n plus 5 all right, very simple. When an n is 0? Suppose we are starting with the 0 th word that time you have got a00, a10, a20. So, the time 0 th, 0 th was 0 th word clock it will be divided as 0, 1, 2, 3, 4, 5. Then first with it will be what? 6, 7 dot, dot,

dot up to 11, if you that way go to n th, we will see 6n plus 0, 6n plus 1, 6n plus 2 like that, if very easy to take.

So, you can view as though this data were coming parallely but I am using a commutator kind of thing just running through, tapping the one after another at very faster rate, 6 times faster rate and whatever coming they are sending you down the line. So, each bit comes down through a very narrow clock pulse, 6 times narrow, or in terms of split to 6 times faster, there is same for A, same for B.

But what I am getting here as I told you, earlier my digital system was doing some processing on a0 and b0 within the result c0. Parallelly, another processor a1 on b1, leaving the real c1 and so on and so forth. So, 6 times I have got the same repetition or I mean, there is an operation which is, which was done 6 parallelly by repeating 6 times. So, hardware and power were going of, you know 6 times.

But here every bit clock. I will take only one a0, one b0 and do some operation. So, I will not require 6 times hardware, hardware will be compressed. So, I gain power, I require less power less hardware that is the gain. But what I am using is speed because I am using 6 times faster clock to run the system. But my original input data rate it does not change, 6 times faster because I have made all the bits coming serially, one after another. I have to do serial processing. So, I am losing 6 times faster clock, but actual input rate or output, does not go up.

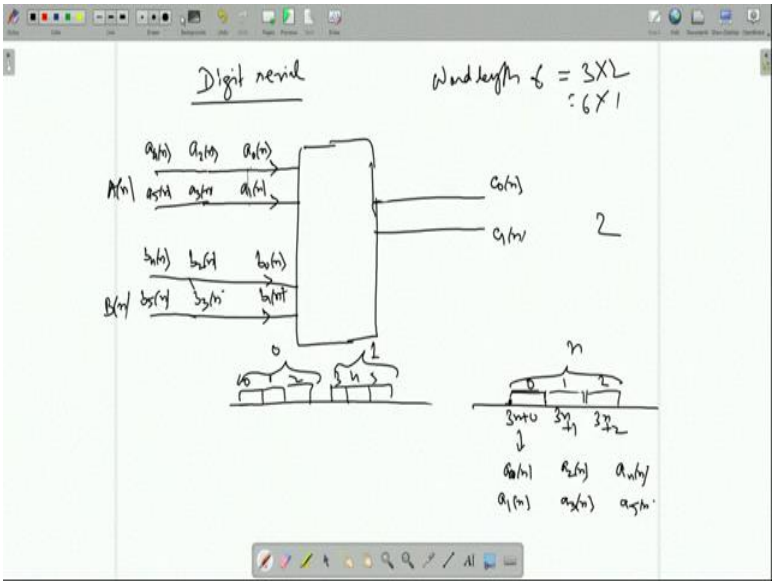
So, it has advantage in terms of power and hardware, 6 times less power, less hardware, but again 6 times more speed. So, loss in speed, gain in power and hardware, this called bit serial, give it a digital serial system. There is a technique called folding, which we will discuss after some time, you can get bit serial, give it a bit serial I will show now very soon you can get back the original digital system bit parallel or word serial by unfolding.

If I unfold this by a factor 6, all the 6 fellows, 6 data they will be made as though they are coming out of a buffer, there will be coming parallelly, there will be 6 parallel lines like this for A, 6 parallel lines like B, hardware inside also if there are some, you know DFG and all that all nodes will be copied 6 times and original structuring will come up. So, from bit serial you can go to word serial by unfolding.

Now, there is these two are two opposite ends, extreme ends, one is fully with parallel. Therefore word serial? Where you pay in hardware, get in speed. Another is fully bit serial.

Where you gain in hardware because your hardware is less, 6 times less, power is 6 times less, but you lose in speed. But there is something in between called digit serial.

(Refer Slide Time: 14:07)



Your word length, in this case 6, a0 to a5, you factorise it, here it is 3 into 2. So, what you can do? You can take two bits at a time a0, a1. Let them come parallel. So, not all 6 are parallel just two or parallel a0, a1 in one clock. In next clock, a2, a3 come parallel, and next clock a4, a5. So, I am not taking 6 bit clocks. I am rather taking 3, what I am saying is? Let me draw a diagram and then will understand.

So, 3 to 2. Suppose I take two lines now, this for A category and I take two lines, this for B category, here I give a0n and a1n. Earlier I had a0, a2, a1, a3 up to a5 parallelly I said no, a0, a1n they will be coming together in one cycle that is called digit cycle that is the timing diagram. If this is my n th what cycle? Earlier it was divided into 6 bit clocks.

That is if you take a factor 6 into 1, 6 bit clocks, then I am taking 3 into 2. So, three digit clock because a0 and a1 together will be called a digit. Earlier a0, it was a bit. Now, together they are coming in two bits, this call it digit. That is earlier it was just binary 1 or 0. Now, it is 00 and 01 or 1011, it is called a digit and three such digits form the full word.

So, here the cycle will be divided not in 6 times. So, it will be divided like these, maybe one, maybe another one, maybe another one. So, 0, 1, 2 that is, it will be 3n plus 0, 3n plus 1, 3n plus 2. Why 3? Because it was start at 0 th, it will have three cycles, digit cycles 0 th it first second.

There is 0 th I send in a0, a1 and b0n, b1n. Then during 3n plus 1, I send the next one, a2n, a3n. They are parallel, here also b2n, b3n and even 3n plus 2, a4n, a5n and here b4n, b5n. So, in one digit cycle, two fellows come here and two fellows. These two bits come, two bits come. So, this will have one processor what we got? a0 and b0. Another processor parallelly what we got? a1, b1 it will give you two outputs.

So a0, b0 to be processed by one processor a1, b1 to be process by another processors, two processors are in parallel. Originally I had 6 by fully digit serial, I could break it down to one, so now I am not bringing to 1, I am bringing down to 1. So, in comparison to the digit serial yes, I will pay more in power, twice in fact, more in hardware. But in comparison to, I mean in comparison, the original one, I am still getting in hardware because from 6 I have brought down to two.

In the, similarly in terms of speed, in the case of digit serial, I was in 6 th time faster a clock, because 0, 1, 2, 3 up to 5. Now, I am using only three times. So, input clock is not going up that much. So, I am not losing that much on speed. I am neither losing too much in power, because instead of I am not, neither getting, neither gaining too much in power.

Because I am not able to reduce power from you know, 6 times to I mean. I am not able to bring down the power by fully 6 times as I was doing in the case of digit serial. Then there are 6 processors parallel as bringing down one processor. Now, I am bringing down to only two.

So, I am paying some in power and bringing down the cost, power cost that is by 3, not by 6. Then at the same time I am not paying that much. There was gain and here loss, I am not paying that much in terms of speed. Earlier the clock went up by 6, now it is by 3, 1 digit clock, another digit clock, another digit clock.

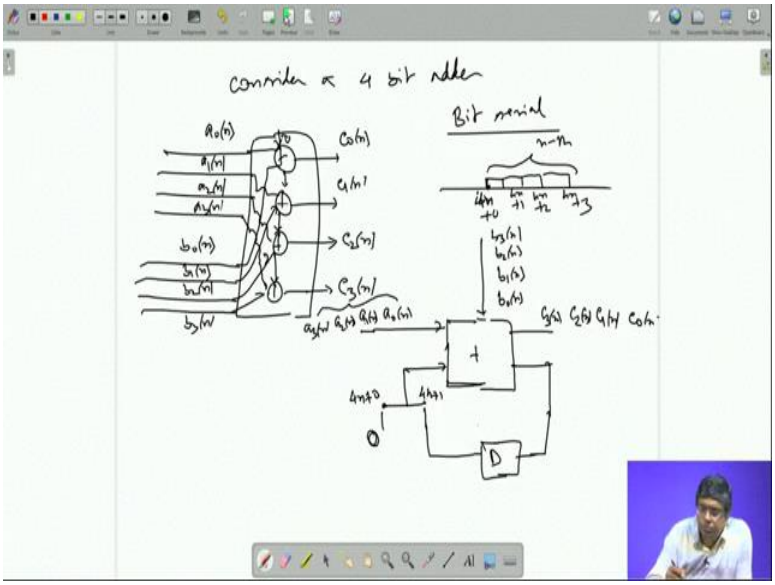
So, during this time, these two fellows come together sorry a0, a then a2, a3 then a4, a5. So, start at 0 th word clock, you break it into 3, 0, 1, 2 next fast, so make it into 3, 3, 4, 5 and so on and so forth. So, it will be 3n plus 0, 3n plus 1, 3n plus 2. So, there is something in between, you do not get full, do not gain fully in terms of power because you are not bringing down hardware by 6 but only by three but do not lose too much in power, in speed because you are not increasing the clock speed by 6, only by 3.

Again from the bit serial you can get back the digit serial just by unfolding, in this case just by factored 2. There I had one bit line only a0, a1, a2 they are coming together. Now, two bits

will be parallel because unfolded by two and you will get this structure. So, if you start with bit serial you unfold by the entire word length, you get the original digit, word serial or bit parallel structure or you unfold by some intermediate into that. Like I could have taken 3 bits parallelly a0, a1, a2, a3, a4, a5.

So, three bit parallel means two cycle, two digit cycle, per word cycle. So, three processors would have been in parallel. So, I would have paid more in power but loss in speed would have been lesser, clock would have been just gone up, would have just gone up by two.

(Refer Slide Time: 21:19)



To give an example consider a 4 bit added, that is if you have, at added 4 bit, originally it is a0n, a1n, a2n, a3n and here you have got what do we do? This is known to all of us we have done a single course on basic digital electronics, you take a0n and take b0n, have a full adder. Initial carry is 0 with the result, new carry. Then again use the same full adder, so word as a parallel new result.

This will be c0n, c1n again use another full adder, this is a carry going in c2n and one more full adder. So, you see as I was telling, the processing in this case is adder between a0, b0 same processing is done between a1, b1 same is done between a2, b2 and same is done between a3 and b3. So, four adders in parallel, so I am paying four times hardware in terms of other hardware. These an example of fully bit parallel or word serial system.

Now, I consider fully bit serial. So, that means as an example I will have n th clock, word clock, it will have four, 4n plus 0, 4n plus 1, 4n plus 2, 4n plus 3, this time a0n. So, I will have to use just only one adder, because system will take only one line for a0, only one line

a1, for b, a0 then a1, then a2, a3, so will b0, then b1, b2, b3. But through only one line. So, at a time only one a bit, one b bit that we added.

Suppose this is a full adder, this is one input I will get a0n, then a1n, then a2n, then a3n, the LSB comes first. Normally these is a convention, least significant bit comes first and then other bits. So, first is this at 4n plus 0, then these 4n plus 1, then this at 4n plus 2, then this at 4n plus 3. So, the way they are coming, it is opposite of time axis, this is a convention we follow, these is a leader, it comes first at 4n plus 0, then the next 4n plus 1, then 4n plus 2, 4n plus 3. Same for this b0n, b1n.

In order to avoid some congestion, here this will be congested. Let me draw b0 from top. So, initial carries 0, there is a carry input line, carry initially is 0. So, there is a switch, the switch which is not fate or mosfets switch, it is done by marks. The switch will move to here, and when I am adding a0 with b0 that time initial carry is 0. What cycle is it for n plus 0? So, at 4n plus 0, for every n switch will connect here.

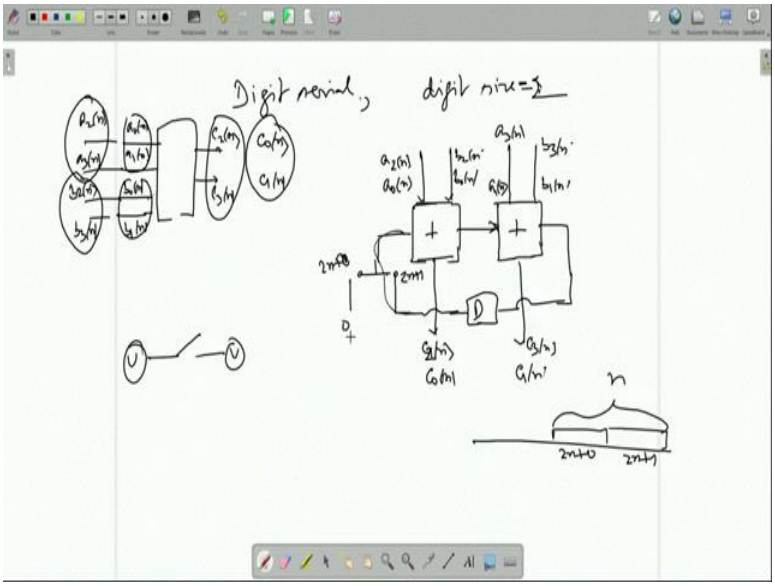
And here I have got a binary 0, 0 vault store. And next time at 4n plus1, whatever be the carry, generating the previous cycle, there is carry out. That will be stored in a flip flop delay that will come out here, it will switch to that. So, when I go to 4n plus 1 carry generated 4n plus 0 will come up and I will move to by switch to the side, 4n plus 1 this side. That time a1 has come, b1 in has come and with that carry generating the previous cycle stored back and now brought out that will get added.

So, first c0n will come, again this, then c1n, again at 4n plus 1 whatever carry generated that will be held in the flip flop and 4n plus 2 that will come here that time a2n and b2n with that carry I will get new result, new carry and I like that. So, this is a bit serial structure. Working under four times faster clock. But there is a switch here because it is very tight I am not giving so much of hardware, same hardware is used to get time, division, multiplexed banner. Sometimes is switching to one side, sometimes it is switching to other side.

Because hardware is too few I have to use it for various jobs now, look you have brought up the hardware that is very switch is required that is the philosophy that whenever you do this bits serial thing you do or even digit serial thing, you get switches. Now, switches have some problem when you try to unfold them, unfold that edge which has a switch, where the switch which discontinuities.

Sometimes there is a path is connected that means path is existing, sometimes path is disconnected is not existing. So, how to unfold such a path or edge. There is an issue I will consider next. But let me now consider a digit serial version of this, digit serial with digit size to.

(Refer Slide Time: 27:59)



So, in terms of timing diagram, n th would be divided into two clocks 2n plus 0, 2n plus 1, like originally 0,1 for 0 th input word. Then for first input word 2, 3 and so and so forth 2n0, 2n plus 1. In 2n plus 0, two bits will going, 2N plus 0, two bits will go in. (())(28:46) it will like this a0n, a1n, will go in and here b0n, b1n will go in.

In the next cycle, next two, so these is one digit, one digit, next cycle a2n, a3n, next cycle there is 2n plus 1 at that time b2n, b3n, and 2n plus 1. Now, put also will be generated, like this in one cycle c0n, c1n one digit. In another cycle c2n, c3n one digit, these are 2n plus 0, these are 2n plus 1. Now, using or engineering common sense, we can design it easily, you understand there will be one full adder between a0 and B0, another between a1 and b1.

So, I need two full adders, here will have a0n added with b0n, result will come out, this is a full adder, there is carry input. So, result is coming out but carry out will then go to the next adder a1 and b1. So, here I have got a1n and this input carry, the full adder, result will be generated, this carry will be stored because a next cycle. So, I am got this edition a2n, b2n, that time, this carry will be used, so I put it in a flip flop. There is a switch initial carry was 0. So, it will move here at 2n plus 0 th cycle.

So, it will move here, so it will take initial carry as 0, new carry that goes in here. Then carrying stored, next cycle at 2n plus 1, this previous carry will be in that time a2n plus b2n and that carry will be added new result. So, originally I had c0n and c1n, in this cycle 2n plus 0. Next time c1n, sorry c2n, c3n, a2n and b2n with that carry added, new result, new carry. And this time we will have a3n, b3n, these is that digit serial structure.

So, from we saw that from our bit serial structure, if we had unfolded it by two, we would have got this. If we had unfolded by four, we would have got the original one. This will be a thing, will discuss next time but then we have to first see how to handle a switch? There is if there is an edge between two nodes U and V.

Forget about delay, earlier if I had U and V I had a line directly connecting bit signal is always going from here to here at all clocks. But whenever a switch it means certain clocks switches closed and single is going at certain other clocks, it is open clocks, it is open, it is not going. So, this continuous kind of thing, sometimes working sometimes not.

So, how to unfold this kind of structures that will be the first thing because you see these switches are coming. This part for instance, it exist in one cycle does not exist another cycle. This will be the first thing and then will use it for this bit serial, digit serial structures, thank you very much.