گزارش کار آزمایشگاه DSD

آزمایش شماره 4

18 فروردین 1400

عرشیا اخوان
محمدحسین عبدی
علیرضا ایلامی

شماره آزمایش: **4**	موضوع: توصیف رفتاری	تاریخ آزمایش: 10 فروردین 1400
عرشیا اخوان **97110422**	محمدحسین عبدی **97110285**	علیرضا ایلامی **97101286**

هدف: طراحی یک پشته با عمق 8 و پهنای 4 بیت

مقدمه: در این آزمایش میخواهیم یک پشته (Stack) طراحی کنیم که اعداد را یکی یکی در آن وارد کنیم. و هرگاه ظرفیت پر شد، دیگر اجازه اضافه کردن ورودی به پشته را ندهد. سپس، یکی یکی اعداد را به ترتیب FILO از پشته بیرون بیاوریم تا زمانی که به کف پشته برسیم و دیگر چیزی برای بیرون آوردن وجود نداشته باشد.

توضیح کد:

یک ماژول به نام Stack طراحی کرده ایم که تعدادی سیگنال برای push, pop و کلاک و ریست و غیره ورودی میگیرد. دو متغیر برای عمق و پهنای پشته نیز داریم که آنها را نیز ست میکنیم. این ماژول سه سیگنال خروجی دارد: یکی سیگنال data_out (به پهنای BANDWIDTH) که عدد خروجی داده شده را در هنگام استفاده از سیگنال pop به ما میدهد، یک بیت برای نشان دادن full بودن پشته و یکی هم برای نشان دادن empty بودن پشته.

سیگنال های full , empty را به این صورت طراحی میکنیم که هرگاه نشانگر پشته یا همان stack pointer به ابتدا یا انتهای پشته اشاره میکرد، یکی از این دو بیت فعال و برابر با 1 شوند.

در ادامه، ابتدا استک را ریست میکنیم. سپس در یک always block، متناسب با سیگنالهای ورودی که کدامشان فعال باشند، اعداد را در پشته درج یا از پشته خارج میکنیم.

```
module Stack (rstn,
              data_in,
              push,
              pop,
              clk,
              data_out,
              full,
              empty);

    parameter DEPTH     = 8;
    parameter BANDWIDTH = 4;

    input wire rstn, push, pop, clk;
    input wire[BANDWIDTH-1:0] data_in;
    output full, empty;
    output reg [BANDWIDTH-1:0] data_out;

    reg[$clog2(DEPTH):0] stack_ptr;
    reg[BANDWIDTH-1:0] memory [0:DEPTH-1];

    assign empty = (stack_ptr == 0) ? 1'b1 : 1'b0;
    assign full  = (stack_ptr == DEPTH) ? 1'b1 : 1'b0;

    // stack reset
    task reset_memory;
        integer i;
        begin
            for (i = 0; i < DEPTH;i++) begin
                memory[i] = {BANDWIDTH{1'b0}};
            end
```

```
    task reset_memory;
        integer i;
        begin
            for (i = 0; i < DEPTH;i++) begin
                memory[i] = {BANDWIDTH{1'b0}};
            end
            stack_ptr = 0;
        end
    endtask

    always @(posedge clk or negedge rstn) begin
        if (rstn) begin
            reset_memory;
        end
        else begin
            // pushing into stack
            if (push && !pop && !full) begin
                memory[stack_ptr] = data_in;
                stack_ptr         = stack_ptr + 1;
            end
            // pop from stack
            if (pop && !push && !empty) begin
                data_out  = memory[stack_ptr - 1];
                stack_ptr = stack_ptr - 1;
            end
        end
    end
endmodule
```

نتایج تست بنچ در پوشه report و فایل result.txt آورده شده اند.

در زیر تصویر آنها به پیوست گزارش آمده است:

توضیحات نتایج تست بنچ:

ابتدا استک ریست میشود. در ابتدا سیگنال empty را میبینیم که فعال است. سپس یکی یکی اعداد را در پشته وارد میکنیم. تا جایی که به اندازه عمق استک (در اینجا 8) عدد وارد شده باشد و استک پر شده باشد. در اینجا اگر عدد 8 را وارد کنیم، میبینیم که چون سیگنال full = 1 است، این عدد وارد نمیشود و stack pointer تغییری نمیکند.

```
                0    push=x, pop=x, data_in= x, stack_ptr= x, empty=x, full=x, data_out= x
                5    push=x, pop=x, data_in= x, stack_ptr= 0, empty=1, full=0, data_out= x
fill the stack
               20    push=0, pop=0, data_in= 0, stack_ptr= 0, empty=1, full=0, data_out= x
               30    push=1, pop=0, data_in= 0, stack_ptr= 0, empty=1, full=0, data_out= x
               35    push=1, pop=0, data_in= 0, stack_ptr= 1, empty=0, full=0, data_out= x
               40    push=0, pop=0, data_in= 1, stack_ptr= 1, empty=0, full=0, data_out= x
               50    push=1, pop=0, data_in= 1, stack_ptr= 1, empty=0, full=0, data_out= x
               55    push=1, pop=0, data_in= 1, stack_ptr= 2, empty=0, full=0, data_out= x
               60    push=0, pop=0, data_in= 2, stack_ptr= 2, empty=0, full=0, data_out= x
               70    push=1, pop=0, data_in= 2, stack_ptr= 2, empty=0, full=0, data_out= x
               75    push=1, pop=0, data_in= 2, stack_ptr= 3, empty=0, full=0, data_out= x
               80    push=0, pop=0, data_in= 3, stack_ptr= 3, empty=0, full=0, data_out= x
               90    push=1, pop=0, data_in= 3, stack_ptr= 3, empty=0, full=0, data_out= x
               95    push=1, pop=0, data_in= 3, stack_ptr= 4, empty=0, full=0, data_out= x
              100    push=0, pop=0, data_in= 4, stack_ptr= 4, empty=0, full=0, data_out= x
              110    push=1, pop=0, data_in= 4, stack_ptr= 4, empty=0, full=0, data_out= x
              115    push=1, pop=0, data_in= 4, stack_ptr= 5, empty=0, full=0, data_out= x
              120    push=0, pop=0, data_in= 5, stack_ptr= 5, empty=0, full=0, data_out= x
              130    push=1, pop=0, data_in= 5, stack_ptr= 5, empty=0, full=0, data_out= x
              135    push=1, pop=0, data_in= 5, stack_ptr= 6, empty=0, full=0, data_out= x
              140    push=0, pop=0, data_in= 6, stack_ptr= 6, empty=0, full=0, data_out= x
              150    push=1, pop=0, data_in= 6, stack_ptr= 6, empty=0, full=0, data_out= x
              155    push=1, pop=0, data_in= 6, stack_ptr= 7, empty=0, full=0, data_out= x
              160    push=0, pop=0, data_in= 7, stack_ptr= 7, empty=0, full=0, data_out= x
try to push again: stack is full
              170    push=1, pop=0, data_in= 7, stack_ptr= 7, empty=0, full=0, data_out= x
              175    push=1, pop=0, data_in= 7, stack_ptr= 8, empty=0, full=1, data_out= x
              180    push=0, pop=0, data_in= 9, stack_ptr= 8, empty=0, full=1, data_out= x
```

در این بخش میخواهیم استک را خالی کنیم. سیگنال pop = 1 میکنیم و سیگنال های push = 0 , empty = 0 به ما این اجازه را میدهد که از پشته عدد بیرون بیاوریم.

همانطور که واضح است. ابتدا آخرین عدد (یعنی 7) بیرون می آید.

در ادامه یکی یکی بقیه خانه های استک را خالی میکنیم تا زمانی که به انتها برسیم.

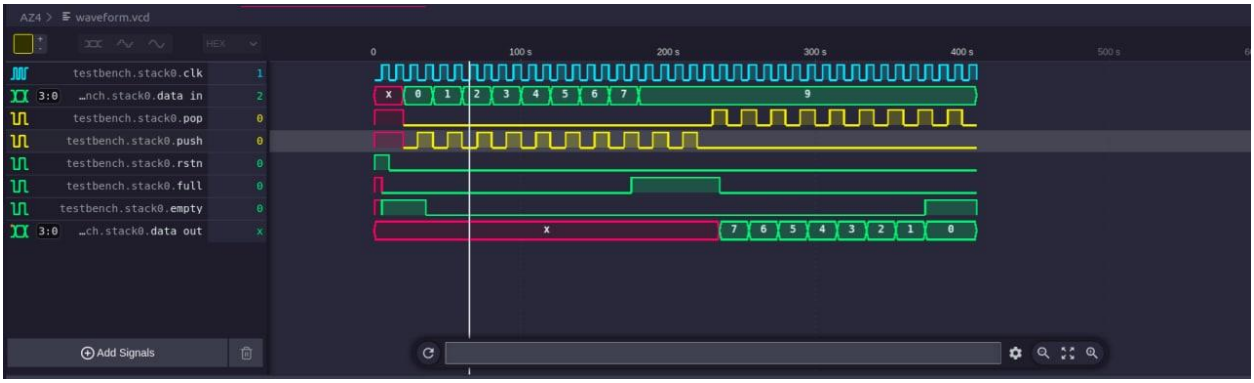
هرگاه stack pointer = 0 شود به این معناست که به کف استک رسیده ایم و سیگنال empty = 1 میگردد. بدین ترتیب دیگر اجازه pop کردن نداریم و کار تمام میشود.

```
try to push again: stack is full
                190    push=1, pop=0, data_in= 9, stack_ptr= 8, empty=0, full=1, data_out= x
                200    push=0, pop=0, data_in= 9, stack_ptr= 8, empty=0, full=1, data_out= x
empty stack
                210    push=1, pop=0, data_in= 9, stack_ptr= 8, empty=0, full=1, data_out= x
                220    push=0, pop=0, data_in= 9, stack_ptr= 8, empty=0, full=1, data_out= x
                230    push=0, pop=1, data_in= 9, stack_ptr= 8, empty=0, full=1, data_out= x
                235    push=0, pop=1, data_in= 9, stack_ptr= 7, empty=0, full=0, data_out= 7
                240    push=0, pop=0, data_in= 9, stack_ptr= 7, empty=0, full=0, data_out= 7
                250    push=0, pop=1, data_in= 9, stack_ptr= 7, empty=0, full=0, data_out= 7
                255    push=0, pop=1, data_in= 9, stack_ptr= 6, empty=0, full=0, data_out= 6
                260    push=0, pop=0, data_in= 9, stack_ptr= 6, empty=0, full=0, data_out= 6
                270    push=0, pop=1, data_in= 9, stack_ptr= 6, empty=0, full=0, data_out= 6
                275    push=0, pop=1, data_in= 9, stack_ptr= 5, empty=0, full=0, data_out= 5
                280    push=0, pop=0, data_in= 9, stack_ptr= 5, empty=0, full=0, data_out= 5
                290    push=0, pop=1, data_in= 9, stack_ptr= 5, empty=0, full=0, data_out= 5
                295    push=0, pop=1, data_in= 9, stack_ptr= 4, empty=0, full=0, data_out= 4
                300    push=0, pop=0, data_in= 9, stack_ptr= 4, empty=0, full=0, data_out= 4
                310    push=0, pop=1, data_in= 9, stack_ptr= 4, empty=0, full=0, data_out= 4
                315    push=0, pop=1, data_in= 9, stack_ptr= 3, empty=0, full=0, data_out= 3
                320    push=0, pop=0, data_in= 9, stack_ptr= 3, empty=0, full=0, data_out= 3
                330    push=0, pop=1, data_in= 9, stack_ptr= 3, empty=0, full=0, data_out= 3
                335    push=0, pop=1, data_in= 9, stack_ptr= 2, empty=0, full=0, data_out= 2
                340    push=0, pop=0, data_in= 9, stack_ptr= 2, empty=0, full=0, data_out= 2
                350    push=0, pop=1, data_in= 9, stack_ptr= 2, empty=0, full=0, data_out= 2
                355    push=0, pop=1, data_in= 9, stack_ptr= 1, empty=0, full=0, data_out= 1
                360    push=0, pop=0, data_in= 9, stack_ptr= 1, empty=0, full=0, data_out= 1
try to pop again: stack is empty
                370    push=0, pop=1, data_in= 9, stack_ptr= 1, empty=0, full=0, data_out= 1
```


```
                360    push=0, pop=0, data_in= 9, stack_ptr= 1, empty=0, full=0, data_out= 1
try to pop again: stack is empty
                370    push=0, pop=1, data_in= 9, stack_ptr= 1, empty=0, full=0, data_out= 1
```


```
                375    push=0, pop=1, data_in= 9, stack_ptr= 0, empty=1, full=0, data_out= 0
                380    push=0, pop=0, data_in= 9, stack_ptr= 0, empty=1, full=0, data_out= 0
try to pop again: stack is empty
                390    push=0, pop=1, data_in= 9, stack_ptr= 0, empty=1, full=0, data_out= 0
                400    push=0, pop=0, data_in= 9, stack_ptr= 0, empty=1, full=0, data_out= 0
                410    push=0, pop=1, data_in= 9, stack_ptr= 0, empty=1, full=0, data_out= 0
```

موج های خروجی به شرح زیر می باشند:



کد تست بنچ نیز در زیر آورده شده است:

```
module testbench;

        parameter clk_c = 10;
        parameter BANDWIDTH = 4;
        parameter DEPTH = 8;
        reg rstn, s_push, s_pop, clk;
        reg[BANDWIDTH-1:0] data_in;
        wire full, empty;
        wire [BANDWIDTH-1:0] data_out;
        integer i;

        Stack #(.DEPTH(DEPTH),.BANDWIDTH(BANDWIDTH)) stack0 (
         .rstn(rstn),
         .data_in(data_in),
         .push(s_push),
         .pop(s_pop),
         .clk(clk),
         .data_out(data_out),
         .full(full),
         .empty(empty));

    initial begin
        $dumpfile("waveform.vcd");
        $dumpvars(0,stack0);
    end


    initial begin
        clk          = 0;
        forever clk = #(clk_c/2) ~clk;
    end

    initial begin
        rstn = 1;
        #clk_c    rstn = 0;

        // fill the stack
        $display("fill the stack");
        for (i=0; i<DEPTH;i=i+1) begin
            #clk_c push(i);
        end

        // try to push again: stack is full
        $display("try to push again: stack is full");
        #clk_c push(4'b1001);

        // try to push again: stack is full
        $display("try to push again: stack is full");
        #clk_c push(4'b1001);

        // empty stack
        $display("empty stack");
        for (i=0; i<DEPTH;i=i+1) begin
            #clk_c pop;
        end

        // try to pop again: stack is empty
        $display("try to pop again: stack is empty");
        #clk_c pop;

        // try to pop again: stack is empty
        $display("try to pop again: stack is empty");
        #clk_c pop;

        $finish;
```

```
        // try to pop again: stack is empty
        $display("try to pop again: stack is empty");
        #clk_c pop;

        $finish;

    end

    initial
        $monitor($time,"\tpush=%b, pop=%b, data_in=%d, stack_ptr=%d, empty=%b, full=%b, data_out=%d", s_push, s_pop, data_in, stack0.stack_ptr, empty, full, data_out);



    task push;
        input [BANDWIDTH-1:0]data;
        begin
            s_push  = 0;
            s_pop   = 0;
            rstn    = 0;
            data_in = data;
            #clk_c
            s_push  = 1;
        end
    endtask

    task pop;
        begin
            s_push  = 0;
            s_pop   = 0;
            rstn    = 0;
            #clk_c
            s_pop   = 1;
        end
    endtask

endmodule //testbench
```

پایان