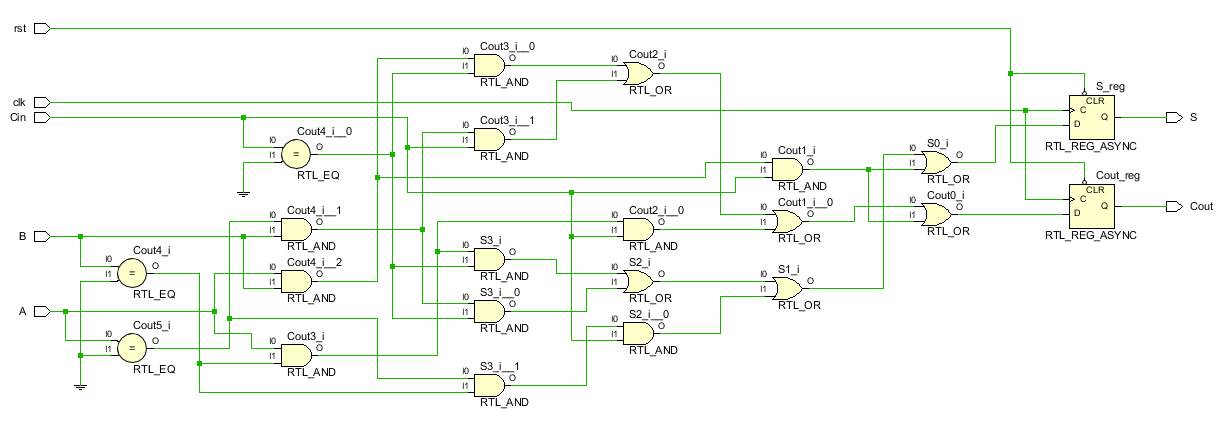
**3η Εργαστηριακή Άσκηση**

**VLSI**

Παντελαίος Δημήτριος Α.Μ.: 03118049

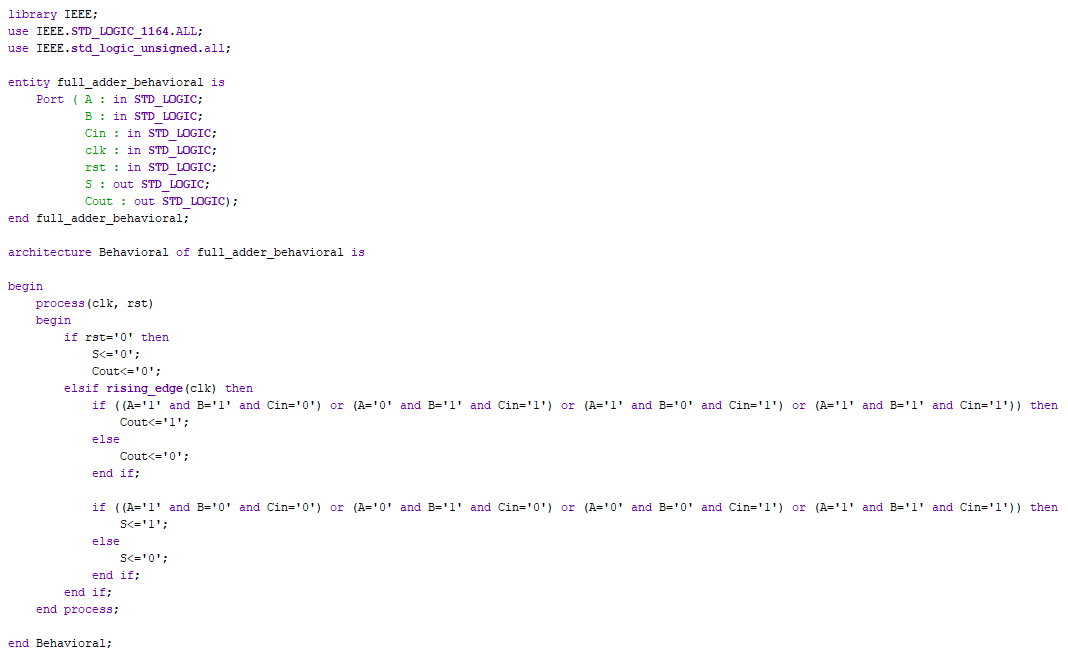
Μοίρας Αλέξανδρος Α.Μ.: 03118081

Ζήτημα 1:

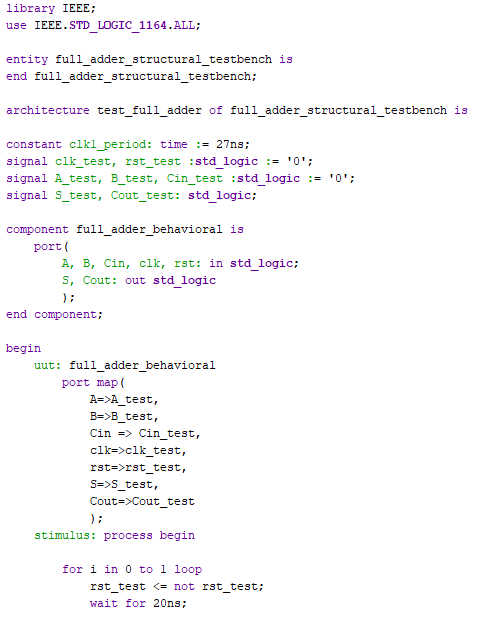
Το RTL schematic του σύγχρονου πλήρη αθροιστή που υλοποιήσαμε:

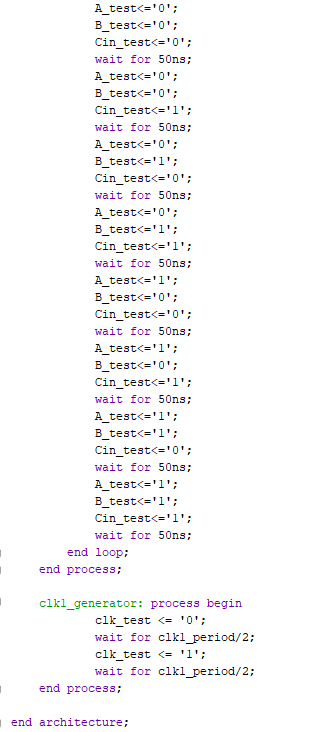
Πρόκειται για ένα ακολουθιακό κύκλωμα, η έξοδος του οποίου αποτελείται από το αποτέλεσμα και το κρατούμενο της πρόσθεσης και ανανεώνεται στην θετική ακμή του ρολογιού. Το αποτέλεσμα της πρόθεσης είναι ‘1’, αν λαμβάνει την τιμή ‘1’ περιττό πλήθος εκ των σημάτων εισόδου Α, Β, Cin, ενώ λαμβάνει την τιμή ‘0’ σε αντίθετη περίπτωση. Το κρατούμενο λαμβάνει την τιμή ‘1’, αν τουλάχιστον δύο από τα τρία σήματα εισόδου έχουν τιμή ‘1’. Ακόμη υπάρχει ένα σήμα reset, το οποίο όταν λαμβάνει την τιμή ‘0’ μηδενίζεται η έξοδος.

O VHDL κώδικας για την behavioral περιγραφή του σύγχρονου full adder:



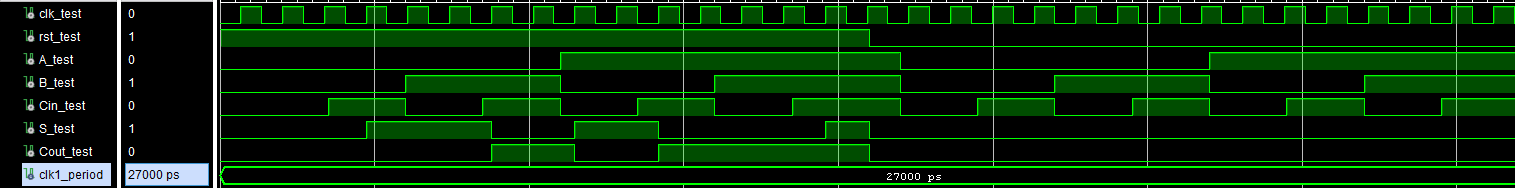
O VHDL κώδικας για τo testbench της behavioral περιγραφής του σύγχρονου full adder:

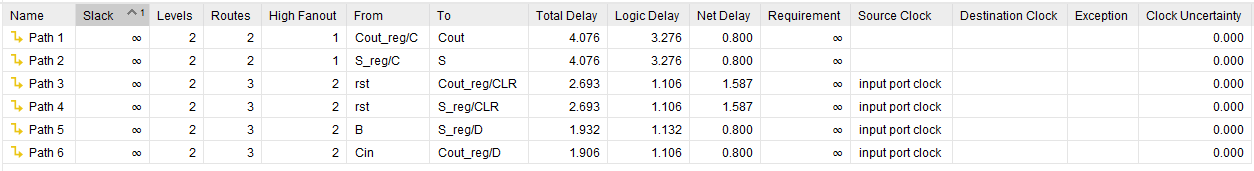




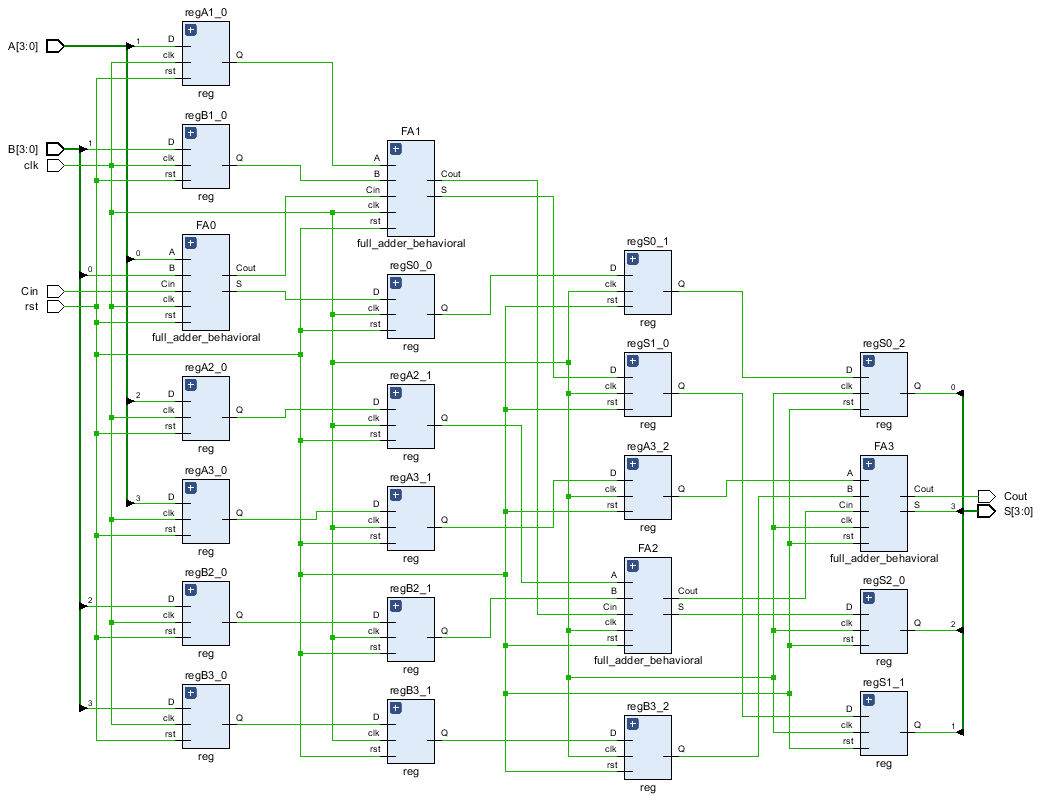
Στο testbench δοκιμάζουμε όλες τις δυνατές τιμές για τα 3 σήματα εισόδου.

Η προσομοίωση που επιβεβαιώνει την ορθή λειτουργία του κυκλώματός μας όπως περιεγράφη και παραπάνω:

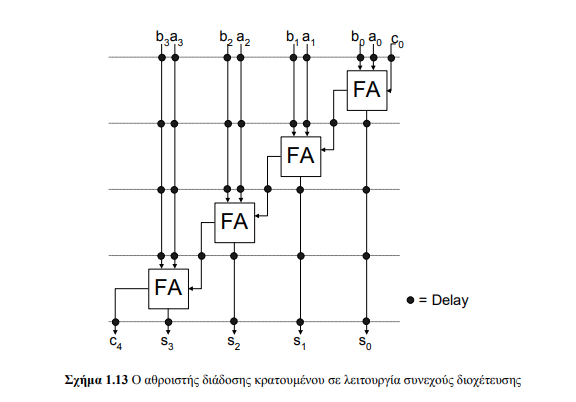


Στα ακολουθιακά κυκλώματα το κρίσιμο μονοπάτι προκύπτει από την έισοδο μέχρι κάποιον καταχωρητή ή από κάποιον καταχωρήτή μέχρι κάποιον άλλο ή από έναν καταχωρητή μέχρι την έξοδο.Το κρίσιμο μονοπάτι του κυκλώματος είναι από τους καταχωρητές Cout\_reg και S\_reg, που αποθηκεύεται το αποτέλεσμα της πρόσθεσης μέχρι την επόμενη θετική ακμή του ρολογιού προκειμένου να πάει στην έξοδο, μέχρι τις αντίστοιχες εξόδους Cout και S. Η χρονική του καθυστέρηση είναι 4.076ns.

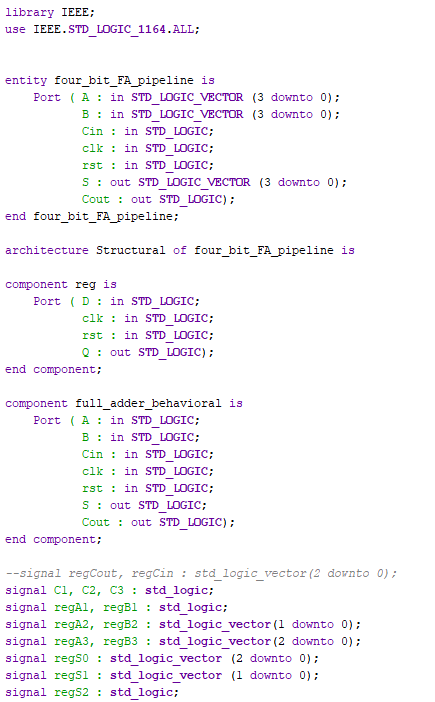
Ζήτημα 2:

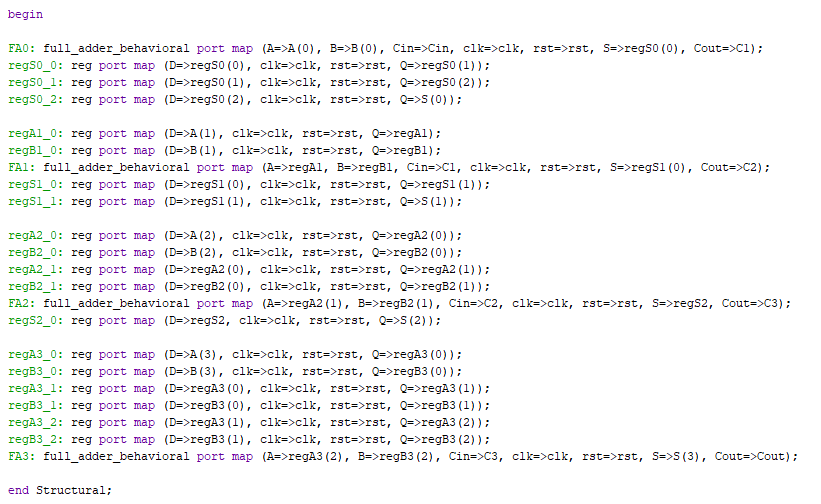
Το RTL schematic του σύγχρονου αθροιστή διάδοσης κρατουμένου των 4 bits που υλοποιήσαμε:

O παραπάνω σύγχρονος αθροιστής διάδοσης κρατουμένου των 4 bits υλοποιήθηκε με χρήση της τεχνικής pipeline. Διαθέτουμε 4 σύγχρονους full adders του ερωτήματος 1. Ο κάθε full adder δίνει το αποτέλεσμα του στην έξοδο και το κρατούμενο του στον επόμενο full adder. Για να ολοκληρωθεί μια πρόθεση δύο αριθμών 4 bit χρειάζονται 4 κύκλοι, κατά τη διάρκεια των οποίων ο κάθε full adder χρησιμοποιείται διαδοχικά μόνο για έναν κύκλο. Συνεπώς όταν ολοκληρώνεται από τον full adder η πρόσθεση των αντίστοιχων ψηφίων δύο αριθμών, μπορούμε να του δώσουμε τα αντίστοιχα ψηφία της επόμενης πρόσθεσης δύο αριθμών που θέλουμε. Για να συμβεί αυτό όμως θα πρέπει να έχει παραχθεί το κρατούμενο από τον προηγούμενο full adder. Για αυτό το λόγο προσθέτουμε κατάλληλο αριθμό καταχωρητών, προκειμένου τα bit Ai, Bi να εισάγονται στον full adder όταν έχει παραχθεί το κρατούμενο Ci-1. Aκόμη προσθέτουμε καταχωρητές στην έξοδο των full adder, προκειμένου το αποτέλεσμα τους να φτάνει ταυτόχρονα στην έξοδο. Οι καθυστερήσεις που αποτελούν έναν καταχωρητή στο κύκλωμα μας συμβολίζονται με τελεία στο παρακάτω σχήμα:

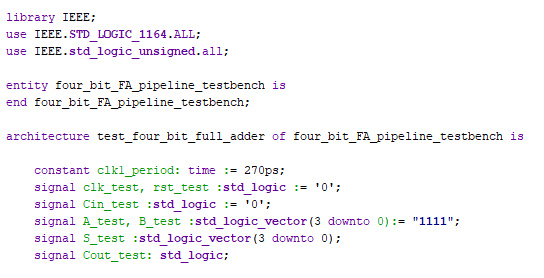


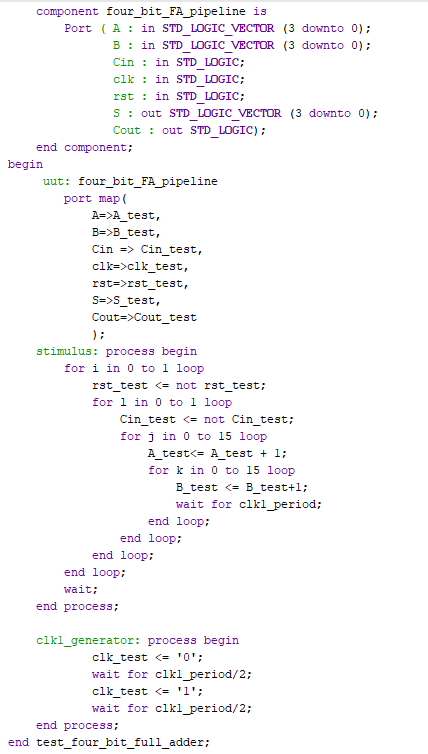
O VHDL κώδικας για την structural περιγραφή του σύγχρονου αθροιστή διάδοσης κρατουμένου των 4 bits με χρήση της τεχνικής Pipeline:



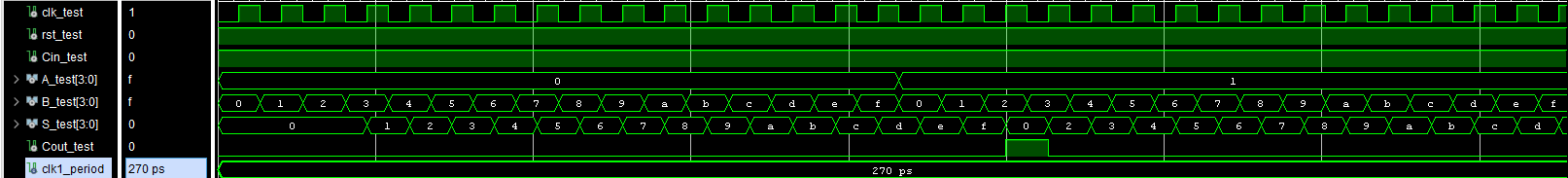


O VHDL κώδικας για το testbench της structural περιγραφής του σύγχρονου αθροιστή διάδοσης κρατουμένου των 4 bits με χρήση της τεχνικής Pipeline:

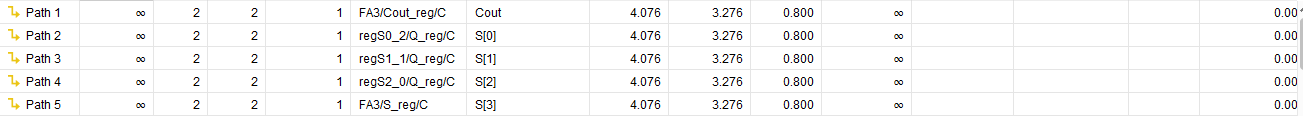




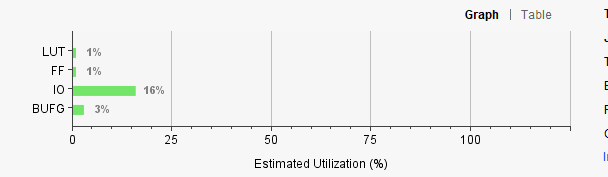
Στο testbench δοκιμάζουμε όλες τις δυνατές τιμές για τα 3 σήματα εισόδου.

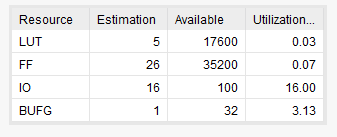
Ένα μέρος της προσομοίωσης που επιβεβαιώνει την ορθή λειτουργία του κυκλώματός μας όπως περιεγράφη και παραπάνω:

Η αρχική καθυστέρηση Tlatency είναι 4 κύκλοι ρολογιού μέχρι να παραχθεί το αποτέλεσμα της πρώτης πρόσθεσης που χρειάζεται 4 κύκλους. Μετά από αυτή την αρχική καθυστέρηση παράγεται ορθό αποτέλεσμα της αντίστοιχης πρόσθεσης σε κάθε κύκλο ρολογιού, καθώς χρησιμοποιήθηκε τεχνική pipeline και όταν ολοκληρώνεται μια πρόσθεση (από τον 4ο full adder), παράγεται ταυτόχρονα και το 3ο κρατούμενο της επόμενης πρόσθεσης από τον 3ο full adder και συνεπώς σε 1 κύκλο ολοκληρώνεται και αυτή η πρόσθεση.

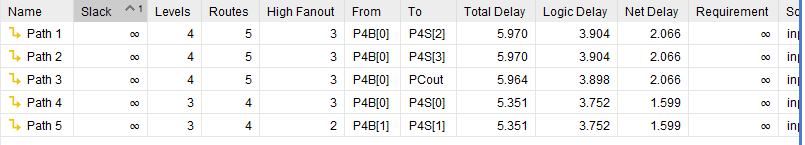
Το κρίσιμο μονοπάτι του κυκλώματος είναι για τους 3 πρώτους full adder από το τελευταίο στάδιο καταχωρήτων μέχρι το αποτέλεσμα να βγει στην έξοδο και για τον 4ο full adder από την στιγμή που παράγεται το αποτέλεσμα μέχρι να βγει στην έξοδο. Η χρονική του καθυστέρηση είναι 4.076ns.

H κατανάλωση πόρων είναι η εξής:

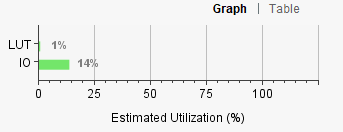
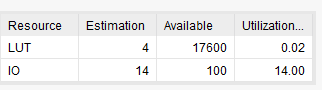




Θεωρώντας τον Παράλληλο Αθροιστή της εργαστηριακής άσκησης 2 ως ένα σύγχρονο κύκλωμα του οποίου η χρονική καθυστέρηση του critical path αντιστοιχεί στην περίοδο ενός μεγάλου κύκλου ρολογιού, η περίοδος αυτού του ρολογιού θα ήταν τουλάχιστον 5.970ns.

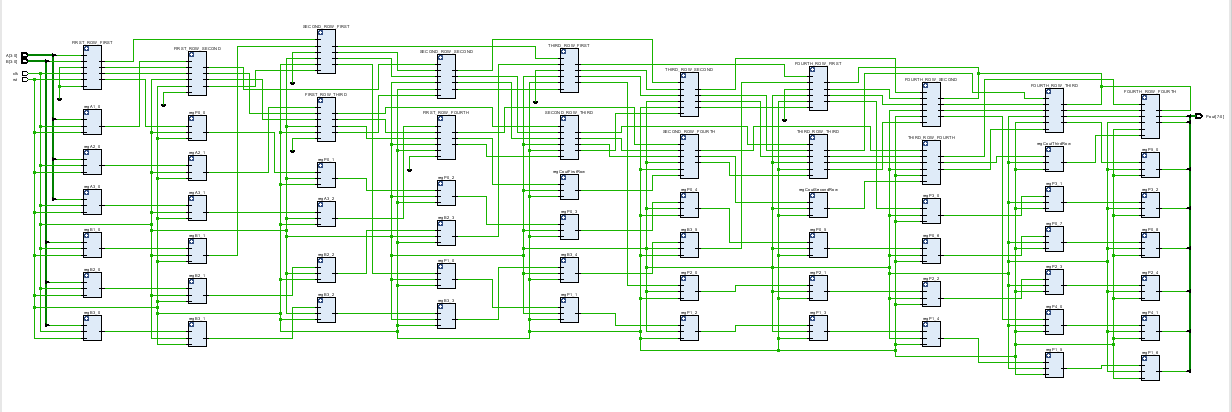
 Για τον παράλληλο αθροιστή με pipeline που υλοποιήσαμε τώρα το critical path έχει καθυστέρηση 4.076ns επομένως η περίοδός του θεωρητικά μπορεί να είναι μικρότερη του παράλληλου αθροιστή της άσκησης 2. Αυτό συμβαίνει διότι ο παράλληλος αθροιστής της άσκησης 2 δε χρησιμοποιεί pipeline οπότε άπαξ και του δοθεί είσοδος χρειάζεται να αναμένει όλες τις βαθμίδες να ολοκληρώσουν τον υπολογισμό τους και να δώσουν στοιχεία στις επόμενες ώστε να παράξει ορθό αποτέλεσμα κάτι που είναι χρονοβόρο. Εν αντιθέσει στον pipelined παράλληλο αθροιστή, όπως εξηγήθηκε παραπάνω, το κρίσιμο μονοπάτι προκύπτει από την είσοδο μέχρι κάποιον καταχωρητή ή από κάποιον καταχωρητή μέχρι κάποιον άλλο ή από έναν καταχωρητή μέχρι την έξοδο, αφού πρόκειται για ακολουθιακό κύκλωμα, οπότε η καθυστέρηση υπολογίζεται για διάδοση δεδομένων από ένα βήμα στο επόμενο από κύκλο σε κύκλο ρολογιού και αυτή η καθυστέρηση αποτελεί κάτω όριο για την περίοδο του ρολογιού. Είναι λογικό λοιπόν να είναι μικρότερη από αυτήν του απλού παράλληλου αθροιστή η οποία αφορά ολόκληρο τον υπολογισμό του αθροίσματος. Οπότε θεωρώντας τον Παράλληλο Αθροιστή της εργαστηριακής άσκησης 2 ως ένα σύγχρονο κύκλωμα του οποίου η χρονική καθυστέρηση του critical path αντιστοιχεί στην περίοδο ενός μεγάλου κύκλου ρολογιού θα δίνει ορθό αποτέλεσμα με αλλαγή της εισόδου σε κάθε κύκλο ρολογιού σε περισσότερο χρόνο από ότι ο pipelined παράλληλος αθροιστής.

Όσον αφορά την κατανάλωση πόρων για τον παράλληλο αθροιστή της εργαστηριακής άσκησης 2 βλέπουμε τα εξής:

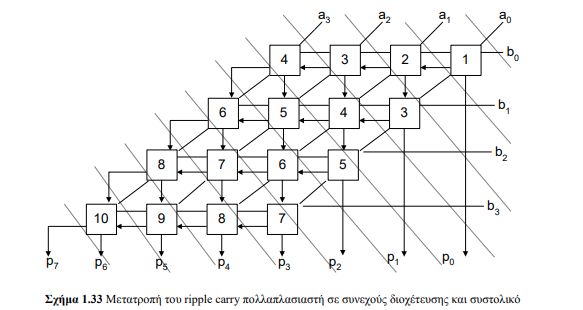
 

Παρατηρούμε ότι ο pipelined παράλληλος αθροιστής καταναλώνει σημαντικά περισσότερους πόρους, κάτι που είναι αναμενόμενο, καθώς χρειάζεται flip flops για τους registers που έχουμε εισάγει οι οποίοι αποτελούν επιπλέον hardware καθώς επίσης και global buffer για το ρολόι. Αλλαγές συνολικά στη δομή του κυκλώματος όπως η υλοποίηση των απλών πλήρων αθροιστών ως ακολουθιακά σύγχρονα κυκλώματα συμβάλουν και αυτές στην αύξηση της κατανάλωσης πόρων τόσο στα flip flops όσο και στα LUTs και IOs.

Ζήτημα 3:

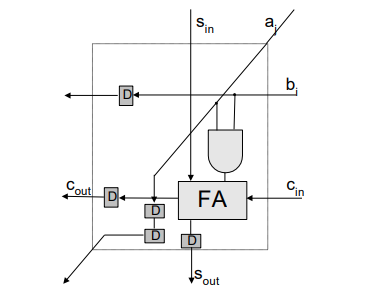
Το RTL schematic του συστολικού πολλαπλασιαστή διάδοσης κρατουμένων των 4 bits που υλοποιήσαμε:

Χρησιμοποιήθηκαν επιπλέον καταχωρητές έτσι ώστε οι είσοδοι για την κάθε δομική μονάδα να φτάνουν ταυτόχρονα στους κατάλληλους κύκλους ρολογιού όπως φαίνεται στο παρακάτω σχήμα, ώστε να υπολογίζεται ορθά το αποτέλεσμα. Ακόμη χρησιμοποιήθηκαν καταχωρητές, ώστε όλα τα bit του τελικού αποτελέσματος να φτάνουν ταυτόχρονα στην έξοδο.



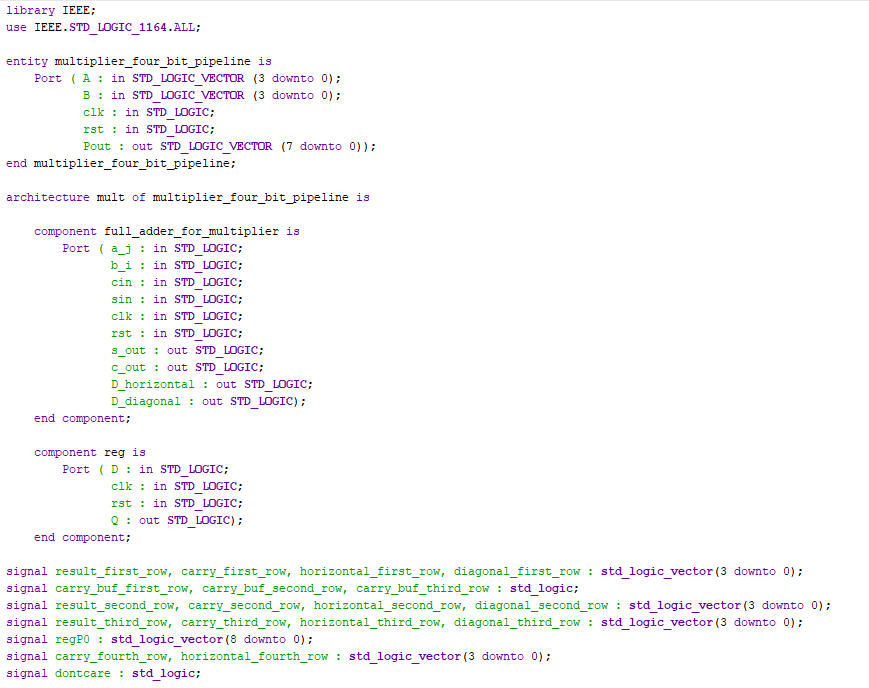
Οι αριθμοί πάνω στις δομικές μονάδες αντιπροσωπεύουν τον κύκλο στον οποίο πρέπει να υπολογίσει η καθεμία το αντίστοιχο αποτέλεσμα.

Η δομική μονάδα που χρησιμοποιήθηκε στο παραπάνω σχήμα είναι η παρακάτω αφαιρώντας του δύο καταχωρητές στην έξοδο του full adder, καθώς έγινε χρήση σύγχρονων full adder και δεν χρειάζεται η αποθήκευση του αποτελέσματος μέχρι την επόμενη θετική ακμή του ρολογιού:

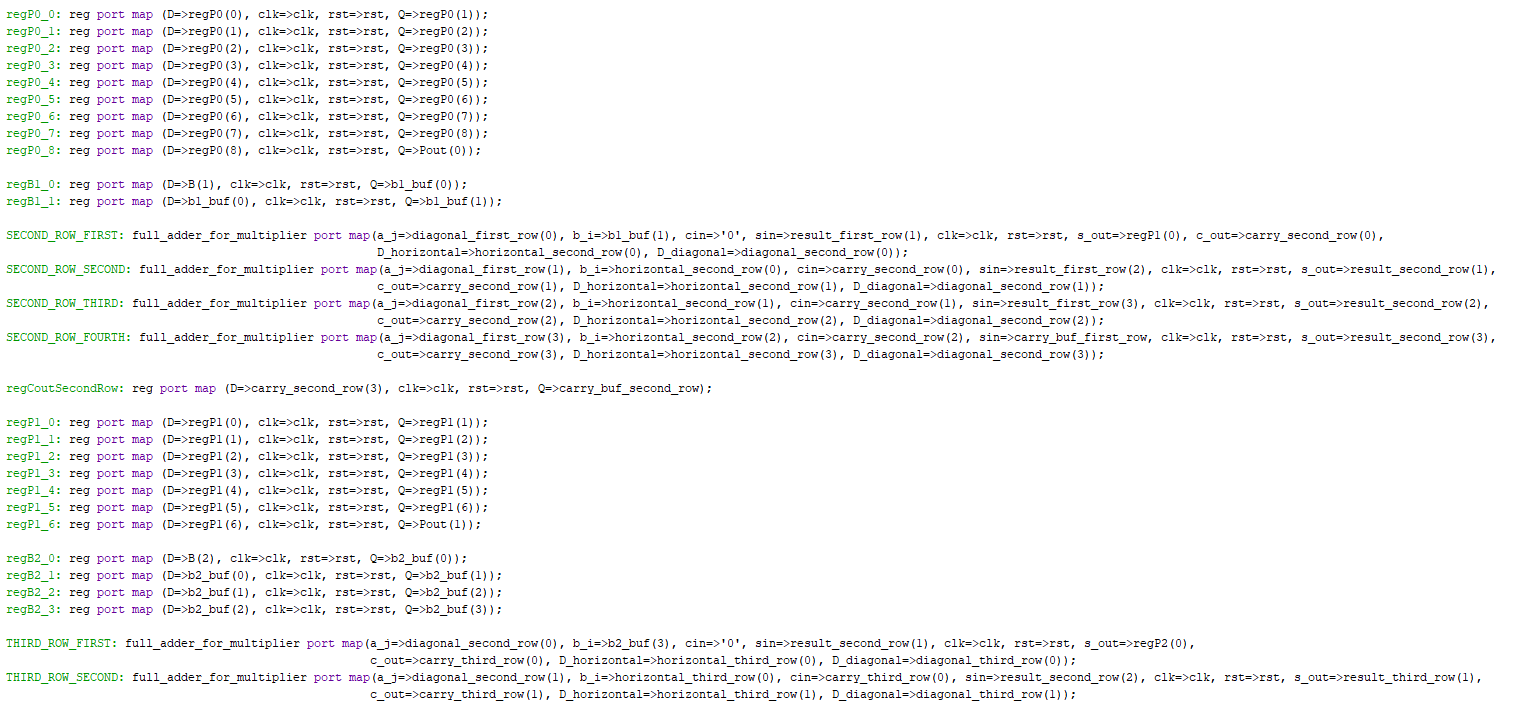


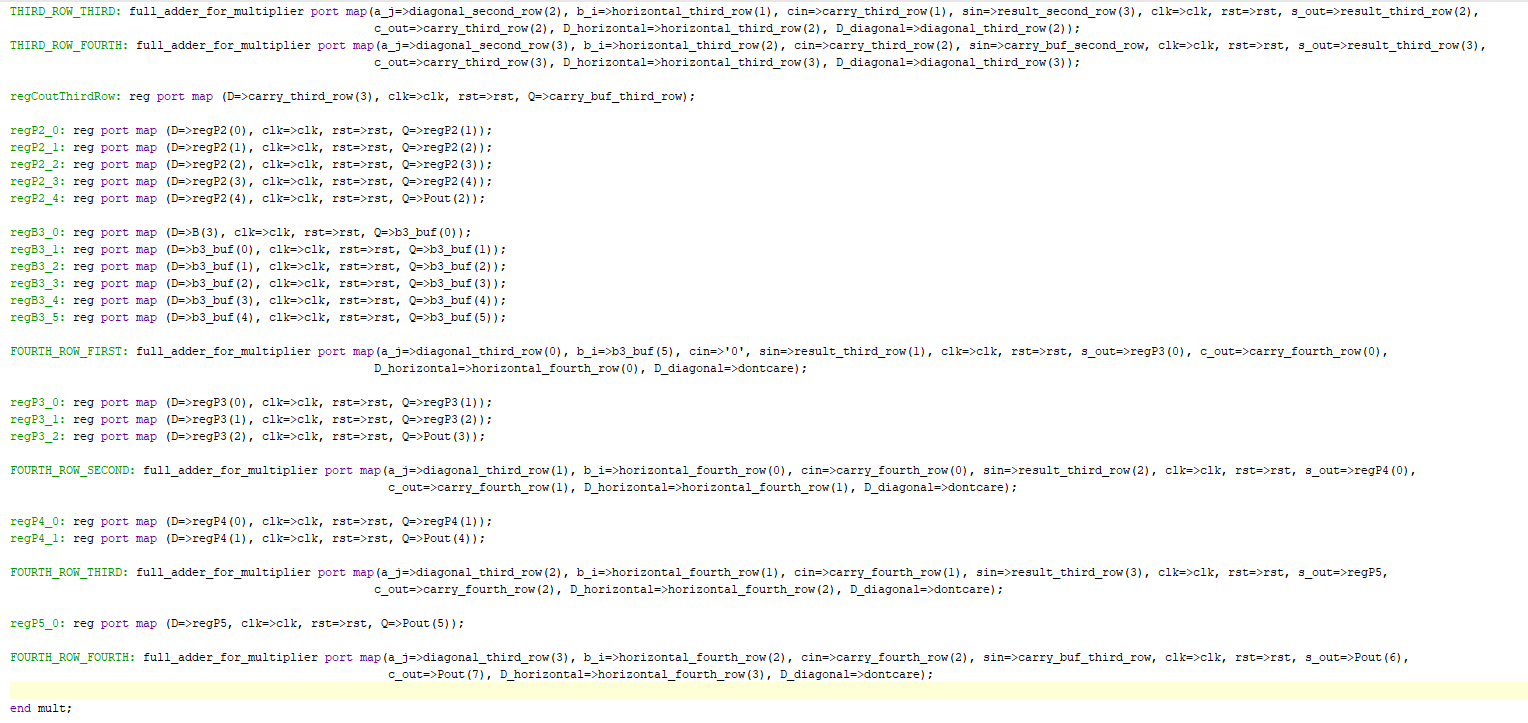
Κάθε γραμμή του σχήματος αποτελεί τον πολλαπλασιασμό ενός bit του Β με τον 4-bit A. Κατα τη διάρκεια του υπολογισμού γίνεται προώθηση του κρατουμένου και του bit του Β, ενώ παράλληλα προωθούνται το αποτέλεσμα του πολλαπλασιασμού, το τελικό κρατούμενο και ο αριθμός Α στην επόμενη γραμμή για τον επόμενο πολλαπλασιασμό.

O VHDL κώδικας για την structural περιγραφή του συστολικού πολλαπλασιαστή διάδοσης κρατουμένων των 4 bits:

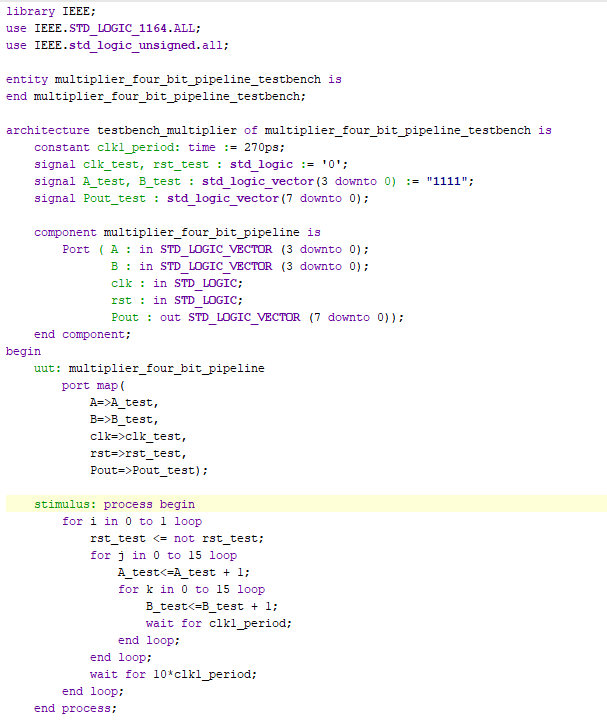


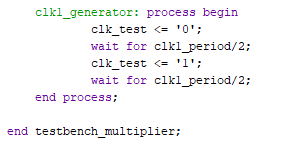


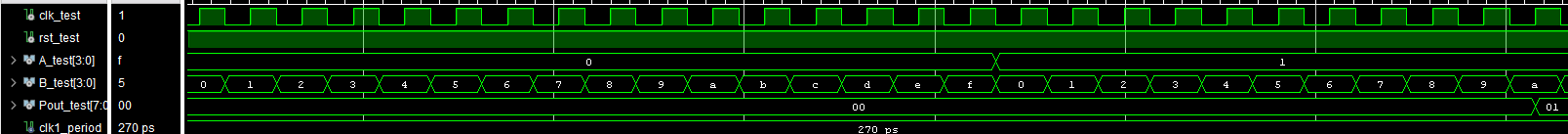
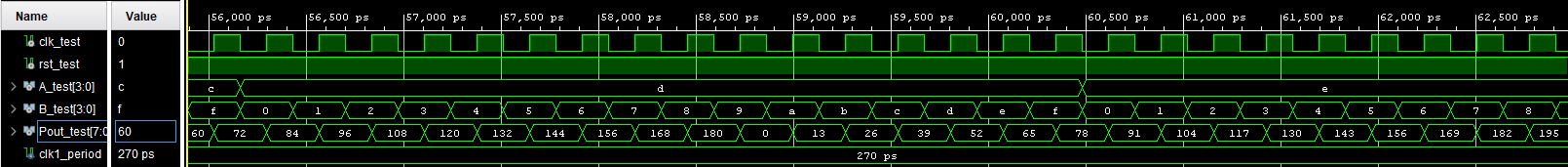




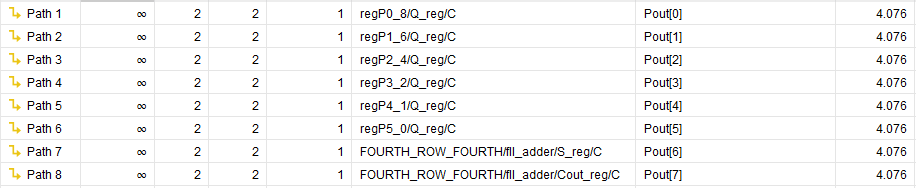
O VHDL κώδικας για το την testbench της structural περιγραφής του συστολικού πολλαπλασιαστή διάδοσης κρατουμένων των 4 bits:





Ένα μέρος της προσομοίωσης που επιβεβαιώνει την ορθή λειτουργία του κυκλώματός μας όπως περιεγράφη και παραπάνω:

Η αρχική καθυστέρηση Tlatency είναι 10 κύκλοι ρολογιού μέχρι να παραχθεί το αποτέλεσμα του πρώτου πολλαπλασιασμού που χρειάζεται 10 κύκλους. Μετά από αυτή την αρχική καθυστέρηση παράγεται ορθό αποτέλεσμα του αντίστοιχου πολλαπλασιασμού σε κάθε κύκλο ρολογιού, καθώς χρησιμοποιήθηκε τεχνική συστολικού pipeline και όταν παράγεται το αποτέλεσμα ενός πολλαπλασιασμόυ στον 10 κύκλο χρησιμοποιώντας μόνο την τελευταία δομική μονάδα, έχουν παραχθεί ήδη οι είσοδοι που θα δοθούν σε αυτή την δομική μονάδα για την ολοκλήρωση του επόμενου πολλαπλασιασμού.

Το κρίσιμο μονοπάτι του κυκλώματος είναι από τους τελικούς καταχωρητές ή την δομική μονάδα που χρησιμοποιείται στον τελευταίο κύκλο μέχρι την έξοδο. Η χρονική του καθυστέρηση είναι 4.076ns.