Module #05 : Verilog HDL Operands

5.1: Literals:

- Literals are constant-valued operands that can be used in Verilog expressions.
- The two common Verilog literals are:
  1) String: A string literal is a one-dimensional array of characters enclosed in double quotes (" ")
  2) Numeric: constant numbers specified in binary, octal, decimal or hexadecimal.

Syntax:

n'Fddd *..., where*

n - integer representing number of bits

F - one of four possible base formats:

b (binary), o (octal), d (decimal),

h (hexadecimal). Default is d.

dddd - legal digits for the base format

Example 5.1:

"time is"// string literal

267 // 32-bit decimal number

2'b01 // 2-bit binary

20'hB36F// 20-bit hexadecimal number

'o62 // 32-bit octal number

Module #05 : Verilog HDL Operands

5.2: Wires, Regs, and Parameters:

- Wires, regs and parameters can also be used as operands in Verilog expressions.

Syntax:

```
reg variable_name;
reg [MSB : LSB] vector_variable_name;
wire wire_variable_name;
parameter par1 = value;
```

Example 5.2:

```
  reg a;
  reg b;
  wire c;
parameter N = 4;
assign c = a + b;
assign c = a + N;
```

Module #05 : Verilog HDL Operands

5.3: Bit-Selects “x[3]” and Part-Selects “x[5:3]”:

- Bit-selects and part-selects are a selection of a single bit and a group of bits, respectively, from a wire, reg or parameter vector using square brackets “[ ]”.
- Bit-selects and part-selects can be used as operands in expressions in much the same way that their parent data objects are used.

Syntax:

```
variable_name[index]
variable_name[MSB:LSB]
```

Example 5.3:

```
reg [7:0] a, b;
reg [3:0] ls;
reg c;
c = a[7] & b[7]; // bit-selects
ls = a[7:4] + b[3:0]; // part-selects
```

Module #05 : Verilog HDL Operands

5.4: Function Calls:

- The return value of a function can be used directly in an expression without first assigning it to a register or wire variable.

Syntax:

function_name (argument_list)

Example 5.4:

```
assign a = b & c & chk_bc(c, b);// chk_bc is a function
/* Definition of the function */
function chk_bc;// function definition
input  c, b;
 chk_bc = b^c;
endfunction
```