Hardware Description Languages

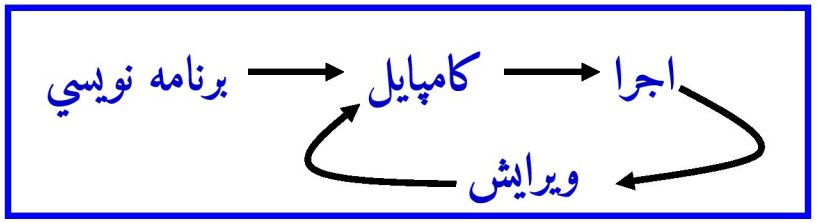
HDLs

Hardware Description Languages

- **An HDL:**
  - Describes circuits and systems in textual format like a software program
  - It is NOT sequential like software programs
  - Can be processed by computers
  - Examples: VHDL, Verilog, AHDL, MODAL
- **Applications:**
  - Simulation
  - Synthesis

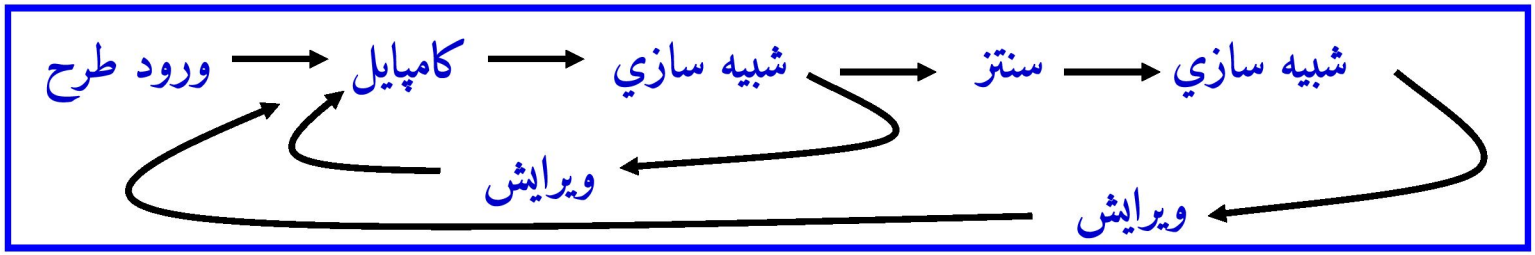
Simulation and Synthesis

- **Simulation**
  - Input waveform and circuit description --> Output waveform
  - Predicts circuit behavior before fabrication (debug before physical implementation)



Simulation and Synthesis

- **Synthesis**
  - Automated circuit generation
  - Software program takes design description and tries to find the equivalent circuit



Verilog HDL

- Verilog or VHDL?
  - Similar concepts and capabilities
  - Different syntax
- Verilog:
  - Similar to C/C++ syntax
  - Case sensitive
  - Comments: //
  - ; at the end of each statement

Example

- **module:**
  - A hardware component
- **ports:**
  - A component's input(s) and output(s)
    - must specify input or output or inout
- **wires:**
  - Intermediate signals

```
//HDL Example 3-1
//------------------------
module smpl_circuit(A,B,C,x,y);
   input A,B,C;
   output x,y;
   wire e;
   and g1(e,A,B);
   not g2(y, C);
   or  g3(x,e,y);
endmodule
```

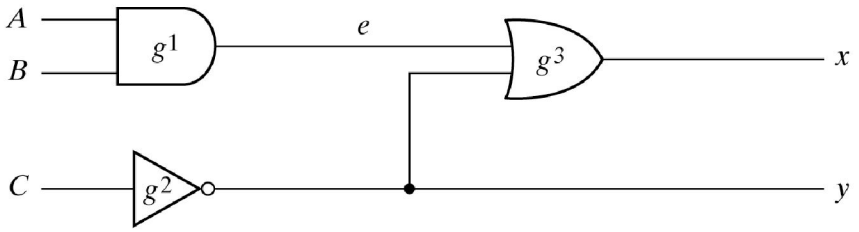


Fig. 3-37 Circuit to Demonstrate HDL

- **Concurrent Execution**
  - **Order not important**

Delays

- **Timescale:**
  - `` `timescale (time unit)/(precision) ``
  - `Example: `timescale 1ns/100ps`
    - `1ns:` time unit
    - `100ps:` precision (unit used for rounding)
    - Default: 1ns/100ps (0.1 ns)
- **Gate Delays:**
  - Used by simulator to generate correct waveforms

Delays

```
//HDL Example 3-2
//--------------------------------
//Description of circuit with delay
module circuit_with_delay (A,B,C,x,y);
   input A,B,C;
   output x,y;
   wire e;
   and #(30) g1(e,A,B);
   or  #(20) g3(x,e,y);
   not #(10) g2(y,C);
endmodule
```

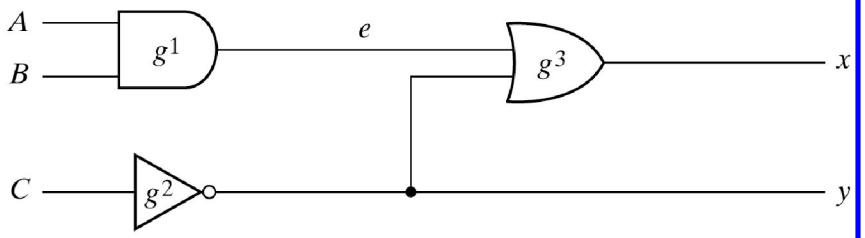


Fig. 3-37 Circuit to Demonstrate HDL

Testbench

- **Testbench:**
  - Program written to test the circuit
  - Applies the input values at the right times
  - Instantiates (calls) the module(s)
  - Reports the outputs for evaluation

Testbench Example

```
//HDL Example 3-3
//----------------------
//Stimulus for simple circuit
module stimcrct;
reg A,B,C;
wire x,y;
circuit_with_delay cwd(A,B,C,x,y);
initial
   begin
        A = 1'b0; B = 1'b0; C = 1'b0;
     #100
        A = 1'b1; B = 1'b1; C = 1'b1;
     #100  $finish;
   end
endmodule


//Description of circuit with delay
module circuit_with_delay (A,B,C,x,y);
   input A,B,C;
   output x,y;
   wire e;
   and #(30) g1(e,A,B);
   or  #(20) g3(x,e,y);
   not #(10) g2(y,C);
endmodule
```

Testbench:

- Doesn't have ports
- Instantiates the module
- Assigns values: <size>'<base><value>
  - Base: b, o, d, h.
- ABC = "000" → "111"
- 100ns delay
- Finishes at 200ns

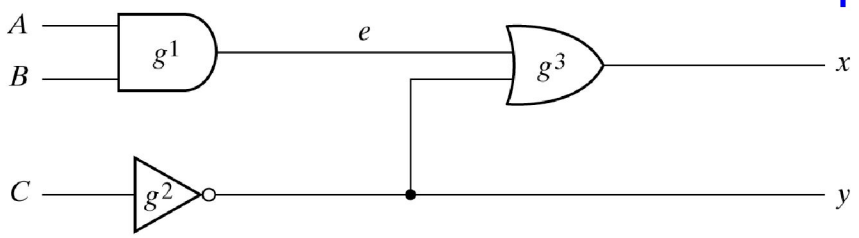


Fig. 3-37 Circuit to Demonstrate HDL

Testbench Example

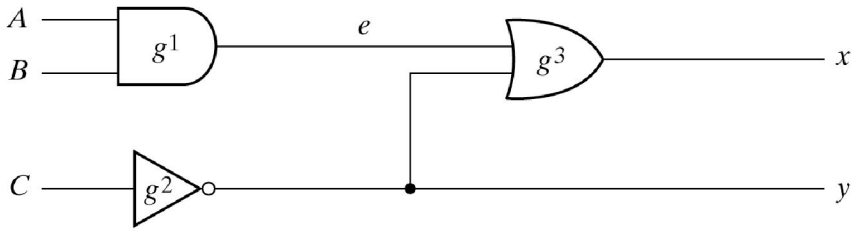


Fig. 3-37 Circuit to Demonstrate HDL

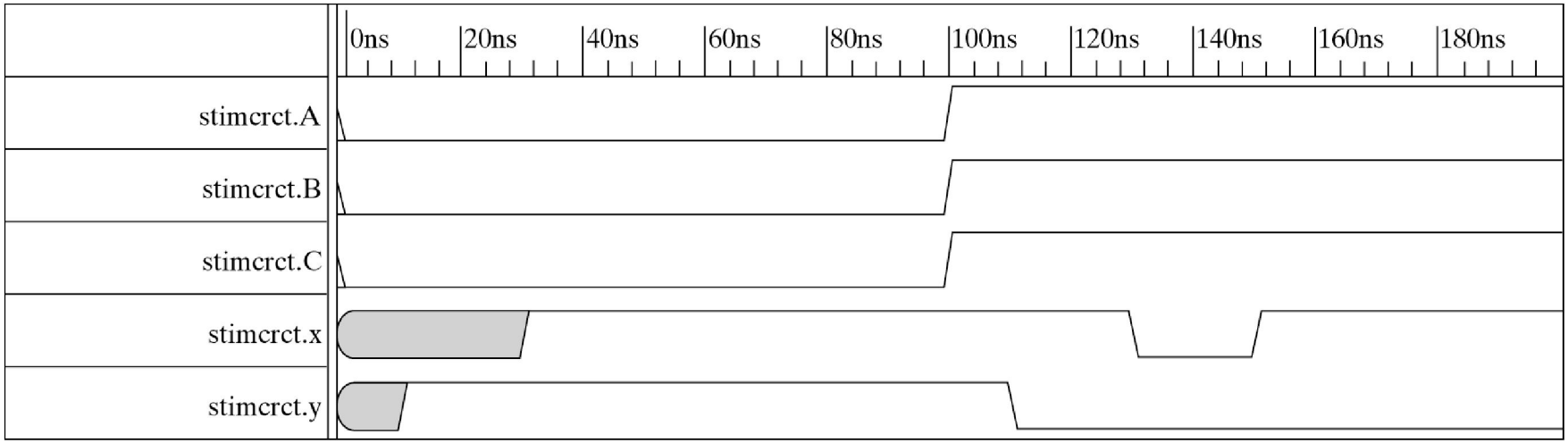


Fig. 3-38 Simulation Output of HDL Example 3-3

Primitives

- **Built-in Primitives:**
  - and, or, not, nand, nor, xor, xnor, buf
- **User-Defined Primitives (UDP):**
  - User can define by truth table
  - Use **`primitive`** keyword (instead of **`module`**)
  - Only one output but many inputs
    - first output, then inputs
  - Inputs' order must correspond with the truth table
  - Truth table between **`table`** and **`end table`**

UDP

```
//HDL Example 3-5
//-----------------------------------
//User defined primitive(UDP)
primitive crctp (x,A,B,C);
   output x;
   input A,B,C;
//Truth table for x(A,B,C) = Minterms (0,2,4,6,7)
   table
//     A   B   C  :  x  (Note that this is only a comment)
       0   0   0  :  1;
       0   0   1  :  0;
       0   1   0  :  1;
       0   1   1  :  0;
       1   0   0  :  1;
       1   0   1  :  0;
       1   1   0  :  1;
       1   1   1  :  1;
   endtable
endprimitive
```

Simulation

- **ModelSim:**
  - An industrial and widely-used simulator
  - Around $25K