Embedded System Lab: Session Second - VHDL

LAB-4: Combinational Logic Design Using VHDL

Objectives

To enable us to write VHDL code for a Field Programmable Gate Array (FPGA) capable of:

- Implementing combinational circuits
- Implementing test benches to verify the working of combinational circuits

Equipment Required

Hardware:

- Spartan-3E or Spartan-3AN FPGA starter kit
- Power cable and Data cable

Software:

- Xilinx ISE (Integrated Synthesis Environment) Design Suite
- iMPACT configuration tool

Background

VHDL stands for Very High Speed Integrated Circuit (VHSIC) Hardware Descriptive Language. It is one of the programming languages which is used to model the digital circuits by using different style of modelling such as dataflow, behavioral and structural. It is an event driven language, it means whenever an event occurs on signals in VHDL, it triggers the execution of a statement. It allows both concurrent as well as sequential modelling. It is case-insensitive and is a strongly typed language, that is, it does not support implicit conversion between data types. It supports code reusability and code sharing via packages and user defined libraries. In VHDL, an entity is used to describe a hardware module. An entity can be described using,

- Entity declaration
- Architecture
- Configuration
- Package Declaration
- Package Body

Lab Exercises:

1. Write VHDL code to implement the logic circuit shown in below figure, which has 4 inputs (x1, x2, x3 and x4) and one output (f).
   - Provide the following architectural styles:
     a) Dataflow Style
     b) Behavioral Style
     c) Structural Style
   - Write a VHDL test bench to verify the operation of the logic circuit.
   - Provide a simulation waveform depicting all possible input cases.

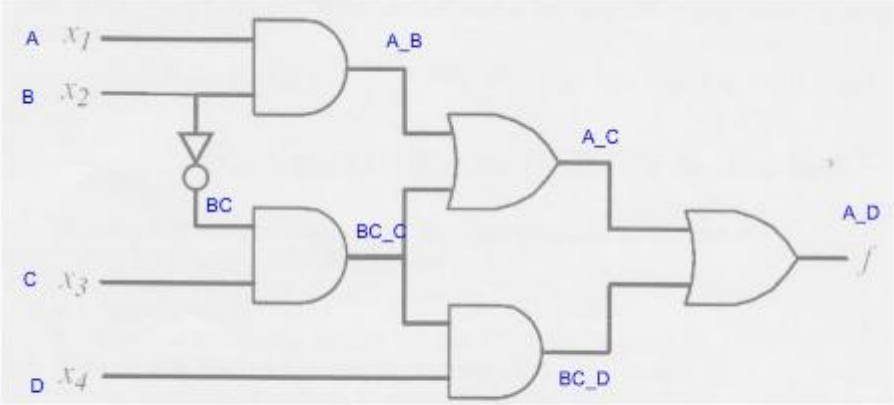


Figure 1: Qno.a: Combinational Circuit with Four Inputs and One Output

2. Write VHDL code to design a logic circuit that implements the truth table of BCD-to-Gray code converter.
   - Use Karnaugh maps to simplify the output functions.
   - Provide the following architectural styles:
     a) Dataflow Style
     b) Behavioral Style
     c) Structural Style using only NOR gates
   - Write a VHDL test bench to verify the operation of the logic circuit.
   - Provide a simulation waveform depicting all possible input cases.

3. Write VHDL code to implement the logic function (F) with the three input variables x1, x2, and x3. The function (F) is equal to 1 if and only if two variables are equal to 1; otherwise, it is equal to zero.

- Draw a truth table for the function (F), and use Karnaugh maps to simplify
- Provide the following architectural styles:
  a) Dataflow Style
  b) Behavioral Style
  c) Structural Style using only NOR gates
- Write a VHDL test bench to verify the operation of the logic circuit.
- Provide a simulation waveform depicting all possible input cases.

4. Write VHDL code to implement the implicit sum of products (SOP) and product of sums (POS) logic functions.

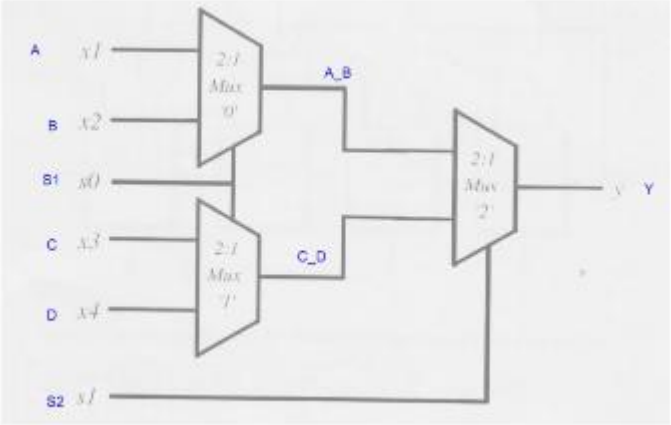
$$f(x_1, x_2\, x_3, x_4) = \sum (m_0, m_1, m_4, m_5, m_8, m_9, m_{14}, m_{15})$$

$$f(x_1, x_2\, x_3, x_4) = \prod (M_0, M_1, M_5, M_8, M_9, M_{13}, M_{15})$$

- Draw a truth table for the function (F), and use Karnaugh maps to simplify
- Provide the following architectural styles:
  a) Dataflow Style
  b) Behavioral Style
  c) Structural Style using only NOR gates
- Write a VHDL test bench to verify the operation of the logic circuit.
- Provide a simulation waveform depicting all possible input cases.

5. Write VHDL code to implement a 2:1 MUX having inputs x1 and x2, select line s and output y.
   - Provide the following architectural implementations:
     a) Using WHEN-ELSE statement
     b) Using IF-THEN-ELSE statement
   - Write a VHDL test bench to verify the operation of the 2:1 MUX.
   - Provide a simulation waveform depicting all possible input cases.

6. Write VHDL code to implement a 4:1 MUX having inputs x1, x2, x3 and x4, select lines s1, s0 and output y using three 2:1 multiplexers as the basic building blocks.



- Use a hierarchical design approach:
  a) Create component definitions in separate (.vhd) files
     Use either Dataflow or Behavioral or Structural design styles
  b) Use Structural design style for the 4:1 MUX architecture:
     i) Make use of 2:1 MUX component declaration
     ii) Make use of 2:1 MUX component instantiation.
- Write a VHDL test bench to verify the operation of the 4:1 MUX.
- Provide a simulation waveform depicting all possible input cases.

7. Write VHDL code to implement a 4-bit adder/subtractor using four 1-bit full adders.
   - Use a Structural architecture style with hierarchical design approach:
     a) Use 1-bit adder as the basic building block
     b) Implement the 4-bit adder/subtractor using four 1-bit full adders.
   - Write a VHDL test bench to verify the operation of the 4-bit adder/subtractor.
   - Provide a simulation waveform depicting all possible input cases.