Introduction to Digital Design and Computer Architecture

4. Hardware Description Language

Lilia Kirakosyan

Russian-Armenian University

2025

Hardware Description Language (HDL)

- **Two leading HDLs:**
  - **SystemVerilog:**
    - Developed in 1984 by Gateway Design Automation.
    - IEEE standard (1364) in 1995.
    - Extended in 2005 (IEEE STD 1800-2009).
  - **VHDL 2008:**
    - Developed in 1981 by the Department of Defense.
    - IEEE standard (1076) in 1987.
    - Updated in 2008 (IEEE STD 1076-2008).

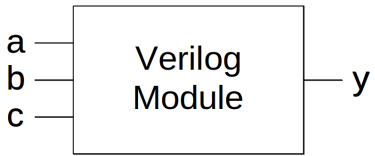
Simulation and Synthesis

- **Simulation:**
  - Inputs applied to circuit.
  - Outputs checked for correctness.
  - Saves millions of dollars by debugging in simulation instead of hardware.
- **Synthesis:**
  - Transforms HDL code into a *netlist*, describing the hardware
    - (i.e., a list of gates and the wires connecting them).

IMPORTANT:
When using an HDL, think of the hardware the HDL should produce.

SystemVerilog Modules

- **Two types of Modules:**
  - **Behavioral:** Describe what a module does.
  - **Structural:** Describe how it is built from simpler modules.

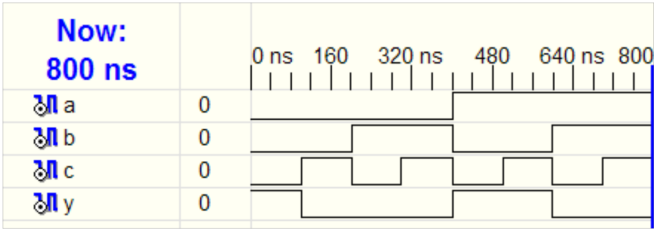


SystemVerilog Example

```
module example(input  logic a, b, c,
               output logic y);
  assign y = ~a & ~b & ~c | a & ~b & ~c | a & ~b &  c;
endmodule
```

SystemVerilog Example with Images

```
module example(input  logic a, b, c,
               output logic y);
  assign y = ~a & ~b & ~c | a & ~b & ~c | a & ~b &  c;
endmodule
```



Reduction Operators

```
module and8(input  logic [7:0] a,
            output logic       y);
   assign y = &a;
```

Reduction Operators

```
module and8(input  logic [7:0] a,
            output logic       y);
   assign y = &a;
   // &a is much easier to write than
   // assign y = a[7] & a[6] & a[5] & a[4] &
   //            a[3] & a[2] & a[1] & a[0];
endmodule
```

Conditional Assignment

```
module mux2(input  logic [3:0] d0, d1,
            input  logic       s,
            output logic [3:0] y);
   assign y = s ? d1 : d0;
endmodule
```

Common SystemVerilog Symbols

Symbol	Meaning
&	Bitwise AND
—	Bitwise OR
	Bitwise NOT
	Bitwise XOR
==	Logical equality
!=	Logical inequality
	Logical AND
——	Logical OR
!	Logical NOT

Numbers

Number	# Bits	Base	Decimal Equivalent	Stored
3'b101	3	binary	5	101
'b11	unsized	binary	3	00...0011
8'b11	8	binary	3	00000011
8'b1010_1011	8	binary	171	10101011
3'd6	3	decimal	6	110
6'o42	6	octal	34	100010
8'hAB	8	hexadecimal	171	10101011
42	Unsized	decimal	42	00...0101010