


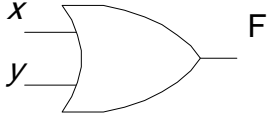
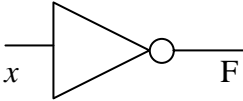
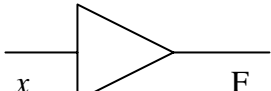
EE200 DIGITAL LOGIC CIRCUIT DESIGN

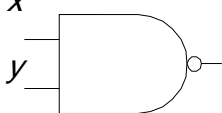
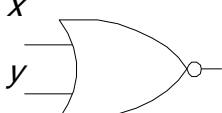


The material covered in this class will be as follows:

- ⇒ Digital logic gates
- ⇒ Extension to multiple inputs
- ⇒ Positive and negative logic
- ⇒ Integrated circuits

Digital Logic Gates

Commonly used digital logic gates are given in the following table.

Name	Graphic Symbol	Algebraic Function	Truth Table															
AND		$F = xy$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>0</td></tr><tr><td>1</td><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	x	y	F	0	0	0	0	1	0	1	0	0	1	1	1
x	y	F																
0	0	0																
0	1	0																
1	0	0																
1	1	1																
OR		$F = x + y$	<table><tr><th>x</th><th>y</th><th>F</th></tr><tr><td>0</td><td>0</td><td>0</td></tr><tr><td>0</td><td>1</td><td>1</td></tr><tr><td>1</td><td>0</td><td>1</td></tr><tr><td>1</td><td>1</td><td>1</td></tr></table>	x	y	F	0	0	0	0	1	1	1	0	1	1	1	1
x	y	F																
0	0	0																
0	1	1																
1	0	1																
1	1	1																
Inverter		$F = x'$	<table><tr><th>x</th><th>F</th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	x	F	0	1	1	0									
x	F																	
0	1																	
1	0																	
Buffer		$F = x$	<table><tr><th>x</th><th>F</th></tr><tr><td>0</td><td>0</td></tr><tr><td>1</td><td>1</td></tr></table>	x	F	0	0	1	1									
x	F																	
0	0																	
1	1																	

NAND		$F = (xy)'$	<table><tr><th>xy</th><th>F</th></tr><tr><td>00</td><td>1</td></tr><tr><td>01</td><td>1</td></tr><tr><td>10</td><td>1</td></tr><tr><td>11</td><td>0</td></tr></table>	xy	F	00	1	01	1	10	1	11	0
xy	F												
00	1												
01	1												
10	1												
11	0												
NOR		$F = (x + y)'$	<table><tr><th>xy</th><th>F</th></tr><tr><td>00</td><td>1</td></tr><tr><td>01</td><td>0</td></tr><tr><td>10</td><td>0</td></tr><tr><td>11</td><td>0</td></tr></table>	xy	F	00	1	01	0	10	0	11	0
xy	F												
00	1												
01	0												
10	0												
11	0												
XOR		$F = xy' + x'y$	<table><tr><th>xy</th><th>F</th></tr><tr><td>00</td><td>0</td></tr><tr><td>01</td><td>1</td></tr><tr><td>10</td><td>1</td></tr><tr><td>11</td><td>0</td></tr></table>	xy	F	00	0	01	1	10	1	11	0
xy	F												
00	0												
01	1												
10	1												
11	0												
XNOR		$F = xy + x'y'$	<table><tr><th>xy</th><th>F</th></tr><tr><td>00</td><td>1</td></tr><tr><td>01</td><td>0</td></tr><tr><td>10</td><td>0</td></tr><tr><td>11</td><td>1</td></tr></table>	xy	F	00	1	01	0	10	0	11	1
xy	F												
00	1												
01	0												
10	0												
11	1												

Extension to Multiple Inputs

The commonly used gates, except the buffer and the inverter, can be extended to have more than two inputs if the binary operation is commutative and associative (e.g. AND, OR).

NAND and NOR are commutative but not associative

$$\left[(x + y)' + z \right]' \neq \left[x + (y + z)' \right]'$$

We define the multiple NOR and NAND gate as a complemented OR or AND gate.

e.g. three variable NOR gate $\rightarrow (x + y + z)'$.

Integrated Circuits

Integrated circuits are classified based on the levels of integration.

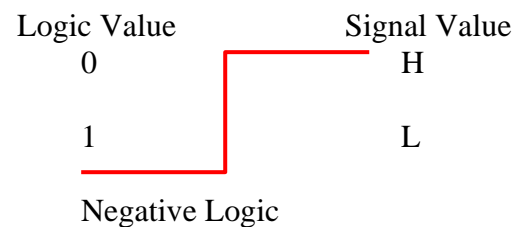
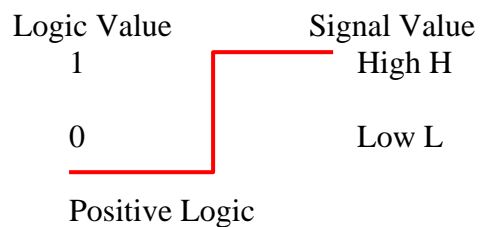
SSI < 10 gates

MSI 10 – 100 gates (e.g., decoders, adders, MUXs)

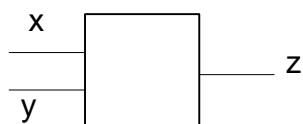
LSI 100 – few 1000s (e.g., processors, memory chips)

VLSI > 1000s

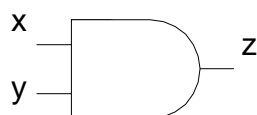
Positive and Negative Logic



Demonstration of positive & Negative Logic

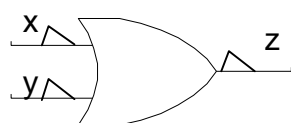


x	y	z
L	L	L
L	H	L
H	L	L
H	H	H



Positive Logic AND Gate

x	y	z
0	0	0
0	1	0
1	0	0
1	1	1



Negative Logic OR Gate

x	y	z
1	1	1
1	0	1
0	1	1
0	0	0