**Week 1: Assignment 1**

**Question 1:**

Which of the following statement(s) is/are correct regarding a die fabricated on a silicon wafer?

A. We perform photolithography for creating transistors on a die before slicing that die out from the wafer  
B. We perform photolithography for creating transistors on a die after slicing that die out from the wafer  
C. We create metallic connections between transistors on a die before slicing that die out from the wafer  
D. We create metallic connections between transistors on a die after slicing that die out from the wafer

**Answer:** A, C  
**Score:** 1

**Explanation:**  
Photolithography and metallization are carried out while the wafer is intact to ensure precise layer alignment and uniformity across all dies. Once all transistors and interconnections are made, the wafer is sliced into individual dies.

**Question 2:**

Which of the following statements is true?

* The Foundry provides layout of an integrated circuit to the Fabless design companies for fabrication
* Foundry directly uses RTL obtained from fabless design companies for fabrication.
* Fabless design companies provide PDK to Foundries for fabrication.
* Nvidia is a fabless design company.

**Answer:** Nvidia is a fabless design company.  
**Score:** 1

**Explanation:**  
Fabless companies like Nvidia focus on design and outsource chip manufacturing to foundries such as TSMC or Samsung, allowing specialization and reduced infrastructure cost.

**Question 3:**

Which statement best describes the features of FPGA-based design compared to standard cell-based design?

* Lower design effort
* Inferior PPA
* Lower fixed cost
* All of the above

**Answer:** All of the above  
**Score:** 1

**Explanation:**  
FPGA designs are faster to implement and cheaper in terms of upfront cost but provide inferior performance, power, and area (PPA) compared to ASIC designs.

**Question 4:**

Among the following statements, which statement correctly describes the abstraction of RTL?

* RTL contains an algorithm typically written in C++
* RTL contains data flow descriptions typically written in Hardware Description Languages such as Verilog and VHDL.
* RTL contains transistor-level description typically written in SPICE
* RTL contains a layout typically written in GDS format

**Answer:** RTL contains data flow descriptions typically written in Hardware Description Languages such as Verilog and VHDL.  
**Score:** 1

**Explanation:**  
RTL represents the flow of data between registers and logic in digital systems. It is expressed using HDLs like Verilog or VHDL, forming the foundation for synthesis and implementation.

**Question 5:**

Which among the following statement(s) is/are true?

A. We use the same mask for all the metal layers while performing photolithography for a typical integrated circuit.  
B. Design description at a higher abstraction level contains more details than description at a lower abstraction level  
C. Full custom design, cell-based design and FPGA-based design are a few examples of design styles.

**Answer:** Only C  
**Score:** 1

**Explanation:**  
Each metal layer requires a separate mask, and higher abstraction levels contain less detail. Full custom, cell-based, and FPGA-based approaches represent different chip design methodologies.

**Question 6:**

Which of the following statements is correct regarding Behavioral Synthesis or High-level Synthesis?

* It converts an un-timed algorithm written in high-level languages to a circuit containing resistors, capacitors, and metallic interconnects with appropriate parameters.
* It converts an un-timed algorithm written in high-level languages to an RTL model.
* It converts an un-timed algorithm written in high-level languages to a transistor-level SPICE netlist.
* It converts an un-timed algorithm written in high-level languages to its corresponding layout.

**Answer:** It converts an un-timed algorithm written in high-level languages to an RTL model.  
**Score:** 1

**Explanation:**  
High-level synthesis tools translate behavioral descriptions (in C/C++) into RTL, enabling designers to work at a higher abstraction and automate hardware generation.

**Question 7:**

Which of the following can be the cost metrics for behavioral synthesis?

* Area
* Latency
* Power dissipation
* All of the above

**Answer:** All of the above  
**Score:** 1

**Explanation:**  
Behavioral synthesis optimizes trade-offs among key design metrics—chip area, latency, and power—to achieve efficient RTL implementations.

**Question 8:**

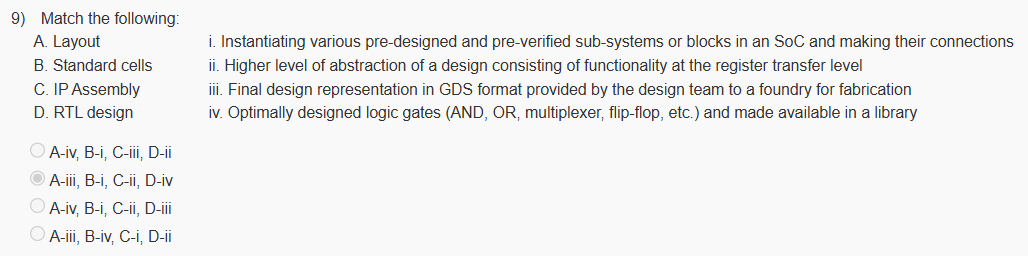
What are the advantages of designing a chip at a higher level of abstraction, such as RTL, compared to directly making a layout at the transistor level?

A. Making changes and exploring trade-offs can take less time and computational effort in RTL.  
B. Timing analysis is more accurate at the RTL compared to layout.  
C. It is more accurate to compute parasitic capacitance at RTL since it contains all hardware details.

**Answer:** Only A  
**Score:** 1

**Explanation:**  
Working at RTL allows faster design iterations and architectural exploration. Lower-level details like parasitics and precise timing are handled later during layout stages.

**Question 9:**

**Answer:** A-iii, B-iv, C-i, D-ii  
**Score:** 1

**Explanation:**  
Layout represents the physical GDS output; standard cells are basic pre-verified building blocks; IP assembly connects larger design blocks, and RTL defines register-level logic.

**Question 10:**

What is the advantage of employing behavioral synthesis in a design flow?

* The generated RTL often lacks readability and debuggability.
* It always increases the design effort for a chip.
* It allows automatic exploration of different possible RTLs for the same algorithm.
* All of the above

**Answer:** It allows automatic exploration of different possible RTLs for the same algorithm.  
**Score:** 1

**Explanation:**  
Behavioral synthesis helps designers automatically generate and compare multiple RTL architectures for the same algorithm, optimizing performance and resource utilization.

**Week 2: Assignment 2**

**Question 1:**

Arrange the following tasks involved in a typical process to create a mask for photolithography in their order of execution (first to last):

A. Etching is performed on the Chromium layer to get the required pattern  
B. A protective layer called a Pellicle is added on to the mask.  
C. Complex features of the given layout are converted to simpler shapes for mask writing.  
D. The features written on the mask are inspected for any defects.

**Answer:** C-A-D-B  
**Score:** 1

**Explanation:**  
First, complex layout patterns are simplified for mask writing. Then, the pattern is etched on the chromium layer. Afterward, the mask is inspected for defects, and finally, a pellicle layer is added for protection.

**Question 2:**

Which of the following statement(s) is/are true regarding the packaging/packages of a chip?

A. After packaging a die in a supporting case, it is directly shipped to the market without any further testing.  
B. Packages do not impact the timing behavior of a chip.  
C. Packages can impact the heat drawn from a die using heat sinks.  
D. Packages prevent any mechanical damage and corrosion on the chip.

**Answer:** Only C, D  
**Score:** 1

**Explanation:**  
Chip packaging helps dissipate heat effectively and protects the die from environmental damage. Testing and characterization are usually performed after packaging to ensure reliability.

**Question 3:**

Suppose a foundry fabricates a die with a yield of 80%. The foundry wants to deliver one million good dies to a customer. On a single wafer, 400 dies are produced. How many wafers does the foundry need to process to make the above delivery to the customer?

**Answer:** 3125  
**Score:** 1

**Explanation:**  
To obtain one million functional dies with an 80% yield, the foundry must fabricate more than one million dies. The total wafers needed = (1,000,000 / 0.8) / 400 = 3125 wafers.

**Question 4:**

Simulation techniques ensure the functional correctness of an RTL using:

* Test stimuli or Test patterns
* Formal Methods
* Deductions
* Property Checking

**Answer:** Test stimuli or Test patterns  
**Score:** 1

**Explanation:**  
Simulation uses test vectors or stimuli to verify that the RTL behaves as intended under various input conditions before synthesis or hardware implementation.

**Question 5:**

Arrange the following tasks of physical design in the sequence of execution (first to last).

A. Placement  
B. Routing  
C. Clock Tree Synthesis  
D. Floorplanning

**Answer:** D-A-C-B  
**Score:** 1

**Explanation:**  
Physical design begins with floorplanning, followed by placement of standard cells, clock tree synthesis to balance delays, and finally routing to connect all components.

**Question 6:**

How can defects manifest themselves in an integrated circuit?

* Short-circuit
* Open-circuit
* Change in gate delay
* All of the above

**Answer:** All of the above  
**Score:** 1

**Explanation:**  
Defects during fabrication can cause shorts, opens, or timing issues by altering physical or electrical properties, thereby affecting circuit performance or functionality.

**Question 7:**

Which of the following inputs are typically given to a physical design tool for creating its layout?

A. Expected timing behavior (such as clock frequency)  
B. Functionality of the design in the form of RTL design  
C. Abstract physical information of standard cells (physical library)  
D. Size and shape of the die

**Answer:** Only A, C, D  
**Score:** 1

**Explanation:**  
Physical design tools use timing constraints, standard cell libraries, and die dimensions to generate optimized layouts. The RTL description is used earlier during logic synthesis.

**Question 8:**

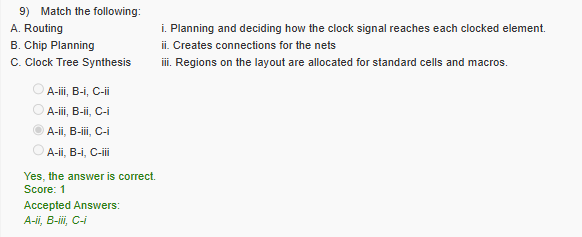
Which of the following statement(s) is/are true?

A. Netlist generated by a logic synthesis tool majorly contains instances of standard cells and their connections.  
B. Ports of a design in a netlist are used to communicate with the external world.  
C. A netlist cannot contain multiple instances of the same standard cell with different instance names.  
D. RTL design, libraries in Liberty format, and constraints in SDC format are typically given as inputs to a logic synthesis tool.

**Answer:** Only A, B, D  
**Score:** 1

**Explanation:**  
A synthesized netlist lists standard cell instances and their connections. It includes ports for I/O communication. Inputs like RTL, Liberty, and SDC constraints guide synthesis optimization.

**Question 9:**

**Answer:** A-ii, B-iii, C-i  
**Score:** 1

**Explanation:**  
Routing connects nets between cells, chip planning allocates layout regions for components, and clock tree synthesis ensures balanced clock distribution across the design.

**Question 10:**

Which task converts a netlist defined in terms of generic gates to a netlist defined in terms of standard cells?

* Hardware-software Partitioning
* Behavior Synthesis
* Routing
* Technology Mapping

**Answer:** Technology Mapping  
**Score:** 1

**Explanation:**  
Technology mapping transforms a generic gate-level netlist into one composed of standard cells from a specific technology library, enabling implementation on silicon.

**Week 3: Assignment 3**

**Question 1:**

A code coverage tool is typically employed to measure and report the code coverage of the given:

* SDC file
* RTL design
* Timing report
* Layout

**Answer:** RTL design  
**Score:** 1

**Explanation:**  
Code coverage tools analyze how much of the RTL code has been exercised during simulation. They measure coverage on statements, branches, and conditions within the HDL code to ensure sufficient testbench verification.

**Question 2:**

Which coverage metrics report design features that have been exercised/not exercised in the given coverage model during simulation?

* Code coverage
* Functional coverage
* Branch coverage
* Toggle coverage

**Answer:** Functional Coverage  
**Score:** 1

**Explanation:**  
Functional coverage tracks how well design functionality (as defined in the verification plan) has been exercised during simulation, focusing on design intent rather than structural code elements.

**Question 3:**

Which of the following statement(s) is/are true?

A. Functions in a Verilog code can only have delays modelled by posedge, but not by negedge  
B. Functions in a Verilog code can call another function but not a task.  
C. Tasks in a Verilog code can call another task but not a function.  
D. Tasks in a Verilog code can have delays modelled by both posedge and negedge

**Answer:** Only B, D  
**Score:** 1

**Explanation:**  
Verilog functions are limited to computations without timing or event controls and can call other functions only. Tasks, however, can contain timing controls such as posedge or negedge and can invoke both tasks and functions.

**Question 4:**

How can we specify connections for an instantiated module in a Verilog code?

A. By specifying connections in implicit order as declared in the instantiated module.  
B. By specifying connections explicitly using the name of the ports of the instantiated module.

**Answer:** Both A and B  
**Score:** 1

**Explanation:**  
Verilog supports both **positional** and **named** port mapping. In positional mapping, connections are made in order, while named mapping uses explicit port names for better readability and flexibility.

**Question 5:**

Which of the following are distinct features of the Hardware Description Languages (like Verilog) and are not present in C?

A. Integer data type  
B. Concurrency  
C. Notion of time  
D. Electrical characteristics (such as driver strength)

**Answer:** Only B, C, D  
**Score:** 1

**Explanation:**  
HDLs like Verilog support concurrency, timing controls, and modeling of hardware behavior (such as drive strength), unlike procedural programming languages such as C, which execute sequentially.

**Question 6:**

What is the internal representation of the integer 8'bz101?

* 0000 z101
* z101 z101
* zzzz z101
* z000 0101

**Answer:** zzzz z101  
**Score:** 1

**Explanation:**  
The value 8'bz101 represents an 8-bit number where the most significant four bits are high-impedance (‘z’) and the remaining bits are 0101. Hence, its internal representation is zzzz0101.

**Question 7:**

Assume a = 3'b101, b = 4'b0011, then what is the output of {a,b}?

* 7'b1010011
* 4'b0011
* 4'b0111
* 7'b0011101

**Answer:** 7'b1010011  
**Score:** 1

**Explanation:**  
The concatenation operator {} in Verilog joins bit vectors. Hence, {a,b} = {3’b101, 4’b0011} = 7’b1010011.

**Question 8:**

The order of execution of various events in a given time slot in Verilog is governed by:

* Race Event Queue
* Stratified Event Queue
* Coverage Event Queue
* Circular Verification Queue at a given simulation time

**Answer:** Stratified Event Queue  
**Score:** 1

**Explanation:**  
Verilog uses a **stratified event queue** mechanism that defines the order of event execution (active, inactive, NBA, monitor regions) within each simulation time slot to ensure deterministic behavior.

**Question 9:**

Which of the following are valid keywords in Verilog language?

A. begin  
B. initial  
C. always  
D. mycomments

**Answer:** Only A, B, C  
**Score:** 1

**Explanation:**  
begin, initial, and always are reserved Verilog keywords used to define procedural blocks. mycomments is not a Verilog keyword.

**Question 10:**

How can the default value of a parameter in a parameterized module be overridden for a given instantiation in Verilog?

* It can be overridden by providing a non-default value outside all of the modules.
* It can be overridden during instantiation by providing a non-default value using the keyword PARAMETER.
* It can be overridden during instantiation by providing non-default values within #().
* It can never be overridden.

**Answer:** It can be overridden during instantiation by providing non-default values within #().  
**Score:** 1

**Explanation:**  
Parameters in Verilog can be modified during module instantiation using the syntax #(parameter\_value), allowing reuse of the same module with different configurations.

**Week 4: Assignment 4**

**Question 1:**

An initial block in a Verilog code has a blocking assignment with a delay specification of 2 ns. Typically, what will be the synthesis result of this assignment?

* It will be synthesized into one latch
* It will be synthesized into two latches
* It will be synthesized into a flip-flop and a latch
* It will be ignored by the synthesis tool

**Answer:** It will be ignored by the synthesis tool  
**Score:** 1

**Explanation:**  
The initial block and timing delays are used only for simulation purposes. Synthesis tools ignore such constructs since they don’t correspond to hardware structures.

**Question 2:**

A Boolean function *y* of three variables {x₁, x₂, x₃} is as follows:

y = x₁′x₂′x₃′ + x₁′x₂x₃′ + x₁x₂x₃′ + x₁x₂x₃

In the hypercube representation, which corners will have a value of 1 (the corners are represented as values of x₁x₂x₃ in the following options)?

* 000, 010, 110, 111
* 111, 101, 001, 000
* 000, 101, 001, 111
* 111, 010, 110, 000

**Answer:** 000, 010, 110, 111  
**Score:** 1

**Explanation:**  
Each product term in the function corresponds to a vertex (corner) in the 3D Boolean cube where the function evaluates to logic ‘1’. Mapping these minterms results in the corners 000, 010, 110, and 111.

**Question 3:**

Which of the following statements is/are true regarding the Heuristic Minimizer designed for two-level logic minimization?

A. It is guaranteed to be the minimum.  
B. It never terminates (ends).  
C. It typically starts with an initial cover, and the solution is iteratively improved by applying some operators to it.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above

**Answer:** Only C  
**Score:** 1

**Explanation:**  
Heuristic minimization methods such as Espresso begin with an initial logic cover and iteratively refine it to approach an optimal solution, though not guaranteed to be the absolute minimum.

**Question 4:**

Consider the following statements about tools ICARUS, GTKWave, and COVERED used in simulation-based verification:

A. We can use the ICARUS tool to write a VCD file containing simulation results for a given Verilog design and testbench.  
B. VCD files can never be used by GTKWave to show the simulation results.  
C. COVERED tool is to measure the size covered by a given layout on the wafer.

Which of the above statements are correct?

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** Only A  
**Score:** 1

**Explanation:**  
ICARUS Verilog generates simulation output as a VCD file. GTKWave reads VCD files to display waveforms, while COVERED is used for code coverage, not layout measurement.

**Question 5:**

What is the prime motivation for “resource unsharing” while synthesizing an RTL code?

* Reduce the area of a circuit
* Reduce the delay of a long or critical path
* Reduce the testability of a circuit
* Reduce the reliability of a circuit

**Answer:** Reduce the delay of a long or critical path  
**Score:** 1

**Explanation:**  
Resource unsharing replicates hardware resources to allow parallel operations, which minimizes critical path delay and improves overall performance at the cost of higher area.

**Question 6:**

Which of the following statements are examples of typical compiler optimizations used in RTL synthesis?

* A dead code is made alive
* An arithmetic operation is replaced by an equivalent, more costly arithmetic operation
* An expression is replaced by a constant if possible
* All of the above

**Answer:** An expression is replaced by a constant if possible  
**Score:** 1

**Explanation:**  
Constant propagation replaces expressions with known constant values during synthesis, reducing unnecessary logic and simplifying hardware implementation.

**Question 7:**

Which of the following statements is/are correct?

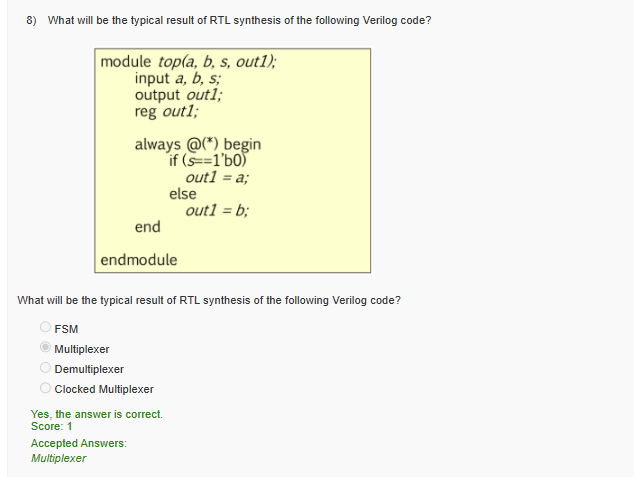
A. While performing RTL synthesis of a Verilog ‘for loop’, the RTL synthesis tool typically does not need to know the number of iterations during compile time for proper synthesis.  
B. The critical path before and after resource sharing always remains the same in RTL synthesis.  
C. Functions in Verilog are always synthesized to a sequential logic block with one output (scalar or vector).

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** None of the above

**Explanation:**  
All statements are incorrect: the synthesis tool must know loop bounds at compile time, resource sharing can change the critical path, and Verilog functions generally map to combinational logic, not sequential blocks.

**Question 8:**

****

**Answer:** Multiplexer

**Explanation:**  
The always block selects one of two inputs (a or b) based on a control signal s. This conditional assignment synthesizes to a 2:1 multiplexer in hardware.

**Question 9:**

While modeling a combinational circuit using an always block in a Verilog code, which of the following methods can avoid unintentional inference of latches?

* Use the negedge of the clock rather than the posedge of the clock
* Use a default statement to write to a variable and cover all the possible paths in a “case statement”
* Use an initial block inside the always block
* Use wire instead of reg for a variable

**Answer:** Use a default statement to write to a variable and cover all the possible paths in a “case statement”  
**Score:** 1

**Explanation:**  
Incomplete assignment in an always block causes latch inference. Assigning default values and covering all cases ensures proper combinational logic synthesis.

**Question 10:**

Which of the following statements is/are true regarding RTL synthesis?

A. During parsing, typically, a hierarchical data structure called a parse tree or syntax tree is built.  
B. During elaboration, modules with distinct interfaces can be created for each distinct set of parameters used during instantiation.  
C. Some Verilog constructs, such as fork-join, force-release, can be ignored by an RTL synthesis tool.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** All of the above  
**Score:** 1

**Explanation:**  
Parsing generates the syntax tree structure of the HDL code. During elaboration, parameterized modules are instantiated properly, and constructs unsupported by synthesis (like fork-join) are safely ignored.

**Week 5: Assignment 5**

**Question 1:**

We can build a Binary Decision Tree for a Boolean function using:

* Fermat’s Last Theorem
* Pythagoras Theorem
* Quine’s Theorem
* Shannon’s Expansion Theorem

**Answer:** Shannon’s Expansion Theorem

**Explanation:**  
Binary Decision Trees (or BDDs) are constructed using **Shannon’s Expansion Theorem**, which expresses a Boolean function in terms of one variable and its complement, forming the basis of decision diagrams.

**Question 2:**

Which of the following statements is/are true regarding simulation-based verification?

A. Design correctness can be proven using simulation-based verification for all non-trivial industrial designs.  
B. Test vectors or stimuli are required for simulation-based verification.  
C. The computational resources needed for simulation-based verification will typically increase with the number of test vectors or stimuli.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** Only B, C

**Explanation:**  
Simulation-based verification checks design behavior using test stimuli. While it cannot formally prove correctness, its accuracy and computational cost scale with the number of test cases.

**Question 3:**

Which of the following statements is/are true?

A. If a Boolean function representation is canonical, two equivalent Boolean functions are represented identically.  
B. ROBDD is a canonical representation of a Boolean function.  
C. The size of an ROBDD for a Boolean function does not depend on the variable order.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** Only A, B

**Explanation:**  
Canonical forms uniquely represent equivalent Boolean functions. **ROBDDs** are canonical, meaning two identical functions yield identical graphs — but their size depends on the chosen variable order.

**Question 4:**

Among the following statements, which statement is correct about state minimization?

* State minimization involves reducing the number of bits in the FSM’s representation by increasing the number of states.
* State minimization involves deriving an FSM that has the minimum number of states and exhibits behavior totally different from the original FSM.
* State minimization involves deriving an FSM that has the minimum number of states and exhibits the same behavior as the original FSM.
* State minimization involves a transformation that allows an FSM to generate random outputs.

**Answer:** State minimization involves deriving an FSM that has the minimum number of states and exhibits the same behavior as the original FSM.

**Explanation:**  
FSM state minimization merges equivalent states while preserving identical input-output behavior, reducing hardware complexity without changing circuit functionality.

**Question 5:**

To represent an FSM with *nₛ* states, how many bits are needed for one-hot encoding?

* log₂(nₛ)
* nₛ
* (nₛ)²
* nₛ + 1

**Answer:** nₛ

**Explanation:**  
In **one-hot encoding**, each state is represented by a dedicated flip-flop bit, where only one bit is high at a time. Hence, *nₛ* bits are required for *nₛ* states.

**Question 6:**

Which of the following statements is/are true?

A. The circuit area of an FSM implementation always reduces if the number of states is increased.  
B. An FSM can be represented pictorially using a state diagram.  
C. We often represent sequential circuits using an FSM in model checking.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** Only B, C

**Explanation:**  
State diagrams graphically represent FSM behavior. FSMs are also widely used in **model checking** to verify temporal properties of sequential systems.

**Question 7:**

Which of the following statements is/are correct?

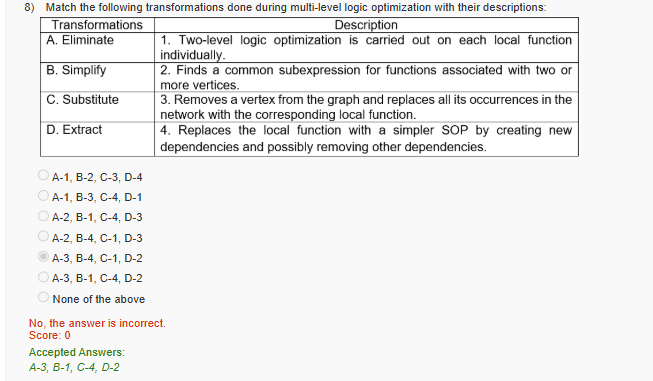
A. Sum of Product (SOP) and Product of Sum (POS) are examples of two-level logic.  
B. Two-level logic will typically have a greater number of logic gates between input and output compared to its equivalent multi-level logic.  
C. Two-level logic is always more compact (having a smaller number of gates) than its equivalent multi-level logic.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** Only A

**Explanation:**  
SOP and POS are classical examples of **two-level logic**, involving a single layer of AND or OR gates followed by a combining gate. Multi-level logic typically reduces gate count and depth.

**Question 8:**

****

**Answer:** A-3, B-1, C-4, D-2

**Explanation:**  
In multi-level optimization:

* **Eliminate** removes redundant nodes,
* **Simplify** optimizes local functions,
* **Substitute** replaces functions to simplify dependencies, and
* **Extract** identifies and shares common subexpressions across the design.

**Question 9:**

Which of the following files are typically given as input to Yosys for logic synthesis?

A. RTL code (such as *top.v*)  
B. VCD file (such as *top.vcd*)  
C. Library file in Liberty format (such as *NangateOpenCellLibrary\_typical.lib*)

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** Only A, C

**Explanation:**  
Yosys performs logic synthesis using the **RTL description** (Verilog) and **cell library (.lib)** files. VCD files are used for waveform viewing or coverage, not synthesis.

**Question 10:**

Which of the following can be used for logic optimization of a digital circuit?

A. Controllability Don’t Cares  
B. Satisfiability Don’t Cares  
C. Observability Don’t Cares

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** All of the above

**Explanation:**  
All three types of don’t-cares — controllability, satisfiability, and observability — are used to simplify Boolean logic during optimization, improving area and performance efficiency.

**Week 6: Assignment 6**

**Question 1:**

Why are libraries used for delay/power computation instead of direct transistor-level SPICE simulation in cell-based design?

* To improve accuracy
* To save runtime
* Delay cannot be computed using SPICE simulation
* Power dissipated cannot be computed using SPICE simulation

**Answer:** To save runtime

**Explanation:**  
SPICE simulations at the transistor level are extremely accurate but computationally expensive. Pre-characterized libraries allow faster delay and power estimation with minimal accuracy loss, saving significant runtime in large designs.

**Question 2:**

Among the following options, on what factor(s) does the setup/hold time of a D flip-flop typically depend?

A. Slew at the D-pin  
B. Slew at the Q-pin  
C. Slew at the clock pin

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** Only A, C

**Explanation:**  
Setup and hold times depend mainly on the **input data slew (D-pin)** and **clock slew**, as they influence the timing window and edge transition. The output slew (Q-pin) does not directly affect timing requirements.

**Question 3:**

Which of the following statements are correct for Sequential Equivalence Checking (SEC) and Combinational Equivalence Checking (CEC)?

A. SEC is computationally more difficult than CEC  
B. CEC explores all reachable states for establishing equivalence  
C. SEC needs to tackle the challenge of state explosion

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** Only A, C

**Explanation:**  
Sequential Equivalence Checking (SEC) compares sequential designs and must consider state transitions, making it more complex and prone to **state explosion**, unlike CEC, which only compares combinational logic.

**Question 4:**

Which of the following statements are correct about the miter circuits created during Combinational Equivalence Checking of two sequential circuits?

A. A miter is a combinational circuit.  
B. A miter is derived using the corresponding combinational logic cones of both the circuits.  
C. A miter typically has an XOR gate at its output.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** All of the above

**Explanation:**  
A **miter circuit** is a combinational structure formed by connecting the outputs of two circuits under test through XOR gates. If the XOR output is always 0, both circuits are functionally equivalent.

**Question 5:**

Which of the following statements is/are true?

A. Model checking can easily tackle the challenge of the state explosion problem for non-trivial cases by enumerating all possible states and verifying at each state.  
B. Bounded model checking tries to find a counterexample in a finite number of clock cycles from the initial state.  
C. Bounded model checking is complete because if a counterexample is not found in 2 clock cycles from the initial state, it implies that it will never be found.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** Only B

**Explanation:**  
**Bounded Model Checking (BMC)** verifies correctness within a limited number of cycles. It searches for counterexamples up to a bound but does not guarantee completeness if none are found.

**Question 6:**

Which of the following statements is correct for a Boolean function F represented in Conjunctive Normal Form (CNF)?

* If a variable assignment exists such that ANY clause in F evaluates to 0, the function is unsatisfiable.
* If a variable assignment exists such that ANY clause in F evaluates to 1, the function is satisfiable.
* If a variable assignment exists such that ALL clauses in F evaluate to 0, the function is unsatisfiable.
* If a variable assignment exists such that ALL clauses in F evaluate to 1, the function is satisfiable.

**Answer:** If a variable assignment exists such that ALL clauses in F evaluate to 1, the function is satisfiable.

**Explanation:**  
A CNF function is satisfiable if **every clause evaluates to logic 1** for some assignment of input variables. Otherwise, the function is unsatisfiable.

**Question 7:**

Which of the following statements is correct about the Hold Time of a flip-flop?

* It is the minimum amount of time the DATA signal should be held steady before the CLOCK edge.
* The Hold Time of a flip-flop is always greater than the setup time of a flip-flop.
* It is the minimum amount of time the DATA signal should be held steady after the CLOCK edge.
* The Hold Time of a flip-flop does not depend on clock slew.

**Answer:** It is the minimum amount of time the DATA signal should be held steady after the CLOCK edge.

**Explanation:**  
Hold time ensures that the data input remains stable immediately after the active clock edge to allow proper latching of the signal.

**Question 8:**

Which of the following statements is/are correct about the open-source EDA tool Yosys?

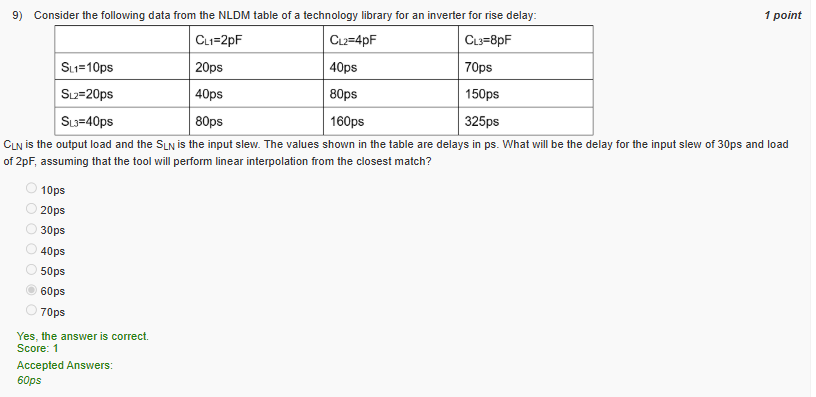
A. Yosys is used to perform detailed routing.  
B. Yosys is used to perform technology mapping.  
C. Yosys is used to perform logic optimization, such as resource sharing.

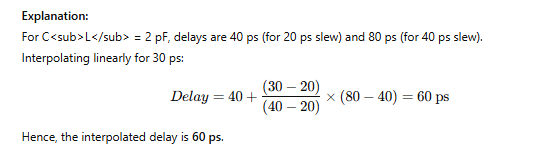
* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** Only B, C

**Explanation:**  
Yosys is an open-source synthesis tool that performs **logic optimization** and **technology mapping**. It does not handle physical design tasks such as placement or routing.

**Question 9:**

**Answer:** 60 ps

****

**Question 10:**

Which of the following statements is/are correct?

A. If a SAT solver encounters a conflict and no more backtracking is possible, the function is satisfiable.  
B. In CEC, all the miters of two equivalent circuits are unsatisfiable.  
C. In CEC, input/output ports and sequential circuit elements are considered as match points.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All of the above
* None of the above

**Answer:** Only B, C

**Explanation:**  
In Combinational Equivalence Checking (CEC), each miter circuit compares two logic cones. For equivalent circuits, miters are **unsatisfiable**, and comparison points include I/O ports and sequential elements.

**Week 7: Assignment 7**

**Question 1:**

The required time at the capture flip-flop for setup analysis depends on which of the following quantities in a synchronous circuit?

A. Setup time of the launch flip-flop  
B. Setup time of the capture flip-flop  
C. Clock-to-Q delay of the launch flip-flop  
D. Clock period

* Only A, B
* Only A, C
* Only A, D
* Only B, C
* Only B, D
* Only C, D

**Answer:** Only B, D

**Explanation:**  
For setup analysis, the required time at the capture flip-flop depends on the **setup time of the capture flip-flop** and the **clock period**, which defines when the data must arrive before the next clock edge.

**Question 2:**

The hold requirement constraint depends on which of the following quantities in a typical synchronous circuit?

A. Clock Period  
B. Setup Time of the capture flip-flop  
C. Hold Time of the launch flip-flop  
D. Hold Time of the capture flip-flop

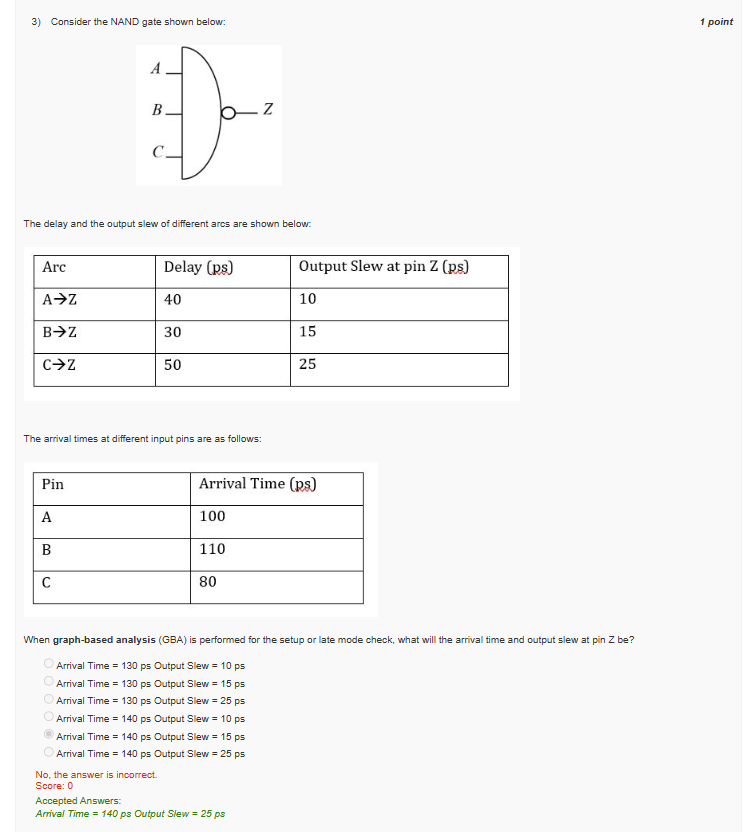
* Only A
* Only B
* Only C
* Only D
* Only A, B
* Only A, C
* Only A, D
* Only B, C
* Only B, D
* Only C, D

**Answer:** Only D

**Explanation:**  
Hold analysis ensures that the data remains stable at the input of the **capture flip-flop** for its required **hold time** after the clock edge. Hence, only the capture flip-flop’s hold time affects this check.

**Question 3:**

Consider the NAND gate shown below.

When graph-based analysis (GBA) is performed for the setup or late mode check, what will the arrival time and output slew at pin Z be?

**Answer:** Arrival Time = 140 ps, Output Slew = 25 ps

**Explanation:**  
In GBA for setup (late mode), the latest arrival time is considered. The **C → Z** path dominates since 80 + 50 = 130 ps, and **B → Z** gives 110 + 30 = 140 ps (maximum). Therefore, the final arrival is 140 ps with an output slew of 25 ps (the largest slew).

s

**Question 4:**

For the same question above (Q3), what will the arrival time and the output slew be at pin Z when **graph-based analysis (GBA)** is performed for the hold or early mode check?

**Answer:** Arrival Time = 130 ps, Output Slew = 10 ps

**Explanation:**  
For hold (early mode), the **earliest arrival** is considered. The path through **A → Z** gives 100 + 40 = 140 ps, and **C → Z** gives 80 + 50 = 130 ps (minimum). Hence, the earliest arrival is 130 ps with the smallest output slew of 10 ps.

**Question 5:**

For the same NAND gate (Q3), what will the arrival time and output slew be at pin Z when **path-based analysis (PBA)** is performed for the hold check or early mode check through the arc **B → Z**?

**Answer:** Arrival Time = 140 ps, Output Slew = 15 ps

**Explanation:**  
In PBA, individual paths are analyzed without merging timing arcs.  
For **B → Z**, arrival = 110 + 30 = 140 ps with the corresponding slew of 15 ps.

**Question 6:**

Consider the following synchronous circuit.

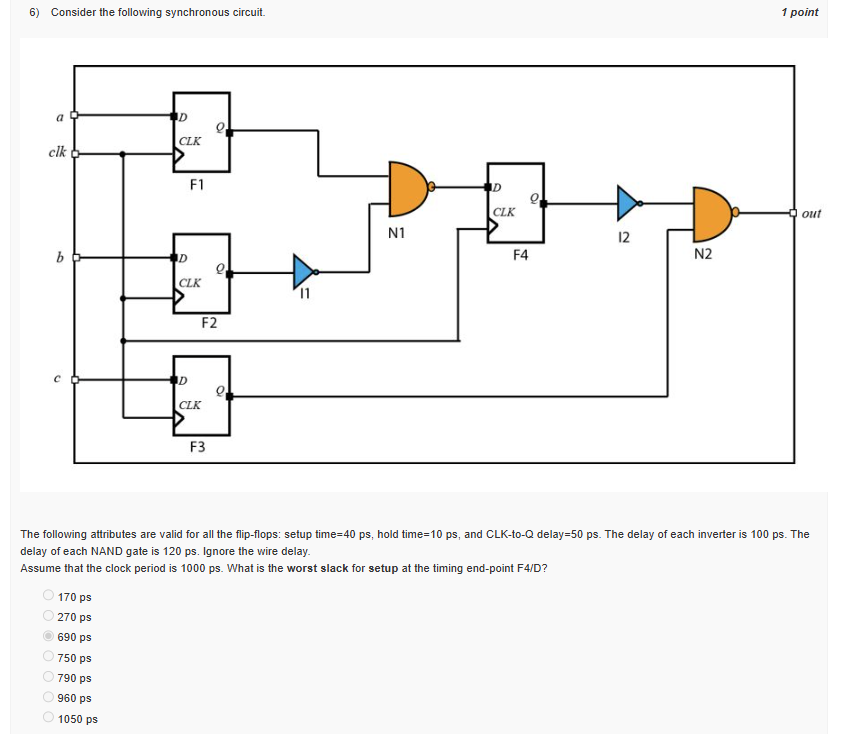
The flip-flop parameters are:

* Setup time = 40 ps
* Hold time = 10 ps
* CLK-to-Q delay = 50 ps
* NAND gate delay = 120 ps
* Inverter delay = 100 ps
* Clock period = 1000 ps  
  (Wire delay ignored)

What is the **worst slack for setup** at the timing end-point F4/D?

**Answer:** 690 ps

**Explanation:**  
Setup slack = Required Time – Arrival Time  
= Clock period – (CLK-to-Q + path delay + setup time).  
After substituting values, total delay ≈ 310 ps → Slack = 1000 – 310 = **690 ps**.



**Answer:** 690 ps

**Explanation:**  
Setup slack = Required Time – Arrival Time  
= Clock period – (CLK-to-Q + path delay + setup time).  
After substituting values, total delay ≈ 310 ps → Slack = 1000 – 310 = **690 ps**.

**Question 7:**

For the same circuit as above, what is the **worst slack for hold** at the timing end-point F4/D?

**Answer:** 160 ps

**Explanation:**  
Hold slack = (CLK-to-Q + path delay) – hold time.  
A positive slack of 160 ps ensures that the data remains stable after the clock edge, satisfying hold requirements.

**Question 8:**

Which of the following options does the delay calculation for a given stage performed during Static Timing Analysis depend on?

A. Driver model of the driving cell  
B. Yield model for the technology  
C. Receiver model of the driven cell

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only A, C

**Explanation:**  
The delay at a stage depends on the **driver characteristics** (output resistance and transition strength) and the **receiver load** (input capacitance). The yield model is unrelated to timing computation.

**Question 9:**

Which tool will you employ to compute the setup slack of a synchronous circuit?

* Icarus
* VCS
* Conformal
* Bambu
* OpenSTA

**Answer:** OpenSTA

**Explanation:**  
**OpenSTA** is an open-source static timing analysis tool used for setup, hold, and slack calculation in synchronous digital designs.

**Question 10:**

Consider the following statements:

A. A negative slack for setup in static timing analysis indicates a timing violation.  
B. Slack for setup can be increased by increasing the clock period.  
C. Slack for hold is independent of the clock period.

Which of the statements are true?

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** All A, B, C

**Explanation:**  
A negative setup slack indicates violation. Increasing the clock period relaxes setup timing, improving slack. Hold timing depends only on local delay differences and is independent of the clock period.

**Week 8: Assignment 8**

**Question 1:**

Which of the following is/are valid SDC commands?

A. create\_clock  
B. generate\_clock  
C. create\_generated\_clock

* only A
* only B
* only C
* only A, B
* only B, C
* only A, C
* all A, B, C

**Answer:** only A, C

**Explanation:**  
SDC defines clocks using create\_clock and derived clocks with create\_generated\_clock. There is no generate\_clock command in SDC.

**Question 2:**

Which of the following is the SDC command that instructs an STA tool to perform a setup and hold check at the **output port**?

* set\_output\_delay
* set\_analysis\_check
* set\_setup\_check
* set\_hold\_check
* set\_both\_check

**Answer:** set\_output\_delay

**Explanation:**  
set\_output\_delay constrains timing at an output port relative to a reference clock, so STA performs setup/hold checks against that external timing requirement.

**Question 3:**

Consider the following statements about timing-driven optimizations:

A. **Resizing** replaces a standard cell with another cell of the same logic and pin interface but different drive/area.  
B. **Retiming** moves logic across flip-flops.  
C. **Fanout optimization** connects high-fanout nets to ground to force zero.

* only A
* only B
* only C
* only A, B
* only B, C
* only A, C
* all A, B, C

**Answer:** only A, B

**Explanation:**  
Resizing changes drive strength; retiming redistributes registers to balance delays. Forcing high-fanout nets to ground is **not** a timing optimization.

**Question 4:**

Consider the following statements about technology mapping in logic synthesis:

A. It may change the logical functionality to minimize area.  
B. PPA of the mapped netlist does **not** depend on the cell library attributes.  
C. It converts a generic gate netlist into one using standard cells from the technology library.

* only A
* only B
* only C
* only A, B
* only B, C
* only A, C
* all A, B, C

**Answer:** only C

**Explanation:**  
Technology mapping preserves functionality while selecting real cells. PPA strongly depends on the library (drive strengths, Vt options, etc.).

**Question 5:**

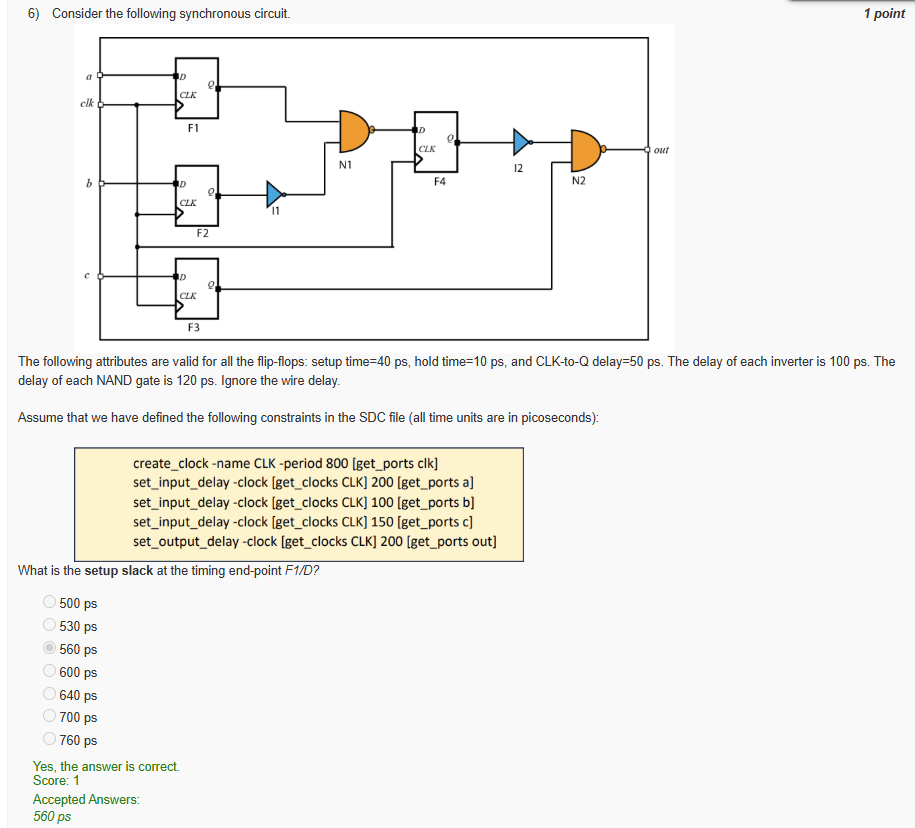
Which optimization technique is typically **NOT** targeted for improving timing?

* Power Gating
* Restructuring
* Retiming
* Resizing

**Answer:** Power Gating

**Explanation:**  
Power gating targets leakage power reduction; the others are timing-driven techniques.

**Question 6:**

Consider the synchronous circuit (attributes: setup=40 ps, hold=10 ps, CLK→Q=50 ps; each inverter=100 ps; each NAND=120 ps; ignore wire). SDC constraints:

**Q6)**

**Answer:** 560 ps

**Explanation:**  
Required time = clock period (800). Arrival time to F1/D is dominated by input delay and small logic on that path; computing Slack = Required − Arrival yields **560 ps** per the provided constraints.

**Question 7:**

For Q6, what is the **hold slack** at **F1/D**?

* −200 ps
* −190 ps
* −180 ps
* 0 ps
* 180 ps
* 190 ps
* 200 ps

**Answer:** 190 ps

**Explanation:**  
Hold slack = (data arrival after launch edge) − (required hold). With CLK→Q + short path delay exceeding the hold time, the margin is **190 ps**.

**Question 8:**

For Q6, what is the **worst setup slack** at the timing end-point **port**out?

* 250 ps
* 330 ps
* 350 ps
* 430 ps
* 600 ps
* 700 ps
* 760 ps
* 800 ps

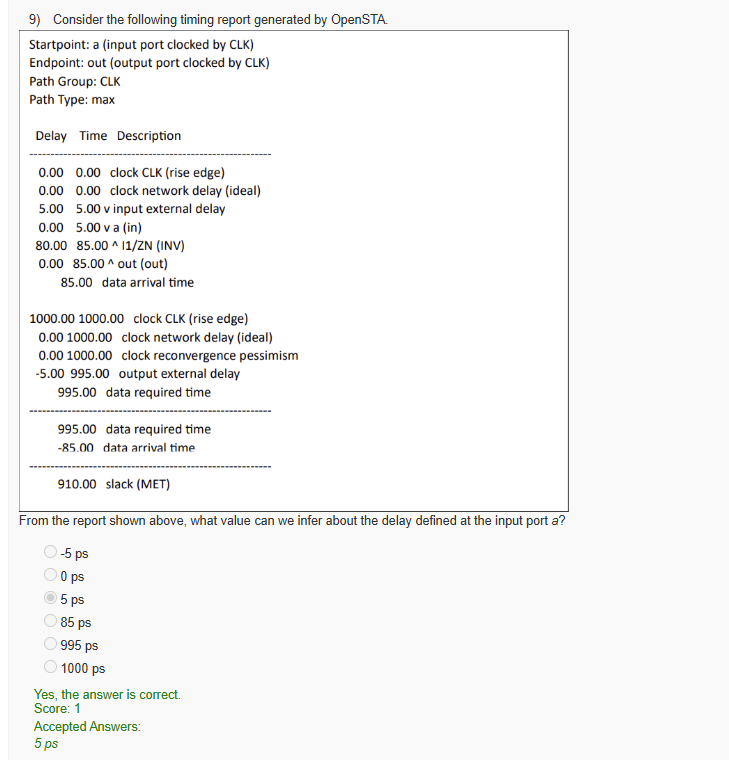
**Answer:** 330 ps

**Explanation:**  
Considering input delays, internal gate delays (NAND/INV), and set\_output\_delay 200, the longest path to out leaves **330 ps** of margin.

**Question 9:**

From the OpenSTA timing report (startpoint a → endpoint out), what value can we infer about the **input delay at port**a?

* −5 ps
* 0 ps
* 5 ps
* 85 ps
* 995 ps
* 1000 ps

**Answer:** 5 ps

**Explanation:**  
The report shows v input external delay of **5.00**, meaning set\_input\_delay for a is 5 ps (numbers shown are in the report’s internal units).

**Question 10:**

From the same report, what is the **arrival time at output port**out?

* −5 ps
* 0 ps
* 5 ps
* 85 ps
* 995 ps
* 1000 ps

**Answer:** 85 ps

**Explanation:**  
The report lists out (out) time as **85.00 ps**, which is the data arrival at the output.

**Week 9: Assignment 9**

**Question 1:**

Consider the following statements about power dissipated in a CMOS inverter:

A. Switching power dissipation occurs in the transistors when the input is fixed to the power rail or the ground rail.  
B. Short-circuit power dissipation occurs when a low resistance path exists between the power and ground rails for a short duration during transition.  
C. Static power dissipation occurs in the output load capacitor when the inverter is toggling at a fixed frequency.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All A, B, C

**Answer:** Only B

**Explanation:**  
Short-circuit power arises during transitions when both PMOS and NMOS conduct simultaneously, briefly connecting VDD to GND. Switching and static power occur under different conditions.

**Question 2:**

Assume that an inverter drives a load capacitance of 0.1 fF and the supply voltage is 1 V. Further assume that the inverter makes 1×10⁹ ZERO-to-ONE and 1×10⁹ ONE-to-ZERO transitions in one second.  
What is the power dissipated when charging and discharging the load capacitance?

* 1×10⁻⁷ W
* 2×10⁻⁷ W
* 5×10⁻⁸ W
* 1×10⁻⁸ W

**Answer:** 1×10⁻⁷ W

**Explanation:**  
Dynamic power = α·C·V²·f  
= (1) × (0.1×10⁻¹⁵) × (1)² × (2×10⁹) = 1×10⁻⁷ W.

**Question 3:**

Consider the following statements:

A. Clock Gating Technique stops unnecessary clock propagation to save power.  
B. Power Gating Technique reroutes data signals to save power.  
C. Dynamic Voltage Frequency Scaling reduces clock frequency, when possible, to reduce dynamic power dissipation.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All A, B, C

**Answer:** Only A, C

**Explanation:**  
Clock gating disables clocking of inactive modules, reducing switching power. DVFS reduces voltage/frequency dynamically. Power gating disconnects supply rails, not data rerouting.

**Question 4:**

Consider the following statements:

A. Short-circuit power dissipation in a CMOS inverter increases when the slew of the input signal increases.  
B. Static power dissipation in a CMOS inverter increases when the supply voltage increases.  
C. Switching power dissipation in a CMOS inverter increases when the time period of input switching increases.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All A, B, C

**Answer:** Only A, B

**Explanation:**  
A slower input transition causes PMOS and NMOS to conduct longer together (more short-circuit power). Higher supply voltage increases leakage, raising static power.

**Question 5:**

Consider a six-input NOR gate. How many single stuck-at faults are possible for this NOR gate?

* 5
* 6
* 7
* 12
* 14
* 16

**Answer:** 14

**Explanation:**  
Each of the 6 inputs and the single output can have two stuck-at faults (0 and 1): (6+1)×2 = **14 faults**.

**Question 6:**

What is the minimum number of test vectors for the five-input NAND gate that can detect **all** possible single stuck-at faults?

* 5
* 6
* 7
* 8

**Answer:** 6

**Explanation:**  
To test all single stuck-at faults in a 5-input NAND, at least **6 input patterns** are required to activate and propagate all faults to the output.

**Question 7:**

In a scan design, the scan cells get organized as a shift register in which of the following modes of operation?

* Functional mode
* Shift mode
* Power-down mode
* Shut-down mode

**Answer:** Shift mode

**Explanation:**  
During **shift mode**, the scan enable signal activates serial shifting through scan cells, forming a shift register to load and observe test data.

**Question 8:**

Consider the following statements about the scan design methodology:

A. Replacing a D flip-flop with a scan cell can increase the overall area because a scan cell is larger than a D flip-flop.  
B. Replacing a D flip-flop with a scan cell can increase the setup slack of some timing paths because scan cells have an extra delay of the multiplexer on the D-pin path.  
C. The purpose of replacing a flip-flop with a scan cell is to improve the controllability and observability of the circuit.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All A, B, C

**Answer:** Only A, C

**Explanation:**  
Scan cells add a multiplexer at the D-input, increasing area and sometimes delay, but primarily improve **controllability** and **observability** for testing.

**Question 9:**

The non-linear power model of the technology library (in Liberty format) typically models the internal power for an inverter as a function of which of the following quantities:

A. Slew at the input  
B. Capacitive load at the output  
C. Clock frequency

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All A, B, C

**Answer:** Only A, B

**Explanation:**  
Internal power is modeled as a function of **input transition slew** and **output load**, which affect internal short-circuit and charging losses. Frequency affects switching count, not intrinsic power per event.

**Question 10:**

Assume that you are running the tool **OpenSTA** to compute power dissipation. Consider the following statements:

A. It obtains the information on the clock frequency from the SDC file.  
B. It obtains the information on leakage power for each cell from the library file.  
C. It computes the power dissipated within a cell using the power model in the library file.

* Only A
* Only B
* Only C
* Only A, B
* Only B, C
* Only A, C
* All A, B, C

**Answer:** All A, B, C

**Explanation:**  
OpenSTA reads the **clock frequency** from SDC constraints, retrieves **leakage and internal power data** from Liberty files, and uses these models to estimate total power.

**Week 10: Assignment 10**

**Question 1:**

Consider the following statements:  
A. The controlling value of a two-input NAND gate is 0  
B. The controlling value of a two-input OR gate is 1  
C. The controlling value of a two-input XOR gate is 0

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

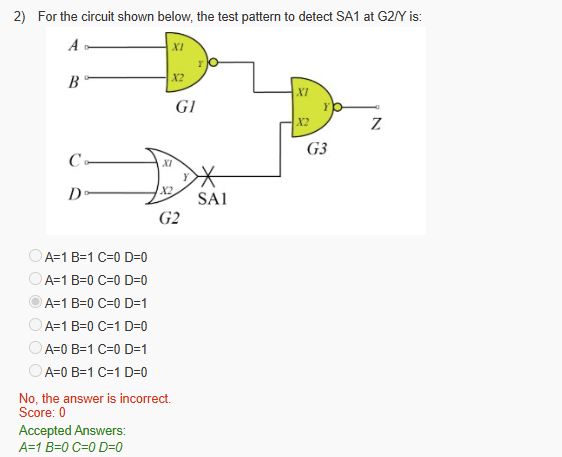
**Answer:** Only A, B

**Explanation:**  
For NAND, any input at **0** forces output **1** after inversion (control value 0). For OR, any input **1** forces output **1** (control value 1). XOR has no single controlling value.

**Question 2:**

For the circuit shown, the test pattern to detect **SA1 at G2/Y** is:

* A=1 B=1 C=0 D=0
* A=1 B=0 C=0 D=0
* A=1 B=0 C=0 D=1
* A=1 B=0 C=1 D=0
* A=0 B=1 C=0 D=1
* A=0 B=1 C=1 D=0

**Answer:** A=1 B=0 C=0 D=0

**Explanation:**  
To detect **Y stuck-at-1**, first **activate** the fault by making the good value at **Y = 0** → set **C=0, D=0** (OR gate output 0). Then **propagate** to output by sensitizing G3 so the input from G2 controls Z: make G1 output **1** (A=1, B=0 for the AND). The good circuit produces Z=0; with SA1 at Y the output flips—fault observed.

**Question 3:**

Consider the following about a **redundant fault** in a combinational circuit:  
A. Its impact cannot be observed at primary outputs.  
B. It cannot be detected by any test vector.  
C. A good ATPG tool should find a non-empty test set for a redundant fault.

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only A, B

**Explanation:**  
By definition, redundant faults **do not affect** the primary outputs and hence are **undetectable**; ATPG will declare them untestable rather than generate vectors.

**Question 4:**

In BIST-based testing, the compact circuit typically used to generate random test patterns is:

* LFSR
* SPEF
* SPICE
* ADDER
* MULTIPLIER

**Answer:** LFSR

**Explanation:**  
A **Linear Feedback Shift Register** generates pseudo-random sequences used as on-chip test stimuli.

**Question 5:**

In BIST-based testing, the signature of a “good” circuit matching the signature of a “faulty” circuit is known as:

* Reducing
* Retiming
* Remapping
* Backtracking
* Aliasing

**Answer:** Aliasing

**Explanation:**  
**Aliasing** occurs when the response compactor (e.g., MISR) maps different output sequences to the same signature.

**Question 6:**

A metal interconnect has sheet resistance **0.08 Ω/□**, length **1 μm**, width **0.01 μm**. What is the resistance?

* 0.8 Ω
* 8 Ω
* 80 Ω
* 800 Ω

**Answer:** 8 Ω

**Explanation:**  
Number of squares = **L/W = 1 / 0.01 = 100**.  
R = **Rs × squares = 0.08 × 100 = 8 Ω**.

**Question 7:**

Consider the following statements:  
A. Dual Damascene is used to fabricate **copper interconnect** layers.  
B. Ion Implantation dopes the semiconductor with required dopants.  
C. Shallow Trench Isolation (STI) deposits a uniform **interconnect** metal layer.

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only A, B

**Explanation:**  
STI provides **isolation trenches** in silicon, not metal deposition.

**Question 8:**

Signal integrity—glitches due to two interconnects on parallel tracks:  
A. Issue becomes more prominent when lines are **closer**  
B. Issue becomes more prominent when **aggressor slew increases**  
C. Issue becomes more prominent when **victim driver strength increases**

* Only A
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only A

**Explanation:**  
Per the given key, only **spacing** is taken as the definitive factor that always worsens capacitive coupling. Effects of aggressor slew and victim strength can be design-dependent and are not universally monotonic.

**Question 9:**

Which file is typically given as an input to a **physical design implementation** tool?

* PDF file
* VCD file
* CPP file
* LEF file

**Answer:** LEF file

**Explanation:**  
**LEF** (Library Exchange Format) describes cell abstracts (size, pins, blockages) for placement and routing.

**Question 10:**

Which tool can perform physical design tasks from **floorplanning to detailed routing**?

* OpenROAD
* VCS
* Icarus
* Yosys

**Answer:** OpenROAD

**Explanation:**  
**OpenROAD** is an open-source PD flow covering floorplanning, placement, CTS, routing, and signoff steps.

**Week 11: Assignment 11**

**Question 1:**

Consider the following statements about chip planning for a hierarchical design:

A. Partitioning divides a given top-level design into smaller parts called blocks.  
B. Budgeting allocates some fraction of a clock cycle to different blocks and the top-level design for signals crossing block boundaries.  
C. Top-level assembly integrates all the blocks at the top level.

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** All A, B, C

**Explanation:**  
Hierarchical chip planning involves **partitioning** to divide the design, **budgeting** to allocate timing for inter-block communication, and **top-level assembly** to integrate the blocks.

**Question 2:**

Consider the following functions:

A. Perform multiplication of binary numbers.  
B. Increase the drive capability of the outgoing signal.  
C. Protect core cells from Electrostatic Discharge (ESD).

Which of the above functions is typically provided by an **IO cell** in an integrated circuit?

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only B, C

**Explanation:**  
IO cells act as **buffers** (drive strength increase) and provide **ESD protection** for internal circuitry.

**Question 3:**

The special circuit element that supplies power to a chip is known as:

* Power Pads
* Filler Cell
* Integrated Clock Gater (ICG)
* Spare Cell

**Answer:** Power Pads

**Explanation:**  
**Power pads** are used to provide the VDD and GND connections from the external package to the internal power distribution network.

**Question 4:**

Consider the following statements about chip planning:

A. Placement of macros (large objects) is typically undertaken during **floorplanning**.  
B. Placement of numerous standard cells is typically undertaken **before** floorplanning.  
C. Fly lines can be used to guide macro placement during floorplanning.

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only A, C

**Explanation:**  
Macro placement is part of **floorplanning**, and **fly lines** (logical connections) help determine macro proximity. Standard cell placement occurs **after** floorplanning.

**Question 5:**

Consider the following statements about **mesh grid topology** in power planning:

A. Rails and straps are organized as a uniformly spaced array.  
B. The resistance from the power pad to any point is typically low due to multiple parallel paths.  
C. The robustness of the mesh grid is low because a wire break causes network failure.

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only A, B

**Explanation:**  
The **mesh grid** offers high robustness and low resistance due to multiple current paths. A wire break has negligible impact.

**Question 6:**

Why do we need to carry out **legalization** after global placement during standard cell placement?

* To reduce leakage in standard cells
* To avoid legal proceedings against the designer
* To remove overlap among standard cells
* To make incremental functional changes

**Answer:** To remove overlap among standard cells

**Explanation:**  
Global placement may cause cell overlaps. **Legalization** adjusts positions to remove overlaps while preserving placement quality.

**Question 7:**

Consider the following statements about a **Decap Cell**:

A. A Decap Cell is a capacitor inserted between power and ground lines.  
B. A Decap Cell helps mitigate IR drop hotspots.  
C. A designer should maximize Decap Cells since they have no area.

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only A, B

**Explanation:**  
Decap cells act as **local charge reservoirs** to stabilize voltage and mitigate **IR drop**, but they **consume area and leakage power**, so excessive use is avoided.

**Question 8:**

The extra cells placed in an IC for potential later use are known as:

* Filler Cell
* Power Pad
* Isolation Cell
* Level Shifter
* Integrated Clock Gater (ICG)
* Pad Cell
* IO Cell
* Spare Cell

**Answer:** Spare Cell

**Explanation:**  
**Spare cells** are unused logic cells placed to simplify **ECO (Engineering Change Order)** modifications post-fabrication.

**Question 9:**

Consider the following statements regarding physical design using **OpenROAD**:

A. read\_sdc can be used to read the constraints file.  
B. place\_pins can be used to synthesize a design.  
C. write\_def can be used to read the DEF file.

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only A

**Explanation:**  
read\_sdc imports design constraints. place\_pins defines IO pin locations, and write\_def **writes**, not reads, DEF files.

**Question 10:**

What is the purpose of the command report\_tns in **OpenROAD**?

* It reports the Total Negative Slack
* It reports the Total Negative Skew
* It reports the Total Netlist Speed
* It reports the Total Netlist Slew
* It reports the Total Netlist Skew
* It reports the Total Netlist Slack

**Answer:** It reports the Total Negative Slack

**Explanation:**  
**TNS (Total Negative Slack)** quantifies the total amount of timing violation in a design, summing all negative slacks across timing paths.

**Week 12: Assignment 12**

**Question 1:**

The clock signal arrives at sink **S1** at 45 ps and at **S2** at 20 ps. What is the clock skew between these two sinks?

* 5 ps
* 10 ps
* 15 ps
* 20 ps
* 25 ps
* 30 ps

**Answer:** 25 ps

**Explanation:**  
Clock skew = |arrival time at S1 – arrival time at S2| = |45 – 20| = **25 ps**.

**Question 2:**

Consider the following statements about **Clock Tree Synthesis (CTS):**

A. Its primary objective is to minimize clock skew.  
B. Its primary objective is to fix functional problems in the design.  
C. It can sometimes introduce useful skews to improve system performance or fix timing violations.

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only A, C

**Explanation:**  
CTS aims to **minimize skew** and **balance clock delay** across sinks. Sometimes, **useful skew** is intentionally introduced to meet timing closure.

**Question 3:**

Consider the following clock architectures:

A. B-tree  
B. H-tree  
C. X-tree

Which of the above are valid **symmetric clock tree architectures**?

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only B, C

**Explanation:**  
**H-tree** and **X-tree** topologies ensure **equal path lengths** and **balanced delay** to achieve low skew. **B-tree** is asymmetric.

**Question 4:**

Consider the following statements about the **mesh architecture** in the global clock distribution network:

A. It exhibits very small clock skews.  
B. It is impacted by process-induced variations more than the tree architecture.  
C. It dissipates zero short-circuit power even with input clock skew at mesh drivers.

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only A

**Explanation:**  
Clock meshes are **highly robust** and exhibit **minimal skew** due to multiple redundant paths. However, they consume more power and are not immune to variations.

**Question 5:**

The planning phase of **Detailed Routing** that produces a high-level routing of nets is called:

* Route Making
* Congestion Routing
* ECO
* Signoff
* Global Routing
* Tape out

**Answer:** Global Routing

**Explanation:**  
**Global routing** provides an approximate routing plan dividing chip area into tiles, ensuring congestion-free detailed routing later.

**Question 6:**

For an edge **e** in a grid graph, usage = 11 and capacity = 10. What is the **congestion** for edge **e**?

* 0
* 0.91
* 1
* 1.1
* 10
* 11

**Answer:** 1.1

**Explanation:**  
Congestion = **usage / capacity = 11 / 10 = 1.1**, indicating overflow and potential routing congestion.

**Question 7:**

Which of the following tasks is carried out primarily to improve the **manufacturability** of a circuit?

* Adding dummy metal fills
* Adding clock-gating cells
* Adding IO cells
* Adding power pads
* Adding scan cells

**Answer:** Adding dummy metal fills

**Explanation:**  
**Dummy metal fills** ensure **planarity during CMP (Chemical Mechanical Polishing)** and improve yield and manufacturability.

**Question 8:**

Consider the following statements about **parasitic extraction:**

A. It determines parasitic resistance, capacitance, and inductance from the layout.  
B. Capacitance extraction partitions layout into smaller windows matched with pre-characterized patterns.  
C. We typically use field solvers directly for all nets since they are extremely fast.

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** Only A, B

**Explanation:**  
Parasitic extraction models **R, C, L** based on layout geometry. Pattern matching accelerates capacitance extraction, while full field solvers are too slow for large circuits.

**Question 9:**

For IC design, what do we mean by **tape out**?

* Measuring the size of an IC using tape
* Taking a backup of the netlist on magnetic tape
* Adding paper tape to the IC design
* Handing over the final IC layout to the foundry for fabrication

**Answer:** Handing over the final IC layout to the foundry for fabrication

**Explanation:**  
**Tape-out** is the final step in the design flow — delivering the **verified layout** (GDSII/OASIS format) to the foundry for chip fabrication.

**Question 10:**

Consider the following statements regarding **physical design using OpenROAD:**

A. clock\_tree\_synthesis performs clock tree synthesis.  
B. write\_spef dumps parasitic information.  
C. check\_antennas detects plasma-induced gate damage violations.

* Only A
* Only B
* Only C
* Only A, B
* Only A, C
* Only B, C
* All A, B, C

**Answer:** All A, B, C

**Explanation:**  
All three commands are key stages in OpenROAD’s PD flow — **CTS**, **SPEF generation**, and **antenna rule checking** before signoff.

✅ **End of Assignment**