

Flip flops, Counters and Registers

CASEST



हैदराबाद विश्वविद्यालय
University of Hyderabad

Lab Report No. 4 + 5

MV407: IC Design Lab-1 (Digital)

Submitted by:

Full Name	Enrollment No.
------------------	-----------------------

Abhinav M	23PMMT11(@uohyd.ac.in)
-----------	------------------------

Lab Instructor: Dr. Bhawna Gomber

Teaching Assistant: Ms. Bhargavi

Center for Advanced Studies in Electronics Science and Technology
University of Hyderabad
Hyderabad, India
November 22, 2023



Contents

1 Asynchronous D Flip flop	1
1.1 Objective	1
1.2 Block diagram	1
1.3 Code	1
1.4 Elaborated design	3
1.5 Timing diagram	3
1.6 Remarks	3
2 Synchronous D flip flop	4
2.1 Objective	4
2.2 Block diagram	4
2.3 Code	4
2.4 Elaborated design	5
2.5 Timing diagram	6
2.6 Remarks	6
3 Synchronous D flip flop with reset	7
3.1 Objective	7
3.2 Block diagram	7
3.3 Code	7
3.4 Elaborated design	8
3.5 Timing diagram	8
3.6 Remarks	8
4 8-bit data register async rst	9
4.1 Objective	9
4.2 Code	9
4.3 Elaborated design	10
4.4 Timing diagram	10
4.5 Remarks	10
5 8-bit data register sync rst	11
5.1 Objective	11
5.2 Code	11
5.3 Elaborated design	12
5.4 Timing diagram	12
5.5 Remarks	12
6 4-bit PIPO register	13
6.1 Objective	13
6.2 Block diagram	13
6.3 Code	13
6.4 Elaborated design	14
6.5 Timing diagram	15
6.6 Remarks	15
7 4-bit SISO register	16
7.1 Objective	16
7.2 Block diagram	16
7.3 Code	16
7.4 Elaborated design	18
7.5 Timing diagram	18
7.6 Remarks	18

8	4-bit Universal Shift Register	19
8.1	Objective	19
8.2	Block diagram	19
8.3	Code	19
8.4	Elaborated design	21
8.5	Timing diagram	21
8.6	Remarks	21
9	8-bit sync up-counter	22
9.1	Objective	22
9.2	Code	22
9.3	Elaborated design	23
9.4	Timing diagram	23
9.5	Remarks	23
10	8-bit async down-counter	24
10.1	Objective	24
10.2	Code	24
10.3	Elaborated design	25
10.4	Timing diagram	25
10.5	Remarks	25
11	References	26
A	Appendix A: GitHub Repo	27

1 | Asynchronous D Flip flop

1.1 | Objective

Design a circuit for Asynchronous D Flip flop. Verify its functionality using testbench.

1.2 | Block diagram

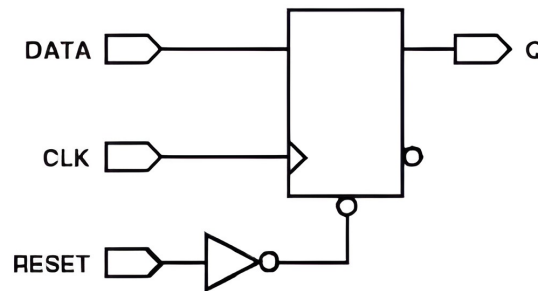


Figure 1.1: Block diagram async DFF

1.3 | Code

1.3.1 | Design

```
1
2 `timescale 1ns / 1ps
3
4 // Company : CASEST, University of Hyderabad:
5 // Author : Abhinav M
6 //
7 // Create Date: 10/25/2023 02:52:37 PM
8 // Design Name:
9 // Module Name: CB_DFF_async
10 // Project Name:
11 // Target Devices:
12 // Tool Versions:
13 // Description:
14 //
15 // Dependencies:
16 //
17 // Revision:
18 // Revision 0.01 - File Created
19 // Additional Comments:
20 //
21 //////////////////////////////////////////////////
22
23
24
25 module CB_DFF_async(input D,clk,rst,en,output reg Q);
26
27
28 always @(posedge clk or posedge rst)
29 begin
30 if(~rst & en)
31 Q <= D;
32 else
33 Q <= 1'b0;
34 end
35
36 endmodule
```

1.3.2 | Testbench



```
1
2 `timescale 1ns / 1ps
3 //////////////////////////////////////////////////
4 // Company:
5 // Author: Abhinav M
6 //
7 // Create Date: 10/25/2023 03:02:13 PM
8 // Design Name:
9 // Module Name: tb_CB_DFF_sync
10 // Project Name:
11 // Target Devices:
12 // Tool Versions:
13 // Description:
14 //
15 // Dependencies:
16 //
17 // Revision:
18 // Revision 0.01 - File Created
19 // Additional Comments:
20 //
21 //////////////////////////////////////////////////
22
23 module tb_CB_DFF_async();
24 reg D;
25 reg clk;
26 wire Q;
27 reg en;
28 reg rst;
29
30 CB_DFF_async uut(D,clk,rst,en,Q);
31
32 initial
33 begin
34     clk=0;
35     rst=0;
36     en=1;
37     forever #10 clk = ~clk;
38 end
39
40 initial
41 begin
42     forever #100 rst = ~rst;
43 end
44
45
46 initial
47 begin
48
49     D <= 0;
50     #50;
51
52     D <= 1;
53     #50;
54
55
56     D <= 0;
57     #50
58
59     en=0;
60     D<= 1;
61
62     #50 $finish;
63 end
64 endmodule
```

1.4 | Elaborated design

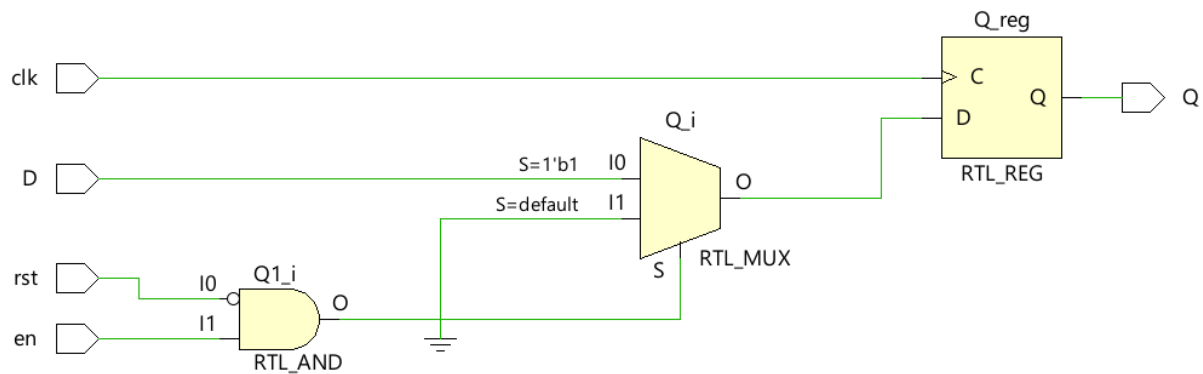


Figure 1.2: Elaborated design

1.5 | Timing diagram

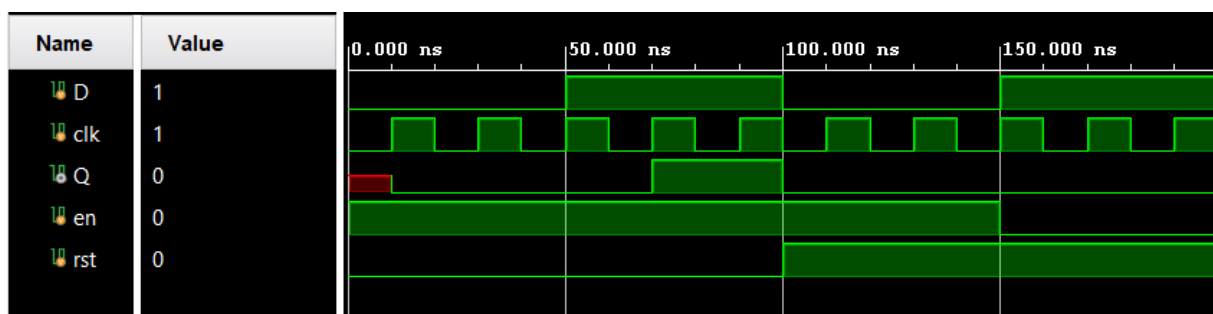


Figure 1.3: Timing diagram

1.6 | Remarks

Circuit was designed and tested successfully, expected functionality was achieved.

2 | Synchronous D flip flop

2.1 | Objective

Design a circuit for Synchronous D flip flop. Verify its functionality using testbench.

2.2 | Block diagram

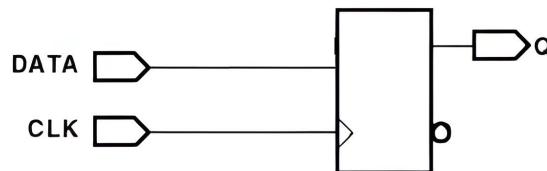


Figure 2.1: Block diagram sync DFF

2.3 | Code

2.3.1 | Design

```
1
2 `timescale 1ns / 1ps
3 //////////////////////////////////////////////////
4 // Company : CASEST, University of Hyderabad:
5 // Author : Abhinav M
6 //
7 // Create Date: 10/25/2023 02:45:45 PM
8 // Design Name:
9 // Module Name: CB_DFF_sync
10 // Project Name:
11 // Target Devices:
12 // Tool Versions:
13 // Description:
14 //
15 // Dependencies:
16 //
17 // Revision:
18 // Revision 0.01 - File Created
19 // Additional Comments:
20 //
21 //////////////////////////////////////////////////
22
23 module CB_DFF_sync(input D,clk,en,output reg Q);
24
25
26
27 always @(posedge clk & en)
28 begin
29   Q <= D;
30 end
31
32
33
34
35
36 endmodule
```

2.3.2 | Testbench

```
1
2 `timescale 1ns / 1ps
3 //////////////////////////////////////////////////
4 // Company:
5 // Author: Abhinav M
6 //
7 // Create Date: 10/25/2023 03:02:13 PM
8 // Design Name:
```

```

9  // Module Name: tb_CB_DFF_sync
10 // Project Name:
11 // Target Devices:
12 // Tool Versions:
13 // Description:
14 //
15 // Dependencies:
16 //
17 // Revision:
18 // Revision 0.01 - File Created
19 // Additional Comments:
20 //
21 ///////////////////////////////////////////////////
22
23 module tb_CB_DFF_sync();
24 reg D;
25 reg clk;
26 wire Q;
27 reg en;
28
29 CB_DFF_sync uut(D,clk,en,Q);
30
31 initial
32 begin
33     clk=0;
34     en=1;
35     forever #10 clk = ~clk;
36 end
37
38
39 initial
40 begin
41
42     D <= 0;
43     #50;
44
45     D <= 1;
46     #50;
47
48
49     D <= 0;
50     #50
51
52     en=0;
53     D<= 1;
54
55     #50 $finish;
56 end
57 endmodule

```

2.4 | Elaborated design

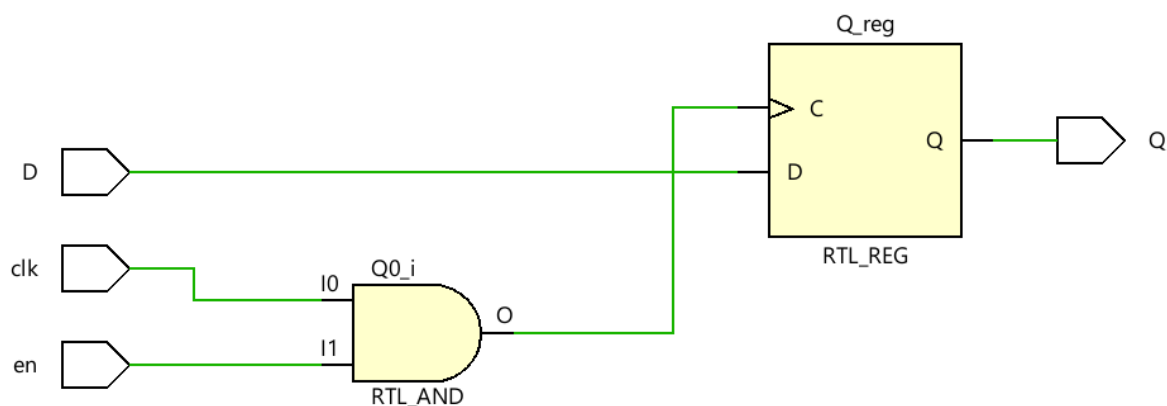


Figure 2.2: Elaborated design

2.5 | Timing diagram

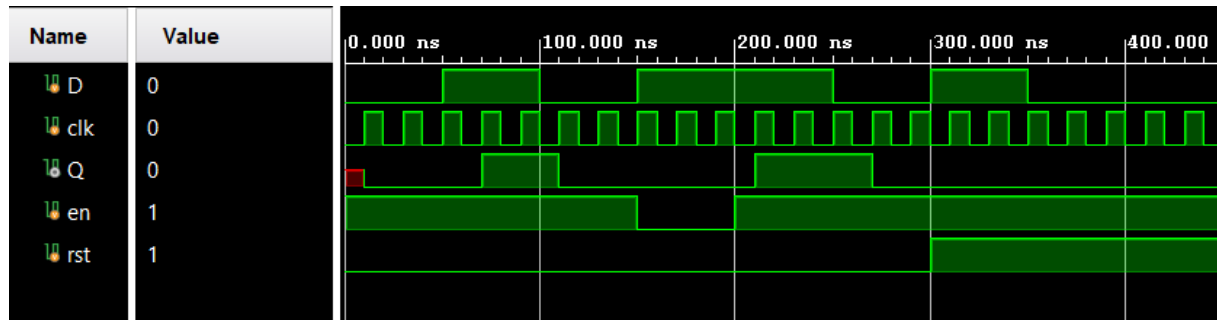


Figure 2.3: Timing diagram

2.6 | Remarks

Circuit was designed and tested successfully, expected functionality was achieved.

3 | Synchronous D flip flop with reset

3.1 | Objective

Design a circuit for Synchronous D flip flop with reset. Verify its functionality using testbench.

3.2 | Block diagram

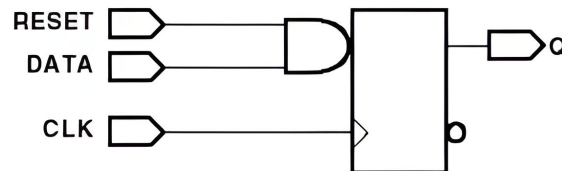


Figure 3.1: Block diagram Sync DFF with rst

3.3 | Code

3.3.1 | Design

```
1  `timescale 1ns / 1ps
2  ///////////////////////////////////////////////////////////////////
3  // Company : CASEST, University of Hyderabad:
4  // Author : Abhinav M
5  //
6  // Create Date: 10/25/2023 02:49:59 PM
7  // Design Name:
8  // Module Name: CB_DFF_syncwrst
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22 module CB_DFF_syncwrst(input D,clk,rst,en,output reg Q);
23
24 always @(posedge clk)
25 begin
26 if(~rst & en)
27   Q <= D;
28 else
29   Q <= 1'b0;
30 end
31
32 endmodule
```

3.3.2 | Testbench

```
1
2 module tb_CB_DFF_syncwrst();
3 reg D;
4 reg clk;
5 wire Q;
6 reg en;
7 reg rst;
8 CB_DFF_syncwrst uut(D,clk,rst,en,Q);
9 initial
10 begin
11   clk=0;
12   rst=0;
```

```

13         en=1;
14         forever #10 clk = ~clk;
15     end
16     initial
17     begin
18         D <= 0;
19         #50;
20
21         D <= 1;
22         #50;
23
24         D <= 0;
25         #50;
26
27         en=0;
28
29         D <= 1;
30         #50;
31
32         en=1;
33
34         D <= 1;
35         #50;
36
37         D <= 0;
38         #50;
39
40         rst=1;
41
42         D <= 1;
43         #50;
44
45         D <= 0;
46         #50;
47         #50 $finish;
48     end
49 endmodule

```

3.4 | Elaborated design

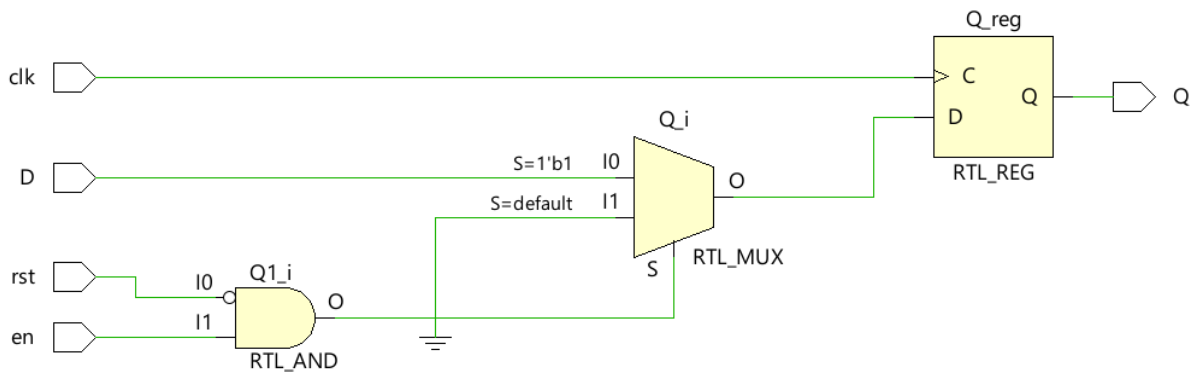


Figure 3.2: Elaborated design

3.5 | Timing diagram

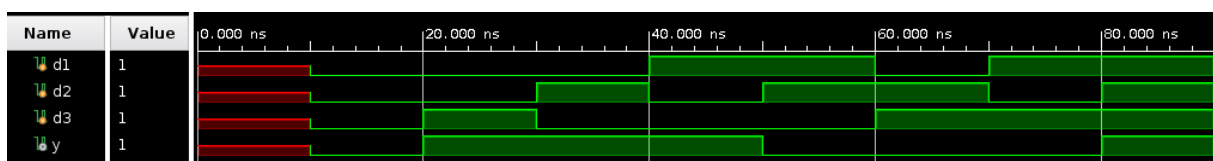


Figure 3.3: Timing diagram.

3.6 | Remarks

Circuit was designed and tested successfully, expected functionality was achieved.



4 | 8-bit data register async rst

4.1 | Objective

Design a circuit for 8-bit data register with async rst. Verify its functionality using testbench.

4.2 | Code

4.2.1 | Design

```
1  `timescale 1ns / 1ps
2  ///////////////////////////////////////////////////////////////////
3  // Company : CASEST, University of Hyderabad:
4  // Author : Abhinav M
5  //
6  // Create Date: 11/01/2023 02:22:58 PM
7  // Design Name:
8  // Module Name: DB_8asynrst
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22 module DB_8asynrst (reset, clk, D, Q);
23 input reset;
24 input clk;
25 input [7:0] D;
26 output [7:0] Q;
27 reg [7:0] Q;
28 always @(posedge clk or posedge reset)
29 if (reset)
30 Q = 0;
31 else
32 Q = D;
33 endmodule
```

4.2.2 | Testbench

```
1
2  `timescale 1ns / 1ps
3  ///////////////////////////////////////////////////////////////////
4  // Company:
5  // Author: Abhinav M
6  //
7  // Create Date: 22.11.2023 21:23:29
8  // Design Name:
9  // Module Name: tb_DB_8_asynrst
10 // Project Name:
11 // Target Devices:
12 // Tool Versions:
13 // Description:
14 //
15 // Dependencies:
16 //
17 // Revision:
18 // Revision 0.01 - File Created
19 // Additional Comments:
20 //
21 ///////////////////////////////////////////////////////////////////
22
23 module tb_DB_8_asynrst();
24 reg clk,rst;
25 reg [7:0] D;
26 wire [7:0] Q;
27 DB_8asynrst uut(rst,clk, D, Q);
28
29
30
31 initial
32 begin
33     clk=0;
34     forever #1 clk = ~clk;
```

```

35     end
36   initial
37   begin
38     rst=1; #5
39
40     rst=0; #5
41
42     D = 8'b10101010; #5
43
44     D = 8'b11111111; #5
45
46     D = 8'b01010101; #5
47
48     #150 $finish;
49   end
50 endmodule

```

4.3 Elaborated design

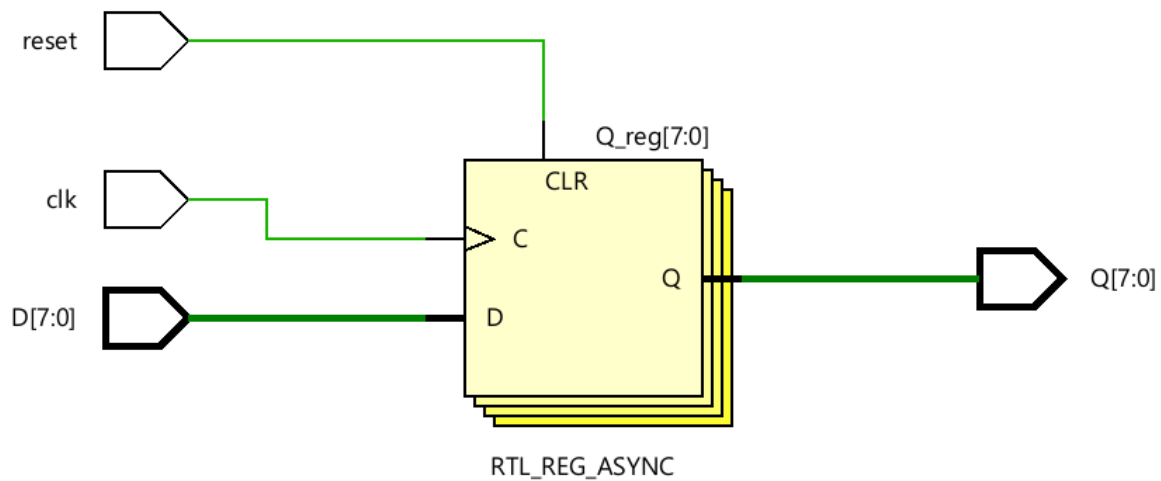


Figure 4.1: Elaborated design

4.4 Timing diagram

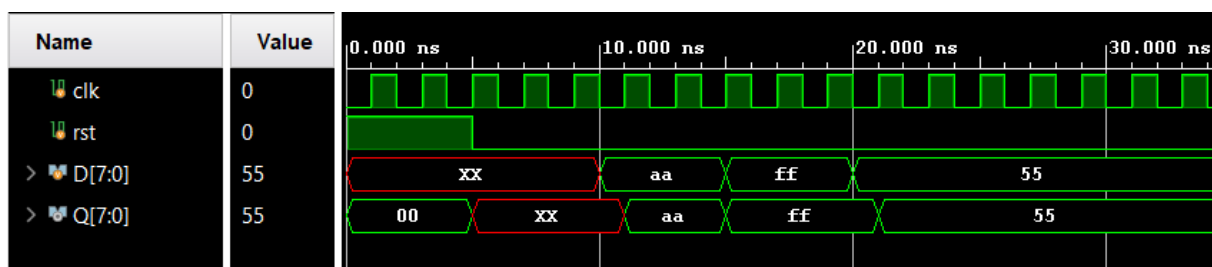


Figure 4.2: Timing diagram

4.5 Remarks

Circuit was designed and tested successfully, expected functionality was achieved.



5 | 8-bit data register sync rst

5.1 | Objective

Design a circuit for 8-bit data register sync rst. Verify its functionality using testbench.

5.2 | Code

5.2.1 | Design

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////
3  // Company : CASEST, University of Hyderabad:
4  // Author : Abhinav M
5  //
6  // Create Date: 11/01/2023 02:40:10 PM
7  // Design Name:
8  // Module Name: DB_8syncrst
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////////////////
21
22 module DB_8syncrst(rst, clk, D, Q);
23 input rst;
24 input clk;
25 input [7:0] D;
26 output [7:0] Q;
27 reg [7:0] Q;
28 always @(posedge clk)
29 if (rst)
30 Q = 0;
31 else
32 Q = D;
33 endmodule
```

5.2.2 | Testbench

```
1
2  `timescale 1ns / 1ps
3  //////////////////////////////////////////////////
4  // Company:
5  // Author: Abhinav M
6  //
7  // Create Date: 22.11.2023 21:23:29
8  // Design Name:
9  // Module Name: tb_DB_8_syncrst
10 // Project Name:
11 // Target Devices:
12 // Tool Versions:
13 // Description:
14 //
15 // Dependencies:
16 //
17 // Revision:
18 // Revision 0.01 - File Created
19 // Additional Comments:
20 //
21 //////////////////////////////////////////////////
22
23
24 module tb_DB_8_syncrst();
25 reg clk,rst;
26 reg [7:0] D;
27 wire [7:0] Q;
28 DB_8syncrst uut(rst,clk, D, Q);
29
30
31 initial
32 begin
33     clk=0;
34     forever #1 clk = ~clk;
```

```

35     end
36   initial
37   begin
38     rst=1; #5
39
40     rst=0; #5
41
42     D = 8'b10101010; #5
43
44     D = 8'b11111111; #5
45
46     D = 8'b01010101; #5
47
48     #150 $finish;
49   end
50 endmodule

```

5.3 Elaborated design

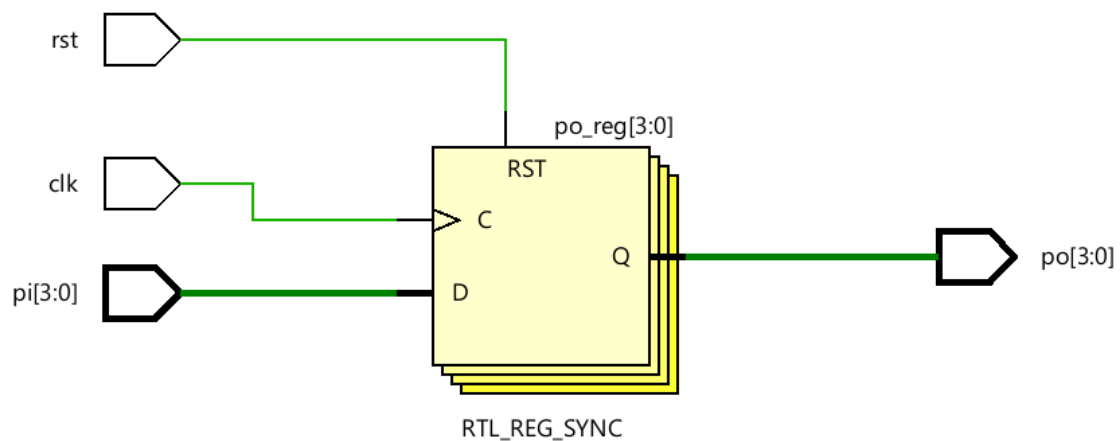


Figure 5.1: Elaborated design

5.4 Timing diagram

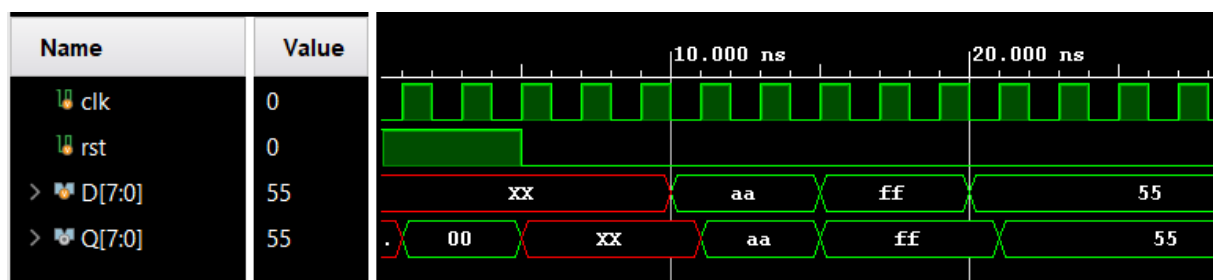


Figure 5.2: Timing diagram

5.5 Remarks

Circuit was designed and tested successfully, expected functionality was achieved.

6 | 4-bit PIPO register

6.1 | Objective

Design a circuit for 4-bit PIPO register. Verify its functionality using testbench.

6.2 | Block diagram

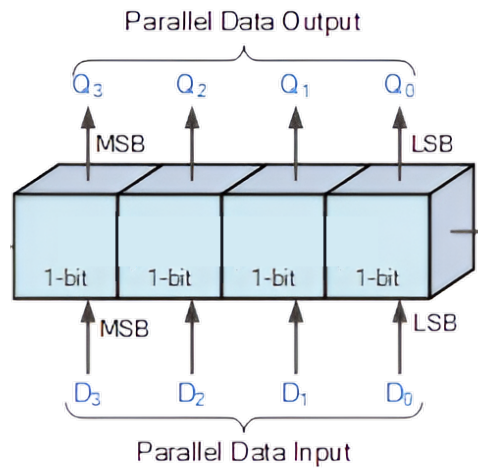


Figure 6.1: 4-bit PIPO register.

6.3 | Code

6.3.1 | Design

```

1  `timescale 1ns / 1ps
2  ///////////////////////////////////////////////////////////////////
3  // Company : CASEST, University of Hyderabad:
4  // Author : Abhinav M
5  //
6  // Create Date: 11/08/2023 02:50:43 PM
7  // Design Name:
8  // Module Name: E_4_PIP0
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22 module E_4_PIP0(input clk,rst, [3:0]pi, output reg [3:0]po);
23
24 always @(posedge clk)
25 begin
26 if (rst)
27 po<= 4'b0000;
28 else
29 po <= pi;
30 end
31 endmodule

```


6.3.2 | Testbench

```

1  `timescale 1ns / 1ps
2  ///////////////////////////////////////////////////////////////////
3  // Company:
4  // Author: Abhinav M
5  //
6  // Create Date: 22.11.2023 21:23:29
7  // Design Name:
8  // Module Name: tb_E_4_PIP0
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22
23 module tb_E_4_PIP0();
24 reg clk,rst;
25 reg [3:0] pi;
26 wire [3:0] po;
27 E_4_PIP0 uut(clk,rst, pi, po);
28
29
30 initial
31     begin
32         clk=0;
33         forever #1 clk = ~clk;
34     end
35 initial
36 begin
37     rst=1; #5
38
39     rst=0; #5
40
41     pi = 4'b1010; #5
42
43     pi = 4'b1111; #5
44
45     pi = 4'b0101; #5
46
47     #150 $finish;
48 end
49 endmodule

```

6.4 | Elaborated design

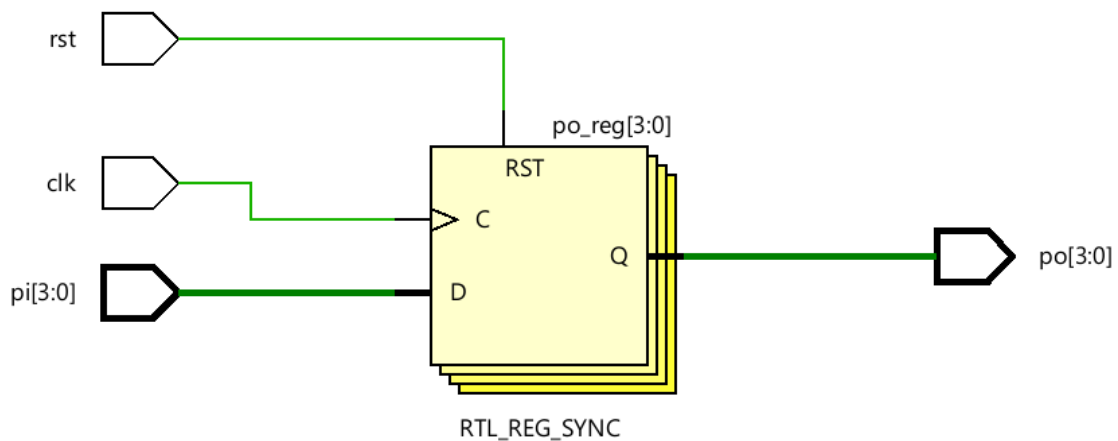


Figure 6.2: Elaborated design

6.5 | Timing diagram

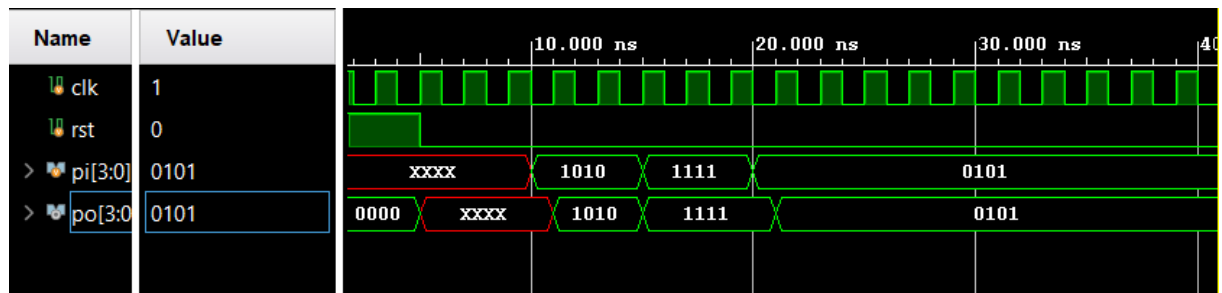


Figure 6.3: Timing diagram

6.6 | Remarks

Circuit was designed and tested successfully, expected functionality was achieved.

7 | 4-bit SISO register

7.1 | Objective

Design a circuit for 4-bit SISO register. Verify its functionality using testbench.

7.2 | Block diagram

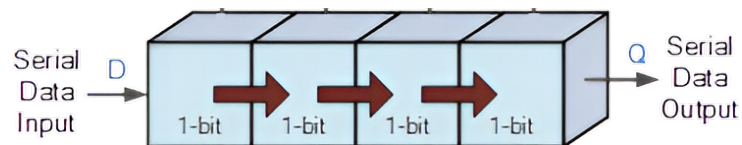


Figure 7.1: 4-bit SISO working (L → R)

7.3 | Code

7.3.1 | Design

```
1  `timescale 1ns / 1ps
2  ///////////////////////////////////////////////////////////////////
3  // Company : CASEST, University of Hyderabad:
4  // Author : Abhinav M
5  //
6  // Create Date: 11/08/2023 02:44:22 PM
7  // Design Name:
8  // Module Name: E_4_SISO
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22 module E_4_SISO(input clk,rst,si,output reg so);
23
24   reg [3:0] tmp;
25
26   always @(posedge clk )
27   begin
28
29
30     if (rst)
31       tmp <= 4'b0000;
32
33     else
34       tmp <= tmp << 1;
35
36     tmp[0] <= si;
37     so = tmp[3];
38
39   end
40 endmodule
41
```



7.3.2 | Testbench

```
1  `timescale 1ns / 1ps
2  ///////////////////////////////////////////////////////////////////
3  // Company:
4  // Author: Abhinav M
5  //
6  // Create Date: 22.11.2023 21:23:29
7  // Design Name:
8  // Module Name: tb_E_4_PIP0
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22
23 module tb_E_4_SISO();
24 reg clk,rst;
25 reg si;
26 wire so;
27 E_4_SISO uut(clk,rst, si, so);
28 initial
29     begin
30         clk=0;
31         forever #5 clk = ~clk;
32     end
33 initial
34     begin
35         rst=1; #10
36
37         rst=0; #10
38
39         si = 1'b1; #5
40
41         si = 1'b1; #5
42
43         si = 1'b0; #5
44
45         si = 1'b0; #5
46
47         si = 1'b1; #5
48
49         si = 1'b1; #5
50
51         si = 1'b0; #5
52
53         si = 1'b0; #5
54
55         si = 1'b1; #5
56
57         si = 1'b1; #5
58
59         si = 1'b1; #5
60
61         si = 1'b1; #5
62
63         si = 1'b1; #5
64
65         si = 1'b1; #5
66
67         si = 1'b1; #5
68
69         si = 1'b1; #5
70
71         si = 1'b1; #5
72
73         si = 1'b1; #5
74         #100 $finish;
75     end
76 endmodule
```

7.4 | Elaborated design

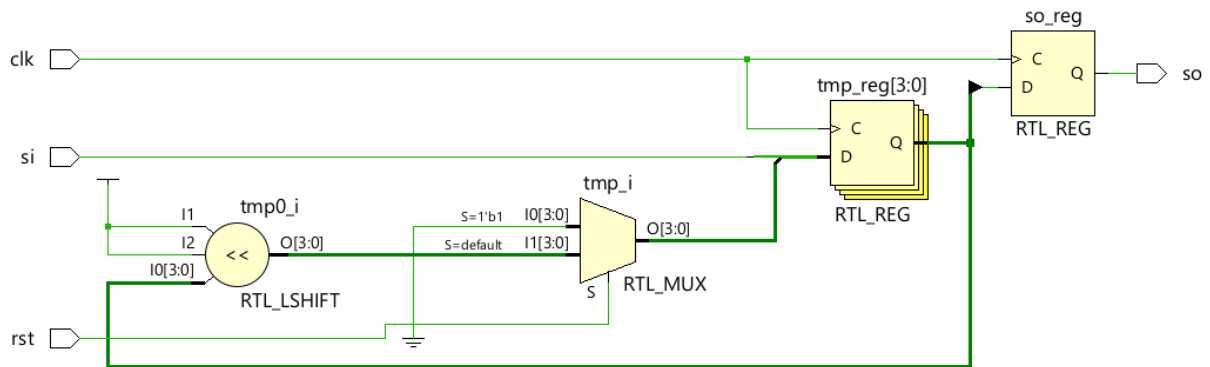


Figure 7.2: Elaborated design

7.5 | Timing diagram

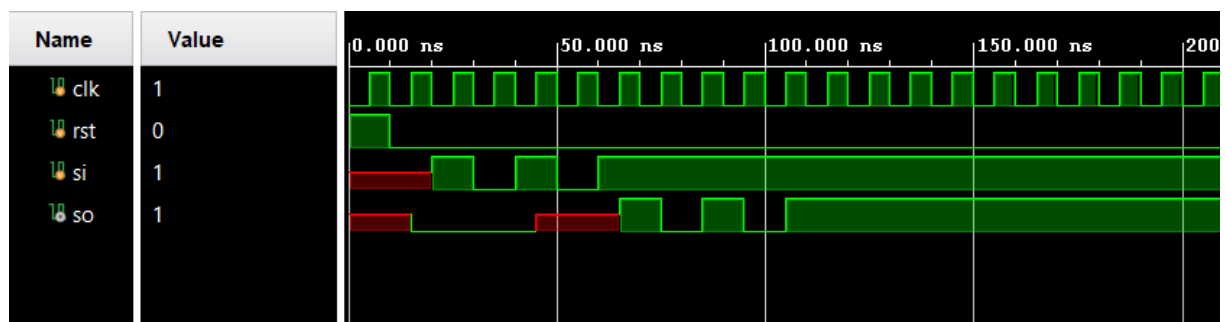


Figure 7.3: Timing diagram

7.6 | Remarks

Circuit was designed and tested successfully, expected functionality was achieved.

8 | 4-bit Universal Shift Register

8.1 | Objective

Design a circuit for 4-bit Universal Shift Register. Verify its functionality using testbench.

8.2 | Block diagram

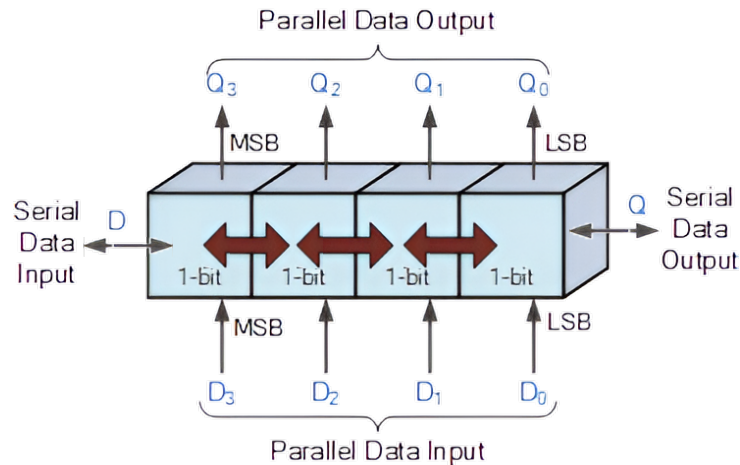


Figure 8.1: 4-bit universal shift register.

8.3 | Code

8.3.1 | Design

```

1  `timescale 1ns / 1ps
2  ///////////////////////////////////////////////////////////////////
3  // Company : CASEST, University of Hyderabad:
4  // Author : Abhinav M
5  //
6  // Create Date: 11/08/2023 04:48:39 PM
7  // Design Name:
8  // Module Name: E_4_USR
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22
23 module E_4_USR(
24     input clk, rst,
25     input [1:0] sel, // select
26     input [3:0] pdi, // parallel data in
27     input sldi, // serial left data in
28     input srdi, // serial right data in
29     output reg [3:0] pdo, //parallel data out
30     output reg [3:0] do,
31     output sldo, // serial left data out
32     output srdo // serial right data out
33 );
34 always@(posedge clk)
35 begin
36     if(rst)
37     begin
38         pdo<=0;
39         do<=0;
40     end
41     else begin

```



```
42     case(sel)
43         2'h1: do <= {srdi,pdo[3:1]}; // Right Shift
44         2'h2: do <= {pdo[2:0],sldi}; // Left Shift
45         2'h3: pdo <= pdi; // PIP0
46         default: do <= do; // Do nothing
47     endcase
48 end
49 end
50 assign sldo = do[0];
51 assign srdo =do[3];
52 endmodule
```

8.3.2 Testbench

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////
3  // Company:
4  // Author: Abhinav M
5  //
6  // Create Date: 22.11.2023 22:06:39
7  // Design Name:
8  // Module Name: tb_E_4_USR
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////////////////
21
22 module tb_E_4_USR;
23     reg clk, rst;
24     reg [1:0] sel;
25     reg [3:0] pdi;
26     reg sldi, srdi;
27     wire [3:0] pdo; //parallel data out
28     wire sldo, srdo;
29
30     E_4_USR uut(clk, rst, sel, pdi, sldi, srdi, pdo, sldo, srdo);
31
32     always #2 clk = ~clk;
33     initial begin
34
35         clk = 0; rst = 1;
36         #3 rst = 0;
37         #5
38
39         pdi = 4'b1001;
40         sel = 2'h3; #4;
41
42
43
44         #100 $finish;
45     end
46
47
48 endmodule
```

8.4 | Elaborated design

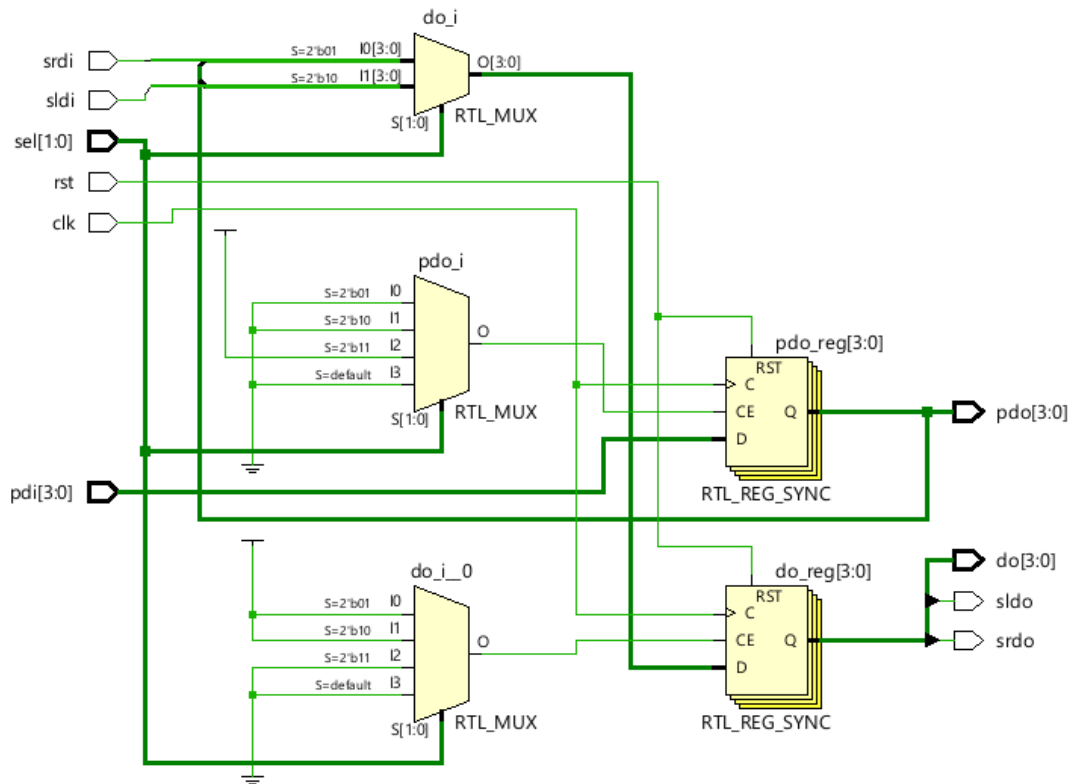


Figure 8.2: Elaborated design

8.5 | Timing diagram

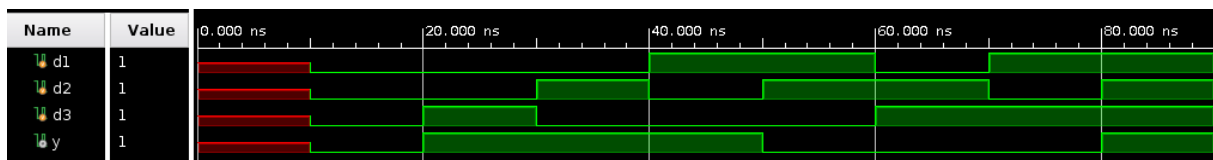


Figure 8.3: Timing diagram.

8.6 | Remarks

Circuit was designed and tested successfully, expected functionality was achieved.

9 | 8-bit sync up-counter

9.1 | Objective

Design a circuit for 8-bit sync up-counter. Verify its functionality using testbench.

9.2 | Code

9.2.1 | Design

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////
3  // Company : CASEST, University of Hyderabad:
4  // Author : Abhinav M
5  //
6  // Create Date: 11/08/2023 03:20:20 PM
7  // Design Name:
8  // Module Name: E_8_upcount_sync
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////////////////
21
22 module E_8_upcount_sync(input clk,output reg [7:0]count);
23 initial
24 count=0;
25 always @(posedge clk )
26 begin
27 if(count<8'hff)
28 count<= count+8'h01;
29 else
30 count=0;
31 end
32 endmodule
33
```

9.2.2 | Testbench

```
1  `timescale 1ns / 1ps
2  //////////////////////////////////////////////////
3  // Company:
4  // Author: Abhinav M
5  //
6  // Create Date: 11/08/2023 03:33:33 PM
7  // Design Name:
8  // Module Name: tb_E_8_downcount_async
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 //////////////////////////////////////////////////
21
22 module tb_E_8_upcount_async();
23 wire [7:0]count ;
24 reg clk;
25 E_8_upcount_sync uut(clk,count);
26 initial
27 begin
28     clk=0;
29     forever #1 clk = ~clk;
30 end
31 #500 $finish;
32 end
33 endmodule
33
```

9.3 | Elaborated design

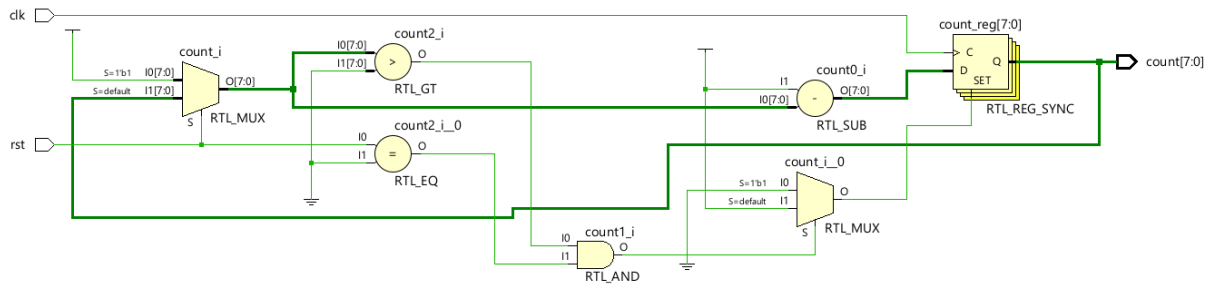


Figure 9.1: Elaborated design

9.4 | Timing diagram

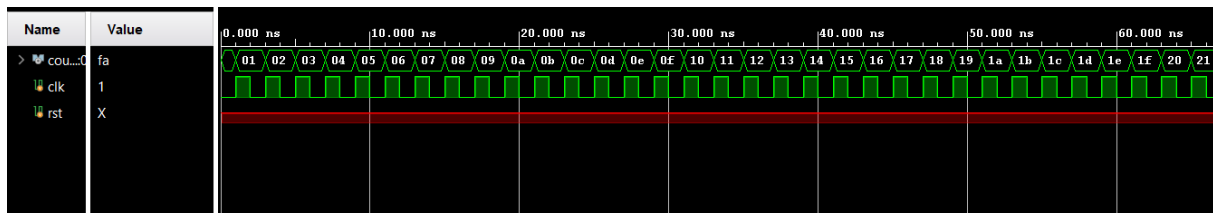


Figure 9.2: Timing diagram

9.5 | Remarks

Circuit was designed and tested successfully, expected functionality was achieved.

10 | 8-bit async down-counter

10.1 | Objective

Design a circuit 8-bit async down-counter. Verify its functionality using testbench.

10.2 | Code

10.2.1 | Design

```
1  `timescale 1ns / 1ps
2  ///////////////////////////////////////////////////////////////////
3  // Company : CASEST, University of Hyderabad:
4  // Author : Abhinav M
5  //
6  // Create Date: 11/08/2023 03:31:05 PM
7  // Design Name:
8  // Module Name: E_8_downcount_async
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22
23 module E_8_downcount_async(input clk,rst, output reg [7:0]count);
24 initial
25 count=8'b11111111;
26
27 always @(posedge clk or posedge rst)
28 begin
29 if(rst)
30 count=8'b11111111;
31
32 if(count>8'b00000000 & rst==0)
33 count<= count-8'b00000001;
34
35 else if(count==8'b00000000);
36 count=8'b11111111;
37 end
38
39
40
41
42
43 endmodule
```

10.2.2 | Testbench

```
1  `timescale 1ns / 1ps
2  ///////////////////////////////////////////////////////////////////
3  // Company:
4  // Author: Abhinav M
5  //
6  // Create Date: 11/08/2023 03:33:33 PM
7  // Design Name:
8  // Module Name: tb_E_8_downcount_async
9  // Project Name:
10 // Target Devices:
11 // Tool Versions:
12 // Description:
13 //
14 // Dependencies:
15 //
16 // Revision:
17 // Revision 0.01 - File Created
18 // Additional Comments:
19 //
20 ///////////////////////////////////////////////////////////////////
21
22 module tb_E_8_downcount_async();
23 wire [7:0]count ;
24 reg clk,rst;
```

```

25 E_8_downcount_async uut(clk,rst,count);
26
27 initial
28 begin
29     clk=0;
30     forever #1 clk = ~clk;
31
32 end
33
34 initial
35 begin
36     rst=0;
37     #20
38     rst=1;
39     #20
40     rst=0;
41     #20
42     rst=1;
43     #20
44     rst=0;
45     #20
46     rst=1;
47     #20
48     rst=0;
49     #20
50     #500 $finish;
51
52 end
53
54 endmodule

```

10.3 | Elaborated design

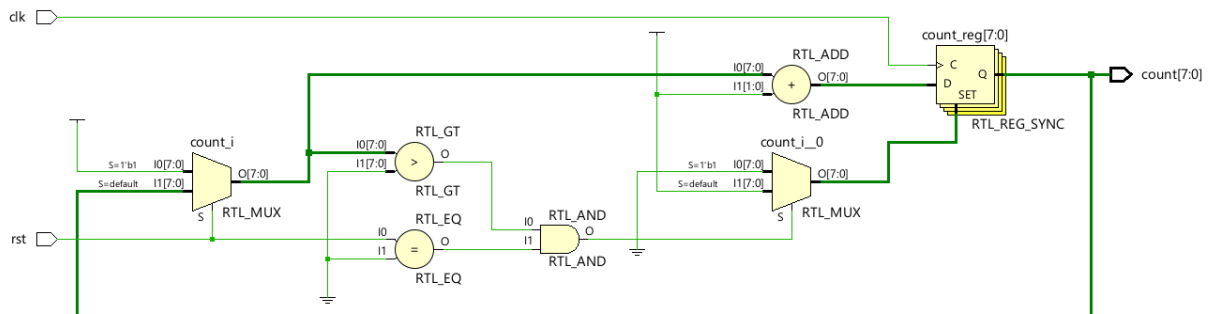


Figure 10.1: Elaborated design

10.4 | Timing diagram

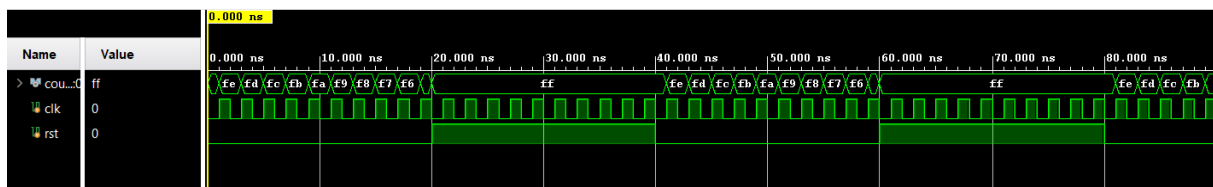


Figure 10.2: Timing diagram

10.5 | Remarks

Circuit was designed and tested successfully, expected functionality was achieved.



11 | References



A | Appendix A: GitHub Repo

The GitHub repo referenced below contains the Verilog code and testbenches mentioned in this report.

<https://github.com/AbhinavM2000/MV401D/>