* Full Adder (4 bits)

|  |  |  |
| --- | --- | --- |
| Model | RTL | Synthesis |
| Dataflow | *module* full\_adder\_4bit\_df(*input* [3:0] a, b,  *input* cin, *output* [3:0]s,  *output* cout);  *assign* {cout,s} *=* a *+* b *+* cin;  *endmodule* |  |
| Structural | *module full\_adder\_4bit\_st(input [3:0] a, b,*  *input cin, output [3:0]s,*  *output cout);*  *wire [2:0] carrys;*  *full\_adder fa1(a[0],b[0],cin,s[0],carrys[0]);*  *full\_adder fa2(a[1],b[1],carrys[0],s[1],carrys[1]);*  *full\_adder fa3(a[2],b[2],carrys[1],s[2],carrys[2]);*  *full\_adder fa4(a[3],b[3],carrys[2],s[3],cout);*  *endmodule* |  |
| Behavioral | *module* full\_adder\_4bit\_bh(*input* [3:0] a, b,  *input* cin, *output* *reg* [3:0]s,  *output* *reg* cout);  *always* @(*\**) *begin*      {cout,s} *=* a *+* b *+* cin;  *end*  *endmodule* |  |

Testbench for Full Adder

|  |  |
| --- | --- |
| *module* full\_adder\_4bit\_DUT;  *reg* [3:0] a, b;  *reg* cin;  *wire* [3:0]s;  *wire* cout;  full\_adder\_4bit\_bh uut (a,b,cin,s,cout);  *//full\_adder\_4bit\_df uut (a,b,cin,s,cout);*  *//full\_adder\_4bit\_st uut (a,b,cin,s,cout);*  *initial* *begin*      a *=* 0; b *=* 0; cin *=* 0;  *repeat* (16) *begin*          #10 a *=* a *+* 1;  *repeat* (16) *begin*              #10 b *=* b *+* 1; | *repeat* (2) *begin*                #10 cin *=* *~* cin;  *end*  *end*  *end*      $stop;  *end*  *initial* *begin*      $monitor ("a = %d\tb = %d\tcin = %d\tsum = %b\tcarry = %d",a,b,cin,s,cout);  *end*  *endmodule* |

* Multiplexer (2x1)

|  |  |  |
| --- | --- | --- |
| Model | RTL | Synthesis |
| Dataflow | *module* mux\_2to1\_df(*input* [1:0] I, *input* sel,  *output* y);  *assign* y *=* sel? I[1] : I[0];  *endmodule* |  |
| Behavioral | *module* mux\_2to1\_bh(*input* [1:0] I,*input* sel,  *output* *reg* y);  *always* @(*\**) *begin*  *if* (sel) *begin*          y *=* I[1];  *end*  *else* *begin*          y *=* I[0];  *end*  *end*  *endmodule* |  |

* Multiplexer (4x1)

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | RTL | | | |
| Dataflow | *module* mux\_4to1\_df(*input* [3:0] I, *input* [1:0] sel,  *output* y);  *assign* y *=* (sel *==* 2'b00)? I[0] : (sel *==* 2'b01)? I[1] :           (sel *==* 2'b10)? I[2] : I[3];  *endmodule* | *module* mux\_4to1\_df3(*input* [3:0] I, *input* [1:0] sel,  *output* y);  *assign* y *=* *~|*sel? I[0] :  *&* sel? I[3] :            sel[0]? I[1] : I[2];  *endmodule* | | *module* mux\_4to1\_df2(*input* [3:0] I, *input* [1:0] sel,  *output* y);  *assign* y *=* I[sel];  *endmodule* |
| Structural | *module* mux\_4to1\_st(*input* [3:0] I, *input* [1:0] sel,  *output* y);  *wire* [1:0] net;  mux\_2to1\_df mux1(I[1:0],sel[0],net[0]);  mux\_2to1\_df mux2(I[3:2],sel[0],net[1]);  mux\_2to1\_df mux3(net,sel[1],y);  *endmodule* | | | |
| Behavioral | *module* mux\_4to1\_bh(*input* [3:0] I, *input* [1:0] sel,  *output* *reg* y);  *always* @(*\**) *begin*  *case* (sel)          2'b00: y *=* I[0];          2'b01: y *=* I[1];          2'b10: y *=* I[2]; | | 2'b11: y *=* I[3];  *default*: y *=* 0;  *endcase*  *end*  *endmodule* | |

|  |  |  |  |
| --- | --- | --- | --- |
| Model | Synthesis | | |
| Dataflow |  |  |  |
| Structural |  | | |
| Behavioral |  | | |

* Decoder (2x4)

|  |  |  |
| --- | --- | --- |
| Model | RTL | Synthesis |
| Dataflow | *module decoder\_2to4\_df(input en, input [1:0] I, output [3:0] y);*  *assign y = ~en? 4'b0 : {(I[0] & I[1]),*  *(I[0] & ~I[1]),*  *(~I[0] & I[1]),*  *(~I[0] & ~I[0])};*  *endmodule* |  |
| Behavioral | *module decoder\_2to4\_bh(input en, input [1:0] I, output reg [3:0] y);*  *always @(\*) begin*  *case ({en,I})*  *3'b100: y = 4'b0001;*  *3'b101: y = 4'b0010;*  *3'b110: y = 4'b0100;*  *3'b111: y = 4'b1000;*  *3'b000,3'b001,3'b010,*  *3'b011: y = 4'b0000;*  *default: $display ("error");*  *endcase*  *end*  *endmodule* |  |