ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΛΟΠΟΝΝΗΣΟΥ

ΣΧΟΛΗ ΜΗΧΑΝΙΚΩΝ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Σχεδιασμός Αρχιτεκτονικής και Υλοποίηση σε FPGA του Αλγορίθμου των Ελλειπτικών Καμπύλων

Ευστάθιος Νικήτας Α.Μ 2756

ΕΠΙΒΛΕΠΩΝ: Παρασκευάς Κίτσος, Καθηγητής

ΠΑΤΡΑ 2024

Εγκρίθηκε από την τριμελή εξεταστική επιτροπήΠάτρα, Ημερομηνία

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. Τζήμας Ιωάννης
2. Χριστοδούλου Σωτήρης
3. Παρασκευάς Κίτσος

Υπεύθυνη Δήλωση Φοιτητή

Βεβαιώνω ότι είμαι συγγραφέας αυτής της εργασίας και ότι κάθε βοήθεια την οποία είχα για την προετοιμασία της είναι πλήρως αναγνωρισμένη και αναφέρεται στην εργασία. Επίσης έχω αναφέρει τις όποιες πηγές από τις οποίες έκανα χρήση δεδομένων, ιδεών ή λέξεων, είτε αυτές αναφέρονται ακριβώς είτε παραφρασμένες. Επίσης βεβαιώνω ότι αυτή η εργασία προετοιμάστηκε από εμένα προσωπικά ειδικά για τη συγκεκριμένη εργασία.

Η έγκριση της διπλωματικής εργασίας από το Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών του Πανεπιστημίου Πελοποννήσου δεν υποδηλώνει απαραιτήτως και αποδοχή των απόψεων του συγγραφέα εκ μέρους του Τμήματος.

Η παρούσα εργασία αποτελεί πνευματική ιδιοκτησία του φοιτητή Ευστάθιου Νικήτα που την εκπόνησε.Στο πλαίσιο της πολιτικής ανοικτής πρόσβασης ο συγγραφέας/δημιουργός εκχωρεί στο Πανεπιστήμιο Πελοποννήσου, μη αποκλειστική άδεια χρήσης του δικαιώματος αναπαραγωγής, προσαρμογής, δημόσιου δανεισμού, παρουσίασης στο κοινό και ψηφιακής διάχυσής τους διεθνώς, σε ηλεκτρονική μορφή και σε οποιοδήποτε μέσο, για διδακτικούς και ερευνητικούς σκοπούς, άνευ ανταλλάγματος και για όλο το χρόνο διάρκειας των δικαιωμάτων πνευματικής ιδιοκτησίας. Η ανοικτή πρόσβαση στο πλήρες κείμενο για μελέτη και ανάγνωση δεν σημαίνει καθ' οιονδήποτε τρόπο παραχώρηση δικαιωμάτων διανοητικής ιδιοκτησίας του συγγραφέα/δημιουργού ούτε επιτρέπει την αναπαραγωγή, αναδημοσίευση, αντιγραφή, αποθήκευση, πώληση, εμπορική χρήση, μετάδοση, διανομή, έκδοση, εκτέλεση, «μεταφόρτωση» (downloading), «ανάρτηση» (uploading), μετάφραση, τροποποίηση με οποιονδήποτε τρόπο, τμηματικά ή περιληπτικά της εργασίας, χωρίς τη ρητή προηγούμενη έγγραφη συναίνεση του συγγραφέα/δημιουργού. Ο συγγραφέας/δημιουργός διατηρεί το σύνολο των ηθικών και περιουσιακών του δικαιωμάτων.

Λίστα εικόνων- σχημάτων/ πινάκων

Κεφάλαιο 2

Κεφάλαιο 3

Κεφάλαιο 4

Κεφάλαιο 5

Abstract

In the modern world, information security has become one of the fundamental requirements. Thus, the advantages of using public key cryptography over private key cryptography include the ease of better key management and increased security. However, due to the complexity of the mathematical algorithms, public key cryptography is slower than traditional secret key cryptography, thus motivating the need to speed up public key cryptosystems.

In the mid-1980s, Miller (1986) and Koblitz (1987) independently proposed an implementation of the elliptic curve cryptosystem. Elliptic curves are a mathematical tool with which known public key cryptosystems can be implemented.

Elliptic curve cryptography has gained great importance due to its limited key size for a high level of security, compared to cryptographic algorithms of the same class. It proves to be an important tool for secure communication in cryptography and provides the same level of security with a smaller key compared to RSA. For example, elliptic curves can have a key length of 160 bits and provide about the same level of security as RSA 1024 bits. Depending on the field of the curves algorithm, and the choice of the appropriate mathematical representation of the curve. Performing calculations on the elliptic curve can be very easy or extremely complex.

This thesis presents a description of several important topics for the elliptic curve algorithm, and also how it can be developed on an FPGA. In order to properly manage material resources in these technologies. The paper begins with an introduction to the most basic concepts of cryptography and moves on to the elliptic curves and algorithms required to implement ECCs. That is, the mathematical background, on which the development of the elliptic curve encryption algorithm is based, and how this algorithm managed to be the "key" for modern encryption systems. Expand to the analysis of FPGA technology and their individual components. Closing with the most basic part of the work with the process and the analysis of the implementation of the elliptic curves at the hardware level.

Keywords: Public key cryptography, Private key cryptography, Elliptic curve cryptography, FPGA

Περίληψη

Στον σύγχρονο κόσμο, η ασφάλεια των πληροφοριών έχει γίνει μια από τις θεμελιώδεις απαιτήσεις. Έτσι τα πλεονεκτήματα της χρήσης κρυπτογραφίας δημόσιου κλειδιού έναντι της κρυπτογραφίας ιδιωτικού κλειδιού περιλαμβάνουν την ευκολία της καλύτερης διαχείρισης κλειδιών και την αυξημένη ασφάλεια. Ωστόσο, λόγω της πολυπλοκότητας των μαθηματικών αλγορίθμων, η κρυπτογράφηση δημόσιου κλειδιού είναι πιο αργή από τη συμβατική κρυπτογραφία μυστικού κλειδιού, υποκινώντας έτσι την ανάγκη επιτάχυνσης των κρυπτοσυστημάτων δημόσιου κλειδιού.

Στα μέσα της δεκαετίας του 1980 προτάθηκε από τους Miller (1986) και Koblitz (1987), η εφαρμογή του κρυπτοσυστήματος των ελλειπτικών καμπυλών. Οι ελλειπτικές καμπύλες αποτελούν ένα μαθηματικό εργαλείο με το οποίο μπορούν να υλοποιηθούν γνωστά κρυπτοσυστήματα δημόσιου κλειδιού.

Η κρυπτογραφία των ελλειπτικών καμπυλών έχει αποκτήσει μεγάλη σημασία χάρη στο μικρότερο μέγεθος κλειδιού της για υψηλό επίπεδο ασφάλειας, σε σύγκριση με τους κρυπτογραφικούς αλγόριθμους της ίδιας κατηγορίας. Αποδεικνύεται ότι είναι ένα σημαντικό εργαλείο για ασφαλή επικοινωνία στην κρυπτογραφία και παρέχει το ίδιο επίπεδο ασφάλειας με μικρότερο κλειδί σε σύγκριση με το RSA. Για παράδειγμα οι ελλειπτικές καμπύλες μπορούν να έχουν μήκος 160 bit κλειδί και παρέχουν περίπου το ίδιο επίπεδο ασφάλειας με το RSA 1024 bit. Ανάλογα με το πεδίο εφαρμογής του αλγόριθμου των καμπυλών, και την επιλογή της κατάλληλης μαθηματικής παράστασης της καμπύλης. Η διεξαγωγή υπολογισμών στην ελλειπτική καμπύλη μπορεί να είναι πολύ εύκολη ή εξαιρετικά περίπλοκη.

Αυτή η πτυχιακή εργασία παρουσιάζει τη περιγραφή πολλών σημαντικών θεμάτων για τον αλγόριθμό των ελλειπτικών καμπυλών, αλλά και πως αυτή μπορεί να αναπτυχθεί σε ένα FPGA, με σκοπό την σωστή διαχείριση των πόρων υλικού στις τεχνολογίες αυτές. Η εργασία ξεκινά με μια εισαγωγή στις βασικότερες έννοιες της κρυπτογραφίας και περνάει στις ελλειπτικές καμπύλες και στους αλγόριθμους που απαιτούνται για την υλοποίηση της κρυπτογράφησης των ελλειπτικών καμπυλών. Δηλαδή το μαθηματικό υπόβαθρο, με το οποίο βασίζεται η ανάπτυξη του αλγόριθμου κρυπτογράφησης των ελλειπτικών καμπύλων, και πώς ο αλγόριθμος αυτός κατάφερε να αποτελεί το «κλειδί» για τα σύγχρονα συστήματα κρυπτογράφησης. Επεκτείνετε στην ανάλυση της τεχνολογίας των FPGA και των επιμέρους στοιχείων από αυτά. Κλείνοντας με το βασικότερο κομμάτι της εργασίας με την ανάλυση και την διαδικασία της υλοποίησης των ελλειπτικών καμπύλων σε επίπεδο hardware.

Λέξεις Κλειδιά: Κρυπτογραφίας δημόσιου κλειδιού, Κρυπτογραφίας ιδιωτικού κλειδιού, Ελλειπτικές Καμπύλες, FPGA.

Περιεχόμενα

Συντομογραφίες

AES	Advanced Encryption Standard
DES	Data Encryption Standard
ECC	Elliptic Curve Cryptography
DLP	Discrete Logarithm Problem
RSA	Rivest-Shamir-Adleman
gcd	Greatest Common Divisor
ECDLP	Elliptic Curve Discrete Logarithm Problem
NIST	National Institute of Standard and Technology
DSA	Digital Signature Algorithm
ECDSA	Elliptic Curve Digital Signature Algorithm
MPL	Montgomery powering ladder
SPA	Simple Power Analysis
FPGA	Field Programmable Gate Array
LUT	Look Up Table
CLB	Configurable Logic Block

Σύμβολα

$\mathbb{F}$	Πεπερασμένο πεδίο αριθμών.
$\mathbb{R}$	Πεδίο πραγματικών αριθμών.
$\mathbb{Z}_p$	Πεπερασμένο πεδίο τάξης πρώτων αριθμών p.

$\mathbf{E_k}$ Κλειδί κρυπτογράφησης.

$\mathbf{D_k}$ Κλειδί αποκρυπτογράφησης.

Κεφάλαιο 1

1.1 Εισαγωγή

Η κρυπτογραφία έχει λάβει σημαντική θέση στον σύγχρονο κόσμο μας, με την άνοδο των υπολογιστικών και δικτυακών υποδομών. Κατά την μετάδοση ευαίσθητων πληροφοριών ώστε να διατηρούνται αυτές οι πληροφορίες μυστικές και ασφαλές, έχει σαν αποτέλεσμα την άνοδο του κλάδου της κρυπτογραφίας και ασφάλειας υλικού τις τελευταίες δεκαετίες.

Η κρυπτολογία είναι το πεδίο έρευνας που διερευνά τις μεθόδους για τον τρόπο επίτευξης της εμπιστευτικότητας, της ακεραιότητας της αυθεντικότητας και την μη-απάρνηση των πληροφοριών.

Η εμπιστευτικότητα έχει να κάνει με την αποτροπή της αποκάλυψης πληροφοριών, που συνήθως επιτυγχάνεται με την κρυπτογράφηση μιας πληροφορίας με ένα μυστικό κλειδί χρησιμοποιώντας συμμετρική κρυπτογράφηση, όπως ο AES. Ο όρος συμμετρικό αναφέρεται ότι και τα δύο μέρη που επικοινωνούν χρησιμοποιούν το ίδιο κλειδί για την κρυπτογράφηση και την αποκρυπτογράφηση πληροφοριών.

Ο όρος ακεραιότητα αναφέρεται στην έννοια ότι θα πρέπει να είναι αδύνατο να τροποποιηθεί ένα μήνυμα σε μια συνομιλία χωρίς να το αντιληφθεί κανένας.

Η αρχή της αυθεντικότητας συνεπάγεται ότι ένα μήνυμα προήλθε όντως από ένα συγκεκριμένο μέρος, κάτι που συνήθως γίνεται μέσω κάποιας υπογραφής που βασίζεται στην ασύμμετρη κρυπτογραφία. Το τελευταίο σημαίνει ότι χρησιμοποιούνται διαφορετικά κλειδιά τόσο για την κρυπτογράφηση όσο και για την αποκρυπτογράφηση ενός μηνύματος, που συχνά αναφέρεται ως ιδιωτικό και δημόσιο κλειδί.

Τέλος η μη-απάρνηση είναι η υπηρεσία κατά την οποία ο παραλήπτης ή αποστολέας δεν μπορεί να απαρνηθεί ότι έλαβε ή έστειλε το μήνυμα αντίστοιχα.

1.2 Εισαγωγή στην κρυπτογραφία

Η κρυπτογραφία είναι η μελέτη τεχνικών ασφαλούς επικοινωνίας που επιτρέπουν μόνο στον αποστολέα και τον παραλήπτη του μηνύματος να δει το περιεχόμενό του. Ουσιαστικά είναι η επιστήμη που έχει αντικείμενο μελέτης το σύνολο των τεχνικών, που έχει σαν σκοπό στην ασφαλή μετάδοση ενός μηνύματος.

Βεβαία η κρυπτογραφία (cryptography) αποτελεί το ένα από τα δύο σκέλη, της βασικής εννοίας της κρυπτολογίας (cryptology). Το δεύτερο σκέλος είναι κρυπτανάλυση (cryptanalysis). Το τελευταίο είναι η επιστήμη που ασχολείται με την αποκρυπτογράφηση του κρυπτοκειμένου χωρίς την κατοχή του κλειδιού, με βάση του οποίου πραγματοποιήθηκε και το κρυπτογραφημένο μήνυμα.

Τα δύο σκέλη απαρτίζουν τον επιστημονικό κλάδο της κρυπτολογίας (cryptology), η οποία ασχολείται με τη μελέτη και τη σχεδίαση κρυπτογραφικών τεχνικών.

Σε οποιαδήποτε επικοινωνία, τα δύο μέρη που συμμετέχουν σε συνομιλία είναι συνήθως γνωστά ως Αλίκη και Μπομπ. Σε αυτήν τη συνομιλία, το αρχικό μήνυμα ονομάζεται καθαρό κείμενο ή απλό κείμενο που συμβολίζεται ως P. Υπάρχουν δύο τρόποι κρυπτογράφησης του απλού κειμένου, ο ένας είναι ο κρυπτογράφησης μπλοκ (Block Ciphers) και ο άλλος κρυπτογράφησης ροής (Stream Ciphers) [1]. Η κρυπτογράφηση σε μπλοκ «τεμαχίζει» το αρχικό κείμενο σε μπλοκ, παίρνει ένα μπλοκ στοιχείων ως είσοδο και παράγει ένα μπλοκ εξόδου για κάθε μπλοκ εισόδου. Ενώ η κρυπτογράφησης ροής λαμβάνει στοιχεία εισόδου συνεχώς και παράγει έξοδο για ένα στοιχείο τη φορά.

Το απλό κείμενο δεν αποστέλλεται απευθείας στον δέκτη, αλλά μετατρέπεται σε κωδικοποιημένη μορφή που δεν είναι κατανοητή. Η κωδικοποιημένη μορφή στην κρυπτογραφία είναι γνωστή ως κρυπτογραφημένο κείμενο (ciphertext). Το απλό κείμενο διέρχεται από κάποια διαδικασία μέσω ενός αλγορίθμου για τη μετατροπή του σε κρυπτογραφημένο κείμενο. Αυτός ο αλγόριθμος ονομάζεται αλγόριθμος κρυπτογράφησης.

Για να γίνει το κρυπτογραφημένο κείμενο αναγνώσιμο, χρησιμοποιείται και πάλι ένας αλγόριθμος για τη μετατροπή του σε απλό κείμενο. Αυτός ο αλγόριθμος είναι γνωστός ως αλγόριθμος αποκρυπτογράφησης.

Για την κρυπτογράφηση και την αποκρυπτογράφηση του μηνύματος, και οι δύο αλγόριθμοι απαιτούν μια μυστική πληροφορία/συνάρτηση που είναι γνωστή ως κλειδί. Η εγγύηση μιας

ασφαλούς επικοινωνίας εξαρτάται σε μεγάλο βαθμό από το κλειδί. Εάν ένα τρίτο άτομο λάβει τη γνώση του κλειδιού, τότε η ασφάλεια της συνομιλίας μπορεί να διαρρεύσει.

1.2.1 Συμμετρική κρυπτογραφία

Η κρυπτογραφία συμμετρικού κλειδιού είναι μια μέθοδος που διασφαλίζει το απόρρητο και την ασφάλεια της επικοινωνίας. Όταν δύο μέρη ας πούμε, η Αλίκη και ο Μπομπ επικοινωνούν μέσω ενός μη ασφαλούς καναλιού χρησιμοποιώντας κάποιο πρωτόκολλο συμμετρικού κλειδιού. Τότε το κλειδί που χρησιμοποιείται για την κρυπτογράφηση και την αποκρυπτογράφηση είναι το ίδιο. Μπορεί να υπάρχουν περιπτώσεις όπου το κλειδί αποκρυπτογράφησης μπορεί να ληφθεί εύκολα από το κλειδί κρυπτογράφησης.

Σε αυτό το πρωτόκολλο απαιτείται η χρήση ασφαλούς καναλιού για την ανταλλαγή του κλειδιού. Επειδή, ο Μπομπ δεν μπορεί να αποκρυπτογραφήσει το μήνυμα εκτός και αν, μέχρι τότε γνωρίζει το κλειδί. Έτσι, η Αλίκη αφού κρυπτογραφήσει το μήνυμα στέλνει το κλειδί μέσω κάποιου ασφαλούς καναλιού το οποίο στη συνέχεια του επιτρέπει να αποκρυπτογραφήσει το μήνυμα. Για παράδειγμα, δημοφιλή συμμετρική κρυπτογράφηση περιλαμβάνουν AES [2], DES [3].

1.2.2 Ασύμμετρη κρυπτογραφία

Η ασύμμετρη κρυπτογραφία κλειδιού επινοήθηκε το 1976 από τους Whitfield Diffie και Martin Hellman. Η κρυπτογραφία συμμετρικού κλειδιού έχει ένα μειονέκτημα ότι μοιράζεται ένα κλειδί και επομένως επινοήθηκε η κρυπτογραφία ασύμμετρου κλειδιού. Είναι μια μέθοδος που χρησιμοποιείται για ασφαλή επικοινωνία. Χρησιμοποιεί δύο κλειδιά, το ένα για κρυπτογράφηση και το άλλο για αποκρυπτογράφηση. Το κλειδί που χρησιμοποιείται για την κρυπτογράφηση είναι γνωστό ως δημόσιο κλειδί του κατόχου (δέκτης) και το κλειδί αποκρυπτογράφησης είναι γνωστό ως ιδιωτικό κλειδί του κατόχου. Δεν απαιτεί ασφαλές κανάλι για την ανταλλαγή κλειδιού. Όλοι έχουν πρόσβαση στο δημόσιο κλειδί αλλά το ιδιωτικό κλειδί είναι γνωστό μόνο στον κάτοχο του (δέκτη).

Ας υποθέσουμε ότι η Αλίκη και ο Μπομπ θέλουν να επικοινωνήσουν μέσω ενός μη ασφαλούς καναλιού χρησιμοποιώντας το ασύμμετρο πρωτόκολλο. Στη συνέχεια, η Αλίκη αποκτά πρώτα το δημόσιο κλειδί του Μπομπ και το χρησιμοποιεί για να κρυπτογραφήσει το μήνυμα. Το κρυπτογραφημένο κείμενο αποστέλλεται στον Μπομπ. Στη συνέχεια, ο Μπομπ χρησιμοποιεί το

ιδιωτικό του κλειδί για να αποκρυπτογραφήσει το κρυπτογραφημένο κείμενο σε απλό κείμενο. Εδώ μπορούμε να παρατηρήσουμε ότι δεν υπάρχει ανάγκη ασφαλούς καναλιού για την ανταλλαγή κλειδιού. Ως εκ τούτου, λύνει το ζήτημα της χρήσης ασφαλούς καναλιού για τη μεταφορά κλειδιού. Για παράδειγμα, μερικά διάσημα πρωτόκολλα ασύμμετρων κλειδιών περιλαμβάνουν το πρωτόκολλο ανταλλαγής κλειδιών Diffie-Hellman: ElGamal, RSA, Ελλειπτικές καμπύλες (ECC) [4].

1.2.3 Κρυπτογραφία και ελλειπτικές καμπύλες

Στον κόσμο της επιστήμης και της τεχνολογίας χρησιμοποιούνται διαφορετικοί τρόποι επικοινωνίας. Η επικοινωνία με ασφαλή τρόπο είναι ένας από τους θεμελιώδεις στόχους σήμερα. Έτσι, για την προστασία των πληροφοριών χρησιμοποιούνται κρυπτογραφικά εργαλεία, ώστε να διασφαλίζεται ότι δεν διακυβεύεται η ασφάλεια του μηνύματος. Μια πρόσφατη νέα προσέγγιση χρήσης κρυπτογραφίας δημόσιου κλειδιού βασίζεται σε ελλειπτικές καμπύλες. Το επόμενο κεφάλαιο είναι αφιερωμένο στην κρυπτογραφία ελλειπτικών καμπυλών.

Κεφάλαιο 2

2.1 Εισαγωγή ECC

Αρχικά οι αλγόριθμοι βασίζονταν σε προβλήματα παραγοντοποίησης ακεραίων ή το πρόβλημα διακριτού λογαρίθμου (Discrete Logarithm Problem – DLP). Μερικοί σημαντικοί αλγόριθμοι περιλαμβάνουν το πρωτόκολλο ανταλλαγής κλειδιών RSA και Diffie Hellman.

Ο RSA προτάθηκε από τους R.Rivest, A.Shamir και L.Adleman το 1976 και βασίζεται στο πρόβλημα παραγοντοποίηση ακεραίων. Επίσης και ο Diffie-Hellman βασίζεται στο πρόβλημα διακριτού λογάριθμου που προτάθηκε την ίδια χρονιά 1976. Στο πρόβλημα παραγοντοποίησης ακεραίων επιλέγονται δύο μεγάλοι πρώτοι αριθμοί και στη συνέχεια πολλαπλασιάζονται για να ληφθεί ένας ακέραιος. Οι αλγόριθμοι που βασίζονται στο πρόβλημα παραγοντοποίησης ακεραίων εξαρτώνται από τη δυσκολία του προϊόντος που παραγοντοποιείται.

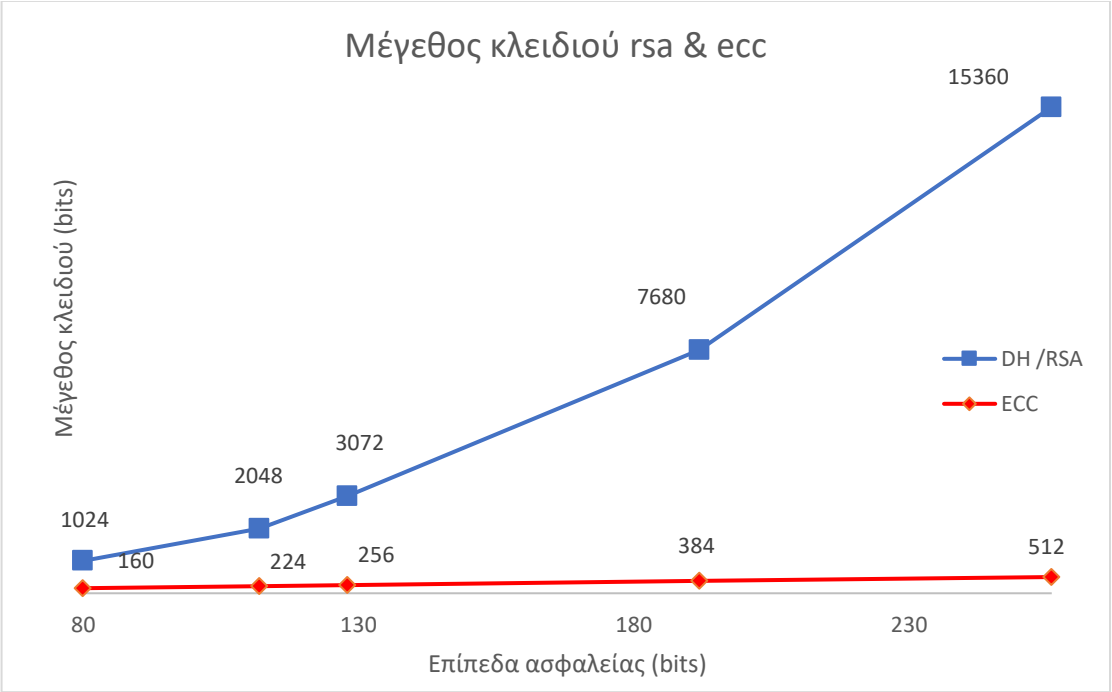
Το πρόβλημα διακριτού λογάριθμου ορίζεται χρησιμοποιώντας στοιχεία σε κυκλικής ομάδας με modulo arithmetic. Έστω g μια γεννήτρια πολλαπλασιαστικής κυκλικής ομάδας $\mathbb{Z}_p$ όπου p είναι πρώτος. Ξέρουμε ότι $g^a = b \in \mathbb{Z}_p$. Τότε το πρόβλημα διακριτού λογαρίθμου είναι να βρεθεί το "a" όταν είναι γνωστή μόνο τα "g" και "b".

Το μέγεθος κλειδιού που έχει προτείνει το Εθνικό Ινστιτούτο Προτύπων και Τεχνολογίας (NIST) για το DLP είναι 1024 bit που σημαίνει ότι, πρέπει να εργαστεί κανείς με τουλάχιστον 1024 bit για να εξασφαλίσει ασφαλή επικοινωνία. Λόγω ενός πολύ μεγάλου μεγέθους κλειδιού οι υπολογισμοί χρειάζονται πολύ χρόνο για να εκτελεστούν.

ECC vs RSA μέγεθος κλειδιού **[5]**

Bits ασφάλειας	DH ή RSA	ECC
80	**1024**	**160**
112	**2048**	**224**
128	**3072**	**256**
192	**7680**	**384**
256	**15360**	**521**

Πίνακας 2.1: Προτεινόμενο μέγεθος κλειδιού συμφωνά με τον NIST



Εικόνα 2.1: Προτεινόμενο μέγεθος κλειδιού συμφωνά με τον NIST

Συνεπώς, χρειαζόταν βελτιωμένη προσέγγιση στην κρυπτογραφία ώστε να μειωθεί το μέγεθος του κλειδιού, αλλά διατηρώντας τα ίδια επίπεδα ασφάλειας. Στη συνέχεια εισήχθη η χρήση της ελλειπτικής καμπύλης και παρατηρήθηκε ότι το πρόβλημα του διακριτού λογαρίθμου μπορεί να γίνει πιο δύσκολο εάν εφαρμοστεί πάνω στην ελλειπτική καμπύλη.

Το κύριο πλεονέκτημα της χρήσης των ελλειπτικών καμπυλών είναι ότι το ίδιο επίπεδο ασφάλειας μπορεί να επιτευχθεί με μικρότερο μήκος κλειδιού και λύνει το πρόβλημα της πολυπλοκότητας για την επίτευξη της επιθυμητής ασφάλειας (Πίνακας 2.1 & Εικόνα 2.1). Στην επόμενη ενότητα, θα δούμε αναλυτικά την ελλειπτική καμπύλη και θα μιλήσουμε για το πώς δημιουργούνται

2.2 Ελλειπτικές καμπύλες κρυπτογράφησης

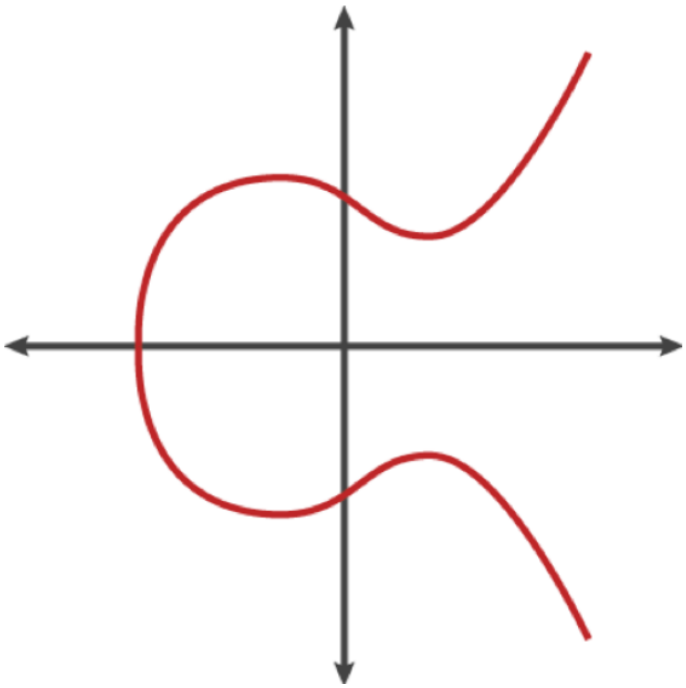
Τα κρυπτοσυστήματα ελλειπτικών καμπυλών δεν είναι νέα κρυπτοσυστήματα. Οι ελλειπτικές καμπύλες αποτελούν ένα μαθηματικό εργαλείο που ανήκουν στην κατηγορία της ασύμμετρης κρυπτογραφίας, γνωστή ως κρυπτογραφία δημοσίου κλειδιού. Η εφαρμογή των ελλειπτικών καμπυλών στην κρυπτογραφία προτάθηκε από τους Miller (1986) και Koblitz (1987). Ο αλγόριθμος αυτός είναι βασισμένος στις μαθηματικές ελλειπτικές καμπύλες, οι οποίες είναι γνωστές για την παραγωγή, ταχύτερων και αποδοτικότερων κρυπτογραφικών κλειδιών.

Συγκριτικά με άλλους κρυπτογραφικούς μεθόδους όπως, Diffie-Hellman και RSA βασίζονται στη δημιουργία κλειδιών χρησιμοποιώντας πολύ μεγάλους πρώτους αριθμούς. Ως εκ τούτου, η δημιουργία κλειδιού απαιτεί μεγάλη υπολογιστική ισχύ.

Οι ελλειπτικές καμπύλες μπορούν να ορισθούν σε διάφορα σώματα, όπως το σώμα των πραγματικών, των μιγαδικών, κτλ. Ειδικότερα στην κρυπτογραφία, οι ελλειπτικές καμπύλες ορίζονται σε πεπερασμένα σώματα. Οι (ECC) για τον λόγο ότι βασίζονται στη θεωρία ελλειπτικών καμπυλών παράγουν κλειδιά μέσω των ιδιοτήτων της εξίσωσης της καμπύλης. Αντί της παραδοσιακής μεθόδου παραγωγής ως γινόμενο πολύ μεγάλων πρώτων αριθμών. Σύμφωνα με ορισμένους ερευνητές, το ECC μπορεί να επιτύχει το ίδιο επίπεδο ασφάλειας με ένα κλειδί 160-bit σε αντίθεση με τα άλλα συστήματα που απαιτούν κλειδί 1024-bit.

Επειδή το ECC βοηθά στη δημιουργία ισοδύναμης ασφάλειας με χαμηλότερη υπολογιστική ισχύ και χαμηλότερη κατανάλωση ενέργειας, χρησιμοποιείται ευρέως για εφαρμογές σε ενσωματωμένα συστήματα, συστήματα χαμηλότερη κατανάλωση ενέργειας και σε εφαρμογές IoT.

Μια ελλειπτική καμπύλη είναι το σύνολο σημείων που ικανοποιούν μια συγκεκριμένη μαθηματική εξίσωση. Η εξίσωση για μια ελλειπτική καμπύλη μοιάζει με αυτή $y^2 = x^3+ax+b$ και παριστάνεται γραφικά όπως η παρακάτω εικόνα.



Εικόνα 2.2: Ελλειπτική καμπύλη $y^2 = x^3+ax+b$

Η εξίσωση της μορφής:

$$Y^2 + a_1XY + a_3Y = X^3 + a_2X^2 + a_4X + a_6 \quad (3.1)$$

Ορίζεται στο πεπερασμένο πεδίο $\mathbb{F}$ είναι μια επίπεδη καμπύλη και γνωστή ως εξίσωση Weierstass [6]. Όπου τα a_i ονομάζονται συντελεστές Weierstass και επιλέγονται από το πεδίο $\mathbb{F}$, θεωρούμε κυβικές εξισώσεις αυτής της μορφής.

- $b_2 = a_1^2 + 4a_2$
- $b_4 = 2a_4 + a_1a_3$
- $b_6 = a_3^2 + 4a_6$
- $b_8 = a_1^2a_6 + 4a_2a_6 - a_1a_3a_4 + a_2a_3^2 - a_4^2$

2.2.1 Ελλειπτικές καμπύλες στο σώμα των πραγματικών αριθμών $\mathbb{R}$

Στην κρυπτογραφία γενικά η εστίασή μας είναι στην απλοποιημένη μορφή της εξίσωσης Weierstrass η οποίο είναι:

$$\mathbf{y^2 = x^3+ax+b} \quad (2.2)$$

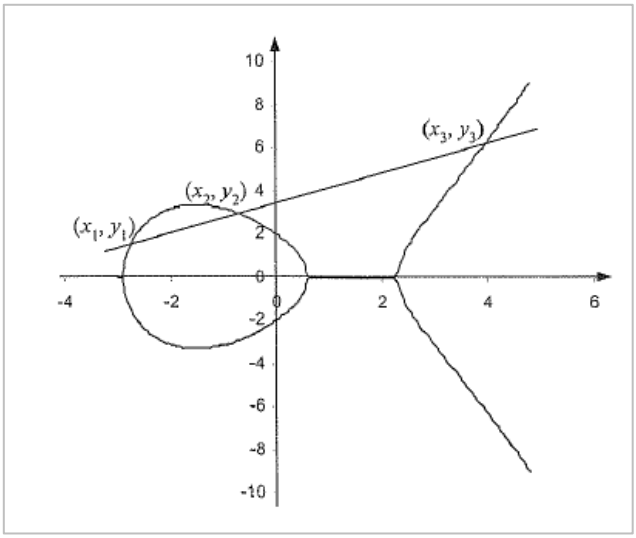
Όπου τα a και b ονομάζονται συντελεστές Weierstass και επιλέγονται από το πεδίο $\mathbb{F}$. Η καμπύλη Weierstrass ονομάζεται ελλειπτική καμπύλη, εάν το πεδίο $\mathbb{F} = \mathbb{R}$. Στην περίπτωση της ελλειπτικής καμπύλης, η εξίσωση της καμπύλης είναι δευτεροβάθμια ως προς το y αλλά τριτοβάθμια ως προς το x. Θεωρούμε δύο διαφορετικά σημεία (x_1, y_1) και (x_2, y_2) της ελλειπτικής καμπύλης, και έστω η ευθεία:

$$\mathbf{y = \lambda x + c}$$

Η οποία τέμνει την καμπύλη στα σημεία αυτά. Αντικαθιστώντας την εξίσωση της ευθείας στην ελλειπτική καμπύλη, θα είναι:

$$\mathbf{(\lambda x + c)^2 = x^3 + ax + b}$$

Η οποία είναι τριτοβάθμια εξίσωση με δύο από τις ρίζες τα x_1 και x_2. Υπάρχει και η ρίζα x_3, που αντιστοιχεί στο σημείο της ευθείας $(x_3, \lambda x_3 + c)$. Τα σημεία τομής είναι τρία μεταξύ της καμπύλης με την ευθεία.



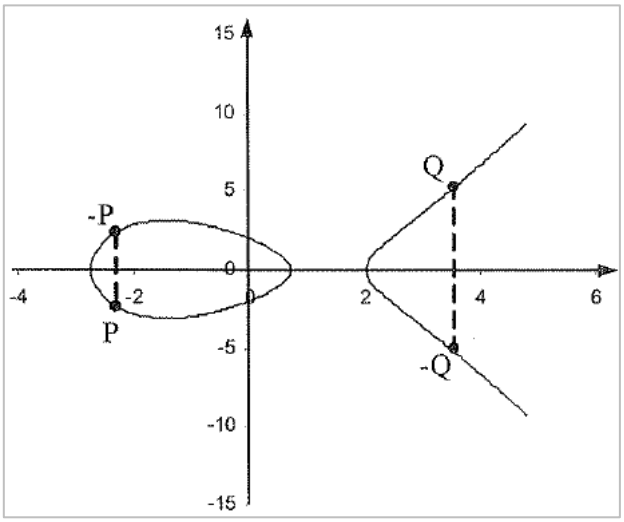
Εικόνα 2.3: Ελλειπτική καμπύλη $y^2 = x^3 - 6x + 4$

Για κάποιον συνδυασμό των a και b, η εξίσωση της ελλειπτικής καμπύλης δεν έχει τρεις διαφορετικές ρίζες (για y = 0). Έτσι η απλοποιημένη μορφή της (2.1) είναι $4a^3 – 27b^2$. Αυτή η καμπύλη λέγεται ομαλή εάν $4a^3 – 27b^2 \neq 0$.

Ας υποθέσουμε ότι έχουμε δύο σημεία P και Q που βρίσκονται σε μια ελλειπτική καμπύλη E: $y^2 = x^3 – 6x + 4$ και επιθυμούμε να τα προσθέσουμε όπως θα δούμε παρακάτω.

2.2.1.1 Πρόσθεση σημείων ελλειπτικής καμπύλης

Η πρόσθεση βασίζεται στο γεγονός ότι μια ευθεία μπορεί να τέμνει μια ελλειπτική καμπύλη σε τρία το πολύ σημεία. Μια ελλειπτική καμπύλη είναι συμμετρική ως προς τον άξονα x'x. Έτσι ορίζουμε το αντίθετο σημείο (-P) ενός σημείου (P) όπως φαίνεται στο παρακάτω σχήμα.



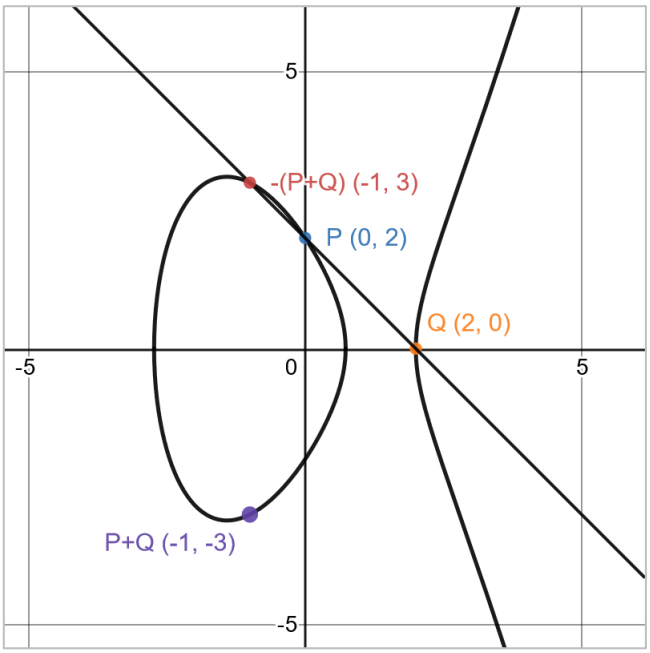
Εικόνα 3.4: Ορισμός αντιθέτου

Βλέπουμε ότι το P = (x, y), συνεπώς το αντίθετο σημείο είναι –P = (x, –y). Από την σκοπιά της γεωμετρίας αυτό περιγράφεται ως εξής. Για αρχή υπολογίζουμε την ευθεία που τέμνει το σημείο P (x, y) και το σημείο **O** (κατακόρυφη). Το τρίτο σημείο της καμπύλης είναι το –P.

Επομένως το ουδέτερο στοιχείο στην πρόσθεση σημείων ελλειπτικής καμπύλης είναι το σημείο **O**:

$$\mathbf{P + O = O + P = P} \text{ και } \mathbf{P + (–P) = O}$$

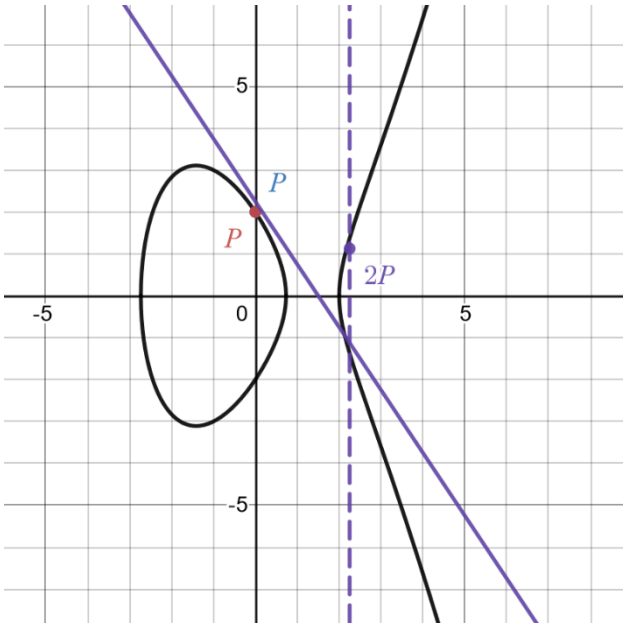
Έστω τα σημεία P = (0, 2) και Q = (2, 0) της ελλειπτικής καμπύληςE: $y^2 = x^3 – 6x + 4$ προβάλουν τα αποτέλεσμα της πρόσθεσης των σημείων στο S = P + Q = (-1, -3), όπως προβάλλεται στο παρακάτω σχήμα.



Εικόνα 2.5: Πρόσθεση σημείων, P + Q της E: $y^2 = x^3 – 6x + 4$

Η ευθεία η οποία διέρχεται από τα P και Q, τέμνει την καμπύλη στο τρίτο σημείο το οποίο είναι το **- (P + Q)** το σημείο (P + Q) θα είναι το συμμετρικό του - (P + Q) ως προς τον άξονα x.

Εάν στην περίπτωση που P = Q, θεωρούμε ότι τα δύο από τα τρία σημεία που τέμνουν την καμπύλη συμπίπτουν.



Εικόνα 2.6: Πρόσθεση σημείων, 2P της E: $y^2 = x^3 - 6x + 4$

Η ευθεία που ορίζεται είναι εφαπτόμενη στο σημείο P. Το δεύτερο σημείο που τέμνει η καμπύλη είναι το -2P δηλαδή το συμμετρικό ως προς τον άξονα x το 2P. Το αποτέλεσμα της πράξεις είναι ο διπλασιασμός του σημείου P.

Είναι πλέον σαφές πώς συμπεριφέρονται τα σημεία στην ελλειπτική καμπύλη όταν προστίθενται. Έχοντας υπόψη την παραπάνω γραφική αναπαράσταση της πρόσθεσης σημείων, βλέπουμε στη συνέχεια τη μαθηματική αναπαράστασή της.

2.2.1.2 Μαθηματική προσέγγιση για πρόσθεση σημείων στην καμπύλη

Με αφορμή ότι είδαμε κατά τον γραφικό υπολογισμό, των δύο σημείων καθώς και το αντίθετο του αθροίσματος αυτών βρίσκονται στην ίδια ευθεία. Έστω ότι η ευθεία αυτή περνάει από το P = (x_1 + y_1) και Q = (x_2 + y_2), τότε η ευθεία:

$$\boldsymbol{y = \lambda x + c}$$

Μένει μόνο να ορίσουμε την κλήση (λ).

- Αν P και Q είναι δύο διαφορετικά σημεία τότε η κλήση είναι:

$$\boldsymbol{\lambda = \frac{y_2 - y_1}{x_2 - x_1}} \qquad (2.3)$$

- Αν P και Q συμπίπτουν τότε η κλήση είναι:

$$\boldsymbol{\lambda = \frac{3x_1^2 + a}{2y_1}} \qquad (2.4)$$

Ότι αφορά το νέο σημείο P + Q = (x_3, y_3) από την πρόσθεση των σημείων P και Q, προκύπτει αν στην εξίσωση (3.2) θέσουμε όπου y την εξίσωση της ευθείας.

$$\boldsymbol{x_3 = \lambda^2 - x_1 - x_2} \qquad (2.5)$$

$$\boldsymbol{y_3 = \lambda(x_1 - x_3) - y_1} \qquad (2.6)$$

Τέλος στη περίπτωση όπου το Q = –P = (x_1, $-y_1$), η κλίση γίνεται άπειρη, γεγονός που μας οδηγεί στο σημείο **O**.

2.2.2 Ελλειπτικές καμπύλες ορισμένες σε πεπερασμένα πεδία στο $\mathbb{F}$

Όταν έχουμε να κάνουμε με ελλειπτική καμπύλη πάνω από πραγματικούς αριθμούς, η γραφική απεικόνιση δείχνει μια ομαλή καμπύλη.

Για να ορίσουμε μια καμπύλη σε πεπερασμένο πεδίο, πρέπει να χρησιμοποιήσουμε αρθρωτή αριθμητική (modular arithmetic). Τώρα, η καμπύλη στην εξίσωση (2.2) έχει την ακόλουθη μορφή:

$$y^2 = x^3 + ax + b \bmod p \qquad (2.7)$$

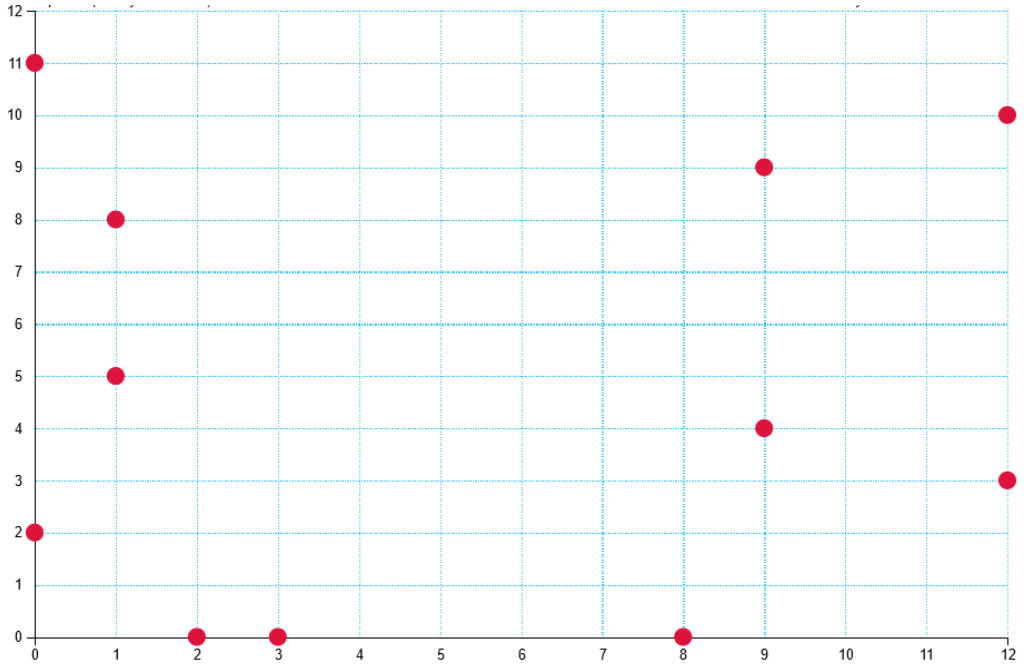
που ορίζεται σε ένα πεπερασμένο πεδίο $\mathbb{F}_p$ και p > 3 είναι πρώτος άρτιος αριθμός.

Οι συντελεστές a, b και οι μεταβλητές x, y προέρχονται από το πεδίο $\mathbb{F}_p$. Το ζευγάρι (x, y) από την εξίσωση (3.7) βρίσκονται πάνω στην ελλειπτική καμπύλη.

Αν θέλουμε να δούμε την γεωμετρική συμπεριφορά της ελλειπτικής καμπύλης σε ένα πεπερασμένο πεδίο $\mathbb{F}_p$, αυτή την φορά δεν βλέπουμε καμία ομαλή καμπύλη, όπως είδαμε στην ενότητα με την καμπύλη στο πεδίο $\mathbb{R}$. Αλλά τα διακριτά σημεία εμφανίζονται στο γράφημα.

Ας εξετάσουμε την καμπύλη για $\mathbb{F}_{13}$ E: $y^2 = x^3 + 7x + 4 \bmod 13$ (2.8). Να σημειωθεί ότι η E: $y^2 = x^3 - 6x + 4$ που είχαμε δει νωρίτερα έχει την ίδια συμπεριφορά με την (2.8)

Στο παρακάτω γράφημα παρατηρούμε τα διακριτά σημεία της καμπύλης.



Εικόνα 2.7: Ελλειπτική καμπύλη E: $y^2 = x^3 + 7x + 4$ στο πεδίο $\mathbb{F}_{13}$

Τα σημεία που βρίσκονται στη δεδομένη καμπύλη φαίνονται στον παρακάτω πίνακα

x	y^2	$y_{1,2}$	P(x, y)	Q(x, y)
0	4	2, 11	(0, 2)	(0, 11)
1	**12**	**5, 8**	**(1, 5)**	**(1, 8)**
2	0	0	(2, 0)	-
3	0	0	(3, 0)	-
4	5	-	-	-
5	8	-	-	-
6	2	-	-	-
7	6	-	-	-
8	0	0	(8, 0)	-
9	3	4, 9	(9, 4)	(9, 9)
10	8	-	-	-
11	8	-	-	-
12	9	3, 10	(12, 3)	(12, 10)

Πίνακας 2.2: Σημεία της καμπύλης E στο πεδίο $\mathbb{F}_{13}$

Τα στοιχεία του πίνακα προκύπτουν από την εξίσωση (2.8). Για κάθε x, δημιουργείτε ένα y^2 και στην συνέχεια οι συντεταγμένες για καθένα ζεύγος P(x, y) και Q(x, y).

Για παράδειγμα για x = 1 από την εξίσωση (2.8) έχω:

$$x \equiv 1 \Rightarrow y^2 \equiv 12 \equiv 5(mod\ 13) \Rightarrow 5, 8\ (\text{mod}13)$$

Επομένως το σημείο της Ε είναι το (1, 5) και (1, 8).

Η ίδια καμπύλη που εμφάνιζε μια ομαλή καμπύλη σε πραγματικούς αριθμούς εμφανίζει τώρα διακριτά σημεία που οφείλεται μόνο στο πεπερασμένο πεδίο $\mathbb{F}_{\mathbf{13}}$. Τα σημεία με κόκκινο χρώμα στην γραφική είναι τα διατεταγμένα ζεύγη (x, y) που ικανοποιούν τη δεδομένη εξίσωση.

Η πρόσθεση των σημείων μιας ελλειπτικής καμπύλης, modulo p, γίνεται με την χρήση των παραπάνω σχέσεων. Με την παρατήρηση ότι ένας ρητός αριθμός $\frac{\alpha}{\beta}$ πρέπει να αντιμετωπιστεί ως $\alpha\beta^{-1}$, όπου $\beta^{-1} * \beta \equiv 1$ (mod p). Αυτό απαιτεί ότι gcd (β, p) = 1 και είναι το σημείο κλειδί για τη χρήση των ελλειπτικών καμπυλών στην παραγοντοποίηση ακεραίων.

- **Για P ≠ Q**

Ας υποθέσουμε ότι θέλουμε να προσθέσουμε τα σημεία P (9, 4) και Q(12, 10) στην ελλειπτική καμπύλη (3.8). Χρησιμοποιώντας τους τύπους στις (3.5) και (3.6) για να πάρουμε τις συντεταγμένες του νέου σημείου R(x3, y3). Αρχικά υπολογίζουμε την κλίση λ σύμφωνα με την (3.3):

$$\boldsymbol{\lambda = \frac{y_2 - y_1}{x_2 - x_1}\ mod\ p}$$

$$\boldsymbol{\lambda = \frac{10 - 4}{12 - 9} mod\ 13}$$

$$\boldsymbol{\lambda = \frac{6}{3} mod\ 13}$$

$$\boldsymbol{\lambda = 6\ (3^{-1}) mod\ 13}$$

Χρησιμοποιώντας την ανεπτυγμένη μορφή του Ευκλείδη [17] παίρνουμε:

$$\mathbf{3^{-1} \equiv 9\ mod\ 13}$$

Έτσι έχουμε

$$\boldsymbol{\lambda = (6 * 9) mod\ 13}$$

$$\boldsymbol{\lambda = 2\ mod\ 13}$$

$$\boldsymbol{\lambda = 2}$$

Στην συνέχεια αντικαθιστούμε την τιμή του λ στις (3.5) και (3.6) ώστε να υπολογίσουμε το $R(x_3, y_3)$ και παίρνουμε ως αποτέλεσμα:

$$\boldsymbol{x_3 = \lambda^2 - x_1 - x_2\ mod\ p}$$

$$\boldsymbol{x_3 = 2^2 - 9 - 12\ mod\ 13}$$

$$\boldsymbol{x_3 = 9\ mod\ 13}$$

$$\boldsymbol{x_3 = 9}$$

$$\boldsymbol{y_3 = \lambda(x_1 - x_3) - y_1\ mod\ p}$$

$$\boldsymbol{y_3 = 2(9 - 9) - 4\ mod\ 13}$$

$$\boldsymbol{y_3 = 9\ mod\ 13}$$

$$\boldsymbol{y_3 = 9}$$

Συνεπώς το σημείο R έχει συντεταγμένες R(9, 9), οι οποίες συνταραγμένες προέκυψαν από την πρόσθεση των σημείων P(9, 4) και Q(9, 9).

- **<u>Για P = Q</u>**

Τώρα αν θέλουμε να προσθέσουμε το σημείο P (9, 4) στον εαυτό του. Για να υπολογίσουμε 2P= P + P, βρίσκουμε πάλι πρώτα το λ χρησιμοποιώντας τον τύπο (3.4)

$$\boldsymbol{\lambda = \frac{3x_1^2 + a}{2y_1} \; mod \; p}$$

$$\boldsymbol{\lambda = \frac{3(9^2) + 7}{2 \cdot 4} \; mod \; 13}$$

$$\boldsymbol{\lambda = 250 \times 8^{-1} \; mod \; 13}$$

$$\boldsymbol{\lambda = 250 \times 5 \; mod \; 13}$$

$$\boldsymbol{\lambda = 2}$$

Οι συντεταγμένες του νέου σημείου από της (3.5) και (3.6) είναι

$$\boldsymbol{x_3 = \lambda^2 - 2 * x_1 \; mod \; p}$$

$$\boldsymbol{x_3 = 2^2 - 2 * 9 \; mod \; 13}$$

$$\boldsymbol{x_3 = 12}$$

$$\boldsymbol{y_3 = \lambda(x_1 - x_3) - y_1 \; mod \; p}$$

$$\boldsymbol{y_3 = 2(9 - 12) - 4 \; mod \; 13}$$

$$\boldsymbol{y_3 = 3}$$

Σαν τελικό αποτέλεσμα από την πρόσθεση του P στον εαυτό του μας δίνει, R(12, 3).

Ομοίως, οποιοδήποτε σημείο της ελλειπτικής καμπύλης μπορεί να προστεθεί σε ένα άλλο σημείο της ελλειπτικής καμπύλης και επίσης, ένα σημείο μπορεί να προστεθεί στον εαυτό του όσες φορές θέλουμε. Ο Πίνακας 3.3 δείχνει την πρόσθεση όλων των σημείων της (3.8).

+	∞	(0,2)	(0,11)	(1,5)	(1,8)	(2,0)	(3,0)	(8,0)	(9,4)	(9,9)	(12,3)	(12,10)
∞	∞	(0,2)	(0,11)	(1,5)	(1,8)	(2,0)	(3,0)	(8,0)	(9,4)	(9,9)	(12,3)	(12,10)
(0,2)	(0,2)	(12,3)	∞	(8,0)	(9,9)	(12,10)	(9,4)	(1,8)	(1,5)	(3,0)	(2,0)	(0,11)
(0,11)	(0,11)	∞	(12,10)	(9,4)	(8,0)	(12,3)	(9,9)	(1,5)	(3,0)	(1,8)	(0,2)	(2,0)
(1,5)	(1,5)	(8,0)	(9,4)	(12,10)	∞	(9,9)	(12,3)	(0,11)	(2,0)	(0,2)	(1,8)	(3,0)
(1,8)	(1,8)	(9,9)	(8,0)	∞	(12,3)	(9,4)	(12,10)	(0,2)	(0,11)	(2,0)	(3,0)	(1,5)
(2,0)	(2,0)	(12,10)	(12,3)	(9,9)	(9,4)	∞	(8,0)	(3,0)	(1,8)	(1,5)	(0,11)	(0,2)
(3,0)	(3,0)	(9,4)	(9,9)	(12,3)	(12,10)	(8,0)	∞	(2,0)	(0,2)	(0,11)	(1,5)	(1,8)
(8,0)	(8,0)	(1,8)	(1,5)	(0,11)	(0,2)	(3,0)	(2,0)	∞	(12,10)	(12,3)	(9,9)	(9,4)
(9,4)	(9,4)	(1,5)	(3,0)	(2,0)	(0,11)	(1,8)	(0,2)	(12,10)	(12,3)	∞	(8,0)	(9,9)
(9,9)	(9,9)	(3,0)	(1,8)	(0,2)	(2,0)	(1,5)	(0,11)	(12,3)	∞	(12,10)	(9,4)	(8,0)
(12,3)	(12,3)	(2,0)	(0,2)	(1,8)	(3,0)	(0,11)	(1,5)	(9,9)	(8,0)	(9,4)	(12,10)	∞
(12,10)	(12,10)	(0,11)	(2,0)	(3,0)	(1,5)	(0,2)	(1,8)	(9,4)	(9,9)	(8,0)	∞	(12,3)

Πίνακας 2.3: Πρόσθεση σημείων (7, 4) της $E_{F_{13}}$

2.3 Πρόβλημα διακριτού λογαρίθμου στις ελλειπτικές καμπύλες ECDLP

Είδαμε προηγούμενος τον τρόπο με τον οποίο γίνεται η πρόσθεση ενός σημείο στον εαυτό του. Δεδομένου ενός σημείου P, μπορούμε να υπολογίσουμε το P + P = 2P, είτε γραφικά με την εφαπτομένη στο P, είτε με τις αλγεβρικές εξισώσεις της πρόσθεσης για P = Q.

Καθώς το πρόβλημα διακριτού λογαρίθμου (discrete logarithm problem – DLP) [7] βασίζεται σε μια κυκλική ομάδα, για μια ελλειπτική καμπύλη που ορίζεται σε 𝔽. Ενα σημείο Q στην ελλειπτική καμπύλη μπορούμε να το υπολογίσουμε προσθέτοντας το P, n φορές στον εαυτό του. Μαθηματικά μπορεί να γραφτεί ως P

$$\mathbf{P + P + P + \ldots + P = nP = Q}$$

Ο τρόπος υπολογισμού του nP είναι παρόμοιος με τον αλγόριθμο «επαναλαμβανόμενου τετραγωνισμού και πολλαπλασιασμού» για τον υπολογισμό ύψωσης ενός αριθμού σε δύναμη. Ο

αλγόριθμος αυτός είναι μια αποτελεσματική μέθοδος να υψώσουμε έναν αριθμό a σε μια δύναμη n. Στις ελλειπτικές καμπύλες ο αντίστοιχος αλγόριθμος ονομάζεται «διπλασιασμού και πρόσθεσης» (double and add). Την λογική του αυτού αναλύεται στην παρακάτω ενότητα Montgomery ladder – Double and add.

Είναι εύκολο να υπολογίσουμε το Q, γνωρίζοντας το n και το P. Αλλά σχετικά δύσκολο να υπολογίσουμε το n όταν γνωρίζουμε τα P και Q. Η δυσκολία εύρεσης του n έχοντας μόνο την γνώση των P και Q είναι γνωστή από την βιβλιογραφία ως το πρόβλημα διακριτού λογαρίθμου (ECDLP) [4].

Στην αρχή του κεφαλαίου αναφέρθηκε το μέγεθος του κλειδιού σύμφωνα με το NIST για το DLP, που απαιτούνται τουλάχιστον 1024bit για να δημιουργηθεί ένα ασφαλές σύστημα. Αλλά στην περίπτωση του ECDLP το ίδιο επίπεδο ασφάλειας μπορεί να επιτευχθεί έχοντας απλώς μια καμπύλη από 128 bit. Αυτός είναι ο λόγος για τον οποίο οι ελλειπτικές καμπύλες έχουν διερευνηθεί σε μεγάλο βαθμό από τους ερευνητές και χρησιμοποιούνται ευρέως.

2.4 Trapdoor function

Με αφορμή με τα προηγούμενα που έχουμε αναφέρει η βασική ιδέα είναι ότι οι σημειακοί πολλαπλασιασμοί, δηλαδή πολλαπλασιάζοντας μια συντεταγμένη στην καμπύλη, υπολογίζονται τα προϊόντα της πράξεις. Από την άλλη είναι πολύ δύσκολο να αντιστραφεί η διαδικασία αυτή. Αυτό σημαίνει ότι κάποιος μπορεί εύκολα να υπολογίσει $Q = n * P$, αλλά αν έχει Q και P, το n είναι σχεδόν αδύνατο να το ανακτήσει σε σύντομο χρονικό διάστημα.

Αυτοί οι πολλαπλασιασμοί σημείων μπορεί να διαρκέσουν πολύ για να υπολογιστούν προσθέτοντας απλώς ένα σημείο στον εαυτό του n-φορές για μεγάλα n. Αλλά υπάρχουν πολλοί αλγόριθμοι που στοχεύουν στη μείωση αυτού του χρόνου υπολογισμού.

Οι περισσότεροι από αυτούς τους αλγόριθμους αποτελούνται από δύο ακόμη βασικούς υπολογισμούς (Elliptic Curve Point (ECP)) των: διπλασιασμό (doubling) ECP S(Q = 2P) και πρόσθεσης (adding) ECP (Q = P + c). Σε σύγκριση με το RSA, οι αριθμοί που απαιτούνται για παρόμοια ποσότητα κρυπτογραφικής ισχύος είναι σημαντικά μικρότεροι. Γεγονός που καθιστά το ECC δημοφιλές σε ενσωματωμένα συστήματα, όπου η απόδοση είναι συχνά υψηλότερη λόγω χρήσης ενέργειας και υψηλού κόστους.

Η ουσία όλων των κρυπτογραφικών αλγορίθμων δημόσιου κλειδιού είναι ότι ο καθένας έχει τη δική του μοναδική trapdoor function. Μια συνάρτηση trapdoor είναι μια συνάρτηση που μπορεί να υπολογιστεί μόνο με έναν τρόπο.

Έχουμε μια συνάρτηση Α + Β = Γ. Αν μας δοθεί το Α και το Β, μπορούμε να υπολογίσουμε το Γ. Ωστόσο εάν μου δοθεί το Β και το Γ, μπορώ επίσης να υπολογίσω το Α. Αυτό που περιεγράφηκε δεν είναι μια trapdoor function.

Από την άλλη πλευρά, αν έχουμε ένα απλό κείμενο (plain text) και ένα δημόσιο κλειδί, τότε μπορούμε να υπολογίσουμε το κρυπτογράφημα (ciphertext). Από την άλλη αν έχω το κρυπτοκείμενο και το δημόσιο κλειδί, τότε δεν μπορώ να βρω το αρχικό κείμενο. Αυτό είναι μια trapdoor function. Η συνάρτηση trapdoor είναι αυτή που κάνει το ECC ξεχωριστό.

Αρχικά, ξεκινάμε με ένα αυθαίρετο σημείο στην καμπύλη. Στη συνέχεια, χρησιμοποιούμε τη συνάρτηση dot για να βρούμε ένα νέο σημείο. Τέλος, συνεχίζουμε να επαναλαμβάνουμε τη λειτουργία «κουκκίδων» για να μεταβούμε (pivot) γύρω από την καμπύλη μέχρι να καταλήξουμε τελικά στο τελευταίο μας σημείο. Ας δούμε τον αλγόριθμο.

Ξεκινάμε από το Α:

- A dot B = -C (σχεδιάζουμε το ευθύγραμμο τμήμα A, B που τέμνει στο -C) (Εικόνα .8)

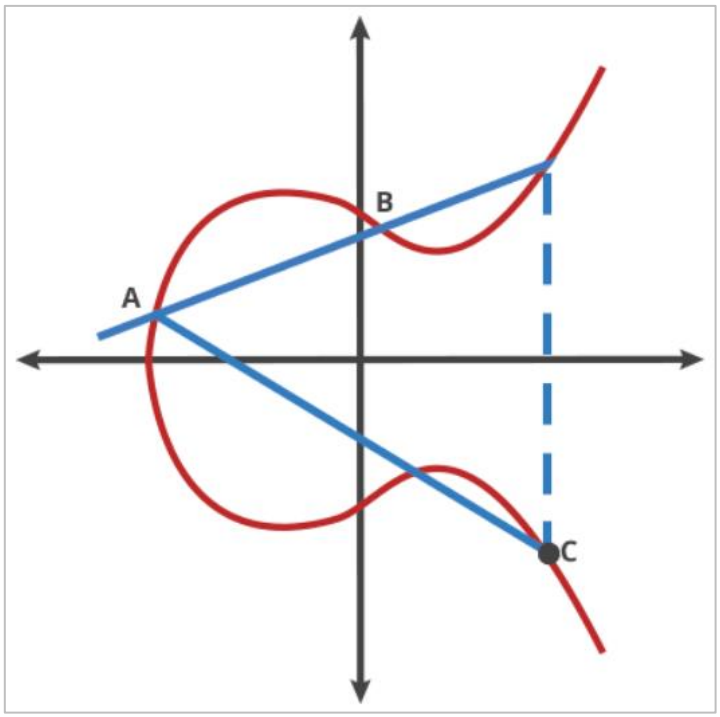
  Συνεχίζουμε την διαδικασία από το συμμετρικό x'x του -C σε C

- A dot C = -D (σχεδιάζουμε το ευθύγραμμο τμήμα A, C που τέμνει στο -D) (Εικόνα 3.9)

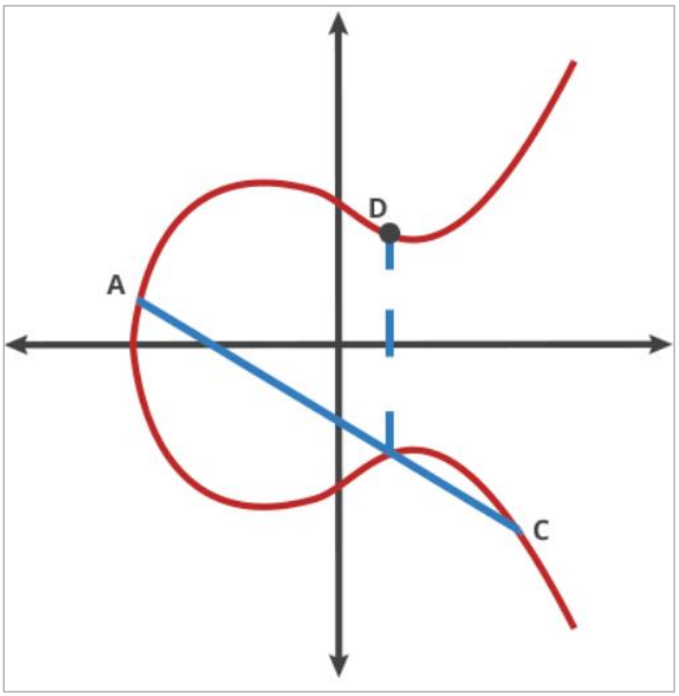
  Συνεχίζουμε την διαδικασία από το συμμετρικό x'x του -D σε D

- A dot D = -E (σχεδιάζουμε το ευθύγραμμο τμήμα A, D που τέμνει στο -E) (Εικόνα 3.10)

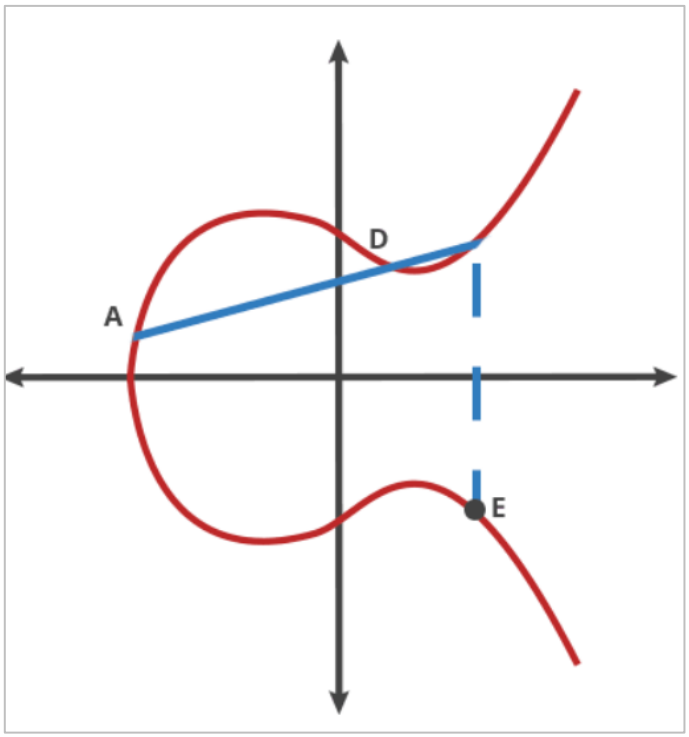
  Συνεχίζουμε την διαδικασία από το συμμετρικό x'x του -E σε E



Εικόνα 2.8: Trapdoor (A dot B = -C)



Εικόνα 2.9: Trapdoor (A dot C = -D)



Εικόνα 2.10: Trapdoor (A dot D = -E)

Αυτή είναι μια μεγάλη trapdoor συνάρτηση, διότι αν γνωρίζετε πού είναι το σημείο εκκίνησης (A) και πόσοι μεταβάσεις (pivot) απαιτούνται για να φτάσουμε στο τελικό σημείο (E), είναι πολύ εύκολο να βρούμε το τελικό σημείο.

Από την άλλη πλευρά, αν το μόνο που γνωρίζετε είναι το σημείο εκκίνησης και το τέλος, είναι σχεδόν αδύνατο να βρείτε πόσα μεταβάσεις χρειάστηκαν για να φτάσουμε εκεί.

Δημόσιο κλειδί: Αρχικό σημείο (A), τελικό σημείο (E)

Ιδιωτικό κλειδί : πλήθος από μεταβάσεις (pivot) μεταξύ των σημείων A και E.

2.5 Πρωτόκολλο ανταλλαγής κλειδιών ελλειπτικών καμπυλών Diffie-Hellman

Για να επικοινωνούν με ασφαλή τρόπο η Αλίκη και ο Μπομπ πρέπει να ανταλλάξουν τα κλειδιά τους ώστε να μπορούν να κρυπτογραφήσουν και να αποκρυπτογραφήσουν τα μηνύματα. Έτσι το 1976 οι Whitefield Diffie και Martin Hellman [8] έδωσαν την ιδέα της ανταλλαγής κλειδιών μέσω ενός δημόσιου καναλιού χωρίς να διακυβεύεται η ασφάλεια.

Έχει σχεδιαστεί χρησιμοποιώντας μια κυκλική ομάδα σημείων ελλειπτικής καμπύλης που βασίζεται στη δυσκολία επίλυσης του ECDLP [9]. Η ακόλουθη περιγραφή δείχνει την διαδικασία για πρωτόκολλο ανταλλαγής κλειδιών Diffie-Hellman.

- Μια ελλειπτική καμπύλη έστω E επιλέγεται αμοιβαία από την Αλίκη και τον Μπομπ σε ένα πεπερασμένο πεδίο $\mathbb{F}$ μαζί με το G ως σημείο βάσης (σημείο γεννήτριας) που δημιουργείται από το E.

- Με την σειρά τους η Αλίκη και ο Μπομπ επιλέγουν ένα τυχαίο αριθμό K_A και K_B αντίστοιχα (μυστικό κλειδί). Υπολογίζουμε $P_A = P_A * G$ και $P_B = P_B * G$ αντίστοιχα, τα οποία αποδεικνύεται ότι είναι ένα σημείο της E

- Τόσο το P_A όσο και το P_B ανταλλάσσονται μεταξύ τους.

- Τότε η Αλίκη υπολογίζει $P_{AB} = K_A * P_B$ (shared key). Αντίστοιχα και ο Μπομπ υπολογίζει $P_{AB} = K_B * P_A$

Το P_{AB} χρησιμοποιείται ως ασφάλεια κλειδιού για την εύρεση K_A και K_B.

Κεφάλαιο 3

3.1 Συστοιχία Επιτόπια Προγραμματιζόμενων Πυλών FPGA

Κατά γενικό τρόπο, μια προγραμματιζόμενη συστοιχίες πύλης πεδίου (**Field-Programmable-Gate-Array** FPGA) αποτελείται από έναν μεγάλο αριθμό διασυνδέσεων μεταξύ των διαφορετικών πόρων μέσα στο FPGA. Οι σύγχρονες εκδόσεις αυτών των συσκευών έχουν πολύ μεγάλο αριθμό τυποποιημένων ψηφιακών πυλών (π.χ. AND, XOR) και άλλων ψηφιακών λειτουργιών όπως απαριθμητές (counter), επεξεργαστές ψηφιακού σήματος (DSP) και στοιχεία μνήμης που μπορεί να είναι από ένα απλό flip-flop ή πιο σύνθετα μπλοκ μνήμης

Τα FPGA διαδραματίζουν σημαντικό ρόλο στην ανάπτυξη των ενσωματωμένων συστημάτων λόγω της ικανότητάς τους να ξεκινούν την ανάπτυξη λογισμικού συστήματος (S/W) ταυτόχρονα με το υλικό (H/W). Επιτρέπουν προσομοιώσεις συστήματος σε πολύ πρώιμο στάδιο της ανάπτυξης. Με δυνατότητα δοκιμής και επαναλήψεις πριν από το τελικό στάδιο της δημιουργίας της αρχιτεκτονικής του συστήματος. Ο προγραμματισμός των FPGA γενικά καθορίζεται χρησιμοποιώντας μια γλώσσα περιγραφής υλικού (HDL, Verilog), παρόμοια με αυτή που χρησιμοποιείται για ένα ολοκληρωμένο κύκλωμα συγκεκριμένης εφαρμογής (ASIC, **Application-Specific-Integrated-Circuit**).

Το FPGA έχει παρόμοιο πεδίο εφαρμογών με άλλα προγραμματιζόμενα ολοκληρωμένα ψηφιακά κυκλώματα όπως τα PLD (programmable logic device) και τα ASIC. Όμως τα ιδιαίτερα χαρακτηριστικά του FPGA είναι τα εξής:

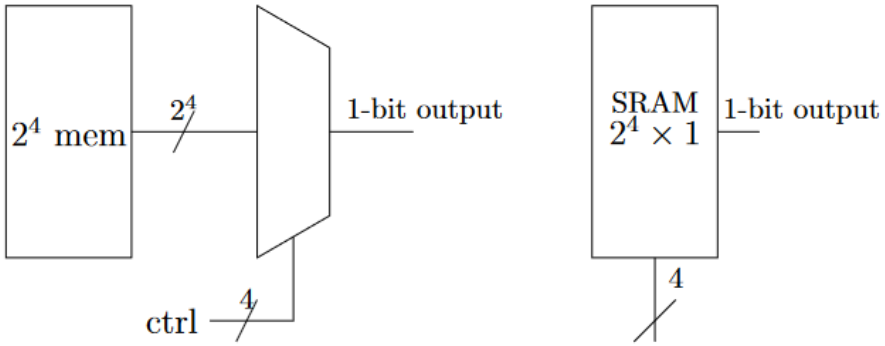
- Το FPGA χάνει τον προγραμματισμό του κάθε φορά που διακόπτεται η τάση τροφοδοσίας του. Επομένως απαιτεί εξωτερικό μικροεπεξεργαστή ή μνήμη με μόνιμη συγκράτηση δεδομένων από τα οποία θα προγραμματίζεται, κάθε φορά που επανέρχεται η τάση τροφοδοσίας.
- Ο προγραμματισμός του FPGA μπορεί να αλλάζει κάθε φορά που τροποποιείται το λογισμικό του μικροεπεξεργαστή ή τα δεδομένα της μνήμης που το ελέγχει.
- Δεν υπάρχει όριο στο πόσες φορές μπορεί να επαναπρογραμματιστεί.
- Η κατανάλωση ισχύος είναι σημαντικά αυξημένη, σε σχέση με τα ASIC.

3.1.1 Λογικά στοιχεία - Logic elements

Ο πίνακας αλήθειας είναι ο πιο συνηθισμένος τρόπος αναπαράστασης ενός κυκλώματος. Μπορούν οι πίνακες αυτοί να ρυθμίσουν σχεδόν οποιοδήποτε είδος κυκλώματος, όποιο κι αν είναι αυτό. Επιπλέον, μπορούν να αντιπροσωπεύουν ψηφιακά συστήματα που συνδέουν τις εισόδους με τις εξόδους [10]. Αυτοί οι πίνακες είναι συναρτήσεις Boolean που ορίζονται με το γράμμα $\boldsymbol{f}$ ως τη σχέση των πιθανών εξόδων που σχετίζονται με όλες τις πιθανές εισόδους ενός ψηφιακού συστήματος [11]. Μπορούμε να πούμε ότι αυτοί οι πίνακες είναι η καρδιά ενός FPGA.

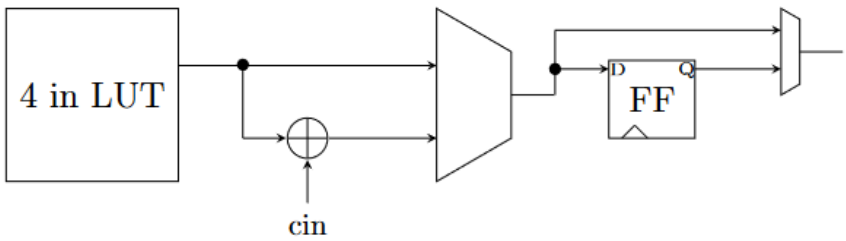
Οι πίνακες αναζήτησης (LUTs) εφαρμόζουν πίνακες αλήθειας σε επίπεδο υλικού. Συνήθως αποτελούνται από N εισόδους και μόνο μία έξοδο, δηλ. παρουσιάζουν 2^{2^n}πιθανές δυαδικές συναρτήσεις N μεταβλητών εισόδων.

Ωστόσο, το FPGA μπορεί να χρησιμοποιήσει περισσότερα LUT συνδεδεμένα μεταξύ τους για την υλοποίηση συναρτήσεων με περισσότερες από N εισόδους. Για παράδειγμα, τα LUT που εφαρμόζονται στην οικογένεια Xilinx 7 έχουν έξι εισόδου [12], αλλά σε παλαιότερες οικογένειες όπως οι 6, 5, 4 και 3, τα LUTs είχαν μόνο τέσσερις διαθέσιμες εισόδους. Το Σχήμα 4.1 δείχνει δύο τρόπους που χρησιμοποιούνται για την υλοποίηση ενός LUT τεσσάρων εισόδων και μιας εξόδου.



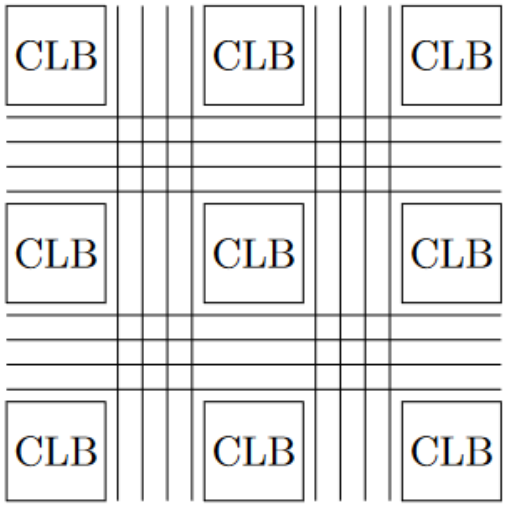
Σχήμα 3.1: Δύο διαφορετικά LUT τεσσάρων εισόδων

Τα LUT έχουν εσωτερικά ομαδοποιημένα ρυθμιζόμενα λογικά μπλοκ (CLB) όπως φαίνεται στη [13], και η οργάνωσή τους ποικίλλει μεταξύ όλων των οικογενειών και των κατασκευαστών FPGA. Ωστόσο, η Xilinx η ονομασία που χρησιμοποιεί για το στοιχειώδες λογικό μπλοκ της είναι Slice. Τα λογικά μπλοκ συνήθως αποτελούνται από πολλαπλά στοιχεία όπως πολυπλέκτες, επιπλέον λογικά στοιχεία, καταχωρητές και εισόδους μεταφοράς. Το Σχήμα 3.2 δείχνει το διάγραμμα ενός LUT τεσσάρων εισόδων με XOR στην έξοδο και έναν καταχωρητή flip-flop.



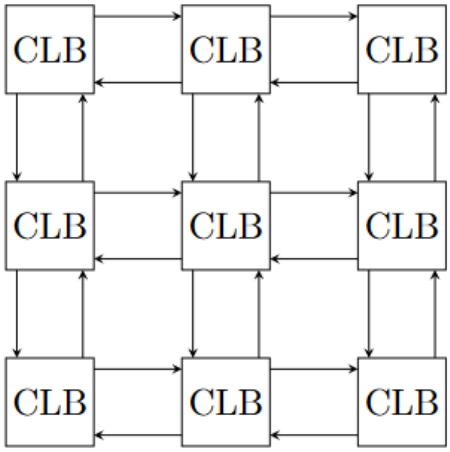
Σχήμα 3.2: LUT με καταχωρητή και XOR

Η διασύνδεση κάθε λογικού μπλοκ για την ανάπτυξη εξαιρετικά πολύπλοκων συστημάτων είναι απαραίτητη στο FPGA. Αυτά τα εξαρτήματα είναι προγραμματιζόμενα/ παραμετροποιήσιμα, συνήθως έχουν διάταξη πίνακα με πολλαπλές διαδρομές κάθετα και οριζόντια. Υπάρχουν τρεις τρόποι που χρησιμοποιούνται για τη διασύνδεση λογικών μπλοκ και το Σχήμα 3.3 δείχνει τη διασύνδεση τοποθέτησης σε FPGA



Σχήμα 3.3: Σύνδεση FPGA μεταξύ CLUBs

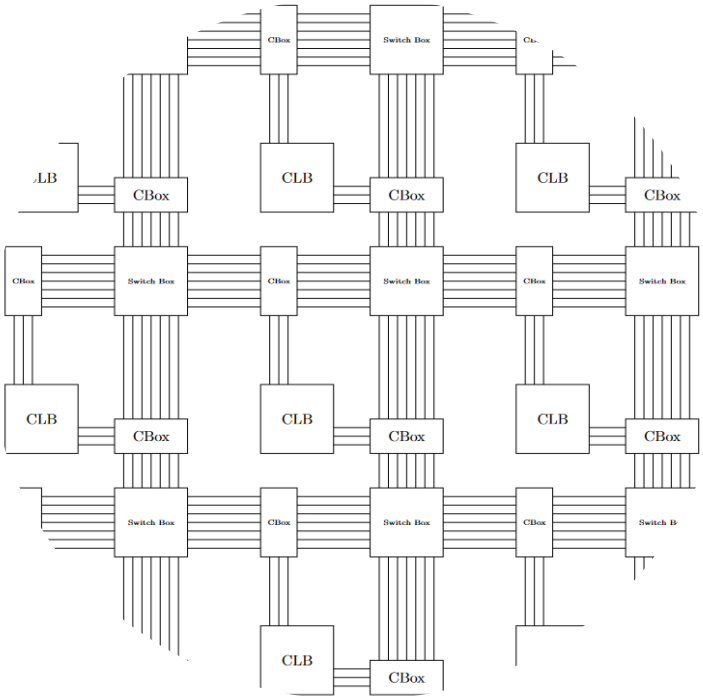
Πλησιέστερος γείτονας (Nearest neighbor): Το Σχήμα 3.4 δείχνει την απλούστερη δομή που χρησιμοποιείται για τη διασύνδεση των λογικών μπλοκ. Μια αμφίδρομη σύνδεση με τον πλησιέστερο γείτονα σε κάθε κατεύθυνση (βόρεια, νότια, ανατολικά και δυτικά) και δεν υπάρχει κανένα είδος στοιχείων που να παρακάμπτουν οποιοδήποτε λογικό μπλοκ. Κάθε σήμα περνά από κάθε λογικό μπλοκ. Αυτή η τελευταία πτυχή επηρεάζει άμεσα την απόδοση και αυξάνει την καθυστέρηση.



Σχήμα 3.4: Η δομή του πλησιέστερου γείτονα

Επίσης, καθώς κάθε CLB διασυνδέεται με ένα άλλο προς οποιαδήποτε κατεύθυνση, επιβάλλει έναν περιορισμό στις δυνατότητες άμεσης αλληλεπίδρασης με άλλα CLB σε άλλες γειτονιές. Αυτό το μειονέκτημα τους επιτρέπει να αλληλοεπιδρούν απευθείας με πιο κοντινά CLB στην ίδια περιοχή.

Στο παρακάτω Σχήμα 3.5 φαίνεται η τμηματοποιημένη δομή (segment structure). Τα επιμέρους στοιχεία που απαρτίζουν τις διασυνδέσεις εξαρτημάτων είναι:



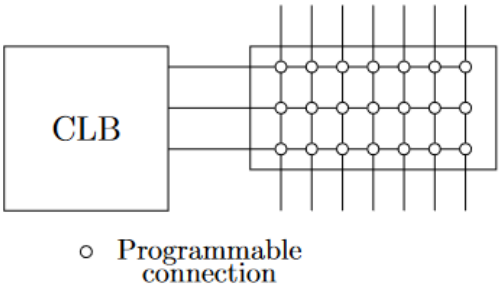
Σχήμα 3.5: Λογικά μπλοκ με νησιώτικο τρόπο με μπλοκ σύνδεσης και κουτιά διακόπτη στην ίδια αρχιτεκτονική

- **Διαδρομές** (Roots): Οι δίαυλοί επικοινωνίας (Buses) έχουν κάθετες και οριζόντιες γραμμές που συνδέονται απευθείας μέσω μπλοκ σύνδεσης (CB – Connection Blocks). Ο «σταυρός» που σχηματίζεται από κάθετες και οριζόντιους διαύλους αποτελείται από ένα διακόπτη που χρησιμοποιείται για τη διασύνδεση των CB με αυτά.

- **Μπλοκ σύνδεσης** (CB): Διαχειρίζονται εισόδους και εξόδους από τα λογικά μπλοκ, συνδέοντας το καθένα από αυτά με πολλά κομμάτια. Αυτό το είδος σύνδεσης επιτρέπει την επιλογή του συνδέσμου που είναι ενεργός.

- **Switch Boxes**: Αυτός είναι ένας τρόπος σύνδεσης λογικών μπλοκ μεταξύ διαύλων διάταξης μεταξύ κάθετων και οριζόντιων. Εάν απαιτείται σύνδεση, ο προγραμματισμός των πλαισίων μεταγωγής δημιουργεί μια νέα διαδρομή για την αποστολή/λήψη δεδομένων μεταξύ των λογικών μπλοκ. Το Σχήμα 4.6 δείχνει ένα CLB με προγραμματιζόμενες συνδέσεις που χρησιμοποιούνται για τη διασύνδεση του CLB με

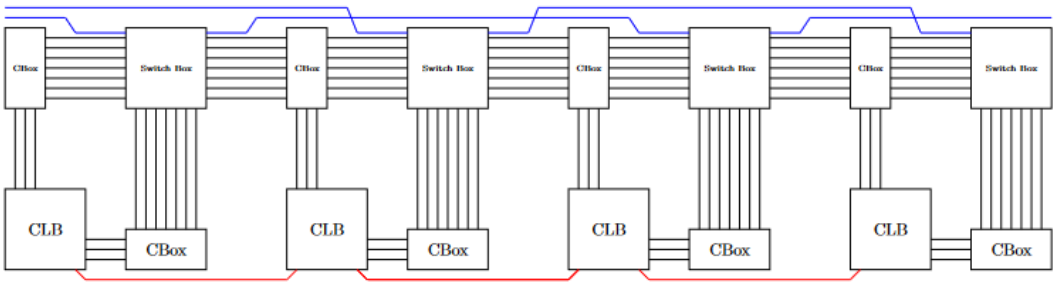
τους γείτονές του και ορισμένες εργασίες που επικεντρώνονται στη σχεδίαση του διακόπτη είναι [14], [15].



Σχήμα 3.6: Προγραμματιζόμενο μπλοκ σύνδεσης

Τμηματοποιημένη αρχιτεκτονική

Η τμηματοποιημένη αρχιτεκτονική προσφέρει εξαιρετική ευελιξία και απόδοση, επιτρέποντας τη σύνδεση των CLBs. Σε αυτήν την αρχιτεκτονική απαιτεί όλα τα σήματα να περνούν μέσα από διακόπτες και μπλοκ σύνδεσης, αυξάνοντας την καθυστέρηση στη σχεδίαση του κυκλώματος. Ωστόσο, σε ορισμένα σενάρια, αυτό έχει το πλεονέκτημα ότι επιτρέπει την άμεση σύνδεση μεταξύ των στοιχείων. Το Σχήμα 4.7 δείχνει την υβριδική αρχιτεκτονική διασύνδεσης των CLBs.

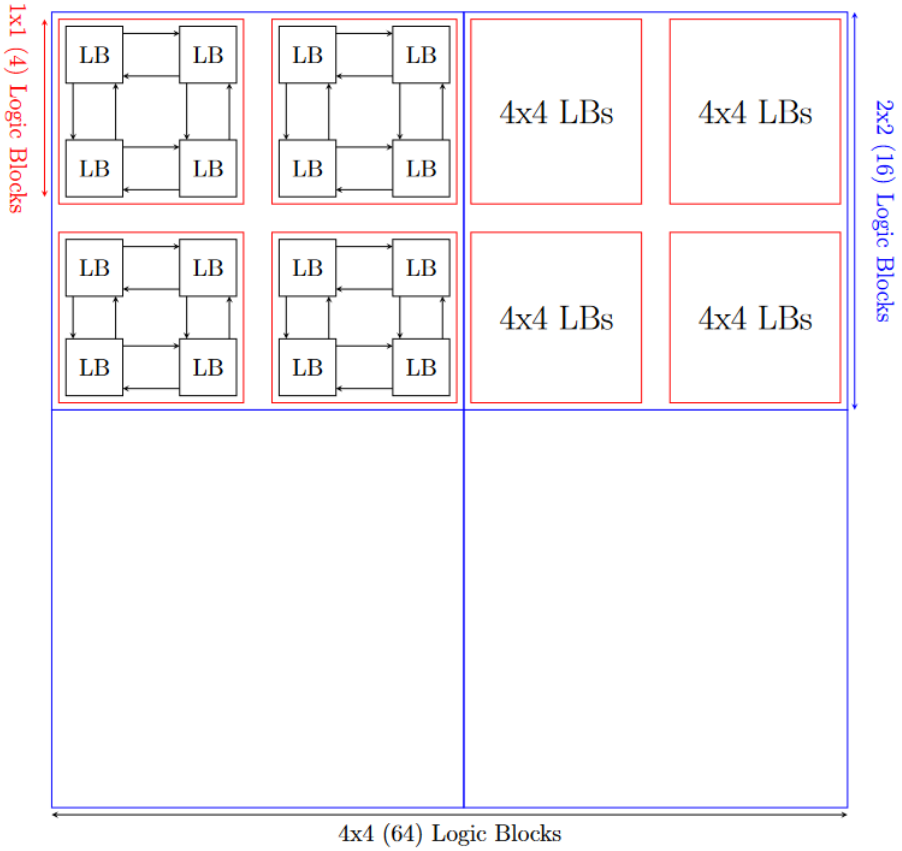


Σχήμα 3.7: Υβριδική δομή του πλησιέστερου γείτονα και τμηματοποιημένη δομή.

Ιεραρχική δομή

Σε ορισμένες αρχιτεκτονικές FPGA, τα λογικά μπλοκ έχουν ιεραρχική διάταξη. Επομένως, η κατάλληλη χρήση αυτών των μπλοκ σημαίνει βελτίωση στη μείωση των καθυστερήσεων και στη βελτίωση της δρομολόγησης του σήματος. Έτσι, τα βασικά στοιχεία αυτού του είδους αρχιτεκτονικής μοιάζουν με τμηματοποιημένη αρχιτεκτονική. Το Σχήμα 3.8 δείχνει την ιεραρχική αρχιτεκτονική

με έως και 64 Logic blocks. Αυτό απαιτεί λιγότερους διακόπτες και έχει ταχύτερη λογική σε σύγκριση με ένα τμηματοποιημένο μοντέλο [16, 17].



Σχήμα 3.8: Ιεραρχική δομή με ένα σύμπλεγμα λογικών μπλοκ.

3.2 DSP BLOCKS (Digital Signal Processing)

Η ψηφιακή επεξεργασία σήματος (DSP) είναι ένας τομέας που σημειώνει συνεχείς προόδους τόσο όσον αφορά τις προσεγγίσεις του λογισμικού όσο και τις πλατφόρμες υλικού (embedded systems). Μερικές από τις πιο συνηθισμένες λειτουργίες σε αυτόν τον τομέα είναι ψηφιακά φίλτρα, κωδικοποιητές, αποκωδικοποιητές και μαθηματικοί μετασχηματισμοί όπως ο μετασχηματισμός Fourier (FFT).

Οι περισσότερες DSP λειτουργίες και αλγόριθμοι είναι αρκετά περίπλοκοι και περιλαμβάνουν μεγάλο αριθμό μεταβλητών, συντελεστών και διαφορετικών σταδίων υλοποίησης. Η βασική λειτουργεία παρέχεται συνήθως από μονάδες MAC (**multiplier–accumulator**) ή MAD (**multiply-add**). Δεδομένου ότι συνήθως απαιτείται υψηλή συχνότητα λειτουργίας ή και απόδοσης,

είναι συχνά απαραίτητο να χρησιμοποιούνται DSP. Των οποίων το υλικό και το σύνολο εντολών είναι βελτιστοποιημένα για την εκτέλεση λειτουργιών MAC.

Η CPU στα DSP έχουν σχεδιαστεί για να εκτελούν εντολές σε λιγότερους κύκλους ρολογιού από ότι σε επεξεργαστές γενικού σκοπού. Για πολλά χρόνια, τα DSP ήταν οι μόνες πλατφόρμες ικανές να εφαρμόσουν αποτελεσματικά και αποδοτικά διάφορους αλγορίθμους. Ωστόσο, τα τελευταία χρόνια, τα FPGA έχουν αναδειχθεί ως σοβαροί φυσικοί ανταγωνιστές σε αυτόν τον τομέα λόγω του εγγενούς παραλληλισμού του hardware, της ικανότητάς τους να εφαρμόζουν πολύ αποτελεσματικά αριθμητικές πράξεις, βάση του τεράστιου όγκου των λογικών πόρων που διατίθενται.

Από την εμφάνιση των πρώτων FPGA στη δεκαετία του 1980, ένας από τους κύριους στόχους των χρηστών ήταν να διασφαλίσουν ότι οι συσκευές τους είναι ικανές να υλοποιούν αποτελεσματικά δυαδικές αριθμητικές πράξεις (κυρίως πρόσθεση, αφαίρεση και πολλαπλασιασμό). Καθώς τα FPGA έγιναν όλο και πιο δημοφιλή, εμφανίστηκαν νέες θέσεις εφαρμογών που απαιτούσαν νέους εξειδικευμένους πόρους υλικού. Η διαθεσιμότητα των ενσωματωμένων μπλοκ μνήμης ήταν ιδιαίτερα χρήσιμη για την υλοποίηση κυκλωμάτων απόκτησης και ελέγχου δεδομένων. Αποφεύγοντας την ανάγκη για εξωτερικές μνήμες και μειώνοντας τους χρόνους απόκρισης πρόσβασης στη μνήμη. Μετά από αυτά, πολλά άλλα εξειδικευμένα μπλοκ υλικού συμπεριλήφθηκαν σταδιακά σε κάθε νέα οικογένεια συσκευών.

Ο ψηφιακός επεξεργαστής σήματος (DSP) είναι απαραίτητος στις σύγχρονες αρχιτεκτονικές FPGA. Αυτές οι συσκευές μπορούν να χειριστούν βασικές λειτουργίες όπως αριθμητικές πράξης, λογικές, καθώς και συγκρίσεις των εισόδων. Έτσι, μπορούν να επεξεργαστούν πολλαπλά δεδομένα μιας εντολής στην εσωτερική τους αρχιτεκτονική.

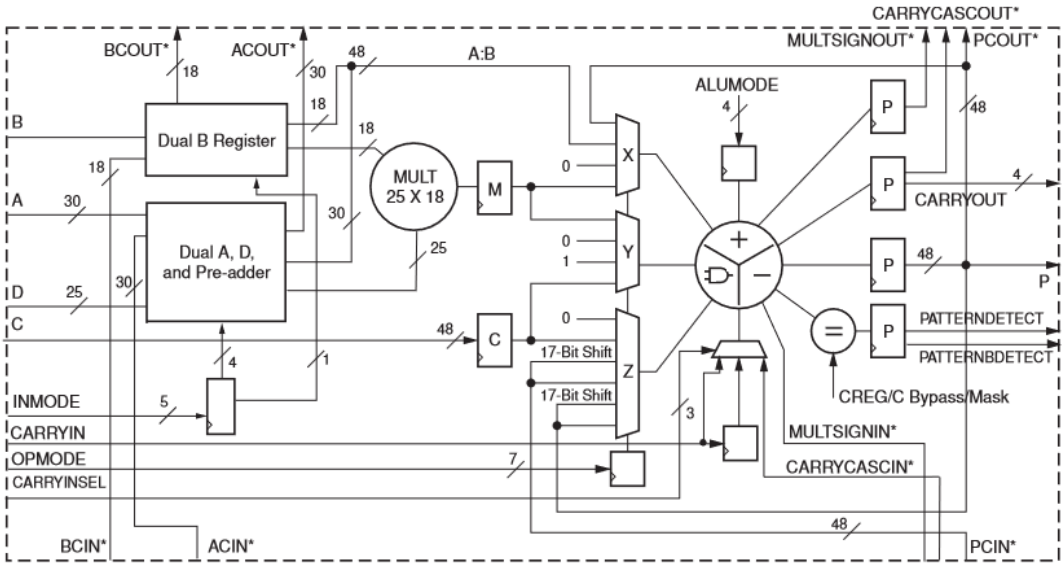
Η ALU (**Arithmetic Logic Unit**) στα συμβατικά DSP περιλαμβάνουν συνήθως από μία έως τέσσερις μονάδες MAC που λειτουργούν παράλληλα. Οι αυστηρές αρχιτεκτονικές τους δεν επιτρέπουν, για παράδειγμα, την παραμετροποίηση του αριθμού των bit των τελεστών σε έναν πολλαπλασιασμό. Επομένως, ο παραλληλισμός και το εύρος ζώνης είναι περιορισμένα σε αυτές τις πλατφόρμες και η αύξηση της συχνότητας λειτουργίας είναι, στις περισσότερες περιπτώσεις, ο μόνος τρόπος βελτίωσης της απόδοσης.

Λαμβάνοντας υπόψη ότι η επεξεργασία σήματος με τα χρόνια ήταν η πιο σημαντική εφαρμογή των ενσωματωμένων πολλαπλασιαστών (embedded multipliers), είναι φυσικό να εξελίχθηκαν σε πιο

σύνθετα μπλοκ, που ονομάζονται DSP blocks, όπως αυτό στο σχήμα 3.9. Το οποίο περιλαμβάνει όλους τους πόρους που απαιτούνται για την υλοποίηση μιας μονάδας MAC.

Υπάρχουν διαφορετικές αρχιτεκτονικές για μπλοκ DSP, αλλά τα περισσότερα από αυτά μοιράζονται τρία κύρια στάδια, δηλαδή τον προ-αθροιστή (pre-adder), τον πολλαπλασιαστή (multiplier) και την ALU. Ανάλογα με τη συσκευή, η ALU μπορεί απλώς να αποτελείται από έναν αθροιστή/αφαιρέτη (adder/subtracto). Όπως και στην περίπτωση των πολλαπλασιαστών, οι καταχωρητές τοποθετούνται τόσο στην είσοδο όσο και στην έξοδο του κυκλώματος στο Σχήμα 3.9. Με αυτόν τον τρόπο μπορούν να υλοποιηθούν δομές «αγωγών» (pipeline structures) που επιτυγχάνουν πολύ υψηλές συχνότητες λειτουργίας.

Η εσωτερική αρχιτεκτονική του DSP διαθέτει τέσσερις θύρες εισόδου, A, B, C και D, και όλες έχουν διαφορετικά μεγέθη διαύλου. Για παράδειγμα, η θύρα A έχει πλάτος 30 bit ενώ η B έχει πλάτος 18 bit, επομένως είναι δυνατό να τα συνδέσετε για να χρησιμοποιήσετε μια θύρα πλάτους 48 bit. Η θύρα D έχει πλάτος 25 bit και η θύρα A χρησιμοποιείται στον προ-αθροιστή για την προσθήκη δύο τελεστών. Τέλος, η θύρα C έχει πλάτος 48-bit και πηγαίνει απευθείας στην Αριθμητική Λογική Μονάδα του DSP (ALU). Αυτό το στοιχείο μπορεί να εκτελέσει διαφορετικές λειτουργίες χρησιμοποιώντας εισόδους πλάτους έως και 48 bit.



Σχήμα 3.9: Αρχιτεκτονική DSP48E1

Έτσι, τα DSP έχουν άλλου είδους είσοδο. Αυτά προέρχονται απευθείας από άλλα DSP, καθιστώντας δυνατή τη διάταξη πλακιδίων των DSP στο FPGA. Επιπλέον, κάθε στήλη DSP έχει ένα αποκλειστικό δίκτυο διασύνδεσης που επιτρέπει σε κάθε DSP να μοιράζεται δεδομένα χρησιμοποιώντας τις θύρες ACIN, BCIN, CARRYIN και PCIN για την εισαγωγή δεδομένων. Επιπλέον, οι θύρες ACOUT, BCOUT, CARRYCASCOUT και PCOUT πρέπει να στέλνουν δεδομένα από το δίκτυο διασύνδεσης των DSP.

Το δίκτυο διασύνδεσης των DSP επιτρέπει τη μετάδοση μιας εισόδου ή εξόδου σε άλλα DSP στο ίδιο δίκτυο. Αυτό μειώνει τους πόρους διασύνδεσης FPGA που χρησιμοποιούνται. Επομένως, η διαδοχική διασύνδεση είναι διαθέσιμη μόνο στην ίδια στήλη και δεν μπορούν να συνδεθούν απευθείας με άλλες στήλες.

Το αριθμητικό τμήμα του DSP (DSP48E1) αποτελείται από δύο πολλαπλασιαστές συμπληρώματος 25 × 18 bit του οποίου προηγείται ένας προ-αθροιστής 25 bit. Η έξοδος προ-αθροιστή συνδέεται απευθείας με τον πολλαπλασιαστή και η έξοδος του πολλαπλασιαστή συνδέεται άμεσα με έναν από τους τρεις πολυπλέκτης διαδρομής δεδομένων 48 bit. Τέλος, τα δεδομένα σχετικά με τα αποτελέσματα αυτών των τριών συνδέονται με την ALU για να εκτελέσουν οποιαδήποτε λειτουργία.

3.2.1 Οικογένειες Xilinx FPGA

Η Xilinx είχε διαφορετικές οικογένειες FPGA από την ίδρυσή της. Σε αυτήν την ενότητα γίνετε μια μικρή ανάπτυξη αυτών και των χαρακτηριστικών τους.

Spartan 7: Είναι η φθηνότερη επιλογή, με αρκετούς πόρους για συνδεσιμότητα και εφαρμογές επεξεργασίας στη βιομηχανία, την αυτοκινητοβιομηχανία και τις επικοινωνίες. Διαθέτει λογικές κυψέλες (logic cells) μεταξύ 6-102k και έχει τη χαμηλότερη κατανάλωση ενέργειας και τα τμήματα DSP, και αυτή η οικογένεια έχει ταχύτητα επεξεργασίας έως και 551 MHz.

Artix 7: Έχουν καλύτερη απόδοση από το Spartan και μεγαλύτερο εύρος ζώνης επεξεργασίας. Έρχονταν με 740 slices DSP48E1 με υποστήριξη μνήμης RAM DDR3 και διαθέτει μεταξύ 13k-200k λογικές κυψέλες, 2,5x μπλοκ RAM και 5,7 φορές περισσότερα slices DSP από τις προηγούμενης γενιάς.

Kintex-7: Αυτός ο τύπος FPGA έχει υψηλές αναλογίες τμημάτων DSP και βασίζεται σε άλλες τεχνολογίες όπως η συνδεσιμότητα ethernet γενιάς PCI express 3 και 10 Gigabit. αυτή η οικογένεια συνήθως επικεντρώνεται σε λύσεις ασύρματης σύνδεσης και βίντεο. Επομένως, έχει έως και 478K λογικά κελιά και το τμήμα DSP του μπορεί να επιτύχει 629 MHz με 1920 DSP slices.

Virtex-7: Αυτή η υπο-μάρκα του FPGA έχει την υψηλότερη ποσότητα πόρων με την υψηλότερη διαθέσιμη ταχύτητα πάνω από τις προηγούμενες. Είναι επίσης βελτιστοποιημένη. Έχουν τους ίδιους πόρους αλλά με καλύτερη απόδοση. Επομένως, μπορούν να έχουν έως και 2 εκατομμύρια λογικά κελιά, 85 Mb Block RAM και 3600 τμήματα DSP. Ο Πίνακας 4.1 δείχνει τους διαθέσιμους πόρου για τις διάφορες οικογένειες που παρουσιάζονται παραπάνω.

	Logic Cells	DSPs	Memory
Spartan-7	600-102400	10-160	180k-4320k
Artix-7	12800-215360	40-740	720k-13140k
Kintex-7	65600-477760	240-1920	4860k-34380k
Virtex-7	582720-1139200	1260-3360	28620k-67680k

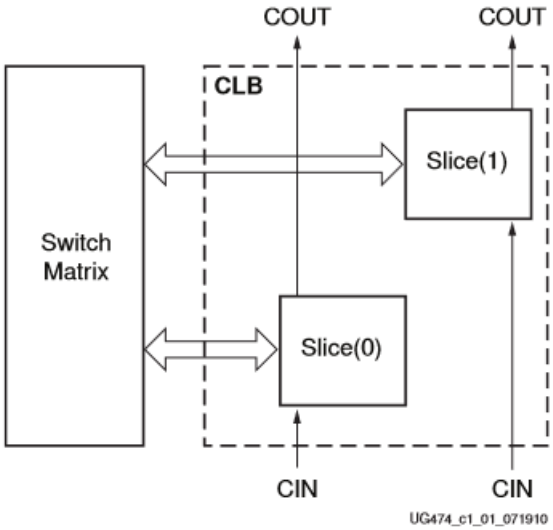
Πίνακας 3.1: Συγκριτικός πίνακας πόρων Xilinx 7-series.

3.3 Configurable Logic Block (CLB)

Οι μεγαλύτερες οικογένειες Xilinx – FPGA έχουν διαφορετικά είδη από CLBs. Είναι απαραίτητο όταν ένας σχεδιαστής αναπτύσσει οποιαδήποτε νέα αρχιτεκτονική, και η κατανόηση των χαρακτηριστικών επιτρέπει σε κάποιον να εκμεταλλευτεί τις δυνατότητές του υλικού. Ωστόσο, τα CLB που βρέθηκαν σε οικογένειες Xlinix-7 [18] έχουν το ίδιο είδος CLB [19] που παρουσιάζεται εδώ.

Οι λογικές πηγές είναι τα CLB για την υλοποίηση διαδοχικών και συνδυαστικών κυκλωμάτων. Κάθε CLB έχει άμεση σύνδεση με τον πίνακα μεταγωγής (switch matrix) που επιτρέπει την πρόσβαση στον γενικό πίνακα δρομολόγησης. Επίσης, κάθε CLB περιέχει ένα ζευγάρι από slices.

Το LUT που εφαρμόζεται σε αυτά τα CLB επιτρέπει τη διαμόρφωση είτε ως LUT 6 εισόδων με μία έξοδο, είτε ως δύο-5 εισόδων με μεμονωμένες εξόδους που μοιράζονται τις διευθύνσεις ή τις λογικές εισόδους. Επιπλέον, υπάρχει η επιλογή αποθήκευσης της εξόδου μιας από τις 5 εισόδους LUT σε ένα flip-flop απευθείας και το σχήμα 4.10 δείχνει τη γενική αρχιτεκτονική του CLB.



Σχήμα 3.10: Εσωτερική διάταξη και διασύνδεση CLB.

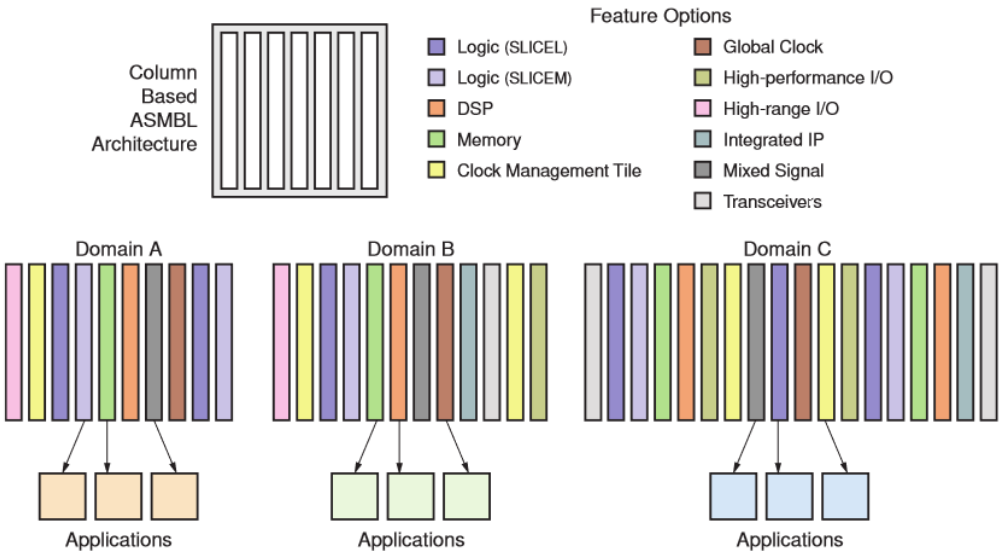
Μια άλλη σημαντική χρήση του LUT είναι η δυνατότητα χρήσης τους ως διαφορετικά εξαρτήματα με άλλες επιλογές. Η πρώτη επιλογή επιτρέπει τη χρήση τους ως κατανεμημένη μνήμη RAM 64-bit ή καταχωρητή μετατόπισης 32-bit ή ως δύο καταχωρητές μετατόπισης 16-bit. Τα σύγχρονα εργαλεία μπορούν να χρησιμοποιήσουν αυτά τα χαρακτηριστικά της λογικής, αριθμητικής και μνήμης. Ένας σχεδιαστής με αρκετή εμπειρία μπορεί να τα δημιουργήσει και να τα διαμορφώσει όπως απαιτείται. Παρακάτω παρουσιάζεται μια λίστα με τα κύρια χαρακτηριστικά ενός CLB

- 6-bit look-up table (LUT).
- Dual 5-input LUTS
- Καταχωρητής μετατόπισης και δυνατότητα κατανεμημένης μνήμης RAM.
- Λογική μεταφοράς υψηλής ταχύτητας για αριθμητικές συναρτήσεις (High-speed carry logic for arithmetic functions)
- Χρήση πολυπλέκτη για αποτελεσματικότητα.

Όπως αναφέρθηκε, όλες οι οικογένειες της σειράς 7 διαθέτουν επεκτάσιμους πόρους, παρέχοντας μια ομοιογενή αρχιτεκτονική. Ωστόσο, η ποσότητα των CLB διαφοροποιεί όλες τις οικογένειες Xilinx FPGA. Επομένως, η χωρητικότητα της συσκευής σχετίζεται άμεσα με τον αριθμό

των λογικών κυψελών που παρέχονται με το ισοδύναμο ενός κλασικού LUT 4 εισόδων και ενός flip-flop.

Κάθε οικογένεια έχει τη δική της διάταξη CLB στη σειρά 7. Η διάταξη είναι σε στυλ στήλης. Χρησιμοποιούν μια αποκλειστική τεχνολογία που αναπτύχθηκε από τη Xilinx με το όνομα Advanced Silicon Modular Block (ASMBL) που επιτρέπει στα FPGA με ένα συνδυασμό χαρακτηριστικών να βελτιστοποιούν τις διασυνδέσεις και τη χρήση των πόρων. Η Εικόνα 3.1 δείχνει τη συνιστώσα σε κάθε στήλη σε όλο το FPGA.

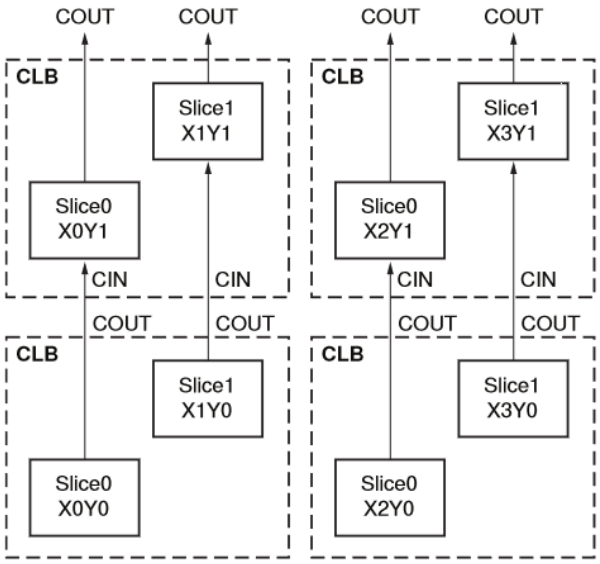


Εικόνα 3.1: Αρχιτεκτονική ASMBL με στοιχεία ως στήλες.

CLBs

Στην οικογένεια που παρουσιάζεται εδώ, κάθε CLBs έχει δύο Slice και κάθε Slice έχει τέσσερα LUTS 6-bit και οκτώ στοιχεία αποθήκευσης. Η εσωτερική διάταξη των CLB είναι ένα Slice0 στο κάτω μέρος του CLB στην αριστερή γωνία και το Slice1 στο επάνω μέρος του CLB.

Τα Slices δεν έχουν άμεση σύνδεση, και η οργάνωση κάθε Slices είναι μια στήλη. Για παράδειγμα, το Σχήμα 3.11 δείχνει ότι κάθε Slice έχει ανεξάρτητο σήμα carry με κάθε άλλο Slice.

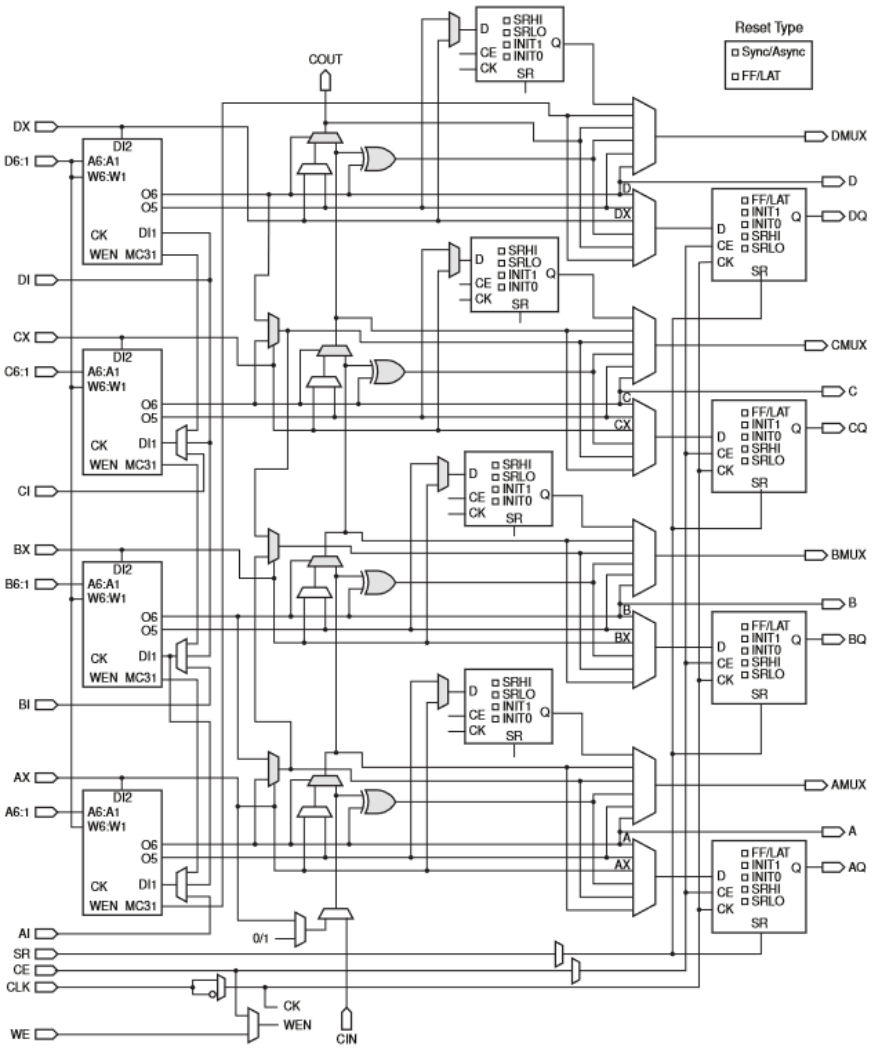


Σχήμα 3.11: CLB και Slices με εισόδους και εξόδους Carry

- Τέσσερα LUTs
- Οκτώ στοιχεία αποθήκευσης
- Πολυπλέκτης που χρησιμοποιούνται σε ευρείες λειτουργίες.
- Carry Logic

Όπως αναφέρθηκε παραπάνω, όλες οι πτυχές επιτρέπουν στα Slices να παρέχουν αριθμητικές συναρτήσεις και λογική, μαζί με επιπλέον εργασίες όπως αποθήκευση, κατανεμημένη μνήμη RAM και καταχωρητές μετατόπισης.

Το σχήμα 3.12 δείχνει την αρχιτεκτονική μνήμης που υλοποιείται από το SLICE – M και το σχήμα SLICE – L δείχνει όλα τα στοιχεία που αποτελούν μέρος οποιασδήποτε αρχιτεκτονικής SLICE. Να θυμάστε ότι κάθε CLB μπορεί να περιέχει δύο Slice τύπου SLICE– L ή συνδυασμό και των δύο.



Σχήμα 3.12: Αρχιτεκτονική Slice M

3.3.1 Look-Up Table (LUT)

Όπως αναφέρθηκε, το LUT στην οικογένεια των σειρών 7 μπορεί να χρησιμοποιήσει πίνακες αναζήτησης των 6 bit. Από το Σχήμα 4.12 αυτές οι μεμονωμένες είσοδοι A1 έως A6, με τις εξόδους O5 και O6, μπορούν να υλοποιήσουν έως και τέσσερις γεννήτριες συναρτήσεων (A έως D) σε ένα μόνο Slice. Η παρακάτω λίστα παρουσιάζει μερικές από τις δυνατότητες των LUT.

- 6-bit boolean συναρτήσεις
- Δύο δυαδικές συναρτήσεις 5-bit με κοινόχρηστες εισόδους μεταξύ τους.
- Έως δύο συναρτήσεις boolean με τρεις ή δύο εισόδους.

Κατά το σχεδιασμό ενός κυκλώματος, ο αριθμός των εισόδων είναι ένα ουσιαστικό θέμα στην περίπτωση των LUT και μπορούν να αλλάξουν τη διάδοση του σήματος τους ανεξάρτητα από τη συνάρτηση που εφαρμόζεται. Συνήθως, οι έξοδοι ενός LUT είναι A, B, C, D, O6 ή οποιοσδήποτε από τους παρακάτω πολυπλέκτες AMUX, BMUX, CMUX, DMUX και η έξοδος O5 μπορεί να χειριστεί την έξοδο όπως απαιτείται. Επιπλέον, τρεις επιπλέον πολυπλέκτες, οι F7AMUX, F7BMUX και F7CMUX, μπορούν να συνδυαστούν έως και τέσσερις μεμονωμένες λειτουργίες, για να παρέχουν λειτουργίες με έως και οκτώ εισόδους χρησιμοποιώντας ένα μόνο Slice. Εάν χρειάζονται περισσότερες από οκτώ είσοδοι, ένας πολυπλέκτης με το όνομα F8MUX συνδυάζει όλα τα LUT σε ένα Slice.

Κεφάλαιο 4

Στην ενότητα αυτή θα γίνει μια εκτενής αναφορά στον σχεδιασμό της καμπύλης και πιο συγκεκριμένα στην Curve25519 σε επίπεδο H/W. Ποιο αναλυτικά θα αναφερθούν τα διαφορετικά στοιχεία (components), η ξεχωριστή λειτουργεία τους και η διαδικασία υλοποίησης τους για την καμπύλη.

4.1 Πυρήνας καμπύλης Curve25519

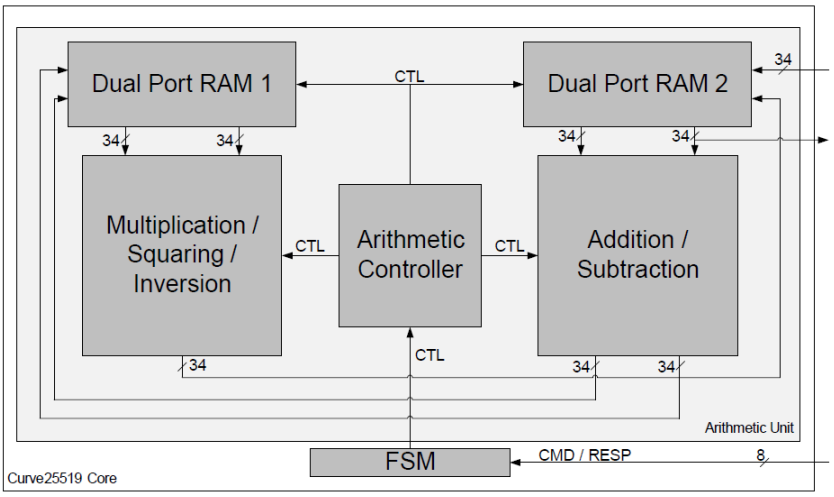
Η υλοποίηση για την καμπύλη Curve25519 έχει σχεδιαστεί για να υποστηρίζει ασύμμετρη κρυπτογραφία ως συμπληρωματική λειτουργία. Ο πυρήνας της καμπύλης είναι ικανός να εκτελέσει έναν σημειακό πολλαπλασιασμό (point multiplication) σε προβολικές συντεταγμένες.

Η χρήση με τα περισσότερα κρυπτογραφικά πρωτόκολλα, στο πυρήνας χρειάζεται επίσης να εφαρμοστεί μια τελική αντιστροφή για να μετατρέψει την έξοδο σε προβολικές συντεταγμένες σε συγγενικές συντεταγμένες. Επομένως, ο επεξεργαστής υποστηρίζει δύο βασικούς τρόπους λειτουργίας: Είτε μια συνδυασμένη συνάρτηση double and add είτε έναν απλό modular πολλαπλασιασμό.

Η πρόσβαση στον modular πολλαπλασιασμό απαιτείται για την τελική αναστροφή που βασίζεται στο μικρό θεώρημα του Fermat, δηλαδή αντιστροφή ενός στοιχείου στο πεδίου $a \in \mathbb{F}_p$ με υπολογισμό του $a^{p-2} mod\ p$. Το οποίο θεώρημα θα αναλυθεί σε παρακάτω ενότητα αναλυτικά.

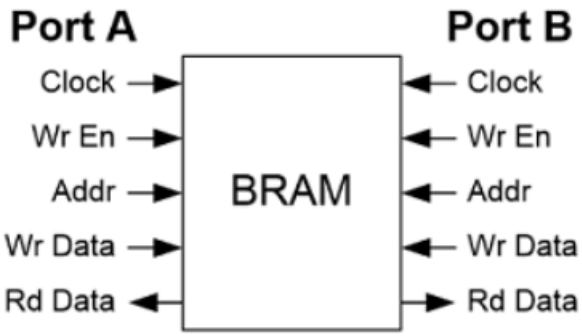
Επιπλέων, για την αποφυγή επιθέσεων χρονισμού, η αριθμητική μονάδα εκτελεί τον σημειακό πολλαπλασιασμό εκτελώντας συνολικά 255 πράξεις double and add και 266 επαναλαμβανόμενους πολλαπλασιασμούς για την αντιστροφή, με σταθερό χρόνο εκτέλεσης και για τις δύο.

Στην υλοποίησή μας ακολουθούμε αρκετές από τις προτάσεις σχεδιασμού για υλοποιήσεις λογισμικού όπως δίνονται στην αρχική εργασία για τους υπολογισμούς Diffie-Hellman πάνω από το Curve25519 [9]. Ειδικότερα, κάθε πρόσθεση ή αφαίρεση ακολουθείται πάντα από έναν επόμενο πολλαπλασιασμό και πάλι διαδέχεται μια επόμενη πρόσθεση ή αφαίρεση. Αυτά τα γεγονότα οδήγησαν στο σχέδιο που παρουσιάζεται στην Εικόνα 4.1 χρησιμοποιώντας δύο BRAM διπλής θύρας σε διαμόρφωση «πεταλούδας»



Εικόνα 4.1 Επισκόπηση του πυρήνα Curve25519

Η διαμόρφωση Dual Port Block RAM (Εικόνα 4.2) συμπεριφέρεται ακριβώς με τον ίδιο τρόπο όπως η διαμόρφωση μιας Single port, με τη διαφορά ότι έχει άλλη διαθέσιμη θύρα για ανάγνωση και εγγραφή δεδομένων. Τόσο η port A όσο και η port B συμπεριφέρονται ακριβώς το ίδιο. Η Θύρα A μπορεί να εκτελέσει μια ανάγνωση στη Διεύθυνση 0 στον ίδιο κύκλο ρολογιού που η θύρα B γράφει στη διεύθυνση (π.χ 200). Επομένως, μια DPRAM μπορεί να εκτελέσει μια εγγραφή σε μια διεύθυνση ενώ διαβάζει από μια εντελώς διαφορετική διεύθυνση.



Εικόνα 4.2 Dual Port Block RAM

Πιο συγκεκριμένα, η πρώτη BRAM λαμβάνει μόνο τα αποτελέσματα της μονάδας πρόσθεσης ή αφαίρεσης και παρέχει την είσοδο στον πολλαπλασιασμό ενώ η δεύτερη BRAM αποθηκεύει το αποτέλεσμα πολλαπλασιασμού και τροφοδοτεί τη μονάδα πρόσθεσης. Με αυτόν τον τρόπο

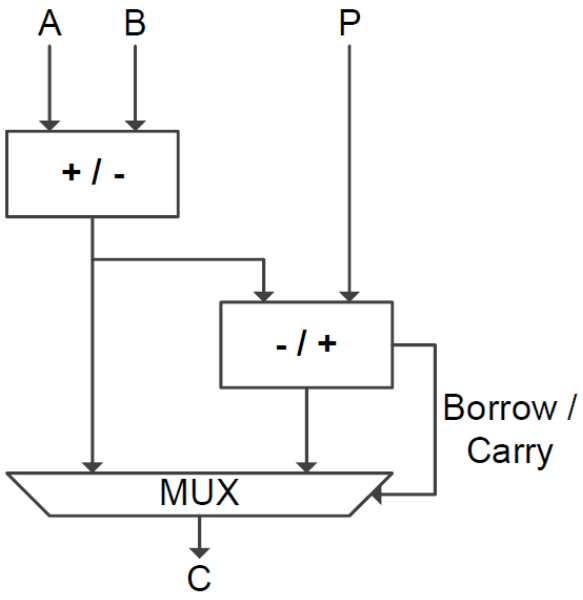
ενεργοποιείται η παράλληλη λειτουργία και μπορούν να αποφευχθούν οι καθυστερήσεις στο pipeline κατά τη φόρτωση και την επιστροφή με ελάχιστη επιβάρυνση στο χρόνο φορτώσεις δεδομένων.

4.1.1 Μονάδα άθροισης – Modular addition unit C = A ± B (mod P)

Κεντρικό στοιχείο της μονάδας πρόσθεσης και αφαίρεσης που υπολογίζει c = a ± b mod p είναι δύο μπλοκ DSP που υποστηρίζουν πολλαπλασιασμούς 25x18 bit και προσθέσεις, αφαιρέσεις ή πράξεις συσσώρευσεις έως 48 bit.

Το πρώτο DSP εκτελεί πάντα την κύρια πράξη (δηλαδή, αφαίρεση ή πρόσθεση c’ = a±b), ενώ το δεύτερο μπλοκ DSP υπολογίζει μια πρόβλεψη για μειωμένο αποτέλεσμα κατά $c'' = c' \mp$ p. Και τα δύο, το c’ και το c’’ αποθηκεύονται στην πρώτη BRAM (βλέπε Εικόνα 4.1) και διακρίνονται από μια σημαία (flag) που λαμβάνεται από την προηγούμενη κρατούμενο/δανεικό στις πράξεις που υποδεικνύει σε ποιους καταχωρητές αποθηκεύεται το σωστό αποτέλεσμα. Συνολικά, η αρθρωτή (modular) πρόσθεση/αφαίρεση απαιτεί 10 κύκλους ρολογιού που μπορούν να εκτελεστούν παράλληλα με οποιαδήποτε λειτουργία πολλαπλασιασμού Σχήμα 5.1. Έτσι, αξιοποιώντας την εναλλασσόμενη λειτουργία.

Όπως αναφέρθηκε παραπάνω, η καθυστέρηση για πρόσθεση ή αφαίρεση απορροφάτε πλήρως στην καθυστέρηση για έναν ταυτόχρονο modular πολλαπλασιασμό.



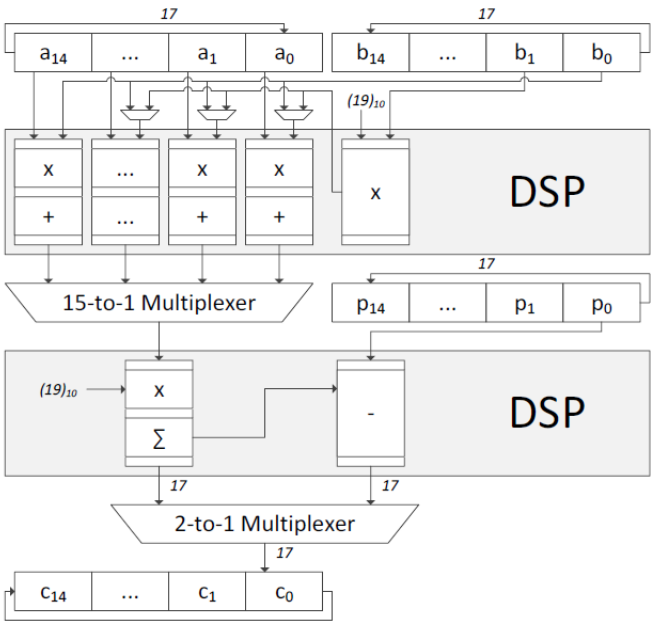
Σχήμα 4.1 Μονάδα αθροιστή / αφαιρέτη

4.1.2 Μονάδα πολλαπλασιασμού – Modular Multiplication/Squaring:C = A x B (mod P)

Το μεγαλύτερο συστατικό της αριθμητικής μονάδας (ALU) είναι η μονάδα πολλαπλασιασμού και βασίζεται σε 18 μπλοκ DSP. 15 μπλοκ χρησιμοποιούνται για τον υπολογισμό των μερικών γινομένων, ένα για μια «προμείωση» και δύο για την τελική αναγωγή (modular reduction p).

Ένας modular πολλαπλασιασμός μπορεί να υπολογιστεί σε 55 κύκλους ρολογιού από τους οποίους απαιτούνται 34 κύκλοι για τον πραγματικό πολλαπλασιασμό και οι υπόλοιποι για φόρτωση και αποθήκευση δεδομένων.

Λόγω του σχεδιασμού που φαίνεται στο Σχήμα 4.2, ο υπολογισμός των επιμέρους προϊόντων (στάδιο 1) μπορεί να παρεμβληθεί με το βήμα αναγωγής (στάδιο 2) κατά τρόπο pipeline. Έτσι, μια επόμενη λειτουργία πολλαπλασιασμού μπορεί ήδη να ξεκινήσει ξανά μόλις το πρώτο στάδιο (μερικά γινόμενα) ολοκληρώσει τον προηγούμενο πολλαπλασιασμό. Έτσι, μόνο ο πρώτος πολλαπλασιασμός παίρνει τους πλήρεις 55 κύκλους ρολογιού, κάθε επόμενος πολλαπλασιασμός γίνεται διαθέσιμος με καθυστέρηση 17 κύκλους ρολογιού. Δεδομένου ότι οι εξαρτήσεις δεδομένων πρέπει να ληφθούν υπόψη. Το συνδυασμός των βημάτων double and add για το Curve25519 διαρκεί 255 κύκλους συνολικά.



Σχήμα 4.2 Μονάδα πολλαπλασιασμού

Πιο αναλυτικά η μονάδα λαμβάνει δύο αριθμούς, που αντιπροσωπεύουν τους τελεστές για modular πολλαπλασιασμό. Αυτοί οι τελεστές μπορούν να τροφοδοτηθούν σε καταχωρητές εντός της μονάδας, και να συμμετάσχουν στην διαδικασία του pipeline. Η βασική λειτουργικότητα περιλαμβάνει τη μονάδα πολλαπλασιαστή DSP (επεξεργαστές ψηφιακού σήματος) όπως αναφέρθηκε στην προηγούμενη ενότητα αλλά και πιο πάνω. Αυτά τα DSP εκτελούν τον πολλαπλασιασμό των δύο τελεστών αλλά χειρίζονται τα bit λαμβάνοντας υπόψη την τιμή του modulo. Στο εσωτερικό τμήμα υπάρχει μια μικρή μονάδα διαχείρισης πιθανών κρατουμένων (Carry Logic Element). Κατά τη διάρκεια του πολλαπλασιασμού, ενδέχεται να υπάρχουν υπερχείλιση σε bit. Η μονάδα αυτή φροντίζει και διασφαλίζει αυτά τα bits ώστε το αποτέλεσμα να είναι εντός του επιθυμητού εύρους. Ο πολυπλέκτης παίζει ρόλο στην επιλογή συγκεκριμένων bits από έναν από τους τελεστές κατά τη διάρκεια της διαδικασίας πολλαπλασιασμού. Σε συνδαισμός με τους καταχωρητές ολίσθησης μπορεί να χρησιμοποιηθούν για την προσωρινή αποθήκευση τελεστών ή ενδιάμεσων αποτελεσμάτων κατά τη διάρκεια του υπολογισμού. Τέλος το τελικό αποτέλεσμα του αρθρωτού πολλαπλασιασμού, που είναι και πάλι ένας ακέραιος modulo πρώτος αριθμός, αποθηκεύεται σε έναν καταχωρητή ή αποστέλλεται έξω από τη μονάδα.

4.2 Montgomery ladder

Το Montgomery ladder [21] είναι μια εξαιρετικά απλή μέθοδος υπολογισμού κλιμακωτών πολλαπλάσιων σημείων σε μια ευρεία κατηγορία ελλειπτικών καμπυλών. Βρίσκει εφαρμογή σε αρκετά κρυπτοσυστήματα. Είναι μια εναλλακτική λύση στην προσέγγιση που χρησιμοποιεί ο αλγόριθμος double-and-add.

Αν και ο Montgomery ladder απαιτεί περισσότερο χρόνο για να υπολογίσει έναν πολλαπλασιασμό σημείων, το γεγονός ότι χρησιμοποιεί ένα σταθερό σύνολο διπλασιασμού και πρόσθεσης σημείων σημαίνει ότι είναι ανθεκτικό σε επιθέσεις χρονισμού και ανάλυσης ισχύος που προσπαθούν να ανακτήσουν bits του μυστικού κλειδιού k.

Ορίζουμε τις ακολουθίες (x_1, x_2,…) και (x_1, x_2,…), ξεκινώντας με τα x_1, z_1, A και τις ακόλουθες εξισώσεις για $n \geq 1$.

$$\boldsymbol{X_{2n} = (X_n^2 - Z_n^2)^2,} \quad (4,1) \qquad \boldsymbol{X_{2n+1} = 4(X_n X_{n+1} - Z_n Z_{n+1})^2 Z_1} \quad (4,2)$$

$$\boldsymbol{Z_{2n} = 4X_nZ_n(X_n^2 + AX_nZ_n + Z_n^2)}, (4,3) \quad \boldsymbol{Z_{2n+1} = 4(X_nX_{n+1} - Z_nZ_{n+1})^2X_1} \quad (4,4)$$

Τότε τα σημεία είναι τα n πολλαπλάσια των σημείων στην καμπύλη Montgomery $By^2 = x^3 + Ax^2 + x$.

$$\left(\frac{\boldsymbol{X_n}}{\boldsymbol{Z_n}}, \pm\sqrt{\frac{\boldsymbol{1}}{\boldsymbol{B}}\left(\frac{\boldsymbol{X_n^3}}{\boldsymbol{Z_n^3}} + \boldsymbol{A}\frac{\boldsymbol{X_n^2}}{\boldsymbol{Z_n^2}} + \frac{\boldsymbol{X_n}}{\boldsymbol{z_n}}\right)}\right)$$

Στην συνέχεια οι παραπάνω εξισώσεις βελτιστοποιημένες υπολογίζουν τα (X_n, Z_n) χρησιμοποιώντας 11 πολλαπλασιασμούς για κάθε bit του n. Έχουν την παρακάτω μορφή.

$$\boldsymbol{X_{2n} = (X_n - Z_n)^2(X_n + Z_n)^2}$$

$$\boldsymbol{Z_{2n} = [(X_n + Z_n)^2 - (X_n - Z_n)^2] * [(X_n + Z_n)^2 + \frac{A-2}{4}[(X_n + Z_n)^2 - (X_n - Z_n)^2}$$

$$\boldsymbol{X_{2n+1} = [(X_n - Z_n)(X_{n+1} + Z_{n+1}) + (X_n + Z_n)(X_{n+1} - Z_{n+1})]^2Z_1}$$

$$\boldsymbol{Z_{2n+1} = [(X_n - Z_n)(X_{n+1} + Z_{n+1}) - (X_n + Z_n)(X_{n+1} - Z_{n+1})]^2X_1}$$

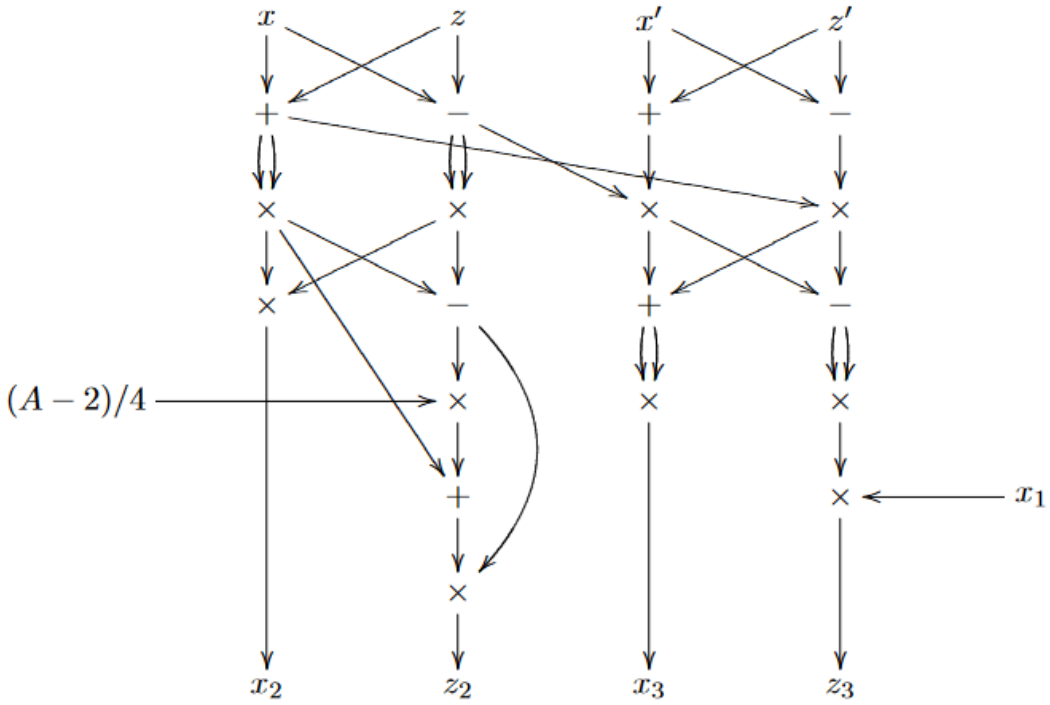
Ο Montgomery εισήγαγε αυτές τις καμπύλες και βελτιστοποίησε τους τύπους σε επιστημονική δημοσίευση [22] το 1987.

➢ Βήματα για Montgomery ladder

Ο παρακάτω αλγόριθμος δηλώνει τους τύπους συντεταγμένων x του Montgomery και οι τύποι λειτουργούν όποτε $Q - Q' / \in \{\infty, (0, 0)\}$.

Το παρακάτω διάγραμμα συνοψίζει τους τύπους του Montgomery ladder και τον βέλτιστο τρόπο υλοποίησης της λειτουργίας double and add, στην περίπτωση $z_1 = 1$.

Αρχικά το x/z και x′ /z′ είναι οι συντεταγμένες x των σημείων Q, Q′. Τα x_2 / z_2 είναι η συντεταγμένη x του 2Q, το x_1 είναι η συντεταγμένη x του (Q − Q′) και x_3 / z_3 είναι η συντεταγμένη x του (Q + Q′) .



Εικόνα 4.3: Οι βελτιστοποιημένοι τύποι του Montgomery για διπλασιασμό και πρόσθεση, υποθέτοντας Z1 = 1

Όπου:

Θεώρημα 1 [15] :

Έστω p πρώτος αριθμός με $p \geq 5$. Έστω A ακέραιος ώστε το $A^2 - 4$ να μην είναι τετράγωνο του modulo p. Ορίζουμε E: $y^2 = x^3 + Ax^2 + x$ την ελλειπτική καμπύλη οριζόμενη στο πεδίο $\mathbb{F}_p$.

- $x_2 = (x^2 - z^2)^2 = (x - z)^2 (x + z)^2$

- $z_2 = 4xz(x^2 + Axz + z^2) =$

$$= [(x + z)^2 - (x - z)^2] \cdot \{(x + z)^2 + \frac{A-2}{4} [(x + z)^2 - (x - z)^2]\}$$

Τότε $X(2Q) = x_2 / z_2$ για κάθε $Q \in E(\mathbb{F}_{p^2})$ όπως και το $X(Q) = x / z$. Το x / z είναι το πηλίκο του x και z που ορίζεται πεδίο $\mathbb{F}_p$ εάν $z \neq 0$. Αν $x \neq 0$ και $z = 0$ τότε τείνει στο ∞. Τέλος αν x=z=0 τότε απροσδιόριστο.

✓ Περίπτωση 1

Για $z = 0$

Έχουμε $x_2 = x^4 \neq 0$ και $z_2 = 0$. Επίσης $X(Q) = x/0 = \infty$ άρα $Q = \infty$ και $2Q = \infty$, επομένως $X(2Q) = \infty = x_2/0 = x_2/z_2$.

✓ Περίπτωση 2

Για $z \neq 0$ και $x = 0$

Έχουμε $x = x^4 \neq 0$ και $z=0$. Επίσης $X(Q) = 0/z = 0$ άρα $Q = (0, 0)$ και $2Q = \infty$, επομένως $X(2Q) = \infty = x_2/0 = x_2/z_2$.

✓ Περίπτωση 3

Για $z \neq 0$ και $x \neq 0$

Έχουμε $Q = (x/z, y)$ για κάποιο $y \in \mathbb{F}_{p^2}$ ικανοποιώντας το $y^2 = (x/z)^2 + A(x/z)^2 + (x + z)$ και έτσι $4y^2z^4 = 4(x^3z + Ax^2z^2 + xz^3) = z_2$

Το μη τετράγωνο του $A^2 - 4$ συνεπάγεται ότι $y \neq 0$ άρα $z_2 \neq 0$. Επίσης $X(2Q) = [(x/z)^2 - 1)^2] / 4y^2$ σύμφωνα με τον διπλασιασμό, έτσι $z_2X(2Q) = z^4 [(x/z)^2 - 1)^2 = (x^2 - z^2)^2 = x_2$.

Θεώρημα 2 [15]:

Σε συνέχεια με το θεώρημα 1 τα $x, z, x', z', x_1, z_1 \in \mathbb{F}_p$ με $(x, z) \neq (0, 0)$, $(x', z') \neq 0$ και $z_1 \neq 0$.

Ορίζουμε

- $\mathbf{x_3 = 4(xx' - zz')^2 z_1 = [(x - z)\cdot(x' + z') + (x + z)\cdot(x' - z')]^2 \cdot z_1}$

- $\mathbf{z_3 = 4(xz' - zx')^2 x_1 = [(x - z)\cdot(x' + z') - (x + z)\cdot(x' - z')]^2 \cdot x_1}$

Τότε $X(Q + Q') = x_3/z_3$ για όλα $Q, Q' \in E(\mathbb{F}_{p^2})$ όπως και το $X(Q) = x/z$, $X(Q') = x'/z'$ και $X(Q-Q') = x_1/z_1$

✓ <u>Περίπτωση 1</u>

<u>Για $Q = Q'$</u>

Έχουμε $X(Q - Q') = X(\infty) = \infty$, έτσι $z_1 = 0$

✓ <u>Περίπτωση 2</u>

<u>Για $Q = \infty$</u>

Έχουμε $z = 0$ και $x \neq 0$, επίσης $X(Q - Q') = X(- Q') = X(Q')$ επομένως $x_1/z_1 = x'/z'$ και $x' , z' \neq 0$.

Το $x_3 = 4(xx')^2 z_1$ και $z_3 = 4(xz')^2 x_1$ επομένως $x_3/z_3 = (x'/z')^2 \cdot (z_1/x_1) = x'/z' = X(Q') = X(Q + Q')$.

✓ <u>Περίπτωση 3</u>

<u>Για $Q' = \infty$</u>

Έχουμε $z' = 0$ και $x' \neq 0$, επίσης $X(Q - Q') = X(Q)$ επομένως $x_1/z_1 = x/z$ και $x, z \neq 0$.

Το $x_3 = 4(xx')^2 z_1$ και $z_3 = 4(zx')^2 x_1$ επομένως $x_3/z_3 = (x/z)^2 \cdot (z_1/x_1) = x/z = X(Q) = X(Q + Q')$.

✓ <u>Περίπτωση 4</u>

<u>Για $Q = Q'$</u>

Έχουμε $X(Q') = X(Q)$ επομένως $x/z = x'/z'$ έτσι $xz' = zx'$ και $z_3 = 0$.

Υποθέτουμε ότι $x_3 = 0$. Τότε $(x - z)(x' + z') + (x + z)(x' - z') = 0$ και $(x - z)(x' + z') - (x + z)(x' - z') = 0$ έτσι $(x - z)(x' + z') = 0$ και $(x + z)(x' - z') = 0$.

Αν $x + z \neq 0$. Τότε $x' - z' = 0$ έτσι $x' + z' = 2x' \neq 0$ συνεπώς $x - z = 0$. Δηλαδή $X(Q) = 1$ και $X(Q') = 1$.

Αλλιώς $x = -z$ έτσι $x - z = 2x \neq 0$ συνεπώς $x' = -z$.

Δηλαδή $X(Q) = -1$ και $X(Q') = -1$.

Το $X(Q - Q') = X(2Q) = (X(Q)^2 - 1)^2/ \ldots = (1 - 1)^2/ \ldots = 0$ σύμφωνα με τον διπλασιασμό, έτσι $x_1 = 0$. Συνεπώς $x_3 \neq 0$, και $x_3 / z_3 = \infty = X(\infty) = X(Q + Q')$.

✓ <u>Περίπτωση 5</u>

<u>Για $Q \neq \infty$, $Q' \neq \infty$, $Q \neq Q'$ και $Q \neq -Q'$</u>

Έχουμε $z \neq 0$, $z' \neq 0$, $x/z \neq x'/z'$ έτσι $z_3 \neq 0$. Βρίσουμε το $y, y' \in E (\mathbb{F}_{p^2})$ όπως και $Q = (x/z, y)$ και $Q' = (x'/z', y')$.

Ορίζουμε τα $\alpha = (x'/z') - (x/z)$ και $\beta = A + (x/z) + (x'/z')$.

Τότε $X(Q + Q') = [(y' - y)/\alpha)^2] - \beta$, εξ ορισμού το $Q \pm Q'$ έτσι.

$X(Q + Q') \cdot X(Q - Q') = \beta^2 - 2\beta[(y')^2 + y^2]/\alpha^2 + [(y')^2 - y^2)^2/\alpha^4$.

<u>Αντικαθιστούμε το</u>

$y^2 = (x/z)^3 + A(x/z)^2 + (x/z)$ και το $(y')^2 = (x'/z')^3 + A(x'/z')^2 + (x'/z')$

Η παραπάνω σχέση απλοποιείται ως εξής

$X(Q + Q') \cdot X(Q - Q') = (xx' - zz')^2/ (xz' - x'z)^2$,

Τέλος $X(Q + Q') = (xx' - zz')^2 z_1 / (xz' - x'z)^2 x_1 = x_3/z$

Τα παραπάνω θεωρήματα συνοψίζουν την αλγεβρική λογική που υπάρχει πίσω από την λειτουργεία του Montgomery ladder.

Με μια γρήγορη ματιά μπορεί να δει κανείς ότι υπάρχουν 4 πράξεις τεραγονικού (διπλά βέλη), ένας πολλαπλασιασμός με το (A-2)/4 και επιπλέων 5 άλλοι πολλαπλασιασμοί. Ακόμα υπάρχουν 4 προσθέσεις και αφαιρέσεις , από τις οποίες καμία δεν παράγει είσοδο σε άλλη πρόσθεση/ αφαίρεση.

Η παράμετρος (A-2)/4 είναι ένας σταθερός ακέραιος, που βοηθάει στον πολλαπλασιασμό ενός μεγάλου με έναν μικρότερο ακέραιο. Η τιμή του εξαρτάτε από ην επιλογή του A. Η υλοποίηση αυτή είναι πιο οικονομική γιατί το (x + z) και το (x – z) επαναχρησιμοποιούνται μεταξύ της διαφορικής πρόσθεσης και του διπλασιασμού.

Για την καμπύλη (curve 25519), οτι αφορά την σταθερά A, ο Montgomery προτείνει να ληφθεί το (A-2)/4 ως ένας μικρός ακέραιος για να επιταχυνθεί ο πολλαπλασιασμός κατά (A-2)/4. Αυτό δεν έχει επίδραση σε επίπεδο ασφάλειας. Οι μικρότερες θετικές επιλογές για το A είναι 358990, 464586 και 486662. Ο Bernstein απέρριψε το A = 358990 επειδή είναι ένας από τους πρώτους που είναι ελαφρώς μικρότερος από τον 2^{255}. Απέρριψε το 464586 για το τον ίδιο λόγο. Έτσι κατέληξε στο A = 486662

4.2.1 Montgomery ladder (VHDL)

Ακολουθώντας την προηγούμενη θεωρητική ανάλυση για τον αλγόριθμο, υλοποιήσαμε σε VHDL τον παρακάτω κώδικα (Κώδικας 4.1) [23].

Πιο συγκεκριμένα δηλώνουμε τις τιμές και τα ζεύγη x, z και δηλώνουμε σαν σταθερές το modulo p, μαζί με την σταθερά A. Δημιουργούμε ενδιάμεσα σήματα/ μεταβλητές με σκοπό να εξασφαλίσουμε την βέλτιστη απόδοση των πράξεων και προσωρινή αποθήκευση των αποτελεσμάτων εμείς όσο διατρέχουμε τον αλγόριθμο. Με το σήμα reset και όταν είναι λογικό «1» έχουμε αρχικοποίηση των τιμών και του αλγορίθμου. Όταν είναι λογικό «0» ξεκινάει η διαδικασία της «σκάλας» όπως βλέπουμε στην (Εικόνα 4.3) αλλά και στο κώδικα.

```
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.NUMERIC_STD.ALL;

entity Montgomery_Ladder is
    port (
        clk : in std_logic;
        reset : in std_logic;
        X2, Z2, X3, Z3, X1 : in unsigned(255 downto 0);
        X4, Z4, X5, Z5 : out unsigned(255 downto 0)
    );
end Montgomery_Ladder;

architecture Behavioral of Montgomery_Ladder is
    constant p : unsigned(255 downto 0) :=
x"0000000000000000000000000000000000000000000000000000000000039217"; -- 2^255 - 19
    constant a24 : unsigned(255 downto 0) :=
x"000000000000000000000000000000000000000000000000000000000011b7d"; -- (A-2)/4, A
= 486662
    signal T1, T2, T3, T4, T5, T6, T7, T8, T9, tmp_x5, tmp_z5, tmp_z4: unsigned(255
downto 0);
begin

    process(clk, reset)
    begin
        if reset = '1' then
            -- Reset values
            X4 <= (others => '0');
            Z4 <= (others => '0');
            X5 <= (others => '0');
            Z5 <= (others => '0');
        elsif rising_edge(clk) then
                -- Montgomery ladder algorithm
            T1 <= (X2 + Z2) mod p;
            T6 <= (T1 * T1) mod p;
            T2 <= (X2 - Z2) mod p;
            T7 <= (T2 * T2) mod p;
            T5 <= (T6 - T7) mod p;
            T3 <= (X3 + Z3) mod p;
            T4 <= (X3 - Z3) mod p;
            T9 <= (T3 * T2) mod p;
            T8 <= (T4 * T1) mod p;
            tmp_z5 <= (T8 - T9) mod p;
            tmp_x5 <= (T8 + T9) mod p;
            tmp_z5 <= (tmp_z5 * tmp_z5) mod p;
            X5 <= (tmp_x5 *  tmp_x5) mod p;
            Z5 <= (tmp_z5 * X1) mod p;
            X4 <= (T6 * T7) mod p;
            tmp_z4 <= (a24 * T5) mod p;
            Z4 <= ((tmp_z4 + T6) * T5) mod p;
        end if;
    end process;

end Behavioral;
```

Απόσπασμα κώδικα 4.1: Montgomery ladder (VHDL) [23]

4.3 Θεώρημα Fermat – Modular inverse.

Μια απαραίτητη λειτουργεία που παρουσιάζεται στην διαδικασία της υλοποίησης του αλγορίθμου, είναι η εύρεση του αντίστροφου αριθμού. Σε πολλές περιπτώσεις η modular διαίρεση είναι η μεγαλύτερη ενεργοβόρα λειτουργία που απαιτεί modular αναστροφή και τουλάχιστον έναν πολλαπλασιασμό. Στην προσέγγιση για τον πυρήνα, παρατηρήσαμε αυτή την αντιστροφή που βασίζεται σε αυτή του θεωρήματος Fermat. Η αναστροφή απαιτεί περίπου το 20% του συνολικού χρόνου ενός υπολογισμού της καμπύλης Curve25519.

Πριν μιλήσουμε για το θεώρημα, θυμηθείτε ότι κάθε θετικός ακέραιος n έχει μια μοναδική παραγοντοποίηση σε πρώτους αριθμούς, για παράδειγμα, 77 = 7·11 και 120= 2·2·2·3·5.

Λέμε ότι ένας ακέραιος αριθμός n διαιρείται με έναν άλλο ακέραιο m για να σημαίνει ότι n = k·m για κάποιον ακέραιο αριθμό k. Αυτό είναι το ίδιο με το να λέμε ότι το n διαιρούμενο με το m δεν αφήνει κανένα υπόλοιπο, ή ισοδύναμα, ότι n ≡ 0 (mod m). Σημειώστε ότι το n διαιρείται με έναν πρώτο αριθμό p ακριβώς όταν ο p είναι ένας από τους πρώτους παράγοντες του n. Επιπλέον, είναι μια ειδική ιδιότητα των πρώτων αριθμών ότι αν m και n είναι ακέραιοι και αν m·n διαιρείται με p, τότε είτε το m διαιρείται με το p είτε το n διαιρείται με το p, επειδή το p πρέπει να είναι στην πρώτη παραγοντοποίηση του m ή του n.

Για παράδειγμα αν m=14, n = 10 και p = 7 έτσι 14 · 10 = 140 διαιρείται με το 7, που σημαίνει ότι είτε το 14 είτε το 10 διαιρείται με το 7. Αυτό λειτουργεί επειδή το 7 είναι πρώτος. Από την άλλη πλευρά, σημειώστε ότι το 10 · 15 = 150 διαιρείται με το 6 (δηλαδή 150 = 6 · 25) αλλά ούτε το 10 ούτε το 15 διαιρούνται με το 6. Αυτό συμβαίνει επειδή το 6 = 2 · 3 είναι σύνθετο και ο παράγοντας, δύο βρίσκεται στο 10 ενώ ο παράγοντας τρία βρίσκεται στο 15.

Το θεώρημα του Fermat δηλώνει ότι αν ο p είναι πρώτος, τότε $a^p \equiv a(mod\ p)\ \forall\ a.$ Μια εναλλακτική μορφή δηλώνει ότι $a^{p-1} \equiv 1(mod\ p)$ όταν ο p είναι πρώτος και ο a είναι οποιοσδήποτε ακέραιος που δεν διαιρείται μ ≡ ε το p. (αυτή η τελευταία συνθήκη χρειάζεται για την εναλλακτική μορφή, αλλά όχι για τη συνήθη μορφή.) Ας δούμε τη συνήθη εκδοχή του μικρού θεωρήματος του Fermat σε πολλά βήματα. Δοκίμασέ το: $p = 3 \implies a^3 \equiv a(mod\ 3)$?

Ελέγχουμε εάν το a είναι το ίδιο πολλαπλάσιο του 3, τότε και το a·p και το a είναι ίσα με 0 (mod 3), και η ταυτότητα είναι αληθής. Επομένως, χρειαζόμαστε να ελέγξουμε μόνο τις περιπτώσεις όπου το a δεν είναι πολλαπλάσιο του 3. Μπορούμε να ξεκινήσουμε βγάζοντας όλα τα 3

από το a, έτσι ώστε να πρέπει να ελέγξουμε μόνο τις περιπτώσεις a = 1 και a = 2. Για a = 1 , είναι προφανές ότι $a^3 = 1 \equiv a(mod\ 3)$. Για a = 2, έχουμε $a^3 = 8 \equiv 2(mod\ 3)$.

Το επόμενο παράδειγμά μας είναι p = 5. Είναι αλήθεια ότι $p = 5 \Longrightarrow a^5 \equiv a(mod\ 5)$? για όλα τα a ≡ 1, 2, 3 ή 4 (mod 5), (Για α ≡0 (mod 5), και πάλι δεν χρειάζεται να ελέγξουμε τίποτα, γιατί τότε η ταυτότητα είναι προφανώς αληθινή)

• α = 1 → Okay

• $\alpha = 2 \Rightarrow a^5 = 32 \equiv 2\ (mod\ 5)$

• $\alpha = 3 \Rightarrow a^2 = 9 \equiv 4\ (mod\ 5)$

$\Rightarrow a^5 = \alpha \times \alpha^2 \times \alpha^2 \equiv 48\ (mod\ 5) \equiv 3(mod\ 5)$

• $a = 4 \Rightarrow \alpha^2 = 16 \equiv 1\ (mod\ 5)$

$$\Rightarrow a^4 \equiv 1\ (mod\ 5)$$

$$\Rightarrow a^5 \equiv a \times a^4 \equiv a \equiv 4\ (mod\ 5)$$

Θα μπορούσαμε να συνεχίσουμε να ελέγχουμε τον έναν πρώτο αριθμό μετά τον άλλο, αλλά αυτό δεν θα ήταν ποτέ αρκετό. Θα μπορούσε πάντα να υπάρχει μια αμφιβολία ότι κάπου, πέρα από τον μεγαλύτερο αριθμό που ελέγξαμε, υπήρχε ένα p για το οποίο η δήλωση δεν θα ήταν αληθινή. Εμείς χρειαζόμαστε ένα διαφορετικό επιχείρημα, που να αποδεικνύει την αλήθεια του θεωρήματος του Fermat χωρίς καμία αμφιβολία.

Θα αφήσουμε p να είναι ο πρώτος αριθμός και a να είναι οποιοσδήποτε ακέραιος. Θέλουμε να αποδείξουμε ότι $a^p \equiv a\ (mod\ p)$. Αν $a \equiv 0\ (mod\ p)$, τότε σαφώς $a^p \equiv 0\ (mod\ p)$ επίσης, και έτσι $a^p \equiv a\ (mod\ p)$ οπότε μένει μόνο να αποδειχθεί ότι $a^p \equiv a\ (mod\ p)$ στην περίπτωση που $a \not\equiv 0\ (mod\ p)$ δηλ. όπου το a δεν διαιρείται με το p.

Η λίστα των πρώτων p μη αρνητικών αριθμών, δηλαδή 0, 1, 2, . . ., p − 2, p − 1. Εξετάστε επίσης τη λίστα που παίρνουμε πολλαπλασιάζοντας κάθε στοιχείο της πρώτης λίστας με α. Αυτή η νέα λίστα είναι 0, a, 2a, . . . , (p − 2)a, (p − 1)a. Τώρα αν μειώσετε αυτούς τους νέους αριθμούς κατά modulo p, θα λάβετε την αρχική λίστα, αλλά ίσως με μια κωδικοποιημένη σειρά. Ας το κάνουμε όταν p = 7 και a = 4. Η αρχική λίστα είναι μόλις 0, 1, 2, 3, 4, 5, 6. Η νέα λίστα είναι 0, 4, 8, 12, 16, 20, 24.

Αν μειώσουμε κατά modulo 7, το νέο η λίστα γίνεται 0, 4, 1, 5, 2, 6, 3, η οποία είναι μόνο η αρχική λίστα σε κωδικοποιημένη σειρά

Ας δούμε γιατί αυτό ισχύει για οποιονδήποτε πρώτο p και κάθε αριθμό a που δεν διαιρείται με το p. Θέλουμε να δείξουμε ότι τα στοιχεία των 0, a, 2a, . . . , (p − 2)a, (p − 1)a αναγωγή συντελεστή p σε 0, 1, 2, . . . , p−2, p−1, αν και όχι απαραίτητα σε αύξουσα σειρά. Υποστηρίζουμε ότι θα αρκεί να δείξουμε ότι κανένα στοιχείο της νέας λίστας δεν είναι ισοδύναμο με οποιοδήποτε άλλο στοιχείο της νέας λίστας κατά modulo p. Αυτό θα είναι αρκετό γιατί όλοι οι αριθμοί αφήνουν ένα υπόλοιπο από το 0 στο p – 1. Όταν διαιρέσουμε με το p, και εάν κανένας αριθμός στη νέα λίστα δεν είναι ισοδύναμοι modulo p, τότε οι αριθμοί p στη νέα λίστα θα αφήσουν p διαφορετικά υπόλοιπα. Τα οποία πρέπει να είναι όλοι οι αριθμοί από το 0 έως το p − 1 με κάποια σειρά

Εδώ είναι η απόδειξη ότι κανένα στοιχείο στη νέα λίστα δεν είναι ισοδύναμο modulo p: Δύο στοιχεία της νέας λίστας, πείτε ja και ka με $0 \leq j < k \leq p - 1$. Θα υποθέσουμε ότι $ja \equiv ka \pmod{p}$. Αν υποθέσουμε αυτό, τότε $ka - ja \equiv 0 \pmod{p}$.

Τότε $(k - j)a \equiv 0 \pmod{p}$, ώστε το (k − j)a να διαιρείται με τον πρώτο p. Εφόσον το a θεωρείται ότι δεν διαιρείται με το p, αυτό αναγκάζει το k − j να διαιρείται με το p. Αλλά στη συνέχεια σημειώστε ότι $0 \leq j < k \leq p - 1$, που σημαίνει ότι $0 < k - j \leq p - 1$. Αλλά δεν υπάρχει θετικός αριθμός που να είναι μικρότερος ή ίσος με p − 1 και ταυτόχρονα διαιρείται με το p. Άρα η υπόθεση μας ότι $ja \equiv ka \pmod{p}$ οδηγεί σε μια αντίφαση, άρα δεν πρέπει να είναι αληθής. Στην πραγματικότητα, όλοι οι αριθμοί στη νέα λίστα είναι διαφορετικά modulo p.

Τώρα λοιπόν γνωρίζουμε ότι 0, a, 2a, . . . , (p − 2)a, (p − 1)a μειώνει το modulo p σε μια λίστα p διαφορετικών υπολειμμάτων, άρα κάθε πιθανό υπόλοιπο 0, 1, 2, . . . , p − 2, p − 1, πρέπει να εμφανιστεί μία φορά.

Δηλαδή 0, a, 2a, . . . , (p − 2)a, (p − 1)a μειώνει το modulo p στη λίστα 0, 1, 2, . . . , p − 2, p − 1, ίσως σε ανακατεμένη σειρά. Αφαιρούμε τις μηδενικές εγγραφές από τις λίστες και συμπεραίνουμε ότι a, 2a, . . . , (p − 2)a, (p − 1)a μειώνει το modulo p σε 1, 2, . . . , p − 2, p − 1, όχι απαραίτητα σε σειρά. Εφόσον οι δύο λίστες έχουν τα ίδια στοιχεία modulo p, έχουν τα ίδια προϊόντα modulo p:

$$a \times 2a \times \ldots \times (p - 2)a \times (p - 1)a \equiv 1 \times 2 \times \ldots \times (p - 2) \times (p - 1) \pmod{p},$$

που μπορούμε να αναδιατάξουμε βάζοντας όλους τους συντελεστές p − 1 του a μπροστά

$a^{p-1} \times 1 \times 2 \times \ldots \times (p-2) \times (p-1) \equiv 1 \times 2 \times \ldots \times (p-2) \times (p-1) \pmod{p}$,

ή ισοδύναμα, με αφαίρεση

$a^{p-1} \times 1 \times 2 \times \ldots \times (p-2) \times (p-1) - 1 \times 2 \times \ldots \times (p-2) \times (p-1) \equiv 0 \pmod{p}$, ή εξίσου

$(a^{p-1} - 1) \times 1 \times 2 \times \ldots \times (p-2) \times (p-1) \equiv 0 \pmod{p}$,

που είναι το ίδιο με το να λέμε ότι $(a^{p-1} - 1) \times 1 \times 2 \times \ldots \times (p-2) \times (p-1)$ διαιρείται με τον πρώτο p.

Λοιπόν, κανένας από τους παράγοντες 1, 2, . . . , το $p - 1$ διαιρείται με το p αφού πρόκειται για αριθμούς μικρότερους του p. Άρα πρέπει να έχουμε $ap-1 - 1$ διαιρούμενο με τον πρώτο p, δηλαδή, $a^{p-1} - 1 \equiv 0 \pmod{p}$, δηλ. $a^{p-1} \equiv 1 \pmod{p}$. Αυτή είναι η εναλλακτική μορφή του μικρού θεωρήματος του Fermat, το οποίο ισχύει μόνο όταν το a δεν διαιρείται με το p (όπως έχουμε υποθέσει). Για να πάρετε τη συνηθισμένη μορφή, πολλαπλασιάζουμε και τις δύο πλευρές με το a για να λάβετε $a^p \equiv a \pmod{p}$ και αποδεικνύετε το θεώρημα του Fermat.

4.3.1 Θεώρημα Fermat υλοποίηση σε H/W.

Στην περίπτωση της διπλωματικής εργασίας η υλοποίηση του Fermat πραγματοποιήθηκε σε γλώσσα προγραμματίσου VHDL. Η οποία περιγραφή και σχεδιάζει την λειτουργικότατα του υλικού.

Με αφορμή την θεωρία και την μεθοδολογία για την εύρεση του αντιστρόφου (modulo inverse) δημιουργήσαμε μια πιο απλή εκδοχή με αρκετά γρήγορο αποτέλεσμα και μικρό σε απαιτήσεις πώρων όσον αφορά το hardware

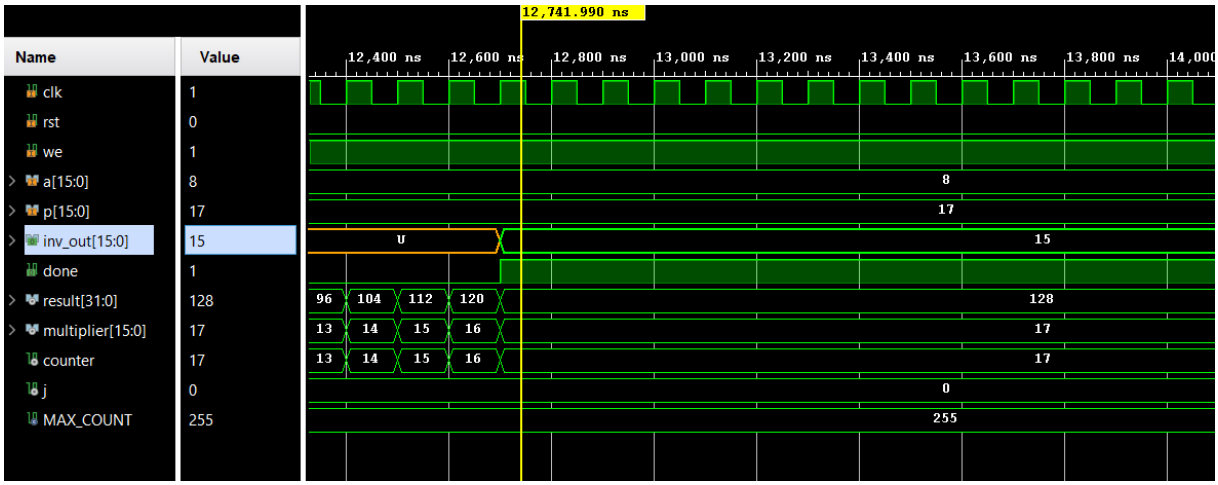
Πιο συγκεκριμένα όπως βλέπουμε και στο παρακάτω τμήμα κώδικα, κατά την επαναφορά (rst = '1'), αρχικοποιούμε τον μετρητή (counter), τον πολλαπλασιαστή (multiplier) και το σήμα ολοκλήρωσης της διαδικασίας (done). Στην ανερχόμενη ακμή του ρολογιού (rising_edge(clk)), εάν το σήμα ενεργοποίησης εγγραφής (we) είναι ενεργό (we = '1'), αυξάνει τον πολλαπλασιαστή και υπολογίζει το αποτέλεσμα (α * multiplier) και αυξάνουμε επίσης τον μετρητή. Εάν το «προσωρινό» αποτέλεσμα με το modulo p ισούται με 1, θέτει το αρθρωτό αντίστροφο στην έξοδο και ορίζει το σήμα done = 1 ως ολοκλήρωση της διαδικασίας .

```vhdl
library IEEE;
use IEEE.STD_LOGIC_1164.ALL;
use IEEE.STD_LOGIC_UNSIGNED.ALL;
use IEEE.numeric_std.all;

entity c is
    Port ( clk : in STD_LOGIC;
           rst, we : in STD_LOGIC;
          a : in STD_LOGIC_VECTOR(15 downto 0); -- input a
          p : in STD_LOGIC_VECTOR(15 downto 0); -- input prime number p
          inv_out : out STD_LOGIC_VECTOR(15 downto 0); -- output modular
          done : out STD_LOGIC);
end ModularInverse;

architecture Behavioral of ModularInverse is
    signal result : STD_LOGIC_VECTOR(31 downto 0);
    signal multiplier : STD_LOGIC_VECTOR(15 downto 0) := (others => '0');
    signal counter, j : integer := 0;
    constant MAX_COUNT : integer := 255; -- maximum count for 8-bit numbers

begin

    process(clk, rst)
    begin
        if rst = '1' then
            counter <= 0;
            multiplier <= (others => '0');
            done <= '0';
        elsif rising_edge(clk) then
           if we ='1' then
            if counter < p then
              multiplier <= multiplier +1 ;
             result <= a * multiplier;
            counter <= counter + 1;
        if to_integer(unsigned(result)) mod to_integer(unsigned(p))= 1 then
                    inv_out <= multiplier-1;
                   done <= '1';
               end if;
            else
                done <= '1';
            end if;
           end if;
        end if;
    end process;

end Behavioral;
```

Απόσπασμα κώδικα 4.2: Modular Inverse-Fermat (VHDL)

Για παράδειγμα εάν θέλουμε να υπολογίσουμε τον αντίστροφο του α=8 και p = 17 έτσι $8^{-1} \equiv 1(mod\ 17)$. Όπως βλέπουμε και στη (Εικόνα 4.4) από την εξομοίωση το αποτέλεσμα είναι 15.

Το τελικό αποτέλεσμα προκύπτει από τον επαναλαμβάνομε έλεγχο αποτελέσματος με το modulo p (120 mod 17 = 1) μέχρι να είναι 1. Δηλαδή **15** * 8 mod 17 = 120 mod 17 = 1.



Εικόνα 4.4 Κυματομορφή Fermat(εξομοίωση)

4.4 FSM

Το FSM είναι υπολογιστικό μοντέλο που χρησιμοποιείται για το σχεδιασμό συστημάτων με πεπερασμένο αριθμό καταστάσεων. Αποτελείται από ένα σύνολο καταστάσεων, ένα σύνολο μεταβάσεων εισόδους και εξόδους που σχετίζονται με αυτές τις μεταβάσεις.

Καταστάσεις: Αντιπροσωπεύουν τις διαφορετικές συνθήκες ή καταστάσεις στις οποίες μπορεί να βρίσκεται ένα σύστημα ανά πάσα στιγμή. Κάθε κατάσταση συνήθως συνδέεται με συγκεκριμένες συμπεριφορές, ενέργειες ή συνθήκες.

Μεταβάσεις: Ορίζουν τους κανόνες ή τις συνθήκες κάτω από τις οποίες το σύστημα μετακινείται από τη μια κατάσταση στην άλλη. Οι μεταβάσεις ενεργοποιούνται από εξωτερικές εισόδους, συμβάντα ή συνθήκες.

Είσοδοι: Αυτά είναι τα σήματα, τα γεγονότα ή τα ερεθίσματα που ενεργοποιούν τις μεταβάσεις κατάστασης. Οι είσοδοι καθορίζουν τη συμπεριφορά του συστήματος και ποια μετάβαση γίνεται.

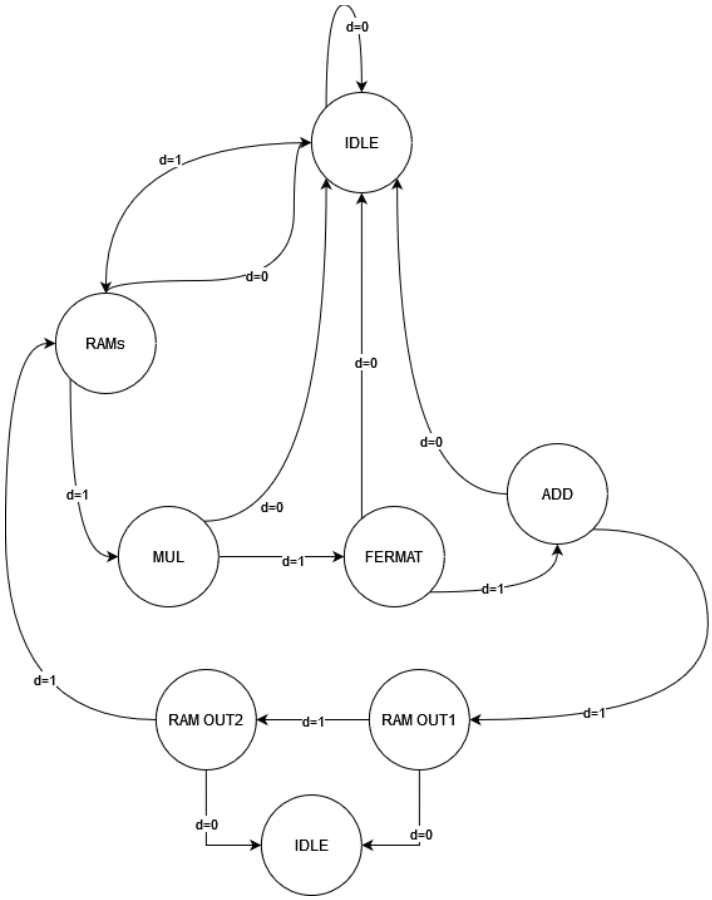
Έξοδοι: Αυτά είναι τα αποτελέσματα ή οι ενέργειες που παράγονται από το FSM ως απόκριση σε εισροές και μεταβάσεις. Οι έξοδοι μπορεί να σχετίζονται με συγκεκριμένες μεταβάσεις, καταστάσεις ή συνδυασμούς τους.

Τα FSM μπορούν να κατηγοριοποιηθούν σε δύο βασικούς τύπους με βάση τη συμπεριφορά εξόδου τους:

Μηχανή Moore: Σε μια μηχανή Moore, η έξοδος εξαρτάται μόνο από την τρέχουσα κατάσταση του συστήματος. Κάθε κατάσταση σχετίζεται με μια συγκεκριμένη τιμή εξόδου.

Μηχανή Mealy: Σε μια μηχανή Mealy, η έξοδος εξαρτάται τόσο από την τρέχουσα κατάσταση όσο και από την είσοδο που ενεργοποιεί τη μετάβαση. Τα αποτελέσματα συνδέονται με μεταβάσεις και όχι μόνο με καταστάσεις.

Πρώτα απ' όλα η καμπύλη Curve 25519 ακολουθεί την παρακάτω λογική του FSM. Η λογική είναι τα bit της συγγενικής συντεταγμένης x του σημείου βάσης αναδιοργανώνονται και χωρίζονται σε 10 ισομερή block bit. Στη συνέχεια, ο πολλαπλασιασμός των σημείων εκτελείται χρησιμοποιώντας μια σκάλα Montgomery επανειλημμένα τόσες φορές, όσες και το μήκος της συντεταγμένης x. Τέλος, λαμβάνονται οι προβολικές συντεταγμένες x' και z'. Μετά από αυτό, το προηγούμενο αποτέλεσμα μετατρέπεται ξανά σε συγγενική συντεταγμένη x αντιστρέφοντας τη συντεταγμένη z (βλπ fermat) και πολλαπλασιάζοντάς την με τη συντεταγμένη x. Τέλος, το τελικό αποτέλεσμα συστέλλεται σε λέξεις 32-bit.



Εικόνα 4.5 Μηχανή πεπερασμένων καταστάσεων (FSM)

Το FSM που δημιουργήθηκε για την καμπύλη μας είναι τύπου Moore. Δηλαδή η τρέχουσες τιμές εξόδου καθορίζονται μονό από την τρέχουσα κατάσταση.

Η μηχανή έχει αρχική κατάσταση (idle). Στην οποία όλα τα σήματα, αλλά και το σύστημα βρίσκετε σε αρχικοποίηση όταν το σήμα reset είναι «1». Αυτό μπορούμε να το δούμε και στον κώδικα (Κώδικας 4.3) στο idle state μέσα στην process της FSM. Όπως παρατηρούμε όταν το σήμα d αλλάξει σε λογικό ¨1¨ τότε μεταπηδάμε στην επόμενη κατάσταση RAMs. Η κατάσταση αυτή ενεργοποιεί τις ram και γενικά έχει σκοπό την φόρτωση και αποθήκευση των δεδομένων. Δεδομένων από την είσοδο, αλλά και τα ενδιάμεσα αποτελέσματα της διαδικασία. Επιπλέων προετοιμάζει το εσωτερικό σήμα ενεργοποίησης για την επόμενη κατάσταση (tmp_MUL<='1' **,** Κώδικας 4.3) Έπειτα μεταβαίνουμε στο Mul-state οπού εκτελείτε διαδοχικά ο Montgomery ladder. Το αποτέλεσμα της διαδικασίας και έπειτα από ένα σήμα που έχει τον ρόλο σημαίας «flag» (nx_state <= FERMAT**)**, περνάει στο state-fermat. Η κατάσταση αυτή σχετίζεται με τον αντίστροφο της συντεταγμένης, εφαρμόζοντας το

θεώρημα Fermat. Εδώ για να μετάβουμε στο επόμενο στάδιο θα πρέπει να ολοκληρωθεί η διαδικασία εύρεσης του αντίστροφου αριθμού. Εξου και η συνθήκη ελέγχου (`IF (d='1' and fer_d='1') THEN` στον (Κώδικα 4.3). Όπου fer_d (Fermat done) το σήμα ολοκλήρωσης της διαδικασίας. Η διαδικασία συνεχίζεται παίρνοντας από τις Rams για την προσωρινή αποθήκευση των αποτελεσμάτων. Οσο το d παραμένει ¨1¨ η μηχανή fsm αλλάζει καταστάσεις. Ενώ με λογικό ¨0¨ επιστρέφουμε στην πρώτη κατάσταση (idle).

```
library IEEE;

use IEEE.STD_LOGIC_1164.ALL;
entity FSM is
  Port (clk, rst, d, fer_d: in std_logic;
      fer_en, mul_en: inout std_logic;
       we_en_ram, we_en_r_out1, we_en_r_out2, ds: inout std_logic
 );
end FSM;

architecture Behavioral of FSM is
    ------- FSM DECLARATION----------
TYPE state IS (ADD, MUL, RAMs, IDLE, RAM_OUT1, RAM_OUT2, FERMAT);-- RAM);
SIGNAL pr_state, nx_state: state; --present...next
SIGNAL tmp_MUL, tmp_we_ram, tmp_ds, tmp_we_ram_out1, tmp_we_ram_out2, tmp_we_fer:
std_logic;
-------- END FSM DECLARATION----------
begin
----- FSM --------- CODE--------
process(rst,clk)
begin
        ---- IDLE STATE---
    if (rst='1') then
       pr_state<=IDLE;
        mul_en<='0';
        fer_en<='0';
        we_en_ram<='0';
        we_en_r_out1<='0';
        we_en_r_out2<='0';
        ds <= '0';
    elsif (clk 'EVENT AND clk='1')then
       pr_state<= nx_state;
        mul_en<=tmp_MUL;
        fer_en<=tmp_we_fer;
        we_en_ram<=tmp_we_ram;
        we_en_r_out1<=tmp_we_ram_out1;
        we_en_r_out2<=tmp_we_ram_out2;
        ds<=tmp_ds;
    end if;
end process;
PROCESS (mul_en, fer_en, we_en_ram, tmp_ds, tmp_MUL, d, fer_d, pr_state, nx_state)
BEGIN
CASE pr_state IS
    WHEN IDLE =>
      tmp_ds <= '0';
       tmp_MUL<='0';
       tmp_we_ram<='0';
       tmp_we_fer<='0';
    IF (d='1') THEN
```

```
            nx_state <= RAMs;
        ELSE nx_state <= IDLE;
        END IF;
        ---- IDLE ADD/SUB---
            WHEN ADD =>
            IF (d='1') THEN
                nx_state <= RAM_OUT1;
           ELSE nx_state <= ADD;
           END IF;
        ---- IDLE MULL/SQUER---
            WHEN MUL =>
            IF (d='1') THEN
                 nx_state <= FERMAT;
            ELSE nx_state <=MUL;
           END IF;

    ---- IDLE RAMs---
    WHEN RAMs =>
             tmp_MUL<='1';
             tmp_ds <= '1';
             tmp_we_ram<='1';
        IF (d='1') THEN
               nx_state <= MUL;
        ELSE nx_state <= IDLE;
        END IF;
        WHEN RAM_OUT1 =>
         tmp_we_ram_out1<='1';
        IF (d='1') THEN
               nx_state <= RAM_OUT2;
        ELSE nx_state <= IDLE;
        END IF;
        WHEN RAM_OUT2 =>
         tmp_we_ram_out2<='1';
        IF (d='1') THEN
               nx_state <= RAMs;
        ELSE nx_state <= IDLE;
        END IF;
    ---- Fermat---
        WHEN FERMAT =>
               tmp_we_fer<='1';
           IF (d='1' and fer_d='1') THEN
               nx_state <= ADD;
            ELSE nx_state <=FERMAT;
           END IF;
    END CASE;
    END PROCESS;
    end Behavioral;
```

Απόσπασμα κώδικα 4.3: FSM (VHDL)

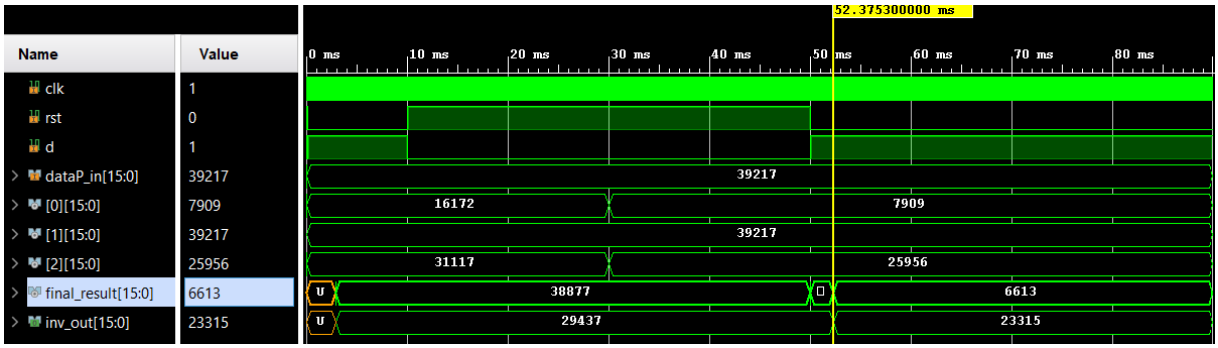
Κεφάλαιο 5

5.1 Εξομοίωση

Η καμπύλη Curve25519 στοχεύει στην παροχή μιας αποτελεσματικής και ασφαλούς λειτουργίας πολλαπλασιασμού σημείων για τη δημιουργία ενός κοινόχρηστου μυστικού κλειδιού με χρήση ενός πρωτοκόλλου Diffie-Hellman.

Στις παρακάτω γραφικές απεικονίσεις βλέπουμε τα αποτελέσματα από τις εξομοιώσεις για την υλοποίησης της καμπύλης ECC25519. Πιο συγκεκριμένα το πρώτο βήμα είναι να γίνει η αρχικοποίηση με το reset στο λογικό «1», ώστε να καθαρίσουνε όλα τα εσωτερικά σήματα. Έπειτα το σήμα d αποκτά την τιμή λογικό «1» ώστε να εκκίνηση το FSM και συνάμα όλοι η λειτουργεία του CORE.

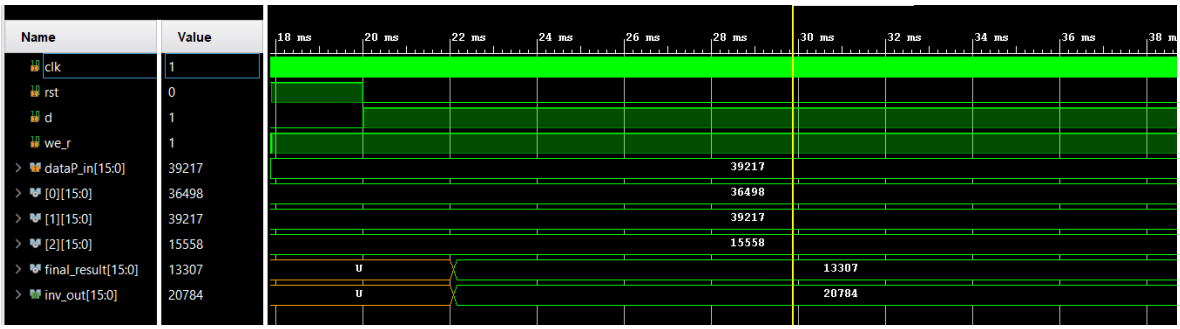
Στην εξομοίωση για σταθερή τιμή modulo p = 39217 (πρώτος αριθμός) και για δύο τυχαίους αριθμούς 9502 και 10014 με σημείο συντεταγμένης χ (0X0900000000000000000000000000000000000000000000000000000000000000) της καμπύλης έχουμε τα εξής αποτελέσματα. Οι δείκτες [0] και [2] έχουν τιμές που προκύπτουν από το ζεύγος του αλγόριθμου Montgomery. Έπειτα προκύπτει η εύρεση του αντίστροφου αριθμού (δείκτη [0]) από τον αλγόριθμο του little Fermat (inv_out). Το τελικό αποτέλεσμα (final_result) προκύπτει από την πράξη a * b mod p. Οπου είναι ο πολλαπλασιασμός του αντιστρόφου με μια τιμή από το ζεύγος του Montgomery (π.χ 23315 * 25956 mod 39217 = 6613)



Εικόνα 5.1 Κυματομορφή καμπύλης (εξομοίωση)

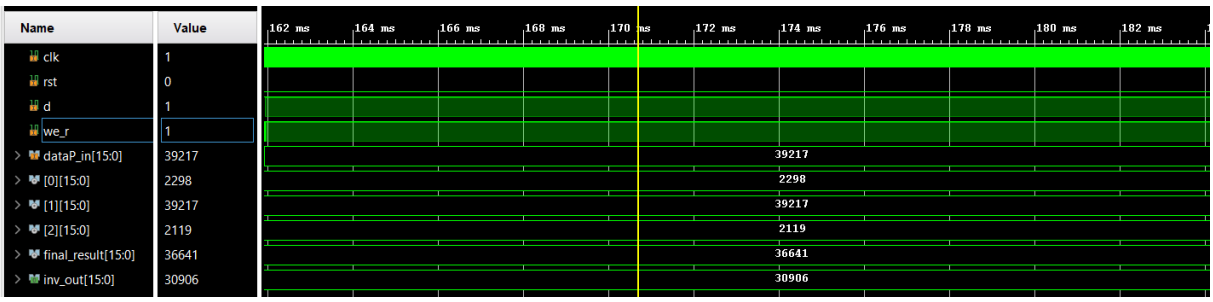
Για παράδειγμα (Εικόνα 5.2) για τυχαίο αριθμό 18206 με σταθερό modulo p = 39217, έπειτα από την εκτέλεση του Montgomery προκύπτουν τα δυο ζευγάρια δείκτη [0]= 36498 και δείκτη [2] = 15558. Το inv_out προκύπτει ο αντίστροφος όπως εξηγήθηκε στο προηγούμενο κεφάλαιο (βλ. Απόσπασμα κώδικα 4.1). Τα εσωτερικά αποτελέσματα οδηγούν το σήμα inv_out στην τελική

πράξη, η οποία είναι a * inv_out mod p. Οπού a ο αριθμός ο δείκτης [2] που προέρχεται από τον Montgomery. Το τελικό αποτέλεσμα είναι 15558 * 20784 mod 39217 = 13307.

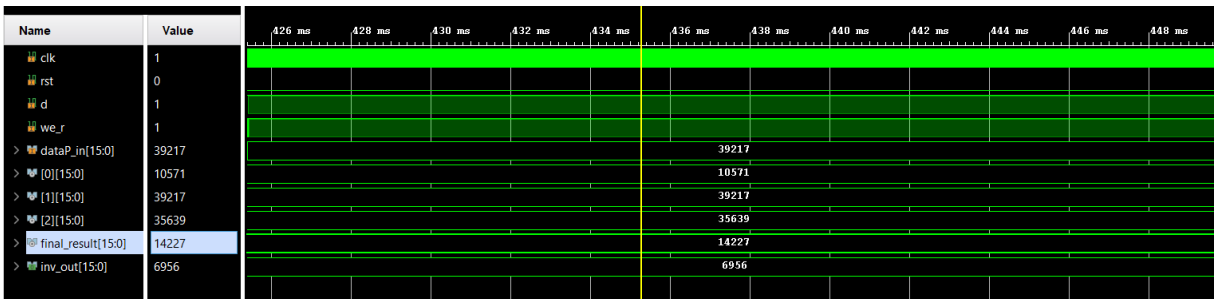


Εικόνα 5.2 Κυματομορφή καμπύλης (εξομοίωση)

Μετά από κάθε τελικό αποτέλεσμα το σήμα reset γίνεται λογικό «1» ώστε να αρχικοποιηθούν όλα τα σήματα μαζί με την είσοδο και την έξοδο. Ακολουθούν επιπλέον παραδείγματα με διαφορετικές τυχαίες τιμές 39454 και 57343, για της εξομοιώσεις (Εικόνα 5.3 & Εικόνα 5.4) αντίστοιχα.



Εικόνα 5.3 Κυματομορφή καμπύλης (εξομοίωση)



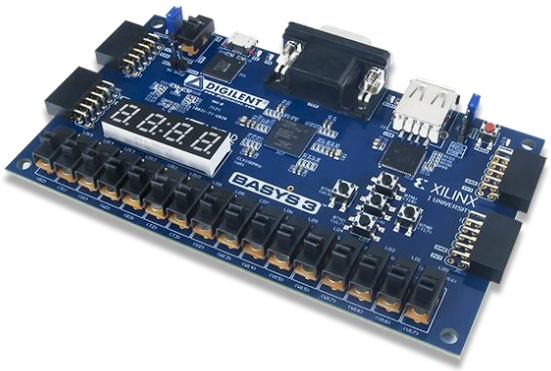
Εικόνα 5.4 Κυματομορφή καμπύλης (εξομοίωση)

Ο Πίνακας 5.1 δείχνει το ποσοστό των καταχωρητών slice, των slice LUT και των DSP48E1 slices που λαμβάνονται από κάθε μονάδα.

Module	Slice Luts	Slice FF	DSP48E1	IO
Available				
Modmul	123(1%)	69(1%)	10(3%)	69(14%)
Ram	209(1%)		10(3%)	105(21%)
Add	41 (1%)			66(13%)
Fsm	5 (1%)	13(1%)		10(2%)
Controller				4(1%)
ModularInverse (ferrmat)	606(1%)	50(1%)	1(1%)	52(10%)
RAM_OUT1	1388(1%)			41(8%)
RAM_OUT2	1392(1%)			42(8%)

Πίνακας 5.1 Κάλυψη περιοχής των εσωτερικών μονάδων

Η υλοποίηση του αλγορίθμου των ελλειπτικών καμπύλης πραγματοποιήθηκε σε FPGA της οικογένειας ATRIX-7. Πιο συγκεκριμένα το Basys 3 το οποίο έχει ως χαρακτηριστικά:



Εικόνα 5.5 Basys 3 FPGA

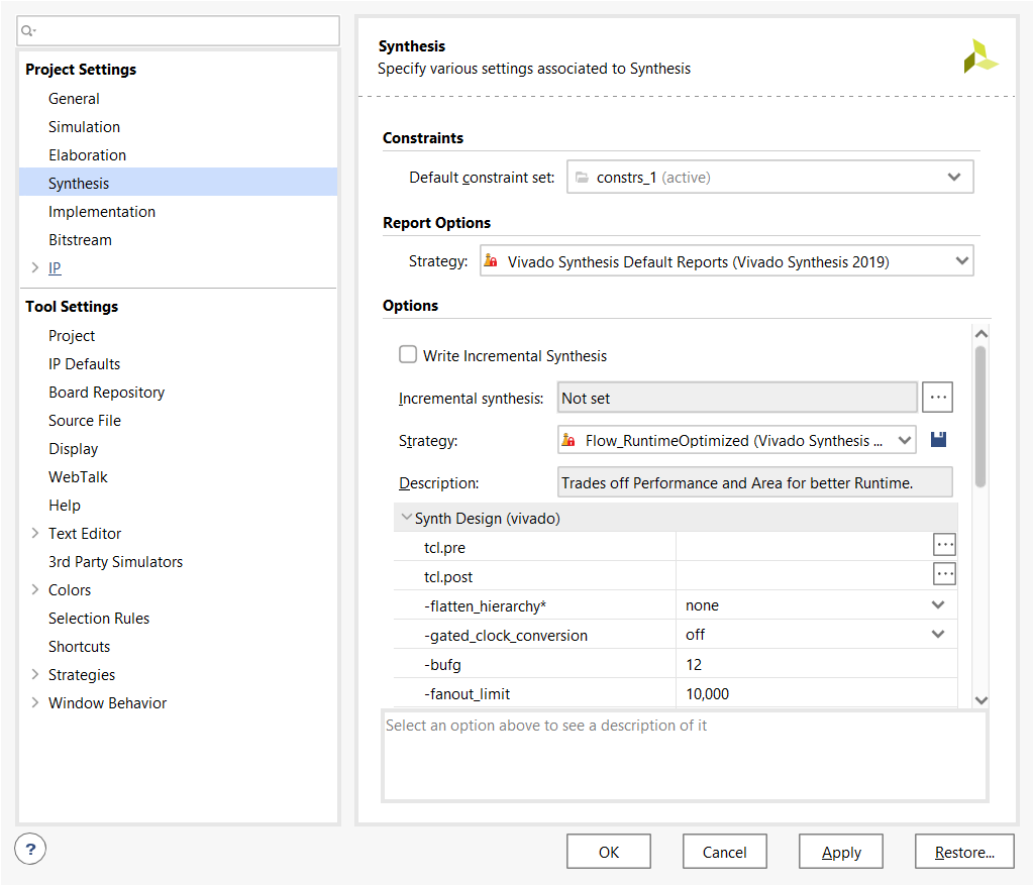
- 33.280 logic cells
- 5200 slices
- 1.800 Kbits block RAM
- Πέντε πλακίδια διαχείρισης ρολογιού
- 90 DSP slices
- Εσωτερικές ταχύτητες ρολογιού που υπερβαίνουν τα 450MHz
- Αναλογικός-ψηφιακός μετατροπέας εντός-του-τσιπ (XADC)

Το Basys 3 προσφέρει επίσης μια βελτιωμένη συλλογή θυρών (Ports) και περιφερειακών, όπως:

- 16 διακόπτες χρήστη
- 16 LEDs
- 5 pushbuttons
- Επιτραπέζια οθόνη 4 ψηφίων με 7 τμήματα

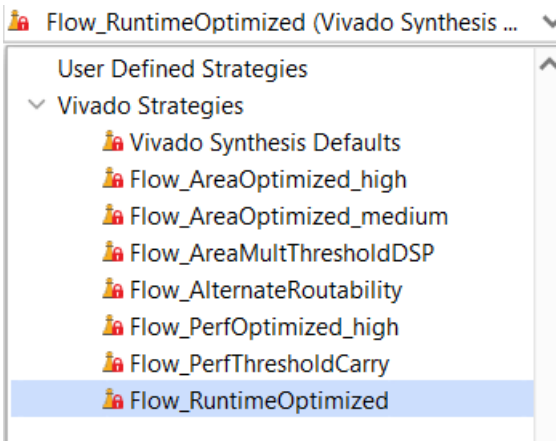
- Θύρα Pmod για σήματα XADC
- Έξοδος VGA 12-bit
- USB-UART
- Serial Flash
- Θύρα Digilent USB-JTAG για προγραμματισμό και σύνδεση με FPGA
- Κεντρική μονάδα USB HID για ποντίκια, πληκτρολόγια και μνήμες stick

Ότι αφορά την εξομοίωση η οποία πραγματοποιήθηκε σε εφαρμογή το VIVADO design software. Οι ρυθμίσεις για την σύνθεση (synthesis) που επιλέξαμε ήταν να μειώσουμε τον χρόνο εκτέλεσης της διαδικασίας. Έτσι στην επιλογή strategy επιλέξαμε το **RuntimeOptimized** (Εικόνα 5.6), το οποίο εξαλείφει κάποιες βελτιστοποιήσεις σε RTL επίπεδο για τη μείωση του χρόνου εκτέλεσης σύνθεσης.



Εικόνα 5.6 Παράθυρο ρυθμίσεων

Στις ρυθμίσεις υπάρχουν κα άλλες επιλογές (Εικόνα 5.7) για την διαδικασία της σύνθεσης. Ενδεικτικά κάποιες από αυτές τις επιλογές **AreaOptimized_high**, οπού εκτελεί βελτιστοποιήσεις για την περιοχή (area) .



Εικόνα 5.7 Παράθυρο επιλογών σύνθεσης

Όλες οι παράμετροι για κάθε επιλογή φαίνεται στον παρακάτω Πίνακα 5.2

Options\Strategies	Default	Flow_Area Optimized_high	Flow_ AreaOptimized_ medium	Flow_Area Multi ThresholdDSP	Flow_ AltRoutability	Performance Optimized	Flow_Perf ThreshholdCarry	Flow_Runtime Optimized
-flatten_hierarchy	rebuilt	rebuilt	rebuilt	rebuilt	rebuilt	rebuilt	rebuilt	none
-gated_clock_conversion	off	off	off	off	off	off	off	off
-bufg	12	12	12	12	12	12	12	12
-fanout_limit	10,000	10,000	10,000	10,000	10,000	400	10,000	10,000
-directive	Default	AreaOptimized _high	AreaOptimized _medium	AreaMuti ThresholdDSP	Alternate Routability	Performanc eOptimized	FewerCarry Chains	RunTime Optimized
-retiming	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked
-fsm_extraction	auto	auto	auto	auto	auto	one_hot	auto	off
-keep_equivalent_registers	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked
-resource_sharing	auto	auto	auto	auto	auto	off	off	auto
-control_set_opt_threshold	auto	1	1	auto	auto	auto	auto	auto
-no_lc	unchecked	unchecked	unchecked	unchecked	checked	checked	checked	unchecked
-no_srlextract	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked
-shreg_min_size	3	3	3	3	10	5	3	3
-max_bram	-1	-1	-1	-1	-1	-1	-1	-1
-max_uram	-1	-1	-1	-1	-1	-1	-1	-1
-max_dsp	-1	-1	-1	-1	-1	-1	-1	-1
-max_b_cascade_height	-1	-1	-1	-1	-1	-1	-1	-1
-max_u_cascade_height	-1	-1	-1	-1	-1	-1	-1	-1
-cascade_dsp	auto	auto	auto	auto	auto	auto	auto	auto
-assert	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked	unchecked

Πίνακας 5.2 Επιλογές σύνθεσης Vivado

5.2 Συμπεράσματα

Συνοψίζοντας όλα όσα υποθηκών προηγουμένας διαπιστώνουμε ότι, χιαζόμαστε στα σύγχρονα συστήματα να εφαρμόζουν υψηλής ταχύτητας αλγορίθμους ασύμμετρης κρυπτογράφησης. Με σκοπό την άμεση και ασφαλής επικοινωνία πομπού δέκτη. Ένα παράδειγμα ανάγκης υψηλής ταχύτητα παρατηρούμε στην Car2Car και Car2Environment επικοινωνία κάτι που τα μηνύματα προστατεύονται με χρήση ασύμμετρης κρυπτογραφίας. Με αφορμή αυτά οι ελπιστικές καμπύλές αποτελούν ένα σημαντικό κρυπτογραφικό σύστημα τόσο για την ταχύτητα οσο και για το μέγεθος κλειδιού και τα υψηλά επίπεδα ασφαλείας όπως είδαμε στο κεφάλαιο 2 (Εικόνα 2.1).

Αναφορές -Βιβλιογραφία

[1] Klein, A. (2013). *Stream Ciphers*. Springer Publishing Company, Incorporated.

[2] National Institute of Standards and Technology. (2001). "Advanced encryption standard," NIST FIPS PUB 197.

[3] Coppersmith, D. (1994). "The data encryption standard (DES) and its strength against attacks." *IBM J. Res. Dev.*, 38(3), 243–250.

[4] Koblitz, N. (1987). "Elliptic curve cryptosystems." *Mathematics of Computation*, 48(177), 203-209.

[5] Mahto, D., & Yadav, D. K. (Year not provided). "Performance Analysis of RSA and Elliptic Curve Cryptography."

[6] Silverman, J. (2009). Chapter 3 of the book *The Arithmetic of Elliptic Curves*.

[7] Lauter, K. E., & Stange, K. E. (2009). "The elliptic curve discrete logarithm problem and equivalent hard problems for elliptic divisibility sequences." In R. M. Avanzi, L. Keliher, & F. Sica (Eds.), *Selected Areas in Cryptography* (pp. 309–327). Berlin, Heidelberg: Springer Berlin Heidelberg.

[8] Diffie, W., & Hellman, M. E. (1976). "New directions in cryptography." *IEEE Transactions on Information Theory*, 22(6), 644–654.

[9] Bernstein, D. J. (2006). "Curve25519: New Diffie-Hellman speed records." In M. Yung, Y. Dodis, A. Kiayias, & T. Malkin (Eds.), *Public Key Cryptography - PKC 2006* (pp. 207–228). Berlin, Heidelberg: Springer Berlin Heidelberg.

[10] Rose, J., El Gamal, A., & Sangiovanni-Vincentelli, A. (1993). "Architecture of field-programmable gate arrays." *Proceedings of the IEEE*, 81(7), 1013–1029.

[11] Brown, S. (1996). "FPGA architectural research: a survey." *IEEE Design & Test of Computers*, 13(4), 9–15.

[12] Xilinx. (2020). *Virtex-7 family overview*.

[13] Chow, P., Seo, S. O., Rose, J., Chung, K., Paez-Monzon, G., & Rahardja, I. (1999). "The design of an SRAM-based field-programmable gate array. i. architecture." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 7(2), 191–197.

[14] Fan, H., Liu, J., Wu, Y.-L., & Cheung, C.-C. (2003). "On optimal hyperuniversal and rearrangeable switch box designs." *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 22(12), 1637–1649.

[15] Fan, H., Liu, J., Wu, Y.-L., & Cheung, C.-C. (2007). "The exact channel density and compound design for generic universal switch blocks." *ACM Trans. Des. Autom. Electron. Syst.*, 12(2), 19–es.+

[16] Lai, Y.-T., & Wang, P.-T. (1997). "Hierarchical interconnection structures for field programmable gate arrays." *IEEE Transactions on Very Large Scale Integration (VLSI) Systems*, 5(2), 186–196.

[17] Savaş, E., & Koç, Ç. K. (2018). "Montgomery inversion." *J Cryptogr Eng*, 8(3), 201–210.

[18] Xilinx. (2022). *7 series fpgas configuration user guide*

[19] Xilinx. (2016). *7 series fpgas configurable logic block*.

[20] Koç, Ç. K., Acar, T., & Kaliski Jr., B. S. (1996). "Analyzing and comparing Montgomery multiplication algorithms." *IEEE Micro*, 16(3), 26–33.

[21] Bernstein, D. J., & Lange. (Year not provided). "Montgomery Curves and the Montgomery Ladder." In J. W. Bos & A. K. Lenstra (Eds.), *Topics in Computation number theory inspired by Peter L. Montgomery* (Chapter 4).

[22] Montgomery, P. L. (1987). "Speeding the Pollard and elliptic curve methods of factorization." *Mathematics of Computation*, 48, 243–264

[23] David Hill (2024). Curve25519 Shared-Key Generation GUI

[24] Bernstein, D. J. (2006). "Curve25519: New Diffie-Hellman speed records." In M. Yung, Y. Dodis, A. Kiayias, & T. Malkin (Eds.), *Public Key Cryptography - PKC 2006* (pp. 207–228). Berlin, Heidelberg: Springer Berlin Heidelberg.

Παράρτημα Α : Διευθύνεις Internet

- Ιστοσελίδα για γραφική απεικόνιση τις πρόσθεσης σημείων της ελλειπτικής καμπύλης. https://www.desmos.com/calculator/caabrmnwxq

- Ιστοσελίδα για γραφική απεικόνιση των σημείων της ελλειπτικής καμπύλης στο πεδίο $\mathbb{F}$ https://graui.de/code/elliptic2/