






RUTGERS UNIVERSITY  
Department of Electrical and Computer Engineering  
14:332:479 Introduction to VLSI Design  
Assignment I  
Assigned: September 13, 2004  
Due September 22, 2004

**Reading Assignment:** Chapters 1 and 2 of Weste and Eshraghian.

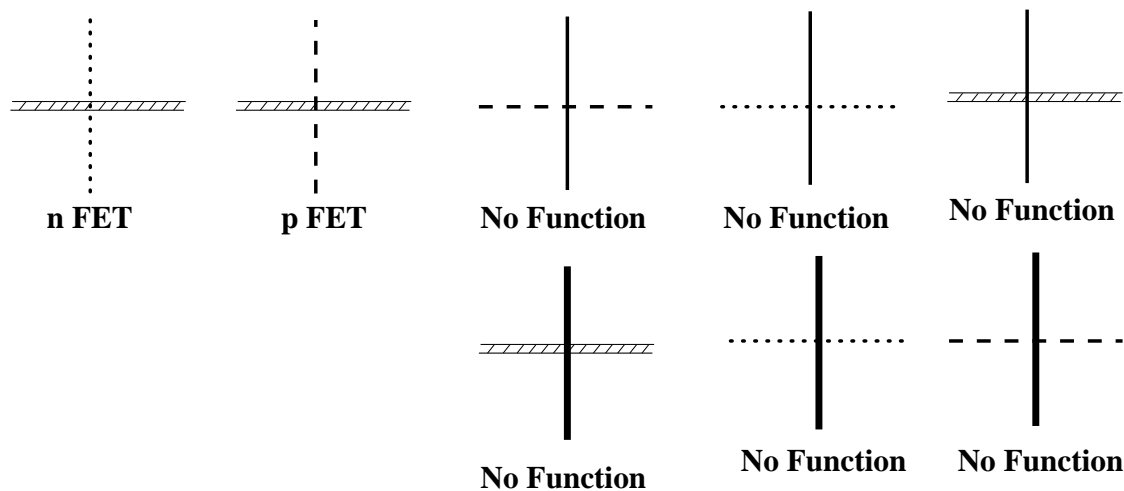
No collaboration is permitted on this assignment. Your work must be your own.

This assignment introduces *sticks diagrams*, which give an abstract view of designing MOS VLSI circuits. The abstraction is surprisingly accurate in topological and electrical (logical) respects. It lacks any sense of reality about geometry and performance, but it simplifies the difficult process of translating a 3-dimensional transistor circuit representation onto a 2-dimensional chip surface.

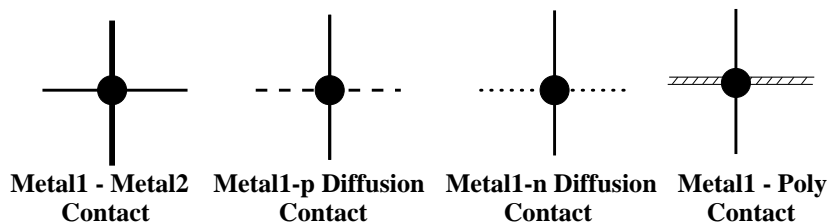
Five kinds of paths can be fabricated, all of which carry current: Metal1 (blue), Metal2 (purple), polysilicon (red), *n*-channel diffusion (green), and *p*-channel diffusion (yellow). Invest in colored pencils and graph paper for drawing the circuits. Paths in different colors may cross. All crossings

<b>Metal1 (blue)</b>	
<b>Metal2 (purple)</b>	
<b>Poly (red)</b>	
<b>N-channel diffusion (green)</b>	
<b>P-channel diffusion (yellow)</b>	

save three are non-functional (no connection is made between the crossing paths). A red/green crossing produces an *n*-channel enhancement transistor (a switch), a red/yellow crossing produces a *p*-channel enhancement transistor (a switch), where red is the control input. Green/yellow crossings are prohibited.



It is possible to form connections between different levels with special “contacts” represented as black disks:



Review the elementary CMOS NAND, NOT, and NOR gate layouts in the inside cover of the text, and read Section 1.4. The questions on the next page ask you to design circuits using these gates; connection paths of red, green, yellow, and blue; contacts; and switches constructed of red/green and red/yellow crossings. Note that you cannot (in this technology) create a contact between green and yellow, and that these two layers must never cross. If lines (wires) cross on the same layer (color), then the wires are shorted together. However, wires carrying different signals can run parallel on the same layer, and will not short circuit.

Concentrate on the topological aspects of the problems. If you want to minimize something, minimize first the number of contacts (black disks); next the “size” of the design (measured in grid squares, where a grid unit is the minimum distance between any two paths in the SAME level). Please note that there is no virtue in using a small number of switches. For the following problems, I will return your homework with a 0 score unless you follow these rules in your sticks layouts:

1. All wires, contacts, and switches must lie on grid lines on your graph paper, not in the space between grid lines.
2. Your graph paper must use a  $0.25 \text{ inch}^2$  grid.
3. It is possible to run certain layers together on top of a single grid line. An example is that the metal1 and metal2 layers may run on top of each other. This causes bad capacitive coupling if you do this for long distances, so this is only permitted for short distances. In this case, draw the lines slightly apart so that I can read your sticks diagram.
4. Your graph paper must be linear. I will not accept logarithmic paper or polar coordinate paper.

The fastest way to draw these diagrams is to work out a rough sketch with colored pens on graph paper, and then correct errors by notating them with a pencil. Then, recopy your sketch onto a clean sheet of graph paper, compacting the sticks layout as you go. If you make errors on your final copy, or want to change it to compact it further, use white-out fluid to erase the errors and then redraw the corrections. Otherwise, you will spend hours endlessly recopying the sticks layout.

Questions to be answered and handed in:

1. (**Static Full-Custom CMOS Design.**) Design a **single** custom static CMOS gate (only one gate delay) that implements the function **(NOT (A AND (B OR C) AND D))**. Turn in a MOS transistor schematic and a sticks diagram. A zero is given to anyone who uses multiple CMOS gates.
2. (**Static Multiple Standard Gate CMOS Design.**) Implement the function **((NOT A) XOR B) OR (B AND (NOT C))** using multiple basic 2-input CMOS NOT, NAND, and NOR gates. Turn in a logic gate schematic, a MOS transistor schematic, and a sticks diagram. A zero is given to anyone who uses a single custom CMOS gate. How could you speed up this gate while using less chip area?
3. (**CMOS MUX Design.**) Consider the *4-way selector* function: the data inputs are **S0, S1, S2, S3**; the selection inputs are **A0** and **A1**. The selector function is given by the following table:

<b>A0</b>	<b>A1</b>	<b>Output</b>
0	0	<b>S0</b>
0	1	<b>S1</b>
1	0	<b>S2</b>
1	1	<b>S3</b>

Design a circuit that implements this function using only CMOS transmission gates. Note the drastic reduction in area requirements over an implementation that uses static CMOS gates. Turn in a transistor schematic and the smallest sticks diagram of the circuit that you can create. When you draw the sticks diagram, please draw all *polysilicon* transistor gates as vertical RED lines on the vertical axis. Draw all CMOS transistor channels as horizontal lines in *Metal1*. Program the transistors by replacing *Metal1* with *n* or *p diffusion* where you want a transistor to be. Separate the *n* and *p* transistors into different areas of your sticks layout. Turn in a transistor layout and a sticks diagram for the circuit.