**NOR FAKHIRA BT YUSOF**

**U2000780**

WIC2003: Tutorial 03

1. Given the following architecture body, write the associated entity declaration:

architecture csa of and\_or is

signal s1, s2 : std\_logic ;

begin

s1 <= a and b;

s2 <= a and c;

f <= s1 or s2;

end csa;

entity and\_or is

port(

a : IN STD\_LOGIC;

b : IN STD\_LOGIC;

c : IN STD\_LOGIC;

f : OUT STD\_LOGIC

);

end and\_or;

1. **What is the difference in operation of a port of mode inout and one of mode buffer?**

BUFFER: Data flows out of the entity, but the entity can read the signal (allowing for internal feedback). However, the signal cannot be driven from outside the entity, so it cannot be used for data input.

INOUT: Data can flow both in and out of the entity, and the signal can be driven from outside the entity.

1. **What information about a design entity does an architecture body provide?**

The architecture body provides an "internal" view; it describes the behaviour or the structure of the component. The connections between components are specified within component instantiation statements.

1. **Into what two parts is an architecture body divided? What is the purpose of each part?**

The architecture part is divided into two sections: declaration and instantiation part.

Declaration part is between the keywords architecture and begin. Here, all the signals, components, constants, variables, … must be declared.

The instantiation part starts after the keyword begin. Four instances of the full adder are constructed and using the port map statements they are connected together.

1. **What are the three different pure coding styles for an architecture body and how is each distinguished?**

Dataflow, Behavioral, Structural

A dataflow architecture uses only concurrent signal assignment statements, a behavioral architecture uses only process statements while a structural architecture uses only component instantiation statements.

1. A two-to-one multiplexer has two data inputs, a and b, and one select control input, s. The multiplexer has a single output c. If s = 0, c is equal to a. If s = 1, c is equal to b.
   1. Write an entity declaration for the multiplexer named mux. Use only the data type std\_logic.

entity mux is

port (a, b, s : in std\_logic;

c : out std\_logic);

end mux;

* 1. Write an architecture for mux that uses Boolean equations in a dataflow style architecture.

ARCHITECTURE dataflow OF mux IS

BEGIN

c <= a WHEN s = '0' ELSE b ;

END dataflow ;

* 1. What other statements must be included in the design file containing the multiplexer’s entity and architecture in order to compile the design entity? Why must these statements be included?

LIBRARY ieee ;

USE ieee.std\_logic\_1164.all ;

These statements must be included because they are known as a library to story all the entities, architectures and packages. If we omit this step, our toolset will be unable to link the design files together.

1. Given the following truth table:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **a** | **b** | **c** | **x** | **y** |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 1 | 0 |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 1 |
| 1 | 1 | 0 | 0 | 1 |
| 1 | 1 | 1 | 1 | 0 |

* 1. Write a canonical sum-of-products Boolean equation for each output.

x = a’b’c + a’bc’ + ab’c’ + abc

y = a’b’c’ + ab’c + abc’

x <= (not a and not b and c) or (not a and b and not c) or (a and not b and not c)

or (a and b and c)

y <= (not a and not b and not c) or (a and not b and c) or (a and b and not c)

* 1. Write a complete VHDL design description of a design entity that accomplishes the function defined by the truth table. Use a simple dataflow architecture where the signal assignment statements are Boolean equations.

library ieee;

use ieee.stg\_logic\_1164.all;

entity q7design is

port( a, b, c: in std\_logic;

x, y : out std\_logic);

end q7design;

architecture dataflow of q7design is

begin

x <= ((not a) and (not b) and c) or ((not a) and

b and (not c)) or (a and (not b) and (not

c)) or (a and b and c);

y <= ((not a) and (not b) and (not c)) or (a and

(not b) and c) or (a and b and (not c));

end dataflow;

1. Write a design description that accomplishes the function defined by the following truth table. Use a dataflow style architecture consisting of Boolean equations.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **a** | **b** | **c** | **x** | **y** |
| 0 | 0 | 0 | 0 | 1 |
| 0 | 0 | 1 | 1 | 0 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 1 |

x = a’b’c + ab’c’

y = a’c’ + abc

library ieee;

use ieee.std\_logic\_1164.all;

entity q8design is

port( a, b, c: in std\_logic;

x, y : out std\_logic);

end q8design;

architecture dataflow of q8design is

begin

x <= ((not a) and (not b) and c) or (a and (not b)

and (not c));

y <= ((not a) and (not c)) or (a and b and c);

end dataflow;

1. A 74HC10 IC contains three 3-input NAND gates.

(a) Write an entity declaration for the design entity triple\_3 that is functionally equivalent to a 74HC10.

entity triple\_3 is

port ( a, b, c: in std\_logic;

f1, f2, f3 : out std\_logic);

end triple\_3;

(b) Write an architecture in the dataflow style for the design entity triple\_3. Use only signal assignment statements with Boolean equations.

architecture dataflow of triple\_3 is

begin

f1 <= (a nand b nand c);

f2 <= (a nand b nand c);

f3 <= (a nand b nand c);

end dataflow;