ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ Η/Υ

ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ & ΥΛΙΚΟΥ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

ΑΝΑΔΙΑΤΑΣΣΟΜΕΝΗ ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΓΙΑ ΤΗΝ ΥΛΟΠΟΙΗΣΗ ΤΟΥ ΑΛΓΟΡΙΘΜΟΥ SCAN ΓΙΑ ΣΥΜΠΙΕΣΗ VIDEO

ΣΟΦΙΚΙΤΗΣ ΗΛΙΑΣ

ΕΠΙΒΛΕΠΩΝ ΚΑΘΗΓΗΤΗΣ: ΚΑΘΗΓΗΤΗΣ ΑΠΟΣΤΟΛΟΣ ΔΟΛΛΑΣ

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ: ΚΑΘΗΓΗΤΗΣ Α. ΔΟΛΛΑΣ
ΚΑΘΗΓΗΤΗΣ Κ. ΚΑΛΑΪΤΖΑΚΗΣ
ΑΝ. ΚΑΘΗΓΗΤΗΣ Δ. ΠΝΕΥΜΑΤΙΚΑΤΟΣ

ΣΕΠΤΕΜΒΡΙΟΣ 2004
ΧΑΝΙΑ

Η ολοκλήρωση της διπλωματικής μου εργασίας έγινε με τη βοήθεια και τη συμπαράσταση ορισμένων ανθρώπων, τους οποίους θα ήθελα να ευχαριστήσω. Συγκεκριμένα, ευχαριστώ τον επιβλέποντα καθηγητή κ. Απόστολο Δόλλα για την πολύτιμη βοήθεια, τις συμβουλές και την υπομονή του κατά τη διάρκεια της συνεργασίας μας. Θα ήθελα ακόμα να ευχαριστήσω τον αναπληρωτή καθηγητή κ. Διονύσιο Πνευματικάτο και τον αναπληρωτή καθηγητή κ. Κωνσταντίνο Καλαϊτζάκη που συμμετέχουν στην εξεταστική επιτροπή για την αποδοχή τους στην αξιολόγηση της εργασίας. Επιπλέον, ευχαριστώ τον κ. Ν. Μπουρμπάκη για τους κώδικες που παρείχε. Για τη διεξαγωγή της εργασίας σημαντική ήταν επίσης η βοήθεια του κ. Μάρκου Κιμιωνή. Θα ήθελα ακόμα να ευχαριστήσω τους προπτυχιακούς και τους μεταπτυχιακούς φοιτητές του Εργαστηρίου Μικροεπεξεργαστών και Υλικού για τη στήριξη και τις συμβουλές που μου παρείχαν κατά τη διάρκεια της εκπόνησης της παρούσας εργασίας. Τέλος, θα ήθελα να ευχαριστήσω τους γονείς μου και τον αδερφό μου που μου συμπαραστάθηκαν στην προσπάθεια αυτή.

Με εκτίμηση,

Σοφικίτης Ηλίας

Περιεχόμενα

Κεφάλαιο 1

Εισαγωγή

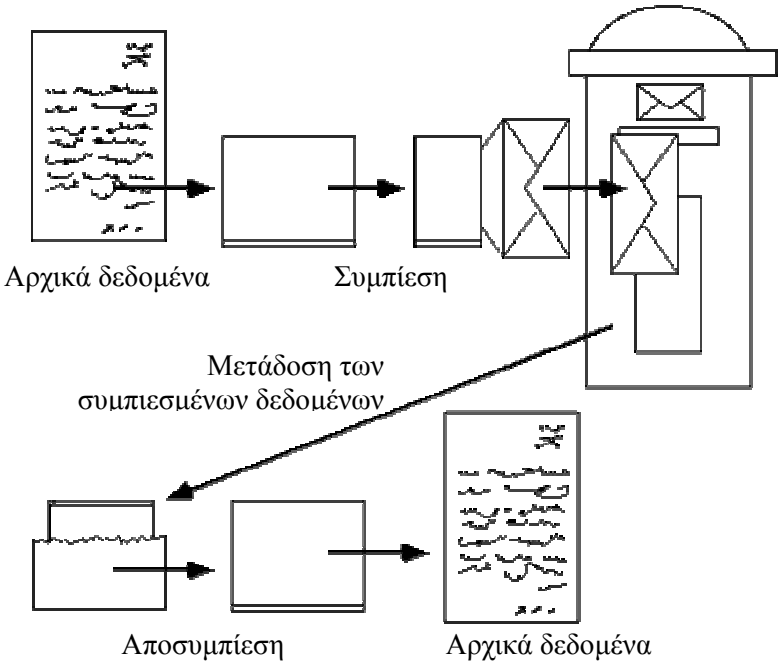
Η ραγδαία τεχνολογική εξέλιξη που έχει συντελεστεί από τα τέλη του 19ου αιώνα μέχρι σήμερα έχει επηρεάσει καταλυτικά την ανθρώπινη ζωή, εφόσον τα επιτεύγματά της αποτελούν αναπόσπαστο κομμάτι της καθημερινότητας. Σημαντικό επίτευγμα αποτέλεσε το 1886 η καταγραφή κινούμενης εικόνας σε φιλμ και η αναπαραγωγή της, καθώς και η μετέπειτα μετάδοσή της (το 1926) που σηματοδότησε την αρχή μιας σειράς εφαρμογών με αντικείμενο την επικοινωνία μέσω κινούμενης εικόνας (video), οι οποίες ακόμα εξελίσσονται.

Από τις πρώτες κιόλας εκπομπές της τηλεόρασης έγινε κατανοητό ότι η μετάδοση video είχε σημαντικά μεγάλες απαιτήσεις σε χωρητικότητα διαύλου συγκριτικά με τις μέχρι τότε διαδεδομένες μεταδόσεις φωνής ή μουσικής, οπότε αμέσως άρχισαν να αναζητούνται μέθοδοι μείωσης αυτής της απαίτησης. Λαμβάνοντας υπόψη τα δεδομένα της εποχής, η ανάπτυξη του συστήματος interlace, όπου η κάθε μεταδιδόμενη εικόνα χωρίζεται σε άρτιες και περιττές γραμμές, οι οποίες μεταδίδονται διαδοχικά και πλέκονται για να δημιουργήσουν την κάθε εικόνα, θεωρήθηκε μια κομψή και πρακτική λύση στο πρόβλημα μια και μείωνε στο μισό την ποσότητα της προς μετάδοση πληροφορίας, αν και σήμερα στέκεται εμπόδιο στην ψηφιακή επεξεργασία του video. Αργότερα, έρευνες έδειξαν ότι το σήμα του video μπορεί να διαχωριστεί σε διαφορετικές ζώνες συχνοτήτων με τα διάφορα τμήματά του να στέλνονται ανεξάρτητα, καταλαμβάνοντας συνολικά μικρότερη χωρητικότητα από αυτή του αρχικού σήματος.

Με την ανάπτυξη της ψηφιακής τεχνολογίας νέες εφαρμογές που σχετίζονται με την επικοινωνία μέσω video εμφανίστηκαν, όπως η μετάδοση video σε πραγματικό χρόνο μέσω του διαδικτύου, οι τηλεδιασκέψεις, η τηλεϊατρική, οι εικονικές τάξεις, η μετάδοση video μέσω κινητών τηλεφώνων και η τηλεόραση υψηλής ευκρίνειας. Η γρήγορη διάδοσή των εφαρμογών αυτών, η δυνατότητα πρόσβασης μεγάλου αριθμού χρηστών σε αυτές και η προβλεπόμενη μελλοντική περαιτέρω ανάπτυξή τους καθιστούν επιτακτική την ανάγκη εύρεσης τρόπων αποδοτικότερης λειτουργίας τους.

Για τη μετάδοση και αποθήκευση video απαιτείται διαχείριση τεράστιας ποσότητας δεδομένων και παρά την πρόοδο στις τεχνολογίες και τις ταχύτητες των σύγχρονων δικτύων και την αύξηση της χωρητικότητας και της ταχύτητας διαμεταγωγής των συσκευών αποθήκευσης, είναι απαραίτητη η συμπίεση των δεδομένων.

Με τον όρο συμπίεση εννοούμε τη μετατροπή δεδομένων σε μια μορφή που απαιτεί λιγότερα bits, έτσι ώστε τα δεδομένα να μπορούν να αποθηκευτούν ή να μεταδοθούν πιο αποδοτικά. Ο λόγος του μεγέθους των αρχικών δεδομένων (O) προς το μέγεθος των δεδομένων σε συμπιεσμένη μορφή (C) είναι γνωστός ως λόγος συμπίεσης (compression ratio, R=O/C). Αν η ανάστροφη διαδικασία, η αποσυμπίεση, παράγει ένα ακριβές αντίγραφο των αρχικών δεδομένων, τότε η συμπίεση λέγεται μη απωλεστική, ενώ αν παράγει προσεγγιστικά τα αρχικά δεδομένα λέγεται απωλεστική. Η μη απωλεστική συμπίεση ενδείκνυται για κείμενο ή δυαδικά δεδομένα, ενώ η απωλεστική για ήχο, εικόνες και video. Γενικά με την απωλεστική συμπίεση επιτυγχάνονται μεγαλύτεροι λόγοι συμπίεσης από ότι με τη μη απωλεστική, ενώ η πιστότητα αποσυμπιεσμένων δεδομένων γίνεται υψηλότερη όσο ο λόγος συμπίεσης μειώνεται. Ένας παραλληλισμός της διαδικασίας συμπίεσης και αποσυμπίεσης (μη απωλεστικής) με το κλασικό ταχυδρομείο φαίνεται στο Σχήμα 1.1.



Σχήμα 1.1 Η συμπίεση είναι ανάλογη με το δίπλωμα ενός γράμματος πριν τοποθετηθεί σε φάκελο, ώστε να μπορεί να μεταφερθεί πιο εύκολα και οικονομικά. Τα συμπιεσμένα δεδομένα, σαν το κλεισμένο στο φάκελο γράμμα είναι δύσκολο να διαβαστούν και πρέπει πρώτα να αποσυμπιεστούν, ή να βγουν από το φάκελο, ώστε να επανέλθουν στην αρχική τους μορφή

Το επιθυμητό για ένα σύστημα συμπίεσης video είναι να επιτυγχάνει όσο το δυνατόν καλύτερη ποιότητα εικόνας και μεγαλύτερη συμπίεση των δεδομένων. Για να επιτευχθεί αυτό γίνεται εκμετάλλευση χωρικών, χρονικών και στατιστικών πλεονασμών. Όμως, όπως έχει ήδη καταστεί σαφές, τα δύο αυτά μεγέθη είναι αντικρουόμενα, οπότε ανάλογα με την εφαρμογή για την οποία προορίζεται το σύστημα, μπορεί να δίνεται παραπάνω έμφαση στην ποιότητα (π.χ. αποθήκευση κινηματογραφικής ταινίας σε οπτικό δίσκο) ή στη συμπίεση (π.χ. αποθήκευση video για μετάδοση μέσω διαδικτύου). Η τεχνολογία της συμπίεσης ψηφιακού video σήμερα προσφέρει αρκετά προϊόντα, τα οποία είτε είναι υλοποιήσεις hardware είτε software. Παρ' όλ' αυτά, η συμπίεση video παραμένει ένα προκλητικό ερευνητικό πεδίο κι ο σχεδιασμός αποδοτικών αλγορίθμων κωδικοποίησης συνεχίζει να είναι γόνιμη περιοχή έρευνας.

Η διπλωματική αυτή εργασία έγινε με σκοπό το σχεδιασμό και την υλοποίηση αναδιατασσόμενης αρχιτεκτονικής για τον αλγόριθμο SCAN [27, 28, 29, 30], ο οποίος χρησιμοποιείται για συμπίεση video, σε γλώσσα VHDL (Very high speed integrated circuits Hardware Description Language). Ο αλγόριθμος αυτός εκμεταλλεύεται τη συσχέτιση των διαδοχικών frames, δηλαδή τον χρονικό πλεονασμό που παρατηρείται στα δεδομένα του video και είναι σχεδιασμένος για να συμπιέζει video που στη συνέχεια πρόκειται να κρυπτογραφηθεί από το αντίστοιχο σύστημα. Το γεγονός αυτό προσδίδει μια ιδιαίτερη σημασία στον αλγόριθμο, διότι η κρυπτογράφηση στη σημερινή εποχή είναι περισσότερο αναγκαία από ποτέ για την εξασφάλιση του απορρήτου των επικοινωνιών. Η αποσυμπίεση πραγματοποιείται μετά και την αποκρυπτογράφηση των δεδομένων. Ο αλγόριθμος έχει ήδη υλοποιηθεί σε software, αλλά η hardware υλοποίησή του προσφέρει μεγαλύτερη ταχύτητα στη συμπίεση και στην αποσυμπίεση.

Η διπλωματική αυτή εργασία αποτελείται από έξι κεφάλαια. Στο παρόν κεφάλαιο περιγράφηκε η έννοια της συμπίεσης, η σημασία της συμπίεσης video και η συνεισφορά της συγκεκριμένης εργασίας στο πεδίο αυτό. Στο δεύτερο κεφάλαιο γίνεται μια αναφορά στις υπάρχουσες τεχνικές συμπίεσης video που είναι υλοποιημένες είτε σε software είτε σε hardware. Μια γενική περιγραφή της software υλοποίησης του αλγορίθμου SCAN ακολουθεί στο τρίτο κεφάλαιο. Στο τέταρτο κεφάλαιο περιγράφεται η αρχιτεκτονική η οποία σχεδιάστηκε για την υλοποίηση του αλγορίθμου σε αναδιατασσόμενη λογική και το μοντέλο του αλγορίθμου σε γλώσσα

VHDL. Στο πέμπτο κεφάλαιο γίνεται η δοκιμή και η πιστοποίηση της λειτουργίας του συστήματος. Τέλος, στο έκτο κεφάλαιο παρουσιάζονται τα συμπεράσματα της εργασίας και οι μελλοντικές επεκτάσεις που μπορεί να έχει το σύστημα.

Κεφάλαιο 2

Σχετική έρευνα

2.1 Εισαγωγή

Από τότε που τα δεδομένα video είτε αποθηκεύονται σε συσκευές αποθήκευσης, όπως CD και DVD, είτε μεταδίδονται μέσω ενός δικτύου επικοινωνίας, το μέγεθος των δεδομένων του ψηφιακού video αποτελεί σημαντικό ζήτημα στην τεχνολογία πολυμέσων. Λόγω των μεγάλων απαιτήσεων εύρους ζώνης των σημάτων video, μια εφαρμογή video που τρέχει σε οποιαδήποτε πλατφόρμα δικτύου μπορεί να υπερπληρώσει τους πόρους του εύρους ζώνης του μέσου επικοινωνίας, εάν τα καρέ του video μεταφέρονται σε ασυμπίεστη μορφή. Για παράδειγμα, ας υποτεθεί ότι ένα καρέ video ψηφιοποιείται σε μορφή διακριτών καννάβων (grids) εικονοστοιχείων με ανάλυση 176 εικονοστοιχεία ανά γραμμή και 144 γραμμές ανά εικόνα. Αν θεωρηθεί ότι η εικόνα είναι σε κλίμακα του γκρι, όταν κάθε τιμή φωτεινότητας αναπαριστάται με ακρίβεια 8 bit, τότε κάθε καρέ του video χρειάζεται περίπου 25 Kbytes για να απεικονίσει το περιεχόμενό του. Εάν τα καρέ του video μεταδίδονται χωρίς συμπίεση με ρυθμό 25 καρέ ανά δευτερόλεπτο, τότε ο ρυθμός των δεδομένων για ακολουθία video είναι περίπου 5,1 Mbps κι ένα video διάρκειας ενός λεπτού θα απαιτεί 38 Mbytes εύρους ζώνης. Για ανάλυση CIF (Common Intermediate Format) 352 × 288, με προσέγγιση 8 bit για κάθε τιμή φωτεινότητας, κάθε εικόνα θα χρειαστεί 101 Kbytes μνήμης για αναπαράσταση του ψηφιακού της περιεχομένου. Για τον ίδιο αριθμό καρέ ανά δευτερόλεπτο, ο ρυθμός των δεδομένων video για την ακολουθία είναι σχεδόν 20,3 Mbps κι ένα video διάρκειας ενός λεπτού θα απαιτεί πάνω από 152 Mbytes εύρους ζώνης. Συνεπώς, τα δεδομένα ψηφιακού video πρέπει να συμπιέζονται πριν από τη μετάδοσή τους με σκοπό να βελτιστοποιηθεί το απαιτούμενο εύρος ζώνης για τη διάθεση των υπηρεσιών πολυμέσων.

Εφόσον η ψηφιακή απεικόνιση σημάτων video απαιτεί μεγάλη χωρητικότητα, αλγόριθμοι κωδικοποίησης video χαμηλής πολυπλοκότητας πρέπει να καθοριστούν για να συμπιέζουν αποδοτικά ακολουθίες video για λόγους αποθήκευσης ή διαμεταγωγής. Η κατάλληλη επιλογή ενός αλγορίθμου κωδικοποίησης video για

εφαρμογές πολυμέσων αποτελεί ένα σημαντικό παράγοντα, ο οποίος σε κανονικές συνθήκες εξαρτάται από τη διαθεσιμότητα του εύρους ζώνης και την ελάχιστη απαιτούμενη ποιότητα. Σε αυτό το κεφάλαιο παρέχεται μια εποπτεία των πιο δημοφιλών τεχνικών κωδικοποίησης video [1] κι εξηγούνται κάποιες κύριες λεπτομέρειες των σύγχρονων προτύπων κωδικοποίησης video. Επιπρόσθετα, παρουσιάζονται κάποιες υλοποιήσεις αλγορίθμων συμπίεσης video σε hardware.

2.2 Μη απωλεστική και απωλεστική συμπίεση

2.2.1 Μη απωλεστική συμπίεση

Σε πολλές εφαρμογές ο αποκωδικοποιητής πρέπει να ανακατασκευάσει τα κωδικοποιημένα δεδομένα χωρίς καμία απώλεια των αρχικών. Για μία διεργασία μη απωλεστικής συμπίεσης τα ανακατασκευασμένα δεδομένα και τα αρχικά δεδομένα πρέπει να είναι ακριβώς ίδια. Αυτό επίσης αναφέρεται ως αντιστρεπτή διεργασία. Στη μη απωλεστική συμπίεση για μια ειδική εφαρμογή η επιλογή μιας μεθόδου συμπίεσης περιλαμβάνει συνδυασμό τριών παραγόντων: αποδοτικότητα κωδικοποίησης, πολυπλοκότητα κωδικοποίησης και καθυστέρηση κωδικοποίησης.

- Αποδοτικότητα κωδικοποίησης: Αυτή συνήθως μετρείται σε bits ανά δείγμα ή bits ανά δευτερόλεπτο (bps). Η αποδοτικότητα κωδικοποίησης συνήθως περιορίζεται από το περιεχόμενο της πληροφορίας.
- Πολυπλοκότητα κωδικοποίησης: Η πολυπλοκότητα μιας διεργασίας συμπίεσης είναι ανάλογη με την υπολογιστική επίδοση που απαιτείται για την υλοποίηση των λειτουργιών κωδικοποίησης κι αποκωδικοποίησης. Η υπολογιστική επίδοση συνήθως μετρείται με απαιτήσεις μνήμης και πλήθος αριθμητικών λειτουργιών.
- Καθυστέρηση κωδικοποίησης: Μία σύνθετη διεργασία συμπίεσης συχνά οδηγεί σε αυξημένες καθυστερήσεις κωδικοποίησης στον κωδικοποιητή και στον αποκωδικοποιητή. Καθυστερήσεις κωδικοποίησης μπορούν να μετριαστούν αυξάνοντας την επεξεργαστική ισχύ της υπολογιστικής μηχανής. Ωστόσο, αυτό μπορεί να μην είναι πρακτικό σε περιβάλλοντα που υπάρχει

περιορισμός ισχύος ή όπου η υπολογιστική μηχανή δε μπορεί να βελτιωθεί. Επιπρόσθετα, σε πολλές εφαρμογές καθυστερήσεις κωδικοποίησης πρέπει να περιοριστούν, όπως για παράδειγμα σε επικοινωνίες πραγματικού χρόνου. Η ανάγκη περιορισμού της καθυστέρησης κωδικοποίησης συχνά αναγκάζει το σχεδιαστή του συστήματος συμπίεσης να χρησιμοποιήσει ένα λιγότερο πολύπλοκο αλγόριθμο για τις διεργασίες συμπίεσης.

2.2.2 Απωλεστική συμπίεση

Η πλειονότητα των εφαρμογών επεξεργασίας δεδομένων εικόνας ή video δεν απαιτεί να είναι ακριβώς ίδια τα ανακατασκευασμένα δεδομένα με τα αρχικά, γι' αυτό κάποιο ποσό απώλειας είναι επιτρεπτό σε αυτά. Μια διεργασία συμπίεσης που καταλήγει σε μια μη τέλεια ανακατασκευή αναφέρεται σαν απωλεστική διεργασία συμπίεσης και είναι μη αντιστρεπτή. Αυτές οι διεργασίες συμπίεσης υποβαθμίζουν ταχύτατα την ποιότητα του σήματος όταν εφαρμόζονται επανειλημμένα σε προηγουμένως ανακατασκευασμένα δεδομένα.

Η επιλογή συγκεκριμένης μεθόδου συμπίεσης περιλαμβάνει συνδυασμό των τριών παραγόντων που αναφέρθηκαν και στη μη απωλεστική συμπίεση, καθώς και σε αυτόν της ποιότητας του σήματος. Λόγω του επιπρόσθετου βαθμού ελευθερίας, η διεργασία απωλεστικής συμπίεσης πετυχαίνει υψηλότερους λόγους συμπίεσης από μια τεχνική μη απωλεστικής συμπίεσης.

- Ποιότητα σήματος: Ο όρος συχνά χρησιμοποιείται για να χαρακτηρίσει το σήμα στην έξοδο του αποκωδικοποιητή. Δεν υπάρχει καθολικώς αποδεκτή μέτρηση για την ποιότητα του σήματος. Μια μέτρηση που συχνά χρησιμοποιείται είναι ο σηματοθορυβικός λόγος (signal to noise ratio-SNR). Υψηλό SNR δεν αντιστοιχεί πάντα σε σήματα με υψηλή ποιότητα. Μια άλλη μέτρηση της ποιότητας σήματος είναι η βαθμολογία της μέσης άποψης, όπου η απόδοση μιας διεργασίας συμπίεσης χαρακτηρίζεται από υποκειμενική ποιότητα του αποκωδικοποιημένου σήματος. Για παράδειγμα, μια κλίμακα με πέντε βαθμώσεις, όπως «πολύ ενοχλητικό», «ενοχλητικό», «ελαφρά ενοχλητικό», «αντιληπτό αλλά όχι ενοχλητικό» και «μη αντιληπτό» μπορεί να

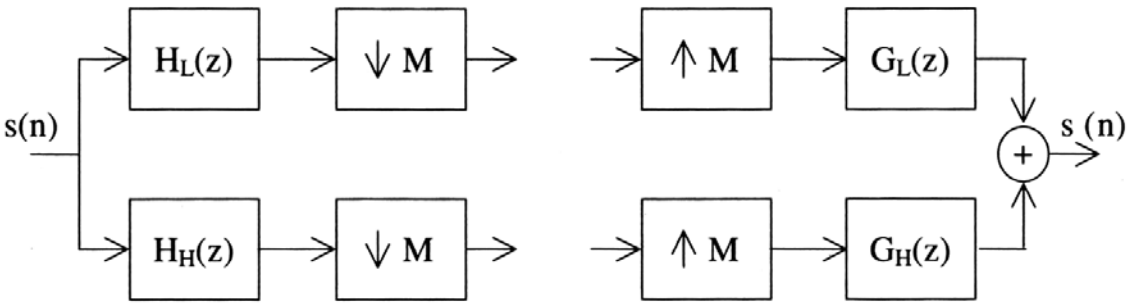
χρησιμοποιηθεί για να χαρακτηρίσει τις απώλειες στην έξοδο του αποκωδικοποιητή.

2.3 Βασικές τεχνικές συμπίεσης video

2.3.1 Κωδικοποίηση υποζώνης (Sub-band Coding)

Η κωδικοποίηση υποζώνης (Sub-band Coding) [2] είναι μία μορφή αποσύνθεσης της συχνότητας. Το σήμα του video αποσυντίθεται σε έναν αριθμό ζωνών συχνοτήτων χρησιμοποιώντας μια συστοιχία φίλτρων. Οι υψηλής συχνότητας συνιστώσες του σήματος συνήθως συνεισφέρουν σε μικρό ποσοστό στην ποιότητα του video και γι' αυτό είτε μπορούν να αποκοπούν είτε να ποσοτικοποιηθούν χονδρικώς. Ακολουθώντας τη διεργασία φιλτραρίσματος, οι συντελεστές που περιγράφουν τις καταληκτικές ζώνες συχνότητας μετασχηματίζονται και ποσοτικοποιούνται ανάλογα με τη σημασία τους και τη συνεισφορά τους στην ανακατασκευασμένη ποιότητα του video. Στον αποκωδικοποιητή σήματα υποζώνης υπερδειγματοληπτούνται με εισαγωγή μηδενικών, φιλτράρονται και αποπολυπλέκονται, ώστε να αποκατασταθεί το αρχικό σήμα του video. Το Σχήμα 2.1 δείχνει μια βασική δομή δύο καναλιών φιλτραρίσματος για κωδικοποίηση υποζώνης.

Εφόσον κάθε καρέ του video που εισάγεται είναι δισδιάστατος πίνακας εικονοστοιχείων, ο κωδικοποιητής υποζώνης τον επεξεργάζεται σε δυο διαστάσεις. Γι' αυτό, όταν το καρέ χωρίζεται σε δυο ζώνες οριζοντίως και καθέτως, αντίστοιχα, παίρνονται τέσσερις ζώνες: χαμηλή-χαμηλή, χαμηλή-υψηλή, υψηλή-χαμηλή και υψηλή-υψηλή. Τότε, ο μετασχηματισμός DCT εφαρμόζεται στη χαμηλότερη

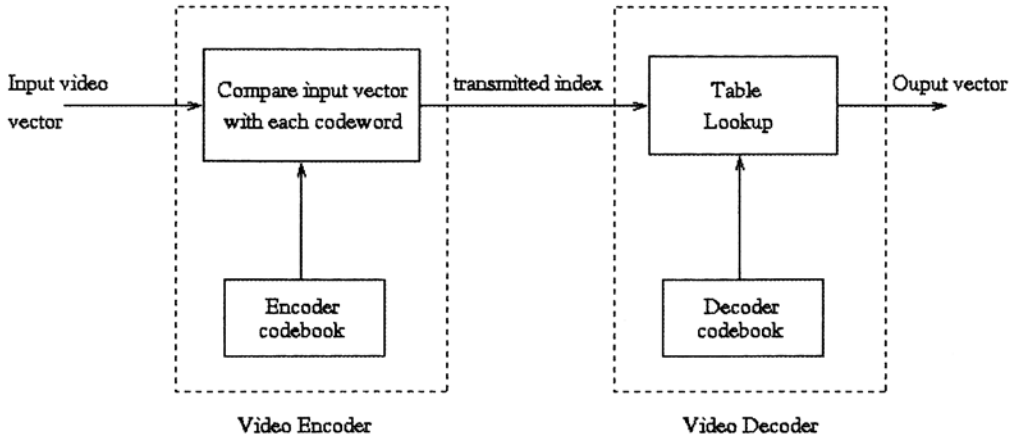


Σχήμα 2.1 Βασική δομή δύο καναλιών φιλτραρίσματος για κωδικοποίηση υποζώνης

υποζώνη κι ακολουθεί ποσοτικοποίηση (quantisation) και κωδικοποίηση run-length. Οι εναπομένουσες υποζώνες ποσοτικοποιούνται χονδρικώς. Ο μετασχηματισμός DCT, η ποσοτικοποίηση και η κωδικοποίηση run-length αναλύονται στην παράγραφο 2.3.3.

2.3.2 Κωδικοποίηση βασισμένη σε διανύσματα (Vector-based Coding)

Ένα διάνυσμα σε video μπορεί να συντεθεί από λάθη πρόβλεψης, συντελεστές μετατροπής ή δείγματα υποζώνης. Η έννοια της κωδικοποίησης διανύσματος [3, 4] αποτελείται από αναγνώριση ενός διανύσματος σε ένα καρέ του video κι αναπαράστασή του από ένα στοιχείο ενός codebook που βασίζεται σε κάποια κριτήρια, όπως η ελάχιστη απόσταση, ελάχιστο bit rate ή ελάχιστο μέσο τετραγωνικό σφάλμα (mean square error-MSE). Όταν αναγνωρίζεται η καλύτερα ταιριασμένη στο codebook είσοδος, ο αντίστοιχος δείκτης στέλνεται στον αποκωδικοποιητή. Χρησιμοποιώντας αυτόν τον δείκτη ο αποκωδικοποιητής μπορεί να ανακτήσει τον κωδικό του διανύσματος από το δικό του codebook που είναι ίδιος με αυτόν που χρησιμοποιείται από τον κωδικοποιητή. Γι' αυτό, η σχεδίαση του codebook αποτελεί το πιο σημαντικό τμήμα της τεχνικής κωδικοποίησης video που είναι βασισμένη σε διανύσματα. Το codebook φυσιολογικά μεταδίδεται στον αποκωδικοποιητή εκτός ζώνης από τη διαμεταγωγή των δεδομένων, π.χ. χρησιμοποιώντας ένα ξεχωριστό τμήμα του διαθέσιμου εύρους ζώνης. Σε δυναμικές δομές codebook το να αναβαθμίζεις το codebook του αποκωδικοποιητή γίνεται ένας μάλλον ενδιαφέρων παράγοντας του συστήματος κωδικοποίησης κι οδηγεί στην αναγκαιότητα η αναβάθμιση των codebooks να γίνει περιοδική διεργασία. Ο αποκωδικοποιητής χρησιμοποιεί τον λαμβανόμενο δείκτη για να βρει αντίστοιχο διάνυσμα στο codebook κι ανακατασκευάζει το block. Το Σχήμα 2.2 απεικονίζει το block διάγραμμα μιας τεχνικής κωδικοποίησης διανύσματος. Το εξερχόμενο bit rate ενός κωδικοποιητή video βασισμένου σε διανύσματα μπορεί να ελεγχθεί από τις παραμέτρους σχεδιασμού του codebook. Το μέγεθος M του codebook (αριθμός διανυσμάτων) και η διάσταση του διανύσματος K (αριθμός των bits ανά διάνυσμα) είναι οι κύριοι παράγοντες που επηρεάζουν το bit rate. Ωστόσο, αυξανόμενο M συνεπάγεται κάποιες δυσκολίες στην ποσοτικοποίηση, όπως μεγάλες απαιτήσεις αποθήκευσης και



Σχήμα 2.2 Block διάγραμμα της βασισμένης σε διανύσματα τεχνικής κωδικοποίησης video

προστιθέμενη πολυπλοκότητα αναζήτησης. Για σκοπούς βελτιστοποίησης ποιότητας / ρυθμού, τα διανύσματα στο codebook είναι κωδικοποιημένα με μεταβλητό μήκος.

2.3.3 Κωδικοποίηση βασισμένη σε συγκροτήματα (Block-based Coding)

Σε τεχνικές κωδικοποίησης video βασισμένες σε συγκροτήματα, κάθε καρέ του video χωρίζεται σε έναν αριθμό πινάκων 16 × 16 ή σε συγκροτήματα (blocks) εικονοστοιχείων που ονομάζονται μακροσυγκροτήματα (macroblocks - MBs). Σε κωδικοποιητές video βασισμένους σε συγκροτήματα δύο μέθοδοι κωδικοποίησης υπάρχουν, συγκεκριμένα οι μέθοδοι INTER και INTRA. Στη μέθοδο INTRA κάθε καρέ του video κωδικοποιείται σαν ανεξάρτητη ακίνητη εικόνα χωρίς αναφορά σε προηγούμενα καρέ. Γι' αυτό, ο μετασχηματισμός DCT και η ποσοτικοποίηση των μετασχηματισμένων συντελεστών εφαρμόζονται μόνο για να αποκρύψουν τους χωρικούς πλεονασμούς ενός καρέ του video. Αντιθέτως, η μέθοδος κωδικοποίησης INTER πετυχαίνει μεγαλύτερη συμπίεση χρησιμοποιώντας προγνωστική κωδικοποίηση. Πρώτα εκτελείται μια έρευνα κίνησης για να καθοριστούν οι ομοιότητες ανάμεσα στο τρέχον καρέ και στο καρέ αναφοράς. Τότε, η εικόνα των διαφορών, γνωστή σαν το υπολειπόμενο λάθος καρέ, υφίσταται μετασχηματισμό DCT και ποσοτικοποιείται. Ο καταληκτικός υπολειπόμενος πίνακας μετατρέπεται ακολούθως σε ένα μονοδιάστατο πίνακα συντελεστών χρησιμοποιώντας την κωδικοποίηση μορφής zigzag με σκοπό να εκμεταλλευτεί τις μεγάλες ακολουθίες μηδενικών που εμφανίζονται στην εικόνα μετά την ποσοτικοποίηση.

Δεδομένης της ποικιλίας των τεχνικών κωδικοποίησης video που είναι διαθέσιμες σήμερα, η επιλογή του κατάλληλου αλγορίθμου κωδικοποίησης για μια συγκεκριμένη υπηρεσία πολυμέσων θεωρείται κρίσιμο ζήτημα. Με τη σύντομη παρουσίαση των τεχνικών κωδικοποίησης video που έγινε πρωτύτερα προκύπτει ευθέως το συμπέρασμα ότι η επιλογή του κατάλληλου κωδικοποιητή video εξαρτάται από τη σχετική εφαρμογή και τις διαθέσιμες πηγές. Οι κωδικοποιητές video που είναι βασισμένοι σε συγκροτήματα φαίνεται να είναι πιο δημοφιλείς στις υπηρεσίες πολυμέσων που είναι διαθέσιμες σήμερα. Επιπλέον, τα περισσότερα σύγχρονα πρότυπα κωδικοποίησης video βασίζονται στο μετασχηματισμό DCT των 16 × 16 συγκροτημάτων εικονοστοιχείων, δηλαδή στη βασισμένη σε συγκροτήματα δομή τους. Ο πρωταρχικός λόγος για την επιτυχία που είχαν οι κωδικοποιητές αυτοί είναι η ποιότητα υπηρεσίας που είναι σχεδιασμένοι να πετυχαίνουν. Η ποιότητα του video μπορεί να είναι αρκετά ικανοποιητική ακόμα και σε χαμηλά bit rates. Επιπρόσθετα, οι κωδικοποιητές αυτού του είδους είναι κατάλληλοι για χρήση σε συστήματα πραγματικού χρόνου.

2.3.3.1 Κωδικοποίηση INTER και INTRA

Δύο διαφορετικοί τύποι κωδικοποίησης υπάρχουν σε έναν κωδικοποιητή video που μετασχηματίζει συγκροτήματα, συγκεκριμένα οι μέθοδοι κωδικοποίησης INTER και INTRA. Σε μια ακολουθία video συνεχόμενα καρέ μπορούν να συσχετιστούν σε μεγάλο βαθμό. Αυτή η χρονική συσχέτιση μπορεί να εκμεταλλευτεί για να επιτευχθούν μεγαλύτερες συμπιέσεις. Η εκμετάλλευση της συσχέτισης μπορεί να γίνει με κωδικοποίηση μόνο των διαφορών μεταξύ του τρέχοντος καρέ και του καρέ αναφοράς. Στις περισσότερες περιπτώσεις το καρέ αναφοράς που χρησιμοποιείται για πρόγνωση είναι το προηγούμενο καρέ στην ακολουθία. Η καταληκτική εικόνα των διαφορών λέγεται υπολειπόμενη εικόνα ή λάθος πρόγνωσης. Αυτός ο τρόπος κωδικοποίησης ονομάζεται κωδικοποίηση INTER καρέ (INTER frame) ή κωδικοποίηση καρέ πρόγνωσης (P-frame, predicted frame). Ωστόσο, αν διαδοχικά καρέ δε συσχετίζονται σε μεγάλο βαθμό λόγω μεταβαλλόμενων σκηνών ή γρήγορων λήψεων της κάμερας, η κωδικοποίηση INTER δε θα επιτύχει αποδεκτή ποιότητα ανακατασκευής. Σε αυτήν την περίπτωση η ποιότητα θα ήταν πολύ καλύτερη αν η πρόγνωση δεν είχε χρησιμοποιηθεί. Εναλλακτικά, το καρέ κωδικοποιείται χωρίς

αναφορά σε πληροφορία του video από τα προηγούμενα καρέ. Αυτή η μέθοδος κωδικοποίησης αναφέρεται σαν κωδικοποίηση INTRA καρέ (I-frame, INTRA frame). Η INTRA μεταχειρίζεται ένα καρέ του video σαν μια ακίνητη εικόνα χωρίς να χρησιμοποιεί καμία χρονική πρόγνωση. Στη μέθοδο κωδικοποίησης INTRA frame όλα τα μακροσυγκροτήματα ενός καρέ κωδικοποιούνται με INTRA. Ωστόσο, στην κωδικοποίηση INTER frame κάποια μακροσυγκροτήματα μπορούν να κωδικοποιηθούν με INTRA εάν το κατώφλι ενεργότητας κίνησης δεν έχει αποκτηθεί. Γι' αυτό το λόγο είναι απαραίτητο σε αυτήν την περίπτωση το κάθε μακροσυγκρότημα να κωδικοποιεί κι ένα σήμα που να δείχνει αν είναι κωδικοποιημένο με INTER ή INTRA. Αν και τα INTER frames επιτυγχάνουν υψηλούς λόγους συμπίεσης, μία συσσώρευση των καρέ που είναι κωδικοποιημένα με INTER θα μπορούσε να οδηγήσει σε μειωμένη ποιότητα εικόνας λόγω της επίδρασης της επαναλαμβανόμενης ποσοτικοποίησης. Γι' αυτό ένα INTRA frame θα μπορούσε να χρησιμοποιηθεί για να ανανεώσει την ποιότητα της εικόνας μετά από έναν καθορισμένο αριθμό καρέ που θα έχουν κωδικοποιηθεί με INTER.

Εκτός από τα I-frames και τα P-frames, υπάρχουν και τα B-frames (Bidirectional frames), που κωδικοποιούνται χρησιμοποιώντας δύο frames αναφοράς, ένα προγενέστερο κι ένα μεταγενέστερο, τα οποία μπορεί να είναι είτε I-frames είτε P-frames. Η συμπίεση που επιτυγχάνουν αυτά είναι πολύ μεγάλη και χρησιμοποιούνται στο πρότυπο MPEG.

2.3.3.2 Εκτίμηση κίνησης (Motion estimation)

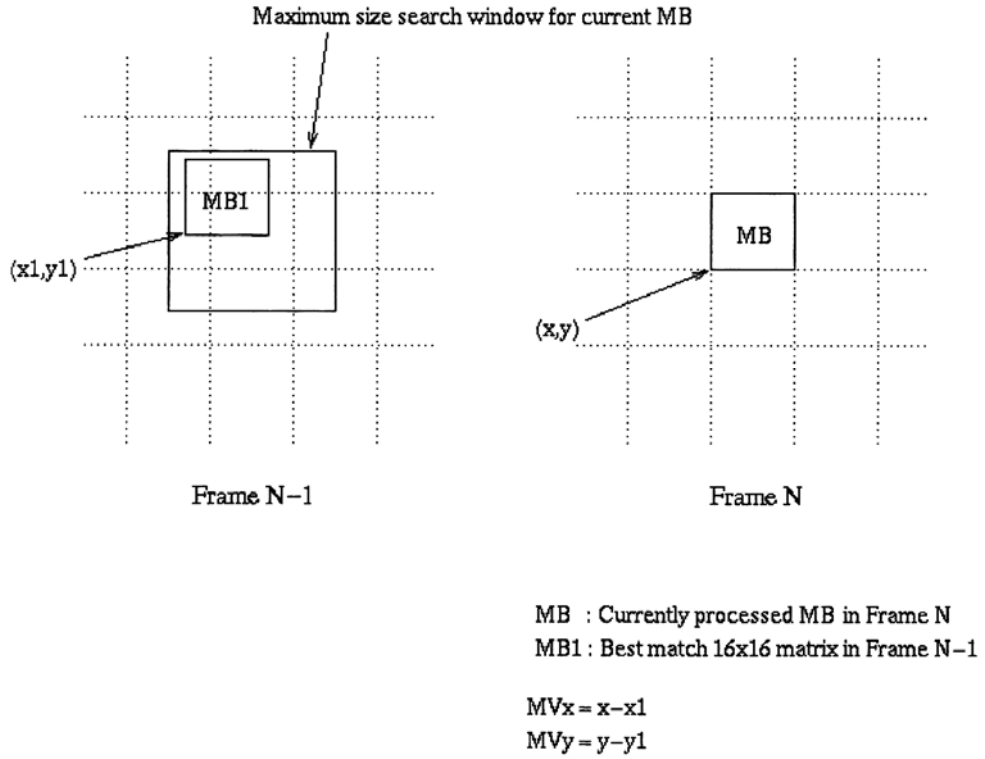
Η μέθοδος κωδικοποίησης INTER χρησιμοποιεί τη διεργασία εκτίμησης κίνησης [5] με ταίριασμα των συγκροτημάτων (block matching-BM), όπου κάθε μακροσυγκρότημα στο τρέχον επεξεργαζόμενο καρέ συγκρίνεται με τα μακροσυγκροτήματα που βρίσκονται στο προηγούμενο ανακατασκευασμένο καρέ μέσω ενός παραθύρου έρευνας με μέγεθος που καθορίζεται από τον χρήστη. Το μέγεθος του παραθύρου έρευνας περιορίζεται τόσο ώστε όλα τα εικονοστοιχεία αναφοράς να είναι μέσα στην επιφάνεια αναφοράς της εικόνας.

Το κριτήριο ταιριάσματος ίσως να είναι οποιοδήποτε μέτρο λάθους, όπως το μέσο τετραγωνικό σφάλμα ή το άθροισμα απόλυτης διαφοράς (sum of absolute difference-

SAD) και μόνο η φωτεινότητα χρησιμοποιείται στην διεργασία εκτίμησης της κίνησης. Ο πίνακας 16 × 16 στο προηγούμενο ανακατασκευασμένο καρέ που καταλήγει στα ελάχιστα SAD θεωρείται ο καλύτερα ταιριαστός με το τρέχον μακροσυγκρότημα. Το διάνυσμα μετατόπισης ανάμεσα στο τρέχον μακροσυγκρότημα και στον καλύτερα ταιριαστό 16 × 16 πίνακα στο προηγούμενο ανακατασκευασμένο καρέ ονομάζεται διάνυσμα κίνησης (motion vector-MV) κι αναπαριστάται από κάθετα κι οριζόντια στοιχεία. Τόσο τα οριζόντια όσο και τα κάθετα στοιχεία του διανύσματος κίνησης πρέπει να σταλούν στον αποκωδικοποιητή για τη σωστή ανακατασκευή του σχετιζόμενου μακροσυγκροτήματος. Τα διανύσματα κίνησης κωδικοποιούνται διαφορικά χρησιμοποιώντας συντεταγμένες ενός διανύσματος κίνησης που κάνει την πρόγνωση. Η διεργασία εκτίμησης κίνησης σε ένα P-frame ενός block-based κωδικοποιητή video φαίνεται στο Σχήμα 2.3.

2.3.3.3 Μετασχηματισμός DCT

Για να μειωθούν οι συσχετισμοί μεταξύ των συντελεστών ενός μακροσυγκροτήματος, η πληροφορία που περικλείει η εικόνα μεταφέρεται από το πεδίο του χώρου στο



Σχήμα 2.3 Εκτίμηση κίνησης σε ένα block-based κωδικοποιητή video

πεδίο της συχνότητας (αφηρημένο πεδίο), όπου η περιγραφή της μπορεί να γίνει με σημαντικά μικρότερο πλήθος bits, για διάφορους λόγους. Υπάρχουν αρκετοί μαθηματικοί μετασχηματισμοί για το σκοπό αυτό, αλλά τα σύγχρονα βασισμένα σε συγκροτήματα πρότυπα, όπως τα MPEG-1, MPEG-2, ITU-T H.261 και H.263, χρησιμοποιούν το Διακριτό Συνημιτονικό Μετασχηματισμό (Discrete Cosine Transform) [6, 19] διότι είναι συγκριτικά γρηγορότερος.

Στους αλγορίθμους κωδικοποίησης video οι οποίοι είναι βασισμένοι σε συγκροτήματα, οι 64 συντελεστές του κάθε 8 × 8 block σε ένα καρέ video υφίστανται δισδιάστατο DCT. Ο μετασχηματισμός αυτός μετατρέπει τα εικονοστοιχεία του κάθε block σε κάθετους και οριζόντιους συντελεστές χωρικής συχνότητας. Ο δισδιάστατος 8 × 8 μετασχηματισμός DCT που χρησιμοποιείται στους βασισμένους σε συγκροτήματα κωδικοποιητές video δίδεται από την Εξίσωση 2.1.

$$F(u,v) = \frac{1}{4}C(u)C(v)\sum_{x=0}^{7}\sum_{y=0}^{7} f(x,y)\cos\left[\frac{\pi(2x+1)u}{16}\right]\cos\left[\frac{\pi(2y+1)v}{16}\right] \qquad (2.1)$$

Η τιμή της F(u, v) αναπαριστά την τιμή του μετασχηματισμένου συντελεστή στη θέση (u, v) και η f(x, y) είναι η αρχική τιμή του εικονοστοιχείου στη θέση (x, y). Για τις τιμές των C(u) και C(v) ισχύει:

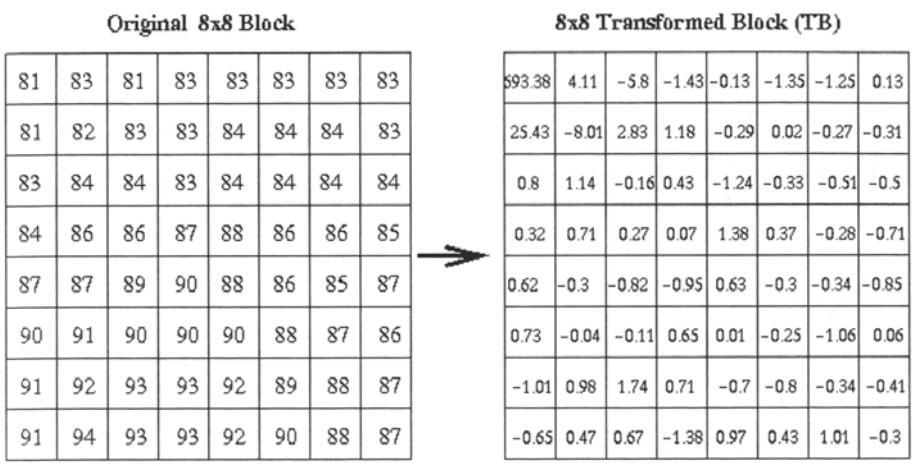
C(u) = $\frac{1}{\sqrt{2}}$ για u = 0, διαφορετικά C(u) = 1

C(v) = $\frac{1}{\sqrt{2}}$ για u = 0, διαφορετικά C(v) = 1

Για u = v = 0, η Εξίσωση 2.1 αποδίδει το μέσο όρο των εικονοστοιχείων του συγκροτήματος, ο οποίος αναφέρεται ως τιμή DC. Αν το αντίστοιχο συγκρότημα έχει υποστεί κωδικοποίηση INTRA, ο συντελεστής αυτός (0, 0) αναφέρεται ως συντελεστής INTRADC και οι υπόλοιποι 63 συντελεστές ως συντελεστές AC. Ο αντίστροφος μετασχηματισμός DCT δίδεται από την Εξίσωση 2.2.

$$f(x,y) = \frac{1}{4}C(u)C(v)\sum_{u=0}^{7}\sum_{v=0}^{7} F(u,v)\cos\left[\frac{\pi(2x+1)u}{16}\right]\cos\left[\frac{\pi(2y+1)v}{16}\right] \qquad (2.2)$$

Ένα παράδειγμα εφαρμογής του ευθύ μετασχηματισμού DCT σε ένα συγκρότημα 8 × 8 απεικονίζεται στο Σχήμα 2.4.



Σχήμα 2.4 Παράδειγμα εφαρμογής του μετασχηματισμού DCT σε ένα συγκρότημα 8 × 8 εικονοστοιχείων

Είναι προφανές από το Σχήμα 2.4 ότι η κατανομή των συντελεστών στο μετασχηματισμένο συγκρότημα απέχει πολύ από την ομοιόμορφη, με λίγους μεγάλους συντελεστές στο πάνω αριστερά τμήμα του συγκροτήματος (ο μεγαλύτερος από τους οποίους είναι ο INTRADC) και μικρούς συντελεστές αλλού. Επομένως, ο μετασχηματισμός DCT έχει μειώσει αξιοσημείωτα το χωρικό πλεονασμό του συγκροτήματος και έχει εκμηδενίσει τους συσχετισμούς μεταξύ των αρχικών εικονοστοιχείων. Η ενέργεια του συγκροτήματος συγκεντρώνεται στο πάνω αριστερά τμήμα του, όπου βρίσκονται οι συντελεστές χαμηλής συχνότητας του αρχικού συγκροτήματος. Εφόσον το ανθρώπινο οπτικό σύστημα είναι πιο ευαίσθητο στους συντελεστές DCT χαμηλής τάξης, οι βασισμένοι σε συγκροτήματα αλγόριθμοι κωδικοποίησης video εκμεταλλεύονται την ευαισθησία αυτή κωδικοποιώντας τον πολύ σημαντικό συντελεστή DC του συγκροτήματος με περισσότερη ακρίβεια από ότι τους υπόλοιπους 63 συντελεστές AC. Κάθε ένας από τους συντελεστές DC ανατίθεται σε μία κωδική λέξη (codeword) πλάτους 8 bits, ενώ οι συντελεστές AC κωδικοποιούνται με τη μέθοδο run-length, η οποία περιγράφεται στην παράγραφο 2.3.3.6.

2.3.3.4 Ποσοτικοποίηση

Η διαδικασία της συμπίεσης σε κωδικοποιητές video βασισμένους σε συγκροτήματα αποδίδεται κυρίως στην ποσοτικοποίηση [7] των μετασχηματισμένων συντελεστών. Ο ποσοτικοποιητής θεωρείται το πιο σημαντικό τμήμα του κωδικοποιητή video,

καθώς ελέγχει τόσο την αποδοτικότητα της κωδικοποίησης, όσο και την ποιότητα του ανακατασκευασμένου video. Στην κωδικοποίηση video υπάρχουν αρκετές τεχνικές για την ποσοτικοποίηση των καρέ.

Ο ποσοτικοποιητής απεικονίζει τις τιμές των μετασχηματισμένων συντελεστών σε μια λιγότερο ευρεία κλίμακα τιμών, με σκοπό να μειώσει τον αριθμό των bits που απαιτούνται για την κωδικοποίηση του κάθε συγκροτήματος. Η ποσοτικοποίηση είναι μια απωλεστική διαδικασία, διότι η ακριβής τιμή του κάθε εικονοστοιχείου δεν μπορεί να αποκατασταθεί μετά και την αντίστροφη ποσοτικοποίηση, οπότε υποβιβάζει την ποιότητα της εικόνας με το πλεονέκτημα της αυξημένης αποδοτικότητας της κωδικοποίησης. Οι εξισώσεις που ακολουθούν παρουσιάζουν τις διεργασίες της ορθής και της αντίστροφης ποσοτικοποίησης που εκτελούνται από έναν κωδικοποιητή / αποκωδικοποιητή H.263 αντίστοιχα. Στις εξισώσεις αυτές το COF είναι ο μετασχηματισμένος συντελεστής που πρόκειται να ποσοτικοποιηθεί, το LEVEL είναι η απόλυτη τιμή του ποσοτικοποιημένου συντελεστή και το COF΄ είναι ο αναδομημένος μετασχηματισμένος συντελεστής μετά την αντίστροφη ποσοτικοποίηση. Το Qp καλείται επίπεδο του ποσοτικοποιητή ή παράμετρος ποσοτικοποίησης και το 2 × Qp είναι το μέγεθος βήματος του ποσοτικοποιητή.

Ποσοτικοποίηση

- INTRADC συντελεστής:

  $LEVEL = COF / 8$

- INTRA AC συντελεστής:

  $LEVEL = |COF| / (2 \times Qp)$

- INTER συντελεστές:

  $LEVEL = (|COF| - Qp / 2) / (2 \times Qp)$

Αντίστροφη ποσοτικοποίηση

- INTRADC συντελεστής:

  $COF' = LEVEL \times 8$

- INTRA ή INTER συντελεστές:

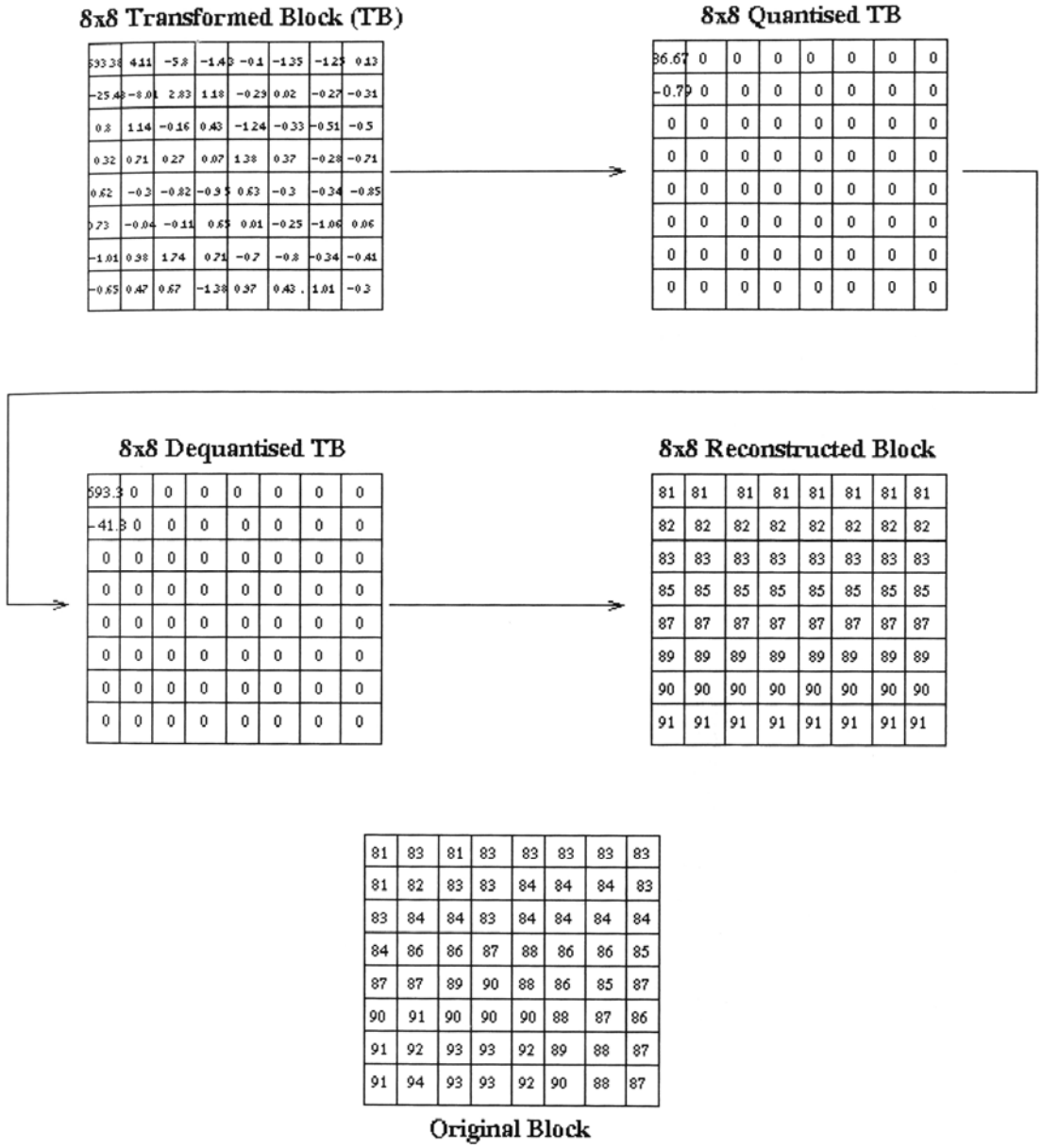
$|COF'| = 0$ εάν LEVEL = 0,

$|COF'| = 2Qp \times LEVEL + Qp$ εάν LEVEL ≠ 0 και Qp περιττό

$|COF'| = 2Qp \times LEVEL + Qp - 1$ εάν LEVEL ≠ 0 και Qp άρτιο

Στη συνέχεια, με τη βοήθεια του προσήμου του COF υπολογίζεται η τιμή του COF΄ ως COF΄ = πρόσημο(COF) × |COF΄|.

Το Σχήμα 2.5 παρουσιάζει την ποσοτικοποίηση, την αντίστροφη ποσοτικοποίηση και τον αντίστροφο μετασχηματισμό DCT των μετασχηματισμένων συντελεστών του συγκροτήματος, όπως αυτές υπολογίστηκαν στο Σχήμα 2.4.



Σχήμα 2.5 Παράδειγμα ποσοτικιποίησης, αντίστροφης ποσοτικοποίησης και αντίστροφου μετασχηματισμού DCT σε ένα συγκρότημα 8 × 8 εικονοστοιχείων για $Q_p = 10$

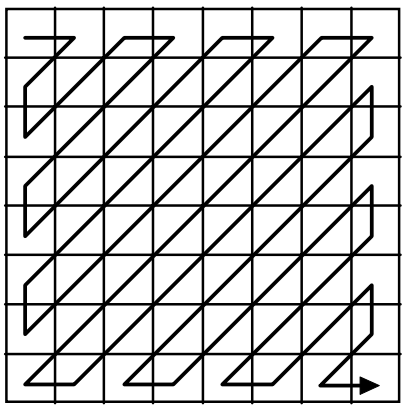
2.3.3.5 Κωδικοποίηση zigzag

Το δισδιάστατο κβαντισμένο συγκρότημα των συντεταγμένων DCT αποτελείται από ένα μικρό αριθμό μη-μηδενικών συντελεστών που βρίσκονται στο πάνω αριστερά τμήμα του συγκροτήματος και από ένα μεγάλο αριθμό μηδενικών συντελεστών, που βρίσκονται στα υπόλοιπα τμήματά του. Η συγκέντρωση αυτή των μη-μηδενικών συντελεστών στο πάνω αριστερά τμήμα του συγκροτήματος μπορεί να εκμεταλλευτεί εκτελώντας σάρωση zigzag, κάτι που γίνεται σε αρκετά σύγχρονα πρότυπα, όπως στα H.261 και H.263. Η σειρά με την οποία εκτελείται η σάρωση zigzag παρουσιάζεται στο Σχήμα 2.5.

Ως αποτέλεσμα της κωδικοποίησης zigzag, οι μη-μηδενικοί συντελεστές των χαμηλών συχνοτήτων συγκεντρώνονται σειριακά σε μία μονοδιάστατη ροή με έναν αριθμό διαδοχικών μηδενικών πολλούς μηδενικούς συντελεστές και ακολουθούνται από πολλά συνεχόμενα μηδενικά.

2.3.3.6 Κωδικοποίηση run – length

Ο κωδικοποιητής run – length δεν κωδικοποιεί κάθε συντελεστή του συγκροτήματος χωριστά, αλλά τη μονοδιάστατη ροή των συντελεστών που προκύπτει από την κωδικοποίηση zigzag. Το μήκος κάθε ομάδας μηδενικών και η προηγούμενη μη μηδενική τιμή κωδικοποιούνται. Ένα επιπλέον bit χρησιμοποιείται σε κάθε ομάδα για να δείξει αν η ομάδα αυτή είναι η τελευταία του συγκροτήματος. Συνεπώς, ένα EVENT (πολλά EVENTS απαρτίζουν την κωδικοποιημένη ροή των συντελεστών) είναι συνδυασμός τριών παραμέτρων:



Σχήμα 2.6 Η σειρά με την οποία εκτελείται η σάρωση zigzag

LAST	0 Υπάρχουν κι άλλοι μη μηδενικοί συντελεστές στο συγκρότημα
	1 Αυτός είναι ο τελευταίος μη μηδενικός συντελεστής στο συγκρότημα
RUN	Αριθμός των μηδενικών συντελεστών που προηγούνται του τρέχοντος μη μηδενικού συντελεστή
LEVEL	Πλάτος του συντελεστή

Μια ακόμα βασική τεχνική συμπίεσης video είναι η object-based video coding, η οποία μαζί με το μετασχηματισμό DCT χρησιμοποιείται στο αρκετά διαδεδομένο σήμερα MPEG-4, αλλά δεν κρίνεται απαραίτητη η ανάλυσή της στην παρούσα εργασία.

2.4 Πρότυπα

Ως τώρα έχουν χρησιμοποιηθεί πολλές από τις ήδη υπάρχουσες τεχνικές και αλγορίθμους για την ανάπτυξη προτύπων κωδικοποίησης video [8]. Η προτυποποίηση της μεθοδολογίας κωδικοποίησης είναι επίσης απαραίτητη για να εξασφαλιστεί σωστή επικοινωνία μεταξύ των κωδικοποιητών και των αποκωδικοποιητών που έχουν αναπτυχθεί από διάφορες ομάδες και βιομηχανίες [9].

Τόσο ο Διεθνής Οργανισμός Τυποποίησης (International Standardisation Organisation-ISO) όσο και η Διεθνής Ένωση Τηλεπικοινωνιών (International Telecommunications Union-ITU) έχουν ανακοινώσει από το 1985 προτάσεις για καθολικούς αλγορίθμους κωδικοποίησης εικόνας και video, οι κυριότερες από τις οποίες αναφέρονται στη συνέχεια. Το πρώτο πρότυπο για κωδικοποίηση video, το MPEG-1 [10], ανακοινώθηκε το 1991 και προοριζόταν για αποθήκευση οπτικοακουστικού υλικού σε CD-ROM με bit rate 1,5-2 Mbps. Το 1993 ακολούθησε το ITU-T H.261 [11] για επικοινωνίες χαμηλού bit rate ($p \times 64$ kbps) σε δίκτυα ISDN. Το MPEG-2 [12] ανακοινώθηκε το 1994 σαν αλγόριθμος κωδικοποίησης για εφαρμογές τηλεόρασης υψηλής ευκρίνειας (HDTV) στα 4-9 Mbps. Το 1996 καθορίστηκε η πρώτη έκδοση του προτύπου ITU-T H.263 [13] για επικοινωνίες πολύ χαμηλού bit rate σε δίκτυα PSTN (< 64 kbps), ενώ οι βελτιωμένες του εκδόσεις

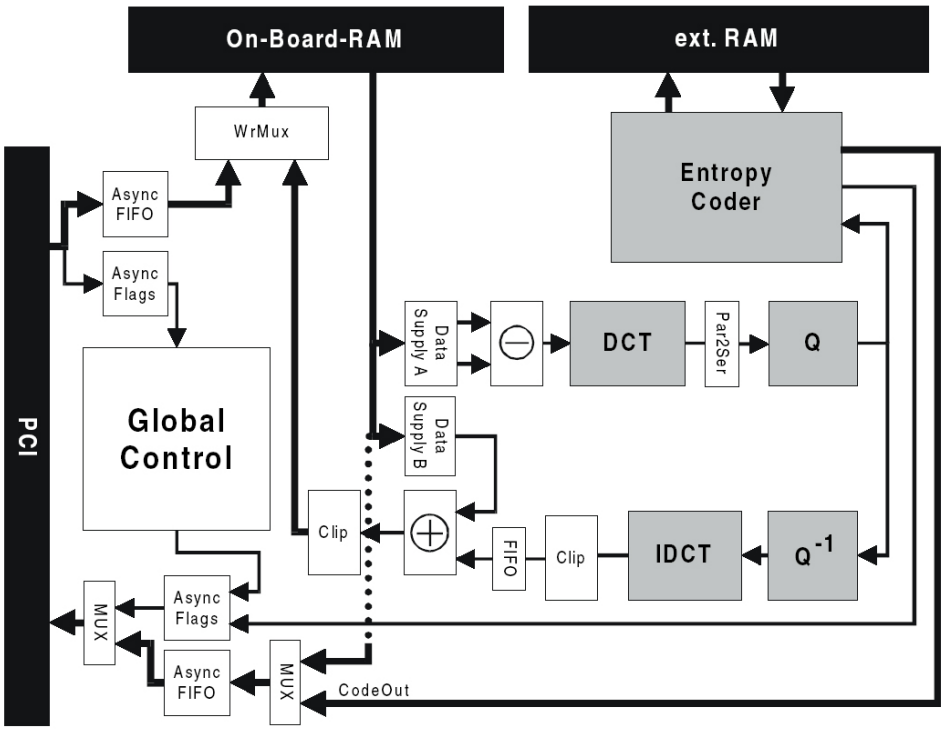
H.263+ [14] και H.263++ ανακοινώθηκαν το 1998 και το 1999 αντίστοιχα. Ο αλγόριθμος κωδικοποίησης MPEG-4 [15] για κινητές οπτικοακουστικές επικοινωνίες ανακοινώθηκε το 2000 και υποστηρίζει χαμηλό bit rate (μέχρι 64 kbps) για video χαμηλής ευκρίνειας και υψηλό bit rate (μέχρι 2 Mbps) για video υψηλής ευκρίνειας. Τέλος, το 2002 ανακοινώθηκε το H.264 [16], το οποίο είναι αρκετά πιο αποδοτικό από το H.263, αφού η ποιότητα που επιτυγχάνει με bit rate 384 kbps είναι εφάμιλλη αυτής που μπορεί να επιτύχει το H.263 με bit rate 768 kbps.

2.5 Υλοποιήσεις σε hardware

Οι υλοποιήσεις αλγορίθμων σε hardware προσφέρουν υψηλές ταχύτητες σε σχέση με αυτές του software, ενώ έχουν και το πλεονέκτημα ότι μπορούν να ενσωματωθούν σε φορητά συστήματα. Επίσης, υλοποιήσεις σε FPGAs (Field Programmable Gate Arrays), αν και γενικά υστερούν σε ταχύτητα σε σχέση με τα ASIC, είναι εύκολες στον επαναπρογραμματισμό τους, κάτι που τις καθιστά ιδιαίτερα ευέλικτες. Στη συνέχεια θα γίνει σύντομη περιγραφή κάποιων χαρακτηριστικών υλοποιήσεων αλγορίθμων συμπίεσης video ή τμημάτων αυτών σε hardware.

2.5.1 Συμπιεστής video συμβατός με το πρότυπο H.263

Η υλοποίηση [17] βασίζεται σε μία κάρτα PCI με ενσωματωμένη την FPGA Xilinx XC4085XLA η οποία φέρει και μνήμη τύπου SRAM, ενώ το σύστημα χρησιμοποιεί κι εξωτερική SRAM. Τα κυριότερα υποσυστήματα που είναι αυτά του ευθύ και του ανάστροφου μετασχηματισμού DCT, του ποσοτικοποιητή και του αποποσοτικοποιητή, καθώς και του κωδικοποιητή εντροπίας. Ένα απλοποιημένο block διάγραμμα του κυκλώματος κωδικοποίησης παρουσιάζεται στο Σχήμα 2.7. Η ανταλλαγή τωνδεδομένων μεταξύ της FPGA και της διεπαφής PCI ελέγχεται από ασύγχρονες FIFOs και ειδικά σήματα. Πριν υποστούν επεξεργασία, τα δεδομένα του video αποθηκεύονται στην ενσωματωμένη SRAM της κάρτας PCI, ενώ εκεί αποθηκεύονται και τα αναδομημένα δεδομένα του video.

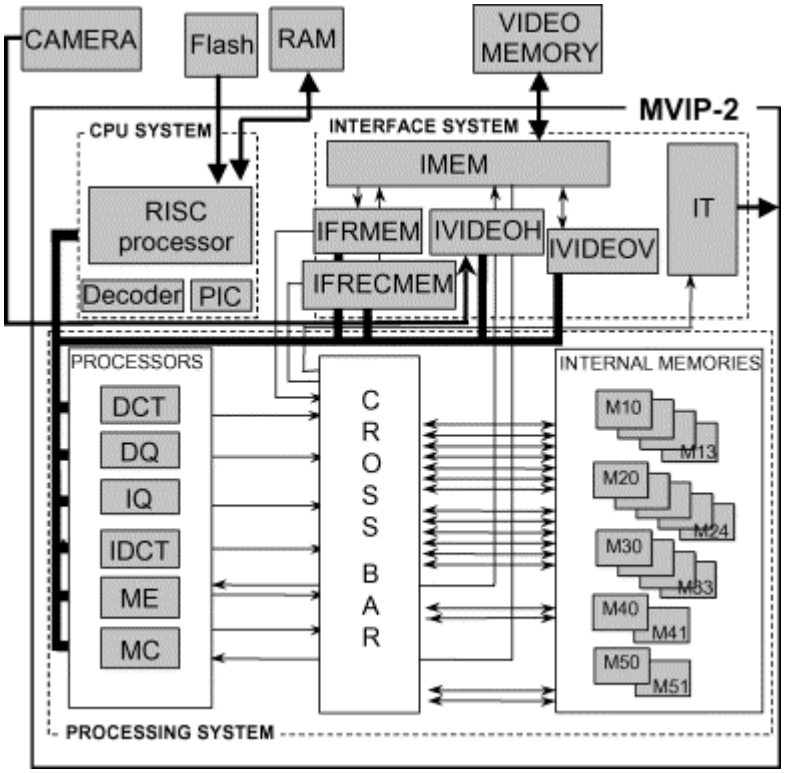


Σχήμα 2.7 Απλοποιημένο block διάγραμμα του κυκλώματος κωδικοποίησης

Το σύστημα λειτουργεί στη συχνότητα των 30 MHz, ενώ η μέγιστη ταχύτητα συμπίεσης ανέρχεται στα 120 Mbps. Αυτό μεταφράζεται σε ρυθμό συμπίεσης της τάξης των 98,6 πλαισίων CIF ανά δευτερόλεπτο. Συνεπώς, τρεις ακολουθίες εικόνων με ρυθμό μετάδοσης 30 καρέ ανά δευτερόλεπτο (frames per second-fps) μπορούν να συμπιεστούν ταυτόχρονα.

2.5.2 Αρχιτεκτονική για συμπίεση video συμβατή με το πρότυπο H.263

Η σχεδίαση αποτελείται από έναν επεξεργαστή RISC που ελέγχει μία ομάδα επεξεργαστών που εκτελούν ευθύ και ανάστροφο διακριτό συνημιτονικό μετασχηματισμό (DCT και IDCT), ευθεία και ανάστροφη ποσοτικοποίηση (DQ και IQ) και εκτίμηση κίνησης (ME). Επίσης περιλαμβάνει τμήματα προ-επεξεργασίας για το σήμα video από την συνδεδεμένη κάμερα και διεπαφή για την εξωτερική μνήμη video και τη δημιουργία της ροής δεδομένων H.263. Το σύστημα ονομάζεται MVIP-2, το οποίο αποτελεί εξέλιξη του MViP [18], και στο Σχήμα 2.8 φαίνεται το block διάγραμμά του. Επίσης, αναφέρεται ότι οι επεξεργαστές έχουν γραφτεί σε synthesizeable Verilog.



Σχήμα 2.8 Block διάγραμμα του MVIP-2

Οι δοκιμές έδειξαν ότι το MVIP-2 μπορεί να κωδικοποιήσει το κάθε μακροσυγκρότημα σε περίπου 4050 κύκλους ρολογιού. Αυτό επιτρέπει την κωδικοποίηση 30 QCIF (176 × 144) fps με ταχύτητα ρολογιού 12 MHz ή 30 CIF fps με ταχύτητα ρολογιού 48 MHz.

Το αναπτυξιακό που χρησιμοποιήθηκε ήταν το HSDT200 με ενσωματωμένη την FPGA EP20K400BC652-1 της ALTERA. Η σχεδίαση κατέλαβε το 93% των logic cells της και το 52% της διαθέσιμής της μνήμης. Τελικά, λόγω των φυσικών περιορισμών του αναπτυξιακού, η μέγιστη συχνότητα που επιτεύχθηκε ήταν 24MHz, που επιτρέπει την κωδικοποίηση 15 CIF fps.

2.5.3 Υλοποίηση σε FPGA του αλγορίθμου LRU για συμπίεση video

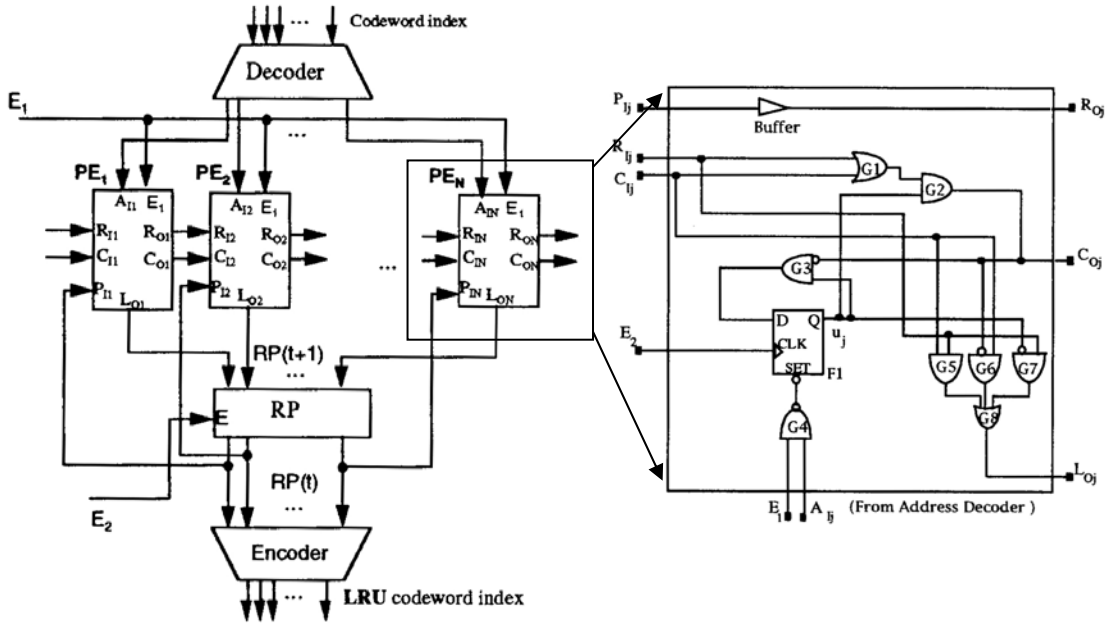
Ο αλγόριθμος LRU (Least Recently Used) χρησιμοποιείται σε μια κατηγορία κωδικοποίησης video βασισμένη σε διανύσματα που χρησιμοποιεί μνήμη cache. Πιο συγκεκριμένα, χρησιμοποιούνται από δύο codebooks στον κωδικοποιητή και στον αποκωδικοποιητή, ένα πρωτεύον κι ένα δευτερεύον. Ο αλγόριθμος LRU χρησιμεύει

στο να κρατά ενημερωμένο το πρωτεύον codebook με τα codewords που χρησιμοποιούνται συχνότερα, ώστε να μειώνονται οι προσβάσεις στο δευτερεύον και να γίνεται πιο γρήγορο το σύστημα.

Η αρχιτεκτονική για την υλοποίηση του αλγορίθμου φαίνεται στο Σχήμα 2.9. Αποτελείται από έναν αποκωδικοποιητή, N επεξεργαστικές μονάδες (Processing Elements-PE), όπου N ο αριθμός των codewords στο πρωτεύον codebook, έναν αφαιρετικό δείκτη (Removal Pointer-RP) που κρατάει τον δείκτη του τελευταίου LRU codeword κι έναν κωδικοποιητή. Κάθε επεξεργαστική μονάδα αποτελείται από ένα flip-flop κι από συνδυαστική λογική.

Η συχνότητα λειτουργίας του συστήματος ανέρχεται στα 16MHz, που αρκεί για την εκτέλεση σε πραγματικό χρόνο του αλγορίθμου συμπίεσης Video.

Εκτός από τις προαναφερθέντα συστήματα, υπάρχουν και πολλά άλλα. Στο [20] γίνεται χρήση ενός μικροελεγκτή ARM συχνότητας 200MHz και μιας ομάδας επεξεργαστών που ελέγχονται από αυτόν και λειτουργούν στα 66MHz. Αυτό το σύστημα υλοποιεί το H.263 για ανάλυση QCIF στα 29 fps. Μία αρχιτεκτονική βασισμένη σε ένα προγραμματιζόμενο address generator κι ένα ελεγκτή pipeline για μια ομάδα επεξεργαστών παρουσιάζεται στο [21]. Με συχνότητα ρολογιού 27MHz υλοποιεί κι αυτό το H.263 για ανάλυση QCIF στα 30 fps. Στο [22] παρουσιάζεται ένα



Σχήμα 2.9 Σχηματικό διάγραμμα της αρχιτεκτονικής για την υλοποίηση του αλγορίθμου LRU και block διάγραμμα μιας επεξεργαστικής μονάδας

chip για κωδικοποίηση / αποκωδικοποίηση των H.261, H.263 και H.263+ με επιδόσεις έως 30 εικόνες QCIF ανά δευτερόλεπτο.

Αρκετά συστήματα υλοποιούν κάποιο τμήμα μόνο του κωδικοποιητή/ αποκωδικοποιητή. Στο [23] παρουσιάζονται δύο υλοποιήσεις του μετασχηματισμού DCT στην FPGA Xilinx XC4010. Στο [24] παρουσιάζεται η υλοποίηση ενός γρήγορου αλγορίθμου για εκτίμηση κίνησης σε FPGA. Το σύστημα αυτό είναι ικανό να επεξεργαστεί έγχρωμες εικόνες μεγέθους έως 1024 × 768 εικονοστοιχείων @ 25 fps και συμμορφώνεται πλήρως με το πρότυπο MPEG-2. Τέλος, στο [25] παρουσιάζεται μια επισκόπηση των αρχιτεκτονικών VLSI για συμπίεση video μέχρι το 1995.

2.6 Επίλογος

Στο κεφάλαιο αυτό έχει γίνει αναφορά στις γενικές αρχές μη απωλεστικής και απωλεστικής συμπίεσης δεδομένων και στους παράγοντες που πρέπει να λαμβάνονται υπ' όψη για την επιλογή της κατάλληλης μεθόδου συμπίεσης. Επίσης, έχουν παρουσιαστεί οι βασικές αρχές αρκετών σύγχρονων αλγορίθμων συμπίεσης video. Οι αλγόριθμοι αυτοί εκμεταλλεύονται τους χωρικούς και χρονικούς πλεονασμούς από τις ακολουθίες ψηφιακών video, τους οποίους προσπαθούν να αφαιρέσουν. Εκτός από τους αλγορίθμους αυτούς έχει γίνει αναφορά στους πρότυπους αλγορίθμους συμπίεσης video, οι οποίοι βασίζονται κατά κύριο λόγο στην κωδικοποίηση συγκροτημάτων. Τέλος, έχουν παρουσιαστεί συνοπτικά κάποιες υλοποιήσεις σε hardware πρότυπων ή μη αλγορίθμων.

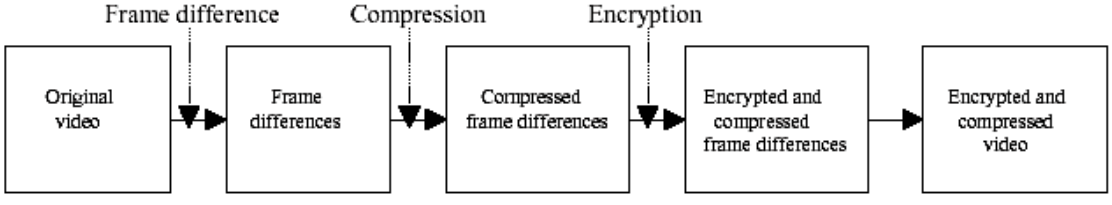
Κεφάλαιο 3

Ο αλγόριθμος SCAN για συμπίεση video

3.1 Γενικά

Η μεθοδολογία SCAN [26, 27, 28, 29] είναι μία κλάση από τυπικές γλώσσες (formal languages), οι οποίες μπορούν να εφαρμοστούν στη συμπίεση, την κρυπτογράφηση, την απόκρυψη δεδομένων (data hiding) ή σε συνδυασμούς αυτών. Ο αλγόριθμος SCAN για συμπίεση video είναι σχεδιασμένος για να χρησιμοποιηθεί στη συμπίεση video το οποίο στη συνέχεια πρόκειται να κρυπτογραφηθεί. Ο αλγόριθμος της κρυπτογράφησης σε hardware έχει ήδη υλοποιηθεί [30, 31, 32, 33] και μελλοντικά υπάρχει η προοπτική σύνδεσής του με την αρχιτεκτονική που περιγράφεται στην παρούσα εργασία. Ο αλγόριθμος καθώς και τα πειραματικά αποτελέσματα δεν είναι κομμάτι αυτής της εργασίας, αλλά περιλαμβάνονται γιατί παρέχουν τις προδιαγραφές της αρχιτεκτονικής και μια βάση η οποία μπορεί να χρησιμοποιηθεί για να συγκριθεί με τα αποτελέσματα της υλοποίησης σε hardware.

Ο αλγόριθμος SCAN για συμπίεση video βασίζεται στις ομοιότητες των γειτονικών καρέ, δηλαδή χρησιμοποιεί μέθοδο συμπίεσης INTER σύμφωνα με όσα έχουν ως τώρα αναφερθεί. Η συμπίεση που πραγματοποιείται είναι απωλεστική, αυτό όμως που πρέπει να επισημανθεί είναι ότι ο βαθμός συμπίεσης μπορεί να καθοριστεί από το χρήστη, γεγονός που μπορεί να οδηγήσει είτε σε μικρό βαθμό συμπίεσης με πολύ καλή ποιότητα ανακατασκευασμένου video είτε σε μεγαλύτερους βαθμούς συμπίεσης, με την όποια υποβάθμιση στην ποιότητα αυτό συνεπάγεται. Στο Σχήμα 3.1 φαίνεται ένα block διάγραμμα του συνδυασμού της κρυπτογράφησης και της συμπίεσης video [34].



Σχήμα 3.1 Block διάγραμμα του συνδυασμού κρυπτογράφησης και συμπίεσης video

3.2 Περιγραφή του αλγορίθμου συμπίεσης video

Παρόλο που ο υλοποιημένος σε software αλγόριθμος συμπίεσης video περιλαμβάνει και την κρυπτογράφηση, θα αναλυθεί μόνο το κομμάτι της συμπίεσης, μια και αυτό έχει υλοποιηθεί στην παρούσα εργασία. Τα βήματα του αλγορίθμου έχουν ως εξής:

- Δημιουργία καρέ διαφορών

  Το δεύτερο καρέ της ακολουθίας video συγκρίνεται με το πρώτο, για να δημιουργηθεί το καρέ με τις διαφορές. Οι διαφορές βρίσκονται συγκρίνοντας την τιμή φωτεινότητας κάθε εικονοστοιχείου του τρέχοντος καρέ (δηλαδή του δευτέρου) με την τιμή φωτεινότητας του αντίστοιχου εικονοστοιχείου του πρώτου καρέ. Αν η τιμή της διαφοράς είναι μεταξύ του –TH και του TH, όπου TH η τιμή κατωφλίου που έχει ορίσει ο χρήστης, το αντίστοιχο εικονοστοιχείο στο καρέ των διαφορών παίρνει την τιμή -1, διαφορετικά παίρνει την τιμή που έχει το εικονοστοιχείο του δεύτερου καρέ. Επομένως, έχει δημιουργηθεί ένα νέο καρέ, το οποίο αποτελείται από εικονοστοιχεία με τιμή είτε -1 για όσα τμήματα των δύο συνεχόμενων καρέ έχουν την ίδια περίπου τιμή φωτεινότητας, είτε με τιμή ίση με αυτή του αντίστοιχου εικονοστοιχείου του τρέχοντος καρέ αν η διαφορά στην τιμή φωτεινότητας των εικονοστοιχείων είναι αξιοσημείωτη. Αυτά που έχουν τιμή -1 αντιπροσωπεύουν τμήματα της ακολουθίας video χωρίς ιδιαίτερη κίνηση, ενώ τα υπόλοιπα αντιπροσωπεύουν τμήματα με κίνηση.

- Δημιουργία καρέ σύγκρισης

  Ταυτόχρονα με το καρέ των διαφορών φτιάχνεται και το καρέ σύγκρισης που θα χρησιμοποιηθεί για τη δημιουργία του επόμενου καρέ διαφορών. Πιο συγκεκριμένα, το τρίτο καρέ της ακολουθίας video δε θα συγκριθεί με το δεύτερο καρέ, αλλά με ένα νέο (το καρέ σύγκρισης), το οποίο δημιουργείται ως εξής: Για κάθε εικονοστοιχείο του πρώτου και του δευτέρου καρέ, αν η τιμή της διαφοράς που υπολογίζεται όπως παραπάνω είναι εντός ορίων του κατωφλίου, τότε κάθε εικονοστοιχείο του καρέ σύγκρισης παίρνει την τιμή φωτεινότητας του αντίστοιχου εικονοστοιχείου του πρώτου καρέ, διαφορετικά παίρνει την τιμή φωτεινότητας του αντίστοιχου εικονοστοιχείου του δευτέρου καρέ.

Στη συνέχεια της διαδικασίας, το τρίτο καρέ θα συγκριθεί με το καρέ σύγκρισης για τη δημιουργία του επόμενου καρέ διαφορών, ενώ το δεύτερο καρέ σύγκρισης, με το οποίο θα συγκριθεί το τέταρτο καρέ της ακολουθίας video, θα αποτελείται από τμήματα του τρίτου καρέ και του πρώτου καρέ σύγκρισης, κ.ο.κ.

- Συμπίεση των καρέ των διαφορών

  Κάθε καρέ διαφορών αναλύεται σε παράθυρα 4 x 4 εικονοστοιχείων.

  Κάθε παράθυρο εξετάζεται για να διαπιστωθεί αν έχει τουλάχιστον ένα στοιχείο διαφορετικό του -1. Στην περίπτωση αυτή, το παράθυρο κωδικοποιείται, διαφορετικά δεν κρατιέται καμία πληροφορία για αυτό στο συμπιεσμένο αρχείο.

  Για κάθε παράθυρο που κωδικοποιείται αυξάνεται κι ένας μετρητής, ο οποίος όταν τελειώσει το καρέ έχει τιμή ίση με τον αριθμό των παραθύρων που έχουν κωδικοποιηθεί, πληροφορία που κρατιέται στο τελικό συμπιεσμένο αρχείο. Για κάθε νέο καρέ, ο μετρητής μηδενίζεται

- Κωδικοποίηση παραθύρου

  Για την κωδικοποίηση, το πρώτο πράγμα που καταγράφεται είναι οι συντεταγμένες του πάνω αριστερά εικονοστοιχείου του παραθύρου στο πλαίσιο των διαφορών, οι οποίες αποθηκεύονται στο τελικό αρχείο που περιέχει το συμπιεσμένο video, αναπαριστώμενες με όσα bits απαιτούνται για την σωστή αναπαράστασή τους.

  Στη συνέχεια, για κάθε εικονοστοιχείο του παραθύρου ακολουθείται η εξής διαδικασία: αν η τιμή του είναι ίση με -1, στο τελικό αρχείο αποθηκεύεται ένα μηδενικό, διαφορετικά αποθηκεύεται μια μονάδα και η τιμή του, αναπαριστώμενη με όσα bits απαιτούνται για τη σωστή αναπαράστασή της.

Ο αλγόριθμος που μόλις περιγράφηκε παρουσιάζεται στη συνέχεια και σε μορφή ψευδοκώδικα. Καθεμία από τις λειτουργίες του αντιστοιχεί σε διαφορετική συνάρτηση του ψευδοκώδικα. Πιο συγκεκριμένα, η δημιουργία των καρέ σύγκρισης και διαφορών πραγματοποιείται από τη συνάρτηση Difference( ), η συμπίεση των καρέ διαφορών από την Compress( ) και η κωδικοποίηση των παραθύρων από την EncodeWindow( ). Επιπλέον, η συνάρτηση VideoEncrypt( ) εκτελεί την

κρυπτογράφηση του video. Σε όλες τις συναρτήσεις τα σύμβολα N, w, και h δηλώνουν τον αριθμό των καρέ του video, το πλάτος και το ύψος τους (σε αριθμό εικονοστοιχείων) αντίστοιχα. Τα σύμβολα m, n και k δηλώνουν τον αριθμό των bits που χρειάζονται για την αναπαράσταση των συντεταγμένων των πάνω αριστερά εικονοστοιχείων των 4 × 4 παραθύρων και τον αριθμό bits που απαιτούνται για την αναπαράσταση ενός εικονοστοιχείου. Το K δηλώνει το κλειδί της κρυπτογράφησης. Το σύμβολο $B_{p,q}$ δηλώνει την δυαδική αναπαράσταση του ακεραίου p σε q bits. Η έκφραση C = Append(A, B) σημαίνει ότι ο πίνακας B προσαρτάται στο τέλος του πίνακα A και το αποτέλεσμα ανατίθεται στον C. Ο ψευδοκώδικας εμπεριέχει σχόλια για την καλύτερη κατανόησή του.

```
VideoEncrypt(V,K)                    //συνάρτηση για την κρυπτογράφηση του video
Inputs: Original video V, Encryption key K   //είσοδος είναι το V και το K
Output: Encrypted video is sent to receiver   //έξοδος είναι το
                                              /κρυπτογραφημένο video
{
Let F1…FN be the frames of V         //με F1…FN συμβολίζονται τα καρέ του video
   Let G1…GN = Difference(F1…FN)     //με G1…GN συμβολίζονται τα καρέ των
                                     //διαφορών
  Set Buffer = G1                    //αρχή της διαδικασίας κρυπτογράφησης
  For (i = 2-N)
  {
      Comp = Compress(Gi)
      Buffer = Append(Buffer, Comp)
      If size of Buffer >= 256 x 256 bytes
        Encrypt first 256 x 256 bytes of Buffer with K and
        send to receiver and empty those bytes from Buffer
      }
      Encrypt remaining bytes in Buffer and send to receiver
}

G1…GN = Difference(F1…FN)   //συνάρτηση για την εύρεση των καρέ διαφορών και των
                            //καρέ σύγκρισης
Input: Video frames F1…FN            //είσοδος είναι τα καρέ F1…FN του video
Output: Difference frames G1…GN  //έξοδος είναι τα καρέ των διαφορών G1…GN
{
```

```
Let G1 = F1, T = F1     //στα καρέ T και G1 τοποδετείται το 1ο καρέ του video (το καρέ T
                                                    //είναι το καρέ σύγκρισης)
Let TH = user specified value     //η τιμή κατωφλίου TH ορίζεται από το χρήστη
For (k = 2-N)                     //για τα υπόλοιπα καρέ εκτελείται η διαδικασία σύγκρισης
  {
  For 1 <= i <= h, 1 <= j <= w       //Τα εικονοστοιχεία καθενός από τα καρέ
//αυτά συγκρίνονται με τα εικονοστοιχεία με του καρέ  σύγκρισης T και δίδονται οι τιμές στα
//Gk και T ανάλογα με το κατώφλι
    If |Fk[i][j] - T[i][j]| <= TH
      Gk[i][j] = -1
    Else
      Gk[i][j] = Fk[i][j]
      T[i][j] = Gk[i][j]
  }
  Return G1…GN                    //με την ολοκλήρωση της διαδικασίας έχουν δημιουργηθεί τα
                                                    //καρέ των διαφορών
}

X = Compress(D)                   //συνάρτηση για τη συμπίεση των καρέ των διαφορών
Input: Difference frame D         //είσοδος είναι το καρέ των διαφορών D
Output: Compressed difference frame X     //έξοδος είναι το συμπιεσμένο
                                                    //καρέ διαφορών X
{
X = Φ, Count = 0                                    //αρχικοποίηση
Decompose D into 4 x 4 windows    //ανάλυση του D σε παράθυρα 4 × 4 (W)
For each window W                 //έλεγχος παραθύρου W για εύρεση στοιχείων με τιμή
                                                    //διαφορετική του -1
  If W has an element other than -1     //εάν υπάρχει τουλάχιστον ένα τέτοιο
                                                    //στοιχείο
    X = Append(X, EncodeWindow(W)) //καλείται η συνάρτηση EncodeWindow( )
//για να κωδικοποιηθεί το παράθυρο και το αποτέλεσμά της προσαρτάται στο καρέ διαφορών X
    Count = Count + 1             //αύξηση μετρητή κωδικοποιημένων παραθύρων
X = Append(BCount, m + n - 4, X)  //ενσωμάτωση της τιμής του Count στο X
Return X      //με την ολοκλήρωση της διαδικασίας έχει δημιουργηθεί το συμπιεσμένο καρέ
}

U = EncodeWindow(W)               //συνάρτηση κωδικοποίησης των 4 × 4 παραθύρων
Input: 4 x 4 window W from difference frame     //είσοδος το παράθυρο W
```

```
Output: Encoded window U          //έξοδος είναι το κωδικοποιημένο παράθυρο U
Let U = Φ and let (x, y) be the top left corner of W
 U = Append(U, B_{x/4, m - 2}) U = Append(U, B_{y/4, n - 2})    //τοποθέτηση στο U
                                                  //των συντεταγμένων του παραθύρου
For each pixel p in W in raster order      //για κάθε εικονοστοιχείο του W
  If p equals -1                                  //αν είναι ίσο με -1
    U = Append(U, 0)                          //τοποθέτηση στο U της τιμής 0
  Else                                            //διαφορετικά
    U = Append(U, 1), U = Append(U, B_{p, k})  //τοποθέτηση στο U της τιμής 1
                                                  //και της τιμής του εικονοστοιχείου
Return U      //με την ολοκλήρωση της διαδικασίας έχει δημιουργηθεί το κωδικοποιημένο
                                                  //παράθυρο
}
```

Η συνάρτηση Difference( ), όπως έχει ήδη αναφερθεί, δημιουργεί τα καρέ σύγκρισης και διαφορών, από τα οποία την έξοδό της αποτελούν μόνο τα καρέ διαφορών, τα οποία οδηγούνται για κωδικοποίηση. Το πρώτο καρέ της ακολουθίας video ανατίθεται στους πίνακες G_1 και T. Ο G_1 αποτελεί το πρώτο καρέ διαφορών, ενώ ο T είναι το πρώτο καρέ σύγκρισης. Το επόμενο καρέ της ακολουθίας video συγκρίνεται με το καρέ σύγκρισης και ανάλογα με την τιμή του κατωφλίου δίνονται οι κατάλληλες τιμές στο δεύτερο καρέ διαφορών (G_2) και ενημερώνεται κατάλληλα το καρέ σύγκρισης (T), ώστε η διαδικασία να συνεχιστεί με τα επόμενα καρέ.

Η συνάρτηση Compress( ) συμπιέζει τα καρέ των διαφορών. Η είσοδός της (D) είναι ένα καρέ διαφορών, το οποίο αναλύεται σε 4 × 4 παράθυρα (W) και αν κάποιο από αυτά έχει τουλάχιστον ένα στοιχείο διαφορετικό του -1 οδηγείται στη συνάρτηση EncodeWindow για κωδικοποίηση. Κάθε κωδικοποιημένο παράθυρο προσαρτάται στον πίνακα X, στον οποίο αποθηκεύεται το συμπιεσμένο καρέ, και για κάθε παράθυρο που κωδικοποιείται ο μετρητής Count αυξάνει κατά 1. Τέλος, όταν τελειώσει η διαδικασία για όλα τα παράθυρα, στον πίνακα X προσαρτάται και η τιμή του μετρητή.

Η συνάρτηση EncodeWindow( ) κωδικοποιεί τα 4 × 4 παράθυρα (W) που παίρνει ως είσοδο από τη συνάρτηση Compress( ). Τα κωδικοποιημένα παράθυρα αποθηκεύονται στον πίνακα U. Για κάθε παράθυρο, αρχικά αποθηκεύονται στον U οι συντεταγμένες του στο καρέ. Στη συνέχεια, γίνεται έλεγχος της τιμής κάθε

εικονοστοιχείου των παραθύρων και ανάλογα με το αν είναι ίση με το – 1 ή όχι προσαρτώνται στον U οι κατάλληλες τιμές, όπως αυτές έχουν ήδη περιγραφεί. Όταν ελεγχθούν οι τιμές ενός ολόκληρου παραθύρου, ο πίνακας U μπορεί να χρησιμοποιηθεί από τη συνάρτηση Compress( ).

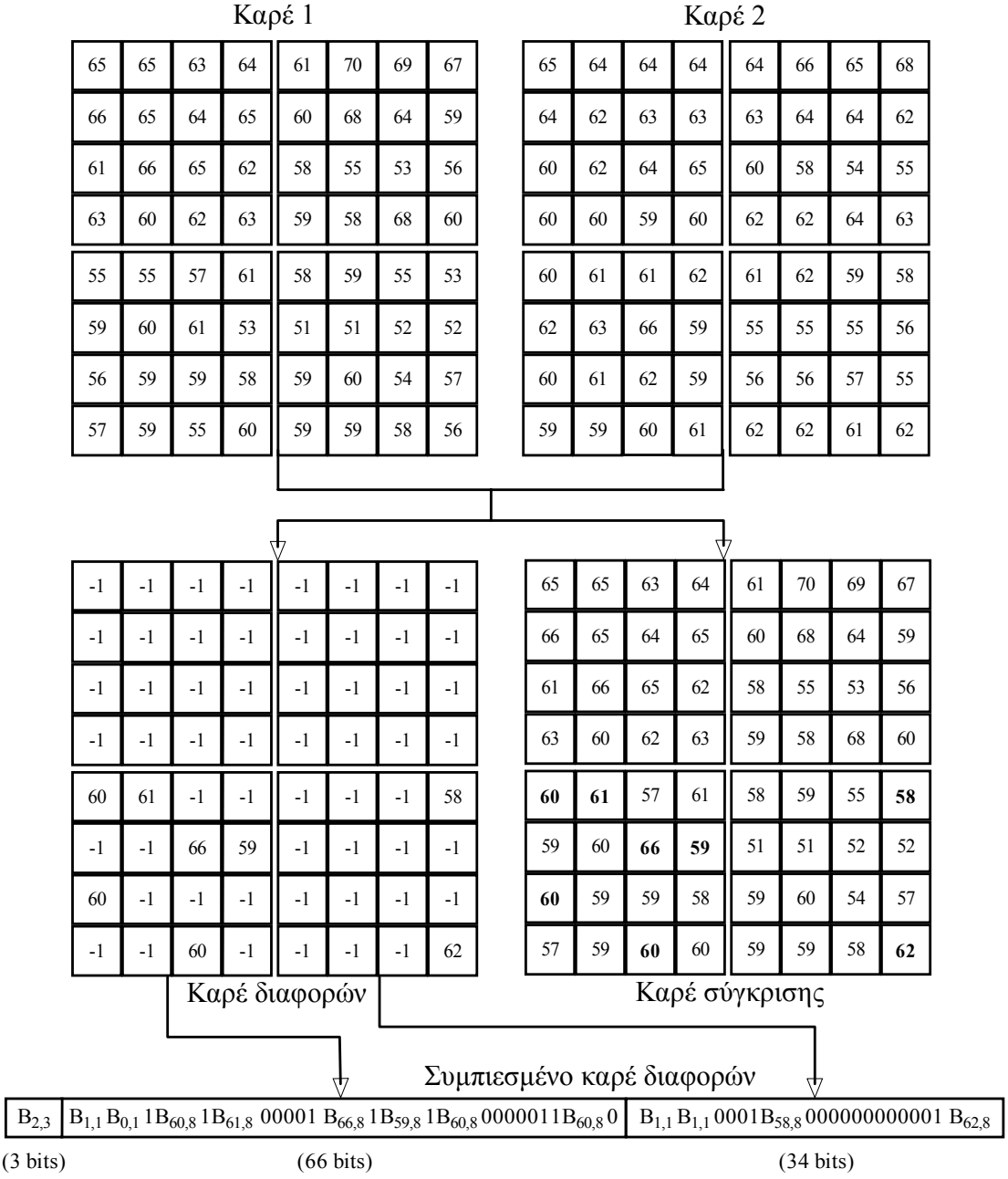
Τέλος, η συνάρτηση VideoEncrypt καλεί όταν χρειάζεται τις συναρτήσεις Difference( ) και Compress( ) και κρυπτογραφεί το συμπιεσμένο video. Η διαδικασία της κρυπτογράφησης δεν κρίνεται σκόπιμο να αναλυθεί.

Πρέπει να σημειωθεί ότι η συνάρτηση Difference( ) περιλαμβάνει απωλεστική λειτουργία η οποία κάνει την συμπίεση video απωλεστική. Ταυτόχρονα όμως εγγυάται ότι το μέγεθος της διαφοράς μεταξύ της τιμής ενός εικονοστοιχείου στο αρχικό video και της αντίστοιχης τιμής του εικονοστοιχείου στο ανακτώμενο video στον δέκτη είναι το πολύ ίση με την τιμή κατωφλίου TH. Χρησιμοποιώντας υψηλή τιμή κατωφλίου επιτυγχάνεται υψηλότερος βαθμός συμπίεσης, αλλά υποβαθμίζεται περισσότερο η ποιότητα του video.

Στο Σχήμα 3.2 παρουσιάζεται ένα παράδειγμα της λειτουργίας του αλγορίθμου. Στο παράδειγμα αυτό φαίνεται η διαδικασία που ακλουθείται για τη συμπίεση των δύο πρώτων καρέ μιας ακολουθίας video ανάλυσης 8 × 8 εικονοστοιχείων × 8 bits / εικονοστοιχείο για τιμή κατωφλίου ίση με 4.

Το πρώτο καρέ δίνεται σαν είσοδος στη συνάρτηση Difference( ) και ανατίθεται στον πίνακα T, σχηματίζεται δηλαδή το πρώτο καρέ σύγκρισης. Όταν στη συνάρτηση αυτή εισάγεται το δεύτερο καρέ, αυτό συγκρίνεται με το καρέ σύγκρισης που στη συγκεκριμένη περίπτωση ταυτίζεται με το πρώτο καρέ. Από τη σύγκριση προκύπτουν τα καρέ διαφορών και σύγκρισης. Το καρέ σύγκρισης θα συγκριθεί με το τρίτο καρέ της ακολουθίας video και το καρέ διαφορών οδηγείται για συμπίεση.

Το καρέ διαφορών δίνεται ως είσοδος στη συνάρτηση Compress( ). Χωρίζεται σε τέσσερα παράθυρα 4 × 4 και σε καθένα από αυτά γίνεται έλεγχος για στοιχεία διαφορετικά του – 1. Τα δύο πρώτα παράθυρα περιέχουν μόνο στοιχεία ίσα με το – 1, οπότε δεν στέλνονται για κωδικοποίηση και δεν κρατιέται καμία πληροφορία γι' αυτά. Τα υπόλοιπα δύο περιέχουν και εικονοστοιχεία με τιμές μεταξύ του 0 και του 255 και στέλνονται για κωδικοποίηση.



Σχήμα 3.2 Παράδειγμα της λειτουργίας του αλγορίθμου για δύο καρέ μεγέθους 8 × 8 εικονοστοιχείων

Το πρώτο παράθυρο από αυτά (το κάτω αριστερά) δίδεται ως είσοδος στη συνάρτηση EncodeWindow( ). Εκεί αρχικά καταγράφονται οι συντεταγμένες του στο καρέ. Πιο συγκεκριμένα, αυτές είναι οι (x, y) = (1, 0). Για την απεικόνισή τους χρειάζονται 1 + 1 bits διότι το μέγεθος του καρέ είναι μόλις 8 × 8 και χωρούν συνολικά τέσσερα παράθυρα σε αυτό. Οι συντεταγμένες αυτές είναι οι $B_{1,1}$ και $B_{0,1}$ στο κωδικοποιημένο παράθυρο. Στη συνέχεια, κάθε τιμή – 1 αντιστοιχεί σε ένα 0 στο κωδικοποιημένο παράθυρο και κάθε τιμή διαφορετική του – 1 αντιστοιχεί ένα 1 ακολουθούμενο από

την τιμή του εικονοστοιχείου απεικονισμένη με 8 bits. Η μορφή του κωδικοποιημένου παραθύρου φαίνεται στο Σχήμα 3.2.

Μετά το πέρας της κωδικοποίησης του πρώτου από τα δύο παράθυρα αυξάνει κατά 1 ο μετρητής Count στη συνάρτηση Compress( ) και με τον ίδιο ακριβώς τρόπο κωδικοποιείται και το τελευταίο παράθυρο. Ο μετρητής Count αυξάνει για μία ακόμη φορά κατά 1 και η τιμή του τοποθετείται μπροστά από τις ακολουθίες των bits που αναπαριστούν τα κωδικοποιημένα παράθυρα. Η τιμή του μετρητή είναι 2 διότι τόσα παράθυρα του τρέχοντος καρέ υπέστησαν κωδικοποίηση και απεικονίζεται με 3 bits. Με την προσθήκη και της τιμής του μετρητή το συμπιεσμένο καρέ των διαφορών έχει πάρει πλέον την τελική του μορφή.

Σε αυτό το παράδειγμα ο λόγος συμπίεσης είναι (4 παράθυρα / καρέ × 16 εικονοστοιχεία / παράθυρο × 8 bits / εικονοστοιχείο) / (3 + 66 + 34 bits) ή περίπου 5/1. Κρυπτογραφώντας τα συμπιεσμένα καρέ διαφορών επιτυγχάνονται ταυτόχρονα γρήγορη κρυπτογράφηση και συμπίεση. Για να ανακτηθεί το video στον δέκτη, το συμπιεσμένο και κρυπτογραφημένο video πρώτα αποκρυπτογραφείται για να προκύψει το συμπιεσμένο video, το οποίο στη συνέχεια αποσυμπιέζεται για να προκύψουν τα καρέ των διαφορών. Αυτά τα καρέ χρησιμοποιούνται για την ανακατασκευή του video. Σημειώνεται ότι οι παράμετροι N, w, h, m, n και K στέλνονται στο δέκτη ξεχωριστά, πριν την μετάδοση του συμπιεσμένου και κρυπτογραφημένου video.

3.3 Πειραματικά αποτελέσματα

Υπενθυμίζεται ότι οι δοκιμές αυτές αφορούν τη software υλοποίηση του αλγορίθμου και δεν έγιναν στα πλαίσια της παρούσας εργασίας, αλλά παρατίθενται όπως δημοσιεύονται στο [34]. Η προτεινόμενη μέθοδος κρυπτογράφησης που χρησιμοποιείται σε συνδυασμό με τη συμπίεση δοκιμάστηκε με αρκετές ακολουθίες video. Στο Σχήμα 3.3 φαίνονται αντιπροσωπευτικά καρέ από τέσσερις από αυτές τις ακολουθίες, καθεμιά από τις οποίες αποτελείται από είκοσι καρέ. Ο Πίνακας 3.1 δείχνει το ποσοστό συμπίεσης και το μέσο τετραγωνικό σφάλμα μεταξύ των αρχικών και των ανακτώμενων ακολουθιών video για διάφορες τιμές κατωφλίου. Στο Σχήμα 3.4 παρουσιάζονται πέντε καρέ από την ακολουθία video «Claire», το

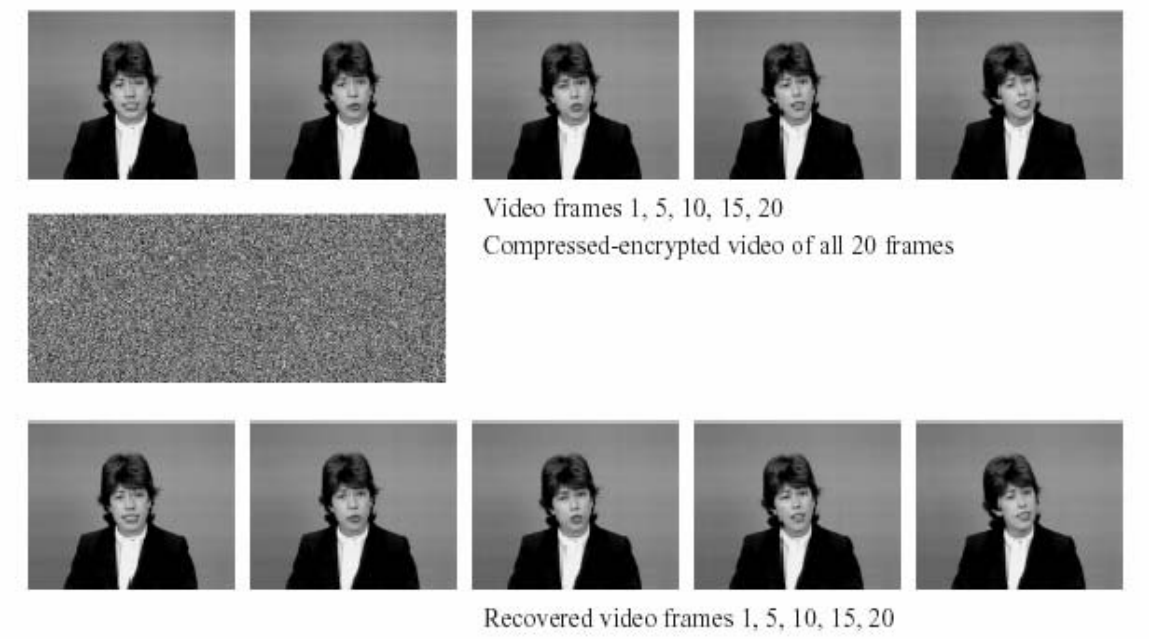
Σχήμα 3.3 Δοκιμαστικά video

Τιμή κατω-φλίου	Claire		Trevor		Discussion		Heart	
	% Συ-μπίεση	Σφά-λμα	% Συ-μπίεση	Σφά-λμα	% Συ-μπίεση	Σφά-λμα	% Συ-μπίεση	Σφά-λμα
0	11,05	0,0000	3,74	0,0000	4,82	0,0000	37,33	0,0000
1	52,26	0,6363	31,73	0,5041	29,27	0,5732	45,47	0,0265
2	73,09	1,0413	48,20	0,9023	50,96	1,0693	50,24	0,4610
3	82,65	1,3416	59,99	1,1752	66,67	1,4481	73,30	1,5104
4	87,24	1,5500	65,31	1,3684	75,70	1,7228	74,65	1,6293
5	89,60	1,7065	67,79	1,5469	80,02	1,9384	77,32	1,8410
6	90,96	1,8374	69,57	1,7417	82,38	2,1317	85,19	2,4320
7	91,89	1,9648	71,12	1,9534	83,95	2,3177	86,92	2,6047
8	92,61	2,0912	72,52	2,1785	85,25	2,5058	88,01	2,8732
9	93,24	2,2191	73,83	2,4140	86,30	2,6978	89,49	2,9241
10	93,77	2,3532	75,07	2,6637	87,21	2,8995	90,17	3,2005

Πίνακας 3.1 Ποσοστό συμπίεσης και μέσο τετραγωνικό σφάλμα για διάφορες τιμές κατωφλίου

συμπιεσμένο και κρυπτογραφημένο video από τα είκοσι καρέ, καθώς και τα αντίστοιχα πέντε ανακτώμενα καρέ όταν η τιμή κατωφλίου είναι πέντε [34].

Ας σημειωθεί ότι το μέσο τετραγωνικό σφάλμα μετράει τη μέση διαφορά μεταξύ του αρχικού και του ανακτώμενου video. Υψηλότερες τιμές κατωφλίου παράγουν μεγαλύτερη συμπίεση και περισσότερη υποβάθμιση των video, αλλά η συμπίεση μεγαλώνει ταχύτερα από ότι το μέσο σφάλμα. Για παράδειγμα, κατά μέσο όρο, τιμή κατωφλίου ίση με πέντε παράγει περίπου 78% συμπίεση video με μέσο σφάλμα μόνο περίπου 1.67 (και μέγιστο σφάλμα ίσο με 5).



Σχήμα 3.4 Συμπίεση και κρυπτογράφηση της ακολουθίας video «Claire»

3.4 Επίλογος

Στο κεφάλαιο αυτό έγινε εκτενής ανάλυση του αλγορίθμου SCAN για συμπίεση video και παρατέθηκαν κάποια πειραματικά αποτελέσματα σχετικά με την απόδοσή του, όπως αυτά παρουσιάζονται στη δημοσίευση [34]. Από όσα έχουν αναφερθεί έως τώρα, συμπεραίνεται ότι ο αλγόριθμος SCAN χρησιμοποιεί μέθοδο συμπίεσης INTER. Επίσης, εφόσον η κωδικοποίηση γίνεται σε παράθυρα ο αλγόριθμος αυτός βασίζεται σε συγκροτήματα, αλλά η προσέγγιση της κωδικοποίησης του κάθε παραθύρου είναι τελείως διαφορετική από αυτή που παρουσιάστηκε στην παράγραφο 2.3.3.

Κεφάλαιο 4

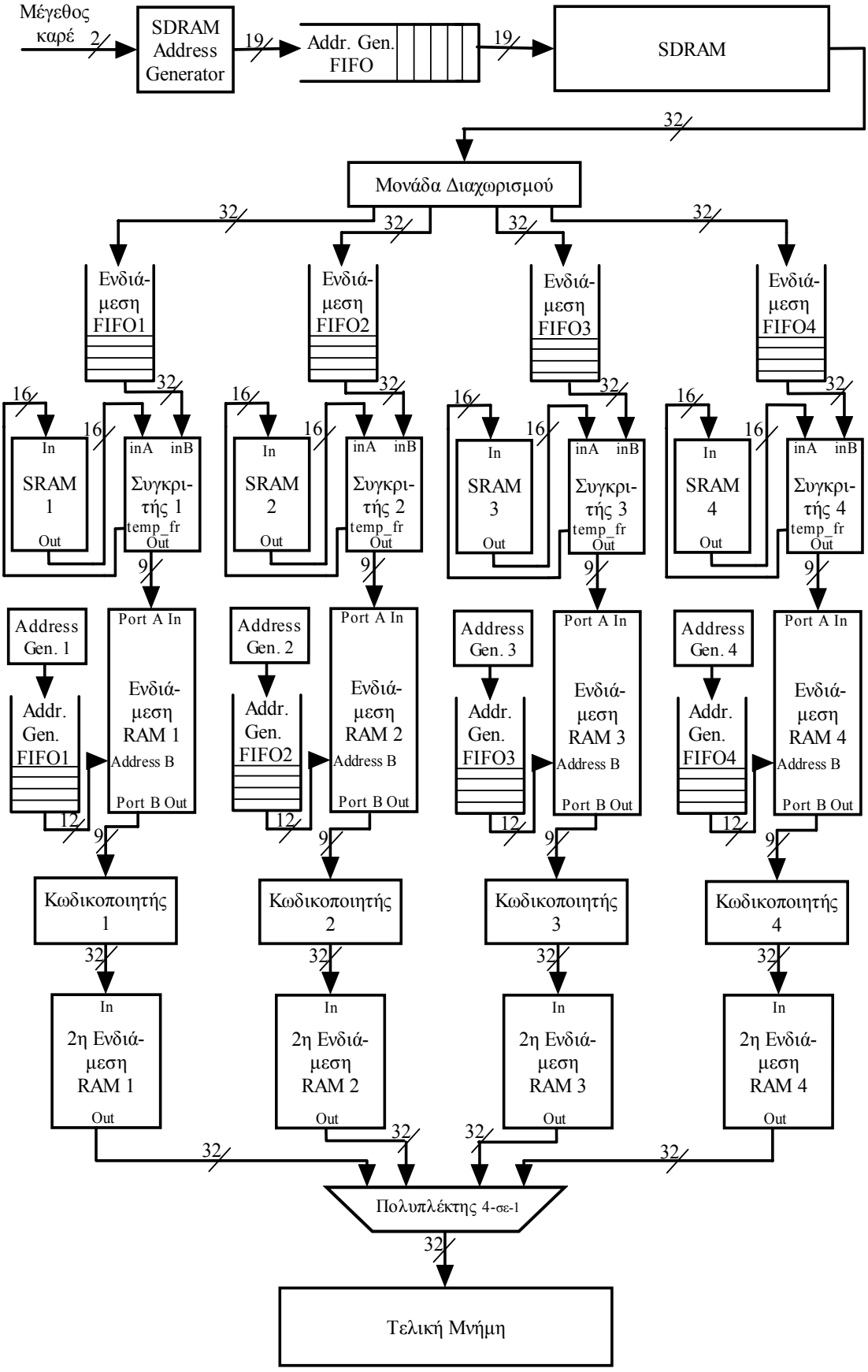
Η αρχιτεκτονική του συστήματος

Σε αυτό το τμήμα της εργασίας γίνεται παρουσίαση της αρχιτεκτονικής που αναπτύχθηκε, προκειμένου να επιτευχθεί η υλοποίηση του αλγορίθμου SCAN για συμπίεση video σε hardware. Η όλη αρχιτεκτονική σχεδιάστηκε με έμφαση την απόδοση, αλλά και την αποφυγή κατασπατάλησης πόρων, οπότε έγινε προσπάθεια να χρησιμοποιηθεί όση λιγότερη μνήμη RAM ήταν δυνατό. Στην παράγραφο 4.1 παρουσιάζεται η γενική αρχιτεκτονική του συστήματος, ενώ στις υπόλοιπες παραγράφους αναλύεται το κάθε κομμάτι της αρχιτεκτονικής αυτής.

4.1 Γενική εποπτεία της αρχιτεκτονικής

Η συνολική σχεδίαση αποτελείται από το τμήμα της μνήμης SDRAM (που αποτελείται από τη μνήμη SDRAM και τον ελεγκτή της), από έναν address generator για τη μνήμη SDRAM, μία μονάδα διαχωρισμού των καρέ σε παράθυρα, τέσσερις μονάδες σύγκρισης των καρέ, τέσσερις μονάδες κωδικοποίησης των καρέ, τέσσερις ακόμη address generators (διαφορετικούς από τον address generator της SDRAM), τέσσερις μνήμες τύπου SRAM και μεγέθους 32K × 16 bits, τέσσερις μνήμες Dual Port RAM μεγέθους 4K × 9 bits, τέσσερις μνήμες RAM μεγέθους 2434 × 32 bits, καθώς και εννέα FIFOs, μεγέθους 16 × 8 bits. Το block διάγραμμα της αρχιτεκτονικής παρουσιάζεται στο Σχήμα 4.1 Κάποια μικρά κομμάτια της αρχιτεκτονικής, όπως ορισμένοι address counters, έχουν παραλειφθεί από το διάγραμμα ώστε αυτό να μη γίνει πολύπλοκο και θα φανούν αναλυτικά στις επόμενες παραγράφους.

Τα καρέ του προς συμπίεση video είναι αποθηκευμένα στη μνήμη SDRAM. Με τη βοήθεια του address generator τα δεδομένα εξάγονται από τη μνήμη αυτή στην κατάλληλη σειρά και οδηγούνται στη μονάδα διαχωρισμού όπου χωρίζονται σε τέσσερα τμήματα. Το κάθε τμήμα αποτελείται από τα εικονοστοιχεία ενός



Σχήμα 4.1 Block διάγραμμα της αρχιτεκτονικής

συγκεκριμένου παραθύρου μεγέθους 64 × 64 του τρέχοντος καρέ. Όλα τα παράθυρα του πρώτου καρέ αποθηκεύονται στις αντίστοιχες μνήμες SRAM, ώστε να συγκριθούν με αυτά του επόμενου. Όταν στην είσοδο των συγκριτών φτάσουν δεδομένα από το δεύτερο καρέ, αυτά συγκρίνονται με τα περιεχόμενα της SRAM και προκύπτει τόσο ένα τμήμα του καρέ διαφορών, όσο κι ένα τμήμα του καρέ σύγκρισης, το οποίο αποθηκεύεται στη μνήμη SRAM, πανωγράφοντας τα περιεχόμενά της. Το κάθε τμήμα του καρέ διαφορών αποθηκεύεται σε Dual Port RAM, απ' όπου, χρησιμοποιώντας τις διευθύνσεις που παράγει ο αντίστοιχος address generator χωρίζονται σε παράθυρα 4 × 4 και οδηγούνται προς τον κωδικοποιητή για συμπίεση και κωδικοποίηση, σύμφωνα με όσα έχουν αναφερθεί στο κεφάλαιο 3. Τα αποτελέσματα των κωδικοποιητών γράφονται σε τέσσερις Dual Port RAMs. Όταν τελειώσει η διαδικασία για τα τρέχοντα παράθυρα, τα επόμενα τέσσερα όταν θα αρχίσουν να κωδικοποιούνται θα αποθηκεύονται σε διαφορετικό τμήμα των Dual Port RAMs απ' αυτό που είναι αποθηκευμένα τα τέσσερα προηγούμενα. Αυτό γίνεται διότι για να γραφούν τα συμπιεσμένα παράθυρα των καρέ σειριακά στην τελική μνήμη χωρίς κατασπατάληση χώρου πρέπει να είναι γνωστό το μέγεθος του καθενός, κάτι που μπορεί να γίνει μόνο αφού τελειώσει η επεξεργασία του. Τότε, με τη βοήθεια ενός πολυπλέκτη τα αποτελέσματα γράφονται στην τελική μνήμη.

Τα αρχικά δεδομένα επιλέχτηκε να είναι αποθηκευμένα σε μνήμη τύπου SDRAM, διότι η συγκεκριμένη μνήμη μπορεί να έχει αρκετά μεγάλο μέγεθος χωρίς να είναι ιδιαίτερα ακριβή. Παρ' όλα αυτά, το μέγεθος της SDRAM στην παρούσα σχεδίαση είναι σχετικά μικρό, 4 MB συνολικά, διότι αρκεί για την αποθήκευση αρκετών καρέ, αλλά ταυτόχρονα υπάρχει δυνατότητα με κάποιες μικρές παρεμβάσεις στον ελεγκτή της το μέγεθός της να πάει ακόμα και στα 512 MB. Κάτι τέτοιο θα ήταν αδύνατο αν αντί για αυτή είχε χρησιμοποιηθεί SRAM.

Οι υποστηριζόμενες αναλύσεις των προς συμπίεση video είναι 128 × 128, 256 × 256 και 512 × 512 εικονοστοιχεία, ενώ το κάθε εικονοστοιχείο αναπαριστάται με ακρίβεια 8 bits. Δεν έγινε περιορισμός μόνο σε μία από τις παραπάνω διαστάσεις για λόγους ευελιξίας. Επίσης, με τη χρήση μεγαλύτερης μνήμης SDRAM και με κάποιες προσθήκες ή τροποποιήσεις στον SDRAM address generator είναι δυνατό να υποστηριχτούν ακόμα μεγαλύτερες αναλύσεις (π.χ. 1024 × 1024) ή και video με αναλογία πλευρών 4 : 3.

Όπως αναφέρθηκε και στη γενική περιγραφή της λειτουργίας του συστήματος, κάθε καρέ του video, ό,τι μεγέθους κι αν είναι αυτό, χωρίζεται σε παράθυρα των 64 × 64 εικονοστοιχείων. Αυτό γίνεται κυρίως για λόγους παραλληλισμού της όλης διαδικασίας, μια και υπάρχει δυνατότητα ταυτόχρονης επεξεργασίας τεσσάρων παραθύρων. Ο αριθμός τέσσερα επιλέχτηκε κυρίως λόγω του πλάτους λέξης της SDRAM, που είναι 32 bits, δηλαδή χωράει ακριβώς τις τιμές φωτεινότητας τεσσάρων εικονοστοιχείων (8 bits για το καθένα). Όσον αφορά το μέγεθος των παραθύρων, αυτό επιλέχτηκε να είναι 64 × 64 ως συμβιβασμός ανάμεσα στην ποσότητα της απαιτούμενης μνήμης που απαιτούνταν για την επεξεργασία τους και στο μέγεθος του συμπιεσμένου αρχείου. Χρησιμοποιώντας μεγαλύτερο μέγεθος παραθύρου, π.χ. 128 × 128 οι απαιτήσεις σε μνήμη γίνονταν πλέον τεράστιες, ενώ χρησιμοποιώντας μικρότερο παράθυρο στο συμπιεσμένο αρχείο εισάγεται επιπλέον πληροφορία από τα επιπλέον πεδία που χρειάζονται για να δείξουν τον αριθμό των παραθύρων που έχουν κωδικοποιηθεί, κάτι που θα φανεί καλύτερα στην ανάλυση της λειτουργίας του συστήματος και πιο συγκεκριμένα στην ανάλυση της μορφής με την οποία τα καρέ αποθηκεύονται στην τελική μνήμη. Το μέγεθος της επιπλέον αυτής πληροφορίας μπορεί να είναι από πολύ μικρό έως και αρκετά μεγάλο σε σχέση με το μέγεθος του συμπιεσμένου αρχείου, ανάλογα με το πόση συμπίεση επιτυγχάνεται.

Παρά το διαχωρισμό του κάθε καρέ σε παράθυρα, η υλοποίηση του αλγορίθμου απαιτούσε να αποθηκεύεται κάπου πληροφορία ίση με το μέγεθος ενός καρέ (του καρέ σύγκρισης). Στην περίπτωση των καρέ μεγέθους 128 × 128 εικονοστοιχείων κάτι τέτοιο ήταν εφικτό, όμως όσο αυτά μεγάλωναν απαιτούσαν αρκετή μνήμη (256 KB για καρέ 512 × 512), γεγονός που επέβαλε τη χρήση μνήμης τύπου SRAM και πιο συγκεκριμένα γρήγορης SRAM. Η SDRAM ήταν ακατάλληλη για τη χρήση αυτή λόγω της αρκετά χαμηλότερης ταχύτητάς της από την SRAM.

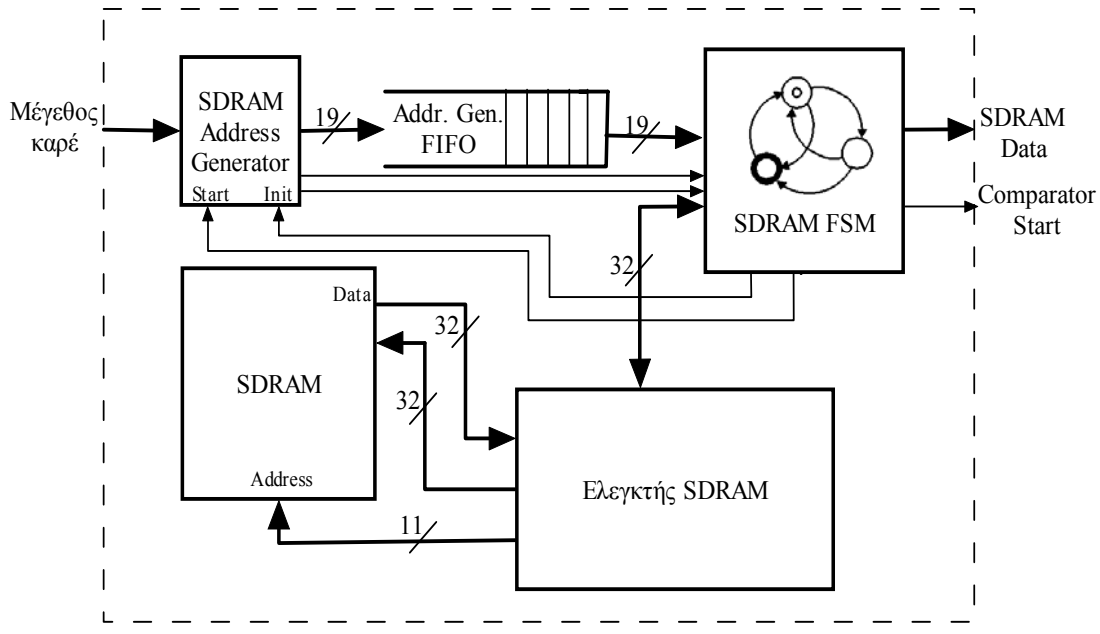
Επίσης, σκόπιμα δεν αναφέρεται πουθενά ο τύπος και το μέγεθος της τελικής μνήμης, καθώς ο ρόλος της είναι εντελώς βοηθητικός. Απλά, στο σημείο εκείνο της σχεδίασης έχει τοποθετηθεί ένα μοντέλο μνήμης ώστε να συγκεντρώνεται το συμπιεσμένο αρχείο, αλλά στην πράξη η έξοδος του συστήματος είναι μία ροή από bits (η έξοδος του πολυπλέκτη 4-σε-1) κι ένα σήμα που δηλώνει αν τα δεδομένα είναι έγκυρα ή όχι. Αυτή η ροή από bits μπορεί να οδηγηθεί στο σύστημα της κρυπτογράφησης.

4.2 Το υποσύστημα της μνήμης SDRAM

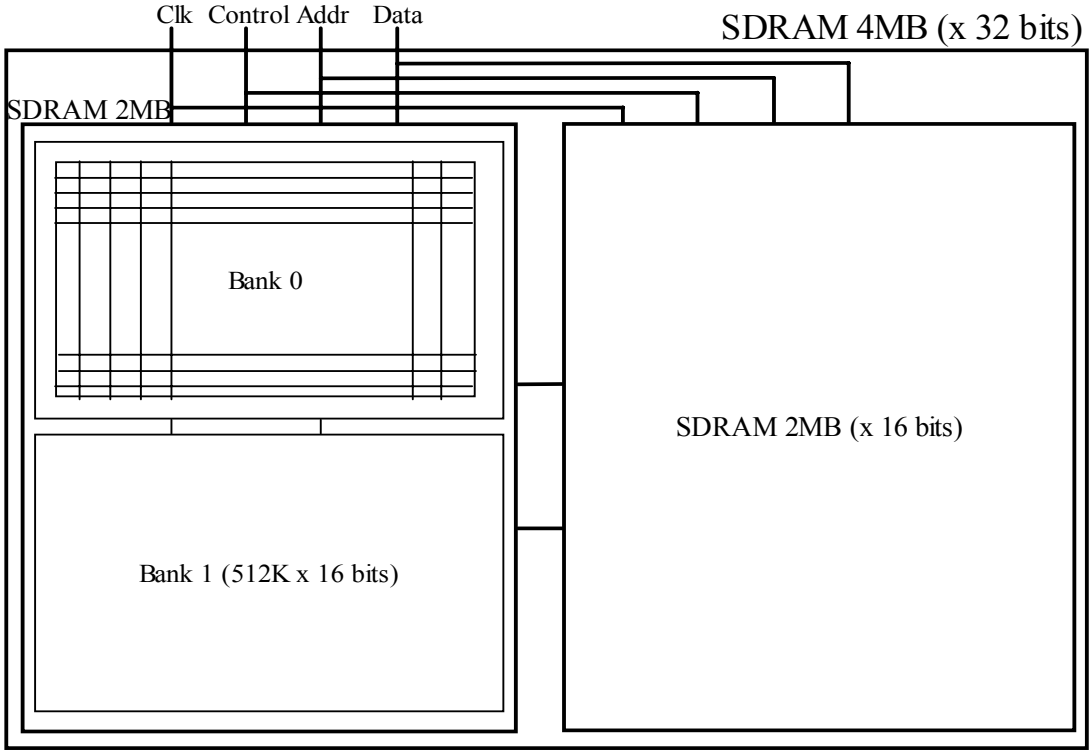
Το υποσύστημα της μνήμης SDRAM αποτελείται από τη μνήμη SDRAM με τον ελεγκτή της, καθώς και από τον SDRAM address generator και την αντίστοιχη FIFO, ενώ η όλη διαδικασία ελέγχεται από μία FSM. Οι μονάδες αυτές θεωρούνται σαν ενιαίο υποσύστημα διότι τροφοδοτούν με δεδομένα το υπόλοιπο σύστημα. Το block διάγραμμα του υποσυστήματος αυτού παρουσιάζεται στο Σχήμα 4.2.

4.2.1 Η μνήμη SDRAM

Το υποσύστημα, που περιέχει αποθηκευμένα τα απαιτούμενα καρέ του video είναι, η μνήμη SDRAM (Synchronous Dynamic Random Access Memory). Αυτή απαρτίζεται από δύο πανομοιότυπες SDRAMs μεγέθους 2 MB και πλάτους λέξης 16 bits. Στο παρόν σύστημα χρησιμοποιήθηκαν μνήμες της εταιρίας Micron [35], ενώ της ίδιας εταιρίας είναι και το μοντέλο προσομοίωσης των μνημών που χρησιμοποιήθηκε και είναι γραμμένο σε γλώσσα Verilog. Η κάθε μία αποτελείται από 2 τράπεζες (banks) των 512K × 16 bits που είναι οργανωμένες ως 2.048 γραμμές επί 256 στήλες επί 16 bits, όπως φαίνεται και στο Σχήμα 4.3. Οι μνήμες αυτές είναι σύγχρονες με όλα τους τα σήματα να καταχωρούνται στη θετική ακμή του ρολογιού.



Σχήμα 4.2 Block διάγραμμα του υποσυστήματος της μνήμης SDRAM



Σχήμα 4.3 Η δομή της μνήμης SDRAM. Αποτελείται από δύο μικρότερες μνήμες με την κάθε μικρότερη μνήμη να αποτελείται με τη σειρά της από δύο τράπεζες οργανωμένες ως 2.048 γραμμές επί 256 στήλες επί 16 bits

Οι υποστηριζόμενες αναλύσεις των καρέ είναι, όπως προαναφέρθηκε, 128 × 128, 256 × 256 και 512 × 512 εικονοστοιχεία. Εφόσον η τιμή φωτεινότητας του κάθε εικονοστοιχείου απεικονίζεται με 8 bits, στην κάθε θέση μνήμης χωράνε δύο εικονοστοιχεία. Συνολικά οι μνήμες μπορούν να φιλοξενήσουν έως και 16 καρέ των 512 × 512 εικονοστοιχείων, έως και 64 καρέ των 256 × 256 εικονοστοιχείων ή έως και 256 καρέ των 128 × 128 εικονοστοιχείων.

Οι προσβάσεις για ανάγνωση και εγγραφή σε μνήμες τύπου SDRAM γίνονται κατά ριπές (bursts), δηλαδή η πρόσβαση αρχίζει σε ένα επιλεγμένο σημείο της μνήμης και συνεχίζει για προγραμματισμένο αριθμό θέσεων σε μία προγραμματισμένη ακολουθία. Οι επιλογές που υπήρχαν για το μέγεθος των ριπών ήταν 1, 2, 4 και 8, από τις οποίες επιλέχθηκε το 8, εφόσον με αυτό η απόδοση μεγιστοποιείται καθώς γίνονται όσο το δυνατό λιγότερες οι καθυστερήσεις από τη στιγμή που θα δοθεί μια εντολή για πρόσβαση στη μνήμη μέχρι τη στιγμή που θα αρχίσει αυτή να εκτελείται. Επίσης, η σειρά με την οποία γίνονται οι προσβάσεις στις μνήμες σε κάθε εντολή για εγγραφή ή ανάγνωση επιλέχθηκε να είναι σειριακή, εφόσον τα δεδομένα είναι γραμμένα σειριακά στη μνήμη, όπως φαίνεται παρακάτω (στην παράγραφο 4.2.4).

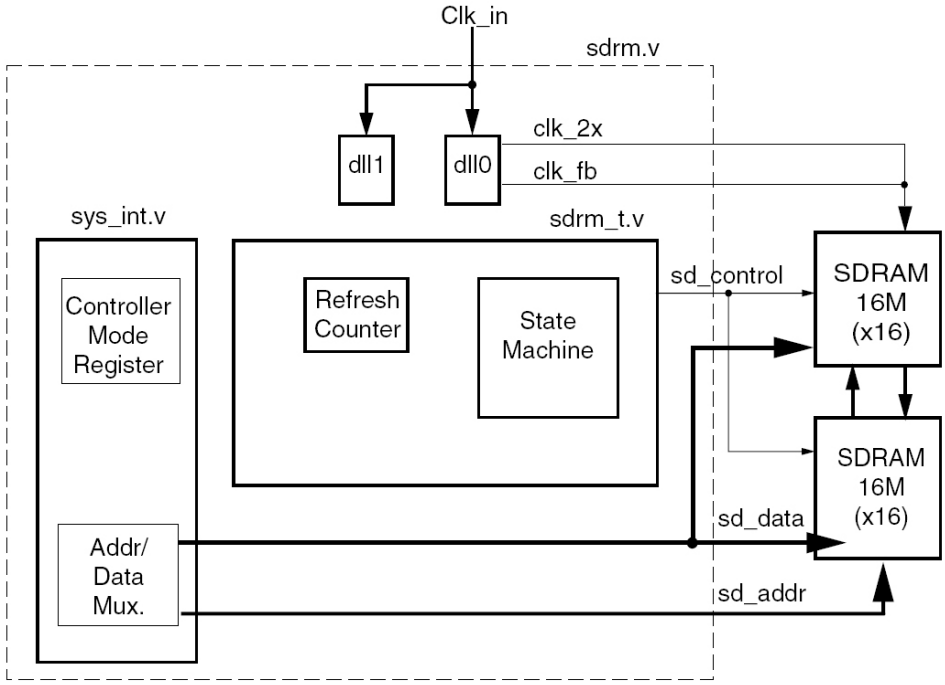
Όσον αφορά στην πρόσβαση στη μνήμη, αυτή γίνεται με εντολή για ενεργοποίηση της γραμμής που ακολουθείται από εντολή για εγγραφή ή για ανάγνωση. Ταυτόχρονα με την εντολή ενεργοποίησης γραμμής δίνεται και η διεύθυνση της επιθυμητής γραμμής, ενώ ταυτόχρονα με την εντολή για εγγραφή ή ανάγνωση δίνεται και η διεύθυνση της στήλης από την οποία θα αρχίσει η πρόσβαση. Επίσης, υπάρχει και είσοδος για την επιλογή της τράπεζας. Πρακτικά όμως, στην παρούσα υλοποίηση, αφού οι προσβάσεις στη μνήμη καθορίζονται από τον ελεγκτή της, ο οποίος επικοινωνεί με το υπόλοιπο σύστημα, δε θα μελετηθεί σε βάθος ο τρόπος με τον οποίο πραγματικά γίνονται αυτές, αλλά ο τρόπος που τις διαχειρίζεται ο ελεγκτής.

4.2.2 Ο ελεγκτής της μνήμης SDRAM

Ο ελεγκτής της μνήμης SDRAM [36] προσφέρεται από τη Xilinx® σε γλώσσα VHDL για την οικογένεια FPGAs Virtex. Από τη μία του πλευρά έχει διεπαφή προς το σύστημα, ενώ από την άλλη με δύο SDRAMs των 2 Mbytes και πλάτους λέξης 16 bits που τις διαχειρίζεται με τρόπο τέτοιο ώστε στο σύστημα να φαίνονται σαν μία ενιαία μνήμη των 4 MB με πλάτος λέξης 32 bits.

Στα χαρακτηριστικά του ελεγκτή περιλαμβάνονται, μεταξύ άλλων, προγραμματιζόμενο μήκος ριπής για πρόσβαση στη μνήμη, προγραμματιζόμενη καθυστέρηση για τη διαθεσιμότητα των δεδομένων στην έξοδο μετά από εντολή για ανάγνωση, διεπαφή με την SDRAM με ταχύτητα έως και 125 MHz και διεπαφή με το σύστημα με ταχύτητα έως και 62,5 MHz με διπλό ρυθμό δεδομένων (δηλαδή τα δεδομένα αλλάζουν τιμή τόσο στη θετική όσο και στην αρνητική ακμή του ρολογιού). Το ρολόι αυτό με τη διπλάσια συχνότητα δίνεται σαν είσοδος στο σύστημα ώστε να χρησιμοποιηθεί από κάποιες μονάδες που επεξεργάζονται τα δεδομένα που εξέρχονται από το υποσύστημα της SDRAM. Επίσης, για μηδενική ασυμμετρία (skew) του ρολογιού μέσα στην FPGA αλλά και μεταξύ αυτής και της μνήμης SDRAM γίνεται χρήση δύο DLLs (Delay Locked Loops)[1] της FPGA. Χωρίς να κρίνεται αναγκαίο να δοθούν λεπτομέρειες για την εσωτερική δομή του ελεγκτή, στο Σχήμα 4.4 παρατίθεται ένα block διάγραμμα αυτού.

[1]Κυκλώματα της FPGA που επιτρέπουν συγχρονισμό εσωτερικού κι εξωτερικού ρολογιού



Σχήμα 4.4 Block διάγραμμα του ελεγκτή μνήμης

Το σύστημα επικοινωνεί με τον ελεγκτή μέσω δύο σημάτων ελέγχου κι ενός αμφίδρομου διαύλου πλάτους 32 bits (AD[31:0]) που χρησιμοποιείται τόσο για τις διευθύνσεις, όσο και για τα δεδομένα. Έτσι, η διεύθυνση που δίδεται πολυπλέκεται στη διεύθυνση της αντίστοιχης γραμμής και στήλης της SDRAM. Πιο συγκεκριμένα το bit 21 (AD[21]) αντιστοιχεί στον αριθμό της τράπεζας (ba) της μνήμης (αφού οι δύο μνήμες φαίνονται στο σύστημα σαν μία, οι τέσσερις συνολικά τράπεζες φαίνονται σαν δύο), τα AD[20:10] αντιστοιχούν στη διεύθυνση της γραμμής της SDRAM, ενώ τα AD[9:2] αντιστοιχούν στη διεύθυνση της στήλης της μνήμης. Θεωρώντας ότι όλες οι προσβάσεις για εγγραφή και ανάγνωση είναι σε ριπές των οκτώ, τα bits AD[1:0] δε χρησιμοποιούνται. Ο Πίνακας 4.1 δείχνει την αντιστοίχιση των διευθύνσεων που δίδονται από το σύστημα σε διευθύνσεις της μνήμης SDRAM, και σημειώνεται ότι οι διευθύνσεις της SDRAM έχουν πλάτος 11 bits.

Επίσης, αναφέρεται ότι μέσω του ίδιου διαύλου, εκτός από εντολές για ανάγνωση και εγγραφή στη μνήμη δίνονται και εντολές ελέγχου και αρχικοποίησής της, όπως η φόρτωση της κατάλληλης τιμής στον καταχωρητή Mode Register που καθορίζει παραμέτρους όπως το μήκος της ριπής για την πρόσβαση στη μνήμη. Αυτό γίνεται δίνοντας τις κατάλληλες τιμές στα bits AD[29:28] και στη συνέχεια τοποθετώντας στον ίδιο δίαυλο την κατάλληλη τιμή.

Διεύθυνση SDRAM	Διεύθυνση γραμμής	Διεύθυνση στήλης
SD_A0	AD[10]	AD[2]
SD_A1	AD[11]	AD[3]
SD_A2	AD[12]	AD[4]
SD_A3	AD[13]	AD[5]
SD_A4	AD[14]	AD[6]
SD_A5	AD[15]	AD[7]
SD_A6	AD[16]	AD[8]
SD_A7	AD[17]	AD[9]
SD_A8	AD[18]	X
SD_A9	AD[19]	X
SD_A10	AD[20]	H
SD_BA	AD[21]	AD[21]

Σημείωση:
H: Υψηλή στάθμη, X: Αδιευκρίνιστη είσοδος

Πίνακας 4.1 Αντιστοίχηση των διευθύνσεων του συστήματος σε διευθύνσεις της SDRAM

Σχετικά με τη δυνατότητα ενσωμάτωσης του ελεγκτή στο σύστημα, είναι αναγκαίο να ελεγχθούν οι χρόνοι για είσοδο/έξοδο της FPGA και της SDRAM, οι οποίοι παρουσιάζονται στον Πίνακα 4.2. Το σύστημα σχεδιάστηκε για την οικογένεια FPGAs Virtex2 (πιο συγκεκριμένα για την XC2V500) [37] της Xilinx, οπότε οι χρόνοι της οικογένειας αυτής παρουσιάζονται στον πίνακα αυτό.

Ο απαιτούμενος χρόνος για την εγγραφή δεδομένων στη μνήμη ισούται με:

Virtex2 T_{AC} + SDRAM T_{SU} + καθυστέρηση διάδοσης ή

2,5 + 2,0 + 1,0 ή

5,5 ns

Ο χρόνος αυτός υπολογίστηκε θεωρώντας ότι η καθυστέρηση στη διάδοση του σήματος από την FPGA στην SDRAM είναι περίπου 1 ns, τιμή απαισιόδοξη. Έτσι, διαπιστώνεται ότι ο ελεγκτής είναι εντός προδιαγραφών, εφόσον για να μπορεί το σύστημα να λειτουργήσει στα 125 MHz ο μέγιστος κύκλος ρολογιού είναι 8 ns, τιμή

Συσκευή	T_{OH}	T_{AC}	T_{SU}	T_{HOLD}
SDRAM-8	3,0 ns	6,0 ns	2,0 ns	1,0 ns
XC2Vxxx-6	1,0 ns	2,5 ns	1,4 ns	0,0 ns

Σημειώσεις:
T_{OH}: Χρόνος συγκράτησης εξόδου
T_{AC}: Χρόνος διάδοσης του ρολογιού έως την έξοδο
T_{SU}: Χρόνος ανύψωσης
T_{HOLD}: Χρόνος συγκράτησης

Πίνακας 4.2 Χρόνοι εισόδου/εξόδου για την οικογένει FPGAs Virtex2 και για τη μνήμη SDRAM

αρκετά μεγαλύτερη από τα 5,5 ns. Όσον αφορά την ανάγνωση δεδομένων, ισχύουν τα ακόλουθα:

SDRAM T_{AC} + Virtex2 T_{SU} + καθυστέρηση διάδοσης ή

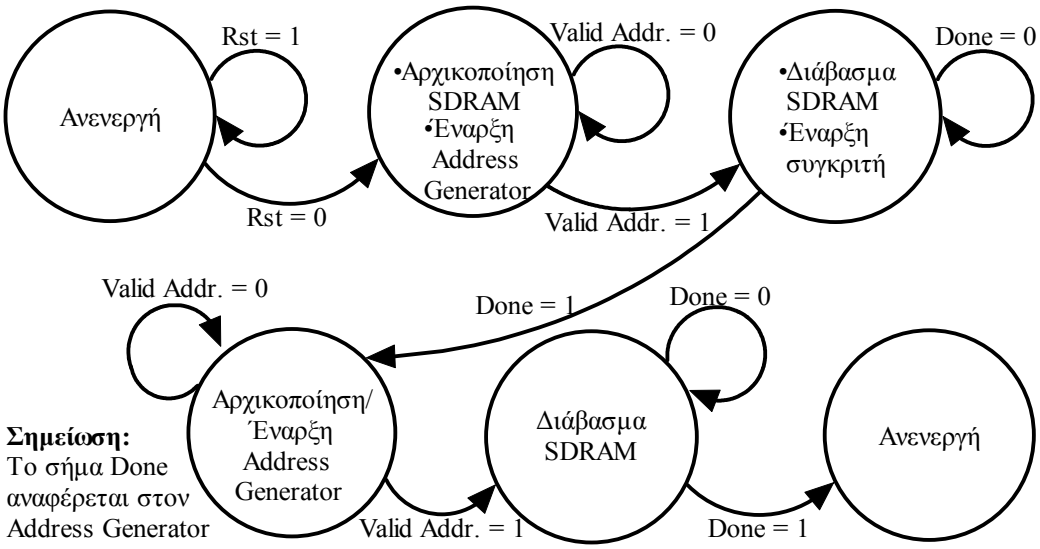
6,0 + 1,4 + 1,0 ή

8,4 ns

Ο χρόνος αυτός ίσως να φαίνεται εκτός προδιαγραφών, αλλά στην πράξη η καθυστέρηση διάδοσης είναι μικρότερη του 1 ns, οπότε οριακά επιτυγχάνεται ο μέγιστος κύκλος ρολογιού, δηλαδή τα 8 ns. Πάντως, σε κάθε περίπτωση η εν λόγω σχεδίαση λειτουργεί σε συχνότητα αρκετά χαμηλότερη από τα 125 MHz, όπως φαίνεται στο κεφάλαιο 5, οπότε δεν τίθεται θέμα δυσλειτουργίας του ελεγκτή.

4.2.3 Η FSM του υποσυστήματος της μνήμης SDRAM

Η λειτουργία του υποσυστήματος της μνήμης SDRAM ρυθμίζεται από μία FSM. Αυτή δίνει τις εντολές για ανάγνωση από τη μνήμη SDRAM, ενώ ελέγχει και τη λειτουργία του address generator για τη σωστή δημιουργία των διευθύνσεων της μνήμης από τις οποίες θα γίνει η ανάγνωση. Επίσης, την κατάλληλη στιγμή δίνει εντολή να αρχίσει τη λειτουργία του ο συγκριτής, όπως αναλύεται πιο κάτω. Στη συνέχεια θα γίνει αναλυτική περιγραφή της FSM, η οποία παρουσιάζεται στο Σχήμα 4.5.



Σχήμα 4.5 Η FSM του υποσυστήματος της μνήμης SDRAM

Το πρώτο που κάνει η FSM είναι η αρχικοποίηση της μνήμης. Μόλις το σήμα reset πάρει την τιμή μηδέν, η SDRAM πρέπει να αρχικοποιηθεί ώστε να είναι έτοιμη για ανάγνωση των δεδομένων που υπάρχουν σε αυτή. Χωρίς να κρίνεται σκόπιμη η ανάλυση όλων των σταδίων της αρχικοποίησης, σημειώνεται ότι διαρκεί περίπου 40 κύκλους ρολογιού. Λίγο πριν η αρχικοποίηση της μνήμης ολοκληρωθεί, η FSM εισάγει εντολή για αρχικοποίηση και έναρξη της λειτουργίας του address generator, ώστε μόλις η SDRAM είναι έτοιμη να αρχίσουν αμέσως οι προσβάσεις για ανάγνωση σε αυτή.

Αφού τελειώσει η αρχικοποίηση της μνήμης, όταν έρθει η πρώτη έγκυρη διεύθυνση από τον address generator μετατρέπεται στην κατάλληλη μορφή και οδηγείται στον ελεγκτή, ώστε να γίνει ανάγνωση των περιεχομένων της διεύθυνσης αυτής. Η κατάλληλη αυτή μορφή είναι η μετατροπή της διεύθυνσης των 19 bits σε 32, ώστε να μπορεί να τοποθετηθεί στο δίαυλο AD[31:0], κάτι που γίνεται τοποθετώντας μηδενικά σε ορισμένες θέσεις. Πιο συγκεκριμένα, τα bits AD[31:22] τίθενται ίσα με το μηδέν, όπως και το AD[21] που αντιστοιχεί στον αριθμό της τράπεζας. Η διεύθυνση των 19 bits τοποθετείται στα AD[20:2] (από τα οποία τα AD[20:10] αφορούν τη γραμμή, ενώ τα AD[9:2] αφορούν τη στήλη), ενώ τα AD[1:0] τίθενται ίσα με το μηδέν. Αφού γίνει η ανάγνωση, τα δεδομένα έρχονται στην FSM από τον ίδιο δίαυλο για τέσσερις κύκλους με διπλό ρυθμό και τοποθετούνται στην έξοδό της για τα δεδομένα της SDRAM. Τότε διαβάζεται η επόμενη έγκυρη διεύθυνση από τον address generator και η ίδια διαδικασία επαναλαμβάνεται μέχρι να διαβαστούν όλα τα

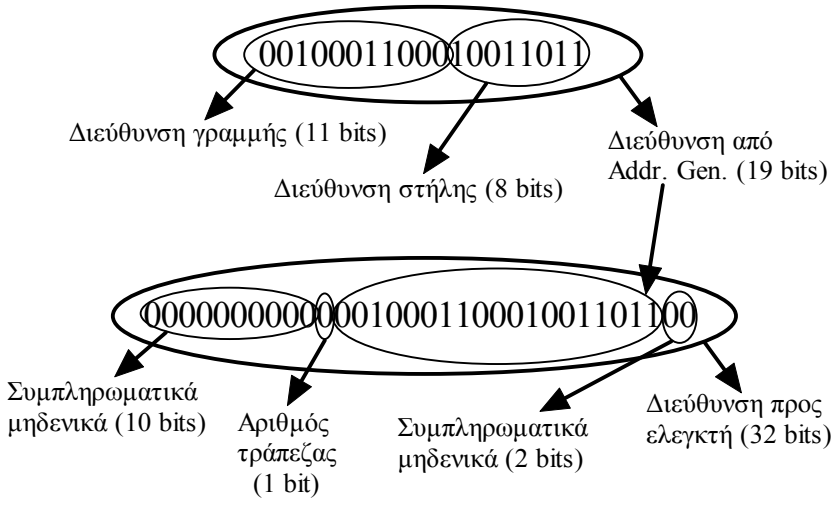
περιεχόμενα της πρώτης τράπεζας. Σημειώνεται ότι μόλις τελειώσει η δεύτερη πρόσβαση για διάβασμα στη μνήμη, δίνεται η εντολή για το ξεκίνημα του συγκριτή.

Όταν διαβαστούν όλα τα περιεχόμενα της πρώτης τράπεζας ο address generator αρχικοποιείται για μία ακόμη φορά και αρχίζει από την αρχή να δημιουργεί τις ίδιες διευθύνσεις. Με αυτές διαβάζεται και πάλι η μνήμη, μόνο που τώρα εξάγονται τα περιεχόμενα της άλλης τράπεζας. Το μόνο που αλλάζει στη λειτουργία της FSM είναι ότι το bit AD[21] έχει πλέον την τιμή ένα. Όταν ολοκληρωθεί το διάβασμα και αυτής της τράπεζας η FSM σταματά να εκτελεί κάποια λειτουργία, καθώς όλα τα δεδομένα της μνήμης έχουν εισαχθεί στο σύστημα. Η διαδικασία μετατροπής των διευθύνσεων των 19 bits σε 32 περιγράφεται στο Σχήμα 4.6.

4.2.4 Ο Address Generator της μνήμης SDRAM

Τα αποθηκευμένα δεδομένα στη μνήμη SDRAM πρέπει να διαβαστούν με απόλυτα καθορισμένο τρόπο ώστε το σύστημα να μπορεί να τροφοδοτηθεί με τα σωστά δεδομένα. Το γεγονός ότι γίνεται επεξεργασία τεσσάρων παραθύρων των καρέ παράλληλα επιβάλλει σε κάθε πρόσβαση στη μνήμη να διαβάζονται δεδομένα για διαφορετικό παράθυρο του ίδιου καρέ, ενώ η υποστήριξη τριών διαφορετικών αναλύσεων των καρέ απαιτεί την αντιμετώπιση κάθε περίπτωσης χωριστά.

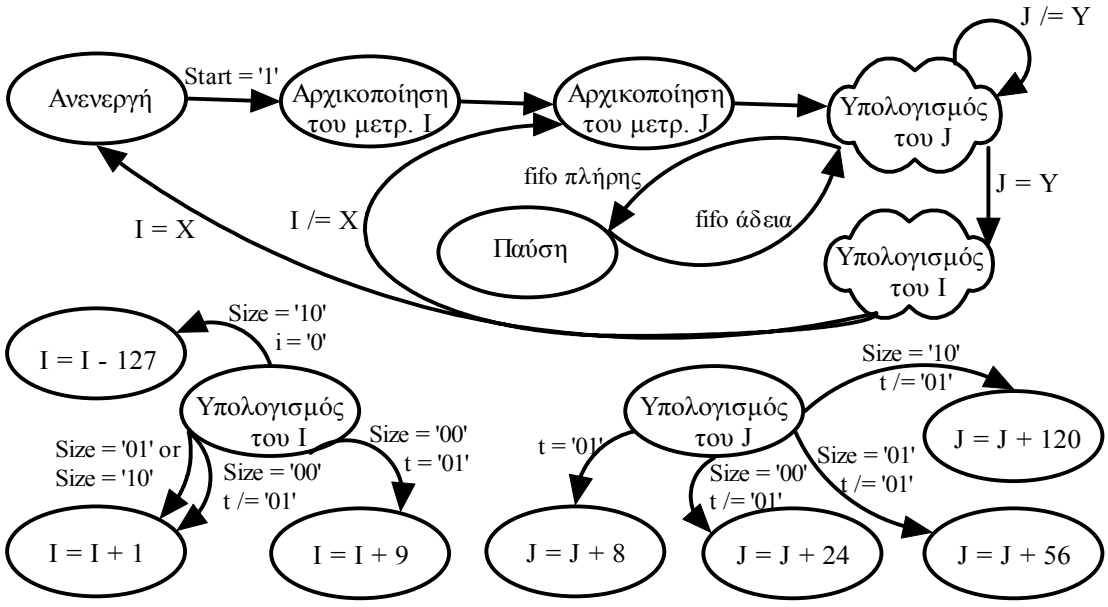
Η υλοποίηση σε hardware του address generator αυτού αποτελείται από μία FSM και μία μονάδα υπολογισμού των τελικών διευθύνσεων. Η FSM ελέγχει δύο μετρητές,



Σχήμα 4.6 Η διαδικασία μετατροπής των διευθύνσεων των 19 bits σε 32 bits

έναν που αντιστοιχεί στις γραμμές (I) κι έναν που αντιστοιχεί στις στήλες της SDRAM (J). Αυτοί συμπεριφέρονται ανάλογα με την ανάλυση των καρέ που είναι αποθηκευμένα. Οι τιμές που παίρνουν αυτοί οι μετρητές οδηγούνται στη μονάδα υπολογισμού, όπου ανάλογα με την τιμή ενός σήματος, του αριθμού μετασχηματισμού (t), και του μεγέθους των καρέ υπολογίζονται οι τελικές διευθύνσεις.

Πιο αναλυτικά, αφού αρχικοποιηθούν οι μετρητές αρχίζει ο υπολογισμός των τιμών του J, όπως φαίνεται και στο Σχήμα 4.7. Ανάλογα με την ανάλυση του καρέ (συμβολίζεται με 'Size' στο Σχήμα 4.7 και η τιμή '00' αντιστοιχεί σε ανάλυση 128 × 128 εικονοστοιχείων, η τιμή '01' σε ανάλυση 256 × 256 εικονοστοιχείων και η '10' σε ανάλυση 512 × 512 εικονοστοιχείων) και με την τιμή του αριθμού μετασχηματισμού το J αυξάνεται κατά την κατάλληλη τιμή. Στη συνέχεια θα φανεί γιατί αυτές οι τιμές είναι το 8, το 24, το 56 και το 120. Όταν η τιμή του J γίνει ίση με αυτή του Y (όπου Y ο αριθμός των στηλών της μνήμης, 256 θέσεις στην προκειμένη περίπτωση) υπολογίζεται η τιμή του I. Ανάλογα με το J, το I αυξάνεται κατά 1 ή κατά 9, ενώ στην ανάλυση των 512 × 512 σε ορισμένες περιπτώσεις μειώνεται κατά 127. Ο λόγος για τον οποίο γίνεται αυτό θα εξηγηθεί στη συνέχεια της παραγράφου. Μόλις τελειώσει η διαδικασία, αν το I δεν είναι ίσο με το X (όπου X ο αριθμός των γραμμών της μνήμης, δηλαδή 2048) αρχικοποιείται ξανά ο μετρητής J και η όλη διαδικασία επαναλαμβάνεται μέχρι το I να γίνει ίσο με το X, οπότε η διαδικασία ολοκληρώνεται.

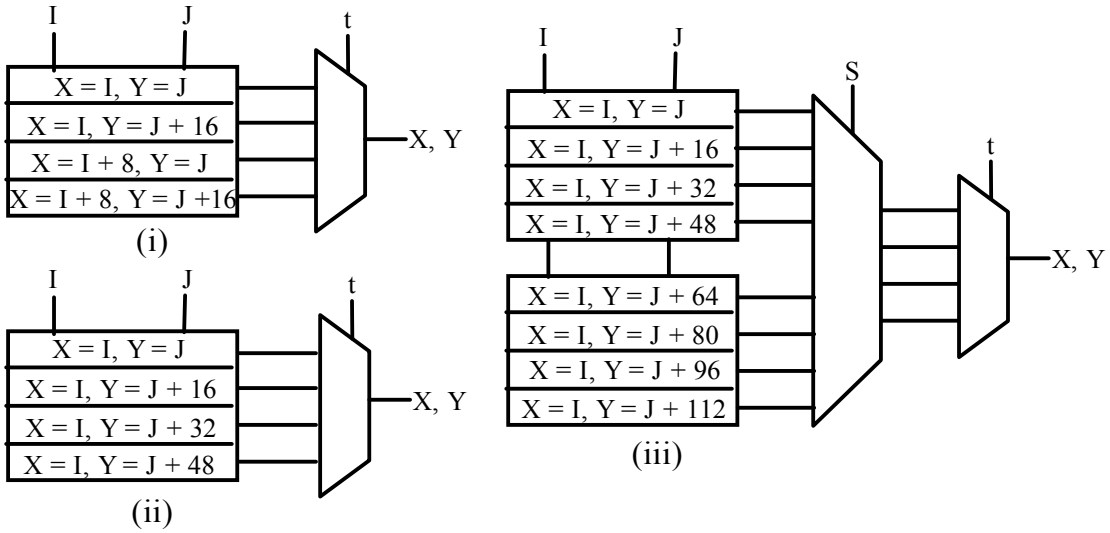


Σχήμα 4.7 Η FSM του address generator

Οι διευθύνσεις που υπολογίζονται, όπως φαίνεται και από το block διάγραμμα του υποσυστήματος της μνήμης SDRAM, τοποθετούνται σε μία FIFO από όπου τις διαβάζει ο ελεγκτής. Αυτό γίνεται διότι η παραγωγή των διευθύνσεων γίνεται πιο γρήγορα από όσο απαιτεί η SDRAM, οπότε όταν η FIFO γεμίζει ο address generator αναστέλλει τη λειτουργία του και συνεχίζει μόλις η FIFO αδειάσει. Αν δε γινόταν, όπως είναι προφανές, θα υπήρχε απώλεια διευθύνσεων, λόγω περιορισμένου χώρου της FIFO. Αυτό φαίνεται και στο Σχήμα 4.7.

Τα I και J οδηγούνται, όπως έχει αναφερθεί, στη μονάδα υπολογισμού των τελικών διευθύνσεων, η οποία παρουσιάζεται στο Σχήμα 4.8. Στο 4.8 (i) η ανάλυση των καρέ είναι 128 × 128, στο 4.8 (ii) είναι 256 × 256 και στο 4.8 (iii) είναι 512 × 512. Στις δύο πρώτες περιπτώσεις ανάλογα με την τιμή του t (αριθμός μετασχηματισμού) προκύπτουν διαδοχικά στην έξοδο οι τέσσερις τιμές των X και Y. Στην τρίτη, το ποια τετράδα διευθύνσεων θα περάσει στον τελικό πολυπλέκτη καθορίζεται από το σήμα S, το οποίο ισούται με μηδέν όταν υπολογίζονται διευθύνσεις για τις στήλες 0 έως 127 της μνήμης, ενώ ισούται με ένα όταν οι υπολογιζόμενες διευθύνσεις αφορούν τις στήλες 128 έως 255. Αξίζει να σημειωθεί οι διευθύνσεις που απαρτίζουν κάθε τετράδα διευθύνσεων αντιστοιχούν σε διαφορετικά παράθυρα του ίδιου καρέ.

Στη συνέχεια αναλύεται για κάθε ανάλυση καρέ χωριστά ο τρόπος με τον οποίο αυτό χωρίζεται σε παράθυρα, καθώς και το πως τα παράθυρα αυτά αντιστοιχίζονται σε θέσεις της SDRAM, ώστε να αιτιολογηθούν τόσο οι τιμές κατά τις οποίες



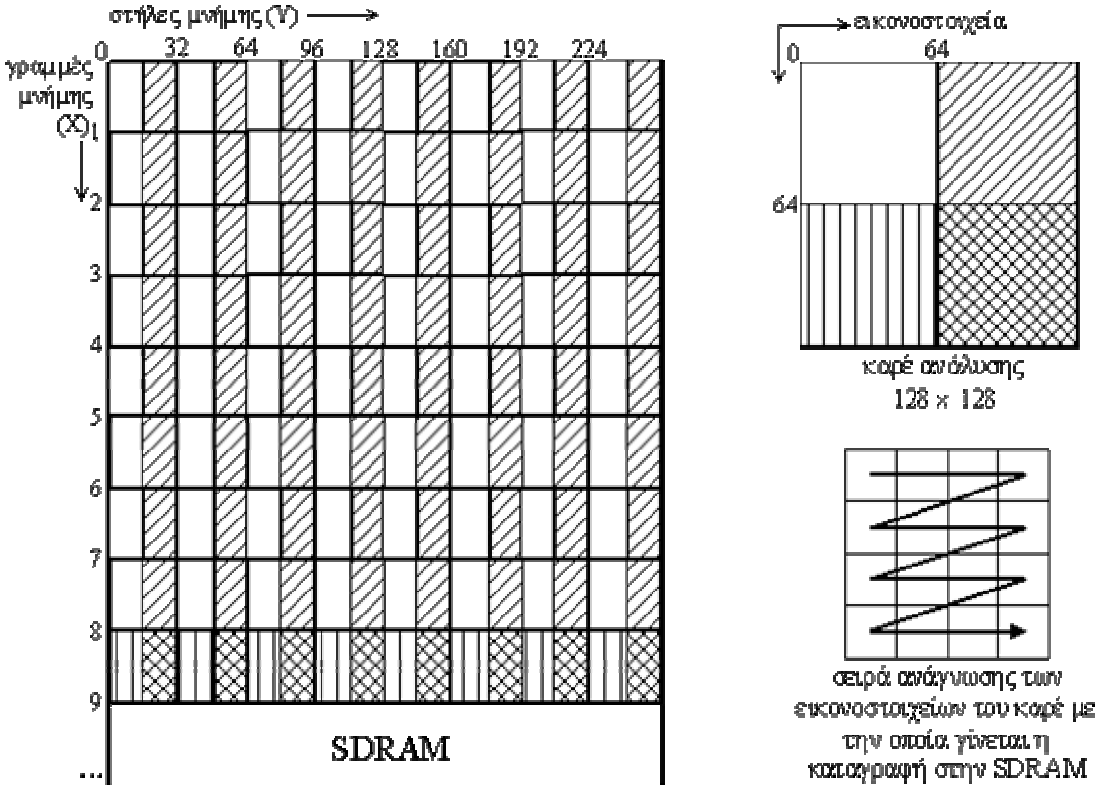
Σχήμα 4.8 Η μονάδα υπολογισμού των τελικών διευθύνσεων του address generator, όπως αυτή διαμορφώνεται για αναλύσεις των καρέ ίσες με (i) 128 × 128, (ii) 256 × 256, (iii) 512 × 512

αυξάνουν οι μετρητές I και J, όσο και οι τιμές που προστίθενται στα I, J από τη μονάδα υπολογισμού των τελικών διευθύνσεων.

1. Ανάλυση καρέ 128 × 128 εικονοστοιχεία

Υπενθυμίζεται ότι κάθε θέση της μνήμης SDRAM έχει χωρητικότητα τεσσάρων εικονοστοιχείων και ότι σε κάθε πρόσβαση σε αυτή για ανάγνωση διαβάζονται 8 συνεχόμενες θέσεις, δηλαδή 32 εικονοστοιχεία.

Το Σχήμα 4.9 δείχνει την κατανομή των 64 × 64 παραθύρων ενός καρέ 128 × 128 στη μνήμη SDRAM, καθώς και τον γραμμή προς γραμμή τρόπο με τον οποίο έχει γίνει η αποθήκευση στη μνήμη. Τα 64 εικονοστοιχεία της πρώτης γραμμής του πρώτου παραθύρου μοιράζονται στις πρώτες 16 θέσεις της μνήμης, ενώ στις επόμενες 16 θέσεις της βρίσκονται τα 64 εικονοστοιχεία της πρώτης γραμμής του δεύτερου παραθύρου, όπως φαίνεται και από τη γραμμοσκίαση. Με τον τρόπο αυτό οι 8 πρώτες γραμμές της μνήμης γεμίζουν με τα 2 πρώτα παράθυρα του καρέ. Τα δύο κάτω παράθυρα είναι αποθηκευμένα στις γραμμές 8 – 15 της μνήμης με τον τρόπο που



Σχήμα 4.9 Κατανομή ενός καρέ ανάλυσης 128 × 128 καρέ στη μνήμη SDRAM

φαίνεται στο Σχήμα 4.9. Τις επόμενες 16 γραμμές της μνήμης τις καταλαμβάνει το καρέ που ακολουθεί, κ.ο.κ.

Από την κατανομή του καρέ στη μνήμη προκύπτουν οι τιμές κατά τις οποίες αυξάνονται τα I, J, καθώς και οι τιμές που τους προστίθενται στη μονάδα υπολογισμού των τελικών διευθύνσεων. Στον Πίνακα 4.3 παρατίθενται οι πρώτες 16 διευθύνσεις που παράγονται, καθώς και ο τρόπος με τον οποίο προέκυψαν.

Σημειώνεται ότι αφού η μονάδα παραγωγής τελικών διευθύνσεων για κάθε τιμή των I, J που δέχεται παράγει διευθύνσεις που αφορούν την τρέχουσα γραμμή, αλλά και οκτώ γραμμές παρακάτω, κάθε φορά που η FSM έχει δώσει στη μονάδα αυτή δεδομένα για οκτώ γραμμές αυξάνει το μετρητή J κατά 8 και όχι κατά 1, ώστε να μην παραχθούν οι ίδιες διευθύνσεις 2 φορές.

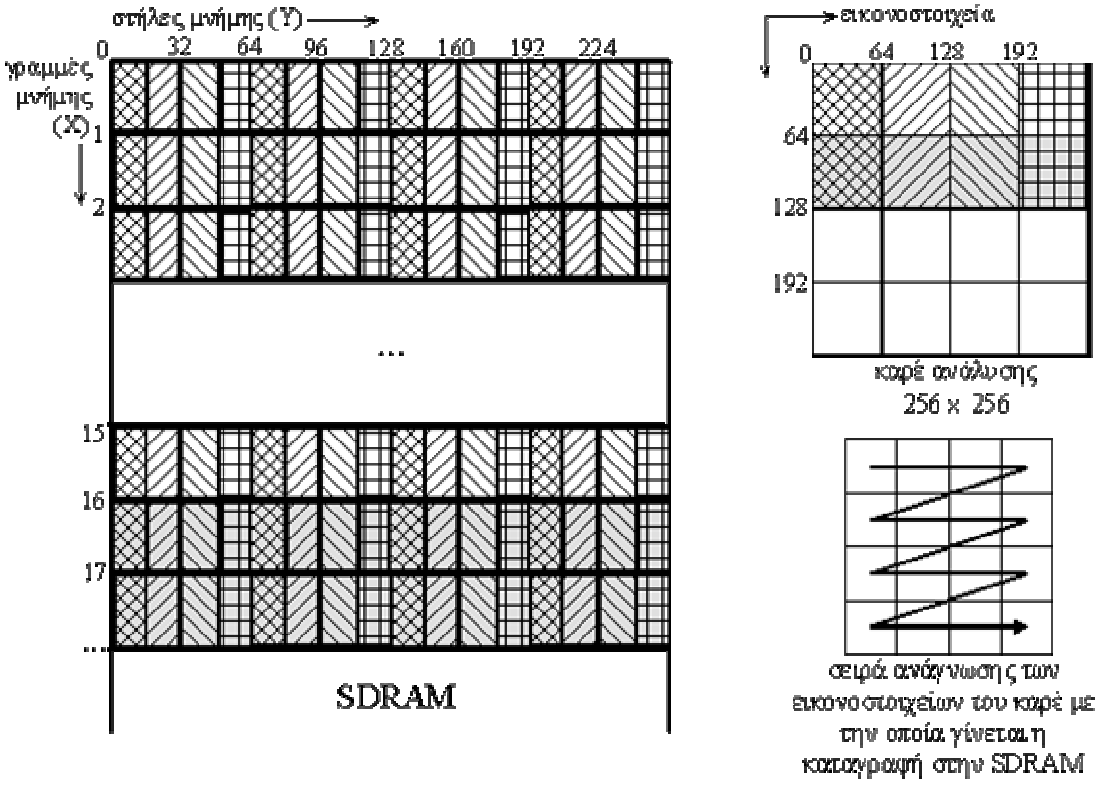
α/α	**Τιμές των I, J από την FSM**	**Τελικές Διευθύνσεις**
1	I = 0, J = 0	**X = 0, Y = 0** (X = I, Y = J)
2		**X = 0, Y = 16** (X = I, Y = J + 16)
3		**X = 8, Y = 0** (X = I + 8, Y = J)
4		**X = 8, Y = 16** (X = I + 8, Y = J + 16)
5	I = 0, J = 8 (J = J + 8)	**X = 0, Y = 8** (X = I, Y = J)
6		**X = 0, Y = 24** (X = I, Y = J + 16)
7		**X = 8, Y = 8** (X = I + 8, Y = J)
8		**X = 8, Y = 24** (X = I + 8, Y = J + 16)
9	I = 0, J = 32 (J = J + 24)	**X = 0, Y = 32** (X = I, Y = J)
10		**X = 0, Y = 40** (X = I, Y = J + 16)
11		**X = 8, Y = 32** (X = I + 8, Y = J)
12		**X = 8, Y = 40** (X = I + 8, Y = J + 16)
13	I = 0, J = 40 (J = J + 8)	**X = 0, Y = 40** (X = I, Y = J)
14		**X = 0, Y = 48** (X = I, Y = J + 16)
15		**X = 8, Y = 40** (X = I + 8, Y = J)
16		**X = 8, Y = 48** (X = I + 8, Y = J + 16)

Πίνακας 4.3 Οι 16 πρώτες διευθύνσεις και ο τρόπος παραγωγής τους για ανάλυση καρέ 128 × 128

2. Ανάλυση καρέ 256 × 256 εικονοστοιχεία

Στο Σχήμα 4.10 φαίνεται η κατανομή των 64 × 64 παραθύρων ενός καρέ 256 × 256 στη μνήμη SDRAM, καθώς και ο γραμμή προς γραμμή τρόπος με τον οποίο έχει γίνει η αποθήκευση στη μνήμη. Τα 64 εικονοστοιχεία της πρώτης γραμμής του παραθύρου που βρίσκεται πάνω αριστερά στο καρέ μοιράζονται στις πρώτες 16 θέσεις της μνήμης, ενώ στις επόμενες 48 θέσεις της βρίσκονται τα 64 εικονοστοιχεία των πρώτων γραμμών των επόμενων παραθύρων, όπως φαίνεται και από τη γραμμοσκίαση. Με τον τρόπο αυτό οι 16 πρώτες γραμμές της μνήμης γεμίζουν με τα 4 πρώτα παράθυρα του καρέ. Στις γραμμές της μνήμης 16 – 63 ακολουθούν τα επόμενα 12 παράθυρα του καρέ, ενώ τις γραμμές 64 – 127 καταλαμβάνει το καρέ που ακολουθεί, κ.ο.κ.

Όμοια με τα καρέ ανάλυσης 128 × 128, από την κατανομή του καρέ στη μνήμη προκύπτουν οι τιμές κατά τις οποίες αυξάνονται τα I, J, καθώς και οι τιμές που τους προστίθενται στη μονάδα υπολογισμού των τελικών διευθύνσεων. Στον Πίνακα 4.4 παρατίθενται οι πρώτες 12 διευθύνσεις που παράγονται, καθώς και ο τρόπος με τον οποίο προέκυψαν.



Σχήμα 4.10 Κατανομή ενός καρέ ανάλυσης 256 × 256 στη μνήμη SDRAM

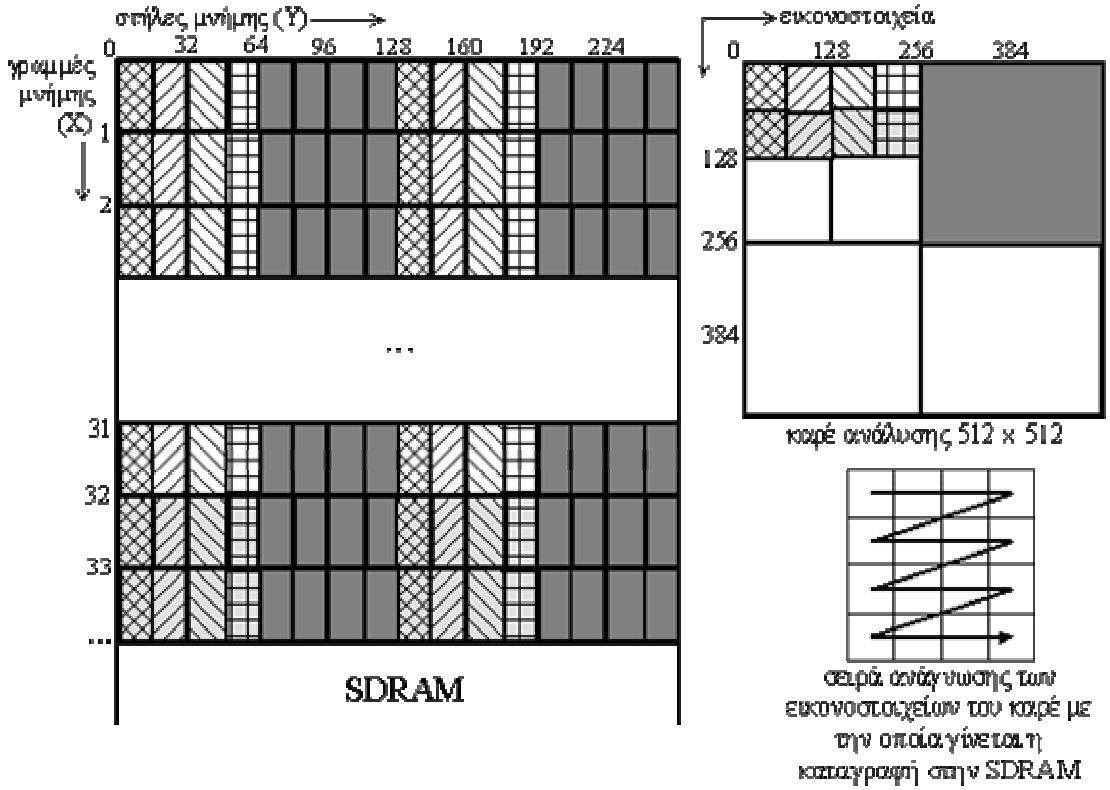
α/α	Τιμές των I, J από την FSM	Τελικές Διευθύνσεις
1	I = 0, J = 0	**X = 0, Y = 0** (X = I, Y = J)
2		**X = 0, Y = 16** (X = I, Y = J + 16)
3		**X = 0, Y = 32** (X = I, Y = J + 32)
4		**X = 0, Y = 48** (X = I, Y = J + 48)
5	I = 0, J = 8 (J = J + 8)	**X = 0, Y = 8** (X = I, Y = J)
6		**X = 0, Y = 24** (X = I, Y = J + 16)
7		**X = 0, Y = 40** (X = I, Y = J + 32)
8		**X = 0, Y = 56** (X = I, Y = J + 48)
9	I = 0, J = 64 (J = J + 56)	**X = 0, Y = 64** (X = I, Y = J)
10		**X = 0, Y = 80** (X = I, Y = J + 16)
11		**X = 0, Y = 96** (X = I, Y = J + 32)
12		**X = 0, Y = 112** (X = I, Y = J + 48)

Πίνακας 4.4 Οι 12 πρώτες διευθύνσεις και ο τρόπος παραγωγής τους για ανάλυση καρέ 256 × 256

3. Ανάλυση καρέ 512 × 512 εικονοστοιχεία

Στο Σχήμα 4.11 φαίνεται η κατανομή των 64 × 64 παραθύρων ενός καρέ 512 × 512 στη μνήμη SDRAM. Τα 64 εικονοστοιχεία της πρώτης γραμμής του παραθύρου που βρίσκεται πάνω αριστερά στο καρέ μοιράζονται στις πρώτες 16 θέσεις της μνήμης, ενώ στις επόμενες 48 θέσεις της βρίσκονται τα 64 εικονοστοιχεία των πρώτων γραμμών των επόμενων τριών παραθύρων, όπως φαίνεται και από τη γραμμοσκίαση. Τα τέσσερα παράθυρα που ακολουθούν στο καρέ καταλαμβάνουν τις θέσεις 64 – 127 της μνήμης, ενώ δεν είναι επιθυμητό να διαβαστούν σε πρώτη φάση, διότι σε τέτοια περίπτωση θα εισαγόταν στο σύστημα πληροφορία για οκτώ καρέ, γεγονός ανεπιθύμητο. Έτσι, μετά τη θέση 63 πρέπει να διαβαστεί η θέση 128 έως και την 191. Το κάθε καρέ καταλαμβάνει συνολικά 256 γραμμές της μνήμης.

Όπως και στις προηγούμενες περιπτώσεις, στον Πίνακα 4.5 παρατίθενται οι πρώτες 12 διευθύνσεις που παράγονται, καθώς και ο τρόπος με τον οποίο προέκυψαν.



Σχήμα 4.11 Κατανομή ενός καρέ ανάλυσης 512 × 512 στη μνήμη SDRAM

α/α	Τιμές των I, J από την FSM	Τελικές Διευθύνσεις
1	I = 0, J = 0	**X = 0, Y = 0** (X = I, Y = J)
2		**X = 0, Y = 16** (X = I, Y = J + 16)
3		**X = 0, Y = 32** (X = I, Y = J + 32)
4		**X = 0, Y = 48** (X = I, Y = J + 48)
5	I = 0, J = 8 (J = J + 8)	**X = 0, Y = 8** (X = I, Y = J)
6		**X = 0, Y = 24** (X = I, Y = J + 16)
7		**X = 0, Y = 40** (X = I, Y = J + 32)
8		**X = 0, Y = 56** (X = I, Y = J + 48)
9	I = 0, J = 128 (J = J + 120)	**X = 0, Y = 128** (X = I, Y = J)
10		**X = 0, Y = 144** (X = I, Y = J + 16)
11		**X = 0, Y = 160** (X = I, Y = J + 32)
12		**X = 0, Y = 176** (X = I, Y = J + 48)

Πίνακας 4.5 Οι 12 πρώτες διευθύνσεις και ο τρόπος παραγωγής τους για ανάλυση καρέ 512 × 512

Για να δοθούν στο σύστημα και τα δεδομένα των στηλών 64 – 127 και 192 – 255 της μνήμης, η FSM αφαιρεί από τον μετρητή J την τιμή 127 όταν αυτός φτάσει για πρώτη φορά τις τιμές 128, 256, 384, κ.ο.κ. Αυτό παρουσιάζεται στον Πίνακα 4.6, όπου υπάρχουν οι διευθύνσεις με αύξοντα αριθμό από 2045 έως 2060, οι οποίες αντιστοιχούν και στις τρεις πρώτες τετράδες του πάνω δεξιά 256 × 256 παραθύρου.

Τελειώνοντας, αναφέρεται ότι η FIFO [38] στην οποία αποθηκεύονται προσωρινά οι διευθύνσεις πριν τις διαβάσει ο ελεγκτής είναι σύγχρονη κι έχει μέγεθος 16 × 19 bits. Φτιάχτηκε, όπως και κάθε άλλη FIFO του συστήματος, με τον Core Generator της

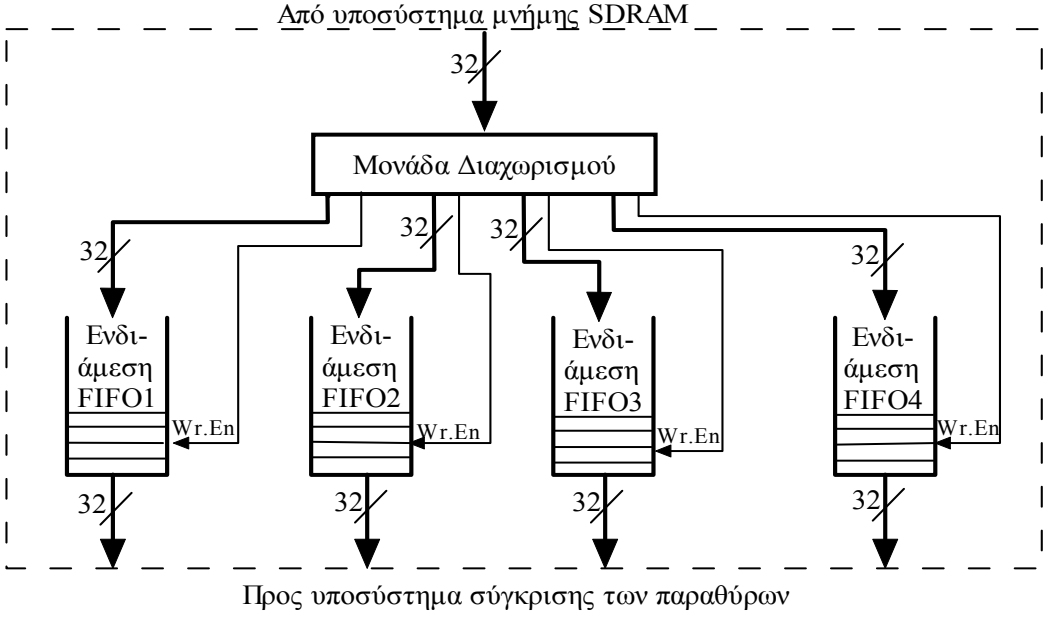
α/α	**Τιμές των I, J από την FSM**	**Τελικές Διευθύνσεις**
2045	I = 127, J = 136	**X = 127, Y = 136** (X = I, Y = J)
2046		**X = 127, Y = 152** (X = I, Y = J + 16)
2047		**X = 127, Y = 168** (X = I, Y = J + 32)
2048		**X = 127, Y = 184** (X = I, Y = J + 48)
2049	I = 0, J = 0 (I = I – 127, J = 0)	**X = 0, Y = 64** (X = I, Y = J + 64)
2050		**X = 0, Y = 80** (X = I, Y = J + 80)
2051		**X = 0, Y = 96** (X = I, Y = J + 96)
2052		**X = 0, Y = 112** (X = I, Y = J + 112)
2053	I = 0, J = 8 (J = J + 8)	**X = 0, Y = 72** (X = I, Y = J + 64)
2054		**X = 0, Y = 88** (X = I, Y = J + 80)
2055		**X = 0, Y = 104** (X = I, Y = J + 96)
2056		**X = 0, Y = 120** (X = I, Y = J + 112)
2057	I = 0, J = 128 (J = J + 120)	**X = 0, Y = 192** (X = I, Y = J + 64)
2058		**X = 0, Y = 208** (X = I, Y = J + 80)
2059		**X = 0, Y = 224** (X = I, Y = J + 96)
2060		**X = 0, Y = 240** (X = I, Y = J + 112)

Πίνακας 4.5 Οι διευθύνσεις με α/α από 2045 έως 2060. Οι διευθύνσεις 2045 – 2048 αντιστοιχούν στην τελευταία τετράδα διευθύνσεων του πάνω αριστερά 256 × 256 παραθύρου του καρέ, τη συνέχεια των οποίων αποτελούν οι διευθύνσεις με α/α 2049 – 2060. Επίσης, φαίνεται ο τρόπος παραγωγής τους για ανάλυση καρέ 512 × 512

εταιρίας Xilinx και προορίζεται για την οικογένεια FPGAs Virtex. Το βάθος της είναι μόνο 16 (το ελάχιστο δυνατό που επιτρέπεται από τον Core Generator), καθώς όσο μεγαλύτερη και αν γινόταν δε θα εξυπηρετούσε σε τίποτε, αφού οι διευθύνσεις που παράγονται από τον address generator είναι πάρα πολλές σε αριθμό και ο ρυθμός που αυτές χρησιμοποιούνται από τον ελεγκτή της μνήμης είναι σημαντικά μικρότερος από αυτόν με τον οποίο παράγονται, οπότε έτσι κι αλλιώς ο address generator θα έπρεπε να αναστέλλει τη λειτουργία του κάποιες φορές. Αξίζει τέλος να σημειωθεί ότι αυτή η αναστολή της λειτουργίας του δε γίνεται με χρήση gate clock ώστε να επηρεάζεται ο συγχρονισμός του ρολογιού, αλλά μέσω της FSM που αυτός περιέχει και με τη βοήθεια των σημάτων full κι empty της FIFO.

4.3 Το υποσύστημα διαχωρισμού των παραθύρων

Το υποσύστημα αυτό αποτελείται από μία μονάδα διαχωρισμού των παραθύρων και από τέσσερις FIFOs. Τα δεδομένα από το υποσύστημα της μνήμης SDRAM εισέρχονται στη μονάδα διαχωρισμού και ανάλογα με το παράθυρο του καρέ στο οποίο ανήκουν οδηγούνται στην αντίστοιχη FIFO. Τα χωρισμένα σε παράθυρα δεδομένα είναι έτοιμα να εισέλθουν στις επόμενες μονάδες του συστήματος, οι οποίες αποτελούν το κύριο τμήμα του αλγορίθμου SCAN. Το block διάγραμμα του υποσυστήματος διαχωρισμού των παραθύρων παρουσιάζεται στο Σχήμα 4.12.

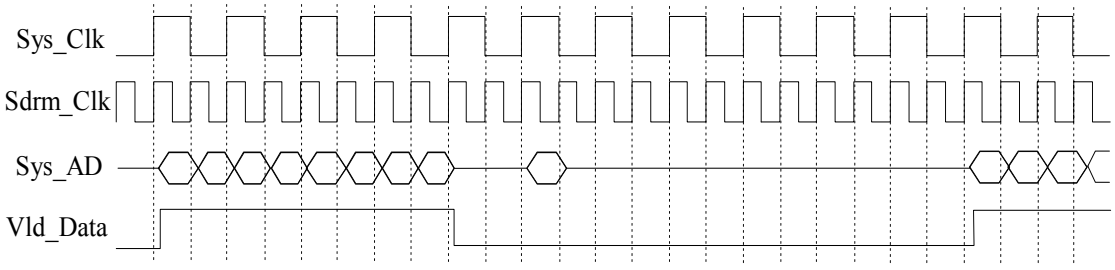


Σχήμα 4.12 Block διάγραμμα του υποσυστήματος διαχωρισμού των παραθύρων

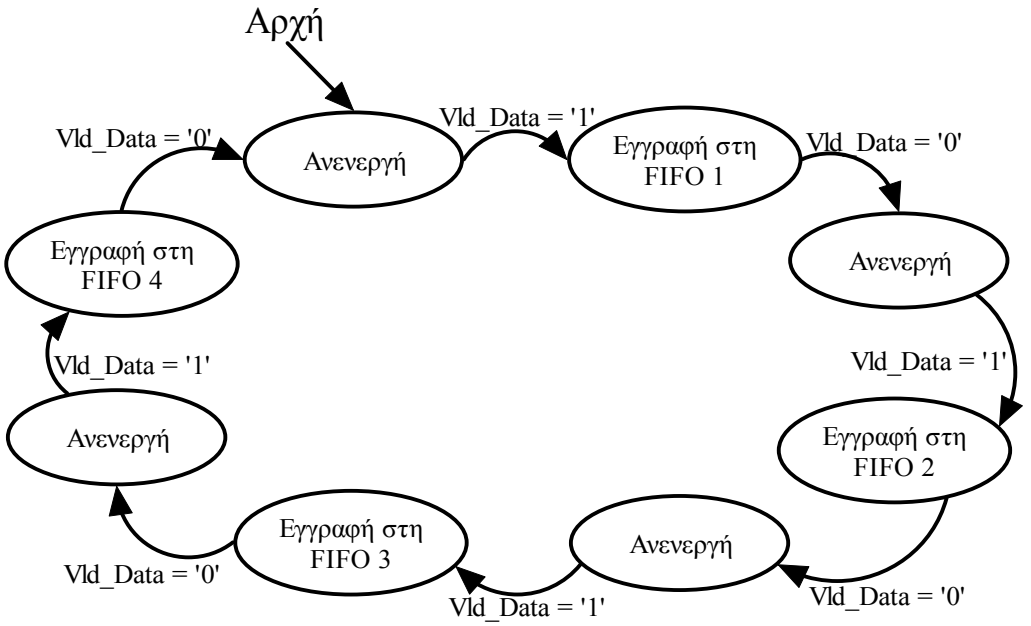
Όπως είναι φανερό, η βασική μονάδα του υποσυστήματος αυτού είναι η μονάδα διαχωρισμού. Αυτή είναι ουσιαστικά μία FSM, η οποία βγάζει τα δεδομένα στην κατάλληλη έξοδο και δίνει τιμή ένα το Write Enable της αντίστοιχης FIFO σύμφωνα με τη σειρά των δεδομένων που δέχεται στην είσοδό της. Ο τρόπος με τον οποίο εισάγονται τα δεδομένα στη μονάδα διαχωρισμού φαίνεται στην κυματομορφή του Σχήματος 4.13. Διακρίνονται καθαρά οι ριπές των οκτώ θέσεων μνήμης που διαβάζονται με μία πρόσβαση, ενώ μεταξύ δύο έγκυρων οκτάδων μεσολαβούν 14 κύκλοι ρολογιού της SDRAM, χρόνος που απαιτείται μέχρι η SDRAM να πάρει τη νέα διεύθυνση και να δώσει τα νέα δεδομένα.

Η μονάδα διαχωρισμού περιμένει μέχρι να πάρει την τιμή ένα για πρώτη φορά το σήμα Valid Data. Τότε, για οκτώ κύκλους ρολογιού της SDRAM (το ρολόι με το οποίο συγχρονίζεται είναι το διπλάσιας συχνότητας ρολόι που επιστρέφει η SDRAM στο σύστημα) το Write Enable της πρώτης FIFO γίνεται ένα, ενώ στην είσοδο της FIFO αυτής οδηγούνται τα δεδομένα που έχουν έρθει από τη μνήμη. Σύμφωνα και με όσα έχουν αναλυθεί ως τώρα, τα δεδομένα που εισάγονται στη FIFO αυτή αφορούν αποκλειστικά και μόνο το πρώτο κατά σειρά παράθυρο του καρέ. Στη συνέχεια, η μονάδα διαχωρισμού περιμένει και πάλι να πάρει την τιμή ένα το σήμα Valid Data. Όταν γίνει αυτό, ενεργοποιείται για οκτώ κύκλους ρολογιού το Write Enable της δεύτερης FIFO κι έτσι γράφονται σε αυτή δεδομένα που αφορούν το επόμενο παράθυρο του καρέ. Η ίδια διαδικασία ακολουθείται για την τρίτη κα τέταρτη FIFO, ενώ στη συνέχεια (δηλαδή την πέμπτη φορά που γίνεται ένα το σήμα Valid Data) τα δεδομένα γράφονται και πάλι στην πρώτη FIFO, εφόσον αφορούν το πρώτο παράθυρο του καρέ, μετά στη δεύτερη, κ.ο.κ. Όλη η διαδικασία παρουσιάζεται και στο Σχήμα 4.14.

Από τη στιγμή που θα γραφτούν τα 8 πρώτα δεδομένα στην κάθε FIFO, το υποσύστημα σύγκρισης των παραθύρων (το οποίο αναλύεται στην επόμενη παράγραφο)



Σχήμα 4.13 Ο τρόπος με τον οποίο εισάγονται τα δεδομένα στη μονάδα διαχωρισμού



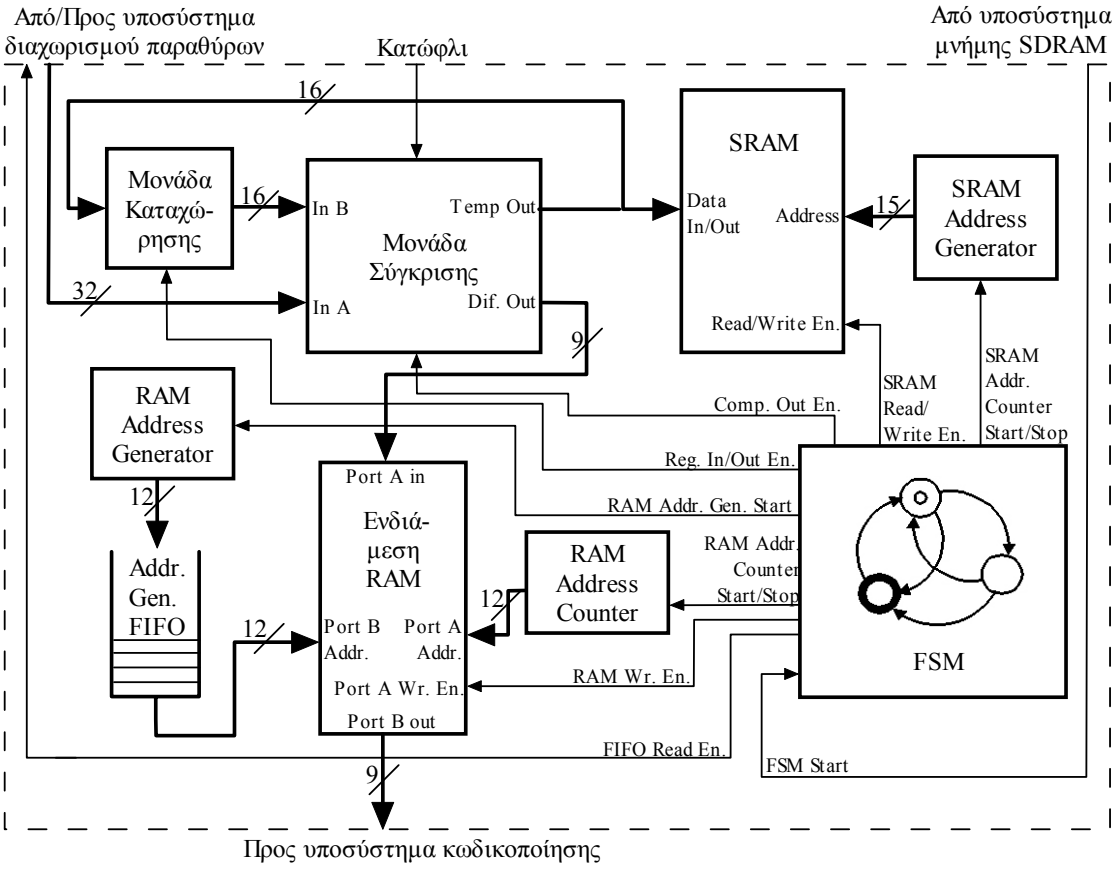
Σχήμα 4.14 Η λειτουργία της μονάδας διαχωρισμού

προλαβαίνει να τα διαβάσει όλα πριν γραφούν στην ίδια FIFO τα επόμενα 8. Αυτό σημαίνει ότι το μέγεθος της κάθε FIFO αρκεί να είναι 8. Αυτό δεν ήταν δυνατό λόγω του ότι το ελάχιστο μέγεθος της FIFO που μπορεί να δημιουργήσει ο Core Generator είναι 16, οπότε οι 4 FIFOs του υποσυστήματος διαχωρισμού των παραθύρων επιλέχτηκε να έχουν 16 θέσεις.

4.4 Το υποσύστημα σύγκρισης των παραθύρων

Το παρόν υποσύστημα, καθώς και το υποσύστημα κωδικοποίησης που υπάρχει μετά από αυτό, υπάρχουν σε συνολικά τέσσερα αντίγραφα στο σύστημα, όπως φαίνεται καθαρά και στο γενικό block διάγραμμα της αρχιτεκτονικής (Σχήμα 4.1). Η ανάλυση της λειτουργίας τους θα γίνει μόνο για το ένα αντίγραφο, εφόσον για τα άλλα ισχύουν ακριβώς τα ίδια.

Το υποσύστημα σύγκρισης παραθύρων αποτελείται από τη μονάδα σύγκρισης, μία μνήμη τύπου SRAM και χωρητικότητας 32K × 16 bits, μία μονάδα καταχώρησης, έναν address counter, μία μνήμη τύπου Dual Port RAM [39] μεγέθους 4K × 9 bits, έναν address generator για τη RAM κι έναν για την SRAM, μία FIFO 16 θέσεων των 12 bits και μία FSM που ελέγχει τη λειτουργία του υποσυστήματος. Ο τρόπος που συνδέονται όλες αυτές οι μονάδες φαίνεται στο block διάγραμμα του Σχήματος 4.15.



Σχήμα 4.15 Block διάγραμμα του υποσυστήματος σύγκρισης των παραθύρων

Τα τμήματα του αλγορίθμου SCAN που αφορούν τη δημιουργία των καρέ σύγκρισης και των καρέ διαφορών υλοποιούνται στο υποσύστημα αυτό. Βέβαια, εφόσον γίνεται επεξεργασία σε ένα παράθυρο μόνο του καρέ, ουσιαστικά δημιουργούνται παράθυρο σύγκρισης και παράθυρο διαφορών. Αφού το παράθυρο ή τα παράθυρα που ανήκουν στο πρώτο καρέ γραφούν στην SRAM, τα επόμενα συγκρίνονται με αυτά (στη μονάδα σύγκρισης) και το παράθυρο διαφορών που δημιουργείται οδηγείται στο Port A της μνήμης RAM, όπου γράφεται με τη βοήθεια του κατάλληλου address generator. Στη συνέχεια, οδηγείται προς το υποσύστημα κωδικοποίησης. Ταυτόχρονα με το παράθυρο διαφορών, δημιουργείται από τον συγκριτή και το παράθυρο σύγκρισης, το οποίο γράφεται στην SRAM και θα χρησιμοποιηθεί για τη σύγκρισή του με το επόμενο παράθυρο, όπως ακριβώς έχει περιγραφεί στην παράγραφο 3.2. Στη συνέχεια ακολουθεί αναλυτική περιγραφή της λειτουργίας και των μονάδων που απαρτίζουν το υποσύστημα αυτό.

4.4.1 Η μνήμη SRAM

Η μνήμη SRAM (Static Random Access Memory) που επιλέχτηκε να χρησιμοποιηθεί [40] έχει μέγεθος 32K × 16 bits και χωράει συνολικά 16 παράθυρα μεγέθους 64 × 64 το καθένα. Οι ανάγκες που έχει το σύστημα για προσωρινή αποθήκευση δεδομένων, πέρα από τις μνήμες RAM που έχουν χρησιμοποιηθεί, απαιτούσαν την ικανότητα αποθήκευσης ενός ολόκληρου καρέ σε γρήγορη μνήμη. Το καρέ αυτό είναι το καρέ σύγκρισης και βρίσκεται διαμοιρασμένο σε παράθυρα και στα τέσσερα τμήματα του συστήματος που επεξεργάζονται τα δεδομένα παράλληλα. Στην περίπτωση που τα καρέ είναι ανάλυσης 128 × 128 εικονοστοιχείων, χρειάζεται μνήμη συνολικού μεγέθους 16 KB, αν είναι 256 × 256 χρειάζονται 64 KB μνήμης, ενώ για καρέ ανάλυσης 512 × 512 οι απαιτήσεις σε μνήμη είναι 256 KB. Αν και μνήμη 16 KB πιθανά να μπορούσε να χωρέσει στην FPGA για την οποία έγινε η σχεδίαση (XC2V500), ακολουθήθηκε η ίδια τακτική για όλες τις αναλύσεις των καρέ που υποστηρίζονται και τα καρέ σύγκρισης αποθηκεύονται εξ' ολοκλήρου στην SRAM. Σημειώνεται ότι με χρήση μιας μεγάλης FPGA της οικογένειας Virtex2 (π.χ. XC2V6000), λόγω της αρκετής διαθέσιμης μνήμης αποφεύγεται η χρήση της SRAM, θα ήταν ασύμφορο όμως να χρησιμοποιηθεί μία πολύ ακριβότερη FPGA μόνο για την παραπάνω διαθέσιμη μνήμη, ενώ η χρησιμοποίηση της SRAM δίνει τη δυνατότητα με πολύ μικρές παρεμβάσεις στο σύστημα για αύξηση του μεγέθους της μνήμης, άρα και δυνατότητα επεξεργασίας καρέ μεγαλύτερης ανάλυσης. Για παράδειγμα, καρέ ανάλυσης 1024 × 1024 απαιτεί 1 MB μνήμης.

Τα 256 KB SRAM που χρειάζονται στο σύστημα αποτελούνται από τέσσερα chips των 64 KB, ένα στο κάθε τμήμα επεξεργασίας. Οι συγκεκριμένες μνήμες επιλέχτηκαν λόγω της ταχύτητάς τους. Ο χρόνος που απαιτείται τόσο για εγγραφή (T_{WC}), όσο και για ανάγνωση (T_{AA}) είναι 12 ns. Υπενθυμίζοντας ότι για τις FPGAs της οικογένειας Virtex2 ισχύει T_{AC} = 2,5 ns και T_{SU} = 1,4 ns (βλέπε παράγραφο 4.2.2) και θεωρώντας καθυστέρηση διάδοσης της τάξης του 1 ns, για την εγγραφή των δεδομένων στην SRAM ισχύει:

Virtex2 T_{AC} + SRAM T_{WC} + καθυστέρηση διάδοσης ή

2,5 + 12 + 1,0 ή

15,5 ns

Αντίστοιχα, για την ανάγνωση των δεδομένων από την SRAM ισχύει:

SRAM T_{AA} + Virtex2 T_{SU} + καθυστέρηση διάδοσης ή

12 + 1,4 + 1,0 ή

14,4 ns

Οι τιμές αυτές επιτρέπουν στο σύστημα να λειτουργήσει σε συχνότητα περίπου 65 MHz, τιμή πλήρως ικανοποιητική, διότι είναι μεγαλύτερη από αυτή που έχει τελικά το σύστημα, κάτι που φαίνεται αναλυτικά στο επόμενο κεφάλαιο, ενώ είναι μεγαλύτερη και από τη μέγιστη συχνότητα στην οποία μπορεί να λειτουργήσει ο ελεγκτής της μνήμης SDRAM (62,5 MHz) όταν αυτός τοποθετείται και δρομολογείται (placed and routed) αυτόματα σε μια FPGA.

Το μοντέλο προσομοίωσης που χρησιμοποιήθηκε [41] δεν είναι της κατασκευάστριας εταιρίας της μνήμης, αλλά είναι παραμετροποιήσιμο, οπότε ουσιαστικά συμπεριφέρεται όπως η πραγματική μνήμη SRAM του συστήματος.

4.4.2 Η μονάδα καταχώρησης

Η μονάδα αυτή αποτελείται από δύο καταχωρητές των 16 bits. Η λειτουργία της είναι να αποθηκεύει προσωρινά τα δεδομένα που έρχονται από την SRAM και προορίζονται για τη μονάδα σύγκρισης. Ελέγχεται από τέσσερα σήματα, δύο από τα οποία καθορίζουν αν η τιμή που βρίσκεται στην είσοδό της θα αποθηκευτεί σε κάποιον από τους δύο διαθέσιμους καταχωρητές, καθώς και σε ποιόν από αυτούς. Αν κάποιο από τα άλλα δύο σήματα πάρει την τιμή ένα, τότε η τιμή του αντίστοιχου καταχωρητή οδηγείται στην έξοδο της μονάδας.

Η μονάδα αυτή έχει μεγάλη χρησιμότητα, διότι τροφοδοτεί τη μονάδα σύγκρισης με δεδομένα τη στιγμή που τα χρειάζεται. Λόγω των συνεχών αναγνώσεων κι εγγραφών στην SRAM, δεν υπάρχει χρόνος να δίνει αυτή στο συγκριτή τα δεδομένα που χρειάζεται τη στιγμή που τα χρειάζεται. Επιπλέον, εφόσον ο δίαυλος των δεδομένων της μνήμης SRAM είναι αμφίδρομος, τη στιγμή που θα γινόταν εγγραφή στη μνήμη θα εισέρχονταν στη μονάδα σύγκρισης λάθος δεδομένα, κάτι που θα οδηγούσε σε μη λειτουργικό σύστημα. Αυτό φαίνεται καθαρά και στο Σχήμα 4.15, όπου αν η έξοδος

Temp_out της μονάδας σύγκρισης ενεργοποιηθεί, τα δεδομένα αυτής εκτός από την SRAM, για όπου και προορίζονται, θα πήγαιναν και στην είσοδο In_B της μονάδας σύγκρισης αν δεν υπήρχε η μονάδα καταχώρησης για να τα αποκόψει.

Ο λόγος που η μονάδα καταχώρησης αποτελείται από δύο κι όχι έναν καταχωρητή είναι ότι κάποια χρονική στιγμή είναι αναγκαίο να βρίσκονται προσωρινά αποθηκευμένες δύο ποσότητες των 16 bits. Η περίπτωση αυτή παρουσιάζεται στην παράγραφο 4.4.4.

4.4.3 Η μονάδα σύγκρισης

Η μονάδα αυτή είναι πολύ σημαντική διότι ουσιαστικά εκτελεί όλες τις συγκρίσεις που απαιτεί ο αλγόριθμος για τη δημιουργία των καρέ διαφορών και σύγκρισης και δημιουργεί τα αντίστοιχα παράθυρα. Όπως αναφέρθηκε και πριν, δημιουργούνται παράθυρα κι όχι καρέ σύγκρισης και διαφορών διότι στις εισόδους της μονάδας σύγκρισης εισάγονται δεδομένα για παράθυρα κι όχι για ολόκληρα καρέ.

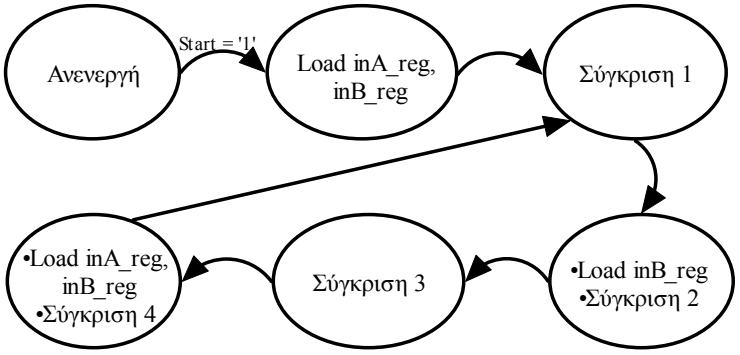
Η παρούσα μονάδα εκτελεί τους υπολογισμούς με τη βοήθεια δύο συγκριτών 8-bit κι ενός προσθετή 8-bit. Οι δύο προς σύγκριση ποσότητες οδηγούνται στον πρώτο συγκριτή και στη συνέχεια η μικρότερη από αυτές αφαιρείται από τη μεγαλύτερη. Το αποτέλεσμα του προσθετή πηγαίνει στο δεύτερο συγκριτή, όπου και συγκρίνεται με την τιμή του κατωφλίου. Ο λόγος που οι συγκριτές και ο προσθετής είναι 8-bit είναι προφανής, αφού συγκρίνονται εικονοστοιχεία και με 8 bits αναπαριστάται η τιμή φωτεινότητας του καθενός, οπότε και η τιμή του κατωφλίου αναγκαστικά αναπαριστάται με 8 bits. Σημειώνεται ότι εφόσον πάντα αφαιρείται μία ποσότητα από μία μεγαλύτερή της, δε συμβαίνει υπερχείλιση κατά την αφαίρεση.

Οι δύο είσοδοι της μονάδας σύγκρισης, όπως φαίνεται και στο Σχήμα 4.15 έχουν πλάτος 32 και 16 bits, δηλαδή κάθε φορά που αλλάζουν τιμή εισάγουν στη μονάδα πληροφορία για τέσσερα και δύο εικονοστοιχεία αντίστοιχα. Τα δεδομένα αυτά πρέπει να διαχειριστούν με τρόπο τέτοιο ώστε να συγκρίνονται ένα προς ένα. Αυτό επιτυγχάνεται αποθηκεύοντάς τα κάθε φορά που αλλάζουν τιμή σε δύο καταχωρητές αντίστοιχου μεγέθους, τους inA_reg και inB_reg. Στη συνέχεια, τα δεδομένα των καταχωρητών αυτών συγκρίνονται ανά οκτώ, δηλαδή συγκρίνονται διαδοχικά τα:

inA_reg(31 downto 24) και inB_reg(15 downto 8), inA_reg(23 downto 16) και inB_reg(7 downto 0), inA_reg(15 downto 8) και inB_reg(15 downto 8) και τέλος τα inA_reg(7 downto 0) και inB_reg(7 downto 0). Προφανώς, μετά τις δύο πρώτες συγκρίσεις ο καταχωρητής inB_reg έχει πάρει νέα τιμή. Η σειρά με την οποία εκτελούνται οι συγκρίσεις στη μονάδα, καθώς και το κάθε πότε φορτώνεται νέα τιμή στους καταχωρητές inA_reg και inB_reg φαίνεται και στο Σχήμα 4.16.

Σε κάθε σύγκριση που εκτελείται, αν η διαφορά των συγκρινόμενων εικονοστοιχείων είναι εντός κατωφλίου, στην έξοδο Dif_Out της μονάδας οδηγείται η τιμή '100000000', ενώ αν είναι εκτός κατωφλίου στην ίδια έξοδο οδηγείται ένα bit με τιμή 0, ακολουθούμενο από τα αντίστοιχα 8 bits του καταχωρητή inA_reg. Τα δεδομένα που εξάγονται από την έξοδο Dif_Out απαρτίζουν τα παράθυρα των διαφορών. Η τιμή '100000000' στα παράθυρα των διαφορών αντιστοιχεί στην τιμή -1 που δίνεται από την υλοποίηση του αλγορίθμου SCAN σε software, σύμφωνα με την περιγραφή που έχει δοθεί για αυτόν στην παράγραφο 3.2. Το επιπλέον bit που προστίθεται στα υπόλοιπα εικονοστοιχεία των παραθύρων διαφορών δεν αλλοιώνει την πληροφορία, ενώ η χρησιμότητά του φαίνεται κατά την ανάλυση της μονάδας κωδικοποίησης.

Ταυτόχρονα με τα δεδομένα των παραθύρων διαφορών, δημιουργούνται και τα δεδομένα για τα παράθυρα σύγκρισης. Όμως, ενώ τα δεδομένα των πρώτων οδηγούνται απευθείας στην αντίστοιχη έξοδο της μονάδας, δε συμβαίνει το ίδιο με αυτά των δεύτερων. Η έξοδος Temp_Out, από την οποία εξέρχονται τα δεδομένα για τα παράθυρα σύγκρισης έχει πλάτος 16 bits, ενώ τα δεδομένα που προκύπτουν από τη σύγκριση των εικονοστοιχείων αποτελούνται από 8 bits. Για το λόγο αυτό γίνεται χρήση δύο ακόμα 8-bit καταχωρητών, των tempf_1 και tempf_2, στους οποίους αποθηκεύονται εναλλάξ τα δεδομένα που αφορούν τα παράθυρα σύγκρισης. Στη



# Σύγκρισης	Συγκρινόμενες Ποσότητες
1	inA_reg(31 downto 24) inB_reg(15 downto 8)
2	inA_reg(23 downto 16) inB_reg(7 downto 0)
3	inA_reg(15 downto 8) inB_reg(15 downto 8)
4	inA_reg(7 downto 0) inB_reg(7 downto 0)

Σχήμα 4.16 Διάγραμμα καταστάσεων της μονάδας σύγκρισης και οι συγκρίσεις οι οποίες εκτελούνται στην κάθε κατάσταση

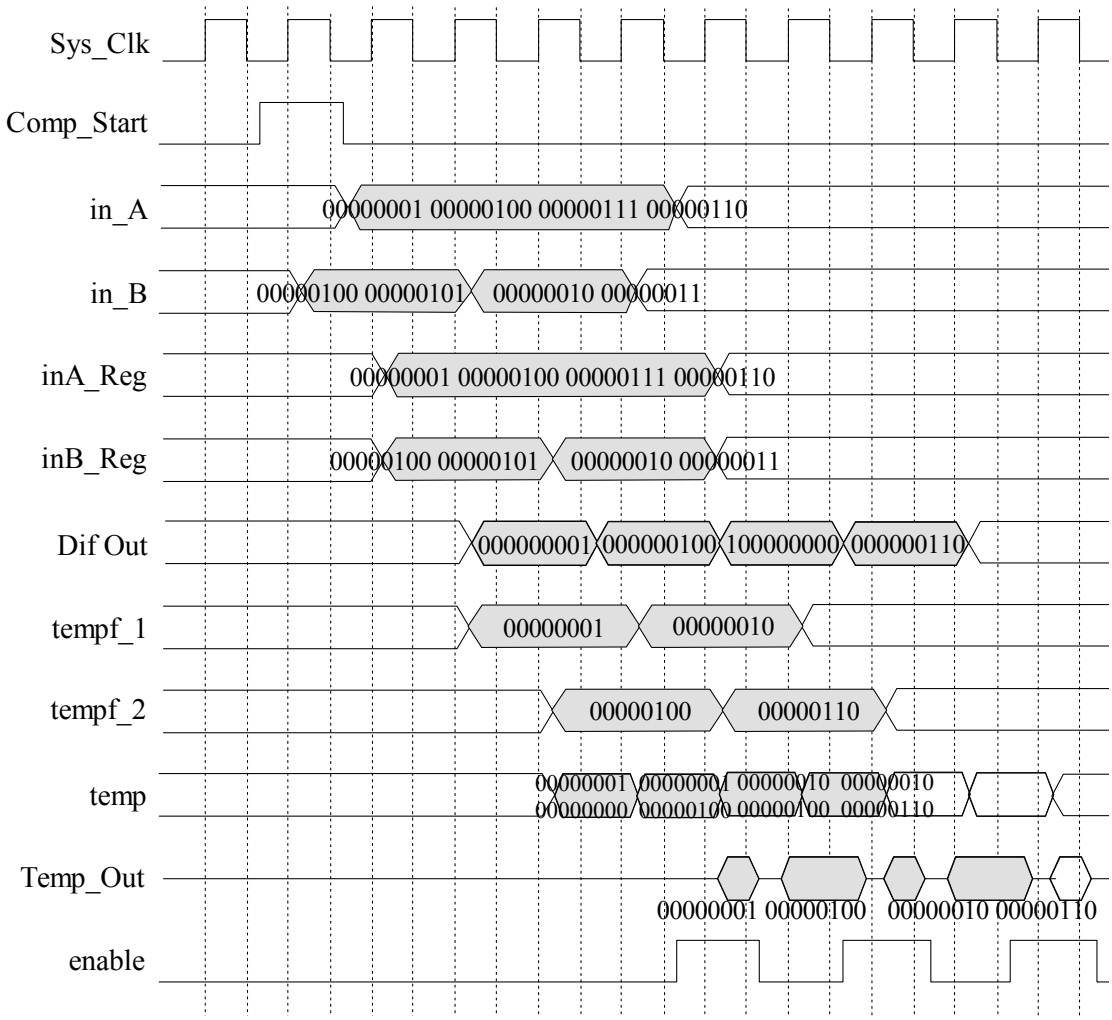
συνέχεια, τα δεδομένα αυτών των καταχωρητών ανά δύο (δηλαδή συνολικά 16 bits) οδηγούνται στον 16-bit καταχωρητή Temp. Όσον αφορά στον τρόπο δημιουργίας των δεδομένων των παραθύρων σύγκρισης, δε διαφέρει πολύ από αυτόν της software υλοποίησης του αλγορίθμου SCAN. Αν η διαφορά των συγκρινόμενων εικονοστοιχείων είναι εντός κατωφλίου, στον αντίστοιχο καταχωρητή οδηγείται η κατάλληλη οκτάδα δεδομένων από τον καταχωρητή inB_reg, ενώ σε αντίθετη περίπτωση σε αυτόν οδηγείται η κατάλληλη οκτάδα δεδομένων από τον καταχωρητή inA_reg.

Στο σημείο αυτό αναφέρεται ότι ένα σήμα (enable) ελέγχει την έξοδο Temp_Out της μονάδας. Αν αυτό έχει τιμή ίση με το ένα, επιτρέπει τη μεταφορά των περιεχομένων του καταχωρητή Temp στην έξοδο. Σε αντίθετη περίπτωση θέτει την έξοδο σε κατάσταση υψηλής εμπέδησης (Hi Z). Αυτό είναι απαραίτητο να γίνεται όταν ο αμφίδρομος δίαυλος της μνήμης SRAM χρησιμοποιείται για ανάγνωση από αυτή, αφού, όπως φαίνεται και στο Σχήμα 4.15, θα υπήρχε δυσλειτουργία του συστήματος αν ταυτόχρονα η SRAM και η μονάδα σύγκρισης εισήγαγαν δεδομένα στον ίδιο δίαυλο. Για να γίνει πιο κατανοητή η ακριβής λειτουργία της μονάδας σύγκρισης, παρατίθεται το διάγραμμα χρονισμού του Σχήματος 4.17, όπου φαίνεται καθαρά πότε και ποια δεδομένα γράφονται στον κάθε καταχωρητή και στέλνονται στην κάθε έξοδο. Στο σχήμα αυτό η τιμή του κατωφλίου είναι ίση με 4.

4.4.4 Η FSM του υποσυστήματος σύγκρισης των παραθύρων

Η FSM του υποσυστήματος σύγκρισης των παραθύρων ελέγχει τη λειτουργία ολόκληρου του υποσυστήματος. Ρυθμίζει την ανάγνωση των δεδομένων από τη FIFO του υποσυστήματος διαχωρισμού των παραθύρων, ελέγχει τους address counters της SRAM και του port A της ενδιάμεσης RAM, δίνει την κατάλληλη στιγμή εντολή να αρχίσει η λειτουργία του address generator του port B της ενδιάμεσης RAM, ελέγχει τα τέσσερα σήματα της μονάδας καταχώρησης και το σήμα enable της μονάδας σύγκρισης και καθορίζει το πότε θα γίνεται εγγραφή και πότε ανάγνωση στην SRAM.

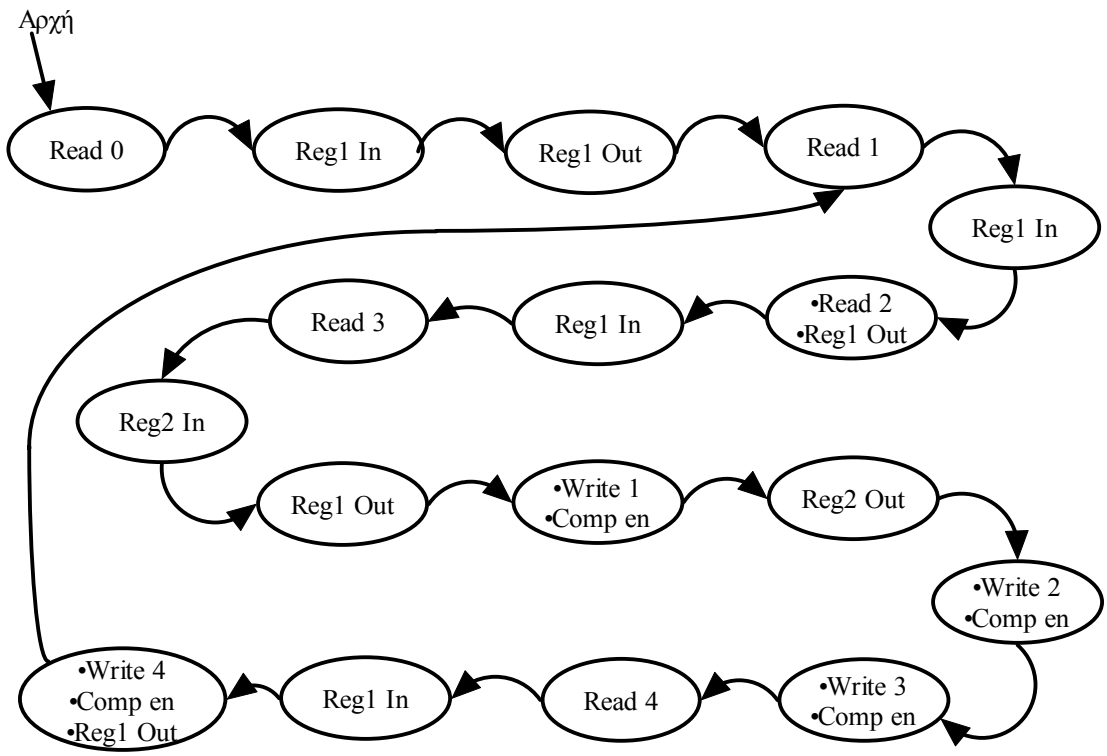
Η ρύθμιση των αναγνώσεων και των εγγραφών στην SRAM απαιτεί απόλυτη ακρίβεια, ενώ συνδυάζεται με τον έλεγχο της μονάδας καταχώρησης, αλλά και του σήματος enable της μονάδας σύγκρισης. Πιο συγκεκριμένα, μετά από κάθε ανάγνωση



Σχήμα 4.17 Διάγραμμα χρονισμού της μονάδας σύγκρισης

ενεργοποιείται η είσοδος ενός από τους δύο καταχωρητές της μονάδας καταχώρησης, ώστε να αποθηκευτούν προσωρινά τα δεδομένα που διαβάστηκαν. Επίσης, όταν χρειάζεται, ενεργοποιείται κι η αντίστοιχη έξοδος, ώστε να διοχετευθούν τα δεδομένα στη μονάδα σύγκρισης, ενώ κατά τη διάρκεια της κάθε εγγραφής ενεργοποιείται το enable της μονάδας σύγκρισης, ώστε να είναι διαθέσιμα στην έξοδο Temp_Out τα προς εγγραφή στην SRAM δεδομένα. Η ακριβής σειρά με την οποία γίνονται αυτές οι ενέργειες παρουσιάζεται στο Σχήμα 4.18. Στο σχήμα αυτό φαίνεται ότι η μονάδα καταχώρησης είναι αναγκαίο να έχει δύο καταχωρητές, διότι αν είχε μόνο έναν δε θα υπήρχε διαθέσιμος χώρος να γραφούν τα δεδομένα από την πρόσβαση για ανάγνωση στη μνήμη που περιγράφεται από την κατάσταση Read 3.

Ο χρόνος που μεσολαβεί από τη στιγμή που στη FIFO του υποσυστήματος διαχωρισμού των παραθύρων είναι διαθέσιμη η πρώτη από τις τιμές που διαβάστηκαν από



Σχήμα 4.18 Οι βασικότερες από τις καταστάσεις της FSM του υποσυστήματος σύγκρισης των παραθύρων

τη μνήμη SDRAM μέχρι και τη στιγμή που εισέρχεται στη FIFO αυτή η ένατη κατά σειρά τιμή είναι αρκετός για την επεξεργασία των οκτώ αυτών τιμών. Ο χρόνος αυτός είναι ουσιαστικά αυτός που μεσολαβεί μεταξύ δύο προσβάσεων για ανάγνωση στην SDRAM. Εφόσον, όπως έχει αναφερθεί ξανά, με κάθε πρόσβαση στην SDRAM διαβάζονται οι τιμές 32 εικονοστοιχείων, στην κάθε FIFO του υποσυστήματος διαχωρισμού των παραθύρων τοποθετούνται οκτώ από αυτές.

4.4.5 Ο address counter της ενδιάμεσης RAM

Ο address counter της ενδιάμεσης RAM είναι ένας 12-bit μετρητής, διότι η μνήμη που καλείται να διευθυνσιοδοτήσει είναι 4K θέσεων. Τα δεδομένα στη RAM αυτή είναι επιθυμητό να γράφονται σειριακά, παρ' όλο που δε διαβάζονται σειριακά. Η ανάγνωση των δεδομένων αρχίζει πριν τελειώσει η εγγραφή και των 4K θέσεων, ώστε να επιτυγχάνεται αποδοτική λειτουργία του συστήματος. Για το λόγο αυτό η ενδιάμεση RAM επιλέχτηκε να είναι dual port, με το ένα port να είναι μόνο για εγγραφή και το άλλο μόνο για ανάγνωση. Ο εν λόγω address counter παράγει

διευθύνσεις για την εγγραφή στη μνήμη, οπότε η έξοδός του οδηγείται στην είσοδο διευθύνσεων του port A.

Η FSM φροντίζει να αυξάνει την τιμή του address counter κάθε φορά που στην έξοδο Dif_Out της μονάδας σύγκρισης εμφανίζονται έγκυρα δεδομένα. Προφανώς, ο μετρητής αυτός αυξάνεται συνολικά κατά οκτώ στη διάρκεια δύο διαδοχικών προσβάσεων στη μνήμη SDRAM.

4.4.6 Ο address generator της μνήμης SRAM

Όσον αφορά στον address generator της SRAM, προτού αναλυθεί η λειτουργία του κρίνεται σκόπιμο να ξεκαθαριστεί σε ποιες θέσεις της SRAM γράφονται τα δεδομένα που προέρχονται από την έξοδο Temp_Out της μονάδας σύγκρισης. Για λόγους οικονομίας χώρου, προτιμήθηκε αντί η SRAM να έχει χωρητικότητα διπλάσια από τα παράθυρα που πρόκειται να αποθηκευτούν σε αυτή, να χωράει ακριβώς το μέγιστο αριθμό αυτών των παραθύρων. Αυτό ήταν εφικτό, διότι από τη στιγμή που διαβάζεται μία θέση μνήμης και τα περιεχόμενά της διοχετεύονται στο υπόλοιπο σύστημα, η θέση αυτή δεν περιέχει χρήσιμη πληροφορία, οπότε εκεί μπορούν να γραφούν νέα δεδομένα.

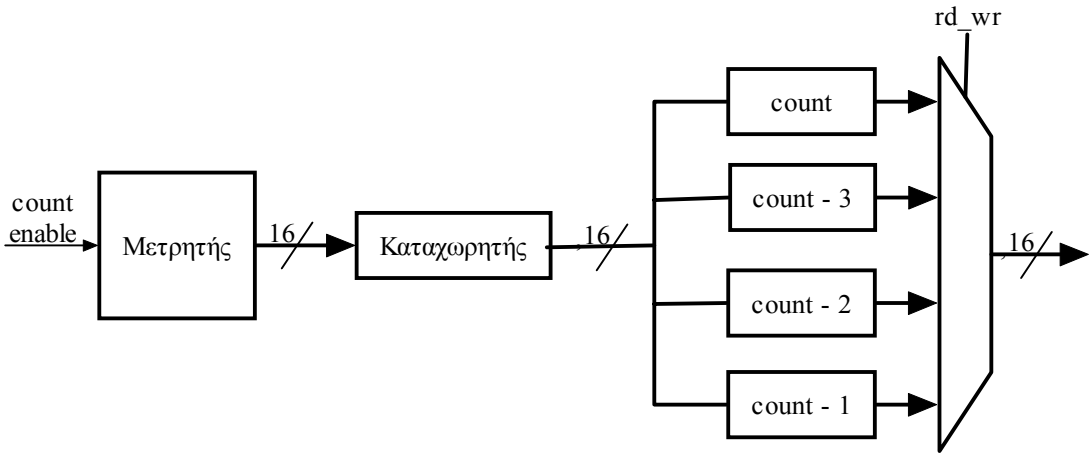
Κατά τη διάρκεια των τεσσάρων πρώτων αναγνώσεων από την SRAM, ο address generator της SRAM λειτουργεί ως address counter διότι τα δεδομένα διαβάζονται σειριακά, ενώ υπενθυμίζεται ότι λόγω του πλάτους λέξης της SRAM (16 bits), σε κάθε πρόσβαση σε αυτή διαβάζονται ή γράφονται δύο τιμές των 8 bits. Οι τέσσερις εγγραφές που ακολουθούν πρέπει να γίνουν στις θέσεις από τις οποίες έγινε η προηγούμενη τετράδα αναγνώσεων, ενώ οι ακόλουθες τέσσερις αναγνώσεις θα αρχίσουν από την επόμενη της θέσης που έγινε η τελευταία εγγραφή, διαδικασία που επαναλαμβάνεται συνεχώς.

Η FSM δίνει τιμές σε ένα 3-bit βοηθητικό σήμα, το rd_wr, σύμφωνα με το οποίο παράγεται η κατάλληλη τιμή στην έξοδο του address generator. Το σήμα αυτό δίνει στον address generator πληροφορία για το αν η διεύθυνση που πρόκειται να δημιουργηθεί αφορά ανάγνωση ή εγγραφή και ποια εγγραφή ($1^{η}$, $2^{η}$, $3^{η}$ ή $4^{η}$) στην SRAM. Στον Πίνακα 4.6 παρατίθενται οι τιμές που παίρνει το σήμα rd_wr, ανάλογα

Τιμή σήματος rd_wr	Αντιπροσωπευόμενη κατάσταση
000	Read 0, Read 1, Read 2, Read 3, Read 4
001	Write 1
010	Write 2
011	Write 3
100	Write 4

Πίνακας 4.6 Οι τιμές του σήματος rd_wr και οι καταστάσεις της FSM που η καθεμία αντιπροσωπεύει

με την κατάσταση της FSM την οποία αντιπροσωπεύει. Οι καταστάσεις αυτές έχουν παρουσιαστεί στο Σχήμα 4.18. Ένα block διάγραμμα του address generator αυτού παρουσιάζεται στο Σχήμα 4.19. Ο 16-bit μετρητής αυξάνει κατά ένα κάθε φορά που το σήμα count enable γίνεται ένα. Το σήμα αυτό ελέγχεται από την FSM και γίνεται ένα συνολικά τέσσερις φορές κατά τη διάρκεια δύο διαδοχικών προσβάσεων στη μνήμη SDRAM. Η τρέχουσα τιμή του μετρητή (count) αποθηκεύεται σε έναν καταχωρητή. Στη συνέχεια, όταν το σήμα rd_wr έχει τιμή '000' ή '100' στην έξοδο του address generator οδηγείται η τιμή count, δηλαδή η τρέχουσα τιμή του μετρητή, ενώ όταν είναι ίση με '001', '010' ή '011' στην έξοδο οδηγούνται οι τιμές count -3, count – 2 ή count – 1 αντίστοιχα.



Σχήμα 4.19 Block διάγραμμα του address generator της μνήμης SRAM

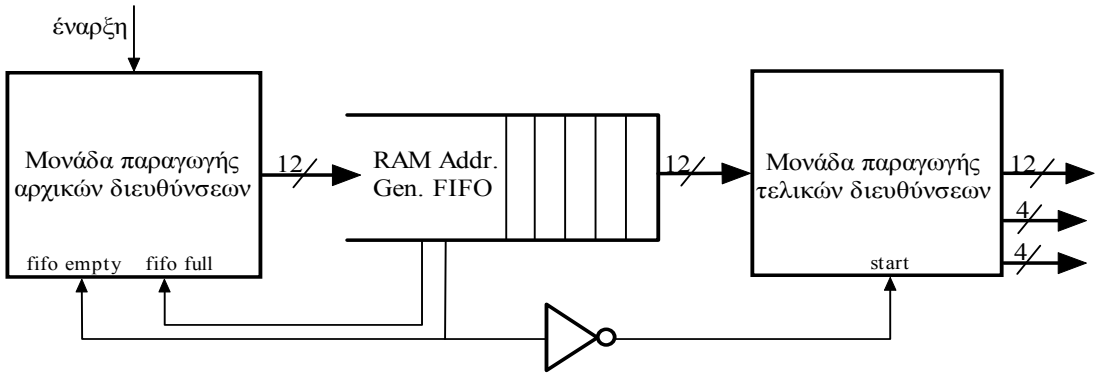
Ο μετρητής είναι 16-bit για να μπορεί να παράγει 64K διευθύνσεις, όσο και τομέγεθος της SRAM. Όμως, ο μέγιστος αριθμός των διευθύνσεων που μπορούν να παραχθούν καθορίζεται από την ανάλυση των καρέ του video. Αν αυτά είναι 128 × 128, στην κάθε SRAM τοποθετείται από ένα παράθυρο 64 × 64 και οι διευθύνσεις που χρειάζεται να παραχθούν είναι συνολικά 4K. Στην περίπτωση αυτή, ο μετρητής κάθε φορά που φτάνει την τιμή 4096 αρχίζει να μετρά από την αρχή, δηλαδή συμπεριφέρεται σαν μετρητής 12-bit. Για video ανάλυσης 256 × 256 εικονοστοιχείων τοποθετούνται από τέσσερα παράθυρα 64 × 64 σε κάθε SRAM, οπότε καταλαμβάνονται 16384 θέσεις της και ο μετρητής συμπεριφέρεται σαν να ήταν 14-bit. Τέλος, ο μετρητής μετράει έως το 65536 για video ανάλυσης 512 × 512, αφού στην περίπτωση αυτή αποθηκεύονται δεδομένα από δεκαέξι παράθυρα στην κάθε SRAM, δηλαδή αυτές γεμίζουν πλήρως.

4.4.7 Ο address generator της ενδιάμεσης μνήμης RAM

Το παράθυρο των διαφορών που αποθηκεύεται στην ενδιάμεση RAM πρέπει να αναλυθεί σε παράθυρα 4 × 4, ώστε αυτά να οδηγηθούν στη μονάδα κωδικοποίησης για περαιτέρω επεξεργασία. Ο address generator της ενδιάμεσης μνήμης RAM φροντίζει ώστε τα περιεχόμενά της να διαβαστούν με τρόπο τέτοιο ώστε να εξαχθούν από αυτή χωρισμένα σε παράθυρα 4 × 4. Όπως είναι φυσικό, η έξοδος του address generator οδηγείται στην είσοδο διευθύνσεων του port B της RAM, εφόσον αυτό το port προορίζεται για ανάγνωση.

Τα τμήματα από τα οποία αποτελείται είναι μία μονάδα παραγωγής αρχικών διευθύνσεων και μία μονάδα παραγωγής τελικών διευθύνσεων, ενώ ανάμεσά τους υπάρχει μία FIFO 16 θέσεων, πλάτους 12 bits. Η λειτουργία της καθεμιάς από τις δύο μονάδες μοιάζει αρκετά με αυτή του address generator της SDRAM, όπως αυτή περιγράφηκε στην παράγραφο 4.2.4. Το Σχήμα 4.20 παρουσιάζει ένα block διάγραμμα του address generator.

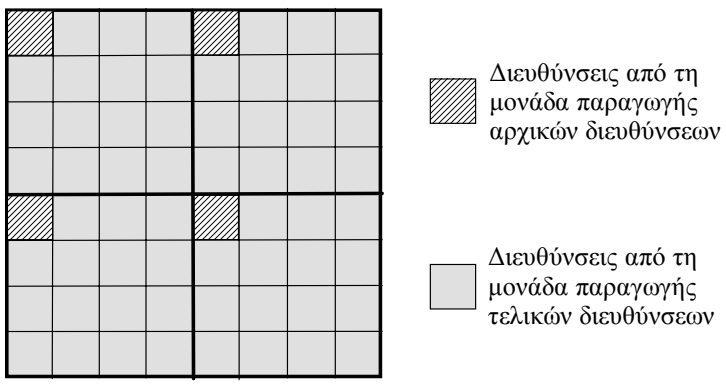
Η μονάδα παραγωγής αρχικών διευθύνσεων παράγει διαδοχικά τις διευθύνσεις των πρώτων πάνω αριστερά εικονοστοιχείων κάθε παραθύρου 4 × 4, οι οποίες προωθούνται στη μονάδα παραγωγής τελικών διευθύνσεων, η οποία παράγει και τις 16 διευθύνσεις του κάθε παραθύρου. Στο Σχήμα 4.21 αυτό φαίνεται πιο καθαρά



Σχήμα 4.20 Block διάγραμμα του address generator της ενδιάμεσης μνήμης RAM

για ένα παράθυρο 8 × 8 εικονοστοιχείων, δηλαδή ποιες διευθύνσεις παράγει η κάθε μονάδα. Μεταξύ των δύο μονάδων υπάρχει μία FIFO, η οποία δέχεται τις διευθύνσεις από τη μονάδα παραγωγής αρχικών διευθύνσεων. Από αυτή τις διαβάζει η μονάδα παραγωγής τελικών διευθύνσεων. Η μονάδα που διαβάζει τα δεδομένα από την FIFO χρειάζεται πάνω από 16 κύκλους ρολογιού για την επεξεργασία καθενός από αυτά (διότι πρέπει για το καθένα να παράγει 16 διευθύνσεις), ρυθμός μικρότερος από αυτό με τον οποίο εισάγονται στη FIFO τα δεδομένα. Για το λόγο αυτό, με τη βοήθεια των σημάτων full κι empty της FIFO η μονάδα παραγωγής αρχικών διευθύνσεων αναστέλλει και ξαναρχίζει, όταν χρειάζεται, τη λειτουργία της. Το μέγεθος της FIFO δεν κρίθηκε αναγκαίο να είναι μεγαλύτερο από 16 θέσεις, αφού κάτι τέτοιο δε θα προσέφερε τίποτα στην απόδοση του συστήματος

Το σύστημα αρχίζει να λειτουργεί όταν γίνει ένα το σήμα της έναρξης της μονάδας παραγωγής αρχικών διευθύνσεων, το οποίο ελέγχεται από την FSM. Όσον αφορά στη μονάδα παραγωγής τελικών διευθύνσεων, αυτή παράγει διευθύνσεις όσο η FIFO έχει δεδομένα. Για το λόγο αυτό οδηγήθηκε στην είσοδο της ‘start’ το αντίστροφο του



Σχήμα 4.21 Οι διευθύνσεις που παράγονται από την κάθε μονάδα του address generator για ένα παράθυρο ανάλυσης 8 × 8

σήματος που δείχνει αν η FIFO είναι άδεια. Με τον τρόπο αυτό σταματά η παραγωγή διευθύνσεων μόλις αδειάσει η FIFO.

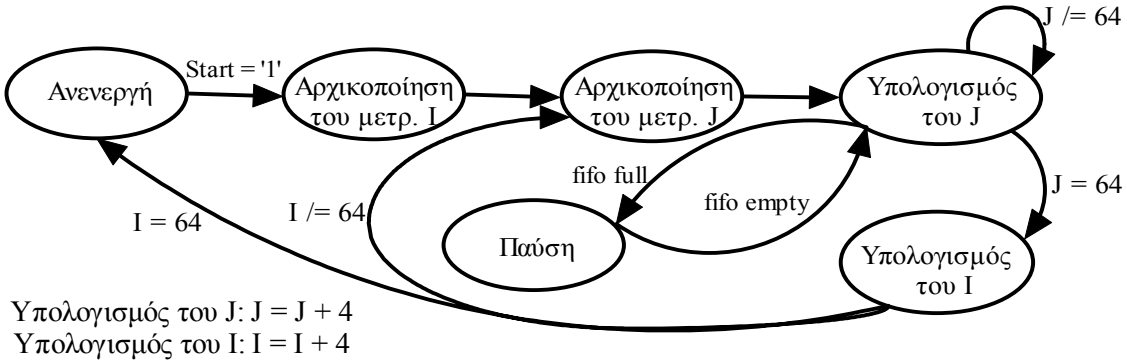
4.4.7.1 Η μονάδα παραγωγής αρχικών διευθύνσεων

Η μονάδα αυτή λειτουργεί ως FSM που ελέγχει δύο μετρητές, έναν που αντιστοιχεί στις γραμμές (I) κι έναν που αντιστοιχεί στις στήλες (J) του κάθε 64 × 64 παραθύρου που είναι αποθηκευμένο στην ενδιάμεση RAM, ενώ η λειτουργία της είναι εντελώς ανάλογη με αυτή της FSM του address generator της SDRAM.

Αφού αρχικοποιηθούν οι δύο μετρητές αρχίζει ο υπολογισμός των τιμών του J, όπως φαίνεται και στο Σχήμα 4.22. Στην κατάσταση υπολογισμού του J, αυτό αυξάνεται σε κάθε κύκλο ρολογιού κατά 4. Όταν το J γίνει 64, σημαίνει ότι έχουν υπολογιστεί διευθύνσεις που αφορούν μια γραμμή από τις συνολικά 64. Τότε αυξάνεται ο μετρητής I κατά 4 και η διαδικασία επαναλαμβάνεται μέχρι να διατρεχτούν και οι 64 γραμμές. Ο μετρητής I αυξάνει κατά τέσσερα κι όχι κατά ένα κάθε φορά, εφόσον οι αρχικές διευθύνσεις που πρέπει να παραχθούν βρίσκονται ανά τέσσερις γραμμές του παραθύρου, φαίνεται και στο Σχήμα 4.21.

4.4.7.2 Η μονάδα παραγωγής τελικών διευθύνσεων

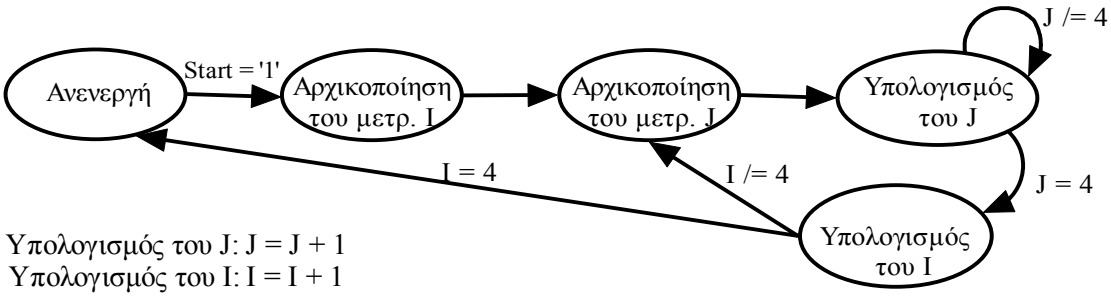
Η μονάδα αυτή αποτελείται από μία FSM και από ένα τμήμα υπολογισμού των διευθύνσεων. Η FSM λειτουργεί όπως ακριβώς και η μονάδα παραγωγής αρχικών διευθύνσεων, με τις διαφορές ότι οι μετρητές αυξάνονται κατά ένα κι όχι κατά τέσσερα, ότι τα όριά τους είναι από το ένα έως το τέσσερα κι ότι δεν υπάρχει



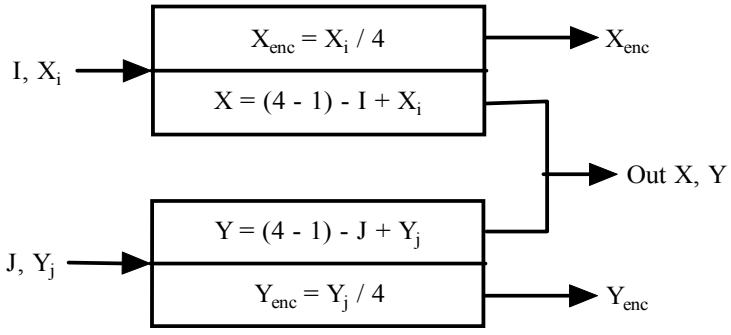
Σχήμα 4.22 Η λειτουργία της μονάδας παραγωγής αρχικών διευθύνσεων

κατάσταση προσωρινής αναστολής της λειτουργίας της, αφού δεν υπάρχει λόγος για κάτι τέτοιο. Οι μικρές αυτές διαφοροποιήσεις αποτυπώνονται στο διάγραμμα καταστάσεων του Σχήματος 4.23. Η μονάδα αυτή παράγει διευθύνσεις για παράθυρα μεγέθους 4 × 4. Για το λόγο αυτό οι μετρητές I και J μετράνε μόνο έως το 4, ενώ αυξάνονται κάθε φορά κατά 1.

Τα I και J που υπολογίζονται οδηγούνται στο τμήμα υπολογισμού των τελικών διευθύνσεων. Εκεί οδηγούνται και τα δεδομένα από τη μονάδα παραγωγής αρχικών διευθύνσεων. Τα 12 bits αυτών χωρίζονται σε δύο ποσότητες των 6 bits, η μία από τις οποίες αντιστοιχεί στον αριθμό των γραμμών (έστω X_i) και η άλλη στον αριθμό των στηλών (έστω Y_j). Αυτές προστίθενται στις τιμές που αφορούν το κάθε 4 × 4 παράθυρο, ώστε να παράγονται οι σωστές διευθύνσεις. Οι δύο επιπλέον 4-bit έξοδοι της μονάδας παραγωγής τελικών διευθύνσεων υπολογίζονται κι αυτές στο τμήμα παραγωγής τελικών διευθύνσεων και προκύπτουν από τη διαίρεση με το 4 των διευθύνσεων της μονάδας παραγωγής αρχικών διευθύνσεων. Αυτές οι έξοδοι θα οδηγηθούν στη μονάδα κωδικοποίησης, όπου θα χρησιμεύσουν για την καταγραφή των συντεταγμένων κάθε παραθύρου 4 × 4 στο παράθυρο 64 × 64 στο οποίο ανήκει. Το τμήμα υπολογισμού των τελικών διευθύνσεων παρουσιάζεται στο Σχήμα 4.24.



Σχήμα 4.23 Η FSM της μονάδας παραγωγής τελικών διευθύνσεων



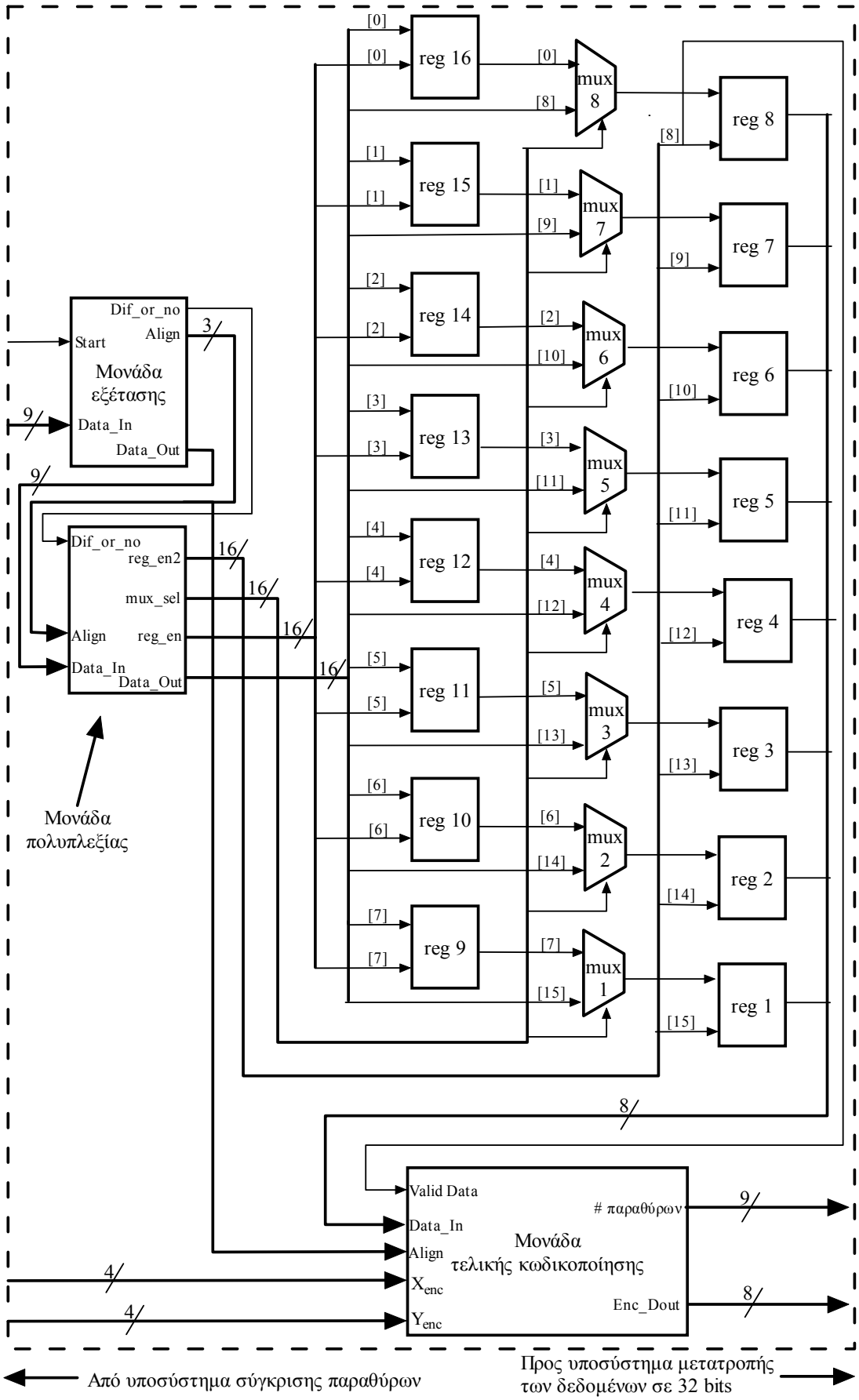
Σχήμα 4.24 Το τμήμα υπολογισμού των τελικών διευθύνσεων της μονάδας παραγωγής των τελικών διευθύνσεων

Τέλος, αναφέρεται ότι ο address generator αρχίζει τη λειτουργία του αφού έχουν γραφεί περίπου 190 τιμές στην ενδιάμεση RAM. Αυτό συμβαίνει διότι, όπως αναφέρθηκε και πριν, τα δεδομένα γράφονται σε αυτή σειριακά, ενώ διαβάζονται μη σειριακά. Οι πρώτες 16 διευθύνσεις που πρέπει να διαβαστούν, δηλαδή αυτές που αφορούν το πρώτο 4 × 4 παράθυρο, είναι διαδοχικά οι 0, 1, 2, 3, 64, 65, 6, 67, 128, 129, 130, 131, 192, 193, 194, 195. Αυτό σημαίνει ότι όταν αρχίσει να υπολογίζει διευθύνσεις ο address generator πρέπει να έχουν γραφεί στη μνήμη οι τιμές που πρόκειται να διαβαστούν, κάτι που δε θα συνέβαινε εάν αυτός άρχιζε μόλις γραφόταν στη μνήμη η πρώτη τιμή του παραθύρου διαφορών.

4.5 Το υποσύστημα κωδικοποίησης των παραθύρων

Το υποσύστημα αυτό δέχεται ως είσοδο τα παράθυρα διαφορών αναλυμένα σε παράθυρα 4 × 4 και εκτελεί όλα τα υπόλοιπα βήματα του αλγορίθμου SCAN. Ελέγχει κάθε παράθυρο (στο παρόν υποσύστημα με τον όρο «παράθυρα» αναφέρονται τα παράθυρα ανάλυσης 4 × 4, αφού αυτά δέχεται ως είσοδο) για να διαπιστωθεί αν πρέπει να κωδικοποιηθεί και μόνο τότε κωδικοποιείται με τον τρόπο που θα περιγράφεται στη συνέχεια της παραγράφου.

Το υποσύστημα αποτελείται από μία μονάδα εξέτασης των παραθύρων, μία μονάδα πολυπλεξίας και μία μονάδα τελικής κωδικοποίησης, ενώ ανάμεσα στις δύο τελευταίες μονάδες μεσολαβεί μία διάταξη 16 καταχωρητών 1-bit και 8 πολυπλεκτών 2-σε-1, όπως φαίνεται και στο block διάγραμμα του Σχήματος 4.25. Τα δεδομένα από το υποσύστημα σύγκρισης των παραθύρων εισέρχονται στη μονάδα εξέτασης των παραθύρων. Εκεί γίνεται ο έλεγχος αν το κάθε παράθυρο πρέπει να κωδικοποιηθεί ή όχι, δηλαδή ουσιαστικά αν περιέχει τουλάχιστον ένα στοιχείο διαφορετικό του 100000000. Ανάλογα με το αποτέλεσμα του κάθε ελέγχου δίδονται στη μονάδα πολυπλεξίας τα κατάλληλα δεδομένα. Η μονάδα αυτή εκτελεί το πολύ σημαντικό έργο του ελέγχου αλλά και της τροφοδοσίας με δεδομένα των καταχωρητών και των πολυπλεκτών. Τελικά, από τις εξόδους των καταχωρητών 1-8 προκύπτουν συνολικά 8 bits δεδομένων, τα οποία οδηγούνται στη μονάδα τελικής κωδικοποίησης. Με τη χρήση των συντεταγμένων X_{enc} και Y_{enc} του κάθε παραθύρου, όπως αυτές παράγονται από τη μονάδα σύγκρισης των παραθύρων, προκύπτουν τα συμπιεσμένα δεδομένα. Η



Σχήμα 4.25 Block διάγραμμα του υποσυστήματος κωδικοποίησης των παραθύρων

μορφή των δεδομένων αυτών είναι ουσιαστικά ίδια με αυτή της υλοποίησης του αλγορίθμου σε software. Οι όποιες μικρές διαφοροποιήσεις αιτιολογούνται στη συνέχεια.

4.5.1 Η μονάδα εξέτασης των παραθύρων

Η μονάδα αυτή αρχίζει να λειτουργεί όταν το σήμα start, το οποίο ελέγχεται από την FSM του υποσυστήματος σύγκρισης των παραθύρων, γίνει ίσο με "1" (πρακτικά αμέσως μετά την έναρξη του address generator της ενδιάμεσης RAM). Τότε, όταν οι τιμές στην είσοδό της είναι έγκυρες, ελέγχεται το πρώτο bit της καθεμιάς. Εάν αυτό είναι "1", στην έξοδο της μονάδας οδηγούνται εννέα μηδενικά και το σήμα Dif_or_no παίρνει την τιμή μηδέν, διαφορετικά στην έξοδο οδηγούνται τα δεδομένα της εισόδου με το πρώτο τους bit ανεστραμμένο και το σήμα Dif_or_no παίρνει την τιμή "1". Γίνεται λοιπόν φανερό ότι το υποσύστημα σύγκρισης πρόσθεσε ένα επιπλέον bit στα δεδομένα της εξόδου του, γιατί με αυτό ελέγχεται αν αυτά περιέχουν χρήσιμη πληροφορία ή μηδενικά. Επίσης, για κάθε έγκυρη τιμή στην έξοδο το σήμα Align αυξάνεται κατά 1, ενώ παίρνει τιμές από ΄000' έως '111'. Η χρησιμότητα του σήματος αυτού φαίνεται στη συνέχεια. Ο Πίνακας 4.7 παρουσιάζει τις τιμές που βγάζει στις εξόδους της η μονάδα αυτή ανάλογα με τα δεδομένα της εισόδου της. Ουσιαστικά, η λειτουργία της μονάδας αυτής είναι βοηθητική της μονάδας πολυπλεξίας, αφού της παρέχει όλα τα απαραίτητα σήματα και δεδομένα.

4.5.2 Η μονάδα πολυπλεξίας

Η μονάδα αυτή ελέγχει τη διάταξη των 16 καταχωρητών και των 8 πολυπλεκτών, η οποία αποτελεί την καρδιά της υπομονάδας κωδικοποίησης. Αυτό που κάνει η

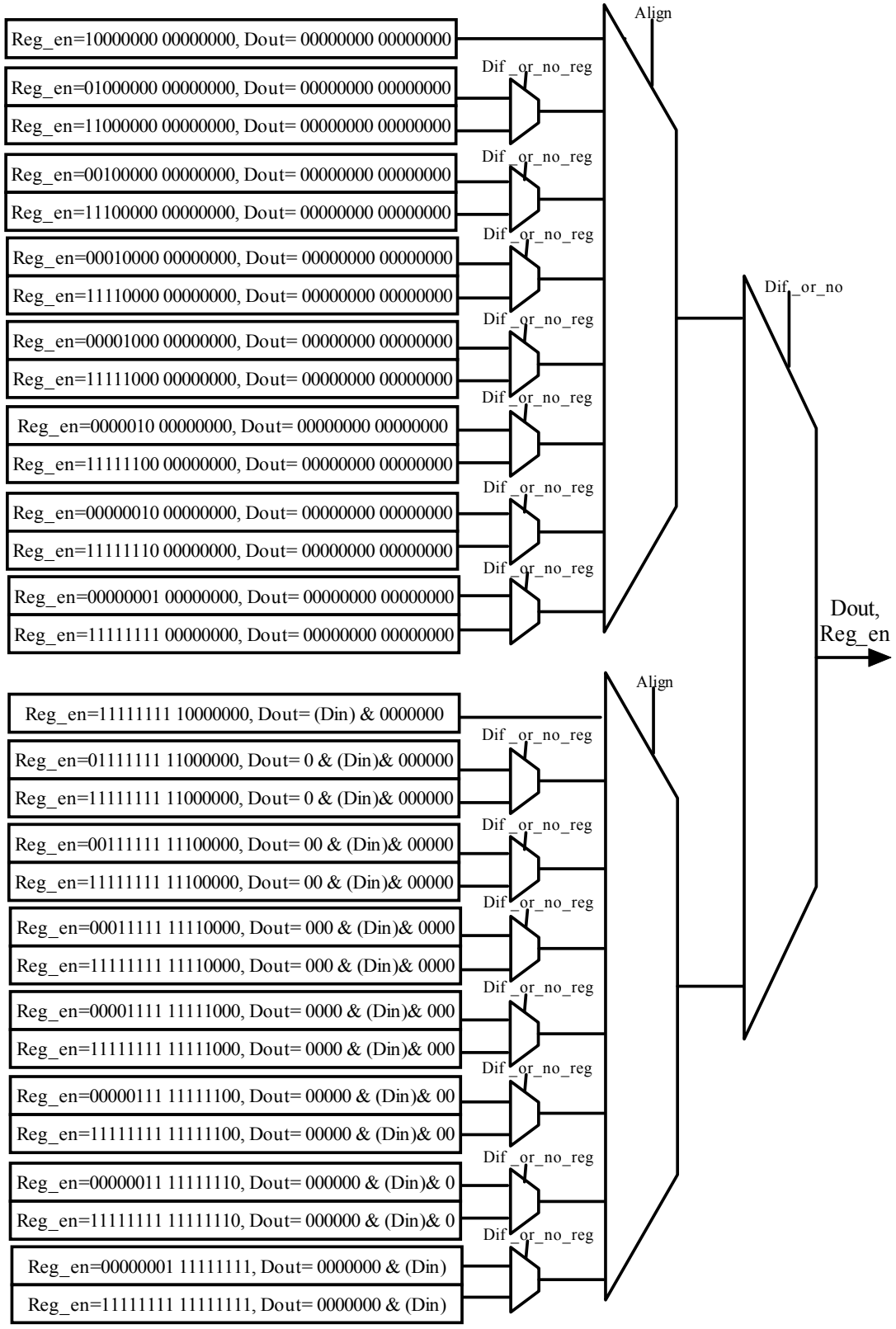
Τιμή δεδομένων εισόδου	**Τιμή δεδομένων εξόδου**	**Τιμή σήματος Dif_or_no**
1 0000 0000	0 0000 0000	0
0 XXXX XXXX	1 XXXX XXXX	1

Πίνακας 4.7 Οι τιμές που βγαίνουν στην έξοδο της μονάδας εξέτασης των παραθύρων σε σχέση με τις τιμές στην είσοδό της

διάταξη αυτή είναι να δέχεται στις εισόδους της δεδομένα ενός ή εννέα bits και να τα ομαδοποιεί με τρόπο τέτοιο ώστε να εξάγονται πάντα ανά οκτώ. Ρίχνοντας μία ματιά στο σημείο του αλγορίθμου SCAN όπου κωδικοποιούνται τα 4 × 4 παράθυρα φαίνεται ότι αν πρόκειται να αποθηκευτεί πληροφορία για εικονοστοιχείο του οποίου η τιμή είναι -1 αποθηκεύεται ένα μηδενικό (1 bit), ενώ αν πρόκειται για εικονοστοιχείο με διαφορετική τιμή αποθηκεύεται η τιμή ένα ακολουθούμενη από την τιμή του εικονοστοιχείου αυτού (συνολικά 9 bits). Η ίδια ακριβώς τακτική ακολουθείται και στη hardware υλοποίηση του αλγορίθμου, ενώ έπρεπε να βρεθεί τρόπος οι ποσότητες των εννέα και του ενός bit να συγκροτούν ομάδες των οκτώ, ώστε να γίνεται η κωδικοποίηση των παραθύρων σύμφωνα με τις προδιαγραφές του αλγορίθμου και να προκύπτει ένα όσο το δυνατόν πιο «συμπαγές» συμπιεσμένο αρχείο.

Τα δεδομένα που εξάγονται από τη μονάδα πολυπλεξίας, αρχίζουν να γράφονται στους καταχωρητές της διάταξης. Αν αυτά είναι 1 bit, πηγαίνουν αρχικά στον πολυπλέκτη 1 και στη συνέχεια στον καταχωρητή 1, τα επόμενα 1 bit δεδομένα πηγαίνουν στον πολυπλέκτη 2 και μετά στον καταχωρητή 2 και η διαδικασία συνεχίζεται μέχρι να φτάσουν δεδομένα στον καταχωρητή 8. Τότε υπάρχει συμπληρωμένη μία οκτάδα έγκυρων δεδομένων που οδηγούνται για επεξεργασία στη μονάδα τελικής κωδικοποίησης, ενώ τα επόμενα δεδομένα αρχίζουν να γράφονται πάλι στον καταχωρητή 1 για να δημιουργηθούν οι επόμενες οκτάδες δεδομένων. Στην περίπτωση που υπάρχουν δεδομένα των 9 bits, τότε αρχίζουν να γράφονται από τον επόμενο καταχωρητή, η εγγραφή τους συνεχίζεται όμως και σε όσους καταχωρητές πέρα από τον όγδοο χρειάζεται. Για παράδειγμα, αν έχουν γραφεί δεδομένα στους καταχωρητές 1 και 2 και στη συνέχεια έρθουν δεδομένα 9 bits, τα 6 πρώτα από αυτά θα οδηγηθούν στους πολυπλέκτες 3 – 8 και τα υπόλοιπα 3 θα αποθηκευτούν στους καταχωρητές 9 – 11. Στον επόμενο κύκλο ρολογιού τα δεδομένα από τους πολυπλέκτες οδηγούνται στους καταχωρητές 3 – 8, οπότε τα περιεχόμενα των καταχωρητών 1 – 8 οδηγούνται στη μονάδα τελικής κωδικοποίησης διότι μόλις συμπληρώθηκε μια οκτάδα δεδομένων. Παράλληλα, τα περιεχόμενα των καταχωρητών 9 – 11 βρίσκονται ήδη στις εισόδους των πολυπλεκτών 1 – 3, ενώ τα νέα δεδομένα που εισέρχονται, άσχετα με το αν είναι 1 ή 9 bits, αρχίζουν να γράφονται από τον καταχωρητή 4 και η όλη διαδικασία επαναλαμβάνεται.

Στο Σχήμα 4.26 παρουσιάζεται ο τρόπος λειτουργίας της μονάδας πολυπλεξίας. Αυτή αποτελείται από 3 επίπεδα πολυπλεκτών και ανάλογα με τις τιμές των σημάτων



Σχήμα 4.26 Ο τρόπος λειτουργίας της μονάδας πολυπλεξίας

ελέγχου τους προκύπτουν οι κατάλληλες τιμές στις εξόδους. Οι πολυπλέκτες του πρώτου επιπέδου ελέγχονται από το σήμα Dif_or_no_reg, το οποίο είναι το σήμα Dif_or_no που βγάζει η μονάδα εξέτασης παραθύρων, καθυστερημένο κατά 1 κύκλο, ενώ και οι υπόλοιποι πολυπλέκτες ελέγχονται από σήματα της μονάδας εξέτασης παραθύρων. Οι υπόλοιπες δύο έξοδοι της μονάδας πολυπλεξίας, οι Reg_en2 και Mux_en, οι οποίες δε φαίνονται στο Σχήμα 4.26 είναι ουσιαστικά η έξοδος Reg_en, καθυστερημένη κατά 1 κύκλο.

Το σήμα Dif_or_no ελέγχει αν οι τιμές που θα οδηγηθούν στις εξόδους προέρχονται από είσοδο ενός ή εννέα bits. Ανάλογα με την τιμή της εισόδου, το ίδιο σήμα καθυστερημένο κατά 1 κύκλο θα καθορίσει τα enables των καταχωρητών και τα select των πολυπλεκτών, ενώ με τη βοήθεια του σήματος Align τα σήματα δίνονται πάντα στους καταχωρητές και τους πολυπλέκτες για τους οποίους προορίζονται. Σημειώνεται ότι οι 2-σε-1 πολυπλέκτες όταν δέχονται μηδέν στην είσοδο ελέγχου τους περνάνε στην έξοδό τους το σήμα που βρίσκεται στην πάνω είσοδό τους, ειδάλλως περνάνε αυτό που βρίσκεται στην κάτω. Αντίστοιχα, οι 8-σε-1 πολυπλέκτες, όταν δέχονται σήμα ελέγχου ‘000’ περνάνε την πρώτη τους είσοδο στην έξοδο, ενώ όσο το σήμα αυτό αυξάνει κατά 1 περνάει στην έξοδο η επόμενη κατά σειρά είσοδος. Θεωρώντας σαν MSB των τιμών των εξόδων της μονάδας πολυπλεξίας το αριστερότερο bit, παρατηρώντας το Σχήμα 4.25 διακρίνεται ποιό bit ακριβώς της κάθε 16-bit τιμής των εξόδων αυτών οδηγείται σε κάθε πολυπλέκτη ή καταχωρητή, αφού στην κάθε είσοδό τους υπάρχουν γραμμένα τα αντίστοιχα νούμερα.

Επίσης, τονίζεται ότι ανάμεσα σε κάθε δύο οκτάδες δεδομένων που εισάγονται στο υποσύστημα κωδικοποίησης των παραθύρων μεσολαβεί ένα μικρό διάστημα χωρίς έγκυρα δεδομένα, οπότε στην περίπτωση που κατά τα όγδοα έγκυρα δεδομένα είναι γεμάτοι οι καταχωρητές 9 – 16, έχουν τον απαιτούμενο 1 κύκλο ρολογιού ώστε να αδειάσουν και να είναι πάλι έτοιμοι να δεχτούν δεδομένα. Το διάστημα χωρίς έγκυρα δεδομένα δεν έγινε με σκοπό να εξυπηρετηθεί η λειτουργία του υποσυστήματος κωδικοποίησης, αλλά προέρχεται από το διάστημα που μεσολαβεί ανάμεσα σε δύο διαδοχικές αναγνώσεις από την SDRAM και διαρκεί 3 κύκλους.

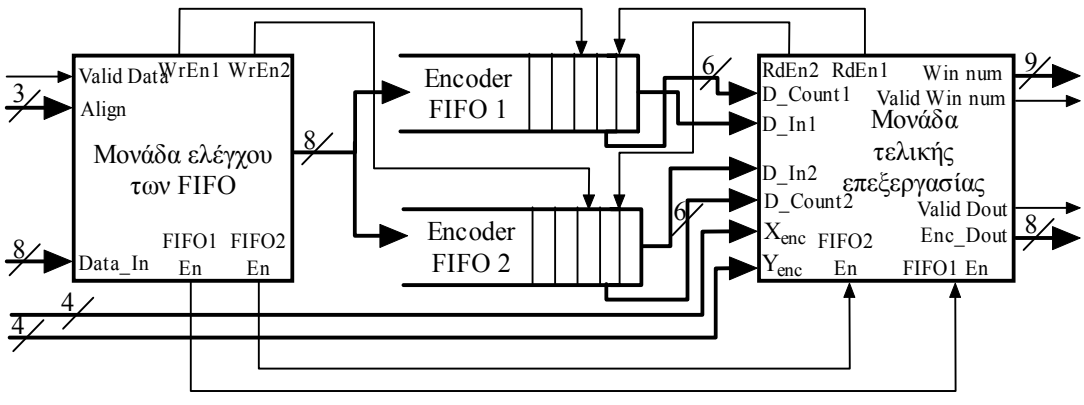
4.5.2 Η μονάδα τελικής κωδικοποίησης

Η μορφή των δεδομένων που εισέρχονται στη μονάδα αυτή προσεγγίζει αρκετά τη συμπιεσμένη τελική μορφή που αυτά πρέπει να έχουν, όμως χρειάζεται ακόμα να υποστούν ορισμένες αλλαγές. Πριν από τα δεδομένα που αφορούν στο κάθε 4 × 4 παράθυρο δεν υπάρχουν οι συντεταγμένες του (δηλαδή η θέση στην οποία βρίσκονται στο παράθυρο των διαφορών), ενώ αν οι τιμές κάποιου παραθύρου είναι όλες εντός κατωφλίου, περίπτωση στην οποία δεν κρατιέται κανένα δεδομένο, προς το παρόν αυτό αντιπροσωπεύεται από δύο οκτάδες μηδενικών.

Τα δεδομένα που εισέρχονται στη μονάδα αυτή, της οποίας το block διάγραμμα παρουσιάζεται στο Σχήμα 4.27, αρχικά γράφονται ανά 22 κύκλους ρολογιού σε καθεμιά από τις δύο διαθέσιμες FIFO. Ο αριθμός 22 επιλέχτηκε διότι ανά τόσους κύκλους οι τιμές που εισέρχονται στη μονάδα αντιστοιχούν σε διαφορετικά 4 × 4 παράθυρα. Στη συνέχεια επεξεργάζονται κατάλληλα στη μονάδα τελικής επεξεργασίας. Η εγγραφή των δεδομένων στις FIFO ελέγχεται από τη μονάδα ελέγχου των FIFO.

4.5.2.1 Η μονάδα ελέγχου των FIFO

Η μονάδα αυτή μεταφέρει τα έγκυρα δεδομένα κάθε παραθύρου που έρχονται στην είσοδό της στην κατάλληλη FIFO. Τα δεδομένα χωρίζονται σε δύο FIFO διότι με τον τρόπο αυτό διευκολύνεται η διαδικασία της επεξεργασίας. Όσο διαβάζεται η μία FIFO από τη μονάδα τελικής επεξεργασίας και τα δεδομένα της επεξεργάζονται, η άλλη FIFO γράφεται από τη μονάδα ελέγχου των FIFO.



Σχήμα 4.27 Block διάγραμμα της μονάδας τελικής κωδικοποίησης

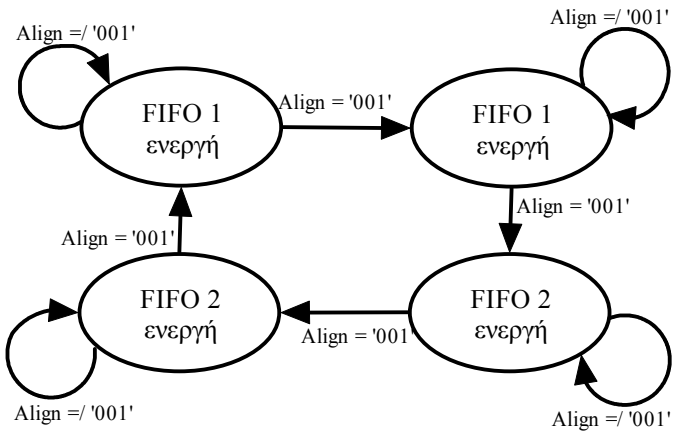
Για να γίνει η επιλογή της κατάλληλης FIFO για τα εισερχόμενα δεδομένα χρησιμοποιείται το σήμα Align. Κάθε φορά που αυτό παίρνει μια συγκεκριμένη τιμή έχουν εισαχθεί στη μονάδα δεδομένα που αφορούν οκτώ εικονοστοιχεία, δηλαδή μισό παράθυρο. Έτσι, κάθε δεύτερη φορά που το Align παίρνει την τιμή '001' η εγγραφή των εισερχόμενων δεδομένων γίνεται και σε διαφορετική FIFO, δηλαδή γίνεται ενεργή διαφορετική FIFO. Αυτό παρουσιάζεται και στο Σχήμα 4.28.

Αν είναι ενεργή η FIFO 1, το σήμα FIFO1_En έχει τιμή ένα, διαφορετικά η τιμή του είναι μηδέν και το FIFO2_En έχει τιμή ένα. Τα σήματα αυτά οδηγούνται στη μονάδα τελικής επεξεργασίας. Σχετικά με τον τρόπο που δίνεται τιμή στα write enable των FIFO, όταν το σήμα Valid είναι ένα, που σημαίνει ότι τα δεδομένα της εισόδου είναι έγκυρα, γίνεται ένα το write enable της ενεργής FIFO.

Σχετικά με το μέγεθος των FIFO, το πλάτος τους είναι φυσικά 8 bits, ενώ το βάθος τους είναι 32 λέξεις. Ο μέγιστος αριθμός δεδομένων που καλείται να αποθηκεύσει η καθεμία είναι 18, διότι αν όλα τα εικονοστοιχεία ενός παραθύρου είναι εκτός κατωφλίου απαιτούνται 16 × 9 = 144 bits ή 18 οκτάδες για την περιγραφή του. Το αμέσως μεγαλύτερο μέγεθος για τη FIFO που ήταν διαθέσιμο μετά το 16 ήταν το 32, το οποίο και επιλέχτηκε.

4.5.2.2 Η μονάδα τελικής επεξεργασίας

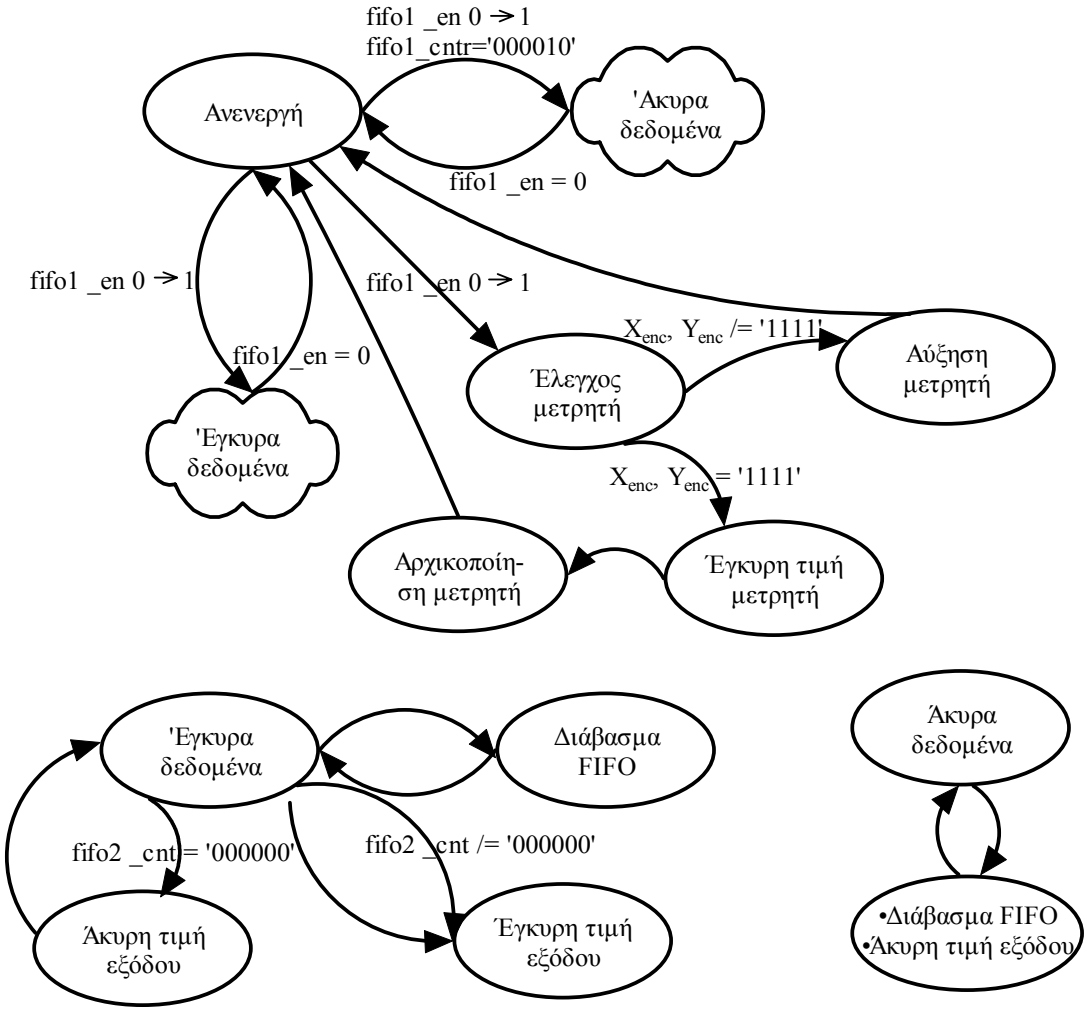
Η μονάδα τελικής επεξεργασίας έχει δύο εξόδους. Στη μία από αυτές οδηγούνται τα δεδομένα των συμπιεσμένων παραθύρων, ενώ στην άλλη ο αριθμός των 4 × 4 παραθύρων κάθε 64 × 64 παραθύρου που έχουν κωδικοποιηθεί. Ο λόγος που τα



Σχήμα 4.28 Η λειτουργία της μονάδας ελέγχου των FIFO

συμπιεσμένα δεδομένα βγαίνουν σε δύο διαφορετικές εξόδους είναι καθαρά πρακτικός και αναλύεται πιο κάτω.

Τα βασικά τμήματα από τα οποία αποτελείται είναι δύο σχεδόν ίδιες FSM, καθεμία από τις οποίες ελέγχει μία FIFO. Η λειτουργία μιας από αυτές, έστω της FIFO 2, φαίνεται στο Σχήμα 4.29, ενώ η μόνη τος διαφορά έγκειται στο ότι η μία από τις δύο δεν έχει καθόλου τις καταστάσεις για την αρχικοποίηση και την έγκυρη τιμή του μετρητή. Όταν το σήμα που δείχνει αν η FIFO 1 είναι ενεργή μεταβεί από το "0" στο "1", σημαίνει ότι στη FIFO 2 βρίσκονται γραμμένα δεδομένα για ένα ολόκληρο παράθυρο και περιμένουν να διαβαστούν. Τη στιγμή της μετάβασης αυτής ελέγχεται η τιμή του μετρητή που δείχνει πόσες θέσεις της FIFO 2 περιέχουν έγκυρα δεδομένα. Αν αυτές είναι 2 σημαίνει ότι όλες οι τιμές του παραθύρου που αντιπροσωπεύουν είναι εντός κατωφλίου και δε χρειάζεται να κωδικοποιηθεί (προφανώς οι 2 τιμές



Σχήμα 4.29 Μία από τις δύο FSM της μονάδας τελικής επεξεργασίας

αποτελούνται μόνο από μηδενικά, αφού σε διαφορετική περίπτωση έπρεπε να είναι τουλάχιστον 3). Τότε ενεργοποιείται το read enable της FIFO 2 ώστε να αδειάσει, αλλά το valid της εξόδου παραμένει στο μηδέν. Αν οι τιμές είναι πάνω από 2, τότε το παράθυρο κωδικοποιείται κανονικά. Τα read enable και valid παίρνουν την τιμή 1, ενώ προτού η πρώτη τιμή της FIFO φτάσει στην έξοδο εκεί οδηγούνται τα X_{enc} και Y_{enc}, των οποίων οι τιμές αντιστοιχούν στις συντεταγμένες του τρέχοντος παραθύρου. Το καθένα είναι από 4 bits, οπότε και τα δύο μαζί καλύπτουν ακριβώς μία θέση των 8 bits, την πρώτη του παραθύρου, σύμφωνα με τον αλγόριθμο. Όταν ο μετρητής των τιμών της FIFO πάρει την τιμή μηδέν το σήμα valid γίνεται μηδέν, εφόσον παύουν να φτάνουν έγκυρα δεδομένα στην έξοδο. Στη συνέχεια, όταν το σήμα που δείχνει ότι η FIFO 2 είναι ενεργή μεταβεί από μηδέν σε ένα, αρχίζει η ίδια διαδικασία για τη FIFO 1, κ.ο.κ.

Σχετικά με την έξοδο που δείχνει τον αριθμό των κωδικοποιημένων παραθύρων σε κάθε μεγαλύτερο παράθυρο, αυτός προκύπτει με τη βοήθεια δύο μετρητών. Η κάθε μία FSM ελέγχει έναν εξ' αυτών. Σε κάθε μετάβαση από μηδέν σε ένα των σημάτων που δείχνουν ποια FIFO είναι ενεργή, όταν υπάρχουν πάνω από 2 τιμές στην αντίστοιχη FIFO αυξάνεται κατά 1 ο αντίστοιχος μετρητής. Το άθροισμα των μετρητών αυτών δείχνει κάθε στιγμή τον αριθμό των παραθύρων που έχουν κωδικοποιηθεί. Όταν τα σήματα X_{enc} και Y_{enc} πάρουν τις τιμές '1111' και '1111' σημαίνει ότι διατρέχεται το τελευταίο 4×4 παράθυρο του 64×64 παραθύρου, οπότε το άθροισμα των μετρητών οδηγείται στην έξοδο Win Number και το αντίστοιχο enable γίνεται ένα. Αμέσως μετά αυτό γίνεται πάλι μηδέν, ενώ οι μετρητές μηδενίζονται ώστε η διαδικασία να ξαναρχίσει για το επόμενο 64×64 παράθυρο.

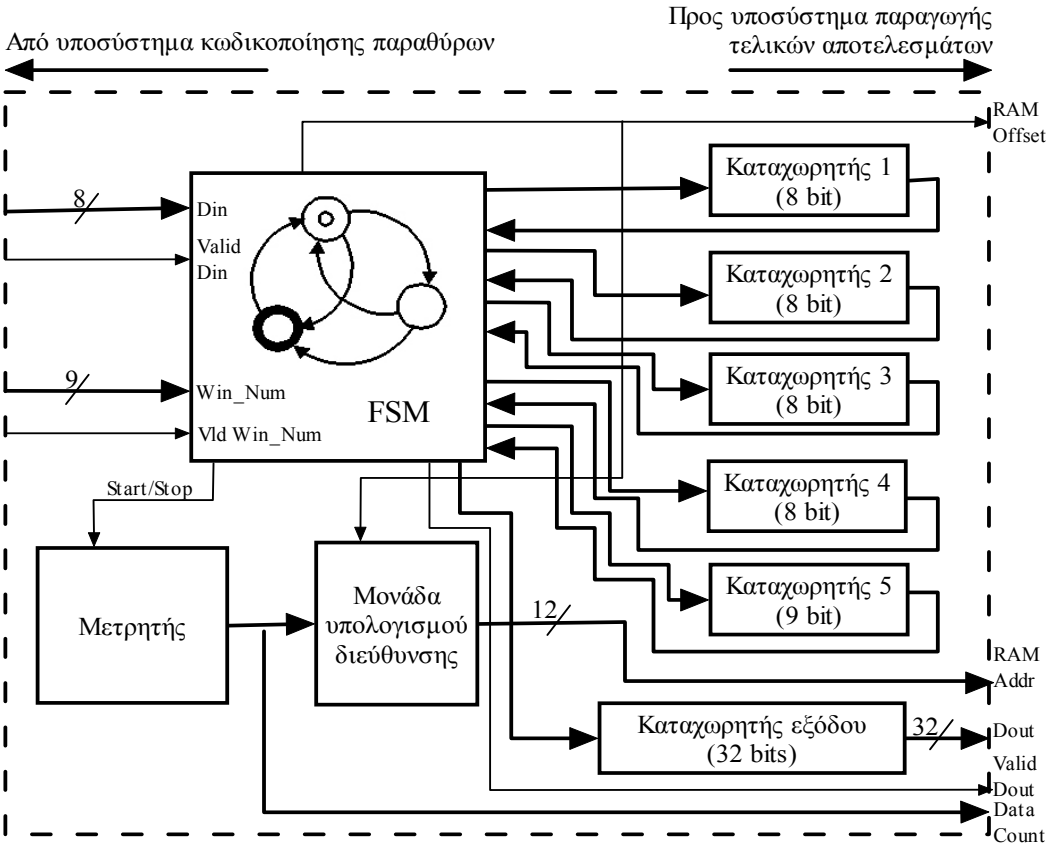
4.6 Το υποσύστημα μετατροπής των δεδομένων σε 32 bits

Το υποσύστημα μετατροπής των δεδομένων σε 32 bits, αν και δε φαίνεται στο block διάγραμμα του Σχήματος 4.1, μεσολαβεί ανάμεσα στο υποσύστημα κωδικοποίησης και στη δεύτερη ενδιάμεση μνήμη RAM [39]. Αυτό που κάνει είναι να μετατρέπει τα δεδομένα των 8 και 9 bits που δέχεται στις εισόδους του σε 32 bits, ώστε να μπορούν να γραφούν στη δεύτερη ενδιάμεση μνήμη. Τα δεδομένα που επεξεργάζονται από το σύστημα είναι 32 bits, οπότε και τα δεδομένα της εξόδου του, άρα κατ' επέκταση τα

δεδομένα που είναι αποθηκευμένα στην κάθε ενδιάμεση μνήμη, αφού αυτά οδηγούνται στην έξοδο, πρέπει να είναι 32 bits.

Το υποσύστημα αποτελείται από τέσσερις 8-bit καταχωρητές, έναν 9-bit καταχωρητή κι έναν 32-bit καταχωρητή, στον οποίο αποθηκεύονται τα δεδομένα πριν οδηγηθούν στην έξοδο του υποσυστήματος. Μία FSM ελέγχει τη λειτουργία του υποσυστήματος. Επίσης, ο address generator της δεύτερης ενδιάμεσης μνήμης είναι ενσωματωμένος στο υποσύστημα και αποτελείται από ένα μετρητή και μια μονάδα υπολογισμού της διεύθυνσης. Όλα αυτά παρουσιάζονται στο block διάγραμμα του Σχήματος 4.30.

Η FSM αρχικά γράφει τα δεδομένα που έρχονται στην είσοδο Din διαδοχικά στους καταχωρητές 1, 2, 3 και 4. Μόλις γεμίσει και ο τέταρτος, τα δεδομένα και των τεσσάρων καταχωρητών οδηγούνται στον καταχωρητή εξόδου, όπου και συνιστούν μια ενιαία ποσότητα 32 bits. Όσο εισέρχονται στο υποσύστημα έγκυρα δεδομένα (δηλαδή όταν Valid_Din = 1) η διαδικασία συνεχίζεται. Στην περίπτωση που η τιμή του αριθμού των παραθύρων είναι έγκυρη, σημαίνει ότι έως τότε έχουν σταλεί δεδομένα για ένα 64 × 64 παράθυρο. Η έγκυρη αυτή τιμή αποθηκεύεται στον

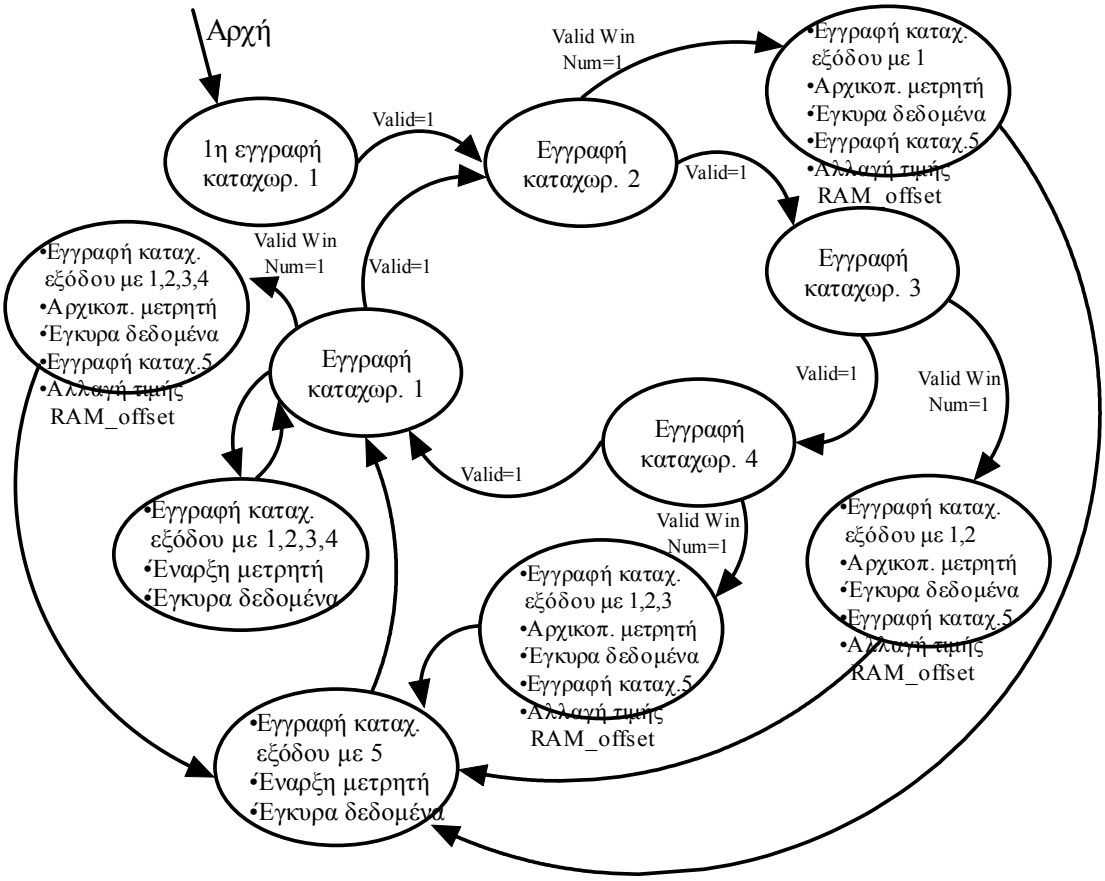


Σχήμα 4.30 Block διάγραμμα του υποσυστήματος μετατροπής των δεδομένων σε 32 bits

καταχωρητή 5, ενώ στον καταχωρητή εξόδου μεταφέρονται όσες από τις τιμές των καταχωρητών 1 – 4 δεν είχαν προλάβει να μεταφερθούν εκεί και οι κενές του θέσεις συμπληρώνονται με μηδενικά. Στη συνέχεια, οδηγείται στον καταχωρητή εξόδου η τιμή του καταχωρητή 5, ακολουθούμενη από 23 μηδενικά. Η διαδικασία αρχίζει από την αρχή για το επόμενο παράθυρο.

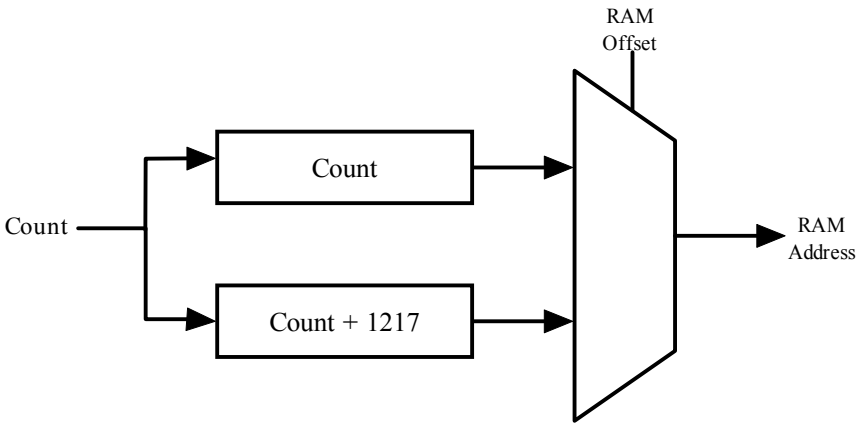
Κάθε φορά που γράφονται τα δεδομένα των καταχωρητών 1 – 4 στον καταχωρητή εξόδου αυξάνεται κατά 1 ο μετρητής που δίνει διευθύνσεις στη RAM, διότι στην έξοδο του παρόντος υποσυστήματος υπάρχουν νέα δεδομένα προς εγγραφή, ενώ γίνεται ένα και το σήμα valid της εξόδου. Σημειώνεται ότι πριν εγγραφεί η τιμή του καταχωρητή 5 ο μετρητής των διευθύνσεων της RAM μηδενίζεται και η τιμή του καταχωρητή 5 γράφεται στην πρώτη θέση της, που έχει μείνει κενή για το σκοπό αυτό. Έτσι, η μορφή των συμπιεσμένων δεδομένων είναι όμοια με αυτή του αλγορίθμου SCAN, διότι πριν από τα δεδομένα για κάθε συμπιεσμένο 64 × 64 παράθυρο υπάρχει το πεδίο που αναφέρει πόσα 4 × 4 παράθυρα αυτού έχουν κωδικοιποιηθεί. Επίσης, όταν το σήμα που δείχνει ότι η τιμή του αριθμού των παραθύρων είναι έγκυρη γίνεται ένα, αλλάζει η τιμή του σήματος RAM_Offset. Η FSM του παρόντος υποσυστήματος παρουσιάζεται στο Σχήμα 4.31.

Το σήμα RAM_Offset, εκτός του ότι είναι έξοδος του υποσυστήματος, ελέγχει και τη μονάδα υπολογισμού διεύθυνσης, η οποία φαίνεται στο Σχήμα 4.32. Πιο συγκεκριμένα, ελέγχει τον πολυπλέκτη της μονάδας αυτής και καθορίζει αν η έξοδός του θα είναι η έξοδος του μετρητή ως έχει, ή αυξημένη κατά 1217. Στη συνέχεια θα εξηγηθεί γιατί προστίθεται αυτή η τιμή. Υπενθυμίζεται ότι το μέγεθος κάθε δεύτερης ενδιάμεσης RAM είναι 2434 × 32 bits. Αυτό επιλέχτηκε να είναι τόσο ώστε η μνήμη να χωράει δύο ολόκληρα συμπιεσμένα παράθυρα 64 × 64 με τη μικρότερη δυνατή συμπίεση που μπορεί να επιτευχθεί. Ίσως να φαίνεται αρχικά παράξενο ότι το μέγιστο μέγεθος δύο συμπιεσμένων παραθύρων μπορεί να είναι μεγαλύτερο από αυτό των αρχικών, όμως αν εξεταστεί προσεκτικά ο τρόπος με τον οποίο γίνεται η συμπίεση κάθε 4 × 4 παραθύρου προκύπτει ότι αν όλα τα εικονοστοιχεία του είναι εκτός κατωφλίου, το συμπιεσμένο παράθυρο έχει μέγεθος ίσο με αυτό 19 εικονοστοιχείων, ενώ αρχικά είχε 16 (το 19 προκύπτει από τα 9 bits που χρειάζονται για την αναπαράσταση κάθε 8-bit εικονοστοιχείου και από το 8-bit πεδίο που δείχνει τις συντεταγμένες του 4 × 4 παραθύρου). Αν όλα τα παράθυρα χρειάζονταν από 19 bits για να συμπιεστούν προκύπτει το μέγεθος της μνήμης που επιλέχτηκε. Για την



Επεξήγηση:
Εγγραφή καταχ. εξόδου με 1,2,3,4: Εγγραφή στον καταχ. εξόδου των τιμών των καταχ. 1, 2, 3, 4.
Αντίστοιχα ισχύουν και για τις υπόλοιπες καταστάσεις.

Σχήμα 4.31 Η FSM του υποσυστήματος μετατροπής των δεδομένων σε 32 bits



Σχήμα 4.32 Block διάγραμμα της μονάδας υπολογισμού διεύθυνσης

ακρίβεια, η κάθε ενδιάμεση RAM έχει δύο επιπλέον θέσεις στις οποίες γράφεται ο αριθμός των 4 × 4 παραθύρων κάθε 64 × 64 παραθύρου που έχουν κωδικοποιηθεί. Πρακτικά αυτή η μνήμη είναι απίθανο να γεμίσει, αλλά για κάθε ενδεχόμενο

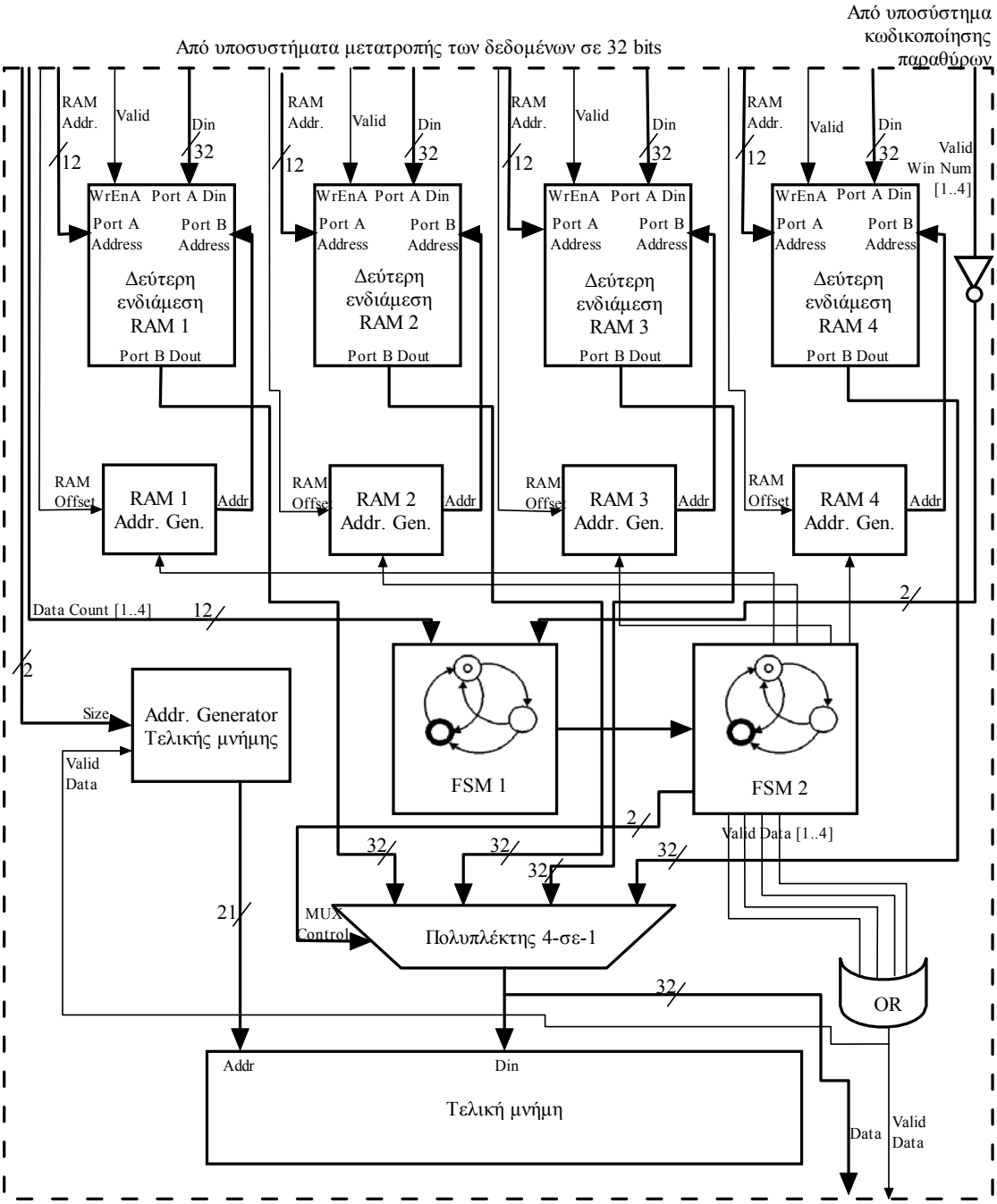
επιλέχτηκε να μη γίνει μικρότερη. Άλλωστε η απαιτούμενη ποσότητα μνήμης υπήρχε διαθέσιμη στην FPGA.

Στην δεύτερη ενδιάμεση μνήμη κρίθηκε αναγκαίο να αποθηκεύονται δύο συμπιεσμένα παράθυρα και ο λόγος για τον οποίο έγινε αυτό θα εξηγηθεί στην επόμενη παράγραφο. Όσον αφορά στη μονάδα υπολογισμού διεύθυνσης και στην τιμή 1217, όπως έχει φανεί από όσα ειπώθηκαν, όταν γράφεται το πρώτο παράθυρο στη μνήμη χρησιμοποιείται η τιμή του μετρητή, ενώ όταν γράφεται το δεύτερο στην τιμή του μετρητή προστίθεται η τιμή 1217 και η εγγραφή αρχίζει από τη μέση της μνήμης. Κατά την εγγραφή του δευτέρου γίνεται ανάγνωση του πρώτου, οπότε το τρίτο παράθυρο εγγράφεται από την αρχή της μνήμης, κ.ο.κ.

4.7 Το υποσύστημα παραγωγής τελικών αποτελεσμάτων

Το υποσύστημα αυτό είναι το τελευταίο του συστήματος. Έχει αναλάβει τον έλεγχο των αναγνώσεων από τις δεύτερες ενδιάμεσες μνήμες και την εγγραφή τους στην τελική μνήμη. Αποτελείται από τις τέσσερις δεύτερες ενδιάμεσες μνήμες με τους αντίστοιχους address generators, από δύο FSM, από έναν πολυπλέκτη 4-σε-1 και από την τελική μνήμη με τον αντίστοιχο address generator. Ο τρόπος με τον οποίο συνδέονται αυτές οι μονάδες παρουσιάζεται στο block διάγραμμα του Σχήματος 4.33. Τα δεδομένα εξάγονται από την κάθε RAM με τη βοήθεια του αντίστοιχου address generator και οδηγούνται στον πολυπλέκτη. Αυτός ελέγχεται από την FSM 2, η οποία καθορίζει και τις διευθύνσεις που θα παραχθούν από τον address generator της τελικής μνήμης. Επίσης, εκτός από την τελική μνήμη, τα δεδομένα των συμπιεσμένων καρέ οδηγούνται στην έξοδο του συστήματος, ενώ η FSM 2 ελέγχει και την τιμή του σήματος Valid, που δείχνει αν τα δεδομένα στην έξοδο είναι έγκυρα.

Πριν αναλυθεί η λειτουργία της κάθε μονάδας θα γίνει μια αναφορά στο λόγο ύπαρξης του υποσυστήματος αυτού, εφόσον τα δεδομένα είναι ήδη συμπιεσμένα. Ο λόγος για τον οποίο πρέπει κάθε συμπιεσμένο 64 × 64 παράθυρο να αποθηκευτεί αρχικά σε μια προσωρινή μνήμη (στη δεύτερη ενδιάμεση RAM) είναι ότι δεν μπορεί να είναι εκ των προτέρων γνωστό πόσες θέσεις μνήμης καταλαμβάνει. Χωρίς την πληροφορία αυτή είναι αδύνατο τα συμπιεσμένα δεδομένα να καταλαμβάνουν συνεχόμενες θέσεις στην τελική μνήμη, οπότε θα υπήρχε κατασπατάληση χώρου.



Σχήμα 4.33 Block διάγραμμα του υποσυστήματος παραγωγής τελικών αποτελεσμάτων

Όταν τα τέσσερα πρώτα συμπιεσμένα παράθυρα εγγραφούν στις αντίστοιχες δεύτερες ενδιάμεσες RAM, το μέγεθός τους είναι γνωστό. Τότε είναι δυνατό στην τελική μνήμη το δεύτερο παράθυρο να αρχίσει να εγγράφεται ακριβώς μετά την τελευταία θέση μνήμης που καταλαμβάνει το πρώτο, οπότε ο χώρος της μνήμης καλύπτεται πλήρως. Στη συνέχεια το τρίτο παράθυρο εγγράφεται αμέσως μετά το δεύτερο και το τέταρτο αμέσως μετά το τρίτο.

Όσο διαρκεί η εγγραφή των τεσσάρων παραθύρων στην τελική μνήμη, στο υπόλοιπο μισό των ενδιάμεσων μνημών γράφονται τα επόμενα παράθυρα. Αυτά πρέπει να γραφούν σε χωριστό τμήμα των μνημών από τα προηγούμενα διότι τα παράθυρα που είναι ήδη γραμμένα δεν διαβάζονται αμέσως, ώστε τα νέα δεδομένα να εγγράφονται στις θέσεις που ελευθερώνονται Σημειώνεται ότι ο μέγιστος χρόνος που απαιτείται για να γραφτεί το κάθε συμπιεσμένο παράθυρο στην τελική μνήμη είναι ίσος με το ¼ του μέγιστου χρόνου που χρειάζεται για να ολοκληρωθεί η εγγραφή των επόμενων συμπιεσμένων παραθύρων στις μνήμες RAM.

Ξεκινώντας την περιγραφή των μονάδων του υποσυστήματος από τους address generators των μνημών RAM, αυτοί αποτελούνται από ένα μετρητή και από μία μονάδα υπολογισμού διεύθυνσης και είναι πανομοιότυποι με τους address generators των υποσυστημάτων μετατροπής των δεδομένων σε 32 bits, οι οποίοι περιγράφηκαν στην προηγούμενη παράγραφο. Ελέγχονται βέβαια από το αντίστροφο του σήματος Align ώστε να γίνεται η ανάγνωση από το μισό της μνήμης στο οποίο δε γίνεται εγγραφή εκείνη τη στιγμή.

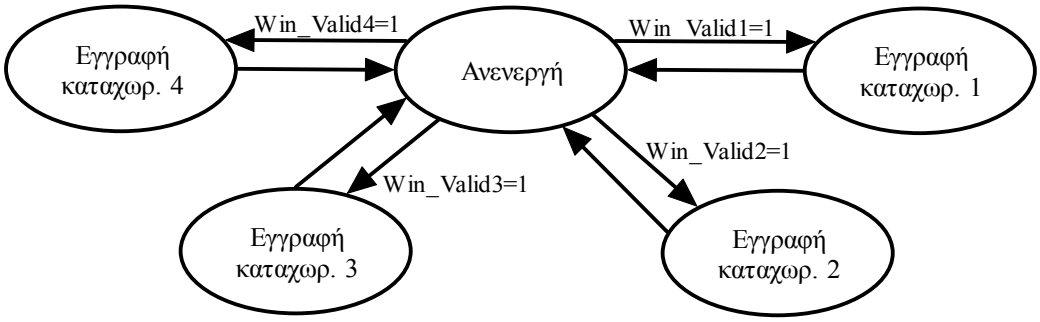
Ο address generator της τελικής μνήμης μοιάζει αρκετά με τους address generators των μνημών RAM. Αποτελείται και αυτός από ένα μετρητή και μια μονάδα υπολογισμού διεύθυνσης. Όσο υπάρχουν δεδομένα προς εγγραφή στην τελική μνήμη από οποιοδήποτε από τα τέσσερα παράθυρα ο μετρητής αυξάνεται. Η χρησιμότητα της μονάδας υπολογισμού διεύθυνσης έγκειται στο ότι ανάλογα με την ανάλυση του αρχικού video προσθέτει την κατάλληλη τιμή σε αυτές του μετρητή. Αυτό γίνεται διότι στην τελική μνήμη έχει ήδη αποθηκευτεί το πρώτο καρέ του video, εφόσον αυτό σε συμπιέζεται αλλά παραμένει ως έχει και χρησιμοποιείται κατά την αποσυμπίεση, σύμφωνα πάντα με τον αλγόριθμο SCAN, οπότε πρέπει η εγγραφή στην τελική μνήμη να μην αρχίζει από την πρώτη θέση, αλλά από συγκεκριμένο αριθμό θέσεων μετά από αυτή. Ένα καρέ ανάλυσης 128 × 128 καταλαμβάνει 4096 θέσεις μνήμης, ένα καρέ 256 × 256 καταλαμβάνει 16384 θέσεις και ένα καρέ 512 × 512 απαιτεί 65536 θέσεις, αναφερόμενοι πάντα σε μνήμη των 32 bits. Προφανώς, η μονάδα υπολογισμού διεύθυνσης προσθέτει στην τρέχουσα τιμή του μετρητή μία από αυτές τις τιμές.

Η FSM 1 αποθηκεύει σε τέσσερις καταχωρητές τον αριθμό των κωδικοποιημένων παραθύρων κάθε 64 × 64 παραθύρου. Όταν η τιμή του πρώτου από τα τέσσερα

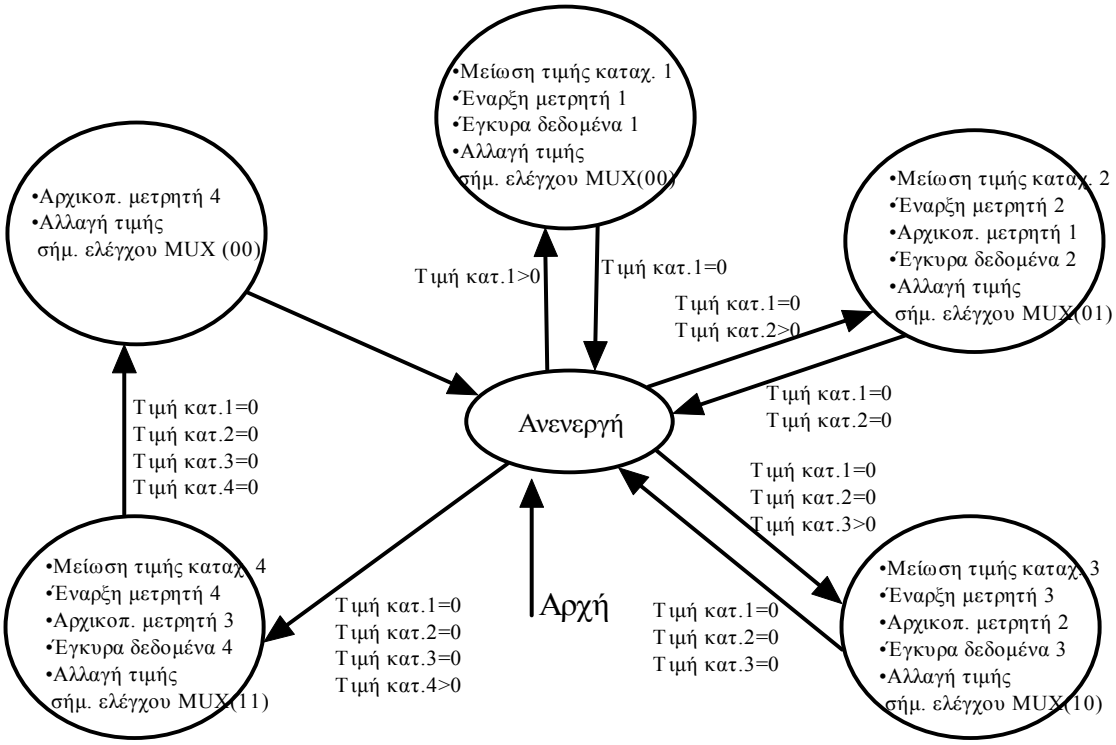
σήματα Valid_Win_Num, που προέρχονται από το υποσύστημα κωδικοποίησης, γίνει ένα, το οποίο σημαίνει ότι έχουν δοθεί στο υποσύστημα δεδομένα για ένα 64 × 64 παράθυρο, αποθηκεύεται στον πρώτο καταχωρητή η τιμή του αντίστοιχου μετρητή που δείχνει τον αριθμό των κωδικοποιημένων παραθύρων. Με τον ίδιο τρόπο γεμίζουν και οι άλλοι καταχωρητές. Το διάγραμμα καταστάσεων της FSM 1 παρουσιάζεται στο Σχήμα 4.34. Τα περιεχόμενα των τεσσάρων καταχωρητών χρησιμοποιούνται από την FSM 2.

Η FSM 2 ελέγχει τον πολυπλέκτη και τους address generators των μνημών RAM. Όταν στον πρώτο από τους τέσσερις καταχωρητές γραφεί κάποια τιμή, η FSM 2 αρχίζει να μειώνει την τιμή αυτή κατά 1 σε κάθε κύκλο ρολογιού, ενώ ταυτόχρονα αυξάνει κατά 1 την τιμή του μετρητή του address generator της πρώτης μνήμης RAM, κάνει 1 το σήμα Valid_Data και δίνει την τιμή ‘00’ στο σήμα ελέγχου του πολυπλέκτη. Μόλις η τιμή του πρώτου καταχωρητή γίνει μηδέν γίνεται το ίδιο και με τους υπόλοιπους τρεις διαδοχικά, διαδικασία που επαναλαμβάνεται. Με τον τρόπο αυτό επιτυγχάνεται να διαβάζονται από την κάθε μνήμη RAM ακριβώς τόσες θέσεις όσες περιέχουν έγκυρα δεδομένα, ενώ δίνοντας στο σήμα ελέγχου του πολυπλέκτη διαφορετική τιμή όταν διαβάζονται δεδομένα από διαφορετικές μνήμες γράφονται στην τελική μνήμη τα δεδομένα που πρέπει. Όσο διαβάζονται δεδομένα από οποιαδήποτε μνήμη RAM το αντίστοιχο σήμα valid είναι ένα. Τα τέσσερα αυτά σήματα εισάγονται σε μια πύλη OR, η έξοδος της οποίας οδηγείται στον address generator της τελικής μνήμης και όσο είναι ένα παράγονται διευθύνσεις για την εγγραφή στην τελική μνήμη. Η FSM 2 παρουσιάζεται στο Σχήμα 4.35.

Τέλος, σχετικά με την τελική μνήμη, όπως έχει ήδη αναφερθεί, δεν αντιστοιχεί σε πραγματική μνήμη, αλλά τοποθετήθηκε εκεί για πρακτικούς λόγους. Τα συμπιεσμένα



Σχήμα 4.34 Η FSM 1 του υποσυστήματος παραγωγής τελικών αποτελεσμάτων



Σχήμα 4.35 Η FSM 2 του υποσυστήματος παραγωγής τελικών αποτελεσμάτων

δεδομένα γράφονται μεν στη μνήμη αυτή, αλλά οδηγούνται και στην έξοδο του συστήματος μαζί με το αντίστοιχο σήμα valid, από όπου μπορεί να τροφοδοτηθεί με δεδομένα το σύστημα κρυπτογράφησης, το οποίο υπενθυμίζεται ότι κρυπτογραφεί τα συμπιεσμένα δεδομένα και τοποθετείται μετά το σύστημα συμπίεσης.

4.8 Επίλογος

Στο κεφάλαιο αυτό αναλύθηκε η αρχιτεκτονική του συστήματος και αιτιολογήθηκαν οι σχεδιαστικές επιλογές που έγιναν. Έχοντας πλέον μια σαφή εικόνα της αρχιτεκτονικής, αξίζει να σημειωθεί ότι τελικά η μνήμη SDRAM καθυστερεί το σύστημα. Όπως έχει αναφερθεί, η μνήμη SDRAM μπορεί να δώσει το πολύ 8 τιμές (των 32 bits) σε 11 κύκλους του ρολογιού του συστήματος. Όπως προέκυψε, οι 8 αυτές τιμές χρειάζονται 10 κύκλους για να επεξεργαστούν (η καθυστέρηση των δύο κύκλων γίνεται στον κωδικοποιητή), οπότε μία λίγο ταχύτερη μνήμη θα ήταν επιθυμητό να υπάρχει ώστε να βελτιωθεί η απόδοση κατά σχεδόν 10%.

Επίσης, σχετικά με το συμπιεσμένο αρχείο, ανακεφαλαιώνοντας τονίζεται ότι έγινε προσπάθεια κάθε ποσότητα των 32 bits που εξάγεται από το σύστημα να περιέχει μόνο χρήσιμη πληροφορία. Τελικά, η πρώτη και η τελευταία 32-μπιτη ποσότητα που αντιστοιχεί σε κάθε συμπιεσμένο 64 × 64 παράθυρο περιέχουν και μη χρήσιμη πληροφορία, γεγονός που τελικά οδηγεί, όπως θα φανεί και στο επόμενο καφάλαιο, σε συμπίεση περίπου 0,1% χειρότερη από την επιθυμητή. Πιο συγκεκριμένα, η πρώτη θέση περιέχει την τιμή που δείχνει πόσα κωδικοποιημένα 4 × 4 παράθυρα υπάρχουν σε όλο το παράθυρο και περιέχει μόνο 9 bits χρήσιμης πληροφορίας και η τελευταία θέση μπορεί να περιέχει 8, 16, 24, ή 32 χρήσιμα bits, ανάλογα με την κωδικοποίηση των παραθύρων.

Κεφάλαιο 5

Πιστοποίηση της λειτουργίας του συστήματος

Στο κεφάλαιο αυτό περιγράφεται αρχικά η διαδικασία των προσομοιώσεων και των δοκιμών που έγιναν. Επίσης, παρουσιάζονται τα αποτελέσματα των δοκιμών και γίνεται σύγκρισή τους με αυτά της software υλοποίησης του αλγορίθμου SCAN. Στη συνέχεια γίνεται αναφορά στους πόρους που απαιτεί η σχεδίαση από την FPGA στην οποία πρόκειται να απεικονιστεί, καθώς και στη μέγιστη συχνότητα ρολογιού στην οποία μπορεί να λειτουργήσει το σύστημα. Τέλος, γίνεται εκτίμηση της απόδοσης του συστήματος, για να διαπιστωθεί αν πληροί προδιαγραφές πραγματικού χρόνου.

5.1 Η διαδικασία της προσομοίωσης του συστήματος

Η προσομοίωση του συστήματος γινόταν συνεχώς και παράλληλα με τη διαδικασία ανάπτυξης της σχεδίασης. Η κάθε μονάδα προσομοιωνόταν σχολαστικά, ώστε να λειτουργεί σύμφωνα με τις προδιαγραφές της. Κάθε φορά που ήταν έτοιμες όλες οι μονάδες ενός υποσυστήματος, γινόταν προσομοίωση ολόκληρου του υποσυστήματος και μετά την ενσωμάτωσή του στα ήδη υπάρχοντα υποσυστήματα ακολουθούσε νέα προσομοίωση.

Ο βασικός τρόπος προσομοίωσης είναι η εισαγωγή επιθυμητών τιμών στις εισόδους μέσω do scripts και η παρακολούθηση των τιμών των εξόδων από κυματομορφές. Επίσης, τα περιεχόμενα της τελικής μνήμης μπορούν ανά πάσα στιγμή να γραφούν σε ένα αρχείο (δυνατότητα που παρέχεται από το μοντέλο της μνήμης που χρησιμοποιήθηκε), ώστε να είναι εύκολα προσβάσιμα. Μετά την ολοκλήρωση του συστήματος, παρά τους πολλούς έως τότε ελέγχους για τη σωστή λειτουργία του, έγινε και δειγματοληπτικός έλεγχος των περιεχομένων της τελικής μνήμης (δεν ήταν βέβαια δυνατό να ελεγχθούν όλα λόγω του πολύ μεγάλου αριθμού τους), τα αποτελέσματα του οποίου ήταν σε κάθε περίπτωση τα αναμενόμενα.

5.2 Η διαδικασία των δοκιμών του συστήματος

Ένα σημαντικό πρόβλημα που έπρεπε να αντιμετωπιστεί ήταν η εισαγωγή στην SDRAM των καρέ της κάθε ακολουθίας video στην κατάλληλη μορφή, η οποία απαιτεί το κάθε καρέ να αποτελείται μόνο από τις τιμές φωτεινότητας των εικονοστοιχείων του. Τα καρέ που εξάγονται από τα video περιέχουν και επιπλέον πληροφορίες οι οποίες είναι τοποθετημένες σε αυτά ως κεφαλίδες (headers). Για να γίνει η αποκοπή τους αναπτύχθηκε πρόγραμμα σε γλώσσα C, η ακριβής λειτουργία του οποίου δεν κρίνεται σκόπιμο να αναλυθεί. Η έξοδος του προγράμματος αυτού μετατρέπεται σε δυαδική μορφή με την κάθε λέξη να αποτελείται από 16 bits, οπότε τα δεδομένα είναι έτοιμα για εισαγωγή στις δύο μνήμες SDRAM, οι οποίες έχουν περιγραφεί στην παράγραφο 4.2.1. Επίσης, για τις δοκιμές της software υλοποίησης του αλγορίθμου που πραγματοποιήθηκαν, χρησιμοποιήθηκε απευθείας η έξοδος του προγράμματος C.

Λόγω της ενσωμάτωσης της διαδικασίας κρυπτογράφησης σε αυτή της συμπίεσης στη software υλοποίηση του αλγορίθμου δεν κατέστη δυνατή η εξαγωγή του συμπιεσμένου video σε κάποιο αρχείο, ώστε να γίνει απευθείας σύγκριση των αποτελεσμάτων software και hardware υλοποίησης. Παρ' όλ' αυτά, το μέγεθος του συμπιεσμένου αρχείου και το μέγεθος του συμπιεσμένου – κρυπτογραφημένου αρχείου είναι ακριβώς ίδια, οπότε για τις ίδιες ακολουθίες video είναι δυνατό να γίνει σύγκριση του λόγου συμπίεσης που επιτυγχάνουν οι δύο υλοποιήσεις. Όπως φαίνεται στην επόμενη παράγραφο, οι λόγοι αυτοί είναι περίπου ίσοι κι οι όποιες μικρές αποκλίσεις αιτιολογούνται.

5.3 Αποτελέσματα των δοκιμών του συστήματος

Το σύστημα δοκιμάστηκε με επτά ακολουθίες video. Ορισμένες από τις ακολουθίες αυτές είναι πρότυπες και έχουν καθοριστεί από τη Διεθνή Ένωση Τηλεπικοινωνιών για την εκτίμηση της απόδοσης συστημάτων συμπίεσης video. Οι υπόλοιπες ακολουθίες δεν είναι πρότυπες και επιλέχτηκαν για συγκεκριμένους λόγους, οι οποίοι αναφέρονται για την καθεμία χωριστά. Στη συνέχεια παρατίθενται τα αποτελέσματα

των δοκιμών κάθε ακολουθίας video, όχι μόνο στη hardware, αλλά και στη software υλοποίηση του αλγορίθμου SCAN.

5.3.1 Δοκιμές με πρότυπες ακολουθίες video

Η πρώτη πρότυπη ακολουθία video με την οποία δοκιμάστηκε το σύστημα είναι η ακολουθία «Claire». Πρόκειται για μία ακολουθία στην οποία εμφανίζεται μια παρουσιάστρια σε ομοιόμορφο φόντο. Η κίνηση που παρουσιάζει το video είναι ελάχιστη και περιορίζεται κυρίως στα μάτια και τα χείλη, οπότε η συμπίεσή του αναμένεται να είναι μεγάλη. Τα καρέ είναι ανάλυσης QCIF (Quadrature Common Intermediate Format), όπως και όλα τα καρέ των πρότυπων ακολουθιών video, δηλαδή αποτελούνται από 144 γραμμές των 176 εικονοστοιχείων. Για να εισαχθούν τα καρέ αυτά στο σύστημα αφαιρέθηκαν 32 εικονοστοιχεία από την κάθε γραμμή τους και τα 144 × 144 καρέ που προέκυψαν υποβιβάστηκαν σε ανάλυση 128 × 128 εικονοστοιχείων. Συνολικά συμπιέστηκαν 16 καρέ, τα 6 πρώτα από τα οποία παρουσιάζονται στο Σχήμα 5.1.

Η δεύτερη πρότυπη ακολουθία είναι η «Miss America». Πρόκειται για μία ακολουθία που μοιάζει με την «Claire». Η κίνηση που παρουσιάζει ολόκληρη η ακολουθία είναι

Σχήμα 5.1 Τα 6 πρώτα καρέ της πρότυπης ακολουθίας video «Claire» από τα 16 που συμπιέστηκαν συνολικά

λίγο μεγαλύτερη, αλλά τα συγκεκριμένα καρέ που επιλέχτηκαν δεν παρουσιάζουν ιδιαίτερη κίνηση, οπότε αναμένεται η συμπίεση που επιτυγχάνεται να είναι συγκρίσιμη με αυτή της «Claire». Τα 6 πρώτα καρέ από τα 16 που συμπιέστηκαν φαίνονται στο Σχήμα 5.2.

Η τρίτη και τελευταία από τις πρότυπες ακολουθίες video, η «Trevor», δείχνει επίσης έναν εκφωνητή σε ομοιόμορφο φόντο. Στην περίπτωση αυτή όμως, η κίνηση δεν περιορίζεται μόνο στα χείλη και τα μάτια του εκφωνητή, εφόσον ο εκφωνητής κουνάει ολόκληρο το κεφάλι και τα χέρια, οπότε η συμπίεση αναμένεται να είναι μικρότερη από τις δύο προηγούμενες ακολουθίες. Τα καρέ που εισήχθησαν στο σύστημα προς συμπίεση είναι και στην περίπτωση αυτή 16, τα 6 πρώτα από τα οποία παρουσιάζονται στο Σχήμα 5.3.

Τα αποτελέσματα της συμπίεσης και για τις τρεις πρότυπες ακολουθίες video, που όπως προαναφέρθηκε αφορούν τόσο στη hardware, όσο και στη software υλοποίηση του αλγορίθμου SCAN για διάφορες τιμές κατωφλίου, παρουσιάζονται στον Πίνακα 5.1. Το ποσοστό συμπίεσης υπολογίζεται ως [(Αρχικό Μέγεθος – Τελικό Μέγεθος) / Αρχικό Μέγεθος] × 100%. Από τις τιμές του πίνακα αυτού φαίνεται ότι επιτυγχάνονται υψηλές τιμές συμπίεσης ακόμα και για μικρές τιμές κατωφλίου. Λόγω των μεγάλων ομοιοτήτων των συνεχόμενων καρέ, που οφείλονται κυρίως στο ομοιόμορφο φόντο που υπάρχει και στις τρεις ακολουθίες, αξιοσημείωτη είναι και η

Σχήμα 5.2 Τα 6 πρώτα καρέ της πρότυπης ακολουθίας video «Miss America» από τα 16 που συμπιέστηκαν συνολικά

Σχήμα 5.3 Τα 6 πρώτα καρέ της πρότυπης ακολουθίας video «Trevor» από τα 16 που συμπιέστηκαν συνολικά

Τιμή κατωφλίου	% Συμπίεση					
	Claire		Miss America		Trevor	
	Software	Hardware	Software	Hardware	Software	Hardware
0	34,68	34,51	31,46	31,32	19,74	19,62
1	79,30	79,15	74,16	74,00	52,34	52,24
2	82,65	82,61	80,62	80,59	57,68	57,62
3	84,68	84,59	84,44	84,36	61,63	61,50
4	86,53	86,41	87,33	87,24	65,29	65,16
5	87,93	87,78	89,61	89,49	68,51	68,52
7	89,95	89,87	92,69	92,60	73,47	73,32
10	92,03	91,89	95,26	95,11	78,53	78,39
15	94,05	93,91	97,29	97,15	84,04	83,89

Πίνακας 5.1 Ποσοστό συμπίεσης της hardware και της software υλοποίησης στις πρότυπες ακολουθίες video «Claire», «Miss America» και «Trevor» για διάφορες τιμές κατωφλίου

συμπίεση που επιτυγχάνεται και με μηδενική τιμή κατωφλίου, η οποία είναι μη απωλεστική. Αξίζει όμως να σημειωθεί ότι για μικρές τιμές κατωφλίου οι απώλειες στην ποιότητα είναι ελάχιστες, ενώ ακόμα και για τιμή κατωφλίου 15 η ποιότητα του

ανακατασκευασμένου video είναι αρκετά καλή. Στο σημείο αυτό αναφέρεται ότι η αποσυμπίεση σε hardware δεν έχει υλοποιηθεί πλήρως, οπότε οι όποιες εκτιμήσεις για την ποιότητα του αποσυμπιεσμένου video γίνονται συμπιέζοντας και αποσυμπιέζοντάς το με τη βοήθεια της software υλοποίησης του αλγορίθμου. Οι ακολουθίες «Claire» και «Miss America» συμπιέζονται περίπου το ίδιο για τιμές κατωφλίου έως 5, ενώ για μεγαλύτερες τιμές η «Miss America» συμπιέζεται περισσότερο, πιθανά λόγω της μικρής διαφοράς στη φωτεινότητα των εικονοστοιχείων των ρούχων και των μαλλιών με αυτά του φόντου. Η ακολουθία «Trevor» συμπιέζεται λιγότερο, όπως ήταν αναμενόμενο, εξαιτίας της λίγο μεγαλύτερης κίνησης που παρουσιάζει σε σχέση με τις άλλες δύο ακολουθίες.

Τα αποτελέσματα της συμπίεσης σε hardware είναι περίπου ίδια με αυτά του software, γεγονός που δείχνει τη σωστή λειτουργία του συστήματος που υλοποιήθηκε. Για την ακρίβεια, το σύστημα συμπιέζει ελάχιστα χειρότερα (περίπου 0,1%) από τη software υλοποίηση. Αυτό είναι αναμενόμενο και οφείλεται στη μη πλήρη χρησιμοποίηση ελάχιστων θέσεων της τελικής μνήμης, όπως έχει περιγραφεί στο τέλος του προηγούμενου κεφαλαίου, με αποτέλεσμα λίγο μεγαλύτερο μέγεθος συμπιεσμένου αρχείου. Αντίθετα, η software υλοποίηση εκμεταλλεύεται πλήρως το χώρο που καταλαμβάνουν τα συμπιεσμένα δεδομένα και επιτυγχάνει τη μεγαλύτερη δυνατή συμπίεση.

Διευκρινίζεται τέλος ότι οι όποιες διαφοροποιήσεις παρατηρούνται στα ποσοστά συμπίεσης των ακολουθιών video «Claire» και «Trevor» με αυτά που έχουν παρουσιαστεί στον Πίνακα 3.1 οφείλονται στη διαφορά ανάλυσης των καρέ, καθώς και στο ποια και πόσα καρέ της κάθε ακολουθίας έχουν χρησιμοποιηθεί.

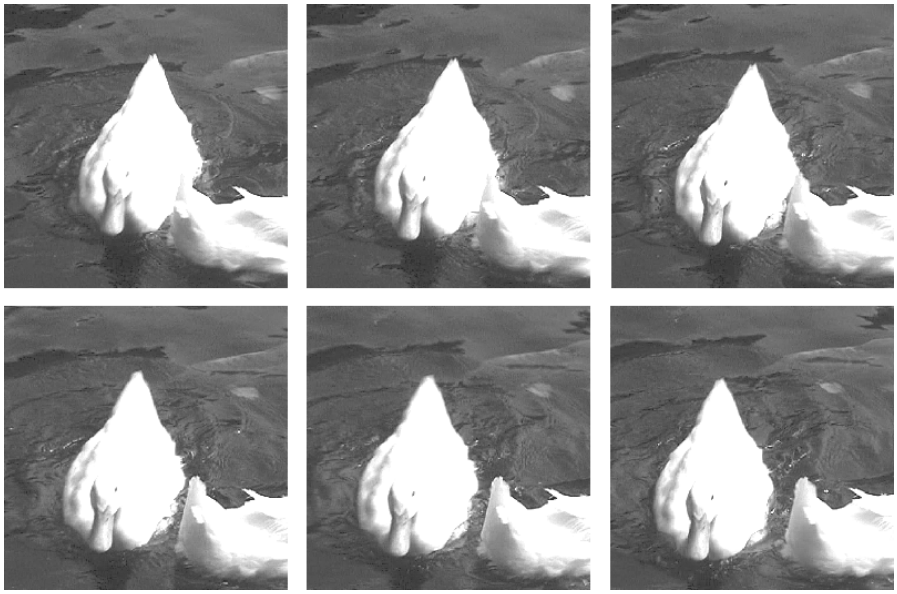
5.3.2 Δοκιμές με μη πρότυπες ακολουθίες video

Για τη διαμόρφωση σφαιρικότερης εικόνας σχετικά με την απόδοση του συστήματος, κρίθηκε σκόπιμο να γίνουν δοκιμές και με ορισμένες μη πρότυπες ακολουθίες video, ώστε να αξιολογηθεί η απόδοση του συστήματος και με video που περιέχουν έντονη κίνηση. Επίσης, η ανάλυση των πρότυπων ακολουθιών video δεν επέτρεπε τη δοκιμή του συστήματος στις αναλύσεις 256 × 256 και 512 × 512, κάτι που έγινε με μία από

τις μη πρότυπες ακολουθίες video. Μετά τις δοκιμές που πραγματοποιήθηκαν με τις πρότυπες ακολουθίες παρατηρήθηκε ότι από το έκτο καρέ και μετά το ποσοστό συμπίεσης άλλαζε ελάχιστα, οπότε οι μη πρότυπες ακολουθίες video επιλέχτηκε να αποτελούνται από 6 καρέ η καθεμία.

Η πρώτη μη πρότυπη ακολουθία με την οποία δοκιμάστηκε το σύστημα είναι η «Ducks», η οποία παρουσιάζεται στο Σχήμα 5.4. Πρόκειται για δύο πάπιες που κινούνται στο νερό. Ίσως με την πρώτη ματιά να φαίνεται ότι δεν έχει ιδιαίτερη κίνηση, αλλά το νερό που καλύπτει όλη την επιφάνεια των καρέ κινείται συνεχώς, σε αντίθεση με το ομοιόμορφο φόντο των πρότυπων ακολουθιών. Η συμπίεση της ακολουθίας αυτής αναμένεται να είναι αρκετά μικρότερη από αυτή των πρότυπων ακολουθιών. Το αρχικό video ήταν ανάλυσης 720 × 540, από το οποίο κρατήθηκε ένα 512 × 512 τμήμα του. Στη συνέχεια αυτό το τμήμα υποβιβάστηκε σε αναλύσεις 256 × 256 και 128 × 128, ώστε να δοκιμαστεί το σύστημα και στις τρεις αναλύσεις καρέ που υποστηρίζει. Τα αποτελέσματα των δοκιμών παρουσιάζονται στον Πίνακα 5.2.

Στον πίνακα αυτό για πρώτη φορά παρατηρείται ότι το σύστημα συμπιέζει αρνητικά (δηλαδή το συμπιεσμένο αρχείο είναι μεγαλύτερο από το αρχικό) για τις πολύ μικρές τιμές κατωφλίου. Αυτό δείχνει πόσο μεγάλες είναι οι διαφορές μεταξύ των καρέ και φανερώνει τους περιορισμούς του αλγορίθμου. Για μεγαλύτερες τιμές κατωφλίου η συμπίεση αυξάνει αρκετά, αλλά σε καμία περίπτωση δεν πλησιάζει τις τιμές που



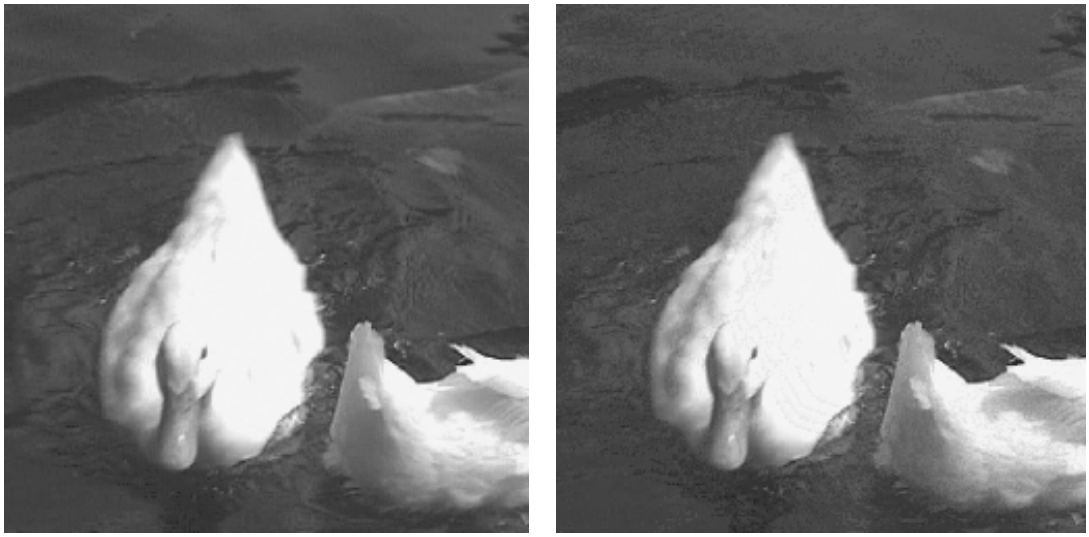
Σχήμα 5.4 Τα 6 καρέ της μη πρότυπης ακολουθίας video «Ducks»

Τιμή κατωφλίου	% Συμπίεση					
	128 × 128		256 × 256		512 × 512	
	Software	Hardware	Software	Hardware	Software	Hardware
0	< 0	< 0	< 0	< 0	< 0	< 0
1	< 0	< 0	0	< 0	0,24	0,11
2	8,42	8,29	9,09	8,97	9,36	9,26
3	15,40	15,26	16,25	16,14	16,49	16,38
4	21,32	21,20	22,18	22,09	22,41	22,29
5	26,31	26,23	27,17	27,22	27,56	27,47
7	34,47	34,38	35,73	35,60	36,31	36,27
10	43,91	43,76	45,41	45,27	46,12	46,00
15	56,04	55,88	58,39	58,24	59,91	59,78

Πίνακας 5.2 Ποσοστό συμπίεσης της hardware και της software υλοποίησης στη μη πρότυπη ακολουθία video «Ducks» για διάφορες τιμές κατωφλίου και για τις τρεις αναλύσεις που υποστηρίζονται από τη hardware υλοποίηση

επιτυγχάνονται στις πρότυπες ακολουθίες video. Επίσης, παρατηρείται ότι το ποσοστό της συμπίεσης αυξάνει ελαφρά όσο αυξάνεται η ανάλυση, γεγονός που αφορά μόνο τις δοκιμές αυτές, καθώς σε άλλες το ποσοστό συμπίεσης είναι πιθανό και να μειώνεται. Συγκρίνοντας τις τιμές που προκύπτουν από το σύστημα που υλοποιήθηκε με αυτές του software, προκύπτει ότι αυτό δουλεύει σωστά και στις δύο μεγαλύτερες αναλύσεις.

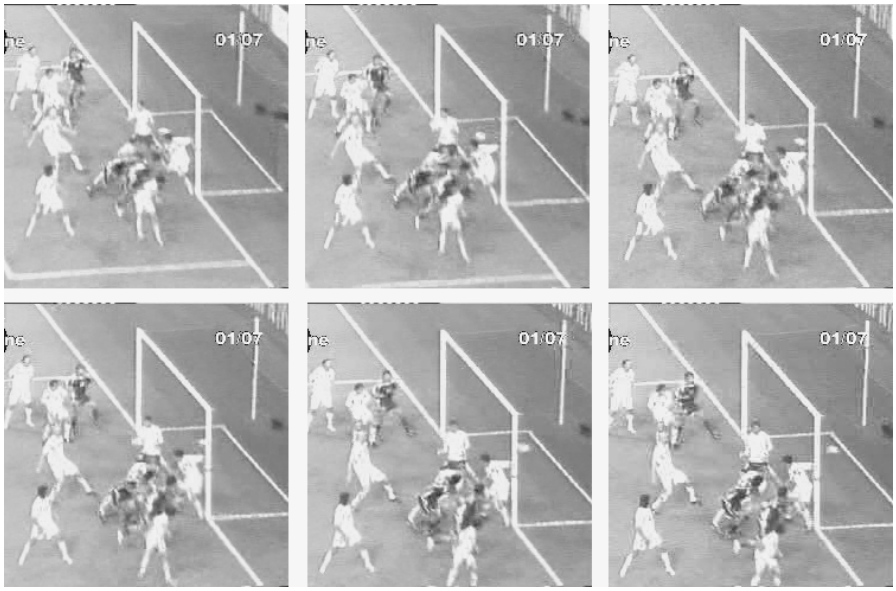
Αν και το σύστημα αποσυμπίεσης δεν έχει υλοποιηθεί πλήρως σε hardware, κρίνεται σκόπιμο να δειχτεί ένα αποσυμπιεσμένο καρέ από τη software υλοποίηση, ώστε να υπάρχει μια καλύτερη εικόνα των αποτελεσμάτων που μπορεί να επιτύχει ο αλγόριθμος. Αυτό έγινε στην παρούσα ακολουθία video λόγω της μεγαλύτερης ανάλυσης των καρέ της. Στο Σχήμα 5.5 παρουσιάζεται ένα ασυμπίεστο κι ένα αποσυμπιεσμένο καρέ ανάλυσης 256 × 256 με τιμή κατωφλίου 15. Η πτώση της ποιότητας είναι αισθητή στη δεύτερη εικόνα. Αυτό φαίνεται κυρίως από τις απότομες διαβαθμίσεις του τόνου του γκρι στην επιφάνεια της θάλασσας και στα φτερά της πρώτης πάπιας. Η τιμή του κατωφλίου είναι όμως αρκετά μεγάλη και η εικόνα έχει υποβαθμιστεί σε αποδεκτά επίπεδα. Πάντως, σε περίπτωση που απαιτείται



Σχήμα 5.5 Ένα αυθεντικό καρέ ανάλυσης 256 × 256 από την ακολουθία video «Ducks» (αριστερά) και το ίδιο καρέ αποσυμπιεσμένο (δεξιά) για τιμή κατωφλίου ίση με 15

μεγαλύτερη συμπίεση, η ποιότητα της αποσυμπιεσμένης εικόνας αφήνει περιθώρια και για περαιτέρω αύξηση του κατωφλίου.

Η δεύτερη μη πρότυπη ακολουθία, η «Soccer», αποτελείται από 6 καρέ που απεικονίζουν μια φάση ενός ποδοσφαιρικού αγώνα και φαίνεται στο Σχήμα 5.6. Είναι μια ακολουθία με έντονη κίνηση που αναμένεται να μη συμπιεστεί πολύ, έχει όμως ενδιαφέρον η σύγκριση των αποτελεσμάτων με αυτά της προηγούμενης ακολουθίας, διότι παρουσιάζει πιο έντονη κίνηση από αυτή, αλλά το φόντο της (το χορτάρι του γηπέδου) είναι σχεδόν ομοιόμορφο.



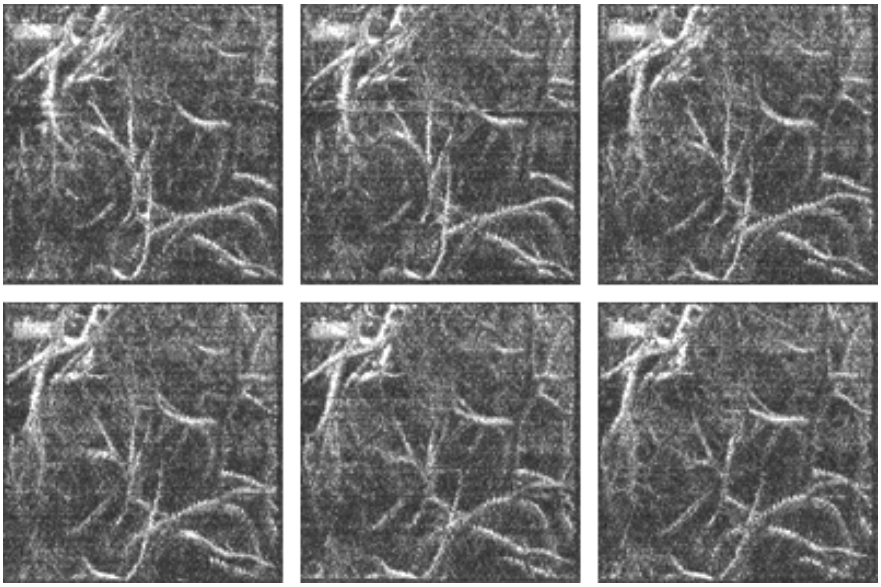
Σχήμα 5.6 Τα 6 καρέ της μη πρότυπης ακολουθίας video «Soccer»

Η τρίτη μη πρότυπη ακολουθία video, η «News», αποτελείται και αυτή από 6 καρέ που παρουσιάζονται στο Σχήμα 5.7. Απεικονίζει ένα παρουσιαστή και η κίνηση που παρουσιάζει είναι ελάχιστη, ενώ το φόντο είναι σταθερό. Αυτή η ακολουθία δοκιμάστηκε με σκοπό να γίνει σύγκριση με τις πρότυπες ακολουθίες, όπου δεν παρουσιάζεται ιδιαίτερη κίνηση.

Η τέταρτη ακολουθία επιλέχτηκε με σκοπό να δοκιμάσει τη συμπεριφορά του αλγορίθμου SCAN και κατ' επέκταση του συστήματος που υλοποιήθηκε σε ακραίες συνθήκες. Παρουσιάζει κάποια φυτά στη θάλασσα που κινούνται συνεχώς, ενώ το κάθε καρέ εμπεριέχει πολύ μεγάλο ποσοστό θορύβου, τόσο μεγάλο που η παρακολούθηση του video αυτού είναι πολύ κουραστική. Η συμπίεση που αναμένεται να επιτευχθεί με μία τέτοια ακολουθία video είναι σίγουρα πολύ μικρή. Όση και να είναι αυτή δεν έχει πρακτική σημασία, απλά λόγω των ελάχιστων ομοιοτήτων που παρουσιάζονται μεταξύ των καρέ έχει νόημα να είναι γνωστό πως συμπεριφέρεται το σύστημα σε πραγματικά δύσκολες συνθήκες. Η ακολουθία αυτή, η «Noisy» παρουσιάζεται στο Σχήμα 5.8.

Ο Πίνακας 5.3 παρουσιάζει τα αποτελέσματα της συμπίεσης για τις τρεις τελευταίες μη πρότυπες ακολουθίες. Η συμπίεση που επιτυγχάνεται στην ακολουθία «Soccer» είναι αξιοσημείωτα υψηλή, ειδικά συγκρινόμενη με αυτή της ακολουθίας «Ducks», παρά την εντονότερη κίνηση που παρουσιάζει από αυτή. Φαίνεται καθαρά λοιπόν η

Σχήμα 5.7 Τα 6 καρέ της μη πρότυπης ακολουθίας video «News»



Σχήμα 5.8 Τα 6 καρέ της μη πρότυπης ακολουθίας video «Noisy»

Τιμή κατωφλίου	% Συμπίεση					
	Soccer		News		Noisy	
	Software	Hardware	Software	Hardware	Software	Hardware
0	< 0	< 0	32,48	32,37	< 0	< 0
1	5,66	5,53	64,91	64,79	< 0	< 0
2	16,81	16,70	79,24	79,16	< 0	< 0
3	25,32	25,23	86,76	86,69	< 0	< 0
4	32,11	31,96	90,07	89,92	< 0	< 0
5	37,54	37,42	91,81	91,73	< 0	< 0
7	46,00	45,90	93,98	93,84	1,77	1,64
10	54,79	54,66	95,88	95,78	9,99	9,90
15	63,87	63,78	97,33	97,21	22,48	22,36

Πίνακας 5.3 Ποσοστό συμπίεσης της hardware και της software υλοποίησης στις μη πρότυπες ακολουθίες video «Soccer», «News» και «Noisy» για διάφορες τιμές κατωφλίου

σημασία του ομοιόμορφου φόντου στην επίτευξη υψηλού βαθμού συμπίεσης. Η συμπίεση της ακολουθίας «News» είναι αναμενόμενα υψηλή, συγκρίσιμη με τις τρεις πρότυπες ακολουθίες video. Παρόλο που στην περίπτωση αυτή το φόντο δεν ήταν ομοιόμορφο, παρέμενε σταθερό, δρώντας ευεργετικά στην επίτευξη μεγάλης

συμπίεσης. Όσον αφορά την ακολουθία «Noisy», συμπιέζεται ελάχιστα και μόνο με υψηλές τιμές κατωφλίου. Αυτό οφείλεται στην έντονη κίνηση, αλλά κυρίως στο θόρυβο, που μεταβάλλει με τυχαίο τρόπο τις τιμές φωτεινότητας όλων των εικονοστοιχείων των καρέ. Υπενθυμίζεται ότι η ακολουθία αυτή δοκιμάστηκε μόνο για να φανεί η συμπεριφορά του συστήματος σε μία πολύ δύσκολη περίπτωση εισόδου, ενώ στην πράξη video τέτοιας ποιότητας είναι μάλλον απίθανο να εισαχθεί στο σύστημα.

5.4 Απεικόνιση της σχεδίασης σε αναδιατασσόμενη λογική

Η αρχιτεκτονική που έχει περιγραφεί στο κεφάλαιο 4 υλοποιήθηκε για την οικογένεια FPGAs Virtex2 της εταιρίας Xilinx και για το λόγο αυτό χρησιμοποιήθηκε η πλατφόρμα ISE 4.2 της ίδιας εταιρίας. Όλα τα αρχεία γράφτηκαν σε γλώσσα VHDL και για τη σύνθεση του κώδικα χρησιμοποιήθηκε το Synplify Pro 7.3.1 Οι προσομοιώσεις έγιναν με τη βοήθεια του ModelSim 5.7f.

Η μικρότερη FPGA της σειράς Virtex2 στην οποία χωράει η σχεδίαση είναι η Virtex XC2V500, της οποίας καταλαμβάνει το 88% του διαθέσιμου χώρου και το 62,5% των διαθέσιμων block μνήμης. Πιο συγκεκριμένα, καταλαμβάνει 2.715 από τα 3.072 διαθέσιμα slices και τα 20 από τα 32 διαθέσιμα block μνήμης που έχουν μέγεθος 18 Kbits το καθένα. Η κατάληψη του χώρου της FPGA από το κάθε υποσύστημα της σχεδίασης παρουσιάζεται αναλυτικά στον Πίνακα 5.4 και η κατάληψη των block μνήμης από τις μνήμες της σχεδίασης παρουσιάζεται στον Πίνακα 5.5.

Η αμέσως μικρότερη FPGA, η Virtex XC2V250, διαθέτει αρκετά block μνήμης (24 συνολικά), ώστε να καλύψει τις απαιτήσεις του συστήματος σε μνήμη, αλλά τα slices που διαθέτει είναι μόλις 1.536, πολύ λιγότερα απ’ όσα απαιτεί το σύστημα. Σημειώνεται ότι τα slices είναι οι βασικές δομικές μονάδες των FPGAs των οικογενειών Vitex και Virtex2. Τέσσερα slices αποτελούν ένα CLB, τη βασική δομική μονάδα των παλαιότερων FPGAs.

Σχετικά με την συχνότητα στην οποία μπορεί να λειτουργήσει το σύστημα, αυτή υπολογίστηκε στα 76,5 MHz από το εργαλείο ISE. Η διαδικασία τοποθέτησης και δρομολόγησης (place and route) βελτιστοποιήθηκε για την επίτευξη όσο το δυνατόν μεγαλύτερης ταχύτητας, ενώ η όλη διαδικασία έγινε αυτόματα. Το πιο αργό υποσύ-

Υποσύστημα	**Slices της XC2V500**	**% Κατάληψη της σχεδίασης**	**% Κατάληψη της XC2V500**
Μνήμης SDRAM	344	12,67	11,20
Διαχωρισμού παραθύρων	78	2,87	2,54
Σύγκρισης παραθύρων	313 × 4	46,11	40,76
Κωδικοποίησης παραθύρων	132 × 4	19,45	17,19
Μετατροπής δεδομένων σε 32 bits	72 × 4	10,61	9,37
Παραγωγής τελικών αποτελεσμάτων	225	8,29	7,32
Σύνολο	**2.715**	**100**	**88,38**

Πίνακας 5.4 Κατανομή του χώρου της FPGA Virtex XC2V500 στη σχεδίαση

Μονάδα	**Block RAMs της XC2V500**	**% Κατάληψη των block RAMs της XC2V500**
Ενδιάμεση RAM	2 × 4	25,00
$2^{η}$ Ενδιάμεση RAM	3 × 4	37,50
Σύνολο	**20**	**62,50**

Πίνακας 5.5 Κατανομή των block μνήμης της FPGA Virtex XC2V500 στη σχεδίαση

στημα αποδείχτηκε αυτό της σύγκρισης των παραθύρων, ενώ η μέγιστη συχνότητα στην οποία μπορεί θεωρητικά να λειτουργήσει ο ελεγκτής της μνήμης, δηλαδή τα 125 MHz δεν επιτευχθηκε. Τονίζεται ότι η συχνότητα του ρολογιού του συστήματος ανάρχεται στα 76,5/2 = 38,25 MHz. Η μνήμη SDRAM λειτουργεί στα 76,5 MHz, συχνότητα στην οποία λειτουργούν ορισμένα μόνο υποσυστήματα του συστήματος.

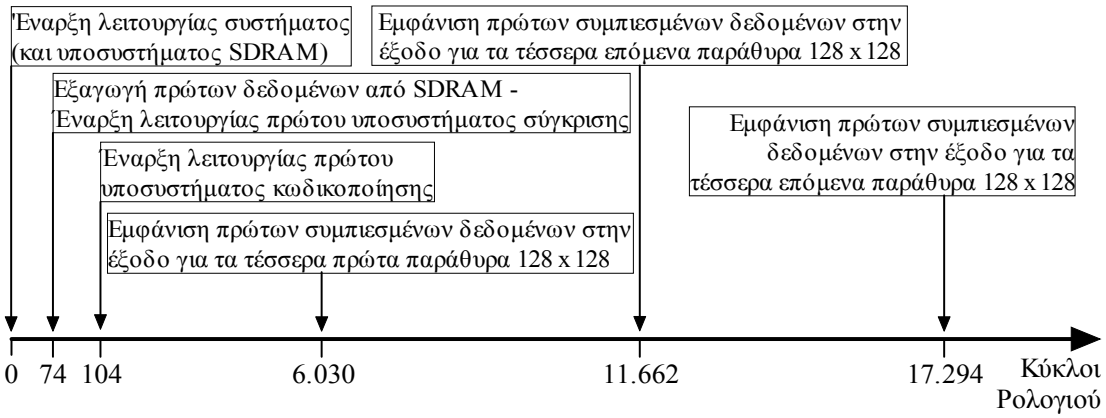
Τα δύο block μνήμης που χρησιμοποιεί κάθε ενδιάμεση RAM καλύπτονται πλήρως, ενώ το ένα από τα τρία που χρησιμοποιεί κάθε $2^{η}$ ενδιάμεση RAM χρησιμοποιείται μόνο κατά ένα μικρό ποσοστό του. Οι FIFO του συστήματος δεν παρουσιάζονται καθόλου στον Πίνακα 5.5, διότι δε χρησιμοποιούν block μνήμης της FPGA, αλλά τμήμα της κατανεμημένης μνήμης της, η οποία κατανέμεται ομοιόμορφα στα slices της FPGA.

5.5 Εκτίμηση της απόδοσης του συστήματος

Έχοντας ως δεδομένο ότι το σύστημα μπορεί να λειτουργήσει στα 76,5 MHz, τιμή που υπολογίστηκε από το εργαλείο ISE, μπορούν να γίνουν κάποιοι υπολογισμοί σχετικά με το throughput του συστήματος. Διευκρινίζεται ότι η συχνότητα αυτή ουσιαστικά είναι η συχνότητα της SDRAM, αλλά αρκετά τμήματα του συστήματος λειτουργούν σε αυτή. Η συχνότητα του συστήματος είναι η μισή από αυτή, δηλαδή 38,25 MHz.

Αρχικά είναι αναγκαίο να γίνει αναφορά στο πόσους κύκλους χρειάζεται το σύστημα για να επεξεργαστεί κάθε καρέ. Από τη στιγμή που το σύστημα τίθεται σε λειτουργία, γίνεται η απαραίτητη αρχικοποίηση της SDRAM (η αρχικοποίηση αυτή δεν αφορά την εισαγωγή των δεδομένων, αλλά κάποιες ενέργειες που πρέπει να γίνουν, ώστε να μπορεί να δεχτεί δεδομένα) και τα πρώτα δεδομένα είναι διαθέσιμα στην είσοδο του πρώτου υποσυστήματος σύγκρισης μετά από 74 κύκλους ρολογιού του συστήματος (του ρολογιού των 38,25 MHz), ενώ 11 κύκλους αργότερα είναι διαθέσιμα δεδομένα και στο δεύτερο υποσύστημα σύγκρισης, κ.ο.κ. Μετά από 30 κύκλους αρχίζει να λειτουργεί το πρώτο υποσύστημα κωδικοποίησης. Τα δεδομένα που εξάγονται από το υποσύστημα αυτό οδηγούνται στα δύο τελευταία υποσυστήματα και τα συμπιεσμένα δεδομένα είναι διαθέσιμα στην έξοδο στον κύκλο 6.030. Εν τω μεταξύ, όλα τα υποσυστήματα τροφοδοτούνται πλέον συνεχώς με νέα δεδομένα, οπότε κάθε 5632 κύκλους ρολογιού αρχίζει η εμφάνιση στην έξοδο των συμπιεσμένων δεδομένων για ένα νέο παράθυρο 128 × 128. Φυσικά, αν συμπιέζεται video ανάλυσης 128 × 128, κάθε 5.632 κύκλους έχει συμπιεστεί ένα ολόκληρο καρέ. Η διαδικασία αυτή συνοψίζεται στο Σχήμα 5.9.

Σύμφωνα με τα παραπάνω, θεωρείται ότι για την επεξεργασία ενός παραθύρου 128 × 128 το σύστημα χρειάζεται 5.632 κύκλους. Αφού η περίοδος του ρολογιού του συστήματος είναι $1/38{,}25 \times 10^9 \approx 26$ ns, οι 400 περίπου επιπλέον κύκλοι που απαιτούνται για την παραγωγή των πρώτων αποτελεσμάτων αντιστοιχούν σε χρόνο ίσο με 10,4 μs, που μπορεί να θεωρηθεί αμελητέος. Στον Πίνακα 5.6 παρατίθενται οι κύκλοι ρολογιού που απαιτούνται για τη συμπίεση ενός καρέ στις υποστηριζόμενες από το σύστημα αναλύσεις.



Σχήμα 5.9 Κύκλοι ρολογιού του συστήματος που απαιτούνται για να γίνουν συγκεκριμένες λειτουργίες

Ανάλυση καρέ	**Απαιτούμενοι κύκλοι ρολογιού για την επεξεργασία ενός καρέ**
128 × 128	5.632
256 × 256	22.528
512 × 512	90.112

Πίνακας 5.6 Απαιτούμενοι κύκλοι ρολογιού για την επεξεργασία ενός καρέ κάθε ανάλυσης από το σύστημα

Το throughput που μπορεί να επιτύχει το σύστημα κυμαίνεται στα 111 MB/s. Η τιμή αυτή είναι αρκετά υψηλή και προκύπτει κυρίως λόγω του παραλληλισμού και της διασωλήνωσης της αρχιτεκτονικής, λόγω της χαμηλής πολυπλοκότητας των υπολογισμών που απαιτούνται από τον αλγόριθμο SCAN, αλλά και της χρησιμοποίησης μιας FPGA από τη σειρά Virtex2, η οποία προσφέρει αρκετή μνήμη και μπορεί να λειτουργήσει σε υψηλές συχνότητες. Ο Πίνακας 5.7 παρουσιάζει το throughput του συστήματος σε Mbytes/sec και σε καρέ ανά δευτερόλεπτο (fps) για διάφορες αναλύσεις των καρέ. Με δεδομένο ότι ένα τυπικό video αποτελείται από 15, 25 ή 30 fps, στον πίνακα αυτό δεν αναφέρονται μόνο οι υποστηριζόμενες από το σύστημα αναλύσεις, αλλά και κάποιες μεγαλύτερες που μπορούν να υποστηριχτούν με τροποποιήσεις στο υποσύστημα της μνήμης SDRAM, ώστε να εισέρχονται στα

Ανάλυση καρέ	Throughput	
	fps	MB/s
128 × 128	6791	111
256 × 256	1697	
512 × 512	424	
1024 × 1024	106	
2048 × 2048	26	

Πίνακας 5.7 Throughput του συστήματος σε fps για διάφορες αναλύσεις των καρέ

υποσυστήματα σύγκρισης των παραθύρων τα σωστά δεδομένα, και με τοποθέτηση μεγαλύτερων μνημών SRAM στα υποσυστήματα αυτά.

Αξίζει να σημειωθεί ότι, αφού λόγω της περιορισμένης ταχύτητας της SDRAM, όπως έχει προαναφερθεί, το σύστημα τροφοδοτείται με 8 τιμές δεδομένων ανά 11 κύκλους, ενώ χρειάζεται 10 κύκλους για να τις επεξεργαστεί, η τοποθέτηση μιας λίγο γρηγορότερης SDRAM είναι σε θέση να βελτιώσει το throughput του συστήματος κατά 9%. Αυτό προκύπτει από το γεγονός ότι από τους 5.632 κύκλους που χρειάζονται για τη συμπίεση ενός 128 × 128 παραθύρου το σύστημα κάνει υπολογισμούς για 5.120 κύκλους και για τους υπόλοιπους 512 αναμένει δεδομένα από την SDRAM.

Για καλύτερη απόδοση του συστήματος προκύπτει ότι η τοποθέτηση μιας γρηγορότερης μνήμης στη θέση της SDRAM με πλάτος λέξης 64-bit, για παράδειγμα μιας DDR-SDRAM (Double Data Rate SDRAM), μπορεί να το τροφοδοτεί συνεχώς με δεδομένα, ενώ λόγω του πλάτους των 64 bits υπάρχει χρόνος και για εγγραφή νέων δεδομένων σε αυτή, ώστε το σύστημα να μπορεί να επεξεργάζεται δεδομένα σε πραγματικό χρόνο.

5.6 Επίλογος

Στο παρόν κεφάλαιο αναλύθηκε η διαδικασία των προσομοιώσεων και των δοκιμών που έγιναν. Τα αποτελέσματα των δοκιμών δείχνουν την απόδοση του αλγορίθμου

για διάφορες ακολουθίες video και η σύγκρισή τους με τα αντίστοιχα της software υλοποίησης του αλγορίθμου πιστοποιούν τη σωστή λειτουργία του συστήματος. Επίσης, διαπιστώθηκε ότι η εκτιμώμενη ταχύτητα του συστήματος επιτρέπει τη συμπίεση σε πραγματικό χρόνο ακολουθιών video ακόμα και μεγαλύτερης ανάλυσης από τη μέγιστη που αυτό υποστηρίζει.

Κεφάλαιο 6

Συμπεράσματα και μελλοντικές επεκτάσεις

6.1 Συμπεράσματα

Στην παρούσα εργασία παρουσιάστηκε ο αλγόριθμος SCAN για συμπίεση video και μία αρχιτεκτονική για την αποδοτική υλοποίηση του αλγορίθμου αυτού σε αναδιατασσόμενη λογική. Τα αποτελέσματα της υλοποίησης δείχνουν ότι είναι δυνατό να συμπιεστούν σε πραγματικό χρόνο video υψηλής ανάλυσης, ακόμα μεγαλύτερης και από τις προδιαγραφές του συστήματος. Πιο συγκεκριμένα, ενώ υποστηρίζονται video μέγιστης ανάλυσης 512 × 512 εικονοστοιχείων × 8 bits / εικονοστοιχείο, μπορούν να συμπιεστούν video μέχρι και ανάλυσης 2048 × 2048 × 8 @ 26 fps. Σημαντικό ρόλο στο να επιτευχθεί η απόδοση αυτή παίζει η παράλληλη επεξεργασία τεσσάρων παραθύρων 64 × 64 εικονοστοιχείων, ενώ με χρήση μεγαλύτερης FPGA είναι δυνατό να επεξεργαστούν περισσότερα παράθυρα παράλληλα.

Η απόδοση του αλγορίθμου κρίνεται αρκετά ικανοποιητική. Η συμπίεση που επιτυγχάνεται είναι πολύ μεγάλη για video που δεν παρουσιάζουν ιδιαίτερη κίνηση (ακόμη και πάνω από 95% για τιμή κατωφλίου ίση με 10), ενώ μειώνεται για video που παρουσιάζουν κίνηση. Η ποιότητα του αποσυμπιεσμένου video είναι ικανοποιητική ακόμα και για τιμές κατωφλίου που ξεπερνούν το 10.

Η πλειονότητα των επιμέρους τμημάτων της αρχιτεκτονικής έχει υλοποιηθεί σε χωριστά κομμάτια κώδικα. Έτσι είναι καθορισμένες οι διεπαφές μεταξύ των υποσυστημάτων και των μονάδων που τις αποτελούν, καθιστώντας το σύστημα σχετικά εύκολα επεκτάσιμο.

6.2 Μελλοντικές επεκτάσεις

Η ολοκλήρωση του συστήματος απαιτεί την πλήρη υλοποίηση και ενσωμάτωση σε αυτό ενός συστήματος αποσυμπίεσης των συμπιεσμένων δεδομένων. Στο παράρτημα προτείνεται μια αρχιτεκτονική για την αποσυμπίεση.

Όπως έχει αναφερθεί, κύριος σκοπός του συστήματος συμπίεσης είναι να συμπιέζει τα δεδομένα που πρόκειται να κρυπτογραφηθούν, οπότε η ενοποίηση των συστημάτων συμπίεσης / αποσυμπίεσης με το ήδη υλοποιημένο σύστημα της κρυπτογράφησης / αποκρυπτογράφησης κρίνεται απαραίτητη.

Μετά την ενοποίηση με το σύστημα κρυπτογράφησης, μπορεί να υλοποιηθεί και το σύστημα για απόκρυψη πληροφορίας (information hiding – [29, 30]), για την υλοποίηση του οποίου μπορεί να γίνει χρησιμοποίηση ενός τμήματος του συστήματος κρυπτογράφησης. Η απόκρυψη πληροφορίας αφορά την απόκρυψη μικρών ποσοτήτων δεδομένων μέσα σε άλλα που είναι αρκετά περισσότερα σε ποσότητα. Για παράδειγμα, για να κρυφτεί μία μικρή φωτογραφία μέσα σε μια μεγαλύτερη αφαιρούνται από αυτή ορισμένα εικονοστοιχεία, την απουσία των οποίων δεν μπορεί να αντιληφθεί εύκολα το ανθρώπινο μάτι και τα εικονοστοιχεία αυτά χρησιμοποιούνται για την απόκρυψη της επιθυμητής πληροφορίας.

Όσον αφορά πιθανές βελτιώσεις στο σύστημα της συμπίεσης, η απόδοσή του μπορεί να βελτιωθεί με αντικατάσταση της υπάρχουσας SDRAM από γρηγορότερη και μεγαλύτερου πλάτους λέξης μνήμη, όπως μία 64-bit DDR-SDRAM, κάτι που έχει αναφερθεί και στο τέλος του προηγούμενου κεφαλαίου. Με τον τρόπο αυτό μπορεί να εκμεταλλευθεί πλήρως η ταχύτητα του συστήματος και λόγω του πλάτους των 64 bits υπάρχει χρόνος όχι μόνο για ανάγνωση, αλλά και για εγγραφή νέων δεδομένων σε αυτή, ώστε τα περιεχόμενα της μνήμης να μπορούν να ανανεώνονται και το σύστημα να μπορεί να επεξεργάζεται συνεχώς δεδομένα σε πραγματικό χρόνο και να μην περιορίζεται στα αποθηκευμένα δεδομένα της μνήμης. Επίσης, είναι δυνατό να αυξηθεί ο αριθμός των παράλληλων μονάδων επεξεργασίας, με τις αντίστοιχες αλλαγές φυσικά στον address generator της SDRAM και στα υποσυστήματα μετατροπής των δεδομένων σε 32 bits και παραγωγής τελικών αποτελεσμάτων.

Τέλος, ενδιαφέρον παρουσιάζει η επέκταση του αλγορίθμου SCAN και του συστήματος, ώστε να υποστηρίζουν έγχρωμα video. Κάτι τέτοιο είναι εφικτό, μόνο που τριπλασιάζεται ο απαιτούμενος χρόνος για συμπίεση video συγκεκριμένης ανάλυσης και τριπλασιάζονται οι απαιτήσεις του συστήματος σε μνήμη.

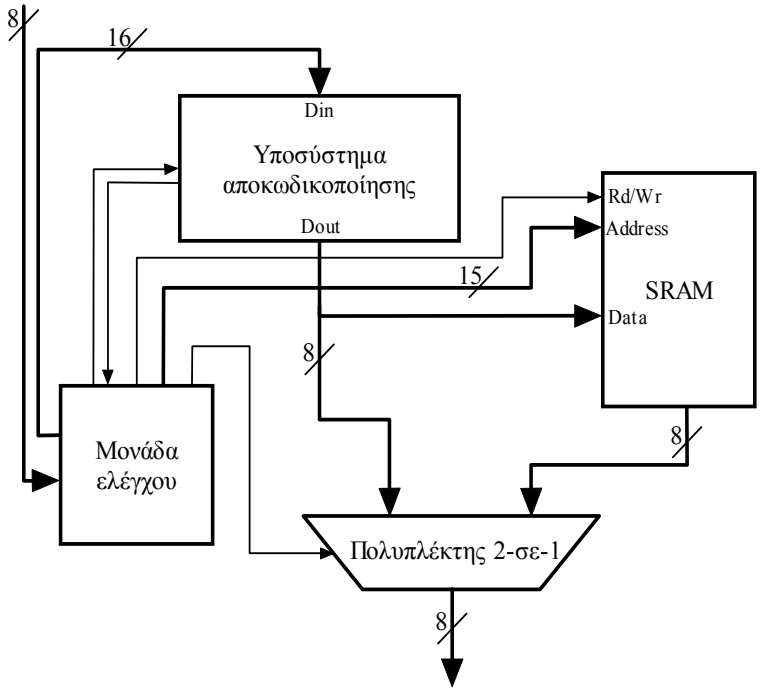
Παράρτημα

Προτεινόμενη αρχιτεκτονική για την αποσυμπίεση

Στο παρόν παράρτημα προτείνεται η αρχιτεκτονική του συστήματος αποσυμπίεσης, η οποία δεν υλοποιήθηκε στα πλαίσια της παρούσας εργασίας. Αξίζει να σημειωθεί ότι τα συμπιεσμένα δεδομένα πρέπει να έχουν πλάτος 8 bits. Το σύστημα της συμπίεσης όμως, όπως αυτό περιγράφηκε στο τέταρτο κεφάλαιο, παράγει συμπιεσμένα δεδομένα πλάτους 32 bits. Αυτό έγινε διότι, εφόσον το σύστημα της συμπίεσης δέχεται δεδομένα πλάτους 32 bits στην είσοδό του, κρίθηκε σωστό τα δεδομένα στην έξοδό του να είναι επίσης 32 bits. Παρ' όλ' αυτά, με μια απλή αφαίρεση από τη σχεδίαση των υποσυστημάτων μετατροπής των δεδομένων σε 32 bits και παραγωγής τελικών αποτελεσμάτων, ως έξοδος του συστήματος συμπίεσης μπορούν να θεωρηθούν οι τέσσερις 8-bit έξοδοι των υποσυστημάτων κωδικοποίησης. Η προσέγγιση αυτή κάνει τη σχεδίαση άμεσα συμβατή και με το ήδη υλοποιημένο σύστημα της κρυπτογράφησης, το οποίο επεξεργάζεται δεδομένα πλάτους 8 bits. Σε περίπτωση όμως που είναι δυνατό να γίνει κρυπτογράφηση σε δεδομένα πλάτους 32 bits, μπορεί η αρχιτεκτονική του συστήματος αποσυμπίεσης να επεκταθεί, ώστε να δέχεται 32 bits δεδομένων και να τα διασπά σε ομάδες των 8 bits, όπως γίνεται και στο σύστημα της συμπίεσης.

Π.1 Γενική εποπτεία της αρχιτεκτονικής

Το block διάγραμμα της προτεινόμενης αρχιτεκτονικής παρουσιάζεται στο Σχήμα Π.1. Αποτελείται από τη μονάδα αποκωδικοποίησης, μία μνήμη τύπου SRAM και μεγέθους 64 K × 8 bits, ένα πολυπλέκτη 2-σε-1 και μία μονάδα ελέγχου, που ρυθμίζει τη λειτουργία του συστήματος. Τα δεδομένα που αφορούν το πρώτο καρέ αποθηκεύονται στη μνήμη SRAM. Τα συμπιεσμένα δεδομένα εισάγονται στο υποσύστημα αποκωδικοποίησης. Για όσα εικονοστοιχεία δεν έχει κρατηθεί πληροφορία, στα αποσυμπιεσμένα δεδομένα οδηγείται η τιμή του αντίστοιχου εικονοστοιχείου του πρώτου παραθύρου. Αντίθετα, για όσα από αυτά έχει κρατηθεί πληροφορία, για όσα δηλαδή ήταν εκτός κατωφλίου κατά τη διαδικασία της



Σχήμα Π.1 Block διάγραμμα της προτεινόμενης αρχιτεκτονικής για το σύστημα της αποσυμπίεσης

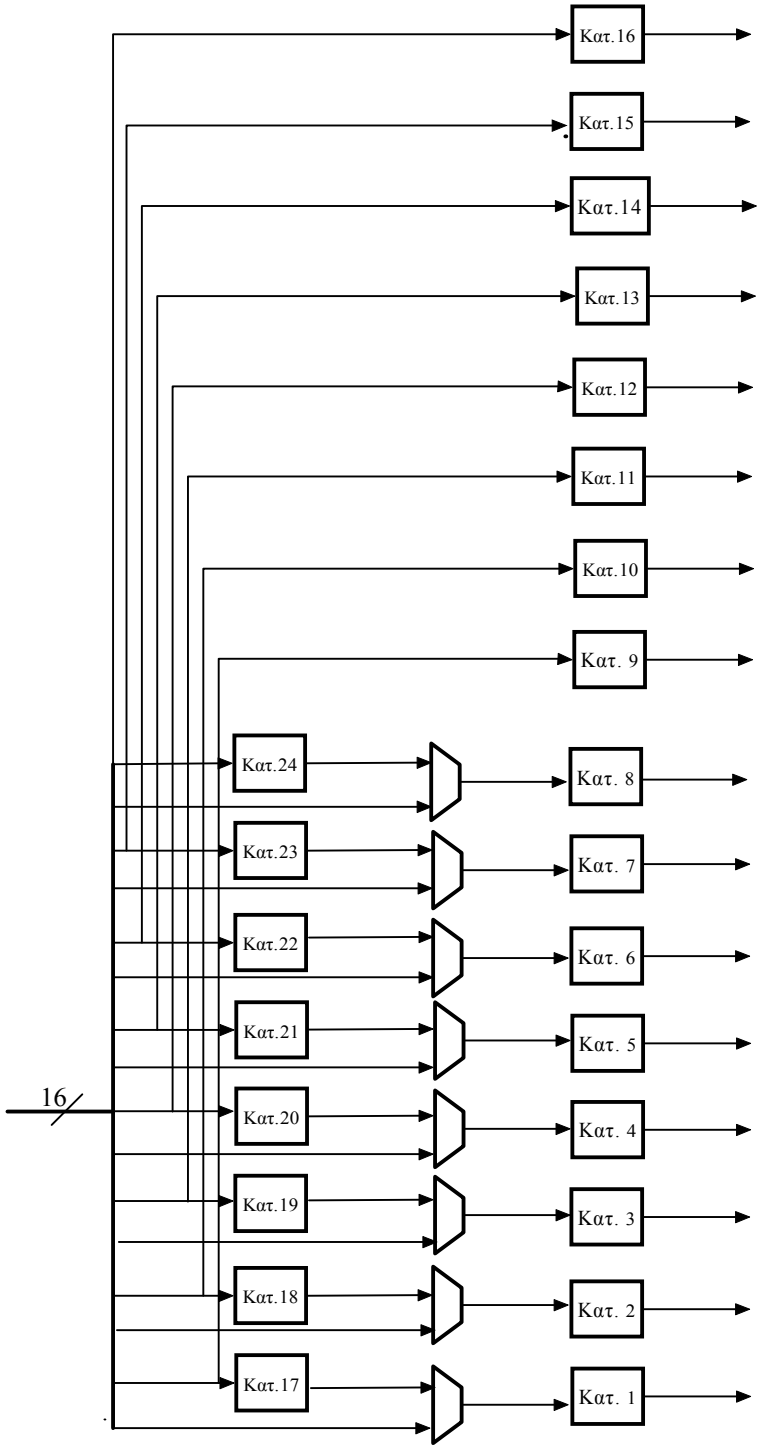
συμπίεσης, η τιμή τους οδηγείται τόσο στα αποσυμπιεσμένα δεδομένα, όσο και στην αντίστοιχη θέση της μνήμης SRAM, ώστε αυτή να περιέχει ενημερωμένα δεδομένα για τη σωστή αποσυμπίεση και των επόμενων παραθύρων.

Τονίζεται τέλος ότι το σύστημα της αποκωδικοποίησης αποτελείται από τέσσερα όμοια τμήματα, καθένα από τα οποία αντιστοιχεί σε μία 8-bit έξοδο του συστήματος συμπίεσης.

Π.2 Το υποσύστημα αποκωδικοποίησης των παραθύρων

Το υποσύστημα αποκωδικοποίησης παίζει τον κύριο ρόλο στη σωστή λειτουργία του συστήματος αποσυμπίεσης. Στην είσοδό του δέχεται τα συμπιεσμένα δεδομένα των 64 × 64 παραθύρων και ανάλογα με αυτά, είτε τροφοδοτεί με δεδομένα την έξοδο του συστήματος και την SRAM, είτε δίνει την κατάλληλη πληροφορία στη μονάδα ελέγχου, ώστε να προωθηθεί στην έξοδο η κατάλληλη τιμή της SRAM.

Ένα block διάγραμμα του υποσυστήματος παρουσιάζεται στο Σχήμα Π.2. Το υποσύστημα αυτό μοιάζει πολύ με τη διάταξη πολυπλεκτών και καταχωρητών του υποσυστήματος κωδικοποίησης των παραθύρων (Σχήμα 4.25). Τα συμπιεσμένα



Σχήμα Π.2 Block διάγραμμα του υποσυστήματος αποκωδικοποίησης των παραθύρων

δεδομένα εισάγονται στη μονάδα ελέγχου, η οποία τα μετατρέπει κατάλληλα ώστε να εισαχθούν στο υποσύστημα. Η μετατροπή αυτή είναι εντελώς ανάλογη με αυτή που γίνεται στη μονάδα πολυπλεξίας του υποσυστήματος κωδικοποίησης των παραθύρων. Η λειτουργία της διάταξης των καταχωρητών διαφοροποιείται αυτή του υποσυστήματος κωδικοποίησης των παραθύρων μόνο στο ότι τα δεδομένα των

καταχωρητών 17 – 24 μεταφέρονται και στους καταχωρητές 9 – 15. Ο σκοπός που εξυπηρετεί αυτή η διαφοροποίηση αναλύεται στη συνέχεια.

Τα πρώτα 8 bits δεδομένων αποθηκεύονται στους καταχωρητές 1 – 8. Αυτά αντιστοιχούν στον αριθμό των κωδικοποιημένων 4 × 4 παραθύρων που υπάρχουν στο τρέχον παράθυρο. Αν ο αριθμός των κωδικοποιημένων παραθύρων είναι ίσος με το μηδέν σημαίνει ότι στο τρέχον παράθυρο δεν έχει κωδικοποιηθεί κανένα 4 × 4 παράθυρο, οπότε στην έξοδο πρέπει να οδηγηθούν αυτούσια τα περιεχόμενα της SRAM που αφορούν το παράθυρο αυτό. Σε διαφορετική περίπτωση, για καθένα από τα κωδικοποιημένα παράθυρα εκτελείται η ακόλουθη διαδικασία: Αρχικά, στους καταχωρητές 1 – 8 αποθηκεύονται οι συντεταγμένες του 4 × 4 παραθύρου στο 64 × 64 παράθυρο. Τότε οδηγούνται στην έξοδο του συστήματος τα περιεχόμενα της SRAM που αντιστοιχούν στα εικονοστοιχεία του παραθύρου που βρίσκονται πριν το πρώτο του κωδικοποιημένο παράθυρο. Για τα επόμενα 16 εικονοστοιχεία γίνεται έλεγχος για να διαπιστωθεί αν αυτά έχουν τιμή ένα ή μηδέν. Αν η τιμή τους είναι μηδέν, στην έξοδο στέλνεται η τιμή του αντίστοιχου εικονοστοιχείου από την SRAM. Αν η τιμή τους είναι ένα, στην έξοδο, αλλά και στην SRAM, πρέπει να σταλούν οι τιμές των επόμενων 8 bits. Για να γίνει αυτό γρήγορα, έχουν τοποθετηθεί 8 επιπλέον καταχωρητές, οι 9 – 15, οι οποίοι περιέχουν τις τιμές των επόμενων 8 εικονοστοιχείων. Με τον τρόπο αυτό, όταν η τιμή του τρέχοντος εικονοστοιχείου είναι ένα, στην έξοδο του συστήματος οδηγούνται οι τιμές των οκτώ επόμενων καταχωρητών. Μόλις ολοκληρωθεί η διαδικασία και για τα 16 εικονοστοιχεία, μειώνεται κατά ένα η τιμή ενός μετρητή στον οποίο κρατιέται ο αριθμός των κωδικοποιημένων παραθύρων σε κάθε 64 × 64 παράθυρο. Η διαδικασία συνεχίζεται για τα επόμενα κωδικοποιημένα παράθυρα. Όταν ο μετρητής στον οποίο έγινε πριν αναφορά πάρει την τιμή μηδέν, δηλαδή όταν έχει αποσυμπιεστεί όλο το τρέχον 64 × 64 παράθυρο, ακολουθείται η ίδια διαδικασία για τα επόμενα παράθυρα.

Για παράδειγμα, αν η τιμή του πρώτου εικονοστοιχείου ενός 4 × 4 κωδικοποιημένου παραθύρου είναι μηδέν, αυτό αποθηκεύεται στον καταχωρητή 1 και στην έξοδο του συστήματος, μέσω της κατάλληλης τιμής στο σήμα ελέγχου του πολυπλέκτη, οδηγείται η αντίστοιχη τιμή από την SRAM. Αν η τιμή του επόμενου εικονοστοιχείου είναι ένα, στην έξοδο οδηγούνται οι τιμές των καταχωρητών 3 – 10. Οι καταχωρητές 9 – 10 ήδη περιέχουν τις τιμές που θα έπαιρναν οι καταχωρητές 1 – 2 στον επόμενο κύκλο ρολογιού, οπότε τα ζητούμενα 8 bits δεδομένων οδηγούνται άμεσα στην έξοδο

του συστήματος, αλλά και στη μνήμη SRAM. Στη συνέχεια οι τιμές των καταχωρητών 11 – 16 αγνοούνται και ο έλεγχος για την τιμή του επόμενου εικονοστοιχείου γίνεται στον καταχωρητή 3, κ.ο.κ.

Π.3 Περιγραφή των υπολοίπων μονάδων της αρχιτεκτονικής

Το σημαντικότερο τμήμα της αρχιτεκτονικής του συστήματος αποσυμπίεσης είναι αυτό της αποκωδικοποίησης των παραθύρων. Φυσικά, το συντονισμό της λειτουργίας ολόκληρου του συστήματος αναλαμβάνει η μονάδα ελέγχου, η οποία δίνει τα κατάλληλα δεδομένα και ελέγχει το υποσύστημα αποκωδικοποίησης, ελέγχει τις αναγνώσεις και τις εγγραφές στην SRAM, και ελέγχει τον πολυπλέκτη για να περνάει η κατάλληλη τιμή στην έξοδο του συστήματος. Η ακριβής λειτουργία του δεν αναλύεται, εφόσον η προτεινόμενη αρχιτεκτονική δεν έχει υλοποιηθεί.

Η μνήμη επιλέχτηκε να είναι SRAM λόγω της ταχύτητάς της, για τον ίδιο λόγο δηλαδή που επιλέχτηκε και στο σύστημα της συμπίεσης. Το μέγεθός της (64 K × 8 bits) είναι τόσο ώστε η μνήμη που βρίσκεται και στα τέσσερα συνολικά συστήματα αποσυμπίεσης να χωράει ένα ολόκληρο καρέ.

Τέλος, σημειώνεται ότι για ανάλυση 512 × 512 εικονοστοιχείων τα αποσυμπιεσμένα παράθυρα που ανήκουν στο ίδιο καρέ χρειάζεται να αποθηκευτούν προσωρινά σε μία μνήμη, ώστε να βγουν στην έξοδο του συστήματος στη σειρά που πρέπει, δηλαδή να αποτελούν συνεχόμενα τμήματα του καρέ.

Αναφορές

[1] A. H. Sadka, *Compressed Video Communications*, Wiley, pp. 14-72, 2002.

[2] P.H. Westerink, J. Biemond, and G. Muller, *Subband coding of image sequences at low bit rates*, Signal Process Image Commun 2, pp. 441–448, 1990.

[3] M. Gray, *Vector Quantisation,* IEEE ASSP Magazine, 1, No. 2, pp. 4-29, Apr. 1994.

[4] Y. Linde, A. Buzo and R. M. Gray, *An algorithm for vector quantizer design*, IEEE Trans. on Communications, 28, pp. 84-95, Jan. 1980.

[5] M.J. Chen, L.G. Chen and T.D Chieuh, *One-dimensional full search motion estimation algorithm for video coding*, IEEE Trans. on circuits and systems for video technology, Vol. 4, No. 5, pp. 504-509, Oct. 1994.

[6] N. Ahmed, T Natarajan, K. R. Rao, *Discrete cosine transform*, IEEE Trans Computers, 23, pp. 90-93, Jan 1974.

[7] J. Max, *Quantizing for minimum distortion*, IEEE Trans. on Information Theory, 6, No. 1, pp. 7-12, Mar. 1960.

[8] V.Bhaskaran and K. Konstantinides, *Image and video compression standards - algorithms and architectures*, Kluwer academic publishers, 1999.

[9] B.G. Haskell et al., *Image and Video Coding – emerging standards and beyond*, IEEE Transaction on Circuits and Systems for Video Technology 8, No. 7, pp. 814-837, 1998.

[10] MPEG-1 video group, *Information technology-coding of moving pictures and associated audio for digital storage media up to about 1.5 Mbits/s: Part 2-Video*, ISO/IEC 11172-2, International Standard, 1993.

[11] ITU-T H.261: *Video codec for audiovisual services at p × 64 kbit/s* (03/93).

[12] MPEG-2 video group, *Information technology-generic coding of moving pictures and associated audio: Part 2-Video*, ISO/IEC 13818-2, International Standard, 1995.

[13] ITU-T H.263: *Video coding for low bit rate communications* (02/98).

[14] ITU-T recommendation H.263 (draft) Version 2 (H.263+): *Video coding for low bit rate communications* (01/98).

[15] MPEG-4 video group, *Generic coding of audio-visual objects: Part 2-Visual*, ISO/IEC JTCI/SC29/WG11 N1902, FDIS of ISO/IEC 14496-2, Atlantic City, November 1998.

[16] ITU-T H.264: *Advanced video coding for generic audiovisual services* (05/03).

[17] G. Lienhart, R. Lay, R. Manner, *An FPGA video compressor for H.263 compatible bitstreams*, International Conference on Consumer Electronics, Digest of Technical Papers, pp. 320–325, 2000.

[18] J.M. Fernández, F. Moreno and J. Meneses, *A high-performance architecture with a macroblock-level-pipeline for MPEG-2 coding*, Real Time Imaging Journal 2 , pp. 331–340, 1996.

[19] K. R. Rao and R. Yip, *Discrete Cosine Transform – Algorithms, Advantages, Aplications*, Academic Press, 1990.

[20] S.K. Jang, S.D. Kim, J. Lee, G.Y. Choi and J.B. Ra, *Hardware–software co-implementation of a H.263 video codec,* IEEE Transactions on Consumer Electronics, 46, No. 1, pp. 191–200, 2000.

[21] C. Honsawek, K. Ito, T. Ohtsuka, T. Isshiki, Li Dongju, T. Adiono, H. Kunieda, *System-MSPA design of H.263+ video encoder LSI for face focused videotelephony*, The 2000 IEEE Asia-Pacific Conference on Circuits and Systems, 2000.

[22] M. Harrand, J. Sanches, A. Bellon, J. Bulone and A. Tournier, *A Single Chip CIF 30-Hz, H261, H263 and H263+ video encoder/decoder with embedded display controller*, IEEE Journal of Solid-State Circuits, 34, No. 11, pp. 1627–1633, 1999.

[23] Y. Y. Chung, N. W. Bergmann, *Video Compression on FPGA-based custom computers,* International Conference on Image Processing (ICIP '97) 3-Volume Set-Volume 1, Oct. 1997.

[24] S. Ramachandran, S. Srinivasan, *FPGA Implementation of a Novel, Fast Motion Estimation Algorithm for Real-Time Video Compression,* International Symposium on Field Programmable Gate Arrays, pp. 213-219, 2001.

[25] P. Pirsch, N. Demassieux and W. Gehrke, *VLSI Architectures for Video Compression – A survey*, Proceedings of the IEEE vol. 83, No. 2, pp. 220–246, February 1995.

[26] N. Bourbakis, C. Alexopoulos, *Picture data encryption using SCAN patterns*, Pattern Recognition Journal, vol. 25, No. 6, pp.576-581, 1992.

[27] S. Maniccam and N. Bourbakis, On *compression and encryption for digital video*, Int. ISCA Conf. on Parallel and Distributed Systems, NV, USA, pp. 652-657, Aug. 2000.

[28] C. Alexopoulos, *SCAN: An efficient data processing-accessing formal methodology*, PhD thesis, Dept. of Computer Engineering and Informatics, University of Patras, 1989.

[29] S.Maniccam, *A Lossless compression, encryption and information hiding methodology for images and video*, PhD thesis, Dept. ECE, Binghamton University, 2000.

[30] N. Bourbakis, A. Dollas, *SCAN-Based Compression – Encryption – Hiding for Video on Demand*, IEEE Multimedia Magazine, July – September 2003, pp. 79-87.

[31] C. Kachris, S. Maniccam, A. Dollas, N. Bourbakis, *A Reconfigurable Logic-Based Processor for the SCAN Image and Video Encryption Algorithm*, WASP 2002.

[32] A. Dollas, C. Kachris, N. Bourbakis, *Performance Analysis of Fixed, Reconfigurable, and Custom Architectures for the SCAN Image and Video Encryption Algorithm,* FCCM 2003.

[33] C. Kachris, N. Bourbakis, A. Dollas, *A Reconfigurable Logic-Based Processor for the SCAN Image and Video Encryption Algorithm,* International Journal of Parallel Programming, Vol. 31, No. 6, Dec. 2003.

[34] S. Maniccam, N. Bourbakis, *Image and Video Encryption using SCAN patterns*, Pattern Recognition Journal, Vol. 37, pp. 725-737, 2004

[35] Micron Technology, Inc., *MT48LC1M16A1 SIT - 512K x 16 x 2 banks Synchronous DRAM*, May 1999

[36] Xilinx, *XAPP134 (v3.2) - Synthesizable High-Performance SDRAM Controllers,* Nov. 2002

[37] Xilinx, *DS031 - Virtex™-II Platform FPGAs: Complete Data Sheet v3.3*, June 2004

[38] Xilinx Logicore, *Synchronous FIFO V3.0*, Oct 2001

[39] Xilinx Logicore, *Dual-Port Block Memory for Virtex, Virtex-E, Virtex-II, Virtex-II Pro, Spartan-II, and Spartan-IIE V4.0*, Oct. 2001

[40] Cypress Semiconductor, *CY7C1020B - 32K x 16 Static RAM*, Aug. 2002

[41] Andrè Klindworth, *A generic VHDL entity for a typical SRAM with complete timing parameters*, Aug. 1996, available from <http://tech-www.informatik. uni-hamburg.de /vhdl/models/sram/sram.html>