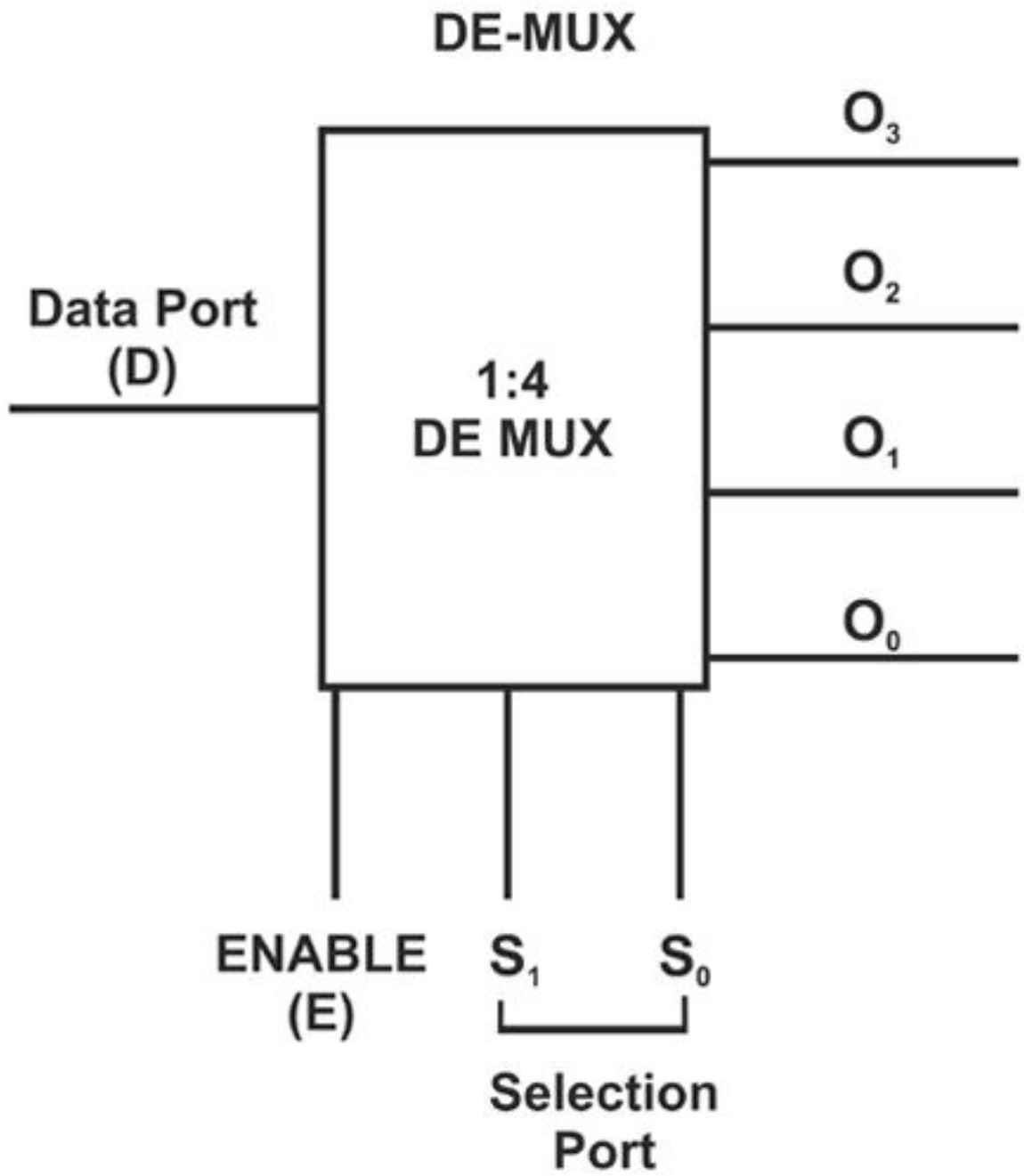
DE-MUX
O_3
O_2
O_1
O_0
Data Port
(D)
1:4
DE MUX
ENABLE
(E)
S_1
S_0
Selection
Port

Data Line(1/0)	Enable	Selection Lines(S1 S0)	Output Ports(O3 O2 O1 O0)
D	0	X	XXXX
D	1	00	D000
D	1	01	0D00
D	1	10	00D0
D	1	11	000D

```
Library ieee;
use ieee.std_logic_1164.all;
entity demux_41 is
port( signal data :in std_logic;
signal select_line: in std_logic_vector(1 downto 0);
signal enable: in std_logic;
signal output:out std_logic_vector(3 downto 0  ));
end demux_41;
architecture sim of demux_41 is
begin
 process(select_line,enable)
begin
if(select_line="00" and enable='1') then
output(3)<=data;
output(2) <='0';
```

```
output(1)<='0';
output(0)<='0';
elsif(select_line="01" and enable='1') then
output(3)<='0';
output(2) <=data;
output(1)<='0';
output(0)<='0';
elsif (select_line="10" and enable='1') then
output(3)<='0';
output(2) <='0';
output(1)<=data;
output(0)<='0';
elsif (select_line="11" and enable='1') then
output(3)<='0';
output(2) <='0';
```

```
output(1)<='0';
output(0)<=data;
else
output(3)<='X';
output(2) <='X';
output(1)<='X';
output(0)<='X';
end if;
end process;
end sim;
```

process(select_line, enable): Since the output is dependent on selection ports and enable signals, its placed under process.

The rest of the code is nothing but declaring the if conditions and directing it to switch the outputs based on the specified values. The architecture segment of the VHDL code is given below for your reference.