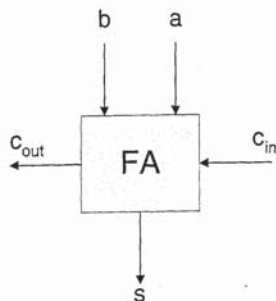


## 1.2 Αθροιστές

### 1.2.1 Βασική δομική μονάδα-Υλοποίηση σε VLSI

Το βασικό δομικό κύτταρο κάθε αριθμητικού κυκλώματος είναι ο πλήρης αθροιστής (Full Adder: FA). Για να αθροίσουμε δύο δυαδικούς αριθμούς, εκτελούμε την πράξη της πρόσθεσης ανάμεσα στα δυαδικά ψηφία ίδιας αξίας και στο κρατούμενο από την προηγούμενη βαθμίδα. Ο γνωστός αλγόριθμος πρόσθεσης δυαδικών αριθμών υλοποιείται με την επανάληψη σε μορφή αλυσίδας μιας βασικής δομικής μονάδας δηλαδή του πλήρη αθροιστή που εκτελεί την προηγούμενη πράξη. Σύμφωνα με τα παραπάνω, το κύτταρο του πλήρη αθροιστή θα πρέπει να δέχεται δύο σήματα από τους προστιθέμενους δυαδικούς αριθμούς και ένα κρατούμενο από τον προηγούμενο πλήρη αθροιστή, σύνολο τρεις εισόδους. Το κρατούμενο στην πρώτη βαθμίδα θα είναι στο λογικό 0. Αφού το κάθε κύτταρο δέχεται ένα bit από κάθε δυαδικό αριθμό, είναι εύλογο ότι θα χρειαζομαστε τόσα κύτταρα, όσα είναι τα bits των αριθμών. Όσον αφορά τις εξόδους, χρειαζόμαστε μια γραμμή για το αποτέλεσμα-άθροισμα των δύο bits και άλλη μια γραμμή για το κρατούμενο που ενδεχομένως να παράγεται, σύνολο δύο εξόδους. Συνεπώς, η βασική δομική μονάδα θα πρέπει να έχει τη μορφή του σχήματος 1.1.



Σχήμα 1.1 Είσοδοι-εξόδοι της βασικής δομικής μονάδας του πλήρη αθροιστή

## 1.2 Αθροιστές

Όπου  $a$  και  $b$  είναι τα bits των προστιθέμενων αριθμών,  $c_{in}$  και  $c_{out}$  τα κρατούμενα εισόδου και εξόδου αντίστοιχα και  $s$  το άθροισμα των δύο bits.

Ας εξετάσουμε τώρα την εσωτερική δομή του κυττάρου. Θα πρέπει να βρούμε τις λογικές σχέσεις με τις οποίες συνδέονται οι εισόδους και οι εξόδοι του σχήματος 1.1. Το άθροισμα σχηματίζεται εκτελώντας την λογική πράξη XOR ανάμεσα στα bits των δύο αριθμών και στο κρατούμενο από την προηγούμενη βαθμίδα δηλαδή:  $s = a \oplus b \oplus c_{in}$ . Κρατούμενο εξόδου παράγεται όταν τα bits που προσθέτονται είναι και τα δύο στο λογικό 1 ή όταν ένα από τα δύο bits και το κρατούμενο εισόδου είναι στο λογικό 1. Η αλγεβρική μετάφραση των παραπάνω παρατηρήσεων είναι η εξής:  $c_{out} = ab + ac_{in} + bc_{in}$ .

Στα ίδια αποτελέσματα καταλήγουμε και με τη δημιουργία του πίνακα αληθείας του κυκλώματος.

Πίνακας 1.1 Πίνακας αληθείας πλήρη αθροιστή

a	b	$c_{in}$	s	$c_{out}$
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Με την βοήθεια των πινάκων Καρνώ εξάγουμε τις αλγεβρικές σχέσεις ανάμεσα στα σήματα εισόδου και εξόδου.

		ab				
		00	01	11	10	
S:	$c_{in}$	0	1	0	1	
	1	1	0	1	0	

$$s = c_{in}ab + a\bar{c}_{in}\bar{b} + b\bar{a}\bar{c}_{in} + c_{in}\bar{a}\bar{b}$$
  

		ab				
		00	01	11	10	
$C_{out}$ :	$c_{in}$	0	0	1	0	
	1	0	1	1	1	

$$c_{out} = c_{in}b + ab + ac_{in}$$

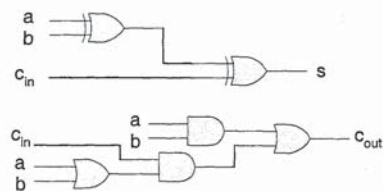
Σχήμα 1. 2 Πίνακες Καρνώ για το άθροισμα και το κρατούμενο

Παρατηρούμε ότι η συνάρτηση s μπορεί να απλοποιηθεί όπως φαίνεται παρακάτω:

$$s = c_{in}ab + a\bar{c}_{in}\bar{b} + b\bar{a}\bar{c}_{in} + c_{in}\bar{a}\bar{b} \Rightarrow s = c_{in}(ab + \bar{a}\bar{b}) + \bar{c}_{in}(a\bar{b} + \bar{a}b) \Rightarrow$$

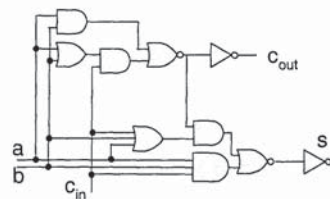
$$s = c_{in}(a \oplus b) + \bar{c}_{in}(a \oplus b) \Rightarrow s = c_{in} \oplus a \oplus b$$

Έχοντας καταλήξει στην δομή του κυκλώματος μπορούμε να περάσουμε σε ένα επίπεδο χαμηλότερο από αυτό του σχήματος 1.1, πράγμα που παρουσιάζει ιδιαίτερο ενδιαφέρον αφού μπορούμε με διαφορετικές τεχνικές να επιτύχουμε μεγαλύτερες ή μικρότερες καθυστερήσεις ανάλογα με τις απαιτήσεις του κάθε κυκλώματος. Στο σχήμα 1.3 φαίνεται ο πλήρης αθροιστής υλοποιημένος σε επίπεδο πυλών.



Σχήμα 1.3 Πλήρης αθροιστής σε επίπεδο πυλών

Με προσεκτικότερη μελέτη των αλγεβρικών σχέσεων του πλήρη αθροιστή, παρατηρούμε ότι είναι δυνατόν να αναπροσαρμόσουμε την διάταξη των πυλών διατηρώντας την ορθότητα λειτουργίας του κυκλώματος μειώνοντας παράλληλα τον αριθμό των λογικών πυλών. Το νέο κύκλωμα φαίνεται στο σχήμα 1.4.

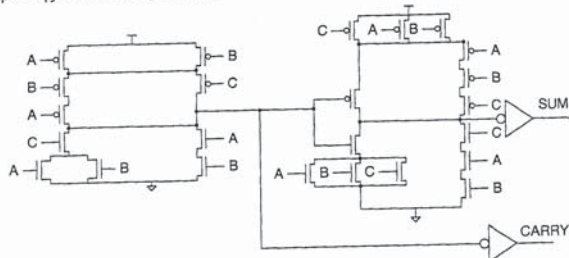


Σχήμα 1.4 Αναπροσαρμοσμένο κύκλωμα πλήρη αθροιστή σε επίπεδο πυλών.

Παρατηρούμε ότι η υλοποίηση του σχήματος 1.4 δεν συμφωνεί τελείως με τις αλγεβρικές σχέσεις που αποδείχτηκαν παραπάνω. Ο τρόπος αυτός προκύπτει από την διαπίστωση ότι ένα τμήμα του κυκλώματος που δίνει το κρατούμενο μπορεί να χρησιμοποιηθεί και για το άθροισμα, παρατήρηση που αποδεικνύεται εύκολα και αλγεβρικά:

$$\begin{aligned}
s &= a_i b_j c_{in} + a_i \bar{b}_j \bar{c}_{in} + \bar{a}_i \bar{b}_j c_{in} + \bar{a}_i b_j \bar{c}_{in} \Rightarrow \\
s &= a_i b_j c_{in} + a_i \bar{b}_j \bar{c}_{in} + \bar{a}_i \bar{b}_j c_{in} + \bar{a}_i b_j \bar{c}_{in} \Rightarrow \\
s &= a_i b_j c_{in} + (a_i + b_j + c_{in})(\bar{a}_i \bar{b}_j + \bar{a}_i \bar{b}_j \bar{c}_{in} + \bar{a}_i b_j \bar{c}_{in}) \Rightarrow \\
s &= a_i b_j c_{in} + (a_i + b_j + c_{in})(\bar{a}_i + \bar{b}_j)(\bar{c}_{in} + \bar{a}_i \bar{b}_j) \Rightarrow \\
s &= a_i b_j c_{in} + (a_i + b_j + c_{in})(\bar{a}_i \bar{b}_j \cdot \bar{c}_{in} + \bar{a}_i \bar{b}_j) \Rightarrow \\
s &= a_i b_j c_{in} + (a_i + b_j + c_{in})(\bar{a}_i \bar{b}_j + c_{in}(\bar{a}_i + \bar{b}_j)) \Rightarrow \\
s &= a_i b_j c_{in} + (a_i + b_j + c_{in})(\bar{a}_i \bar{b}_j + c_{in}(\bar{a}_i + \bar{b}_j)) \Rightarrow \\
s &= a_i b_j c_{in} + (a_i + b_j + c_{in})(\bar{a}_i \bar{b}_j + c_{in}(\bar{a}_i + \bar{b}_j)) \Rightarrow \\
s &= a_i b_j c_{in} + (a_i + b_j + c_{in}) \cdot \bar{c}_{out}
\end{aligned}$$

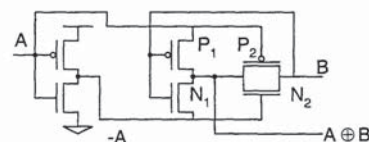
Από την παραπάνω αλγεβρική ανάλυση παρατηρούμε ότι το άθροισμα τελικά μπορεί να γραφτεί σαν συνάρτηση του κρατούμενου εξόδου. Επομένως, είναι δυνατόν να υλοποιήσουμε τη συνάρτηση του αθροίσματος χρησιμοποιώντας το κύκλωμα του κρατούμενου εξόδου και κάποιες επιπλέον πύλες. Στο σχήμα 1.5 φαίνεται ο πλήρης αθροιστής σε επίπεδο τρανζίστορ.



Σχήμα 1.5 Σχηματικό διάγραμμα πλήρη αθροιστή

Με αυτήν την υλοποίηση χρησιμοποιούμε 28 τρανζίστορ, ενώ με την χρήση μιας XOR πύλης τριών εισόδων, χρειαζόμαστε 32 τρανζίστορ. Όπως φαίνεται και από το σχήμα 1.5, τα σήματα του αθροίσματος και του κρατούμενου δεν εξάγονται την ίδια

χρονική στιγμή. Συγκεκριμένα, το άθροισμα καθυστερεί περισσότερο από το κρατούμενο. Πολλές φορές δεν ενδιαφέρει ο συγχρονισμός των δύο σημάτων. Στο κύκλωμα του παράλληλου αθροιστή για παράδειγμα, θα δούμε ότι η αυξημένη καθυστέρηση στο σήμα του αθροίσματος δεν δημιουργεί πρόβλημα. Υπάρχουν όμως περιπτώσεις κυκλωμάτων, όπου είναι απαραίτητη η εξαγωγή του αθροίσματος και του κρατούμενου ταυτόχρονα. Στα κυκλώματα αυτά εφαρμόζεται μια διαφορετική υλοποίηση του αθροίσματος η οποία χρησιμοποιεί μια πρωτότυπη XOR πύλη. Το κύκλωμα της πύλης σε επίπεδο τρανζίστορ φαίνεται στο σχήμα 1.6.

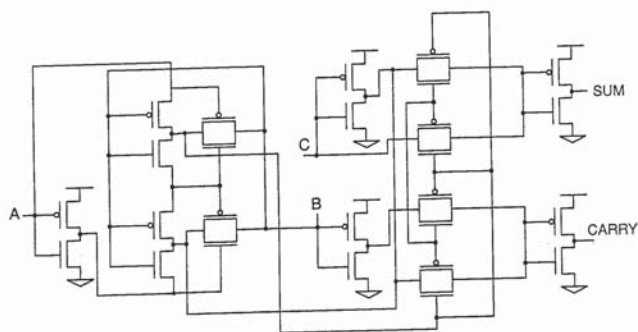


Σχήμα 1.6 XOR πύλης μετάδοσης

Όταν το σήμα A είναι στο λογικό 1, το -A είναι στο λογικό 0 και το ζευγάρι των τρανζίστορ P<sub>1</sub> και N<sub>1</sub> λειτουργεί ως αντιστροφέας. Αυτό έχει ως αποτέλεσμα να εμφανίζεται το B στην έξοδο αντεστραμμένο. Αντίθετα, όταν το σήμα A είναι στο λογικό 0, το σήμα -A είναι στο λογικό 1 και η πύλη μετάδοσης που δημιουργείται από τα τρανζίστορ P<sub>2</sub> και N<sub>2</sub> άγει, με αποτέλεσμα να βγαίνει το σήμα B ανέπαφο στην έξοδο. Για τη δημιουργία της παραπάνω πύλης χρησιμοποιούμε 6 τρανζίστορ.

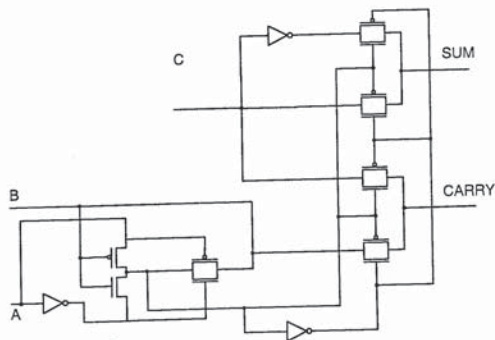
Η χρήση της πύλης που περιγράφηκε στο σχήμα 1.6 σε συνδυασμό με τέσσερις πύλες μετάδοσης (transmission gates) και δύο αντιστροφείς, μπορούν να μας οδηγήσουν σε έναν πλήρη αθροιστή όπως φαίνεται και στο σχήμα 1.7.

Το μεγάλο πλεονέκτημα του πλήρη αθροιστή του σχήματος 1.7, είναι ότι τα σήματα του αθροίσματος και του κρατούμενου εξάγονται την ίδια χρονική στιγμή. Η μορφή αυτή είναι χρήσιμη σε σύγχρονα κυκλώματα πράξεων όπως για παράδειγμα στον σειριακό αθροιστή καθώς και σε αριθμητικά κυκλώματα συνεχούς διοχέτευσης που θα παρουσιαστούν σε επόμενες παραγράφους.



Σχήμα 1.7 Πλήρης αθροιστής πύλης μετάδοσης

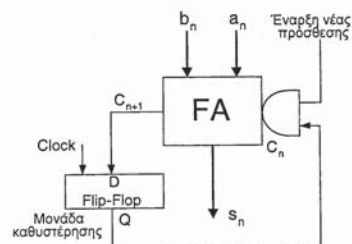
Ο αριθμός των τρανζίστορ που χρησιμοποιούνται μπορεί να μειωθεί αν η ταχύτητα δεν αποτελεί πρωταρχικό στόχο, όπως φαίνεται στο σχήμα 1.8.



Σχήμα 1.8 Βελτιστοποιημένης επιφάνειας αθροιστής πύλης μετάδοσης

### 1.2.2 Σειριακός Αθροιστής

Σκοπός μας είναι να προσθέσουμε δύο N-bit αριθμούς που εισάγονται σειριακά, δηλαδή τα bits των αριθμών έρχονται το ένα μετά το άλλο. Το αποτέλεσμα προκύπτει επίσης στην ίδια μορφή. Εδώ, θεωρούμε ότι η είσοδος αρχίζει με το LSB του κάθε αριθμού πρώτα, το αμέσως επόμενη τάξης μετά, κτλ. Ένας σειριακός αθροιστής αποτελείται από έναν πλήρη αθροιστή και έναν μανδαλωτή. Από κάθε πρόσθεση στον πλήρη αθροιστή παράγεται ένα άθροισμα και ένα κρατούμενο. Το κρατούμενο κάθε τάξης πρέπει να προστεθεί με τα ψηφία της επόμενης χρονικής στιγμής. Για αυτό το λόγο η έξοδος που αντιστοιχεί στο κρατούμενο  $c_n$ , οδηγείται σε ένα μανδαλωτή όπου και αποθηκεύεται μέχρι να έρθουν τα άλλα δύο bits της επόμενης χρονικής στιγμής ( $a_{n+1}$ ,  $b_{n+1}$ ), οπότε και προστίθεται σε αυτά με την μορφή του κρατουμένου εισόδου. Το διάγραμμα του σειριακού αθροιστή φαίνεται στο σχήμα 1.9.



Σχήμα 1.9 Σειριακός αθροιστής

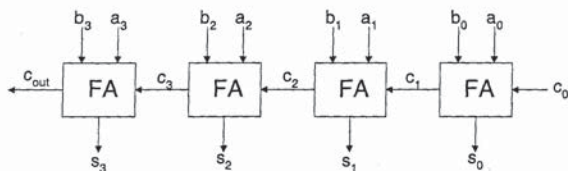
Διακρίνεται καθαρά η ανάδραση του κρατουμένου εξόδου από την πρόσθεση μέσω ενός μανδαλωτή στο κρατούμενο εισόδου. Είναι προφανές ότι στην πρώτη πρόσθεση πρέπει να έχουμε στην είσοδο του κρατουμένου εισόδου λογικό 0. Για να ολοκληρωθεί η πρόσθεση N-bit αριθμών χρειάζονται N+1 κύκλοι. Στον τελευταίο κύκλο εξάγεται το κρατούμενο  $c_{N+1}$ .



### 1.2.3 Παράλληλοι αθροιστές

#### 1.2.3.1 Εισαγωγή

Ένας από τους βασικότερους και πιο συνηθισμένους παράλληλους αθροιστές είναι ο αθροιστής διάδοσης κρατούμενου, ο οποίος αποτελείται από την απλή διασύνδεση πλήρων αθροιστών όπως φαίνεται και στο σχήμα 1.10.



Σχήμα 1.10 Αθροιστής διάδοσης κρατούμενου 4-bits

Τα A και B είναι οι αριθμοί που προστίθενται ενώ τα S και C είναι τα αθροίσματα και τα κρατούμενα αντίστοιχα. Τα bits ίδιας τάξης προστίθενται παράλληλα, ενώ το κρατούμενο κάθε βαθμίδας διαδίδεται προς τα αριστερά παίρνοντας μέρος κάθε φορά στην πρόσθεση της αμέσως επόμενης βαθμίδας. Το κρατούμενο εισόδου του πρώτου πλήρη αθροιστή είναι 0. Για να δούμε το πραγματικό αποτέλεσμα της πράξης, θα πρέπει να περιμένουμε για χρόνο ίσο με την καθυστέρηση του χειρότερου μονοπατιού του κυκλώματος. Θα πρέπει δηλαδή να διαδοθεί το κρατούμενο μέχρι τον τελευταίο πλήρη αθροιστή για να κάνει και αυτός σωστά την πράξη.

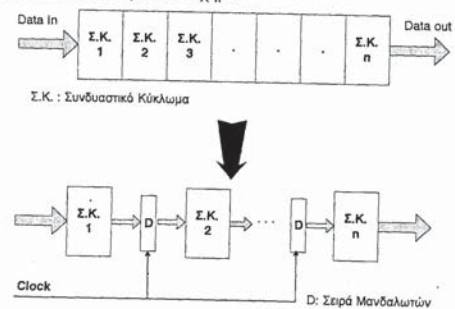
Παρατηρώντας τα σχήματα του σειριακού και του παράλληλου αθροιστή, βλέπουμε ότι υπάρχει μια αντιστοιχία ανάμεσά τους. Ο αθροιστής διάδοσης κρατούμενου αποτελεί το χρονικό ξεδίπλωμα του σειριακού ενώ αντίστροφα, ο σειριακός το χρονικό δίπλωμα του παράλληλου. Πραγματικά, κάθε ένας πλήρης αθροιστής (FA) του σχήματος 1.10 ξεκινώντας από τα δεξιά, αναπαριστά τον αθροιστή του σχήματος 1.9 σε αντίστοιχες χρονικές στιγμές. Την χρονική στιγμή 1 λοιπόν, ο σειριακός αθροιστής έχει τις ίδιες εισόδους και εξόδους με τον πρώτο FA από δεξιά του σχήματος 1.10. Την χρονική στιγμή 2 υπάρχει αντιστοίχιση των σημάτων του

### 1.2 Αθροιστές

σειριακού με αυτά του δεύτερου FA του παράλληλου και ούτω κάθε εξής. Τα παραπάνω μπορούν να μεταφερθούν και σε άλλα κυκλώματα εκτός από τους αθροιστές. Αποτελούν μια μέθοδο αντιστοίχισης παράλληλων και ακολουθιακών κυκλωμάτων που επιτελούν την ίδια λειτουργία. Τα παράλληλα είναι το χρονικό ξεδίπλωμα των ακολουθιακών ενώ αντίστροφα, τα ακολουθιακά είναι το χρονικό δίπλωμα των παράλληλων. Πολλά παραδείγματα εφαρμογής θα δούμε στα κυκλώματα των πολλαπλασιαστών.

#### 1.2.3.2 PIPELINING – Μετατροπή σε συνεχούς διοχέτευσης - συστολिकότητα

Όπως αναφέρθηκε παραπάνω, για να πάρουμε ένα σωστό αποτέλεσμα από τον παράλληλο αθροιστή πρέπει να περιμένουμε χρόνο T ίσο με την καθυστέρηση (latency) του κυκλώματος από την στιγμή που θα δώσουμε τους αριθμούς στις εισόδους του. Τι συμβαίνει αν θέλουμε μετά το τέλος της πρόσθεσης, να προσθέσουμε άλλους δύο αριθμούς; Προφανώς, θα πρέπει να περιμένουμε άλλον τόσο χρόνο για την εξαγωγή του αποτελέσματος. Για να βελτιώσουμε τον ρυθμό λειτουργίας του κυκλώματος (Throughput) χρησιμοποιούμε την τεχνική της συνεχούς διοχέτευσης (pipelining). Η μετατροπή ενός παράλληλου κυκλώματος σε κύκλωμα διοχέτευσης φαίνεται στο παρακάτω σχήμα:



Σχήμα 1.11 Μετατροπή παράλληλου σε συνεχούς διοχέτευσης

Στο πάνω μέρος του σχήματος 1.11 φαίνεται ένα κύκλωμα που αποτελείται από η υποκυκλώματα. Με το D συμβολίζουμε πλήθος μανδαλωτών όσες είναι οι γραμμές σημάτων που συνδέουν τα υποκυκλώματα μεταξύ τους. Η μετατροπή του σε κύκλωμα διοχέτευσης παρουσιάζεται στο κάτω μέρος του σχήματος 1.11. Παρατηρούμε ότι το αρχικό κύκλωμα έχει κοπεί και διαιρεθεί σε η επιμέρους υποκυκλώματα, ενώ παράλληλα παρεμβάλλονται και μονάδες καθυστέρησης. Οι μονάδες αυτές μπορεί να είναι μανδαλωτές ή D flip-flops. Με την εισαγωγή των μονάδων καθυστέρησης αυξάνουμε βέβαια την καθυστέρηση του κυκλώματος από την είσοδο μέχρι την έξοδο. Από την άλλη πλευρά όμως, επιτυγχάνουμε την αύξηση του ρυθμού λειτουργίας γιατί κάθε υποσύστημα ενεργεί πλέον ανεξάρτητα από τα υπόλοιπα με αποτέλεσμα να μην μένει ποτέ άεργο στο διάστημα που περιμένει για να πάρει δεδομένα. Έτσι, στο συγκεκριμένο παράδειγμα του παράλληλου αθροιστή, αν υποθέσουμε ότι χρειαζόμαστε χρόνο  $T$  για την εξαγωγή του συνολικού αποτελέσματος, ο πρώτος πλήρης αθροιστής εξάγει τα αποτελέσματά του μετά από χρόνο  $T/4$  (έχουμε τέσσερις βαθμίδες) και όλον τον υπόλοιπο χρόνο ( $3T/4$ ) παραμένει άεργος. Το ίδιο συμβαίνει με κάθε πλήρη αθροιστή του κυκλώματος. Με την μετατροπή του σε αθροιστή συνεχούς διοχέτευσης όπως θα δούμε παρακάτω, αρκεί ένα χρονικό διάστημα αρχικοποίησης (που είναι βέβαια μεγαλύτερο από την απλή καθυστέρηση του παράλληλου) μέχρι να εξαχθεί το πρώτο αποτέλεσμα και μετά μπορούμε να παίρνουμε τα αποτελέσματα των επόμενων αθροίσεων σε κάθε παλμό του ρολογιού του οποίου η περίοδος περιορίζεται από την καθυστέρηση ενός πλήρη αθροιστή και ενός μανδαλωτή.

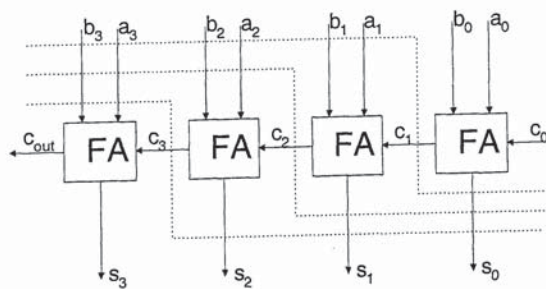
Εξετάζοντας τις παραπάνω παρατηρήσεις σε σχέση με το ρυθμό λειτουργίας ενός συγκεκριμένου κυκλώματος, για παράδειγμα αυτό του σχήματος 1.9, μπορούμε να καταλήξουμε σε κάποια πολύ γενικά και χρήσιμα συμπεράσματα. Έστω ότι  $T_{FA}$  είναι η καθυστέρηση του πλήρη αθροιστή και  $T_D$  η καθυστέρηση του κάθε μανδαλωτή. Είναι επόμενο, ότι η καθυστέρηση του αθροιστή του σχήματος 1.9 θα είναι  $T = 4 \cdot T_{FA}$ . Μετά την μετατροπή του σε αθροιστή διοχέτευσης (δεδομένου ότι θα χρησιμοποιήσουμε τέσσερις μανδαλωτές) η συνολική καθυστέρηση γίνεται  $T_{pre} = 4 \cdot (T_{FA} + T_D)$ . Ο ρυθμός όμως λειτουργίας γίνεται  $R = 1/(T_{FA} + T_D)$ . Παρατηρούμε ότι επειδή  $R = 1/(T_{FA} + T_D) \gg 1/4T_{FA}$ , ο αθροιστής διοχέτευσης μπορεί να λειτουργήσει σε

μεγαλύτερες συχνότητες από τον παράλληλο και μάλιστα όσο μεγαλύτερος είναι ο αριθμός των bit τόσο μεγαλύτερη βελτίωση θα έχουμε στον ρυθμό.

Όπως αναφέρθηκε και παραπάνω, η μετατροπή ενός κυκλώματος από παράλληλο σε συνεχούς διοχέτευσης, προϋποθέτει μερικές αλλαγές στο κύκλωμα, όπως για παράδειγμα την εισαγωγή στοιχείων καθυστέρησης. Το πρόβλημα είναι να βρεθεί η θέση και ο αριθμός των στοιχείων αυτών που πρόκειται να χρησιμοποιηθούν. Αναμφισβήτητα, θα πρέπει κάθε βαθμίδα να είναι απομονωμένη από την προηγούμενη και την επόμενη για να μπορεί να λειτουργεί αυτόνομα. Επομένως είναι σίγουρο ότι σε κάθε σήμα διασύνδεσης δύο βαθμίδων θα έχουμε τουλάχιστον ένα μανδαλωτή. Σκοπός μας είναι να καταφέρουμε να συγχρονίσουμε τα δεδομένα που εισέρχονται σε κάθε βαθμίδα. Η μεθοδολογία που ακολουθούμε είναι ότι προσπαθούμε με ευθείες γραμμές που διατρέχουν το κύκλωμα από άκρη σε άκρη, να "κόψουμε" όλα τα σήματα της ίδιας φοράς που συνδέουν δύο βαθμίδες περνούν μέσα από λογικά κυκλώματα και υφίστανται καθυστερήσεις. Για παράδειγμα, σήματα που απλώς μεταδίδονται στο κύκλωμα γιατί χρειάζονται σαν είσοδο σε κάθε κύτταρο, δεν είναι απαραίτητο να κοπούν δεδομένου ότι δεν καθυστερούν. Σε κάθε σημείο τομής των σημάτων με τις ευθείες γραμμές, εισάγουμε ακριβώς μια μονάδα καθυστέρησης. Για να είναι δυνατή μια τέτοια μετατροπή, πρέπει ο γράφος που αντιστοιχεί στο κύκλωμα να είναι άκυκλος και κατευθυνόμενος δηλαδή να μην υπάρχουν ανατροφοδοτήσεις και η μεταφορά δεδομένων όλων των σημάτων να είναι προς την ίδια κατεύθυνση.

Η παραπάνω περιγραφόμενη θεωρία είναι δυνατόν να γίνει πιο κατανοητή με την βοήθεια ενός παραδείγματος. Εφαρμογή θα γίνει στον παράλληλο αθροιστή. Το κύκλωμα του αθροιστή διάδοσης κρατούμενου "κόβεται" σε κομμάτια όπως φαίνεται στο σχήμα 1.12. Οι διακεκομμένες γραμμές σχηματίζουν τομές, πάνω στις οποίες βάζουμε στοιχεία καθυστέρησης (ένα σε κάθε τομή).





Σχήμα 1.12 Επιλογή σημείων παρεμβολής καθυστερήσεων

Στο σχήμα 1.12 παρατηρούμε ότι όλα τα σήματα που τέμνονται από μια διακεκομμένη γραμμή, έχουν την ίδια κατεύθυνση. Αυτό, όπως αναφέρθηκε παραπάνω είναι απαραίτητη προϋπόθεση. Αξίζει να εξετάσουμε την λειτουργία του κυκλώματος συνεχούς διοχέτευσης. Για να φανεί καλύτερα η λειτουργία, αναδιατάσσουμε το κύκλωμα του σχήματος 1.12 χωρίς να επιφέρουμε καμία αλλαγή. Η νέα μορφή του αθροιστή διάδοσης με τοποθετημένες τις μονάδες καθυστέρησης φαίνεται στο σχήμα 1.13. Έστω ότι θέλουμε να κάνουμε πρόσθεση δύο τετράμπτων αριθμών. Οι γραμμές εισόδου των bit των αριθμών που πρόκειται να προστεθούν, έχουν διαφορετικό αριθμό καθυστερήσεων και προσεγγίζουν τους πλήρεις αθροιστές με χρονική ολίσθηση, πράγμα που σημαίνει ότι μόνο τα  $a_0$  και  $b_0$  προσεγγίζουν σε πρώτο χρόνο τον δεξιότερο πλήρη αθροιστή που είναι και η πρώτη βαθμίδα του κυκλώματος. Αφού γίνει η πράξη στον αθροιστή, το αποτέλεσμα αποθηκεύεται στον μανδαλωτή που συνδέει τις δύο πρώτες βαθμίδες. Με τον επόμενο παλμό του ρολογιά, φθάνουν τα bits  $a_1$  και  $b_1$  στον δεύτερο πλήρη αθροιστή ενώ παράλληλα φτάνει και το αποθηκευμένο κρατούμενο που είναι απαραίτητο για να γίνει η πράξη. Έτσι επιτυγχάνεται συγχρονισμός των κρατουμένων που καθυστερούν με τα bit των αριθμών. Τα άλλα bits προχωρούν ακόμη μια θέση προς τα κάτω χωρίς όμως να έχουν φτάσει ακόμη στο κύκλωμα. Ο πρώτος αθροιστής έχει συμπληρώσει την λειτουργία του και είναι έτοιμος να δεχτεί τα πρώτα bits από το επόμενο ζευγάρι των

αριθμών που θέλουμε να προσθέσουμε. Τις επόμενες χρονικές στιγμές παρατηρούμε ότι έρχονται συνεχώς δεδομένα που γεμίζουν τους καταχωρητές μέχρι τους πλήρεις αθροιστές, ενώ παράλληλα βγαίνουν τα αποτελέσματα του αθροίσματος στις εξόδους των FA. Οι μανδαλωτές στις εξόδους επανασυγχρονίζουν τα αθροίσματα της κάθε βαθμίδας. Η συνεχής αυτή λειτουργία φαίνεται παραστατικά στον πίνακα 1.2 όπου έχουμε ένα στιγμιότυπο άθροισης πέντε ζευγαριών δεδομένων.

Πίνακας 1.2 Άθροιση τεσσάρων αριθμών 4-bit

	$a_2^4 b_2^4$	$a_1^4 b_1^4$	$a_0^4 b_0^4$
$a_3^3 b_3^3$	$a_2^3 b_2^3$	$a_1^3 b_1^3$	$a_0^3 b_0^3$
$a_2^2 b_2^2$	$a_1^2 b_1^2$	$a_0^2 b_0^2$	$s_0^3$
$a_1^1 b_1^1$	$a_0^1 b_0^1$	$s_1^2$	$s_0^2$
$a_0^0 b_0^0$	$s_2^1$	$s_1^1$	$s_0^1$
$s_3^0$	$s_2^0$	$s_1^0$	$s_0^0$

Στον πίνακα 1.2, τα γραμμοσκιασμένα κελιά παίζουν τον ρόλο των πλήρων αθροιστών όπως φαίνονται διατεταγμένοι στο σχήμα 1.12. Με τον όρο  $s_n^m$  συμβολίζουμε το n-bit του m αριθμού. Σε κάθε στήλη, ο κάτω δείκτης παραμένει ίδιος αφού ο κάθε πλήρης αθροιστής δέχεται bits συγκεκριμένης αξίας. Αντίστοιχα, σε κάθε σειρά, ο πάνω δείκτης δεν μεταβάλλεται αφού πρόκειται για bits του ίδιου αριθμού. Στο στιγμιότυπο που φαίνεται στον πίνακα 1.2, παρατηρούμε ότι τα αθροίσματα των τριών πρώτων bit των αριθμών με δείκτη 0 έχουν εξαχθεί ενώ του τέταρτου υπολογίζεται. Όπως φαίνεται και από τον πίνακα 1.2, με την μετατροπή του κυκλώματος σε συνεχούς διοχέτευσης, επιτυγχάνουμε την άφιξη των bits κάθε αριθμού με διαφορά μιας χρονικής στιγμής. Κάνοντας την νοητή αντιστοίχιση της κάθε σειράς του πίνακα 1.2 σε μια χρονική στιγμή, παρατηρούμε ότι για να έχουμε ολοκληρωμένο το αποτέλεσμα του αθροίσματος των αριθμών με δείκτη 0, πρέπει να περιμένουμε πέντε χρονικές στιγμές. Τα αποτελέσματα όμως φτάνουν με καθυστέρηση μιας χρονικής στιγμής.

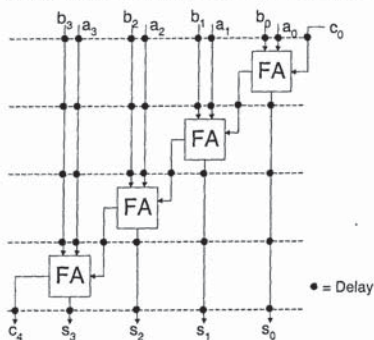
Μια ακόμη πιο εξειδικευμένη περίπτωση συνεχούς διοχέτευσης, αποτελεί η συστολικότητα. Για να είναι ένα κύκλωμα συστολικό θα πρέπει να πληρούνται οι εξής τρεις προϋποθέσεις:

Τα κύτταρα που αποτελούν το κύκλωμα πρέπει να είναι πανομοιότυπα.

Πρέπει να υποστηρίζεται η λογική της συνεχούς διοχέτευσης.

Απαγορεύεται να υπάρχουν γραμμές σημάτων μεταξύ των κυττάρων που να μην διακόπτονται από στοιχεία καθυστέρησης (latches).

Το βασικό πλεονέκτημα των συστολικών δικτύων είναι η κατασκευαστική τους απλότητα, αφού επαναλαμβάνεται ένα βασικό κύτταρο. Έτσι, η όλη δυσκολία έγκειται στην σωστή υλοποίηση μιας δομικής μονάδας, η οποία έχει την ίδια λειτουργία, είτε βρίσκεται στο κέντρο του συστολικού δικτύου, είτε στις άκρες, όπου φροντίζει και για την είσοδο και έξοδο των δεδομένων του κυκλώματος. Σε περίπτωση όμως που έχουμε ένα κύκλωμα συνδυαστικό που αποτελείται από ίδια κύτταρα, αλλά που δεν είναι συστολικό, υπάρχει δυνατότητα (όχι όμως πάντα) να το μετατρέψουμε σε συστολικό. Χρησιμοποιούμε πάλι την μέθοδο των τομών. Προσπαθούμε με συνεχόμενες γραμμές που διατρέχουν το κύκλωμα να κόψουμε όλα τα σήματα εισόδου-εξόδου των κυττάρων. Στα σημεία τομών των γραμμών αυτών με τα σήματα εισάγουμε καθυστερήσεις. Κάθε σήμα πρέπει να έχει τουλάχιστον έναν μανδαλωτή.

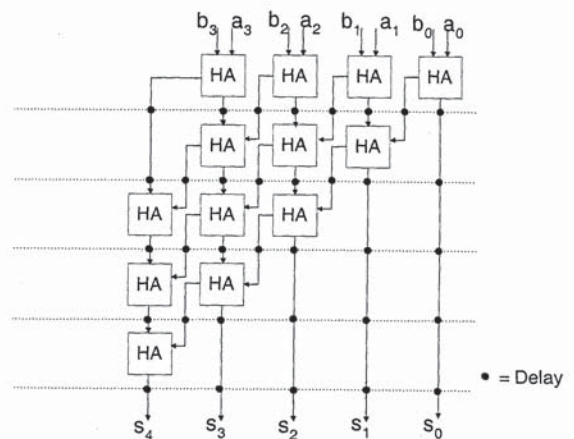


Σχήμα 1.13 Ο αθροιστής διάδοσης κρατούμενου σε λειτουργία συνεχούς διοχέτευσης

Στο παράδειγμα του σχήματος 1.13, οι τομές για την μετατροπή παράλληλου κυκλώματος σε συστολικό είναι σχεδόν οι ίδιες με τις τομές για την μετατροπή σε συνεχούς διοχέτευσης, χωρίς αυτό να αποτελεί κανόνα όπως θα δούμε σε παρακάτω κεφάλαιο.

Παρατηρούμε ότι οι θέσεις των τομών πληρούν τις προϋποθέσεις για την συστολικότητα. Το κύκλωμα είναι συνεχούς διοχέτευσης, αποτελείται από ίδια κύτταρα και όλες οι γραμμές διακόπτονται από στοιχεία καθυστέρησης. Βέβαια, οι έννοιες συστολικότητας και συνεχούς διοχέτευσης δεν είναι ισοδύναμες. Ένα συστολικό κύκλωμα είναι σίγουρα και συνεχούς διοχέτευσης ενώ το αντίστροφο δεν ισχύει πάντα.

Παράλληλος συστολικός αθροιστής μπορεί να υλοποιηθεί και με την χρήση ημιαθροιστών. Το κύκλωμα φαίνεται στο σχήμα 1.14.



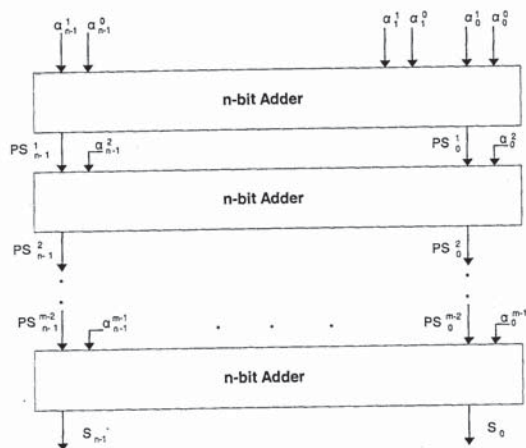
Σχήμα 1.14 Συστολικό κύκλωμα παράλληλου αθροιστή με ημιαθροιστές



### 1.2.3.3 Δίκτυα παραλλήλων αθροιστών M αριθμών

Στις προηγούμενες ενότητες ασχοληθήκαμε βασικά με πρόσθεση δύο αριθμών. Έστω τώρα ότι πρέπει να προστεθούν m αριθμοί  $A^k$  όπου  $k=0,1,\dots,m-1$  που έχουν μήκος n bits ο καθένας. Το σχήμα 1.15 δείχνει τον τρόπο με τον οποίο διατάσσονται τα bits των προστιθέμενων αριθμών σε έναν παράλληλο αθροιστή.

Το κάθε bit συμβολίζεται με  $a_i^k$ , όπου k ο δείκτης του προσθετέου και i η τάξη του bit του αριθμού αυτού. Για παράδειγμα, το bit  $a_2^3$  είναι το δεύτερο bit του αριθμού  $A^3$  που πρόκειται να προστεθεί. Το κύκλωμα αποτελείται από m-1 αθροιστές των n bits. Το σήμα  $PS_i^k$  (partial sum) είναι το i τάξεως bit του μερικού αθροίσματος k αριθμών σύμφωνα με την σχέση  $PS^k = \sum_{j=0}^k A^j$ . Με την διαδοχική πρόσθεση των m αριθμών θα προκύψει το συνολικό άθροισμα S. Είναι βέβαια προφανές ότι  $PS^{m-1} = \sum_{k=0}^{m-1} A^k = S$  που είναι το ζητούμενο συνολικό άθροισμα.



Σχήμα 1.15 Πρόσθεση m αριθμών n-bit

Παρατηρούμε ότι εδώ δεν έχουμε μια απλή σειρά πλήρων αθροιστών, αλλά ένα δίκτυο, αφού πρέπει να προσθέσουμε παραπάνω από δύο αριθμούς. Το δίκτυο διαφοροποιείται ανάλογα με το είδος του αθροιστή που χρησιμοποιείται. Θα εξετάσουμε στη συνέχεια την λειτουργία ενός τέτοιου κυκλώματος χρησιμοποιώντας ripple carry, carry save και carry look-ahead αθροιστές. Τα δύο τελευταία είδη αθροιστών δεν έχουν παρουσιαστεί μέχρι στιγμής και θα παρουσιαστούν μαζί με τα αντίστοιχα δίκτυά τους. Σε όλες τις περιπτώσεις θα θεωρήσουμε πως δεν υπάρχει υπερχειλίση.

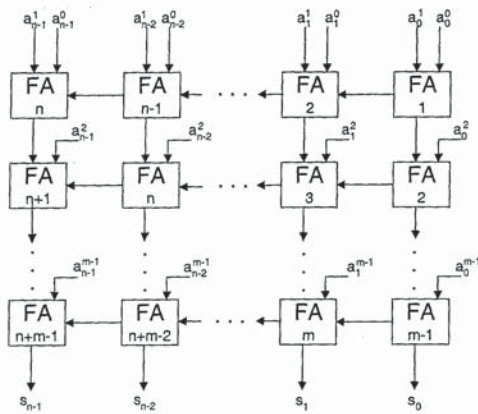
Για την μετατροπή του δικτύου παράλληλων αθροιστών σε συνεχούς διοχέτευσης, όπως έχουμε παρουσιάσει σε προηγούμενη παράγραφο, αρκεί να διακόψουμε την απευθείας σύνδεση των παράλληλων αθροιστών με μανδαλωτές. Για να συγχρονιστεί το κύκλωμα όμως, θα πρέπει ο παλμός με τον οποίο θα τροφοδοτούνται τα στοιχεία καθυστέρησης να είναι μεγαλύτερος ή ίσος με την συνολική καθυστέρηση ενός επιπέδου. Με αυτήν την προϋπόθεση, θα προλαβαίνει το άθροισμα κάθε επιπέδου να εξάγεται και να τροφοδοτεί το επόμενο επίπεδο. Δεδομένου επίσης ότι οι αριθμοί που θέλουμε να προσθέσουμε έρχονται την ίδια χρονική στιγμή, θα πρέπει να καθυστερήσουμε τους αριθμούς που προστίθενται από το δεύτερο επίπεδο και κάτω κατά αντίστοιχους παλμούς ρολογιού. Έτσι, ο τρίτος αριθμός που εισέρχεται στο κύκλωμα στο δεύτερο επίπεδο για παράδειγμα, θα καθυστερηθεί κατά δύο παλμούς ρολογιού (δηλαδή στα σήματά του θα παρεμβάλουμε δύο επίπεδα μανδαλωτών), ώστε να συγχρονιστεί με το άθροισμα των δύο πρώτων αριθμών από το πρώτο επίπεδο.

Για να εξάγουμε το πρώτο αποτέλεσμα πρέπει να περιμένουμε m+1 κύκλους ρολογιού. Όπως αναφέρθηκε παραπάνω, ο κάθε κύκλος ρολογιού πρέπει να διαρκεί χρόνο τουλάχιστον ίσο με την καθυστέρηση του ενός επιπέδου. Ο συνολικός χρόνος για το πρώτο αποτέλεσμα δίνεται από την παρακάτω σχέση:  $T = (m+1) \cdot n \cdot T_{FA}$  όπου m οι αριθμοί που θέλουμε να προσθέσουμε και  $n \cdot T_{FA}$  η καθυστέρηση του ενός επιπέδου. Μετά τη συμπλήρωση του παραπάνω χρόνου, τα αποτελέσματα εξάγονται ανά παλμό ρολογιού. Η αύξηση του ρυθμού λειτουργίας έχει σαν αποτέλεσμα την επιβάρυνση του κυκλώματος με μανδαλωτές. Η ανάλυση αυτή είναι μια πρώτη προσέγγιση. Όπως θα δούμε στη συνέχεια, κατεβαίνοντας στο επίπεδο του bit

αποκτούμε μια πληρέστερη εικόνα της λειτουργίας και των επεμβάσεων που μπορούμε να κάνουμε.

#### 1.2.3.4 Δίκτυο με αθροιστές διάδοσης κρατούμενου.

Πρόκειται για τον αθροιστή που έχουμε ήδη περιγράψει σε γενικές γραμμές στις προηγούμενες ενότητες. Το κύκλωμα αποτελείται από πλήρεις αθροιστές (FAs) οι οποίοι διασυνδέονται με την γνωστή μέθοδο που παρουσιάστηκε στο σχήμα 1.15. Πιο συγκεκριμένα, σε επίπεδο FA το κύκλωμα του δικτύου φαίνεται στο σχήμα 1.16.



Σχήμα 1.16 Δίκτυο αθροιστών διάδοσης κρατούμενου

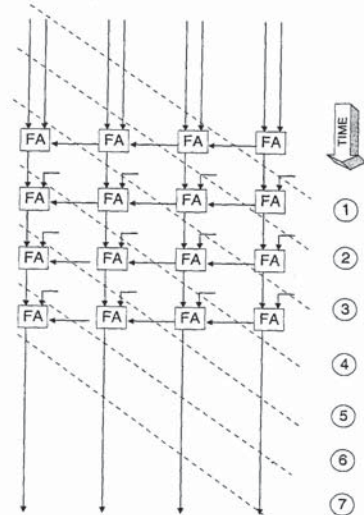
Στο πρώτο επίπεδο προστίθενται οι δύο πρώτοι αριθμοί και μόλις εξαχθούν τα αθροίσματα προστίθενται στο δεύτερο επίπεδο μαζί με τον τρίτο αριθμό κ.ο.κ. Για να μπορούμε να περάσουμε στο επόμενο επίπεδο βέβαια, δεν είναι ανάγκη να περιμένουμε να ολοκληρωθεί όλη η πρόσθεση στο προηγούμενο επίπεδο. Αρκεί να έχει ολοκληρωθεί η πρόσθεση στους προηγούμενους πλήρεις αθροιστές. Αυτό

σημαίνει ότι αν έχουμε ένα δίκτυο από  $m$  αθροιστές των  $n$  bits, τότε η συνολική καθυστέρηση δίνεται από την σχέση:

$$T = n \cdot T_{FA} + (m-1) \cdot T_{FA} \Rightarrow T = (n+m-1) \cdot T_{FA} \text{ όπου } T_{FA} \text{ η καθυστέρηση}$$

ενός πλήρη αθροιστή, θεωρώντας ότι είναι ίδια και για τις δύο εξόδους  $s, c$  του FA. Παρατηρείστε ότι όποια διαδρομή και να ακολουθήσουμε η καθυστέρηση θα δίνεται από την προηγούμενη σχέση.

Για να μετατρέψουμε ένα τέτοιο κύκλωμα σε συνεχούς διοχέτευσης, μπορούμε να το "κόψουμε" με διαγώνιες γραμμές όπως φαίνεται στο σχήμα 1.17 για τον αθροιστή 4X4.



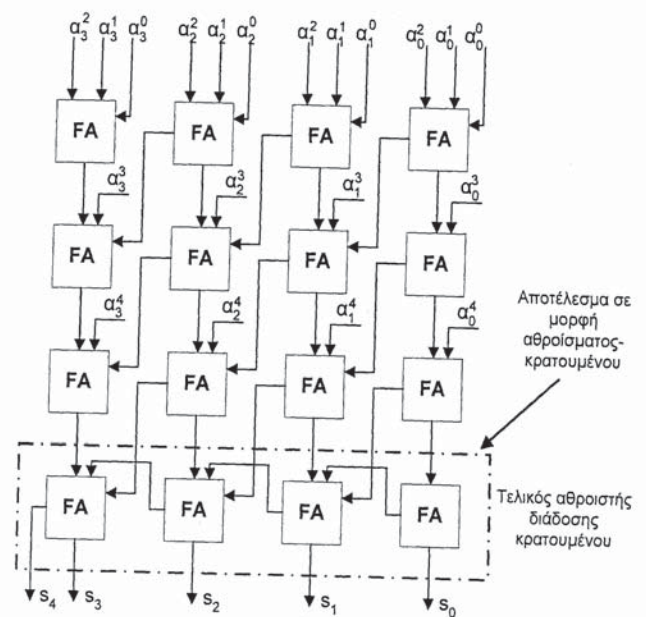
Σχήμα 1.17 Μετατροπή του 4X4 ripple carry αθροιστή σε συνεχούς διοχέτευσης

Στις τομές των διακεκομμένων γραμμών με τα σήματα, εισάγονται μανδαλωτές, όπως και στα προηγούμενα κυκλώματα συνεχούς διοχέτευσης. Στα δεξιά του κυκλώματος φαίνονται οι χρονικές στιγμές που αντιστοιχούν στην διάδοση των δεδομένων στο κύκλωμα. Την χρονική στιγμή 1 για παράδειγμα, τα δεδομένα φτάνουν μέχρι την πρώτη διακεκομμένη γραμμή όπου και κλειδώνονται στους μανδαλωτές. Την χρονική στιγμή 2 τα δεδομένα κλειδώνονται στους μανδαλωτές της δεύτερης γραμμής κ.ο.κ. Οι πλήρεις αθροιστές που βρίσκονται ανάμεσα σε δύο διακεκομμένες γραμμές ενεργοποιούνται την ίδια χρονική στιγμή. Παρατηρούμε ότι για το 4-bit δίκτυο παράλληλου αθροιστή διάδοσης κρατουμένου απαιτούνται 7 παλμοί για εξαχθεί το πρώτο αποτέλεσμα. Μπορούμε να γενικεύσουμε την παρατήρηση αυτή για ένα δίκτυο nxm. Το πλήθος των διακεκομμένων γραμμών που χρειάζονται, δηλαδή τα επίπεδα των μανδαλωτών, είναι  $n+m-1$ . Η συνολική καθυστέρηση του δικτύου είναι  $T = (n+m-1)(T_{\text{match}} + T_{FA})$  όπου  $T_{\text{match}}$  η καθυστέρηση του μανδαλωτή και  $T_{FA}$  η καθυστέρηση του πλήρη αθροιστή.

Μια ακόμη σημαντική παρατήρηση είναι ότι όλα τα σήματα του δικτύου τέμνονται από τις διακεκομμένες γραμμές. Με αυτόν τον τρόπο τομής του κυκλώματος πληρούνται και οι τρεις προϋποθέσεις για την συστολικότητα. Επομένως, το κύκλωμα είναι και συστολικό εκτός από συνεχούς διοχέτευσης.

### Δίκτυο αθροιστών με σώσιμο κρατουμένου

Πρόκειται για μια διαφορετική φιλοσοφία διασύνδεσης πλήρων αθροιστών για την δημιουργία ενός δικτύου στο οποίο τα κρατούμενα δεν διαδίδονται στο ίδιο επίπεδο αλλά μεταδίδονται στο επόμενο. Πιο συγκεκριμένα, το κρατούμενο κάθε πλήρη αθροιστή πρέπει να διαδίδεται μια θέση αριστερότερα από το άθροισμα, αφού είναι μια τάξη μεγέθους μεγαλύτερη από το bit του αθροίσματος. Ενώ στον αθροιστή διάδοσης κρατουμένου διαδίδεται το κρατούμενο στο ίδιο επίπεδο, τώρα έχουμε μετάδοση κρατουμένου στο αμέσως πιο κάτω επίπεδο. Ένας τέτοιος αθροιστής φαίνεται στο σχήμα 1.18, όπου  $n=4$  και  $m=5$ . Προσθέτουμε δηλαδή πέντε αριθμούς των τεσσάρων bit. Κάθε σειρά πλήρων αθροιστών του δικτύου αυτού αποτελεί ένα παράλληλο αθροιστή (4-bit) με σώσιμο κρατουμένου.



Σχήμα 1.18 Δίκτυο 4x4 αθροιστών με σώσιμο κρατουμένου

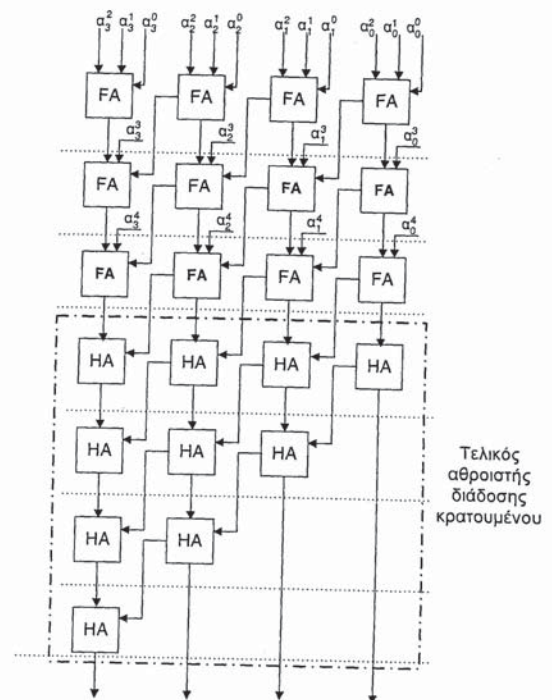
Όπως φαίνεται και στο σχήμα 1.18, τα κρατούμενα διαδίδονται προς τα κάτω μέχρι να προστεθούν όλοι οι αριθμοί που θέλουμε να προσθέσουμε. Στο σχήμα 1.18 προσθέτουμε και τους πέντε αριθμούς στα πρώτα τρία επίπεδα. Η έξοδος των πλήρων αθροιστών του τρίτου επιπέδου περιέχει το άθροισμα που θέλουμε αλλά σε μορφή αθροίσματος-κρατουμένου. Για να την μετατρέψουμε σε συμβατική δυαδική μορφή, πρέπει γίνει μια τελική άθροιση με τη χρήση ενός ακόμη επιπέδου πλήρων



αθροιστών. Μπορούμε να χρησιμοποιήσουμε έναν αθροιστή διάδοσης ή έναν γρήγορο αθροιστή πρόβλεψης κρατουμένου (ο τελευταίος θα περιγραφεί στην επόμενη ενότητα). Το τελευταίο επίπεδο στο σχήμα 1.18 περικλείεται σε ένα πλαίσιο για να τονίσει αυτή ακριβώς την ιδιαιτερότητά του.

Από τα παραπάνω καταλήγουμε στο συμπέρασμα ότι η καθυστέρηση του αθροίσματος  $m$  αριθμών  $n$  bits, θα είναι:  $T_s = (m-2) \cdot T_{FA} + T_A$  όπου  $T_{FA}$  είναι ο χρόνος καθυστέρησης του πλήρη αθροιστή και  $T_A$  ο χρόνος καθυστέρησης του αθροιστή του τελευταίου επιπέδου. Αν χρησιμοποιήσουμε έναν carry look-ahead αθροιστή για την τελική πρόσθεση, τότε ισχύει  $T_A = (2 + \log_2 n) \cdot T_{FA}$ , ενώ στην περίπτωση ενός carry propagate ή ενός ripple carry αθροιστή, θα ισχύει  $T_A = n \cdot T_{FA}$ .

Η μετατροπή του παραπάνω κυκλώματος σε συνεχούς διοχέτευσης είναι αρκετά απλή αν παρατηρήσουμε ότι δεν υπάρχουν οριζόντια σήματα που παίρνουν μέρος στην λογική κάποιου κυττάρου και επομένως πρέπει να "κοπούν". Οι οριζόντιες διακεκομμένες γραμμές που φαίνονται και στο παρακάτω σχήμα υπαγορεύουν και τις θέσεις που πρέπει να εισαχθούν οι μονάδες καθυστέρησης. Βέβαια, είναι απαραίτητη και η δυνατότητα λειτουργίας του τελευταίου επιπέδου σαν κύκλωμα συνεχούς διοχέτευσης έτσι ώστε να μην "φρενάρει" τον ρυθμό παραγωγής αποτελεσμάτων του αθροιστή με σώσιμο κρατουμένου. Για να γίνει αυτό, μπορούμε για παράδειγμα να χρησιμοποιήσουμε τους αθροιστές συνεχούς διοχέτευσης που εξετάσαμε στην προηγούμενη ενότητα.

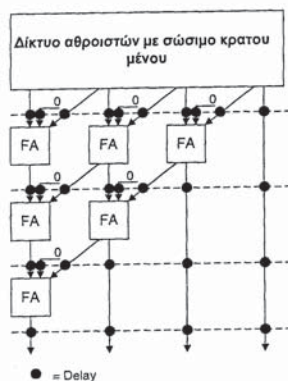


Σχήμα 1.19 Μετατροπή του 4x4 αθροιστή με σώσιμο κρατουμένου σε συνεχούς διοχέτευσης

Στην τελική πρόσθεση χρησιμοποιούμε τον αθροιστή διάδοσης κρατουμένου του σχήματος 1.14. Παρατηρούμε ότι τα bits των αριθμών που θέλουμε να προσθέσουμε εισέρχονται συγχρονισμένα στους πλήρεις αθροιστές. Σε κάθε παλμό

του ρολογιού τα δεδομένα μετακινούνται μια θέση προς τα κάτω, ενώ καινούρια bits μπαίνουν στο πρώτο επίπεδο του κυκλώματος.

Για να μετατρέψουμε το παραπάνω κύκλωμα σε συστολικό πρέπει να γίνει μια μικρή μετατροπή του τελευταίου επιπέδου, αφού το υπόλοιπο κύκλωμα πληρεί τις προϋποθέσεις της συστολικότητας. Το τελευταίο επίπεδο δεν πληρεί την προϋπόθεση της ομοιομορφίας με το υπόλοιπο κύκλωμα. Για να γίνει λοιπόν το δίκτυο του σχήματος 1.19 συστολικό, είναι προφανές ότι πρέπει να χρησιμοποιήσουμε τα ίδια κύτταρα σε όλο το κύκλωμα. Αυτό επιτυγχάνεται αντικαθιστώντας τους ημιαθροιστές της τριγωνικής δομής, με πλήρεις αθροιστές, θέτοντας ταυτόχρονα την μια τους είσοδο στο λογικό 0.



Σχήμα 1.20 Κύκλωμα τελικής πρόσθεσης για συστολικότητα

Όπως αναφέρθηκε παραπάνω, η μια είσοδος των πλήρων αθροιστών είναι πάντα στο λογικό 0 αφού στην πραγματικότητα χρειαζόμαστε ημιαθροιστές δεδομένου ότι μας ενδιαφέρει να αθροίσουμε τα αθροίσματα και τα κρατούμενα από τον carry save αθροιστή και όχι να προσθέσουμε νέους αριθμούς. Τοποθετούμε όμως πλήρεις αθροιστές για λόγους ομοιομορφίας, απαραίτητη προϋπόθεση για συστολικότητα. Με

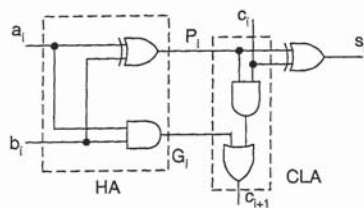
την μορφή αυτή και φυσικά με τους απαραίτητους μανδαλωτές στα σημεία τομής των διακεκομμένων γραμμών με τα σήματα, μπορούμε να κάνουμε το δίκτυο συστολικό.

Αξίζει εδώ να αναφερθεί ότι η μετατροπή και αυτού του κυκλώματος σε συνεχούς διοχέτευσης και σε συστολικό σχεδόν ταυτίζεται. Αυτό συμβαίνει λόγω της μεγάλης ομοιομορφίας και της κανονικότητας που έχουν τα δίκτυα αθροιστών. Στην πραγματικότητα όμως η μετατροπή ενός κυκλώματος σε συστολικό είναι πιο αυστηρή από αυτή της συνεχούς διοχέτευσης και πολλές φορές απαιτείται διαφορετικός χρονισμός για την επίτευξή της.

### Αθροιστές πρόβλεψης κρατουμένου

Όπως αναφέρθηκε, στο κύκλωμα του αθροιστή διάδοσης κρατουμένου, αλλά και στα κυκλώματα στην παράγραφο των δικτύων παράλληλων αθροιστών υπάρχει το πρόβλημα της διάδοσης του κρατουμένου. Το αποτέλεσμα είναι ότι τα κυκλώματα αυτά δουλεύουν σωστά, δεν δουλεύουν όμως με τον αποδοτικότερο τρόπο. Το γιατί, φαίνεται πολύ παραστατικά στο σχήμα 1.21α. Υποθέτουμε ότι τα bits των προσθετέων εφαρμόζονται ταυτόχρονα στις εισόδους των πλήρων αθροιστών. Από το λογικό διάγραμμα του πλήρους αθροιστή μπορούμε να διαπιστώσουμε ότι το κρατούμενο  $C_{i+1}$  εξέρχεται από το κύτταρο με καθυστέρηση δύο λογικών επιπέδων (πύλη AND και πύλη OR) από τη στιγμή που το κρατούμενο  $C_i$  εισέρχεται σε αυτό.

Στο πρώτο κύτταρο λοιπόν, από τη στιγμή που εφαρμόζονται τα bits των προσθετέων στις εισόδους των αθροιστών έχουμε την καθυστέρηση μιας λογικού πύλης, ώσπου να οριστικοποιηθούν οι τιμές των  $P_0$  και  $G_0$ , καθυστέρηση 2 λογικών πυλών για την παραγωγή του  $S_0$  και τέλος καθυστέρηση 3 λογικών πυλών ώσπου να οριστικοποιηθεί και η τιμή του κρατουμένου εξόδου (το κρατούμενο εισόδου είναι εδώ 0). Στο δεύτερο κύτταρο, όπως και σε όλα τα υπόλοιπα, οι τιμές των σημάτων  $P_i$  και  $G_i$  έχουν ήδη οριστικοποιηθεί με χρονική καθυστέρηση ενός λογικού επιπέδου από την εφαρμογή των προσθετέων. Συνεπώς οι σωστές έξοδοι καθυστερούν να εμφανιστούν εξαιτίας του κρατουμένου εισόδου. Το κρατούμενο εξόδου, που αποτελεί κρατούμενο εισόδου για την επόμενη βαθμίδα, παίρνει την σωστή τιμή του με καθυστέρηση δύο λογικών πυλών από την άφιξη του κρατουμένου εισόδου για την παρούσα βαθμίδα.



Σχήμα 1.21α Κύκλωμα αθροιστή

Αξίζει στο σημείο αυτό να εξετάσουμε το συμβολισμό και το ρόλο των σημάτων  $P_i$  και  $G_i$ . Το σήμα  $G_i$  ονομάζεται παραγόμενο σήμα κρατούμενου. Είναι προφανές από το παραπάνω λογικό διάγραμμα, ότι το σήμα αυτό είναι η αιτία "γέννησης" κρατούμενου εξόδου από τα δύο εισερχόμενα bits των προσθετέων, εφόσον είναι και τα δύο 1, ανεξάρτητα από την τιμή του κρατούμενου εισόδου. Εάν όμως δεν είναι και τα δύο bits των προσθετέων 1, τότε, μόνο όταν έχουμε κρατούμενο εισόδου μπορεί να προκύψει κρατούμενο εξόδου. Λέμε τότε ότι έχουμε διάδοση κρατούμενου. Το σήμα  $P_i$  ονομάζεται διαδιδόμενο σήμα κρατούμενου γιατί μόνον όταν αυτό είναι 1 (δηλαδή όταν ένα μόνο από τα bits των προσθετέων είναι 1), η τιμή του κρατούμενου εισόδου διαδίδεται στο σήμα κρατούμενου εξόδου, ανεξάρτητα της τιμής του  $G_i$  που στην περίπτωση αυτή είναι 0.

Συνοψίζοντας την προηγούμενη ανάλυση είναι εύκολο να διαπιστώσουμε ότι έχοντας έναν παράλληλο αθροιστή των  $n$  bits το τελικό αποτέλεσμα, άθροισμα και κρατούμενο εξόδου μέχρι και την τελευταία βαθμίδα, παράγεται με καθυστέρηση ίση με δύο λογικά επίπεδα επί τον αριθμό των βαθμίδων συν ένα λογικό επίπεδο καθυστέρησης για τον ταυτόχρονο σχηματισμό των  $P_i$  και  $G_i$ . Συνολική καθυστέρηση δηλαδή, για τον αθροιστή των  $n$  bits ίση με  $(2 \cdot n + 1) \cdot T$  όπου  $T$  η καθυστέρηση ενός λογικού επιπέδου, με την προϋπόθεση ότι χρησιμοποιούμε πλήρεις αθροιστές υλοποιημένους όπως στο σχήμα 1.21α για όλες τις βαθμίδες.

Από την άλλη μεριά είναι προφανές ότι, ιδανικά, το σωστό αποτέλεσμα θα μπορούσε να προκύψει με καθυστέρηση τριών λογικών επιπέδων αν το κρατούμενο

εισόδου σε κάθε βαθμίδα έπαιρνε τη σωστή του τιμή ταυτόχρονα με την εφαρμογή των bits των προσθετέων. Αξίζει λοιπόν τον κόπο να διερευνήσουμε το κατά πόσον είναι δυνατό να προσεγγισθεί αυτή η από άποψη ταχύτητας, ιδανική συμπεριφορά του κυκλώματος.

Ένας τρόπος για να λύσουμε αυτό το πρόβλημα είναι να σχεδιάσουμε ένα κύκλωμα που να προβλέπει το κρατούμενο εισόδου σε κάθε βαθμίδα με δεδομένα τα bits των προσθετέων. Όπως έχει ήδη αναφερθεί, ισχύουν οι σχέσεις  $G_i = A_i \cdot B_i$  για το σήμα γέννησης κρατούμενου και  $P_i = A_i \oplus B_i$  για το σήμα διάδοσης κρατούμενου. Σκοπός μας είναι να εκφράσουμε το κρατούμενο μιας βαθμίδας σε συνάρτηση με τα σήματα εισόδου της. Για να έχουμε κρατούμενο εξόδου στην βαθμίδα  $i$  για παράδειγμα, θα πρέπει να συμβαίνουν τα εξής:

"Γεννιέται" το κρατούμενο από τα προστιθέμενα bits των αριθμών ( $G_i = 1$ ).

"Διαδίδεται" το κρατούμενο εισόδου που προέρχεται από προηγούμενη βαθμίδα ( $P_i \cdot C_i$ ).

Η σύνθεση των παραπάνω παρατηρήσεων μας οδηγεί στην σχέση

$$C_{i+1} = G_i + P_i \cdot C_i$$

όπου  $C_{i+1}$  το κρατούμενο εξόδου και  $C_i$  το κρατούμενο εισόδου, κρατούμενο που προέρχεται από την προηγούμενη βαθμίδα. Αντίστοιχα το άθροισμα  $S$  που δίνεται από την γνωστή σχέση  $S_i = C_i \oplus A_i \oplus B_i$  μετασχηματίζεται σε  $S_i = C_i \oplus P_i$  όπου το σήμα διάδοσης κρατούμενου είναι  $P_i = A_i \oplus B_i$ . Αυτό σημαίνει ότι η καθυστέρηση του κυκλώματος ενός αθροιστή πρόβλεψης κρατούμενου συνίσταται στην εξαγωγή των κρατούμενων αφού τα σήματα διάδοσης κρατούμενων μπορούν να υπολογιστούν αμέσως. Ο αριθμός των βαθμίδων της πρόβλεψης περιορίζεται συνήθως στις τέσσερις χωρίς να αποκλείεται η επέκταση σε περισσότερα bits. Πιο συγκεκριμένα, οι σχέσεις που δίνουν τα κρατούμενα σε έναν 4-bit αθροιστή πρόβλεψης κρατούμενου, είναι οι εξής:

$$C_1 = G_0 + P_0 C_0$$

$$C_2 = G_1 + P_1 G_0 + P_1 P_0 C_0$$

$$C_3 = G_2 + P_2 G_1 + P_2 P_1 G_0 + P_2 P_1 P_0 C_0$$

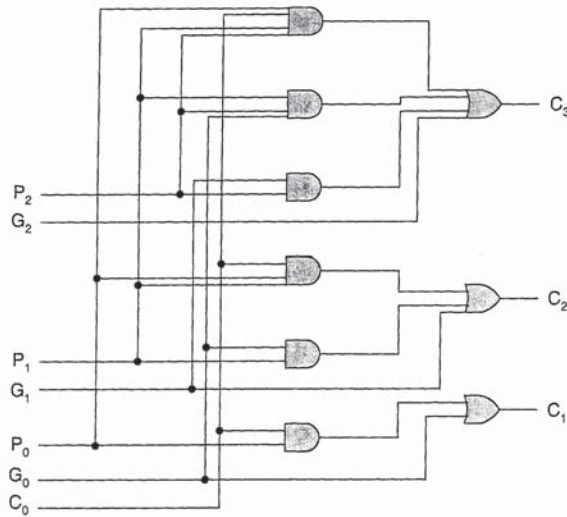
$$C_4 = G_3 + P_3 G_2 + P_3 P_2 G_1 + P_3 P_2 P_1 G_0 + P_3 P_2 P_1 P_0 C_0$$



Η γενική μορφή της σχέσης για το κρατούμενο είναι η παρακάτω:

$$C_{n+1} = G_n + P_n(G_{n-1} + P_{n-1}(G_{n-2} + P_{n-2}(\dots + P_2(G_1 + P_1C_1)))) = G_n + P_nG_{n-1} + P_nP_{n-1}G_{n-2} + P_nP_{n-1}P_{n-2}G_{n-3} + \dots + P_nP_{n-1}P_{n-2}\dots P_2G_1 + P_nP_{n-1}P_{n-2}\dots P_2P_1C_1$$

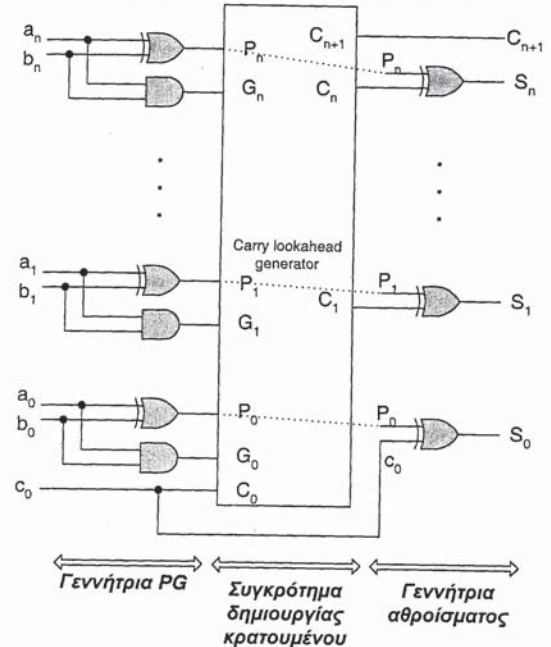
Σύμφωνα με τα παραπάνω μπορούμε να σχεδιάσουμε την γεννήτρια πρόβλεψης κρατουμένου για 4 bits σε επίπεδο πυλών όπως φαίνεται στο σχήμα 1.21β.



Σχήμα 1.21β Γεννήτρια πρόβλεψης κρατουμένου

Σχηματικά, ένας αθροιστής πρόβλεψης κρατουμένου αποτελείται από τρία συνδυαστικά μέρη. Το πρώτο μέρος περιλαμβάνει την γεννήτρια των P και G

σημάτων. Το δεύτερο μέρος που φαίνεται στο σχήμα 1.21β, αποτελεί το βασικό κύκλωμα του αθροιστή και ονομάζεται συγκρότημα δημιουργίας κρατουμένου. Το τελευταίο μέρος αποτελείται από πύλες XOR και είναι η γεννήτρια αθροίσματος. Στο σχήμα 1.22 φαίνεται η τμηματοποίηση που περιγράφηκε παραπάνω.



Σχήμα 1.22 Αθροιστής πρόβλεψης κρατουμένου

Πολλές μελέτες έχουν γίνει πάνω στον υπολογισμό των κρατούμενων με όσο το δυνατόν πιο αποδοτικό τρόπο. Μια από αυτές παρουσιάζεται στην επόμενη παράγραφο.

### Υλοποίηση αθροιστών πρόβλεψης κρατουμένου

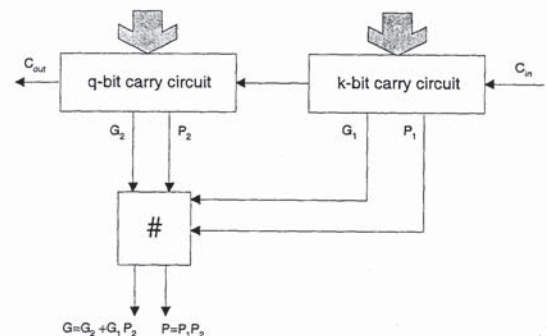
Το μοντέλο που παρουσιάζεται σε αυτήν την παράγραφο προτάθηκε από τους Brent και Kung. Έστω δύο θετικοί αριθμοί δυαδικοί  $X$  και  $Y$  που υπακούουν στην γνωστή μας σχέση

$$X = x_{n-1}x_{n-2}\dots x_0 = \sum_{i=0}^{n-1} x_i 2^i \text{ και } Y = y_{n-1}y_{n-2}\dots y_0 = \sum_{i=0}^{n-1} y_i 2^i$$

Για οποιαδήποτε σειρά από ζευγάρια ψηφίων  $(x_i, y_i)$  έχουμε δύο λογικές ποσότητες  $P$  και  $G$  όπως αναφέρθηκε και στην προηγούμενη παράγραφο. Πρόκειται για την γεννήτρια  $PG$  του σχήματος 1.22, όπου έχουμε στις εισόδους τα bits των εισερχόμενων αριθμών και στην έξοδο την παραγωγή των σημάτων διάδοσης και γέννησης κρατουμένου. Σε κάθε θέση bit, τα κρατούμενα εισόδου και εξόδου ικανοποιούν την σχέση  $C_{out} = C_{in}P + G$ . Ονομάζουμε το ζευγάρι  $(G, P)$  "προ-κρατούμενο" και το συμβολίζουμε ως  $\Gamma = (G, P)$ . Αξίζει να σημειωθεί ότι η νέα οντότητα αυτή είναι ανεξάρτητη από το κρατούμενο εισόδου και μπορεί να αφορά συνεχόμενα bits δύο αριθμών. Αν χωρίσουμε τα συνεχόμενα bits δύο εισαγόμενων αριθμών σε δύο υποσύνολα τότε μπορούμε να υπολογίσουμε το συνολικό προ-κρατούμενο, αν γνωρίζουμε τα προ-κρατούμενα των υποσυνόλων, χρησιμοποιώντας έναν καινούριο τελεστή που συμβολίζουμε με  $\#$  και ορίζουμε ως εξής:

$$(G, P) = \Gamma = \Gamma_2 \# \Gamma_1 = (G_2, P_2) \# (G_1, P_1) = (G_2 + G_1 P_2, P_1 P_2)$$

Στο σχήμα 1.22a φαίνεται ο διαχωρισμός των ψηφίων σε υποσύνολα και ο υπολογισμός των συνολικών σημάτων διάδοσης και γέννησης κρατουμένου.



Σχήμα 1.22a Ο συνδυασμός δύο σημάτων γέννησης και διάδοσης κρατουμένου σε ένα συνολικό σήμα.

Αποδεικνύεται ότι ο τελεστής  $\#$  είναι προσεταιριστικός όπως φαίνεται παρακάτω:

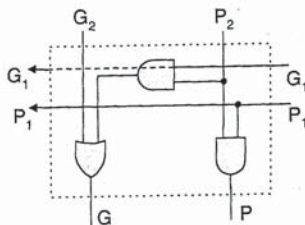
$$\Gamma_3 \# (\Gamma_2 \# \Gamma_1) = (G_3, P_3) \# (G_2 + G_1 P_2, P_1 P_2) = (G_3 + P_3 G_2 + P_2 P_3 G_1, P_1 P_2 P_3)$$

$$(\Gamma_3 \# \Gamma_2) \# \Gamma_1 = (G_3 + G_2 P_3, P_2 P_3) \# (G_1, P_1) = (G_3 + P_3 G_2 + P_2 P_3 G_1, P_1 P_2 P_3)$$

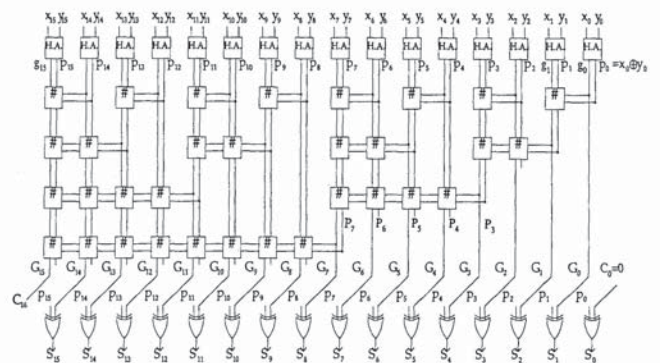
Η ιδιότητα αυτή είναι πολύ χρήσιμη γιατί επιτρέπει τον παράλληλο υπολογισμό των  $P_i$  και  $G_i$  (από την μηδενική τάξη μέχρι την  $i$ -τάξη) από τα  $p_i$  και  $g_i$  (που αναφέρονται σε ένα μόνο ζευγάρι bits) χρησιμοποιώντας ένα δυαδικό δέντρο με  $\log_2 n$  επίπεδα όπου  $n$  είναι ο αριθμός των ψηφίων των αριθμών που προσθέτουμε. Ξεκινώντας από το πρώτο επίπεδο, πρέπει να υπολογίσουμε το προ-κρατούμενο ενός ζευγαριού bit, το οποίο δίνεται από την σχέση  $\gamma_i = (g_i, p_i) = (x_i y_i, x_i \oplus y_i)$ . Παρατηρούμε ότι η προηγούμενη σχέση υλοποιείται από έναν ημιαθροιστή. Το συνολικό προ-κρατούμενο  $\Gamma$  προκύπτει από την σχέση  $\Gamma = (\gamma_1 \# (\gamma_2 \# (\dots (\gamma_{i-1} \# \gamma_i) \dots))) = (P_i, G_i)$

Τα bits του αθροίσματος όπως περιγράψαμε σε προηγούμενη παράγραφο δίνονται από τη σχέση  $s_i = x_i \oplus y_i \oplus c_i = p_i \oplus (G_{i-1} + P_{i-1} c_0)$ . Επομένως, αρκεί να υπολογίσουμε όλες τις ποσότητες  $G_i$ ,  $P_i$ . Ο πιο γρήγορος τρόπος υπολογισμού είναι η χρήση ενός

δυναμικού δέντρου που αποτελείται από κυκλώματα που υλοποιούν τον τελεστή #. Το κύκλωμα του τελευταίου φαίνεται στο σχήμα 1.23 ενώ ολόκληρος ο αθροιστής πρόβλεψης κρατούμενου παρουσιάζεται στο σχήμα 1.24. Στον αθροιστή αυτόν, θεωρούμε το αρχικό κρατούμενο εισόδου μηδέν ( $c_0=0$ ). Συνεπώς, η προηγούμενη σχέση του αθροίσματος γίνεται  $s = p_i \oplus G_{i-1}$ . Αν το  $c_0$  μπορεί να έχει οποιαδήποτε τιμή, τότε θα χρειαστεί μια βαθμίδα ακόμη για τον υπολογισμό της ποσότητας  $G_i + P_{i-1} \cdot c_0$ .



Σχήμα 1.23 Το κύκλωμα που υλοποιεί την πράξη "#"



Σχήμα 1.24 Αθροιστής 16-bit πρόβλεψης κρατούμενου

### 1.3 Αφαιρέτες

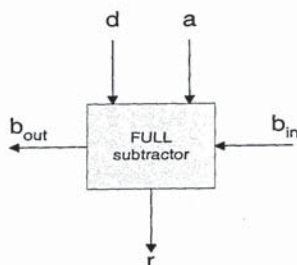
#### 1.3.1 Βασική δομική μονάδα

Σε αντιστοιχία με την ανάλυση που έγινε για το κύτταρο του πλήρη αθροιστή θα μπορούσαμε στο σημείο αυτό να αναλογιστούμε με ποιο τρόπο μπορεί να γίνει η αφαίρεση δύο δυαδικών ψηφίων κατασκευάζοντας το αρχικό κύτταρο του δυαδικού αφαιρέτη. Στη συνέχεια, θα μπορούσαμε να σκεφτούμε πώς το κύτταρο αυτό θα μπορούσε να χρησιμοποιηθεί σε κυκλώματα που θα εκτελούν την πράξη της αφαίρεσης μεταξύ δυαδικών αριθμών και όχι απλών ψηφίων.

Από τον ορισμό της αφαίρεσης υπαγορεύεται ότι το κύτταρο αυτό θα πρέπει να δέχεται τρεις εισόδους, δηλαδή τα δύο ψηφία των αριθμών (αφαιρετέος, εδώ συμβολικά  $a$  - αφαιρέτης, εδώ συμβολικά  $d$ ), και εφόσον το κύτταρο αυτό προορίζεται να χρησιμοποιηθεί και για την αφαίρεση αριθμών με περισσότερα του ενός ψηφία, θα πρέπει να δέχεται και το "δανεικό" από την αφαίρεση των ψηφίων της μικρότερης τάξης (συμβολικά  $b_n$ ). Οι έξοδοι του κυττάρου θα πρέπει να είναι δύο, εκ των οποίων



η μια πρέπει να αποτελεί το υπόλοιπο της αφαίρεσης που διενεργείται σε αυτό ( $r$ ), ενώ η δεύτερη θα αποτελεί το δανεικό που θα μεταφερθεί στο κύτταρο επόμενης τάξης ( $b_{out}$ ).



Σχήμα 1.25 Βασική δομική μονάδα αφαιρέτη

Από τα παραπάνω ανακύπτει ένα πρόβλημα που δεν υπήρχε στην πρόσθεση: στο κύτταρο του δυαδικού αφαιρέτη οι εισόδους δεν είναι ισοδύναμες. Στην περίπτωση του πλήρη αθροιστή, σε οποιαδήποτε από τις τρεις εισόδους και αν εφαρμόζαμε τα bits των δύο προσθετέων και του κρατούμενου, τόσο το άθροισμα όσο και το κρατούμενο εξόδου προέκυπταν σωστά. Στην περίπτωση όμως της εισόδου που προσδιορίζεται ως ψηφίο αφαιρετέου στο δυαδικό αφαιρέτη κάτι τέτοιο δεν ισχύει. Η είσοδος αυτή δεν είναι ισοδύναμη με καμία από τις δύο άλλες (δανεικό και bit αφαιρέτη), ενώ αυτές είναι μεταξύ τους ισοδύναμες. Αυτό είναι προφανές, επιβεβαιώνεται όμως πολύ απλά θέτοντας σαν εισόδους του κυττάρου  $a=1$ ,  $b=0$ ,  $b_{in}=1$ , οπότε οι εξοδοί είναι  $r=0$  και  $b_{out}=0$ , ενώ αν εναλλάξουμε τις τιμές των  $a$  και  $b$  οι εξοδοί γίνονται  $r=0$  και  $b_{out}=1$ . Κατά συνέπεια, κατά την κατασκευή του κυττάρου του δυαδικού αφαιρέτη, η μια από τις εισόδους του προσρίζεται αποκλειστικά για το bit του αφαιρετέου.

Με βάση την παραπάνω ανάλυση μπορούμε να συμπληρώσουμε τον πίνακα αληθείας για το κύτταρο του δυαδικού αφαιρέτη.

Πίνακας 1.3 Πίνακας αληθείας πλήρη αφαιρέτη

a	d	$b_{in}$	r	$B_{out}$
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

Παρατηρούμε ότι η έξοδος  $r$  του κυττάρου είναι περιττή συνάρτηση των εισόδων του. Άρα, όπως και στον πλήρη αθροιστή για το άθροισμα, πραγματοποιείται από την λογική πράξη XOR μεταξύ των τριών εισόδων του δηλαδή την ίδια λογική συνάρτηση με αυτή που παρουσιάσαμε στον πλήρη αθροιστή. Όσον αφορά στο δανεικό  $b_{out}$  μπορούμε να απλοποιήσουμε την προκύπτουσα συνάρτηση με τη βοήθεια χάρτη Karnaugh. Ο χάρτης αυτός είναι :

		ad			
		00	01	11	10
$b_{in}$	0	0	1	0	0
	1	1	1	1	0

από τον οποίο με τη γνωστή διαδικασία απλοποίησης προκύπτει η έκφραση για το  $b_{out}$  :

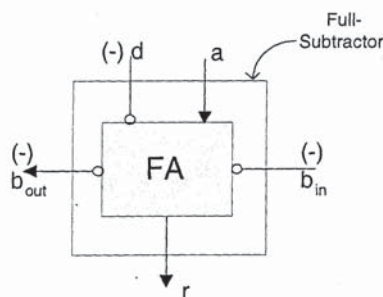
$$b_{out} = \bar{a} b_{in} + d b_{in} + \bar{a} d \text{ και } \bar{b}_{out} = \bar{a} \bar{b}_{in} + \bar{d} \bar{b}_{in} + \bar{a} \bar{d}$$

Αν συγκρίνουμε την έκφραση αυτή με την έκφραση που προέκυψε για το κρατούμενο εξόδου του πλήρους αθροιστή παρατηρούμε ότι η μόνη διαφορά τους έγκειται στην αντιστροφή που υπάρχει στο  $a$  του  $b_{out}$ . Εάν λοιπόν στο κύκλωμα του

πλήρης αθροιστή προσθέσουμε μια αντιστροφή σε μια από τις εισόδους του και αυτή η είσοδος παίζει το ρόλο του  $a$ , από την έξοδο του κρατούμενου του πλήρης αθροιστή θα πάρουμε το δανεικό του δυαδικού αφαιρέτη. Αντιστρέφοντας το  $a$  όμως επηρεάζουμε και το  $r$ . Επειδή όμως αυτό, σύμφωνα με την παρατήρηση που κάναμε πάνω στον πίνακα αλήθειας 1.3, δίνεται από την περιττή συνάρτηση των εισόδων, είναι εύκολο να κάνουμε την απαραίτητη μετατροπή ώστε να λύσουμε και αυτό το πρόβλημα: Αντιστρέφοντας την τιμή της μιας από τις τρεις μεταβλητές εισόδου, η έξοδος του αθροίσματος στον πλήρη αθροιστή δίνει τώρα '1' όπου πριν την αλλαγή δίνει '0' και το αντίστροφο κάτι που επιβεβαιώνεται και από τις ιδιότητες της πύλης XOR. Πράγματι, ισχύει  $\bar{a} \oplus d \oplus b = \bar{a} \oplus d \oplus \bar{b} = \bar{r}$ . Για να πάρουμε λοιπόν την σωστή έξοδο  $r$  από την έξοδο αθροίσματος του πλήρης αθροιστή με αντεστραμμένη τη μια είσοδό της, πρέπει να προσθέσουμε σε αυτή την έξοδο μια αντιστροφή.

Συνεπώς το τελικό λογικό διάγραμμα του κυττάρου του δυαδικού αφαιρέτη είναι όμοιο με αυτό του πλήρη αθροιστή με μόνη την προσθήκη τριών αντιστροφών, ενός στην είσοδο αρνητικής αξίας  $d$ , ενός στην είσοδο του κρατούμενου και ενός στην έξοδο του κρατούμενου, όπως φαίνεται και στο σχήμα 1.26.

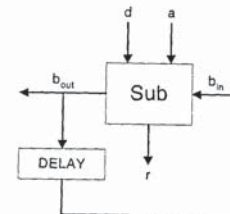
$$r = a \oplus d \oplus b_{in} = a \oplus \bar{d} \oplus \bar{b}_{in}$$



Σχήμα 1.26 Μετατροπή του πλήρη αθροιστή σε πλήρη αφαιρέτη

### 1.3.2 Σειριακός αφαιρέτης

Το κύτταρο του δυαδικού αφαιρέτη που μόλις περιγράψαμε μπορεί να χρησιμοποιηθεί για την κατασκευή ενός σειριακού αφαιρέτη, ο οποίος δέχεται τα ζεύγη των bit, (ίδιες τάξεις ξεκινώντας από την μηδενική τάξη), των δύο αριθμών (πάντα στην είσοδο  $a_i$  ο αφαιρετέος), και παράγει στην έξοδο  $r_i$  το υπόλοιπο και στην έξοδο  $b_{out}$  το τυχόν δανεικό για τη συγκεκριμένη τάξη, για όσο χρονικό διάστημα εφαρμόζονται τα αντίστοιχα bits στις εισόδους (ιδανικά, στην πραγματικότητα σε επίπεδο υλικού υπάρχουν περιορισμοί). Η τιμή του δανεικού θα πρέπει να αφαιρεθεί κατά την διαδικασία της αφαίρεσης που θα γίνει για τα bits της αμέσως επόμενης τάξης, που για το συγκεκριμένο τύπο αφαιρέτη σημαίνει ότι θα πρέπει να διατηρηθεί χρονικά για το κατάλληλο χρονικό διάστημα και να εφαρμοστεί στην είσοδο  $b_{in}$ . Σε αντιστοιχία με τον σειριακό αθροιστή που έχει ήδη αναφερθεί, θα πρέπει να κλείσουμε ένα βρόχο γύρω από το κύτταρο του δυαδικού αφαιρέτη με αφετηρία την έξοδο  $b_{out}$  και τερματισμό την είσοδο  $b_{in}$ , παρεμβάλλοντας μια καθυστέρηση που κρατάει το δανεικό για το απαραίτητο χρονικό διάστημα. Όπως όμως και με τον σειριακό αθροιστή, κατά την υλοποίηση τέτοιων κυκλωμάτων θα πρέπει να προσεχτούν τα θέματα του συγχρονισμού που ανακύπτουν από τη φύση της σχεδίασης.



Σχήμα 1.27 Σειριακός αφαιρέτης

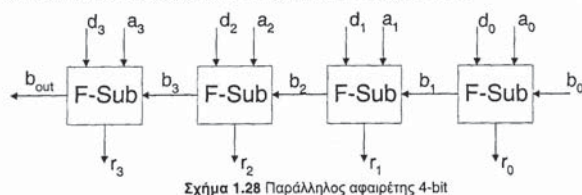
### 1.3.3 Παράλληλος αφαιρέτης

Το κύτταρο του δυαδικού αφαιρέτη μπορεί να χρησιμοποιηθεί για το σχηματισμό παράλληλου αφαιρέτη. Σε αντιστοιχία με τον παράλληλο αθροιστή,

χρησιμοποιούμε τόσα κύτταρα όσα είναι και τα ψηφία του αφαιρετέου (θα δούμε ότι μόνον όταν ο αφαιρετέος είναι μεγαλύτερος από τον αφαιρέτη το αποτέλεσμα που προκύπτει από τον παράλληλο, αλλά και από τον σειριακό αφαιρέτη, αποτελεί την πραγματική διαφορά των δύο αριθμών). Η αφαίρεση δύο δυαδικών αριθμών, όπως είναι γνωστό, υλοποιείται με την δημιουργία του συμπληρώματος ως προς 2 του αφαιρετέου και την δυαδική πρόσθεση των δύο αριθμών. Υπενθυμίζουμε, ότι το συμπλήρωμα ως προς 2 ενός αριθμού, σχηματίζεται αντιστρέφοντας τα ψηφία του και προσθέτοντας 1. Για την μετατροπή αυτή και την χρήση της στα ψηφιακά συστήματα VLSI για πράξεις με προσημασμένους αριθμούς θα αναφερθούμε εκτενώς στο δεύτερο κεφάλαιο. Στο παρακάτω παράδειγμα παρουσιάζεται ο παραπάνω αλγόριθμος υλοποίησης της αφαίρεσης:

Έστω οι αριθμοί  $1110_2 = 14_{10}$  και  $1001_2 = 9_{10}$ . Η αφαίρεσή τους θα πρέπει να δώσει τον αριθμό  $5_{10}$ , δηλαδή  $0101$  στο δυαδικό σύστημα. Σύμφωνα με τον αλγόριθμο, αντιστρέφουμε τον  $1001 \rightarrow 0110$  και προσθέτουμε '1', δηλαδή, ο τελικός δυαδικός αριθμός είναι  $0111$ . Αυτόν τον αριθμό τον προσθέτουμε στον  $1110$  οπότε το εξαγόμενο  $0101$  συμπίπτει με τον αριθμό που περιμέναμε.

Για να υλοποιήσουμε έναν παράλληλο αφαιρέτη, αρκεί να συνδέσουμε κύτταρα του δυαδικού αφαιρέτη, όπως φαίνεται στο σχήμα 1.28.

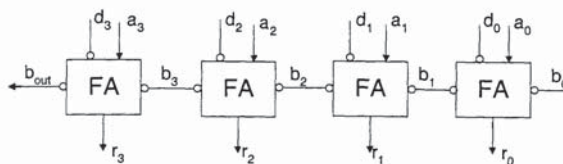


Σχήμα 1.28 Παράλληλος αφαιρέτης 4-bit

Όπως βλέπουμε υπάρχει ένα κύτταρο για κάθε ψηφίο του αφαιρετέου. Στο κάθε ένα από αυτά διοχετεύονται τα bits μιας συγκεκριμένης τάξης του αφαιρετέου και του αφαιρέτη καθώς και το  $b_{out}$  του κυττάρου της αμέσως κατώτερης τάξης. Η είσοδος  $b_n$  του κυττάρου μηδενικής τάξης πρέπει προφανώς να είναι πάντα μηδέν. Η έξοδος  $r_i$  του κάθε κυττάρου δίνει το αντίστοιχο bit του υπολοίπου όταν περάσει το χρονικό

διάστημα που απαιτείται ώστε όλες οι εισοδοί του να έχουν αποκτήσει την τελική τους τιμή.

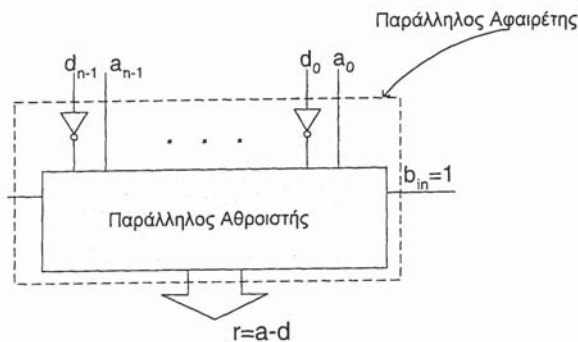
Αν "ανοίξουμε" τα κύτταρα του δυαδικού αφαιρέτη αντικαθιστώντας τα με αυτά του πλήρη αθροιστή, παρατηρούμε ότι μπορούμε να κάνουμε αρκετές απλοποιήσεις όπως φαίνεται και στο σχήμα 1.29.



Σχήμα 1.29 Υλοποίηση παράλληλου αφαιρέτη με πλήρεις αθροιστές και αντιστροφείς

Οι αντιστροφείς που βρίσκονται ανάμεσα στα κύτταρα μπορούν να απλοποιηθούν αφού αλληλοαναιρούνται. Η αντιστροφή που έχει το πρώτο κύτταρο στην είσοδο του κρατούμενου μπορεί επίσης να καταργηθεί, αφού το κρατούμενο εισόδου ήταν γνωστό και ίσο με το '0'. Αρκεί να θέσουμε την είσοδο αυτή στο λογικό '1' για να διώξουμε τον αντιστροφέα. Τελικά, διαπιστώνουμε ότι σε έναν παράλληλο αθροιστή, αν αντιστρέψουμε τα bits του αριθμού που αφαιρείται και θέσουμε το κρατούμενο εισόδου του πρώτου κυττάρου στο λογικό '1', τότε επιτυγχάνουμε την αφαίρεση των δύο αριθμών, όπου ο αντεστραμμένος παίζει το ρόλο του αφαιρετέου. Οι παραπάνω διαπιστώσεις φαίνονται στο σχήμα 1.30.





Σχήμα 1.30 Μετατροπή ενός παράλληλου αθροιστή σε παράλληλο αφαιρέτη

## 1.4 Πολλαπλασιαστές

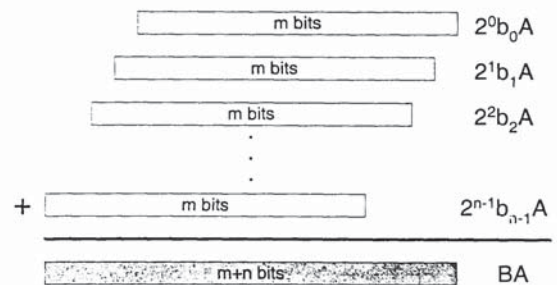
### 1.4.1 Εισαγωγή

Έστω ότι θέλουμε να πολλαπλασιάσουμε δύο δυαδικούς αριθμούς A και B. Ο A έχει m bits μήκος ενώ ο B n bits. Έστω ότι ο A είναι ο πολλαπλασιαστέος και ο B ο πολλαπλασιαστής. Μπορούμε να αναλύσουμε τον πολλαπλασιαστή ως εξής:

$$A \cdot B = A \cdot \{ b_0 + 2^1 b_1 + 2^2 b_2 + \dots + 2^{n-1} b_{n-1} \} = A \cdot \sum_{i=0}^{n-1} 2^i b_i$$

Στην ουσία συντελείται η λογική πράξη AND μεταξύ του κάθε bit του B με τον A και μετά προσθέτουμε τα επιμέρους γινόμενα. Στο σχήμα 1.30 φαίνεται η φιλοσοφία της λειτουργίας του πολλαπλασιασμού.

## 1.4 Πολλαπλασιαστές



Σχήμα 1.30 Πολλαπλασιασμός mXn

Τα μερικά γινόμενα φαίνονται στο δεξιό μέρος του σχήματος 1.30. Οι πολλαπλασιαστές διακρίνονται σε πολλές κατηγορίες ανάλογα:

με το είδος των αριθμών που προσθέτουμε (θετικοί ή προσημασμένοι).

με την κωδικοποίηση των αριθμών (Booth, Modified Booth, Signed Digit, Canonical Signed Digit)

με τον τρόπο πρόσθεσής τους (παράλληλα, σειριακά, σειριακά-παράλληλα)

με τον τρόπο που εισέρχονται τα bits των αριθμών στο κύκλωμα του πολλαπλασιαστή (LSB-first, MSB-first).

με τον ρυθμό λειτουργίας τους (πολλαπλασιαστές διοχέτευσης, συστολικοί)

Στο κεφάλαιο αυτό, θα γίνει η παρουσίαση της λειτουργίας των παράλληλων του σειριακού-παράλληλου και του σειριακού πολλαπλασιαστή για θετικούς αριθμούς. Θα εξετάσουμε βασικά δίκτυα παράλληλων πολλαπλασιαστών με κανονικό σχήμα (CSA, CLA και CPA πολλαπλασιαστές) καθώς και πολλαπλασιαστές που δεν έχουν κανονικό σχήμα που επιτρέπουν όμως μεγαλύτερες ταχύτητες. Θα γίνει εφαρμογή των πολλαπλασιαστών σε παραδείγματα, θα υπολογιστούν οι καθυστερήσεις και οι χρονισμοί και τέλος θα μετατραπούν σε συνεχούς διοχέτευσης και σε συστολικούς.

### 1.4.2 Παράλληλοι Πολλαπλασιαστές

Ο παράλληλος πολλαπλασιαστής βασίζεται στην παρατήρηση ότι τα μερικά γινόμενα μπορούν να υπολογιστούν ανεξάρτητα. Για παράδειγμα, θεωρήστε τους απρόσημους δυαδικούς ακέραιους αριθμούς  $A$  και  $B$  που δίνονται από τις σχέσεις

$$A = \sum_{i=0}^{m-1} a_i 2^i \text{ και } B = \sum_{j=0}^{n-1} b_j 2^j. \text{ Το γινόμενο δίνεται από τη σχέση}$$

$$P = A \times B = \sum_{i=0}^{m-1} a_i 2^i \cdot \sum_{j=0}^{n-1} b_j 2^j = \sum_{i=0}^{m-1} \sum_{j=0}^{n-1} (a_i b_j) 2^{i+j} = \sum_{k=0}^{m+n-1} p_k 2^k. \text{ Για αριθμούς των}$$

4-bits η παραπάνω έκφραση μπορεί να αναπτυχθεί όπως φαίνεται στον πίνακα 1.3.

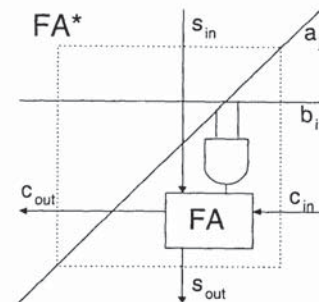
Πίνακας 1.3 Μερικά γινόμενα πολλαπλασιαστή των 4-bit

				$a_3$	$a_2$	$a_1$	$a_0$	Πολ/στέος
				$b_3$	$b_2$	$b_1$	$b_0$	Πολ/στής
				$a_3b_0$	$a_2b_0$	$a_1b_0$	$a_0b_0$	
			$a_3b_1$	$a_2b_1$	$a_1b_1$	$a_0b_1$		
		$a_3b_2$	$a_2b_2$	$a_1b_2$	$a_0b_2$			
	$a_3b_3$	$a_2b_3$	$a_1b_3$	$a_0b_3$				
$p_7$	$p_6$	$p_5$	$p_4$	$p_3$	$p_2$	$p_1$	$p_0$	Γινόμενο

### 1.4 Πολλαπλασιαστές

#### 1.4.2.1 Παράλληλος πολλαπλασιαστής με αθροιστή διάδοσης κρατούμενου

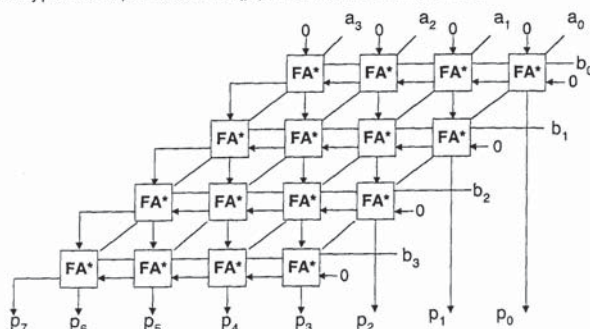
Ένας παράλληλος πολλαπλασιαστής αποτελείται γενικά από ένα δίκτυο παραγωγής των μερικών γινόμενων  $a_i b_j$  το οποίο αποτελείται από λογικές πύλες AND και έναν αθροιστή για την πρόσθεση των παραπάνω, στην συγκεκριμένη περίπτωση έναν carry propagate αθροιστή. Είναι αρκετά βολικό να εξετάσουμε τον πολλαπλασιασμό δύο τετράμπιτων αριθμών  $A$  και  $B$ . Η βασική δομική μονάδα του κυκλώματος φαίνεται στο σχήμα 1.31. Παρατηρούμε ότι το κύτταρο αποτελείται από έναν πλήρη αθροιστή και μια πύλη AND που την ενσωματώνουμε μέσα στο κύτταρο για ομοιομορφία. Η πύλη αυτή χρησιμεύει στην δημιουργία του ενδιάμεσου γινομένου των δύο bits των αριθμών, το οποίο εισέρχεται στον πλήρη αθροιστή για να προστεθεί με το ενδιάμεσο γινόμενο και το κρατούμενο της προηγούμενης βαθμίδας.



Σχήμα 1.31 Η βασική δομική μονάδα του πολλαπλασιαστή με διάδοση κρατούμενου

Τα σήματα του κυττάρου έχουν όλα μια συγκεκριμένη φορά, εκτός από τα  $a_i$  και  $b_i$ . Τα σήματα αυτά μπορούν να μπαίνουν είτε από την μια είτε από την άλλη πλευρά του κυττάρου χωρίς να επηρεάζουν τη λειτουργία του. Η διακεκομμένη γραμμή συμβολίζει τα εξωτερικά όρια του κυττάρου που ονομάζουμε  $FA^*$ .

Με βάση την βασική δομική μονάδα του σχήματος 1.31, μπορούμε πλέον να κατασκευάσουμε τον πολλαπλασιαστή με διάδοση κρατουμένου. Παρατηρούμε, ότι η διάταξη των κυττάρων είναι σε πλήρη αντιστοιχία με τον πίνακα 1.3.



Σχήμα 1.32 Παράλληλος πολλαπλασιαστής με διάδοση κρατουμένου

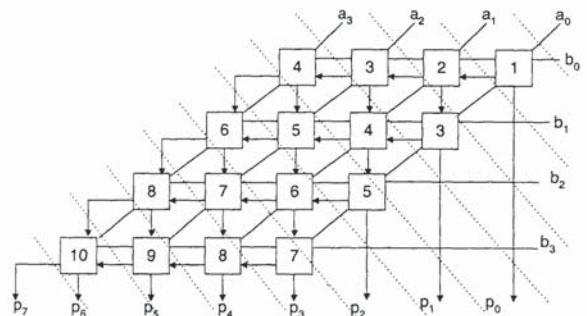
Για τον υπολογισμό της καθυστέρησης του κυκλώματος, πρέπει να βρούμε το αριθμό των κυττάρων που αποτελούν το κρίσιμο μονοπάτι και να τον πολλαπλασιάσουμε με την καθυστέρηση του ενός FA\*. Στην περίπτωση αυτή, τίθεται το ερώτημα, αν πρέπει να υπολογίσουμε εκ νέου την καθυστέρηση της βασικής δομικής μονάδας ή να χρησιμοποιήσουμε την καθυστέρηση ενός πλήρη αθροιστή αφού όπως φαίνεται και στο σχήμα 1.9 το κύτταρο περιέχει μια πύλη AND παραπάνω. Η απάντηση είναι προφανής αν εξετάσουμε την διαδρομή των εισόδων της πύλης AND. Παρατηρούμε, ότι τα σήματα των  $a_i$  και  $b_i$  απλώς διαδίδονται μέσα στο δίκτυο του πολλαπλασιαστή, χωρίς να παρεμβάλλεται καθυστέρηση. Είναι λοιπόν επόμενο, όταν τα  $s_m$  και τα  $c_m$  φτάσουν στο κύτταρο, να έχει ήδη εκτελεστεί η λογική πράξη AND. Συνεπώς, η καθυστέρηση της πύλης χρειάζεται να ληφθεί υπ' όψιν μόνο στο πάνω-δεξιά γωνιακό κύτταρο.

Το κρίσιμο μονοπάτι του κυκλώματος του σχήματος 1.32 περιλαμβάνει  $m + (n-1) + (n-1)$  κύτταρα και επειδή στο παράδειγμά μας ισχύει  $n=m=4$ , η μέγιστη

καθυστέρηση θα είναι ίση με την καθυστέρηση ενός κυττάρου (και κατ' επέκταση του πλήρη αθροιστή) επί 10.

Μπορούμε να παρατηρήσουμε ότι το κρίσιμο μονοπάτι δεν είναι μοναδικό αλλά οποιαδήποτε διαδρομή με οριζόντιες και κατακόρυφες γραμμές από το πάνω-δεξιά μέχρι και το κάτω-αριστερά γωνιακό κύτταρο και να ακολουθήσουμε θα έχουμε την ίδια καθυστέρηση των 10 πλήρων αθροιστών. Αυτό βέβαια οφείλεται στην κανονικότητα του κυκλώματος. Ένα από τα κρίσιμα μονοπάτια είναι γραμμοσκιασμένο στο σχήμα 1.32. Στον ίδιο αριθμό κυττάρων καταλήγουμε όμως αν διατρέξουμε το κύκλωμα βαθμωτά. Πιο αναλυτικά, η οριζόντια σειρά του μονοπατιού αποτελείται από  $m$  κύτταρα. Η κατακόρυφη έχει  $n-1$  αφού το πρώτο κύτταρο συμπεριλήφθηκε στην οριζόντια. Τέλος, η υπόλοιπη οριζόντια διαδρομή έχει πάλι  $n-1$  κύτταρα όπως φαίνεται και στο σχήμα 1.32. Η συνολική καθυστέρηση δίνεται από τον τύπο  $T=(m+n-1+n-1) \cdot T_{FA} \Rightarrow T=(2n+m-2) \cdot T_{FA}$  όπου  $T_{FA}$  ο χρόνος καθυστέρησης του πλήρη αθροιστή η οποία θεωρούμε ότι είναι ίδια και για τις δύο εξόδους του.

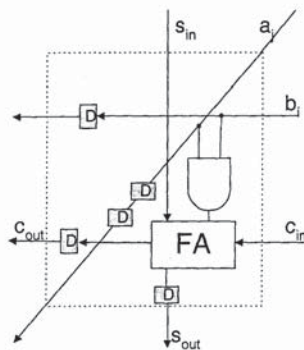
Η μετατροπή του πολλαπλασιαστή σε συνεχούς διοχέτευσης και σε αυτή την περίπτωση είναι ταυτόσημη με την μετατροπή του κυκλώματος σε συστολικό. Αρκεί να παρεμβάλουμε μανδαλωτές στα σημεία τομής των σημάτων και των διακεκομμένων γραμμών, όπως φαίνεται στο σχήμα 1.33.



Σχήμα 1.33 Μετατροπή του ripple carry πολλαπλασιαστή σε συνεχούς διοχέτευσης και συστολικό



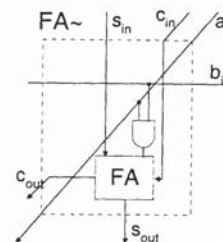
Ο χρονισμός του κυκλώματος διακρίνεται στα κύτταρά του. Κάθε φέτα που σχηματίζεται από δύο διαδοχικές διακεκομμένες γραμμές περιλαμβάνει κύτταρα που είναι συγχρονισμένα και δεν ανταλλάσσουν σήματα μεταξύ τους. Αξίζει να σημειωθεί ότι ο τρόπος τομής που επιλέξαμε είναι κοινός και για συστολικότητα και για συνεχή διοχέτευση για τον απλό λόγο ότι είναι ο μόνος τρόπος για να "κόψουμε" τα σήματα του κάθε κυττάρου που περνούν μέσα από λογική. Παράλληλα με τον τρόπο αυτό δεν αφήνουμε κανένα σήμα χωρίς στοιχείο καθυστέρησης πράγμα που προϋποθέτει η συστολικότητα. Παρατηρούμε ότι κάποια σήματα κόβονται δυο και τρεις φορές. Ανάλογα με το πλήθος τομών ενός σήματος, βάζουμε τα αντίστοιχα στοιχεία καθυστέρησης. Από το σχήμα 1.33 προκύπτει ότι στις γραμμές  $b$ ,  $s_{out}$  και  $c_{out}$  χρειάζεται ένας μανδαλωτής ενώ στη γραμμή  $a$  δύο. Συνηθίζουμε να βάζουμε τα στοιχεία καθυστέρησης στην έξοδο των γραμμών, μέσα στην βασική δομική μονάδα του κάθε κυκλώματος. Η μορφή του συστολικού κυττάρου φαίνεται στο σχήμα 1.34.



Σχήμα 1.34 Δομική μονάδα του συστολικού πολλαπλασιαστή διάδοσης κρατούμενου

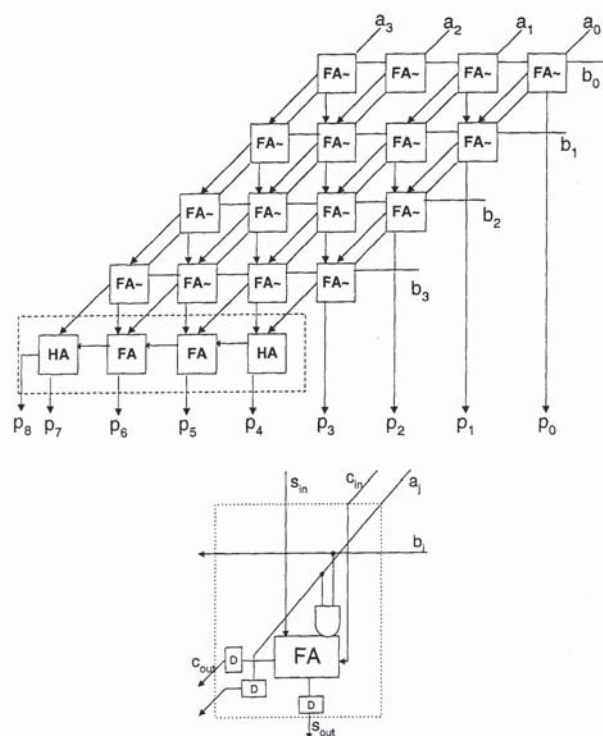
#### 1.4.2.2 Παράλληλος πολλαπλασιαστής με αθροιστή αποθήκευσης κρατούμενου.

Πρόκειται για έναν αρκετά διαδεδομένο τύπο πολλαπλασιαστή. Η γενικότερη φιλοσοφία υλοποίησης είναι παρόμοια με την προηγούμενη, μόνο που τώρα, τα κρατούμενα δεν διαδίδονται στο επόμενο κύτταρο του ίδιου επιπέδου, αλλά οδηγούνται στο επόμενο επίπεδο. Η βασική δομική μονάδα του πολλαπλασιαστή δεν αλλάζει ως προς την ουσία αλλά ως προς την διάταξη των γραμμών, ονομάζεται FA~ και φαίνεται στο σχήμα 1.35.



Σχήμα 1.35 Δομική μονάδα του πολλαπλασιαστή με σώσιμο κρατούμενου

Η μορφή του δικτύου φαίνεται στο σχήμα 1.36.



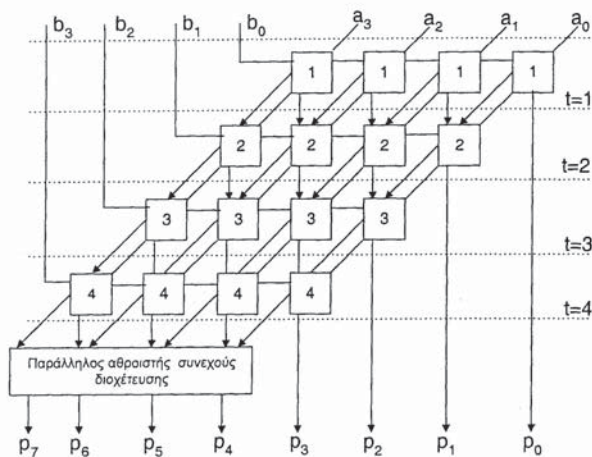
Σχήμα 1.36 Παράλληλος πολλαπλασιαστής 4X4 με σώσιμο κρατούμενου.

Στο τέλος της τέταρτης βαθμίδας έχουμε τα τέσσερα πιο σημαντικά bits του αποτελέσματος σε μορφή αθροίσματος-κρατούμενου. Για τον λόγο αυτό, πρέπει να χρησιμοποιήσουμε έναν απλό τετράμπιτο αθροιστή με διάδοση κρατούμενου για την

τελική πρόσθεση. Την ίδια δουλειά βέβαια, μπορούμε να την κάνουμε ταχύτερα και με έναν αθροιστή πρόβλεψης κρατούμενου όπως περιγράφηκε στην προηγούμενη ενότητα. Στον τελικό αθροιστή (της τελευταίας σειράς) δεν εισέρχονται τα σήματα  $a_i$  και  $b_i$  γιατί δεν παίρνουν μέρος στην πρόσθεση των κρατουμένων και των μερικών γινόμενων.

Ο χρόνος καθυστέρησης του πολλαπλασιαστή με σώσιμο κρατούμενου, είναι άμεσα εξαρτώμενος από τον τύπο του αθροιστή της τελευταίας βαθμίδας. Σε περίπτωση που χρησιμοποιούμε αθροιστή διάδοσης κρατούμενου η καθυστέρηση δίνεται από την σχέση:  $T = n \cdot T_{FA} + m \cdot T_{FA} \Rightarrow T = (n+m) \cdot T_{FA}$ , όπου  $n \cdot T_{FA}$  είναι η καθυστέρηση του αθροιστή της τελευταίας βαθμίδας, ενώ  $m \cdot T_{FA}$  του υπολοίπου δικτύου. Όπως αναφέρθηκε παραπάνω, η καθυστέρηση λόγω της πύλης AND μέσα στα κύτταρα δεν λαμβάνεται υπ' όψιν. Σε περίπτωση που χρησιμοποιήσουμε αθροιστή πρόβλεψης κρατούμενου, έχουμε ακόμη πιο μικρή καθυστέρηση:  $T = (2 + \log_2 n) \cdot T_{FA} + m \cdot T_{FA} \Rightarrow T = (\log_2 n + m + 1) \cdot T_{FA}$ . Παρατηρούμε ότι για  $n=m=4$  οι καθυστερήσεις των δύο πολλαπλασιαστών, με CPA και CSA αντίστοιχα, είναι ίσες. Η χρήση του αθροιστή πρόβλεψης έχει νόημα όταν το  $n$  έχει μεγάλη τιμή.

Το κύκλωμα του σχήματος 1.36 μπορεί να γίνει συνεχούς διαχείτευσης με την γνωστή μέθοδο. Προσπαθούμε να "κόψουμε" με γραμμές τα σήματα που βγαίνουν από κάθε κύτταρο και παίρνουν μέρος σε συνδυαστική λογική. Για να το επιτύχουμε αυτό, αρκεί να κόψουμε το κύκλωμα όπως φαίνεται στο σχήμα 1.37.

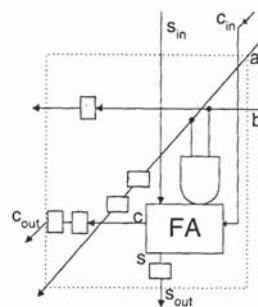


Σχήμα 1.37 Μετατροπή του παράλληλου πολλαπλασιαστή με σώσιμο κρατούμενου σε συνεχούς διοχέτευσης.

Οι διακεκομμένες γραμμές δείχνουν τα σημεία στα οποία πρέπει να παρεμβληθούν οι μανδαλωτές. Οι χρονισμοί φαίνονται μέσα στις δομικές μονάδες του δικτύου. Βέβαια στην συγκεκριμένη περίπτωση παρατηρούμε ότι υπάρχει πρόβλημα με τα σήματα  $b$ , αφού ο συγχρονισμός δεν επιτυγχάνεται εύκολα. Όπως φαίνεται και στο σχήμα 1.37 μπορούμε να εισάγουμε τα σήματα  $b$  από πάνω έτσι ώστε να τέμνονται και αυτά από τις διακεκομμένες γραμμές και επομένως να συγχρονίζονται με τα υπόλοιπα σήματα του κυκλώματος. Ο αριθμός  $B$  μπορεί να εισαγεί και από τη δεξιά πλευρά του κυκλώματος χωρίς να αλλάξει τίποτε στη λειτουργία του. Ας εξετάσουμε την λειτουργία του πολλαπλασιαστή από πιο κοντά. Την χρονική στιγμή 1 εισέρχονται τα bits  $a_0, a_1, a_2, a_3$  και  $b_0$  στο πρώτο επίπεδο του κυκλώματος όπου και γίνονται οι προσθέσεις ενώ τα bits  $b_1, b_2, b_3$  κλειδώνονται στο δεύτερο επίπεδο

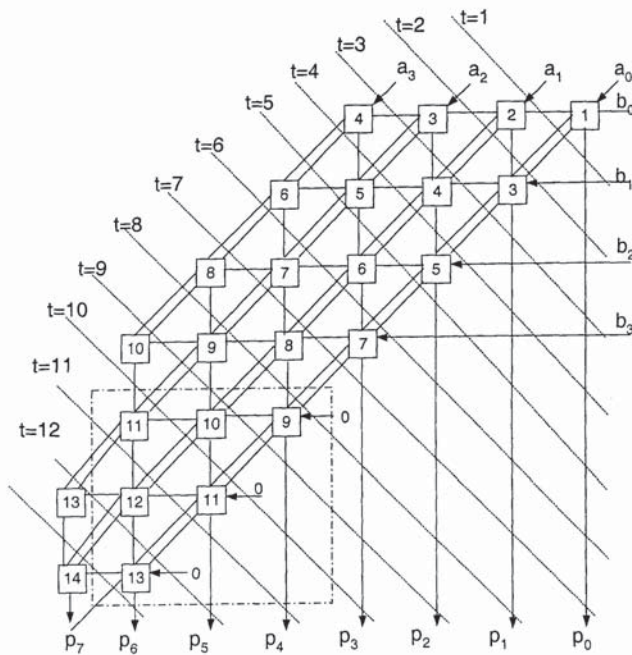
μανδαλωτών μαζί με τα αποτελέσματα από τις λογικές πράξεις της πρώτης σειράς των κυττάρων. Στην επόμενη χρονική στιγμή μπορεί να έρθει το νέο ζευγάρι αριθμών χωρίς να επηρεάζει τα αποτελέσματα των προηγούμενων αριθμών. Ο χρονισμός του κυκλώματος δηλώνεται με αριθμούς μέσα στα κύτταρα του σχήματος 1.37.

Ωστόσο, ο τρόπος τομής του κυκλώματος που παρουσιάζεται παραπάνω δεν είναι ο κατάλληλος για την μετατροπή του πολλαπλασιαστή σε συστολικό. Αυτό είναι εύκολο να αποδειχθεί, αν εξετάσουμε τις γραμμές  $b$  ανάμεσα στα κύτταρα του κυκλώματος. Τα σήματα αυτά δεν τέμνονται από καμία διακεκομμένη γραμμή, και συνεπώς δεν παρεμβάλλεται μανδαλωτής πράγμα που δεν επιτρέπεται στα συστολικά δίκτυα. Είναι ανάγκη λοιπόν, να βρούμε άλλον τρόπο για να "κόψουμε" το κύκλωμα. Οι τομές του σχήματος 1.39, υπακούουν στους κανόνες της συστολικότητας και η δομή του συστολικού κυττάρου που προκύπτει φαίνεται στο σχήμα 1.38.



Σχήμα 1.38 Δομική μονάδα συστολικού δικτύου παράλληλου πολλαπλασιαστή με σώσιμο κρατούμενου





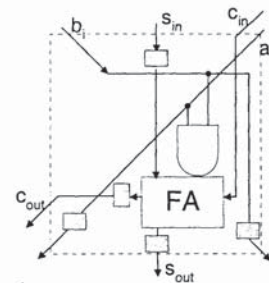
Σχήμα 1.39 Μετατροπή του παράλληλου πολλαπλασιαστή με σώσιμο κρατούμενου σε συστολικό δίκτυο.

Στο σχήμα 1.39, αντικαταστάθηκε το τελευταίο επίπεδο του αθροιστή διάδοσης κρατούμενου, με ένα τριγωνικό δίκτυο από κύτταρα όμοια με αυτά του υπόλοιπου σχήματος. Αυτό βέβαια είναι απαραίτητο για να ισχύει η ομοιομορφία. Εξετάζοντας πόσες φορές τέμνονται τα σήματα από διακεκομμένες γραμμές μπορούμε να καθορίσουμε τον αριθμό των μανδαλωτών που θα παρεμβάλουμε για να

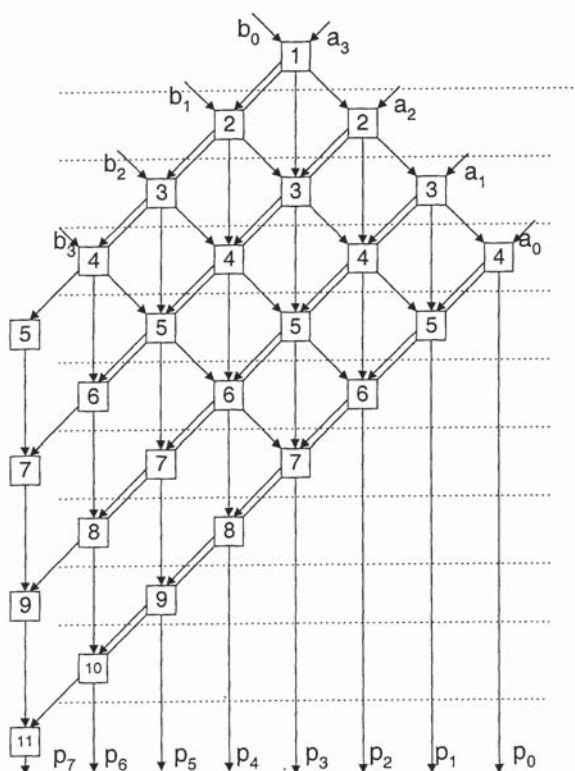
γίνει το κύκλωμα συστολικό. Στο συγκεκριμένο παράδειγμα παρατηρούμε ότι σε κάθε βασική δομική μονάδα, τα σήματα  $b$  και  $s_{out}$  θέλουν έναν μανδαλωτή, ενώ τα σήματα  $c_{out}$  και  $a$  χρειάζονται δύο πράγματα που εξηγεί και τη δομή του κυττάρου. Οι χρονικές στιγμές που ενεργοποιείται κάθε κύτταρο δηλώνονται από τους αριθμούς μέσα σε αυτά.

Ο παραπάνω τρόπος μετατροπής του παράλληλου πολλαπλασιαστή με σώσιμο κρατούμενου σε συστολικό δίκτυο δεν είναι ο μοναδικός. Ένας ακόμη τρόπος προτάθηκε από τον McCanny, όπως φαίνεται στο σχήμα 1.41.

Ο χρονισμός κάθε κυττάρου του δικτύου αναγράφεται μέσα σε αυτό. Παρατηρούμε ότι η πρώτη πράξη αρχίζει με το MSBit του πολλαπλασιαστέου. Τα bits του πολλαπλασιαστή  $b$  εισέρχονται στο κύκλωμα με αντίστροφη διάταξη από αυτή του σχήματος 1.39. Παρατηρούμε επίσης, ότι χρειάζονται μόνο 10 χρονισμοί για να εξάγουμε το αποτέλεσμα, τρεις λιγότεροι από τον παράλληλο πολλαπλασιαστή του σχήματος 1.39. Σε λειτουργία συνεχούς διοχέτευσης, η καθυστέρηση εξαγωγής κάθε αποτελέσματος περιορίζεται μόνο από την καθυστέρηση ενός κυττάρου και ενός μανδαλωτή. Τα σήματα  $b$  και  $a$  κόβονται μια φορά από διακεκομμένες γραμμές, άρα χρειάζονται από έναν μανδαλωτή. Αντίθετα, το σήμα του αθροίσματος  $s$  διακόπτεται δύο φορές, επομένως χρειάζονται δύο μανδαλωτές. Η δομή του βασικού κυττάρου φαίνεται στο σχήμα 1.40.



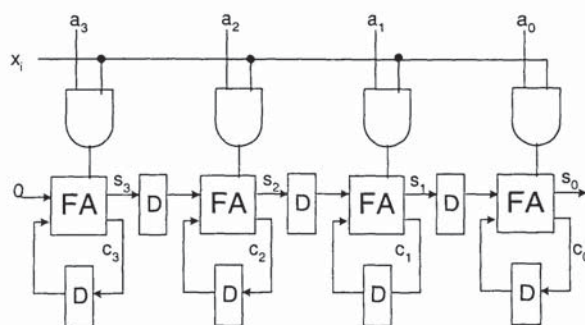
Σχήμα 1.40 Δομική μονάδα παράλληλου πολλαπλασιαστή McCanny



Σχήμα 1.41 Άλλος τρόπος μετατροπής παράλληλου πολλαπλασιαστή με σώσιμο κρατουμένου σε συστατικό.

### 1.4.3 Σειριακός- Παράλληλος Πολλαπλασιαστής

Στο κύκλωμα αυτό εισέρχεται ο πολλαπλασιαστέος  $A$  παράλληλα ενώ ο πολλαπλασιαστής  $X$  σειριακά. Αποτελείται από πύλες AND, πλήρεις αθροιστές και μανδαλωτές. Η συνδεσμολογία φαίνεται στο σχήμα 1.42.



Σχήμα 1.42 Σειριακός- παράλληλος πολλαπλασιαστής

Τα κρατούμενα κάθε πλήρη αθροιστή, αποθηκεύονται σε στοιχεία καθυστέρησης και επαναπροστίθενται με τον επόμενο κύκλο ρολογιού, στο νέο ζευγάρι  $x_i a_j$ . Τα ενδιάμεσα αθροίσματα διαδίδονται από τα αριστερά προς τα δεξιά περνώντας μέσα από μανδαλωτές. Τα bits του αριθμού  $a$  διατηρούνται στις εισόδους των πυλών AND ενώ με κάθε παλμό ρολογιού, εισάγονται τα bits του αριθμού  $x$  ξεκινώντας από το  $x_0$ . Αφού σχηματιστούν τα μερικά γινόμενα ( $a_i x_j$ ) και προστεθούν στους αντίστοιχους πλήρεις αθροιστές με κρατούμενα και αθροίσματα από προηγούμενη βαθμίδα, ολισθαίνουν κατά μια θέση προς τα δεξιά, αφήνοντας θέση για τα επόμενα μερικά γινόμενα.

Παρατηρούμε, ότι το κύκλωμα του σχήματος 1.37 και αυτό του σχήματος 1.42 έχουν κάποια ομοιότητα όχι τόσο σχηματικά όσο λειτουργικά. Πιο συγκεκριμένα, υπάρχει μια χρονική αντιστοίχιση του σειριακού-παράλληλου και του παράλληλου

πολλαπλασιαστή με σώσιμο κρατούμενου. Στο πρώτο στάδιο, στον παράλληλο πολλαπλασιαστή με σώσιμο κρατούμενου, παράγονται τα ενδιάμεσα γινόμενα  $a_3x_0$ ,  $a_2x_0$ ,  $a_1x_0$ ,  $a_0x_0$ . Το ίδιο ακριβώς γίνεται και στον πρώτο κύκλο ρολογιού του σειριακού-παράλληλου πολλαπλασιαστή, όπου τα αποτελέσματα αποθηκεύονται στους μανδαλωτές. Στο δεύτερο στάδιο, στο σχήμα 1.37, δεδομένα και κρατούμενα οδηγούνται προς τα κάτω ενώ παράλληλα ολισθαίνουν και μια θέση αριστερά. Εξετάζοντας το σχήμα 1.42, παρατηρούμε ότι με τον δεύτερο κύκλο ρολογιού, η σειρά των κυττάρων βρίσκεται ακριβώς στην ίδια κατάσταση με το δεύτερο επίπεδο του παράλληλου, όσον αφορά τις εισόδους και εξόδους. Το ίδιο ισχύει και για τα υπόλοιπα επίπεδα του παράλληλου. Καταλήγουμε δηλαδή στο συμπέρασμα ότι το σχήμα 1.37 είναι το χρονικό ξεδίπλωμα του σειριακού-παράλληλου πολλαπλασιαστή, πράγμα που αναφέρθηκε και στο προηγούμενο κεφάλαιο.

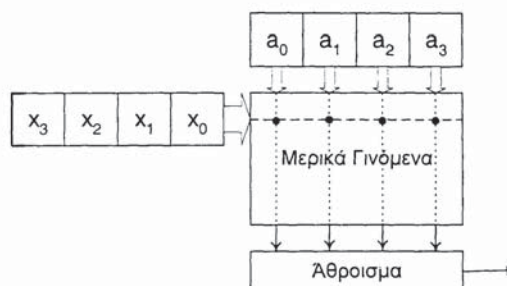
Το πρώτο αποτέλεσμα του γινομένου ( $p_0$ ) μπορούμε να το πάρουμε στον πρώτο παλμό ρολογιού όπου και εισάγουμε το πρώτο bit του  $X$  ( $x_0$ ). Με κάθε παλμό παίρνουμε και ένα αποτέλεσμα στην έξοδο του δεξιότερου πλήρη αθροιστή. Για τον συγκεκριμένο τετράμπιτο αθροιστή χρειαζόμαστε οκτώ παλμούς ρολογιού για το ολικό γινόμενο. Γενικότερα, σε ένα  $n \times m$  πολλαπλασιαστή, χρειαζόμαστε  $m$  κύτταρα (όπου  $m$  ο αριθμός των bits του πολλαπλασιαστέου) και στους  $n$  πρώτους παλμούς εισάγεται ο πολλαπλασιαστής και εξάγονται τα  $n$  LSB του γινομένου. Στους επόμενους  $n$  παλμούς εισάγονται μηδενικά bit στην είσοδο  $A$  και ο πολλαπλασιαστής απλώς αδειάζει αφού επιτελέσει την τελική πρόσθεση μεταξύ αθροισμάτων και κρατουμένων το αποτέλεσμα. Η φάση αυτή αντιστοιχεί στο παράλληλο αθροιστή του σχήματος 1.19.

### Σειριακός Πολλαπλασιαστής διοχέτευσης

Όπως έχουμε δει και σε προηγούμενες παραγράφους, στον πολλαπλασιασμό δύο δυαδικών αριθμών τα μερικά γινόμενα ίδιας τάξεως είναι:

$2^0$ τάξης	$a_0x_0$
$2^1$ τάξης	$a_0x_1 + a_1x_0$
$2^2$ τάξης	$a_0x_2 + a_1x_1 + a_2x_0$
$2^3$ τάξης	$a_0x_3 + a_1x_2 + a_2x_1 + a_3x_0$
$2^4$ τάξης	$a_0x_4 + a_1x_3 + a_2x_2 + a_3x_1 + a_4x_0$

Παρατηρούμε ότι οι συντελεστές  $a_i$  είναι οι ίδιοι κατά στήλη. Αντίστοιχα, ο κάθε συντελεστής  $x_i$  ολισθαίνει προς τα δεξιά, ενώ την κενή αρχική θέση καταλαμβάνει ο καινούριος συντελεστής  $x_{i+1}$ . Οι παραπάνω συνδυασμοί των μερικών γινομένων μπορούν να προκύψουν από το εξής σχήμα:

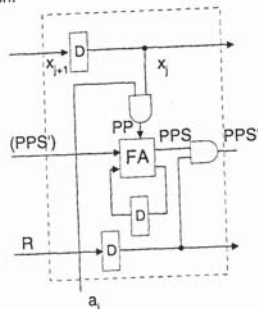


Σχήμα 1.43 Παραγωγή ίδιου βάρους μερικών γινομένων του πολλαπλασιασμού

Παρατηρούμε η διάταξη των συντελεστών  $a_i$  είναι διαφορετική από αυτή του σειριακού - παράλληλου πολλαπλασιαστή της προηγούμενης παραγράφου. Σε κάθε μετακίνηση των  $x_i$  ψηφίων προς τα δεξιά, παράγονται τα μερικά γινόμενα προς τα



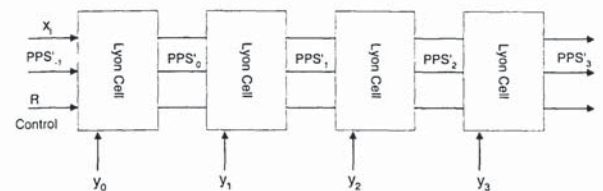
κάτω από τους επικαλυπτόμενους συντελεστές. Εφαρμογή της παραπάνω διάταξης παρουσιάζεται στον σειριακό πολλαπλασιαστή διοχέτευσης του Lyon. Ο πολλαπλασιαστής αυτός, περικόπτει τα μισά ψηφία του αποτελέσματος ξεκινώντας βέβαια από το LSB, στρογγυλεύοντας έτσι το αποτέλεσμα. Ένα χαρακτηριστικό παράδειγμα εφαρμογής της περικοπής γινόμενου είναι τα ψηφιακά φίλτρα, όπου ενδιαφέρει μόνο το ακέραιο τμήμα του παραγόμενου αποτελέσματος. Για την υλοποίηση της αποκοπής και στρογγυλοποίησης του γινόμενου, χρησιμοποιείται ένα ειδικό σήμα που μηδενίζει τα bits του μερικού γινόμενου που δεν χρειάζονται. Αντίστοιχα, για την υλοποίηση της λειτουργίας της διοχέτευσης, εισάγονται μανδαλωτές στην σειριακή είσοδο. Στο σχήμα 1.44 φαίνεται το βασικό κύτταρο του πολλαπλασιαστή του Lyon.



Σχήμα 1.44 Βασικό κύτταρο πολλαπλασιαστή Lyon

Παρατηρούμε ότι υπάρχουν κάποιες διαφορές σε σχέση με το βασικό κύτταρο του σειριακού-παράλληλου πολλαπλασιαστή του σχήματος 1.41. Η ροή των ψηφίων που εισέρχονται σειριακά (D) διακόπτεται από έναν μανδαλωτή. Η παράλληλη είσοδος των bits γίνεται από την βάση του κυττάρου ( $y_i$ ). Τέλος, το σήμα R και η λογική πράξη AND με το άθροισμα του πλήρη αθροιστή έχει σκοπό την αποκοπή των μερικών γινομένων μικρότερης αξίας. Σε κάθε νέο βήμα υπολογίζεται το μερικό γινόμενο

διπλάσιου βάρους για αυτό και τα κρατούμενα του κάθε πλήρη αθροιστή φυλάσσονται και συγχρονίζονται με τα μερικά γινόμενα του επόμενου βήματος. Στο σχήμα 1.44, φαίνεται ολόκληρος ο πολλαπλασιαστής συνεχούς διοχέτευσης του Lyon:



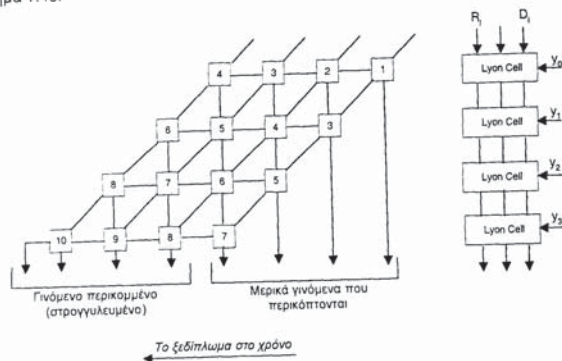
Σχήμα 1.45 Σειριακός πολλαπλασιαστής διοχέτευσης Lyon

Το σήμα R διαδίδεται σειριακά, όπως και τα δεδομένα D. Εισάγουμε στην είσοδο R μια λέξη που ξεκινάει με "0" και στις άλλες τρεις θέσεις έχει "1". Με αυτόν τον τρόπο, αποκόπτουμε στην πρώτη χρονική στιγμή, το μερικό γινόμενο του πρώτου κυττάρου. Το τινούμενο σήμα  $PPS_i$  ( $PPS'_i$ ) αποτελεί το αποτέλεσμα της πράξης του μερικού γινόμενου με το σήμα R. Την δεύτερη χρονική στιγμή, το λογικό "0" του σήματος R βρίσκεται στο δεύτερο κύτταρο, όπου και αποκόπτεται το  $PPS_{i+1}$ . Το "0" του σήματος R "ταξιθεύει" με τα ψηφία του D αποκόπτοντας τα μερικά γινόμενα που δεν χρειάζονται. Το πλεονέκτημα της αποκοπής των LSBs του γινόμενου είναι ότι δεν χρειάζεται στο τέλος μια σειρά από "0" για να πάρουμε όλο το αποτέλεσμα, όπως συμβαίνει στον σειριακό-παράλληλο πολλαπλασιαστή. Την θέση της σειράς των λογικών "0" μπορεί ο καινούριος αριθμός που εισέρχεται σειριακά στον πολλαπλασιαστή ενώ αντίστοιχα, τη θέση των LSBs του γινόμενου παίρνει το γινόμενο των καινούριων αριθμών. Βέβαια, για να επιτευχθεί η αποκοπή ανά κύτταρο, θα πρέπει τα πρώτα τέσσερα μερικά γινόμενα να είναι τα λιγότερο σημαντικά. Έτσι εξηγείται και η διαφορετική διάταξη των bits της παράλληλης εισόδου του σχήματος 1.45.

Το σήμα  $PPS'_i$  προσφέρει μια επιπλέον δυνατότητα στην εξαγωγή του γινομένου. Έστω στον πολλαπλασιαστή 4x4 ότι τα 4 LSBs είναι δεκαδικά, γι' αυτό και

αποκόπτονται. Θέτοντας το σήμα αυτό στο λογικό 1, προσθέτουμε τον 0,5 στο αποτέλεσμα πριν αποκοπούν τα λιγότερο σημαντικά ψηφία. Με αυτόν τον τρόπο επιτυγχάνεται όχι απλώς αποκοπή των μισών bits αλλά και στρογγυλοποίηση του γινομένου, μειώνοντας έτσι το εισαγόμενο σφάλμα.

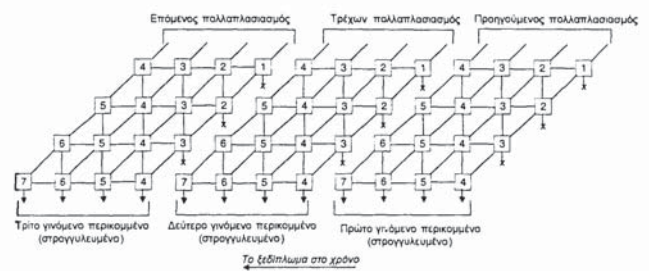
Ο πολλαπλασιαστής του Lyon παρουσιάζει κάποια ομοιότητα όσον αφορά την λειτουργία του με τον παράλληλο πολλαπλασιαστή διάδοσης κρατουμένου του σχήματος 1.32. Η αντιστοίχιση είναι παρόμοια με αυτήν του σειριακού-παράλληλου και του παράλληλου πολλαπλασιαστή με σώσιμο κρατουμένου που περιγράφηκε στην προηγούμενη παράγραφο. Ο παράλληλος πολλαπλασιαστής διάδοσης κρατουμένου είναι το χρονικό "ξεδίπλωμα" του πολλαπλασιαστή του Lyon ενώ αντίθετα, ο πολλαπλασιαστής του Lyon αποτελεί το χρονικό "δίπλωμα" του παράλληλου πολλαπλασιαστή διάδοσης κρατουμένου. Η χρονική αντιστοίχησή τους φαίνεται στο σχήμα 1.46.



Σχήμα 1.46: Αντιστοίχιση παράλληλου πολλαπλασιαστή διάδοσης κρατουμένου και σειριακού πολλαπλασιαστή διοχέτευσης Lyon.

Οι αριθμοί μέσα στα κύτταρα δείχνουν πια χρονική στιγμή είναι ενεργά. Έτσι, την χρονική στιγμή 1 το πρώτο κύτταρο είναι ενεργό του οποίου το αποτέλεσμα όμως

περικόπτεται λόγω του λογικού "0" του σήματος R. Την χρονική στιγμή 2, τα δύο πρώτα κύτταρα είναι ενεργά. Το "0" του σήματος R έχει μεταφερθεί στο δεύτερο κύτταρο μπλοκάροντας έτσι την έξοδό του. Το ίδιο συμβαίνει μέχρι το "0" να περάσει και από τα τέσσερα κύτταρα. Το μονοπάτι που διανύει το "0" του σήματος R φαίνεται στο σχήμα 1.45 από τα γραμμοσκιασμένα κύτταρα. Από την δεύτερη χρονική στιγμή, αρχίζει και ο επόμενος πολλαπλασιασμός αφού τα τέσσερα ψηφία χαμηλότερης αξίας είναι αποκομμένα και δεν πρόκειται να μπερδευτούν με τα μερικά γινόμενα της επόμενης πράξης. Έτσι επιτυγχάνεται η συνεχής διοχέτευση αριθμών στο κύκλωμα και η περικοπή ή στρογγύλευση των γινομένων. Το χρονικό ξεδίπλωμα διαδοχικών πολλαπλασιασμών και η περικοπή των ψηφίων χαμηλότερης αξίας φαίνεται στο σχήμα 1.47.



Σχήμα 1.47 Το χρονικό ξεδίπλωμα διαδοχικών πολλαπλασιασμών

Για την σωστή λειτουργία συνεχούς διοχέτευσης του πολλαπλασιαστή του Lyon, μπορούμε να προσθέσουμε έναν μανδαλωτή στο σήμα του μερικού γινομένου κάθε κυττάρου, διακόπτοντας έτσι το μονοπάτι συνδυαστικής λογικής που ενδεχομένως να περιορίζει τον ρυθμό λειτουργίας. Βέβαια, για τον συγχρονισμό των σημάτων, χρειάζονται και άλλοι δύο μανδαλωτές στα σήματα σειριακής εισόδου D και R. Η μορφή του κυττάρου με τις παραπάνω προσθήκες φαίνεται στο σχήμα 1.48:





Ο πολλαπλασιαστής του σχήματος 1.50 είναι ο πολλαπλασιαστής του Lyon σε ανάποδη διάταξη από αυτή του Σχ 1.45. Για να εξάγουμε όλο το αποτέλεσμα πρέπει να εισάγουμε  $n$ -πλήθος μηδενικών όπως ακριβώς και στην περίπτωση του σειριακού-παράλληλου. Προσθέτοντας όμως το σήμα ελέγχου και εισάγοντας την διαδικασία της στρογγυλοποίησης ( $R$ ) όπως φαίνεται στο σχήμα 1.44 επιτυγχάνουμε την αποδοτικότερη λειτουργία του (απόδοση 100%). Αυτό είναι και το μεγάλο πλεονέκτημα του σειριακού πολλαπλασιαστή συνεχούς διοχέτευσης σε εφαρμογές όπου δεν χρειαζόμαστε μεγαλύτερη ακρίβεια στο αποτέλεσμα του πολλαπλασιασμού από αυτήν των bit του πολλαπλασιαστή ( $X$ ).



## ΠΡΑΞΕΙΣ ΜΕ ΠΡΟΣΗΜΑΣΜΕΝΟΥΣ ΑΡΙΘΜΟΥΣ ΚΥΚΛΩΜΑΤΙΚΕΣ ΕΦΑΡΜΟΓΕΣ

### 2.1 Εισαγωγή

Ο κάθε αριθμός αποτελείται από ένα διάνυσμα ψηφίων που μπορούν να παίρνουν τιμές που ανήκουν στο σύνολο  $\{0, 1, \dots, r-1\}$  όπου  $r$  είναι η βάση του αριθμητικού συστήματος. Στο πολύ γνωστό μας δεκαδικό σύστημα (σύστημα με βάση το 10) ισχύει  $r=10$  επομένως τα ψηφία του διανύσματος ανήκουν στο σύνολο  $\{0, 1, 2, \dots, 9\}$ . Στο κεφάλαιο αυτό θα ασχοληθούμε με διαφορετικά αριθμητικά συστήματα. Θα ξεκινήσουμε από το γνωστό δυαδικό σύστημα θετικών αριθμών, θα αναπτύξουμε την μορφή συμπληρώματος ως προς δύο και την κωδικοποίηση Booth και θα παρουσιάσουμε την αριθμητική προσημασμένων ψηφίων (Signed Digit) με όλες τις τροποποιήσεις που υπάρχουν στην βιβλιογραφία και έχουν παρουσιαστεί κατά καιρούς σε δημοσιεύσεις. Για την καλύτερη κατανόηση των αριθμητικών συστημάτων και των πράξεων σε αυτά, πρόκειται να γίνουν κυκλωματικές εφαρμογές στους γνωστούς μας από το προηγούμενο κεφάλαιο αθροιστές και πολλαπλασιαστές ενώ θα παρουσιαστούν και νέα κυκλώματα που αποτελούν σημεία αναφοράς στις πράξεις με προσημασμένους αριθμούς.