ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Αποδοτική σχεδίαση Multiplier-Adder/Accumulator για αριθμούς σε μορφή sign-magnitude

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΓΕΩΡΓΙΟΥ ΜΙΧΑΗΛ

Επιβλέπων : Κιαμάλ Πεκμεστζή.

Καθηγητής Ε.Μ.Π.

ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ
ΣΧΟΛΗ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ
ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ
ΤΟΜΕΑΣ ΤΕΧΝΟΛΟΓΙΑΣ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΥΠΟΛΟΓΙΣΤΩΝ

Αποδοτική σχεδίαση Multiplier-Adder/Accumulator για αριθμούς σε μορφή sign-magnitude

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

του

ΓΕΩΡΓΙΟΥ ΜΙΧΑΗΛ

Επιβλέπων : Κιαμάλ Πεκμεστζή
Καθηγητής Ε.Μ.Π.

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή την .

.....	.....	.....
Κ.Πεκμεστζή.	Δ. Σούντρης.	Γ. Οικονομάκος
Καθηγητής Ε.Μ.Π	Καθηγητής Ε.Μ.Π	Επ.Καθηγητής Ε.Μ.Π

Αθήνα, Ιούνιος 2016

.....

Γεώργιος Μιχαήλ.

Διπλωματούχος Ηλεκτρολόγος Μηχανικός και Μηχανικός Υπολογιστών Ε.Μ.Π.

Περίληψη

Σκοπός της διπλωματικής αυτής εργασίας ήταν σχεδίαση διαφόρων τοπολογιών multiplier-accumulator (MAC) και multiplier-adder (MAD) για υπολογισμό ενός και δύο γινομένων, με εισόδους και εξόδους στη μορφή προσήμου-μέτρου. Για κάθε τοπολογία έγινε διερεύνηση ως προς τις επιδόσεις της και ειδικότερα ως προς τη μέγιστη δυνατή συχνότητα ρολογιού στην οποία μπορεί να λειτουργήσει.

Στη μελέτη αύξησης της συχνότητας ρολογιού, έγινε προσαρμογή του αθροιστή πρόβλεψης κρατουμένου ώστε να επιτευχθεί επιτάχυνση του υπολογισμού της απόλυτης τιμής του αποτελέσματος και να μειωθεί το ελάχιστο μονοπάτι. Επιπλέον, κάθε υλοποίηση μελετήθηκε σε συνδυαστική λειτουργία όπου το αποτέλεσμα υπολογίζεται σε ένα κύκλο ρολογιού και σε λειτουργία συνεχούς διοχέτευσης όπου το αποτέλεσμα υπολογίζεται σε δύο κύκλους ρολογιού.

Για το κύκλωμα πολλαπλασιασμού χρησιμοποιήθηκε παράλληλος πολλαπλασιαστής με μετασχηματισμό του πολλαπλασιαστή σύμφωνα με τη μέθοδο του MacSorley-Booth (Modified Booth) ώστε να επιτευχθεί μείωση των μερικών γινομένων προς άθροιση. Για την συμπίεση των μερικών γινομένων ώστε να είναι κατάλληλα για είσοδο στο τελικό αθροιστή χρησιμοποιήθηκε ένας δεντρικός συμπιεστής Wallace.

Στα πλαίσια μελέτης της επίδοσης της προσαρμογής που έγινε στον αθροιστή πρόβλεψης κρατουμένου έγινε σύγκρισή του τροποποιημένου αθροιστή με μία συνηθέστερη μέθοδο όπου υπολογίζεται το αποτέλεσμα αρχικά σε μορφή συμπληρώματος ως προς δύο και ύστερα γίνεται μετατροπή του σε πρόσημο-μέτρο. Επιπλέον έγινε σύγκριση των προτεινόμενων υλοποιήσεων και με αντίστοιχα κυκλώματα που επεξεργάζονται αριθμούς σε μορφή συμπληρώματος ως προς δύο. Για τις τρεις αυτές υλοποιήσεις έγινε χρονική ανάλυση, δηλαδή μελέτη συμπεριφοράς των κύριων χαρακτηριστικών λειτουργίας των κυκλωμάτων για ένα εύρος συχνοτήτων ρολογιού.

Συγκεκριμένα, έγινε περιγραφή όλων των κυκλωμάτων που αναλύονται σε αυτήν την εργασία σε γλώσσα περιγραφής υλικού Verilog, για μήκη λέξης 8,16,24,32 και 64 bit. Η σύνθεσή τους έγινε με βάση μία standard cell CMOS βιβλιοθήκη 65nm της Artisan, Από τα αποτελέσματα μελετήθηκε η συμπεριφορά κάθε κυκλώματος ως προς την *ελάχιστη περίοδο ρολογιού (ns)* στην οποία μπορεί να γίνει η σύνθεσή του, την μετρική *επιφάνεια επί περίοδο ρολογιού ($um^2 * ns$),* και την *κατανάλωση ενέργειας (mW∗ns).*

Λέξεις Κλειδιά:

Multiplier-Accumulator, Modified Booth, Verilog, δεντρικός συμπιεστής Wallace, Αθροιστής πρόβλεψης κρατουμένου, αναπαράσταση σε πρόσημο-μέτρο, απόλυτη τιμή, CMOS 65nm, μείωση ελάχιστου μονοπατιού.

Abstract

The scope of this thesis was the design of various multiplier-accumulator (MAC) and multiplier-adder (MAD) schemes, for one and two products, with both inputs and output represented in sign-magnitude form. Each scheme was tested for its performance and especially as far as their critical path is concerned.

In the scope of increasing the clock frequency, a modification at a carry-lookahead (CLA) adder is proposed, in order to accelerate computing the absolute value of the result and shorten the critical path. Each scheme was designed to compute the result in one clock cycle and in pipeline operation for computing in two clock cycles.

The multiplication scheme was designed in parallel, and in all designs the multiplicand is transformed according to the MacSoreley-Booth method (Modified Booth) in order to reduce the number of partial products which will be added. For the compression of all the partial products into to two vectors in order to be added at a carry-lookahead adder a Wallace tree was used for fast results.

In order to test the performance of the modification, a direct comparison is represented between the modified CLA Adder and a common method of conversion between two complement and sign magnitude form. Additionally the performance of respective designs which receive and extract data in two complement form is also represented. For these three designs a time analysis was performed to show how the main parameters change according to clock time period.

More specifically, all designs that are represented in this thesis were designed using Verilog HDL for input lengths of 8, 16, 24, 32, and 64 bit. The designs were synthesized using a 65nm standard cell CMOS library by Artisan. All designs are analyzed according to their *critical path (ns), area∗clock period (um^2∗ns),* and *energy consumption (mW∗ns).*

Keywords:

Multiplier-accumulator, Modified Booth, Verilog, Wallace tree, Carry-lookahead Adder, sign-magnitude form representation, Absolute value, CMOS 65nm, Critical Path length reduction.

Πίνακας Περιεχομένων

1 ΕΙΣΑΓΩΓΗ

1.1 Πεδία Εφαρμογών Multiplier-Accumulator και sign-magnitude αριθμητικής

Στην σημερινή εποχή, η τεχνολογία και κυρίως η σχεδίαση ψηφιακών κυκλωμάτων εξελίσσεται με γρήγορους ρυθμούς, οι απαιτήσεις των καταναλωτών και των εφαρμογών επιτάσσουν αυξημένη ταχύτητα, μικρή επιφάνεια και χαμηλή κατανάλωση ενέργειας. Στα αριθμητικά κυκλώματα τα χαρακτηριστικά αυτά είναι άμεσα εξαρτώμενα από την επιλογή της μορφής αναπαράστασης των δεδομένων εισόδου και εξόδου, αν και πλέον η πιο κοινή και πιο αποδοτική έχει καθιερωθεί η αναπαράσταση σε μορφή συμπληρώματος ως προς δύο, υπάρχουν ακόμα εφαρμογές στις οποίες χρησιμοποιείται η μορφή αναπαράστασης προσήμου-μέτρου. Για παράδειγμα στην αριθμητική floating point όπου οι αριθμοί βρίσκονται σε μορφή μέτρου (mantissa) και εκθέτη (exponent) η mantissa τους είναι σε μορφή προσήμου-μέτρου, επίσης πολλές DSP εφαρμογές για την υλοποίηση φίλτρων για την επεξεργασία ασθενών σημάτων χρησιμοποιούν αυτήν την αναπαράσταση για χαμηλή κατανάλωση ενέργειας λόγω μικρών εναλλαγών των bit των αριθμών σε συχνές αλλαγές του σήματος γύρω από το μηδέν.

Συνεπώς υπάρχει επιτακτική ανάγκη για αποδοτικό σχεδιασμό κυκλωμάτων, τα οποία χρησιμοποιούν δεδομένα στη μορφή αναπαράστασης προσήμου-μέτρου, ώστε τα χαρακτηριστικά τους να είναι παρόμοια με εκείνα κυκλωμάτων που χρησιμοποιούν μορφή συμπληρώματος ως προς δύο. Ο MAC είναι ένα συνήθες ψηφιακό κύκλωμα το οποίο χρησιμοποιείται κυρίως αλλά όχι μόνο, σε εφαρμογές ψηφιακής επεξεργασίας σημάτων ήχου και εικόνας, σε αισθητήρες, σε συστήματα ραντάρ.

Ο τομέας της αποδοτικής σχεδίασης ψηφιακών κυκλωμάτων είναι ο τομέας στον οποίο γίνεται η μεγαλύτερη έρευνα για την βελτίωση εφαρμογών. Ο σχεδιασμός πριν αρκετά χρόνια ήταν πολύ περίπλοκος καθώς δεν υπήρχαν τα κατάλληλα εργαλεία και οι συνηθέστερη περίπτωση ήταν η σχεδίαση ψηφιακών κυκλωμάτων σε μεγάλους χάρτες με αναλυτικό σχεδιασμό όλων των λογικών πυλών του κυκλώματος. Πλέον με την ανάπτυξη των γλωσσών περιγραφής υλικού, όπως η VHDL και η Verilog είναι εύκολη η περιγραφή των κυκλωμάτων είτε σε υψηλό επίπεδο με χρήση διαφόρων αλγορίθμων είτε σε χαμηλό επίπεδο, αυτό των λογικών πυλών, και η σύνθεσή τους για τον έλεγχο της επίδοσής τους μπορεί να γίνει αξιόπιστα με πολλά λογισμικά.

1.2 Αντικείμενο διπλωματικής

Η παρούσα διπλωματική εργασία έχει ως αντικείμενο τη μελέτη σχεδιασμών multiplier-accumulator (MAC), και multiplier-adder (MAD) για αριθμητική πρόσημου-μέτρου. Παρουσιάζονται και συγκρίνονται δύο διαφορετικές τρόποι σχεδίασης τους, ο πρώτος περιλαμβάνει τη μετατροπή των αριθμών εισόδων προς επεξεργασία από τη μορφή πρόσημου-μέτρου σε μορφή συμπληρώματος ως προς δύο, στο τέλος του υπολογισμού χρησιμοποιείται ένας μετατροπέας για την αναπαράσταση του αποτελέσματος σε μορφή πρόσημου-μέτρου, αυτή η μέθοδος είναι γνωστή ως έμμεσος σχεδιασμός αφού περιλαμβάνει το μετασχηματισμό των δεδομένων από τη μία μορφή στην άλλη. Ο δεύτερος τρόπος σχεδίασης γίνεται με έναν προσαρμοσμένο αθροιστή πρόβλεψης κρατουμένου προκειμένου να υπολογίζεται το αποτέλεσμα απευθείας σε μορφή πρόσημου-μέτρου ώστε να αποφευχθεί η χρήση μετατροπέα, γνωστή και ως άμεση σχεδίαση. Επιπλέον γίνεται απευθείας σύγκριση με MAC και MAD οι οποίοι δέχονται και παράγουν αποτελέσματα σε μορφή συμπληρώματος ως προς δύο.

Οι τρεις αυτοί τρόποι σχεδίασης εφαρμόστηκαν στα εξής κυκλώματα.

- Για MAD που υπολογίζει το γινόμενο δύο αριθμών και το προσθέτει σε έναν τρίτο αριθμό ως είσοδο, δηλαδή το ψηφιακό κύκλωμα για υπολογισμό της αριθμητικής πράξης $A * X + D$.
- Για MAC που εκτελεί συσσώρευση ενός γινομένου δύο αριθμών, δηλαδή εκτελείται η αριθμητική πράξη $\sum A_i * X_i$.
- Για MAC που εκτελεί ξανά συσσώρευση ενός γινομένου δύο αριθμών, αλλά λειτουργεί σε λειτουργία συνεχούς διοχέτευσης δύο σταδίων (two-stage pipeline), δηλαδή εκτελείται η αριθμητική πράξη $\sum A_i * X_i$ σε δύο κύκλους ρολογιού.
- Για MAD που υπολογίζει δύο γινόμενα τεσσάρων αριθμών και τα προσθέτει σε έναν πέμπτο αριθμό δηλαδή το ψηφιακό κύκλωμα για υπολογισμό της αριθμητικής πράξης $A * X + B * Y + D$, γνωστό και ως double MAD.
- Για MAC που εκτελεί συσσώρευση δύο γινομένων τεσσάρων αριθμών, δηλαδή για την εκτέλεση της αριθμητικής πράξης $\sum(A_i * X_i + B_i * Y_i)$, γνωστό και ως double MAC.
- Για double MAC που εκτελεί ξανά συσσώρευση δύο γινομένων τεσσάρων αριθμών αλλά λειτουργεί σε λειτουργία συνεχούς διοχέτευσης δύο σταδίων, δηλαδή για την εκτέλεση της αριθμητικής πράξης $\sum(A_i * X_i + B_i * Y_i)$ σε δύο κύκλους ρολογιού.

Δόθηκε ιδιαίτερη βάση στο στάδιο άθροισης των μερικών γινομένων με αποδοτικό τρόπο για την εξαγωγή του αποτελέσματος σε sign-magnitude μορφή και λιγότερη στο στάδιο δημιουργίας και συμπίεσης μερικών γινομένων για την οποία χρησιμοποιήθηκε Modified Booth κωδικοποίηση και ένας δενδρικός συμπιεστής Wallace σε όλους τους σχεδιασμούς.

Όλες οι υλοποιήσεις που παρουσιάζονται σε αυτή την εργασία σχεδιάστηκαν με την γλωσσα περιγραφής υλικού verilog σε επίπεδο λογικών πυλών εκτός από τον δενδρικό συμπιεστή Wallace για το κύκλωμα του οποίου χρησιμοποιήθηκε έτοιμη κυκλωματική υλοποίηση της Synopsys. Η λειτουργία όλων των κυκλωμάτων προσομοιώθηκε και επαληθεύτηκε με το περιβάλλον modelsim για μήκη λέξης 8, 16, 24, 32 και 64 bit ενώ η σύνθεσή τους έγινε με τον Synopsys Design Compiler

με χρήση μίας standard cell βιβλιοθήκης της Artisan 65nm. Τα χαρακτηριστικά ως προς το ελάχιστο μονοπάτι, την επιφάνεια και την κατανάλωση ενέργειας εξάχθηκαν από το Synopsys Design Compiler και το Synopsys Prime Power.

1.3 Συνεισφορά Διπλωματικής

Η συνεισφορά της διπλωματικής συνοψίζεται ως εξής:

1. Υλοποιήθηκαν δύο διαφορετικοί τρόποι σχεδίασης MAC και MAD, ο ένας με υπολογισμό του αποτελέσματος σε μορφή συμπληρώματος ως προς δύο και ύστερα μετατροπή σε μορφή πρόσημου-μέτρου και ο δεύτερος με χρήση προσαρμοσμένου αθροιστή πρόβλεψης κρατουμένου για απευθείας υπολογισμό απόλυτης τιμής.
2. Έγινε σύγκριση και των δύο μεθόδων με αντίστοιχα κυκλώματα που λειτουργούν σε συμπλήρωμα ως προς δύο.
3. Παρουσιάζονται τα μειονεκτήματα και τα πλεονεκτήματα της κάθε μεθόδου.
4. Παρουσιάζονται οι διαφορές μεταξύ λειτουργίας συνεχούς διοχέτευσης και κανονικής λειτουργίας.
5. Αξιολογήθηκε η απόδοση των κυκλωμάτων και συγκεκριμένα μετρήθηκε: η ελάχιστη δυνατή περίοδος ρολογιού στην οποία ο Compiler μπορεί να κάνει σύνθεση με ασφάλεια, η επιφάνεια σχεδίασης επί περίοδο ρολογιού($um^2 * ns$) και κατανάλωση ενέργειας ($mW*ns$)
6. Παρουσιάζεται η συμπεριφορά τους ως προς αυτά τα χαρακτηριστικά ανάλογα με το μήκος λέξης και ανάλογα με την περίοδο στην οποία λειτουργεί το ρολόι.

1.4 Οργάνωση κειμένου

Στο κεφάλαιο 2 παρουσιάζεται όλη η απαραίτητη θεωρία προκειμένου η ανάγνωση της παρούσας διπλωματικής εργασίας να είναι δυνατή από κάθε αναγνώστη.

Στο κεφάλαιο 3 παρουσιάζονται όλα τα κυκλώματα MAC και MAD που υλοποιήθηκαν, αναλύεται ο τρόπος σχεδιασμού τους με βάση τις μικρότερες μονάδες που παρουσιάζονται στο δεύτερο κεφάλαιο και γίνεται υπολογισμός με βάση τη θεωρία της επιφάνειας και της καθυστέρησης που πρόκειται να εμφανιστούν.

Στο κεφάλαιο 4 παρουσιάζονται τα πειραματικά αποτελέσματα των συνθέσεων όπως το *critical path, Design Area∗Clock Period* και *Energy Consumption*. Γίνεται σύγκριση των αποτελεσμάτων, και κλείσιμο της διπλωματικής με τα συμπεράσματα της διπλωματικής και πιθανές επεκτάσεις της.

Στο κεφάλαιο 5 γίνεται ανάλυση των αποτελεσμάτων, και κλείσιμο της διπλωματικής με τα συμπεράσματα και πιθανές επεκτάσεις της.

Στο κεφάλαιο 6 παρουσιάζεται η βιβλιογραφία και οι πηγές πάνω στις οποίες στηρίχτηκαν όσα παρουσιάζονται.

2 ΘΕΩΡΗΤΙΚΟ ΥΠΟΒΑΘΡΟ

2.1 Εισαγωγή

Σε αυτό το κεφάλαιο θα καλυφθεί η απαραίτητη θεωρία προκειμένου η παρούσα διπλωματική εργασία να γίνει κατανοητή για κάθε αναγνώστη.

Αρχικά παρουσιάζεται το δυαδικό σύστημα αναπαράστασης αριθμών και μελετώνται οι τρόποι με τους οποίους πραγματοποιούνται οι συνήθεις αριθμητικές πράξεις πάνω σε αυτό, επιπλέον παρουσιάζονται όλες οι μέθοδοι αναπαράστασης προσημασμένων αριθμών, καθώς και τα μειονεκτήματα και τα πλεονεκτήματα της κάθε μεθόδου.

Στη συνέχεια αναλύονται οι βασικές αριθμητικές μονάδες που αποτελούν τη βάση πάνω στην οποία χτίζονται οι πιο σύνθετες, όπως ένας multiplier adder/accumulator, που παρουσιάζονται στο τρίτο κεφάλαιο. Αναλύονται οι βασικές τοπολογίες αθροιστών και πολλαπλασιαστών και επιπλέον μελετάται ο αλγόριθμος Modified Booth που βοηθάει στην κωδικοποίηση με τρόπο ώστε να βελτιώνεται η ταχύτητα και η απόδοση του πολλαπλασιασμού.

2.2 Τα αριθμητικά συστήματα

Για την αναπαράσταση κάθε αριθμητικού συστήματος αναγκαίο είναι ο καθορισμός της βάσης του, η οποία συμβολίζεται με r. Κάθε αριθμός σε ένα σύστημα με βάση r είναι ένα string ψηφίων όπως:

$$(d_{n-1}d_{n-2}\dots d_0)_r$$

Τα d_i αποτελούν τα ψηφία του αριθμού, με $0 \le \iota \le n-1$ και $d_i \in \{0,1,\dots,r-1\}$. Η θέση κάθε ψηφίου μέσα στο string καθορίζει τη σημαντικότητα του ψηφίου (ή αλλιώς το βάρος του) μέσα στον αριθμό, όσο πιο αριστερά βρίσκεται το ψηφίο τόσο πιο σημαντικό είναι με το ψηφίο που βρίσκεται στην ακραία αριστερή θέση να ονομάζεται το πιο σημαντικό (most significant digit MSD) και το ψηφίο που βρίσκεται τέρμα δεξιά το λιγότερο σημαντικό (less significant digit LSD).

Το μέτρο N του παραπάνω αριθμού δίνεται από την σχέση:

$$|N| = d_{n-1} * r^{n-1} + \ d_{n-2} * r^{n-2} + \dots + d_0 r^0 = \ \sum_{i=0}^{n-1} d_i r^i \qquad (2.1)$$

Η παραπάνω αναπαράσταση αριθμών ισχύει και για αριθμούς οι οποίοι περιέχουν κλασματικό μέρος. Έστω ότι ο αριθμός αποτελείται από n ψηφία στο ακέραιο και k ψηφία στο κλασματικό μέρος τα οποία χωρίζονται με την γνωστή υποδιαστολή (“,”) τότε η αναπαράσταση του αριθμού είναι η εξής:

$$(d_{n-1}d_{n-2}\dots\ d_0\ .\ d_{-1}\dots\ d_{-k})_r$$

Τότε το μέτρο του αριθμού δίνεται και πάλι από τον τύπο (2.1), όπως για παράδειγμα, για τον υπολογισμό του μέτρου του παρακάτω κλασματικού αριθμού

$$N = (13.4)_8 = 1 * \ 8^1 + 3 * 8^0 + 4 * 8^{-1} = (11,5)_{10}$$

Τα παραπάνω ισχύουν για όλα τα συστήματα με διαφορετικές βάσεις, τα πιο ευρέως χρησιμοποιούμενα αριθμητικά συστήματα είναι:

- Το δυαδικό σύστημα που χρησιμοποιεί σαν βάση το r=2 και τα ψηφία του είναι $d_i \in \{0,1\}$,
- Το δεκαδικό σύστημα στο οποίο έχει προσαρμοστεί ο άνθρωπος που έχει βάση το $r = 10$ και ψηφία $d_i \in \{0,1,2,4,5,6,7,8,9\}$.
- Το οκταδικό σύστημα που έχει βάση $r = 8$ και ψηφία τα $d_i \in \{0,1,2,4,5,6,7\}$,
- Το δεκαεξαδικό σύστημα που έχει βάση $r = 16$ και ψηφία:
  $d_i \in \{0,1,2,4,5,6,7,8,9,A,B,C,D,E,F\}$,.

2.3 Το δυαδικό σύστημα

Όπως αναφέρθηκε και προηγουμένως η αναπαράσταση αριθμών στο δυαδικό σύστημα γίνεται με τον ίδιο ακριβώς τρόπο όπως στο δεκαδικό όμως χρησιμοποιείται σαν βάση το 2 και τα ψηφία 0,1. Η εξοικείωση με το δυαδικό σύστημα είναι αναγκαία για την κατανόηση της λειτουργίας όλων των ψηφιακών κυκλωμάτων. Στα ψηφιακά κυκλώματα χρησιμοποιούνται δύο επίπεδα τάσης, το υψηλό επίπεδο "HIGH" και το χαμηλό επίπεδο "LOW", όπως φαίνεται υπάρχει πλήρης αντιστοιχία με το δυαδικό σύστημα το οποίο χρησιμοποιεί μόλις δύο ψηφία το 0 και το 1. Το ψηφίο 1 αναπαριστά επίπεδο τάσης "HIGH" ενώ το ψηφίο 0 αναπαριστά επίπεδο τάσης "LOW". Επίσης στο δυαδικό σύστημα τα ψηφία αναφέρονται ως bits με το ακραίο δεξιά bit να είναι το περισσότερο σημαντικό bit (MSB) και το ακραίο αριστερό το λιγότερο σημαντικό bit (LSB).

Προφανώς για την αναπαράσταση των αριθμών στο δυαδικό σύστημα χρειάζονται περισσότερα ψηφία από ότι στα άλλα συστήματα, για την αναπαράσταση ενός αριθμού με μέτρο N στο δυαδικό σύστημα χρειάζονται $\lfloor log_2 N \rfloor + 1$ ψηφία. Το σύμβολο $\lfloor x \rfloor$ συμβολίζει τον μεγαλύτερο ακέραιο αριθμό που δεν είναι μεγαλύτερος ή ίσος με τον x, όπου x μπορεί να είναι ακέραιος ή πραγματικός αριθμός. Ένα σύντομο παράδειγμα για την αναπαράσταση του αριθμού $(10)_{10}$. Τα απαιτούμενα bits που χρειαζόμαστε είναι:

$$\lfloor \log_2 10 \rfloor + 1 = \lfloor 3.322 \rfloor + 1 = 3 + 1 = 4$$

Για την μετατροπή ενός αριθμού από το δυαδικό στο δεκαδικό σύστημα αρκεί να εφαρμοστεί ο τύπος 2.1 για να υπολογιστεί το μέτρο του. Για την μετατροπή ενός δεκαδικού αριθμού στο δυαδικό σύστημα μπορούμε να ακολουθήσουμε την παρακάτω μεθοδολογία:

- Κάνουμε συνεχείς ακέραιες διαιρέσεις του δεκαδικού αριθμού με το δύο, μετά από κάθε διαίρεση κρατάμε το υπόλοιπο της διαίρεσης και το πηλίκο της.
- Το υπόλοιπο της διαίρεσης αντιστοιχεί στο LSB του δυαδικού αριθμού, ενώ στο πηλίκο που προέκυψε κάνουμε νέα ακέραια διαίρεση και κρατάμε το νέο πηλίκο που αντιστοιχεί στο επόμενο bit του αριθμού.
- Επαναλαμβάνουμε τη διαδικασία στο νέο υπόλοιπο μέχρι το πηλίκο που προκύπτει να είναι ίσο με μηδέν.

Παραδείγματος χάρη, ο αριθμός $(18)_{10}$ μετατρέπεται στο δυαδικό σύστημα ως εξής:

$$18/2 = 9 \; και \; υπόλοιπο \; 0$$

$$9/2 = 4 \; και \; υπόλοιπο \; 1$$

$$4/2 = 2 \; και \; υπόποιπο \; 0$$

$$2/2 = 1 \; και \; υπόλοιπο \; 0$$

$$1/2 = 0 \; και \; υπόλοιπο \; 1$$

Άρα δυαδικός ισοδύναμος του αριθμού 18 στο δεκαδικό είναι ο 10010 στο δυαδικό σύστημα΄. Η παραπάνω διαδικασία μπορεί να εφαρμοστεί και στα άλλα συστήματα αρκεί να αντικατασταθεί το 2 με τη βάση του συστήματος στο οποίο γίνεται μετατροπή.

2.3.1 Αναπαράσταση προσημασμένων αριθμών

Στη προηγούμενη παράγραφο αναφέρθηκε πως είναι δυνατή η αναπαράσταση οποιουδήποτε αριθμού στο δυαδικό σύστημα, όμως η διαδικασία που αναφέρθηκε αφορούσε μόνο για μη προσημασμένους αριθμούς (αριθμούς που δεν μας ενδιαφέρει το πρόσημό τους), Η αναπαράσταση των προσημασμένων αριθμών είναι περισσότερο περίπλοκη καθώς κατά την διάρκεια των χρόνων έχουν αναπτυχθεί πολλές μέθοδοι για την αναπαράστασή τους. Στις περισσότερες από αυτές τις μεθόδους το MSB καθορίζει το πρόσημο του αριθμού, για παράδειγμα έστω ο αριθμός

$$N = (d_{n-1} d_n \dots d_0)_2$$

Το bit d_{n-1} καθορίζεται από το πρόσημο του αριθμού N δηλαδή

$$d_{n-1} = \begin{cases} 0 & Αν\ N \geq 0 \\ 1 & Αν\ N < 0 \end{cases}$$

Σε κάθε αναπαράσταση αν το MSB του αριθμού είναι μηδέν δηλαδή πρόκειται για θετικό αριθμό τότε το μέτρο του αριθμού δίνεται από τον τύπο (2.1). Αν το MSB του αριθμού είναι όμως άσος, τότε το μέτρο του αριθμού εξαρτάται από την μέθοδο αναπαράστασης που χρησιμοποιείται. Υπάρχουν τρεις κύριες μορφές αναπαράστασης των προσημασμένων αριθμών οι οποίες θα αναλυθούν στη συνέχεια:

I. Μορφή προσήμου-μέτρου (Sign-Magnitude form)
II. Μορφή συμπληρώματος ως προς ένα (One's complement form)
III. Μορφή Συμπληρώματος ως προς δύο (Two complement form)

Μορφή προσήμου-μέτρου

Η αξία ενός δυαδικού αριθμού ο οποίος είναι γραμμένος σε μορφή προσήμου-μέτρου δίνεται από τον ακόλουθο τύπο

$$N = \begin{cases} \sum_{i=0}^{n-2} d_i & Αν\ d_{n-1} = 0 \\ -1 * \sum_{i=0}^{n-2} d_i & Αν\ d_{n-1} = 1 \end{cases}$$

Η μορφή προσήμου-μέτρου είναι η μορφή που είναι η περισσότερο κοντινή σε αυτή που χρησιμοποιεί ο άνθρωπος στο δεκαδικό σύστημα και με την οποία θα ασχολείται περισσότερο αυτή η διπλωματική εργασία. Το μέτρο των αριθμών μεγέθους n σε αυτή τη μορφή αναπαράστασης εξαρτάται από τα bits $d_{n-2}d_{n-3}\dots d_0$ και το πρόσημο του αριθμού καθορίζεται από το MSB d_{n-1} όπως αναφέρθηκε και προηγουμένως, για μηδενικό MSB προκύπτουν θετικοί αριθμοί ενώ για MSB ίσο με την μονάδα αρνητικοί, Αναφέρονται μερικά παραδείγματα για αριθμούς μεγέθους $n = 4$ bits για καλύτερη κατανόηση.

$$0110 = +(1 * 2^2 + 1 * 2^1 + 0 * 2^0) = +6$$

$$1011 = -(0 * 2^2 + 1 * 2^1 + 1 * 2^0) = -3$$

$$1110 = -(1 * 2^2 + 1 * 2^1 + 0 * 2^0) = -6$$

Παρόλου που αυτή η αναπαράσταση είναι πολύ κοντά στην ανθρώπινη μορφή επικοινωνίας δεν σημαίνει ότι είναι και αποδοτική για τη σχεδίαση αριθμητικών ψηφιακών κυκλωμάτων. Το βασικό μειονέκτημα είναι ότι χρειάζεται ξεχωριστό κύκλωμα για την πρόσθεση και την αφαίρεση αριθμών αυτής της μορφής αφού τα bit έχουν άλλοτε αρνητική και άλλοτε θετική αξία ανάλογα με τη τιμή του MSB, θα αναφερθούν σε επόμενη παράγραφο οι ακριβείς δυσκολίες που αντιμετωπίζει κανείς στη σχεδίαση όταν χρησιμοποιεί αυτή τη μορφή αναπαράστασης καθώς και τους τρόπους για να τις ξεπεράσει.

Μορφή συμπληρώματος ως προς ένα

Η αξία ενός δυαδικού αριθμού ο οποίος αναπαρίσταται σε μορφή συμπληρώματος ως προς ένα δίνεται από τον ακόλουθο τύπο:

$$N = -d_{n-1} * (2^{n-1} - 1) + \sum_{i=0}^{n-2} d_i * 2^i \qquad (2.2)$$

Έστω N θετικός αριθμός, σε αυτή τη μορφή αναπαράστασης ο αντίθετος του N $(-N)$ αναπαρίσταται με τον εξής τρόπο:

$$1\ \overline{d_{n-2}}\ \overline{d_{n-1}} \dots \overline{d_0}$$

Όπου

- $\overline{d_\iota} = (2-1) - d_i \quad 0 \le i \le n-2$
- d_i τα bits του αντίστοιχου θετικού αριθμού

Προφανώς προκύπτει ότι το $\overline{d_\iota}$ είναι το αντίστροφο του bit d_i δηλαδή όταν το ένα είναι μηδέν το άλλο είναι ένα και το αντίστροφο. Για να αναπαρασταθεί ο αντίθετος αριθμός του N $(-N)$ αντιστρέφονται όλα τα bit του αριθμού N, η ίδια διαδικασία ισχύει και αντίστροφα για την

μετατροπή αρνητικών αριθμών σε θετικούς. Παρουσιάζονται μερικά παραδείγματα για καλύτερη κατανόηση.

$$0110 = -0 * (2^3 - 1) + (1 * 2^2 + 1 * 2^1 + 0 \ * \ 2^0) = +6 = -(1001)$$

$$1011 = -1 * (2^3 - 1) + (0 * 2^2 + 1 * \ 2^1 + 1 * \ 2^0) = (3 - 7) = -\ 4 = -(0100)$$

$$1110 = -1 * (2^3 - 1) + (1 * 2^2 + 1 * \ 2^1 + 0 * \ 2^0) = (6 - 7) = -1 = \ -(0001)$$

Μορφή συμπληρώματος ως προς δύο

Η μορφή συμπληρώματος ως προς δύο, είναι η πιο διαδεδομένη μορφή στην αναπαράσταση των προσημασμένων αριθμών στο δυαδικό σύστημα. Σε αυτή τη μορφή αναπαράστασης το MSB έχει αρνητική αξία ενώ όλα τα υπόλοιπα έχουν πάντα θετική. Η τιμή ενός δυαδικού αριθμού N που αποτελείται από n bits και είναι σε μορφή συμπληρώματος ως προς 2 δίνεται από τον παρακάτω τύπο:

$$N = \ -d_{n-1} * 2^{n-1} + \ \sum_{i=0}^{n-2} d_i * \ 2^i \qquad (2.3)$$

Σε αυτή τη μορφή αναπαράστασης η μορφή των αρνητικών αριθμών είναι η ακόλουθη:

$$1\overline{d_{n-2}}\ \overline{d_{n-1}} \dots \ \overline{d_0} + 1$$

Όπου

- $\overline{d_\iota} = (2 - 1) - \ d_i \quad 0 \leq i \ \leq n - 2$
- d_i τα bits του αντίστοιχου θετικού αριθμού

Έστω ο αριθμός *N*, όπως φαίνεται για την αναπαράσταση του αντίθετου του *N* αρκεί να αντιστρέψουμε όλα τα bit του και έπειτα να προσθέσουμε μία μονάδα. Αναφέρονται επίσης μερικά παραδείγματα αναπαράστασης της μορφής συμπληρώματος ως προς δύο για καλύτερη κατανόηση

$$0110 = -(0 * 2^3) + (1 * 2^2 + 1 * 2^1 + 0 \ * \ 2^0) = +6 = -(1010)$$

$$1011 = -(1 * 2)^3 + (0 * 2^2 + 1 * \ 2^1 + 1 * \ 2^0) = (3 - 8) = -\ 5 = -(0101)$$

$$1110 = -(1 * 2)^3 + (1 * 2^2 + 1 * \ 2^1 + 0 * \ 2^0) = (6 - 8) = -2 = \ -(0010)$$

Τέλος παρουσιάζεται ένας πίνακας με όλους του αριθμούς μεγέθους τριών bit και η αξία τους ανάλογα με την μορφή αναπαράστασης που χρησιμοποιείται.

Πίνακας 2.1 *Αξία αριθμών μεγέθους 3 bit ανάλογα με τη μορφή αναπαράστασης.*

Δυαδικός αριθμός	πρόσημο-μέτρο	Συμπλήρωμα ως προς 1	Συμπλήρωμα ως προς 2
000	+0	+0	+0
001	+1	+1	+1
010	+2	+2	+2
011	+3	+3	+3
100	-0	-3	-4
101	-1	-2	-3
110	-2	-1	-2
111	-3	-0	-1

Παρατηρείται ότι η μορφή του συμπληρώματος ως προς 2 έχει το πλεονέκτημα ότι μπορεί να αναπαραστήσει έναν αριθμό περισσότερο από τις άλλες αναπαραστάσεις διότι το μηδέν έχει μοναδική αναπαράσταση σε αυτή τη μορφή ενώ οι άλλες μορφές έχουν δύο αναπαραστάσεις του μηδέν (μία για θετικό και μία για αρνητικό). Δηλαδή με έναν αριθμό N μεγέθους n bit είναι δυνατή η αναπαράσταση τιμών αξίας $2^{n-1} - 1 < N < -(2^{n-1} - 1)$ στη μορφή πρόσημου-μέτρου και συμπληρώματος ως προς ένα και $2^{n-1} - 1 < N < -2^n$στη μορφή συμπληρώματος ως προς δύο.

2.3.2 Αριθμητικές πράξεις στο δυαδικό σύστημα

Στην προηγούμενη παράγραφο αναλύθηκαν οι δυνατοί τρόποι αναπαράστασης θετικών και αρνητικών αριθμών στο δυαδικό σύστημα, σε αυτή τη παράγραφο θα αναλυθούν οι στοιχειώδεις αριθμητικές πράξεις όπως η πρόσθεση και η αφαίρεση δύο αριθμών.

Η λογική των αριθμητικών πράξεων όταν επεξεργάζονται μη-προσημασμένοι αριθμοί είναι πλήρως ανάλογη με τις πράξεις στο δεκαδικό σύστημα. Όταν αθροίζονται δύο αριθμοί στο δεκαδικό σύστημα, αθροίζονται τα ψηφία ίδιου βάρους ξεκινώντας από το λιγότερο σημαντικό, και αν το άθροισμα των ψηφίων υπερβαίνει τη βάση (δηλαδή το 10) προστίθενται ένα ‘κρατούμενο’ στο αποτέλεσμα της επόμενης βαθμίδας, κρατώντας μόνο το λιγότερο σημαντικό ψηφίο από την άθροιση, για παράδειγμα η πρόσθεση του 3_{10} στο 6_{10} στο δυαδικό σύστημα είναι:

$$\begin{array}{r} 1\,1\,0 \\ \hat{\ }\,\hat{1}\,\hat{1}\,0 \\ +\,0\,1\,1 \\ \hline 1\,0\,0\,1 \end{array}$$

Όπου οι τα ψηφία πάνω από τον πρώτο αριθμό αντιστοιχούν στα κρατούμενα της επόμενης βαθμίδας. Παρατηρείται ότι με την πρόσθεση δύο αριθμών μεγέθους n bit χρειάζονται n + 1 bit για να την αναπαράσταση του αποτελέσματος, παρόμοια η πράξη της δυαδικής αφαίρεσης δύο θετικών αριθμών βρίσκεται σε πλήρη αντιστοιχία με αυτήν της δεκαδικής αφαίρεσης. Δηλαδή όταν γίνεται η αφαίρεση ψηφίων ίδιου βάρους αν το ψηφίο του αφαιρετέου είναι μικρότερο από αυτό του αφαιρέτη τότε προστίθεται η βάση (δηλαδή το 10) στον αφαιρετέο και προστίθεται ένα

‘δανεικό’ στον αφαιρέτη του αμέσως αριστερού ψηφίου. Για παράδειγμα η αφαίρεση του 3_{10} από το 6_{10} στο δυαδικό σύστημα είναι:

```
 0 1 1
 ^ ^ ^
  1 1 0
- 0 1 1
-------
 0 0 1 1
```

Αν πραγματοποιηθεί η πράξη της αφαίρεσης με αφαιρετέο μεγαλύτερο από τον αφαιρέτη παρατηρείται ότι το αποτέλεσμα θα είναι αρνητικό στη μορφή συμπληρώματος ως προς δύο, με το πιο σημαντικό bit το d_n. Η δυαδική αφαίρεση μπορεί να γίνει έμμεσα μέσω της πρόσθεσης του αντίθετου αριθμού από τη γνωστή ιδιότητα της πρόσθεσης: $A - B = A + (-B)$ αλλά για να προκύψουν σωστά αποτελέσματα πρέπει οι αρνητικοί αριθμοί να είναι στη μορφή συμπληρώματος ως προς δύο.

Η πράξη του πολλαπλασιασμού γίνεται και αυτή όμοια με το πολλαπλασιασμό στο δεκαδικό σύστημα. Ξεκινώντας από το LSB του πολλαπλασιαστή παράγουμε τα μερικά γινόμενα το καθένα μετατοπισμένο μία θέση αριστερά από το προηγούμενο. Επειδή στο δυαδικό σύστημα τα ψηφία είναι μόνο μηδέν και ένα τα μερικά γινόμενα θα είναι είτε ίσα με το πολλαπλασιαστέο όταν το ψηφίο του πολλαπλασιαστή είναι μονάδα, είτε ίσο με το μηδέν όταν το ψηφίο του πολλαπλασιαστή είναι ίσο με το μηδέν. Για παράδειγμα ο πολλαπλασιασμός του 3_{10} με το 6_{10} είναι:

```
    0 1 1
  X 1 1 0
  -------
    0 0 0
  0 1 1
+ 0 1 1
-------
  1 0 0 1 0
```

Παρατηρείται ότι για πολλαπλασιασμό δύο αριθμών μεγέθους n-bit χρειάζονται 2n bit για την αναπαράσταση του αποτελέσματος, τα ακόλουθα ισχύουν για θετικούς αριθμούς, Αν γίνει πολλαπλασιασμός δύο προσημασμένων αριθμών, τότε αν οι αριθμοί βρίσκονται στη μορφή πρόσημου-μέτρου τότε μπορούν να πολλαπλασιαστούν τα μέτρα τους, και σαν περισσότερο σημαντικό bit στο αποτέλεσμα να τοποθετηθεί η μονάδα αν οι δύο αριθμοί έχουν διαφορετικό πρόσημο ή το μηδέν αν οι δύο αριθμοί έχουν το ίδιο πρόσημο, από τη γνωστή ιδιότητα του πολλαπλασιασμού. Αν οι αριθμοί βρίσκονται στη μορφή αναπαράστασης συμπληρώματος ως προς δύο, τότε η διαδικασία είναι πιο περίπλοκη, στο πολλαπλασιασμό πρέπει να ληφθεί υπόψη η αρνητική αξία του περισσότερου σημαντικού bit και των δύο αριθμών, αυτό σημαίνει ότι αν ο πολλαπλασιαστής είναι αρνητικός το τελευταίο μερικό γινόμενο θα αποτελείται από τα ανεστραμμένα bit του πολλαπλασιαστή συν μία μονάδα για να προκύψει το σωστό μερικό γινόμενο, στη περίπτωση που ο πολλαπλασιαστέος είναι αρνητικός πρέπει να εφαρμοστεί προέκταση πρόσημου (sign extension) στα μερικά γινόμενα ώστε να έχουν αρνητική αξία στην τελική πρόσθεση, για παράδειγμα αν οι δύο αριθμοί που πολλαπλασιάσαμε πριν ήταν προσημασμένοι στη μορφή συμπληρώματος ως προς δύο τότε ο πολλαπλασιασμός θα είχε την ακόλουθη μορφή:

```
        0  1  1        +3
     X  1  1  0        -2
  ---------------
        0  0  0
     0  1  1
  1  0  0
+       1
  ---------------
  1  1  0  1  0        -6
```

Παρατηρείται ότι για το πολλαπλασιασμό δύο προσημασμένων αριθμών μεγέθους n-bit το αποτέλεσμα χρειάζεται 2*n-1 για να αποτυπωθεί σωστά, αυτό συμβαίνει γιατί το μέτρο των αριθμών μπορεί να αναπαρασταθεί σε n-1 bit οπότε το μέτρο του αποτελέσματος χρειάζεται 2*n-2 bit για σωστή αναπαράσταση, συν ένα επιπλέον ψηφίο για το πρόσημο του αποτελέσματος, άρα 2*n-1 bit συνολικά. Στο παράδειγμα αυτό μπορεί να παρατηρηθεί το πλεονέκτημα της αναπαράστασης σε μορφή πρόσημου-μέτρου όταν πρόκειται για το πολλαπλασιασμό δύο αριθμών, έναντι της μορφής συμπληρώματος ως προς δύο.

2.4 Αριθμητικά κυκλώματα VLSI

Προτού αναλύσουμε τα είδη ψηφιακών κυκλωμάτων που χρησιμοποιούνται για τις αριθμητικές πράξεις αναφέρονται σύντομα όλες οι δυνατές λογικές πράξεις (ή λογικές πύλες) μεταξύ των 2 bit καθώς και η πολυπλοκότητα της κάθε μίας σε επιφάνεια και καθυστέρηση

- AND
- OR
- XOR
- NAND
- NOR
- XNOR

Ο πίνακας αληθείας αυτών των λογικών πράξεων παρουσιάζεται παρακάτω:

Πίνακας 2.2 *Πίνακας αληθείας όλων των δυνατών λογικών πράξεων μεταξύ 2 bit*

BIT 1	**BIT 2**	***AND***	***OR***	***XOR***	***NAND***	***NOR***	***XNOR***
0	0	0	0	0	1	1	1
0	1	0	1	1	1	0	0
1	0	0	1	1	1	0	0
1	1	1	1	0	0	0	1

Η υλοποίηση κάθε λογικής πράξης προσθέτει μια καθυστέρηση και μία επιφάνεια στο συνολικό κύκλωμα, επειδή διαφορετικές τεχνολογίες έχουν διαφορετική καθυστέρηση και επιφάνεια στις λογικές πύλες τους, δεν θα καθοριστεί η επιφάνεια και η καθυστέρηση ως απόλυτη τιμή αλλά θα

χρησιμοποιηθεί ο συμβολισμός Δ_T για την καθυστέρηση σε χρόνο και A_T η επιφάνεια της πύλης και θα τις υπολογιστούν ως ακέραια πολλαπλάσια των T_G και A_G που είναι η καθυστέρηση και η επιφάνεια μιας πύλης NAND ή μιας πύλης NOR οι οποίες είναι οι πύλες με την ελάχιστη καθυστέρηση και επιφάνεια. Παρατίθεται ο πίνακας με τα χαρακτηριστικά της κάθε λογικής πύλης [1].

Πίνακας 2.3 *Επιφάνεια και καθυστέρηση των λογικών πυλών.*

Λογική πύλη	**Καθυστέρηση Δ_T**	**Επιφάνεια A_T**
AND	$2*T_G$	$2*A_G$
OR	$2*T_G$	$2*A_G$
NOT	$1*T_G$	$1*A_G$
XOR	$2*T_G$	$3*A_G$
NAND	$1*T_G$	$1*A_G$
NOR	$1*T_G$	$1*A_G$
XNOR	$2*T_G$	$3*A_G$

Στη συνέχεια παρουσιάζονται τα ψηφιακά κυκλώματα τα οποία δημιουργούνται με βάση τις παραπάνω λογικές πύλες.

2.5 Κυκλώματα αθροιστών

Στη παράγραφο 2.3.2 αναφέρθηκε πώς γίνεται η άθροιση δύο δυαδικών αριθμών, εύκολα μπορεί να σχεδιαστεί ο πίνακας αληθείας της άθροισης 2 bit και του κρατουμένου εισόδου.

Πίνακας 2.4 *Πίνακας αληθείας άθροισης.*

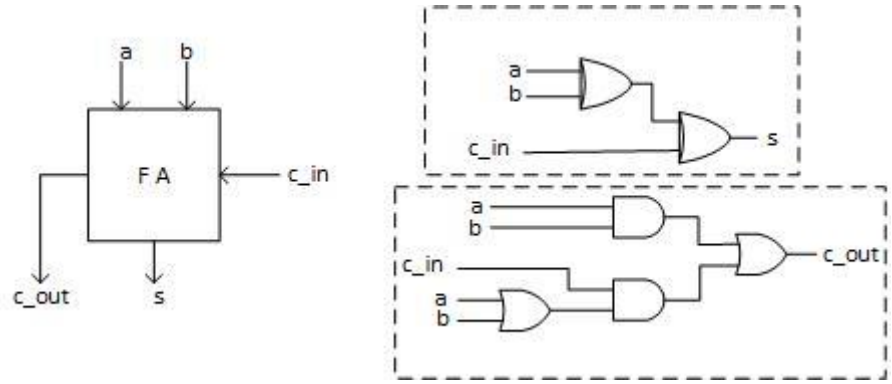
Είσοδοι			**Έξοδοι**	
a	b	c_{in}	c_{out}	s
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Το κύκλωμα που υλοποιεί την άθροιση 2 bit με το κρατούμενο εισόδου ονομάζεται πλήρης αθροιστής (Full Adder), σε αρκετές βιβλιογραφίες ο πλήρης αθροιστής αναφέρεται και ως μετρητής γιατί η έξοδος του είναι ο αριθμός μονάδων στις εισόδους του. Το αντίστοιχο κύκλωμα που υλοποιεί την άθροιση 2bit χωρίς κρατούμενο εισόδου ονομάζεται ημι-αθροιστής (Half adder) και έχει τον ίδιο πίνακα αληθείας με το πλήρη αθροιστή για κρατούμενο εισόδου ίσο με το μηδέν. Από τον πίνακα αληθείας εξάγουμε τις ακόλουθες λογικές συναρτήσεις για το s και το c_{in}

$$s = a \oplus b \oplus c_{in} \quad (2.4)$$

$$c_{out} = (a * b) + (a * c_{in}) + (b * c_{in}) \quad (2.5)$$

Σημειώνεται ότι τα σύμβολα " + " και " * " δεν αντιστοιχούν στη πρόσθεση και τον πολλαπλασιασμό αλλά στη λογική πράξη OR και AND αντίστοιχα, ενώ το σύμβολο "$\oplus$" αντιστοιχεί στη λογική πύλη XOR. Η μονάδα του πλήρη αθροιστή φαίνεται στο παρακάτω σχήμα.

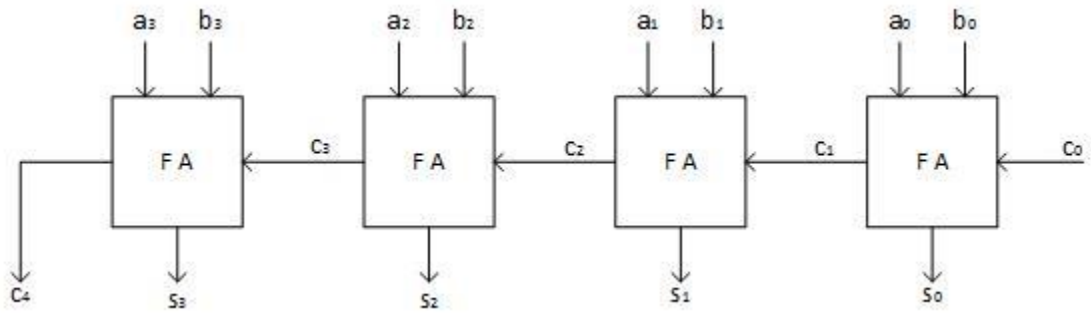


Σχήμα 2.1 Κύκλωμα πλήρη άθροιση σε επίπεδο πυλών.

Έχει παρατηρηθεί ότι ο πλήρης αθροιστής επιφέρει καθυστέρηση ίση με $4T_G$, ίση με την καθυστέρηση δύο πυλών XOR, δηλαδή καθορίζεται από την παραγωγή του s. Το κύκλωμα του πλήρη αθροιστή είναι η βασική μονάδα για την άθροιση δύο αριθμών, ο πιο γνωστός αθροιστής στα ψηφιακά κυκλώματα είναι ο αθροιστής διάδοσης κρατουμένου (ripple-carry adder) ο οποίος μελετάται στην επόμενη παράγραφο.

2.5.1 Αθροιστής διάδοσης κρατουμένου

Το κύκλωμα ενός αθροιστή διάδοσης κρατουμένου φαίνεται στο ακόλουθο σχήμα

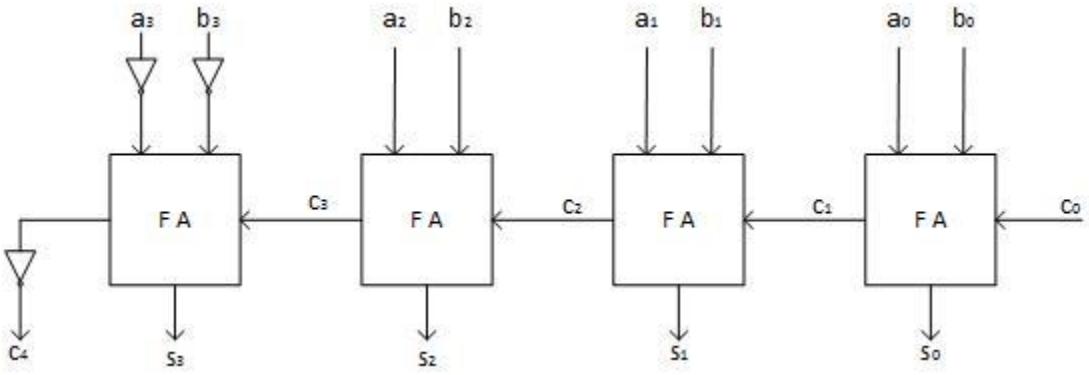


Σχήμα 2.2 Αθροιστής διάδοσης κρατουμένου.

Ο αθροιστής διάδοσης κρατουμένου είναι από τις βασικότερες και πιο συνηθέστερες μονάδες άθροισης, στο σχήμα φαίνεται η άθροιση δύο αριθμών μεγέθους 4 bit ενώ το αποτέλεσμα έχει μέγεθος 5 bit προκειμένου να καλυφθεί η περίπτωση που παρουσιάζεται υπερχείλιση. Στην περίπτωση που η άθροιση γίνεται μεταξύ προσημασμένων αριθμών το κύκλωμα του αθροιστή διάδοσης κρατουμένου πρέπει να προσαρμοστεί ανάλογα με την αναπαράσταση που χρησιμοποιείται.

- Συμπλήρωμα ως προς δύο.

Όταν χρησιμοποιείται η αναπαράσταση σε συμπλήρωμα ως προς δύο, τότε στο κύκλωμα της πρόσθεσης πρέπει να αντιστρέψουμε τα MSB των αριθμών και αυτό γιατί όπως αναφέρθηκε στο τρόπο αναπαράστασης τους τα πιο σημαντικά bit έχουν αρνητική αξία. Ο αθροιστής που προσθέτει δύο αριθμούς σε συμπλήρωμα ως προς δύο έχει αυτή τη μορφή:



Σχήμα 2.3 Αθροιστής διάδοσης κρατουμένου για συμπλήρωμα ως προς δύο αναπαράσταση

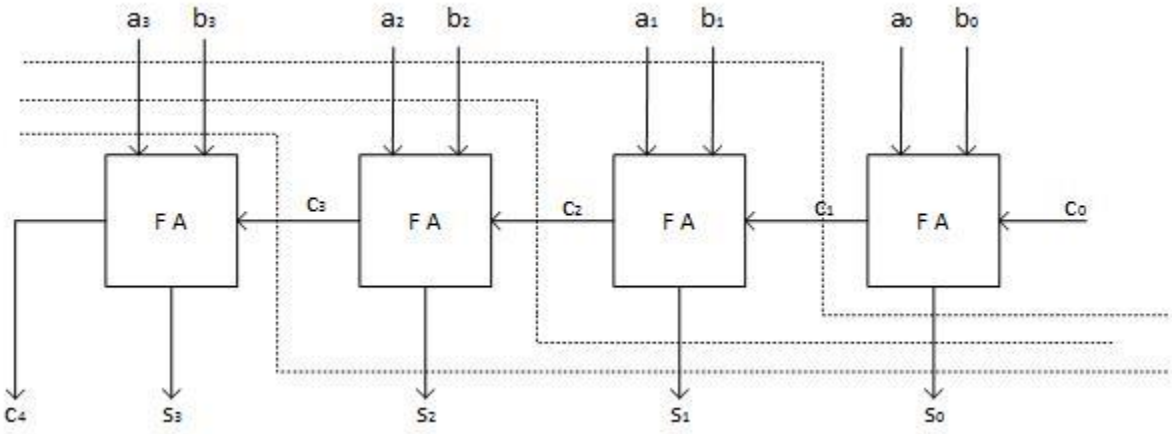
Όπως μπορεί να φανεί και από το την εικόνα του κυκλώματος το περισσότερο σημαντικό bit του αποτελέσματος, δηλαδή το κρατούμενο εξόδου της τελευταίας βαθμίδας του πλήρη αθροιστή, είναι το bit πρόσημου το οποίο καθορίζει αν ο αριθμός είναι θετικός η αρνητικός. Επιπλέον έχουμε υπερχείλιση, δηλαδή χρειαζόμαστε n+1 bit για την αναπαράσταση του αποτελέσματος όταν έχουμε πρόσθεση αριθμών ίδιου πρόσημου, η πρόσθεση ετερόσημων αριθμών παράγει αποτέλεσμα το οποίο πάντα μπορεί να αναπαρασταθεί από n bit. Επίσης σημειώνεται ότι στους αριθμούς συμπληρώματος ως προς δύο η αφαίρεση συνήθως γίνεται μέσω της πρόσθεσης, δηλαδή για την εκτέλεση της αριθμητικής πράξης A – B προστίθεται στον A ο αντίθετος του B δηλαδή υλοποιείται η πράξη A + (-B).

- Μορφή πρόσημου-μέτρου

Σε αυτήν την αναπαράσταση το κύκλωμα της εικόνας 2.2 λειτουργεί μόνο αν οι δύο αριθμοί οι οποίοι προστίθενται είναι ομόσημοι, τότε απλά γίνεται πρόσθεση των μέτρων τους και διατηρείται το πρόσημο του ενός από τους δύο. Όταν οι αριθμοί είναι ετερόσημοι τότε πρέπει να μετατραπεί ο αρνητικός αριθμός σε μορφή συμπληρώματος ως προς δύο. Το αποτέλεσμα θα προκύψει προφανώς σε μορφή συμπληρώματος ως προς δύο οπότε αν είναι αρνητικό χρειάζεται η εκ νέου μετατροπή του σε μορφή πρόσημου-μέτρου. Στο τρίτο κεφάλαιο θα παρουσιαστεί μία πειραματική διάταξη όπου θα σχεδιαστεί ένας αφαιρέτης απόλυτης τιμής ώστε αν αποφεύγεται η συνεχείς μετατροπή μεταξύ των δύο μορφών αναπαράστασης.

Ο αθροιστής διάδοσης κρατουμένου έχει καθυστέρηση $T_{\Delta} = n * 4T_G$ όπου n είναι το μήκος των αριθμών εισόδου, δηλαδή η συνολική του καθυστέρηση αυξάνεται αναλογικά με το μέγεθος των αριθμών οι οποίοι προστίθενται, για πολύ μικρά μήκη αριθμών όπως 3-4 bit η καθυστέρηση που προσθέτει στο συνολικό κύκλωμα είναι σχετικά μικρή, όταν όμως γίνεται επεξεργασία μεγάλων αριθμών, κάτι το οποίο είναι πιο συνηθέστερο, η καθυστέρηση με την οποία επιβαρύνει ο αθροιστής την συνολική διάταξη γίνεται απαγορευτική για τη χρήση του. Αυτός είναι και ο κύριος λόγος που έχουν αναπτυχθεί άλλα μοντέλα αθροιστών τα οποία εκτελούν την πράξη της

πρόσθεσης σε πολύ μικρότερο χρόνο. Σημειώνεται ότι για να βελτιωθεί η απόδοση ένας ripple-carry αθροιστή συνήθως εφαρμόζεται λειτουργία συνεχούς διοχέτευσης του κυκλώματος (pipelining). Η μεθοδολογία που χρησιμοποιείται είναι η εξής: με ευθείες γραμμές που διατρέχουν το κύκλωμα από άκρη σε άκρη, "κόβονται" όλα τα σήματα της ίδιας φοράς που συνδέουν δύο βαθμίδες και περνούν μέσα από λογικά κυκλώματα και καθυστερήσεις. Στα σημεία όπου κόβονται τα σήματα τοποθετούνται μανδαλωτές. Ένας συστολικός αθροιστής διάδοσης κρατουμένου φαίνεται στο ακόλουθο σχήμα:



Σχήμα 2.4 Λειτουργία συνεχούς διοχέτευσης αθροιστή διάδοσης κρατουμένου.

Χρησιμοποιώντας λειτουργία συνεχούς διοχέτευσης σε κυκλώματα δεν αυξάνεται η ταχύτητα λειτουργίας τους, αντίθετα επιβαρύνεται αν τα στάδια στα οποία χωρίζεται το κύκλωμα δεν είναι ισοσταθμισμένα σε καθυστέρηση, επιπλέον οι μανδαλωτές, με τους οποίου χωρίζονται τα στάδια προσθέτουν μία μικρή καθυστέρηση προκειμένου να μεταφέρουν τα δεδομένα στο επόμενο στάδιο. Με τη λειτουργία συνεχούς διοχέτευσης αυξάνεται ο ρυθμός λειτουργίας του κυκλώματος (Throughput), επειδή κάθε στάδιο επεξεργάζεται δεδομένα σε ταχύτερο ρυθμό. Όπως αναφέρθηκε και προηγουμένως, για να αθροίζονται αριθμοί μεγάλου μήκους η καθυστέρηση γίνεται πολύ μεγάλη, για να αυξηθεί η ταχύτητα της άθροισης μελετάται ένας άλλος τύπος αθροιστή, ο αθροιστής πρόβλεψης κρατουμένου.

2.5.2 Αθροιστής πρόβλεψης κρατουμένου

Στον προηγούμενο αθροιστή για να παραχθεί το κρατούμενο εξόδου του κάθε πλήρη αθροιστή έπρεπε να περιμένει την ολοκλήρωση όλων των προηγούμενων σταδίων. Ο αθροιστής πρόβλεψης κρατουμένου (Carry Look-ahead Adder) στηρίζεται στη λογική ότι η πληροφορία του κρατούμενου εισόδου δεν χρειάζεται πάντα για την παραγωγή του κρατούμενου εξόδου, για παράδειγμα όταν αθροίζονται δύο μονάδες ή δύο μηδενικά τότε το κρατούμενο εξόδου είναι μονάδα ή μηδέν αντίστοιχα ανεξάρτητα από τη τιμή του κρατούμενου εισόδου, αυτό που περιεγράφηκε είναι το σήμα παραγωγής (generate) το οποίο συμβολίζεται με g, αντίστοιχα όταν γίνεται άθροιση μεταξύ μίας μονάδας και ενός μηδενικού τότε το κρατούμενο εξόδου είναι ίσο με το κρατούμενο εισόδου, δηλαδή γίνεται διάδοση κρατουμένου (propagate), η οποία συμβολίζεται με p. Από τη παραπάνω περιγραφή προκύπτουν και οι τύποι των σημάτων παραγωγής και διάδοσης κρατουμένου:

$$g_i = x_i * y_i \qquad (2.6)$$

$$p_i = x_i \oplus y_i \qquad (2.7)$$

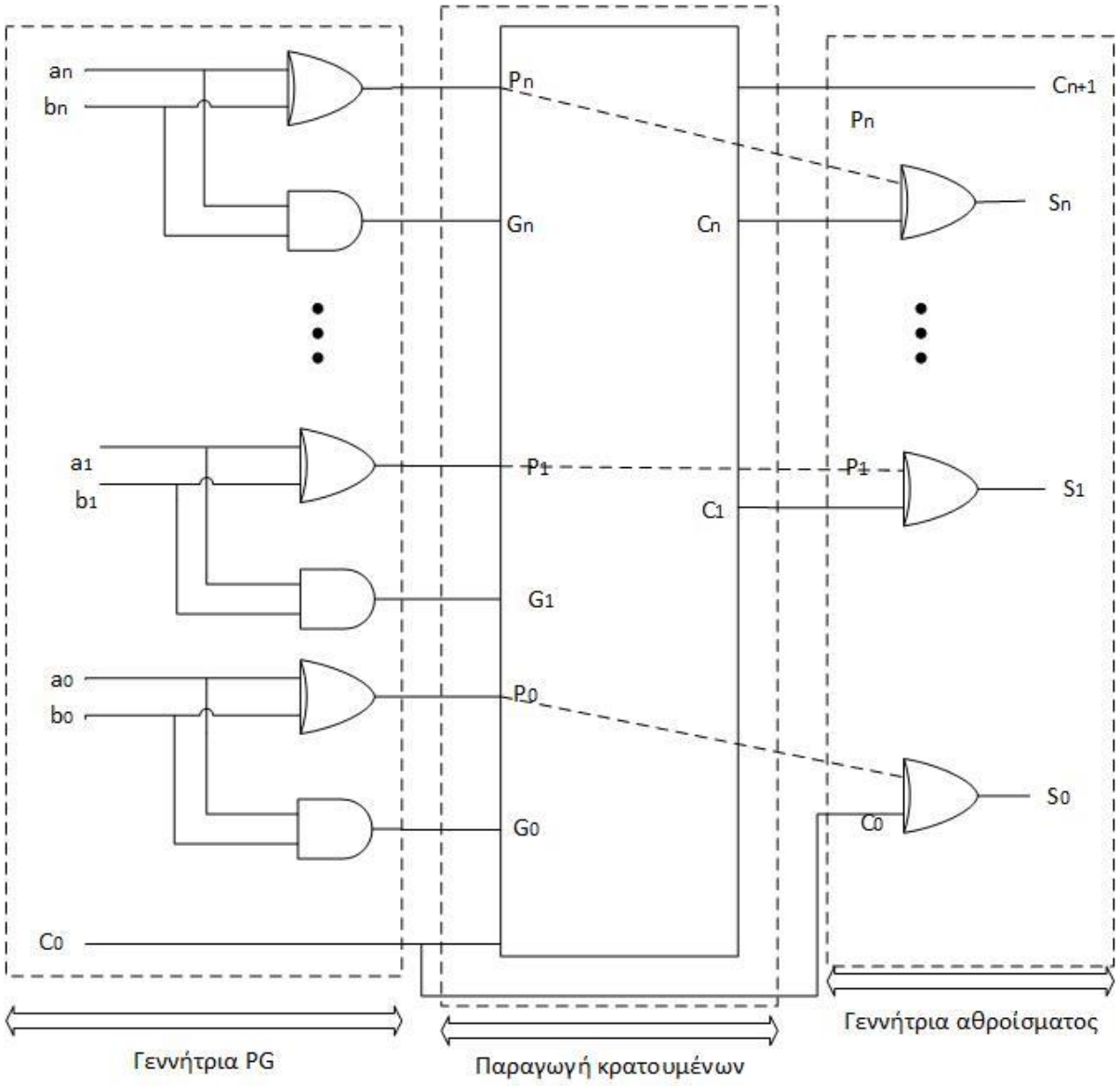
Παρατηρείται ότι οι συναρτήσεις των p και g ταυτίζονται με τις συναρτήσεις c_{out} και s αντίστοιχα ενός ημιαθροιστή. Συνεπώς για να υπάρχει κρατούμενο εξόδου στη βαθμίδα i πρέπει είτε να δημιουργείται λόγω σήματος παραγωγής κρατουμένου της προηγούμενης βαθμίδας, είτε να διαδίδεται λόγω του σήματος διάδοσης κρατουμένου της προηγούμενης βαθμίδας, δηλαδή η γενική μορφή της σχέσης για το κρατούμενο είναι η ακόλουθη

$$C_{n+1} = G_n + P_{n-1}(G_{n-2} + P_{n-2}(G_{n-3} + P_{n-3}(\dots + P_2(G_1 + P_1 C_1)) \qquad (2.8)$$

Άρα είναι δυνατός ο διαχωρισμός του αθροιστή σε τρία στάδια, το πρώτο το οποίο θα υπολογίζει τα σήματα g_i και p_i της κάθε βαθμίδας, το δεύτερο στάδιο στο οποίο θα υπολογίζονται τα κρατούμενα της κάθε βαθμίδας ανάλογα με τα σήματα g_i και p_i μέσω του τύπου 2.8, και το τρίτο στάδιο όπου θα υπολογίζονται τα s_i μέσω του τύπου

$$s = c_i \oplus x_i \oplus y_i = c_i \oplus p_i \qquad (2.9)$$

Συνεπώς η γενική εικόνα ενός αθροιστή διάδοσης κρατουμένου φαίνεται στο παρακάτω σχήμα:



Σχήμα 2.5 Γενική μορφή Αθροιστή πρόβλεψης κρατουμένου

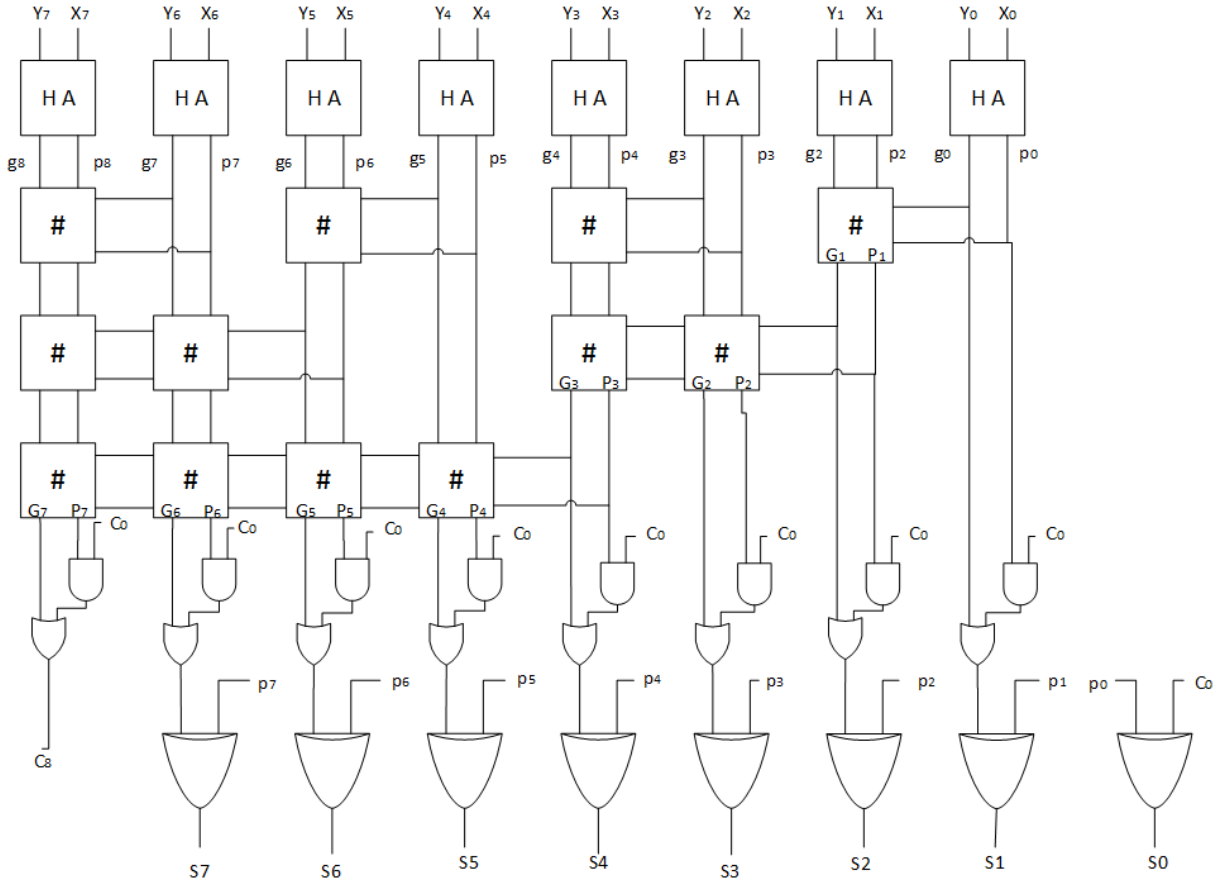
Το πιο σημαντικό κομμάτι λοιπόν ενός αθροιστή πρόβλεψης κρατουμένου είναι η παραγωγή κρατουμένων πάνω στο οποίο έχουν γίνει και πολλές μελέτες ώστε ο υπολογισμός να γίνεται όσο το δυνατόν πιο αποδοτικά. Στην εργασία αυτή θα χρησιμοποιηθεί το μοντέλο που προτάθηκε από τους Brent και Kung, το οποίο προτείνει το παράλληλο υπολογισμό των κρατουμένων μέσω ενός δυαδικού δέντρου με $log_2 n$ λογικά επίπεδα. Η λογική πάνω στην οποία βασίζεται αυτό το μοντέλο είναι ο διαχωρισμός των σημάτων παραγωγής και διάδοσης κρατουμένων σε υποσύνολα και υπολογισμός των κρατουμένων αυτών των υποσυνόλων χρησιμοποιώντας έναν καινούργιο τελεστή ο οποίος συμβολίζεται με "#" και ορίζεται ως:

$$(G, P) = (G_2, P_2)\#(G_1, P_1) = (G_2 + G_1 * P_2, P_1 * P_2) \qquad (2.10)$$

Αποδεικνύεται ότι ο τελεστής “#” είναι προσεταιριστικός δηλαδή:

$$(G_3, P_3)\#[(G_2, P_2)\#(G_1, P_1)] = [(G_3, P_3)\#(G_2, P_2)]\#(G_1, P_1) \qquad (2.11)$$

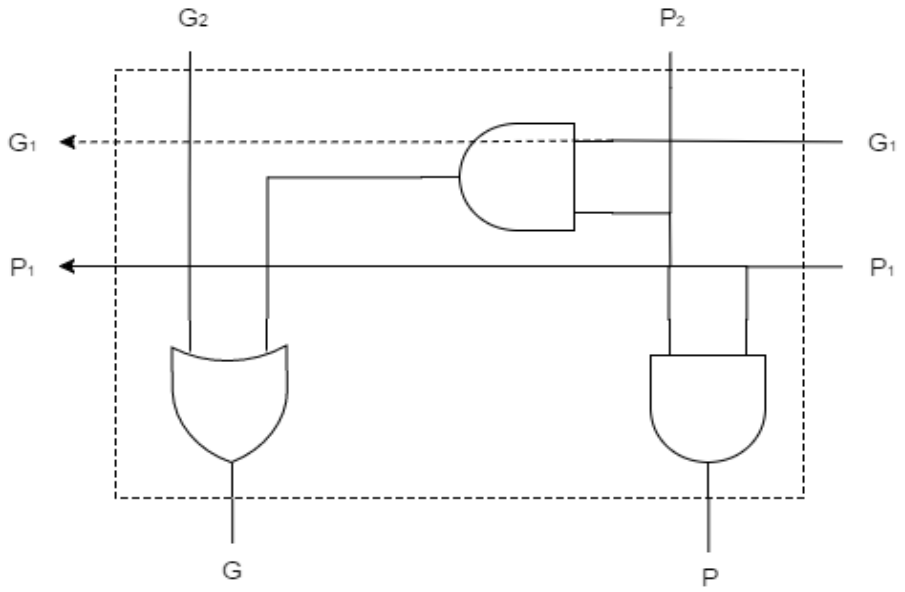
έτσι επιτρέπεται ο παράλληλος υπολογισμός των P και G. Η μορφή αθροιστή πρόβλεψης κρατουμένου με τη χρήση του δυαδικού δέντρου για μέγεθος εισόδων ίσο με 8bit φαίνεται στο επόμενο σχήμα:



Σχήμα 2.6 Αθροιστής πρόβλεψης κρατουμένου με χρήση δυαδικού δέντρου για δύο αριθμούς μήκους 8 bit.

Από το σχήμα φαίνεται ότι για εισόδους μεγέθους 8bit το Critical Path του αθροιστή είναι ίσο με την καθυστέρηση ενός ημιαθροιστή συν τα τρία λογικά επίπεδα από τελεστές “#” συν το επίπεδο τελικού υπολογισμού κρατουμένου λόγω του κρατουμένου εισόδου C_0 συν το λογικό επίπεδο μίας πύλης XOR. Η καθυστέρηση των ημιαθροιστών και των πυλών XOR είναι σταθερή αφού είναι ανεξάρτητη του μεγέθους των εισόδων, αντίθετα τα λογικά επίπεδα του δέντρου αυξάνονται λογαριθμικά με το μέγεθος των εισόδων, για παράδειγμα για εισόδους μεγέθους 64bit θα υπήρχαν

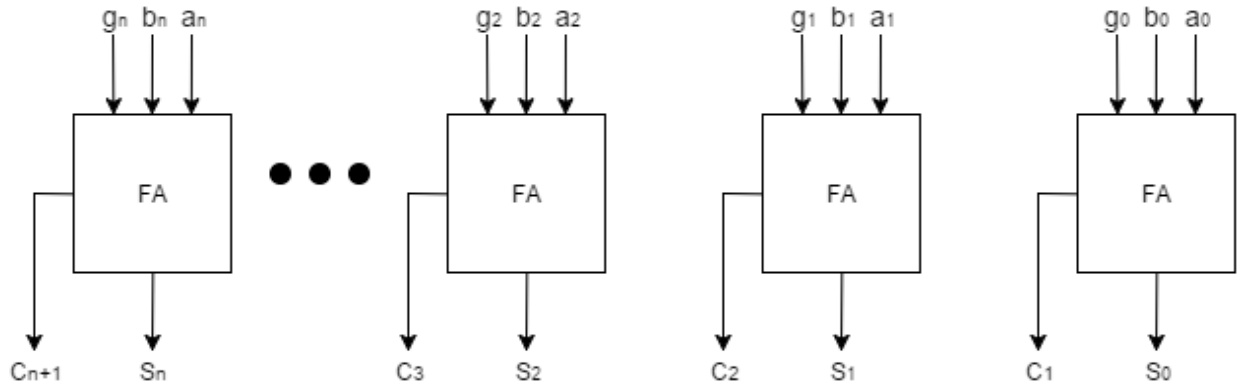
μόλις 6 λογικά επίπεδα του τελεστή "#" ενώ σε έναν αθροιστή διάδοσης κρατουμένου υπήρχε καθυστέρηση ίση με 64 επίπεδα ενός πλήρη αθροιστή. Αν στην άθροιση χρησιμοποιούνται αριθμοί στη μορφή συμπληρώματος ως προς δύο, τότε πρέπει να αντιστραφούν τα MSB των αριθμών καθώς και το κρατούμενο εξόδου προκειμένου η τελική αναπαράσταση σε n bit να είναι ορθή. Επιπλέον σε αρκετές εφαρμογές η άθροιση των δύο αριθμών γίνεται αρκετά συχνά με μηδενικό κρατούμενο εισόδου, σε αυτήν την περίπτωση η μονάδα υπολογισμού του κρατουμένου μέσω της σχέσης $C_i = G_i + P_i * C_0$ μπορεί να παραληφθεί αφού όταν το κρατούμενο εισόδου είναι μηδέν τότε τα κρατούμενα της κάθε βαθμίδας ταυτίζονται με τα τελικά σήματα παραγωγής κρατουμένου. Για την επεξεργασία αριθμών σε μορφή πρόσημου-μέτρου πολλές φορές θα χρειαστεί ο έλεγχος του κρατουμένου οπότε χρησιμοποιείται το σχήμα του αθροιστή που παρουσιάστηκε. Η πράξη «δίεση» σε επίπεδο λογικών πυλών φαίνεται στο ακόλουθο σχήμα:



Σχήμα 2.7 Κύκλωμα που υλοποιεί την πράξη "#"

2.5.3 Αθροιστής με σώσιμο κρατουμένου

Από τον πίνακα αληθείας του πλήρη αθροιστή (πίνακας 2.4) μπορεί κάποιος να τον παρομοιάσει ως έναν συμπιεστή, λόγω ότι δέχεται τρία bit σαν εισόδους και σαν εξόδους παράγει μόνο δύο. Πάνω σε αυτή τη λογική υλοποιείται ο αθροιστής με σώσιμο κρατουμένου (Carry-Save adder), ο οποίος δέχεται σαν είσοδο τρεις αριθμούς και παράγει στην έξοδό του δύο διανύσματα, το άθροισμα των οποίων είναι και το τελικό άθροισμα των τριών αριθμών. Η μορφή ενός αθροιστή με σώσιμο κρατούμενου φαίνεται στο ακόλουθο σχήμα:



Σχήμα 2.8 Αθροιστής με σώσιμο κρατουμένου

Τα διανύσματα S και C προκύπτουν στην μορφή $C = \{C_{n+1} \cdots C_2 C_1 0\}$ και $S = \{S_n \cdots S_2 S_1 S_0\}$, αντί για μηδέν στο LSB του C μπορεί να εισαχθεί ένα κρατούμενο εισόδου, επίσης το διάνυσμα C είναι κατά ένα bit μεγαλύτερο από το διάνυσμα S, τα bit που παράγονται από τους πλήρους αθροιστές για το διάνυσμα C είναι μετατοπισμένα μία θέση αριστερά αφού σαν κρατούμενα έχουν μεγαλύτερο βάρος από τα bit του S. Η αναπαράσταση του τελικού αθροίσματος σε μορφή δύο διανυσμάτων συνήθως ονομάζεται αναπαράσταση αθροίσματος-κρατουμένου, έχει τα μειονεκτήματα ότι το αποτέλεσμα δεν είναι άμεσα αντιληπτό και επιπλέον η απεικόνιση του αποτελέσματος σε αυτή τη μορφή δεν είναι αμφιμονοσήμαντη, δηλαδή δεν υπάρχει μοναδικός τρόπος αναπαράστασης κάθε αριθμού σε μορφή αθροίσματος-κρατουμένου, έτσι για τον υπολογισμό του τελικού αποτελέσματος πρέπει να χρησιμοποιηθεί είτε ένας αθροιστής διάδοσης κρατουμένου είτε ένας αθροιστής πρόβλεψης κρατουμένου. Το βασικό πλεονέκτημα αυτής της μονάδας άθροισης είναι ότι όλοι οι πλήρεις αθροιστές λειτουργούν ανεξάρτητα μεταξύ τους οπότε η συνολική καθυστέρηση όλης της μονάδας είναι ίση με ενός πλήρη αθροιστή, η οποία αναλύθηκε πριν και βρέθηκε ίση με $4*T_g$. Η χρήση του αθροιστή με σώσιμο κρατουμένου είναι συνηθέστερη όταν χρειαζόμαστε τον υπολογισμό του αθροίσματος πολλών αριθμών, σε αυτή τη περίπτωση αντί να προσθέσουμε όλους τους αριθμούς ξεχωριστά μπορούμε να χρησιμοποιήσουμε carry-save αθροιστές παράλληλα ώστε να συμπιέσουμε όλους τους αριθμούς σε δύο διανύσματα και μετά να εκτελέσουμε την τελική πρόσθεση. Στη συνέχεια όπου θα αναλυθούν τα κυκλώματα που εκτελούν τον πολλαπλασιασμό, θα εξεταστεί η χρήση αυτού του είδους αθροιστών ώστε να γίνεται ο γρήγορος υπολογισμός του αθροίσματος των μερικών γινομένων.

2.6 Κυκλώματα πολλαπλασιαστών

Σε προηγούμενη παράγραφο παρουσιάστηκε ο τρόπος με τον οποίο γίνεται πολλαπλασιασμός δύο θετικών αριθμών και παρατηρήθηκε ότι βρίσκεται σε πλήρη αντιστοιχία με τον πολλαπλασιασμό στο δεκαδικό σύστημα. Σε επίπεδο λογικού κυκλώματος παρατηρείται ότι η δημιουργία του κάθε μερικού γινομένου μπορεί να παραχθεί από τον λογική πράξη AND μεταξύ του κάθε bit πολλαπλασιαστή και ενός bit του πολλαπλασιαστέου. Κάθε μερικό γινόμενο που παράγεται είναι μετατοπισμένο κατά μία θέση αριστερά από το προηγούμενο του αφού το bit με το οποίο εκτελείται η λογική πράξη AND είναι μία θέση πιο σημαντικό, η λογική αυτή παρουσιάζεται στον ακόλουθο πίνακα.

Πίνακας 2.5 *Παραγωγή μερικών γινομένων για πολλαπλασιαστή 4-bit.*

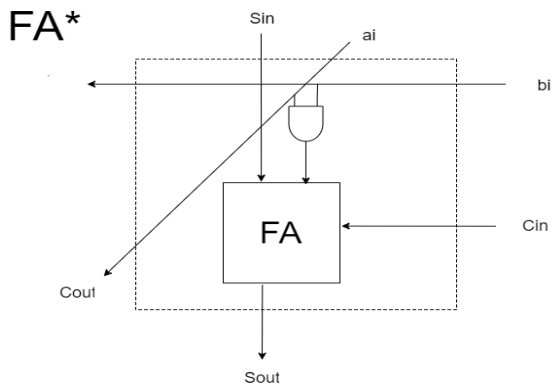
				A3 B3	A2 B2	A1 B1	A0 B0	**Πολ/στέος** **Πολ/στής**
				A3*B0	A2*B0	A1*B0	A0*B0	**ΜΕΡΙΚΑ**
			A3*B1	A2*B1	A1*B1	A0*B1		**ΓΙΝΟΜΕΝΑ**
		A3*B2	A2*B2	A1*B2	A0*B2			
	A3*B3	A2*B3	A1*B3	A0*B3				
P7	P6	P5	P4	P3	P2	P1	P0	**ΓΙΝΟΜΕΝΟ**

Όπου το σύμβολο του πολλαπλασιασμού συμβολίζει τη λογική πράξη AND μεταξύ των δύο bit. Από τον πίνακα 2.5 λαμβάνονται τα ακόλουθα σημαντικά συμπεράσματα. Το αποτέλεσμα του πολλαπλασιασμού δυο αριθμών μεγέθους τεσσάρων bit χρειάζεται οχτώ bit για να παραχθεί, αυτό μπορεί να γενικευτεί στο πολλαπλασιασμό δύο αριθμών έστω n-bit ο ένας και m-bit ο άλλος, το τελικό αποτέλεσμα για να αποτυπωθεί σωστά θα χρειάζεται n+m-bit. Ένα άλλο συμπέρασμα που προκύπτει είναι ότι η πράξη του πολλαπλασιασμού ουσιαστικά ανάγεται στη πράξη n αριθμών μεγέθους m-bit (n είναι το μέγεθος του πολλαπλασιαστή και m το μέγεθος του πολλαπλασιαστέου). Οι πολλαπλασιαστές μπορούν να διακριθούν σε διάφορες κατηγορίες ανάλογα:

- Με το είδος των αριθμών που πολλαπλασιάζονται (απρόσημου - προσημασμένοι).
- Την κωδικοποίηση των αριθμών (Booth, Modified Booth,Sign Digit κτλ.).
- Τον τρόπο πρόσθεσής τους (παράλληλα, σειριακά, σειριακά-παράλληλα).
- Τον ρυθμό λειτουργίας τους (pipeline, συστολικοί).

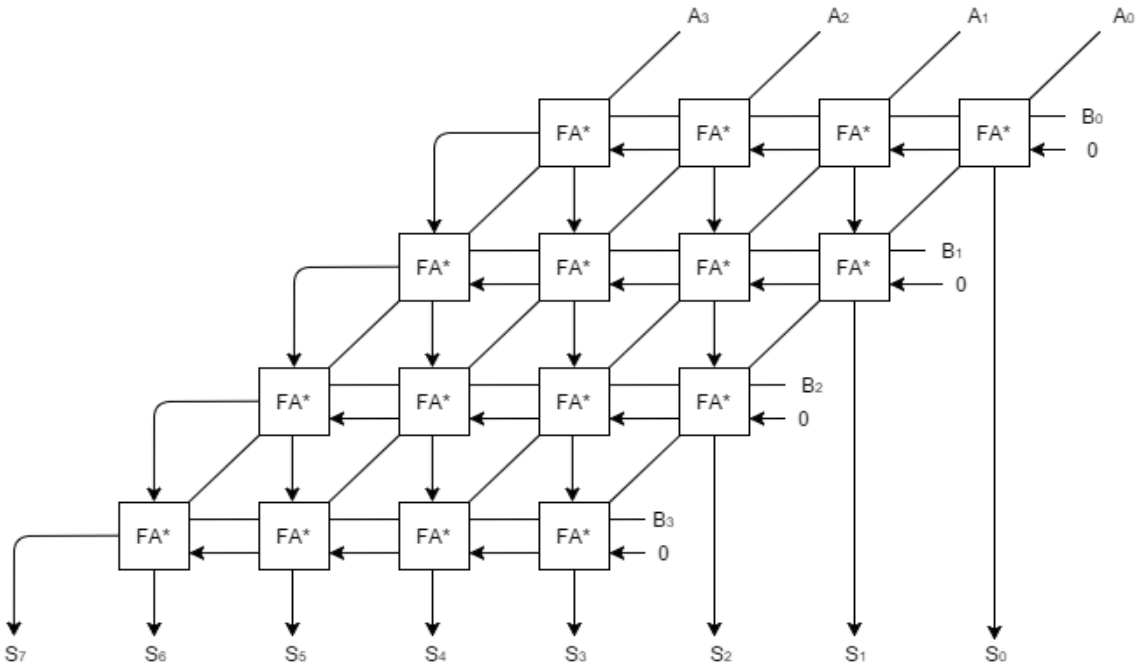
2.6.1 Παράλληλος πολλαπλασιαστής με αθροιστές διάδοσης κρατουμένου.

Για την υλοποίηση ενός πολλαπλασιαστή τέτοιου είδους χρησιμοποιείται ripple-carry αθροιστές για την άθροιση των μερικών γινομένων. Η Βασική μονάδα ενός τέτοιου πολλαπλασιαστή φαίνεται στο ακόλουθο σχήμα.



Σχήμα 2.9 Η βασική μονάδα πολλαπλασιαστή με διάδοση κρατουμένου.

Για την σχεδίαση του τελικού πολλαπλασιαστή θα πρέπει να γίνει διάταξη της βασικής μονάδας του πολλαπλασιαστή που παρουσιάστηκε όπως στον πίνακα 2.5 ώστε να προκύψει το ακόλουθο ψηφιακό κύκλωμα που εκτελεί τον πολλαπλασιασμό δύο αριθμών μεγέθους τεσσάρων bit.



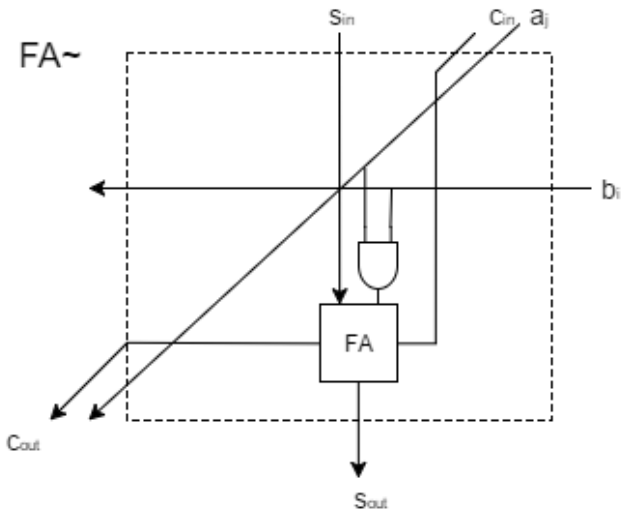
Σχήμα 2.10 Παράλληλος πολλαπλασιαστής διάδοσης κρατουμένου.

Επισημαίνεται ότι η επιπλέον πύλη AND προκειμένου να υπολογιστεί η μία είσοδος του πλήρη αθροιστή δεν καθυστερεί επιπλέον το κύκλωμα αφού οι είσοδοι την πύλης αυτής εισέρχονται ταυτόχρονα στην αρχή λειτουργίας του κυκλώματος, άρα καθυστερούν μόνο μία φορά το επάνω δεξιά κελί. Το κρίσιμο μονοπάτι αυτού του πολλαπλασιαστή m + (n-1) + (n-1) κύτταρα πλήρη αθροιστή. Παρατηρείται ότι η καθυστέρηση ενός τέτοιου είδους πολλαπλασιαστή είναι αρκετά μεγάλη, επιπλέον η πρώτη βαθμίδα του κυκλώματος μπορεί να απλοποιηθεί αρκετά εύκολα αφού υλοποιεί την πρόσθεση ενός αριθμού με το μηδέν. Ένας παράλληλος πολλαπλασιαστής τέτοιου είδους δεν χρησιμοποιείται ιδιαίτερα λόγω μεγάλης καθυστέρησης και μεγάλης επιφάνειας του, παρόλα αυτά η συμμετρική μορφή του κάνει την υλοποίηση της λειτουργίας συνεχούς διοχέτευσης του αρκετά απλή.

Στο σχήμα 2.9 παρουσιάζεται το κύκλωμα του πολλαπλασιασμού μη προσημασμένων αριθμών, αν οι αριθμοί που πρόκειται να πολλαπλασιαστούν είναι προσημασμένοι και αναπαρίστανται στη μορφή πρόσημου-μέτρου τότε μπορεί να χρησιμοποιηθεί το ίδιο κύκλωμα για τον πολλαπλασιασμό των μέτρων τους και το πρόσημο του αριθμού να προκύψει μέσω μιας πύλης XOR των πρόσημων εισόδων. Αν οι αριθμοί αναπαρίστανται στη μορφή συμπληρώματος ως προς δύο τότε πρέπει να λάβουμε υπόψιν μας την αρνητική αξία ορισμένων ψηφίων και να προσαρμόσουμε κατάλληλα το κύκλωμα μας, στην παράγραφο 2.3.2 περιεγράφηκε η μεθοδολογία πολλαπλασιασμού προσημασμένων αριθμών στη μορφή συμπληρώματος ως προς δύο, και στο σχήμα 2.3 περιγράφεται ο αθροιστής διάδοσης κρατουμένου ο οποίος δέχεται σαν είσοδο δύο αριθμούς στη μορφή συμπληρώματος ως προς δύο και αποτυπώνει το αποτέλεσμα σε n+1 bit, αυτό είναι ιδιαίτερα χρήσιμο ώστε να μην χρειαστεί η εφαρμογή προέκτασης προσήμου σε κάθε μερικό γινόμενο και να επιτευχθεί εξοικονόμηση χώρου και μείωση στην κατανάλωση ενέργειας του κυκλώματος.

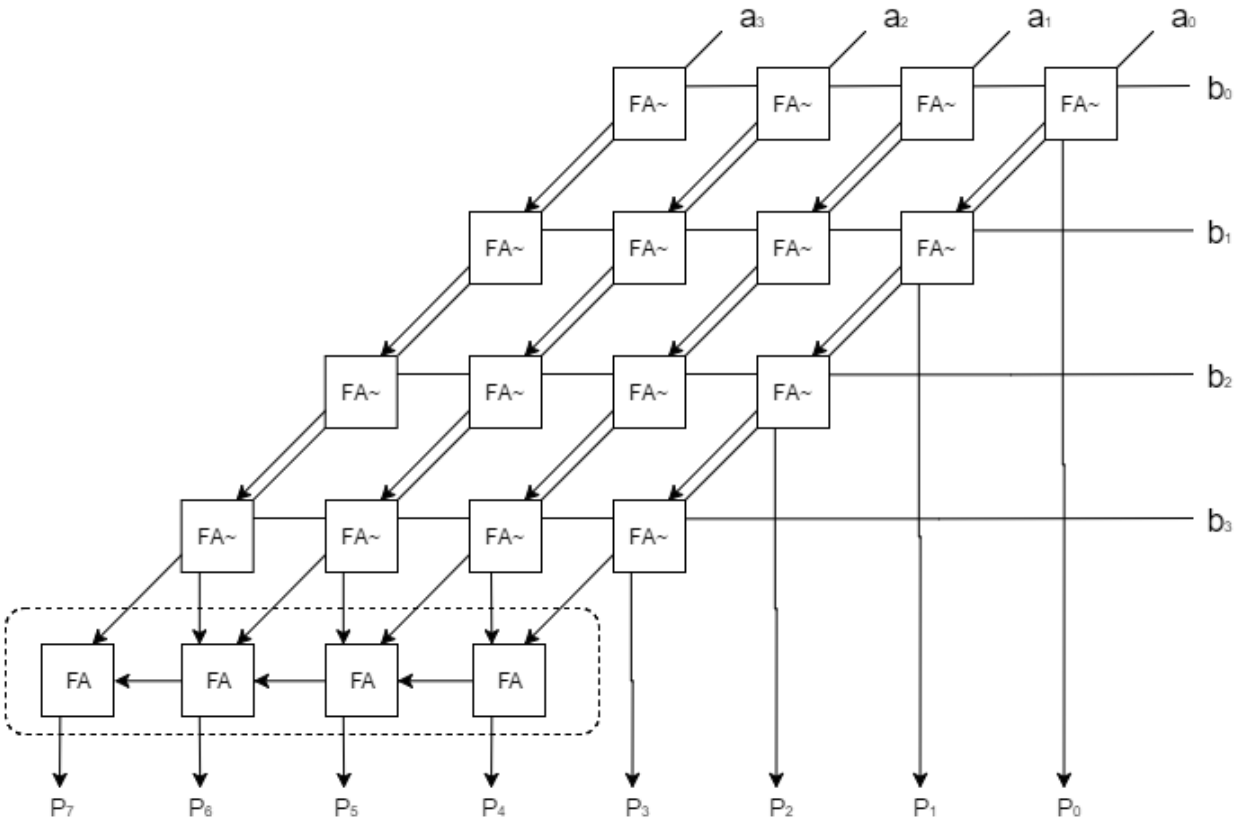
2.6.2 Παράλληλος πολλαπλασιαστής με αθροιστή αποθήκευσης κρατουμένου

Σε αυτή την υλοποίηση τα κρατούμενα δεν διαδίδονται στο επόμενο κύτταρο του ίδιου επιπέδου, αλλά οδηγούνται στο επόμενο επίπεδο, ενώ το βασικό κύτταρο δεν αλλάζει την κύρια λειτουργία του αλλά αλλάζει τη διάταξη των γραμμών, και παρουσιάζεται στο επόμενο σχήμα:



Σχήμα 2.11 Δομική μονάδα πολλαπλασιαστή σώσιμο κρατουμένου

Το συνολικό δίκτυο του πολλαπλασιαστή φαίνεται στο επόμενο σχήμα:

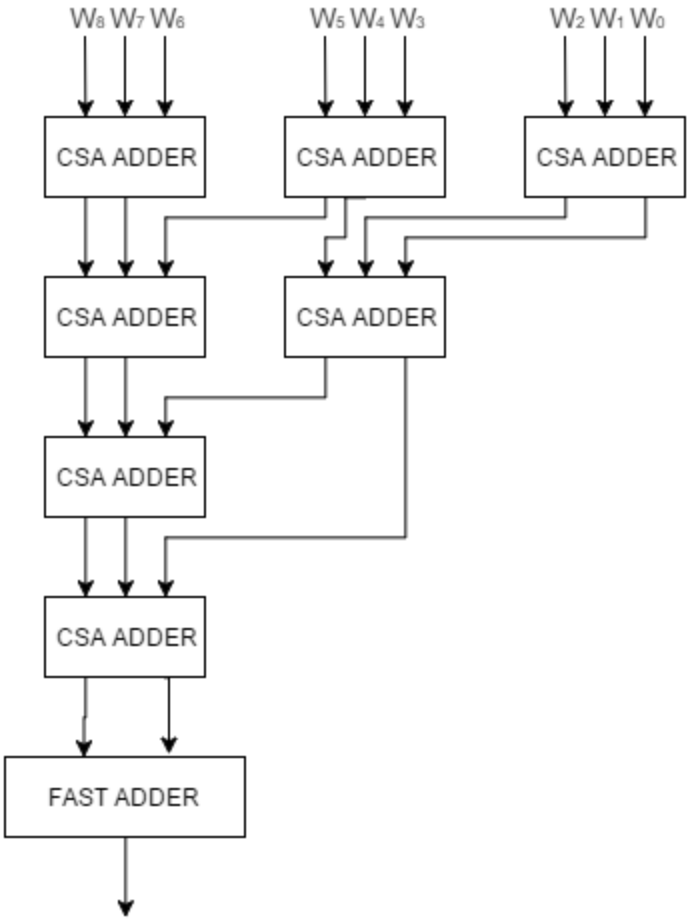


Σχήμα 2.12 Παράλληλος πολλαπλασιαστής με σώσιμο κρατουμένου δύο αριθμών μεγέθους 4bit

Παρατηρείται ότι μετά τον τελευταίο carry-save αθροιστή είναι απαραίτητη μία μονάδα αθροιστή διάδοσης κρατουμένου προκειμένου το τελικό αποτέλεσμα να αναπαρασταθεί σωστά, αυτή η μονάδα αθροιστή δεν είναι απαραίτητη για όλα τα bit εξόδου παρά μόνο για τα τελευταία τέσσερα (στη περίπτωση του σχήματος), γιατί τα αρχικά bit παράγονται απευθείας από τις μονάδες πλήρων αθροιστών στην αρχή κάθε λογικού επιπέδου. Ο χρόνος καθυστέρησης του πολλαπλασιαστή με σώσιμο κρατούμενου, είναι άμεσα εξαρτώμενος από τον τύπο του αθροιστή της τελευταίας βαθμίδας. Σε περίπτωση που χρησιμοποιήσουμε αθροιστή διάδοσης κρατουμένου η καθυστέρηση δίνεται από την σχέση: $T = n * T_{FA} + m * T_{FA} = (n + m) * T_{FA}$ όπου n και m τα μήκη των δύο αριθμών προς πολλαπλασιασμό. Αν είναι επιθυμητή η αύξηση της ταχύτητα πολλαπλασιασμού στην περίπτωση χρήσης αριθμών μεγάλου μήκους, τότε αντί ενός αθροιστή διάδοσης κρατουμένου να χρησιμοποιείται ένας αθροιστής πρόβλεψης κρατουμένου, ο οποίος περιεγράφηκε σε προηγούμενη παράγραφο. Από τα παραπάνω μπορούμε να συμπεράνουμε ότι ο πολλαπλασιασμός με δίκτυο αθροιστών αθροίσματος-κρατούμενου είναι σαφώς καλύτερος από τον πολλαπλασιασμό με δίκτυο αθροιστών διάδοσης κρατουμένου, η προσαρμογή του κυκλώματος ώστε να εκτελεί πολλαπλασιασμό προσημασμένων μπορεί να γίνει με τον ίδιο τρόπο που περιεγράφηκε στη προηγούμενη παράγραφο για αριθμούς στη μορφή προσήμου-μέτρου και συμπληρώματος ως προς δύο.

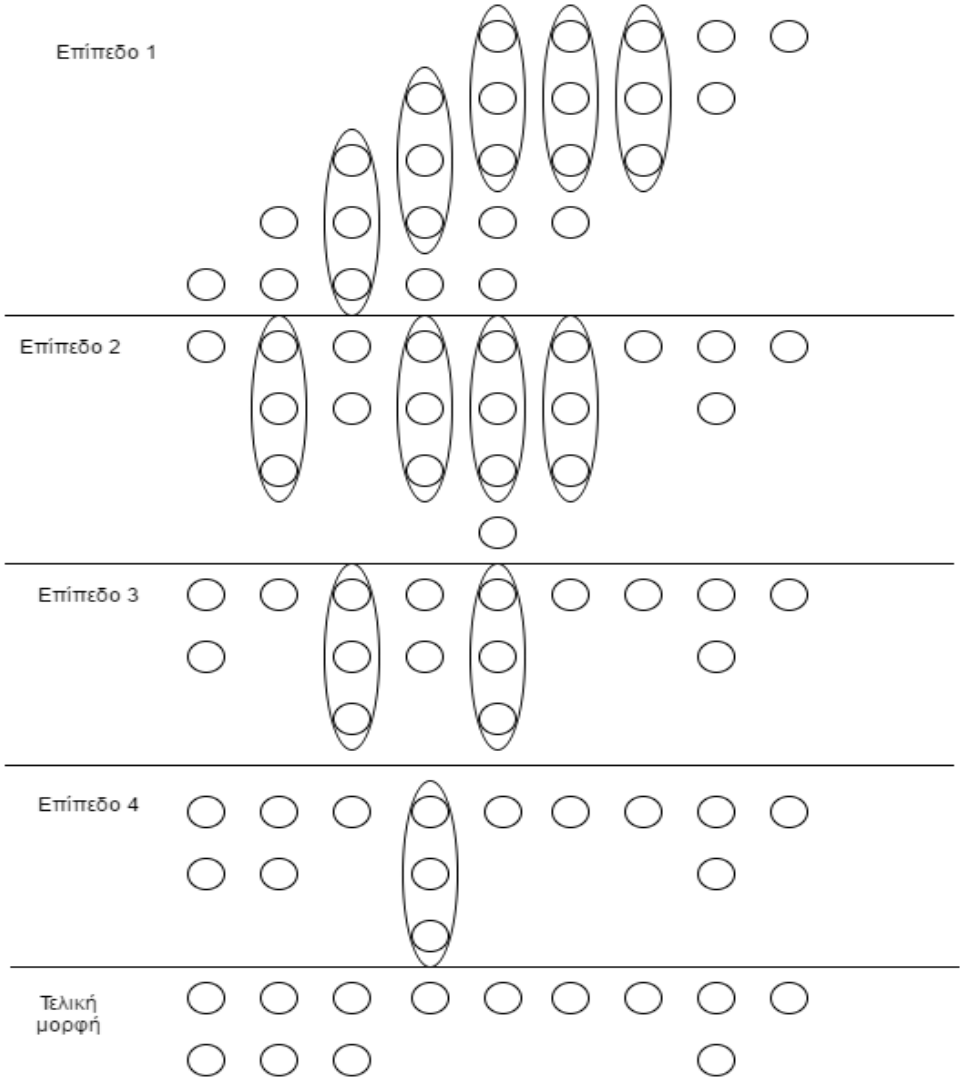
2.6.3 Παράλληλος πολλαπλασιαστής με δέντρο Wallace

Το προηγούμενο κύκλωμα πολλαπλασιαστή χρησιμοποίησε τη συμπιεστική δυνατότητα των carry-save αθροιστών προκειμένου να βελτιώσει τη ταχύτητα πολλαπλασιασμού, μπορεί εύκολα να παρατηρηθεί ότι αντί να συμπιέζονται τα μερικά γινόμενα ένα-ένα μπορούν εύκολα να ομαδοποιηθούν σε ομάδες των τριών ώστε να συμπιέζονται παράλληλα σε ομάδες, μετά την συμπίεση της πρώτης ομαδοποίησης όπου τα τρία διανύσματα μετατρέπονται σε δύο ισοδύναμα γίνεται νέα ομαδοποίηση και επαναλαμβάνεται η διαδικασία μέχρι να δημιουργηθούν δύο μερικά γινόμενα τα οποία πρέπει να αθροιστούν προκειμένου να παραχθεί το τελικό αποτέλεσμα, για παράδειγμα αν ένας πολλαπλασιασμός ο οποίος παράγει εννιά μερικά γινόμενα τότε το δέντρο Wallace που πρέπει να σχηματιστεί θα έχει την ακόλουθη μορφή:



Σχήμα 2.13 Πολλαπλασιαστής με δέντρο Wallace

Για τον υπολογισμό των επιπέδων ενός δέντρου Wallace αρκεί να διαιρεθεί ο αριθμός των μερικών γινομένων με το δύο, το πηλίκο πολλαπλασιασμένο με το δύο συν το υπόλοιπο της διαίρεσης αποτελεί τα μερικά γινόμενα που θα έχουν σχηματιστεί μετά από ένα επίπεδο συμπίεσης, επαναλαμβάνεται η διαδικασία μέχρι να σχηματιστούν μόνο δύο μερικά γινόμενα, τα οποία θα εισαχθούν στον τελικό αθροιστή. Ο τελικός αθροιστής πρέπει να προσθέσει δύο αριθμούς μεγέθους διπλάσιου από το μήκος των αριθμών οι οποίοι πολλαπλασιάζονται, έτσι χρειάζεται ένας μεγαλύτερος σε όγκο αθροιστής σε σχέση με αυτόν του πολλαπλασιαστή με χρήση αθροιστών με σώσιμο κρατουμένου που αναλύθηκε στη προηγούμενη παράγραφο, επιπλέον η πολυπλοκότητα των διασυνδέσεων είναι αρκετά μεγαλύτερη από του προηγούμενο πολλαπλασιαστή αφού δεν έχουν κανονικότητα, η ακολουθία συμπίεσης που χρησιμοποιεί το δέντρο Wallace μπορεί να γίνει κατανοητή από το ακόλουθο σχήμα:



Σχήμα 2.14 Μεθοδολογία κατασκευής δέντρου Wallace για πέντε μερικά γινόμενα μήκους πέντε bit

Στο σχήμα οι άσπροι κύκλοι συμβολίζουν τα bit των μερικών γινομένων ενώ οι ελλείψεις που τα περιέχουν συμβολίζουν τους πλήρεις αθροιστές στους οποίους θα εισαχθούν ως είσοδοι, μετά από κάθε έλλειψη φαίνεται ότι οι τρείς άσπροι κύκλοι μετατρέπονται σε δύο, ο ένας στην ίδια στήλη με την έλλειψη, ενώ ο άλλος στην αμέσως αριστερή στήλη, αν τα μερικά γινόμενα είναι προσημασμένοι αριθμοί τότε πρέπει να εφαρμοστεί η τεχνική επέκτασης πρόσημου (sign extension) προκειμένου οι αριθμοί να έχουν την επιθυμητή αξία, και επίσης να προστεθούν οι κατάλληλες μονάδες προκειμένου να αναπαρασταθούν σωστά οι αντίθετοι αριθμοί.

2.7 Κωδικοποίηση Modified Booth

Από τα κυκλώματα των πολλαπλασιαστών που αναλύθηκαν στη προηγούμενη παράγραφο βγαίνει το συμπέρασμα ότι η πράξη του πολλαπλασιασμού ουσιαστικά ανάγεται στη πρόσθεση όλων των παραγμένων μερικών γινομένων, και η ταχύτητα του πολλαπλασιαστή εξαρτάται άμεσα από τον αριθμό των μερικών γινομένων. Ένας τρόπος για να μειωθεί ο αριθμός των μερικών γινομένων είναι να κωδικοποιηθούν οι είσοδοι κατάλληλα ώστε να έρθουν σε μορφή μικρότερου μήκους η οποία παράγει λιγότερα μερικά γινόμενα, ένας τέτοιος τρόπος κωδικοποίησης είναι και ο αλγόριθμος Modified Booth ο οποίος βασίζεται στην δυνατότητα αναπαράστασης αριθμών σε μορφή προσημασμένων ψηφίων. Η κωδικοποίηση γίνεται σύμφωνα με τις παραπάνω σχέσεις:

$$Y = \sum_{i=0}^{n-1} z_i 2^i = \sum_{j=0}^{\frac{n}{2}-1} z_{2j} 2^{2j} + z_{2j+1} 2^{2j+1} = \sum_{j=0}^{\frac{n}{2}-1} (z_{2j} + 2 * z_{2j+1}) 2^{2j} = \sum_{j=0}^{\frac{n}{2}-1} {B^{mb}}_j 4^j \quad (2.12)$$

Όπου B^{mb}_j είναι το ψηφίο της κωδικοποίησης Modified Booth το οποίο υπολογίζεται σύμφωνα με τη σχέση:

$${B^{mb}}_j = z_{2j} + 2 * z_{2j+1} = b_{2j-1} - b_{2j} + 2 * (b_{2j} - b_{2j+1}) = -2b_{2j+1} + b_{2j} + b_{2j-1} \quad (2.13)$$

Από την παραπάνω σχέση γίνεται αντιληπτό ότι η ομαδοποίηση των ψηφίων γίνεται ανά τρία bit. Στον επόμενο πίνακα φαίνεται η αντιστοίχιση των κωδικοποιημένων ομάδων σε στα αντίστοιχα ψηφία.

Πίνακας 2.6 *Κωδικοποίηση Modified Booth*

Ομαδοποιημένα ψηφία			**Ψηφίο κωδικοποίησης MB**
Y_{i+1}	Y_i	Y_{i-1}	B^{MB}
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

Επειδή η κωδικοποίηση αυτή χρησιμοποιείται στα ψηφιακά κυκλώματα δεν είναι δυνατών να αναπαρασταθούν τα κωδικοποιημένα ψηφία Modified Booth από ένα διάνυσμα, έτσι είναι χρησιμοποιούνται τρία διανύσματα: ένα για το πρόσημο, ένα για τη μονάδα, και ένα για το δύο, προφανώς τα διανύσματα για το ένα και για το δύο, δεν μπορούν να έχουν μονάδα σε ψηφίο ίδιου

βάρους. Για να γίνει σωστά η κωδικοποίηση Modified Booth είναι αναγκαίο στην πρώτη ομάδα προς κωδικοποίηση να συμπεριληφθούν μόνο τα δύο λιγότερα σημαντικά bit και να θεωρηθεί ένα εικονικό μηδενικό στην αρχή του αριθμού το οποίο ομαδοποιείται με τα άλλα δύο. Η επόμενη ομάδα προς ομαδοποίηση είναι το πιο σημαντικό bit της προηγούμενης ομάδας μαζί με τα δύο αριστερά του. Αυτή η διαδικασία επαναλαμβάνεται μέχρι να ομαδοποιηθεί και το MSB του αριθμού, αν το MSB δεν μπορεί να ομαδοποιηθεί σύμφωνα με τον παραπάνω διαδικασία τότε εφαρμόζεται επέκταση πρόσημου ώστε να σχηματιστεί η τελευταία ομάδα, αν ο κωδικοποιημένος αριθμός είναι προσημασμένος, διαφορετικά συμπληρώνονται μηδενικά στην αρχή του αριθμού.

Παρουσιάζεται ένα παράδειγμα κωδικοποίησης για να γίνει καλύτερα αντιληπτή η κωδικοποίηση, έστω ότι θέλουμε να κωδικοποιήσουμε τον αριθμό B = 10011001 οι ομάδες οι οποίες σχηματίζονται είναι οι ακόλουθες:

$$010\ \textit{αντιστοιχεί σε}\ 1$$

$$100\ \textit{αντιστοιχεί σε}\ -2$$

$$011\ \textit{αντιστοιχεί σε}\ 2$$

$$100\ \textit{αντιστοιχεί σε}\ -2$$

Ο νέος κωδικοποιημένος αριθμός που παράγεται είναι λοιπόν ο $B^{MB} = \bar{2}2\bar{2}1$ είναι δυνατόν να ελέγξουμε την ορθότητα της κωδικοποίησης εφαρμόζοντας τον τύπο (2.12) ώστε να επαληθεύονται τα αποτελέσματα, ο αριθμός B αντιστοιχεί στον αριθμό -103 στην αναπαράσταση συμπληρώματος ως προς δύο. Με την εφαρμογή του τύπου (2.12) προκύπτει: $B^{MB} = -2 * 4^3 + + 2 * 4^2 - 2 * 4 + 1 = -103.$

2.7.1 Εφαρμογή κωδικοποίησης Modified Booth για πολλαπλασιασμό.

Όπως αναφέρθηκε και στη προηγούμενη παράγραφο ο λόγος που μελετάμε την κωδικοποίηση Modified Booth είναι λόγω της δυνατότητάς της να μειώνει τα μερικά γινόμενα, που πρέπει να προστεθούν για τη παραγωγή του αποτελέσματος του πολλαπλασιασμού. Παρατηρήθηκε ότι μια κωδικοποίηση Modified Booth ενός αριθμού μειώνει τα ψηφία που χρειάζονται για την αναπαράστασή του στο μισό, αυτό σημαίνει ότι με την κωδικοποίηση Modified Booth μειώνονται τα μερικά γινόμενα επίσης στο μισό, όμως τα μερικά γινόμενα παράγονται με λίγο περισσότερο περίπλοκο τρόπο. Στο πολλαπλασιασμό επιλέγεται για κωδικοποίηση ένας από τους δύο αριθμούς, ο πολλαπλασιαστής, έτσι κάθε κωδικοποιημένο bit μπορεί να γίνει αντιληπτό ως μία εντολή, έστω δηλαδή ότι έχουμε να πολλαπλασιάσουμε δύο αριθμούς τον A και τον B και επιλέγουμε ως πολλαπλασιαστή και αριθμό προς κωδικοποίηση τον B τότε κάθε ψηφίο του κωδικοποιημένου αριθμού B είναι μία από τις παρακάτω λειτουργίες σύμφωνα με τον ακόλουθο πίνακα.

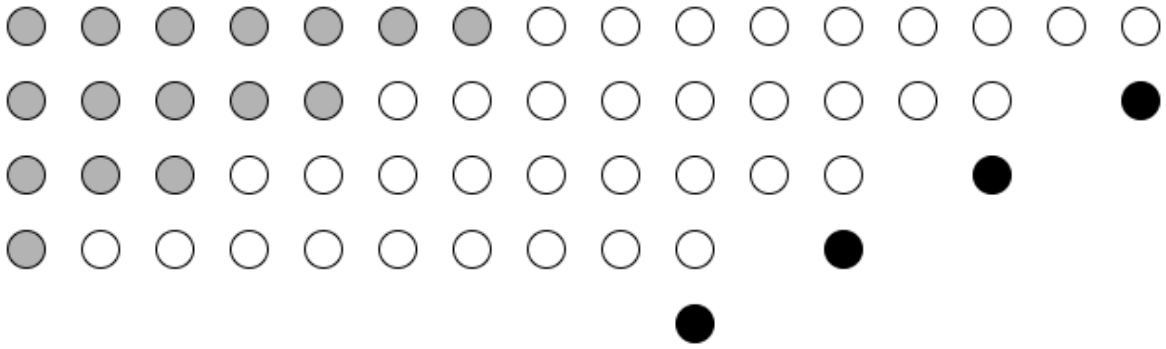
Πίνακας 2.7 *Λειτουργίες κωδικοποιημένων ψηφίων.*

Κωδικοποιημένο ψηφίο	**Λειτουργία**
0	Πρόσθεσε το 0
+1	Πρόσθεσε το A
+2	Πρόσθεσε το 2A
-2	Αφαίρεσε το 2A
-1	Αφαίρεσε το A

Σημειώνεται ότι κάθε επόμενο μερικό γινόμενο που παράγεται με την κωδικοποίηση Modified Booth έχει διαφορά δύο θέσεις από το προηγούμενο λόγω του τύπου (2.12). Οι αριθμοί A και 2A καθώς και οι αντίστοιχοι αρνητικοί είναι εύκολο να παραχθούν αφού ο πολλαπλασιασμός με το δύο στο δυαδικό σύστημα ουσιαστικά είναι η μετατόπιση του αριθμού μία θέση αριστερά, αυτός είναι και ο λόγος που όπως θα φανεί και στη συνέχεια τα μερικά γινόμενα ενός πολλαπλασιαστή που χρησιμοποιεί Modified Booth κωδικοποίηση έχουν μέγεθος n+1 (όπου n είναι το μέγεθος των εισόδων του πολλαπλασιαστή) λόγω της μετατόπισης. Τα αρνητικά μερικά γινόμενα μπορούν να παραχθούν με αντιστροφή των ψηφίων και προσθήκη μίας μονάδας κατά τα γνωστά, η πρόσθεση όλων των απαραίτητων μονάδων γίνεται με ένα επιπλέον διορθωτικό όρο, ο σχηματισμός του οποίου θα αναλυθεί στην επόμενη παράγραφο, και βοηθά επίσης στο να μην εφαρμόζεται επέκταση προσήμου σε όλα τα μερικά γινόμενα κάτι το οποίο σπαταλά ενέργεια και επιφάνεια.

2.7.2 Δημιουργία διορθωτικού όρου

Τα μερικά γινόμενα μίας MB κωδικοποίησης, σύμφωνα με όσα έχουν αναλυθεί μέχρι στιγμής έχουν την ακόλουθη μορφή:



Σχήμα 2.15 Μερικά γινόμενα MB κωδικοποίησης για 8x8 πολλαπλασιασμό.

Στο παραπάνω σχήμα οι άσπρες κουκίδες συμβολίζουν τα ψηφία των μερικών γινομένων, οι μαύρες τους απαραίτητους άσους που πρέπει να προστεθούν όπου υπάρχει αντιστροφή,

προκειμένου να προκύψει σωστή αναπαράσταση αντίθετων αριθμών, οι γκρι κουκκίδες συμβολίζουν την επέκταση του πρόσημου που πρέπει να εφαρμοστεί σε όλα τα μερικά γινόμενα ώστε να έχουν την σωστή αξία, η επέκταση προσήμου είναι μία μη αποδοτική χρήση του υλικού, αφού στην ουσία προστίθενται άσοι και μηδενικά τα οποία θα μπορούσαν να είχαν προϋπολογιστεί, αυτή τη λειτουργία κάνει το διορθωτικό μερικό γινόμενο ct το οποίο σχεδιάζεται με την ακόλουθη μαθηματική λογική.

Στο σχήμα 2.14 φαίνεται ότι στα μερικά γινόμενα εφαρμόζεται επέκταση προσήμου από το MSB μέχρι το βάρος 15, δηλαδή το ct γίνεται

$$ct = \sum_{\kappa=0}^{3} s_k * \sum_{i=\,8+2k}^{15} 2^i$$

Στη γενική περίπτωση η παραπάνω σχέση παίρνει τη μορφή:

$$\mathrm{ct} = \sum_{k=0}^{\frac{N}{2}} s_k * \sum_{i=N+2k}^{2N} 2^i$$

Όπου: N το μήκος της λέξης.

Σύμφωνα με την ιδιότητα:

$$\sum_{l=j}^{k} 2^q = 2^{k+1} - 2^j$$

Η γενική σχέση γίνεται:

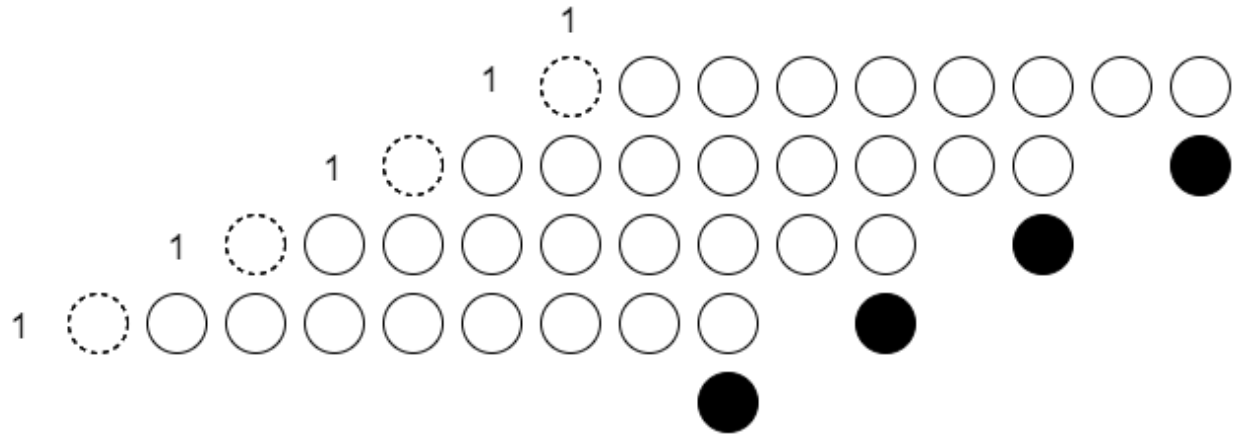
$$\mathrm{ct} = \sum_{k=0}^{N/2} s_k * (2^{2N} - 2^{N+2k}) = \sum_{0}^{N/2} -s_k 2^{N+2k}$$

και χρησιμοποιώντας την ιδιότητα: $-s_k = \overline{s_k} - 1$ Η σχέση παίρνει τη μορφή:

$$ct = \sum_{k=0}^{N/2} \overline{S_k} * 2^{N+2k} + \{2^{2N-1} + \cdots + 2^{N+1} + 2^N\} \qquad (2.14)$$

Από την εφαρμογή αυτού του τύπου συμπεραίνεται ότι για την δημιουργία του μερικού γινομένου αρκεί να προστεθεί το συμπλήρωμα του αντίστοιχου bit προσήμου σε κάθε μερικό γινόμενο και μία μονάδα στο βάρος αμέσως αριστερά από το ανεστραμμένο bit, επιπλέον τοποθετείται μία

μονάδα στο βάρος της συμπλήρωσης του πρώτου μερικού γινομένου: όλα τα παραπάνω παρουσιάζονται στο επόμενο σχήμα όπου παρουσιάζονται τα μερικά γινόμενα με τον επιπλέον διορθωτικό όρο:



Σχήμα 2.16 Μερικά γινόμενα μαζί με τον διορθωτικό όρο

Στο παραπάνω σχήμα τα οι κύκλοι με διακεκομμένη περιφέρεια συμβολίζουν το αντεστραμμένο bit του μερικού γινομένου σύμφωνα με το τύπο (2.14).

2.7.3 Κυκλωματική υλοποίηση Modified Booth κωδικοποίησης.

Σε αυτή τη παράγραφο αναλύεται ο τρόπος με τον οποίο υλοποιείται σε επίπεδο ψηφιακού κυκλώματος η MB κωδικοποίηση. Όπως αναφέρθηκε και σε προηγούμενη παράγραφο κάθε κωδικοποιημένο ψηφίο του MB αναπαρίστανται από τρία ξεχωριστά σήματα, το one που δείχνει πότε το κωδικοποιημένο ψηφίο είναι μονάδα, το two που δείχνει πότε το ψηφίο είναι δύο και το sign που δείχνει το πρόσημο του ψηφίου, όταν και το one και το two είναι μηδέν στο ίδιο βάρος αυτό σημαίνει ότι το κωδικοποιημένο ψηφίο είναι μηδέν, τα παραπάνω παρουσιάζονται στον ακόλουθο πίνακα:

Πίνακας 2.8 *Πίνακας κωδικοποίησης Modified Booth*

Ομάδα bit προς κωδικοποίηση			**Ψηφίο Modified Booth**	**Ψηφία αναπαράστασης κωδικοποίησης**		
Y_{2i+1}	Y_{2i}	Y_{2i-1}		**Sign**	**One**	**Two**
0	0	0	0	0	0	0
0	0	1	1	0	1	0
0	1	0	1	0	1	0
0	1	1	2	0	0	1
1	0	0	-2	1	0	1
1	0	1	-1	1	1	0
1	1	0	-1	1	1	0
1	1	1	-0	1	0	0

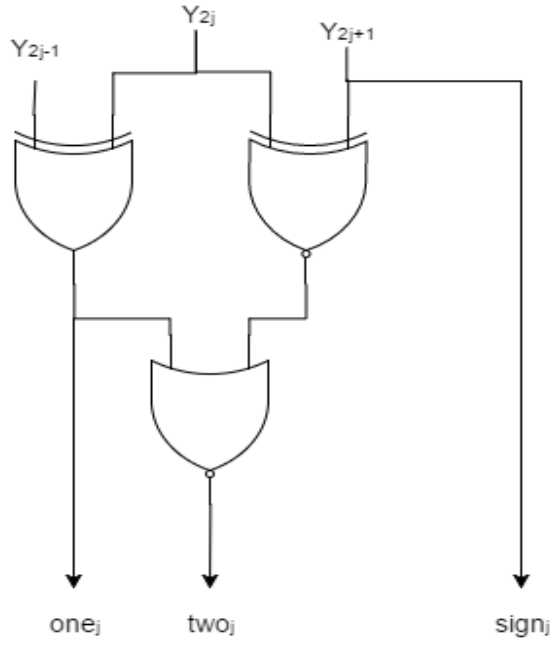
Από τον πίνακα κωδικοποίησης γίνονται αντιληπτές οι ακόλουθες λογικές σχέσεις για τα sign, one και two:

$$sign_j = Y_{i+1}$$

$$one_j = Y_{2i-1} \oplus Y_{2j}$$

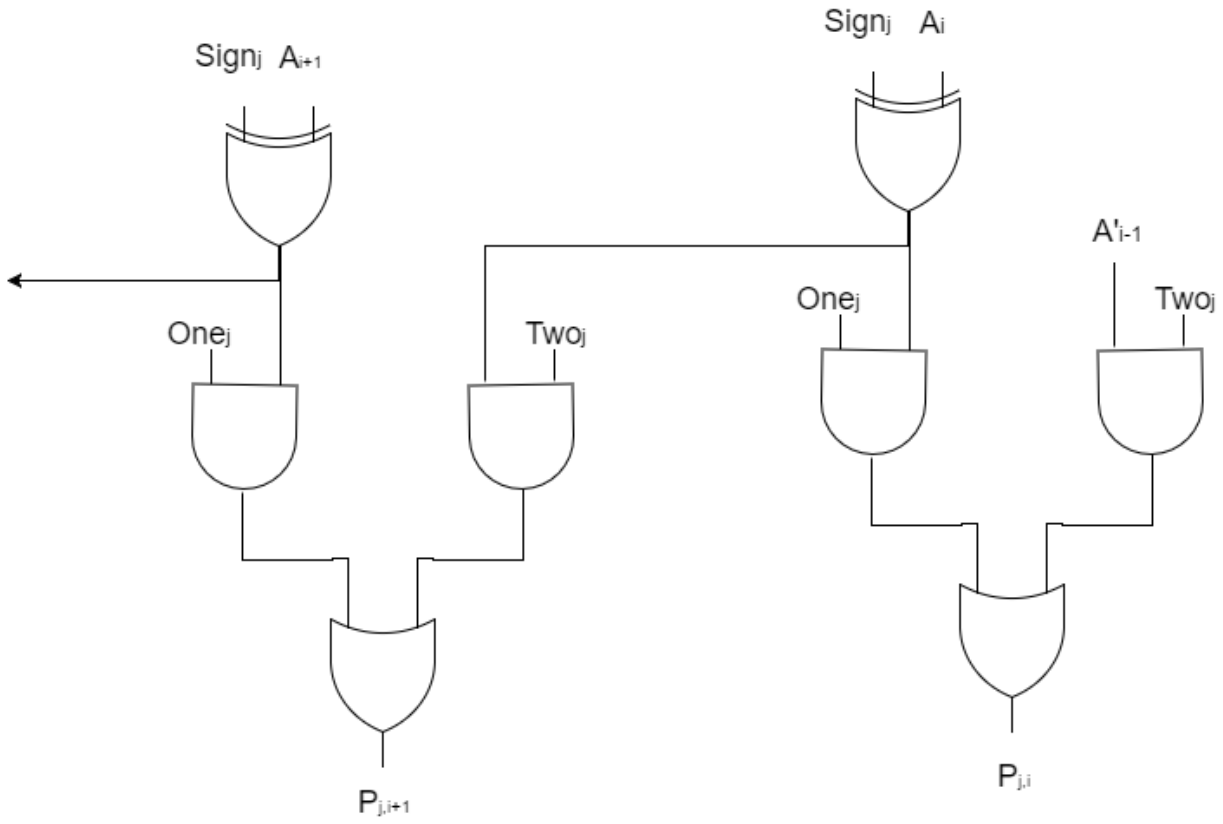
$$two_j = \left(Y_{2i+1} \oplus Y_{2j}\right) * \overline{one_j}$$

Σε επίπεδο λογικών πυλών οι παραπάνω σχέσεις διαμορφώνονται ως εξής:



Σχήμα 2.17 Κωδικοποίηση MB με λογικές πύλες

Έχοντας υλοποιήσει τα ψηφία του MB σε λογικό επίπεδο αρκεί να σχεδιαστεί το κύκλωμα για την παραγωγή των μερικών γινόμενα, αυτό γίνεται ακολουθώντας τις εντολές του πίνακα 2.6 δηλαδή αν το ψηφίο one είναι ένα τότε το μερικά γινόμενο είναι ίδιο με το πολλαπλασιαστή (έστω A), όταν το ψηφίο two είναι μονάδα τότε το μερικό γινόμενο είναι ίδιο με το μετατοπισμένο πολλαπλασιαστή A κατά μία θέση αριστερά, αν το ψηφίο sign είναι μονάδα τότε επαναλαμβάνεται η ίδια διαδικασία μόνο που σε αυτή τη περίπτωση αντιστρέφονται να ψηφία του μερικού γινομένου, πρέπει να δοθεί προσοχή στο πιο σημαντικό bit κάθε μερικού γινομένου καθώς θα πρέπει να αντιστραφεί προκειμένου να δημιουργηθεί ο σωστός διορθωτικός όρος, τα παραπάνω αποτυπώνονται στην ακόλουθη διάταξη:



Σχήμα 2.18 Σχηματισμός των μερικών γινομένων σε επίπεδο πυλών.

Στο σχήμα, φαίνεται ότι ο αν το two είναι μονάδα τότε γίνεται η μετατόπιση προς τα αριστερά όπως δείχνει και το βέλος, στο σχηματισμό του τελευταίου bit του μερικού γινομένου $P_{j,n}$ θα χρησιμοποιηθεί πύλη NOR αντί για OR λόγω της αντιστροφής που πρέπει να γίνει, επίσης στο σχήμα φαίνεται μια πολύ χρήσιμη ιδιότητα της πύλης XOR, η οποία θα χρησιμοποιηθεί σε μεγάλο βαθμό στην επόμενη ενότητα, η οποία μπορεί να αντιστρέφει ένα bit εισόδου αν η άλλη είσοδος είναι μονάδα ή να το περνά όπως είναι αν η άλλη είσοδος είναι μηδέν, αυτό φαίνεται και από το πίνακα αληθείας της.

Για την τοποθέτηση των σωστών κρατουμένων (μαύρων κύκλων στα σχήματα 2.15, 2.14) πρέπει να τοποθετηθεί η μονάδα μόνο αν το $sign_j$ και τουλάχιστον ένα από τα one_j και two_j είναι μονάδα, επίσης για την αναπαράσταση του διορθωτικού όρου τοποθετείται μία μονάδα στη θέση N και N+1 και μετέπειτα στις θέσεις N+3, N+5 έως ότου φτάσουμε στη θέση 2N-1 [2].

3 ΣΧΕΔΙΑΣΜΟΣ MULTIPLIER- ADDER/ ACCUMULATOR ΓΙΑ SIGN-MAGNITUDE ΑΡΙΘΜΗΤΙΚΗ

3.1 Στόχος εργασίας

Έως και το προηγούμενο κεφάλαιο έχει αναλυθεί πλήρως η κατασκευή των βασικών αριθμητικών μονάδων οι οποίες υλοποιούν πρόσθεση και πολλαπλασιασμό, σε αυτό το κεφάλαιο, το οποίο αποτελεί και επίκεντρο της παρούσας διπλωματικής εργασίας, θα γίνει συνδυασμός αυτών των κυκλωμάτων προκειμένου να κατασκευαστούν πιο σύνθετα κυκλώματα, όπως ένας multiplier-adder (MAD) και ένας multiplier-accumulator (MAC). Ο MAD είναι μία αριθμητική μονάδα η οποία εκτελεί πολλαπλασιασμό και πρόσθεση μεταξύ των εισόδων του, ενώ η λειτουργία του MAC είναι να υπολογίζει το γινόμενο δύο αριθμών, να το αποθηκεύει, και σε αυτό να προσθέτει το γινόμενο των επόμενων εισόδων, όταν ο MAC και ο MAD υπολογίζουν ένα δύο γινόμενα ονομάζονται double MAD και double MAC αντίστοιχα.

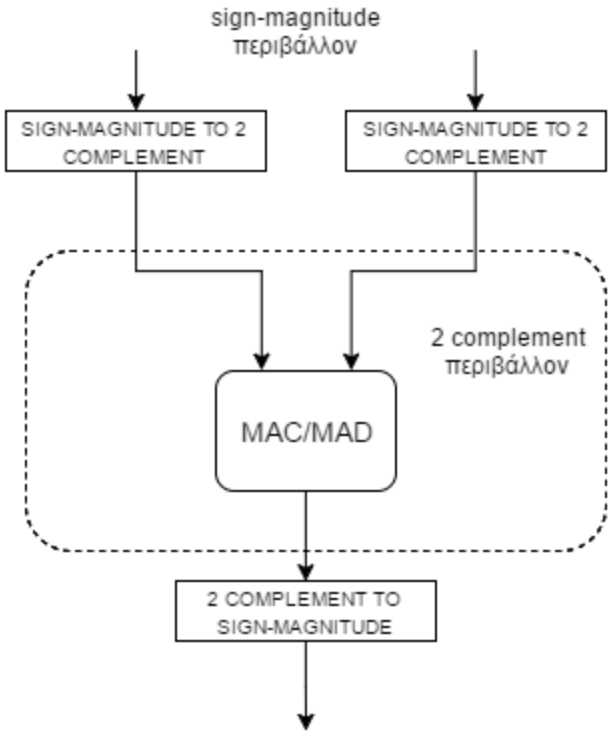
Στόχος της εργασίας είναι η αποδοτική σχεδίαση MAD και MAC οι οποίοι να δέχονται ως είσοδο και να παράγουν ως έξοδο αριθμούς στη μορφή πρόσημου-μέτρου, στο προηγούμενο κεφάλαιο βγήκε το συμπέρασμα ότι αυτός ο τρόπος αναπαράστασης είναι αρκετά δύσχρηστος όταν σχεδιάζονται ψηφιακά κυκλώματα και ιδιαίτερα όταν σε αυτά εκτελείται η αριθμητική πράξη της πρόσθεσης, παρόλα αυτά σε αρκετές εφαρμογές όπως παρουσιάστηκε και στην εισαγωγή της εργασίας ο σχεδιαστής είναι αναγκασμένος να χρησιμοποιήσει αυτή την αριθμητική. Σε αυτό το κεφάλαιο θα μελετηθούν δύο διαφορετικοί τρόποι επεξεργασίας sign-magnitude αριθμητικής προκειμένου να εφαρμοστούν σε έξι διαφορετικές τοπολογίες, ώστε να γίνει σύνθεση τους και να ληφθούν τα κατάλληλα συγκριτικά αποτελέσματα και συμπεράσματα ως προς το ποιο είδος σχεδίασης είναι αποδοτικότερο σε κάθε σχεδιασμό.

3.2 Μέθοδοι επεξεργασίας sign-magnitude αριθμητικής

Όταν ένας σχεδιαστής βρίσκεται αντιμέτωπος με το σχεδιασμό ψηφιακών αριθμητικών κυκλωμάτων τα οποία χρησιμοποιούν αριθμητική διαφορετική από αυτή του συμπληρώματος ως προς δύο υπάρχουν δύο τρόποι σχεδίασης που μπορεί να ακολουθήσει.

- Έμμεσος τρόπος σχεδίασης (indirect method).
- Άμεσος τρόπος σχεδίασης (direct method).

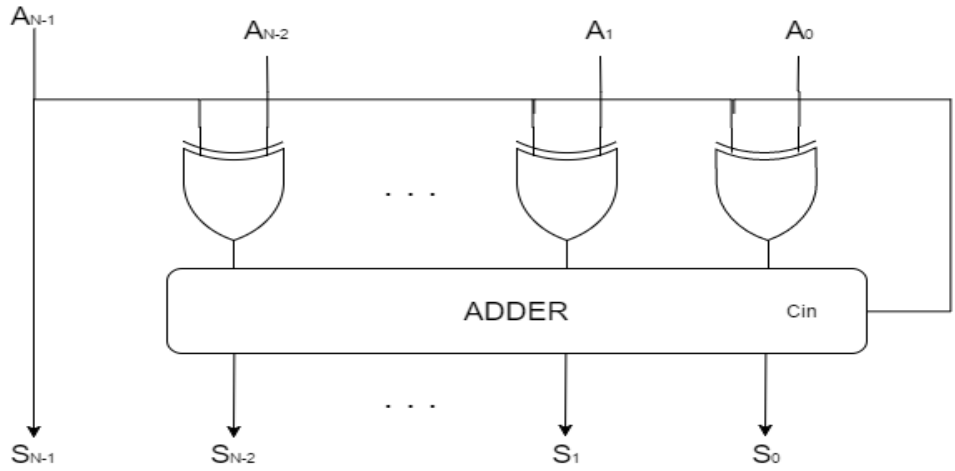
Εάν ακολουθηθεί ένας έμμεσος τρόπος σχεδίασης τότε μπορεί, αντί για ένα κύκλωμα το οποίο επεξεργάζεται αριθμούς στη μορφή πρόσημο-μέτρο ή σε οποιαδήποτε άλλη αριθμητική, μπορεί να σχεδιαστεί ένα το οποίο επεξεργάζεται αριθμούς στην μορφή συμπληρώματος ως προς δύο, η σχεδίαση του οποίου είναι περισσότερο απλή και διαδεδομένη, και στις εισόδους και εξόδους του να χρησιμοποιηθούν μονάδες αντιστροφής αριθμών από sign-magnitude σε 2 complement μορφή, αυτός ο τρόπος σχεδίασης είναι προφανώς δυνατόν να εφαρμοστεί σε οποιαδήποτε μονάδα η οποία επεξεργάζεται διαφορετικό είδος αριθμητικής από εκείνη του συμπληρώματος ως προς δύο. Μία έμμεση σχεδίαση κυκλώματος φαίνεται στο ακόλουθο σχήμα.



Σχήμα 3.1 Έμμεσος τρόπος σχεδίασης MAC/MAD για πρόσημο-μέτρο αριθμητική.

Όπως φαίνεται και στο σχήμα αντί να γίνεται επεξεργασία των αριθμών σε sign-magnitude, ουσιαστικά γίνεται επεξεργασία των αριθμών σε συμπλήρωμα ως προς δύο, αυτή η σχεδίαση αν και είναι αρκετά βολική για τον σχεδιαστή δεν είναι τόσο αποδοτική λόγω του ότι για να λειτουργήσει πρέπει να επιβαρυνθεί το συνολικό κύκλωμα με τις μονάδες μετατροπής που φαίνονται στο σχήμα και έτσι να αυξηθεί η επιφάνεια και η καθυστέρηση.

Σημειώνεται ότι οι μονάδες μετατροπής από πρόσημο-μέτρο σε συμπλήρωμα ως προς δύο και οι αντίστροφες (από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο) δεν έχουν διαφορά καθώς ακολουθείται ακριβώς η ίδια διαδικασία η οποία αναλύεται στη συνέχεια. Όταν ένας αριθμός είναι θετικός τότε η αναπαράσταση του σε πρόσημο μέτρο και συμπλήρωμα ως προς δύο ταυτίζονται, όταν όμως ο αριθμός είναι αρνητικός, τότε οι δύο αυτές αναπαραστάσεις έχουν διαφορετική μορφή. Ο αρνητικός αριθμός σε πρόσημο μέτρο είναι ίδιος με τον αντίστοιχο θετικό σε συμπλήρωμα ως προς δύο με μόνη διαφορά στο MSB, όπως αναλύθηκε και στο προηγούμενο κεφάλαιο για από έναν αρνητικό αριθμό σε συμπλήρωμα ως προς δύο για να υπολογιστεί ο αντίστοιχος θετικός (και το αντίστροφο) αρκεί να αντιστραφούν όλα τα ψηφία του και να προστεθεί μία μονάδα, αυτό το κύκλωμα παρουσιάζεται στο επόμενο σχήμα.



Σχήμα 3.2 Μονάδα μετατροπής μεταξύ πρόσημο-μέτρο και συμπλήρωμα ως προς δύο αναπαράσταση.

Από το σχήμα παρατηρείται ότι το πιο σημαντικό bit είναι το ίδιο και στις δύο αναπαραστάσεις, αυτό είναι λογικό αφού το MSB είναι υπεύθυνο για το πρόσημο του αριθμού ανεξάρτητα από τι αναπαράσταση έχει επιλεχτεί, επίσης και εδώ χρησιμοποιήθηκε η ιδιότητα της πύλης XOR να αντιστρέφει τη μία είσοδο εάν η άλλη είναι σταθερά ίση με τη μονάδα, έτσι όταν το περισσότερο σημαντικό bit είναι άσος τότε αντιστρέφεται όλος ο αριθμός και το κρατούμενο εισόδου του αθροιστή γίνεται 1, επίσης η επιφάνεια και η καθυστέρηση που προσθέτει μία τέτοια μονάδα στο κύκλωμα είναι άμεσα εξαρτώμενη από τον επιλεγμένο αθροιστή. Για CLA αθροιστές η μετατροπή θα γίνει σχετικά γρήγορα, αλλά η επιφάνεια του αθροιστή θα είναι μεγάλη, ενώ ένας αθροιστής διάδοσης κρατουμένου θα καθυστερήσει πολύ το κύκλωμα ειδικά αν επεξεργάζονται αριθμοί μεγάλου μήκους, αλλά η επιφάνεια του είναι μικρή. Επίσης είναι σημαντικό να παρατηρηθεί ότι στην περίπτωση θετικών αριθμών αυτό το κύκλωμα ουσιαστικά δεν κάνει τίποτα αφού προσθέτει στην είσοδο απλά το μηδέν.

Ο άμεσος τρόπος σχεδίασης τις περισσότερες φορές είναι πιο περίπλοκος από τον έμμεσο αλλά όταν εφαρμόζεται μπορεί να επιτύχει καλύτερα αποτελέσματα. Με τον άμεσο τρόπο σχεδίασης ουσιαστικά αποφεύγεται η χρήση μονάδων μετατροπής αριθμητικής τόσο στην είσοδο όσο και στην έξοδο, αυτό γίνεται με την προσαρμογή του κυκλώματος του MAC ώστε να επεξεργάζεται και να παράγει αριθμούς απευθείας σε sign-magnitude αριθμητική. Επισημαίνεται ότι στη γενική περίπτωση δεν υπάρχει καλύτερη και χειρότερη σχεδίαση από τις δύο, μπορεί στην απόπειρα προσαρμογής ενός κυκλώματος ώστε να δουλεύει με μία άμεση σχεδίαση να το επιβαρυνθεί τόσο πολύ ώστε να είναι προτιμότερο να εισάγουμε μονάδες μετατροπής στις εισόδους και τις εξόδους του οπότε η μελέτη και σύγκριση των δύο σχεδιασμών είναι απαραίτητη.

3.3 Υλοποιήσεις MAD και MAC

Για το σκοπό αυτής της διπλωματικής εργασίας, σχεδιάστηκαν έξι διαφορετικά κυκλώματα που υλοποιούν ξεχωριστές πράξεις, όμως όλα έχουν κάποια κοινά χαρακτηριστικά, τα οποία παρουσιάζονται στη συνέχεια:

- Κωδικοποίηση Modified Booth.
- Δεντρικός συμπιεστής Wallace
- Όλες οι αθροίσεις μεταξύ δύο αριθμών γίνονται με CLA αθροιστές

Το ψηφιακό κύκλωμα το οποίο εκτελεί πολλαπλασιασμό δύο (ή ακόμα και περισσότερων) αριθμών και προσθέτει το αποτέλεσμα του πολλαπλασιασμού με το περιεχόμενο ενός καταχωρητή, το περιεχόμενο του καταχωρητή καθορίζεται από μία είσοδο του κυκλώματος (MAD) ενώ σε άλλες το περιεχόμενο του καταχωρητή καθορίζεται από το αποτέλεσμα του προηγούμενου πολλαπλασιασμού (MAC). Συνεπώς λοιπόν τα κυκλώματα MAD και MAC είναι ικανά να εκτελούν τις ακόλουθες αριθμητικές πράξεις:

- $S = \sum(A_\iota * X_\iota)$ (MAC)
- $S = A * X + D$ (MAD)

Αυτές οι αριθμητικές πράξεις είναι πολύ συχνές στην ψηφιακή επεξεργασία σήματος, και σε άλλες εφαρμογές. Επιπλέον σχεδιάστηκαν MAD και MAC που εκτελούν τις ίδιες πράξεις όμως με τον υπολογισμό δύο γινομένων αντί ενός, έτσι σχεδιάστηκαν και κυκλώματα που υπολογίζουν τις αριθμητικές πράξεις:

- $S = \sum(A_i * X_i + B_i * Y_i)$(double MAC)
- $S = A * X + B * Y + D$ (double MAD)

Για την πρώτη και τρίτη περίπτωση θα γίνει σχεδίαση με και χωρίς λειτουργία συνεχούς διοχέτευσης, προκειμένου να υπολογιστεί το κέρδος που έχει η pipeline υλοποίηση. Όλες οι υλοποιήσεις παρουσιάζονται με δύο διαφορετικούς σχεδιασμούς, έναν άμεσο όπου προσαρμόζεται το κύκλωμα έτσι ώστε να μην χρειάζεται μετατροπέα από συμπλήρωμα ως προς

δύο σε μορφή προσήμου-μέτρου, και μία πιο έμμεση σχεδίαση όπου χρησιμοποιείται ένας τέτοιος μετατροπέας μόνο στην έξοδο του MAC.

3.4 Σχεδίαση αφαιρέτη υπολογισμού απόλυτης τιμής.

Το βασικότερο πρόβλημα που αντιμετωπίζεται στην λειτουργία MAC ή MAD που επεξεργάζεται αριθμούς στην μορφή προσήμου-μέτρου βρίσκεται στο πώς θα υπολογιστεί το τελικό αποτέλεσμα σε αυτή την αναπαράσταση αποδοτικά. Παρόλο που στο πρώτο κεφάλαιο εξηγήθηκε ότι η αναπαράσταση sign-magnitude είναι αποδοτικότερη στην περίπτωση του πολλαπλασιασμού από αυτή του συμπληρώματος ως προς δύο, λόγω του ότι μπορούν απλά να πολλαπλασιαστούν τα μέτρα των αριθμών και το τελικό πρόσημο του αποτελέσματος να υπολογιστεί μέσω μίας πύλης XOR των πρόσημων εισόδου, κάτι τέτοιο δεν ισχύει στη περίπτωση του MAC και MAD λόγω του ότι εκτός από την πράξη του πολλαπλασιασμού εκτελείται και η πράξη της πρόσθεσης.

Σε αυτή την παράγραφο αναλύεται η μετατροπή που έγινε στον τελικό αθροιστή, ο οποίος προσαρμόστηκε στο να παράγει πάντα θετικό αποτέλεσμα, ακόμα και όταν προσθέτει ετερόσημους αριθμούς με το μέτρο του θετικού μικρότερο από το μέτρο του αρνητικού. Οι αλλαγές που παρουσιάζονται στην συνέχεια της παραγράφου αφορούν έναν αθροιστή πρόβλεψης κρατουμένου όμοιος με εκείνον που μελετήθηκε στο σχήμα 2.5, ο οποίος δέχεται σαν είσοδο δύο αριθμούς ίδιου μήκους και ένα σήμα ελέγχου και εκτελεί είτε τη πράξη της πρόσθεσης είτε της αφαίρεσης ανάμεσα στους δύο αριθμούς ανάλογα με τη τιμή του σήματος ελέγχου.

Η βασική ιδέα πάνω στην οποία στηρίχτηκε η ακόλουθη προσαρμογή είναι στην δυνατότητα να παραχθεί το αποτέλεσμα της απόλυτης τιμής της αφαίρεσης δύο αριθμών ($|A-B|$) χωρίς να πραγματοποιηθούν δύο διαφορετικές αφαιρέσεις. Για να επιτευχθεί κάτι τέτοιο πρέπει να είναι δυνατή η αναπαράσταση της διαφοράς $B-A$ μέσω της διαφοράς $A-B$, υπενθυμίζεται ότι ο αντίθετος ενός αριθμού A είναι ο $-A$ οι οποίοι συνδέονται μέσω της σχέσης: $-A=\bar{A}+1$ όπου $\bar{A}$ είναι ο αριθμός ίδιος με τον A αλλά με ανεστραμμένα όλα τα ψηφία του, με άλλα λόγια υπολογίζεται ο αντίθετος ενός αριθμού με την αναπαράσταση συμπληρώματος ως προς δύο. Η λογική με την οποία συνδέονται οι δύο αφαιρέσεις είναι η εξής:

$$B-A \Leftrightarrow B-A+1-1 \Leftrightarrow -(A-B-1)-1 \Leftrightarrow$$

$$\overline{(A-B-1)}+1-1 \Leftrightarrow \overline{(A-B-1)}$$

Συνεπώς μπορούμε να εκφράσουμε τις δύο διαφορετικές αφαιρέσεις με τους αριθμούς A και B μέσω των σχέσεων:

$$A-B=A+\bar{B}+1 \quad (3.1)$$

$$B-A=\overline{A+\bar{B}}+0 \quad (3.2)$$

Το μηδέν και το ένα στις παραπάνω σχέσεις μπορούν να εισαχθούν σαν κρατούμενο εισόδου στον αθροιστή, άρα το αποτέλεσμα της αφαίρεσης είναι άμεσα εξαρτώμενο από το κρατούμενο εισόδου, άρα είναι περισσότερο εύχρηστο να χρησιμοποιηθεί ένας τύπος αθροιστή ο οποίος μπορεί να υπολογίσει όλα τα κρατούμενα C_i της άθροισης με κρατούμενο εισόδου μηδέν και κρατούμενο εισόδου μονάδα ταυτόχρονα, ένας τέτοιου είδους αθροιστής είναι ο αθροιστής πρόβλεψης κρατουμένου που παρουσιάζεται στο σχήμα 2.6 του προηγούμενου κεφαλαίου. Σε αυτόν τον αθροιστή τα τελικά κρατούμενα C_i παράγονται μέσω των σχέσεων:

$$C_i = [g_i + (p_i g_{i-1} + p_i p_{i-1} g_{i-2} + p_i p_{i-1} p_{i-2} g_{i-3} + \cdots + g_0) + (p_i p_{i-1} p_{i-2} \dots p_0)]C_{in} \quad (3.3)$$

Όπου

- $g_i = A_i * \overline{B_i}$
- $p_i = A_i \oplus \overline{B_i}$

Τα bit του αριθμού B είναι ανεστραμμένα σύμφωνα με τις σχέσεις 3.1 και 3.2, γιατί η σχέση προς υπολογισμό είναι η $|A - B|$, προφανώς λόγω της βασικής ιδιότητας της απόλυτης τιμής να είναι ανεξάρτητη από το ποιος αριθμός επιλέγεται σαν αφαιρέτης θα προέκυπταν όμοια αποτελέσματα αν η αντιστροφή γινόταν στον A αντί για τον B. Σχεδιάζεται το κύκλωμα λοιπόν ώστε να υπολογίζονται όλα τα κρατούμενα της άθροισης για κρατούμενο εισόδου μηδέν και κρατούμενο εισόδου μονάδα σύμφωνα με τις σχέσεις:

$$C_i = G_i + P_i * C_{in} \begin{cases} G_i \ για \ C_{in} = 0 \\ G_i + P_i \ για \ C_{in} = 1 \end{cases}$$

Όπου τα σήματα G_i και P_i είναι τα τελικά σήματα παραγωγής και διάδοσης κρατουμένου αντίστοιχα, στο τέλος του δυαδικού δέντρου του σχήματος 2.6.

Σε επίπεδο λογικών πυλών οι δύο αφαιρέσεις υπολογίζονται μέσω των σχέσεων:

$$(A - B)_i = A_i \oplus \overline{B_i} \oplus C_{i-1}$$

Όμως από τη σχέση $p_i = A_i \oplus \overline{B_i}$ η σχέση μετατρέπεται σε

$$(A - B)_i = p_i \oplus C_{i-1}$$

Όμως στην αφαίρεση $A - B$ το κρατούμενο εισόδου είναι μονάδα άρα η σχέση παίρνει την μορφή:

$$(A - B)_i = p_i \oplus (G_{i-1} + P_{i-1}) \quad (3.4)$$

Η αντίστοιχη διαδικασία αν εφαρμοστεί και για την σχέση $B - A$ θα προκύψει ο ακόλουθος τύπος

$$(B-A)_i = \overline{p_\iota \oplus \ G_{\iota-1}}$$

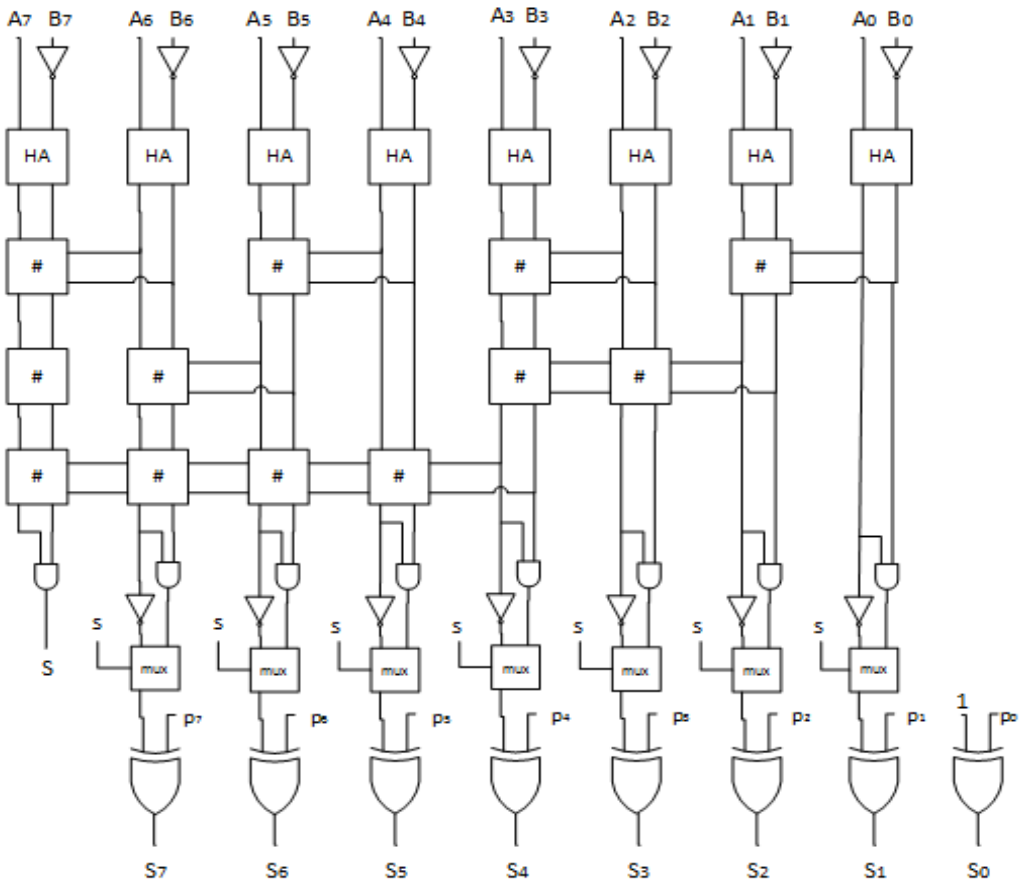
Όμως από το πίνακα αληθείας της πύλης XOR γίνεται αντιληπτό ότι αντί για αντιστροφή του αποτελέσματός της μπορεί να αντιστραφεί μία από τις δύο εισόδους της, έτσι το τελικό αποτέλεσμα είναι:

$$(B-A)_i = p_i \oplus \ \overline{G_{\iota-1}} \quad (3.5)$$

Συνοψίζοντας από όλα τα παραπάνω γίνεται κατανοητό ότι η κάθε πράξη προς υπολογισμό εξαρτάται άμεσα από σήματα G_i και P_i στο τέλος του δυαδικού δέντρου καθώς και από την αντιστροφή της μίας εισόδου του αθροιστή δηλαδή:

- Ο υπολογισμός της πράξης (A-B) γίνεται με επιλογή $C_i=G_i + P_i$ με αντιστροφή του B
- Ο υπολογισμός της πράξης (A-B) γίνεται με επιλογή $C_i=\overline{G_\iota}$ με αντιστροφή του B
- Ο υπολογισμός της πράξης (A+B) γίνεται με επιλογή $C_i=G_i$ χωρίς αντιστροφή του B

Το πρόσημο του αποτελέσματος της κάθε πράξης είναι άμεσα εξαρτώμενο από το κρατούμενο εξόδου αν οι δύο αριθμοί είναι μη-προσημασμένοι. Μία τέτοια μονάδα φαίνεται στο παρακάτω σχήμα για δύο αριθμούς μεγέθους 8bit:



Σχήμα 3.3 Αφαιρέτης απόλυτης τιμής δύο θετικών αριθμών

Τα σήματα p_0-p_7 σχηματίζονται από το πρώτο λογικό στάδιο ημιαθροιστών, όπως περιεγράφηκε στο προηγούμενο κεφάλαιο. Στο τέλος του υπολογισμού και των δύο κρατουμένων υπάρχει ένας πολυπλέκτης ο οποίος υπολογίζει το τελικό κρατούμενο κάθε βάρους ανάλογα με το πρόσημο του αποτελέσματος της αφαίρεσης $A - B$, το πρόσημο του αποτελέσματος ισοδυναμεί με το κρατούμενο εξόδου γι' αυτό το λόγο συμβολίζεται και ως S, αν το S είναι 1 τότε ο πολυπλέκτης στην έξοδο του εμφανίζει το κρατούμενο $C_{out,i} = \overline{C_{0,\iota}}$, όπου $C_{0,i}$ είναι το κρατούμενο με βάρος i και κρατούμενο εισόδου ίσο με το μηδέν, αν το κρατούμενο εξόδου είναι 0 τότε τα τελικά κρατούμενα κάθε βάρους είναι $C_{out,i} = C_{1,i}$ όπου $C_{1,i}$ είναι το κρατούμενο με βάρος i και κρατούμενο εισόδου ίσο με το 1. Η περίπτωση που το πρόσημο του αποτελέσματος ταυτίζεται με το κρατούμενο εξόδου του αθροιστή ισχύει μόνο για την περίπτωση που οι δύο αριθμοί είναι μη προσημασμένοι, αν οι αριθμοί είναι σε μορφή συμπληρώματος ως προς δύο τότε το πρόσημο μπορεί να ληφθεί από το ανεστραμμένο κρατούμενο εξόδου με τα αντεστραμμένα τα MSB των δύο αριθμών προς αφαίρεση. Αν οι δύο αριθμοί προέρχονται από ένα δέντρο Wallace όμως θα μελετηθεί στις επόμενες παραγράφους τότε το πρόσημο της αφαίρεσης δίνεται από το MSB του αποτελέσματος για κρατούμενο εισόδου 1. Αυτή η μονάδα θα χρησιμοποιηθεί για την υλοποίηση ειδών MAC ώστε να παράγεται το αποτέλεσμα απευθείας σε μορφή προσήμου-μέτρου ώστε να μην χρειάζεται η μετατροπή από συμπλήρωμα ως προς δύο.

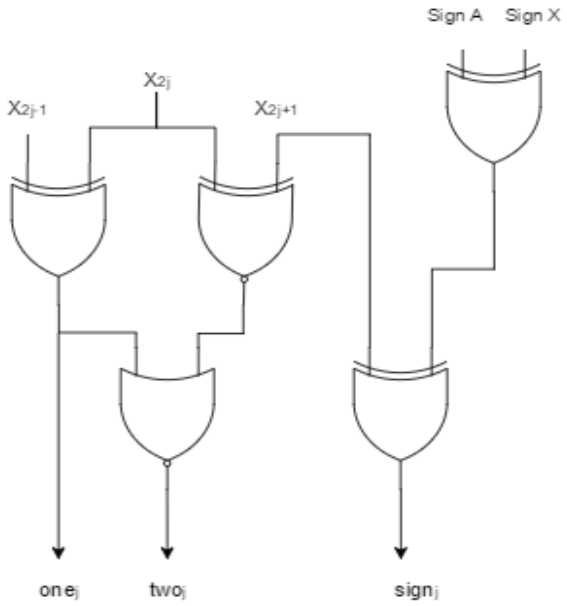
3.5 Υλοποίηση MAD

Σε αυτή τη παράγραφο θα αναλυθούν δύο διαφορετικοί σχεδιασμού κυκλώματος MAD το οποίο υπολογίζει την αριθμητική πράξη $S = A * X + D$.

3.5.1 Σχεδίαση με χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο

Για αυτή τη σχεδίαση θα χρησιμοποιηθεί μία μονάδα μετατροπής του τελικού αποτελέσματος από αναπαράσταση συμπληρώματος ως προς δύο σε πρόσημο μέτρο η οποία φαίνεται στο σχήμα 3.2, για να υπολογιστεί το τελικό αποτέλεσμα σωστά σε μορφή συμπληρώματος ως προς δύο πρέπει να γίνει αλλαγή μορφής αναπαράστασης και στις εισόδους του κυκλώματος. Αυτή η μετατροπή μπορεί να γίνει χωρίς τη χρήση του σχήματος 3.2, για να επιτευχθεί αυτό θα γίνει προσαρμογή του κωδικοποιητή Modified Booth ώστε να παράγει μία κωδικοποίηση είτε για τον αριθμό που δέχεται σαν είσοδο είτε για τον αντίθετό του.

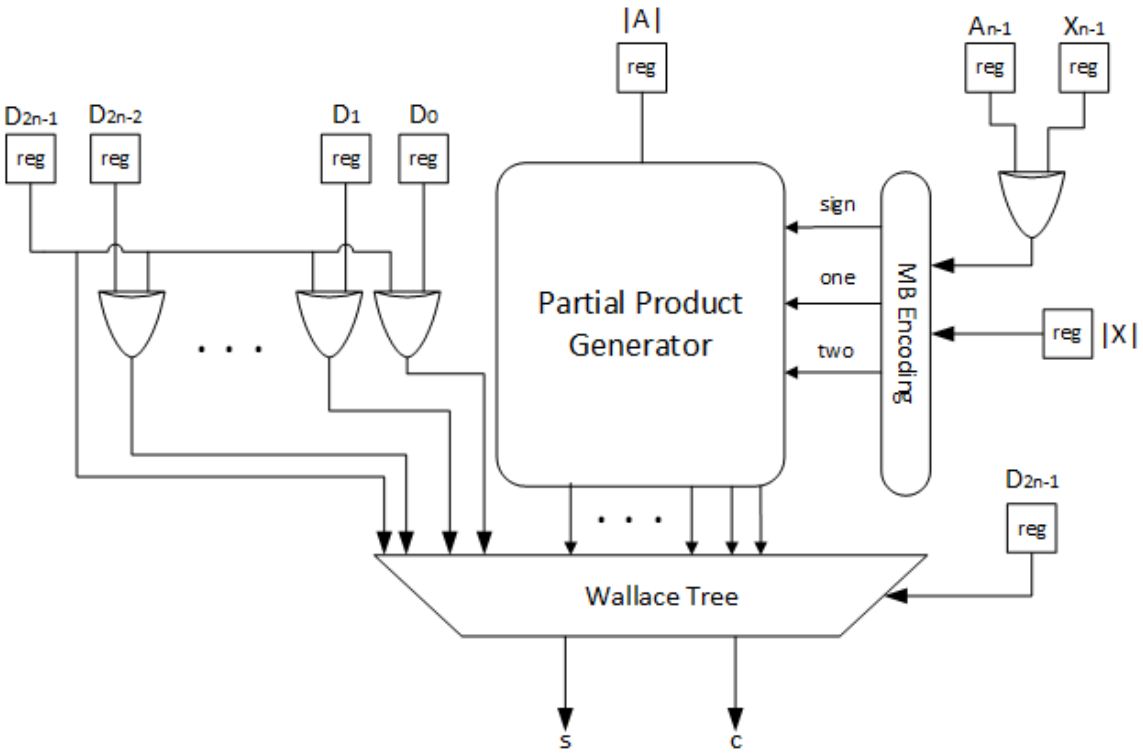
Για να επιτευχθεί αυτή τη μετατροπή στη MB κωδικοποίηση το σήμα sign το οποίο παράγεται θα είναι ανεστραμμένο ανάλογα με το αν το πρόσημο του πολλαπλασιασμού είναι θετικό ή αρνητικό. Αυτή είναι μια βασική ιδιότητα όλων των αναπαραστάσεων σε μορφή προσημασμένων ψηφίων όπου αν αντιστραφούν τα πρόσημα όλων των ψηφίων το αποτέλεσμα θα είναι ο αντίθετος αριθμός. Η προσαρμοσμένη κυκλωματική υλοποίηση της MB κωδικοποίησης φαίνεται στο επόμενο σχήμα:



Σχήμα 3.4 Προσαρμογή κυκλώματος MB ώστε να παράγει αντίθετους αριθμούς

Στο σχήμα φαίνεται ότι τα πρόσημα του κωδικοποιημένου αριθμού X^{MB} αντιστρέφονται μόνο στην περίπτωση όπου η πύλη XOR των πρόσημων των αριθμών A και X είναι 1 δηλαδή μόνο όταν οι δύο αριθμοί είναι ετερόσημοι, το bit X_{n-1} που δέχεται σαν είσοδο το κύκλωμα κωδικοποίησης είναι σταθερά ίσο με το μηδέν, αντίστοιχα και το bit A_{n-1} δέχεται σαν είσοδο το κύκλωμα παραγωγής μερικών γινομένων, αυτά τα επιπλέον bit εφαρμόζονται ώστε η λειτουργία του κυκλώματος να είναι εύκολη στη κατανόηση, στην τελική σύνθεση του κυκλώματος ο compiler θα αντιληφθεί ότι αυτά τα bit είναι σταθερά μηδέν και θα εφαρμόσει τις σχετικές υλοποιήσεις.

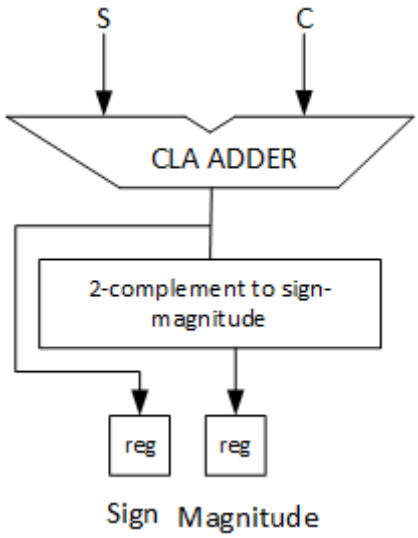
Για την είσοδο D θα εφαρμοστεί μετατροπή σε συμπλήρωμα ως προς δύο όμως και σε αυτή τη περίπτωση θα αποφευχθεί η χρήση μετατροπέα. Τα bit του αριθμού D αντιστρέφονται όταν το πρόσημο του D είναι αρνητικό, τα bit αυτά στη συνέχεια εφαρμόζονται ως είσοδο στον δεντρικό συμπιεστή Wallace, μαζί με το πρόσημο του, όλα τα παραπάνω γίνονται περισσότερο κατανοητά από το παρακάτω σχήμα:



Σχήμα 3.5 Γεννήτρια παραγωγής μερικών γινομένων με κωδικοποίηση MB και δέντρο Wallace.

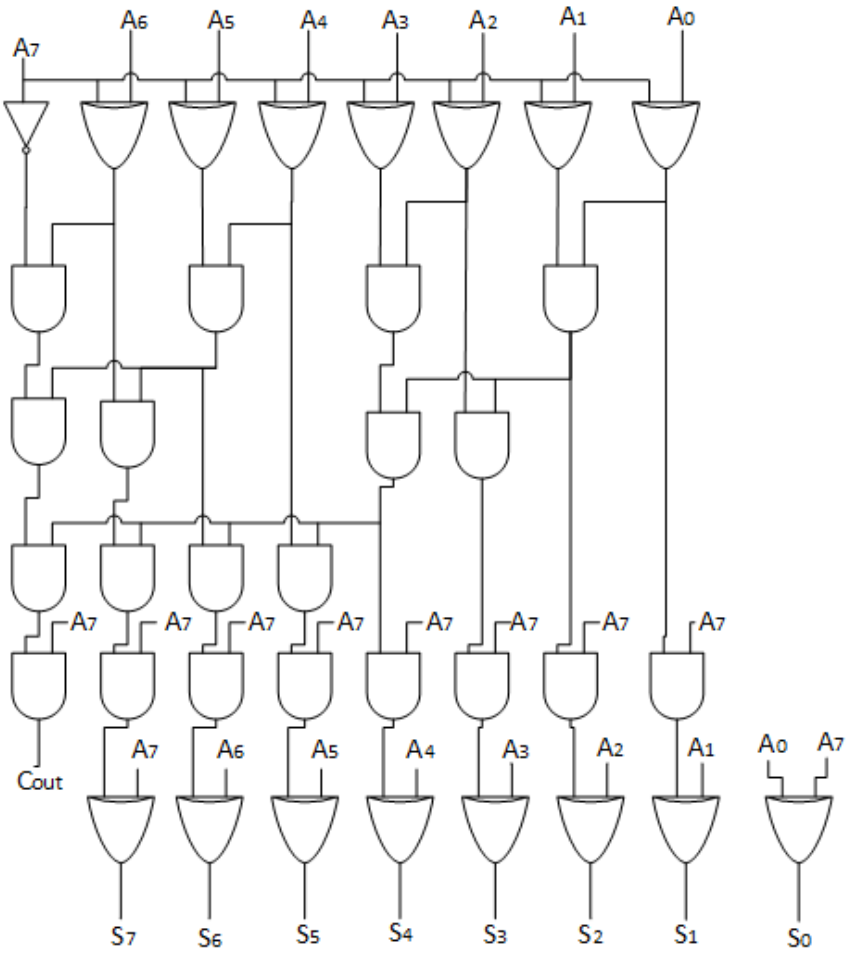
Στο ακόλουθο σχήμα τα ψηφία A_{n-1} , X_{n-1} , D_{2n-1} αντιπροσωπεύουν τα πρόσημα των αριθμών A,X,D αντίστοιχα, το πρόσημο του αριθμού D εισέρχεται στο δέντρο Wallace ως μία επιπρόσθετη μονάδα, ώστε να ολοκληρωθεί η μετατροπή του αριθμού D σε συμπλήρωμα ως προς 2. Μετά από όλες αυτές τις μετατροπές το κύκλωμα συμπεριφέρεται σαν οι είσοδοι να είχαν δοθεί σε μορφή συμπληρώματος ως προς δύο. Το τελικό αποτέλεσμα προκύπτει μέσω της χρήσης ενός αθροιστή ο οποίος παράγει το τελικό αποτέλεσμα, αν το αποτέλεσμα είναι αρνητικό τότε είναι αναγκαστική

η χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε μέτρο πρόσημο, αυτό φαίνεται στο ακόλουθο σχήμα:



Σχήμα 3.6 Αθροιστής πρόβλεψης κρατουμένου και μετατροπέας στη μορφή sign-magnitude.

Για το μετατροπέα από συμπλήρωμα ως προς δύο σε sign-magnitude χρησιμοποιήθηκε ένας CLA αθροιστής, σύμφωνα με το σχήμα 3.2, παρατηρείται ότι ο αθροιστής που χρησιμοποιείται για την υλοποίηση του μετατροπέα είναι αρκετά απλοποιημένος σε σχέση με τον προηγούμενο CLA διότι η μία είσοδος του είναι πάντα ίση με το μηδέν. Η απλοποίηση αυτή σε επίπεδο λογικών πυλών φαίνεται στο επόμενο σχήμα:



Σχήμα 3.7 Κύκλωμα μετατροπής της αναπαράστασης από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο για έναν αριθμό μεγέθους 8bit.

Είναι εμφανές ότι το πρώτο επίπεδο ημιαθροιστών δεν είναι απαραίτητο, διότι τα σήματα παραγωγής g_i είναι ίσα με το μηδέν πάντα ενώ τα σήματα διάδοσης p_i είναι ίσα με την είσοδο A_i, Τα κυκλώματα υλοποίησης της πράξης # μπορούν να απλοποιηθούν και αυτά ώστε να υπολογίζουν μόνο τα τελικά σήματα P_i, τέλος το τελευταίο επίπεδο πυλών XOR υπολογίζει το τελικό αποτέλεσμα. Στη περίπτωση που το κρατούμενο εισόδου (δηλαδή το MSB του αριθμού εισόδου) είναι 0 είναι εμφανές ότι η είσοδος και η έξοδος ταυτίζονται. Επίσης ακριανή στήλη με λογικές πύλες AND παραλείπεται λόγω ότι στον αθροιστή εισέρχεται ο αριθμός χωρίς το MSB όπως φαίνεται στο σχήμα 3.2 απλά για λόγους πληρότητας συμπεριλαμβάνονται όλες οι στήλες.

Η συνολική καθυστέρηση με την οποία επιβαρύνεται το συνολικό κύκλωμα μέσω της χρήσης μίας τέτοιας μονάδας η οποία ονομάζεται increment στη συνήθη βιβλιογραφία είναι:

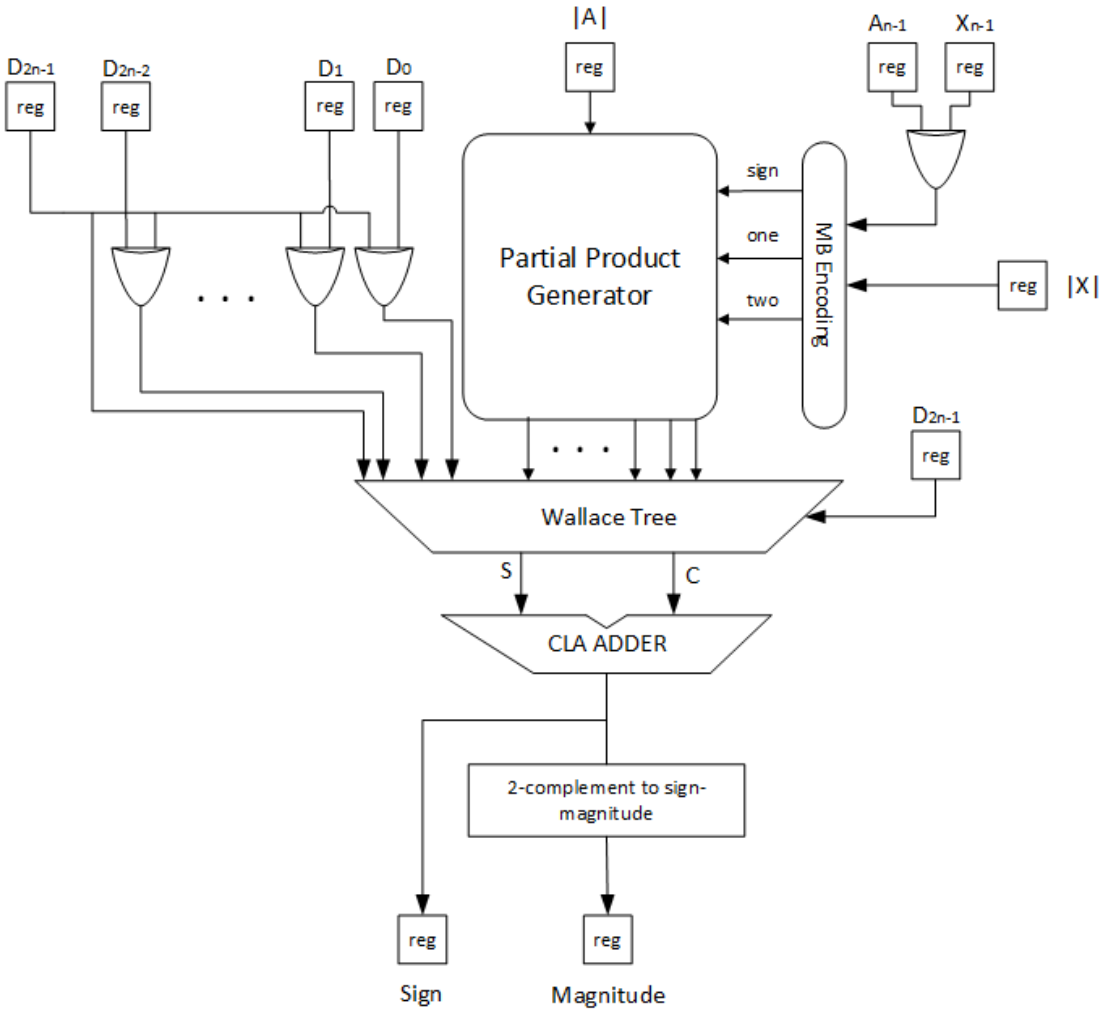
$$T_{increment} = [log(2N) + 1] * T_{and} + 2 * T_{XOR} \quad (3.6)$$

Όπου N το μέγεθος των εισόδων A και X. Παρατηρείται λοιπόν ότι η επιβάρυνση στην ταχύτητα του κυκλώματος είναι άμεσα εξαρτώμενη από το μέγεθος των αριθμών προς επεξεργασία, το ίδιο συμβαίνει και με την επιφάνεια του κυκλώματος η οποία επιβαρύνεται σύμφωνα με τον τύπο

$$A_{increment} = [(N-1) * log(2N) + 2N] * A_{AND} + 4N * A_{XOR} \quad (3.7)$$

Και στις δύο περιπτώσεις ο λογάριθμος log(2N) συμβολίζει τα λογικά επίπεδα του δυαδικού δέντρου ενός αθροιστής CLA.

Το συνολικό κύκλωμα έχει την ακόλουθη μορφή:



Σχήμα 3.8 Σχεδίαση MAC για τον υπολογισμό της αριθμητικής πράξης $A * X + D$ με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο.

Στο κύκλωμα αυτό οι είσοδοι A και X έχουν μέγεθος n-bit ενώ η είσοδος D μπορεί να έχει μέγεθος μέχρι 2*n-bit, περίπτωση overflow μπορεί να υπάρχει μόνο όταν η είσοδος D έχει μήκος 2*n-bit

και προφανώς στην περίπτωση όπου οι αριθμοί A∗X και D είναι ομόσημοι, τότε αν το τελικό αποτέλεσμα προκύψει με πρόσημο αντίθετο από τα πρόσημα της άθροισης υπάρχει περίπτωση overflow. Στον ακόλουθο πίνακα περιγράφονται αυτές οι περιπτώσεις:

Πίνακας 3.1 *Σχέσεις μεταξύ πρόσημων εισόδου-εξόδου και υπερχείλισης.*

Πρόσημο A∗X	**Πρόσημο D**	**Πρόσημο Αποτελέσματος**	**overflow**
Θετικό	Θετικό	Θετικό	0
Θετικό	Θετικό	Αρνητικό	1
Θετικό	Αρνητικό	Θετικό	0
Θετικό	Αρνητικό	Αρνητικό	0
Αρνητικό	Θετικό	Θετικό	0
Αρνητικό	Θετικό	Αρνητικό	0
Αρνητικό	Αρνητικό	Θετικό	1
Αρνητικό	Αρνητικό	Αρνητικό	0

Η περίπτωση υπερχείλισης συμβαίνει όταν το μέτρο του τελικού αποτελέσματος χρειάζεται 2*n-bit για να υπολογιστεί, τότε το τελευταίο bit χάνεται και λόγω ότι ο καταχωρητής εξόδου διαθέτει 2*n-1 για την απεικόνιση του μέτρου του αποτελέσματος, σε αυτή τη περίπτωση το τελικό αποτέλεσμα που απεικονίζεται στην έξοδο του MAC είναι λανθασμένο οπότε ο δείκτης overflow γίνεται μονάδα, είναι σημαντικό να σημειωθεί ότι η προσαρμογή των μερικών γινομένων έχει γίνει ώστε το τελικό αποτέλεσμα να απεικονίζεται σωστά σε 2*n bit οπότε δεν είναι δυνατός εντοπισμός overflow μέσω της σύγκρισης των κρατουμένων C_{2n-1} και C_{2n-2}, κάτι το οποίο ισχύει στη γενική περίπτωση άθροισης δύο αριθμών σε μορφή συμπληρώματος ως προς 2.

Η συνθήκη overflow που απεικονίζει ο πίνακας 3.1 ισχύει πάντα εκτός από μία περίπτωση, αυτή συμβαίνει όταν το τελικό αποτέλεσμα είναι ο αριθμός -2^{n-1}, αυτός ο αριθμός είναι ιδιαίτερος διότι όπως αναλύθηκε και στο προηγούμενο κεφάλαιο μπορεί να απεικονιστεί σε n-bit μόνο στη μορφή συμπληρώματος ως προς δύο ενώ για αναπαράσταση σε πρόσημο μέτρο χρειάζονται n+1 bit. Το overflow σε αυτή τη περίπτωση μπορεί να εντοπιστεί από το κρατούμενο εξόδου στον αθροιστή μετατροπής από 2 complement σε sign-magnitude.

Το σήμα ελέγχου overflow είναι απαραίτητο στη περίπτωση MAC όπου γίνεται accumulation των αριθμών εισόδου δηλαδή εκτελείται η πράξη $\sum A_i * X_i$. Σε αυτή τη περίπτωση το ενδεχόμενο overflow είναι αρκετά συχνό οπότε χρειάζεται να γίνει είτε ένα reset είτε ένα saturation στον καταχωρητή όπου γίνεται accumulation, αυτές οι περιπτώσεις αναλύονται και στις απόμενες παραγράφους όπου μελετώνται MAC που εκτελούν αντίστοιχες πράξεις.

3.5.2 Σχεδίαση με χρήση αθροιστή απόλυτης τιμής.

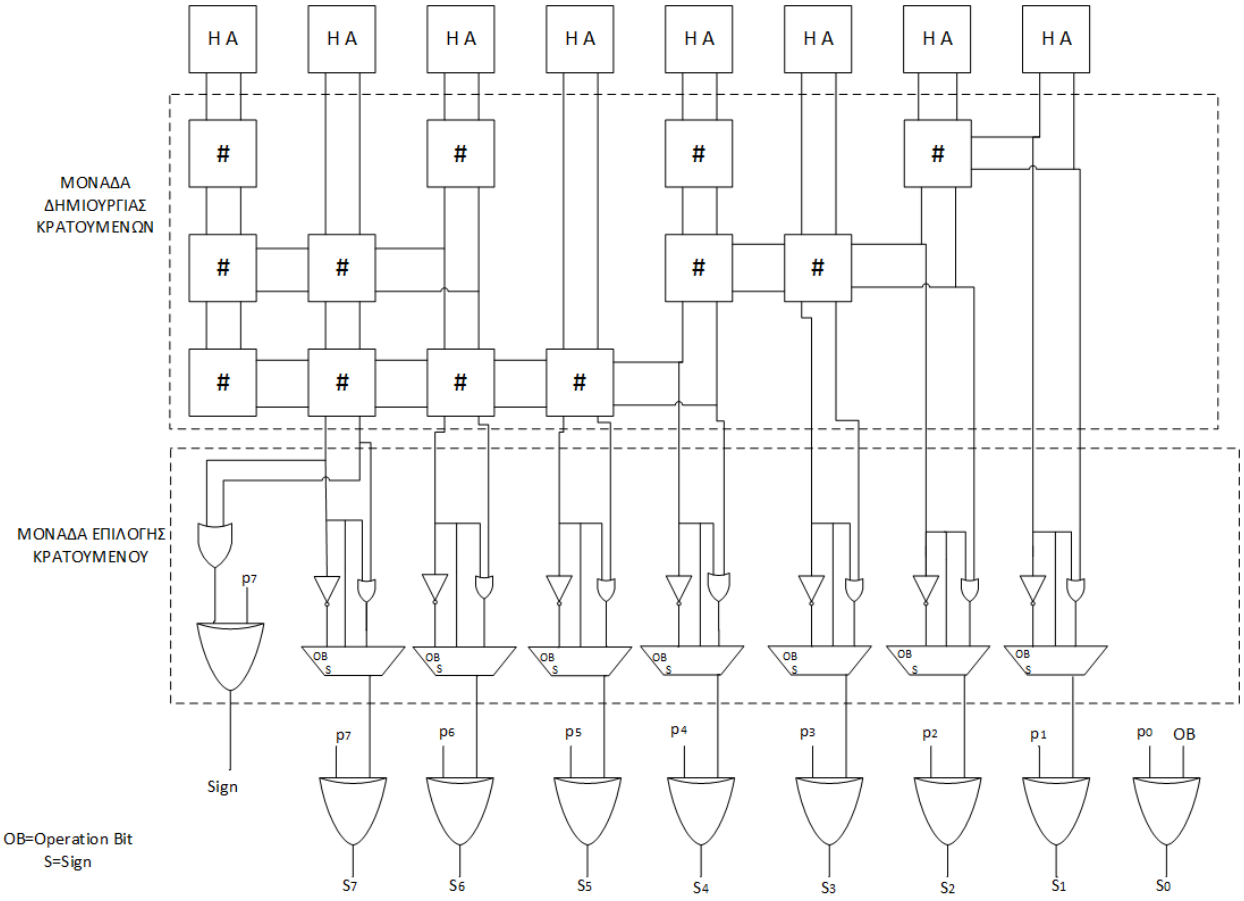
Η υλοποίηση με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο αριθμητική έχει ένα βασικό μειονέκτημα, αυτό είναι ότι ο μετασχηματισμός στη μορφή πρόσημου-μέτρο προσθέτει μια χρονική καθυστέρηση ανάλογη με το λογάριθμο log(2∗N), συνεπώς για αριθμούς μεγάλου μήκους, η καθυστέρηση είναι αρκετά σημαντική, εξίσου σημαντική είναι και η επιβάρυνση στη επιφάνεια του κυκλώματος. Σε αυτή τη παράγραφο θα γίνει μία προσαρμογή στον αθροιστή του σχήματος 3.3 ώστε να γίνεται δυνατή η λειτουργία σε έναν MAC για να εκτελείται η αριθμητική πράξη $A * X + D$, χωρίς τη χρήση μετατροπέα, οι αριθμοί A και B έχουν μήκος λέξης N-bit ενώ ο αριθμός D έχει μήκος λέξης 2∗N bit.

Τα προβλήματα που προκύπτουν στην προσπάθεια χρήσης ενός τέτοιου αθροιστή είναι τα εξής:

- Στην περίπτωση υπολογισμού της πράξης $-A * X + D$ είναι αρκετά δύσκολος ο υπολογισμός της ποσότητας $\overline{A * X}$ για την εισαγωγή στον αφαιρέτη απόλυτης τιμής,
- Στην περίπτωση που το γινόμενο και ο αριθμός προς πρόσθεση έχουν το ίδιο πρόσημο τότε πρέπει να εκτελέσει πρόσθεση και όχι αφαίρεση των δύο αριθμών, χωρίς τη χρήση δύο ξεχωριστών μονάδων.

Η πρώτη δυσκολία μπορεί να αντιμετωπιστεί αρκετά εύκολα, χρησιμοποιώντας την ιδιότητα της απόλυτης τιμής: $|D - (A * X)| = |(A * X) - D|$ με άλλα λόγια ανεξάρτητα από το ποιος αριθμός είναι αρνητικός και ποιος θετικός, αν έχουν διαφορετικό πρόσημο τότε πάντα αντιστρέφεται ο αριθμός D. Το πρόσημο του αποτελέσματος είναι αυτό που παράγει ο αθροιστής αν το D είναι αρνητικό δηλαδή αν ήταν σωστή η αντιστροφή του και το ανάποδο από αυτό που παράγει ο αθροιστής αν το πρόσημο του D είναι θετικό δηλαδή αν υπολογίστηκε η απόλυτη τιμή της αντίθετης αφαίρεσης. Αν και ο αριθμός $A * X$ και ο αριθμός D έχουν το ίδιο πρόσημο τότε το πρόσημο του αποτελέσματος είναι το ίδιο με το πρόσημο του D, αφού η πρόσθεση δύο ομόσημων αριθμών παράγει αποτέλεσμα με το ίδιο πρόσημο.

Προκειμένου να αντιμετωπιστεί η δεύτερη σχεδιαστική δυσκολία θα πρέπει να μετατρέψουμε το κύκλωμα του σχήματος 3.3 ώστε να προσαρμοστεί στις ανάγκη υπολογισμού είτε πρόσθεσης είτε της αφαίρεσης δύο αριθμών, αυτό επιτυγχάνεται μέσω μίας προσαρμογής στους πολυπλέκτες έτσι ώστε να επιλέγουν τα κατάλληλα κρατούμενα στην περίπτωση της πρόσθεσης. Όταν οι αριθμοί είναι ομόσημοι τότε το σωστό κρατούμενο επιλογής είναι το $C_{i,0} = G_i$, δηλαδή τα σήματα G_i, στο τέλος του δυαδικού δέντρου. Η προσαρμογή των πολυπλεκτών γίνεται με την προσθήκη ενός bit ελέγχου (ονομάστηκε operation bit) το οποίο είναι 0 όταν οι εκτελείται πρόσθεση και τα τελικά κρατούμενα είναι τα G_i, όταν το operation bit είναι 1 τότε εκτελείται αφαίρεση όπου τα τελικά κρατούμενα είναι τα $G_i + P_i$ αν το πρόσημο εξόδου είναι θετικό και τα $\overline{G_i}$ αν το πρόσημο εξόδου είναι αρνητικό. Όλα τα παραπάνω μπορούν να γίνουν κατανοητά μέσω του ακόλουθου σχήματος:



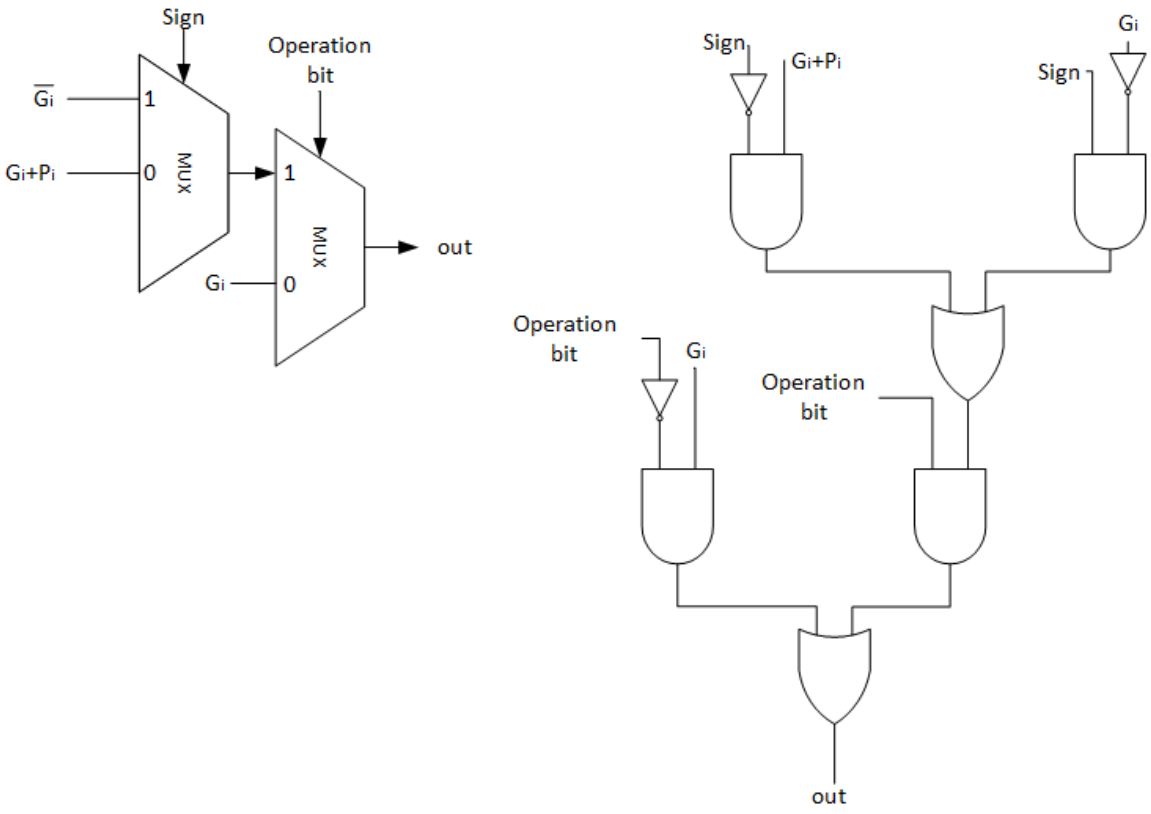
Σχήμα 3.9 Προσαρμογή στον αφαιρέτη απόλυτης τιμής για λειτουργία στον MAC

Ο πίνακας αληθείας του bit εξόδου των πολυπλεκτών σαν συνάρτηση των bit εισόδου (operation bit και sign είναι ο εξής:

Πίνακας 3.2 *Πίνακας αληθείας των πολυπλεκτών του αθροιστή απόλυτης τιμής.*

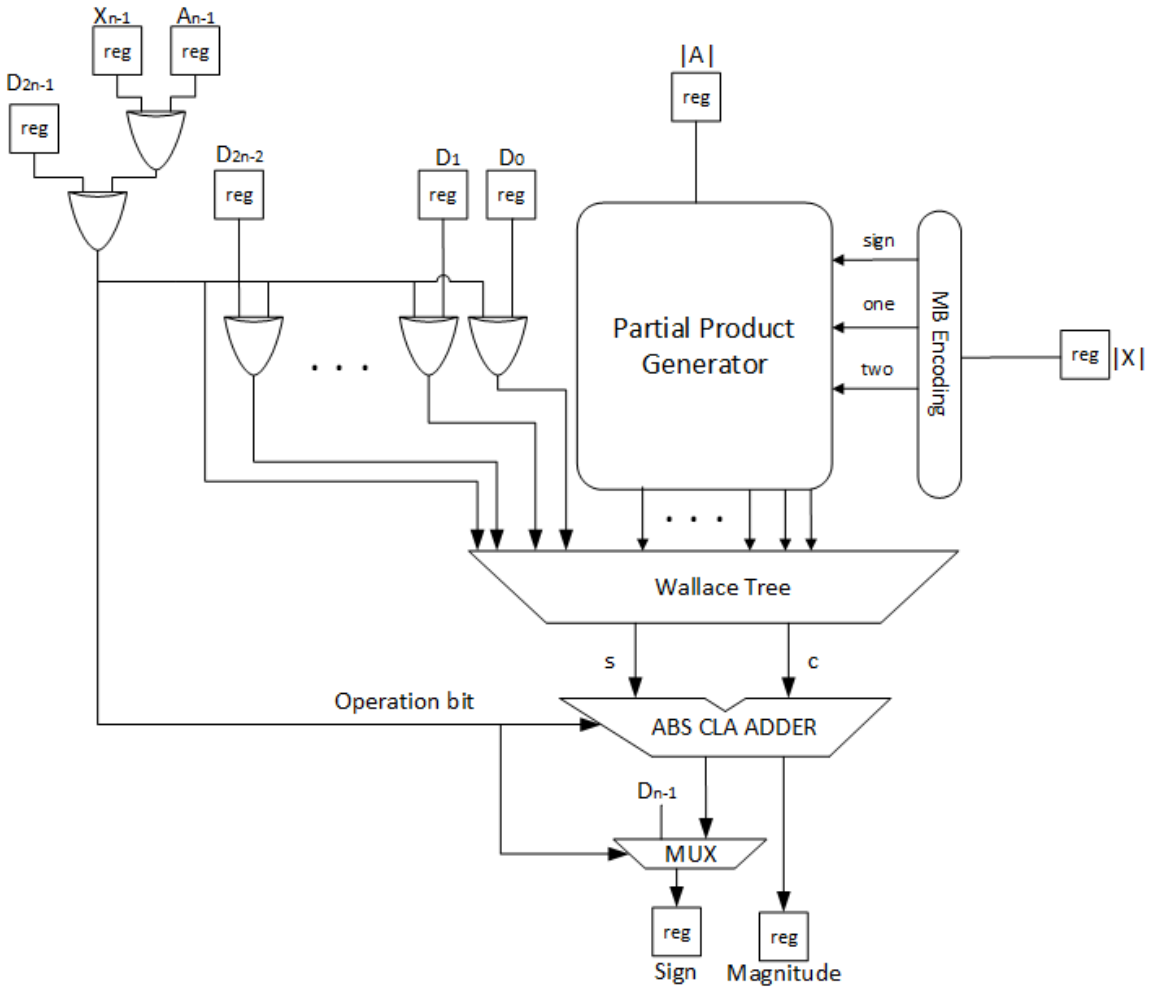
Operation bit	Sign bit	Κρατούμενο εξόδου
0	1	G_i
0	0	G_i
1	0	$G_i + P_i$
1	1	$\overline{G_i}$

Η ανάλυση της λειτουργίας τους σε επίπεδο λογικών πυλών παρουσιάζεται στο επόμενο σχήμα.



Σχήμα 3.10 Ανάλυση λειτουργίας πολυπλεκτών σε επίπεδο λογικών πυλών.

Μπορεί να παρατηρηθεί αρκετά εύκολα ότι η ακραία αριστερή στήλη από κυκλώματα που υλοποιούν την λογική πράξη (#) μπορεί να παραληφθεί, αυτό γίνεται διότι η προσαρμογή των μερικών γινομένων στο Wallace έγινε με βάση ώστε το αποτέλεσμα να αποτυπώνεται σε 2N bit, συνεπώς δεν μπορεί να ληφθεί η πληροφορία για το πρόσημο του αποτελέσματος από το κρατούμενο εξόδου, αυτό έχει μια επιβάρυνση στην καθυστέρηση του κυκλώματος ίση με T_{XOR} αλλά και μια ελάφρυνση στην επιφάνεια κατά $\log(2 * N) * A_{\#}$, Το προσαρμοσμένο κύκλωμα του MAC ώστε να γίνεται η σωστή λειτουργία της μονάδας φαίνεται στο ακόλουθο σχήμα:



Σχήμα 3.11 Λειτουργία MAC με χρήση αθροιστή απόλυτης τιμής.

Όπου με ABS CLA ADDER έχει ονομαστεί ο αθροιστής απόλυτης τιμής που αναλύθηκε προηγουμένως ενώ η λειτουργία του πολυπλέκτη MUX εξηγήθηκε προηγουμένως για το τελικό υπολογισμό του προσήμου. Οι κύριες διαφορές μεταξύ των δύο τρόπων σχεδίασης είναι ότι δεν προστίθεται μονάδα στο δέντρο Wallace, σε περίπτωση αντιστροφής του αριθμού D, στη MB κωδικοποίηση το τελευταίο bit του σήματος προσήμου είναι πάντα 0 και κυρίως δεν χρησιμοποιείται η μονάδα αντιστροφής των αρνητικών αριθμών σε θετικούς.

Ο πίνακας αληθείας του τελικού πολυπλέκτη για τον υπολογισμό του τελικού πρόσημου είναι:

Πίνακας 3.3 *Πίνακας αληθείας πολυπλέκτη υπολογισμού προσήμου.*

Operation bit	Πρόσημο αριθμού D	Πρόσημο εξόδου αθροιστή	Τελικό πρόσημο
0	0	1	0
0	0	0	0
0	1	1	1
0	1	0	1
1	0	1	0
1	0	0	1
1	1	1	1
1	1	0	0

Ο πολυπλέκτης σε επίπεδο λογικών κατασκευάζεται παρόμοια με τους προηγούμενους πολυπλέκτες επιλογής κρατουμένου του αθροιστή απόλυτης τιμής. Ο υπολογισμός του overflow σε αυτή τη περίπτωση είναι αρκετά πιο απλός, διότι ταυτίζεται με το MSB του αποτελέσματος του αθροιστή, αυτό συμβαίνει διότι στη περίπτωση της άθροισης οι δύο αριθμοί είναι πάντα θετικοί αφού λαμβάνονται τα μέτρα τους ως είσοδοι, όταν υπάρχει overflow το μέτρο χρειάζεται 2*n bit για να αποτυπωθεί, έτσι το bit στη θέση 2*n-1 γίνεται 1, όμως στην έξοδο το περισσότερο σημαντικό bit διατίθεται για το πρόσημο του αποτελέσματος οπότε δεν γίνεται σωστή παρουσίαση του. Επίσης ο τελευταίος πολυπλέκτης λειτουργεί παράλληλα με τους υπόλοιπους του αθροιστή και άρα δεν επιβαρύνει το κύκλωμα με επιπλέον καθυστέρηση.

3.5.3 Θεωρητική ανάλυση επιφάνειας και καθυστέρησης των δύο σχεδιασμών.

Προκειμένου να γίνει κατανοητή η θεωρητική ανάλυση θα παρουσιαστεί ο πίνακας με την ανάλυση που έχει γίνει σε βασικές μονάδες που χρησιμοποιούνται στους παραπάνω σχεδιασμούς ως προς επιφάνεια και καθυστέρηση.

Πίνακας 3.4 *Μετρικές επιφάνειας και καθυστέρησης σε λογικές πύλες στοιχείων κοινής χρήσης.*

Κύκλωμα	Επιφάνεια (A_G)	Καθυστέρηση (T_G)
Ημιαθροιστής (HA)	3 A_G	2 T_G
Πλήρης αθροιστής (FA)	7 A_G	4 T_G
Πολυπλέκτης 2 σε 1(MUX)	3 A_G	2 T_G
Καταχωρητής (REG)	6 A_G	2 T_G

Χρήση αθροιστή απόλυτης τιμής

Παρατηρείται ότι οι επιπλέον πολυπλέκτες που έχουν τοποθετηθεί έχουν συνολική επιφάνεια και καθυστέρηση ίση με δύο πολυπλέκτες 2 σε 1, άρα όσον αφορά την επιφάνεια και καθυστέρηση που προσθέτουν στο συνολικό κύκλωμα είναι:

$$A_{mux} = 6A_G * (2N) \quad (3.8)$$

$$T_{mux} = 4 * T_G \quad (3.9)$$

Για την MB κωδικοποίηση και παραγωγή μερικών γινομένων η επιφάνεια και η καθυστέρηση με την οποία επιβαρύνεται το κύκλωμα είναι παρόμοια με εκείνη για αριθμούς σε μορφή συμπληρώματος ως προς δύο:

$$A_{MBPPG} = 5 * \left(\frac{N}{2} - 1\right) * A_G + 5 * (N) * \left(\frac{N}{2} - 1\right) * A_G \quad (3.10)$$

$$T_{MBPPG} = 5 * T_G \quad (3.11)$$

Για την αντιστροφή του D χρησιμοποιούνται 2 πύλες XOR για τον υπολογισμό του operation bit και 2N-1 πύλες XOR για την αντιστροφή του αριθμού D, συνολικά λοιπόν η επιφάνεια και καθυστέρηση είναι:

$$A_{reverseD} = (2N - 1)A_G + 2A_G \quad (3.12)$$

$$T_{reverseD} = 6 * T_G \quad (3.13)$$

Επειδή η αντιστροφή του αριθμού D και η δημιουργία των μερικών γινομένων γίνεται παράλληλα η συνολική καθυστέρηση είναι η μεγαλύτερη από τις δύο δηλαδή η $6T_G$.

Για το δεντρικό συμπιεστή Wallace δεν υπάρχει ακριβής ακολουθία για τον υπολογισμό των επιπέδων του, όπως έχει αναφερθεί και σε προηγούμενη παράγραφο για να βρεθούν τα επίπεδα ενός δέντρου Wallace με N μερικά γινόμενα αρκεί να γίνει διαίρεση του αριθμού N με το τρία, να αθροιστεί το υπόλοιπο της διαίρεσης και το διπλάσιο του πηλίκου του και στο νέο αριθμό να

επαναληφθεί η διαδικασία μέχρι να εμφανιστεί ο αριθμός δύο. Ο αριθμός των επιπέδων του Wallace είναι ίσος με τις διαιρέσεις που έγιναν, παρουσιάζεται ένας πίνακας για τα συνηθέστερα πλήθη μερικών γινομένων στο δέντρο Wallace.

Πίνακας 3.5 *Επίπεδα δεντρικού συμπιεστή Wallace ανάλογα με τον αριθμό των μερικών γινομένων(PP).*

Επίπεδα Δέντρου Wallace						
PP	5-6	7-9	10-13	14-19	20-28	29-42
Depth	3	4	5	6	7	8

Για την επιφάνεια χρησιμοποιούνται θεωρητικά 2N πλήρεις αθροιστές για κάθε CSA αθροιστή μέσα στο δέντρο, αλλά πρακτικά αυτό δεν συμβαίνει διότι πολλές θέσεις από τις 2N είναι κενές, αυτό περιπλέκει αρκετά το τύπο υπολογισμού τις επιφάνειας διότι πρέπει να ληφθεί υπόψιν η κατανομή των μερικών γινομένων μέσα στο δέντρο, γι' αυτό το λόγο θα συμβολιστεί απλά A_{tree} η επιφάνεια του δέντρου.

Ο υπολογισμός της θεωρητική καθυστέρησης του δέντρου είναι πιο απλός αφού εξαρτάται μόνο από τον αριθμό των επιπέδων δηλαδή μόνο από τον αριθμό των μερικών γινομένων.

$$T_{Wtree} = Depth\left(\frac{N}{2} + 2\right) * T_{FA} = Depth\left(\frac{N}{2} + 2\right) * 4T_G \qquad (3.14)$$

Ο αριθμός των καταχωρητών που χρησιμοποιούνται είναι 6N, 2N λόγω των αριθμών A και B, 2N λόγω του αριθμού D και 2N για το αποτέλεσμα. Για τον CLA αθροιστή που στο τέλος του δέντρου Wallace υπολογίστηκε η επιφάνεια και η καθυστέρηση των πολυπλεκτών, για το κομμάτι της άθροισης αν το μέγεθος των εισόδων είναι δύναμη του δύο τότε το υλικό και η καθυστέρηση που χρησιμοποιείται είναι:

$$A_{CLA} = 2 * N * A_{HA} + \log(2N) * (N-1) * A_{\#} + 2 * N * A_{XOR} + 2 * N\, A_{OR}$$
$$A_{CLA} = 2 * N * 3A_G + \log(2N) * (N-1) * 3A_G + 2 * N * 2A_G + 2 * N * A_G \qquad (3.15)$$

$$T_{CLA} = T_{HA} + \log(2N)\, T_{\#} + T_{OR} + 2T_{XOR}$$
$$T_{CLA} = 2T_G + \log(2N) * 2T_G + T_G + 4T_G \qquad (3.16)$$

Το συνολικό υλικό που χρησιμοποιείται φαίνεται στο παρακάτω πίνακα.

Πίνακας 3.6 *Απαραίτητο υλικό καθώς και επιπτώσεις στην επιφάνεια και τη καθυστέρηση.*

Στοιχείο	Επιφάνεια	Καθυστέρηση
MB encoding-PPG-Reverse D	$A_{MBPPG}+A_{reverseD}$	$6T_G$
Wallace Tree	A_{TREE}	$Depth(N/2+2)*4T_G$
CLA Absolute value	$A_{CLA}+A_{mux}$	$T_{CLA}+T_{MUX}$
Registers	$36N*A_G$	$4T_G$

Χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο.

Η κύρια διαφοροποίηση μεταξύ των δύο υλοποιήσεων βρίσκεται στη μετατροπή του αποτελέσματος σε μορφή προσήμου-μέτρου, μονάδα αυτή προσθέτει επιπλέον επιφάνεια και καθυστέρηση ίση με

$$A_{increment} = [(N-1)*log(2N)+2N]*A_{AND}+4N*A_{XOR} = [(N-1)*\log(2N)+2N]*A_G+4N*2A_G \qquad (3.17)$$

$$T_{increment} = [log(2N)+1]*T_{and}+2*T_{XOR} = [log(2N)+1]*T_G+4*T_G \qquad (3.18)$$

Ο αθροιστής πρόβλεψης κρατουμένου που χρησιμοποιείται είναι όμοιος με αυτόν που αναλύθηκε στη προηγούμενη σχεδίαση με μόνη διαφορά ότι το κρατούμενο εισόδου του είναι πάντα μηδέν οπότε:

$$A_{CLA} = 2*N*A_{HA}+\log(2N)*(N-1)*A_{\#}+2*N*A_{XOR} \qquad (3.19)$$

$$T_{CLA} = T_{HA}+\log(2N)\,T_{\#}+T_{XOR} \qquad (3.20)$$

Η MB κωδικοποίηση έχει και αυτή λίγες διαφορές στην επιφάνεια και την καθυστέρηση λόγω ότι υπάρχει η επιπλέον αντιστροφή του σήματος sign στην κωδικοποίηση όπως φαίνεται και στο σχήμα 3.4 έτσι η επιφάνεια και καθυστέρηση της κωδικοποίησης σε αυτό το σχεδιασμό είναι:

$$A_{MBPPG} = 5*\left(\frac{N}{2}-1\right)*A_G+N*A_G+2A_G+5*N*\left(\frac{N}{2}\right)*A_G \qquad (3.21)$$

$$T_{MBPPG} = 8 * T_G \qquad (3.22)$$

Το δέντρο Wallace έχει αντίστοιχα την ίδια επιφάνεια και καθυστέρηση με πριν αφού ο αριθμός των μερικών γινομένων παραμένει ίδιος, ενώ η αντιστροφή του αριθμού D γίνεται με μια λιγότερη πύλη XOR και συνέπεια με μία βελτίωση στη καθυστέρηση ίση με $2T_G$ η οποία όμως δεν βελτιώνει τη συνολική καθυστέρηση αφού εκτελείται παράλληλα με την MB κωδικοποίηση και παραγωγή μερικών γινομένων η οποία ολοκληρώνεται σε $8T_G$. Το συνολικό υλικό που χρησιμοποιείται φαίνεται στο παρακάτω πίνακα.

Πίνακας 3.7 *Απαραίτητο υλικό καθώς και επιπτώσεις στην επιφάνεια και την καθυστέρηση.*

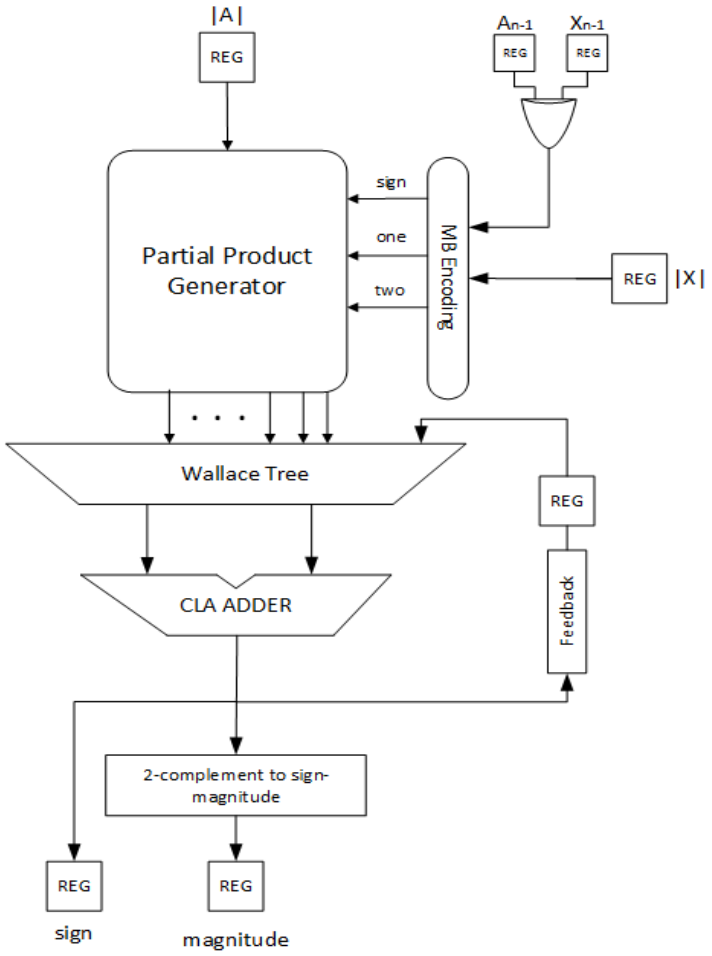
Στοιχείο	**Επιφάνεια**	**Καθυστέρηση**
MB encoding-PPG-Reverse D	$A_{MBPPG}+A_{reverseD}$	$8T_G$
Wallace Tree	A_{TREE}	$Depth(N/2+2) * 4T_G$
CLA-increment	$A_{CLA}+A_{increment}$	$T_{CLA}+T_{increment}$
Registers	$36N*A_G$	$4T_G$

3.6 Υλοποίηση MAC

Σε αυτή τη παράγραφο θα αναλυθούν δύο διαφορετικοί σχεδιασμού κυκλώματος MAC το οποίο υπολογίζει την αριθμητική πράξη $S = \sum A_i * X_i$.

3.6.1 Σχεδίαση με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο – μέτρο

Αυτή η υλοποίηση είναι αρκετά παρόμοια με τη προηγούμενη αλλά αντί για τρεις αριθμούς ως εισόδους, υπάρχουν μόνο δύο των οποίων υπολογίζεται το γινόμενο και προστίθεται στο προηγούμενο αποτέλεσμα. Αυτή η λειτουργία MAC χρησιμοποιείται κυρίως στην ψηφιακή επεξεργασία σημάτων και ιδιαίτερα στην σχεδίαση φίλτρων. Η λειτουργία αυτή έχει πολλές ομοιότητες με την προηγούμενη καθώς το προηγούμενο αποτέλεσμα στο οποίο προστίθενται το γινόμενο των δύο εσόδων παίζει παρόμοιο ρόλο με τη προηγούμενη τρίτη είσοδο D. Στη σχεδίαση με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο η ανατροφοδότηση του αποτελέσματος μπορεί να γίνει απευθείας στη μορφή συμπληρώματος ως προς δύο για να αποφεύγεται η μετατροπή του, κάτι που γινόταν στο προηγούμενο σχέδιο. Η σχεδίαση που εφαρμόστηκε ήταν η ακόλουθη:



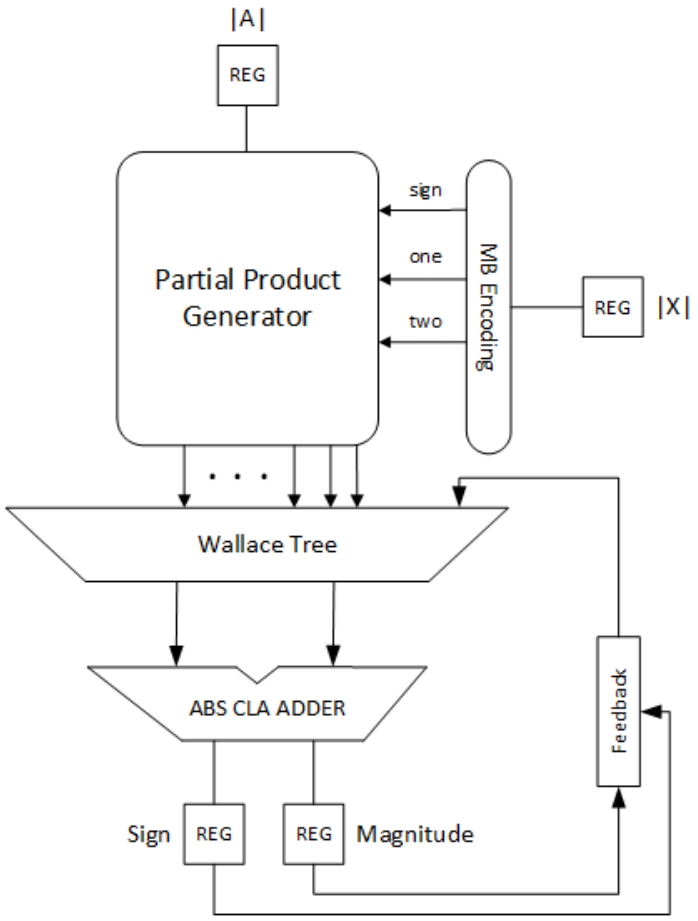
Σχήμα 3.12 Σχεδίαση MAC με χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο.

Στο παραπάνω σχήμα με την ονομασία REG ονομάζονται οι καταχωρητές, οι οποίοι έχουν την ιδιότητα να περνούν την πληροφορία του σήματος εισόδου τους στην έξοδό τους στον θετικό παλμό του ρολογιού, μία μονάδα είναι απαραίτητο να απομονώνει τις εισόδους και τις εξόδους μέσω καταχωρητών προκειμένου τα δεδομένα να εισέρχονται και να εξέρχονται συγχρονισμένα, στο παραπάνω σχήμα μπορεί να παρατηρηθεί ένας καταχωρητής που ανατροφοδοτεί το σήμα του πρώτου αθροιστή δηλαδή το αποτέλεσμα σε μορφή συμπληρώματος ως προς δύο στο δέντρο Wallace σαν ένα επιπλέον μερικό γινόμενο.

Η ανατροφοδότηση μπορεί αν γίνει και από την έξοδο των καταχωρητών του τελευταίου σταδίου όμως τότε το αποτέλεσμα θα επιστρέφει σε μορφή πρόσημου-μέτρου οπότε χρειάζεται η εκ νέου μετατροπή του σε συμπλήρωμα ως προς δύο προκειμένου να εισαχθεί σαν είσοδο στο δέντρο Wallace. Στο κύκλωμα feedback του σχήματος γίνεται ο μηδενισμός (reset) ή ο κορεσμός (saturation) του αποτελέσματος όταν το σήμα overflow γίνει 1, δηλαδή στην περίπτωση υπερχείλισης το αποτέλεσμα μπορεί να αντικατασταθεί είτε με το μέγιστο δυνατό (saturation) είτε να γίνει ένα reset στον MAC και οι υπολογισμοί να αρχίσουν από την αρχή, οι δύο περιπτώσεις είναι παρόμοιες καθώς για τον μηδενισμό μπορεί να εκτελεστεί η λογική πράξη AND μεταξύ των bit του αποτελέσματος και του αντίστροφου overflow, ενώ στη περίπτωση κορεσμού (saturation) εκτελείται η λογική πράξη OR μεταξύ των bit και του overflow.

3.6.2 Σχεδίαση με χρήση αθροιστή απόλυτης τιμής

Σε αυτή τη παράγραφο αναλύεται πως το κύκλωμα μπορεί να μετατραπεί έτσι ώστε να γίνεται η αποθήκευση του αποτελέσματος ώστε το νέο γινόμενο να προστίθεται στον αποθηκευμένο αριθμό. Όπως και προηγουμένους η σχεδίαση είναι αρκετά παρόμοια με εκείνη της εκτέλεσης της πράξης $A * X + D$ μόνο που το ρόλο του αριθμού D τον παίζει το αποτέλεσμα που υπολογίστηκε στον προηγούμενο κύκλο. Αυτό γίνεται μέσω μίας ανατροφοδότησης του τελικού μέτρου και πρόσημου του αποτελέσματος στο δέντρο Wallace. Η σχεδίαση που εφαρμόστηκε φαίνεται στο παρακάτω σχήμα:



Σχήμα 3.14 Σχεδίαση MAC με χρήση αθροιστή απόλυτης τιμής

Λόγω ότι το αποτέλεσμα ανατροφοδοτείται απευθείας από το τελευταίο στάδιο σε μορφή προσήμου-μέτρου δεν χρειάζεται επιπλέον καταχωρητής στην ανάδραση, το κύκλωμα feedback σε αυτή τη περίπτωση εκτός από το μηδενισμό (reset) του αποτελέσματος ή το κορεσμό (saturation) του περιλαμβάνει και την αντιστροφή του προκειμένου να γίνει σωστά η επόμενη πρόσθεση.

Σημειώνεται ότι η προσαρμογή για το reset ή το saturation προηγείται της αντιστροφής του αποτελέσματος προκειμένου η λειτουργία να είναι σωστή, επιπλέον παρατηρείται ότι η λειτουργία του κυκλώματος feedback γίνεται παράλληλα με εκείνη της παραγωγής των μερικών γινομένων οπότε η καθυστέρηση του δεν υπολογίζεται αθροιστικά στη συνολική καθυστέρηση του κυκλώματος.

Ο αθροιστής απόλυτης τιμής είναι ο ίδιος που χρησιμοποιήθηκε και στη προηγούμενη σχεδίαση γι’ αυτό το λόγο στο σχήμα δεν περιλαμβάνεται ο επιπλέον πολυπλέκτης για το σωστό υπολογισμό του τελικού πρόσημου γιατί θεωρείται ενσωματωμένος στον αθροιστή απόλυτης τιμής, επίσης παρόλο που δεν φαίνεται στο σχήμα το operation bit, υπάρχει και ο υπολογισμός του είναι ακριβώς

ανάλογος με την προηγούμενη σχεδίαση με αθροιστή απόλυτης τιμής αν αντικατασταθεί το πρόσημο του αριθμού D με το πρόσημο του προηγούμενο αποτελέσματος.

3.6.3 Θεωρητική ανάλυση επιφάνειας και καθυστέρησης των δύο σχεδιασμών.

Χρήση αθροιστή απόλυτης τιμής

Όπως και η σχεδίαση, έτσι και η θεωρητική ανάλυση είναι αρκετά παρόμοια με εκείνη του προηγούμενου σχεδιασμού γι' αυτό το λόγο θα γίνει επικέντρωση στις διαφορές τους. Η κύρια διαφορά είναι η προσθήκη κυκλώματος για reset ή saturation του αποτελέσματος και έλεγχος για αντιστροφή του για την ορθή εκτέλεση του επόμενου υπολογισμού. Το κύκλωμα feedback προσθέτει επιφάνεια και έχει καθυστέρηση ίση με:

$$A_{feedback} = A_{reset} + A_{reverse} = (2N-1) * A_{AND} + (2N-1) * A_{XOR} + 2 * A_{XOR}$$

$$A_{feedback} = 2N * A_G + 2N * 2A_G + 2 * 2A_G \qquad (3.23)$$

$$T_{feedback} = T_{reverse} = 3 * T_{XOR} = 6 * T_G \qquad (3.24)$$

Στη καθυστέρηση υπολογίστηκε μόνο το $T_{reverse}$ λόγω ότι το ελάχιστο μονοπάτι αφορά μόνο την αντιστροφή. Όπως φαίνεται και στο σχήμα το υποκύκλωμα feedback λειτουργεί παράλληλα με τη MB κωδικοποίηση και παραγωγή μερικών γινομένων οπότε η καθυστέρηση του δεν αθροίζεται στο κύκλωμα αλλά επικρατεί η μεγαλύτερη των δύο. Επίσης σημειώνεται ότι το overflow αντιστοιχεί στο MSB της εξόδου του αθροιστή απόλυτης τιμής οπότε δεν είναι απαραίτητο να χρησιμοποιηθεί υποκύκλωμα για τον υπολογισμό του. Το συνολικό υλικό που χρησιμοποιείται φαίνεται στο παρακάτω πίνακα.

Πίνακας 3.8 *Απαραίτητο υλικό καθώς και επιπτώσεις στην επιφάνεια και τη καθυστέρηση.*

Στοιχείο	Επιφάνεια	Καθυστέρηση
MB encoding-PPG-Feedback	$A_{MBPPG} + A_{Feedback}$	$6T_G$
Wallace Tree	A_{TREE}	$Depth(N/2+2) * 4T_G$
CLA Absolute value	$A_{CLA} + A_{mux}$	$T_{CLA} + T_{mux}$
Registers	$24N * A_G$	$4T_G$

Χρήση Μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο.

Σε αυτή τη περίπτωση χρησιμοποιείται υλικό για τον υπολογισμό του overflow, συγκεκριμένα χρησιμοποιήθηκαν είναι 3 πύλες XOR και μία πύλη AND, οι δύο πύλες XOR χρησιμεύουν για τον υπολογισμό του bit που καθορίζει αν οι δύο αριθμοί που προστίθενται είναι ομόσημοι, η τρίτη πύλη για τον έλεγχο της αντιστροφής του πρόσημου αποτελέσματος και η πύλη AND για το αν αυτές οι δύο συνθήκες συμβαίνουν ταυτόχρονα, για το reset χρησιμοποιούνται μία σειρά από 2N-1 πύλες AND. Οπότε το κύκλωμα feedback σε αυτή τη περίπτωση έχει τα εξής χαρακτηριστικά:

$$A_{feedback} = A_{overflow} + A_{reset} = 6 * A_G + A_G + 2N * A_G \qquad (3.25)$$

$$T_{feedback} = T_{overflow} + T_{reset} = 5 * T_G + T_G = 6 * T_G \qquad (3.26)$$

Παρατηρείται ότι το υποκύκλωμα feedback μπορεί να τοποθετηθεί είτε πριν είτε μετά από τη σειρά καταχωρητών, η επιλογή τοποθέτησής του δεν επηρεάζει τη συνολική καθυστέρηση του κυκλώματος καθώς εκτελείται παράλληλα με άλλα στοιχεία με παρόμοια ή μεγαλύτερη καθυστέρηση. Το συνολικό υλικό που χρησιμοποιείται φαίνεται στο παρακάτω πίνακα.

Πίνακας 3.9 *Απαραίτητο υλικό καθώς και επιπτώσεις στην επιφάνεια και τη καθυστέρηση.*

Στοιχείο	**Επιφάνεια**	**Καθυστέρηση**
MB encoding-PPG-Feedback	$A_{MBPPG}+A_{feedback}$	$8T_G$
Wallace Tree	A_{TREE}	$Depth(N/2+2) * 4T_G$
CLA-increment	$A_{CLA}+A_{increment}$	$T_{CLA}+T_{increment}$
Registers	$36N*A_G$	$4T_G$

3.7 Υλοποίηση MAC με λειτουργία συνεχούς διοχέτευσης

Σε αυτή τη παράγραφο θα αναλυθούν δύο διαφορετικοί τρόποι σχεδιασμού MAC με λειτουργία συνεχούς διοχέτευσης δηλαδή υπολογισμού της αριθμητικής πράξης $S = \sum A_i * X_i$ σε δύο κύκλους ρολογιού.

3.7.1 Σχεδίαση με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο

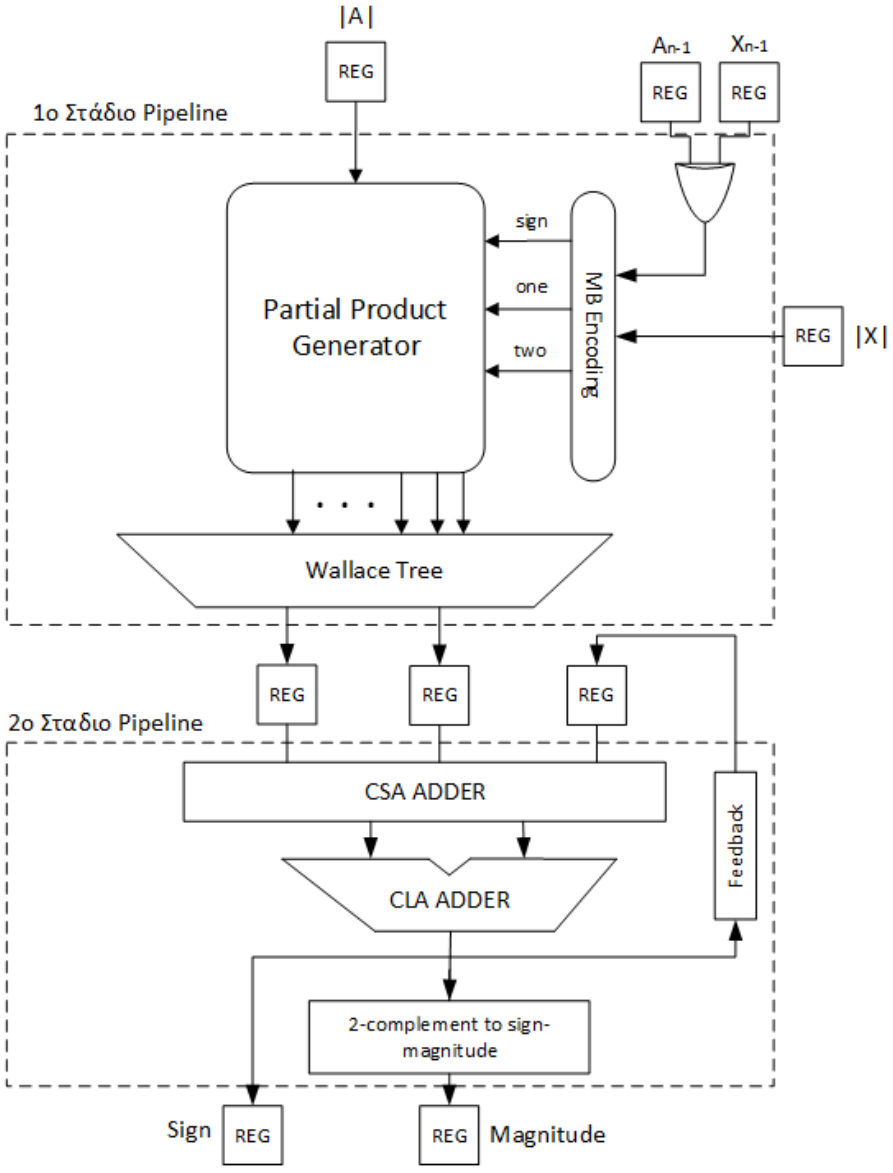
Μέχρι στιγμής έχει αναλυθεί ο τρόπος σχεδίασης του MAC προκειμένου να εκτελεί την αριθμητική πράξη $\sum A_i * X_i$, παρόλο που η προηγούμενη σχεδίαση λειτουργεί και παράγει σωστά αποτελέσματα έχει ένα κύριο μειονέκτημα, αυτό είναι ότι δεν γίνεται να αυξήσουμε περισσότερο τη συχνότητα του ρολογιού εφαρμόζοντας λειτουργία συνεχούς διοχέτευσης του κυκλώματος. Υπενθυμίζεται ότι με τον όρο λειτουργία συνεχούς διοχέτευσης εννοείται το «κόψιμο» του κυκλώματος σε κομμάτια προκειμένου τα κομμάτια αυτά να λειτουργούν παράλληλα ώστε να αυξάνεται ο ρυθμός επεξεργασίας δεδομένων. Η λειτουργία συνεχούς διοχέτευσης, ή αλλιώς λειτουργία pipeline, δεν εφαρμόζεται τόσο απλά σε κυκλώματα τα οποία περιέχουν ανάδραση πληροφορίας και αυτό διότι, αν για παράδειγμα χωριστεί το κύκλωμα σε δύο κομμάτια στο κομμάτι συμπιεστή Wallace και στο κομμάτι άθροισης, το Wallace θα πρέπει να δέχεται το προηγούμενο αποτέλεσμα σε κάθε ένα κύκλο ρολογιού παρότι λόγω pipeline αυτό θα υπολογίζεται σε δύο, οπότε τα δύο μέρη δεν θα λειτουργούν συγχρονισμένα, αυτό μπορεί να ξεπεραστεί με τη χρήση επιπλέον καταχωρητών αλλά δεν αποτελεί τη βέλτιστη λύση καθώς το κύκλωμα επιβαρύνεται περαιτέρω.

Η λύση που προτείνεται είναι η ανάδραση να γίνεται απευθείας στον αθροιστή και όχι στο δεντρικό συμπιεστή Wallace, έτσι το κομμάτι με την ανάδραση θα συνεχίζει να εκτελείται σε έναν μόνο κύκλο ρολογιού με το δέντρο Wallace να επεξεργάζεται μόνο τα μερικά γινόμενα που προκύπτουν από την MB κωδικοποίηση των δύο εισόδων. Λόγω ότι με αυτό το σχεδιασμό ο τελικός αθροιστής πρέπει να αθροίζει τρείς αριθμούς: τα διανύσματα s και c που παράγει το δέντρο Wallace και το προηγούμενο αποτέλεσμα που επιστρέφει ως ανάδραση, χρειάζεται η χρήση ενός αθροιστή με σώσιμο-κρατουμένου (carry-save adder) ο οποίος εφαρμόζει μία επιπλέον συμπίεση δεδομένων προκειμένου να μετατρέψει τα τρία διανύσματα προς άθροιση σε δύο.

Στη λειτουργία συνεχούς διοχέτευσης κυκλωμάτων, η περίοδος του ρολογιού είναι ουσιαστικά το μεγαλύτερο critical path από τα επιμέρους κομμάτια στα οποία το κύκλωμα έχει χωριστεί, έτσι για αυξημένη χρονική απόδοση είναι προτιμότερο το σπάσιμο σε κομμάτια που έχουν παρόμοιο critical path. Στη περίπτωση της λειτουργία συνεχούς διοχέτευσης του MAC όπου ο διαχωρισμός γίνεται σε δύο κομμάτια, το δεντρικό συμπιεστή Wallace και τον αθροιστή, η καθυστέρηση αυξάνεται με διαφορετικό τρόπο σε κάθε κομμάτι με την αύξηση του μήκους της λέξης, συγκεκριμένα η καθυστέρηση του αθροιστή είναι ανάλογη του λογαρίθμου του μήκους της λέξης

ενώ για το δέντρο Wallace η αύξηση δεν είναι γραμμική και τα επίπεδα συμπίεσης εξαρτώνται από το πλήθος των μερικών γινομένων, γενικά κάθε επίπεδο του δέντρου Wallace έχει καθυστέρηση $T_{FA}=4T_G$, μεγαλύτερη από το επίπεδο άθροισης που έχει καθυστέρηση $T_{\#}=2T_G$. Όσο το μήκος της λέξης είναι μικρό αναμένεται το δέντρο Wallace να έχει μικρότερη καθυστέρηση από τον αθροιστή, όσο όμως το μήκος της λέξης αυξάνεται τότε το στάδιο συμπίεσης θα γίνει πιο αργό από το στάδιο άθροισης.

Η σχεδίαση που προτείνεται για την λειτουργία συνεχούς διοχέτευσης του MAC φαίνεται στο παρακάτω σχήμα:



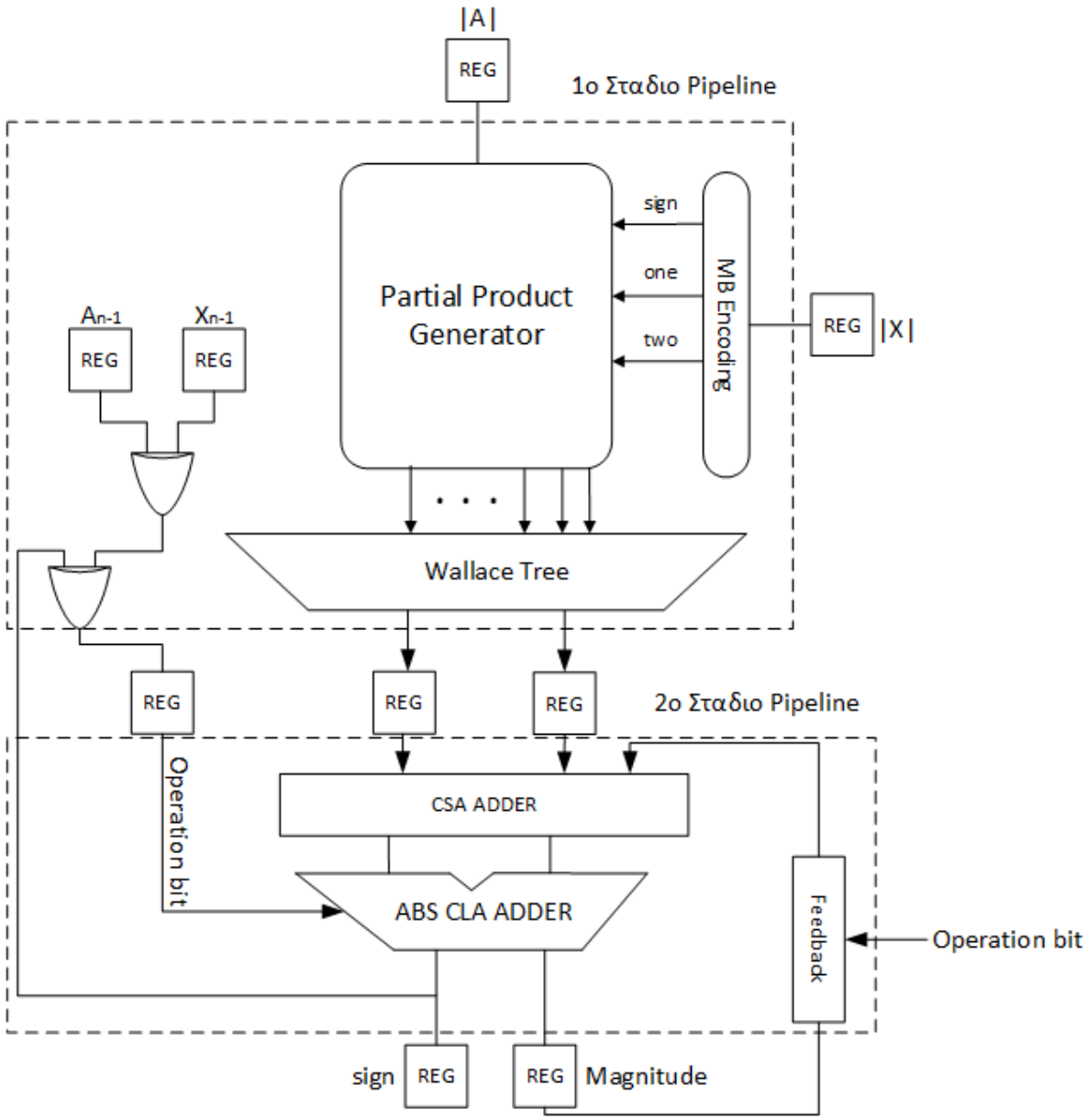
Σχήμα 3.14 Λειτουργία συνεχούς διοχέτευσης MAC με χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο.

Παρατηρείται ότι επιλέχτηκε ξανά να γίνει ανατροφοδότηση από το σημείο που το αποτέλεσμα είναι σε μορφή συμπληρώματος ως προς δύο δηλαδή μετά το στάδιο άθροισης του πρώτου αθροιστή, ώστε να αποφεύγεται η μετατροπή του αποτελέσματος από πρόσημο-μέτρο σε συμπλήρωμα ως προς δύο για την τελική άθροιση, το κύκλωμα feedback χρησιμοποιείται ξανά για reset ή saturation του αποτελέσματος και επειδή λειτουργεί παράλληλα με την μετατροπή συμπληρώματος ως προς δύο σε πρόσημο μέτρο δεν επιφέρει πρόσθετη καθυστέρηση στο συνολικό κύκλωμα.

3.7.2 Σχεδίαση με αθροιστή απόλυτης τιμής

Στη σχεδίαση με χρήση αθροιστή απόλυτης τιμής το αποτέλεσμα ανατροφοδοτείται από το τελευταίο καταχωρητή σε μορφή προσήμου-μέτρου αφού ο αθροιστής που σχεδιάστηκε λαμβάνει εισόδους σε αυτή τη μορφή. Είναι ξανά αναγκαία η χρήση αθροιστή carry-save προκειμένου τα τρία διανύσματα (οι δύο έξοδοι του Wallace και το προηγούμενο αποτέλεσμα) να μετατραπούν σε δύο διανύσματα κατάλληλα προς άθροιση, το σύστημα feedback λειτουργεί για τη διόρθωση του αποτελέσματος σε περίπτωση overflow με reset ή saturation, σε αυτή τη περίπτωση όμως προσθέτει επιπλέον καθυστέρηση στο δεύτερο στάδιο αφού δεν λειτουργεί παράλληλα με κάποιο άλλο υποκύκλωμα, επιπλέον για την σωστή εφαρμογή της λειτουργίας συνεχούς διοχέτευσης χρειάζεται ακόμα ένας καταχωρητής για αποθήκευση των πρόσημων των αριθμών εισόδων του MAC, λόγω ότι είναι αναγκαία η πληροφορία του πρόσημου αποτελέσματος του προηγούμενου κύκλου και του νέου γινομένου. Η πληροφορία αυτή χρειάζεται προκειμένου να σχηματιστεί σωστά το operation bit του αθροιστή απόλυτης τιμής καθώς και η αντιστροφή του αποτελέσματος σε περίπτωση αφαίρεσης, υπενθυμίζεται ότι πάντα υπολογίζεται το θετικό γινόμενο των δύο εισόδων και αφαιρείται το προηγούμενο αποτέλεσμα αν οι δύο αριθμοί έχουν διαφορετικό πρόσημο, διαφορετικά αθροίζονται.

Στη σχεδίαση που παρουσιάζεται στη συνέχεια, το operation bit είναι αυτό που ευθύνεται για την λειτουργία της μονάδας ABS CLA ADDER ως αθροιστή ή ως αφαιρέτη, ενώ είναι υπεύθυνο και για την αντιστροφή του αποτελέσματος στο σύστημα feedback.



Σχήμα 3.15 Λειτουργία συνεχούς διοχέτευσης MAC με χρήση αθροιστή απόλυτης τιμής.

Παρατηρείται ότι ο υπολογισμός του σήματος που ελέγχει τον αθροιστή μπορεί να παραχθεί και με διαφορετικό τρόπο, αν η ανάδραση του πρόσημου γίνει από την έξοδο του καταχωρητή και παράλληλα ο καταχωρητής του σήματος ελέγχου μεταφερθεί πριν την XOR πύλη. Αυτοί οι δύο τρόποι είναι ισοδύναμοι και παρουσιάζουν ελάχιστη διαφορά ως προς τη συνολική καθυστέρηση του κυκλώματος.

3.7.3 Θεωρητική ανάλυση επιφάνειας και καθυστέρησης των δύο σχεδιασμών.

Χρήση αθροιστή απόλυτης τιμής.

Σε αυτόν τον σχεδιασμό, το δέντρο Wallace έχει ένα λιγότερο μερικό γινόμενο στην είσοδό του καθώς δεν δέχεται σήμα ανατροφοδότησης, επιπλέον προστίθεται ένας CSA αθροιστής ο οποίος έχει καθυστέρηση ίση με έναν Full Adder και επιφάνεια ίση με 2N Full adders.

$$A_{CSA} = 2N * 7A_g \qquad (3.26)$$

$$T_{CSA} = 4 * T_G \qquad (3.27)$$

$$T_{Tree} = Depth\left(\frac{N}{2} + 1\right) * 4T_G \qquad (3.28)$$

Από άποψη καταχωρητών χρησιμοποιούνται 4N επιπλέον για τα σήματα s και c του δέντρου Wallace και το διαχωρισμό της λειτουργίας του κυκλώματος σε δύο στάδια, καθώς και ένας για τη δημιουργία του operation bit στο σωστό κύκλο για τη λειτουργία του αθροιστή απόλυτης τιμής, το σύστημα Feedback δεν εκτελείται παράλληλα με κάποιο άλλο σύστημα σε αυτή τη σχεδίαση οπότε η καθυστέρηση του προστίθεται στη συνολική καθυστέρηση ολόκληρου του κυκλώματος. Το συνολικό υλικό που χρησιμοποιείται για τη σχεδίαση των δύο σταδίων λειτουργίας συνεχούς διοχέτευσης φαίνεται στον ακόλουθο πίνακα:

Πίνακας 3.10 *Απαραίτητο υλικό καθώς και επιπτώσεις στην επιφάνεια και τη καθυστέρηση.*

Στοιχείο	**Επιφάνεια**	**Καθυστέρηση**
Πρώτο Στάδιο λειτουργίας συνεχούς διοχέτευσης		
MB encoding-PPG	A_{MBPPG}	$5T_G$
Wallace Tree	A_{TREE}	$Depth(N/2+1) * 4T_G$
Δεύτερο Στάδιο λειτουργίας συνεχούς διοχέτευσης		
CSA Adder	A_{CSA}	T_{CSA}
CLA Absolute value	$A_{CLA}+A_{mux}$	$T_{CLA}+T_{mux}$
Feedback	$A_{Feedback}$	$T_{Feedback}$
Operation bit	$4A_G$	$2T_G$
Σύνολο καταχωρητών ολόκληρου του κυκλώματος		
Registers	$(8N+1)A_{Reg}$	$6T_G$

Χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο.

Η θεωρητική ανάλυση αυτής της σχεδίασης είναι πλήρως ανάλογη με τη προηγούμενη, ο CSA αθροιστής και το δέντρο Wallace έχουν τα ίδια χαρακτηριστικά με πριν, οι διαφορές βρίσκονται στη MB κωδικοποίηση καθώς και στη χρήση μετατροπέα αντί αθροιστή απόλυτης τιμής, οι αναλύσεις σχετικά με το σύστημα ανατροφοδότησης και το σύστημα μετατροπής σε πρόσημο μέτρο είναι ίδιες με το συνδυαστικό κύκλωμα. Το συνολικό υλικό που χρησιμοποιείται για τη σχεδίαση των δύο σταδίων λειτουργίας συνεχούς διοχέτευσης φαίνεται στον ακόλουθο πίνακα:

Πίνακας 3.11 *Απαραίτητο υλικό καθώς και επιπτώσεις στην επιφάνεια και τη καθυστέρηση.*

Στοιχείο	**Επιφάνεια**	**Καθυστέρηση**
Πρώτο Στάδιο λειτουργίας συνεχούς διοχέτευσης		
MB encoding-PPG	A_{MBPPG}	$8T_G$
Wallace Tree	A_{TREE}	$Depth(N/2+1) * 4T_G$
Δεύτερο Στάδιο λειτουργίας συνεχούς διοχέτευσης		
CSA Adder	A_{CSA}	T_{CSA}
CLA-increment-feedback	$A_{CLA}+A_{increment}+A_{Feedback}$	$T_{CLA}+T_{increment}$
Σύνολο καταχωρητών ολόκληρου του κυκλώματος		
Registers	$(10N)A_{Reg}$	$6T_G$

3.8 Υλοποίηση double MAD.

Σε αυτή τη παράγραφο αναλύονται δύο διαφορετικοί τρόποι σχεδιασμού κυκλώματος double MAD δηλαδή κυκλώματος που υπολογίζει την αριθμητική πράξη $S = A * X + B * Y + D$.

3.8.1 Σχεδίαση με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο.

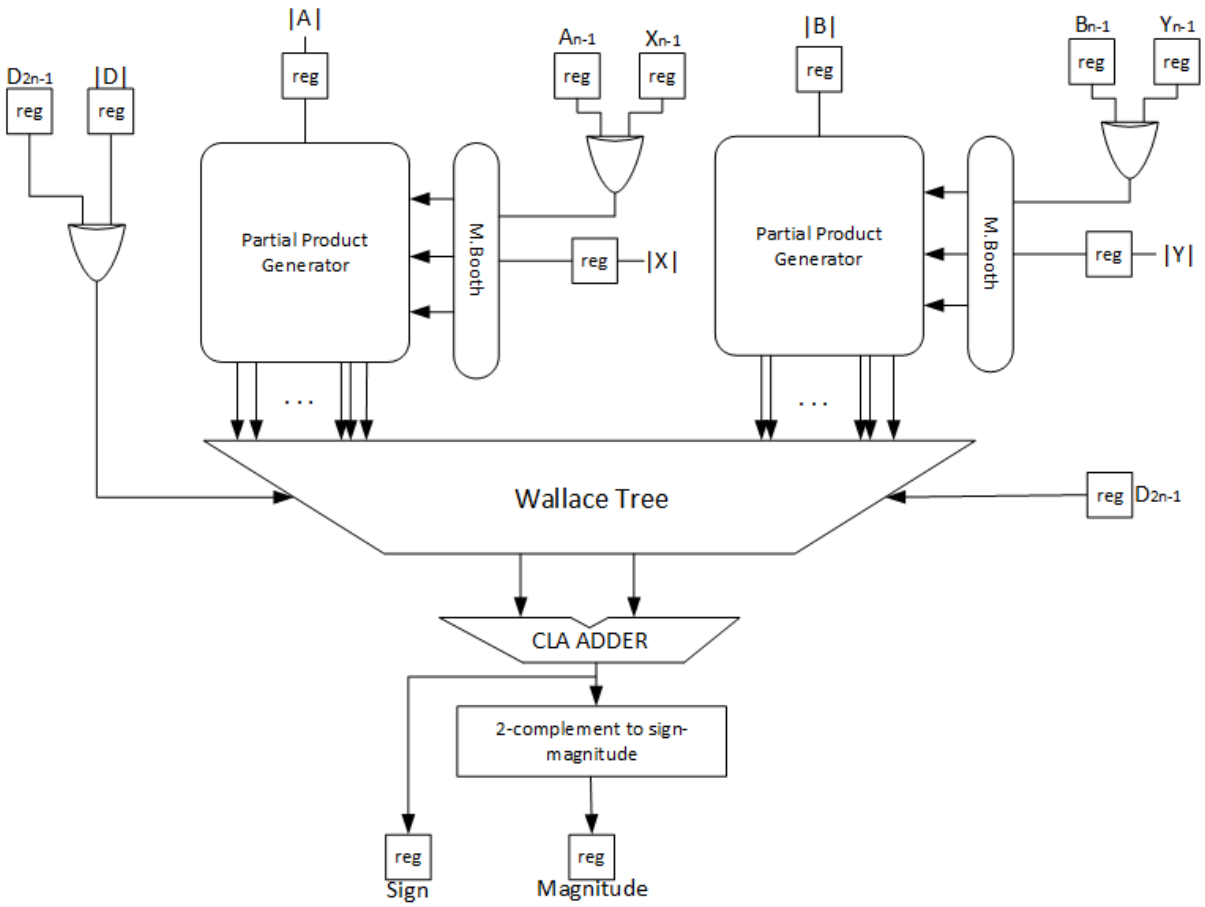
Μέχρι στιγμής αναλύθηκαν τρόποι σχεδιασμού κυκλωμάτων που υπολογίζουν ένα γινόμενο δύο αριθμών, και το αθροίζουν σε έναν άλλον αριθμό, στις επόμενες παραγράφους θα μελετηθούν οι τρόποι σχεδιασμού τύπων κυκλωμάτων που υπολογίζουν δύο γινόμενα, δηλαδή δέχονται τέσσερεις αριθμούς ως είσοδο, παρόλο που η λογική σχεδίασης δεν αλλάζει σε μεγάλο βαθμό σε σχέση με τα προηγούμενα σχέδια, το κομμάτι του δεντρικού συμπιεστή Wallace είναι αρκετά πιο επιβαρυντικό σε επιφάνεια και καθυστέρηση αφού καλείται να συμπιέσει το διπλάσιο αριθμό μερικών γινομένων.

Η σχεδίαση που προτείνεται με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο είναι αρκετά παρόμοια με τη προηγούμενη μορφή σχεδίασης: αντιστρέφονται τα σήματα των πρόσημων της MB κωδικοποίησης αν το γινόμενο που προκύπτει είναι αρνητικό, το σήμα D το οποίο δεν αποτελεί γινόμενο δύο αριθμών αντιστρέφεται, και στο Wallace δέντρο προστίθεται μία μονάδα προκειμένου να ολοκληρωθεί η μετατροπή του από πρόσημο-μέτρο σε συμπλήρωμα ως προς δύο σε περίπτωση αρνητικής τιμής.

Αν το αποτέλεσμα της τελικής πρόσθεσης των σημάτων s και c του Wallace είναι αρνητικό τότε πρέπει να γίνει η μετατροπή του σε μορφή πρόσημου-μέτρου κατά τα γνωστά με έναν μετατροπέα όπως και στη προηγούμενη σχεδίαση.

Μία δυσκολία η οποία εμφανίζεται σε τέτοιου είδους κυκλώματα που δεν υπάρχει στα προηγούμενα σχέδια είναι η εμφάνιση overflow σε περίπτωση αφαίρεσης. Αυτό συμβαίνει διότι υπάρχει η πιθανότητα τα δύο γινόμενα τα οποία αθροίζονται να είναι ετερόσημα και το άθροισμά τους να έχει το ίδιο πρόσημο με τον αριθμό D. Σε αυτή τη περίπτωση υπάρχει πιθανότητα overflow και είναι αναγκαίος ο εντοπισμός του. Παρόλο που η λογική εντοπισμού είναι ίδια με εκείνη που θα χρησιμοποιηθεί στη συνέχεια με χρήση αθροιστή απόλυτης τιμής, η πλήρης εξήγηση αφήνεται για την επόμενη παράγραφο όπου θα μπορεί να γίνει πιο εύκολα κατανοητή από τον αναγνώστη.

Η σχεδίαση που προτείνεται παρουσιάζεται στο επόμενο σχήμα:



Σχήμα 3.16 Σχεδίαση MAC υπολογισμού 2 γινομένων και άθροισή τους στον αριθμό D με χρήση μετατροπέα συμπληρώματος ως προς δύο σε πρόσημο μέτρο.

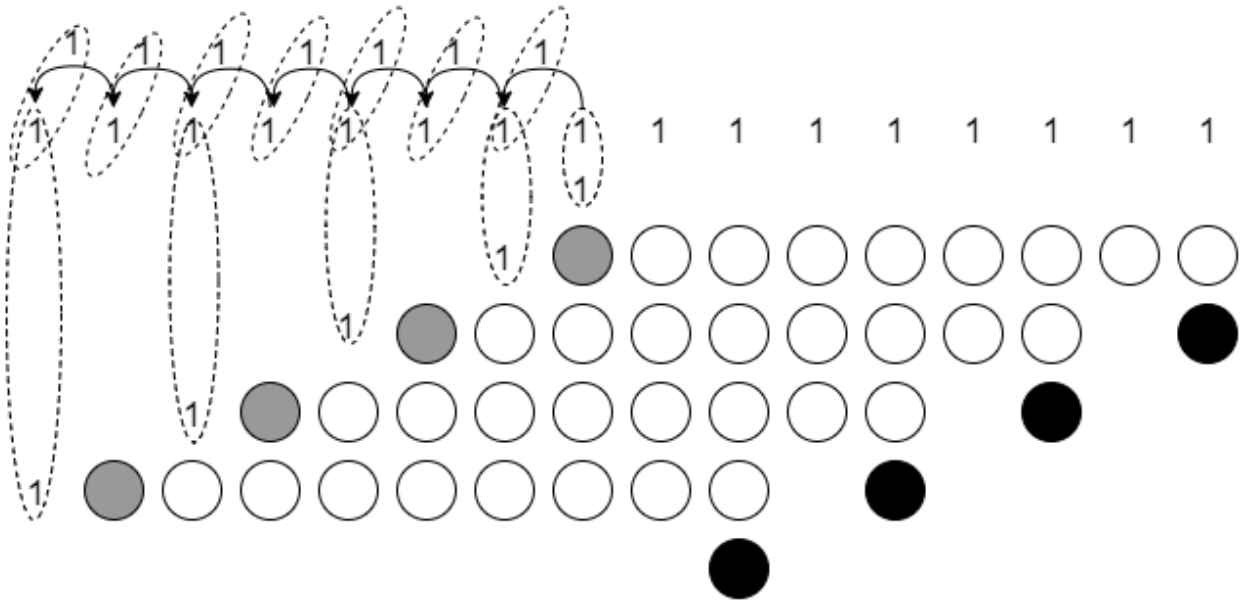
Φαίνεται η πλήρης αντιστοιχία της σχεδίασης με τη προηγούμενη σχεδίαση για υπολογισμό της αριθμητικής πράξης $A * X + D$ όπου πρέπει απλά να υπολογιστούν τα αντίθετα σήματα προσήμων όταν τα γινόμενα είναι αρνητικά και το συμπλήρωμα ως προς δύο όταν το πρόσημο του αριθμού D είναι αρνητικό. Αυτό σημειώνεται για να παρατηρηθεί το κύριο πλεονέκτημα αυτής της σχεδίασης, παρόλο που προϋποθέτει τη χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο η λογική σχεδίασης είναι αρκετά απλή και μπορεί ακριβώς όπως με τον ίδιο τρόπο να εφαρμοστεί για τον υπολογισμό περισσότερων γινομένων.

3.8.2 Σχεδίαση με χρήση αθροιστή απόλυτης τιμής.

Η σχεδίαση γίνεται πιο περίπλοκη ώστε να γίνει δυνατή η χρήση αθροιστή απόλυτης τιμής, το κύριο μειονέκτημα της είναι ότι δεν υπάρχει αποδοτικός τρόπος υπολογισμού ενός γινομένου με ανεστραμμένα bit δηλαδή ο σχηματισμός του γινομένου $\overline{A * X}$. Ο υπολογισμός αυτού του όρου δεν μπορεί να αποφευχθεί με κάποιο τρόπο όπως έγινε στη προηγούμενη σχεδίαση υπολογισμού της αριθμητικής πράξης $A * X + D$. Ο τρόπος ο οποίος προτείνεται είναι σε περίπτωση αρνητικών γινομένων να προστίθεται ο αριθμός -1 με μορφή ενός επιπλέον μερικού γινομένου, λόγω της γνωστής ιδιότητας:

$$-A * X = \overline{A * X} + 1 \leftrightarrow \overline{A * X} = -A * X - 1$$

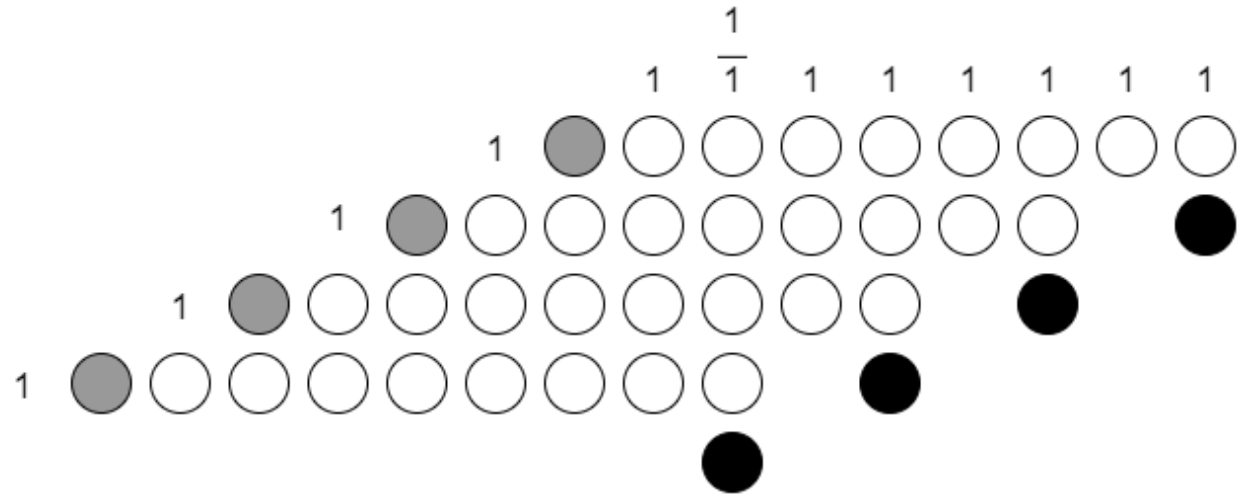
Οπότε υπολογίζεται το ανεστραμμένο γινόμενο αν από τον αντίθετο γινόμενο προστεθεί το -1. Επειδή τα μερικά γινόμενα που προστίθενται έχουν μήκος 2n ο αριθμός -1 θα αποτελείται από 2n άσους, όμως με κατάλληλη επεξεργασία τους μαζί με τους όρους του διορθωτικού γινομένου γίνεται εμφανές ότι ουσιαστικά χρειάζεται η πρόσθεση μόνο n - 2 άσους, Αυτή η επεξεργασία των άσων από τον αριθμό -1 μαζί με τους όρους του διορθωτικού γινομένου παρουσιάζεται σχηματικά παρακάτω:



Σχήμα 3.17 Επεξεργασία μερικών γινομένων για αποδοτική πρόσθεση του -1 στο τελικό αποτέλεσμα.

Στο σχήμα όπως και στο παρόμοιο που παρουσιάστηκε στο προηγούμενο κεφάλαιο οι άσπροι κύκλοι συμβολίζουν τους όρους των μερικών γινομένων, οι μαύροι κύκλοι τις επιπλέον μονάδες που προστίθενται προκειμένου να γίνεται σωστή αποτύπωση των αρνητικών μερικών γινομένων και οι γκρι κύκλοι τα ανεστραμμένα bit των μερικών γινομένων, οι επιπλέον άσοι συμβολίζουν τον αριθμό -1. Όπως φαίνεται και στο σχήμα οι άσοι του όρου -1 μπορούν να ομαδοποιηθούν μαζί με τα ψηφία διορθωτικού όρου, τα βελάκια πάνω από τις μονάδες συμβολίζουν τα κρατούμενα

εξόδου της ομαδοποίησης τα οποία όπως φαίνεται είναι παντού 1 αφού πάντα ομαδοποιούνται τουλάχιστον δύο άσοι. Η ομαδοποίηση δύο μονάδων έχει αποτέλεσμα μηδέν ενώ η ομαδοποίηση τριών μονάδων έχει αποτέλεσμα 1 έτσι η τελική μορφή του πίνακα μερικών γινομένων έχει την ακόλουθη μορφή:



Σχήμα 3.18 Τελική μορφή μερικών γινομένων με άθροιση του -1

Όπως φαίνεται έγινε δυνατή η μείωση των άσων απαραίτητων για την αφαίρεση μίας μονάδας περίπου στο μισό, παρόλα αυτά όμως αυτοί οι άσοι σημαίνουν επιβάρυνση κυρίως σε ενέργεια του κυκλώματος ιδιαίτερα όσο το μήκος των εισόδων αυξάνεται άρα θα αυξάνονται και οι μονάδες οι οποίες πρέπει να προστεθούν, κάτι που δεν εμφανίστηκε στη προηγούμενη σχεδίαση. Όταν δεν προστίθεται το -1 στο τελικό αποτέλεσμα προφανώς η μορφή των μερικών γινομένων είναι ίδια με την προηγούμενη σχεδίαση, ενώ το διορθωτικό γινόμενο προσαρμόστηκε κατάλληλα ώστε να έχει μορφή παρόμοια με πριν.

Για τη σωστή λειτουργία του αθροιστή απόλυτης τιμής πρέπει να γίνει διάκριση σε όλους τους δυνατούς συνδυασμούς των πρόσημων των αριθμών $A * X$, $B * Y$ και D, ο πίνακας όλων των περιπτώσεων καθώς και η λειτουργία του κυκλώματος σε κάθε περίπτωση παρουσιάζονται στον επόμενο πίνακα:

Πίνακας 3.12 *Λειτουργία κυκλώματος ανάλογα με τα πρόσημα των αριθμών εισόδου.*

Πρόσημο $A*X$	**Πρόσημο $B*Y$**	**Πρόσημο D**	**Λειτουργία κυκλώματος**
Θετικό	Θετικό	Θετικό	Καμία αντιστροφή, απλή πρόσθεση των μέτρων των αριθμών.
Θετικό	Θετικό	Αρνητικό	Υπολογισμός $\overline{\mathrm{D}}$ με αντιστροφή των bit του D.
Θετικό	Αρνητικό	Θετικό	Υπολογισμός του $\overline{\mathrm{B}*\mathrm{Y}}$ με αντιστροφή του σήματος προσήμου του Modified Booth και άθροιση του -1.
Θετικό	Αρνητικό	Αρνητικό	Υπολογισμός του $-B*Y$ και $\overline{D}$ με αντιστροφή του σήματος προσήμου του Modified Booth και αντιστροφή του D.
Αρνητικό	Θετικό	Θετικό	Υπολογισμός του $\overline{A*X}$ με αντιστροφή του σήματος προσήμου του Modified Booth και άθροιση του -1.
Αρνητικό	Θετικό	Αρνητικό	Υπολογισμός του $-A*X$ και $\overline{D}$ με αντιστροφή του σήματος προσήμου του Modified Booth και αντιστροφή του D.
Αρνητικό	Αρνητικό	Θετικό	Υπολογισμός $\overline{\mathrm{D}}$ με αντιστροφή των bit του D.
Αρνητικό	Αρνητικό	Αρνητικό	Καμία αντιστροφή, απλή πρόσθεση των μέτρων των αριθμών.

Γενικά ο πίνακας παρουσιάζει τις εξής περιπτώσεις:

- $A*X$ και $B*Y$ και D έχουν ίδιο πρόσημο: τότε δεν αντιστρέφεται κανένας αριθμός αθροίζονται τα μέτρα και το πρόσημο παραμένει ίδιο.
- $A*X$ και $B*Y$ ομόσημοι και D έχει διαφορετικό πρόσημο από αυτούς: τότε αντιστρέφεται ο αριθμός D πριν εισέρθει στο Wallace.
- $A*X$ και $B*Y$ ετερόσημοι και D αρνητικός: τότε αντιστρέφεται το σήμα πρόσημου Modified Booth του αρνητικού γινομένου και επιπλέον αντιστρέφεται ο αριθμός D.
- $A*X$ και $B*Y$ ετερόσημοι D θετικός: τότε αντιστρέφεται το σήμα προσήμου του αρνητικού γινομένου και προστίθεται -1 στο Wallace.

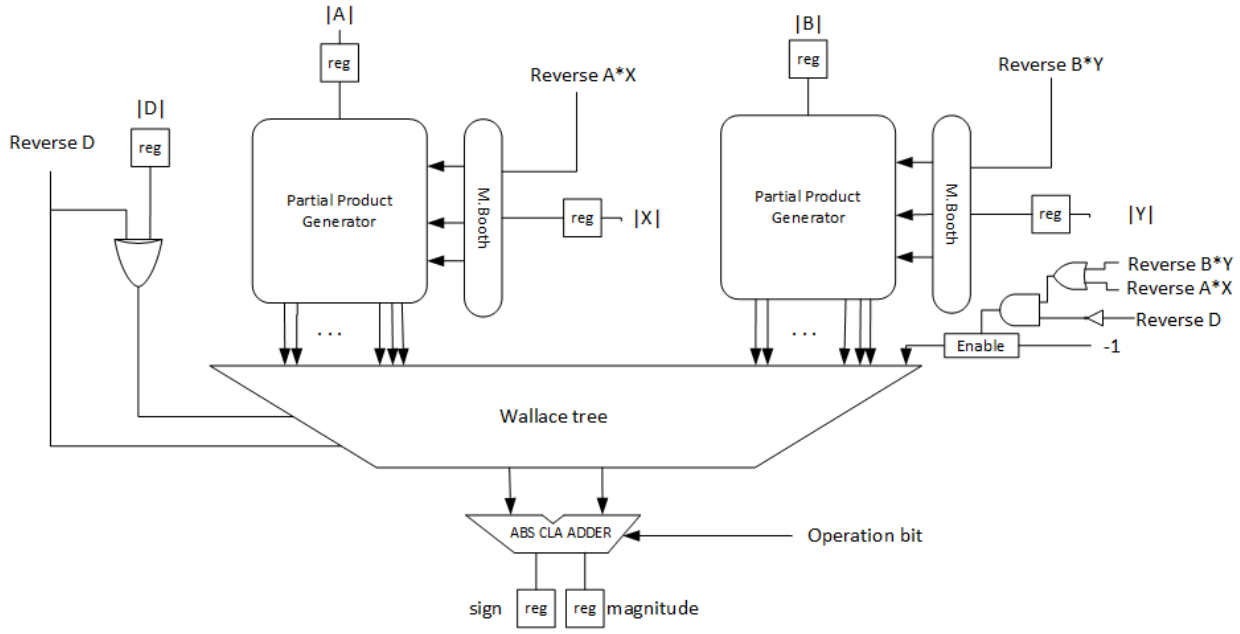
Για ολοκλήρωση της παρουσίασης του κυκλώματος μένει να εξηγηθεί ο τρόπος εντοπισμού της υπερχείλισης, σε αντίθεση με τη προηγούμενη σχεδίαση υπολογισμού ενός μόνο γινομένου, στη πρόσθεση τριών αριθμών υπάρχει η δυνατότητα υπερχείλισης και στη περίπτωση που τρείς αριθμοί έχουν διαφορετικό πρόσημο, όμως επειδή ο αριθμός D έχει μήκος 2n bit ενώ τα γινόμενα $A*X$ και $B*Y$ έχουν μήκος 2n-1 bit, παρατηρείται ότι υπερχείλιση από αφαίρεση μπορεί να συμβεί μόνο στη περίπτωση που οι αριθμοί $A*X$ και $B*Y$ είναι ετερόσημοι, αυτό συμβαίνει γιατί όπως έχει εξηγηθεί το άθροισμα δύο αριθμών μεγέθους 2n-1 χρειάζεται 2n bit για να αναπαρασταθεί, μέγεθος το οποίο ο καταχωρητής εξόδου έχει τη δυνατότητα να αναπαραστήσει,

οπότε στη περίπτωση που $A * X$ και $B * Y$ είναι ομόσημοι αριθμοί και το D έχει διαφορετικό πρόσημο από αυτούς η υπερχείλιση είναι αδύνατη.

Παρόμοια συλλογιστική πορεία ακολουθείται για την τελική εύρεση της συνθήκης overflow, λόγω περιορισμένου μεγέθους των γινομένων να είναι ένα bit μικρότερο από το καταχωρητή εξόδου του αποτελέσματος. Στην περίπτωση ετερόσημων γινομένων, το αποτέλεσμα της αφαίρεσης τους μπορεί να αναπαρασταθεί σε 2n-1 bit. Όταν το MSB του μέτρου του D είναι 0 τότε στην άθροιση των δύο γινομένων είναι αδύνατον να δημιουργηθεί overflow αφού η πρόσθεση δύο αριθμών μεγέθους 2n-1 είναι δυνατόν να αναπαρασταθεί στον καταχωρητή εξόδου. Αντίστοιχα όταν το MSB του μέτρου του D είναι 1 τότε, και μόνο τότε, η πρόσθεση ετερόσημων γινομένων μπορεί να προκαλέσει overflow, το overflow μπορεί να παρατηρηθεί ως μία αντιστροφή στο πρόσημο του αποτελέσματος λόγω περιορισμένου μεγέθους εξόδου. Η αντιστροφή αυτή του αποτελέσματος είναι αδύνατον να μην οφείλεται στο overflow αφού το MSB του αριθμού D ίσο με 1 δηλώνει ότι το μέτρο του D είναι σίγουρα μεγαλύτερο από το άθροισμα των ετερόσημων γινομένων οπότε το πρόσημο πρέπει να παραμένει αμετάβλητο.

Στην περίπτωση που και οι τρείς αριθμοί είναι έχουν το ίδιο πρόσημο το overflow μπορεί να εντοπιστεί κατά τα γνωστά με τον έλεγχο του MSB του αποτελέσματος του αθροιστή απόλυτης τιμής το οποίο ταυτίζεται με τη τιμή του overflow.

Το τελικό σχήμα της σχεδίασης για τον υπολογισμό της αριθμητικής πράξης: $A * X + B * Y + D$ με χρήση απόλυτης τιμής φαίνεται στο παρακάτω σχήμα:



Σχήμα 3.19 Σχεδίαση MAC υπολογισμού 2 γινομένων και άθροισή τους στον αριθμό D με χρήση αθροιστή απόλυτης τιμής.

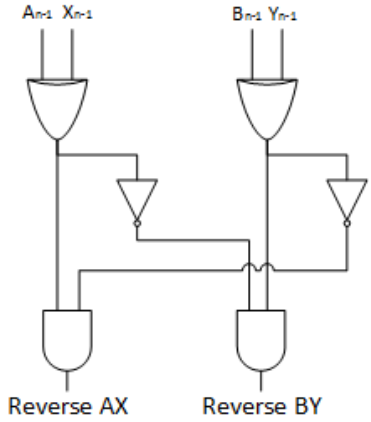
Για να γίνει ευκολότερα κατανοητό το σχήμα έχουν παραληφθεί τα αναλυτικά κυκλώματα υπολογισμού των σημάτων reverse $A * X$, reverse $B * Y$ και reverse D τα οποία συμβολίζουν τα σήματα αντιστροφής των αριθμών $A * X$, $B * Y$ και D αντίστοιχα και σχηματίζονται σύμφωνα με το πίνακα 3.12. Το -1 προστίθεται στο δέντρο Wallace ως ένα επιπλέον μερικό γινόμενο όταν τουλάχιστον ένα από τα σήματα reverse $A * X$ και reverse $B * Y$ είναι 1 και το σήμα reverse D είναι 0. Εναλλακτικά είναι δυνατό να προστεθεί το -1 μόνο στην περίπτωση που ένα από τα σήματα reverse $A * X$ ή reverse $B * Y$ είναι μονάδα και να προστίθεται ένας επιπλέον άσος στην περίπτωση που το σήμα reverse D είναι 1, με τον τρόπο που προτείνεται σε αυτή την εργασία ουσιαστικά προϋπολογίζεται η άθροιση 1-1=0.

Οι υπολογισμοί όλων των σημάτων αντιστροφής προσθέτουν μία επιπλέον καθυστέρηση αφού ο υπολογισμός τους χρειάζεται για την λειτουργία της γεννήτριας παραγωγής μερικών γινομένων, κάτι που όπως θα παρατηρηθεί επιβαρύνει κυρίως την λειτουργία συνεχούς διοχέτευσης του κυκλώματος αφού τότε το κομμάτι δεντρικής συμπίεσης Wallace καθορίζει την περίοδο του ρολογιού για τα περισσότερα μήκη λέξης.

3.8.3 Θεωρητική ανάλυση επιφάνειας και καθυστέρησης

Χρήση Αθροιστή απόλυτης τιμής

Η μεγαλύτερη διαφορά σε ότι αφορά τη σχεδίαση double MAD με χρήση αθροιστή απόλυτης τιμής βρίσκεται στη δημιουργία των σημάτων ελέγχου για τον έλεγχο της MB κωδικοποίησης. Για τα σήματα Reverse AX και Reverse BY απαιτείται επιφάνεια και καθυστέρηση ίση με 1 πύλη XOR και μία πύλη AND για το κάθε σήμα όπως φαίνεται και στο σχήμα:

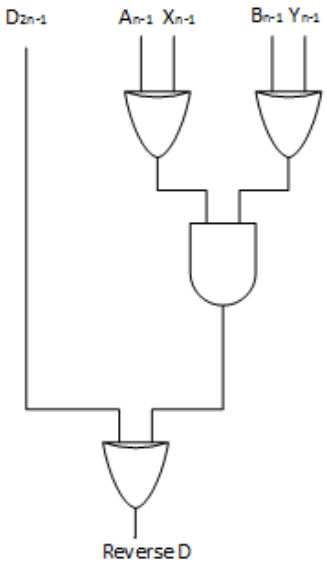


$$A_{revsignals} = 2 * 3A_G \qquad (3.29)$$

$$T_{revsignals} = 3T_G \qquad (3.30)$$

Σχήμα 3.20 Κύκλωμα υπολογισμού σημάτων αντιστροφής των γινομένων με την αντίστοιχη επιφάνεια και καθυστέρηση.

Για την παραγωγή του σήματος αντιστροφής του σήματος D και την αντιστροφή του αριθμού εισόδου D γίνεται με χρήση μίας πύλης AND μεταξύ των πρόσημων των δύο γινομένων και μία πύλη XOR μεταξύ της εξόδου της πύλης AND και του πρόσημου του αριθμού D όπως φαίνεται στο σχήμα:



$$A_{reverseD} = 3A_G + (2N-1) * 2A_G \qquad (3.31)$$

$$T_{reverseD} = (2+1+2+2) * T_G = 7T_G \qquad (3.32)$$

Σχήμα 3.21 Κύκλωμα υπολογισμού σήματος αντιστροφής του αριθμού D με την αντίστοιχη επιφάνεια και καθυστέρηση

Η επιπλέον καθυστέρηση των MB κωδικοποιήσεων είναι ίση με την καθυστέρηση της δημιουργίας των σημάτων αντιστροφής και της κωδικοποίησης modified με μία αντιστροφή στα πρόσημα όταν είναι απαραίτητο. Συνολικά υπάρχουν δύο κωδικοποιήσεις για τα δύο γινόμενα οπότε η επιφάνεια και η καθυστέρηση είναι:

$$A_{MBPPG} = 2 * \left[5 * \left(\frac{N}{2} - 1\right) + 3 + N + 5 * N * \frac{N}{2}\right] * A_G \qquad (3.33)$$

$$T_{MBPPG} = 3T_G + 6T_G = 9T_G > T_{reverseD} \qquad (3.34)$$

Ο έλεγχος προσθήκης του -1 και η εισαγωγή του στο δέντρο Wallace είναι ένα υποκύκλωμα με επιφάνεια και καθυστέρηση:

$$A_{enable-1} = 3 + (N-2) * A_G \qquad (3.35)$$

$$T_{enable-1} = 7T_G \leq T_{MBPPG} \qquad (3.36)$$

Ο δεντρικός συμπιεστής Wallace έχει σαν είσοδο το διπλάσιο αριθμό μερικών γινομένων αφού υπολογίζονται δύο πολλαπλασιασμοί άρα

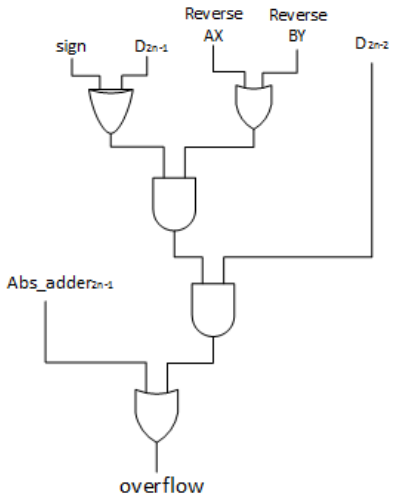
$$T_{tree} = Depth(N+4) * 4T_G \qquad (3.37)$$

Για το σχηματισμό του operation bit χρησιμοποιούνται 3 πύλες XOR οι οποίες εκτελούνται παράλληλα και 3 πύλες OR άρα συνολική επιφάνεια και καθυστέρηση:

$$A_{operation\ bit} = 6A_G + 3A_G = 9A_G \qquad (3.38)$$

$$T_{operation\ bit} = T_{XOR} + 2T_{OR} = 4T_G \qquad (3.39)$$

Ο υπολογισμός του overflow σε αυτόν τον σχεδιασμό προσθέτει και αυτός κάποια επιπλέον καθυστέρηση αφού δεν ταυτίζεται πλέον με το MSB του μέτρου εξόδου του αθροιστή απόλυτης τιμής λόγω ότι υπάρχει overflow και στην περίπτωση ετερόσημων αριθμών. Για υπολογισμό του κυκλώματος overflow χρειάζονται 2 πύλες OR, 2 πύλες AND και μία πύλη XOR αφού το overflow εμφανίζεται όταν ικανοποιείται η συνθήκη: Το MSB του αθροιστή απόλυτης τιμής είναι 1 Ή υπάρχει αλλαγή προσήμου μεταξύ του αποτελέσματος και του αριθμού D ΚΑΙ το MSB του μέτρου του αριθμού D είναι 1 ΚΑΙ τα δύο γινόμενα εισόδου είναι ετερόσημα. Όσον αφορά την καθυστέρηση οι πύλες XOR και OR και AND στο αρχικό επίπεδο δεν καθυστερούν επιπλέον το κύκλωμα αφού μπορούν να υπολογιστούν στην αρχή της λειτουργίας. Το κύκλωμα για εντοπισμό της υπερχείλισης φαίνεται στο παρακάτω σχήμα:



$$A_{overflow} = A_{XOR} + 2A_{AND} + 2A_{OR} = 6A_G \qquad (3.40)$$

$$T_{overflow} = 2T_G \qquad (3.41)$$

Σχήμα 3.22 Κύκλωμα υπολογισμού υπερχείλισης και η αντίστοιχη επιφάνεια και καθυστέρηση.

Η ανάλυση για τον αθροιστή απόλυτης τιμής παραμένει ίδια, όπως με τους προηγούμενους σχεδιασμούς. Το συνολικό υλικό που χρησιμοποιείται φαίνεται στο παρακάτω πίνακα.

Πίνακας 3.13 *Απαραίτητο υλικό καθώς και επιπτώσεις στην επιφάνεια και τη καθυστέρηση.*

Στοιχείο	Επιφάνεια	Καθυστέρηση
MB encoding-PPG-operation bit-reverse d - enable	$A_{MBPPG} + A_{operation\ bit} + A_{reverseD} + A_{enable}$	$9 * T_G$
Wallace Tree	A_{TREE}	$Depth(N+4) * 4T_G$
CLA Absolute value-overflow	$A_{CLA} + A_{mux} + A_{overflow}$	$T_{CLA} + T_{mux} + T_{overflow}$
Registers	$6N * 7A_G$	$4 * T_G$

Χρήση Μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο

Σε αυτή τη σχεδίαση για τη MB κωδικοποίηση και τη γεννήτρια παραγωγής μερικών γινομένων χρησιμοποιείται διπλάσιο υλικό αλλά η καθυστέρηση είναι ίδια με πριν λόγω ότι οι δύο κωδικοποιήσεις εκτελούνται παράλληλα και ανεξάρτητα η μία από την άλλη. Η επιφάνεια και η καθυστέρηση που προσθέτουν στο συνολικό κύκλωμα είναι ίση με:

$$A_{MBPPG} = 2 * \left[5 * \left(\frac{N}{2} - 1\right) * A_G + N * A_G + 2A_G + 5 * N * \left(\frac{N}{2}\right) * A_G\right] \quad (3.42)$$

$$T_{MBPPG} = 8 * T_G \quad (3.43)$$

Η αντιστροφή του D γίνεται πιο απλά αφού αντιστρέφεται μόνο που στη περίπτωση που είναι αρνητικό οπότε

$$A_{reverseD} = 2N * 2A_G \quad (3.44)$$

$$T_{reverseD} = 4T_G < T_{MBPPG} \quad (3.45)$$

Το δέντρο Wallace δέχεται ένα λιγότερο μερικό γινόμενο σε σχέση με τη σχεδίαση αθροιστή απόλυτης τιμής αφού δεν προστίθεται το -1 ως διόρθωση άρα:

$$T_{Tree} = Depth(N + 3) * 4T_G \quad (3.46)$$

Για το overflow του αποτελέσματος χρησιμοποιείται ακριβώς η ίδια ανάλυση με πριν μόνο που η πληροφορία του προσήμου η οποία είναι αναγκαία για τον υπολογισμό του overflow αποκτάται στο τέλος του πρώτου αθροιστή οπότε υπολογίζεται παράλληλα με τη μονάδα increment. Ολες οι αναλύσεις σχετικά με τη μονάδα increment παραμένουν ίδιες. Το συνολικό υλικό που χρησιμοποιείται φαίνεται στο παρακάτω πίνακα.

Πίνακας 3.14 *Απαραίτητο υλικό καθώς και επιπτώσεις στην επιφάνεια και τη καθυστέρηση.*

Στοιχείο	**Επιφάνεια**	**Καθυστέρηση**
MB encoding-PPG-reverse D	$A_{MBPPG}+A_{reverseD}$	$8T_G$
Wallace Tree	A_{TREE}	$Depth(N+3) * 4T_G$
CLA-increment-overflow	$A_{CLA}+A_{increment}+A_{overflow}$	$T_{CLA}+T_{increment}$
Registers	$6N*7A_G$	$4T_G$

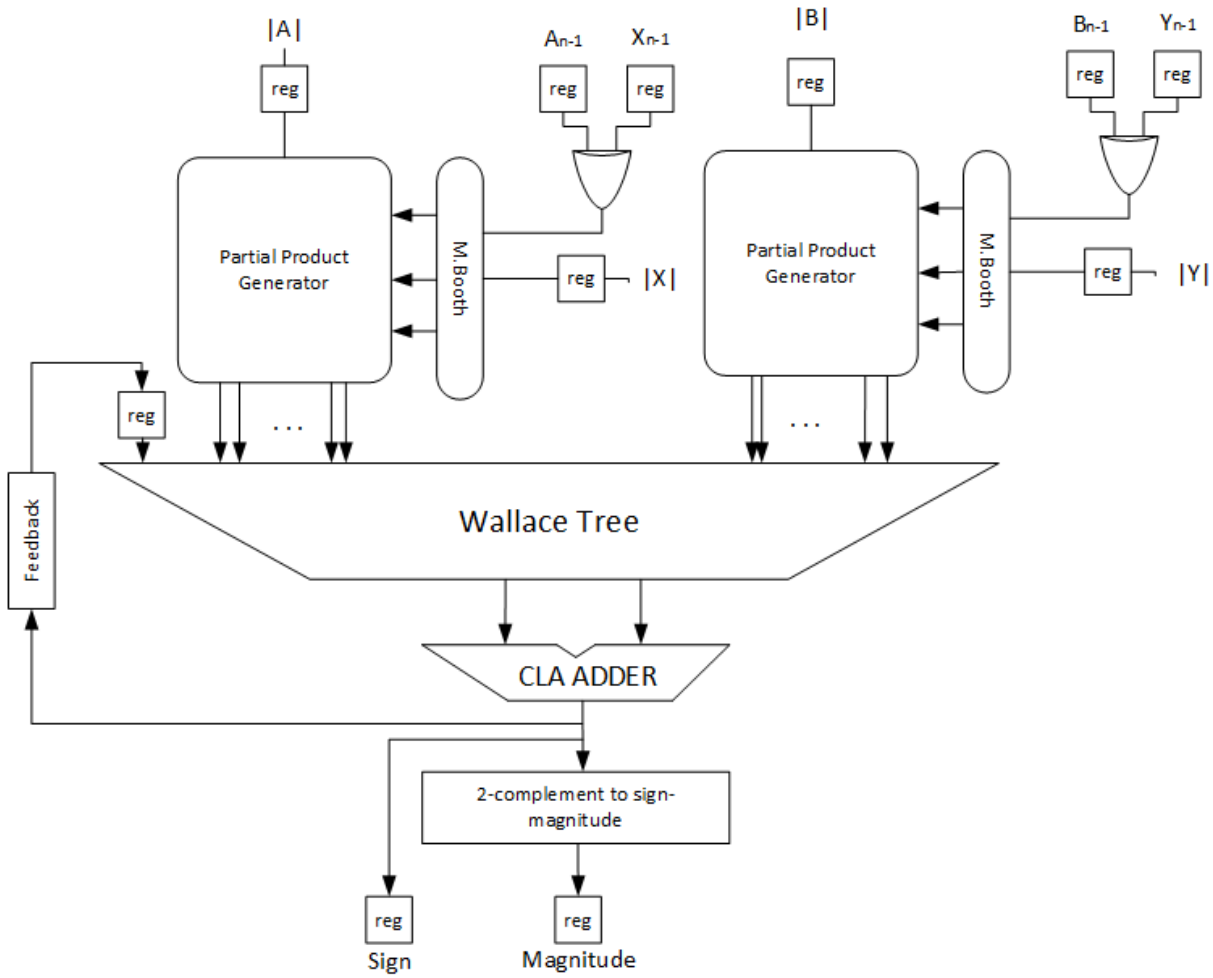
3.9 Υλοποίηση double MAC

Σε αυτή τη παράγραφο θα αναλυθούν δύο διαφορετικοί τρόποι σχεδίασης double MAC δηλαδή κυκλώματος για υπολογισμό της αριθμητικής πράξης $S = \sum(A_i * X_i + B_i * Y_i)$.

3.9.1 Σχεδίαση με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο

Σε αυτή τη παράγραφο παρουσιάζεται η σχεδίαση ενός MAC ο οποίος κάνει αποθήκευση του αποτελέσματος και πάνω σε αυτό προσθέτει το άθροισμα των δύο γινομένων των εισόδων, η όλη διαδικασία ολοκληρώνεται σε ένα κύκλο ρολογιού και βρίσκεται σε πλήρη αντιστοιχία με την μελέτη της προηγούμενη σχεδίασης με τη διαφορά ότι δεν υπάρχει η είσοδος D και το ρόλο της παίζει το αποτέλεσμα της λειτουργίας του προηγούμενου κύκλου, επειδή σε αυτό το σχεδιασμό γίνεται μετατροπή του αποτελέσματος από τη μορφή προσήμου-μέτρου στη μορφή συμπληρώματος ως προς δύο η ανατροφοδότηση του αποτελέσματος γίνεται από την έξοδο του πρώτου αθροιστή όπου εκεί το αποτέλεσμα είναι αποτυπωμένο στη μορφή συμπληρώματος ως προς δύο ώστε να μην χρειάζεται η εκ νέου μετατροπή.

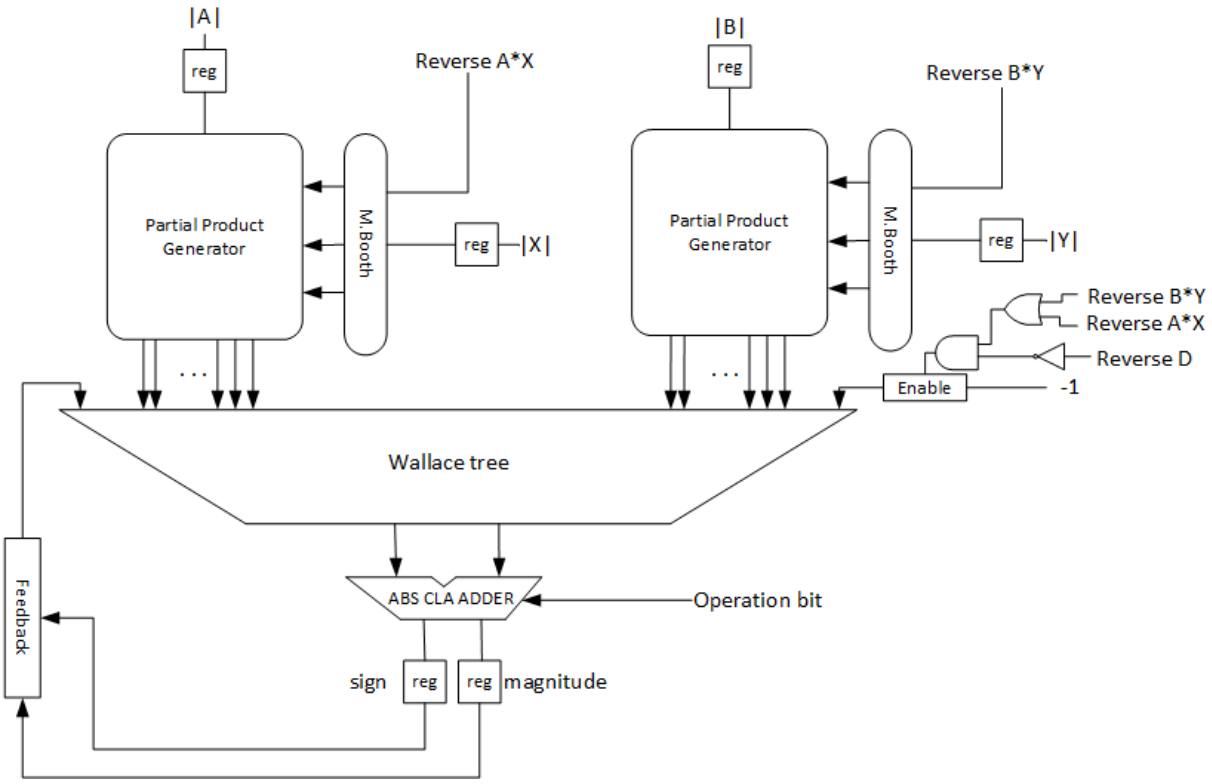
Ο τρόπος υπολογισμού του overflow είναι αντίστοιχος με εκείνος που μελετήθηκε στη προηγούμενη παράγραφο, ενώ το πρόσημο του αποτελέσματος καθορίζεται από το MSB της εξόδου του πρώτου αθροιστή. Ο σχεδιασμός που προτείνεται παρουσιάζεται στο επόμενο σχήμα:



Σχήμα 3.23 Σχεδίαση MAC για accumulation δύο γινομένων με χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο.

3.9.2 Σχεδίαση με αθροιστή απόλυτης τιμής.

Και αυτή η σχεδίαση βρίσκεται σε πλήρη αντιστοιχία με την σχεδίαση MAC για υπολογισμό της αριθμητικής πράξης $A * X + B * Y + D$ μόνο που τον ρόλο του καταχωρητή της εισόδου D τον αναλαμβάνει ο καταχωρητής εξόδου, η ολοκλήρωση της πράξης γίνεται σε ένα κύκλο ρολογιού και ισχύουν ακριβώς οι ίδιες συνθήκες για τα πρόσημα και τον εντοπισμό υπερχείλισης του αποτελέσματος, σε περίπτωση υπερχείλισης του αποτελέσματος είτε γίνεται reset είτε saturation του αποτελέσματος. Η προτεινόμενη σχεδίαση φαίνεται στο ακόλουθο σχήμα:



Σχήμα 3.24 Σχεδίαση MAC για accumulation δύο γινομένων με χρήση αθροιστή απόλυτης τιμής.

Όπως και στο accumulation ενός γινομένου στο σύστημα feedback εκτός από την αλλαγή του αποτελέσματος στην περίπτωση overflow γίνεται και η κατάλληλη αντιστροφή του για τη σωστή λειτουργία του κυκλώματος, το σήμα αντιστροφής του σχηματίζεται ακριβώς όπως το σήμα reverse D της προηγούμενης ενότητας. Λόγω της μεγάλης ομοιότητας των σχεδιασμών με τη προηγούμενη υλοποίηση double MAD η θεωρητική ανάλυση παραλείπεται αφού η κύρια διαφορά της είναι στον αριθμό των καταχωρητών όπου πλέον χρησιμοποιούνται 6N αντί για 8N.

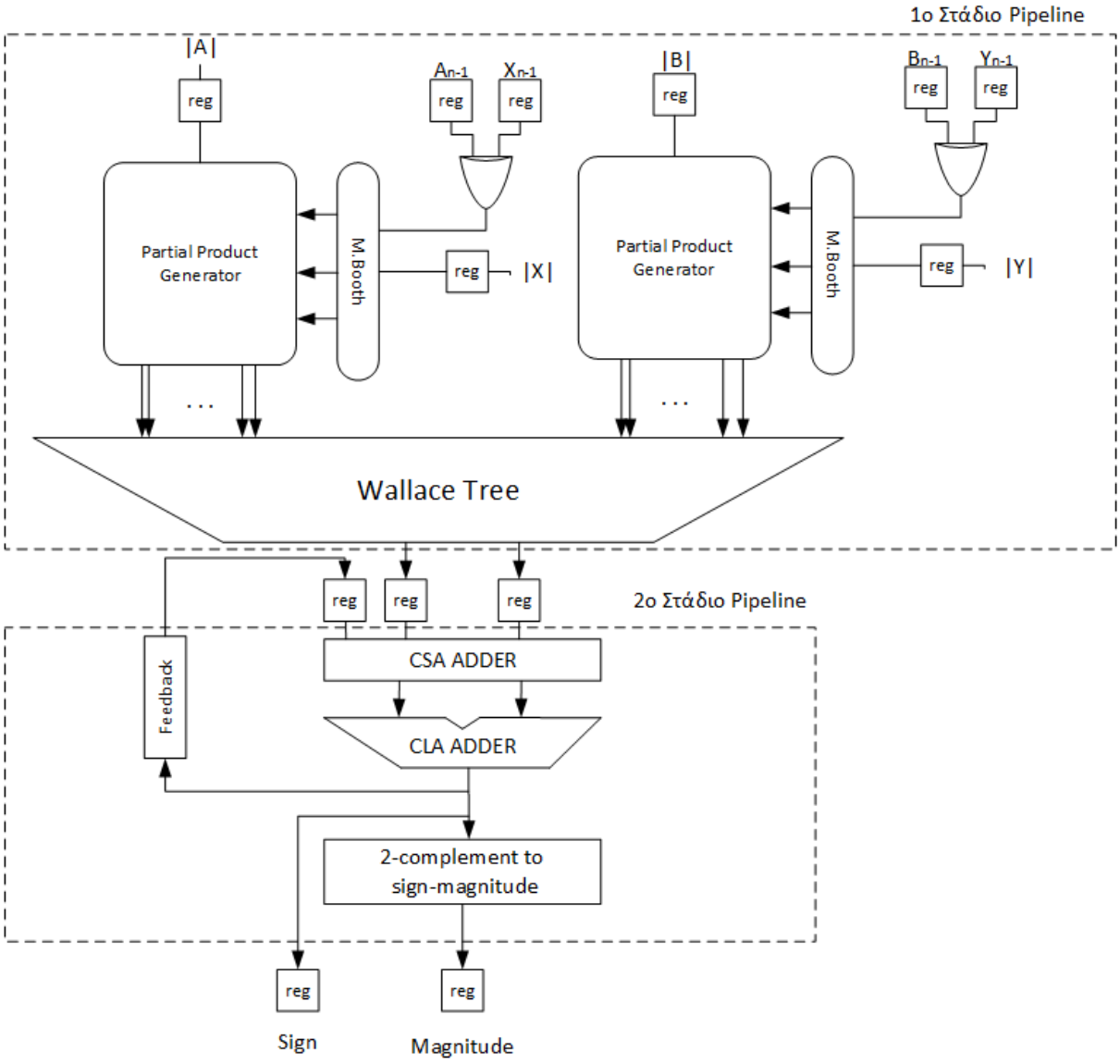
3.10 Υλοποίηση double MAC με λειτουργία συνεχούς διοχέτευσης.

Σε αυτή τη παράγραφο αναλύονται δύο διαφορετικοί τρόποι σχεδίασης double MAC, δηλαδή κυκλώματος για υπολογισμό της αριθμητικής πράξης $S = \sum(A_\iota * X_\iota + B_\iota * Y_\iota)$

3.10.1 Σχεδίαση με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο

Για την περαιτέρω μείωση του ρολογιού προτείνεται ο διαχωρισμός της λειτουργίας του MAC σε δύο στάδια. Το πρώτο στάδιο αναλαμβάνει τη MB κωδικοποίηση και τον δεντρικό συμπιεστή Wallace, ενώ το δεύτερο στάδιο αναλαμβάνει την άθροιση των δύο διανυσμάτων εξόδου s και c του δέντρου και το αποτέλεσμα του προηγούμενου κύκλου. Όπως και στη μελέτη της προηγούμενης λειτουργίας συνεχούς διοχέτευσης για την άθροιση των τριών διανυσμάτων θα χρησιμοποιηθεί ένας carry-save αθροιστής ο οποίος μετατρέπει τα τρία διανύσματα σε δύο ισοδύναμα. Το τελικό αποτέλεσμα παράγεται σε δύο κύκλους ρολογιού ενώ συνεχίζεται να ακολουθείται ακριβώς η ίδια συλλογιστική πορεία που έχει εφαρμοστεί μέχρι στιγμής όταν πρόκειται να χρησιμοποιηθεί μετατροπέας από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο, δηλαδή αντιστρέφονται οι τα σήματα προσήμων της MB κωδικοποίησης όταν τα γινόμενα είναι αρνητικά, η ανατροφοδότηση του τελικού αποτελέσματος γίνεται σε μορφή συμπληρώματος ως προς δύο δηλαδή από την έξοδο του πρώτου αθροιστή ώστε να μην χρειάζεται η εκ νέου μετατροπή του κάτι που θα αύξαινε την καθυστέρηση του σταδίου άθροισης. Στη λειτουργία συνεχούς διοχέτευσης accumulation δύο γινομένων, το δέντρο Wallace είναι αρκετά πιο αργό σε σχέση με το αθροιστή διότι συμπιέζει το διπλάσιο αριθμό μερικών γινομένων σε σχέση με τη λειτουργία συνεχούς διοχέτευσης accumulation ενός γινομένου, ενώ το στάδιο άθροισης παραμένει ίδιο.

Ο προτεινόμενος σχεδιασμός παρουσιάζεται στο επόμενο σχήμα:



Σχήμα 3.25 Λειτουργία συνεχούς διοχέτευσης MAC δύο σταδίων για accumulation δύο γινομένων με χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο.

3.10.2 Σχεδίαση με αθροιστή απόλυτης τιμής

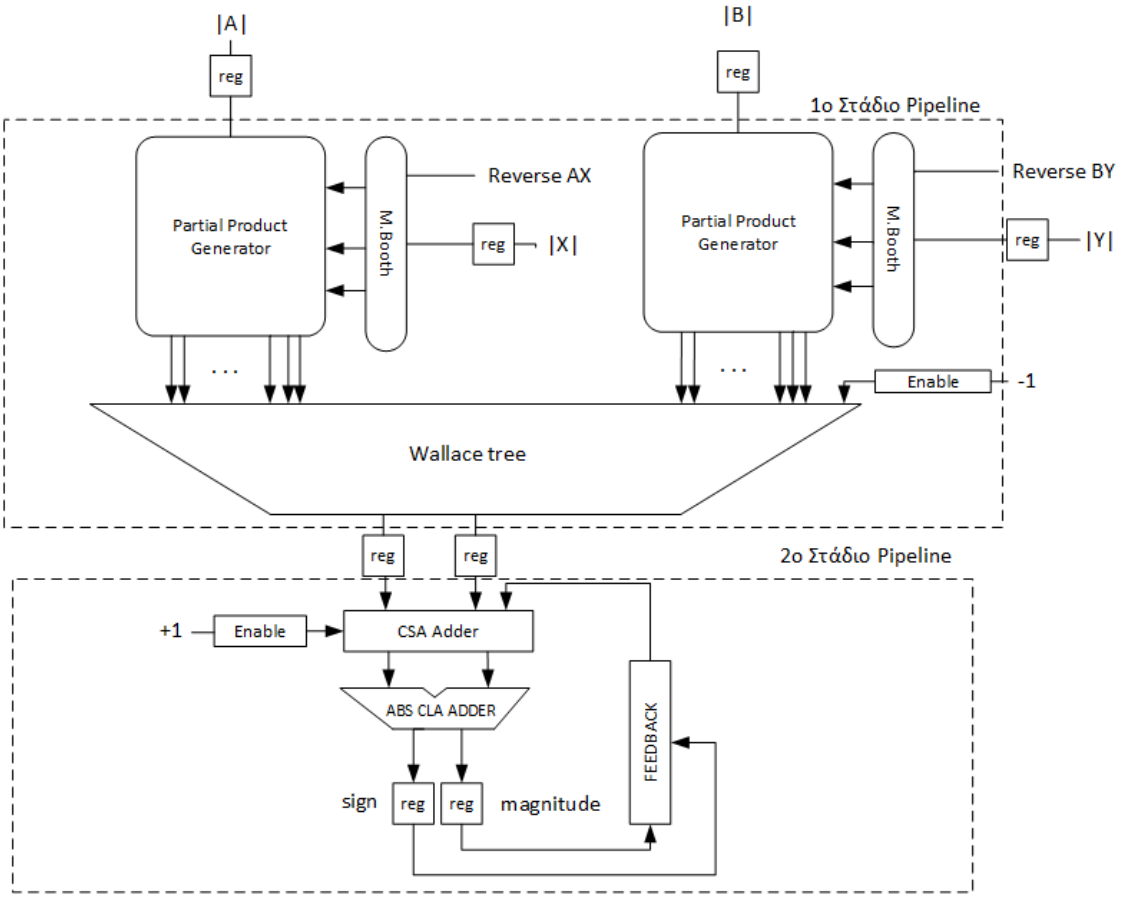
Προκειμένου να γίνει δυνατή η χρήση αθροιστή απόλυτης τιμής σε λειτουργία συνεχούς διοχέτευσης δύο σταδίων πρέπει να γίνουν ορισμένες αλλαγές. Σε αντίθεση με τη προηγούμενη σχεδίαση με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο η λειτουργία συνεχούς διοχέτευσης, η σχεδίαση με αθροιστή απόλυτης τιμής έχει ορισμένες διαφορές σε σχέση με τη συνδυαστική λειτουργία, δηλαδή την ολοκλήρωση της πράξης σε ένα κύκλο ρολογιού. Η κύρια διαφορά είναι ότι στην αρχή του πρώτου σταδίου που περιλαμβάνει MB κωδικοποίηση και δέντρο Wallace, υπάρχει η πληροφορία του πρόσημου μόνο των δύο γινομένων, το πρόσημο του αποτελέσματος πάνω στο οποίο θα προστεθούν τα δύο γινόμενα γίνεται γνωστό στο τέλος του σταδίου άθροισης. Αυτό είναι σημαντικό διότι σε περίπτωση που τα γινόμενα έχουν το ίδιο πρόσημο δεν γνωρίζεται αν ο αθροιστής μας θα λειτουργήσει ως αθροιστής ή ως αφαιρέτης απόλυτης τιμής.

Για να ξεπεραστεί το παραπάνω πρόβλημα, το operation bit το οποίο καθορίζει τη λειτουργία του αθροιστή υπολογίζεται στην αρχή του δεύτερου κύκλου, όπου υπάρχει η πληροφορία για το πρόσημο του προηγούμενου αποτελέσματος, Επιπλέον τα δύο γινόμενα εισόδου αντιστρέφονται μόνο στην περίπτωση που έχουν διαφορετικό πρόσημο, όπου και αντιστρέφεται το αρνητικό από τα δύο, ενώ η προσθήκη του -1 σαν ένα επιπλέον μερικό γινόμενο είναι πάλι αναγκαία προκειμένου να σχηματιστεί είτε ο αριθμός $\overline{A * X}$ είτε ο $\overline{B * Y}$ για την λειτουργία του αθροιστή απόλυτης τιμής, Στην περίπτωση που τα δύο γινόμενα έχουν το ίδιο πρόσημο τότε δεν αντιστρέφεται κανένα, ακόμα και αν τα πρόσημά τους είναι αρνητικά, αλλά αντιστρέφεται το αποτέλεσμα πάνω στο οποίο τα δύο γινόμενα αθροίζονται όταν το πρόσημο του προηγούμενου αποτελέσματος έχει διαφορετικό πρόσημο από τα δύο γινόμενα.

Στην περίπτωση που τα δύο γινόμενα είναι ετερόσημα τότε υπάρχει μια διαφοροποίηση σε σχέση με τη συνδυαστική λειτουργία, προηγουμένως ήταν γνωστά όλα τα πρόσημα στην αρχή εκτέλεσης του κυκλώματος και ήταν δυνατή η απλοποίηση του -1 ως μερικό γινόμενο και του +1. Αυτός ο γρήγορος υπολογισμός στη λειτουργία συνεχούς διοχέτευσης είναι αδύνατος λόγω άγνοιας του προσήμου του αποτελέσματος του αθροιστή, οπότε είναι αναγκαίο να γίνεται πάντα η αφαίρεση του -1 ως μερικό γινόμενο στο δέντρο Wallace στην περίπτωση ετερόσημων γινομένων. Το +1 στην περίπτωση μετατροπής του αποτελέσματος σε συμπλήρωμα ως προς δύο γίνεται στον CSA αθροιστή που υπάρχει στο τέλος του δέντρου Wallace. Όπως έχει αναλυθεί και στο δεύτερο κεφάλαιο η έξοδος του CSA αθροιστή αποτελείται από δύο διανύσματα, το διάνυσμα s και το διάνυσμα c, το LSB του διανύσματος c είναι πάντα μηδέν αφού τα κρατούμενα κάθε βαθμίδας είναι μετατοπισμένα μία θέση αριστερά λόγω αυξημένου βάρους τους, έτσι μπορεί να γίνει εύκολα και αποδοτικά η πρόσθεση μίας μονάδας στο τελικό αποτέλεσμα αντιστρέφοντας το λιγότερο σημαντικό ψηφίο του διανύσματος c από μηδέν σε ένα. Η άθροιση αυτή της μονάδας γίνεται στην περίπτωση που το αποτέλεσμα πάνω στο οποίο προστίθενται τα δύο γινόμενα είναι αρνητικό ενώ τα δύο γινόμενα είναι ετερόσημα. Όλα τα παραπάνω συνοψίζονται στις εξής περιπτώσεις:

- $A * X$, $B * Y$ και αποτέλεσμα αθροιστή έχουν το ίδιο πρόσημο: Κανένα μέτρο δεν αντιστρέφεται, ο αθροιστής εκτελεί πρόσθεση μέτρων.
- $A * X$, $B * Y$ ομόσημα και αποτέλεσμα αθροιστή έχει διαφορετικό πρόσημο: αντιστρέφεται το μέτρο του αποτελέσματος αθροιστή.
- $A * X$, $B * Y$ ετερόσημα και αποτέλεσμα αθροιστή είναι θετικό: Αντιστρέφεται το αρνητικό γινόμενο προστίθεται το -1 στο δέντρο Wallace.
- $A * X$, $B * Y$ ετερόσημα και αποτέλεσμα αθροιστή είναι αρνητικό: Αντιστρέφεται το αρνητικό γινόμενο προστίθεται το -1 στο δέντρο Wallace, αντιστρέφεται το μέτρο του αποτελέσματος αθροιστή, προστίθεται +1 στο CSA αθροιστή.

Ο προτεινόμενος σχεδιασμός MAC για την υλοποίηση που περιεγράφηκε φαίνεται στο παρακάτω σχήμα:



Σχήμα 3.26 Λειτουργία συνεχούς διοχέτευσης MAC δύο σταδίων για accumulation δύο γινομένων με χρήση αθροιστή απόλυτης τιμής.

Ομοίως με πριν, στο σχήμα δεν φαίνεται ο αναλυτικός υπολογισμός των σημάτων ελέγχου reverse AX, reverse BY, και των σημάτων ενεργοποίησης της άθροισης +1 και -1, αυτό έγινε για να γίνεται ο σχεδιασμός εύκολα κατανοητός. Όλα τα σήματα ελέγχου κατασκευάζονται απλά σύμφωνα με τη περιγραφή της λειτουργίας του κυκλώματος. Σημειώνεται επίσης ότι όπως και

στις προηγούμενες σχεδιάσεις με ανάδραση αποτελέσματος και χρήση αθροιστή απόλυτης τιμής το κύκλωμα feedback εκτός από reset ή saturation του αποτελέσματος σε περίπτωση υπερχείλισης κάνει και αντιστροφή του μέτρου του αποτελέσματος όταν αυτό είναι απαραίτητο.

3.10.3 Θεωρητική ανάλυση των δύο σχεδιασμών

Χρήση αθροιστή απόλυτης τιμής

Στο σχεδιασμό αυτόν η δημιουργία των σημάτων reverse AX και reverse BY είναι όμοια με τη προηγούμενη σχεδίαση μία αλλαγή που υπάρχει είναι στο κύκλωμα enable το οποίο αφήνει το -1 να γίνει είσοδος στο Wallace όταν τα γινόμενα έχουν διαφορετικό πρόσημο δηλαδή λογική πύλη XOR μεταξύ των πρόσημων των γινομένων άρα η επιφάνεια είναι:

$$A_{enable-1} = 2A_G + (N-2) * A_G \qquad (3.47)$$

$$T_{enable-1} = 3T_G \qquad (3.48)$$

Το κύκλωμα δημιουργίας του σήματος enable για το +1 στην έξοδο του CSA αθροιστή έχει επιφάνεια μίας OR πύλης και μίας AND αφού το 1 προστίθεται αν έχει αντιστραφεί τουλάχιστον ένα γινόμενο και το σήμα ανατροφοδότησης.

$$A_{enable+1} = 2A_G \qquad (3.49)$$

$$T_{enable+1} = 2T_G \qquad (3.50)$$

Το σύστημα Feedback προσθέτει μία επιπλέον καθυστέρηση στη λειτουργία του κυκλώματος αφού δεν λειτουργεί παράλληλα με κάποιο άλλο στοιχείο. Τα σήματα ελέγχου αντιστροφής του τελικού αποτελέσματος έχουν αναλυθεί σε προηγούμενη ενότητα, η συνολική καθυστέρηση και επιφάνεια που προσθέτει το σύστημα feedback είναι

$$A_{Feedback} = 6A_G + 2N * A_{XOR} + 2NA_{AND} = 6A_G + 4N * A_G + 2NA_G \qquad (3.51)$$

$$T_{Feedback} = T_{overflow} + T_{reset} + T_{reverse} = 6T_G \qquad (3.58)$$

Πίνακας 3.15 *Απαραίτητο υλικό καθώς και επιπτώσεις στην επιφάνεια και τη καθυστέρηση.*

Στοιχείο	**Επιφάνεια**	**Καθυστέρηση**
Πρώτο Στάδιο λειτουργία συνεχούς διοχέτευσης		
MB encoding-PPG-enable-1	$A_{MBPPG}+A_{enable-1}$	$9T_G$
Wallace Tree	A_{TREE}	$Depth(N+3) * 4T_G$
Δεύτερο Στάδιο λειτουργία συνεχούς διοχέτευσης		
CSA Adder	$A_{CSA}+A_{enable+1}$	T_{CSA}
CLA Absolute value	$A_{CLA}+A_{mux}$	$T_{CLA}+T_{mux}$
Feedback	$A_{Feedback}$	$T_{Feedback}$
Operation bit	$A_{operationbit}$	$T_{operationbit}$
Σύνολο καταχωρητών ολόκληρου του κυκλώματος		
Registers	$(10N+6)A_{Reg}$	$6T_G$

Χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο

Σε αυτή τη σχεδίαση οι κύριες διαφορές που υπάρχουν σε σχέση με το συνδυαστικό κύκλωμα είναι ότι το κύκλωμα feedback λειτουργεί παράλληλα με τη μονάδα increment οπότε σαν συνολική καθυστέρηση επικρατεί η μεγαλύτερη από τις δύο, για μέγεθος εισόδων ίσο με 8bit από τους θεωρητικούς τύπους οι καθυστερήσεις είναι παρόμοιες όμως με την αύξηση του μεγέθους εισόδων πάντα θα επικρατεί η καθυστέρηση της μονάδας increment

Πίνακας 3.16 *Απαραίτητο υλικό καθώς και επιπτώσεις στην επιφάνεια και τη καθυστέρηση.*

Στοιχείο	**Επιφάνεια**	**Καθυστέρηση**
Πρώτο Στάδιο λειτουργία συνεχούς διοχέτευσης.		
MB encoding-PPG	A_{MBPPG}	$8T_G$
Wallace Tree	A_{TREE}	$Depth(N+2) * 4T_G$
Δεύτερο Στάδιο λειτουργία συνεχούς διοχέτευσης.		
CSA Adder	A_{CSA}	T_{CSA}
CLA-increment-feedback	$A_{CLA}+A_{increment}+A_{Feedback}$	$T_{CLA}+T_{increment}$
Σύνολο καταχωρητών ολόκληρου του κυκλώματος.		
Registers	$(12N+1)A_{Reg}$	$6T_G$

3.11 Σχεδίαση με διαχωρισμό του δέντρου Wallace για βελτίωση λειτουργίας συνεχούς διοχέτευσης μεγάλων αριθμών

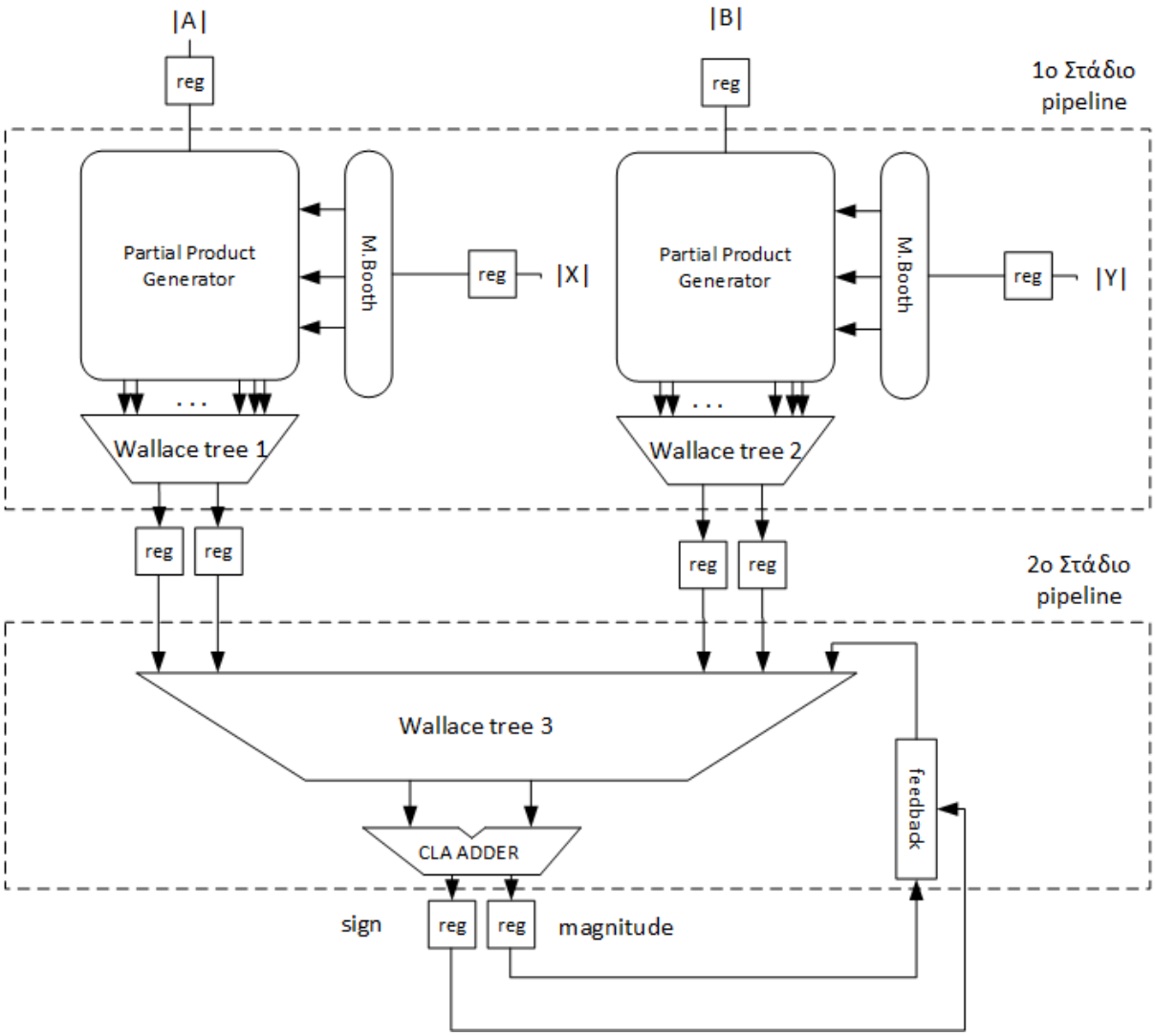
Μπορεί εύκολα να παρατηρηθεί ότι στη λειτουργία συνεχούς διοχέτευσης δύο γινομένων το δέντρο Wallace είναι πολύ πιο «βαρύ» σε επιφάνεια και καθυστέρηση από το στάδιο άθροισης, αυτό έχει σαν επίπτωση η σχεδίαση με χρήση αθροιστή απόλυτης τιμής και με μετατροπέα να παρουσιάζουν ελάχιστες διαφορά στο ελάχιστο μονοπάτι στη λειτουργία συνεχούς διοχέτευσης αφού το δέντρο Wallace είναι αυτό που καθορίζει τη συχνότητα λειτουργίας του κυκλώματος.

Το ελάχιστο μονοπάτι είναι δυνατό να μειωθεί περαιτέρω με χρήση με τον διαχωρισμό του Wallace σε δύο κομμάτια, ώστε να γίνει ελάφρυνση του πρώτου σταδίου και επιβάρυνση του δεύτερου. Αυτό βασίζεται στη βασική αρχή της αποδοτικής σχεδίασης λειτουργίας συνεχούς διοχέτευσης, δηλαδή όταν πρόκειται να διαχωριστεί ένα κύκλωμα σε κομμάτια η αποδοτικότερη σχεδίαση είναι εκείνη όπου τα κομμάτια είναι ίσα σε καθυστέρηση.

Για λόγους ο οποίοι αναλύονται στο επόμενο κεφάλαιο, όπου εξηγείται αναλυτικά ο τρόπος εξαγωγής αποτελεσμάτων σύνθεσης, ήταν αδύνατο να διαχωριστεί το δέντρο Wallace σε δύο ισοδύναμα μέρη με τρόπο που τα αποτελέσματα να είναι δίκαια και συγκρίσιμα, γι’ αυτό το λόγο παρουσιάζεται μία σχεδίαση η οποία βοηθά να γίνει κατανοητή η μείωση στο critical path από το διαχωρισμό του Wallace, όμως πρέπει να τονιστεί ότι δεν είναι η βέλτιστη σχεδίαση και παρουσιάζεται για συγκριτικούς λόγους στα πλαίσια της μελέτης μείωσης της περιόδου ρολογιού στη λειτουργία συνεχούς διοχέτευσης.

Η σχεδίαση που παρουσιάζεται περιλαμβάνει το διαχωρισμό των μερικών γινομένων από τις δύο γεννήτριες σε δύο ξεχωριστά δέντρα, το πρώτο στάδιο λειτουργίας συνεχούς διοχέτευσης σταματάει στο τέλος κάθε δέντρου με τέσσερεις καταχωρητές μεγέθους 2N bit για τα διανύσματα s και c. Στην συνέχεια ακολουθεί ένα τρίτο δέντρο που παίζει τον ίδιο ρόλο με τον CSA αθροιστή του προηγούμενου σχεδιασμού δηλαδή μετατρέπει τα 4 διανύσματα από το προηγούμενο στάδιο μαζί το διάνυσμα ανατροφοδότησης σε δύο διανύσματα κατάλληλα προς πρόσθεση, για χρήση με αθροιστή απόλυτης τιμής πρέπει να επαναληφθεί η πρόσθεση +1 όπου στις περιπτώσεις που είναι απαραίτητο.

Σημειώνεται ξανά ότι αυτή η σχεδίαση δεν αποτελεί την βέλτιστη υλοποίηση, θα ήταν περισσότερο αποδοτική η εισαγωγή όλων των μερικών γινομένων σε ένα ενοποιημένο δέντρο Wallace το οποίο να παράγει τέσσερα γινόμενα ως έξοδο. Ο διαχωρισμός των δέντρων Wallace φαίνεται στο το ακόλουθο σχήμα:



Σχήμα 3.27 Λειτουργία συνεχούς διοχέτευσης MAC με διαχωρισμό των δέντρων Wallace.

Το σχήμα που παρουσιάζεται αφορά τη γενική περίπτωση και δεν είναι συγκεκριμένα για αθροιστή απόλυτης τιμής ή με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο.

4 ΠΕΙΡΑΜΑΤΙΚΑ ΑΠΟΤΕΛΕΣΜΑΤΑ ΣΥΝΘΕΣΕΩΝ

4.1 Οργάνωση πειραμάτων

Όλα τα σχέδια που παρουσιάστηκαν στο προηγούμενο κεφάλαιο, περιεγράφηκαν σε γλώσσα υλικού Verilog σε επίπεδο λογικών πυλών εκτός από τον δεντρικό συμπιεστή Wallace για το οποίο χρησιμοποιήθηκε έτοιμο κύκλωμα της Synopsys το DW02_Tree, το δέντρο Wallace είναι σχεδιασμένο παραμετρικά και δέχεται σαν παραμέτρους εισόδου το μήκος τον μερικών γινομένων και το πλήθος τους και παράγει ως έξοδο δύο διανύσματα s και c το άθροισμα των οποίον είναι ίσο με το άθροισμα όλων των μερικών γινομένων. Αυτός είναι και ο λόγος που στην παράγραφο 3.11 χρησιμοποιήθηκαν δύο δέντρα έναντι ενός.

Η σύνθεση των κυκλωμάτων υλοποιήθηκε από το Synopsys Design Compiler κάνοντας χρήση της βιβλιοθήκης κελιών tsmc 65nm. Τα κυκλώματα προσομοιώθηκαν στο περιβάλλον ModelSim και έγινε επιβεβαίωση της ορθής λειτουργίας τους μέσω testbench τυχαίων αριθμών.

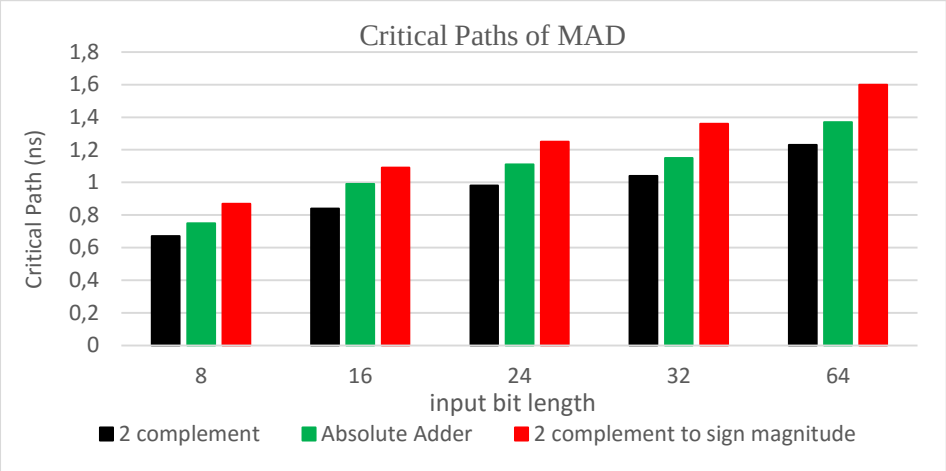
Από το περιβάλλον του Design Compiler, εξήχθησαν οι τιμές καθυστέρησης και επιφάνειας κάθε κυκλώματος, ενώ από το περιβάλλον του Synopsys PrimePower (με βάση το αρχείο .vcd που προέκυψε από την προσομοίωση του ModelSim) υπολογίστηκε η μέση κατανάλωση κάθε κυκλώματος.

Για την εύρεση της ελάχιστης περίοδος ρολογιού ο Design Compiler ξεκίναγε τις συνθέσεις από ένα πολύ χαμηλό ρολόι στο οποίο ήταν αδύνατη η σύνθεση, έπειτα αύξανε συνεχώς τη περίοδο του ρολογιού μέχρι να καταφέρει να κάνει επιτυχή σύνθεση, στη συνέχεια έκανε άλλες 10 συνθέσεις με βήμα 0.01ns. Αυτό έγινε διότι ο compiler μπορεί να έκανε μία επιτυχή σύνθεση αλλά αυτό δεν σημαίνει ότι και οι 10 επόμενες συνθέσεις θα είναι επιτυχείς, οπότε για ασφάλεια δεν επιλέχτηκε ως καθυστέρηση το ελάχιστο ρολόι στο οποίο γίνεται η σύνθεση αλλά το ρολόι στο οποίο όλες οι επόμενες συνθέσεις ήταν επιτυχείς. Έπειτα έγιναν 8-10 ακόμα συνθέσεις με βήμα 0.1ns για να εξεταστεί η συμπεριφορά των κυκλωμάτων για χαλαρότερους περιορισμούς ρολογιού

Τέλος από όλες τις παραμέτρους κρατήθηκαν: η ελάχιστη καθυστέρηση (Critical Path), Επιφάνεια σχεδίασης επί περίοδο ρολογιού ($um^2 * ns$), Ενέργεια κατανάλωσης ($mW * ns$). Όλα τα παραπάνω επαναλήφθηκαν για όλα τα σχέδια για μήκη λέξης 8,16,24,32,64 bit.

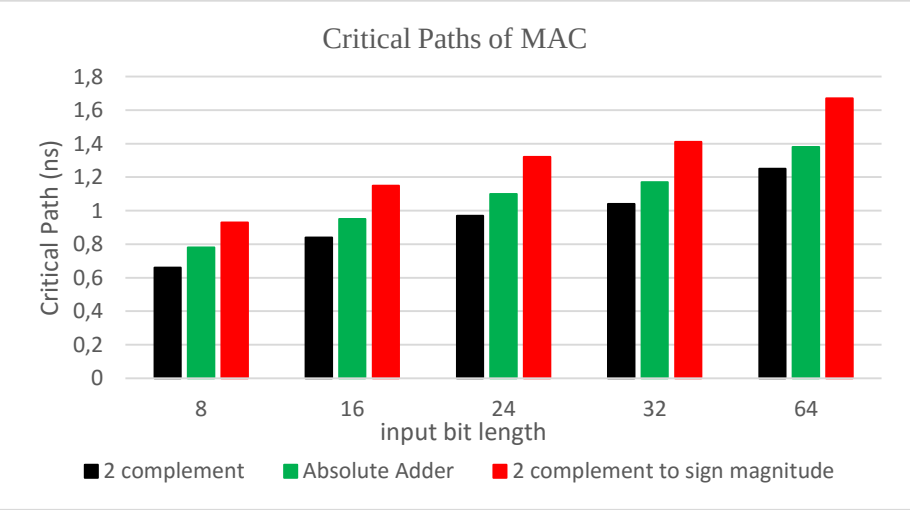
4.2 Ελάχιστη δυνατή καθυστέρηση υλοποιήσεων

Σύγκριση υλοποιήσεων MAD για υπολογισμό της αριθμητικής πράξης $A * X + D$ με αθροιστή απόλυτης τιμής, με μετατροπή από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο, με επεξεργασία αριθμών σε συμπλήρωμα ως προς δύο.



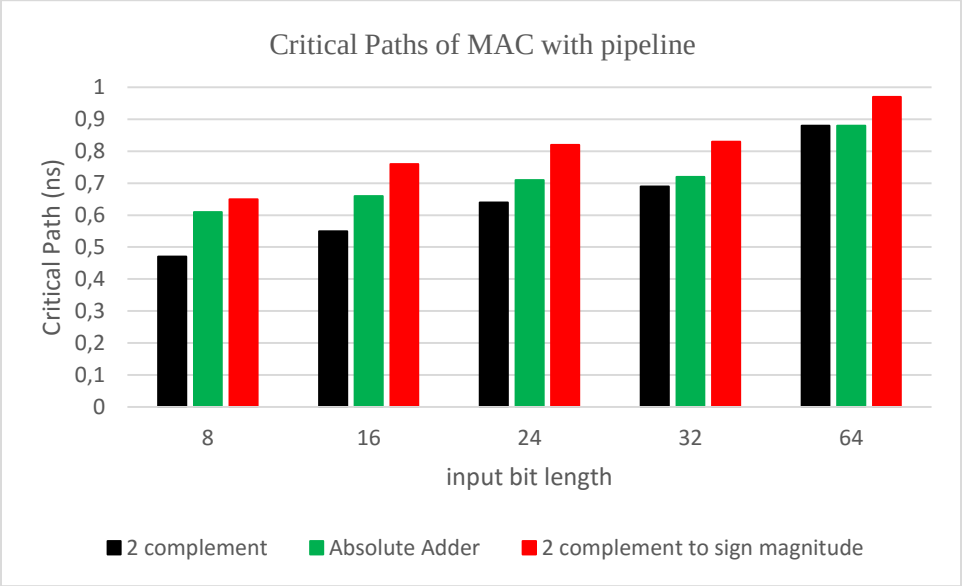
Σχήμα 4.1 Σύγκριση Critical Path για διάφορες υλοποιήσεις MAD.

Σύγκριση υλοποιήσεων MAC για υπολογισμό της αριθμητικής πράξης $\sum A_i * X_i$ σε ένα κύκλο ρολογιού με αθροιστή απόλυτης τιμής, με μετατροπή από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο, επεξεργασία αριθμών σε συμπλήρωμα ως προς δύο.



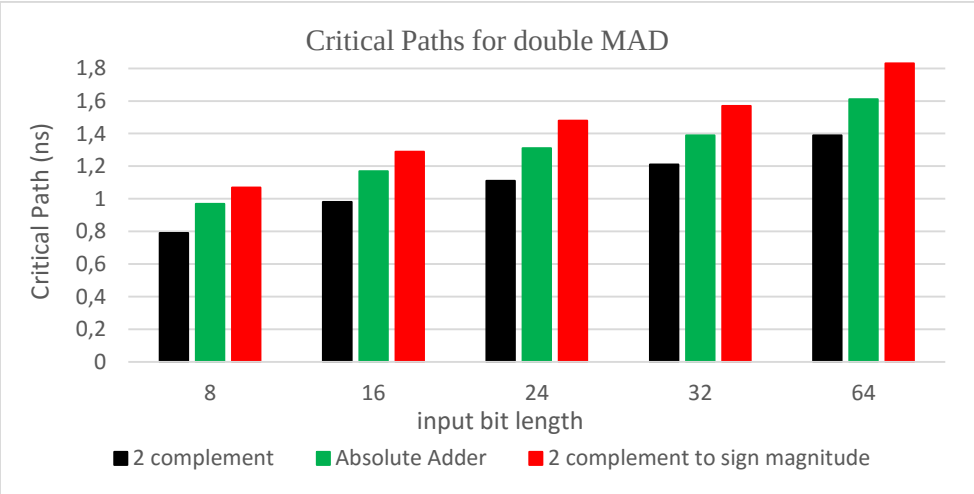
Σχήμα 4.2 Σύγκριση Critical Path για διάφορες υλοποιήσεις της αριθμητικής πράξης $\sum A_i * X_i$ σε ένα κύκλο ρολογιού.

Σύγκριση υλοποιήσεων MAC για υπολογισμό της αριθμητικής πράξης $\sum A_i * X_i$ με λειτουργία συνεχούς διοχέτευσης με αθροιστή απόλυτης τιμής, με μετατροπή από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο, με επεξεργασία αριθμών σε συμπλήρωμα ως προς δύο.



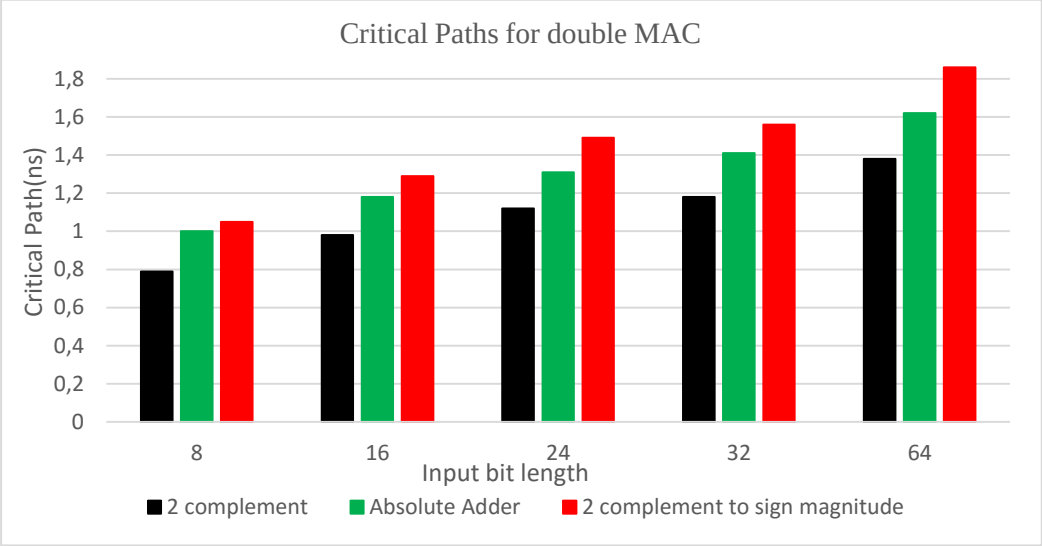
Σχήμα 4.3 Σύγκριση Critical Path για διάφορες υλοποιήσεις MAC λειτουργίας συνεχούς διοχέτευσης.

Σύγκριση υλοποιήσεων double MAD για υπολογισμό προς αριθμητικής πράξης $A * X + B * Y + + D$ με αθροιστή απόλυτης τιμής, με μετατροπή από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο, επεξεργασία αριθμών σε συμπλήρωμα ως προς δύο.



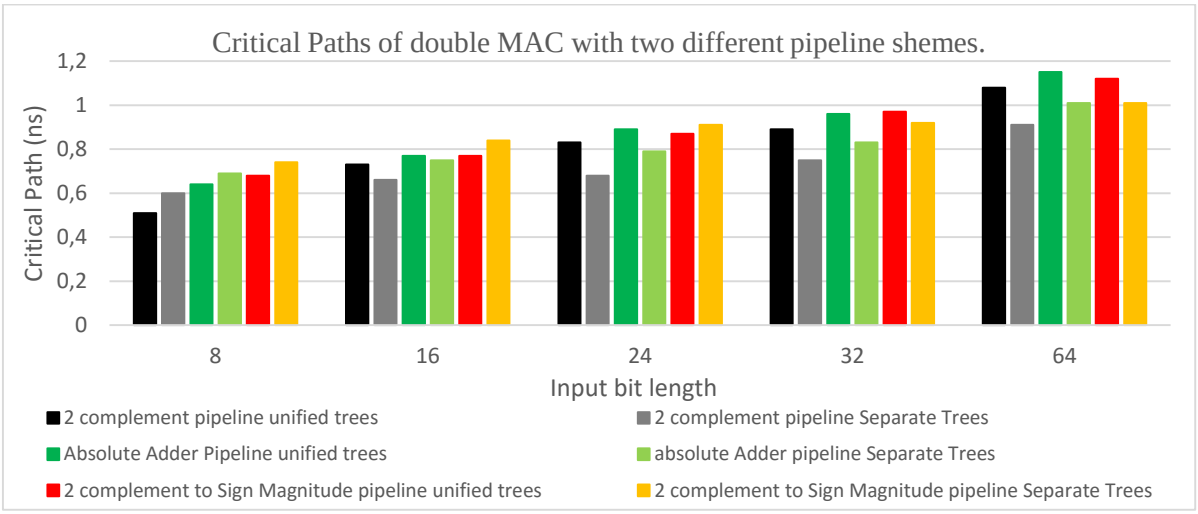
Σχήμα 4.4 Σύγκριση Critical Path για διάφορες υλοποιήσεις της αριθμητικής πράξης $A * X + B * Y + D$.

Σύγκριση λειτουργίας συνεχούς διοχέτευσης MAC για υπολογισμό της αριθμητικής πράξης $\sum A_i * X_i + B_i * Y_i$ με συνδυαστική λειτουργία σε ένα κύκλο ρολογιού και σχεδίαση: με αθροιστή απόλυτης τιμής, με μετατροπή από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο, επεξεργασία αριθμών σε συμπλήρωμα ως προς δύο.



Σχήμα 4.5 Σύγκριση Critical Path για διάφορες υλοποιήσεις της αριθμητικής πράξης $\sum A_i * X_i + B_i * Y_i$ σε ένα κύκλο ρολογιού.

Σύγκριση υλοποιήσεων MAC για υπολογισμό της αριθμητικής πράξης $\sum A_i * X_i$ για δύο διαφορετικούς σχεδιασμούς λειτουργίας συνεχούς διοχέτευσης(ενοποιημένο και διαχωρισμένο δέντρο Wallace) και σχεδίαση: με αθροιστή απόλυτης τιμής, με μετατροπή από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο, επεξεργασία αριθμών σε συμπλήρωμα ως προς δύο.



Σχήμα 4.6 Σύγκριση Critical Path για διάφορες υλοποιήσεις της αριθμητικής πράξης $\sum A_i * X_i + B_i * Y_i$ για δύο διαφορετικούς σχεδιασμούς λειτουργίας συνεχούς διοχέτευσης.

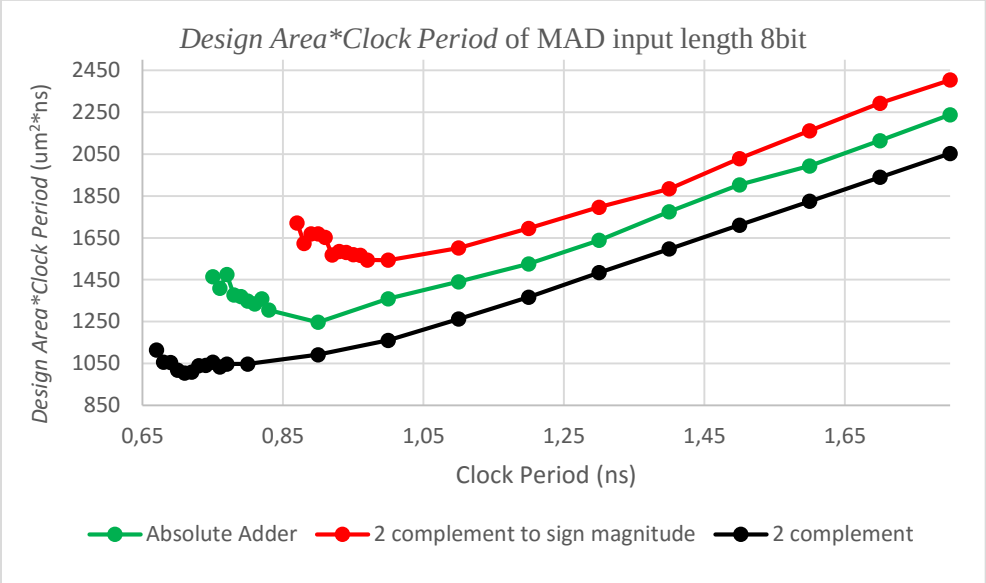
Παρατηρείται ότι στην ολοκλήρωση της πράξης σε έναν κύκλο ρολογιού η υλοποίηση με αθροιστή απόλυτης τιμής έχει πάντα καλύτερα αποτελέσματα από την υλοποίηση με μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο. Στην pipeline υλοποίηση, στην σχεδίαση MAC για accumulation ενός γινομένου η σχεδίαση με απόλυτο αθροιστή έχει μικρότερο critical path από τη σχεδίαση με μετατροπέα και στα 64bit παρατηρείται ότι επιτυγχάνεται critical path ίσο με τον MAC συμπληρώματος ως προς δύο, αυτό συμβαίνει διότι το στάδιο συμπίεσης είναι πιο αργό από το στάδιο άθροισης. Στην pipeline υλοποίηση για accumulation δύο γινομένων στην χρήση ενός ενιαίου δεντρικού συμπιεστή Wallace, ο MAC με αθροιστή απόλυτης τιμής παρουσιάζει παρόμοιο critical path (ελάχιστα μεγαλύτερο) από εκείνο με τη χρήση μετατροπέα, διότι ο υπολογισμός των σημάτων ελέγχων των πρόσημων της MB κωδικοποίησης επιβαρύνουν ως προς τη καθυστέρηση το στάδιο συμπίεσης το οποίο είναι κυρίαρχο για κάθε μήκος λέξης εκτός από τα 8 bit. Στη σχεδίαση με δύο ξεχωριστά δέντρα Wallace η καθυστέρηση μειώνεται σε σχέση με τη σχεδίαση με ενοποιημένο δέντρο όσο αυξάνεται το μήκος των εισόδων, όσο πιο γρήγορο είναι το στάδιο άθροισης η βελτίωση εμφανίζεται σε μικρότερο μήκος λέξης, ενώ όσο πιο αργό είναι το στάδιο άθροισης η βελτίωση εμφανίζεται για μεγαλύτερα μήκη λέξης.

4.3 Χρονικό ξεδίπλωμα των υλοποιημένων κυκλωμάτων

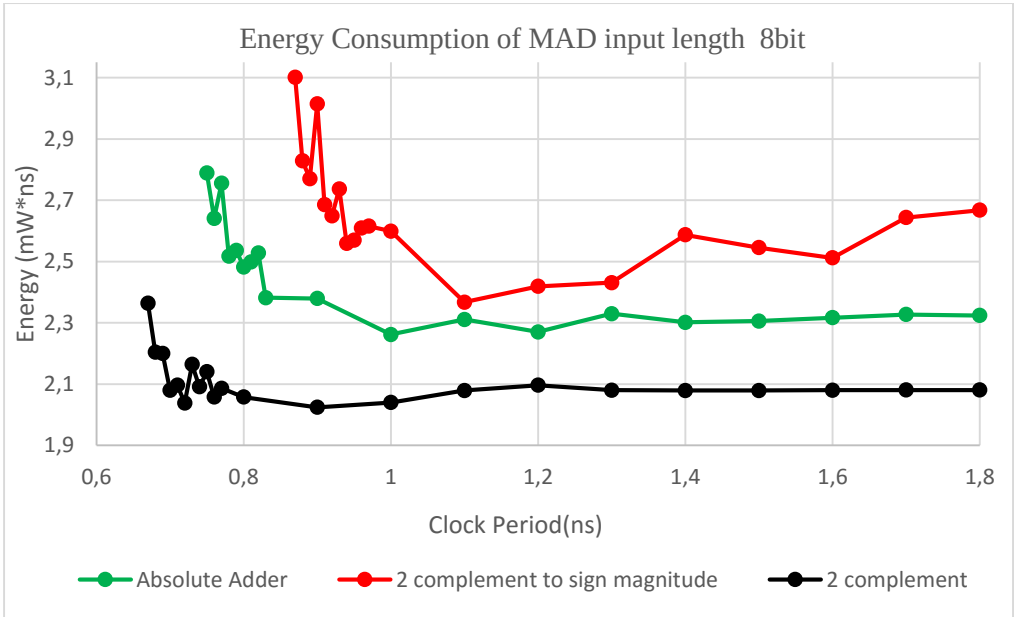
Σε αυτή τη παράγραφο παρουσιάζεται η ανάλυση των κυκλωμάτων με αύξηση της περιόδου του ρολογιού σημειώνεται ότι η προτεινόμενη λειτουργία των κυκλωμάτων είναι κοντά στην ελάχιστη περίοδο ρολογιού στην οποία γίνεται επιτυχής σύνθεση τους, η χρονική ανάλυση για χαλαρότερα ρολόγια γίνεται για συγκριτικούς λόγους.

Σύγκριση ως προς επιφάνεια σχεδίασης επί περίοδο ρολογιού και καταναλισκόμενη ενέργεια MAD για υπολογισμό της αριθμητικής πράξης $A * B + D$ με χρήση: αθροιστή απόλυτης τιμής, μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο, επεξεργασία αριθμών σε συμπλήρωμα ως προς δύο για 8,16,24,32,64 bit μέγεθος εισόδων.

- Μέγεθος εισόδων A,B: 8 bit, D:16 bit

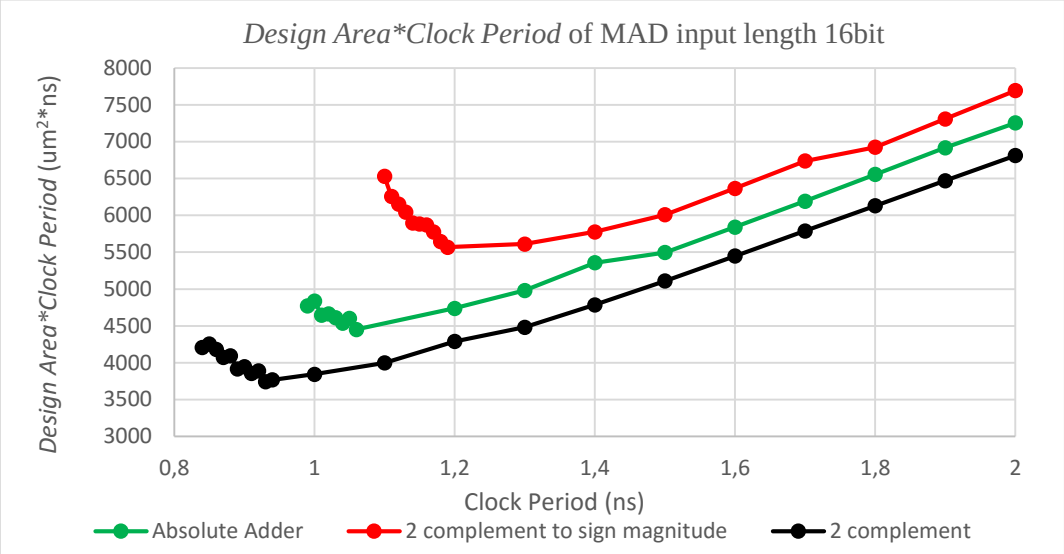


Σχήμα 4.7 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + D$.

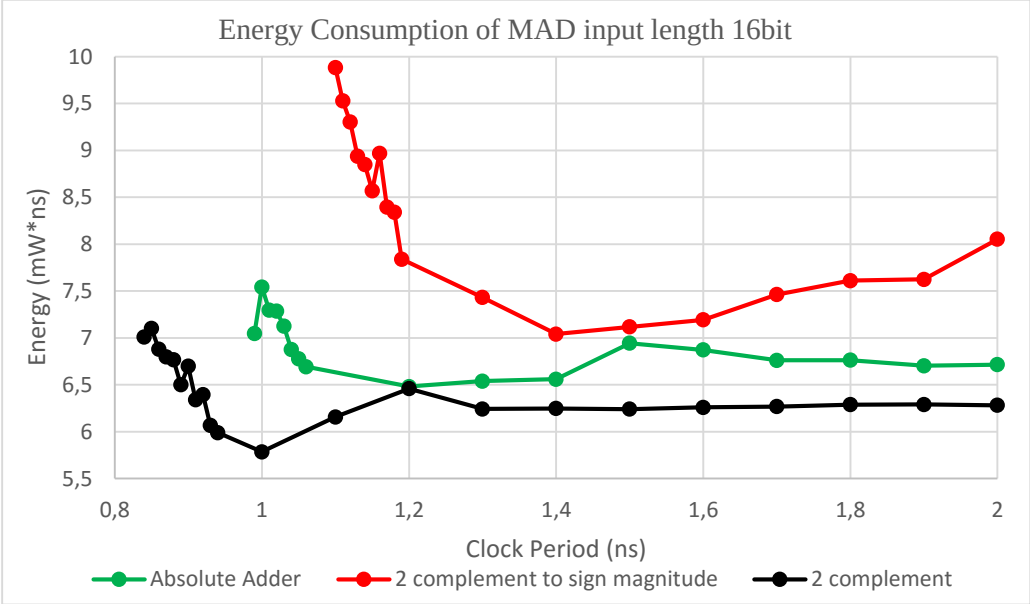


Σχήμα 4.8 Σύγκριση καταναλισκόμενης ενέργειας για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + D$.

- Μέγεθος εισόδων A,B: 16 bit, D:32 bit

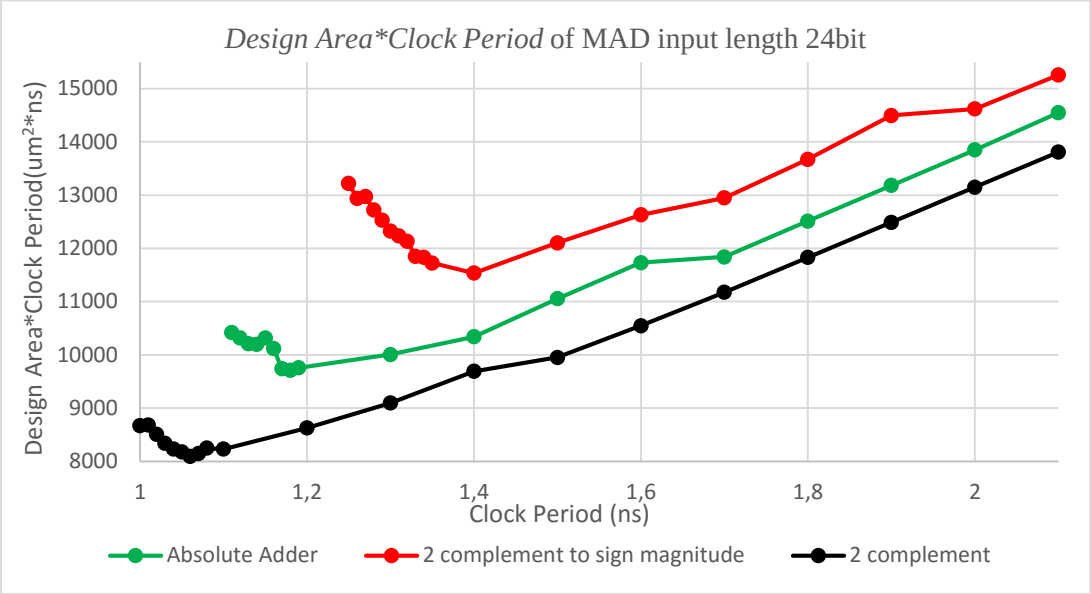


Σχήμα 4.9 Σύγκριση επιφάνειας σχεδίασης επί περίοδο για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + D$.

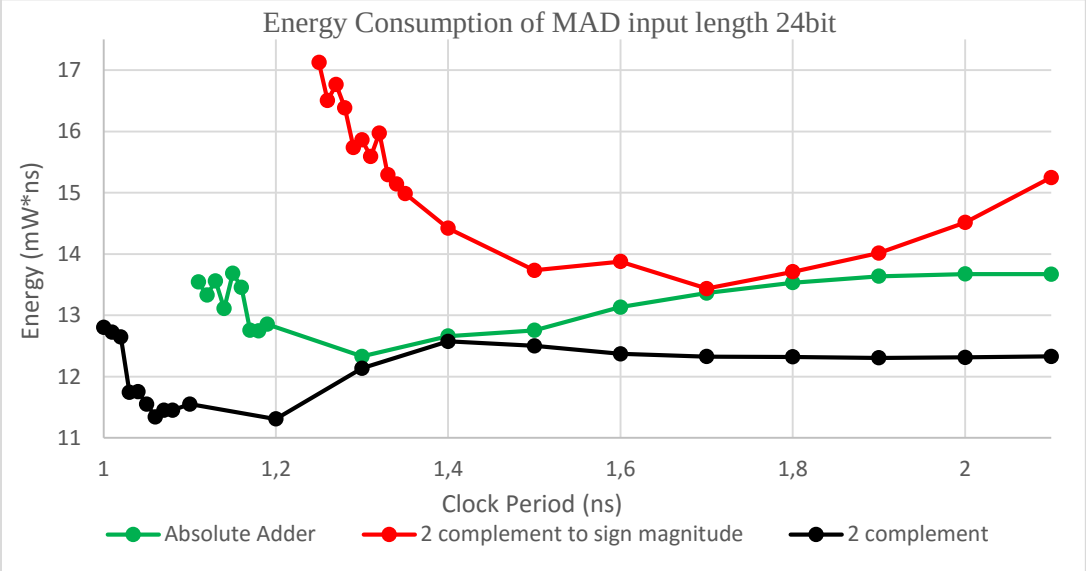


Σχήμα 4.10 Σύγκριση καταναλισκόμενης ενέργειας για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + D$.

- Μέγεθος εισόδων A,B: 24 bit, D:48 bit

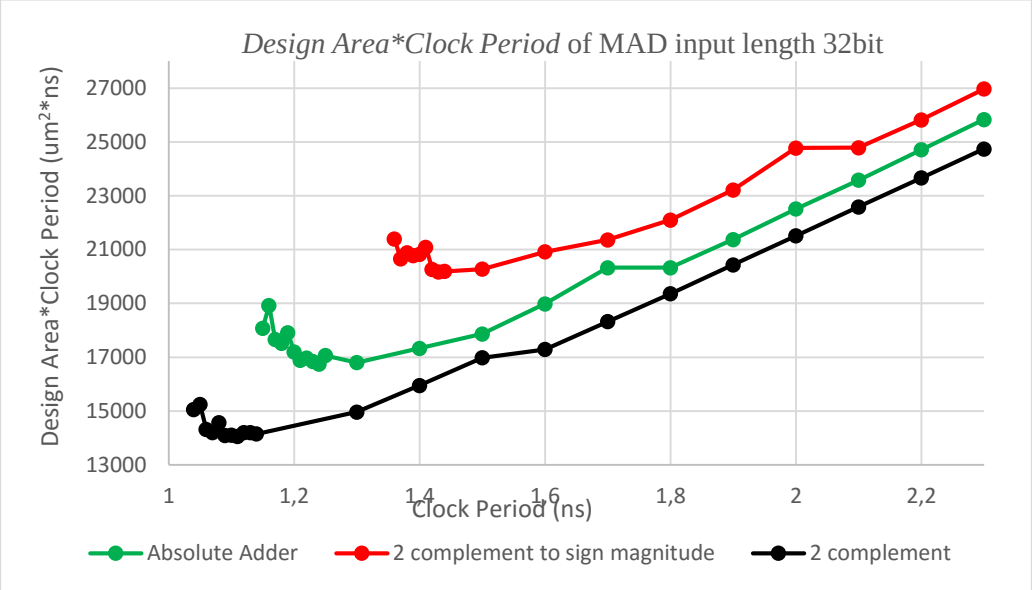


Σχήμα 4.11 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + D$.

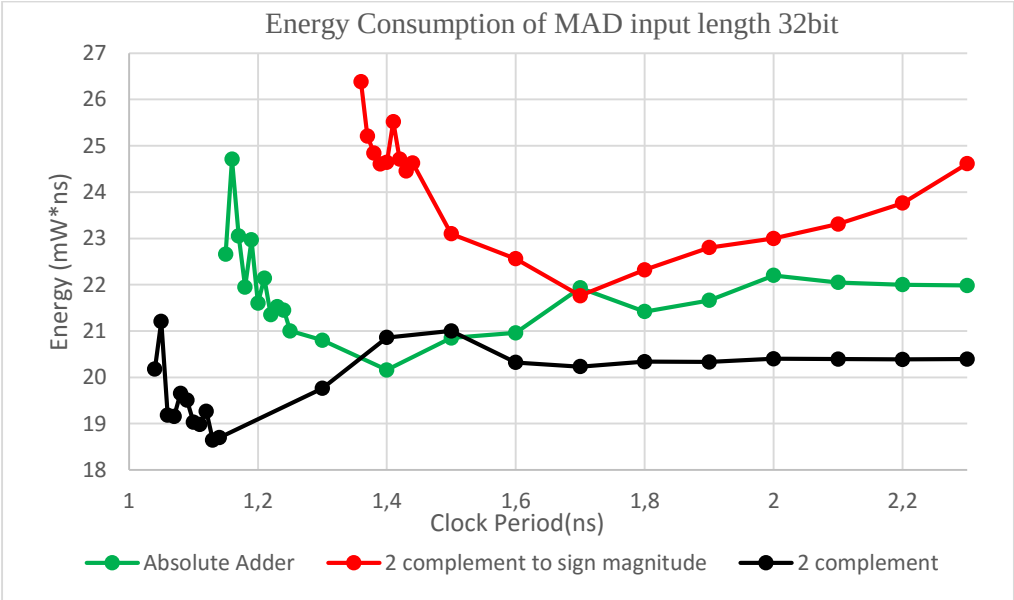


Σχήμα 4.12 Σύγκριση καταναλισκόμενης ενέργειας για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + D$.

- Μέγεθος εισόδων A,B: 32 bit, D:64 bit

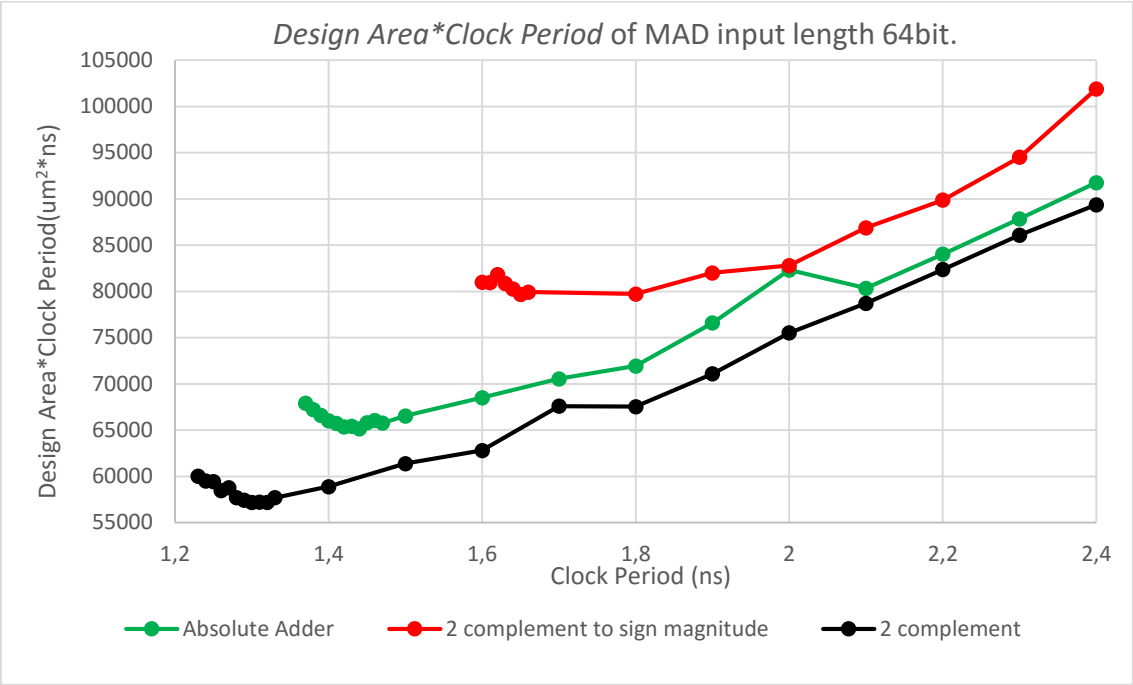


Σχήμα 4.11 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + D$.

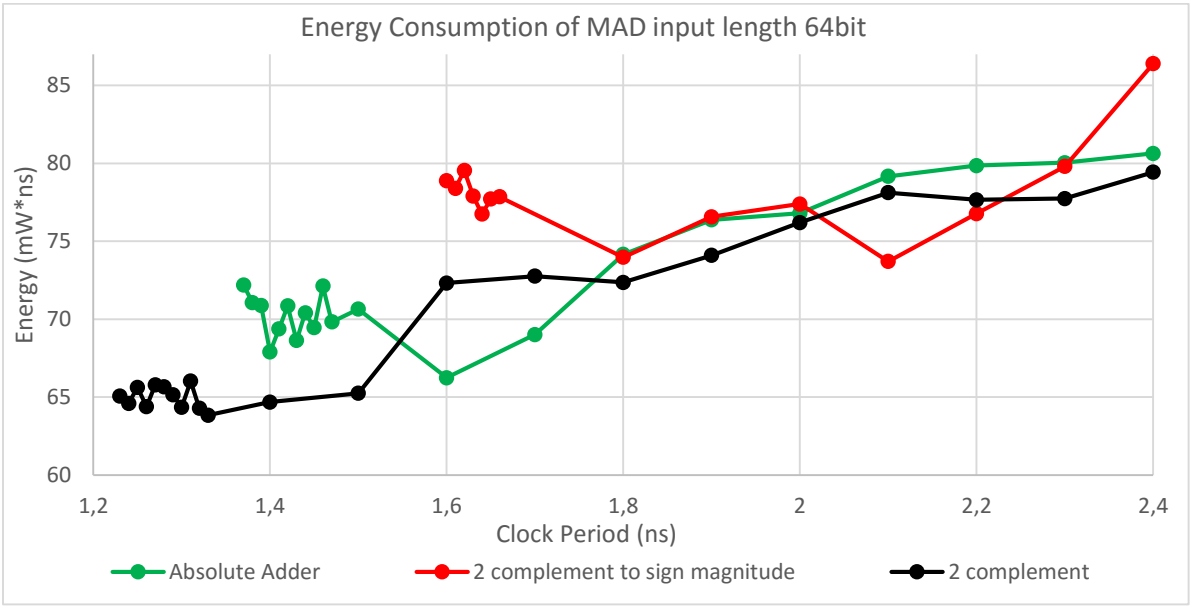


Σχήμα 4.12 Σύγκριση καταναλισκόμενης ενέργειας για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + D$.

- Μέγεθος εισόδων A,B: 64 bit, D:128 bit



Σχήμα 4.13 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + D$.

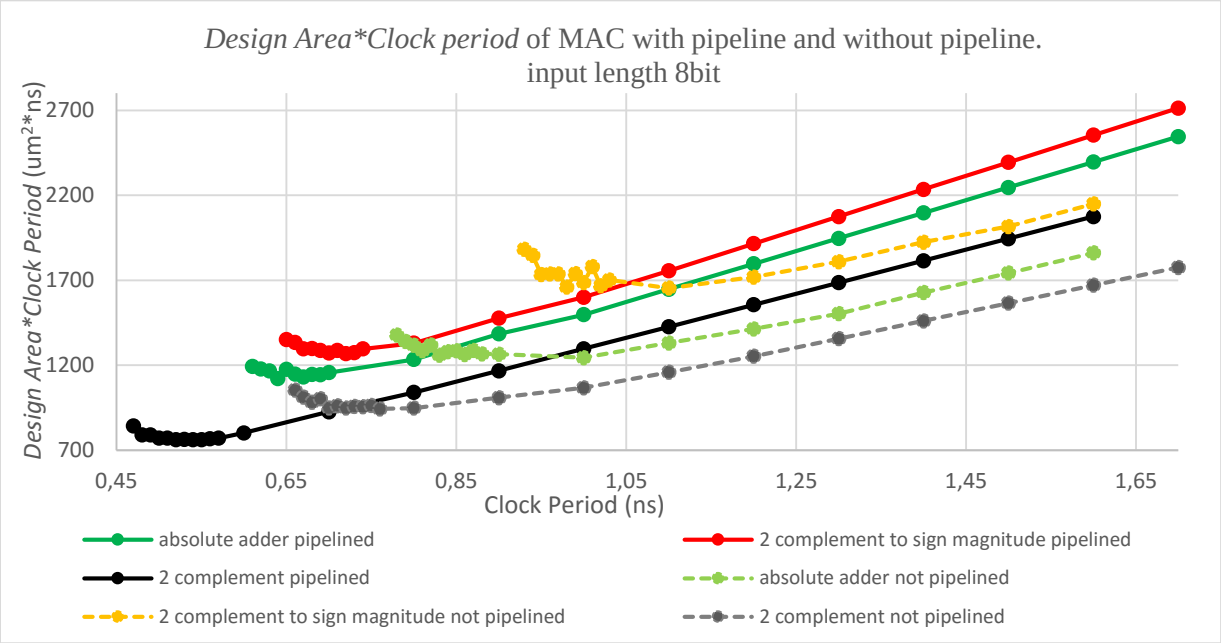


Σχήμα 4.14 Σύγκριση καταναλισκόμενης ενέργειας για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + D$.

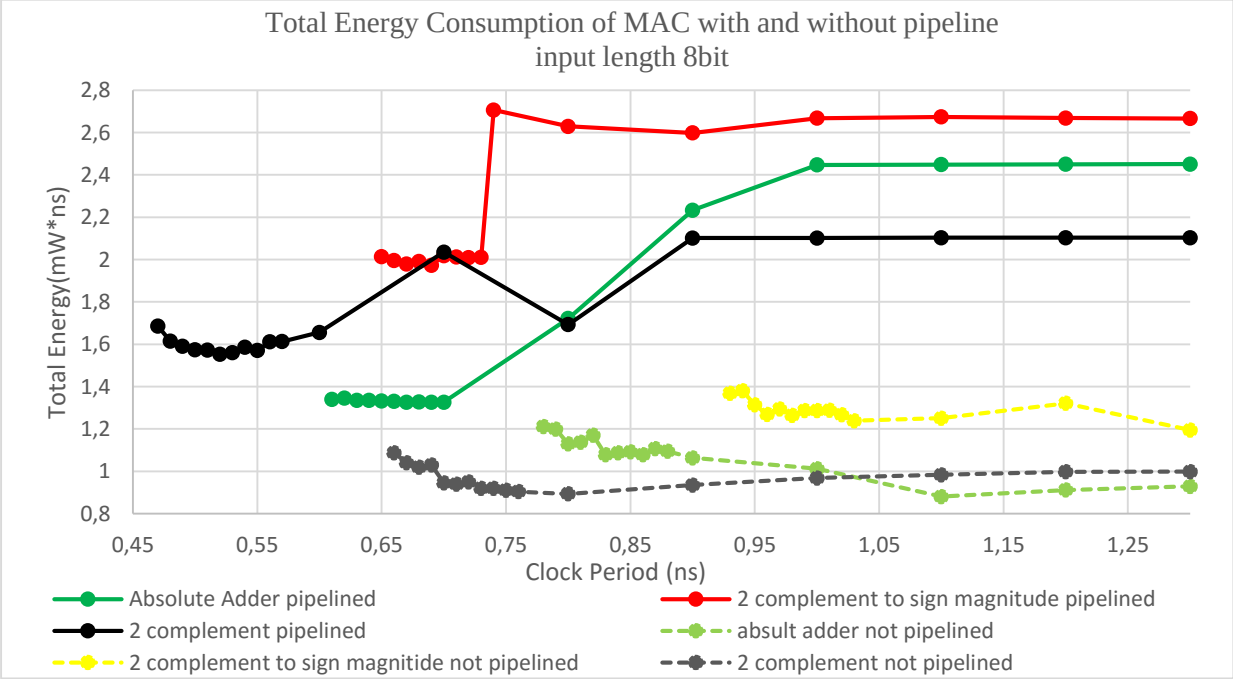
Παρατηρείται ότι η χρήση αθροιστή απόλυτης τιμής παρουσιάζει βελτίωση στη μετρική επιφάνεια επί περίοδο ρολογιού και στην ενέργεια κατανάλωσης. Επιπλέον όσο το μέγεθος των εισόδων αυξάνεται η σχεδίαση με αθροιστή απόλυτης τιμής πλησιάζει σε χαρακτηριστικά τη λειτουργία του MAC για αριθμούς συμπληρώματος ως προς δυο σε επιφάνεια και κατανάλωση.

Σύγκριση σχεδίασης MAC για accumulation ενός γινομένου δηλαδή για εκτέλεση της αριθμητικής πράξης $\sum A_i * B_i$ με λειτουργία συνεχούς διοχέτευσης σε δύο κύκλους ρολογιού και λειτουργία σε ένα κύκλο ρολογιού για υλοποιήσεις με χρήση: αθροιστή απόλυτης τιμής, μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο και επεξεργασία με αριθμούς σε συμπλήρωμα ως προς δύο

- Μέγεθος εισόδων: 8bit

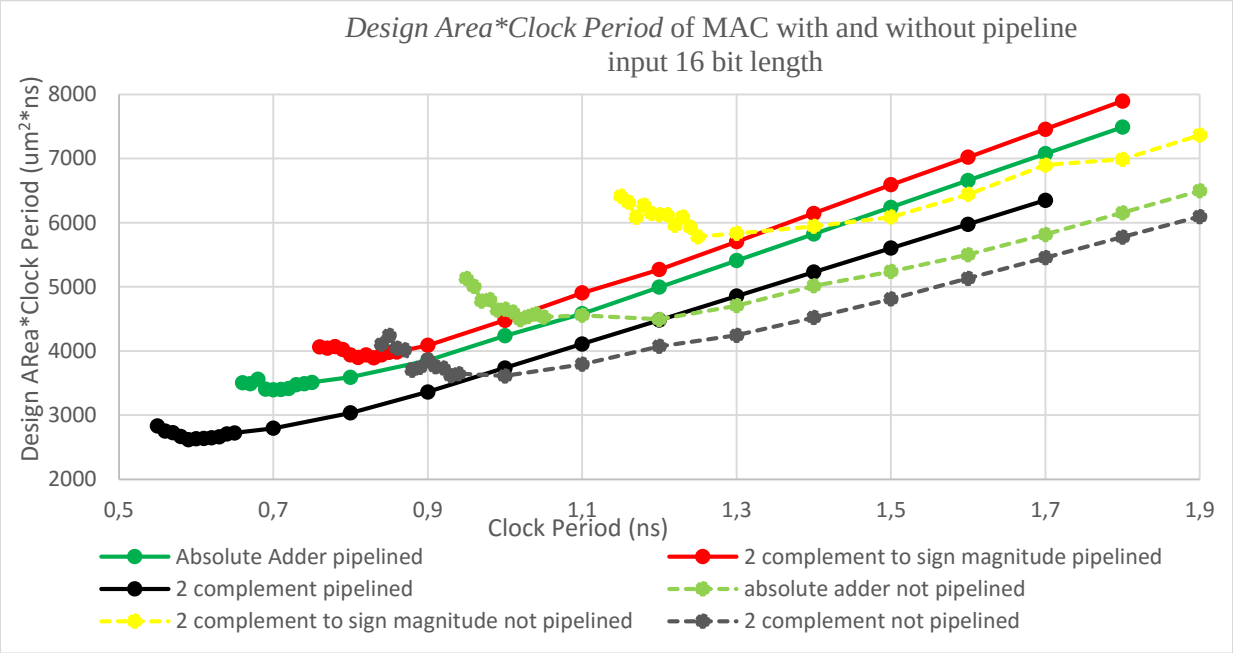


Σχήμα 4.15 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς για accumulation ενός γινομένου σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

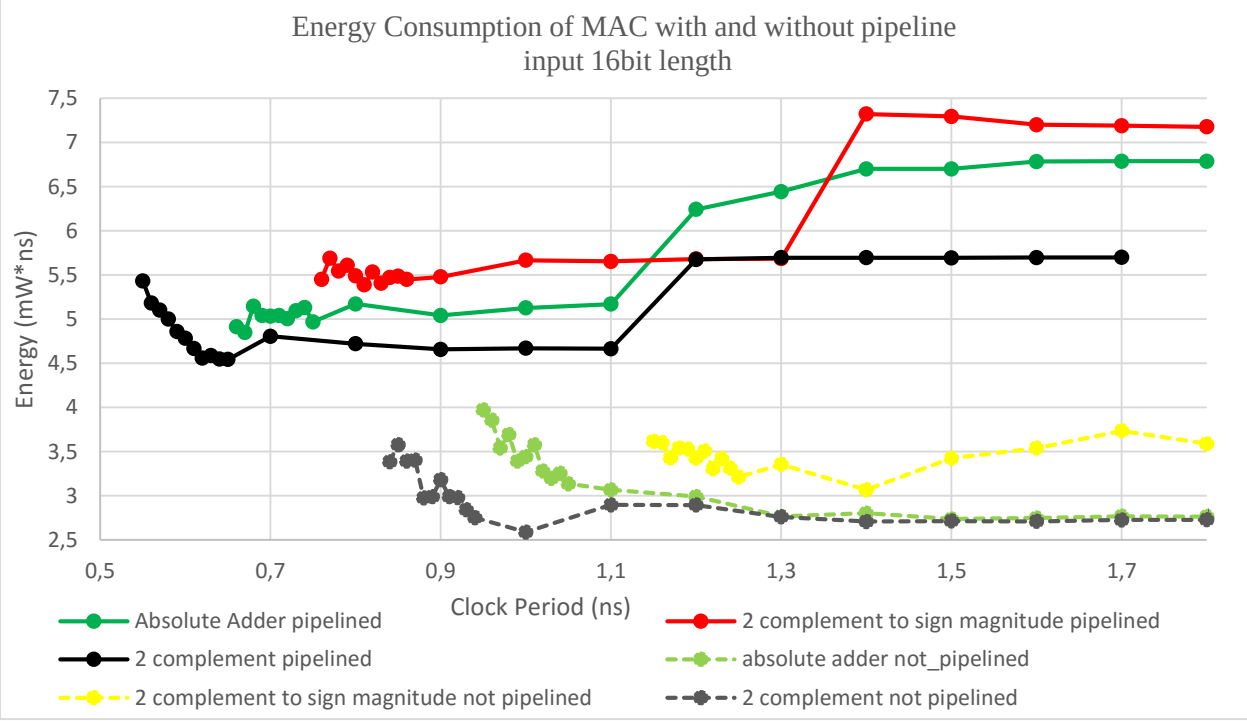


Σχήμα 4.16 Σύγκριση καταναλωμένης ενέργειας για τους διαφορετικούς σχεδιασμούς για accumulation ενός γινομένου σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

- Μέγεθος εισόδων 16bit

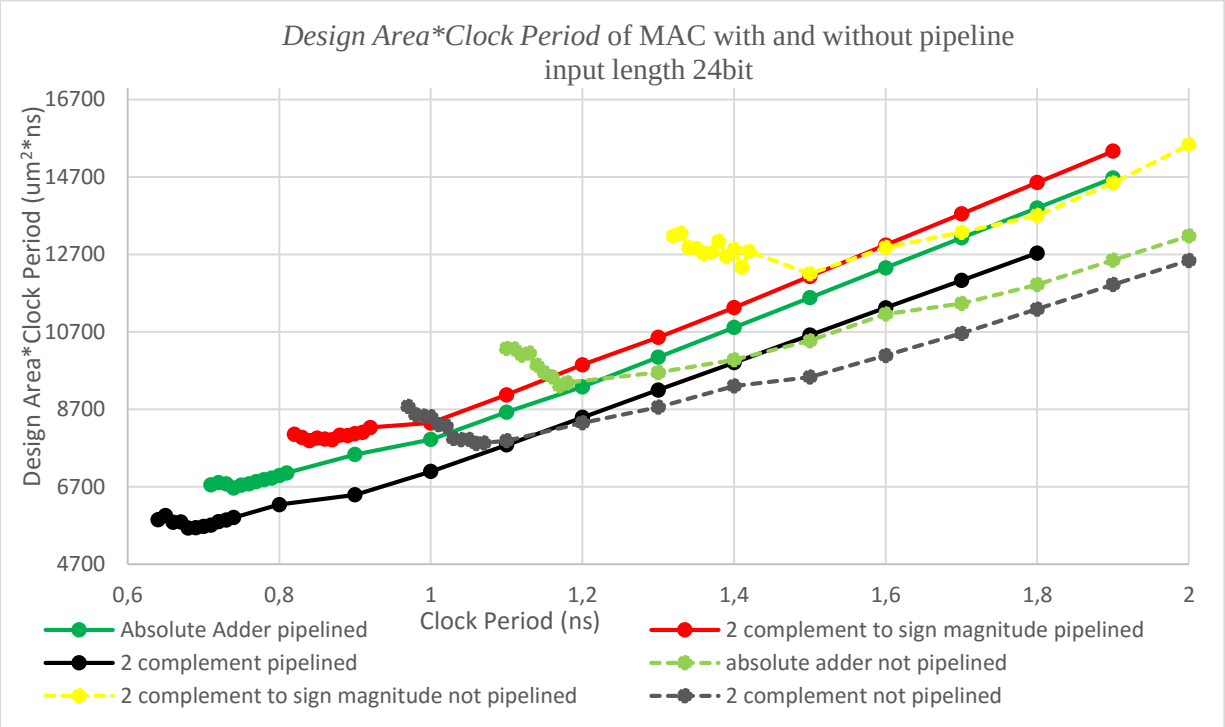


Σχήμα 4.17 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς για accumulation ενός γινομένου σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

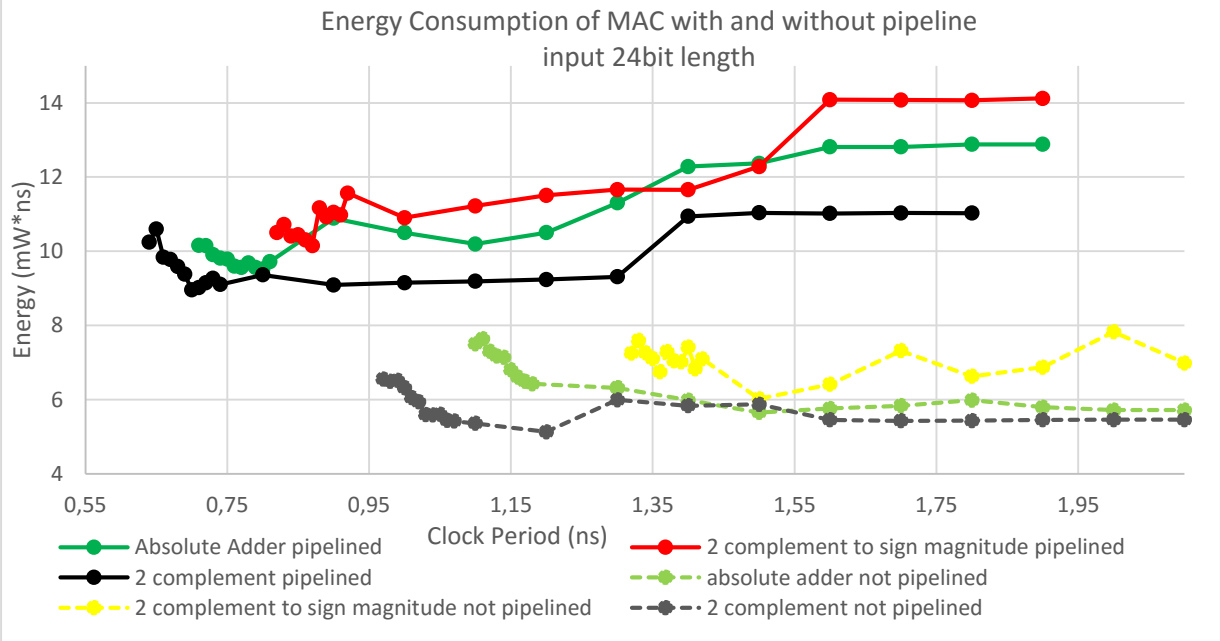


Σχήμα 4.18 Σύγκριση καταναλωμένης ενέργειας για τους διαφορετικούς σχεδιασμούς για accumulation ενός γινομένου σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

- Μέγεθος εισόδων 24bit

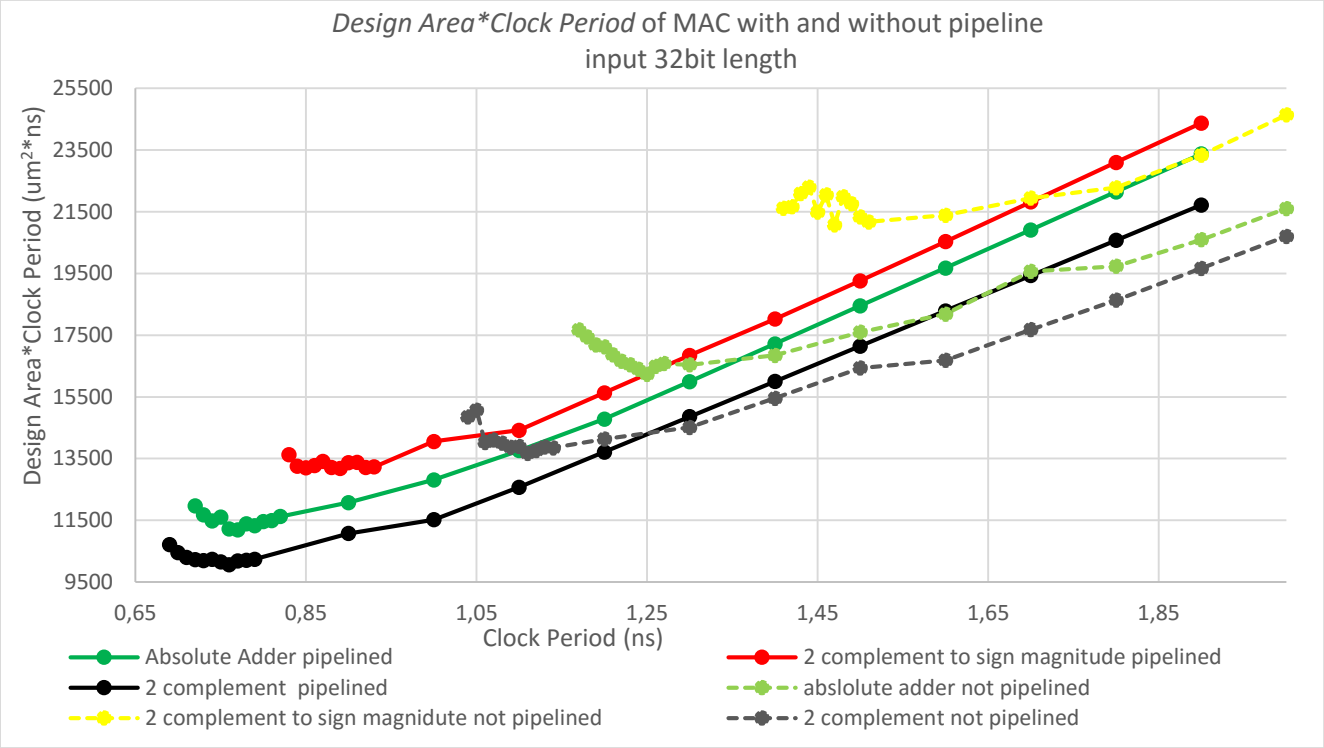


Σχήμα 4.19 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς για accumulation ενός γινομένου σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

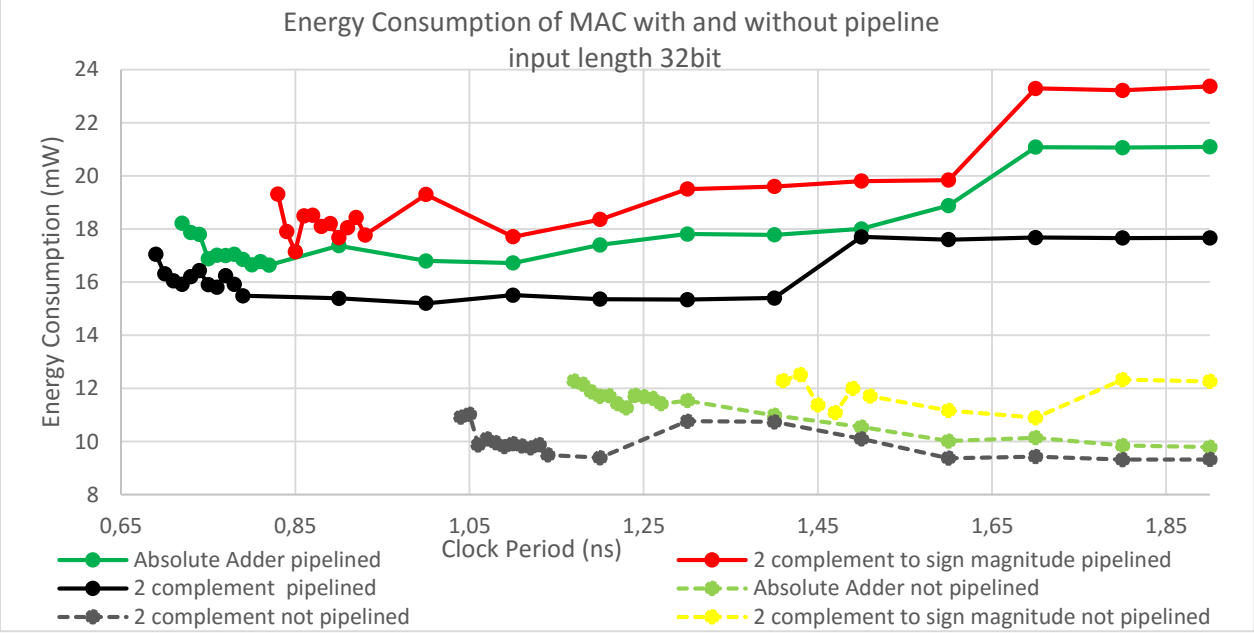


Σχήμα 4.20 Σύγκριση καταναλωμένης ενέργειας για τους διαφορετικούς σχεδιασμούς για accumulation ενός γινομένου σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

- Μέγεθος εισόδων 32bit

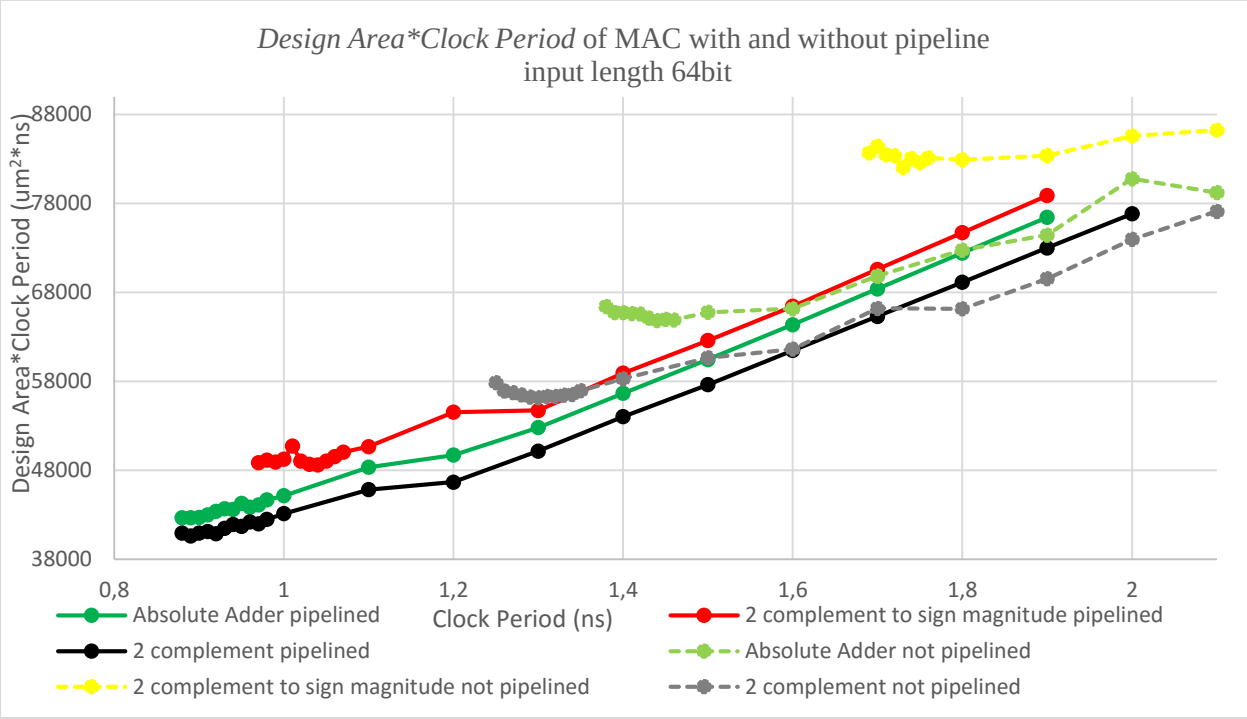


Σχήμα 4.21 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς για accumulation ενός γινομένου σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

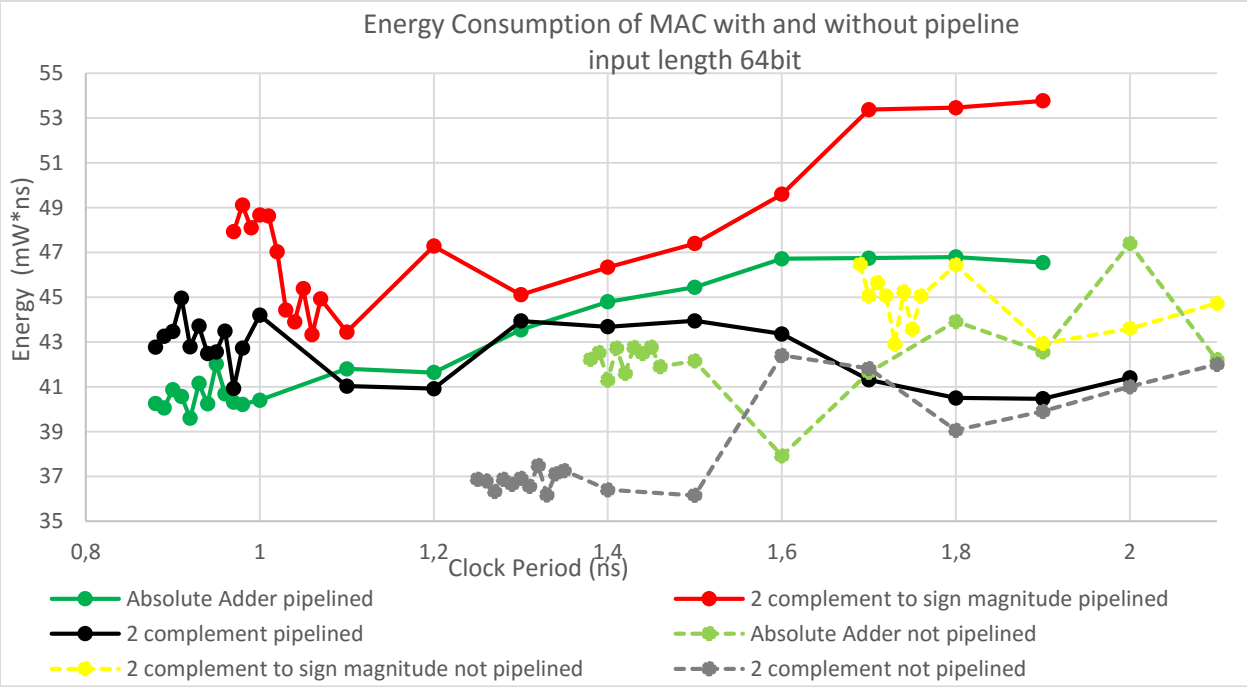


Σχήμα 4.22 Σύγκριση καταναλωμένης ενέργειας για τους διαφορετικούς σχεδιασμούς για accumulation ενός γινομένου σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

- Μέγεθος εισόδων 64bit



Σχήμα 4.23 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς για accumulation ενός γινομένου σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

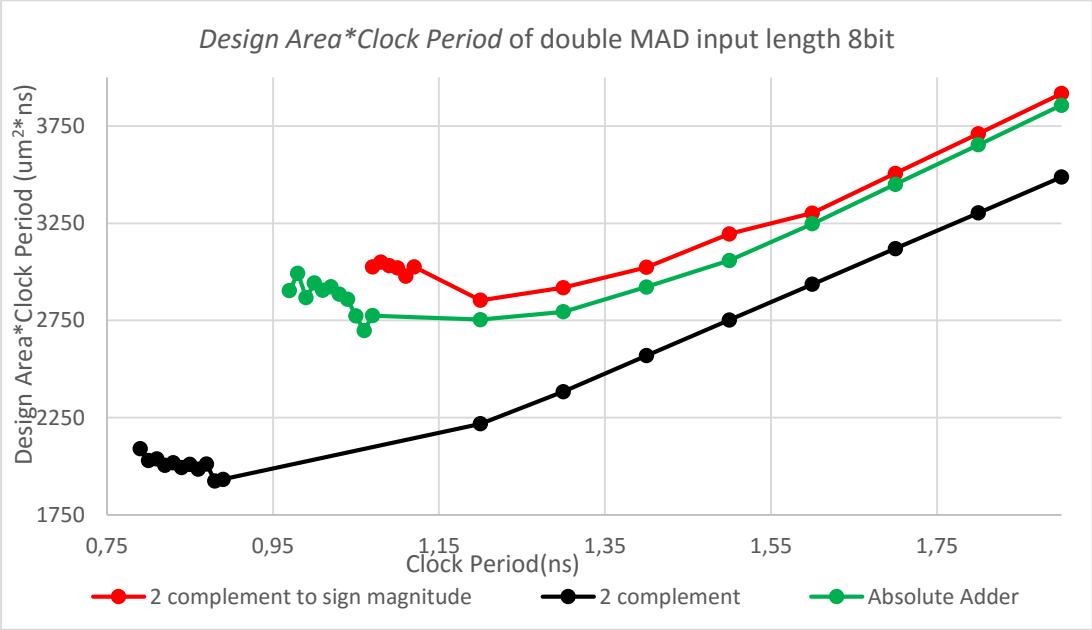


Σχήμα 4.24 Σύγκριση καταναλωμένης ενέργειας για τους διαφορετικούς σχεδιασμούς για accumulation ενός γινομένου σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

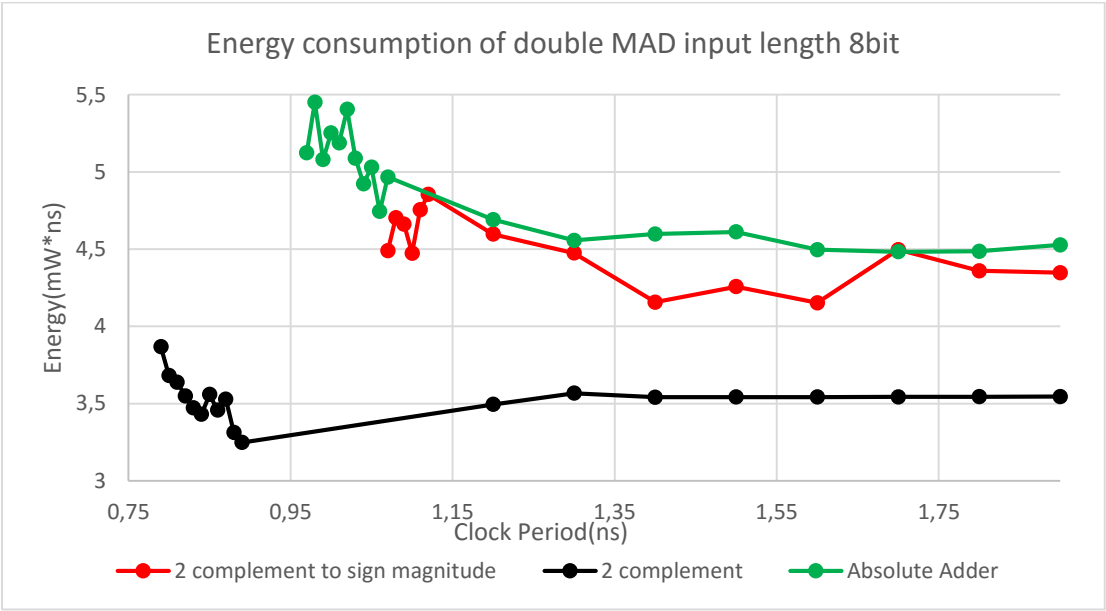
Παρατηρείται ότι η λειτουργία συνεχούς διοχέτευσης είναι καλύτερη ως προς την επιφάνεια σχεδίασης επί περίοδο ρολογιού για χαμηλές περιόδους, όσο η περίοδος αυξάνεται η λειτουργία συνεχούς διοχέτευσης γίνεται χειρότερη από τα συνδυαστικά κυκλώματα ως προς την επιφάνεια. Όσον αφορά την ενέργεια κατανάλωσης η λειτουργία συνεχούς διοχέτευσης έχει αυξημένη κατανάλωση όμως η αύξηση αυτή είναι σχετικά μικρή όσο η λειτουργία γίνεται κοντά στην ελάχιστη περίοδο. Όσο το μέγεθος της λέξης αυξάνει τόσο μειώνεται η διαφορά στην ενέργεια μεταξύ λειτουργίας συνεχούς διοχέτευσης και κανονικής λειτουργίας. Όσον αφορά τον τρόπο σχεδίασης ο αθροιστής απόλυτης τιμής εξακολουθεί να έχει πλεονέκτημα σε όλες τις παραμέτρους έναντι της χρήσης μετατροπέα, στα 64 και στα 8 bit επίσης παρατηρείται ότι η κατανάλωση ενέργειας του αθροιστή απόλυτης τιμής είναι μικρότερη (για ελάχιστο) ακόμα και το MAC επεξεργασίας αριθμών σε συμπλήρωμα ως προς δύο μόνο στα 64bit, αυτό συμβαίνει διότι τα δύο στάδια pipeline είναι περισσότερο ισοσταθμισμένα σε αυτή τη σχεδίαση γι’ αυτά τα μήκη λέξης.

Σύγκριση ως προς επιφάνεια σχεδίασης επί περίοδο ρολογιού και καταναλισκόμενη ενέργεια MAC με χρήση: αθροιστή απόλυτης τιμής, μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο, επεξεργασία αριθμών σε συμπλήρωμα ως προς δύο για 8,16,24,32,64 bit μέγεθος εισόδων.

- Μέγεθος A,X,B,Y:8 bit μέγεθος D:16 bit

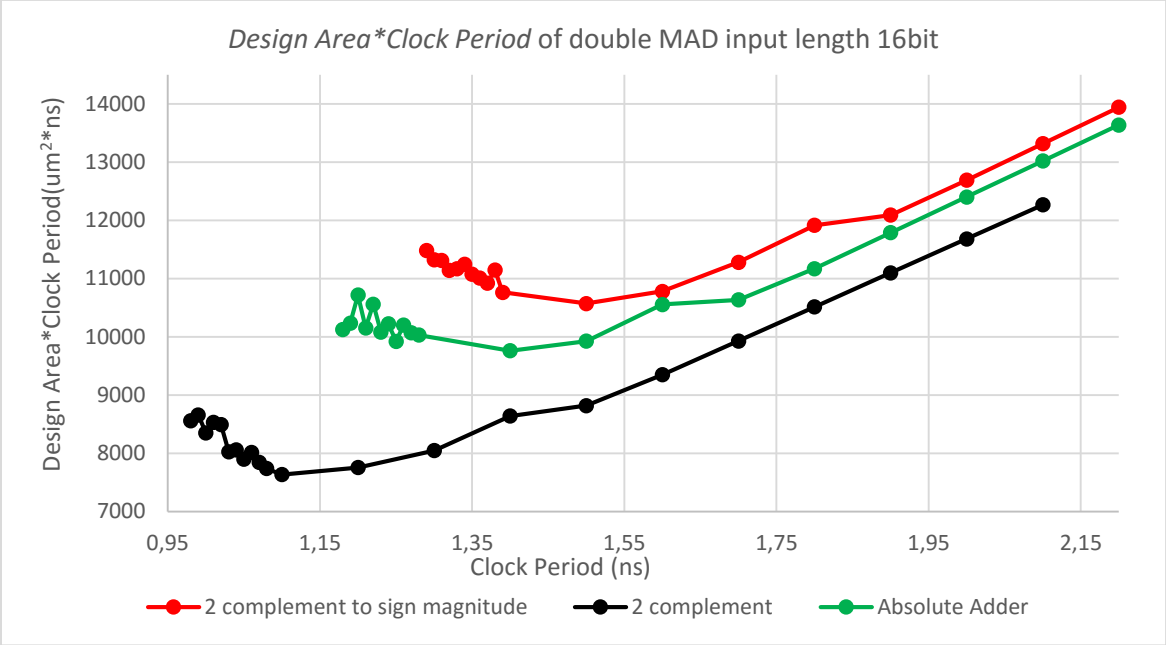


Σχήμα 4.25 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + B * Y + D$.

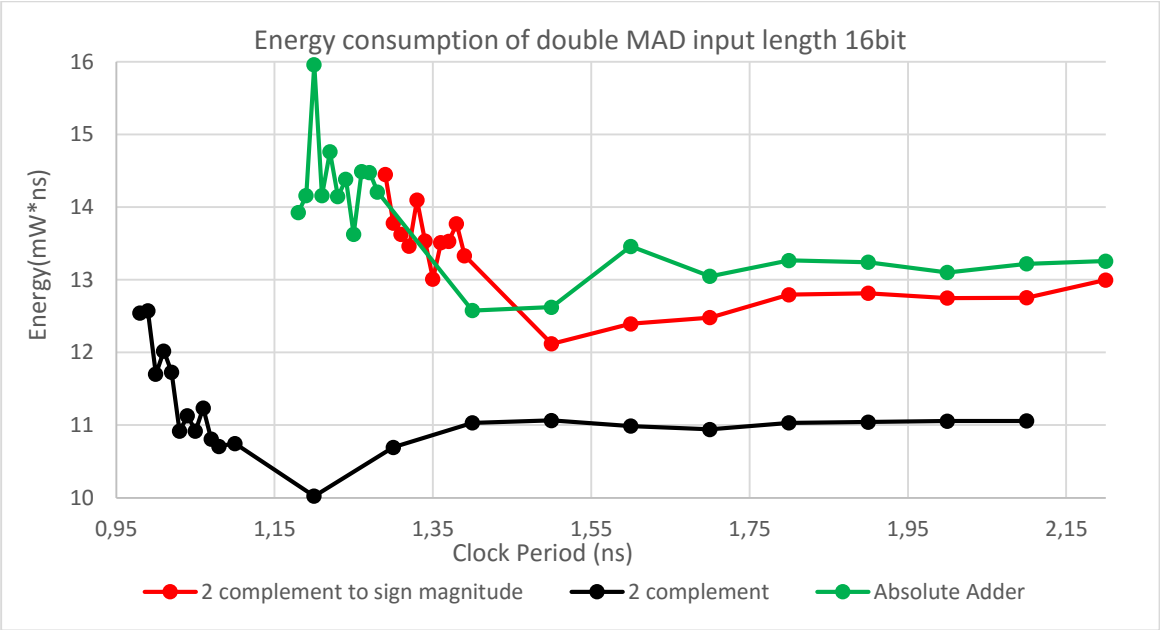


Σχήμα 4.26 Σύγκριση καταναλισκόμενης ενέργειας για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + B * Y + D$.

- Μέγεθος A,X,B,Y:16 bit μέγεθος D:32 bit

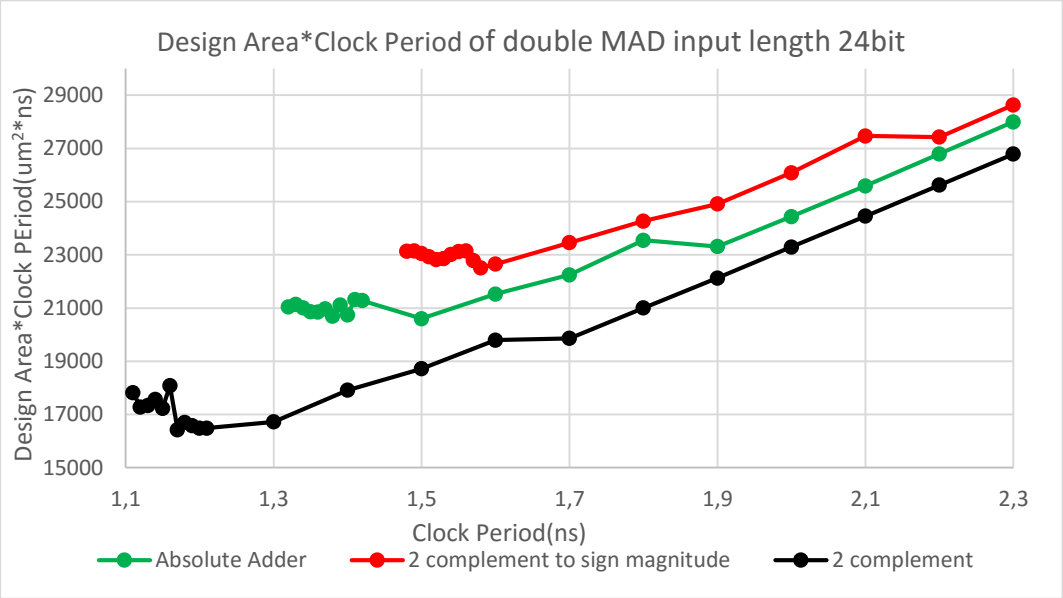


Σχήμα 4.27 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + B * Y + D$.

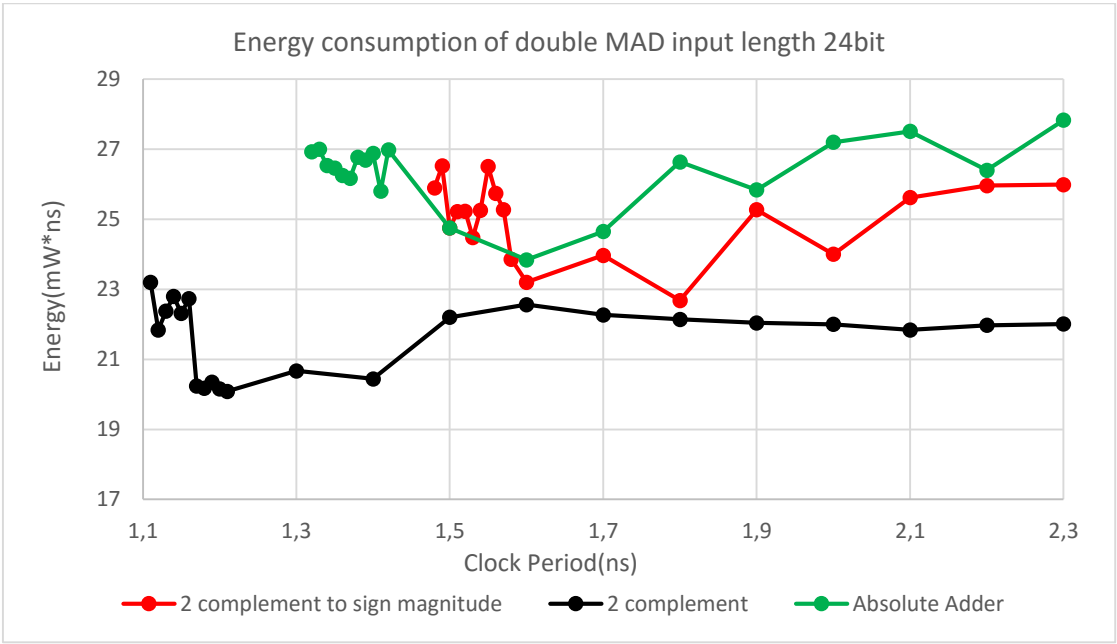


Σχήμα 4.28 Σύγκριση καταναλισκόμενης ενέργειας για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + B * Y + D$.

- Μέγεθος A,X,B,Y:24 bit μέγεθος D:48 bit

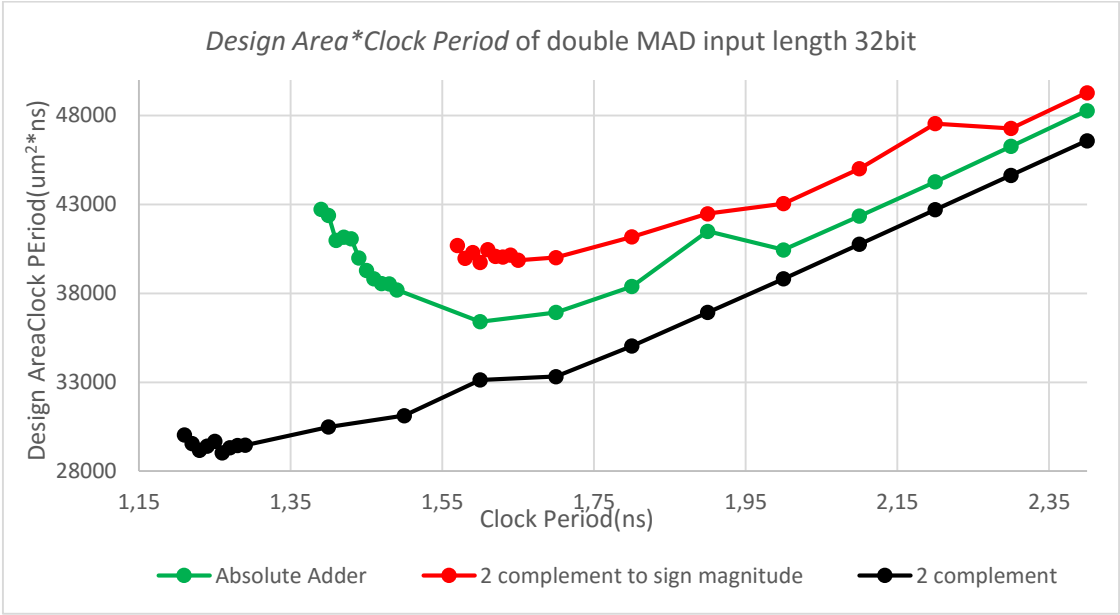


Σχήμα 4.29 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + B * Y + D$.

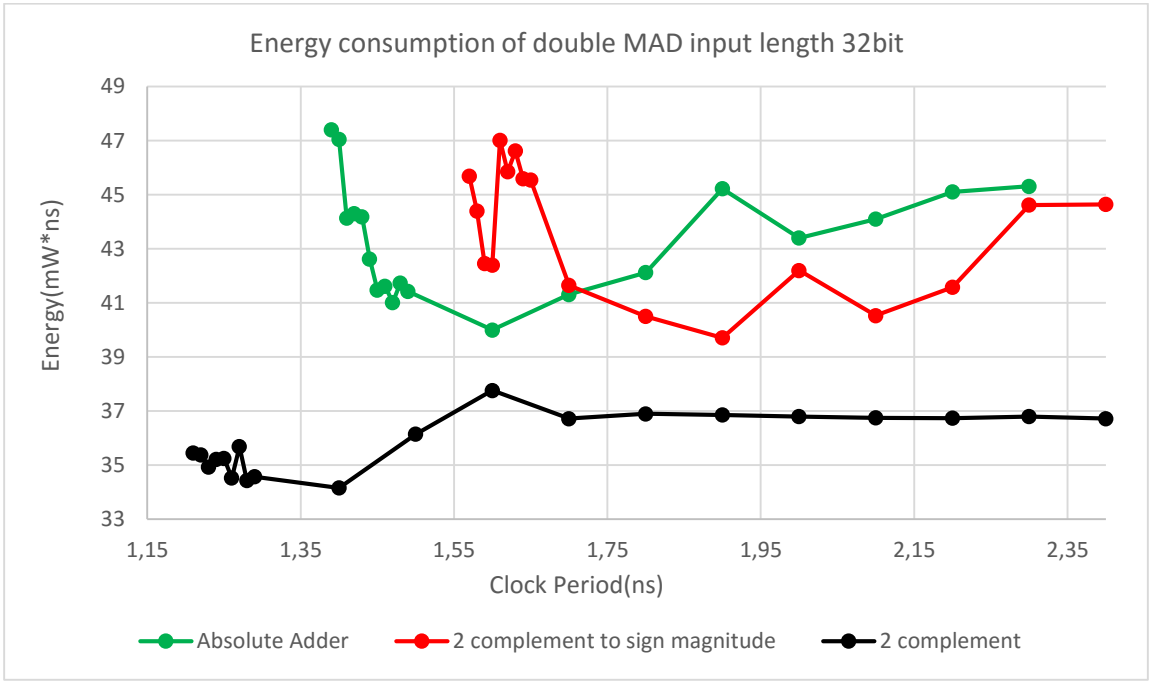


Σχήμα 4.30 Σύγκριση καταναλισκόμενης ενέργειας για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + B * Y + D$.

- Μέγεθος A,X,B,Y:32 bit μέγεθος D:64 bit

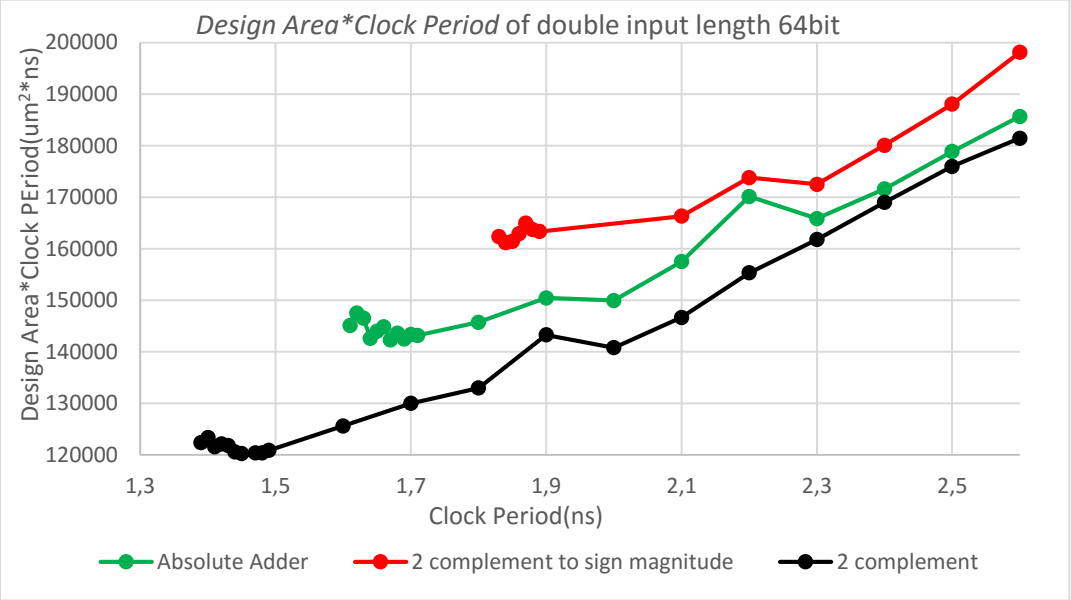


Σχήμα 4.31 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + B * Y + D$.

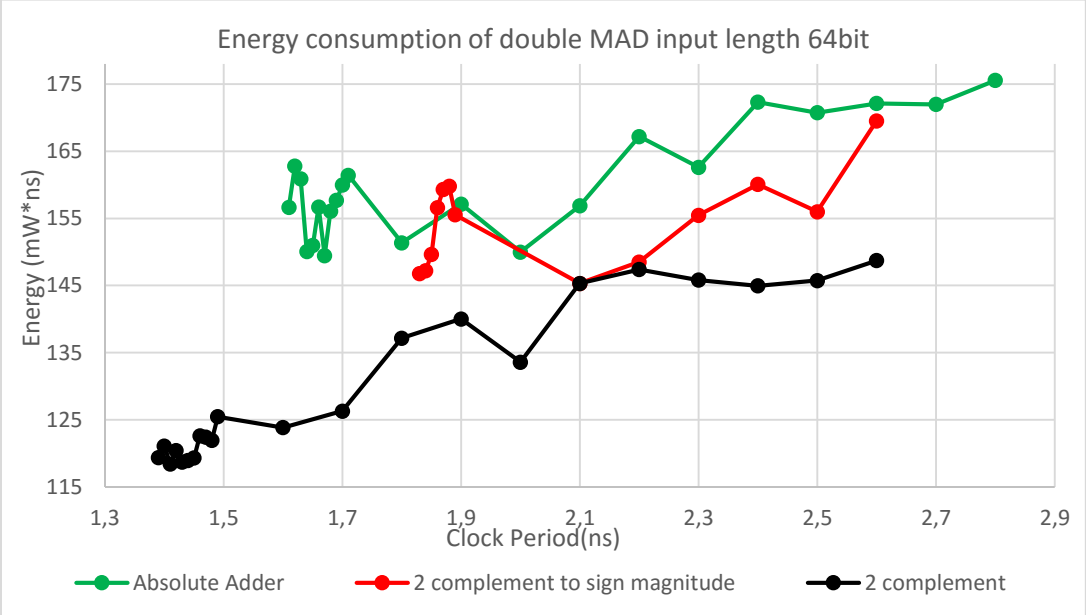


Σχήμα 4.32 Σύγκριση καταναλισκόμενης ενέργειας για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + B * Y + D$.

Μέγεθος A,X,B,Y:64 bit μέγεθος D:128 bit



Σχήμα 4.33 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + B * Y + D$.

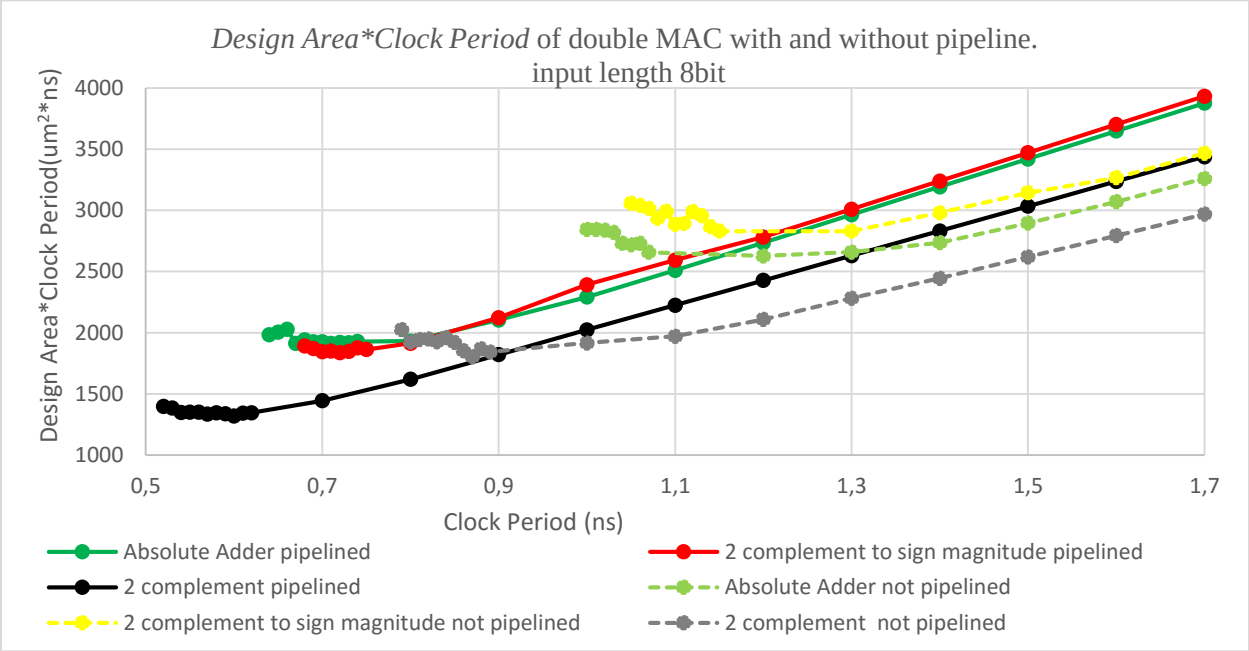


Σχήμα 4.34 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς υλοποίησης της αριθμητικής πράξης $A * X + B * Y + D$.

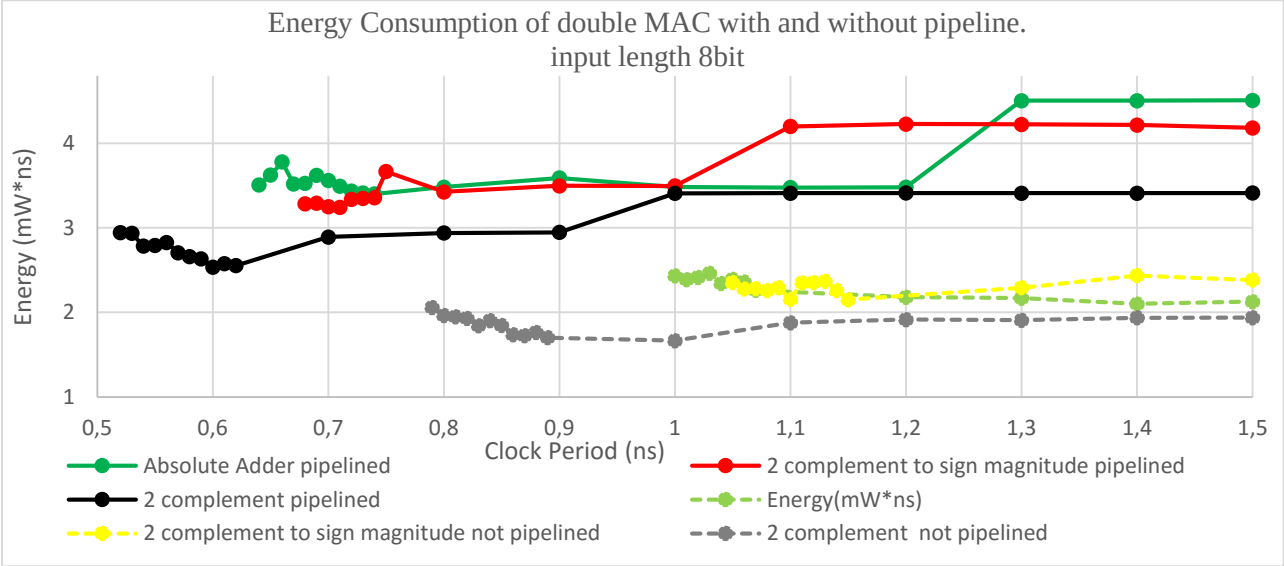
Παρατηρείται ότι η σχεδίαση με αθροιστή με αθροιστή απόλυτης τιμής εξακολουθεί να υπερτερεί ως προς την μετρική *area*×*delay* σε όλα τα μήκη λέξης. Όσον αφορά την κατανάλωση ενέργειας η σχεδίαση με μετατροπέα παρουσιάζει καλύτερα αποτελέσματα, αυτό οφείλεται στο επιπλέον μερικό γινόμενο που χρειάζεται η χρήση με αθροιστή απόλυτης τιμής το οποίο αποτελείται από πολλές μονάδες, όσο το μέγεθος της λέξης αυξάνεται η *area*×*delay* του αθροιστή πλησιάζει την λειτουργία του συμπληρώματος ως προς δύο ενώ η κατανάλωση ενέργειας των δύο σχεδιασμών παρουσιάζει παρόμοια αποτελέσματα μεταξύ των δύο συνδυασμών.

Σύγκριση σχεδίασης MAC για accumulation δύο γινομένων δηλαδή για εκτέλεση της αριθμητικής πράξης $(\sum A_i * X_i + B_i Y_i)$ με λειτουργία συνεχούς διοχέτευσης σε δύο κύκλους ρολογιού και σε λειτουργία σε ένα κύκλο ρολογιού για υλοποιήσεις με χρήση: αθροιστή απόλυτης τιμής, μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο και επεξεργασία με αριθμούς σε συμπλήρωμα ως προς δύο.

- Μέγεθος εισόδων: 8bit

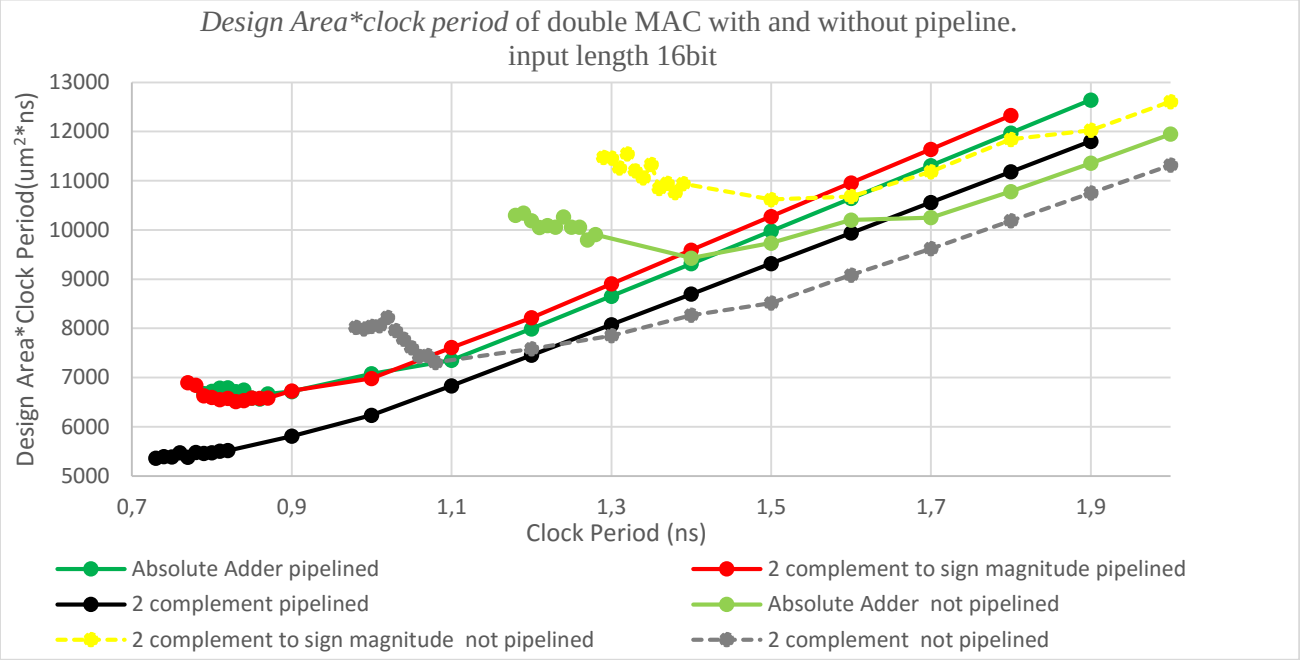


Σχήμα 4.35 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς για accumulation δύο γινομένων σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

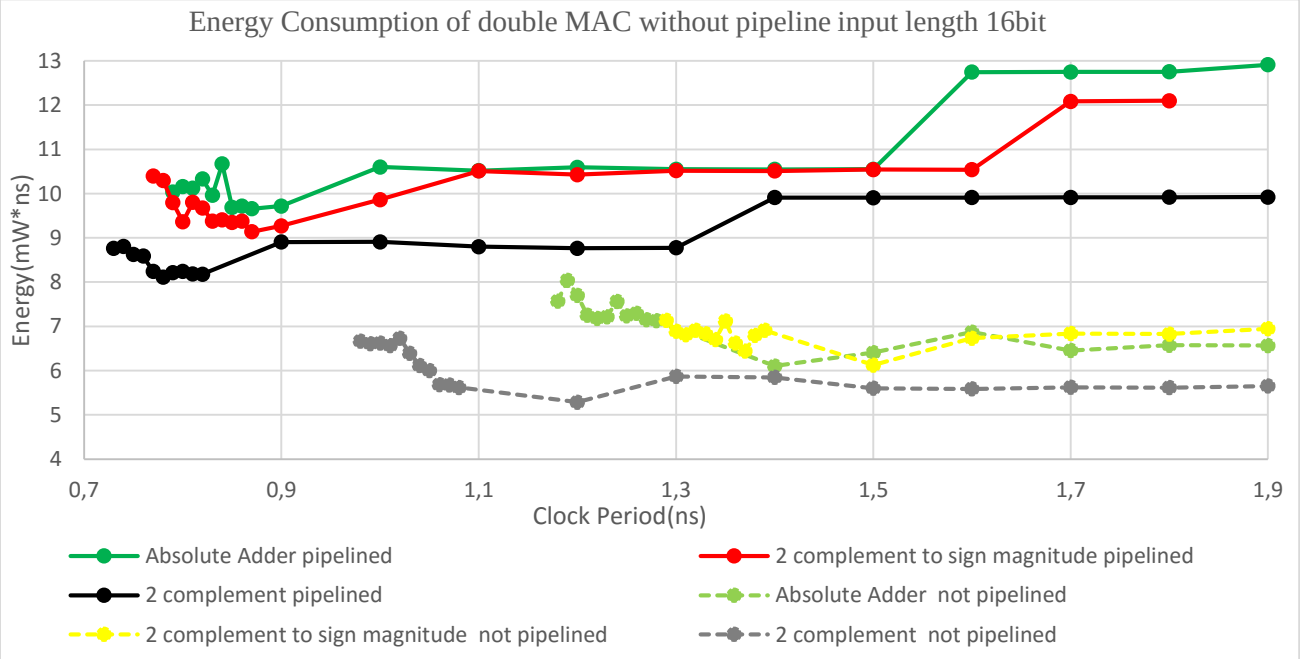


Σχήμα 4.36 Σύγκριση καταναλωμένης ενέργειας για τους διαφορετικούς σχεδιασμούς για accumulation δύο γινομένων σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

- Μέγεθος εισόδων: 16bit

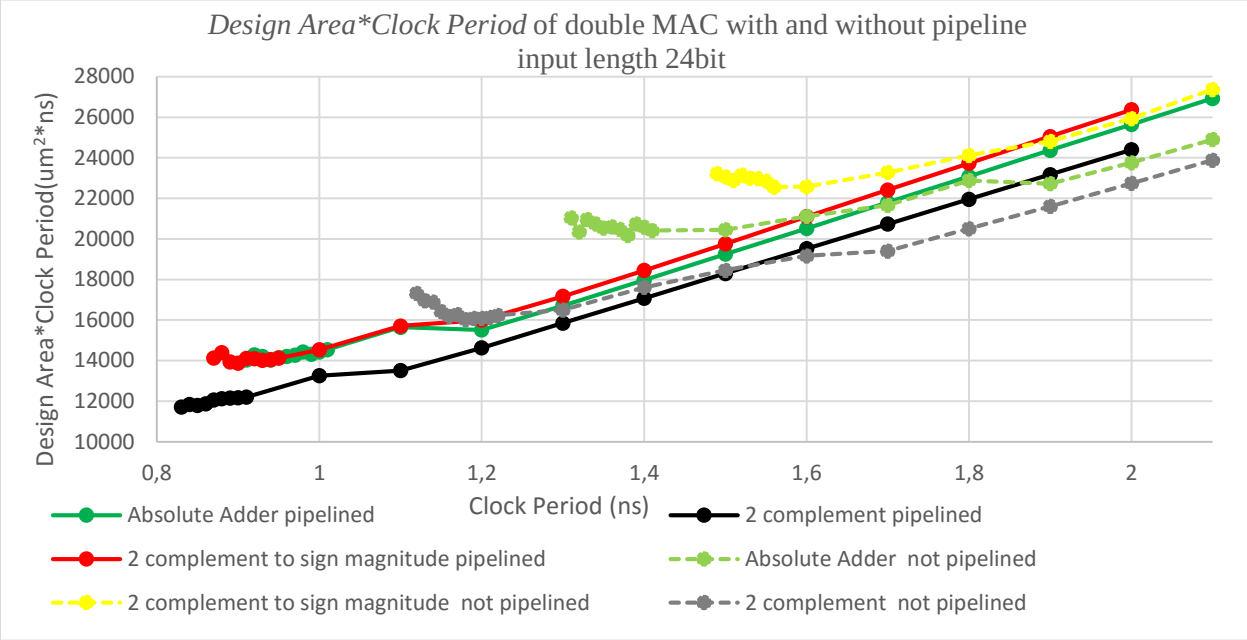


Σχήμα 4.37 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς για accumulation δύο γινομένων σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

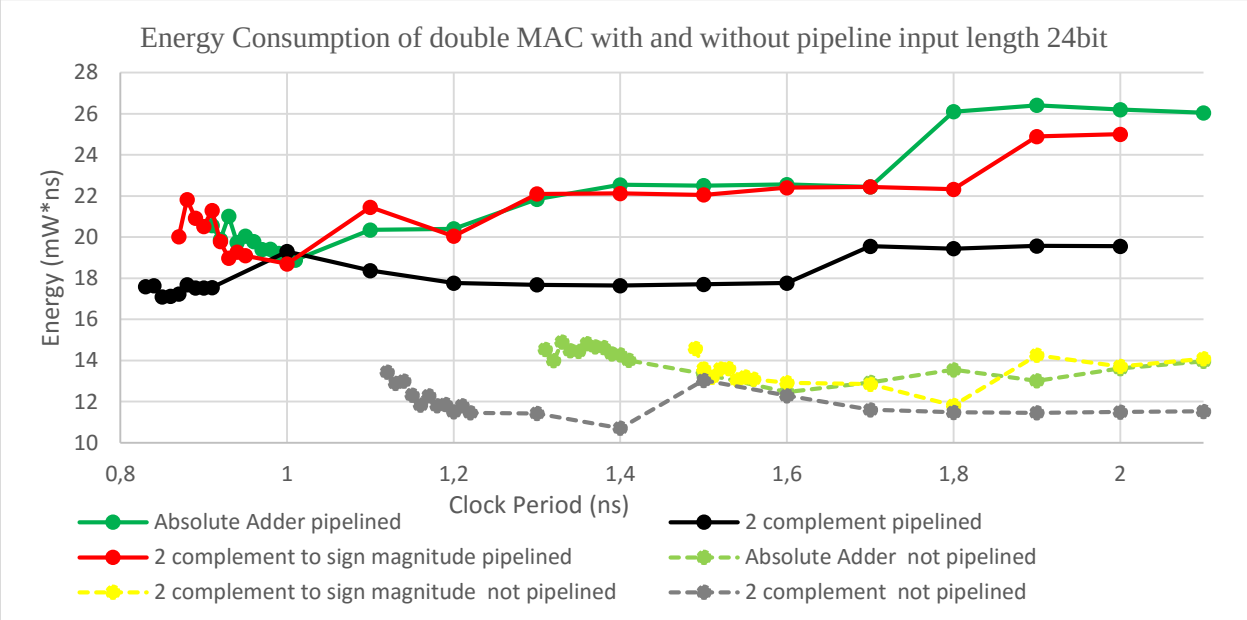


Σχήμα 4.38 Σύγκριση καταναλωμένης ενέργειας για τους διαφορετικούς σχεδιασμούς για accumulation δύο γινομένων σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

- Μέγεθος εισόδων: 24bit

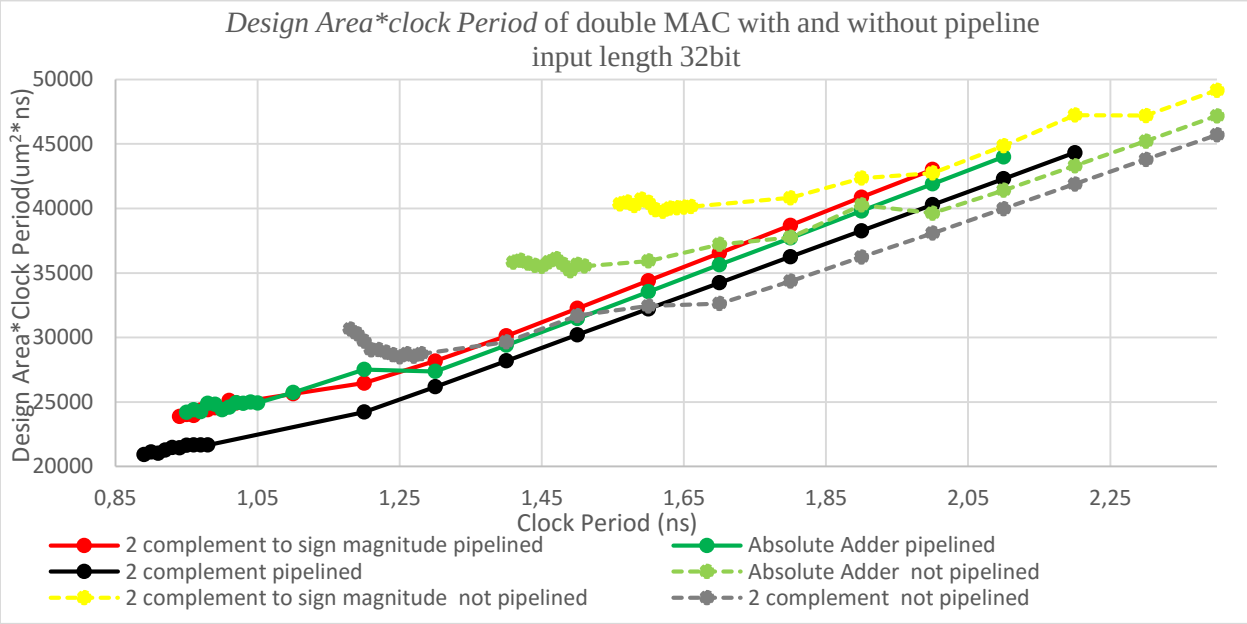


Σχήμα 4.39 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς για accumulation δύο γινομένων σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

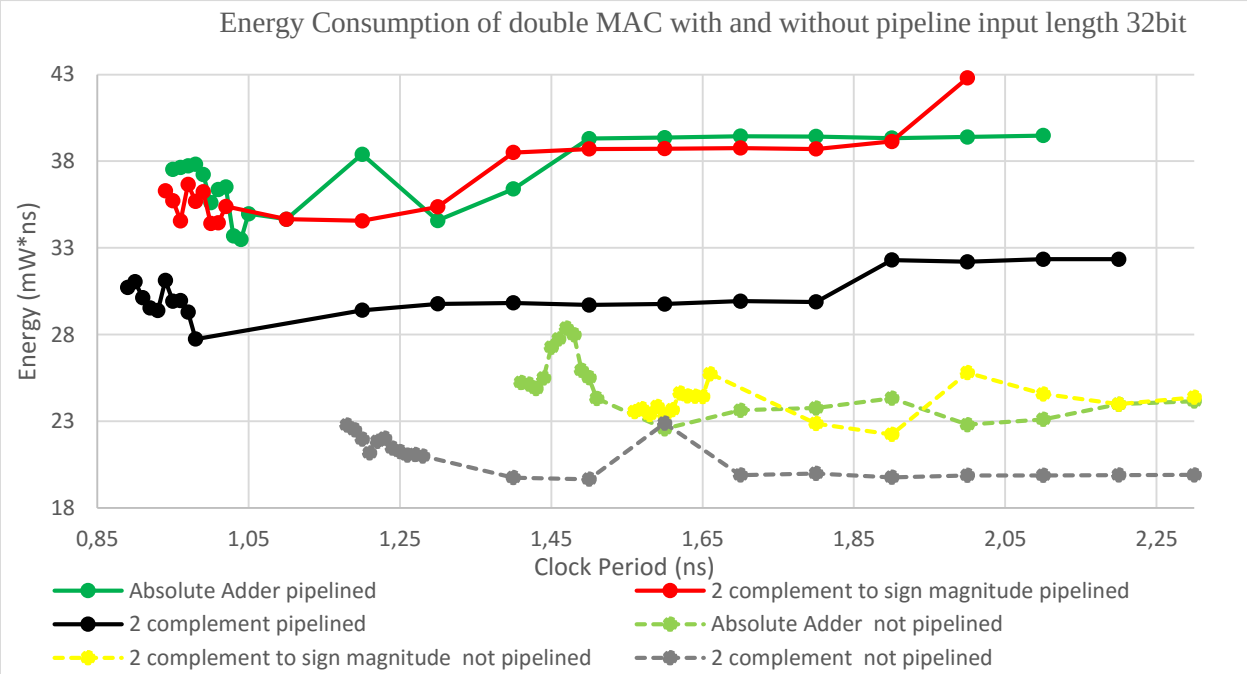


Σχήμα 4.40 Σύγκριση καταναλωμένης ενέργειας για τους διαφορετικούς σχεδιασμούς για accumulation δύο γινομένων σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

- Μέγεθος εισόδων: 32bit

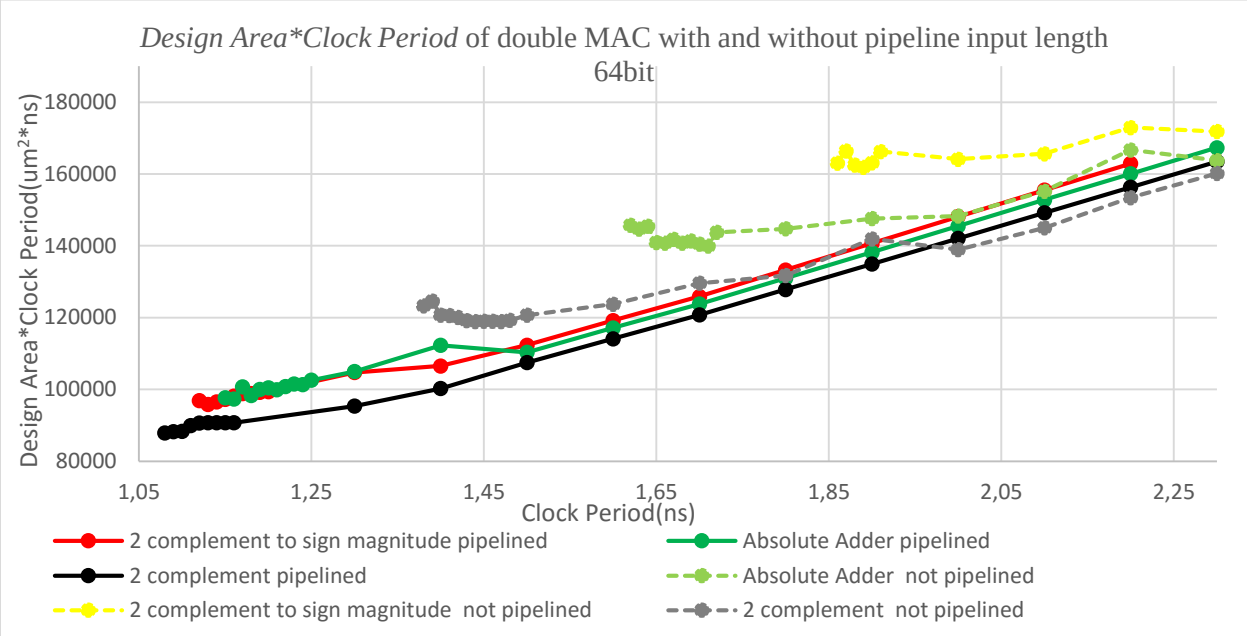


Σχήμα 4.41 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς για accumulation δύο γινομένων σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

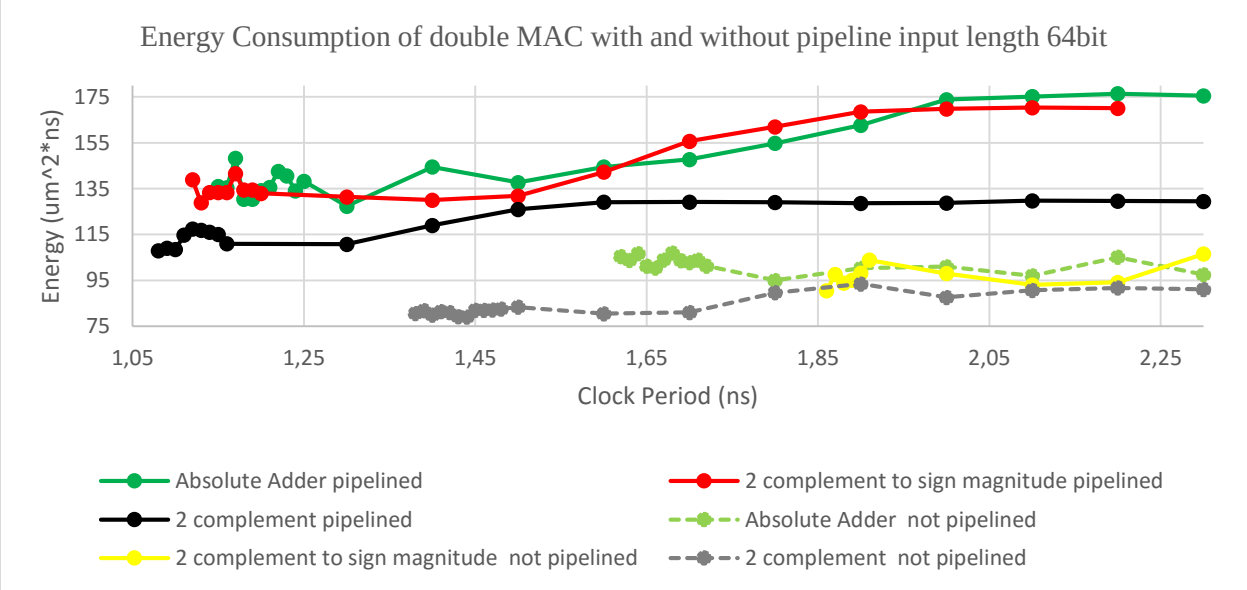


Σχήμα 4.42 Σύγκριση καταναλωμένης ενέργειας για τους διαφορετικούς σχεδιασμούς για accumulation δύο γινομένων σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

- Μέγεθος εισόδων: 64bit



Σχήμα 4.43 Σύγκριση επιφάνειας σχεδίασης επί περίοδο ρολογιού για τους διαφορετικούς σχεδιασμούς για accumulation δύο γινομένων σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.



Σχήμα 4.44 Σύγκριση καταναλωμένης ενέργειας για τους διαφορετικούς σχεδιασμούς για accumulation δύο γινομένων σε λειτουργία συνεχούς διοχέτευσης και λειτουργία σε ένα κύκλο.

Παρατηρείται ότι όσον αφορά το *area*×*delay* τα αποτελέσματα εξακολουθούν να είναι ίδια δηλαδή η σχεδίαση με αθροιστή απόλυτης τιμής είναι έχει μικρότερη επιφάνεια από την σχεδίαση με μετατροπέα, όσον αφορά την ενέργεια τα αποτελέσματα είναι παρόμοια και δεν παρατηρείται σημαντική βελτίωση, Επίσης επαληθεύεται ξανά ότι η λειτουργία συνεχούς διοχέτευσης παρουσιάζει καλύτερη συμπεριφορά όσο αφορά την επιφάνεια σχεδίασης για περίοδο ρολογιού κοντά στην ελάχιστη δυνατή, όσον αφορά την ενέργεια η λειτουργία συνεχούς διοχέτευσης εμφανίζει ελαφρά μεγαλύτερη κατανάλωση λόγω των επιπλέον ενδιάμεσων καταχωρητών που χρησιμοποιεί.

Για την λειτουργία συνεχούς διοχέτευσης με διαχωρισμό του Wallace σε δύο ξεχωριστά δέντρα έγινε μελέτη μόνο ως τη μείωση του ελάχιστου μονοπατιού αφού όπως αναφέρθηκε και στο προηγούμενο κεφάλαιο δεν αποτελεί βέλτιστη λύση ως προς την επιφάνεια και την κατανάλωση ενέργειας.

5 ΣΥΜΠΕΡΑΣΜΑΤΑ ΚΑΙ ΕΠΕΚΤΑΣΕΙΣ

5.1 Σύνοψη και συμπεράσματα.

Συνοψίζοντας στην παρούσα εργασία μελετήθηκε ο σχεδιασμός multiplier adder/accumulator για συγκεκριμένες εφαρμογές που χρησιμοποιούν αριθμητική πρόσημου-μέτρου, παρόλο που η αναπαράσταση σε μορφή συμπληρώματος ως προς δύο εμφανίζει καλύτερα χαρακτηριστικά, υπάρχουν εφαρμογές όπου ο σχεδιαστής είναι υποχρεωμένος να αναπαραστήσει τους αριθμούς σε sign-magnitude μορφή. Από τα πειραματικά αποτελέσματα των συνθέσεων προκύπτουν, τα εξής συμπεράσματα:

- Αν η εφαρμογή χρήσης MAD/MAC υπολογίζει ένα γινόμενο τότε η χρήση αθροιστή απόλυτης τιμής παρουσιάζει καλύτερη συμπεριφορά όσον αφορά το *area*×*delay, critical path* και *energy consumption*, σε σχέση με τη χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο.
- Στις εφαρμογές οι οποίες χρησιμοποιούν αριθμούς μεγάλου μήκους η βελτίωση είναι μεγαλύτερη αφού η μετατροπή από συμπλήρωμα ως προς δύο σε πρόσημο μέτρο επιβαρύνει σημαντικά το συνολικό κύκλωμα σε όλα τα παραπάνω χαρακτηριστικά.
- Η λειτουργία συνεχούς διοχέτευσης εφαρμόζεται με κύριο σκοπό την μείωση της περιόδου του ρολογιού και όσο αυξάνεται η περίοδος τόσο μειώνεται η απόδοσή της.
- Η λειτουργίας συνεχούς διοχέτευσης (κοντά στην ελάχιστη δυνατή περίοδο) παρουσιάζει βελτίωση στο *critical path* και στο *area*×*delay* ενώ αυξάνει το *energy consumption*.
- Σε εφαρμογές οι οποίες υπολογίζουν δύο γινόμενα (double MAD/MAC) αν δεν είναι ανάγκη η λειτουργία σε υψηλές συχνότητες τότε η χρήση αθροιστή απόλυτης τιμής θα παρουσιάσει βελτίωση στο *critical path* και *area*×*delay* σε σχέση με τη χρήση μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο.
- Για τον υπολογισμό γινομένων περισσότερων των δύο, ο σχεδιασμός με χρήση αθροιστή απόλυτης τιμής γίνεται αρκετά περίπλοκος αφού γίνεται πιο σύνθετη η δημιουργία των σημάτων αντιστροφής των γινομένων.
- Για λειτουργία συνεχούς διοχέτευσης σε double MAC όσον αφορά το critical path προτείνεται ο διαχωρισμός του Wallace σε δύο κομμάτια, το πρώτο να μετατρέπει τα μερικά γινόμενα σε τέσσερα ισοδύναμα και το δεύτερο τα πέντε σε δύο. Όσον αφορά τον τρόπο σχεδίασης, η χρήση αθροιστή απόλυτης τιμής εξακολουθεί να παρουσιάζει καλύτερα αποτελέσματα ως προς το *area*×*delay* ενώ η κατανάλωση ενέργειας είναι παρόμοια με εκείνη της χρήσης μετατροπέα από συμπλήρωμα ως προς δύο σε πρόσημο-μέτρο.
- Και οι δύο τρόποι σχεδίασης φαίνεται ότι είναι χειρότεροι προς όλες τις μετρικές σε σχέση με την λειτουργία των αντίστοιχων κυκλωμάτων τα οποία επεξεργάζονται αριθμούς σε συμπλήρωμα ως προς δύο.

5.2 Μελλοντικές επεκτάσεις

- Χρήση των MAC που σχεδιάστηκαν σε φίλτρα επεξεργασίας σημάτων με πολλές μεταβολές γύρω από το μηδέν, και σύγκριση των αποτελεσμάτων κυρίως όσον αφορά την ενέργεια κατανάλωσης με φίλτρα που επεξεργάζονται αριθμούς σε συμπλήρωμα ως προς δύο.
- Εφαρμογή του σχεδιασμένου αθροιστή απόλυτης τιμής στην κατασκευή MAC για επεξεργασία αριθμών σε floating point αριθμητική.
- Μελέτη των σχεδιασμένων κυκλωμάτων σε άλλες τεχνολογίες εκτός των 65nm.
- Χρήση των κυκλωμάτων που σχεδιάστηκαν σε εφαρμογές που υπολογίζουν απόλυτες τιμές, αφού η απόλυτη τιμή ταυτίζεται με το μέτρο του αποτελέσματος.
- Χρήση του αθροιστή απόλυτης τιμής που σχεδιάστηκε σε αντίστοιχα κυκλώματα για υπολογισμό περισσότερων των δύο γινομένων.
- Υλοποίηση των κυκλωμάτων που αναφέρθηκαν σε τεχνολογία FPGA.

6 ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] M. LU, Arithmetic and logic in computer Systems, 2010.

[2] K. Pekmestzi, Digital VLSI Systems. NTUA Lecture Notes, 2003.

[3] J. DODNSON, "Low Power area-efficient, absolute value arithmetic unit". Patent WO 93/24880, 1993.

[4] ModelSim Corporation, [Online]. Available: www.modelsim.com.

[5] Synopsys Primetime PX, [Online]. Available: www.Synopsys.com.

[6] "Synopsys," DW02_tree, [Online]. Available: www.synopsys.com/dw/ipdir.php?c=DW02_tree.

[7] "Synopsys Design Compiler," [Online]. Available: www.Synopsys.com.