**Verilog code :**

**Workbench**

module decoder\_3X8 (A, Enable, Clock, Y);

input [2:0] A;

input Enable, Clock;

output reg [7:0] Y;

// Initialize output

initial begin

Y = 8'b00000000;

end

always @ (posedge Clock)

begin

if (Enable)

begin

case (A)

3'b000: Y <= 8'b00000001;

3'b001: Y <= 8'b00000010;

3'b010: Y <= 8'b00000100;

3'b011: Y <= 8'b00001000;

3'b100: Y <= 8'b00010000;

3'b101: Y <= 8'b00100000;

3'b110: Y <= 8'b01000000;

3'b111: Y <= 8'b10000000;

default: Y <= 8'b00000000;

endcase

end

else

begin

Y <= 8'b00000000;

end

end

Endmodule

**Testbench:**

`timescale 1ns/1ns

`include "decoder\_3X8\_rtl.v" // includes the module definition for the 3-8 decoder

module testbench;

reg [2:0] A;

reg Enable, Clock;

wire [7:0] Y;

// Instantiate the module under test

decoder\_3X8 dut (.A(A), .Enable(Enable), .Clock(Clock), .Y(Y));

// Clock generation

always #5 Clock = ~Clock; // Clock signal with a period of 10 ns

// Stimulus

initial begin

$fsdbDumpvars();

//Tool specific command. Creates novas.fsdb file. Used for waveform generation

//// Reset inputs

//

A <= 0; Enable <= 0; Clock <= 0;

// Test with Enable = 0 (decoder disabled)

#20 A <= 3'b000; Enable <= 0;

$display("A = %b, Enable = %b, Y = %b", A, Enable, Y);

#20 A <= 3'b101; Enable <= 0;

$display("A = %b, Enable = %b, Y = %b", A, Enable, Y);

// Test all combinations with Enable = 1 (decoder enabled)

#20 A <= 3'b000; Enable <= 1; // Output Y[0] should be high

$display("A = %b, Enable = %b, Y = %b", A, Enable, Y);

#20 A <= 3'b001; Enable <= 1; // Output Y[1] should be high

$display("A = %b, Enable = %b, Y = %b", A, Enable, Y);

#20 A <= 3'b010; Enable <= 1; // Output Y[2] should be high

$display("A = %b, Enable = %b, Y = %b", A, Enable, Y);

#20 A <= 3'b011; Enable <= 1; // Output Y[3] should be high

$display("A = %b, Enable = %b, Y = %b", A, Enable, Y);

#20 A <= 3'b100; Enable <= 1; // Output Y[4] should be high

$display("A = %b, Enable = %b, Y = %b", A, Enable, Y);

#20 A <= 3'b101; Enable <= 1; // Output Y[5] should be high

$display("A = %b, Enable = %b, Y = %b", A, Enable, Y);

#20 A <= 3'b110; Enable <= 1; // Output Y[6] should be high

$display("A = %b, Enable = %b, Y = %b", A, Enable, Y);

#20 A <= 3'b111; Enable <= 1; // Output Y[7] should be high

$display("A = %b, Enable = %b, Y = %b", A, Enable, Y);

#100 $finish;

end

Endmodule  
  
**Dc\_run.tcl :**

source -echo -verbose ./rm\_setup/dc\_setup.tcl

set RTL\_SOURCE\_FILES ./../rtl/decoder\_3X8\_rtl.v

define\_design\_lib WORK -path ./WORK

#set\_dont\_use [get\_lib\_cells \*/FADD\*]

#set\_dont\_use [get\_lib\_cells \*/HADD\*]

set\_dont\_use [get\_lib\_cells \*/AO\*]

#set\_dont\_use [get\_lib\_cells \*/OA\*]

set\_dont\_use [get\_lib\_cells \*/NAND\*]

set\_dont\_use [get\_lib\_cells \*/XOR\*]

#set\_dont\_use [get\_lib\_cells \*/NOR\*]

#set\_dont\_use [get\_lib\_cells \*/XNOR\*]

#set\_dont\_use [get\_lib\_cells \*/MUX\*]

analyze -format verilog ${RTL\_SOURCE\_FILES}

elaborate ${DESIGN\_NAME}

current\_design

read\_sdc ./../CONSTRAINTS/decoder\_3X8.sdc

#compile

compile\_ultra

report\_qor

report\_timing

write -format verilog -hierarchy -output ${RESULTS\_DIR}/${DCRM\_FINAL\_VERILOG\_OUTPUT\_FILE}

**Floorplan:**

open\_lib DECODER\_3X8\_LIB

set PDK\_PATH ./../ref

#create\_lib -ref\_lib $PDK\_PATH/lib/ndm/saed32rvt\_c.ndm DECODER\_3X8\_LIB

read\_verilog {./../DC/results/decoder\_3X8.mapped.v} -library DECODER\_3X8\_LIB -design decoder\_3X8 -top decoder\_3X8

#open the lib and block after saving

#scenerio 6

initialize\_floorplan -shape L -core\_offset 2 -coincident\_boundary true

set\_individual\_pin\_constraints -ports [get\_ports {A B}] -sides 6

place\_pins -self

create\_placement -floorplan -effort high

#save\_lib

#save\_block

#open\_block DECODER\_3X8\_LIB

#open\_lib <DECODER\_3X8\_LIB>

#open\_block <block name>

#

####################################End of Floorplan Scenarios###############################

Clock constraint is 2.5 ns