**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **2/06/2020** | **Name:** | **DHAMINI C L** |
| **Course:** | **Digital Design Using HDL** | **USN:** | **4AL17EC025** |
| **Topic:** | **1] FPGA Basics: Architecture, Applications**  **and Uses**  **2] Verilog HDL Basics by Intel**  **3] Verilog Testbench code to verify the**  **design under test (DUT)** | **Semester & Section:** | **6TH & A** |
| **Github Repository:** | **DHAMINI-CL-Course** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
|  |
| **Report:**  **Architecture**  ** A basic FPGA architecture (Figure 1) consists of thousands of fundamental elements called**  **configurable logic blocks (CLBs) surrounded by a system of programmable interconnects,**  **called a fabric, that routes signals between CLBs. Input/output (I/O) blocks interface between**  **the FPGA and external devices.**  ** Depending on the manufacturer, the CLB may also be referred to as a logic block (LB), a logic**  **element (LE) or a logic cell (LC).**  **The fundamental FPGA architecture.**  ** An individual CLB (Figure 2) is made up of several logic blocks. A lookup table (LUT) is a**  **characteristic feature of an FPGA. An LUT stores a predefined list of logic outputs for any**  **combination of inputs: LUTs with four to six input bits are widely used. Standard logic functions**  **such as multiplexers (mux), full adders (FAs) and flip-flops are also common.**  **A simplified CLB: The four-input LUT is formed from two three-input units.**  ** The number and arrangement of components in the CLB varies by device; the simplified**  **example in Figure 2 contains two three-input LUTs (1), an FA (3) and a D-type flip-flop (5), plus**  **a standard mux (2) and two mux’s, (4) and (6), that are configured during FPGA programming.**  ** This simplified CLB has two modes of operation. In normal mode, the LUTs are combined with**  **Mux 2 to form a four-input LUT; in arithmetic mode, the LUT outputs are fed as inputs to the**  **FA together with a carry input from another CLB. Mux 4 selects between the FA output or the**  **LUT output. Mux 6 determines whether the operation is asynchronous or synchronized to the**  **FPGA clock via the D flip-flop.**  ** Current-generation FPGAs include more complex CLBs capable of multiple operations with a**  **single block; CLBs can combine for more complex operations such as multipliers, registers,**  **counters and even digital signal processing (DSP) functions.**  **Verilog HDL Basics**  ** Verilog is a HARDWARE DESCRIPTION LANGUAGE (HDL). It is a language used for describing a**  **digital system like a network switch or a microprocessor or a memory or a flip−flop. It means,**  **by using HDL we can describe any digital hardware at any level.**  ** Designs, which are described in HDL are independent of technology, very easy for designing**  **and debugging, and are normally more useful than schematics, particularly for large circuits.**  ** Verilog supports a design at many levels of abstraction. The major three are:**  ** Behavioral level**  ** This level describes a system by concurrent algorithms (Behavioral). Every algorithm is**  **sequential, which means it consists of a set of instructions that are executed one by**  **one. Functions, tasks and blocks are the main elements. There is no regard to the**  **structural realization of the design.**  ** Register−Transfer Level**  ** Designs using the Register−Transfer Level specify the characteristics of a circuit using**  **operations and the transfer of data between the registers. Modern definition of an RTL**  **code is "Any code that is synthesizable is called RTL code".**  ** Gate Level**  ** Within the logical level, the characteristics of a system are described by logical links and**  **their timing properties. All signals are discrete signals. They can only have definite**  **logical values (`0', `1', `X', `Z`). The usable operations are predefined logic primitives**  **(basic gates). Gate level modelling may not be a right idea for logic design. Gate level**  **code is generated using tools like synthesis tools and his netlist is used for gate level**  **simulation and for backend.**  ** Some of the operators used in Verilog HDL**  ** Arithmetic Operators - These operators perform arithmetic operations (+, -, /, \*, %).**  ** Relational Operators - These operators compare two operands and return the result in**  **a single bit, 1 or 0 (==, !=, >, <, >=, <=).**  ** Bit-wise Operators - Bit-wise operators which are doing a bit-by-bit comparison**  **between two operands (&, |, ^, ~, ^~).**  ** Logical Operators - Logical operators are bit-wise operators and are used only for**  **single-bit operands. They return a single bit value, 0 or 1 (!, &&, ||).**  ** Reduction Operators- Reduction operators are the unary form of the bitwise operators**  **and operate on all the bits of an operand vector (&, |, ~&, ~|, ^, ~^).**  ** Shift Operators - Shift operators, which are shifting the first operand by the number of**  **bits specified by second operand in the syntax (>>, <<).**  **Task (DAY - 2)**  **Implement a 4:1 MUX and write the test bench code to verify the module**  **Logic circuit for 4:1 MUX**  **Verilog code:**  **module m41 (input a,**  **input b,**  **input c,**  **input d,**  **input s0, s1,**  **output out);**  **assign out = s1 ? (s0 ? d : c) : (s0 ? b : a);**  **endmodule** |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date:2/6/2020** |  | **Name: DHAMINI C L** |  | |
| **Course:PYTHON** |  | **USN:4AL17EC025** |  | |
|  |  |  |  | |
| **Topic: Web-based Financial Graph** |  | **Semester & Section:6TH A SEC** |  | |
| **AFTERNOON SESSION DETAILS** | | | |
| **Image of session** | | | |
| **Report – Report can be typed or hand written for up to two pages.**   1. he basic function of the developed application includes representation of stock market data into graphical format. Creating a graph is a simple approach for giving a brief idea of the current trends in the market. Constructing and analysing Stock Charts provide an easy-to-read graphical representation of a stock's price alteration over a definite interval of time. They sometimes called stock market graphs   and are a component of Technical Analysis and are an essential component in stock trading.  Firstly, web scraping is performed to scrape the required data. We are using real-time stock market data for scraping and then store it into a CSV file format using Python libraries. Data is extracted from the web using Python's Beautiful Soup module. Beautiful Soup is an inbuilt package in Python that is used for parsing HTML and XML documents (including having distorted or abnormal mark- up, i.e. non-closed tags, so named after tag soup). It develops a parse tree for already parsed documents that can further be used to extract data from HTML file or document, which is useful in case of web scraping.  Secondly, the graph is a plot on our designed website from the CSV files containing the scraped data from the web. This task is done by using c3.js and Papa Parse libraries and their respective dependencies. C3 provides an easy way to construct D3-based charts by encapsulating the code that is required to generate the entire chart/graph. C3 library of JavaScript provides a wide range of APIs and Callbacks (Callback is a function in JavaScript that is executed after another program has finished its execution) to access the status of the chart at a particular time. By using this C3 library, we can update the chart/graph even after it is accomplished.  Another library that we have used in graph making is Papa Parse which is the fastest in-browser CSV (or delimited text) parser for JavaScript. Papa Parse is the world's first multi- threaded CSV parser used for the browser. It is reliable and easy to use. Papa-parse is an effective and convenient CSV parser that can handle files having size in gigabytes without crashing. It is capable of manipulating your CSV files in many ways. First off, the input. This component can read your data from anywhere, via a URL, from a raw string or even from your local storage. The output will be an array of rows, where each row is an array of table data, and it will be returned if the header flag is not set. Otherwise, an array of objects will be the product, where each object is a map comprising of the column name and its corresponding value or the row (e.g., {col1: value1, col2: value2}). The recent format is in the vicinity of a JSON file.  The homepage of our dynamically created website is shown below. | | | |