Modeling Digital Systems

Systems Hierarchy

Increasing
• Fidelity
• # of events

Level of Abstraction

Functions
CPUs, memories
ALUs, registers
Switches
Describing Systems

- From Webster’s Dictionary:
  - **System**: “An assemblage of objects united by some form of regular interaction or dependence”

- What aspects of a digital system do we want to describe?
  - Interface
  - Function: behavioral and structural

What Elements Should be in a Description?

- Descriptions should be at multiple levels of abstraction
  - The descriptive elements must be common to multiple levels of hierarchy

- The elements should enable meaningful and accurate simulation of hardware described using the elements
  - Elements should have attributes of time as well as function

- The elements should enable the generation of hardware elements that realize a correct physical implementation
  - Existence of a mapping from elements to VLSI devices
What Elements Should be in a Description?

• VHDL was conceived for the description of digital systems
  – From switches to networked systems

• Keep in mind the pragmatic issues of design re-use and portability of descriptions
  – Portability across technology generations
  – Portability across a range of cost/performance points

• Attributes of digital systems serve as the starting point
  – Language features designed to capture the key attributes

Attributes of Digital Systems

• Digital systems are about signals and their values
• Events, propagation delays, concurrency
  – Signal value changes at specific points in time
• Time ordered sequence of events produces a waveform
Attributes of Digital Systems: Timing

- Timing: computation of events takes place at specific points in time
- Need to "wait for" an event: in this case the clock
- Timing is an attribute of both synchronous and asynchronous systems

Example: Asynchronous communication
- No global clock
- Still need to wait for events on specific signals
Attributes of Digital Systems: Signal Values

- We associate logical values with the state of a signal

<table>
<thead>
<tr>
<th>Value</th>
<th>Interpretation</th>
</tr>
</thead>
<tbody>
<tr>
<td>U</td>
<td>Uninitialized</td>
</tr>
<tr>
<td>X</td>
<td>Forcing Unknown</td>
</tr>
<tr>
<td>0</td>
<td>Forcing 0</td>
</tr>
<tr>
<td>1</td>
<td>Forcing 1</td>
</tr>
<tr>
<td>Z</td>
<td>High Impedance</td>
</tr>
<tr>
<td>W</td>
<td>Weak Unknown</td>
</tr>
<tr>
<td>L</td>
<td>Weak 0</td>
</tr>
<tr>
<td>H</td>
<td>Weak 1</td>
</tr>
<tr>
<td>-</td>
<td>Don’t Care</td>
</tr>
</tbody>
</table>

Attributes of Digital Systems: Multiple Drivers

- Shared Signals
  - multiple drivers

- How is the value of the signal determined?
  - arbitration protocols
  - wired logic
Modeling Digital Systems

• We seek to describe attributes of digital systems common to multiple levels of abstraction
  – events, propagation delays, concurrency
  – waveforms and timing
  – signal values
  – shared signals

• Hardware description languages must provide constructs for naturally describing these attributes of a specific design
  – simulators use such descriptions for "mimicing" the physical system
  – synthesis compilers use such descriptions for synthesizing manufacturable hardware specifications that conform to this description

Execution Models for VHDL Programs

• Two classes of execution models govern the application of VHDL programs

• For Simulation
  – Discrete event simulation
  – Understanding is invaluable in debugging programs

• For Synthesis
  – Hardware inference
  – The resulting circuit is a function of the building blocks used for implementation
    • Primitives: NAND vs. NOR
    • Cost/performance
Simulation vs. Synthesis

- Simulation and synthesis are complementary processes

Simulation of Digital Systems

- Digital systems are modeled as the generation of events – value transitions – on signals
- Discrete event simulations manage the generation and ordering of events
  - Correct sequencing of event processing
  - Correct sequencing of computations caused by events
Discrete Event Simulation: Example

Initial state: $a = b = 1$, sum = carry = U

Simulation Time | Event List Head | Initial state: $a = b = 1$, sum = carry = U
--- | --- | ---
0ns | $U \rightarrow 1$: carry@5ns | $U \rightarrow 0$: sum@5ns
5ns | $U \rightarrow 1$: carry@5ns | $U \rightarrow 0$: sum@5ns
10ns | 1$\rightarrow 0$: carry@10ns | 0$\rightarrow 1$: sum@10ns
15ns | 1$\rightarrow 0$: a@15ns | 0$\rightarrow 1$: b@10ns

Discrete Event Simulation

- Management of simulation time: ordering of events
- Two step model of the progression of time
  - Evaluate all affected components at the current time: events on input signals
  - Schedule future events and move to the next time step: the next time at which events take place
Simulation Modeling

- VHDL programs describe the generation of events in digital systems
- Discrete event simulator manages event ordering and progression of time
- Now we can quantitatively understand accuracy vs. time trade-offs
  - Greater detail → more events → greater accuracy
  - Less detail → smaller number of events → faster simulation speed

Synthesis and Hardware Inference

- Both processes can produce very different results!
Summary

• VHDL is used to describe digital systems and hence has language constructs for key attributes
  – Events, propagation delays, and concurrency
  – Timing, and waveforms
  – Signal values and use of multiple drivers for a signal

• VHDL has an underlying discrete event simulation model
  – Model the generation of events on signals
  – Built in mechanisms for managing events and the progression of time
  – Designer simply focuses on writing accurate descriptions