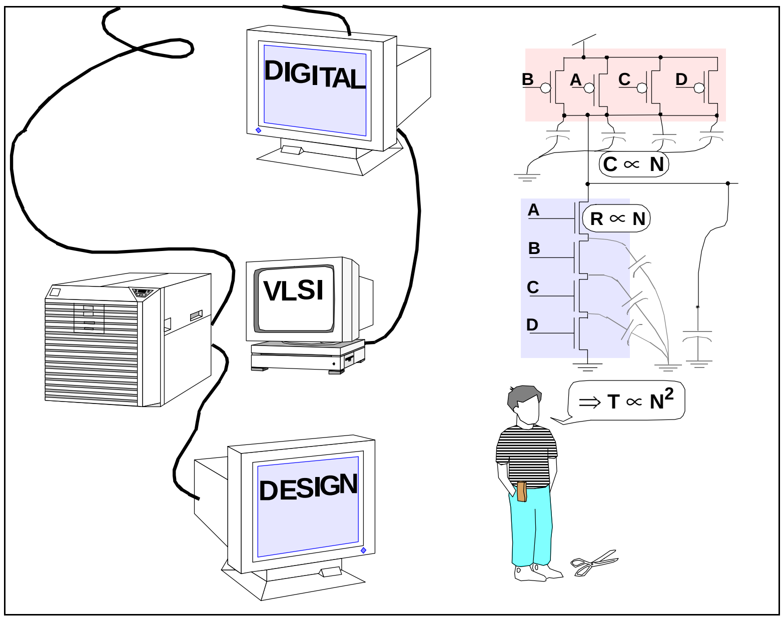
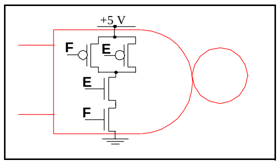
Digital VLSI Design



Carleton 2003

Good Afternoon

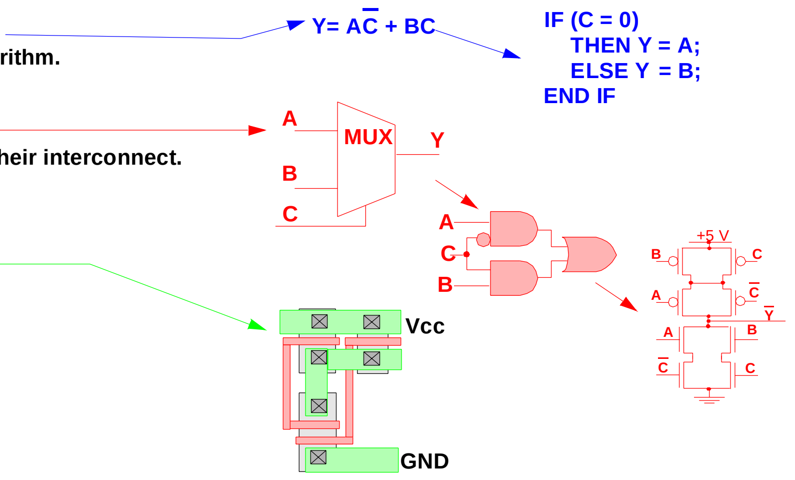
Digital Design



Abstraction

Hiding postponable design details inside higher level model.
Digital is more successful in this than analog.

The Three Axes for Digital Design Methods



IF (C = 0)
    THEN Y = A;
    ELSE Y = B;
END IF

Behavioural

Describes the algorithm.

Structural

Components and their interconnect.

Physical

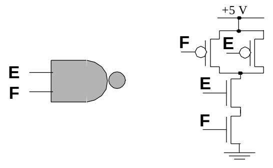
How it is built.

The Universe of Digital Abstraction

Abstraction

A model which hides postponable details so that Engineers can think on a grander scale.

The Gate Abstraction



The most common digital abstraction is the gate.

- It allows us to think only of 1 and 0 signals.
- We ignore real voltage levels, real switching thresholds.
- We lump timing into a few delays, T_{rise}, T_{fall}, T_{prop}

Three Axes for Design Methods

There are different levels of abstraction along each axes

Behavioural

Think about the result and the algorithm to produce it; ignore internal components; ignore their interconnections.

Structural

Think about the components and how they are interconnected; ignore how they are actually built.

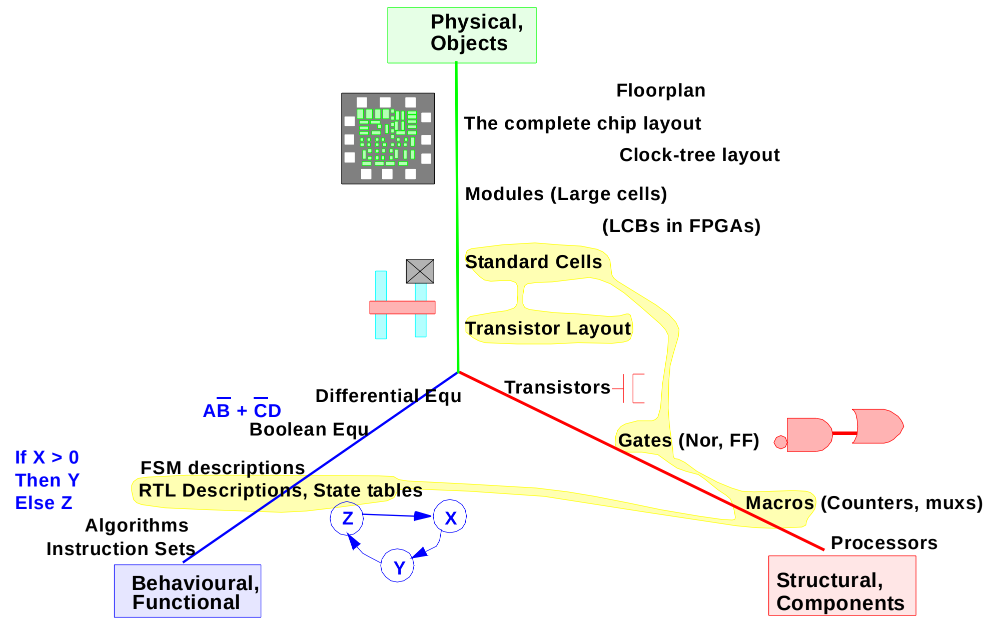
Mux symbol most abstract, gate design medium, the transistor mux design is least abstract.

Physical

Think about how it is built.

The Universe of Digital Abstraction

The Y Diagram



The Universe of Digital Abstraction

The Y Diagram

Abstraction

The further away from the origin one gets, the more abstract the concept.

Compare: the transistor MUX, the gate MUX, and the symbol for the MUX.

The Path for a Computer Aided Design (CAD) Tool

Most tools

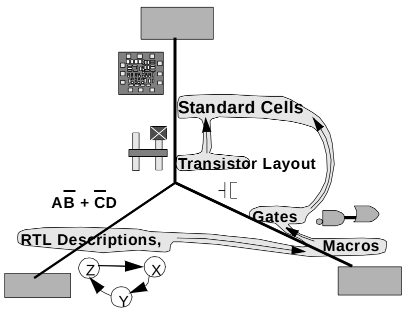
- start in the behavioural axis,.
- transform the design to the structural axis.
- travel down the structural axis.
- Finally transfer over to layout on the physical axis.

However floorplanning goes from the structural at a high level to physical at a high level.

Clock tree design is physical.

Path for Most Design in This Course, the Digital Part

1. Start at a behavioural RTL Description, (Register Transfer Level).
2. Go over to macros and/or gates on the structural axis
3. Then go to the physical axis to predesigned standard cells.
4. The cells were designed bottom up from transistors.



Other Things Besides Raw Circuit Design

Other Things

Circuit Design

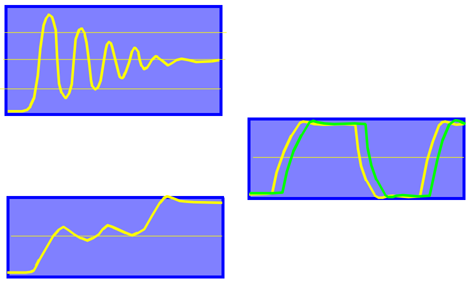
1. Transmission line effects.
2. Clock distribution, skew.
3. Asynchronous inputs, metastability.
4. Verification.

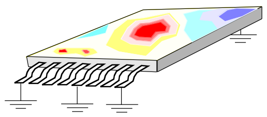
Production Issues

1. Production Testing.
2. Packaging, Cooling.
3. Reliability
4. Pads, connections, protection.
5. Power Supply.

Group Dynamics

1. Incomplete fuzzy specification.
2. Division of labour, communication!!!
3. Design style, coding style. Reusability, documentation.





The Under-Advertised Aspects of VLSI Design

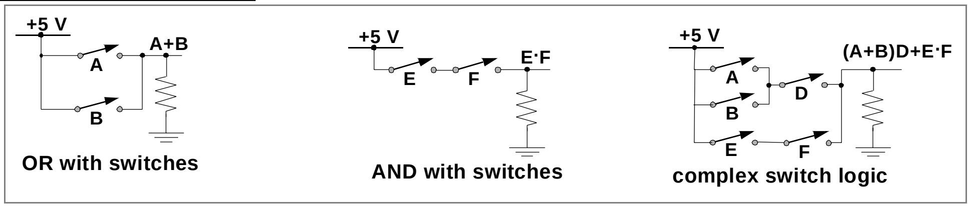
Some Less Common Topics^That All Should Be Covered in This Course

1. Power Supply
   distribution droop, ground bounce, single supply
2. High-frequency lines,
   When is a connection a transmission line, ringing, terminations, open-collector glitch
3. Clock distribution
   clock-skew, direction of data-flow v.s. direction of clock-flow, latching for skew tolerance
   divided clocks, resynchronization, clock gating, clock trees.
4. Production Testing
   BIST (built-in-self-test). scan-chain testing, I_{DDQ} testing, boundary-scan
5. Packaging,
   surface mount, ball-pin grid, heat dissipation, pad protection, ground and power connections.
6. Pads and connections
   Drivers, lead inductance, electrostatic protection, latch up, pad models
7. Reliability,
   Hot electrons, metal migration, Arrhenius model,
8. Group Dynamics
   Fuzzy specifications, communications, software people make different assumptions, documentation.
9. Asynchronous and Synchronous
   What synchronous means, asynchronous inputs, asynchronous reset, metastability.
10. Design style
    Coding style, reusability, using cores, documentation.
11. Verification
    Simulation, synchronous simulation, test benches, static timing analysis, formal verification.

Switching Circuits

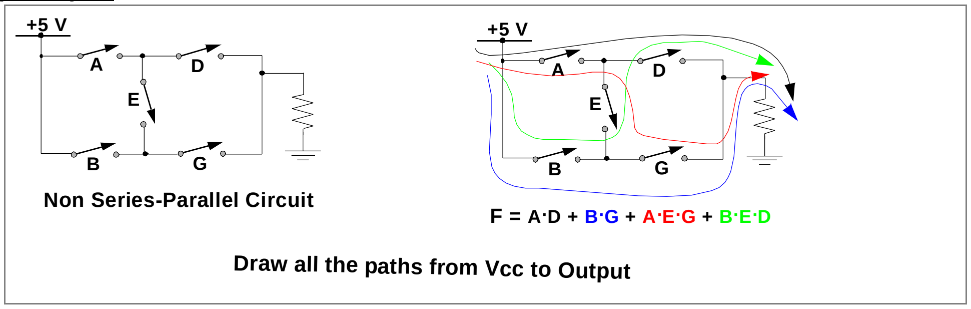
Logic With Switches

Series and Parallel Circuits



General Switch Logic

Loop Analysis


+5 V
A
D
E
B
G
Non Series-Parallel Circuit
+5 V
A
D
E
B
G
F = A·D + B·G + A·E·G + B·E·D
Draw all the paths from Vcc to Output

Switching Circuits

Logic With Switches

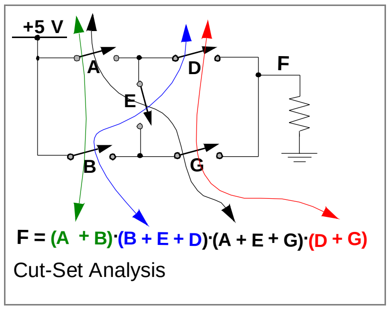
Logic can be done with switches as well as gates.

a. A parallel connection implements OR.
b. A series connection implements AND.
c. Series and parallel combinations can do complex logic.

General Switch Logic

However not all switching circuits can be solved using series and parallel combinations.

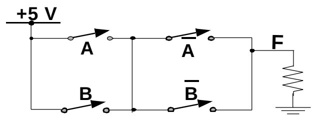
Loop Analysis



Construct all paths between a logic “1” and the output. Each path is a string of ANDs. which are ORed together.
The expression comes out as a *Sum-of-Products* (Σ of Π)

- **Cut-Set Analysis**.
- Make all the cuts that completely separate the output and Vcc.
- The cuts must only pass through switches.
- The switches in the cut are ORed together.
- The expression comes out as a *Product-of-Sums* (Π of Σ)

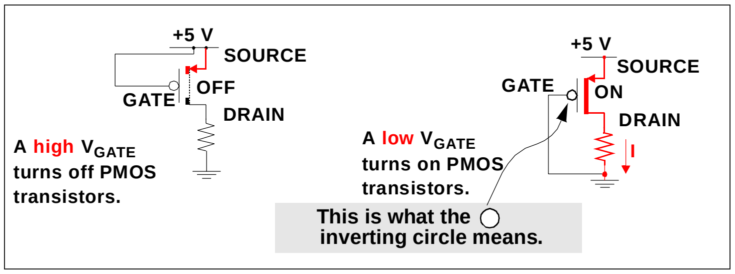
Example: What logic function does this circuit implement?.

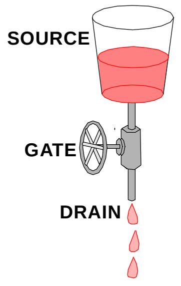


One can do this by loops or cutsets and remove $x\cdot\bar{x}$ algebraically.
Alternately one can note a loop with $x\cdot\bar{x}$ can never be completed, or a cut set with $x + \bar{x}$ is always true.

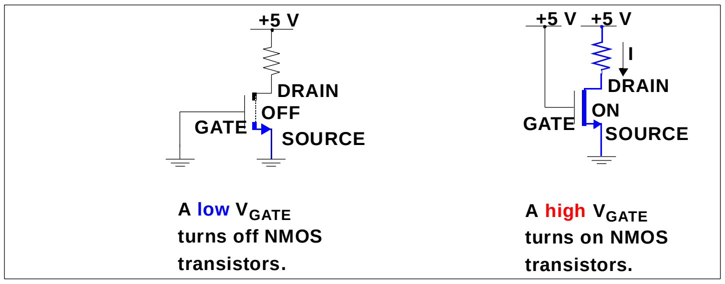
MOS Transistors, (in case you forgot)

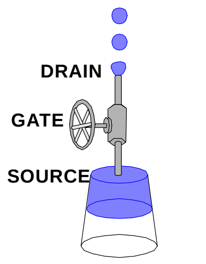
The PMOS transistor


+5 V
SOURCE
OFF
GATE
DRAIN
A high V_{GATE} turns off PMOS transistors.
+5 V
SOURCE
GATE
ON
DRAIN
I
A low V_{GATE} turns on PMOS transistors.
This is what the ◯ inverting circle means.



The NMOS transistor


+5 V
DRAIN
OFF
GATE
SOURCE
+5 V
+5 V
I
DRAIN
ON
GATE
SOURCE
A low V_{GATE} turns off NMOS transistors.
A high V_{GATE} turns on NMOS transistors.



Think “Electrons are lighter than holes.”

MOS transistors

Conventions and Symbols

The p-channel MOS transistor was built first. It was natural to connect it to the high voltage and *drain* into a load. This is the reason for the names *drain* and *source*.
The n-channel MOS transistor kept the same terminology even though it had to drain uphill.

Analog symbols

The MOS transistors used in logic are *enhancement mode*. Enhancement mode means they are shut off for zero gate-source voltage. The analog symbol shows the source-drain connection is off by using a broken line for the channel. (A *depletion mode* transistor, which conducts when $V_{G\text{-}S}=0$, has a solid line).

An NMOS transistor forms a thin N-channel (barely shown) between the two N+ sections on top of the P-type body (P-well). This channel forms just under the thin oxide (not labelled) under the gate.

The body (well) of the transistor and the channel form an NP junction. The arrow on the analog symbol (c) shows the direction conventional current could flow in this junction. However the body is always back biased so this current will not flow.

The analog symbol information is mostly redundant in digital circuits:

- The body of an NMOS transistor is nearly always grounded (PMOS is connected to V_{DD}).
- The source and drain are physically identical and are not distinguished. Which is which is determined by the circuit.

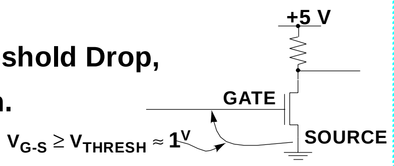
Thus the symbol (a) below is used in digital descriptions and in these notes.

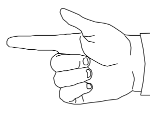
Threshold Voltage in MOS Transistors

In a <u>Conducting</u> MOS Transistor

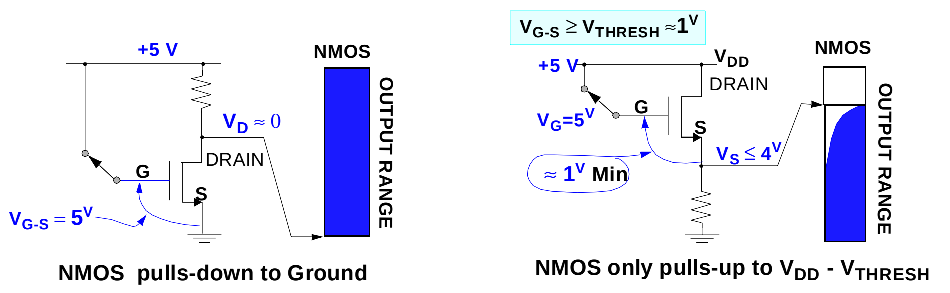
The Gate-Source Voltage ≥ a Threshold Drop,

Much More for Good Conduction.

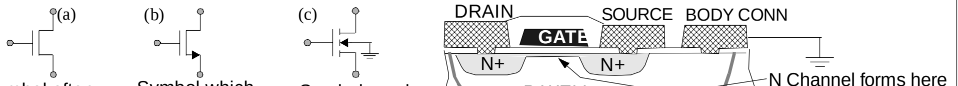




Replacing the switches with transistors



NMOS transistor symbols



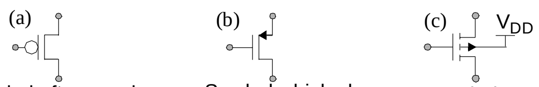
Symbol often used for CMOS circuits

Symbol which shows current direction at the source

Symbol used in many analog circuits

N Channel forms here between N+ and N+ under thin oxide.

PMOS transistor symbols.



Symbol often used used for CMOS circuits

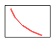
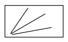
Symbol which shows current direction at the source

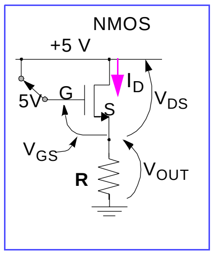
Symbol used in many analog circuits

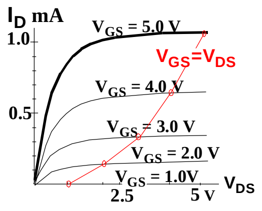
The inverting circle on the left hand PMOS symbol indicates the transistor is turned on by a low gate signal.

Voltage Loss From Using NMOS With Its Feet In the Air

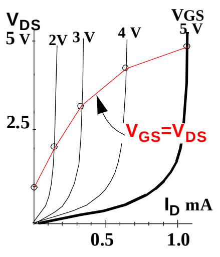
Output Voltage Lowering With Common Drain

- Vout = 5 - V_{DS}
- $V_{GS} = V_{DS}$
- $\Rightarrow V_{OUT} = 5 - V_{GS}$
- (Below Right) V_{OUT} for various I_{DS} 
  The straight lines show Vout $= I_{DS}R$ for a few R. 
  The intersections show V_{OUT} for that value of R.
- One needs over 100K to get V_{OUT} near V_{DD}-$V_{THRESHOLD}$



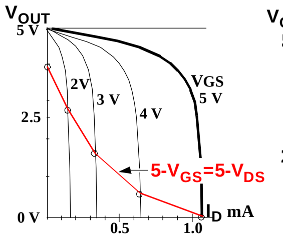


I_D vs V_{DS}



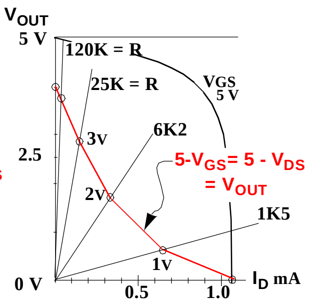
V_{DS} vs I_D

Rotate the axes



$V_{OUT} = (5\text{-}V_{DS})$
$= (5\text{-}V_{GS})$
vs I_D

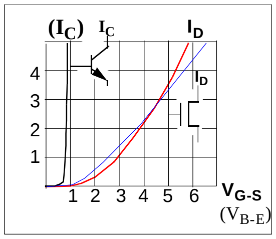
5v - the previous graph



V_{OUT} for different R
Gate at 5V

The Threshold Voltage

The Minimum Gate Voltage to Turn On the Transistor



- Can be adjusted by ion implanting or body biasing.
- For 5 V logic it is about a volt.
- For 3.3V or 2.5V logic it is lowered to about 0.5 V.
- If made too low, transistors will have a high leakage current when OFF.
- $V_{G\text{-}S}$ does not stay constant near the threshold like the the 0.7 $V_{B\text{-}E}$ junction voltage of a Si bipolar transistor.
- $I_{D(MAX)}$ v.s. V_{GS} is quadratic for long channel transistors. ( )
  It is roughly linear for short channel transistors due to velocity saturation.( )

The Model and Spice Parameters For the Threshold Voltage

NMOS $$V_T = V_{T0} + \gamma\left\{\sqrt{|V_{SB}| + 2|\phi_F|} - \sqrt{2|\phi_F|}\right\}$$

V_{TO} = VT0 = Zero-bias threshold voltage. Typically 0.8 V)

γ = GAMMA= Bulk threshold parameter (body coefficient). Typically 0.37 $V^{1/2}$

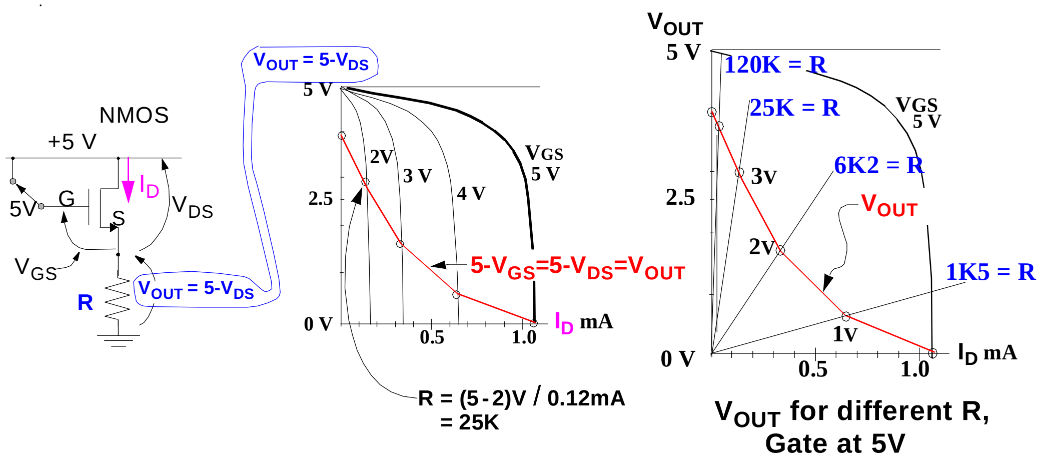
$2\phi_F$=PHI = Surface Potential. Typically 0.65 V

V_{SB} = Source-to-body voltage.

For PMOS transistors used in CMOS (enhancement type) the threshold voltage V_T and the zero-bias threshold voltage V_{T0} are negative:

PMOS $$V_T = -|V_{T0}| - \gamma\left\{\sqrt{|V_{SB}| + 2|\phi_F|} - \sqrt{2|\phi_F|}\right\}$$

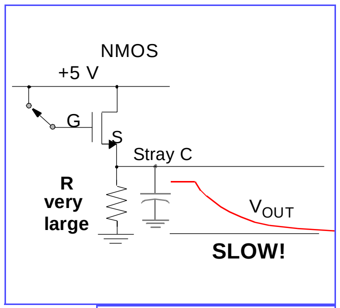
Output Voltage Loss from NMOS With Its Feet In the Air

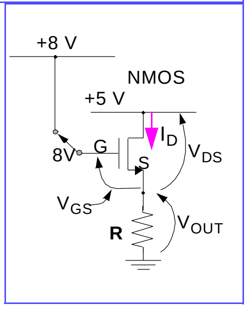


V_{OUT} for different R, Gate at 5V

- Use approximate transistor curves
- Plot where VDS=VGS
- Note: Vout = 5-VDS
- Vout /(Current from curve) = R
- **Vout MAX is 4V, but unless R is very high, it is much less.**

Do not use NMOS up in the air





The curves show.

The voltage will be very low unless the resistance to ground is very high.

If the resistance is very high, the circuit will take a long time to discharge its stray capacitance.

It will be very slow

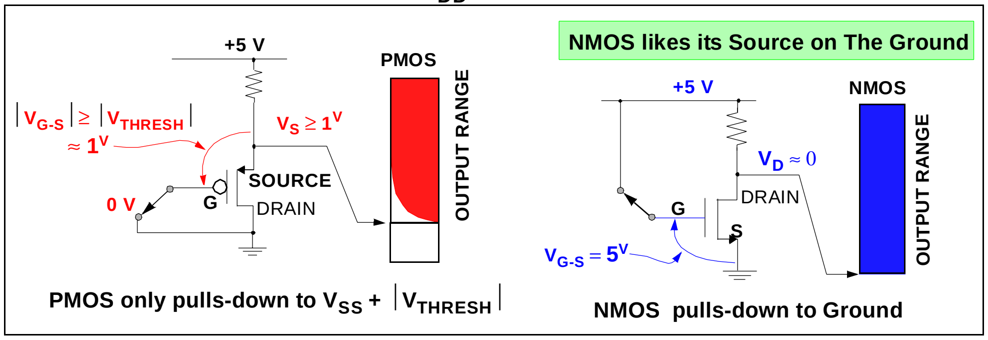
One way to use NMOS up in the air

One can overcome these problems by raising the gate voltage above V_{DD}.

The down side is one must generate another supply.

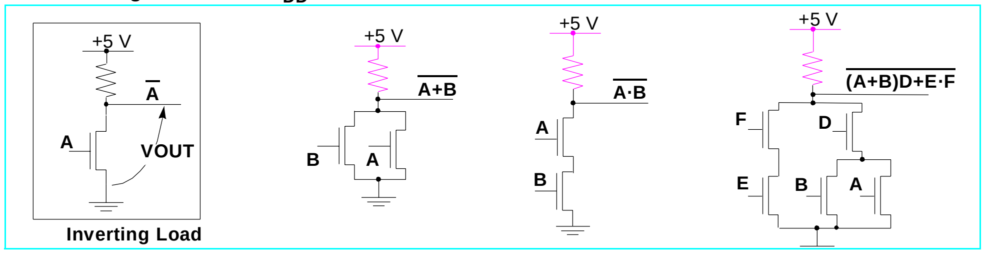
Replacing the switches with transistors

Use NMOS For Loads Connected To V_{DD}.


+5 V
PMOS
$|V_{G\text{-}S}| \geq |V_{THRESH}|$
$\approx 1^V$
$V_S \geq 1^V$
0 V
G
SOURCE
DRAIN
OUTPUT RANGE
PMOS only pulls-down to V_{SS} + $|V_{THRESH}|$
NMOS likes its Source on The Ground
+5 V
NMOS
$V_D \approx 0$
DRAIN
G
S
$V_{G\text{-}S} = 5^V$
OUTPUT RANGE
NMOS pulls-down to Ground

NMOS gives the full logic swing

But connecting the load to V_{DD} inverts the function.

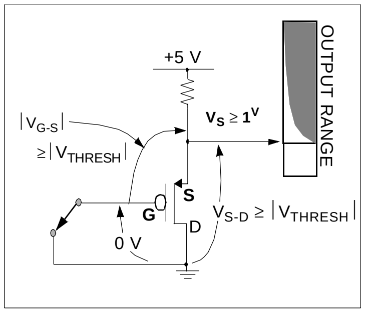


PMOS transistors don't like their feet on the ground.

The minimum V_{GATE} is 0 V.
The SOURCE must be at least $V_{THRESHOLD}$ above the gate for the transistor to be on.

Using Kirchoff,
the output cannot go below $V_{THRESHOLD}$.

The PMOS threshold is written $|V_{THRESH}|$ with absolute value signs. This is because it is negative and writing ≥ with negative quantities is confusing.



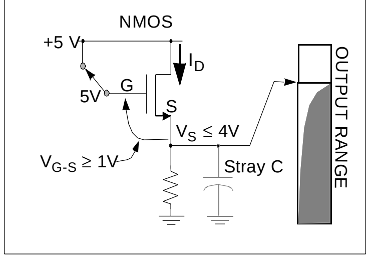
NMOS does not like its feet off the ground

In logic, the gate voltage is limited to 5V.
With a 1V threshold drop, the
output voltage = 5V - 1V = 4V maximum.

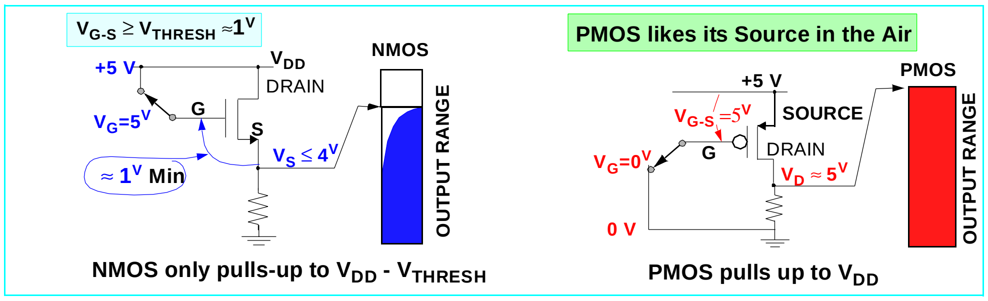
With only a 1V threshold drop, the current I_D will be very low, and can only charge the stray capacitance slowly.
Thus even if one tolerates the 4 V maximum output, gates built this way will be very slow.

The shaded curve is supposed to look like a slow rise time.

If transistor is on, $V_{G\text{-}S} \geq V_{THRESH}$ 1V

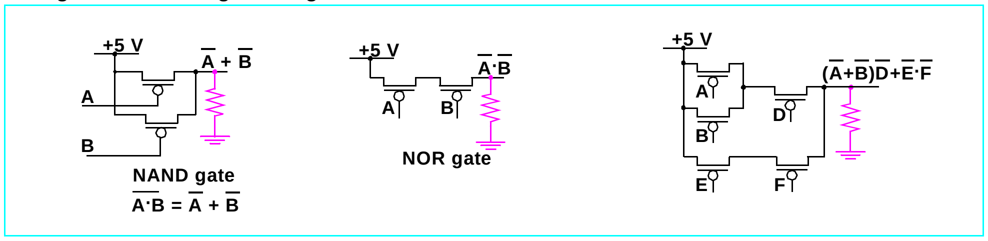


Replacing The Switches With Transistors (cont.)



Use PMOS For Loads Connected To Ground.

PMOS gives the full logic swing.

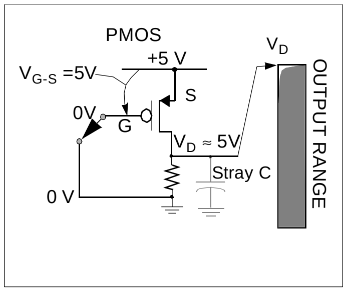

+5 V
$\overline{A} + \overline{B}$
A
B
NAND gate
$\overline{A \cdot B} = \overline{A} + \overline{B}$
+5 V
$\overline{A} \cdot \overline{B}$
A
B
NOR gate
+5 V
$(\overline{A}+\overline{B})\overline{D}+\overline{E}\cdot\overline{F}$
A
B
D
E
F

Using PMOS inverts the inputs and “DeMorgan’s” the function.

PMOS loves dangling its feet

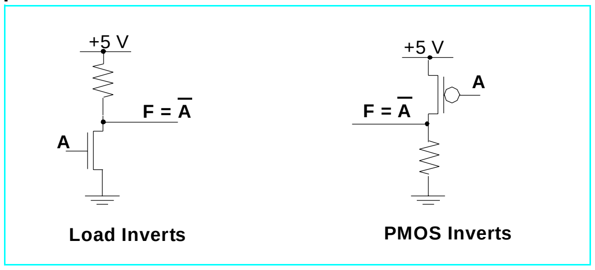
A low gate voltage turns on PMOS.

Here V_{GS} = 5V.
The transistor conducts very well and can supply a full 5V at the drain.
The large current can charge the stray capacitance quickly.

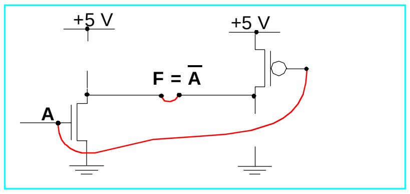


Combining NMOS and PMOS.

- NMOS pulls down for A=1 to make F=0.
- PMOS pulls up when A=0 to make F=1..



- Combine the circuits and throw out the resistors.



Active Pull-up and Pull-Down

Circuits with Resistive Pull-Ups (Pull-Downs)

Circuits with a transistor too pull the output down and a resistor to pull it up are:

- Are called ratioed logic because there low output voltage is the ratio:- $V_{DD}(R_{CHANNEL}/R_{RESISTOR})$
- They pull-low quickly, but pull-high with time constant $R_{RESISTOR}C$.
- They consume power whenever the transistor is on.

Circuits with Transistors to Pull-Up and to Pull-Down

- These circuit can give a low output of zero and a high output of $V_{DD.}$
- They can switch quickly in both directions.
- They do not consume power except during the switching time.

CMOS

The CMOS Configuration

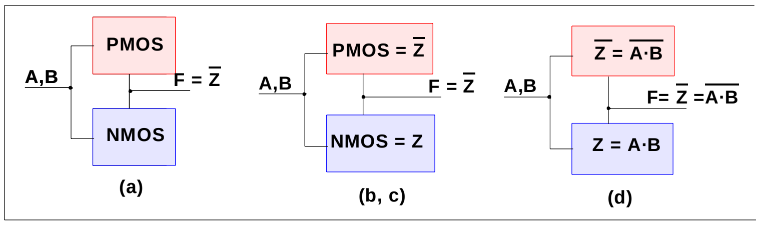
a. A CMOS gate is inverting. Call its output $\overline{Z}$ to indicate this.

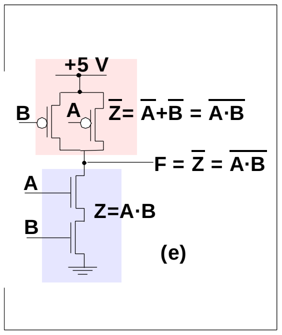
b. The switch function in the upper box and the output are the same. Thus the PMOS function will also be $\overline{Z}$.

c. The switch function in the lower box is the inverse of the output. The NMOS function will be Z.

d. An example of the NAND function.

e. The NAND function showing the transistor switches.



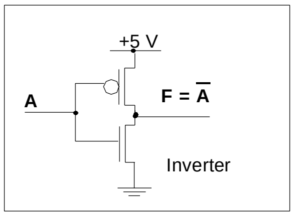

+5 V
B
A
$\overline{Z}$= $\overline{A}$+$\overline{B}$ = $\overline{A·B}$
F = $\overline{Z}$ = $\overline{A·B}$
A
B
Z=A·B
(e)

CMOS

Combining NMOS and PMOS

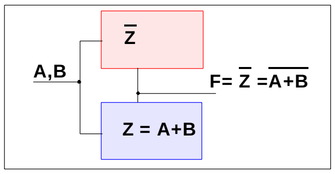
Complimentary Logic

This type always has *one-and-only-one* transistor turned on.
One transistor always gives a path to a supply (VDD or ground).
It never gives a path through both transistors at the same time.



1.• PROBLEM

Draw the circuit for a CMOS NOR gate.



Demorgan’s Theorem

Demorgan’s Theorems, Simple Forms

$\overline{\bar{A} + \bar{B}} = A\cdot B$ $\overline{A\cdot B} = \bar{A} + \bar{B}$ $\overline{D + E} = \bar{D}\cdot\bar{E}$ $\overline{\bar{D}\cdot \bar{E}} = D + E$

Demorgan’s Theorems, General Form

$\overline{F(A,B,C, \ldots +, \cdot,)} = F(\bar{A},\bar{B},\bar{C}, \ldots ,\cdot, +,)$

$F = [\bar{A}\cdot B\cdot C + D\cdot(A\cdot B + C)]\cdot\bar{A}$

a) Bracket all groups of ANDs

$F = \{[\ \{\bar{A}\cdot B\cdot C\ \} + \{D\cdot(\ \{A\cdot B\} + C)\}\ ]\cdot\bar{A}\}$

b) Change AND to OR and OR to AND
Clean brackets

$\{[\{\bar{A} + B + C\}\cdot\{D + (\{A + B\}\cdot C)\}] + \bar{A}\}$

$\{\bar{A} + B + C\}\cdot\{D + \{A + B\}\cdot C\} + \bar{A}$

c) Invert all variables

$\bar{F} = \{A + \bar{B} + \bar{C}\}\cdot\{\bar{D} + \{\bar{A} + \bar{B}\}\cdot\bar{C}\} + A$

Examples

$F = \bar{A}\cdot B\cdot C \Rightarrow \{\bar{A}\cdot B\cdot C\} \Rightarrow \bar{F} = \{A+\bar{B}+\bar{C}\}$

$F = \bar{A}\cdot B\cdot C + A\cdot\bar{B} \Rightarrow \{\bar{A}\cdot B\cdot C\} + \{A\cdot\bar{B}\ \} \Rightarrow \bar{F} = \{A+\bar{B}+\bar{C}\}\cdot\{\bar{A}+B\ \}$

$F = \bar{A}\cdot B\cdot(C + \overline{A\cdot B}) \Rightarrow \{\bar{A}\cdot B\}\cdot(C+ \{\ \overline{A\cdot B}\ \}) \Rightarrow \bar{F} = \{A+\bar{B}\}+(\bar{C}\cdot\{\overline{\bar{A}+\bar{B}}\ \})$

Duality

Duality is a concept closely related to DeMorgan’s Laws.
Duality says that if

$F(A,B,C, \ldots +, \cdot,)$ is a valid formula, then

$F(A,B,C, \ldots ,\cdot, +,)$ is also a valid formula.

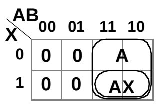
Example

Suppose you remember the most useful of the simplification formulas -

$X + AX = X$

Then the dual is also a valid simplification formula -

$X(A + X) = X$



Example

Everybody remember the distributive law -

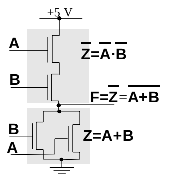
$X(A + B) = AX + BX$

Nobody remembers the other distributive law which is its dual -

$X + (AB) = XA + XB$

- -

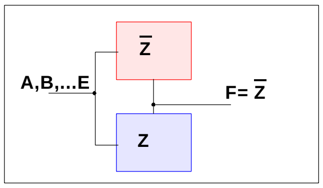
Solution for the CMOS NOR gate.



Using DeMorgan's Law to Design CMOS

In PMOS implement $\overline{Z}$

In NMOS implement Z



Design a CMOS gate for $\overline{Z}$

a. Apply generalized DeMorgan to $\overline{Z}$

$$\overline{Z(a,b...c,\ +,\ \cdot,\ 0,1)} = Z(\overline{a},\overline{b}...\overline{c},\ \cdot,+,\ 1,0)$$

b. Implement the DeMorgan form of $\overline{Z}$ with PMOS switches.

c. Implement the noninverted Z with NMOS switches.

d. Put NMOS and PMOS together.

<u>Call this a $\overline{Z}$/Z gate</u>

<u>Example</u>

a) F = $\overline{Z} = \overline{(A+B)\cdot C+D\cdot E}$

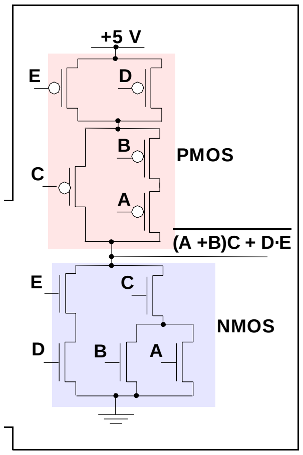
= $\overline{((A+B)\cdot C)+(D\cdot E)}$ Put brackets around AND terms

= $((\overline{A}\cdot\overline{B})+\overline{C})\cdot(\overline{D}+\overline{E})$ Change to DeMorgan form of $\overline{Z}$

b) PMOS = $\overline{Z} = ((\overline{A}\cdot\overline{B}) + \overline{C})\cdot(\overline{D} + \overline{E})$

c) NMOS = Z = (A + B)·C + D·E

d)



Using the Sum of Products (Σ of Π) for the PMOS function

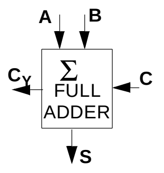
Carry circuit for full adder

$C_Y = A\cdot B + B\cdot C + C\cdot A$

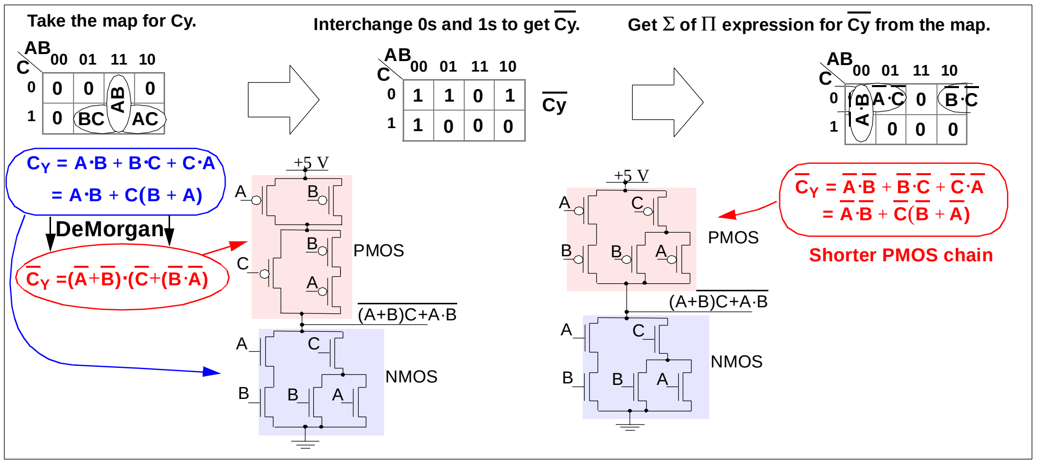
DeMorgan changes F in (Σ of Π) into a $\overline{F}$ in (Π of Σ).

$C_Y = A\cdot B + B\cdot C + C\cdot A$ => $\overline{C_Y} = (\overline{A} + \overline{B})\cdot(\overline{B} + \overline{C})\cdot(\overline{C} + \overline{A})$

Sometimes converting $\overline{F}$ back to (Σ of Π
make it simpler,
or have smaller AND chains,



$S = A \oplus B \oplus C$
$C_Y = A\cdot B + B\cdot C + C\cdot A$


Take the map for Cy.
Interchange 0s and 1s to get $\overline{Cy}$.
Get Σ of Π expression for $\overline{Cy}$ from the map.
AB
C
00 01 11 10
BC
AB
AC
$\overline{Cy}$
$\overline{A}\cdot\overline{B}$
$\overline{A}\cdot\overline{C}$
$\overline{B}\cdot\overline{C}$
$C_Y = A\cdot B + B\cdot C + C\cdot A$
$= A\cdot B + C(B + A)$
DeMorgan
$\overline{C_Y} = (\overline{A}+\overline{B})\cdot(\overline{C}+(\overline{B}\cdot\overline{A})$
+5 V
PMOS
NMOS
$\overline{(A+B)C+A\cdot B}$
$\overline{C_Y} = \overline{A}\cdot\overline{B} + \overline{B}\cdot\overline{C} + \overline{C}\cdot\overline{A}$
$= \overline{A}\cdot\overline{B} + \overline{C}(\overline{B} + \overline{A})$
Shorter PMOS chain

PMOS Circuit

It is usual to change NMOS parallel into PMOS parallel
NMOS series into PMOS parallel.

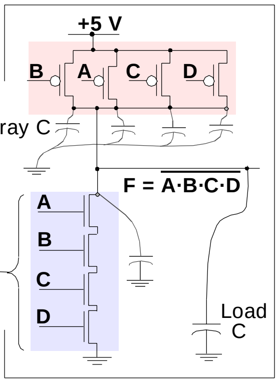
However sometimes rearranging the logic is better.

Circuits in which the PMOS and NMOS connections are the same (as here) are called self-dual.

CMOS Complex Gates

General Properties

- **Can implement any function with a single inversion bar. Only one "bar" in the function, and that over the whole thing.**
  Example $\overline{A\cdot B + C + A\cdot D}$



Max Size on ANDs

Gates are limited to about 4 series transistors.

The total channel resistance of the 4 series transistors is ≈ 4 x (R of a single transistor).

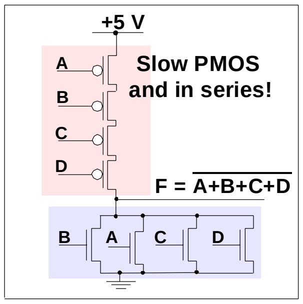
The output time-constant is at least 4 x larger.

If (∑ transistors stray C) >> (C Load)
The output time constant approaches

(4R)(4C) or 16 x(an inverter)

Max Size on NOR.

- **PMOS is 1/2 to 1/3 the speed of NMOS**
- **A 4-input NOR is *Really* Slow.**
  for output going low to high.
- **Output going high to low is faster.**



Gate Speed

Discharging an η input NAND is quadratic in η

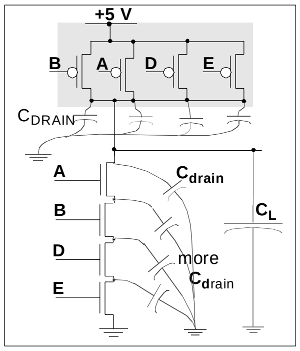
The total NMOS channels resistance = $\eta R_{CHANNEL}$.
The total PMOS drain-to-substrate capacitance = ηC_{DRAIN}.
If C_L is the load capacitance external to the gate. Then the-

Discharge time-constant
$\approx \eta R_{CHANNEL} * (\eta C_{DRAIN}$ + (effects of C_{drain}s)+ C_L)

If $C_L >> C_{DRAIN}$ or C_{drain}, then the discharge time constant is about ηx longer than for a single transistor.

Discharge time constant $\approx \eta R_{CHANNEL} * C_L$

If $C_L << C_{DRAIN}$ then-
Discharge time-constant $\approx \eta R_{CHANNEL} * (\eta C_{DRAIN}$ +(effects of C_{drain}s))



The NMOS drain capacitances, distributed between the Rs, contributes less than the sum of the C_{drains}. An approximation (called the Elmore delay[1]) gives-
$\eta R_{CHANNEL}$ (effects of C_{drain}s)
$\approx R_{CHANNEL} * C_{drain} * (\eta+1)\eta/2)$

For η=4 this gives a discharge time-constant = 14RC = 14x an inverter.

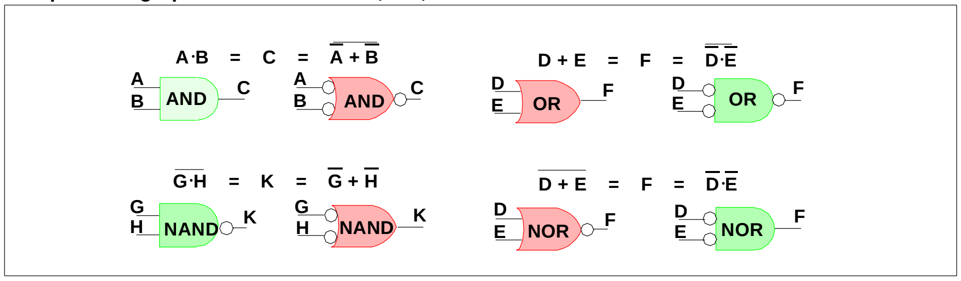
- $C_{drain\text{-}to\text{-}gate}$ is lumped in with C_{drain} (drain-to-substrate) here.
- Because η series transistors will act like a long transistor with less velocity saturation. $R_{CHANNEL}$ will increase less than linearly with η.

Though not quite 16x, the rapid time constant increase discourages large η.

1. [RABAEY96] Example 4.5 p. 476.

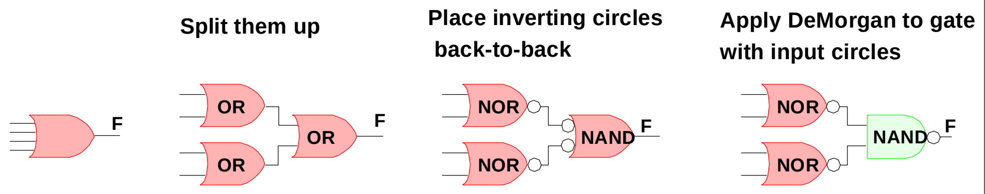
Using DeMorgan Graphically (Review)

Equivalent graphical forms for AND, OR, NAND and NOR.


$A \cdot B = C = \overline{\overline{A} + \overline{B}}$
A
B
AND
C
$D + E = F = \overline{\overline{D} \cdot \overline{E}}$
D
E
OR
F
$\overline{G \cdot H} = K = \overline{G} + \overline{H}$
G
H
NAND
K
$\overline{D + E} = F = \overline{D} \cdot \overline{E}$
NOR

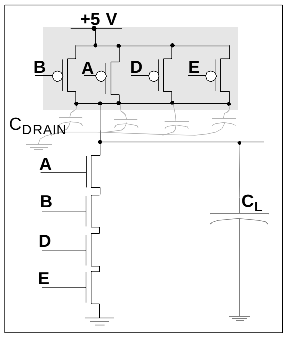
Getting Rid of Big ORs.

Using DeMorgan graphically to partition big ORs into faster structures.


Split them up
Place inverting circles back-to-back
Apply DeMorgan to gate with input circles
F
OR
NOR
NAND

Charging C_L through parallel transistors

With parallel transistors only one might be on, so the worst case charging resistance is as for one. Parallel transistors can share the same substrate well so additional transistors add less to the total C_{DRAIN} than the first[1].



NANDs beat NORs

Since electrons are more mobile than holes, PMOS transistors are 1/2 to 1/3 of the speed of NMOS. NORs have a PMOS string which makes them slow.

To compensate cell designers make the PMOS transistors double width or more. Check how well the compensation is done by comparing the cell high-to-low propagation delay (t_{PHL}) and the low-to-high propagation delay (t_{PLH}). Two-to-one differences are common for the 2-input NOR.

Large Fan-In Gates

CMOS gate propagation delay changes according to the following approximate formula.

η= fan-in; the number of input leads coming into the gate.[2]

$$t_{PROP} = a_1\eta + a_2\eta^2$$

The $a_1\eta$ is important with large $C_L >> C_{DRAIN}$.

The $a_2\eta^2$ is important when $C_L << C_{DRAIN}$,

2.• PROBLEM

Break a 6-input NAND into 2-input gates (one 3-input gate is allowed).

1. [Rabaey96] p.201 shows how overlap capacitance is shared. Also the Miller effect doubles C but only for the first transistor to switch.

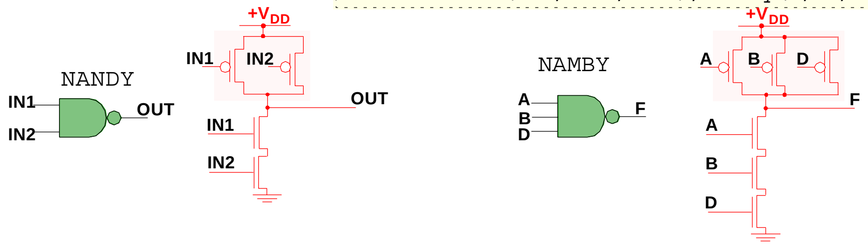
2. [Rabaey96] p.196 discusses this quadratic relationship.

Basic CMOS Gates

NAND

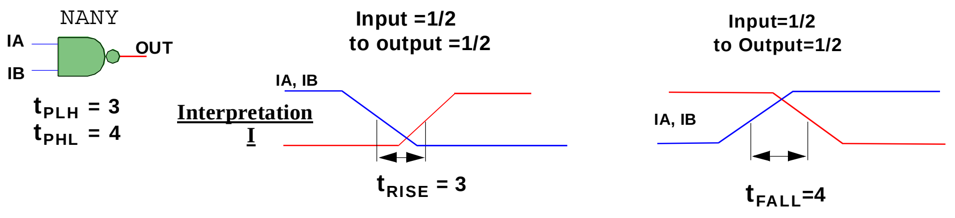
Verilog textual description

```
nand NANDY(OUT, IN1, IN2), namby(F, A, B, D);
```



Nand with propagation delays

```
nand #(3 4) NANY(OUT, IA, IB);
```



Verilog Gate Description

The Interpretation of Delay Times Depends on the Number of Arguments

NAND #(All_transition_delay) instance_name(output, ina, inb, ...)

NAND #(Rise_time, Fall_time) instance_name(output, ina, inb, ...)

NAND #(Rise_time, Fall_time, Turn_off_delay) instance_name(output, ina, inb, ...)

- Rise_time. The output starts in X, Z or 0 and ends in 1.
  Interpretation I. The delay from the 1/2 value of an input change until the output reaches 1/2.
  Interpretation II. (older) The delay from the start of an input change until the corresponding output= 1.
  The cell designers tend to use interpretation I. Circuit designers must follow cell designers.
- Fall_time. The output starts in X, Z or 1 and ends in 0.
  Interpretation I. (More common) The delay from the 1/2 value of an input change until the output reaches 1/2.
  Interpretation II. (older) The delay from the start of an input change until the corresponding output= 0.
- Turn_off_delay. The output starts in X, 0 or 1. Turn_off_delay is the time from an input change until the output= Z. The built-in NANDs do not have 3-state control but `bufif0`, and `bufif1` gates do. See page 44.
- All_transition_delay. One number is used for all of the above delays.

Time Units

The delays are in simulator units. Thus 3 could be 3ns, 3µs or 3 years, whichever you say it is. However the timescale command can specify units. This is useful for combining a modules specified in say ps with one using ns.

```
`timescale  100 ps / 10 ps    // Reference time unit is 100ps with accuracy of 10 ps.
                              // Only 1, 10, and 100 are allowed as numbers.
```

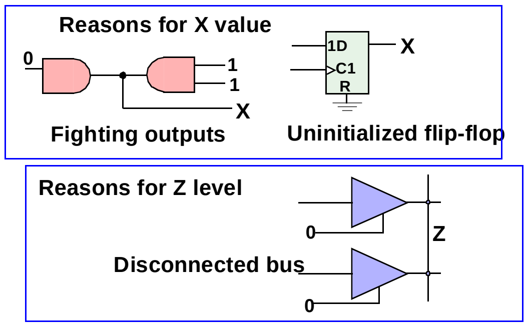
Case Sensitivity

- Verilog is case sensitive
- Reserved words are in lower case.
- User variables can be either case but `NANDY` is not `Nandy`, which is not `nandy`.

NAND Gate Definition Using 4 Logic Values

Four-Value Logic

- 1
- 0
- X = unknown to simulator
- Z = disconnected, tri-state®



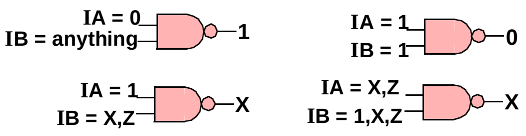
Four-Value NAND

IB

NAND	0	1	X	Z
0	1	1	1	1
1	1	0	X	X
X	1	X	X	X
Z	1	X	X	X

IA

Meaning of NANDing with X and Z



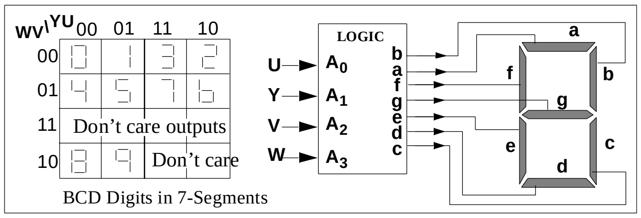
- A simulated input Z gives the output as input X for NANDs, NORs, ANDs and ORs.
- X is unknown in the simulator. Do not confuse with “don’t care” in logic design.

Don’t Care Outputs are not Simulation Unknowns

BCD to 7-segment display

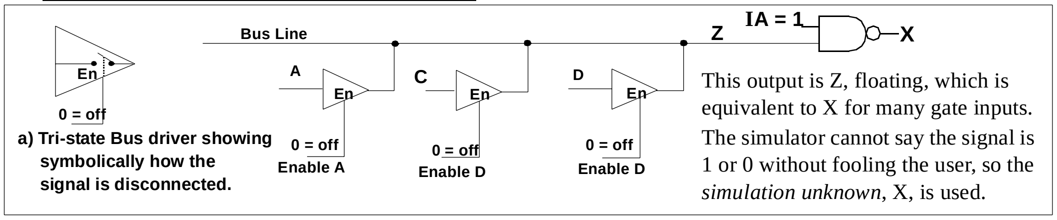
Inputs WVYU are always decimal numbers, so inputs 1100, 1101, 1111, 1110, 1011 and 1010, never happen.

The output for those inputs are never displayed and are *don't care outputs*. These can be used to simplify the logic.



BCD Digits in 7-Segments

Simulation Unknown From a Turned Off Bus



a) Tri-state Bus driver showing symbolically how the signal is disconnected.

This output is Z, floating, which is equivalent to X for many gate inputs. The simulator cannot say the signal is 1 or 0 without fooling the user, so the *simulation unknown*, X, is used.

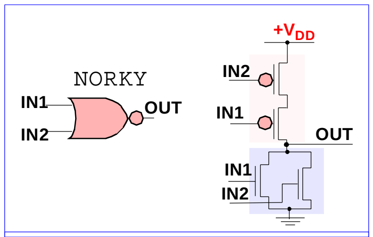
- For CMOS, the value on a disconnected bus may be the previous value stored in the bus capacitance. Verilog has a model `trireg` which can be used to simulate this.
- Once started, X can propagate through a circuit like cholera.
- Another unknown type is the *don’t care input*. It means the output is independent of that input. Verilog uses a “?” for this in functions defined by a table.
  For example the table for an OR might be written:
  `table  0 0 ; 0;   1 ? : 1;   ? 1 : 1;  endtable`
  The “?” stands for any of 1, 0, X, or Z.
- “?” has a different meaning in a `casex` or `casez` statements. There it is equivalent to Z.

Basic CMOS Gates (continued)

NOR

Verilog textual description

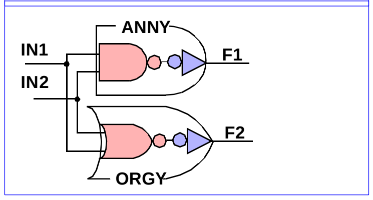
```
nor #(RISE_T, FALL_T) NORKY(OUT, IN1, IN2);
```



AND and OR

These are implemented as NAND and NOR with an extra inverter

```
and #(RISE, FALL) ANNY(F1, IA, IB);
or  #(RISE, FALL) ORGY(F2, IA, IB);
```



XOR and XNOR

Two very useful gates

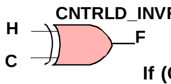
<u>They act as controlled inverters.</u>

F=B if C=0; $F = \overline{B}$ if C=1.

<u>They compare two inputs</u>

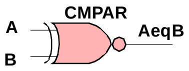
A⊕B is True if A≠B

$\overline{A \oplus B}$ is True if A=B



If (C= =0) then F = B, else $F = \overline{B}$;

```
assign F= (C==0) ? B : B̄;
```



AeqB is true if A equals B;

```
assign AeqB = (A==B);
```

Basic CMOS Gates (continued)

<u>Built-in Gates</u>

There are 6 built-in primitive gates: nand, nor, and, or, xor, xnor.

<u>The Number of Inputs</u>

Verilog builds the gate to agree with the number of inputs in the expression.

<u>Optional Arguments</u>

The rise_time and fall_time arguments are optional. The default is zero delay.

Behavioural Statements

This is behavioural Verilog, part of the Register Transfer Level, which is the level of abstraction at which many circuits are synthesized.
See "The Path for a Computer Aided Design (CAD) Tool" on page 5.

<u>Conditional Statements</u>

The general form is

<condition> **?** *<expression_if_condition is true>* **:** *<expression_if_condition is false>*

<u>Relational Statements</u>

A= = B   gives output 1 if A=0=B, or A=1=B. If either input contains an X or a Z the output is X.

A = = = B also gives output 1 if A=X=B, or A=Z=B. It never gives an X output.

3.• PROBLEM

Construct the truth table for an XOR gate. The inputs may be 0, 1, X or Z. There is no such thing as level $\overline{X}$.

The solution should contain 12 X entries.

XOR Construction

XOR Gate Construction

XOR is Not a $\bar Z$/Z Gate

XOR needs internal inversions.

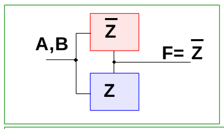
Build an XOR from:

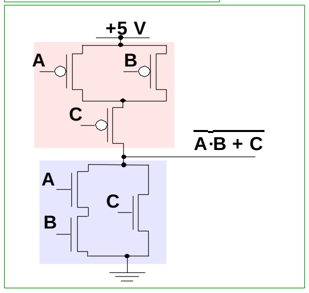
C= $\overline{A+B}$ and F= $\overline{A\cdot B+C}$
see side block

Construct a $\bar Z$/Z gate for F

$\bar Z$ = $\overline{A\cdot B+C}$ = $(\bar A + \bar B)\bar C$

Z = A·B+C





Equation and map of XOR

A⊕B = $\bar A\cdot B + A\cdot\bar B$

A \ B	00	01
0	0	1
1	1	0

map of $\overline{A\oplus B}$

A \ B	00	01
0	1	0
1	0	1

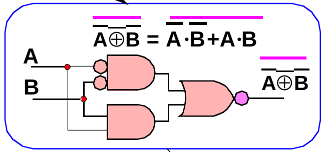
equation of $\overline{A\oplus B}$

A \ B	00	01
0	$\bar A\cdot\bar B$	0
1	0	AB

$\overline{A\oplus B} = \bar A\cdot\bar B + A\cdot B$

A⊕B = $\overline{\overline{A\oplus B}}$

Put on extra ● to make XNOR into XOR.

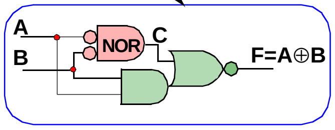

$\overline{\overline{A\oplus B}} = \overline{\bar A\cdot\bar B + A\cdot B}$
A
B
$\overline{\overline{A\oplus B}}$

Convert to NOR gate

C= $\overline{A+B}$

and complex CMOS gate

F= $\overline{\bar A\cdot\bar B+C}$



XOR Constructed With Two $\bar Z$/Z Gates

An alternate way to get the expression for XOR is to start with the product-of-sums representation for XOR:

F= $(\bar A + \bar B)(A + B)$

Apply DeMorgan

$\bar F$= $(A\cdot B) + (\bar A\cdot\bar B)$

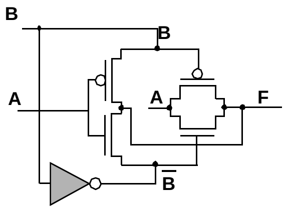
Apply DeMorgan again

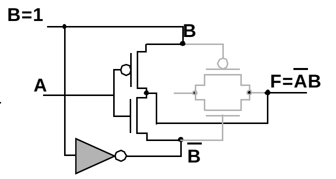
$\bar{\bar F} = F = \overline{(A\cdot B) + (\bar A\cdot\bar B)}$

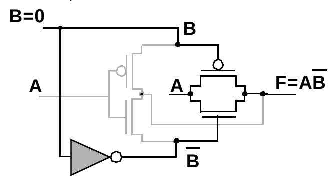
XOR Built With Transmission Gates

This 6-transistor XOR implementation is used in some cell libraries.

It uses a transmission gate, a PMOS and an NMOS transistor in parallel which are both turned on and off together. A is connected to F when B is 1 and disconnected when B=0.







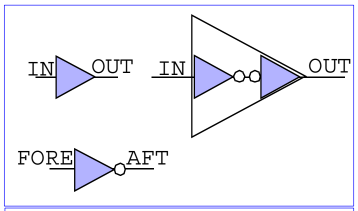
- When B=1, the transmission gate is off, and B=1, $\bar B$=0 powers the inverter.
  The output is $\bar A$, but since B=1, one can say the output is $\bar A$B
- When B=0, the inverter is off, but the transmission gate is on.
  The output is A but since B=0, it is actually A$\bar B$.

CMOS Gates (continued)

Buffer and Not Gates Construction

Regular Buffer and NOT

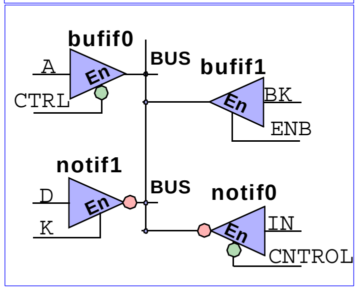
```
buf #(RISE, FALL) DRV(OUT, IN);
not #(RISE, FALL) DRV(AFT, FORE);
```



TRI-STATE® Buffer and NOT.

```
bufif1 #(Tlh, Thl, Tz) DRV1(BUS, BK, ENB);
bufif0 #(Tlh, Thl, Tz) DRV0(BUS, A, CTRL);

notif1 #(Tlh, Thl, Tz) Not1(BUS, D, K);
notif0 #(Tlh, Thl, Tz) Not0(BUS, IN, CNTROL);
```



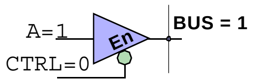
`bufif0` **means**
normal buffer if control=0,
tristate if control=1.

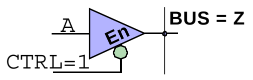
Four-Value Truth Tables

CTRL

A	bufif0	0	1	X	Z
	0	0	Z	Z	Z
	1	1	Z	Z	Z
	X	X	Z	X	X
	Z	X	Z	X	X

bufif0





Legal Notice

TRI-STATE is the registered trade mark of National Semiconductor Corp. Their lawyers have notified at least one text book writer[1] to display this fact. It is not clear if this applies only to the spelling "tristate." The word "3-state" appears to be in the public domain.

Gate Timing

Time to go 3-State

There are three times that can be associated with `bufif` and `notif` type gates.

- The rise time or t_{pLH} (time propagation high to low), the first argument
- The fall time or $t_{pHL,}$ the 2nd argument.
- The time to go into (and come out) of 3-state, or $t_{Z.}$

In user-defined gates one can use a construction called a *specify block* to make more complex timing. For example one could have a different t_Z for turn-on and turn-off. This is not available in the built-in gates.

Min:Typical:Max Delays

Verilog simulators such as Cadance's Verilog-XL simulator can run either slow, typical or fast gates. This is specified by a 3 numbers arranged min:typ:max

```
buf #(35:25:15, 31:25:19)  DRV(OUT, IN);
```

When the simulator is invoked, which delays will be used is chosen as an option.

In the Cadence verilog-XL simulator a command-line options is used when starting the simulator. For example to run with t_{RISE}=35 and t_{FALL}=31 for this gate, use

```
> verilog your_file_name.v  +maxdelays
```

The other command-lines options are `+mindelays` and `+typdelays`.

1. John Wakerly in *Digital Design, Principles and Practices*, second edition, p.122.

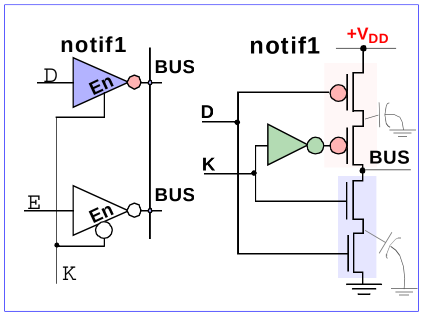
Buffer Construction

Simple Buffer

K=0 isolates BUS from D.
Leaves bus floating as far as D is concerned.

Connecting control K to the middle transistors makes the bus slightly faster.

It disconnects the internal stray capacity shown from the bus, when driver is off (K=0).

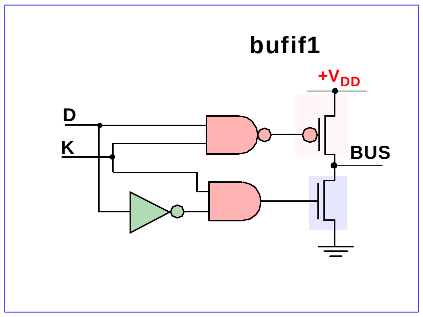


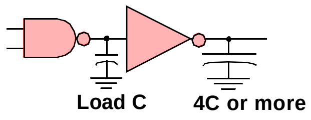
Faster Buffer

This has only one channel resistance in the bus charge/discharge path.

If
- the bus capacitance is large,
- D and K are small transistors,

the NAND and NOR can be made stepped drivers for minimum delay.





Stepped Drivers

These consider the fact that when buffer transistors are made wider to charge their output load faster, their gate width increases and puts more load on the buffer's driver stage.

Rabaey[1] shows that the optimum increase in transistor widths is about 3.6, and that the buffer load must be over 4 times the buffer driver load before one can get any improvement.

1. [RABAEY96], pp. 448-454