1)ALU
Code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use IEEE.NUMERIC_STD.ALL;
entity ALU_4bit is
Port ( A : in STD_LOGIC_VECTOR (3 downto 0);
B : in STD_LOGIC_VECTOR (3 downto 0);
F : in STD_LOGIC_VECTOR (2 downto 0);
Y : out STD_LOGIC_VECTOR (3 downto 0);
C_B : out STD_LOGIC
);
end ALU_4bit;
architecture ALU_4bit_arch of ALU_4bit is
signal result:STD_LOGIC_VECTOR(4 downto 0):="00000";
begin
process(A,B,F)
begin
CASE F IS
when "000" =>
result <= '0' & (A AND B);
when "001" =>
result <= '0' & (A NAND B);
when "010" =>
result <= '0' & (A OR B);
when "011" =>
result <= '0' & (A XOR B);
when "100" =>
result <= '0' & (A XNOR B);
when "101" =>
```

```
result <= '0' & (A NOR B);

when "110" =>

result <= ('0' & A)+('0' & B);
when others =>

if A < B then

result <= '0' & (NOT B);

result <= result+1;

result <= ('0' & A) + result;

result <= (NOT result) +1;

result <= (NOT(('0' & A) + ('0' &(NOT B)) + 1))+1;

else

result <=('0' & A)-('0' & B);

end if ;

end CASE;

end process;

Y <= result(3 downto 0);

C_B <= result(4);

end ALU_4bit_arch;
```

Test Bench:

```
LIBRARY ieee;

USE ieee.std_logic_1164.ALL;

USE ieee.std_logic_unsigned.ALL;

ENTITY ALU_4bit_tb IS

END ALU_4bit_tb;

ARCHITECTURE behavior OF ALU_4bit_tb IS

-- Component Declaration for the Unit Under Test (UUT)

COMPONENT ALU_4bit

PORT(

A : IN std_logic_vector(3 downto 0);

B : IN std_logic_vector(3 downto 0);

F : IN std_logic_vector(2 downto 0);
```

```
Y : OUT std_logic_vector(3 downto 0);
C_B : OUT std_logic


);
END COMPONENT;
--Inputs
signal A : std_logic_vector(3 downto 0) := "0010";
signal B : std_logic_vector(3 downto 0) := "1111";
signal F : std_logic_vector(2 downto 0) := (others => '1');
--Outputs
signal Y : std_logic_vector(3 downto 0);
signal C_B : std_logic;
-- No clocks detected in port list. Replace <clock> below with
-- appropriate port name

BEGIN
-- Instantiate the Unit Under Test (UUT)
uut: ALU_4bit PORT MAP (
A => A,
B => B,
F => F,
Y => Y,
C_B => C_B
);
-- Stimulus process
stim_proc_F: process
begin
F <= F + 1;
wait for 25 ns;
end process;
END;
```

2nd)
Code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use IEEE.NUMERIC_STD.ALL;

entity UNI_Shift_Register is
Port ( rst : in STD_LOGIC;

clk : in STD_LOGIC;

Sin : in STD_LOGIC;
mode : in STD_LOGIC_VECTOR (1 downto 0);
Pin : in STD_LOGIC_VECTOR (3 downto 0);
Sout : out STD_LOGIC;
Pout : out STD_LOGIC_VECTOR (3 downto 0)

);
end UNI_Shift_Register;
architecture UNI_Shift_Register_arch of UNI_Shift_Register is
SIGNAL temp : STD_LOGIC_VECTOR (3 downto 0):="0000";
begin
PROCESS(rst, clk, mode, Sin, Pin)
BEGIN
IF rst = '1' THEN
Pout <= "0000";
Sout <= '0';
ELSIF FALLING_EDGE(clk) THEN
```

```
CASE mode IS
WHEN "00" =>
temp(3 downto 1) <= temp(2 downto 0);
temp(0) <= Sin;
Sout <= temp(3);
Pout <= "0000";
WHEN "01" =>
temp(3 downto 1) <= temp(2 downto 0);
temp(0) <= Sin;
Pout <= temp;
Sout <= '0';
WHEN "10" =>
temp <= Pin;
Sout <= temp(3);
temp(3 downto 1) <= temp(2 downto 0);
Pout <= "0000";
WHEN OTHERS =>
Pout <= Pin;
Sout <= '0';

END CASE;
END IF;
END PROCESS;
end UNI_Shift_Register_arch;
```

test bench:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY UNI_Shift_Register_tb IS
END UNI_Shift_Register_tb;
ARCHITECTURE behavior OF UNI_Shift_Register_tb IS
```

```
-- Component Declaration for the Unit Under Test (UUT)
COMPONENT UNI_Shift_Register
PORT(
rst : IN std_logic;
clk : IN std_logic;
mode : IN std_logic_vector(1 downto 0);
Sin : IN std_logic;
Pin : IN std_logic_vector(3 downto 0);
Sout : OUT std_logic;
Pout : OUT std_logic_vector(3 downto 0)
);
END COMPONENT;
--Inputs
signal rst : std_logic := '0';
signal clk : std_logic := '1';
signal mode : std_logic_vector(1 downto 0) := (others => '0');
signal Sin : std_logic := '0';
signal Pin : std_logic_vector(3 downto 0) := "1010";
--Outputs
signal Sout : std_logic;
signal Pout : std_logic_vector(3 downto 0);
-- Clock period definitions
constant clk_period : time := 10 ns;
BEGIN
-- Instantiate the Unit Under Test (UUT)
uut: UNI_Shift_Register PORT MAP (
rst => rst,
clk => clk,
mode => mode,
Sin => Sin,
Pin => Pin,
```

```
Sout => Sout,
Pout => Pout
);
-- Clock process definitions
clk_process :process
begin

clk<=NOT(clk);
wait for clk_period/2 ;

end process;

-- Stimulus process
stim_proc_mode: process
begin
mode<="00";
wait for 80 ns;
mode<="01";
wait for 50 ns;
mode<="10";
wait for 50 ns;
mode<="11";
wait for 20 ns;
end process;

stim_proc_Sin:process
begin
wait for 10 ns;
Sin<='1';
wait for 10 ns;
Sin<='0';
```

```
wait for 10 ns;
Sin<='1';
wait for 10 ns;
Sin<='0';
wait for 10 ns;
Sin<= '0';
wait for 40 ns;
Sin<='1';
wait for 10 ns;
Sin<='0';
wait for 10 ns;
Sin<='1';
wait for 10 ns;
Sin<='0';
wait for 10 ns;
Sin<= '0';
wait ;
end process;

stim_proc_rst:process
begin
rst<='0';
wait for 300 ns;
rst<='1';
wait for 10 ns;
end process;
END;
```

3) Code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
```

```
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity mod25 is
Port ( rst : in STD_LOGIC;

pr : in STD_LOGIC;

clk : in STD_LOGIC;
dir : in STD_LOGIC;
Q : out STD_LOGIC_VECTOR (4 downto 0));
end mod25;
architecture mod25_arch of mod25 is
signal Qtemp : STD_LOGIC_VECTOR (4 downto 0) := "00000";
begin
process(rst,pr,clk,dir)
begin
if rst ='1' then
Qtemp <= (OTHERS =>'0');
elsif pr='1' then
Qtemp <= (OTHERS =>'1');
elsif falling_edge(clk) then
if dir = '1' then

if Qtemp < 24 then
Qtemp <= Qtemp + 1;
else
Qtemp <= "00000";
end if;

else

if Qtemp > 7 then
```

```
Qtemp <= Qtemp - 1;
else
Qtemp <= "11111";
end if;

end if;
end if;
end process;
Q<=Qtemp;

end mod25_arch;
```

Test bench:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY mod25_tb IS
END mod25_tb;
ARCHITECTURE behavior OF mod25_tb IS
-- Component Declaration for the Unit Under Test (UUT)
COMPONENT mod25
PORT(
rst : IN std_logic;

pr : IN std_logic;

clk : IN std_logic;
dir : IN std_logic;
Q : OUT std_logic_vector(4 downto 0)
);
END COMPONENT;
```

```
--Inputs
signal rst : std_logic := '0';
signal pr : std_logic := '0';
signal clk : std_logic := '0';
signal dir : std_logic := '0';
--Outputs
signal Q : std_logic_vector(4 downto 0);
-- Clock period definitions
constant clk_period : time := 10 ns;
BEGIN
-- Instantiate the Unit Under Test (UUT)
uut: mod25 PORT MAP (
rst => rst,

pr => pr,

clk => clk,
dir => dir,
Q => Q
);
-- Clock process definitions
clk_process :process
begin

clk <= '0';
wait for clk_period/2;
clk <= '1';
wait for clk_period/2;

end process;
-- Stimulus process
```

```
stim_proc_dir: process
begin

dir <= not(dir);
wait for 320 ns;
end process;
stim_proc_rst: process
begin
wait for 680 ns;
rst <= '1';
wait for 40 ns;
rst <= '0';
wait;
end process;
stim_proc_pr: process
begin
wait for 750 ns;
pr <= '1';
wait for 40 ns;
pr <= '0';
wait;
end process;
END;
```

4)
Code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity FIFO_4x8 is
Port ( rst : in STD_LOGIC;
clk : in STD_LOGIC;
addr : in STD_LOGIC_VECTOR (1 downto 0) := "00";
```

```
d_in : in STD_LOGIC_VECTOR (7 downto 0);
rd_wr : in STD_LOGIC;
d_out : out STD_LOGIC_VECTOR (7 downto 0) := "00000000";
empty : out STD_LOGIC := '1';
full : out STD_LOGIC := '0');
end FIFO_4x8;
architecture FIFO_4x8_arch of FIFO_4x8 is
TYPE mem IS ARRAY(3 DOWNTO 0) OF STD_LOGIC_VECTOR (7 DOWNTO 0);
SIGNAL memory : mem := (others=>(others=>'0'));
begin
PROCESS(rst, clk, addr, d_in, rd_wr)
begin
if rst = '1' then
d_out <= "00000000";
empty <= '1';
full <= '0';
memory <= (others=>(others=>'0'));

elsif falling_edge(clk) then
case rd_wr is

when '0' =>
d_out <= memory(conv_integer(addr));
empty <= '0';

full <= '1';
when others =>
memory(conv_integer(addr)) <= d_in;
empty <= '0';
if addr = "11" then
full <= '1';
```

```
else
full <= '0';
end if;

end case;
end if;
end process;
end FIFO_4x8_arch;
```

Test Bench:

```
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
use ieee.numeric_std.ALL;
ENTITY FIFO_4x8_tb IS
END FIFO_4x8_tb;
ARCHITECTURE behavior OF FIFO_4x8_tb IS
-- Component Declaration for the Unit Under Test (UUT)
COMPONENT FIFO_4x8
PORT(
rst : IN std_logic;
clk : IN std_logic;
addr : IN std_logic_vector(1 downto 0);
d_in : IN std_logic_vector(7 downto 0);
rd_wr : IN std_logic;
d_out : OUT std_logic_vector(7 downto 0);
empty : OUT std_logic;
full : OUT std_logic
);
END COMPONENT;

--Inputs
signal rst : std_logic := '0';
```

```
signal clk : std_logic := '1';
signal addr : std_logic_vector(1 downto 0) := (others => '0');
signal d_in : std_logic_vector(7 downto 0) := (others => '0');
signal rd_wr : std_logic := '0';
--Outputs
signal d_out : std_logic_vector(7 downto 0);
signal empty : std_logic;
signal full : std_logic;
-- Clock period definitions
constant clk_period : time := 10 ns;
BEGIN
-- Instantiate the Unit Under Test (UUT)
uut: FIFO_4x8 PORT MAP (
rst => rst,
clk => clk,
addr => addr,
d_in => d_in,
rd_wr => rd_wr,
d_out => d_out,
empty => empty,
full => full
);
-- Clock process definitions
clk_process :process
begin

clk <= not(clk);
wait for clk_period/2;

end process;
```

```
-- Stimulus process
stim_proc: process
begin

rd_wr <= '1';
for address in 0 to 3 loop
addr <= std_logic_vector(to_unsigned(address, 2));
d_in <= std_logic_vector(to_unsigned(63*(address + 1), 8));
wait for clk_period*2;
end loop;
d_in <= std_logic_vector(to_unsigned(0, 8));
rd_wr <= '0';
for address in 0 to 3 loop
addr <= std_logic_vector(to_unsigned(address, 2));
wait for clk_period*2;
end loop;

end process;
stim_proc_reset :PROCESS
BEGIN
wait for clk_period*12;

rst <= '1';

wait for clk_period*2;
rst <= '0';
wait ;
end process;

END;
```

5th:
Code:

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
entity LCD_FSM is
Port ( rst : in std_logic; -- reset
clk_12Mhz : in std_logic; -- high freq. clock
lcd_rs : out std_logic; -- LCD RS control
lcd_en : out std_logic; -- LCD Enable
lcd_data : out std_logic_vector(7 downto 0)); -- LCD Data port
end LCD_FSM;
architecture Behavioral of LCD_FSM is
signal div : std_logic_vector(15 downto 0); --- delay timer 1
signal clk_fsm,lcd_rs_s: std_logic;
-- LCD controller FSM states
type state is (reset,func,mode,cur,clear,d0,d1,d2,d3,d4,hold);
signal ps1,nx : state;
signal dataout_s : std_logic_vector(7 downto 0); --- internal data command multiplexer
begin
----- clk divider ---------------------------------
process(rst,clk_12Mhz)
begin
if(rst = '1')then
div <= (others=>'0');
elsif( clk_12Mhz'event and clk_12Mhz ='1')then
div <= div + 1;
end if;
end process;
-------------------------------------------------
clk_fsm <= div(15);
```

```
----- Presetn state Register ----------------------
process(rst,clk_fsm)
begin
if(rst = '1')then
ps1 <= reset;
elsif (rising_edge(clk_fsm)) then
ps1 <= nx;
end if;
end process;
----- state and output decoding process
process(ps1)
begin
case(ps1) is
when reset =>

nx <= func;
lcd_rs_s <= '0';

dataout_s <= "00111000"; -- 38h

when func =>

nx <= mode;
lcd_rs_s <= '0';
dataout_s <= "00111000"; -- 38h

when mode =>

nx <= cur;
lcd_rs_s <= '0';
dataout_s <= "00000110"; -- 06h
```

```
when cur =>

nx <= clear;
lcd_rs_s <= '0';
dataout_s <= "00001100"; -- 0Ch curser at starting point of

line1
when clear=>

nx <= d0;
lcd_rs_s <= '0';
dataout_s <= "00000001"; -- 01h

when d0 =>

lcd_rs_s <= '1';
dataout_s <= "01010000"; -- P ( Decimal = 80 , HEX = 50 )
nx <= d1;

when d1 =>

lcd_rs_s <= '1';
dataout_s <= "01001001"; -- I ( Decimal = 73 , HEX = 49 )
nx <= d2;

when d2 =>

lcd_rs_s <= '1';
dataout_s <= "01000011"; -- C ( Decimal = 67 , HEX = 43 )
nx <= d3;
```

```
when d3 =>

lcd_rs_s <= '1';
dataout_s <= "01010100"; -- T ( Decimal = 84 , HEX = 54 )
nx <= d4;
when d4 =>

lcd_rs_s <= '1';
dataout_s <= "00100000"; -- space ( Decimal = 32 , HEX = 20 )
nx <= hold;

when hold =>

lcd_rs_s <= '0';
dataout_s <= "00000000"; -- hold ( Decimal = 32 , HEX = 00 ) ,

NULL

nx <= hold;

when others=>

nx <= reset;
lcd_rs_s <= '0';
dataout_s <= "00000001"; -- CLEAR ( Decimal = 1 , HEX = 01 )

end case;
end process;

lcd_en <= clk_fsm;
```

```
lcd_rs <= lcd_rs_s;
lcd_data <= dataout_s;
end Behavioral;
test bench:
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
ENTITY LCD_Test IS
END LCD_Test;
ARCHITECTURE behavior OF LCD_Test IS
-- Component Declaration for the Unit Under Test (UUT)
COMPONENT LCD_FSM
PORT(
rst : IN std_logic;
clk_12Mhz : IN std_logic;
lcd_rs : OUT std_logic;
lcd_en : OUT std_logic;
lcd_data : OUT std_logic_vector(7 downto 0)
);
END COMPONENT;

--Inputs
signal rst : std_logic := '0';
signal clk_12Mhz : std_logic := '0';
--Outputs
signal lcd_rs : std_logic;
signal lcd_en : std_logic;
signal lcd_data : std_logic_vector(7 downto 0);
-- Clock period definitions
constant clk_12Mhz_period : time := 10 ns;
BEGIN
-- Instantiate the Unit Under Test (UUT)
```

```
uut: LCD_FSM PORT MAP (
rst => rst,
clk_12Mhz => clk_12Mhz,
lcd_rs => lcd_rs,
lcd_en => lcd_en,
lcd_data => lcd_data
);
-- Clock process definitions
clk_12Mhz_process :process
begin
```



```
clk_12Mhz <= '0';
wait for clk_12Mhz_period/2;
clk_12Mhz <= '1';
wait for clk_12Mhz_period/2;

end process;

-- Stimulus process
stim_proc: process
begin
rst <= '1';
wait for 20 ns;
rst <= '0';
-- insert stimulus here
wait;
end process;
END;
```