**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **13/06/2020** | **Name:** | **Namratha S Hipparagi** |
| **Course:** | **VLSI design** | **USN:** | **4AL16EC040** |
| **Topic:** | **Digital VLSI Design Virtual lab** | **Semester & Section:** | **8 A** |
| **Github Repository:** | **namrathahipparagi\_1** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
|  |
| **Report**  In electronics, a multiplexer, also known as a data selector, is a device that selects between several analog or digital input signals and forwards it to a single output line. A multiplexer of inputs has select lines, which are used to select which input line to send to the output. A multiplexer or mux is a combinational circuits that selects several analog or digital input signals and forwards the selected input into a single output line. A multiplexer of 2n inputs has n selected lines, are used to select which input line to send to the output.    Figure 2 shows how a 4:1 MUX can be constructed out of two 2:1 MUXs.    **LOGIC GATES:**  A logic gate is an idealized or physical electronic device implementing a Boolean function, a logical operation performed on one or more binary inputs that produces a single binary output. Static logic is a design methodology in integrated circuit design where there is at all times some mechanism to drive the output either high or low. For example, in many of the popular logic families, such as TTL and traditional CMOS, there is always a low-impedance path between the output and either the supply voltage or the ground. The function of the PUN is to provide a connection between the output and VDD anytime the output of the logic gate is meant to be 1 (based on the inputs). Similarly, the function of the PDN is to connect the output to VSS when the output of the logic gate is meant to be 0 (based on the inputs). The most widely used logic style is static CMOS. A static CMOS gate is a combination of two networks, called the pull-up network (PUN) and the pull-down network (PDN).  **MOSFET:**  The metal–oxide–semiconductor field-effect transistor, also known as the metal–oxide–silicon transistor, is a type of insulated-gate field-effect transistor that is fabricated by the controlled oxidation of a semiconductor, typically silicon. The channel can be of n-type or p-type, and is accordingly called an nMOSFET or a pMOSFET. The metal–oxide–semiconductor field-effect transistor (MOSFET) is a transistor used for amplifying or switching electronic signals. In MOSFETs, a voltage on the oxide-insulated gate electrode can induce a conducting channel between the two other contacts called source and drain.    Fig: Circuit symbols for nMOS and pMOS respectively |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date:** | **13/6/2020** | **Name:** | **Namratha S Hipparagi** | |
| **Course:** | **Java** | **USN:** | **4al16ec040** | |
| **Topic:** | **The Equals Method……** | **Semester & Section:** | **8 A** | |
| **AFTERNOON SESSION DETAILS** | | | |
| **Report**  **Serialization**:  Serialization is a mechanism of converting the state of an object into a byte stream.  Scan each element in a string, calculate its length and append it with a string and a element separator or deliminator (that deliminator should not be present in the string). To make a Java object serializable we implement the java. io. Serializable interface.  **Deserialized Function:**  Deserialization is the reverse process where the byte stream is used to recreate the actual Java object in memory.  // CPP program to serialize and  // deserialize the array of string  #include<iostream>  using namespace std;  // Function to serialized the array of string  string serialize(string str[], int ln)  {  string temp = "";  for (int i=0; i<ln; i++)  {  int ln = str[i].length();  temp.push\_back('0' + ln);  temp = temp + "~" + str[i];  }  return temp;  }  // Function to deserialize the string  void deserialized(string str, string deserialize[], int ln)  {  int len, pos=0;  string temp = "";  int i = 0;  while(pos>-1)  {  pos = str.find("~", pos+1);  if(pos>0)  {  len = str[pos-1] - 48;  temp.append(str, pos+1, len);  deserialize[i++] = temp;  temp = "";  }  }  }  // Driver function  int main()  {  string str[] = {"geeks", "are", "awesome"};  int ln = sizeof(str)/sizeof(str[0]);  string serializedstr = serialize(str, ln);  cout << "Serialized String : " << serializedstr <<endl;  string deserialize[ln];  deserialized(serializedstr,deserialize,ln);  cout << "Deserialized String : ";  for(int i=0; i<ln; i++)  cout << deserialize[i] << " ";  return 0;  } | | | |