آزمایشگاه طراحی سیستمهای دیجیتال
پیش‌گزارش آزمایش ۴

۲۳ تیر ۱۴۰۳

امیرحسین ملک محمدی ۴۰۱۱۰۶۵۷۷
متین محمدی ۴۰۱۱۱۰۳۲۹
صادق محمدیان ۴۰۱۱۰۸۴۷۷

۱ شرح آزمایش

در این آزمایش می‌خواهیم یک پشته را به صورت توصیف رفتاری طراحی کنیم. این پشته می‌تواند ۸ کلمه ۴ بیتی را در خود نگه دارد.

۱.۱ ورودی‌ها

ورودی‌های این پشته سیگنال‌های کلاک، ریست، داده ورودی، پوش، و پاپ اند.

ورودی ریست یک سیگنال ناهمگام(asynchronous) است و وقتی که فعال شود پشته خالی می‌شود.

زمانی که سیگنال پوش فعال شود و پشته پر نباشد باید داده ورودی را در پشته قرار دهیم و زمانی که سیگنال پاپ فعال شود و پشته خالی نباشد، داده آخر را خروجی می‌دهیم و آن را از پشته پاک می‌کنیم.

۲.۱ خروجی‌ها

خروجی‌های این پشته سیگنال‌های داده خروجی، پر بودن، و خالی‌بودن اند.

زمانی که هیچ داده‌ای در پشته نباشد(یعنی نشانگر به خانه صفر اشاره کند) خروجی خالی‌بودن فعال می‌شود. زمانی که پشته پر باشد(یعنی نشانگر به خانه هشتم اشاره کند) خروجی پربودن فعال می‌شود.

اگر ورودی پاپ فعال باشد و پشته خالی نباشد آخرین داده را بر داده خروجی قرار می‌دهیم.

۲ کد وریلاگ

در شکل .Fig ۱ کد وریلاگ پشته آمده.

در خط اول ورودی‌ها و خروجی‌های را مشخص کردیم. سپس حافظه اصلی را تعریف کردیم که یک آرایه ۸ تایی از کلمات ۴ بیتی است. در خط بعد یک متغییر index را معرفی می‌کنیم که به آخرین خانه خالی پشته اشاره می‌کند. این متغییر باید از ۰ تا ۸ تغییر کند پس لازم است ۴ بیتی باشد.

خروجی‌های خالی یا پر بودن با بررسی index بدست می‌آیند؛ اگر index برابر ۰ باشد یعنی پشته خالی است و اگر index برابر ۸ باشد یعنی پشته پر است.

در خط بعد داده خروجی را به رجیستر data متصل می‌کنیم. این خروجی در صورتی که پاپ فعال باشد، پشته خالی نباشد، و ریست غیر فعال باشد به آخرین خانه پشته متصل است. و در غیر این صورت z است.

در خط بعد یک بلاک always داریم که با لبه مثبت کلاک یا ریست فعال می‌شود.

داخل بلاک اگر ریست فعال شده بود index را برابر ۰ می‌کنیم و پشته ریست می‌شود.

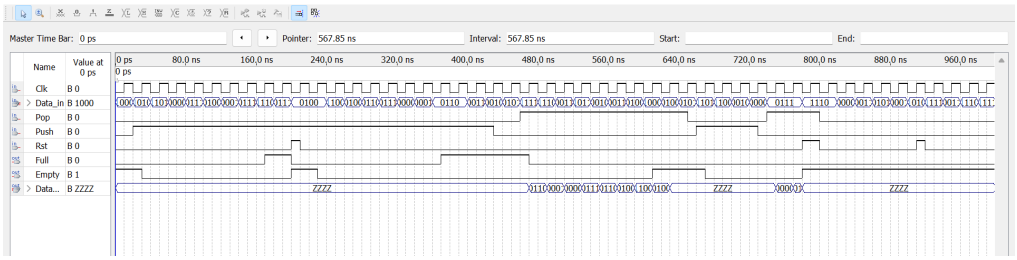
در غیر این صورت اگر پاپ فعال شده بود و پشته خالی نبود اول index را یکی کم می‌کنیم سپس داده آخر را خروجی می‌دهیم.

اگر پوش فعال شده بود و پشته پر نبود داده را می‌نویسیم و index را یکی زیاد می‌کنیم.

```
module STACK_Q (input Clk, input Rst, input Push, input Pop, input [3:0] Data_in,
                output Full, output Empty, output [3:0] Data_out);
    reg [3:0] memory [7:0];
    reg [3:0] index = 3'b0;
    assign Empty = (index == 0);
    assign Full = (index == 8);
    reg [3:0] data;
    assign Data_out = data;

    always @( posedge Clk, posedge Rst) begin
        if (Rst) begin
            index = 3'b0;
            data = 4'bz;
        end
        else begin
            data = 4'bz;
            if (Pop && ~Empty ) begin
                index = index - 3'b1;
                data = memory[index];
            end
            else if (Push && ~Full ) begin
                memory[index] = Data_in;
                index = index + 3'b1;
            end
        end
    end
endmodule
```

شکل ۱: کد وریلاگ پشته



شکل ۲: خروجی ویوفرم

۳ تست ماژول

به دو صورت مدار را تست خواهیم کرد. ابتدا از ویوفرم استفاده می‌کنیم و سپس با یک تست‌بنچ عملکرد ماژول را تست می‌کنیم.

۱.۳ ویوفرم

اول یک پروژه در برنامه کوارتوس باز می‌کنیم و کد وریلاگ را کامپایل می‌کنیم. سپس با ویوفرم خروجی می‌گیریم.

در شکل .Fig ۲ نتیجه ویوفرم آمده است. در این ویوفرم تمام قابلیت‌ها و خروجی‌های پشته را بررسی کردیم و درست عمل می‌کردند.

۲.۳ تست‌بنچ

در شکل .Fig ۳ کد تست‌بنچ آمده است. همچنین اصل کد در فایل‌ها به اسم tb.v ذخیره شده و در دسترس است.

در این تست‌بنچ اول پشته را ریست می‌کنیم، در کلاک بعد یک عدد در آن پوش می‌کنیم و در کلاک بعد همان را پاپ می‌کنیم. سپس دوباره پشته را ریست می‌کنیم و ۸ عدد را وارد پشته می‌کنیم. باید بعد از آن خروجی full پشته فعال شود. بعد از آن همین ۸ عدد را پاپ می‌کنیم.

خروجی این تست بنچ در شکل .Fig ۴ آمده.

خروجی مطابق انتظار است. اول خروجی empty فعال است، سپس با پوش کردن داده‌ای در پشته آن خروجی خاموش می‌شود. کلاک بعد از پشته پاپ می‌کنیم و همان عدد را می‌گیریم. سپس ۸ عدد را پوش می‌کنیم و خروجی full فعال می‌شود. در مرحله بعد که پاپ می‌کنیم پشته همین اعداد را به ترتیب برعکس خروجی می‌دهد.

۴ نتیجه‌گیری

در این آزمایش ابتدا با زبان وریلاگ و به صورت توصیف رفتاری یک پشته طراحی کردیم. سپس با یک تست‌بنچ و یک خروجی ویوفرم از درستی عملکرد آن اصمینان حاصل کردیم.

```
module tb();
    initial
    begin
        clk=0;
        rst = 1;
        #10
        $display("time: %d, reset: %b push: %b, pop: %b, Data_in: %b, full: %b, empty: %b, Data_out: %b" , $time, rst, push, pop, Data_in, full, empty , Data_out);
        rst = 0;
        Data_in = 3;
        push = 1; pop = 0;
        #10
        $display("time: %d, reset: %b push: %b, pop: %b, Data_in: %b, full: %b, empty: %b, Data_out: %b" , $time, rst, push, pop, Data_in, full, empty , Data_out);
        push = 0;
        Data_in = 1;
        push = 1; pop = 0;
        #10
        $display("time: %d, reset: %b push: %b, pop: %b, Data_in: %b, full: %b, empty: %b, Data_out: %b" , $time, rst, push, pop, Data_in, full, empty , Data_out);
        push = 0; pop = 1;
        #10
        $display("time: %d, reset: %b push: %b, pop: %b, Data_in: %b, full: %b, empty: %b, Data_out: %b" , $time, rst, push, pop, Data_in, full, empty , Data_out);
        pop = 0; rst = 1;
        #10
        $display("time: %d, reset: %b push: %b, pop: %b, Data_in: %b, full: %b, empty: %b, Data_out: %b" , $time, rst, push, pop, Data_in, full, empty , Data_out);
        rst = 0;
        for(i = 1 ; i <= 8 ; i = i + 1)
        begin
            push = 0;
            Data_in = i;
            push = 1; pop = 0;
            #10;
        end
        $display("time: %d, reset: %b push: %b, pop: %b, Data_in: %b, full: %b, empty: %b, Data_out: %b" , $time, rst, push, pop, Data_in, full, empty , Data_out);
        #5 Data_in = 4'bx;
        for(i = 1 ; i <= 8 ; i = i + 1)
        begin
            push = 0;
            push = 0; pop = 1;
            #10;
            $display("time: %d, reset: %b push: %b, pop: %b, Data_in: %b, full: %b, empty: %b, Data_out: %b" , $time, rst, push, pop, Data_in, full, empty , Data_out);
        end
    end
```

شکل ۳: کد تست‌بنچ

```
VSIM 8> run
# time:                10, reset: 1 push: x, pop: x, Data_in: xxxx, full: 0, empty: 1, Data_out: zzzz
# time:                20, reset: 0 push: 1, pop: 0, Data_in: 0011, full: 0, empty: 0, Data_out: zzzz
# time:                30, reset: 0 push: 1, pop: 0, Data_in: 0001, full: 0, empty: 0, Data_out: zzzz
# time:                40, reset: 0 push: 0, pop: 1, Data_in: 0001, full: 0, empty: 0, Data_out: 0001
# time:                50, reset: 1 push: 0, pop: 0, Data_in: 0001, full: 0, empty: 1, Data_out: zzzz
run
# time:               130, reset: 0 push: 1, pop: 0, Data_in: 1000, full: 1, empty: 0, Data_out: zzzz
# time:               145, reset: 0 push: 0, pop: 1, Data_in: xxxx, full: 0, empty: 0, Data_out: 1000
# time:               155, reset: 0 push: 0, pop: 1, Data_in: xxxx, full: 0, empty: 0, Data_out: 0111
# time:               165, reset: 0 push: 0, pop: 1, Data_in: xxxx, full: 0, empty: 0, Data_out: 0110
# time:               175, reset: 0 push: 0, pop: 1, Data_in: xxxx, full: 0, empty: 0, Data_out: 0101
# time:               185, reset: 0 push: 0, pop: 1, Data_in: xxxx, full: 0, empty: 0, Data_out: 0100
# time:               195, reset: 0 push: 0, pop: 1, Data_in: xxxx, full: 0, empty: 0, Data_out: 0011
run
# time:               205, reset: 0 push: 0, pop: 1, Data_in: xxxx, full: 0, empty: 0, Data_out: 0010
# time:               215, reset: 0 push: 0, pop: 1, Data_in: xxxx, full: 0, empty: 1, Data_out: 0001
VSIM 9> run
```

شکل ۴: خروجی تست‌بنچ