**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **29-05-2020** | **Name:** | **Anand kumar k** |
| **Course:** |  | **USN:** | **4al16ec002** |
| **Topic:** | **Logic design** | **Semester & Section:** | **8thsem ‘A’ sec** |
| **Github Repository:** | **Anand-courses** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
|  |
| **Analysis of clocked sequential circuit**  The clocked sequential circuits have flip-flops or gated latches for its memory elements. There is a periodic clock connected to the clock inputs of all the memory elements of the circuit to synchronize all the internal changes of state.  A Sequential logic circuits is a form of the binary circuit; its design employs one or more inputs and one or more outputs, whose states are related to some definite rules that depend on previous states.  There are two types of sequential circuit, synchronous and asynchronous. Synchronous types use pulsed or level inputs and a clock input to drive the circuit (with restrictions on pulse width and circuit propagation). Asynchronous sequential circuits do not use a clock signal as synchronous circuits do.   * Some flip-flops have asynchronous inputs that are used to force the flip-flop to a particular state independently of the clock * The input that sets the flip-flop to 1 is called preset or direct set. The input that clears the flip-flop to 0 is called clear or direct reset. * When power is turned on in a digital system, the state of the flip-flops is unknown. The direct inputs are useful for bringing all flip-flops in the system to a known starting state prior to the clocked operation. * The knowledge of the type of flip-flops and a list of the Boolean expressions of the combinational circuit provide the information needed to draw the logic diagram of the se­quential circuit. The part of the combinational circuit that gene rates external outputs is de­scribed algebraically by a set of Boolean functions called output equations. The part of the circuit that generates the inputs to flip-flops is described algebraically by a set of Boolean func­tions called flip-flop input equations (or excitation equations). * The information available in a state table can be represented graphically in the form of a state diagram. In this type of diagram a state is represented by a circle and the (clock-triggered) transitions between states are indicated by directed lines connecting the circles. * The time sequence of inputs, outputs, and flip-flop states can be enumerated in a state table (transition table). The table has four parts present state, next state, inputs and outputs. * In general a sequential circuit with 'm' flip-flops and 'n' inputs needs 2m+n rows in the state table.   **Digital clock design**  Timer 555: Responsible for generating the clock pulses for the counters, the frequency of the output shoul be 1 hz which means 1 second for each pulse.  Counters: Responsible for generating the time in BCD.  Decoders : Takes the BCD of the counter as input and produces 7 segment output .  7 segments : Displays the time, of course |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date:** | **29-05-2020** | **Name:** | **Anand kumar k** | |
| **Course:** |  | **USN:** | **4al16ec002** | |
| **Topic:** | **python** | **Semester & Section:** | **8thsem ‘A’ sec** | |
| **AFTERNOON SESSION DETAILS** | | | |
| **Image of session** | | | |
| * Functions can have more than one **parameter**:  1. def volume(a, b, c): 2. return a \* b \* c  * Functions can have **default** parameters (e.g. coefficient):  1. def converter(feet, coefficient = 3.2808): 2. meters = feet / coefficient 3. return meters 5. print(converter(10))   Output: 3.0480370641306997  Arguments can be passed as **non-keyword** (positional) arguments (e.g. a) or **keyword** arguments (e.g. b=2 and c=10):   1. def volume(a, b, c): 2. return a \* b \* c 4. print(volume(1, b=2, c=10))  * An **\*args**parameter allows the  function to be called with an arbitrary number of non-keyword arguments:  1. def find\_max(\*args): 2. return max(args) 3. print(find\_max(3, 99, 1001, 2, 8))   Output: 1001   * An **\*\*kwargs** parameter allows the function to be called with an arbitrary number of keyword arguments:  1. def find\_winner(\*\*kwargs): 2. return max(kwargs, key = kwargs.get) 4. print(find\_winner(Andy = 17, Marry = 19, Sim = 45, Kae = 34))   Output: Sim | | | |