Test Document

برای تست این ماژول ابتدا instance های مورد نیاز را از دو ماژول تعریف شده در قسمت Design، تعریف کرده و ورودی و خروجی های هر یک تنظیم گردید.

module TB;

    wire Process;

    wire Dir;

    wire [4:0] PushedButtons;

    reg clk;

    reg [4:0] InToggle;

    reg [4:0] OutToggle;

    wire [4:0] WhichFloor;

    wire [1:0] FloorInOut [4:0];

    Asansor asansor (

        .Clk(clk),

        .State({Dir, Process}),

        .WhichFloor(WhichFloor),

        .FloorInOut(FloorInOut)

    );

    Handler handler (

        .clk(clk),

        .WhichFloor(WhichFloor),

        .InToggle(InToggle),

        .OutToggle(OutToggle),

        .Dir(Dir),

        .Process(Process),

        .PushedButtons(PushedButtons)

    );

پس از آن در بلاک Always، سیگنال کلاک به ازای واحد زمانی 10، مقداردهی شد.

دو task در این ماژول تعریف شده است که یکی برای دریافت فشرده شدن دکمه های درون و دیگری برای دکمه های بیرونی است که پس از گذشت مقدار delay\_time دکمه آن طبقه، غیر فعال میشود.

    always #10 clk = ~clk;

    task handle\_in\_toggle(input [4:0] floor, input integer delay\_time);

        begin

            InToggle[floor] = 1'b1;

            #delay\_time InToggle[floor] = 1'b0;

        end

    endtask

    task handle\_out\_toggle(input [4:0] floor, input integer delay\_time);

        begin

            OutToggle[floor] = 1'b1;

            #delay\_time OutToggle[floor] = 1'b0;

        end

    endtask

در یک بلاک initial نیز سناریوی مورد نظر به آسانسور داده شد که در ادامه روند آن توضیح داده خواهد شد.

    initial begin

        clk = 0;

        InToggle = 0;

        OutToggle = 0;

        handle\_out\_toggle(0, 20);

        handle\_in\_toggle(2, 20);

        #400

        handle\_out\_toggle(1, 20);

        handle\_in\_toggle(2, 20);

        handle\_out\_toggle(3, 20);

        handle\_in\_toggle(0, 20);

        #400

        handle\_out\_toggle(2, 20);

        handle\_out\_toggle(4, 20);

        #400

        handle\_in\_toggle(0, 20);

        handle\_in\_toggle(2, 20);

        #1000;

        $stop();

    end

در انتها نیز در یک بلاک Always، تغییرات حالات طبقات، حرکت و جهت آسانسور و فشرده شدن دکمه های آسانسور، نمایش داده شد.

    reg [1:0] TmpFloor [4:0];

    reg TmpProcess;

    reg TmpDir;

    reg [4:0] TmpButtons;

    integer i;

    always @(posedge clk) begin

        if (FloorInOut !== TmpFloor || Process !== TmpProcess || Dir !== TmpDir || PushedButtons !== TmpButtons) begin

            $display("Time: %3d | PushedButtons: %b | Process: %b | Dir: %b | Floors: %s | %s | %s | %s | %s",

                     $time, PushedButtons, Process, Dir,

                     FloorInOut[0] == 2'b00 ? "STOP" : (FloorInOut[0] == 2'b01 ? "IN" : (FloorInOut[0] == 2'b10 ? "OUT" : "-")),

                     FloorInOut[1] == 2'b00 ? "STOP" : (FloorInOut[1] == 2'b01 ? "IN" : (FloorInOut[1] == 2'b10 ? "OUT" : "-")),

                     FloorInOut[2] == 2'b00 ? "STOP" : (FloorInOut[2] == 2'b01 ? "IN" : (FloorInOut[2] == 2'b10 ? "OUT" : "-")),

                     FloorInOut[3] == 2'b00 ? "STOP" : (FloorInOut[3] == 2'b01 ? "IN" : (FloorInOut[3] == 2'b10 ? "OUT" : "-")),

                     FloorInOut[4] == 2'b00 ? "STOP" : (FloorInOut[4] == 2'b01 ? "IN" : (FloorInOut[4] == 2'b10 ? "OUT" : "-")));

        end

        TmpFloor <= FloorInOut;

        TmpProcess <= Process;

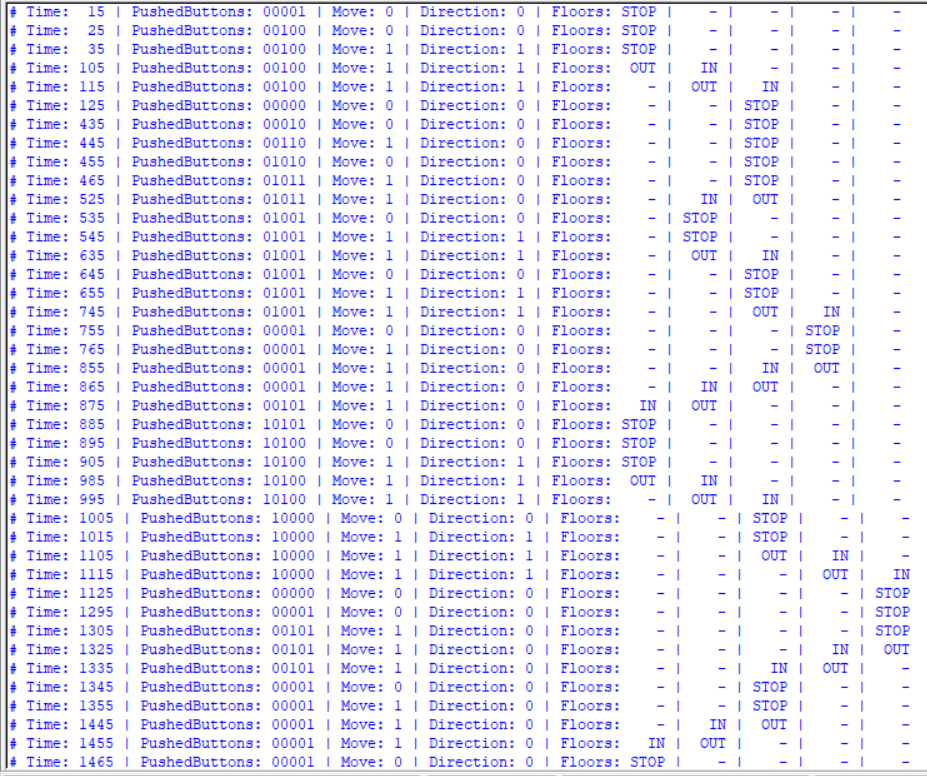
        TmpDir <= Dir;

        TmpButtons <= PushedButtons;

    end

endmodule

پس از simulate کردن تست طراحی شده خروجی زیر مشاهده شد



سناریوی تعریف شده برای آسانسور بدین صورت است که در ابتدا یک شخص دکمه طبقه همکف را میفشارد و پس از ورود به آسانسور، مقصد طبقه دوم را برای خود انتخاب میکند. پس از آن در لحظاتی بعد فردی در طبقه اول دکمه آسانسور را فشرده و مقصد طبقه دوم را برای خود انتخاب میکند و به طور همزمان فرد دیگری در طبقه سوم دکمه آسانسور را میفشارد که آسانسور ابتدا در طبقه اول نفر اول را سوار میکند . سپس او را در طبقه دوم پیاده کرده و پس از آن به طبقه سوم رفته و نفر بعدی را سوار کرده و به طبقه همکف میرود تا او نیز پیاده شود. پس از گذشت یک بازه زمانی، فرد دیگری در طبقه دوم دکمه آسانسور را فشرده و سوار میشود و نفر دیگری در طبقه چهارم نیز دکمه آسانسور را میفشارد که در ابتدا فرد در طبقه دوم و پس از آن فرد در طبقه چهارم سوار آسانسور میشوند. پس از گذشت یک بازه زمانی یکی از آنها ابتدا دکمه طبقه همکف را میفشارد و دیگری با تاخیر، دکمه طبقه دوم را میفشارد که آسانسور ابتدا در طبقه دوم توقف میکند و سپس به طبقه همکف میرود تا نفر دوم نیز پیاده شود.

در سناریوی طراحی شده سعی شد تا حرکات در جهات مختلف برای آسانسور در نظر گرفته شود تا تمامی موارد پوشش داده شوند. همچنین در آخرین بخش نیز مسئله بیان شده در صورت سوال که اگر در میانه راه آسانسور، طبقه دیگری فشرده شود، باید آنجا توقف کند نیز آورده شده است که در خروجی تست، صحت عملکرد آسانسور به ازای این ورودی مشهود است.