9: Modelsim Tutorial

Vasileios Tenentes

University of Ioannina

Outline

1. Πως θα προμηθευτούμε το λογισμικό
2. Εγκατάσταση Quartus
3. Εκτέλεση Modelsim
4. Εκτέλεση του Modelsim
5. Παράδειγμα Νέου Project στο Modelsim
6. Παράδειγμα Νέου Project στο Modelsim
7. Παράδειγμα Compilation
8. Κονσόλα του Modelsim
9. Παράδειγμα Προσομοίωσης
10. Τερματισμός Προσομοίωσης

Πως θα προμηθευτούμε το λογισμικό

Το Modelsim για τις ασκήσεις Verilog που θα ακολουθήσουν είναι μέρος του Quartus.

Το Quartus μπορείτε να το κατεβάσετε απευθείας από των Altera/Intel εδώ
https://fpgasoftware.intel.com/13.0sp1/?edition=web

Πρέπει να εγγραφείτε για να το κατεβάσετε.
Προσοχή θέλουμε μόνο την έκδοση 13.0sp1.

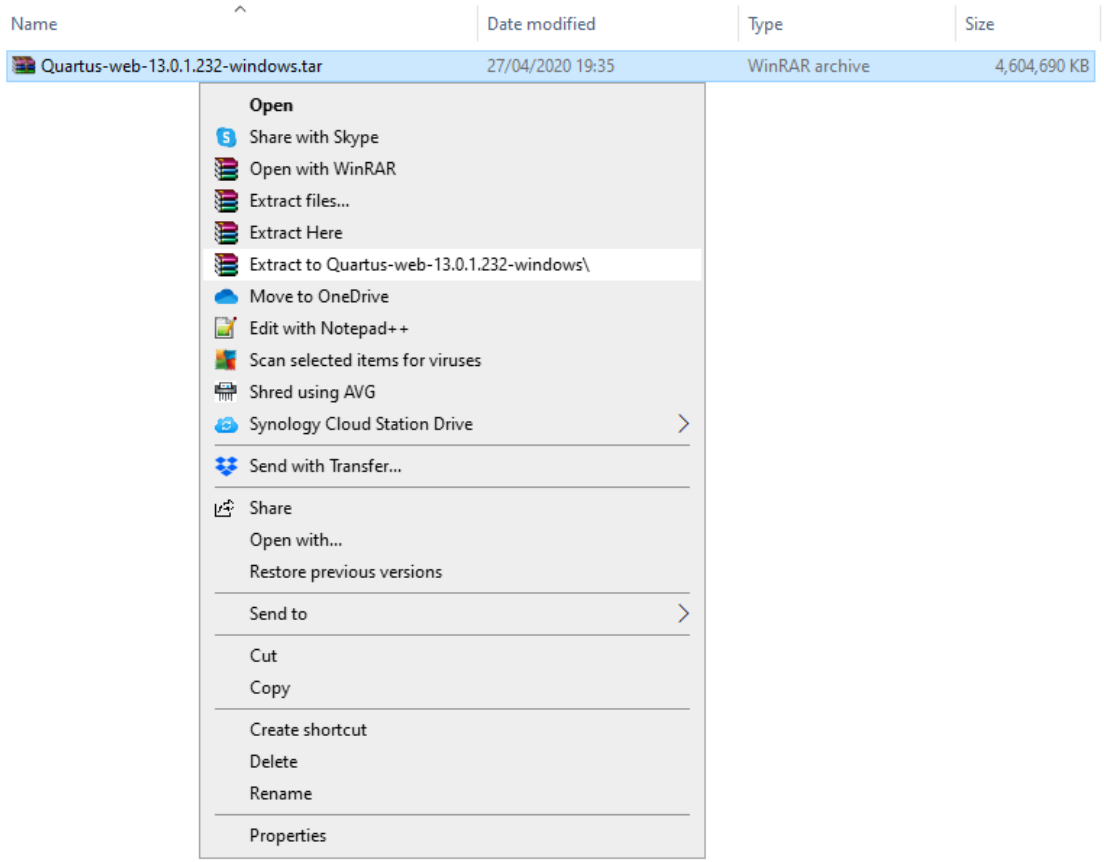
Το λογισμικό τρέχει σε windows και linux.
ΕΝΝΑΛΑΚΤΙΚΑ - **το έχω ήδη κατεβάσει** και θα το βρείτε στα παρακάτω links:
Για Windows:
http://vcas.cs.uoi.gr/labsoftware/Quartus-web-13.0.1.232-windows.tar
Για Linux:
http://vcas.cs.uoi.gr/labsoftware/Quartus-web-13.0.1.232-linux.tar
Παρακαλώ προχωρήστε στην εγκατάσταση για να είναι έτοιμο για το μάθημα, επειδή είναι μεγάλο το αρχείο.
Το λογισμικό υπάρχει ήδη στα εργαστήρια εγκατεστημένο, επομένως αν μπορείτε να συνδεθείτε απομακρυσμένα τότε μπορείτε να κάνετε και τις εργασίας με απομακρυσμένη σύνδεση, αν δεν θέλετε να το εγκαταστήσετε.

Ακολουθεί ένα βασικό tutorial για να ξεκινήσετε με το Modelsim.

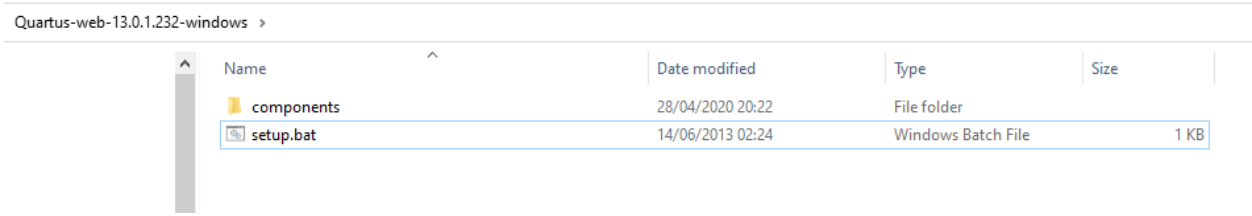
Εγκατάσταση Quartus

Εγκατάσταση

1. Κατεβάζουμε το Quartus και αποσυμπιέζουμε το αρχείο που κατεβάσαμε. Μπορείτε να το κάνετε με το winrar, όπως φαίνεται στην παρακάτω εικόνα:



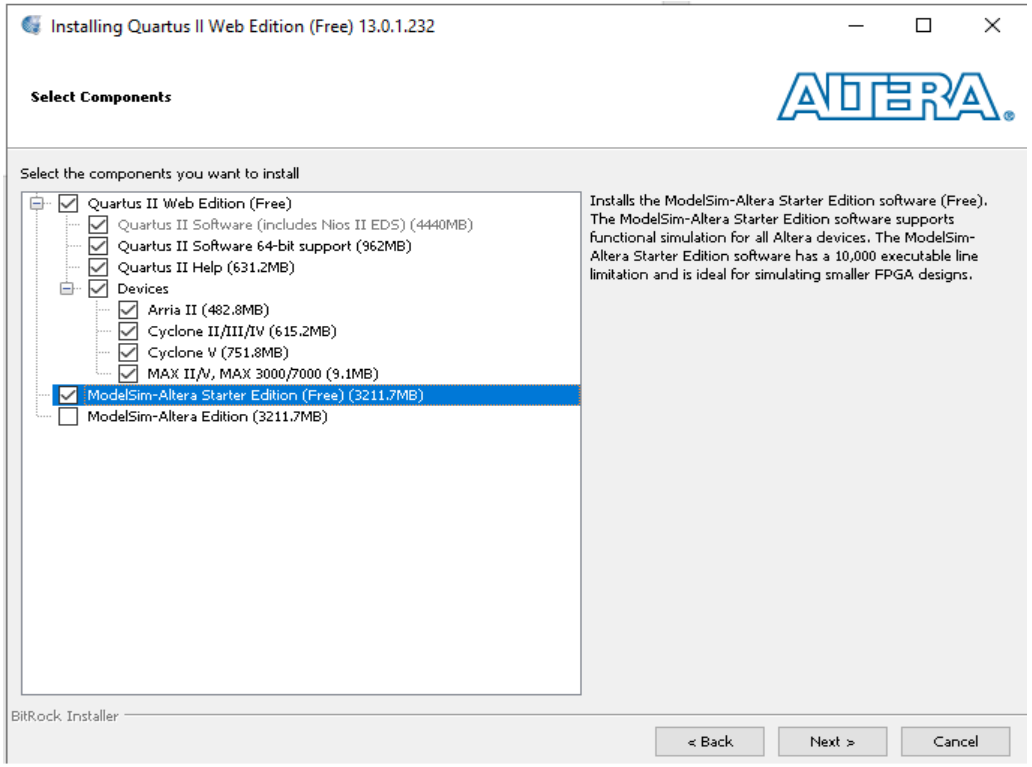
2. Κατεβάζουμε το Quartus και αποσυμπιέζουμε το αρχείο που κατεβάσαμε. Μπορείτε να το κάνετε με το winrar, όπως φαίνεται στην παρακάτω εικόνα:



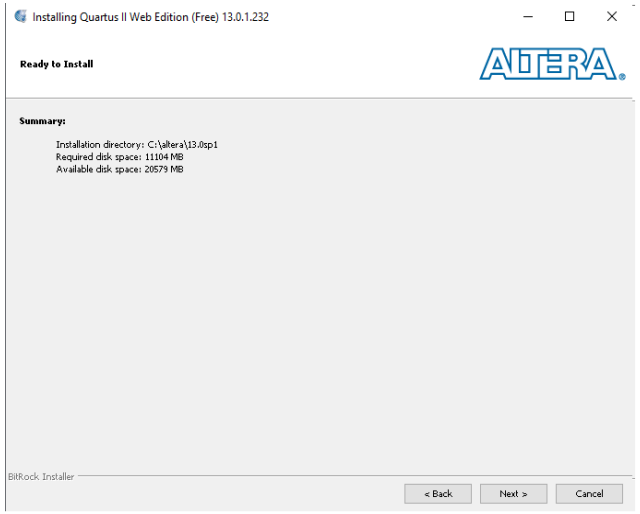
Επιλέγουμε που θα γίνει η εγκατάσταση και προχωράμε στην επιλογή πακέτων.

Εγκατάσταση

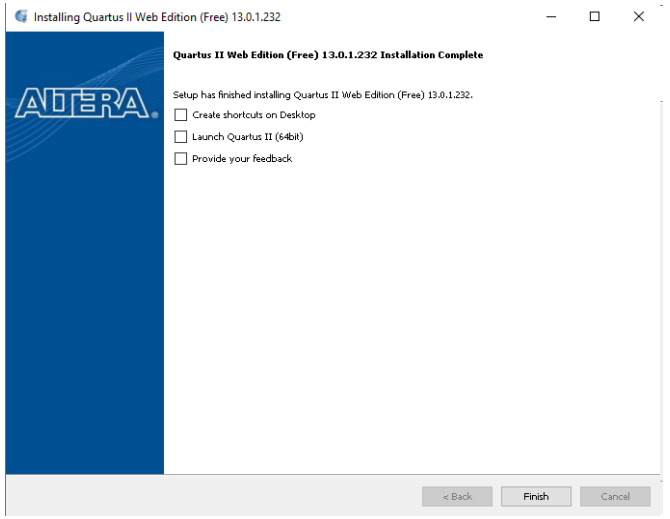
3. Θέλουμε **το Modelsim Altera Started Edition (Free),** που είναι προεπιλεγμένο. Ωστόσο βάλτε και το Quartus που είναι και αυτό προεπιλεγμένο. ΜΗΝ επιλέξετε την άλλη έκδοση modelsim που έχει κάτω-κάτω γιατί δεν είναι free και θα σας ζητάει άδειες.



4. Ξεκινάμε την εγκατάσταση με τις επιλογές μας. Θα πάρει αρκετή ώρα, επομένως περιμένουμε.



5. Μόλις τελειώσει η εγκατάσταση θα μας βγάλει το παρακάτω παράθυρο:

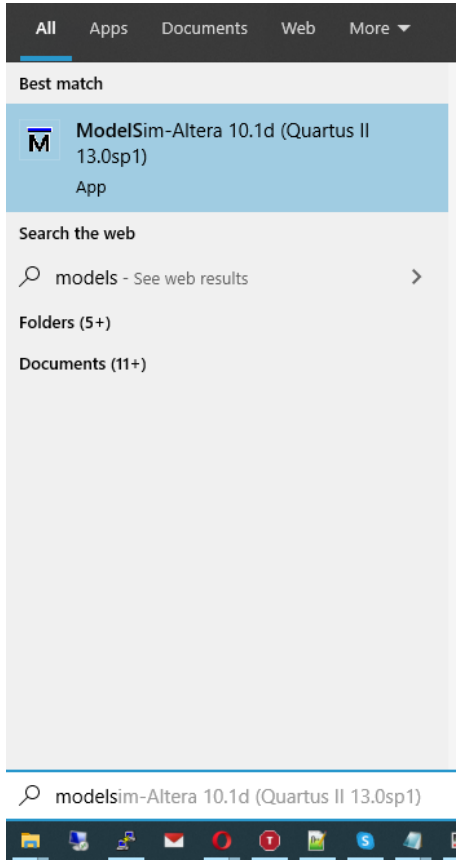


6. Απενεργοποιήστε τις επιλογές. Δηλαδή δεν θέλουμε μην φτιάξει shortcut ούτε θέλουμε να εκτελέσει το Quartus ούτε θέλουμε να δώσουμε feedback στην εταιρία. Αφού απενεργοποιήσουμε τις επιλογές πατάμε finish.
Συγχαρητήρια, η εγκατάσταση ολοκληρώθηκε και θα προχωρήσουμε στην εκτέλεση του προσομοιωτή Modelsim.

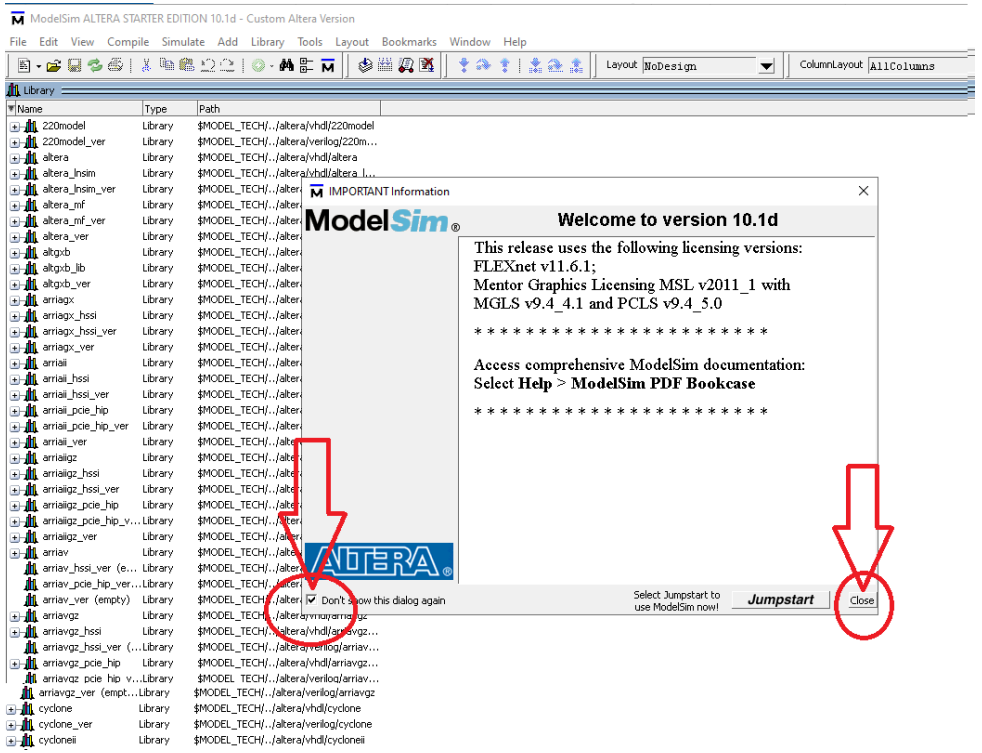
Εκτέλεση Modelsim

Εκτέλεση του Modelsim

Από το start στα Windows βρίσκουμε το “Modelsim-Altera 10.1d (Quartus II 13.0sp1)” και το εκτελούμε.



Θα πάρει λίγη ώρα να ξεκινήσει.
Μόλις ξεκινήσει θα μας βγάλει το παρακάτω παράθυρο:

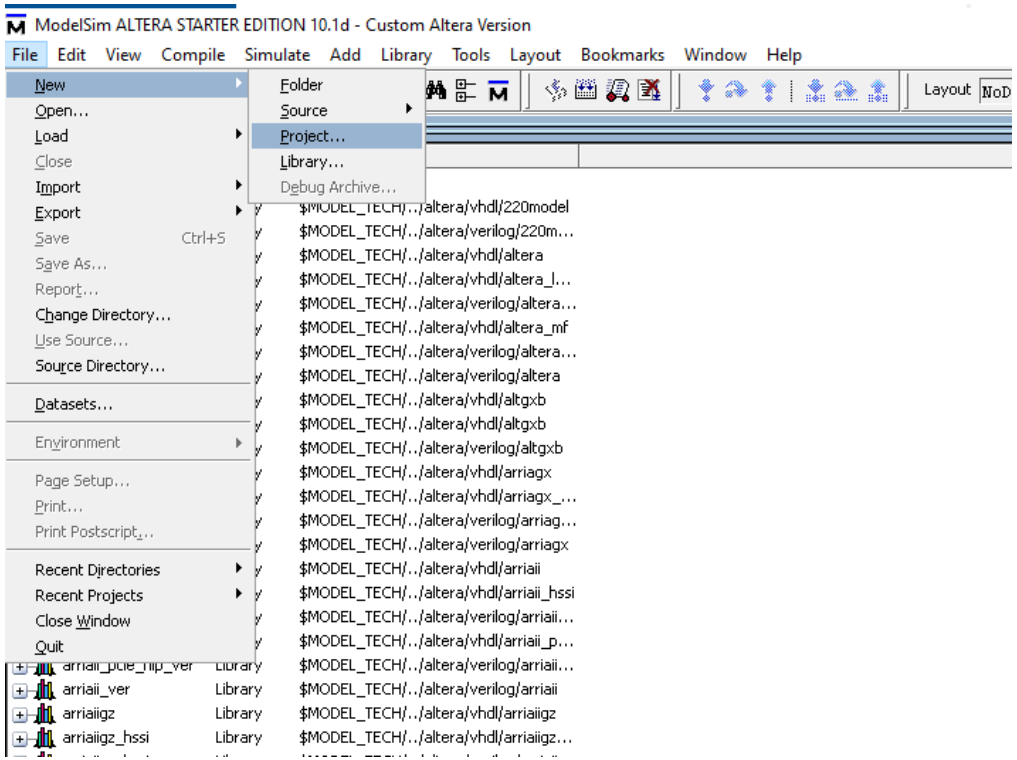


Επιλέγουμε “Don’t show this dialog again” και μετά πατάμε την επιλογή “close”.

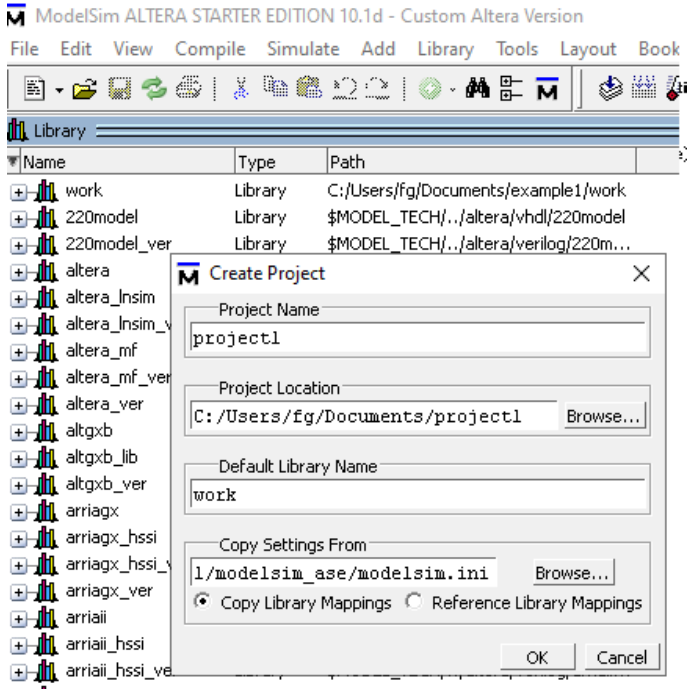
Παράδειγμα Νέου Project στο Modelsim

Παράδειγμα Νέου Project στο Modelsim

Όπως φαίνεται και από την παρακάτω εικόνα, φτιάχνουμε ένα νέο project με την επιλογή File->New->Project.



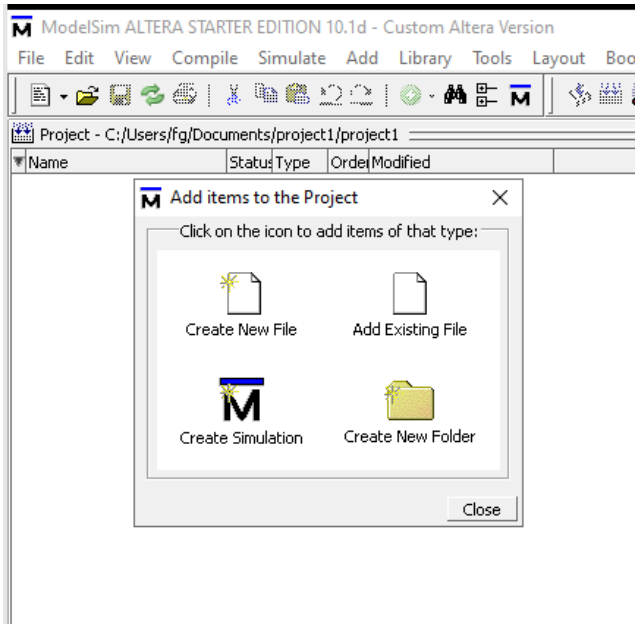
Έπειτα επιλέγουμε όνομα για το project και κατάλογο στον οποίο θα το αποθηκεύσουμε. Στο παράδειγμα αυτό έχω δώσει το όνομα project1 στον κατάλογο project1.



ΠΡΟΣΟΧΗ: το default library name πρέπει να είναι “work”.
Κάθε άσκηση που θα λύνουμε θα έχει το δικό της project.

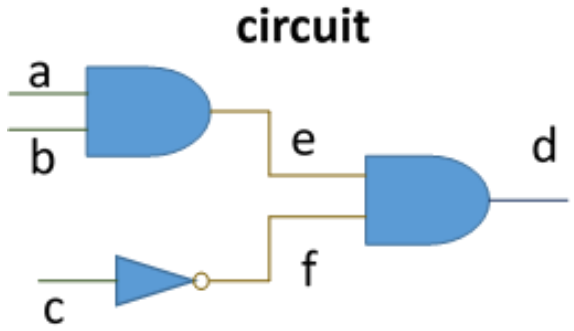
Παράδειγμα Νέου Project στο Modelsim

Στο επόμενο στάδιο πρέπει να επιλέξουμε αρχεία για το project μας με την παρακάτω επιλογή:



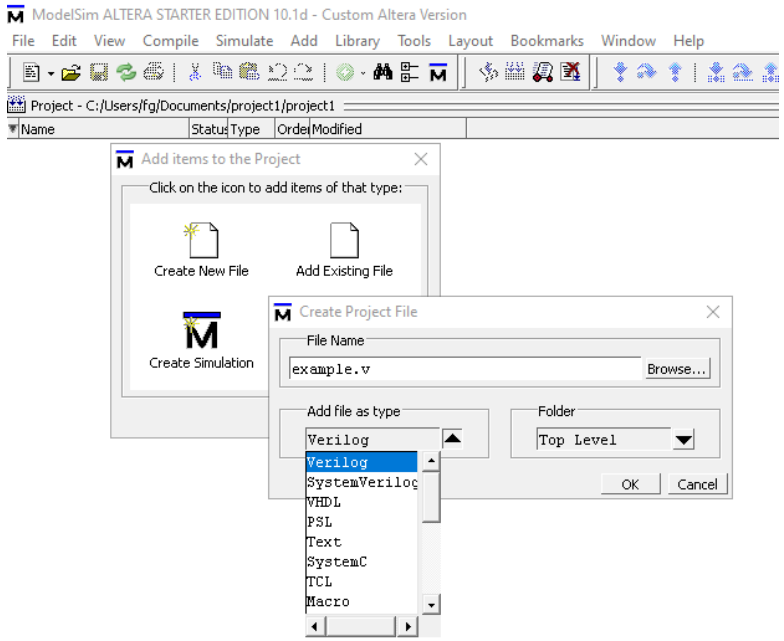
Μπορείτε είτε να φτιάξετε ένα νέο αρχείο είτε να βάλετε αρχεία που υπάρχουν ήδη. Τα αρχεία που θα δίνουμε θα είναι της μορφής Verilog και θα έχουν κατάληξη “.v”.

Στο project που φτιάξαμε θα σχεδιάσουμε με Verilog το γνωστό μας πια κύκλωμα:

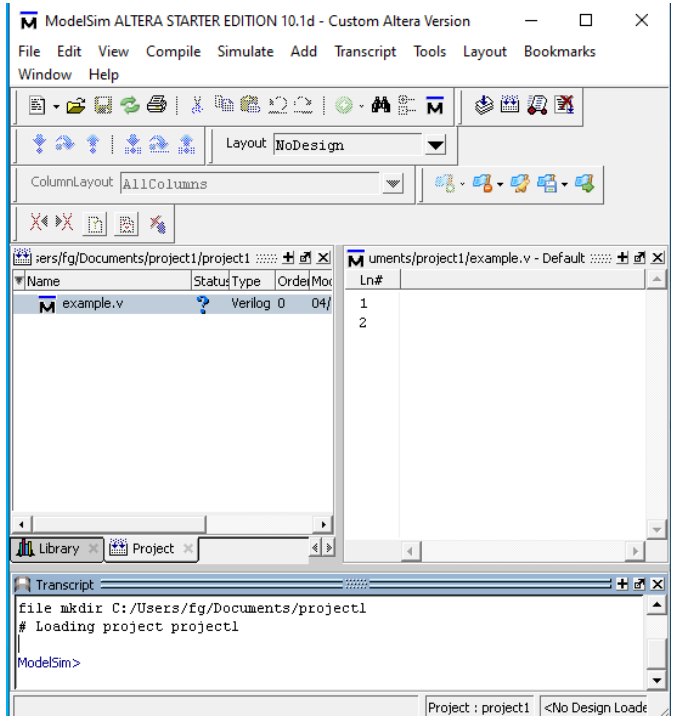


Παράδειγμα Νέου Project στο Modelsim

Αρχικά, θα φτιάξουμε ένα νέο αρχείο στο project μας και θα το ονομάσουμε *example.v*.



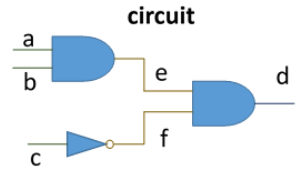
Προσέξτε ότι πρέπει να πείτε στο Modelsim ότι το αρχείο είναι Verilog αρχείο επιλέγοντας το filetype.



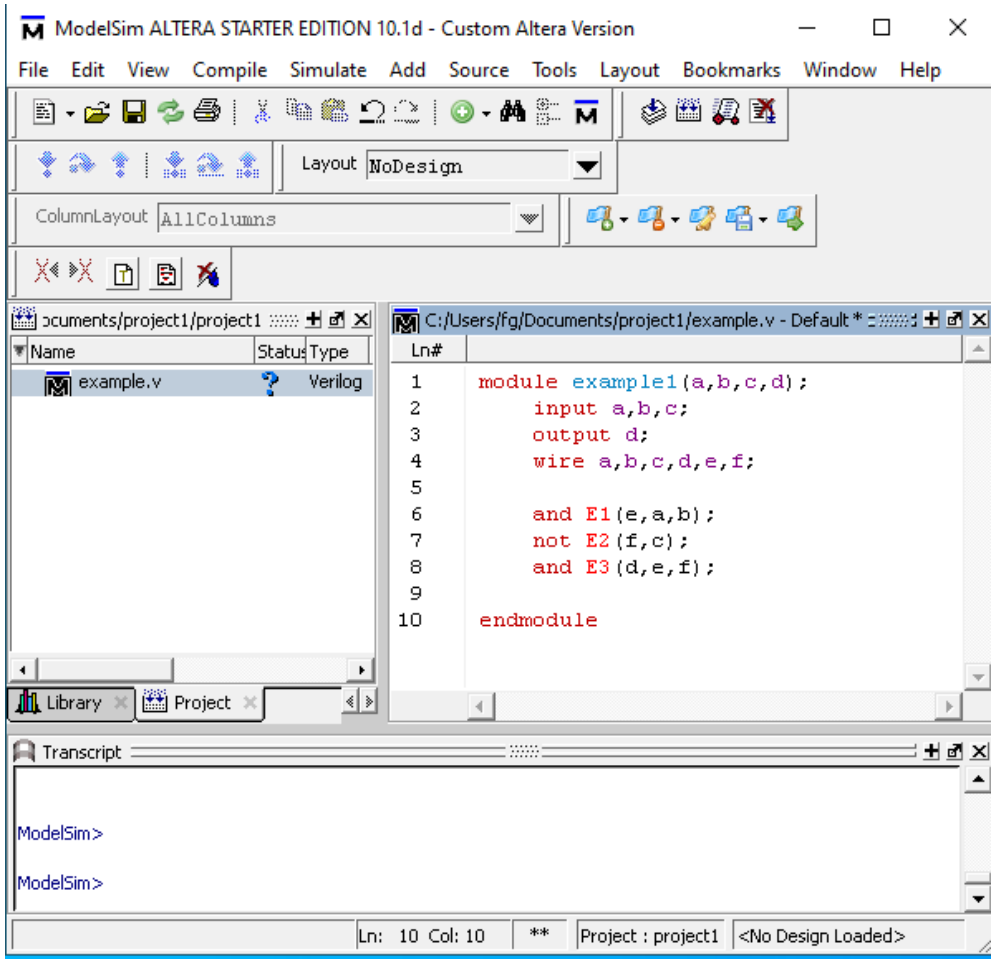
Στην παραπάνω εικόνα φαίνεται το project αριστερά και δεξιά τα περιεχόμενα του αρχείου example.v, το οποίο είναι άδειο. Αν δεν φαίνονται τα περιεχόμενα του αρχείου, κάντε διπλό κλικ πάνω στο αρχείο από το παράθυρο του project και θα το ανοίξει. Πρέπει να είναι άδειο, γιατί μόλις το φτιάξαμε.

Παράδειγμα Νέου Project στο Modelsim

Μέσα στο αρχείο example.v γράφουμε κώδικα Verilog με το γνωστό μας κύκλωμα/παράδειγμα:



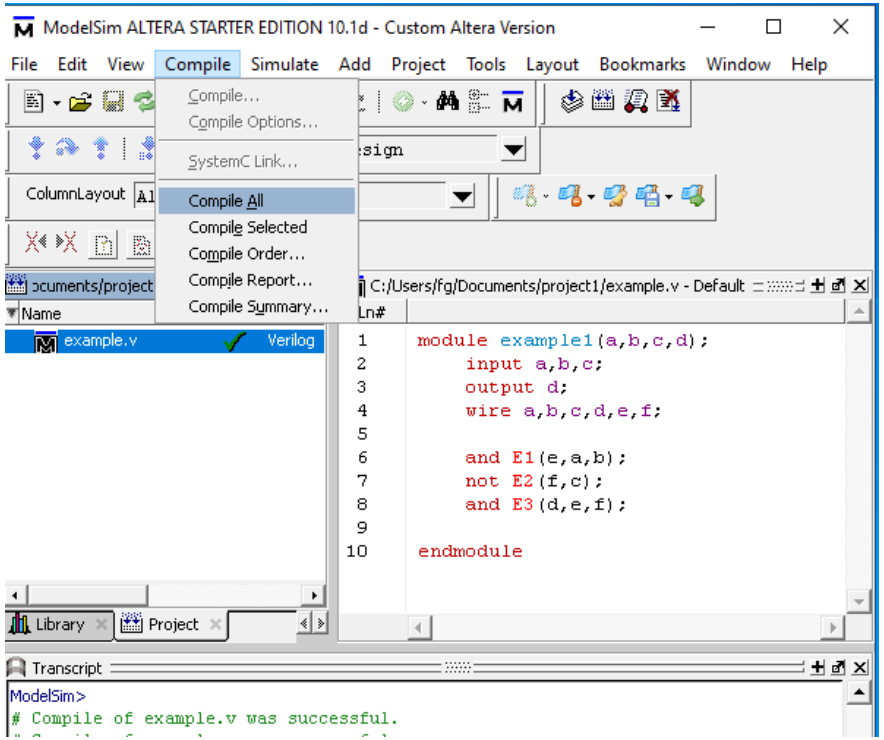
Ο κώδικας φαίνεται στην παρακάτω εικόνα:



Παράδειγμα Compilation

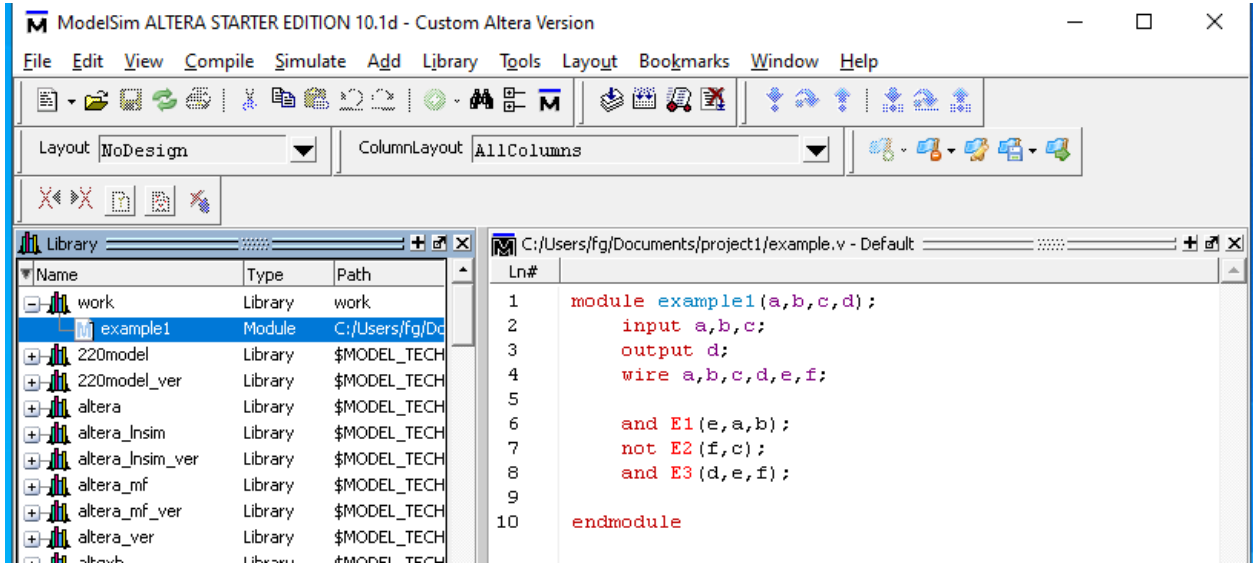
Παράδειγμα Compilation

Επόμενο βήμα είναι να δούμε αν έχουμε κάνει κάποιο λάθος στον κώδικα κάνοντας compilation, εκτελώντας την εντολή Compile->”compile all”, από το menu, όπως φαίνεται στην παρακάτω εικόνα:

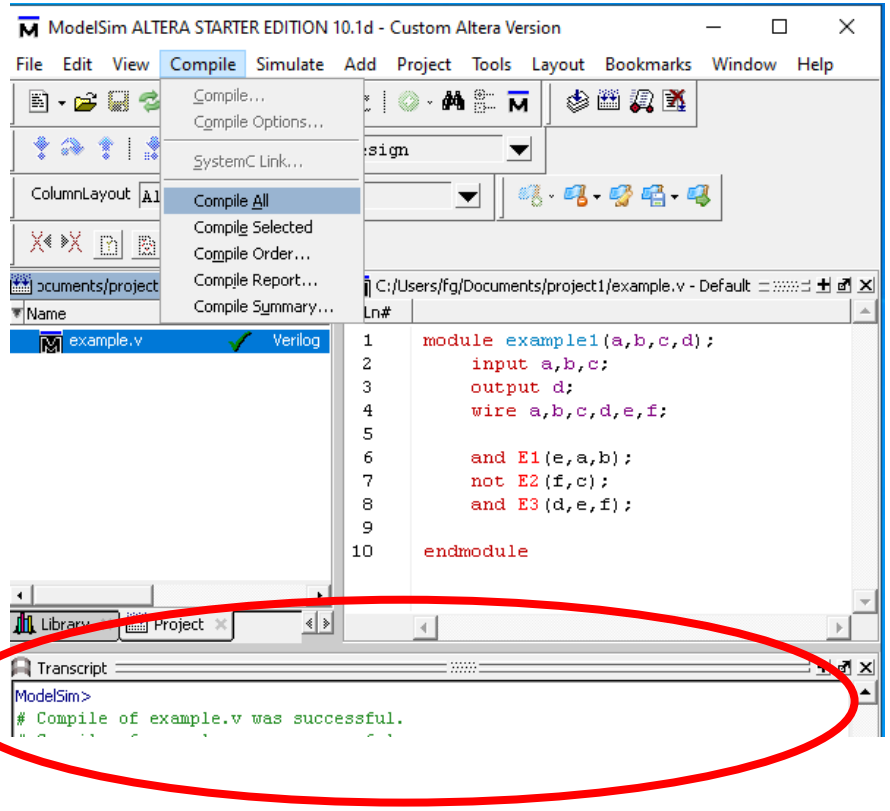


Αν όλα είναι καλά θα πάρουμε μήνυμα επιτυχούς ολοκλήρωσης της διαδικασίας και θα υπάρχει ένα check με πράσινο χρώμα δίπλα από το αρχείο μας στο υποπαράθυρο του project. Αν όχι τότε θα πάρουμε μηνύματα λαθών στο παράθυρο της κονσόλας (Transcript) και με διπλό κλικ πάνω τους μπορούμε να δούμε παραπάνω πληροφορίες που θα μας βοηθήσουν να τα κατανοήσουμε και να τα διορθώσουμε.

Το module που μόλις σχεδιάσαμε σε Verilog έχει τώρα δημιουργηθεί και έχει τοποθετηθεί μέσα στην βιβλιοθήκη work, όπως φαίνεται στην παρακάτω εικόνα:



Κονσόλα του Modelsim



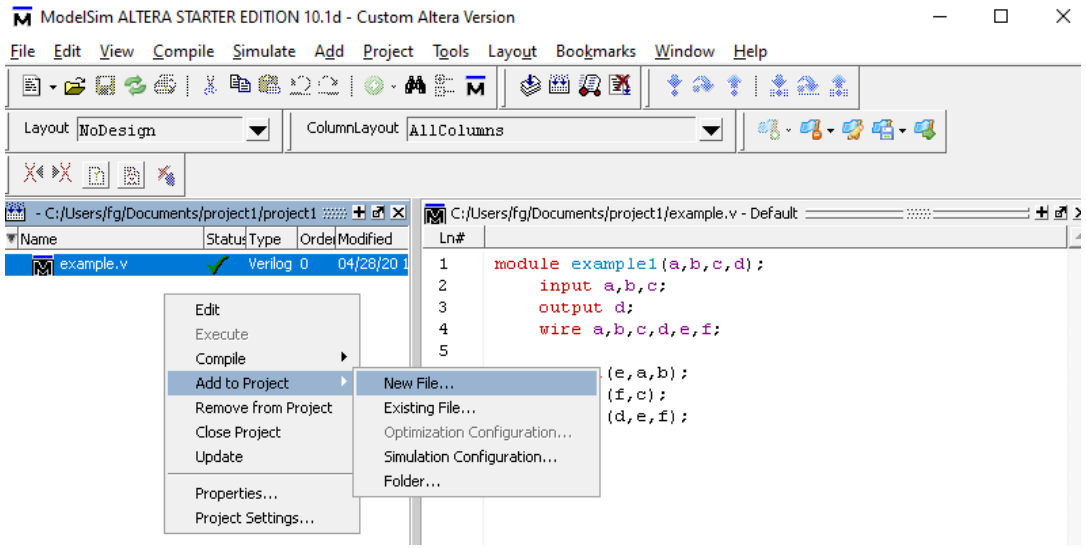
Κονσόλα του Modelsim

Στο κάτω-κάτω παράθυρο φαίνεται η κονσόλα του Modelsim με την προτροπή/prompt:
Modelsim>
Modelsim>
Εκεί θα τυπώνονται μηνύματα από το Modelsim. Επίσης με την κονσόλα μπορούμε να δώσουμε και εντολές. Μάλιστα, οτιδήποτε μπορούμε να κάνουμε με το γραφικό περιβάλλον μέσα στο Modelsim, μπορούμε να το κάνουμε και με την κονσόλα αυτή. Γράψτε για παράδειγμα
Modelsim> help commands
Και θα πάρετε μια λίστα με τις εντολές αυτής τη κονσόλας.
Μπορείτε επίσης να γράψετε
Modelsim> help write
Και να δείτε τι κάνει η write και ποια είναι η σύνταξή της.
Επίσης όταν γράφετε μια εντολή και πατάτε tab, τότε βγάζει σε παραθυράκι (κάτω από το modelsim) πληροφορίες για την εντολή.

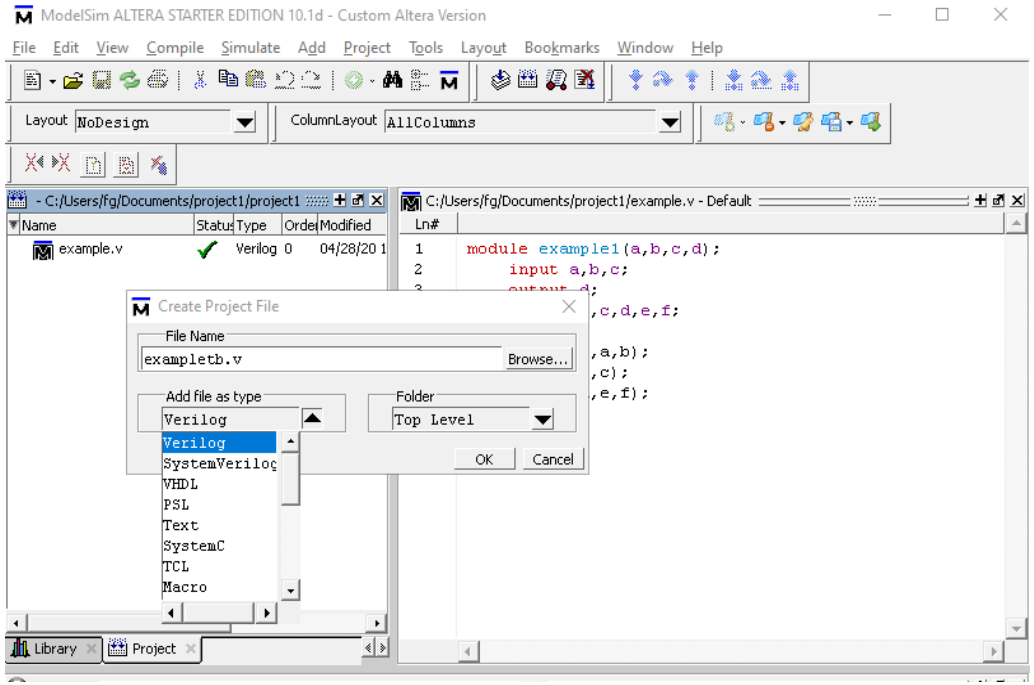
Παράδειγμα Testbench

Παράδειγμα Testbench

Χωρίς testbench δεν μπορούμε να προσομοιώσουμε το module *example1* που μόλις φτιάξαμε.
Για να φτιάξουμε testbench, φτιάχνουμε ένα ακόμα module το οποίο το ονομάζουμε exampletb και το γράφουμε σε ένα αρχείο που ονομάζουμε exampletb.v. Το αρχείο αυτό το κατασκευάζουμε από το menu που εμφανίζεται με δεξί κλικ στο ποντίκι μέσα στο παράθυρο του project, όπως φαίνεται στην παρακάτω εικόνα:



Χωρίς testbench δεν μπορούμε να προσομοιώσουμε το module *example1* που μόλις φτιάξαμε.
Για να φτιάξουμε testbench, φτιάχνουμε ένα ακόμα module το οποίο το ονομάζουμε exampletb και το γράφουμε σε ένα αρχείο που ονομάζουμε exampletb.v. Το αρχείο αυτό το κατασκευάζουμε από το menu που εμφανίζεται με δεξί κλικ στο ποντίκι μέσα στο παράθυρο του project, όπως φαίνεται στην παρακάτω εικόνα:



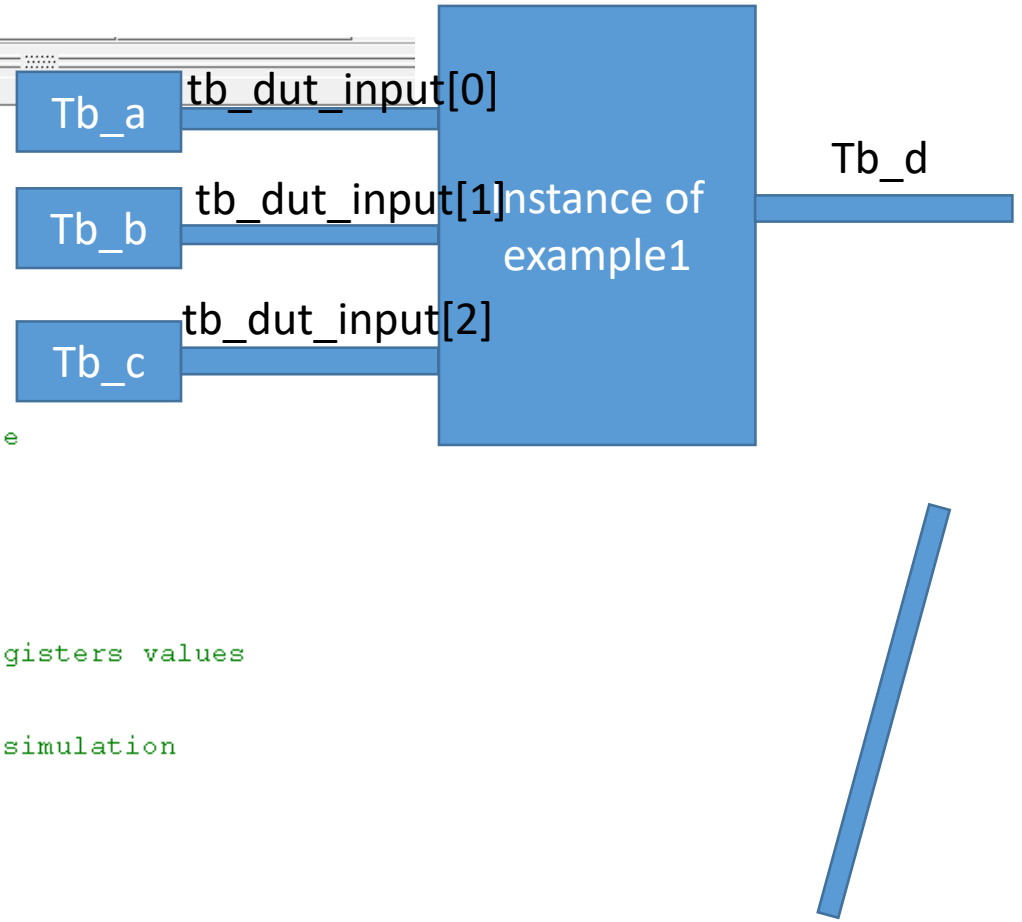
Παράδειγμα Testbench

Example1 dut(.b(tb_b), .a(tb_a),…)

Έπειτα γράφουμε τον κώδικα του testbench, ο οποίος φαίνεται παρακάτω:

C:/Users/fg/Documents/project1/exampletb.v - Default *

```
Ln#
 1  module example1tb();
 2
 3  //input registers to our instantiated module
 4  reg tb_a;
 5  reg tb_b;
 6  reg tb_c;
 7
 8  //bus for writing data to the inputs
 9  wire [2:0] tb_dut_inputs;
10
11  //wire for reading the output of the instantiated module
12  wire tb_d;
13
14  //this is the instantiated module example1.
15  //the name of the instance is dut
16  example1 dut(tb_a, tb_b, tb_c, tb_d);
17
18  //now we create the bus that consists of three input registers values
19  assign tb_dut_inputs={tb_a, tb_b, tb_c};
20
21    //this block is running only at the beginning of the simulation
22  initial begin
23    {tb_a, tb_b, tb_c}=3'b000;
24
25    //the following line runs forever every 5 time units
26    forever #5 {tb_a, tb_b, tb_c}=tb_dut_inputs+1;
27  end //initial begin
28
29  endmodule
```



Παράδειγμα Testbench

Το testbench αποτελείται από 3 καταχωρητές τους οποίους τους χρησιμοποιούμε ως είσοδο για το module example1, είναι τα tb_a, tb_b και tb_c. Τις εξόδους αυτών των τριών καταχωρητών τις ομαδοποιούμε με το bus tb_dut_inputs:

```
//input registers to our instantiated module
reg tb_a;
reg tb_b;
reg tb_c;
//bus for writing data to the inputs
wire [2:0] tb_dut_inputs;
//now we create the bus that consists of three input registers values
assign tb_dut_inputs={tb_a, tb_b, tb_c};
```

Προσέξτε πως στο testbench συνδέσαμε τους 3 καταχωρητές εισόδου σε ένα bus για να μπορούμε να τους διαβάζουμε σαν ένα δυαδικό αριθμό των $3^{ων}$ bits tb_dut_inputs. Χρησιμοποιήσαμε την εντολή assign:

```
assign tb_dut_inputs={tb_a, tb_b, tb_c};
```

Φτιάχνουμε και ένα wire tb_d για να διαβάσουμε την έξοδο του module που τεστάρουμε με το testbench:

```
//wire for reading the output of the instantiated module
wire tb_d;
```

Παράδειγμα Testbench

Στην πορεία φτιάχνουμε ένα instance του module example που σχεδιάσαμε προηγουμένως:

```
//this is the instantiated module example1.
//the name of the instance is dut
example1 dut(tb_a, tb_b, tb_c, tb_d);
```

το ονομάσαμε ως dut (από το design-under-test).

Η καρδιά του testbench είναι το παρακάτω μπλοκ, το οποίο εκτελείτε στην έναρξη της προσομοίωσης:

```
  //this block is running only at the beginning of the simulation
initial begin
  {tb_a, tb_b, tb_c}=3'b000;
  //the following line runs forever every 5 time units
  forever #5 {tb_a, tb_b, tb_c}=tb_dut_inputs+1;
end //initial begin
```

Μηδενίζει τους καταχωρητές:

```
{tb_a, tb_b, tb_c}=3'b000;
```

Και στην πορεία για πάντα και κάθε 5 μονάδες χρόνου (αν δεν δηλώσουμε μονάδα χρόνου τότε είναι σε picoseconds), διαβάζει τους καταχωρητές σε μορφή bus, ως ένα δυαδικό αριθμό τριών bits, από το tb_dut_inputs και την τιμή που διάβασε την αυξάνει κατά 1. Το αποτέλεσμα το γράφει πάλι στους καταχωρητές.

Παράδειγμα Testbench

Στην πορεία φτιάχνουμε ένα instance του module example που σχεδιάσαμε προηγουμένως:

```
//this is the instantiated module example1.
//the name of the instance is dut
example1 dut(tb_a, tb_b, tb_c, tb_d);
```

το ονομάσαμε ως dut (από το design-under-test).

Η καρδιά του testbench είναι το παρακάτω μπλοκ, το οποίο εκτελείτε στην έναρξη της προσομοίωσης:

```
  //this block is running only at the beginning of the simulation
initial begin
  {tb_a, tb_b, tb_c}=3'b000;
  //the following line runs forever every 5 time units
  forever #5 {tb_a, tb_b, tb_c}=tb_dut_inputs+1;
end //initial begin
```

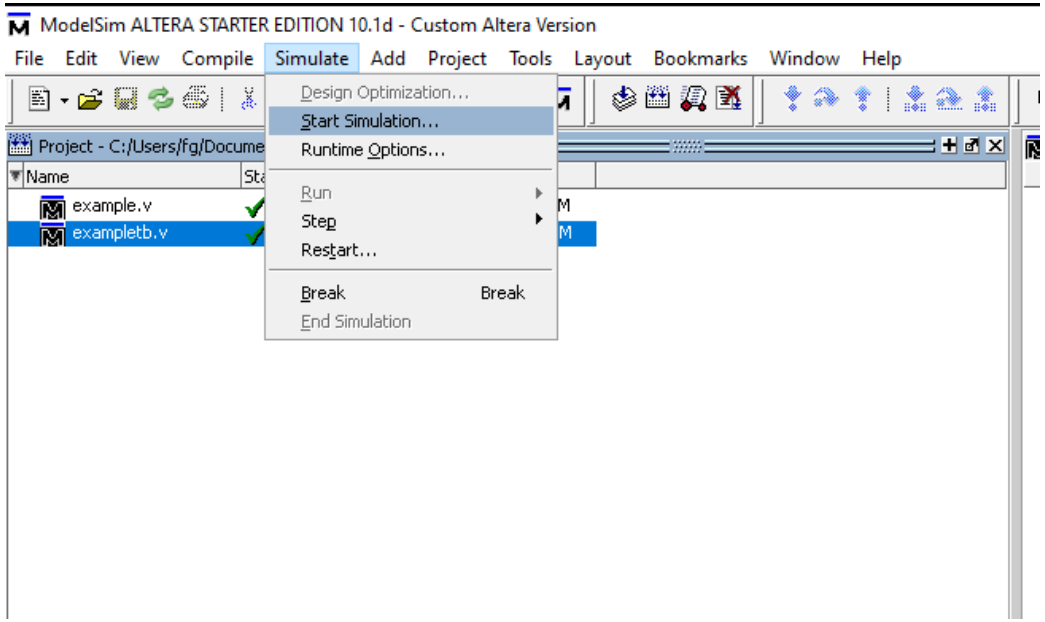
Μηδενίζει τους καταχωρητές:

```
{tb_a, tb_b, tb_c}=3'b000;
```

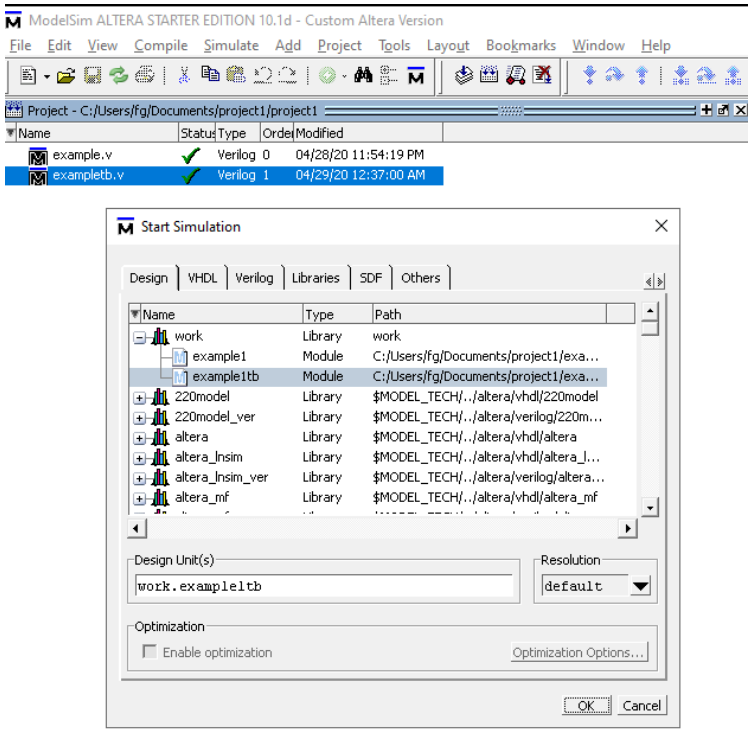
Και στην πορεία για πάντα και κάθε 5 μονάδες χρόνου (αν δεν δηλώσουμε μονάδα χρόνου τότε είναι σε picoseconds), διαβάζει τους καταχωρητές σε μορφή bus, ως ένα δυαδικό αριθμό τριών bits, από το tb_dut_inputs και την τιμή που διάβασε την αυξάνει κατά 1. Το αποτέλεσμα το γράφει πάλι στους καταχωρητές.

Παράδειγμα Προσομοίωσης

Παράδειγμα Προσομοίωσης



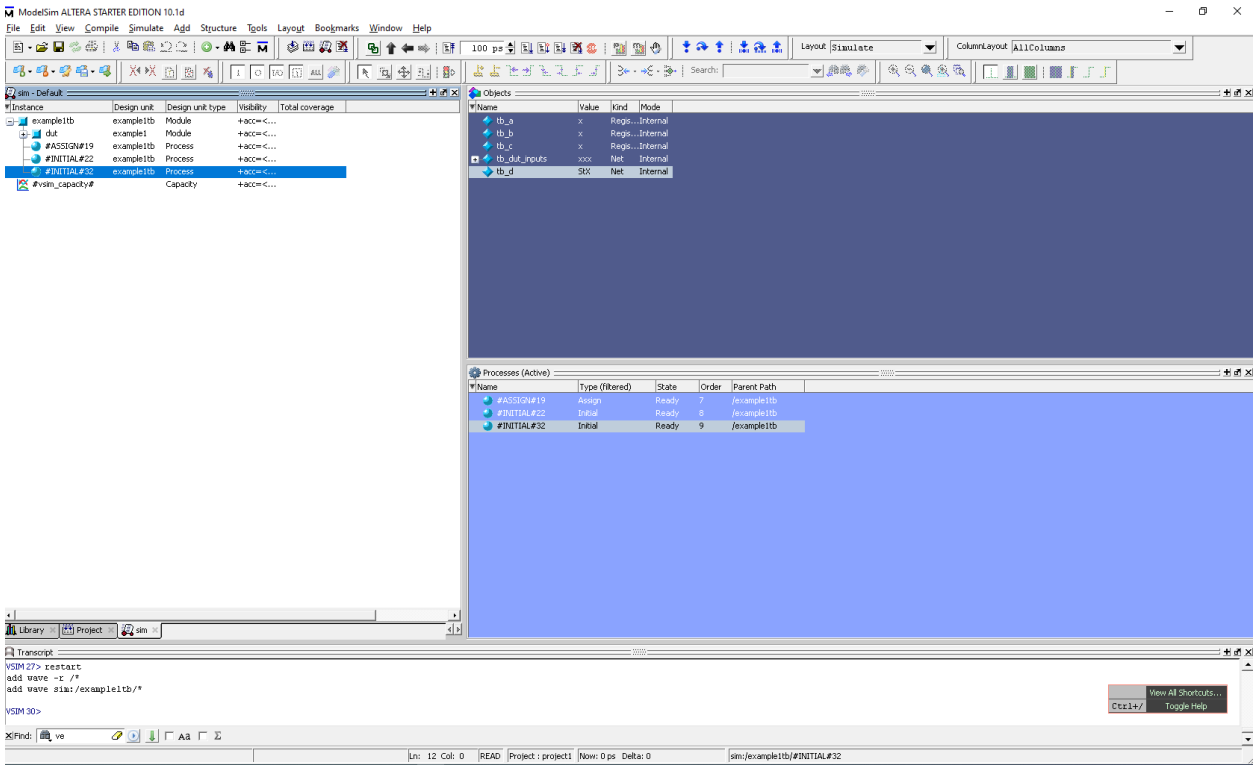
Για να ξεκινήσουμε την προσομοίωση εκτελούμε Simulate->”Start Simulation”. Θα ανοίξει το παρακάτω παράθυρο, στο οποίο πρέπει να επιλέξουμε το module example1tb, που είναι το testbench που μόλις φτιάξαμε



ΠΡΟΣΟΧΗ: Πάντα προσομοιώνουμε ένα testbench, μέσα στο οποίο είναι ως instance module το βασικό μας design. Μετά επιλέγουμε «OK» και το Modelsim θα μπει σε κατάσταση προσομοίωσης. Περιμένουμε λίγο μέχρι το Modelsim να αλλάξει μορφή, μπορεί να χρειαστούν κάποια δεκάδες δευτερόλεπτα.

Παράδειγμα Προσομοίωσης

Στο στάδιο που είμαστε το Modelsim μπήκε σε κατάσταση προσομοίωσης και θα πρέπει να δείχνει κάπως έτσι:



Η προσομοίωση έχει ξεκινήσει, αλλά έχει μείνει «παγωμένη» στην χρονική στιγμή t=0

Παράθυρο Sim: Προσέξτε ότι το αριστερά παράθυρο είναι νέο και ονομάζεται Sim – Default. Έχει τα αντικείμενα και τα processes του κώδικά μας ιεραρχικά. Φροντίζουμε να έχουμε επιλεγμένο το testbench εκεί (το example1tb), γιατί θέλουμε να παρατηρήσουμε τι γίνεται σε αυτό.

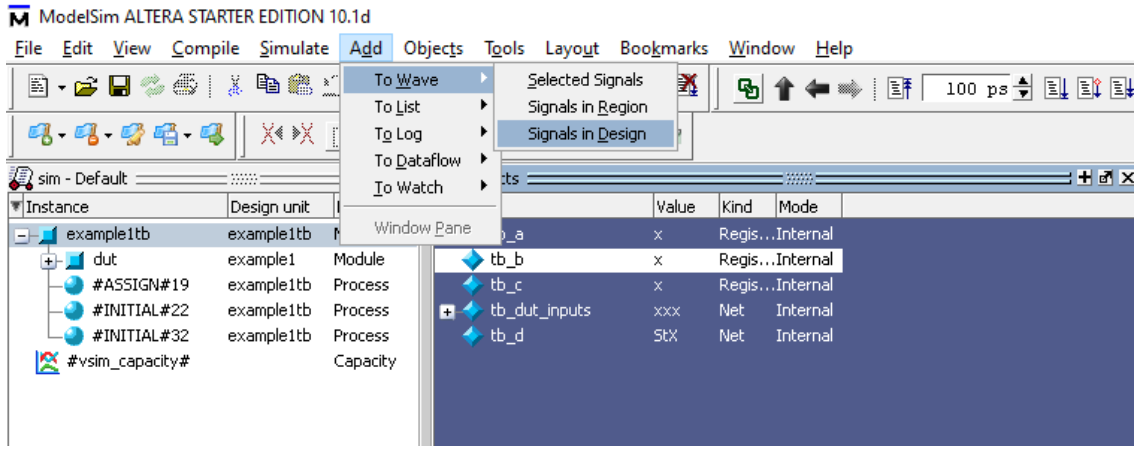
Παράθυρο Objects: Προσέξτε το κεντρικό μπλε σκούρο παράθυρο που λέγεται Objects, αυτά είναι αντικείμενα του Simulator την ώρα που τρέχει η προσομοίωσή μας.

Παράθυρο Processes: Επίσης προσέξτε το ανοιχτόχρωμο μπλε παράθυρο, που λέγεται Processes, αυτά είναι τα Processes/blocks που είχε το testbench μας.

Σε αυτά τα παράθυρα αν πατήσουμε σε κάποια επιλογή διπλό κλικ, μας πηγαίνει στον κώδικα τους.

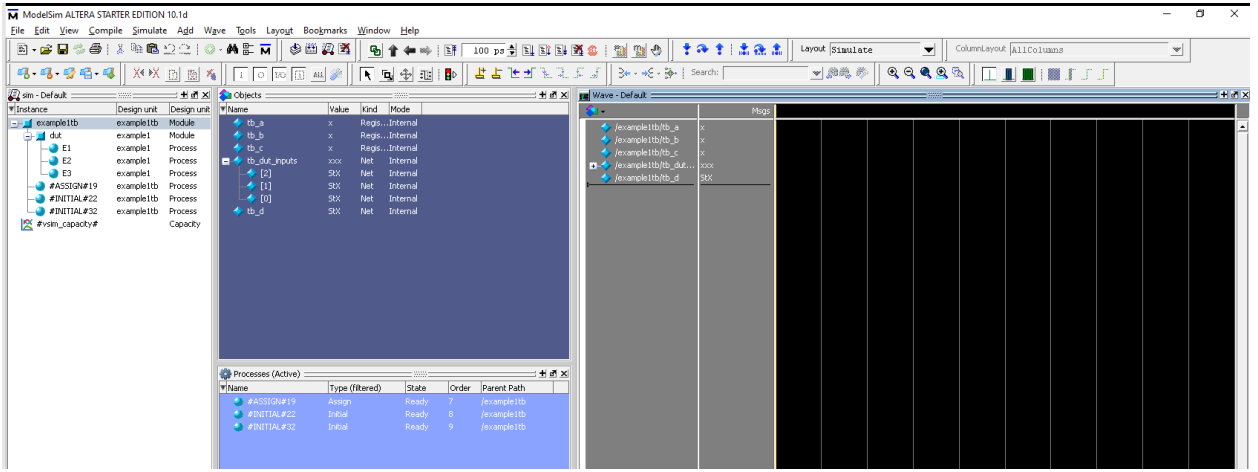
Παράδειγμα Προσομοίωσης

Για να προχωρήσουμε την προσομοίωση (και να καταλάβουμε ότι έχει ξεκινήσει), πρέπει να φτιάξουμε ένα παράθυρο κυμματομορφών (wave window). Αυτό το κάνουμε επιλέγοντας από το Menu Add→"To Wave"→"All items in in Region", όπως φαίνεται στην παρακάτω εικόνα.



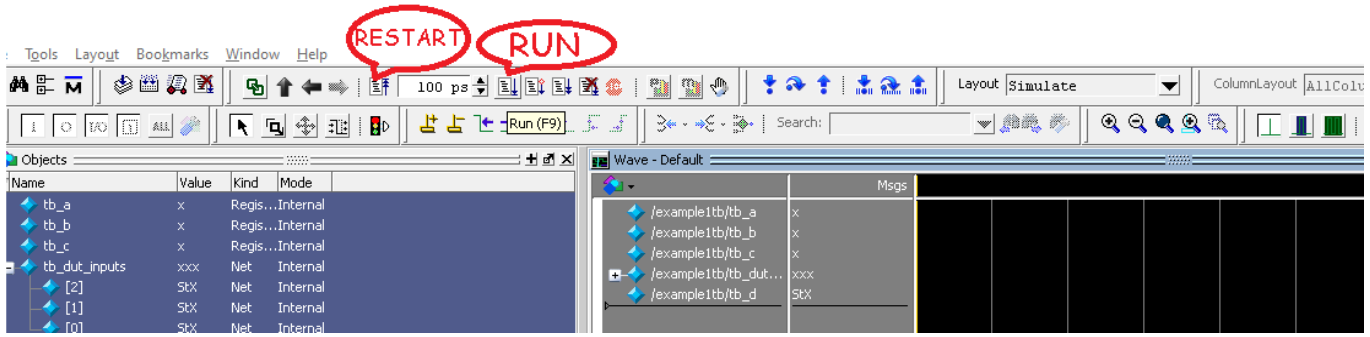
Προσοχή για να δουλέψει σωστά αυτή η επιλογή, πρέπει το στο παράθυρο Sim να είναι επιλεγμένο το testbench, αλλιώς θα μας δείξει τα σήματα του region που είναι επιλεγμένο. Το region επομένως είναι σαν το scope στον προγραμματισμό λογισμικού.

Θα ανοίξει ένα νέο παράθυρο δεξιά που θα ονομάζεται Wave- Default. Εκεί πρέπει να περιέχει τα σήματα του testbench και θα μοιάζει κάπως έτσι:

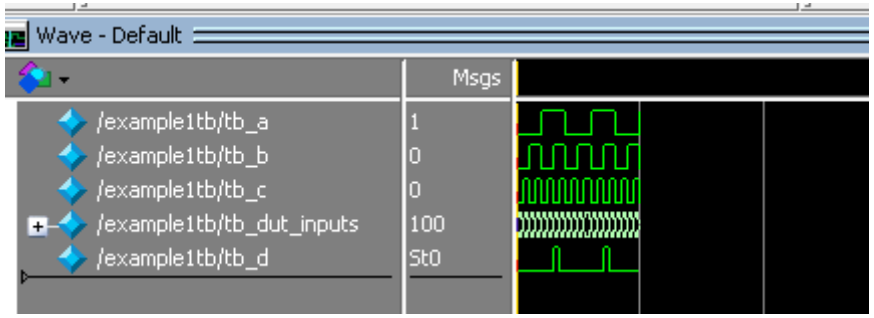


Παράδειγμα Προσομοίωσης

Προσέξτε ότι οι κυμματομορφές είναι κενές γιατί ακόμα είμαστε στην αρχή της προσομοίωσης, ο χρόνος είναι t=0. Επόμενο βήμα είναι να πούμε στον προσομοιωτή να προχωρήσει το χρόνο. Το κάνουμε αυτό με την επιλογή run που φαίνεται παρακάτω:



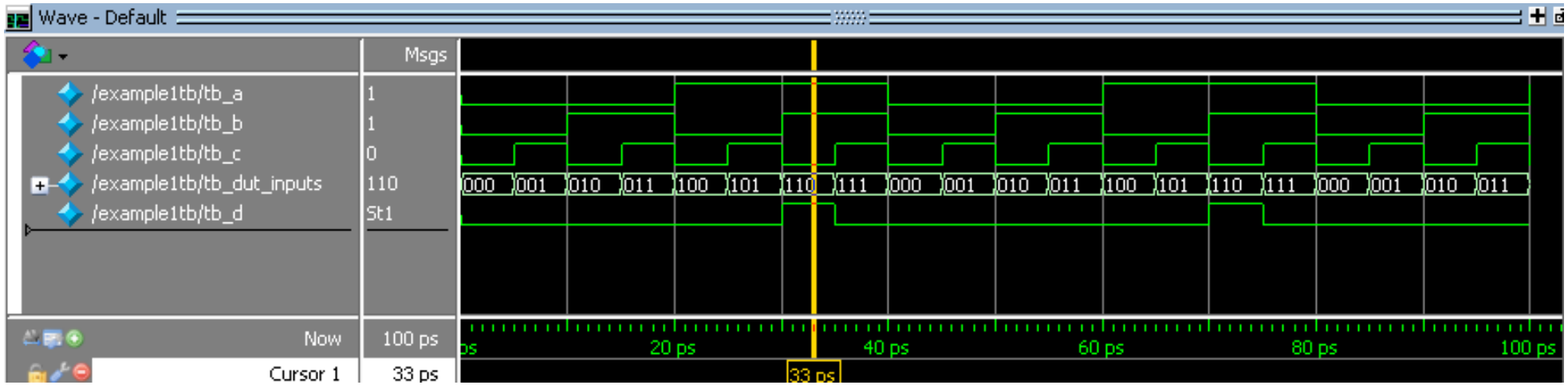
Η επιλογή αυτή θα προχωρήσει την προσομοίωση κατά όσο χρόνο λέει το κουτάκι αριστερά της, δηλαδή 100 ps. Μόλις πατήσουμε run προκύπτει το παρακάτω:



Στο παράθυρο αυτό μπορούμε να περιηγηθούμε και αν βρούμε τις τιμές των σημάτων σε κάθε χρονική στιγμή και να ελέγξουμε οπτικά αν όλα τρέχουν σωστά. Η περιήγηση γίνεται με το πλήκτρο ctrl από το πληκτρολόγιο πατημένο και την ρόδα από το ποντίκι για zoom-in και zoom-out. Επίσης μπορούμε να πατήσουμε περισσότερες από μία φορές το run και να προχωρήσουμε τον χρόνο παρακάτω.

Παράδειγμα Προσομοίωσης

Παρατηρούμε ότι το σήμα tb_d που είναι η έξοδος του κυκλώματος example1 είναι συνήθως στο λογικό ‘0’ και γίνεται λογικό ‘1’ μόνο όταν οι είσοδοι είναι tb_a=1, tb_b=1 και tb_c=0, που είναι και η σωστή συμπεριφορά που περιμέναμε από τον πίνακα αληθείας του κυκλώματός μας.



Τερματισμός Προσομοίωσης

Τερματισμός Προσομοίωσης

Για να τερματίσουμε την προσομοίωση επιλέγουμε Simulate→”End Simulation”.