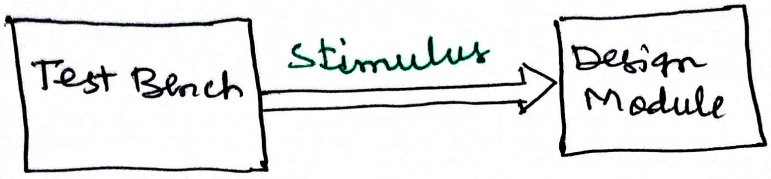
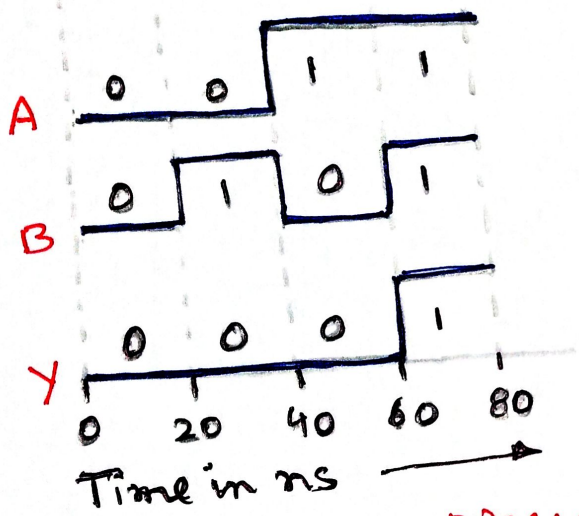
TEST BENCH



⇒ Once the design module is completed, functionality must be tested by applying
- stimulus (input)
- and then checking the results

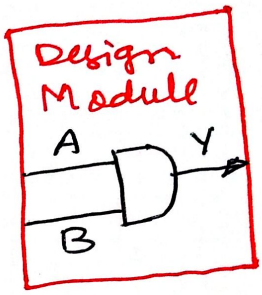
TIMING DIAGRAM



TRUTH TABLE

A	B	Y
0	0	0
0	1	0
1	0	0
1	1	1

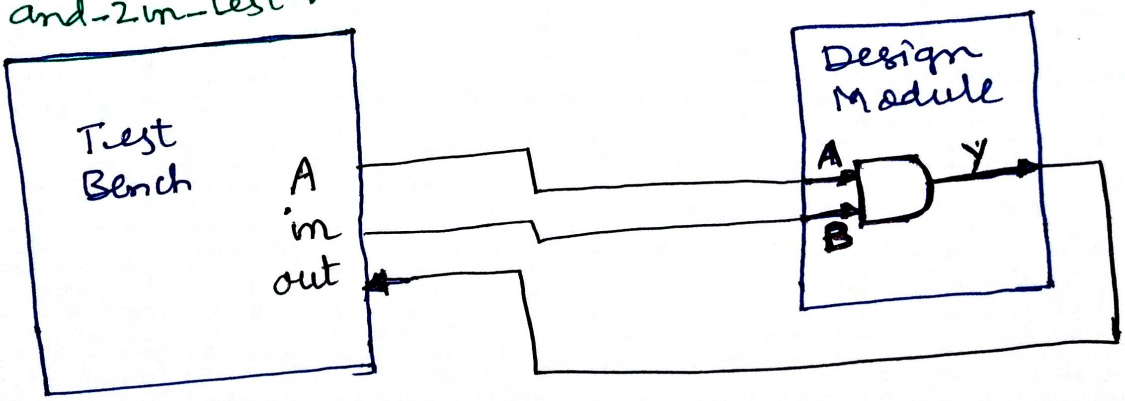
Design file
and_2in.v



NOTE:- Timing Diagram can be checked using simulator

Test Bench file
and_2in_test.v

Design file
and_2in.v



```
// THE DESIGN MODULE (and_2in.v)
module and_2in(A, B, Y);
        // Declare the design module
input A, B; // Declare the design inputs
output Y; // and output
wire Y;  // Declare the output as net
assign Y = A & B; // Realize the 2 input AND
                  // gate
endmodule
```

// TEST BENCH FOR FUNCTIONAL CHECKING
// OF THE DESIGN

```
`include "and_2in.v"
                    // This is the design file

`timescale 1ns/100ps
                    // Time base, refer Timing Diagram Page 1
                    // 1ns second denotes time unit
                    // 100ps denotes division
                    // time unit / division always
                    // smaller division shows high
                    // resolution

module and_2in_test;
                    // Denote the test module

  reg A, in; // refer timing diagram - 20ns
  // Declare design inputs as registers since we
  // need to hold the values

  wire out;
  // Declare outputs as wire (meaning net) since
  // we need to inter-connect other sub module

  and_2in U1 (
                    // Instantiate the design module
                  // U1 stand for the first instantiation

    .A(A),  // connect ports by name
    .B(in),
    .Y(out),
```

// NO INPUT/OUTPUT IN TEST-BENCH AS IT IS
// THE HIGHEST-LEVEL ABSTRACTION

```
initial
    // we want to apply those stimulus
    // refer timing diagram Page No. 1
begin
    A = 0;  in = 0; // Apply stimulus at time 0.
#20 A = 0;  in = 1; // change inputs at time 20ns
                    // # → indicate time
#20 A = 1;  in = 0; // 40ns, and
#20 A = 1;  in = 1; // 60ns. (commulative time)
                    // if no timing, running simultan.
#40                 // run long enough to test
                    // all possibilities,
$stop;              // and stop
$finish;            // Terminate simulation
end                 // Matching end with begin
endmodule           // Matching end with module
```

—x——x——x—