Multiplexer and Demultiplexer

The Basic function of multiplexer is used very frequently in the digital circuit technology. With the help of **multiplexer** a purposeful selected input is passed to the output. This selection is made by using the required **select signals**. The reverse procedure takes place with the use of **Demultiplexer**. In Demultiplexer the input is passed to the selected output depending on the select signals. The **Fig.-1** below illustrates the procedure decribed before.

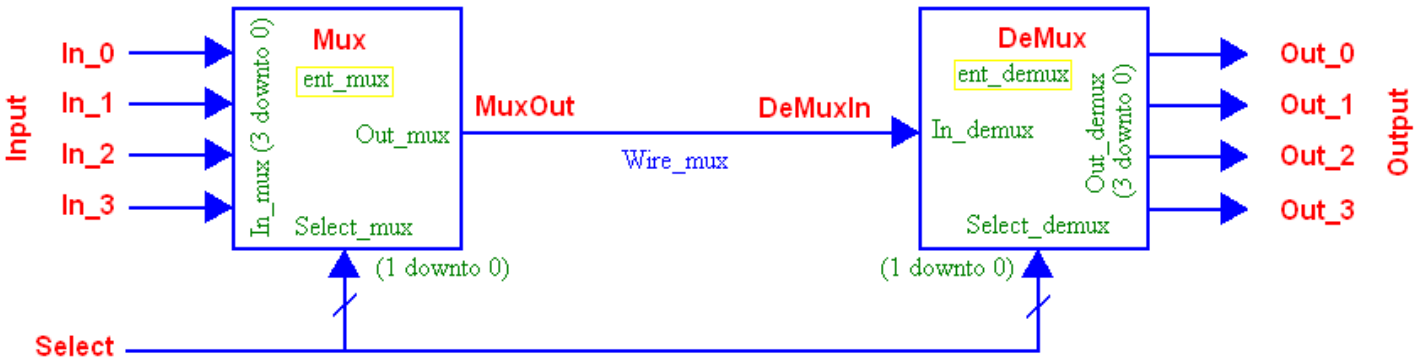

Fig.-1: Multiplexer and Demultiplexer

VHDL CODE

==========

MUX.vhdl

==========

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

-- Hendra Kesuma

entity ent_mux is
    Port ( In_mux     : in std_logic_vector (3 downto 0);
           Select_mux : in std_logic_vector (1 downto 0);
           Out_mux    : out std_logic);
end ent_mux;

architecture arch_mux of ent_mux is
begin
process (Select_mux, In_mux)
  begin
    case Select_mux is
      when "00" => Out_mux <= In_mux(0);
      when "01" => Out_mux <= In_mux(1);
      when "10" => Out_mux <= In_mux(2);
      when "11" => Out_mux <= In_mux(3);
      when others => Null;
```

```
    end case;

  end process;
end arch_mux;
```

VHDL CODE

==========

DEMUX.vhdl

==========

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

-- Hendra Kesuma

entity ent_demux is
    Port ( In_demux       : in std_logic;
           Out_demux      : out std_logic_vector (3 downto 0) ;
           Select_demux : in std_logic_vector (1 downto 0));
end ent_demux;

architecture arch_demux of ent_demux is
begin
  process (Select_demux, In_demux)
  begin
    if  Select_demux = "00" then
           Out_demux(0) <= In_demux;
           Out_demux(1) <= '0';
           Out_demux(2) <= '0';
           Out_demux(3) <= '0';
    elsif Select_demux = "01" then
           Out_demux(0) <= '0';
           Out_demux(1) <= In_demux;
           Out_demux(2) <= '0';
           Out_demux(3) <= '0';
    elsif Select_demux = "10" then
           Out_demux(0) <= '0';
           Out_demux(1) <= '0';
           Out_demux(2) <= In_demux;
           Out_demux(3) <= '0';
    elsif Select_demux = "11" then
           Out_demux(0) <= '0';
           Out_demux(1) <= '0';
           Out_demux(2) <= '0';
           Out_demux(3) <= In_demux;
    end if;
  end process;
end arch_demux;
```

VHDL CODE

```
================================================================================
MUX_DEMUX.vhdl
================================================================================
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

-- Hendra Kesuma

entity ent_mux_demux is
    Port ( In_mux_demux      : in std_logic_vector  (3 downto 0);
           Out_mux_demux     : out std_logic_vector (3 downto 0);
           Select_mux_demux : in std_logic_vector (1 downto 0));
end ent_mux_demux;

architecture arch_mux_demux of ent_mux_demux is

component ent_mux
port ( In_mux       : in std_logic_vector (3 downto 0);
       Select_mux : in std_logic_vector (1 downto 0);
       Out_mux     : out std_logic);
end component;

component ent_demux
Port ( In_demux       : in std_logic;
       Out_demux     : out std_logic_vector (3 downto 0) ;
       Select_demux : in  std_logic_vector (1 downto 0));
end component;

signal wire_mux : std_logic;
  begin
    mux: ent_mux
    port map ( In_mux       => In_mux_demux,
               Select_mux  => Select_mux_demux,
               Out_mux      => wire_mux);

    demux: ent_demux
    port map ( In_demux       => wire_mux,
               Select_demux  => Select_mux_demux,
               Out_demux      => Out_mux_demux);

end arch_mux_demux;
```

TESTBENCH CODE

```
================================================================================
tb_JK.vhd
================================================================================
LIBRARY ieee;
USE ieee.std_logic_1164.ALL;
USE ieee.numeric_std.ALL;
use work.comp_bit.ALL; -- Include package

-- Hendra Kesuma

ENTITY ent_mux_demux_tb_mux_demux_vhd_tb IS
END ent_mux_demux_tb_mux_demux_vhd_tb;

ARCHITECTURE behavior OF ent_mux_demux_tb_mux_demux_vhd_tb IS

	COMPONENT ent_mux_demux
	PORT(
		In_mux_demux	: IN std_logic_vector(3 downto 0);
		Select_mux_demux : IN std_logic_vector(1 downto 0);
		Out_mux_demux	: OUT std_logic_vector(3 downto 0)
		);
	END COMPONENT;

	SIGNAL In_mux_demux	:  std_logic_vector(3 downto 0);
	SIGNAL Out_mux_demux	:  std_logic_vector(3 downto 0);
	SIGNAL Select_mux_demux :  std_logic_vector(1 downto 0);

BEGIN

	uut: ent_mux_demux PORT MAP(
		In_mux_demux	=> In_mux_demux,
		Out_mux_demux	=> Out_mux_demux,
		Select_mux_demux => Select_mux_demux
	);


-- *** Test Bench - User Defined Section ***tb1 : PROCESS
tb : PROCESS
BEGIN
  ApplyData_Procedure(In_mux_demux(0), "1111111100000000", 5 ns);
  ApplyData_Procedure(In_mux_demux(1), "1111111100000000", 5 ns);
  ApplyData_Procedure(In_mux_demux(2), "1111111100000000", 5 ns);
  ApplyData_Procedure(In_mux_demux(3), "1111111100000000", 5 ns);

  ApplyData_Procedure(Select_mux_demux(0), "0101010101010101", 5 ns);
  ApplyData_Procedure(Select_mux_demux(1), "0011001100110011", 5 ns);

wait; -- will wait forever
END PROCESS;
```

```
-- *** End Test Bench - User Defined Section ***

END;
```

SIMULATION