**DAILY ASSESSMENT FORMAT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **09/06/2020** | **Name:** | **Namratha S Hipparagi** |
| **Course:** | **VLSI design** | **USN:** | **4AL16EC040** |
| **Topic:** | **MOSFET - Enhancement Type MOSFET Explained (Construction, Working and Characteristics Explained)**  **MOSFET vth based problems** | **Semester & Section:** | **8 A** |
| **Github Repository:** | **namrathahipparagi\_1** |  |  |

|  |
| --- |
| **FORENOON SESSION DETAILS** |
| **Image of session** |
| **Report**  **Advantages of using transmission gate logic.**   * A CMOS transmission gate can be constructed by parallel combination of nMOS and pMOS transistors, with complementary gate signals. * It allows full rail transition i.e. ratioless logic * The equivalent resistance is relatively constant during transition. * The main advantage of CMOS transmission gate compared to nMOS transmission gate is to allow the input signal to be transmitted to the output without the threshold voltage attenuation. * Some gates are efficient implemented using transmission gate.   **4:1 mux using pass transistor logic**   **MOS Field-Effect-Transistors**  |  |  | | --- | --- | | **The linear model** |  |  |  | | --- | | The linear model describes the behavior of a MOSFET biased with a small drain-to-source voltage. As the name suggests, the linear model, describes the MOSFET acting as a linear device. More specifically, it can be modeled as a linear resistor whose resistance is modulated by the gate-to-source voltage. In this regime, the MOSFET can be used as a switch for analog and digital signals or as an analog multiplier. |  |  | | --- | | The general expression for the drain current equals the total charge in the inversion layer divided by the time the carriers need to flow from the source to the drain: |  |  |  | | --- | --- | |  |  |  |  | | --- | | where *Q*inv is the inversion layer charge per unit area, *W* is the gate width, *L* is the gate length and *t*r is the transit time. If the velocity of the carriers is constant between source and drain, the transit time equals: |  |  |  | | --- | --- | |  |  |  |  | | --- | | where the velocity, *v*, equals the product of the mobility and the electric field: |  |  |  | | --- | --- | |  |  |  |  | | --- | | The constant velocity also implies a constant electric field so that the field equals the drain-source voltage divided by the gate length. This leads to the following expression for the drain current: |  |  |  | | --- | --- | |  |  |  |  | | --- | | We now assume that the charge density in the inversion layer is constant between source and drain. We also assume that the basic assumption described in section [6.3.2](https://ecee.colorado.edu/~bart/book/book/chapter6/ch6_3.htm#6_3_2) applies, namely that the charge density in the inversion layer equals minus the product of the capacitance per unit area and the gate-to-source voltage minus the threshold voltage: |  |  |  | | --- | --- | |  |  |  |  | | --- | | The inversion layer charge is zero if the gate voltage is lower than the threshold voltage. Replacing the inversion layer charge density in the expression for the drain current yields the linear model: |  |  |  | | --- | --- | |  |  |  |  | | --- | | Note that the capacitance in the above equations is the gate oxide capacitance per unit area. Also note that the drain current is zero if the gate-to-source voltage is less than the threshold voltage. The linear model is only valid if the drain-to-source voltage is much smaller than the gate-to-source voltage minus the threshold voltage. This insures that the velocity, the electric field and the inversion layer charge density is indeed constant between the source and the drain. |  |  |  | | --- | --- | | An example of the linear current-versus-voltage (*I*-*V*) characteristics of a MOSFET | | | **The quadratic model** |  |  |  | | --- | | The quadratic model uses the same assumptions as the linear model. However, this model allows the inversion layer charge to vary between the source and the drain. |  |  | | --- | | The derivation is based on the fact that the current is continuous throughout the channel. The current is also related to the local channel voltage, >I>VC. |  |  | | --- | | We now consider a small section within the device with width *dy>/I> and channel voltage VC + VS. The linear model as described by equation (*[*7.3.6*](https://ecee.colorado.edu/~bart/book/book/chapter7/ch7_eq.htm#eq7_3_6)*), still applies to such section, yielding:* |  |  |  | | --- | --- | |  | ) |  |  | | --- | | where the drain-source voltage is replaced by the channel voltage. Both sides of the equation can be integrated from the source to the drain, so that *y* varies from 0 to the gate length, *L*, and the channel voltage *VC* varies from 0 to the drain-source voltage, *V*DS. |  |  |  | | --- | --- | |  |  |  |  | | --- | | The drain current, *ID*, is constant so that integration results in: |  |  |  | | --- | --- | |  |  |  |  | | --- | | The drain current first increases linearly with the applied drain-to-source voltage, but then reaches a maximum value. According to the above equation the current would even decrease and eventually become negative. The charge density at the drain end of the channel is zero at that maximum and changes sign as the drain current decreases. As, the charge in the inversion layer does go to zero and reverses its sign as holes are accumulated at the interface. However, these holes cannot contribute to the drain current since the reversed-biased p-n diode between the drain and the substrate blocks any flow of holes into the drain. Instead the current reaches its maximum value and maintains that value for higher drain-to-source voltages. A depletion layer located at the drain end of the gate accommodates the additional drain-to-source voltage. This behavior is referred to as *drain current saturation*. |  |  | | --- | | Drain current saturation therefore occurs when the drain-to-source voltage equals the gate-to-source voltage minus the threshold voltage. The value of the saturated drain current, *ID,sat*. is then given by the following equation: |  |  |  | | --- | --- | |  |  |  |  | | --- | | The quadratic model explains the typical current-voltage characteristics of a MOSFET, which are normally plotted for different gate-to-source voltages. An example is shown in Figure [7.3.2](https://ecee.colorado.edu/~bart/book/book/chapter7/ch7_3.htm#fig7_3_2). The saturation occurs to the right of the dotted line which is given by *I*D = m *C*ox *W*/*L* *V*DS2. |      |  | | --- | |  | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Date:** | **09/6/2020** | **Name:** | **Namratha S Hipparagi** | |
| **Course:** | **Java** | **USN:** | **4al16ec040** | |
| **Topic:** | **Loops**  **Switches** | **Semester & Section:** | **8 A** | |
| **AFTERNOON SESSION DETAILS** | | | |
| **REPORT** **Switch Statement in Java** A **switch statement** allows a variable to be tested for equality against a list of values. Each value is called a case, and the variable being switched on is checked for each case.  The switch statement is a multi-way branch statement. It provides an easy way to dispatch execution to different parts of code based on the value of the expression. Basically, the expression can be byte, short, char, and int primitive data types. Beginning with JDK7, it also works with enumerated types ( [Enums](https://www.geeksforgeeks.org/enum-in-java/) in java), the [String](https://www.geeksforgeeks.org/string-class-in-java/) class and [Wrapper](https://www.geeksforgeeks.org/primitive-wrapper-classes-are-immutable-in-java/) classes.  **Syntax of Switch-case :**  // switch statement  switch(expression)  {  // case statements  // values must be of same type of expression  case value1 :  // Statements  break; // break is optional    case value2 :  // Statements  break; // break is optional    // We can have any number of case statements  // below is default statement, used when none of the cases is true.  // No break is needed in the default case.  default :  // Statements  } The Do While Loop Java do-while loop is used to execute a block of statements continuously **until** the given condition is true. The do-while loop in Java is similar to while loop except that the condition is checked after the statements are executed, so do while loop guarantees the loop execution at least once. The do-while loop is one of the most common constructs in programming. Versions of it can be found in virtually every programming language. By itself, the loop is fairly straightforward. It tells the computer: “Do these things as long as these conditions are true”. There is one crucial difference between the Do-While and While loop. In the While loop, the action is performed only if the condition is true. A Do-While loop, on the other hand, will run at least once because the body (Do) is executed before the condition (While) is tested.  do {  something  } while (condition)  Thus, the something part is always performed at least once before the program moves on to testing the condition provided under While.  do {  statement1;  statement2; //and so on  } while (condition);  This simple program from the official Java tutorials counts to 10:  class LoopDemo {  public static void main(String[] args) {  int count = 1;  do {  System.out.println(“Count is: “ + count);  count++;  } while (count < 11);  }  } | | | |
|  | | | |