Digital VLSI System Design

Prof. Dr. S. Ramachandran

Department of Electrical Engineering

Indian Institute of Technology, Madras

Lecture - 31

Synplify Tool – Schematic Circuit Diagram View

Previous Lecture

Synthesis Tool (Continued…)

Synthesis Tool
- Command Summary

Analysis of Design Example using Synplify Tool

Contents of Lecture 31

(Refer Slide Time: 01:52)

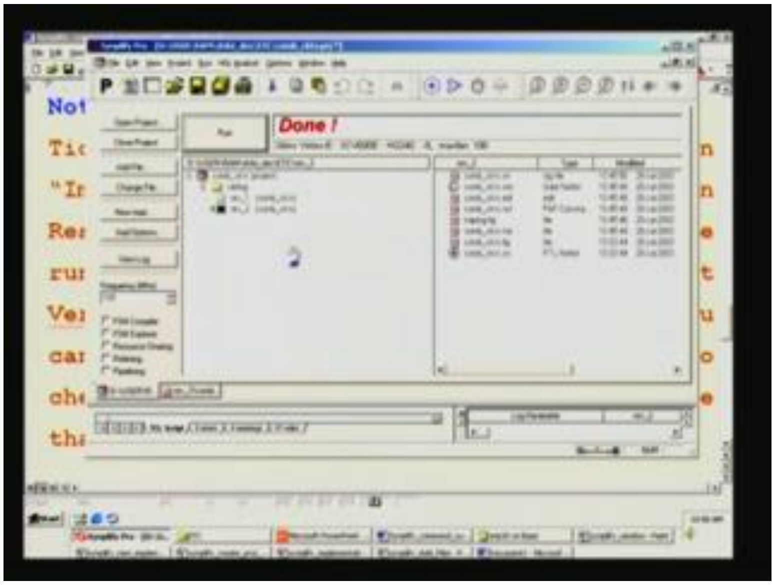
Analysis of the Report File generated by Synplify Tool

Commands Continued
- Optimized Verilog File using Synplify Tool

Viewing Verilog Code as RTL Schematic Circuit Diagrams

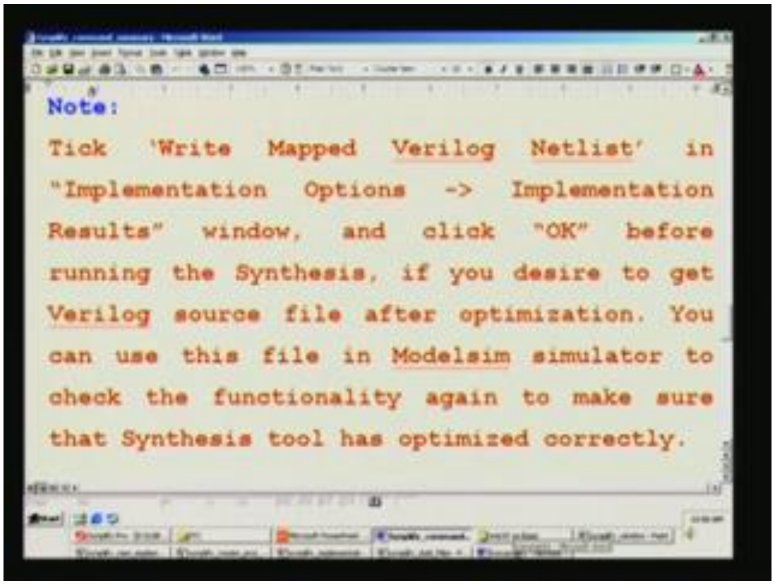
We were looking at the report file for synthesis using simplified tool.

(Refer Slide Time: 02:02)



We will continue from where we left.

(Refer Slide Time: 02:11)

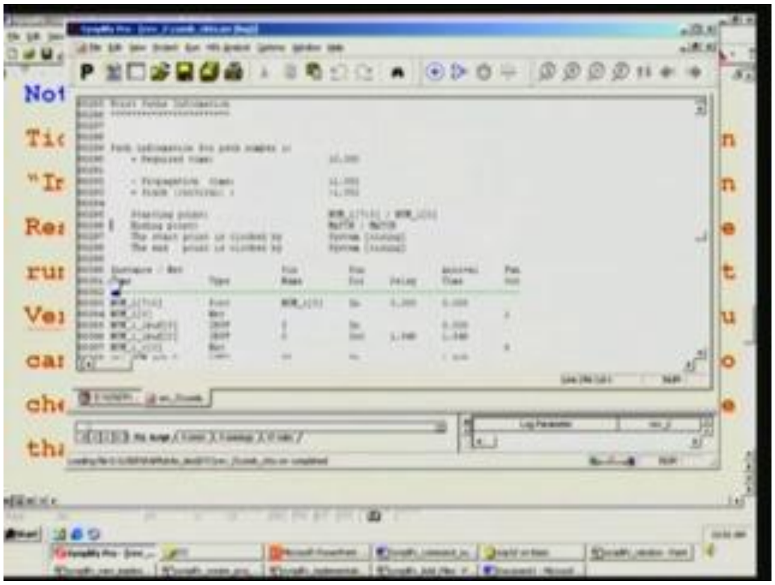


Note:

Tick 'Write Mapped Verilog Netlist' in "Implementation Options -> Implementation Results" window, and click "OK" before running the Synthesis, if you desire to get Verilog source file after optimization. You can use this file in Modelsim simulator to check the functionality again to make sure that Synthesis tool has optimized correctly.

We have seen so far how to add files, then run the synthesis tool and as a result of which we had got report file. We have been analyzing the report file. What is really required for our design - only which we are covering. There may be so many hundreds of reports made there and which you may not really require immediately. Gradually you can add up to your knowledge. What is absolutely required from my experience, only which I am trying to give you. If you have any question you can ask.

(Refer Slide Time: 02:55)

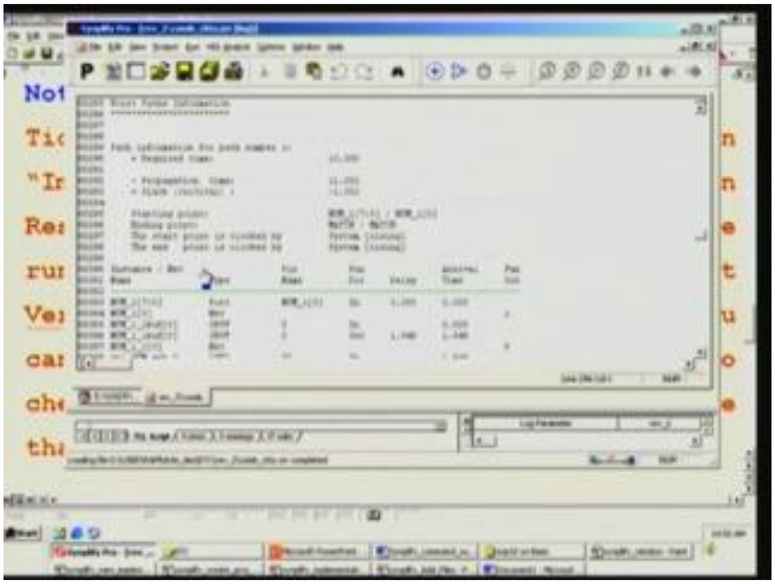


Suppose half way through the report file or half way through the design I find that my chip is not adequate. Can you go back and change it and do the whole thing once over? Suppose you have mapped on to a particular device and your design has outgrown that device. What do you do for that? That is your question, right? Let us say you have chosen a gate of 6, 00,000 capacity and you find that it is not adequate. You can remap it. You have also done the hardware. Everything as I mentioned earlier - so what do we do? Are we crippled by this? No.

The reason is that we have different devices with the same package with a higher capacity. You can at any point of time migrate to a higher capacity chip and the only unfortunate thing, will be is if you had already tested the highest possible available, in that series you will have no room and you have you are helpless there.

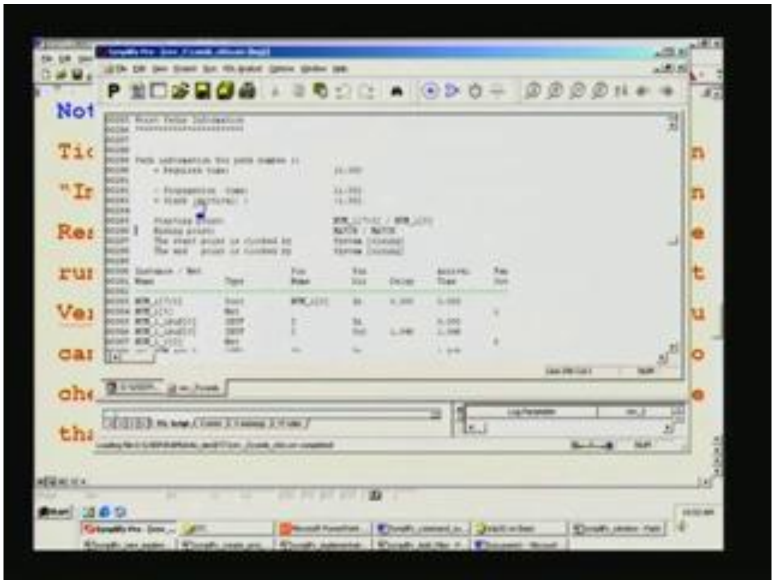
The only way you can be saved is by a new device coming in, in future times. Even then it will be done; the only thing is that your design will get stalled. That is the only situation you will have to face. Is your question answered? Do you have any other? Just for a rough estimate, could you tell me how much image processing implementation? How many gates will it take? How many gates with let us say mpeg implementation take? In my experience I had done what you have mentioned for example.

(Refer Slide Time: 04:37)



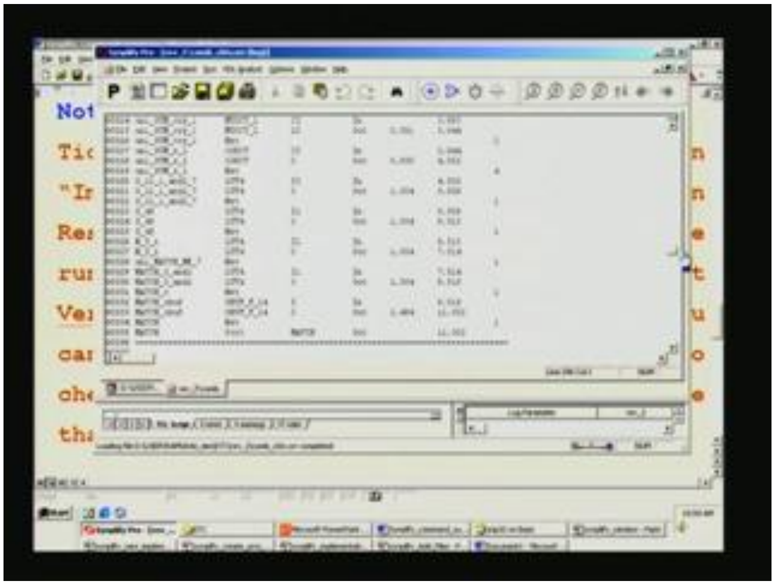
We will be covering in one of this. Say, D C T quantization we will be covering. That is required followed by what is called VLC. This module put together you can make a jpeg as such. If it is for mpeg in addition to this we need its inverse, for example, inverse quantization, inverse D C T, in addition to that, you need also what is called motion estimation. This is to process motion picture so you need motion estimation for that. I have done research in these areas so I am able to give figures.

(Refer Slide Time: 05:20)



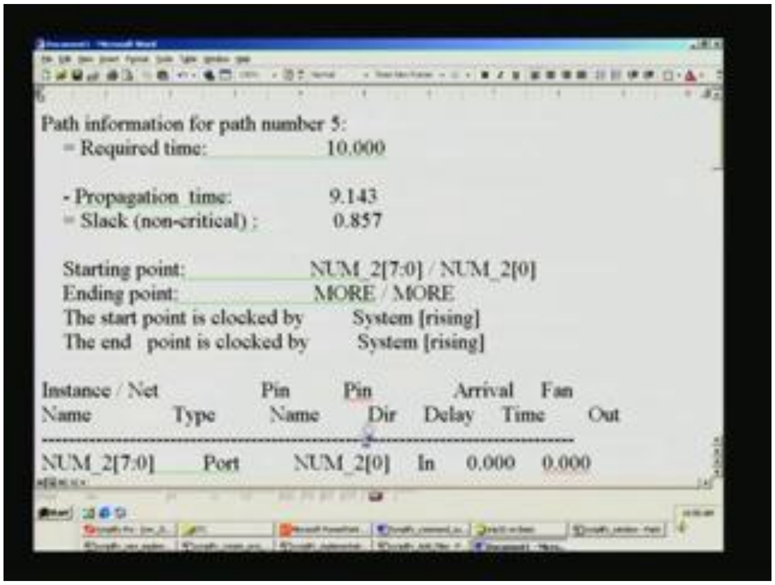
From my experience I think it is around 100000 gates that I have got for D C T quantization and maybe for VLC 40000 gates or so. For motion estimation maybe another 50000 gates. You can put it up, add it together and see where you stand as far as jpeg is concerned or mpeg is concerned. We will cover more details later on, when we really look into the application so is your question answered.

(Refer Slide Time: 05:49)



We will continue with the report that we had been going through. We saw that for different signals we have listing of the progressive delays, that you have gone through the path of, let us say, here is one more. We will take another typical thing signal. Similar thing we have to take. I will just take the very last thing that is reported here.

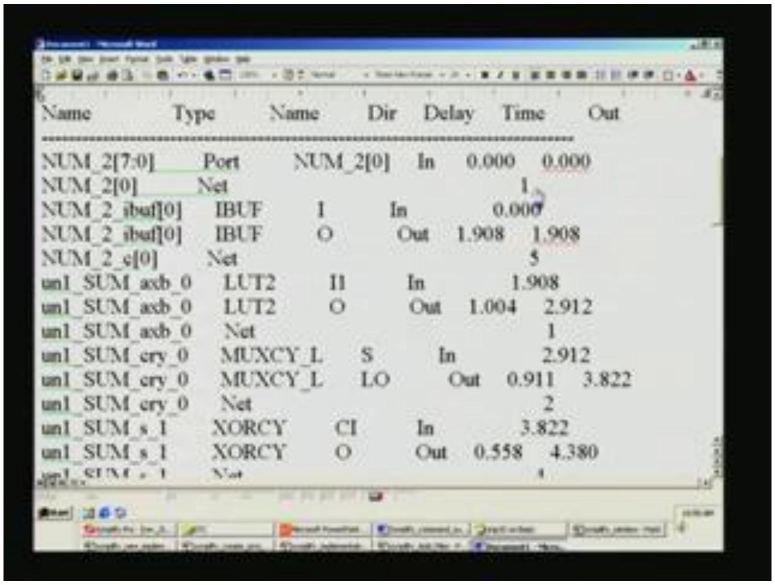
(Refer Slide Time: 06:49)



```
Path information for path number 5:
  = Required time:              10.000

  - Propagation  time:           9.143
  = Slack (non-critical) :       0.857

  Starting point:               NUM_2[7:0] / NUM_2[0]
  Ending point:                 MORE / MORE
  The start point is clocked by      System [rising]
  The end   point is clocked by      System [rising]

Instance / Net            Pin        Pin          Arrival   Fan
Name           Type       Name       Dir   Delay   Time        Out
---------------------------------------------------------------------
NUM_2[7:0]     Port       NUM_2[0]   In    0.000   0.000
```

This signal is actually number 2. The same example that we have seen earlier and we have that arrival time and delay the start point is number 2. Here also it happens to be 0th bit here. It reports only the critical phases of different signals. Or the worst case delays that have been encountered, and here it says non-critical because the slack time is positive here. So if it is negative only it becomes critical because it is unable to meet your constraints.

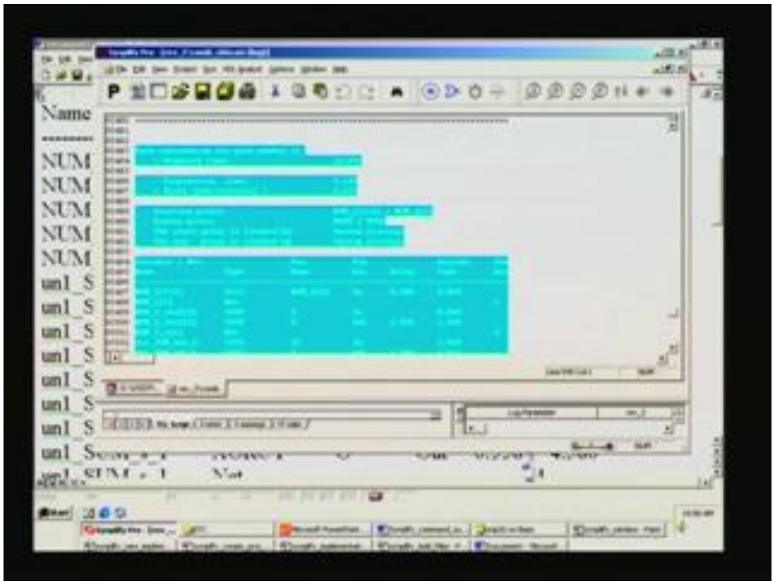
(Refer Slide Time: 07:32)



```
Name            Type        Name       Dir    Delay    Time       Out
-------------------------------------------------------------------------
NUM_2[7:0]        Port        NUM_2[0]   In     0.000    0.000
NUM_2[0]          Net                                    1
NUM_2_ibuf[0]     IBUF        I          In              0.000
NUM_2_ibuf[0]     IBUF        O          Out    1.908    1.908
NUM_2_c[0]        Net                                    5
un1_SUM_axb_0     LUT2        I1         In              1.908
un1_SUM_axb_0     LUT2        O          Out    1.004    2.912
un1_SUM_axb_0     Net                                    1
un1_SUM_cry_0     MUXCY_L     S          In              2.912
un1_SUM_cry_0     MUXCY_L     LO         Out    0.911    3.822
un1_SUM_cry_0     Net                                    2
un1_SUM_s_1       XORCY       CI         In              3.822
un1_SUM_s_1       XORCY       O          Out    0.558    4.380
un1_SUM_s_1       Net                                    4
```

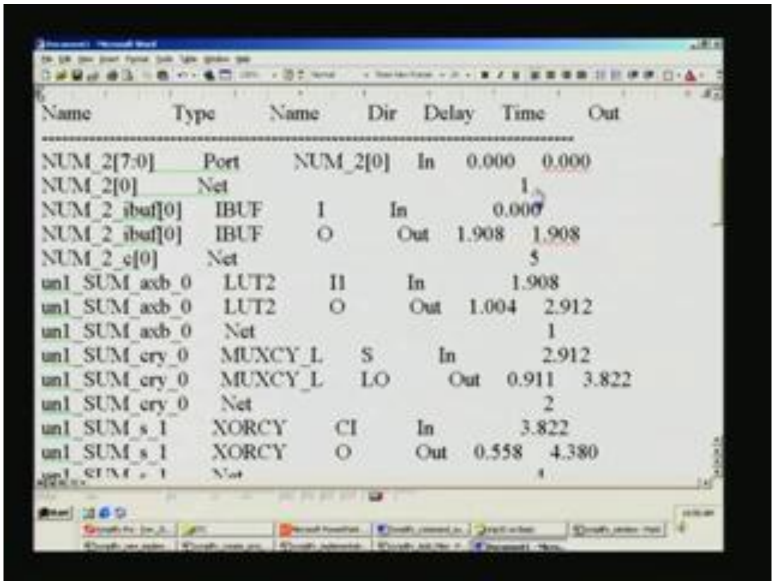
Once again you can see for net there is no delay. What is this one? Let us see what.

(Refer Slide Time: 07:50)



```
Name
--------
NUM
NUM
NUM
NUM
NUM
un1_S
un1_S
un1_S
un1_S
un1_S
un1_S
un1_S
un1_S
un1_SUM_s_1    Net
```

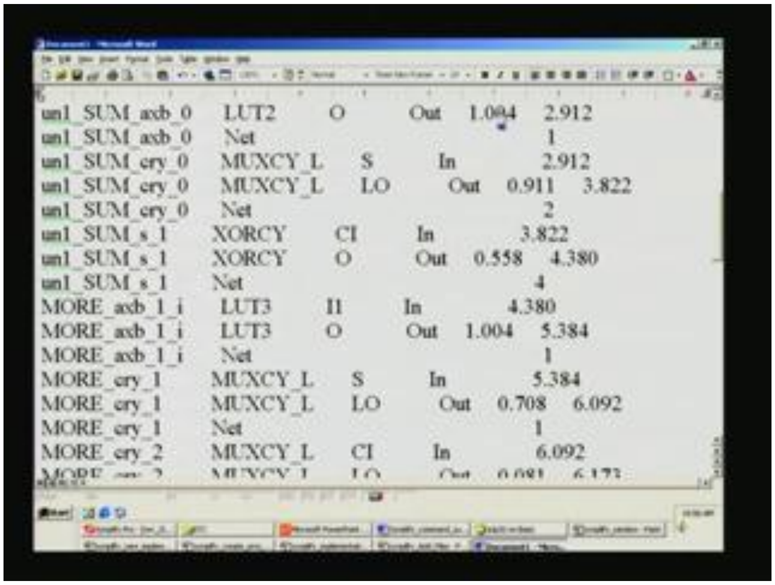
(Refer Slide Time: 07:53)



```
Name            Type        Name        Dir    Delay    Time     Out
========================================================================
NUM_2[7:0]       Port        NUM_2[0]    In     0.000    0.000
NUM_2[0]         Net                                    1
NUM_2_ibuf[0]    IBUF        I           In             0.000
NUM_2_ibuf[0]    IBUF        O           Out    1.908    1.908
NUM_2_c[0]       Net                                    5
un1_SUM_axb_0    LUT2        I1          In             1.908
un1_SUM_axb_0    LUT2        O           Out    1.004    2.912
un1_SUM_axb_0    Net                                    1
un1_SUM_cry_0    MUXCY_L     S           In             2.912
un1_SUM_cry_0    MUXCY_L     LO          Out    0.911    3.822
un1_SUM_cry_0    Net                                    2
un1_SUM_s_1      XORCY       CI          In             3.822
un1_SUM_s_1      XORCY       O           Out    0.558    4.380
un1_SUM_s_1      Net                                    4
```

It is only fan-out. Fan-out you know say number of additional inputs it can connect to a particular output of any particular cell or gate. You have here input buffers and then LUTs then MUX, 2 input MUX.

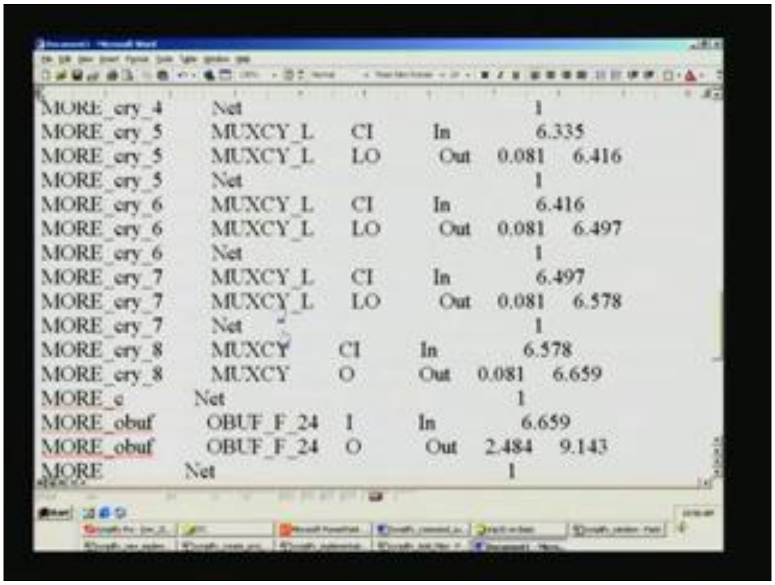
(Refer Slide Time: 08:16)



```
un1_SUM_axb_0    LUT2       O       Out   1.004   2.912
un1_SUM_axb_0    Net                              1
un1_SUM_cry_0    MUXCY_L    S       In            2.912
un1_SUM_cry_0    MUXCY_L    LO      Out   0.911   3.822
un1_SUM_cry_0    Net                              2
un1_SUM_s_1      XORCY      CI      In            3.822
un1_SUM_s_1      XORCY      O       Out   0.558   4.380
un1_SUM_s_1      Net                              4
MORE_axb_1_i     LUT3       I1      In            4.380
MORE_axb_1_i     LUT3       O       Out   1.004   5.384
MORE_axb_1_i     Net                              1
MORE_cry_1       MUXCY_L    S       In            5.384
MORE_cry_1       MUXCY_L    LO      Out   0.708   6.092
MORE_cry_1       Net                              1
MORE_cry_2       MUXCY_L    CI      In            6.092
MORE_cry_2       MUXCY_L    LO      Out   0.081   6.173
```

Then one exclusive OR gate is there and you can see that independent delays are mentioned here and corresponding cumulative delays rather the propagation delays are also listed here.

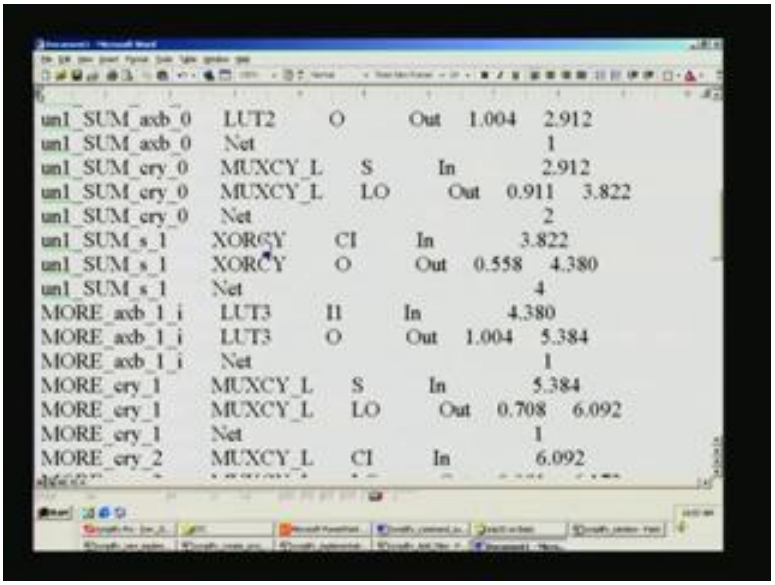
(Refer Slide Time: 08:31)



```
MORE_cry_4       Net                              1
MORE_cry_5       MUXCY_L    CI      In            6.335
MORE_cry_5       MUXCY_L    LO      Out   0.081   6.416
MORE_cry_5       Net                              1
MORE_cry_6       MUXCY_L    CI      In            6.416
MORE_cry_6       MUXCY_L    LO      Out   0.081   6.497
MORE_cry_6       Net                              1
MORE_cry_7       MUXCY_L    CI      In            6.497
MORE_cry_7       MUXCY_L    LO      Out   0.081   6.578
MORE_cry_7       Net                              1
MORE_cry_8       MUXCY      CI      In            6.578
MORE_cry_8       MUXCY      O       Out   0.081   6.659
MORE_c           Net                              1
MORE_obuf        OBUF_F_24  I       In            6.659
MORE_obuf        OBUF_F_24  O       Out   2.484   9.143
MORE             Net                              1
```

Finally, when you come to the end, you see so many MUX are being used. The characteristic of the FPGA is basically 2 inputs MUX and LUTs used in plenty. Then off course input buffer output buffers will be there. They are mainly for connecting to the actually external vault so to the pins of the device, and in addition to that you may have some exclusive OR gates like this.

(Refer Slide Time: 08:58)

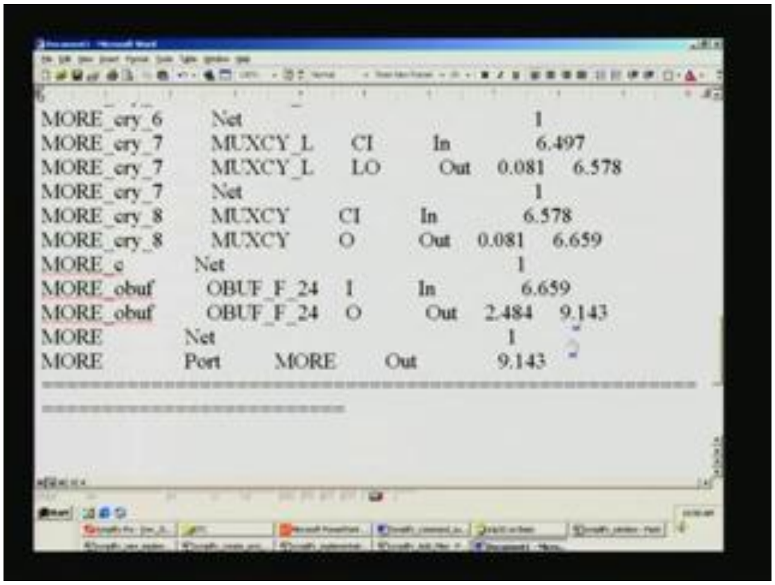


```
un1_SUM_axb_0    LUT2       O      Out   1.004   2.912
un1_SUM_axb_0    Net                             1
un1_SUM_cry_0    MUXCY_L    S      In            2.912
un1_SUM_cry_0    MUXCY_L    LO     Out   0.911   3.822
un1_SUM_cry_0    Net                             2
un1_SUM_s_1      XORCY      CI     In            3.822
un1_SUM_s_1      XORCY      O      Out   0.558   4.380
un1_SUM_s_1      Net                             4
MORE_axb_1_i     LUT3       I1     In            4.380
MORE_axb_1_i     LUT3       O      Out   1.004   5.384
MORE_axb_1_i     Net                             1
MORE_cry_1       MUXCY_L    S      In            5.384
MORE_cry_1       MUXCY_L    LO     Out   0.708   6.092
MORE_cry_1       Net                             1
MORE_cry_2       MUXCY_L    CI     In            6.092
```

A majority you see will be the MUX and LUTs, and LUT also they come as 4 input LUTs it is equivalent to having a rom, or simply a truth table of 4 inputs. That means 2 to power of 4 means 16 such words you can have. Did you mention before that we have control over the i/o pins? Yes. Where do you define that? You do not define it here, you do it only in place and route when we come to place and route. We have seen earlier frequency operation as one of the constraints.

Another constrain is the pin see for example if you do not mention which signal to be connected to which pin so it will do it so by default. If you do not like that assignment of the place and route you can always redesignate them and you have the choice. We will see how to do it when we come to place and route.

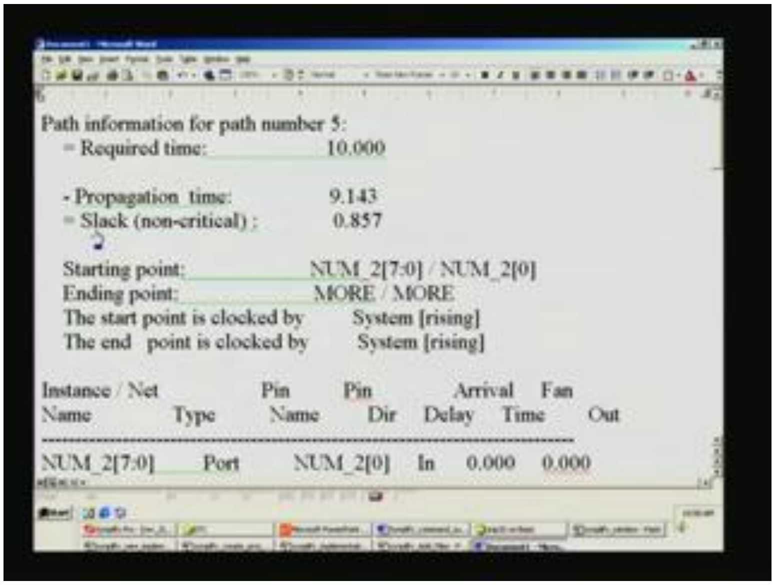
(Refer Slide Time: 10:04)



```
MORE_cry_6      Net                              1
MORE_cry_7      MUXCY_L    CI      In            6.497
MORE_cry_7      MUXCY_L    LO       Out    0.081    6.578
MORE_cry_7      Net                              1
MORE_cry_8      MUXCY      CI      In          6.578
MORE_cry_8      MUXCY      O       Out    0.081    6.659
MORE_c         Net                             1
MORE_obuf      OBUF_F_24   I       In          6.659
MORE_obuf      OBUF_F_24   O       Out    2.484    9.143
MORE          Net                            1
MORE          Port      MORE      Out          9.143
===========================================================
===========================
```

You see that cumulative delays are mentioned here. More is the output and what you have here is only 9.143 and 10 minus this. That is what you had seen earlier.

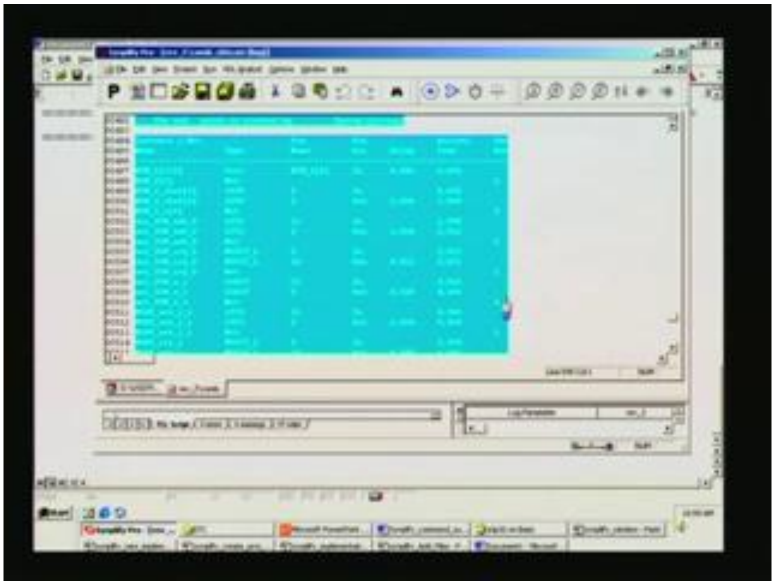
(Refer Slide Time: 10:20)



```
Path information for path number 5:
   = Required time:                10.000

   - Propagation  time:             9.143
   = Slack (non-critical) :          0.857

    Starting point:                 NUM_2[7:0] / NUM_2[0]
    Ending point:                   MORE / MORE
    The start point is clocked by       System [rising]
    The end   point is clocked by       System [rising]

Instance / Net              Pin        Pin            Arrival    Fan
Name            Type        Name       Dir    Delay    Time       Out
------------------------------------------------------------------
NUM_2[7:0]       Port        NUM_2[0]    In      0.000     0.000
```

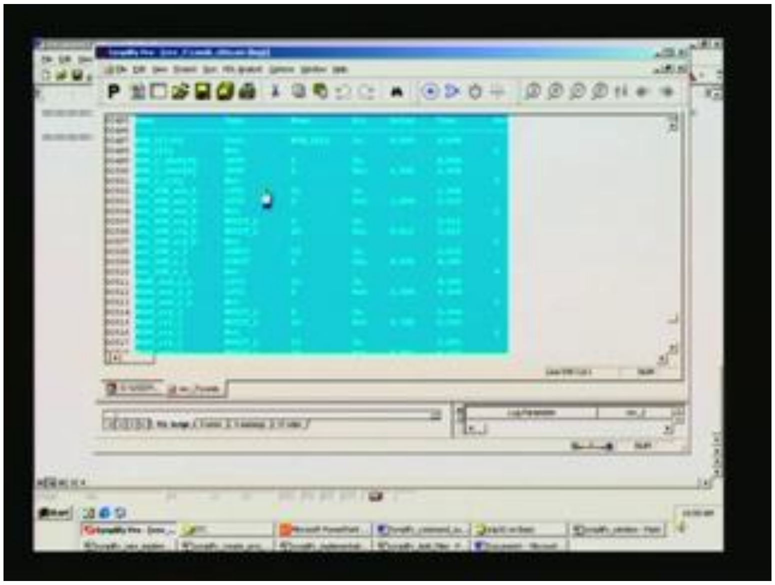
In the net shell, it says this total propagation is 9.143 and slack time is 0.857 because your asking was 10 nanoseconds. Fan-out I have already mentioned, it is a number of outputs, I mean you can take from a particular point. It lists 1 here then 5. It means, from here five branches take place. It has got to be connected to other internal devices. So naturally it calls for multiple connections rising from one signal.

(Refer Slide Time: 11:09)



What you see here is 1 5 1 2 4 1 and so on. It nearly lists the fan-outs there for independent primitive cells for example MUX it is only fan-out 1 and output buffer 1.

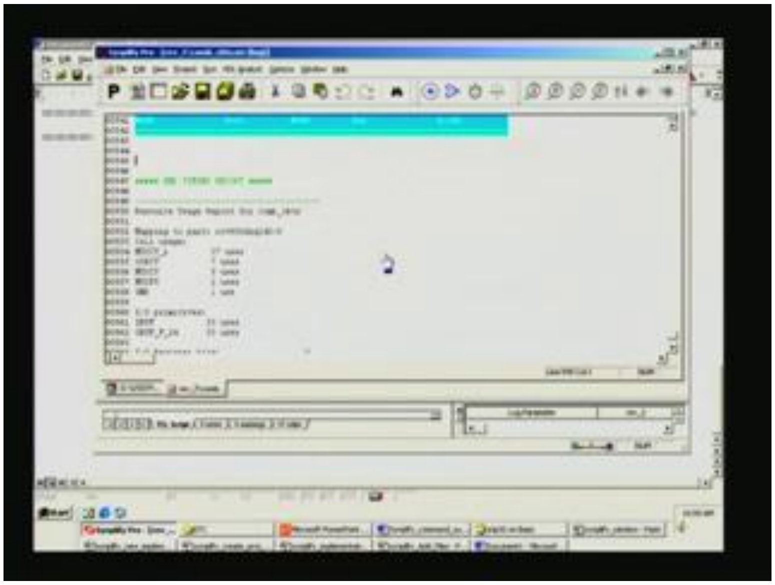
(Refer Slide Time: 11:24)



This is actually net from net it goes to 5 and all nets primarily are more than one.

(Refer Slide Time: 11:38)

(Refer Slide Time: 11:50)

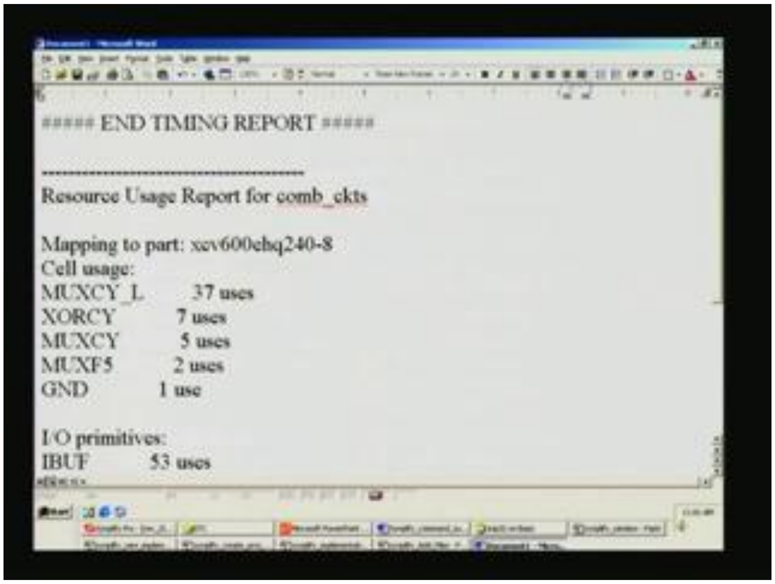


We do not have it normally, but if the fan-out also has a role to play too much it will again have problem later on. It is good practice that as far as your design is concerned you do not increase the fan-out appreciably so you may had to explore ways and means to overcome that.

How far we trust the synthesis tool? Do we hand out our complete control to it or can we do some optimization apart from what it does? You are the master. So no tool can be a master. So it only serves your ends. If it doesn’t serve your desire, naturally you withdraw it and do something else, right? As far as the intelligence is concerned, it has to emanate only from you.

It cannot do all on your own. After all, it is a man like you who has designed the tool. So whatever is this, intelligence only a subset of that will appear as a tool here? That is why I said you are the real master. Finally, before we wind up this log report, we will see how many gates it has taken all this will be of interest.

(Refer Slide Time: 13:27)

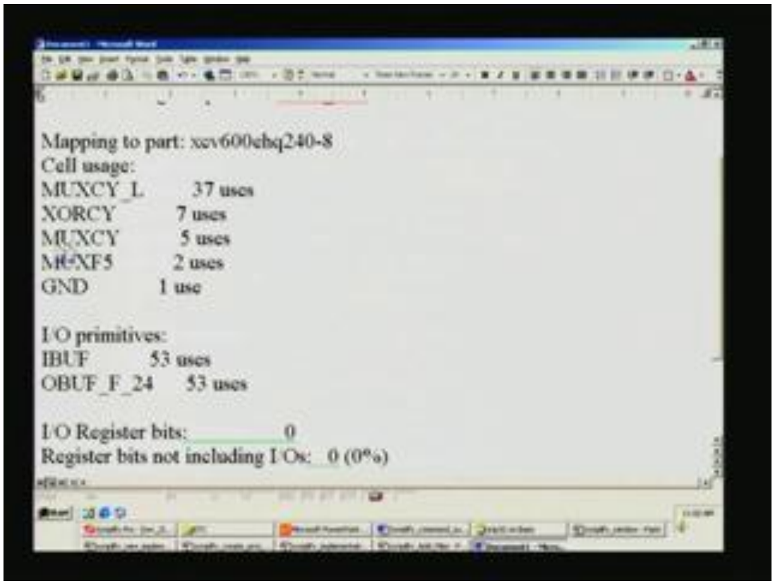


```
##### END TIMING REPORT #####

-----------------------------------
Resource Usage Report for comb_ckts

Mapping to part: xcv600ehq240-8
Cell usage:
MUXCY_L       37 uses
XORCY          7 uses
MUXCY          5 uses
MUXF5          2 uses
GND            1 use

I/O primitives:
IBUF          53 uses
```

This is final timing report and this is the module that we have had, and what device we have mapped, that is also listed here. So it is XCV 600 EHQ. This is one type of packet and e stands for this vertex e this is vertex e, and that is the latest thing available in Xilinx. There are higher capacities we have seen earlier and I think it is 3200. That means it is 3.2 million gates. In fact it is more than 3.5 or almost touching 4 million gates, in addition to that there will be a ram as well.

It is much beyond 4 million gates, you can say is the capacity in single chip and you have a different number of packages going right up to 1152 or some number of pins and right now what we have mapped is on 240 pin type and once again they speed here. The higher the number higher the speed you get so had we run this same thing with lower one we should have seen how much speed it would have given. We will consider by doing that.

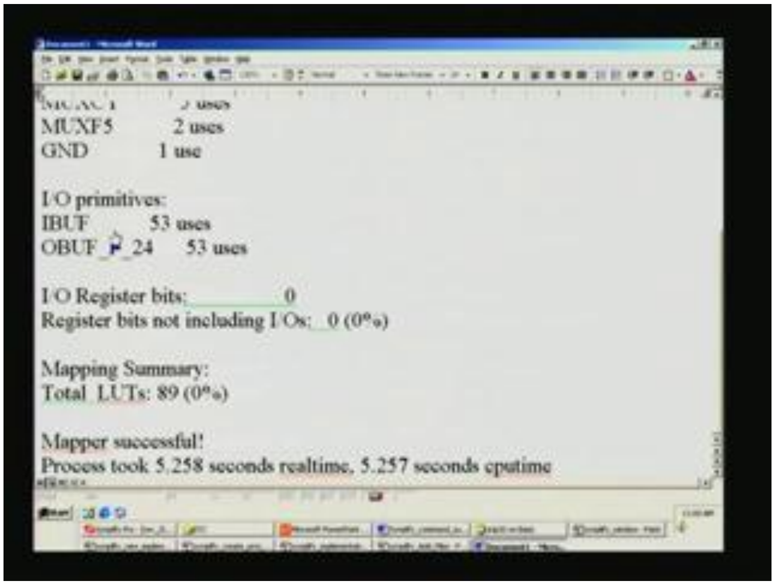
(Refer Slide Time: 14:39)



```
Mapping to part: xcv600ehq240-8
Cell usage:
MUXCY_L        37 uses
XORCY          7 uses
MUXCY          5 uses
MUXF5          2 uses
GND            1 use

I/O primitives:
IBUF           53 uses
OBUF_F_24      53 uses

I/O Register bits:              0
Register bits not including I/Os:   0 (0%)
```

Here, it reports not only the device but also what the primitive cells that have been used in the FPG. For example, we have already seen that it has used MUX, and how many times this MUX have been used. It is listed here. 37 times it has used here and exclusive OR gates 7 times and MUX. There are some slight changes, what they are we can have a look later on, and one more variety of MUX is available here. They are listed here 7 5 2 and of course ground is not a component as such.

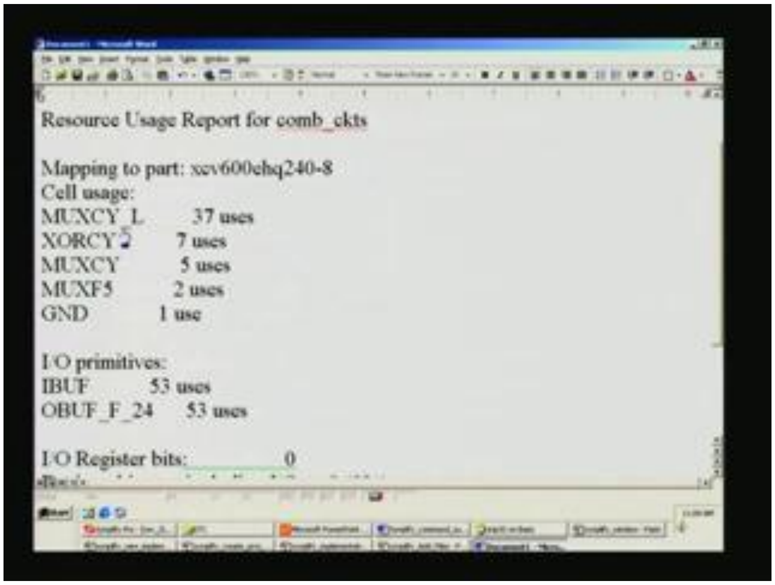
(Refer Slide Time: 15:19)



```
MUXCY          5 uses
MUXF5          2 uses
GND            1 use

I/O primitives:
IBUF           53 uses
OBUF_F_24      53 uses

I/O Register bits:              0
Register bits not including I/Os:   0 (0%)

Mapping Summary:
Total  LUTs: 89 (0%)

Mapper successful!
Process took 5.258 seconds realtime, 5.257 seconds cputime
```

Some in addition to this, you need input buffer that is for all input pins are not directly taken, but only through buffers for obvious reasons and especially from fan-out consideration internally, it has to be mapped onto several other internal devices. You need to buffer it, that is why and you need a buffer for that. Especially that is coming from outside world, you have always to buffer and perhaps even noise margin may improve, and if it is especially Schmitt trigger, it may improve. But I am not sure whether they have Schmitt trigger as input. Earth lesser buffer is to be used and similarly the output buffer when you connect to the external world you need output buffer and this is higher speed version, which we have already seen and each of these buffers are 53 in numbers for this particular design these combination circuits that we have put are all covered within this.

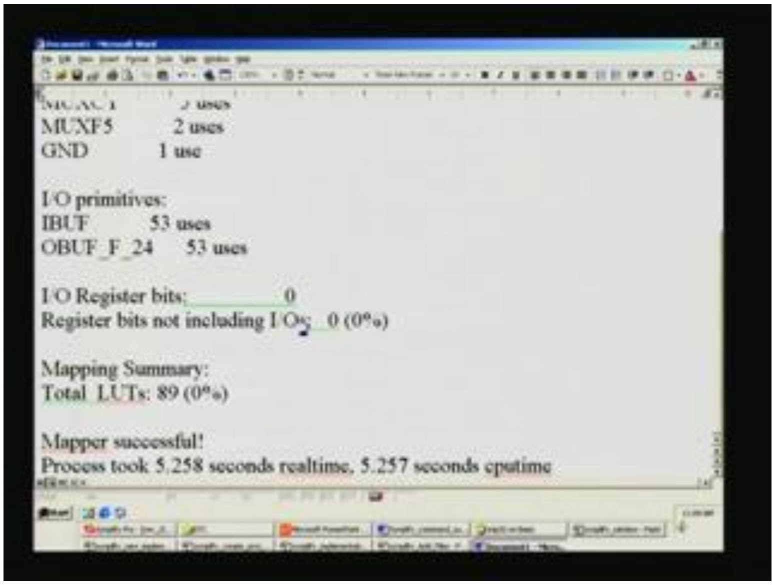
(Refer Slide Time: 16:23)



```
Resource Usage Report for comb_ckts

Mapping to part: xcv600ehq240-8
Cell usage:
MUXCY_L       37 uses
XORCY          7 uses
MUXCY          5 uses
MUXF5          2 uses
GND            1 use

I/O primitives:
IBUF          53 uses
OBUF_F_24     53 uses

I/O Register bits:            0
```

Are you clear about what gate count etc but it doesn't give in terms of gate count here. In space and count you will get a gate count also which is I think standardized to 2 input NAND gates.

(Refer Slide Time: 16:40)

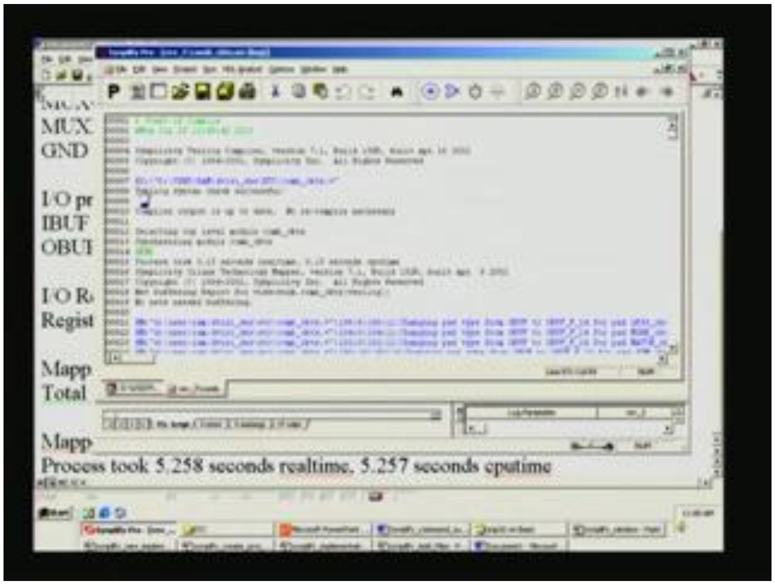


```
MUXCY         [illegible] uses
MUXF5         2 uses
GND           1 use

I/O primitives:
IBUF          53 uses
OBUF_F_24     53 uses

I/O Register bits:            0
Register bits not including I/Os:   0 (0%)

Mapping Summary:
Total  LUTs: 89 (0%)

Mapper successful!
Process took 5.258 seconds realtime, 5.257 seconds cputime
```

Then some i o register bits are also available. Right now doesn't use any of this. You have already seen LUTs which is not listed here. So, in addition to this gate you need look up tables as well and that is quite substantial 89 %.

It reports that the 89 is only 0 %. In fact it is not 0, actually it is something less than 1 %. So it is reporting as 0, so actually how many such will be available, will have to look into the vendor data sheet as to how many LUTs you can have and or in place and route it may report that as well out of how many. Once this mapping is successful it reports that it is successful and how much time it took for processing whether be it real time or CPU time it reports this here.

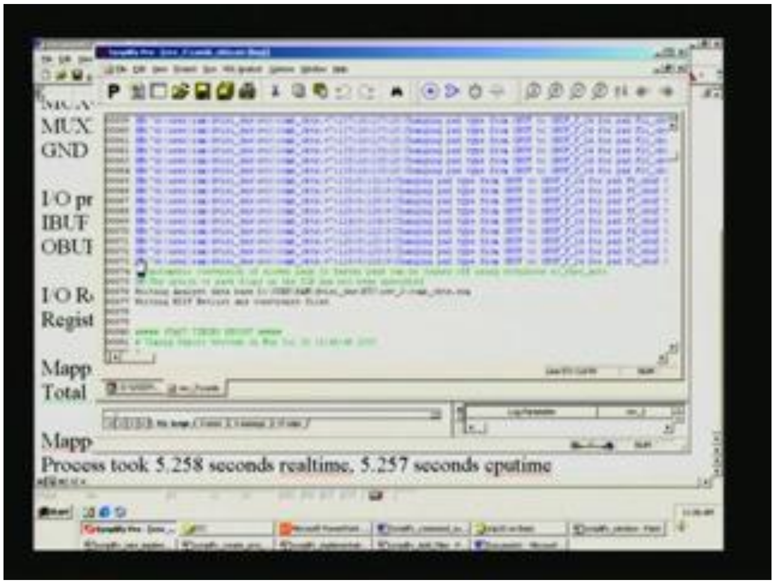
(Refer Slide Time: 17:38)



Process took 5.258 seconds realtime, 5.257 seconds cputime

We have seen how to analyze the report file. Basically, what you had to in summary what you had to see is i stands for the information, and n for note. Have a look at the notes and what it has to say and where it is critical you had to take a note and do the need full.

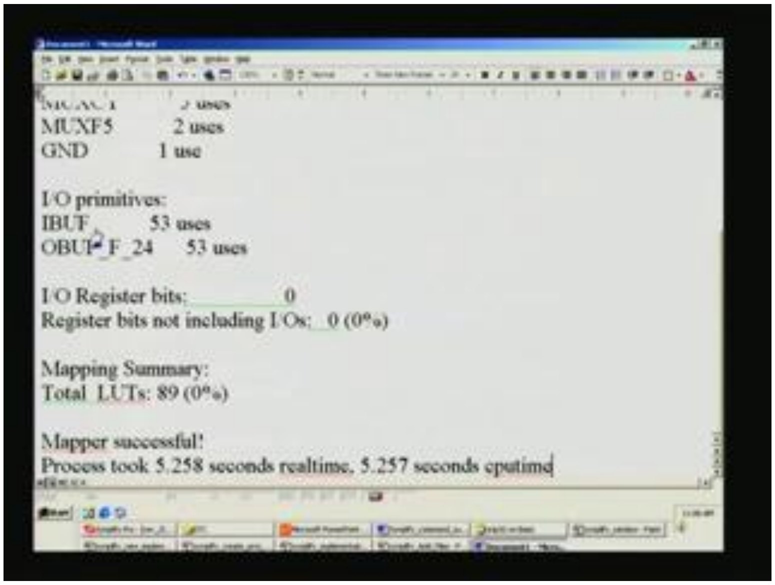
(Refer Slide Time: 17:57)



Process took 5.258 seconds realtime, 5.257 seconds cputime

See for example, you have a warning it is not right here, that is no warning. So look for warning and then take corrective action. You have a question. Going back to that report if it says 53 uses, it means 53 instances of that have been used is it?

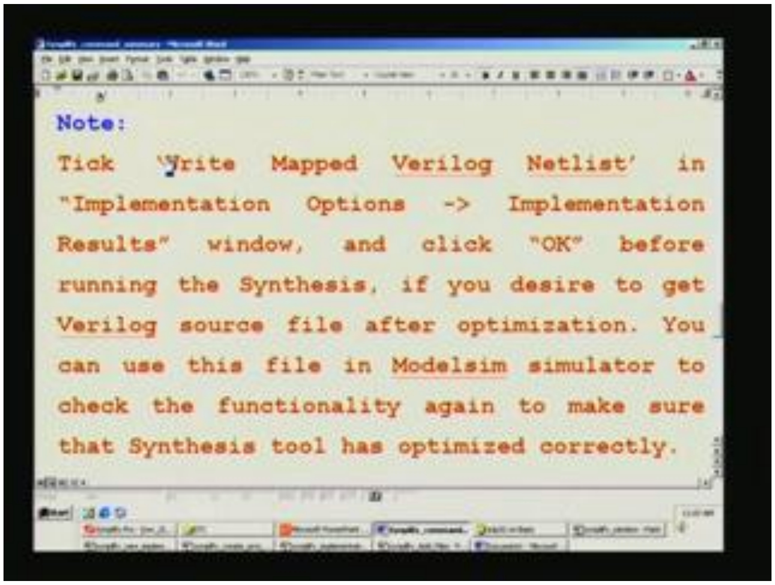
(Refer Slide Time: 18:18)



```
MUXCY      3 uses
MUXF5      2 uses
GND        1 use

I/O primitives:
IBUF       53 uses
OBUF_F_24     53 uses

I/O Register bits:          0
Register bits not including I/Os:  0 (0%)

Mapping Summary:
Total  LUTs: 89 (0%)

Mapper successful!
Process took 5.258 seconds realtime, 5.257 seconds cputime
```

Yes, we will see when you look at the schematic. We will have a look we can physically count them and see verify whether it is tallying. 53 input buffers have been used. Yes, that is true or the input buffer has been touched 53 times is it? No it has been used. What we are doing now is we are actually mapping the hardware. So physically they have got to be there so many buffers will be mapped.

(Refer Slide Time: 18:49)



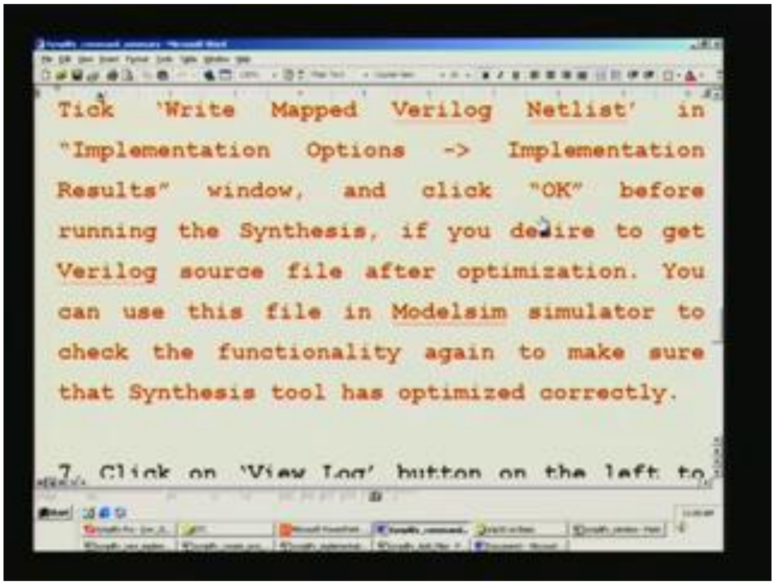
Note:

Tick 'Write Mapped Verilog Netlist' in "Implementation Options -> Implementation Results" window, and click "OK" before running the Synthesis, if you desire to get Verilog source file after optimization. You can use this file in Modelsim simulator to check the functionality again to make sure that Synthesis tool has optimized correctly.

Actually we were in step 6 of the summary after run, we have not come back. Now we will do that. So we are going to learn one more thing here, and what it says is, so in the path of this run and we have seen the report file so far. So I have put a note here. I will just read out this. Tick, write mapped Verilog net list in implementation options, and implementation results window and click ok before running the synthesis. You had used the run for synthesis running that is to compile and map and finally get the report file. Also create a file and so when you do this one before that you have an option of creating Verilog file output. What I mean is, after optimization, it will create a Verilog file which you can take into a modelsim, any simulator for that matter and then make sure that after optimization your functionality is still intact. Are you clear?

This is a necessary step and many designers do not bother much about this and I do not really emphasize this. It is up to you once you have gained some confidence at the beginning stage, you can do this and make sure it is. Once you're satisfied, you can always bypass this step and go to the place and route. After that also, you have to come back as we have seen earlier. So we are anyway going to come back number of times. So perhaps, towards the end if you do this, that may be sufficient or at start with you as a beginner you can do first time, and then differ it to the very final thing. You have a question? Is there ever a possibility that the synthesis tool might agree and the modelsim might not like once you get the Verilog output Verilog file?

(Refer Slide Time: 20:50)

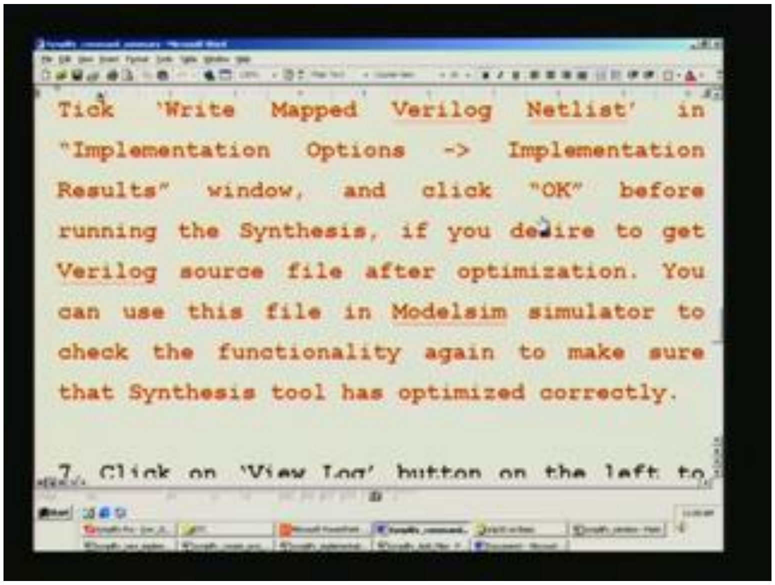


Tick 'Write Mapped Verilog Netlist' in "Implementation Options -> Implementation Results" window, and click "OK" before running the Synthesis, if you desire to get Verilog source file after optimization. You can use this file in Modelsim simulator to check the functionality again to make sure that Synthesis tool has optimized correctly.

7. Click on 'View Log' button on the left to

Yes, and you go back to modelsim and run it. Is there a possibility that there might be a discrepancy? How do you say there will be discrepancy? This step is not all that necessary like you said. Neither simplified tool or for that matter, neither any synthesis tool nor the simulator will actually decide. In fact who is the deciding authority are you yourself. You have to know right? So the simulation tool cannot say it is wrong. That much intelligence, if you had injected in the test bench then, it can take care. If not you have to analyze it by yourselves, by looking at the wave form and making sure that the functionality is intact. So it is up to make the decision.

(Refer Slide Time: 21:41)



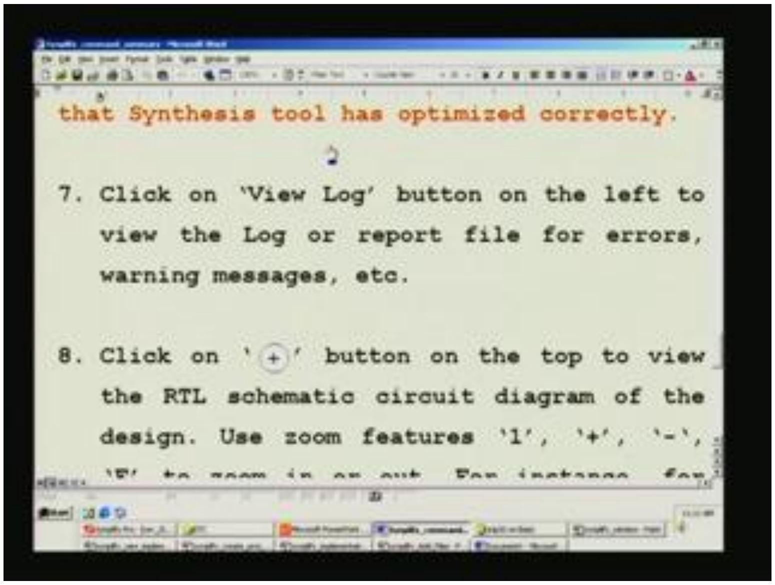
Tick 'Write Mapped Verilog Netlist' in "Implementation Options -> Implementation Results" window, and click "OK" before running the Synthesis, if you desire to get Verilog source file after optimization. You can use this file in Modelsim simulator to check the functionality again to make sure that Synthesis tool has optimized correctly.

7. Click on 'View Log' button on the left to

What we were reading out is tick mapped Verilog net list in implementation options, implementation results window. Click before running the synthesis, if you desire to get Verilog source file after optimization. That is what we have already seen. You can use this file in modelsim simulator, to check the functionality again to make sure that synthesis tool has optimized correctly. If you analyze to this simulator once again, it will reveal, whether it has done over optimized things. Sometimes it will.

So do not take this synthesis tool for granted, it is for you to find out it has not over optimized. As I mentioned, if you recollect guidelines, I might have shown might delay being realized, by using number of cascading number of buffers or invertors. If you put so many buffers and run through this synthesis tool, this tool will unceremoniously remove all the buffers and just retain only your input buffer. Here so what happened to your functionality? Right? You had so much trust on the tool and it has removed it.

(Refer Slide Time: 23:02)



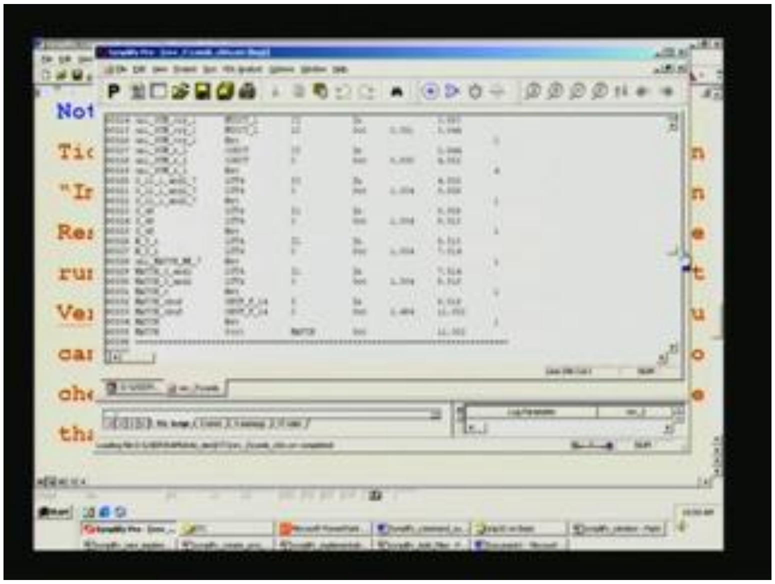
that Synthesis tool has optimized correctly.

7. Click on 'View Log' button on the left to view the Log or report file for errors, warning messages, etc.

8. Click on '⊕' button on the top to view the RTL schematic circuit diagram of the design. Use zoom features '1', '+', '-', 'F' to zoom in or out. For instance, for

If you do not decide that one, you will have to find out ways and means of circumventing this is it. May be difficult, especially with that sort of thing, what I have mentioned. It may be difficult and synthesis tool will remove it deliberately so.

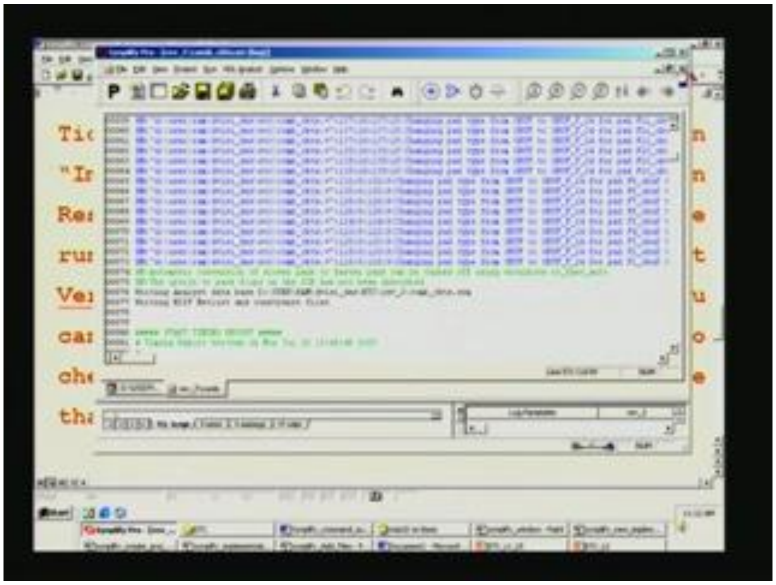
(Refer Slide Time: 23:18)



You have to be on alert and that is found only if you go through this. This take back this optimized file and find out whether the functionality has not changed. I have one more question.

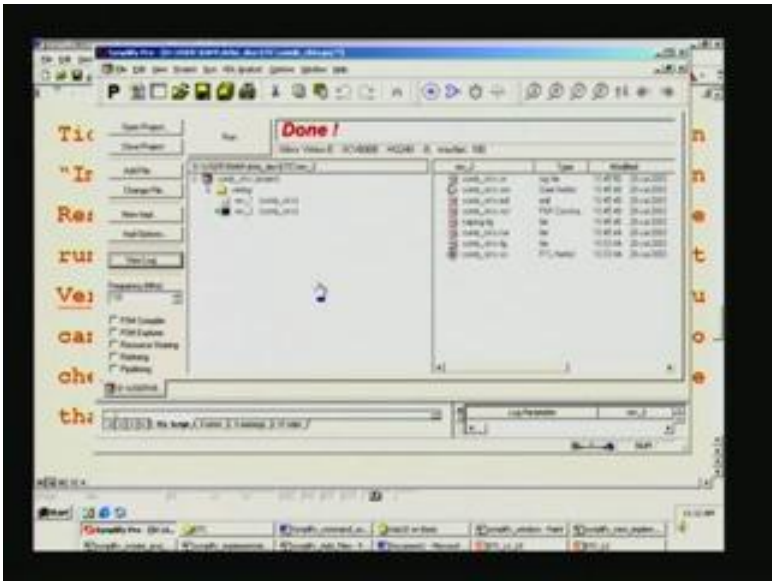
Have you ever mentioned why this synthesis tool deliberately removes the buffers? Why this synthesis tool, deliberately removes the delayed buffers? Because that is the role of synthesis tool is precisely that. So we will see this particular thing here. How to create this?

(Refer Slide Time: 23:56)



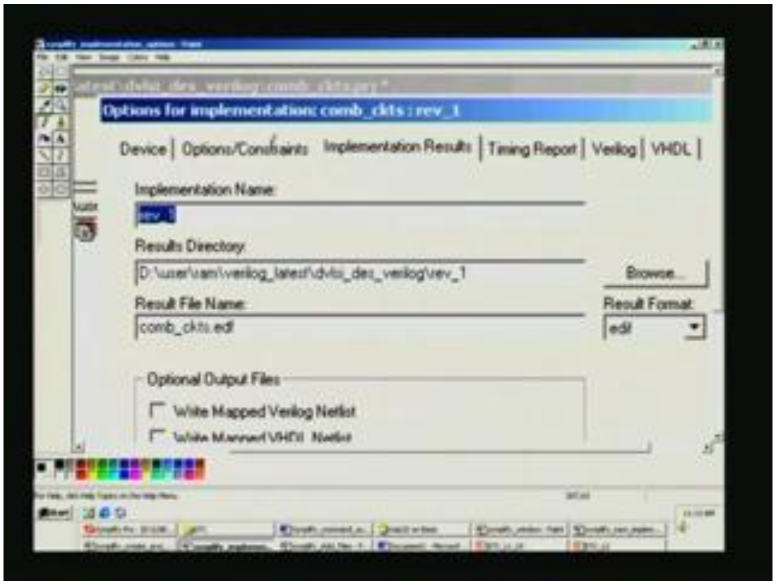
We will go to the simplified tool and we do not require this log file so we can just displace it.

(Refer Slide Time: 24:01)



Done !

(Refer Slide Time: 24:10)



Options for implementation: comb_ckts : rev_1

Implementation Name: rev_1

Results Directory: D:\user\ram\verilog_latest\dvlsi_des_verilog\rev_1

Result File Name: comb_ckts.edf

Result Format: edif

Optional Output Files

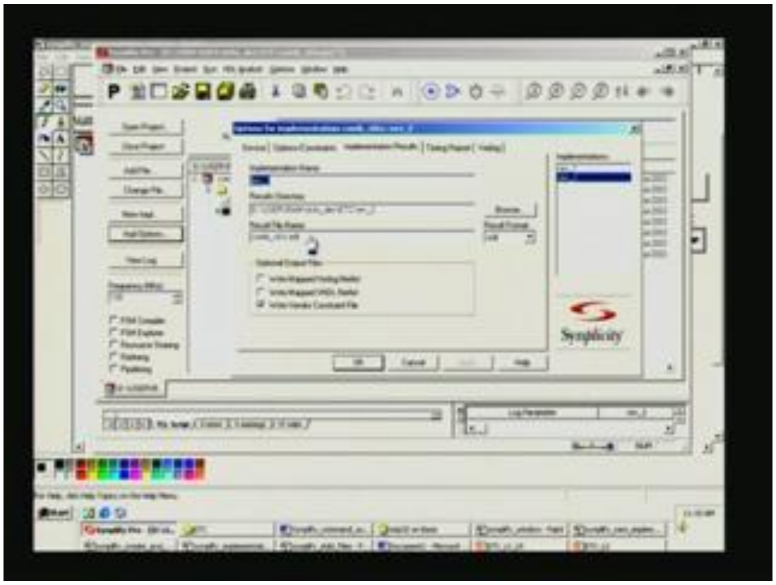
□ Write Mapped Verilog Netlist

□ Write Mapped VHDL Netlist

Before that the window is here. I will just show this as zoomed version. You see this in implementation options window and you have the different menus. The device option constraints implementation results, if you click on this implementation results, you get this sort of window. Here, what it says is your file was actually combination circuits, and may be the revision was, revision 1.

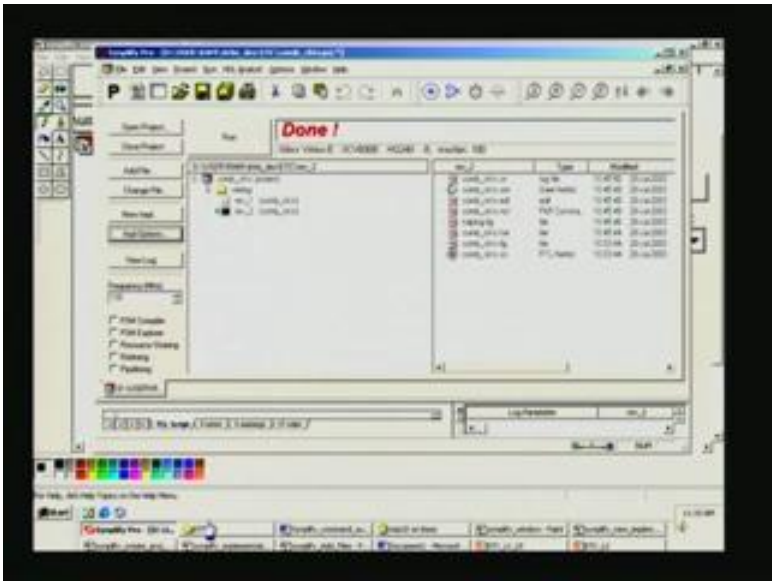
Because every time you recompile may be with a change of something, in order to prevent over writing so you have different revisions available. It also creates a result file and as I mentioned you need an EDIF file and that is what is here. So combination circuits for your design and it creates an EDIF file with an extension of dot, I mean edf here. And just now we have read in a command summary. We need Verilog net list what it should do is, just tick this. If you tick this one then only this optimized Verilog net list will be created. This will be a dot b m extension, and it is same as dot v. So you can use that and run this simulation.

(Refer Slide Time: 25:57)



Actually, we will just see that here, so in implement option we have already seen. So there is one implementation results here. It lists the combination circuits dot edf, this is the output file and this is very important in the sense that this will be export to the place en-route later on and here is what I mentioned write mapped Verilog let net list here and if I say.

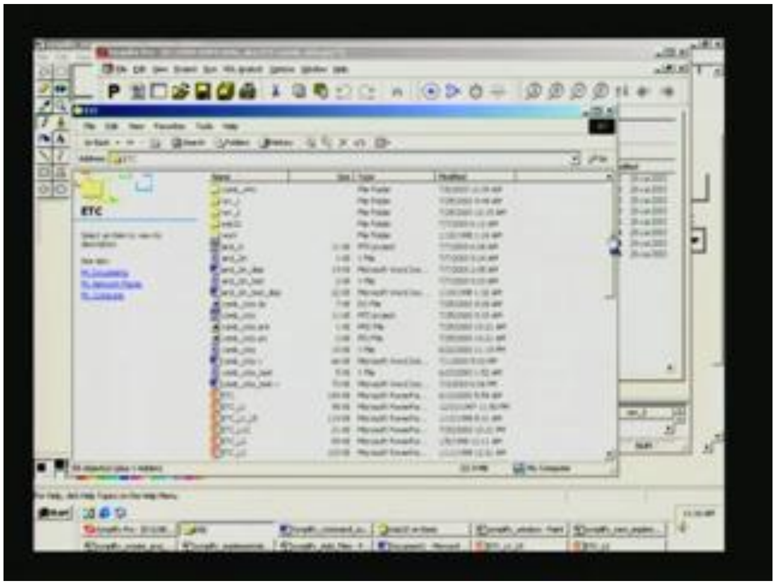
(Refer Slide Time: 26:25)



Done !

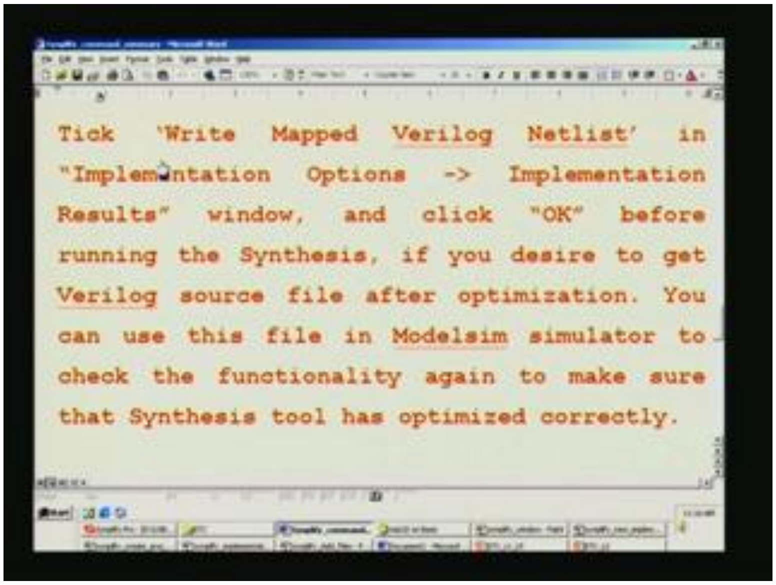
Then say run. I mean dot v file will be created. Let us have a look now. Where are we? Let us go to this.

(Refer Slide Time: 27:03)



I will just put details so that will be easier. We will run this again. So I mean add file, we do not have to give. Because, it is the same design file that we have. So what all difference it makes is it creates one dot v m file. Let us see, whether I think this is the path, you just make a note how the path, otherwise you make it lost and it may go to some other folder. We will have to keep track of it. Now what we have here is.

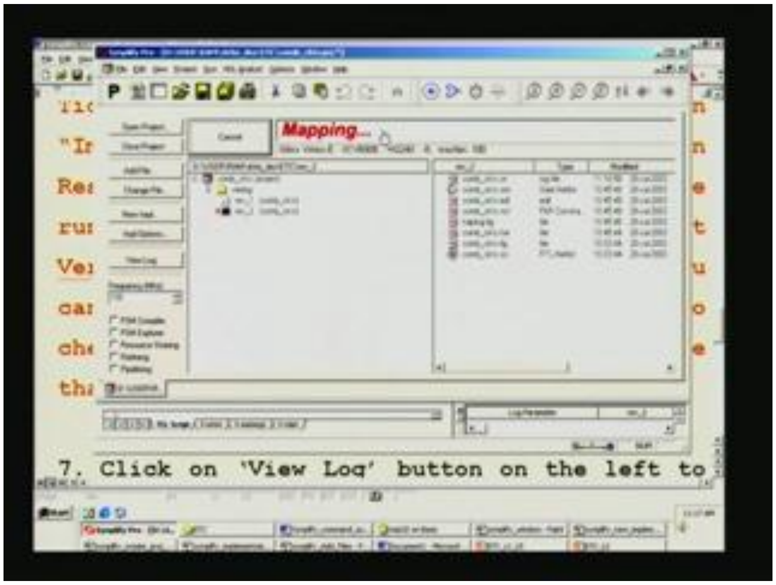
(Refer Slide Time: 27:47)



Tick 'Write Mapped Verilog Netlist' in "Implementation Options -> Implementation Results" window, and click "OK" before running the Synthesis, if you desire to get Verilog source file after optimization. You can use this file in Modelsim simulator to check the functionality again to make sure that Synthesis tool has optimized correctly.

This is what we have seen tick write Verilog net list in implementation options, implementation results window. Click before running the synthesis if you desire to get Verilog source file after optimization so this is all that you have seen here so now coming back to this.

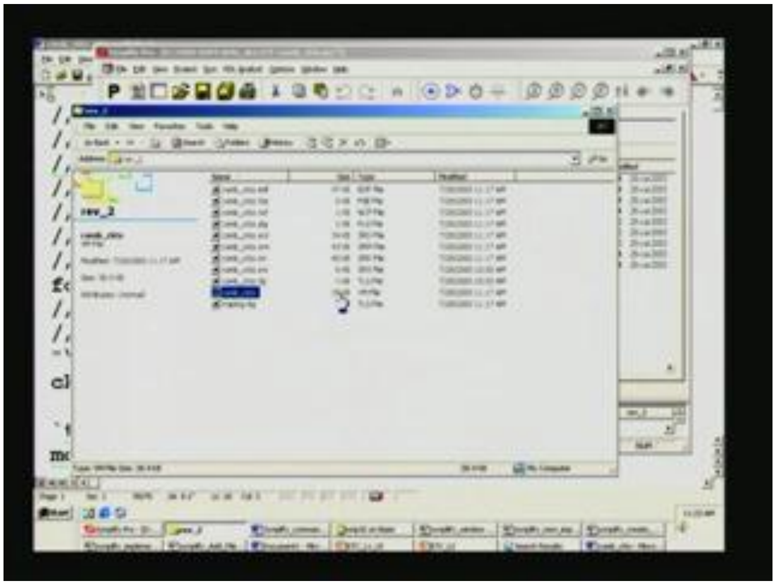
(Refer Slide Time: 28:03)



7. Click on 'View Log' button on the left to

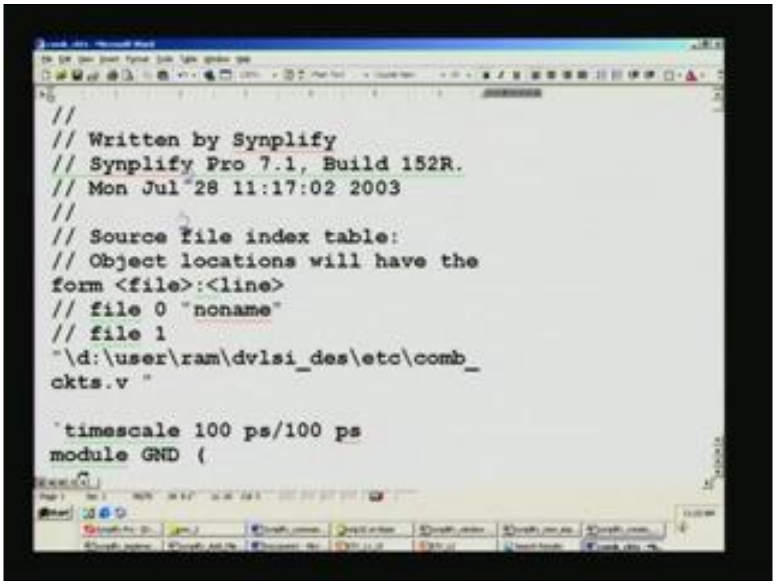
Let us run this, so then the compiling then it is going onto mapping and it is done. Let us see whether it has created. So let it be since that dot v m file is created and after this there is one more folder called revision 2 in that folder that has been created.

(Refer Slide Time: 28:38)



Actually this is they wanted and it may be difficult for you. I will just locate it for you and where is it? This is the one. Note that, this is quite a huge file, because it will have to reflect all the primitive cells etc.

(Refer Slide Time: 29:01)

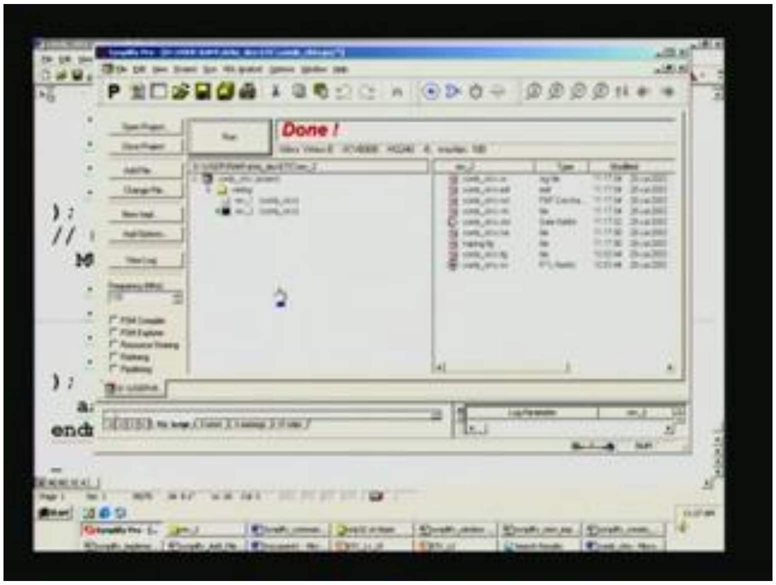


```
//
// Written by Synplify
// Synplify Pro 7.1, Build 152R.
// Mon Jul 28 11:17:02 2003
//
// Source file index table:
// Object locations will have the
form <file>:<line>
// file 0 "noname"
// file 1
"\d:\user\ram\dvlsi_des\etc\comb_
ckts.v "

`timescale 100 ps/100 ps
module GND (
```

We can see this file. This is the one. Synplify has created at what time it was created and so on. Precisely, the same date time just now done and the path it gives and your design file is identified and this is your own design file. It will be very difficult for you to make out your own design after having look at this, except for timescale which might you have, this also we did not use. This is being inserted by synthesis tool. We did not give this.

There are various say for example ground is also considered as some module, and it has used once again as assign statements and the module is ended. Every small function will be treated here as a module and reported and you can see 2 input LUT used here. One output of course and I not I 1 from your design has been taken as input for this LUT and then this modules worked out and once again declaration there, whether their wires and so on. And in this fashion you can make out. It will be very tedious for you to wait through this and if you see here if I run it; it keeps on running, you see everywhere it is same LUT 2 or LUT 4 or LUT 3. LUT 3 again so in this fashion you can go a little quickly. IMUX buffer we have already seen that is also being used here. Then a MUX here, exclusive or and all your signals you can recognize of course. This is quite a big one because this is the actual design. Your design is this combination circuits so it is listing that module. All that you have keyed in is put here. Again input output listing here then wire all the declaration that you have had in your design file. Once again you see LUT 4, LUT 3 used then output buffers. Finally it ends with some MUX so by now you would be clear as to what FPGA contains basically, the basic components obviously are MUX then LUT 2 inputs 3 inputs 4 inputs. There are a few version of MUX and then this is purely a combination circuits so you did not see any d flip flops etc., if you see sequential circuits you can view those.

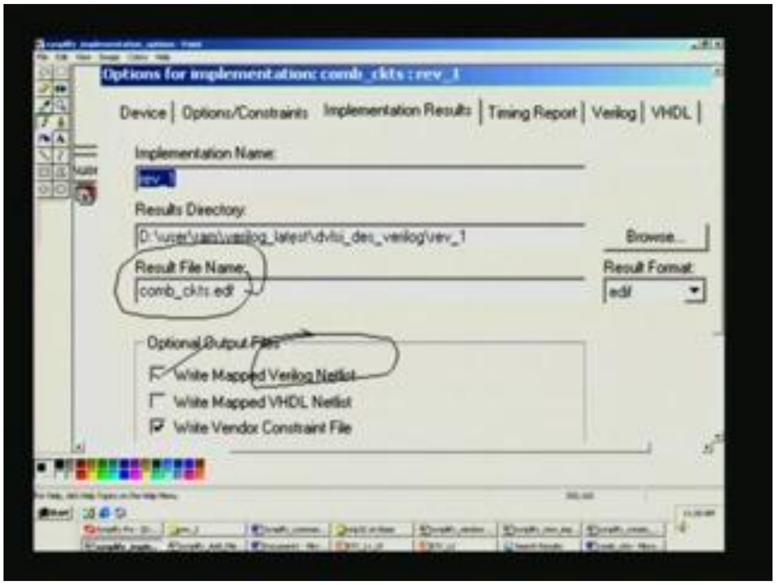
(Refer Slide Time: 32:47)



Done !

What we will see next is, we have so far seen how to analyze using a log file and also seen how to create the optimized dot v file and dot v m file and dot v file are same. If you want you can rename it as dot v file and use in modelsim, whatever you had done earlier with your design you can do precisely same thing with optimized dot v file. Make sure that the functionality is not changed and that will confirm the optimization has taken place in the right way.

(Refer Slide Time: 34:00)



Options for implementation: comb_ckts : rev_1

Implementation Name: rev_1

Results Directory: D:\user\ram\verilog_latest\dvlsi_des_verilog\rev_1

Result File Name: comb_ckts.edf

Result Format: edf

Optional Output Files:

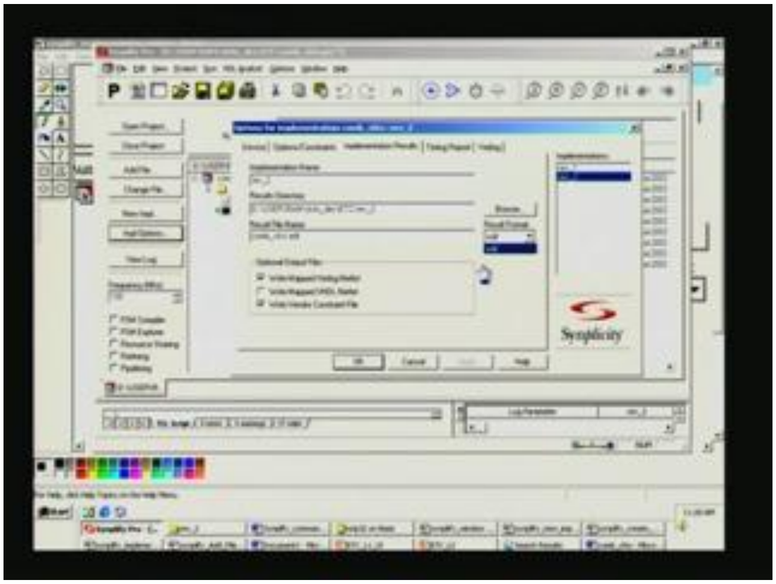
☑ Write Mapped Verilog Netlist

☐ Write Mapped VHDL Netlist

☑ Write Vendor Constraint File

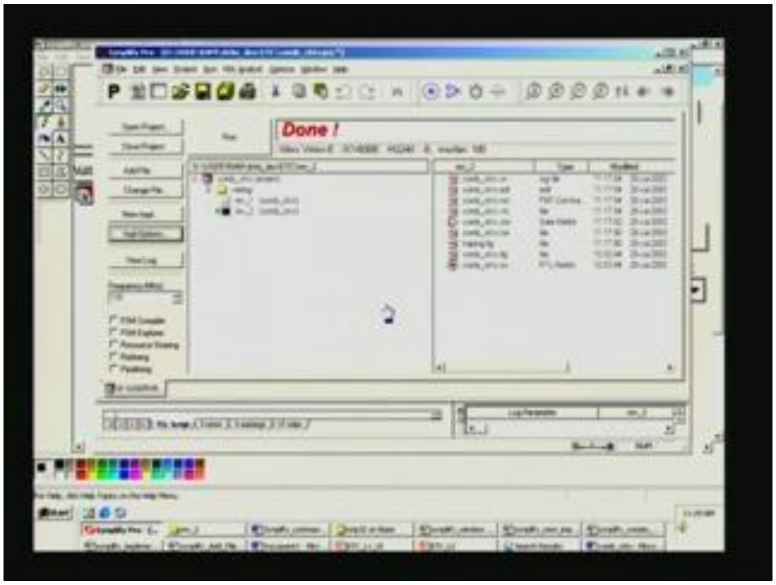
This is the window we had ticked here. For getting Verilog dot v m file and I mentioned that here as one edf file created, this is an output created for use in place-n-route and here if you click on this you may get some other option. I do not think any further option is available other than this format. It only says the result in what format you want and this is the path where your results are stored.

(Refer Slide Time: 34:41)



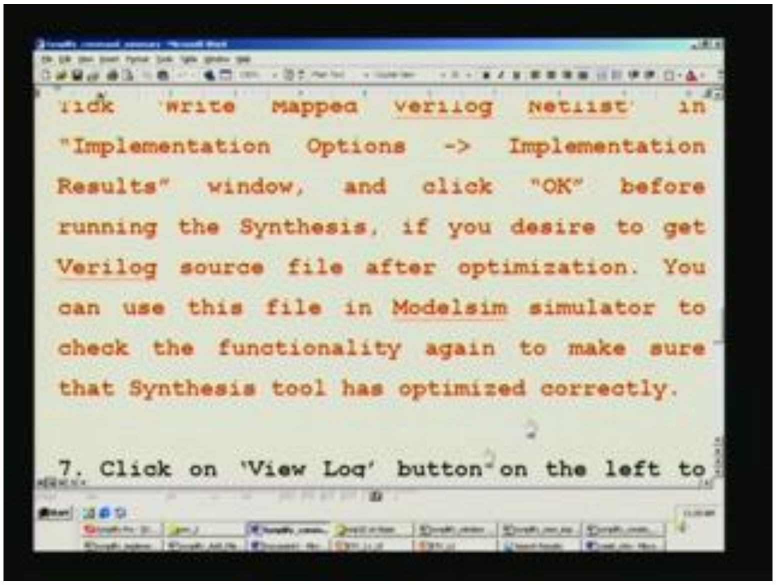
Here this was the window here and if you click here you may not get any other option, see it lists only EDIF here. That is the only option available because this is adequate for taking it into place-n-route.

(Refer Slide Time: 34:56)



You can have a look at it and it will look more or less like this dot v m file so I will leave it as an exercise for you to use any word I think we can use and find out. It may be little shorter list, fundamentally it should contain an information about all these cells and what actual cells are put in. These are all required for place-n-route without which it cannot place or route. The next thing we will consider is we will go to that command summary.

(Refer Slide Time: 35:30)



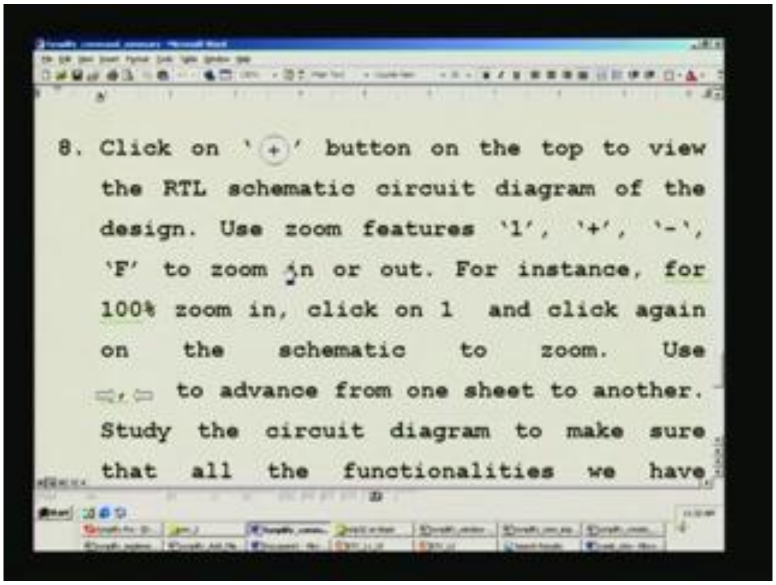
Tick 'Write Mapped Verilog Netlist' in "Implementation Options -> Implementation Results" window, and click "OK" before running the Synthesis, if you desire to get Verilog source file after optimization. You can use this file in Modelsim simulator to check the functionality again to make sure that Synthesis tool has optimized correctly.

7. Click on 'View Log' button on the left to

We have seen this and creating of the Verilog file and then this I am not showing right now, in future I might show. After optimization you can have I will leave it as some exercise for the time being, if time permits we will have a look into that also. This we have already seen, click on view log button on the left to view the log or report file for errors warning messages etc, there were no warning messages. If you create some errors deliberately you will get some warning messages. This we can see later on and first what we will do is we will see how to look at this schematic. Whatever design you have, it will be helpful if you can have a schematic version of that. So fortunately you have that in this simplified tool. Do you have any questions?

Can you tell us the error correction cycle, of how you correct an error and finding error synthesis? I will go back to Verilog, rewrite the code come to modelsim. Yes, simulate it there, we have already seen in a flow chart earlier. How this whole thing is an iterative process, I will be covering deliberately injecting as I mentioned earlier, errors and then see both modelsim as well as synthesis how it behaves. We can get a feel of how the compiling errors will arrive and the ways and means to correct them. I will read out this. We have what are called to views. Earlier I had shown in the slide, right now I cannot spot it out and in this case that you want to view this schematic there are actually two views so one is called RTL view, RTL we are already familiar we have seen RTL guidelines. You have what is called RTL view and this is a concise view of the schematic of your design and there is another view called technology view so this technology view will reflect your primitive cells in the design. It will be more elaborate and it may take just one sheet in RTL view whereas same design will be in several sheets as far as the technology view is concerned.

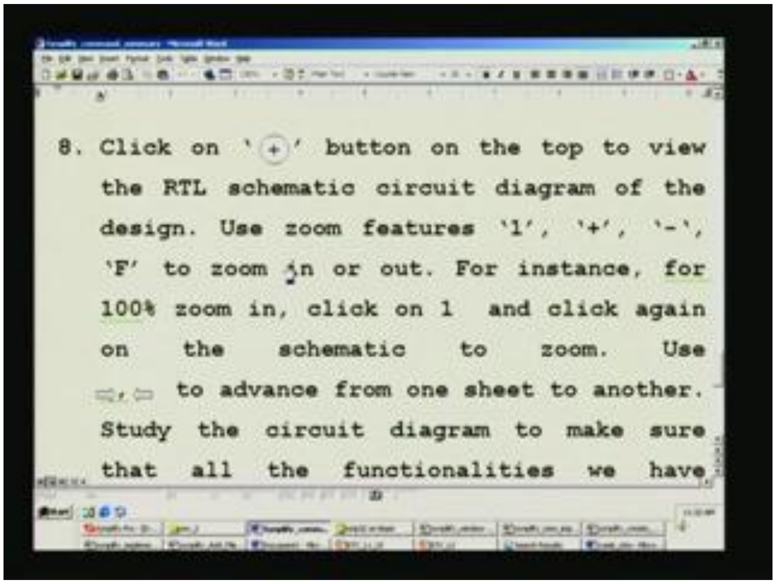
(Refer Slide Time: 38:10)



8. Click on '⊕' button on the top to view the RTL schematic circuit diagram of the design. Use zoom features '1', '+', '-', 'F' to zoom in or out. For instance, for 100% zoom in, click on 1 and click again on the schematic to zoom. Use ⇨, ⇦ to advance from one sheet to another. Study the circuit diagram to make sure that all the functionalities we have

Technology view as the name implies it has the actual technology which goes into the FPGA, so namely the primitive cells that we have seen.

(Refer Slide Time: 38:31)

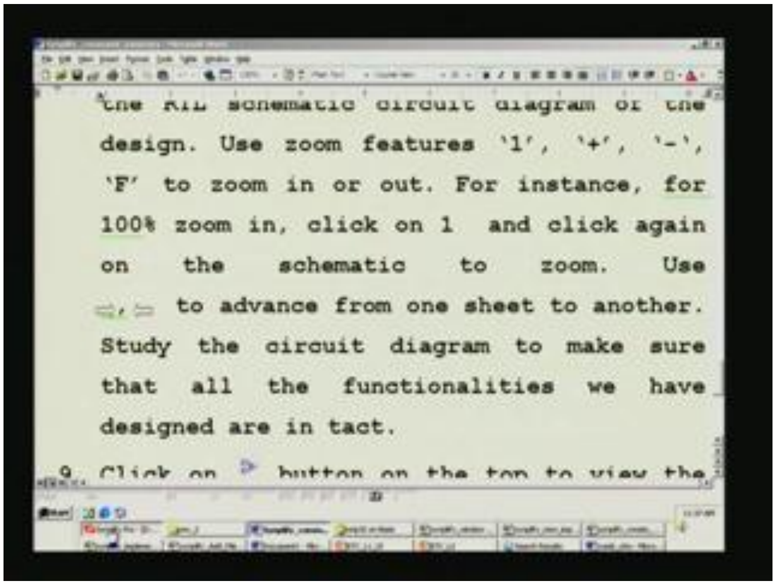


8. Click on '⊕' button on the top to view the RTL schematic circuit diagram of the design. Use zoom features '1', '+', '-', 'F' to zoom in or out. For instance, for 100% zoom in, click on 1 and click again on the schematic to zoom. Use ⇨, ⇦ to advance from one sheet to another. Study the circuit diagram to make sure that all the functionalities we have

I will just read out this before we go onto that. Click on that is a plus symbol, click on this button on the top to view the RTL schematic circuit diagram of the design. Use zoom features 1, there are various buttons available for zooming. If you press 1, you will get full view. Full in the sense highly zoomed view and plus is to zooming to fold every time you click on that and downsizing will be minus. F stands for full view that means the entire schematic will be shown in single page, this is different from this. This is blotted of version they counter, I mean, the reverse of this so that is for zoom and here using this you can either zoom in or zoom out for instance for cent percent zoom in click on 1. Click on 1 first and then go to schematic portion and click on that once again, only then it will be zoomed. On this schematic to zoom use these arrows which will be there in that one of the menus and to advance from one sheet to another and that means there will be several drawing sheets available here. In fact I mentioned earlier the schematic approach even using a cad will be very tedious thing. In fact when you look into some of the complex in the sense, it will rend to several sheets, several tens or even hundreds and depending upon the complexity of the design and especially the more than the RTL view, the technology view will be much more cumbersome to read through. Nevertheless you can go through one sheet after another by using this symbol and study this circuit diagram, once you have seen the circuit diagram study the circuit diagram to make sure that all the functionalities we have designed are intact.

As I mentioned, it will be a very tedious process even for small number of gates, it will be very tedious I think we will give it up in despair and we will see as we progress. Fortunately we do not had to wait through these drawing sheets, once you have gained confidence for small cross checks. The reason is after optimization we are anyway creating the dot v file which we take it back to this simulation tool and then verify the functionality. It boils down to only checking the functionality. You as a designer will be satisfied more than satisfied if your functionality is intact. So how does it matter, what circuit or what gate has been put into your actual design? If you are interested you can have a look but analyzing that is going to be task in itself. The best quickest way of design is to take this dot v file and put into the simulator and even this can be bypass as I mentioned earlier because we had to go for place-n-route even after that stage you had to do what is called back-annotation and that will generate another similar dot v file which will be as laborious as this. That also you can put into the simulator and verify and at that point of time actual gate delays will come into picture. After synthesis if you look at it, some gate delays may get reflected and whereas most of the gate delays are not reflected when you do simulation with this.

(Refer Slide Time: 42:28)

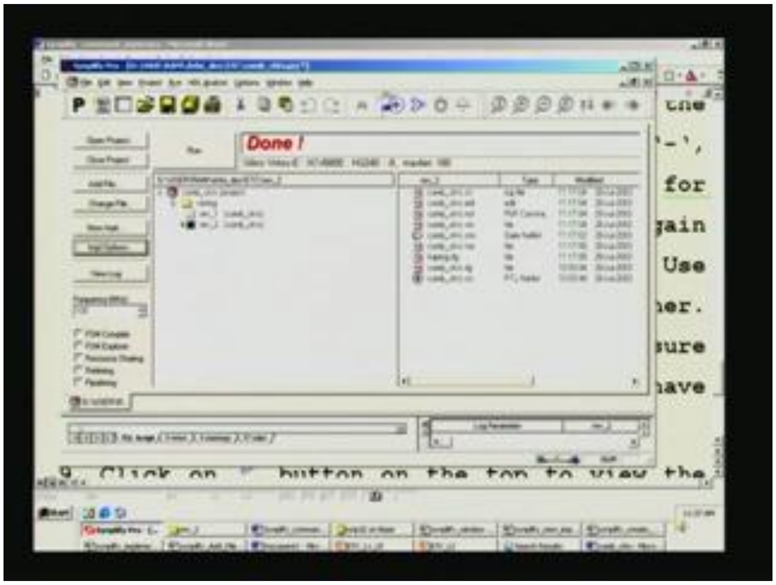


the RTL schematic circuit diagram of the design. Use zoom features '1', '+', '-', 'F' to zoom in or out. For instance, for 100% zoom in, click on 1 and click again on the schematic to zoom. Use ⇨, ⇦ to advance from one sheet to another. Study the circuit diagram to make sure that all the functionalities we have designed are in tact.

9. Click on button on the top to view the

The next step is, let us confine our attention to this.

(Refer Slide Time: 42:41)

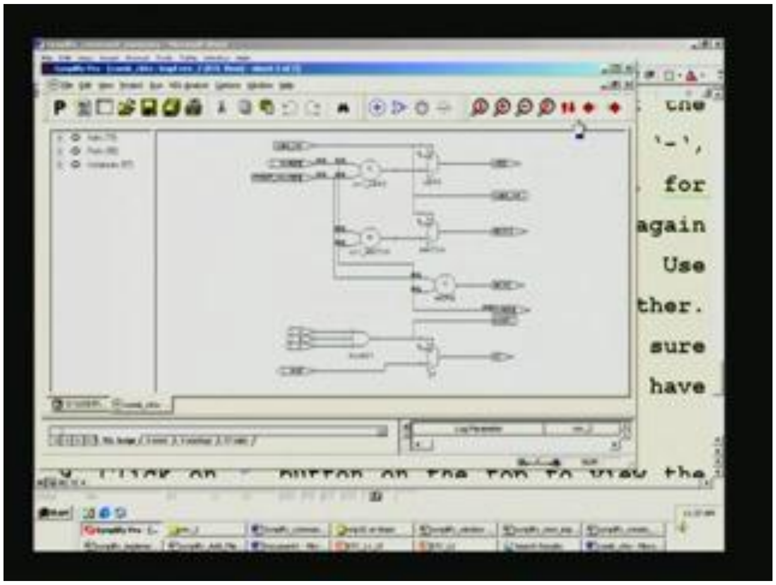


The next step is, let us confine our attention to this.

Let us go to the actual simplified window. As I mentioned we have plus symbol here, if you press this one you should get the schematic of your design, your design is combinational circuits.

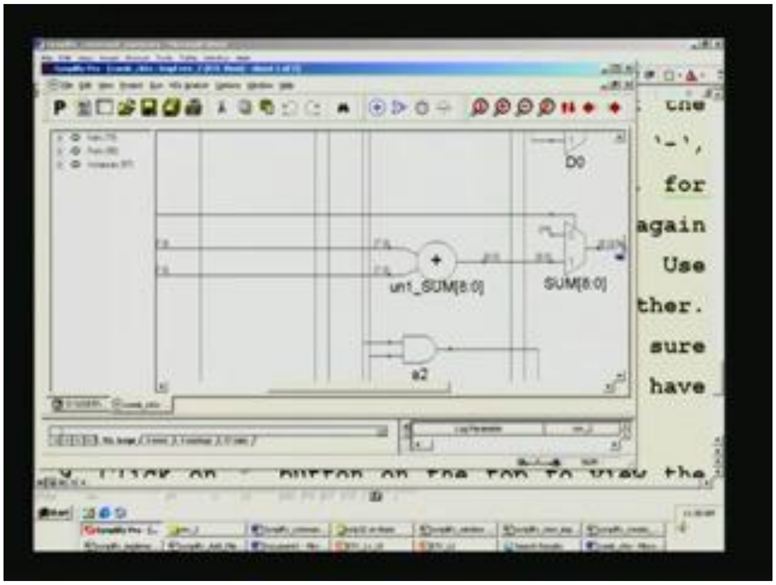
I will just press this, you have the entire circuit here this is RTL. If you see it is not complete actually here if you see I will read it for you. This is called sheet 1 of 2 that means, there are two sheets here. I also mentioned that you can go from one sheet to another, see there is a symbol here I think you can see this symbol. If you click this one, it will go to the next sheet, it has changed here.

(Refer Slide Time: 43:22)



If I again click here, it will revolve around. You can keep on pressing the same thing or go back. There are too many sheets it is better to use back. Even there it will keep on rolling around. Then we have also seen how to zoom. Let us zoom this we will click on 1 and if you do what happened? Nothing happened, it only got highlighted so if you want to really zoom you take your cursor inside the schematic and press that.

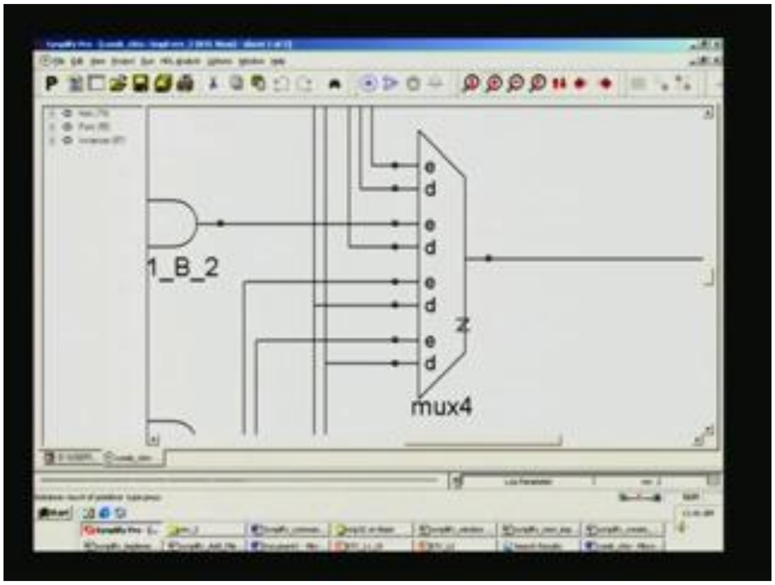
(Refer Slide Time: 43:58)



It is completely zoomed. Can you see it clearly on the monitor? This is you want but unfortunately readability is rather very poor, even after making it big it is pretty difficult. You had to use this here and analyze part by part. For example let me pan it horizontally, you can just see one input popping up here as b and lot many MUXes are used here. 820, 819 they are all numbering to identify different MUX. It may be the same MUX fundamentally. It is nothing but a gate with bubbled here to denote inverter and you see that it is MUX. What you have seen as the primitive cells in the report or rather in that dot vm file. Here it reflects us the actual circuit.

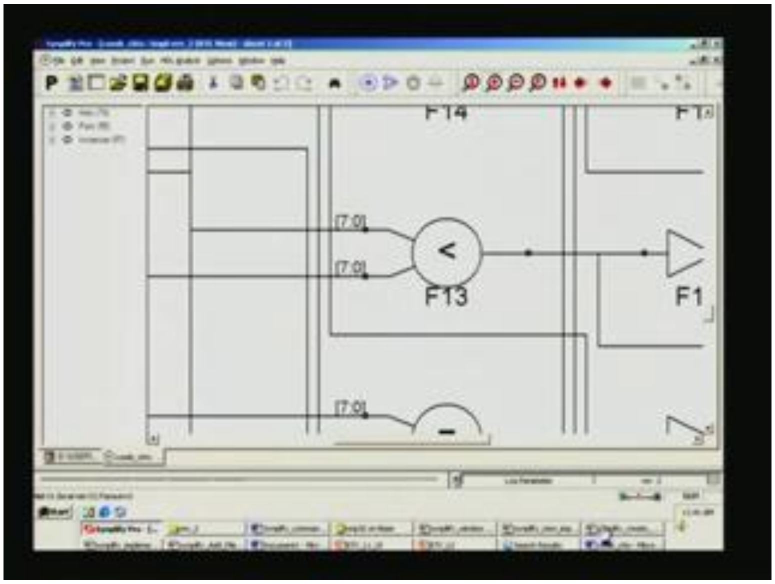
You can see D_6 D_5 are all the outputs that we had in our design and you can see this MUX 8 then D_5 then A B C are the inputs. All are listed here and f_{10} f_{11} f_{12} and so you can have D_4 D_3 and D_2 D_1 f_2 D_0 then SUM here. In some places these are all the numbers you had in 2 numbers compared that is here and it is an eight bit here and if you are not happy with this zoom, you can zoom further. I can just click on plus here and then zoom here. I can go to any amount, only thing is you have to get to the relevant point. So you can number 1, number 2 and enable sum is there. Here it is merely symbolic. The nut divulges what exactly goes into that summing. What we need to do is summing and symbolically represented here. You do not really get the total circuit as such. Once again there is an enabled sum. We will enable this sum to go through and naturally you need a MUX there and you can take some of these circuits, go through patiently see whether it is doing what it is supposed to do. You can see group of MUX here, for MUX 4 actually d stands for data here and e is the enable so there are actually 2 input MUXes inside.

(Refer Slide Time: 47:03)



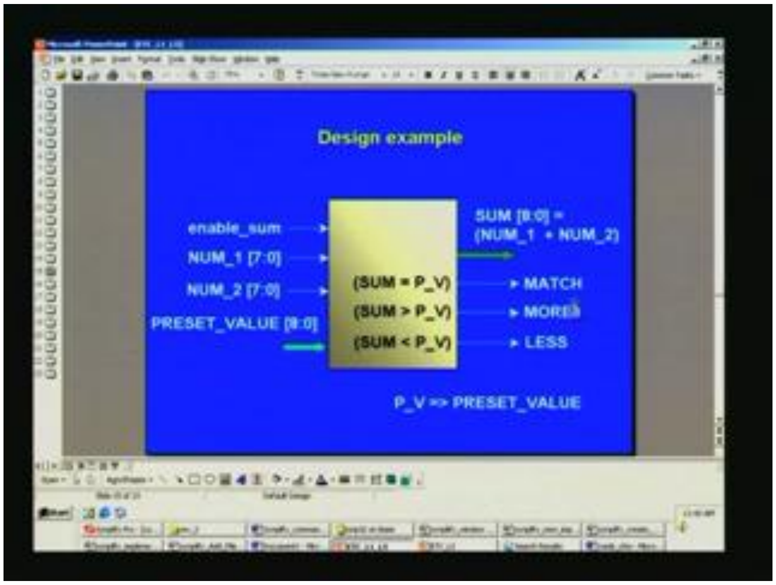
This to abbreviate he has used this notation. That is why you see so many e's here e stands for the enable and d for the data. This is for 4 inputs MUX and we had more-less. That is what it is here.

(Refer Slide Time: 47:38)



N_1 N_2 number compared here. You can see this less than symbol. We had this in our PowerPoint.

(Refer Slide Time: 47:38)



Design example

enable_sum

NUM_1 [7:0]

NUM_2 [7:0]

PRESET_VALUE [8:0]

SUM [8:0] = (NUM_1 + NUM_2)

(SUM = P_V) ► MATCH

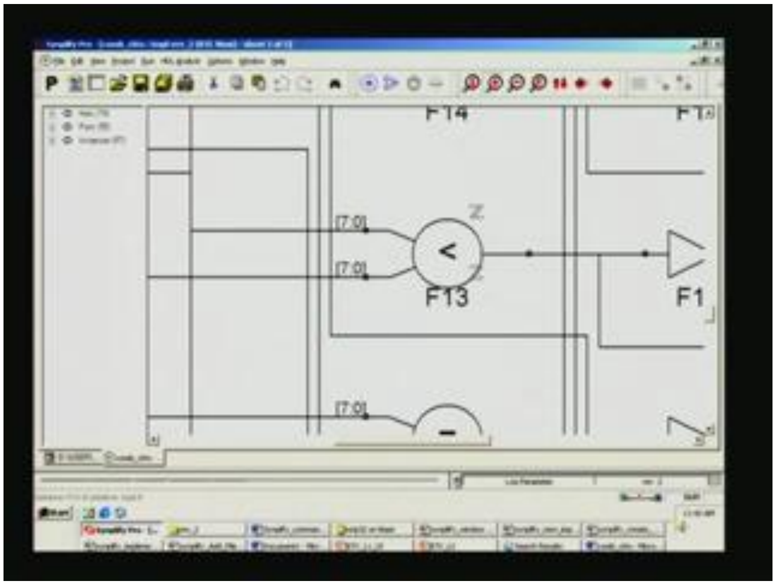
(SUM > P_V) ► MORE

(SUM < P_V) ► LESS

P_V => PRESET_VALUE

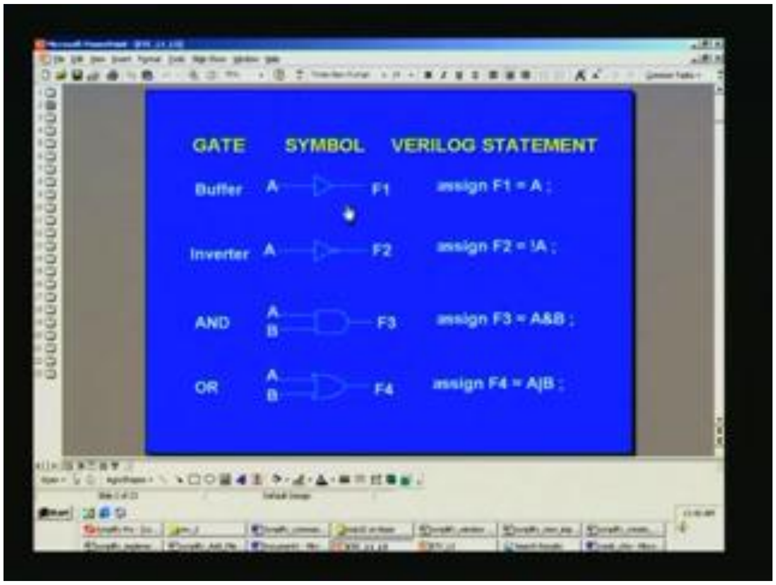
Here number 1, number 2, then preset value, then enable sum was there. Match more than sum. Precisely this same circuit we will see here.

(Refer Slide Time: 47:52)



You can go on to the next. F 1 is there. Let us see just one simple case.

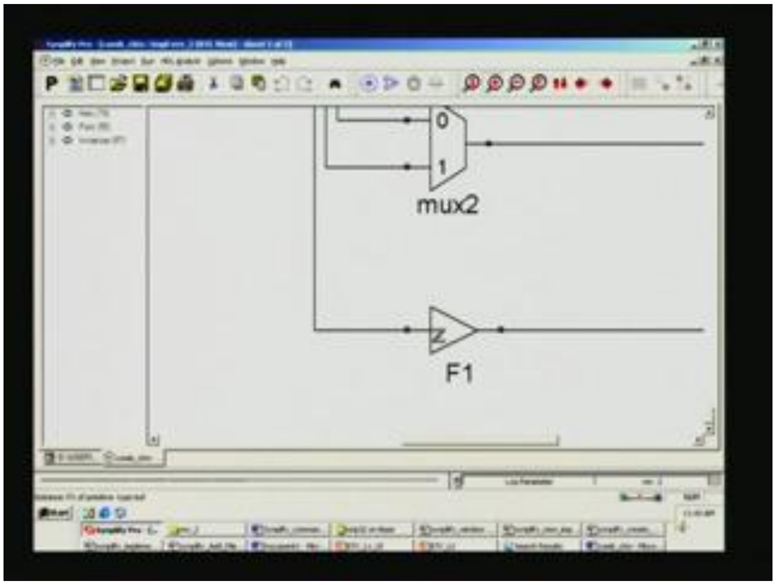
(Refer Slide Time: 48:20)



GATE	SYMBOL	VERILOG STATEMENT
Buffer	A F1	assign F1 = A ;
Inverter	A F2	assign F2 = !A ;
AND	A B F3	assign F3 = A&B ;
OR	A B F4	assign F4 = A|B ;

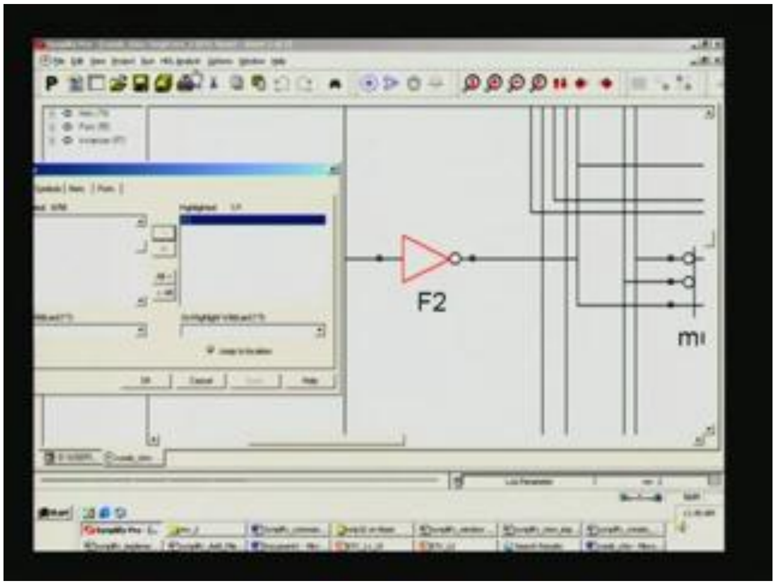
We had just a buffer, buffering A and outputting as $F_{1.}$ If you view this schematic, you should get this. Let us make sure whether precise thing has taken place. Probably we can inspect inverter also so beyond that it becomes very cumbersome for you to track.

(Refer Slide Time: 48:40)



Let us see how we can do. This F_1 is just what we have seen that is buffer here. F_1 is output here. That is a buffer, to that extent it has done its job. Now let us see we will disable this zoom and then you will see the cursor moving here. I will read out for you, something popped up there. What it says is a with its fan out 80 so this is the signal coming from a. If you trace back you will see that it lands up finally in a. This is the left most here. It goes to left and goes up here. This one I suppose I did not miss. See this one but you do not had to go all the way here, so in order to help you, go to the relevant thing, just put at any point here, so it says which signal it is coming from. Straight away it came from pin A. You had just one buffer here. That is what is here and that is energizing F_1 here. Likewise F_2 etc you can analyze. What do we do? How to go to the F_2? Let me have a look. In HDL analyzed here, you have what is called find. Let us find out. All are listed here. All the signals are listed here. What we want is say F_2. F_1 is here, F_2 is here. Let us see what happens. We just say click this arrow here and let us see whether it has gone there.

(Refer Slide Time: 50:42)

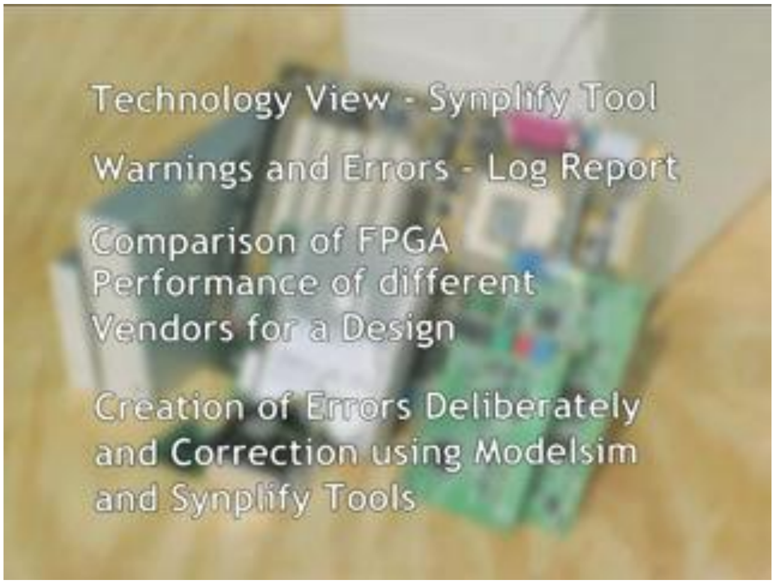


It has gone to that. You can use HDL analyst and then click on the desired signal and click this symbol here, one arrow here and immediately it will go to that point and you see that you can remove that if you want. The color also will change here when compared to other one. Do you see a change of colour? You can go through that and find out. Once as I said that is an inverter here and this signal is A and this signal is F_2 that is what we want. This is how you analyze. As I said you do not want to go through such a laborious way of analysis because you have a dot v m file created. We will continue with technology view in next class. Thank You.

Next Lecture:

Technology View using Synplify Tool

(Refer Slide Time: 52:37)



Technology View - Synplify Tool

Warnings and Errors - Log Report

Comparison of FPGA Performance of different Vendors for a Design

Creation of Errors Deliberately and Correction using Modelsim and Synplify Tools