ΤΕΙ ΚΡΗΤΗΣ – ΠΑΡΑΡΤΗΜΑ ΧΑΝΙΩΝ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΝΙΚΗΣ

Μελέτη του αλγορίθμου κρυπτογράφησης Advanced Encryption Standard (AES) και υλοποίησή του σε FPGA με την VHDL.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

Γιώργου Σιδέρη

Επιβλέπων : Δρ. Μηχ. Νικόλαος Στ. Πετράκης

Καθηγητής Εφαρμογών

Χανιά 2009

Πίνακας Περιεχομένων

Περίληψη

Στην παρούσα Πτυχιακή Εργασία μελετώνται τόσο οι θεμελιώδεις έννοιες όσο και η ορολογία της κρυπτογραφίας, προσεγγίζοντας σταδιακά τις διάφορες τεχνικές της, τις αρχές σχεδιασμού και τα είδη κρυπτογράφησης. Δίνεται μεγαλύτερη βαρύτητα στις μεθόδους ιδιωτικού κλειδιού σε σχέση με αυτές δημόσιου κλειδιού. Έχει γίνει αναφορά σε μια πληθώρα μεθόδων κρυπτογράφησης/αποκρυπτογράφησης ξεκινώντας από την αρχαιότητα και φτάνοντας μέχρι τις ημέρες μας με έμφαση στις σύγχρονες μεθόδους που στηρίζονται στην ψηφιακή τεχνολογία. Υλοποιήθηκε σε υλικό (hardware) χρησιμοποιώντας την γλώσσα περιγραφής υλικού VHDL και το ολοκληρωμένο περιβάλλον λογισμικού της Xilinx, ISE 9.2i, ο αλγόριθμος AES ο οποίος αποτελεί πρότυπο από 2001. Τέλος, έγιναν οι κατάλληλες προσομοιώσεις οι οποίες αφενός απέδειξαν την ορθότητα της σχεδίασης και αφετέρου μας βοήθησαν στην εκτίμηση των επιδόσεων.

Abstract

The above dissertation studies the fundamental concepts of cryptography as well as the pertinent terminology. A variety of encrypt / decrypt techniques have mentioned starting from ancient times up to our days. Emphasis has been placed on contemporary methods that are based on digital technology and it contrasts the private key to those of public key. The implementation of AES algorithm (standardized since 2001) has been performed on hardware using the hardware description language VHDL and the complete Xilinx ISE 9.2i environment. The necessary simulations have been made proving the correctness of the design and estimating it's throughput.

1. Εισαγωγή

Θεμελιώδη ρόλο για την παρουσία και λειτουργία μιας επιχείρησης στο σύγχρονο επιχειρηματικό περιβάλλον παίζει η ασφάλεια. Κι αυτό γιατί οι περισσότερες επιχειρήσεις χρησιμοποιούν πλέον τις πληροφορίες σε ψηφιακή μορφή για να διεκπεραιώσουν τις καθημερινές τους λειτουργίες. Έχουν αποθηκευμένα δεδομένα για τους πελάτες τους, τα προϊόντα, τα οικονομικά αποτελέσματα, το προσωπικό τους κ.λπ., τα οποία οφείλουν να προστατεύουν από τους ανταγωνιστές τους αλλά και από τον κίνδυνο της συνολικής ή μερικής απώλειας.

Η κρυπτογράφηση έρχεται να εξασφαλίσει το απόρρητο των προσωπικών πληροφοριών. Πρόκειται για μια επιστήμη που βασίζεται στα μαθηματικά για την κωδικοποίηση και αποκωδικοποίηση των δεδομένων. Οι μέθοδοι κρυπτογράφησης καθιστούν τα ευαίσθητα προσωπικά δεδομένα προσβάσιμα μόνο σε όσους είναι κατάλληλα εξουσιοδοτημένοι. Εξασφαλίζουν έτσι το απόρρητο στις ψηφιακές επικοινωνίες αλλά και στην αποθήκευση ευαίσθητων πληροφοριών.

Το 1977 σχεδιάστηκε από την IBM και υιοθετήθηκε από το εθνικό ίδρυμα προτύπων και τεχνολογίας (NIST) ο αλγόριθμος DES ο οποίος χρησιμοποιείται ακόμα και σήμερα. Για την εξέλιξη του εφευρέθηκε ο Triple DES ο οποίος χρησιμοποιεί κι αυτός την διαδοχή: κρυπτογράφηση – αποκρυπτογράφηση - κρυπτογράφηση (EDE –encryption – decryption - encryption). Συνέχεια στην ιστορία τον κρυπτογραφικών αλγορίθμων έφερε ο Rijndael ο οποίος ήταν προτεινόμενος από το NIST το 1998 μεταξύ αρκετών αλγορίθμων.

Μέσα στα πλαίσια του Τ.Ε.Ι ασχολήθηκα με τις βασικές αρχές των ψηφιακών κυκλωμάτων και γενικά με τον σχεδιασμό και ανάπτυξη ψηφιακών συστημάτων. Αντικείμενο της πτυχιακής εργασίας είναι ο σχεδιασμός σε hardware του προτύπου AES. Πρόκειται για ένα σύστημα που προσφέρει σε βάθος γνώση με τη διαδικασία σχεδίασης και υλοποίησης ενός ψηφιακού κυκλώματος κρυπτογράφησης που βασίζεται στην τεχνολογία των ολοκληρωμένων προγραμματιζόμενων κυκλωμάτων FPGA με χρήση της γλώσσας περιγραφής υλικού (VHDL).

Για την εκπόνηση της πτυχιακής πραγματοποιήθηκε μελέτη σε συγκεκριμένα πεδία. Στο πρώτο στάδιο μελετήθηκαν τα κυκλώματα FPGA, η γλώσσα περιγραφής υλικού VHDL, ο χώρος σχεδίασης που περιλαμβάνει η υλοποίηση του κυκλώματος εσωτερικά του Chip και τα εργαλεία υλοποίησης και ανάπτυξης. Σε επόμενο στάδιο έγινε η ανάλυση του προτύπου AES σε βάθος έτσι ώστε να γίνει ο σωστός σχεδιασμός σε ψηφιακό σχέδιο και να αποφευχθούν λάθη.

Στο δεύτερο κεφάλαιο αναλύονται βασικές έννοιες που χρησιμοποιούνται για την κρυπτογράφηση, μια ιστορική αναδρομή η οποία αναφέρεται σε παλιές τεχνικές κρυπτογράφησης, επίσης αναλύονται κρυπτογραφικοί αλγόριθμοι καθώς και παραλλαγές αλγορίθμων με τους τρόπους λειτουργίας τους. Στο επόμενο κεφάλαιο αναλύεται το πρότυπο AES, αρχικά μια γενική περιγραφή του μαθηματικού υπόβαθρου που χρησιμοποιεί ο αλγόριθμος κρυπτογράφησης και αποκρυπτογράφησης και στη συνέχεια λεπτομερής ανάλυση των πράξεων που κάνει ο αλγόριθμος σε κάθε στάδιο. Στο τέταρτο και τελευταίο

κεφάλαιο αναλύεται η υλοποίηση του αλγορίθμου σε ψηφιακό σχέδιο δίνοντας έμφαση σε συγκεκριμένα σημεία του αλγορίθμου καθώς και μια ελάχιστη εφαρμογή από αυτή που χρησιμοποιεί ο κώδικας από το κομμάτι της FSM, επίσης δίνεται και κώδικας της γλώσσας (VHDL).

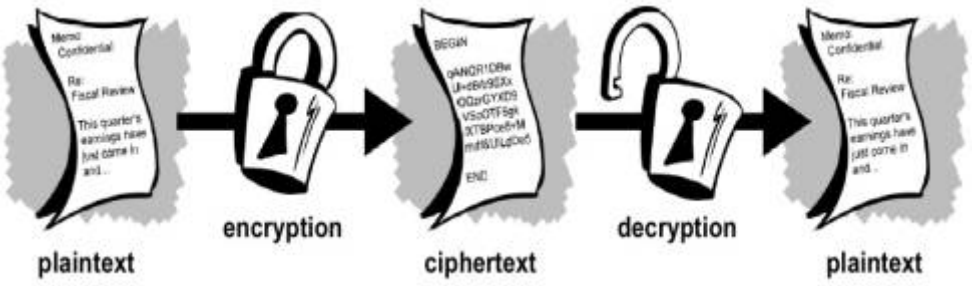
Στόχος της εργασίας είναι να φτιαχτεί ένα σύστημα που σκοπό θα έχει να κρυπτογραφεί δεδομένα τα οποία δέχεται στην είσοδο του άσχετα με το τι είναι(εικόνα, βίντεο, κείμενο) και σαν έξοδο το αποτέλεσμα να είναι δυσνόητο, επίσης με βάση μιας ρύθμισης ενός διακόπτη το δυσνόητο αυτό αρχείο αυτή την φορά σαν είσοδο, να μπορεί να ξαναέρθει στην αρχική του μορφή στην έξοδο. Αυτό το σύστημα εφαρμογή θα μπορεί να έχει για ασφαλή μεταφορά δεδομένων όπου θα απαιτούνται δύο συσκευές μα στον αποστολέα και μια στον παραλήπτη.

Σ’ αυτό το σημείο θα ήθελα να ευχαριστήσω τον κύριο Πετράκη Νίκο, αρχικά για την ευκαιρία που μου έδωσε να μπω στην ομάδα (VHDL) που ο ίδιος έφτιαξε, επίσης για τις πολύτιμες συμβουλές και γνώσεις που μου παρείχε κατά την εκπόνηση της εργασίας. Θα ήταν μεγάλη παράλειψη να μην αναφέρω τα άτομα τα οποία μου συμπαραστάθηκαν όπως μπορούσαν. Συγκεκριμένα τους συμφοιτητές και φίλους Αρτζουχαλτζή Χρήστο και Πέντε Γιώργο.

2. Τεχνικές κρυπτογράφησης

2.1. Βασικές έννοιες και ορολογία

Η κρυπτογράφηση και αποκρυπτογράφηση ενός μηνύματος γίνεται με τη βοήθεια ενός αλγόριθμου κρυπτογράφησης (cipher) και ενός κλειδιού κρυπτογράφησης (key). Συνήθως ο αλγόριθμος κρυπτογράφησης είναι γνωστός, οπότε η εμπιστευτικότητα του κρυπτογραφημένου μηνύματος που μεταδίδεται βασίζεται ως επί το πλείστον στην μυστικότητα του κλειδιού κρυπτογράφησης. Το μέγεθος του κλειδιού κρυπτογράφησης μετριέται σε αριθμό bits. Γενικά ισχύει ο εξής κανόνας: όσο μεγαλύτερο είναι το κλειδί κρυπτογράφησης, τόσο δυσκολότερα μπορεί να αποκρυπτογραφηθεί το κρυπτογραφημένο μήνυμα από επίδοξους εισβολείς. Διαφορετικοί αλγόριθμοι κρυπτογράφησης απαιτούν διαφορετικά μήκη κλειδιών για να πετύχουν το ίδιο επίπεδο ανθεκτικότητας κρυπτογράφησης.



Σχήμα 2.1 Διαδικασία κρυπτογράφησης-αποκρυπτογράφησης.

Κρυπτογράφηση (*encryption*) ονομάζεται η διαδικασία μετασχηματισμού ενός μηνύματος σε μία ακατανόητη μορφή με την χρήση κάποιου κρυπτογραφικού αλγορίθμου ούτως ώστε να μην μπορεί να διαβαστεί από κανέναν εκτός του νόμιμου παραλήπτη.

Η αντίστροφη διαδικασία όπου από το κρυπτογραφημένο κείμενο παράγεται το αρχικό μήνυμα ονομάζεται **αποκρυπτογράφηση (*decryption*)**.

Κρυπτογραφικός αλγόριθμος (cipher) είναι η μέθοδος μετασχηματισμού δεδομένων σε μία μορφή που να μην επιτρέπει την αποκάλυψη των περιεχομένων τους από μη εξουσιοδοτημένα μέρη. Κατά κανόνα ο κρυπτογραφικός αλγόριθμος είναι μία πολύπλοκη μαθηματική συνάρτηση.

Αρχικό κείμενο (*plaintext*) είναι το μήνυμα το οποίο αποτελεί την είσοδο σε μία διεργασία κρυπτογράφησης.

Κλειδί (*key*) είναι ένας αριθμός αρκετών bit που χρησιμοποιείται ως είσοδος στην συνάρτηση κρυπτογράφησης.

Κρυπτογραφημένο κείμενο (*ciphertext*) είναι το αποτέλεσμα της εφαρμογής ενός κρυπτογραφικού αλγόριθμου πάνω στο αρχικό κείμενο.

Κρυπτανάλυση (*cryptanalysis*) είναι μία επιστήμη που ασχολείται με το "σπάσιμο" κάποιας κρυπτογραφικής τεχνικής ούτως ώστε χωρίς να είναι γνωστό το κλειδί της κρυπτογράφησης, το αρχικό κείμενο να μπορεί να αποκωδικοποιηθεί.

2.2. Ιστορική αναδρομή

Η κρυπτογράφηση δεν είναι νέα υπόθεση. Ακόμη και στην αρχαιότητα χρησιμοποιούνταν διάφορες μέθοδοι κρυπτογράφησης, με χαρακτηριστικότερη αυτή του Ιουλίου Καίσαρα, ο οποίος επινόησε έναν απλό αλγόριθμο για να επικοινωνεί με τους επιτελείς του, με μηνύματα που δεν θα ήταν δυνατόν να τα διαβάσουν οι εχθροί του. Ο αλγόριθμος βασιζόταν στην αντικατάσταση κάθε γράμματος του αλφαβήτου με κάποιο άλλο, όχι όμως τυχαία. Ο αλγόριθμος κρυπτογράφησης είναι η ολίσθηση των γραμμάτων του αλφαβήτου προς τα δεξιά. Κάθε γράμμα αντικαθίσταται από κάποιο άλλο με κάποιο κλειδί, π.χ. το 3. Η κρυπτογράφηση δηλαδή του μηνύματος γίνεται με αντικατάσταση κάθε γράμματος από το γράμμα που βρίσκεται τρεις θέσεις δεξιότερά του στο αλφάβητο. Διατηρώντας τον ίδιο αλγόριθμο κρυπτογράφησης και επιλέγοντας διαφορετικό κλειδί, παράγονται διαφορετικά κρυπτογραφημένα μηνύματα.

Αν, για παράδειγμα, το απλό κείμενο είναι η λέξη secret, θα προκύψει το κρυπτογράφημα wigvix. Για να το αποκρυπτογραφήσει κάποιος θα πρέπει να αντιστρέψει τη διαδικασία κρυπτογράφησης, με άλλα λόγια να αντικαταστήσει κάθε γράμμα με αυτό που βρίσκεται τρεις θέσεις αριστερά στο αλφάβητο. Δεν αρκεί να γνωρίζει ότι ο κατάλληλος αλγόριθμος αποκρυπτογράφησης είναι η ολίσθηση των γραμμάτων του αλφαβήτου προς τα αριστερά, αλλά και πόσες θέσεις χρειάζεται να τα ολισθήσει. Πρέπει να γνωρίζει λοιπόν το κλειδί, που σε αυτή την περίπτωση είναι ο αριθμός 3. Στην αρχαία Σπάρτη για την αποστολή απόρρητων στρατιωτικών μηνυμάτων, το μήνυμα γραφόταν σ' ένα κύλινδρο που γύρω του είχε τυλιχτεί μία στενή λωρίδα δέρματος σε διαδοχικές σειρές. Αυτή ήταν η περιβόητη σκυτάλη. Ο κύλινδρος αφαιρούνταν κι έμενε η λωρίδα που μπορούσε να ξαναδιαβαστεί μόνο αν τυλιγόταν με τον ίδιο τρόπο πάνω σε ολόιδιας διαμέτρου κύλινδρο. Κάθε άλλη διαφορετική διάμετρος κυλίνδρου έδινε ακατανόητα μηνύματα. Πολλές φορές γραφόταν σε συνδυασμό με καθρέπτη, ώστε να απαιτείται καθρέπτης και στην ανάγνωση. Άλλη απλούστερη μέθοδος ήταν η αντιστροφή συλλαβών όπως «δημοκρατία» που θα φαινόταν σαν «ηδομαρκίτα». Άλλη μέθοδος χρησιμοποιούσε την ουροδόχο κύστη κάποιου ζώου που φουσκωνόταν και πάνω της γραφόταν με οριακά μικρά γράμματα το μήνυμα. Όταν ξεφουσκωνόταν το μήνυμα έδειχνε πια σαν λεκές. Κατά την αποστολή της συνήθως κρυβόταν καλά, π.χ. σε δοχείο με λάδι και ο παραλήπτης έπρεπε να την φουσκώσει και πάλι για να μπορέσει να διαβάσει το μήνυμα. Στην αρχαία Κίνα το μήνυμα γραφόταν σε λεπτή μεταξωτή κορδέλα η οποία τυλιγόταν σαν μικρό μπαλάκι και καλυπτόταν με κερί. Το μικρό κέρινο μπαλάκι το κατάπινε ο αγγελιοφόρος και έτσι το μετέφερε με την μέγιστη δυνατή ασφάλεια! Ακόμα σπανιότερη ήταν η μέθοδος που το μήνυμα γραφόταν στο ξυρισμένο κεφάλι κάποιου δούλου. Αφού μεσολαβούσε ο απαραίτητος χρόνος επαρκούς ανάπτυξης των μαλλιών του, ο δούλος στελνόταν να παραδώσει το μήνυμα και μετά το επόμενο ξύρισμα κεφαλιού το μήνυμα έβρισκε επιτέλους τον παραλήπτη του. Αυτό μάλλον δείχνει αρκετά καθαρά τις ταχύτητες επικοινωνίας που θεωρούσαν αποδεκτές στον αρχαίο κόσμο. Οι αρχαίοι Εβραίοι χρησιμοποιούσαν το αλφάβητό τους αντεστραμμένο (το τελευταίο γράμμα σαν πρώτο κλπ) για να πετύχουν παρόμοια κρυπτογράφηση. Οι Ρωμαίοι χρησιμοποίησαν

απλούστερες μεθόδους μετάθεσης γραμμάτων κατά μία ή δύο θέσεις. Ο Ιούλιος Καίσαρ χρησιμοποίησε μέθοδο στην οποία υπήρχε μετατόπιση δύο θέσεων, το Α γινόταν Γ κλπ. Ο Αύγουστος Καίσαρ σχεδόν το ίδιο αλλά με μετατόπιση μιας θέσης. Το αόρατο μελάνι ήταν μία ακόμα μέθοδος που χρησιμοποιούνταν αρκετά. Πάνω συνήθως από κάποιο άλλο κείμενο αδιάφορου περιεχομένου γραφόταν με χυμό λεμονιού αντί για μελάνι το κρυφό μήνυμα. Μετά μπορούσε να διαβαστεί στο φως κεριού μόνο από τον υποψιασμένο παραλήπτη.

Ακόμα και βρασμένα αυγά χρησιμοποιήθηκαν για την ασφαλή μεταφορά μηνυμάτων. Τον 16ο αιώνα στην Ιταλία ο Τζιοβάνι Πόρτα έγραφε με μελάνι φτιαγμένο από σκόρδο και ξύδι πάνω στο τσόφλι του αβγού. Το μελάνι απορροφούταν στο εσωτερικό και εξωτερικά δεν φαινόταν τίποτα. Το μήνυμα όμως παρέμενε αποτυπωμένο πάνω στο ασπράδι του βρασμένου αβγού.

2.3. Κρυπτογραφικοί αλγόριθμοι

Οι κρυπτογραφικοί αλγόριθμοι χρησιμοποιούν, κατά κανόνα, (κρυπτογραφικά) κλειδιά, η τιμή των οποίων επηρεάζει την κρυπτογράφηση και την αποκρυπτογράφηση. Το σύνολο των δυνατών τιμών των κλειδιών λέγεται πεδίο τιμών (keyspace). Υπάρχουν δυο κατηγορίες κρυπτογραφικών αλγορίθμων: οι συμμετρικοί και οι ασύμμετροι αλγόριθμοι. Οι συμμετρικοί αλγόριθμοι χρησιμοποιούν το ίδιο κλειδί για την κρυπτογράφηση και την αποκρυπτογράφηση, και για αυτό το λόγο καλούνται επίσης αλγόριθμοι μυστικού κλειδιού ή αλγόριθμοι μονού κλειδιού. Οι ασύμμετροι αλγόριθμοι χρησιμοποιούν ένα ζεύγος κρυπτογραφικών κλειδιών: το δημόσιο κλειδί για την κρυπτογράφηση και το ιδιωτικό για την αποκρυπτογράφηση.

Παραλλαγές

DES είναι το ακρωνύμιο των λέξεων *Data Encryption Standard*. Αντιπροσωπεύει την τυποποίηση *Federal Information Processing Standard (FIPS)* 46-1 που επίσης περιγράφει τον *Data Encryption Algorithm (DEA)*. Αρχικά αναπτύχθηκε από την IBM, ενώ σημαντικό ρόλο στην ανάπτυξη του έπαιξε η NSA και το *National Institute of Standards and Technology (NIST)*. Είναι ο πιο γνωστός και παγκόσμια χρησιμοποιούμενος συμμετρικός αλγόριθμος.

Ο DES είναι block cipher, πιο συγκεκριμένα Feistel cipher, με μέγεθος block 64 bit. Χρησιμοποιεί κλειδί 64 bits από τα οποία τα 8 αποτελούν bits ισοτιμίας. Όταν χρησιμοποιείται για την επικοινωνία, αποστολέας και παραλήπτης μοιράζονται το ίδιο κλειδί.. Επίσης, μπορεί να χρησιμοποιηθεί για κρυπτογράφηση αρχείων αποθηκευμένα σε σκληρό δίσκο σε περιβάλλοντα ενός χρήστη. Για την διανομή των κλειδιών σε περιβάλλον πολλών χρηστών, συνδυάζεται με ασύμμετρο κρυπτοσύστημα.

Triple-DES

Είναι μια παραλλαγή του DES όπου το μήνυμα κρυπτογραφείται και αποκρυπτογραφείται διαδοχικά με διαφορετικά κλειδιά για την ενίσχυση του βασικού αλγόριθμου. Υπάρχουν τέσσερις διαφορετικοί τρόποι για να επιτευχθεί αυτό:

- DES-EEE3 *(Encrypt-Encrypt-Encrypt)*: πραγματοποιούνται τρεις συνεχόμενες κρυπτογραφήσεις με το τρία διαφορετικά κλειδιά.

- DES-EDE3 *(Encrypt-Decrypt-Encrypt)*: το μήνυμα διαδοχικά κρυπτογραφείται, αποκρυπτογραφείται και τέλος κρυπτογραφείται με χρήση τριών διαφορετικών κλειδιών.
- DES-EEE2: είναι η ίδια με την πρώτη διαδικασία εκτός του ότι απαιτούνται δύο διαφορετικά κλειδιά.
- DES-EDE2: είναι η ίδια με την δεύτερη διαδικασία εκτός του ότι απαιτούνται δύο κλειδιά.

Τα επιπλέον κλειδιά δημιουργούνται από το κοινό μυστικό κλειδί με κατάλληλο αλγόριθμο. Από αυτούς τους τρόπους, ο πιο ασφαλής είναι ο DES-EEE3, με την τριπλή κρυπτογράφηση και τα τρία διαφορετικά κλειδιά.

DESX

Ο DESX είναι μια άλλη παραλλαγή του DES. Η διαφορά του DES και του DESX είναι ότι η είσοδος στο DESX περνάει από μια X-OR πράξη με ένα επιπλέον κλειδί 64 bits και ομοίως η έξοδος της κρυπτογράφησης. Η αιτία ανάπτυξης του DESX είναι η δραματική αύξηση της αντοχής του DES σε γνωστές επιθέσεις.

AES (Advanced Encryption Standard)

Το ακρωνύμιο AES προέρχεται από την φράση *Advanced Encryption Standard*. Είναι ένας block cipher που προορίζεται να γίνει τυποποίηση του FIPS και να αντικαταστήσει τον DES.

DSS (Digital Signature Algorithm)

Το National Institute of Standards and Technology (NIST) δημοσιοποίησε το *Digital Signature Algorithm (DSS)*, που είναι μέρος του *Capstone Project* της κυβέρνησης των Ηνωμένων Πολιτειών, τον Μάιο του 1994. Έχει καθιερωθεί σαν το επίσημο αλγόριθμο παραγωγής ψηφιακών υπογραφών της κυβέρνησης των Η.Π.Α.

Βασίζεται στο πρόβλημα του διακριτού λογαρίθμου και χρησιμοποιείται μόνο για παραγωγή ψηφιακών υπογραφών. Η διαφορά από τις υπογραφές του RSA είναι ότι ενώ στο DSA η παραγωγή των υπογραφών είναι πιο γρήγορη από την επιβεβαίωση τους, στο RSA συμβαίνει το αντίθετο: η επιβεβαίωση είναι ταχύτερη από την υπογραφή. Παρ' όλο που μπορεί να υποστηριχθεί ότι η γρήγορη παραγωγή υπογραφών αποτελεί πλεονέκτημα, επειδή ένα μήνυμα υπογράφεται μία φορά αλλά η υπογραφή του μπορεί να επαληθευτεί πολλές φορές, κάτι τέτοιο δεν ανταποκρίνεται στην πραγματικότητα.

RC4, RC5

Ο RC4 είναι ένας stream cipher που σχεδιάστηκε από την Ron Rivest για λογαριασμό της RSA Inc. Έχει μεταβλητό μήκος κλειδιού και λειτουργεί στο επίπεδο του byte. Θεωρείται εξαιρετικά ασφαλής και οι υλοποιήσεις του σε λογισμικό τρέχουν πολύ γρήγορα. Χρησιμοποιείται για κρυπτογράφηση τοπικά αποθηκευμένων αρχείων και για την διασφάλιση της επικοινωνίας μεταξύ δύο απομακρυσμένων σημείων μέσω του πρωτοκόλλου SSL.

Ο RC5 είναι ένας γρήγορος block cipher από τον Ron Rivest για λογαριασμό της RSA Inc το 1994. Έχει πολλούς παραμέτρους: μεταβλητό μήκος κλειδιού, μεταβλητό μέγεθος block και μεταβλητό αριθμό επαναλήψεων. Τυπικές επιλογές για το μέγεθος του block είναι 32 bits (για πειραματικές εφαρμογές), 64 bits (για αντικατάσταση του DES) και 128 bits. Ο αριθμός των

επαναλήψεων μπορεί να είναι από 0 έως και 255. Ο RC5 είναι πολύ απλός στην λειτουργία, πράγμα που τον κάνει εύκολο στην ανάλυση.

IDEA (International Data Encryption Algorithm)

Ο IDEA είναι ένας block cipher που αναπτύχθηκε από τους Lai και Massey. Χρησιμοποιεί block μεγέθους 64 bits και κλειδιά 128 bits. Η διαδικασία της κρυπτογράφησης απαιτεί 8 σύνθετες επαναλήψεις. Παρ' όλο που δεν έχει την κατασκευή ενός Feistel cipher, η αποκρυπτογράφηση γίνεται με τον ίδιο τρόπο που γίνεται και η κρυπτογράφηση. Έχει σχεδιαστεί για να εύκολα εφαρμόσιμος τόσο hardware σε όσο και σε software. Μερικές, όμως, αριθμητικές διεργασίες που χρησιμοποιεί ο IDEA καθιστούν τις λογισμικές εφαρμογές αργές, παρόμοιες σε ταχύτητα με τον DES. Ο IDEA αποτελεί ένα πολύ δυνατό αλγόριθμο που είναι απρόσβλητος από τα περισσότερα είδη επιθέσεων.

Blowfish

Ο Blowfish είναι ένας block cipher που κατασκευάστηκε από τον Schneier. Είναι ένας Feistel cipher με μέγεθος block 64 bits και μεταβλητό μήκος κλειδιού, με μέγιστο μήκος 448 bits. Όλες οι διεργασίες βασίζονται σε X-OR πράξεις και προσθέσεις λέξεων των 32 bits. Από το κλειδί παράγεται πίνακας με τα subkeys που χρησιμοποιούνται σε κάθε γύρο επανάληψης της κρυπτογράφησης. Έχει σχεδιασθεί για 32-bit μηχανές και είναι σημαντικά ταχύτερος από τον DES. Παρ' όλες τις αδυναμίες που έχουν ανακαλυφθεί καθ' όλη την διάρκεια της ύπαρξης του, θεωρείται ακόμα ασφαλής αλγόριθμος.

2.4. Τρόποι λειτουργίας

Για τους κρυπτογράφους μπλοκ, έχουν επινοηθεί αρκετοί τρόποι λειτουργίας (modes) ώστε να βελτιωθούν κάποια χαρακτηριστικά τους όπως η ασφάλεια που προσφέρουν ή να γίνουν πιο κατάλληλοι για διάφορες εφαρμογές. Τέσσερις είναι οι κυριότεροι τρόποι λειτουργίας :

Electronic Codebook (ECB)

Αυτός ο τρόπος λειτουργίας είναι ο απλούστερος και ο πλέον προφανής. Το μυστικό κλειδί χρησιμοποιείται για την κρυπτογράφηση κάθε μπλοκ δεδομένων του plaintext. Κατά συνέπεια με την χρήση του ίδιου κλειδιού, το ίδιο plaintext μπλοκ θα μετατρέπεται πάντα στο ίδιο ciphertext μπλοκ. Είναι ο πλέον κοινός τρόπος λειτουργίας των κρυπτογράφων μπλοκ γιατί είναι ο απλούστερος και άρα ο πιο εύκολα υλοποιήσιμος και συνάμα ο πιο γρήγορος καθώς δεν χρησιμοποιείται κάποιου είδους ανατροφοδότηση. Μειονέκτημα του είναι ότι είναι ο πιο ευάλωτος τρόπος κρυπτογράφησης σε επιθέσεις τύπου brute-force (ως επίθεση brute-force θεωρείται η προσπάθεια εύρεσης του μυστικού κλειδιού με την εξαντλητική δοκιμή πιθανών κλειδιών).

Cipher Block Chaining (CBC)

Χρησιμοποιώντας την CBC λειτουργία, προστίθεται σε έναν κρυπτογράφο μπλοκ ένας μηχανισμός ανατροφοδότησης. Ο τρόπος αυτός λειτουργίας ορίζει ότι προτού να γίνει η κρυπτογράφηση ενός νέου μπλοκ plaintext, γίνεται XOR (αποκλειστικό-Ή) του μπλοκ αυτού και του ciphertext μπλοκ που μόλις πριν έχει παραχθεί. Με τον τρόπο αυτό, 2 ταυτόσημα μπλοκ plaintext δεν κρυπτογραφούνται ποτέ στο ίδιο ciphertext. Σε σχέση με τον ECB προσφέρεται μεγαλύτερη ασφάλεια, με κόστος όμως κυρίως στην ταχύτητα κρυπτογράφησης καθώς για να ξεκινήσει η επεξεργασία ενός μπλοκ plaintext είναι απαραίτητο να έχει

ολοκληρωθεί πλήρως η κρυπτογράφηση του προηγούμενου μπλοκ. Αποτρέπεται έτσι η χρήση τεχνικών pipelining (software ή hardware) που μπορούν να επιταχύνουν την διαδικασία.

Cipher Feedback (CFB)

Ο τρόπος αυτός λειτουργίας επιτρέπει σε έναν κρυπτογράφο μπλοκ να συμπεριφερθεί σαν ένας κρυπτογράφος ροής. Αυτό είναι θεμιτό όταν πρέπει να κρυπτογραφούνται δεδομένα που μπορεί να έχουν μέγεθος μικρότερο από ένα μπλοκ. Παράδειγμα τέτοιας εφαρμογής μπορεί να είναι η διαδικασία κρυπτογράφησης ενός terminal session. Περιληπτικά, κατά την CFB λειτουργία χρησιμοποιείται ένας shift καταχωρητής στο μέγεθος του block μέσα στον οποίο τοποθετούνται τα δεδομένα προς κρυπτογράφηση. Όλος ο καταχωρητής κρυπτογραφείται και αυτό που προκύπτει είναι το ciphertext. Η ποσότητα των δεδομένων που μπαίνουν μέσα στον shift καταχωρητή καθορίζεται από την εφαρμογή.

Output Feedback (OFB)

Στόχος και αυτού του τρόπου λειτουργίας των μπλοκ κρυπτογράφων είναι να εξασφαλίσει ότι το ίδιο plaintext μπλοκ δεν μπορεί να παράγει το ίδιο ciphertext μπλοκ. Σε σχέση με το CBC, χρησιμοποιείται και εδώ ένας μηχανισμός ανατροφοδότησης παρόλα αυτά είναι εσωτερικός και ανεξάρτητος από τα plaintext και ciphertext δεδομένα.

Σημαντικοί αλγόριθμοι αυτής της κατηγορίας είναι οι DES (Data Encryption Standard), 3DES, DESX, ο AES (Advanced Encryption Standard), οι RC2, RC4, RC5 και IDEA (International Data Encryption Algorithm). Οι αλγόριθμοι της σειράς DES είναι οι πλέον χρησιμοποιούμενοι σήμερα αλγόριθμοι, αν και πλέον αντικαθιστούνται από τον AES. Επινοήθηκαν από την IBM την δεκαετία του '70 και υιοθετήθηκαν από το National Bureau of Standards (νυν NIST) των ΗΠΑ. Οι DES αλγόριθμοι χρησιμοποιούν κλειδιά μήκους 56 bits ο 3DES και ο DESX επεκτείνουν κατάλληλα αυτόν τον αριθμό χρησιμοποιώντας περισσότερα κλειδιά) και επεξεργάζονται μπλοκ των 64 bits. Ο AES αλγόριθμος είναι το πρότυπο που καθιερώθηκε από το NIST ως διάδοχος του DES και πλέον αποτελεί τον προτεινόμενο αλγόριθμο κρυπτογράφησης για εφαρμογές υψηλής ασφάλειας. Οι αλγόριθμοι RC είναι αλγόριθμοι μεταβλητού κλειδιού από την RSA Security ενώ ο IDEA χρησιμοποιείται στο πρότυπο PGP (Pretty Good Privacy).

3. Ο αλγόριθμος AES

3.1. Γενικά

Ο αλγόριθμος AES αναλύει μια μπλοκ διαδικασία κρυπτογράφησης μυστικού κλειδιού. Το πρότυπο βασίζεται στον **Rijndael** τον αλγόριθμο. Ανάλογα με τη μήκος κλειδί χρησιμοποιούμε, λέμε για συντόμευση AES-128, AES-192 και AES-256 αντίστοιχα. Ανεξάρτητα από το μήκος κλειδιού, ο αλγόριθμος ενεργεί πάνω σε μπλοκ δεδομένων μήκους 128 bits. Σε κάθε μπλοκ δεδομένων γίνεται μια επεξεργασία η οποία επαναλαμβάνεται έναν αριθμό από φορές ανάλογα με το μήκος κλειδιού. Κάθε επανάληψη ονομάζεται γύρος (round). Στον πρώτο γύρο επεξεργασίας ως είσοδος είναι ένα plaintext και το κλειδί, ενώ στους γύρους που ακολουθούν ως είσοδος είναι το μπλοκ που έχει προκύψει από τον προηγούμενο γύρο καθώς και ένα κλειδί που έχει παραχθεί από το αρχικό με βάση κάποια διαδικασία.

3.2. Μαθηματικό Υπόβαθρο

Σαν είσοδο παίρνει 128 bits (μπλοκ) καθώς και κλειδί ανάλογα με το μήκος, που μπορεί να έχουν μέγεθος 128, 192 ή 256 bits. Τα κλειδιά αυτά ονομάζονται (cipher keys). Ο AES επεξεργάζεται τα δεδομένα με byte. Έτσι τα bits ενός μπλοκ ή ενός κλειδιού χωρίζονται σε ομάδες των 8 για να σχηματιστούν τα bytes. Κάθε byte στον AES αντιστοιχεί σε ένα πολυώνυμο (αριθμητική πεπερασμένων πεδίων - finite field arithmetic). Αν υποθέσουμε ότι τα bits που αποτελούν ένα byte είναι τα $\{b_7, b_6, b_5, b_4, b_3, b_2, b_1, b_0\}$, τότε το byte αυτό αναπαριστά το πολυώνυμο :

$$b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x + b_0 = \sum_{i=0}^{7} b_i x^i$$

Έτσι για παράδειγμα το byte {11001101} αντιστοιχεί στο πολυώνυμο $x^7 + x^6 + x^3 + x^2 + 1$. Το Σχήμα 3. 1 δείχνει την αντιστοιχία bits & bytes.

Input bit sequence	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	...
Byte number	0								1								2								...
Bit numbers in byte	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	...

Σχήμα 3.1 Δεικτοδότηση των bits και bytes.

Όλη η επεξεργασία που εκτελεί ο αλγόριθμος γίνετε πάνω σε ένα δισδιάστατο πίνακα που αποκαλείται Κατάσταση (State). Ο πίνακας αυτός περιλαμβάνει τέσσερις γραμμές από bytes, με κάθε μία γραμμή να αποτελείται από **Nb** bytes. Εφόσον στον AES υποστηρίζονται μπλοκ μεγέθους μόνο 128 bits, το Nb θα έχει τιμή 4. Το μπλοκ εισόδου περιλαμβάνει 16 bytes, τα οποία δεικτοδοτούνται in_0 έως in_{15}. Το κρυπτογραφημένο μπλοκ εξόδου περιλαμβάνει επίσης 16 bytes που δεικτοδοτούνται ως out_0 έως out_{15}. Η State χρησιμοποιεί την μεταβλητή s με δύο δείκτες που δηλώνουν την θέση κάθε byte στον πίνακα. Η πρώτη λοιπόν και τελευταία λειτουργία που μπορεί να υποτεθεί ότι γίνεται στον AES είναι να αντιστοιχηθούν τα bytes εισόδου σε κάποια θέση του πίνακα της State και το αντίστροφο στην έξοδο. Το Σχήμα 3.2 δείχνει πώς γίνεται αυτό.

In0	In4	In8	In12
In1	in5	In9	In13
In2	In6	In10	In14
In3	in7	In11	In15

S0,0	S0,1	S0,2	S0,3
S1,0	S1,1	S1,2	S1,3
S2,0	S2,1	S2,2	S2,3
S3,0	S3,1	S3,2	S3,3

Out0	Out4	Out8	Out12
Out1	Out5	Out9	Out13
Out2	Out6	Out10	Out14
Out3	Out7	Out11	Out15

Σχήμα 3.2 Αντιστοίχηση bytes εισόδου στην State και από την State στην έξοδο.

Η αντιστοίχηση που περιγράφηκε παραπάνω μπορεί να περιγραφεί μαθηματικά. Η αντιστοίχηση εισόδου στην State περιγράφεται από την σχέση :

$$s[r, c] = in[r+4c] \quad \text{για } 0 \le r < 4 \text{ και } 0 \le c < Nb$$

ενώ η αντιγραφή της State στην έξοδο από την σχέση :

$$out[r+4c] = s[r, c] \quad \text{για } 0 \le r < 4 \text{ και } 0 \le c < Nb$$

Μια άλλη ανάλυση που μπορεί να καταλάβει κάποιος συνολικά το state είναι σαν word δηλαδή 32-bit λέξεις. Ένα word περιέχει 4 bytes (σχήμα3.3):

$$w_0 = s_{0,0}s_{1,0}s_{2,0}s_{3,0}$$
$$w_1 = s_{0,1}s_{1,1}s_{2,1}s_{3,1}$$
$$w_2 = s_{0,2}s_{1,2}s_{2,2}s_{3,2}$$
$$w_3 = s_{0,3}s_{1,3}s_{2,3}s_{3,3}$$

Σχήμα 3.3 Ένδειξη ενός Word

3.3. Ανάλυση Αλγορίθμου

Το πρότυπο AES ορίζει ότι τα μπλοκ που επεξεργάζεται ο αλγόριθμος έχουν μέγεθος 128 bits και αυτό ορίζεται από την ποσότητα **Nb** = 4, που συμβολίζει τον αριθμό των 32-bit λέξεων στο μπλοκ. Από την άλλη, τα κλειδιά που χρησιμοποιούνται για την κρυπτογράφηση, μπορούν να έχουν μήκος 128, 192 ή 256 bits. Η μεταβλητή **Nk** συμβολίζει

τον αριθμό των 32-bit λέξεων που μπορεί να περιλαμβάνει ένα κλειδί και κατά συνέπεια μπορεί να πάρει τις τιμές 4, 6 και 8.

Ανάλογα με το μήκος κλειδιού που θα επιλεχθεί για την κρυπτογράφηση, ο αλγόριθμος ορίζει έναν αριθμό από γύρους επεξεργασίας που απαιτούνται για την ολοκλήρωση της. Η μεταβλητή **Nr** χρησιμοποιείται για να δηλώσει το πλήθος των γύρων. Αν χρησιμοποιηθεί μήκος κλειδιού 128 bits τότε απαιτούνται 10 γύροι επεξεργασίας (σχήμα 3.3).

	μήκος κλειδιού (**Nk**)	μήκος λέξεων (**Nb**)	αριθμός γύρων (**Nr**)
AES-128	4	4	10
AES-192	6	4	12
AES-256	8	4	14

Σχήμα 3.3 Αντιστοίχηση **Nk-Nb-Nr** για AES-128, AES-192 και AES-256.

Να σημειωθεί ότι οι παραπάνω συνδυασμοί μήκους λέξεων, μήκους κλειδιού και γύρων επεξεργασίας είναι αυτοί που ορίζονται αυστηρά στο πρότυπο AES.

Τόσο κατά την διάρκεια της διαδικασίας κρυπτογράφησης όσο και αποκρυπτογράφησης, κάθε γύρος επεξεργασίας αποτελείται από μια σειρά μετασχηματισμών σε επίπεδο byte. Οι κυκλικές συναρτήσεις του **Rijndael** αποτελούνται από τέσσερα στρώματα:

Στο πρώτο στρώμα, ένα 8*8 s-box εφαρμόζεται σε κάθε ψηφιολέξη. Στο δεύτερο και τρίτο στρώμα οι γραμμές του πίνακα μετατοπίζονται και οι στήλες αναμιγνύονται. Στο τέταρτο στρώμα τα κλειδιά και οι λέξεις υπόκεινται σε XOR.

(1) SubByte (2) ShiftRow (3) MixColumns (4) AddRoundKey

3.3.1. Λεπτομέρειες Κρυπτογράφησης

Αρχικά ένα μπλοκ εισόδου (plaintext) αντιγράφεται στην State. Μετά από έναν αρχικό γύρο πρόσθεσης κλειδιού, ακολουθούν 10, 12 ή 14 γύροι επεξεργασίας, με τον τελευταίο γύρο να διαφέρει από τους υπόλοιπους. Η τελική State αντιγράφεται στην έξοδο και η επεξεργασία για το συγκεκριμένο block ολοκληρώνεται (παραγωγή του ciphertext μπλοκ).

```
Cipher (byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
  byte state[4,Nb]

  state = in

  AddRoundKey(state, w)

  for round = 1 step 1 to Nr-1
    SubBytes(state)
    ShiftRows(state)
    MixColumns(state)
    AddRoundKey(state, w+round*Nb)
  end for

  SubBytes(state)
  ShiftRows(state)
  AddRoundKey(state, w+Nr*Nb)
```

Σχήμα 3.4 Ο ψευδοκώδικας της διαδικασίας κρυπτογράφησης. Όπου w είναι η επέκταση κλειδιού.

Το μυστικό κλειδί κρυπτογράφησης που χρησιμοποιείται σαν είσοδος στον αλγόριθμο είναι το κλειδί που προστίθεται στο μπλοκ εισόδου πριν αρχίσει η επεξεργασία. Σε καθέναν από τους γύρους επεξεργασίας, όπως αναφέρθηκε παραπάνω, υπάρχει μια φάση κατά την οποία προστίθεται στο μπλοκ και ένα κλειδί. Το κλειδί που προστίθεται στις περιπτώσεις αυτές, δεν είναι το αρχικό μυστικό κλειδί αλλά κάποιο που έχει προκύψει με μια συγκεκριμένη διαδικασία από το μυστικό κλειδί και είναι διαφορετικό για κάθε γύρο. Για τον λόγο αυτό, τα κλειδιά αυτά ονομάζονται round keys. Η διαδικασία με την οποία προκύπτουν τα round κλειδιά ονομάζεται Επέκταση Κλειδιού.

3.3.1.1. Ο Μετασχηματισμός SubBytes

Ο μετασχηματισμός SubBytes αποτελεί μια μη γραμμική αντικατάσταση των bytes της State με την χρήση ενός πίνακα αντικατάστασης (S-Box). Ο πίνακας S-Box *τυπικά* δεν υπολογίζεται κατά την διαδικασία της κρυπτογράφησης, αλλά οι τιμές του έχουν προϋπολογιστεί. Στο Σχήμα 3.5 που ακολουθεί παρατίθενται οι τιμές του πίνακα S-Box όπως τις παρουσιάζει το NIST στο επίσημο έγγραφο για τον AES.

Να σημειωθεί ότι ένας αριθμός στο δεκαεξαδικό σύστημα χρειάζεται 4 bits για να αναπαρασταθεί. Κατά συνέπεια, ένα byte αναπαρίσταται από 2 δεκαεξαδικά ψηφία χωρίζοντας το σε 2 ομάδες των 4 bits. Έτσι η γραμμή x του πίνακα αναφέρεται στα πρώτα 4 bits του byte και η στήλη y στα επόμενα 4.

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

Σχήμα 3.5 Ο πίνακας αντικατάστασης S-Box.

Καλό θα ήταν να διευκρινιστεί τι σημαίνει πολλαπλασιασμός στο GF(2^8). Είναι ο πολλαπλασιασμός μεταξύ πολυωνύμων modulo ένα irreducible πολυώνυμο βαθμού 8. Irreducible ονομάζεται ένα πολυώνυμο αν διαιρείται μονάχα από τον εαυτό του και την μονάδα. Το πολυώνυμο που έχει επιλεχθεί για το AES είναι το:

$$m(x) = x^8 + x^4 + x + 1$$

Η πράξη modulo εξασφαλίζει ότι το πολυώνυμο που θα προκύψει θα είναι ένα δυαδικό πολυώνυμο βαθμού μικρότερου του 8, άρα θα μπορεί να αναπαρασταθεί από ένα byte. Το σύμβολο που χρησιμοποιείται για να διακρίνει την πράξη αυτή από έναν κοινό αριθμητικό

πολλαπλασιασμό είναι το $\bullet$. *[π.χ. {57}*$\bullet${83}={C1}]*

3.3.1.2. Ο Μετασχηματισμός ShiftRows

Ο μετασχηματισμός αυτός επιβάλει την κυκλική ολίσθηση των bytes των γραμμών της State. Η πρώτη γραμμή παραμένει ανέπαφη, ενώ στις υπόλοιπες τα bytes ολισθαίνουν. Το Σχήμα 3. 6 παρουσιάζει ενδεικτικά πώς γίνεται ο μετασχηματισμός αυτός.

Όπως μπορεί να παρατηρηθεί από το Σχήμα, η δεύτερη γραμμή ολισθαίνει αριστερά κατά μία θέση με αποτέλεσμα το πρώτο byte της γραμμής να βρεθεί τελευταίο (κυκλική ολίσθηση). Με αντίστοιχο τρόπο ολισθαίνουν και οι γραμμές 3 και 4 αλλά κατά 2 και 3 θέσεις αντίστοιχα.

State s			
$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,1}$	$s_{1,2}$	$s_{1,3}$	$s_{1,0}$
$s_{2,2}$	$s_{2,3}$	$s_{2,0}$	$s_{2,1}$
$s_{3,3}$	$s_{3,0}$	$s_{3,1}$	$s_{3,2}$

------- no shift ---------->
-> left-shift by C1 (1) ->
-> left-shift by C2 (2) ->
-> left-shift by C3 (3) ->

State s´			
$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

Σχήμα 3.6 Ο μετασχηματισμός ShiftRows.

3.3.1.3. Ο Μετασχηματισμός MixColumns

Ο μετασχηματισμός αυτός εφαρμόζεται στις στήλες της State. Η κάθε στήλη θεωρείται σαν πολυώνυμο τρίτης τάξης με συντελεστές τις τιμές των bytes της στήλης.

$$s(x)_i = s_{3,i} \cdot x^3 + s_{2,i} \cdot x^2 + s_{1,i} \cdot x + s_{0,i}$$

Τα πολυώνυμα πολλαπλασιάζονται modulo ($x^4 + 1$) με ένα καθορισμένο πολυώνυμο που δίνεται από την σχέση :

$$a(x) = (03) \cdot x^3 + (01) \cdot x^2 + (01) \cdot x + (02)$$

Η διαδικασία αυτή του υπολογισμού της αρχικής πράξης $s'(x) = a(x) \otimes s(x)$, όπου με $\otimes$ συμβολίζεται ο modulo πολλαπλασιασμός μετασχηματίζεται τελικά στις εξής σχέσεις (στο παρακάτω σχήμα το modulo συμβολίζεται με $\oplus$) :

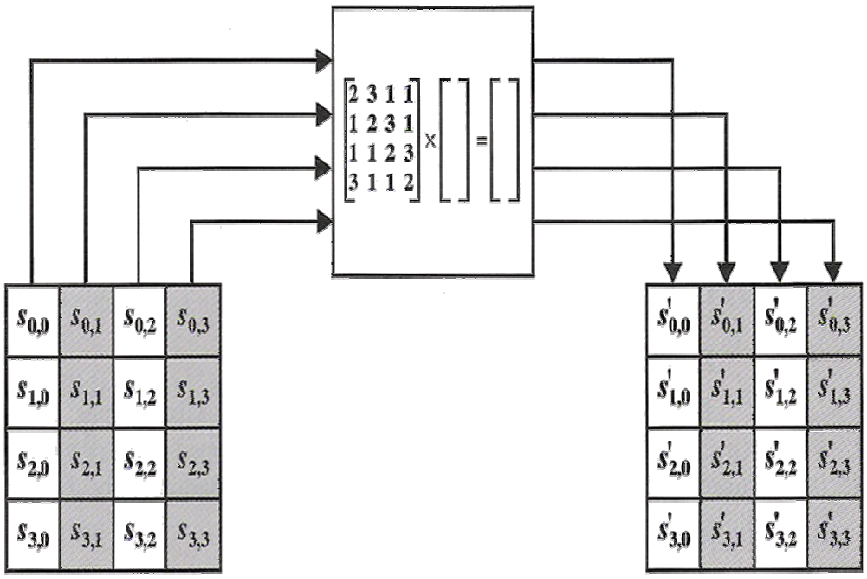
$$s'_{0,c} = (\{02\} * s_{0,c}) \oplus (\{03\} * s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$

$$s'_{1,c} = s_{0,c} \oplus (\{02\} * s_{1,c}) \oplus (\{03\} * s_{2,c}) \oplus s_{3,c}$$

$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} * s_{2,c}) \oplus (\{03\} * s_{3,c})$$

$$s'_{3,c} = (\{02\} * s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} * s_{3,c})$$

για $0 \le c < Nb$



Σχήμα 3.7 Αριστερά παρατηρούμε την είσοδο της MixCollumn και δεξιά την έξοδο της.

3.3.1.4. Ο Μετασχηματισμός AddRoundKey

Ο μετασχηματισμός αυτός επιβάλει την πρόσθεση της τιμής της ποσότητας round key στα bytes των στηλών του πίνακα State. Επειδή κάθε τιμή του round key αποτελείται από Nb λέξεις, επιλέγεται κάθε φορά η επιθυμητή λέξη. Η πράξη αυτή υλοποιείται σαν απλή XOR πράξη ανάμεσα στα bits των ποσοτήτων (bitwise XOR). Η πράξη αυτή μεταφράζεται μαθηματικά στην εξής σχέση:

$$[s'_{0,c}, s'_{1,c}, s'_{2,c}, s'_{3,c}] = [s_{0,c}, s_{1,c}, s_{2,c}, s_{2,c}, s_{3,c}] \oplus [w_{round+Nb+c}]$$

για $0 \leq c < Nb$

3.3.2. Ανάλυση επέκτασης Κλειδιού

Η διαδικασία επέκτασης του κλειδιού (Key Expansion) δέχεται ως είσοδο το αρχικό μυστικό κλειδί (cipher key K), μήκους Nb λέξεων, και παράγει μια ακολουθία κλειδιών. Παράγονται συνολικά από αυτή την διαδικασία Nb (Nr+1) λέξεις καθώς σε κάθε έναν από τους Nr γύρους επεξεργασίας απαιτούνται Nb λέξεις από δεδομένα κλειδιού. Το τελικό key schedule αποτελείται από έναν γραμμικό πίνακα w_i με $0 \leq i < Nb (Nr+1)$ που περιέχει λέξεις των τεσσάρων bytes. Στο Σχήμα 3.8 παρατίθεται σε μορφή ψευδοκώδικα, η διαδικασία επέκτασης κλειδιού.

```
keyExpasion(byte key[4*Nk], word w[Nb*(Nr+1)],Nk)
begin
  word  temp

  i = 0

  while(I < Nk)
    w(i) = word(key[4*i], key[4*i+2], key[4*i+3])
    i = i+1
  end while

  i = Nk

  while (I < Nb * (Nr+1)
    temp = w [i-1]
    if (i mod Nk = 0)
       temp = Subword(Rotword(temp))xor Rcon[i/Nk]
    else if (Nk > 6 and i mod Nk - 4)
       temp = Subword(temp)
    end if
    w(i) =  w(i-Nk) xor temp
    i  =  i+1
  end while
end
```

Σχήμα 3.8 Ο ψευδοκώδικας της διαδικασίας επέκτασης κλειδιού.

Η διαδικασία SubWord εκτελεί ουσιαστικά την ίδια διαδικασία με την συνάρτηση SubByte μόνο που αυτή την φορά προσδιορίζεται η αντίστοιχη τιμή μιας ολόκληρης λέξης, δηλαδή 4 bytes, μέσω του πίνακα S-box. Είναι, τελικά, σαν να εκτελείται διαδοχικά 4 φορές η συνάρτηση SubByte ().

Η συνάρτηση RotWord () δέχεται ως είσοδο μια λέξη και ολισθαίνει κυκλικά τα bytes που την αποτελούν. Αν για παράδειγμα, δοθεί σαν είσοδος μια λέξη [a0, a1, a2, a3] τότε σαν έξοδο έχουμε την λέξη [a1, a2, a3, a0].

Τέλος, το διάνυσμα Rcon[i] (Round constant word array) περιέχει τις τιμές που δίνονται από την σχέση [x^{i-1}, {00, {00}, {00}] με τις τιμές του i να ξεκινούν από 1 και όχι 0.

3.3.3. Ο Αλγόριθμος Αποκρυπτογράφησης

Οι μέθοδος της διαδικασίας κρυπτογράφησης μπορεί να αντιστραφεί και να τοποθετηθούν σε αντίστροφη σειρά ώστε να παραχθεί μια διαδικασία που θα αποκρυπτογραφεί ένα ciphertext του AES. Επίσης και στην διαδικασία της αποκρυπτογράφησης, υπάρχουν τέσσερις διακριτοί μετασχηματισμοί που ενεργούν πάνω στην State κατά την αποκρυπτογράφηση, οι InvShiftRows, InvSubBytes, InvMixColumns και AddRoundKey.

Στο σχήμα 3.9 παρουσιάζεται ο ψευδοκώδικας που περιγράφει την διαδικασία αποκρυπτογράφησης. Η διαδικασία παραγωγής των κλειδιών είναι ταυτόσημη με αυτή που περιγράφηκε στη κρυπτογράφηση.

```
invCipher (byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
  byte state[4,Nb]

  state = in

  AddRoundKey(state, w)

  for round = 1 step 1 to Nr-1
     invSubBytes(state)
     invShiftRows(state)
     invMixColumns(state)
     AddRoundKey(state, w+round*Nb)
end for

invSubBytes(state)
invShiftRows(state)
invAddRoundKey(state, w+Nr*Nb)
```

Σχήμα 3.9 Ο ψευδικώδικας της διαδικασίας αποκρυπτογράφησης.

3.3.3.1. Ο Μετασχηματισμός InvShiftRows

Ο μετασχηματισμός InvShiftRows είναι ο αντίστροφος της ShiftRows της διαδικασίας κρυπτογράφησης. Τα bytes στις τελευταίες τρεις γραμμές της State ολισθαίνουν με αντίθετη φορά από ότι στην ShiftRow διαδικασία. Στο Σχήμα 3.10 παρουσιάζεται ο ακριβής μηχανισμός.

State s			
$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,1}$	$s_{1,2}$	$s_{1,3}$	$s_{1,0}$
$s_{2,2}$	$s_{2,3}$	$s_{2,0}$	$s_{2,1}$
$s_{3,3}$	$s_{3,0}$	$s_{3,1}$	$s_{3,2}$

------- no shift ---------->
-> right-shift by C1 (1) ->
-> right-shift by C2 (2) ->
-> right-shift by C3 (3) ->

State s´			
$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,3}$	$s_{1,0}$	$s_{1,1}$	$s_{1,2}$
$s_{2,2}$	$s_{2,3}$	$s_{2,0}$	$s_{2,3}$
$s_{3,3}$	$s_{3,2}$	$s_{3,3}$	$s_{3,0}$

Σχήμα 3.10 Ο μετασχηματισμός InvShiftRows ολίσθηση αριστερά.

3.3.3.2. Ο Μετασχηματισμός InvSubBytes

Ο μετασχηματισμός αυτός, όπως δηλώνει και το όνομα του, είναι ο αντίστροφος του μετασχηματισμού αντικατάστασης bytes της κρυπτογράφησης. Έτσι, στην περίπτωση αυτή, αντί για το S-Box πίνακα χρησιμοποιείται ο αντίστροφος του (inverse S-Box), ο οποίος και παρουσιάζεται στο Σχήμα 3.11.

		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
	1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
	2	54	7b	94	32	a6	c2	23	3d	ee	4c	95	0b	42	fa	c3	4e
	3	08	2e	a1	66	28	d9	24	b2	76	5b	a2	49	6d	8b	d1	25
	4	72	f8	f6	64	86	68	98	16	d4	a4	5c	cc	5d	65	b6	92
	5	6c	70	48	50	fd	ed	b9	da	5e	15	46	57	a7	8d	9d	84
	6	90	d8	ab	00	8c	bc	d3	0a	f7	e4	58	05	b8	b3	45	06
	7	d0	2c	1e	8f	ca	3f	0f	02	c1	af	bd	03	01	13	8a	6b
	8	3a	91	11	41	4f	67	dc	ea	97	f2	cf	ce	f0	b4	e6	73
	9	96	ac	74	22	e7	ad	35	85	e2	f9	37	e8	1c	75	df	6e
	a	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
	b	fc	56	3e	4b	c6	d2	79	20	9a	db	c0	fe	78	cd	5a	f4
	c	1f	dd	a8	33	88	07	c7	31	b1	12	10	59	27	80	ec	5f
	d	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
	e	a0	e0	3b	4d	ae	2a	f5	b0	c8	eb	bb	3c	83	53	99	61
	f	17	2b	04	7e	ba	77	d6	26	e1	69	14	63	55	21	0c	7d

Σχήμα 3.11 Ο πίνακας inverse S-Box.

3.3.3.3. Ο Μετασχηματισμός InvMixColumns

Είναι ο αντίστροφος του μετασχηματισμού MixColumns. Όπως και ο MixColumns εφαρμόζεται πάνω στις στήλες της State, θεωρώντας κάθε μία από αυτές ένα πολυώνυμο τεσσάρων όρων. Κάθε στήλη, θεωρείται πολυώνυμο του GF(2^8) και πολλαπλασιάζεται modulo $x^4 + 1$ με ένα καθορισμένο πολυώνυμο $a^{-1}(x)$:

$$a^{-1}(x) = \{0b\}x^3 + \{0d\}x^2\{09\}x + \{0e\}$$

Σαν αποτέλεσμα του παραπάνω πολλαπλασιασμού, τα 4 bytes σε μια στήλη αντικαθίστανται από τα ακόλουθα bytes :

$$s'_{0,c} = (\{0e\} * s_{0,c}) \oplus (\{0b\} * s_{1,c}) \oplus (\{0d\} * s_{2,c}) \oplus (\{09\} * s_{3,c})$$
$$s'_{1,c} = (\{09\} * s_{0,c}) \oplus (\{0e\} * s_{1,c}) \oplus (\{0b\} * s_{2,c}) \oplus (\{0d\} * s_{3,c})$$
$$s'_{2,c} = (\{0d\} * s_{0,c}) \oplus (\{09\} * s_{1,c}) \oplus (\{0e\} * s_{2,c}) \oplus (\{0b\} * s_{3,c})$$
$$s'_{3,c} = (\{0b\} * s_{0,c}) \oplus (\{0d\} * s_{1,c}) \oplus (\{09\} * s_{2,c}) \oplus (\{0e\} * s_{3,c})$$

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 0e & 0b & 0d & 09 \\ 09 & 0e & 0b & 0d \\ 0d & 09 & 0e & 0b \\ 0b & 0d & 09 & 0e \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Σχήμα 3.12 Ο παραπάνω πίνακας απεικονίζει τον πολλαπλασιασμό τον bytes.

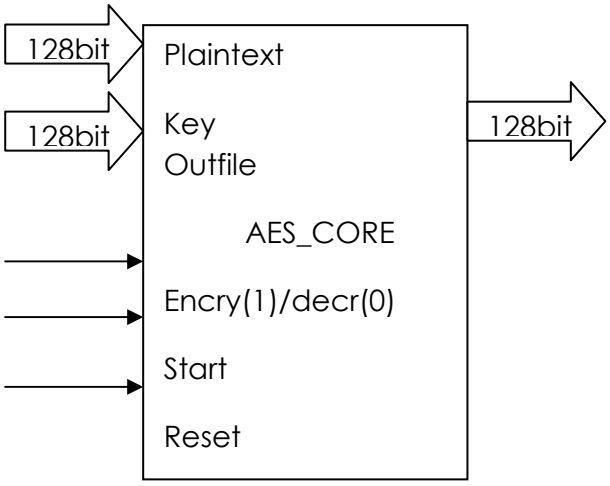
3.3.3.4. Ο Αντίστροφος Μετασχηματισμός AddRoundKey

Εφόσον ο μετασχηματισμός αυτός είναι μια απλή XOR πράξη, είναι από μόνος του αντιστρέψιμος και κατά συνέπεια είναι ταυτόσημος με τον μετασχηματισμό που περιγράφηκε στην ενότητα 3.3.1.4.

4. Υλοποίηση AES

4.1. Μπλοκ διάγραμμα AES

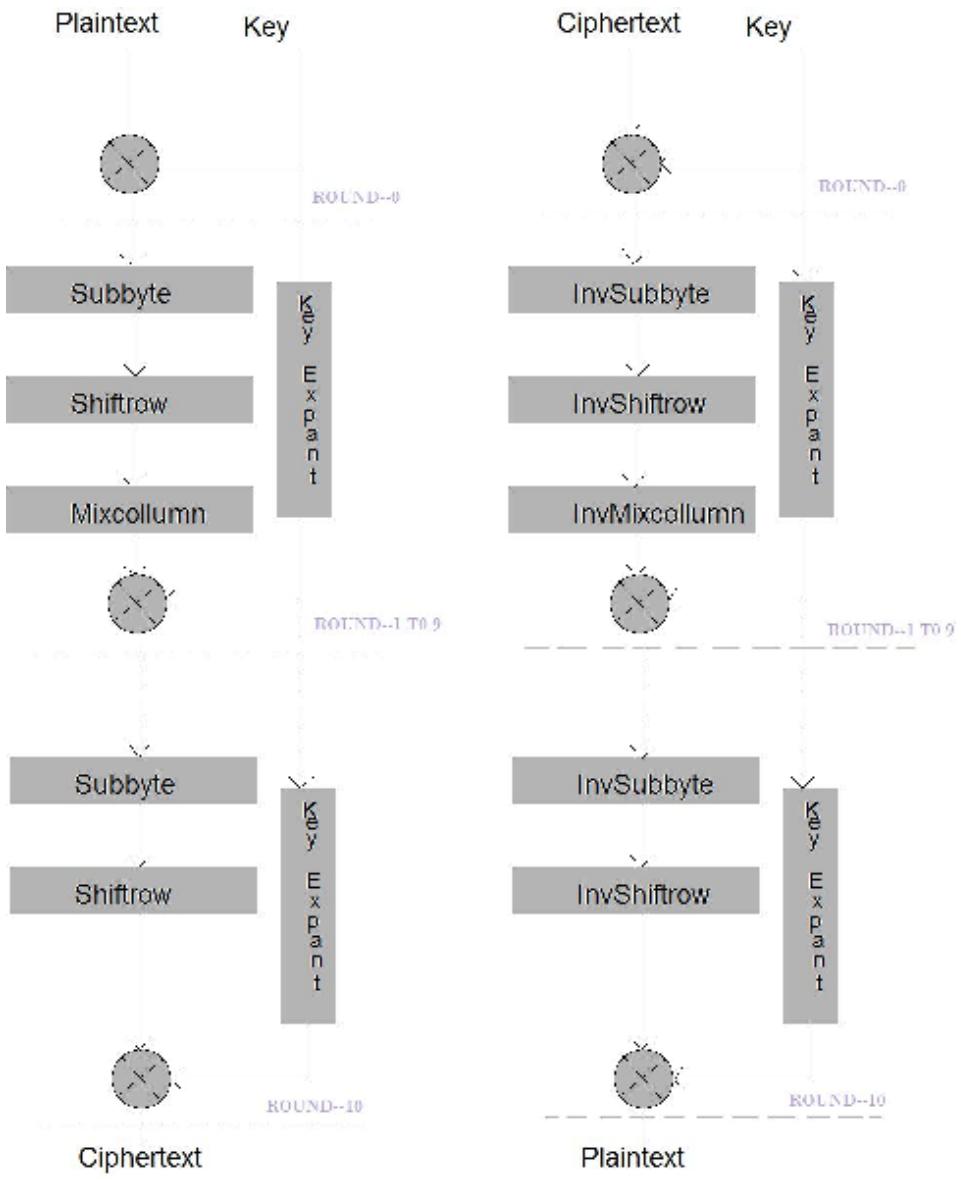
Στο σχήμα 4.1 παραθέτεται το μπλοκ διάγραμμα, από τον NIST FIPS AES αλγόριθμο στα 128bit. Αρχικά ο διακόπτης encry/decr πρέπει να είναι στο λογικό ‘1’, για να ξεκινήσει η διαδικασία της κρυπτογράφησης καθώς επίσης και ο διακόπτης start—>‘1’, για να λειτουργήσει ο αλγόριθμος του κλειδιού. Η κρυπτογράφηση έχει είσοδο 128 bit (plaintext) και αναπαράγει μαζί με το 128 bit (key) το κρυπτογραφημένο αρχείο128bit (ciphertext). Όταν θέλω να κάνω αποκρυπτογράφηση η διαδικασία είναι ίδια εκτός από τον διακόπτη encry/decr→’0’.



Σχήμα 4.1 Μπλοκ διάγραμμα AES επεξεργαστή.

4.2. Αναλυτική περιγραφή των πράξεων ενός γύρου

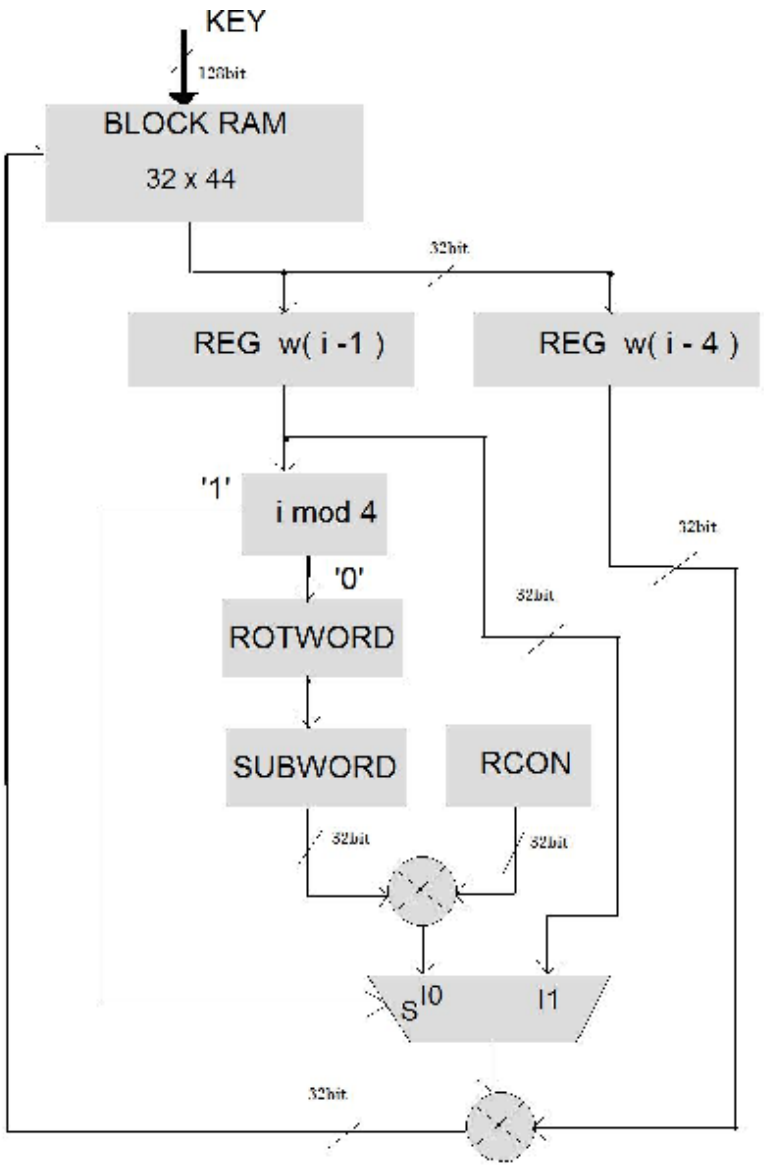
Η διαδικασία της κρυπτογράφησης έχει ως εξής : Στον αρχικό γύρο (round0) γίνεται μια πράξη xor μεταξύ plaintext και key. Στην συνέχεια από τον round1 έως τον round9 ακολουθείται μια σειρά πράξεων (SubByte, ShiftRow, MixCollumn) το αποτέλεσμα γίνεται xor με το key. Τέλος στον round10 ακολουθείται η ίδια διαδικασία εκτός από την πράξη MixCollumn. Η διαδικασία της αποκρυπτογράφησης είναι ακριβώς ίδια με την κρυπτογράφηση μόνο που οι πράξεις είναι αντίστροφες. Να σημειώσουμε ότι όλη η επεξεργασία του σήματος γίνεται στα 32bit, εκτός από την πράξη MixCollumn η οποία γίνεται στα 8 bit.



Σχήμα 4.2 Μονοπάτι υπολογισμού (α) κρυπτογράφηση (β) αποκρυπτογράφηση.

4.3. Περιγραφή της διαδικασίας επέκτασης κλειδιού

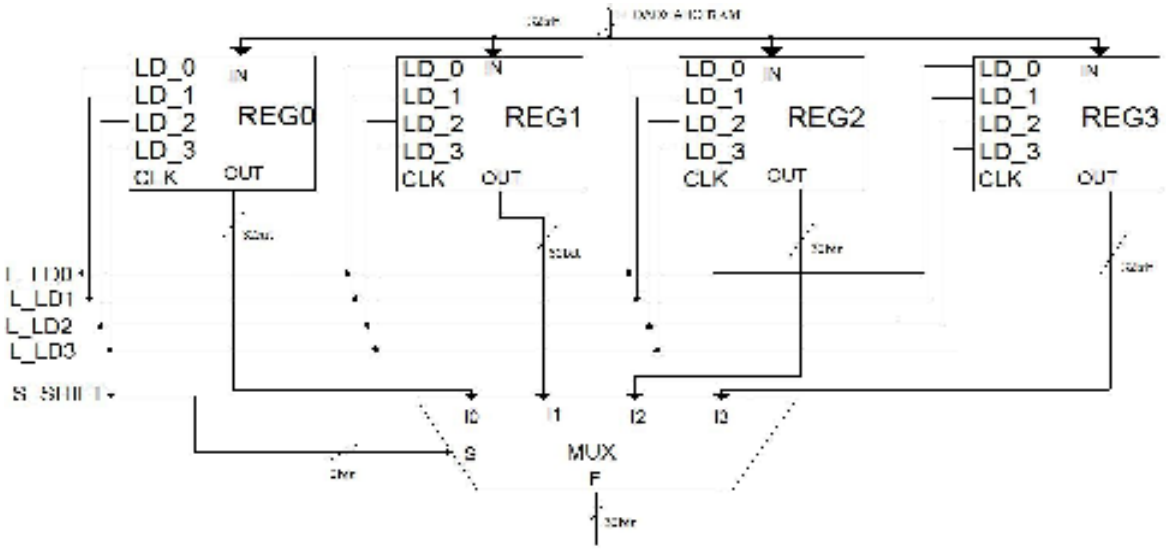
Αρχικά γεμίζουμε τις 4 πρώτες θέσεις της RAM και ορίζουμε με i τον αριθμό των θέσεων της μνήμης. Το επόμενο βήμα είναι να γεμίσουμε τους καταχωρητές με (i-1, i-4). Ας ονομάσουμε με A -> (i-1) και B -> (i-4) . Ανάλογα με τι τιμή έχει ο μετρητής i δίνει αποτέλεσμα στην πράξη i mod 4 (όπου 4 ορίζουμε τον αριθμό λέξεων του κλειδιού, κάθε λέξη 32bit). Όταν το υπόλοιπο της διαίρεσης είναι μηδέν η λέξη ακολουθεί τις πράξεις ([Rotword, Subword] xor Rcon) και έχουμε σαν αποτέλεσμα Ά το οποίο γίνεται xor με το B, το αποτέλεσμα καταχωρείται στην RAM. Όταν το υπόλοιπο είναι άσσος , γίνεται μόνο xor μεταξύ A και B. Αυτή η διαδικασία επαναλαμβάνετε 44 φορές.



Σχήμα 4.3 Μονάδα επεξεργασίας κλειδιού.

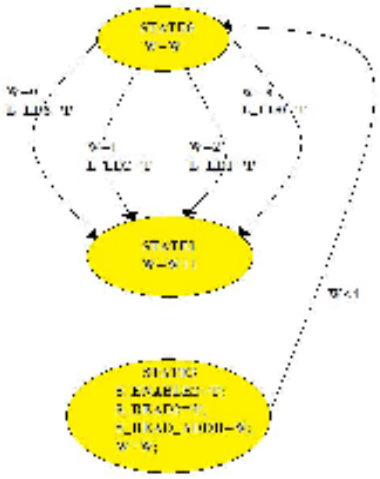
4.4. Πλήρης ανάλυση της πράξης ShiftRow

Στο σχήμα 4.4 και 4.5 παρατηρούμε ολόκληρη την διαδικασία για την πράξη ShiftRow. Στο σχήμα 4.4 έχουμε 4 καταχωρητές και έναν πολυπλέκτη. Ο καθένας από τους καταχωρητές έχει μια είσοδο των 32bit,μια έξοδο επίσης των 32bit,σήματα χρονισμού και σήματα δεικτοδότησης. Όπως παρατηρούμε από το σχήμα οι καταχωρητές έχουν κοινή είσοδο την οποία την παίρνουν από την έξοδο της RAM. Τα σήματα δεικτοδότησης αντιστοιχούν και σε ένα byte το καθένα (4 σήματα άρα 4byte). Ο κώδικας των καταχωρητών είναι έτσι γραμμένος ώστε κάθε φορά που είναι στον άσσο κάποιο από αυτά τα σήματα να βγαίνει στην έξοδο του καταχωρητή το ανάλογο byte (σχήμα 4.6), όσο αφορά την σειρά με την οποία θα βγαίνουν τα byte από καταχωρητή σε καταχωρητή είναι διαφορετική. Ο πολυπλέκτης παίρνει σήμα select από την διεύθυνση εγγραφής της RAM Αυτό που πετυχαίνουμε με αυτό το κύκλωμα είναι ότι μόλις σε 13 παλμούς έχουμε καταφέρει να φτιάξουμε 4 λέξεις των 32bit που αντιστοιχούν στην ShiftRow.



Σχήμα 4.4 Ψηφιακό σχέδιο στο εσωτερικό του κυκλώματος για την πράξη ShiftRow

Το παρακάτω σχήμα μας δείχνει πως μπορούμε να χρησιμοποιήσουμε την FSM. Αρχικά βάζουμε έναν μετρητή να μετράει ως το 4. Όταν ξεκινάει ο κώδικας στο state0, ανάλογα με τι τιμή έχει ο μετρητής περνάει και από τον αντίστοιχο κλάδο, όπως παρατηρούμε σε αυτό το σημείο γίνεται και η δεικτοδότηση. Στο επόμενο state1 γίνεται μια αύξηση του μετρητή. Στο τελευταίο state3 εμφανίζουμε μια άλλη λέξη στην κοινή είσοδο των καταχωρητών και η διαδικασία επαναλαμβάνεται.



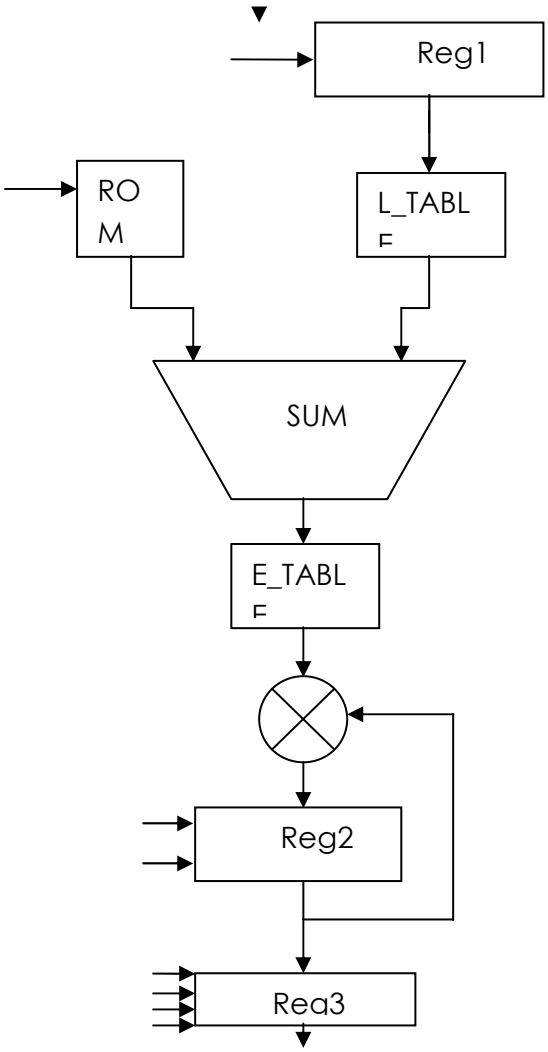
Σχήμα 4.5 Κομμάτι FSM για την υλοποίηση της ShiftRow

```
begin
 process(clk) begin
  if (clk 'event and clk ='1') then
```

```
        if ld_3='1' then
           out_reg(31 downto 24)<= in_reg(31 downto 24);
        elsif ld_2='1' then
           out_reg(23 downto 16)<= in_reg(23 downto 16);
        elsif ld_1='1' then
           out_reg(15 downto 8)<= in_reg(15 downto 8);
        elsif ld_0='1' then
           out_reg(7 downto 0)<= in_reg(7 downto 0);
        end if;
    end
```

Σχήμα 4.6 Κώδικας καταχωρητή

4.5. Περιγραφή υλοποίησης MixCollumn



Σχήμα 4.7 Σχηματικό Mixcollumn στο εσωτερικό του κυκλώματος

Για την υλοποίηση της MixCollumn ήταν απαραίτητη η μετάφραση του μαθηματικού προτύπου (παρ3.3.1.3)το οποίο το χρησιμοποίησα με τον τρόπο του Adam Berent(aes by example). Η επεξεργασία της MixCollumn γίνεται στα 8bit. Χρησιμοποιεί 3 καταχωρητές, μια ROM (4 x 8), μια πράξη XOR, 2 πίνακες (ROM 256 x 8) και έναν ιδιαίτερο αθροιστή. Η πράξη η οποία πρέπει να σχηματιστεί έχει ως εξής: π.χ. έχουμε την λέξη 6353E08C (32bit), για το byteOut (0) = E[L(63)+L(02)] xor E[L(53)+L(03)] xor e0 xor 8c, όπου 02,03 αποθηκευμένες τιμές στην ROM. Η παραπάνω πράξη θα εκτελεστεί άλλες 3 φορές, σε κάθε

κύκλο θα γίνεται μια ολίσθηση δηλαδή : 63 xor E[L(53)+L(02)] xor E[L(e0)+L(03)] xor 8c. Το σχήμα 4.7 παρουσιάζεται την παραπάνω μαθηματική έκφραση σε ψηφιακό σχέδιο. Για το συγχρονισμό του κυκλώματος χρησιμοποιείται FSM. Στο σχήμα 4.8 βλέπουμε τον κώδικα του αθροιστή, ο οποίος εκτός ότι αθροίζει δυο 8bit λέξεις κάνει και αφαίρεση το αποτέλεσμα της άθροισης, όταν αυτό βγει παραπάνω από 8bit, με το FF.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_ARITH.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;

entity sum is
    Port ( in_a,in_b : in  STD_LOGIC_VECTOR (7 downto 0);
           s : out  STD_LOGIC_VECTOR (7 downto 0));

end sum;
  architecture Behavioral of sum is

signal a,b  : unsigned(8 downto 0);
signal ssum : unsigned(8 downto 0);
signal st:std_logic_vector(8 downto 0);

  begin

a(8)<='0';
a(7 downto 0)<= unsigned(in_a);
b(8)<='0';
b(7 downto 0)<= unsigned(in_b);

ssum<= a+b;
st<= std_logic_vector(ssum);

   process(st,in_b)
begin
        if st(8) = '0' then
          s<= st(7 downto 0);
               elsif st(8)  = '1' then
          s<=st(7 downto 0)+1;
        end if;
   end process;
end Behavioral;
```

Σχήμα 4.8 Κώδικας του αθροιστή

4.6. Κώδικας αρχείου εξομοίωσης

Στο σχήμα 4.9 παρατηρούμε τον κώδικα εξομοίωσης του συστήματός μας κατά την διαδικασία κρυπτογράφησης. Έχουμε βάλει για κλειδί μια λέξη των 128bit και για κείμενο άλλη μια λέξη πάλι των 128bit, σε αυτό το σημείο ορίζουμε και τον διακόπτη να κάνει κρυπτογράφηση (encr (1)/decr (0)). Η συχνότητα στην οποία λειτουργεί είναι 333,3MHZ με απόδοση 14,7Mbps.

```
ARCHITECTURE behavior OF aes_t1b_vhd IS

	-- Component Declaration for the Unit Under Test (UUT)
	COMPONENT A_E_S
	PORT(
		in_key : IN std_logic_vector(127 downto 0);
		plaintext : IN std_logic_vector(127 downto 0);
		reset : IN std_logic;
		start : IN std_logic;
		clk : IN std_logic;
		encr : IN std_logic;
		out_file : OUT std_logic_vector(127 downto 0)

		);
	END COMPONENT;

	--Inputs
	SIGNAL reset :  std_logic := '0';
	SIGNAL start :  std_logic := '0';
	SIGNAL clk :  std_logic := '0';
	SIGNAL encr :  std_logic := '0';
	SIGNAL in_key :  std_logic_vector(127 downto 0) := (others=>'0');
	SIGNAL plaintext :  std_logic_vector(127 downto 0) := (others=>'0');

	--Outputs
	SIGNAL out_file :  std_logic_vector(127 downto 0);

BEGIN
	-- Instantiate the Unit Under Test (UUT)
	uut: A_E_S PORT MAP(
		in_key => in_key,
		plaintext => plaintext,
		out_file => out_file,
		reset => reset,
		start => start,
		clk => clk,
		encr => encr
	);
	tb : PROCESS
	BEGIN

		start<='0';
		reset<='1';
		clk <='0';
		wait for 3 ns;
		clk <='1';
		wait for 3 ns;
		reset<='0';
		clk <='0';
		wait for 3 ns;
		clk <='1';
		wait for 3 ns;

		start<='1';
		in_key<= x"0c0d0e0f"&   --w3
			x"08090a0b"&   --w2
```

```
                              x"04050607"&   --w1
                              x"00010203";   --w0

        encr <='0';         --   1 = encryption 0 =decryption

        plaintext<=x"ccddeeff"&   --3b
                    x"8899aabb"&   --2b
                                x"44556677"&   --1b
                    x"00112233";   --0b

         for t in 1 to 2900 loop
              clk<='0';
              wait for 3 ns;
              clk<='1';
             wait for 3 ns;
         end loop;
            wait;
        END PROCESS;
END;
```

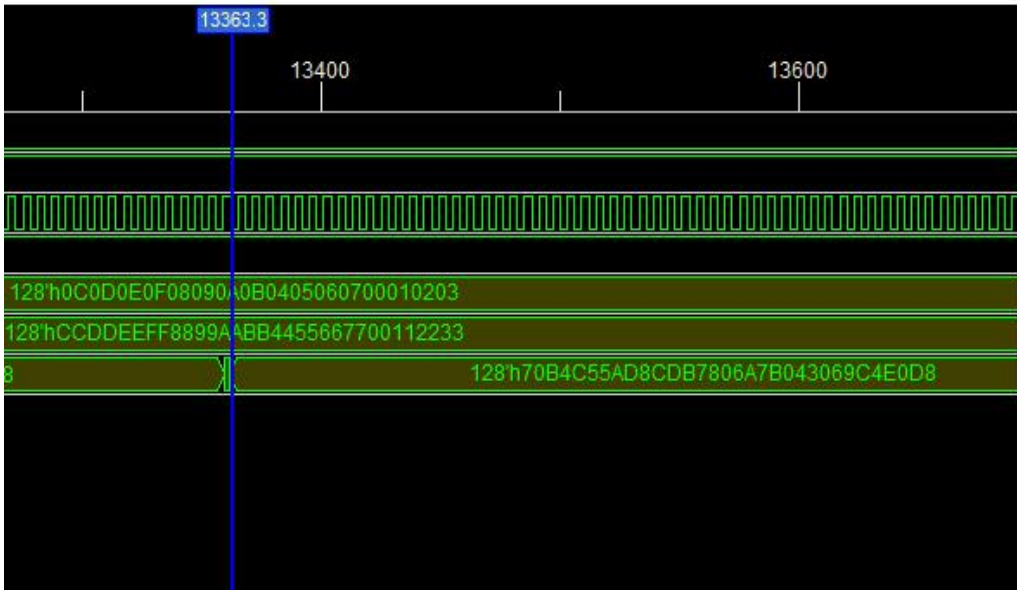
Σχήμα 4.9 Παρατηρούμε κώδικα εξομοίωσης.

4.7. Αποτελέσματα εξομοίωσης

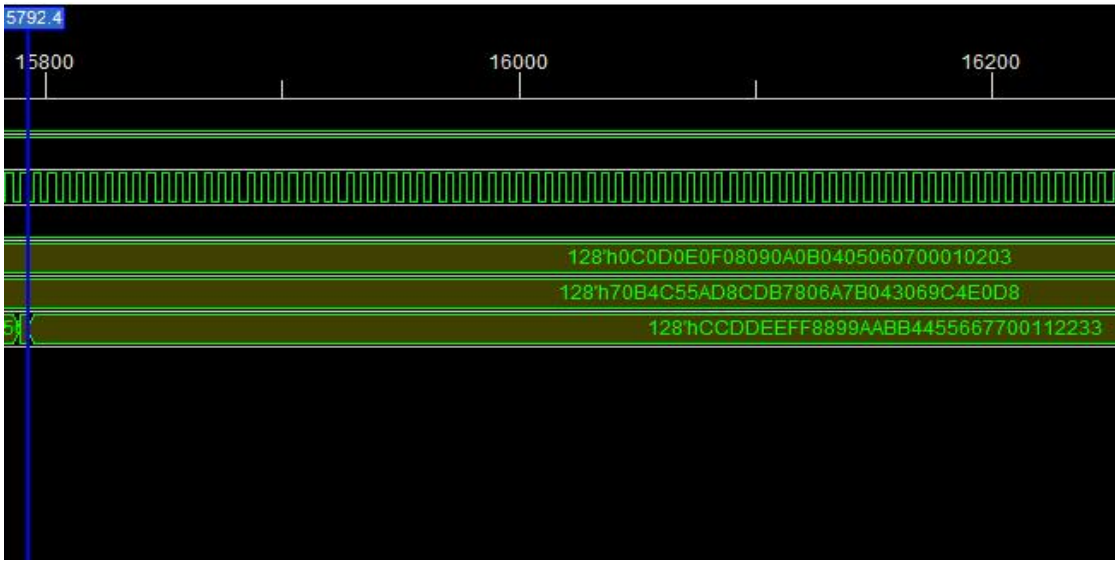
Στο σχήμα 4.10 βλέπουμε την προσομοίωση της κρυπτογράφησης, σαν είσοδο δέχεται: ccddeeff8899aabb4455667700112233 και για κλειδί : 0c0d0e0f08090a0b0405060700010203 Το αποτέλεσμα είναι :

70b4c55ad8cdb7806a7b043069c4e0d8 (ciphertext)

Στο σχήμα 4.11 βλέπουμε την προσομοίωση της αποκρυπτογράφησης σαν είσοδο έχει το κρυπτογραφημένο : 70b4c55ad8cdb7806a7b043069c4e0d8(ciphertext) το κλειδί παραμένει το ίδιο και όπως παρατηρούμε ξαναπαίρνουμε το αρχικό κείμενο.



Σχήμα 4.10 Προσομοίωση κρυπτογράφησης.



Σχήμα 4.11 Προσομοίωση αποκρυπτογράφησης.

5. Συμπεράσματα

Η κρυπτογραφία είναι μια επιστήμη που έχει τις ρίζες της στην αρχαιότητα και εφαρμόζεται σε πολλούς τομείς της σύγχρονης ζωής ανάμεσα στους οποίους και η επιστήμη υπολογιστών. Κρυπτογραφία είναι η μελέτη των μαθηματικών τεχνικών που σχετίζονται με τις πτυχές της ασφάλειας όπως είναι η εμπιστευτικότητα, η ακεραιότητα των δεδομένων, η αυθεντικότητα και η πιστοποίηση αυθεντικότητας. Η κρυπτογραφία δεν είναι μόνο το μέσο που προστατεύει την πληροφορία αλλά ένα σύνολο τεχνικών.

Απαραίτητοι για την εφαρμογή της κρυπτογραφίας είναι οι αλγόριθμοι υλοποίησης του DES, Triple DES, AES, RC4-RC5 και ο αλγόριθμος IDEA. Οι παραπάνω είναι κάποιοι από τους αλγόριθμους κρυπτογράφησης ιδιωτικού κλειδιού, καθένας από τους οποίους είναι κατάλληλος για την υλοποίηση μιας ή περισσοτέρων από τις υπηρεσίες που προσφέρει η κρυπτογραφία.

Ο AES είναι το επόμενο πρότυπο μετά τον DES. Επικράτησε στον διαγωνισμό του NIST (National Institute of Standards and Technology), (πρόταση Rijndael) και λειτουργεί με ομάδες των 128bits (block cipher) χρησιμοποιώντας κλειδιά των 128, 192, και 256 bits. Η κρυπτογράφηση μετατρέπει τα στοιχεία σε μια ακατανόητη μορφή αποκαλούμενη κρυπτογράφημα και η αποκρυπτογράφηση του κρυπτογραφήματος μετατρέπει ξανά τα στοιχεία πίσω στην αρχική τους μορφή, αποκαλούμενη (plaintext).

Η υλοποίηση έγινε σε 128-bit σύμφωνα με το πρότυπο FIPS 197. Η επεξεργασία της κρυπτογράφησης δέχεται σαν είσοδο μια λέξη (128-bit plaintext) και αναπαράγει μια αντίστοιχη λέξη (128-bit ciphertext) χρησιμοποιώντας ένα κλειδί (128-bit). Η αποκρυπτογράφηση έχει για είσοδο το ciphertext και αναπαράγει το plaintext με χρήση του ίδιου κλειδιού. Οι μηχανισμοί της κρυπτογράφησης-αποκρυπτογράφησης καθώς και της επεξεργασίας κλειδιού βρίσκονται στον ίδιο επεξεργαστή. Το κύκλωμα δουλεύει στα 32-bit, περιέχει μνήμες RAM, ROM, καταχωρητές καθώς και πολυπλέκτες. Ήταν προτιμότερο για λόγους ταχύτητας και ισχύος η επεξεργασία κλειδιού να ολοκληρωθεί προτού ξεκινήσει η διαδικασία κρυπτογράφησης ή αποκρυπτογράφησης. Συγκεκριμένα ο χρήστης επιλέγει τον διακόπτη start στο on (‘1’). Σε 403 ρολόγια και με συχνότητα 333,3MHZ έχει ολοκληρωθεί η διαδικασία της επέκτασης κλειδιού. Για την κρυπτογράφηση ο 1^{o} κύκλος ο οποίος κάνει μόνο μια πράξη υλοποιείται σε 21 ρολόγια οι υπόλοιποι εκτός του τελευταίου κύκλου υλοποιούνται σε 2380 ρολόγια και ο τελευταίος γύρος σε 33 ρολόγια. Η αποκρυπτογράφηση στο σύνολό της βγαίνει 30 ρολόγια παραπάνω. Παρακάτω έχει γίνει ο υπολογισμός του throughput.

t_{min} = 3ns,

n = 2900 (αριθμός ρολογιών)

Άρα n*t =8700ns

128bit /8700ns = 0,0147Gbps = 14,7Mbps.

Μια λύση για καλύτερα αποτελέσματα από τα παραπάνω είναι η τεχνική διοχέτευσης (pipeline) αυτό θα βοηθούσε την διάσπαση μιας διαδικασίας σε διαφορετικές φάσεις, συνεπώς το σύστημα θα μπορούσε να γίνει αρκετά πιο γρήγορο.

Επίσης, για παραπάνω ασφάλεια υπάρχουν παραλλαγές που αφορούν το μήκος του κλειδιού, δηλαδή 192bit ή ακόμα και 256bit.

Το σύστημα υλοποιήθηκε σε FPGA της Xillinx συγκεκριμένα της οικογένειας Virtex 5 και μοντέλο XC5VLX110T χρησιμοποιεί 18.624 πύλες και εργάζεται από 0.950V—1.050V.

Τέλος οι εξελίξεις στην τεχνολογία των κυκλωμάτων FPGA μπορούν να επιφέρουν ουσιαστικές αλλαγές στην σχεδίαση ψηφιακών συστημάτων και να δώσουν νέες προοπτικές.

Βιβλιογραφία

Βιβλία:

1. "Applied Cryptography", Bruce Schneier, εκδ. John Wiley & Sons, 1996.
2. «Ασφάλεια δικτύων Υπολογιστών», Πομπόρτσης Α., Παπαδημητρίου Γ., Εκδόσεις Τζιόλα, 2003.

Ιστότοποι:

1. http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf/

2. http://www.iaik.tu-graz.ac.at/research/krypto/AES/

3. http://www.garykessler.net/library/crypto.html/

4. http://jsnerd.googlepages.com/index01a.htm/

5. http://www.net-security.org/dl/articles/AESbyExample.pdf/