



CSE140L: Components and Design Techniques for Digital Systems Lab

Introduction

Tajana Simunic Rosing

Welcome to CSE 140L!

- **Instructor: Tajana Simunic Rosing**
 - Email: tajana@ucsd.edu; please put CSE140L in the subject line
 - Ph. 858 534-4868
 - Office Hours: W 11:00-12:00pm; Th 11:30-12:30pm; CSE 2118
- **Instructor's Assistant: Sheila Manalo**
 - Email: shmanalo@ucsd.edu
 - Phone: (858) 534-8873
- **TA: Gautham Reddy**
Email: greddy@ucsd.edu
TA office hrs: Tu/Fri?
- **Class Website:**
 - <http://www.cse.ucsd.edu/classes/wi10/cse140L/>
- **Grades:** <http://webct.ucsd.edu>

Course Description

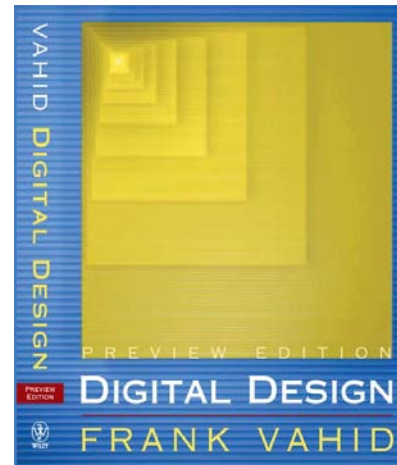
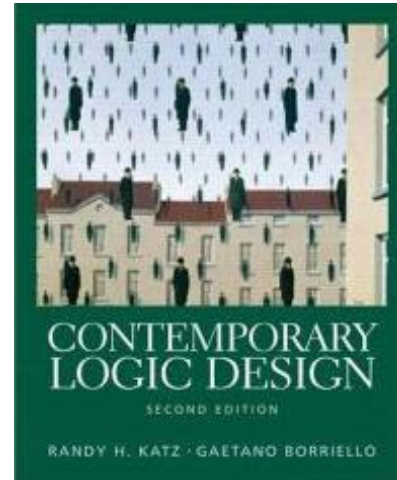
- Prerequisites:
 - CSE 20 or Math 15A, and CSE 30.
 - CSE 140 must be taken concurrently
- Objective:
 - Introduce digital components and system design concepts through hands-on experience in a lab
- Grading
 - Labs (4): 70%
 - First two labs will use simulation only, the 2nd two labs will use Xilinx HW
 - 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
 - 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



Software and 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 tools to implement and test your design; this is all that will be needed for the first two labs.**
www.xilinx.com/ise/logic_design_prod/webpack.htm



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

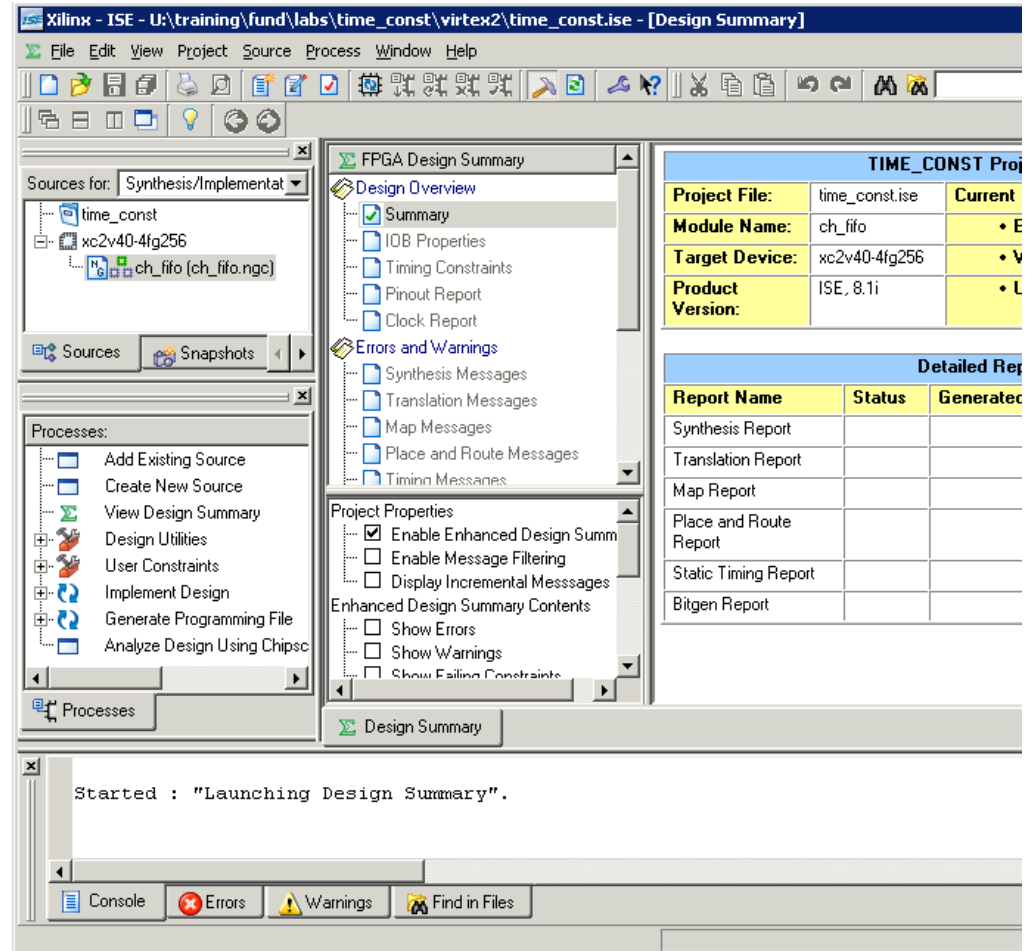


Quick intro to Xilinx board and tools

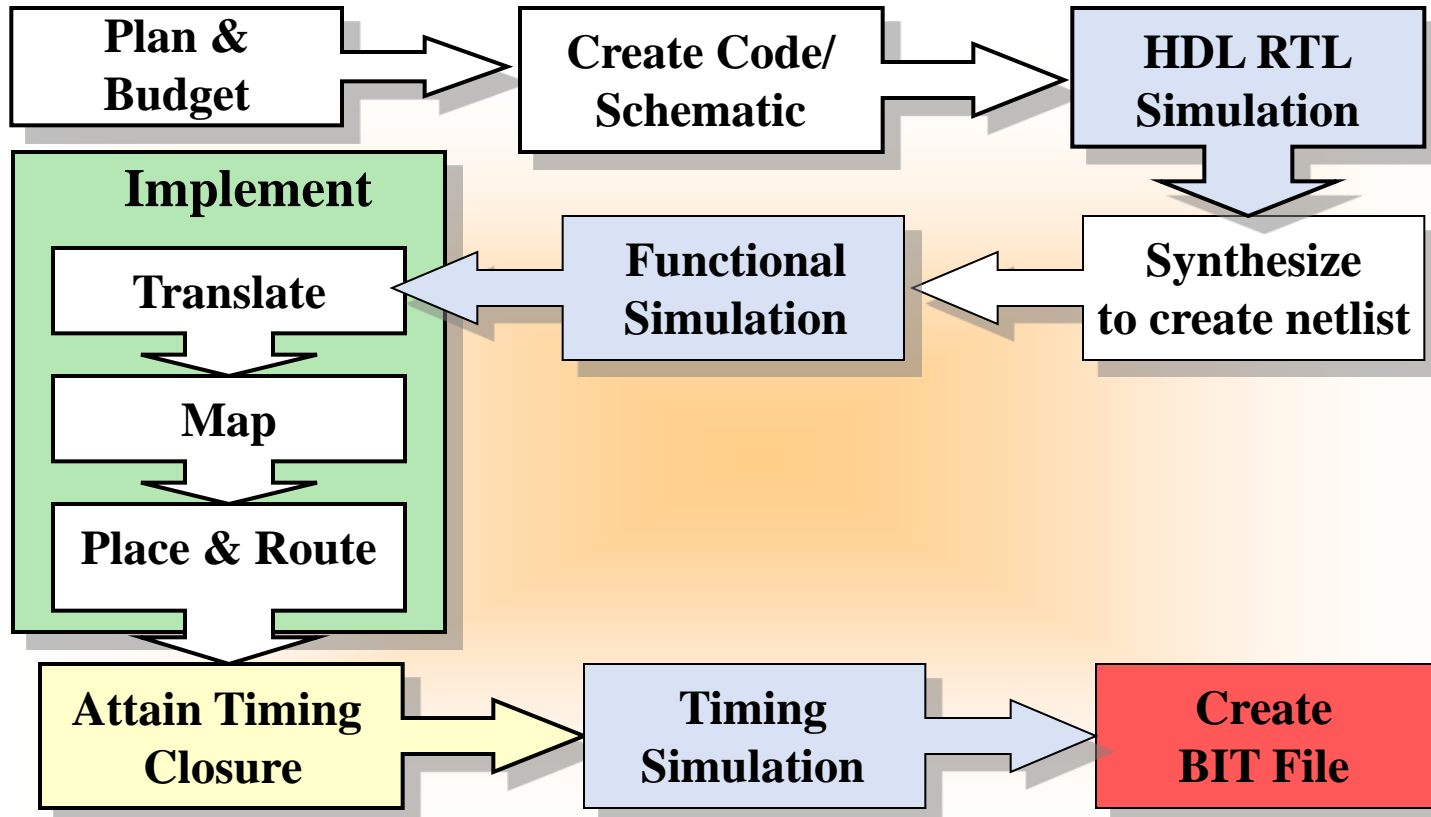



ISE Project Navigator

- 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



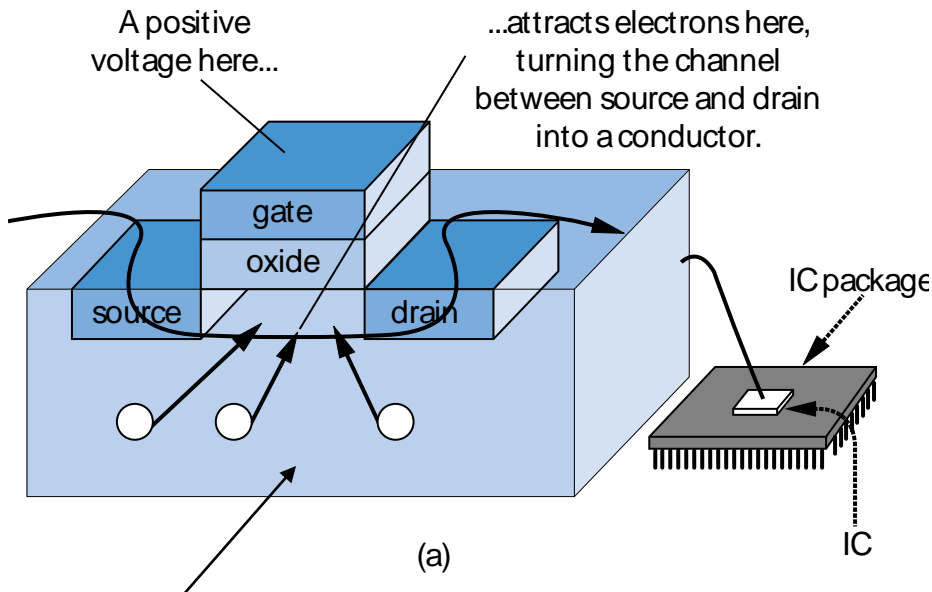


Combinational circuit building blocks: Transistors, gates and timing

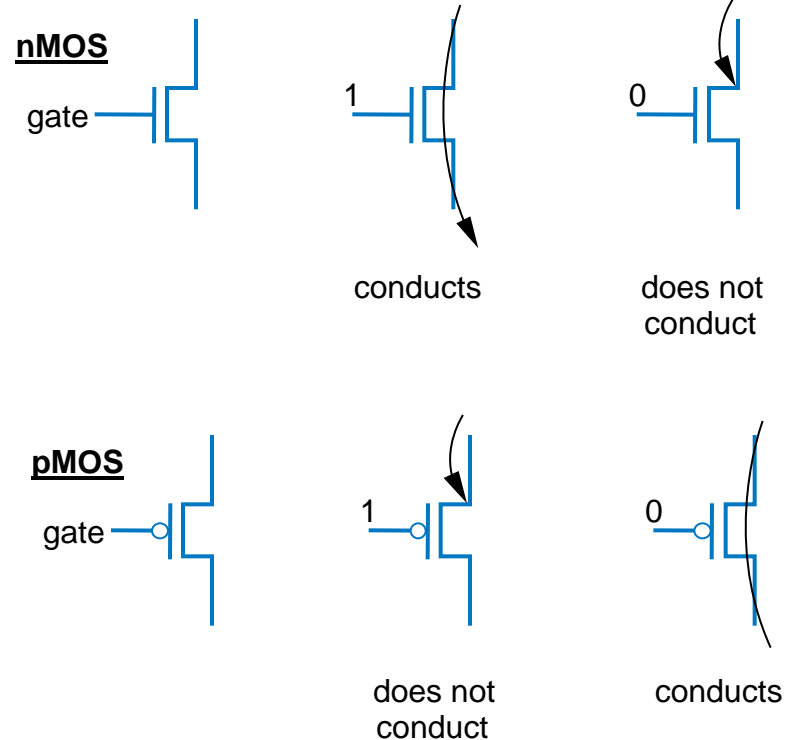
Tajana Simunic Rosing

The CMOS Circuit

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

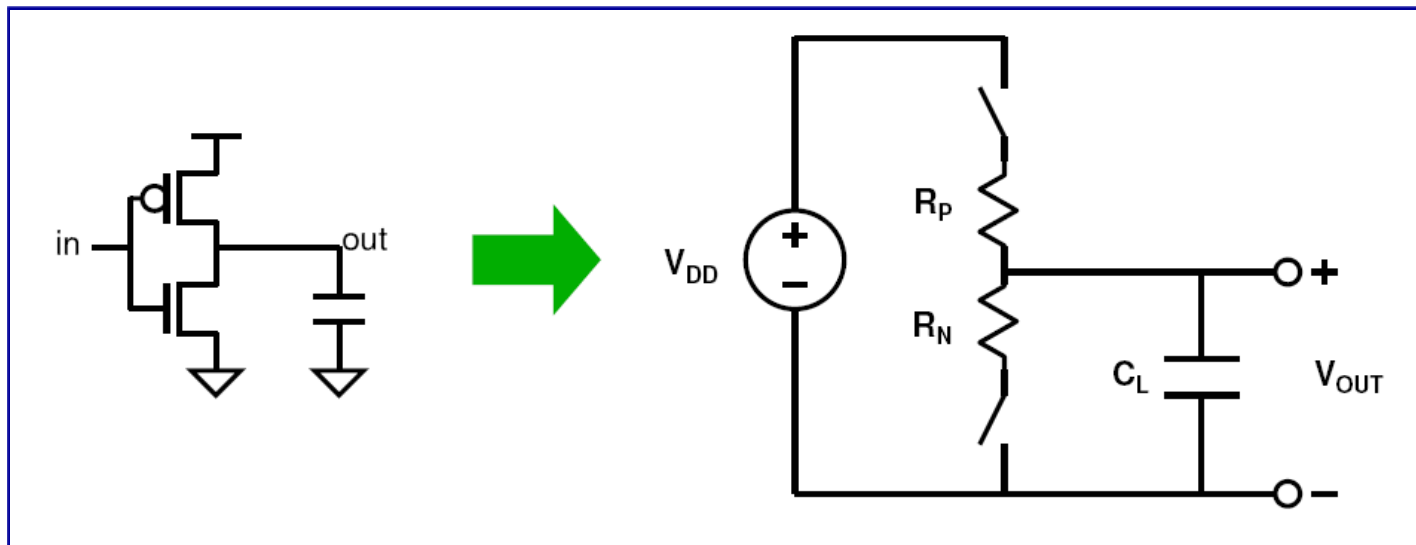


Silicon -- not quite a conductor or insulator:
Semiconductor

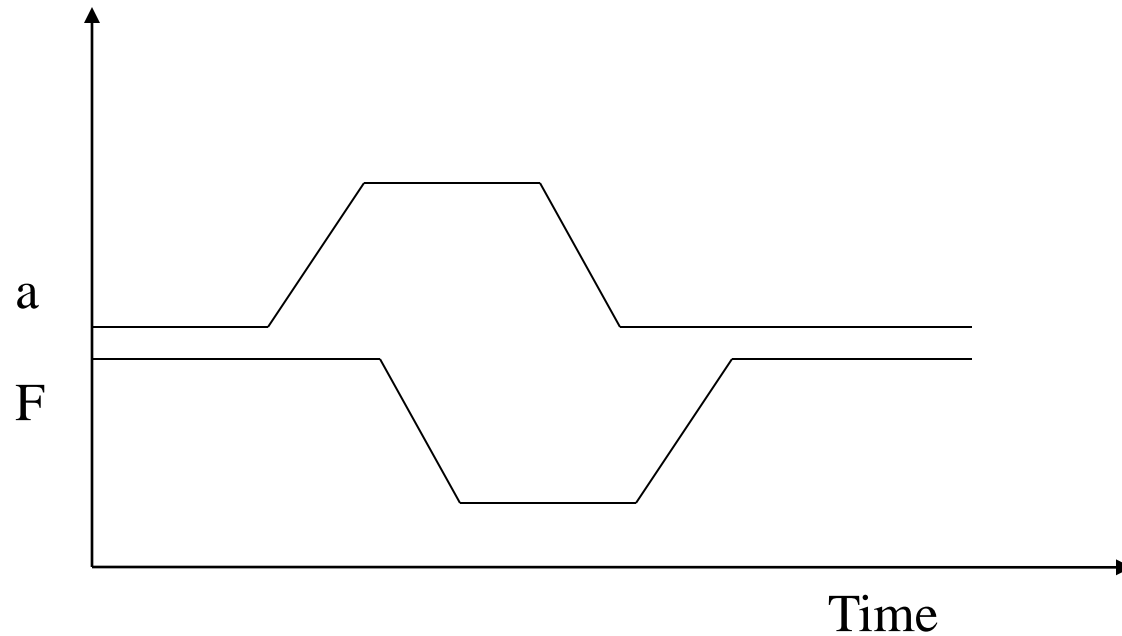
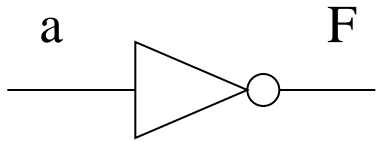


Charge/discharge in CMOS

- Calculate on resistance
- Calculate capacitance of the gates circuit is driving
- Get RC delay & use it as an estimate of circuit delay
 - $V_{out} = V_{dd} (1 - e^{-t/R_p C})$
- $R_p \sim 2R_n$



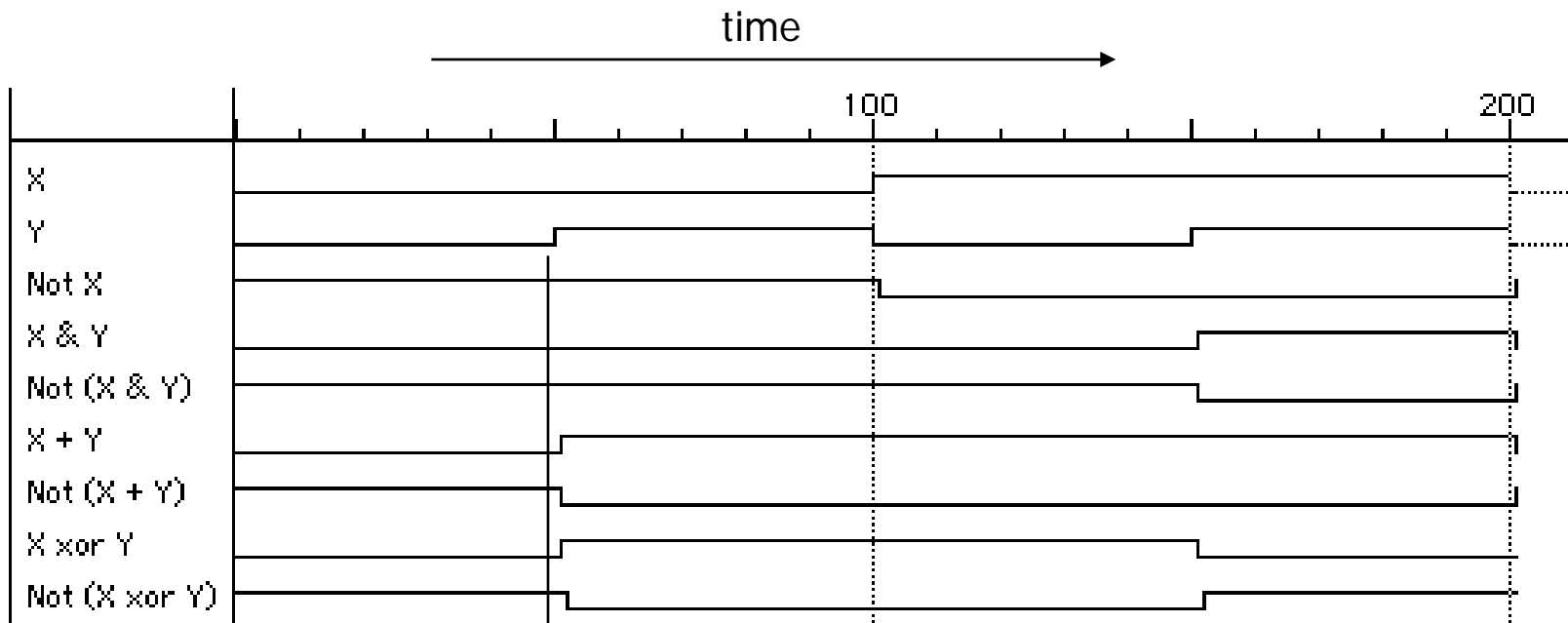
Non-Ideal Gate Behavior – Delay



- Real gates don't respond immediately to input changes
 - Rise/fall time
 - Delay
 - Pulse width

Waveform view of logic functions

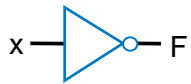
- Just a sideways truth table
 - but note how edges don't line up exactly
 - it takes time for a gate to switch its output!



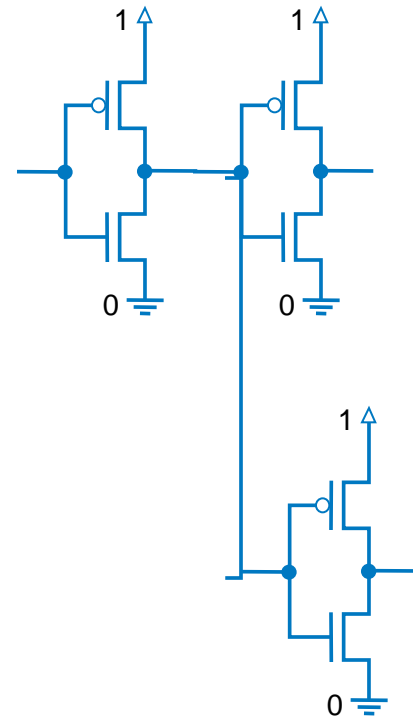
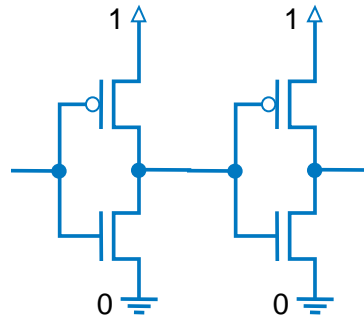
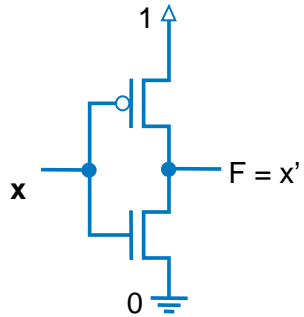
change in Y takes time to "propagate" through gates

Timing analysis: Inverter

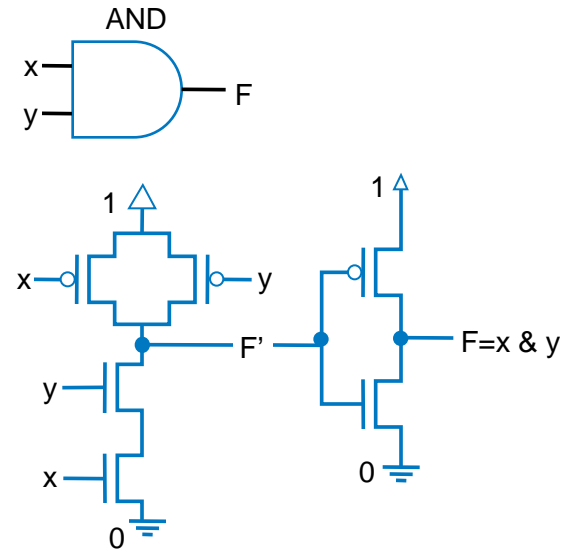
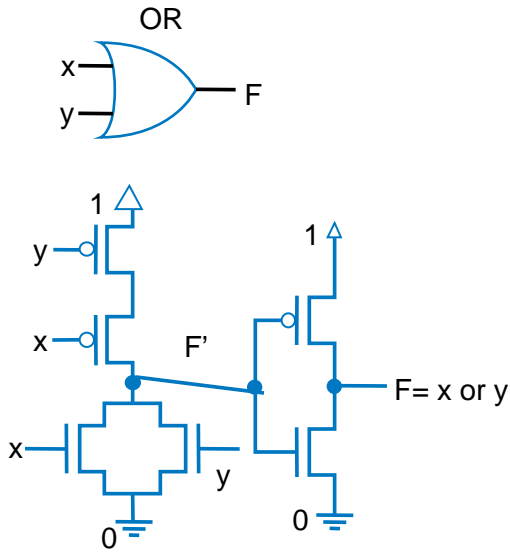
NOT



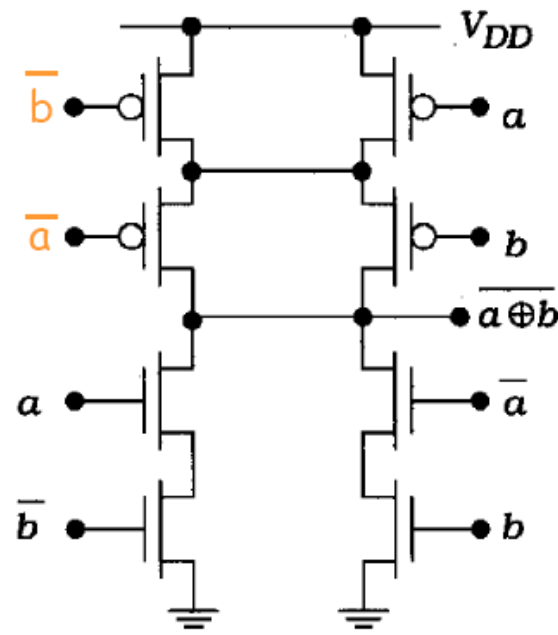
x	F
0	1
1	0



Timing analysis in gates

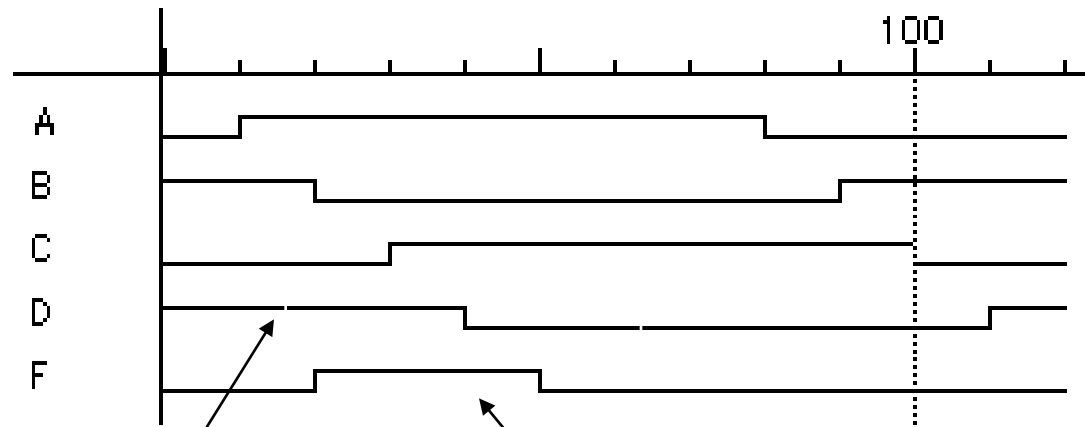
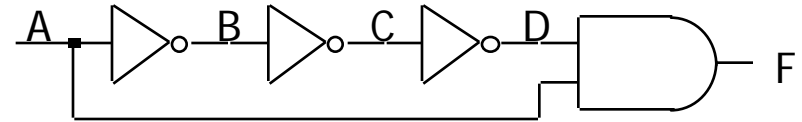


More complex gates



When is non-ideal gate behavior a good thing?

- Can be useful — pulse shaping circuits
- Can be a problem — incorrect circuit operation
- Example: pulse shaping circuit
 - $A' \cdot A = 0$
 - delays matter



D remains high for three gate delays after A changes from low to high

F is not always 0 pulse 3 gate-delays wide

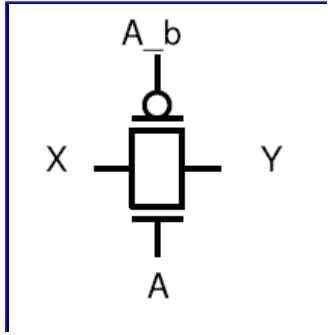


CSE140: Components and Design Techniques for Digital Systems

Muxes and demuxes

Tajana Simunic Rosing

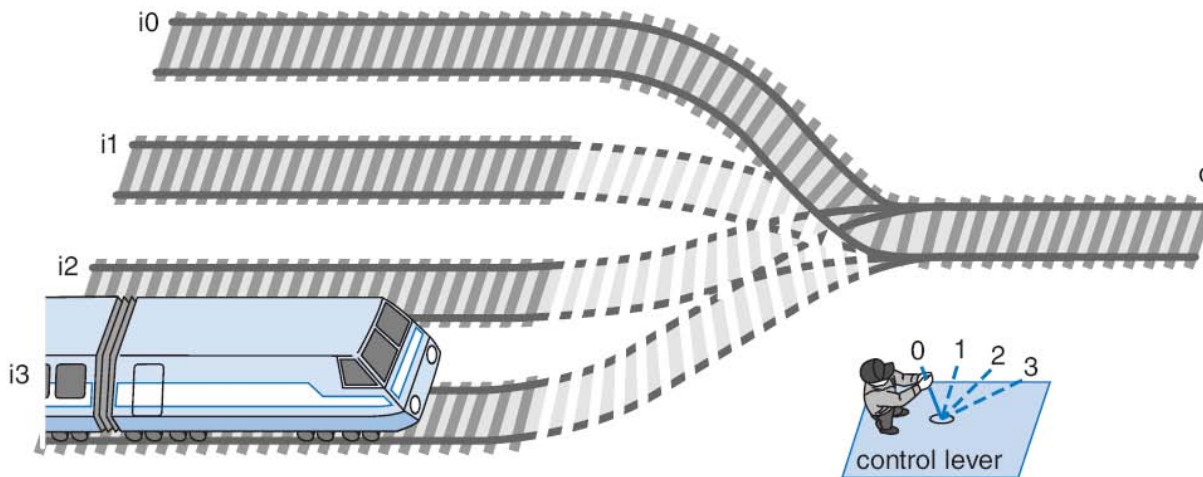
Pass transistor – Mux building block



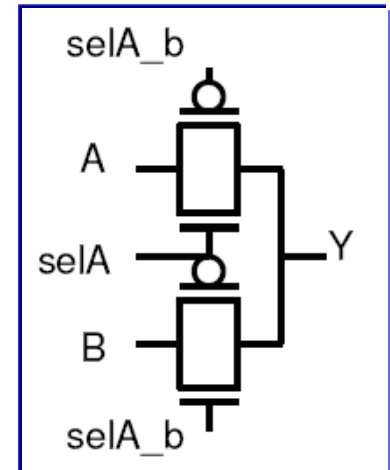
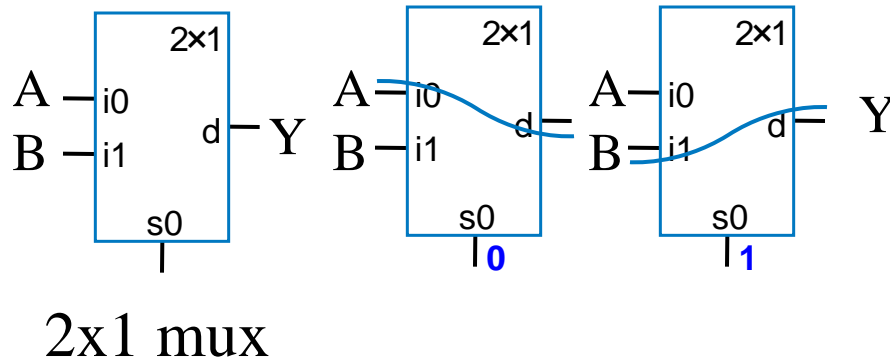
- Connects X & Y when $A=1$, else X & Y disconnected
 - $A_b = \text{not}(A)$

Multiplexor (Mux)

- Mux routes one of its N data inputs to its one output, based on binary value of select inputs
 - 4 input mux \rightarrow needs 2 select inputs to indicate which input to route through
 - 8 input mux \rightarrow 3 select inputs
 - N inputs $\rightarrow \log_2(N)$ selects



Mux Internal Design

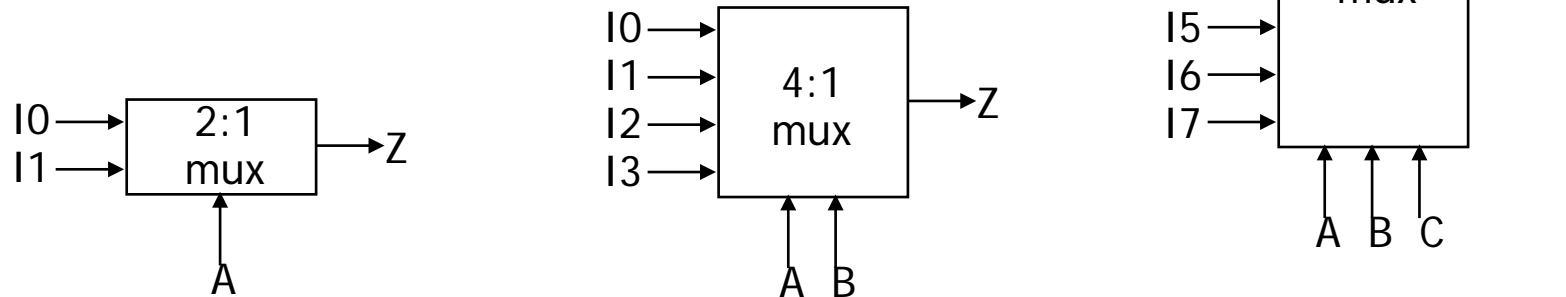


- Selects input to connect to Y
 - selA == 1: connects A to Y
 - selB == 1: connects B to Y

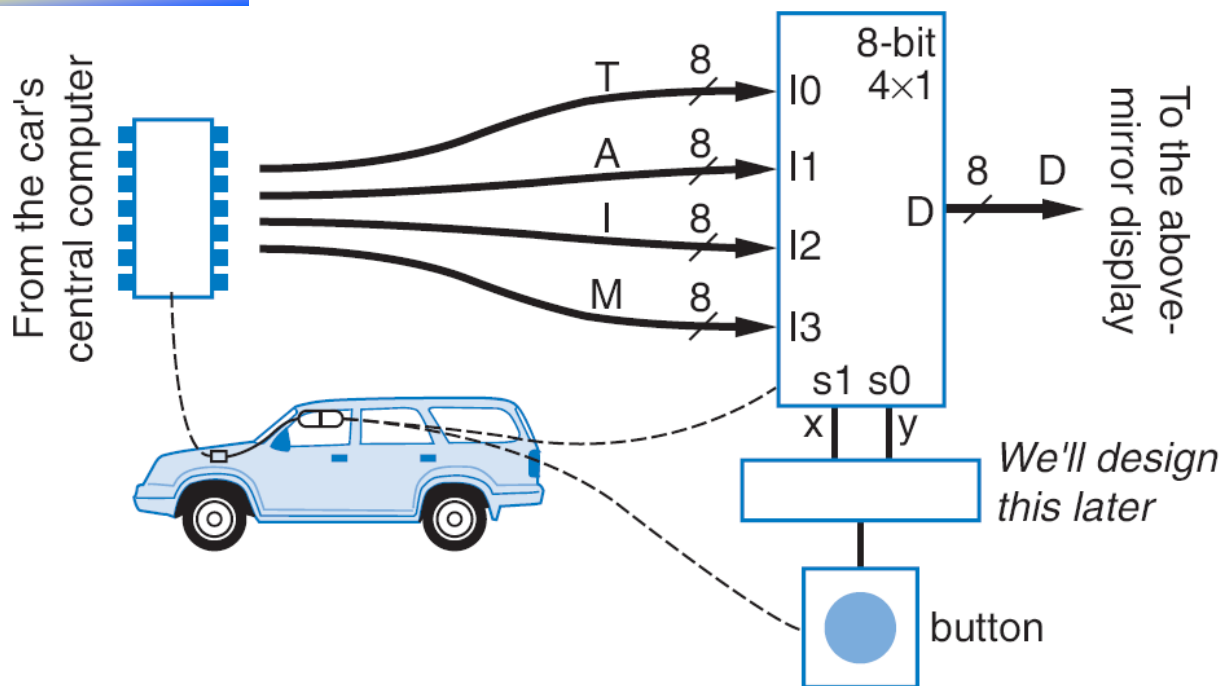
Multiplexers/selectors

- 2:1 mux: $Z = A'I_0 + AI_1$
- 4:1 mux: $Z = A'B'I_0 + A'BI_1 + AB'I_2 + ABI_3$
- 8:1 mux: $Z = A'B'C'I_0 + A'B'CI_1 + A'BC'I_2 + A'BCI_3 + AB'C'I_4 + AB'CI_5 + ABC'I_6 + ABCI_7$
- In general: $Z = \sum_{k=0}^{2^n-1} m_k I_k$

– in minterm shorthand form for a $2^n:1$ Mux



N-bit Mux Example



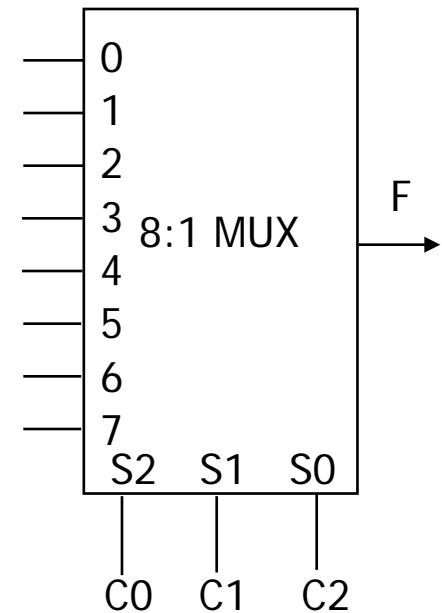
- Four possible display items
 - Temperature (T), Average miles-per-gallon (A), Instantaneous mpg (I), and Miles remaining (M) -- each is 8-bits wide
 - Choose which to display using two inputs x and y
 - Use 8-bit 4x1 mux

Multiplexers as general-purpose logic

- A $2^{n-1}:1$ multiplexer can implement any function of n variables
 - with $n-1$ variables used as control inputs and
 - the data inputs tied to the last variable or its complement
- Example: $F(A,B,C) = ABC + ABC' + A'BC + AB'C$

Mux example: Logical function unit

C0	C1	C2	Function	Comments
0	0	0	1	always 1
0	0	1	$A + B$	logical OR
0	1	0	$(A \cdot B)'$	logical NAND
0	1	1	$A \text{ xor } B$	logical xor
1	0	0	$A \text{ xnor } B$	logical xnor
1	0	1	$A \cdot B$	logical AND
1	1	0	$(A + B)'$	logical NOR
1	1	1	0	always 0



Demultiplexers/decoders

- Decoders/demultiplexers: general concept
 - single data input, n control inputs, 2^n outputs
 - control inputs (called “selects” (S)) represent binary index of output to which the input is connected
 - data input usually called “enable” (G)

1:2 Decoder:

$$O0 = G \bullet S'$$

$$O1 = G \bullet S$$

2:4 Decoder:

$$O0 = G \bullet S1' \bullet S0'$$

$$O1 = G \bullet S1' \bullet S0$$

$$O2 = G \bullet S1 \bullet S0'$$

$$O3 = G \bullet S1 \bullet S0$$

3:8 Decoder:

$$O0 = G \bullet S2' \bullet S1' \bullet S0'$$

$$O1 = G \bullet S2' \bullet S1' \bullet S0$$

$$O2 = G \bullet S2' \bullet S1 \bullet S0'$$

$$O3 = G \bullet S2' \bullet S1 \bullet S0$$

$$O4 = G \bullet S2 \bullet S1' \bullet S0'$$

$$O5 = G \bullet S2 \bullet S1' \bullet S0$$

$$O6 = G \bullet S2 \bullet S1 \bullet S0'$$

$$O7 = G \bullet S2 \bullet S1 \bullet S0$$

Gate level implementation of demultiplexers

- 1:2 decoders

1:2 Decoder:

$$O0 = G \bullet S'$$

$$O1 = G \bullet S$$

- 2:4 decoders

2:4 Decoder:

$$O0 = G \bullet S1' \bullet S0'$$

$$O1 = G \bullet S1' \bullet S0$$

$$O2 = G \bullet S1 \bullet S0'$$

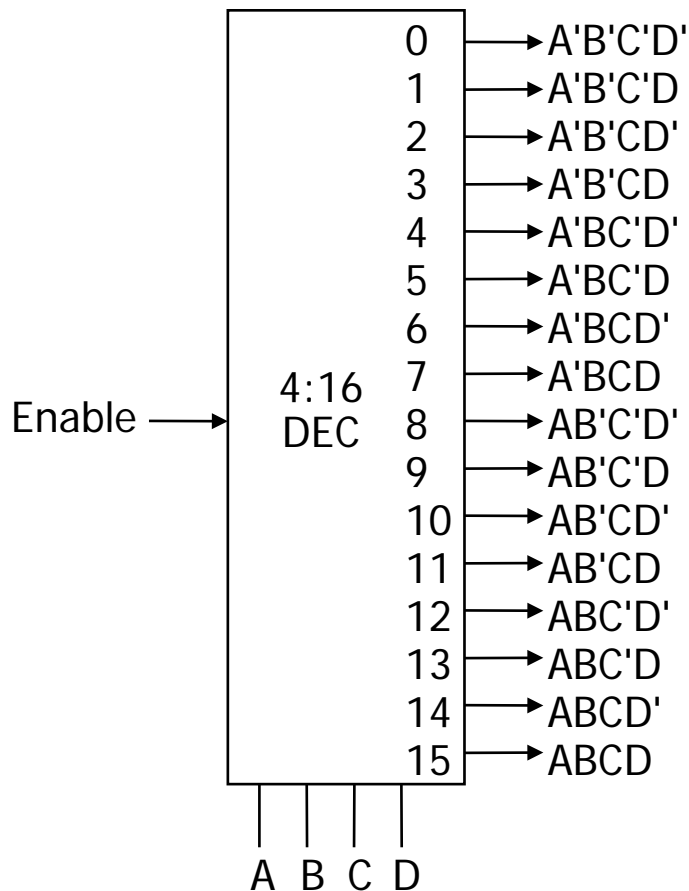
$$O3 = G \bullet S1 \bullet S0$$

Demultiplexers as general-purpose logic (cont'd)

$$F1 = A'BC'D + A'B'CD + ABCD$$

$$F2 = ABC'D' + ABC$$

$$F3 = (A' + B' + C' + D')$$



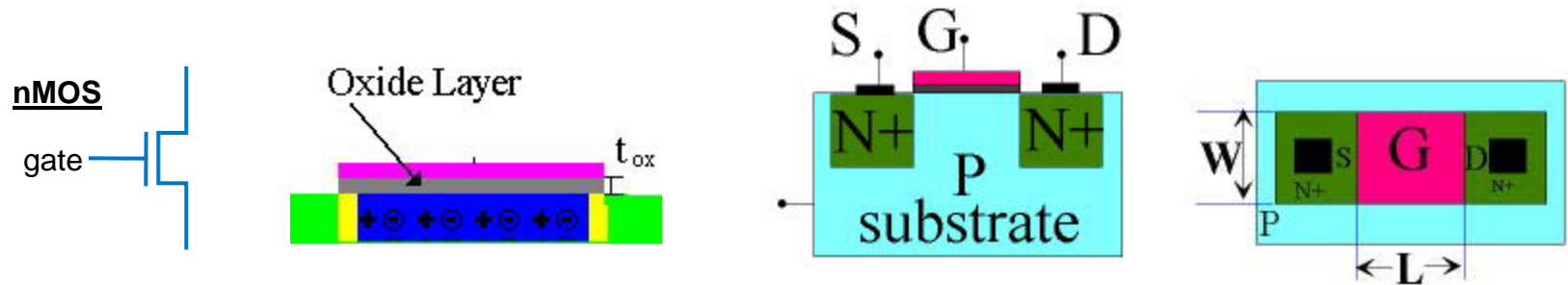
What we've covered thus far



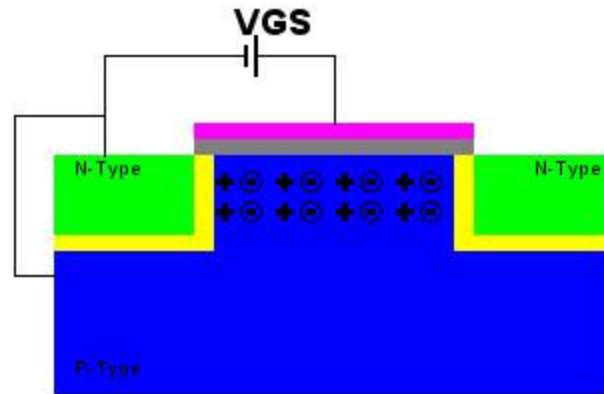
- Xilinx Virtex II Pro board and tools
- Transistor design
- Building basic gates from CMOS
- Delay estimates
- Pass transistors
- Muxes

N-MOS Tutorial – channel formation

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



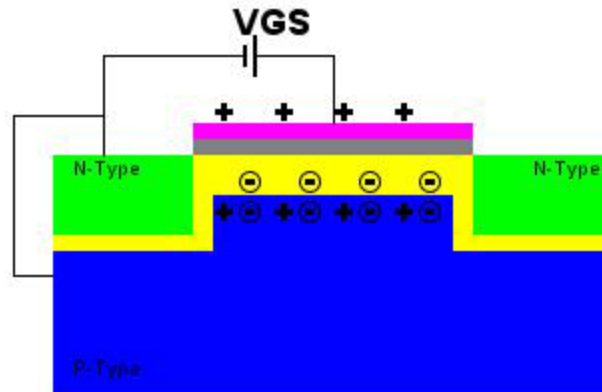
- $V_{gs} = 0$ -> lots of positive charge in p-type material, no current



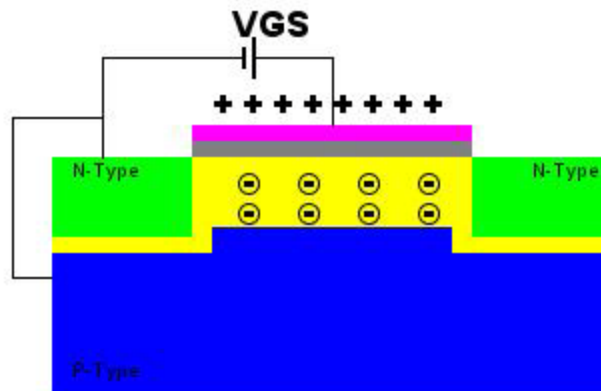
- Source: <http://www.netsoc.tcd.ie/~mcgettrs/hvmenu/tutorials/TOCcmostran.htm>

N-MOS Tutorial – channel formation (cont.)

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

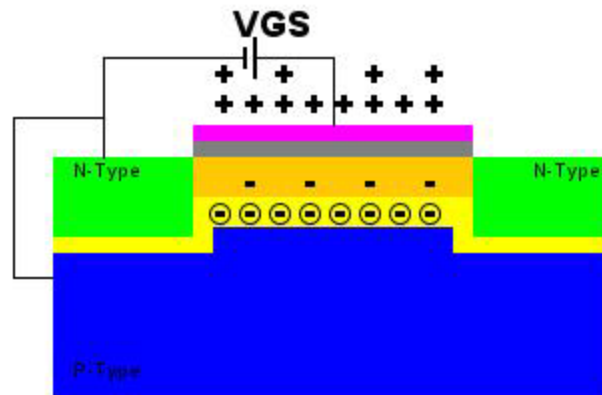


- $V_{gs} = V_t \rightarrow$ channel of negative charge forms under the oxide; the oxide is depleted of + charge; $V_t =$ threshold voltage

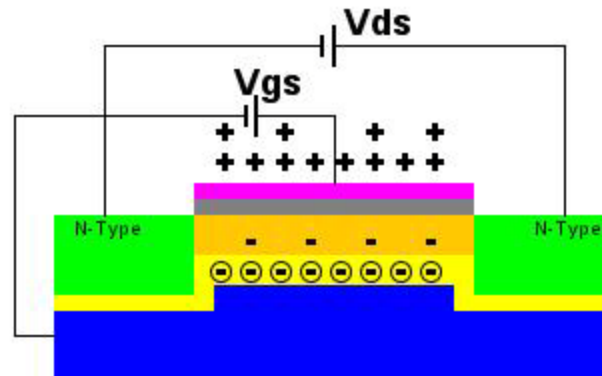


N-MOS Tutorial – channel formation (cont.)

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

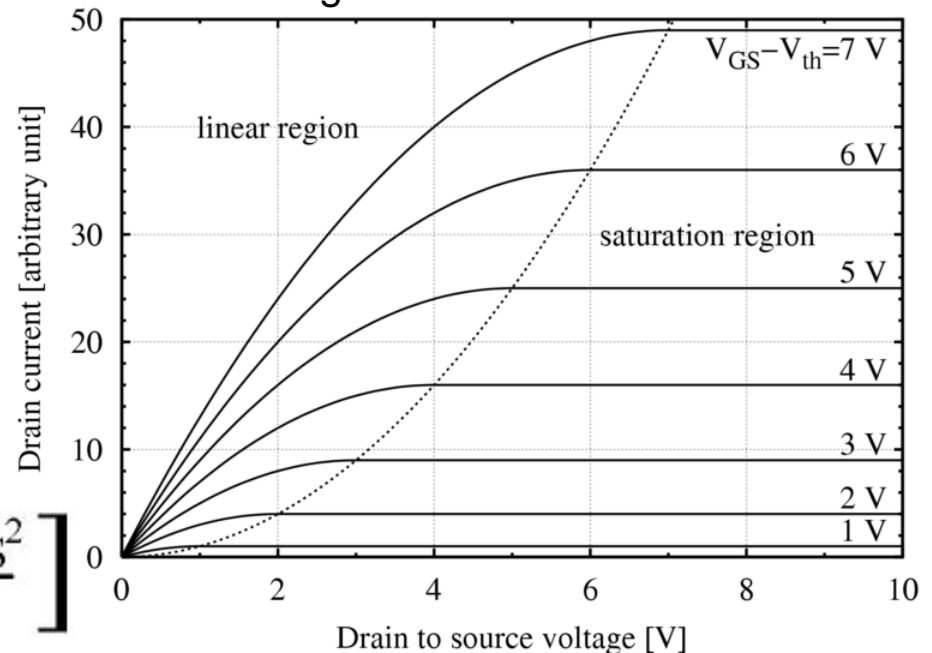
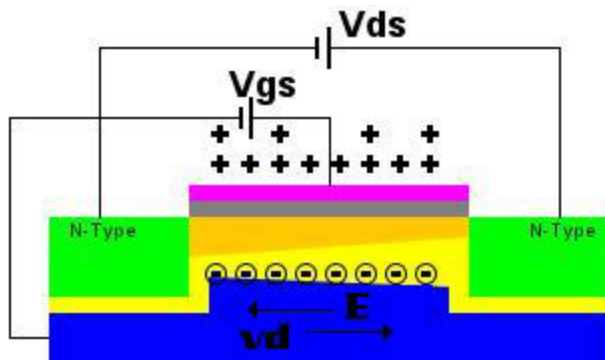


- $V_{gs} > V_t$ & $V_{ds} = 0$ -> channel present, but no current flows



N-MOS Tutorial: Current flow

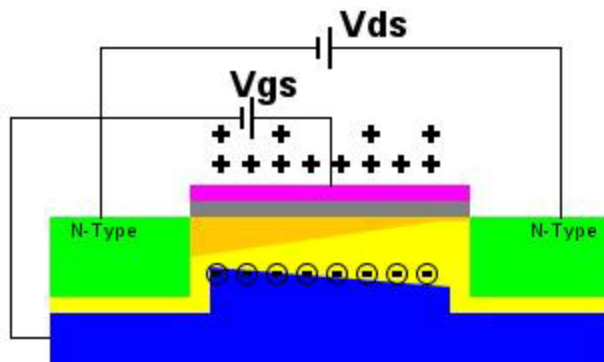
- $V_{ds} > 0$ -> electric field (E) set up between source and drain, accelerates electrons with velocity v_d , small current forms between source and drain
 - C_{ox} : oxide capacitance = ϵ_{ox} / t_{ox} (oxide permittivity ϵ_{ox} and thickness t_{ox});
 - μ : mobility of charge carriers; W/L gate width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



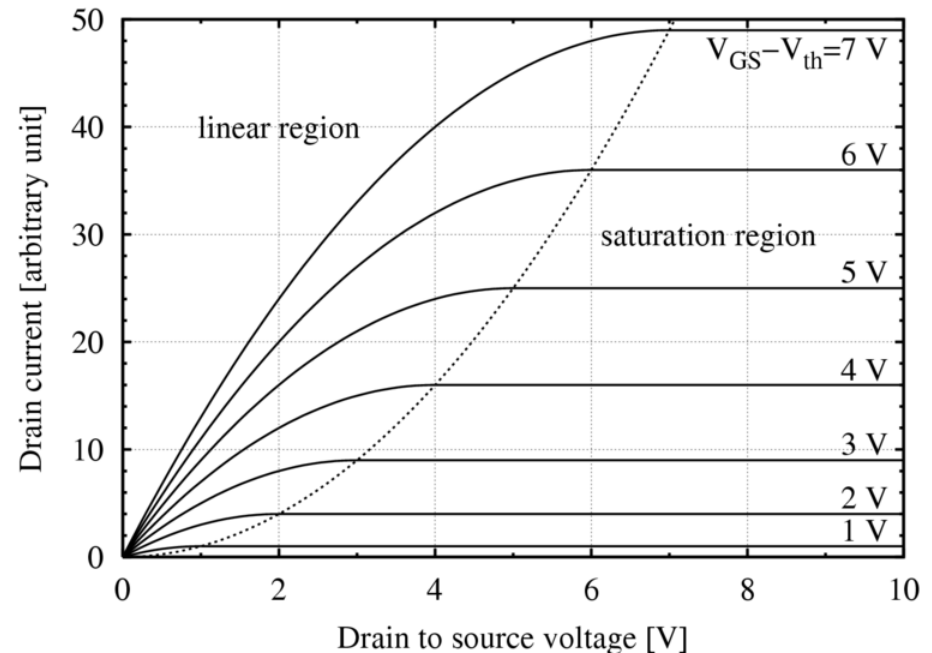
$$I_D = \mu C_{ox} \frac{W}{L} \left[(V_{GS} - V_T) V_{DS} - \frac{V_{DS}^2}{2} \right]$$

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 = ϵ_{ox} / t_{ox} (oxide permittivity ϵ_{ox} and thickness t_{ox});
 μ : mobility of charge carriers; W/L gate width and length

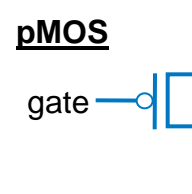
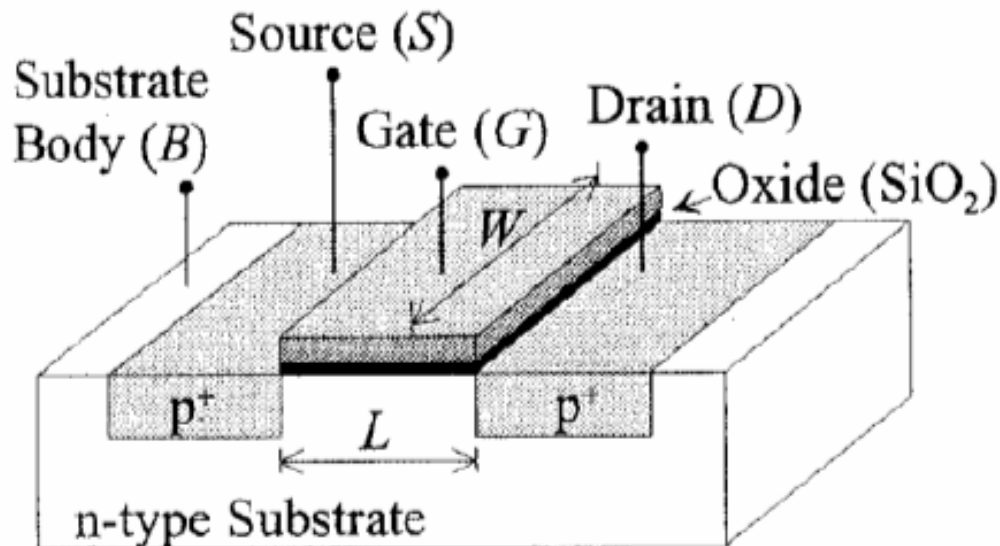


$$I_{DSat} = \frac{\mu_n C_{ox} W}{2 L} [V_{GS} - V_T]^2$$



How about P-MOS?

- Everything is the same, but polarities of voltages reverse. Mobility (μ) 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

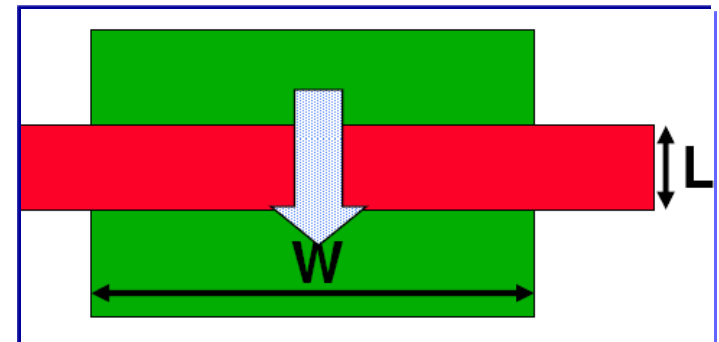
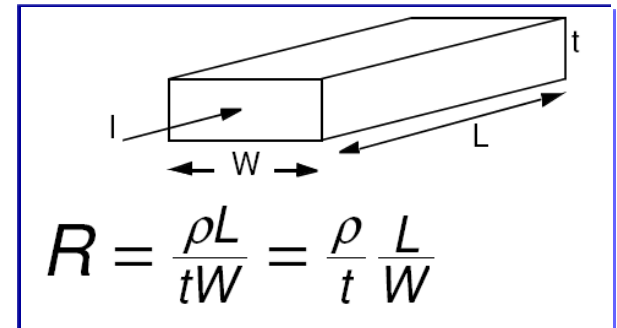
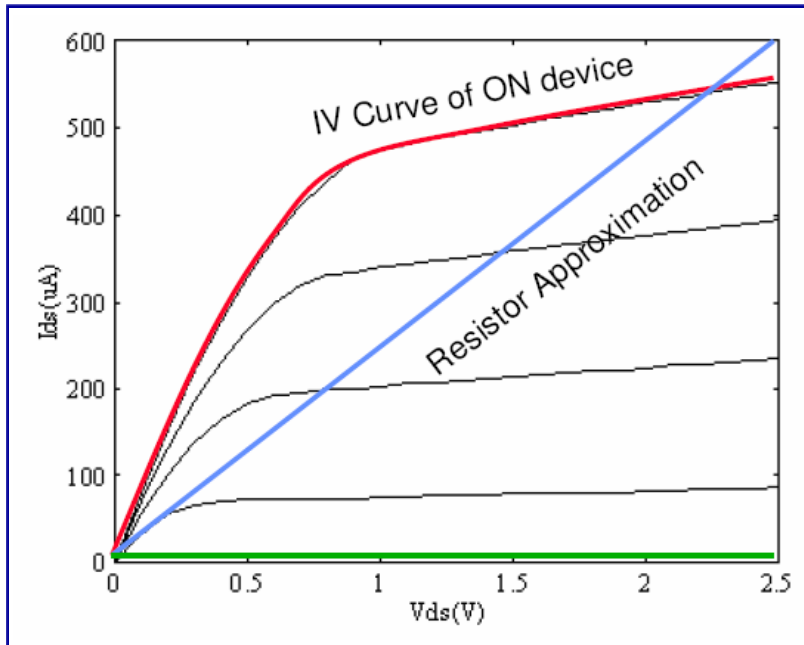
- Resistivity

- 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



Capacitance & Timing estimates

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

