## Q.1         What are different characteristics of VHDL? Elaborate in detail.

**1. Multiple Design Representations:**

VHDL provides multiple ways to represent a digital design, including structural, behavioral, and dataflow descriptions. This flexibility allows designers to choose the most appropriate representation for each part of their design.

**2. Concurrent Execution:**

VHDL is a concurrent language, which means that multiple statements can be executed simultaneously. This is a natural way to model the behavior of digital circuits, where multiple signals can change at the same time.

**3. Data Abstraction:**

VHDL supports data abstraction through the use of data types and packages. This allows designers to create their own custom data types to represent the signals and variables in their designs.

**4. Hierarchical Design:**

VHDL supports hierarchical design, which means that complex designs can be broken down into smaller, more manageable modules. This makes it easier to design and verify large systems.

## Q2.Differentiate between PLA and PAL devices.

|  |  |  |
| --- | --- | --- |
| Feature | PLA- | PAL- |
| AND array | Programmable | Programmable |
| OR array | Programmable | Fixed |
| Flexibility | More flexible | Less flexible |
| Speed | Slower | Faster |
| Complexity | Higher | Lower |
| Cost | Higher | Lower |
| Availability | Less available | More available |

## Q3. Explain different types of VHDL modelling style.

Structural modeling describes the design as a hierarchy of interconnected components. Each component is represented by an instantiated entity, and the connections between components are specified using component instantiation statements. Structural modeling is a good choice for designs that are already well-defined in terms of their components and their interconnections.

Advantages:

* Easy to understand and visualize
* Good for designs with well-defined components
* Efficient for synthesis

Disadvantages:

* Can be verbose for large designs
* Does not explicitly specify the behavior of the components
* Can be difficult to modify

2. Behavioral Modeling:

Behavioral modeling describes the design using sequential statements and processes. The behavior of each signal is defined by a process statement, which specifies how the signal's value changes in response to changes in other signals. Behavioral modeling is a good choice for designs that are not well-defined in terms of their components, or for designs that need to be described in a more abstract way.

Advantages:

* Concise and expressive
* Good for designs with complex behavior
* Easy to modify

Disadvantages:

* Can be difficult to understand for large designs
* Less efficient for synthesis than structural modeling

3. Dataflow Modeling:

Dataflow modeling describes the design using concurrent signal assignment statements. The value of each signal is defined by an expression that depends on the values of other signals. Dataflow modeling is a good choice for designs that are highly concurrent, or for designs that need to be described in a way that emphasizes the flow of data.

Advantages:

* Very expressive and concise
* Good for highly concurrent designs
* Easy to understand for large designs

Disadvantages:

* Can be difficult to write for complex designs
* Less efficient for simulation than behavioral modeling

**Q4.** Differentiate between CPLD and FPGA devices.

|  |  |  |
| --- | --- | --- |
| Feature | CPLD | FPGA |
| Logic blocks | Smaller (10s to 100s) | Larger (100s to 10,000s) |
| Interconnect | Less flexible | More flexible |
| Speed | Slower | Faster |
| Power consumption | Lower | Higher |
| Cost | Lower | Higher |
| Programming | EEPROM | SRAM or Flash |
| Volatility | Non-volatile | Volatile |
| Application | Glue logic, simple state machines | Complex designs, high-performance applications |

5 What is difference between signal and variable in VHDL?

|  |  |  |
| --- | --- | --- |
| Feature | Signal | Variable |
| Declaration | Declarative region | Process statement |
| Scope | Architecture | Process |
| Update | At the end of a process | Immediate |
| Typical use | Communication | Temporary storage |

Q6. What do you mean by synthesis?

synthesis refers to the process of converting a high-level description of a digital circuit into a gate-level netlist. The gate-level netlist is a list of the logic gates and their interconnections that make up the circuit. Synthesis is a crucial step in the VLSI design process, as it allows the designer to verify that the circuit will function correctly and to estimate its performance characteristics.

There are two main types of synthesis: logic synthesis and gate-level synthesis

Q7 What are different programming technologies used in PLDs?

* **Mask programming:** This is the oldest and simplest programming technology. It involves physically modifying the chip to create the desired circuit. Mask programming is a very expensive and time-consuming process, and it is not practical for most PLDs.
* **Fuse programming:** This is a more flexible and less expensive programming technology than mask programming. It involves blowing fuses to create the desired circuit. Fuse programming is a destructive process, which means that once a chip has been programmed, it cannot be reprogrammed.
* **EPROM programming:** This is a more flexible programming technology than fuse programming. It involves using an EPROM (Erasable Programmable Read-Only Memory) to store the circuit configuration. The EPROM can be erased and reprogrammed multiple times.
* **EEPROM programming:** This is the most flexible programming technology used in PLDs. It involves using an EEPROM (Electrically Erasable Programmable Read-Only Memory) to store the circuit configuration. The EEPROM can be erased and reprogrammed electrically, without removing the chip from the circuit.
* **FPGA programming:** Field Programmable Gate Arrays (FPGAs) are a type of PLD that can be programmed using SRAM (Static Random Access Memory). SRAM is a volatile memory, which means that the circuit configuration is lost when power is lost. However, FPGAs can be reprogrammed quickly and easily, which makes them a good choice for applications where the circuit configuration needs to be changed frequently.
* Q 8 Explain JTAG in detail.

JTAG, which stands for Joint Test Action Group, is an industry standard for testing and debugging integrated circuits (ICs). It is a hardware interface that allows external test equipment to access and control the internal circuitry of an IC. This makes it possible to test the IC for faults and to debug problems with its operation.

**JTAG Architecture**

The JTAG architecture consists of four signals and a TAP (Test Access Port) controller. The four signals are:

* **TMS (Test Mode Select):** This signal controls whether the IC is in test mode or normal operating mode.
* **TCK (Test Clock):** This signal clocks the TAP controller.
* **TDI (Test Data In):** This signal is used to input data to the TAP controller.
* **TDO (Test Data Out):** This signal is used to output data from the TAP controller.

Q9 What do you mean MOD-N counter? Calculate the number of F/F required for designing MOD-10 counter.

A MOD-N counter is a counter that counts up to N-1 and then resets to 0. For example, a MOD-4 counter would count 0, 1, 2, 3, and then reset back to 0.

The number of flip-flops required to design a MOD-N counter is equal to the number of bits required to represent N in binary. For example, a MOD-10 counter would require 4 flip-flops because 10 in binary is 1010.

Q10 Differentiate between synchronous and asynchronous design

|  |  |  |
| --- | --- | --- |
| Feature | Synchronous Design | Asynchronous Design |
| Clocking | Global clock signal | Local events |
| Predictability | Very predictable | More difficult to analyze |
| Flexibility | Less flexible | Very flexible |
| Scalability | Less scalable | Very scalable |

Q11 Write VHDL statements for positive edge triggered clock and negative edge triggered clock.

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity dff is

port (

data : in STD\_LOGIC;

clk : in

STD\_LOGIC;

q : out

STD\_LOGIC

);

end dff;

architecture behav of dff is

begin

process (clk)

begin

if (clk'event

and clk = '1') then

q <= data;

end

if;

end

process;

end behav;

Negative edge triggered clock

VHDL

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity dff\_neg is

port (

data : in STD\_LOGIC;

clk : in

STD\_LOGIC;

q : out

STD\_LOGIC

);

end dff\_neg;

architecture rtl of dff\_neg is

begin

process (clk)

begin

if (clk'event

and clk = '0') then

q <= data;

end

if;

end

process;

end rtl;

Q 12 Explain how faster clock can be converted into slower clock using VHDL?

Converting a faster clock into a slower clock using VHDL can be achieved using a clock divider or a clock PLL (Phase-Locked Loop).

**Clock Divider**

A clock divider is a simple method to generate a slower clock from a faster one. It works by dividing the input clock frequency by an integer value. For example, to generate a 1 MHz clock from a 10 MHz clock, you would use a clock divider with a division ratio of 10.

Clock PLL

A clock PLL is a more sophisticated method to generate a slower clock from a faster one. It uses a feedback loop to ensure that the output clock is phase-locked to the input clock, even if the input clock frequency varies. This makes it a better choice for applications where the output clock frequency needs to be stable and accurate.

Q 13 What are different control signals in LCD?

1. **RS (Register Select):** This signal determines whether data is being sent to the LCD's data register or command register. When RS is low, data is sent to the data register, controlling the displayed characters or graphics. When RS is high, data is sent to the command register, altering the LCD's configuration, such as cursor position, display mode, or backlight intensity.
2. **R/W (Read/Write):** This signal indicates whether the LCD is reading or writing data. When R/W is low, the LCD is writing data from the microcontroller to the LCD's registers. When R/W is high, the LCD is reading data from the LCD's data register to the microcontroller.
3. **E (Enable):** This signal triggers the execution of a command or data transfer. When E is pulsed high, the LCD acknowledges the data or command and initiates the corresponding action.
4. **Data Lines (D0-D7):** These parallel data lines transmit information to the LCD, such as character codes, pixel data, or command instructions. The number of data lines may vary depending on the LCD's resolution and interface.
5. **Backlight Control:** This signal regulates the intensity of the LCD's backlight, adjusting the overall brightness of the displayed information.
6. **Contrast Control:** This signal adjusts the contrast ratio between the displayed characters or graphics and the background, ensuring optimal readability.
7. **Power Control:** This signal powers on or off the LCD, enabling or disabling its operation.
8. **Reset:** This signal resets the LCD to its default state, clearing any pending commands or data and initializing the display.
9. **Clock Signal:** This signal provides a timing reference for data transfers and synchronizes the LCD's operation with the microcontroller.
10. **Synchronization Signals (HSYNC, VSYNC):** These signals synchronize the timing of image data with the LCD's scan lines and refresh rate, ensuring proper image display.

Q14 Differentiate between Mealy and Moore FSM

|  |  |  |
| --- | --- | --- |
| Feature | Mealy Machine | Moore Machine |
| Output generation | Based on current state and current input | Based on current state only |
| Output changes | Can change at any time | Can only change when the state changes |
| Hardware complexity | Generally simpler | Generally more complex |
| Reaction time | Reacts faster to inputs | Reacts one clock cycle later |
| Applications | Systems where output is needed immediately | Systems where output can be delayed |

Q15 Explain following.

* 1. Types of scaling
  2. Inverter V-I Characteristics

c. Parasitic in MOSFET

**a. Types of scaling**

There are three main types of scaling in digital circuits:

* **Lithographic scaling:** This is the most common type of scaling, and it involves reducing the physical dimensions of the transistors on a chip. This can be done by using smaller transistors, or by using a process called "shrink-and-fix" that reduces the size of the transistors while maintaining their performance.
* **Voltage scaling:** This type of scaling involves reducing the supply voltage to the transistors. This can be done by using a lower-voltage process, or by using a technique called "overdriving" that temporarily increases the voltage to the transistors to improve their performance.
* **Frequency scaling:** This type of scaling involves increasing the clock frequency of the chip. This can be done by using a higher-frequency process, or by using a technique called "overclocking" that increases the clock frequency beyond its specified limit.

**b. Inverter V-I Characteristics**

An inverter is a digital circuit that inverts the input signal. This means that if the input signal is high, the output signal is low, and vice versa. The V-I characteristics of an inverter are the current that flows through the inverter as a function of the input voltage.

The V-I characteristics of an inverter can be divided into three regions:

* **Linear region:** In the linear region, the current through the inverter is proportional to the input voltage. This is the region where the inverter is most efficient.
* **Saturation region:** In the saturation region, the current through the inverter is no longer proportional to the input voltage. This is the region where the inverter is least efficient.
* **Breakdown region:** In the breakdown region, the current through the inverter increases rapidly with increasing input voltage. This is the region where the inverter can be damaged.
  1. **Parasitic in MOSFET**

Parasitic elements are unwanted components in a circuit that can affect its performance. In a MOSFET, there are several parasitic elements, including:

* **Source resistance:** This is the resistance of the source terminal of the MOSFET. It can affect the rise time of the output signal.
* **Drain resistance:** This is the resistance of the drain terminal of the MOSFET. It can affect the fall time of the output signal.
* **Channel resistance:** This is the resistance of the channel between the source and drain terminals of the MOSFET. It can affect the overall gain of the MOSFET.
* **Gate resistance:** This is the resistance of the gate terminal of the MOSFET. It can affect the charging and discharging time of the gate capacitor.
* **Gate capacitance:** This is the capacitance of the gate terminal of the MOSFET. It can affect the switching speed of the MOSFET.
* **Parasitic inductance:** This is the inductance of the wiring and interconnects in the MOSFET. It can cause ringing and overshoot in the output signal.

Q16 Explain following.

* 1. Static Power dissipation
  2. Dynamic Power dissipation
  3. Strong ‘0’ and weak ‘1’ in NMOS

**Static power dissipation**

Static power dissipation is the power that a circuit consumes when it is idle. This power is due to leakage currents that flow through the transistors even when they are not switching. Leakage currents are caused by imperfections in the manufacturing process of the transistors.

Static power dissipation is a major concern in modern electronic circuits, as it can significantly reduce battery life and increase heat generation. There are a number of techniques that can be used to reduce static power dissipation, including:

* **Using transistors with lower leakage currents.**
* **Power gating.**
* **Multi-threshold voltage (MTV) transistors.**

**Dynamic power dissipation**

Dynamic power dissipation is the power that a circuit consumes when it is switching. This power is due to the charging and discharging of the capacitances in the circuit. The dynamic power dissipation of a circuit is proportional to the switching frequency of the circuit and the capacitance of the circuit.

Dynamic power dissipation is a major concern in high-performance circuits, as it can limit the maximum operating frequency of the circuit. There are a number of techniques that can be used to reduce dynamic power dissipation, including:

* **Reducing the switching frequency of the circuit.**
* **Using low-power transistors.**
* **Using low-power design techniques, such as clock gating and power gating.**

\*\*Strong ‘0’ and weak ‘1’ in NMOS \*\*

In NMOS logic circuits, the output is typically connected to the drain of an NMOS transistor. When the NMOS transistor is turned on, the output is pulled down to ground (a strong ‘0’). When the NMOS transistor is turned off, the output is pulled up to the supply voltage (a weak ‘1’).

The reason for this is that the NMOS transistor is a pull-down transistor. This means that it is easier to pull the output down to ground than it is to pull it up to the supply voltage. This is because the NMOS transistor has a lower resistance when it is turned on than when it is turned off.

The weak ‘1’ in NMOS logic circuits can be a problem, as it can make the circuit more susceptible to noise. There are a number of techniques that can be used to improve the noise immunity of NMOS logic circuits, including:

* **Using pull-up resistors.**
* **Using totem-pole output stages.**
* **Using differential signaling.**

**Q17.** Explain following

1. Noise margin
2. Hot electron effect
3. Body Effect

noise margin

Noise margin is the maximum amount of noise that can be added to a signal without causing the signal to be misinterpreted. It is typically measured in dB (decibels). The higher the noise margin, the more resistant the signal is to noise.

There are two types of noise margin:

* Static noise margin (SNM): This is the maximum amount of DC noise that can be added to a signal without causing the signal to be misinterpreted.
* Dynamic noise margin (DNM): This is the maximum amount of AC noise that can be added to a signal without causing the signal to be misinterpreted.

The noise margin of a circuit is determined by a number of factors, including:

* The input voltage range of the circuit.
* The noise sources in the circuit.
* The propagation delay of the circuit.

Hot electron effect

The hot electron effect is a phenomenon that occurs in MOSFETs (metal-oxide-semiconductor field-effect transistors) when the drain-source voltage is high. When this occurs, the electrons in the channel of the MOSFET are accelerated by the high electric field and gain a lot of energy. These energetic electrons can then collide with the gate oxide, causing damage to the oxide. This damage can lead to a number of problems, including:

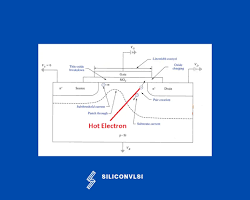
* Increased leakage current.
* Decreased threshold voltage.
* Increased gate oxide breakdown.

The hot electron effect is more likely to occur in MOSFETs with thin gate oxides and high drain-source voltages. There are a number of techniques that can be used to reduce the hot electron effect, including:

* Using thicker gate oxides.
* Using lower drain-source voltages.
* Using gate material with a higher work function.

Q18 Implement 2:1 mux using

* 1. Conventional Logic
  2. Transmission Gate

[Opens in a new window[](https://siliconvlsi.com/hot-electron-effect-in-mosfet/)siliconvlsi.com](https://siliconvlsi.com/hot-electron-effect-in-mosfet/)

Hot electron effect in transistor

Body effect

The body effect is a phenomenon that occurs in MOSFETs when the substrate voltage is not equal to the source voltage. When this occurs, the electric field between the substrate and the channel of the MOSFET attracts or repels the electrons in the channel, depending on the polarity of the substrate voltage. This attraction or repulsion can change the threshold voltage of the MOSFET.

The body effect is more pronounced in MOSFETs with thin channels and high substrate voltages. There are a number of techniques that can be used to reduce the body effect, including:

* Using thicker channels.
* Using lower substrate voltages.
* Using back-gate bias.

## Q19  Implement 2:1 mux using

## Conventional Logic

## Transmission Gate

edit

**Conventional Logic**

A 2:1 mux can be implemented using conventional logic by using a combination of AND gates, OR gates, and inverters. The following circuit shows an example of how to implement a 2:1 mux using conventional logic:

A0

|

\ /

AND

| \

|\_\_\_\_\MUX OUT

/ |

\ /

AND

| /

|\_\_\_\_/

|

\ /

INVERTER

| \

|\_\_\_\_\SELECT

/ |

\ /

INVERTER

| /

|\_\_\_\_/

|

VCC

In this circuit, the select line (SELECT) determines which input (A0 or A1) is passed to the output (MUX OUT). When SELECT is high, A0 is passed to the output. When SELECT is low, A1 is passed to the output.

**Transmission Gate**

A 2:1 mux can also be implemented using transmission gates. A transmission gate is a type of transistor circuit that can be used to selectively connect two signals together. The following circuit shows an example of how to implement a 2:1 mux using transmission gates:

A0

|

\ /

TG

| \

|\_\_\_\_\MUX OUT

/ |

\ /

TG

| /

|\_\_\_\_/

|

SELECT

In this circuit, the select line (SELECT) determines which input (A0 or A1) is passed to the output (MUX OUT). When SELECT is high, A0 is connected to the output. When SELECT is low, A1 is connected to the output.

Transmission gates are generally more efficient than conventional logic, as they do not require any power to be dissipated when they are not switching. However, transmission gates can be more difficult to design, as they can introduce unwanted signal reflections.

Q20 Explain enhancement type MOSFET and depletion type MOSFET

**Enhancement-type MOSFET**

An enhancement-type MOSFET (metal-oxide-semiconductor field-effect transistor) is a type of MOSFET that requires a gate voltage to turn on. When the gate voltage is zero, the MOSFET is off and no current can flow between the source and drain. When a positive gate voltage is applied, the channel between the source and drain is enhanced, allowing current to flow.

**Depletion-type MOSFET**

A depletion-type MOSFET is a type of MOSFET that is normally on and conducts current between the source and drain when the gate voltage is zero. When a negative gate voltage is applied, the channel between the source and drain is depleted, reducing the current flow.

## Q21 Explain Half adder &full adder in details

Half and full adders are fundamental building blocks in digital logic circuits. They are used to perform basic arithmetic operations such as addition.

Half Adder

A half adder is a combinational logic circuit that adds two one-bit binary digits (bits). It has two inputs, A and B, and two outputs, Sum (S) and Carry (C). The sum output represents the sum of the two input bits, while the carry output represents whether or not there was a carry generated during the addition.

The following table shows the truth table for a half adder:

|  |  |  |  |
| --- | --- | --- | --- |
| A | B | S | C |
| 0 | 0 | 0 | 0 |
| 0 | 1 | 1 | 0 |
| 1 | 0 | 1 | 0 |
| 1 | 1 | 0 | 1 |

drive\_spreadsheetExport to Sheets

As you can see, the sum output is simply the XOR of the two input bits. The carry output is 1 if and only if both input bits are 1.

Full Adder

A full adder is a combinational logic circuit that adds three one-bit binary digits (bits). It has three inputs, A, B, and Cin (carry-in), and two outputs, Sum (S) and Carry (Cout). The carry-in input represents the carry from the previous addition stage. The sum output represents the sum of the three input bits, while the carry-out output represents whether or not there was a carry generated during the addition.

The following table shows the truth table for a full adder:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | Cin | S | Cout |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 1 |

drive\_spreadsheetExport to Sheets

As you can see, the sum output is the XOR of the three input bits. The carry-out output is 1 if and only if two or more of the input bits are 1.

Applications

Half adders and full adders are used in a wide variety of digital circuits, including:

* Arithmetic logic units (ALUs)
* Counters
* Encoders
* Decoders
* Sum-of-products (SOP) implementations