پیش گزارش آزمایش سوم

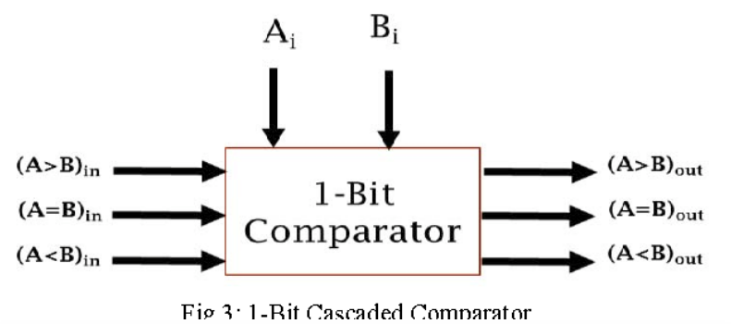
اعضای گروه:

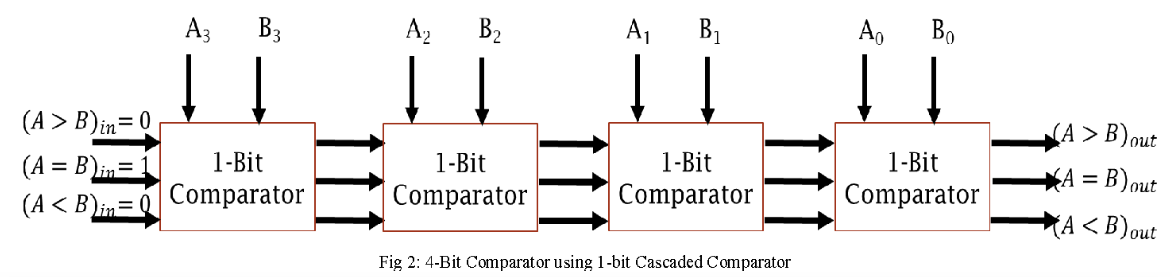
صادق محمدیان:۴۰۱۱۰۹۴۷۷

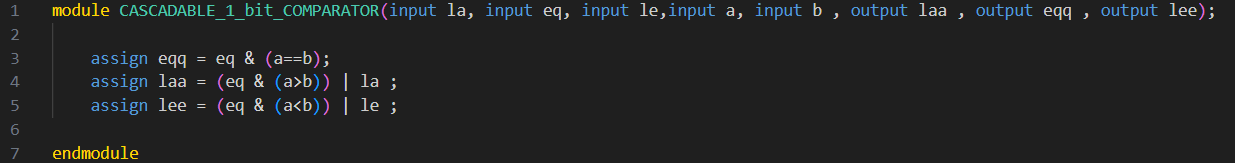
متین محمدی:401110329

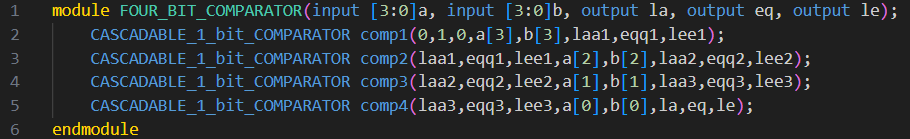
امیرحسین ملک محمدی:401106577

قسمت اول:

برای این قسمت ابتدا ماژول cascadable\_1\_bit\_comparator را طراحی می کنیم که شمای کلی آن به شکل زیر است:

سپس با کنار هم قرار دادن ۴ cascadable\_1\_bit\_comparator یک 4bit comparator می سازیم که شمای کلی آن به شکل زیر است:

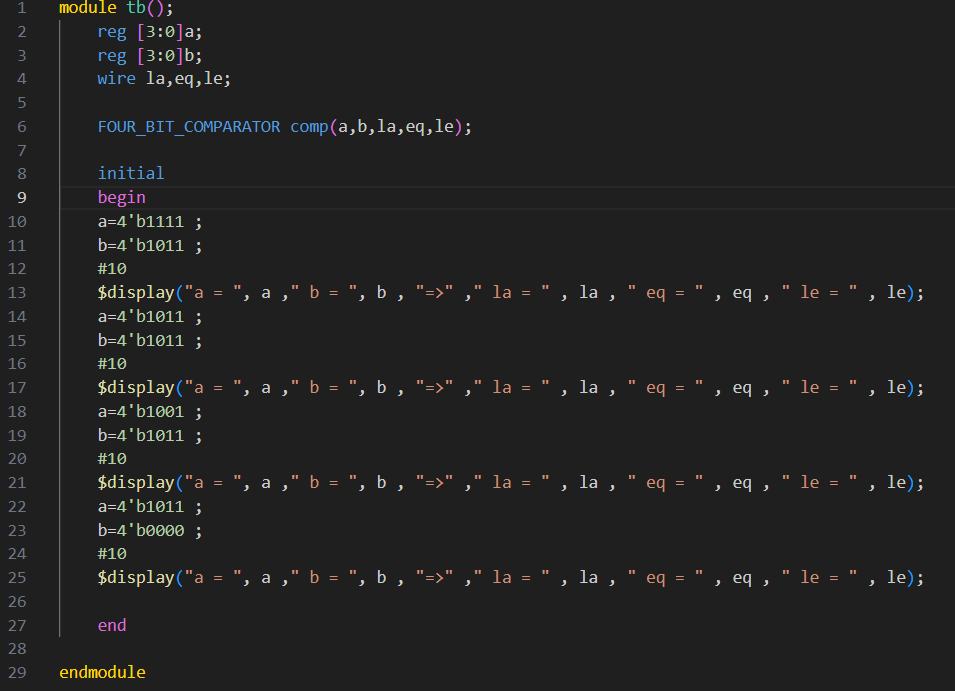
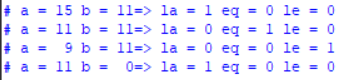
ماژول cascadable\_1\_bit\_comparator :

ماژول four\_bit\_comparator‌ :

روش کار به این صورت است که با مقایسه پر ارزش ترین بیت شروع می کنیم و بیت به بیت مقایسه می کنیم و و با استفاده از نتایج مقایسه بیت قبلی مقدار های la, eq, le را آپدیت می کنیم.باید توجه کنیم مقادیر ورودی

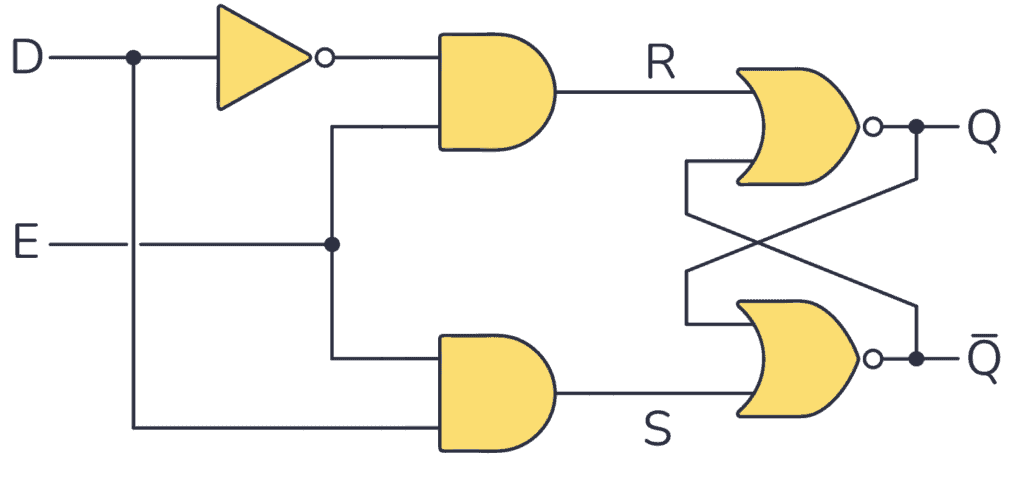
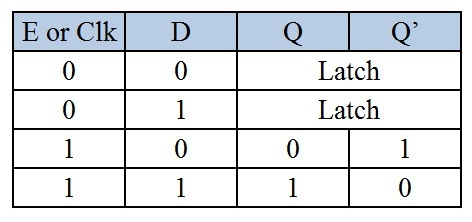
laa,eqq,lee ورودی به اولین cascadable\_1\_bit\_comparator باید به ترتیب 0,1,0 باشند.

ماژول تست آن به صورت زیر می باشد:

و نتیجه حاصل از شبیه سازی نیز به شکل زیر است:

قسمت دوم:

طراحی ما باید شامل یک ماژول باشد و ما باید نتیجه مقایسه تا بیت قبلی را نگه داریم بدین منظور می توانیم از

Latch استفاده کنیم.

با توجه به نتیجه مقایسه تا بیت قبلی و بیت های ورودی جدید ۳ حالت رخ می دهد:

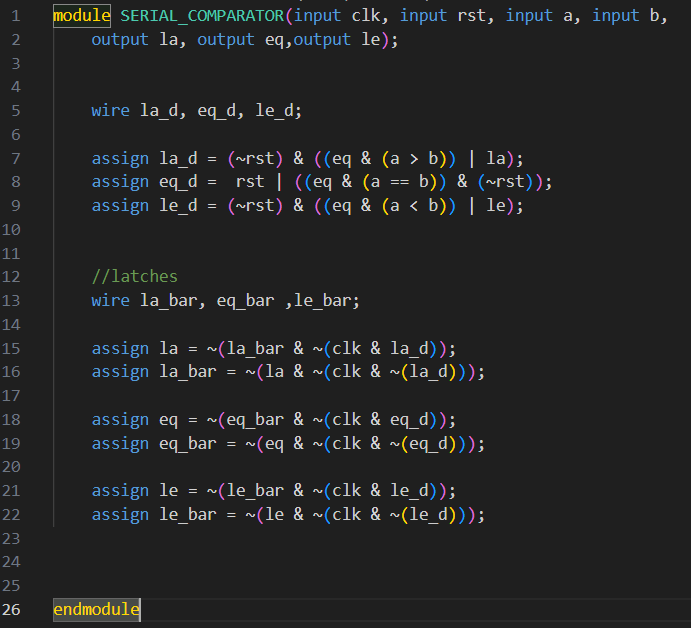
1. عدد اول بزرگتر از عدد دوم باشد: در اینصورت با ورود بیت های جدید همچنان عدد اول بزرگتر می ماند.
2. عدد اول کوچکتر از عدد دوم باشد: در اینصورت با ورود بیت های جدید همچنان عدد اول کوچکتر می ماند.
3. دو عدد مساوی باشند: در این صورت

اگر بیت های ورودی عدد اول بزرگتر باشد عدد اول بزرگتر خواهد بود

اگر دو بیت برابر باشند عدد نهایی بزرگتر خواهد بود

و در صورتی که بیت عدد دوم بزرگتر باشد عدد اول کوچکتر خواهد بود.

طراحی ما برای این ماژول به صورت زیر خواهد بود:



ماژول آزمون ما نیز به صورت زیر خواهد بود:

module SERIAL\_COMPARATOR\_TB();

    reg rst, clk, a, b;

    wire la, eq, le;

    SERIAL\_COMPARATOR cfomparator(clk, rst, a, b, la, eq, le);

    always  begin

        #5 clk = ~clk;

    end

    initial begin

        clk = 0;

        rst = 1;

        #10

        $display("time: %d, reset: %b a\_in: %b, b\_in: %b,  la: %b, eq: %b, le: %b", $time, rst, a, b, la, eq, le);

        rst = 0;

        a = 1;

        b = 0;

        #10

        $display("time: %d, reset: %b a\_in: %b, b\_in: %b,  la: %b, eq: %b, le: %b", $time, rst, a, b, la, eq, le);

        a = 0;

        b = 1;

        #10

        $display("time: %d, reset: %b a\_in: %b, b\_in: %b,  la: %b, eq: %b, le: %b", $time, rst, a, b, la, eq, le);

        a = 0;

        b = 1;

        #10

        $display("time: %d, reset: %b a\_in: %b, b\_in: %b,  la: %b, eq: %b, le: %b", $time, rst, a, b, la, eq, le);

        rst = 1;

        #10

        $display("time: %d, reset: %b a\_in: %b, b\_in: %b,  la: %b, eq: %b, le: %b", $time, rst, a, b, la, eq, le);

        rst = 0;

        a = 1;

        b = 1;

        #10

        $display("time: %d, reset: %b a\_in: %b, b\_in: %b,  la: %b, eq: %b, le: %b", $time, rst, a, b, la, eq, le);

        a = 0;

        b = 1;

        #10

        $display("time: %d, reset: %b a\_in: %b, b\_in: %b,  la: %b, eq: %b, le: %b", $time, rst, a, b, la, eq, le);

        a = 1;

        b = 0;

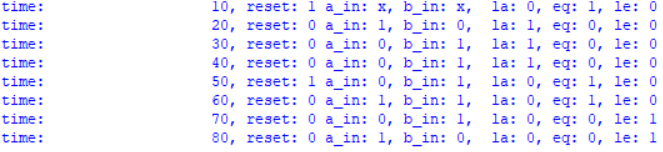
        #10

        $display("time: %d, reset: %b a\_in: %b, b\_in: %b,  la: %b, eq: %b, le: %b", $time, rst, a, b, la, eq, le);

    end

endmodule

و نتایج آن به صورت زیر است که نشان می دهد کد ما به درستی کار می کند:

ابتدا دو سریال 100 و 011 را مقایسه می کنیم و سپس بعد از ریست کردن مدار دوسریال 101 و 110را تست می کنیم.

