**DAILY ASSESSMENT REPORT**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date:** | **13/06/2020** | **Name:** | **GaganashreeP** |
| **Course:** | **Vlsi** | **USN:** | **4al15EC024** |
| **Topic:** | **Digital VLSI Design Virtual lab** | **Semester & Section:** | **8th &"A" section** |
| **Github Repository:** | **Gaganashree-P** |  |  |
| **FORENOON SESSION DETAILS** | | | |
| **Image of session** | | | |
| **Report–**  Complementary MOSFET (CMOS) technology is widely used to day to form circuits in numerous and varied applications.  Today’s computers CPUs and cell phones make use of CMOS due to several key advantages.  CMOS offers low power dissipation, relatively high speed, high noise margins in both states, and will operate over a wide range of source and input voltages (provided the source voltage is fixed).  Next I will attempt to explain just how this logic gate works now that you have some idea of how important CMOS is in your day-to-day life.         |  | | --- | | **CMOS Inverter Basics** |   As you can see from Figure 1, a CMOS circuit is composed of two MOSFETs.  The top FET (MP) is a PMOS type device while the bottom FET (MN) is an NMOS type.  The body effect is not present in either device since the body of each device is directly connected to the device’s source.  Both gates are connected to the input line.  The output line connects to the drains of both FETs.    Take a look at the VTC in Figure 2.  The curve represents the output voltage taken from node 3.  You can easily see that the CMOS circuit functions as an inverter by noting that when VIN is five volts, VOUT is zero, and vice versa. Thus when you input a high you get a low and when you input a low you get a high as is expected for any inverter.  You might be wondering what happens in the middle, transition area of the curve.  You might also be curious as to what modes of operation the MOSFETs are in.  We will look at these issues next.       |  |  | | --- | --- | | https://courseware.ee.calpoly.edu/~dbraun/courses/ee307/F02/02_Shelley/cmos_files/image002.gif                            Figure1:  CMOS Inverter | https://courseware.ee.calpoly.edu/~dbraun/courses/ee307/F02/02_Shelley/cmos_files/image004.jpg                Figure 2:  Basic Voltage Transfer Characteristic |          |  | | --- | | **DC Analysis** |   Figure 3 shows a more detailed VTC.  Before we begin our analysis it is important to mention three items.     * The MOSFETS must be perfectly matched for optimum operation, that is, they must have the same threshold voltage magnitude and conduction parameter. * The drain current (ID) through the NMOS device equals the drain current through the PMOS device at all times.  MOSFET gates have a high input impedance and we assume the circuit’s output sees no significant loading. * VDD equals the voltage across the PMOS plus the voltage across the NMOS by KVL.     https://courseware.ee.calpoly.edu/~dbraun/courses/ee307/F02/02_Shelley/cmos_files/image006.gif  **Figure 3:  VTC with Input Signal**    **Region I**    First we focus our attention on **region I**.  In this case when we apply an input voltage between 0 and VTN.  The PMOS device on since a low voltage is being applied to it.  The NMOS is already negative enough and has no use for more free electrons so it refuses to conduct and turns into a large resistor.  Since the NMOS device is on vacation, there is no current flow through either device.  VDD is available at the Vo terminal since no current is going through the PMOS device and thus no voltage is being dropped across it.     * The PMOS device is forward biased (VSG > -VTP) and therefore on.  This MOSFET is in the linear region (VSD<=VSG+VTP=VDD-Vo+VTP). * The NMOS device is cut off since the input voltage is below VTN (Vi=VGS<VTN). * The power dissipation is zero.     **Region II**    Here we raise the input voltage above VTN.  We find that the PMOS device remains in the linear region since it still has adequate forward bias.  The NMOS turns on and jumps immediately into saturation since it still has a relatively large VDS across it.     * The PMOS device is in the linear region (VSD<=VSG+VTP). * The NMOS device is in the saturation region (Vi=VDS>=VGS-VTN=Vo-VTN). * Current now flows through both devices.  Power dissipation is no longer zero.     The maximum allowable input voltage at the low logic state (**VIL**) occurs in this region.  VIL is the value of Vi at the point where the slope of the VTC is -1.  Put another way, VIL occurs at (dVo/dVi)=-1.    **Region III**    In the middle of this region there exists a point where Vi=Vo.  We label this point VM and identify it as the gate threshold voltage.  The voltage dropped across the NMOS device equals the voltage dropped across the PMOS device when the input voltage is VM.  For a very short time, both devices see enough forward bias voltage to drive them to saturation.     * The PMOS device is in the saturation region (VSD>=VSG+VTP=VDD-Vo+VTP). * The NMOS device is in the saturation region (VDS>=VGS-VTN=Vo-VTN). * Power dissipation reaches a peak in this region, namely at where VM=Vi=Vo.     **Region IV**    Region IV occurs between an input voltage slightly higher than VM but lower than VDD-VTP.  Now the NMOS device is conducting in the linear region, dropping a low voltage across VDS.  Since VDS is relatively low, the PMOS device must pick up the tab and drop the rest of the voltage (VDD-VDS) across its VSD junction.  This, in turn, drives the PMOS into saturation.  This region is effectively the reverse of region II.     * The PMOS device is in the saturation region (VSD>=VSG+VTP=VDD-Vo+VTP). * The NMOS device is forward biased (Vi=VGS > VTN) and therefore on.  This MOSFET is in the linear region (Vi=VDS<=VGS-VTN=Vo-VTN).     The minimum allowable input voltage at the logic high state (**VIH**) occurs in this region.  VIH occurs at the point where the slope of the VTC is –1 (dVo/dVi)=-1.    **Region V**    The NMOS wants to conduct but its drain current is severely limited due to the PMOS device only letting through a tiny leakage current.  The PMOS is out to lunch since it is seeing a positive drive but it is already positive enough and has no use for more.  This drain current let through by the PMOS is too small to matter in most practical cases so we let ID=0.  With this information we can conclude that VDS=Vo=0 V for the NMOS since no current is going through the device.  We have, in effect, sent in VDD and found the inverter’s output to be zero volts.  **For CMOS inverters, VOH=VDD.**  VOL is defined to be the output voltage of the inverter at an input voltage of VOH.  We have just proven that **VOL=0**.     * The PMOS device is cut off when the input is at VDD (VSG=0 V). * The NMOS device is forward biased (Vi=VGS > VTN) and therefore on.  This MOSFET is in the linear region (Vi=VDS<=VGS-VTN). * The total power dissipation is zero just as in region I.          |  | | --- | |  |   Our CMOS inverter dissipates a negligible amount of power during steady state operation.  Power dissipation only occurs during switching and is very low.  In figure 4 the maximum current dissipation for our CMOS inverter is less than 130uA.  Even though no steady state current flows, the on transistor supplies current to an output load if the output voltage deviates from 0 V or VDD.  This makes CMOS technology useable in low power and high-density applications.      https://courseware.ee.calpoly.edu/~dbraun/courses/ee307/F02/02_Shelley/cmos_files/image008.gif  Figure – Drain Current Verses Input Voltage | | | |

**DAILY ASSESSMENT FORMAT**

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  |  |  | | --- | --- | --- | --- | | Date: | 13-06-20 | Name: | GaganashreeP | | Course: | MySQL | USN: | 4AL15EC024 | | Topic: | Real Life PHP Introduction  About The Author | Semester & Section: | 8th A | | Github Repository: | Gaganashree-P |  |  |  |  |  |  |  | | --- | --- | --- | --- | |  |  |  |  | |  |  |  |  | |  |  |  |  | | **AFTERNOON SESSION DETAILS** | | | | | **Image of session** | | | | | **Report – PHP** started out as a small open source project that evolved as more and more people found out how useful it was. Rasmus Lerdorf unleashed the first version of PHP way back in 1994.   * PHP is a recursive acronym for "PHP: Hypertext Preprocessor". * PHP is a server side scripting language that is embedded in HTML. It is used to manage dynamic content, databases, session tracking, even build entire e-commerce sites. * It is integrated with a number of popular databases, including MySQL, PostgreSQL, Oracle, Sybase, Informix, and Microsoft SQL Server. * PHP is pleasingly zippy in its execution, especially when compiled as an Apache module on the Unix side. The MySQL server, once started, executes even very complex queries with huge result sets in record-setting time. * PHP supports a large number of major protocols such as POP3, IMAP, and LDAP. PHP4 added support for Java and distributed object architectures (COM and CORBA), making n-tier development a possibility for the first time. * PHP is forgiving: PHP language tries to be as forgiving as possible. * PHP Syntax is C-Like.   **Common uses of PHP**   * PHP performs system functions, i.e. from files on a system it can create, open, read, write, and close them. * PHP can handle forms, i.e. gather data from files, save data to a file, through email you can send data, return data to the user. * You add, delete, modify elements within your database through PHP. * Access cookies variables and set cookies. * Using PHP, you can restrict users to access some pages of your website. * It can encrypt data.   **Characteristics of PHP**  Five important characteristics make PHP's practical nature possible −   * Simplicity * Efficiency * Security * Flexibility * Familiarity   PHP is a server-side scripting language, mainly used for web development but also used as a general-purpose programming language. Object-Oriented Programming (PHP OOP), is a type of programming language principle added to php5 that helps in building complex, reusable web applications.  The Object Oriented concepts in PHP are:   * Class − This is a programmer-defined data type, which includes local functions as well as local data. You can think of a class as a template for making many instances of the same kind (or class) of object. * Object − An individual instance of the data structure defined by a class. You define a class once and then make many objects that belong to it. Objects are also known as instance. * Inheritance − When a class is defined by inheriting existing function of a parent class then it is called inheritance. Here child class will inherit all or few member functions and variables of a parent class. * Polymorphism − This is an object oriented concept where same function can be used for different purposes. For example function name will remain same but it make take different number of arguments and can do different task. * Overloading − a type of polymorphism in which some or all of operators have different implementations depending on the types of their arguments. Similarly functions can also be overloaded with different implementation. * Data Abstraction − Any representation of data in which the implementation details are hidden (abstracted). \* Encapsulation − refers to a concept where we encapsulate all the data and member functions together to form an object. * Constructor − refers to a special type of function which will be called automatically whenever there is an object formation from a class. * Destructor − refers to a special type of function which will be called automatically whenever an object is deleted or goes out of scope.   Picking up the right **CMS** is the most vital decision when you are starting a web development process. Because it lets you to plan your app requirements and its solutions in advance. In the way, you get the idea about your proposed app design, features and performance. Therefore, opting the best PHP CMS is the most important part of your app development cycle. And hence requires appropriate knowledge to choose the best one.  While today we have many open source CMS available in the market. The choice depends upon our app requirements. Whether you want to build a simple dynamic website or a fully functional ecommerce store. There are multiple types of CMSs available in the market which could lessen your workload. As not every developer wants to write orthodox HTML and CSS to build web pages.  Joomla is an open source [content management system](https://en.wikipedia.org/wiki/Content_management_system) (CMS). It helps you build powerful dynamic websites and applications. It has an intuitive interface that helps you use its features and functionality to the fullest.  Joomla has gained huge popularity over the last decade and successfully grown into one of the most widely used content management systems globally. Since its inception in 2005, it has successfully captured the massive following and it certainly has over 99 + million downloads till date.  Joomla is written in [PHP](https://blog.templatetoaster.com/php-7-tutorial-features/) and use MySQL database to store the data while using object-oriented programming techniques. It can be set up with one-click install through web hosting control panel. There are hundreds of articles exist on the web to help you with [how to install Joomla](https://blog.templatetoaster.com/how-to-install-joomla/).  Joomla uses [Model-View-Controller](https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller) (MVC) design architecture. According to the MVC pattern when Joomla process a request, it first analyzes the URL to evaluate which component will process the request. The model contains the data used by the component. It is also the Model’s responsibility to update the database when and where required. The view is accountable for producing the output. It can contact with the model to get the needed information. After the view has produced the output, the component gives back the control to the Joomla framework which then executes the template | | | | |