## Congratulations! You passed!

Grade received 95% Latest Submission Grade 95% To pass 80% or higher

Go to next item

**1.** In VHDL the following assignment statements operate in parallel: 1 / 1 point

```
Y <= A + B;

Z <= Y - C;
```

- ○ From assignment to assignment within a process
- ◉ From process to process within an architecture
- ○ From port to port within an entity

**Correct**
Yes, each process operates in parallel and independently.

**2.** In the following VHDL architecture code: 1 / 1 point

**2.** In the following VHDL architecture code: 1 / 1 point

```
begin
 with SEL select
 z <= A when "00",
 B when "01",
 C when "10",
 D when others;
end
```

The **when others** covers and catches the following conditions:

- [x] D when "11",

  Correct
  Yes, 11 is not in the compare list.

- [ ] D when "00",
- [x] D when "UU",

  Correct
  Yes, others includes all values 1 0 X U Z.

- [x] D when "X0",

  Correct
  Yes, others includes all values 1 0 X U Z.

3. Choose the VHDL architecture signal assignment for a selected 2:1 MUX:

1 / 1 point

- (●) begin

  with SEL select

  Z_out <= A_in when '0',

  B_in when others;

  end sel_arch;

- ( ) begin mux_proc: process (A_in, B_in, SEL, Z_out)

  begin

  if SEL = '0' then Z_out <= A_in;

  else Z_out <= B_in;

  end if;

  end process mux_proc;

- ( ) begin

**4.** In the code below, which statement is correct regarding the concatenation assignment? **1 / 1 point**

```
entity bus_build is port (
A: in std_logic_vector(3 downto 0);
Z: out std_logic_vector(7 downto 0) );
end entity bus_build;
architecture bus_arch of bus_build is
begin bus_proc: process (A) begin
Z <= "000" & A & '1';     -- This is the Bus Concatenation
end process bus_proc;
end architecture bus_arch;
```

- ○ we cannot combine signals with different width
- ○ Adding '1' is illegal in this assignment
- ◉ Z has to be exactly 8 bits otherwise there will be an error.
- ○ Z can be wider than the result of concatenation operation in terms of bit-width.

**Correct**
Correct, the bus width shall match the result of concatenation operation which is 8-bits here.

**5.** An unintended Latch is generated from the following VHDL code, because … ? **1 / 1 point**

```
begin latch_proc: process(clock, data)
begin
if ( rising_edge(clock) ) then
q <= data;
end if;
end process latch_proc;
```

- [ ] clock is included in the sensitivity list
- [x] data is included in the sensitivity list

**Correct**
Correct, this will give unintended results.

- [ ] clock is a synchronous process
- [x] there is no else statement for data

**Correct**
Yes, else for data needs to be defined also.

Correct
Yes, else for data needs to be defined also.

6. Which of the following statements are correct about state machines? (**Mark all that apply**) **1 / 1 point**

- [ ] Gray encoding technique consumes more logic cells than other encoding methods
- [x] One-Hot encoding results in the greatest number of logic cells among other encoding methods.

Correct
This is correct.

- [ ] In Johnson encoding there will be more than one bit transitioning from one state to the immediate next one.
- [x] Binary encoding is the most efficient way to implement state machines in terms of number of logic cells.

Correct
Correct.

7. In the following VHDL snippet, select all correct answers: **0.5 / 1 point**

**7.** In the following VHDL snippet, select all correct answers: **0.5 / 1 point**

```
Constant T = 10: time:= 20ns;   -- clock period
process begin
clock <= '0';
wait for T/2;
clock <='1';
wait for T/2;
end process;
z_out <= A_bus and B_bus after 2ns;
```

- [ ] 2ns is synthesized into the design
- [ ] Delays can only be used in simulation.
- [x] A clock signal with period of 20 ns will be synthesized in the FPGA fabric.

**This should not be selected**
Incorrect. Clock is an input signal to the unit under test (UUT).

- [x] The value of T cannot be changed inside the process block

**Correct**
Correct, because it's a constant and defined once at the top.

8. The following VHDL statements infer a flip-flop: **(Mark all that apply)** 1 / 1 point

- [ ] if (clk = '1') then q <= d;
- [x] if (rising_edge (clk) ) then q <= d;

  Correct
  Yes.

- [x] if ( clk'event and clk='1') then q<=d;

  Correct
  Yes.

- [ ] q <= d when (sel = '1') else '0';

9. choose which VHDL assignments create a flip-flop: **(Mark all that apply)** 1 / 1 point

if (rising_edge(clk) ) then

X <= A and B;

**9.** choose which VHDL assignments create a flip-flop: (**Mark all that apply**)

**1 / 1 point**

```
if (rising_edge(clk) ) then
X <= A and B;
Y:= C nand D;
Z <= E or F;
end if;
```

- [x] X

**Correct**
Yes.

- [ ] Y
- [x] Z

**Correct**
Yes.

**10.** The following VHDL code generates a divided half clock:

1 / 1 point

- process (clk) begin

  if ( rising_edge(clk) ) then q <= div2 clk;

  end if;

  end process;

- (●) process (clk) begin

  if ( rising_edge(clk) ) then q <= not d;

  end if;

  end process;

**Correct**
Yes, the not inversion creates a divide by 2.