|  |  |  |  |
| --- | --- | --- | --- |
| **Course Name:** | **Digital Design Laboratory** | **Semester:** | **III** |
| **Date of Performance:** | **\_\_\_ / \_\_\_ / \_\_\_\_\_\_** | **Batch No:** | **B4** |
| **Faculty Name:** |  | **Roll No:** | **16010122221** |
| **Faculty Sign & Date:** |  | **Grade/Marks:** | **\_\_\_/25** |

**Experiment No: 8**

**Title: 1-bit adder on VHDL**

|  |
| --- |
| **Aim and Objective of the Experiment:** |
| To implement 1-bit adder on VHDL |

|  |
| --- |
| **COs to be achieved:** |
| **CO4**: Implement digital networks using VHDL |

|  |
| --- |
| **Tools used:** |
| Quartus, ModelSim |

|  |
| --- |
| **Theory:** |
| A 1-bit adder, a fundamental component of digital circuits, performs binary addition of two 1-bit numbers. It utilizes logic gates to generate the sum and carry-out outputs. A half-adder adds two bits without considering the carry from the previous stage, while a full-adder accounts for the carry input.  Using VHDL, a hardware description language, the 1-bit adder can be designed as a combinational circuit. VHDL facilitates the creation of a structural and behavioral description of the adder. In practice, this simple unit serves as a building block for constructing larger multi-bit adders, enabling arithmetic operations in microprocessors and digital systems. |

|  |
| --- |
| **Implementation Details** |
| 1st Code:  library ieee;  use ieee.std\_logic\_1164.all;  entity HA is  port (a,b:in std\_logic;  s,c: out std\_logic);  end HA;  architecture HA\_arch of HA is  begin  s<=a xor b;  c<=a and b;  end HA\_arch;  2nd code:  library ieee;  use ieee.std\_logic\_1164.all;  entity tb is  end tb;  architecture tb\_arch of tb is  component HA is  port(a,b:in std\_logic;  s,c:out std\_logic);  end component;  signal a,b,s,c:std\_logic;  begin  tbm:HA port map(a,b,s,c);  process  begin  a<='1';  b<='0';  wait for 5ns;  a<='0';  b<='1';  wait for 5ns;  a<='1';  b<='1';  wait for 5ns;  end process;  WAVEFORM :  ddl exp 8 wave form  RTL VIEWER:  image |
| **Post Lab Subjective/Objective type Questions:** |
| 1. How can 1-bit adder be used to implement a 4-bit adder?   ANS1) You can use a 1-bit adder as a building block to implement a 4-bit adder by cascading it four times and managing the carry-out from one stage to the next. Here's how you can do it step by step:  1. \*\*1-Bit Adder\*\*: Start with a 1-bit adder. This is a basic unit that adds two 1-bit numbers and produces a sum and a carry-out.  2. \*\*Cascading\*\*: Connect four of these 1-bit adders in series, one after the other, to create a 4-bit adder. Each stage takes in a pair of bits from the two numbers you want to add and produces a sum bit and a carry-out bit. The carry-out from one stage feeds into the carry-in of the next stage.  Here's a simplified representation, where A and B are the input bits, S represents the sum, and C represents the carry:  ```  Stage 1: A0, B0, S0, C0  Stage 2: A1, B1, S1, C1 (with C1 connected to C0 from the previous stage)  Stage 3: A2, B2, S2, C2 (with C2 connected to C1 from the previous stage)  Stage 4: A3, B3, S3, C3 (with C3 connected to C2 from the previous stage)  ```  3. \*\*Input Management\*\*: Ensure that you feed the corresponding bits of the two 4-bit numbers (A and B) into the respective stages of the 4-bit adder. For example, A0 and B0 go to Stage 1, A1 and B1 go to Stage 2, and so on.  4. \*\*Output Management\*\*: The sum bits (S0, S1, S2, S3) from each stage are the bits of the final 4-bit sum, and the carry-out from the last stage (C3) is the carry-out of the overall 4-bit addition.  5. \*\*Overflow\*\*: Be aware of overflow conditions. If the final carry-out (C3) is 1, it indicates that there's an overflow in the addition.  By connecting the 1-bit adders in this way, you can perform 4-bit binary addition. This approach is scalable, and you can extend it to create larger n-bit adders by adding more 1-bit adders in series.   1. What is VHDL used for?   ANS2) VHDL (VHSIC Hardware Description Language) is a high-level programming language primarily used for the design, simulation, and synthesis of digital electronic circuits and systems. It plays a crucial role in the field of digital design and electronic design automation. Here are some of the key uses of VHDL:  1. \*\*Digital Circuit Design:\*\* VHDL is used for describing the behavior and structure of digital circuits. Designers use VHDL to model digital systems, including logic gates, flip-flops, multiplexers, memory elements, and complex digital devices.  2. \*\*Simulation:\*\* VHDL allows for the simulation of digital designs before they are implemented in hardware. Engineers can use VHDL models to perform various types of simulations, such as functional simulation, timing simulation, and fault simulation. This helps in verifying the correctness and functionality of the design.  3. \*\*Synthesis:\*\* VHDL code can be synthesized into actual hardware implementations. This process involves converting the high-level VHDL description into a netlist, which can then be used for FPGA (Field-Programmable Gate Array) or ASIC (Application-Specific Integrated Circuit) design. Synthesis tools optimize the design for a target hardware platform.  4. \*\*Verification:\*\* VHDL is instrumental in the verification and testing of digital designs. Engineers can write testbenches in VHDL to verify that a design behaves as expected and meets the design specifications. VHDL is often used for testbench development to check the functionality and performance of digital systems.  5. \*\*Hardware Modeling:\*\* VHDL allows the modeling of digital systems at various levels of abstraction. Designers can use VHDL to model at a high level for architectural exploration and then refine the design at lower levels of abstraction as needed.  6. \*\*Rapid Prototyping:\*\* VHDL is used for rapid prototyping of digital systems, allowing designers to quickly implement and test new concepts without the need for physical hardware. This is particularly useful in the early stages of development.  7. \*\*Education and Research:\*\* VHDL is widely used in academic settings to teach digital design concepts and facilitate research in the field of digital design, including the development of new algorithms and techniques.  8. \*\*Reusability:\*\* VHDL promotes the creation of reusable design modules, enabling designers to develop IP (Intellectual Property) cores that can be used in various projects and shared with others.  9. \*\*Documentation:\*\* VHDL serves as a form of documentation for digital designs, making it easier for designers to understand and maintain complex systems.  Overall, VHDL is a versatile language that is crucial for the development and maintenance of digital hardware, enabling efficient design, verification, and implementation of complex electronic systems. |

|  |
| --- |
| **Conclusion:** |
| We implemented 1-bit adder on VHDL |

|  |
| --- |
| **Signature of faculty in-charge with Date:** |