*PROJECT 5TH* *ΚΑΡΑΒΙΑ ΜΑΡΙΑ ΣΠΥΡΙΔΟΥΛΑ (2026202100049)*   
*FSM ELEVATOR* *ΣΑΜΑΚΟΒΛΗ ΕΥΦΡΟΣΥΝΗ (2026202100142)*

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

--Ορισμός Εισόδων και Εξόδων

entity Elevetor\_C is

Port ( clk : in STD\_LOGIC;

rst : in STD\_LOGIC;

r\_up : in STD\_LOGIC\_VECTOR(9 downto 0);

r\_down : in STD\_LOGIC\_VECTOR(9 downto 0);

floor\_s : in STD\_LOGIC\_VECTOR(9 downto 0);

motor\_c : out STD\_LOGIC\_VECTOR (3 downto 0);

door\_c : out STD\_LOGIC;

current\_floor : out STD\_LOGIC\_VECTOR(3 downto 0));

end Elevetor\_C;

--Aρχιτεκτονικη Συμπεριφορας

architecture Behavioral of Elevetor\_C is

type Elevator\_C is (Idle, MovingUp, MovingDown, OpenDoor, CloseDoor, Waiting,OpeningDoor,ClosingDoor);

--Δηλωσεις ενδιαμεσων σηματων

signal current\_state: Elevetor\_C;

signal next\_state : Elevator\_C;

signal door\_state : Idle ;

signal next\_door\_state : Elevator\_C;

signal r\_floors : STD\_LOGIC\_VECTOR(9 downto 0);

signal t\_floor : integer range 0 to 9;

begin

--Εναρξης Διεργασιας

process (clk, rst)

begin

if rst = '1' then

current\_state <= Idle;

r\_floors <= (others => '0');

t\_floor <= 0;

elsif rising\_edge(clk) then

current\_state <= next\_state;

end if;

end process;

process (current\_state, r\_up, r\_down, floor\_s)

begin

--Περιπτωσεις Καταστασης Ανελκυστηρα(Λογικη Συμπεριφορα)

case current\_state is

when Idle =>

if r\_up /= (others => '0') or r\_down /= (others => '0') then

t\_floor <= t\_floor;

next\_state <= MovingUp;

else

next\_state <= Idle;

end if;

when MovingUp =>

if floor\_s = "t\_floor" then

next\_state<=OpenDoor;

elsif r\_up /= (others => '0') then

next\_state<= MovingUp;

else

next\_state<=Waiting;

current\_state <= MovingUp;

end if;

when MovingDown=>

if floor\_s = "t\_floor" then

next\_state<=OpenDoor;

elsif r\_down /= (others => '0') then

current\_state<=MovingDown;

next\_state<= OpenDoor;

else

next\_state<=Waiting;

end if;

end case;

end process;

process (door\_state,next\_door\_state,current\_state,OpenDoor, CloseDoor,Waiting,OpeningDoor,ClosingDoor)

begin

--Καταστασεις Πορτας

case door\_state is

when OpeningDoor =>

if door\_state = OpenDoor then

next\_door\_state <= OpenDoor;

elsif door\_state = ClosingDoor then

next\_door\_state <= ClosingDoor;

else

next\_door\_state <= Waiting;

end if;

when ClosingDoor =>

if door\_state = CloseDoor then

next\_door\_state <= CloseDoor;

elsif door\_state = OpeningDoor then

next\_door\_state <= OpeningDoor;

else

next\_door\_state <= Waiting;

end if;

when others =>

next\_door\_state <= Waiting;

end case;

end process;

--Ελέγχου Κινητήρα

motor\_c <= "01" when current\_state = MovingUp else

"10" when current\_state = MovingDown else

"00" when current\_state = Idle ;

--Ελέγχου Θυρών

door\_c <= '1' when current\_state = OpenDoor else '0';

motor\_c <= motor\_c ;

door\_c <= door\_c;

current\_floor <= (others => '0');

end Behavioral;

*--TEST BENCH*

library IEEE;

use IEEE.STD\_LOGIC\_1164.ALL;

--ΔΗΛΛΩΣΗ ΟΝΤΟΤΗΤΑΣ

entity Elevetor\_c\_tb is

end Elevetor\_c\_tb;

--ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΣΥΜΠΕΕΡΙΦΟΡΑΣ

architecture Behavioral of Elevetor\_c\_tb is

--ΕΝΔΙΑΜΕΣΑ ΛΟΓΙΚΑ ΣΗΜΑΤΑ

signal clk, rst : STD\_LOGIC := '0';

signal r\_up, r\_down, floor\_s : STD\_LOGIC\_VECTOR(9 downto 0) := (others => '0');

signal motor\_c, door\_c : STD\_LOGIC;

signal current\_floor : STD\_LOGIC\_VECTOR(3 downto 0);

--ΚΑΘΟΡΙΣΜΟΣ ΘΥΡΩΝ

component Elevator\_C

Port (

clk : in STD\_LOGIC;

rst : in STD\_LOGIC;

r\_up : in STD\_LOGIC\_VECTOR(9 downto 0);

r\_down : in STD\_LOGIC\_VECTOR(9 downto 0);

floor\_s : in STD\_LOGIC\_VECTOR(9 downto 0);

motor\_c : out STD\_LOGIC;

door\_c : out STD\_LOGIC;

current\_floor : out STD\_LOGIC\_VECTOR(3 downto 0)

);

end component;

--ΧΑΡΤΗΣ ΘΥΡΩΝ

begin

UUT: Elevator\_C

port map (

clk => clk,

rst => rst,

r\_up => r\_up,

r\_down => r\_down,

floor\_s => floor\_s,

motor\_c => motor\_c,

door\_c => door\_c,

current\_floor => current\_floor

);

--ΔΙΕΡΓΑΣΙΑ ΠΑΡΑΓΩΓΗΣ ΠΡΟΣΟΜΕΙΩΜΕΝΟΥ ΣΗΜΑΤΟΣ ΡΟΛΟΓΙΟΥ

CLK\_GEN: process

begin

while now < 1000 ns loop

clk <= '0';

wait for 5 ns;

clk <= '1';

wait for 5 ns;

end loop;

wait;

end process;

--ΔΙΕΡΓΑΣΙΑ ΓΙΑ ΤΗΝ ΔΟΚΙΜΗ ΤΟΥ ΑΝΕΛΚΥΣΤΗΡΑ

STIMULUS: process

begin

rst <= '1';

wait for 10 ns;

rst <= '0';

wait for 10 ns;

r\_up <= "0000000001";

wait for 100 ns;

r\_up <= (others => '0');

wait for 10 ns;

r\_down <= "0000000001";

wait for 100 ns;

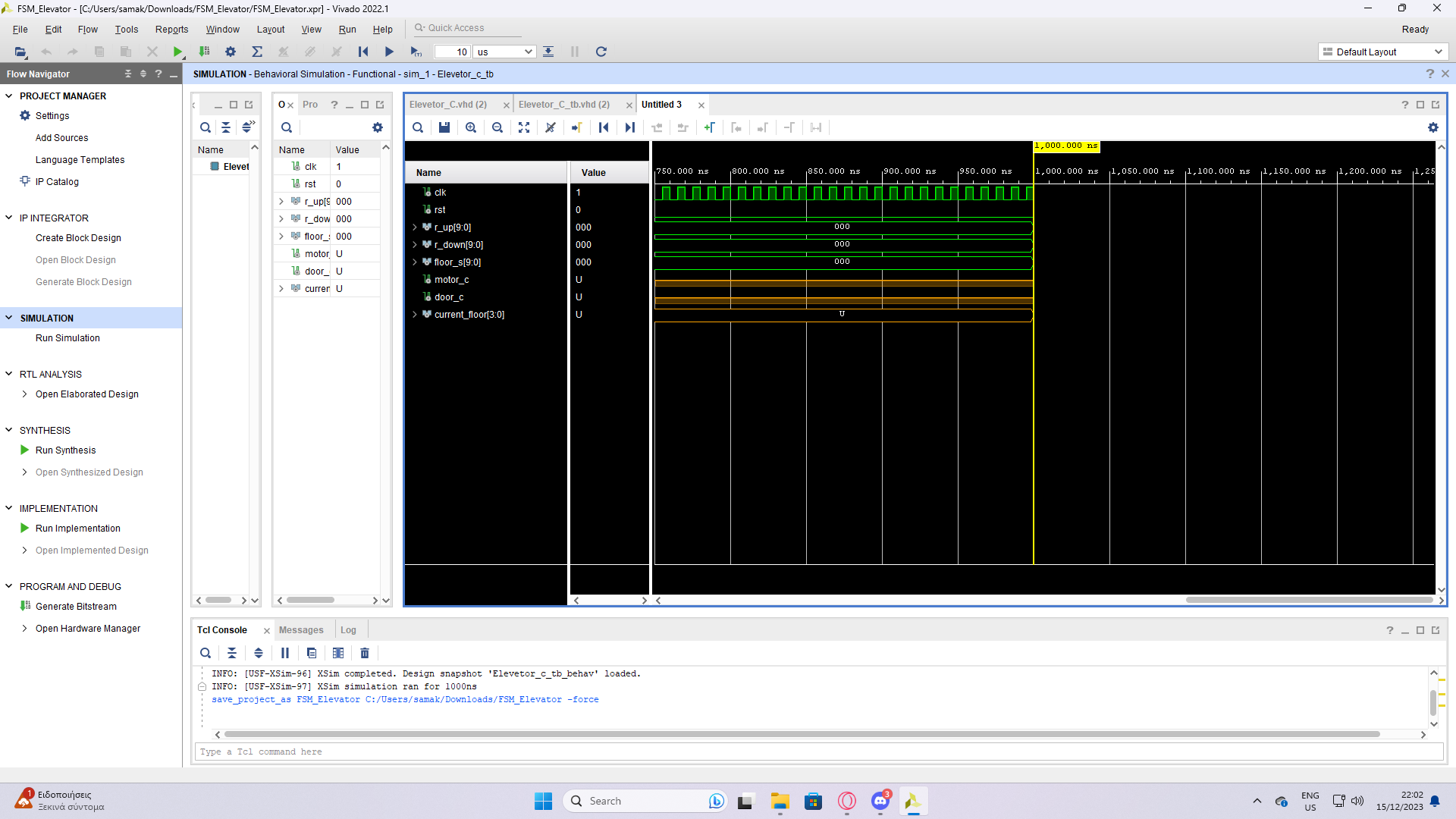
r\_down <= (others => '0');

wait for 10 ns;

wait;

end process;

end Behavioral;



ΑΝΑΛΥΣΗ ΤΗΣ ΣΧΕΔΙΑΣΗΣ

1. Δηλώνουμε τις εισόδους και τις εξόδους της οντότητας
2. Στην αρχιτεκτονική συμπεριφοράς:
3. Δημιουργούμε μια διεργασία για το ρολόι.
4. Δημιουργούμε μια διεργασία την λογική συμπεριφορά του ανελκυστήρα ,δηλαδή πως συμπεριφέρεται εάν βρίσκεται σε αδρανή κατάσταση, όταν ανεβαίνει ή όταν κατεβαίνει.
5. Δημιουργούμε μια διεργασία για τις καταστάσεις της πόρτας
6. Για τον έλεγχο του κινητήρα, δίνουμε τιμές όπου η κάθε μια αντιστοιχεί σε διαφορετική κατάσταση του κινητήρα.
7. Τέλος, κάνουμε τον έλεγχο των θυρών.

