Addendum to Lecture Notes #1

CAD Based Logic Design

ECE 368
Discrete Event Simulation — Event Queue for Simulation w/ Delays

**Statements:**

- \( z + 1 = 1 \)
- \( x = 1 \)
- \( x = 0 \) or \( y \) after \( 2 \) ns

At time \( t+2 \): disp \( x=1 \) & execute \( Z \leq x \) or \( y \) after \( 2 \) ns;

Result −> 1 event: \( Z: 0 \rightarrow 1 \) @ \( t+4 \) ns;

At time \( t+4 \): disp \( Z=1 \), no statement execution

**DES with Delays**

\( x \) or \( y \) after \( 2 \) ns

\( z + 1 = 1 \)

**DES with Delays**

\( x \) or \( y \) after \( 2 \) ns

\( z + 1 = 1 \)

**Statements:**

- \( z + 1 = 1 \)
- \( x = 0 \) or \( y \) after \( 2 \) ns
- \( z + 1 = 1 \)
- \( x = 0 \) or \( y \) after \( 2 \) ns
- \( z + 1 = 1 \)
- \( x = 0 \) or \( y \) after \( 2 \) ns

At time \( t+2 \): disp \( x=1 \) & execute \( Z \leq x \) or \( y \) after \( 2 \) ns;

Result −> 1 event: \( Z: 0 \rightarrow 1 \) @ \( t+4 \) ns;

At time \( t+4 \): disp \( Z=1 \), no statement execution
In actuality, statements with delays are simulated in a similar way as statements without delays by assigning a "fictitious" delay to a statement w/o delays.

The concept of fictitious delays allows treatment of statements with and without delay specification to be realized in an uniform manner, as illustrated below:

### Statements:

<table>
<thead>
<tr>
<th>Round 1</th>
<th>Round 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>S1=1</td>
<td>Available at time t: S1=0, S2=1, Sbar1=1, Sbar2=1, x=0, y=0, Z=0</td>
</tr>
<tr>
<td>T=t</td>
<td>Event Q Result -&gt; 2 events: Sbar1: 1-&gt;0 @ t+del, x: 0-&gt;1 @ t+del.</td>
</tr>
<tr>
<td>Z=1</td>
<td>At time t: display S1 = 1 &amp; execute sbar1 &lt;= not(s1); x &lt;= s1 and sbar2;</td>
</tr>
<tr>
<td>T=t+2del: disp Z=1, no statement execution</td>
<td>At time t+del: disp Z=0, x=1, T=t+del, x=1</td>
</tr>
<tr>
<td></td>
<td>execute: y &lt;= s2 and sbar1; Z &lt;= x or y after 2ns;</td>
</tr>
<tr>
<td></td>
<td>Result -&gt; 1 event: Z: 0 -&gt; 1 @ t+2*del;</td>
</tr>
</tbody>
</table>

DES without delays:

- These rules and the concept of fictitious delays allows treatment of statements with and without delay however small (e.g., 1 fs).
- The following rules apply to such statements:
  1. \( \forall x, y \wedge x < y \Rightarrow x < y \) if \( x \leq y \).
  2. \( \forall x \wedge x = y \Rightarrow x = y \) if \( x \neq y \).
  3. \( \forall x \wedge x = y \rightarrow x = y \) if \( x \neq y \).

So a statement \( x \rightarrow s1 \) and \( s2 = 1 \) becomes equivalent to \( x \rightarrow s1 \) and \( s2 = 1 \) after a "fictitious" delay to a statement w/o delays.

### Event Queue for Simulation w/o Delays
Discussed the following:

- Uses of HDL
- Concepts of \( \wedge \)-delays to tackle statements w/o delays
- Concepts of events, event \( \wedge \), simulation time
- Discrete event simulation of statements w/ and w/o delays

- Concurrent statements, sequential statements & processes—basic concepts
- Discrete events simulation of statements w/ and w/o delays
- Concurrent statements, sequential statements & processes

- Basic concepts
- Incorporates elements of 2 or more of the above types
- Behavioral descriptions can be done at many levels. A mixed description

Three types of architecture descriptions: behavioral, dataflow, structural

- Behavioral, direct instantiation, etc., descriptions of a VHDL module
- Discrete event simulation of statements w/ and w/odelays
- Concurrent statements, sequential statements & processes
- Basic concepts

- Concurrent statements, sequential statements & processes
- Incorporates elements of 2 or more of the above types
- Behavioral descriptions can be done at many levels. A mixed description

Three types of architecture descriptions: behavioral, dataflow, structural

- Behavioral, direct instantiation, etc., descriptions of a VHDL module
- Discrete event simulation of statements w/ and w/odelays
- Concurrent statements, sequential statements & processes
- Basic concepts