Simulation Timing Models (1A)

- Gate-level Timing Model
- Procedural Timing Model

- Gate-level Models and Timing
- Dataflow Models and Timing
- Behavioral Models and Timing

Young Won Lim
05/14/2013

Copyright (c) 2012 Young W. Lim.

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.2 or any later version published by the Free Software Foundation; with no Invariant Sections, no Front-Cover Texts, and no Back-Cover Texts. A copy of the license is included in the section entitled "GNU Free Documentation License".

Please send corrections (or suggestions) to youngwlim@hotmail.com.

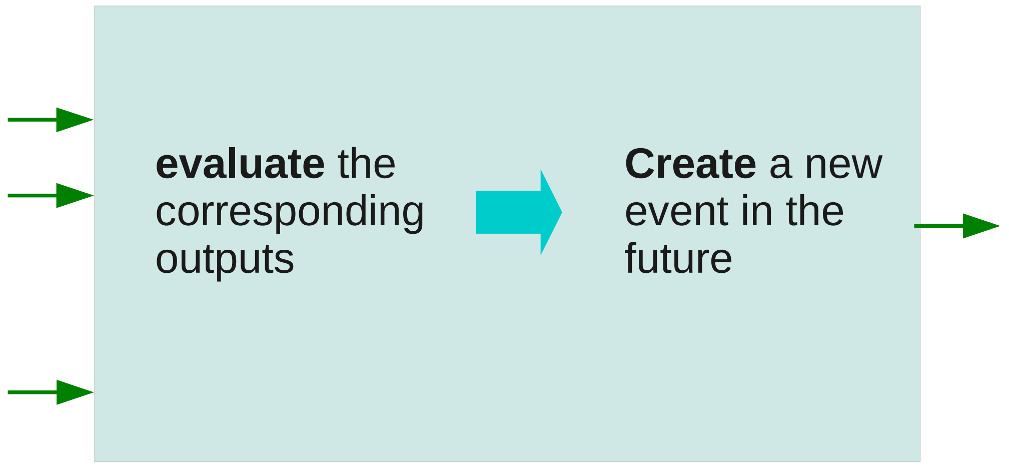
This document was produced by using OpenOffice and Octave.

Gate-level Timing Model

```
nor #2
        (z, a, b),
        (x, c, d);
```

sensitive to <u>all</u> <u>inputs</u>

<u>any</u> input can change at <u>any</u> time


evaluate the corresponding outputs
Create a new event in the future

The <u>*previously*</u> scheduled event is always *canceled* by a <u>*new*</u> event

A pulse shorter than the propagation delay will not affect the output

An **inertial delay** : min time for a set of inputs must hold to affect the output

must be greater than propagation delay

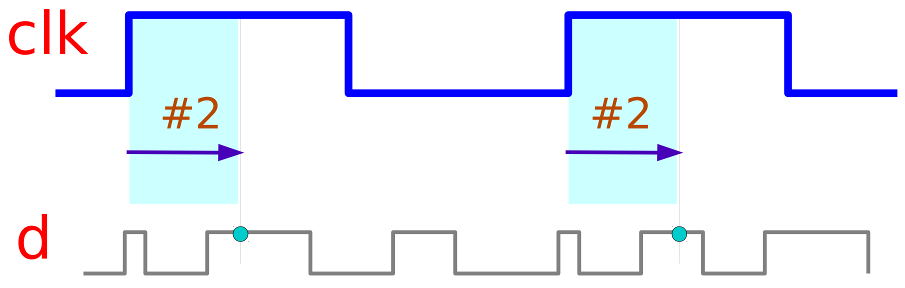
- Gate primitives
- User defined primitives
- Continuous assignment statements
- Procedural continuous assignments

Procedural Timing Model

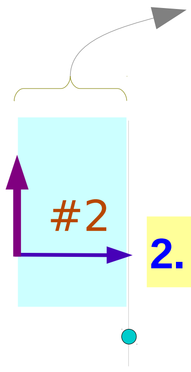
```
always @ (posedge clk)
        #2 q = d;
```

~~Any input changes at any time~~

the two inputs (clk, d)



During this 2 unit delay, any input changes will have no effect, including another posedge clk



1. posedge clk

2. two unit delay

3. fetch input

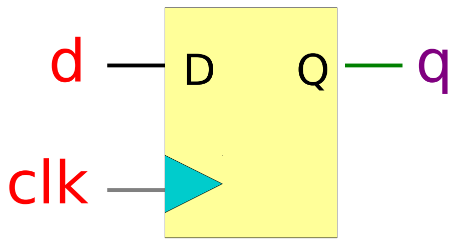
Only *sensitive* to positive clk edges *when* execution of the model is stopped at the "@"

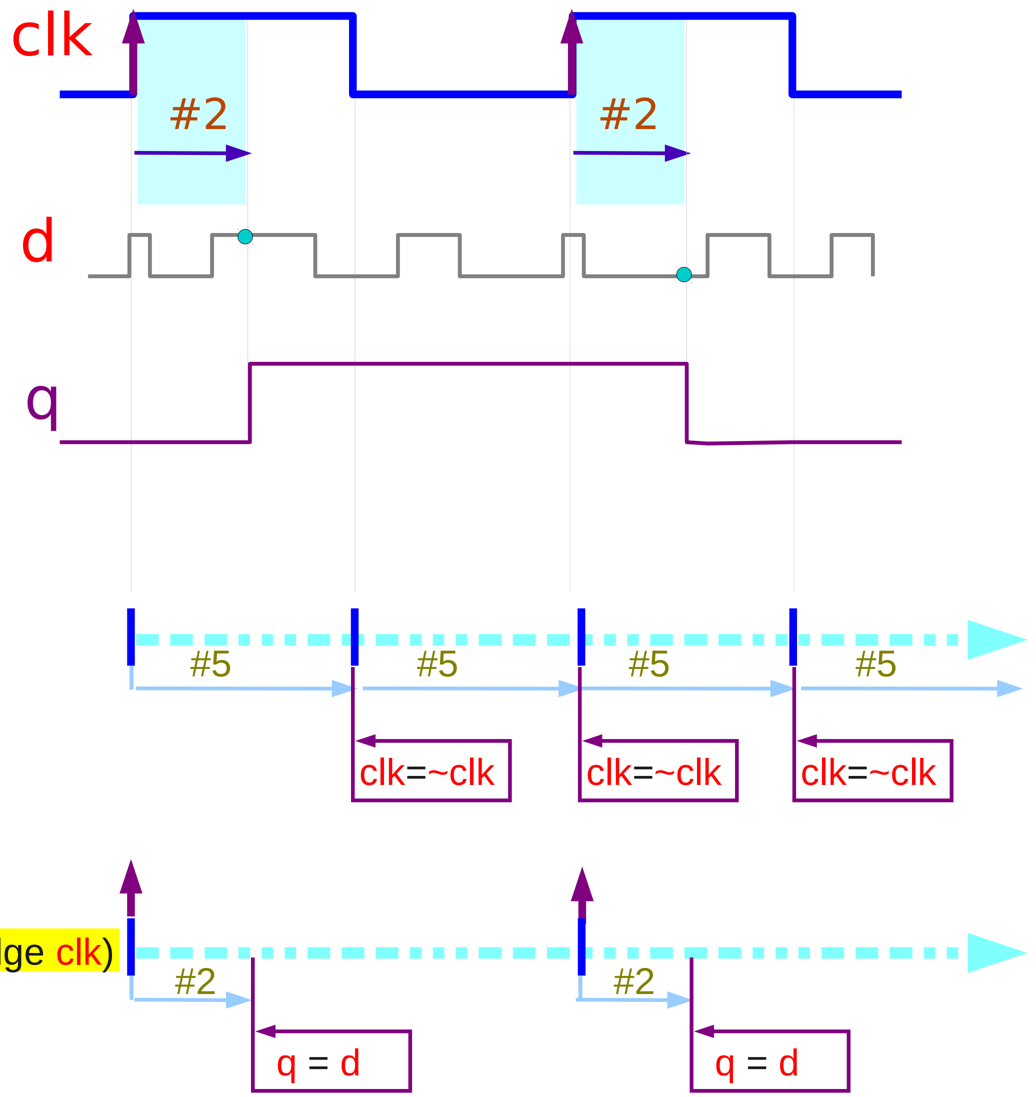
Only sensitive to a subset of of their inputs – sensitivity list

This sensitivity changes over time with the execution of the model

The previously scheduled events will not be canceled – multiple events: indeterminate execution

Behavioral Modeling – Sequential





```
always
        #5 clk = ~clk;
```

```
always @(posedge clk)
        #2 q = d;
```

Procedural Timing Model – simulating gate

```
nor #2
        (z, a, b);
```

```
always @ (a, b)
        #2 z = ~(a | b);
```

In the gate level timing model, the previously scheduled event will be canceled by a new event

Procedural assignment makes the behavioral model insensitive to the inputs during the propagation delay of the gate

Sequential Assignment (2)

References

[1] http://en.wikipedia.org/
[2] T.R. Padmanabhan, B.T. Sundari, “Design Through Verilog HDL
[3] D.E. Thomas, P.R. Moorby, “The Verilog Hardware Description Language”, 3rd ed