

---

# Digital Design Principles – Introduction

---

ECE 152A – Winter 2012

---

## Reading Assignment

- Brown and Vranesic
  - 1 Design Concepts
    - 1.1 Digital Hardware
      - 1.1.1 Standard Chips
      - 1.1.2 Programmable Logic Devices
      - 1.1.3 Custom-Designed Chips
    - 2 Introduction to Logic Circuits
      - 2.1 Variables and Functions
      - 2.2 Inversion

## Reading Assignment

- **Brown and Vranesic** (cont)
  - 2 Introduction to Logic Circuits (cont)
    - 2.3 Truth Tables
    - 2.4 Logic Gates and Networks
      - 2.4.1 Analysis of a Logic Network
  - 3 Implementation Technology
    - 3.1 Transistor Switches
    - 3.3 CMOS Logic Gates

## Reading Assignment

- **Roth**
  - 1 Introduction
    - 1.1 Digital Systems and Switching Circuits
  - 2 Boolean Algebra
    - 2.1 Introduction
    - 2.2 Basic Operations

## Course Overview

- ECE 15A (CS 30)
  - Static, combinational logic
    - No clock and no concept of “timing”
      - Necessary building blocks for ECE 152A
- ECE 152A
  - Dynamic, sequential circuit design
    - Clocked, synchronous circuits
    - Electrical and physical aspects of digital system design
    - Emphasis on design

## Course Overview

- ECE 152A (cont)
  - Design methodologies
    - Computer Aided Design (CAD) tools
      - Simulators, Schematic Capture, etc.
    - Hardware Description Languages
      - Synthesis and Programmable Logic Devices
  - Digital Logic Circuits
    - TTL (Transistor – Transistor Logic)
    - CMOS (Complementary Metal Oxide Semiconductor)

# Design of Digital Systems

- Four Levels of Abstraction
  - The Architecture (or System) Level
    - ECE 152B, ECE 154
  - The Logic Design Level
    - ECE 152A
  - The (Digital) Circuit Design Level
    - ECE 124A, ECE 125
  - The Physical Design Level
    - ECE 124B, ECE 124C, ECE 124D

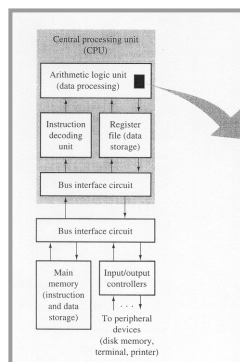
January 9, 2012

ECE 152A - Digital Design Principles

7

# The Architecture Level

- Design with high-level blocks (subsystems)



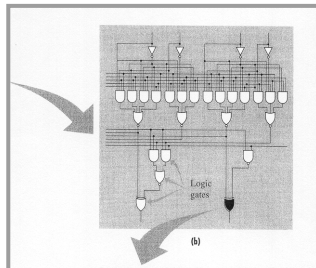
January 9, 2012

ECE 152A - Digital Design Principles

8

## The Logic Design Level

- How do you build the high-level blocks from logic gates?



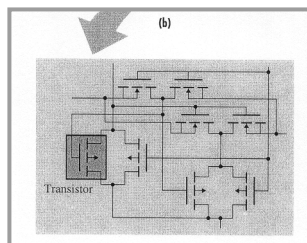
January 9, 2012

ECE 152A - Digital Design Principles

9

## The Circuit Design Level

- How do you build the logic gates from transistors?



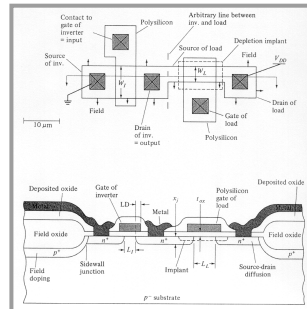
January 9, 2012

ECE 152A - Digital Design Principles

10

## The Physical Design Level

- How do you build and connect the transistors in an integrated circuit (IC) process?



January 9, 2012

ECE 152A - Digital Design Principles

11

## Design Methodologies

- Schematic Based Design
  - Design circuit using manual and/or CAD based design techniques
    - Karnaugh maps, state diagrams, timing diagrams, etc.
  - Select components
    - Gates, flip-flops, registers, Megacells, etc.
  - Define interconnection
    - Schematic (which generates a netlist)
  - Simulate and build prototype (discrete or IC)

January 9, 2012

ECE 152A - Digital Design Principles

12

## Design Methodologies

### ■ Code Based Design

- Write code describing and defining the register transfer operation of device
  - Looks like programming language
- CAD Tool “synthesizes” design
  - Generates logical model of design
  - Selects components from a library and defines interconnection
  - Maps design onto target architecture (programmable logic device, FPGA, ASIC, etc.)

## Design Methodologies

### ■ Languages for Code Based Design



- Verilog
  - Commercially developed language and simulator
- VHDL
  - VHSIC (Very High Speed Integrated Circuit) Hardware Description Language
    - Department of Defense
- Other languages
  - ABEL and (other) vendor proprietary languages

## Design Methodologies

- We will do both schematic and code based designs in ECE 152A
  - Learn the manual design techniques
    - Utilized in schematic based designs
    - Incorporated into CAD tools
- Verilog will be the hardware description language used in ECE 152A

## Logic Gates and Functions

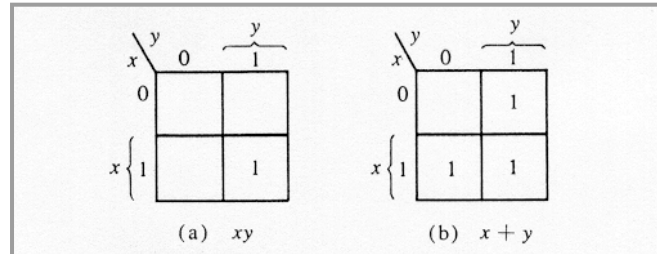
- AND and OR
  - Symbols, Functions and Truth Tables

AND		$F = xy$	<table border="1"><thead><tr><th>x</th><th>y</th><th>F</th></tr></thead><tbody><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></tbody></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 border="1"><thead><tr><th>x</th><th>y</th><th>F</th></tr></thead><tbody><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></tbody></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																



# Karnaugh Maps

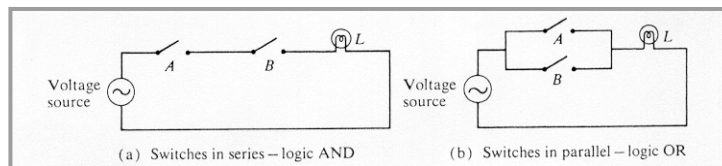
- Graphical Representation of AND and OR Functions



# Switches

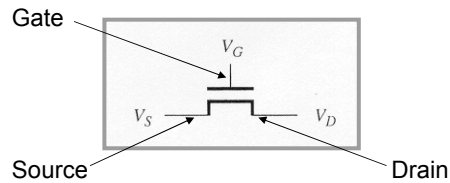
- Switch Abstraction of AND and OR Functions

- Switch is Open when  $A$  or  $B = 0$ , Closed when  $A$  or  $B = 1$
- Light is illuminated when “function” equals logic 1



## Transistor Implementation

- The n-channel, Metal Oxide Semiconductor, Field Effect Transistor (NMOS FET) behaves like the switches in the previous abstraction
  - 3 terminal device
    - Gate controls current flow from Drain to Source



January 9, 2012

ECE 152A - Digital Design Principles

19

## Transistor Implementation

- Functional operation of the NMOS FET
  - Closed (short circuit) when a logic 1 applied to gate of transistor
    - In “positive logic”, logic 1 is the most positive voltage in the circuit
      - 5 volts in the following example
  - Open (open circuit) when a logic 0 applied to gate of transistor
    - In “positive logic”, logic 0 is the least positive voltage
      - Zero volts, or ground in the following example

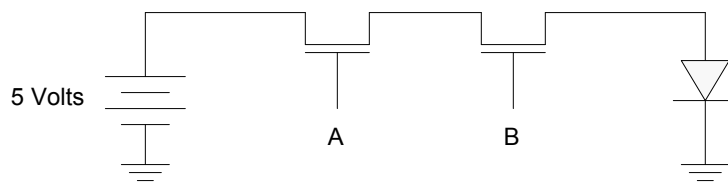
January 9, 2012

ECE 152A - Digital Design Principles

20

# Transistor Implementation

## ■ Series (AND) Circuit Example



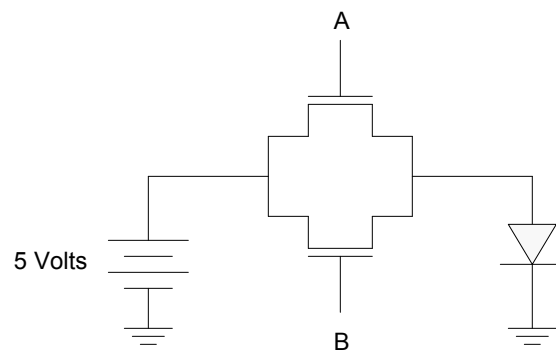
January 9, 2012

ECE 152A - Digital Design Principles

21

# Transistor Implementation

## ■ Parallel (OR) Circuit Example





January 9, 2012

ECE 152A - Digital Design Principles

22

# Logic Gates and Functions

- Inverter (Complement or NOT function) and Buffer
  - Symbols, Functions and Truth Tables

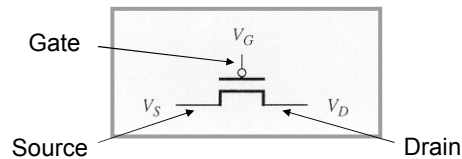
Inverter	$x$ —  — $F$	$F = x'$	<table border="1"><tr><th><math>x</math></th><th><math>F</math></th></tr><tr><td>0</td><td>1</td></tr><tr><td>1</td><td>0</td></tr></table>	$x$	$F$	0	1	1	0
$x$	$F$								
0	1								
1	0								
Buffer	$x$ —  — $F$	$F = x$	<table border="1"><tr><th><math>x</math></th><th><math>F</math></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								

# Inverter Implementation

- The Inverter is the fundamental circuit in all logic families and technologies
  - Most logic families and technologies are inherently inverting
  - All basic logic gates (functions) can be derived from the inverter
    - Logic functions realized by connecting transistors in series or parallel

## Inverter Implementation

- The p-channel Metal Oxide Semiconductor, Field Effect Transistor (PMOS FET)
  - Closed when a logic 0 applied to gate of transistor
    - “Bubble” on gate indicates “active low”
  - Open when a logic 1 applied to gate of transistor



January 9, 2012

ECE 152A - Digital Design Principles

25

## CMOS Inverter Implementation

- The Complementary Metal Oxide Semiconductor (CMOS) Inverter
  - Both NMOS and PMOS devices on a common substrate
  - Inherently “digital” technology
    - Allows for simple logic gate implementation
      - Fewest number of transistors
    - Superior DC electrical characteristics
  - Vast majority of modern digital systems are implemented in CMOS

January 9, 2012

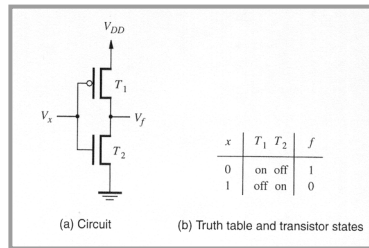
ECE 152A - Digital Design Principles

26

# The CMOS Inverter

- Static characteristics

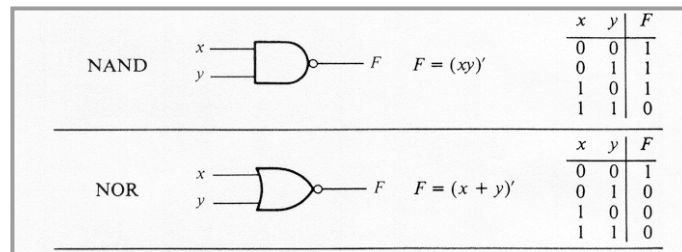
- Since one device is always off, there is no static current flowing (and thus no static power)



# Logic Gates and Functions

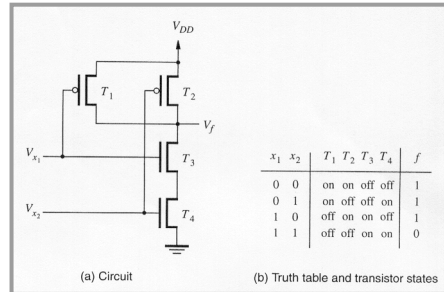
- NAND and NOR

- Symbols, Functions and Truth Tables



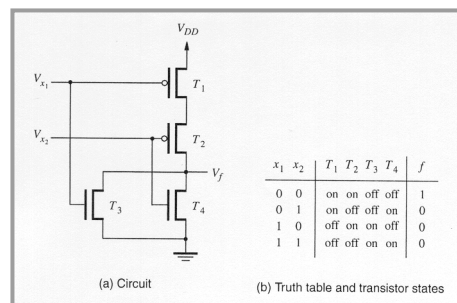
# CMOS Logic Gates

## ■ NAND Gate



# CMOS Logic Gates

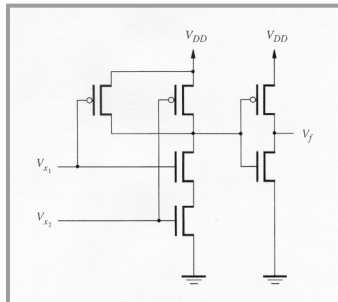
## ■ NOR Gate



# CMOS Logic Gates

- AND Gate

- NAND gate followed by inverter



# Logic Gates and Functions

- Exclusive OR (XOR) and Exclusive NOR (XNOR, or Equivalence)

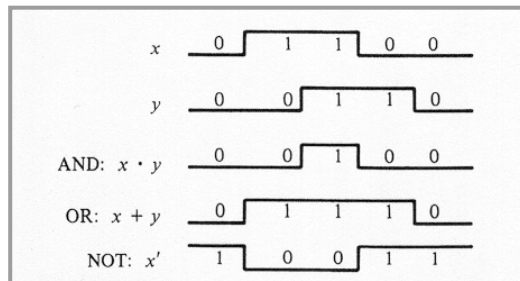
- Symbols, Functions and Truth Tables

Exclusive-OR (XOR)		$F = xy' + x'y$ $= x \oplus y$	$x$	$y$	$F$
			0	0	0
			0	1	1
			1	0	1
			1	1	0
Exclusive-NOR or equivalence		$F = xy + x'y'$ $= x \odot y$	$x$	$y$	$F$
			0	0	1
			0	1	0
			1	0	0
			1	1	1



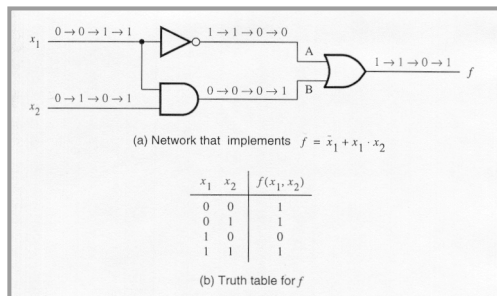
# Timing Diagrams

- Dynamic Representation of AND, OR and NOT Functions
  - Inputs change over time to verify Truth Table



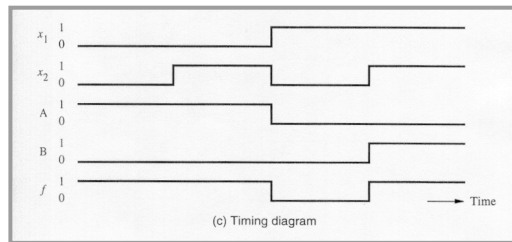
# Logic Networks

- Networks of Logic Gates (Logic Circuits)
  - Network and Truth Table
    - $f = x_1' + (x_1 \cdot x_2)$



# Logic Networks

- Networks of Logic Gates (Logic Circuits, cont)
  - Functional Timing Diagram (zero gate delays)
    - $f = x_1' + (x_1 \cdot x_2)$



# Propagation Delay

- Outputs of logic gates do not change instantaneously with input changes
  - NAND Gate Timing Diagrams

Zero gate delay (functional)

Unit gate delay  
( $t_{pd}$  = propagation delay)

