



دانشگاه صنعتی امیرکبیر
(پلی تکنیک تهران)
دانشکده مهندسی کامپیوتر و فناوری اطلاعات

گزارش تمرین چهارم

درس طراحی سیستم های دیجیتال برنامه پذیر

استاد درس
دکتر صاحب الزمانی

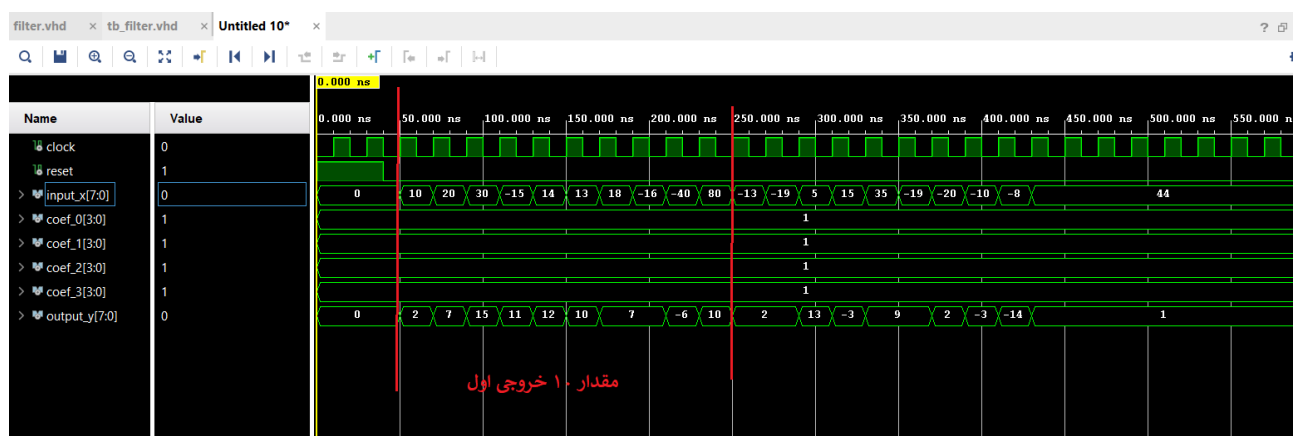
نگارش
آرش حاجی صفی - ۹۶۳۱۰۱۹

خرداد ۱۳۹۹

گزارش:

سوال ۴-۲:

ب) شکل موج ۱۰ خروجی اول به صورت زیر می‌باشد:

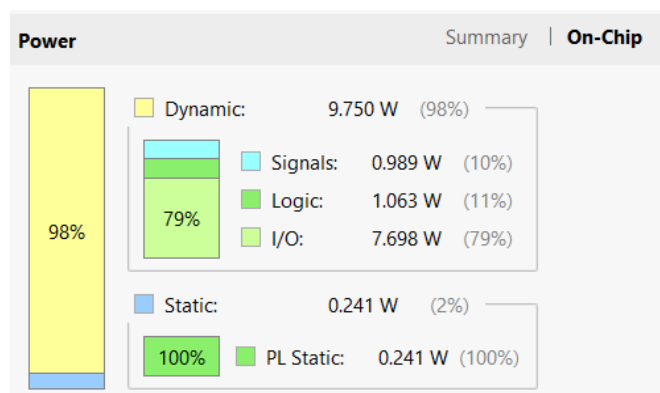


ج)

توان مصرفی:

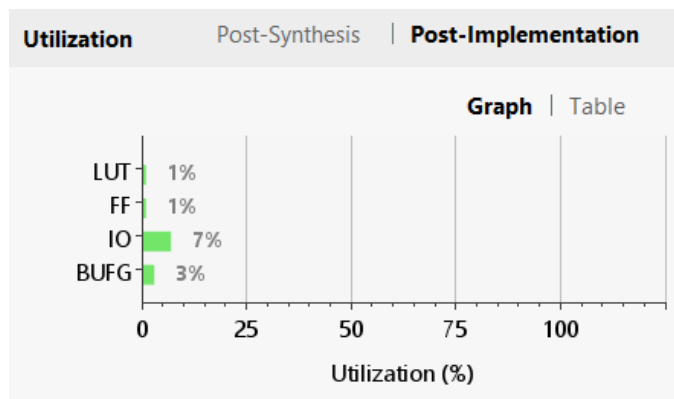
در مجموع مصرف توان چیپ، ۹.۹۹۲ وات، به شرح زیر است:

Power		Summary	On-Chip
Total On-Chip Power:		9.992 W	
Junction Temperature:		42.7 °C	
Thermal Margin:		42.3 °C (23.1 W)	
Effective θ_{JA} :		1.8 °C/W	
Power supplied to off-chip devices:		0 W	
Confidence level:		Low	
Implemented Power Report			



منابع مصرفی: تعداد منابع مصرفی به شرح زیر است:

تعداد منابع مصرفی	نوع منبع
پس از پیاده سازی	
137	Look-up Table
24	Flip-Flop
0	BRAM
0	DSP
34	IO
1	BUFF-Gates



Utilization Post-Synthesis | Post-Implementation

Graph | Table

Resource	Utilization	Available	Utilization...
LUT	137	203800	0.07
FF	24	407600	0.01
IO	34	500	6.80
BUFG	1	32	3.13

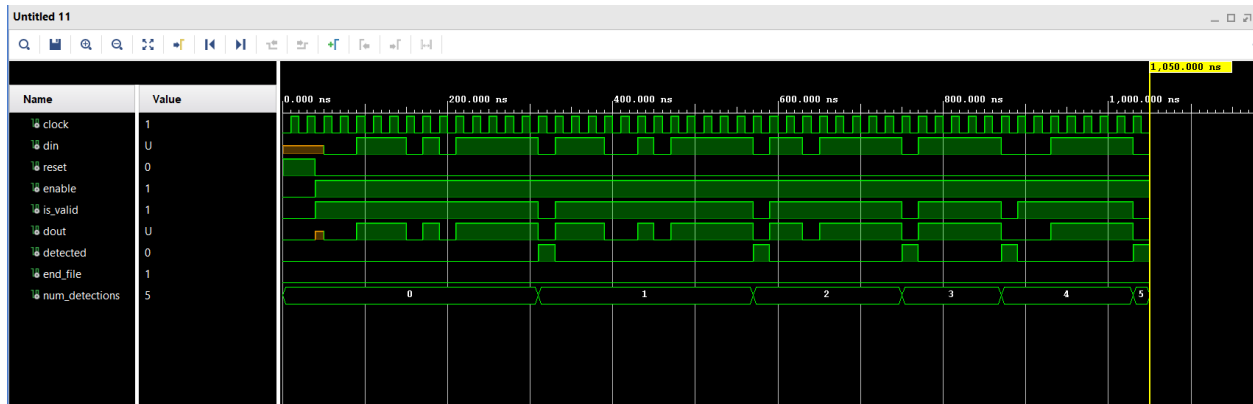
سوال ۴-۳:

(ج)

خروجی فایل تکست:

dataout.txt	
1	0
2	0
3	1
4	1
5	1
6	0
7	1
8	0
9	1
10	1
11	1
12	1
13	1
14	1
15	1
16	1
17	0
18	0
19	1
20	0
21	1
22	1
23	1
24	1
25	1
26	1
27	1
28	0
29	1
30	1
31	1
32	1
33	1
34	1
35	1
36	1
37	1
38	1
39	0
40	0
41	1
42	1
43	1
44	1
45	1
46	

شکل موج خروجی:



نتیجه ی دستور assert در کنسول:

```

SIMULATION - Behavioral Simulation - Functional - sim_1 - tb_zero_deletion_module

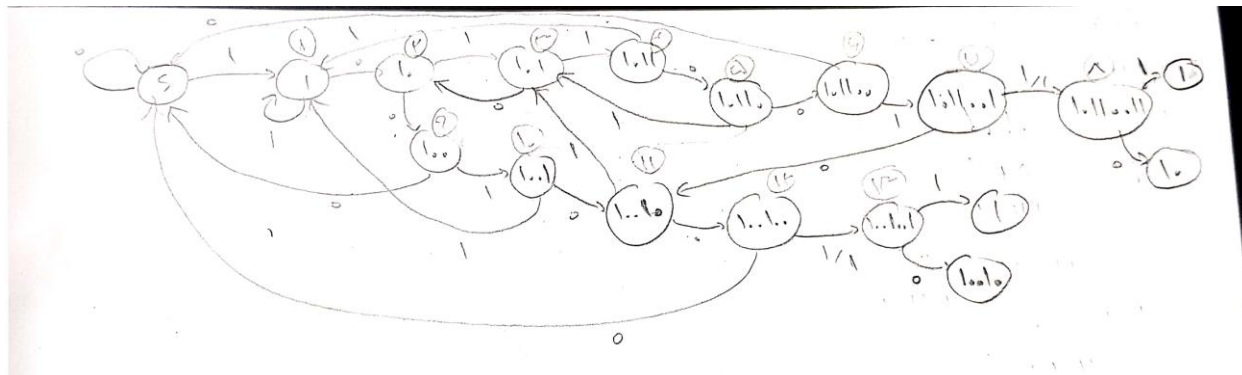
Tcl Console x Messages Log
? - ? ?

# if { [llength [get_objects]] > 0 } {
#   add_wave /
#   set_property needs_save false [current_wave_config]
# } else {
#   send_msg_id Add_Wave-1 WARNING "No top level signals found. Simulator will start without a wave window. If you want to open a wave window go to 'File->New Waveform
# }
# }
# run 1050ns
Note: zero deletion detected at 0 ps
total number of deletions: 0
Time: 0 ps Iteration: 0 Process: /tb_zero_deletion_module/line_138 File: H:/Bachelor/Design Automation/HW4/Programming/Problem43/Problem43.srscs/sim_1/new/tb_zero_delet
Note: zero deletion detected at 0 ps
total number of deletions: 0
Time: 0 ps Iteration: 2 Process: /tb_zero_deletion_module/line_138 File: H:/Bachelor/Design Automation/HW4/Programming/Problem43/Problem43.srscs/sim_1/new/tb_zero_delet
Note: zero deletion detected at 330000 ps
total number of deletions: 1
Time: 330 ns Iteration: 3 Process: /tb_zero_deletion_module/line_138 File: H:/Bachelor/Design Automation/HW4/Programming/Problem43/Problem43.srscs/sim_1/new/tb_zero_del
Note: zero deletion detected at 590000 ps
total number of deletions: 2
Time: 590 ns Iteration: 3 Process: /tb_zero_deletion_module/line_138 File: H:/Bachelor/Design Automation/HW4/Programming/Problem43/Problem43.srscs/sim_1/new/tb_zero_del
Note: zero deletion detected at 770000 ps
total number of deletions: 3
Time: 770 ns Iteration: 3 Process: /tb_zero_deletion_module/line_138 File: H:/Bachelor/Design Automation/HW4/Programming/Problem43/Problem43.srscs/sim_1/new/tb_zero_del
Note: zero deletion detected at 890000 ps
total number of deletions: 4
Time: 890 ns Iteration: 3 Process: /tb_zero_deletion_module/line_138 File: H:/Bachelor/Design Automation/HW4/Programming/Problem43/Problem43.srscs/sim_1/new/tb_zero_del
Note: zero deletion detected at 1050000 ps
total number of deletions: 5
Time: 1050 ns Iteration: 3 Process: /tb_zero_deletion_module/line_138 File: H:/Bachelor/Design Automation/HW4/Programming/Problem43/Problem43.srscs/sim_1/new/tb_zero_de
INFO: [USF-XSim-96] XSim completed. Design snapshot 'tb_zero_deletion_module_behav' loaded.
INFO: [USF-XSim-97] XSim simulation ran for 1050ns
launch_simulation: Time (s): cpu = 00:00:04 ; elapsed = 00:00:06 . Memory (MB): peak = 948.371 ; gain = 0.000

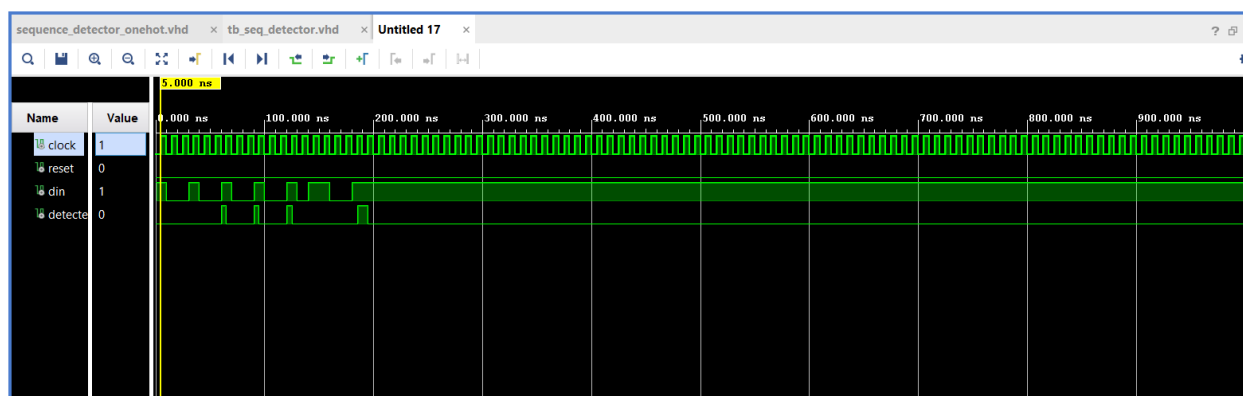
Sim Time: 1050 ns
  
```

سوال ۴-۴:

ماشین حالت طراحی شده که هردو دنباله را به صورت همپوشان تشخیص می‌دهد:



شکل موج خروجی این ماشین برای دنباله ی 10010010010010110011:



الف) با دستور زیر، نوع state encoding را one-hot قرار دادم:

```
attribute fsm_encoding : string;  
attribute fsm_encoding of cur_state : signal is "one-hot";
```

خروجی log سنتز را به صورت فایل one-hot synthesis log.txt در فولدر تمرین 4-4 ضمیمه کردم، قسمت قابل توجه که نوع encoding را نشان می‌دهد به این صورت می‌باشد:

Tcl ConsoleMessagesLog x ReportsDesign Runs

Q

||

📄

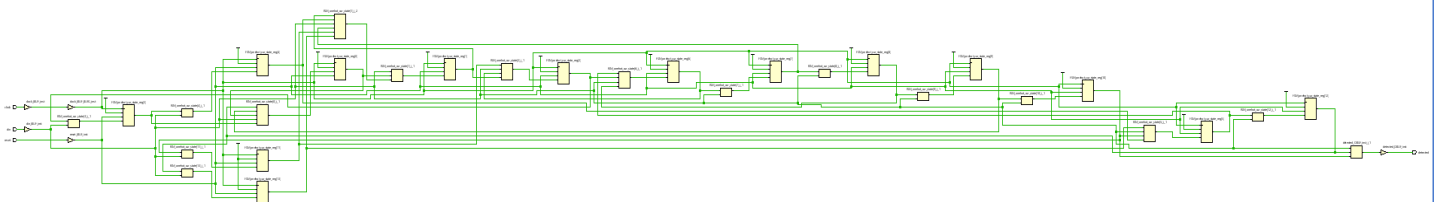
🔍

INFO: [Device 21-403] Loading part xc7k325tffg900-2
INFO: [Synth 8-802] inferred FSM for state register 'cur_state_reg' in module 'sequence_detector'

State	New Encoding	Previous Encoding
start	000000000000001	0000
s1	000000000000010	0001
s2	000000000000100	0010
s9	000000000001000	1001
s10	000000000010000	1010
s11	000000000100000	1011
s3	000000001000000	0011
s4	000000010000000	0100
s5	000001000000000	0101
s6	000010000000000	0110
s7	000100000000000	0111
s8	001000000000000	1000
s12	010000000000000	1100
s13	100000000000000	1101

INFO: [Synth 8-3354] encoded FSM with state register 'cur state req' using encoding 'one-hot' in module 'sequence_detector'

خروجی شماتیک هم به صورت one-hot schematic.sch در فولدر این تمرین ضمیمه کردم، شماتیک این مدار به این صورت می‌باشد:



ب) با دستور زیر، نوع state encoding را johnson قرار دادیم:

```
attribute fsm_encoding : string;  
attribute fsm_encoding of cur_state : signal is "johnson";
```

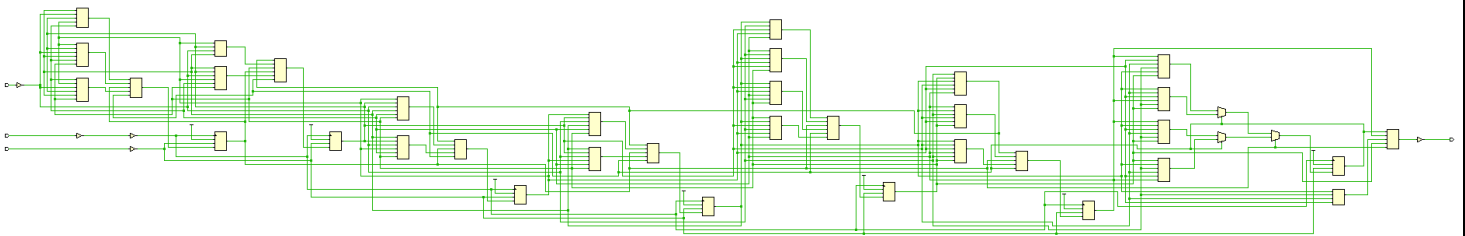
خروجی log سنتز را به صورت فایل johnson synthesis log.txt در فولدر تمرین 4-4 ضمیمه کردم، قسمت قابل توجه که

نوع encoding را نشان می‌دهد به این صورت می‌باشد:

```
INFO: [Device 21-403] Loading part xc7k325tffg900-2  
INFO: [Synth 8-802] inferred FSM for state register 'cur_state_reg' in module 'sequence_detector'  
-----  
State | New Encoding | Previous Encoding  
-----  
start | 0000000 | 0000  
s1 | 1000000 | 0001  
s2 | 1100000 | 0010  
s9 | 1110000 | 1001  
s10 | 1111000 | 1010  
s11 | 1111100 | 1011  
s3 | 1111110 | 0011  
s4 | 1111111 | 0100  
s5 | 0111111 | 0101  
s6 | 0011111 | 0110  
s7 | 0001111 | 0111  
s8 | 0000111 | 1000  
s12 | 0000011 | 1100  
s13 | 0000001 | 1101  
-----  
INFO: [Synth 8-3354] encoded FSM with state register 'cur_state_reg' using encoding 'johnson' in module 'sequence_detector'  
-----  
Finished RTL Optimization Phase 2 : Time (s): cpu = 00:00:07 ; elapsed = 00:00:06 . Memory (MB): peak = 622.824 ; gain = 321.949  
-----
```

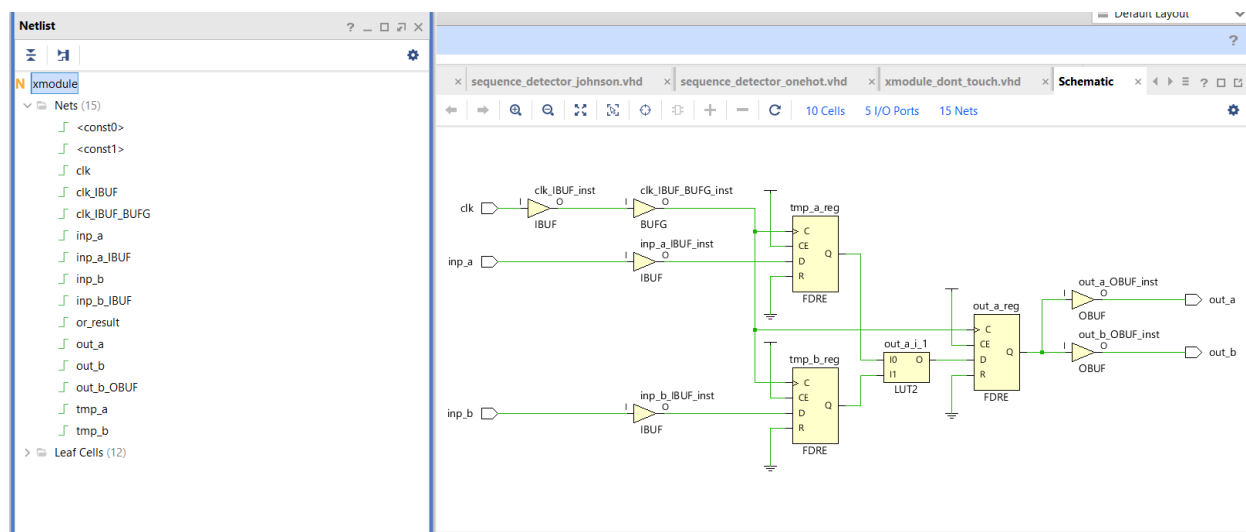
خروجی شماتیک هم به صورت johnson schematic.sch در فولدر این تمرین ضمیمه کردم، شماتیک این مدار به این صورت

می‌باشد:



(ج)

لیست اتصالات پس از سنتز به صورت xmodule_not_changed_netlist.v و xmodule_not_changed_netlist در فولدر این تمرین ذخیره شد. نتیجه ی هردو با هم به صورت زیر است که می‌بینیم wire_a و wire_b حذف شده اند:

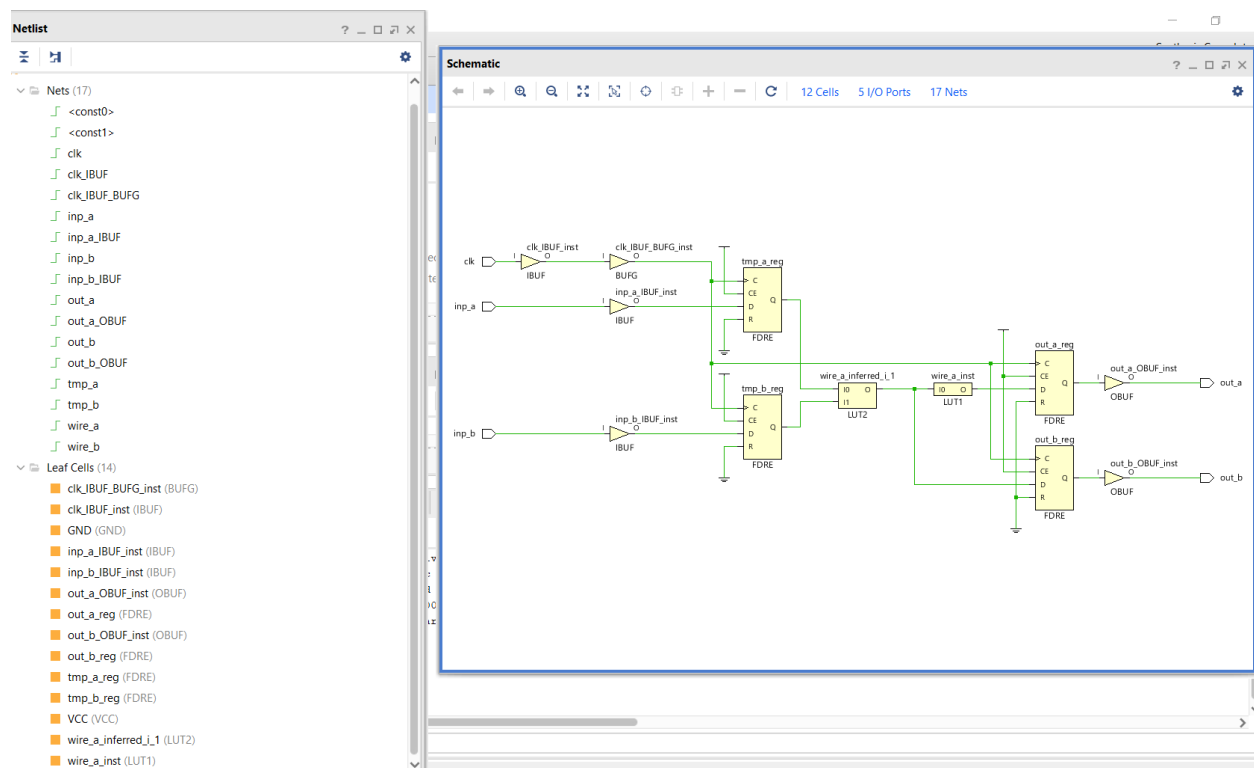


(۵)

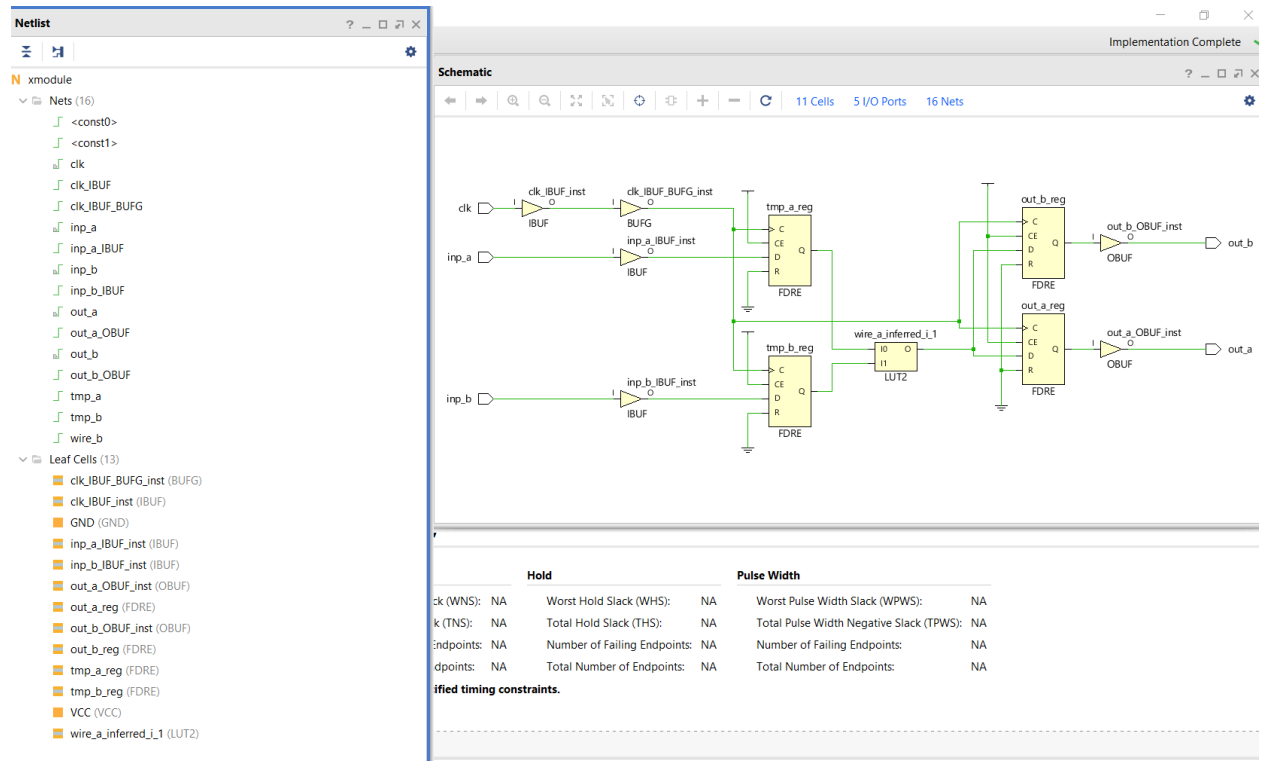
با دستور زیر، keep را برای wire_a و wire_b اعمال می‌کنیم:

```
-- write your code here, add keep attribute for wire_a and wire_b signals
attribute keep : string;
attribute keep of wire_a : signal is "true";
attribute keep of wire_b : signal is "true";
```

لیست اتصالات و شماتیک پس از سنتز به صورت زیر است:



لیست اتصالات و شماتیک پس از پیاده سازی به صورت زیر است:

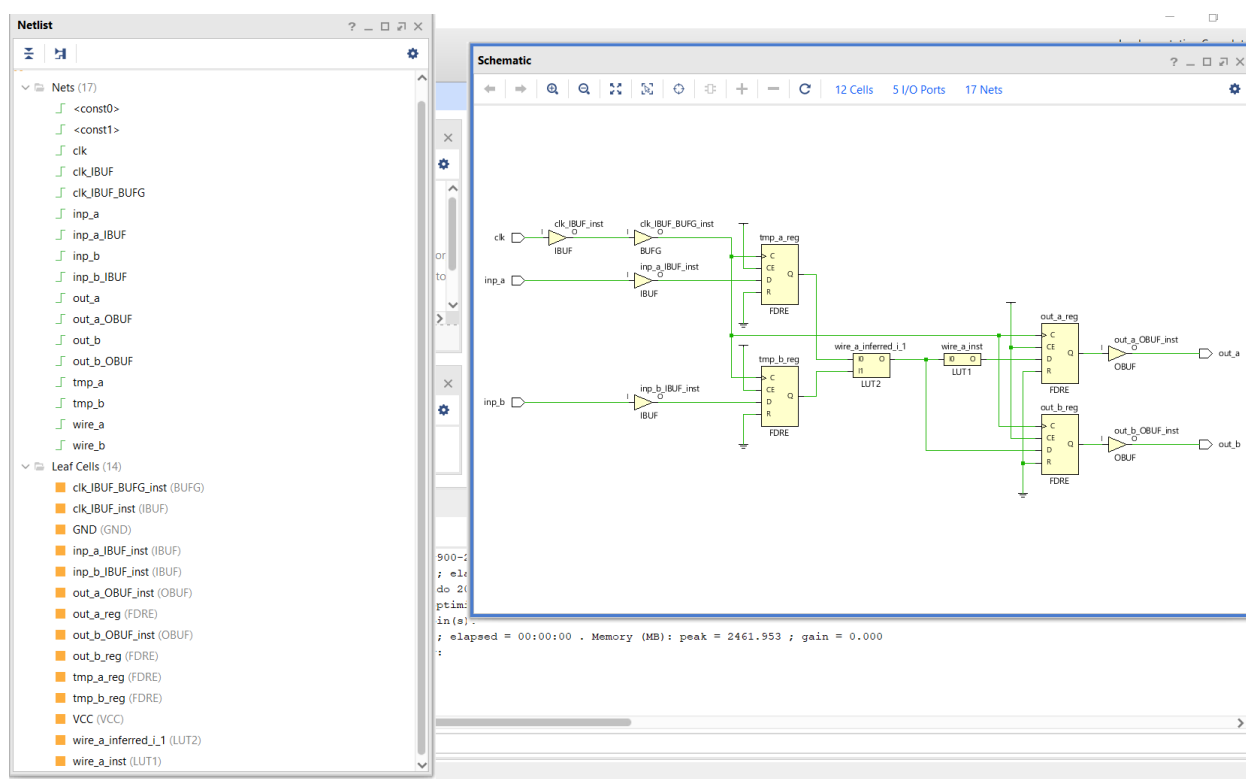


فایل های شماتیک و netlist در فولدر سوال ذخیره شدند.

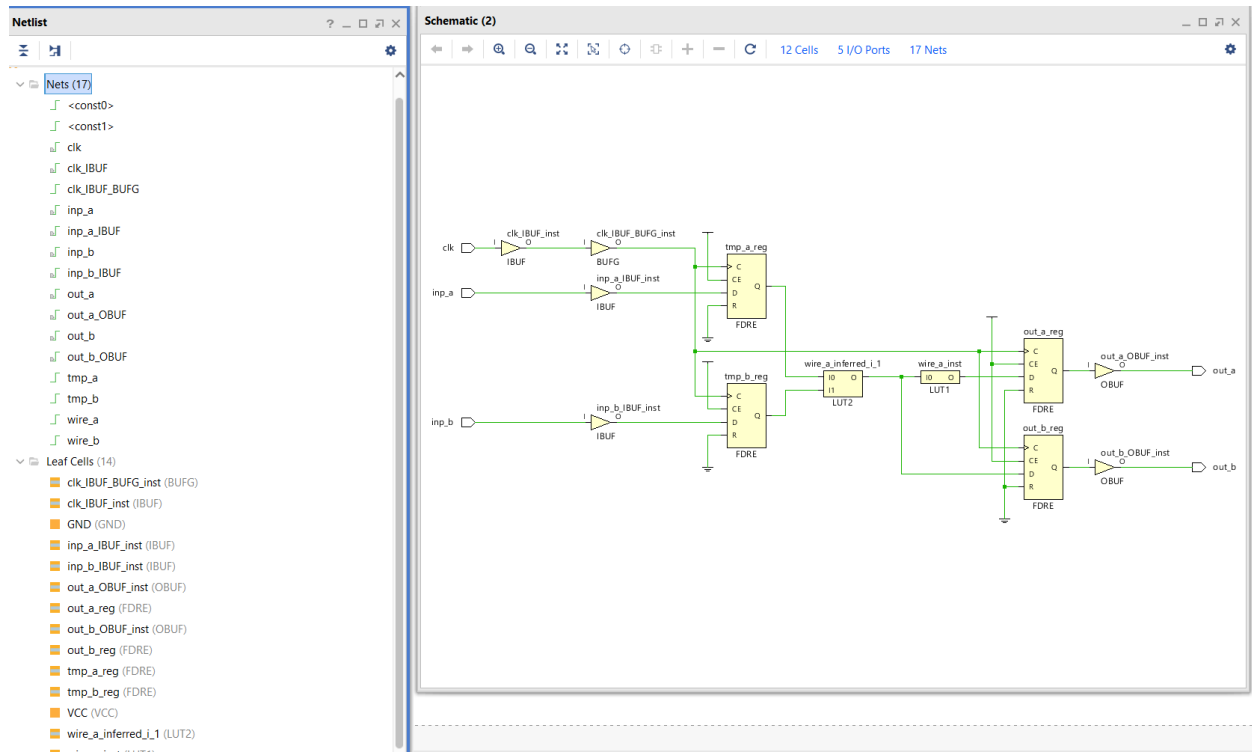
با دستور زیر، dont_touch را برای wire_a و wire_b اعمال می‌کنیم:

```
-- write your code here, add don't touch attribute for wire_a and wire_b signals
attribute dont_touch : string;
attribute dont_touch of wire_a : signal is "true";
attribute dont_touch of wire_b : signal is "true";
```

لیست اتصالات و شماتیک پس از سنتز به صورت زیر است:



لیست اتصالات و شماتیک پس از پیاده سازی به صورت زیر است:



فایل های شماتیک و netlist در فولدر سوال ذخیره شدند.

همانطور که از نتایج مشخص است، KEEP از بهینه سازی روی سیگنال ها در مرحله ی سنتز جلوگیری می کند و نمی گذارد ابزار سنتز سیگنالهای تکراری را به عنوان مثال در این xmodule حذف کند و روی logic block جذب کند ولی در مرحله ی place and route که در implementation اتفاق می افتد، بهینه سازی ممکن است صورت بگیرد (بهینه سازی روی منطق) و سیگنالی حذف یا بهینه شود و keep جلوی آنرا نمی گیرد.

DONT_TOUCH هم به طور مشابه جلوی بهینه سازی روی سیگنالها را در مرحله ی سنتز می گیرد ولی تفاوت آن با KEEP در این است که DONT_TOUCH جلوی بهینه سازی در مرحله ی place and route که در implementation اتفاق می افتد را هم می گیرد و از بهینه سازی روی logic جلوگیری می کند و هیچ سیگنالی حذف یا بهینه نمی شود.

به عنوان مثال در همین xmodule، بدون اعمال هیچکدام از KEEP و DONT_TOUCH می بینیم که بهینه سازی صورت گرفته و سیگنالهای wire_a و wire_b از بین رفته اند چراکه نیازی به آنها نبوده و تکراری بوده اند.

در حالتی که از KEEP استفاده کرده ایم جلوی هرگونه بهینه سازی روی این سیگنالها در فاز سنتز گرفته شده است، در حالت استفاده از DONT_TOUCH هم دقیقاً مشابه همین است در فاز سنتز، ولی پس از implementation می بینیم که در حالتی که از KEEP استفاده کرده ایم یک بهینه سازی اتفاق افتاده و یک LUT که به نام wire_a_ins است حذف شده، البته هنوز هم جلوی بعضی بهینه سازی های دیگر که اگر از KEEP و DONT_TOUCH استفاده نمی کردیم، اعمال می شد، گرفته شده. اما در زمان استفاده از DONT_TOUCH این بهینه سازی اتفاق نیفتاده و این LUT حذف نشده و نتیجه ی بعد از سنتز و بعد از implementation دقیقاً یکی است.