INTERVIEW /VIVA QUES

VLSI DESIGN AND TECHNOLOGY

c@ S. Balpande. faculty in ET dept ,RCET,Bhilai.

CMOS interview questions.
1/ What is latch up?

Latch-up pertains to a failure mechanism wherein a parasitic thyristor (such as a parasitic silicon controlled rectifier, or SCR) is inadvertently created within a circuit, causing a high amount of current to continuously flow through it once it is accidentally triggered or turned on. Depending on the circuits involved, the amount of current flow produced by this mechanism can be large enough to result in permanent destruction of the device due to electrical overstress (EOS)

2)Why is NAND gate preferred over NOR gate for fabrication?
NAND is a better gate for design than NOR because at the transistor level the mobility of electrons is normally three times that of holes compared to NOR and thus the NAND is a faster gate.
Additionally, the gate-leakage in NAND structures is much lower. If you consider t_phl and t_plh delays you will find that it is more symmetric in case of NAND (the delay profile), but for NOR, one delay is much higher than the other(obviously t_plh is higher since the higher resistance p mos's are in series connection which again increases the resistance).

3)What is Noise Margin? Explain the procedure to determine Noise Margin
The minimum amount of noise that can be allowed on the input stage for which the output will not be effected.

4)Explain sizing of the inverter?
In order to drive the desired load capacitance we have to increase the size (width) of the inverters to get an optimized performance.

5) How do you size NMOS and PMOS transistors to increase the threshold voltage?

6) What is Noise Margin? Explain the procedure to determine Noise Margin?
The minimum amount of noise that can be allowed on the input stage for which the output will not be effected.

7) What happens to delay if you increase load capacitance?
delay increases.

8)What happens to delay if we include a resistance at the output of a CMOS circuit?
Increases. (RC delay)

9)What are the limitations in increasing the power supply to reduce delay?
The delay can be reduced by increasing the power supply but if we do so the heating

effect comes because of excessive power, to compensate this we have to increase the die size which is not practical.

10)How does Resistance of the metal lines vary with increasing thickness and increasing length?
R = (*l) / A.
11)For CMOS logic, give the various techniques you know to minimize power consumption?
Power dissipation=CV2f ,from this minimize the load capacitance, dc voltage and the operating frequency.

12) What is Charge Sharing? Explain the Charge Sharing problem while sampling data from a Bus?
In the serially connected NMOS logic the input capacitance of each gate shares the charge with the load capacitance by which the logical levels drastically mismatched than that of the desired once. To eliminate this load capacitance must be very high compared to the input capacitance of the gates (approximately 10 times).

13)Why do we gradually increase the size of inverters in buffer design? Why not give the output of a circuit to one large inverter?
Because it can not drive the output load straight away, so we gradually increase the size to get an optimized performance.

14)What is Latch Up? Explain Latch Up with cross section of a CMOS Inverter. How do you avoid Latch Up?
Latch-up is a condition in which the parasitic components give rise to the Establishment of low resistance conducting path between VDD and VSS with Disastrous results.

15) Give the expression for CMOS switching power dissipation?
CV2

16) What is Body Effect?
In general multiple MOS devices are made on a common substrate. As a result, the substrate voltage of all devices is normally equal. However while connecting the devices serially this may result in an increase in source-to-substrate voltage as we proceed vertically along the series chain (Vsb1=0, Vsb2 0).Which results Vth2>Vth1.

17) Why is the substrate in NMOS connected to Ground and in PMOS to VDD?
we try to reverse bias not the channel and the substrate but we try to maintain the drain,source junctions reverse biased with respect to the substrate so that we dont loose our current into the substrate.

1 What is the fundamental difference between a MOSFET and BJT ?
In MOSFET, current flow is either due to electrons(n-channel MOS) or due to holes(p-channel MOS) - In BJT, we see current due to both the carriers.. electrons and holes. BJT is a current controlled device and MOSFET is a voltage controlled device.

19)Which transistor has higher gain. BJT or MOS and why?
BJT has higher gain because it has higher transconductance.This is because the current in BJT is exponentially dependent on input where as in MOSFET it is square law.

20)Why do we gradually increase the size of inverters in buffer design when trying to drive a high capacitive load? Why not give the output of a circuit to one large inverter?
We cannot use a big inverter to drive a large output capacitance because, who will drive the big inverter? The signal that has to drive the output cap will now see a larger gate capacitance of the BIG inverter.So this results in slow raise or fall times .A unit inverter can drive approximately an inverter thats 4 times bigger in size. So say we need to drive a cap of 64 unit inverter then we try to keep the sizing like say 1,4,16,64 so that each inverter sees a same ratio of output to input cap. This is the prime reason behind going for progressive sizing.

21)In CMOS technology, in digital design, why do we design the size of pmos to be higher than the nmos.What determines the size of pmos wrt nmos. Though this is a simple question try to list all the reasons possible?
In PMOS the carriers are holes whose mobility is less[aprrox half] than the electrons, the carriers in NMOS. That means PMOS is slower than an NMOS. In CMOS technology, nmos helps in pulling down the output to ground ann PMOS helps in pulling up the output to Vdd. If the sizes of PMOS and NMOS are the same, then PMOS takes long time to charge up the output node. If we have a larger PMOS than there will be more carriers to charge the node quickly and overcome the slow nature of PMOS . Basically we do all this to get equal rise and fall times for the output node.

22)Why PMOS and NMOS are sized equally in a Transmission Gates?
In Transmission Gate, PMOS and NMOS aid each other rather competing with each other. That's the reason why we need not size them like in CMOS. In CMOS design we have NMOS and PMOS competing which is the reason we try to size them proportional to their mobility.

23)All of us know how an inverter works. What happens when the PMOS and NMOS are interchanged with one another in an inverter?
I have seen similar Qs in some of the discussions. If the source & drain also connected properly...it acts as a buffer. But suppose input is logic 1 O/P will be degraded 1

Similarly degraded 0;

24)A good question on Layouts. Give 5 important Design techniques you would follow when doing a Layout for Digital Circuits?
a)In digital design, decide the height of standard cells you want to layout.It depends upon how big your transistors will be.Have reasonable width for VDD and GND metal paths.Maintaining uniform Height for all the cell is very important since this will help you use place route tool easily and also incase you want to do manual connection of all the blocks it saves on lot of area.
b)Use one metal in one direction only, This does not apply for metal 1. Say you are using metal 2 to do horizontal connections, then use metal 3 for vertical connections, metal4 for horizontal, metal 5 vertical etc...
c)Place as many substrate contact as possible in the empty spaces of the layout.
d)Do not use poly over long distances as it has huge resistances unless you have no other choice.
e)Use fingered transistors as and when you feel necessary.
f)Try maintaining symmetry in your design. Try to get the design in BIT Sliced manner.

25)What is metastability? When/why it will occur?Different ways to avoid this?

Metastable state: A un-known state in between the two logical known states.This will happen if the O/P cap is not allowed to charge/discharge fully to the required logical levels.
One of the cases is: If there is a setup time violation, metastability will occur,To avoid this, a series of FFs is used (normally 2 or 3) which will remove the intermediate states.

26)Let A and B be two inputs of the NAND gate. Say signal A arrives at the NAND gate later than signal B. To optimize delay of the two series NMOS inputs A and B which one would you place near to the output?
The late coming signals are to be placed closer to the output node ie A should go to the nmos that is closer to the output.

Digital design interview questions & answers.

1) Explain about setup time and hold time, what will happen if there is setup time and hold tine violation, how to overcome this?
Set up time is the amount of time before the clock edge that the input signal needs to be stable to guarantee it is accepted properly on the clock edge.
Hold time is the amount of time after the clock edge that same input signal has to be held before changing it to make sure it is sensed properly at the clock edge.
Whenever there are setup and hold time violations in any flip-flop, it enters a state where

its output is unpredictable: this state is known as metastable state (quasi stable state); at the end of metastable state, the flip-flop settles down to either '1' or '0'. This whole process is known as metastability

2) What is skew, what are problems associated with it and how to minimize it?
In circuit design, clock skew is a phenomenon in synchronous circuits in which the clock signal (sent from the clock circuit) arrives at different components at different times. This is typically due to two causes. The first is a material flaw, which causes a signal to travel faster or slower than expected. The second is distance: if the signal has to travel the entire length of a circuit, it will likely (depending on the circuit's size) arrive at different parts of the circuit at different times. Clock skew can cause harm in two ways. Suppose that a logic path travels through combinational logic from a source flip-flop to a destination flip-flop. If the destination flip-flop receives the clock tick later than the source flip-flop, and if the logic path delay is short enough, then the data signal might arrive at the destination flip-flop before the clock tick, destroying there the previous data that should have been clocked through. This is called a hold violation because the previous data is not held long enough at the destination flip-flop to be properly clocked through. If the destination flip-flop receives the clock tick earlier than the source flip-flop, then the data signal has that much less time to reach the destination flip-flop before the next clock tick. If it fails to do so, a setup violation occurs, so-called because the new data was not set up and stable before the next clock tick arrived. A hold violation is more serious than a setup violation because it cannot be fixed by increasing the clock period.

Clock skew, if done right, can also benefit a circuit. It can be intentionally introduced to decrease the clock period at which the circuit will operate correctly, and/or to increase the setup or hold safety margins. The optimal set of clock delays is determined by a linear program, in which a setup and a hold constraint appears for each logic path. In this linear program, zero clock skew is merely a feasible point.
Clock skew can be minimized by proper routing of clock signal (clock distribution tree) or putting variable delay buffer so that all clock inputs arrive at the same time

3) What is slack?
'Slack' is the amount of time you have that is measured from when an event 'actually happens' and when it 'must happen'.. The term 'actually happens' can also be taken as being a predicted time for when the event will 'actually happen'.
When something 'must happen' can also be called a 'deadline' so another definition of slack would be the time from when something 'actually happens' (call this Tact) until the deadline (call this Tdead).
Slack = Tdead - Tact.
Negative slack implies that the 'actually happen' time is later than the 'deadline' time...in other words it's too late and a timing violation....you have a timing problem that needs some attention.

4) What is glitch? What causes it (explain with waveform)? How to overcome it?

The following figure shows a synchronous alternative to the gated clock using a data path. The flip-flop is clocked at every clock cycle and the data path is controlled by an enable. When the enable is Low, the multiplexer feeds the output of the register back on itself. When the enable is High, new data is fed to the flip-flop and the register changes its state

5) Given only two xor gates one must function as buffer and another as inverter?

Tie one of xor gates input to 1 it will act as inverter.
Tie one of xor gates input to 0 it will act as buffer.

6) What is difference between latch and flipflop?

The main difference between latch and FF is that latches are level sensitive while FF are edge sensitive. They both require the use of clock signal and are used in sequential logic. For a latch, the output tracks the input when the clock signal is high, so as long as the clock is logic 1, the output can change if the input also changes. FF on the other hand, will store the input only when there is a rising/falling edge of the clock.

7) Build a 4:1 mux using only 2:1 mux?

Difference between heap and stack?

The Stack is more or less responsible for keeping track of what's executing in our code (or what's been "called"). The Heap is more or less responsible for keeping track of our objects (our data, well... most of it - we'll get to that later.).
Think of the Stack as a series of boxes stacked one on top of the next. We keep track of what's going on in our application by stacking another box on top every time we call a method (called a Frame). We can only use what's in the top box on the stack. When we're done with the top box (the method is done executing) we throw it away and proceed to use the stuff in the previous box on the top of the stack. The Heap is similar except that its purpose is to hold information (not keep track of execution most of the time) so anything in our Heap can be accessed at any time. With the Heap, there are no constraints as to what can be accessed like in the stack. The Heap is like the heap of clean laundry on our bed that we have not taken the time to put away yet - we can grab what we need quickly. The Stack is like the stack of shoe boxes in the closet where we have to take off

the top one to get to the one underneath it.

9) Difference between mealy and moore state machine?

A) Mealy and Moore models are the basic models of state machines. A state machine which uses only Entry Actions, so that its output depends on the state, is called a Moore model. A state machine which uses only Input Actions, so that the output depends on the state and also on inputs, is called a Mealy model. The models selected will influence a design but there are no general indications as to which model is better. Choice of a model depends on the application, execution means (for instance, hardware systems are usually best realized as Moore models) and personal preferences of a designer or programmer
B) Mealy machine has outputs that depend on the state and input (thus, the FSM has the output written on edges)
Moore machine has outputs that depend on state only (thus, the FSM has the output written in the state itself.
Adv and Disadv
In Mealy as the output variable is a function both input and state, changes of state of the state variables will be delayed with respect to changes of signal level in the input variables, there are possibilities of glitches appearing in the output variables. Moore overcomes glitches as output dependent on only states and not the input signal level.
All of the concepts can be applied to Moore-model state machines because any Moore state machine can be implemented as a Mealy state machine, although the converse is not true.
Moore machine: the outputs are properties of states themselves... which means that you get the output after the machine reaches a particular state, or to get some output your machine has to be taken to a state which provides you the output.The outputs are held until you go to some other state Mealy machine:
Mealy machines give you outputs instantly, that is immediately upon receiving input, but the output is not held after that clock cycle.

10) Difference between onehot and binary encoding?
Common classifications used to describe the state encoding of an FSM are Binary (or highly encoded) and One hot.
A binary-encoded FSM design only requires as many flip-flops as are needed to uniquely encode the number of states in the state machine. The actual number of flip-flops required is equal to the ceiling of the log-base-2 of the number of states in the FSM.
A onehot FSM design requires a flip-flop for each state in the design and only one flip-flop (the flip-flop representing the current or "hot" state) is set at a time in a one hot FSM design. For a state machine with 9- 16 states, a binary FSM only requires 4 flip-flops while a onehot FSM requires a flip-flop for each state in the design
FPGA vendors frequently recommend using a onehot state encoding style because flip-flops are plentiful in an FPGA and the combinational logic required to implement a onehot FSM design is typically smaller than most binary encoding styles. Since FPGA

performance is typically related to the combinational logic size of the FPGA design, onehot FSMs typically run faster than a binary encoded FSM with larger combinational logic blocks

11) What are different ways to synchronize between two clock domains?

12) How to calculate maximum operating frequency?

13) How to find out longest path?

You can find answer to this in timing.ppt of presentations section on this site

14) Draw the state diagram to output a "1" for one cycle if the sequence "0110" shows up (the leading 0s cannot be used in more than one sequence)?

15)How to achieve 180 deree exact phase shift?

Never tell using inverter

a) dcm's an inbuilt resource in most of fpga can be configured to get 180 degree phase shift.

b) Bufgds that is differential signaling buffers which are also inbuilt resource of most of FPGA can be used.

16) What is significance of ras and cas in SDRAM?

SDRAM receives its address command in two address words.

It uses a multiplex scheme to save input pins. The first address word is latched into the DRAM chip with the row address strobe (RAS).

Following the RAS command is the column address strobe (CAS) for latching the second address word.

Shortly after the RAS and CAS strobes, the stored data is valid for reading.

17) Tell some of applications of buffer?

a)They are used to introduce small delays

b)They are used to eliminate cross talk caused due to inter electrode capacitance due to close routing.

c)They are used to support high fanout,eg:bufg

18) Implement an AND gate using mux?

This is the basic question that many interviewers ask. for and gate, give one input as select line,incase if u r giving b as select line, connect one input to logic '0' and other input to a.

19) What will happen if contents of register are shifter left, right?

It is well known that in left shift all bits will be shifted left and LSB will be appended with 0 and in right shift all bits will be shifted right and MSB will be appended with 0 this is a straightforward answer

What is expected is in a left shift value gets Multiplied by 2 eg:consider 0000_1110=14 a left shift will make it 0001_110=28, it the same fashion right shift will Divide the value by 2.

20)Given the following FIFO and rules, how deep does the FIFO need to be to prevent underflow or overflow?

RULES:

1) frequency(clk_A) = frequency(clk_B) / 4

2) period(en_B) = period(clk_A) * 100

3) duty_cycle(en_B) = 25%

Assume clk_B = 100MHz (10ns)

From (1), clk_A = 25MHz (40ns)
From (2), period(en_B) = 40ns * 400 = 4000ns, but we only output for 1000ns,due to (3), so 3000ns of the enable we are doing no output work. Therefore, FIFO size = 3000ns/40ns = 75 entries

21) Design a four-input NAND gate using only two-input NAND gates.
A:Basically, you can tie the inputs of a NAND gate together to get an inverter, so...

22)Difference between Synchronous and Asynchronous reset.?
Synchronous reset logic will synthesize to smaller flip-flops, particularly if the reset is gated with the logic generating the d-input. But in such a case, the combinational logic gate count grows, so the overall gate count savings may not be that significant.
The clock works as a filter for small reset glitches; however, if these glitches occur near the active clock edge, the Flip-flop could go metastable. In some designs, the reset must be generated by a set of internal conditions. A synchronous reset is recommended for these types of designs because it will filter the logic equation glitches between clock.
Disadvantages of synchronous reset:
Problem with synchronous resets is that the synthesis tool cannot easily distinguish the reset signal from any other data signal.
Synchronous resets may need a pulse stretcher to guarantee a reset pulse width wide enough to ensure reset is present during an active edge of the clock[if you have a gated clock to save power, the clock may be disabled coincident with the assertion of reset. Only an asynchronous reset will work in this situation, as the reset might be removed prior to the resumption of the clock.
Designs that are pushing the limit for data path timing, can not afford to have added gates and additional net delays in the data path due to logic inserted to handle synchronous resets.
Asynchronous reset :
The biggest problem with asynchronous resets is the reset release, also called reset removal. Using an asynchronous reset, the designer is guaranteed not to have the reset added to the data path. Another advantage favoring asynchronous resets is that the circuit can be reset with or without a clock present.
Disadvantages of asynchronous reset: ensure that the release of the reset can occur within one clock period. if the release of the reset occurred on or near a clock edge such that the flip-flops went metastable.

23) Why are most interrupts active low?
This answers why most signals are active low
If you consider the transistor level of a module, active low means the capacitor in the output terminal gets charged or discharged based on low to high and high to low transition respectively. when it goes from high to low it depends on the pull down resistor that pulls it down and it is relatively easy for the output capacitance to discharge rather

than charging. hence people prefer using active low signals.

24)Give two ways of converting a two input NAND gate to an inverter?

(a) short the 2 inputs of the nand gate and apply the single input to it.
(b) Connect the output to one of the input and the other to the input signal.

25) What are set up time & hold time constraints? What do they signify? Which one is critical for estimating maximum clock frequency of a circuit?

set up time: - the amount of time the data should be stable before the application of the clock signal, where as the hold time is the amount of time the data should be stable after the application of the clock. Setup time signifies maximum delay constraints; hold time is for minimum delay constraints. Setup time is critical for establishing the maximum clock frequency.

26) Differences between D-Latch and D flip-flop?
D-latch is level sensitive where as flip-flop is edge sensitive. Flip-flops are made up of latches.

27) What is a multiplexer?
Is combinational circuit that selects binary information from one of many input lines and directs it to a single output line. (2n =>n).

28)How can you convert an SR Flip-flop to a JK Flip-flop?
By giving the feed back we can convert, i.e !Q=>S and Q=>R.Hence the S and R inputs will act as J and K respectively.

29)How can you convert the JK Flip-flop to a D Flip-flop?
By connecting the J input to the K through the inverter.

30)What is Race-around problem?How can you rectify it?
The clock pulse that remains in the 1 state while both J and K are equal to 1 will cause the output to complement again and repeat complementing until the pulse goes back to 0, this is called the race around problem.To avoid this undesirable operation, the clock pulse must have a time duration that is shorter than the propagation delay time of the F-F, this is restrictive so the alternative is master-slave or edge-triggered construction.

31)How do you detect if two 8-bit signals are same?
XOR each bits of A with B (for e.g. A[0] xor B[0]) and so on.the o/p of 8 xor gates are then given as i/p to an 8-i/p nor gate. if o/p is 1 then A=B.

32)7 bit ring counter's initial state is 0100010. After how many clock cycles will it return to the initial state?
6 cycles

33) Convert D-FF into divide by 2. (not latch) What is the max clock frequency the circuit can handle, given the following information?
T_setup= 6nS T_hold = 2nS T_propagation = 10nS
Circuit: Connect Qbar to D and apply the clk at clk of DFF and take the O/P at Q. It gives freq/2. Max. Freq of operation: 1/ (propagation delay+setup time) = 1/16ns = 62.5 MHz

34)Guys this is the basic question asked most frequently. Design all the basic gates(NOT,AND,OR,NAND,NOR,XOR,XNOR) using 2:1 Multiplexer?
Using 2:1 Mux, (2 inputs, 1 output and a select line)
(a) NOT
Give the input at the select line and connect I0 to 1 & I1 to 0. So if A is 1, we will get I1 that is 0 at the O/P.
(b) AND
Give input A at the select line and 0 to I0 and B to I1. O/p is A & B
(c) OR
Give input A at the select line and 1 to I1 and B to I0. O/p will be A | B
(d) NAND
AND + NOT implementations together
(e) NOR
OR + NOT implementations together
(f) XOR
A at the select line B at I0 and ~B at I1. ~B can be obtained from (a) (g) XNOR
A at the select line B at I1 and ~B at I0

35)N number of XNOR gates are connected in series such that the N inputs (A0,A1,A2.....) are given in the following way: A0 & A1 to first XNOR gate and A2 & O/P of First XNOR to second XNOR gate and so on..... Nth XNOR gates output is final output. How does this circuit work? Explain in detail?

If N=Odd, the circuit acts as even parity detector, ie the output will 1 if there are even number of 1's in the N input...This could also be called as odd parity generator since with this additional 1 as output the total number of 1's will be ODD.
If N=Even, just the opposite, it will be Odd parity detector or Even Parity Generator.

36)An assembly line has 3 fail safe sensors and one emergency shutdown switch.The line should keep moving unless any of the following conditions arise:
(i) If the emergency switch is pressed
(ii) If the senor1 and sensor2 are activated at the same time.
(iii) If sensor 2 and sensor3 are activated at the same time.
(iv) If all the sensors are activated at the same time
Suppose a combinational circuit for above case is to be implemented only with NAND Gates. How many minimum number of 2 input NAND gates are required?
No of 2-input NAND Gates required = 6 You can try the whole implementation.

37)Design a circuit that calculates the square of a number? It should not use any multiplier circuits. It should use Multiplexers and other logic?
This is interesting....
1^2=0+1=1
2^2=1+3=4
3^2=4+5=9
4^2=9+7=16
5^2=16+9=25
and so on
See a pattern yet?To get the next square, all you have to do is add the next odd number to the previous square that you found.See how 1,3,5,7 and finally 9 are added.Wouldn't this be a possible solution to your question since it only will use a counter,multiplexer and a couple of adders?It seems it would take n clock cycles to calculate square of n.

3  How will you implement a Full subtractor from a Full adder?
all the bits of subtrahend should be connected to the xor gate. Other input to the xor being one.The input carry bit to the full adder should be made 1. Then the full adder works like a full subtractor

39)A very good interview question... What is difference between setup and hold time.
The interviewer was looking for one specific reason , and its really a good answer too..The hint is hold time doesn't depend on clock, why is it so...?
Setup violations are related to two edges of clock, i mean you can vary the clock

frequency to correct setup violation. But for hold time, you are only concerned with one edge and does not basically depend on clock frequency.

40)In a 3-bit Johnson's counter what are the unused states?
2(power n)-2n is the one used to find the unused states in johnson counter.
So for a 3-bit counter it is 8-6=2.Unused states=2. the two unused states are 010 and 101

41)The question is to design minimal hardware system, which encrypts 8-bit parallel data. A synchronized clock is provided to this system as well. The output encrypted data should be at the same rate as the input data but no necessarily with the same phase.

The encryption system is centered around a memory device that perform a LUT (Look-Up Table) conversion. This memory functionality can be achieved by using a PROM, EPROM, FLASH and etc. The device contains an encryption code, which may be burned into the device with an external programmer. In encryption operation, the data_in is an address pointer into a memory cell and the combinatorial logic generates the control signals. This creates a read access from the memory. Then the memory device goes to the appropriate address and outputs the associate data. This data represent the data_in after encryption. 41) What is an LFSR .List a few of its industry applications.?

LFSR is a linear feedback shift register where the input bit is driven by a linear function of the overall shift register value. coming to industrial applications, as far as I know, it is used for encryption and decryption and in BIST(built-in-self-test) based applications..

42)what is false path?how it determine in ckt? what the effect of false path in ckt?

By timing all the paths in the circuit the timing analyzer can determine all the critical paths in the circuit. However, the circuit may have false paths, which are the paths in the circuit which are never exercised during normal circuit operation for any set of inputs. An example of a false path is shown in figure below. The path going from the input A of the first MUX through the combinational logic out through the B input of the second MUS is a false path. This path can never be activated since if the A input of the first MUX is activated, then Sel line will also select the A input of the second MUX.
STA (Static Timing Analysis) tools are able to identify simple false paths; however they are not able to identify all the false paths and sometimes report false paths as critical paths. Removal of false paths makes circuit testable and its timing performance predictable (sometimes faster)

43)Consider two similar processors, one with a clock skew of 100ps and other with a clock skew of 50ps. Which one is likely to have more power? Why?
Clock skew of 50ps is more likely to have clock power. This is because it is likely that

low-skew processor has better designed clock tree with more powerful and number of buffers and overheads to make skew better.

44)What are multi-cycle paths?
Multi-cycle paths are paths between registers that take more than one clock cycle to become stable.
For ex. Analyzing the design shown in fig below shows that the output SIN/COS requires 4 clock-cycles after the input ANGLE is latched in. This means that the combinatorial block (the Unrolled Cordic) can take up to 4 clock periods (25MHz) to propagate its result. Place and Route tools are capable of fixing multi-cycle paths problem.

45)You have two counters counting upto 16, built from negedge DFF , First circuit is synchronous and second is "ripple" (cascading), Which circuit has a less propagation delay? Why?
The synchronous counter will have lesser delay as the input to each flop is readily available before the clock edge. Whereas the cascade counter will take long time as the output of one flop is used as clock to the other. So the delay will be propagating. For Eg: 16 state counter = 4 bit counter = 4 Flip flops Let 10ns be the delay of each flop The worst case delay of ripple counter = 10 * 4 = 40ns The delay of synchronous counter = 10ns only.(Delay of 1 flop)

46) what is difference between RAM and FIFO?
FIFO does not have address lines
Ram is used for storage purpose where as fifo is used for synchronization purpose i.e. when two peripherals are working in different clock domains then we will go for fifo.

47)The circle can rotate clockwise and back. Use minimum hardware to build a circuit to indicate the direction of rotating.?

2 sensors are required to find out the direction of rotating. They are placed like at the drawing. One of them is connected to the data input of D flip-flop,and a second one - to the clock input. If the circle rotates the way clock sensor sees the light first while D input (second sensor) is zero - the output of the flip-flop equals zero, and if D input sensor "fires" first - the output of the flip-flop becomes high.

4 Draw timing diagrams for following circuit.?

49)Implement the following circuits:
(a) 3 input NAND gate using min no of 2 input NAND Gates

(b) 3 input NOR gate using min no of 2 inpur NOR Gates
(c) 3 input XNOR gate using min no of 2 inpur XNOR Gates
Assuming 3 inputs A,B,C?

3 input NAND:
Connect :
a) A and B to the first NAND gate
b) Output of first Nand gate is given to the two inputs of the second NAND gate (this basically realizes the inverter functionality)
c) Output of second NAND gate is given to the input of the third NAND gate, whose other input is C
((A NAND B) NAND (A NAND B)) NAND C Thus, can be implemented using '3' 2-input NAND gates. I guess this is the minimum number of gates that need to be used.
3 input NOR:
Same as above just interchange NAND with NOR ((A NOR B) NOR (A NOR B)) NOR C
3 input XNOR:
Same as above except the inputs for the second XNOR gate, Output of the first XNOR gate is one of the inputs and connect the second input to ground or logical '0'
((A XNOR B) XNOR 0)) XNOR C

50) Is it possible to reduce clock skew to zero? Explain your answer ?
Even though there are clock layout strategies (H-tree) that can in theory reduce clock skew to zero by having the same path length from each flip-flop from the pll, process variations in R and C across the chip will cause clock skew as well as a pure H-Tree scheme is not practical (consumes too much area).

51)Design a FSM (Finite State Machine) to detect a sequence 10110?

52)Convert D-FF into divide by 2. (not latch)? What is the max clock frequency of the circuit , given the following information?
T_setup= 6nS
T_hold = 2nS
T_propagation = 10nS
Circuit:
Connect Qbar to D and apply the clk at clk of DFF and take the O/P at Q. It gives freq/2.
Max. Freq of operation:
1/ (propagation delay+setup time) = 1/16ns = 62.5 MHz

53)Give the circuit to extend the falling edge of the input by 2 clock pulses?The waveforms are shown in the following figure.

54) For the Circuit Shown below, What is the Maximum Frequency of Operation?Are there any hold time violations for FF2? If yes, how do you modify the circuit to avoid them?

The minumum time period = 3+2+(1+1+1) = 8ns Maximum Frequency = 1/8n= 125MHz.
And there is a hold time violation in the circuit,because of feedback, if you observe, tcq2+AND gate delay is less than thold2,To avoid this we need to use even number of inverters(buffers). Here we need to use 2 inverters each with a delay of 1ns. then the hold time value exactly meets.

55)Design a D-latch using (a) using 2:1 Mux (b) from S-R Latch ?
56)How to implement a Master Slave flip flop using a 2 to 1 mux?

57)how many 2 input xor's are needed to inplement 16 input parity generator ?
It is always n-1 Where n is number of inputs.So 16 input parity generator will require 15 two input xor's .

58)Design a circuit for finding the 9's compliment of a BCD number using 4-bit binary adder and some external logic gates?
9's compliment is nothing but subracting the given no from 9.So using a 4 bit binary adder we can just subract the given binary no from 1001(i.e. 9).Here we can use the 2's compliment method addition.

59) what is Difference between writeback and write through cache?
A caching method in which modifications to data in the cache aren't copied to the cache source until absolutely necessary. Write-back caching is available on many microprocessors , including all Intel processors since the 80486. With these microprocessors, data modifications to data stored in the L1 cache aren't copied to main memory until absolutely necessary. In contrast, a write-through cache performs all write operations in parallel -- data is written to main memory and the L1 cache simultaneously. Write-back caching yields somewhat better performance than write-through caching because it reduces the number of write operations to main memory. With this performance improvement comes a slight risk that data may be lost if the system crashes. A write-back cache is also called a copy-back cache.
60)Difference between Synchronous,Asynchronous & Isynchronous communication?
Sending data encoded into your signal requires that the sender and receiver are both using the same enconding/decoding method, and know where to look in the signal to find data. Asynchronous systems do not send separate information to indicate the encoding or clocking information. The receiver must decide the clocking of the signal on it's own. This means that the receiver must decide where to look in the signal stream to find ones and zeroes, and decide for itself where each individual bit stops and starts. This information is not in the data in the signal sent from transmitting unit.
Synchronous systems negotiate the connection at the data-link level before

communication begins. Basic synchronous systems will synchronize two clocks before transmission, and reset their numeric counters for errors etc. More advanced systems may negotiate things like error correction and compression.
Time-dependent. it refers to processes where data must be delivered within certain time constraints. For example, Multimedia stream require an isochronous transport mechanism to ensure that data is delivered as fast as it is displayed and to ensure that the audio is synchronized with the video.
61) What are different ways Multiply & Divide?

Set quotient to zero

Repeat while dividend is greater than or equal to divisor

Subtract divisor from dividend

Add 1 to quotient

End of repeat block

quotient is correct, dividend is remainder

STOP

Binary Division by Shift and Subtract
Basically the reverse of the mutliply by shift and add.

Set quotient to 0

Align leftmost digits in dividend and divisor

Repeat

If that portion of the dividend above the divisor is greater than or equal to the divisor

Then subtract divisor from that portion of the dividend and

Concatentate 1 to the right hand end of the quotient

Else concatentate 0 to the right hand end of the quotient

Shift the divisor one place right

Until dividend is less than the divisor

quotient is correct, dividend is remainder

STOP

Binary Multiply - Repeated Shift and Add
Repeated shift and add - starting with a result of 0, shift the second multiplicand to correspond with each 1 in the first multiplicand and add to the result. Shifting each position left is equivalent to multiplying by 2, just as in decimal representation a shift left is equivalent to multiplying by 10.

Set result to 0

Repeat

Shift 2nd multiplicand left until rightmost digit is lined up with leftmost 1 in first multiplicand

Add 2nd multiplicand in that position to result

Remove that 1 from 1st multiplicand

Until 1st multiplicand is zero

Result is correct

STOP

62)What is a SoC (System On Chip), ASIC, "full custom chip", and an FPGA?

There are no precise definitions. Here is my sense of it all. First, 15 years ago, people were unclear on exactly what VLSI meant. Was it 50000 gates? 100000 gates? was is just anything bigger than LSI? My professor simply told me that; VLSI is a level of complexity and integration in a chip that demands Electronic Design Automation tools in

order to succeed. In other words, big enough that manually drawing lots of little blue, red and green lines is too much for a human to reasonably do. I think that, likewise, SoC is that level of integration onto a chip that demands more expertise beyond traditional skills of electronics. In other words, pulling off a SoC demands Hardware, Software, and Systems Engineering talent. So, trivially, SoCs aggressively combine HW/SW on a single chip. Maybe more pragmatically, SoC just means that ASIC and Software folks are learning a little bit more about each other's techniques and tools than they did before. Two other interpretations of SoC are 1) a chip that integrates various IP (Intellectual Property) blocks on it and is thus highly centered with issues like Reuse, and 2) a chip integrating multiple classes of electronic circuitry such as Digital CMOS, mixed-signal digital and analog (e.g. sensors, modulators, A/Ds), DRAM memory, high voltage power, etc.

ASIC stands for "Application Specific Integrated Circuit". A chip designed for a specific application. Usually, I think people associate ASICs with the Standard Cell design methodology. Standard Cell design and the typical "ASIC flow" usually means that designers are using Hardware Description Languages, Synthesis and a library of primitive cells (e.g. libraries containing AND, NAND, OR, NOR, NOT, FLIP-FLOP, LATCH, ADDER, BUFFER, PAD cells that are wired together (real libraries are not this simple, but you get the idea..). Design usually is NOT done at a transistor level. There is a high reliance on automated tools because the assumption is that the chip is being made for a SPECIFIC APPLICATION where time is of the essence. But, the chip is manufactured from scratch in that no pre-made circuitry is being programmed or reused. ASIC designer may, or may not, even be aware of the locations of various pieces of circuitry on the chip since the tools do much of the construction, placement and wiring of all the little pieces.

Full Custom, in contrast to ASIC (or Standard Cell), means that every geometric feature going onto the chip being designed (think of those pretty chip pictures we have all seen) is controlled, more or less, by the human design. Automated tools are certainly used to wire up different parts of the circuit and maybe even manipulate (repeat, rotate, etc.) sections of the chip. But, the human designer is actively engaged with the physical features of the circuitry. Higher human crafting and less reliance on standard cells takes more time and implies higher NRE costs, but lowers RE costs for standard parts like memories, processors, uarts, etc.

FPGAs, or Field Programmable Gate Arrays are completely designed chips that designers load a programming pattern into to achieve a specific digital function. A bit pattern (almost like a software program) is loaded into the already manufactured device which essentially interconnects lots of available gates to meet the designers purposes. FPGAs are sometimes thought of as a "Sea of Gates" where the designer specifies how they are connected. FPGA designers often use many of the same tools that ASIC designers use, even though the FPGA is inherently more flexible. All these things can be intermixed in hybrid sorts of ways. For example, FPGAs are now available that have **microprocessor** embedded within them which were designed in a full custom manner, all of which now demands "SoC" types of HW/SW integration skills from the designer.

63)What is "Scan" ?

§Scan Insertion and ATPG helps test ASICs (e.g. chips) during manufacture. If you know what JTAG boundary scan is, then Scan is the same idea except that it is done inside the chip instead of on the entire board. Scan tests for defects in the chip's circuitry after it is manufactured (e.g. Scan does not help you test whether your Design functions as intended). ASIC designers usually implement the scan themselves and occurs just after synthesis. ATPG (Automated Test Pattern Generation) refers to the creation of "Test Vectors" that the Scan circuitry enables to be introduced into the chip. Here's a brief summary:

· Scan Insertion is done by a tool and results in all (or most) of your design's flip-flops to be replaced by special "Scan Flip-flops". Scan flops have additional inputs/outputs that allow them to be configured into a "chain" (e.g. a big shift register) when the chip is put into a test mode.

· The Scan flip-flops are connected up into a chain (perhaps multiple chains)

· The ATPG tool, which knows about the scan chain you've created, generates a series of test vectors.

· The ATPG test vectors include both "Stimulus" and "Expected" bit patterns. These bit vectors are shifted into the chip on the scan chains, and the chips reaction to the stimulus is shifted back out again.

· The ATE (Automated Test Equipment) at the chip factory can put the chip into the scan test mode, and apply the test vectors. If any vectors do not match, then the chip is defective and it is thrown away.

· Scan/ATPG tools will strive to maximize the "coverage" of the ATPG vectors. In other words, given some measure of the total number of nodes in the chip that could be faulty (shorted, grounded, "stuck at 1", "stuck at 0"), what percentage of them can be detected with the ATPG vectors? Scan is a good technology and can achive high coverage in the 90% range.

· Scan testing does not solve all test problems. Scan testing typically does not test memories (no flip-flops!), needs a gate-level netlist to work with, and can take a long time to run on the ATE.

· FPGA designers may be unfamiliar with scan since FPGA testing has already been done by the FPGA manufacturer. ASIC designers do not have this luxury and must handle all the manufacturing test details themselves.

· Check out the Synopsys WWW site for more info.

1) Write a verilog code to swap contents of two registers with and without a temporary register?

With temp reg ;

```
always @ (posedge clock)
begin
temp=b;
b=a;
a=temp;
end
```

Without temp reg;

```
always @ (posedge clock)
begin
```
a <= b; b <= a; end 2) Difference between blocking and non-blocking?(Verilog interview questions that is most commonly asked)

The Verilog language has two forms of the procedural assignment statement: blocking and non-blocking. The two are distinguished by the = and <= assignment operators. The blocking assignment statement (= operator) acts much like in traditional programming languages. The whole statement is done before control passes on to the next statement. The non-blocking (<= operator) evaluates all the right-hand sides for the current time unit and assigns the left-hand sides at the end of the time unit. For example, the following Verilog program // testing blocking and non-blocking assignment module blocking; reg [0] A, B; initial begin: init1 A = 3; #1 A = A + 1; // blocking procedural assignment B = A + 1; $display("Blocking: A= %b B= %b", A, B); A = 3; #1 A <= A + 1; // non-blocking procedural assignment
B <= A + 1; #1 $display("Non-blocking: A= %b B= %b", A, B); end endmodule produces the following output: Blocking: A= 00000100 B= 00000101 Non-blocking: A= 00000100 B= 00000100 The effect is for all the non-blocking assignments to use the old values of the variables at the beginning of the current time unit and to assign the registers new values at the end of the current time unit. This reflects how register transfers occur in some hardware systems. blocking procedural assignment is used for combinational logic and non-blocking procedural assignment for sequential

Click to view more

Difference between task and function?

Function:
A function is unable to enable a task however functions can enable other functions.
A function will carry out its required duty in zero simulation time. (The program time will not be incremented during the function routine)
Within a function, no event, delay or timing control statements are permitted
In the invocation of a function their must be at least one argument to be passed.
Functions will only return a single value and can not use either output or inout statements.

Tasks:
Tasks are capable of enabling a function as well as enabling other versions of a Task
Tasks also run with a zero simulation however they can if required be executed in a non zero simulation time.
Tasks are allowed to contain any of these statements.
A task is allowed to use zero or more arguments which are of type output, input or inout.
A Task is unable to return a value but has the facility to pass multiple values via the output and inout statements .

4) Difference between inter statement and intra statement delay?

```
//define register variables
reg a, b, c;

//intra assignment delays
initial
begin
a = 0; c = 0;
b = #5 a + c; //Take value of a and c at the time=0, evaluate
//a + c and then wait 5 time units to assign value
//to b.
end

//Equivalent method with temporary variables and regular delay control
initial
begin
a = 0; c = 0;
temp_ac = a + c;
#5 b = temp_ac; //Take value of a + c at the current time and
//store it in a temporary variable. Even though a and c
//might change between 0 and 5,
//the value assigned to b at time 5 is unaffected.
end
```

5) What is delta simulation time?

6) Difference between $monitor,$display & $strobe?

These commands have the same syntax, and display text on the screen during simulation. They are much less convenient than waveform display tools like cwaves?. $display and $strobe display once every time they are executed, whereas $monitor displays every time one of its parameters changes.
The difference between $display and $strobe is that $strobe displays the parameters at the very end of the current simulation time unit rather than exactly where it is executed. The format string is like that in C/C++, and may contain format characters. Format characters include %d (decimal), %h (hexadecimal), %b (binary), %c (character), %s (string) and %t (time), %m (hierarchy level). %5d, %5b etc. would give exactly 5 spaces for the number instead of the space needed. Append b, h, o to the task name to change default format to binary, octal or hexadecimal.
Syntax:
$display ("format_string", par_1, par_2, ...);
$strobe ("format_string", par_1, par_2, ...);
$monitor ("format_string", par_1, par_2, ...);

7) What is difference between Verilog full case and parallel case?

A "full" case statement is a case statement in which all possible case-expression binary patterns can be matched to a case item or to a case default. If a case statement does not include a case default and if it is possible to find a binary case expression that does not match any of the defined case items, the case statement is not "full."
A "parallel" case statement is a case statement in which it is only possible to match a case expression to one and only one case item. If it is possible to find a case expression that would match more than one case item, the matching case items are called "overlapping" case items and the case statement is not "parallel."

 What is meant by inferring latches,how to avoid it?

Consider the following :
always @(s1 or s0 or i0 or i1 or i2 or i3)
case ({s1, s0})
2'd0 : out = i0;
2'd1 : out = i1;
2'd2 : out = i2;
endcase

in a case statement if all the possible combinations are not compared and default is also not specified like in example above a latch will be inferred ,a latch is inferred because to reproduce the previous value when unknown branch is specified.

For example in above case if {s1,s0}=3 , the previous stored value is reproduced for this storing a latch is inferred.
The same may be observed in IF statement in case an ELSE IF is not specified.
To avoid inferring latches make sure that all the cases are mentioned if not default condition is provided.

9) Tell me how blocking and non blocking statements get executed?

Execution of blocking assignments can be viewed as a one-step process:
1. Evaluate the RHS (right-hand side equation) and update the LHS (left-hand side expression) of the blocking assignment without interruption from any other Verilog statement. A blocking assignment "blocks" trailing assignments in the same always block from occurring until after the current assignment has been completed

Execution of nonblocking assignments can be viewed as a two-step process:
1. Evaluate the RHS of nonblocking statements at the beginning of the time step. 2. Update the LHS of nonblocking statements at the end of the time step.

10) Variable and signal which will be Updated first?

Signals

11) What is sensitivity list?

The sensitivity list indicates that when a change occurs to any one of elements in the list change, begin…end statement inside that always block will get executed.

12) In a pure combinational circuit is it necessary to mention all the inputs in sensitivity disk? if yes, why?

Yes in a pure combinational circuit is it necessary to mention all the inputs in sensitivity disk other wise it will result in pre and post synthesis mismatch.

13) Tell me structure of Verilog code you follow?

A good template for your Verilog file is shown below.

```
// timescale directive tells the simulator the base units and precision of the simulation
`timescale 1 ns / 10 ps
module name (input and outputs);
// parameter declarations
parameter parameter_name = parameter value;
// Input output declarations
input in1;
input in2; // single bit inputs
output [msb] out; // a bus output
```

```
// internal signal register type declaration - register types (only assigned within always statements). reg register variable 1;
reg [msb] register variable 2;
// internal signal. net type declaration - (only assigned outside always statements) wire net variable 1;
// hierarchy - instantiating another module
reference name instance name (
.pin1 (net1),
.pin2 (net2),
.
.pinn (netn)
);
// synchronous procedures
always @ (posedge clock)
begin
.
end
// combinatinal procedures
always @ (signal1 or signal2 or signal3)
begin
.
end
assign net variable = combinational logic;
endmodule
```

14) Difference between Verilog and vhdl?

Compilation
VHDL. Multiple design-units (entity/architecture pairs), that reside in the same system file, may be separately compiled if so desired. However, it is good design practice to keep each design unit in it's own system file in which case separate compilation should not be an issue.

Verilog. The Verilog language is still rooted in it's native interpretative mode. Compilation is a means of speeding up simulation, but has not changed the original nature of the language. As a result care must be taken with both the compilation order of code written in a single file and the compilation order of multiple files. Simulation results can change by simply changing the order of compilation.

Data types
VHDL. A multitude of language or user defined data types can be used. This may mean dedicated conversion functions are needed to convert objects from one type to another. The choice of which data types to use should be considered wisely, especially enumerated (abstract) data types. This will make models easier to write, clearer to read

and avoid unnecessary conversion functions that can clutter the code. VHDL may be preferred because it allows a multitude of language or user defined data types to be used.

Verilog. Compared to VHDL, Verilog data types a re very simple, easy to use and very much geared towards modeling hardware structure as opposed to abstract hardware modeling. Unlike VHDL, all data types used in a Verilog model are defined by the Verilog language and not by the user. There are net data types, for example wire, and a register data type called reg. A model with a signal whose type is one of the net data types has a corresponding electrical wire in the implied modeled circuit. Objects, that is signals, of type reg hold their value over simulation delta cycles and should not be confused with the modeling of a hardware register. Verilog may be preferred because of it's simplicity.

Design reusability
VHDL. Procedures and functions may be placed in a package so that they are avail able to any design-unit that wishes to use them.

Verilog. There is no concept of packages in Verilog. Functions and procedures used within a model must be defined in the module. To make functions and procedures generally accessible from different module statements the functions and procedures must be placed in a separate system file and included using the `include compiler directive.

15) What are different styles of Verilog coding I mean gate-level,continuous level and others explain in detail?

16) Can you tell me some of system tasks and their purpose?

$display, $displayb, $displayh, $displayo, $write, $writeb, $writeh, $writeo.
The most useful of these is $display.This can be used for displaying strings, expression or values of variables.
Here are some examples of usage.
$display("Hello oni");
--- output: Hello oni
$display($time) // current simulation time.
--- output: 460
counter = 4'b10;
$display(" The count is %b", counter);
--- output: The count is 0010
$reset resets the simulation back to time 0; $stop halts the simulator and puts it in interactive mode where the
user can enter commands; $finish exits the simulator back to the operating system

17) Can you list out some of enhancements in Verilog 2001?

In earlier version of Verilog ,we use 'or' to specify more than one element in sensitivity

list . In Verilog 2001, we can use comma as shown in the example below.
// Verilog 2k example for usage of comma
always @ (i1,i2,i3,i4)

Verilog 2001 allows us to use star in sensitive list instead of listing all the variables in RHS of combo logics . This removes typo mistakes and thus avoids simulation and synthesis mismatches,
Verilog 2001 allows port direction and data type in the port list of modules as shown in the example below
module memory (
input r,
input wr,
input [7] data_in,
input [3] addr,
output [7] data_out
);

18)Write a Verilog code for synchronous and asynchronous reset?

Synchronous reset, synchronous means clock dependent so reset must not be present in sensitivity disk eg:
always @ (posedge clk)

begin if (reset)
. . . end
Asynchronous means clock independent so reset must be present in sensitivity list.
Eg
Always @(posedge clock or posedge reset)
begin
if (reset)
. . . end

19) What is pli?why is it used?

Programming Language Interface (PLI) of Verilog HDL is a mechanism to interface Verilog programs with programs written in C language. It also provides mechanism to access internal databases of the simulator from the C program.
PLI is used for implementing system calls which would have been hard to do otherwise (or impossible) using Verilog syntax. Or, in other words, you can take advantage of both the paradigms - parallel and hardware related features of Verilog and sequential flow of C - using PLI.

20) There is a triangle and on it there are 3 ants one on each corner and are free to move along sides of triangle what is probability that they will collide?

Ants can move only along edges of triangle in either of direction, let's say one is represented by 1 and another by 0, since there are 3 sides eight combinations are possible, when all ants are going in same direction they won't collide that is 111 or 000 so probability of collision is 2/8=1/4

21) Tell me about file I/O?

21)What is difference between freeze deposit and force?

$deposit(variable, value);
This system task sets a Verilog register or net to the specified value. variable is the register or net to be changed; value is the new value for the register or net. The value remains until there is a subsequent driver transaction or another $deposit task for the same register or net. This system task operates identically to the ModelSim
force -deposit command.

The force command has -freeze, -drive, and -deposit options. When none of these is specified, then -freeze is assumed for unresolved signals and -drive is assumed for resolved
signals. This is designed to provide compatibility with force files. But if you prefer -freeze
as the default for both resolved and unresolved signals.

Verilog interview Questions
22)Will case infer priority register if yes how give an example?

yes case can infer priority register depending on coding style
reg r;
// Priority encoded mux,
always @ (a or b or c or select2)
begin
r = c;
case (select2)
2'b00: r = a;
2'b01: r = b;
endcase
end

Verilog interview Questions
23)Casex,z difference,which is preferable,why?

CASEZ :
Special version of the case statement which uses a Z logic value to represent don't-care bits. CASEX :
Special version of the case statement which uses Z or X logic values to represent don't-care bits.

CASEZ should be used for case statements with wildcard don't cares, otherwise use of CASE is required; CASEX should never be used.
This is because:
Don't cares are not allowed in the "case" statement. Therefore casex or casez are required. Casex will automatically match any x or z with anything in the case statement. Casez will only match z's -- x's require an absolute match.

Verilog interview Questions
24)Given the following Verilog code, what value of "a" is displayed?

always @(clk) begin
a = 0;
a <= 1; $display(a); end This is a tricky one! Verilog scheduling semantics basically imply a four-level deep queue for the current simulation time: 1: Active Events (blocking statements) 2: Inactive Events (#0 delays, etc) 3: Non-Blocking Assign Updates (non-blocking statements) 4: Monitor Events ($display, $monitor, etc). Since the "a = 0" is an active event, it is scheduled into the 1st "queue". The "a <= 1" is a non-blocking event, so it's placed into the 3rd queue. Finally, the display statement is placed into the 4th queue. Only events in the active queue are completed this sim cycle, so the "a = 0" happens, and then the display shows a = 0. If we were to look at the value of a in the next sim cycle, it would show 1. 25) What is the difference between the following two lines of Verilog code?
#5 a = b;
a = #5 b;

#5 a = b; Wait five time units before doing the action for "a = b;".
a = #5 b; The value of b is calculated and stored in an internal temp register,After five time units, assign this stored value to a.

26)What is the difference between:

c = foo ? a : b;
and
if (foo) c = a;
else c = b;

The ? merges answers if the condition is "x", so for instance if foo = 1'bx, a = 'b10, and b = 'b11, you'd get c = 'b1x. On the other hand, if treats Xs or Zs as FALSE, so you'd always get c = b.

27)What are Intertial and Transport Delays ??

28)What does `timescale 1 ns/ 1 ps signify in a verilog code?

'timescale directive is a compiler directive.It is used to measure simulation time or delay

time. Usage : `timescale

34)what is verilog case (1) ?

```
wire [3] x;
always @(...) begin
case (1'b1)
x[0]: SOMETHING1;
x[1]: SOMETHING2;
x[2]: SOMETHING3;
x[3]: SOMETHING4;
endcase
end
```

The case statement walks down the list of cases and executes the first one that matches. So here, if the lowest 1-bit of x is bit 2, then something3 is the statement that will get executed (or selected by the logic).

35) Why is it that "if (2'b01 & 2'b10)..." doesn't run the true case?

This is a popular coding error. You used the bit wise AND operator (&) where you meant to use the logical AND operator (&&).

36)What are Different types of Verilog Simulators ?

There are mainly two types of simulators available.

Event Driven
Cycle Based

Event-based Simulator:

This Digital Logic Simulation method sacrifices performance for rich functionality: every active signal is calculated for every device it propagate .....