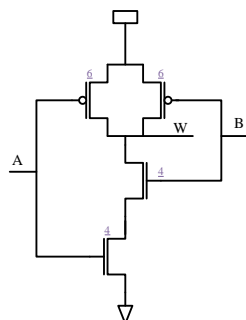
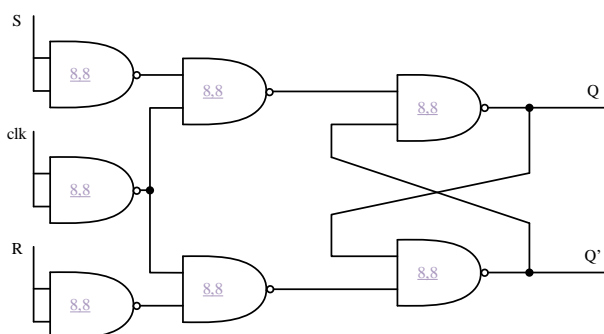


۱

در ابتدا باید ببینیم گیت nand چه تأخیرهایی با توجه به delay داده شده در صورت پروژه خواهد داشت. با توجه به ساختار زیر که برای nand در نظر گرفته می‌شود، بدترین تأخیر to1 و to0 برای یک nand برابر ۸ نانوثانیه می‌باشد.



با توجه به خواسته سوال (active low S, R and clock input with nands) ساختاری مشابه شکل زیر برای این SR-latch متصور می‌شود.



برای تعریف کردن این ماژول در سیستم وریلاگ، کد زیر نوشته شد.

```

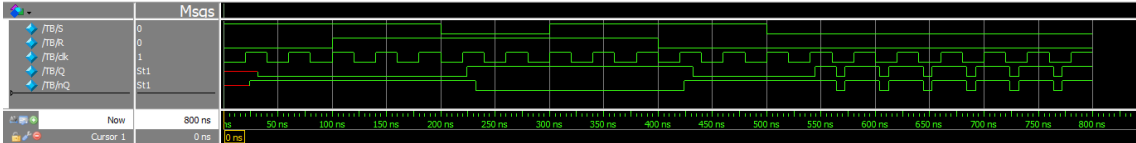
1  `timescale 1ns/1ns
2
3  module SR_latch(input S, R, clk, output Q, nQ);
4      wire i, j, nS, nR;
5      nand #8 not_clk(nclk,clk,clk),
6              not_S(nS,S,S),
7              not_R(nR,R,R),
8              nand1(i,nS,nclk),
9              nand2(j,nR,nclk),
10             nand3(Q,i,nQ),
11             nand4(nQ,j,Q);
12  endmodule

```

انتظار می‌رود این ماژول از جدول درستی زیر پیروی کند.

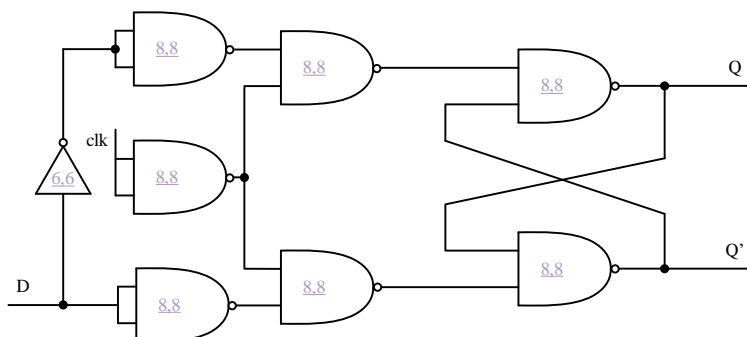
clk	S	R	Q+	Q'+
1	-	-	Q	Q'
0	1	1	Q	Q'
0	1	0	0	1
0	0	1	1	0
0	0	0	*	*

لازم به ذکر است تست این ماژول به طور کامل در قسمت بعدی (قسمت ۲) انجام شده است.

شماره سوال	محمد مهدی معینی منش ۸۱۰۱۹۸۴۷۵	سیستم‌های دیجیتال CA4	تاریخ تحویل: ۳روز + ۱۴۰۰/۳/۹	۲/۱۶
۲	<p>برای بررسی نتیجه، تست‌بنچ زیر را اجرا می‌کنیم. مشاهده می‌گردد که خروجی‌ها دقیقاً مطابق انتظار است. در قسمت پایانی هر دو ورودی S و R صفر می‌شوند تا از دست رفتن حافظه را مشاهده کنیم.</p> <div><pre>1 module TB (); 2 reg S = 1, R = 0, clk = 0; //First Initialize 3 wire Q, nQ; 4 SR_latch my_ic(.S(S), .R(R), .clk(clk), .Q(Q), .nQ(nQ)); 5 always #20 clk = ~clk; 6 initial begin 7 #100 R = 1; 8 #100 S = 0; 9 #100 S = 1; 10 #100 R = 0; 11 #100 S = 0; 12 #300 \$stop; 13 end 14 endmodule</pre></div> <div></div>			

۳

برای آنکه D-latch به همراه کلاک داشته باشیم، به کمک یک not مدار را به صورت زیر بازطراحی می‌کنیم (از آنجا که در صورت پروژه خواسته شده به کمک یک inverter اضافه این طراحی صورت گیرد، مدار به شکل زیر درآمد و گرنه می‌توانستیم بدون استفاده از آن نیز ساختاری ساده‌تر رسم کنیم). تأخیر این not به سادگی برای to1 و to0، ۶ نانوثانیه به دست می‌آید.



برای تعریف کردن این ماژول در سیستم وریلاگ، کد زیر نوشته شد.

```

1  `timescale 1ns/1ns
2
3  module D_latch(input D, clk, output Q, nQ);
4      logic i, j, nD1, nD2, nclk, nnD;
5      nand #8 Dnot(nD2, D, D),
6              Dnotnot(nnD, nD1, nD1),
7              clknot(nclk, clk, clk),
8              nand1(i, nnD, nclk),
9              nand2(j, nD2, nclk),
10             nand3(Q, i, nQ),
11             nand4(nQ, j, Q);
12     not #6 SRnot(nD1, D);
13 endmodule

```

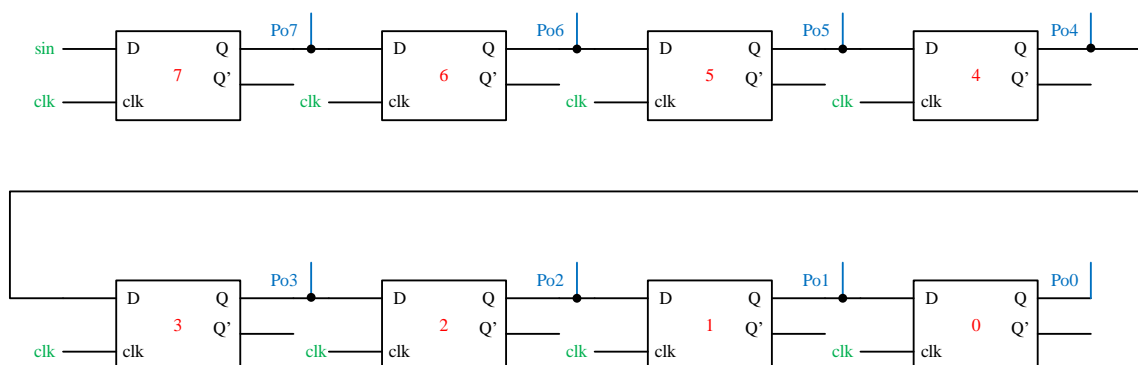
انتظار می‌رود این ماژول از جدول درستی زیر پیروی کند.

clk	D	Q+
1	-	Q
0	1	1
0	0	0

۴/۱۶	<div>تاریخ تحویل:</div> <div>۳روز + ۱۴۰۰/۳/۹</div>	<div>سیستم‌های دیجیتال</div> <div>CA4</div>	<div>محمد مهدی معینی منش</div> <div>۸۱۰۱۹۸۴۷۵</div>	<div>شماره</div> <div>سوال</div>
			<div>در اینجا، چند حالت متفاوت، تست و بررسی می‌شود تا نشان داده شود مازول به درستی کار می‌کند.</div> <div>۳</div>	

۴

برای ساختن شیفت رجیستر خواسته شده مدار زیر بسته شد.



برای تعریف کردن این ماژول در سیستم وریلاگ، کد زیر نوشته شد. تست این ماژول در قسمت‌های بعدی انجام می‌شود.

```
1  `timescale 1ns/1ns
2
3  module Shift_Reg_8bit(input sin, clk, output [7:0] Po);
4      wire [8:0] inputs;
5      assign inputs[8] = sin;
6      assign Po [7:0] = inputs [7:0];
7      genvar i;
8      generate
9          for(i=0;i<8;i=i+1)begin:datches
10             D_latch dlatch(.D(inputs[8-i]), .clk(clk), .Q(inputs[7-i]));
11         end
12     endgenerate
13 endmodule
```

شماره سوال	محمد مهدی معینی منش ۸۱۰۱۹۸۴۷۵	سیستم‌های دیجیتال CA4	تاریخ تحویل: ۳روز + ۱۴۰۰/۳/۹	۶/۱۶
---------------	----------------------------------	--------------------------	---------------------------------	------

۵

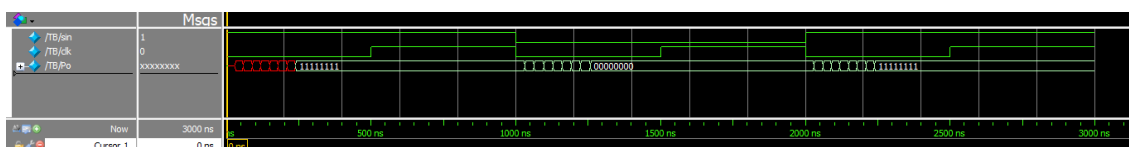
جهت تست و شبیه‌سازی ماژول قسمت قبل، تست‌بنچی که در تصویر زیر مشاهده می‌شود، نوشته شد.

```

1  module TB ();
2      reg sin = 1, clk = 0; //First Initialize
3      wire [7:0] Po;
4      Shift_Reg_8bit my_ic(sin, clk, Po);
5      always #500 clk = ~clk;
6      initial begin
7          #1000 sin=0;
8          #1000 sin=1;
9          #1000 $stop;
10     end
11 endmodule

```

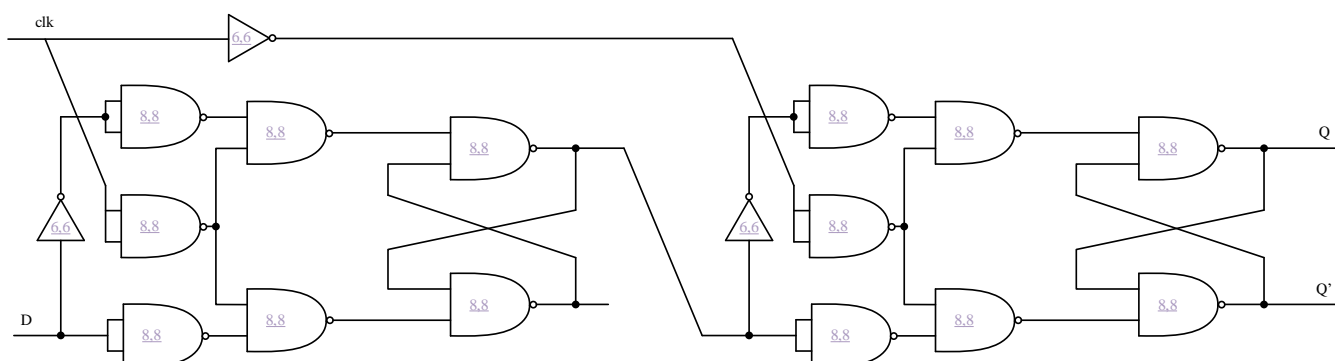
شکل موج خروجی به صورت زیر می‌باشد.



مخصوصاً زمان کلاک‌ها طولانی در نظر گرفته شد تا ایرادی که این شیفت‌رجیستر دارد، نمایان شود. همانطور که مشاهده می‌گردد وقتی کلاک صفر می‌شود، با سرعت بالا ورودی به انتهای شیفت‌رجیستر می‌رسد و هیچ کنترلی روی داده‌ها وجود ندارد.

۶

مدار قطعه خواسته شده به صورت زیر می‌باشد.



برای تعریف کردن این ماژول در سیستم وریلاگ، کد زیر نوشته شد.

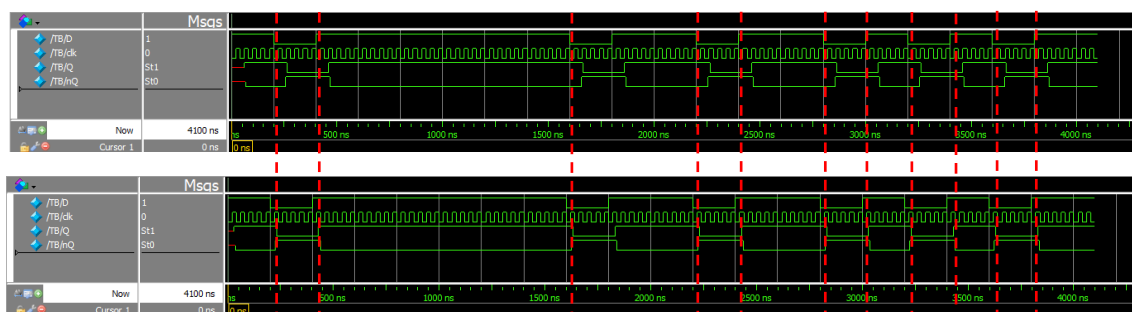
```

1  `timescale 1ns/1ns
2
3  module MSDFF(input D, clk, output Q, nQ);
4      wire mid, nclk;
5      not #6 notclk(nclk, clk);
6      D_latch first_part(.D(D), .clk(clk), .Q(mid));
7      D_latch second_part(.D(mid), .clk(nclk), .Q(Q), .nQ(nQ));
8  endmodule

```

انتظار می‌رود این ماژول درست مانند بخش سوم کار کند ولی تغییرات خروجی صرفاً با صفر شدن کلاک نباشد. بلکه کلاک باید یک بار صفر و سپس صفر شود تا خروجی تغییر کند.

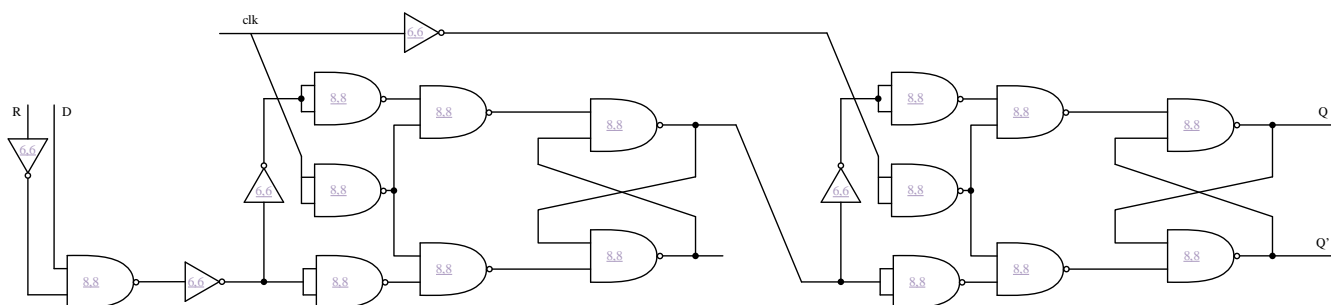
برای تست ماژول از همان تست‌بنچ سوال ۳ استفاده می‌شود. مشاهده می‌گردد که خروجی با تأخیر بیشتری تغییر می‌کند (منتظر یک شدن کلاک می‌ماند).



برای مقایسه، تصویر دوم مربوط به قسمت سوم (D latch) می‌باشد و تصویر بالای آن مربوط به همین قسمت (Master Slave D FF).

۷

برای آن که ریستی بسازیم که هنگام یک بودن، خروجی بدون در نظر گرفتن داده D، صفر شود، طبق جدول درستی که در ادامه آمده است، می‌توان از مدار زیر استفاده کرد (ترجیح بر آن است که از not و nand استفاده شود).



برای تعریف کردن این ماژول در سیستم وریلاگ، کد زیر نوشته شد.

```

1  `timescale 1ns/1ns
2
3  module MSDFFSR(input D, clk, rs, output Q, nQ);
4      wire mid, ni, nrs, nclk;
5      not #6 notclk(nclk, clk),
6          noti(ni, i),
7          notR(nrs, rs);
8      nand #8 nand1(i, D, nrs);
9      D_latch first_part(.D(ni), .clk(clk), .Q(mid));
10     D_latch second_part(.D(mid), .clk(nclk), .Q(Q), .nQ(nQ));
11 endmodule

```

انتظار می‌رود این ماژول از جدول درستی زیر پیروی کند به طوری که $Q+$ را معادل Q بدانیم وقتی کلاک یک بار صفر و پس از آن یک شده‌است.

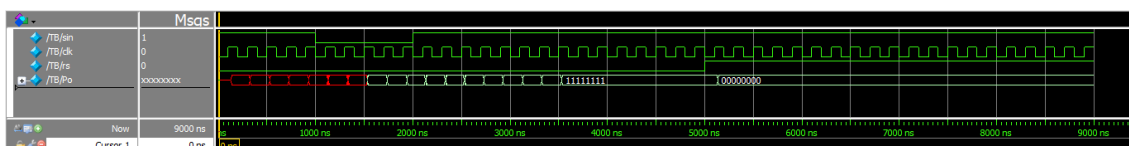
rs	D	$Q+$
1	-	0
0	1	1
0	0	0

شماره سوال	محمد مهدی معینی منش ۸۱۰۱۹۸۴۷۵	سیستم‌های دیجیتال CA4	تاریخ تحویل: ۳روز + ۱۴۰۰/۳/۹	۹/۱۶
۷	<p>در اینجا، چند حالت متفاوت، تست و بررسی می‌شود تا نشان داده شود ماژول به درستی کار می‌کند.</p> <div><pre>13 module TB (); 14 reg D = 1, rs = 0, clk = 0; //First Initialize 15 wire Q, nQ; 16 MSDFFSR my_ic(D, clk, rs, Q, nQ); 17 always #50 clk = ~clk; 18 initial begin 19 #200 rs = 1; 20 #200 D = 0; 21 #200 rs = 0; 22 #200 D = 1; 23 #200 rs = 1; 24 #200 \$stop; 25 end 26 endmodule</pre></div> <div></div>			

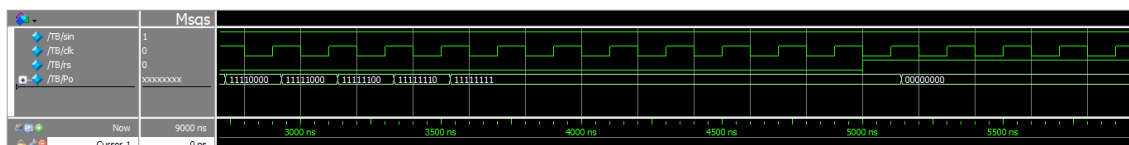
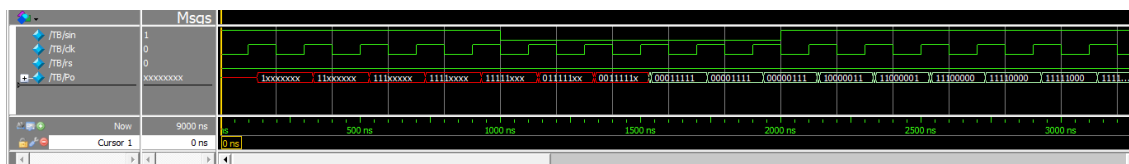
شماره سوال	محمد مهدی معینی منش ۸۱۰۱۹۸۴۷۵	سیستم‌های دیجیتال CA4	تاریخ تحویل: ۳روز + ۱۴۰۰/۳/۹	۱۰/۱۶
۸	<p>همچون قسمت ۴، کدی جهت طراحی شیفت رجیستر نوشته شد. منتهی این بار به جای استفاده از D latch از Master Slave D Flip Flop استفاده شده است.</p> <pre>1 `timescale 1ns/1ns 2 3 module Shift_Reg_8bit_sr(input sin, clk, rs, output [7:0] Po); 4 wire [8:0] inputs; 5 assign inputs[8] = sin; 6 assign Po [7:0] = inputs [7:0]; 7 genvar i; 8 generate 9 for(i=0;i<8;i=i+1)begin:MSDFFSRs 10 MSDFFSR MSDFFSRi(.D(inputs[8-i]), .clk(clk), .rs(rs), .Q(inputs[7-i])); 11 end 12 endgenerate 13 endmodule</pre> <p>جهت تست و شبیه‌سازی این ماژول، تست‌بنچی که در تصویر زیر مشاهده می‌شود، نوشته شد.</p> <pre>15 module TB (); 16 reg sin = 1, clk = 0, rs = 0; //First Initialize 17 wire [7:0] Po; 18 Shift_Reg_8bit_sr my_ic(sin, clk, rs, Po); 19 always #100 clk = ~clk; 20 initial begin 21 #1000 sin=0; 22 #1000 sin=1; 23 #3000 rs =1; 24 #4000 \$stop; 25 end 26 endmodule</pre>			

۸

شکل موج خروجی به صورت زیر می‌باشد.



برای واضح‌تر بودن اتفاقاتی که رخ می‌دهد. دو تصویر زیر هنگامی که در نقاط مختلف شکل موج بزرگنمایی شده، آورده شده است.



اکنون کاملاً واضح است که هر بار صفر و یک شدن کلاک سبب می‌شود خروجی یکی به جلو شیفت کند. همانطور که قبلاً نیز گفته شد، دیگر صفر شدن کلاک به تنهایی سبب جلو رفتن ورودی به اندازه نامعلوم نمی‌شود و منتظر می‌ماند تا کلاک یک شود. این مثال درست شبیه این است که چند نفر می‌خواهند وارد خانه شوند و اگر در ورودی برای زمان محدود باز شود بسته به شرایط ممکن است هیچکس نتواند وارد شود یا اینکه تعداد نامعلومی از اشخاص وارد شوند. اما می‌توان یک در ثانویه گذاشت که هر بار یک نفر بین دو در گیر بیفتد و با باز شدن در ثانویه فقط کسی که بین دو در است بتواند وارد شود.

شماره سوال	محمد مهدی معینی منش ۸۱۰۱۹۸۴۷۵	سیستم‌های دیجیتال CA4	تاریخ تحویل: ۳روز + ۱۴۰۰/۳/۹	۱۲/۱۶
------------	----------------------------------	--------------------------	---------------------------------	-------

۹

برای تعریف کردن این ماژول در سیستم وریرلاگ به کمک always statement کد زیر نوشته شد.

```

1  `timescale 1ns/1ns
2
3  module Shift_Reg_8bit_sr2(input sin, clk, rs, output reg [7:0] Po);
4      always @(negedge clk) begin
5          if (rs)
6              Po = 8'd0;
7          else
8              Po = {sin, Po [7:1]};
9      end
10 endmodule

```

برای تست این ماژول از همان تست‌بنچی که در سوال ۸ نوشته شد، استفاده می‌گردد. منتهی با توجه به اینکه خروجی Po این بار از نوع reg است، تغییرات کوچکی در تست‌بنچ صورت گرفته است.

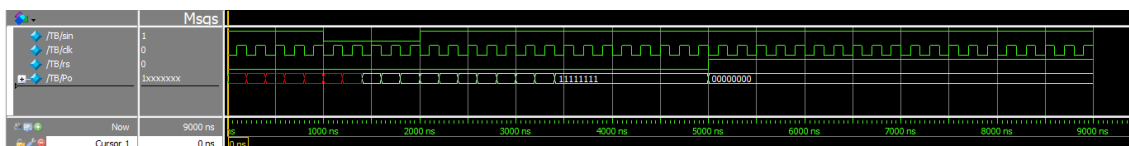
```

12 module TB ();
13     reg sin = 1, clk = 0, rs = 0; //First Initialize
14     reg [7:0] Po;
15     Shift_Reg_8bit_sr2 my_ic(sin, clk, rs, Po);
16     always #100 clk = ~clk;
17     initial begin
18         #1000 sin=0;
19         #1000 sin=1;
20         #3000 rs =1;
21         #4000 $stop;
22     end
23 endmodule

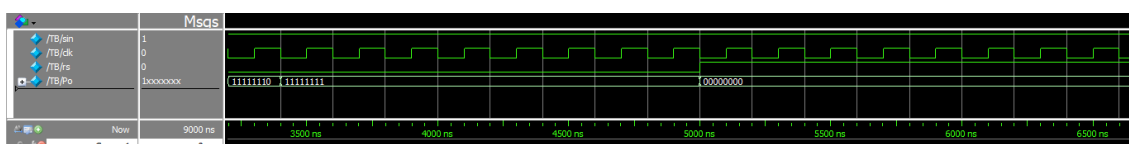
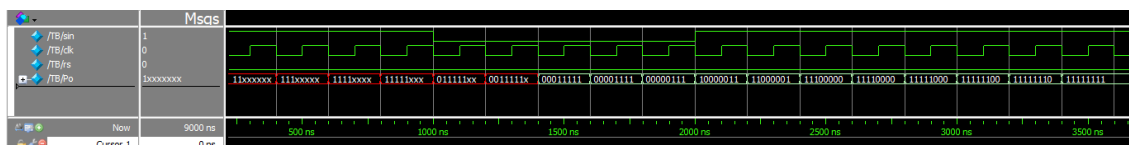
```

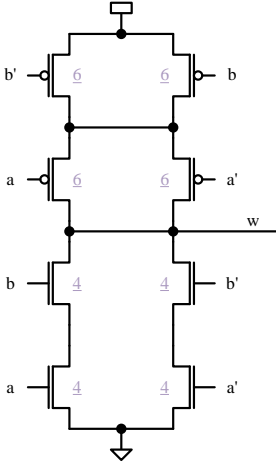
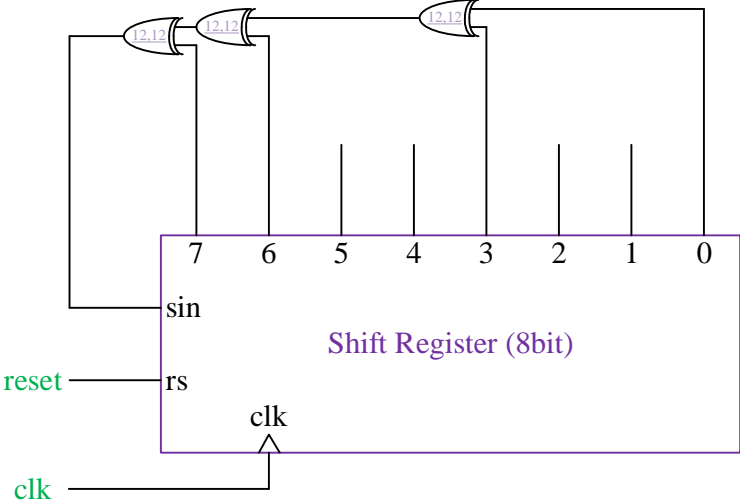
۹

شکل موج خروجی به صورت زیر می‌باشد.



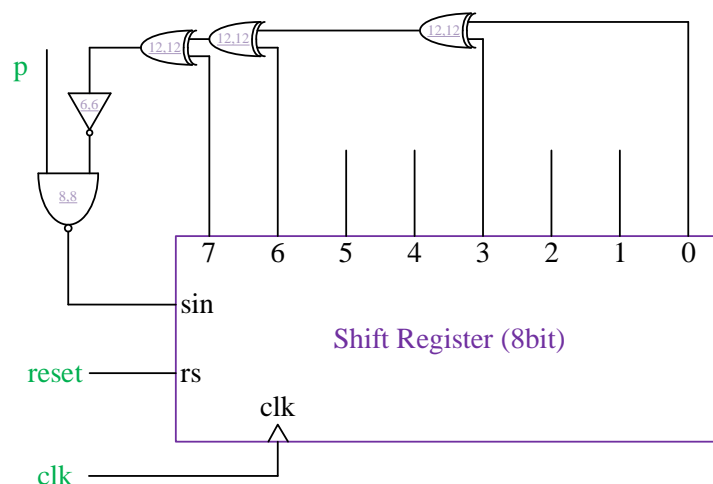
برای واضح‌تر بودن اتفاقاتی که رخ می‌دهد. دو تصویر زیر هنگامی که در نقاط مختلف شکل موج بزرگنمایی شده، آورده شده است.



شماره سوال	محمد مهدی معینی منش	سیستم‌های دیجیتال	تاریخ تحویل:
۸۱۰۱۹۸۴۷۵	CA4	۱۴۰۰/۳/۹ + ۳روز	۱۴/۱۶
۱۰	<p>برای ساختن مدار خواسته شده، از xor استفاده شده است. بنابراین لازم است ابتدا این ساختار آن را رسم و بدترین تأخیرهای آن را محاسبه کنیم.</p> <p>برای بدترین تأخیر to1 و to0 برابر 12ns است و داریم:</p> <div></div> <p>مدار بسته شده به صورت زیر می‌باشد.</p> <div></div>		

۱۰

مشخص است که اگر مقادیر اولیه Po تماماً صفر باشد، با پالس‌های وارد شده از طرف کلاک، تغییری در خروجی مشاهده نمی‌گردد و خروجی همیشه صفر باقی می‌ماند. بنابراین، مدار به شکل زیر بازطراحی می‌گردد تا بتوان با صفر کردن p (preset) ورودی اولیه را یک کرد.



برای تعریف کردن این ماژول در سیستم وریلاگ کد زیر نوشته شد.

```

1  `timescale 1ns/1ns
2
3  module LFSR(input p, clk, rs, output [7:0] Po);
4      wire x, y, z, nz, t;
5      Shift_Reg_8bit_sr SR1(t, clk, rs, Po);
6      xor #12 xor1(x, Po[0], Po[3]),
7          xor2(y, x, Po[6]),
8          xor3(z, y, Po[7]);
9      not #6 not1(nz, z);
10     nand #8 nand1(t, nz, p);
11 endmodule

```

تست‌بنچ این ماژول نیز به شکل زیر طراحی گردید.

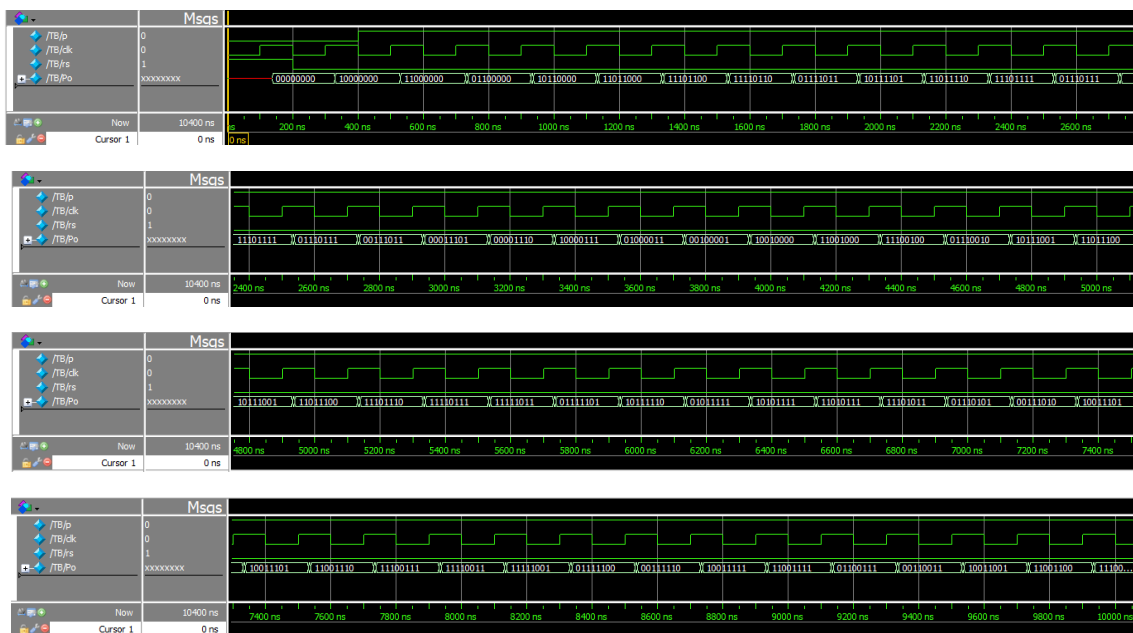
```

13 module TB ();
14     reg p = 0, clk = 0, rs = 1; //First Initialize
15     wire [7:0] Po;
16     LFSR my_ic(p, clk, rs, Po);
17     always #100 clk = ~clk;
18     initial begin
19         #200 rs = 0;
20         #200 p = 1;
21         #10000 $stop;
22     end
23 endmodule

```

۱۰

چند تصویر از لحظات مختلف خروجی در اینجا آورده شده است و مشاهده می‌گردد اعداد تصادفی در حال تولید هستند.



حداکثر دوره تناوب برای اعداد ساخته شده برابر است با 2^{n-1} که n بالاترین درجه‌ای از x در چندجمله‌ای داده شده است به شرط آنکه چندجمله‌ای اول باشد یعنی ریشه حقیقی نداشته باشد.