Σχεδιασμός Ολοκληρωμένων Συστημάτων με τεχνικές VLSI

# PROJECT

# Floating Point Adder

ΟΜΑΔΑ 5

Αρχοντά Χριστίνα 7219

Παπαϊωάννου Μαγδαληνή 7359

## ΣΚΟΠΟΣ ΤΗΣ ΕΡΓΑΣΙΑΣ

Σκοπός της εργασίας ήταν να συνδυάσουμε όλες τις γνώσεις σχεδιασμού ολοκληρωμένων κυκλωμάτων που έχουμε αποκτήσει μέχρι τώρα και να σχεδιάσουμε ένα πολύπλοκο κύκλωμα, έναν αθροιστή κινητής υποδιαστολής.

Η αναπαράσταση κινητής υποδιαστολής χρησιμοποιείται σε πληθώρα εφαρμογών και αυτό έχει ως αποτέλεσμα την ενσωμάτωση τέτοιων αθροιστών σε διάφορα ενσωματωμένα κυκλώματα.

Αναλύσαμε λοιπόν θεωρητικά την αναπαράσταση κινητής υποδιαστολής και την πράξη της πρόσθεσης με αριθμούς τέτοιου είδους αναπαράστασης, και στη συνέχεια σχεδιάσαμε έναν δικό μας αθροιστή, χρησιμοποιώντας την γλώσσα περιγραφής υλικού VHDL. Τέλος χρησιμοποιήσαμε τη σουίτα ISE της Xilinx για την σύνθεση του σχεδιασμού του υλικού σε πλατφόρμες αναδιατασσόμενης λογικής (FPGA).

## ΤΟ ΠΡΟΤΥΠΟ IEEE 754

Οι αριθμοί κινητής υποδιαστολής αναπαρίστανται σύμφωνα με το πρότυπο IEEE standard 754 floating point, το οποίο έχει επικρατήσει σήμερα για τους πραγματικούς αριθμούς.

Το πρότυπο ορίζει τον τρόπο αναπαράστασης των αριθμών κινητής υποδιαστολής, τον τρόπο αποθήκευσης και αποστολής τους, κανόνες στρογγυλοποίησης, πράξεις μεταξύ τους, καθώς και τρόπους αντιμετώπισης των exceptions.

Το IEEE 754 format μπορεί να αναπαραστήσει:

- Πεπερασμένους αριθμούς, δυαδικούς ή δεκαδικούς. Κάθε αριθμός περιγράφεται από 3 ακεραίους: το πρόσημο (s), τον κλάσμα (c ), και τον εκθέτη (q).
- Το θετικό και αρνητικό άπειρο.
- NaN: Not a number.

Η κωδικοποίηση των αριθμών κινητής υποδιαστολής για το IEEE 754 standard μονής και διπλής ακρίβειας φαίνονται στον παρακάτω πίνακα:

| Μονή Ακρίβεια | | Διπλή Ακρίβεια | | Αντικείμενο που συμβολίζεται |
|---|---|---|---|---|
| *Εκθέτης* | *Κλάσμα* | *Εκθέτης* | *Κλάσμα* | |
| 0 | 0 | 0 | 0 | 0 |
| 0 | Μη μηδενικό | 0 | Μη μηδενικό | +- μη κανονικοποιημένος αριθμός |
| 1-254 | Οτιδήποτε | 1-2046 | Οτιδήποτε | +- αριθμός κινητής υποδιαστολής |
| 255 | 0 | 2047 | 0 | +- άπειρο |
| 255 | Μη μηδενικό | 2047 | Μη μηδενικό | NaN (όχι αριθμός) |

Παρατηρούμε ότι οι τιμές «0000…0000» και «1111…1111» του εκθέτη δεσμεύονται, για την αναπαράσταση των ειδικών συμβόλων 0, άπειρο και NaN, καθώς και των μη κανονικοποιημένων αριθμών.

Η αριθμητική τιμή του κάθε πεπερασμένου αριθμού υπολογίζεται απ' τον τύπο:
τιμή = $(-1)^s \times c \times b^q$
όπου b η βάση του αριθμητικού συστήματος στο οποίο αναπαρίσταται ο αριθμός (δεκαδικό ή δυαδικό).

Το εύρος των τιμών που μπορούν να αναπαρασταθούν από το πρότυπο καθορίζεται από το αριθμητικό σύστημα, καθώς και από των αριθμό των ψηφίων που αντιπροσωπεύουν το κλάσμα, και τον εκθέτη. Το μέγεθος του εκθέτη καθορίζει τη μέγιστη απόλυτη τιμή των αριθμών που μπορούν να αναπαρασταθούν, ενώ το μέγεθος του κλάσματος καθορίζει τη μέγιστη δυνατή ακρίβεια.

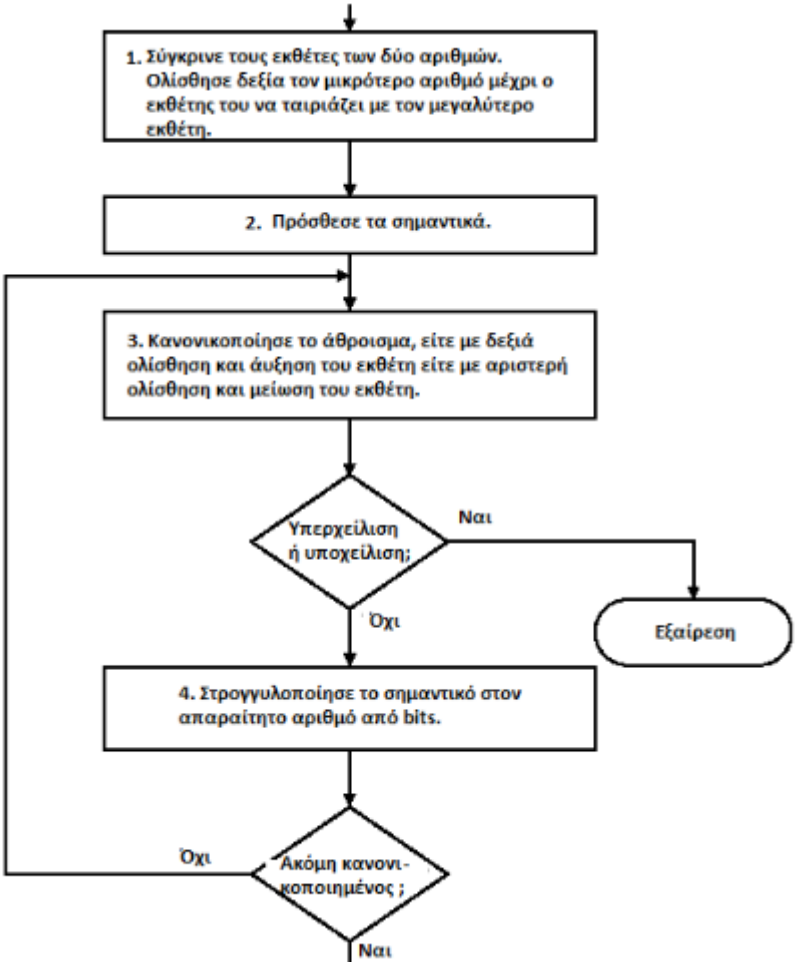
Το πρότυπο ορίζει πέντε βασικά φορμάτ αριθμών, τρία για το δυαδικό και δύο για το δεκαδικό σύστημα. Τα: binary32, binary64, binary128, decimal32, decimal64. Το όνομα του καθενός καθορίζει το σύστημα της αναπαράστασης, καθώς και τον αριθμό των ψηφίων του κάθε αριθμού. Τα επικρατέστερα εξ αυτών είναι τα binary32, ή αλλιώς single precision floating point, και binary64, ή double precision floating point.

Για να επιτευχθεί μεγαλύτερη ακρίβεια, η αναπαράσταση που χρησιμοποιούμε είναι πάντα κανονικοποιημένη. Δηλαδή υπάρχει πάντα ένα ψηφίο πριν την υποδιαστολή. Μ' αυτόν τον τρόπο, το πρώτο ψηφίο υπονοείται και μπορούμε να το παραλείπουμε από την αναπαράσταση του κλάσματος. Έτσι έχουμε τη δυνατότητα να αφιερώσουμε ένα επί πλέον ψηφίο για την αύξηση της ακρίβειας του αριθμού.

# ΠΡΟΣΘΕΣΗ ΑΡΙΘΜΩΝ ΚΙΝΗΤΗΣ ΥΠΟΔΙΑΣΤΟΛΗΣ

Το θέμα της εργασίας μας ήταν η πρόσθεση δύο αριθμών κινητής υποδιαστολής, μονής ακρίβειας (binary 32).

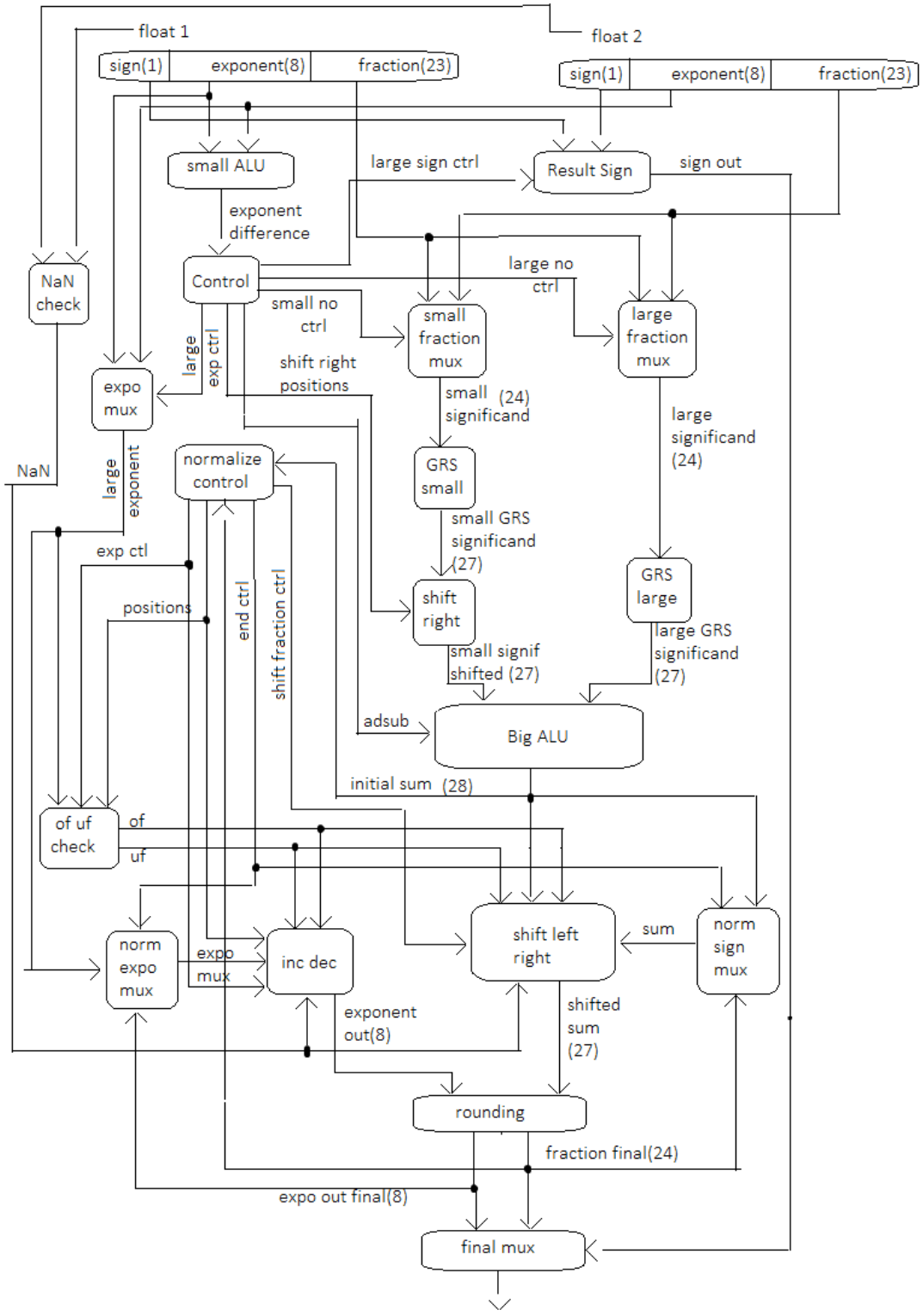
Ο αλγόριθμος της πρόσθεσης είναι αρκετά περίπλοκος. Το αφαιρετικό διάγραμμα ροής του είναι το εξής:




Οι εξαιρέσεις, όπως αναφέρονται στο διάγραμμα ροής, μπορεί να προκύψουν ως εξής:

| Όρος 1 | Όρος 2 | Άθροισμα |
|---|---|---|
| +άπειρο | αριθμός | +άπειρο |
| -άπειρο | αριθμός | -άπειρο |
| +άπειρο | -άπειρο | NaN |
| NaN | αριθμός | NaN |
| NaN | +άπειρο | NaN |
| NaN | -άπειρο | NaN |
| NaN | NaN | NaN |

Το αναλυτικό σχηματικό διάγραμμα του κυκλώματος που υλοποιεί το παραπάνω, είναι αυτό:




Στη συνέχεια θα επεξηγήσουμε τη λειτουργία του αλγορίθμου, μέσω της περιγραφής του κάθε δομικού στοιχείου που φαίνεται στο διάγραμμα:

Small ALU:

Για την πρόσθεση δύο αριθμών κινητής υποδιαστολής, είναι απαραίτητη η σύγκριση των εκθετών τους, και η εύρεση του μεγαλύτερου.

Για να είναι ευκολότερη η σύγκριση των εκθετών, στο IEEE 754 standard οι εκθέτες είναι biased. Δηλαδή για τη δημιουργία της αναπαράστασης κινητής υποδιαστολής, στην κανονική τιμή τους προστίθεται ένας σταθερός αριθμός, που εξασφαλίζει πως η αναπαράσταση του εκθέτη θα είναι πάντα θετική. Για την εύρεση του αριθμού, από την αναπαράσταση, χρειάζεται φυσικά ο σταθερός αριθμός να αφαιρεθεί από τον εκθέτη. Η τιμή του αριθμού αυτού είναι 127 για τους αριθμούς μονής ακρίβειας.

Τη λειτουργία της σύγκρισης των εκθετών εκτελεί η small ALU: αφαιρεί τους εκθέτες.

Control:

Το στοιχείο control λαμβάνει ως είσοδο τη διαφορά των εκθετών και αποφασίζει ποιος αριθμός είναι ο μεγαλύτερος.

Με βάση αυτή την πληροφορία παράγει στην έξοδό του σήματα ελέγχου για τα επόμενα υποκυκλώματα.

Result sign:

Το πρόσημο του αθροίσματος είναι το ίδιο με το πρόσημο του μεγαλύτερου όρου, το οποίο επιλέγεται εδώ από έναν πολυπλέκτη, με γνώμονα το αντίστοιχο σήμα ελέγχου.

Expo mux, Small και Large fraction mux:

Πολυπλέκτες που με βάση τα σήματα ελέγχου, αποφασίζουν ποια από τις εισόδους είναι η μεγαλύτερη και ποια η μικρότερη, και δίνουν αντίστοιχα στην έξοδό τους το μεγαλύτερο εκθέτη, καθώς και το μεγαλύτερο και το μικρότερο κλάσμα.

Οι πολυπλέκτες των κλασμάτων επιτελούν και μια επί πλέον διεργασία: προσθέτουν έναν επί πλέον άσσο στην αρχή του κάθε κλάσματος. Είναι ο άσσος που υπονοείται εφόσον η αναπαράσταση είναι κανονικοποιημένη, για την εξοικονόμηση ενός bit ακρίβειας.

Στην έξοδό τους δίνουν λοιπόν το «σημαντικό», που προέκυψε από το αντίστοιχο κλάσμα συν τον ακέραιο άσσο.

GRS small, GRS large:

Μετά την πρόσθεση των εισόδων μας, ίσως χρειαστεί να κάνουμε στρογγυλοποίηση του αποτελέσματος, στην περίπτωση που η ακρίβεια του αθροίσματος δεν μπορεί να αναπαρασταθεί με τα ψηφία που έχουμε στη διάθεσή μας. Για να γίνει σωστά η

στρογγυλοποίηση θα πρέπει να έχουμε προνοήσει από αυτό το στάδιο, προσθέτοντας τρία μηδενικά στο τέλος κάθε σημαντικού.

Shift right:

Στην περίπτωση που οι εκθέτες των προσθετέων είναι διαφορετικοί, θα πρέπει πριν την πρόσθεση το σημαντικό του μικρότερου να ολισθηθεί προς τα δεξιά τόσες θέσεις, όσες και η απόλυτη τιμή της διαφοράς των εκθετών. Μόνο έτσι θα προσθέσουμε τα ομοειδή δυαδικά ψηφία μεταξύ τους.

Η απόλυτη τιμή της διαφοράς παρέχεται στο στοιχείο ως έξοδος του control.

Big ALU:

Εκτελείται η πρόσθεση των σημαντικών, όπως αυτά έχουν διαμορφωθεί ως τώρα.

Normalize Control:

Για να ακολουθείται το πρότυπο, θα πρέπει το άθροισμα να έρθει και πάλι σε κανονικοποιημένη μορφή, δηλαδή να έχει ακριβώς έναν άσσο πριν την υποδιαστολή. Για να γίνει αυτό σωστά, θα πρέπει να γίνει ολίσθηση του αθροίσματος μέχρι να βρεθεί ο πρώτος άσσος στη δεύτερη θέση του (στην υλοποίησή μας έχουμε θεωρήσει δύο ψηφία πριν την υποδιαστολή, μετά το στάδιο της πρόσθεσης, ώστε να μη χαθεί το πιο σημαντικό ψηφίο.) και αντίστοιχα να αυξηθεί ή να μειωθεί ο εκθέτης.

Όλα τα παραπάνω ελέγχονται με τα κατάλληλα σήματα ελέγχου που παράγονται από το υποκύκλωμα normalize control.

Overflow, underflow, NaN check:

Κατά την κανονικοποίηση του αριθμού, μπορεί να χρειαστεί να αυξήσουμε τον εκθέτη τόσο, ώστε να μην μπορεί πλέον να αναπαρασταθεί με τα bit που έχουμε στη διάθεσή μας.

Μπορεί επίσης να λάβουμε ως είσοδο το άπειρο.

Και στις δύο περιπτώσεις θα συμβεί υπερχείλιση.

Υπάρχει ακόμα περίπτωση, ο αριθμός που θα προκύψει απ' την πρόσθεση να είναι μικρότερος απ' τη μικρότερη δυνατή κανονικοποιημένη αναπαράσταση. Σ' αυτή την περίπτωση θα συμβεί υποχείλιση.

Τέλος, στην περίπτωση που κάποια απ' τις εισόδους δεν είναι αριθμός (NaN) ή ζητείται να αφαιρέσουμε το άπειρο απ' το άπειρο, το άθροισμα που θα προκύψει, επίσης δεν θα είναι αριθμός.

Τα συγκεκριμένα υποκυκλώματα παράγουν σήματα ελέγχου για την αντιμετώπιση των παραπάνω εξαιρετικών καταστάσεων.

Inc-dec, shift left-right :

Όπως ανέφερα παραπάνω, εδώ εκτελείται η απαραίτητη ολίσθηση του σημαντικού, και η αυξομείωση του εκθέτη, για την κανονικοποίηση του αθροίσματος.

Για να γίνει σωστά η στρογγυλοποίηση, στο στάδιο που ακολουθεί, πρέπει να προσέξουμε πως αν κατά τη διάρκεια της ολίσθησης το sticky bit (το τελευταίο ψηφίο του αθροίσματος) γίνει 1 έστω και μια φορά, πρέπει να παραμείνει 1 και μετά το τέλος της διεργασίας.

Σ' αυτό το στάδιο διαχειριζόμαστε επίσης τις εξαιρετικές περιπτώσεις της υπερχείλισης, της υποχείλισης και του NaN.

Σε περίπτωση υπερχείλισης, το αποτέλεσμα της πρόσθεσης είναι το άπειρο. Αγνοούμε λοιπόν όλες τις πράξεις που έχουν γίνει μέχρι στιγμής και θέτουμε τον εκθέτη «11111111» και το κλάσμα «00000000000000000000000».

Σε περίπτωση υποχείλισης, για να καλυφθεί το κενό μεταξύ του μικρότερου δυνατού κανονικοποιημένου αριθμού και του 0, το IEEE 754 standard επιτρέπει τη χρήση μη κανονικοποιημένων αριθμών. Ταυτόχρονα ο εκθέτης μηδενίζεται, κάτι που δε συμβαίνει σε καμία άλλη περίπτωση.

Στην περίπτωση που το αποτέλεσμα είναι NaN, θέτουμε και πάλι χειροκίνητα τον εκθέτη «11111111» και το κλάσμα «10000000000000000000000». Αυτή είναι η πιο διαδεδομένη αναπαράστασή του.

Rounding:

Σε περίπτωση που το άθροισμα που έχει προκύψει απαιτεί περισσότερα ψηφία για την αναπαράστασή του απ' όσα έχουμε στη διάθεσή μας, απαιτείται στρογγυλοποίηση.

Υπάρχουν διάφορες μέθοδοι στρογγυλοποίησης. Εμείς χρησιμοποιήσαμε τη μέθοδο "Rounding to nearest, ties to even". Αυτή η μέθοδος κόβει τα τρία τελευταία ψηφία του αθροίσματος, τα ψηφία που είχαμε προσθέσει στο GRS, με ή χωρίς να προσθέσει 1 στο αποτέλεσμα. Αυτό εξαρτάται απ' τα τέσσερα τελευταία ψηφία του αθροίσματος, μετά την κανονικοποίηση.

Last mux:

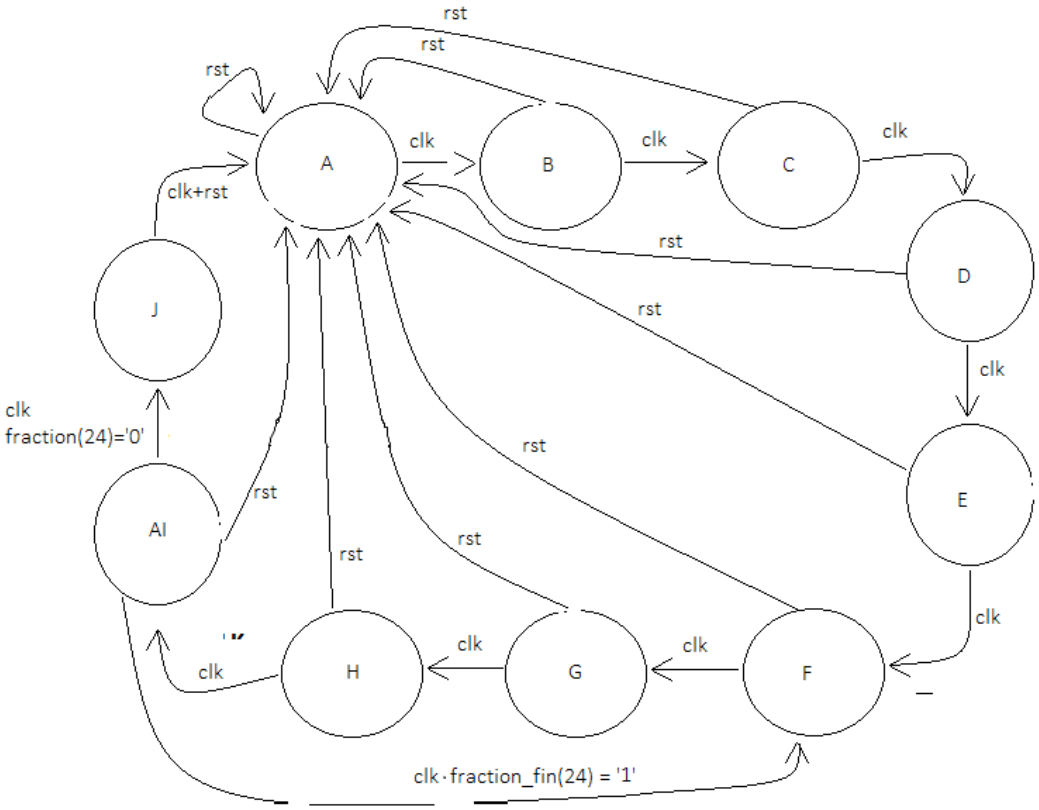
Υπάρχει περίπτωση, μετά την πρόσθεση του άσσου στο στάδιο της στρογγυλοποίησης, να καταλήξουμε σε μη κανονικοποιημένο αριθμό. Τότε πρέπει να

επιστρέψουμε ξανά στο στάδιο της κανονικοποίησης και να επαναλάβουμε τη διαδικασία.

Αντίθετα, αν η στρογγυλοποίηση καταλήξει σε κανονικοποιημένο αριθμό, είμαστε έτοιμοι να βγάλουμε έξοδο!!

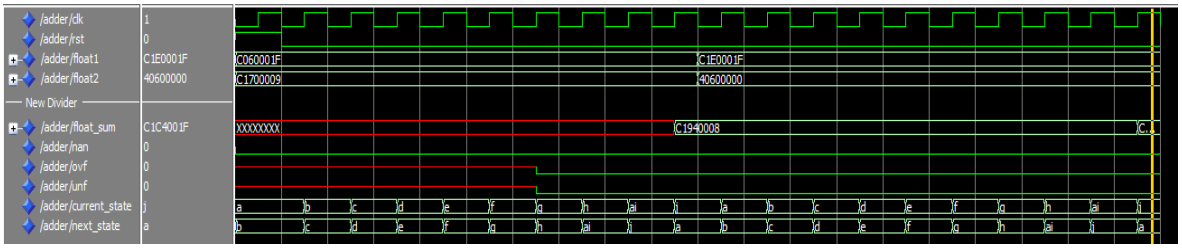
## FSM

Τα παραπάνω στάδια, συντονίζονται υπό τον έλεγχο μιας μηχανής πεπερασμένων καταστάσεων, το σχηματικό διάγραμμα της οποίας φαίνεται παρακάτω.

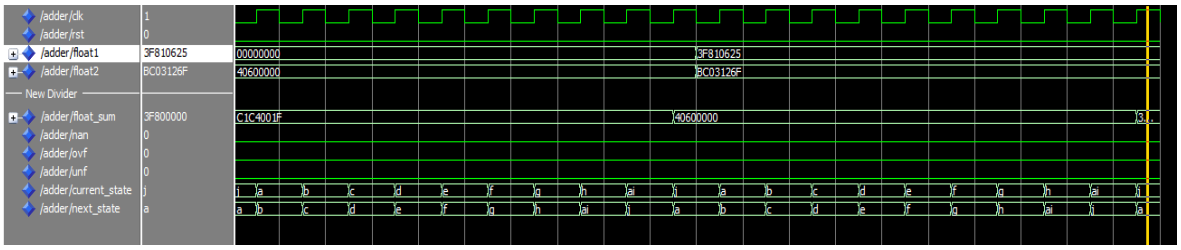


# ΠΡΟΣΟΜΟΙΩΣΕΙΣ

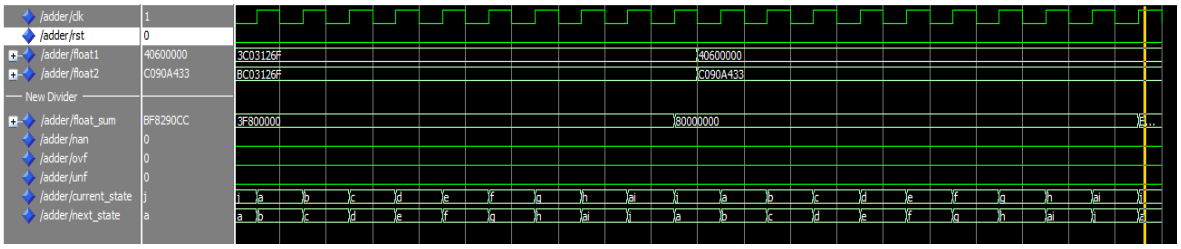
Τα αποτελέσματα του ModelSim τα οποία αποδεικνύουν την ορθή λειτουργία του κυκλώματος μας φαίνονται παρακάτω:




όπου ξεκινάμε με ασύγχρονο rst=1 με σκοπό να αρχικοποιηθεί η fsm μας στην κατάσταση A. Στη συνέχεια εισάγουμε τους αριθμούς: c060 001f hex ( -3.5000074 dec) στον float1 και c170 0009 hex (-15.000009 dec)στον float2. Το αποτέλεσμα εμφανίζεται στον float_sum μετά από 10 κύκλους ρολογιού (στην κατάσταση J) ως c194 0008 hex (-18.500015 dec). Παρατηρούμε πως τα σήματα nan, overflow και underflow είναι 0. Έπειτα, βάζουμε στην είσοδο c1e0 001f hex (-28.00006 dec) και 4060 0000 hex (+3.5 dec) και μετά από 10 κύκλους ρολογιού έχουμε float_sum = c1c4 001f hex (- 24.50006 dec).

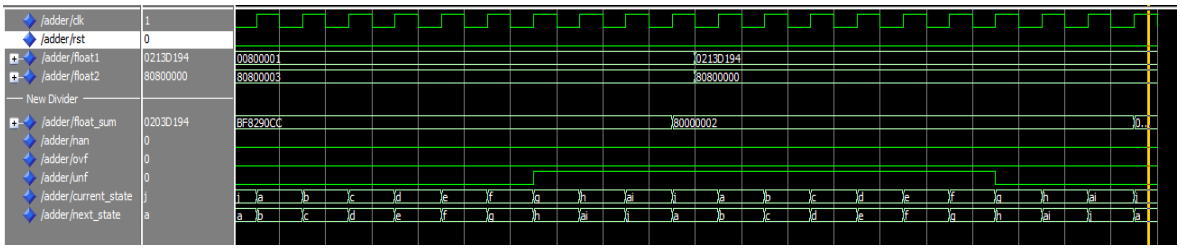



Παρόμοια, εισάγουμε 0000 0000 hex (+0.0 dec) και 4060 0000 hex (+3.5 dec) και παίρνουμε στο στάδιο J, 4060 0000 hex (+3.5 dec). Στην συνέχεια 3f81 0625 hex (+1.008 dec) + bc03 126f hex (-0.008 dec) = 3f80 0000 hex (+1.0 dec).

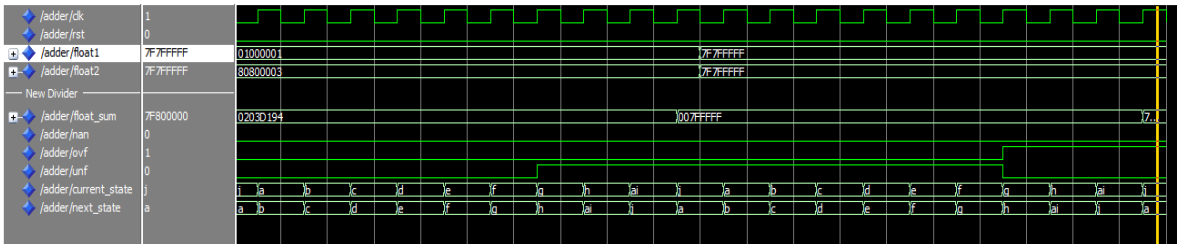



Εδώ προσθέσαμε το 3c03 126f hex (+0.008 dec) με το bc03 126f hex (-0.008) και πήραμε όπως αναμενόταν 8000 0000 hex (-0.0 dec). Έπειτα, 4060 0000 hex (+3.5 dec) + c090 a433 hex (-4.520044 dec) = bf82 90cc hex (-1.0200438 dec) όπου φαίνεται

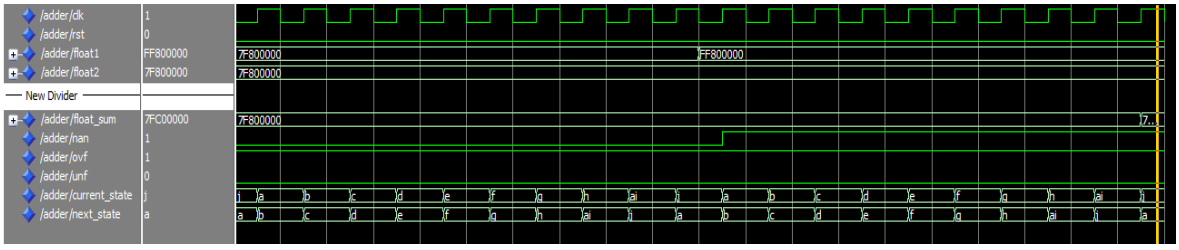
πως έγινε rounding καθώς στο δεκαδικό το αποτέλεσμα θα ήταν -1,020044 (bf82 90cd hex).



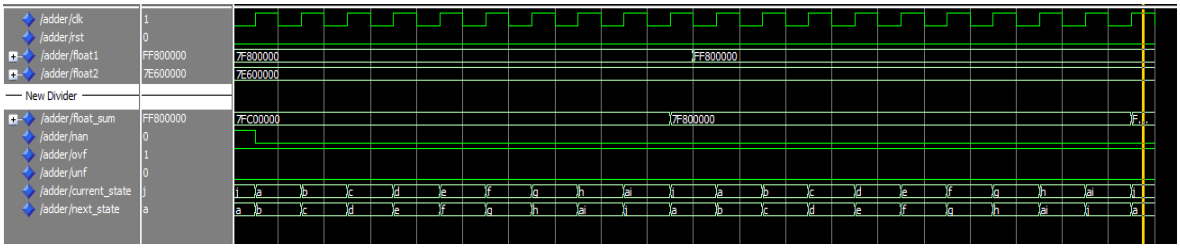

Προσθέτουμε το 0080 0001 hex (+1.1754945 E-38 dec) με το 8080 0003 hex (-1.1754948 E-38 dec) με αποτέλεσμα float_sum = 8000 0002 hex (-2.8 E-45 dec). Επειδή το αποτέλεσμα της πρόσθεσης για το exponent ήταν πολύ μικρό για να χωρέσει στην αναπαράσταση μας έχουμε Underflow και έτσι αφού πρέπει να δείξουμε denormalized number, βγάλαμε στην έξοδο 00h για το exponent και non zero fraction. Στη συνέχεια έχουμε float1=0213 d194 hex (+1.086 E-37 dec) και float2=8080 0000 hex (-1.17549435 E-38 dec) και έτσι μετά από 10 κύκλους έχουμε 0203 d194 hex (+9.684506 E-38 dec).



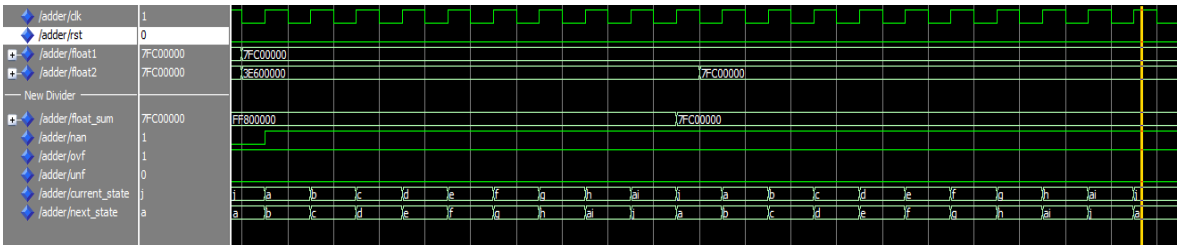

Σε αυτό το σημείο προσθέτουμε το 0100 0001 hex (+2.350989 E-38 dec) με το 8080 0003 hex (-1.1754948 E-38 dec) και παίρνουμε το denormalized 007f ffff hex (1.1754942 E-38 dec) όπως και underflow='1'. Έπειτα, προσθέτουμε τα 7f7f ffff hex (+3.4028235 E38 dec) με αποτέλεσμα να παρουσιάζεται Overflow καθώς ο εκθέτης είναι πολύ μεγάλος και έτσι το δείχνουμε με infinity 7f80 0000 hex.



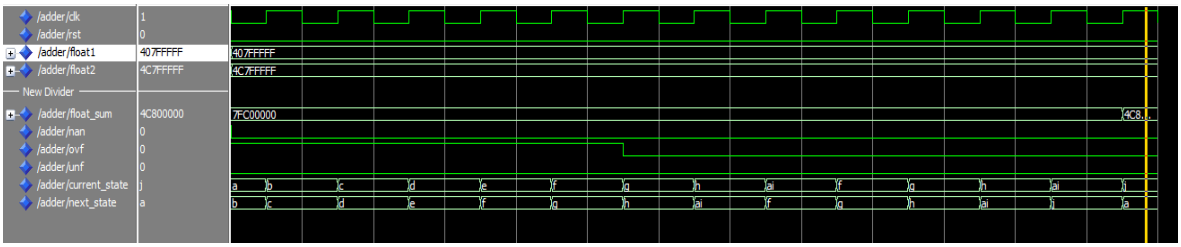

Προσθέτουμε το +άπειρο (7f80 0000 hex) με το +άπειρο και το float_sum προκύπτει +άπειρο. Όμως στη συνέχεια προσθέτουμε το +άπειρο με το -άπειρο (ff80 0000 hex) και προκύπτει NaN (7fc0 0000 hex).




Όπως παρατηρούμε προσθέτοντας οποιονδήποτε Normalized αριθμό με το +άπειρο ή το -άπειρο, προκύπτει +άπειρο και - άπειρο αντίστοιχα.




Επίσης, η πρόσθεση ενός NaN αριθμού με τον 3e60 0000 hex (+0.21875 dec) δίνει NaN αποτέλεσμα, όπως και η πρόσθεση δύο NaN.

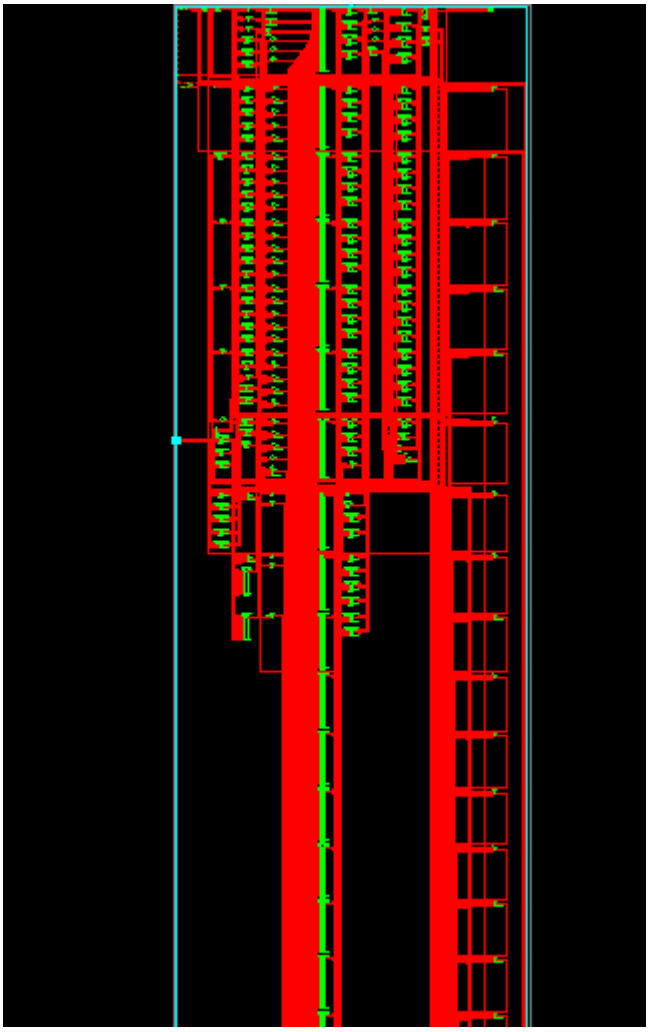



Τέλος, επαληθεύσαμε πως σε περίπτωση που χρειαστεί, ξαναγυρνάει στο επίπεδο του normalize. Προσθέσαμε το 407f ffff hex (+3.9999998 dec) με το 4c7f ffff hex (+6.710886 E7 dec). Επειδή στο στάδιο του rounding (κατάσταση AI) προσθέσαμε το 1 σε μια σειρά από άσσους, προέκυψε denormalized αριθμός με αποτέλεσμα να πρέπει να γυρίσουμε στην κατάσταση F αντί να βγάλουμε έξοδο στην κατάσταση J. Προφανώς το αποτέλεσμα καθυστέρησε κατά 4 κύκλους ρολογιού παραπάνω απ΄ ότι αν δεν χρειαζόταν να γίνει Normalized πάλι.

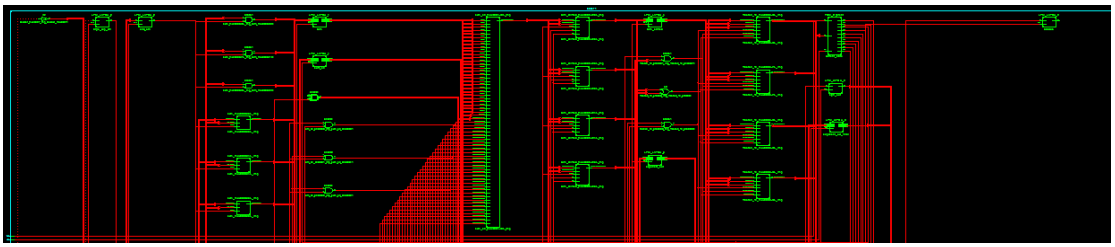
## RTL SCHEMATIC

Το κύκλωμα μας έκανε χρήση: μιας FSM 10 καταστάσεων, 9 adders/subtractors (1 9-bits για την αφαίρεση των exponents, 1 8-bits για την περίπτωση που έχουμε διαφορετικά πρόσημα και πρέπει να κάνουμε αφαίρεση και 1 8-bits για την περίπτωση της πρόσθεσης, 1 25-bits για την εμφάνιση του fraction_final, 2 32-bits για να ελέγξουμε την πρόσθεση ή την αφαίρεση του large_exponent με τον αριθμό των θέσεων, 1 5-bits για να βρούμε την θέση του πρώτου άσσου και να κάνουμε τα ανάλογα shifts, 1 10-bits για να βρούμε κατά πόσες θέσεις θα αλλάξει το exponent και 1 28-bit για την big alu), 5 comparators (1 10-bits για το σήμα decider, 1 32-bits για το σήμα overflow, 1 9-bits για το σήμα sign_out, 1 27-bits για το σήμα small_signigicand_shifted και 1 32-bits για το σήμα underflow), 1 xor2 για να επιλέξει αν θα κάνει πρόσθεση ή αφαίρεση, 1 28-bits logical shifter για το σήμα sum_shifted και 26 latches.

Η αρχιτεκτονική σε RTL επίπεδο:



Η οποία είναι αρκετά περίπλοκη για να παρουσιαστεί με ακρίβεια. Ένα κομμάτι της:



## ΣΥΝΘΕΣΗ

Για την σύνθεση σχεδιασμού του υλικού σε πλατφόρμες αναδιατασσόμενης λογικής (FPGA), χρησιμοποιήσαμε την σουίτα ISE της Xilinx. Επιλέξαμε την οικογένεια Spartan 3E (η οποία στοχεύει σε εφαρμογές χαμηλής κατανάλωσης και το κόστος της είναι προσιτό), την συσκευή XC3S500E, το package FT256 και speed rate -4. Η επιλογή keep hierarchy έμεινε στην default τιμή της, NO, με στόχο την βελτιστοποίηση και κατ' επέκταση την αύξηση της συχνότητας και την μείωση της επιφάνειας. Τέλος, η σύνθεση έγινε με κύριο στόχο την αύξηση της ταχύτητας.

Κατά την σύνθεση του κυκλώματος, η αναφορά για την επιφάνεια του FPGA που καταλαμβάνει ο σχεδιασμός μας έδωσε τα παρακάτω στοιχεία:

| Logic Utilization | Used | Available | Utilization |
|---|---|---|---|
| Number of Slices | 590 | 4656 | 12% |
| Number of Slice Flip Flops | 280 | 9312 | 3% |
| Number of 4 input LUTs | 1036 | 9312 | 11% |
| Number of IOs | 98 | - | - |
| Number of bonded IOBs | 98 | 190 | 51% |
| IOB Flip Flops | 32 | - | - |
| Number of GCLKs | 8 | 24 | 33% |

Όσον αφορά στο timing report:

Minimum period 2.638ns (=> Maximum Frequency: 379.075MHz) (60.8% logic, 39.2% route) το οποίο προκύπτει από: FDC -> LUT3 -> FDC.

Minimum input arrival time before clock: 8.949ns (64.3% logic, 35.7% route)το οποίο προκύπτει από: IBUF -> LUT4-> LUT4-> LUT4-> LUT4-> LUT3-> LUT4-> LD.

Maximum output required time after clock: 4.368ns (90.4% logic, 9.6% route) το οποίο προκύπτει από: LD -> OBUF.

## ΣΥΜΠΕΡΑΣΜΑΤΑ

Σχεδιάζοντας το κύκλωμα παρατηρήσαμε πως το υλικό του αθροιστή κινητής υποδιαστολής είναι αρκετά πιο πολύπλοκο από του ακεραίου αθροιστή και πως αν η άθροιση εκτελεστεί σε έναν κύκλο ρολογιού, θα πρέπει η συχνότητα να μειωθεί πολύ.

Για να αυξήσουμε την επίδοση του κυκλώματος λοιπόν, υλοποιήσαμε pipeline 10 σταδίων με την βοήθεια μιας FSM 10 καταστάσεων. Πετύχαμε μ’ αυτόν τον τρόπο σημαντική άυξηση της μέγιστης συχνότητας.

Επιπλεόν, μέσω αυτής της εργασίας βελτιώσαμε τις σχεδιαστικές μας ικανότητες υλοποιώντας ένα πολύπλοκο ολοκληρωμένο κύκλωμα, το οποίο μάλιστα χρησιμοποιείται ευρέως ως component σε μεγαλύτερες σχεδιάσεις οι οποίες κάνουν χρήση αριθμών κινητής υποδιαστολής.

Τέλος, μέσω της σύνθεσης είχαμε τη δυνατότητα να αναλύσουμε το υλικό, το χρονισμό και την επιφάνεια του σχεδιασμού μας και να προβούμε σε βελτιώσεις, όπου αυτές ήταν δυνατές.