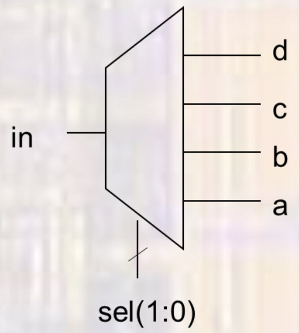
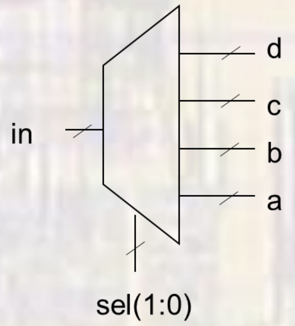
Demultiplexors

Last updated 8/22/24

Demultiplexor

- A demultiplexor routes the input to one of several outputs
  - Unused outputs are forced to '0'
  - The input and outputs can be single wires or busses

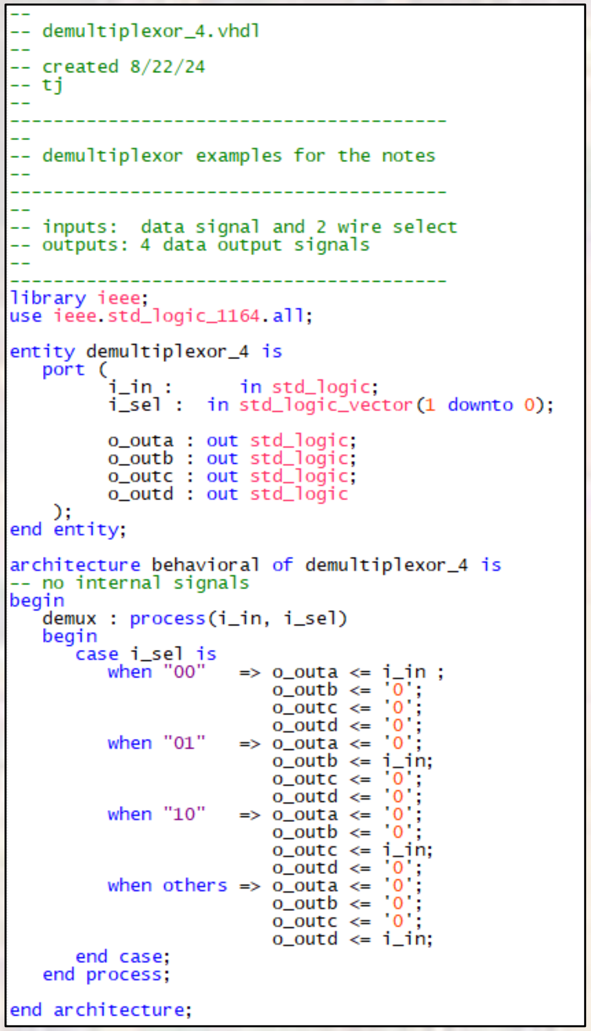




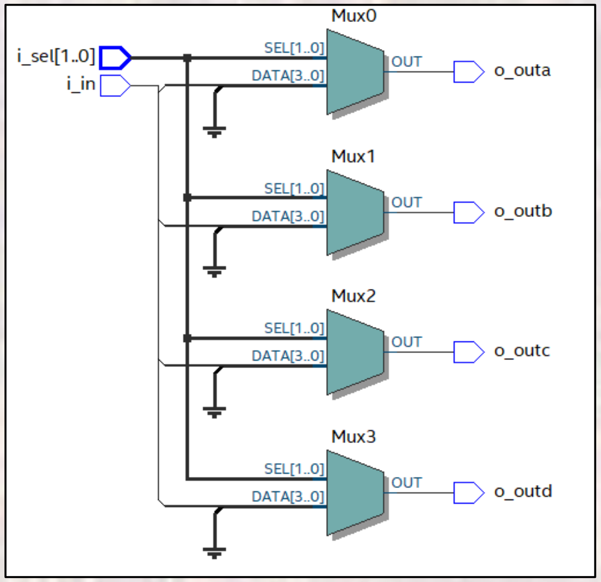
sel1	sel0	d	c	b	a
0	0	0	0	0	in
0	1	0	0	in	0
1	0	0	in	0	0
1	1	in	0	0	0

demultiplexor

- 4 output demultiplexor

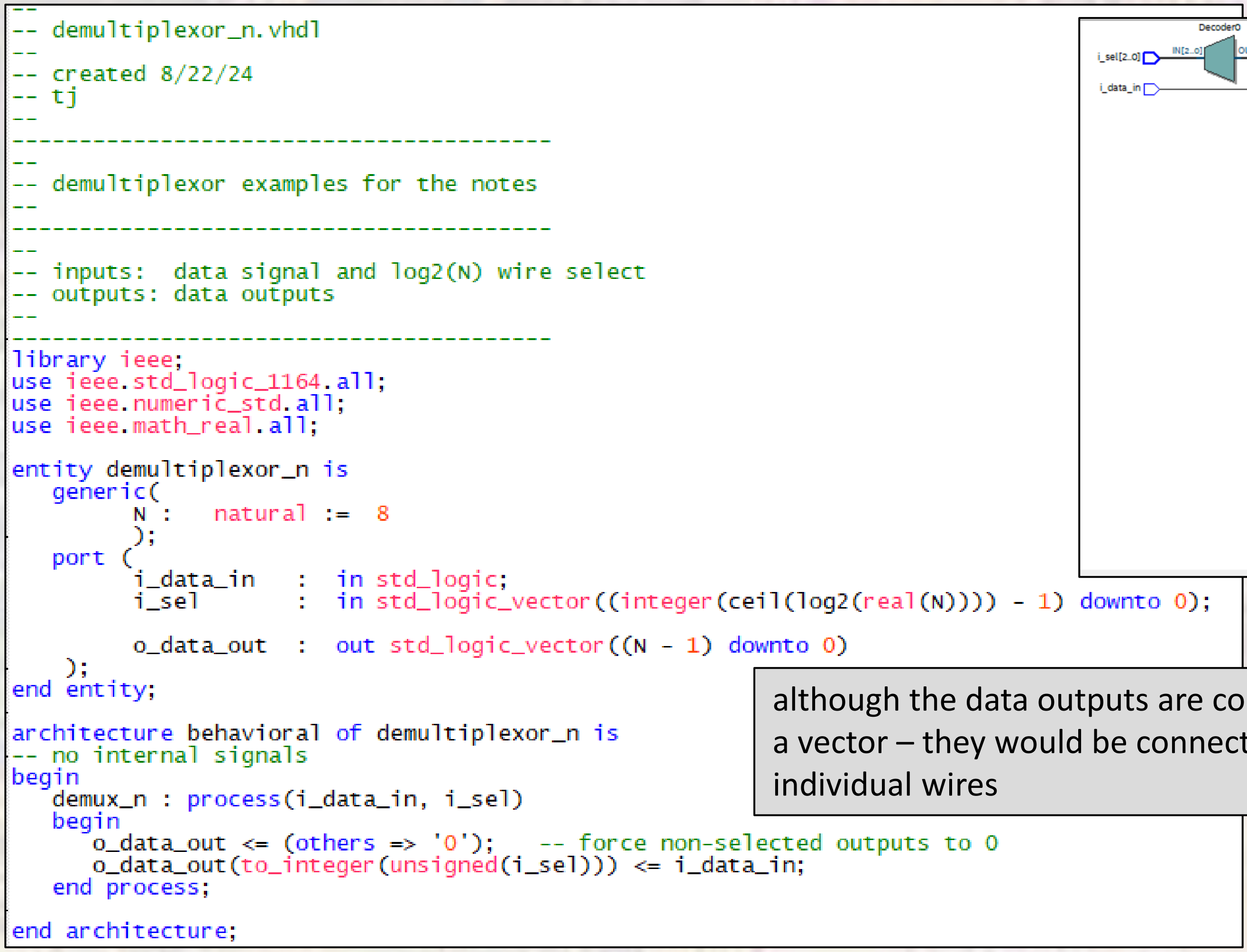


```
--
-- demultiplexor_4.vhdl
--
-- created 8/22/24
-- tj
--
-----------------------------------------
--
-- demultiplexor examples for the notes
--
-----------------------------------------
--
-- inputs:  data signal and 2 wire select
-- outputs: 4 data output signals
--
-----------------------------------------
library ieee;
use ieee.std_logic_1164.all;

entity demultiplexor_4 is
    port (
         i_in :       in std_logic;
         i_sel :  in std_logic_vector(1 downto 0);

         o_outa : out std_logic;
         o_outb : out std_logic;
         o_outc : out std_logic;
         o_outd : out std_logic
     );
end entity;

architecture behavioral of demultiplexor_4 is
-- no internal signals
begin
    demux : process(i_in, i_sel)
    begin
       case i_sel is
          when "00"   => o_outa <= i_in ;
                         o_outb <= '0';
                         o_outc <= '0';
                         o_outd <= '0';
          when "01"   => o_outa <= '0';
                         o_outb <= i_in;
                         o_outc <= '0';
                         o_outd <= '0';
          when "10"   => o_outa <= '0';
                         o_outb <= '0';
                         o_outc <= i_in;
                         o_outd <= '0';
          when others => o_outa <= '0';
                         o_outb <= '0';
                         o_outc <= '0';
                         o_outd <= i_in;

       end case;
    end process;

end architecture;
```

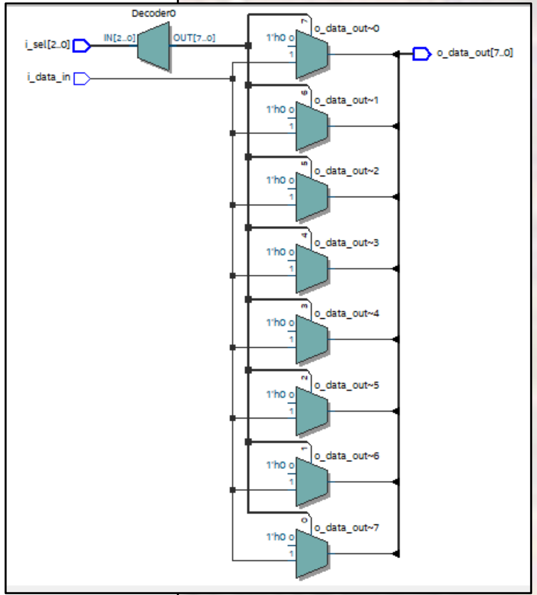


Demultiplexor

- n output demultiplexor – N must be a power of 2



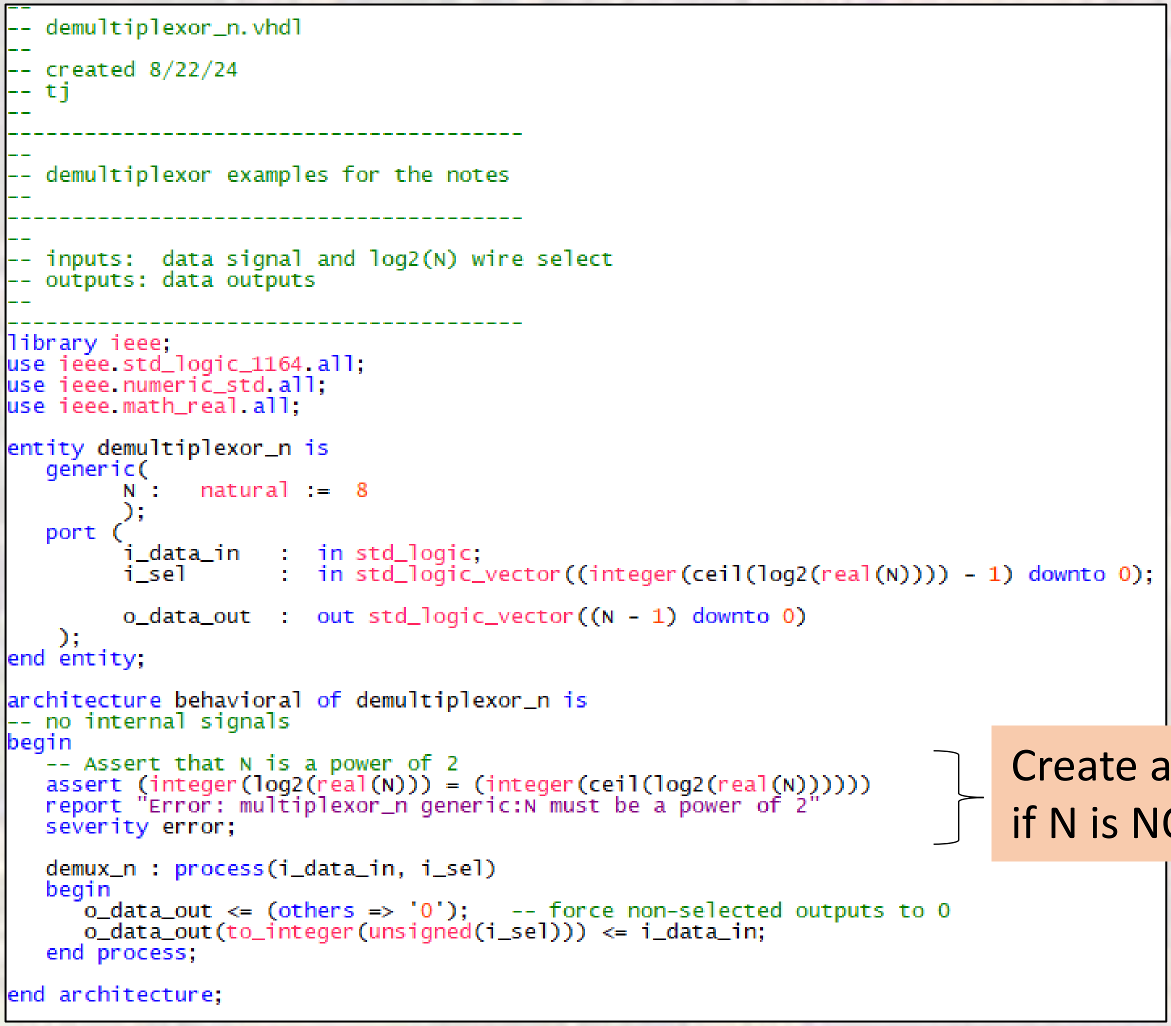
```vhdl
--
-- demultiplexor_n.vhdl
--
-- created 8/22/24
-- tj
--
----------------------------------------
--
-- demultiplexor examples for the notes
--
----------------------------------------
--
-- inputs:  data signal and log2(N) wire select
-- outputs: data outputs
--
----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.math_real.all;

entity demultiplexor_n is
   generic(
         N :   natural :=  8
         );
   port (
         i_data_in   :  in std_logic;
         i_sel       :  in std_logic_vector((integer(ceil(log2(real(N)))) - 1) downto 0);

         o_data_out  :  out std_logic_vector((N - 1) downto 0)
      );
end entity;

architecture behavioral of demultiplexor_n is
-- no internal signals
begin
   demux_n : process(i_data_in, i_sel)
   begin
      o_data_out <= (others => '0');    -- force non-selected outputs to 0
      o_data_out(to_integer(unsigned(i_sel))) <= i_data_in;
   end process;

end architecture;
```



although the data outputs are configured as a vector – they would be connected as individual wires

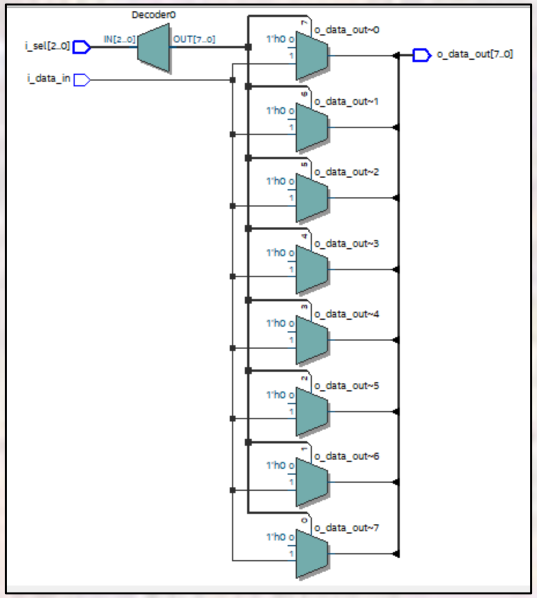
Demultiplexor

- n input demultiplexor – check if N is a power of 2



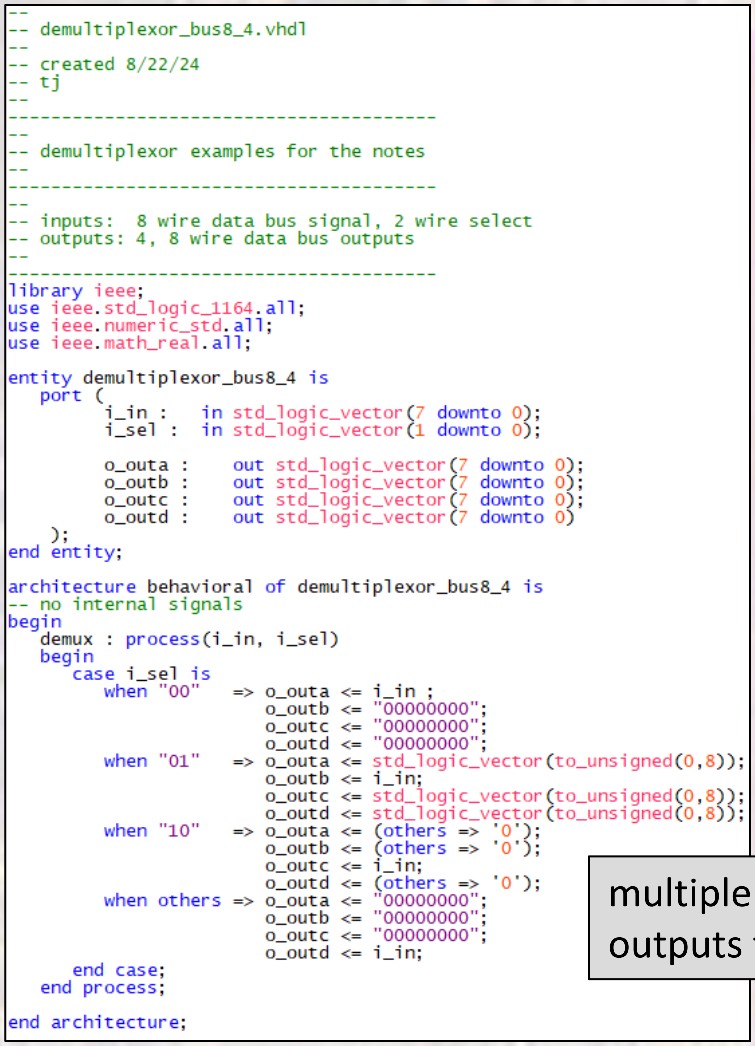
```
--
-- demultiplexor_n.vhdl
--
-- created 8/22/24
-- tj
--
-----------------------------------------
--
-- demultiplexor examples for the notes
--
-----------------------------------------
--
-- inputs:  data signal and log2(N) wire select
-- outputs: data outputs
--
-----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.math_real.all;

entity demultiplexor_n is
   generic(
         N :    natural :=  8
         );
   port (
         i_data_in    :  in std_logic;
         i_sel        :  in std_logic_vector((integer(ceil(log2(real(N)))) - 1) downto 0);

         o_data_out   :  out std_logic_vector((N - 1) downto 0)
    );
end entity;

architecture behavioral of demultiplexor_n is
-- no internal signals
begin
   -- Assert that N is a power of 2
   assert (integer(log2(real(N))) = (integer(ceil(log2(real(N))))))
   report "Error: multiplexor_n generic:N must be a power of 2"
   severity error;

   demux_n : process(i_data_in, i_sel)
   begin
      o_data_out <= (others => '0');    -- force non-selected outputs to 0
      o_data_out(to_integer(unsigned(i_sel))) <= i_data_in;
   end process;

end architecture;
```

Create a compiler error if N is NOT a power of 2

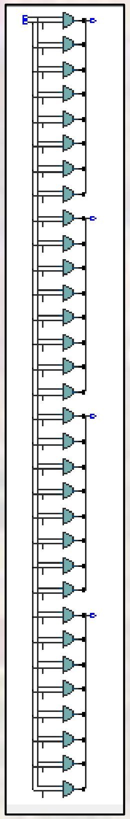


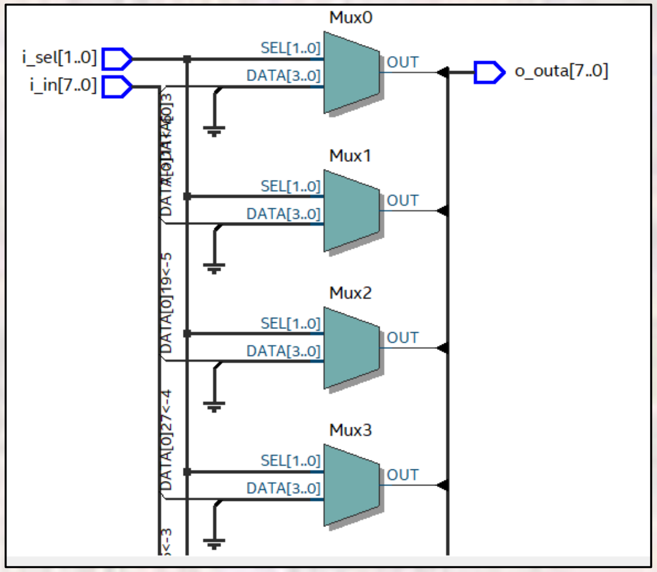
Demultiplexor

- bus demultiplexor – 8 wire, 4 way



```vhdl
--
-- demultiplexor_bus8_4.vhdl
--
-- created 8/22/24
-- tj
--
-----------------------------------------
--
-- demultiplexor examples for the notes
--
-----------------------------------------
--
-- inputs:  8 wire data bus signal, 2 wire select
-- outputs: 4, 8 wire data bus outputs
--
-----------------------------------------
library ieee;
use ieee.std_logic_1164.all;
use ieee.numeric_std.all;
use ieee.math_real.all;

entity demultiplexor_bus8_4 is
   port (
         i_in :   in std_logic_vector(7 downto 0);
         i_sel :  in std_logic_vector(1 downto 0);

         o_outa :    out std_logic_vector(7 downto 0);
         o_outb :    out std_logic_vector(7 downto 0);
         o_outc :    out std_logic_vector(7 downto 0);
         o_outd :    out std_logic_vector(7 downto 0)
      );
end entity;

architecture behavioral of demultiplexor_bus8_4 is
-- no internal signals
begin
   demux : process(i_in, i_sel)
   begin
      case i_sel is
         when "00"    => o_outa <= i_in ;
                         o_outb <= "00000000";
                         o_outc <= "00000000";
                         o_outd <= "00000000";
         when "01"    => o_outa <= std_logic_vector(to_unsigned(0,8));
                         o_outb <= i_in;
                         o_outc <= std_logic_vector(to_unsigned(0,8));
                         o_outd <= std_logic_vector(to_unsigned(0,8));
         when "10"    => o_outa <= (others => '0');
                         o_outb <= (others => '0');
                         o_outc <= i_in;
                         o_outd <= (others => '0');
         when others => o_outa <= "00000000";
                         o_outb <= "00000000";
                         o_outc <= "00000000";
                         o_outd <= i_in;

      end case;
   end process;

end architecture;
```





multiple ways to set the unselected outputs to 0

Demultiplexor

- bus demultiplexor – M wire, N way

see if you can do this