ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΫΠΟΛΟΓΙΣΤΩΝ & ΨΗΦΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ
ΑΚΑΔΗΜΑΪΚΟ ΕΤΟΣ 2020 - 2021

Πεγειώτη Νάταλυ
03117707

Γενικά Θέματα για το Μάθημα «ΨΗΦΙΑΚΑ ΣΥΣΤΗΜΑΤΑ VLSI»

4. α. Δίνεται ο αριθμός 1011111011 σε συμπλήρωμα ως προς 2. Να μετατραπεί ο αριθμός σε παράσταση Booth, Modified Booth και σε Canonic Signed Digit μορφή. Ποιο είναι το βασικό πλεονέκτημα της τελευταίας μορφής έναντι των υπολοίπων και της παράστασης συμπληρώματος ως προς 2; Στον πολλαπλασιασμό δυο αριθμών (n×k) που είναι σε παράσταση συμπληρώματος ως προς 2, το αποτέλεσμα μπορεί να παρασταθεί με n+k-1 bit εκτός από μία περίπτωση. Ποια είναι αυτή;

β. Να σχεδιαστεί με τις κατάλληλες απλοποιήσεις (να αξιοποιούνται όλες οι είσοδοι των πλήρων-αθροιστών της 1ης σειράς, βλ. 2ο θέμα) ένας παράλληλος πολλαπλασιαστής A·X (τύπου CS array) όπου το X είναι θετικός των 6 bit και A ο σταθερός θετικός αριθμός 111101. Στην συνέχεια να κάνετε χρήση της Κανονικής Παράστασης Προσημασμένου Ψηφίου (CSDR) για την κωδικοποίηση του αριθμού A, να δοθούν τα αντίστοιχα κυκλώματα που εδώ θα διαχειρίζονται και προσημασμένους αριθμούς. Υποδ.: Η αφαίρεση ενός όρου γίνεται λαμβάνοντας το συμπλήρωμά του ως προς 2 και όπου χρειάζεται να εφαρμόζεται η επέκταση προσήμου.

α. $1011111011 = (-261)_{10}$

Booth Encoding
00 => 0
01 => +1
10 => -1
11 => 0

Για τη μετατροπή του πιο πάνω αριθμό σε μορφή **Booth** προσθέτουμε στο τέλος του αριθμού ένα μηδενικό και στη συνέχεια με χρήση του παραπάνω πίνακα μετατρέπουμε τους αριθμούς ανά δύο ως ακολούθως:

1011111011<u>0</u>

$$10111110110 \Rightarrow \bar{1}10000\bar{1}10\bar{1}$$

Binary bits			Modified Booth's Digit
b_{2j+1}	b_{2j}	b_{2j-1}	
0	0	0	**0**
0	0	1	**+1**
0	1	0	**+1**
0	1	1	**+2**
1	0	0	**-2**
1	0	1	**-1**
1	1	0	**-1**
1	1	1	**0**

Για τη μετατροπή του πιο πάνω αριθμό σε μορφή **Modified Booth** προσθέτουμε στο τέλος του αριθμού ένα μηδενικό και στη συνέχεια με χρήση του παραπάνω πίνακα μετατρέπουμε τους αριθμούς ανά δύο ως ακολούθως:

10111110110

$$10111110110 => \mathbf{\bar{1}00\bar{1}\bar{1}}$$

Για τη μετατροπή του πιο πάνω αριθμό σε μορφή **Canonic Signed Digit** υπολογίζουμε τα ακόλουθα:

1η επανάληψη: $1011111011 => \bar{1}10000\bar{1}10\bar{1}$

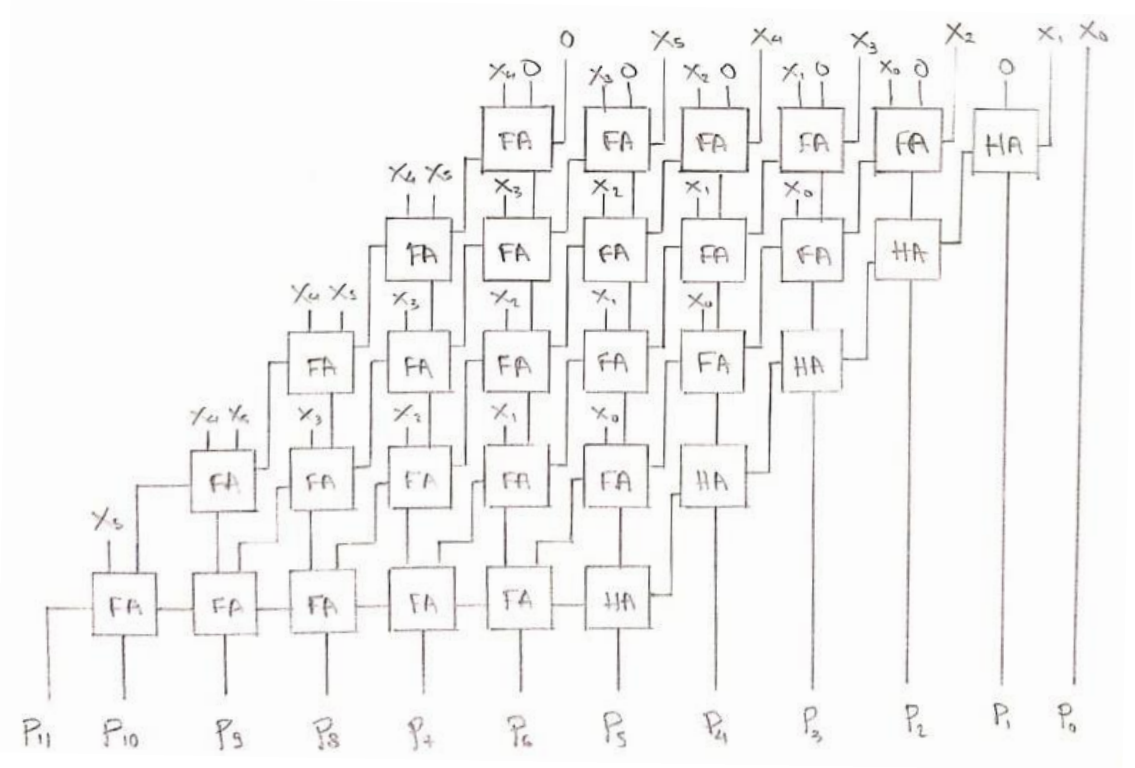
2η επανάληψη: $\bar{1}10000\bar{1}10\bar{1} => \mathbf{0\bar{1}00000\bar{1}0\bar{1}} = -2^8 - 2^2 - 2^0 = -256 - 4 - 1 = -261$

Το βασικό πλεονέκτημα της μορφής CSD έναντι των άλλων δύο είναι η σημαντική μείωση του αριθμό μη μηδενικών στοιχείων (1,-1). Επίσης είναι αδύνατη η ύπαρξη δύο μη μηδενικών στοιχείων δίπλα δίπλα, γεγονός που είναι πολύ χρήσιμο σε αριθμητικές πράξεις όπως στον πολλαπλασιασμό καθώς μειώνει την πολυπλοκότητα. Συγκεκριμένα, αν ο αριθμός που είναι σε CSD μορφή πολλαπλασιαστεί με ένα σταθερό αριθμό τότε το σύνολο των πράξεων θα ισούται με τον πλήθος των μη μηδενικών στοιχείων του αριθμού που είναι σε μορφη CSD.

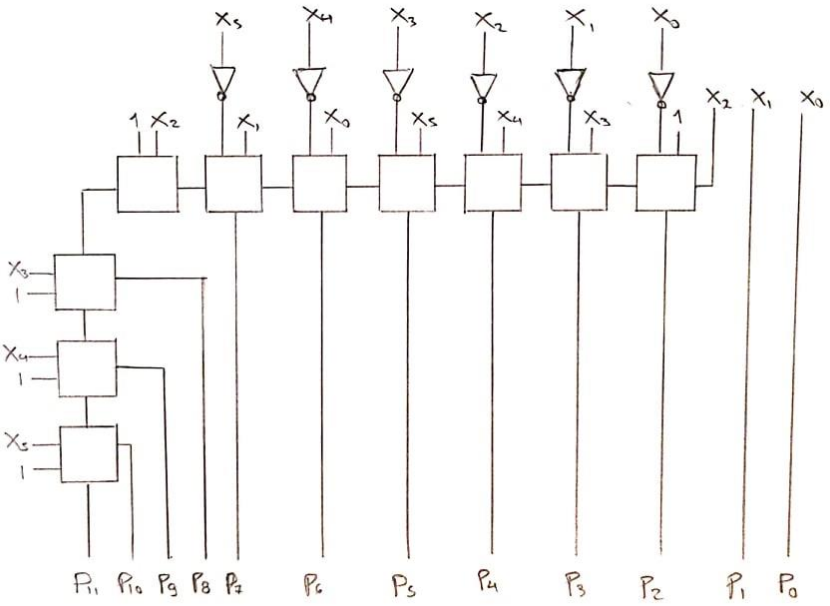
Το αποτέλεσμα δεν θα μπορεί να παρασταθεί με n+k-1 bit είναι όταν πολλαπλασιάζεις τον μεγαλύτερο αρνητικό αριθμό που αναπαρίσταται με n bits με τν μεγαλύτερο αρνητικό αριθμό που αναπαρίσταται με k bits. Θα υπάρχει υπερχείλιση για την συμπλήρωση του θετικού προσήμου.

β. Παράλληλος Πολλαπλασιαστής (τύπου CS array) A·X όπου X θετικός των 6 bit και A σταθερός θετικός αριθμός 111101.

Στο ακόλουθο σχήμα όπου υπάρχει X(i) κανονικά έχουμε μία πύλη AND μεταξύ του A(i) και X(i).



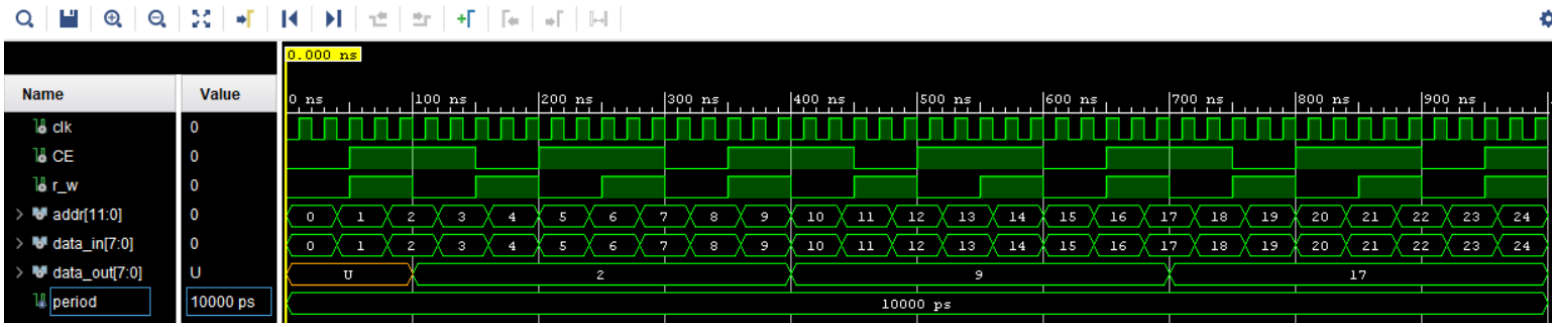
Σε CSDR είναι $A = 1000\bar{1}01$
Άρα το νέο κύκλωμα του παράλληλου πολλαπλασιαστή είναι:



7. Να δοθεί περιγραφή VHDL επιπέδου συμπεριφοράς (behavioral) ή μεταφοράς καταχωρητών (RTL) μιας σύγχρονης μνήμης RAM 4096×8bit που υποθέτουμε ότι λειτουργεί ως εξής: Στο θετικό μέτωπο του clk γίνεται η ανάγνωση (εισαγωγή) της διεύθυνσης addr και στο αρνητικό μέτωπο εισάγονται τα δεδομένα data_in για r/w=1 και για r/w=0 εξάγονται τα δεδομένα data_out. Για την εισαγωγή και εξαγωγή των δεδομένων πρέπει το σήμα CE=1(ασύγχρονη με το ρολόι).

Ο **source κώδικας** της άσκησης και το αντίστοιχο **testbench** βρίσκονται σε αρχεία VHDL στον φάκελο "Άσκηση 7".

Η προσομοίωση (**simulation**) του κυκλώματος φαίνεται πιο κάτω:



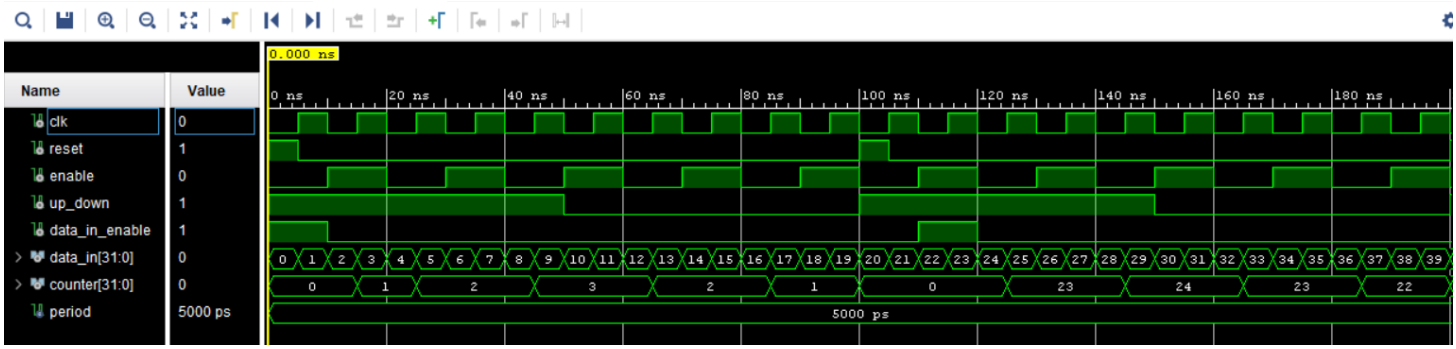
Στον θετικό παλμό του ρολογιού γίνεται η εισαγωγή της διεύθυνσης. Στον αρνητικό παλμό όταν r/w=1 γίνεται η εισαγωγή των δεδομένων data_in, ενώ για r/w=0 γίνεται η εξαγωγή data_out. Όλα αυτά πραγματοποιούνται μόνο υπό την προϋπόθεση ότι CE = 1.

8. Αναπτύξτε το κώδικα VHDL, ο οποίος να υλοποιεί ένα σύγχρονο μετρητή των 32 bit. Ο μετρητής αυτός θα πρέπει να διαθέτει τα εξής χαρακτηριστικά (να κάνετε επίσης όποιες παραδοχές θεωρείτε πως χρειάζονται):

- ***επαναφορά στην αρχική του τιμή (0) μέσω ενός σήματος "reset"=1 (ασύγχρονη με το ρολόι)***
- ***ενεργοποίηση της λειτουργίας μέτρησης μέσω ενός σήματος "enable"=1 (σύγχρονη με το ρολόι)***
- ***δυνατότητα μέτρησης προς τα πάνω ή κάτω (σήμα up/dn)***
- ***φόρτωση αρχικής (από το χρήστη) τιμής στο μετρητή, η οποία θα διαβάζεται από μια είσοδο data_in. Η απάντηση σας θα πρέπει να περιέχει το κώδικα VHDL με τα απαραίτητα σχόλια, το testbench, καθώς και ένα σύντομο report με ενδεικτικές καταστάσεις στις κυματομορφές εισόδου/εξόδου.***

Ο **source κώδικας** της άσκησης και το αντίστοιχο **testbench** βρίσκονται σε αρχεία VHDL στον φάκελο "Άσκηση 8".

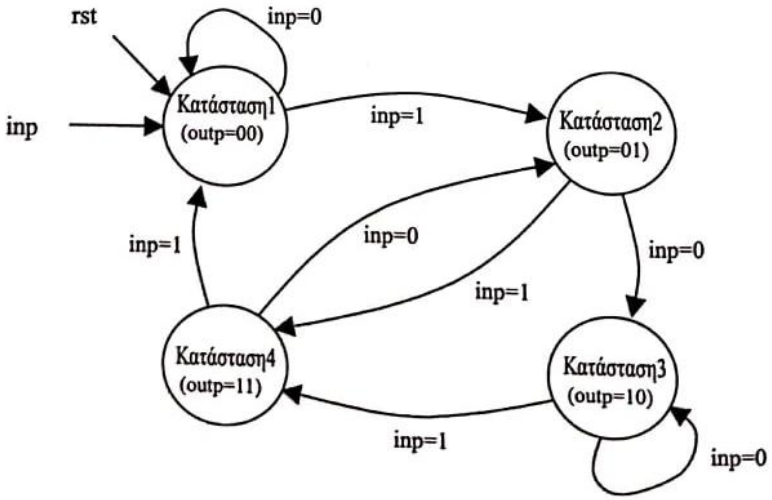
Η προσομοίωση (**simulation**) του κυκλώματος φαίνεται πιο κάτω:



**Το vector data_in είναι η είσοδος στην οποία φορτώνεται μια αρχική τιμή. Η υλοποίηση έγινε για διάφορα νούμερα εισόδου ώστε να παρατηρήσουμε καλύτερα την προσομοίωση.*
Η φόρτωση του data_in, γίνεται για την τιμή που εμφανίζεται στον κύκλο ρολογιού όπου το enable_in_data ισούται με 1.

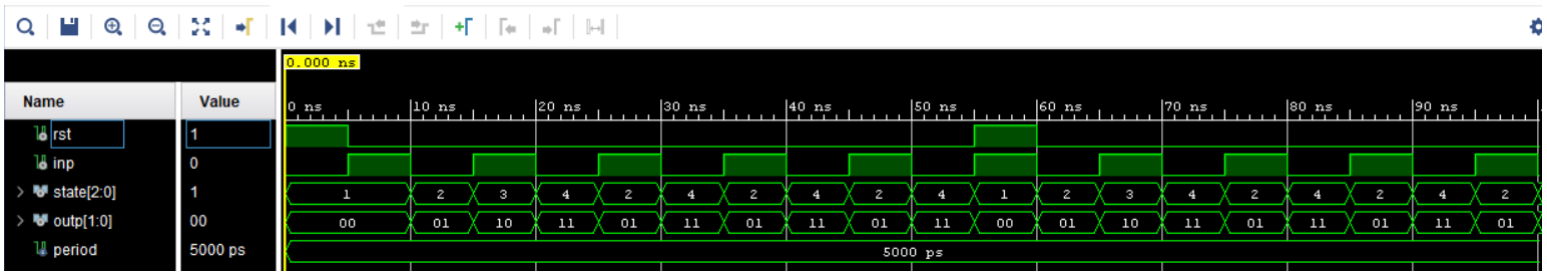
Παρατηρούμε ότι αρχικά πραγματοποιείται το reset επομένως ο counter παίρνει την τιμή 0. Στη συνέχεια καθώς up_down = 1 ο counter αυξάνεται, ενώ όταν όταν up_down = 0 ο counter μειώνεται. Ακολούθως πραγματοποιείται ξανά το reset. Όταν enable = 1 και enable_in_data = 1, στον counter φορτώνεται η τιμή του data_in η οποία αυξάνεται καθώς up_down = 1, ενώ μειώνεται όταν up_down = 0. Η προσομοίωση είναι ορθή και όλες οι ζητούμενες λειτουργίες υλοποιούνται.

9. Γράψτε κώδικα VHDL που να υλοποιεί την FSM που περιγράφεται από το διάγραμμα καταστάσεων της εικόνας.



Ο **source κώδικας** της άσκησης και το αντίστοιχο **testbench** βρίσκονται σε αρχεία VHDL στον φάκελο "Άσκηση 9".

Η προσομοίωση (**simulation**) του κυκλώματος φαίνεται πιο κάτω:

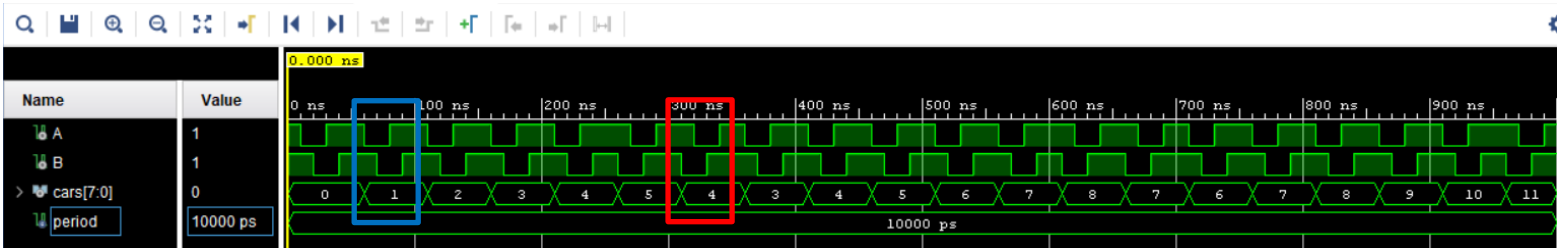


Αρχικά πραγματοποιείται το **rst** επομένως έχουμε state = 1 και outp = 00.
Με inp = 0 παραμένουμε στην ίδια κατάσταση. Με inp = 1 μεταβαίνουμε στην κατάσταση state = 2 και outp = 01, από την οποία με inp=0 μεταβαίνουμε στην κατάσταση state = 3 και outp = 10.
Παρατηρούμε ότι το μοτίβο συνεχίζει κανονικά, αντίστοιχα με την εικόνα, επομένως η FSM έχει υλοποιηθεί σωστά.

10. Να δοθεί περιγραφή VHDL επιπέδου συμπεριφοράς (behavioral) ή μεταφοράς καταχωρητών (RTL) του συστήματος ελέγχου ενός χώρου στάθμευσης με κοινή διέλευση εισόδου και εξόδου. Η πύλη διέλευσης είναι εφοδιασμένη με δύο φωτοευαίσθητους αισθητήρες Α και Β, οι οποίοι αποτελούν την είσοδο του συστήματος ελέγχου. Όταν ένα αυτοκίνητο περνάει από έναν αισθητήρα και διακόπτει τη ροή του φωτός, αυτός δίνει είσοδο 0, αλλιώς 1. Συνεπώς, κατά την είσοδο ενός αυτοκινήτου, το σύστημα ελέγχου δέχεται διαδοχικά τις εισόδους ΑΒ=11, 01, 00, 10, 11. Η έξοδος του συστήματος ελέγχου είναι ο αριθμός των αυτοκινήτων που υπάρχουν στο χώρο στάθμευσης, ο οποίος είναι αρχικά 0. Περιπτώσεις όπου οι τιμές των αισθητήρων δεν έχουν νόημα (π.χ. από 11 σε 00) να θεωρούνται σφάλματα και να αγνοούνται. Υποθέτουμε ότι κάθε φορά μόνο ένα αυτοκίνητο μπορεί να περνάει.

Ο **source κώδικας** της άσκησης και το αντίστοιχο **testbench** βρίσκονται σε αρχεία VHDL στον φάκελο “Άσκηση 10”.

Η προσομοίωση (**simulation**) του κυκλώματος φαίνεται πιο κάτω:



Κάθε φορά που παρατηρείται το μοτίβο που φαίνεται στο μπλε περίγραμμα (11 -> 01 -> 00 -> 10 -> 11), ο αριθμός των οχημάτων αυξάνεται κατά 1 αφού τότε εισέρχεται ακόμη ένα όχημα στον χώρο στάθμευσης.
Κάθε φορά που παρατηρείται το μοτίβο που φαίνεται στο κόκκινο περίγραμμα (11 -> 10 -> 00 -> 01 -> 11), ο αριθμός των οχημάτων μειώνεται κατά 1 αφού τότε εξέρχεται ένα όχημα από τον χώρο στάθμευσης.

RTL schematic