Use of multiplexer and demultiplexer

As a human, if you were asked to select your favourite food or favourite car, how would you compare your choices and conclude an answer? Of course, the answer for the above varies from person to person depending upon their personal preferences and the result will depend on individual choices. It’s a cardinal aspect of our life but the main picture here is that we humans have a selection process to compare between our options and reach a conclusion. Now, what if we want to do the same thing with electronics? Is there any device/component that paves the way to select the data depending on our choice? Yes, there is a solution to this question. In the digital electronic world, a combinational circuit named “Multiplexer" is a device that is commonly used for the selection process. Let us delve deep into what is a Multiplexer, its specifications, and how can we design one using VHDL on Modelsim. If you are new here, do check out our previous VHDL tutorials using ModelSim. What is a Multiplexer? Just like a normal combinational circuit, the Multiplexer has input and output ports. Multiplexer is a combinational circuit that enables us to select/switch between the available input data and map it to the output port. Also, there is an Enable Port which means ON/OFF. When the Enable Port(E) is “1” à ON, Enable Port is “0” à OFF. This is something similar to an ON/OFF switch in electronic devices. The additional feature of this device is the “selection Ports”. Selection Ports enable us to select/switch between the available input data and map the corresponding selected input data to the output port. So, there are three ports in this device Input Port (i.e. Data Port), Selection Ports, and Output Port. There is some relationship between the number of input and selection ports that a Multiplexer can have. Let's see what is the relationship between these two ports. Relationship between Input and Selection Ports Each Multiplexer can have “n” Selection Ports. Then the number of inputs that the Multiplexer can have is 2n input Ports. There is always only “one” output Port. The general expression for the number of ports that a Multiplexer can have is given below. Enable Port: 1 No. of Selection Ports: n No. of Input Ports(N): 2n No. of Output Ports: 1 In general, the Multiplexer is expressed by “N X 1 Multiplexer” or “N X 1 Mux” in which “N” stands for the number of Input Ports as mentioned above. For example, if No of Selection Ports: n=2, then-No of Input Ports(N): 22=4. So then, it's denoted as “4 X 1 Multiplexer” or “4 X 1 Mux” which means “4 Inputs and 1 Output”. The block diagram and the Truth Table of the Multiplexer with No. of Selection Ports: n=2 is given below. Now have a look at the truth table. When Enable is “0” which means it is in the “OFF” state whatever may be the input and Selection Port values, the output Port (Y) is “X”. “X” means it's undefined/unknown. When a circuit is in an “OFF” state, we can't determine its values, hence it is declared as undefined. When Enable is “1” which means it's in the “ON” state, the circuit operates, selects/switches data, and maps the output port with corresponding inputs depending on the selection port (S0, S1) values as given below. Data(Input) Enable S0 S1 Y(output) D3 D2 D1 D0 0 X X X D3 D2 D1 D0 1 0 0 D3 D3 D2 D1 D0 1 0 1 D2 D3 D2 D1 D0 1 1 0 D1 D3 D2 D1 D0 1 1 1 D0 Now that we know what multiplexer is, we should not limit ourselves just by knowing the circuits. Let's implement the same using VHDL in ModelSim. To get an overview of VHDL Programming Basics refer to ”Getting Started with VLSI and VHDL using ModelSim -A Beginners Guide” How to Design Multiplexer using VHDL? First, I will provide with entire code that is scripted for designing the Multiplexer and disintegrate the code for a better understanding. The VHDL code for Multiplexer is given below. For understanding purpose, I am implementing 4 X 1 Multiplexer VHDL Code of 4 X 1 Multiplexer (a.k.a 4 X 1 Data Selector): Library ieee; use ieee.std_logic_1164.all; entity mux_41 is port(signal data :in std_logic_vector(3 downto 0); signal select_line: in std_logic_vector(1 downto 0); signal enable: in std_logic; signal output:out std_logic); end mux_41; architecture sim of mux_41 is begin output