**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **1/06/2020** | **Name:** | **Pv sai suraksha** |
| **Course:** | **Digital Design Using HDL** | **USN:** | **4AL17EC064** |
| **Topic:** | **Industry Applications of FPGA, FPGA Business Fundamentals, FPGA vs ASIC Design Flow, FPGA Basics-A Look Under the Hood.** | **Semester & Section:** | **6th sem**  **B section** |
| **GitHub Repository** | **surakshacourses** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report – Report can be typed or hand written for up to two pages.**  **\*Industry Applications of FPGA, FPGA Business Fundamentals, FPGA vs ASIC Design Flow, FPGA Basics-A Look Under the Hood.**  **Specific application of an FPGA includes digital signal processing, bioinformatics, device controllers, software-defined radio, random logic, ASIC prototyping, medical imaging, computer hardware emulation, integrating multiple SPLDs, voice recognition, cryptography, filtering and communication encoding and many more**  **FPGAs are predominantly programmed using HDLs (hardware description languages) such as Verilog and VHDL. These languages, which date back to the 1980s and have seen few revisions, are very low level in terms of the abstraction offered to the user**  **Summary: Page on 1.an ASIC is a unique type of integrated circuit meant for a specific application while an FPGA is a reprogrammable integrated circuit. Page on 2.an ASIC can no longer be altered once created while an FPGA can. ... FPGA is better than an ASIC when building low volume production circuits.** |

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **1/06/2020** | **Name:** | **Pv sai suraksha** |
| **Course:** | **Python** | **USN:** | **4AL17EC064** |
| **Topic:** | **Collections Module-orderedDict**  **Collections Module-namedtuple**  **Datetime.** | **Semester & Section:** | **6th sem**  **B section** |
| **AFTERNOON SESSION DETAILS** | | | |
| **Image of session**      \* **Collections Module-orderedDict collections Module-namedtuple Datetime**  ***class*collections.ChainMap(*\*maps*)**  **A [ChainMap](https://docs.python.org/3/library/collections.html" \l "collections.ChainMap" \o "collections.ChainMap) groups multiple dicts or other mappings together to create a single, updateable view. If no *maps* are specified, a single empty dictionary is provided so that a new chain always has at least one mapping.**  **The underlying mappings are stored in a list. That list is public and can be accessed or updated using the *maps* attribute. There is no other state.**  **Lookups search the underlying mappings successively until a key is found. In contrast, writes, updates, and deletions only operate on the first mapping.**  **A [ChainMap](https://docs.python.org/3/library/collections.html" \l "collections.ChainMap" \o "collections.ChainMap) incorporates the underlying mappings by reference. So, if one of the underlying mappings gets updated, those changes will be reflected in [ChainMap](https://docs.python.org/3/library/collections.html" \l "collections.ChainMap" \o "collections.ChainMap).**  **All of the usual dictionary methods are supported. In addition, there is a *maps* attribute, a method for creating new subcontexts, and a property for accessing all but the first mapping:** | | | |