Digital VLSI System Design

Prof. S. Srinivasan

Dept of Electrical Engineering

Indian Institute of Technology, Madras

Lecture - 9

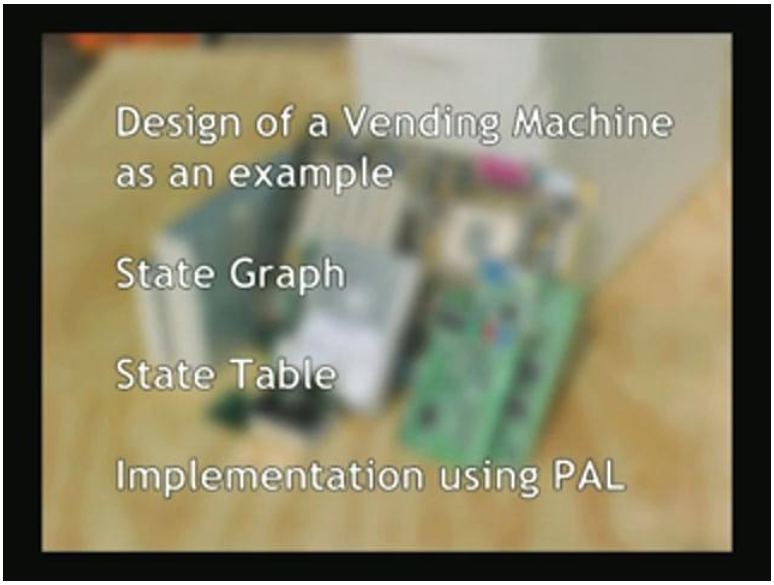
Design of Sequential Circuits One-hot Controller

Slide – Summary of contents covered in previous lecture.

(Refer Slide Time: 01:05)

Slide – Summary of contents covered in this lecture.

(Refer Slide Time: 01:35)

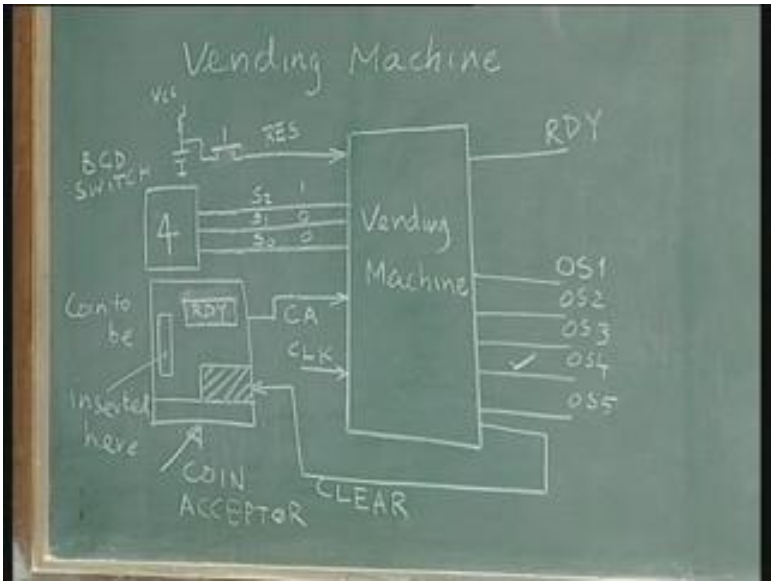


In the last two lectures, we have seen how to design sequential circuits, and also how to implement the design, using MSI components like multiplexers ROM, PAL, PLA etc. for the combinational part. As we know the sequential circuit will have a combinational part and a sequential part, the sequential part is usually the Flip-flops Registers and the combinational part can be implemented using any technique that you are familiar with. We can use gates. In the first lecture, we saw the methodology of implementation where, we only used gates for the combinational part of the sequential circuit design. In the second lecture, the last lecture, we saw that the combinational part can be replaced by the MSI (Medium Scale Integration) components such as multiplexers ROM, PLA etc. Now I want to take that design procedure a little further and show you how to fit in a design of a sequential circuit in its totality.

For example, take a system. In the earlier example we just took equations and implemented them. Now, we will take a whole system and see how a system is implemented. This is how in practice we will have to do. In practice, they will give you a design, a system to be designed with input-output specifications properly drawn and you ask the questions to get a clear picture of what is expected of you. Then you go to the same procedure and then try to find hardware to implement the design. When you do that we will try to choose a hardware which can best fit in the design. You can best fit the design in the smallest of the simplest hardware that will be used to fit in the design

adequately. It should have all the features that we want but, it should not be an overkill for the problem. From this point of view, I thought I will take one final example, the PAL design and that will be explained. That is why I call it a system design example.

(Refer Slide Time: 05:55)



This is a vending machine. The system we are going to design is a vending machine. We are all familiar with the vending machine. You have this machine where you have this slot in which you put a coin and you get a service: a coffee, tea or some candy bar or a can of a soft drink. This is a very common thing and can be very easily done. It looks like a complex device but in practice it is not. In reality, a vending machine is a very simple concept. I will give you the concept of how to go about designing a practical system and how to fit it into an existing hardware, like a PAL. What are the features? To make this design simple again, in a class room lecture; a lecturer always has to make it simple to avoid getting into the complexity of the hardware. In order to drive home the concepts and the points, it is better to keep the system design simple.

We are going to design a vending machine, which will dispense let us say about 5 items. The 5 items to be dispensed are the output. See how these things are done. There will be a sort of a mechanical device which holds the item to be dispensed. Some sort of a relay has to operate and release the mechanical device so that one item will fall into a tray and pick it up from the tray. Suppose if it is a candy, 5 different types of candy bars will be kept in 5 different trays which are held in place by mechanical devices tightly. Just by

operating a relay you release one of the holders and the content of the holder falls on a common tray and you pick it from the tray. That is how it is. It can be a candy bar, it can be some soft drink or it can be different types of merchandise, fruits or whatever. To make it simpler I am going to make it all of equal cost and also by a simple coin operation.

We have a coin, a 5 rupee coin. You insert a 5 rupee coin into the slot and if it is a proper coin it will be recognized as an input and any other coin other than 5 rupees will not be recognized. It will be rejected and there is no provision for giving change. Suppose you do not have 5 rupees change and you put in a 10 rupee note, it will not accept it. I am making these extra restrictions to make design simple; so that we will concentrate on the design aspect. These features can be improved later on; once you can understand the concept behind the design of these things you can always improve the features by incorporating all these additional things. In order not lose track of the design concept, I am making it simple. I am going to have a separate coin acceptor. This is a unit which is independent of the machine. I am not even going to design this. I am going to assume that this is available, that is a coin accepting mechanism is available, which is a slot into which we insert the coin. You can make it a 5 rupee coin or a 2 rupee coin, depending on the type of, that is what you want to sell. If it is a candy bar you would probably like to have 2 rupee or a soft drink may want to have 5 rupees, whatever beverage.

Once the correct coin is inserted then the optical mechanism which is not going to the part of this design. These things are available, you can buy them readymade. There will be an optical mechanism and you can set it for 2 rupees, 5 rupees, whatever. Or if it is multiple coins also, you can make it one at a time. Suppose I want to put change for the 5 rupees – 2, 2, 1 it can keep count, all that I am not going to discuss here. I am going to assume that there is a coin accepting mechanism. Maybe the coin can be inserted here and then once the coin is ready it will say Coin accept. It will give a signal, and I will call the signal Coin accepted (Refer Slide Time: 10:10). That means only with the Coin accepted the vending machine will work. Otherwise the coin will fall and there will be a a small tray here with coins that fall; the coins which are not accepted. If it is a false or not a correct coin, or if it is not a coin at all, and it is a metallic object or, maybe a stone; some people due to mischief put a stone into this, in such cases it will fall into these trays

and you can take it back. Only if it is a right coin it will go in to a safety box here (Refer Slide Time: 10:51).

Inside it will get stored here, like a collection box we find in temples. It will go into that, keep it there and then it will give the signal. If it is not a valid coin, it will come out here and has to be removed, taken away and that is only signal is expected of this machine. This design is independent of these machines. I am assuming this is available. You can buy this integrated optical mechanism and all those things. Of course once a coin is accepted you have to make a selection, only then the selected option will come. When the selection is over and when the coin is accepted, there should be a signal received back from the machine showing that this has cleared. I should have a clear signal, so that next coin can be accepted. Until the clear signal is exhibited it will not accept the next coin. Only when the clear signal comes, this Coin accept signal will disappear and the machine is ready for the new coin. You can have some sort of a clear that is linked to ready lamp this can be a ready lamp (Refer Slide Time: 12:30). This can also be outside. The vending machine can have it outside. Only when the ready lamp is on, the clear can control the ready lamp and only when the ready lamp is on it will also be ready here.

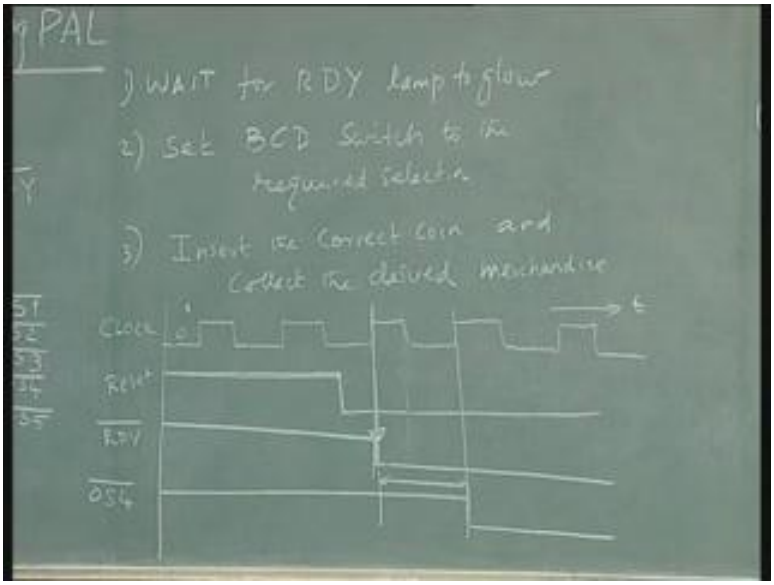
The vending machine will be ready along with the coin accepted and both of them will be ready. I need the ready output here also (Refer Slide Time: 13:12). So these are my 5 dispensable goods. I will call this output as output select1, output select2, output select3, output select4, and output select5. I will have to make one of these selections. Then there will be a selection mechanism. The vending machine needs not only coin accept signal but it also needs signal for which of these 5 objects, which of these 5 merchandise I want. That will be given by a separate switch setting for 5 objects. I can code into 3 bits; I can have a LED display here or I can have a BCD switch (Refer Slide Time: 14:10). I can set this switch to the position I want. This switch will give me the combination of what is required. If it is a 001 it is OS1; if it is 010, it is OS2; 101 it is OS5. What I want will be given by this switch here (Refer Slide Time: 14:35).

Suppose it is 4 I set here, it becomes 100, this is high, this low; this is the most significant and becomes 100 and this OS4 is selected. That is how it is. Of course, you need a clock for this and a reset signal. Now the whole thing is starting. All of them should be off and once we reset, the ready will glow and when the ready glows, this ready and this ready

will be combined. These are the same signals that I am actually showing in two places (Refer Slide Time: 15:30). This ready and this ready, both glow. This will accept a coin and then the procedure is you set the switch first and then put the coin in. The coin accept signal will glow and then the corresponding thing will be dispensed and the clear signal will come and the clear signal will keep ready again. Then it is time for setting one more selection and one more coin. This is the procedure for the operation of this. Though reset can be done in different ways, you can have a reset by a switch pattern and you might have seen this by a simple switch. I am not going to talk about the microprocessor circuitry, as you know how to reset if you have a simple RC circuit which can do this job. Normally this is high (Refer Slide Time: 16:38), this $V_{cc.}$ When you short this what will happen? This capacitor gets charged to full value. When you close the switch, the voltage across the capacitor apply to this reset position and keep going on for a while, and then that is the indication for the reset.

This is a normal procedure we adopt in many circuitry for Power on reset they call it. This is how you put the button to get the reset signal. The clock has to be very, very slow because it is a very slow process. There is no point in doing this at a very fast rate because the signal has to be recognized and the vending machine has to dispense. There is no hurry whole about the whole thing. You can have a very low clock but then the clock cannot be very, very low because it very difficult with extremely very low clock frequencies, that is it is very difficult to generate because capacitance becomes very large in 555 timer. All of you are familiar with 555 timers. 555 timers require large capacitors with low frequencies and also with low frequencies waveforms are generally not stable. It is very difficult to have a very stable waveform with low frequency. Otherwise, there is no need for these frequencies to be high. So we can have anything like a few hertz will do, something like 2 hertz, 3 hertz.

(Refer Slide Time: 18:13)



The way it works is as I said, the mechanism is: wait for ready lamp to glow, set BCD switch to the required selection, Insert the coin there is only one possibility, insert the correct coin and collect the desired merchandise. The procedure is very simple, so what we are doing is, the clock will go on (Refer slide Time: 19:44) Let us say this clock 01 time; initial reset - when the reset is on we will take several clock cycles. The reset is a non-synchronous operation and it can happen any time because you are going to push the switch when you reset. (Refer Slide Time: 20:28) They are going to push the switch and then it is going to give the reset signal momentarily and it can happen any time and the next clock cycle ready will go. At the next clock edge, this will be a ready signal; we can have ready signals as active low signal, it is alright (Refer Slide Time: 20:57). This will be Ready signal. It can happen only at this point (Refer Slide Time: 21:09). Even though the reset has gone low here, only at the next clock edge we may have a Ready signal available. I am going to call it Ready Low; the Ready lamp will glow when it is low. We can have the LED connected low or whenever the voltage is high or you can have a LED connected when the voltage is high. You may have seen these in your earlier circuits, earlier course in digital design. You can make an LED on for 1 or off for 0 or you can make an LED on for 0 and off for 1. By connecting the anode and cathode proper polarities, we can always make it glow for 1 or glow for 0.

In this case we are choosing RDY to be low. When the signal is low, the LED will glow, so this lamp will glow. After the output, it takes some time because you have already set the Ready. After the RDY signal, only then you are going to set the selection. Then we set the coin so it is going to take a while. I am not going to show the delays.

After sometime the output will go to the selected output. After the Ready, this is the delay required for processing and again this has to be synchronous (Refer Slide Time: 22:55). I am just saying in 1 clock cycle; it does not matter; it can be 2 clock cycles or can be 3 clock cycles. It can be part of a clock cycle but then action can always happen in synchronous logic, in a synchronous design the action can always take place along the clock edge. The delay from the Ready signal to the time the merchandise will be dispensed will depend on the circuitry. If it is less than a clock cycle period, the action will take place in next clock edge, if it is more than a clock period, let us say, somewhere here it takes (Refer Slide Time: 22:32), then it will go here. I am just giving estimation; we do not know the logic inside. Assuming it takes to make a 1 clock cycle the next clock edge you are going to have the output generated here (Refer slide Time: 23:49). Again for some more reason we can also have an active low or active high, so we will have this signal active low. I will say this is the merchandise 4 since you have selected 4.

Since your selected merchandise 4, OS4 will go low at this point, meaning the availability of the merchandise. Then at the same time, clear signal will come and clear the... even if you are not taking the merchandise, there is no way you can find out whether you have taken the merchandise - the can of soft drink or the candy bar is in the tray now. It has been released from the mechanical holder and it has already fallen in the tray. Whether you take it or leave it, the machine is ready to take the next turn. If you do not take it, it is not the machine's fault and you have to look for it; once it falls, it usually makes a thud noise. You may have probably seen those vending machines. Once this can falls in this tray, there is a noise, a sound and you know there is a fall, so remove it.

Even if you do not remove it, it does not matter because the machine does not know you have not removed it. There is no intelligence built into the machine, so as soon as the machine has dispensed the can, it will clear the signal and the next Ready will go on and it is Ready to accept the next input. This is the logic. We are given a problem statement; it has taken us so long. That is the beauty of the design. You will have to be very clear

about what you want. The conceptual design, the concept of the problem of design, also of the various inputs and outputs, what is the interaction between them, what is the way in which you want the machine to work and what are the signals and how they are going to interact? Everything has been very clearly mentioned, only then you are ready for the design.

There are a couple of minor points I would like to make on the original specifications on the vending machine .We earlier discussed that the number BCD corresponding to the selection that you want. Actually what happens is, since we have 3 bits s_0, s_1 and s_2, when all of them are 0, the first selection is made when all the bits are 0 0 0. That means really speaking BCD 0 should be given the first selection. BCD 1 should be given the second selection; BCD 2 the third selection, 4 the fifth selection, for 4. So actually what I am saying is when you put a 4 here, it is 1 0 0 but since this is 0 1 2 3 4 this is not selected OS4 is not selected; OS5 is selected (Refer Slide Time: 26:55). For example, if you want to select OS1, you should put 0 here and 0 will make 0 0 0, OS1 will select. This is only a minor matter. We could have done it the other way. We could have retained this and then called these signals OS0, OS1.

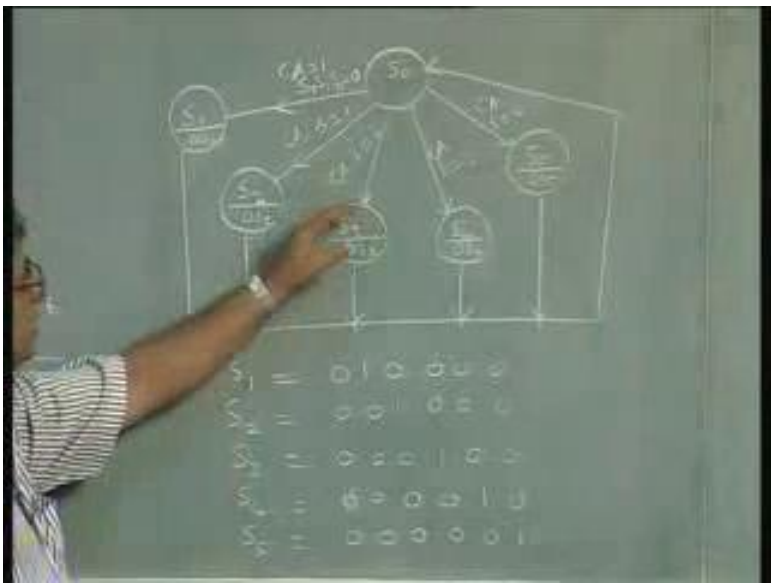
We could retain this as it is and made a combination for 0 0 1, 0 1 0, 1 0 1 etc (Refer Slide Time: 27:16). That is alright, it is a question of convenience. The second thing is about the reset. We set the reset and it can be done once the power is switched on. It is a power on reset, actually. When you first switch it on, the power is switched on and there is a V_{CC} and a capacitance, it is an RC circuit. The capacitance will get charged but then it takes a while and this is a Schmitt trigger inverter (Refer Slide Time: 27:44). There is a Schmitt trigger which does the inverting. So when the capacitor is low, that is when the capacitor is not charged to a particular value and the threshold value, the Schmitt trigger inverter, this will become, this will be low and this will be high. That is the initial reset.

The initial reset as the power on, the reset input will be high; but, once a capacitor has reached a particular value - the threshold of inverting the Schmitt trigger, this becomes high input, which means this will be low (Refer Slide Time: 28:19). So reset will be low and that is where there is normal functioning. Reset high is a period where it will not accept the signal from the input or the coin Ready and once the reset is high only then the Ready signal will go on. That is why the Ready signal has been shown as coming after

the reset is over. It is asynchronous; the timing of this depends on the timing of the charging of the capacitor.

RC time constant decides the time for which the reset is going to be high. Once the reset is low, it may not happen at the clock edge; after that the next clock edge Ready signal will go low and that is the time when you can put the coin and then set the combination of the coin. The reason I told you this Ready, OS1, OS2, OS3, OS4 and OS5, all these things are active low. There is no need for them to be low but then there is a practical reason. When we show you later on the implementation using the PAL, I will tell you why the signals are low, as it can be low or high. These are the minor explanations I want to give on these before proceeding further. Having given these specifications and having understood the timing sequence we draw the state graph as that is the procedure.

(Refer Slide Time: 29:39)



This is the state graph. A simple state graph SR, the reset state, no output and whenever the CA is 1, the Coin Accept is Ready. Only with the Coin Accept being high any activity will happen because when the Coin Accept is low this is going to wait. If nobody puts a coin in, the machine has no work. If the Coin Accept is low the machine stays in the reset state. If the coin will be accepted - somebody has put in a coin and it is a right combination, the right combination of coin or right combinations of the coins, CA 1 and then depending on the selection made as I said s_2, s_1, s_0 if it is 0 the first choice is 1 0 0 1, the second choice 0 1 0, the third choice 0 1 1, the fourth choice 1 0 0 and the fifth

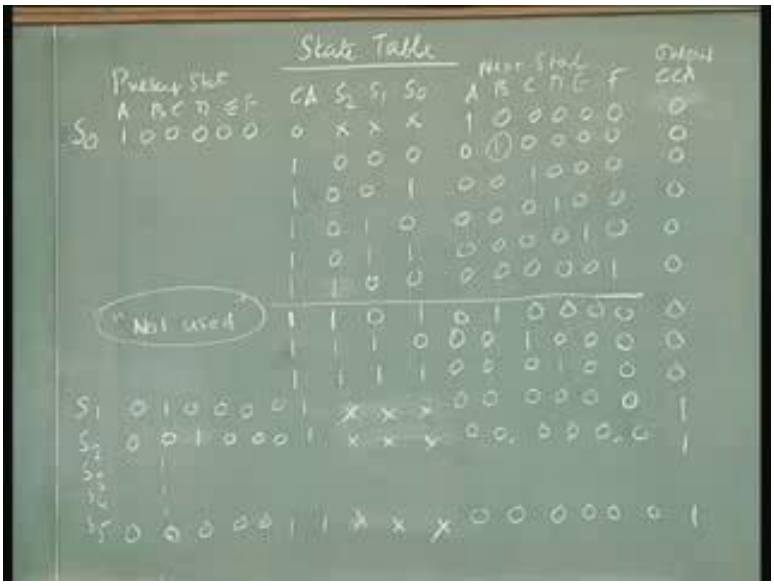
choice. So based on the choice and CA being 1 it will go to the next state, the state being s_1, s_2, s_3, s_4, s_5 for each of the combination of OS0, OS1, OS2, OS3, OS4. That is the reason we called this signal [Refer Slide Time: 31:00] OS1, OS2, OS3, OS4, OS5. Even though according to the binary value here BCD switch 0 1 2 3 4 at the 5 values, we have called it 1 2 3 4 5 because here these are five states, that is state 1, state 2, state 3, state 4, state 5 and state 0 is being the reset.

Now let us say we have the combinational CA and S is equal to 2, S_3 will be selected. To make the design simple; I can make an arbitrary assignment there are only 5 states; 3 plus 2, 5 plus 1, 6 states. The 6 states can be realized using 3 Flip-flops; we can call it Flip-flop ABC. We know how to do it as we have done earlier in the designs of combinational sequential circuits using Flip-flops and number of states which is served by the number of states decided by the number of Flip-flops. In this case you make the design simpler also because we have the PAL which has more number of Flip-flops available than we want. We have decided this assignment which is very simple. We will have one Flip-flop active for each state; if all the Flip-flops are 0 – not active, then it is a reset state and any one of these Flip-flops active corresponds to that state. Let us say there are Flip-flops, s_0 should be 1. How many Flip-flops are there? 1 2 3 4 5 6, there are six Flip-flops .We will call these Flip-flop variables A B C D E F.

I am going to use 6 Flip-flops in these 6 states (Refer Slide Time: 32:49). This technique is called one-hot assignment. The reason is, one of the Flip-flops is always active; hot is active. One of the Flip-flops is hot, others are cold. This means one of the Flip-flops is active and the other Flip-flops are inactive. The state decides the Flip-flops; depending on which Flip-flop is active you know the state it is in. If the Flip-flop A is active we know the state is s_0, Flip-flop B is active we know it is s_1, C is active and we know it is s_2, D is active we know it is s_3, if E is active we know it is an s_4, if F is active, and you know it is s_5. This is assignment of assigning 1 Flip-flop per state, as against the number of Flip-flops decided by the number of states, in the logarithmic fashion. For example, for 8 states you need 3 Flip-flops and you have 8 combinations. This is a normal assignment procedure. This assignment procedure gives the simpler design. If you have more Flip-flops than you need in any system, we will use this method because it is simpler. In our case, we are choosing a PAL with a large number of states, a large number of Flip-flops

and since we do not have enough number of states, we are going to assign 1 Flip-flop per a state. This is the one-hot assignment procedure. That way, I am going to make the Flip-flop A active for s_0 state and output of that will be the s_0 output. Whichever Flip-flop is active that can be the corresponding output also; that is the advantage. If s_1 is active, that means you are in state s_1 which is OS 1 state. This means the output of Flip-flop B can be the output OS1; that is the advantage of this method.

(Refer Slide Time: 34:40)



Now you are to write the state table for this. You know how to write state table. I will do it in a simple way, quickly as it s a long table because there are so many Flip-flops. So present state s_0 1 0 0 0 0 0 0. These are Flip-flops ABCDEF (Refer Slide Time: 35:07). You know that signals are CA S_2 S_1 S_0 these are the 4 signals into the vending machine. CA- (Refer Slide Time: 35:23)… reset is only once; after that, the machine is working and the clock is also not to be considered. These are the 4 inputs to the vending machine which decide the state transition. What is the present state, what is the input – that decides the next state and that input is decided by the 4 inputs. M is the Coin Accept and 3 signals correspond to the choice of the beverage or the merchandise. They can be if it is 0, the remaining state s_0; next state remains s_0 and output is Coin Accept that means, this signal will clear as long these signals are coming, the signal is also called CCA that means, coin will be accepted.

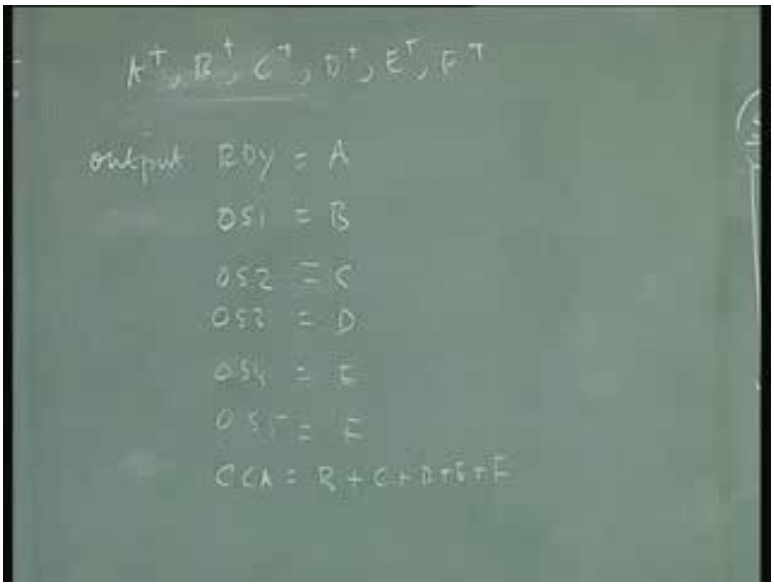
Once this is clear, keep it Ready; but this also 0, output CCA will be 0, no signal will come. We are still in this state Ready signal is already there.

Now you are in state s_0, CA is 1; there can be 5 combinations 1 0 0, 0 1 1, 0 0 1, 1 1 0, 1 0 0; and then if it is a 1 0 1, this is not going to happen. You are going to be in state A and this is going to be in state B, state C, state D, state E and state F (Refer Slide Time: 37:56). Now all others are 0, because in the one-hot method, as I told you there is no need for an output separately written because as I said the output is the same as the Flip-flop states. If you are in this state, this also is the output B; this is next state as well as the output and the other output to be given is when the transaction is completed. That will be given only when you are in one state and then you go to the next state. Now what will happen to these inputs - 1 0 1, 1 1 0, 1 1 1, which were not used?

There are two ways of doing it. You can make them as Don't cares. If you make them as Don't cares, what happens is the design will become flawed because some random number we will come and also possible there is a glitch occurring. If you design the state machine, sometimes these states - one of these non-used states may lead to some other state which is the permitted state and then it may dispense merchandise which has not been selected. The other way to do it is to use this also as…. hold it back (Refer Slide Time: 39:26). Suppose by mistake you input 5, you can make it the same availability as 1 or something like that. You can say for example, 5 can be made as s_1. You go back to s_1, that means I can make it 1 and this can be 2, 3 (Refer Slide Time: 39:51). If you make a 5 combination, the same output as 0 will come 0 1 2 3 4 5 will give you the same output as 0. If you make a 6 combination, it will give you the same output as 1, 7 combination of course, these 3 combinations can lead you to 3 accepted combinations so you can either put a board here (Refer slide Time: 40:29), you can make a mention here for the beverage OS1 as a try as I said. You can say select 1 for this, select 0 for this, you can say select 0 or 5 for this, select 1 or 6 for this, select 2 or 7 for this; 3, 4 and you can leave (Refer Slide Time: 40:45). Otherwise, automatically it can come; you make a choice of 5 or 6 which is not permitted. There are only 5 beverages, so you put 0 1 2 3 4 5, make a choice of 5 and put a coin. Without your knowledge it will come and you may not like it. There are 2 ways of doing it and you can say by default 5 will give this, 6 will give this, 7 will give this and that is without the customer's knowledge.

With customer knowledge, you can put a label here saying that this will give this and can be got by pressing 0 or a 5, 1 or 6, 2 or 7, whichever it is. Then the next state is s_1; I am not going to complete this table, you can do it yourself. Whatever happens, it goes to the next state. Only when you go to the next state you are going to produce the signal. We are not going to produce the signal; the clear signal - CCA signal will be generated (Refer Slide Time: 42:00) which will clear the coin and then make it one more Ready for accepting the next coin; this ready maybe, these 2 may be tied together; only when we go from this state. After dispensing the beverage, it will go to the next state so this CCA will be 1. That is only when s_2 then s_3 s_4 s_5; these are Don't cares (Refer Slide Time: 43:31). Hereafter, you can write equations and then simplify it; instead what we are going to do is, to directly go to PAL realization. You can write equations for A plus in terms of this A and this inputs, B in terms of this and inputs, C in terms of this and inputs, and etc, etc. For each of the outputs, as I said, the outputs are same as the states as for ABCDEF are concerned output which is not inside, outside this is a CCA and then Ready is same as A, so the output should be A…(Refer Slide Time: 44:38).

(Refer Slide Time: 44:33)



The state equations I am skipping as you know how to do state equations: A plus, B plus, C plus, D plus, E plus, F plus and then the output should be Ready signal is A**.** The state s_0 will have a Ready signal showing and OS1 the output will be B, C will be OS2, will be D will be OS3, OS4 will be E, OS5 will be F and clear will be when CCA will be any of

these B or C or D or E of F. That is what it is because the CCA signal will be B or C or D or E or F. This is CA 1…I think I made a mistake here in this table; this should be 0 (Refer Slide Time: 46:15) any of these signals will lead you to the next state.

(Refer Slide Time: 48:57)

The design we have now completed can be fitted into programmable array logic. The choice of the PAL is important. We need registered outputs and mainly all these are registered outputs - states. We know that B C D E F, all of them are registers states and there are outputs are also state outputs. We need registers and also need a combinational output; because, the clear signal etc. require CCA, which requires a combinational output B plus C plus B plus E plus F. We need a device with registered outputs as well as combinational outputs and enough number of inputs. One choice which is 16 or 6 is a medium 20 series PAL with 16 inputs 6 registers. So 16 inputs means there are (Refer slide Time: 47:17) 8 inputs you can see here. 9 inputs on this side - on the left-hand side and on the right-hand side the input number… see this is a 12 input, 12, 13. These are all - 13, 14, 15 16, 17, 18 these are 6 registers. Now how many do we have 8, 9 plus 6, 15 and that is the programmable IO, which is 1 here and 1 on the top; there are 2 inputs; so totally it comes to 16.

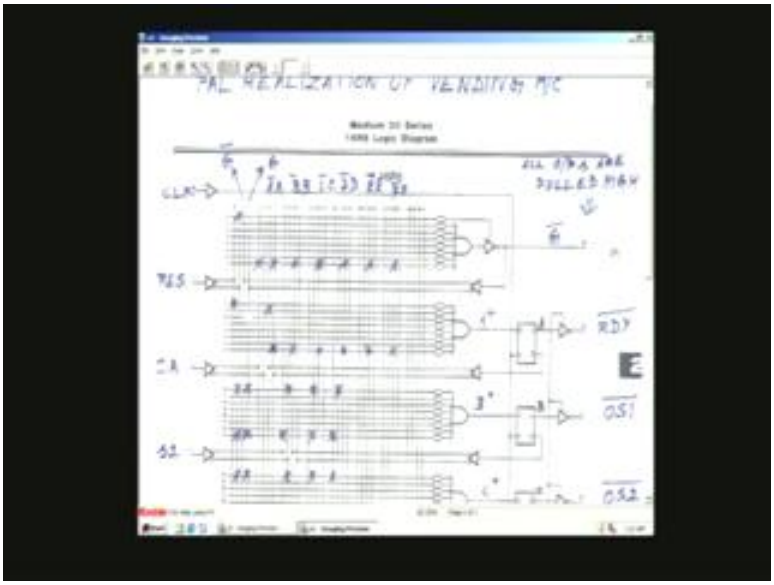
I do not know whether it comes to 9 plus 6 equals 15. 6 registers, 9 inputs, actually the input number 1 is not counted as a clock; so input 2 to 9 or 8 inputs, 6 registers, 14 and another 2 I/O inputs. There are 6 registers as I have already shown you. Let us see how to

fit this into a design that is the device. This is how you have realized the vending machine using these six outputs OS1, OS2, OS3, OS4, and OS5 are the outputs of this Flip-flop? Now you know why these are negative signals because there is an inverter for each of these Flip-flop outputs. That is why it is a negative output here and then there are only 5 of them. The sixth register is used for Ready signal. ready the signal, we know that we got this from the combination logic which is in the state A. State A B C D E F are the 5 states for dispensing merchandise candies or candy bars or soft drinks.

A is the reset state as you may know; corresponding to that and you should have a Ready signal. The Ready signal also comes out of these Flip-flops. So A B C D E F are the six registered outputs. The corresponding outputs are bared RDY OS1, OS2 etc. bar and inputs are of course I have seen earlier and you have shown reset input which comes from that Schmitt triggered combination of the RC combination. The clock is given to the first then S_1, S_2, S_3, 3 inputs and no connections. The other connections are not used. NC means No Connections and we are not connected. NC is connected and CCA is as I said the combinational output of B C D E F. So, B C D E F, are also fed back. B C D E F are the Flip-flop outputs; OS1, OS2, OS3, OS4, OS5 also fed back as inputs so we can get this CCA output. From this again there is a negative logic here; there is an inverter, the output of the OR gate so its CCA bar and reset is given here because it resets all the Flip-flops (Refer Slide Time: 51:00).

The reset is given here as the input; there is a reset input here also - the reset input here that acts as the same as this. This also is used for that purpose. What is this reset? This reset is also used the same as that. One final here, you have the z bar which is not a term which is output or input but z bar is an intermediate term. Since, we did not write the individual equations for the A plus B plus C plus etc. If you write them there will be a combination which is common to all of these; that term we call z bar and we realized. This combination is applicable to all the inputs. Hence z bar input is taken as the input to all the other registered outputs. This is how it is implemented. This design shows you how to fit in the given specification fully utilizing the capabilities. 6 registers and you have used all the 6 registers and we have used combinational outputs.

(Refer Slide Time: 52:26)



We are used to many of the sequence combinational inputs except a couple of them and most of the gates of use, one of the optimum designs you can say. This is the way in which you specify, go through the design specification, write the state equation and then implement it using PAL. Of course, there are automatic software tools now available where you give the specifications. The tool itself selects the proper hardware and then fixes it to the design. I am doing it manually for you to understand the whole concept. This concludes the design of the review of the combinational logic and sequential logic, you can design using a MSIs and PLDs etc. From the next lecture, we will be talking about implementing some other things using verilog hardware description language. Thank you very much.