MAHARASHTRA STATE BOARD OF TECHNICAL EDUCATION
(Autonomous)
(ISO/IEC - 27001 - 2005 Certified)

SUMMER– 16 EXAMINATION

Subject Code: 17659 **Model Answer**

Important Instructions to examiners:

1) The answers should be examined by key words and not as word-to-word as given in the model answer scheme.
2) The model answer and the answer written by candidate may vary but the examiner may try to assess the understanding level of the candidate.
3) The language errors such as grammatical, spelling errors should not be given moreimportance (Not applicable for subject English and Communication Skills.
4) While assessing figures, examiner may give credit for principal components indicated in thefigure. The figures drawn by candidate and model answer may vary. The examiner may give credit for any equivalent figure drawn.
5) Credits may be given step wise for numerical problems. In some cases, the assumed constant values may vary and there may be some difference in the candidate's answers and model answer.
6) In case of some questions credit may be given by judgment on part of examiner of relevant answer based on candidate's understanding.
7) For programming language papers, credit may be given to any other program based on equivalent concept.

Q.1. Attempt any TEN of the following: **20M**

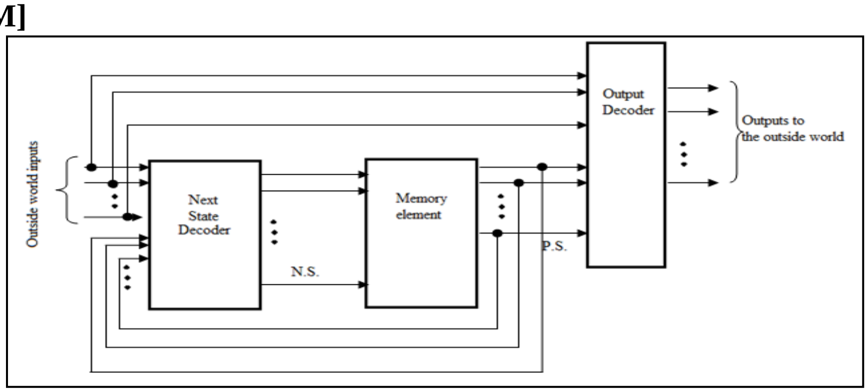
(a) Define the term 'Noise Margins'.

Ans: [Define: 2 M]

It is a measure of noise immunity of a gate or circuit (noise immunity is the ability of a gate or circuit to tolerate any noise present in a signal without performing a wrong operation).

(b) Draw the block diagram of Mealy Machine.

Ans: [Diagram: 2 M]



(c) Define 'skew' w.r.t. sequential logic.

Ans: [Define: 2 M]

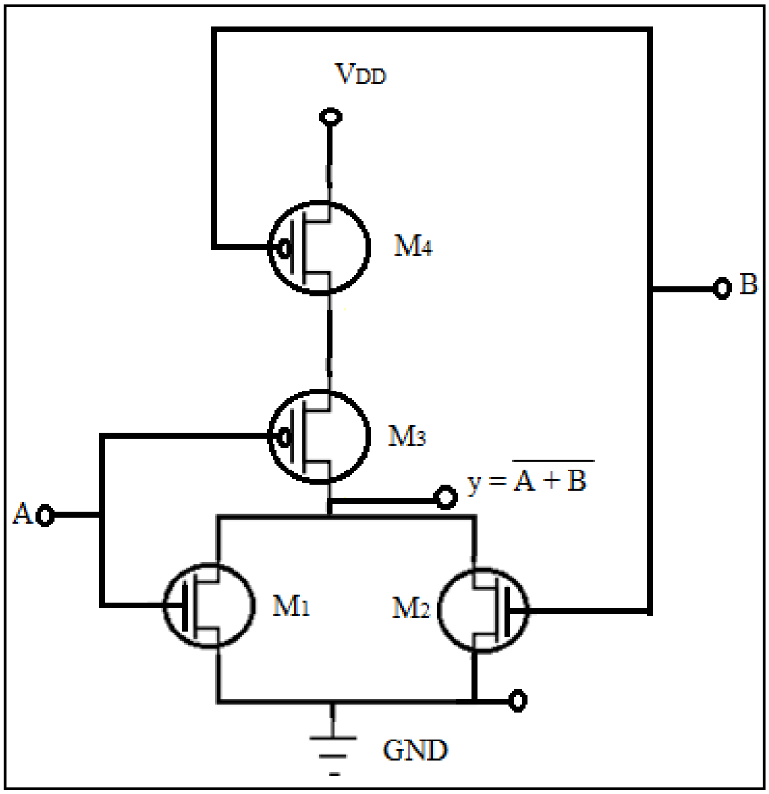
The clock signal, which is said to be applied simultaneously to all the flip-flops, may cause a minute delay changes due to some variation in the wiring between the components. Due to this, it may happen that the clock signal may arrive at the clock inputs of different flip-flops at different times. This delay is termed as skew. OR The difference in the clock arrival time is call clock skew.

(d) State the applications of finite state machines.
Ans: [Applications: 2 M]

(Any other correct applications shall be given M)

- Vending Machines
- Traffic Lights
- Video Games
- Text Parsing
- CPU Controllers
- Protocol Analysis
- Natural Language Processing
- Speech Recognition

(e) Draw two input NOR Gate using CMOS technology.
Ans: [Diagram: 2 M]

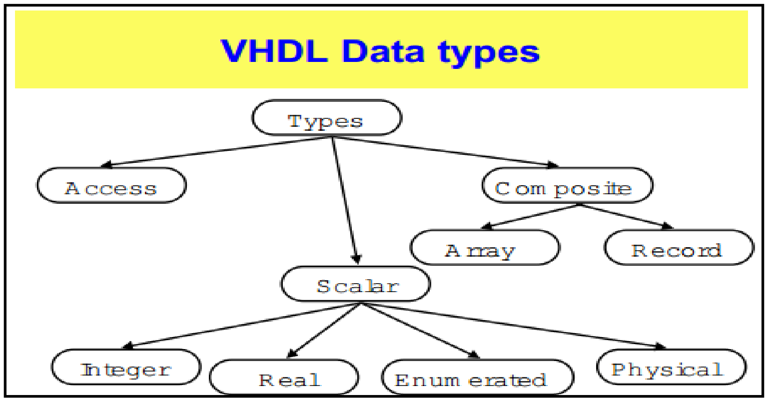


(f) List different capacitances related to CMOS transistor.
Ans. [Any 2, 1 M each]

1) Gate capacitance.
2) Diffusion capacitance.
3) Routing capacitance.

(g) List different Data types in VHDL.
Ans: [Data types in VHDL: 2 M]

[M to be given if listed]



(h) Give the syntax of signal used in VHDL.
Ans: [Syntax:2 M]

SIGNAL signal_name : signal_type [:= initial value];

(i) Write the syntax of CASE statement.
Ans: [Syntax:2 M]

Case expression is
  When choices =>
    {sequential-statement}
  When choices =>
    {sequential-statement}
  end case;

(j) List the different sequential statements used in VHDL.
Ans: [Any 4 – ½ M each]

1) Process Statement
2) If Statement
3) Case Statement
4) Loop Statement
5) Assert Statement
6) Wait Statement

(k) What do you mean by 'simulation'?
Ans: [Mean: 2 M]
Simulation is functional emulation of a circuit design through software programs that use models to replicate how a device will perform in term of timing and results.

(l) List the different software's related to VHDL.
Ans: [Any 2, 1 M each]

Xilinx, Aldec, Cadence, Mentor Graphics, Synopsys.

(m) State the advantages of PLD's.
Ans. [2 advantages 1 M each, any other relevant advantage M to be given]

1) Inexpensive at low quantities.
2) The design inside the chip is flexible, so a change in the logic does not require any rewiring of the board.
3) PLDs are often used for address decoding.

(n) State any four features of XILINX.
Ans: [**Any other relevant features M to be given**]

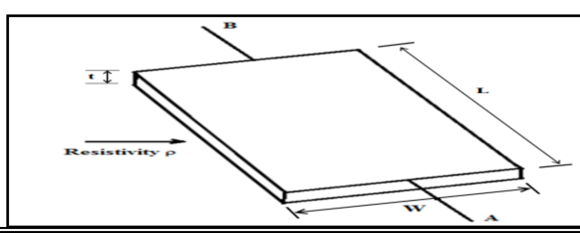
1) Xilinx is a Synthesis Tool which converts Schematic/HDL Design Entry into functionally equivalent logic gates on Xilinx FPGA, with optimized speed & area.
2) Xilinx Tool generates Post-Process Simulation Model after every Implementation Step.
3) Allows Mixed Mode HDL Design Entry.
4) Xilinx ISE allows integration with other Synthesis Engine from Mentor Graphics/Exemplar, Synopsys and Simplicity.

Q.2. Attempt any FOUR of the following: [16M]
(a) Distinguish asynchronous sequential circuit and synchronous sequential circuits.
Ans: [Distinguish-Each point 1 M]

SR.NO.	ASYNCHRONOUS SEQUENTIAL CIRCUIT	SYNCHRONOUS SEQUENTIAL CIRCUITS.
1	Output can be changed at any instant of time by changing the input	Output changes at discrete interval of time
2	The status of memory element will change any time as soon as input is changed.	The status of memory is affected only at the active edge of clock, if input is changed.
3	These circuits are difficult to design	These circuits are easy to design.
4	They are faster	They are slower

(b) Explain estimation of channel resistance of CMOS.
Ans:

- Consider a uniform slab of conducting material of resistivity ρ. Let W be the width, t the thickness and *L* the length of the slab.



Hence the resistance between A and B terminal.

$R_{AB} = \frac{\rho L}{A}$ ohms.

Where A = cross-sectional area.

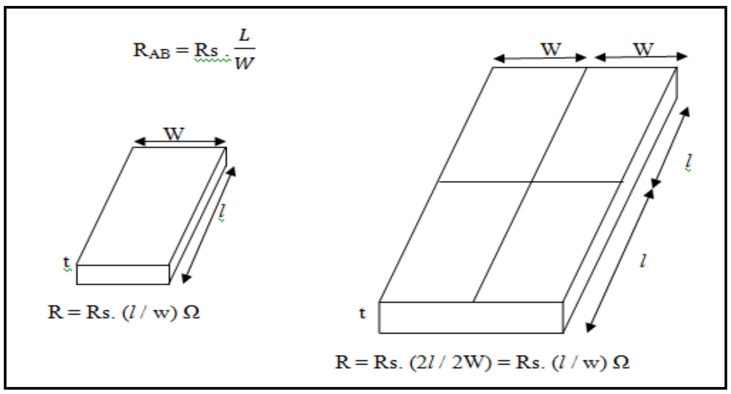
Thus $R_{AB} = \frac{\rho L}{t.W}$ ohms.

Consider the case in which L = W, that is a square of resistive material then

$$R_{AB} = \frac{\rho}{t} = Rs$$

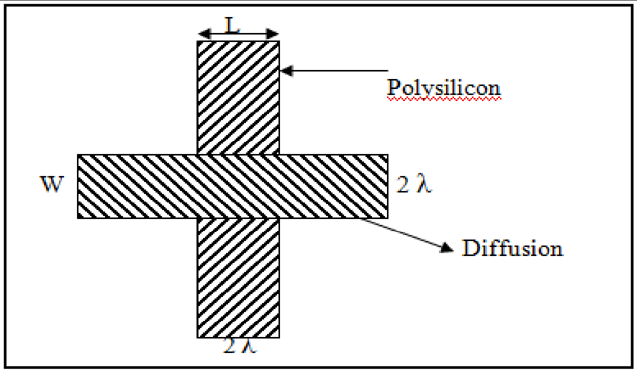
Where

- Rs = ohm per square or sheet resistance
- Therefore, $Rs = \frac{\rho}{t}$ ohm per square
- Hence Rs is completely independent of the area of the square.
- Thus,



- Thus to obtain the resistance of a conductor on a layer multiply the sheet resistance Rs, by the ratio of length to width of the conductor as shown in the equation. For examples, resistances of the two shapes shown in the above figure are same because the length to width ratio of both the slabs is same, even though the sizes are different. Although the voltage – current characteristics of a MOS transistor are generally non-linear, it is used to approximate its behavior in terms of a 'change resistance' to estimate the performance.
- The channel resistance Rc
- $Rc = K\left(\frac{L}{W}\right)$
- Where $K = \frac{1}{\mu\ Cox\ (vgs - Vt)}$

  μ = surface mobility of majority carriers. (i.e. electrons in n-device and holes in p-device)
- Since mobility and threshold voltage are temperature dependent parameters, the channel resistance change with temperature. But as given in equation of Rc, channel resistance mainly depends on length to width ratio of the channel.
- The transistor is formed when the polysilicon (poly) and diffusion cross rach other. Its layout representation is shown below.



- In the above diagram both poly and diffusion are of 2λ widths. The overlapping region is called a 'channel', with length and width 2 λ, as shown in figure. The thinnox is only in the channel region.
- In the above example channel length L = 2 λ and width W =2 λ.
- The channel is square in shape and channel resistance.

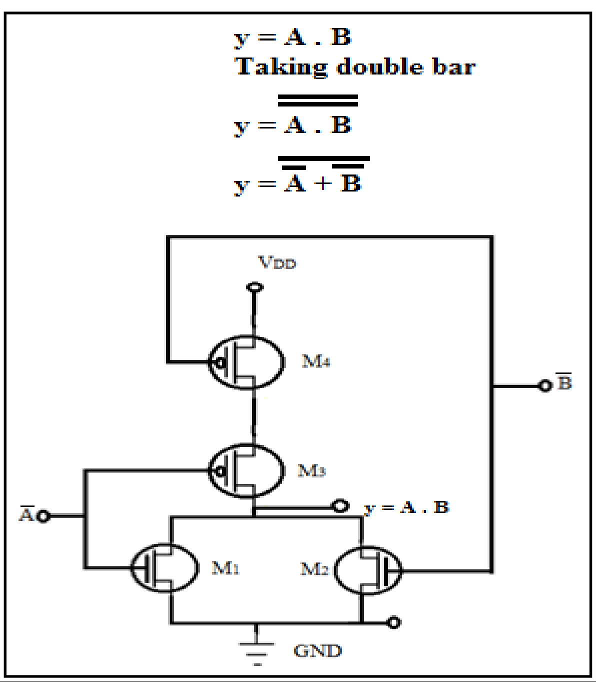
R = Rs $(\frac{L}{W})$

Therefore, R = Rs $(\frac{2\lambda}{2\lambda})$

R = 1^2 . Rs ohm / square.

Therefore, R = Rs ohms.

(c) Draw CMOS AND gate and write it with Truth Table. [2 M input]
Ans:



Inputs		MOS				Output
A'(bar)	**B'(bar)**	**M1**	**M2**	**M3**	**M4**	**y**
0	0	OFF	OFF	OFF	OFF	0
0	1	OFF	ON	OFF	ON	0
1	0	ON	OFF	ON	OFF	0
1	1	ON	ON	ON	ON	1

(d) State various features of VHDL.
Ans: [Any 4 Features,1 M each]

- VHDL is a Hardware Description language used for design entry and simulation of digital circuits.
- VHDL is an event-driven language; that is, whenever an event occurs on signals in VHDL it triggers the execution of the statement.
- VHDL is technology platform- independent language and portable by allowing different vendors to use same design descriptions across different target technologies for implementation.
- VHDL allows both concurrent as well as sequential modeling which are effective in capturing the features of real-life complex circuits consisting of both concurrent combinational logics as well as synchronous sequential logic.
- It gives user the flexibility to define the data types that are specific to user needs apart from the predefined libraries and subprograms.
- It includes advance features of configuration allowing multiple architectures that are existing for the same design.
- It is a case sensitive language, that is, it does not differentiate between lower case and upper case letters.
- VHDL is also said to be stronged typed language, that is, it does not support implicit conversation between data types.
- It allows general purpose designs with the help of generics and attributes.

(e) Write a VHDL code of half adder.
Ans:

```
Library IEEE;
Use IEEE. Std_logic_1164.all;
entity HALF_ADDER is
port(A,   B:     in  bit;
SUM, CARRY: out bit);
end HALF_ADDER;
architecture behavioral of HALF_ADDER is
begin
SUM   <= A xor B;
CARRY <= A and B;
end behavioral;
```

(f) List different levels of simulation and explain in brief.
Ans: [Explanation: 4M]

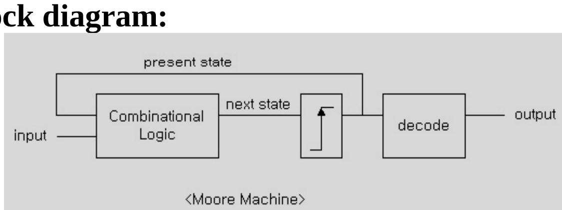
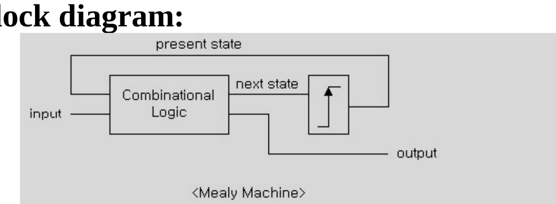
Behavorial Simulation:
Behavioral simulation employs a high level of abstraction to model the design. A behavioral design might, for example, contain high-level operations, such as a four-bit addition operator (this is not an adder, as in a structural design.

Pre- synthesis simulation:
RTL simulation is pre-synthesis simulation where the VHDL code for circuit under design is verified for correct result. This is a designer's first step to begin with a design.

Post- synthesis simulation:
A functional simulation refers to post synthesis simulation after a required optimization is performed and constraints applied. Its main function is not just to verify the functionality of design of design alone as done in RTL simulation, but to see the timing constraints, critical path analysis, total interconnect delays, and whether the functional simulation is performed.

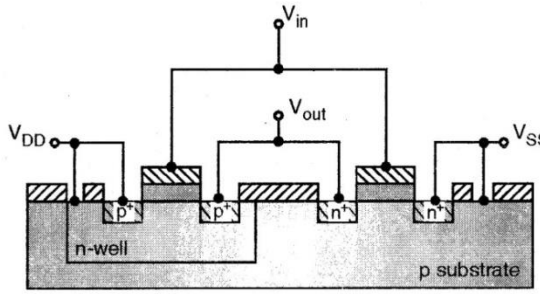
Q3: Attempt [any 4]: **16 M**

(a) Compare Mealy and Moore Machine.
Ans: [any 4 points with diagram – 1 M each]

SR NO	MOORE MACHINE	MELAY MACHINE
1.	Output is function of state of machine.	Output is function of state of machine and present input condition.
2.	Requires more number of states.	Requires less number of states.
3.	Faster.	Slower.
4.	Simple design.	Complex design.
5.	Output in state.	Output is at the time of state transition.
6.	**Block diagram:** 	**Block diagram:** 

(b) Explain fabrication of N-well process.
Ans: [diagram -2 M and explanation- 2 M]

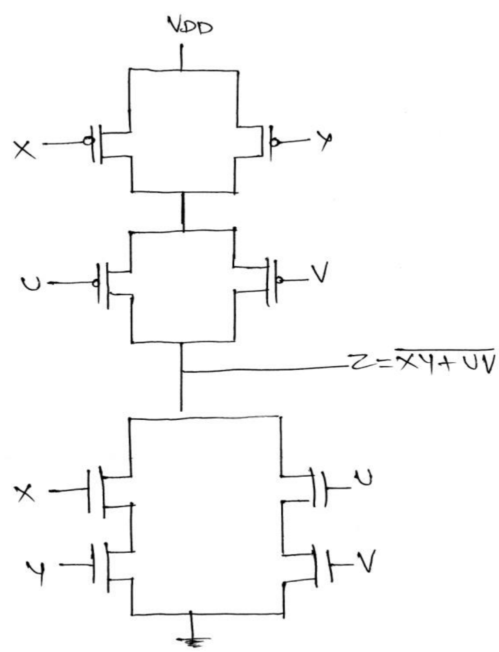
N-Well process: The N-well CMOS circuits are getting more popular because of the lower substrate bias effect on transistor threshold voltage and lower parasitic capacitances associated with source and drain regions.



The fabrication steps are as follows:

- Thick SiO_2 layer is grown on p-type silicon wafer.
- After defining the area for N-well diffusion, using a mask, the SiO_2 layer is etched off and n-well diffusion process is carried out.
- Oxide in the n transistor region is removed and thin oxide layer is grown all over the surface to insulate gate and substrate.
- The polysilicon is deposited and patterned on thin oxide regions using a mask to form gate of both the transistors. The thin oxide on source and drain regions of both the transistors is removed by proper masking steps.
- Using n^+ mask and complementary n^+ mask, source and drain of both nMOS and pMOS transistors are formed one after another using respective diffusion processes. These same masks also include the V_{DD} and V_{SS} contacts.
- The contacts are made using proper masking procedure and metal is deposited and patterned on the entire chip surface.
- An overall passivation layer is formed and the openings for accessing bonding pads are defined.

(c) Design $Z = \overline{XY + UV}$ using CMOS logic.
Ans: [correct diagram -4 M]



(d) Give the syntax of Entity and Architecture using in VHDL programming.
Ans: [2 M each with syntax]

Entity declaration: It defines the names. Input output signals and modes of a hardware module. Also it provides the external interface of an entity. It is a black box view.

Syntax:

entity entity _ name is
    Port declaration.
end entity_name.

An entity declaration should starts with starts with 'entity' ends with 'end' keywords.
Port are interfaces through which an entity can communicate with its environment. Each port must have a name. direction and a type. An entity may have no port declaration also. The direction will be input. output or in out.

In: port can be read
Out: port can be written
Inout :port can be read and written
Buffer: port can be read and written. It can have only one source.

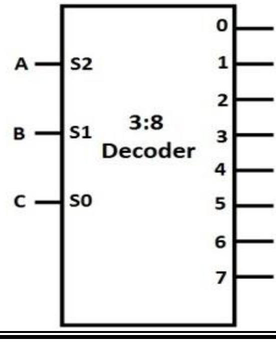
Architecture: It describes the internal description of it tells what is there inside design. Each entity has at least one architecture and an entity can have many architecture. Architecture can be described using structural, dataflow, behavioral or mixed style. Architecture can be used to described a design at different levels of abstraction like gate level (RTL) or behavior level.

Syntax:

architecture architecture _name of entity_ name
Architecture_ declaration_ name;
begin
    Statement;
end architecture_ name;

Here we should specify the entity name for which we are writing the architecture body. The architecture statements should be inside the end keyword. Architecture declarative part may contain variables, constants, or component declaration.

(e) Draw 3:8 decoder and right VHDL code of it.
Ans: [diagram – 1 M, Code – 3 M]



```
Library IEEE ;
Use IEEE .std _logic =1164.all ;
Entity  decoder is
Port ( a,b,c: in std _logic;
        Stb  :in std_logic ;
          Y : out std_logic _vector(7 downto 0) );
End decoder;
Architecture beh of decoder is
 signal temp : std_ logic_vector(3 downto 0);
   begin
temp <=stb & a & b &c ;
Y  <=”01111111” when temp =” 0000” else
      “10111111” when temp = “0001” else
    ( and so on for each input )

       “11111110” when temp=”0111”else
        “zzzzzzzz;
End beh;
```

(f) Explain event scheduling with suitable examples.
Ans:[explanation -2 M , example – 2 M]

- Event is change on target signal which is to be updated.
- Ex: X<= a after 0.5 ns when select =0 else
  X<= b after 0.5ns.
- The assignment of signal X does not happen instantly. Each of the values assigned to X contain an after clause.
- The mechanism for delaying the new value is called scheduling an event. By assigning port X a new value, an event was scheduled 0.5 ns in the future that contains the new value for signal x. When the event matures, signal receives a new value.
- Ex:

```
ARCHITECTURE dataflow OF mux IS
  SIGNAL select : INTEGER;
BEGIN
  select <= 0 WHEN s0 = '0' AND s1 = '0' ELSE
            1 WHEN s0 = '1' AND s1 = '0' ELSE
            2 WHEN s0 = '0' AND s1 = '1' ELSE
            3;

  x <= a AFTER 0.5 NS WHEN select = 0 ELSE
       b AFTER 0.5 NS WHEN select = 1 ELSE
       c AFTER 0.5 NS WHEN select = 2 ELSE
       d AFTER 0.5 NS;

END dataflow;
```

Q4: Attempt [any 4]: 16 M

a) Define the following terms:

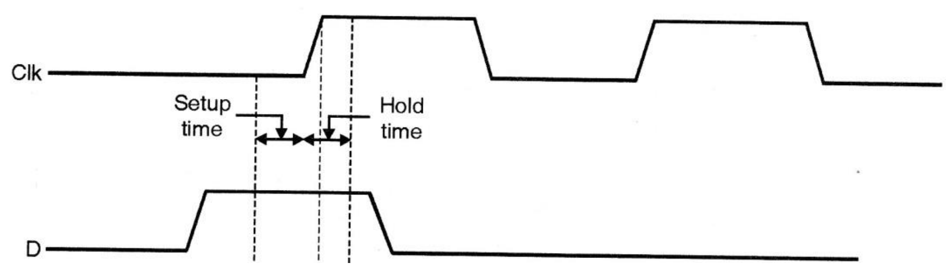
(i) Metastability
(ii) Set-up time
(iii) Hold time
(iv) Fan-out

Ans: [1 M each]

(i) Metastability: Metastability is the ability of digital system to persist for an unbounded time in an unstable equilibrium or metastable state. In this state the circuit may be unable to settle into a stable 0 or 1 logic level within the time required for proper circuit operation.

(ii) Set-up time: Set up time during which the input must be stable before the clock transistion takes place.

(iii)



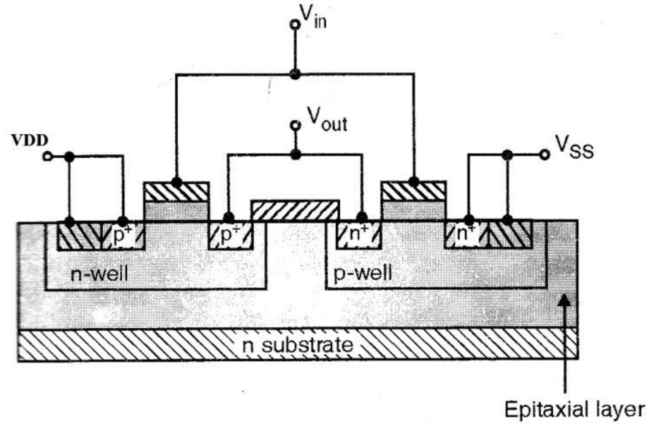
(iv) Hold time: Hold time is the minimum time for which the input must be held constant after the transistion occurs.

(v) Fan-out: Fan-out of an output a logic gate output is the number of gate inputs it can feed or connect to. The maximum fan-out of an output measures its load driving capacity. It is the greatest number of inputs of gates of the same type to which the output can be safely connected.

b) Explain twin tube process with suitable diagram.

Ans: [diagram – 2 M; explanation – 2 M]

In this process the substrate can be of any type. Consider n type silicon substrate. The twin tub fabrication process is:



1. The process is carried out on N type silicon substrate with lower doping or higher resistivity so that the lesser current flows through the substrate. On this, the n^+ Si substrate is grown further i.e. epitaxial layer of required thickness is grown.
2. SiO_2 layer is grown all over the surface and the areas of P well and N well are defined. P well is diffused by masking N well area and N well is diffused by masking P well area.
3. A thin layer of SiO_2 thin ox is deposited all over the surface. Using masking and etching process unrequired thin ox is removed. The thin ox is required only on gate areas of both the transistors.
4. The polysilicon is deposited all over the surface and using a mask it is removed from areas other than the gate area.
5. Then the P well is covered with a photoresist mask and p^+ diffusion is carried out to form the source and drain of pMOS transistor.
6. Now the N well is covered with a photoresist mask and n^+ diffusion is carried out to form the source and drain of nMOS transistor.
7. The thick layer of SiO_2 is grown all over the surface for isolation. This SiO_2 layer is etched off to expose all the terminals.
8. The metal is deposited and patterned all over the wafer surface so that it makes contact with source, drain and gate terminals.

c) Explain various operators used in VHDL.
Ans:[any four with example – 1M each]

The various operators in VHDL are:

- **Logical Operators:** These are defined for type bit and Boolean, one dimensional array of bit and Boolean type. The logical operators are: AND, OR, NAND, NOR, XOR, XNOR, NOT
- For example :

```
c (<=) a and b;
z <= A nand B nand C; -- is illegal.
```

- **Relational operators**: The relational operators are used to check the conditions. The relational operators are:
  1. = Equality
  2. /= Inequality
  3. < Less than
  4. <= Less than or equal
  5. > Greater than
  6. >= Greater than or equal.

The equality and inequality are defined for all types except file types. For array types operands are aligned in the left and compared to the right. These operators return the TRUE logic value where the condition is met.

- For example :

```
type MUL is ('U', '0', '1', 'Z')
type MUL ('U') < MUL ('Z'),
```

is true, since 'U' occurs to the left of 'Z'.

- **Shift Operators**: These are defined for one dimensional array with elements of the type bit and Boolean. The various shift operators are:
  1. sll shift left logical
  2. srl shift right logical
  3. sla shift left arithmetic
  4. sra shift right arithmetic
  5. rll rotate left logical
  6. rrl rotate right locial.

  Assume A is a bit_vector equal to '1001 0100' and then
  1. A sll 2 is '01010100' (shift left logical, filled with '0' at LSB side)
  2. A srl 3 is '00001010' (shift right logical, filled with '0' at MSB side)
  3. A sla 3 is '10101111' (shift left arithmetic, filled with MSB at LSB side)

- **Adding Operators:** The three adding operators are +, - and &. The + and – operators must be of same numeric type and the result is of the same numeric type. Whereas & (concatenation) operator can be one dimensional array type as an element type.
  - For example :
    1. **Signal** A : bit_vector (3 **downto** 0);
       Signal B : bit_vector (3 **downto** 0);
       Signal C : bit_vector (7 **downto** 0;
       C <= not A & not B

    Now, C will be having element of complements of A & B array element.

- **Multiplying operators**: These operators are predefined in VHDL for all integer and real data types and the result is of same type.

A rem B = A – (A / B) * B.

It gives the remainder of the division operator.

A mod B = A – B * N …… N = same integer.

- **Miscellaneous operators:** The two miscellaneous operators are **abs** [absolute] and ** [exponential]. The abs is defined for any numeric type and ** is defined for integer or floating point number.

- **For example :**

  2 ** 8 = 256

  3.8 ** 3 = 54.872

- **Unary operators:** These are sign operators and there are two sign operators positive and negative.

d) Explain how to write test bench for any VHDL code.
Ans: (Explain:4M)

Test Bench:

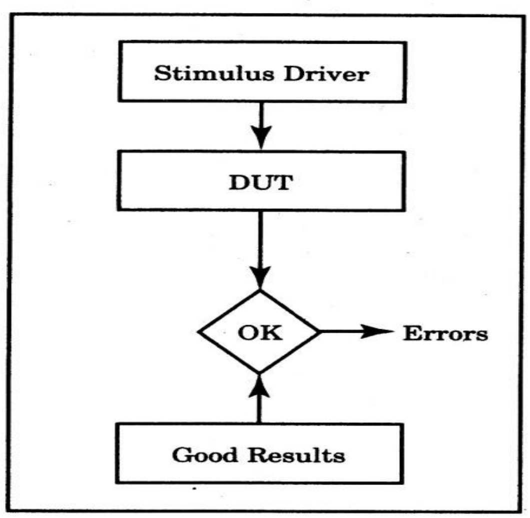
A test bench is HDL code that allows you to provide a documented, repeatable set of stimuli that is portable across different simulators. A test bench can be as simple as a file with clock and input data or a more complicated file that includes error checking, file input and output, and conditional testing.

A test bench or testing workbench is an (often virtual) environment used to verify the correctness or soundness of a design or model, for example, that of a software product.

A test bench refers to an environment in which the product under development is tested with the aid of software and hardware tools. The suite of testing tools is often designed specifically for the product under test. The software may need to be modified slightly in some cases to work with the test bench but careful coding can ensure that the changes can be undone easily and without introducing bugs.

Block diagram of Test Bench:

It encapsulates the stimulus driver, known good results, and DUT and contains internal signals to make the proper connections. The stimulus driver drives the input into DUT which responds and produces results. Finally a compare function within the test bench compares the result from the DUT against those known good results and reports any errors.



Typical Test Bench Format is:

entity TEST_BENCH is
end;

architecture TB_BEHAVIORof TEST_BENCHis
component ENTITY_UNDER_TEST
port (list-of-ports-their-types-and-modes);
end component;
Local-signal-declarations ;
begin
Generate-waveforms-using-behavioral-constructs;
Apply-to-entity-under-test ;
EUT : ENTITY_UNDER_TEST **port map** *(port-associations)* ;
Monitor-values-and-compare-with-expected-values;
end TB_BEHAVIOR;

e) Explain Event based and cycle based simulator.
Ans: [2 M each]

1. **Event based simulator** :

- Event driven signal keeps track Of any change in the signal in the event queue.
- The simulator starts simulation as soon as any signal in event list changes its value.
- For this the simulator has to keep record of all the scheduled events in future. This causes a large memory overload but gives high accuracy for asynchronous design. It simulates events only.
- Gates whose inputs have events are called active and are placed in activity list.
- The simulation proceeds by removing a gate from the activity list. The process Of evaluation stops when the activity list becomes empty.

2. **Cycle-based Simulator** :

- Cycle-based simulation ignores intra—cycle state transitions. i.e. they check the Status Of target signals periodically irrespective of any events. This can boost performance by 10 to SO times compared to traditional event-driven simulators.
- Cycle-based technology offers greater memory efficiency and faster simulation run-time than traditional pure event-based simulators.
- Cycle-based simulators work best with synchronous design but give less timing accuracy with asynchronous design.
- Signals are treated as variables. Functions such as AND. OR etc. are directly converted to program statements.
- Signal level functions such as memory blocks, adders. Multiplier’s etc. are modeled as subroutines.
- For every input vector. the code is repe executed until all variables have attained steady value.
- Compiled code simulator is efficient when used for high-level design verification. Inefficiency is incurred by the evaluation of the design when only few inputs are changing.

f) Explain sensitivity list and zero modelling.
Ans: [2M each]

Sensitivity List:

- The sensitivity list is a compact way of specifying the set of signals, events on which may resume a process. A sensitivity list is specified right after the keyword process.
- The sensitivity list is equivalent to the **wait on** statement, which is the last statement of the process statement section.

- Only static signal names, for which reading is permitted, may appear in the sensitivity list of a process, i.e. no function calls are allowed in the list.

Example 1:

```
DFF : PROCESS (CLK,RST)
BEGIN
 IF RST = '1'
THEN Q <= '0';
ELSIF (CLK'EVENT) AND (CLK = '1')
THEN Q <= D;
END IF;
END PROCESS DFF;
```

```
DFF : PROCESS
BEGIN
IF RST = '1'
THEN Q <= '0';
ELSIF (CLK'EVENT) AND (CLK = '1')
THEN Q <= D;
END IF;
WAIT ON RST, CLK;
END PROCESS DFF;
```

Here, the process is sensitive to the RST and CLK signals, i.e. an event on any of these signals will cause the process to resume. This process is equivalent to the one described in the comment section.

A process with a sensitivity list may not contain any explicit wait statements. Also, if such a process statement is a parent of a procedure, then that procedure may not contain a wait statement as well.

Zero Modelling:

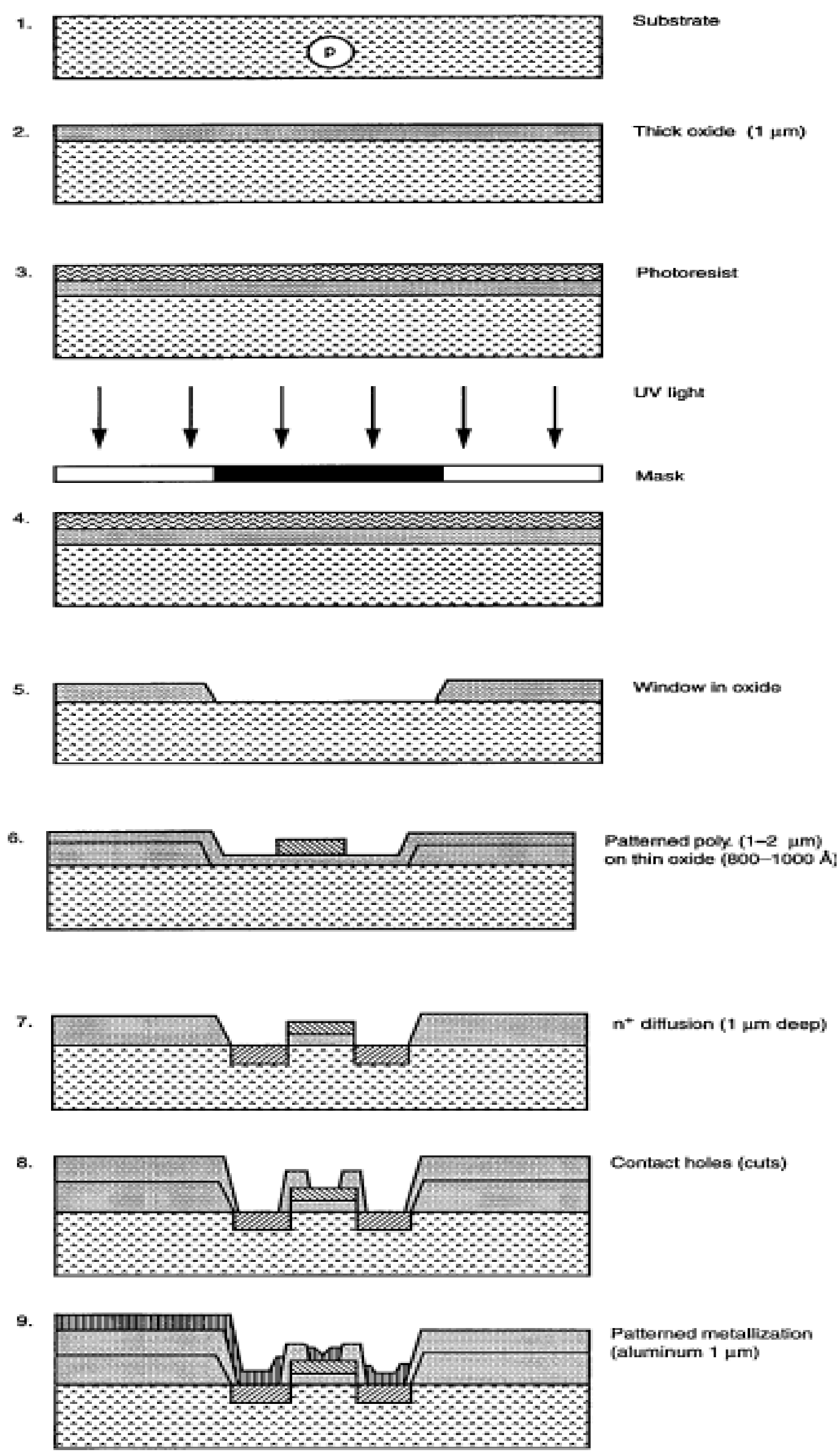
- While describing any system for synthesis circuit delays are determined by the target technology. While writing VHDL for synthesis signal, assignment statements never included a delay assignment.
- All digital circuit elements have a delay [propagation delay] which is very small in terms of nano sec. This nano sec delta delay will have little impact while writing the VHDL code. But for circuit realization this delay must be incorporated. The physical circuits always have finite delays.
- In VHDL zero delay circuits and designs that depends on zero delay components can never be built. Simulation deltas are used to order some types of events during simulation. Specifically zero delay events must be ordered to produce consistent results. If they are not properly ordered results can vary between different simulation runs.

5. Attempt any FOUR of the following : 16

(a) Write various steps of well process.

Ans: Steps in Well Process:

[2M for diagram/Names of steps in Proper Order + 2 M for explanation]



(b) List different concurrent statements and explain any two.

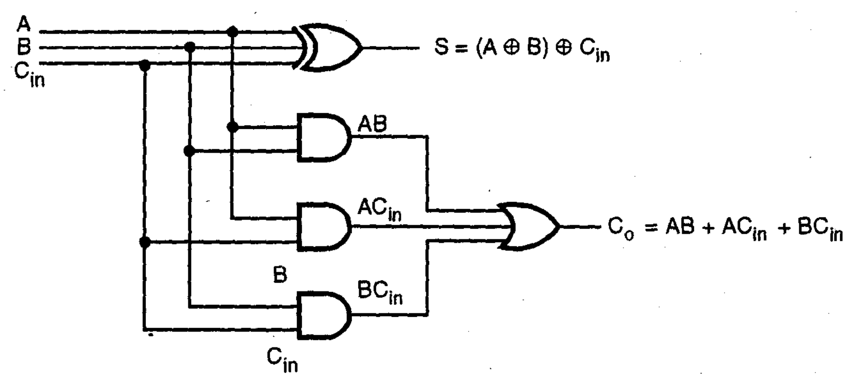
Ans: Concurrent Statements in VHDL
[2 M for explanation + 2 M for statements /examples]

The operations in concurrent statements are executed concurrently.They are all active at same time. The VHDL language models real systems as a set of subsystems that operate concurrently. Concurrent statements are:

- With –select statement
- When –else statement
- Generate statement
- Block statement

(c) Draw full Adder using gates and write VHDL code for it.

Ans: [1 M diagram+3 M for VHDL]



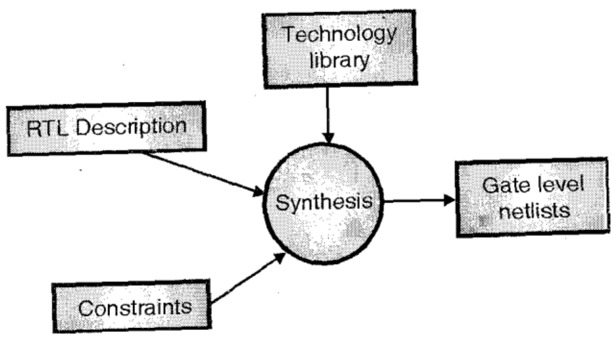

```
library IEEE,
use IEEE.std_logic_1164.all,
entity f_Adder is
port ( a, b,cin: in BIT;
Sum,Carry : out BIT  );
end f_Adder;
architecture dataflow of f_adder is
begin
sum <= A xor B xor Cin;
carry <= (A and B) or (A and Cin) or (Cin and B);
end dataflow;
```

(d) Write the various steps of synthesis and explain in short.

Ans: Synthesis [Block diagram 2 M + explanation 2M]
Synthesis = Translation + Optimization.

Synthesis is an automatic method of converting higher level of abstraction to lower level of abstraction. i.e.

- The process that converts user, hardware description into structural logic description. Synthesis is a means of converting hdl into real world hardware. It generates a gate level net list for the target technology. The synthesis tool converts register transfer level (rtl) description to gate level netlist. These gate level netlists consist of interconnected gate level macrocells.
- The inputs to the synthesis process are rtl (register transfer level) vhdl description, circuit constraints and attributes for the design, and a technology library.



- The synthesis process produces an optimized gate level net list from all these inputs. The translation from RTL description to Boolean equivalent description is usually not user controllable.
- The intermediate form that is generated is a format that is optimized for a particular tool and may not even be viewable by the user. All the conditional signal assignments and selected signal assignment statements are converted to their boolean equivalent in this intermediate form. The optimization process takes an unoptimized Boolean description and converts it to an optimized Boolean description. For this it uses number of algorithm and rules. This process aims to improve structure of Boolean equations by applying rules of boolean algebra. This removes the redundant logic and reduces the area requirement.

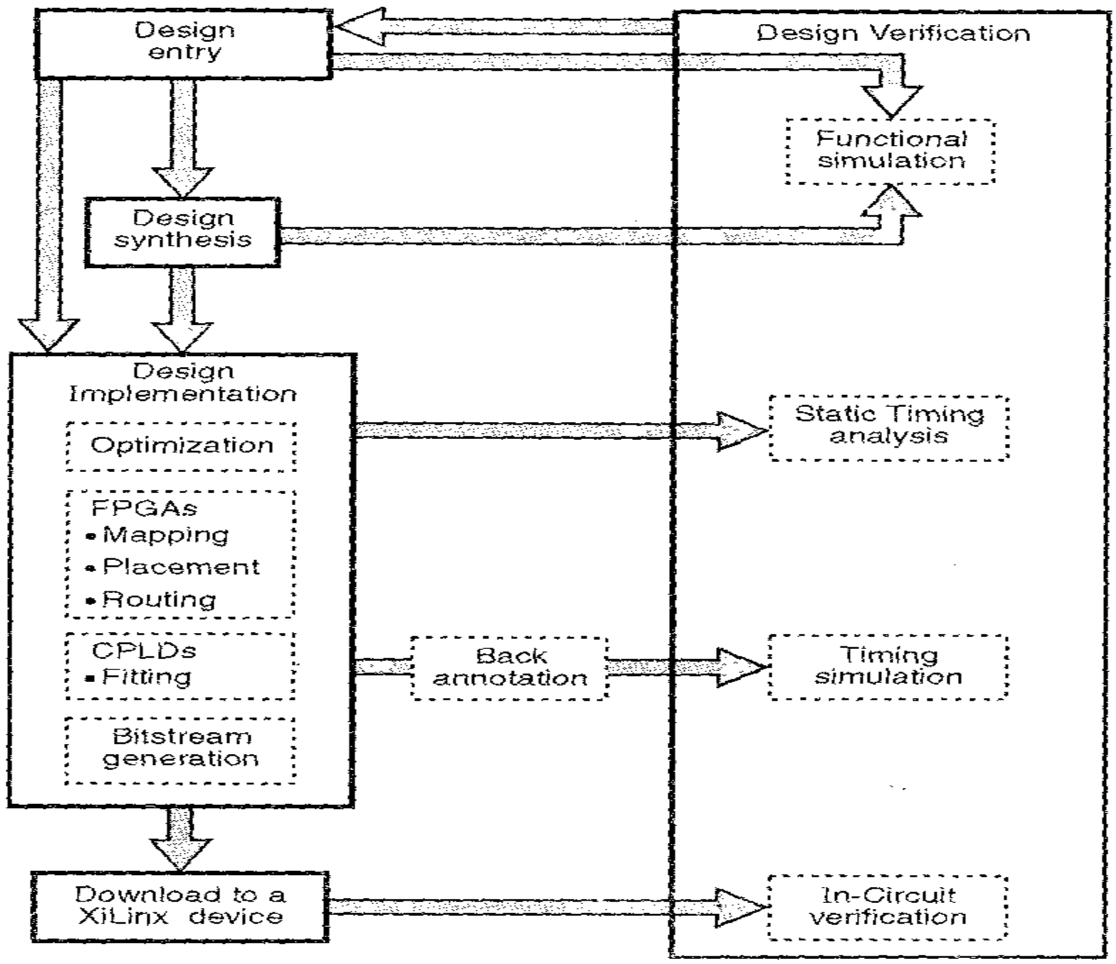
(e) Explain efficient coding styles.

Ans: Effective Coding Technique [4 M]

- There may be more than one method to model a particular design part but only a few would yield better performance.
- The essence of VHDL coding lies in understanding which style yields the ultimate performance under the given set of specifications.
- The key to higher performance is to avoid writing code that needlessly creates additional work for the HDL compiler and synthesizer, which, in turn, generates designs with greater number of gates.
- Basically, any coding style that gives the HDL simulator information about the design that cannot be passed onto the synthesis tool is a bad coding style.

(f) Draw design flow of ASIC and explain it.

Ans: ASIC Design [diagram 3 M+ 1 M for suitable explanation]



6. Attempt any FOUR of the following : 16

(a) Compare CMOS and BJT Technology.

Ans: Compare BJT and CMOS[1 M each point – any 4 points]

Sr. No.	Bipolar Junction Transistor	Complementary Metal Oxide Semiconductor
1	BJT junctions are emitter base and collector	CMOS junctions are gate, source, drain and substrate
2	LOW power applications	High power applications
3	Bipolar device	Unipolar Device
4	Low input Impedance	High Input Impedance
5	Low current gain	High Current gain
6	More fan out	Less fan out
7	Low packing density	High Packing density
8	Connecting BJT's together gives rise to a family of logic gates known as TTL	Connecting NMOS and PMOS transistors together gives rise to the CMOS family of logic gates

(b) Explain the shift operations and logical operations.

Ans: Shift and Logical Operations: (SHIFT *[any 4]* 2 M + Logical*[any 4]* 2 M)

LOGICAL
AND
OR
NAND
NOR
XOR
XNOR
NOT

SHIFT	
SLL	Shift Left Logical
SRL	Shift Right Logical
SLA	Shift Left Arithmetic
SRA	Shift Right Arithmetic
ROL	Rotate Left Logical
ROR	Rotate Right Logical

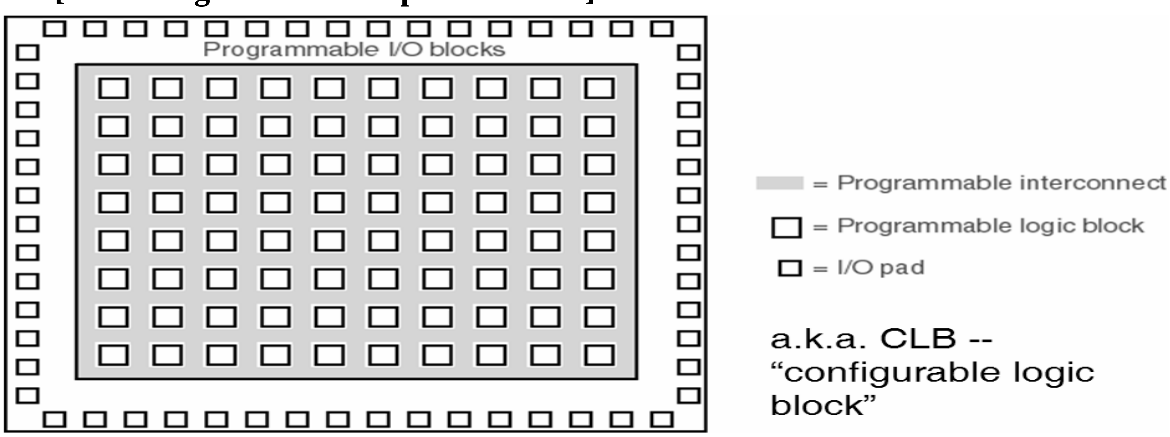
(c) Write VHDL programme for 1 : 4 Demux using when-else statement.

Ans:1:4 demux [Entity 1 M+ When-else statement 3 M]

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity demux1_4 is
Port ( I : IN STD_LOGIC;
S : IN STD_LOGIC_VECTOR(1 downto 0);
Y : OUT STD_LOGIC_VECTOR(3 downto 0));
END demux1_4;
ARCHITECTURE concurrent OF demux1_4 IS
BEGIN
Y<=(I & “000”) when s=”00” else
     (‘0’&I & “00”) when s=”01” else
     (“00”&I & ‘0’) when s=”01” else
     (“000”&I);
END concurrent;
```

(d) Draw FPGA configurable logic block diagram.

Ans: FPGA [Block diagram 2 M + Explanation 2M]



Generalized Block Diagram of FPGA

- A field programmable gate array (FPGA) is like a CPLD turned inside out. The logic is broken into a large number of programmable logic blocks that are individually smaller than a PLD. They are distributed across the entire chip in a sea of programmable interconnections, and the entire array is surrounded by programmable I /O blocks. An FPGA's programmable logic block is less capable than a typical PLD but an FPGA chip contains a lot more logic blocks than a CPLD of the same die size has PLDs.

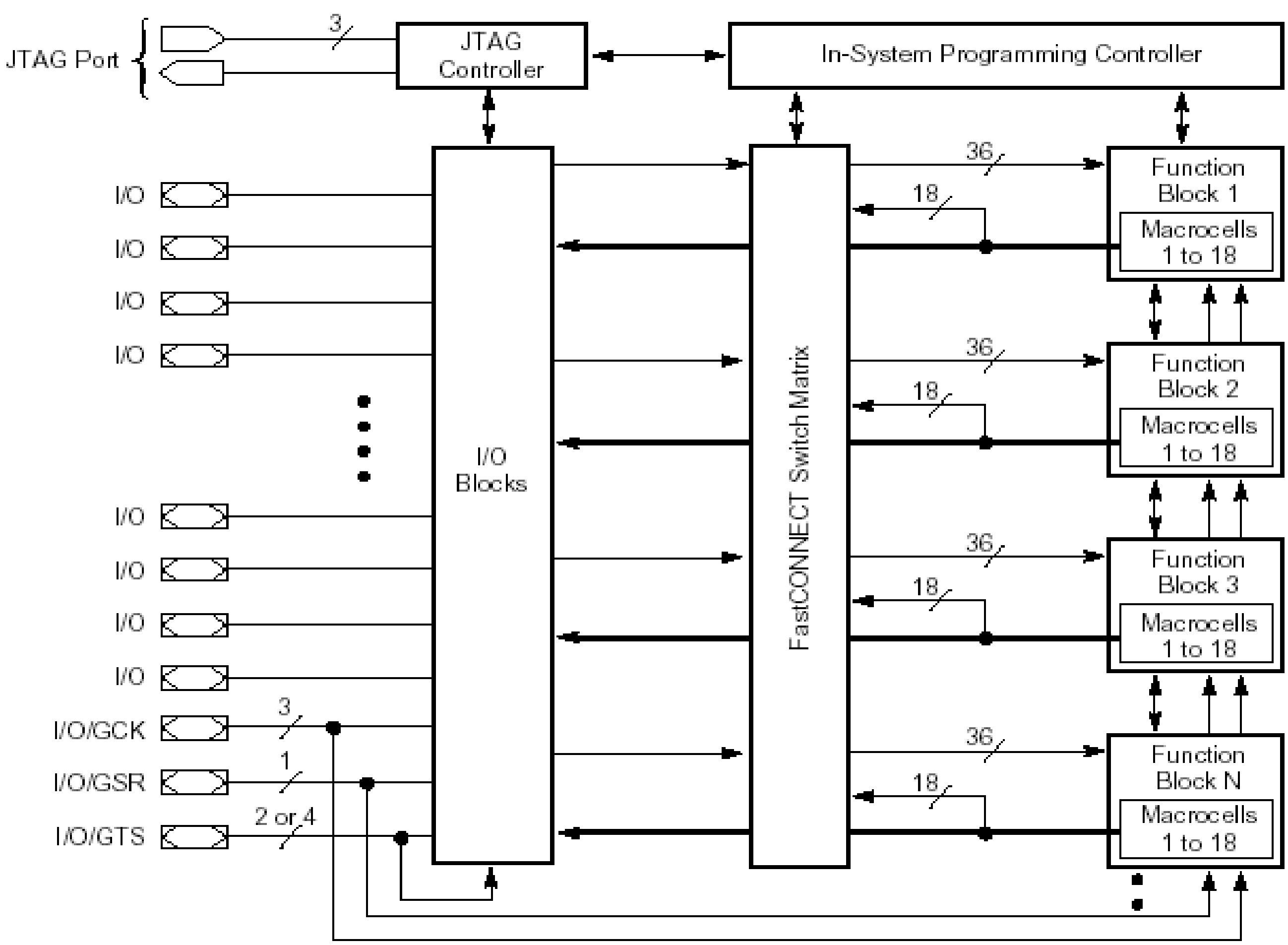
(e) Write various factors for selection of FPGA.

Ans: Factors for selection of FPGA [any 4 factors – 4 M]

1. Technical Feasiblity
2. Cost
3. External Devices
4. No of Pins
5. Frequency of operation
6. No of LUT
7.Look up tables

(f) Draw architecture of CPLD and explain in brief.

Ans: Architecture of CPLD[Block Diagram 2 M+ Explanation *[any 6 points]* 2 M]



Architectural Description:

- Each external I/O pin can be used as an input, an output, or a bidirectional pin according to device programming. The I/O pins at the bottom are also used for special purposes.
- Any of the 3 pins can be used as "Global Clocks" (GCK).Each macro cell can be programmed to use a selected clock input.
- One pin can be used as a "Global Set/Reset "(GSR).Each macro cell can use this signal as an asynchronous Preset or Clear.
- Two or Four pins depending on the devices can be used as "Global Three State Controls"(GTS).One of the signals can be selected in each macro cell to output enable the corresponding output driver when the macro cell's output is hooked to an external I/O pin.
- Only four Functional Blocks(FB) are shown but XC9500 scales to accommodate 16 FB's in the XC95288.Regardless of the specific family member each FB programmable receives 36 signals from the switch matrix. The inputs to the switch matrix are the 18 macro cell outputs from each of the functional blocks and the external inputs from the I/O pins.
- Each Functional block also has 18 outputs that run under the switch matrix and connect to the I/O blocks. These are the output-enable signals for the I/O block output drives; they're used when FB macro cells output is hooked up to an external I/O pin. Each Functional Block has programmable logic capability with 36 inputs and 18 outputs. Fast Connect Switch Matrix connects all Functional Block outputs to the I/O blocks and the input signals from the I/O block to the Functional Block.