PIPO CODE

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

use IEEE.STD\_LOGIC\_UNSIGNED.ALL;

entity pipo is

Port ( clk : in STD\_LOGIC;

slow\_clk: inOUT STD\_LOGIC;

load1 : in STD\_LOGIC;

in4 : in STD\_LOGIC\_VECTOR (3 downto 0);

out4 : inout STD\_LOGIC\_VECTOR (3 downto 0));

end pipo;

architecture Behavioral of pipo is

signal counter: std\_logic\_vector(1 downto 0):=(others =>'0');

signal s4:STD\_LOGIC\_VECTOR(3 DOWNTO 0);

begin

process(slow\_clk,in4,load1,out4)

begin

if(load1='1') THEN

s4 <= in4;

else

s4 <= out4;

end if;

if(slow\_clk'EVENT and slow\_clk='1') THEN

out4 <= s4;

end if;

end process;

process(clk)

begin

if (clk'event and clk='1')then

counter <=counter+'1';

end if;

end process;

slow\_clk <= counter(1);

end Behavioral;

PIPO\_TB CODE

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

entity pipo\_tb is

-- Port ( );

end pipo\_tb;

architecture Behavioral of pipo\_tb is

component pipo is

Port ( clk : in STD\_LOGIC;

slow\_clk: inout STD\_LOGIC;

load1 :in STD\_LOGIC;

in4 : in STD\_LOGIC\_VECTOR (3 downto 0);

out4 : inout STD\_LOGIC\_VECTOR (3 downto 0));

end component;

signal clk,slow\_clk, load1: std\_logic;

signal in4,out4 : std\_logic\_vector( 3 downto 0);

begin

u1: pipo port map ( clk,slow\_clk,load1,in4,out4);

process

begin

clk<= '0';

wait for 10 ns;

clk<= '1';

wait for 10 ns ;

end process;

process

begin

load1<= '1';

in4<= "1010";

wait for 300 ns;

load1<= '0';

in4<= "0011";

wait for 100 ns;

load1<= '1';

in4<= "1111";

wait for 150 ns ;

end process;

end Behavioral;