EECS-140/141 Introduction to Digital Logic Design
Lecture 3: Introduction to Gate Technology

I. TRANSISTOR SWITCHES AND GATES

I.A MOS Switches

Switches used for digital electronics are *MOSFETs*: Metal Oxide Semiconductor Field Effect Transistors.

3 Terminals: Gate (G), Source (S), Drain (D)

2 Basic Types: n-channel (NMOS) and p-channel (PMOS)

You will learn more about these in EECS 312.

Diagrams:

NMOS Switch:

PMOS Switch:

I.B NMOS Logic Gates

I.B.1 NOT Gate

NMOS switch by itself is *almost* a NOT gate: just need to:

V_x	T_1	V_f	Current Thru T_1
Lo			
Hi			

Truth Table: Lo=0 and Hi=1

x	$f = \bar{x}$
0	
1	

I.B.2 AND Gate

Switches in *series*.

V_{x_1}	V_{x_2}	V_N	V_A	Current Flows Thru
Lo	Lo			
Lo	Hi			
Hi	Lo			
Hi	Hi			

x_1	x_2	$f = x_1 \cdot x_2$
0	0	
0	1	
1	0	
1	1	

I.B.3 NAND Gate

This is a new logic function equivalent to AND *followed* by NOT.

So NAND followed by NOT is AND: this is how we implemented AND above with NMOS.

So, we can build a NAND gate as above, except:

V_{x_1}	V_{x_2}	V_N	Current Thru T_1 and T_2?
Lo	Lo		
Lo	Hi		
Hi	Lo		
Hi	Hi		

NAND Truth Table

x_1	x_2	$f = \overline{(x_1 \cdot x_2)}$
0	0	
0	1	
1	0	
1	1	

I.B.4 NOR Gate

Similar to NAND, this is OR *followed* by NOT.

We would put the switches in *parallel*.

V_{x_1}	V_{x_2}	V_{NOR}	Current Thru:
Lo	Lo		
Lo	Hi		
Hi	Lo		
Hi	Hi		

NOR continued...

NOR Truth Table

x_1	x_2	$f = \overline{(x_1 + x_2)}$
0	0	
0	1	
1	0	
1	1	

I.B.5 OR Gate

We can convert NOR to OR by just adding an inverter after the NOR, as we did to convert NAND to AND.

Again, NOR requires:

I.B.6 Third New Gate: eXclusive-OR (XOR)

This will be useful later in adder circuits.

Truth Table:

x_1	x_2	$f = x_1 \quad x_2$
0	0	
0	1	
1	0	
1	1	

Logic Symbol:

I.C CMOS Logic Gates

I.C.1 Motivation:

Problem: Outputs of NMOS gates must be "tied" to Hi voltage via *R* to "pull-up" to Hi when NMOS Pull-Down-Network (PDN) is *open*.

Result: When NMOS PDN is *closed*, current flows through *R* and PDN. This consumes power/energy.

Solution: Replace pull-up *resistor* with Pull-Up-*Network* (PUN) of *P*MOS transistors arranged such that PUN is open when PDN is closed, and vice versa. *Result*: no current flow, so *much* less power consumed.

I.C.2 CMOS NOT Gate

V_x	T_D	T_U	V_f	Current
Lo				
Hi				

I.C.3 CMOS NAND Gate

V_{x_1}	V_{x_2}	T_{U_1}	T_{U_2}	PUN	T_{D_1}	T_{D_2}	PDN	V_f	Current
Lo(0)	Lo(0)								
Lo(0)	Hi(1)								
Hi(1)	Lo(0)								
Hi(1)	Hi(1)								

Similar for CMOS NOR.

Again, we make an AND gate by following NAND with NOT (*two* more transistors), and the same for making an OR from a NOR.

Note: If NMOS gate requires N transistors, CMOS version will *always* require $2N$!

This is a high cost in transistor count, but *worth* it because of power savings!

II. USING NEW GATES (NAND, NOR)

II.A NAND-Only Networks for SoP Synthesis

II.A.1 Why?

For custom fabrication of integrated circuits (ICs): from previous section, AND and OR require *more* transistors than NAND and NOR (1 more for NMOS, 2 more for CMOS).

For circuits using pre-packaged gates: several of the *same* type are packaged together, so using NAND-only or NOR-only usually requires fewer *packages* (because fewer gates are "wasted").

Handout of quad 2-input NAND gate.

II.A.2 NAND Equivalents

By DeMorgan:

So:

Also, since $x \cdot x = x$, then $\overline{x \cdot x} = \bar{x}$, so:

II.A.3 Converting SoP to NAND-Only

SoP always has several "first-level" AND gates connected to single "second-level" OR gate.

Following "trick" uses $\overline{\bar{x}} = x$:

Converting SoP to NAND-Only (continued):

Direct input to NOR? Still double-invert:

II.B NOR-only Networks for PoS Synthesis

II.B.1 Why?

Same as for NAND-only.

II.B.2 NOR Equivalents

By DeMorgan:

So:

and by $x + x = x$, then $\overline{(x + x)} = \bar{x}$, so:

II.B.3 Converting PoS to NOR-Only

Same trick: double-invert:

