**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **2/06/2020** | **Name:** | **Prajwal Kamagethi Chakravarti P L** |
| **Course:** | **Python** | **USN:** | **4AL17EC073** |
| **Topic:** | * **Interactive Data Visualization with Bokeh** * **Webscraping with Python Beautiful Soup** | **Semester & Section:** | **6 & B** |
| **Github Repository:** | **https://github.com/alvas-education-foundation/Prajwal-Kamagethi.git** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Report – Report can be typed or hand written for up to two pages.**   1. **Interactive Data Visualization with Bokeh**        * **Bokeh is a data visualization library in Python that provides high-performance interactive charts and plots.** * **It is possible to embed bokeh plots in Django and flask apps.** * **Bokeh provides two visualization interfaces to users: bokeh models :** * **A low level interface that provides high flexibility to application developers.**   **2.Webscraping with Python Beautiful Soup**   * **Web scraping is a term used to describe the use of a program or algorithm to extract and process large amounts of data from the web.** * **Whether you are a data scientist, engineer, or anybody who analyzes large amounts of datasets, the ability to scrape data from the web is a useful skill to have.** * **Let's say you find data from the web, and there is no direct way to download it, web scraping using.** * **Python is a skill you can use to extract the data into a useful form that can be imported.** |
| |  |  |  |  | | --- | --- | --- | --- | | **Date:** | **02-06-2020** | **Name:** | **Prajwal Kamagethi Chakravarti P L** | | **Course:** | **DIGITAL DESIGN USING HDL** | **USN:** | **4AL17EC073** | | **Topic:** | * **FPGA Basics: Architecture, Applications and Uses** * **Verilog HDL Basics by Intel** * **Verilog Testbench code to verify the design under test (DUT)** | **Semester & Section:** | **6TH & B** | | **Github Repository:** | **https://github.com/alvas-education-foundation/Prajwal-Kamagethi.git** |  |  | |
| **Report:**      **The**[**field-programmable gate array (FPGA)**](https://www.arrow.com/en/categories/programmable-devices/programmable-logic-devices/fpgas)**is an integrated circuit that consists of internal hardware blocks with user-programmable interconnects to customize operation for a specific application. The interconnects can readily be reprogrammed, allowing an FPGA to accommodate changes to a design or even support a new application during the lifetime of the part.**    **The FPGA has its roots in earlier devices such as programmable read-only memories (PROMs) and programmable logic devices (PLDs). These devices could be programmed either at the factory or in the field, but they used fuse technology (hence, the expression “burning a PROM”) and could not be changed once programmed. In contrast, FPGA stores its configuration information in a re-programmable medium such as static RAM (SRAM) or flash memory. FPGA manufacturers include**[**Intel**](https://www.arrow.com/en/manufacturers/intel)**, Xilinx,**[**Lattice Semiconductor**](https://www.arrow.com/en/manufacturers/lattice-semiconductor)**,**[**Microchip Technology**](https://www.arrow.com/en/manufacturers/microchip-technology)**and**[**Microsemi**](https://www.arrow.com/en/manufacturers/microsemi)**.**  **Implement a 4:1 MUX and write the test bench code to verify the module**  **Verilog design**  **module mux41(**  **input i0,i1,i2,i3,sel0,sel1,**  **output reg y);**    **always @(\*)**  **begin**  **case ({sel0,sel1})**  **2'b00 : y = i0;**  **2'b01 : y = i1;**  **2'b10 : y = i2;**  **2'b11 : y = i3;**  **endcase**  **end**    **endmodule**    **TestBench**  **module tb\_mux41;**  **reg I0,I1,I2,I3,SEL0,SEL1;**  **wire Y;**    **mux41 MUX (.i0(I0),.i1(I1),.i2(I2),.i3(I3),.sel0(SEL0),.sel1(SEL1),.y(Y));**    **initial begin**  **I0 =1'b0;**  **I1= 1'b0;**  **I2 =1'b0;**  **I3 =1'b0;**  **SEL0 =1'b0;**  **SEL1 =1'b0;**  **#45 $finish;**  **end**    **always #2 I0 = ~I0;**  **always #4 I1 =~I1;**  **always #6 I2 =~I1;**  **always #8 I3 =~I1;**  **always #3 SEL0 = ~SEL0;**  **always #3 SEL1 = ~SEL1;**    **always @(Y)**  **$display( "time =%0t INPUT VALUES: \t I0=%b I1 =%b I2 =%b I3 =%b SEL0 =%b SEL1 =%b \t output value Y =%b ",$time,I0,I1,I2,I3,SEL0,SEL1,Y);**    **endmodule**  **OUTPUT**  **time =0 INPUT VALUES: I0=0 I1 =0 I2 =0 I3 =0 SEL0 =0 SEL1 =0 output value Y =0**  **time =2 INPUT VALUES: I0=1 I1 =0 I2 =0 I3 =0 SEL0 =0 SEL1 =0 output value Y =1**  **time =3 INPUT VALUES: I0=1 I1 =0 I2 =0 I3 =0 SEL0 =1 SEL1 =1 output value Y =0**  **time =6 INPUT VALUES: I0=1 I1 =1 I2 =0 I3 =0 SEL0 =0 SEL1 =0 output value Y =1**  **time =8 INPUT VALUES: I0=0 I1 =0 I2 =0 I3 =0 SEL0 =0 SEL1 =0 output value Y =0**  **time =14 INPUT VALUES: I0=1 I1 =1 I2 =1 I3 =0 SEL0 =0 SEL1 =0 output value Y =1**  **time =15 INPUT VALUES: I0=1 I1 =1 I2 =1 I3 =0 SEL0 =1 SEL1 =1 output value Y =0** |