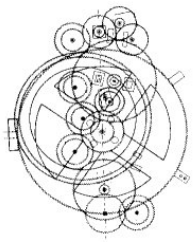
ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ
&
ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ



Εργαστήριο Μικροεπεξεργαστών και Υλικού

Διπλωματική Εργασία

Αρχιτεκτονικές Υλικού για υπολογισμό τιμών χρηματοοικονομικών παραγώγων και υλοποίησή τους σε αναδιατασσόμενη λογική

Κωνσταντίνα Γ. Μιτελούδη

Επιβλέπων: **Ιωάννης Γ. Παπαευσταθίου**,
Αναπληρωτής Καθηγητής

Διπλωματική Εργασία

Αρχιτεκτονικές Υλικού για υπολογισμό τιμών χρηματοοικονομικών παραγώγων και υλοποίησή τους σε αναδιατασσόμενη λογική.

Κωνσταντίνα Γ. Μιτελούδη

Εξεταστική Επιτροπή:

Παπαευσταθίου Ι. , Αναπληρωτής Καθηγητής (επιβλέπων)	Δόλλας Α. , Καθηγητής	Πνευματικάτος Δ. , Καθηγητής

Περίληψη

Η παρούσα διπλωματική εργασία παρουσιάζει αρχιτεκτονικές σε αναδιατασσόμενη λογική, για την αποτίμηση δικαιωμάτων προαίρεσης, που εκμεταλλεύονται τη δυνατότητα των συσκευών FPGAs για μαζική εκτέλεση παράλληλων υπολογισμών και επιτάχυνση της διαδικασίας. Το μοντέλο αποτίμησης που χρησιμοποιήθηκε είναι το Black-Scholes. Έγινε χρήση της μεθοδολογίας των πεπερασμένων διαφορών για την αριθμητική επίλυση της εξίσωσης Black-Scholes και υλοποιήθηκε το σχήμα Crank-Nicolson. Με τη χρήση του αλγόριθμου Odd-Even reduction για την επίλυση του τριδιαγώνιου συστήματος εξίσωσεων, που προέκυψε από το σχήμα Crank-Nicolson, επετεύχθηκε η μεγαλύτερη παραλληλοποίηση της διαδικασίας. Ως αποτέλεσμα, δημιουργήθηκε σύστημα με 64 παράλληλες βασικές υπολογιστικές μονάδες στην FPGA XC5VLX330, συχνότητας 203.957MHz, που επιταχύνει τη διαδικασία 6 φορές σε σχέση με έναν επεξεργαστή core 2 duo 1.60Ghz.

Λέξεις Κλειδιά:Option pricing, FPGA, Black-Scholes, Finite Difference, Crank-Nicolson

στους γονείς μου, που
μου χάρισαν το ζην και
μου δίδαξαν το ευ ζην

Ευχαριστίες

Ευχαριστώ τον επιβλέποντα καθηγητή κ. Ιωάννη Παπαευσταθίου για την εμπιστοσύνη που έδειξε στις ικανότητές μου, για την ευκαιρία που μου έδωσε να ασχοληθώ με ένα τόσο ενδιαφέρον θέμα και για τη μεγάλη υπομονή του.

Επίσης, θα ήθελα να ευχαριστήσω τους Καθηγητές κ. Απόστολο Δόλλα και κ. Διονύσιο Πνευματικάτο για την συμμετοχή τους ως μέλη της εξεταστικής επιτροπής.

Ακόμη θα ήθελα να ευχαριστήσω τον ερευνητή του Ινστιτούτου Τηλεπικοινωνιακών Συστημάτων και Μηχανικό κ. Ανδρέα Μπροκαλάκη για την πολύτιμη καθοδήγησή του και τις συμβουλές που μου έδωσε.

Οπωσδήποτε πρέπει να ευχαριστήσω δυο πραγματικούς φίλους και συναδέλφους, πλέον, το Μανώλη Ματιγάκη και το Γιάννη Κιμιωνή για τη στήριξη που μου προσέφεραν στο δύσκολο ξεκίνημα αυτής της εργασίας.

Επίσης, ευχαριστώ το σύντροφό μου Αλκαίο Σακελλάρη για όλα. Μαζί κάνουμε ένα δυνατό team.

Τέλος, το μεγαλύτερο ευχαριστώ το χρωστάω στους γονείς μου, που μου έμαθαν να ονειρεύομαι και με στηρίζουν με υπομονή και επιμονή στην πραγματοποίηση των ονείρων μου.

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

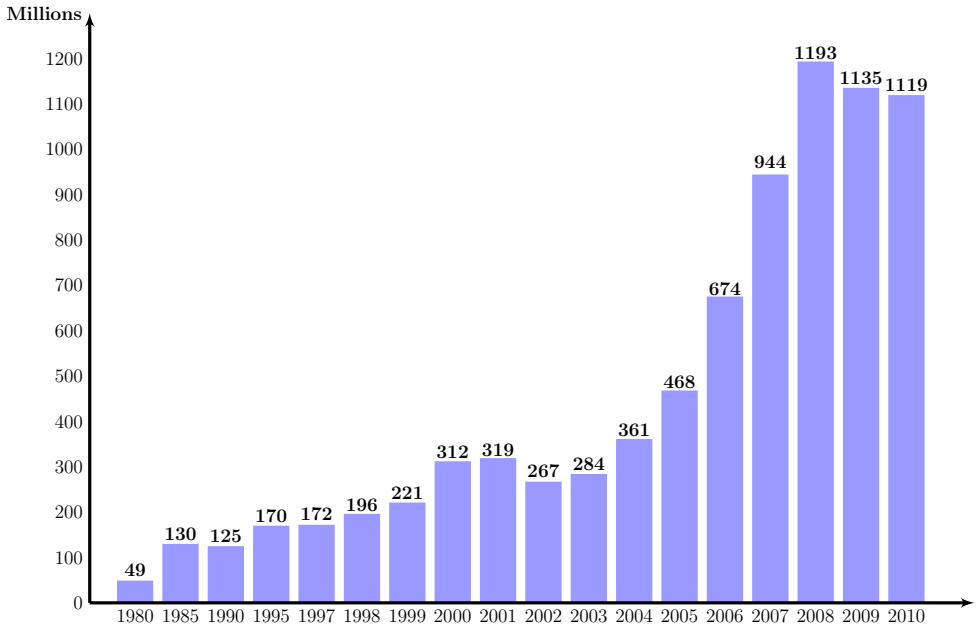
1.1 Γενικά

Τα παράγωγα χρηματοοικονομικά προϊόντα δεν αποτελούν καινοτομία των σύγχρονων αγορών κεφαλαίου και χρήματος. Η χρήση τους μετατίθεται πριν από πολλούς αιώνες, όταν οι Αρχαίοι Φοίνικες αλλά και οι Αρχαίοι Έλληνες πωλούσαν ολόκληρα φορτία πλοίων προθεσμιακά, δηλαδή με προκαθορισμένη τιμή και παράδοση στο μέλλον. Η αναγέννηση των γραμμάτων και των τεχνών στην Ευρώπη, έφερε μεταξύ των άλλων και νεωτερισμούς στις αγορές κυρίως των Κάτω Χωρών (του Βελγίου και της Ολλανδίας) που αποτελούσαν το κέντρο του Ευρωπαϊκού εμπορίου.Στο Άμστερνταμ της Ολλανδίας, οι συναλλαγές σε προθεσμιακά συμβόλαια (Futures) χρονολογούνται από την εποχή της τουλιπομανίας, τη δεκαετία του 1630. Κατά τις δεκαετίες του 1970 και του 1980, η απελευθέρωση των αγορών συναλλάγματος, αλλά και η συμβολή των ακαδημαϊκών στην τιμολόγηση των παραγώγων χρηματοοικονομικών προϊόντων και κυρίως των δικαιωμάτων προαίρεσης (Options), κατόρθωσαν να αλλάξουν ριζικά το τοπίο και να διευρύνουν σημαντικά τη χρήση τους.

Τα χρηματοοικονομικά παράγωγα αποτελούν σήμερα ένα ισχυρό εργαλείο στα χέρια χρηματοπιστωτικών ιδρυμάτων και επενδυτών και συγκεντρώνουν όλο και μεγαλύτερο ενδιαφέρον. Στο σχήμα (1.1) φαίνεται ο ετήσιος όγκος συναλλαγών των συμβολαίων δικαιωμάτων προαίρεσης σε εκατομμυρία, στην αγορά παραγώγων του Chicago Board Options Exchange . Παρατηρείται η αλματώδης αύξηση του όγκου συναλλαγών την τελευταία δεκαετία και παρά την χρηματοικονομική κρίση του 2008, φαίνεται πως διατηρείται αυτός ο όγκος συναλλαγών, με μια μικρή μείωση.

Όσο αυξάνεται στις χρηματοοικονομικές αγορές ο όγκος των συναλλάγων παράγωγων

προϊόντων, τόσο αυξάνεται και το πλήθος των διαφορετικών χρεογράφων που διαπραγματεύονται σε αυτές. Καινούρια υπολογιστικά μοντέλα έχουν αναπτυχθεί για την αποτίμησή τους και τη μελέτη της συμπεριφοράς τους. Αποτελούν, μεταξύ άλλων, αντικείμενο μελέτης ενός σχετικά νέου επιστημονικού κλάδου, του Financial Engineering. Το πλήθος των μαθηματικών μοντέλων, καθώς και η πολυπλοκότητά τους, σε συνδυασμό με την ημερήσια διαπραγμάτευση εκατομμυρίων συμβολαίων στις χρηματοοικονομικές αγορές, θέτουν καινούργιες προκλήσεις στον κλάδο του Financial Computing. Προκλήσεις για υπολογιστικά συστήματα με καλύτερη απόδοση όσον αφορά τη ταχύτητα, την ακρίβεια και την κατανάλωση ενέργειας.



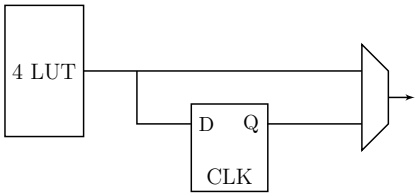
Σχήμα 1.1: Ετήσιος όγκος συναλλαγών συμβολαίων δικαιωμάτων προαίρεσης σε εκατομμυρία στην αγορά του CBEO. (Πηγή: Chicago Board Options Exchange Annual Report)

Τα χρηματοοικονομικά ιδρύματα για να ανταπεξέλθουν στα καινούργια δεδομένα, που διαμορφώνονται στις αγορές, στρέφονται σε λύσεις που προσφέρει το high performance computing (HPC), όπως τα clusters και τα grids και πιο πρόσφατα σε GPUs και FPGAs. Παράδειγμα τέτοιων ιδρυμάτων είναι η J.P Morgan και η Deutsche Bank που υλοποιούν τα μοντέλα τους για την εκτίμηση χαρτοφυλακίου σε συσκευές αναδιατασσόμενης λογικής, FPGAs, οδηγούμενες από την ανάγκη για υπεροχή στο πεδίο της ταχύτητας. Επίσης, σημαντικό κριτήριο στην επιλογή τους αυτή είναι και η μείωση της κατανάλωσης ενέργειας, καθώς σε μεγάλα υπολογιστικά συστήματα το ενεργειακό κόστος είναι τεράστιο.

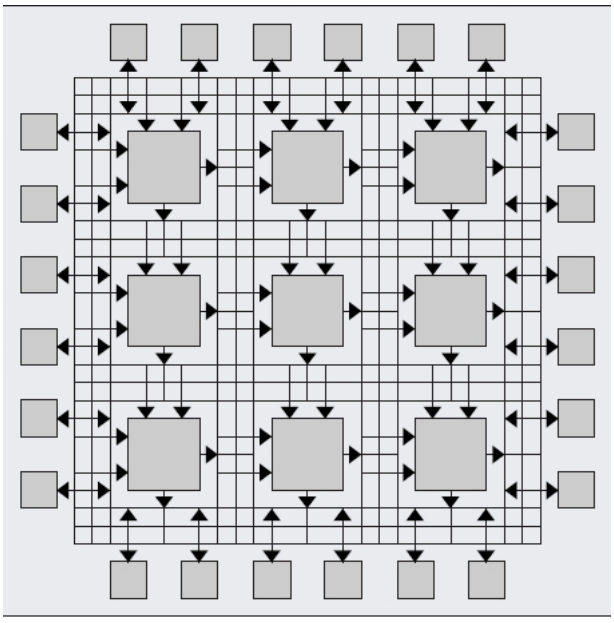
Αναδιατασσόμενη Λογική

Από τα τέλη της δεκαετίας του ′80 εμφανίστηκε η αναδιατασσόμενη λογική με το όνομα *Επί του Πεδίου Προγραμματιζόμενη Συστοιχία Πυλών* (**Field Programmable Gate Array - FPGA**). Αναδιατασσόμενη λογική είναι μια μορφή ψηφιακών κυκλωμάτων με επαναλαμβανόμενους πόρους για υλοποίηση συνδυαστικών και ακολουθιακών λογικών κυκλωμάτων, καθώς και προγραμματιζόμενο τρόπο διασύνδεσης αυτών.

Οι FPGAs συνδυάζουν πλεονεκτήματα τόσο του software όσο και του hardware. Τα κυκλώματα τους είναι υλοποιημένα όπως στο hardware, παρέχοντας τεράστιο πλεονέκτημα σε ενέργεια, χώρο και απόδοση, αλλά μπορούν να επαναπρογραμματιστούν φθηνά και γρήγορα για την υλοποίηση διάφορων εργασιών, όπως γίνεται στο software [1]. Οι FPGAs έχουν αποδείξει την ικανότητά τους στην επιτάχυνση της επεξεργασίας σε διάφορους επιστημονικούς τομείς, όπως στη βιοπληροφορική και την επεξεργασία εικόνας.



Σχήμα 1.2: A simple lookup table logic block.



Σχήμα 1.3: An abstract view of an FPGA; logic cels are embedded in a general routing.

1.2 Βιβλιογραφική έρευνα

Διάφορες προσεγγίσεις έχουν προταθεί στο πρόβλημα αποτίμησης δικαιωμάτων προαίρεσης και γενικότερα στην μελέτη χρηματοοικονομικών παραγώγων, τόσο σε αλγοριθμικό επίπεδο όσο και σε επίπεδο εφαρμογών. Οι πιο συχνά εφαρμοζόμενες μέθοδοι είναι η προσομοίωση Monte-Carlo και το διωνυμικό μοντέλο αποτίμησης. Οι πιο διαδεδομένες πλατφόρμες στο high performance computing (HPC), όπου έχουν υλοποιηθεί οι παραπάνω μέθοδοι, είναι οι FPGAs και οι GPUs.

Οι Zhang et al. [2], εφαρμόζουν τη προσομοίωση Monte-Carlo σε μία γενική αρχιτεκτονική βασισμένη σε FPGA, που συνδυάζει ταχύτητα και ευελιξία με την ενσωμάτωση ενός pipelined Monte-Carlo πυρήνα και ενός on-chip επεξεργαστή εντολών. Πετυχαίνει 25× επιτάχυνση της διαδικασίας σε σχέση με το software.

Άλλη μια έρευνα που υλοποιεί τη προσομοίωση Monte-Carlo είναι των D. Thomas, J. Bower, W. Luk [3]. Η έρευνα αυτή προσπαθεί να εκμεταλευθεί τη φύση του μοντέλου και τη δυνατότητα της FPGA για εκτενή παραλληλοποίηση. Προσπαθεί να δώσει ένα πλαίσιο για τη αυτόματη δημιουργία των τυχαίων μονοπατιών της προσομοίωσης Monte-Carlo, μέσω πέντε διαφορετικών μοντέλων όπως τα log-normal, Value-at-Risk, GARCH. Η αρχιτεκτονική που προτείνει είναι fully pipelined ώστε να μεγιστοποιηθεί το throughput και πετυχαίνει 80× επιτάχυνση της διαδικασίας.

Οι R. Baxter et al. [4] παρουσιάζουν ένα υπερυπολογιστή γενικής χρήσης δομημένο απο 64 FPGAs. Μία από τις υλοποιήσεις σε αυτόν τον υπερυπολογιστή είναι και η εφαρμογή της προσομοίωσης Monte-Carlo. Ο πυρήνας του Monte-Carlo μπορεί να επαναληφθεί 10 φορές σε μία συσκευή FPGA και να επεξεργαστεί από 16 FPGAs . Τα αποτελέσματα δείχνουν ότι οι FPGAs μπορούν να ξεπεράσουν πάνω από δύο τάξεις μεγέθους τη CPU.

Στον ίδιο υπερυπολογιστή Maxwell οι X. Tian and K. Benkrid [5] υλοποιούν την προσομοίωση Monte-Carlo χρησιμοποιώντας το μοντέλο Box-Muller για τη δημιουργία των τυχαίων μονοπατιών, πετυχαίνοντας επιτάχυνση 750× και για Asian Option 600× - η προηγούμενη εργασία των R. Baxter et al. είχε επιτάχυνση 323×.

Σε μία άλλη εργασία των X. Tian and K. Benkrid [6][7] υλοποιείται σε FPGA η μέθοδος Least-Squares Monte Carlo (LSMC), για την αποτίμηση δικαιωμάτων προαίρεσης Αμερικάνικου τύπου. Για την δημιουργία των μονοπατιών χρησιμοποιήθηκε η μέθοδος Quasi-Monte Carlo. Η συνολική επιτάχυνση της διαδικασίας ήταν 20× σε σχέση με τη CPU.

Οι A. Kaganov, P. Chow, and A. Lakhany [8],[9] στις εργασίες τους παρουσιάζουν μία αρχιτεκτονική υλικού για την αποτίμηση Collateralized Debt Obligations (CDOs) βασισμένη στην προσομοίωση Monte-Carlo. Εφαρμόζουν δύο μοντέλα στις εργασίες τους , το One-Factor Gaussian Copula (OFGC) και το Multi-Factor Gaussian Copula (MFGC)

και η επιτάχυνση που παρουσιάζουν είναι 63× και 71× αντίστοιχα.

Στην εργασία των Q. Jin et al. [10] εφαρμόζεται το διωνυμικό μοντέλο αποτίμησης δικαιωμάτων προαίρεσης. Υλοποιείται σε FPGA και GPU με fixed-point αριθμητική ακρίβεια. Τα αποτελέσματα τους δείχνουν 250 φορές πιο γρήγορη την υλοποίηση σε FPGA σε σχέση με την CPU και περισσότερο από 2 φορές πιο γρήγορη από την GPU.

Οι Q. Jin et al. [11] στην εργασία τους υλοποιούν σε FPGA και GPU την μέθοδο πεπερασμένων διαφορών για την εξίσωση Black-Scholes. Χρησιμοποιώντας το ρητό σχήμα (explicit scheme) παρουσιάζουν επιτάχυνση 12× στην FPGA η οποία συγκρινόμενη με την GPU είναι πιο αργή 3.6×, αλλά 9 φόρες πιο αποδοτική ενεργειακά.

Στην εργασία των A. Tse et al. [12] εφαρμόζεται με βάση τη μέθοδο Quadrature αριθμητική ολοκλήρωση. Αυτή η μέθοδος μπορεί να χρησιμοποιηθεί για παράγωγα με περισσότερα του ενός υποκείμενου προϊόντος. Παρουσιάζεται ένα αυτοματοποιημένο σύστημα για τη δημιουργία αρχιτεκτονικών υλικού για την εξέταση της αύξησης των διαστάσεων του συστήματος. Τα αποτελέσματα δείχνουν επιτάχυνση 23× στην FPGA σε σχέση με τη CPU. Η απόδοση σε ταχύτητα της GPU είναι περίπου ίδια με την FPGA.

Στις εργασίες όπου εφαρμόζεται η προσομοίωση Monte-Carlo, μπορούν να προστεθούν η εργασία των A. Tse et al. [13] για αποτίμηση δικαιωμάτων προαίρεσης Ασιατικού τύπου και η εργασία [14] για αποτίμηση Exotic Option. Ακόμα οι εργασίες των Schryver et al. [15],των X. Tian and C. Bouganis [16] και Chow et al. [17].

Τέλος, στις εργασίες των Q. Jin, W. Luk, and D. Thomas [18][19] γίνεται προσπάθεια για τη δημιουργία ενός ενιαίου πλαισίου όσον αφορά τα μέτρα απόδοσης που χρησιμοποιούνται για την αξιολόγηση των υλοποιήσεων των πεπερασμένων διαφορών σε FPGAs.

1.3 Σκοπός της εργασίας

Η πληθώρα των εφαρμογών που βασίζονται στο Monte-Carlo και το διωνυμικό μοντέλο αποτίμησης καθώς και τα αποτελέσματα τους, δείχνουν ότι σε αυτό το πεδίο έχει γίνει εξαντλητική έρευνα. Από την άλλη πλευρά, η μέθοδος πεπερασμένων διαφορών έχει υλοποιηθεί μόνο με χρήση του ρητού σχήματος.

Σκοπός της παρούσας διπλωματικής εργασίας είναι η δημιουργία ενός συστήματος αποτίμησης δικαιωμάτων προαίρεσης, σε αναδιατασσόμενη λογική. Η υλοποίηση του σχήματος πεπερασμένων διαφορών Crank-Nicolson δίνει πιο γρήγορη σύγκλιση και μεγαλύτερη ακρίβεια στην αριθμητική επίλυση της εξίσωσης Black-Scholes, από ότι το ρητό ή το πεπλεγμένο σχήμα . Η χρήση του αλγόριθμου Cyclic Odd-Even Reduction έχει ως σκοπό την εκμετάλευση της δυνατότητας παραλληλοποίησης που παρέχει η FPGA.

1.4 Δομή της εργασίας

Στο κεφάλαιο 2 της εργασίας, γίνεται μια εισαγωγή στα χρηματοοικονομικά παράγωγα και ειδικότερα στα Δικαιώματα Προαίρεσης (Options). Αυτό κρίνεται απαραίτητο ώστε να μπορεί ο αναγνώστης να κατανοήσει τη χρήση των χρηματοοικονομικών παραγώγων. Στη συνέχεια γίνεται περιγραφή των μοντέλων αποτίμησης Δικαιωμάτων Προαίρεσης, όπου έχουμε το διακριτό μοντέλο της Διωνυμικής αποτίμησης και τα στοχαστικά μοντέλα Black-Scholes και Monte-Carlo.

Στο κεφάλαιο 3, παρουσιάζεται η μέθοδος των πεπερασμένων διαφορών. Γίνεται λύση της εξίσωσης διάχυσης η Θερμότητας ως παράδειγμα για τη χρήση των πεπερασμένων διαφορών. Στη συνέχεια περιγράφεται αναλυτικά η επίλυση της μερικής διαφορικής εξίσωσης Black-Scholes με τη μέθοδο των πεπερασμένων διαφορών, μέσω τριών σχημάτων Explicit, Implicit και Crank-Nicolson.

Στο κεφάλαιο 4, παρουσιάζονται και συγκρίνονται μέθοδοι για την επίλυση τριδιαγώνιων γραμμικών συστημάτων. Η LU δίασπαση και ο αλγόριθμος Cyclic Odd-Even Reduction εξετάζονται, καθώς και μία παραλλαγή του τελευταίου, που είναι αυτή που υλοποιήθηκε σε αναδιατασσόμενη λογική.

Στο κεφάλαιο 5, παρουσιάζεται η μοντελοποίηση των φάσεων του παραλλαγμένου αλγόριθμου Cyclic Odd-Even Reduction για την επίλυση του σχήματος πεπερασμένων διαφορών Crank-Nicolson που προσεγγίζει αριθμητικά τη μερική διαφορική εξίσωση Black-Scholes. Καθώς και οι αρχιτεκτονικές αυτών σε αναδιατασσόμενη λογική.

Στο κεφάλαιο 6, εξετάζονται η παράμετροι της υλοποίησης ως προς την απόδοση της αρχιτεκτονικής και τα αποτελέσματα της διπλωματικής εργασίας.

Τέλος, στο κεφάλαιο 7, αναφέρονται τα συμπεράσματα από τη συγκεκριμένη διατριβή και προτείνονται ιδέες για μελλοντική επέκταση ή και βελτιώσεις της υπάρχουσας αρχιτεκτονικής.

Κεφάλαιο 2

Χρηματοοικονομικά Παράγωγα

2.1 Εισαγωγή

Τα παράγωγα είναι ειδικής μορφής χρεόγραφα τα οποία βασίζονται στην ύπαρξη άλλων χρεογράφων και χρηματοοικονομικών προϊόντων (υποκείμενο προϊόν). Ειδικότερα, τα παράγωγα είναι συμβόλαια μεταξύ δύο αντισυμβαλλόμενων μελών για την πραγματοποίηση μιας συναλλαγής (αγοράς ή πώληση) επί ενός συγκεκριμένου υποκείμενου προϊόντος. Η συναλλαγή θα πραγματοποιηθεί κάποια στιγμή στο μέλλον με προσυμφωνημένους όρους όσον αφορά την τιμή και τη χρονική στιγμή στην οποία θα διεκπεραιωθεί.

Ο βασικός λόγος της εισαγωγής και χρήσης των παραγώγων είναι η ανάγκη ανάπτυξης μηχανισμών διασφάλισης έναντι του κινδύνου που ενέχει η επένδυση σε ένα χρεόγραφο. **Τα παράγωγα δηλαδή αποτελούν εργαλεία μείωσης του κινδύνου**.

Οι πλέον διαδεδομένες μορφές παραγώγων είναι τα δικαιώματα προαίρεσης (options) και τα συμβόλαια μελλοντικής εκπλήρωσης (futures). Στις ενότητες που ακολουθούν αναλύονται τα βασικά θέματα που αφορούν τη λειτουργία και την αποτίμηση αυτών των παράγωγων προϊόντων.

2.2 Δικαιώματα προαίρεσης

Ένα δικαίωμα προαίρεσης είναι ένα συμβόλαιο το οποίο δίνει στον κάτοχό του το δικαίωμα να αγοράσει ή να πουλήσει ένα συγκεκριμένο χρεόγραφο (υποκείμενο προϊόν) σε προκαθορισμένη τιμή μέσα σε ένα δεδομένο χρονικό διάστημα. Η αξία του δικαιώματος προσδιορίζεται συναρτήσει της τιμής του υποκείμενου προιόντος πάνω στο οποίο βασίζεται. Υπάρχουν δύο είδη δικαιωμάτων προαίρεσης, τα δικαιώματα αγοράς (call options ή απλά calls) και τα δικαιώματα πώλησης (put options ή απλά puts).

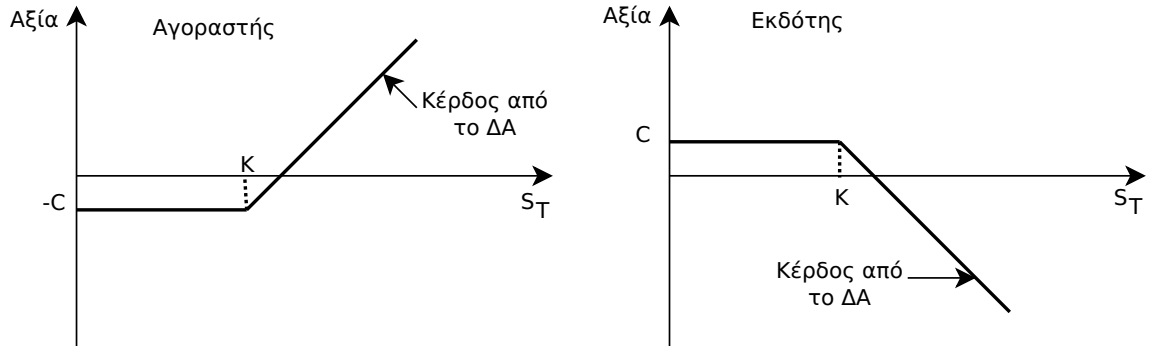
2.2.1 Δικαιώματα αγοράς

Όπως υποδηλώνει και το όνομά του, ένα δικαίωμα αγοράς (ΔΑ) δίνει στον κάτοχό του το δικαίωμα να αγοράσει μία συγκεκριμένη ποσότητα του υποκείμενου προϊόντος σε συγκεκριμένη τιμή στο μέλλον. Υπάρχουν δύο είδη ΔΑ, τα Ευρωπαϊκά (European calls) και τα Αμερικάνικα (American calls).Στα Ευρωπαϊκά ΔΑ η αγορά μπορεί να πραγματοποιηθεί μόνο στο τέλος της προκαθορισμένης περιόδου, ενώ αντίθετα στα Αμερικανικά ΔΑ η αγορά μπορεί να πραγματοποιηθεί ανά πάσα στιγμή μέσα στο προκαθορισμένο χρονικό διάστημα.

Ουσιαστικά ένα ΔΑ είναι ένα «στοίχημα» μεταξύ δύο επενδυτών όσον αφορά την τιμή του υποκείμενου προϊόντος. Ο αγοραστής του ΔΑ πιστεύει ότι θα υπάρξει άνοδος της τιμής του στο μέλλον, ενώ αντίθετα ο πωλητής του ΔΑ πιστεύει ότι η τιμή του υποκείμενου προϊόντος θα μειωθεί.

Θεωρώντας την περίπτωση ενός Ευρωπαϊκού ΔΑ, ο αγοραστής του έχει το δικαίωμα να αγοράσει το υποκείμενο προϊον σε μια δεδομένη τιμή. Η τιμή αυτή ονομάζεται **τιμή εξάσκησης** (strike price ή exercise price) και στο εξής θα συμβολίζεται ως **K**. Στον αντίποδα, ο εκδότης του ΔΑ είναι υποχρεωμένος να πουλήσει το υποκείμενο προϊόν στον αγοραστή εάν ο δεύτερος αποφασίσει να εξασκήσει το δικαίωμά του. Το κέρδος του καθενός από τη συναλλαγή αποδίδεται γραφικά στο Σχήμα 2.1.

Κατά την αγορά του ΔΑ ο αγοραστής πληρώνει στον εκδότη την αξία του δικαιώματος η οποία στο εξής θα συμβολίζεται ως C. Με τη λήξη της προκαθορισμένης περιόδου (λήξη του ΔΑ) ο αγοραστής μπορεί να εξασκήσει το δικαίωμά του για την αγορά του υποκείμενου προϊόντος ή να μην το εξασκήσει. Η απόφαση που θα λάβει εξαρτάται από τη σχέση μεταξύ της τιμής S_T του υποκείμενου προϊόντος κατά τη στιγμή στην οποία λήγει το ΔΑ και της τιμής εξάσκησης K του ΔΑ.



Σχήμα 2.1: Κέρδη και ζημιές για τον αγοραστή και τον εκδότη ενος ΔΑ

Συγκεκριμένα, εάν η τιμή του υποκείμενου προϊόντος είναι υψηλότερη από την τιμή εξάσκησης ($S_T > K$), τότε ο αγοραστής πρέπει να εξασκήσει το δικαίωμα και να αγοράσει

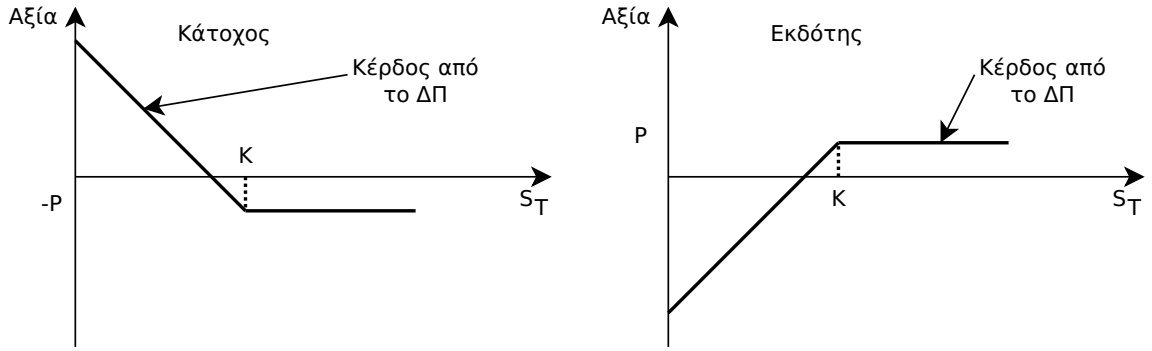
το υποκείμενο προϊόν στην τιμή Κ. Αφού $S_T > K$, προφανώς ο αγοραστής έχει μία εισροή κεφαλαίων ίση με $S_T - K$. Το καθαρό κέρδος του είναι $S_T - K - C$. Στην ίδια περίπτωση ο πωλητής του ΔΑ έχει καθαρή ζημία ίση με $S_T - K - C$.

Έστω τώρα η περίπτωση όπου η τιμή του υποκείμενου προϊόντος κατά τη χρονική στιγμή στην οποία λήγει το ΔΑ είναι μικρότερη από την τιμή εξάσκησης του δικαιώματος ($S_T < K$). Στην περίπτωση αυτή ο αγοραστής δεν θα εξασκήσει το δικαίωμα και η ζημία του θα περιοριστεί στο ποσό που κατέβαλλε για την αγορά του ΔΑ, δηλαδή το ποσό C. Αν εξασκηθεί το δικαίωμα, η ζημία αυτή θα αυξηθεί κατά τη διαφορά $K - S_T$. Στον αντίποδα, ο πωλητής του ΔΑ βγαίνει κερδισμένος στην περίπτωση αυτή και το κέρδος του είναι ίσο με το ποσό που εισέπραξε από τον αγοραστή, δηλαδή ίσο με C.

2.2.2 Δικαιώματα πώλησης

Σε αντίθεση με τα ΔΑ, τα δικαιώματα πώλησης (ΔΠ, puts) δίνουν στον κάτοχό τους το δικαίωμα να πουλήσει το υποκείμενο προϊόν μέσα σε ένα προκαθορισμένο χρονικό διάστημα και σε προσυμφωνημένη τιμή. Ο πωλητής του ΔΠ, στην περίπτωση αυτή, είναι υποχρεωμένος να αγοράσει το υποκείμενο προϊόν από τον αγοραστή του ΔΠ, εφόσον αυτός αποφασίσει να εξασκήσει το δικαίωμά του.

Όπως και στην περίπτωση των ΔΑ, υπάρχουν ΔΠ Ευρωπαϊκού τύπου στα οποία η εξάσκηση του δικαιώματος μπορεί να γίνει μόνο κατά τη λήξη του και ΔΠ Αμερικανικού τύπου, όπου η εξάσκηση του δικαιώματος μπορεί να γίνει ανά πάσα στιγμή μέχρι τη λήξη του.



Σχήμα 2.2: Κέρδη και ζημιές για τον αγοραστή και τον εκδότη ενός ΔΠ

Στο Σχήμα 2.2 παρουσιάζονται τα κέρδη που αποφέρει ένα ΔΠ στον κάτοχό του και στον εκδότη.

Όπως και στην περίπτωση των ΔΑ, τα κέρδη προσδιορίζονται από τη σχέση ανάμεσα στην τιμή S_T του υποκείμενου προϊόντος κατά τη στιγμή εξάσκησης και στην τιμή εξάσκησης K του δικαιώματος. Αναλυτικότερα διακρίνονται οι εξής δύο περιπτώσεις:

- Περίπτωση $S_T > K$: Στην περίπτωση αυτή ο αγοραστής του ΔΠ δεν θα εξασκήσει το δικαίωμα, καθώς μπορεί να πουλήσει το υποκείμενο προϊόν σε τιμή S_T που είναι υψηλότερη από την προσυμφωνημένη τιμή K του ΔΠ. Η ζημία του, στην περίπτωση αυτή, θα είναι ίση με το ποσό που κατέβαλε για την αγορά του δικαιώματος, δηλαδή P. Από την άλλη πλευρά ο εκδότης του ΔΠ θα έχει ένα ισόποσο κέρδος ίσο με C.

- Περίπτωση $S_T < K$: Στην περίπτωση αυτή ο αγοραστής του ΔΠ θα εξασκήσει το δικαίωμα, καθώς μπορεί να πουλήσει το υποκείμενο προϊόν στον πωλητή του ΔΠ, σε τιμή υψηλότερη από την πραγματική του αξία. Το καθαρό κέρδος του αγοραστή θα είναι $S_T - K$ μειωμένο κατά την αξία αγοράς του ΔΠ (P). Αντίθετα ο πωλητής έχει ζημία από τη συναλλαγή, η οποία είναι ίση με το κέρδος του κατόχου του ΔΠ.

2.3 Αποτίμηση δικαιωμάτων προαίρεσης

Η αποτίμηση των δικαιωμάτων προαίρεσης κατά τα τελευταία 30 χρόνια είναι ένα από τα πλέον σημαντικά πεδία έρευνας της χρηματοοικονομικής επιστήμης. Το θέμα αυτό δεν παρουσιάζει μόνο σημαντικό ερευνητικό ενδιαφέρον, αλλά παράλληλα έχει και αυξημένο πρακτικό ενδιαφέρον, καθώς τεχνικές αποτίμησης των δικαιωμάτων προαίρεσης, αλλά και άλλων μορφών παραγώγων χρησιμοποιούνται καθημερινά στην πράξη από διαχειριστές χαρτοφυλακίων. Η διάδοση τους συμπίπτει με τη δημοσίευση της εργασίας των Fischer Black και Myron Scholes [20] σχετικά με την αποτίμηση των δικαιωμάτων προαίρεσης.

Στις ενότητες που ακολουθούν παρουσιάζονται δύο βασικές μεθοδολογίες αποτίμησης δικαιωμάτων προαίρεσης, το διωνυμικό μοντέλο και το μοντέλο των Black & Scholes. Η ανάλυση των δύο αυτών μοντέλων πραγματοποιείται για την περίπτωση όπου το εξεταζόμενο δικαίωμα προαίρεσης είναι ένα ΔΑ Ευρωπαϊκού τύπου. Όπως θα παρουσιαστεί, συνήθως τα ΔΑ Αμερικανικού τύπου δεν εξασκούνται πριν τη λήξη τους. Συνεπώς, παρόμοια ανάλυση με αυτή που θα παρουσιαστεί μπορεί να χρησιμοποιηθεί και στην περίπτωση ΔΑ Αμερικάνικου τύπου. Δεδομένης μιας σχέσης που συσχετίζει τη τιμή ενός ΔΑ Ευρωπαϊκού τύπου και την τιμή ενός ΔΠ Ευρωπαϊκού τύπου, τα παραπάνω δύο μοντέλα μπορούν να χρησιμοποιηθούν και για την αποτίμηση ΔΠ. Πριν λοιπόν, αναλυθούν τα δύο προαναφερθέντα μοντέλα αποτίμησης δικαιωμάτων προαίρεσης θα αναπτυχθούν ορισμένες βασικές αρχές αποτίμησης των δικαιωμάτων.

2.3.1 Ελάχιστη τιμή δικαιωμάτων αγοράς

Έστω ότι σε κάποια χρονική στιγμή t ένας επενδυτής έχει διαθέσιμες τις ακόλουθες δύο επιλογές:

- **Επιλογή Α:** Αγορά ενός ΔΑ στην τιμή C και ταυτόχρονα την επένδυση ποσού $K/(1+i)$ με επιτόκιο i για το διάστημα μέχρι τη λήξη του ΔΑ.

- **Επιλογή Β:** Άμεση αγορά του υποκείμενου προϊόντος του ΔΑ της επιλογής Α, στην τρέχουσα αξία του S.

Στον Πίνακα [2.1] που ακολουθεί παρουσιάζονται οι εισροές και εκροές κεφαλαίων για τις δύο επιλογές, τόσο στην στιγμή t όσο και στη λήξη. Όπως είναι εμφανές, σε κάθε περίπτωση τα έσοδα από το χαρτοφυλάκιο Α στη λήξη των δικαιωμάτων υπερβαίνουν τα έσοδα της επιλογής Β. Επομένως, αντίστοιχη θα πρέπει να είναι και η σχέση μεταξύ της αξίας των δύο επιλογών σήμερα. Δηλαδή:

$$C + \frac{K}{1+i} > S \Rightarrow C > S - \frac{K}{1+i}$$

Η σχέση αυτή προσδιορίζει την ελάχιστη τιμή του δικαιώματος αγοράς.

		Ληξη	
	t	$S_T < K$	$S_T > K$
Επιλογή Α			
Αγορά ΔΑ	$-C$	0	$S_T - K$
Επένδυση	$-K/(1+i)$	K	K
Σύνολο Α	$-C - K/(1+i)$	K	S_T
Επιλογή Β	$-S$	S_T	S_T

Πίνακας 2.1: Εισροές και εκροές κεφαλαίων για τη στιγμή t και τη λήξη.

2.3.2 Εξάσκηση δικαιωμάτων Αμερικάνικου τύπου

Όπως έχει ήδη αναφερθεί, κύριο χαρακτηριστικό των δικαιωμάτων Αμερικάνικου τύπου είναι ότι μπορούν να εξασκηθούν οποιαδήποτε στιγμή μέχρι τη λήξη τους. Έστω λοιπόν ότι κάποια χρονική στιγμή t, πριν τη λήξη του δικαιώματος η αξία του υποκείμενου προϊόντος είναι S και ότι μέχρι τη λήξη το υποκείμενο προϊόν δεν θα αποφέρει έσοδα με τη μορφή τοκομεριδίων (για ομόλογα) ή μερισμάτων (για μετοχές/δείκτες). Τότε με την εξάσκηση του δικαιώματος ο επενδυτής θα έχει έσοδα ύψους $max\{0, S-K\}$. Εάν αντίθετα προχωρήσει στην πώληση του δικαιώματος, τότε θα έχει έσοδα:

$$C > S - \frac{K}{1+i}$$

Η εξάσκηση του δικαιώματος έχει νόημα μόνο εάν $S > K$. Στην περίπτωση όμως αυτή $C > S - K$, δηλαδή η πώληση του δικαιώματος θα αποφέρει μεγαλύτερα έσοδα σε σχέση με την εξάσκησή του. Άρα η εξάσκηση του δικαιώματος πριν τη λήξη του δεν θα πρέπει να προτιμηθεί από τον επενδυτή. Εναλλακτικά, μπορεί να προχωρήσει στην πώληση του δικαιώματος.

2.3.3 Σχέση μεταξύ της τιμής δικαιωμάτων αγοράς και πώλησης

Η αξία ενός ΔΑ συνδέεται με την αξία ενός αντίστοιχου ΔΠ μέσω μίας σχέση η οποία είναι γνωστή ως call-put parity. Έστω ότι σε κάποια χρονική στιγμή t εξετάζονται δύο επενδυτικές επιλογές. Η επιλογή Α αφορά την αγορά του υποκείμενου προϊόντος και ενός ΔΠ αξίας P και παράλληλα το δανεισμό κεφαλαίου $K/(1+i)$ με επιτόκιο i μέχρι τη λήξη του ΔΠ, όπου Κ είναι η τιμή εξάσκησης του δικαιώματος. Η επιλογή Β αφορά την αγορά ενός ΔΑ αξίας C. Τα αποτελέσματα (εισροές/εκροές) αυτών των δύο εναλλακτικών κατά τη λήξη συνοψίζονται στον πίνακα που ακολουθεί.

		Λήξη	
	t	$S_T < K$	$S_T > K$
Επιλογή Α			
Αγορά μετοχής	$-S$	S_T	S_T
Αγορά ΔΠ	$-P$	0	$K - S_T$
Δανεισμός	$K/(1+i)$	$-K$	$-K$
Σύνολο Α	$-S - P + K/(1+i)$	$S_T - K$	0
Επιλογή Β	$-C$	$S_T - K$	0

Πίνακας 2.2: Εισροές/Εκροές κατά τη λήξη

Όπως φαίνεται τα αποτελέσματα που αποφέρουν οι δύο εναλλακτικές κατά τη λήξη της χρονικής περιόδου είναι απολύτως ταυτόσημα και συνεπώς έχουν την ίδια ακριβώς αξία. Δηλαδή :

$$C = S + P - \frac{K}{1+i} \tag{2.1}$$

Η σχέση αυτή είναι ιδιαίτερα χρήσιμη καθώς επιτρέπει τον προσδιορισμό της τιμής P του ΔΠ δεδομένης της τιμής του ΔΑ και αντίστροφα. Η σχέση αυτή είναι γνωστή ως call-put parity. Έχοντας υπόψη τη σχέση αυτή, τα αποτελέσματα των μοντέλων που θα παρουσιαστούν ακολούθως για τον προσδιορισμό της τιμής ενός ΔΑ μπορούν να χρησιμοποιηθούν για να καθοριστεί και η τιμή ενός ΔΠ.

2.4 Μοντέλα αποτίμησης

2.4.1 Το διωνυμικό μοντέλο

Το διωνυμικό μοντέλο (binomial model, Cox et al. [21], 1979, Cox και Rubinstein [22], 1985) αποτελεί την απλούστερη προσέγγιση στην αποτίμηση των δικαιωμάτων προαίρεσης. Είναι ένα μοντέλο διακριτού χρόνου σύμφωνα με το οποίο, η ανάλυση βασίζεται στις μεταβολές του υποκείμενου προϊόντος σε διακριτά χρονικά σημεία μέχρι τη λήξη του δικαιώματος.

Έστω ότι σε κάποια χρονική στιγμή t ένα χαρτοφυλάκιο συνδυάζει την έκδοση ενός ΔΑ και την αγορά a τεμαχίων του υποκείμενου προϊόντος. Η αξία του ΔΑ είναι C, έχει τιμή εξάσκησης και λήγει μετά από μία χρονική περίοδο. Η αξία του υποκείμενου προϊόντος στη δεδομένη χρονική στιγμή t είναι S. Κατά τη χρονική περίοδο μέχρι τη λήξη του ΔΑ η αξία του υποκείμενου προϊόντος μπορεί να αυξηθεί στην τιμή uS ή να μειωθεί στην τιμή dS, όπου οι συντελεστές u και d προσδιορίζονται με βάση τις αντίστοιχες λογαριθμικές αποδόσεις.

Οι ταμειακές ροές του χαρτοφυλακίου στη λήξη του ΔΑ παρουσιάζονται στον ακόλουθο πίνακα. Στον πίνακα αυτό ως C_u (C_d) συμβολίζεται η αξία του ΔΑ στην περίπτωση που αυξηθεί (μειωθεί) η τιμή του υποκείμενου προϊόντος. Βάσει της ανάλυσης που έχει πραγματοποιηθεί για την αξία ενός ΔΑ, τα C_u και C_d υπολογίζονται ως εξής: $C_u = max\{0, uS - K\}$ και $C_d = max\{0, dS - K\}$.

		Λήξη	
	t	Αύξηση	Μείωση
Έκδοση ΔΑ	C	$-C_u$	$-C_d$
Αγορά α τεμαχίων του ΥΠ	$-aS$	auS	adS

Πίνακας 2.3: Ταμειακές ροές του χαρτοφυλακίου στη λήξη του ΔΑ

Έστω ότι ο επενδυτής ενδιαφέρεται να συνθέσει το χαρτοφυλάκιό του έτσι ώστε το αποτέλεσμα της επένδυσης στη λήξη να μην επηρεάζεται από μεταβολές στην τιμή του υποκείμενου προϊόντος. Ένα τέτοιο χαρτοφυλάκιο λέγεται ότι αντισταθμίζει τον κίνδυνο (hedged portfolio).Για ένα τέτοιο χαρτοφυλάκιο πρέπει:

$$-C_u + auS = -C_d + adS \Rightarrow a = \frac{C_u - C_d}{S(u-d)} \tag{2.2}$$

Ο λόγος αυτός προσδιορίζει τον αριθμό των τεμαχίων του υποκείμενου προϊόντος που πρέπει να αγοραστούν ώστε τα αποτελέσματα του χαρτοφυλακίου να μην επηρεάζονται από τις μεταβολές στην τιμή του υποκείμενου προϊόντος και αναφέρεται ως συντελεστής αντιστάθμισης κινδύνου (hedge ratio).

Το βέβαιο αποτέλεσμα που αποφέρει με τον τρόπο αυτό το παραπάνω χαρτοφυλάκιο, μπορεί να επιτευχθεί και με την επένδυση κάποιου κεφαλαίου για το χρονικό διάστημα μέχρι τη λήξη του ΔΑ, με ένα σταθερό επιτόκιο i. Το κεφάλαιο X που θα πρέπει να επενδυθεί είναι:

$$X = \frac{-C_u + auS}{1+i} = \frac{-C_d + adS}{1+i}$$

Εφόσον υπάρχουν δύο τρόποι (χαρτοφυλάκιο ή επένδυση) για να επιτευχθεί το ίδιο αποτέλεσμα, θα πρέπει η παρούσα αξία τους να είναι ίση, διαφορετικά θα υπάρχει δυνατότητα πραγματοποίησης άμεσου κέρδους, δηλαδή μια ανισορροπία η οποία δεν μπορεί να διαρκέσει πολύ. Συνεπώς θα πρέπει:

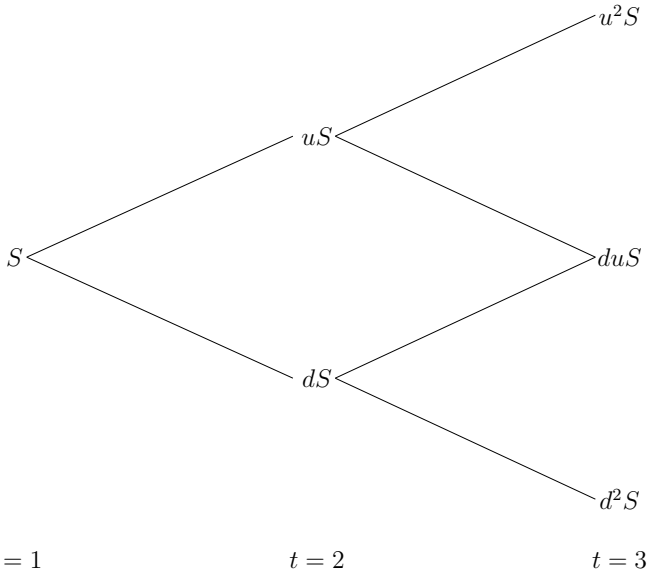
$$C - aS = -X \Rightarrow C - aS = \frac{-C_d + adS}{1+i} \Rightarrow C = \frac{a(1+i)S + C_d - adS}{1+i} \tag{2.3}$$

Αντικαθιστώντας στη σχέση αυτή το πλήθος των τεμαχίων του υποκείμενου προϊόντος που περιλαμβάνονται στο χαρτοφυλάκιο, όπως αυτό προσδιορίστηκε από τη σχέση 2.2, προκύπτει ότι:

$$C = \frac{C_u p + C_d(1-p)}{r} \tag{2.4}$$

όπου $r = 1 + i$ και $p = (r - d)/(u - d)$.

Αυτή η σχέση προσδιορίζει την αξία του ΔΑ, όταν απομένει μια χρονική περίοδος μέχρι τη λήξη του. Βάσει της σχέσης αυτής και ακολουθώντας την ίδια συλλογιστική, είναι δυνατή η αποτίμηση του ΔΑ κάθε χρονική στιγμή πριν τη λήξη του και βέβαια στη χρονική στιγμή 0. Στην περίπτωση όπου απομένουν δύο χρονικές περίοδοι μέχρι τη λήξη του ΔΑ, η πορεία της μετοχής μέχρι τη λήξη του ΔΑ αποδίδεται στο παρακάτω Σχήμα.



Σχήμα 2.3: Η πορεία της μετοχής όταν απομένουν δύο χρονικές περίοδοι μέχρι τη λήξη του ΔΑ.

2.4.2 Η προσομοίωση Monte-Carlo

Η προσομοίωση Monte-Carlo χρησιμοποιείται για την αποτίμηση δικαιωμάτων προαίρεσης της ουδέτερης κινδύνου εκτίμησης. Αρχικά, δειγματοληπτούνται μονοπάτια αποδόσεων, έτσι ώστε να βρεθεί η αναμενόμενη απόδοση σε περιβάλλον ουδέτερο κινδύνου και στη συνέχεια, για την απόδοση αυτή, εκτιμάται η παρούσα αξία.

Έστω ένα χρηματοοικονομικό παράγωγο το οποίο εξαρτάται από μία και μόνο μεταβλητή S της αγοράς και παρέχει κάποια απόδοση σε χρόνο T. Υποθέτοντας ότι το επιτόκιο παραμένει σταθερό, η αποτίμηση του παραγώγου μπορεί να γίνει ως εξής:

1. Δειγματοληψία ενός τυχαίου μονοπατιού για τη μεταβλητή S σε περιβάλλον χωρίς κίνδυνο.

2. Υπολογισμός απόδοσης του παραγώγου.

3. Επανάληψη των βημάτων 1 και 2 ώστε να εκτιμηθούν όσο το δυνατόν περισσότερες τιμές απόδοσης.

4. Υπολογισμός του μέσου όρου του δείγματος των αποδόσεων ώστε να εκτιμηθεί η αναμενόμενη απόδοση σε περιβάλλον χωρίς κίνδυνο.

5. Προεξόφληση της αναμενόμενης απόδοσης με το χωρίς κίνδυνο επιτόκιο, στην παρούσα αξία.

Έστω ότι η διαδικασία η οποία ακολουθείται από την μεταβλητή S σε περιβάλλον χωρίς κίνδυνο είναι:

$$dS = \hat{\mu} S dt + \sigma S dz \tag{2.5}$$

Όπου dz είναι μία διαδικασία Wiener ,$\hat{\mu}$ είναι η αναμενόμενη απόδοση σε περιβάλλον χωρίς κίνδυνο και σ είναι η μεταβλητότητα. Για να προσομοιωθεί το μονοπάτι που ακολουθείται από τη μεταβλητή S, διαιρείται η διάρκεια ζωής του παραγώγου σε N χρονικά διαστήματα διάρκειας Δt και η εξίσωση 2.5 προσεγγίζεται ως εξής:

$$S(t + \Delta t) - S(t) = \hat{\mu} S(t) \Delta t + \sigma S(t) \Delta z + \sigma S(t) \epsilon \sqrt{\Delta t} \tag{2.6}$$

Όπου $S(t)$ συμβολίζει την τιμή του S τη χρονική στιγμή t, ϵ είναι το τυχαίο δείγμα από μια κανονική κατανομή με μέσο όρο μηδέν και τυπική απόκλιση ένα. Αυτή η διαδικασία διευκολύνει τον υπολογισμό της τιμής του S τη χρονική στιγμή Δt από την αρχική τιμή του S. Μία προσομοίωση περιλαμβάνει την δημιουργία ενός ολοκληρωμένου μονοπατιού για το S χρησιμοποιώντας N τυχαία δείγματα από μια κανονική κατανομή.

Πρακτικά, συνήθως πιο ακριβής είναι η προσομοίωση του lnS αντί του S. Από το λήμμα

του $Itô$ για το lnS προκύπτει:

$$dlnS = \left(\mu - \frac{\sigma^2}{2}\right) dt + \sigma\sigma dz \tag{2.7}$$

Έτσι ώστε

$$lnS(t + \Delta t) - lnS(t) = \left(\mu - \frac{\sigma^2}{2}\right) \Delta t + \sigma\epsilon\sqrt{\Delta t} \tag{2.8}$$

Η παραπάνω εξίσωση χρησιμοποιείται για την κατασκευή του μονοπατιού για το S.

Το βασικό πλεονέκτημα της προσομοίωσης Monte-Carlo είναι ότι μπορεί να χρησιμοποιηθεί και για αποδόσεις εξαρτημένες τόσο από το μονοπάτι που ακολουθεί η μεταβλητή S όσο και από την τελική της τιμή. Επίσης οι αποδόσεις μπορεί να λαμβάνουν χώρα σε διάφορες χρονικές στιγμές στο χρόνο μέχρι τη λήξη του παραγώγου.

2.4.3 Το μοντέλο των Black και Scholes

Σε προηγούμενη ενότητα αναλύθηκε το διωνυμικό μοντέλο αποτίμησης ΔΑ, το οποίο βασίζεται στην υπόθεση του διακριτού χρόνου. Όταν ο αριθμός των περιόδων μέχρι τη λήξη του ΔΑ είναι πολύ μεγάλος, τότε μπορεί να εξαλειφθεί η υπόθεση του διακριτού χρόνου και να αναπτυχθεί ένα **μοντέλο αποτίμησης σε συνεχές χρόνο**. Το μοντέλο αυτό είναι γνωστό ως το **μοντέλο των Black και Scholes**, από τα ονόματα των δύο ερευνητών που το ανέπτυξαν το 1973 και αποτελεί το κύριο εργαλείο που χρησιμοποιείται σήμερα για την αποτίμηση ΔΑ.

Η ανάπτυξη του μοντέλων των Black και Scholes βασίζεται στην ίδια λογική με αυτήν του διωνυμικού μοντέλου. Υποθέτει δηλαδή, ότι είναι δυνατή η κατασκευή ενός χαρτοφυλακίου αποτελούμενο από ένα ΔΑ και το υποκείμενο προϊόν, έτσι ώστε αντισταθμιστεί ο κίνδυνος, δηλαδή να επιτευχθεί ένα βέβαιο αποτέλεσμα στη λήξη του ΔΑ. Έστω λοιπόν, ότι το χαρτοφυλάκιο συνίσταται στην αγορά q_s τεμαχίων του υποκείμενου προϊόντος και στην έκδοση q_c συμβολαίων ΔΑ. Τότε εάν η αξία του υποκείμενου προϊόντος σε κάποια χρονική στιγμή είναι S και του ΔΑ είναι C, η αξία V του χαρτοφυλακίου είναι $V = q_sS + q_cC$. Συνεπώς, η μεταβολή στην αξία του χαρτοφυλακίου για μεταβολές στην αξία του υποκείμενου προϊόντος και του ΔΑ (οι μεταβολές συμβολίζονται ως dS και dC αντίστοιχα) είναι $dV = q_sdS + q_cdC$. Εφόσον το χαρτοφυλάκιο υποτίθεται ότι κατασκευάζεται έτσι ώστε να αντισταθμίζει τον κίνδυνο, θα πρέπει η μεταβολή της αξίας του σε οποιαδήποτε χρονική περίοδο dt να αντιστοιχεί στην απόδοση ενός ακίνδυνου χρεογράφου. Συγκεκριμένα, εάν η αξία του χαρτοφυλακίου V επενδυόταν για μια χρονική περίοδο dt με βέβαιη απόδοση r, τότε στο τέλος της περιόδου dt θα απέφερε κέρδος ίσο με $rVdt$. Συνεπώς, για να θεωρηθεί ότι το χαρτοφυλάκιο αντισταθμίζει τον κίνδυνο, θα πρέπει:

$$rVdt = q_sdS + q_cdC \Rightarrow r(q_sS + q_cC)dt = q_sdS + q_cdC \tag{2.9}$$

Εάν ο επενδυτής αγοράσει ένα τεμάχιο του υποκείμενου προϊόντος ($q_s = 1$), και ταυτόχρονα εκδώσει q_c τεμάχια από το ΔΑ, για να θεωρηθεί ότι το χαρτοφυλάκιο αντισταθμίζει τον κίνδυνο, θα πρέπει η αξία του χαρτοφυλακίου να παραμένει σταθερή και ανεξάρτητη από μεταβολές στην τιμή του υποκείμενου προϊόντος. Δηλαδή θα πρέπει:

$$\frac{\partial V}{\partial S} = 0 \Leftrightarrow 1 + q_c \frac{\partial C}{\partial S} = 0 \Leftrightarrow q_c = -\frac{1}{\partial C/\partial S}$$

Το αποτέλεσμα αυτό αντικαθίσταται στη σχέση 2.9, και λαμβάνοντας παράλληλα υπόψη ότι ($q_s = 1$), προκύπτει ότι:

$$r\left(S - \frac{C}{\partial C/\partial S}\right)dt = dS - \frac{1}{\partial C/\partial S}dC \Rightarrow dC = \frac{\partial C}{\partial S}dS - rS\frac{\partial C}{\partial S}dt + rCdt \qquad (2.10)$$

Δεδομένης αυτής της σχέσης απαιτείται τώρα ο προσδιορισμός μιας διαδικασίας η οποία θα μπορεί να μοντελοποιήσει επαρκώς τη μεταβολή στην τιμή του υποκείμενου προϊόντος dS. Αυτό γίνεται μέσω της διαδικασίας Wiener (Wiener process). Μια στοχαστική διαδικασία $W = (W_t : t > 0)$ ονομάζεται διαδικασία Wiener,εάν:

- κάθε τιμή W_t της διαδικασίας στην στιγμή t είναι συνεχής με $W_0 = 0$, και
- η μεταβολή ακολουθεί την κανονική κατανομή και είναι ανεξάρτητη από την πορεία της διαδικασίας μέχρι τη στιγμή t.

Βάσει της θεώρησης αυτής, η μεταβολή της τιμής του υποκείμενου προϊόντος μπορεί να αποδοθεί επαρκώς μέσω της ακόλουθης σχέσης:

$$\frac{dS}{S} = \mu dt + \sigma dW \Rightarrow dS = \mu S dt + \sigma S dW \qquad (2.11)$$

όπου μ είναι η στιγμιαία αναμενόμενη απόδοση του υποκείμενου προϊόντος, σ η αντίστοιχη τυπική απόκλιση, και dW είναι ένας τυχαίος παράγοντας ο οποίος έχει τις ιδιότητες μιας διαδικασίας Wiener, και συνεπώς ακολουθεί την κανονική κατανομή $(0, \sqrt{dt})$ με μέση τιμή μηδέν και τυπική απόκλιση $\sqrt{dt}$.

Σημειώνεται ότι εάν $\sigma = 0$, τότε η λύση της διαφορικής εξίσωσης 2.11 είναι $S = S_0 e^{\mu t}$, όπου S_0 είναι η αξία του υποκείμενου προϊόντος τη χρονική στιγμή $t = 0$.

Δεδομένου ότι η τιμή του δικαιώματος είναι συνάρτηση δύο μεταβλητών, της αξίας του υποκείμενου προϊόντος S και του χρόνου t, μπορεί να γίνει χρήση ενός γνωστού θεωρήματος στο χώρο των στοχαστικών διαδικασιών, του θεωρήματος του Ito. Έστω μια συνεχής και διπλά διαφορίσιμη συνάρτηση $f(x, t)$ όπου x είναι μια τυχαία μεταβλητή που ακολουθεί μια στοχαστική διαδικασία της μορφής $dx = \alpha dt + \beta dW$. Τότε, σύμφωνα με

το θεώρημα του Ito ισχύει ότι:

$$df = \beta \frac{\partial f}{\partial x} dW + \left(\alpha \frac{\partial f}{\partial x} + \frac{\partial f}{\partial t} + \frac{1}{2} \beta^2 \frac{\partial^2 f}{\partial x^2} \right) dt$$

Χρησιμοποιώντας το αποτέλεσμα αυτό και αντικαθιστώντας τα f, x, a, b, με τα $C, S, \mu S, \sigma S$, αντίστοιχα, προκύπτει ότι:

$$\begin{aligned} dC &= \sigma S \frac{\partial C}{\partial S} dW + \left(\mu S \frac{\partial C}{\partial S} + \frac{\partial C}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} \right) dt \\ &= \frac{\partial C}{\partial S} (\mu S dt + \sigma S dW) + \frac{\partial C}{\partial t} dt + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} dt \\ &= \frac{\partial C}{\partial S} dS + \frac{\partial C}{\partial t} dt + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} dt \end{aligned} \tag{2.12}$$

Αντικαθιστώντας στη σχέση 2.10 προκύπτει ότι:

$$\frac{\partial C}{\partial t} + rS \frac{\partial C}{\partial S} + \frac{1}{2} \frac{\partial^2 C}{\partial S^2} \sigma^2 S^2 = rC \tag{2.13}$$

Αυτή είναι μια στοχαστική διαφορική εξίσωση η οποία είναι γνωστή ως η στοχαστική διαφορική εξίσωση του μοντέλου των Black και Scholes. Σημειώνεται ότι η ανάπτυξη της εξίσωσης αυτής δεν βασίστηκε σε κάποια υπόθεση σχετικά με τα χαρακτηριστικά του εξεταζόμενου παράγωγου προϊόντος (στην προκειμένη περίπτωση του ΔΑ). Το στοιχείο αυτό δείχνει ότι η εξίσωση 2.13 έχει γενική ισχύ , δηλαδή κάθε παράγωγο προϊόν, η αξία του οποίου προσδιορίζεται συναρτήσει της αξίας S του υποκείμενου προϊόντος και του χρόνου t, ικανοποιεί την εξίσωση 2.13. Επιπλέον, σημειώνεται ότι στην εξίσωση 2.13 δεν εμφανίζεται η αναμενόμενη απόδοση μ του υποκείμενου προϊόντος, αλλά μόνο η μεταβλητότητα σ. Αυτό υποδεικνύει, ότι η αξία του δικαίωματος είναι ανεξάρτητη της αναμενόμενης απόδοσης και προσδιορίζεται μόνο από τη μεταβλητότητα και την αξία S του υποκείμενου προϊόντος, καθώς και από το επιτόκιο r.

2.4.3.1 Αναλυτική επίλυση

Η λύση της στοχαστικής διαφορικής εξίσωσης 2.13 απαιτεί τον καθορισμό κάποιων οριακών συνθηκών οι οποίες διαφέρουν ανάλογα με το παράγωγο προϊόν που εξετάζεται. Στην περίπτωση της αποτίμησης ενός ΔΑ ευρωπαϊκού τύπου, αυτές οι οριακές συνθήκες προκύπτουν από την αξία του ΔΑ κατά τη λήξη του T:

$$C(S_T, T) = \begin{cases} (S_T - K), & \text{εάν } S_T > K \\ 0, & \text{εάν } S_T \leq K \end{cases}$$

Εάν σε κάποια χρονική στιγμή t πριν τη λήξη του ΔΑ, η αξία του υποκείμενου προϊόντος είναι $S = 0$, τότε από τη σχέση 2.11 είναι εμφανές ότι δεν μπορεί να υπάρξει καμία μεταβολή στην αξία του υποκείμενου προϊόντος, και συνεπώς είναι βέβαιο ότι το δικαίωμα δεν θα εξασκηθεί στη λήξη του. Επομένως $C(0,t) = 0$. Αντίθετα, εάν σε κάποια στιγμή t πριν τη λήξη του ΔΑ, η αξία του υποκείμενου προϊόντος είναι $S \to \infty$, τότε είναι σχεδόν βέβαιο ότι $S_T \gg K$ οπότε το δικαίωμα θα εξασκηθεί αποφέροντας έσοδα $S_T - K \approx S_T$. Άρα, η αξία του ΔΑ είναι $C(S,t) \approx S$ όταν $S \to \infty$.

Υπό αυτές τις οριακές συνθήκες η λύση της στοχαστικής διαφορικής εξίσωσης της σχέσης 2.13 δίνει την αξία του ΔΑ:

$$C = SN(d_1) - Ke^{-rT}N(d_2) \tag{2.14}$$

Επιπλέον, χρησιμοποιώντας τη σχέση 2.1 η οποία προσδιορίζει την αξία ενός ΔΠ σε σχέση με την αξία ενός αντίστοιχου ΔΑ, το μοντέλο των Black και Scholes μπορεί εύκολα να χρησιμοποιηθεί για τον προσδιορισμό της αξίας ενός ΔΠ. Δεδομένου ότι το μοντέλο των Black και Scholes είναι ένα μοντέλο συνεχούς χρόνου, η σχέση 2.1 πρέπει να τροποποιηθεί χρησιμοποιώντας συνεχή ανατοκισμό ως εξής:

$$C = S + P - Ke^{-rT}$$

Αντικαθιστώντας όπου C την αξία του ΔΑ σύμφωνα με το μοντέλο των Black και Scholes, προκύπτει ότι:

$$P = Ke^{-rT}N(-d2) - SN(-d1) \tag{2.15}$$

όπου:

$$d_1 = \frac{\ln(S/K) + (r + \sigma^2/2)T}{\sigma\sqrt{T}}$$
$$d_2 = \frac{\ln(S/K) + (r - \sigma^2/2)T}{\sigma\sqrt{T}} = d_1 - \sigma\sqrt{T}$$

και

- $N(d_1)$, $N(d_2)$ είναι το εμβαδόν κάτω από την καμπύλη της τυπικής κανονικής κατανομής έως τα σημεία d_1 και d_2, αντίστοιχα,
- σ είναι η ετησιοποιημένη τυπική απόκλιση των αποδόσεων του υποκείμενου προϊόντος,
- T είναι ο χρόνος που απομένει μέχρι τη λήξη του ΔΑ εκφρασμένος σε έτη.

Οι σχέσεις 2.14 και 2.15 αποτελούν τη μαθηματική έκφραση του μοντέλου των Black και Scholes για την αποτίμηση ΔΑ και ΔΠ Ευρωπαϊκού τύπου αντίστοιχα [23].

Κεφάλαιο 3

Πεπερασμένες Διαφορές

3.1 Εισαγωγή

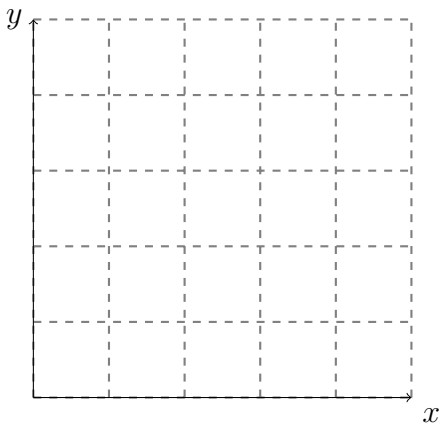
Η μέθοδος των πεπερασμένων διαφορών χρησιμοποιείται για την προσεγγιστική επίλυση διαφορικών εξισώσεων, αντικαθιστώντας τις τελευταίες με εξισώσεις διαφορών. Με τον τρόπο αυτό, το πρόβλημα ανάγεται τελικά στην επίλυση ενός (συνήθως μεγάλου) συστήματος αλγεβρικών, πλέον, εξισώσεων, συνήθως εφαρμόζοντας επαναληπτικές τεχνικές. Η υλοποίηση της συγκεκριμένης μεθόδου μπορεί να αναλυθεί σε τρία βασικά βήματα:

α) στη διακριτοποίηση του χώρου με κατάλληλο πλέγμα κόμβων, οι οποίοι αντιστοιχούν στις θέσεις υπολογισμού του ζητούμενου μεγέθους,

β) στη διατύπωση των εξισώσεων διαφορών, αντικαθιστώντας τους διαφορικούς τελεστές με προσεγγιστικούς,

γ) στην επίλυση των εξισώσεων διαφορών, λαμβάνοντας υπόψη τις αρχικές και οριακές συνθήκες του προβλήματος.

Σχετικά με το πρώτο βήμα, ο πιο συνηθισμένος τύπος πλέγματος που χρησιμοποιείται είναι ο ορθογωνικός, ωστόσο διαφορετικές επιλογές αποδεικνύονται καταλληλότερες σε συγκεκριμένες περιπτώσεις. Χωρίς να είναι απαραίτητο το βήμα διακριτοποίησης να είναι παντού το ίδιο, η ευκολία κατασκευής ενός ορθογωνικού πλέγματος με ομοιόμορφη πυκνότητα σε όλη την έκταση του υπολογιστικού χώρου, το καθιστά ως τη συνηθέστερη επιλογή.



Σχήμα 3.1: Ορθογωνικό πλέγμα

Οι αριθμητικές προσεγγίσεις των διαφορικών τελεστών βασίζονται στον υπολογισμό των τιμών τους με βάση τις τιμές των εξαρτημένων μεταβλητών σε γειτονικά σημεία. Για παράδειγμα, ο υπολογισμός της $1^{ης}$ παραγώγου μιας συνάρτησης f σε ένα σημείο x_0 μπορεί να πραγματοποιηθεί προσεγγιστικά από τις ακόλουθες εκφράσεις, οι οποίες χρησιμοποιούν τιμές της συνάρτησης σε απόσταση h από το σημείο ενδιαφέροντος (ως h θεωρείται το βήμα διακριτοποίησης):

- Προς τα εμπρός προσέγγιση: $f'(x_0) \simeq \dfrac{f(x_0+h)-f(x_0)}{h}$
- Προς τα πίσω προσέγγιση: $f'(x_0) \simeq \dfrac{f(x_0)-f(x_0-h)}{h}$
- Κεντρικά ορισμένη προσέγγιση: $f'(x_0) \simeq \dfrac{f(x_0+h)-f(x_0-h)}{2h}$

Από τις παραπάνω εκφράσεις η τρίτη είναι και η πιο αξιόπιστη, αφού η ακρίβειά της χαρακτηρίζεται ως δεύτερης τάξης (σε αντιδιαστολή με την πρώτη τάξη των δυο πρώτων εκφράσεων). Σημειώνεται πως, η τάξη ακρίβειας μιας προσέγγισης καθορίζεται από την τάξη των σφαλμάτων αποκοπής. Για το λόγο αυτό όσο μεγαλύτερη είναι η τάξη, τόσο ταχύτερη είναι η σύγκλιση των υπολογιζόμενων μεγεθών προς τις πραγματικές τιμές. Όσον αφορά τη δεύτερη παράγωγο της συνάρτησης f , η κεντρικά ορισμένη έκφραση δεύτερης τάξης έχει την ακόλουθη μορφή τριών σημείων:

$$f''(x_0) \simeq \frac{f(x_0+h)-2f(x_0)+f(x_0-h)}{h^2}$$

Στη επόμενη ενότητα θα προσεγγισθεί η εξίσωση διάχυσης με τη μέθοδο των πεπερασμένων διαφορών, *ως παράδειγμα* για την μετέπειτα ανάλυση της διαφορικής εξίσωσης Black-Scholes . Η εξίσωση θερμότητας ή διάχυσης αποτελεί αντιπροσωπευτική εξίσωση των παραβολικών εξισώσεων. Επίσης, οι αναλυτικές λύσεις της εξίσωσης θερμότητας θεωρούνται γνωστές.

3.2 Η εξίσωση θερμότητας ή διάχυσης

Στην παρούσα παράγραφο περιγράφονται οι πλέον βασικές μέθοδοι πεπερασμένων διαφορών που εφαρμόζονται στην αριθμητική (υπολογιστική) επίλυση της εξίσωσης θερμότητας. Εξετάζονται το τυπικό ρητό σχήμα, το τυπικό πεπλεγμένο σχήμα και η μέθοδος Crank-Nicolson.

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} \tag{3.1}$$

με αρχική συνθήκη $u(x,0) = f(x)$ και οριακές συνθήκες $u(0,t) = g_0(t)$ και $u(1,t) = g_1(t)$, όπου οι συναρτήσεις $f(x)$, $g_0(t)$ και $g_1(t)$ και είναι γνωστές.

Το συνεχές πεδίο ορισμού διακριτοποιείται και αντικαθίσταται από το υπολογιστικό πλέγμα. Για την κατασκευή του πλέγματος απαιτείται ο προσδιορισμός των αποστάσεων Δx και Δt στις διαστάσεις x και t αντίστοιχα. Τα όρια του υπολογιστικού πλέγματος ταυτίζονται με τα αντίστοιχα όρια του συνεχούς πεδίου ορισμού. Στη διάσταση x τα όρια ορίζονται από τις οριακές συνθήκες στα $x = 0$ και $x = 1$ και στη $1t$ διάσταση από την αρχική συνθήκη στο $t = 0$. Στην κατεύθυνση $t > 0$ το όριο παραμένει ανοικτό, χαρακτηριστικό των παραβολικών πεδίων. Για καθαρά υπολογιστικούς λόγους που σχετίζονται με το διαθέσιμο υπολογιστικό χρόνο και χώρο του Η/Υ συνήθως επιλέγουμε έναν μέγιστο υπολογιστικό χρόνο t^* πέρα του οποίου δεν εκτελούνται υπολογισμοί ανάλογα με τη φυσική του προβλήματος που εξετάζεται αλλά και το αναγκαίο εύρος και πλήθος των αποτελεσμάτων.

3.2.1 Ρητό σχήμα (explicit scheme)

Η άγνωστη συνάρτηση u σε ένα τυχαίο κόμβο (x_i, t^n) του πλέγματος, όπου i συμβολίζει την χωρική γραμμή και n το χρονικό βήμα, συμβολίζεται με u_i^n. Στον ίδιο κόμβο (x_i, t^n) οι μερικές παράγωγοι της εξίσωσης θερμότητας προσεγγίζονται με εκφράσεις πεπερασμένων διαφορών. Η πρώτη παράγωγος ως προς το χρόνο και η δεύτερη παράγωγος ως προς x, αντικαθίστανται με κατάντη και κεντρώα εκφράσεις πεπερασμένων διαφορών αντίστοιχα. Δηλαδή:

$$\frac{\partial u}{\partial t} = \frac{u_i^{n+1} - u_i^n}{dt} + O(dt) \tag{3.2}$$

και

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{dx^2} + O(dx^2) \tag{3.3}$$

Αντικαθιστώντας τις εκφράσεις (3.2) και (3.3) στην εξίσωση (3.1) προκύπτει η αλγεβρική εξίσωση

$$\frac{u_i^{n+1} - u_i^n}{dt} = \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{dx^2} + O(dt, dx^2) \tag{3.4}$$

που ονομάζεται εξίσωση πεπερασμένων διαφορών. Η εξίσωση (3.4) παίρνει τη μορφή

$$u_i^{n+1} = \lambda u_{i-1}^n + (1-2\lambda)u_i^n + \lambda u_{i+1}^n \tag{3.5}$$

όπου $\lambda = dt/dx^2$. Όπως φαίνεται από την εξίσωση (3.5) ο υπολογισμός της εξαρτημένης μεταβλητής u στον κόμβο $(i, n+1)$ γίνεται απ' ευθείας από τις τιμές της u στους κόμβους $(i-1, n)$, (i, n) και $(i+1, n)$. Σχήματα όπως αυτό που διατυπώνεται με την εξίσωση (3.5) ονομάζονται ρητά επειδή η μετακίνηση από το ένα χρονικό βήμα στο επόμενο γίνεται χωρίς να απαιτείται η επίλυση ενός αλγεβρικού συστήματος. Το αριθμητικό σφάλμα του συγκεκριμένου αριθμητικού σχήματος είναι $O(dt, dx^2)$.

Όπως αποδεικνύεται η παρούσα αριθμητική μέθοδος συγκλίνει μόνο όταν $\lambda < 1/2$. Παρόμοιες συνθήκες ισχύουν για άλλα ρητά σχήματα που βασίζονται σε διαφορετικές εκφράσεις πεπερασμένων διαφορών και εφαρμόζονται στην επίλυση της εξίσωσης θερμότητας η άλλων πιο σύνθετων διαφορετικών εξισώσεων. Οι περιοριστικές αυτές συνθήκες δημιουργούν προβλήματα στην αποτελεσματική αριθμητική επίλυση των μερικών διαφορικών εξισώσεων, που σχετίζονται με το μέγιστο όριο στο μέγεθος του χρονικού βήματος dt. Όταν όμως το χρονικό βήμα είναι μικρό το υπολογιστικό κόστος είναι μεγάλο. Το μειονέκτημα αυτό βελτιώνεται σημαντικά με την εφαρμογή των πεπλεγμένων σχημάτων που εξετάζονται στην επόμενη παράγραφο.

3.2.2 Πεπλεγμένο σχήμα (implicit scheme)

Έστω ότι εξετάζεται το πρόβλημα (3.1) με τις ίδιες αρχικές και οριακές συνθήκες. Τώρα η εξίσωση (3.1) προσεγγίζεται τη χρονική στιγμή $n+1$, αντί για τη χρονική n και εφαρμόζονται μια ανάντη και μια κεντρώα διαφορά στις μερικές παραγώγους ως προς t και x αντίστοιχα. Δηλαδή

$$\frac{\partial u}{\partial t} = \frac{u_i^{n+1} - u_i^n}{dt} + O(dt) \tag{3.6}$$

και

$$\frac{\partial^2 u}{\partial x^2} = \frac{u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}}{dx^2} + O(dx^2) \tag{3.7}$$

Η εξίσωση πεπερασμένων διαφορών παίρνει τη μορφή

$$\frac{u_i^{n+1} - u_i^n 1}{dt} = \frac{u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}}{dx^2} + O(dt, dx^2) \tag{3.8}$$

Επομένως, η λύση της u τη χρονική στιγμή $n+1$ δε γίνεται με ρητό τρόπο αλλά με τη λύση του τριδιαγώνιου συστήματος

$$-\lambda u_{i-1}^{n+1} + (1-2\lambda)u_i^{n+1} - \lambda u_{i+1}^{n+1} = u_i^n \tag{3.9}$$

όπου $\lambda = dt/dx^2$. Το σφάλμα του συγκεκριμένου πεπλεγμένου αριθμητικού σχήματος είναι και πάλι $O(dt, dx^2)$ αλλά το σχήμα συγκλίνει πάντα ανεξάρτητα από τις τιμές του λ.

3.2.3 Μέθοδος Crank-Nicolson

Στο ρητό και πεπλεγμένο σχήμα των προηγούμενων ενοτήτων αντίστοιχα, το σφάλμα διακριτοποίησης που οφείλεται στην αποκοπή όρων από τη σειρά Taylor είναι $O(dt, dx^2)$. Πολλές φορές είναι επιθυμητό το σφάλμα να είναι ίδιας τάξης στις μεταβλητές x και t. Αυτό επιτυγχάνεται προσεγγίζοντας την μ.δ.ε (3.1) στο σημείο $(i, n+1/2)$ ενδιάμεσα των σημείων (i, n) και $(i, n+1)$.

Εφαρμόζοντας κεντρώες εκφράσεις πεπερασμένων διαφορών και στις δύο μερικές παραγώγ1ους έχουμε

$$\frac{\partial u}{\partial t} = \frac{u_i^{n+1} - u_i^n}{dt} + O(dt) \tag{3.10}$$

και

$$\frac{\partial^2 u}{\partial x^2} = \theta \left.\frac{\partial^2 u}{\partial x^2}\right|_i^{n+1} + (1-\theta) \left.\frac{\partial^2 u}{\partial x^2}\right|_i^{n} = \theta \delta_x^2 u_i^{n+1} + (1-\theta)\delta_x^2 u_i^n \tag{3.11}$$

όπου δ_x^2 είναι ο κεντρώος τελεστής δεύτερης τάξης

$$\delta_x^2 u_i^n = \frac{u_{i-1}^n - 2u_i^n + u_{i+1}^n}{dx^2} \tag{3.12}$$

και

$$\delta_x^2 u_i^{n+1} = \frac{u_{i-1}^{n+1} - 2u_i^{n+1} + u_{i+1}^{n+1}}{dx^2} \tag{3.13}$$

Για την ειδική περίπτωση που $\theta = 1/2$ προκύπτει η μέθοδος Crank Nicolson που διατυπώνεται από την εξίσωση πεπερασμένων διαφορών

$$-\lambda u_{i-1}^{n+1} + 2(1+\lambda)u_i^{n+1} - \lambda u_{i+1}^{n+1} = \lambda u_{i-1}^n + 2(1-\lambda)u_i^n + \lambda u_{i+1}^n \tag{3.14}$$

με $\lambda = dt/dx^2$. Απαιτείται λοιπόν η επίλυση ενός τριγωνικού αλγεβρικού συστήματος σε κάθε χρονική στιγμή.

Παρατηρούμε ότι θέτοντας $\theta = 0$ ή $\theta = 1$ στην σχέση (3.14) προκύπτει το απλό ρητό και το απλό πεπλεγμένο σχήμα που διατυπώνονται από τις εξισώσεις πεπερασμένων διαφορών (3.5) και (3.9) αντίστοιχα. Για $0 < \theta < 1$ οδηγούμεθα σε μια ομάδα πεπλεγμένων μεθόδων που έχουν χαρακτηριστικά αντίστοιχα με αυτά της μεθόδου Crank Nicolson ($\theta = 1/2$). Η μέθοδος Crank Nicolson είναι πεπλεγμένη ακριβείας1 $O(dt^2, dx^2)$ και συγκλίνει πάντα ανεξάρτητα από τις τιμές της παραμέτρου λ.

3.3 Η εξίσωση Black-Scholes

Η μέθοδος των πεπερασμένων διαφορών εφαρμόστηκε, στην αποτίμηση δικαιωμάτων προαίρεσης, για πρώτη φορά από τους Brennan και Schwartz [24]. Σε αυτή την ενότητα περιγράφονται οι βασικές αρχές της μεθόδου όπως εφαρμόζονται στην εξίσωση Black-Scholes.

$$rC = \frac{\partial C}{\partial t} + rS\frac{\partial C}{\partial S} + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}\sigma^2 S^2$$

Οι ανεξάρτητες μεταβλητές της εξίσωσης Black-Scholes είναι η τιμή του υποκείμενου προϊόντος S και ο χρόνος t που έχει περάσει από την υπογραφή του χρεόγραφου. Αυτές οι δύο μεταβλητές σχηματίζουν ένα δισδιάστατο χώρο, όπου η συνάρτηση που περιγράφει την αποτίμηση του δικαιώματος προαίρεσης $C(S,t)$ έχει την μορφή μιας τρισδιάστατης καμπύλης. Με αυτό το τρόπο, αντί να προσπαθούμε να βρούμε μια έκφραση κλειστής μορφής για την συνάρτηση $C(S,t)$, με τη μέθοδο των πεπερασμένων διαφορών προσεγγίζουμε κάθε τιμή πάνω στα διακριτά σημεία του χώρου που σχηματίζουν ένα πλέγμα.

3.3.1 Επιλογή πλέγματος

Ο χρόνος t παίρνει τιμές στο διάστημα $[0,T]$ όπου 0 είναι η στιγμή που γίνεται η πράξη της αγοράς ή πώλησης του δικαιώματος προαίρεσης και T όταν έχει ωριμάσει το δικαίωμα, δηλαδή στη ημερομηνία λήξης του.

Αντικαθιστούμε τον χρόνο t με $\tau = (T-t)$, όπου $0 \leq \tau < T$, ώστε οι τελικές συνθήκες

$$C(S,t) = max(S-K,0) \text{ όπου } t = T$$

να μετασχηματιστούν σε αρχικές συνθήκες

$$C(S,\tau) = max(S-K,0) \text{ όπου } \tau = 0.$$

Μετά το μετασχηματισμό αυτό η μερική διαφορική εξίσωση Black-Scholes γίνεται:

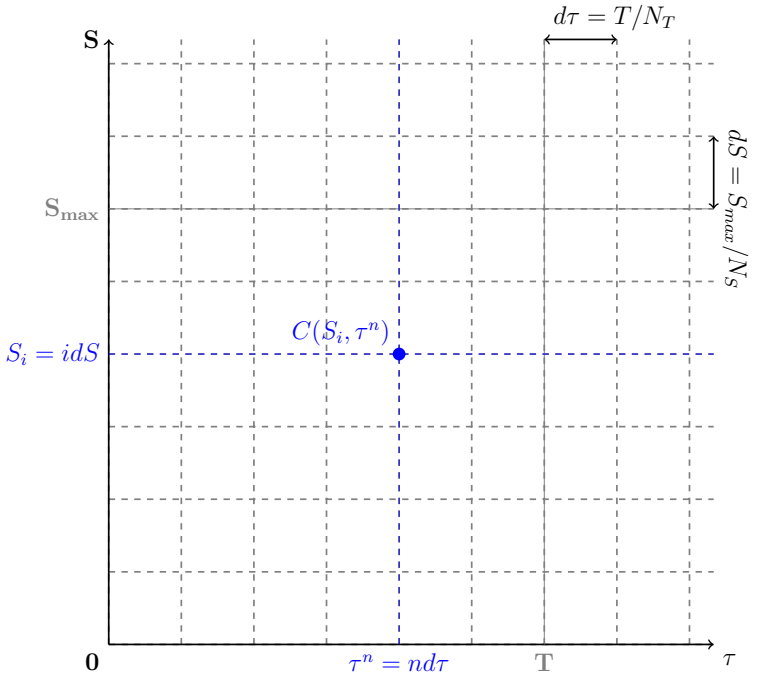
$$rC = -\frac{\partial C}{\partial \tau} + rS\frac{\partial C}{\partial S} + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}\sigma^2 S^2 \tag{3.15}$$

Η τιμή του υποκείμενου προϊόντος S παίρνει τιμές στο διάστημα $[0,\infty]$ αφού ένα προϊόν μπορεί να πάρει μηδενική ή θετική τιμή. Για να μπορέσουμε να ορίσουμε ένα πεπερασμένο αριθμό από διακριτά σημεία στην διάσταση του προϊόντος, πρέπει να θέσουμε μία μέγιστη τιμή S_{max} που μπορεί να πάρει το προϊόν.

Εκτός από την επιλογή ενός ανώτατου ορίου για την τιμή της S, ένα ακόμη ζήτημα είναι και το πόσα σημεία πρέπει να έχει το πλέγμα, αυτό πολλές φορές έχει να κάνει με τον τύπο της μέθοδου πεπερασμένων διαφορών που θα χρησιμοποιηθεί. Στη συγκεκριμένη

περίπτωση θεωρούμε ότι ο χώρος (S, τ) διαιρείται σε N_τ στη διάσταση του χρόνου και N_S στη διασταση της μεταβλητής S , άρα έχουμε ένα πλέγμα φραγμένο στο $[0, S_{max}]$ για S και $[0, T]$ για τ.

Ένα ακόμα στοιχείο που πρέπει να γνωρίζουμε πριν καταλήξουμε στη μορφή του πλέγματος είναι το βήμα διακριτοποίησης. Όπως αναφέρθηκε και στην εισαγωγή του κεφαλαίου, ο πιο συνηθισμένος τύπος πλέγματος που χρησιμοποιείται είναι ο ορθογωνικός γιατί η ευκολία κατασκευής ενός ορθογωνικού πλέγματος με ομοιόμορφη πυκνότητα σε όλη την έκταση του υπολογιστικού χώρου το καθιστά ως καταλληλότερο για την υλοποίησή του σε αναδιατασσόμενη λογική.



Σχήμα 3.2: Πεπερασμένο πλέγμα του υπολογιστικού χώρου (S, τ) που σχηματίζουν οι ανεξάρτητες μεταβλητές της μ.δ.ε. Black-Scholes.

Για ένα ομοιόρφο πλέγμα μεγέθους $N_S \times N_\tau$, μεγέθους $[0, T]$ για τ και $[0, S_{max}]$ για S η απόσταση dS μεταξύ δύο συνεχόμενων σημείων είναι $dS = S_{max}/N_S$ στην κατεύθυνση S και $d\tau = T/N_T$ στην κατεύθυνση τ (Σχήμα 3.2). Η τιμή της συνάρτησης στο σημείο του πλέγματος (S_i, τ^n) στο εξής θα συμβολίζεται με C_i^n και θα προσεγγίζει τη πραγματική τιμή στο σημείο αυτό $C_i^n = C(S_i, \tau^n)$.

Οπότε η εξίσωση Black-Scholes παίρνει την εξής μορφή στο διακριτό υπολογιστικό χώρο:

$$rC_i^n = -\frac{\partial C}{\partial \tau} + rS_i \frac{\partial C}{\partial S} + \frac{1}{2}\frac{\partial^2 C}{\partial S^2}\sigma^2 S_i^2 \tag{3.16}$$

Το επόμενο βήμα είναι να εφαρμόσουμε τα σχήματα πεπερασμένων διαφορών όπως αναλύθηκαν στις προηγούμενες ενότητες για να προσεγγίσουμε τις μερικές παραγώγους

της μ.δ.ε. Εδώ πρέπει να σημειώσουμε ότι στο Κεφάλαιο 2 έγινε ανάλυση για όλες τις περιπτώσεις δικαιωμάτων προαίρεσης Ευρωπαϊκού και Αμερικάνικου τύπου, στη συνέχεια η ανάλυση γίνεται για Δικαίωμα Αγοράς Ευρώπαϊκού τύπου. Οπότε είναι αναγκαίο να αναφερθούμε ξανά στις οριακές συνθήκες αυτού του τύπου δικαιώματος προαίρεσης λαμβάνοντας υπόψιν και την ανάλυση που κάνουμε σε αυτό το κεφάλαιο με τη θεωρία των πεπερασμένων διαφορών.

3.3.2 Οριακές συνθήκες

Οι οριακές συνθήκες για την εξίσωση Black-Scholes, στην περίπτωση Δικαιώματος αγοράς Ευρωπαϊκού τύπου, εκφρασμένες στο διακριτό πλέγμα που χρησιμοποιούν οι πεπερασμένες διαφορές και εκφράζουν τη συμπεριφορά της εξίσωσης στα σύνορα του πλέγματος (κατά Dirichlet) είναι:

- Η συνθήκη $C(S,t) = max(S-K,0)$ με $t = T$, που μας δίνει τη τιμή του ΔΑ στη λήξη του και είναι η τελική συνθήκη για την εξίσωση Black-Scholes στο συνεχές, στο διακριτό γίνεται:

  $$C_i^n = max(S_i - K, 0) \text{ με } \tau = 0$$

  όπου τώρα είναι η αρχική μας συνθήκη.

- Όταν η τιμή του υποκείμενου προϊόντος τείνει στο άπειρο $S \to \infty$ η αξία του ΔΑ είναι ίση με την αξία του υποκείμενου προϊόντος μείον την τιμή εξάσκησης μειώμενη από τον τόκο που θα είχε μία επένδυση χωρίς κίνδυνο ,όπως η κατάθεση του ποσού K σε ένα τραπεζικό λογαριασμό, δηλαδή στο συνεχές έχουμε $C(S_T,t) = S_T - Ke^{-rdt}$. Επειδή όμως $S_T >> K$ πολλές φορές χρησιμοποιείται $C(S_T,t) \approx S_T$. Στο διακριτό έχουμε:

  $$C_{N_S}^n = S_{max} - Ke^{-rnd\tau} \text{ όταν } S \to \infty$$

  ή

  $$C_i^n \approx S_{max} \text{ όταν } S \to \infty$$

- Όταν η αξία του υποκείμενου προϊόντος είναι $S = 0$, τότε είναι εμφανές ότι δεν μπορεί να υπάρξει καμία μεταβολή στην αξία του υποκείμενου προϊόντος, και συνεπώς είναι βέβαιο ότι το δικαίωμα δεν θα εξασκηθεί στη λήξη του. Επομένως $C(0,t) = 0$. Στο διακριτό η συνθήκη γίνεται:

  $$C_0^n = 0 \text{ με } S = 0$$

Για τις οριακές συνθήκες που αφορούν τη τιμή των παραγώγων στην εξίσωση στα όρια του συστήματος (κατά Von Neumann) έχουμε:

- Όταν η τιμή του υποκείμενου προϊόντος τείνει στο άπειρο $S \to \infty$ έχουμε:

$$\frac{\partial C}{\partial S} = 1 \text{ όταν } S \to \infty$$

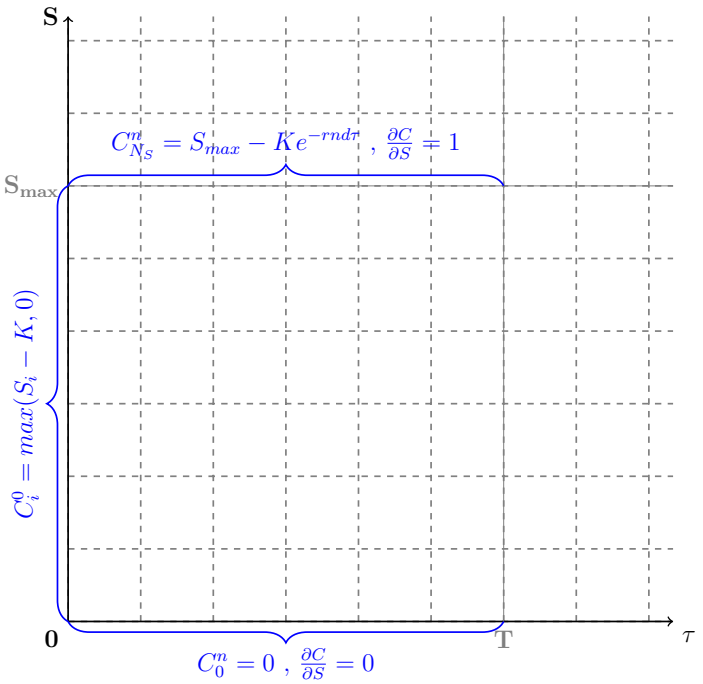
και

$$\frac{\partial^2 C}{\partial S^2} = 0 \text{ όταν } S \to \infty$$

- Όταν η αξία του υποκείμενου προϊόντος είναι $S = 0$, τότε

$$\frac{\partial C}{\partial S} = 0 \text{ με } S = 0$$

Στο διακριτό έχουν την ιδια μορφή με το συνεχές αλλά ανταποκρίνονται στα διακριτά όρια $[0, S_{max}]$.



Σχήμα 3.3: Οριακές συνθήκες στα άκρα του διακριτού πλέγματος.

Στην εφαρμογή των μεθόδων πεπερασμένων διαφορών κάνουμε χρήση των Von Neumann οριακών συνθήκων. Αυτό γίνεται γιατί δεν χρειάζεται να υπολογίσουμε επιπλέον τιμές για τα όρια. Οι συνθήκες Dirichlet για S_{max} πηγάζουν από ένα περίπλοκο τύπο και η εφαρμογή του σε υλικό αναδιατασόμενης λογικής είναι ένα πρόβλημα από μόνο του. Ένας τρόπος για να χρησιμοποιηθούν αυτές οι συνθήκες είναι ο υπολογισμός τους στον κεντρικό υπολογιστή και την αποστολή τους στον αποτιμητή στην FPGA . Αυτή η επιλογή έρχεται με αυξημένο κόστος καθυστέρησης I/O. Οι Von Neumann από την

άλλη πλευρά είναι ισοδύναμες με τις Dirichlet όπως παράγονται από αυτές και μπορούν να ενσωματωθούν στην εφαρμοζόμενη μεθόδο [25].

3.3.3 Ρητό σχήμα (explicit scheme)

Στην προηγούμενη ενότητα 3.3.3 αναφέραμε ως χαρακτηριστικό παράδειγμα την εξίσωση διάχυσης, τώρα θα ακολουθήσουμε την ίδια μεθοδολογία για την εξίσωση Black-Scholes . Η άγνωστη συνάρτηση C σε ένα τυχαίο κόμβο (S_i, τ^n) του πλέγματος, όπου i συμβολίζει την γραμμή αξίας του προϊόντος και n το χρονικό βήμα, συμβολίζεται με C_i^n. Στον ίδιο κόμβο (S_i, τ^n) οι μερικές παράγωγοι της εξίσωσης Black-Scholes προσεγγίζονται με εκφράσεις πεπερασμένων διαφορών.

Η πρώτη παράγωγος ως προς το χρόνο αντικαθίσταται με κατάντη έκφραση πεπερασμένων διαφορών:

$$\frac{\partial C}{\partial \tau} = \frac{C_i^{n+1} - C_i^n}{d\tau} + O(d\tau) \tag{3.17}$$

Η πρώτη και η δεύτερη παράγωγος ως προς S, αντικαθίστανται με κεντρώα έκφραση πεπερασμένων διαφορών. Δηλαδή:

$$\frac{\partial C}{\partial S} = \frac{C_{i+1}^n - C_{i-1}^n}{2dS} + O(dS^2) \tag{3.18}$$

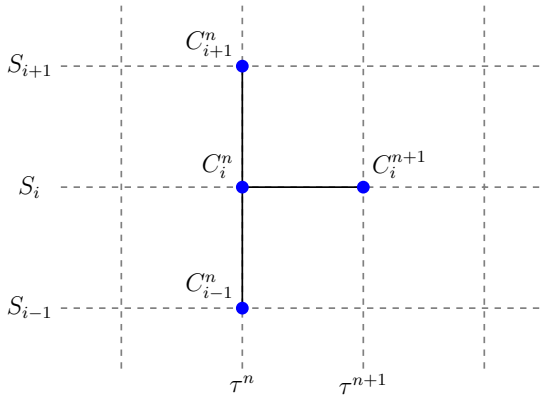
και

$$\frac{\partial^2 C}{\partial S^2} = \frac{C_{i+1}^n - 2C_i^n + C_{i-1}^n}{dS^2} + O(dS^2) \tag{3.19}$$

Αντικαθιστώντας τις εκφράσεις (3.17) - (3.19) στην εξίσωση (3.16) προκύπτει η αλγεβρική εξίσωση

$$rC_i^n = -\frac{C_i^{n+1} - C_i^n}{d\tau} + rS_i\frac{C_{i+1}^n - C_{i-1}^n}{2dS} + \frac{1}{2}\frac{C_{i+1}^n - 2C_i^n + C_{i-1}^n}{dS^2}\sigma^2 S_i^2 \tag{3.20}$$

που ονομάζεται εξίσωση πεπερασμένων διαφορών.



Σχήμα 3.4: Ρητό σχήμα για τον υπολογισμό της αξίας του δικαιώματος πάνω στο πλέγμα.

Η εξίσωση (3.20) παίρνει τη μορφή

$$C_i^{n+1} = a_i \cdot C_{i-1}^n + b_i \cdot C_i^n + c_i \cdot C_{i+1}^n \tag{3.21}$$

όπου

$$\begin{aligned} a_i &= -\frac{1}{2} rid\tau + \frac{1}{2}\sigma^2 i^2 d\tau \\ b_i &= 1 - rd\tau - \sigma^2 i^2 d\tau \\ c_i &= \frac{1}{2} rid\tau + \frac{1}{2}\sigma^2 i^2 d\tau \end{aligned}$$

Όπως φαίνεται από την εξίσωση (3.21) ο υπολογισμός της εξαρτημένης μεταβλητής C στον κόμβο $(i, n+1)$ γίνεται απ' ευθείας από τις τιμές της C στους κόμβους $(i-1, n)$, (i, n) και $(i+1, n)$. Σχήματα όπως αυτό που διατυπώνεται με την εξίσωση (3.21) ονομάζονται ρητά επειδή η μετακίνηση από το ένα χρονικό βήμα στο επόμενο γίνεται χωρίς να απαιτείται η επίλυση ενός αλγεβρικού συστήματος. Το αριθμητικό σφάλμα του συγκεκριμένου αριθμητικού σχήματος είναι $O(dt, dx^2)$.

Η αναγκαία συνθήκη για την ευστάθεια της μεθόδου είναι ότι οι συντελεστές a_i, b_i, c_i πρέπει να μην είναι αρνητικοί για όλα τα i [24]. Μπορεί να αποδειχθεί ότι πρέπει $(d\tau/d^2S) \approx 1$ για να ισχύει $b_i > 0$. Αυτή η συνθήκη μας δείχνει ότι τα χρονικά βήματα πρέπει να είναι ίσα με τα σημεία του τετραγώνου του υποκείμενου προϊόντος. Οι περιοριστικές αυτές συνθήκες δημιουργούν προβλήματα στην αποτελεσματική αριθμητική επίλυση των μερικών διαφορικών εξισώσεων, που σχετίζονται με το μέγιστο όριο στο μέγεθος του χρονικού βήματος $d\tau$. Όταν όμως το χρονικό βήμα είναι μικρό το υπολογιστικό κόστος είναι μεγάλο. Το μειονέκτημα αυτό βελτιώνεται σημαντικά με την εφαρμογή των πεπλεγμένων σχημάτων που εξετάζονται στην επόμενη παράγραφο.

3.3.4 Πεπλεγμένο σχήμα (implicit scheme)

Στο πεπλεγμένο σχήμα η εξίσωση (3.16) προσεγγίζεται τη χρονική στιγμή n χρησιμοποιώντας την τιμή της C_i^{n-1} στο προηγούμενο χρονικό βήμα (Σχήμα 3.4).

Η πρώτη παράγωγος ως προς το χρόνο αντικαθίσταται με ανάντη έκφραση πεπερασμένων διαφορών:

$$\frac{\partial C}{\partial \tau} = \frac{C_i^n - C_i^{n-1}}{d\tau} + O(d\tau) \tag{3.22}$$

Η πρώτη και η δεύτερη παράγωγος ως προς S, αντικαθίστανται με κεντρώα έκφραση πεπερασμένων διαφορών. Δηλαδή:

$$\frac{\partial C}{\partial S} = \frac{C_{i+1}^n - C_{i-1}^n}{2dS} + O(dS^2) \tag{3.23}$$

και

$$\frac{\partial^2 C}{\partial S^2} = \frac{C_{i+1}^n - 2C_i^n + C_{i-1}^n}{dS^2} + O(dS^2) \tag{3.24}$$

Αντικαθιστώντας τις εκφράσεις (3.22) - (3.24) στην εξίσωση (3.16) προκύπτει η αλγεβρική εξίσωση

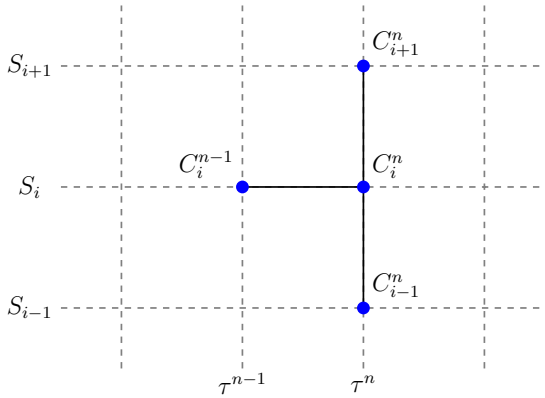
$$rC_i^n = -\frac{C_i^n - C_i^{n-1}}{d\tau} + rS_i\frac{C_{i+1}^n - C_{i-1}^n}{2dS} + \frac{1}{2}\frac{C_{i+1}^n - 2C_i^n + C_{i-1}^n}{dS^2}\sigma^2 S_i^2 \tag{3.25}$$

που ονομάζεται εξίσωση πεπερασμένων διαφορών. Η εξίσωση (3.24) παίρνει τη μορφή

$$C_i^{n-1} = a_i \cdot C_{i-1}^n + b_i \cdot C_i^n + c_i \cdot C_{i+1}^n \tag{3.26}$$

όπου

$$\begin{aligned} a_i &= -\frac{1}{2}rid\tau + \frac{1}{2}\sigma^2 i^2 d\tau \\ b_i &= 1 - rd\tau - \sigma^2 i^2 d\tau \\ c_i &= \frac{1}{2}rid\tau + \frac{1}{2}\sigma^2 i^2 d\tau \end{aligned}$$



Σχήμα 3.5: Πεπλεγμένο σχήμα για τον υπολογισμό της αξίας του δικαιώματος πάνω στο πλέγμα.

Η εξίσωση (3.25) φραγμένη στο διάστημα $0 < i < N_s$ αποτελεί ένα τριγωνικό σύστημα που πρέπει να λύνεται για κάθε χρονική στιγμή. Το σφάλμα του συγκεκριμένου πεπλεγμένου αριθμητικού σχήματος είναι και πάλι $O(dt, dx^2)$ αλλά συγκλίνει πάντα, έτσι

η καλύτερη ευστάθεια ισοσταθμίζει το μεγάλο υπολογιστικό φόρτο σε σχέση με το ρητό σχήμα.

Το σχήμα σε μορφή πινάκων γίνεται:

$$\begin{bmatrix} b_0 & c_0 & 0 & \cdots & 0 \\ a_1 & b_1 & c_1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & a_{N_S-1} & b_{N_S-1} & c_{N_S-1} \\ 0 & \cdots & 0 & a_{N_S} & b_{N_S} \end{bmatrix} \begin{bmatrix} cC_0^n \\ C_1^n \\ \vdots \\ C_{N_S-1}^n \\ C_{N_S}^n \end{bmatrix} = \begin{bmatrix} cC_0^{n-1} \\ C_1^{n-1} \\ \vdots \\ C_{N_S-1}^{n-1} \\ C_{N_S}^{n-1} \end{bmatrix}$$

Πολλές φορές είναι επιθυμητό το σφάλμα να είναι ίδιας τάξης στις μεταβλητές S και τ. Ώστε να μπορούμε να κάνουμε μεγαλύτερα χρονικά βήματα και να έχουμε μικρό σφάλμα. Αυτό επιτυγχάνεται με τη μέθοδο Crank-Nicolson , όπως θα δούμε στην επόμενη παράγραφο.

3.3.5 Μέθοδος Crank-Nicolson

Στο ρητό και πεπλεγμένο σχήμα των προηγούμενων ενοτήτων αντίστοιχα, το σφάλμα διακριτοποίησης που οφείλεται στην αποκοπή όρων από τη σειρά Taylor είναι $O(d\tau, dS^2)$. Στη μέθοδο Crank-nicolson το σφάλμα διακριτοποίησης $O(d\tau^2, dS^2)$, αυτό επιτυγχάνεται προσεγγίζοντας την μ.δ.ε (3.16) στο σημείο $(i, n+1/2)$ ενδιάμεσα των σημείων $(i, n+1)$ και (i, n):

$$C_i^{(n+1)/2} = \frac{C_i^{n+1} - C_i^n}{2} \tag{3.27}$$

Εφαρμόζοντας κεντρώες εκφράσεις πεπερασμένων διαφορών σε όλες, πρώτης και δευτέρας τάξης, μερικές παραγώγους έχουμε για την μεταβολή ως προς το χρόνο:

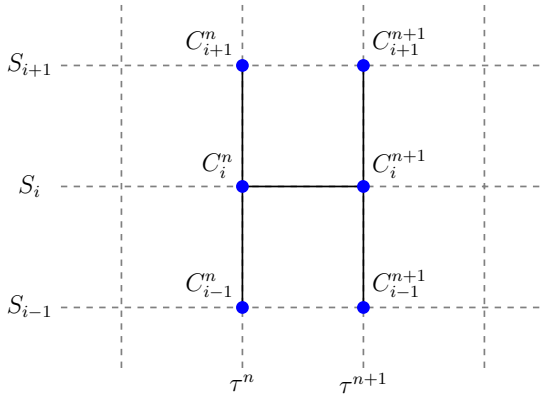
$$\frac{\partial C}{\partial \tau} = \frac{C_i^{n+1} - C_i^n}{d\tau} \tag{3.28}$$

Για την μεταβολή ως προς την τιμή του υποκείμενου προϊόντος:

$$\frac{\partial C}{\partial S} = \frac{C_{i+1}^{n+1} + C_{i+1}^n - C_{i-1}^{n+1} - C_{i-1}^n}{4dS} \tag{3.29}$$

και

$$\frac{\partial^2 C}{\partial S^2} = \frac{C_{i+1}^{n+1} + C_{i+1}^n - 2C_i^{n+1} - 2C_i^n - C_{i-1}^{n+1} - C_{i-1}^n}{2dS^2} \tag{3.30}$$



Σχήμα 3.6: Crank-nicolson σχήμα για τον υπολογισμό της αξίας του δικαιώματος πάνω στο πλέγμα.

Αντικαθιστώντας τις εκφράσεις (3.27) - (3.30) στην εξίσωση (3.16) προκύπτει η εξίσωση πεπερασμένων διαφορών:

$$\begin{aligned} r\frac{C_i^{n+1}-C_i^n}{2} = &-\frac{C_i^{n+1}-C_i^n}{d\tau} \\ &+ rS_i\frac{C_{i+1}^{n+1}+C_{i+1}^n-C_{i-1}^{n+1}-C_{i-1}^n}{4dS} \\ &+ \frac{1}{2}\frac{C_{i+1}^{n+1}+C_{i+1}^n-2C_i^{n+1}-2C_i^n-C_{i-1}^{n+1}-C_{i-1}^n}{2dS^2}\sigma^2 S_i^2 \end{aligned} \tag{3.31}$$

Η εξίσωση (3.31) παίρνει τη μορφή:

$$a_i \cdot C_{i-1}^{n+1} + b_i \cdot C_i^{n+1} + c_i \cdot C_{i+1}^{n+1} = e_i^n \tag{3.32}$$

με

$$e_i^n = d_{0,i} \cdot C_{i-1}^n + d_{1,i} \cdot C_i^n + d_{2,i} \cdot C_{i+1}^n \tag{3.33}$$

και όπου

$$\begin{aligned} a_i &= \frac{1}{2}rid\tau - \frac{1}{2}\sigma^2 i^2 d\tau \\ b_i &= 2 + rd\tau + \sigma^2 i^2 d\tau \\ c_i &= -\frac{1}{2}rid\tau - \frac{1}{2}\sigma^2 i^2 d\tau \\ d_{0,i} &= -a_i \\ d_{1,i} &= 2 - rd\tau - \sigma^2 i^2 d\tau \\ d_{2,i} &= -c_i \end{aligned}$$

Το παραπάνω σύστημα εξισώσεων μαζί με τις οριακές συνθήκες της ενότητας (3.3.2) σχηματίζουν ένα γραμμικό τριδιαγώνιο σύστημα με αγνώστους την τιμή του ΔΑ για το

χρονικό βήμα $n+1$ χρησιμοποιώντας το n χρονικό βήμα. Σε μορφή πινάκων έχουμε:

$$\begin{bmatrix} b_0 & c_0 & 0 & \cdots & 0 \\ a_1 & b_1 & c_1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & a_{N_S-1} & b_{N_S-1} & c_{N_S-1} \\ 0 & \cdots & 0 & a_{N_S} & b_{N_S} \end{bmatrix} \begin{bmatrix} C_0^{n+1} \\ C_1^{n+1} \\ \vdots \\ C_{N_S-1}^{n+1} \\ C_{N_S}^{n+1} \end{bmatrix} = \begin{bmatrix} e_0^n \\ e_1^n \\ \vdots \\ e_{N_S-1}^n \\ e_{N_S}^n \end{bmatrix}$$

με:

$$\begin{bmatrix} e_0^n \\ e_1^n \\ \vdots \\ e_{N_S-1}^n \\ e_{N_S}^n \end{bmatrix} = \begin{bmatrix} d_{1,0}C_0^n + d_{2,0}C_1^n \\ d_{0,1}C_0^n + d_{1,1}C_1^n + d_{2,1}C_2^n \\ \vdots \\ d_{0,N_S-1}C_{N_S-2}^n + d_{1,N_S-1}C_{N_S-1}^n + d_{2,N_S-1}C_{N_S}^n \\ d_{0,N_S}C_{N_S-1}^n + d_{1,N_S}C_{N_S}^n \end{bmatrix}$$

Το σχήμα Crank-Nicolson χρειάζεται την επίλυση ενός τριδιαγώνιου συστήματος για κάθε χρονικό βήμα και την ανανέωση του δεξιού μέλους της εξίσωσης με καινούργιες τιμές για το e_i, όπου $0 < i < N_s$. Για αυτό είναι το πιο απαιτητικό σχήμα πεπερασμένων διαφορών από τα τρία που αναλύθηκαν , όσον αφορά τη υπολογιστική πολυπλοκότητα. Όπως αναφέρθηκε και στην αρχή της παραγράφου, στα θετικά είναι η δευτέρας τάξης ακρίβεια ως προς τη χρονική μεταβολή και ότι συγκλίνει χωρίς προϋποθέσεις. Η ακρίβεια στην κατεύθυνση του υποκείμενου προϊόντος παραμένει δευτέρας τάξης [26]. Το σχήμα Crank-Nicolson είναι αυτό που υλοποιείται στα πλαίσια της παρούσας εργασίας.

Κεφάλαιο 4

Επίλυση τριδιαγώνιων γραμμικών συστημάτων

4.1 Εισαγωγή

Στο προηγούμενο κεφάλαιο έγινε ανάλυση της μ.δ.ε Black-Scholes με τη μέθοδο πεπερασμένων διαφορών. Όπως αναλύθηκε στις ενότητες (3.3.4) και (3.3.5), το πεπλεγμένο και το Crank-Nicolson σχήμα αντίστοιχα, οδηγούν σε ένα τριδιαγώνιο σύστημα γραμμικών εξισώσεων που επίλυσή τους για κάθε χρονικό βήμα, απαιτεί την χρησιμοποίηση μεθόδων της αριθμητικής γραμμικής άλγεβρας.

Τέτοια συστήματα μπορούν να λυθούν απευθείας με αποτελεσματικότερους αλγόριθμους από αυτούς που χρησιμοποιούνται για πιο γενικά γραμμικά συστήματα, όπως η απαλοιφή Gauss [27]. Δύο τέτοιοι αλγόριθμοι είναι η LU-παραγοντοποίηση (ή Τριγωνική παραγοντοποίηση) και ο Cyclic Odd-Even reduction . Στις επόμενες ενότητες θα γίνει παρουσίαση αυτών των δύο μεθόδων.

4.2 Παραγοντοποίηση LU

Η μέθοδος παραγοντοποίησης LU αναφέρεται στο πρόβλημα της διάσπασης ενός πίνακα A στη μορφή

$$A = LU$$

όπου ο L είναι κάτω τριγωνικός και ο U άνω τριγωνικός πίνακας. Αν ο πίνακας A ενός γραμμικού συστήματος $Ax = e$ μπορεί να διασπαστεί με τον τρόπο αυτό, τότε το σύστημα γράφεται $LUx = e$ και όπως είναι φανερό, η επίλυσή του ανάγεται στην επίλυση δυο απλών συστημάτων: του $Ly = e$ ως προς y με εμπρός αντικατάσταση και του $Ux = y$ ως

προς x με πίσω αντικατάσταση.

Θεωρούμε ένα τριδιαγώνιο γραμμικό σύστημα μεγέθους N που έχει την παρακάτω μορφή:

$$\underbrace{\begin{bmatrix} b_0 & c_0 & 0 & \cdots & 0 \\ a_1 & b_1 & c_1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & a_{N-2} & b_{N-2} & c_{N-2} \\ 0 & \cdots & 0 & a_{N-1} & b_{N-1} \end{bmatrix}}_{A} \underbrace{\begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-2} \\ x_{N-1} \end{bmatrix}}_{x} = \underbrace{\begin{bmatrix} e_0 \\ e_1 \\ \vdots \\ e_{N-2} \\ e_{N-1} \end{bmatrix}}_{e}$$

Η πρώτη φάση της παραγοντοποίησης LU είναι η διάσπαση του A στους πίνακες L και U.

$$\underbrace{\begin{bmatrix} b_0 & c_0 & 0 & \cdots & 0 \\ a_1 & b_1 & c_1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & a_{N-2} & b_{N-2} & c_{N-2} \\ 0 & \cdots & 0 & a_{N-1} & b_{N-1} \end{bmatrix}}_{A} = \underbrace{\begin{bmatrix} 1 & 0 & \cdots & 0 \\ a_1' & 1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & a_{N-1}' & 1 \end{bmatrix}}_{L} \cdot \underbrace{\begin{bmatrix} b_0' & c_o & \cdots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 0 & \cdots & b_{N-2}' & c_{N-2} \\ 0 & \cdots & 0 & b_{N-1}' \end{bmatrix}}_{U}$$

Το επόμενο βήμα της μεθόδου είναι να βρεθεί η λύση του συστήματος $Ly = e$ ως προς y με αντικατάσταση πρός τα εμπρός και τελευταίο βήμα είναι να βρεθεί η λύση του συστήματος $Ux = y$ ως προς x με αντικατάσταση προς τα πίσω, όπως φαίνεται και στον αλγόριθμο (1).

Αλγόριθμος 1 LU-παραγοντοποίηση

$b_0' = b_0$
$e_0' = e_0$
for $i = 1$ **to** $N - 2$ **do**
    { *Διάσπαση* }
    $a_i' = a_i / b_{i-1}'$
    $b_i' = b_i - a_i' \cdot c_{i-1}$
    { *Εμπρός αντικατάσταση* }
    $y_i = e_i - a_i' \cdot y_{i-1}$
end for
{ *Πίσω αντικατάσταση* }
$x_{N-1} = e_{N-1} / b_{N-1}'$
for $i = N - 2$ **to** 0 **do**
    $x_i = (y_i - c_i \cdot x_{i+1}) / b_i'$
end for

Η μέθοδος αυτή είναι αποδοτική στη περίπτωση πολλών συστημάτων με κοινό A και διαφορετικά δεύτερα μέλη e, οπότε η παραγοντοποίηση γίνεται μια μόνο φορά. Βάση για την ανάπτυξη της μεθόδου διάσπασης LU είναι η μέθοδος Gauss . Ο υπολογιστικός

χρόνος για τη τριδιαγώνια LU παραγοντοποίηση είναι γραμμικός σε σχέση με το μέγεθος του συστήματος. Όμως αυτός ο αλγόριθμος, αν και είναι γρήγορος και απλός, δεν μπορεί να παραλληλοποιηθεί χωρίς εκτενείς μετασχηματισμούς, [28] αφού υπάρχουν αλληλεξαρτήσεις δεδομένων σε κάθε επανάληψη του βρόγχου (1) τόσο κατά την προς τα εμπρός αντικατάσταση, όσο και στην προς τα πίσω αντικατάσταση. Μόνο οι δύο τελευταίες πράξεις στην ίδια επανάληψη του βρόγχου μπορούν να παραλληλοποιηθούν. Έτσι, αυτός ο αλγόριθμος δεν μπορεί να κλιμακωθεί σε μαζικά παράλληλα συστήματα.

Ένας άλλος αλγόριθμος για την επίλυση τριδιαγώνιων συστημάτων είναι ο Cyclic Odd-Even reduction, που μαζί με μια παραλλαγή του, περιγράφεται αναλυτικά στην επόμενη ενότητα.

4.3 Cyclic Odd-Even reduction

Η κυκλική μείωση (Cyclic reduction) είναι μια οικόγενεια μεθόδων για την επίλυση τριδιαγώνιων συστημάτων που επινοήθηκε από τους G.H. Golub και R. W. Hockney. Η βασική ιδέα πίσω από τις μεθόδους αυτές είναι ο μετασχηματισμός ενός συστήματος γραμμικών εξισώσεων σε δύο (ή περισσότερα) συστήματα γραμμικών εξισώσεων που είναι ανεξάρτητα μεταξύ τους, ώστε να μπορούν να λύνονται ανεξάρτητα το ένα από το άλλο.

Ο αλγόριθμος Odd-Even Cyclic reduction είναι μία επαναληπτική τεχνική για την επίλυση τριδιαγώνιων συστημάτων. Ο αλγόριθμος χωρίζεται σε δύο φάσεις, αρχικά απαλοίφει επαναληπτικά τους μισούς από τους αγνώστους, δηλαδή τα ζυγά στοιχεία του πίνακα και δημιουργεί ένα καινούργιο σύστημα μισού μεγέθους μόνο με τους μονούς αγνώστους, μέχρι να φτάσει σε σύστημα μεγέθους ένα, όπου μπορεί εύκολα να υπολογιστεί η άγνωστη μεταβλητή. Στη συνέχεια υπολογίζονται πηγαίνοντας προς τα πίσω τα ζυγά στοιχεία του πίνακα, που είχαν απαλοιφθεί, χρησιμοποιώντας τα ήδη υπολογισμένα μονά στοιχεία. Αυτές οι δύο φάσεις ονομάζονται «μείωση προς τα εμπρός» (forward elimination) και «αντικατάσταση προς τα πίσω» (backward substitution) αντίστοιχα.

Έστω $Ax = e$ γραμμικό σύστημα, όπου A τριδιαγώνιος πίνακας μεγέθους NxN, e και x διανύσματα μεγέθους N.

$$\begin{bmatrix} b_0 & c_0 & 0 & \cdots & 0 \\ a_1 & b_1 & c_1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & a_{N-2} & b_{N-2} & c_{N-2} \\ 0 & \cdots & 0 & a_{N-1} & b_{N-1} \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-2} \\ x_{N-1} \end{bmatrix} = \begin{bmatrix} e_0 \\ e_1 \\ \vdots \\ e_{N-2} \\ e_{N-1} \end{bmatrix}$$

Σε κάθε βήμα του αλγόριθμου, κατά τη μείωση προς τα εμπρός, απαλείφονται οι μισοί άγνωστοι (τις ζυγές γραμμές) και παράγεται ένας καινούριος τριδιαγώνιος πίνακας, με

μέγεθος το μισό από τον αντίστοιχο του προηγούμενου βήματος. Η διαδικασία επαναλαμβάνεται μέχρι το σύστημά μας να γίνει μεγέθους ένα.

$$\begin{bmatrix} b_0' & c_0' & 0 & \cdots & 0 \\ a_2' & b_2' & c_2' & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & a_{N-4}' & b_{N-4}' & c_{N-4}' \\ 0 & \cdots & 0 & a_{N-2}' & b_{N-2}' \end{bmatrix} \begin{bmatrix} x_0 \\ x_2 \\ \vdots \\ x_{N-4} \\ x_{N-2} \end{bmatrix} = \begin{bmatrix} e_0' \\ e_2' \\ \vdots \\ e_{N-4}' \\ e_{N-2}' \end{bmatrix}$$

$$\Rightarrow$$

$$\vdots$$

$$\Rightarrow$$

$$b_0 \cdot x_0 = e_0 \Rightarrow x_0 = \frac{e_0}{b_0}$$

Όταν το σύστημα γίνει ένα, καταλήγει σε μια εξίσωση της μορφής $b_0 \cdot e_0 = x_0$ όπου υπολογίζεται το x_0 άμεσα, αφού τα b_0 και e_0 είναι γνωστά.

Στη συνέχεια, κατά την αντικατάσταση προς τα πίσω, υπολογίζονται οι άγνωστοι του διανύσματος x. Έχοντας γνωστό το x_0 από το τελευταίο βήμα του forward elimination, υπολογίζονται τα ζυγά στοιχεία του διανύσματος για το συγκεκριμένο σύστημα μεγέθους N_k. Σε κάθε επόμενο βήμα k του αλγόριθμου, χρησιμοποιούνται τα x του προηγούμενου βήματος για να εξάχθούν νέα x για τις ζυγές γραμμές του διανύσματος. Η διαδικασία επαναλαμβάνεται μέχρι να φτάσουμε στο τελευταίο βήμα όπου υπολογίζονται όλα τα x και ο αλγόριθμος ολοκληρώνεται.

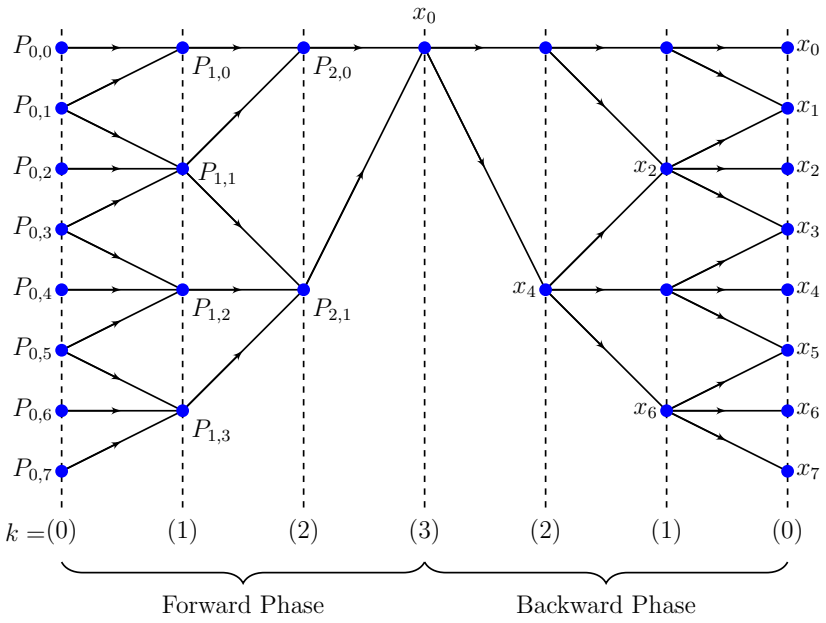
Γενικά ο αλγόριθμος για ένα σύστημα μεγέθους N θα έχει ολοκληρώσει την μείωση, προς τα μπροστά, του συστήματος μετά από k βήματα:

$$k = log_2 N$$

Και το αρχικό σύστημα μειώνεται για κάθε k βήμα σε μέγεθος:

$$N_k = N/2^k$$

Η όλη διαδικασία φαίνεται στο Σχήμα (4.1), για σύστημα μεγέθους $N = 8$, όπου $P_k = (a_k, b_k, c_k, e_k)$ είναι το διάνυσμα των συντελεστών και του δεξιού μέλους του συστήματος (3.33) για το k βήμα.



Σχήμα 4.1: Ροή υπολογισμών του αλγόριθμου Odd-Even Cyclic reduction για σύστημα μεγέθους $N = 8$.

Στις επόμενες ενότητες περιγράφονται δύο εκδοχές του Odd-Even Cyclic reduction, η κλασσική μέθοδος και μια παραλλαγή του αλγορίθμου κατά την οποία η κύρια διαγώνιος του τριδιαγώνιου πίνακα είναι αρχικά ένα και σε κάθε βήμα του forward step γίνεται ένα.

4.3.1 Ο αλγόριθμος Odd-Even reduction

Ο κλασσικός Odd-Even reduction εξαλείφει τους αγνώστους στις ζυγές γραμμές του πίνακα του κάθε βήματος και παράγει τους συντελεστές του καινούργιου μειωμένου συστήματος όπως φαίνεται απο τις πράξεις στον αλγόριθμο (2). **Οι συντελεστές που βρίσκονται εκτός ορίων για την πρώτη και τελευταία επανάληψη θεωρούνται ίσοι με μηδέν:** $a_{-1} = c_{-1} = e_{-1} = 0, a_{N_k} = c_{N_k} = e_{N_k} = 0$.

Αλγόριθμος 2 Forward Phase step of traditional Cyclic Odd- Even Reduction

for $i = 1$ **to** $N_k - 1$ **do**

$$a_{k+1,i/2} = -a_{k,i-1}a_{k,i}b_{k,i+1}$$
$$b_{k+1,i/2} = b_{k,i-1}b_{k,i}b_{k,i+1} - c_{k,i-1}a_{k,i}b_{k,i+1} - b_{k,i-1}c_{k,i}c_{k,i+1}$$
$$c_{k+1,i/2} = -b_{k,i-1}c_{k,i}c_{k,i+1}$$
$$e_{k+1,i/2} = b_{k,i-1}e_{k,i}b_{k,i+1} - e_{k,i-1}a_{k,i}b_{k,i+1} - b_{k,i-1}c_{k,i}e_{k,i+1}$$
$$i \leftarrow i + 2$$

end for

Η φάση της «αντικατάστασης προς τα πίσω» (backward substitution) του αλγορίθμου, κατα την οποία υπολογίζει τους αγνώστους, δίνεται στον αλγόριθμο (3).

Αλγόριθμος 3 Backward Phase step of traditional Cyclic Odd- Even Reduction

for $i = 1$ **to** $N_k - 1$ **do**
  $x_{i2^k} = \left(e_{k,i} - a_{k,i}x_{(i-1)2^k} - c_{k,i}x_{(i+1)2^k}\right)/b_{k,i}$
  $i \leftarrow i + 2$
end for

Από τις πράξεις και των δύο φάσεων του Cyclic Odd-Even Reduction, γίνεται φανερό ότι μέσα σε μία επανάληψη του βρόγχου οι πράξεις είναι ανεξάρτητες μεταξύ τους και μπορούν να εκτελούνται παράλληλα. Η μόνη εξάρτηση δεδομένων βρίσκεται μεταξύ των επαναληπτικών βημάτων k του αλγόριθμου, δηλαδή για να υπολογιστεί νέος, μειωμένος κατά το ήμισυ πίνακας, χρειάζονται τα δεδομένα απ΄τον πίνακα του προηγούμενου βήματος.

Ο αλγόριθμος αυτός έχει ένα βασικό μειονέκτημα: το αποτέλεσμα της εξίσωσης για τον υπολογισμό του $b_{k+1,i}$ μεγαλώνει εκθετικά όσο μεγαλώνει το βήμα k του αλγόριθμου. Έτσι, υπάρχει περίπτωση - για αριθμητική πεπερασμένης ακρίβειας - να μην μπορεί να αναπαρασταθεί η μεταβλητή b_i (δηλαδή η κύρια διαγωνίος) κι ως εκ τούτου, το σύστημα να μην είναι σταθερό [25]. Το ζήτημα αυτό λύνεται με μία παραλλαγή της μεθόδου που κανονικοποιεί την κύρια διαγώνιο σε κάθε βήμα [29] και περιγράφεται στην επόμενη υποενότητα.

4.3.2 Η παραλλαγή του Odd-Even reduction

Μια παραλλαγή του κλασικού odd-even cyclic reduction που αποφεύγει την υπερχείληση της μεταβλητής b_i, είναι αυτή κατά την οποία η κύρια διαγώνιος του πίνακα είναι αρχικά ένα και κανονικοποιείται σε κάθε βήμα της «μείωσης προς τα εμπρός» (forward elimination). Η κανονικοποίηση γίνεται διαιρώντας και τα δύο μέλη των εξισώσεων με την υπάρχουσα κύρια διαγώνιο, γεγονός που δε μειώνει τη γενικότητα της μεθόδου, αφού με αυτό τον τρόπο παράγεται το σύστημα με τις επιθυμητές ιδιότητες.

$$\begin{bmatrix} 1 & c_0 & 0 & \cdots & 0 \\ a_1 & 1 & c_1 & \cdots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \cdots & a_{N-2} & 1 & c_{N-2} \\ 0 & \cdots & 0 & a_{N-1} & 1 \end{bmatrix} \begin{bmatrix} x_0 \\ x_1 \\ \vdots \\ x_{N-2} \\ x_{N-1} \end{bmatrix} = \begin{bmatrix} e_0 \\ e_1 \\ \vdots \\ e_{N-2} \\ e_{N-1} \end{bmatrix}$$

Η διαδικασία συνοψίζεται στον αλγόριθμο (4), ο οποίος προήλθε από τον αλγόριθμο (2) αντικαθιστώντας με $b_{0,i} = 1$. Μετά, διαιρώντας όλους τους συντελεστές με την ποσότητα $1 - c_{0,i-1} \cdot a_{0,i} - c_{0,i} \cdot a_{0,i+1}$ κανονικοποιείται η κύρια διαγώνιος και για κάθε k γίνεται η ίδια διαδικασία ($b_{k+1,i/2} = 1$). Ο αλγόριθμος (4) παρουσιάζεται για ένα βήμα k.

Αλγόριθμος 4 Forward Phase step of the Cyclic Odd-Even Reduction Variant

for $i = 0$ **to** $N_k - 1$ **do**
$temp = 1/(1 - c_{k,i-1} \cdot a_{k,i} - c_{k,i} \cdot a_{k,i+1})$
$a_{k+1,i/2} = -temp \cdot a_{k,i-1} \cdot a_{k,i}$
$c_{k+1,i/2} = -temp \cdot c_{k,i} \cdot c_{k,i+1}$
$e_{k+1,i/2} = temp \cdot (e_{k,i} - e_{k,i-1} \cdot a_{k,i} - c_{k,i} \cdot e_{k,i+1})$
$i \leftarrow i + 2$
end for

Η φάση της «αντικατάστασης προς τα πίσω» (backward substitution) της εφαρμοζόμενης παραλλαγής φαίνεται στον αλγόριθμο (5), ο οποίος είναι ίδιος με τον αλγόριθμο (3) αντικαθιστώντας το $b_{k,i}$ με μονάδα και το $x_0 = e_{log_2 N,0}$ αρχικά.

Αλγόριθμος 5 Backward Phase of the Cyclic Odd- Even Reduction variant

for $i = 1$ **to** $N_k - 1$ **do**
$x_{i2^k} = e_{k,i} - a_{k,i} \cdot x_{(i-1)2^k} - c_{k,i} \cdot x_{(i+1)2^k}$
$i \leftarrow i + 2$
end for

Στην επόμενη ενότητα γίνεται σύγκριση των αλγορίθμων που εξετάστηκαν έως τώρα από πλευράς υπολογιστικού κόστους, ώστε στη συνέχεια να επιλεγεί ο αλγόριθμος που θα υλοποιηθεί στη φάση της σχεδίασης της αρχιτεκτονικής.

4.4 Σύγκριση των μεθόδων και Πολυπλοκότητα

Ο αριθμός των πράξεων των αλγορίθμων, όπως παρουσιάστηκαν στην προηγούμενη ενότητα, **για μία επανάληψη του βρόχου**, ανά φάση, φαίνεται στον πίνακα (4.1)

	Forward loop pass			**Backward loop pass**		
Operation type	# mul	# add/ # sub	# div	# mul	# add/ # sub	# div
LU Decomp.	2	2	1	1	1	1
Traditional Odd-even Red.	16	4	0	2	2	1
Variant Odd-even Red.	9	4	1	2	2	0

Πίνακας 4.1: Αριθμός των πράξεων για ένα Forward/Backward loop pass των αλγορίθμων

Ο συνολικός αριθμός των πράξεων υπολογίζεται πολλαπλασιάζοντας τον αριθμό των πράξεων μιας εκτέλεσης του loop με το συνολικό αριθμό των εκτελέσεων του loop, για όλες τις φάσεις. Για την LU-διάσπαση, ο συνολικός αριθμός των εκτελέσεων του loop

είναι $N-1$, τόσο στην προς τα εμπρός φάση όσο και στην προς τα πίσω. Το ίδιο ισχύει και για τον Cyclic Odd-Even reduction και για την παραλλαγή του. Ο συνολικός αριθμός πράξεων, φαίνεται στον πίνακα (4.2).

Operation type	# mul	# add/ # sub	# div
LU Decomp.	$2(N-1)$	$3(N-1)$	$2(N-1)$
Traditional Odd-even Red.	$18(N-1)$	$6(N-1)$	$(N-1)$
Variant Odd-even Red.	$11(N-1)$	$6(N-1)$	$(N-1)$

Πίνακας 4.2: Συνολικός αριθμός των πράξεων των αλγορίθμων που εξετάζονται

Είναι φανερό ότι LU-διάσπαση απαιτεί το μικρότερο αριθμό των πράξεων στο σύνολο, ακολουθούμενη από την παραλλαγή του Cyclic Odd-Even reduction. Όμως, όταν το σύστημα έχει c πυρήνες (cores), η πολυπλοκότητα στο πεδίο του χρόνου για την LU-διάσπαση είναι $O(N)$, αφού δεν μπορεί να παραλληλοποιηθεί, ενώ του Odd-Even reduction γίνεται $O(N/c+c)$. Ο odd-even reduction μπορεί να παραλληλοποιηθεί, γιατί οι πράξεις στον ίδιο βρόχο είναι ανεξάρτητες μεταξύ τους.

Η πολυπλοκότητα στο πεδίο του χώρου και για τις τρεις μεθόδους είναι γραμμική σε σχέση με το μέγεθος του συστήματος: η LU-διάσπαση χρειάζεται μνήμη $5N$ θέσεων για τις τρεις διαγωνίους (a, b, c), το δεξί μέλος (e) και τη λύση του συστήματος (x). Ο παραδοσιακός Odd-Even reduction χρειάζεται επιπλέον $4(N-1)$ μνήμη για τις καινούργιες διαγώνιους και το δεξί μέλος των πινάκων που δημιουργούνται κατά τη διάρκεια της προς τα εμπρός φάσης. Η παραλλαγή του Odd-Even reduction χρειάζεται $4(N)$ μνήμη αρχικά για τα a, c, e, x (η κύρια διαγώνιος b είναι 1 και δε χρειάζεται να αποθηκευτεί) και $3(N-1)$ μνήμη παραπάνω για τα παραγόμενα a, c, e της προς τα εμπρός φάσης [30].

Η διάσπαση LU είναι το ίδιο σταθερή με τη Gaussian απαλοιφή χωρίς οδήγηση. Ο odd-even reduction μπορεί να αποδειχθεί ότι έχει ισοδύναμη ακρίβεια με τη Gaussian απαλοιφή χωρίς οδήγηση, σε ένα μετασχηματισμένο σύστημα [31].

Από την παραπάνω ανάλυση γίνεται κατανοητό ότι η παραλλαγή του αλγορίθμου Cyclic Odd-Even reduction είναι η καταλληλότερη για υλοποίηση σε αναδιατασσόμενη λογική (FPGA) και εφαρμόζεται σε συστήματα που το μέγεθός τους είναι δύναμη του δύο. Το ίδιο ισχύει και για τον αριθμό των cores των αρχιτεκτονικων που υλοποιήθηκαν.

Κεφάλαιο 5

Μοντελοποίηση και Αρχιτεκτονικές

5.1 Εισαγωγή

Στα προηγούμενα κεφάλαια αναλύθηκε η θεωρία πίσω από τα χρηματοοικονομικά παράγωγα και το μαθηματικό υπόβαθρο για τα μοντέλα αποτίμησης χρηματοοικονομικών παραγώγων. Στη συνέχεια παρουσιάστηκε μια μεθοδολογία για την αριθμητική επίλυση της εξίσωσης Black-Scholes μέσω των Πεπερασμένων Διαφορών - για αποτίμηση Δικαίωματος Αγοράς (Call Option Pricing) - όπου με τη χρήση του σχήματος Crank-Nicolson εξάγεται ένα τριδιαγώνιο σύστημα γραμμικών εξισώσεων (3.32). Έπειτα, παρουσιάστηκαν αλγόριθμοι από το πεδίο της αριθμητικής άλγεβρας για την επίλυση του σχήματος πεπερασμένων διαφορών Crank-Nicolson.

Σε αυτό το κεφάλαιο, γίνεται μοντελοποίηση της μεθόδου Crank–Nicolson και της παραλλαγής του αλγόριθμου Cyclic odd-even reduction, διαχωρίζοντας τα βασικά στάδια από τα οποία αποτελούνται, με σκοπό τον πιο εύκολο σχεδιασμό της αρχιτεκτονικής.

Επίσης, παρουσιάζονται οι αρχιτεκτονικές που υλοποιήθηκαν.

5.2 Μοντελοποίηση του αλγόριθμου

Η επίλυση του σχήματος Πεπερασμένων Διαφορών Crank–Nicolson χωρίζεται σε δύο μέρη:

- Στην επίλυση του τριδιαγώνιου γραμμικού συστήματος 3.32:

$$a_i \cdot x_{i-1}^{n+1} + b_i \cdot x_i^{n+1} + c_i \cdot x_{i+1}^{n+1} = e_i^n$$

- Και στην ανανέωση του δεξιού μέλους (Update Right Hand Side) e της παραπάνω εξίσωσης μέσω της σχέσης 3.33:

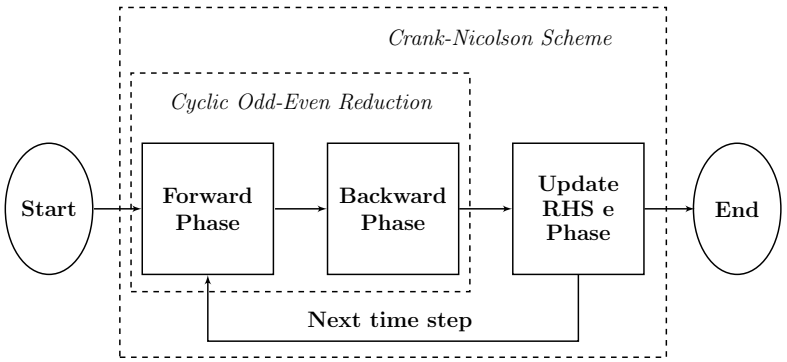
$$e_i^n = d_{0,i} \cdot x_{i-1}^n + d_{1,i} \cdot x_i^n + d_{2,i} \cdot x_{i+1}^n$$

Αυτές οι δύο διαδικασίες πρέπει να εκτελούνται **για κάθε χρονικό βήμα** n.

Η επίλυση του τριδιαγώνιου γραμμικού συστήματος γίνεται με μία παραλλαγή του κλασσικού αλγόριθμου Cyclic odd-even reduction (Ενότητα 4.3.2) που έχει την κύρια διαγώνιο $b_i = 1$. Ο αλγόριθμος με τη σειρά του, χωρίζεται και αυτός σε δύο φάσεις:

- τη «μείωση προς τα εμπρός» (forward elimination) ή φάση προς τα εμπρός (Forward Phase)(Αλγόριθμος 4),
- και την «αντικατάσταση προς τα πίσω» (backward substitution) ή φάση προς τα πίσω (Backward Phase)(Αλγόριθμος 5).

Συνολικά, η διαδικασία για την αριθμητική επίλυση της εξίσωσης Black-Scholes φαίνεται στο σχήμα (5.1).



Σχήμα 5.1: Διάγραμμα ροής για το σχήμα Crank–Nicolson.

Στις υποενότητες που ακολουθούν, αναλύονται ξεχωριστά οι τρεις φάσεις και γίνεται μοντελοποίηση των πράξεων τους.

5.2.1 Η φάση προς τα εμπρός (Forward Phase)

Στη Forward Phase αναλυτικά τα βήματα του αλγορίθμου είναι τα εξής:

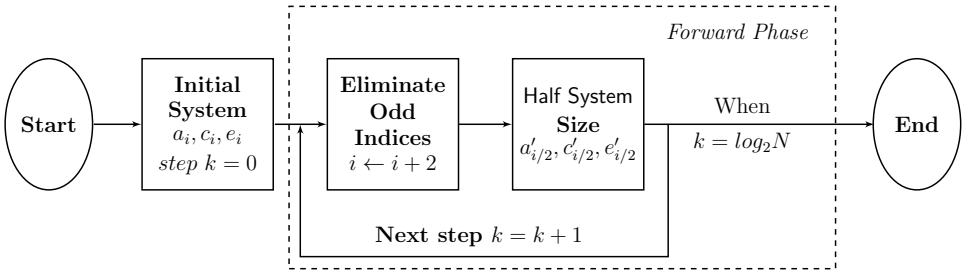
0. Διαβάζονται τα στοιχεία $a_{0,i}$,$c_{0,i}$ και $e_{0,i}$ για κάθε i από 0 έως $N-1$ ενός αρχικού συστήματος μεγέθους N, βήμα αλγόριθμου $k = 0$.

1. Υπολογίζεται ένα καινούργιο σύστημα, μεγέθους $N/2$, εξαλείφοντας τις ζυγές γραμμές του πίνακα μέσω των πράξεων του Αλγόριθμου (4), βήμα $k = 1$.

2. Διαβάζονται τα στοιχεία $a_{k,i}$,$c_{k,i}$ και $e_{k,i}$ του μειωμένου συστήματος και υπολογίζεται νέο, μεγέθους $N_k/2$, βήμα $k = k + 1$.

3. Επαναλαμβάνεται το **2** μέχρι το σύστημα να γίνει μεγέθους, $N_k = 1$, με βήμα $k = log_2 N$. Η έξοδος της Forward Phase είναι όλα τα μειωμένα συστήματα.

Τα βήματα της Forward Phase φαίνονται στο σχήμα (5.2).

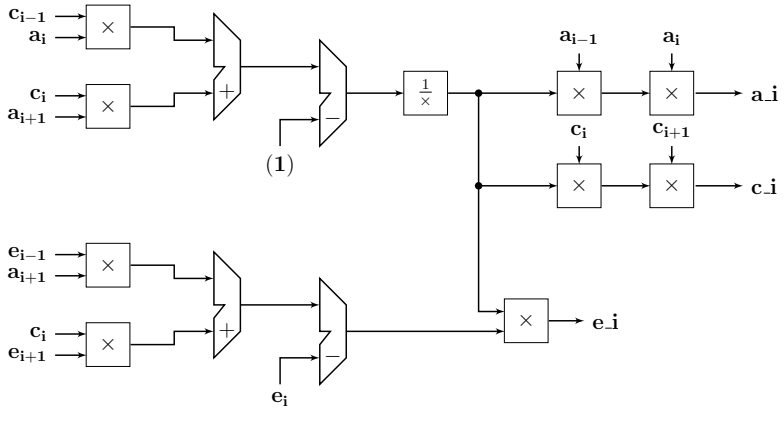


Σχήμα 5.2: Διάγραμμα ροής για τη Forward Phase του odd-even reduction.

Οι πράξεις που εκτελούνται για τη διαδικασία της μείωσης, όπως αναλύθηκε παραπάνω, για ένα loop pass του Αλγόριθμου (4), είναι οι εξής:

$$
\begin{aligned}
temp &= 1/(1 - c_{i-1} \cdot a_i - c_i \cdot a_{i+1}) \Rightarrow -temp = 1/-(c_{i-1} \cdot a_i + c_i \cdot a_{i+1} - 1) \\
a_{k+1,i/2} &= -temp \cdot a_{i-1} \cdot a_i \\
b_{k+1,i/2} &= 1 \\
c_{k+1,i/2} &= -temp \cdot c_i \cdot c_{i+1} \\
e_{k+1,i/2} &= temp \cdot (e_i - e_{i-1} \cdot a_i - c_i \cdot e_{i+1}) = -temp(e_{i-1} \cdot a_i + c_i \cdot e_{i+1} - e_i)
\end{aligned} \tag{5.1}
$$

Οι πράξεις ήρθαν στην παραπάνω μορφή, ώστε να διευκολυνθεί η μοντελοποίησή τους. Όπως φαίνεται και στο σχήμα (5.3), η αλληλουχία πράξεων 2 (ταυτόχρονοι) πολλαπλασιασμοί → πρόσθεση → αφαίρεση, επαναλαμβάνεται 2 φορές, μία για τον υπολογισμό της μεταβλητής $-temp$ και μία για τον υπολογισμό της e_i.



Σχήμα 5.3: Μοντελοποίηση των πράξεων της Forward Phase.

5.2.2 Η φάση προς τα πίσω (Backward Phase)

Η Backward Phase ξεκινάει μόλις τελειώσει η Forward Phase. Αναλυτικά, τα βήματα της φάσης αυτής είναι τα εξής:

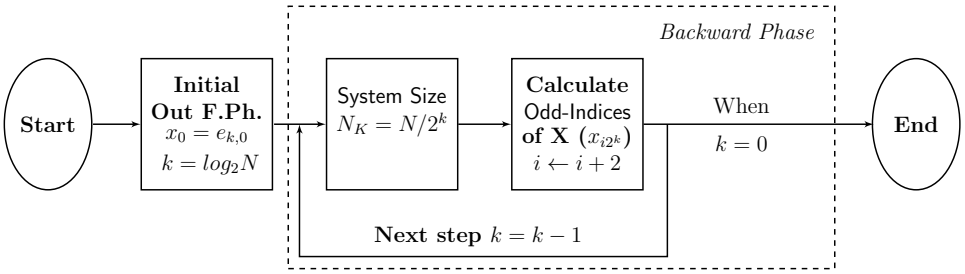
0. Υπολογίζεται το $x_0 = e_{k,0}$, αρχικό βήμα $k = log_2 N$.

1. Διαβάζονται τα μονά στοιχεία $a_{k,i}$,$c_{k,i}$ και $e_{k,i}$, και x_0 και υπολογίζεται το $x_{N/2}$, βήμα $k = (log_2 N) - 1$.

2. Με τα υπολογισμένα x των προηγούμενων βημάτων και τα μονά στοιχεία αυτού του βήματος $a_{k,i}$,$c_{k,i}$ και $e_{k,i}$, για κάθε i από 1 έως $N_k - 1$, υπολογίζονται τα νέα x, βήμα $k = k - 1$.

3. Επαναλαμβάνεται το **2** μέχρι το βήμα του αλγορίθμου να γίνει $k = 0$, όπου έχουν υπολογιστεί όλα τα στοιχεία του διανύσματος X, μεγέθους N.

Τα βήματα της Backward Phase φαίνονται στο σχήμα (5.4).

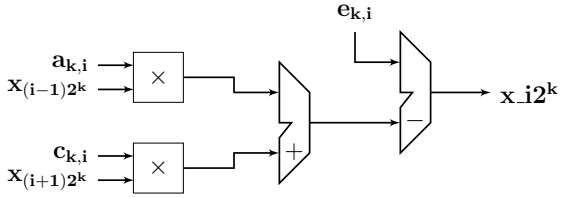


Σχήμα 5.4: Διάγραμμα ροής για τη Backward Phase του odd-even reduction.

Οι πράξεις που εκτελούνται για τη διαδικασία της αντικατάστασης, όπως αναλύθηκε παραπάνω, για ένα loop pass του Αλγόριθμου (5) είναι οι εξής:

$$x_{i2^k} = e_{k,i} - a_{k,i} \cdot x_{(i-1)2^k} - c_{k,i} \cdot x_{(i+1)2^k} = e_{k,i} - (a_{k,i} \cdot x_{(i-1)2^k} + c_{k,i} \cdot x_{(i+1)2^k})$$

Για κάθε στοιχείο x_{i2^k} που υπολογίζεται, χρειάζονται 2 πολλαπλασιασμοί, 1 πρόσθεση και 1 αφαίρεση με την ίδια σειρά που γίνονται και στη Forward Phase. Όπως φαίνεται και στο σχήμα (5.5) υπολογίζεται πρώτα η πράξη της παρένθεσης και μετά αφαιρείται από τη μεταβλητή $e_{k,i}$.



Σχήμα 5.5: Μοντελοποίηση των πράξεων της Backward Phase.

5.2.3 Η φάση ανανέωσης του δεξιού μέλους (Update RHS e Phase)

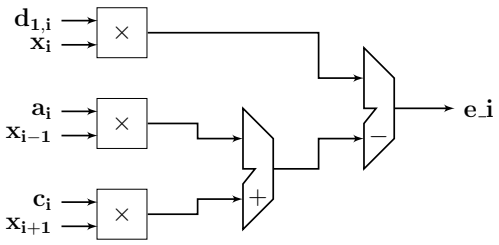
Η Update RHS e ξεκινάει μόλις τελειώσει η διαδικασία επίλυσης του τριδιαγώνιου συστήματος, μέσω της παραλλαγής του αλγόριθμου Cyclic Odd-Even Reduction που περιγράφηκε στις προηγούμενες υποενότητες.

Μόλις η Backward Phase εξάγει όλα τα x, ανανεώνεται το δεξί μέλος του συστήματος (3.32) μέσω των πράξεων της εξίσωσης (3.33).

$$e_i = -a_i x_{i-1} + d_{1,i} x_i - c_i x_{i+1} = d_{1,i} x_i - (a_i x_{i-1} + c_i x_{i+1})$$

Στη συνέχεια, τα a, c και το ανανεωμένο e σχηματίζουν ένα καινούργιο τριδιαγώνιο σύστημα (*σε επόμενο χρονικό βήμα n του το σχήματος Crank-Nicolson*), που μπαίνει ως είσοδος στη Forward Phase (σχήμα 5.1).

Για κάθε ανανεωμένο e_i που υπολογίζεται, χρειάζονται 3 πολλαπλασιασμοί, 1 πρόσθεση και 1 αφαίρεση με την ίδια σειρά που γίνονται και στη Backward Phase. Όπως φαίνεται και στο σχήμα (5.6) υπολογίζεται πρώτα η πράξη της παρένθεσης και μετά αφαιρείται από την ποσότητα ($d_{1,0} \times x_i$).



Σχήμα 5.6: Μοντελοποίηση των πράξεων της Update RHS e Phase.

5.3 Πρώτη Αρχιτεκτονική

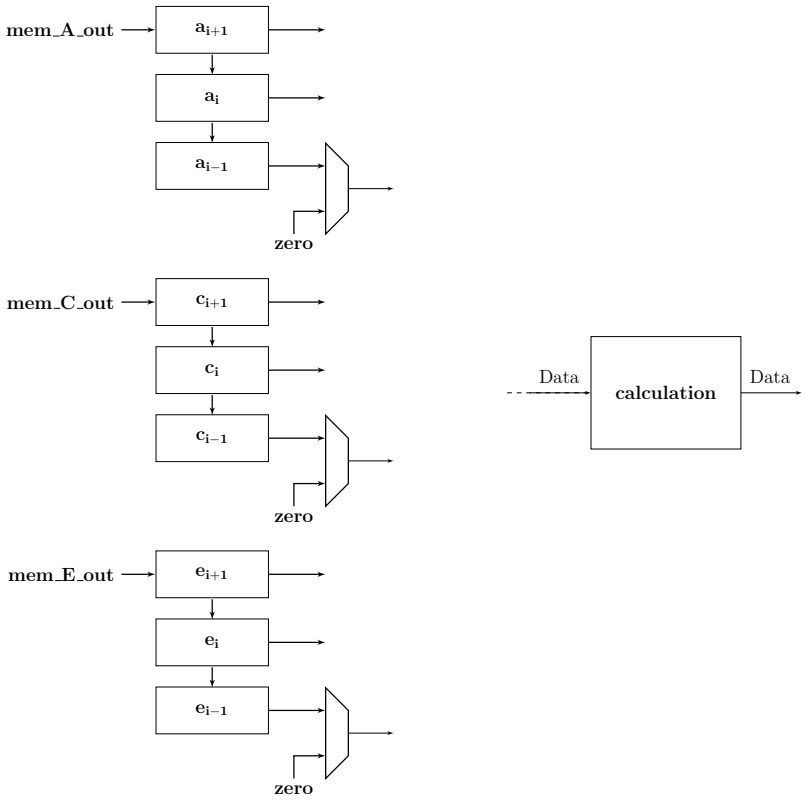
Σε αυτή την ενότητα αναλύεται η πρώτη αρχιτεκτονική που υλοποιήθηκε. Αναλύονται ξεχωριστά οι μονάδες που την απαρτίζουν καθώς και η λειτουργία τους σε κάθε μία από τις τρεις φάσεις που έχουν περιγραφεί στην προηγούμενη ενότητα.

5.3.1 Forward Phase

5.3.1.1 Μνήμη και μονάδα Input Registers

Κάθε στοιχείο του αρχικού τριδιαγώνιου συστήματος αποθηκεύεται σε μία θέση μνήμης. Τα a, c, e μπαίνουν σε 3 ξεχωριστές μνήμες, τις mem A, mem C και mem E αντίστοιχα, στις θέσεις 0 έως $N-1$, όπου N το μέγεθος του συστήματος. Όπως παρουσιάστηκε

στον αλγόριθμο (2), για κάθε νέο στοιχείο a',c' και e' που παράγεται σε μία επανάληψη του loop κατά τη Forward Phase, χρειάζονται 3 στοιχεία από τον υπάρχοντα πίνακα: a, c, e_{i-1} & a, c, e_i & a, c, e_{i+1}. Τα στοιχεία αυτά διαβάζονται σειριακά από τις μνήμες και αποθηκεύονται προσωρινά στη μονάδα Input Registers η διάταξη της οποίας φαίνεται στο σχήμα (5.7).



Σχήμα 5.7: Input Registers της Forward Phase.

Οι έξοδοι της κάθε μνήμης A,C,E συνδέονται μόνο με τις εισόδους των καταχωρητών $i+1$. Οι έξοδοι των $i+1$ καταχωρητών συνδέονται με τη μονάδα Calculation και με τις εισόδους των i καταχωρητών. Με τη σειρά τους οι έξοδοι των i συνδέονται με το Calculation και με τις εισόδους των $i-1$. Τέλος, οι έξοδοι των $i-1$ συνδέονται μόνο με το Calculation.

Όταν ξεκινάει η forward phase, διαβάζεται το στοιχείο της θέσης 0 και γράφεται στον καταχωρητή $i+1$. Στον επόμενο κύκλο, διαβάζεται το στοιχείο της θέσης 1, γράφεται στον $i+1$ και το στοιχείο που υπήρχε εκεί από τον προηγούμενο κύκλο, πηγαίνει στον i. Ως στοιχείο $i-1$ της θέσης -1 θεωρείται η τιμή 0. Τα δεδομένα μπαίνουν στη μονάδα Calculation όπου εκτελούνται οι πράξεις που φαίνονται στο σχήμα (5.3) (Το Calculation αναλύεται παρακάτω).

Στην επόμενη επανάληψη, θα διαβαστούν τα στοιχεία των θέσεων 2 και 3 και θα γραφτούν στους i και $i+1$ καταχωρητές αντίστοιχα, με τον ίδιο τρόπο. Το στοιχείο της θέσης 1 από τον $i+1$ καταλήγει στον $i-1$. Τα δεδομένα είναι έτοιμα για επεξεργασία. Η διαδικασία

συνεχίζεται με τον ίδιο τρόπο.

Κάθε καινούριος (μισός σε μέγεθος) πίνακας που παράγεται σε κάθε βήμα k της Forward Phase, αποθηκεύται στις ίδιες μνήμες με τον αρχικό. Το πρώτο στοιχείο κάθε "νέου" πίνακα, αποθηκεύεται στην επόμενη θέση του τελευταίου στοιχείου του "παλιού" πίνακα.

$$\begin{array}{r} \text{θέση } 0 \\ \\ \\ \\ \text{θέση } N-1 \\ \text{θέση } N \\ \\ \end{array} \begin{bmatrix} 0 \\ a_1 \\ \vdots \\ a_{N-2} \\ a_{N-1} \\ a'_0 \\ a'_1 \\ \vdots \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{N-2} \\ 0 \\ c'_0 \\ c'_1 \\ \vdots \end{bmatrix} \begin{bmatrix} e_0 \\ e_1 \\ \vdots \\ e_{N-2} \\ e_{N-1} \\ e'_0 \\ e'_1 \\ \vdots \end{bmatrix}$$

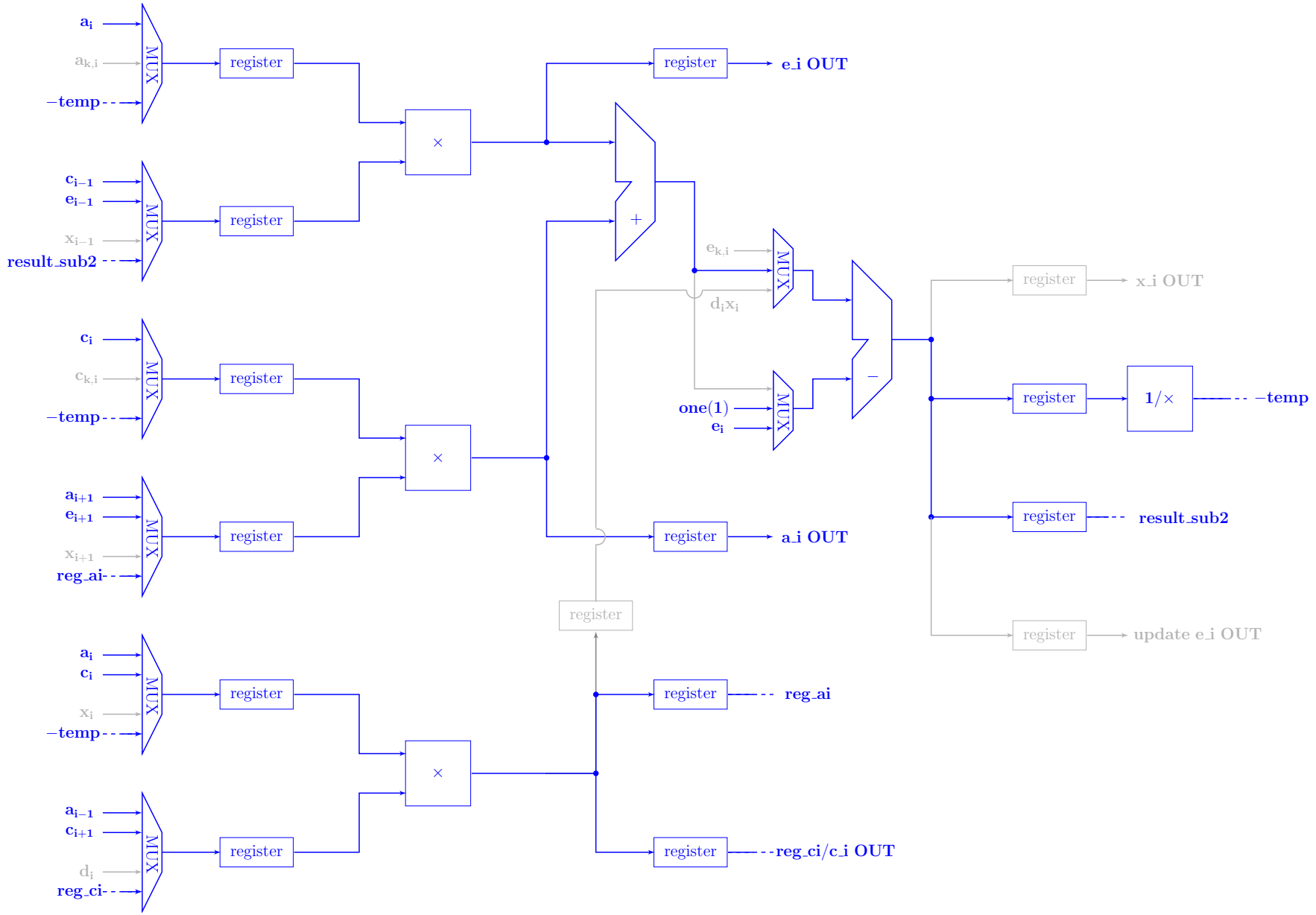
5.3.1.2 Μονάδα Calculation

Στη μονάδα Calculation υλοποιούνται οι πράξεις και των τριών φάσεων του αλγόριθμου (Forward Phase, Backward Phase, Update RHS e). Οι πράξεις είναι ανάμεσα σε αριθμούς κινητής υποδιαστολής μονής ακρίβειας, εύρους 32 bits και γίνονται με τους **floating-point operators (FPO)** που διαθέτει η FPGA σε περιορισμένο αριθμό.

Για τη Forward Phase, η μονάδα Calculation αρχικά, σχεδιάστηκε ακολουθώντας πιστά τη μοντελοποίηση των πράξεων όπως φαίνονται στο σχήμα (5.3). Όμως, σε μια τέτοια σχεδίαση απαιτούνται συνολικά 14 FPO, που χρησιμοποιούνται μόνο μία φορά ο καθένας, καταναλώνοντας άσκοπα τους πόρους της FPGA. Έτσι, θα ήταν αδύνατο να προστεθούν και άλλες μονάδες Calculation στη συνολική αρχιτεκτονική. Επίσης, όλες οι πράξεις εκτελούνται σειριακά γεγονός που οδήγησε σε κύκλωμα με μεγάλο latency.

Σε επόμενο στάδιο, ο στόχος ήταν να μειωθούν οι FPO ανά Calculation και να γίνονται όσο περισσότερες πράξεις ταυτόχρονα. Παρατηρείται ότι η μεταβλητή *temp* πρέπει να υπολογίζεται πρώτη γιατί χρειάζεται για την εξαγωγή και των τριών νέων a, c και e. Έτσι, έγινε επαναδιάταξη των πράξεων σε σχέση με τη μοντελοποίησή τους και κρατήθηκε η αλληλουχία: 2 (παράλληλοι) πολλαπλασιασμοί → πρόσθεση → αφαίρεση → διαίρεση. Χρησιμοποιείται ένας FPO ανά πράξη. Ο μεγάλος αριθμός πολλαπλασιασμών που πρέπει να γίνουν κατά τη Forward Phase, οδήγησε στην προσθήκη ενός επιπλέον πολλαπλασιαστή.

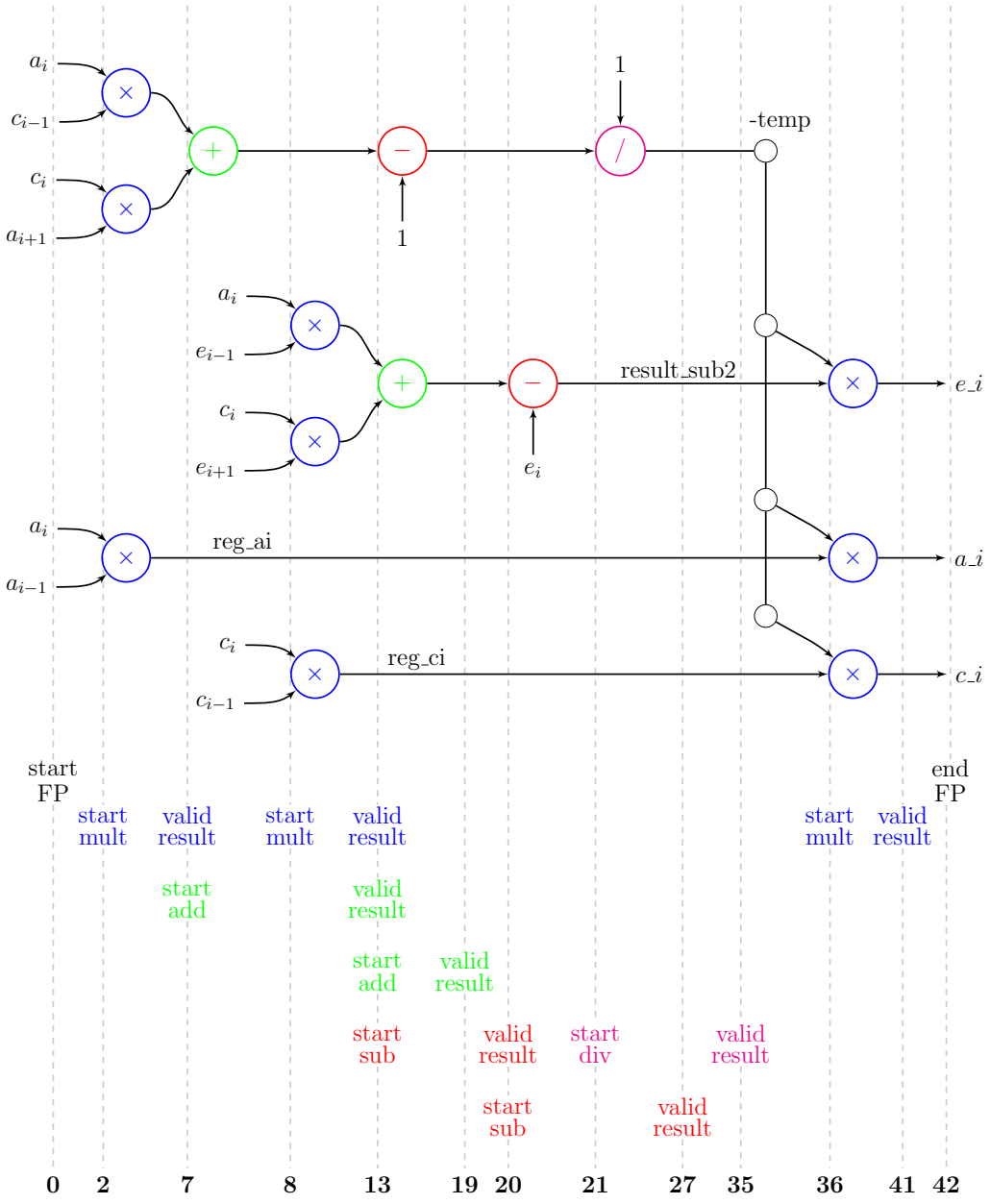
Στο σχήμα (5.8) παρουσιάζεται η μονάδα Calculation και για τις 3 φάσεις (Forward, Backward, Update RHS e). Με μπλε χρώμα τονίζεται το datapath μόνο της Forward Phase.



Σχήμα 5.8: Μονάδα Calculation.

Όπως φαίνεται και στο σχήμα (5.9), υπάρχουν πολυπλέκτες πριν τους πολλαπλασιαστές και πριν τον αφαιρετή, αφού οι συγκεκριμένοι FPO χρησιμοποιούνται 3 και 2 φορές αντίστοιχα. Οι καταχωρητές που υπάρχουν πριν από τους πολλαπλασιαστές και πριν το διαιρέτη, μπήκαν στο τέλος της σχεδίασης, και αφού έγινε Synthesis στο σύστημα, για να μειωθεί το critical path και να αυξηθεί η συχνότητα λειτουργίας του ρολογιού του κυκλώματος. Οι υπόλοιποι καταχωρητές, κρατούν ενδιάμεσες τιμές που θα επαναχρησιμοποιηθούν σε πράξεις για να εξαχθεί το τελικό αποτέλεσμα.

Το σχήμα (5.9) δείχνει την αλληλουχία των πράξεων της Forward Phase στο χρόνο. Απεικονίζονται οι πράξεις (*όχι floating-point operators*) και ο χρόνος μετριέται σε κύκλους.

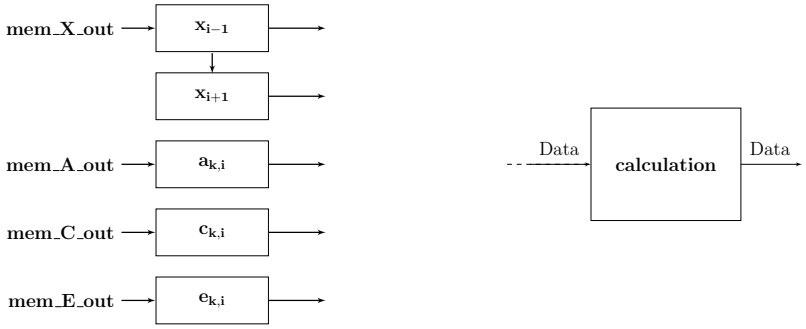


Σχήμα 5.9: Διάγραμμα αλληλουχίας πράξεων της Forward Phase

5.3.2 Backward Phase

5.3.2.1 Μνήμη και μονάδα Input Registers

Η Backward Phase ξεκινάει μόλις τελειώσει η Forward Phase. Όπως φαίνεται και στο σχήμα (4.1), στο βήμα $k = log2N$ υπολογίζεται το στοιχείο e_0 που ισούται με x_0. Σε κάθε επόμενο βήμα $k = k-1$ χρησιμοποιούνται τα x που υπολογίστηκαν στα προηγούμενα βήματα και τα -υπολογισμένα από τη Forward Phase- $a_{k,i}$,$c_{k,i}$,$e_{k,i}$ αυτού του βήματος. Τα στοιχεία αυτά διαβάζονται από τις μνήμες και αποθηκεύονται προσωρινά στη μονάδα Input Registers η διάταξη της οποίας φαίνεται στο σχήμα (5.10).

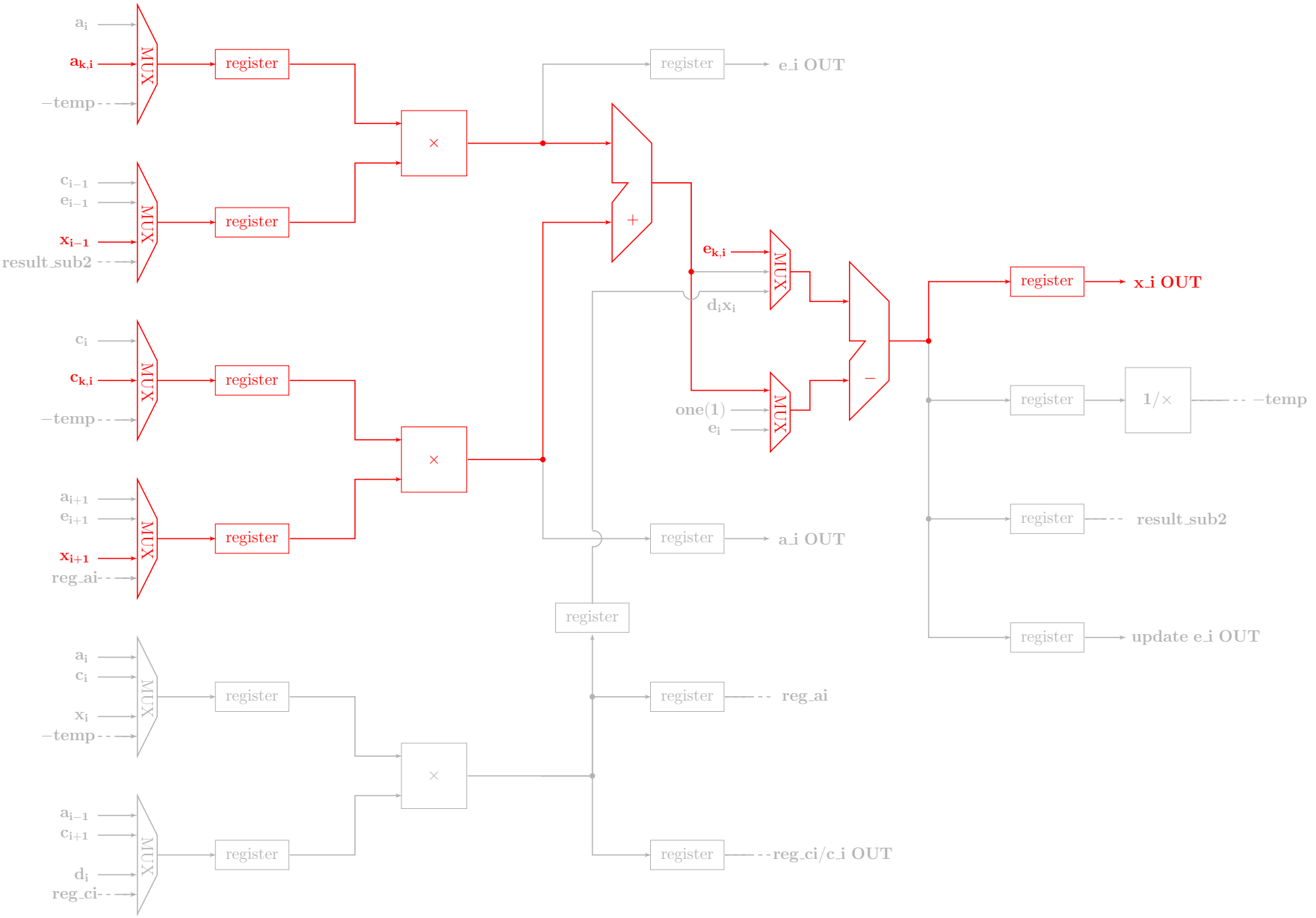


Σχήμα 5.10: Input Registers της Backward Phase.

Κάθε στοιχείο του διανύσματος X αποθηκεύεται σε μία θέση μνήμης, άρα απαιτείται μνήμη N θέσεων. Στο βήμα $k = log_2N - 1$ διαβάζονται τα x_0 και x_N και υπολογίζεται το $x_{N/2}$. Στο επόμενο k, από τα x_0 και $x_{N/2}$ παράγεται το $x_{N/4}$ και από τα $x_{N/2}$ και x_N παράγεται το $x_{3N/4}$. Κάθε καινούριο x που εξάγεται, γράφεται στο μέσον των δύο θέσεων των x που χρησιμοποιήθηκαν για τον υπολογισμό του (σχήμα 4.1). Έτσι, η διεύθυνση εγγραφής στη μνήμη υπολογίζεται με βάση τη διεύθυνση ανάγνωσης.

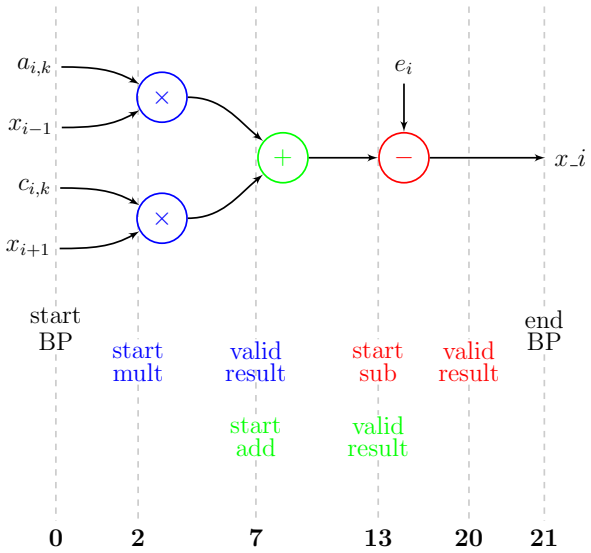
5.3.2.2 Μονάδα Calculation

Για τη Backward Phase, η διάταξη των πράξεων είναι ίδια με αυτή της Forward. Επομένως, δεν έγιναν αλλαγές στη μονάδα Calculation, απλά συνδέθηκαν οι μεταβλητές με τους αντίστοιχους FPO. Στο σχήμα (5.11) παρουσιάζεται η μονάδα Calculation για όλες τις φάσεις. Με κόκκινο χρώμα τονίζεται το datapath μόνο της Backward.



Σχήμα 5.11: Μονάδα Calculation.

Το σχήμα (5.12) δείχνει την αλληλουχία των πράξεων της Backward Phase στο χρόνο. Απεικονίζονται οι πράξεις (*όχι floating-point operators*) και ο χρόνος μετριέται σε κύκλους.



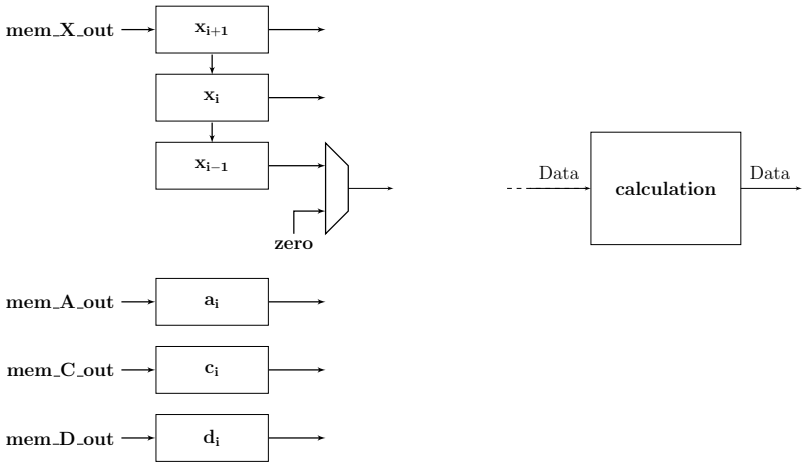
Σχήμα 5.12: Διάγραμμα αλληλουχίας πράξεων της Backward Phase

5.3.3 Update RHS e Phase

5.3.3.1 Μνήμη και μονάδα Input Registers

Η Update RHS e Phase ξεκινάει μετά το τέλος της Backward Phase. Τα στοιχεία a και c του **αρχικού συστήματος**, τα στοιχεία του διανύσματος d και τα υπολογισμένα x εξάγουν το καινούριο διάνυσμα E, που αποθηκεύεται στις θέσεις 0 έως $N-1$ της μνήμης mem E, πανωγράφοντας το διάνυσμα που ήταν αποθηκευμένο εκεί.

Για κάθε νέο στοιχείο e' που παράγεται χρειάζονται 3 στοιχεία από το διάνυσμα X, τα x_{i-1} & x_i & x_{i+1} καθώς και τα a_i & c_i & d_i. Τα στοιχεία αυτά διαβάζονται σειριακά από τις μνήμες και αποθηκεύονται προσωρινά στη μονάδα Input Registers η διάταξη της οποίας φαίνεται στο σχήμα (5.13). Η τριάδα καταχωρητών για τις μεταβλητές x έχει ίδια δομή με τους αντίστοιχους των a, c, e της Forward Phase. Οι καταχωρητές a και c είναι κοινοί με τους αντίστοιχους της Backward Phase.



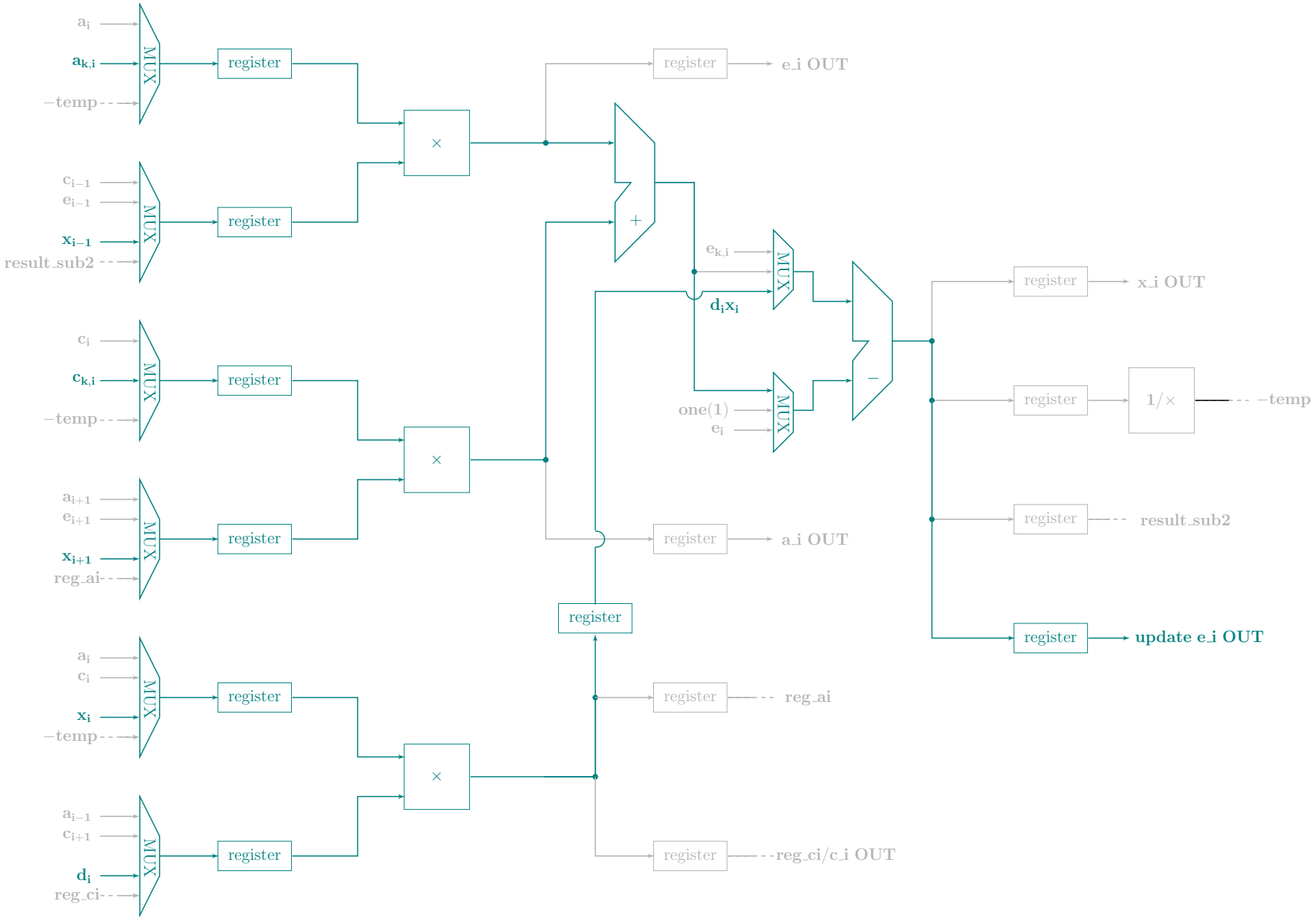
Σχήμα 5.13: Input Registers της Update RHS e Phase.

Όταν ξεκινάει η update e phase, διαβάζεται το στοιχείο της θέσης 0 και γράφεται στον καταχωρητή x_{i+1}. Στον επόμενο κύκλο, διαβάζεται το στοιχείο της θέσης 1, γράφεται στον x_{i+1} και το στοιχείο που υπήρχε εκεί από τον προηγούμενο κύκλο, πηγαίνει στον x_i. Ως στοιχείο x_{i-1} της θέσης -1 θεωρείται η τιμή 0. Στους καταχωρητές a_i, c_i, d_i μπαίνουν τα a_0, c_0, d_0. Τα δεδομένα μπαίνουν στη μονάδα Calculation όπου εκτελούνται οι πράξεις που φαίνονται στο σχήμα (5.6).

Στην επόμενη επανάληψη, θα διαβαστεί το στοιχείο της θέσης 2 και θα γραφτεί στον καταχωρητή $i+1$. Τα στοιχεία των θέσεων 0 και 1 από τους i και $i+1$ καταλήγουν στους $i-1$ και i αντίστοιχα. Στους καταχωρητές a, c, d μπαίνουν τα a_1, c_1, d_1. Τα δεδομένα είναι έτοιμα για επεξεγασία. Η διαδικασία συνεχίζεται με τον ίδιο τρόπο.

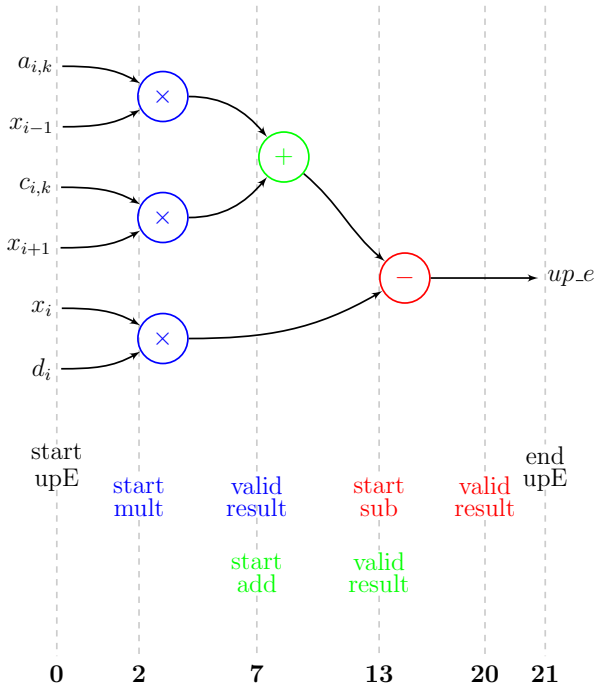
5.3.3.2 Μονάδα Calculation

Για την Update RHS e Phase, η διάταξη των πράξεων είναι ίδια με αυτή της Backward και της Forward. Επομένως, δε χρειάστηκε να προστεθεί κάτι στη μονάδα Calculation, απλά συνδέθηκαν οι μεταβλητές με τους αντίστοιχους FPO. Στο σχήμα (5.14) παρουσιάζεται το Calculation και έχει τονιστεί με χρώμα το datapath μόνο της Update RHS e Phase.



Σχήμα 5.14: Μονάδα Calculation.

Η αλληλουχία των πράξεων της φάσης Update RHS e φαίνεται στο σχήμα (5.15).



Σχήμα 5.15: Διάγραμμα αλληλουχίας πράξεων της Update RHS e Phase

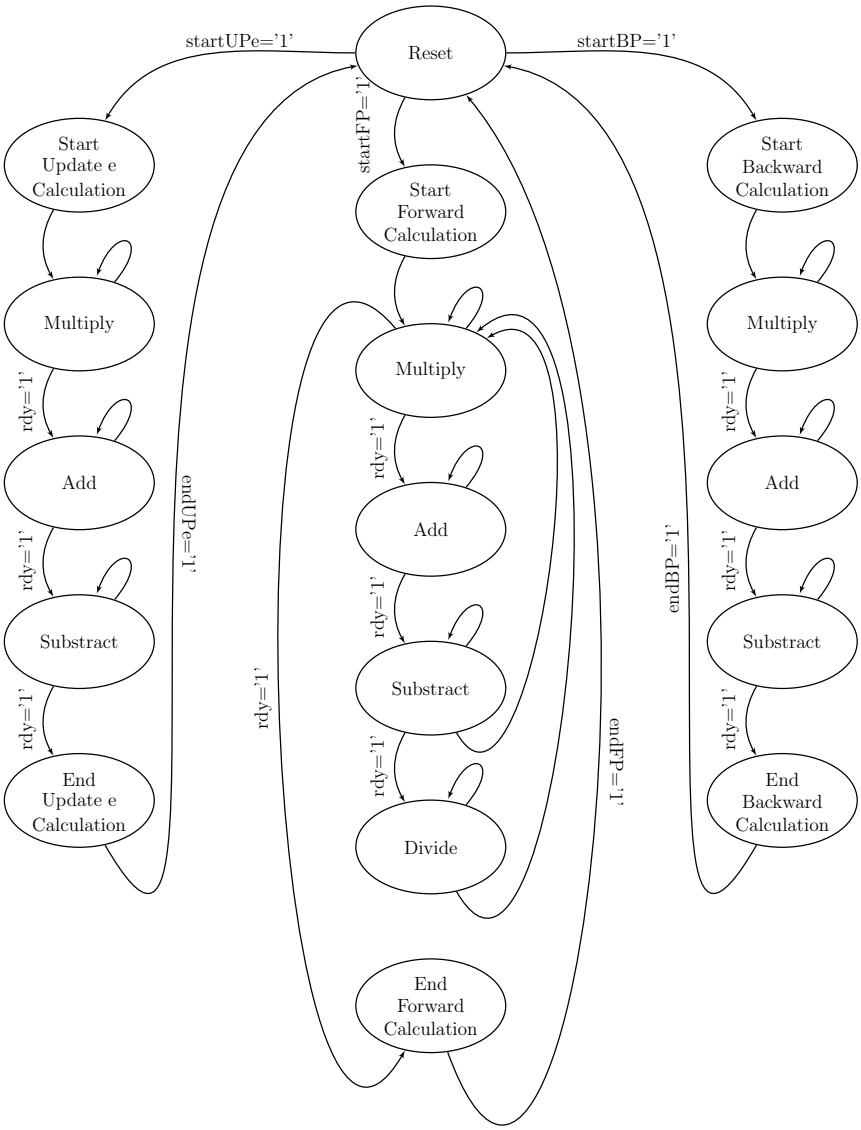
5.3.4 Εξωτερική και Εσωτερική Μονάδα Ελέγχου

Για να λειτουργήσει σωστά το μονοπάτι δεδομένων είναι απαραίτητη η παρουσία μιας μονάδας ελέγχου. Η μονάδα αυτή αποτελείται από σύνολο πεπερασμένων καταστάσεων (Finite State Machine-FSM) και δημιουργεί τα σωστά σήματα για να δουλέψουν οι καταχωρητές, οι πολυπλέκτες, οι FPO καθώς και για εγγραφή ή ανάγνωση από τις μνήμες. Υπάρχουν 2 μηχανές καταστάσεων που συντονίζουν τη λειτουργία της αρχιτεκτονικής, η εξωτερική και η εσωτερική.

Η εξωτερική FSM (5.19) ελέγχει την ανάγνωση των στοιχείων από τις μνήμες (διεύθυνση μνήμης και σήματα enable των θυρών) καθώς και την εγγραφή τους στους Input Registers. Μόλις οι σωστές μεταβλητές μπουν στους Input Registers, σηκώνεται ένα σήμα start (start_FP, start_BP, start_UPe, ανάλογα με τη φάση) και θέτει σε λειτουργία την εσωτερική FSM. Όταν πάρει από εκείνη ένα σήμα end (end_FP, end_BP, end_UPe, ανάλογα με τη φάση), ελέγχει την εγγραφή των τιμών στις κατάλληλες θέσεις μνήμης.

Η εσωτερική FSM, ανάλογα με το σήμα start που πήρε, ελέγχει τη διαδρομή των δεδομένων μέσα από τους κατάλληλους καταχωρητές και FPO του Calculation. Μόλις υπάρξει τελικό αποτέλεσμα, στέλνει στην εξωτερική FSM ένα σήμα end.

Στο σχήμα (5.16) παρουσιάζεται η εσωτερική μονάδα ελέγχου. Κατά τη Forward Phase ενεργοποιούνται οι καταστάσεις της κεντρικής διακλάδωσης, κατά τη Backward ενεργοποιούνται οι καταστάσεις της δεξιάς και κατά την Update e, οι καταστάσεις της αριστερής.

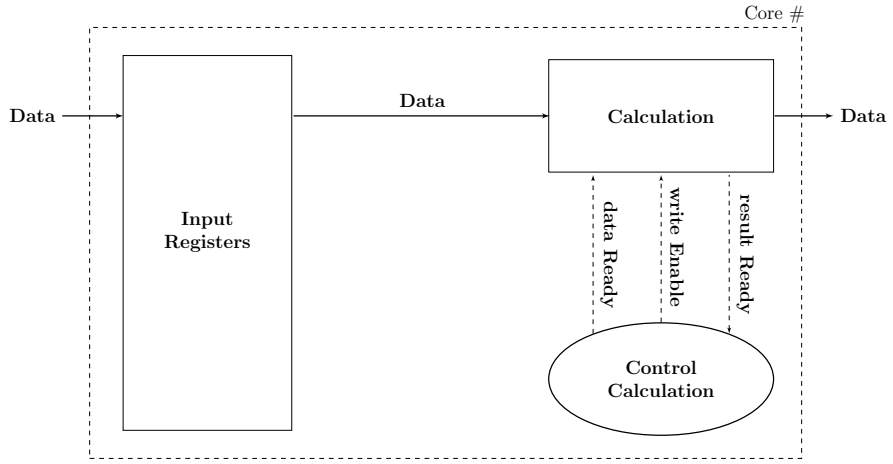


Σχήμα 5.16: Μηχανή Πεπερασμένων Καταστάσεων που ελέγχει τη Μονάδα Calculation και για τις τρεις φάσεις του αλγόριθμου.

5.3.5 Μονάδα Core

Το Calculation μαζί με την εσωτερική FSM (control calculation) και τους Input Registers αποτελούν τη μονάδα Core, το βασικό υποσύστημα της αρχιτεκτονικής, όπως φαίνεται στο σχήμα (5.17).

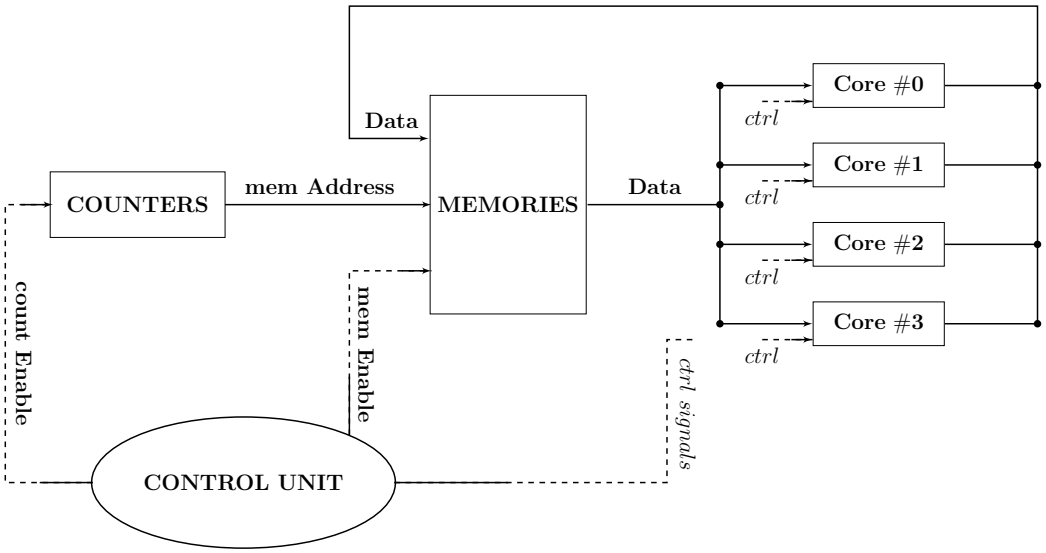
Τα cores μπορούν να πολλαπλασιαστούν ανάλογα με τους πόρους που διαθέτει η FPGA (DSP48E slices, LUTs και Flip-Flops). Υλοποιήθηκαν συστήματα με ένα, με τέσσερα και με δεκαέξι cores.



Σχήμα 5.17: Μονάδα Core

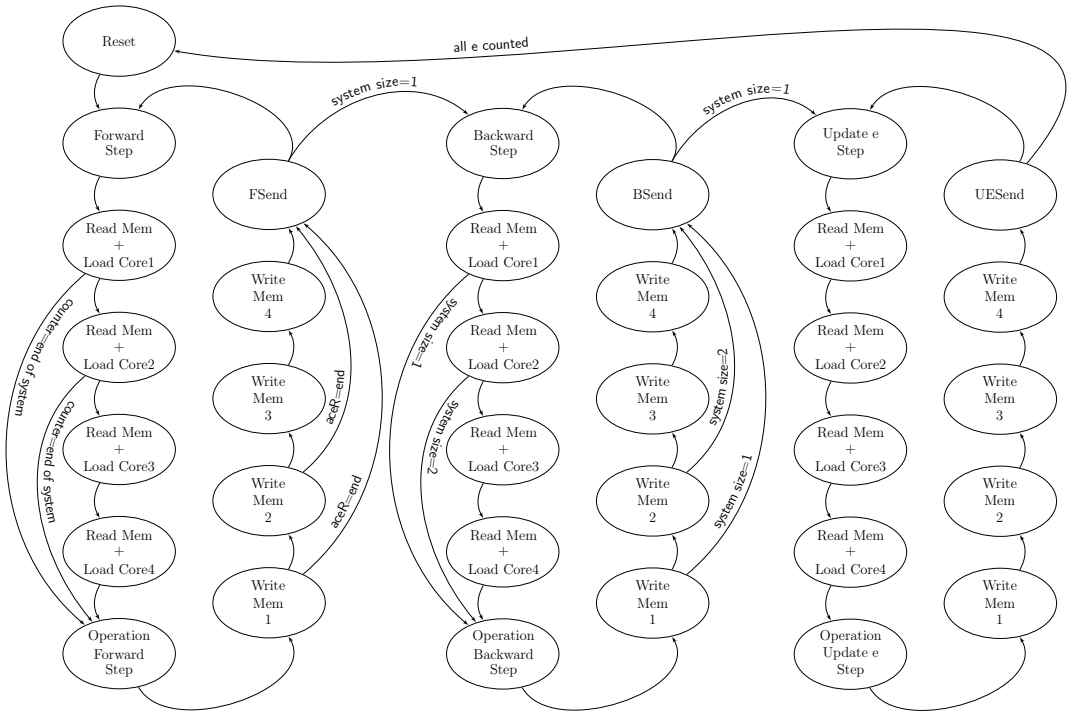
5.3.6 Συνολική Εικόνα της Αρχιτεκτονικής

Στις προηγούμενες ενότητες παρουσιάστηκαν και αναλύθηκαν τα επιμέρους υποσυστήματα που απαρτίζουν την αρχιτεκτονική. Το σχήμα (5.18) απεικονίζει το Block Diagram της συνολικής σχεδίασης, για 4 cores. Μπορούν να προστεθούν εύκολα cores χωρίς να αλλάξει η συνολική εικόνα του συστήματος. Η μονάδα Counters υπολογίζει τις διευθύνσεις ανάγνωσης και εγγραφής στις μνήμες.



Σχήμα 5.18: Top level architecture για 4 cores.

Η Εξωτερική FSM (Control Unit) *για την αρχιτεκτονική των 4 cores* φαίνεται στο σχήμα (5.19). Οι καταστάσεις Read Mem+Load Core και οι Write Mem αυξάνονται με την αύξηση των cores.



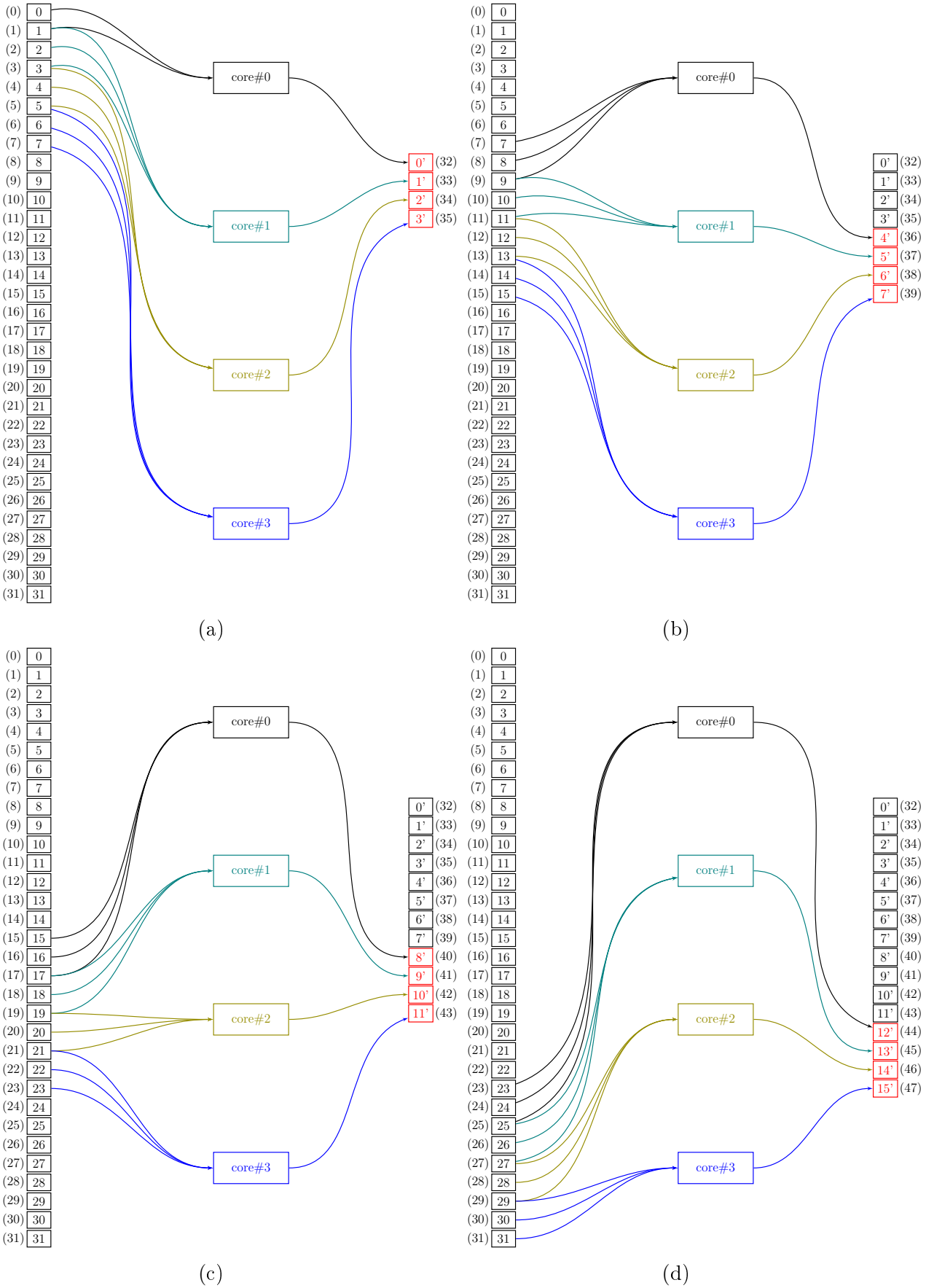
Σχήμα 5.19: Εξωτερική FSM για αρχιτεκτονική με 4 cores

5.3.7 Σχηματική Απεικόνιση των Φάσεων

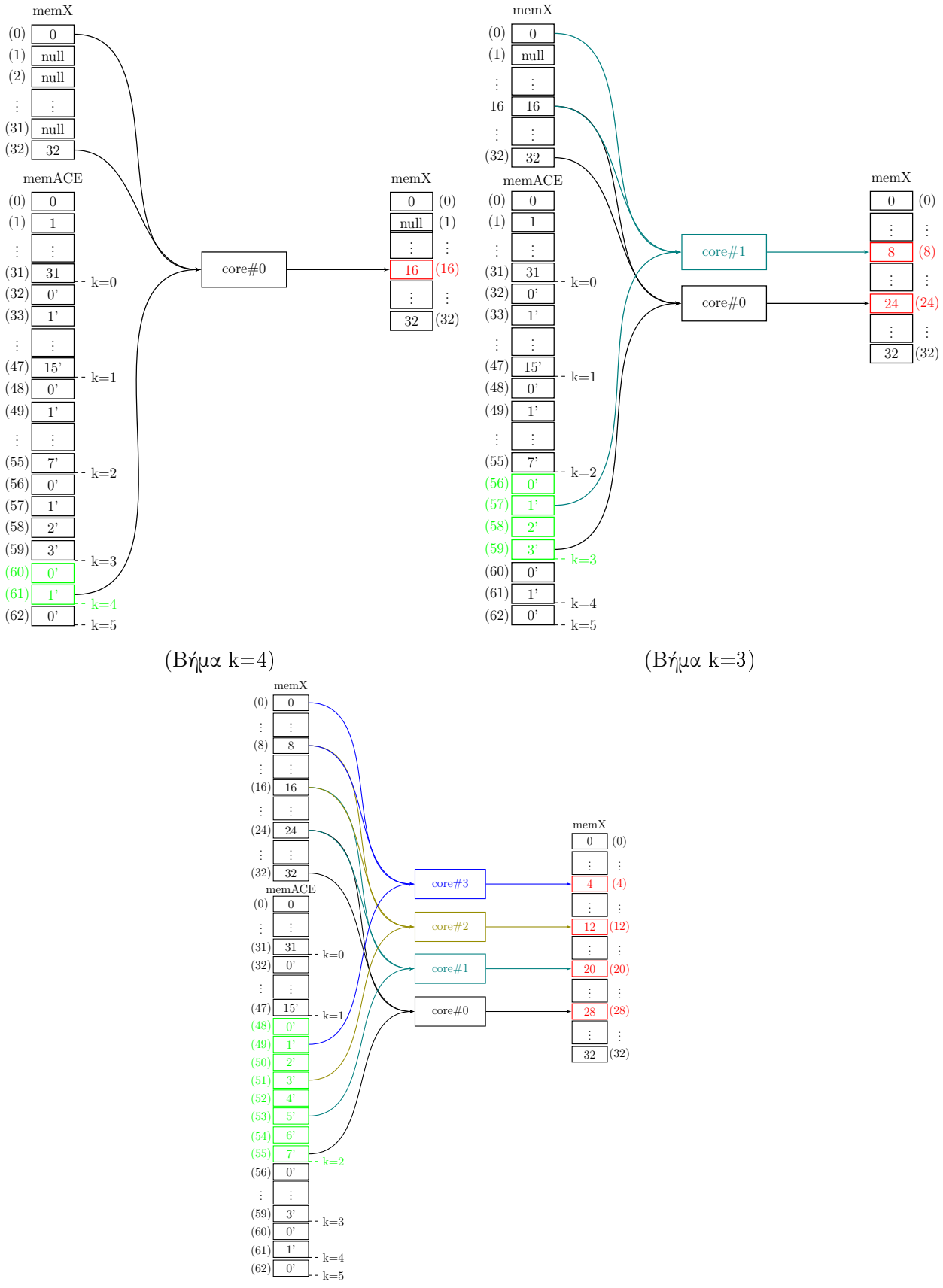
Στην ενότητα (5.3.1) αναλύθηκε η Forward Phase. Στο σχήμα (5.20) παρουσιάζεται η διαδικασία ανάγνωσης και εγγραφής στις μνήμες στη φάση αυτή, για ένα τριδιαγώνιο σύστημα μεγέθους 32, σε μια αρχιτεκτονική σχεδίαση με 4 cores, για ένα βήμα k. Τα κουτάκια αντιπροσωπεύουν τις θέσεις των μνημών A,C,E και οι αριθμοί στην παρένθεση, δεξιά και αριστερά, το index των θέσεων αυτών. Οι αριθμοί μέσα στα κουτιά δείχνουν τη γραμμή των στοιχείων του τριδιαγώνιου. Το αρχικό σύστημα βρίσκεται αριστερά των cores (αρχικό βήμα $k = 0$) και όπως φαίνεται, το index του συστήματος ταυτίζεται με αυτό των μνημών (0 έως 31). Δεξιά των cores είναι τα στοιχεία που υπολογίζονται στο Calculation και γράφονται στις επόμενες θέσεις - 32 έως 47 (βήμα $k = 1$).

Χρειάζονται 2 κύκλοι για να γεμίσει κάθε core. Μόλις οι μεταβλητές φορτωθούν στους Input Registers, "σηκώνεται" το σήμα start_FP και το Calculation ξεκινάει την επεξεργασία. Μόλις έχουμε έγκυρα αποτελέσματα, σηκώνεται το σήμα end_FP και τα νέα a,c,e πηγαίνουν στη μνήμη για εγγραφή. Χρειάζονται 4 κύκλοι για να γραφούν τα στοιχεία στις μνήμες.

Για σύστημα μεγέθους N=32 στοιχείων, η Forward Phase θέλει $k = log_2 N = 5$ βήματα για να ολοκληρωθεί. Στο βήμα $k = 3$ το μέγεθος του συστήματος προς επεξεργασία είναι 4 και θα δουλέψουν μόνο τα δύο πρώτα cores, ενώ στο $k = 4$ τα στοιχεία είναι 2 και θα δουλέψει μόνο το ένα core.



Σχήμα 5.20: Ανάγνωση και εγγραφή μνημών στη Forward Phase για ένα βήμα k



(Βήμα k=4) (Βήμα k=3)

(Βήμα k=2)

Σχήμα 5.21: Ανάγνωση και εγγραφή μνημών στη Backward Phase για τρία βήματα k

Στο σχήμα (5.21), παρουσιάζεται αναλυτικά η διαδικασία ανάγνωσης και εγγραφής στη μνήμη κατά τη Backward Phase για τα τρία πρώτα βήματα k. Το αρχικό τριδιαγώνιο σύστημα είχε μέγεθος 32 και η σχεδίαση έχει 4 cores. Το βήμα $k = log_2 N = 5$ είναι κοινό και στις δύο φάσεις και σάυτό εξάγεται το x_0. Στο επόμενο, $k = 4$, το x_0 και το $x_{32} = 0$ μαζί με τα $(a, c, e)_{61}$, που είχαν υπολογιστεί στο $k = 4$ της Forward Phase, υπολογίζουν το x_{16} (δουλεύει μόνο ένα core). Στο επόμενο βήμα, τα $x_{32} = 0$, x_{16} και $(a, c, e)_{59}$ υπολογίζουν το x_{24} (δουλεύει το core 0) και τα x_{16}, x_0 και $(a, c, e)_{57}$ εξάγουν το x_8 (δουλεύει το core 1).
Στο $k = 2$ δουλεύουν και τα τέσσερα cores:

- core 0: παίρνει τα $x_{32} = 0$, x_{24}, $(a, c, e)_{55}$ και δίνει το x_{28}
- core 1: παίρνει τα x_{24}, x_{16}, $(a, c, e)_{53}$ και δίνει το x_{20}
- core 2: παίρνει τα x_{16}, x_8, $(a, c, e)_{51}$ και δίνει το x_{12}
- core 3: παίρνει τα x_8, x_0, $(a, c, e)_{49}$ και δίνει το x_4

Σε κάθε επόμενο βήμα επαναλαμβάνεται η διαδικασία μέχρι να υπολογιστούν όλα τα x.

Στην Update e Phase, τα cores γεμίζουν με τον ίδιο τρόπο όπως και στη Forward (με τη διαφορά ότι στη Forward ξεκινούν να δουλεύουν ανά δύο κύκλους ενώ στην Update e ανά ένα κύκλο). Τα αποτελέσματα γράφονται στη μνήμη mem E στις θέσεις 0-31, πανωγράφοντας το διάνυσμα E που υπήρχε εκεί.

Με το τέλος της Update e Phase, το τριδιαγώνιο σύστημα με το καινούριο διάνυσμα E, μπαίνει ξανά στη Forward Phase (επόμενο **χρονικό βήμα** n του σχήματος Crank-Nicolson).

Το μειονέκτημα της αρχιτεκτονικής που παρουσιάστηκε σάυτή την ενότητα ήταν η αποθήκευση των δεδομένων στις μνήμες και η διασύνδεση των μνημών με τα cores. Η χρήση μιας μνήμης για κάθε διάνυσμα και η σειριακή ανάγνωση και εγγραφή των στοιχείων από και προς αυτές είναι χρονοβόρα. Στο συγκεκριμένο παράδειγμα, το σύστημα με 4 cores χρειάζεται 8 κύκλους για να τα "γεμίσει" όλα και 4 κύκλους για να γράψει τα νέα δεδομένα στη μνήμη. Με κάθε διπλασιασμό των cores, διπλασιάζεται και ο αριθμός των κύκλων: 16 cores $\rightarrow$ 32 κύκλοι, 32 cores $\rightarrow$ 64 κύκλοι κ.ο.κ. δηλαδή, οι συνέπειες αυτού του bottleneck είναι πιο άμεσες όσο μεγαλώνει το κύκλωμα αφού τα cores δε δουλεύουν παράλληλα αλλά περιμένει το ένα να γεμίσουν όλα τα προηγούμενα του για να ξεκινήσει την επεξεργασία. Η λύση είναι η αποθήκευση των διανυσμάτων σε περισσότερες από μία μνήμες και συγκεκριμένα κάθε core να έχει τη δική του μνήμη. Έτσι υλοποιήθηκε η 2η και προτεινόμενη αρχιτεκτονική.

5.4 Δεύτερη Αρχιτεκτονική

Σάυτή την προτεινόμενη αρχιτεκτονική, το σύστημα δεν αποθηκεύεται σε μία μνήμη αλλά χωρίζεται σε περισσότερες ανάλογα με τον αριθμό των cores - κάθε core έχει τη δική του μνήμη. Χρησιμοποιήθηκαν true-dual port brams ώστε σε έναν κύκλο να βγαίνουν 2 στοιχεία. Η μονάδα Core (Calculation-Control Calculation & Input Registers) παρέμεινε ίδια. Αυξήθηκαν οι μνήμες και άλλαξαν ο τρόπος αποθήκευσης των δεδομένων σε αυτές και η διασύνδεσή τους με τα cores.

5.4.1 Forward Phase

Στα σχήματα (5.22) γίνεται η αναπαράσταση ανάγνωσης και εγγραφής από και προς τις μνήμες κατά τη Forward Phase. Χρησιμοποιήθηκε το ίδιο παράδειγμα με την προηγούμενη ενότητα, δηλαδή ένα τριδιαγώνιο σύστημα μεγέθους 32, σε μια αρχιτεκτονική σχεδίαση με 4 cores, για ένα βήμα k. Τα 32 a, c, e "σπάνε" σε 4 οκτάδες ως εξής:

- Τα στοιχεία 0-7 πάνε στη μνήμη 0 (index μνήμης 0-7).
- Τα στοιχεία 8-11 στη μνήμη 1 (index μνήμης 0-7).
- Τα 12-23 στη μνήμη 2 (index μνήμης 0-7).
- Τα 24-31 στη μνήμη 3 (index μνήμης 0-7).

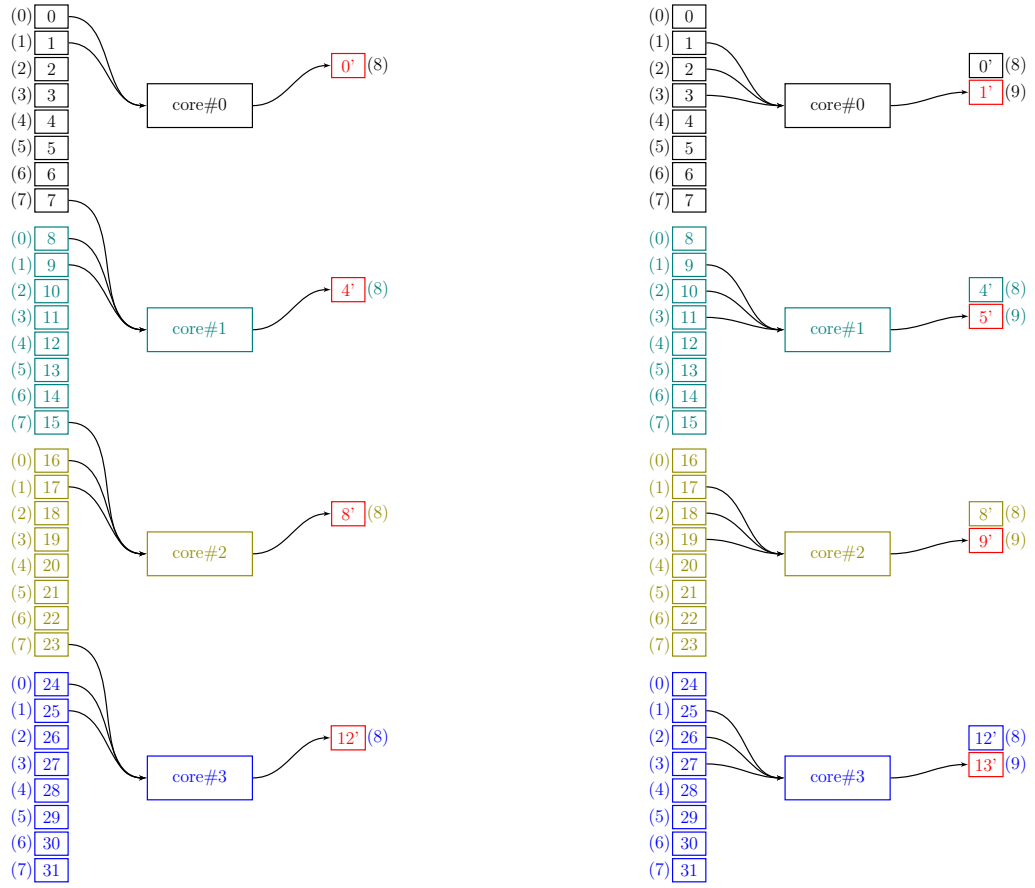
Με αυτό τον τρόπο κάθε core έχει το δικό του τριδιαγώνιο σύστημα το οποίο μειώνεται στο μισό και αποθηκεύεται πίσω στη δική του μνήμη στις θέσεις 8-11. Στην αρχή του βήματος απαιτούνται 2 κύκλοι για να γεμίσουν όλα τα cores. Σε κάθε επόμενη επανάληψη, απαιτείται ένας κύκλος.

Στο βήμα $k = 3$ το σύστημα έχει μέγεθος 4, δηλαδή κάθε μνήμη θα δώσει από μία μεταβλητή για επεξεργασία. Σάυτή την περίπτωση και για τα υπόλοιπα k τα cores δουλεύουν όπως στο σχήμα (4.1).

Στο συγκεκριμένο παράδειγμα: Το core 0 θα πάρει από τη μνήμη 0 τα $(a, c, e)_{0'}$ και από τη μνήμη 1 τα $(a, c, e)_{1'}$ και θα υπολογίσει τα $(a, c, e)_{0'}$ του $k = 4$ που θα αποθηκευτούν στη μνήμη 0. Το core 2 θα πάρει από τη μνήμη 1 τα $(a, c, e)_{1'}$, από τη μνήμη 2 τα $(a, c, e)_{2'}$ και από τη μνήμη 3 τα $(a, c, e)_{3'}$ και θα υπολογίσει τα $(a, c, e)_{1'}$ του $k = 4$ που θα αποθηκευτούν στη μνήμη 2.

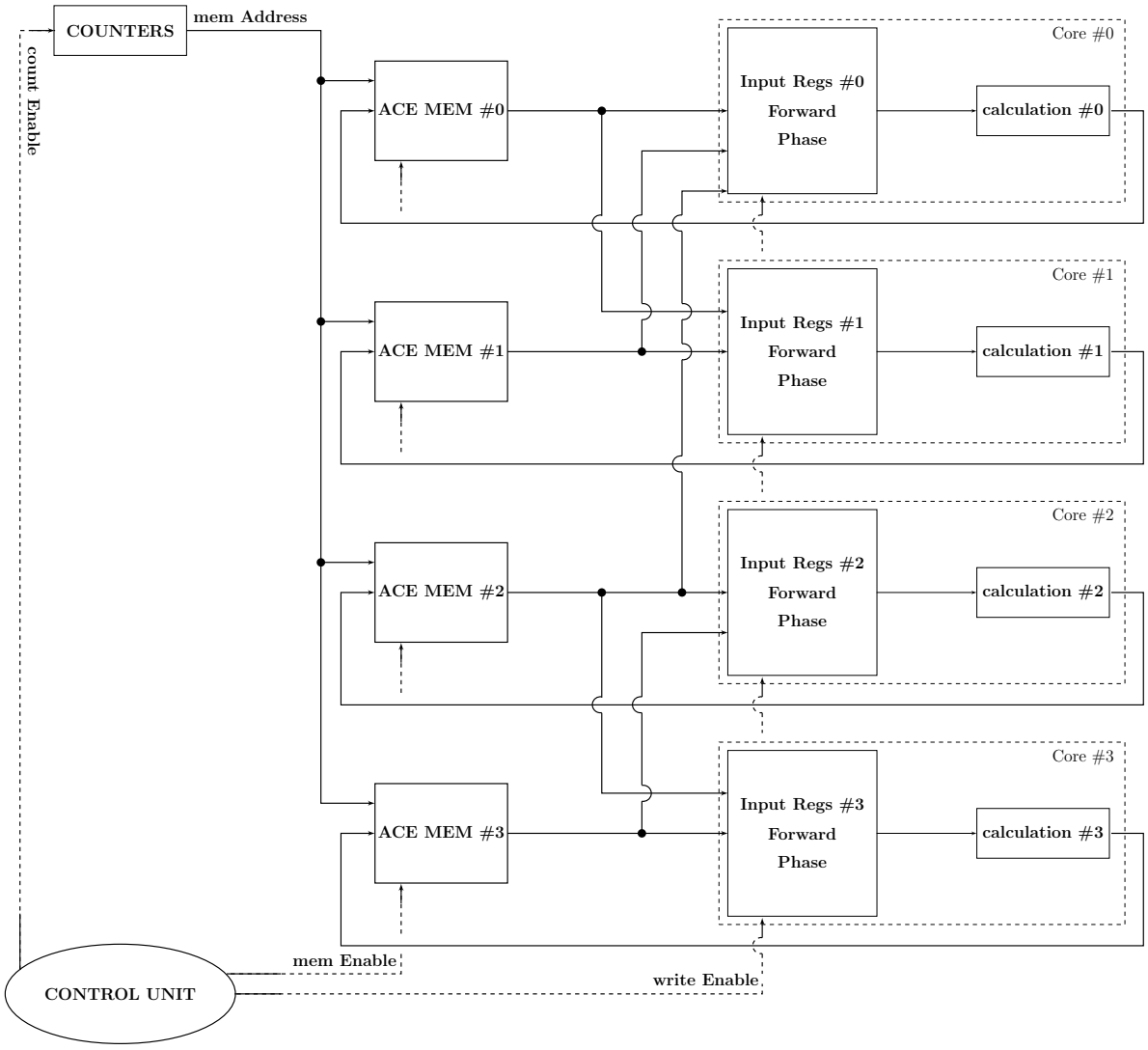
Στο επόμενο και τελευταίο βήμα το core 0 θα πάρει από τη μνήμη 0 τα $(a, c, e)_{0'}$ και από τη μνήμη 2 τα $(a, c, e)_{1'}$ και θα υπολογίσει το $(a, c, e)_{0'}$ που θα αποθηκευτούν στη μνήμη 0.

Η Forward Phase έχει ολοκληρωθεί.



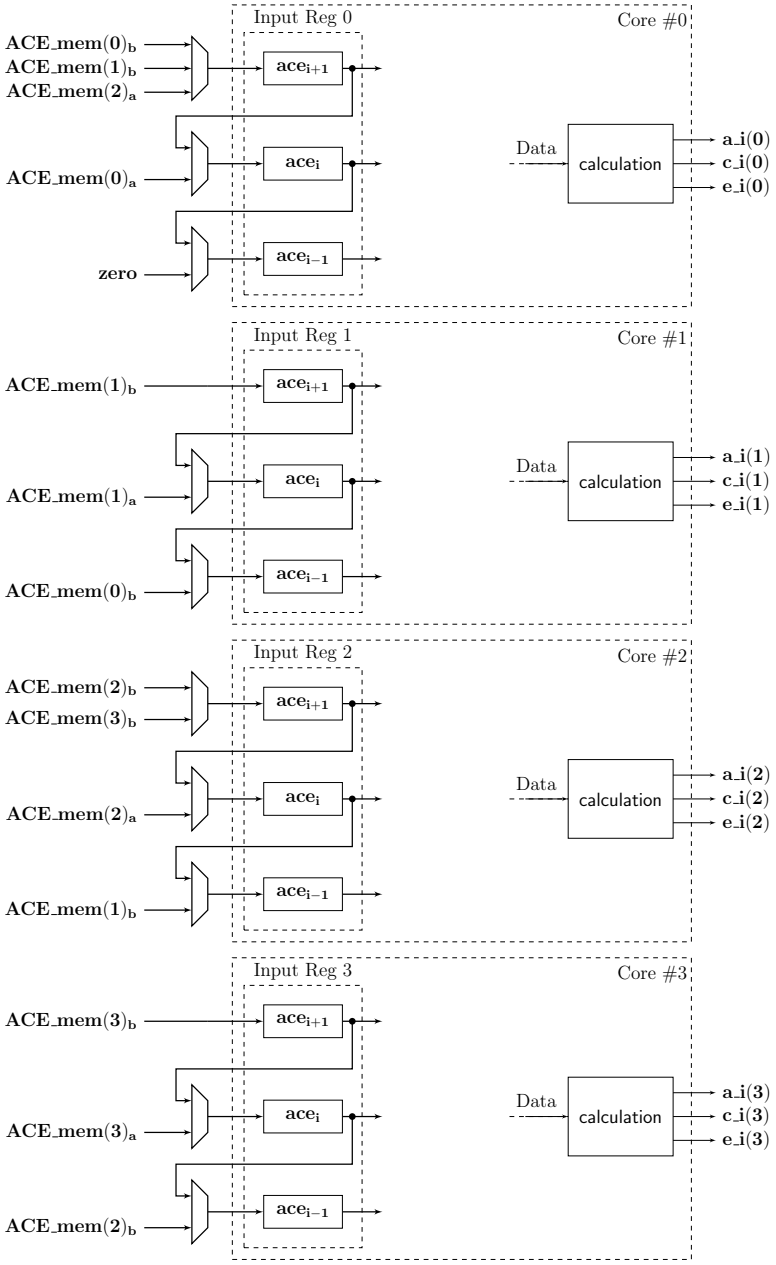
Σχήμα 5.22: Ανάγνωση & εγγραφή μνημών στη Forward Phase της δεύτερης σχεδίασης

Το σχήμα (5.23) απεικονίζει το Block Diagram της νέας αρχιτεκτονικής μόνο για τη Forward Phase, για ένα σύστημα με 4 cores. Για λόγους απλότητας και ευκολότερης κατανόησης των σχημάτων, η συνολική αρχιτεκτονική θα παρουσιαστεί σε ξεχωριστά διαγράμματα ανά φάση. Για το ίδιο λόγο έχει παραληφθεί από όλα τα σχήματα το ζοντρολ ζαλζυλατιον. Ανάλογα με το βήμα k ένα ζορε μπορεί να πάρει στοιχεία τόσο από τη μνήμη του όσο και από άλλες μνήμες αλλά τα αποτελέσματα που εξάγει πάνε μόνο στη δική του μνήμη.



Σχήμα 5.23: Block Diagram της δεύτερης σχεδίασης για τη Forward Phase

Στο σχήμα (5.24) φαίνεται η διασύνδεση των μνημών με τους Input Registers μόνο για τη Forward Phase. Οι αριθμοί δηλώνουν τη μνήμη από την οποία έρχονται τα δεδομένα και τα a, b τη θύρα της μνήμης. Παρατηρείται ότι έχουν μπει πολυπλέκτες μόνο πριν την είσοδο των μεταβλητών στα cores και η διαδρομή δεδομένων Input Registers $\rightarrow$ Calculation $\rightarrow$ Μνήμη είναι σταθερή.

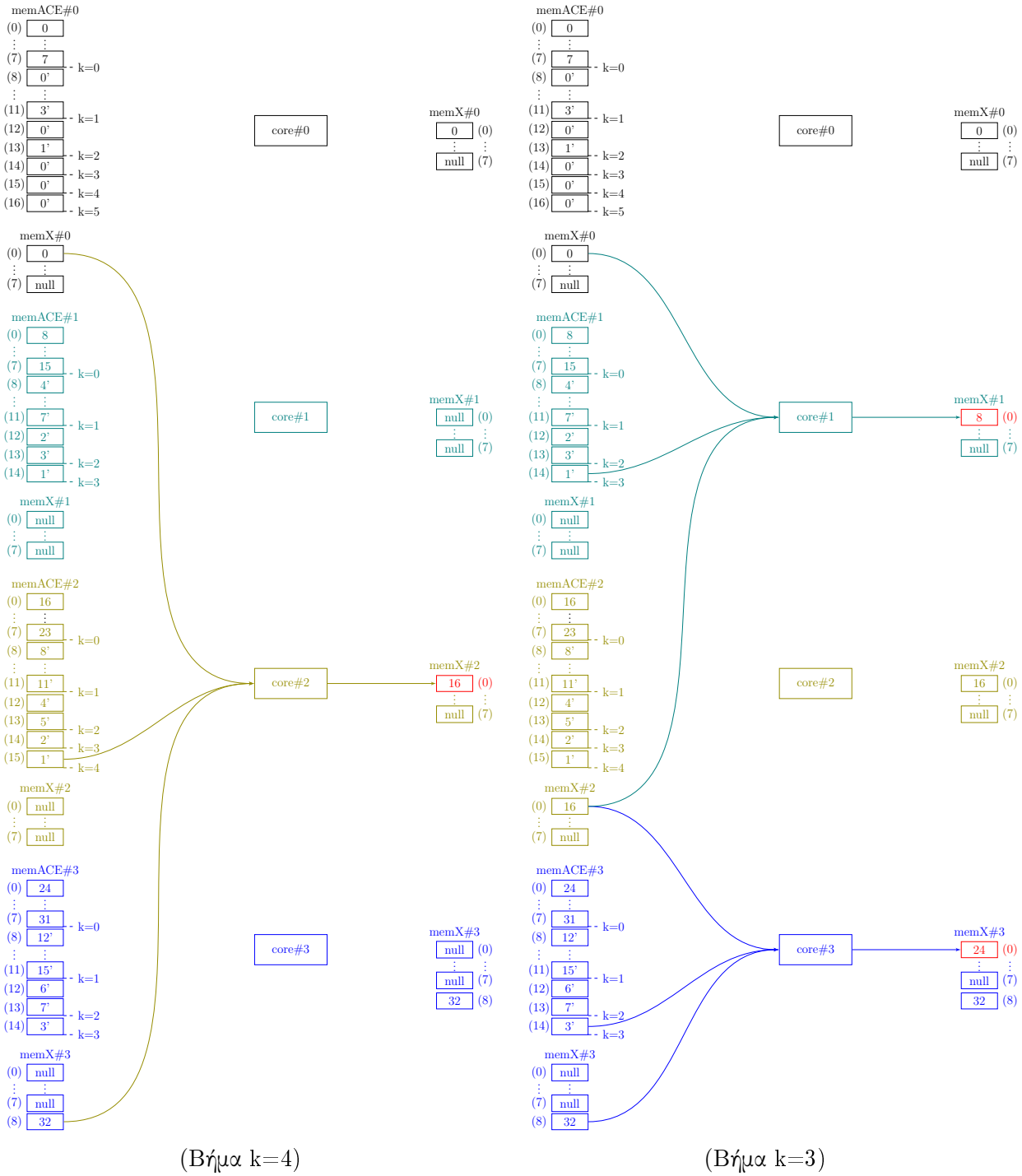


Σχήμα 5.24: Διασύνδεση μνημών - cores για τη Forward Phase

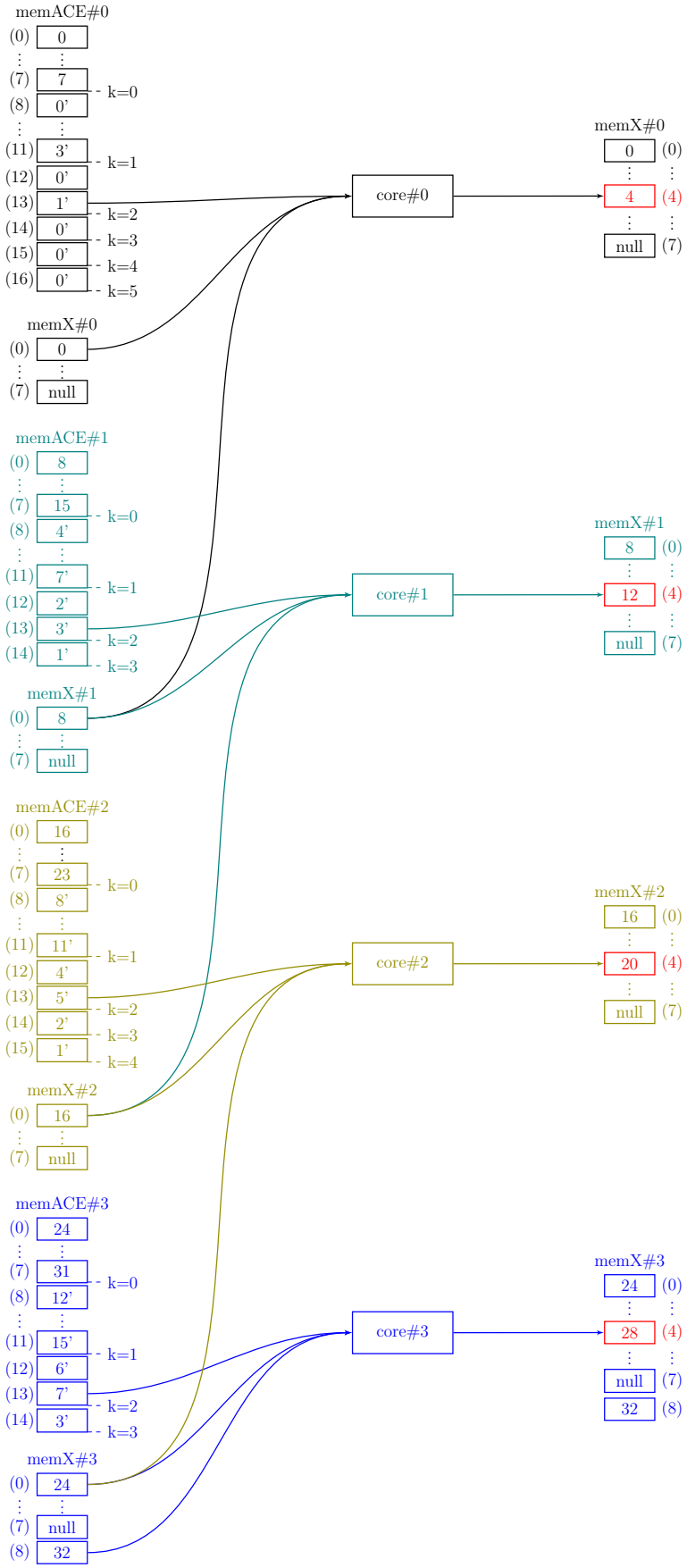
5.4.2 Backward Phase

Στα σχήματα (5.25)&(5.26) γίνεται η αναπαράσταση ανάγνωσης και εγγραφής από και προς τις μνήμες κατά τη Backward Phase για τα τρία πρώτα βήματα. Τα στοιχεία x που θα υπολογιστούν, θα αποθηκευτούν και αυτά σε τέσσερις μνήμες.

Στα βήμα $k = 4$ δουλεύει μόνο ένα core και αυτό είναι το 2, ενώ στο $k = 3$ δουλεύουν δύο, τα 1 και 3. Τα cores δουλεύουν με αυτή τη σειρά για να μπορούν να παίρνουν τα a, c, e από τις δικές τους μνήμες - όπως αυτά αποθηκεύτηκαν στα διάφορα βήματα της FP - και να γράφουν τα αποτελεσμάτα x πίσω στις δικές τους μνήμες. Με αυτό το τρόπο αποφεύγονται να χρησιμοποιηθούν επιπλέον πολυπλέκτες.



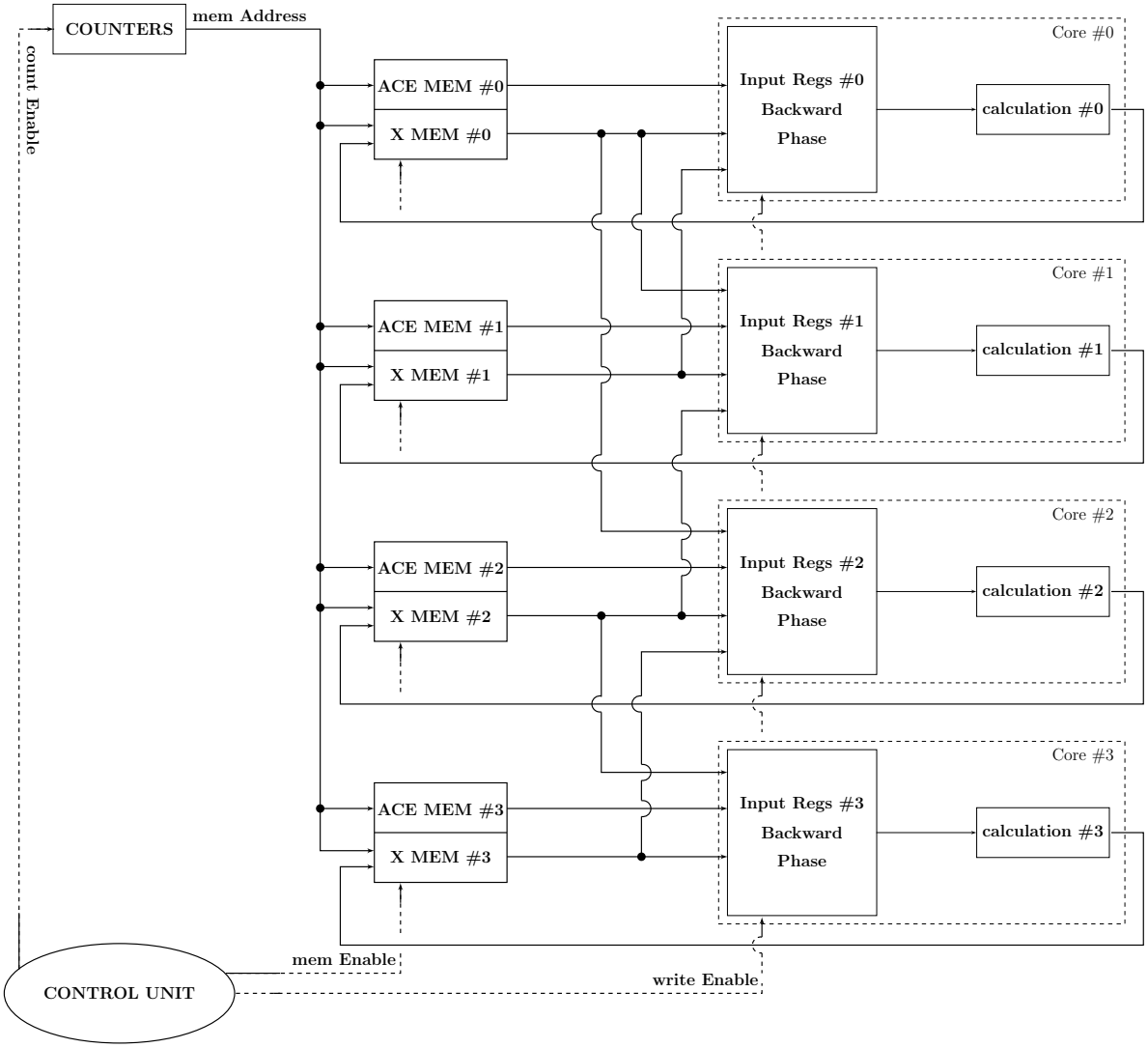
Σχήμα 5.25: Ανάγνωση & εγγραφή μνημών στη Backward Phase της δεύτερης σχεδίασης



(Βήμα k=4)

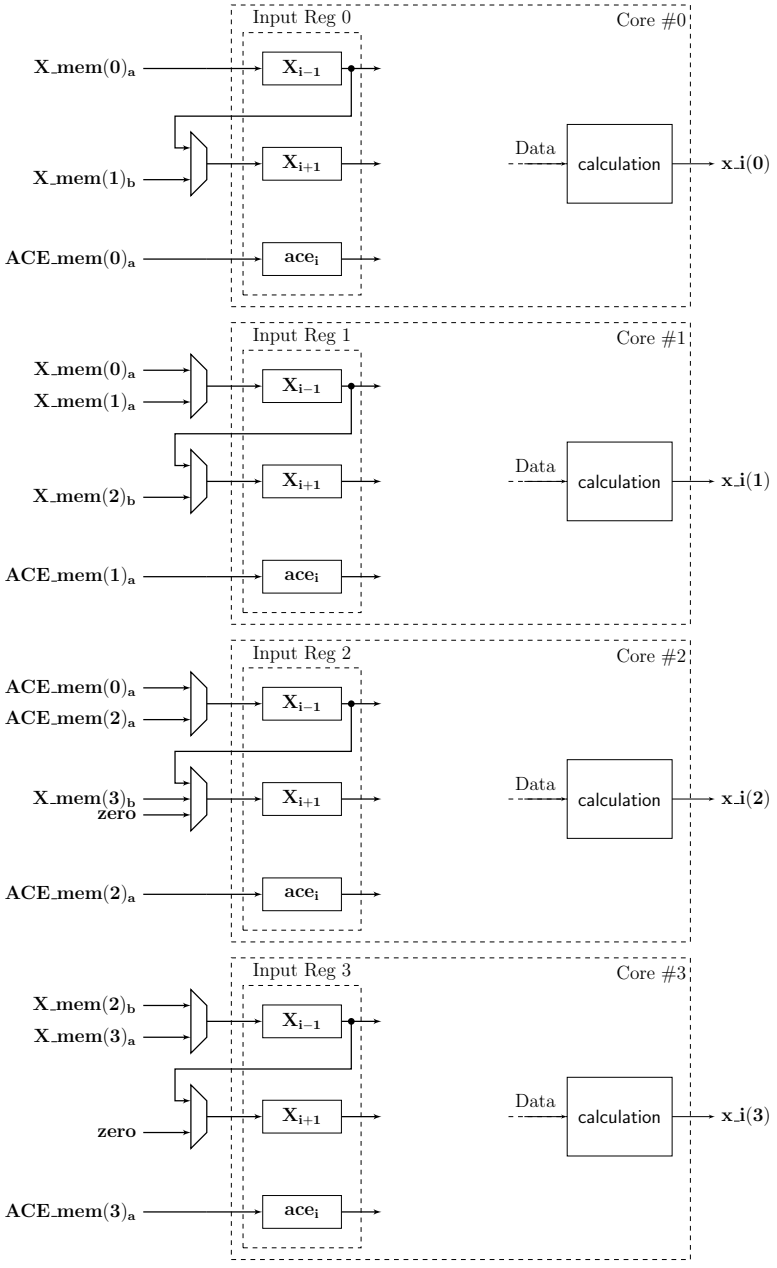
Σχήμα 5.26: Ανάγνωση & εγγραφή μνημών στη Backward Phase της δεύτερης σχεδίασης

Το σχήμα (5.27) απεικονίζει το Block Diagram της νέας αρχιτεκτονικής μόνο για τη Backward Phase, για ένα σύστημα με 4 cores. Ανάλογα με το βήμα k, ένα core μπορεί να πάρει στοιχεία x τόσο από τη μνήμη του όσο και από άλλες μνήμες. Ωστόσο, τα στοιχεία a, c, e τα παίρνει μόνο από τη δική του μνήμη και τα αποτελέσματα x που εξάγει πάνε μόνο στη δική του μνήμη.



Σχήμα 5.27: Block Diagram της δεύτερης σχεδίασης για τη Backward Phase

Στο σχήμα (5.28) φαίνεται η διασύνδεση των μνημών με τους Input Registers μόνο για τη Backward Phase. Όπως και στη Forward Phase, έχουν μπει πολυπλέκτες μόνο πριν την είσοδο των μεταβλητών x στα cores και η διαδρομή δεδομένων Input Registers → Calculation → Μνήμη είναι σταθερή.

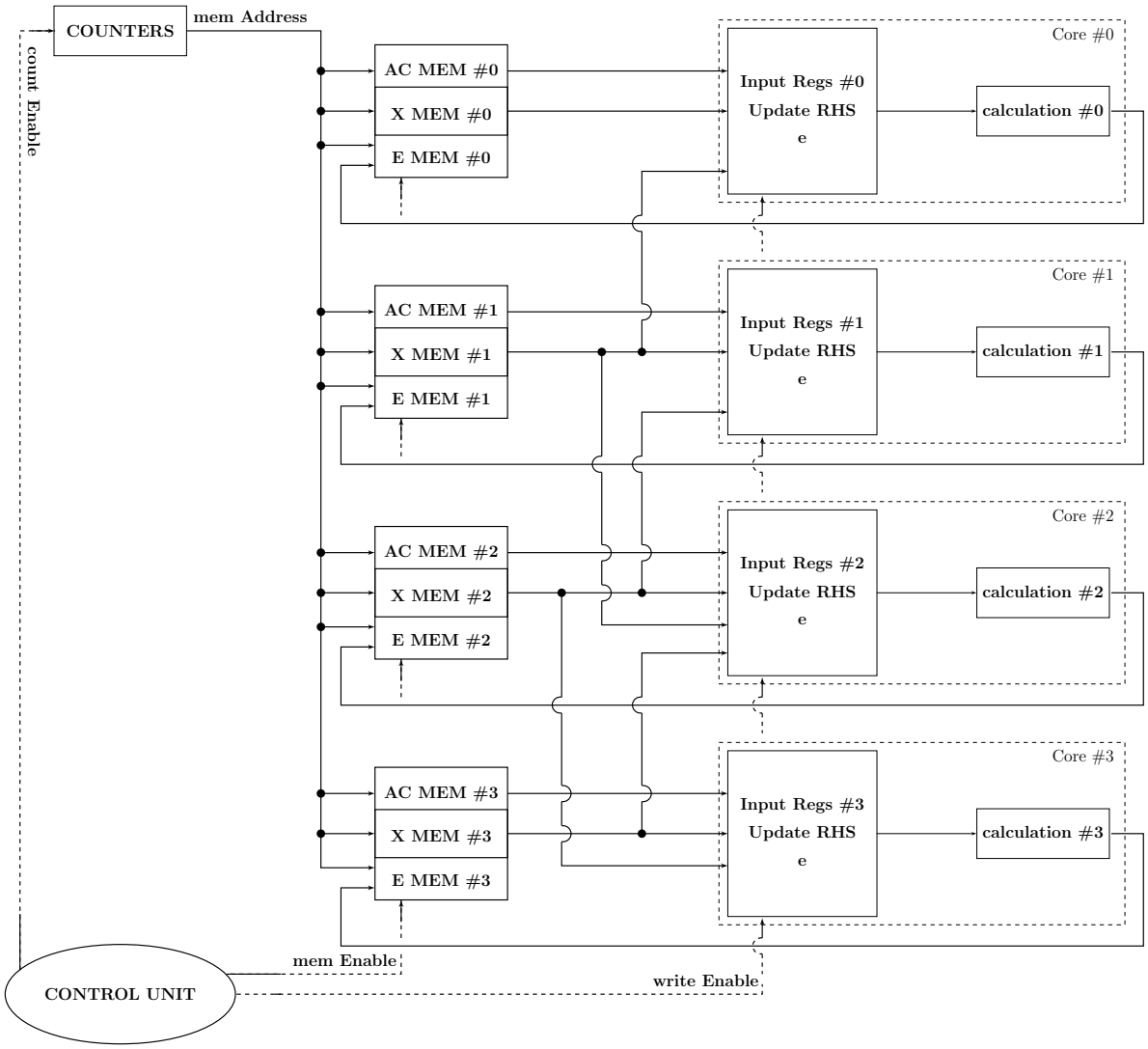


Σχήμα 5.28: Διασύνδεση μνημών - cores για τη Backward Phase

5.4.3 Update RHS e Phase

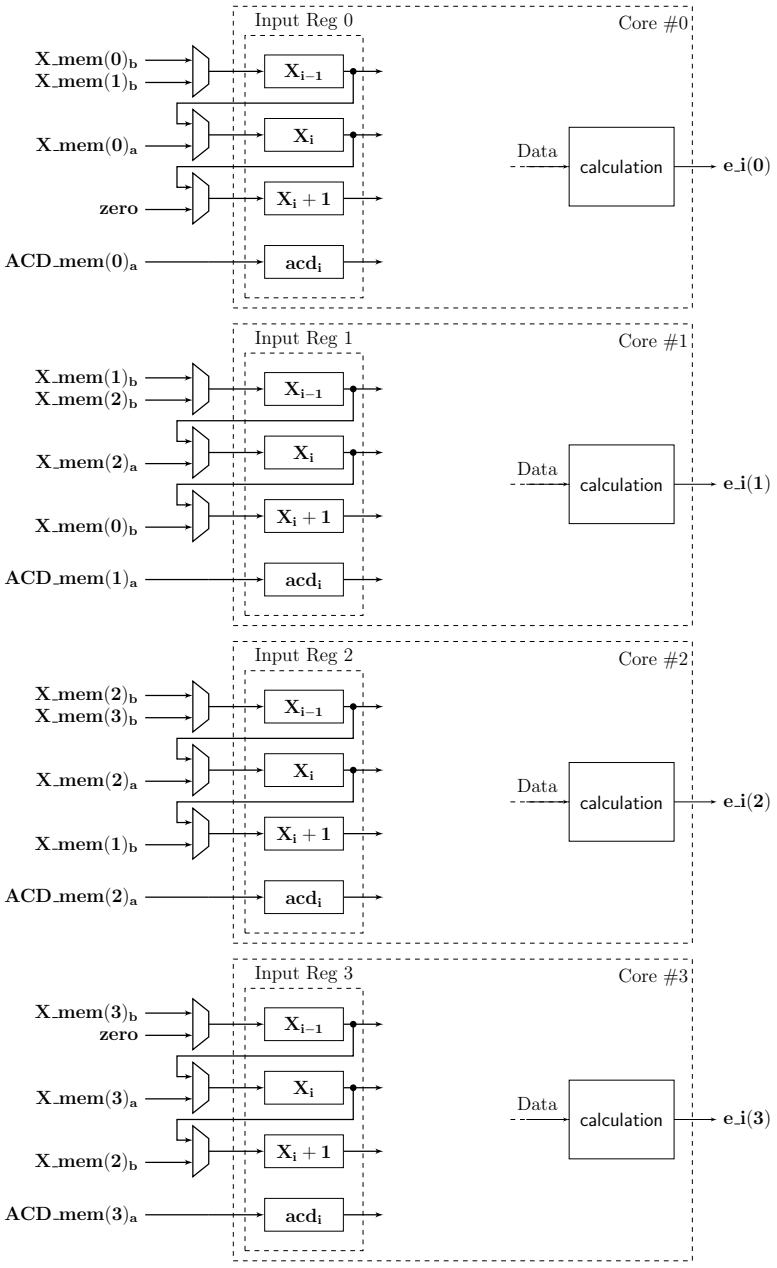
Στην Update e Phase, η διαδικασία ανάγνωσης και εγγραφής γίνεται με τον ίδιο τρόπο όπως και στη Forward, ενώ τα αποτελέσματα γράφονται στη μνήμες E πανωγράφοντας το αρχικό διάνυσμα που υπήρχε εκεί.

Το σχήμα (5.29) απεικονίζει το Block Diagram της νέας αρχιτεκτονικής μόνο για τη Update e Phase, για ένα σύστημα με 4 cores. Ανάλογα με το βήμα k, ένα core μπορεί να πάρει στοιχεία x τόσο από τη μνήμη του όσο και από άλλες μνήμες. Ωστόσο, τα στοιχεία a, c, d τα παίρνει μόνο από τη δική του μνήμη και τα αποτελέσματα e που εξάγει πάνε μόνο στη δική του μνήμη.



Σχήμα 5.29: Block Diagram της δεύτερης σχεδίασης για την Update e Phase

Στο σχήμα (5.30) φαίνεται η διασύνδεση των μνημών με τους Input Registers μόνο για τη Update e Phase. Όπως και στη Backward Phase, έχουν μπει πολυπλέκτες μόνο πριν την είσοδο των μεταβλητών x στα cores και η διαδρομή δεδομένων Input Registers → Calculation → Μνήμη είναι σταθερή.



Σχήμα 5.30: Διασύνδεση μνημών - cores για τη Update e Phase

5.4.4 Εξωτερική FSM

Η εξωτερική FSM της δεύτερης αρχιτεκτονικής, για 4 cores φαίνεται στο παρακάτω σχήμα. Σε αντίθεση με την αντίστοιχη της πρώτης αρχιτεκτονικής, όλοι οι πυρήνες φορτώνουν στην ίδια κατάσταση.

Σχήμα 5.31: Εξωτερική FSM για αρχιτεκτονική με 4 cores

Κεφάλαιο 6

Υλοποίηση, Απόδοση και Αποτελέσματα

6.1 Εισαγωγή

Στο κεφάλαιο αυτό εξετάζονται η παράμετροι της υλοποίησης ως προς την απόδοση της αρχιτεκτονικής και τα αποτελέσματα της διπλωματικής εργασίας.

6.2 Υλοποίηση

Ολοκληρώθηκαν και προσομοιώθηκαν αρχιτεκτονικές και των δύο εναλλακτικών που αναλύθηκαν στο κεφάλαιο 5. Χρησιμοποιήθηκε η γλώσσα VHDL και το εργαλείο Xilinx ISE Design Suite 13.1 για την ανάπτυξη των συστημάτων. Η FPGA που επιλέχθηκε είναι η XC5 VLX330T της οικογένειας Virtex 5.

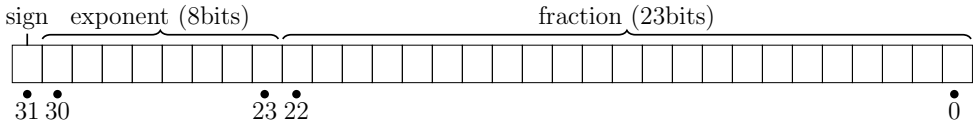
6.2.1 Αριθμοί κινητής υποδιαστολής μονής ακρίβειας

Όλες οι αριθμητικές πράξεις που εκτελούνται στις αρχιτεκτονικές είναι πράξεις κινητής υποδιαστολής μονής ακρίβειας (floating-point single-precision). Οι αριθμοί κινητής υποδιαστολής αποτελούνται από τρία μέρη: το πρόσημο (sign), το κλάσμα (fraction) και τον εκθέτη (exponent).

$$(-1)^{sign}(1.\underbrace{b_{-1}b_{-2}...b_{-23}}_{fraction})_2 \times 2^{exponent}$$

Η αναπαράσταση των αριθμών αυτών ακολουθεί το πρότυπο IEEE 754. Το συγκεκριμένο πρότυπο χρησιμοποιείται σχεδόν από όλους τους υπολογιστές που σχεδιάστηκαν από το 1980 και μετά. Η αρχική μονάδα ενός κανονικοποιημένου αριθμού παραλείπεται κατά την αναπαράρασταση σε bits. Το πεδίο του κλάσματος περιέχει μόνο το κλασματικό μέρος.

Κατά το πρότυπο IEEE 754, οι αριθμοί μονής ακρίβειας χρησιμοποιούν 8 bits για τον εκθέτη και 23 bits για το κλάσμα όπως φαίνεται στο Σχήμα (6.1).



Σχήμα 6.1: Αναπαράσταση των single-precision binary floating-point (IEEE 754 standard).

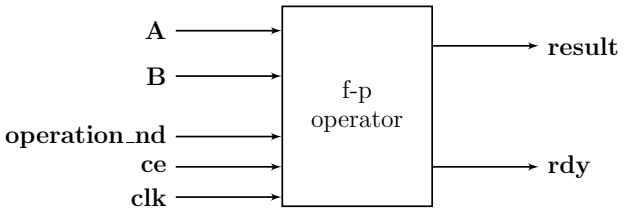
Αν s το πρόσημο, e ο biased εκθέτης και b το 23 bits κλάσμα, τότε η δεκαδική τιμή ενός αριθμού κινητής υποδιαστολής μονής ακρίβειας δίνεται από τον τύπο:

$$Value = (-1)^s \times (1 + \sum_{i=1}^{23} \lceil b_{-i} \rceil 2^{-i}) \times 2^{(e-127)} \tag{6.1}$$

Το μηδέν αναπαρίσταται ως $0...00_2$ και όχι σε μορφή κανονικοποιημένου επιστημονικού συμβολισμού.

6.2.2 Floating-point operators

Οι Floating-point operators (FPO) που χρησιμοποιήθηκαν, δημιουργήθηκαν με το εργαλείο Xilinx Core Generator 13.1 [32]. Εκτελούν πράξεις κινητής υποδιαστολής μονής ακρίβειας. Οι διεπαφές τους φαίνονται στο Σχήμα (6.2).



Σχήμα 6.2: Διεπαφές ενός Floating-point operator

Στον Πίνακα [6.1] παρουσιάζονται τα σήματα εισόδου-εξόδου ενός FPO καθώς και η περιγραφή της λειτουργίας τους.

signal	Width(bits)	Type	Description
A	32	Input	Input Operand A
B	32	Input	Input Operand B
operation_nd	1	Input	Signal to indicate new operands
ce	1	Input	Clock enable
clk	1	Input	Clock signal
result	32	Output	Operation Result (AopB)
rdy	1	Output	Signal to indicate valid result

Πίνακας 6.1: Σημάτα εισόδου-εξόδου Floating-point operator.

Οι FPO, που χρησιμοποιήθηκαν ειναι οι εξής: Πολλαπλασιαστής, Αθροιστής, Αφαιρέτης και Διαίρετης.

Κάθε πολλαπλασιαστής έχει 1 DSP48E (medium usage) και maximum latency 8. Κάθε αθροιστής και αφαιρέτης δεν χρησιμοποιεί κανένα DSP48E (logic only) και έχουν maximum latency 12. Τέλος, ο κάθε διαιρέτης δε χρησιμοποιεί κανένα DSP48E (logic only) και έχει maximum latency 28.

Operator	Maximum latency	Used latency	DSP48E
mult	8	5	1
add	12	6	logic only
sub	12	7	logic only
div	28	14	logic only

Πίνακας 6.2: Τιμές Latency και αριθμός των DSP48Es ανά Floating-point operator.

Στην αρχή της σχεδίασης, οι FPO είχαν τις μέγιστες τιμές latency. Αφού ολοκληρώθηκε το σύστημα και το Synthesis έδωσε την περίοδο του ρολογιού, το latency μειώθηκε σταδιακά. Μετά από κάθε μείωση, γίνοταν synthesize και ελέγχονταν η τιμή του ρολογιού. Ο συνδυασμός των μικρότερων latency που δεν αυξάνει την περίοδο του ρολογιού είναι αυτός του πίνακα (6.2).

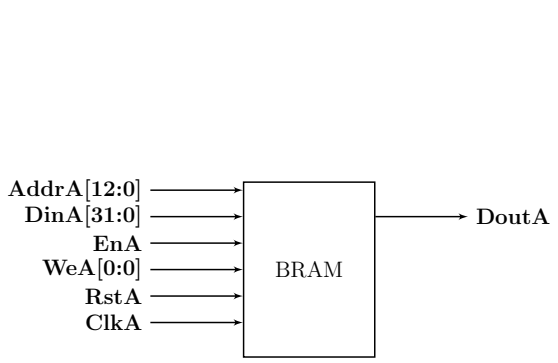
6.2.3 Μνήμη BRAM

Οι μνήμες που χρησιμοποιήθηκαν στην πρώτη αρχιτεκτονική, για την επεξεργασία ενός συστήματος 8192 στοιχείων, είναι οι εξής:

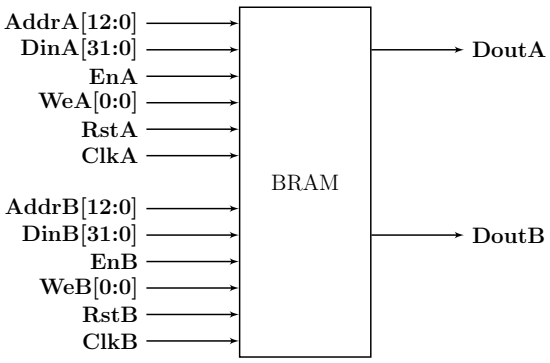
- 3 simple dual port brams με πλάτος 32 bits και βάθος 16384 θέσεις για τα a, c, e
- 1 simple dual port bram με πλάτος 32 bits και βάθος 8192 θέσεις για τα x
- 1 simple port bram με πλάτος 32 bits και βάθος 8192 θέσεις για τα d

Για τη δεύτερη αρχιτεκτονική, οι simple dual port brams αντικαταστάθηκαν με true dual port brams ώστε να διαβάζονται 2 στοιχεία σε ένα κύκλο ρολογιού. Επίσης, το βάθος

των μνημών διαμορφώθηκε ανάλογα με τον αριθμό των cores που έχει η αρχιτεκτονική. Πιο συγκεκριμένα, στα 4 cores οι a, c, e έχουν βάθος 4096 θέσεις, στα 32 cores βάθος 512 και στα 64 έχουν 256 θέσεις. Αντίστοιχα διαιρούνται και οι μνήμες για τα x και d, δηλαδή στα 4 cores έχουν βάθος 2048 θέσεις, στα 32 cores βάθος 256 και στα 64 έχουν 128 θέσεις.



Simple port BRAM



True dual port BRAM

Όλες οι brams δημιουργήθηκαν με το εργαλείο Xilinx Core Generator 13.1. Στον Πίνακα [6.3] παρουσιάζονται τα σήματα εισόδου-εξόδου των true dual port καθώς και η περιγραφή της λειτουργίας τους.

signal	Width(bits)	Type	Description
AddrA	13	Input	Input Address for Port A
AddrB	13	Input	Input Address for Port B
DinA	32	Input	Input Data for Port A
DinB	32	Input	Input Data for Port B
EnA	1	Input	Enable Signal for Port A
EnB	1	Input	Enable Signal for Port B
WeA	1	Input	Write Enable Signal for Port A
WeB	1	Input	Write Enable Signal for Port B
RstA	1	Input	Reset Signal for Port A
RstB	1	Input	Reset Signal for Port B
ClkA	1	Input	Clock Signal for Port A
ClkB	1	Input	Clock Signal for Port B
DoutA	32	Output	Output Data Read from Port A
DoutB	32	Output	Output Data Read from Port B

Πίνακας 6.3: Περιγραφή σημάτων εισόδου-εξόδου των True dual port BRAMs.

6.3 Ταυτοποίηση λειτουργίας

Όλες οι σχεδίασεις προσομοιώθηκαν με το εργαλείο Xilinx ISim 13.1. Αρχικά, προσομοιώθηκαν τα επιμέρους υποσυστημάτα ανά φάση και διαπιστώθηκε η σωστή λειτουργία τους. Έπειτα, έγινε η σύνδεση των υποσυστημάτων και προσομοιώθηκαν οι συνολικές αρχιτεκτονικές.

Για τη διαπίστωση της ορθής λειτουργίας τους έγινε σύγκριση των αποτελεσμάτων με πρόγραμμα που είχε δημιουργήσει ο κος Χατζηπαρασκευάς στα πλαίσια της μεταπτυχιακής του εργασίας [25]. Συγκεκριμένα, είχε υλοποιηθεί σε C++ το σχήμα Crank Nicolson για ένα πλέγμα $N_S \times N_T = 8192 \times 8192$ κόμβων. Οι παράμετροι της εξίσωσης Black-Scholes (3.3.1) είναι:

- $S_{max} = 100$, μέγιστη τιμή υποκείμενου προιόντος
- $K = 5$, τιμή εξάσκησης
- $T = 0.25$, χρόνος μέχρι τη λήξη του ΔΑ
- $r = 0.8$, επιτόκιο
- $\sigma = 0.33$, μεταβλητότητα

Το T και το σ είναι ετησιοποιημένα. $T = 0.25$ σημαίνει ότι η περίοδος μέχρι τη λήξη του ΔΑ είναι 3 μήνες. Το S_{max} είναι ένα ανώτατο όριο για τη διακριτοποίηση του πλέγματος (3.2).

Αυτές οι παράμετροι χρησιμοποιήθηκαν για τον υπολογισμό των συντελεστών a, b, c, e, d των εξισώσεων (3.32)&(3.33), που γίνεται στο software. Η κανονικοποίηση του b, σύμφωνα με την παραλλαγή του odd-even reduction (4), γίνεται και αυτή στο software. Έτσι, τα κανονικοποιημένα a, c, e, d αποτελούν το σύστημα, που μπαίνει ως είσοδος στις υλοποιημένες αρχιτεκτονικές.

6.4 Απόδοση

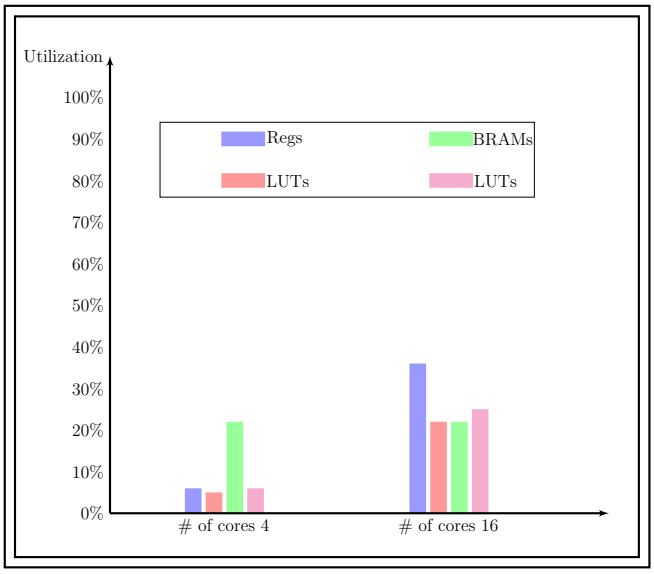
Αρχικά, δημιουργήθηκε σχεδίαση με ένα core. Στη συνέχεια, για την πρώτη εναλλακτική, σχεδιάστηκαν αρχιτεκτονικές με 4 και με 16 cores. Έγινε Synthesis των δύο αρχιτεκτονικών και προέκυψαν τα ακόλουθα αποτελέσματα όσον αφορά τη συχνότητα του ρολογιού και τους πόρους της FPGA.

- 4 cores: 191,205 MHz
- 16 cores 173,070 MHz

Device Utilization Summary					
		Number of cores			
		# 4		# 16	
Resource	Available	Used	Util.	Used	Util.
Slice Regs	207360	12987	6%	75546	36%
Slice LUTs	207360	11523	5%	46230	22%
BRAMs	288	66	22%	66	22%
DSP48Es	192	12	6%	48	25%

Πίνακας 6.4: Κατανάλωση πόρων της FPGA XC5VLX330T

Παρατηρείται μη γραμμική αύξηση των Registers και των LUTs που καταναλώνονται. Αυτό οφείλεται στο bottleneck που έχει η αρχιτεκτονική, όπως αναλύθηκε στο κεφάλαιο 5, όπου με κάθε διπλασιασμό των cores, διπλασιάζεται ο αριθμός των κύκλων που απαιτούνται για την πλήρη λειτουργία όλων των πυρήνων, με αποτέλεσμα το διπλασιασμό και των αντίστοιχων καταστάσεων της FSM. Με αυτή τη σχεδίαση αναμένεται ότι τα συστήματα με περισσότερους πυρήνες (32 ή 64), θα έχουν FSM που θα ξεπερνούν τις 200 καταστάσεις.



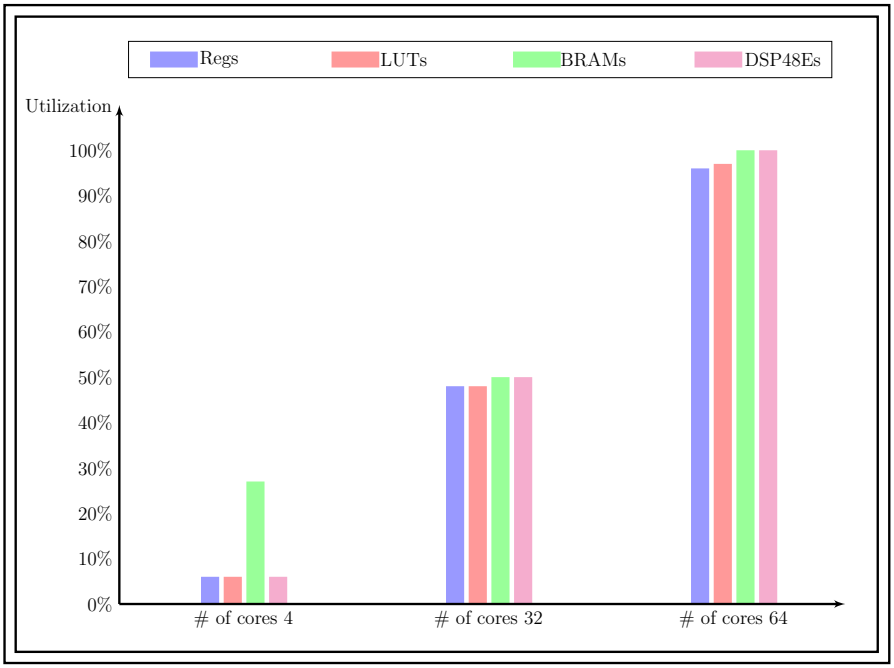
Σχήμα 6.3: Πόροι που χρησιμοποιούνται για την FPGA XC5VLX330T

Για τη δεύτερη εναλλακτική, σχεδιάστηκαν αρχιτεκτονικές με 4, 32 και 64 cores. Τα αποτελέσματα που προέκυψαν μετά το Synthesis όσον αφορά τη συχνότητα του ρολογιού και τους πόρους της FPGA είναι τα ακόλουθα.

- 4 cores: 210,421 MHz
- 32 cores 217,232 MHz
- 64 cores 203.957 MHz

Device Utilization Summary							
		Number of cores					
		# 4		# 32		# 64	
Resource	Available	Used	Util.	Used	Util.	Used	Util.
Slice Regs	207360	12749	6%	99811	48%	199422	96%
Slice LUTs	207360	13985	6%	100993	48%	202882	97%
BRAMs	288	80	27%	144	50%	288	100%
DSP48Es	192	12	6%	96	50%	192	100%

Πίνακας 6.5: Κατανάλωση πόρων της FPGA XC5VLX330T



Σχήμα 6.4: Πόροι που χρησιμοποιούνται για την FPGA XC5VLX330T

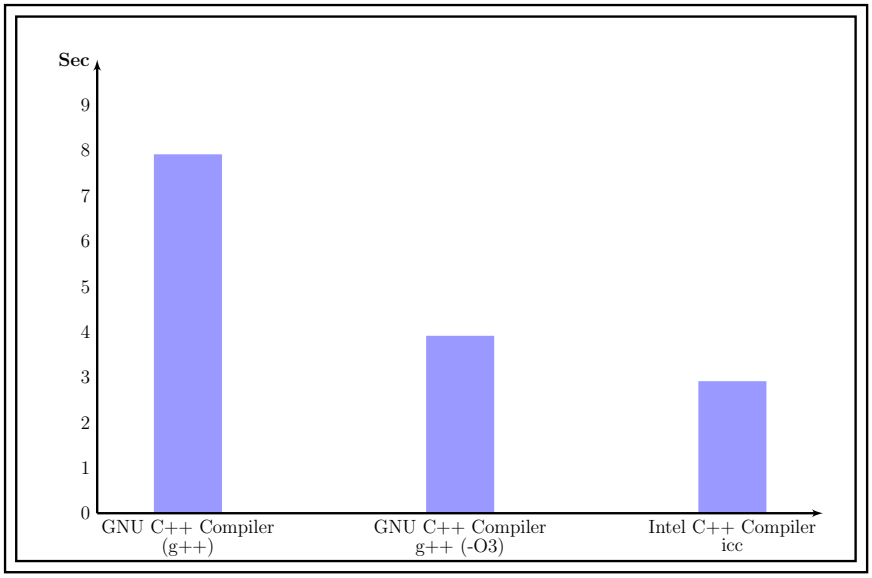
6.5 Αποτελέσματα

Μετρήθηκε η απόδοση του software του κου Χατζηπαρασκευά σε επεξεργαστή Intel Centrino Core Duo συχνότητας 1.60 GHz με μνήμη 1 GB και λειτουργικό σύστημα Ubuntu 10.04. Το compile του κώδικα έγινε με τους **GNU C++ Compiler** (*g++*) και **Intel C++ Compiler** (*icc*). Με τον g++, το compile έγινε και με τις αυτόματες ρυθμίσεις που διαθέτει και με ενεργοποιημένη την παραμέτρο βελτιστοποίησης -O3. Ένα σύστημα μεγέθους 8192 εκτελείται 8192 φορές (αντιστοιχεί στο πλέγμα 8192×8192 κόμβων) και μετριέται αυτός ο χρόνος εκτέλεσης με τη συνάρτηση **clock()**. Τα αποτελέσματα παρουσιάζονται στον πίνακα (6.6) και είναι ο μέσος όρος 10 μετρήσεων για κάθε περίπτωση.

	System Size 8192	# Execution Times 8192	
compiler	GNU C++ (g++)	GNU C++ (-O3)	Intel C++ (icc)
time (sec)	7,93	3,89	2,91

Πίνακας 6.6: Χρόνοι εκτέλεσης με διαφορετικούς compilers

Η παράμετρος -O3 του g++ κάνει διανυσματοποίηση (vectorization) με αποτέλεσμα να μειώνεται ο χρόνος εκτέλεσης 50%. Ο icc διαθέτει αυτόματη διανυσματοποίηση.



Σχήμα 6.5: Χρόνοι εκτέλεσης με διαφορετικούς compilers

Επίσης, για κάθε σύστημα που υλοποιήθηκε, μετρήθηκαν οι απαιτούμενοι κύκλοι ρολογιού που χρειάστηκε για την επεξεργασία των δεδομένων και από το σύνολο των κύκλων υπολογίστηκε ο εκτιμώμενος χρόνος εκτέλεσης.

	System Size 8192	# Execution Times 8192
# cores	4	16
time (sec)	9,8	5,6

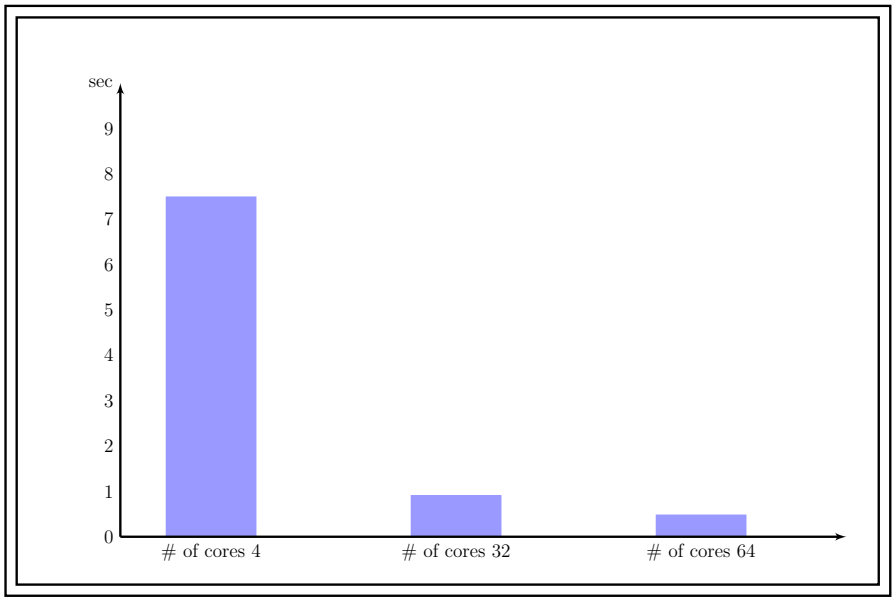
Πίνακας 6.7: Εκτιμώμενοι χρόνοι εκτέλεσης στην XC5VLX330T για την πρώτη αρχιτεκτονική

Παρατηρείται ότι η σχεδίαση των 16 cores είναι μόνο 1,75 φορές πιο γρήγορη από τη σχεδίαση με τα 4 cores, παρά τον τετραπλασιασμό των πυρήνων. Η μείωση του ρολογιού κατά 10% δεν είναι τόσο μεγάλη ώστε να δικαιολογείται τόσο μικρή επιτάχυνση. Αυτό οφείλεται στο μειονέκτημα της συγκεκριμένης αρχιτεκτονικής που αναλύθηκε παραπάνω.

	System Size 8192	# Execution Times 8192	
# cores	4	32	64
time (sec)	7,49	0,91	0,49

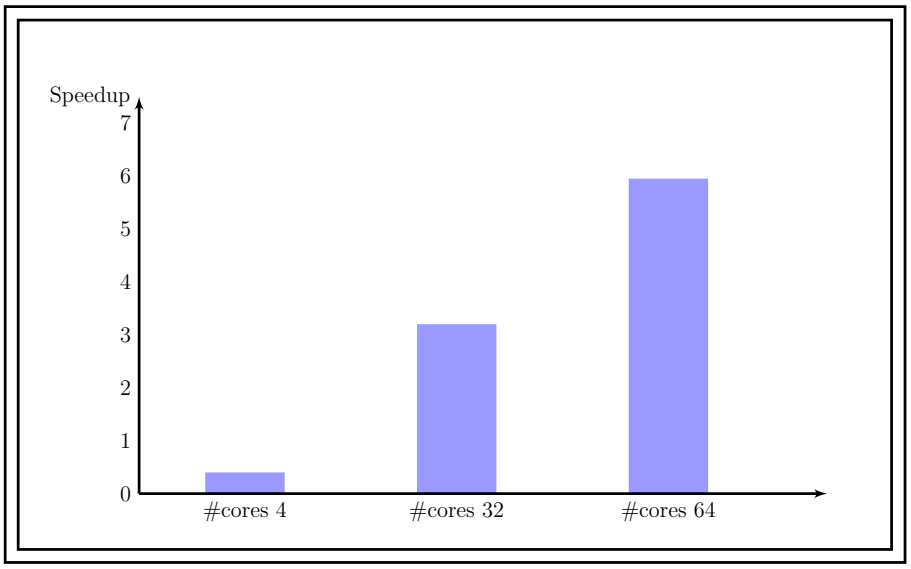
Πίνακας 6.8: Εκτιμώμενοι χρόνοι εκτέλεσης στην XC5VLX330T για τη δεύτερη αρχιτεκτονική

Παρατηρείται ότι η σχεδίαση με 4 πυρήνες της δεύτερης αρχιτεκτονικής είναι πιο γρήγορη από την αντίστοιχη της πρώτης, γεγονός που οφείλεται τόσο στη μεγαλύτερη συχνότητα του ρολογιού όσο και στο νέο τρόπο επικοινωνίας μνημών-cores. Επίσης, στη δεύτερη αρχιτεκτονική έχει επιτευχθεί σχεδόν τέλειο scaling. Συγκεκριμένα, ο αριθμός των πυρήνων διπλασιάζεται ($32 \rightarrow 64$) και η ταχύτητα του συστήματος αυξάνεται 1,8 φορές με μια μικρή μείωση (6%) του ρολογιού. Με μια προβολή στα 128 cores αναμένεται να μειωθεί ο χρόνος εκτέλεσης περίπου στα 0,25 secs.



Σχήμα 6.6: Εκτιμώμενοι χρόνοι εκτέλεσης στην XC5VLX330T για τη δεύτερη αρχιτεκτονική

Στο σχήμα (6.7) φαίνεται η επιτάχυνση που πραγματοποιείται με την FPGA σε σχέση με τον Intel Centrino Core Duo. Για τον υπολογισμό της επιτάχυνσης, χρησιμοποιήθηκε ο καλύτερος χρόνος του software, 2,91 secs (6.6).



Σχήμα 6.7: Επιτάχυνση της δεύτερης αρχιτεκτονικής για το σχήμα Crank-Nicolson

Κεφάλαιο 7

Συμπεράσματα και Μελλοντικές προεκτάσεις

Στόχος της παρούσας διπλωματικής εργασίας, όπως αυτός έχει οριστεί, ήταν η δημιουργία ενός συστήματος αποτίμησης δικαιωμάτων προαίρεσης, σε αναδιατασσόμενη λογική.

Η αρχιτεκτονική που υλοποιήθηκε υπολογίζει την τιμή ενός συμβολαίου ΔΑ Ευρωπαϊκού τύπου με τρόπο πιο αποδοτικό από ένα PC με επεξεργαστή intel core 2 duo 1.6GHz. Αυτή η απόδοση οφείλεται σε ένα μεγάλο μέρος στην κατανάλωση όλων των πόρων της FPGA XC5VLX330T αλλά και στο πολύ καλό scaling που έχει η προτεινόμενη αρχιτεκτονική. Όσο πιο μεγάλη είναι η FPGA, τόσο πιο πολλά cores θα μπορεί να έχει η σχεδίαση και θα μπορεί να επεξεργάζεται όλο και μεγαλύτερα συστήματα.

Η αρχιτεκτονική που παρουσιάστηκε, δημιουργήθηκε αποκλειστικά για την αποτίμηση δικαιωμάτων προαίρεσης Ευρωπαϊκού τύπου. Θα μπορούσε να εξεταστεί η δυνατότητα μετασχηματισμού της μονάδας Calculation ώστε να εκτελεί πράξεις με πλήρη ομοχειρία. Επίσης, μια σημαντική επέκταση θα ήταν η υλοποίηση μιας γενικής σχεδίασης για την αποτίμηση και άλλων τύπων δικαιωμάτων προαίρεσης ώστε να δημιουργηθεί ένα ολοκληρωμένο σύστημα που θα κάνει option pricing.

Βιβλιογραφία

[1] Scott Hauck, André DeHon, *Reconfigurable Computing: The Theory and Practice of FPGA-Based Computation*. Elsevier, 2008.

[2] G.L. Zhang et al., "Reconfigurable acceleration for monte carlo based financial simulation," in *Field-Programmable Technology, 2005. Proceedings. 2005 IEEE International Conference on*, pp. 215 –222, dec. 2005.

[3] D. Thomas, J. Bower, and W. Luk, "Automatic generation and optimisation of reconfigurable financial monte-carlo simulations," in *Application -specific Systems, Architectures and Processors, 2007. ASAP. IEEE International Conf. on*, pp. 168 –173, july 2007.

[4] R. Baxter et al., "Maxwell - a 64 fpga supercomputer," in *Adaptive Hardware and Systems, 2007. AHS 2007. Second NASA/ESA Conference on*, pp. 287 –294, aug. 2007.

[5] X. Tian and K. Benkrid, "Design and implementation of a high performance financial monte-carlo simulation engine on an fpga supercomputer," in *ICECE Technology, 2008. FPT 2008. International Conference on*, pp. 81 –88, dec. 2008.

[6] X. Tian and K. Benkrid, "American option pricing on reconfigurable hardware using least-squares monte carlo method," in *Field-Programmable Technology, 2009. FPT 2009. International Conference on*, pp. 263 –270, dec. 2009.

[7] X. Tian and K. Benkrid, "High-performance quasi-monte carlo financial simulation: Fpga vs. gpp vs. gpu," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 3, pp. 26:1–26:22, Nov. 2010.

[8] A. Kaganov, P. Chow, and A. Lakhany, "Fpga acceleration of monte-carlo based credit derivative pricing," in *Field Programmable Logic and Applications, 2008. FPL 2008. International Conference on*, pp. 329 –334, sept. 2008.

[9] A. Kaganov, A. Lakhany, and P. Chow, "Fpga acceleration of multifactor cdo pricing," *ACM Trans. Reconfigurable Technol. Syst.*, vol. 4, pp. 20:1–20:17, May 2011.

[10] Q. Jin, D. Thomas, W. Luk, and B. Cope, "Exploring reconfigurable architectures for binomial-tree pricing models," in *Reconfigurable Computing: Architectures, Tools and Applications*, pp. 245–255, Springer Berlin / Heidelberg, 2008.

[11] Q. Jin, D. Thomas, and W. Luk, "Exploring reconfigurable architectures for explicit finite difference option pricing models," in *Field Programmable Logic and Applications, 2009. FPL 2009. International Conference on*, pp. 73 –78, sept. 2009.

[12] A. Tse, D. Thomas, and W. Luk, "Option pricing with multi-dimensional quadrature architectures," in *Field-Programmable Technology, 2009. FPT 2009. International Conference on*, pp. 427 –430, dec. 2009.

[13] A. H. Tse, D. B. Thomas, K. H. Tsoi, and W. Luk, "Efficient reconfigurable design for pricing asian options," *SIGARCH Comput. Archit. News*, vol. 38, pp. 14–20, Jan. 2011.

[14] A. Tse, D. Thomas, K. Tsoi, and W. Luk, "Reconfigurable control variate monte-carlo designs for pricing exotic options," in *Field Programmable Logic and Applications (FPL), 2010 International Conference on*, pp. 364 –367, 31 2010-sept. 2 2010.

[15] C. de Schryver, I. Shcherbakov, F. Kienle, N. Wehn, H. Marxen, A. Kostiuk, and R. Korn, "An energy efficient fpga accelerator for monte carlo option pricing with the heston model," in *Reconfigurable Computing and FPGAs (ReConFig), 2011 International Conference on*, pp. 468 –474, 30 2011-dec. 2 2011.

[16] X. Tian and C. Bouganis, "A run-time adaptive fpga architecture for monte carlo simulations," in *Field Programmable Logic and Applications (FPL), 2011 International Conference on*, pp. 116 –122, sept. 2011.

[17] G. C. T. Chow, A. H. T. Tse, Q. Jin, W. Luk, P. H. Leong, and D. B. Thomas, "A mixed precision monte carlo methodology for reconfigurable accelerator systems," in *Proceedings of the ACM/SIGDA international symposium on Field Programmable Gate Arrays*, FPGA '12, (New York, NY, USA), pp. 57–66, ACM, 2012.

[18] Q. Jin, W. Luk, and D. Thomas, "Unifying finite difference option-pricing for hardware acceleration," in *Field Programmable Logic and Applications (FPL), 2011 International Conference on*, pp. 6 –9, sept. 2011.

[19] Q. Jin, W. Luk, and D. Thomas, "On comparing financial option price solvers on fpga," in *Field-Programmable Custom Computing Machines (FCCM), 2011 IEEE 19th Annual International Symposium on*, pp. 89 –92, may 2011.

[20] F. Black and M. Scholes, "The pricing of options and corporate liabilities," *Journal of Political Economy*, vol. 81, pp. 637–654, May - Jun. 1973.

[21] J. C. Cox, S. A. Ross, and M. Rubinstein, "Option pricing: A simplified approach," *Journal of Financial Economics*, vol. 7, no. 3, pp. 229 – 263, 1979.

[22] J. C. Cox and M. Rubinstein, *Options Markets*. Prentice-Hall, 1985.

[23] J. Hull, *Options, Futures and Other Derivatives*. Prentice Hall, 6 ed., 2005.

[24] M. J. Brennan and E. S. Schwartz, "Finite difference methods and jump processes arising in the pricing of contingent claims: A synthesis," *Journal of Financial and Quantitative Analysis*, vol. 13, no. 3, pp. 461–474, 1979.

[25] G. Chatziparaskevas, "A reconfigurable architecture for black-scholes option pricing using finite-difference schemes," Master's thesis, Technical University of Crete, Chania, 2010.

[26] W. F. Ames, *Numerical Methods for Partial Differential Equations*. Academic Press, 3 ed., 1992.

[27] L. N. Trefethen and D. Bau, *Numerical Linear Algebra*. SIAM, 1997.

[28] B. Neta and H.-M. Tai, "Lu factorization on parallel computers," *Computers and Mathematics with Applications*, vol. 11, no. 6, pp. 573 – 579, 1985.

[29] H. S. Stone, "Parallel tridiagonal equation solvers," *ACM Trans. Math. Softw.*, vol. 1, pp. 289–307, Dec. 1975.

[30] A. Chatziparaskevas, G. Brokalakis and I. Papaefstathiou, "An fpga-based parallel processor for black-scholes option pricing using finite differences schemes," in *DATE*, Mar. 2012.

[31] W. Gadner and G. H. Golub, "Cyclic reduction - history and applications," in *Proceedings of the Workshop on Scientifc Computing*, pp. 73 –85, Mar. 1997.

[32] xilinx, "Floating-point operator v5.0," March 2011.