

Logic Simulation

- Purpose
 - ◆ Design Verification
 - ◆ Performance Evaluation
 - ◆ Evaluation of Alternative Designs
 - ◆ Debugging
 - ▲ observe internal signals
 - ▲ can change clock rate, gate delays
 - ▲ can start the simulation in any desired state
 - ▲ can be interfaced to partial designs

Outline

- Types of Simulation
- Logic Simulation in Presence of Unknowns
- Simulation of Synchronous Sequential Circuits
- Gate Evaluation Methods
- Event-Driven Simulation
- Hazard Detection

Types of Simulation

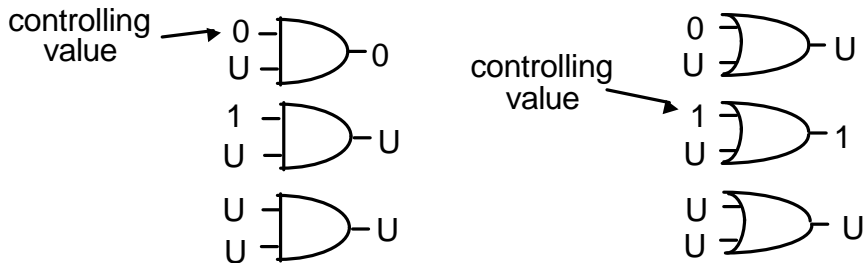
- Compiled Code
 - ◆ functional verification, timing not incorporated
- Table Driven
 - ◆ primitives are evaluated using tables
- Event Driven (activity-directed)
 - ◆ 1% - 10% of lines are active in a simulation
- Time Driven / Cycle Based

Event: change in signal line --> “active line”
Evaluate gates only when inputs change.
Level of simulation depends on level of the model.

Unknown Value

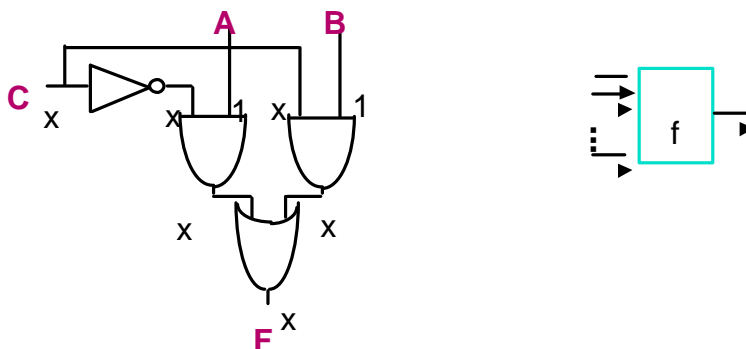
- Unlike real circuits, which have only two signal values (0, 1), a simulator cannot restrict itself to 0, 1. Why? (0, 1, U, Z, Rising, Falling)
- At the start, flip-flops are in an “unknown” state.
- Formally, unknown is a set of two values {0,1}
 - ◆ Boolean Operation op
 - ◆ $0 \text{ op } U = \{0\} \text{ op } \{0,1\} = \{0 \text{ op } 0, 0 \text{ op } 1\}$
 - ◆ $op = \text{AND}$
 - ▲ $0 \text{ AND } U = \{0 \text{ AND } 0, 0 \text{ AND } 1\} = \{0,0\} = \{0\} = 0$
 - ▲ $1 \text{ AND } U = \{1 \text{ AND } 0, 1 \text{ AND } 1\} = \{0,1\} = U$

Unknown Value



- A subtle difference between unknown U and don't care X
- In Boolean algebraic operations, they are the same, but when differences do exist, care must be taken.

Logic Simulation in Presence of Unknowns



- Use high-level Boolean functions of modules.

Coding for Three Values

- Values: 0, 1, U
- An encoding: 00, 11, 01

AND	0	1	x
0	0	0	0
1	0	1	x
x	0	x	x

AND	00	11	01
00	00	00	00
11	00	11	01
01	00	01	01

bit-wise AND
on the codes

$$\text{AND } \begin{array}{l} 0: 00 \\ 1: 11 \\ 0: 00 \end{array}$$

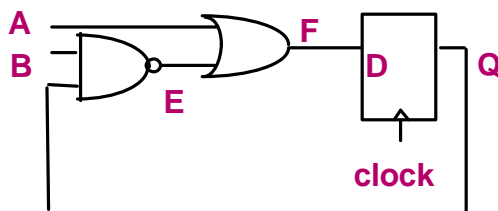
$$\text{AND } \begin{array}{l} U: 01 \\ 1: 11 \\ U: 01 \end{array}$$

$$\text{OR } \begin{array}{l} 0: 00 \\ 1: 11 \\ 1: 11 \end{array}$$

$$\text{OR } \begin{array}{l} U: 01 \\ 1: 11 \\ 1: 11 \end{array}$$

NOT 00 $\rightarrow \bar{0}\bar{0} \rightarrow 11$
 NOT U $\rightarrow \bar{0}\bar{1} \rightarrow 10 \xrightarrow{\text{swap}} 01$

Simulation of Synchronous Sequential Circuits



```

Procedure CKT1
inputs A, B; static Q;
begin
  E = B NAND Q
  F = A OR B
  Q = F
end
    
```

Every call advances
the clock by 1.

Assume data is stable at inputs to
flip-flops when clock signal arrives

Gate Evaluation Methods

Controlling value and **Inversion** value:

- Controlling value controls the output of the gate
- If there is a controlling input, output of gate is $c \oplus i$
otherwise $c' \oplus i$

	c	i
AND	0	0
OR	1	0
NAND	0	1
NOR	1	1

Gate Evaluation Methods

- Input scanning
 - ◆ Look for controlling value c
 - ◆ If found then output = c
 - ◆ else if unknown found then output = X
 - ◆ else output = c
- Count based
 - ◆ C_0 = count of 0's
 - ◆ C_x = count of x's
 - ◆ if $C_0 > 0$ then output 0
 - ◆ else if $C_x > 0$ then output x
 - ◆ else output 1
- Table based -- indexed look-up

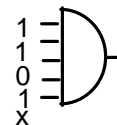
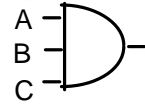


Table-Based Indexed Look-up



- $\{0, 1, x\}^3 = 27$ possible vectors
- $2^6 = 64$ entries in table, since 0,1,x require 2 bits

Index I =

	0			
--	---	--	--	--

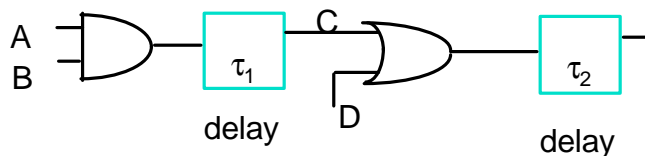
A B C

A	B	C	D[I]
00	00	00	0
00	00	01	0
...			
11	11	11	1

Of 64 entries, 27 are useful

zoom tables: gate type is part of index

Event-Driven Simulation



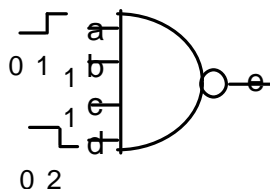
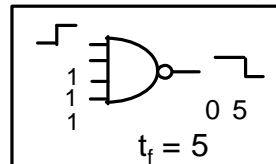
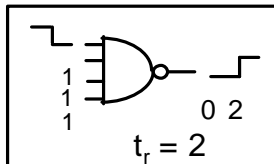
- All $\tau = 0$: zero delay simulation
 - ◆ **easiest, fastest**
- All $\tau = 1$: unit delay simulation
 - ◆ **moderate**
- Different τ 's: variable delay, assignable delay simulation
 - ◆ **hardest, slowest**

Event-Driven Simulation

- Different gates may have different delays
 - ◆ In the project you will implement a zero-delay simulator
- **Logic Simulation Algorithm with Delay**
 - Two-pass strategy when delay is not zero:
 - ◆ Need to schedule events, that is, changes of logic values
 - ◆ Need to keep a queue of activated gates

Observation on Logic Simulation Algorithm with Delay

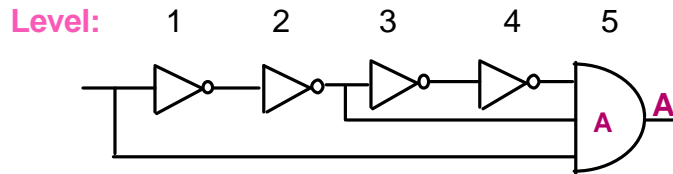
- Correct only if an event at time $t+k$ does not schedule an event earlier than an already scheduled event for the same gate.
- Example:



event list

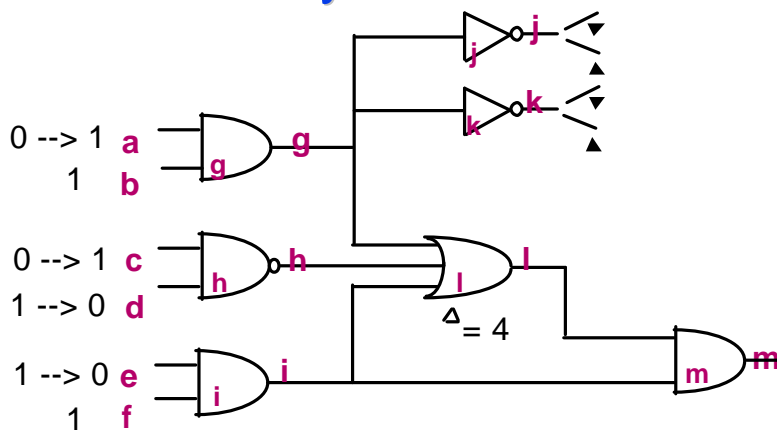
- 0: (a,0)
 - 1: (a,1)
 - 6: (e,0)
 - 2: (d,0)
 - 4: (e,1)
- Need to cancel 6: (e,0)

Zero-Delay vs. Unit-Delay Simulation



- In unit-delay simulation, gate A is evaluated 3 times
- In zero-delay simulation, gate A is evaluated once
 - ◆ zero-delay gives steady-state values of nodes

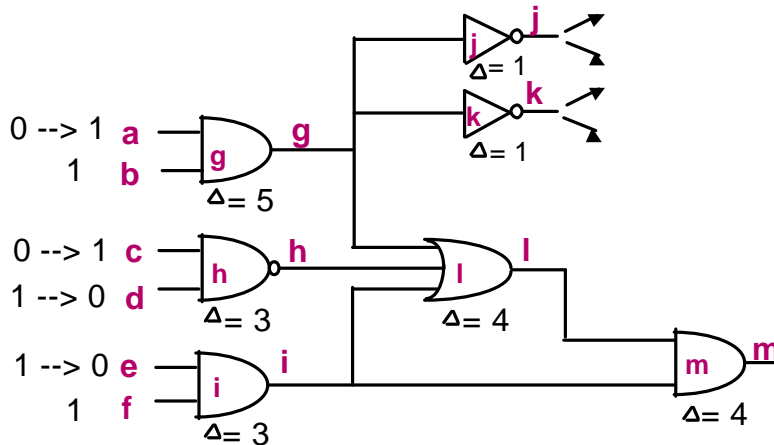
Zero-Delay Simulation Example



Level 0 Level 1 Level 2 Level 3

conceptually, one event queue per level

Assigned-Delay Simulation Example



ECE 1767

University of Toronto

Assigned-Delay Simulation Example

Time t_1

Pass 1: $a \neq a' \rightarrow (a'1)$ is an event, add g to Activated
 $c \neq c' \rightarrow (c'1)$ is an event, add h to Activated
 $d \neq d' \rightarrow (d'0)$ is an event, add h to Activated
 $e \neq e' \rightarrow (e'1)$ is an event, add i to Activated

Pass 2: $g' = 1$ at time $1+5 = 6$
 $h' = 1$ at time $1+3 = 4$
 $i' = 0$ at time $1+3 = 4$

Time t_4

Pass 1: $h' = h$, so no event
 $i' \neq i \rightarrow (i'0)$ is an event, add l, m to Activated

Pass 2: $l' = 1$ at time $4+4 = 8$
 $m' = 0$ at time $4+4 = 8$

ECE 1767

University of Toronto

Assigned-Delay Simulation Example

Event List

t1: (a',1), (c',1), (d',0), (e',0)

t6: (g',1)

t4: (h',1), (i',0) ← Must schedule since there may be
 (h,0) scheduled before time t4

t8: (l',1), (m',0)

t7: (j',0), (k',0)

t10: (l',1)

Time Event

t6: (g',1) means at time t6, g will take on value 1.

Unit-Delay Simulation Example

