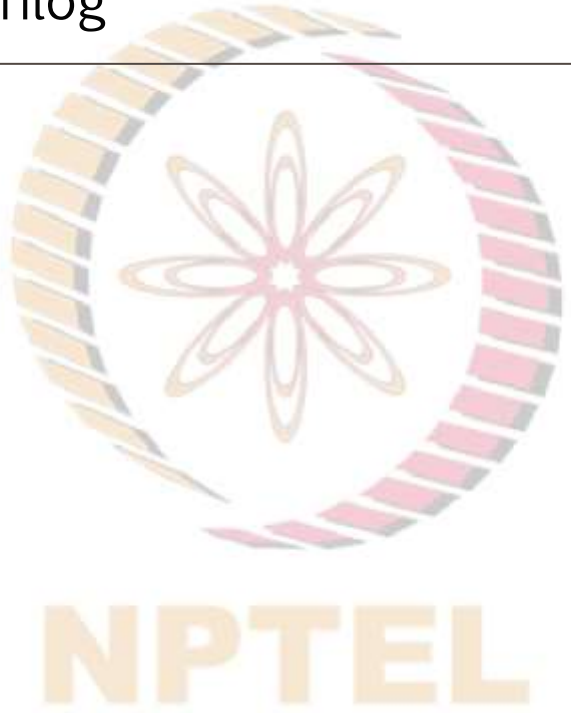
VLSI DESIGN FLOW: RTL TO GDS

Lecture 9
Hardware Modeling: Introduction to Verilog-I

Sneh Saurabh
Electronics and Communications Engineering
IIIT Delhi

Lecture Plan

- Features of Hardware Description Languages (HDLs)
- Language constructs of Verilog



Hardware Description Languages (HDL)

HDL: Distinct Features (1)

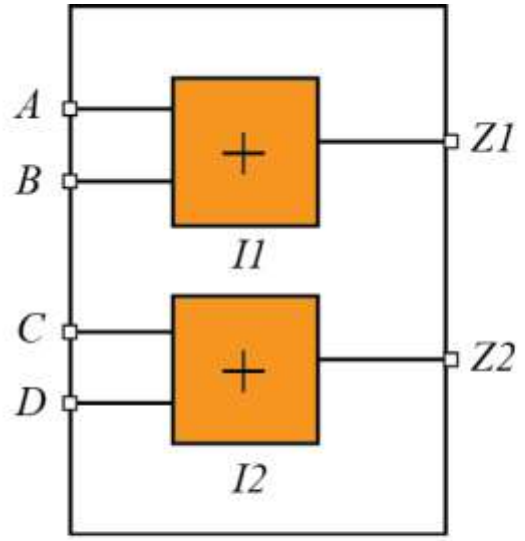
- Several features that are similar to programming languages such as C, C++
- Additional features in HDL to model hardware easily and realistically

Concurrency

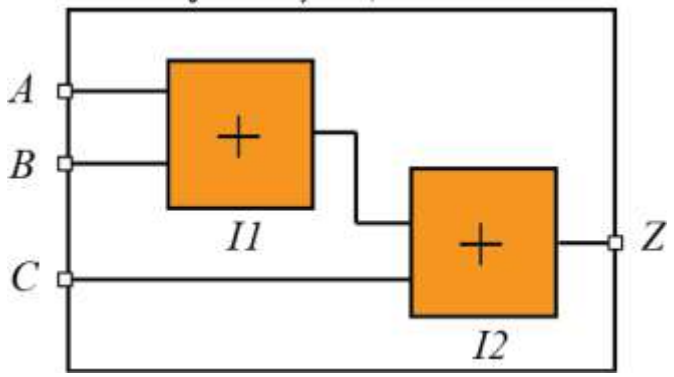
- Computation can be done in parallel in hardware
- HDL must support syntax/semantics to distinguish parallel and sequential operations

Notion of Time

- Describe behavior of circuit with respect to time
- Concurrent/sequential operations
- Ability to create waveform (periodic signal)



S. Saurabh, "Introduction to VLSI Design Flow". *Cambridge University Press*, 2023.



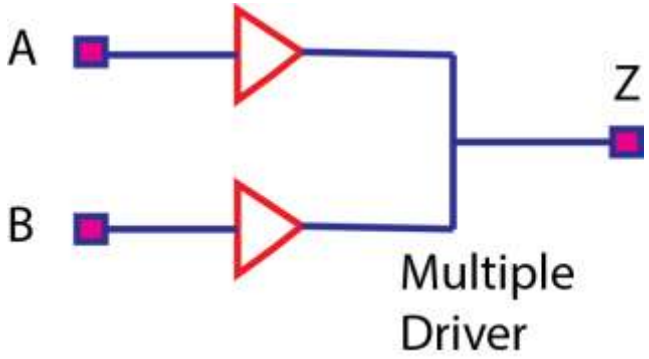
HDL: Distinct Features (2)

Electrical Characteristics

- Tristate
- Driver Strength

Bit-true data types

- Behaviour of buses and how individual bits behave



HDLs : Verilog and VHDL

- Two common HDLs are: Verilog and VHDL

Verilog:

- Created in 1983-84 at Gateway Design Automation
- **Verilog** = **Veri**fication + **Log**ic
- Started for the purpose of verification using simulation (fast simulation using Verilog XL)
- Logic synthesis using Design Compiler
- Gained popularity due to simplicity (similarity with C)
- IEEE standardized in 1995 and 2001
- SystemVerilog: superset of Verilog with added functionality for design verification (standardized in 2009)

VHDL: VHSIC Hardware Description Language

- VHSIC: Very High Speed Integrated Circuit
- Initially started as documenting language for integrated circuits in early 1980s
- Verbose and strict type checking
- IEEE standardized in 1987 and 2019

Introduction to Verilog Language Features

Meaning of Words…

‘When I use a word,’ Humpty Dumpty said in rather a scornful tone, ‘it means just what I choose it to mean—neither more nor less.’

‘The question is,’ said Alice, ‘whether you can make words mean different things.’

‘The question is,’ said Humpty Dumpty, ‘which is to be master—that’s all.’

—Lewis Carroll, *Through the Looking-Glass*, Chapter 6, 1871

Source: https://commons.wikimedia.org/wiki/File:LewisCarrollSelfPhoto.jpg Lewis Carroll, Public domain, via Wikimedia Commons

Lexical Tokens

- A Verilog file is a stream of lexical tokens
- Lexical Rules
  - Similar to C
  - Case sensitive

Tokens

- White spaces, Comments, Keywords, Operators
- Identifiers
- Numbers, Strings

White space

- White space can contain the characters for spaces, tabs, newlines, and form feeds.
- Used as separators for tokens

Comments

- Single line comment: //
- Multiple line comments: /* … */

```
//  This is a comment
/*
This is a block comment
*/
```

Keywords

- Reserved word for Verilog
- In lower case only
- Examples: **module**, **input**, **output**, **initial**, **begin**, **end**, **always**, **endmodule**, etc.

Operators

- Predefined sequences of one, two, or three characters used in an expression
- Examples: ! + – && == !== etc.

Syntax and Semantics: Identifiers

Identifiers

- Unique names given to an object so that it can be referred to in the Verilog code
- Objects can be modules, ports, nets, registers, functions, etc.

Rules for Identifiers

- Must begin with an alphabetic character or underscore (a-z A-Z _)
- Subsequent characters can be a-z A-Z 0-9 _ $
- Case sensitive
- Maximum allowed length < 1024 [by language]

Mymodule_top
Register_123

Net_1
net_1

Escaped Identifiers

- Any character can be used in an identifier by "***escaping***" the identifier
  - Preceding the identifier with a backslash "\"
  - Ending with a white space

*\net_(a + b)*c*

S. Saurabh, "Introduction to VLSI Design Flow". *Cambridge University Press*, 2023.

Syntax and Semantics: Numbers (1)

Numbers

- Can be integers or real number
- Convenient/readable representation in code
  - Can be specified in decimal, hexadecimal, octal or binary format
- Internally represented as sequence of bits

Integers:

- In traditional format like 169, -123
- In the format:

  -<size>'<base><value>
  - - for negative sign (optional)
  - <size> number of bits (default 32)
  - <base> can be b/B for binary, o/O, for octal, d/D for decimal, h/H hexadecimal
  - <value> value of the integer

Verilog	Internal Representation
1	0000 0000 0000 0000 0000 0000 0000 0001
1'b1	1
8'ha1	1010 0001
6'o71	111 001

S. Saurabh, "Introduction to VLSI Design Flow". *Cambridge University Press*, 2023.

Syntax and Semantics: Numbers (2)

Rules for Integers: -<size>'<base><value>

- For hexadecimal, octal, and binary constants, x/X represent the unknown/don't care value and z/Z/? represent the high-impedance value
  - ➢ Prefer ? when high impedance is don't care
- When <size> is smaller than <value> then leftmost bits from the <value> are truncated
- When <size> is greater than <value> then leftmost bits are filled with
  - 0 if leftmost bit in <value> is 0/1
  - Z if leftmost bit in <value> is Z
  - X if leftmost bit in <value> is X
- _ can be used in the middle of number to enhance readability
- Negative numbers are internally represented in two's complement form.

Verilog	Internal Representation
8'b100z00?1	100z 00z1

Verilog	Internal Representation
6'h88	00 1000

Verilog	Internal Representation
8'b11	0000 0011
8'bz1	zzzz zzz1

Verilog	Internal Representation
8'b1010_1010	1010 1010

Verilog	Internal Representation
-8'd6	1111 1010

S. Saurabh, "Introduction to VLSI Design Flow". *Cambridge University Press*, 2023.

Syntax and Semantics: Real Numbers, Strings

Real Numbers:

- Can be represented in decimal ◇.◇
  - Example: 3.14159
- Can be represented in scientific notation <mantissa>E<exponent>
  - Example: 2.99E8
- Internally, real numbers are represented in IEEE standard for double-precision floating-point numbers

String:

- Is a sequence of characters enclosed by double quotes and contained on a single line
- Each character is represented by its corresponding 8-bit ASCII value.
- Example: “Hello”

Data Values and Data Types

Verilog supports four-valued data:

- 0: logic false or Boolean zero
- 1: logic true or Boolean one
- X: unknown value
- Z: high-impedance state

Nets:

- Represent structural connections
- Cannot store value
- **wire, supply0, supply1, wand, wor**

Variables:

- Element that store value in simulation
- Declared using keyword **reg**
- Store last assigned value, until changed by another assignment

Two primary data types:

1. Nets
2. Variables

```
wire w1, w2;
wire w3=1'b1;
supply0 gnd;
supply1 vdd;
```

```
reg r1, r2;
```

- We can assign **reg** in procedural blocks
- Can model flip-flops, latches, and also combinational elements

S. Saurabh, "Introduction to VLSI Design Flow". *Cambridge University Press*, 2023.

Data Types: Vectors and Arrays

- Nets and variables are 1-bit wide (scalar) by default
- Can declare them as vectors by preceding the declaration with vector definition in the following format:

  [⟨left_range⟩:⟨right_range⟩].
- ⟨left_range⟩ is the most significant bit (MSB) and ⟨right_range⟩ is the least significant bit (LSB).

wire *[31:0]databus;*
reg *[7:0]addressbus;*

databus[4] = 1'b0;
addressbus[3:0] = 4'b1001;

- **Bit-select:** select a bit of a vector by specifying the address within the square bracket ([]).
- **Part-select:** select a portion of a vector by specifying the range of MSB and LSB separated by a colon (:)

reg *r[15:0];*
wire *matrix[9:0][9:0];*

- **Arrays:** can be used for grouping elements into multidimensional objects (specify [⟨left_range⟩:⟨right_range⟩] after the identifier

S. Saurabh, "Introduction to VLSI Design Flow". *Cambridge University Press*, 2023.

References

- S. Saurabh, “Introduction to VLSI Design Flow”. Cambridge: *Cambridge University Press*, 2023.
- R. Seisyan. S. Palnitkar, “Verilog HDL: a guide to digital design and synthesis”, *Pearson Education India*, 2003
- “IEEE standard Verilog hardware description language.” IEEE Std 1364-2001 (2001), pp. 1–792.