



# SNS COLLEGE OF TECHNOLOGY

Coimbatore-35  
An Autonomous Institution



Accredited by NBA – AICTE and Accredited by NAAC – UGC with 'A+ +' Grade  
Approved by AICTE, New Delhi & Affiliated to Anna University, Chennai

## DEPARTMENT OF ELECTRONICS & COMMUNICATION ENGINEERING

### 16EC303–VLSI DESIGN

III YEAR/ V SEMESTER

UNIT 5-SPECIFICATION USING VERILOG HDL

TOPIC 3,4 –GATE PRIMITIVES, GATE DELAYS



# OUTLINE



- INTRODUCTION
- VERILOG – GATE DELAYS
- DESCRIPTION OF CIRCUIT WITH DELAY
- DATAFLOW MODELING
- MODULE CIRCUIT\_WITH\_DELAY & HALF ADDER- WAVE FORM COFIGURATIONS
- BEHAVIORAL MODELING DELAYS
  - ACTIVITY
  - DESCRIPTION STYLES
  - DELAY SPECIFICATION IN PRIMITIVES
  - DELAY AND TIME SCALES
  - USER DEFINED PRIMITIVES-DEFINITIONS & SYMBOLS
  - ASSESSMENT
- SUMMARY



# VERILOG STRUCTURAL PRIMITIVES



- Gate-level
  - One-output boolean operators: **and, or, xor, nand, nor, xnor**
    - E.g.,  $C = A+B$   
**or (C, A, B);**
    - E.g.,  $C = A+B+D$   
**or (C, A, B, D);**
  - One-input operators: **not**
    - E.g.,  $A = \text{not } Z$   
**not (A, Z);**
    - E.g.,  $A = \text{not } Z, B = \text{not } Z$   
**not (A, B, Z);**
    - Buf is like not but just replicates signals – we don't need
- Transistor-level primitives too
  - We will not use



## 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
```

**Delay:** Time duration between assignment from RHS to LHS

All continuous assignment statements execute concurrently

Order of the statement does not impact the design



# DATAFLOW MODELING



Delay can be introduced

Example: `assign #2 sum = a ^ b;`

“#2” indicates 2 time-units

No delay specified : 0 (default)

Associate time-unit with physical time

``timescale` time-unit/time-precision

Example: ``timescale 1ns/100 ps`

Timescale

``timescale 1ns/100ps`

1 Time unit = 1 ns

Time precision is 100ps (0.1 ns)

10.512ns is interpreted as 10.5ns

- To specify the amount of delay from the input to the output of gates.
- The delay is specified in terms of time units and the symbol #.
- The association of a time unit with physical time is made using ***timescale*** compiler directive.
- Compiler directive starts with the “backquote ( ` )” symbol.
  - ``timescale 1ns/100ps`
- The first number specifies the *unit of measurement* for time delays.
- The second number specifies the *precision* for which the delays are rounded off, in this case to 0.1ns.



# MODULE CIRCUIT\_WITH\_DELAY & HALF ADDER



## 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
```

```
`timescale 1ns/100ps
```

```
module HalfAdder (A, B, Sum, Carry);
```

```
  input A, B;
```

```
  output Sum, Carry;
```

```
  assign #3 Sum = A ^ B;
```

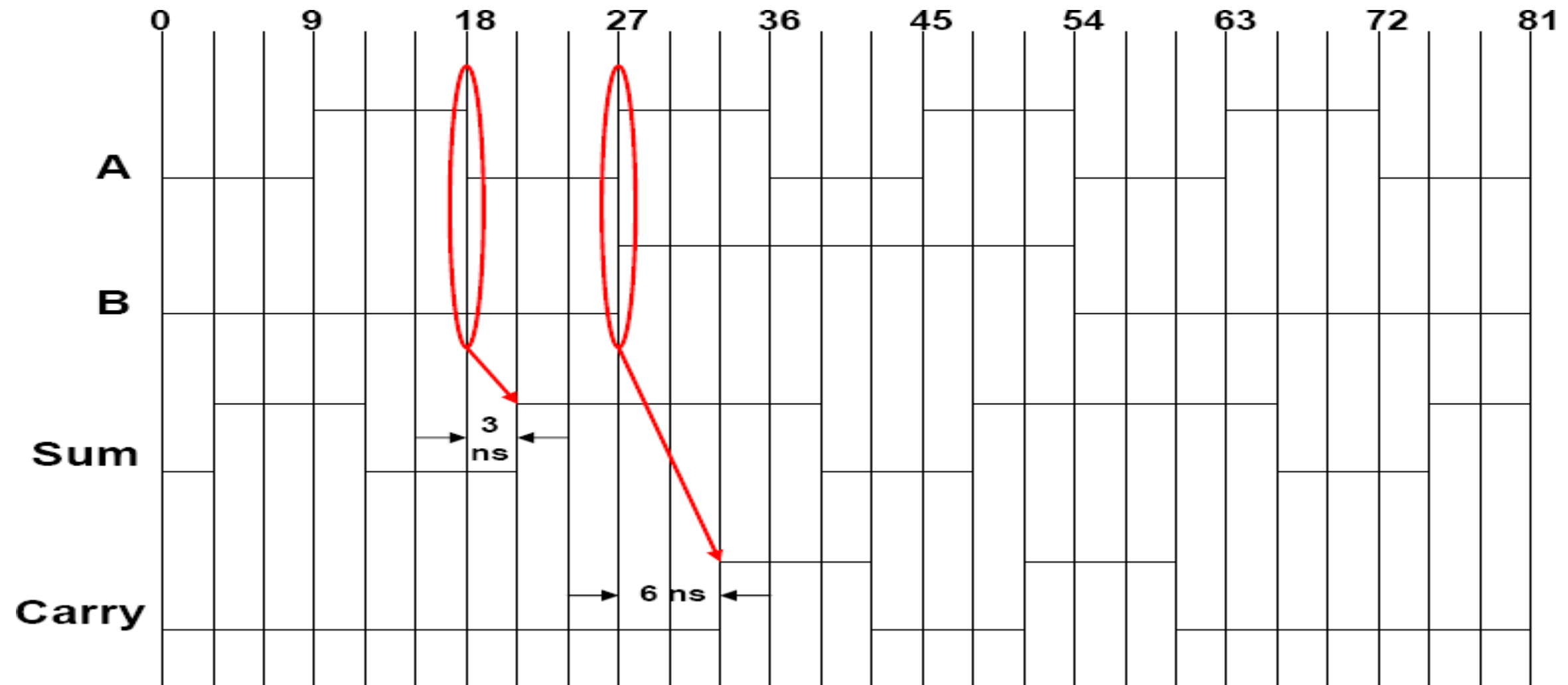
```
  assign #6 Carry = A & B;
```

```
endmodule
```

10/30/2023

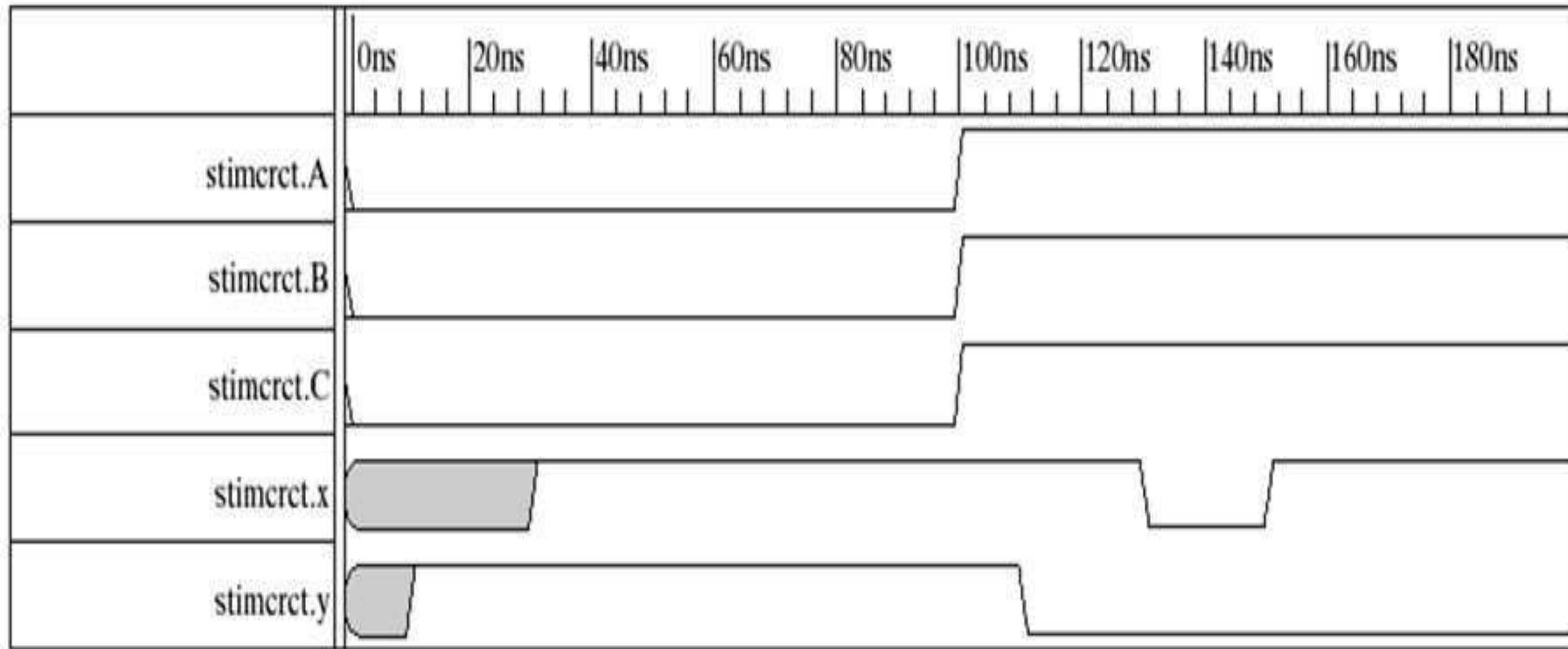


# WAVE FORM COFIGURATION HALF ADDER





# WAVE FORM COFIGURATION MODULE WITH DELAY



In the above example, **cwd** is declared as one instance **circuit\_with\_delay**. (similar in concept to object<->class relationship)





# BEHAVIORAL MODELING



Statements with a Sequential Block: Procedural Assignments

Delay in Procedural Assignments

Inter-Statement Delay

Intra-Statement Delay

- Inter-Assignment Delay

- Example:

- Sum = A ^ B;

- #2 Carry = A & B;

- Delayed execution

- Intra-Assignment Delay

- Example:

- Sum = A ^ B;

- Carry = #2 A & B;

- Delayed assignment



# ACTIVITY



# DEBATE

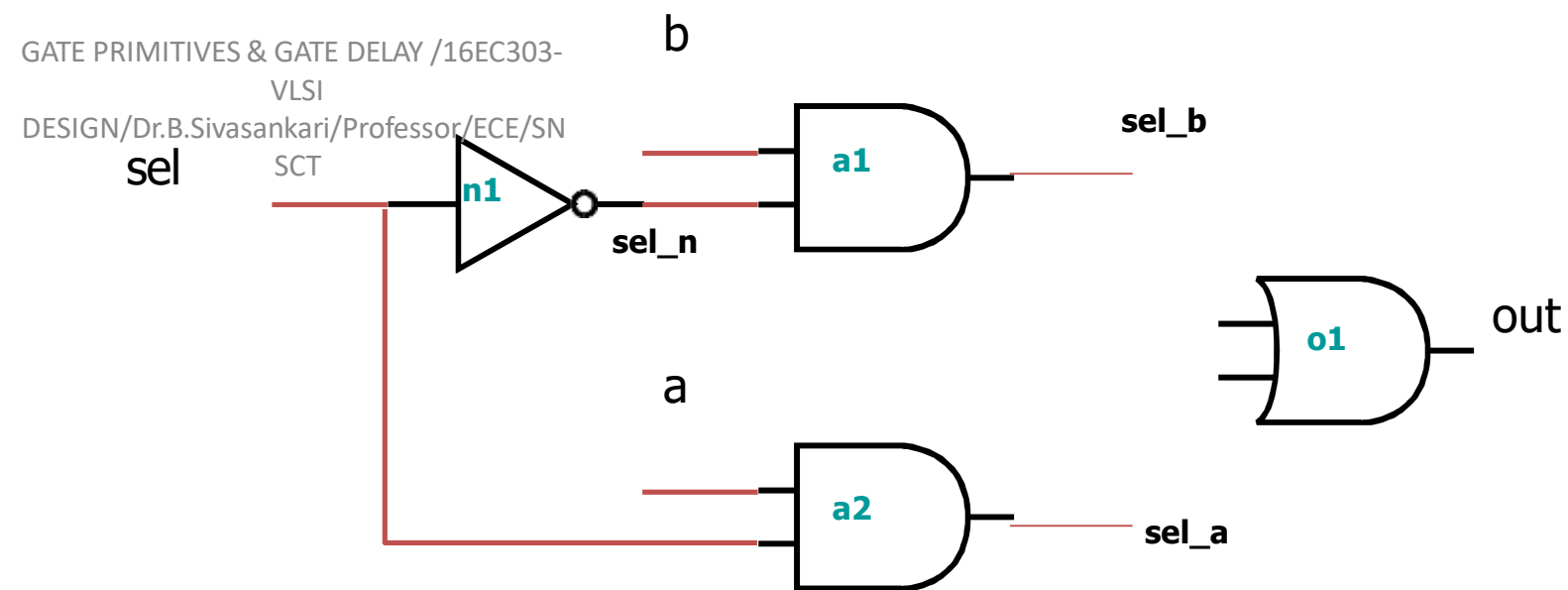


# DESCRIPTION STYLES



- **Structural:** Logic is described in terms of Verilog gate primitives
- Example:

```
not n1(sel_n, sel);
and a1(sel_b, b, sel_b);
and a2(sel_a, a, sel);
or o1(out, sel_b, sel_a);
```



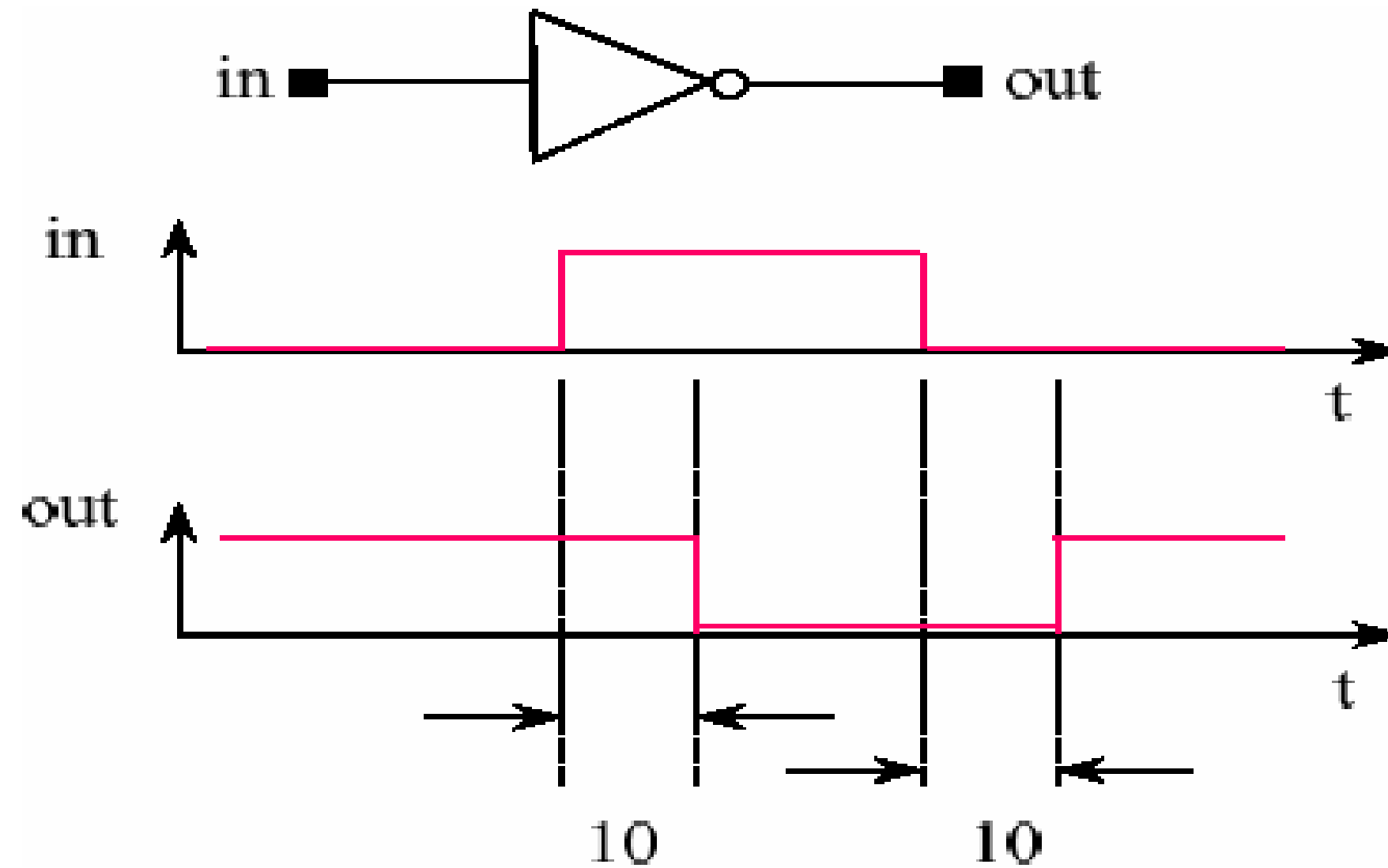


# DELAY SPECIFICATION IN PRIMITIVES



- Delay specification defines the propagation delay of that primitive gate.

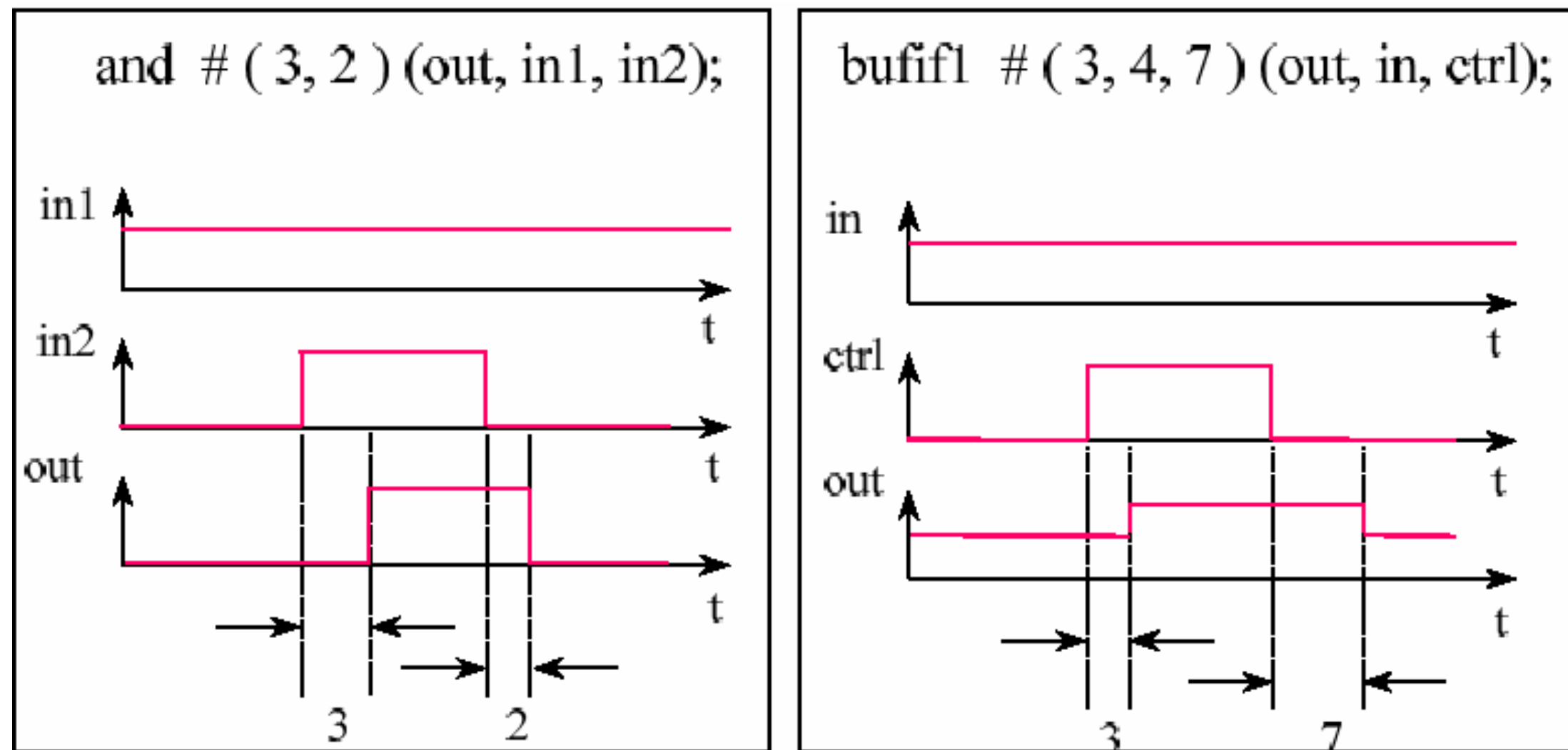
not #10 u0(out, in);





## DELAY SPECIFICATION IN PRIMITIVES

- Support (rise, fall, turn-off) delay specification.





## DELAY AND TIME SCALES

### Gate Description

buf #<delay> buf0(X,A);

where <delay> is:

<delay time> or

(<minimum delay>:<typical delay>:<maximum delay>)

*example:* buf #(3:4:5) buf0(X,A);  
or #1 u0(out, in0, in1);

### Modeling Separate Rise and Fall Delays

not #<delay> not0(X,A);

where <delay> is

(<rise delay>,<fall delay>)

*example:* not #(2.23:2.33:2.66,3.33:3.47:3.9) not0(X,A);



## DELAY AND TIME SCALES...

- Three-state drivers: include rise, fall, and **turn off** delays

*example:* `bufif1 #(2.2:2.3:2.6, 3.3:3.4:3.9, 0.2:0.2:0.2) u0(out, in);`

- **Timescales**

The ``timescale` compiler directive can be used to specify delays in explicit time units.

**Syntax of the ``timescale` compiler directive:**

``timescale <unit>/<precision>`

``timescale 1ns/10ps`

*example:*

then the design will be simulated in units of 10 ps.

*example:*

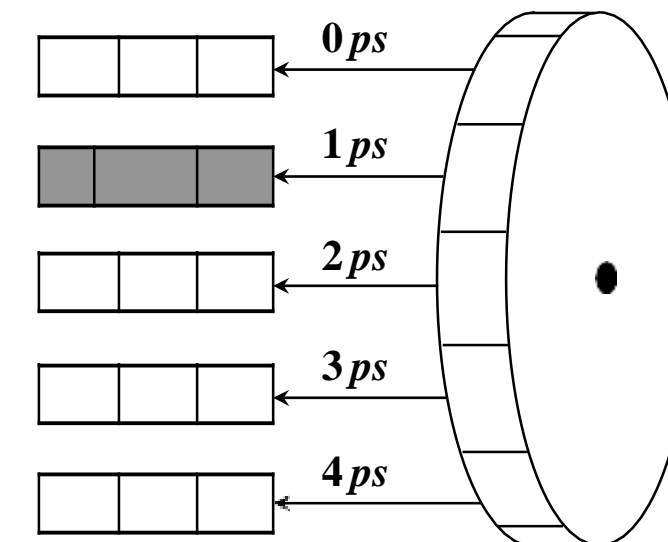
`not #(2.337,3.472) not1(X, A);` 2.337ns will be scaled to 2.34 ten pico-second units for simulation purposes.



## DELAY AND TIME SCALES...

- The smallest precision of all the 'timescale determines the time unit of simulation.

```
`timescale 1ns/10ps module m1(...);  
...  
`timescale 100ns/1ns module m2(...);  
...  
`timescale 1ps/1ps module m3(...);  
...
```







# USER DEFINED PRIMITIVES & UDP TABLE SYMBOLS



- UDPs permit the user to augment the set of pre-defined primitive elements.
- Use of UDPs may reduce the amount of memory required for simulation.
- Both level-sensitive and edge-sensitive behavior is supported.

symbol	Interpretation	Comments
0	Logic 0	
1	Logic 1	
x	Unknown	
?	Iteration of 0, 1, and x	input field input field output field
b	Iteration of 0 and 1 No change	
-	Change of value from v to w Same as (??)	
(vw)	Same as (01)	Any value change on input Rising edge on input Falling edge on input
* r f p n	Same as (10) Iteration of (01), (0x), and (x1)	Positive edge including x Negative edge including x
	Iteration of (10), (1x), and (x0)	



## USER DEFINED PRIMITIVES (UDP) DEFINITION

- **Pure combinational Logic**

```
primitive mux(o,a,b,s); output o;  
input a,b,s;
```

**table**

```
// a b s : o
```

```
0 ? 1 : 0;
```

```
1 ? 1 : 1;
```

```
? 0 0 : 0;
```

```
? 1 0 : 1;
```

```
0 0 x : 0;
```

```
1 1 x : 1;
```

```
endtable
```

```
endprimitive
```

- The output port must be the first port.
- UDP definitions occur outside of a module
- All UDP ports must be declared as scalar inputs or outputs. UDP ports cannot be inout.
- Table columns are inputs in order declared in primitive statement-colon, output, followed by a semicolon.
- Any combination of inputs which is not specified in the table will produce an 'x' at the output.



# USER DEFINED PRIMITIVES (UDP) DEFINITION...



- Level-sensitive Sequential Behavior

```
primitive latch(q,clock,data); output q;  
reg q;  
input clock,data;
```

```
table  
// clock data : state_output : next_state  
0 1 : ? : 1;  
0 0 : ? : 0;  
1 ? : ? : -;  
endtable  
endprimitive
```

▪The '?' is used to represent don't care condition in either inputs or current state.

▪The '-' in the output field indicates 'no change'.



## ASSESSMENT

1. How the time scale is used?
2. "#2" indicates -----
3. Draw the waveform configuration of following code

```
`timescale 1ns/100ps
```

```
module HalfAdder (A, B, Sum, Carry);
```

```
input A, B;
```

```
output Sum, Carry;
```

```
assign #3 Sum = A ^ B;
```

```
assign #6 Carry = A & B;
```

```
Endmodule
```

4. In UDP, The '?' is used to represent -----  
condition in either ----- or ----- state.



## SUMMARY & THANK YOU