



# CSE140L: Components and Design Techniques for Digital Systems Lab

## Introduction

Tajana Simunic Rosing

# Welcome to CSE 140L!

- **Course time:** W 2-2:50pm, WLH 2205
- **Discussion session:** F 12-12:50pm, CSE 3219
- **Instructor: Tajana Simunic Rosing**
  - Email: [tajana@ucsd.edu](mailto:tajana@ucsd.edu); please put CSE140L in the subject line
  - Ph. 858 534-4868
  - Office Hours: Tu/Th 1-2pm, CSE 2118
- **Instructor's Assistant: Sheila Manalo**
  - Email: [shmanalo@ucsd.edu](mailto:shmanalo@ucsd.edu)
  - Phone: (858) 534-8873
- **TA: Ling Zhang**
  - Email: [lizhang@cs.ucsd.edu](mailto:lizhang@cs.ucsd.edu)
  - Office hours: M 2-3pm; W 10-11am in CSE 3219
- **TA: Chun Chen Liu**
  - Email: [chl084@ucsd.edu](mailto:chl084@ucsd.edu)
  - Office hours: T 10am-12pm in CSE 3219
- **Class Website:**
  - <http://www.cse.ucsd.edu/classes/sp08/cse140L/>
- **Grades:** <http://webct.ucsd.edu>

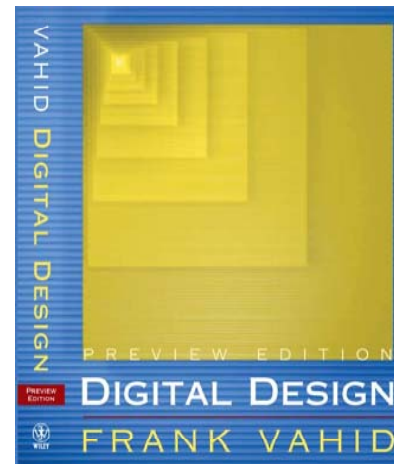
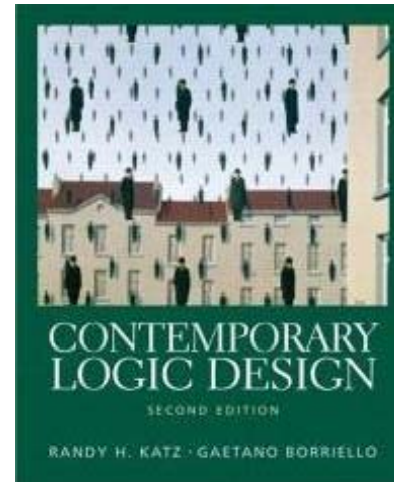
# Course Description



- **Prerequisites:**
  - CSE 20 or Math 15A, and CSE 30.
  - CSE 140 must be taken concurrently
- **Objective:**
  - Introduce digital components and system design concepts through hands-on experience in a lab
- **Grading**
  - Labs (4): 70%
    - We have 15 Xilinx platforms with PCs – organize in teams of two
    - Schedule for lab access; need to schedule a demo to TA by lab due date
    - Go to Robin Knox [rsknox@cs.ucsd.edu] office in CSE 2248 to program your student ID for access to CSE 3219
      - Monday-Thursday 10-12:30 and 2:00-4:00
  - Final exam: 30%
  - Regrade requests: turn in a written request at the end of the class where your work is returned

# Textbook and Recommended Readings

- **Required textbook:**
  - Contemporary Logic Design by R. Katz & G. Borriello
- Recommended textbook:
  - Digital Design by F. Vahid
- Lecture slides are derived from the slides designed for both books



# Hardware we will use

- **Freely available in CSE 3219 lab:**
  - **Xilinx Virtex-II Pro Development System (XUPV2P)**  
<http://www.xilinx.com/univ/xupv2p.html>
  - **PC in the lab already have ISE tools installed. You can program boards in the lab only!**
  - **You can download on your own PC Webpack to implement and test your design before programming the board**  
[www.xilinx.com/ise/logic\\_design\\_prod/webpack.htm](http://www.xilinx.com/ise/logic_design_prod/webpack.htm)
- **Alternative: Altera Board you can buy in the bookstore for \$100**



# Outline



- Introduction to Xilinx board & tools
- Transistors
  - How they work
  - How to build basic gates out of transistors
  - How to evaluate delay
- Pass gates
- Muxes




# Basic FPGA Architecture

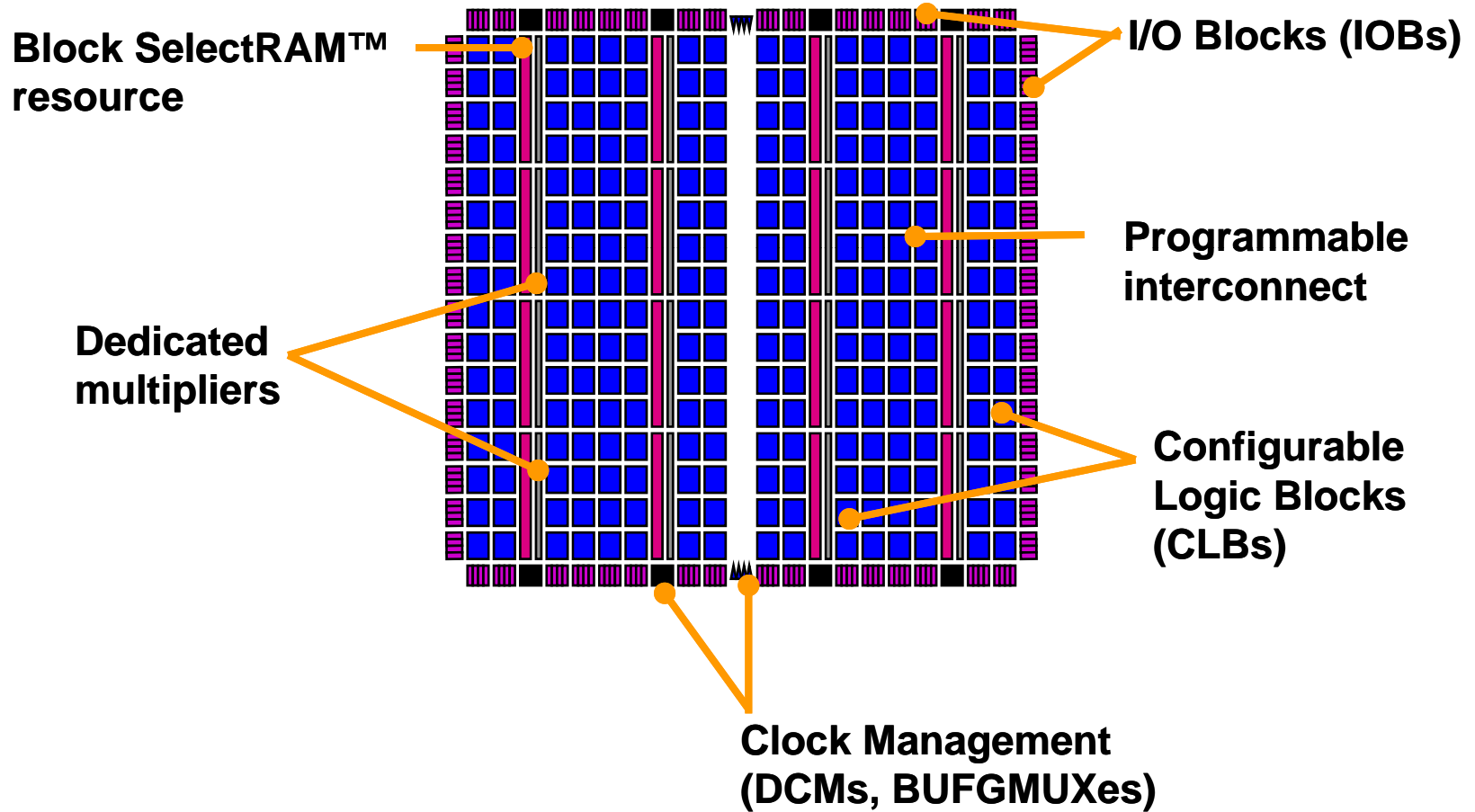


# Overview



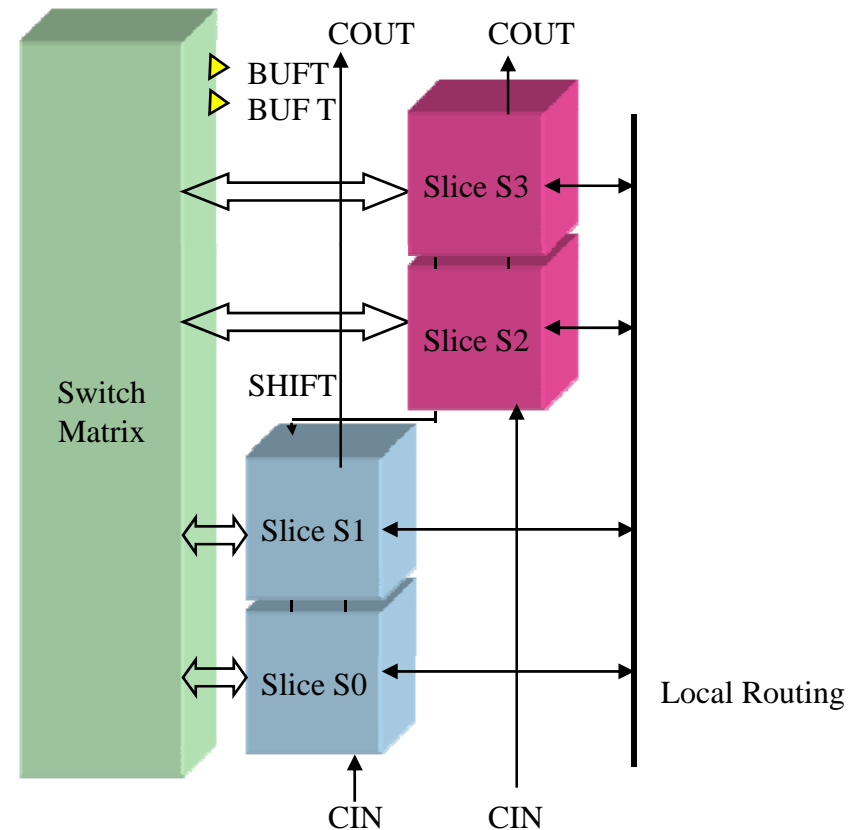
- All Xilinx FPGAs contain the same basic resources
    - Slices grouped into configurable logic blocks - CLBs
      - Contain combinatorial logic and register resources
    - Input/Output Blocks - IOBs
      - Interface between the FPGA and the outside world
    - Programmable interconnect
    - Other resources
      - Processor
      - Memory
      - Multipliers
      - Global clock buffers
      - Boundary scan logic
- 

# Virtex-II Architecture



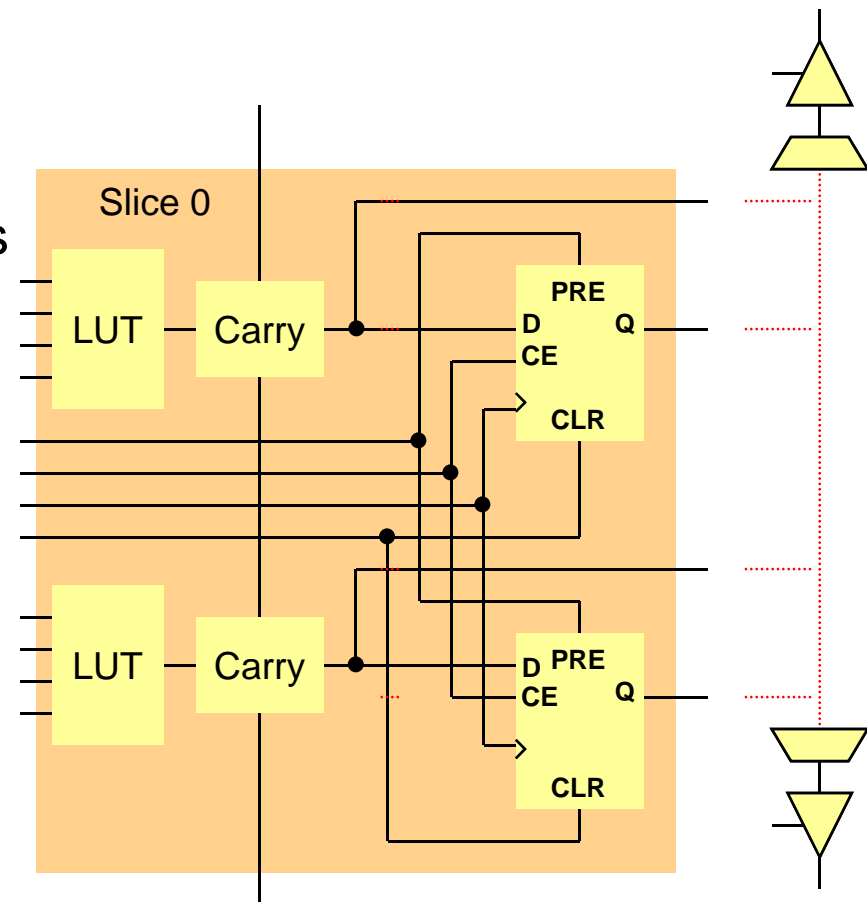
# Slices and CLB

- Each Virtex™-II CLB contains four slices
  - Local routing provides feedback between slices in the same CLB, and it provides routing to neighboring CLBs
  - A switch matrix provides access to general routing resources



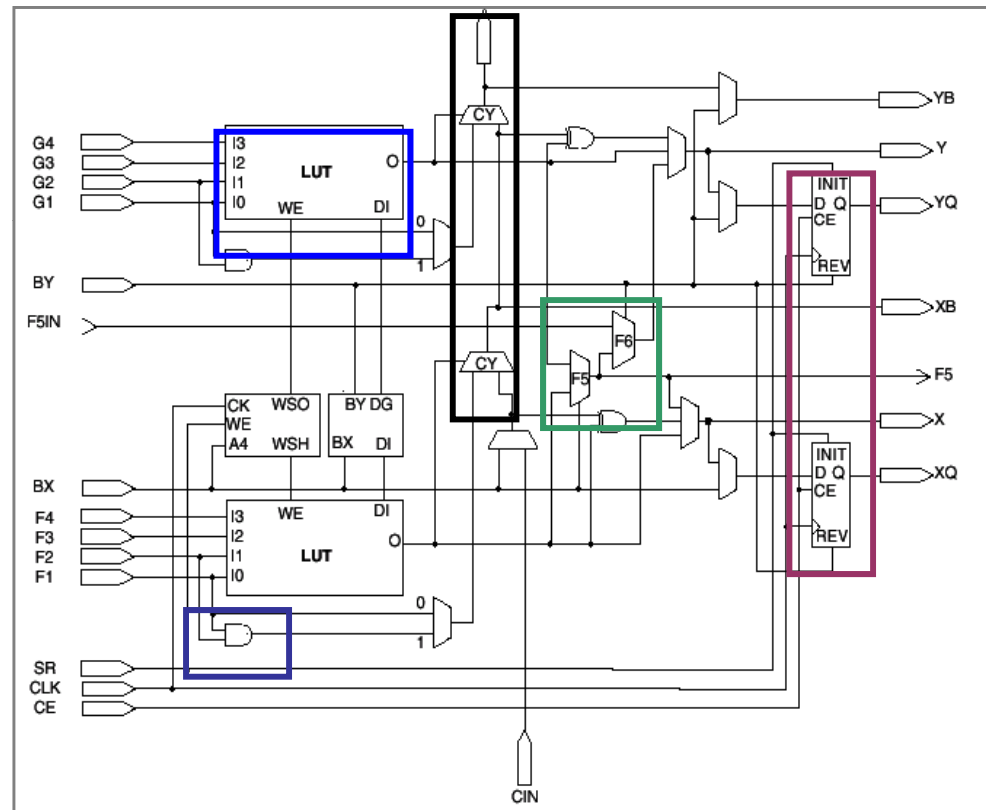
# Simplified Slice Structure

- Each slice has four outputs
  - Two registered outputs, two non-registered outputs
  - Two BUFTs associated with each CLB, accessible by all 16 CLB outputs
- Carry logic runs vertically, up only
  - Two independent carry chains per CLB



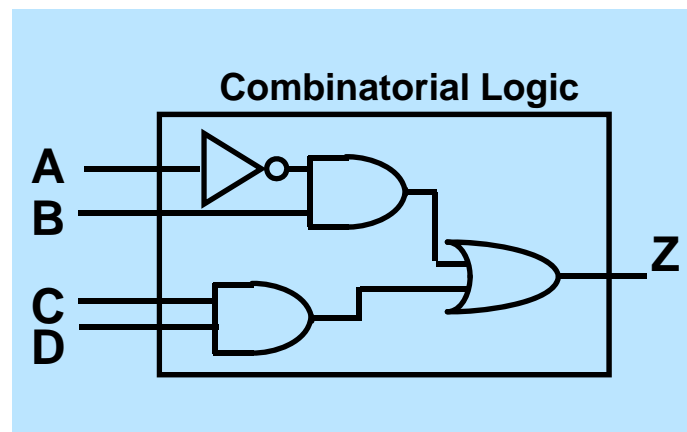
# Detailed Slice Structure

- The next few slides discuss the slice features
  - LUTs
  - MUXF5, MUXF6, MUXF7, MUXF8 (only the F5 and F6 MUX are shown in this diagram)
  - Carry Logic
  - MULT\_ANDs
  - Sequential Elements



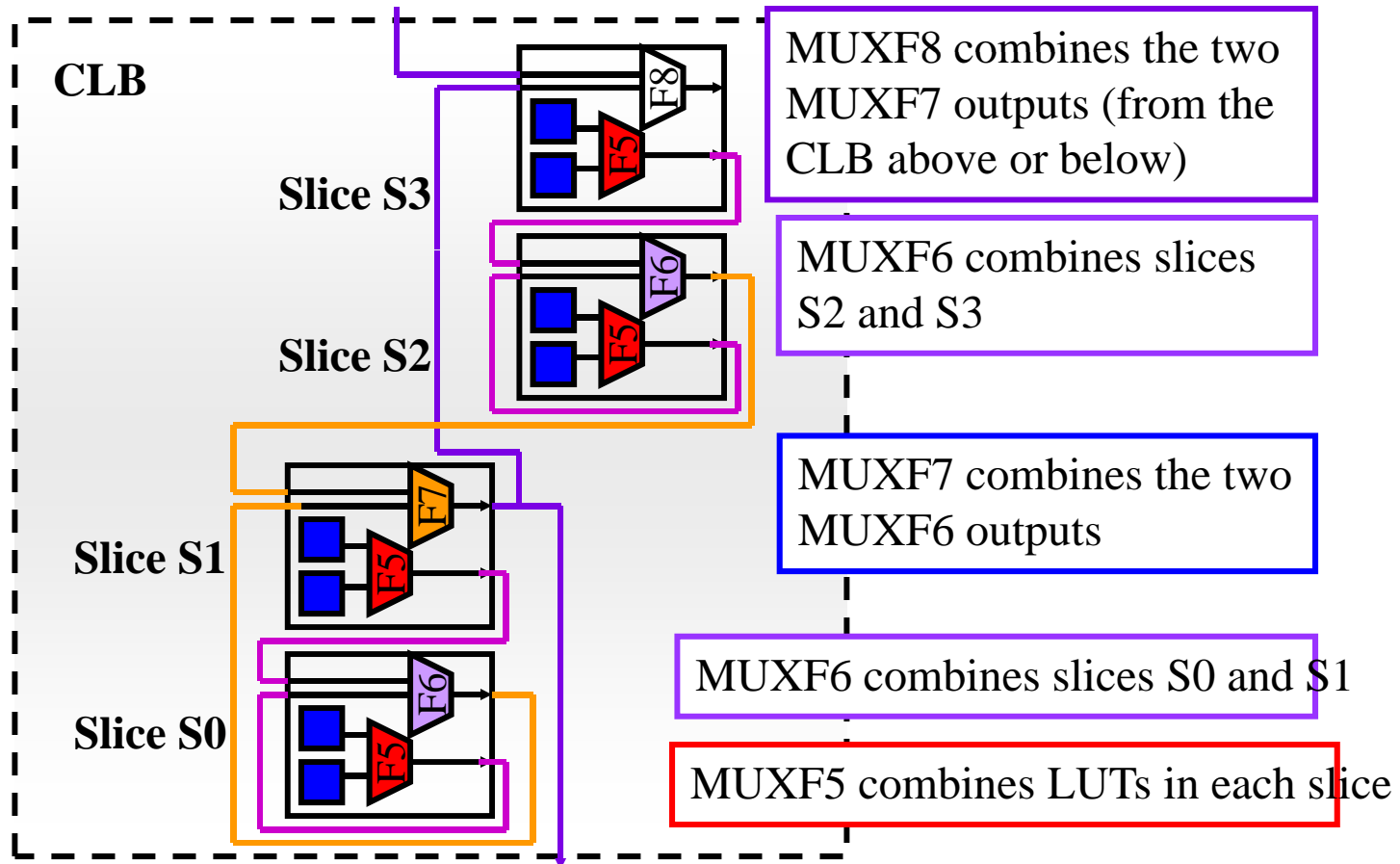
# Look-Up Tables

- Combinatorial logic is stored in Look-Up Tables (LUTs)
  - Also called Function Generators (FGs)
  - Capacity is limited by the number of inputs, not by the complexity
- Delay through the LUT is constant



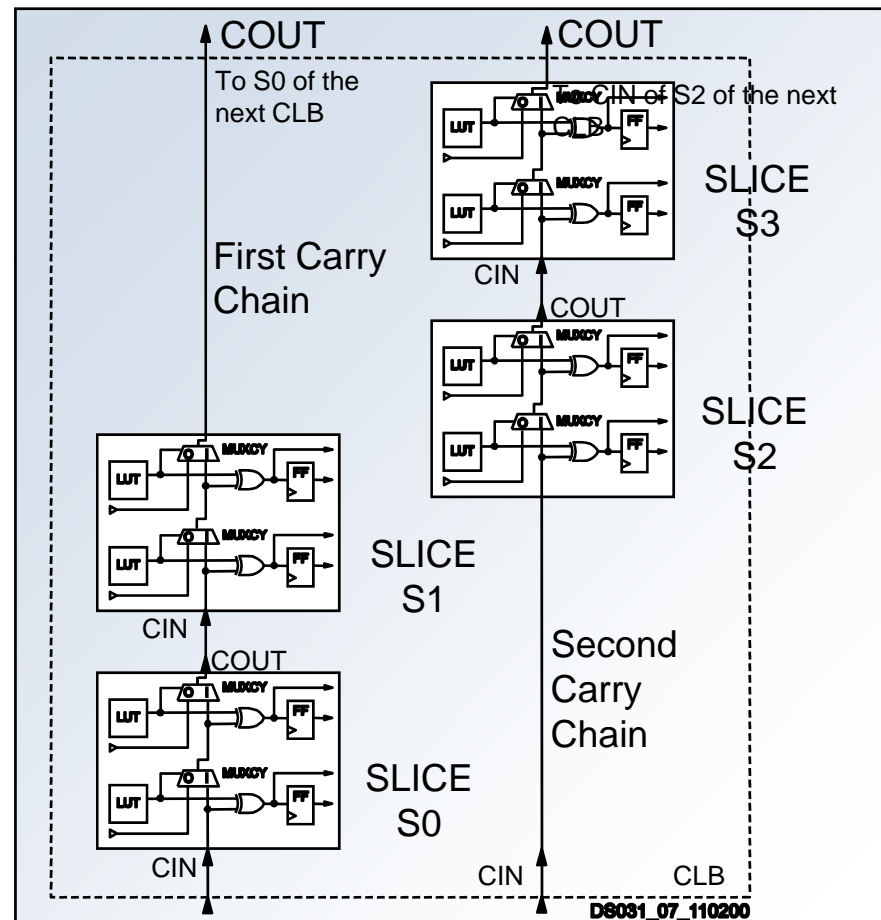
A	B	C	D	Z
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	1
0	1	0	0	1
0	1	0	1	1
.	.	.	.	.
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

# Connecting Look-Up Tables



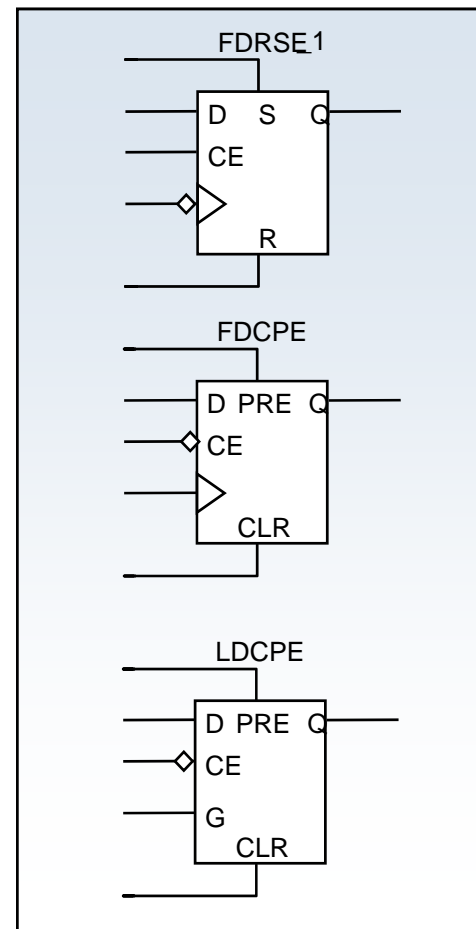
# Fast Carry Logic

- Simple, fast, and complete arithmetic Logic
  - Dedicated XOR gate for single-level sum completion
  - Uses dedicated routing resources
  - All synthesis tools can infer carry logic



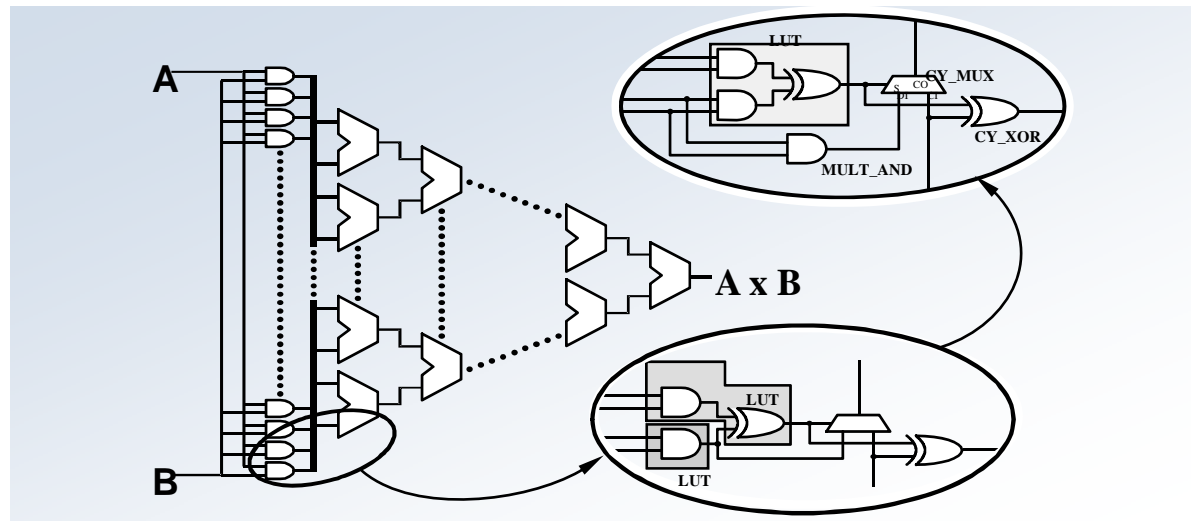
# Flexible Sequential Elements

- Either flip-flops or latches
- Two in each slice; eight in each CLB
- Inputs come from LUTs or from an independent CLB input
- Separate set and reset controls
  - Can be synchronous or asynchronous
- All controls are shared within a slice
  - Control signals can be inverted locally within a slice



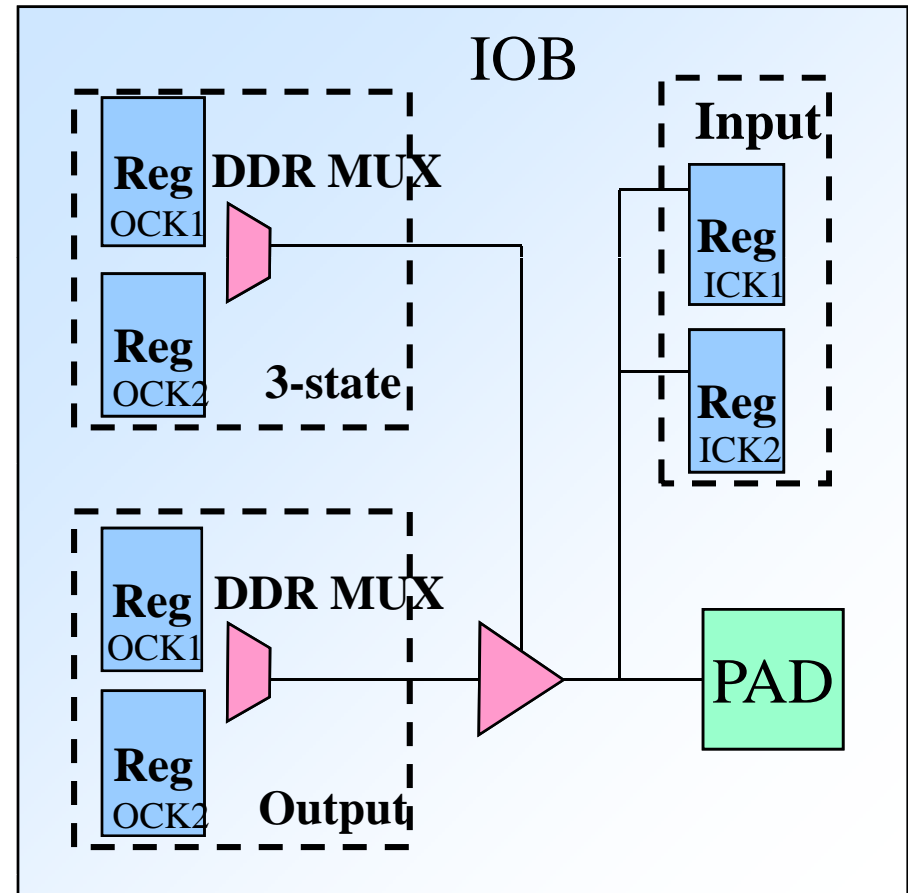
# MULT\_AND Gate

- Highly efficient multiply and add implementation
  - Earlier FPGA architectures require two LUTs per bit to perform the multiplication and addition
  - The MULT\_AND gate enables an area reduction by performing the *multiply* and the *add* in one LUT per bit



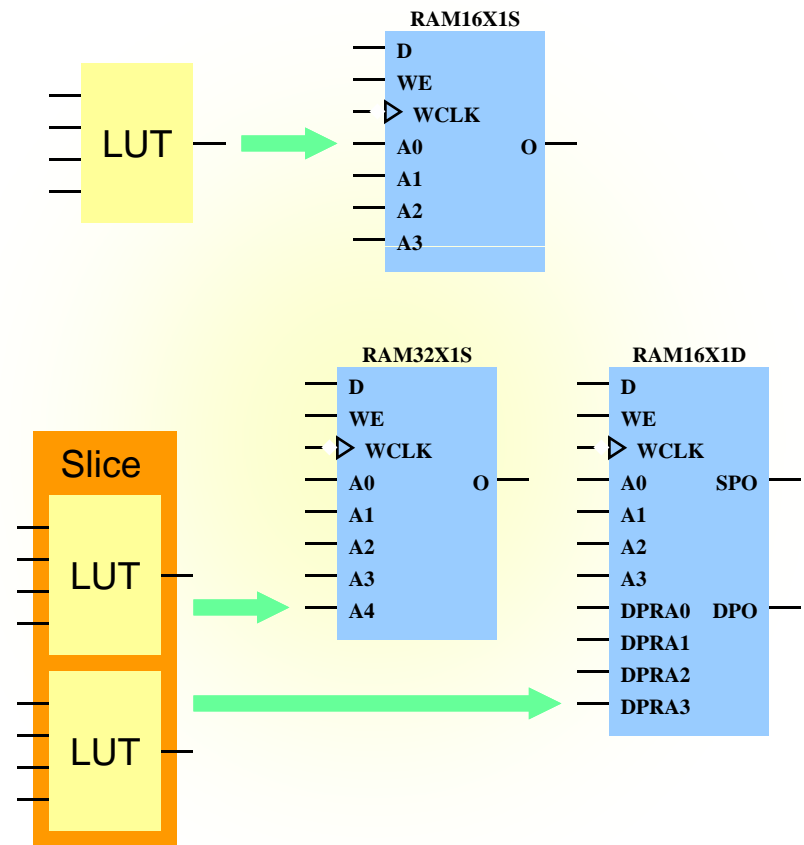
# IOB Element

- Input path
  - Two DDR registers
  - DDR: double data rate
- Output path
  - Two DDR registers
  - Two 3-state enable DDR registers
- Separate clocks and clock enables for I and O
- Set and reset signals are shared



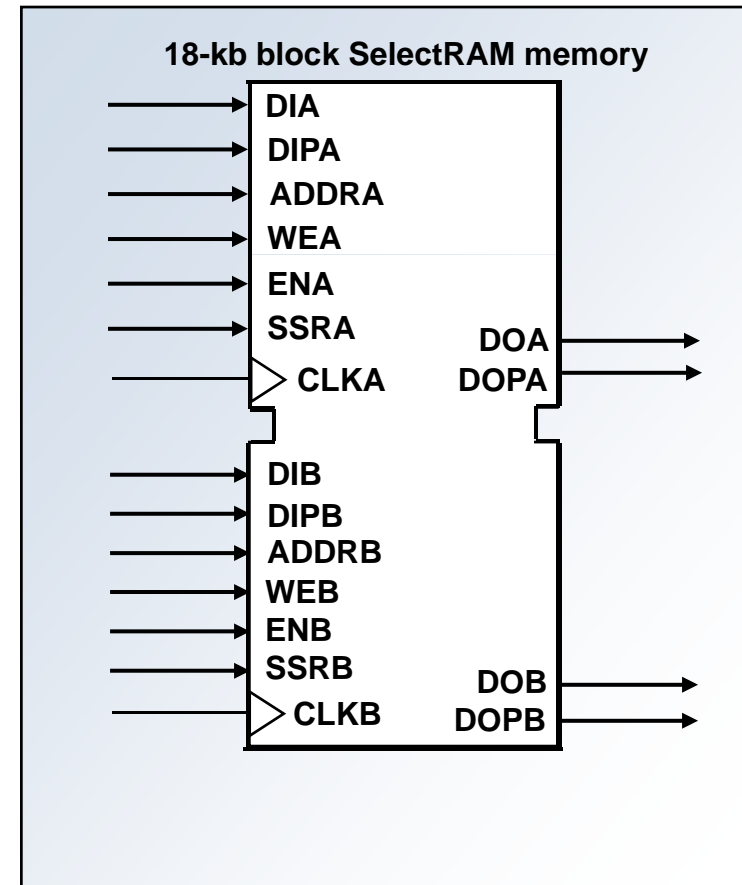
# Distributed SelectRAM Resources

- 1 LUT = 16 bits RAM
- Synchronous write
- Asynchronous read
  - Accompanying flip-flops can be used to create synchronous read
- RAM and ROM are initialized during configuration
  - Data can be written to RAM after configuration
- Emulated dual-port RAM
  - One read/write port
  - One read-only port



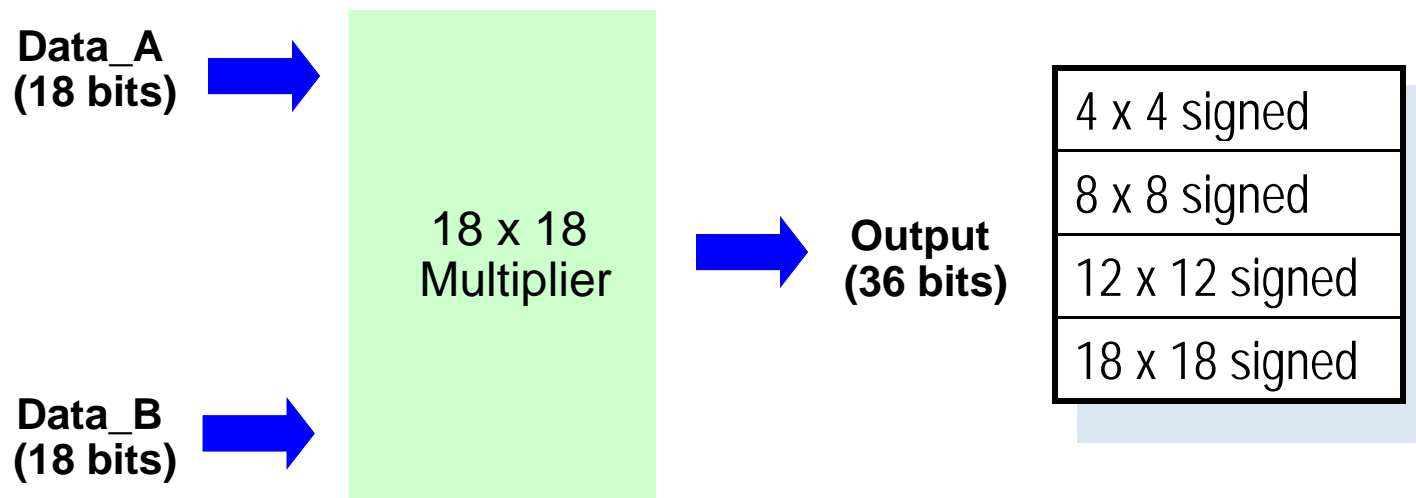
# Block SelectRAM Resources

- Up to 3.5 Mb of RAM in 18-kb blocks
  - Synchronous read and write
- True dual-port memory
  - Each port has synchronous read and write capability
  - Different clocks for each port
- Supports initial values
- Synchronous reset on output latches
- Supports parity bits
  - One parity bit per eight data bits



# Dedicated Multiplier Blocks

- 18-bit two's complement signed operation
- Optimized to implement Multiply and Accumulate functions
- Multipliers are physically located next to block SelectRAM™ memory



# Virtex-II Pro Features



- 0.13 micron process
- Up to 24 RocketIO™ Multi-Gigabit Transceiver (MGT) blocks
  - Serializer and deserializer (SERDES)
  - Fibre Channel, Gigabit Ethernet, XAUI, Infiniband compliant transceivers, and others
  - 8-, 16-, and 32-bit selectable FPGA interface
  - 8B/10B encoder and decoder
- PowerPC™ RISC processor blocks
  - Thirty-two 32-bit General Purpose Registers (GPRs)
  - Low power consumption: 0.9mW/MHz
  - IBM CoreConnect bus architecture support

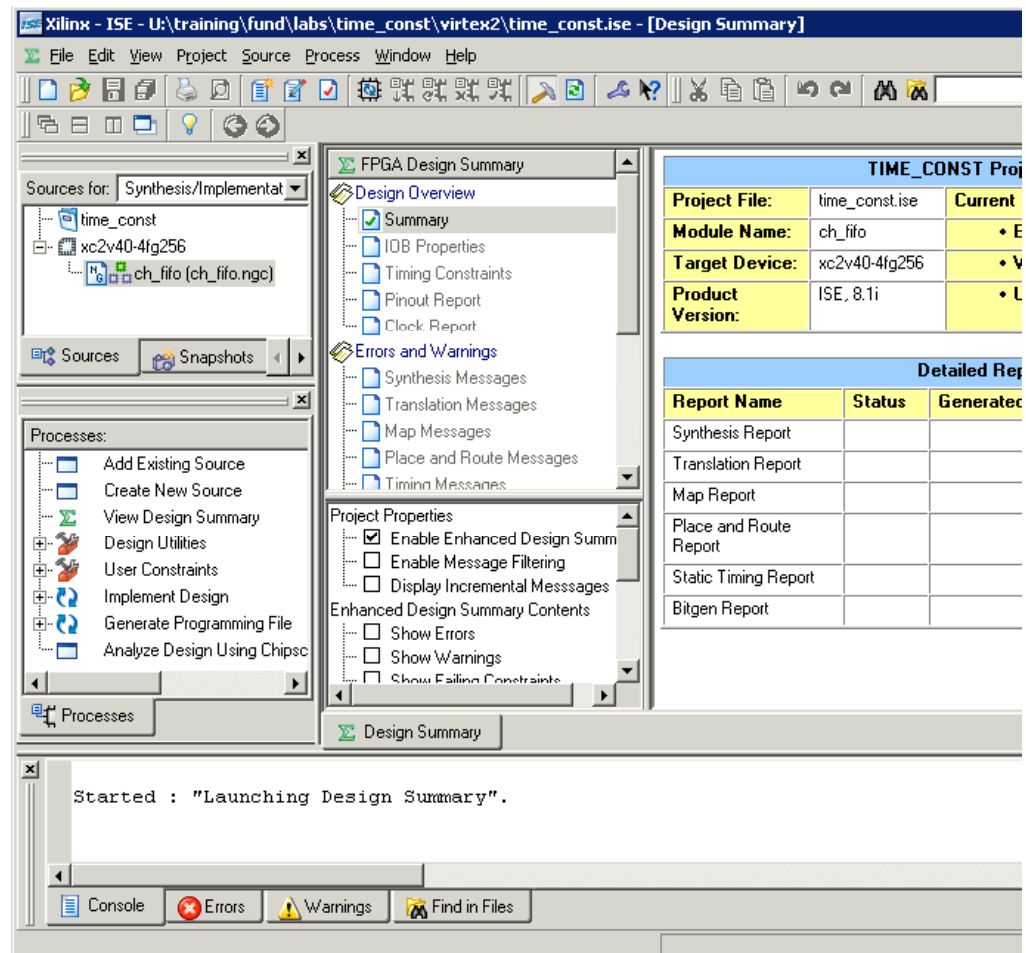


# Xilinx Design Flow

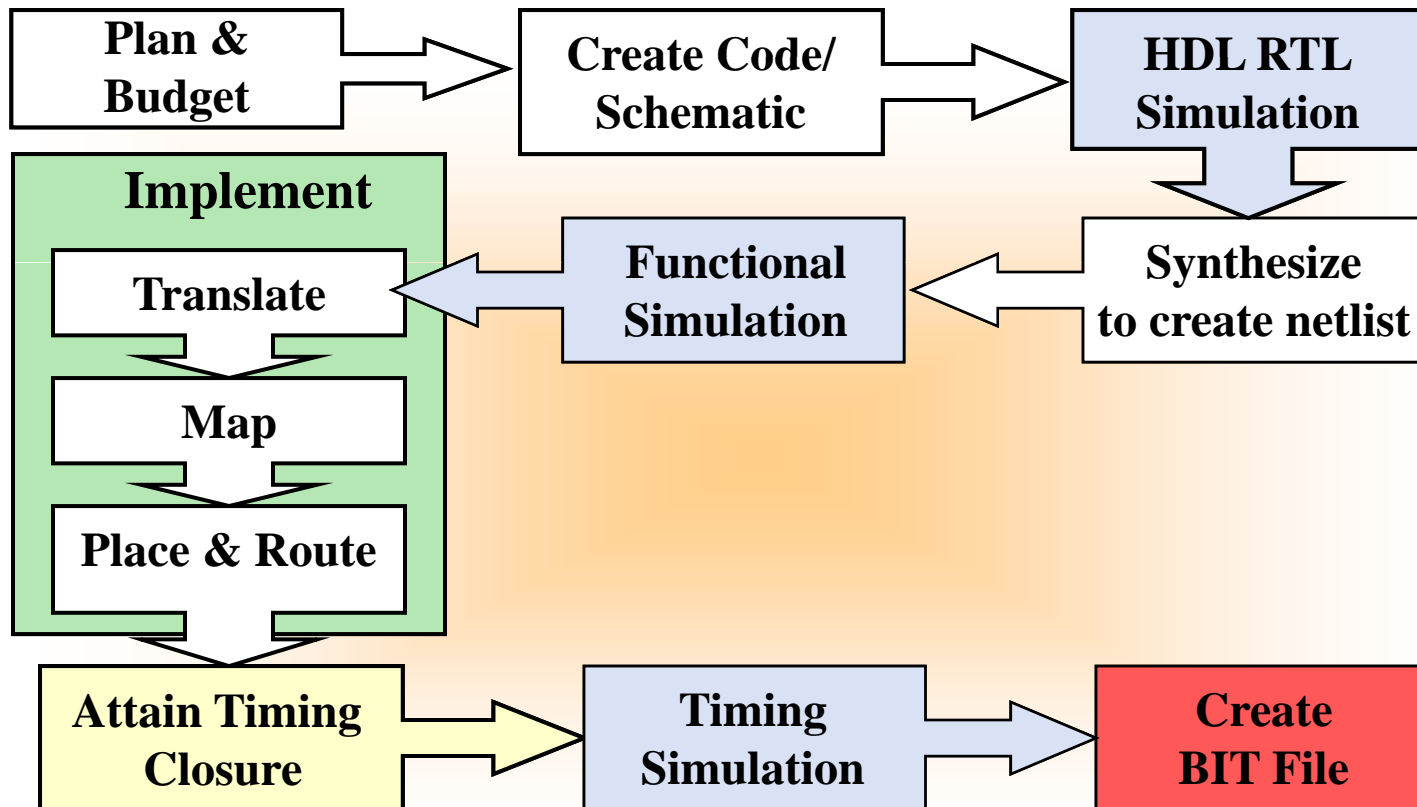


# ISE Project Navigator

- Built around the Xilinx design flow
  - Access to synthesis and schematic tools
    - Including third-party synthesis tools
  - Implement your design with a simple double-click
    - Fine-tune with easy-to-access software options




# Xilinx Design Flow



# Download & Run!!

- Once a design is implemented, you must create a file that the FPGA can understand
  - This file is called a bitstream: a BIT file (.bit extension)
- The BIT file can be downloaded directly into the FPGA, or the BIT file can be converted into a PROM file, which stores the programming information
- **Execute!!!**
  - You will be responsible for a demo to TA
  - Turn in a report with answers to questions asked, schematics, truth tables, equations etc.



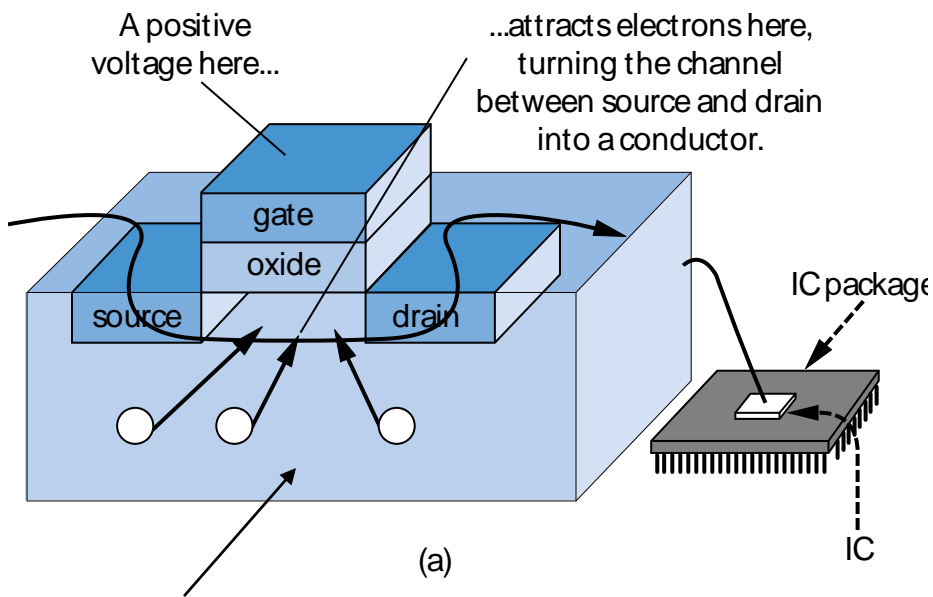


# Combinational circuit building blocks: Transistors, gates and timing

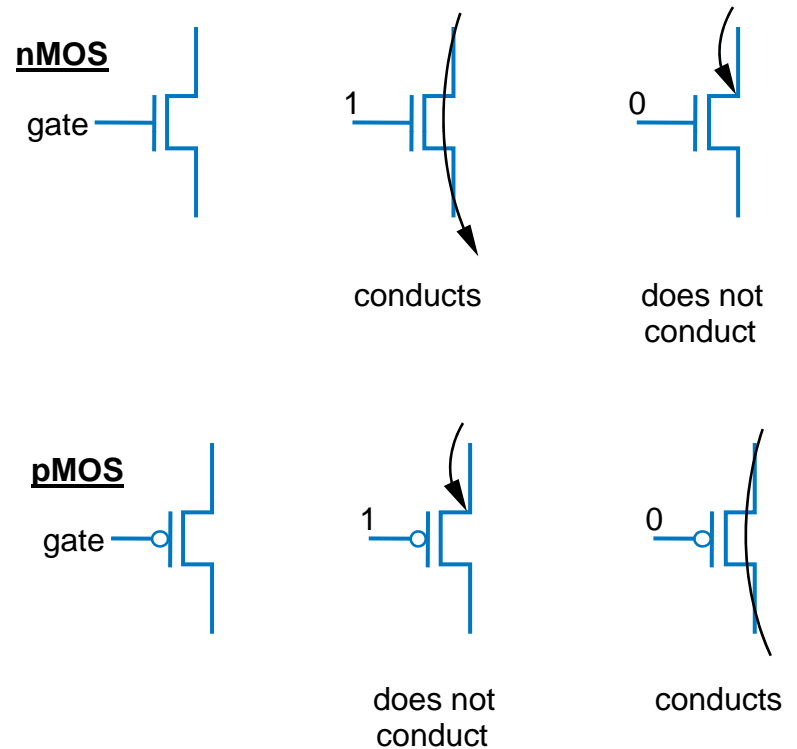
Tajana Simunic Rosing

# The CMOS Circuit

- CMOS circuit
  - Consists of N and PMOS transistors
  - Both N and PMOS operate similar to basic switches

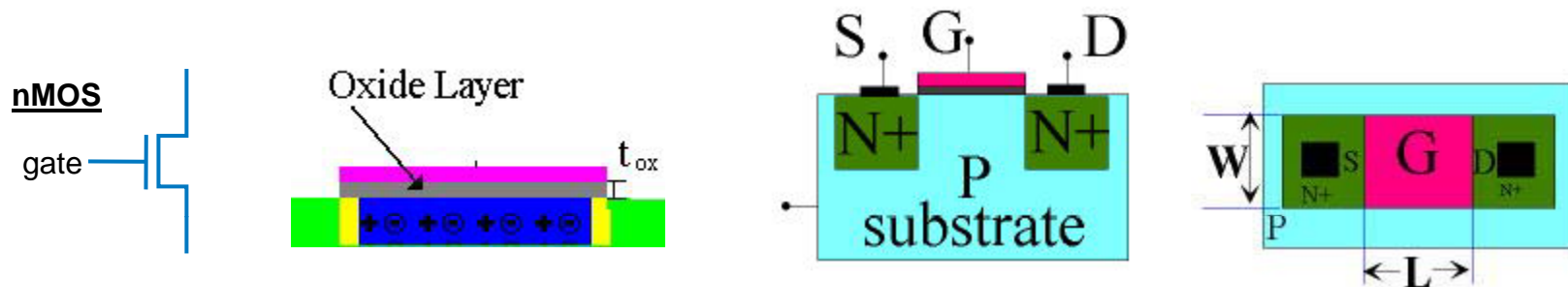


Silicon -- not quite a conductor or insulator:  
***Semiconductor***

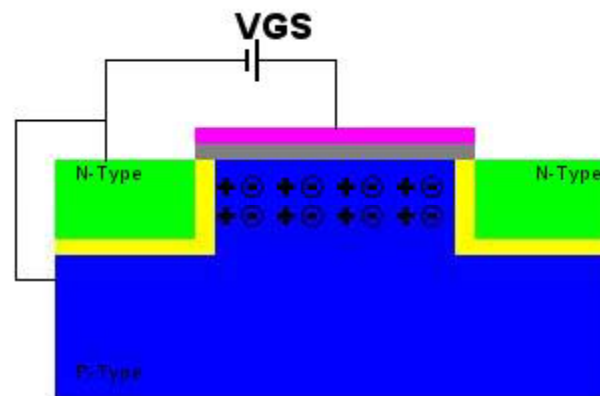


# N-MOS Tutorial – channel formation

- The Semiconductor-Oxide-Metal Combination in the Gate is effectively a **Parallel Plate Capacitor**



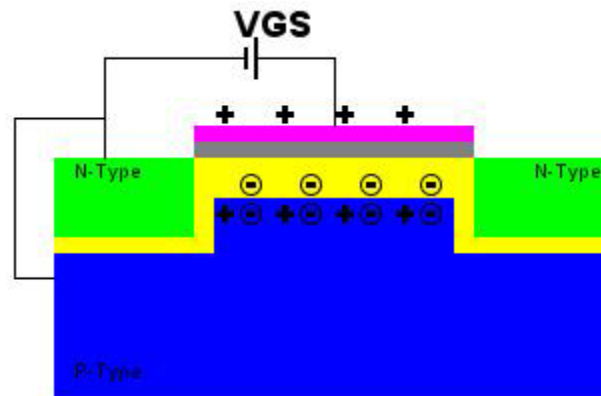
- $V_{gs} = 0 \rightarrow$  lots of positive charge in p-type material, no current



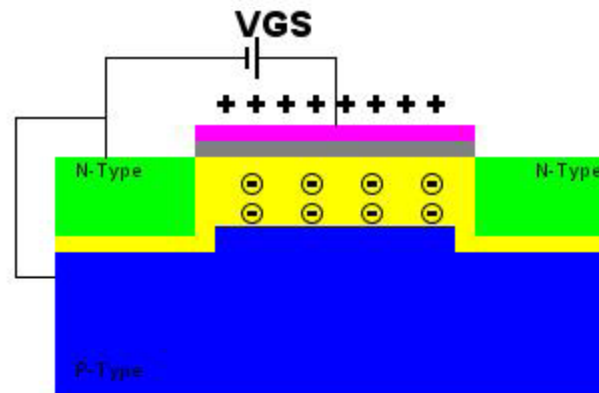
- Source: <http://www.netsoc.tcd.ie/~mcgettrs/hvmenu/tutorials/TOCcmostran.htm>

# N-MOS Tutorial – channel formation (cont.)

- $V_{gs} > 0$  -> + charge on the gate, - charge attracted to the oxide, + charge chased away from the oxide

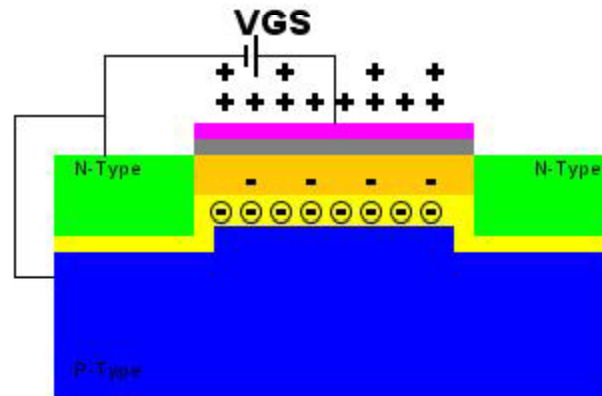


- $V_{gs} = V_t$  -> channel of negative charge forms under the oxide; the oxide is depleted of + charge;  $V_t$  = threshold voltage

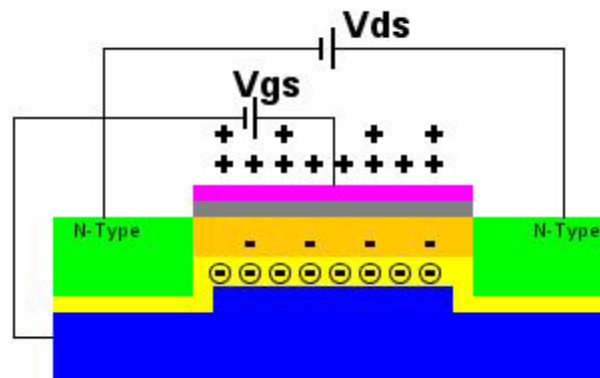


# N-MOS Tutorial – channel formation (cont.)

- $V_{gs} > V_t$  -> negative charge carriers form under the oxide; free electrons are thermally generated and form a conduction channel through which current can flow

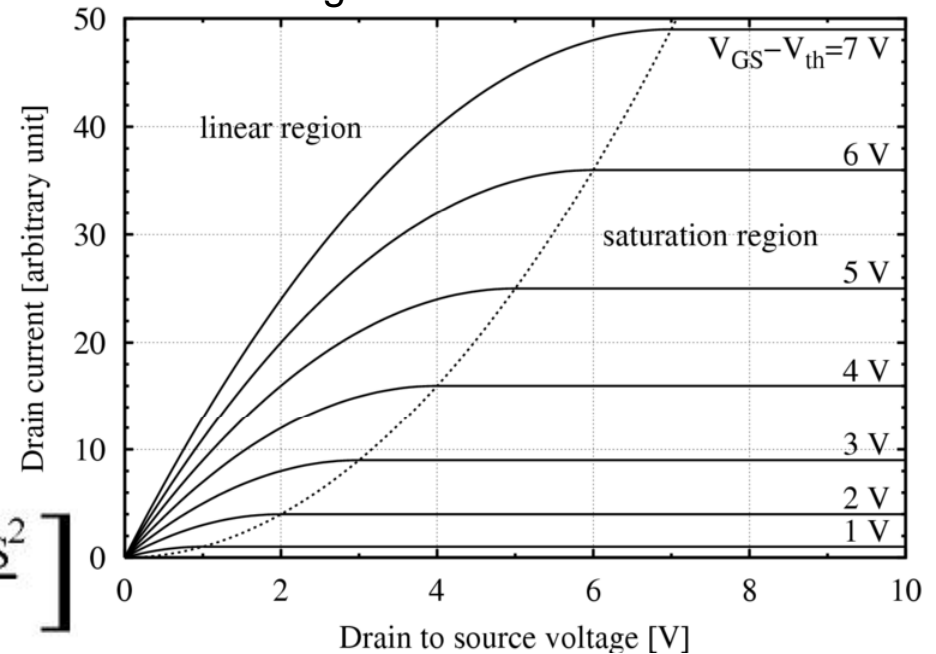
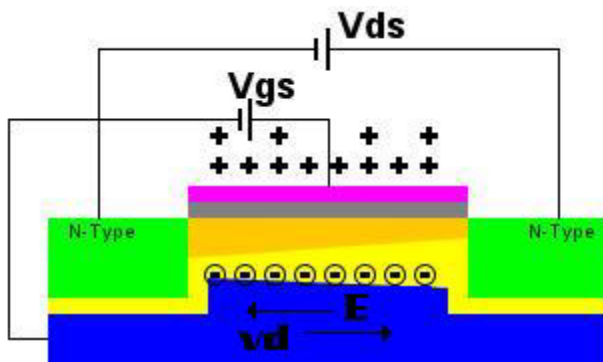


- $V_{gs} > V_t$  &  $V_{ds} = 0$  -> channel present, but no current flows



# N-MOS Tutorial: Current flow

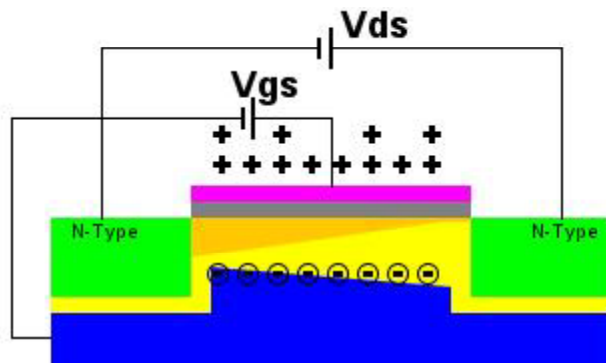
- $V_{ds} > 0$  -> electric field (E) set up between source and drain, accelerates electrons with velocity  $v_d$ , small current forms between source and drain
  - $C_{ox}$  : oxide capacitance =  $\epsilon_{ox} / t_{ox}$  (oxide permittivity  $\epsilon_{ox}$  and thickness  $t_{ox}$ );
  - $\mu$ : mobility of charge carriers; W/L gate width and length



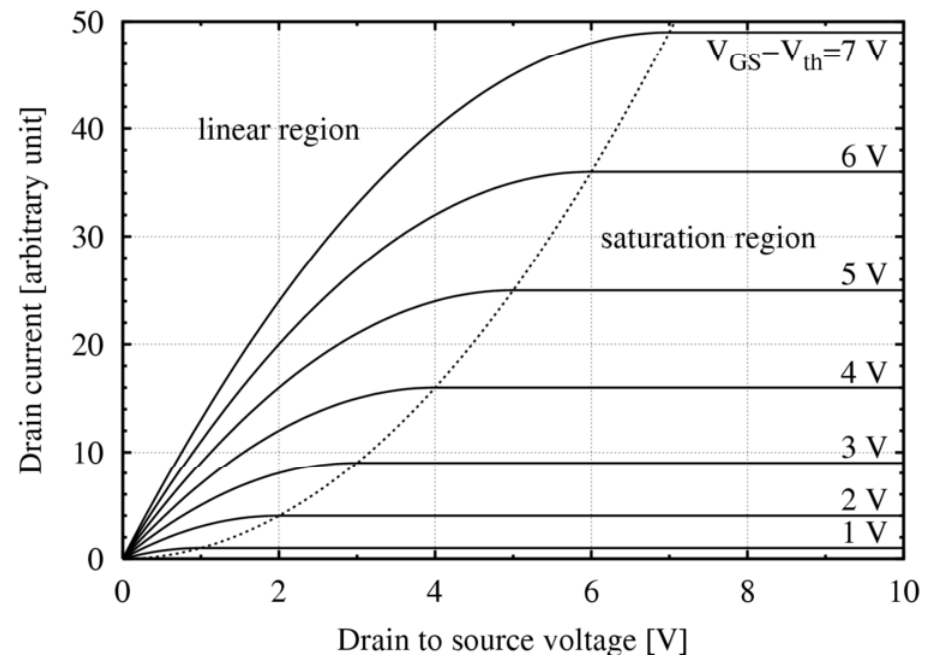
$$I_D = \mu C_{ox} \frac{W}{L} \left[ (V_{GS} - V_T) V_{DS} - \frac{V_{DS}^2}{2} \right]$$

# N-MOS Tutorial: Current flow

- $V_{ds} \geq V_{gs} - V_t \rightarrow$  channel pinched off, saturated; constant current flows from drain to source
  - $C_{ox}$  : oxide capacitance =  $\epsilon_{ox} / t_{ox}$  (oxide permittivity  $\epsilon_{ox}$  and thickness  $t_{ox}$ );  
 $\mu$ : mobility of charge carriers;  $W/L$  gate width and length

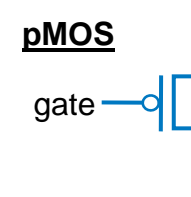
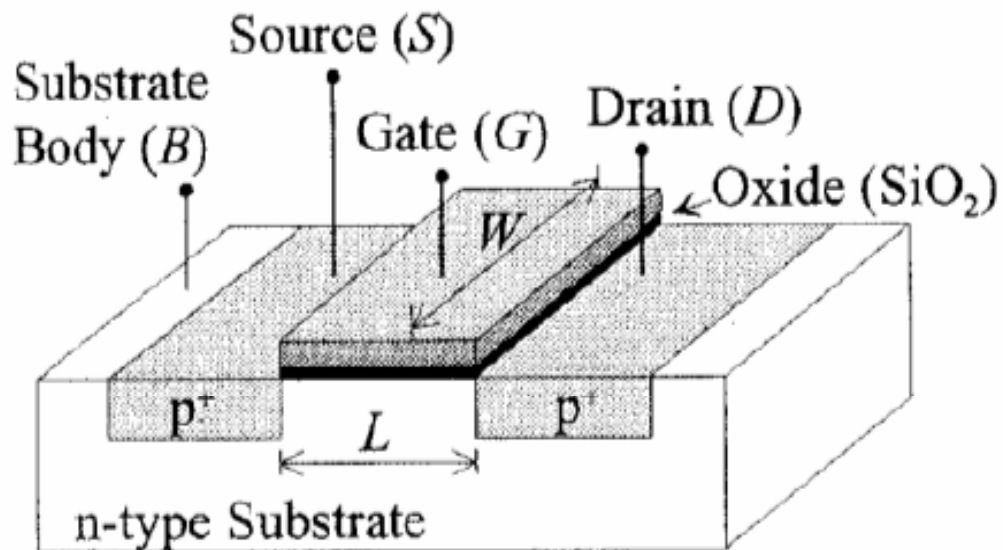


$$I_{DSat} = \frac{\mu_n C_{ox} W}{2 L} [V_{GS} - V_T]^2$$



# How about P-MOS?

- Everything is the same, but polarities of voltages reverse. Mobility ( $\mu$ ) is 2x smaller, so 2x less current is generated if all other parameters are kept constant
  - e.g. PMOS turns on when  $V_{gs} < V_t$  and both are  $< 0$



# Resistance

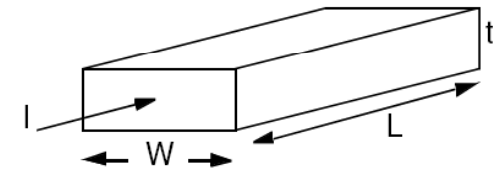
- Resistivity

- Function of:

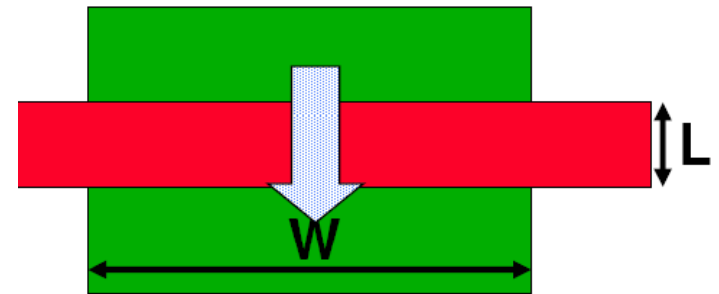
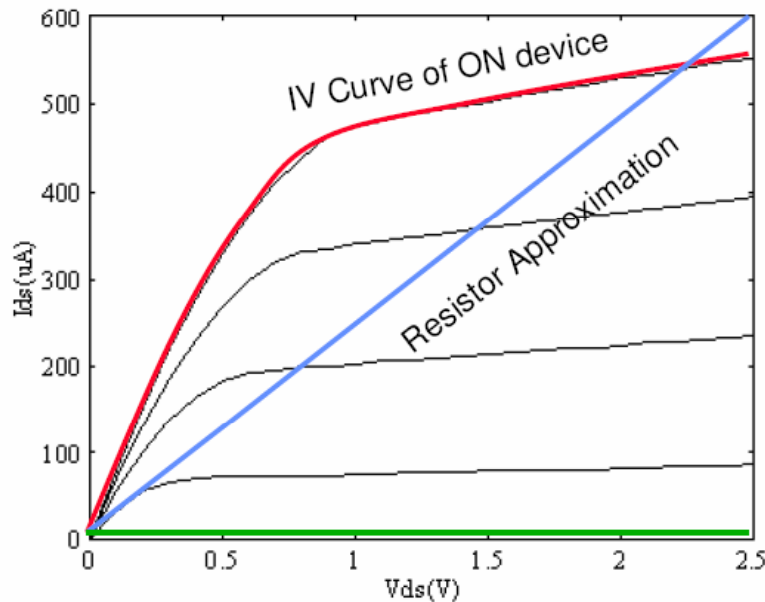
- resistivity  $r$ , thickness  $t$  : defined by technology
- Width  $W$ , length  $L$ : defined by designer

- Approximate ON transistor with a resistor

- $R = r' L/W$
- $L$  is usually minimum; change only  $W$

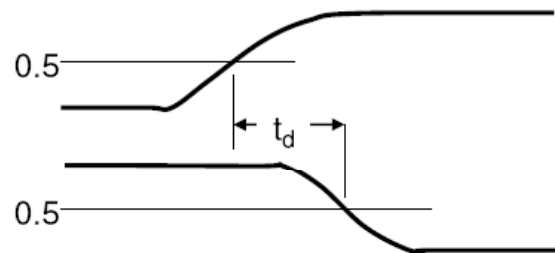


$$R = \frac{\rho L}{tW} = \frac{\rho}{t} \frac{L}{W}$$



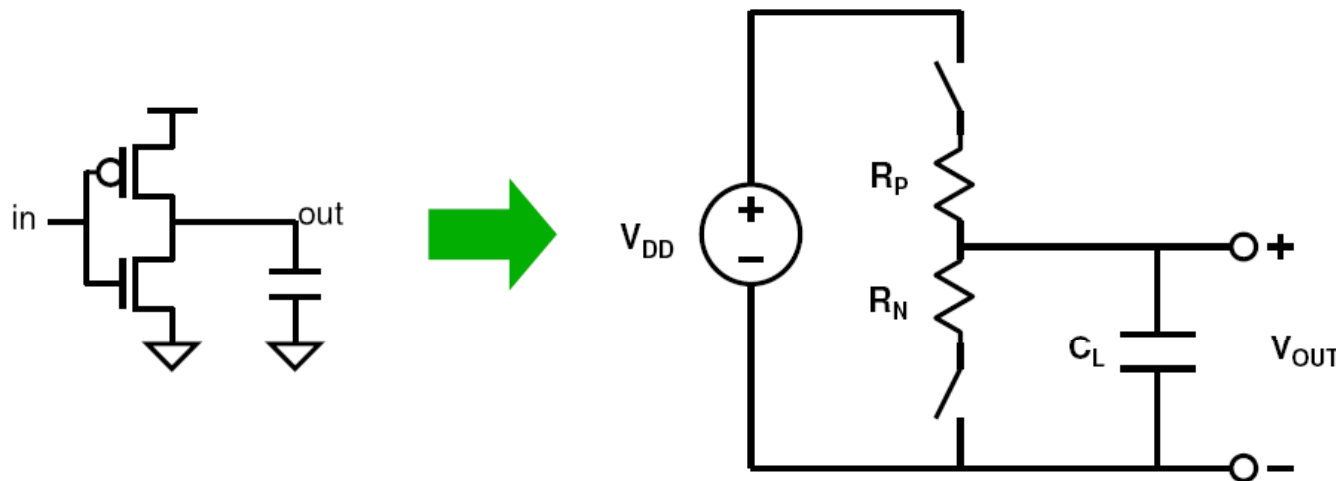
# Capacitance & Timing estimates

- Capacitor
  - Stores charge  $Q = C V$  (capacitance  $C$ ; voltage  $V$ )
  - Current:  $dQ/dt = C dV/dt$
- Timing estimate
  - $D t = C dV / i = C dV / (V/R_{\text{trans}}) = R_{\text{trans}} C dV/V$
- Delay: time to go from 50% to 50% of waveform



# Charge/discharge in CMOS

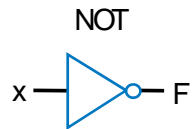
- Calculate on resistance
- Calculate capacitance of the gates circuit is driving
- Get RC delay & use it as an estimate of circuit delay
  - $V_{out} = V_{dd} (1 - e^{-t/R_p C})$



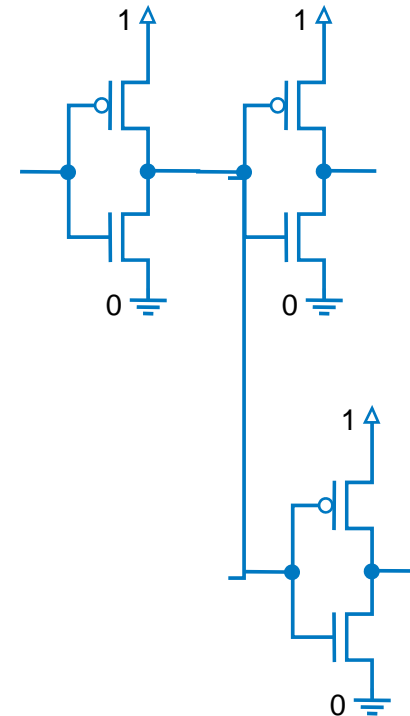
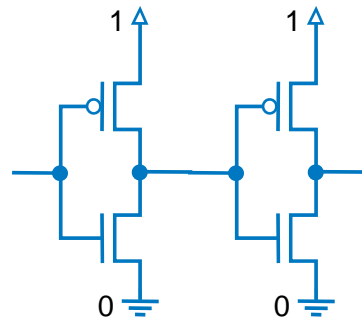
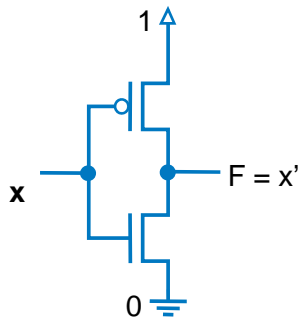
# Inverter delay



- Delay: estimate using RC time constants

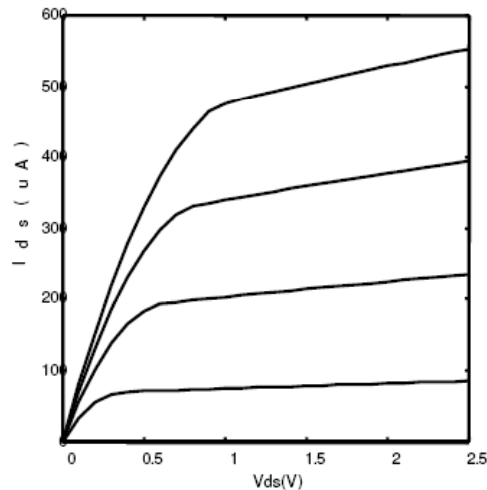


x	F
0	1
1	0

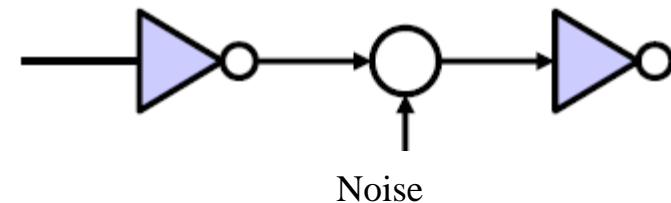
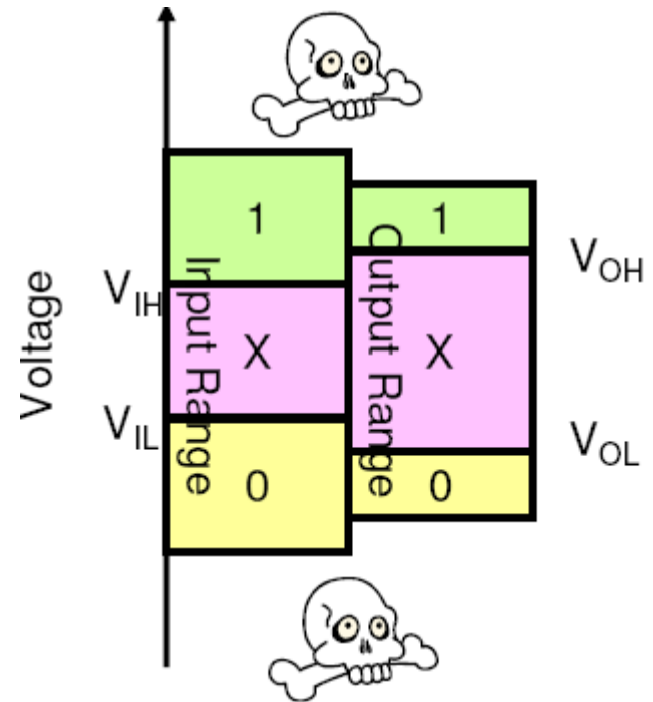


# Digital logic abstraction

- Real transistors: voltage at the gate controls the current between source and drain
  - Too complex! Simplify!

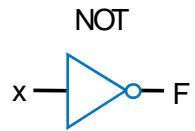


- Guarantee that voltage always falls within two regions: logic 0 and logic 1
  - Level restore
  - Great to filter out noise (noise margins)
- Use RC time constants to approximate delay

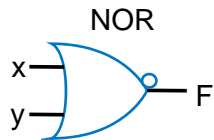
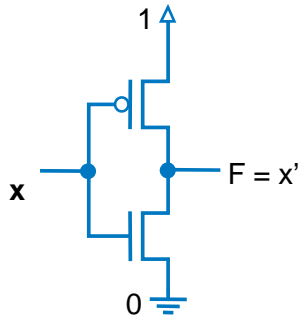


# Basic CMOS gates

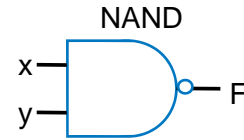
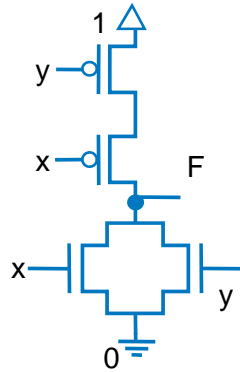
- Delay: estimate using RC time constants



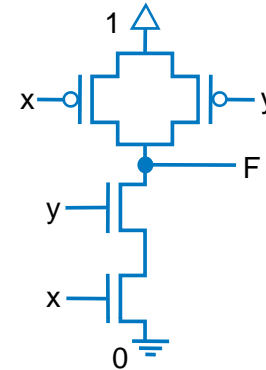
x	F
0	1
1	0



x	y	F
0	0	1
0	1	0
1	0	0
1	1	0



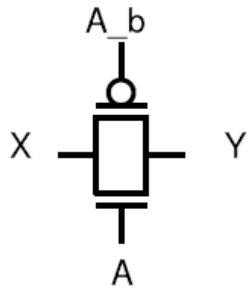
x	y	F
0	0	1
0	1	1
1	0	1
1	1	0



# CMOS Example

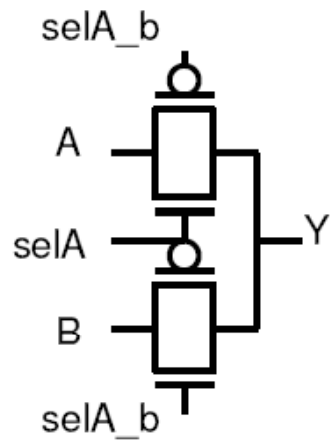
- The rules:
  - NMOS connects to GND, PMOS to power supply Vdd
  - Duality of NMOS and PMOS
  - $R_p \sim 2 R_n \Rightarrow$  PMOS in series is much slower than NMOS in series
- Implement Z using CMOS:  $Z = (A + BC)'$

# Pass transistor



- Connects X & Y when  $A=1$ , else X & Y disconnected
  - $A_b = \text{not}(A)$

# Mux



- Selects input to connect to Y
  - selA == 1: connects A to Y
  - selB == 1: connects B to Y

# What we've covered thus far



- Xilinx Virtex II Pro board and tools
- Transistor design
- Building basic gates from CMOS
- Delay estimates
- Pass transistors
- Muxes