Digital Switching

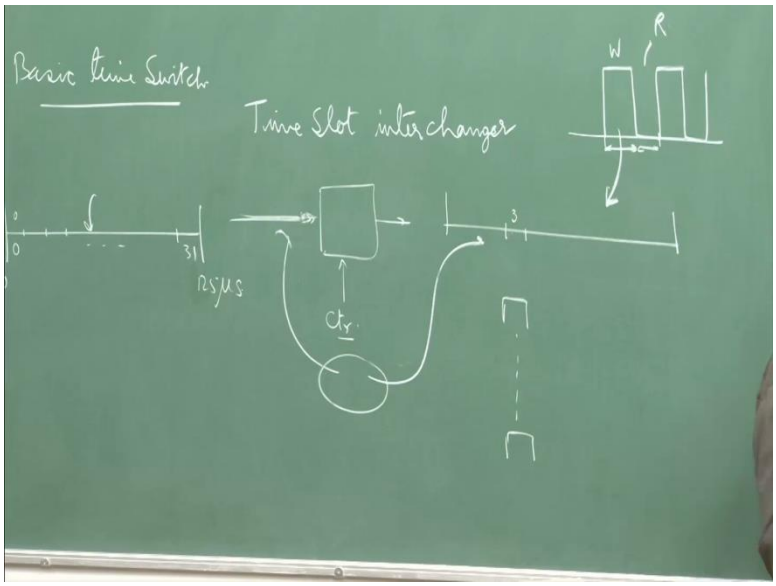
Prof. Y. N. Singh
Department of Electrical Engineering
Indian Institute of Technology, Kanpur

Lecture - 13

So, today last in the previous one, previous lecture what we have been done was, the counter network, and with that we figure out what is the bound on the strictly non blocking switch, which was O on O N block 2 N square actually. So, with that we can always create a strictly non blocking switch that was the last thing which we did. So, with that now, more or less most of the theoretic stuff is, kind of over. So, now I would actually be doing something to do a time switching, how to build up a time switch, using a access random memory. It is a basically a electronic circuit schematic, which I will be discussing, and how this is actually functions. And then we will explain it onto what is to called super multiplex switch, and then of course, T S T configuration, and you will get an idea that the switch is as far as operating system of switch is concerned.

Switch is nothing but will be abstracted as nothing but a random access memory, where the controller software has to just inside nothing but the control words, and once you write a control word certain connection will automatically will gets set up actually. So, as far as the control action which is required, write the appropriate word things will happen in the backend an automatically. You need not worry about how the switching is implemented. So, that is how the restricted in the software system, because one has to write even an operating system of such a switch.

(Refer Slide Time: 01:55)



So, first of all lets go to the basic times switch; implantation, so now what is there with us before we start designing this. We can take a even carrier system; even carrier system will have, as all of us know total 32 voice slots. A 32 actually octets per 125 microsecond, and we are already aware of this that, out of this there will be 2 slots which will be already occupied, by design, one for signaling purpose, and other one is for framing purpose. So, that you can identify the frame at the receiver end so frame boundary that detected by these two.

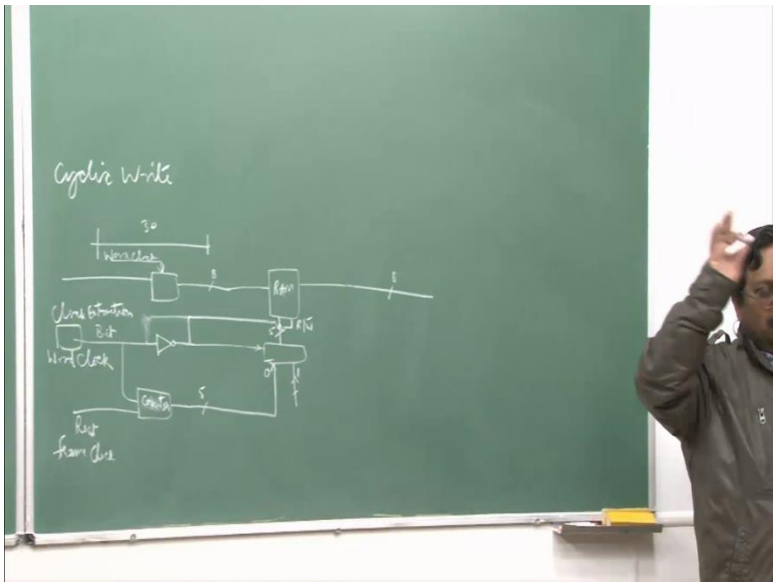
So, usually the idea is, when you get such kind of even frame I should be able to push it into a time switch. There will be some controller, some control action, which we have to do, which will now identify how actually it functions, and at the end of it again I will get an nothing but even only coming. And of course, your framing in signaling which is your 15, because I am counting from 0 and 31. These 2 slots will be signaling and framing slots, so I am not bothered about them, but the remaining slots will get swapped actually. So, you might have some data 0 slot, it might end up in this octet might we are arriving at third slot and so on.

So, I am basically change during the time slot. So, r being changed interchanged actually with each other. So, this kind of switch is also known as time slot interchanger. In fact, something like this kind also can be done in optical domain, for implementing all optical switching; circuit switching actually, and that is a optical time slot interchangers; that is

done usually by delay lines. Delay line can act as a memory, faber looped inside used there. Here I will be using simply random access memory. So, now with that thing now the problem is this, what I will do with the signaling and framing, these are not required when I am doing time switching. So, usually when these will be coming at the interface; first thing will be done is, the information which is coming in your signaling thing, will be listened by the interface board, will be taken out and this information then can be routed, because it is like a virtual channel, will be going to the controller. The software which is running inside the switch, which is a master basically, which runs everything it is like a operating system of a P C. Some said P C also designed exactly in same with architecturally, it is not much different, it is a special kind of computer; that is only thing, which are switching functionality.

So, once this is bypassed that signaling slot does not matter actually, signaling happens in 2 ends. Similarly, the signaling information will be injected in this e 1 till it goes to the next guy, rate to be again taken out. So, signaling information is never switched. If it has to be switched, it has to come here, it will have identified where it has to be forwarded, and it forwarded over a signaling channel. So, signaling channel provides a virtual connection, or a virtual channel between two end points over a link always. This is not like a circuit switching, where the voice is simply just passing through this switch. So, these are basically forth, so that these entities can to talk each other. Similarly once we have identified the frame here, this frame boundary designation octet is also not required to be switched.

(Refer Slide Time: 06:17)



So, usually what you will get is, a 30 frame or 30 slot frames; that is what you will get not 32. So, the board interface board itself will identify that, it will also give you the timing signal, which is important. Timing recovery it would have done, so it will give you a frame clock, a pulse when the frame starts and when frame ends, because within this board you require this, within the switch. It will give you a octet clock, every word; this is known as word clock also, and then there is bit clock, there are 3 clock which will be there in the system. And you will be writing into ram, you will be reading into ram as per certain thing. When it comes to the output, you will again be actually getting all the 3 clocks. You will be getting the frame, which is going to come out serially, and then this will be used to create whatever is the frame boundary; thing the 30 first slot. The 16 slot will be inserted by the controller, information; signaling information you will get back new e 1, which will be transported further.

So, this will be more or less mechanism, in all kind of time switches, for all kind of multiplexes switching. So, there is going to be a, important thing in the interface board itself. You will be stripping out, your signaling and framing octets. You will be actually retrieving out the all 3 clocks, which will be, now we will be used all through this particular board, and there will be combined back again, because once you are on this serial line. In fact, your clocking information, your framing information, signaling all 3 are embedding in 1 single serial line. So, I keep them in parallel while, doing the switching. So, I am not going to discuss how this will be done. This I think already be

done at some other places; a frame delineation mechanism, clock recovery has been done in some other courses, but it is done that is it; that is my assumption, or that actually even happens in the reality or you can do it through P R L or whatever kind of line coding mechanism switch are used to recover the clock, but those are extremely important; if the clocks in case not there, you cannot receive anything it is impossible. This switch will only function and operate if you can retrieve the clocks.

So, usually most of the line termination unit, because anything where you terminate a line, coming in line is a line termination unit L T U. There you will be always having a indicator whether my clock is in sync or not. If your P R L says that time is synchronization, then you will interpret whatever is coming on the line, otherwise you simply discarded. You should not start interpreting even without synchronization; that is not possible. So, there is an assumption that there is synchronize, this true with all kind of systems; C D M A it becomes even severe, because C D M A even the codes also need to be synchronized p N sequences which are there at both ends. So, same concept applies here.

So, my assumption is, I got a 30 slot, and I am also got getting the timing signal. So, this is my octets which are coming in, and In fact, another thing which I will do is, these are the serial octets, and I will be converting into a parallel stream now. This serial I will be converting it to parallel 8 bit wide bus, will be there, which will be carrying the bits of the octet or a word. So, already this will be this requires of word clock remember otherwise you cannot do it, because whenever the word boundary will come at that point a time you have 2 signal, it is a basically serial in parallel out. There is going to be clock here which is a word clock.

So, there is some clock exception mechanism which is sitting here. So, what I have to do is, these words which are coming in and I have to write them into a memory. So, I will use a random access memory here; a technically data in the data out have going to form a same bus, they are not from different buses actually, in actual ram, but I am here showing it as if we are coming from 2 buses. Only thing when you are having a read mode the same bus will be used for reading purpose. When it is a write mode it will be for the writing purpose. So, I am not showing that actually, but I think you can build up even that part.

Now, I am only going to write from the input stream read to output stream, but technically it is done from the same bus. I have already converted to the parallel 8 bit. You have to write in the parallel, there is nothing like serial into a ram here. A random access memory will have, whatever is the memory size, memory watt is a sixty 4 bit, you have to have a 64. Thus whatever the word on that whenever you give that clock that time it should be written, in the address which is there on the address lines. If you cannot have those many wires, internally inside the chip you will be doing this job. So, whatever I am doing you will be done internally inside the chip.

So, at you do actually you have less fan out or less pins on the chip, but in a exception wise yes those many rams are required. You require number of data lines equal to data words size, and whatever is the address, those many bits also are required to transform the address. So, I am explicitly showing them separately. I am also showing input bus and output bus also separately, but technically it is actually same bus which will be coming in. You will be reading here, and there will be and get this is basically if the write is enabled, and if the read is I called it. This will be something like this.

So, there is only 1 bus which is going inside. So, whenever the read is in enabled, read is one. So, this will be 0, and these are end gate. So, it is just stopping, nothing can be written into it actually at this point, but here you can read it. So, whatever is there on the line. So, whatever address you will fit the same data will simply come and lies onto the bus, and then that will become visible to outgoing bus, and whenever you say I am going to right into this, there is a right bar. So, this will get inverted this will be enabled this will be disabled, whatever is the address in that location you will be writing, and this read write, read write bar also will be going into the chip, for writing purpose, but you can design in V L S I whatever way you wish, and In fact, V L S I chips for doing these things are already available. So, you can build up a limited 16 by 16 telephone exchange, it is possible for you to build it actually, in the lab and then fabricate as do whatever you wish that.

So, a smaller size it is pretty simple, even largest size you can do, because crossbars are also technically are built using digital logic gates, and whatever mechanism I am telling that exactly is the part of that design. So, this is what is going to be the ram. So, I have to tell in that the ram at which location I have to write. So, I am assuming that the first octet will be written as first location, second will be at second location, third will be at third

location and so on, and 38 octet will be written at 38 location. They will be written sequentially. So, that actually means, whatever is the word clock I have to run a counter with this, without this I cannot work actually. So, I will take this, I will run a counter. When this counter will be reset to 0 when transforms complete 30 when the frame clock will come. Remember they are 3 clock; frame, bit, and words, and since I have converted to serial to parallel, your word clock will be actually used here, for serially parallel out. This is what the clock which we are using is the bit clock, which will be nothing equal to word clock this for parallel system.

So, technically you do not require a bit clock, you only work with 2 clock; one is frame other one is the word clock, because my bit size is now increase, whatever is 1 of octet size. Now my bit is of this duration, but they are parallel lock. There are ei8ght parallel bits on 8 parallel lines; that is what has been done here. Now this counter, is always going to be reset by frame clock. So, whenever the frame actually starts that time set to be 0. So, first octet, zeroth octet will be written zeroth location, first will be in first, and 38 will be in 38 locations, this will be automatically happened.

So, this will give you the address. This will be used for writing purpose remember, these are write address, and what will be the size of this bus, this will be also again parallel line, if you implementing side the v l s I, then this will be depending on how many words are there. If it is a even systems you have to 3, so you require how many bits; 5 bits, because 2 square 5 is 32. So, this will be going to 5 bit bus. So, this technically depends on how many words will be stored inside the ram; ram size will govern this. I have to also read out remember, so read all address also has to fade into the system, and this will also going to the same port where the write address goes. So, I require a switch because of that.

So, there is a switch or a multiplexer. This is the address of 5 bits, this address will come from another location I will tell how it comes. I think some of you can immediately guess it, because this is a very intuitive design, there is nothing to understanding. And of course, now when you are writing, that is very important, because read and write both cannot happen simultaneously in this switch. So, what we do is, the word clock or any clock will be going high and low for 50 percent of time it is high 50 percent low, then the next clock pulse will come. So, the first path when it is 1, I can use it for reading or writing whatever I wish actually, does not matter for me. So, if I want to use a first half

for reading or writing, the second half for reading, which is perfect. So, whatever you write in the current frame can be read in current frame, but if you read first and then you write in the next half, whatever you are going to write now will always be read in next frame, not in the current frame, that is only the implication.

So, usually because we do not want to create 1 additional 125 microsecond delay, inside the switch you always do a writing first and read later on; that is usually the strategy, this is a very small thing. So, deign will work correctly, but you start introducing 125 microsecond delay, additionally inside the switch, so that we can actually avoid, in this scenario. And of course, one important thing is that, whatever you are writing now, maybe actually read in the next frame that is possible, but the question is, when you are reading here in this cycle may not be reading the word when you are writing from here, that is important. So, part of it will be read later on, part of it will be read on the current cycle, that is possible. So, it is not that everything remains in the same frame, whatever is this all octets which are in 1 frame at the input may not be in 1 frame at the output, that design requires 2 memories actually.

So, what do you do is the whole frame will be retained in the first, frame and nothing will be read out, and in the next frame you will be reading the whole thing. So, that whole frame remains intact. Here I am not doing, I am not reading 2 rams. I am using only 1 single ram. Those are also required if I if you put a time expender. Currently number of slots; 30 slots are here, 30 slots are at their output side, but then we will for example, implement t s t time space time, and I have to build up a strictly non blocking switch.

The incoming number of slots in 125 microsecond is 30 outgoing will be 2 into 30 minus 1 which will be 59. So, 59 slots at the output and 30 slots are at the input, there is a clocking problem actually. I cannot use this kind of mode; you read here and write here. This can only be done if incoming slots and outgoing slots per unit frame time are same, there only then it can be done. So, in that case we use twine memory system, which can be clocked the 2 different clocks. So, that design can be made by modifying this, if time permits I can actually, I will tell you that thing also.

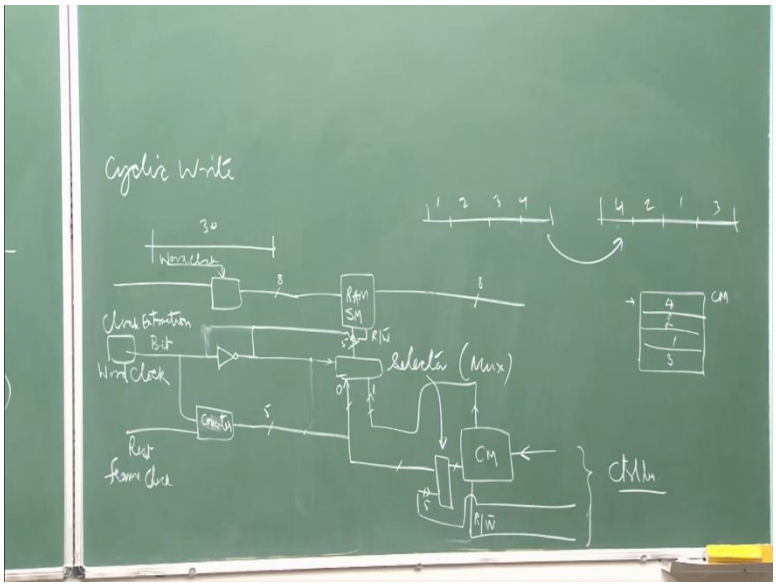
Student: Sir, these durations are in which you have written sir w and r.

This is the 1 clock pulse 1 p, 1 word clock. This will be technically 125 microsecond divide by number of slots which are there per frame. So, in this case it is 30. This is the value of p, if it is a even carrier system t 1 carrier system 24 slots it will be divided by 24, but more or less I think the basic concept once you understand you can almost, start designing almost all kind of things. So, I am going to write it. So, this actually means when this clock is going to be 1, then this will be 0, and I have to put this address. So, 0 means this and 1 you have to select this. This is the controller entity, and I have to also fit read write bar here also.

So, I will take this particular pulse, sorry not from here, because I have to write. So, write bar has to be. It has to be 0, then only the write will happen. So, when it is 1, it gets inverted you have a 0, when you fit 0 the write happens, and this address is what is going to fade into the ram. You can write in sequence, whatever is coming at the input now, is ok. let me explain it again, this is a clock pulse which is coming, whenever the clock pulse has to be 1 I have to write into the ram, and whatever is the value of the counter, it has to go that address. Counter is always reset at the beginning of the frame, it starts counting from 0. So, whenever the case will count from 0 goes to 30 then again to be reset to 0 it will generate 5 bit address, which I am taking out. this address has to be send to the ram, to give the location where the incoming octet or incoming sample; why sample, which is 8 bits sample has to be written, and it has to be done in the positive cycle, because negative cycle I will be using for reading purpose.

So, when it is when the clock is high I have, because remember read write bar actually means whenever read write bar input, is going to be 0, write is 1, because w bar is 0. So, w is 1, and r is 0. So, it will be doing only write operation; that is why this after the inversion I have connected to read write bar, and that that point of time this value which I am creating for control to this much is 0. So, this input will get connected to the address input; not this one, when this value is 1 this will get connected. This I will use it for reading purpose. So, I will be writing sequentially or cyclically into the ram. This cyclic write actually, this cyclic write configuration, and I will do as cyclic read cyclic write and a cyclic read. So, far this was the writing part, I am going to show another bus which is the outgoing bus, but these 2 buses are going to be attached the way I have shown, using 2 gates and a common bus coming to the ram. It will be exactly that configuration, but I have not shown that thing explicitly here.

(Refer Slide Time: 25:39)



Now the question is, how to read. So, there has to be somewhere some information. Remember when I did t s I, I said there are going to be 4 slots, and these have to be mapped. So, 4 have to be read here, 2 has to be here, 1 has to be here, and 3 has to be here. I want to generate this mapping, this mapping has to store at some place, some controlled information. Drawing it on board is fine; I am storing it in my mind. So, equivalent of that is, some control memory.

So, what I can do is, I can always say in the first slot. So, the first location of the control this is the control memory, I will read them and I will put the value 4. So, while reading, in slot 1 of the output frame, I have to read from location 4, or fourth slot of the input frame should be here, has to be transformed here. Next I have to put 2, next I have to put 1, next I have to put 3. So, I will do that exactly same thing, I will take this as an output from, we called it a control memory, this is known as speech memory s m. now which is the location which has to chosen in one particular slot, this should be given by this counter only.

So, I can just take this counter value, use it again I will use a mux there is a reason for doing that I will explain, what the other address will be used for, and this is what I am going to put as an address. And when I am actually activating this, these 3 addresses going to be connected, that time this address should be faded to control memory. So, this same control signal I can actually use. So, when it is read, when this is 1 you are reading,

and I am also reading at the same time from here. We will also write into this control memory; that is what your operating system actually will do. So, this bus will be connected to interface, or a bus of your P C which will be connected to your CPU or virtual a l u. A L U, Ram, Rom, everything will be there connected to bus, and this will also be connected to the bus where you will write something. You will have the address that will be what we called memory mapped address or I E O mapped address kind of addressing can be used, and you can write the words into this, and they will be once it is written switching functionality will happen, as simple as that. As far as the machine is concerned it has to write this to the word. So, you are switching technology totally independent of the computing, but both can be interfaced together.

So, when this is in the read mode, when this is 1; this will become 1, this address will be fade from here which is coming from this control memory, whose address is coming from the counter. So, in the first slot, I need to read 4. So, 1 will be coming from here. it will be connected, because this is the read mode, whatever is written in the first slot which is 4 will be coming out all the way, and this is also once it will be get connected. This is in the read mode. So, it will be read out and to be coming on this side. So, 4 slots will be read in the first octet location, at the outgoing frame. Yes this is the control memory.

So, what is written in the ram machine. Speech memory is the actually speech words, whatever is your voice samples. This is not having your voice samples, is circuit switch basically. So, whatever the input frame whatever bit octet is coming, I am not bother whether it is a voice or whether its start actually, as far as I am concerned, I am able to implement a time slot interchangers.

Student: Who is deciding this information, it is slot.

This has to be done by the controller; the intelligent entities are running the switch. So, when you for example, you are having a phone, and you connect to a exchange. You dial a certain number, that number will be process by the exchange, interface board. I told you that there is interface board in the exchange, in the first lecture, and that particular thing, interface board will know the number to figure out whether routine has be done. Once it figures out from this incoming ports for this out going to port, the connection has to be setup for your call; that time it will fade till find out. You are actually the address

slot number at the input, the outgoing to which I am connecting to some other guy, some other exchange or some other person, is on this slot number on this frame, corresponding to actually put the words inside the control memory that time; once it is put the path is already setup, or activating a cross point is equivalent to that.

Now, this particular address, and there is one more thing; the data which we will write written into this, will be coming from your controller. In fact, it does not matter, I can even modify this design, I need not take this r w bar from here. I can keep all the time this is on read mode, because anyway I am selecting at this point. So, even this r w bar can also come from this side. So, this circuit has to always keep these thing in r mode or read mode most of time, whenever it wants to write it can write, at any point of time whatever values it wants. Let this be taking care of by controller, and you are able to build up a very simple an elementary basic time switch elementary, any doubts so far.

Student: Sir what is that element called just below the ram is it a.

This one this is the selector; selector or multiplexor you can call it, which taking 2 inputs and putting into 1 it is a multiplexor, is a space multiplexor. It is selecting from 1 of the 2 space of this. It is not t d m or w d m it is a space, it is a selector basically. The same unit is present here selector same. They are basically gate 8 inputs and gate, and only 1 of the 2 can be selected at any point of time. So, 2 end and 1 or is good enough to build this. So, all are 8 input gates and 8 outputs; that is everything.

So, same thing is to here, only thing is that this is what, one has to also understand that when this counter going to be reset. So, you will require 2 clocks; word and frame clock here. Word clock will also be used to generate 8 bit parallel system. So, most of the switching will be done in parallel. This does have implication; actually if you work it is extremely high bit rates. Parallel lines always do create a problem, because path lines for 1 bit in path line for another bit, maybe actually different, which may create clock history, as your bit rates increase your size has to go down, so this clock history can be controlled, where that is the only 1 implication which we have.

I think it should have another parallel serial converter at the end also because. You are right, you need a parallel to serial converter. In fact, there will be complete interface board which will take all, parallel to serial thing, which will insert whatever is the control thing, inside the framing stuff, and then form a even frame which will go out. So, there is

the interface board. So, most of the interface boards will do all this job, and all telecom switch is the interface boards job is to take out, all signaling, take care of framing deframing, and make into a raw format, which is internal to the switch, which need not follow any standard, because all switches usually come from one single vendor, and he knows about everything, inside these 2 boundaries. So, only when you are actually talking somebody else at the bother itself only the standard comes to the picture, not inside. Inside you try to optimize, to reduce the hardware, and improve the performance. So, performance per unit of hardware, has to be maximize; that is the objective of any good design. Now I can just move into something.

Sir what is the typical value of this word slot that we using with telephone. It depends on the frame, if for example, if the word clock if you are going to have, even carrier system its 125 microsecond by 30; that is the word period; that said p. So, 30 divide by 125 microseconds that much hertz, or that much pulse is per second is the clock rate. If you are going to use e 3 system, because number of words per 125 microsecond that different, is going to be different. If it is H D H it is going to be still different actually, but h d h walks in a different way. So, that is actually usually part of the optical networks course; the H D H framing and deframing thing. So, that I am not going to cover it here, but conceptually that also 125 microsecond frame static, but you can actually look into there is already a recorded lecture on that, is there already on the site, which talks about the h d h, but that is in e 646.

Student: Sir why is that control memory you have not selected or nor what is the second input for that. It is coming from the controller, C P U.

So what?

There is some master command of the switch, which has to give this map.

Student: Sir that map will be return through the data.

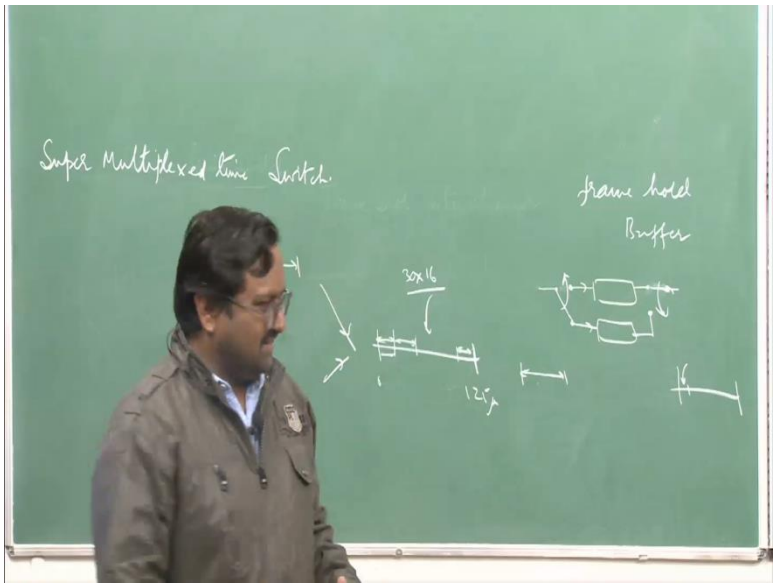
Through this.

Student: Control memory we have a data input.

Now, this is the data input, this is the address sorry this is the address where you will be writing, this is the data, but where you are going to write what data has. Here are we not

writing sequentially in this, like this. Yes if you do that; in that case, you require this framing thing. You know that in the first slot I will be writing only in the first location, second slot in the second location, third slot in third location only. If you put that constant additionally, then you can do away even with this statistic, but only thing now the machine, or the controller should know what are the timing here, because it has to know when the first slots turn will come when I can write it in the first slot, first location. because in this location you can write only in the second slot, here you can write only in the third slot, not in arbitrary in any slot is that case; that is the only additional constant; that also is implemented by hardware manufactures this particular trick, because you save on track, and track is important in any circuit. You reduce the track length, you reduce the cost. If this lot of complexity in manufacturing, so that is the, I think another common trick which is used.

(Refer Slide Time: 38:34)



So, now let us go to the, we 5 more minutes. So, I can quickly explain a super multiplex time switch. A time switching is good, because you can easily operate it these bit rates. It is better to always do it in time mode, because remember all the clocking rates are within your limits, this can be handled. Even 1 gigahertz is actually nothing for us nowadays. Our computer works at even 2 gigahertz or 3 gigahertz kind of rates. The frequency which we are talking about is pretty small compare to that, for all of us circuit elements, but I have only told you for a framing of 30 slots. Suddenly you get a requirement that you are going to get a termination of 16 even carrier systems, as the incoming 16 even

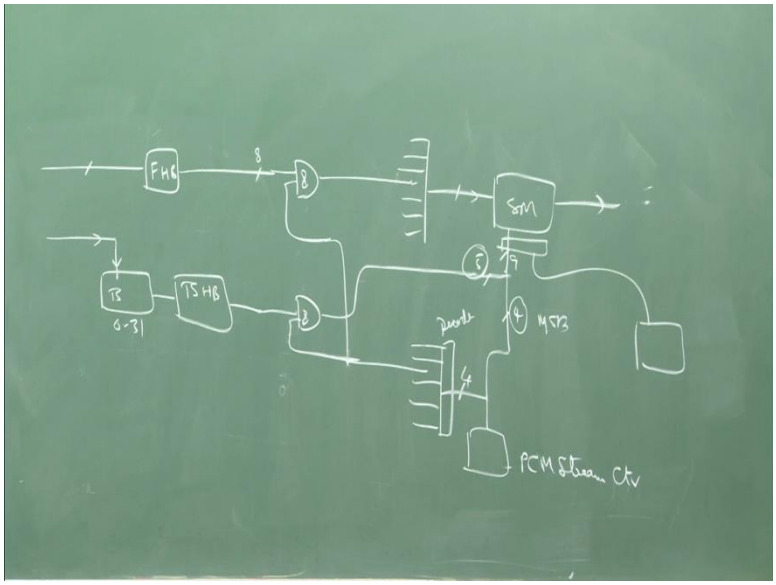
carrier which are going outgoing. once this happens how you will build up a switch; one way is you can say for even carrier system I can use this particular time switch; 30 frames in and 30 frames out, and then I can create a space switch of 16 by 16 and then again time switch T S T configuration. I can do that I can build up the rearrangeble non blocking switch, but then suddenly you will realize. You can actually do even simply with time switching only, but you have to do some trick. The trick is, that whatever the frames are coming 16 frames. You will get these frames, all of them are 125 microsecond duration, and they are all synchronize. I can combine all these, and create 1 frame, which is again 125 microsecond, but now the number of octets which are setting inside it is 13 to 16.

So, what do you have do is, you have to take the first one; compress it in time, and push it here. The second one will be compressed and pushed here, and 16 one will be pushed here. This is known as super multiplexing, this is already a super multiplex stream, I am doing further super multiplexing of that, without taking a part the whole system. So, only thing which we need to know is, how time compressing will be done on this. A trick is again this very similar circuit which has to be used, but remember now your incoming clock rates and output clock rates are different; that is only the consequence which you have. So, this is known as frame hold buffer actually. So, time compressor is known as frame hold buffer. So, circuit will be very similar, but you will be having 2 rams, not one. So, what we will do is, you will keep on writing in 1 ram and there will be a switching functionality here, and whenever this is going to be read out, this will be connected for reading in.

So, whenever this is going to be switched here, this will be switched on this side; that is the only thing which has to happen, rest everything remains the same. And what you do is, the whole circuit not only this path have to be switched, your timing circuit also have to be switched; whatever we have at the other end. And of course, with this actually means, I will be able to put 125 microsecond whatever is coming here, will be coming at this point, but there is a problem. If you want to put it only here, what will happen to the remaining period, do you want to make it silent. Probably it is a synchronize system, it is not a good idea to do that. You let it read multiple times, 16 times how does it matter. I will get this same thing, replicated 16 times. So, that is what is going to be done. So, do not keep it silent here. So, reread 16 times faster, just keep the frame synchronization,

and you will get a same thing repeated 16 times in the super frame. Once I am able to do this job, then I can build up the whole structure.

(Refer Slide Time: 43:15)



So, I am just quickly drawing that. So, this is a frame hold buffer. So, this is now a repeated, 16 times repeated actually frame, after time compression. These are time compression which is done here. So, this is a dual memory system. So, I have not written in the diagram explicitly I think you can do it as an assignment, and if you get stucked let me know, I will just then probably build up a write upon that and submit, or anyone of you who can finish it first, he can scan it and upload on our l m s system, and I will publish it for the remaining people, so that is good. So, I leave it to you, by this much creativity I think it should be possible, I can expect. So, this one is going to have a clock rate, which is going to be. So, I have to generate 0 to 31 counter, it is a 5 bit counter which will be generated. This will also be remembered that 5 bit counter, is counting 0 1 2 3 in 125 microsecond, again this also has to go to frame hold buffer. The way I am compressing this, I have to now generate a count from 0 1 2 3 31, 0 1 2 3 31, 16 times in 125 microsecond. So, that will again be done. This is known as times slot hold buffer, so both are functionality wise same.

There are 16 such elements which will be created, and what you can do is, I can do an ending, for remember this is a 8 bit bus, and for this also I can create an ending. So, for the first one, this input will only be enabled in the first one by sixteenth period of 125

microsecond. For the second input, in the second part only this will be one, this control which I am ending. So, for the first one I will time compress first one by sixteenth of second. Second one second one, third one, so I simply add them wire ending. I will create a super multiplexes thing, I will get that actually. So, this is what is going to happens, a wire end system, then speech memory out, but how this will be generated. So, both this will be coming from at decoder, this will be counting on steam 1 2 3 4 5 16, 1 2 3 4 5 16; this is what we got p c m steam counter. So, it will generate 4 bits, and I will be using a decoder. So, when this value is 0, the first one will be active, and that particular 1 by 60th frame will be going to widen gate, only one of them will be getting the input, at any point of time. When next one will come, so next ones input will be going here to create a super multiplex stream.

Now, speech memory requires not 5 bit address, but 9 bit address now. So, what we will do is, whatever this address coming in; are 5 bits, the 4 bits will be taken from this side, combine together. This will be going in m s b most significant bit size, this will be going towards L S B, because its rate of change is, only 16 times it will change in 125 microsecond. This will change 32 times or 30 times in 1 by 16th of 125 microsecond. So, you will get a 9 bit thing, and you will have a switch, selected switch and all that options will be there, and usually it will be again write cyclic and read a cyclic kind of configuration.

You will have one control memory, which will be now storing, larger number of locations. So, depending on how many slots, so this will be also giving out 9, those many locations are required it is a strictly non blocking esculent of crossbar. So, again this requires similar kind of structure which was there earlier. And on their inreverse side again you will have similar kind of mechanisms, to retrieve back the 16 streams. So, from any incoming port, any slot I should be able to map, any outgoing port, any outgoing even frame, or any outgoing port any slot. So, this mapping can be done, this is a super multiplex time, super time multiplexed switching structure.

So, lot of things I have not covered here, but since I have done the basic time self. I actually request that all of you now you should buildup in detail, this particular design. So, if there is going to be any issue, then we will probably discuss the thing in more detail, because my feeling is that you should be able to do it now, so leaving it you. So, with that I think we close today’s lecture, and we will actually looking into T S T and it

is a control structure of P S T in the next one, and then we will go over to packet switching systems. Voice over I p will be covering actually later on. what do you want me to cover it first, because I would like the packet switching to be done first, and then ultimately the signaling and voice over I p framing formats, everything will be done later on.