دانشکده مهندسی کامپیوتر

طراحی کامپیوتری سیستم‌های دیجیتال

استاد

دکتر بیت الهی

دانشجویان

زمستان ۱۴۰۳

Q1)

Part A:

Describe the differences between scalar, composite, and physical data types in VHDL. Provide one real-world application where each type is useful.

Part B:

Given the following VHDL code snippet, identify and correct the errors:

```
library IEEE;
use IEEE.std_logic_1164.all;
use IEEE.std_logic_unsigned.all;

entity DataTest is
    port(
        input_A: in integer;
        input_B: in std_logic_vector(7 downto 0);
        output_C: out integer
    );
end DataTest;

architecture Behavioral of DataTest is
begin
    process(input_A, input_B)
    begin
        output_C <= input_A + input_B; -- Error?
    end process;
end Behavioral;
```

Explain why the error occurs and propose a type conversion approach to fix it.

Q2)

Design a 4-bit Arithmetic Logic Unit (ALU) supporting the following operations based on a 2-bit selector input:

Implementation Requirements:

1. Use a behavioral modeling approach for arithmetic operations.
2. Use a concurrent modeling approach for the NAND operation.

Selector	Operation	Formula
00	Increment	Y = A + 1
01	Subtract	Y = A - B
10	NAND	Y = A NAND B
11	Multiplication	Y = A × B (only LSB 4-bit result)

3. Introduce inertial delay for the addition and subtraction operations.
4. Demonstrate usage of constants and generics in the implementation.
5. Provide a testbench covering all four operations.

Q3)

Design a Data Format Converter that accepts two inputs:

1. An integer input (A) within the range 0 to 1023.
2. A std_logic_vector input (B) of 10 bits.

The converter should process and output multiple representations of both inputs using user-defined records. The output must include two records:

- Record X (for integer input A) containing:
    1. Binary representation (`std_logic_vector` of 10 bits)
    2. Hexadecimal representation (`std_logic_vector` of 4 bits)
    3. Unsigned integer representation (0-1023)
- Record Y (for `std_logic_vector` input B) containing:
    1. Decimal representation (`integer`)
    2. Gray Code representation (`std_logic_vector` of 10 bits)
    3. BCD (Binary Coded Decimal) representation (`std_logic_vector(11 downto 0)`)

The conversion should be implemented using conditional statements (`when-else`) and subprograms (functions or procedures). The output records must be updated synchronously with a clock signal.

Q4)

A) Briefly explain the difference between inertial delay and transport delay in VHDL.

B) Give a short code snippet showing how each is used (for example, assigning a delayed signal value with inertial vs. transport).

C) Then discuss a scenario or example circuit where inertial delay might be more realistic, and another scenario where transport delay more accurately represents the hardware behavior.

Q5)

In Lecture 3, you saw that generics allow design parameters (like width or propagation delay) to be passed into an entity from the outside. Suppose you need to implement a simple 4-bit AND gate whose propagation delay is adjustable via a generic parameter.

A) Entity:

Write a VHDL entity declaration named and4_generic with:

• A generic integer parameter called DELAY (default = 10 ns).

• Two 4-bit input ports (A and B) of type STD_LOGIC_VECTOR(3 DOWNTO 0).

• One 4-bit output port (Y) of the same type.

B) Architecture:

Inside the architecture, use a concurrent signal assignment to implement the bitwise AND. Apply the generic delay so that the output Y changes only after DELAY nanoseconds whenever any input bit changes.

C) Discussion:

Why is a generic more flexible than a constant when you want to reuse the same component multiple times but with different parameter values?

Q6)

You need a 2-to-4 decoder with an active-high enable (EN) input. The decoder has two input bits (A1, A0) and four output bits (Y3, Y2, Y1, Y0). When EN = '1', exactly one output should be '1' based on the input A1A0, while the other outputs are '0'. If EN = '0', then all outputs should be '0'.

A) First Approach:

Write the architecture using conditional signal assignments (when ... else).

B) Second Approach:

Rewrite the same decoder's architecture using the selected signal assignment (with ... select ... when).

C) Discussion:

Compare how each approach handles the decoding logic in a "parallel" fashion.

Q7)

You have two 8-bit input signals, A and B, both of type STD_LOGIC_VECTOR(7 DOWNTO 0). You want to add them together (arithmetic addition) and store the sum in another 8-bit output signal, SUM, also of type STD_LOGIC_VECTOR(7 DOWNTO 0).

1. Why can't you directly write SUM <= A + B; in standard VHDL without extra steps or packages?

2. How can you perform this addition correctly using data conversion (either via SIGNED/UNSIGNED or conversion functions like conv_integer)? Provide a short code snippet that:

• Declares the necessary libraries.

- Converts A and B to a type that supports arithmetic.
- Performs the addition.
- Assigns the result back to SUM (still a STD_LOGIC_VECTOR).