dscal
DIGITAL SYSTEMS & COMPUTER ARCHITECTURE LABORATORY

Εργαστήριο Λογικής Σχεδίασης

6η Διάλεξη

Οι τρεις τύπο αρχιτεκτονικής
Conditional & Selected Assignments

Βασιλόπουλος Διονύσης

Ε.ΔΙ.Π. Τμήματος Πληροφορικής & Τηλεπικοινωνιών

ΤΜΗΜΑ
ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

VHDL - Παράδειγμα

Πραγματικό πρόβλημα

Σε ένα Computer Room υπάρχουν δύο (2) αισθητήρες θερμοκρασίας (**Sensor_1** και **Sensor_2**) και δύο (2) κλιματιστικά (**AirCond_1** και **AirCond_2**). Το πρώτο κλιματιστικό (AirCond_1) λειτουργεί εάν τουλάχιστον ένας από τους δύο αισθητήρες ανιχνεύσουν θερμοκρασία άνω των 35 βαθμών στο computer room. Το δεύτερο κλιματιστικό (AirCond_2) λειτουργεί εάν και οι δύο αισθητήρες ανιχνεύσουν θερμοκρασία άνω των 35 βαθμών στο computer room. Θεωρείστε ότι κάθε αισθητήρας δίνει σήμα (‘1’) μόνο όταν η θερμοκρασία που ανιχνεύει γίνει μεγαλύτερη των 35 βαθμών (>35). Σε άλλη περίπτωση ο αισθητήρας στέλνεις την τιμή ‘0’. Σχεδιάστε και υλοποιήστε το λογικό κύκλωμα που περιγράφει το ανωτέρω πρόβλημα. Το όνομα του Vivado Project θα είναι Lab_1, της οντότητας θα είναι επίσης Lab_1 ενώ το όνομα της αρχιτεκτονικής Lab1_Beh.

Σχεδιάστε και υλοποιήστε το λογικό κύκλωμα που περιγράφει το ανωτέρω πρόβλημα.

VHDL - Παράδειγμα

Υλοποίηση Αρχιτεκτονικής (Dataflow)

```
architecture Dataflow of CR_AC is
begin

        AirCond_1<=Sensor_1 or Sensor_2;
        AirCond_2<=Sensor_2 and Sensor_3;

end architecture Dataflow;
```

VHDL - Παράδειγμα

Υλοποίηση Αρχιτεκτονικής (Dataflow)

1. Αποτελείται από απλές εντολές ανάθεσης τιμών σε σήματα
2. Κάθε εντολή εκτελείται όταν μεταβληθεί η τιμή ενός σήματος στο αριστερό μέρος.
3. Όλες οι εντολές ανάθεσης εκτελούνται ταυτόχρονα (παράλληλα)
4. Οι εντολές ανάθεσης αντιστοιχούν σε λογικές πράξεις της άλγεβρας Boole.
5. Το RTL διάγραμμα που προκύπτει είναι μία απεικόνιση της άλγεβρας Boole που εκφράζουν οι εντολές ανάθεσης σε στοιχειώδεις λογικές πύλες (AND, OR, NOT) και πολυπλέκτες.

VHDL - Παράδειγμα

Υλοποίηση Αρχιτεκτονικής (Behavioral)

```
architecture behavioral of CR_AC is
begin

room:  process (Sensor_1, Sensor_2) is
          begin
               AirCond_1<=Sensor_1 or Sensor_2;
               AirCond_2<=Sensor_1 and Sensor_2;
          end process room;

end architecture Behavioral;
```

VHDL

Υλοποίηση Αρχιτεκτονικής (Behavioral)

1. Όμοιο RTL Design με το Dataflow
2. ΔΕΝ χρειάζεται να αλλάξουμε τίποτα στην προσομοίωση
3. Εισαγωγή της δομής process (με λίστα ευαισθησίας)
4. Σε περίπτωση που υπάρχει μόνο η εντολή process, τότε ουσιαστικά οι εντολές μας εκτελούνται σειριακά
5. Σε περίπτωση που υπάρχουν και απλές εντολές ανάθεσης στην αρχιτεκτονική, τότε μπορείτε να θεωρήσετε όλη τη δομή process σαν μία εντολή που εκτελείται παράλληλα με τις εντολές ανάθεσης (που είναι εκτός process)

VHDL - Παράδειγμα

Υλοποίηση Αρχιτεκτονικής (Structural – Δήλωση Οντοτήτων)

```
entity OR_gate is
port(A  : in  std_logic;
     B  : in  std_logic;
     O  : out std_logic);
end entity OR_gate;

architecture Dataflow of OR_gate is
begin
   O<=A or B;
end Dataflow;
```

VHDL - Παράδειγμα

Υλοποίηση Αρχιτεκτονικής (Structural – Δήλωση Οντοτήτων)

```
entity AND_gate is
port(A  : in  std_logic;
     B  : in  std_logic;
     O  : out std_logic);
end entity AND_gate;

architecture Dataflow of AND_gate is
begin
   O<=A AND B;
end Dataflow;
```

VHDL - Παράδειγμα

Υλοποίηση Αρχιτεκτονικής (Structural – Δήλωση Αρχιτεκτονικής Κύριας Οντότητας)

```
architecture Structural of CR_AC is

component AND_gate is
port(A  : in  std_logic;
     B  : in  std_logic;
     O  : out std_logic);
end component AND_gate;

component OR_gate is
port(A  : in  std_logic;
     B  : in  std_logic;
     O  : out std_logic);
end component OR_gate;

begin

or_comp : OR_gate  port map (A=>Sensor_1,
B=>Sensor_2, O=>AirCond_1);
and_comp: AND_gate port map (A=>Sensor_1,
B=>Sensor_2, O=>AirCond_2);

end architecture Structural;
```

VHDL

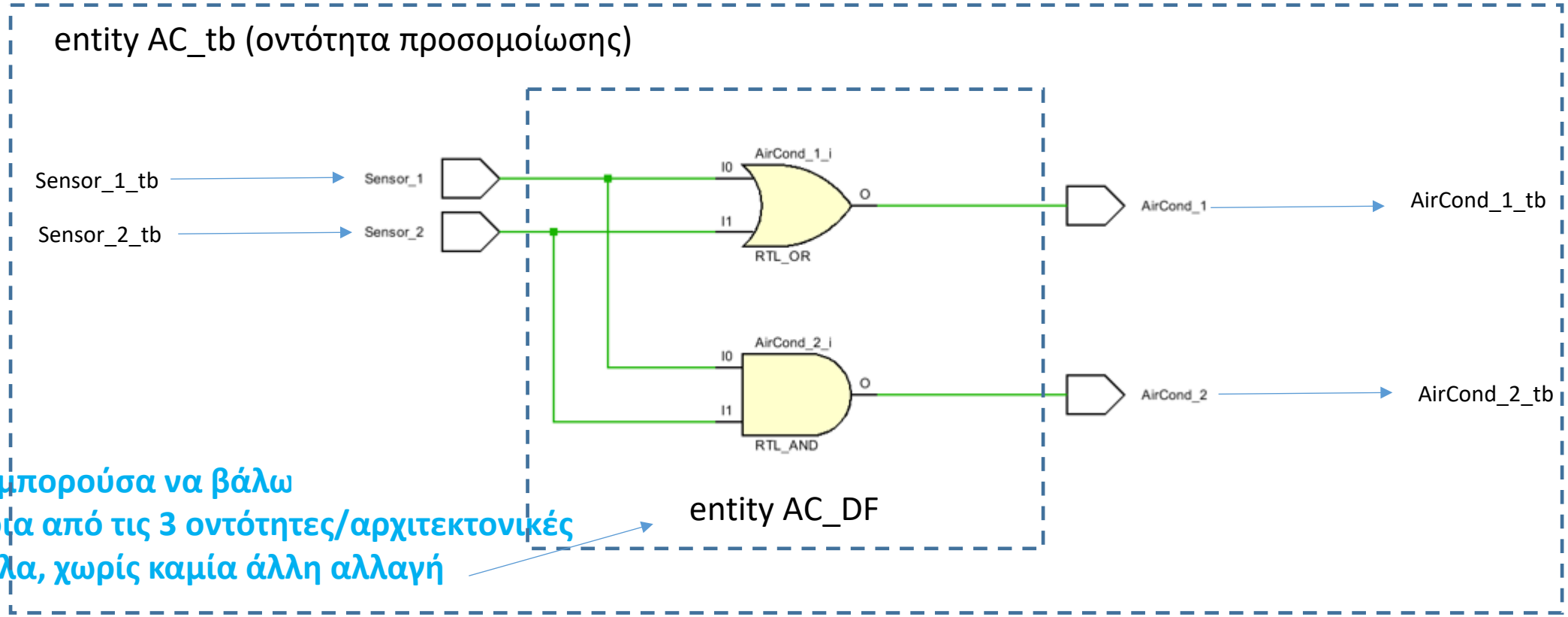
Υλοποίηση Αρχιτεκτονικής (Structural)

1. Για την υλοποίηση της αρχιτεκτονικής χρησιμοποιούμε τη λειτουργικότητα άλλων Οντοτήτων που έχουμε ήδη υλοποιήσει.
2. Τις οντότητες που θα χρησιμοποιήσουμε τις δηλώνουμε (όνομα και port) ανάμεσα στο is και το begin της αρχιτεκτονικής. Όμως πλέον έχουν την έννοια της συνιστώσας (component).
3. Η αρχιτεκτονική πλέον αποτελείτε μόνο από εντολές που καλούν(δημιουργούν) τα components, τα οποία μπορεί και να επικοινωνούν μεταξύ τους (συνηθέστερη περίπτωση).
4. Άρα μια οντότητα μπορεί να χρησιμοποιεί άλλες οντότητες (με τη μορφή component), οι οποίες μπορεί να είναι υλοποιημένες με οποιοδήποτε από τις 3 αρχιτεκτονικές. Στη περίπτωση της structural δομής βλέπουμε ότι σχηματίζεται ένα δέντρο, τα φύλλα του οποίου είναι οντότητες. **Τα φύλλα που είναι τερματικά έχουν δομή Dataflow ή Behavioral**.
5. **ΔΕΝ αλλάζει τίποτα στην προσομοίωση**

https://buzztech.in/vhdl-modelling-styles-behavioral-dataflow-structural/

VHDL – Παράδειγμα σε Vivado

Ψηφιακό κύκλωμα - VHDL: Προσομοίωση - DF



VHDL – Παράδειγμα σε Vivado

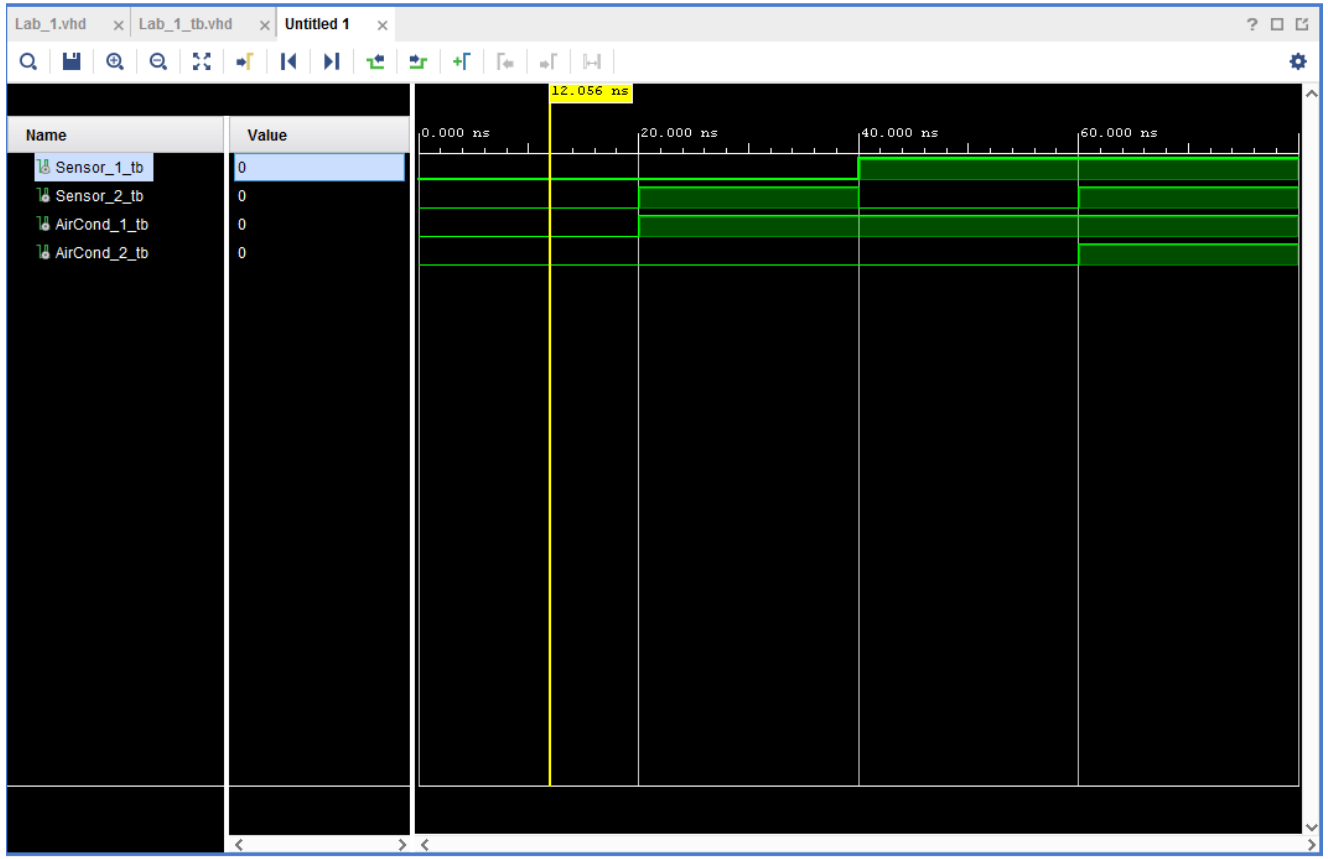
Προσομοίωση (Πρόγραμμα testbench)

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

entity CR_AC_tb is
end CR_AC_tb;

architecture Behavioral of CR_AC_tb is

component CR_AC is
Port (
Sensor_1    :in std_logic;
Sensor_2    :in std_logic;
AirCond_1  :out std_logic;
AirCond_2  :out std_logic);
end component CR_AC;

-- Internal Signals: One for each port
signal Sensor_1_tb    :std_logic; signal Sensor_2_tb    :std_logic;
signal AirCond_1_tb   :std_logic; signal AirCond_2_tb   :std_logic;
```

```
begin

-- create entity into testbench entity
test_entity: CR_AC port map (Sensor_1=>Sensor_1_tb,
Sensor_2=>Sensor_2_tb, AirCond_1=>AirCond_1_tb,
AirCond_2=>AirCond_2_tb);

enter_test_values: process is
begin

Sensor_1_tb<='0';Sensor_2_tb<='0';wait for 20 ns;
Sensor_1_tb<='0';Sensor_2_tb<='1';wait for 20 ns;
Sensor_1_tb<='1';Sensor_2_tb<='0';wait for 20 ns;
Sensor_1_tb<='1';Sensor_2_tb<='1';wait for 20 ns;

end process enter_test_values;

end  architecture Behavioral;
```

VHDL – Παράδειγμα σε Vivado

Προσομοίωση Behavioral (Χρονοσειρά)



VHDL –Selected assignment

Example 1

Ένα σύστημα δέχεται στην είσοδο το σήμα a των 2 bit και στην έξοδο υπάρχει το σήμα b των 3 bit που αντιστοιχεί στη μετάδοση του σήματος a με επιπλέον bit άρτιας ισοτιμίας (b=parity_bit&a).

VHDL – Selected assignment (Dataflow)

Example 2: With Select

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;USE ieee.numeric_std.ALL;



entity selected is
port (a  :   in   std_logic_vector(1 downto 0);
       b  :  out std_logic_vector(2 downto 0)
);

end entity selected;
```

```
architecture Dataflow of selected is

begin

with a select
b<="000" when "00",
     "101" when "01",
     "110" when "10",
     "011" when "11",
     "000" when others;

end architecture Dataflow ;
```

Λογική Συνθήκη

Ανάθεση με επιλογή
(διακριτές τιμές συγκεκριμένου σήματος)

Μία εντολή: Ανάθεση τιμής σε ένα σήμα

Πρέπει να ελέγχω ΟΛΕΣ τις τιμές

VHDL – Selected assignment (Behavioral)

Example 2: Case

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;USE ieee.numeric_std.ALL;



entity selected is
port (a  :   in   std_logic_vector(1 downto 0);
        b  :  out std_logic_vector(2 downto 0)
);

end entity selected;
```

```
architecture Behavioral of selected is

begin

test: process (a) is

begin

case a is
when “00” => b<=“000";
when “01” => b<=“101";
when “10” => b<=“110";
when “11” => b<=“011";
when others => b<=“000";
end case;
end process test;
end architecture Behavioral;
```

Μπορεί και πολλαπλές εντολές ανά περίπτωση (when)

case
Διακριτές τιμές

Πρέπει να ελέγχω ΟΛΕΣ τις τιμές

VHDL –Conditional assignment

Example 2

Σε ένα Computer Room υπάρχουν 3 κλιματιστικά και ένας αισθητήρας θερμοκρασίας (θερμόμετρο). Εάν το θερμόμετρο δείξει θερμοκρασία κάτω των 25 βαθμών τότε δεν λειτουργεί το κλιματιστικό. Αν η θερμοκρασία είναι μέχρι 30 βαθμούς λειτουργεί το 1ο. Αν η θερμοκρασία είναι μέχρι 45 βαθμούς λειτουργεί και το 2ο. Εάν είναι πάνω από 45 λειτουργεί και το 3ο. Θεωρείστε ότι η σειρά λειτουργίας των κλιματιστικών είναι σαφώς ορισμένη και δεν μας απασχολεί για το πρόβλημά μας. Είσοδος του συστήματος το σήμα temperature και έξοδος το σήμα action (έχει 4 δυνατές τιμές).

VHDL – Conditional assignment (Dataflow)

Example 2: When

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;USE ieee.numeric_std.ALL;

entity conditional is
port (temperature:  in std_logic_vector(5 downto 0);
        action          :  out std_logic_vector(1 downto 0)
);
end entity conditional;

architecture Dataflow of conditional is
signal temp :unsigned(5 downto 0);
begin

temp<= unsigned(temperature);
action<="00" when temp<25 else
      "01" when (temp>=25 and temp<30) else
      "10"  when (temp>=30 and temp<45) else
      "11";


end architecture Dataflow ;
```

max=2^5-1=63

action:
00: Δεν δουλεύει τίποτα
01: Δουλεύει 1 AC
10: Δουλεύουν 2 AC
11: Δουλεύουν και τα 3 AC

Λογική Συνθήκη

Μία εντολή: Ανάθεση τιμής σε ένα σήμα

VHDL – Conditional assignment (Behavioral)

Example 2: If

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;USE ieee.numeric_std.ALL;

entity conditional is
port (temperature:  in std_logic_vector(5 downto 0);
        action          :   out std_logic_vector(1 downto 0)
);
end entity conditional;

architecture Behavioral of conditional is
signal temp :unsigned(5 downto 0);
begin
temp<= unsigned(temperature);

action_temp: process(temp) is
begin
if (temp<25) then
   action<="00";
elsif (temp<30) then
   action<="01";
elsif (temp<45) then
   action<="10";
else
   action<="11";
end if;
end process action_temp;

end architecture Behavioral;
```

Μπορεί και πολλαπλές εντολές ανά περίπτωση (if/elsif/else)

if, **elsif**, else
Μόνο μέσα
σε process

Πρέπει να
ελέγχω ΟΛΕΣ
τις τιμές

VHDL – Selected & Conditional assignment

Σύνοψη

1.
signal <= value when condition else …

2.
with signal_1 select
signal_2<=value when (discrete signal_1 value),
…

Και στις 2 περιπτώσεις είμαστε απευθείας στο σώμα της αρχιτεκτονικής και η εντολή αφορά απόδοσης τιμής σε ένα συγκεκριμένο σήμα

3.
If condition then action;elsif … else end if

4.
case signal is
when value => assignment (discrete signal value),
…

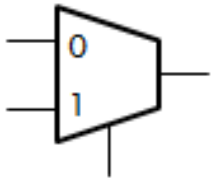
Και στις 2 περιπτώσεις είμαστε στο σώμα process (ή procedure) και αφορά την εκτέλεση μίας η περισσότερων εντολών ανάλογα τη συνθήκη, και μπορεί να αφορά πάνω από ένα σήματα.

http://www.pldworld.com/_hdl/2/_ref/acc-eda/language_overview/concurrent_statements/conditional_vs__selected_assignment.htm

VHDL – Multiplexer

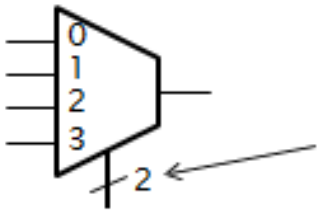
- Επιλέγει μεταξύ εισόδων δεδομένων με βάση μια είσοδο επιλογής

2-σε-1 πολυπλέκτης



sel	z
0	a_0
1	a_1

4-σε-1 πολυπλέκτης



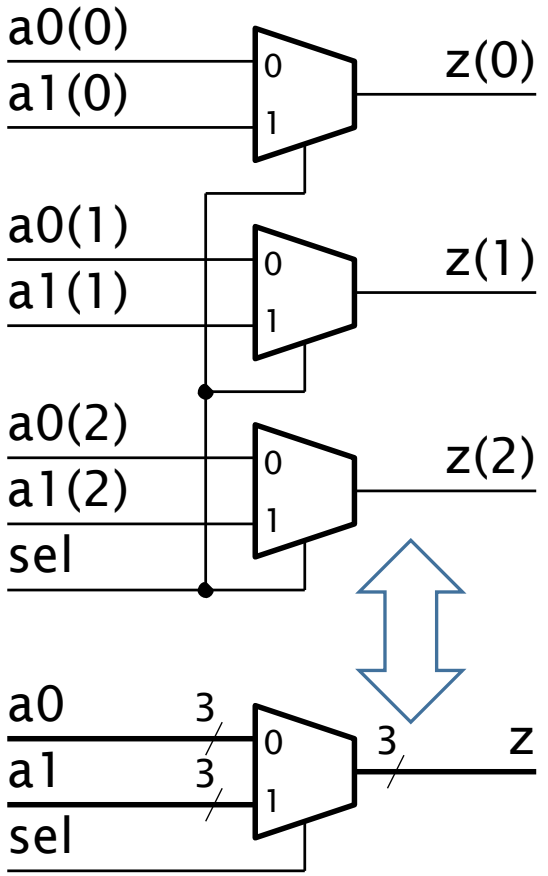
δύο bit επιλογής

sel	z
00	a_0
01	a_1
10	a_2
11	a_3

- Ένας N-σε-1 πολυπλέκτης απαιτεί $\lceil \log_2 N \rceil$ bit επιλογής

VHDL – Multiplexer

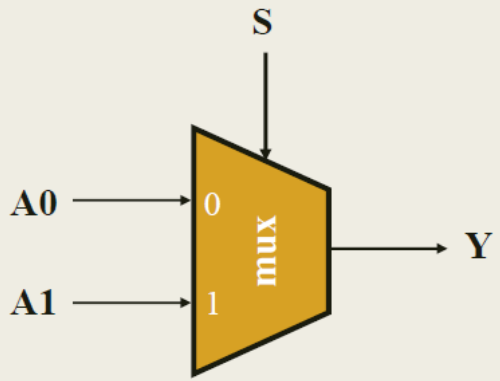
- Για να επιλέξουμε μεταξύ *N* κωδικών λέξεων των *m* bit
  - Συνδέουμε παράλληλα *m* πολυπλέκτες των *N* εισόδων



VHDL – Multiplexer

Entity (2to1)

```
entity MUX_2_to_1 is
    port (
      A0: in STD_LOGIC;
      A1: in STD_LOGIC;
      S: in STD_LOGIC;
      Y: out STD_LOGIC);
end MUX_2_to_1;
```



VHDL – Multiplexer

Architecture (2to1 - behavioral)

Περιγραφή συμπεριφοράς

```
architecture BEHAVIORAL of MUX_2_to_1 is
begin
  process (A0, A1, S)
  begin
    if (S = '0') then
      Y <= A0;
    else
      Y <= A1;
    end if;
  end process;
end BEHAVIORAL;
```

Στη λίστα ευαισθησίας συμπεριλαμβάνονται όλες οι είσοδοι του συνδυαστικού κυκλώματος

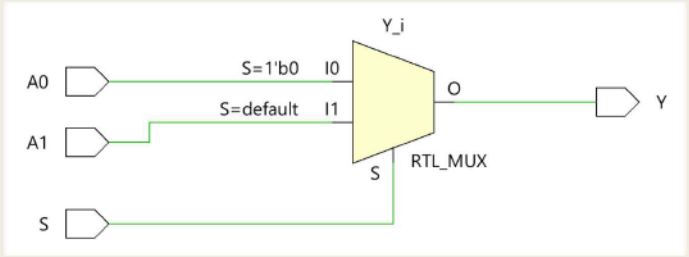
y<=A0 when s='0' else A1;

VHDL – Multiplexer

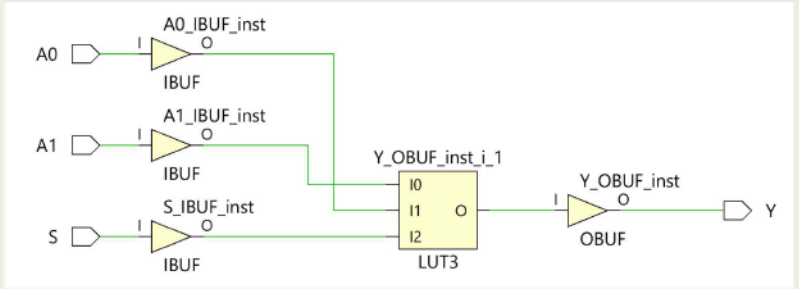
Architecture mux 2to1

Η αρχιτεκτονική του πολυπλέκτη 2 σε 1 στη VHDL

- Σχηματικό διάγραμμα RTL



- Σχηματικό διάγραμμα σε τεχνολογία FPGA

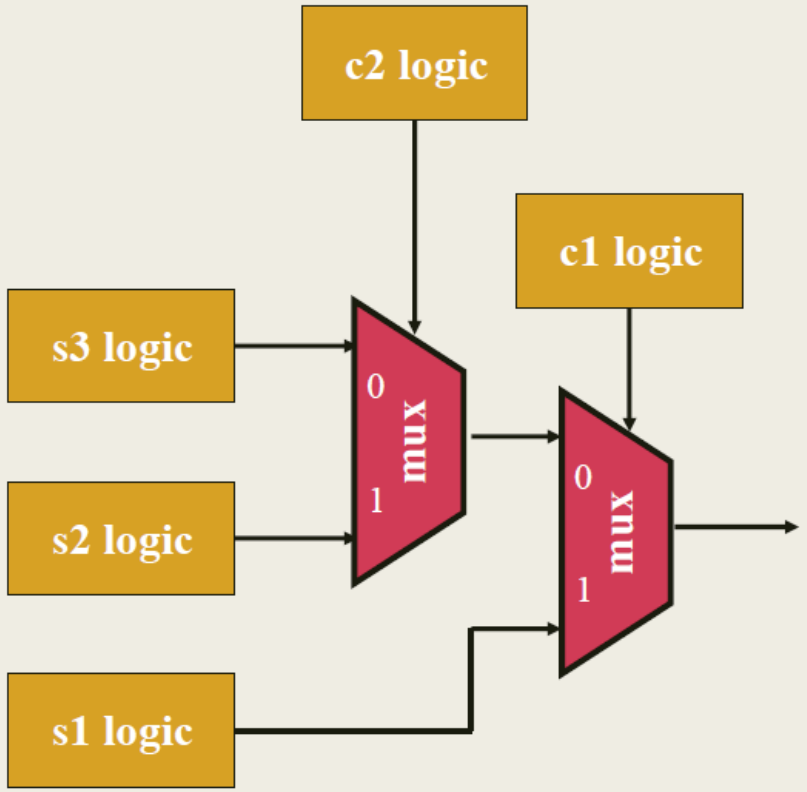


VHDL – Multiplexer

Υλοποίηση If

- Υλοποίηση εντολής IF με τη χρήση πολυπλεκτών 2 σε 1

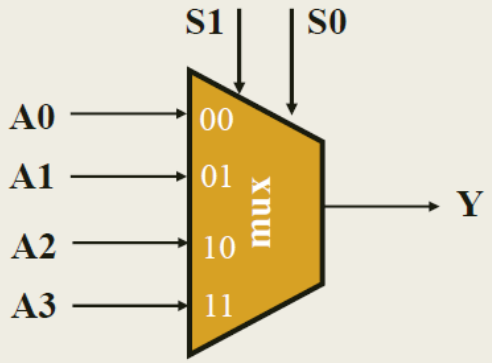
```
if c1 then
    s1;
elsif c2 then
    s2;
else
    s3;
end if;
```



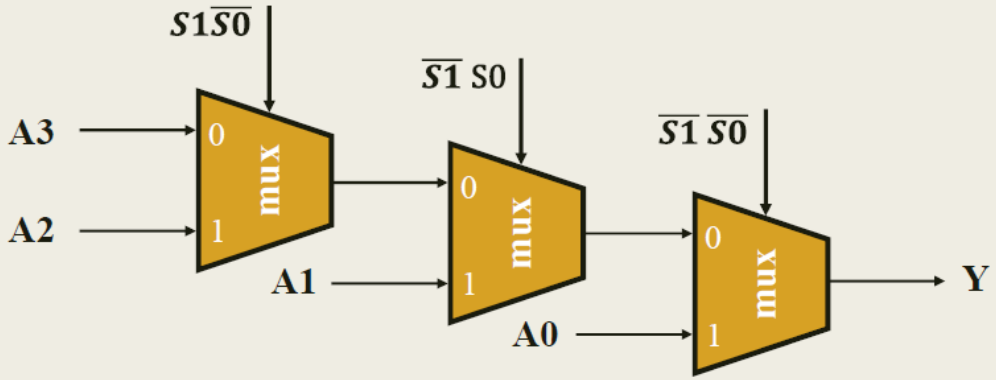
VHDL – Multiplexer

Entity (4to1)

```
entity MUX_4_to_1 is
    port (
      A0: in STD_LOGIC;
      A1: in STD_LOGIC;
      A2: in STD_LOGIC;
      A3: in STD_LOGIC;
      S0: in STD_LOGIC;
      S1: in STD_LOGIC;
      Y: out STD_LOGIC);
end MUX_4_to_1;
```



Λύση 1: Υλοποίηση με πολυπλέκτες 2 σε 1 σε δομή αλυσίδας



VHDL – Multiplexer

Architecture (4to1)

Η αρχιτεκτονική του πολυπλέκτη 4 σε 1 στη VHDL
Περιγραφή συμπεριφοράς – Λύση 1

```
architecture BEHAVIORAL of MUX_4_to_1 is
begin
  process (A0, A1, A2, A3, S0, S1)
  begin
    if    (S1 = '0' and S0 = '0') then Y <= A0;
    elsif (S1 = '0' and S0 = '1') then Y <= A1;
    elsif (S1 = '1' and S0 = '0') then Y <= A2;
    else                               Y <= A3;
    end if;
  end process;
end BEHAVIORAL;
```

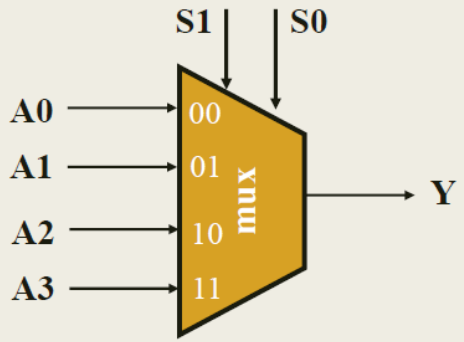
Στη λίστα ευαισθησίας συμπεριλαμβάνονται όλες οι είσοδοι του συνδυαστικού κυκλώματος

VHDL – Multiplexer

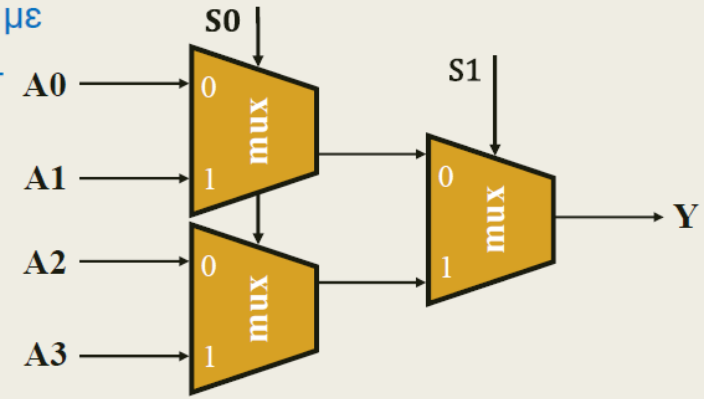
Entity (4to1)

Η οντότητα του πολυπλέκτη 4 σε 1 στη VHDL

```
entity MUX_4_to_1 is
    port (
      A0: in STD_LOGIC;
      A1: in STD_LOGIC;
      A2: in STD_LOGIC;
      A3: in STD_LOGIC;
      S0: in STD_LOGIC;
      S1: in STD_LOGIC;
      Y: out STD_LOGIC);
end MUX_4_to_1;
```



Λύση 2: Υλοποίηση με πολυπλέκτες 2 σε 1 σε δομή δένδρου



VHDL – Multiplexer

Architecture (4to1)

Η αρχιτεκτονική του πολυπλέκτη 4 σε 1 στη VHDL
Περιγραφή συμπεριφοράς – Λύση 2

```
architecture BEHAVIORAL of MUX_4_to_1 is
begin
  process (A0, A1, A2, A3, S0, S1)
  begin
    if (S1 = '0') then
      if (S0 = '0') then Y <= A0;
      else Y <= A1;
      end if;
    else
      if (S0 = '0') then Y <= A2;
      else Y <= A3;
      end if;
    end if;
  end process;
end BEHAVIORAL;
```

Στη λίστα ευαισθησίας συμπεριλαμβάνονται όλες οι είσοδοι του συνδυαστικού κυκλώματος

VHDL

Παράδειγμα Process

VHDL - Περίληψη

- Selected Assignment
- Conditional Statements
- Διαβάζετε τις παραγράφους 2.4, 3.2 (θεωρία και VHDL) από Ashenden και 2.7, 2.8, 4.2.4, 4.2.6, 4.3, 4.5.1, 4.5.2, 4.5.3 (ΟΧΙ το κομμάτι της VERILOG) από το βιβλίο των Harris.