curs4.

LIBRĂRii...
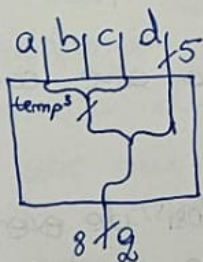ENTITATE...
ARCHITECTURE...
   - zonă declarativă
BEGIN
   - zonă instrucțiunii concurente
END

> Atribuirea directă a semnalelor.
>
> Se face prin operatorul " <= "

ex] Să se descrie un modul digital care realizea-
tă operații de atribuire concurente pt mai
multe tipuri de semnale.

'0', '1', 'z', 'H', 'L'

```
LIBRARY IEEE;
USE IEEE.std_logic_1864.all;
ENTITY ex_mod is
PORT ( a,b,c: IN std_logic;
          d: IN std_logic_vector (4 DOWNTO 0);
          g: OUT std_logic_vector (7 DOWNTO 0));
END ex_mod;
ARCHITECTURE dscr OF ex_mod is
```

```
SIGNAL temp: std_logic_vector (2 DOWNTO 0);
BEGIN
    temp <= std_logic_vector'(a,b,c);
    q <= temp & d;
END dscr;
```

$$\boxed{q <= a \, \& \, b \, \& \, c \, \& \, d;}$$

$$\boxed{\& = \text{şi}}$$

Atribuirea condiţională
a semnalelor

Atribuirea unui semnal sau a unei expresii se realizează prin specificaţia "WHEN/ELSE"

```
semnal <= expresie1 WHEN   condiţie_1 ELSE
          expresie 2 WHEN   condiţie_2 ELSE
          . . . . .
          expresie N;
```

Aceasta este o specificaţie care implică prioritizarea condiţiilor;
Prioritatea cea mai mare o are condiţia 1.

Această specificaţie se poate folosi de exemplu pt. testarea semnalelor prioritare de tip:

- reset
- load
- set
- init

În urma sintezei această specificaţie generează o cascadă de mai multe multiplexoare.

expresie 1
expresie 2
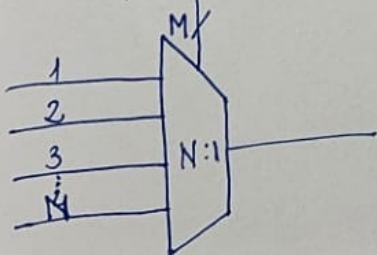conditie 1

expresie N
conditie N-1
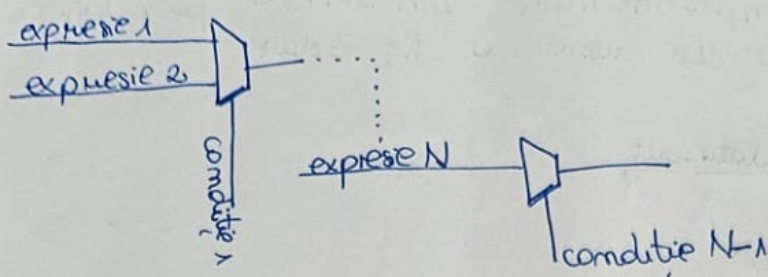
Atribuirea selectivă a unui semnal.

Atribuirea selectivă se realizează prin specificația "WITH / SELECT"

```
WITH expresie selectie SELECT
semnale <= expresie_1 WHEN selectie_1;
          expresie_2 WHEN selectie_2;
          ⋮
          expresie_N WHEN OTHERS;
```

Această specificație nu implică atribuiri condiționale prioritare, practic, fiecare expresie e atribuită după o alternativă (relație), în cazul în care nu sunt atribuite toate alternativele, clauza "OTHERS" preia automat alternativele rămase.

În urma sintezei se va genera un circuit multiplexor N:1

ex] Să se implementeze um BUFFER pe 8 biți,
cònaia portul de iesire nà fie 3 stăte:

data_im ▷ data_out
   /8   /8
   └ emable
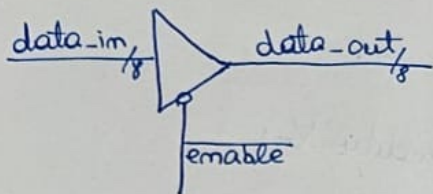
data_in	emable	data_out
X	0	X
X	1	Z/Hiz

```
LIBRARY iEEE;
USE iEEE. std_logic_1164. all.
ENTITY buff_8b is
    PORT (data_iN std_logic_vecton (7 DOWNTO 0);
          data_OUT std_logic_vecton (7 DOWNTO 0);
          emable : iN std_logic );
END buff_8b;
ARCHITECTURE dscn OF buff_8b is
    BEGiN
        data_out <= data_im WHEN emable = 0 ELSE
                    (OTHERS => 'z')

END dscn;
```