Lecture 11 Chapter 10
Simulation part 2

Switch-Level Simulation Model (1)

- **Circuit model: nets interconnected by transistors.**
- **A signal is a pair (s, v):**
  - **strength(s): associated with (possibly discrete) impedance.**
  - **level(v): associated with voltage.**
    Possible values include: '0', '1' and 'X'.

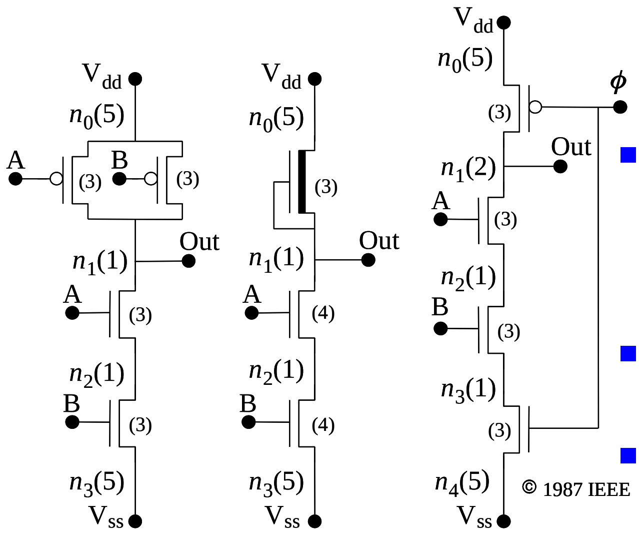
Switch-Level Simulation Model (2)

- There are two types of nets:
  - storage nets: they have a capacitance value; often the set of values is discrete.
  - input nets: they act as sources of fixed value and can supply unlimited current.
- The transistors:
  - act as bidirectional switches;
  - have a strength value (signals passing through a transistor have their strength reduced to this value).

Bryant's Model of Strength Values

- There are w distinct strength values: 1, 2, ... , k, ... , w.
- $s = w \Rightarrow s$ is the strength of an input signal.
- $k < s < w \Rightarrow s$ is the strength of a transistor.
- $1 \leq s \leq k \Rightarrow s$ is the strength of a storage net.

Strength Model Examples



- Key point is choosing strength values consistent with electrical characteristics of circuit
- Transistor limits the strength of its output signal
- Strongest signal wins

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* 5

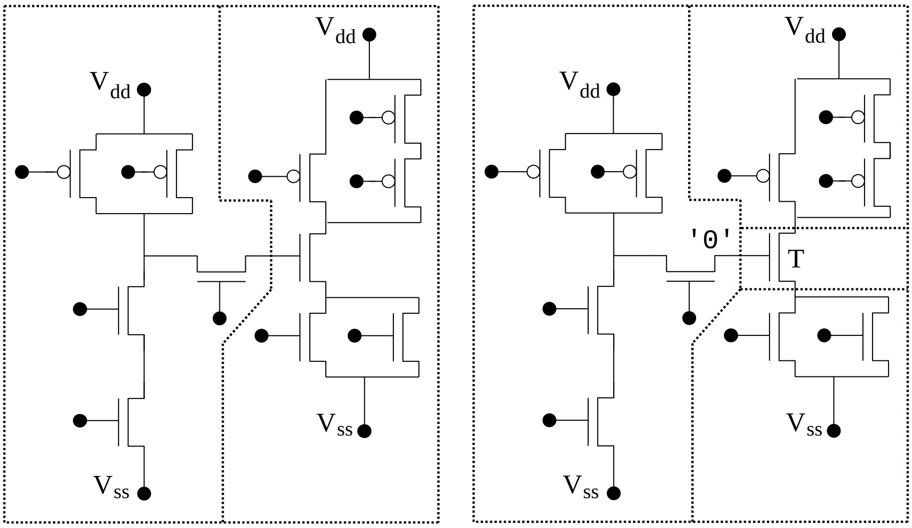
Switch-Level Simulation Techniques

Main principles:

- **Partition the circuit into unidirectional subcircuits**
  - **Channel-connected components are bi-directional**
  - **Gates bound uni-directional elements**
  - **Off-state transistors (and input nets) can also bound uni-directional elements**
  - **Interaction between these subcircuits can be handled similar to gate-level simulation.**
- **Two types of partitioning exist: static and dynamic (= accounting for signal values).**
- **Apply special methods to compute the “steady-state” of the channel-connected components.**

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* 6

Circuit Partitioning Example

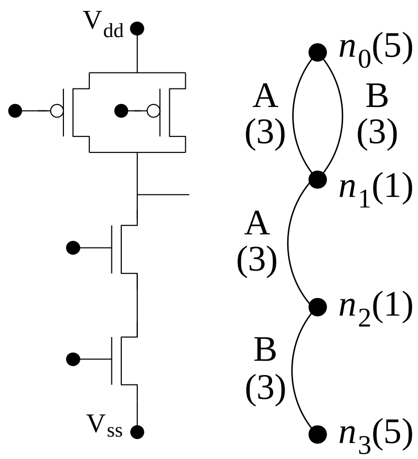


Static partitioning

Dynamic partitioning

Circuit Representation

- A convenient representation for switch-level circuits is a multigraph G(V, E) rather than the more general cell-port-net model.
- Vertices represent nets and are labeled with the net name and strength.
- Edges represent transistors and are labeled with a transistor ID and strength.



Signals and Signal Propagation (1)

- A signal on a vertex $u \in V$ is denoted by (σ_u, λ_u).
- The strength of a transistor $(u, v) \in E$ is given by $\varepsilon_{u,v}$.
  $\varepsilon_{u,v} = 0$ when the transistor is off.
- $\sigma_{u \to v}$ denotes the strength of the signal flowing from $u \in V$ to $v \in V$.

$$\sigma_{u \to v} = \min(\sigma_u, \varepsilon_{u,v})$$

- The level of the signal flowing remains λ_u.
- There are two types of nets:
  - driven nets: nets having a conducting path to an input net.
  - charged nets: nets electrically isolated from input nets.

Signals and Signal Propagation (2)

- Suppose that a driven net $v \in V$ has edges $(u_1,v), \ldots, (u_m,v) \in V$, then:

$$\sigma_v = \max_{1 \leq i \leq m} \sigma_{u_i \to v} \qquad \text{Strength of signal on net v}$$

- For a charged net, the net's own signal should be taken into account:

$$\sigma_v = \max(\sigma_v, \max_{1 \leq i \leq m} \sigma_{u_i \to v})$$

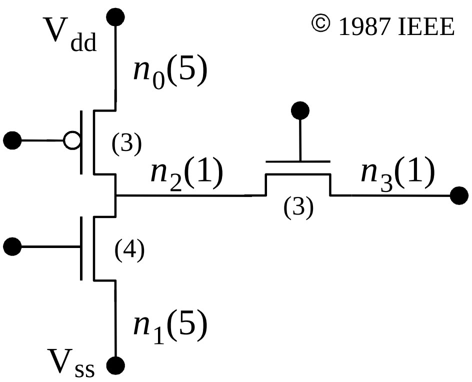
- When combining signals from different directions, the level of the new signal equals the level of the strongest signals. In case of multiple signals with equal strength and different levels, the new level becomes 'X'.

Simulation Algorithm Principles

- **The algorithm is based on a repeated application of:**

  $\sigma_v = \max(\sigma_v, \sigma_{u \rightarrow v})$

- **This should be done carefully: propagate the strongest signals first.**
- **Implement with an array of queues, one array position for each strength value.**



ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* *11*

Simulation Algorithm: Discussion

- **The algorithm operates in linear time with respect to the number of nets and transistors.**
- **The algorithm is static: changes to input signals require repeating the complete propagation. It can, however, be modified for dynamic simulation.**
- **This algorithm does not incorporate any type of delays related to the physical implementation. Switch-level timing simulation can deal with actual R and C values derived from the layout.**

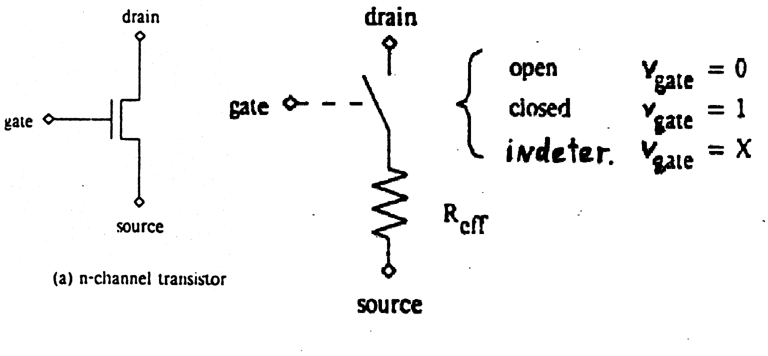
ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* *12*

Simulation Conclusion

Gate level	huge circuits, not appropriate for MOS
Discrete switch level	appropriate for MOS, no timing information
Linear switch level	appropriate for MOS, timing estimation
Timing simulation	accurate timing, circuit size limited
Circuit simulation	most accurate, “small” circuits
Cycle based simulation	fast, no timing, synchronic circuits

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs 10/7/2010 13

RSIM Linear Switch Level Model



(b) RSIM model

	R_{ds}	
V_{gate}	N	P
0	∞	R_{eff}
1	R_{eff}	∞
X	$[R_{eff} - \infty]$	$[R_{eff} - \infty]$

Interval Arithmetic for X-states

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs 10/7/2010 14

Linear Switch Level Model

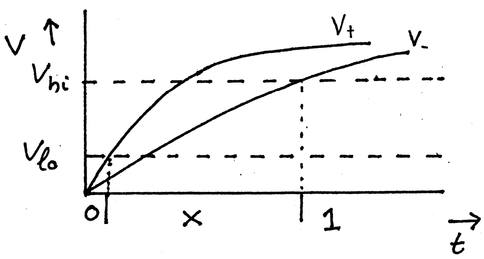
R_{eff} depends on:

- **Transistor dimension**
- **Transistor type**
- **Transistor context**
  pull-down, pass, …

$$R_{eff} = f(type, context) \times \frac{length}{width}$$

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* *15*

Logic Level



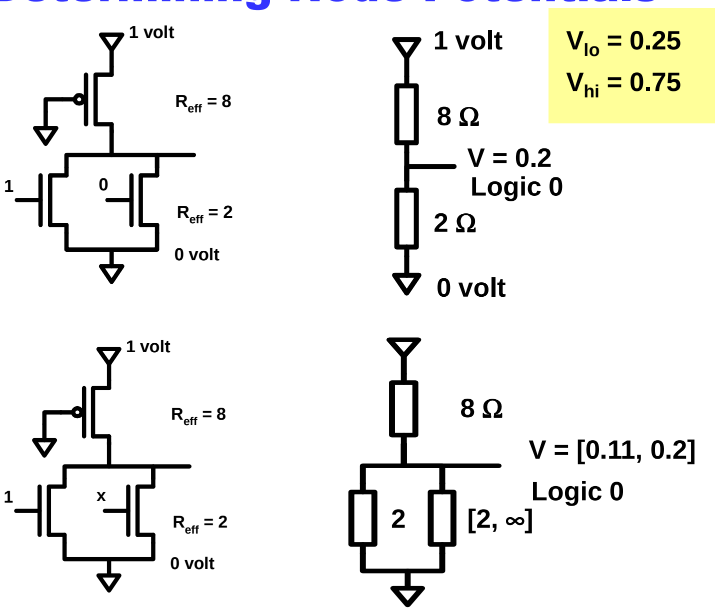
$V = [V_-, V_+]$ ⟺

Logic States

0	$V_+ \leq V_{lo}$
1	$V_- \geq V_{hi}$
X	**otherwise**

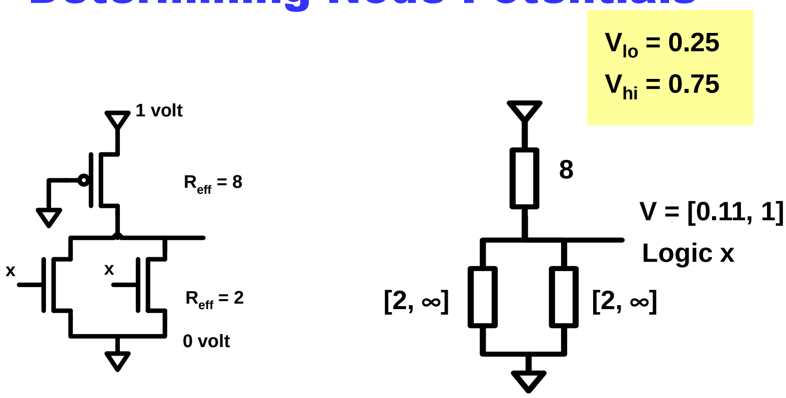
ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* *16*

Determining Node Potentials



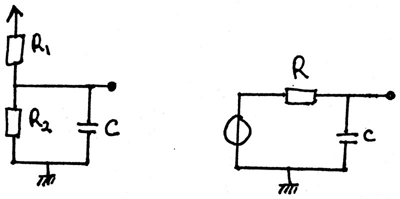
ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* *17*

Determining Node Potentials



ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs *10/7/2010* *18*

Determining Transition Time



Calibration

R_{static}	Final node potential
R_{dynlow}	Used for RC time high → low
$R_{dynhigh}$	Used for RC time low → high

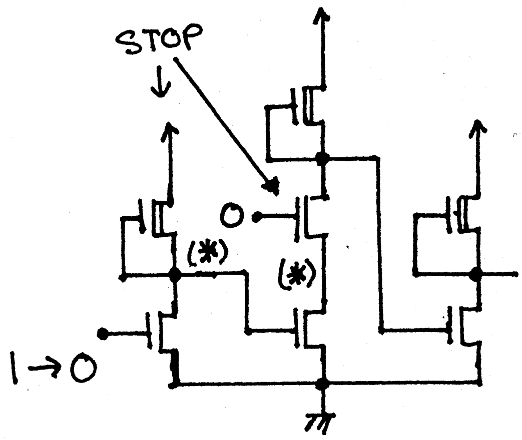
RSIM Simulation Algorithm

Event Driven

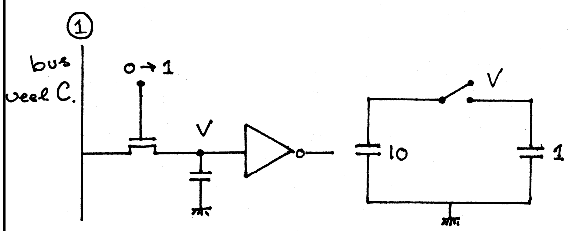
- Event = (net, new logic value, time)
- Algorithm: While event list $\neq \emptyset$
  1. take event from list
  2. set node on new value
  3. determine effect on other nodes
     - Limited number of nodes (stage, vicinity)
     - Determine charge-sharing (immediately)
     - Determine final value (RC time)

RSIM Simulation Algorithm

- (*) = nodes which are influenced



Charge Sharing



Logic States	
0	$V_+ \leq V_{lo}$
1	$V_- \geq V_{hi}$
X	otherwise

$$V_- = \frac{\text{total capacitance on logic level 1}}{\text{total capacitance}}$$

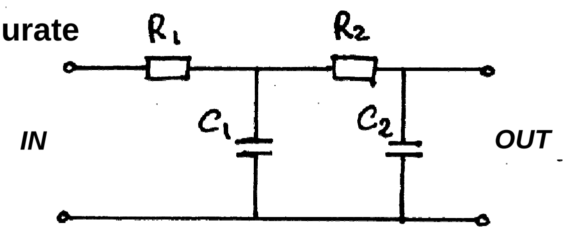
$$V_+ = \frac{\text{total capacitance on logic level 1 or x}}{\text{total capacitance}}$$

Voltage on V	
Before switched on	After switched on
1	1
0	10/11
X	[10/11, 1]

RC Time

R: R_{dynhi} or R_{dynlo}
C: total cap. on logic level 0 or x if final state 1
total cap. on logic level 1 or x if final state 0

RC time inaccurate



Better:

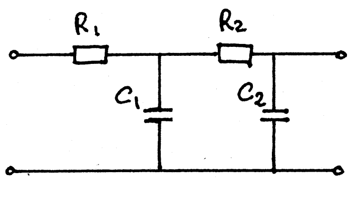
- $R_1C_1 + (R_1 + R_2)C_2$ (Elmore Delay)

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs 10/7/2010 23

Switch Level Simulation Conclusion

RSIM

- Linear switch level (switch level timing)
- Interval arithmetic
- Conversion to logic states
- Lumped (concentrated) capacitances

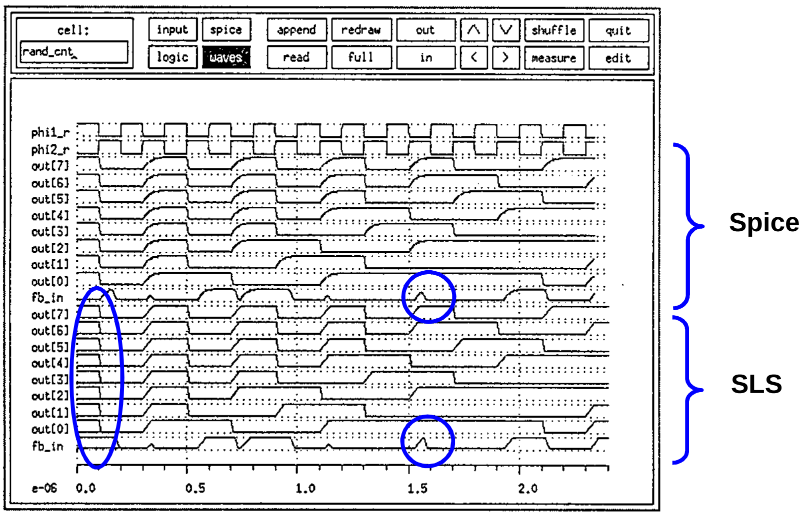


SLS

- Linear switch level
- Interval arithmetic (consistent)
- Conversion to logic states only for transistor state
- Elmore delay

ET 4255 - Electronic Design Automation 10/11 © Nick van der Meijs 10/7/2010 24

Switch Level Timing (SLS) vs SPICE



SPICE: 39 min 41 sec SLS: 2.4 sec (HP9000840, ~anno 1990)