Chapter 2:

Questions:

1. Clock skew:
   a. What is clock skew? Is it important? Why?
   b. What causes clock skew? How to reduce the clock skew?
   c. Is global clock skew or local clock skew important? Why?
2. Why is transition time measured from 10 to 90%?
3. Does improving gate delay also reduce wire delay?
4. What is the plus/minus of increasing the transistor width?
5. What are the components of power dissipation in a design? Which one is the most important?
6. What is crosstalk? When does crosstalk become significant? How to avoid/reduce crosstalk

Answers:

1. Clock skew

a. Clock skew refers to the variance in arrival times of clock signals across different parts of a digital circuit or system. It's crucially important in synchronous digital systems, where all components operate based on a common clock signal. Clock skew can lead to timing violations, where signals arrive too early or too late, causing errors in data transmission and potentially leading to system malfunction. **So, yes, clock skew is important because it directly affects the reliability and performance of digital systems.**

b. Several factors can cause clock skew, including variations in signal propagation delay, differences in routing lengths of clock signals, temperature variations, and manufacturing process variations. To reduce clock skew, designers employ techniques such as careful routing of clock signals to minimize path length differences, balancing load on different clock distribution paths, using clock tree synthesis algorithms, and implementing clock skew scheduling during timing optimization.

c. Both global and local clock skew are important, but their significance may vary depending on the specific context of the digital system. Global clock skew refers to the variance in arrival times of the clock signal across the entire system, which can impact the overall timing performance and may lead to system-wide timing violations. Local clock skew, on the other hand, pertains to the variance in arrival times within a particular region or block of the system. While global clock skew affects system-level timing, local clock skew can impact the timing within specific functional blocks or modules. Both need to be managed and minimized to ensure the reliable operation of the digital system.

2. Why is transition time measured from 10 to 90%?

Transition time is typically measured from the 10% to 90% points of the signal's amplitude rather than from 0% to 100% for several reasons:

1. Noise Immunity: Transition times measured from 10% to 90% are less sensitive to noise and signal jitter than measurements taken from 0% to 100%. The initial portion of the transition (0% to 10%) and the final portion (90% to 100%) are more susceptible to noise and can give inaccurate results.

2. Accuracy: The 10% to 90% range provides a more accurate representation of the time taken for the signal to transition between logic states. This range is considered the most stable and consistent part of the transition.

3. Standardization: It's a common industry practice to measure transition times from 10% to 90%, making it easier to compare specifications across different devices and systems. Standardizing the measurement methodology enhances clarity and ensures consistency in performance evaluation.

4. Repeatability: Transition times measured from 10% to 90% tend to be more repeatable across different measurements and test conditions compared to measurements taken from 0% to 100%. This repeatability is essential for reliable characterization and testing of digital circuits.

Overall, measuring transition times from 10% to 90% provides a robust and standardized approach that balances accuracy, noise immunity, and repeatability in characterizing signal transitions in digital circuits.

3. Does improving gate delay also reduce wire delay?

Improving gate delay does not directly reduce wire delay, but it can indirectly affect overall system performance, including reducing the impact of wire delay. Here's how:

Reduced Propagation Delay: Gate delay refers to the time it takes for a logic gate to produce output after receiving input. By optimizing gate design and using faster gate technologies, you can reduce gate delay. This reduction in gate delay means that signals spend less time inside individual gates, which can mitigate the effect of wire delay to some extent.

Balancing Timing: In a digital circuit, both gate delay and wire delay contribute to overall signal propagation time. By reducing gate delay, you may be able to balance the timing between gate and wire delays more effectively. This balance can help in meeting timing requirements and reducing the overall latency of the circuit.

Impact on Critical Paths: Gate delay improvements can have a significant impact on critical paths in a digital design. Critical paths are the longest paths in a circuit, typically composed of a

combination of gate and wire delays. By reducing gate delay on critical paths, you may indirectly reduce the overall propagation delay, improving system performance.

System-Level Optimization: Ultimately, optimizing gate delay is just one aspect of system-level optimization. To fully address wire delay, designers often employ techniques such as careful routing, buffering, and minimizing the length of critical interconnects. These strategies focus directly on mitigating the effects of wire delay in the overall system.

In summary, while improving gate delay does not directly reduce wire delay, it can contribute to overall system performance improvements, which may help mitigate the impact of wire delay and enhance the overall efficiency of digital circuits.

4. What is the plus/minus of increasing the transistor width?

Increasing the transistor width in digital circuit design has both advantages and disadvantages, often depending on the specific context and design goals. Here are the plus/minus aspects:

Advantages:

Increased Drive Strength: One of the primary advantages of increasing transistor width is the enhancement of drive strength. A wider transistor can source or sink more current, enabling it to drive larger loads or reduce signal propagation delays.

Improved Performance: With increased drive strength comes improved performance. Wider transistors can switch faster and drive signals with less voltage droop, leading to faster operation and better signal integrity.

Reduced Resistance: Increasing transistor width reduces the resistance between its source and drain terminals, which can help in reducing voltage drops and improving noise margins in the circuit.

Better Noise Margins: Wider transistors can better tolerate variations in supply voltage and process variations, resulting in improved noise margins and robustness against environmental factors.

Enhanced Current Handling: Wider transistors can handle higher currents without saturation, which is beneficial in applications where high current levels are required, such as power amplifiers or high-speed interfaces.

Disadvantages:

Area Overhead: Increasing transistor width consumes more silicon area on the integrated circuit (IC) chip. This can lead to larger chip sizes and increased manufacturing costs, especially for designs with many wide transistors.

Increased Capacitance: Wider transistors have larger parasitic capacitances, including gate capacitance and junction capacitance. This can lead to higher power consumption and slower operation due to increased charging and discharging times.

Reduced Flexibility: Using wide transistors limits the flexibility of the design, as it may restrict the placement of other components or routing options. This can make it challenging to optimize the layout and may limit the overall design space.

Risk of Hotspots: Wide transistors can concentrate current flow in certain regions, increasing the risk of localized heating and reliability issues, such as electromigration or transistor aging.

In summary, while increasing transistor width offers benefits in terms of improved performance and drive strength, it also comes with trade-offs such as increased area overhead, capacitance, and potential design restrictions. Designers must carefully weigh these factors to optimize transistor sizing for their specific applications and design constraints.

5. What are the components of power dissipation in a design? Which one is the most important?

Total Power = P_switch + P_short_circuit + P_leakage

Depending upon which application that which type of power will dominate -> corresponding optimization

6. What is crosstalk? When does crosstalk become significant? How to avoid/reduce crosstalk -> Slides

1. Loại kiểm tra (testing) nào cần kiến thức về cấu trúc bên trong và logic trong RTL verification? Black box testing

2. Logic synthesis là quá trình chuyển đổi HDL code thành một gate netlist được tối ưu hóa để mô tả phần cứng.

3. SiP vs SoC?
- SiP: Cho phép linh hoạt hơn về mặt thiết kế và tích hợp các thành phần
- SoC: hiệu năng tốt hơn, hình dạng và kích thước nhỏ hơn
-

4. Đoạn mã Verilog sau vi phạm lỗi nào trong thiết kế VLSI?

```
// reset data when rst is 0
// Update data for each rising clock when rst is 1
always@(posedge clk) begin
        if (rst == 1'b0) output <= 0;
        else output <= input;
end
```

=> Lỗi phép gán (assignment), phép gán non-blocking (<=) không nên được dùng trong always block

5. Đoạn mã Verilog nào sau đây thực hiện một multiplexer?
=> always@(in1, in2, select) out = (select) ? in1 : in2;

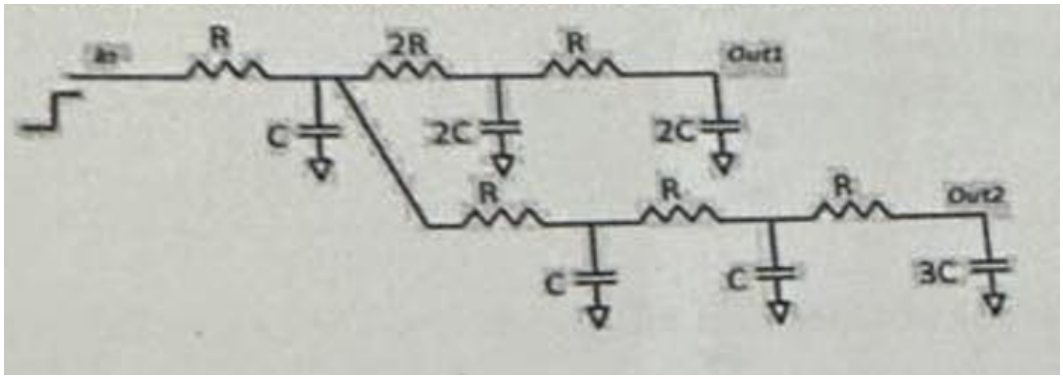
6. Đoạn mã Verilog HDL sau đây gặp lỗi gì?

```
input i, state;
reg i_next;

always@(i, state) begin
        if (state == 0) i_next = i + 1; end
always@(state) begin
        if (state == 1) i_next = state + 1; end
```

=> Không có lỗi ???

Đoạn mạch sau đây được sử dụng cho 2 câu hỏi tiếp theo



7. Độ trễ Elmore tại Out1 là:
T = R*(C + 2C + 2C + C + C + 3C) + 2R(2C + 2C) + R2C = 20RC

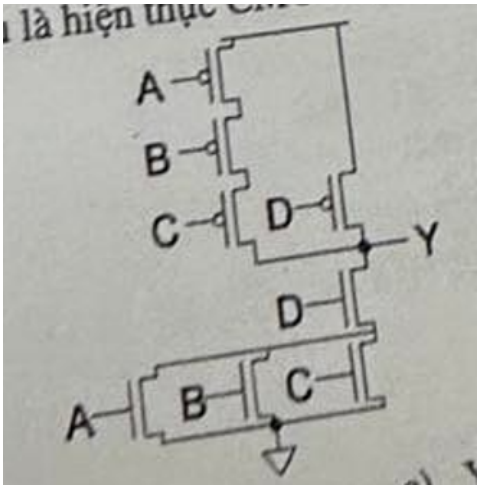
8. Độ trễ elmore tại Out2 là
T = R*(C + 2C + 2C + C + C + 3C) + R(C + 3C) + R(3C) = 22RC

9. Trong lĩnh vực vi mạch bán dẫn, mô hình kinh doanh nào sở hữu nhà máy sản xuất?
=> Foundry

10. Các mức độ tích hợp vi mạch (integration scale) được định nghĩa dựa trên tiêu chí nào?
=> Số lượng cổng logic tích hợp trên một chip

11. Loại transistor nào sử dụng công nghệ CMOS?
=> MOSFET

12. Sơ đồ mạch sau là hiện thực CMOS của hàm logic nào?



Đáp án C: Y = ((A + B + C).D)'

13. Cần ít nhất bao nhiêu MOSFET để hiện thực cổng NOR 3 ngõ vào (3-input) bằng công nghệ NMOS?
=> 4 MOSFET

14. Loại MOSFET nào được dùng trong mạng kéo xuống (pull-down network) trong mạch CMOS?
=> nMOS

15. Mức logic ngõ ra của flip-flop trong trạng thái metastability là?
=> Không thể xác định mức logic

16. Hiện tượng metastability khi xảy ra thì kéo dài trong bao lâu?

=> Không thể xác định

17. Điện năng (công suất) tiêu thụ của mạch cMOS phụ thuộc vào?
=> Công suất động và công suất tĩnh

18. Giải pháp nào sau đây có thể dùng để giảm điện năng tiêu thụ động của mạch?
=> Giảm tần số hoạt động của mạch, giảm mức điện áp hoạt động của mạch, tăng điện dung tải của mạch

19. Thuật ngữ PPAS nhắc đến 4 yếu tố nào?
=> Performance, power, area, cost ($)

20. Set-up time của một flip-flop được dùng
=> Để đảm bảo rằng ngõ vào của flip-flop ổn định ít nhất một khoảng thời gian = thời gian thiết lập trước cạnh xung clock

21. Định nghĩa hold time của một flip-flop là
=> Thời gian tối thiểu mà ngõ vào của flip-flop ổn định sau cạnh của clock

22. Để falling time và rising time tương đương nhau cho một cổng NOT là:
=> Thiết kế pMOS rộng hơn nMOS một chút

23. Để tăng clock skew trong VLSI, ta có thể sử dụng:
=> Clock spines, Clock mesh, Htree

24. Để giảm short-circuit power trong thiết kế VLSI, ta có thể:
=> Giảm độ rộng transistor, giảm điện áp, giảm Vt (điện áp ngưỡng)

25. Trong một mạng phân bổ clock, tín hiệu clock có độ lệch (skew) 100ps ở những chân FF clock cuối. Nếu chu kỳ của clock là 5ns thì phần trăm độ lệch của clock (clock skew) là bao nhiêu?
=> 100ps/5ns = 2%

26. Nguyên nhân chính của Crosswalk trong các mạch số là:
=> Không được che chắn đầy đủ (inadequate shielding)

27. Định nghĩa Fanout trong mạch số là:
=> Số lượng input tối đa mà một cổng logic có thể kết nối với

28. Mục đích của quá trình verification trong IC design là:
=> Chứng minh tính đúng đắn về chức năng (functional correctness) của thiết kế

29. Mục đích của static verification

=> Để xác minh (verify) thiết kế theo một số quy tắc được xác định trước mà không cần thực thi.

30. Mục đích của golden model trong RTL verification là:
=> Cung cấp một tài liệu tham khảo để so sánh kết quả của thiết kế

31. Kết quả của quá trình Functional Simulation là:
=> Waveform

32. Lợi thế của emulator so với simulator là:
=> Dễ dàng thiết lập cài đặt