



Integrated Circuits Design by FPGA

م.م. أحمد مؤيد عبدالحسين
جامعة الفرات الأوسط التقنية / الكلية التقنية الهندسية / نجف

Lecture 3

Lecture 3 :

Part 1 : Using the FPGA board

Objectives of this Lecture

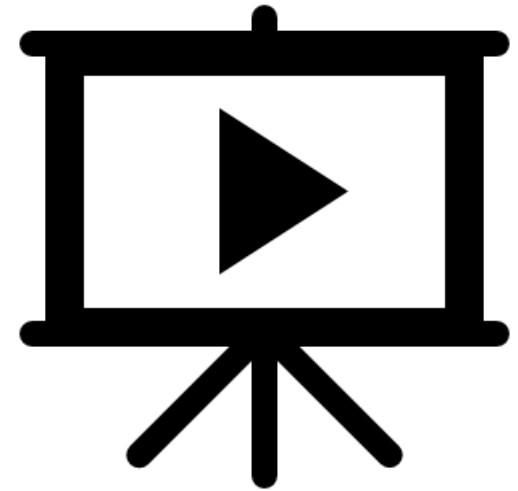
- **To learn how to implement a design on an FPGA board.**

Contents of this Lecture

- **Implementation of a Shift Register design on an FPGA board.**

Implementation of a Shift Register on FPGA board

- Shift Registers are used for data storage or for the movement of data.
- Therefore commonly used inside calculators or computers to store data as binary numbers.
- Convert the data from either a serial to parallel or parallel to serial format.



[What is a shift register ?](#)

Implementation of a Shift Register on FPGA board

```
1  -- Library's
2  library IEEE;
3  use IEEE.STD_LOGIC_1164.ALL;
4  use IEEE.numeric_std.all;
5
6  -- Entity Declaration
7  entity Shift_Reg is
8  port (
9      A      : out std_logic;  -- single bit in register
10     B      : out std_logic;  -- single bit in register
11     C      : out std_logic;  -- single bit in register
12     D      : out std_logic;  -- single bit in register
13     data_in : in  std_logic;  -- data input (1 or 0)
14     reset   : in  std_logic;  -- when this signal goes high clear
15     -- all bit values on a clock cycle
16     clk     : in  std_logic); -- input clock
17 end Shift_Reg;
18
19 -- Architecture Body
20 architecture behavior of Shift_Reg is
21
22     -- Defined Signals used in Architecture
23     signal A_reg, B_reg : std_logic := '0';
24     signal C_reg, D_reg : std_logic := '0';
25
```

```
26  -- Begin Architecture
27  begin
28
29     -- Signal Assignments
30     A <= A_reg;
31     B <= B_reg;
32     C <= C_reg;
33     D <= D_reg;
34
35     -- Process that is used to shift values
36     -- ** HINT **
37     -- (We want this process to be evaluated
38     -- on every clock cycle)
39     reg_process: process(clk)
40     begin
41         if(rising_edge(clk)) then
42             if(reset = '1') then
43                 A_reg <= '0';
44                 B_reg <= '0';
45                 C_reg <= '0';
46                 D_reg <= '0';
47             else
48                 -- ** HINT **
49                 -- This is where the shifting actually occurs
50                 -- depending on how you code this, you can have
51                 -- a shift right or shift left register
52                 A_reg <= data_in;
53                 B_reg <= A_reg;
54                 C_reg <= B_reg;
55                 D_reg <= C_reg;
56             end if;
57         end if;
58     end process reg_process;
59 end behavior;
```


Implementation of a Shift Register on FPGA board

Sources

Design Sources (1)

- Shift_Reg(behavior) (Shift_Reg.vhd)

Constraints (1)

- constrs_1 (1)
 - Arty_A7_Master.xdc

Simulation Sources (1)

Hierarchy Libraries Compile Order

Project Summary x Arty_A7_Master.xdc x

D:/Users/dell/Mivado_projects/Shift_Register/Shift_Register.srcs/constrs_1/imports/Research tools/Arty_A7_Master.xdc

```
16 set_property -dict {PACKAGE_PIN A8 IOSTANDARD LVCOS33} [get_ports data_in]
17 #set_property -dict {PACKAGE_PIN C11 IOSTANDARD LVCOS33} [get_ports {sw[0]}]
18 #set_property -dict {PACKAGE_PIN C10 IOSTANDARD LVCOS33} [get_ports {sw[1]}]
19 #set_property -dict {PACKAGE_PIN A10 IOSTANDARD LVCOS33} [get_ports {sw[2]}]
20
21 ##RGB LEDs
22
23 #set_property -dict { PACKAGE_PIN E1 IOSTANDARD LVCOS33 } [get_ports { led0_b }]; #IO_L18N_T2
24 #set_property -dict { PACKAGE_PIN F6 IOSTANDARD LVCOS33 } [get_ports { led0_g }]; #IO_L19N_T3
25 #set_property -dict { PACKAGE_PIN G6 IOSTANDARD LVCOS33 } [get_ports { led0_r }]; #IO_L19P_T3
26 #set_property -dict { PACKAGE_PIN G4 IOSTANDARD LVCOS33 } [get_ports { led1_b }]; #IO_L20P_T3
27 #set_property -dict { PACKAGE_PIN J4 IOSTANDARD LVCOS33 } [get_ports { led1_g }]; #IO_L21P_T3
28 #set_property -dict { PACKAGE_PIN G3 IOSTANDARD LVCOS33 } [get_ports { led1_r }]; #IO_L20N_T3
29 #set_property -dict { PACKAGE_PIN H4 IOSTANDARD LVCOS33 } [get_ports { led2_b }]; #IO_L21N_T3
30 #set_property -dict { PACKAGE_PIN J2 IOSTANDARD LVCOS33 } [get_ports { led2_g }]; #IO_L22N_T3
31 #set_property -dict { PACKAGE_PIN J3 IOSTANDARD LVCOS33 } [get_ports { led2_r }]; #IO_L22P_T3
32 #set_property -dict { PACKAGE_PIN K2 IOSTANDARD LVCOS33 } [get_ports { led3_b }]; #IO_L23P_T3
33 #set_property -dict { PACKAGE_PIN H6 IOSTANDARD LVCOS33 } [get_ports { led3_g }]; #IO_L24P_T3
34 #set_property -dict { PACKAGE_PIN K1 IOSTANDARD LVCOS33 } [get_ports { led3_r }]; #IO_L23N_T3
35
36 ##LEDs
37
38 set_property -dict { PACKAGE_PIN H5 IOSTANDARD LVCOS33 } [get_ports A]; #IO_L24N_T3_35 Sch=led
39 set_property -dict { PACKAGE_PIN J5 IOSTANDARD LVCOS33 } [get_ports B]; #IO_25_35 Sch=led[5]
40 set_property -dict { PACKAGE_PIN T9 IOSTANDARD LVCOS33 } [get_ports C]; #IO_L24P_T3_A01_D17_1
41 set_property -dict { PACKAGE_PIN T10 IOSTANDARD LVCOS33 } [get_ports D]; #IO_L24N_T3_A00_D16_1
```

Lecture 3

Lecture 3 :

Part 2 : VHDL Test Bench

Objectives of this Lecture

- **To define and introduce a basic Test Bench.**
- **To learn how to construct a Test Bench.**

Contents of this Lecture

- **What is a Test Bench?**
- **Test Bench Outline**
- **Shift Register Example**

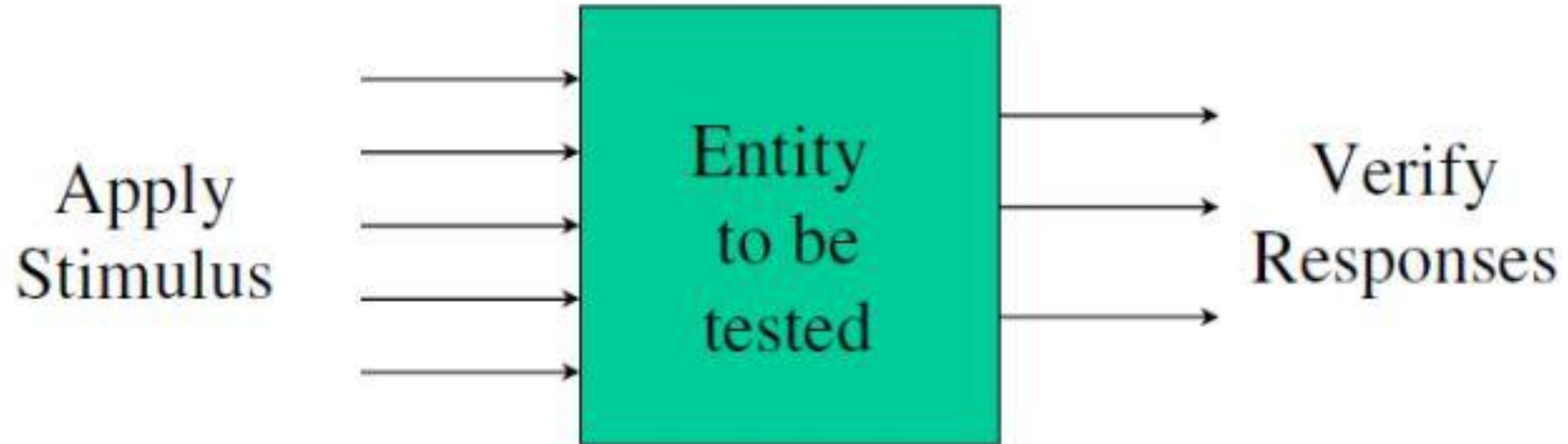
What is a Test Bench?

- A test bench model is used to exercise and verify that your VHDL design is working correctly as expected.
- A test bench **cannot** be synthesized and placed onto an FPGA board.
- You can code the test benches using Notepad++ , or Vivado IDE.
- Vivado IDE is used to execute the simulation.

What is a Test Bench?

- Test Benches consist of:
 - Entity
 - Contains no ports or generics
 - Architecture
 - Declares, instantiates, and wires together the driver model and the model under test
 - The driver model provides stimulus and verifies the response of the model under test

What is a Test Bench?

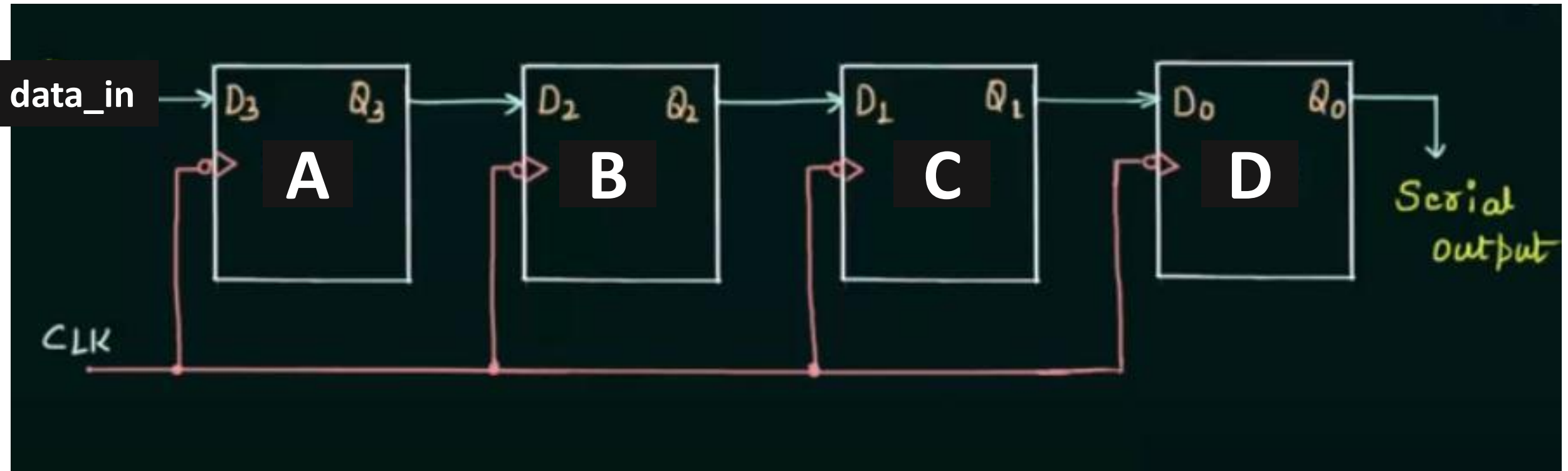


Test Bench Outline

```
1  --Libraries
2  library IEEE;
3  use IEEE.STD_LOGIC_1164.ALL;
4  use IEEE.numeric_std.all;
5
6  --these two libraries are only
7  --supported in Test Bench (these
8  --two libraries will not be
9  -- synthesized)
10 use std.textio.all ;
11 use ieee.std_logic_textio.all ;
12
13 -- Entity
14 entity Test_Bench_name is
15 end;
```

```
17 -- Architecture
18 architecture testing of Test_Bench_name is
19 -- instantiate component of the VHDL entity you
20 -- are testing
21 component LFSR3
22 port (
23     clk : in STD_LOGIC;
24     outp: out STD_LOGIC_VECTOR ( 2 downto 0 ));
25 end component;
26
27 -- Simulation signals
28 signal clk_sim           : std_logic := '0';
29 signal outp_sim          : std_logic_vector(2 downto 0);
30
31 begin
32     -- Sometimes this is called UUT (Unit Under Test)
33     dev_to_test: LFSR3
34         port map(clk_sim, outp_sim);
35
36     -- Simulate the input clock to your design
37     clk_proc : process
38     begin
39         wait for 50 ns;
40         clk_sim <= not clk_sim;
41     end process clk_proc;
42
43 end testing;
```

Shift Register Example

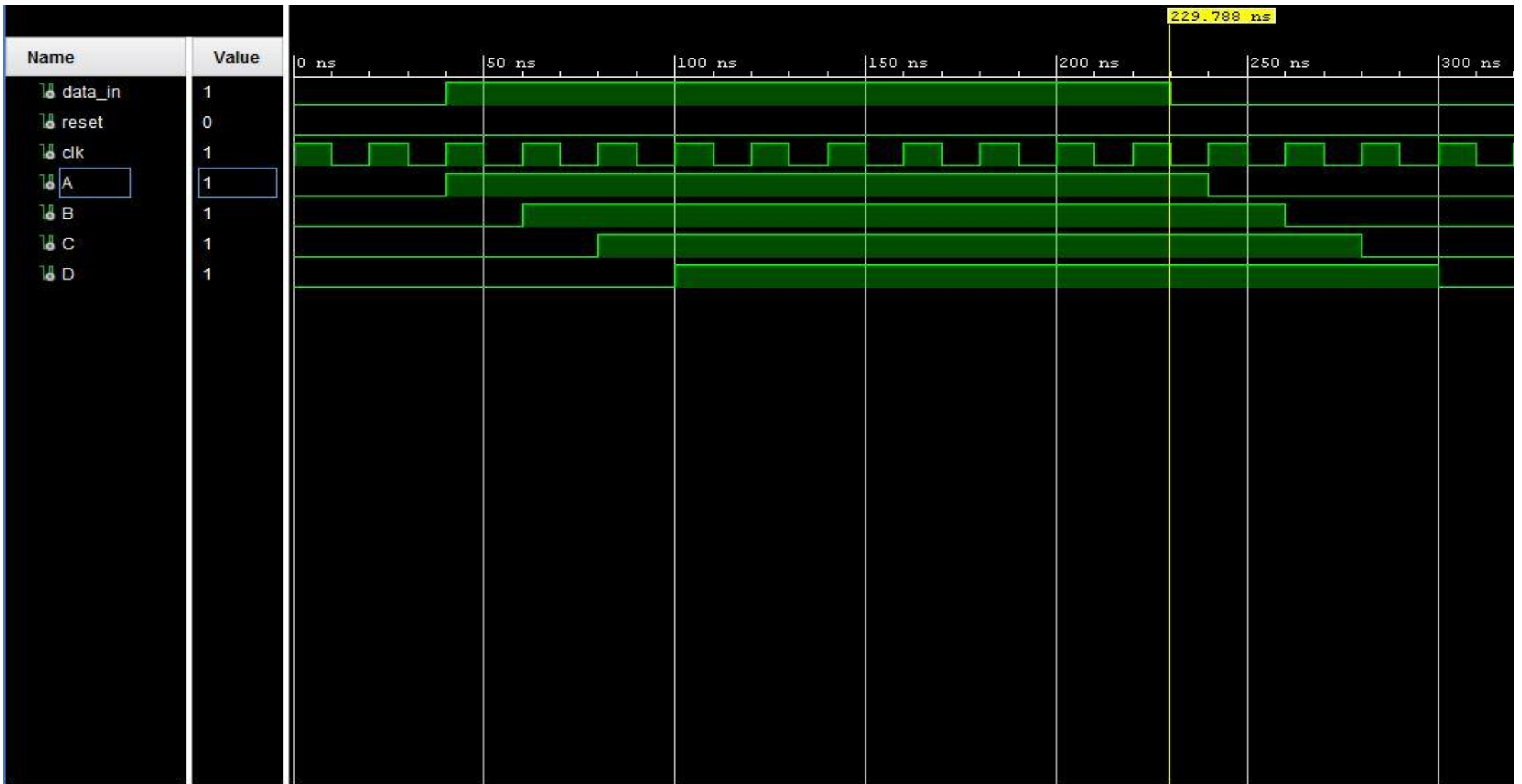


Shift Register Example

```
1  library IEEE;
2  use IEEE.STD_LOGIC_1164.ALL;
3  use IEEE.numeric_std.all;
4
5  entity test_Shift_Reg is
6  end;
7
8  architecture test of test_Shift_Reg is
9
10 component Shift_Reg
11 port (A:          out std_logic;
12       B:          out std_logic;
13       C:          out std_logic;
14       D:          out std_logic;
15       data_in:    in  std_logic;
16       reset:      in  std_logic;
17       clk:        in  std_logic);
18 end component;
19
20 signal data_in : std_logic := '0';
21 signal reset  : std_logic := '0';
22 signal clk    : std_logic := '1';
23 signal A, B, C, D: std_logic;
24
```

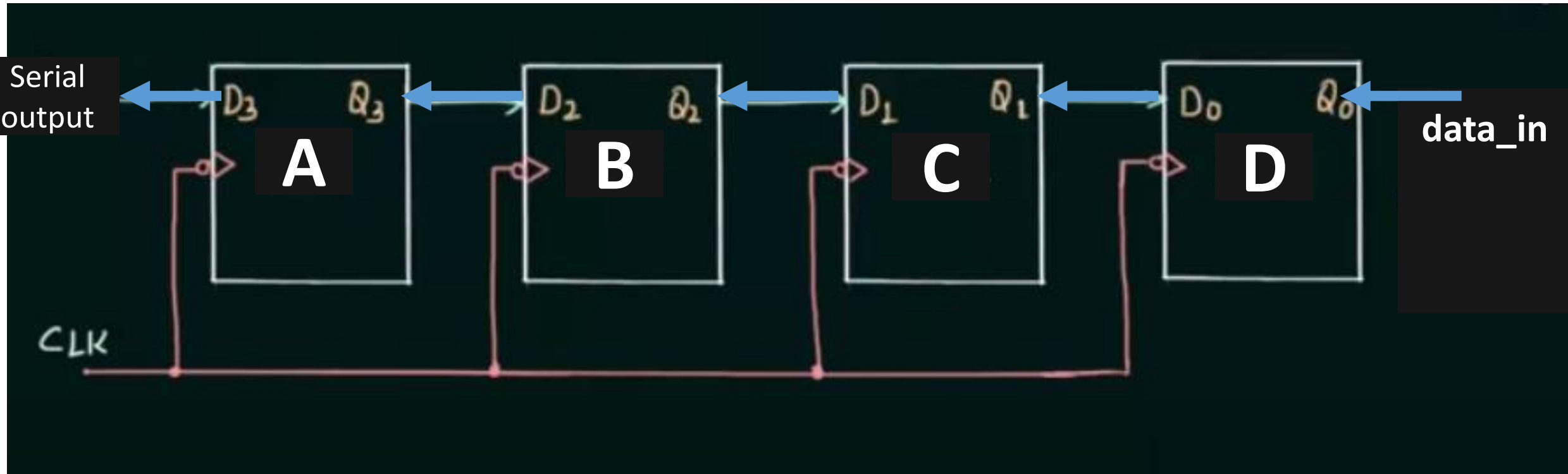
```
25 begin
26
27     dev_to_test: shift_reg
28         port map(A, B, C, D, data_in, reset, clk);
29
30
31     clk_stimulus: process
32     begin
33         wait for 10 ns;
34         clk <= not clk;
35     end process clk_stimulus;
36
37     data_stimulus: process
38     begin
39         wait for 40 ns;
40         data_in <= not data_in;
41         wait for 150 ns;
42     end process data_stimulus;
43
44 end test;
```

Shift Register Example



Assignment

- For the same VHDL code of shift register in this lecture. How you can make the shifting process in the **left** direction?



End of lecture 3

Any Questions ?