NPTEL Workshop: Homework 5

Wadhwani Electronics Lab, IIT Bombay

July 13, 2022

1 Sequence Generator

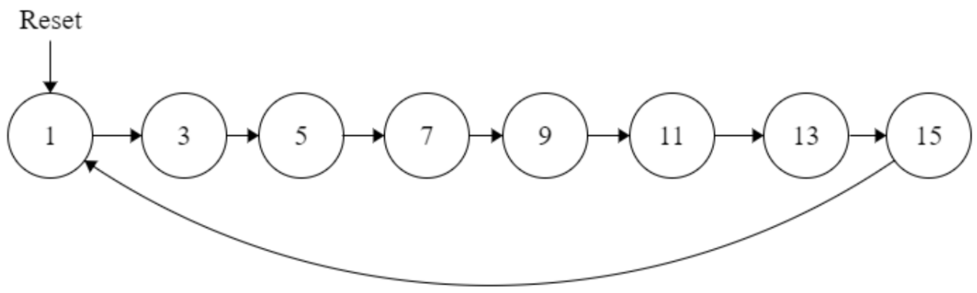
Design and implement the below odd sequence generator in both behavioural and structural-dataflow model and perform RTL, Gate level simulation and scan chain with the given TRACEFILE.

Hint: Unused sequence should be mapped to one of the known sequence i.e reset sequence.

Note: The reset is asynchronous in nature i.e reset affects the output sequence irrespective of the input clock arrival.

Inputs: Reset, Clock
Output(4 bit): y3 y2 y1 y0
Tracefile format < *reset* >< *clock* >   < $y_3y_2y_1y_0$ >   1111

Click on above TRACEFILE text highlighted to get the TRACEFILE



2 Design Procedure:

2.1 State Table:

Present State(Q3 Q2 Q1 Q0)	Next state(nQ3 nQ2 nQ1 nQ0)	D3 D2 D1 D0
0000(0)	0001(1)	0001
0001(1)	0011(3)	0011
0010(2)	0001(1)	0001
0011(3)	0101(5)	0101
0100(4)	0001(1)	0001
0101(5)		
0110(6)		
0111(7)		
1000(8)		
1001(9)		
1010(10)		
1011(11)		
1100(12)		
1101(13)		
1110(14)		
1111(15)		

Complete the above state table.

D =f (Q_3, Q_2, Q_1, Q_0,Reset), Y=f(Q_3, Q_2, Q_1, Q_0,Reset)

From the above state table with the help of K-Maps generate equations for D3, D2, D1, D0 in terms of present state. Here, the outputs are :

$Y_3 = Q_3$
$Y_2 = Q_2$
$Y_1 = Q_1$
$Y_0 = Q_0$

NOTE: When reset is applied the output sequence must be decimal 1 i.e in binary
0001
Particular D flip flop can be set or reset
D flip flop with set=1 implies flip flop output(Q)=1
D flip flop with reset=1 implies flip flop output(Q)=0

3 Code Snippet for D Flipflop

```
library ieee;
use ieee.std_logic_1164.all;
package Flipflops is

component dff_set is port(D,clock,set:in std_logic;Q:out std_logic); end component
dff_set;

component dff_reset is port(D,clock,reset:in std_logic;Q:out std_logic); end component
dff_reset;
end package Flipflops;

--D flip flop with set
library ieee;
use ieee.std_logic_1164.all;
entity dff_set is port(D,clock,set:in std_logic;Q:out std_logic); end entity dff_set;
architecture behav of dff_set is
begin
dff_set_proc: process (clock,set)
begin
if(set='1')then -- set implies flip flip output logic high Q <= ; -- write the flip flop
output when set
elsif (clock'event and (clock='1')) then
Q <= ; -- write flip flop output when not set
end if ;
end process dff_set_proc;
end behav;

--D flip flop with reset
library ieee;
use ieee.std_logic_1164.all;
entity dff_reset is port(D,clock,reset:in std_logic;Q:out std_logic); end entity dff_reset;
architecture behav of dff_reset is
begin
dff_reset_proc: process (clock,reset)
begin
if(reset='1')then -- reset implies flip flip output logic low Q <= ; -- write the flip flop
output when reset
elsif (clock'event and (clock='1')) then
```

```
Q <= ; -- write flip flop output when not reset
end if ;
end process dff_reset_proc;
end behav;
```

4 Code Snippet for Sequence Generator in Structural-Dataflow model

```
library ieee;
use ieee.std_logic_1164.all;
-- write the Flipflops packege declaration

entity Sequence_generator_stru_dataflow is
port (reset,clock: in std_logic;
y:out std_logic_vector(3 downto 0));
end entity Sequence_generator_stru_dataflow;
architecture struct of Sequence_generator_stru_dataflow is signal D
:std_logic_vector(3 downto 0);
signal Q:std_logic_vector(3 downto 0);
begin

-- write the equations in dataflow e.g z=a+bc written as z <= a or (b and c) -- for D(3), D(2), D(1),
D(0) which was derived.
-- Instantiate components dff_reset
-- and dff_set appropriately using port map statements.

end struct;
```

4 Code Snippet for Sequence Generator in Behavioural model

```
library ieee;
use ieee.std_logic_1164.all;

entity Sequence_generator_beh  is
port (reset,clock: in std_logic;
y:out std_logic_vector(3 downto 0));
end entity;

architecture beh of Sequence_generator_beh is
signal state :std_logic_vector(3 downto 0);
constant s_0 :std_logic_vector(3 downto 0) := ”0000”;

-- write rest states in similar way


begin


-- write the code here


end beh;
```