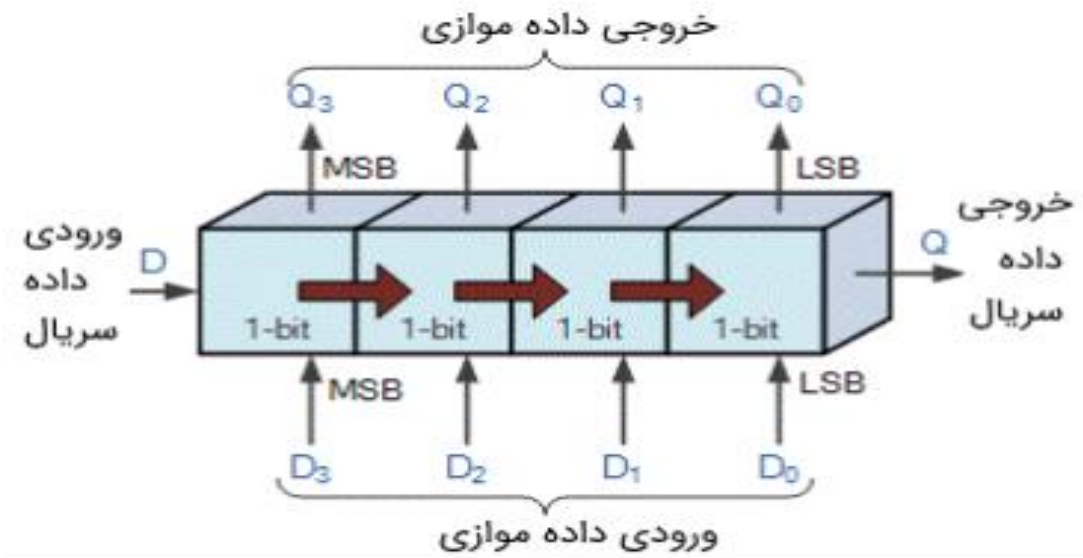
آزمایشگاه سیستم های دیجیتال2

آزمایش 4

شیفت رجیستر

شیفت رجیستر (Shift Register)

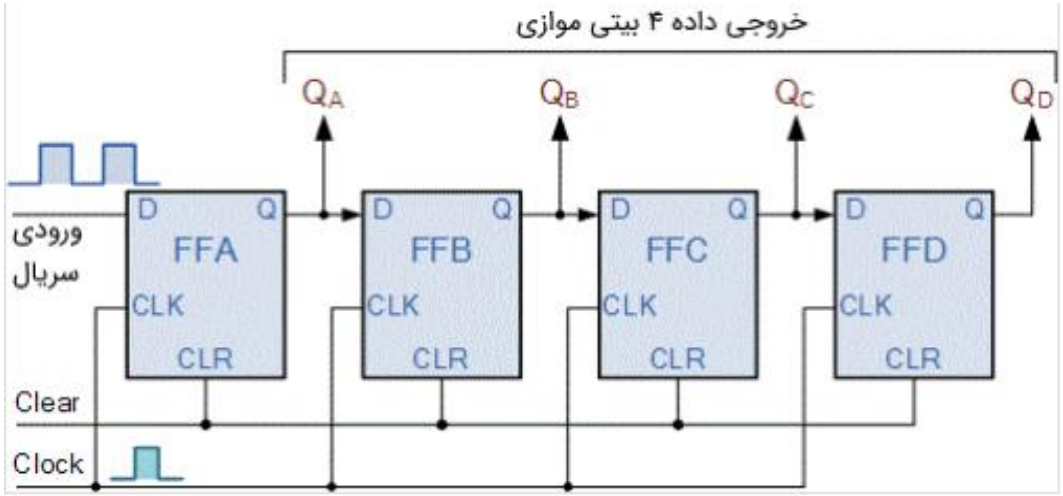


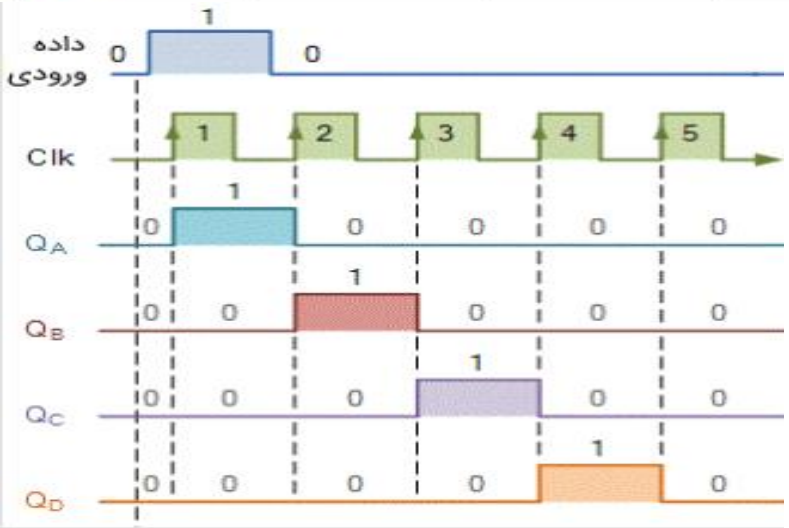
ادامه

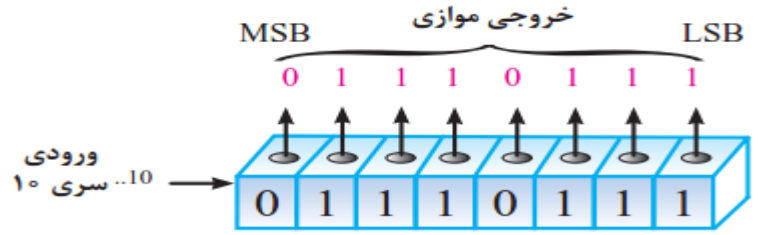
مدهای کاری شیفت رجیسترها :

- **ورودی سریال-خروجی موازی (Serial-in to Parallel-out) یا SIPO**
- **ورودی سریال-خروجی سریال (Serial-in to Serial-out) یا SISO**
- **ورودی موازی-خروجی سریال (Parallel-in to Serial-out) یا PISO**
- **ورودی موازی-خروجی موازی (Parallel-in to Parallel-out) یا PIPO**

SIPO Shift Registers

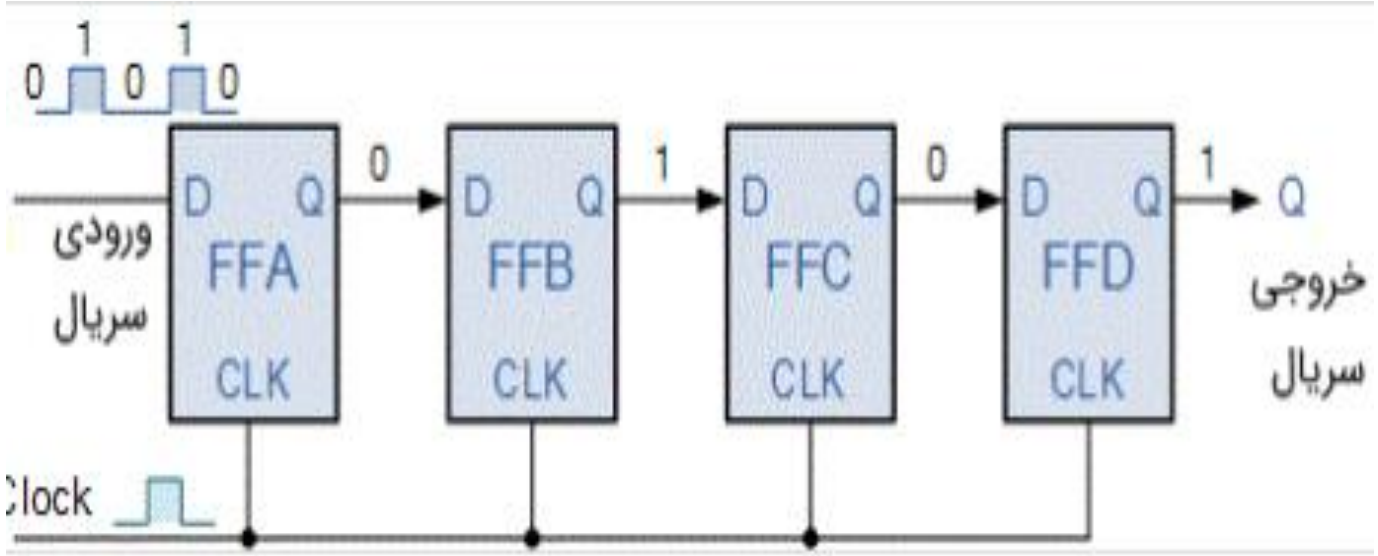




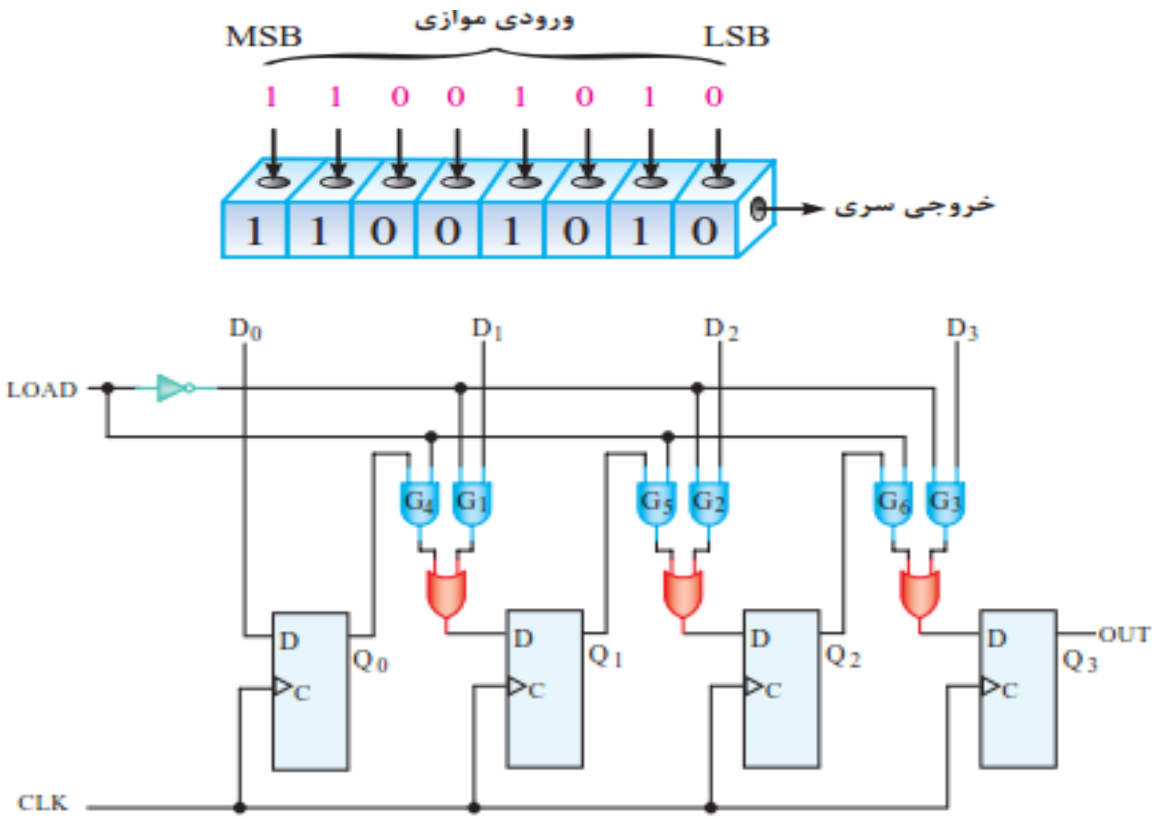


شماره پالس ساعت	Q_A	Q_B	Q_C	Q_D
۰	۰	۰	۰	۰
۱	۱	۰	۰	۰
۲	۰	۱	۰	۰
۳	۰	۰	۱	۰
۴	۰	۰	۰	۱
۵	۰	۰	۰	۰

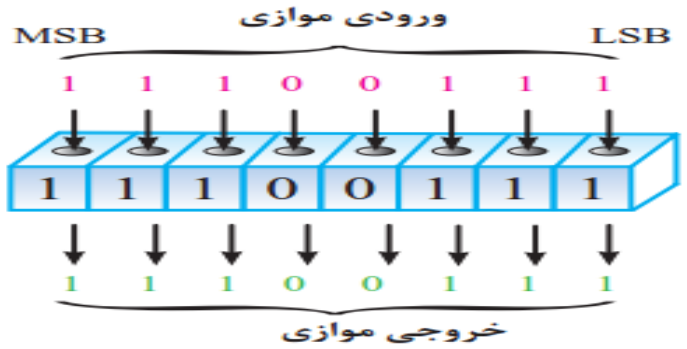
SISO Shift Registers

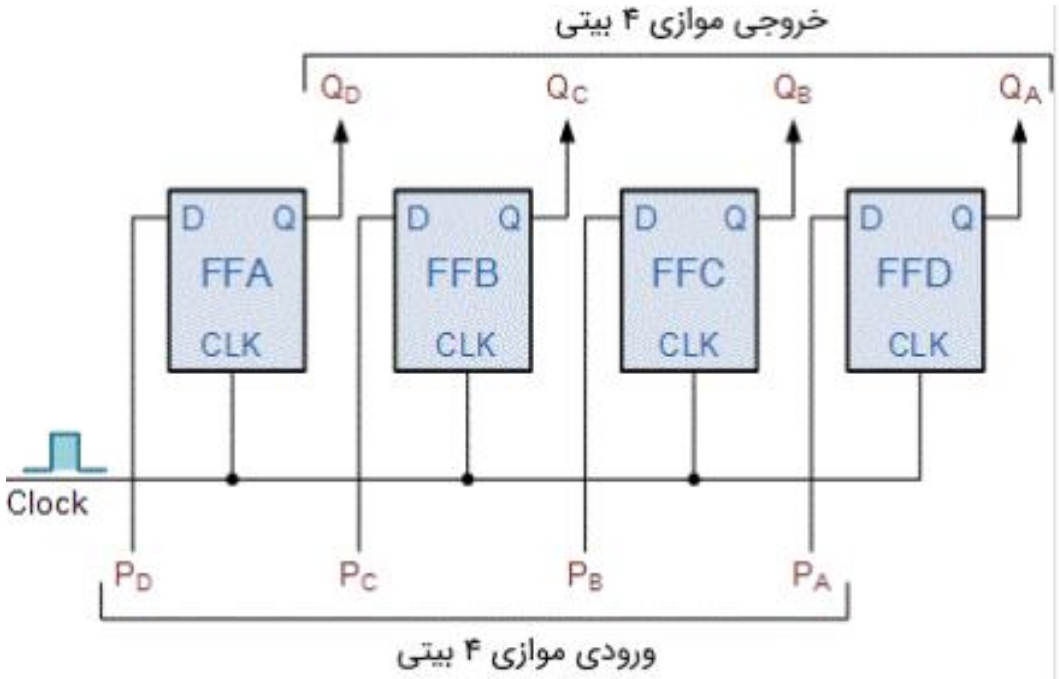


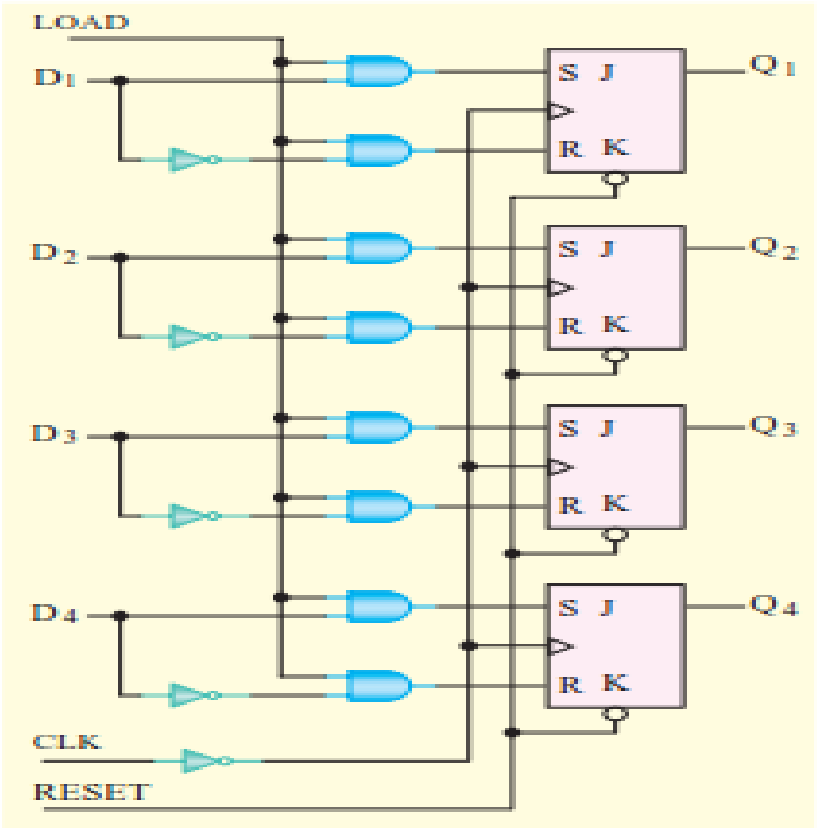
PISO Shift Registers



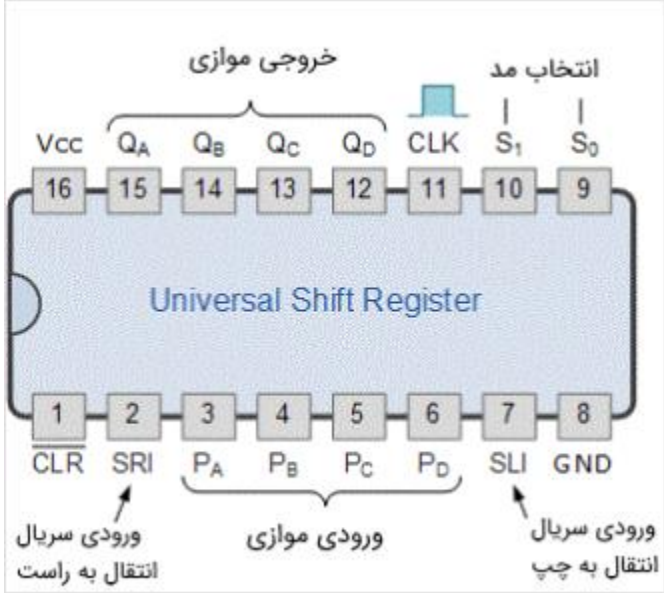
PIPO Shift Registers







شیفت رجیستر عمومی (Universal)



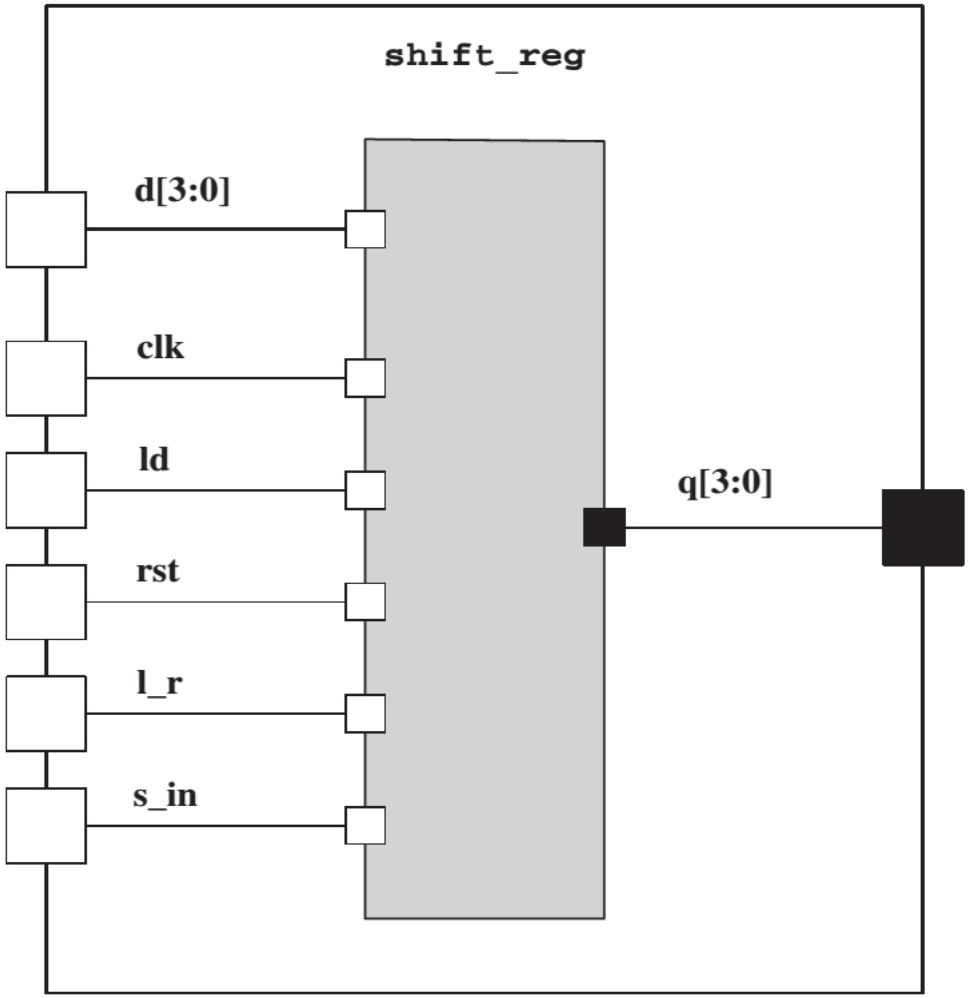
74LS194

نوع رجیستر	S_2	S_1
نگه‌دار	۰	۰
شیفت به راست	۱	۰
شیفت به چپ	۰	۱
مد موازی	۱	۱

Sequential Circuit Description

Functional Registers

A Basic Shift Register



Sequential Circuit Description

Functional Registers

A Basic Shift Register

```
module shift_reg (input [3:0] d, input clk, ld, rst, l_r, s_in,
                  output reg [3:0] q);
   always @( posedge clk ) begin
      if( rst )
         #5 q <= 4'b0000;
      else if( ld )
         #5 q <= d;
      else if( l_r )
         #5 q <= {q[2:0], s_in};
      else
         #5 q <= {s_in, q[3:1]};
   end

endmodule
```

Sequential Circuit Description

Functional Registers

A Basic Shift Register

```
module tb_shift;
    // Inputs
    reg [3:0] d;
    reg clk;
    reg ld;
    reg rst;
    reg l_r;
    reg s_in;
    // Outputs
    wire [3:0] q;
    // Instantiate the Unit Under Test (UUT)
    shift_reg uut (
        .d(d),
        .clk(clk),
        .ld(ld),
        .rst(rst),
        .l_r(l_r),
        .s_in(s_in),
        .q(q)
    );
    initial repeat (40) #20 clk = ~clk;
    initial begin
        // Initialize Inputs
        d = 0;
        clk = 0;
        ld = 0;
        rst = 0;
        l_r = 0;
        s_in = 0;
        #10 rst = 1; d = 3;
        #20 rst = 0;
        #40 ld = 1;
        #40 d = 4;
        #40 d = 10;
        #40 ld = 0; l_r = 1;
        #80 s_in = 1;
        #80 l_r = 0;
        #80 s_in = 0;
    end

endmodule
```

Sequential Circuit Description

Functional Registers

Universal Shift Register

```
module shift_reg (input clk, rst, r_in, l_in, en, s1, s0,
                  inout [7:0] io);
   reg [7:0] q_int;
   assign io = (en) ? q_int : 8'bz;
   always @( posedge clk ) begin
      if( rst )
         #5 q_int = 8'b0;
      else
         case ( {s1,s0} )
            2'b01 : // Shift right
            q_int <= { r_in, q_int[7:1] };
            2'b10 : // Shift left
            q_int <= { q_int[6:0], l_in };
            2'b11 : // Parallel load
            q_int = io;
            default : // Do nothing
            q_int <= q_int;
         endcase
      end

endmodule
```

Sequential Circuit Description

Functional Registers

Universal Shift Register

```
module tb_shift;
    // Inputs
    reg clk;
    reg rst;
    reg r_in;
    reg l_in;
    reg en;
    reg s1;
    reg s0;
    // Bidirs
    wire [7:0] io;
    // Instantiate the Unit Under Test (UUT)
    shift_reg uut (
        .clk(clk),
        .rst(rst),
        .r_in(r_in),
        .l_in(l_in),
        .en(en),
        .s1(s1),
        .s0(s0),
        .io(io) );
    initial repeat (40) #20 clk = ~clk;
    initial begin
        // Initialize Inputs
        clk = 0;
        rst = 0;
        r_in = 0;
        l_in = 0;
        en = 0;
        s1 = 0;
        s0 = 0;
        #10 rst = 1;
        #20 rst = 0; io = 5;
        #40 s1 = 0; s0 = 1;
        #40 r_in = 1; en = 1;
        #80 io = 10;
        #40 s1 = 1; s0 = 0;
        #40 l_in = 1;
        #80 s1 = 1; s0 = 1;
        #40 io = 12;
        #40 en = 0;
    end

endmodule
```

Sequential Circuit Description

Separate register&combinational blocks

Multi-bit Shifter with Separate Register Block