Digital Systems Design with PLDs and FPGAs

Kuruvilla Varghese
Department of Electronic Systems Engineering
Indian Institute of Science – Bangalore

Lecture-31
Evolution of PLDs

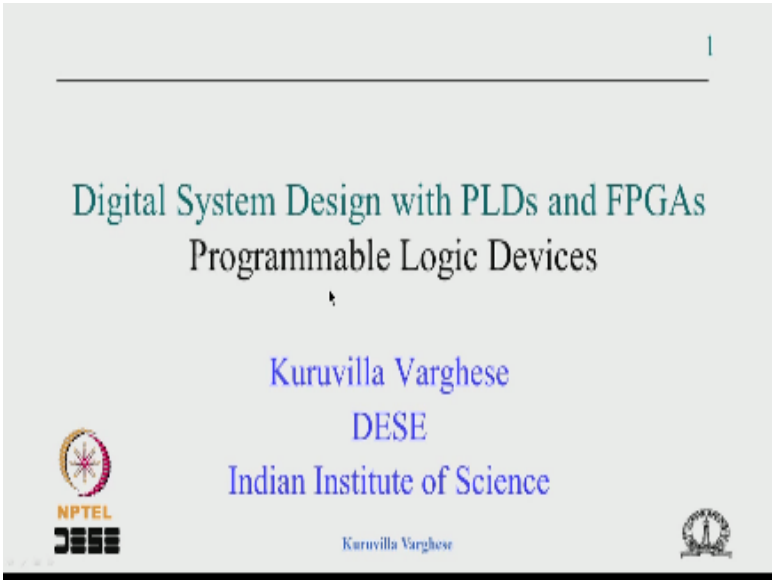
So welcome to this lecture on programmable logic devices in the course digital system design with PLDs and FPGAs. We have completed our advanced digital design part, I have covered almost everything, there were design controller design, issues in the controller design, then some essential topics on synchronisation, top down design, everything is completed. VHDL we have as far as the synthesis is concerned we have completed everything pretty much.

What is remaining is maybe that has test bench then we have to cover to write the test bench for simulation, we will do that and that 2 topics are left, one is programmable logic devices and field programmable gate arrays and we will next few lectures we will complete try to complete the programmable logic devices or PLDs and I must tell you that there are two types of devices in this PLDs.

One is SPLDs and CPLDs one is simple programmable logic devices and the second one is complex programmable logic devices. SPLDs are very very rarely used, but it is better for an academic interest go through that part. So that when it comes to the CPLDs, it is very easy to understand and also maybe that gives you a little bit of all these devices are evolved maybe it is useful.

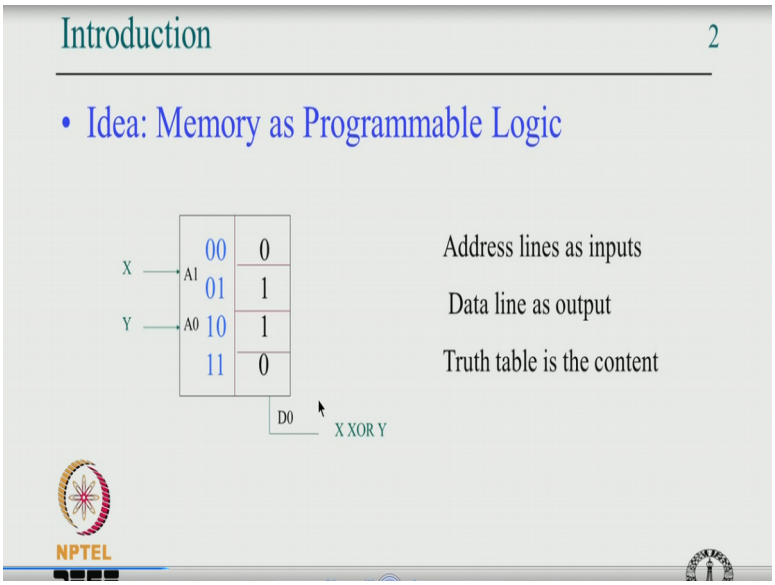
I mean I am not very sure about it, but CPLDs have some use at least not that is not as useful as FPGAs because nowadays the designs of very complex and people try to build lot of things into a single chip to that extent CPLDs are not much but they have to have a place in the keys of low and medium convexity design and they have some advantages and disadvantages which we will see. So let us start with the programmable logic devices, so let us look at the slides.

(Refer Slide Time: 03:03)



So we are going to pull together the programmable logic devices which are called PLDs and as I said that there are 2 parts, 1 is SPLD and CPLD.

(Refer Slide Time: 03:14)



Basic idea is now when you look at the evolution this is started this idea of the programmable logic devices has started quite some time might maybe 25 to 30 years back and there were I means the VLSI circuit density was not much, people needed some kind of programmable logic and earlier this designers used to work with discrete gates and discrete functions like multiplexers, demultiplexers, encoder, Decoder.

So when you had normally a logic design it was a printed circuit board full of small integrated circuits ok, which was not very complex you know you will have some gates, you will also have a multiplexers, encoders, decoders as we the design was pretty much saying

what we have seen, but it was implemented not in a single chip plenty of chips, it is huge number of chips ok.

So you should when we talk about the revolution you should think of that scenario not today's scenario. So there was some need of programmable logic the first thing comes to mind is the chips select decoding in a microprocessor based design that means in a microprocessor you have lot of memory and peripherals, again those days it was not there be multiple memory chips to cover the memory space, multiple peripherals.

All these need some kind of you know the memory map know everything is map to the memory map of the process and that is done by chip select decoding by the higher address bits and that some kind of flexibility was required in kind of one mapping these devices in the appropriate places depending on the need ok. So there was at least for a chip select decoding there was some kind of programmable key was required.

So I think that was a kind of starting point of this programmable logic that, so the only program and then available was the memories ok, what is called programmable read only memory, so that is a contact so basically the idea coming back to the slide the idea is how to use memory as sine programmable logic like you know like your standard memory, how to use it is a programmable logic that was the basic idea.

So let us clear that part, so that your kind of not lost in all the old technology because if I show a picture of one of the PROM kind of architecture then you will be wondering what is going on. So let us read this idea suppose we take a 2 you know variables x and y and you want implement this Boolean function of X and Y which is nothing but x xor y ok. So do not confuse with this these to access.

This is variable, this is a function and this is y ok. Now idea is how to use a memory to implement this x xor y, that is basic idea and we take a 2 address line memory that means 4 location with us, and that means we take a 4x1 memory and you know that there are 4 location and when address line is 00 the zeroth location is selected, address line is 0,1 first 1,0 second and 1,1 third location ok.

So internally there is an address decoder depending on the value put here appropriate location is selected ok for writing or reading. Now assume the truth table of x xor y is programmed in this 4 locations ok. So basically x xor y is the function is either of the input is 1 then the output is one. So if both inputs are zero that value is 0, 0, 1 that means this is x bar y is 1 x y bar is 1 xy is 0 ok.

So what we are doing is that we are programming that the truth table into these 4 location, that is a 2 address lines are connected to this variables ok and this is some program, we just write that truth table into this location. Now you check what happened, suppose we give 00 as the as the input value, the writing is kind of part of the programming this memory with the truth table.

And when you want to kind of used as a logic what you do is that you put this memory in the read mode. So I am not showing that I have assumed that it is written and this is already in the read mode ok. So the logic prox when the memory is in the read mode, when the memory is in the right mode we write the truth table okay and I am not shown I am not showing the writing part which is understood.

So if you give 0,0 to x and y then this zeroth location 00 location is axis and you get a zero here and if it is 0,1 a it access this location and then 1 comes on the data line. So the data line is output, address line is input and again you give 1,0, then one comes here 1,1 0 comes here, so basically you take a memory of the required capacity that mean suppose you want to implement of a variable function then you take a memory with 5 address line or 32 location into 1 bit, 1 data line you require.

Connect all the variables to the address line, 5 variables to 5 address line in which our order does not matter but then you have to appropriately program the truth table ok. So you program the truth table, then use the data line as a Boolean function the output of the logic know that is what is the required. So that is how we use the memory as a kind of programmable logic, address lines are the inputs, data line of the output.

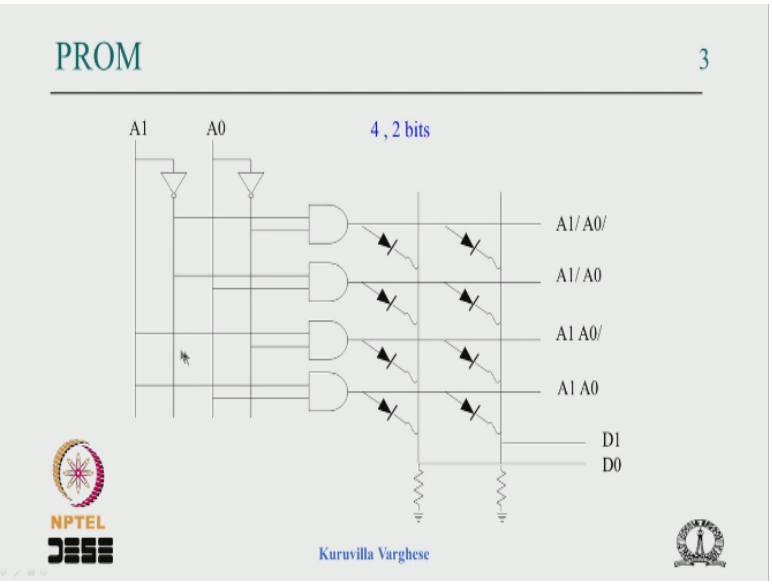
This is while working as logic, truth table content of course when you write the truth table into this memory then this data line is definitely the input, you not output ok. But in operation

on the logic is an operation address line is input data line her, the output. Now if users kind of static RAM which is kind of volatile each time you power up the locations are not defined ok.

We do not know what is the content of the location it could be zero one because these are kind of you know the lattice inside sales are nothing but kind of course couple latest will come up with some random values. So you have to write, so you cannot use a SRam as a programmable logic you can use but have to write it each time but so if you again once again go back to the old scenario that was not kind of possible.

And so some kind of non-volatile t was required that means you program it once so all maybe so but then it should retain these truth table inside ok. So such a device was available then that is called programmable read only memories or called PROM ok.

(Refer Slide Time: 12:01)



So the architecture of the PROM was like this and we are showing now exactly similar one that means for location 2 address line is 1 data but I am showing 2 database, it is only difference. So this was the architecture of the PROM or programmable read only memory ok. So that is called programmable read only memory. That means you can once you program it you can only read it you cannot write it.

So you can say the right one send me any time memory or something like that. So you have here a 4 location memory but 2 bits output and if you see there are two address line A1 and A0, A1 this is a complement of that, A0 complement of that and this part is the address

Decoder. So this line when both are 0,0 A1 bar A0 bar this AND gate is 1. This is A1 bar and A0 this is 1, so its chooses, these are the minterms.

And sorry A1 A0 bar A1 A0. So these 4 AND gates or the 4 locations ok. Now take one data line and their connections okay like there are is a diode and a fuse and this is a normal fuse which can be blown by passing a larger current ok. So there are 4 connections 2 diode and we will see why the diode is required and this is we are discussing the whole architecture.

And this data line is pull down to the ground with full resistance and this is output okay, if you need a buffer you can imagine a buffer here and similarly a any number of output can be there, which allows you to implement multiple outputs ok. Now assume you want program your xor function that means 0,1,1,0 ok. Now what you do is that you blow the fuse you will see how to blow the fuse.

You just blow this fuse and retain this connection and blow the fuse ok. So it is equal to do programming 0,1,1,0 exactly like here 0,1,1,0 because we are trying to implement the xor of the input ok. Here we have to import we can assume it is X and Y. Now like if you program 0,1,1,0 you see what happened to give 00 here, then this AND gate is 1 and rest all 0 ok. So the moment it is zero this diode on contact ok.

And this one but you see that this fuse is blown ok, so this there is no connection here and it is pull down you get a 0 so but if it is kind of 0,1 then this and get output is 1 rest as 0, so since this is 1 and this is pull low this diode will conduct the fuses retain, so you get a 1. Similarly when you have 1,0 this diode will conduct the fuses in that 1, and when it is 1, 1 this is active rest all are 0.

But this fuse is blown, so it one conductive to pull low and then you get this ok. So you can say that this is acting like a programmable or ok, so we are programming the OR function we can imagine this is AND and an OR ok. So this is a fix that you have to inputs, so you have 2 raise to 2 minterms A1 bar A0 bar, A1 bar A0, A1 A0 bar and A1 A0. All the minterms are there.

So suppose you have N inputs then you can imagine there will be 2N vertical lines and inputs and their compliments and there will be 2 raise to N AND gates here, each AND gate will

have N inputs ok, because you need to choose an input or its complement, so for an input case there will be 2n vertical lines representing N inputs and its complement and there will be 2 raise to N gates representing all the minterms.

And there are each and get there will be N input selecting the appropriate minterms ok. Now you program the truth table by blowing the fuse ok. Now the next before we see how to blow the fuse, what is the purpose of this diode ok, you might be wondering why diode is required because it is definitely not to do anything do the zero because when you want program is zero you blow the fuse.

There is no connection at all. So it is something do with the connections you are retaining, so in the case of XOR gate we have programme 0,1,1, and 0 that means these 2 fuses are blown they are not there, these 2 are retain ok assume the input is 1,0 ok, when the input is 1,0 the AND gate this AND gate is high one and this AND gate is low 0. Suppose it is not diode as a connection then you can imagine that what happens then there is a direct connection to the fuse.

One output is one other output is 0, so there will be a short circuit, directly from the VCC to ground, VCC of this through the Pos are then maybe earlier it was a kind of BJT is used but does not matter but then there will be from VCC it is coming through that and it goes to the ground to the transistor pulling lows there is a short circuit, so this diode are you know kind of block that zeros.

So that there is no direct connection between the VCC and the ground. So this is kind of 1, 0 then this is 1, this since it is full of low, this diode will conduct the one reaches here but since this is a reverse Bios you know the diode is one is here and one is here it is reverse bios, you need to have at lease points on volte difference being identical gate that cannot happen and these will not contact this will not create short circuit that side of the diode.

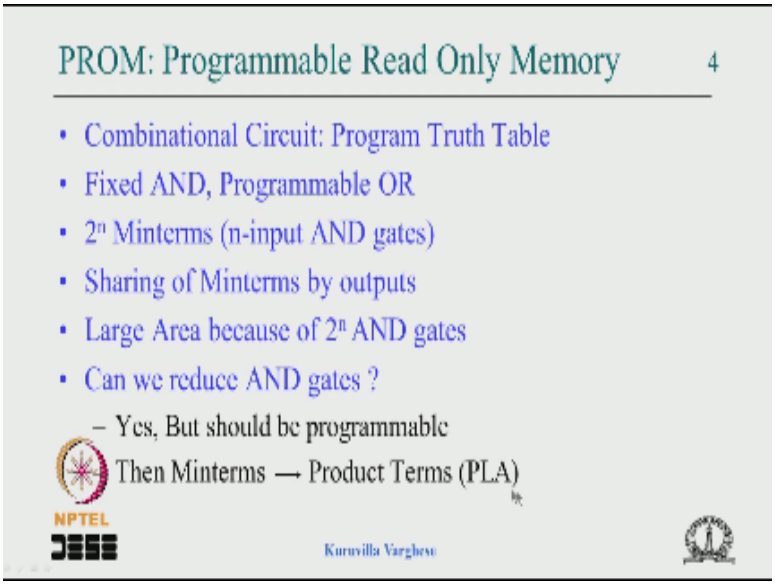
So as I said the programmable read only memory is a kind of fixed and a programmable OR ok. Now how to blow the fuse it is enough if you know like the current rating is a (()) (19:44) you draw larger current may be the 30 mm or 40 mm here then this fuse is gone, it is a onetime business what is done is that at this point ok. Suppose you want blow this fuse and naturally this has to shows the current.

There will be some mechanism to source a large current wherever you are connecting that output is there that will be large current sourcing, say you want to blow the fuse what is done is that you give 00 here now what you do is at this point and apply a negative voltage maybe say this is grounded. So you give kind of -5 volt and the supply 5 volt earlier the potential difference was you know somewhere around 5 volt little less than 5 volt.

Now it will be near around 9 volt because it is double with some drops you know removed or near to the 10 volt, so the current flowing you know through the fuse is double that of the normal current and the blow fuse and it just for a short duration have negative power supplied and if you care to see the old time devices there is a pin code vpp that is called the programming voltage V is the voltage P is a programming.

So there you apply negative voltage and then you can appropriately choosing the minterms you can blow the fuse rest will be retain but as I said it is a kind of non-volatile once you program you cannot reprogram it then you get it ok. So that was a basic idea of the programmable read only memory.

(Refer Slide Time: 21:36)



You can implement a combination cute essentially you program the truth table and we have seen that it is a fixed and it is a programmable OR ok. The AND is fixed by the storage to an AND gate in the case of any input by blowing or retain the fuses are we are programming 0 or 1, so you can you are able to program the truth table and also like when in the case of multiple outputs multiple like in this case 2 output same minterm can be chosen ok.

You can say you can share a minterm between two outputs and it is possible that the first output may not use that minterms the other second output may use or both be used but in any case it possible that a single minterm can be shared between two output that possible ok. So that is the idea , but basically there is a problem as you go increase the number of input say assume that in a chip select decoding maybe you have to kind of use as a decode 10 address line ok.

Suppose there was a 16 bit microprocessor then only say A15-A9 need to be or A6 need to be decoded that means 10 address line need to be coded, then you will have 10 address lines and to raise to 10 then 24 locations I mean 10, 24 AND gates ok. Now it is really funny if you look at the particular application of chip select decoding you were napping the device to a particular location.

And many times it requires only one and gate okay like all the higher bits are 1, 1 was some pattern ok many times you can get 1 AND gate but you have 1024 AND gates, so these AND gates for many like you know that even if any 5 variable normal Boolean function you are not going to use 2 raise to 5 min terms of use all the minterms and output is one always. So this is an overkill this number of AND gate overkill.

So the question we are asking is that can we reduce the number of AND gates, ok so that was a natural question people ask, so it all started with designers using the PROM as a programmable logic, which was not kind of intended function of the PROM though it is obvious for us now that time it was a kind of invention or an creative use of the PROM. So people felt that actually lot of wastage of the AND gate in this kind of architecture.

So why not why do not we use less number of AND gate, the moment you useless number of AND gates then you should realise that those AND gates we cannot fix the min terms okay, suppose we say that let us remove these two AND gates then what happens is that you have only minterms A1 bar A0 bar A1 bar A0, then you would not be able to program they are for function ok.

So the moment you try to reduce the number of AND gates then you should make sure that this AND gates are programmable that means this min term should be program, that means

you should be instead of 2 fix connection with AND gate with 2 input you should have AND gate with 4 input and with maybe fuse is connected at the input, some kind of programmable mechanism.

We will see what is that mechanism, how it is really implemented, this is just a kind of logic logical schematic not the actual implementation and we will see how this is implemented using transistor, we will see that, but essentially required AND gates with 4 inputs, 4 programmable input then we can reduce we can I have only using some statistics okay like to study the Boolean function of different variables 5 variable, 6 variable from practical cases.

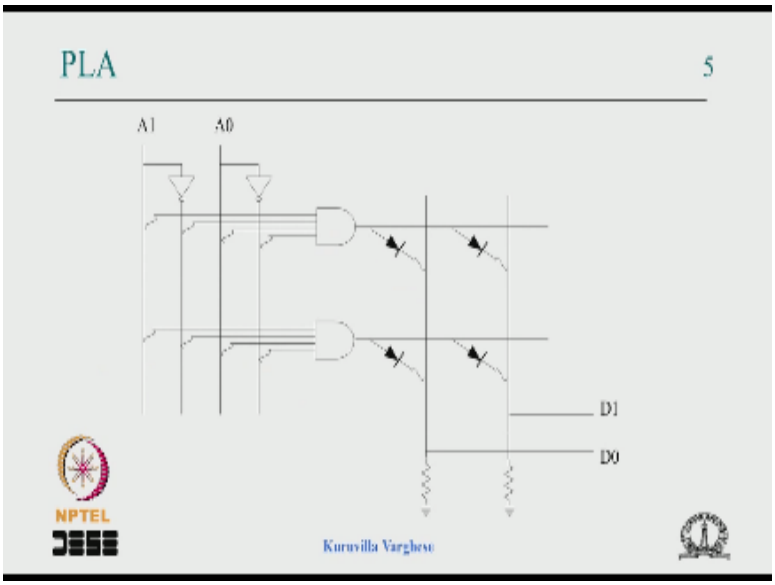
And say in 90% of 80% of the time for a 5 input scenario you do not require more than same b14 min terms or something like that, then you can usually 14 and it is ok. Now the moment user programmable gate it is even possible there is another advantage that you do not need to kind of stick on with the min terms now you can stick on with the product also, so earlier suppose we had used A1 bar A0 bar and A1 bar A0 ok.

So both min terms where there, but we know that A1 bar A0 or A1 bar, so A1 bar A0 bar or A1 bar A0 is nothing but A1 bar only. So it possible that in such a scenario you can minimise your Boolean equation ok and the product now if you have two min terms in the output then what you do is that you only implement A1 bar here that means you are there are 4 connections you retain the A1 bar connection.

And though all the other connections, so the moment you have lesser we can use lesson number of AND gate we would definitely make these AND gate programmable and at that point we do not need anymore we are not working with the min terms we will work to the product term which will even reduce the requirement of number of AND gates ok. So that was the next step against people started building this device ok from idea taken for this PROM that this is too much of an overkill.

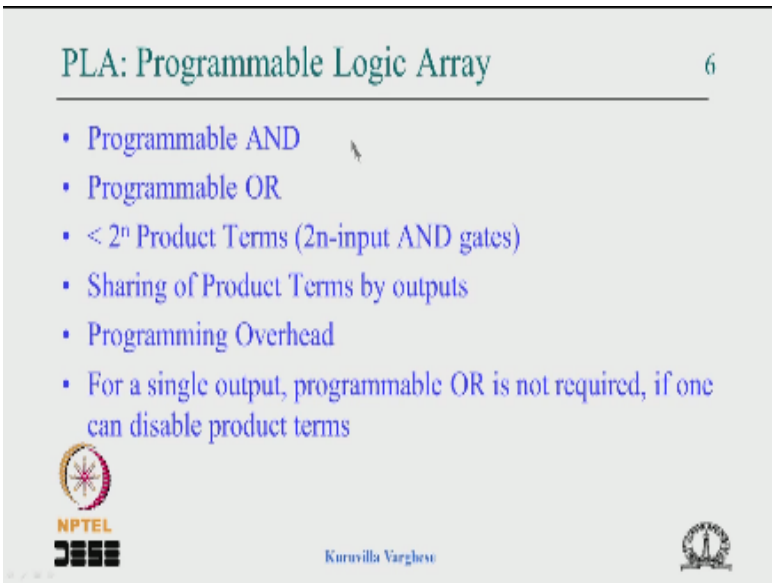
So most application does not require this number of AND gate, but chip select decoding very few 1 And gate or 1 AND gates and even in normal Boolean function the number AND gates required less, so people reduce a number AND gate but made AND gate programmable and the moment it is programmable the requirement of minterms has vanished and we have come to the product sense to you kind of minimise equation and start implementing it that.

(Refer Slide Time: 29:16)



So that what we said we can reduce and get but the AND gates should be programmable the minterms become the product comes and such a device was PLA okay. The Programmable logic array it was called, it is called programmable logic array ok that I am not written that in the previous slide.

(Refer Slide Time: 29:22)



The Programmable Logic array now somewhere in the lecture I have told I do not worry too much about these terminology and do not ask me why it is called Programmable Logic array, I mean just a mean it may have some kind of you know justification by the those invented it but we need not worry too much about this terminologies because all the more we go to the next step in be kind of confusing.

So this was architecture of Programmable Logic array again we are taking the case of 2 input case where there are two inputs and its complement. Now instead of 2 raise to AND gate we have only 2 AND gates and each AND gate has you know the 4 inputs connect to the two inputs and its complement with a Programmable P with a fuse now the with some kind of programmability like you know simplest case you can imagine the fuse.

And you want to program say A1 bar that will retain this fuse though all other fuse, or you want A1 bar A0, then you retain these 2 fuses and remove these fuse, and naturally you can do the programmable ok, now this become pretty much complicated because you have multiple output what program, how to program this is how to program that because now when you are multiple output is possible that a product and can be shared between the output.

Now we have talked little bit about the multi output minimization when we have done an overview of the beginning of the course and when you have a multi output minimization have to find basically the common sub-expression in the equation and try to implement that ok that is what need to be done that is common the largest common sub-expression should be implemented and that can be shared between these two.

But you see that this is a little bit of an overkill because there is lot of programmability ok, now like you have that like your Programmable AND and Programmable OR, so summarising the PLA was Programmable AND and Programmable OR, PROM was fix that because all min terms were there and Programmable OR when it comes to Programmable Logic array you have the program of AND and Programmable Or.

We have less than 2 raise to N product terms and you can share the product terms by output and the command is programming overhead because you have to kind you know program this fuse at the other input of the AND gate and the output of the AND gate and so on, lot of programming overhead was there but much more than that, suppose you assume a single output you do not worry about multiple output scenario.

You take just take a single output you are able to program whatever the product term you require, the moment you do that there is a less requirement, les need for a Programmable OR or kind of thing ok. Because suppose you are trying to program X xor Y ok or A xor B like A

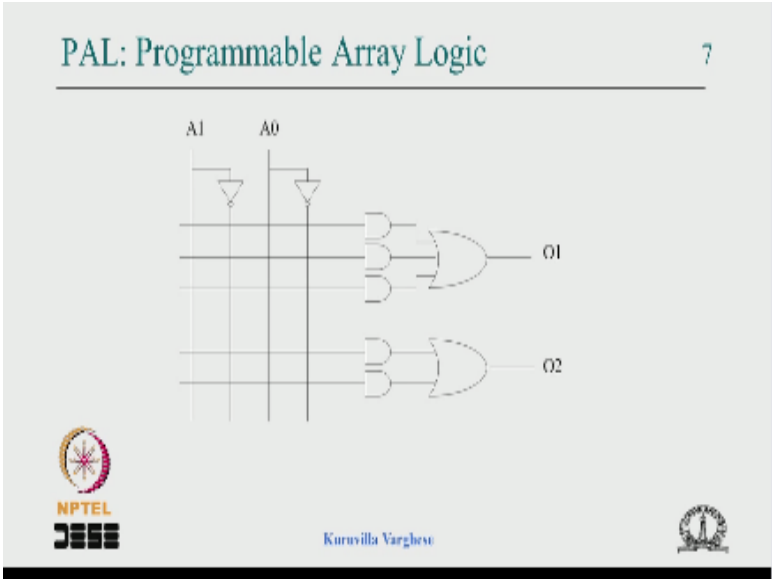
is connected here, B is connected here. Then we will program A bar B here, AB bar here, but no need to kind of program this ok.

So assume there are 4 AND gates ok and if somebody you can disable these 2 AND gates then there is no need to kind of you know how make the output of AND gate 0, then there is no need for this Programmable or section we can fix you know make the connection fix and that will make all programming overhead reduce. So that is that was the next evolution for a single output you do not need a Programmable Or.

Because if you have multiple AND gates support in this case we are put AND gates and you need only 2 product term, somehow make the and get output zero we will see that how to do that then there is no need to have a Programmable or section and output that was a basic idea and such a device is called Programmable Array Logic ok. Now this is Programmable Logic array and this is Programmable Array Logic and as I said do not worry too much about the name ok.

What is called PLA and another is called PAL, I know it is kind of programmable logic or since earlier was one was called PLA, this is called PAL.

(Refer Slide Time: 34:46)



And now I will kind of shorthand for a representing the AND gate, so let us come to PLA, now onwards it is very difficult like when we are going to see some practical devices which has lot of input, now in a picture if there is a 10 input, then you will have 20 vertical lines and is very difficult to show AND gate with all the 20 connection. So now what we are going to

next slide onwards when we have a 4 input AND gate will only show a single line going like that.

Assume that there is programmability P at the cross ok, so we are going to show so like this you have an AND gate a single line assume that there is Programmable P I mean assume that is a 4 input AND gate ok that best way to do that we will see how it is implemented as I said. So we have a shorthand representation of a 4 input AND gate with programmability at all these inputs ok.

So this is the kind of architecture of programmable array logic where you have some fixed number of programmable and in this case there are 3 AND gates which is connected to an OR gate ok. So can you take this out what you can program you can implement a Boolean function of two variables up to 3 products ok, so up to like your only one product that we programmed in the first AND gate.

And how make sure that this AND gate and this AND gate are disable ok. We will see that how to do that very simple and elegant way. So this is the architecture of programmable array logic where the AND gates are Programmable but the number OF or GATE fixed that means we have 3 AND gates which is connected to an OR gate permanently and get an output.

And maybe like in a real device that could be variation like it not that all the outputs are kind of the consists of three product times maybe there are some output which consist of only 2 products and things like that ok. So that is how and this is the basic architecture of a PLD or Programmable Logic devices ok. The PAL is a real architecture of a Programmable Logic devices.

And we were talking about their evolution and the real kind of logical reasoning behind how this is evolved ok. So though we are not using any of those very much definitely not using the PROM and the PLA and the PALs are used but then it is a worthwhile academic exercise how this is evolved and why such an evolution happened ok to understand that will help in some other logic design I hope.

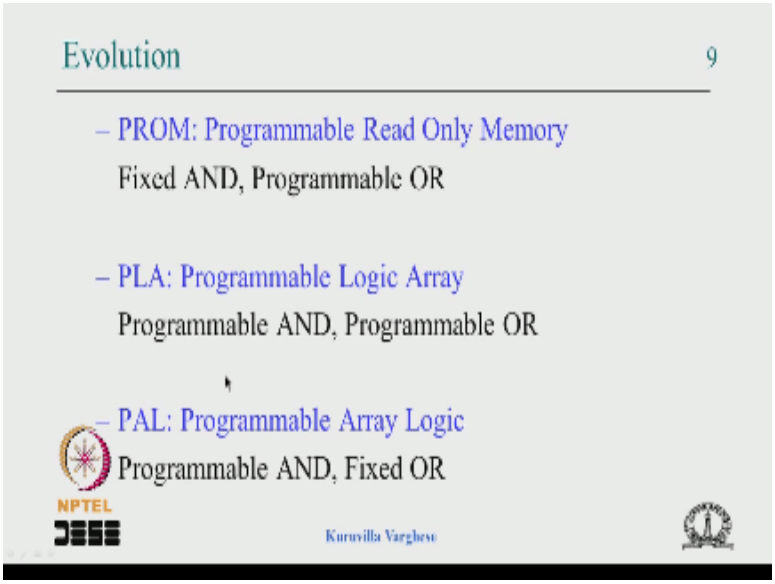
So the PAL is Programmable and fix not to be started with a Programmable fix that and Programmable or we have come to both Programmable then we have kind of inverted it, it

was from a fixed and Programmable or we have ended up with a Programmable and fix the for all the reason we have stated here for an input tools to less than 2 raise to N product on each AND gate of the product term has 2 N input to be able to program any min term or product term.

Each AND gate for the product term has 2n input to be able to program any min term or product term you have dedicated product time for output then for a particular output the product terms are fixed that it is a architecture of the PAL. So this is the evolution we have started with the PROM which is Programmable read only memory which had basically a real memory where the address Decoder access a min terms of the Boolean function fix time.

So N input have to fix AND gate, each AND gate has fixed connection to the N line, then a Programmable or basically it works as programming the location with truth table and in a PROM that is implemented that programmability is implemented through the fuses which we blow by you know flowing a large current and diodes are kind of in series, not kind of a cross drive from the VCC to the ground.

(Refer Slide Time: 39:50)



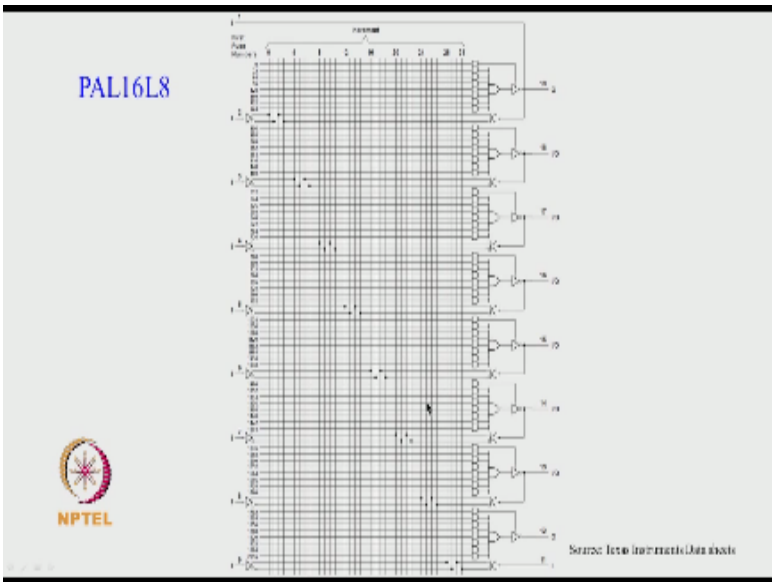
The next one was at this AND was an overkill because in a chip select decoding very few AND gates are usually 1 AND gate is used even in real Boolean functions number of products or min terms are less. So we reduce the number of AND gate but then it necessary rated to make it Programmable that means increasing the number of inputs to 2n and which is program and that enable not only them min terms of the product which was much more kind of use, if it allowed to reduce the number of the AND gates.

Because the moment you talk about the product you were talking about minimise implementation of Boolean equation is minimised get the product of that is implemented and Programmable or lot of it because of program blocked. Again for a single output we said that this programmable OR does not make sense because we program whatever the product and where can disable AND gates that automatically implement a programmable logic.

Suppose you are fixed or and by playing with the AND gates disabling the AND gates we could get that programmability in the kind of in the present a language we can say we can get a virtual Programmable or something like that. So that prompted this particular architecture the PAL which has a program AND section and fixed OR section ok. Now so that what we have is about the idea of this PAL has come from using memory as a Programmable Logic .

Because of non-volatile the PROM was used and which is a fixed and Programmable or number of AND gates use was less, so we may reduce the number of AND gates making it Programmable with unable to program the product terms and that was PLA and for single output again the moment of AND gates are Programmable programmer OR was not very much necessary. So that was prompted the architecture of the PAL which is nothing but Programmable AND and fix OR, so let us turn now to do some real device.

(Refer Slide Time: 42:20)



The first device okay gain the bit of a history because nobody use nobody makes this PAL 16 L8 anymore but if I remember if I can collect this was one of the devices which came into the

existence of people design 16 L8 and the then the MD used to make this particular device excess instrument. This particular diagram is from the excess instruments old data sheet ok.

So that shows the internal of the device and it do not be kind of alarm by the number of lines its very simple. So I hope you can kind of it is visible. So take this input you know that this was a 20 pin you are in line package bit package which in todays standard was very huge but hardly implemented very low density logic function ok. So take this is input, this is 1 input to see this one is coming and going through a directly through a buffer and threw an inverter.

That shows a buffer and inverter together and two lines. Sn instead of showing it clear to save space it is shown on the side. So this line number 3 like 2 and 3 are the compliment and the particular line of 1, 1 and 1 bar is these 2. So you have another input 2, so this that line 2 and the input is here. So these lines are nothing but the inputs or its complement and so you have a dedicated input 1,2,3,4,5,6,7,8,9 and this particular one.

So there are 10 dedicated input, so you can up to 20s say 0 to 19 up to here is a 10 inputs and its complement ok, maybe that some connection may be kind of the you can see these two connections are here and I am just telling the number of lines 20 lines of 10 dedicated input and its complement and this AND gate now you take one of them AND gate here this connection to all the input with which is Programmable with some kind of programmability at this junction ok.

So if you see there are 0-31 so there are 32 vertical lines that that kind of tells us that there are 16 input and its complement ok. Now we have located the 10 input, so what does that additional 6 input come. So look at this maybe look at this structure, so you come to 6 additional input, so here you see there are 1,2,3,4,5,6,7 AND gates permanently connected to an OR gate which is going through a tri-state invertor.

And it is available as an output ok and the tri-state inverter enable is controlled by another AND gate, we can call as a control AND gate, okay that means if this AND gate output is high this is output is available if it is low, it is tri-state ok. But so this is a dedicated output which can be enabled or cut off. Similarly come that is pin 19 and pin tall is also a dedicated output, so there are if you see 1,2,3,4,5,6,7,8 output. So that is what is shown here 8 output.

But on the top one and bottom one are kind of dedicated output and look at this section 1,2,3, ,4,5,6 you see that it is when you enable this tri-state gate it act as an output but if you cut it off then this act as an input ok. So along with 10 dedicated inputs we have 6 IOs ok, which can be can use as output or input, so this now allows this IO pin to be used as input also ok. So in addition to 10 dedicated inputs you have 6 IOs which can be used as input.

So there are maximum possible number of input is 10+6 so you have 16 inputs and its complement. So there are 32 vertical line, that is how the 32 lines come. So now you imagine this each AND gate as 32 input with programmability, if you can choose any of the 16 inpout or its complement for my product or min term whatever you call or so and in one section you can add up to 7 productive up to 7.

So we can use three of them or you can leave the 4 unused ok we will we will quickly see how this kind of to be able I mean how can you disable and gate, so we will see that we will in a moment I will tell you. So now you look at this output kind of this dedicated output there is hardly any reason for tri-state gate most of the time this need to be permanently enable unless you connect to some kind of a data bus where it is tri-state or you know shared bus or something like that.

So how to permanently enable it, so idea is that basically you cannot like you have studied the each then it is bit ridiculous to think of a kind of 16 input 32 input AND gates ok. So definitely this is a wired AND gate ok and I hope you have studied wired and wire more we will see that we will I will show you how it is really implemented, but do not worry this is not like a conventional kind of TTL circuit with 32 input or not even a conventional CMOS AND Gate with 32 input it is wired and wired or we will see that ok.

So now assume that the wire gate with a single line with the resistance pulled up to the VDD or VCC which is what you do is that to permanently make the AND gate 1, you blow all the connection you you do not you know remove all the connections then this input is pulled up and since the input all the input 1, the output will be 1, it is permanently enable ok. So that is how the AND gate is permanently enable ok.

But there is a need suppose we have 7 product terms here in a particular for a particular Boolean function we need only kind of 6 product terms and we need kind of disable this

AND gate, ok how to do that is very simple ok, you know that A and A bar 0 the same principle is applied here, what you do is that do not do any programming on this input, retain all the connection ok all 32 connections are retain.

Now assume there has to be 4 some function at least one input has to be there, so even if there is one input, say A is connected here, B is connected here. So for a particular this AND gate A and A bar is connected to input. So irrespective of the input is 0, 1 this AND gate output will be 0 always and that is disable ok. So you want usually 3 AND gate no problem, rest of the 4 AND gates you retain all the connection do not do any programming.

Then those output are disable automatically we are doing the Programming of OR ok. By programming the AND gate 0 output 0, we are implementing the Programmable or functionality, so that the basic idea of how to manage the number of products and less ok that has to be managed. So now what we do is that to understand a little more better okay essentially it was you know it was a kind of general purpose Programmable device where you connect input, it is very simple matter.

And that was first time you know something of this was happening like you connect the input lines anywhere, so depending on your PC design you could say you are not convenient to bring A here, maybe it is convenient to bring A here, does not matter you connected that cause you could program it appropriately. So you could connect the various inputs here, various outputs, and depending on your DD program the number of AND gates and the product term, that was basic idea, it also say combinational circuit.

And we will see how this used as a the devices which was made for sequences circuit data path and though we cannot kind of seriously use these kind of simple PLDs for a great you know the datapath and things like that, that was not possible but still you know compared to earlier at that time the discrete Gates this was a major league major programmability to that extend the study in this is a very good idea.

So now let us come back to the slide, let us take this kind of say a part of it maybe we will take this first resection will they magnified to understand it better ok. So that is what is shown the inputs like 1, 2, 3, 4 inputs ok and 3 output section ok and now this is a dedicated output

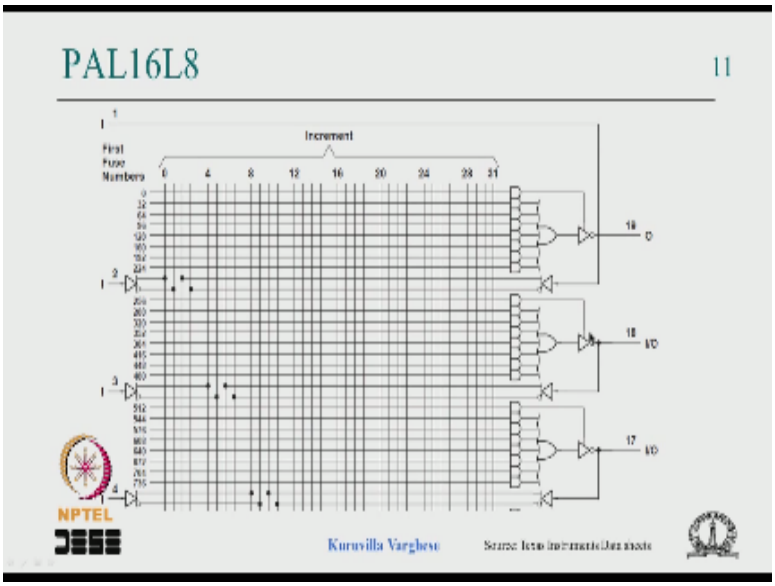
ok and which we can use it as early as an output definitely this can be kind of cut off, if it was a connected to a shared bus, it is possible to cut off by this control AND gate.

That can you can permanently cut it off or we can a product can control it ok. So that was a possible here, but take this section which is much more interesting as I said say this is an output and mind you there is a this is at tri state invertor ok and this was an active low kind of circuit and that is how this L comes in the picture say PAL 16 L8 ok, you might want suppose you have a Boolean function why is you know some of some product .

Then you implement if you directly implement that product term here, then what happens when you will get the Y bar instead of Y, so that was an issue but very easy suppose you have Y to be implemented then you apply De Morgan theorem you say y bar then you apply the De Morgan theorem convert that into to equal y bar term product terms and you implement it.

Then you will you are implementing y bar at this AND/OR gate, so which is invited by this whatever then you can I know that is how this was used ok. So this invertor itself is not a problem because the complement of that was taken that was implemented by the tools and maybe we are the structure though it is like this kind of output combine with the input though it is a very you know it is a very kind of very simple looking structure.

(Refer Slide Time: 56:00)



But it has many use you know it is really kind of very elegant design though it is very simple, it is lot of lot of advantages this particular connection this one connection we will soon see

maybe we are coming to the end of this lecture. So I will kind of wind up here because we do not have time do the kind of go through that. So in the next lecture we will go through it.

But what we have seen is a kind of commercial device then use now it is not available which call PAL 16L8 with 16 kind of 10 dedicated input, 6 IOs, so the maximum number of inputs was 16, so there are 32 lines vertical lines or 32 input, and its complement and there are 2 dedicated outputs, each OR gate has 7 Programmable AND gates, each AND gate has you know 32 inputs which is which can be connected to 16 input or compliments.

And then there is a tri state gate of the output of the OR gate tri-state inverter, they enable is control by another gate and we also have seen how to permanently enable that and there was 7 product I am going to AND gate. We have seen suppose you need only 3 AND gate suppose 3 AND gate how to disable, so 4 product and by retain all the connection which bring in the Programmable or facility in the program AND itself.

And we will we will see you more in detail that IO section of 16L8 which is very elegant and then we will want to some kind of devices with flip flop it allows you to implement data path and sequential circuits then pretty much we can move to the at least one device which is probably available even now which can be used in simple PLDs then we will move on to the complex PLD. So as I said though it is not useful directly. This will enable us to understand the CPLDs which have some used today we will be able to understand architecture of it very nice if you understand this. So please revise what you have done today and I wish you all the best.