Experiment 4: Combinational Circuit 4

Anubhav Bhatla | 200070008

EE-214, WEL, IIT Bombay

September 21, 2021

## Overview of the experiment:

|  |
| --- |
| The purpose of this experiment was to design an Arithmetic Logic Unit using the given logic, coming up with a design to implement the logic, and then implementing the design using VHDL.  Quartus Prime was used to implement the given specifications and my own designs with VHDL. RTL and Gate Level simulations were then carried out using the created Testbench to check for the correctness of the design and its implementation. Scanchain was also used to test the correctness of the design on Krypton hardware.  The report consists of the following sections:   * **Approach:** An overview of how I approached the experiment and the logic behind my designs. * **Design document and VHDL code:** A brief description of all the design components. * **RTL View:** A screenshot of the RTL View generated using Quartus Prime. * **RTL Simulation:** Screenshots for the RTL and Gate Level Simulations. * **Scanchain:** Testing the correctness of my design using Scanchain. |

## Approach to the experiment:

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| We are required the Arithmetic Logic Unit circuit using the logic given below:     1. CONCAT(AB)   This can be directly implemented using the concatenate operator (**&**).   1. A+B   In case of Structural Modelling, we made a 4-bit Adder using 4 Full Adders in series. Let’s look at the truth table for a Full Adder circuit:   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **A(n)** | **B(n)** | **Cn-1** | **Cn** | **S(n)** | | 0 | 0 | 0 | 0 | 0 | | 0 | 0 | 1 | 0 | 1 | | 0 | 1 | 0 | 0 | 1 | | 0 | 1 | 1 | 1 | 0 | | 1 | 0 | 0 | 0 | 1 | | 1 | 0 | 1 | 1 | 0 | | 1 | 1 | 0 | 1 | 0 | | 1 | 1 | 1 | 1 | 1 |   Using this, we find out that:  S(n) = ((A(n) **XOR** B(n)) **XOR** Cn-1)  Cn = ((A(n) **AND** B(n)) **OR** (Cn-1 **AND** (A(n) **OR** B(n))))   1. A **XOR** B   This can be directly implemented using the **XOR** operator.   1. 2A   This can be implemented using the logic used in part 2. By replacing B with A.  In order to implement the MUX function, we can use **if-else** statements with sel(1) and sel(0) as the boolean logics. |

## Design document and VHDL code if relevant:

|  |
| --- |
| Given below is the code for the architecture of the entity **alu\_beh** which is the 4x3 Multiplier circuit we had to implement in Quartus using our own design: |

## RTL View:

|  |
| --- |
|  |

## DUT Input/Output Format:

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | Input  <s1 s0 a3 a2 a1 a0 b3 b2 b1 b0> | Output  <Y7 Y6 Y5 Y4 Y3 Y2 Y1 Y0> | | 0010011010 | 10011010 | | 0101001111 | 00010011 | | 1000110111 | 00000100 | | 1101001111 | 00001000 | |

## RTL Simulation:

|  |
| --- |
|  |

## Gate-level Simulation:

|  |
| --- |
|  |

## Krypton board:

|  |
| --- |
| Map the logic circuit to the Krypton board and attach the images of the pin assignment and output observed on the board (switches/LEDs). |

## Observations:

|  |
| --- |
| After running Scanchain on my design, I found the following result:    I got “Success” for all the testcases which means that my design is correct. |

## References:

|  |
| --- |
| None |