**Τμήμα Πληροφορικής**

**Μάθημα: Προηγμένες Εφαρμογές Ψηφιακής Σχεδίασης**

**Ονοματεπώνυμο : Ζαρκάδας Φώτης**

**ΑΕΜ: 5486**

**Εξάμηνο: 9ο**

**ΑΠΑΝΤΗΣΕΙΣ 1ης Σειράς Ασκήσεων**

**Μέρος 1.**

**Nor41:**

O κώδικας nor41.sv είναι:

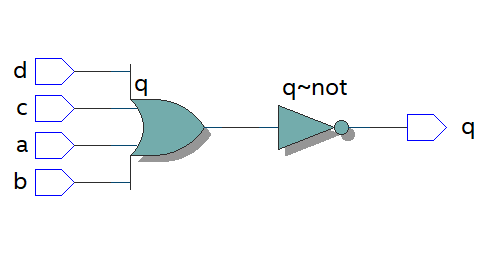
module nor41 (input logic a,b,c,d,

output logic q);

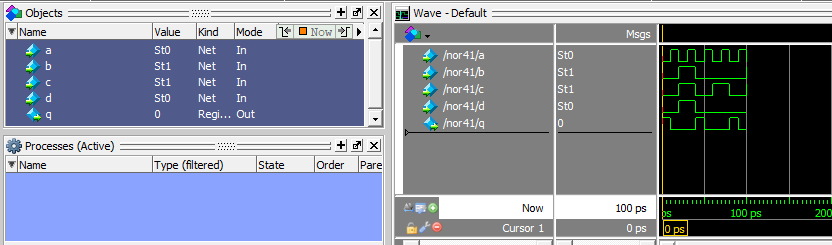
assign q = ~(a | b | c | d);

Endmodule

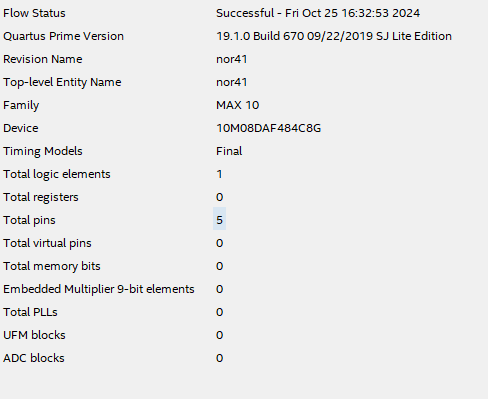
Η εικόνα απο το netlist viewer:



Η εικόνα παραθύρου προσομοίωσης ModelSIM:



Το compilation report:



**CompFlags:**

Ο κώδικας CompFlags.sv είναι:

module CompFlags(input logic C,N,V,Z,

output logic HS, LS, HI, LO, GE, LE, GT, LT);

logic not\_z;

assign HS = C;

not not1(not\_z,Z);

and andgate1(HI,not\_z,C);

not not2(LO,C);

or orgate1(LS,z,LO);

xnor xnor1(GE,N,V);

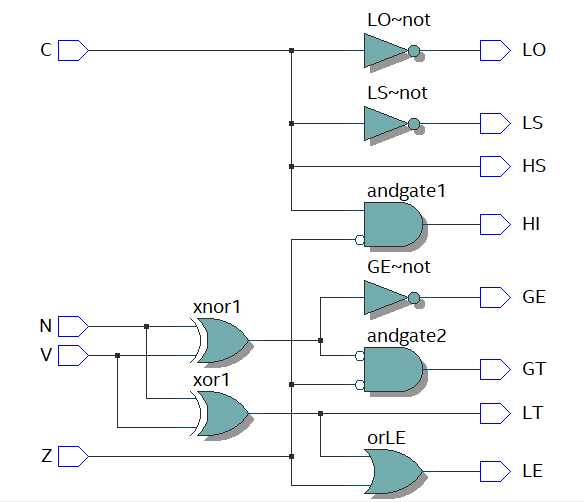
xor xor1(LT,N,V);

or orLE(LE,Z,LT);

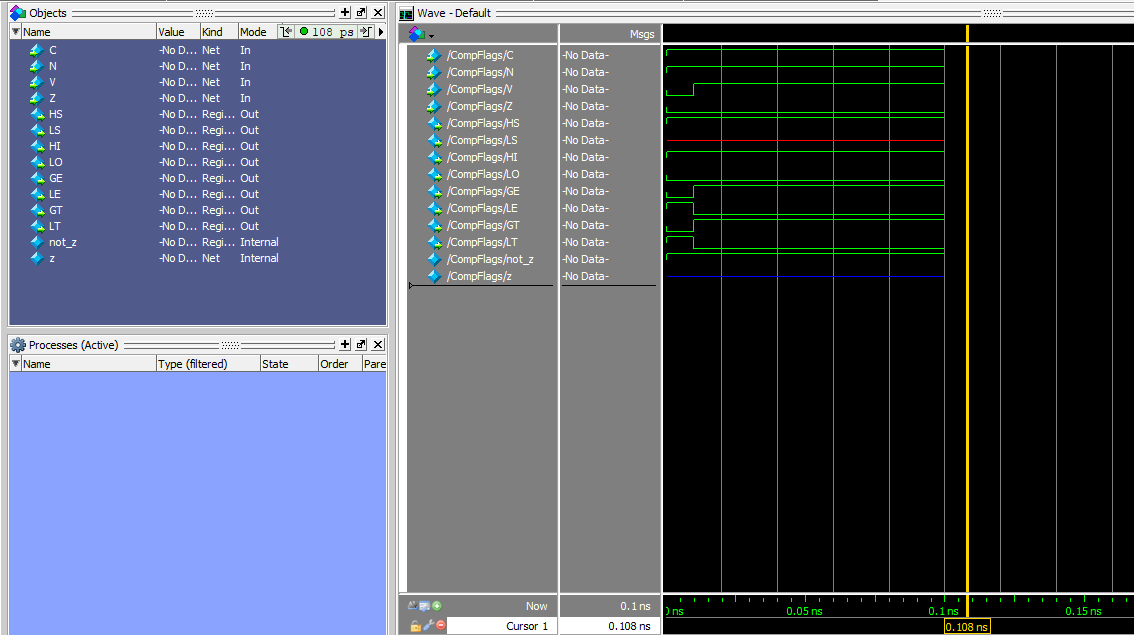
and andgate2(GT,not\_z,GE);

Endmodule

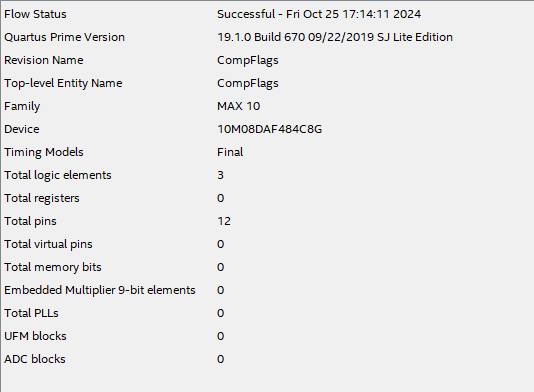
Η εικόνα απο το netlist viewer:



Η εικόνα παραθύρου προσομοίωσης ModelSIM:



Το compilation report:



**Μέρος 2.**

Ο κώδικας mux2.sv είναι:

module mux2(input logic x,y,

input logic s,

output logic m);

assign m = s ? x : y;

endmodule

Ο κώδικας mux28bit.sv είναι:

module mux28bit(input logic [7:0] x,y,

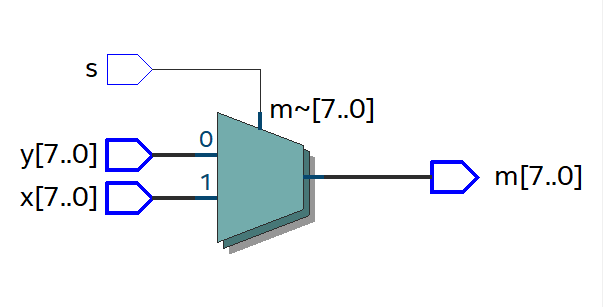
input logic s,

output logic [7:0] m);

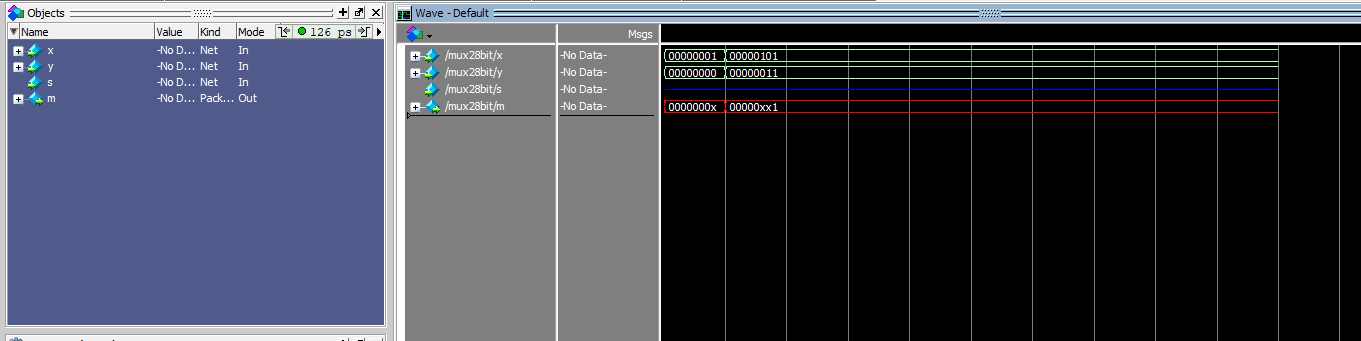
assign m = s ? x : y;

endmodule

Η εικόνα απο το netlist viewer:



Η εικόνα παραθύρου προσομοίωσης ModelSIM:



Το compilation report:



**Μέρος 3.**

**Ο κώδικας mux23.sv είναι:**

module mux23 (input logic [2:0] x, y,

input logic s,

output logic [2:0] m);

assign m= s ? y : x;

Endmodule

**Ο κώδικας mux53.sv είναι:**

module mux53 (input logic [2:0] u, v, w, x, y,

input logic [2:0] s,

output logic [2:0] m);

logic [0:2] m\_out1, m\_out2, m\_out3;

mux23 multi1(u,v,s[0],m\_out1);

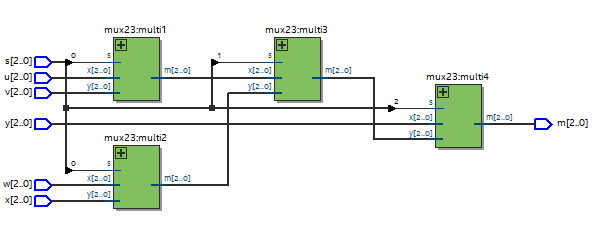
mux23 multi2(w,x,s[0],m\_out2);

mux23 multi3(m\_out1, m\_out2, s[1], m\_out3);

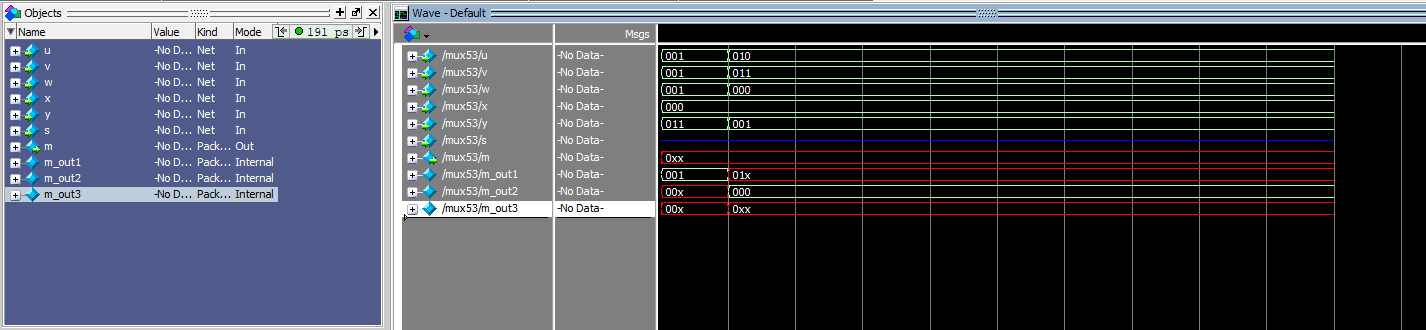
mux23 multi4(y, m\_out3, s[2], m);

Endmodule

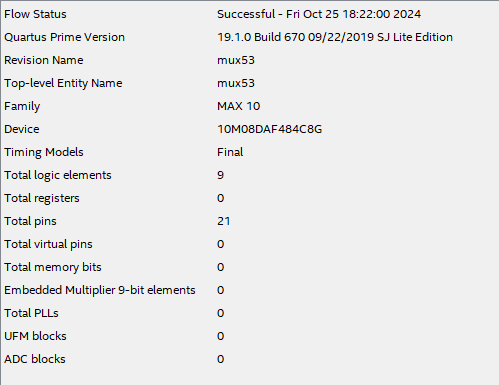
Η εικόνα απο το netlist viewer:



Η εικόνα παραθύρου προσομοίωσης ModelSIM:



Το compilation report:



**Μέρος 4.**

**Ο κώδικας mux53beh.sv είναι:**

module mux53beh(input logic[2:0] u,v,w,x,y,s,

output logic[2:0]m);

always\_comb

if (s==0) m = u;

else if (s==1) m = v;

else if (s==2) m = w;

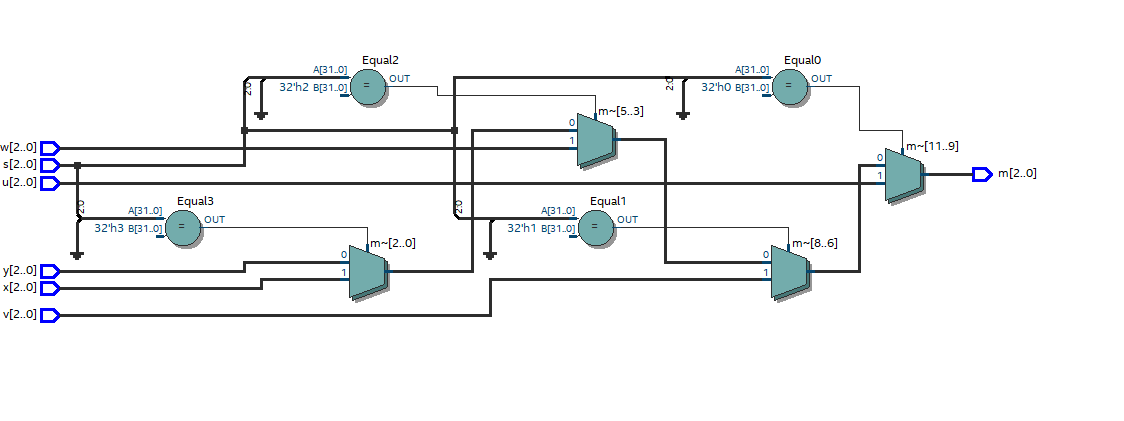
else if (s==3) m = x;

else if (s==4) m = y;

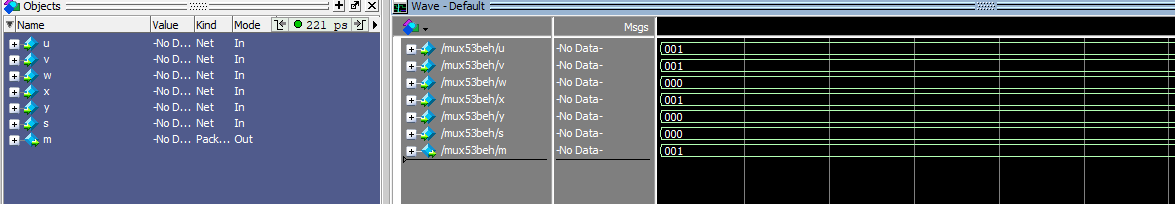
else m = y;

Endmodule

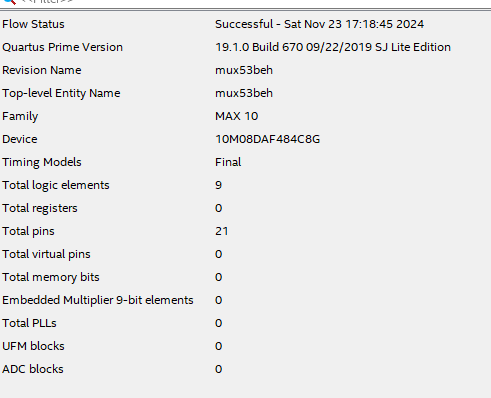
Η εικόνα απο το netlist viewer:



Η εικόνα παραθύρου προσομοίωσης ModelSIM:



Το compilation report:



**Μέρος 5.**

**Ο κώδικας prm.sv είναι:**

module prm(input logic D,A,E,F,

output logic [3:0]y);

always\_comb begin

y = 4'b0000;

if (D)

y = 4'b1000;

else if (A)

y = 4'b0100;

else if (E)

y = 4'b0010;

else if (F)

y= 4'b0001;

end

endmodule

**Ο κώδικας sevensegprm.sv είναι:**

module sevensegprm ( input logic D,A,E,F,

output logic a,b,c,d,e,f,g);

logic [3:0]y;

logic [6:0]seg;

prm prm1(D,A,E,F,y);

always\_comb begin

seg = 7'b111\_1111;

case (y)

4'b0001: seg = 7'b100\_0111; //F

4'b0010: seg = 7'b100\_1111; //E

4'b0100: seg = 7'b111\_0111; //A

4'b1000: seg = 7'b011\_0111; //d

default: seg = 7'b000\_0000; //reset

endcase

a = seg[0];

b = seg[1];

c = seg[2];

d = seg[3];

e = seg[4];

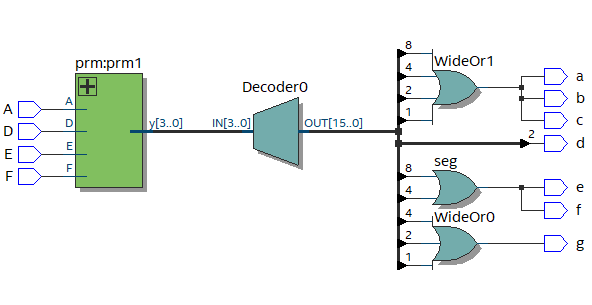
f = seg[5];

g = seg[6];

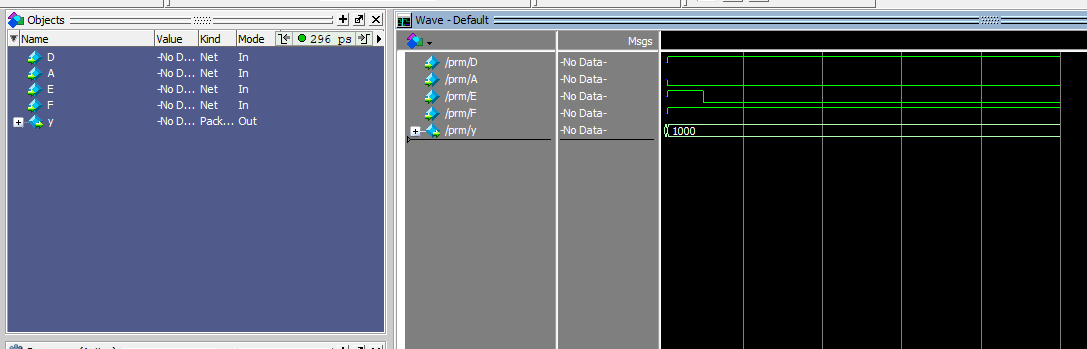
end

endmodule

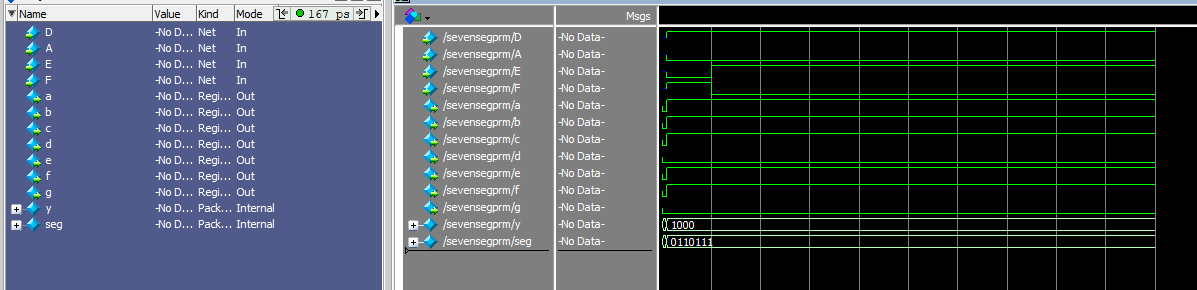
Η εικόνα απο το netlist viewer:



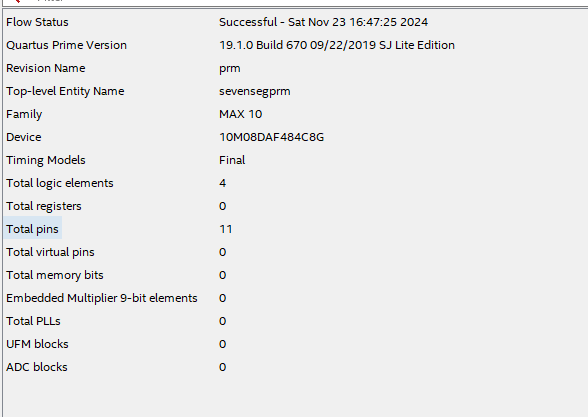
Η εικόνα παραθύρου προσομοίωσης ModelSIM prm.sv:



εικόνα παραθύρου προσομοίωσης ModelSIM sevensegprm.sv:



Το compilation report:



**Μέρος 6.**

**Ο κώδικας fa1bit.sv είναι:**

module fa1bit(input logic a, b, cin,

output logic s, cout);

logic p, g; // Εσωτερικοί κόμβοι 1-bit

assign p = a ^ b;

assign g = a & b;

assign s = p ^ cin;

assign cout = g | (p & cin);

endmodule

**Ο κώδικας farc4bit.sv είναι:**

module farc4bit( input logic [3:0]a,

input logic [3:0]b,

input logic cin,

output logic [3:0] sum,

output logic cout);

logic [3:0] carry;

fa1bit f0 (

.a (a[0]),

.b(b[0]),

.cin(cin),

.s (sum[0]),

.cout(carry[0]));

fa1bit f1 (

.a (a[1]),

.b(b[1]),

.cin(carry[0]),

.s (sum[1]),

.cout(carry[1]));

fa1bit f2 (

.a (a[2]),

.b(b[2]),

.cin(carry[1]),

.s (sum[2]),

.cout(carry[2]));

fa1bit f3 (

.a (a[3]),

.b(b[3]),

.cin(carry[2]),

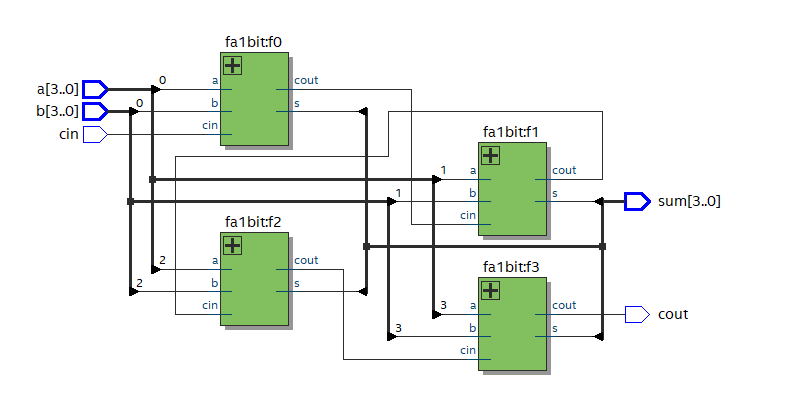
.s (sum[3]),

.cout(carry[3]));

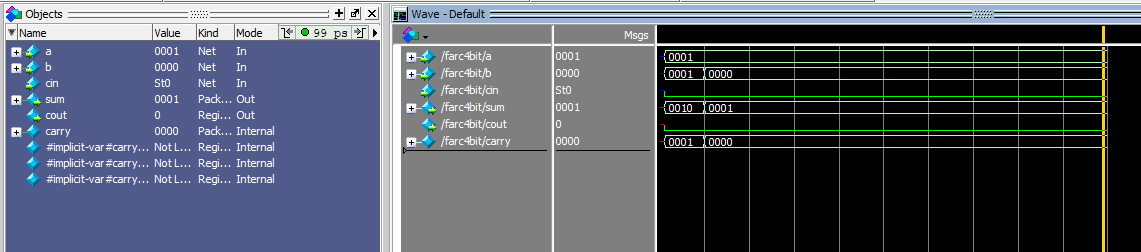
assign cout = carry[3];

endmodule

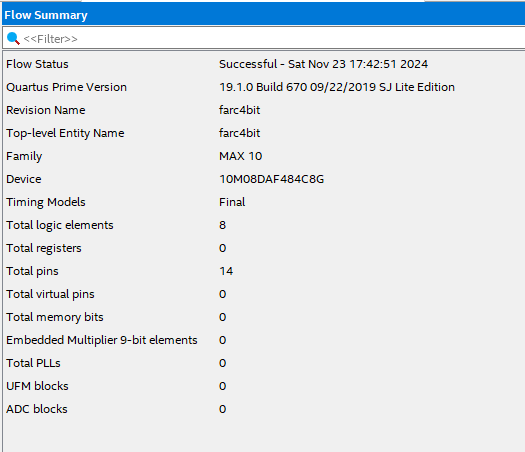
Η εικόνα απο το netlist viewer:

****

Η εικόνα παραθύρου προσομοίωσης ModelSIM

****

Το compilation report:

****

**Μέρος 7**

**Ο κώδικας alunbit.sv είναι:**

module alunbit #(parameter n = 8)

(input logic [n-1:0] a,b,

input logic ALUControl,

output logic [n-1:0] Result,

output logic N,Z,C,V );

logic [n-1:0] condinvb;

logic [n:0] sum;

logic [3:0]ALUFlags;

assign condinvb = ALUControl ? ~b : b;

assign sum = a + condinvb + ALUControl;

always\_comb

casex (ALUControl)

2'b0?: Result = sum;

2'b10: Result = a & b;

2'b11: Result = a | b;

default: Result = 'bx;

endcase

always\_comb begin

ALUFlags[0] = (Result == 4'b0000);

ALUFlags[1] = Result[3];

ALUFlags[2] = Result[4];

if (ALUControl == 1'b0) begin

// Στην πρόσθεση, υπάρχει overflow αν τα πιο σημαντικά bits του a και b είναι ίδια και το αποτέλεσμα έχει αντίθετο πρόσημο

ALUFlags[3] = (a[n-1] == b[n-1]) && (Result[n-1] != a[n-1]);

end else begin

// Στην αφαίρεση, υπάρχει overflow αν τα πιο σημαντικά bits του a και b είναι διαφορετικά και το αποτέλεσμα έχει το ίδιο πρόσημο με το a

ALUFlags[3] = (a[n-1] != b[n-1]) && (Result[n-1] == a[n-1]);

end

end

assign Z = ALUFlags[0];

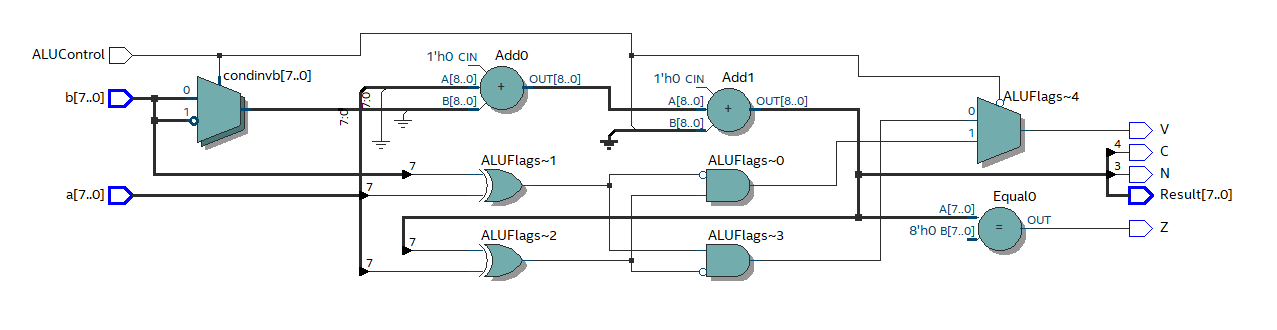
assign N = ALUFlags[1];

assign C = ALUFlags[2];

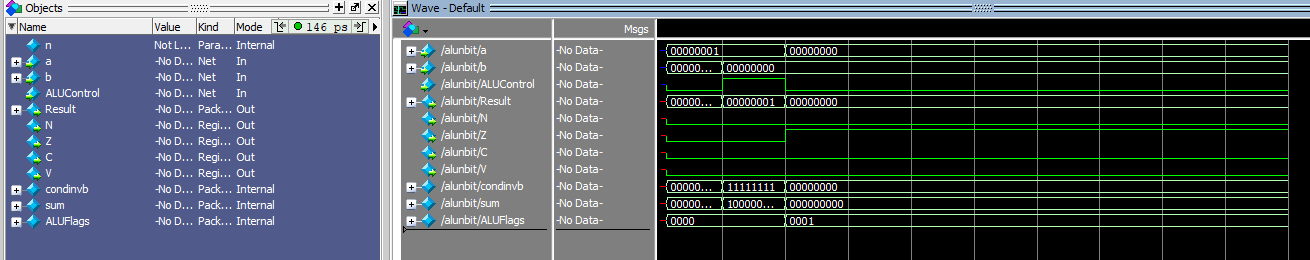
assign V = ALUFlags[3];

endmodule

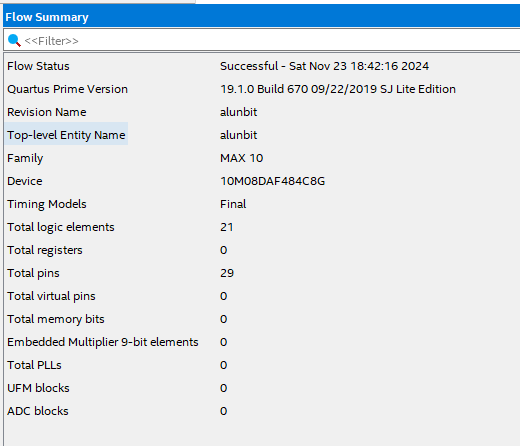
Η εικόνα απο το netlist viewer:



Η εικόνα παραθύρου προσομοίωσης ModelSIM



Το compilation report:



**Μέρος 8**

**Ο κώδικας Barrel8bit.sv είναι:**

module Barrel8bit(input logic[7:0] a,input logic[2:0] amt,

input logic[1:0]oppbarrel,output logic[7:0] Result);

logic[7:0] LSL,LSR,ASR,ROR;

always\_comb begin

LSL = a << amt;

LSR = a >> amt;

case(amt)

3'b000: ASR = a;

3'b001: ASR = {a[7], a[7], a[6], a[5], a[4], a[3], a[2], a[1]};

3'b010: ASR = {a[7], a[7], a[7], a[6], a[5], a[4], a[3], a[2]};

3'b011: ASR = {a[7], a[7], a[7], a[7], a[6], a[5], a[4], a[3]};

3'b100: ASR = {a[7], a[7], a[7], a[7], a[7], a[6], a[5], a[4]};

3'b101: ASR = {a[7], a[7], a[7], a[7], a[7], a[7], a[6], a[5]};

3'b110: ASR = {a[7], a[7], a[7], a[7], a[7], a[7], a[7], a[6]};

3'b111: ASR = {a[7], a[7], a[7], a[7], a[7], a[7], a[7], a[7]};

default: ASR = 'bx;

endcase

ROR = (a >> amt) | (a << 8 - amt);

case(oppbarrel)

2'b00: Result=LSR;

2'b01: Result=LSL;

2'b10: Result=ROR;

2'b11: Result=ASR;

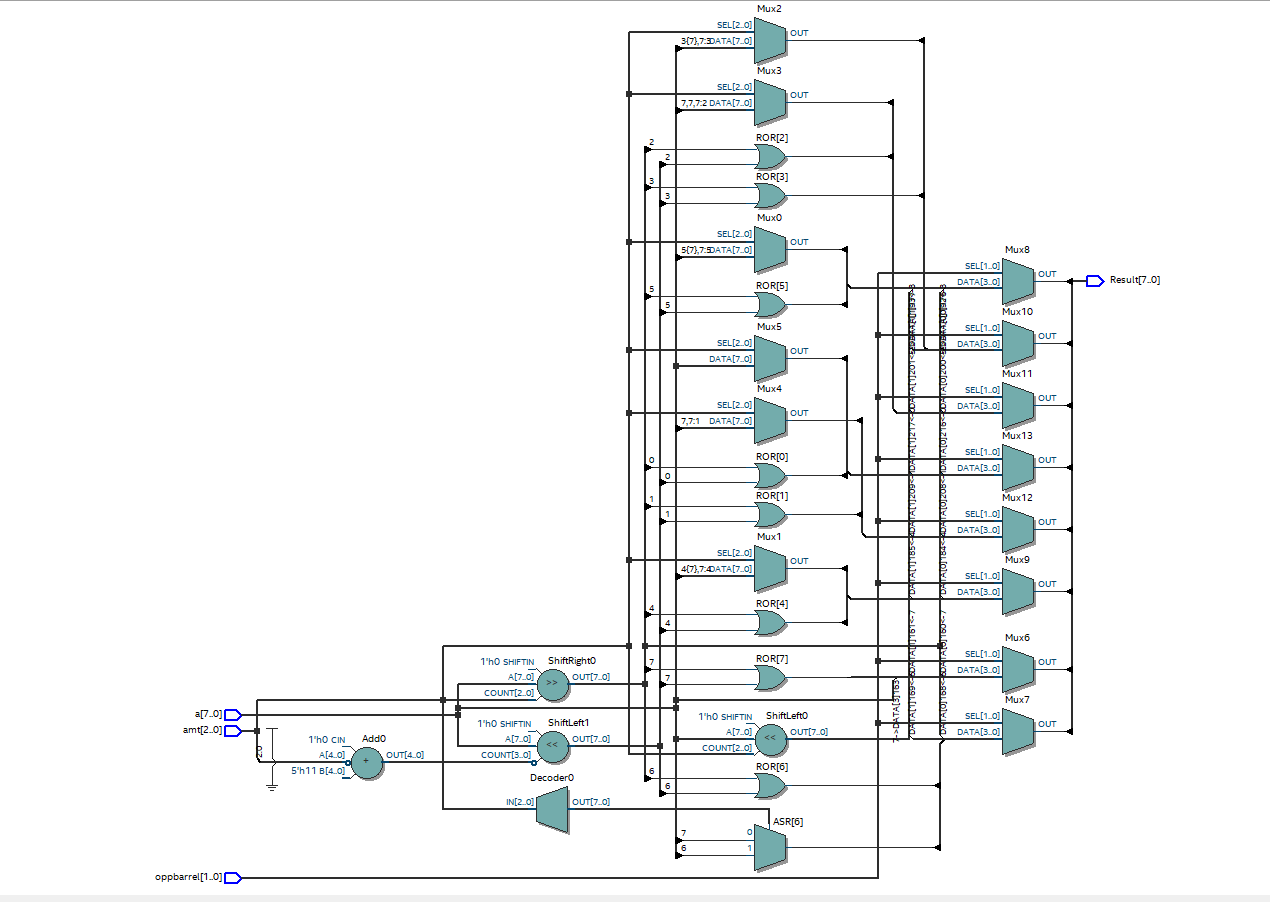
default Result='bx;

endcase

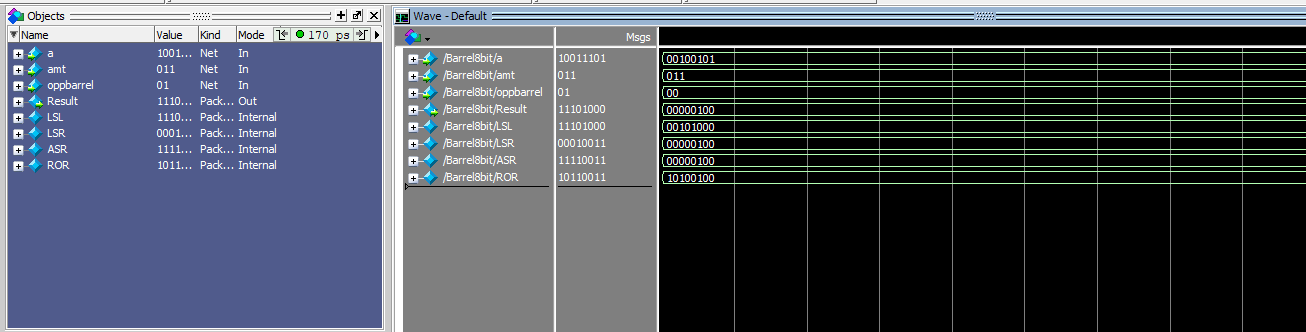
end

endmodule

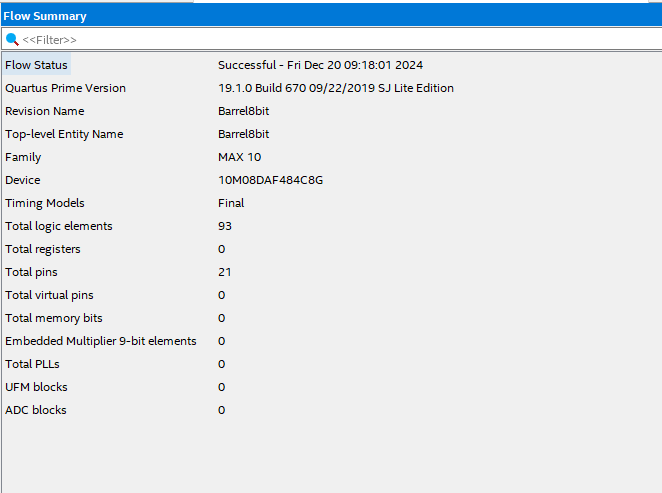
Η εικόνα απο το netlist viewer:



Η εικόνα παραθύρου προσομοίωσης ModelSIM



Το compilation report:



**Μέρος 9**

**Ο κώδικας ALUBarrel.sv είναι:**

module ALUBarrel #(parameter Num=8)

(input logic[Num-1:0] Rn,Rm,input logic[2:0] amt,

input logic[1:0] opbarrel, ALUControl,

output logic[Num-1:0] Rd,output logic N, Z, V, C, HS, LS, HI, LO, GE, LE, GT, LT);

logic[3:0] ALUFlags;

logic[7:0] BarrelOut;

Barrel8bit barrel(Rm[Num-1:0],amt[2:0],opbarrel[1:0],BarrelOut[7:0]);

alunbit #(Num) ALU(Rn[Num-1:0],BarrelOut[7:0],ALUControl[1:0],Rd[Num-1:0],ALUFlags[3:0]);

CompFlags Flags(ALUFlags[0], ALUFlags[1], ALUFlags[2],

ALUFlags[3], HS, LS, HI, LO, GE, LE, GT, LT);

assign C = ALUFlags[0];

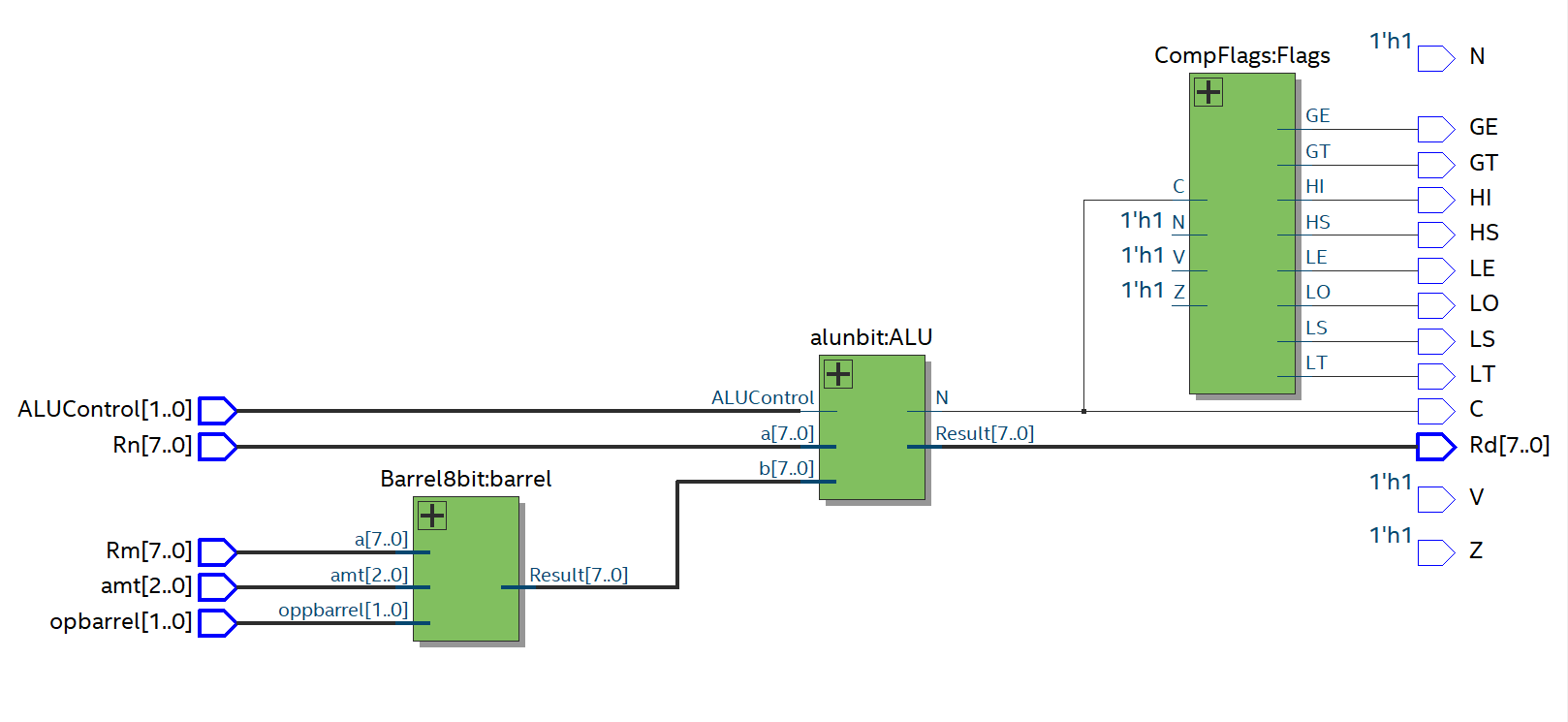
assign N = ALUFlags[1];

assign V = ALUFlags[2];

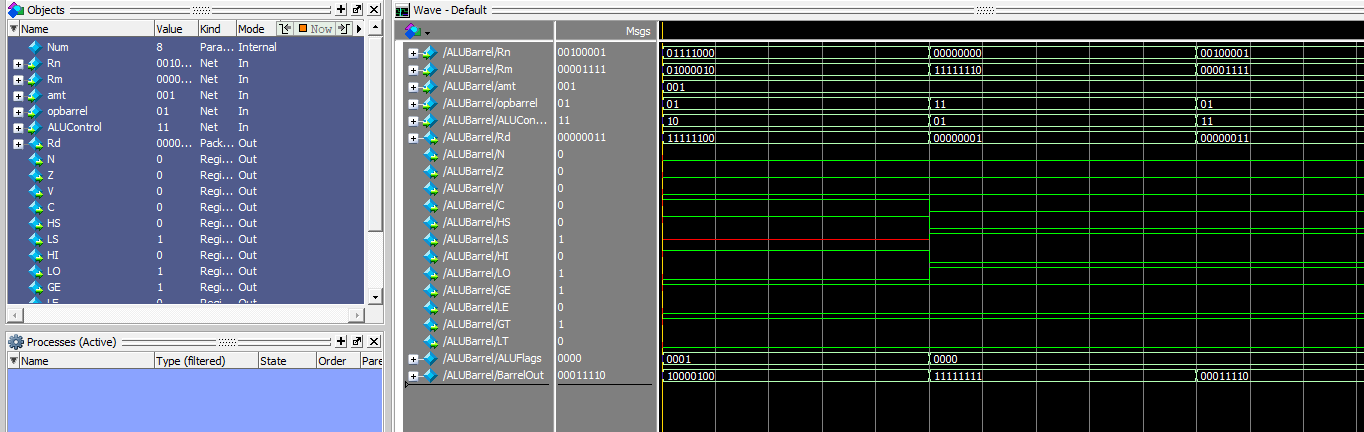
assign Z = ALUFlags[3];

endmodule

Η εικόνα απο το netlist viewer:



Η εικόνα παραθύρου προσομοίωσης ModelSIM



Το compilation report:

