|  |  |  |  |
| --- | --- | --- | --- |
| Họ tên sinh viên | MSSV | Lớp (thứ - tiết) |  |
| Hoàng Ngọc Dung | 23139006 | Lớp thứ bảy tiết 7-9 |

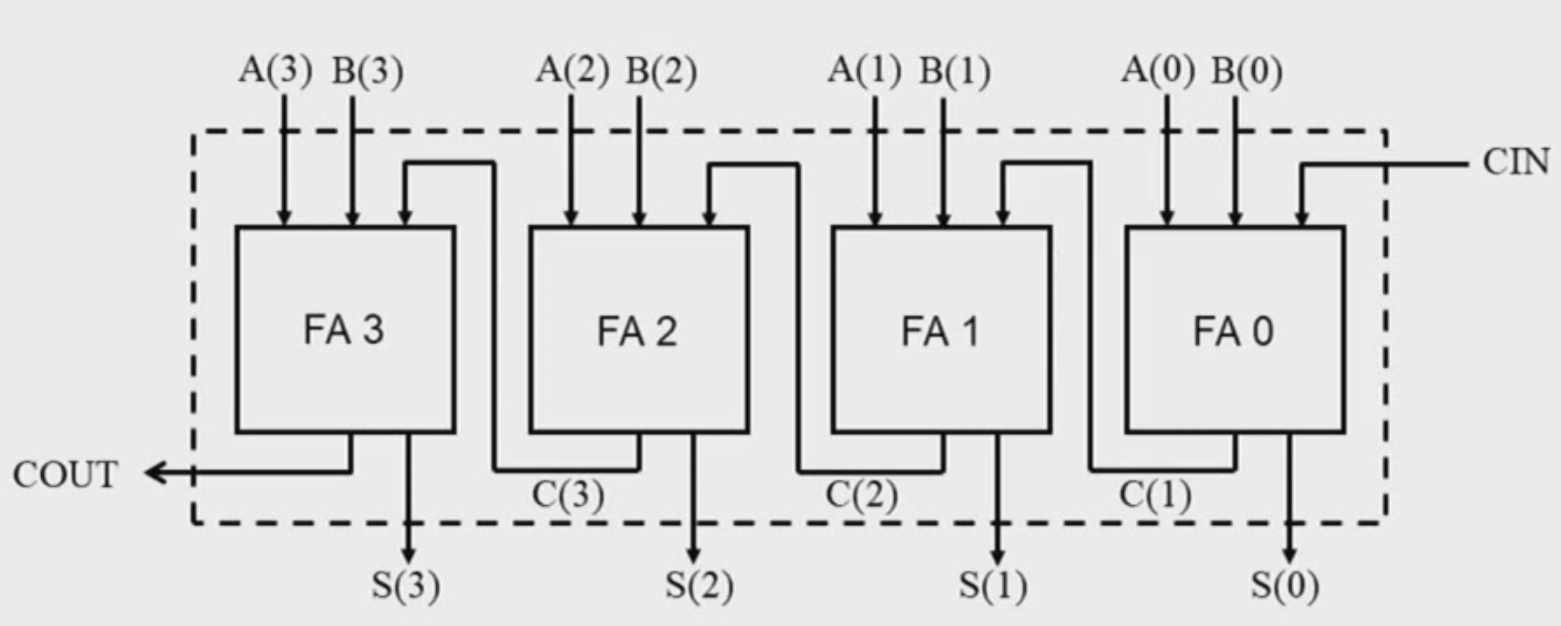
Chú ý: Sinh viên thay bằng QR code của mã số sinh viên (ví dụ: 23119012), có thể tham khảo tại https://barcode.tec-it.com)

Quick question : chapter 04

*Lưu ý: Trong mỗi thiết kế yêu cầu sinh viên thực hiện*

* *Sơ đồ khối (nguyên lý, cấu trúc)*
* *Bảng trạng thái*
* *Mô tả bằng ngôn ngữ Verilog cho module cần thiết kế,*
* *Mô tả Verilog cho module dùng để kiểm tra thiết kế*
* *Kết quả mô phỏng quá trình kiểm tra, có phân tích*
* *Module test được đặt tên theo cú pháp: tensv\_testbench\_tenmodule, ví dụ để test module encoder, sinh viên Nguyen Van An phải đặt tên module test như sau: An\_testbench\_encoder. Các kết quả mô phỏng phải được chụp màn hình bao gồm cả tên của module test trong đó có tên sinh viên thì mới hợp lệ*

1. Thiết kế và mô phỏng kiểm chứng mạch cộng 4 bít từ mạch cộng toàn phần 1 bit, sử dụng mô tả cấu trúc.



Hình : Mạch cộng toàn phần 4 bit

Bảng trạng thái

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| A | B | Cin | Sum | Carry |
| 0 | 0 | 1 | 0 | 1 |
| 0 | 1 | 0 | 0 | 1 |
| 0 | 1 | 1 | 1 | 0 |
| 1 | 0 | 0 | 0 | 1 |
| 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 |

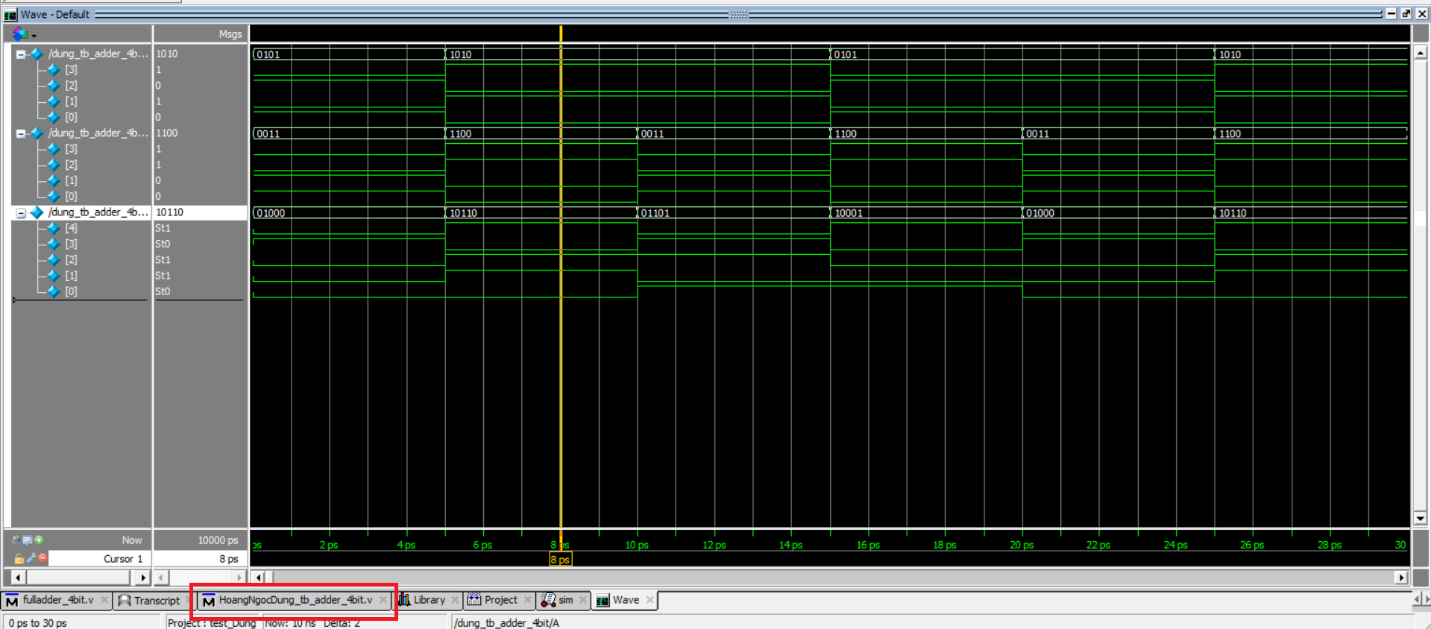
Mô tả bằng ngôn ngữ Verilog

|  |
| --- |
| module full\_adder(      input wire a, input wire b,input wire cin,      output wire sum, output wire carry  );  assign sum = a ^ b ^ cin;  assign carry = (a & b) | (a & cin) | (b & cin);  endmodule  module adder\_4bit(      input wire [3:0] A,      input wire [3:0] B,      output wire [4:0] R  );  // carry signals  wire c1, c2, c3;  full\_adder add0 (A[0], B[0], 0 , R[0], c1);  full\_adder add1 (A[1], B[1], c1, R[1], c2);  full\_adder add2 (A[2], B[2], c2, R[2], c3);  full\_adder add3 (A[3], B[3], c3, R[3], R[4]);  endmodule |

Mô tả Verilog cho module dùng để kiểm tra thiết kế

|  |
| --- |
| module dung\_tb\_adder\_4bit();  reg [3:0] A;  reg [3:0] B;  wire [4:0] R;  initial begin      A = 4'b0101; B = 4'b0011;  end  always forever #5 A = ~A;  always forever #5 B = ~B;  always forever #10 A = ~A;  adder\_4bit uut (A,B,R);  endmodule |

Kết quả mô phỏng

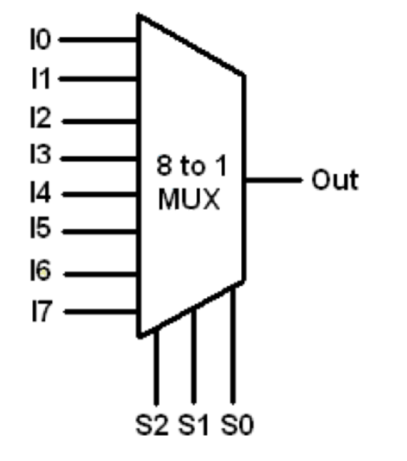


Hình : Hình ảnh sóng cho thấy sự thay đổi của các tín hiệu A, B, và R theo thời gian dựa vào module thiết kế

* Thời điểm ban đầu (Now): A có giá trị 0101 (5). B có giá trị 0011 (3), R có giá trị 01000 (8), là kết quả của phép cộng 5 + 3.
* Tại thời điểm 5 ps: A thay đổi thành 1010 (10), B thay đổi thành 1100 (12), R thay đổi thành 10110 (22), là kết quả của phép cộng 10 + 12.
* Tại thời điểm 10 ps: A thay đổi trở lại thành 0101 (5), B thay đổi trở lại thành 0011 (3), R thay đổi trở lại thành 01000 (8), là kết quả của phép cộng 5 + 3.
* Tại thời điểm khoảng 15 ps: A thay đổi thành 1010 (10), B thay đổi thành 1100 (12), R thay đổi thành 10110 (22), là kết quả của phép cộng 10 + 12.

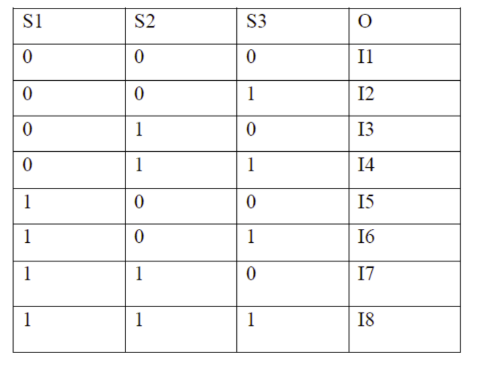
Vậy mạch cộng toàn phần 4 bit hoạt động chính xác

1. Thiết kế và mô phỏng kiểm chứng mạch đa hợp 8 sang 1



Hình 3: Sơ đồ khối mạch đa hợp 8 sang 1

Bảng trạng thái



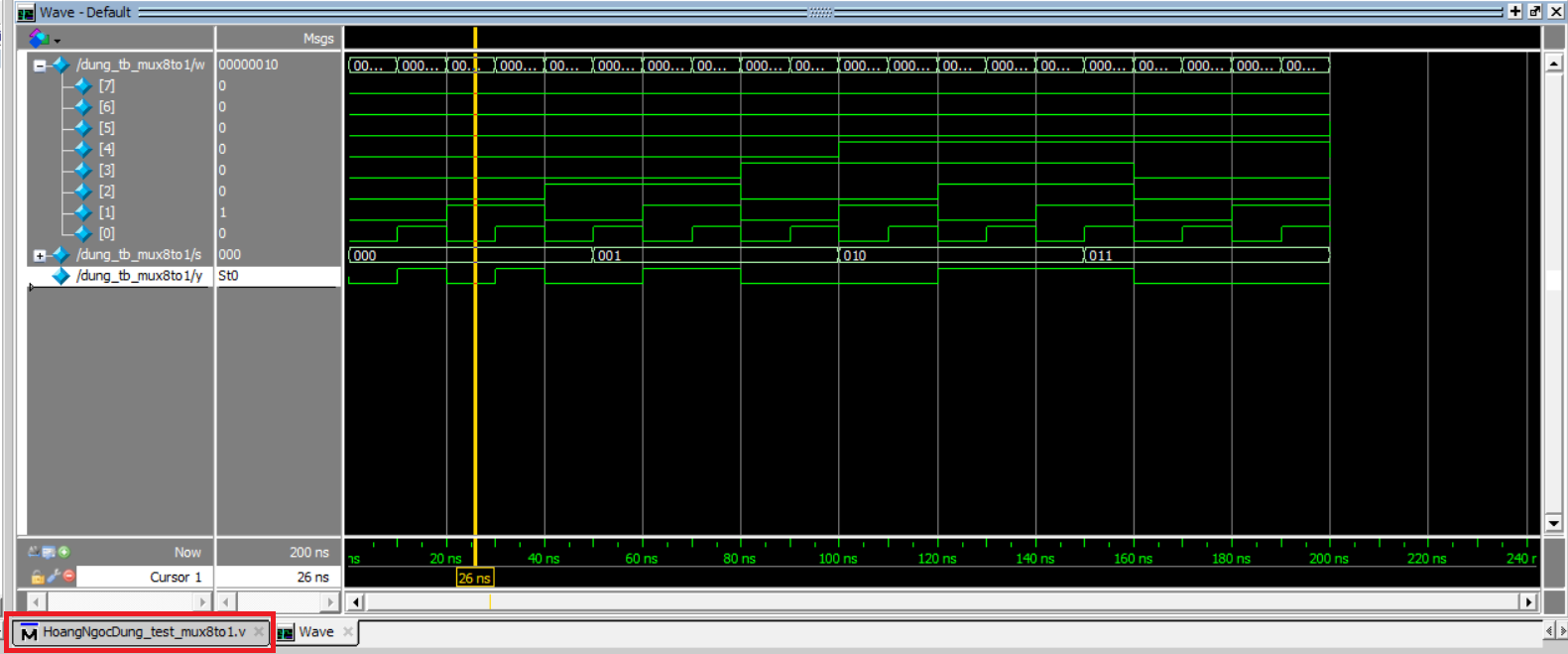
Mô tả bằng ngôn ngữ Verilog

|  |
| --- |
| module mux8to1(  input wire [7:0] w,  input wire [2:0] s,  output reg y);  always @(w,s)  case(s)  0: y = w[0];  1: y = w[1];  2: y = w[2];  3: y = w[3];  4: y = w[4];  5: y = w[5];  6: y = w[6];  default: y = w[7];  endcase  endmodule |

Mô tả Verilog cho module dùng để kiểm tra thiết kế

|  |
| --- |
| `timescale 1ns/1ns // define timescale  module dung\_tb\_mux8to1();  reg [7:0] w;  reg [2:0] s;  wire y;  // initialize w and s to 000 and 00, respectively  initial begin  w = 0 ;  s = 0;  end  // generate w0- w3  always forever #10 w[0] = ~w[0];  always forever #20 w[1] = ~w[1];  always forever #40 w[2] = ~w[2];  always forever #80 w[3] = ~w[3];  always forever #100 w[4] = ~w[4];  always forever #200 w[5] = ~w[5];  always forever #400 w[6] = ~w[6];  always forever #800 w[7] = ~w[7];  //generate 4 state of s by increasing s by 1  always forever #500 s = s + 1;  // conect w,s,y to moduke mux41  mux8to1 m0(w,s,y);  endmodule |

Kết quả mô phỏng



Hình 4: Tín hiệu s từ tín hiệu này tăng dần giá trị từ 000 (0) đến 001 (1), 010 (2), 011 (3), và tiếp tục đến 111 (7) rồi quay lại 000 và ta thấy :

- Khi s = 000, y bằng w[0].

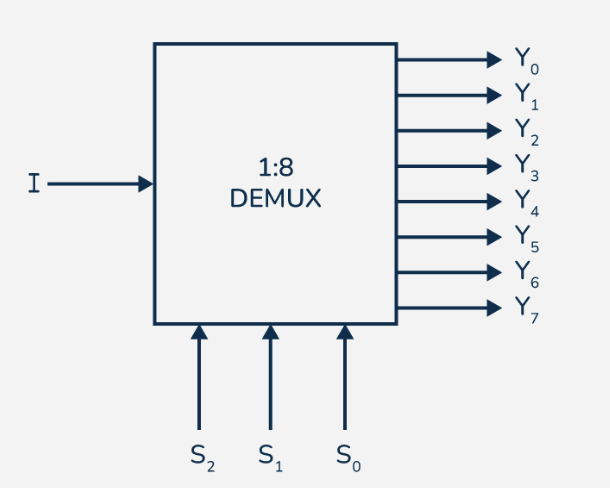
- Khi s = 001, y bằng w[1].

- Khi s = 010, y bằng w[2].

- Khi s = 011, y bằng w[3]....

Vậy mạch đa hợp 8 sang 1 hoạt dộng chính xác

1. Thiết kế và mô phỏng kiểm chứng mạch giải đa hợp 1 sang 8



Hình 4: Sơ đồ mạch giải đa hợp 1 sang 8

Bảng trạng thái

| Selection Inputs | | | Outputs | | | | | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| S2 | S1 | S0 | Y7 | Y6 | Y5 | Y4 | Y3 | Y2 | Y1 | Y0 |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | I |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | I | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | I | 0 | 0 |
| 0 | 1 | 1 | 0 | 0 | 0 | 0 | I | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | I | 0 | 0 | 0 | 0 |
| 1 | 0 | 1 | 0 | 0 | I | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | I | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | I | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

Mô tả bằng ngôn ngữ Verilog

|  |
| --- |
| module mux8to1(  input wire [7:0] w,  input wire [2:0] s,  output reg y);  always @(w,s)  case(s)  0: y = w[0];  1: y = w[1];  2: y = w[2];  3: y = w[3];  4: y = w[4];  5: y = w[5];  6: y = w[6];  default: y = w[7];  endcase  endmodule |

Mô tả Verilog cho module dùng để kiểm tra thiết kế

|  |
| --- |
| `timescale 1ns/1ns // define timescale  module dung\_tb\_demux1to8();  reg in;  reg [2:0] sel;  wire [7:0] out;  initial begin  in = 1;  sel = 0;  end  // generate sel0- sel2  always forever #10 sel[0] = ~sel[0];  always forever #20 sel[1] = ~sel[1];  always forever #40 sel[2] = ~sel[2];  always forever #80 in = in + 1;  demux1to8 m1(in,sel,out);  endmodule |

Kết quả mô phỏng



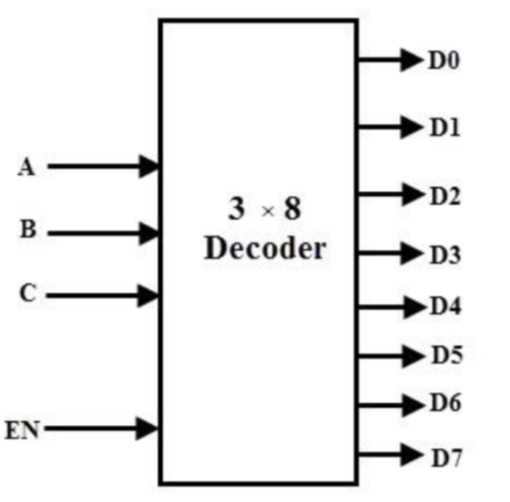
Hình 5: Ta cấp tín hiệu in = 1 (mức cao) và thayddoi sau mỗi 80ns

Tín hiệu sel từ 000 và đảo bit sau mỗi thay đổi sau mỗi 10ns , 20ns, 40ns

Ta thấy Khi sel = 000, thì out[0], sel = 001, thì out[1]. tại 24ns sel = 010 thì out = [2]

Vậy mạch giải đa hợp 1 sang 8 hoạt động chính xác

1. Thiết kế và mô phỏng mạch giả mã 3 sang 8 có tín hiệu cho phép (enable –EN) ngõ ra tích cực mức thấp



Hình 6: tín hiệu cho phép mức cao (EN = 1) của mạch giải mã

Bảng trạng thái

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | | | Output | | | | | | | |
| En | A | B | C | Y7 | Y6 | Y5 | Y4 | Y3 | Y2 | Y1 | Y0 |
| 1 | x | x | x | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 |
| 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 |
| 0 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 |
| 0 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 0 | 1 | 1 | 1 |
| 0 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 |
| 0 | 1 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 |
| 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | 1 | 1 |

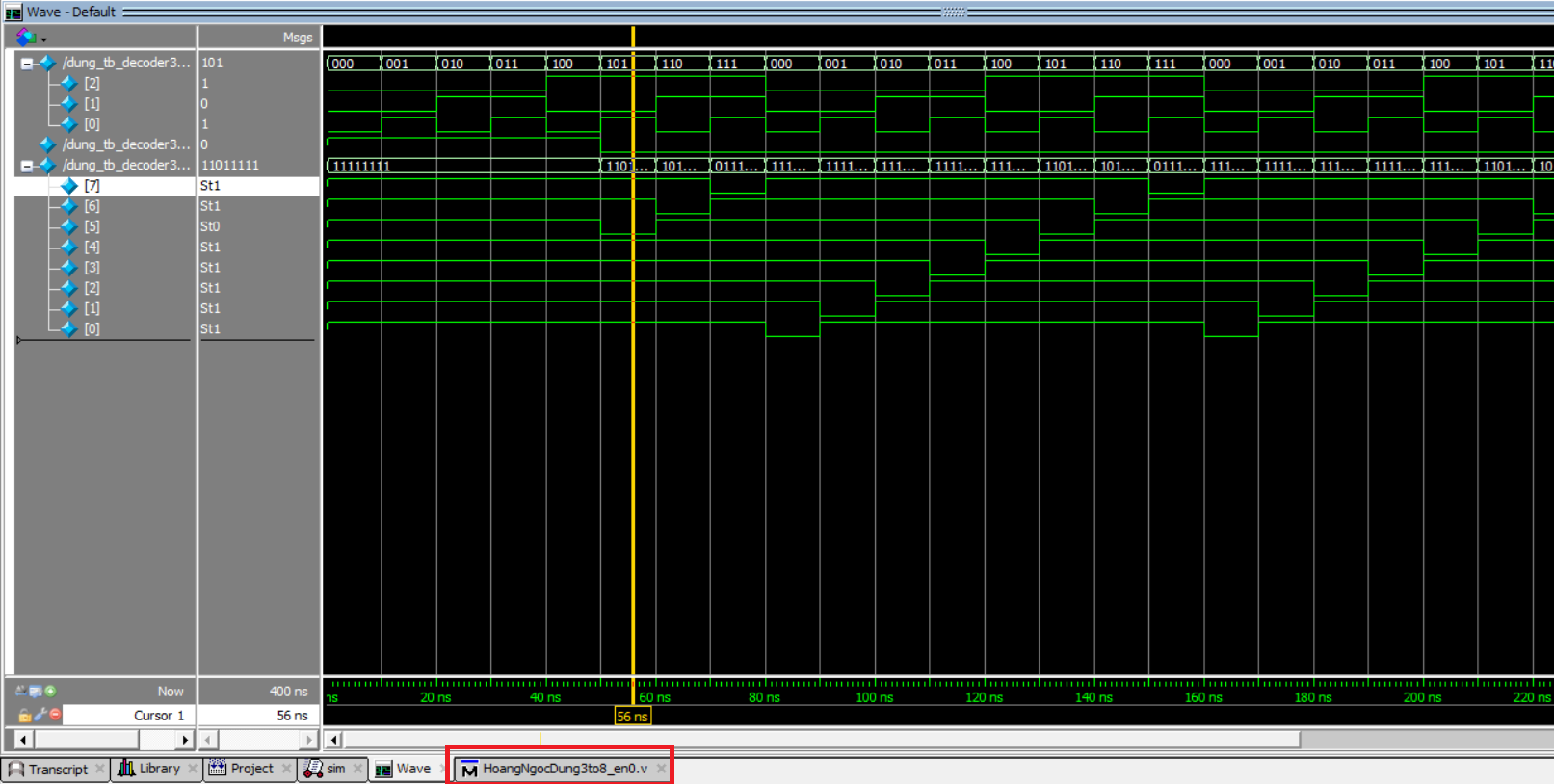
Mô tả bằng ngôn ngữ Verilog

|  |
| --- |
| module decoder3to8\_en0(  input wire [2:0] in,  input wire  en,  output reg [7:0] out);  always @(in or en)      if (en == 0)          out = ~(8'b00000001 << in); // dịch bit trái      else              out = 8'b11111111;  endmodule |

Mô tả Verilog cho module dùng để kiểm tra thiết kế

|  |
| --- |
| `timescale 1ns/1ns // define timescale  module dung\_tb\_decoder3to8\_en0();  reg [2:0] in;  reg en;  wire [7:0] out;  initial begin  en = 1;  in = 0;  end  always forever #10 in[0] = ~in[0];  always forever #20 in[1] = ~in[1];  always forever #40 in[2] = ~in[2];  always forever #50 en = 0;  decoder3to8\_en0 m3(in,en,out);  endmodule |

Kết quả mô phỏng



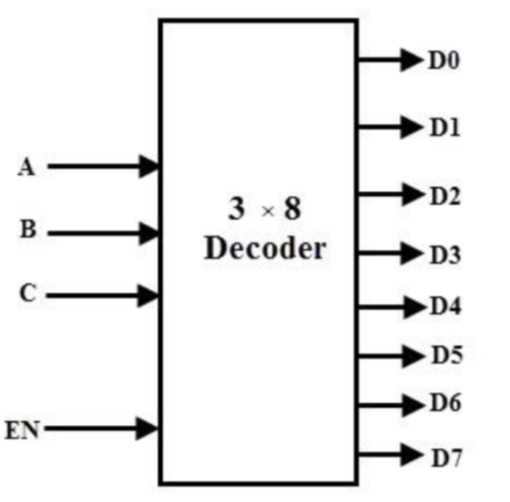
Hình 8: Hình ảnh sóng mô tả ban đầu cấp tín hiệu enable = 1 , in = 0 và sel[0] thay đổi sau mỗi 10ns, sel[1] thay đổi sau mỗi 20n, sel[2] thay đổi sau mỗi 40ns.

- Trước 50ns , en = 1 nên out = 0 bất kể input

- Sau 50ns, en = 0 ta thấy in = 010, thì out[2] = 0, các bit còn lại của out bằng 1. Tương tự in = 101 thì out[3] = 0 ,..

Vậy mạch giả mã 3 sang 8 có tín hiệu cho phép (enable –EN) ngõ ra tích cực mức thấp hoạt động chính xác

1. Thiết kế và mô phỏng mạch giả mã 3 sang 8 có tín hiệu cho phép (enable –EN), ngõ ra tích cực mức cao



Hình 9: Tín hiệu cho phép mức cao (EN = 1) của mạch giải mã 3 sang 8

Bảng trạng thái

|  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Input | | | | Output | | | | | | | |
| En | A | B | C | Y7 | Y6 | Y5 | Y4 | Y3 | Y2 | Y1 | Y0 |
| 0 | x | x | x | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 |
| 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

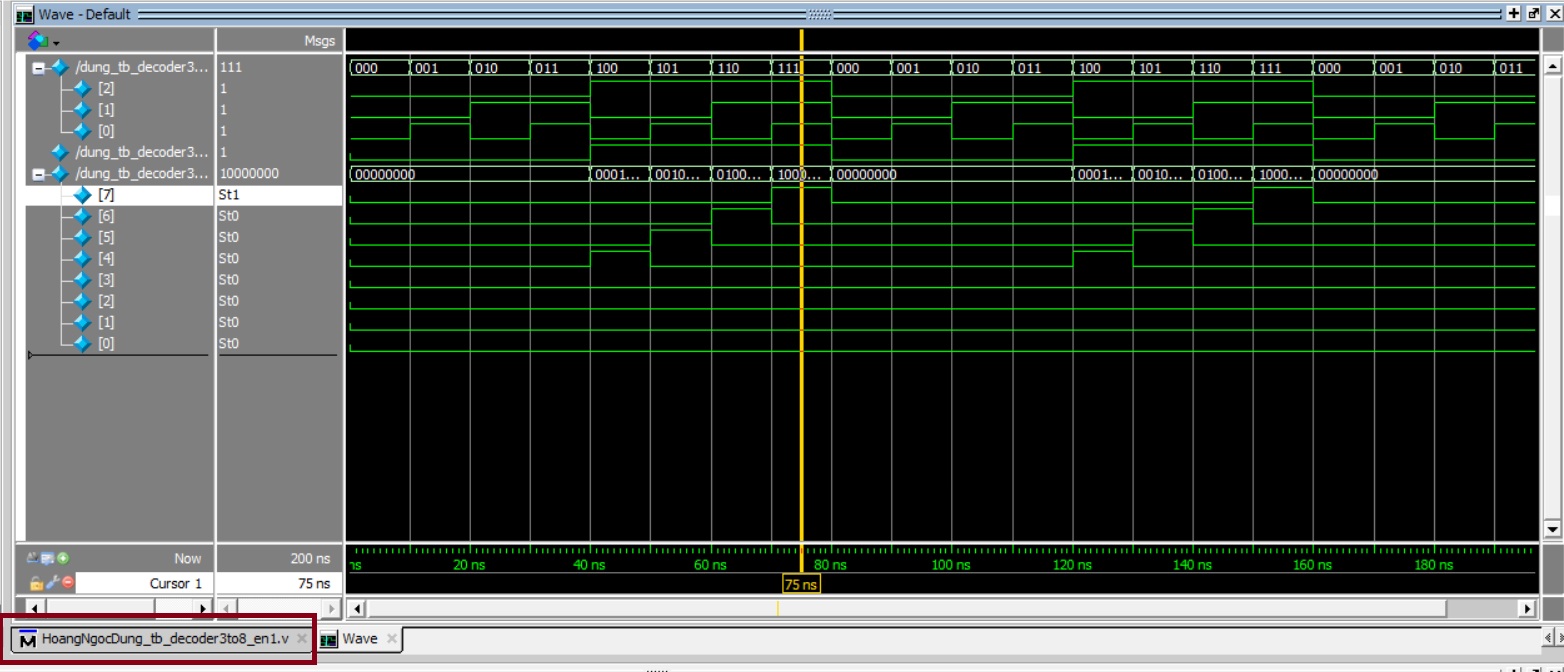
Mô tả bằng ngôn ngữ Verilog

|  |
| --- |
| module decoder3to8\_en1(  input wire [2:0] in,  input wire  en,  output reg [7:0] out);  always @(in or en)      if (en == 1)          out = (8'b00000001 << in);      else              out = 8'b00000000;  endmodule |

Mô tả Verilog cho module dùng để kiểm tra thiết kế

|  |
| --- |
| `timescale 1ns/1ns // define timescale  module dung\_tb\_decoder3to8\_en1();  reg [2:0] in;  reg en;  wire [7:0] out;  initial begin  en = 0;  in = 0;  end  always forever #10 in[0] = ~in[0];  always forever #20 in[1] = ~in[1];  always forever #40 in[2] = ~in[2];  always forever #50 en = ~en;  decoder3to8\_en1 m2(in,en,out);  endmodule |

Kết quả mô phỏng



Hình 9: Hình ảnh sóng mô tả ban đầu cấp tín hiệu enable = 0 , in = 0 và sel[0] thay đổi sau mỗi 10ns, sel[1] thay đổi sau mỗi 20n, sel[2] thay đổi sau mỗi 40ns.

- Trước 50ns , en = 0 nên out = 0 bất kể input

- Sau 50ns, en = 1 ta thấy in = 110, thì out[6] = 1, các bit còn lại của out bằng 0. Tương tự in = 111 thì out[7] = 0 ,..

Vậy mạch giả mã 3 sang 8 có tín hiệu cho phép (enable –EN) ngõ ra tích cực mức cao hoạt động chính xác