Sequence Generator

EE214 Autumn 2021

September 29, 2021

1 Introduction:

Consider the following Sequence generator which generates a certain sequence upon every input clock pulse and on reset the sequence will go into default sequence i.e 6 in this case.

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

Inputs: Reset, clock
Output(3 bit):Y2Y1Y0

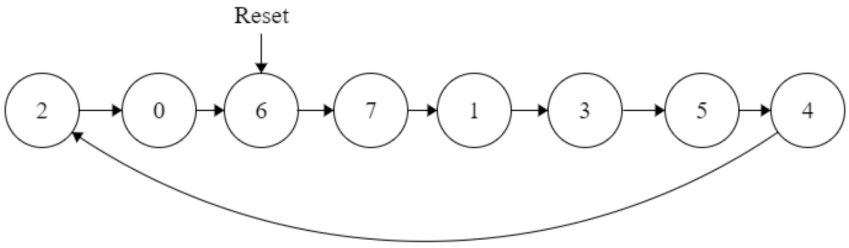

Figure 1: Sequence generator

2 Design Procedure:

2.1 State Table:

Present State(Q2Q1Q0)	Next state(nQ2nQ1nQ0)	D2D1D0
000(0)	110(6)	110
001(1)	011(3)	011
010(2)	000(0)	000
011(3)	101(5)	101
100(4)	010(2)	010
101(5)	100(4)	100
110(6)	111(7)	111
111(7)	001(1)	001

Here next state and outputs are same

2.2 Next state and output Equations:

nQ=f (Q_2, Q_1, Q_0,Reset), Y=f(Q_2, Q_1, Q_0,Reset)
nQ_2**:**

Q_2 \ Q_1Q_0	00	01	11	10
0	1	0	1	0
1	0	1	0	1

D_2=Q_2 xnor (Q_1 xor Q_0)

nQ_1**:**

Q_2 \ Q_1Q_0	00	01	11	10
0	1	1	0	0
1	1	0	0	1

D_1=$Q_2.\overline{Q_0} + \overline{Q_2}.\overline{Q_1}$

nQ_0:

Q_2 \ Q_1Q_0	00	01	11	10
0	0	1	1	0
1	0	0	1	1

D_0=$\overline{Q_2}.Q_0 + Q_2.Q_1$

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

NOTE: When reset is applied the output sequence must be decimal 6 i.e in binary 110
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

2.3 Circuit Diagram:

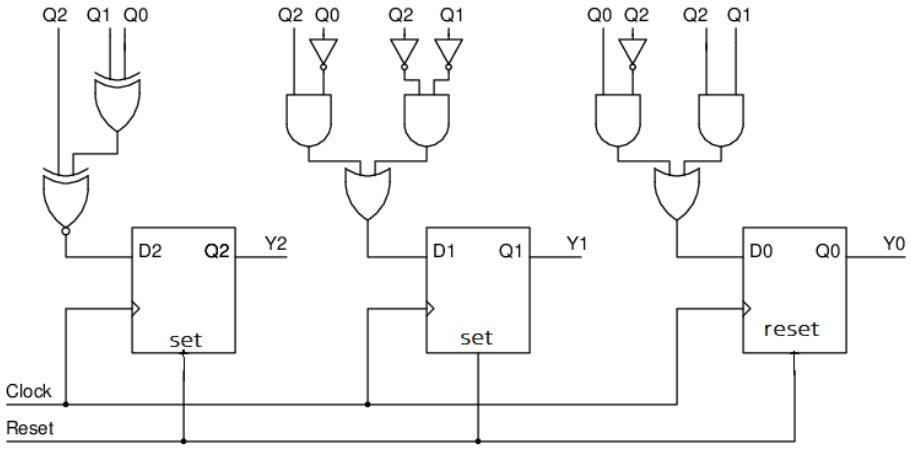

Figure 2: Circuit diagram of the sequence generator

3 VHDL Code:

3.1 Structural Style:

Instantiate D flip flops and interconnect with required combinational logic so that the code should depict above circuit.

D flip flop with set, D flip flop with reset descriptions in behavioral:

```
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;
  --write the component instantiation for the entity 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 <= '1';
elsif (clock'event and (clock='1')) then
Q <= D;
end if ;
end process dff_set_proc;
end behav;


-- D Flipflop having 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 () --write the sensitivity list
begin
if(reset='1')then -- reset implies flip flip output logic low
Q <= ; -- write the D Flipflop output when reset
elsif (clock'event and (clock='1')) then
Q <= ; ---- write the D Flipflop output when posedge clock is triggered
end if ;
end process dff_reset_proc;
end behav;
```

Template code for structural + data flow

```
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(2 downto 0));
end entity Sequence_generator_stru_dataflow;

architecture struct of Sequence_generator_stru_dataflow is
signal D :std_logic_vector(2 downto 0); -- D flip flop inputs
signal Q:std_logic_vector(2 downto 0);  -- D flip flop outputs
begin

-- write the eqations in dataflow  e.g z=a+bc written z<= a or (b and c)
D(2)<=;

D(1)<= ;

D(0)<=;

y(2 downto 0)<= ;

--Q0
dff_0  : dff_reset port map();

--Q1
dff_1  : dff_set port map();

--Q2
dff_2  : dff_set port map();
end struct;
```

3.2 Behavioral Style:

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

entity sequence_generator_behavior is
port (reset,clock: in std_logic;
y:out std_logic_vector(2 downto 0));
end entity sequence_generator_behavior;

architecture behav  of sequence_generator_behavior is
--state binary encoding
signal state:std_logic_vector(2 downto 0);
constant s_2:std_logic_vector(2 downto 0): ="010";
-- write the remaining constants
begin
-- process for next state andflipflop logic
reg_process: process(clock,reset)
begin
if(reset='1')then
state<=; -- write the reset state
elsif(clock'event and clock='1')then
case state is
     when s_6=>
    state<=s_7;
    -- write the remaining  all choices of states

--this is needed  because std_logic can take values other than 0,1 i.e high imepedance
     when others=>
       state<=; -- write the reset state
     end case;
end if;
end process reg_process;
-- output logic concurrent statemet or one more process
y<=state;
end behav;
```

4 LAB TASK:

Complete the VHDL description of sequence generator and Perform RTL and Gate level simulation and scan chain with the given **TRACEFILE**

Tracefile format $< reset\ clock >< y_2y_1y_0 > 111$

5 Home Work(to be submitted before next lab)

Design and implement both structural (7 marks) and behavioral (3 marks) below sequence generator and Perform RTL and 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

Tracefile format $< reset\ clock >< y_3y_2y_1y_0 > 1111$

Click on above TRACEFILE text highlighted to get the TRACEFILE