
Test Benches (Test Fixtures)

Verilog for Testing

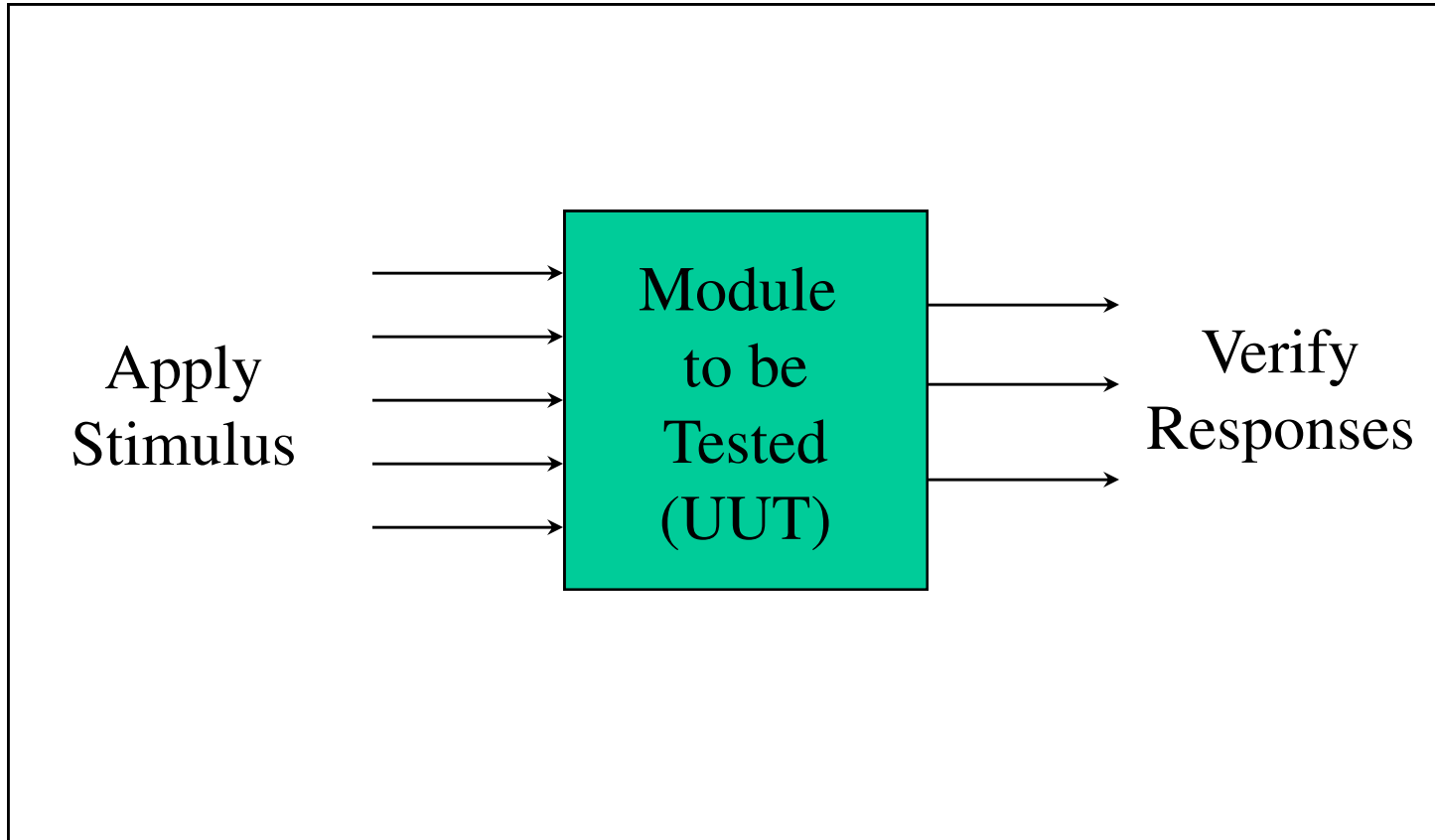
Overview

- We have concentrated on Verilog for synthesis
- Can also use Verilog as a test language
- Very important to conduct comprehensive verification on your design
- To simulate your design you need to produce an additional module that includes your *synthesizable* Verilog design.
 - Usually referred to as a TEST BENCH or TEST FIXTURE
 - Not hardware, just additional Verilog!

Test Bench

- A virtual platform containing the design to be tested (UUT) and virtual wires connected to the UUT inputs and outputs.
- To exercise and verify the correctness of a design to be implemented in hardware
- Has three main purposes
 - to generate stimulus for simulation
 - to apply this stimulus to the module under test and to collect output responses
 - to compare output responses with expected values
- Test Bench should be created by a different engineer than the one who created the synthesizable Verilog

Test Bench – Virtual Platform

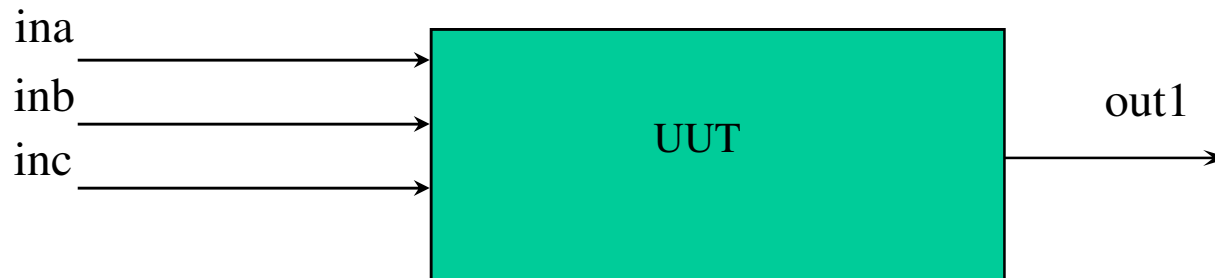


Test Bench Overview

- A Test Bench module consists of
 - Port list has NO ports
 - Instantiate module to be tested (UUT)
 - Declare internal signals to wire to UUT inputs and outputs
 - Verilog statements to provide stimulus and verify UUT responses
 - Designing a test bench that has good coverage can be a very involved project!

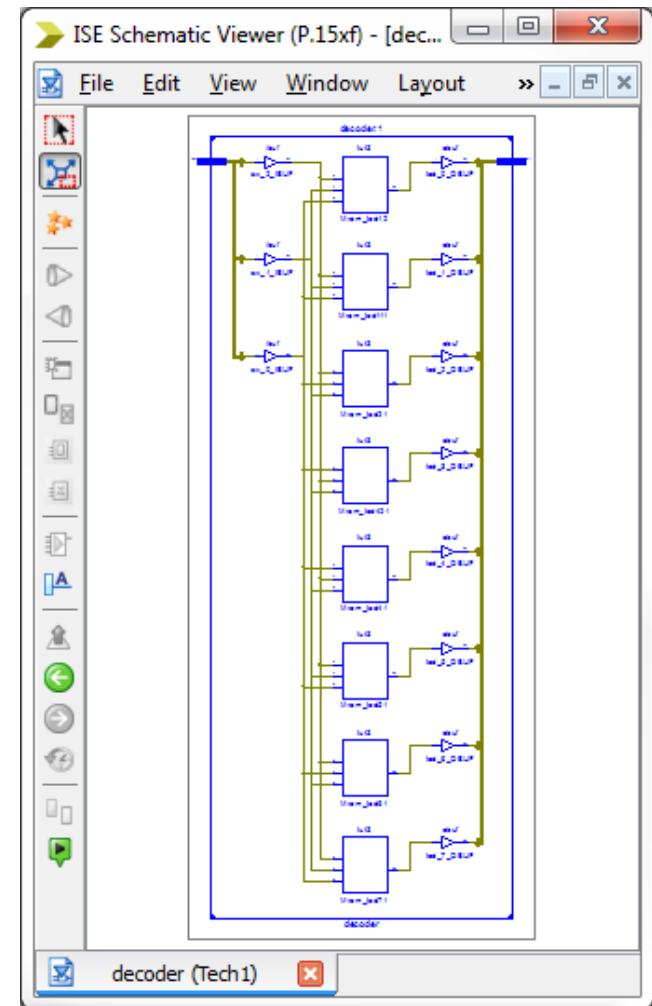
Test Bench - general structure

```
module decoder_tf;  
  
    internal signal declarations  
  
    UUT: test_component instantiation  
  
    signals to generate stimulus  
  
    statements to verify responses  
  
endmodule
```



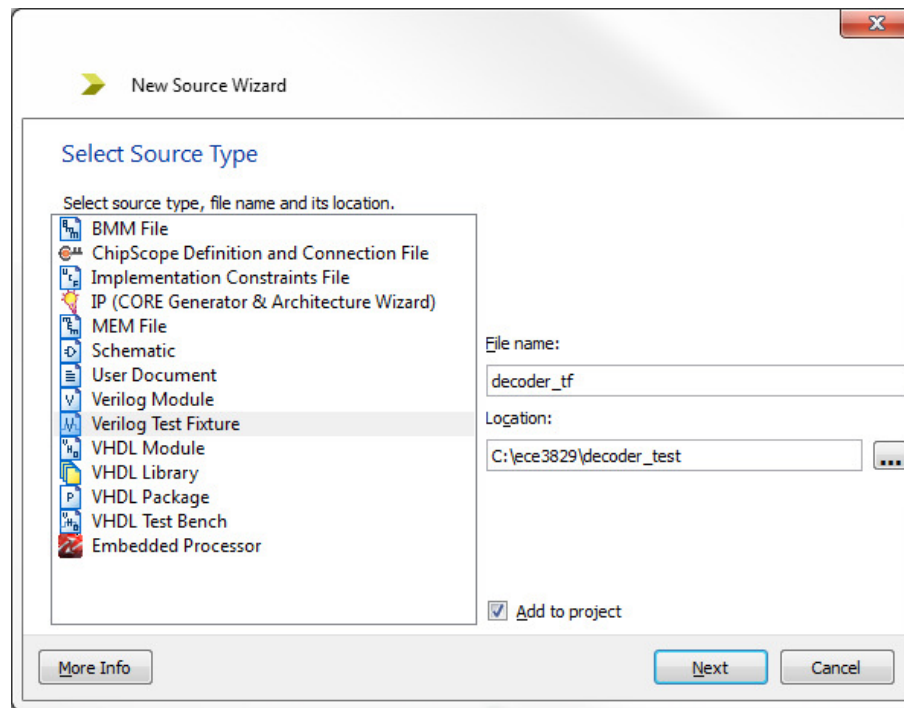
Example Decoder – is the design correct?

```
ISE Text Editor (P.15xf) - [decoder.v]
File Edit View Window Layout Help
21
22 module decoder(
23     input    [2:0] sw,
24     output reg [7:0] led
25 );
26
27 always @ (sw)
28     case (sw)
29         0: led = 8'b00000001;
30         1: led = 8'b00000010;
31         2: led = 8'b00000100;
32         3: led = 8'b00011000; // mistake
33         4: led = 8'b00010000;
34         5: led = 8'b00100000;
35         6: led = 8'b01000000;
36         7: led = 8'b10000000;
37     endcase
38
39 endmodule
decoder.v
```



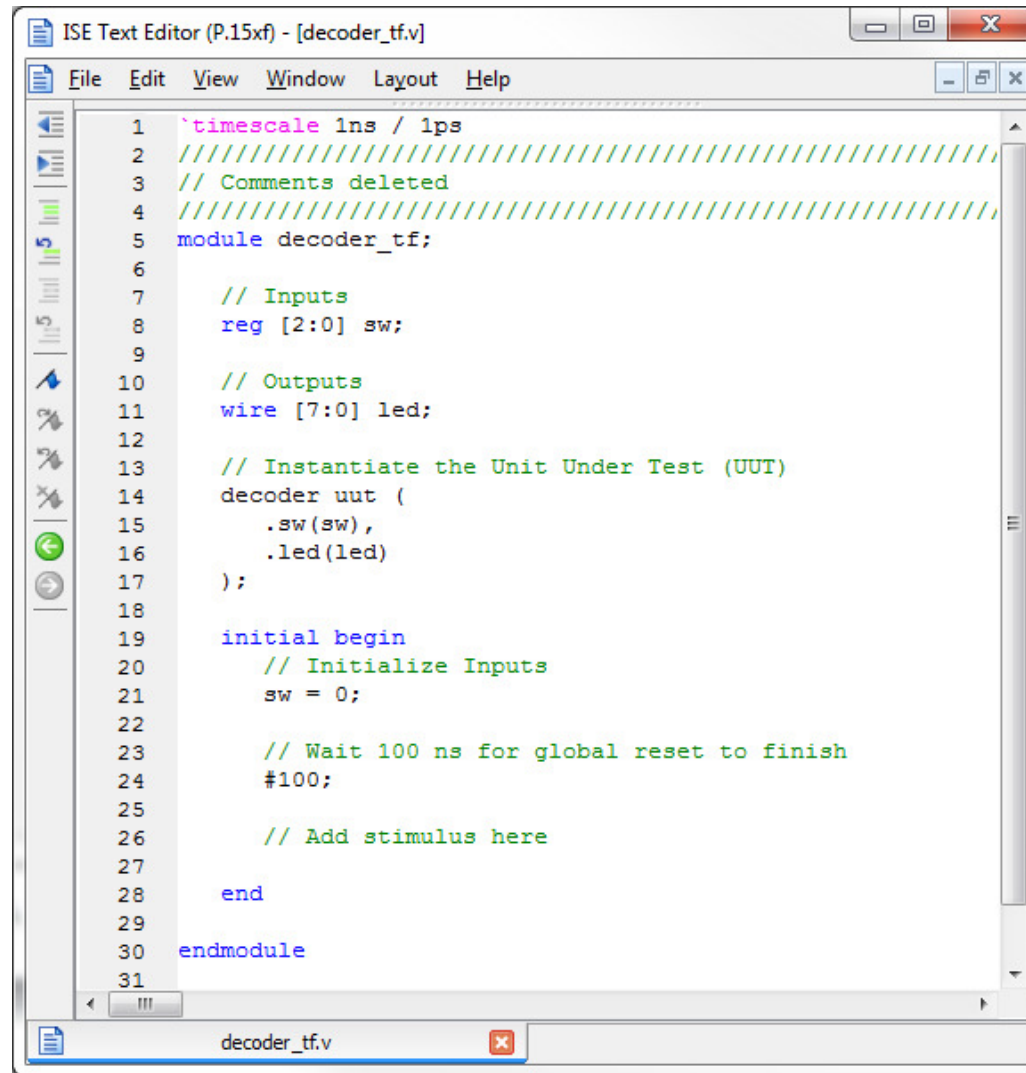
Create Test Fixture

- Select Simulation View
- Select Project => New Source



- Select decoder source to associate with test fixture

Outline of Test Fixture produced

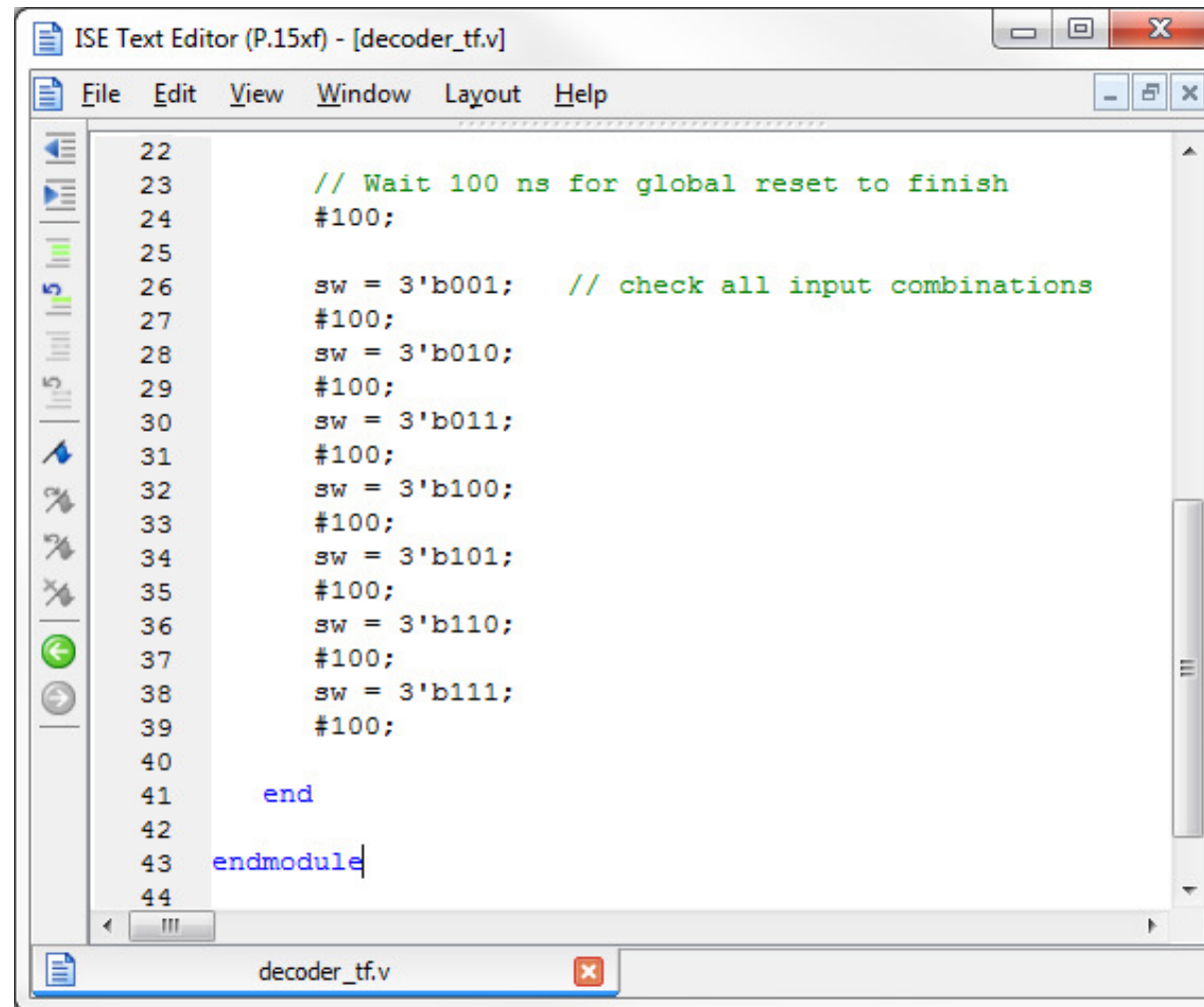


The screenshot shows the ISE Text Editor window titled "ISE Text Editor (P.15xf) - [decoder_tf.v]". The editor displays a Verilog test fixture for a module named "decoder_tf". The code is as follows:

```
1 `timescale 1ns / 1ps
2 //////////////////////////////////////
3 // Comments deleted
4 //////////////////////////////////////
5 module decoder_tf;
6
7     // Inputs
8     reg [2:0] sw;
9
10    // Outputs
11    wire [7:0] led;
12
13    // Instantiate the Unit Under Test (UUT)
14    decoder uut (
15        .sw(sw),
16        .led(led)
17    );
18
19    initial begin
20        // Initialize Inputs
21        sw = 0;
22
23        // Wait 100 ns for global reset to finish
24        #100;
25
26        // Add stimulus here
27
28    end
29
30 endmodule
31
```

The code defines a test fixture for a decoder module. It sets a timescale of 1ns / 1ps, declares a 3-bit input register "sw" and a 8-bit output wire "led". It instantiates the "decoder" module as "uut", connecting "sw" to the input and "led" to the output. The test fixture includes an initial block that initializes "sw" to 0 and waits for 100 ns. The test fixture is enclosed in a "module decoder_tf" block.

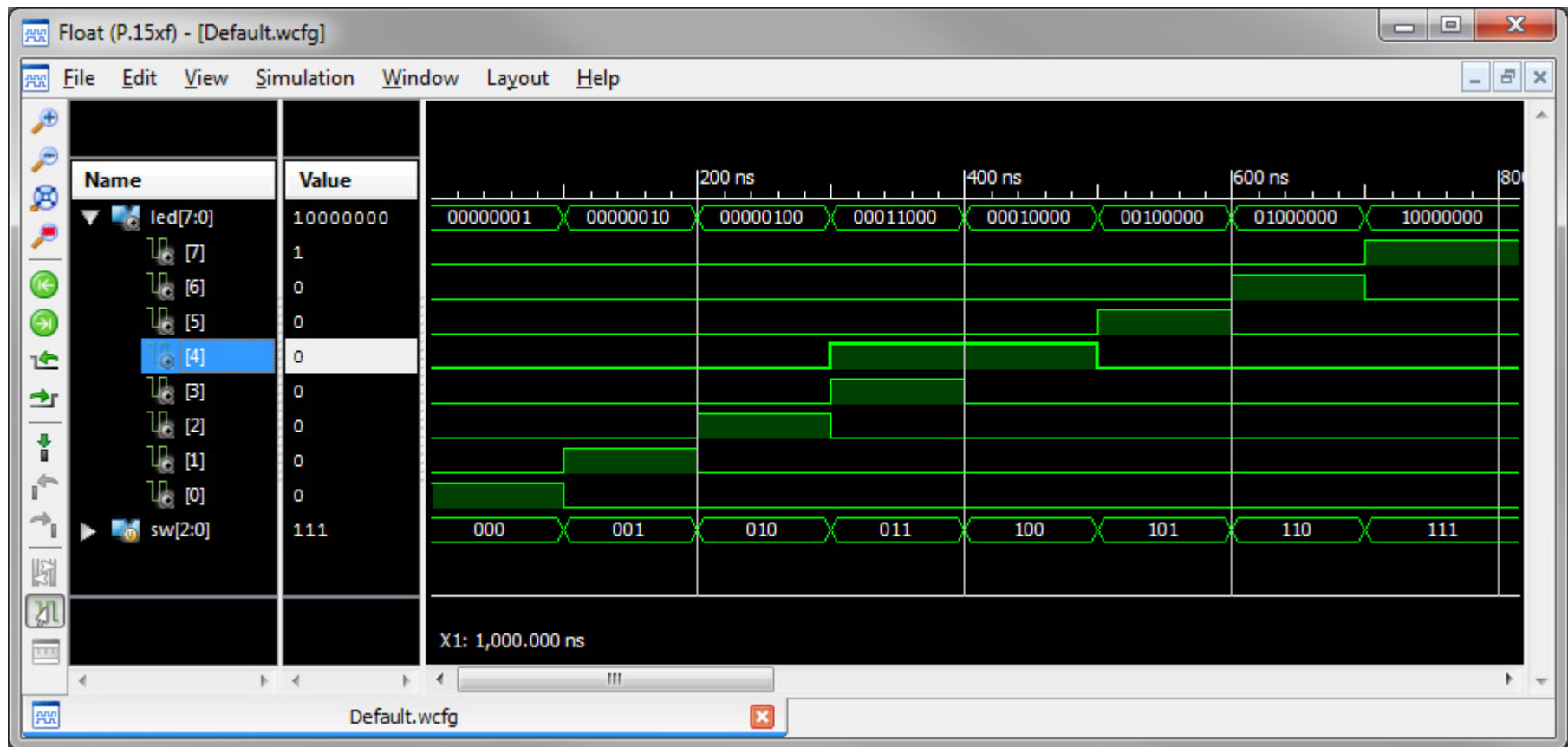
Apply input stimulus



The screenshot shows the ISE Text Editor window titled "ISE Text Editor (P.15xf) - [decoder_tf.v]". The window has a menu bar with "File", "Edit", "View", "Window", "Layout", and "Help". On the left is a vertical toolbar with icons for file operations and editing. The main text area contains Verilog code for a module named "decoder_tf". The code includes comments and assignments for a 3-bit input "sw", with delays of 100 ns between each input combination. The code ends with "end" and "endmodule". The status bar at the bottom shows the file name "decoder_tf.v".

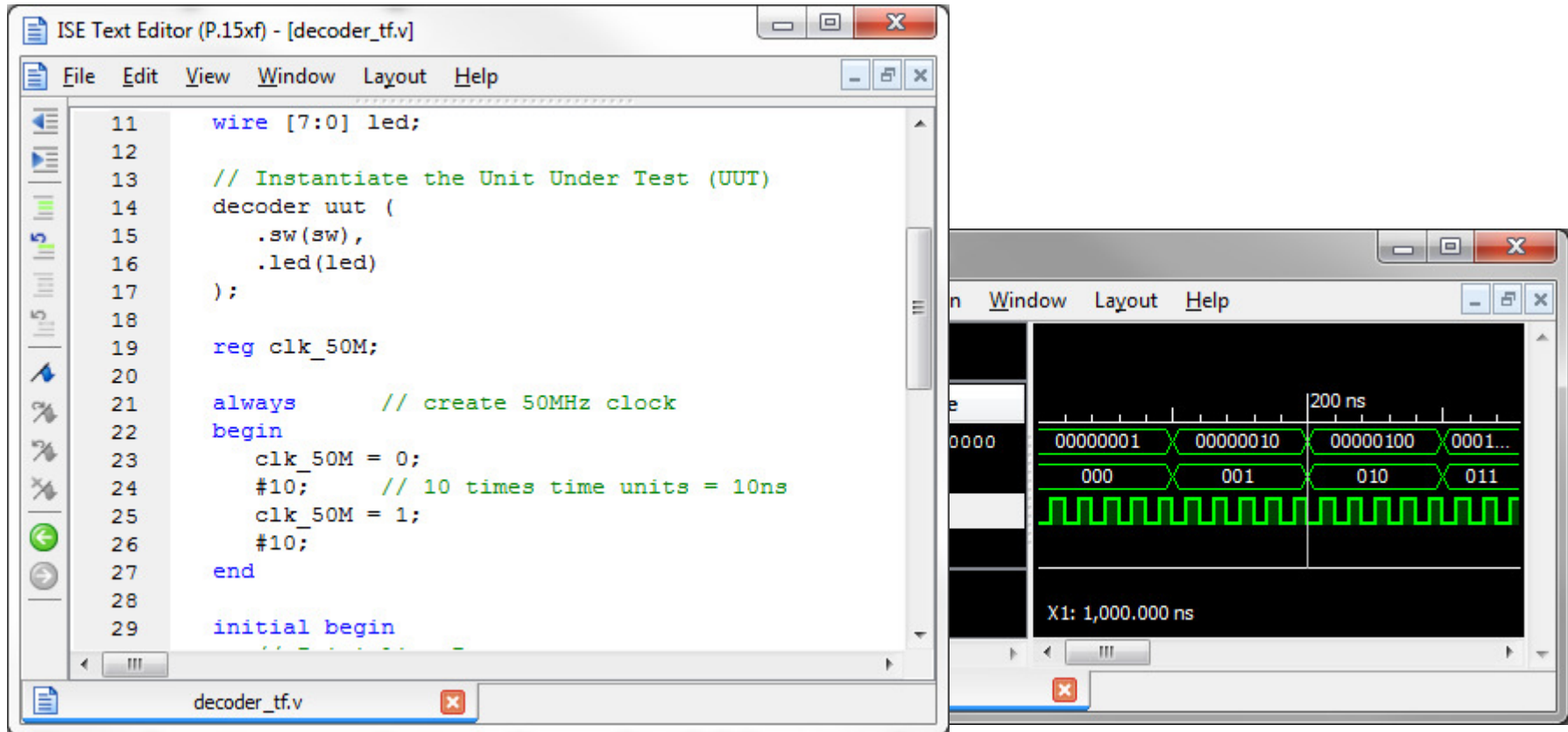
```
22
23     // Wait 100 ns for global reset to finish
24     #100;
25
26     sw = 3'b001;    // check all input combinations
27     #100;
28     sw = 3'b010;
29     #100;
30     sw = 3'b011;
31     #100;
32     sw = 3'b100;
33     #100;
34     sw = 3'b101;
35     #100;
36     sw = 3'b110;
37     #100;
38     sw = 3'b111;
39     #100;
40
41     end
42
43 endmodule
44
```

Simulate Behavioral Model



Testing sequential logic

- Creating clock signal



- Can also 'restart', 'run', and add internal signals to wave