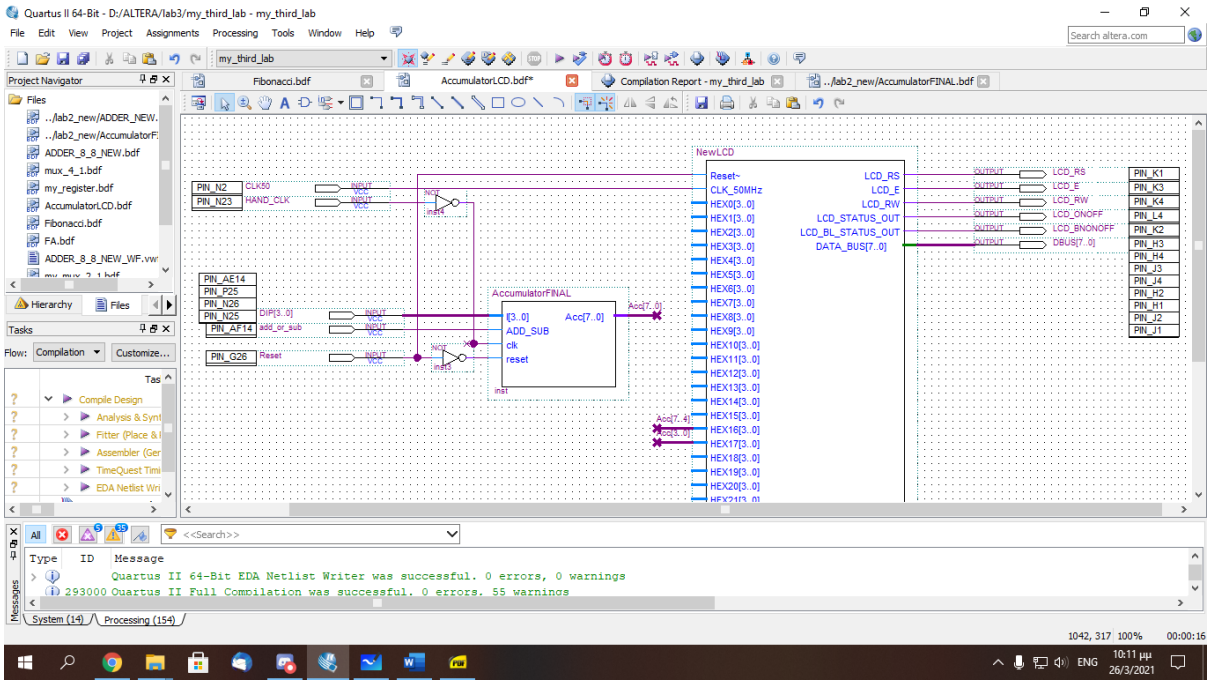
ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ 2

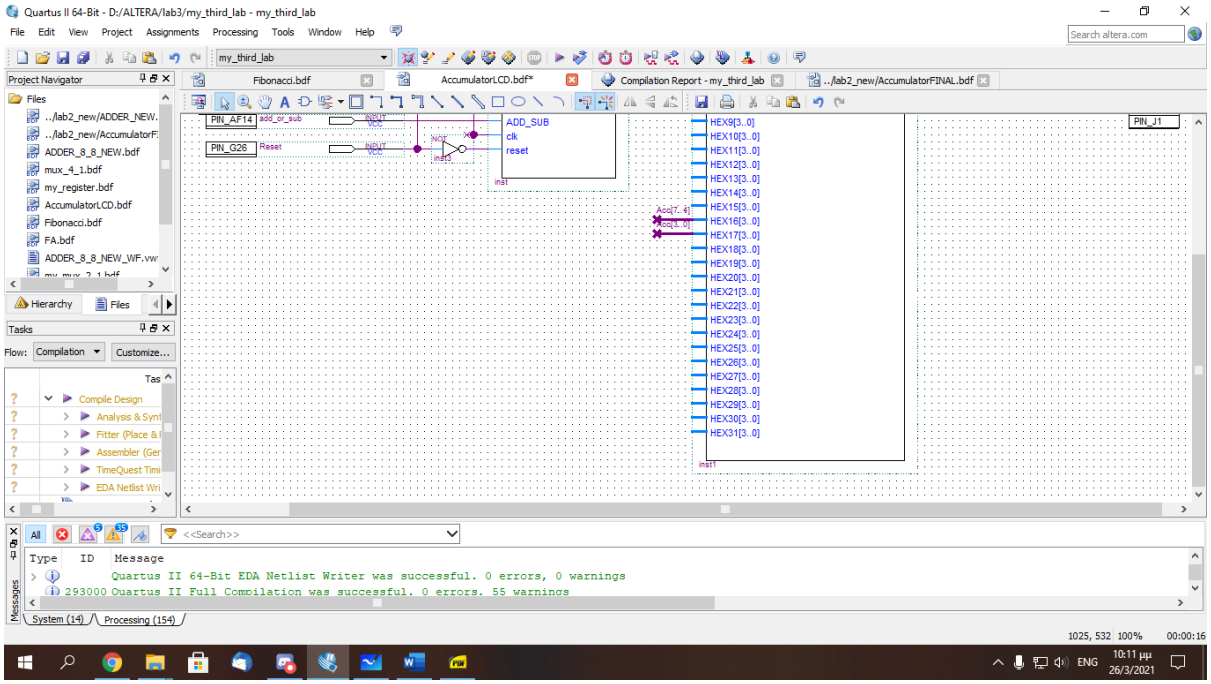
3η Εργαστηριακή Άσκηση

Γκότσης Βασίλης 3206

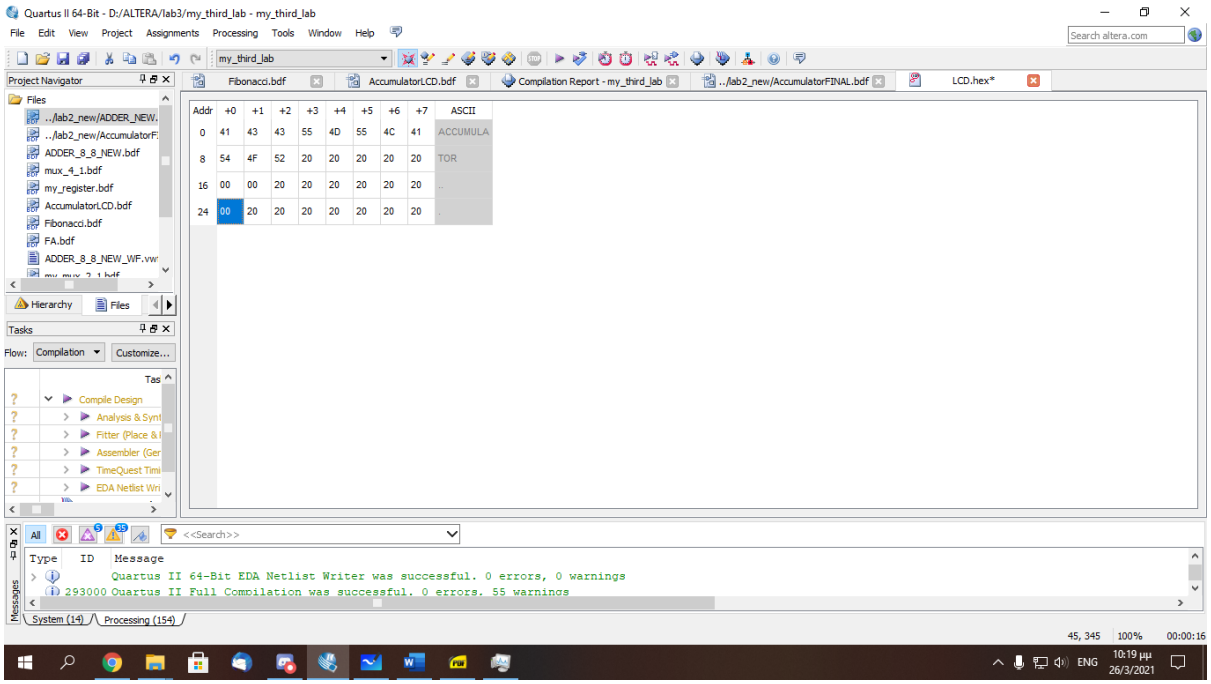
Πρίφτη Ιωάννης 3321

Στις παρακάτω εικόνες φαίνεται το κύκλωμα του συσσωρευτή(από το δεύτερο εργαστήριο) σε συνδυασμό με την NewLCD όπως ακριβώς δίνεται στην εκφώνηση και με τα αντίστοιχα pin συνδεδεμένα:



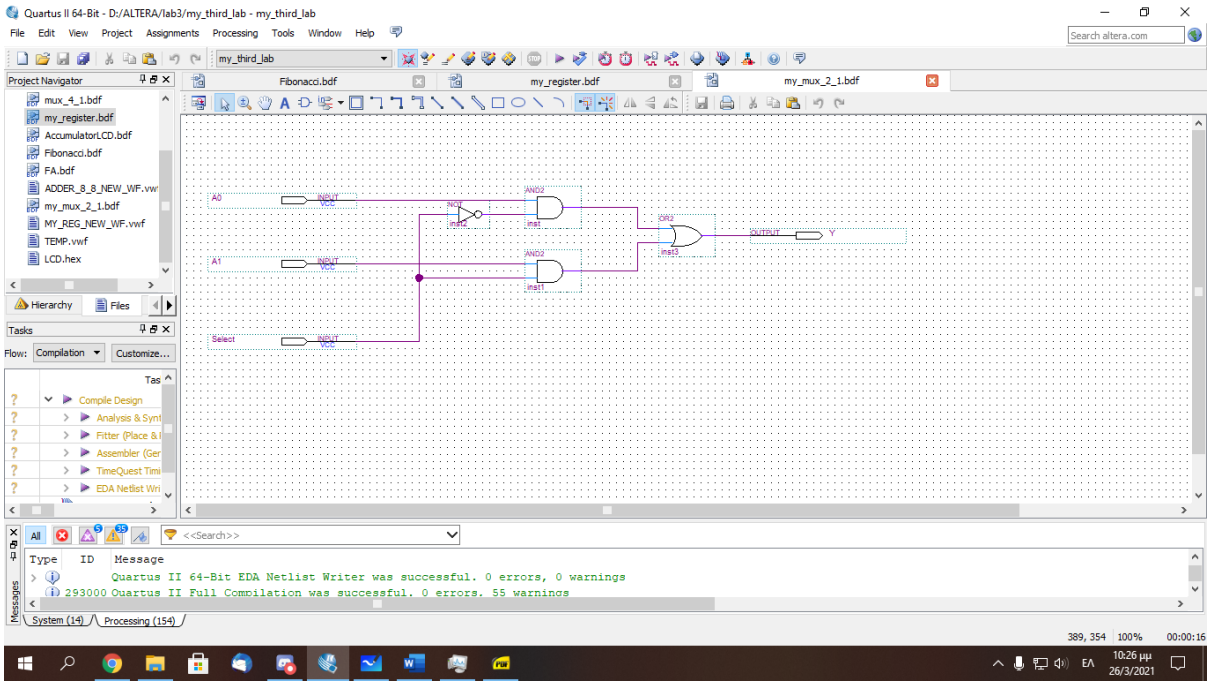


Αντίστοιχα, συμπληρώσαμε και τις θέσεις μνήμης τις οθόνης όπως ακριβώς αναγράφεται στην εκφώνηση, φαίνεται στην παρακάτω εικόνα:



Υπολογισμός Ακολουθίας Fibonacci

Αρχικά ξεκινήσαμε φτιάχνοντας έναν πολυπλέκτη 2 σε 1 με εξίσωση Y=(Select'*A0)+(Select*A1), όπως φαίνεται στην παρακάτω εικόνα:



Ο λόγος που φτιάξαμε τον πολυπλέκτη είναι για να μπορούμε να τον προσθέσουμε στον καταχωρητή των 8bit έτσι ώστε να μπορούμε να επιλέγουμε ανάμεσα σε παράλληλη φόρτωση είτε δεξιά ολίσθηση των δεδομένων.

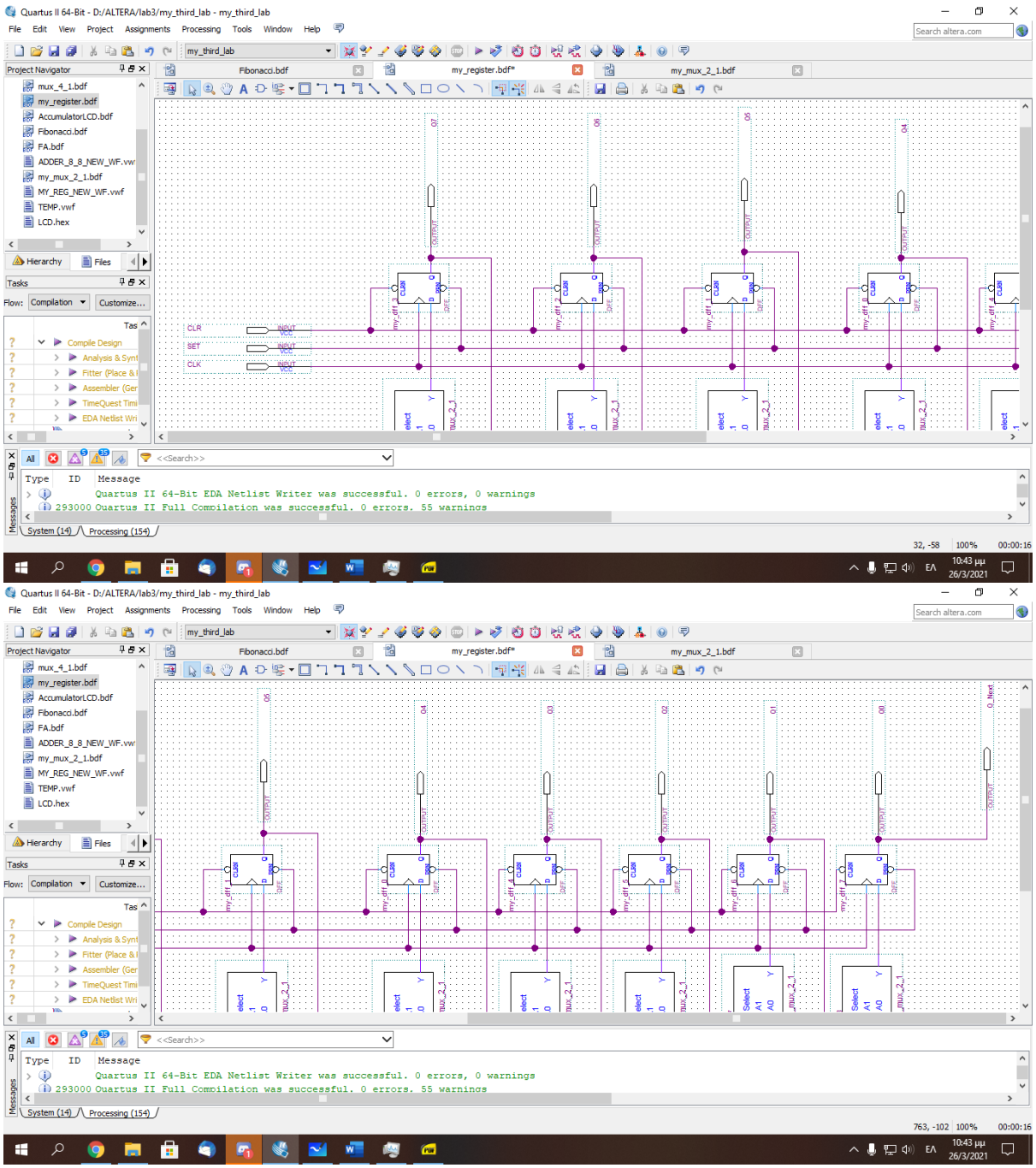
Ο καταχωρητής αποτελείται από 8 πολυπλέκτες 2σε1 και από 8 d-ff.

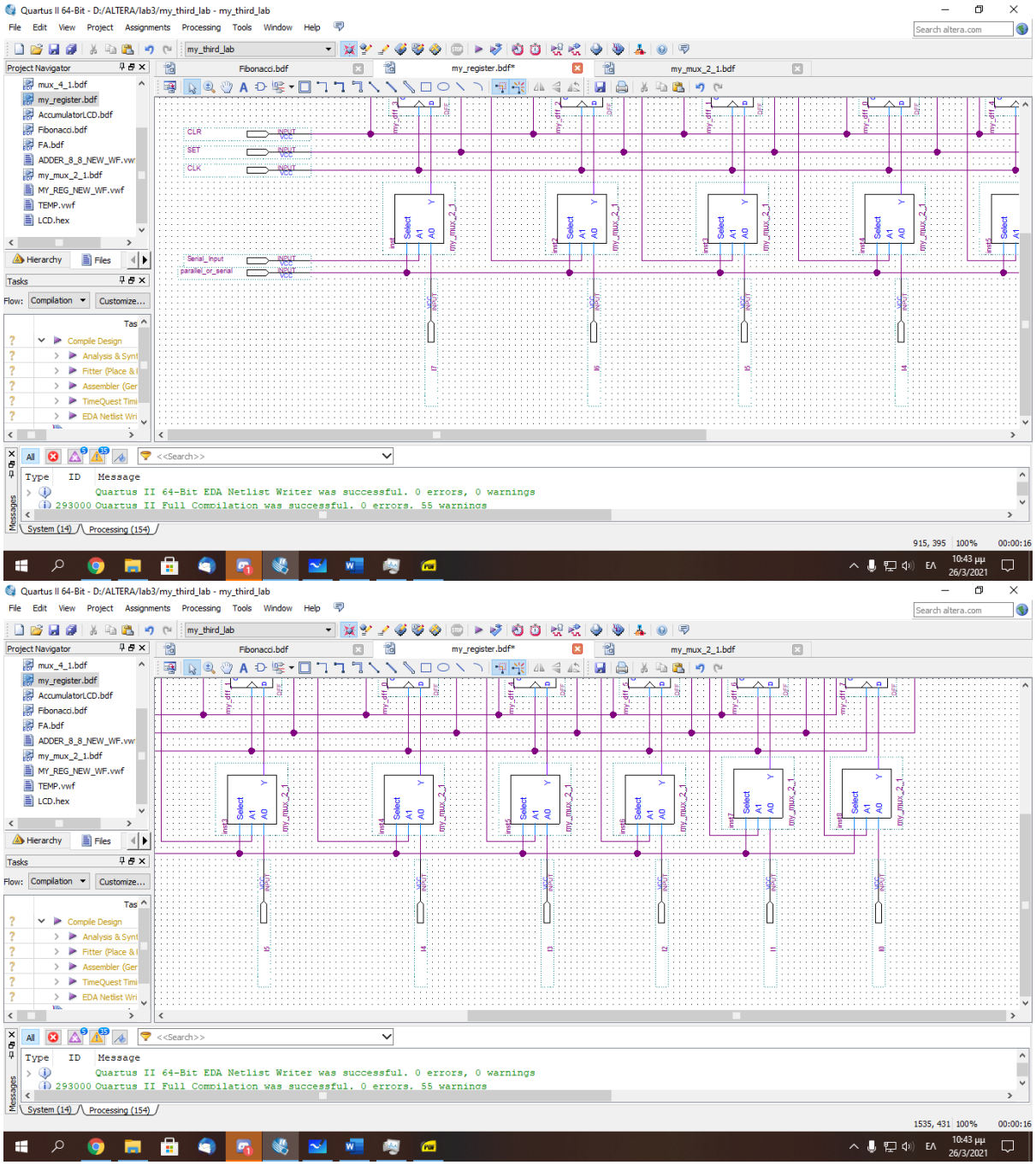
Οι πολυπλέκτες έχουν μια κοινή γραμμή επιλογής του ενός bit (parallel_or_serial)

Βάζοντας parallel_or_serial=0 αυτό που επιτυγχάνουμε είναι να γίνεται παράλληλη φόρτωση από τις εισόδους I[7..0] ενώ όταν βάζουμε parallel_or_serial=1 επιτυγχάνουμε την δεξιά ολίσθηση των περιεχομένων του καταχωρητή και ταυτόχρονα εισάγει την τιμή της εισόδου Serial_Input.

Ο καταχωρητής έχει τις εξόδους Q[7..0] και Q_Next(όπου Q_Next=Q[0], η οποία είναι ουσιαστικά η Q0 την οποία και προσθέσαμε για να φαίνεται πιο καθαρά στο κύκλωμα)

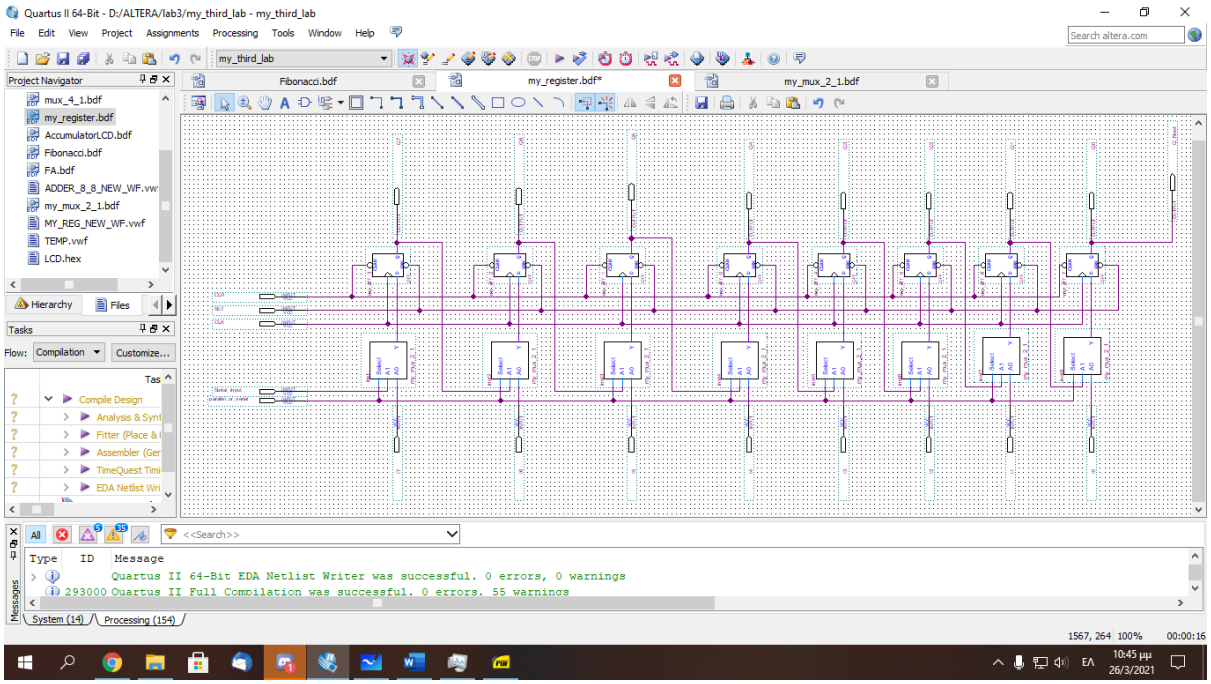
Την Q_Next την βάλαμε για τον λόγο ότι θέλουμε να ολισθαίνει σειριακά και στον δεύτερο καταχωρητή τις τιμές που περιέχει, όπως φαίνεται στις παρακάτω εικόνες:



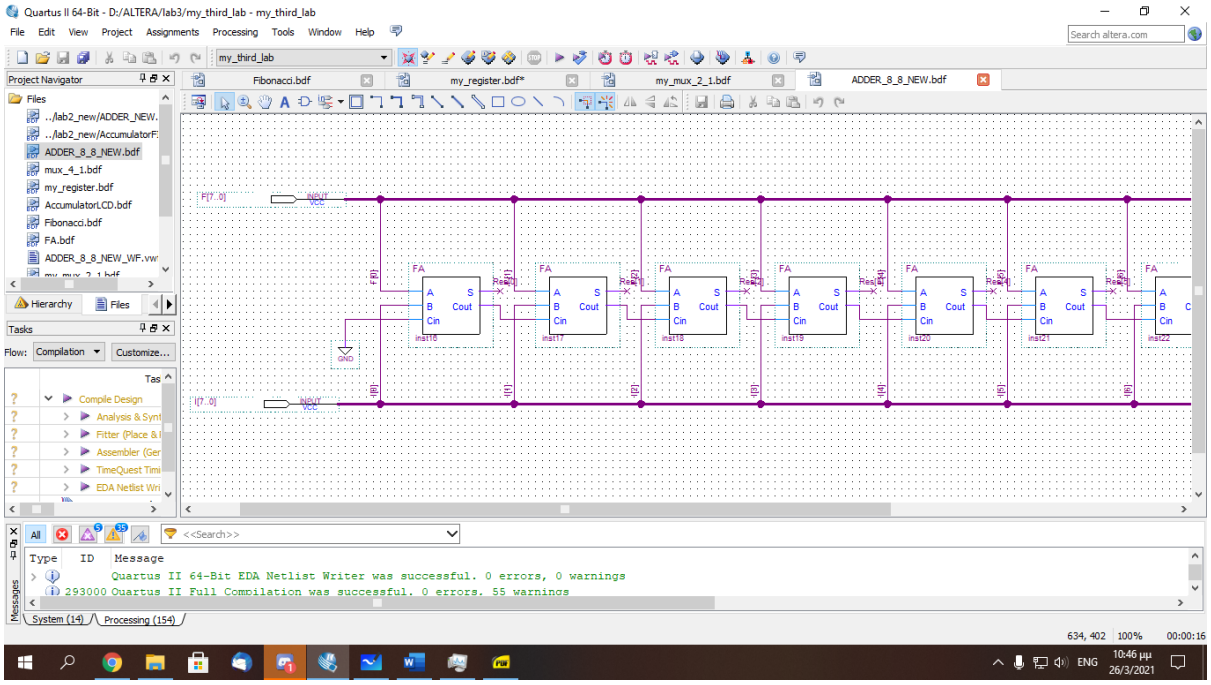

Quartus II 64-Bit - D:/ALTERA/lab3/my_third_lab - my_third_lab
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
my_third_lab
Project Navigator
Fibonacci.bdf
my_register.bdf*
my_mux_2_1.bdf
mux_4_1.bdf
my_register.bdf
AccumulatorLCD.bdf
Fibonacci.bdf
FA.bdf
ADDER_8_8_NEW_WF.vwf
my_mux_2_1.bdf
MY_REG_NEW_WF.vwf
TEMP.vwf
LCD.hex
Hierarchy
Files
Tasks
Flow: Compilation
Customize...
Compile Design
Analysis & Synt
Fitter (Place & I
Assembler (Ger
TimeQuest Timi
EDA Netlist Wri
CLR
SET
CLK
Serial_input
parallel_or_serial
Select
A1
A0
Y
my_mux_2_1
Type ID Message
Quartus II 64-Bit EDA Netlist Writer was successful. 0 errors, 0 warnings
293000 Quartus II Full Compilation was successful. 0 errors, 55 warnings
System (14)
Processing (154)
915, 395 100%
00:00:16
10:43 μμ
26/3/2021

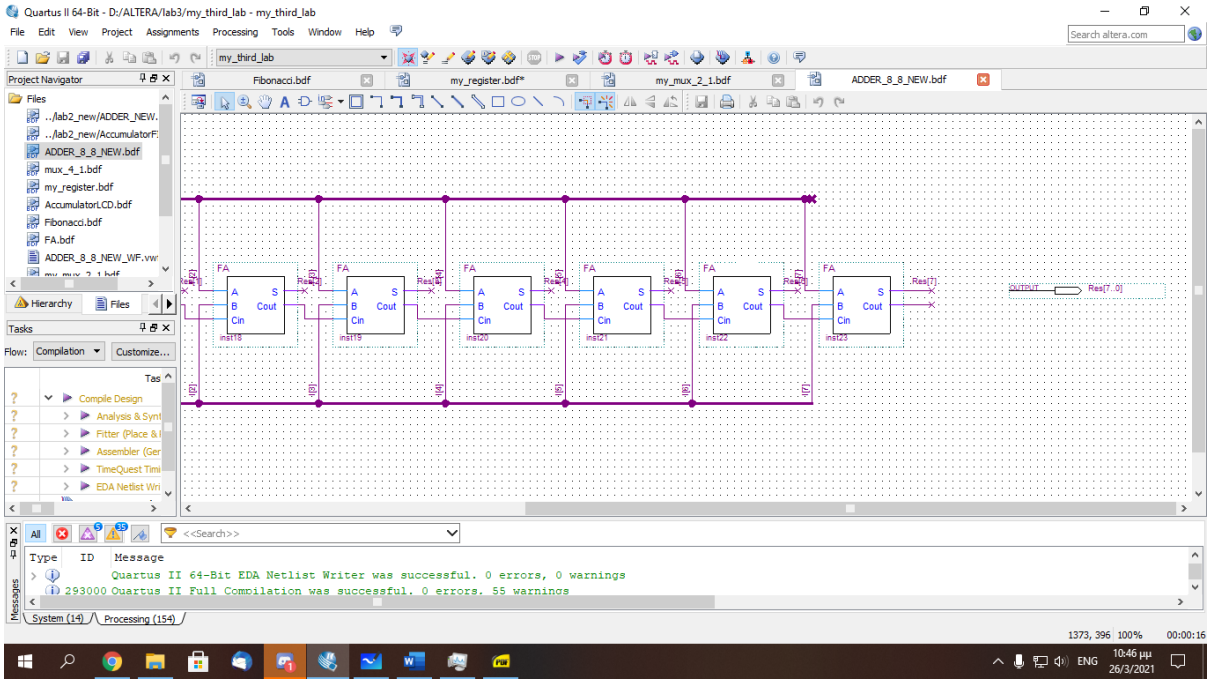
Quartus II 64-Bit - D:/ALTERA/lab3/my_third_lab - my_third_lab
File Edit View Project Assignments Processing Tools Window Help
Search altera.com
my_third_lab
Project Navigator
Fibonacci.bdf
my_register.bdf*
my_mux_2_1.bdf
mux_4_1.bdf
my_register.bdf
AccumulatorLCD.bdf
Fibonacci.bdf
FA.bdf
ADDER_8_8_NEW_WF.vwf
my_mux_2_1.bdf
MY_REG_NEW_WF.vwf
TEMP.vwf
LCD.hex
Hierarchy
Files
Tasks
Flow: Compilation
Customize...
Compile Design
Analysis & Synt
Fitter (Place & I
Assembler (Ger
TimeQuest Timi
EDA Netlist Wri
Select
A1
A0
Y
my_mux_2_1
Type ID Message
Quartus II 64-Bit EDA Netlist Writer was successful. 0 errors, 0 warnings
293000 Quartus II Full Compilation was successful. 0 errors, 55 warnings
System (14)
Processing (154)
1535, 431 100%
00:00:16
10:43 μμ
26/3/2021

All_in_one



Όσον αφορά τον full-adder των 8bit χρησιμοποιήσαμε 8 Full-Adders των 1bit και με παρόμοια λογική με εκείνη του δεύτερου εργαστηρίου υλοποιήσαμε το κύκλωμα όπως φαίνεται στις παρακάτω εικόνες:





Όσον αφορά το τελικό κύκλωμα που υπολογίζει την ακολουθία Fibonacci αρχικά πακετάραμε τα προηγούμενα κυκλώματα σε κουτάκια.

Αρχικά συνδέσαμε τις εξόδους(Q[7..0]) των δύο καταχωρητών στον πλήρη αθροιστή ώστε να γίνεται η πρόσθεση και να προκύψει έτσι ο επόμενος αριθμός από την ακολουθία Fibonacci.

Στην συνέχεια συνδέσαμε την έξοδο Q_Next του πρώτου καταχωρητή(A) με την είσοδο Serial_Input του δεύτερου καταχωρητή(B) ώστε να περνάνε σειριακά τα δεδομένα από τον πρώτο στον δεύτερο.

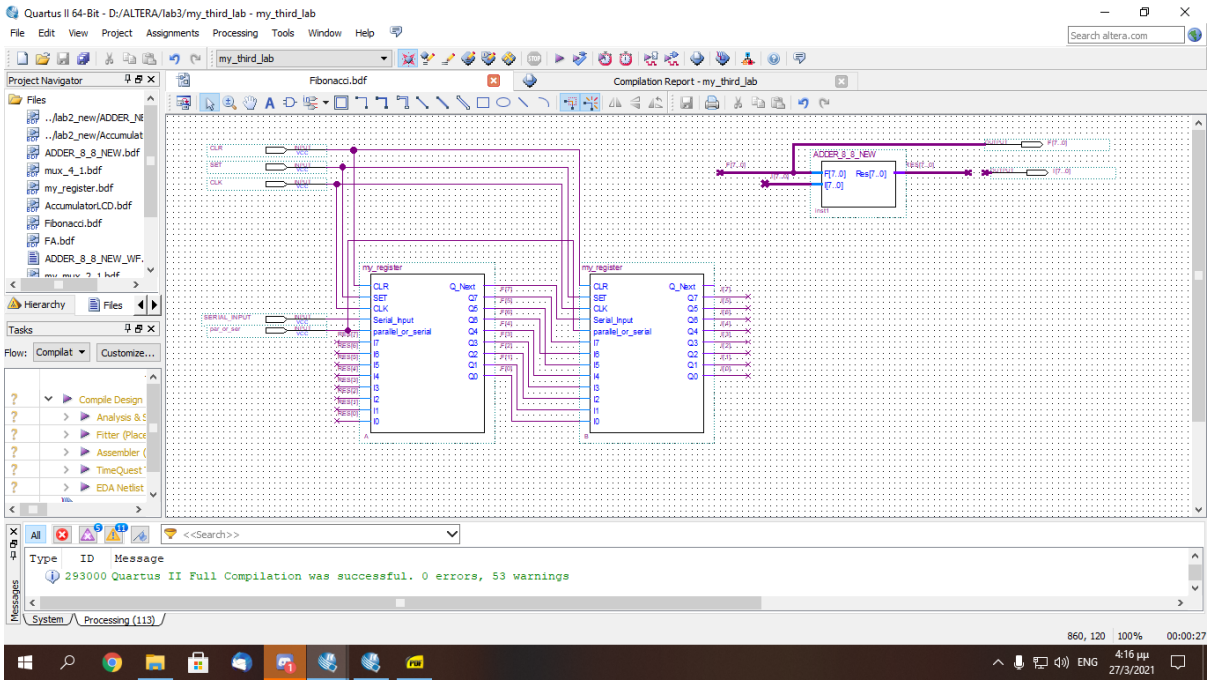
Και οι δύο καταχωρητές έχουν κοινή είσοδο par_or_ser για την επιλογή της παράλληλης ή της σειριακής φόρτωσης των δεδομένων.

Επίσης έχουμε τοποθετήσει και την είσοδο SERIAL_INPUT στον πρώτο καταχωρητή για την σειριακή είσοδο των δεδομένων από τον χρήστη, φυσικά στην περίπτωση που ο διακόπτης par_or_ser=1

Στις παράλληλες εισόδους(I[7..0]) του πρώτου καταχωρητή(A) συνδέσαμε τις εξόδους(RES[7..0]) του πλήρη αθροιστή ώστε να μπορέσουμε κρατάμε τον N όρο της ακολουθίας, ενώ στις παράλληλες εισόδους(I[7..0]) του δεύτερου καταχωρητή(B) συνδέσαμε τις εξόδους(Q[7..0]) του πρώτου καταχωρητή(A) ώστε να κρατάμε τον N-1 όρος της ακολουθίας(ουσιαστικά βάζουμε par_or_ser=0 και τα δεδομένα από τον πρώτο περνάνε παράλληλα στον δεύτερο).

Όσον αφορά τις εξόδους, έχουμε την έξοδο I[7..0] του N-1 όρου που προκύπτει από τον δεύτερο καταχωρητή(B), ενώ έχουμε την έξοδο F[7..0] του N όρου που προκύπτει από τον πρώτο καταχωρητή(A).

Το κύκλωμα φαίνεται στην παρακάτω εικόνα:



Αρχικά ενεργοποιούμε την σειριακή φόρτωση των δεδομένων θέτοντας το par_or_ser=1 για 16 χτύπους του ρολογιού και επιπλέον 1 στην αρχή που γίνεται και ο μηδενισμός του περιεχομένου των καταχωρητών.

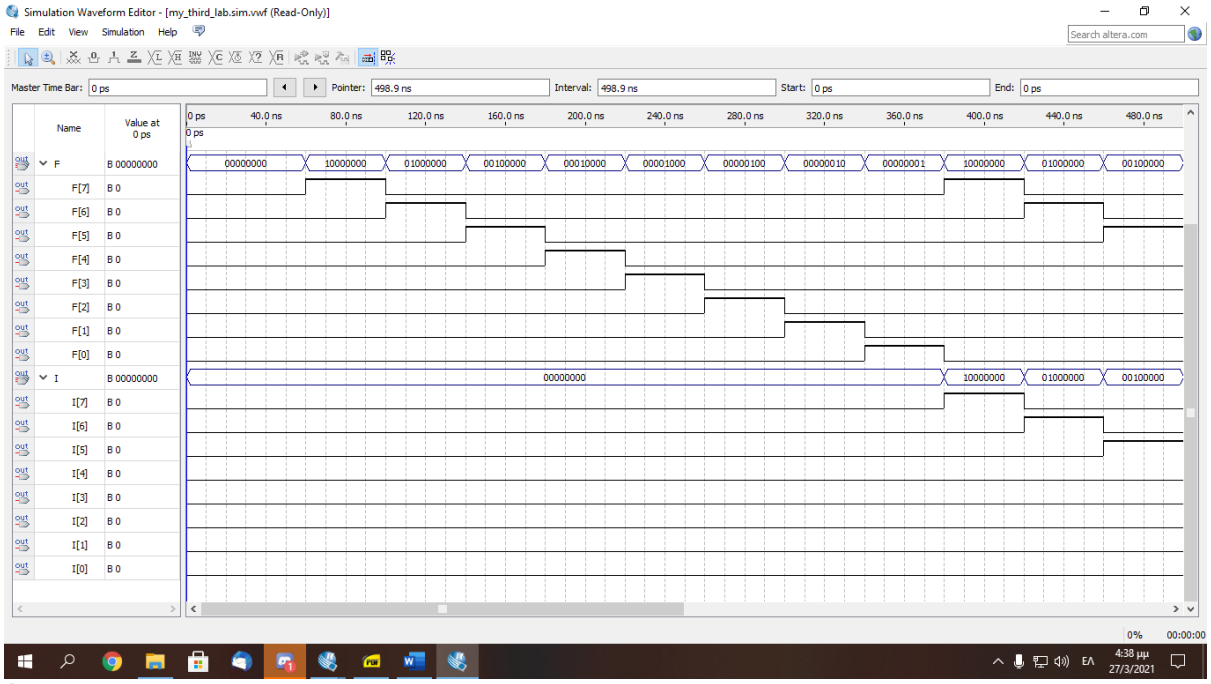
Στον δεύτερο χτύπο του ρολογιού βάζουμε το SERIAL_INPUT=1 για έναν χτύπο, στην συνέχεια βάζουμε το SERIAL_INPUT=0 και μετά πάλι στον ένατο χτύπο βάζουμε το SERIAL_INPUT=1 και τέλος μέχρι τον δέκατο έβδομο βάζουμε το SERIAL_INPUT=0.

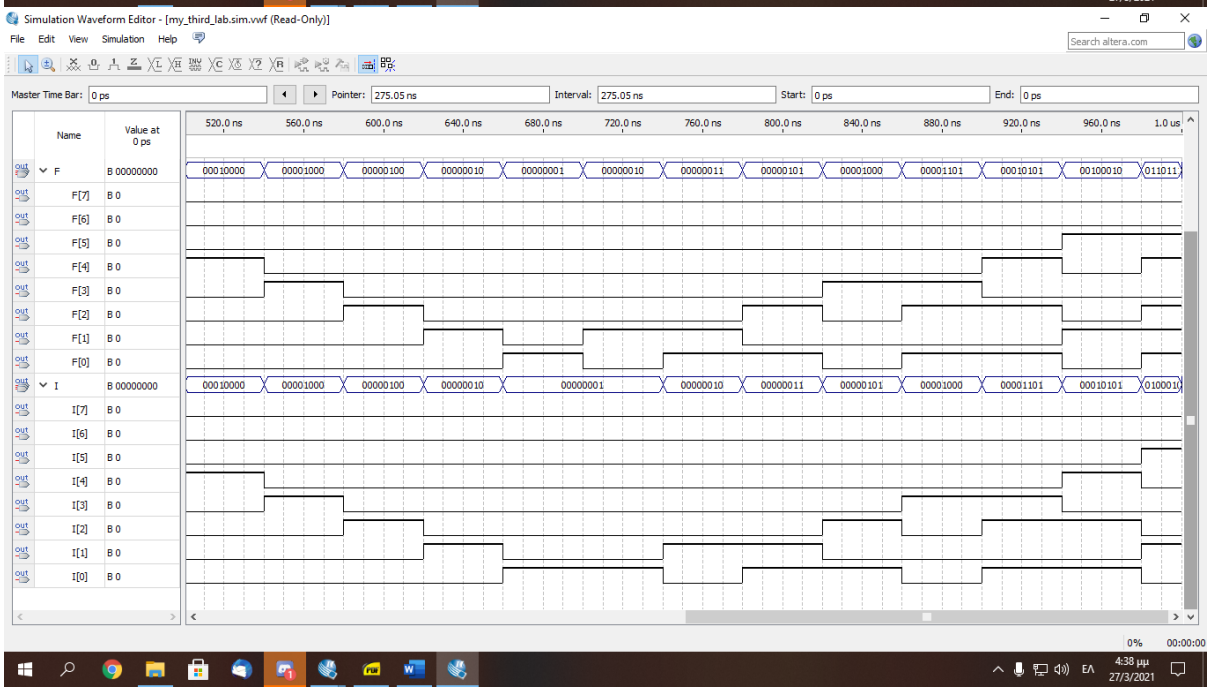
Αφού ολοκληρωθούν οι 17 χτύποι του ρολογιού θα έχει φορτωθεί και στους δύο καταχωρητές η τιμή 00000001.

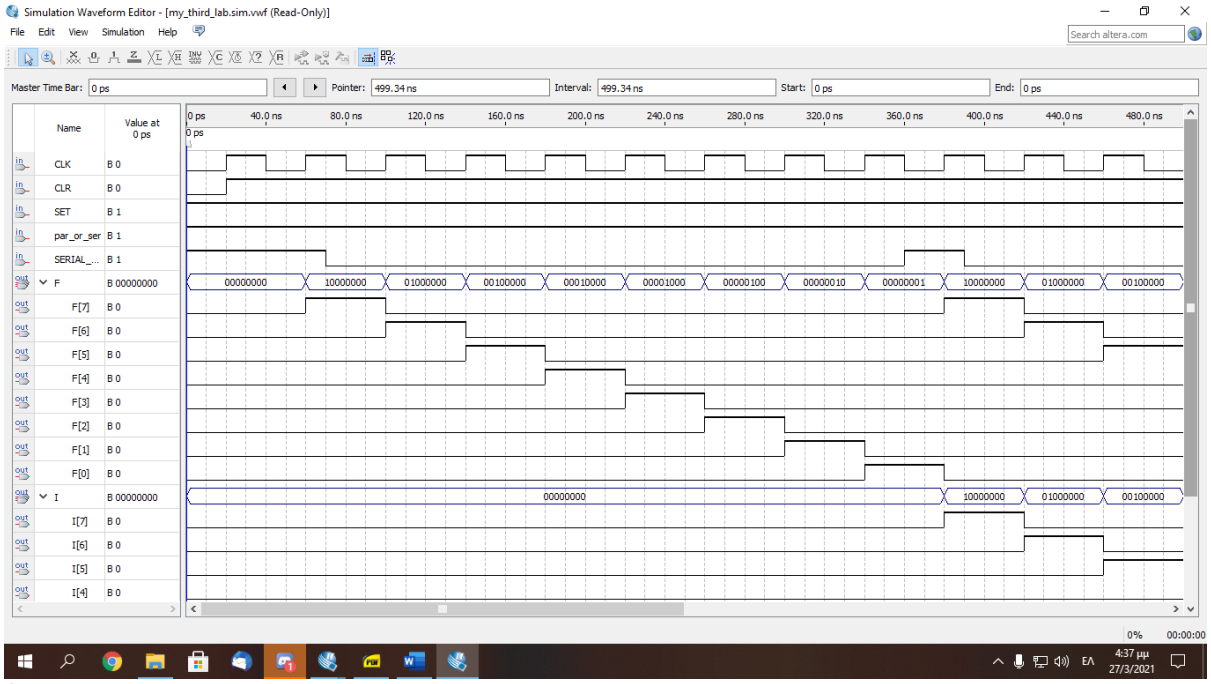
Στην συνέχεια βάζουμε τον διακόπτη par_or_ser=0 ώστε να ενεργοποιηθεί η παράλληλη φόρτωση των δεδομένων και να ξεκινήσει ο υπολογισμός της ακολουθίας.

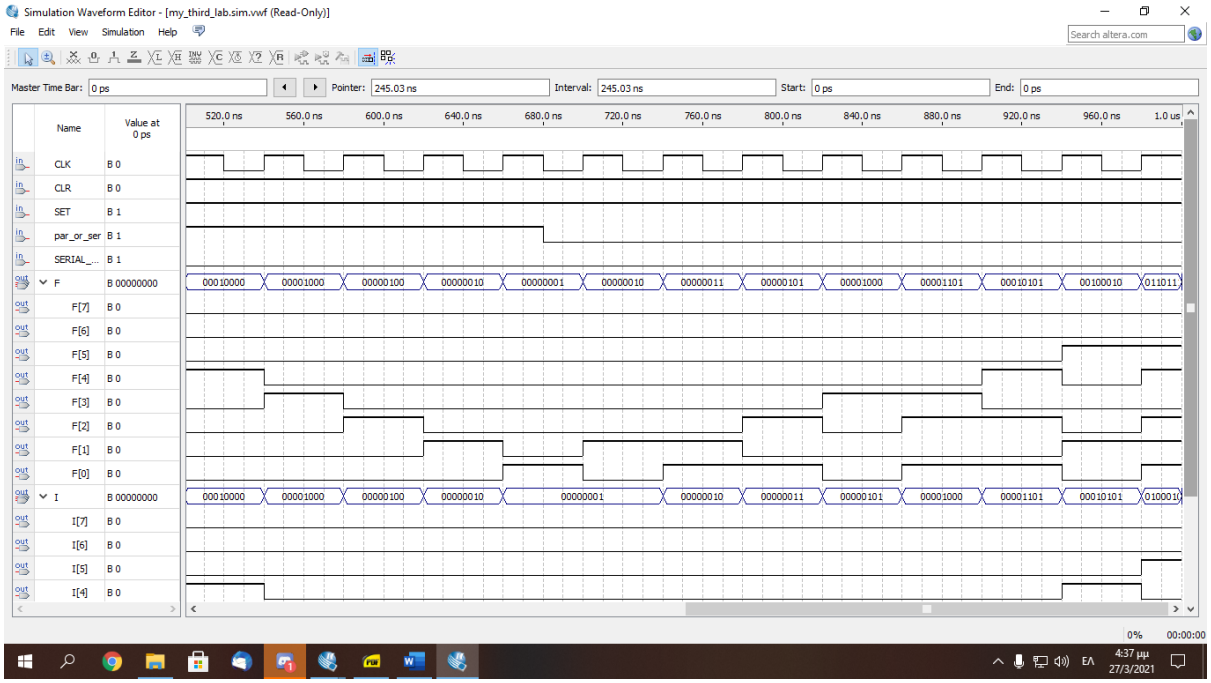
Εκτελώντας Functional Simulation, το κύκλωμα φαίνεται να λειτουργεί σωστά στις παρακάτω εικόνες:

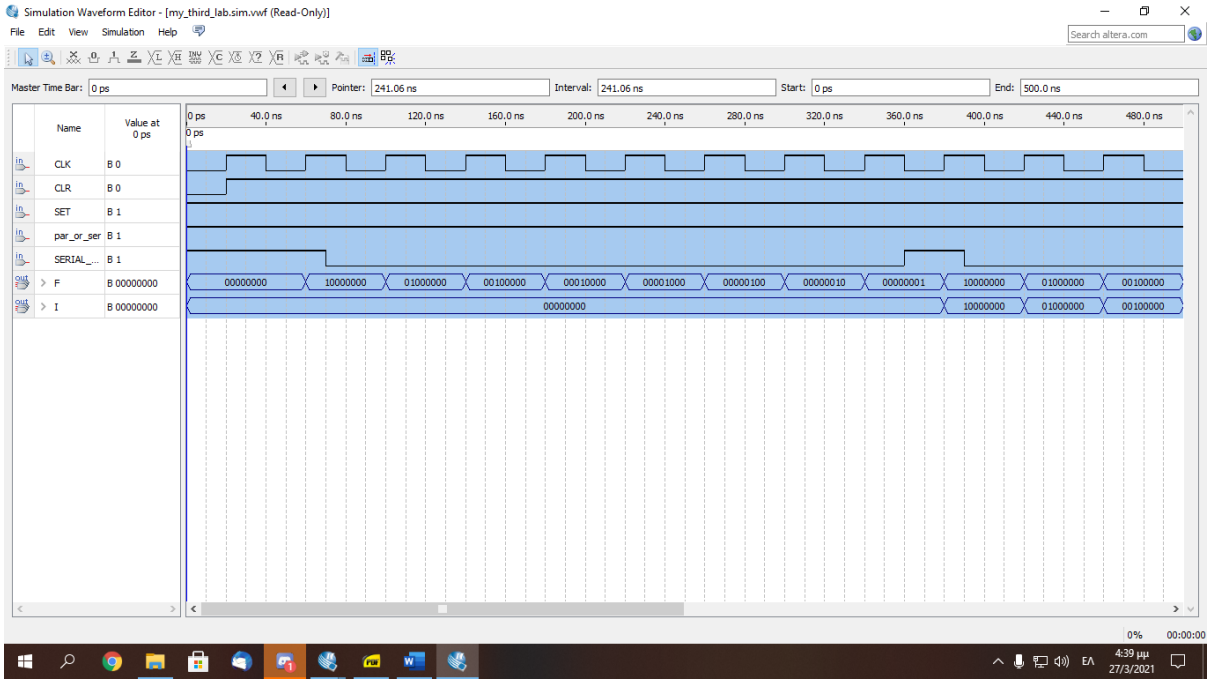
Στις πρώτες 6 εικόνες έχουμε επιλέξει **binary** αναπαράσταση των δεδομένων για να φαίνεται ξεκάθαρα η σειριακή φόρτωση, όπου η έξοδος F είναι ο N όρος της ακολουθίας και η έξοδος I είναι ο N-1 όρος της ακολουθίας.

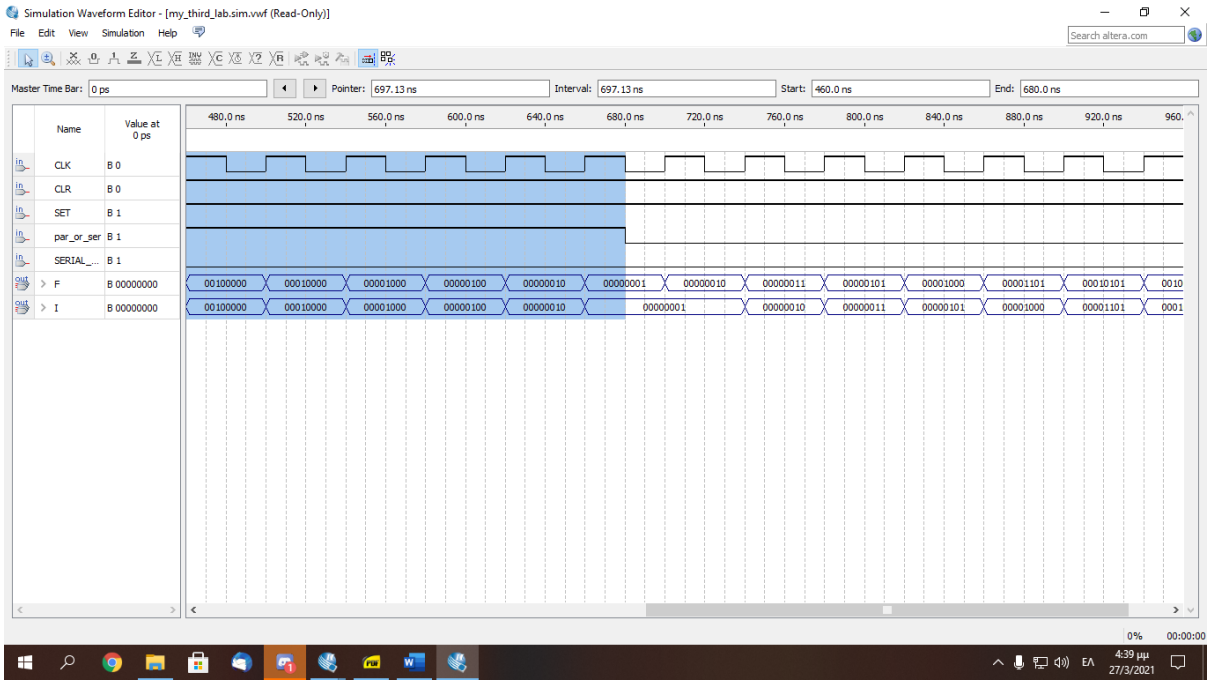




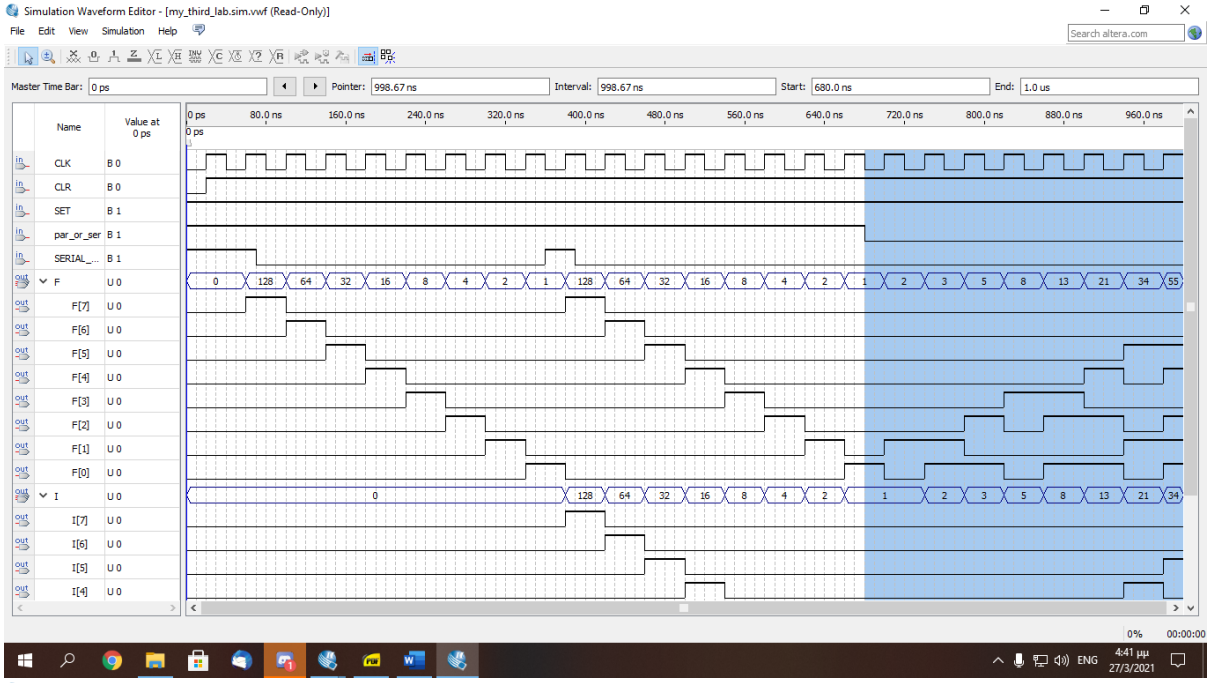

Simulation Waveform Editor - [my_third_lab.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 499.34 ns
Interval: 499.34 ns
Start: 0 ps
End: 0 ps
Name
Value at 0 ps
CLK B 0
CLR B 0
SET B 1
par_or_ser B 1
SERIAL_... B 1
F B 00000000
F[7] B 0
F[6] B 0
F[5] B 0
F[4] B 0
F[3] B 0
F[2] B 0
F[1] B 0
F[0] B 0
I B 00000000
I[7] B 0
I[6] B 0
I[5] B 0
I[4] B 0
0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns 440.0 ns 480.0 ns
00000000 10000000 01000000 00100000 00010000 00001000 00000100 00000010 00000001 10000000 01000000 00100000
00000000 10000000 01000000 00100000
0%
00:00:00
4:37 μμ
27/3/2021

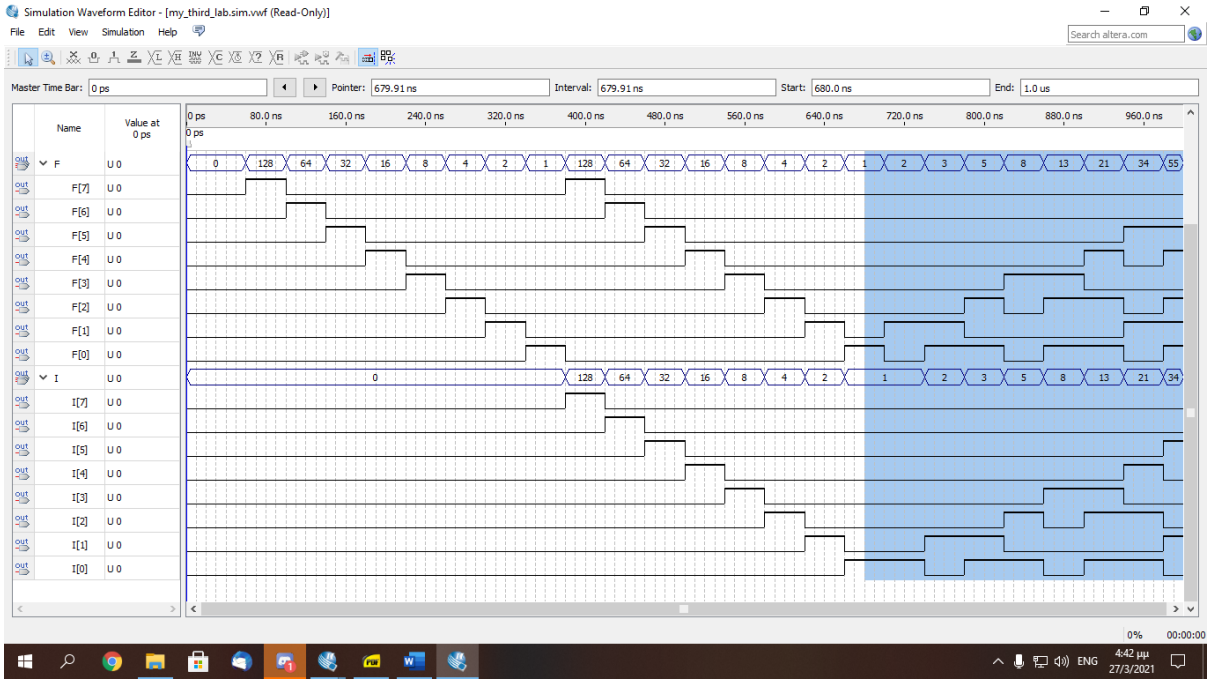

Simulation Waveform Editor - [my_third_lab.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 245.03 ns
Interval: 245.03 ns
Start: 0 ps
End: 0 ps
Name
Value at 0 ps
CLK B 0
CLR B 0
SET B 1
par_or_ser B 1
SERIAL_... B 1
F B 00000000
F[7] B 0
F[6] B 0
F[5] B 0
F[4] B 0
F[3] B 0
F[2] B 0
F[1] B 0
F[0] B 0
I B 00000000
I[7] B 0
I[6] B 0
I[5] B 0
I[4] B 0
520.0 ns 560.0 ns 600.0 ns 640.0 ns 680.0 ns 720.0 ns 760.0 ns 800.0 ns 840.0 ns 880.0 ns 920.0 ns 960.0 ns 1.0 us
00010000 00001000 00000100 00000010 00000001 00000010 00000011 00000101 00001000 00001101 00010101 00100010 011011
00010000 00001000 00000100 00000010 00000001 00000010 00000011 00000101 00001000 00001101 00010101
0%
00:00:00
4:37 μμ
27/3/2021


Simulation Waveform Editor - [my_third_lab.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 241.06 ns
Interval: 241.06 ns
Start: 0 ps
End: 500.0 ns
Name
Value at 0 ps
CLK B 0
CLR B 0
SET B 1
par_or_ser B 1
SERIAL_... B 1
F B 00000000
I B 00000000
0 ps 40.0 ns 80.0 ns 120.0 ns 160.0 ns 200.0 ns 240.0 ns 280.0 ns 320.0 ns 360.0 ns 400.0 ns 440.0 ns 480.0 ns
00000000 10000000 01000000 00100000 00010000 00001000 00000100 00000010 00000001 10000000 01000000 00100000
00000000 10000000 01000000 00100000
0%
00:00:00
4:39 μμ
27/3/2021


Simulation Waveform Editor - [my_third_lab.sim.vwf (Read-Only)]
File Edit View Simulation Help
Search altera.com
Master Time Bar: 0 ps
Pointer: 697.13 ns
Interval: 697.13 ns
Start: 460.0 ns
End: 680.0 ns
Name
Value at 0 ps
CLK B 0
CLR B 0
SET B 1
par_or_ser B 1
SERIAL_... B 1
F B 00000000
I B 00000000
480.0 ns 520.0 ns 560.0 ns 600.0 ns 640.0 ns 680.0 ns 720.0 ns 760.0 ns 800.0 ns 840.0 ns 880.0 ns 920.0 ns
00100000 00010000 00001000 00000100 00000010 00000001 00000010 00000011 00000101 00001000 00001101 00010101
00100000 00010000 00001000 00000100 00000010 00000001 00000010 00000011 00000101 00001000 00001101
0%
00:00:00
4:39 μμ
27/3/2021

Στις επόμενες 2 εικόνες έχουμε επιλέξει **unsigned decimal** αναπαράσταση των δεδομένων για να φαίνεται ξεκάθαρα η εκτέλεση της ακολουθίας, όπου η έξοδος F είναι ο N όρος της ακολουθίας και η έξοδος I είναι ο N-1 όρος της ακολουθίας.





Ο υπολογισμός της ακολουθίας ξεκινάει στον 17 χτύπο(1ος κύκλος fibonacci) του ρολογιού με N=1 και N-1=1

Στον επόμενο χτύπο του ρολογιού έχουμε N=2 και N-1=1 (2ος κύκλος fibonacci)

Στον επόμενο χτύπο του ρολογιού έχουμε N=3 και N-1=2 (3ος κύκλος fibonacci)

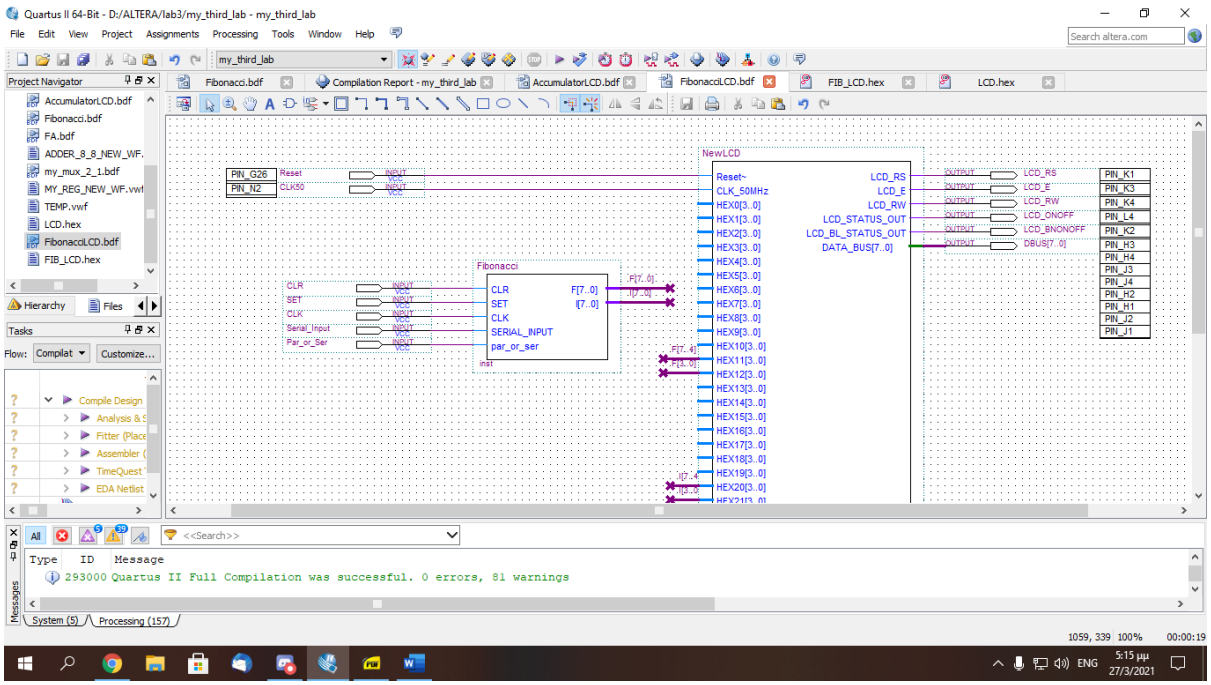
Στον επόμενο χτύπο του ρολογιού έχουμε N=5 και N-1=3 (4ος κύκλος fibonacci)

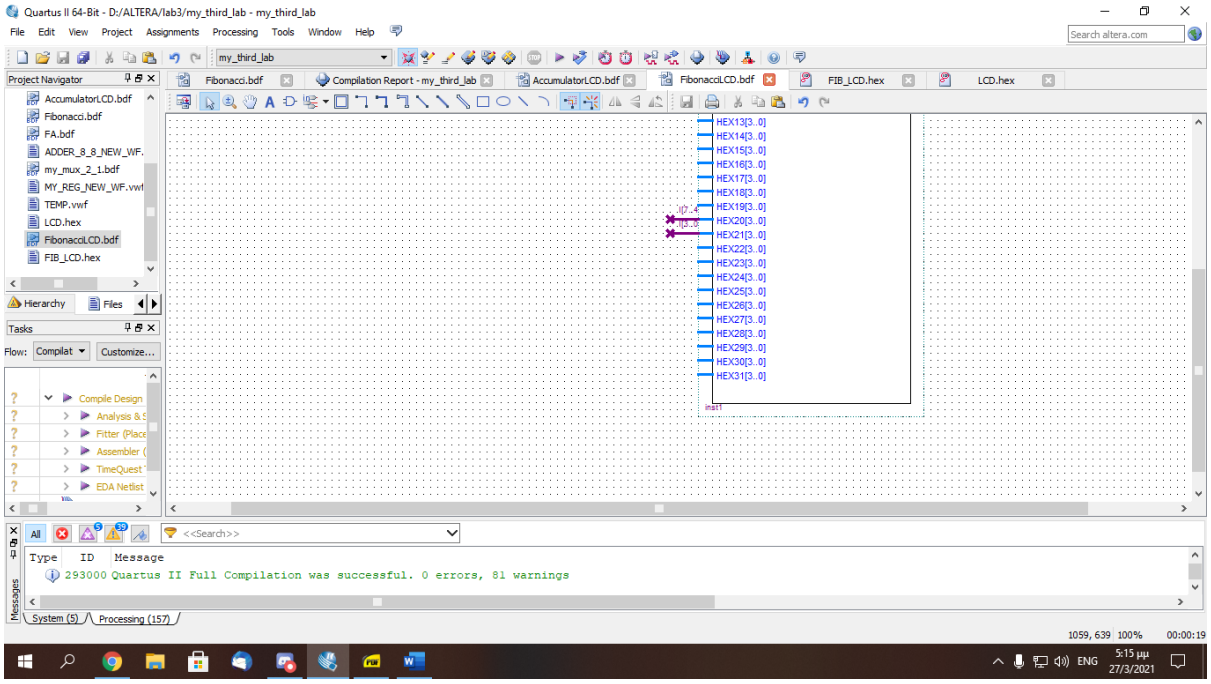
Στον επόμενο χτύπο του ρολογιού έχουμε N=8 και N-1=5 (5ος κύκλος fibonacci)

Στον επόμενο χτύπο του ρολογιού έχουμε N=13 και N-1=8 (6ος κύκλος fibonacci)

Στον επόμενο χτύπο του ρολογιού έχουμε N=21 και N-1=13(7ος κύκλος fibonacci)

Τέλος, πακετάραμε το κύκλωμα που υπολογίζει την ακολουθία και το συνδέσαμε με ένα LCD όπως φαίνεται στην παρακάτω εικόνα:





Και συμπληρώσαμε τις αντίστοιχες θέσεις μνήμης του LCD, όπως φαίνεται στην παρακάτω εικόνα: