ΠΑΝΕΠΙΣΤΗΜΙΟ
ΘΕΣΣΑΛΙΑΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΒΙΒΛΙΟΘΗΚΗ ΛΟΓΙΣΜΙΚΟΥ ΓΙΑ ΤΗΝ ΕΠΙΛΥΣΗ ΠΡΟΒΛΗΜΑΤΩΝ ΣΥΝΔΥΑΣΤΙΚΗΣ ΛΟΓΙΚΗΣ

ΧΡΗΣΤΟΣ ΜΕΛΙΣΣΑΡΗΣ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΥΠΕΥΘΥΝΟΣ

ΑΝΤΩΝΙΟΣ ΔΑΔΑΛΙΑΡΗΣ
ΕΠΙΚΟΥΡΟΣ ΚΑΘΗΓΗΤΗΣ

Λαμία 10 Οκτωβρίου έτος 2021

«Με ατομική μου ευθύνη και γνωρίζοντας τις κυρώσεις [(1)], που προβλέπονται από της διατάξεις της παρ. 6 του άρθρου 22 του Ν. 1599/1986, δηλώνω ότι:

*1. Δεν παραθέτω κομμάτια βιβλίων ή άρθρων ή εργασιών άλλων αυτολεξεί **χωρίς να τα περικλείω σε εισαγωγικά** και χωρίς να αναφέρω το συγγραφέα, τη χρονολογία, τη σελίδα. Η αυτολεξεί παράθεση χωρίς εισαγωγικά χωρίς αναφορά στην πηγή, είναι λογοκλοπή. Πέραν της αυτολεξεί παράθεσης, λογοκλοπή θεωρείται και η παράφραση εδαφίων από έργα άλλων, συμπεριλαμβανομένων και έργων συμφοιτητών μου, καθώς και η παράθεση στοιχείων που άλλοι συνέλεξαν ή επεξεργάσθηκαν, χωρίς αναφορά στην πηγή. Αναφέρω πάντοτε με πληρότητα την πηγή κάτω από τον πίνακα ή σχέδιο, όπως στα παραθέματα.*

*2. Δέχομαι ότι η αυτολεξεί **παράθεση χωρίς εισαγωγικά**, ακόμα κι αν συνοδεύεται από αναφορά στην πηγή σε κάποιο άλλο σημείο του κειμένου ή στο τέλος του, είναι αντιγραφή. Η αναφορά στην πηγή στο τέλος π.χ. μιας παραγράφου ή μιας σελίδας, δεν δικαιολογεί συρραφή εδαφίων έργου άλλου συγγραφέα, έστω και παραφρασμένων, και παρουσίασή τους ως δική μου εργασία.*

3. Δέχομαι ότι υπάρχει επίσης περιορισμός στο μέγεθος και στη συχνότητα των παραθεμάτων που μπορώ να εντάξω στην εργασία μου εντός εισαγωγικών. Κάθε μεγάλο παράθεμα (π.χ. σε πίνακα ή πλαίσιο, κλπ), προϋποθέτει ειδικές ρυθμίσεις, και όταν δημοσιεύεται προϋποθέτει την άδεια του συγγραφέα ή του εκδότη. Το ίδιο και οι πίνακες και τα σχέδια

4. Δέχομαι όλες τις συνέπειες σε περίπτωση λογοκλοπής ή αντιγραφής.

Ημερομηνία: 28/9/2021

Ο – Η Δηλ.

ΧΡΗΣΤΟΣ ΜΕΛΙΣΣΑΡΗΣ

(1) «Όποιος εν γνώσει του δηλώνει ψευδή γεγονότα ή αρνείται ή αποκρύπτει τα αληθινά με έγγραφη υπεύθυνη δήλωση
του άρθρου 8 παρ. 4 Ν. 1599/1986 τιμωρείται με φυλάκιση τουλάχιστον τριών μηνών. Εάν ο υπαίτιος αυτών των πράξεων
σκόπευε να προσπορίσει στον εαυτόν του ή σε άλλον περιουσιακό όφελος βλάπτοντας τρίτον ή σκόπευε να βλάψει άλλον, τιμωρείται με κάθειρξη μέχρι 10 ετών.»

Ευχαριστίες

Θα ήθελα να ευχαριστήσω θερμά τον επιβλέποντα καθηγητή μου κ. Αντώνιο Δαδαλιάρη κυρίως για την εμπιστοσύνη που μου έδειξε , και την υπομονή που έκανε κατά την διάρκεια υλοποίησης της πτυχιακής μου εργασίας, αλλά και για την πολύτιμη βοήθεια που μου έδωσε και καθοδήγηση για την επίλυση των προβλημάτων που συνάντησα.

Επίσης οφείλω ένα ευχαριστώ σε όλους όσους συνέβαλαν στην ολοκλήρωση της εργασίας μου είτε ψυχικά είτε πρακτικά.

Τέλος ένα μεγάλο ευχαριστώ στους γονείς μου που με στήριξαν οικονομικά και ψυχικά καθώς και τους φίλους μου και συγγενείς για την ηθική υποστήριξη σε όλο το διάστημα των σπουδών μου.

ΠΕΡΙΛΗΨΗ

Στην παρούσα πτυχιακή εργασία ασχοληθήκαμε με την ανάπτυξη μίας βιβλιοθήκης λογισμικού για την επίλυση προβλημάτων συνδυαστικής λογικής, με χρήση της γλώσσας Python. Αρχικά, γίνεται μία γενική περιγραφή του αντικειμένου της Λογικής Σχεδίασης και αναπτύσσονται οι βασικές έννοιες που την αποτελούν. Κατόπιν, αναλύονται τα πλεονεκτήματα και μειονεκτήματα της γλώσσας Python ενώ υπάρχει και περιληπτική αναφορά στα βασικότερα στοιχεία της (βασικές βιβλιοθήκες, συναρτήσεις κ.λ.π.). Η σύνδεση του αντικειμένου της Λογικής Σχεδίασης με την Python αποτελεί το επόμενο κομμάτι στο οποίο εστιάζουμε, παρουσιάζοντας βιβλιοθήκες που ασχολούνται με επίλυση προβλημάτων συνδυαστικής λογικής. Στη συνέχεια, παραθέτουμε αναλυτικά το περιεχόμενο της βιβλιοθήκης και περιγράφουμε την λειτουργία της εκάστοτε συνάρτησης. Για την κάθε συνάρτηση αποτυπώνονται διαγραμματικά συγκρίσεις του μέσου χρόνου εκτέλεσης, ενώ λαμβάνονται υπόψιν και περιπτώσεις αυξανόμενης πολυπλοκότητας εισόδου. Καταλήγοντας, τονίζεται η χρηστικότητα της βιβλιοθήκης σε περιπτώσεις όπου απαιτείται η επίλυση προβλημάτων συνδυαστικής λογικής καθώς και η ευκολία χρήσης της από τον εκάστοτε ενδιαφερόμενο.

ABSTRACT

In this dissertation we developed a software library for solving combinational logic problems, using the Python language. Firstly, a general description of the object of Logical Design is presented and the basic concepts that constitute it are developed. The advantages and disadvantages of the Python language are then analyzed while there is a brief reference to the language's basic elements (libraries, functions, etc.). The connection of the Logic Design object with Python is the next part we focus on, listing already existing libraries that deal with combinational logic problems. Following, we quote in detail the content (functions) of the library and describe the way each function works. For each function, comparisons of the average execution time for one hundred tests are plotted, while cases of increasing input complexity are also taken into account. In conclusion, the usefulness of the library is emphasized in cases where the solution of combinational logic problems is required as well as the ease of use by the individual concerned.

Περιεχόμενα

ΠΕΡΙΛΗΨΗ 1
ABSTRACT 2

ΚΕΦΑΛΑΙΟ 1 ΕΙΣΑΓΩΓΗ 5

1.1 ΣΤΟΧΟΣ 5

ΚΕΦΑΛΑΙΟ 2 ΒΙΒΛΙΟΓΡΑΦΙΚΗ ΕΠΙΣΚΟΠΗΣΗ 7

ΚΕΦΑΛΑΙΟ 3 ΛΟΓΙΚΗ / ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ 9

3.1 ΛΟΓΙΚΗ ΣΧΕΔΙΑΣΗ 9
3.1.1 ΓΕΝΙΚΗ ΠΕΡΙΓΡΑΦΗ 9
3.1.2 ΆΛΓΕΒΡΑ BOOLE 10
3.1.3 ΛΟΓΙΚΕΣ ΠΥΛΕΣ 10
3.1.4 ΠΙΝΑΚΑΣ ΑΛΗΘΕΙΑΣ 12
3.1.5 ΧΑΡΤΗΣ KARNAUGH 13
3.1.6 QUINE - MCCLUSKEY 14
3.1.7 ΑΡΙΘΜΗΤΙΚΑ ΣΥΣΤΗΜΑΤΑ & ΚΩΔΙΚΕΣ 16
3.1.8 ΔΙΑΓΡΑΜΜΑΤΑ BDD'S 17
3.1.10 ΣΥΝΔΥΑΣΤΙΚΑ ΚΥΚΛΩΜΑΤΑ 18
3.1.9 ΑΚΟΛΟΥΘΙΑΚΑ ΚΥΚΛΩΜΑΤΑ 18

ΚΕΦΑΛΑΙΟ 4 PYTHON 19

4.1 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ 19
4.2 ΧΡΗΣΗ & ΕΦΑΡΜΟΓΕΣ ΤΗΣ PYTHON 19
4.3 ΠΛΕΟΝΕΚΤΗΜΑΤΑ & ΜΕΙΟΝΕΚΤΗΜΑΤΑ ΤΗΣ PYTHON 20
4.3.Α ΠΛΕΟΝΕΚΤΗΜΑΤΑ 20
4.3.Β ΜΕΙΟΝΕΚΤΗΜΑΤΑ 21
4.4 ΒΙΒΛΙΟΘΗΚΕΣ ΣΤΗΝ PYTHON 21
4.5 ΕΓΚΑΤΑΣΤΑΣΗ ΤΗΣ PYTHON 23
4.6 ΒΑΣΙΚΑ ΣΤΟΙΧΕΙΑ ΤΗΣ PYTHON 23
4.6.1 ΑΝΑΓΝΩΡΙΣΤΙΚΑ 23
4.6.2 ΛΕΞΕΙΣ – ΚΛΕΙΔΙΑ 23
4.6.3 ΕΣΟΧΗ 24
4.6.4 ΕΝΤΟΛΗ HELP 24
4.6.5 ΔΗΛΩΣΗ ΠΟΛΛΩΝ ΓΡΑΜΜΩΝ 24
4.6.6 ΣΧΟΛΙΑ 24
4.6.7 ΕΝΤΟΛΗ PRINT 24
4.6.8 ΕΝΤΟΛΗ INPUT 24
4.6.9 ΠΟΛΛΑΠΛΕΣ ΔΗΛΩΣΕΙΣ ΣΕ ΜΙΑ ΓΡΑΜΜΗ 25
4.6.10 ΠΟΛΛΑΠΛΕΣ ΟΜΑΔΕΣ ΔΗΛΩΣΕΩΝ 25
4.6.11 ΜΕΤΑΒΛΗΤΕΣ 25

4.6.12 Μετατροπη τυπου δεδομενων 26
4.6.13 Σταθερες 26
4.6.14 Τελεστες 26
4.7 Δομες Ελεγχου & Επαναληψης 27
4.7.1 Δομη Ελεγχου if 27
4.7.2 Δομες Επαναληψης 27
4.7.3 Δηλωση with & break 28
4.7.4 Συναρτησεις 28
4.8 Python & Λογικη Σχεδιαση 28

ΚΕΦΑΛΑΙΟ 5 ΠΕΡΙΕΧΟΜΕΝΑ & ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΒΙΒΛΙΟΘΗΚΗΣ 30

ΚΕΦΑΛΑΙΟ 6 ΑΠΟΤΕΛΕΣΜΑΤΑ & ΜΕΛΛΟΝΤΙΚΕΣ ΠΡΟΕΚΤΑΣΕΙΣ 35

6.1 Αποτελεσματα 35
6.2 Συμπερασματα & Μελλοντικες Προεκτασεις 42

ΒΙΒΛΙΟΓΡΑΦΙΑ 43

ΚΕΦΑΛΑΙΟ 1 Εισαγωγή

1.1 Στόχος

Η παρούσα διπλωματική εργασία, είχε στόχο την ανάπτυξη βιβλιοθήκης σε γλώσσα Python για την γρήγορη και εύκολη επίλυση προβλημάτων συνδυαστικής λογικής.

Στην πληροφορική, η συνδυαστική λογική είναι ένα απλοποιημένο μοντέλο υπολογισμού, το οποίο ανεξάρτητα από την απλή δομή του, περιλαμβάνει πληθώρα υπολογιστικών χαρακτηριστικών. Τα προβλήματα συνδυαστικής λογικής που επιλύθηκαν στην συγκεκριμένη εργασία περιέχονται στο μάθημα της Λογικής / Ψηφιακής Σχεδίασης. Γενικά, αποτελούν πονοκέφαλο για κάποιον που θα χρειαστεί να ασχοληθεί με τέτοια ζήτηματα χωρίς να είναι εξοικειωμένος με το αντικείμενο της Λογικής Σχεδίασης. Η επίλυση προβλημάτων τέτοιου τύπου αποτελεί χρονοβόρα και περίπλοκη διαδικασία, διότι υπάρχει πληθώρα δεδομένων που πρέπει να ληφθούν υπόψιν. Δεδομένου ότι η Python αποτελεί μία εύκολη, κατανοητή, υψηλού επιπέδου και με δυνατότητες επεκτασιμότητας γλώσσα προγραμματισμού, επιλέχθηκε ως η γλώσσα υλοποίησης της συγκεκριμένης βιβλιοθήκης.

Για τον σκοπό αυτό, αναγκαία προέκυψε η μελέτη του μαθήματος της Λογικής / Ψηφιακής Σχεδίασης έτσι ώστε το θεωρητικό του κομμάτι να γίνει πλήρως κατανοητό. Εν συνεχεία, έγινε εκτεταμένη μελέτη της γλώσσας Python για να υπάρχει η δυνατότητα ανταπόκρισης στο επίπεδο προγραμματισμού που απαιτείται για την συγγραφή του ζητούμενου κώδικα. Αντίστοιχα, διερευνήθηκε η ύπαρξη τυχόν έτοιμων βιβλιοθηκών που να σχετίζονται με τέτοια ζητήματα ή γενικότερα με το αντικείμενο της Λογικής Σχεδίασης. Η κατασκευή της συγκεκριμένης βιβλιοθήκης είναι σημαντική διότι, παρέχει την δυνατότητα σε οποιονδήποτε είναι εξοικειωμένος με βασικά στοιχεία προγραμματισμού, να μπορεί με απλό τρόπο να εγκαταστήσει την βιβλιοθήκη και να καλέσει την εκάστοτε συνάρτηση για την επίλυση των επιμέρους αντίστοιχα προβλημάτων που τον απασχολούν.

Πιο συγκεκριμένα, κατασκευάστηκε βιβλιοθήκη με πληθώρα συναρτήσεων, όπως: εμφάνιση και υπολογισμός αναπτύγματος για δυαδικούς, οκταδικούς, δεκαδικούς και δεκαεξαδικούς αριθμούς, την μετατροπή συστημάτων αρίθμησης με βάση το δύο, τον υπολογισμό λογικών πυλών (AND, OR, NOT, XOR, XNOR, NAND), και τον υπολογισμό συμπληρώματος ως προς ένα και ως προς δύο. Επίσης, συμπεριλαμβάνει τους κώδικες BCD, Gray, Biquinary και 2-4-2-1, εύρεση και μετατροπή αθροίσματος ελαχιστόρων και γινόμενο μεγιστόρων, την απλοποίηση λογικών συναρτήσεων με Χάρτες Karnaugh και Quine – McCluskey και τέλος διαγράμματα BDD's.

Με την υλοποίηση της βιβλιοθήκης παρέχεται στον προγραμματιστή η δυνατότητα άμεσης επίλυσης προβλημάτων συνδυαστικής λογικής με το κάλεσμα της κατάλληλης συνάρτησης. Η ύπαρξή της θα καταστήσει πιο εύκολη και χρονικά αποδοτική την οποιαδήποτε συγγραφή κώδικα που σχετίζεται με θέματα Λογικής και Ψηφιακής Σχεδίασης.

Επιπλέον, στόχος της συγκεκριμένης εργασίας είναι τα προγράμματα γραμμένα σε Python να μπορούν να τρέξουν τις συναρτήσεις που υλοποιήθηκαν, γρήγορα και χωρίς να παρουσιάζουν σφάλματα. Για αυτό το λόγο εκτελέστηκαν όλες

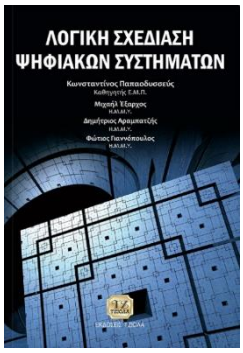
οι συναρτήσεις της βιβλιοθήκης ξεχωριστά, έτσι ώστε να επιβεβαιωθεί η ορθότητα των αποτελεσμάτων τους και η χρονική απόκριση τους. Πραγματοποιήθηκαν πειράματα για κάθε συνάρτηση και όπου αυτό ήταν δυνατό χρησιμοποιήθηκε διαφορετικό μέγεθος εισόδων, ώστε να υπάρχει μια πλήρης εικόνα των χρόνων εκτέλεσής τους.

ΚΕΦΑΛΑΙΟ 2 Βιβλιογραφική Επισκόπηση

Για την υλοποίηση της παρούσας εργασίας κατέστη αναγκαία η αναζήτηση βιβλιογραφίας που αφορά το αντικείμενο της Λογικής Σχεδίασης και της γλώσσας Python.

Για την Λογική Σχεδίαση:

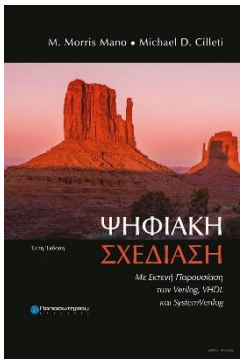
- Λογική σχεδίαση ψηφιακών συστημάτων (1):
  Εκδόσεις Τζιόλα, έτος 2018
  Συγγραφέας : Συλλογικό έργο Παπαοδυσσεύς,
  Κώστας Έξαρχος,
  Μιχαήλ Αραμπατζής,
  Δημήτριος Γιαννόπουλος, Φώτιος



Εικόνα 1. Βιβλίο – ΛΟΓΙΚΗ ΣΧΕΔΙΑΣΗ ΨΗΦΙΑΚΩΝ ΚΥΚΛΩΜΑΤΩΝ.

Περιέχει συστήματα αρίθμησης που βασίζονται στη θέση του ψηφίου, εισαγωγή στην άλγεβρα Boole. Συνδυαστικά κυκλώματα όπως: λογικές πύλες, απλοποίηση συναρτήσεων με Χάρτη Karnaugh, αποκωδικοποιητές, πολυπλέκτες και κωδικοποιητές. Τέλος, ακολουθιακά κυκλώματα όπως: flip – flop, καταχωρητές, συγχρονοι και ασύγχρονοι μετρητές.

- Ψηφιακή Σχεδίαση (2):
  Εκδόσεις Παπασωτηρίου, έτος 2018
  Συγγραφέας : Morris Mano, Michael Ciletti

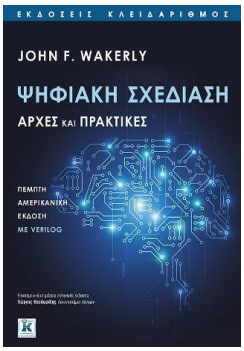


Εικόνα 2. Βιβλίο – ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ.

Περιλαμβάνει όλα τα βασικά βασικά στοιχεία της Λογικής Σχεδίασης, όπως ψηφιακά συστήματα και δυαδικούς αριθμούς, άλγεβρα Boole και λογικές πύλες, ελαχιστοποίηση σε επίπεδο πυλών, σύγχρονη και ασύγχρονη ακολουθιακή λογική, καταχωρητές και μετρητές. Έχουν προστεθεί ακόμα και παραδείγματα τόσο σε Verilog όσο και σε VHDL.

- Ψηφιακή Σχεδίαση : Αρχές και Πρακικές (3):
  Εκδόσεις Κλειδάριθμος, έτος 2019
  Συγγραφέας : John F. Wakerly

Το βιβλίο καλύπτει τα βασικά δομικά υλικά της Ψηφιακής Σχεδίασης σε δίαφορα αφαιρετικά επίπεδα, από τις πύλες μέχρι τις γλώσσες περιγραφής υλικού. Σε κάθε επίπεδο αναλύονται οι λειτουργίες όπως λογικές πύλες, αποκωδικοποιητές, πολυπλέκτες, κυκλώματα flip – flop, καταχωρητές και μετρητές.



Εικόνα 3. Βιβλίο - ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ : ΑΡΧΕΣ ΚΑΙ ΠΡΑΚΤΙΚΕΣ.

Για την Python:

- Υπολογισμοί και προγραμματισμός με την Python (4):
  Εκδόσεις Κλειδάριθμος έτος 2015
  Συγγραφέας : John V. Guttag

Αποτελεί ένα ιδανικό εισαγωγικό εγχειρίδιο για την εκμάθηση επίλυσης προβλημάτων με χρήση της γλώσσας Python και των βιβλιοθηκών της. Το βιβλίο καλύπτει μεγάλο εύρος θεμάτων όπως: η οπτικοποίηση δεδομένων, προσομοιώσεις τυχαιότητας μοντέλων, υπολογιστικές τεχνικές για την κατανόηση δεδομένων και στατιστικές τεχνικές για πληροφόρηση. Εξετάζει επίσης πιο προχωρημένα θέματα όπως: προβλήματα βελτιστοποίησης, δυναμικό προγραμματισμό και μηχανική μάθηση.

Εικόνα 4. Βιβλίο - Υπολογισμοί και προγραμματισμός με την Python.

Εκτός από τα παραπάνω βιβλία έγινε αναζήτηση για έτοιμες βιβλιοθήκες σε Python που σχετίζονται με το αντικείμενο της Λογικής Σχεδίασης. Βρέθηκε αρκετά μεγάλος αριθμός πακέτων που ασχολείται με την επίλυση ζητημάτων συνδυαστικής λογικής. Τα πακέτα τα οποία επιλέχθηκαν και μελετήθηκαν στο Κεφάλαιο 4 είναι τα ακόλουθα:

- LogicPy
- logic-py
- BinPy
- truth-table-generator
- quine-mccluskey

ΚΕΦΑΛΑΙΟ 3 Λογική / Ψηφιακή Σχεδίαση

3.1 Λογική Σχεδίαση

3.1.1 Γενική Περιγραφή

Τα ψηφιακά συστήματα καταλαμβάνουν πλέον βασικό μέρος της καθημερινής ζωής με την τρέχουσα τεχνολογική περίοδο να χαρακτηρίζεται ως ψηφιακή εποχή. Αυτά υπάρχουν παντού, στην επικοινωνία, στις συναλλαγές, στα οχήματα, στην βιομηχανία, στην ιατρική κ.ο.κ. Παίζουν σημαντικό ρόλο στην βελτίωση της καθημερινότητάς, ειδικά αν αναλογιστεί κανείς ότι όλες οι συσκευές που χρησιμοποιούνται σε καθημερινή βάση περιλαμβάνουν ένα ηλεκτρονικό υπολογιστή, ο οποίος εκτελεί οδηγίες και αποτελείται από κάποιο ψηφιακό σύστημα. Ένα τέτοιο σύστημα δέχεται είσοδο από το πληκτρολόγιο, την επεξεργάζεται και παράγει μια έξοδο σε διάφορες μορφές (π.χ. εικόνα, ήχος). Τα ψηφιακά σύστηματα αποτελούται από ψηφιακές υπομονάδες. Για να κατανοηθεί η λειτουργία των υπομονάδων και ολόκληρου του συστήματος θα πρέπει να έχει κανείς βασικές γνώσεις της λογικής λειτουργίας των ψηφιακών κυκλωμάτων, γεγονός που επιτυγχάνεται με τη κατανόηση του αντικειμένου της Λογικής Σχεδίασης.

Η Λογική Σχεδίαση δεν μελετά ένα φυσικό κύκλωμα αλλά ένα λογικό, δηλαδή μία αφηρημένη αντίληψη του φυσικού στην οποία έχουν επισημανθεί τα χαρακτηριστικά που χρειάζονται για την εφαρμογή της Άλγεβρας Boole. Ένα λογικό κύκλωμα αποτελείται από λογικές πύλες και μεταβλητές όπου η έξοδος εξαρτάται από την είσοδο η οποία δέχεται τιμές 0 και 1, την κάθε χρονική στιγμή.

Η Λογική Σχεδίαση λοιπόν επικεντρώνεται στην μελέτη, τη σχεδίαση και την απλοποίηση ακολουθιακών κυκλωμάτων όπως τα flip – flops, καθώς και συνδυαστικών κυκλωμάτων όπως είναι ένας ημιαθροιστής, ένας πλήρης αθροιστής, ένας αποκωδικοποιητής κ.ο.κ. Η ελαχιστοποίηση των συναρτήσεων που περιέχουν μέχρι 4 μεταβλητές επιτυγχάνεται με την χρήση του Χάρτη Karnaugh και για πάνω από 4 με τον αλγόριθμο Quine – McCluskey.

Καλύπτονται ζητήματα μετατροπής συστημάτων αρίθμησης μεταξύ δεκαδικού, δυαδικού, οκταδικού και δεκαεξαδικού συστήματος, αριθμητικές αναπαραστάσεις των παραπάνω συστημάτων όπως και κώδικες BCD, Biquinary, 2-4-2-1 και Gray για την μετατροπή δεκαδικών ψηφίων. Περιέχονται πράξεις με λογικές πύλες αφού τα κυκλώματα αποτελούνται από αυτές καθώς και δημιουργία πινάκων αληθείας. Τέλος, δίνεται η δυνατότητα υλοποίησης διαγραμμάτων BDD's που αποτελούν τον τρόπο απεικόνισης λογικών συναρτήσεων.

Το μάθημα της Λογικής Σχεδίασης αποτελεί τη βάση για την εισαγωγή στο σχεδιασμό και την μελέτη ψηφιακών κυκλωμάτων και συστημάτων. Αποτελεί μάθημα κορμού σε ακαδημαϊκά ιδρύματα και διδάσκεται σε τμήματα Πληροφορικής, Τεχνολογίας και Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών. Το μάθημα είθισται να περιλαμβάνει θεωρητικό και εργαστηριακό σκέλος.

3.1.2 Άλγεβρα Boole

Ο George Boole ήταν μαθηματικός και το 1847 (5) με τα δεδομένα των τότε μαθηματικών εννοιών παρουσίασε μια αλγεβρική δομή την οποία γνωρίζουμε σήμερα σαν άλγεβρα Boole, όπου εκφράζει την Αριστοτέλεια λογική χρησιμοποιώντας μεταβλητές δύο τιμών 0 και 1. Ο όρος άλγεβρα Boole χρησιμοποιήθηκε για πρώτη φορά το 1913 από τον Sheffer και εισήχθη το 1854. Είναι βασική για την σχεδίαση του λογισμικού και των κυκλωμάτων των ηλεκτρονικών υπολογιστών. Έχει θεμελιώδη σημασία για τον κλάδο της Πληροφορικής και είναι σημαντική για την θεωρητική μελέτη του πεδίου της λογικής σχεδίασης. Επίσης αποτελεί εργαλείο και για άλλα πεδία όπως η στατιστική και ο προγραμματισμός.

Η άλγεβρα αυτή περιέχει τρεις κύριες πράξεις, την σύζευξη που συμβολίζεται με ‘∧’, την διάζευξη που συμβολίζεται με ‘∨’ και την άρνηση που συμβολίζεται με ‘¬ ‘, όπου οι τιμές των μεταβλητών είναι τιμές 0 και 1 (αληθής ή ψευδής αντίστοιχα).

Έναν αιώνα αργότερα, το 1938, ο Claude Shannon διαπίστωσε ότι απλά αντιστοιχίζοντας τις ‘αληθής’ και ‘ψευδής’ στο κλειστό και ανοιχτό διακόπτη, μπορεί να εφαρμοστεί η άλγεβρα Boole και σε κυκλώματα διακοπτών. Το θεώρημα του Shannon χρησιμοποιείται στην μετατροή λογικής συνάρτησης σε μορφή αθροίσματος ελαχιστόρων ή γινομένου μεγιστόρων.

Ένα παράδειγμα απλοποίησης με άλγεβρα Boole παρουσιάζεται παρακάτω:

$$F = x'y'z + x'yz + xy' \Rightarrow$$

$$= x'z(y' + y) + xy' \Rightarrow$$

$$= x'z \cdot 1 + xy' \Rightarrow$$

$$= x'z + xy' \Rightarrow$$

$$F = x'z + xy'$$

Η απλοποίηση γίνεται βάση των ιδιοτήτων και αξιομάτων τις άλγεβρας Boole. Απο την γραμμή 1 στην γραμμή 2 χρησιμοποιείται το αξίωμα $x(y + z) = xy + xz$. Από την 2 στην 3 το αξίωμα $x + x' = 1$ και απο την 3 στην 4 για την απλοποίηση εφαρμόζεται το αξίωμα $x \cdot 1 = x$.

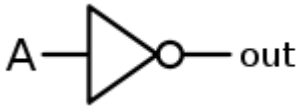
3.1.3 Λογικές Πύλες

Οι λογικές πύλες είναι οι θεμελιώδεις δομικές μονάδες των ψηφιακών κυκλωμάτων (6). Μια λογική πύλη είναι ηλεκτρονικό κύκλωμα που υλοποιεί μία λογική πράξη και παράγει μία έξοδο. Λειτουργούν σε δυαδικό σύστημα, δηλαδή παίρνουν τιμές 0, που σημαίνει ότι δεν υπάρχει ροή πληροφορίας, και 1 όπου υπάρχει ροή πληροφορίας. Όπως φαίνεται και από την ονομασία τους λειτουργούν

σαν διακόπτες ανοίγοντας και κλείνοντας, ώστε να επιτρέπουν ή όχι τη ροή ψηφιακής πληροφορίας.

Στις ακόλουθες εικόνες παρουσιάζονται όλες οι βασικές λογικές πύλες και ο τρόπος λειτουργίας τους:

- Πύλη NOT

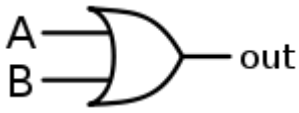


ΕΙΣΟΔΟΣ	ΕΞΟΔΟΣ
A	Not A
0	1
1	0

- Πύλη AND

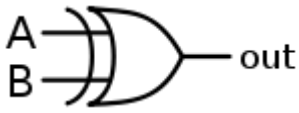
ΕΙΣΟΔΟΣ		ΕΞΟΔΟΣ
A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

- Πύλη OR



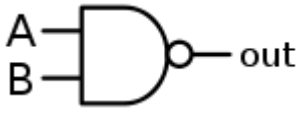
ΕΙΣΟΔΟΣ		ΕΞΟΔΟΣ
A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

- Πύλη XOR



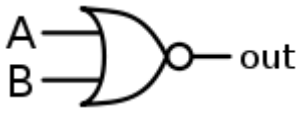
ΕΙΣΟΔΟΣ		ΕΞΟΔΟΣ
A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

- Πύλη NAND



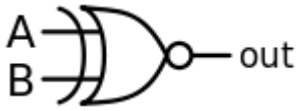
ΕΙΣΟΔΟΣ		ΕΞΟΔΟΣ
A	B	A NAND B
0	0	1
0	1	1
1	0	1
1	1	0

- Πύλη NOR



ΕΙΣΟΔΟΣ		ΕΞΟΔΟΣ
A	B	A NOR B
0	0	1
0	1	0
1	0	0
1	1	0

- Πύλη XNOR



ΕΙΣΟΔΟΣ		ΕΞΟΔΟΣ
A	B	A XNOR B
0	0	1
0	1	0
1	0	0
1	1	1

3.1.4 Πίνακας Αληθείας

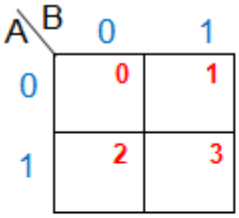
Είναι ένας πίνακας που χρησιμοποιείται σε επιστήμες όπως είναι τα μαθηματικά και η πληροφορική. Λειτουργία του είναι η ανάδειξη αποτελέσματος λογικών πυλών. Υπολογίζει τις λογικές πράξεις AND, NOT, OR, XOR, NOR. Ο πίνακας αποτελείται από στήλες όπου, στην αριστερή πλευρά περιέχονται οι μεταβλητές και στην δεξιά το αποτέλεσμα, ενώ οι γραμμές περιέχουν όλους τους δυνατούς συνδυασμούς των τιμών που έχουν οι μεταβλητές. Οι τιμές που δέχεται ένας πίνακας ειναι False (ψευδής) και True (αληθής). Ένα παράδειγμα ενός πίνακα αληθείας για λογική ισότητα είναι ο παρακάτω:

p	q	p ≡ q
T	T	T
T	F	F
F	T	F
F	F	T

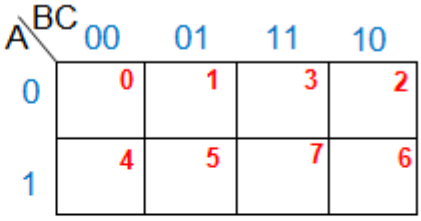
Πίνακας 1. Πίνακας Αληθείας για ισότητα

3.1.5 Χάρτης Karnaugh

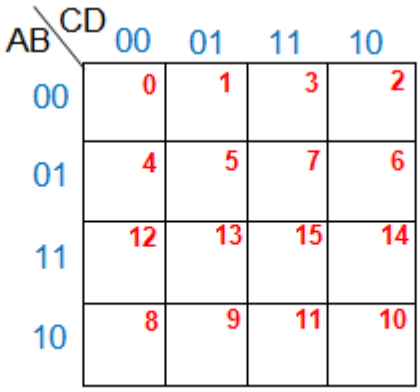
Η απλοποίηση των εκφράσεων Boolean είναι η βάση για την σχεδίαση ψηφιακού κυκλώματος. Οι χάρτες Karnaugh αποτελούν μία από αυτές τις τεχνικές απλοποίησης. Ο Maurice Karnaugh τον εισήγαγε το 1953 (7) ως βελτίωση του διαγράμματος Veitch του Edward W. Veitch ένα χρόνο πριν. Ο Χάρτης Karnaugh είναι εύχρηστος όταν οι μεταβλητές που περιέχονται είναι μικρότερες ή ίσες με τέσσερις, διότι χρησιμοποιεί δισδιάστατους πίνακες για την απλοποίηση. Αυτό έχει σαν αποτέλεσμα με την αύξηση των μεταβλητών να αυξάνεται πολύ το μέγεθος των πινάκων κάνοντας την συγκεκριμένη τεχνική δύσχρηστη. Γενικά αν έχουμε x εισόδους τότε το μέγεθος των κελιών στον χάρτη θα είναι 2^x.



Εικόνα 5. Χάρτης Karnaugh 2 μεταβλητών.



Εικόνα 6. Χάρτης Karnaugh 3 μεταβλητών.



Εικόνα 7. Χάρτης Karnaugh 4 μεταβλητών.

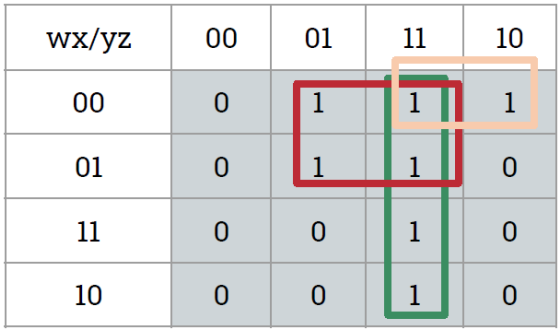
Στις Εικόνες 5, 6, 7 εμφανίζονται οι Χάρτες Karnaugh για 2, 3, 4 μεταβλητές αντίστοιχα και όπως φαίνεται έχουν την ίδια δομή. Σε κάθε Χάρτη Karnaugh οι μεταβλητές εισόδου εμφανίζονται στην γωνία πάνω αριστερά και είναι οι εμπλεκόμενες μεταβλητές στην λογική έκφραση που θα απλοποιηθεί. Οι τιμές τους εμφανίζονται οριζόντια και κάθετα σε δυαδική μορφή. Τα κελιά στον Χάρτη παίρνουν τιμές από την συνάρτηση, η οποία δίνεται σε άθροισμα ελαχιστόρων. Οι τιμές που δέχονται είναι 1 (δηλώνει ότι το κελί αυτό αντιστοιχεί σε ελάχιστο όρο) και 0 (το κελί αντιστοιχεί σε μέγιστο όρο). Αφού γίνει η εισαγωγή των στοιχείων στον Χάρτη ομαδοποιούνται τα κελιά που αναγράφουν τον αριθμό 1 σε ομάδες όπου ο αριθμός μελών είναι δυνάμεις του 2. Τέλος, επιλέγονται οι τιμές των μεταβλητών που είναι σταθερές στα κελιά της κάθε ομάδας και το άθροισμά τους είναι η τελική απλοποιημένη μορφή της λογικής έκφρασης.

Στη συνέχεια παρατίθεται παράδειγμα με χρήση Χάρτη Karnaugh 4 μεεταβλητών

$$F(w,x,y,z) = \Sigma(1,2,3,5,7,11,15)$$

wx/yz	00	01	11	10
00	0	1	1	1
01	0	1	1	0
11	0	0	1	0
10	0	0	1	0

Εικόνα 5. Πίνακας Χάρτη Karnaugh 4 μεταβλητών.



Εικόνα 6. Επιλογή ομάδων σε πίνακα Χάρτη Karnaugh.

Μετά από επιλογή στην εικόνα 6 το αποτέλεσμα της απλοποιημένης συνάρτησης είναι:

$$F = w'z + yz + w'x'y$$

3.1.6 Quine - McCluskey

Ο αλγόριθμος Quine – McCluskey είναι επίσης μία τεχνική για την απλοποίηση εκφράσεων Boolean που ξεκίνησε από τον Willard V. Quine το 1952 (8) και συνεχίστηκε από τον Edward J. McCluskey το 1956 (8). Ο αλγόριθμος λειτουργικά είναι παρόμοιος με την μέθοδο του Χάρτη Karnaugh, αλλά η μορφή πίνακα που έχει του δίνει πλεονέκτημα σε σχέση με τον Karnaugh για χρήση σε αλγορίθμους υπολογιστών. Επίσης, έχει την δυνατότητα να ελέγξει εάν η συνάρτηση έχει φτάσει στην ελάχιστη μορφή.

Για να απλοποιήσουμε μια συνάρτηση με την μέθοδο Quine – McCluskey αρχικά, εκφράζουμε την συνάρτηση ως άθροισμα γινομένων. Μετατρέπουμε τους όρους της συνάρτησης σε δυαδικούς και τους τοποθετούμε κατά αύξουσα σειρά ανάλογα με το πλήθος των 1 που έχουν. Έτσι η πρώτη ομάδα έχει πλήθος 1 ίσο με 1, η δεύτερη ίσο με 2 κ.ο.κ. Συγκρίνουμε τους όρους την πρώτης ομάδας με όλους της επόμενης και εξετάζουμε εάν ο πρώτος όρος έχει διαφορά μόνο ένα ψηφίο από τον δεύτερο. Εφόσον ισχύει αυτό, δημιουργούμε ένα νέο όρο ο οποίος περιέχει τα κοινά ψηφία των δύο προηγούμενων και στο ψηφίο που διαφέρουν έχει μία παύλα. Μόλις ολοκληρώσουμε το προηγούμενο βήμα έχουμε έναν δεύτερο πίνακα στον οποίο κάνουμε την ίδια διαδικασία. Οι συγκρίσεις θα γίνονται έως ότου οι ομάδες που θα υπάρχουν θα έχουν πάνω από 2 διαφορετικά ψηφία. Οι όροι που δεν συγκρίθηκαν ονομάζονται prime implicants. Δημιουργούμε τον πίνακα των prime implicants και των όρων της αρχικής συνάρτησης και τοποθετούμε το x σε κάθε διασταύρωση όπου ο prime implicant περιέχει τον αντίστοιχο όρο της αρχικής

συνάρτησης. Τέλος, προσπαθούμε να επιλέξουμε τους κατάλληλους και όσο το δυνατόν λιγότερους essential prime implicants έτσι ώστε να έχουμε επιλέξει όλους τους όρους της αρχικής συνάρτησης. Στο επόμμενο παράδειγμα φαίνεται αναλυτικά η διαδικασία.

$$F(w,x,y,z) = \Sigma(m0,m1,m2,m5,m7,m8,m9,m10,m13,m15)$$

	w	x	y	z
0	0	0	0	0 √
1	0	0	0	1 √
2	0	0	1	0 √
8	1	0	0	0 √
5	0	1	0	1 √
9	1	0	0	1 √
10	1	0	1	0 √
7	0	1	1	1 √
13	1	1	0	1 √
15	1	1	1	1 √

Εικόνα 7. Quine - McCluskey Πίνακας 1.

	w	x	y	z
0, 1	0	0	0	- √
0, 2	0	0	-	0 √
0, 8	-	0	0	0 √
1, 5	0	-	0	1 √
1, 9	-	0	0	1 √
2, 10	-	0	1	0 √
8, 9	1	0	0	- √
8, 10	1	0	-	0 √
5, 7	0	1	-	1 √
5, 13	-	1	0	1 √
9, 13	1	-	0	1 √
7, 15	-	1	1	1 √
13, 15	1	1	-	1 √

Εικόνα 8. Quine- McCluskey 1η ομαδοποίηση.

	w	x	y	z	
0, 1, 8, 9	-	0	0	-	x'y'
0, 2, 8, 10	-	0	-	0	x'z'
1, 5, 9, 13	-	-	0	1	y'z
5, 7, 13, 15	-	1	-	1	xz

Εικόνα 9. Quine- McCluskey 2η ομαδοποίηση.

	0	1	2	5	7	8	9	10	13	15
x'y'	X	X				X	X			
x'z'	X		X			X		X		
y'z		X		X			X		X	
xz				X	X				X	X
	√	√	√	√	√	√	√	√	√	√

Εικόνα 10. Επιλογή κατάλληλων Essential prime implicants.

$$F = xy'z' + wz' + wy$$

3.1.7 Αριθμητικά Συστήματα & Κώδικες

Τα κύρια αριθμητικά συστήματα είναι το δεκαδικό που περιέχει αριθμούς απο 0 έως 9, το δυαδικό από 0 έως 1, το οκταδικό από 0 έως 8 και το δεκαεξαδικό από 0 έως 9 και από A έως F. Τα αριθμητικά συστήματα περιέχουν μετατροπή αριθμών από ένα σύστημα αρίθμησης α σε ένα σύστημα αρίθμησης β (9).
Παρακάτω εμφανίζεται πίνακας που περιέχει τα τέσσερα βασικά συστήματα αρίθμησης:

Δεκαδικό	Δυαδικό	Οκταδικό	Δεκαεξαδικό
0	0000	0	0
1	0001	1	1
2	0010	2	2
3	0011	3	3
4	0100	4	4
5	0101	5	5
6	0110	6	6
7	0111	7	7
8	1000	10	8
9	1001	11	9
10	1010	12	A
11	1011	13	B
12	1100	14	C
13	1101	15	D
14	1110	16	E
15	1111	17	F

Πίνακας 2. Κύρια Συστήματα Αρίθμησης.

Μια προσέγγιση για παράσταση θετικών και αρνητικών δυαδικών αριθμών είναι το συμπλήρωμα ως προς ένα και το συμπλήρωμα ως προς δύο. Στο συμπλήρωμα ως προς ένα ένας αριθμός μετατρέπεται από θετικός σε αρνητικό. Αυτό επιτυγχάνεται με την αλλαγή των 0 σε 1 και αντίστροφα. Το συμπλήρωμα ως προς δύο συσχετίζεται με την αλλαγή προσήμου του αριθμού. Στη συνέχεια, οι κυριότεροι κώδικες του δεκαδικού συστήματος είναι οι παρακάτω (10):

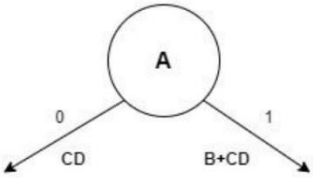
Δεκαδικό	BCD	Biquinary	2-4-2-1	Gray
0	0000	0100001	0000	0000
1	0001	0100010	0111	0001
2	0010	0100100	0110	0011
3	0011	0101000	0101	0010
4	0100	0110000	0100	0110
5	0101	1000001	1011	0111
6	0110	1000010	1010	0101
7	0111	1000100	1001	0100
8	1000	1001000	1000	1100
9	1001	1010000	1111	1101

Πίνακας 3. Κύριοι κώδικες δεκαδικού συστήματος.

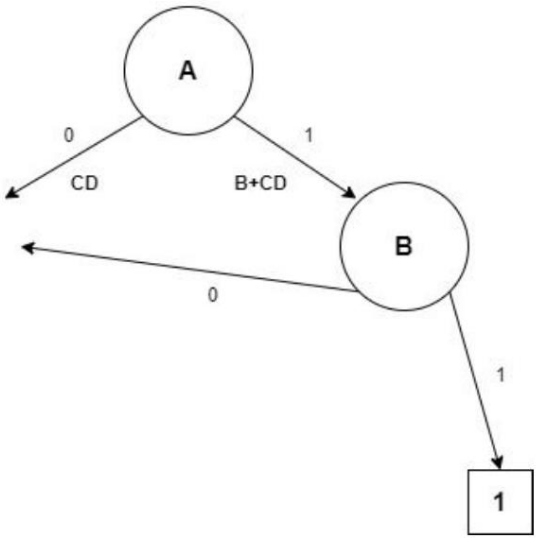
3.1.8 Διαγράμματα BDD's

Στην πληροφορική τα διαγράμματα BDD's αποτελούν μια δομή δεδομένων που χρησιμοποιείται για την απεικόνιση μίας συνάρτησης Boolean (11). Γενικά είναι ένας άκυκλικος, κατευθυνόμενος γράφος που αποτελείται από πολλούς κόμβους εκ των οποίων, οι δύο τερματικοί και δείχνουν στις τιμές 0 και 1. Εφόσον οι μεταβλητές εμφανίζονται με την ίδια πάντα σειρά, μέσα στο γράφο, το διάγραμμα ονομάζεται διατεταγμένο. Επίσης, μπορεί να ονομάζεται και ελαχιστοποιημένο όταν υπάρχει συγχώνευση ισομορφικών υπογράφων και αφαίρεση κόμβου όπου τα παιδιά είναι ισομορφικά. Ένα παράδειγμα διαγράμματος BDD παρουσιάζεται παρακάτω:

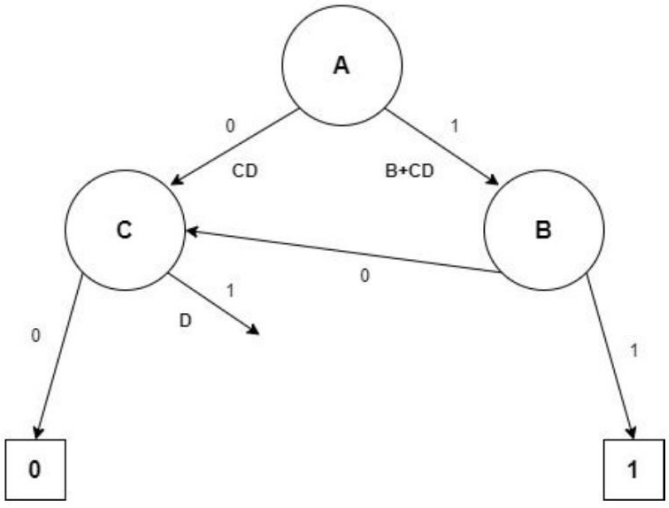
$$F = AB + CD, \quad A > B > C > D.$$



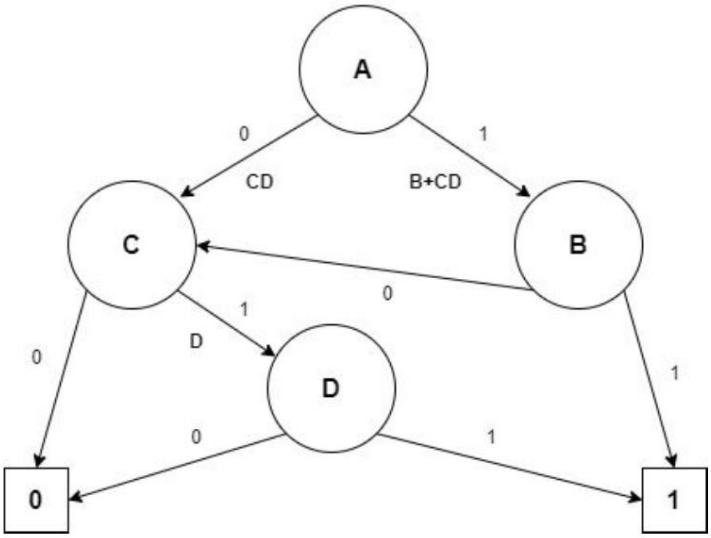
Εικόνα 11. Πρώτο βήμα για την μεταβλητή Α.



Εικόνα 12. Δεύτερο βήμα για την μεταβητή Β.



Εικόνα 13. Τρίτο βήμα για την μεταβλητή C.



Εικόνα 14. Τέταρτο βήμα για την μεταβλητή D.

3.1.10 Συνδυαστικά Κυκλώματα

Συνδυαστικά κυκλώματα ονομάζονται τα κυκλώματα στα οποία η τιμή της εξόδου ή των εξόδων τους εξαρτώνται αποκλειστικά την κάθε χρονική στιγμή από τον συνδυασμό των τρέχοντων τιμών των εισόδων. Τα βασικά στοιχεία που αποτελούν αυτά τα κυκλώματα είναι οι λογικές πύλες, δηλαδή αποτελούνται από εισόδους, λογικές πύλες και εξόδους. Η λειτουργία του κυκλώματος περιγράφεται με πίνακα αληθείας και για x μεταβλητές εισόδου υπάρχουν 2^x πιθανοί συνδυασμοί τιμών εξόδου (13). Η τελική σχεδίαση ενός συνδυαστικού κυκλώματος αποτελείται από την ανάλυση και την σύνθεση.

3.1.9 Ακολουθιακά Κυκλώματα

Ακολουθιακά είναι τα κυκλώματα που απομνημονεύουν προηγούμενες καταστάσεις και εξόδους του κυκλώματος. Η έξοδος τους εξαρτάται όχι μόνο από τις τιμές των εισόδων αλλά και από τις τιμές των εξόδων του κυκλώματος κάποια προηγούμενη χρονική στιγμή. Χαρακτηρίζονται ως κυκλώματα ανάδρασης διότι οι έξοδοι τους μπορούν να χρησιμοποιηθούν σαν είσοδοι στο κύκλωμα. Τα ακολουθιακά κυκλώματα χωρίζονται σε δύο βασικές κατηγορίες: τα σύγχρονα και τα ασύγχρονα. Τα σύγχρονα ακολουθιακά κυκλώματα λειτουργούν βάσει των τιμών στις εξόδους και στα στοιχεία μνήμης, σε διακριτές στιγμές του χρόνου με κάποιον παλμό χρονισμού, όπως είναι ένα ρολόι. Τα ασύγχρονα ακολουθιακά κυκλώματα λειτουργούν με σειρά αλλαγών των τιμών στις εισόδους σε συνεχή χρόνο και οι τιμές στις εξόδους τους μπορούν να αλλάξουν ανά πάσα στιγμή χωρίς κανένα συγχρονισμό. Ένα συνδυαστικό κύκλωμα με ανατροφοδότηση είναι ένα κύκλωμα ακολουθιακό το οποίο μπορεί να γίνει εύκολα ασταθές (12).

ΚΕΦΑΛΑΙΟ 4 Python

4.1 Ιστορική Αναδρομή

Η γλώσσα προγραμματισμού Python δημιουργήθηκε το 1989 (13) από τον Guido Van Rossum Ολλανδικής καταγωγής στο ερευνητικό κέντρο Centrum Wiskunde & Informatica, το οποίο επιτελεί βασική έρευνα στο πεδίο της Επιστήμης Υπολογιστών και Μαθηματικών. Η Python θεωρείται διάδοχος της γλώσσας ABC, η οποία υπήρξε και η βασική έμπνευση για την δημιουργία της. Αρχικά χρησιμοποιήθηκε σαν γλώσσα σεναρίων για το κατανεμημένο λειτουργικό σύστημα Amoeba. Ο Guido Van Rossum αναφέρει πως η μεγαλύτερη συνεισφορά της είναι ό,τι μπορεί να επεκτείνεται εύκολα.

Η Python σχεδιάστηκε έτσι ώστε να είναι ευέλικτα επεκτάσιμη, να παρέχει ενσωματωμένα στοιχεία και να δίνει την δυνατότητα στον προγραμματιστή να προσθέτει δικά του στοιχεία ανάλογα με τις ανάγκες του και το σύστημα που χρησιμοποιεί. Η πρώτη έκδοση της γλώσσας κυκλοφόρησε το 1991 (14) και η ονομασία της οφείλεται στην κωμική σειρά Monty Python's Flying Circus του BBC στην Μεγάλη Βρετανία στην δεκαετία του ’70. Ο δημιουργός της ήθελε ένα σύντομο όνομα το οποίο να είναι μυστηριώδες και μοναδικό. Συνεπώς δεν υπάρχει καμία σχέση στην ονομασία της με το είδος φιδιού, τον πύθωνα.

Η Python είναι μία γλώσσα προγραμματισμού που έχει γρήγορη εξέλιξη μέσω των νέων εκδόσεων με βασικό εργαλείο τα Python Enhancement Proposals. Αυτά είναι τυποποιημένα κείμενα που περιέχουν γενικές πληροφορίες, οδηγίες και περιγραφές για νέα χαρακτηριστικά της γλώσσας. Όταν γράφονταν τα κείμενα η Python είχε δύο σειρές εκδόσεων την 2.x και την 3.x. Η 2.x κυκλοφόρησε το 2000 και διακόπηκε το 2020 με την έκδοση 2.7.18, ενώ η 3.x κυκλοφόρησε το 2008 με νεότερη την 3.10.0 στις 4 Οκτωβρίου του 2021 (15). Γενικά η γλώσσα είναι ίδια και στις δύο εκδόσεις με πολλές μικρές διαφορές στα λεξικά και τις συμβολοσειρές.

4.2 Χρήση & Εφαρμογές της Python

Η Python έχει γίνει πολύ δημοφιλής τα τελευταία χρόνια και είναι από τις πιο διαδεδομένες γλώσσες προγραμματισμού. Υποστηρίζει αντικειμενοστραφή προγραμματισμό, δομημένο προγραμματισμό και λειτουργικά πρότυπα προγραμματισμού. Επιπλέον έχει την δυνατότητα να αναλύει και να απεικονίζει δεδομένα και να υλοποιεί μεγάλης κλίμακας εφαρμογές.

Έχει χρησιμοποιηθεί από πολλές γνωστές σε όλους μας εταιρείες, κάποιες εκ των οποίων αποτελούν την ελίτ του χώρου της πληροφορικής. Η Google (16) για παράδειγμα, χρησιμοποιεί την Python σαν μία από τις επίσημες γλώσσες του διακομιστή της. Μάλιστα, ο διευθυντής έρευνάς της αναφέρει πως «η Python ήταν ένα σημαντικό μέρος της Google από την αρχή και παραμένει έτσι καθώς το σύστημα μεγαλώνει και εξελίσσεται».

Το Facebook (16) χρησιμοποιεί την Python σε μεγάλο βαθμό για την διασφάλιση της ομαλής λειτουργίας της πλατφόρμας του, κατά την χρήση της αντίστοιχης εφαρμογής από περίπου 2 δισεκατομμύρια χρήστες.

Η PayPal (16) χρησιμοποιεί την Python σε ολόκληρη την υποδομή της με σκοπό την διασφάλιση ασφαλών χρηματικών συναλλαγών για τους χρήστες της. Το Instagram επίσης βασίζεται στην Python για την εύρυθμη λειτουργία του, τροφοδοτώντας τις αλληλεπιδράσεις πάνω από ενός δισεκατομμυρίου χρηστών. Το Spotify (16), χρησιμοποιεί την Python για τις back-end υπηρεσίες του και την ανάλυση δεδομένων. Το Netflix (16) επίσης χρησιμοποιεί την Python για την συλλογή δεδομένων και την ομαλή λειτουργία του.

Η NASA (16) στο σύστημα αυτοματισμού ροής εργασίας για σχεδιασμό αποστολών και διαχείριση δεδομένων χρησιμοποιεί Python. Οι μηχανικοί του Dropbox (16) βασίζονται στην Python για πολλούς λόγους. Η πολλαπλή υποστήριξη της γλώσσας, η ευκολία εκμάθησης και η αναγνωσιμότητα είναι οι κυριότεροι. Ακόμα και η ανάλυση δεδομένων και η κατασκευή τραπεζών κυματομορφών για την ανίχνευση γεγονότων στην αστρονομία βαρυτικών κυμμάτων χρησιμοποιεί Python.

Η Python είναι μια γλώσσα προγραμματισμού σχετικά εύκολη στην εκμάθισή της και με πληθώρα αποκλειστικών βιβλιοθηκών. Αυτά έχουν σαν αποτέλεσμα πολλές εταιρίες να προτιμούν την Python για να μπορούν να βρίσκουν πακέτα δεδομένων σε κάθε τομέα, τα οποία είναι έτοιμα και έχουν την δυνατότητα να χρησιμοποιηθούν άμεσα.

4.3 Πλεονεκτήματα & Μειονεκτήματα της Python

4.3.α Πλεονεκτήματα

Η Python είναι μία γλώσσα προγραμματισμού απλή, δεδομένου ότι το να διαβάζει κανείς ένα πρόγραμμα γραμμένο σε Python είναι σαν να διαβάζει Αγγλικά. Η ομοιότητα που έχει με ψευδοκώδικα είναι ένα από το πιο ισχυρά πλεονεκτήματα της, γεγονός που επιτρέπει στον προγραμματιστή να δίνει περισσότερη βάση στην αντιμετώπιση του εκάστοτε προβλήματος και λιγότερο στην συγγραφή του προγράμματος. Επίσης η απλότητα και η ομοιότητά της με την Αγγλική γλώσσα την καθιστά εύκολη στην εκμάθησή ακόμα και από άτομα τα οποία δεν είναι εξοικειωμένα με τον προγραμματισμό.

Ένα άλλο πλεονέκτημα της είναι ότι σαν γλώσσα προγραμματισμού είναι ελεύθερη και ανοιχτού κώδικα που σημαίνει ότι υπάρχει η δυνατότητα να μοιραστούν αντίγραφα αυτού του λογισμικού, να διαβαστεί ο πηγαίος κώδικας , να γίνουν αλλαγές σε αυτόν καθώς και να χρησιμοποιηθεί κομμάτι αυτού σε νέα προγράμματα. Αυτός είναι ένας σημαντικός λόγος για τον οποίο η Python συναντά συνεχείς βελτιώσεις.

Λόγω του ανοιχτού κώδικα, έχει προσαρμοστεί έτσι ώστε να μπορεί να λειτουργεί σε διαφορετικές πλατφόρμες, γεγονός το οποίο έχει σαν αποτέλεσμα ένα πρόγραμμα να μπορεί να τρέξει σε διάφορες πλατφόρμες χωρίς να χρειαστεί να γίνει κάποια μετατροπή. Μερικά λογισμικά στα οποία μπορεί να τρέξει η Python είναι

τα Windows, το Linux, το Mac OS, το Solaris, η Amiga, σε PlayStation, το FreeBSD, το AROS και πολλά άλλα.

Είναι γλώσσα πολύ υψηλού επιπέδου και παρέχεται δωρεάν σε όλους, με σημαντικό χαρακτηριστικό της την επεκτασιμότητα, το να μπορεί δηλαδή να χρησιμοποιήσει κομμάτια κώδικα γραμμένα σε άλλη γλώσσα προγραμματισμού για την εξοικονόμηση χρόνου.

Υποστηρίζει τόσο τον αντικειμενοστραφή προγραμματισμό, στον οποίο τα προγράμματα δομούνται πάνω σε αντικείμενα τα οποία συνδυάζουν δεδομένα και λειτουργικότητα, όσο και τον διαδικασιοστρεφή όπου τα προγράμματα δομούνται σε διαδικασίες και συναρτήσεις.

Στην Python υπάρχει τεράστιος αριθμός απο βιβλιοθήκες που απευθύνονται σε κάθε είδους προγραμματιστική εργασία και μπορούν εύκολα και άμεσα να χρησιμοποιηθούν.

4.3.β Μειονεκτήματα

Πέραν των πολλών πλεονεκτημάτων που έχει η Python υπάρχουν και κάποια μειονεκτήματα. Τα προγράμματα που είναι γραμμένα σε αυτή την γλώσσα δεν εκτελούνται τόσο γρήγορα όσο σε άλλες γλώσσες προγραμματισμού όπως η C++ , γεγονός που πηγάζει από το ότι είναι μία Interpreted Language, δηλαδή δεν μεταγλωττίζει τον κώδικα, αλλά τρέχει απευθείας τον πηγαίο κώδικα. Αυτό το μειονέκτημα είναι σοβαρό διότι τις περισσότερες φορές επιδιώκεται από τον χρήστη όσο το δυνατόν συντομότερος χρόνος εκτέλεσης.

Ένα περαιτέρω μειονέκτημα είναι ότι η Python είναι δύσκολα μεταφράσιμη σε άλλες γλώσσες προγραμματισμού πιο σύνθετες όπως, η Java και η C++ και σε ορισμένες περιπτώσεις χρειάζεται περισσότερη υπολογιστική δύναμη από αυτές. Παράλληλα δεν διαθέτει την δυνατότητα ταυτόχρονης εκτέλεσης εργασιών, το λεγόμενο multithreading.

4.4 Βιβλιοθήκες στην Python

Μια βιβλιοθήκη στην Python είναι ένα επαναχρησιμοποιήσιμο κομμάτι κώδικα. Ουσιαστικά, βιβλιοθήκη είναι μία συλλογή από ενότητες που μπορεί να εγκατασταθεί χρησιμοποιώντας έναν διαχειριστή πακέτων όπως rubygems ή npm. Η Python διαθέτει την μεγαλύτερη γκάμα βιβλιοθηκών με πάνω από 72.000 βιβλιοθήκες οι οποίες συνεχώς αυξάνονται και παραμένουν δωρεάν. Ο καθένας μπορεί να φτιάξει την δική του βιβλιοθήκη δωρεάν, να την χρησιμοποιήσει για δική του ευκολία αλλά και να την παρέχει σε τρίτους. Παρακάτω αναφέρονται μερικές από τις βασικές βιβλιοθήκες της γλώσσας:

- Pandas

Είναι μία βιβλιοθήκη για τον χειρισμό και την ανάλυση δεδομένων. Το Pandas δημιουργήθηκε το 2008 από τον Wes McKinney και το 2009 (17) έγινε ανοιχτού κώδικα. Δίνει την δυνατότητα στον χρήστη να εισάγει δεδομένα από διάφορες μορφές αρχείων και επιτρέπει διάφορες λειτουργίες χειραγώγησης δεδομένων.

- NumPy

Είναι μία βιβλιοθήκη που παρέχει υποστήριξη για πολυδιάστατες συστοιχίες και πίνακες όπως επίσης και μία συλλογή από μαθηματικές συναρτήσεις υψηλού επιπέδου για να βοηθήσουν στην λειτουργία των πινάκων. Αρχικά η βιβλιοθήκη αυτή με δημιουργό τον Travis Oliphant (18) ονομαζόταν Numeric και έκανε την εμφανισή της το 1995 εώς το 2006 όποτε και άλλαξε ονομασία σε NumPy.

- Scipy

Είναι μία βιβλιοθήκη η οποία παρέχει ευκολότερη και γρηγορότερη διαχείριση πινάκων όλων των διαστάσεων. Για την λειτουργία της απαραίτητη είναι η ύπαρξη της NumPy, από την οποία χρησιμοποιεί τους αριθμητικούς πίνακες. Η Scipy παρέχει ενότητες για βελτιστοποίηση, γραμμική άλγεβρα και άλλες εργασίες στην επιστήμη και τη μηχανική. Δημιουργήθηκε περίπου το 2001 (19) από τον Travis Oliphant.

- Matplotlib

Είναι μία βιβλιοθήκη για την δημιουργία γραφημάτων και γενικά για απεικόνιση. Έχει την δυνατότητα να δημιουργεί ιστογράμματα, διαγράμματα διασποράς, τρισδιάστατα διαγράματα, γραμμές πλοκής, διαγράμματα πίτας και άλλα πολλά είδη απεικόνισης. Η Matplotlib γράφτηκε αρχικά από τον John D. Hunter (20) το 2003.

- Scikit-learn

Είναι μία βιβλιοθήκη εκμάθησης μηχανικών λογισμικού, το λεγόμενο Machine Learning. Η βιβλιοθήκη περιέχει αλγόριθμους ταξινόμησης, παλινδρόμησης και ομαδοποίησης και λειτουργεί συνδυαστικά με τις βιβλιοθήκες NumPy και SciPy. Το Scikit-learn πρωτοεμφανίστηκε από τον David Cournapeau (21) το 2007.

- Ipython

Η Ipython είναι κέλυφος εντολών για διαδραστικό υπολογισμό σε διάφορες γλώσσες προγραμματισμού, και προσφέρει πιο λειτουργικό διερμηνέα στην γλώσσα με ένα κελί που περιέχει ενδοσκόπηση, ολοκλήρωση καρτελών, εμπλουτισμένα μέσα, σύνταξη κελιού, και ανάκτηση ιστορικού εντολών. Αποτελεί επίσης και τον ενσωματωμένο διερμηνέα για τα προγράμματα, που μπορεί να είναι πολύ χρήσιμος στο να εντοπιστούν σφάλματα. Η Ipython συγγράφηκε αρχικά από τον Fernando Perez (22) το 2001.

4.5 Εγκατάσταση της Python

Για την χρήση της γλώσσας προγραμματισμού Python θα χρειαστεί η εγκατάσταση και χρήση του Anaconda, το οποίο είναι μια δωρεάν διανομή των γλωσσών προγραμματισμού Python και R για επιστημονική υπολογιστική. Προσφέρει μια απλή και εύχρηστη ανάπτυξη και διαχείρηση των πακέτων της γλώσσας. Το Anaconda είναι συμβατό με Windows, macOS και Linux. Ιδρύθηκε από τους Peter Wang και Travis Oliphant το 2012. Σήμερα αναβαθμίζεται και συντηρείται από την Anaconda, Inc (23).

Ακόμα και με το Anaconda εγκατεστημένο η Python δεν διαθέτει όλα τα αναγκαία για έναν προγραμματιστή χαρακτηρηστικά, ο οποίος θέλει να αξιοποιήσει όλα τα πλεονεκτήματα που παρέχει η γλώσσα. Για να αποκτηθούν θα πρέπει να γίνει η αναζήτηση και η εγκατάσταση του κάθε πακέτου που θα χρειαστεί ξεχωριστά. Ο διαχειριστής πακέτων στην Python είναι το conda, ενώ μπορεί επίσης να χρησιμοποιηθεί και το pip.

4.6 Βασικά στοιχεία της Python

4.6.1 Αναγνωριστικά

Αναγνωριστικό ονομάζεται μια μεταβλητή, ή συνάρτηση, ή κλάση ή ένα αντικείμενο. Ένα αντικείμενο μπορεί να περιέχει γράμματα, αριθμούς όπως και το σύμβολο της κάτω παύλας (_). Απαγορεύεται να ξεκινάει από αριθμούς και δεν μπορεί να περιέχει άλλα σύμβολα και ειδικούς χαρακτήρες όπως για παράδειγμα $, @ και άλλα. Τέλος, τα πεζά με τα κεφαλαία δεν θεωρούνται ίδιοι χαρακτήρες.

4.6.2 Λέξεις – Κλειδιά

Είναι λέξεις που χρησιμοποιούνται για βασικές λειτουργίες σε ένα πρόγραμμα και δεν μπορούν να χρησιμοποιηθούν σαν μεταβλητή ή άλλο

αναγνωριστικό όνομα. Αποτελούνται μόνο από πεζά γράμματα της αγγλικής γλώσσας. Οι Λέξεις – Κλειδιά είναι οι παρακάτω:

and, def, del, as, assert, print, break, class, continue, elif, else, return, exec, finally, for, from, global, if, while, import, in, is, lambda, with, not, or, pass, raise, try, except, yield.

4.6.3 Εσοχή

Στην Python μια ομάδα εντολών ξεχωρίζεται από μια άλλη με διαφορετική εσοχή καθώς δεν υπάρχει εντολή κλεισίματος, γεγονός που καθιστά την χρήση της εσοχής υποχρεωτική. Οι εντολές που έχουν την ίδια εσοχή βρίσκονται στην ίδια ομάδα, και ο χώρος της εσοχής είναι μεταβλητός.

4.6.4 Εντολή help

Η σύνταξη της είναι: help (όνομα_εντολής) και χρησιμοποιείται για την λειτουργία και την σύνταξη κάποιας εντολής.

4.6.5 Δήλωση πολλών γραμμών

Γενικά οι δηλώσεις τελειώνουν όταν μεταβαίνει το πρόγραμμα σε νέα γραμμή. Σε περιπτώσεις όμως που ο πραγραμματιστής επθυμεί να την συνεχίσει σε νέα γραμμή, αυτό επιτυγχάνεται με την χρήση του συμβόλου (/). Το (/) δηλώνει ότι η γραμμή στην οποία βρίσκεται συνεχίζεται.

4.6.6 Σχόλια

Για την προσθήκη σχολίων (comments) σε κάποιο πρόγραμμα χρησιμοποιείται το σύμβολο της δίεσης (#). Αυτό υποδηλώνει ότι όλα τα υπόλοιπα τα οποία αναγράφονται στην ίδια γραμμή θεωρούνται σχόλια και δεν λαμβάνονται υπόψιν από τον κώδικα. Για την προσθήκη πολλών γραμμών με σχόλια θα πρέπει στην αρχή της κάθε γραμμής να υπάρχει επίσης το σύμβολο της δίεσης.

4.6.7 Εντολή print

Η εντολή print είναι ο απλούστερος τρόπος για την εμφάνιση δεδομένων στην οθόνη. Η print χρησιμοποιείται για να εμφανίζεται το αποτέλεσμα ενός κώδικα ή το αποτέλεσμα από ένα κομμάτι κώδικα στην οθόνη. Η συνταξή της είναι, print( ), όπου μέσα στις παρενθέσεις μπορούν να γραφούν οι μεταβλητές και με την χρήση εισαγωγικών μπορούν να εισαχθούν μηνύματα που σχετίζονται με την μεταβλητή που θα εμφανιστεί.

4.6.8 Εντολή input

Για την εισαγωγή μίας τιμής από το πληκτρολόγιο κατά την εκτέλεση του προγράμματος χρησιμοποιείται η εντολή input. Η συνταξή της είναι, input = (“

μήνυμα για εμφάνιση"). Όταν εκτελείται αυτή η εντολή το πρόγραμμα εμφανίζει στην οθόνη το μήνυμα που βρίσκεται μέσα στις παρενθέσεις και περιμένει από τον προγραμματιστή να εισάγει μια τιμή και να πατήσει enter για να συνεχίσει την εκτέλεσή του.

4.6.9 Πολλαπλές δηλώσεις σε μια γραμμή

Σε ένα πρόγραμμα υπάρχει η δυνατότητα να γίνει πάνω από μία δήλωση στην ίδια γραμμή αρκεί να χρησιμοποιηθεί το ερωτηματικό (;). Επομένως επιτρέπεται η πολλαπλή δήλωση σε μία γραμμή με την προϋπόθεση όλες να ανήκουν στην ίδια ομάδα εντολών του κώδικα.

4.6.10 Πολλαπλές ομάδες δηλώσεων

Με την χρήση τους δημιουργείται μία ενιαία ομάδα εντολών στον κώδικα και χαρακτηρίζονται ως σύνθετες καταστάσεις. Αυτές είναι οι εντολές, def, if, while και class. Οι δηλώσεις τους περιέχουν στην πρώτη γραμμή στην αρχή το όνομα της εντολής το οποίο είναι και Λέξη – κλειδί και στο τέλος της γραμμής την άνω και κάτω τελεία (:). Στις κάτω γραμμές υπάρχουν οι δηλώσεις που αποτελούν την ομάδα κώδικα.

4.6.11 Μεταβλητές

Το αναγνωριστικό το οποίο περιέχει ένα αντικείμενο ονομάζεται μεταβλητή. Το όνομα μίας μεταβλητής μπορεί να είναι οποιοσδήποτε χαρακτήρας εφόσον ο πρώτος χαρακτήρας είναι γράμμα και δεν είναι Λέξη – κλειδί. Όταν δημιουργείται μία μεταβλητή καταλαμβάνει θέσεις μνήμης για να μπορεί να αποθηκεύσει όποια τιμή της δωθεί. Υπάρχουν τέσσερις βασικοί τύποι μεταβλητών και αναγράφονται στην συνέχεια:

- Integer (int) : Ακέραιους αριθμούς, θετικούς ή αρνητικούς.
- Float : Πραγματικούς αριθμούς, δηλαδή με υποδιαστολή.
- Boolean : Εκφράσεις που είναι αληθείς ή ψευδείς.
- String (str) : Αλφαριθμητικά.

Δεν είναι αναγκαίο να δηλωθεί ο τύπος της μεταβλητής αφού καταχωρηθεί σε αυτή μια τιμή. Ταυτόχρονα με την καταχώρηση γίνεται και η δήλωση της μεταβλητής ώστε να αποκτήσει χώρο στη μνήμη. Η εντολή type (όνομα_μεταβλητής) είναι για να δώσει τον τύπο της μεταβλητής ενώ η εντολή id (όνομα_μεταβλητής) για να δώσει τη θέση μνήμης της μεταβλητής. Τέλος με τους τελεστές is και is not συγκρίνονται οι θέσεις μνήμης δύο μεταβλητών. Ο is επιστρέφει true αν είναι στις ίδιες θέσεις μνήμης και false όταν είναι σε διαφορετικές, αντιστρόφως ο is not.

Το εύρος μεταβλητών περιέχει δύο είδη, τις τοπικές μεταβλητές (local) και τις καθολικές μεταβλητές (global). Καθολική μεταβλητή είναι η μεταβλητή που έχει δηλωθεί στον κύριο κώδικα και όχι μέσα σε κάποια συνάρτηση. Μπορεί να χρησιμοποιηθεί σε όλα τα κομμάτια του κώδικα. Αντίθετα, μια τοπική μεταβλητή είναι δηλωμένη μέσα σε μια συνάρτηση και μπορεί να χρησιμοποιηθεί μόνο μέσα σε αυτή και όχι στον υπόλοιπο κώδικα. Μέσα σε μια συνάρτηση ανάμεσα σε μια τοπική και μια καθολική μεταβλητή υπεριοχύει η τοπική.

4.6.12 Μετατροπή τύπου δεδομένων

Σε μερικές περιπτώσεις υπάρχει η ανάγκη μετατροπής του τύπου κάποιας μεταβλητής. Η μετατροπή επιτυγχάνεται χρησιμοποιώντας σαν όνομα συνάρτησης τον τύπο δεδομένων που θέλουμε και μέσα στις παρενθέσεις το όνομα της μεταβλητής. Παρακάτω αναφέρονται ενδεικτικά μερικές απο αυτές:

Συνάρτηση	***Περιγραφή***
int(x)	Μετατρέπει το x σε ακέραιο.
float(x)	Μετατρέπει το x σε πραγματικό.
oct(x)	Μετατρέπει το x σε οκταδικό.
chr(x)	Μετατρέπει το x σε χαρακτήρα.
tuple(x)	Μετατρέπει το x σε πλειάδα.
list(x)	Μετατρέπει το x σε λίστα.
str(x)	Μετατρέπει το x σε αλφαριθμητικό.

Πίνακας 4. Συναρτήσεις μετατροπής τύπου δεδομένων.

4.6.13 Σταθερές

Οι σταθερές στην Python είναι οι:

- e , όπου είναι η βάση του λογαρίθμου $\log_e x$
- pi , όπου είναι η σταθερά π = 3,14

4.6.14 Τελεστές

Οι τελεστές αποτελούν σύμβολα πράξεων. Τα βασικά είδη των τελεστών είναι τα ακόλουθα:

Αριθμητικοί Τελεστές	
Τελεστής	***Λειτουργία***
+	Πρόσθεση
-	Αφαίρεση
*	Πολλαπλασιασμός
/	Διαίρεση
%	Υπόλοιπο διαίρεσης
**	Δύναμη
//	Ακέραιο πηλίκο διαίρεσης

Πίνακας 5. Αριθμητικοί Τελεστές.

Τελεστές Σύγκρισης	
Τελεστής	***Λειτουργία***
==	Ισότητα
>	Μεγαλύτερο
<	Μικρότερο
>=	Μεγαλύτερο ή Ίσο
<=	Μικρότερο ή Ίσο
!=	Ανισότητα

Πίνακας 6. Τελεστές Σύγκρισης.

Δυαδικοί Τελεστές	
Τελεστής	***Λειτουργία***
&	AND
<<	Κύλιση αριστερά
>>	Κύλιση δεξιά
~	Συμπλήρωμα
^	XOR
|	OR

Πίνακας 7. Δυαδικοί Τελεστές.

4.7 Δομές Ελέγχου & Επανάληψης

4.7.1 Δομή Ελέγχου if

Εφόσον μέσα σε ένα πρόγραμμα υπάρχει η ανάγκη να πληρείται μια προϋπόθεση για την εκτέλεση ορισμένων ή όλων των εντολών, χρησιμοποιείται η δομή ελέγχου if. Όταν η εκτέλεση του κώδικα συναντήσει την if θα ελέγξει εάν η συνθήκη είναι αληθής και ως αποτέλεσμα θα εκτελέσει τις εντολές που βρίσκονται μέσα σε αυτή. Ενώ αν η συνθήκη είναι ψευδής θα μεταβεί αμέσως μετά τις εντολές που βρίσκονται μέσα στην if για να συνεχίσει την εκτέλεση του προγράμματος. Όταν οι επιλογές είναι περισσότερες από μία τότε χρησιμοποιείται η if…else ή if…elif…else.

4.7.2 Δομές Επανάληψης

Μία δομή επανάληψης (βρόγχος επανάληψης) είναι ομάδα εντολών που χρειάζεται να εκτελεστούν πολλές διαδοχικές φορές και να δηλωθούν μία φορά στον κώδικα. Οι εντολές αυτές εκτελούνται όσο η συνθήκη που υπάρχει είναι αληθής.

Υπάρχουν δύο δομές επανάληψης, η while και η for. Η συνταξή τους είναι η ακόλουθη:

- while : while συνθήκη:
  εντολές

- for : for δείκτης = αρχική τιμή: βήμα: τελική τιμή
  εντολές

4.7.3 Δήλωση with & break

Το with μπορεί να χρησιμοποιηθεί για το άνοιγμα ενός αρχείου και τον έλεγχο του επιτυχούς ανοίγματός του. Γενικά είναι εντολή με πολλαπλές χρήσεις. Η break χρησιμοποιείται για έχει την δυνατότητα το πρόγραμμα που εκτελείται να βγεί από ένα βρόχο επανάληψης ή μια δομή if.

4.7.4 Συναρτήσεις

Οι συναρτήσεις γενικά χρησιμοποιούνται σε κομμάτια κώδικα που πρέπει να εκτελεστούν πάνω από μία φορά. Η εκτέλεση μίας συνάρτησης γίνεται είτε από κάποια άλλη συνάρτηση είτε από το κυρίως πρόγραμμα γράφοντας το όνομα της συνάρτησης που θα καλεστεί και μέσα σε παρενθέσεις τις παραμέτρους που δέχεται. Οι συναρτήσεις για να λειτουργήσουν πρέπει να ισχύουν τα ακόλουθα:

- Πρέπει να ξεκινάει με την λέξη – κλειδί def, μετά το όνομα της συνάρτησης, παρενθέσεις με τις παραμέτρους εισόδου και τέλος άνω και κάτω τελεία.
- Οι εντολές που περιέχει η συνάρτηση πρέπει να έχουν εσοχή.
- Για να επιστρέψει τιμή η συνάρτηση πρέπει να κλείνει με τη εντολή return.

4.8 Python & Λογική Σχεδίαση

Καμία από τις ήδη υπάρχουσες βιβλιοθήκες δεν περιέχει επαρκές υλικό για την επίλυση προβλημάτων συνδυαστικής λογικής. Αυτό το γεγονός υπήρξε αφορμή για την δουλειά που πραγματοποιήθηκε στην παρούσα πτυχιακή εργασία.

Ορισμένες βιβλιοθήκες που βρέθηκαν και είναι παρόμοιες με αυτή που δημιουργήθηκε αναφέρονται στον παρακάτω πίνακα:

Όνομα Βιβλιοθήκης	Λειτουργία
LogicPy (24)	Υλοποιεί βασικές λογικές συναρτήσεις όπως είναι η μετατροπή συστημάτων αρίθμησης, δυαδικό, οκταδικό, δεκαδικό, δεκαεξαδικό όπως επίσης και μετατροπή από δυαδικό σε κώδικα Gray και σε κώδικα BCD. Επίσης μπορεί να υλοποιήσει και πράξεις με όλες τις λογικές πύλες και flip flops.
logic-py (25)	Χρησιμεύει για την μετατροπή συστημάτων αρίθμησης, μετατροπή σε κώδικα BCD όπως και σε κώδικα Gray.
BinPy (26)	Χρησιμεύει για την επίλυση μόνο λογικών πυλών και δεν εξαρτάται από καμία άλλη βιβλιοθήκη.
truth-table-generator (27)	Δημιουργεί τον πίνακα αληθείας για μία λογική έκφραση.
quine-mccluskey (28)	Υλοποιεί τον αλγόριθμο Quine - McCluskey και δέχεται όσα ορίσματα θελήσει ο χρήστης να δώσει.

Πίνακας 8. Βιβλιοθήκες που σχετίζονται Λογική Σχεδίαση.

Γενικά υπάρχει πληθώρα βιβλιοθηκών με πολλές ομοιότητες μεταξύ τους. Στην αναζήτηση που έγινε βρέθηκαν αρκετές που υλοποιούσαν πράξεις με λογικές πύλες, άλλες που υλοποιούσαν μετατροπές συστημάτων αρίθμησης, πίνακες αληθείας ή τον αλγόριθμο Quine McCluskey. Δεν βρέθηκε όμως καμία η οποία να αναφέρεται στο συμπλήρωμα του 1 και του 2, στα BDD διαγράμματα, στο άθροισμα ελαχιστόρων και γινόμενο μεγιστόρων καθώς και στην μεταξύ τους μετατροπή.

Η βιβλιοθήκη που δημιουργήθηκε για την συγκεκριμένη πτυχιακή εργασία περιέχει όλα τα παραπάνω όπως και αυτά που δεν περιλαμβάνονται σε υπάρχουσες βιβλιοθήκες.

ΚΕΦΑΛΑΙΟ 5 Περιεχόμενα & Χαρακτηριστικά βιβλιοθήκης

Η βιβλιοθήκη που δημιουργήθηκε αποσκοπεί στην επίλυση προλβημάτων συνδυαστικής λογικής. Οι συναρτήσεις που περιέχονται στην βιβλιοθήκη αναλύονται επιγραμματικά παρακάτω:

- **devbin(number):**
  Υπολογισμός και εμφάνιση του αναλυτικού αναπτύγματος μιας αριθμητικής αναπαράστασης. Δέχεται ως είσοδο έναν αριθμό σε δυαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή αλφαριθμητικού.

- **devoct(number):**
  Υπολογισμός και εμφάνιση του αναλυτικού αναπτύγματος μιας αριθμητικής αναπαράστασης. Δέχεται ως είσοδο έναν αριθμό σε οκταδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή αλφαριθμητικού.

- **devdem(number):**
  Υπολογισμός και εμφάνιση του αναλυτικού αναπτύγματος μιας αριθμητικής αναπαράστασης. Δέχεται ως είσοδο έναν αριθμό σε δεκαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή αλφαριθμητικού.

- **devhex(number):**
  Υπολογισμός και εμφάνιση του αναλυτικού αναπτύγματος μιας αριθμητικής αναπαράστασης. Δέχεται ως είσοδο έναν αριθμό σε δεκαεξαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή αλφαριθμητικού.

- **detobi(number):**
  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής ενός δεκαδικού αριθμού σε δυαδικό. Δέχεται ως είσοδο έναν αριθμό σε δεκαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή ακεραίου.

- **dektobi(number):**
  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής ενός δεκαδικού αριθμού με δεκαδικό μέρος σε δυαδικό. Δέχεται ως είσοδο έναν αριθμό σε δεκαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή αλφαριθμητικού.

- **retwo(number):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής ενός πραγματικού δεκαδικού αριθμού με ή χωρίς δεκαδικό μέρος σε δυαδικό. Δέχεται ως είσοδο έναν αριθμό σε δεκαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή ακρεραίου εάν η είσοδος είναι ακέραιος και με την μορφή αλφαριθμητικού εάν η είσοδος έχει δεκαδικό μέρος.

- **basetwo(number):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής ενός οκταδικού αριθμού ή δεκαεξαδικού αριθμού σε δυαδικό. Δέχεται ως είσοδο έναν αριθμό σε οκταδική ή δεκαεξαδική αναπαράσταση και επιστρέφει το αποτέλεσμα σε μορφή ακεραίου.

- **caland(*argv):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της λογικής πύλης AND. Δέχεται ως είσοδο από 2 μέχρι N αριθμούς 0 ή 1 και επιστρέφει το αλφαριθμητικό αποτέλεσμα « True » ή « False ».

- **calnand(*argv):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της λογικής πύλης NAND. Δέχεται ως είσοδο από 2 μέχρι N αριθμούς 0 ή 1 και επιστρέφει το αλφαριθμητικό αποτέλεσμα « True » ή « False ».

- **calor(*argv):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της λογικής πύλης OR. Δέχεται ως είσοδο από 2 μέχρι N αριθμούς 0 ή 1 και επιστρέφει το αλφαριθμητικό αποτέλεσμα « True » ή « False ».

- **calxor(*argv):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της λογικής πύλης XOR. Δέχεται ως είσοδο από 2 μέχρι N αριθμούς 0 ή 1 και επιστρέφει το αλφαριθμητικό αποτέλεσμα « True » ή « False ».

- **calxnor(*argv):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της λογικής πύλης XNOR. Δέχεται ως είσοδο από 2 μέχρι N αριθμούς 0 ή 1 και επιστρέφει το αλφαριθμητικό αποτέλεσμα « True » ή « False ».

- **calnot(number):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της λογικής πύλης NOT. Δέχεται ως είσοδο έναν δυαδικό αριθμό από 1 μέχρι N bits και επιστρέφει το αποτέλεσμα με την μορφή αλφαριθμητικού.

- **supone(number):**

  Υπολογισμός και εμφάνιση του αποτελέσματος του συμπληρώματος ως προς ένα. Δέχεται ως είσοδο έναν αριθμό σε δυαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή ακεραίου.

- **suptwo(number):**

  Υπολογισμός και εμφάνιση του αποτελέσματος του συμπληρώματος ως προς δύο. Δέχεται ως είσοδο έναν αριθμό σε δυαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή ακεραίου.

- **subase(a,b):**

  Υπολογισμός και εμφάνιση του αποτελέσματος αφαίρεσης συμπληρώματος ως προς δύο. Δέχεται ως είσοδο δύο αριθμούς σε δυαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή ακεραίου.

- **numbcd(number):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής ενός δυαδικού αριθμού σε κώδικα BCD. Δέχεται ως είσοδο έναν αριθμό σε δυαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή αλφαριθμητικού.

- **numbiqu(number):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής ενός δυαδικού αριθμού σε κώδικα Biquinary. Δέχεται ως είσοδο έναν αριθμό σε δυαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή αλφαριθμητικού.

- **numgray(number):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής ενός δυαδικού αριθμού σε κώδικα Gray. Δέχεται ως είσοδο έναν αριθμό σε δυαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή αλφαριθμητικού.

- **num2421(number):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής ενός δυαδικού αριθμού σε κώδικα 2421. Δέχεται ως είσοδο έναν αριθμό σε δυαδική αναπαράσταση και επιστρέφει το αποτέλεσμα με την μορφή αλφαριθμητικού.

- **suminima(s):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής μιας συνάρτησης σε άθροισμα ελαχιστόρων. Δέχεται ως είσοδο μία συνάρτηση αλφαριθμητική αναπαράσταση και επιστρέφει το αποτέλεσμα σε μία λιστα με ακεραίους.

- **mulmax(s):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής μιας συνάρτησης σε γινόμενο μεγιστόρων. Δέχεται ως είσοδο μία συνάρτηση αλφαριθμητική αναπαράσταση και επιστρέφει το αποτέλεσμα με τη μορφή αλφαριθμητικού.

- **consuminimamulmax(s):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής από άθροισμα ελαχιστόρων σε γινόμενο μεγιστόρων. Δέχεται ως είσοδο ένα άθροισμα ελαχιστόρων με την μορφή αλφαριθμητικού και επιστρέφει το αποτέλεσμα με τη μορφή αλφαριθμητικού.

- **conmulmaxsuminima(s):**

  Υπολογισμός και εμφάνιση του αποτελέσματος της μετατροπής από γινόμενο μεγιστόρων σε άθροισμα ελαχιστόρων. Δέχεται ως είσοδο ένα γινόμενο μεγιστόρων με την μορφή αλφαριθμητικού και επιστρέφει το αποτέλεσμα με τη μορφή αλφαριθμητικού.

- **truthtable(s,y):**

  Υπολογισμός και εμφάνιση του πίνακα αληθείας μιας συνάρτησης. Δέχεται ως είσοδο μία συνάρτηση σε μορφή αλφαριθμητικής αναπαράστασης, έναν ακέραιο που δηλώνει πόσες μεταβλητές υπάρχουν και εμφανίζει τον πίνακα σε αλφαριθμητική αναπαράσταση.

- **karntwo(a,b,c,d):**

  Υπολογισμός και εμφάνιση της απλοποιημένης συνάρτησης με την χρήση του χάρτη Karnaugh 2 μεταβλητών. Δέχεται ως είσοδο τα 4 στοιχεία του πίνακα σε ακέραια μορφή και επιστρέφει το αποτέλεσμα σε αλφαριθμητική αναπαράσταση.

- **karntwositu(a,b,c,d,k):**

  Υπολογισμός και εμφάνιση της απλοποιημένης συνάρτησης με την χρήση του χάρτη Karnaugh 2 μεταβλητών με καταστάσεις αδιαφορίας. Δέχεται ως είσοδο τα 4 στοιχεία του πίνακα σε ακέραια μορφή, τις καταστάσεις αδιαφορίας σε αλφαριθμητική μορφή και επιστρέφει το αποτέλεσμα σε αλφαριθμητική αναπαράσταση.

- **karnthree(a,b,c,d,e,f,g,h):**

  Υπολογισμός και εμφάνιση της απλοποιημένης συνάρτησης με την χρήση του χάρτη Karnaugh 3 μεταβλητών. Δέχεται ως είσοδο τα 8 στοιχεία του πίνακα σε ακέραια μορφή και επιστρέφει το αποτέλεσμα σε αλφαριθμητική αναπαράσταση.

- **karnthree(a,b,c,d,e,f,g,h,k):**

  Υπολογισμός και εμφάνιση της απλοποιημένης συνάρτησης με την χρήση του χάρτη Karnaugh 3 μεταβλητών με καταστάσεις αδιαφορίας. Δέχεται ως είσοδο τα 8 στοιχεία του πίνακα σε ακέραια μορφή, τις καταστάσεις αδιαφορίας σε αλφαριθμητική μορφή και επιστρέφει το αποτέλεσμα σε αλφαριθμητική αναπαράσταση.

- **karnfour (m0,m1,m2,m3,m4,m5,m6,m7,m8,m9,m10,m11,m12,m13,m14,m15):**

  Υπολογισμός και εμφάνιση της απλοποιημένης συνάρτησης με την χρήση του χάρτη Karnaugh 4 μεταβλητών. Δέχεται ως είσοδο τα 16

στοιχεία του πίνακα σε αλφαριμθητική μορφή και επιστρέφει το αποτέλεσμα σε αλφαριθμητική αναπαράσταση.

- **karnfour (m0,m1,m2,m3,m4,m5,m6,m7,m8,m9,m10,m11,m12,m13,m14,m15,k):**
  Υπολογισμός και εμφάνιση της απλοποιημένης συνάρτησης με την χρήση του χάρτη Karnaugh 4 μεταβλητών με καταστάσεις αδιαφορίας. Δέχεται ως είσοδο τα 16 στοιχεία του πίνακα σε αλφαριθμητική μορφή, τις καταστάσεις αδιαφορίας σε αλφαριθμητική μορφή και επιστρέφει το αποτέλεσμα σε αλφαριθμητική αναπαράσταση.

- **mccluskey(vars, minterms):**
  Υπολογισμός και εμφάνιση του αποτελέσματος μετά την χρήση του αλγορίθμου Quine – McCluskey. Δέχεται ως είσοδο τις μεταβλητές που υπάρχουν σε αλφαριθμητική μορφή, τις εισροές σαν λίστα με ακεραίους και εμφανίζει το αποτέλεσμα σε αλφαριθμητική μορφή.

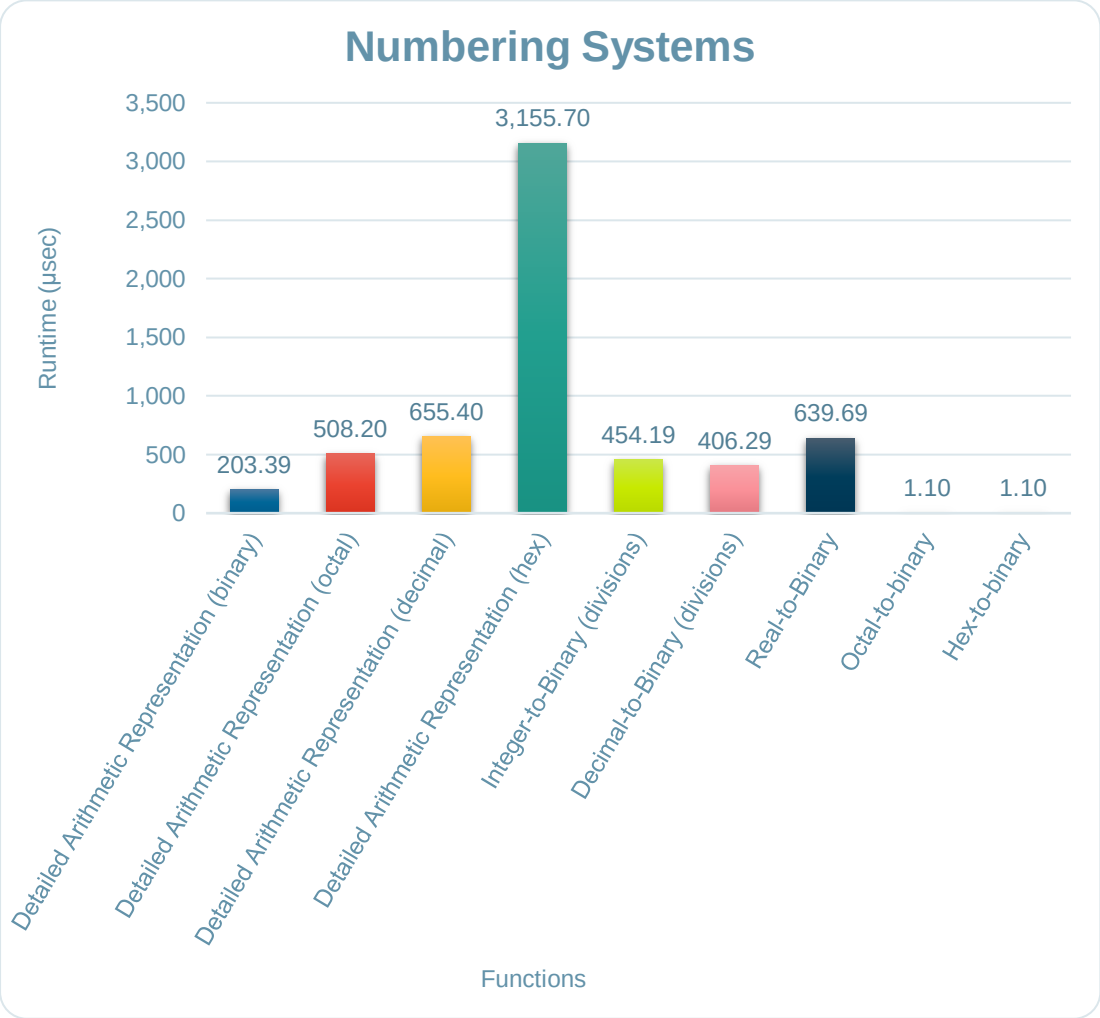
- **mccluskeysoi(mt,dc):**
  Υπολογισμός και εμφάνιση του αποτελέσματος μετά την χρήση του αλγορίθμου Quine – McCluskey με καταστάσεις αδιαφορίας. Δέχεται ως είσοδο τις εισροές της συνάρτησης σε αλφαριθμητική μορφή, τις καταστάσεις αδιαφορίας σε αλφαριθμητική μορφή και εμφανίζει το αποτέλεσμα σε αλφαριθμητική μορφή.

- **bdds(f,x,order):**
  Υπολογισμός και εμφάνιση του διαγράμματος BDD. Δέχεται σαν είσοδο τη συνάρτηση σε αλφαριθμητική μορφή, τον αριθμό των μεταβλητών σε ακέραια μορφή, την σειρά των μεταβλητών σε αλφαριθμητική μορφή και εμφανίζει τα βήματα του διαγράμματος.

ΚΕΦΑΛΑΙΟ 6 Αποτελέσματα & Μελλοντικές Προεκτάσεις

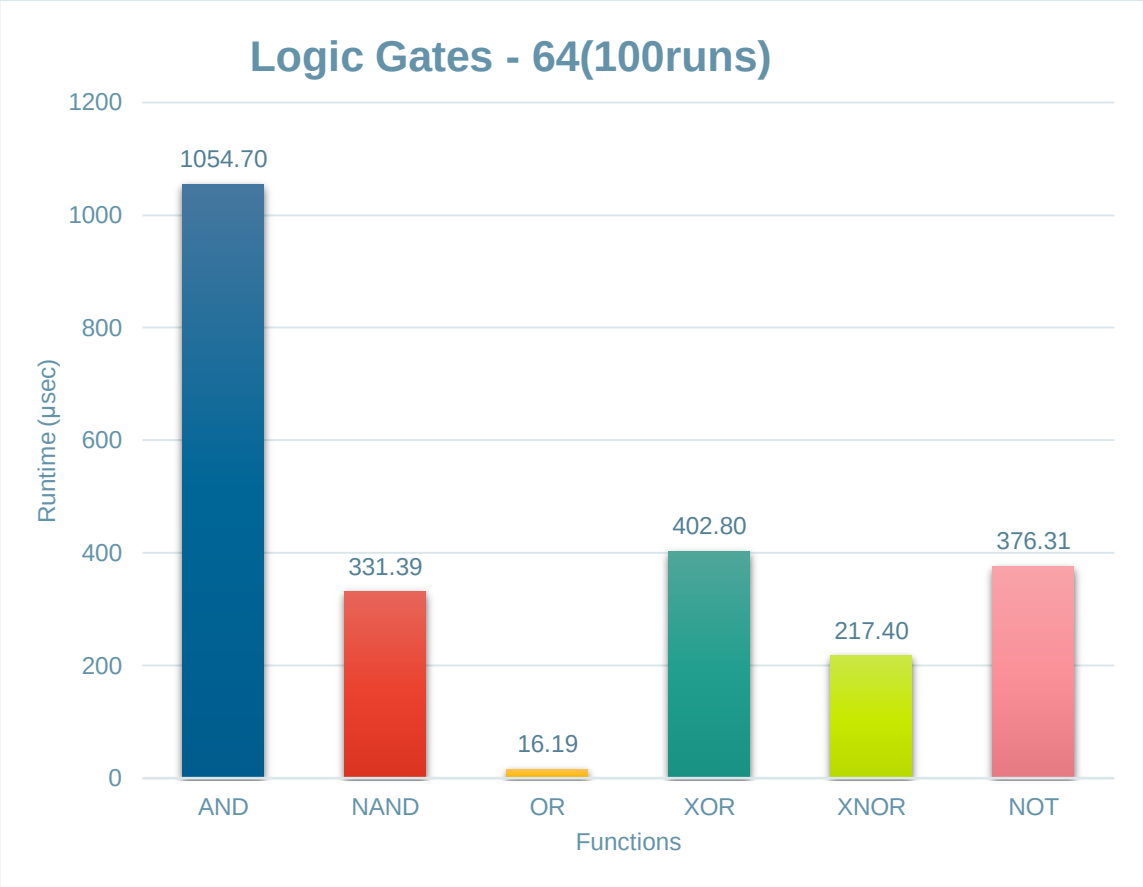
6.1 Αποτελέσματα

Η βιβλιοθήκη που υλοποιήθηκε περιέχει 35 συναρτήσεις, οι χρόνοι εκτέλεσης των οποίων συλλέχθηκαν μετά από διαδοχικά τρεξίματα στο Spyder (έκδοση 4.1.5). Ο υπολογιστής που χρησιμοποιήθηκε για την εκτέλεση των συναρτήσεων διαθέτει επεξεργαστή IntelCore i7 3.40 GHz και RAM 8Gb. Οι τιμές που συλλέχθηκαν προήλθαν από 100 διαδοχικές εκτελέσεις της εκάστοτε συνάρτησης με τυχαίες εισόδους. Η πλειοψηφία των συναρτήσεων εκτελέστηκε και για διαφορετικό μέγεθος εισόδων έτσι ώστε να διαμορφωθεί μια πληρέστερη εικόνα για το πώς, και αν, αυξάνεται ο χρόνος εκτέλεσης τους, αν τα δεδομένα που δίνονται είναι μεγαλύτερα σε όγκο. Παρακάτω παρατίθενται τα διαγράμματα χρόνων εκτέλεσης των συναρτήσεων.



Εικόνα 9. Μέσοι χρόνοι εκτέλεσης των προβαλλόμενων συναρτήσεων για τυχαίες εισόδους.

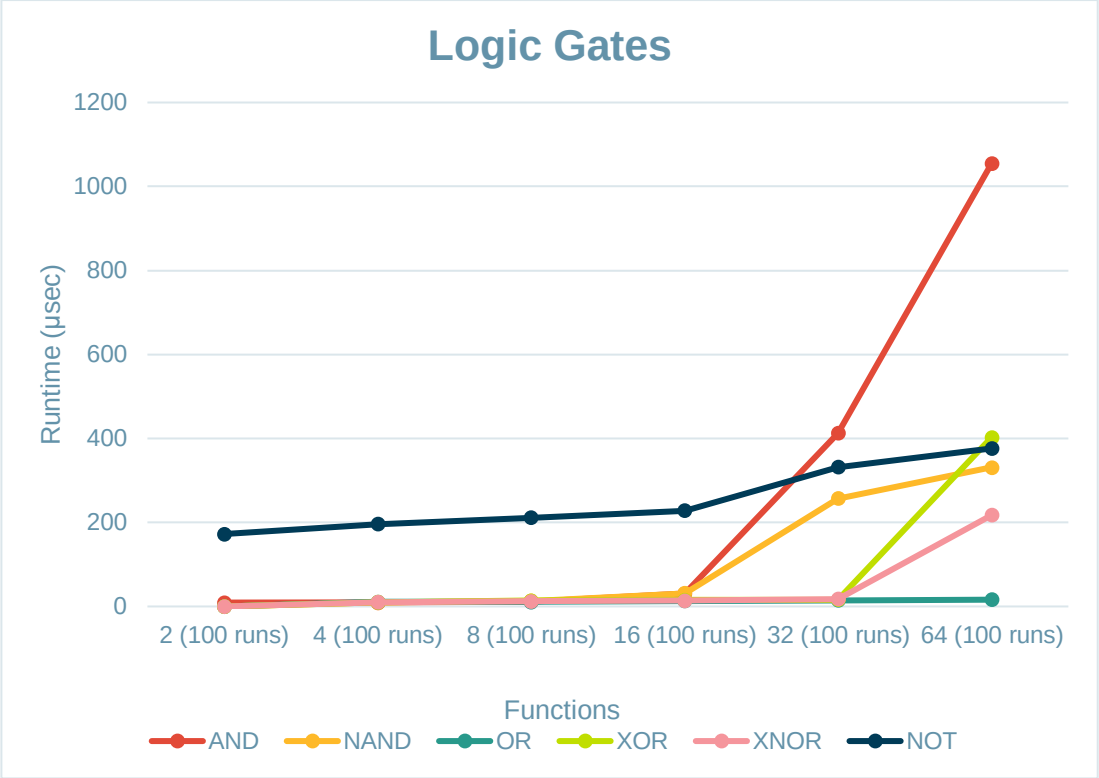
Στην Εικόνα 9 αναπαρίστανται οι μέσοι χρόνοι εκτέλεσης των 8 πρώτων συναρτήσεων για τυχαίες εισόδους από 100 τρεξίματα. Παρατηρούμε ότι οι χρόνοι εκτέλεσης κυμαίνονται απο 1,10 μsec έως 3.155,70 μsec. Βλέπουμε ότι η συνάρτηση Detailed Arithmetic Representation (hex) εμφανίζει τον μεγαλύτερο μέσο όρο εκτέλεσης ενώ οι υπόλοιπες εμφανίζουν περίπου ίδιας τάξης μεγέθους χρόνους εκτέλεσης. Πιθανόν αυτό έχει άμεση συσχέτιση με τον τρόπο γραφής της συγκεκριμένης συνάρτησης ο οποίος αυξάνει την πολυπλοκότητά της, γεγονός που αποτυπώνεται στον μεγάλο σχετικά μέσο χρόνο εκτέλεσής της.



Εικόνα 10. Μέσοι χρόνοι εκτέλεσης των προβαλλόμενων λογικών πυλών για τυχαίες εισόδους 64 bits.

Στην Εικόνα 10 φαίνονται οι μέσοι χρόνοι εκτέλεσης των συναρτήσεων που αφορούν τις λογικές πύλες (AND, NAND, OR, XOR, XNOR, NOT) για τυχαίες εισόδους 64 bits από 100 τρεξίματα. Παρατηρούμε ότι χρόνοι κυμαίνονται από 16,16 μsec έως 1054,70 μsec.

Στην Εικόνα 11 φαίνονται οι μέσοι χρόνοι εκτέλεσης των συναρτήσεων που αφορούν τις λογικές πύλες (AND, NAND, OR, XOR, XNOR, NOT) για τυχαίες εισόδους για 2-4-8-16-32-64 bits από 100 τρεξίματα. Παρατηρούμε μια αυξητική τάση στους χρόνους εκτέλεσης ανάλογα με την αύξηση των bits της εισόδου το οποίο και είναι αναμενόμενο.



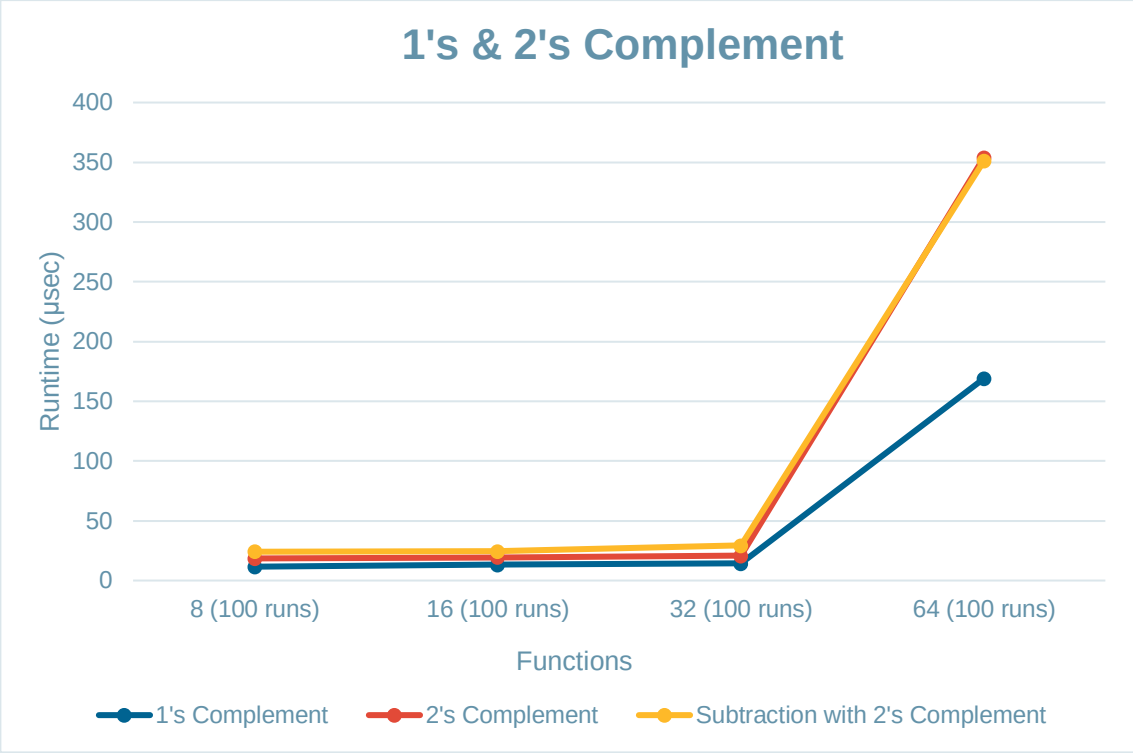
Εικόνα 11. Μέσοι χρόνοι εκτέλεσης των προβαλλόμενων λογικών πυλών για τυχαίες εισόδους διαφόρων μεγεθών.

Στην Εικόνα 12 φαίνονται διαγραμματικά οι μέσοι χρόνοι εκτέλεσης των συναρτήσεων που αφορούν τα συμπληρώματα 1 και 2 καθώς και την αφαίρεση με βάση το συμπλήρωμα του 2, για τυχαίες εισόδους για 8-16-32-64 bits από 100 τρεξίματα. Και εδώ υπάρχει μια αυξητική τάση στους χρόνους εκτέλεσης ανάλογα με την αύξηση των bits της εισόδου το οποίο και είναι αναμενόμενο.

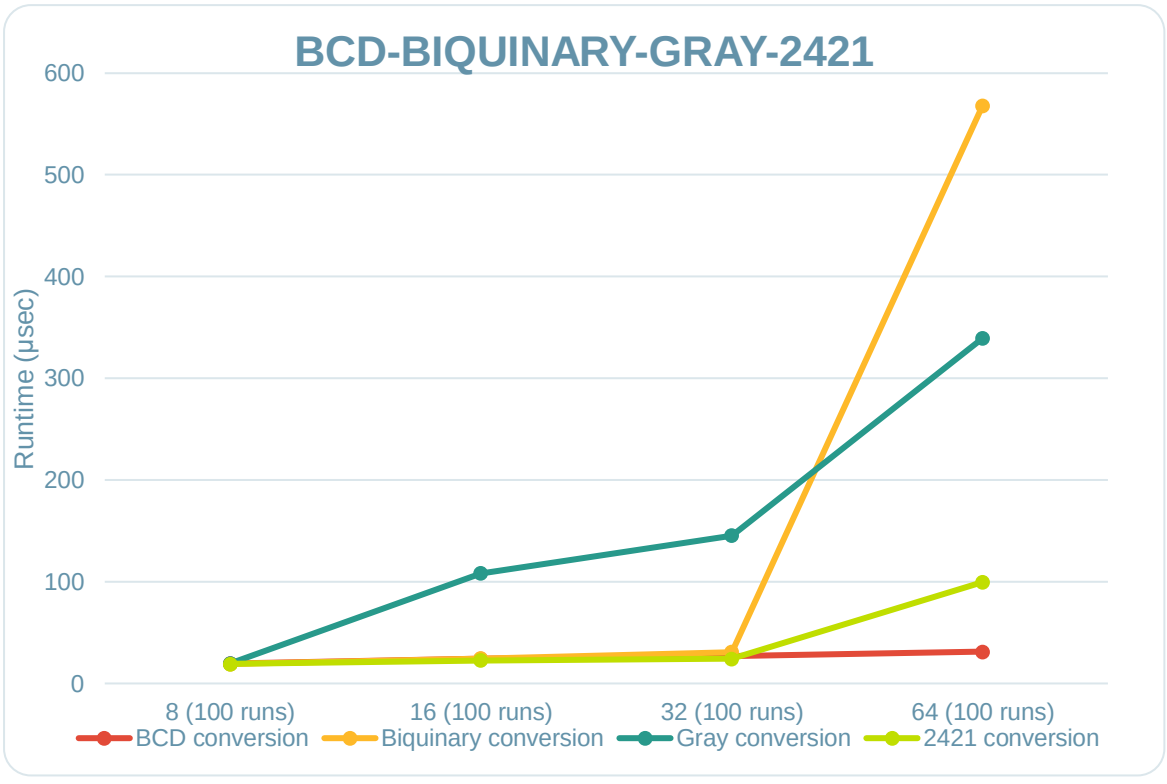
Στην Εικόνα 13 φαίνονται διαγραμματικά οι μέσοι χρόνοι εκτέλεσης των συναρτήσεων που αφορούν τον κώδικα BCD, τον κώδικα Biquinary, τον κώδικα Gray και τον κώδικα 2421, για τυχαίες εισόδους για 8-16-32-64 bits από 100 τρεξίματα. Υπάρχει μια αυξητική τάση στους χρόνους εκτέλεσης ανάλογα με την αύξηση των bits της εισόδου το οποίο και είναι αναμενόμενο. Την μεγαλύτερη αύξηση την έχει η Biquinary conversion.

Στην Εικόνα 14 αναπαρίστανται διαγραμματικά οι μέσοι χρόνοι εκτέλεσης των συναρτήσεων για το άθροισμα ελαχιστόρων, γινόμενο μεγιστόρων καθώς και για την μετατροπή τους, για τυχαίες εισόδους για 2-4-8-16-32-64 bits από 100 τρεξίματα. Και εδώ υπάρχει μια αυξητική τάση στους χρόνους εκτέλεσης ανάλογα με την αύξηση των bits της εισόδου το οποίο και είναι αναμενόμενο. Οι συναρτήσεις που παρουσιάζουν την μεγαλυτερη αύξηση είναι η Sum of Minterms

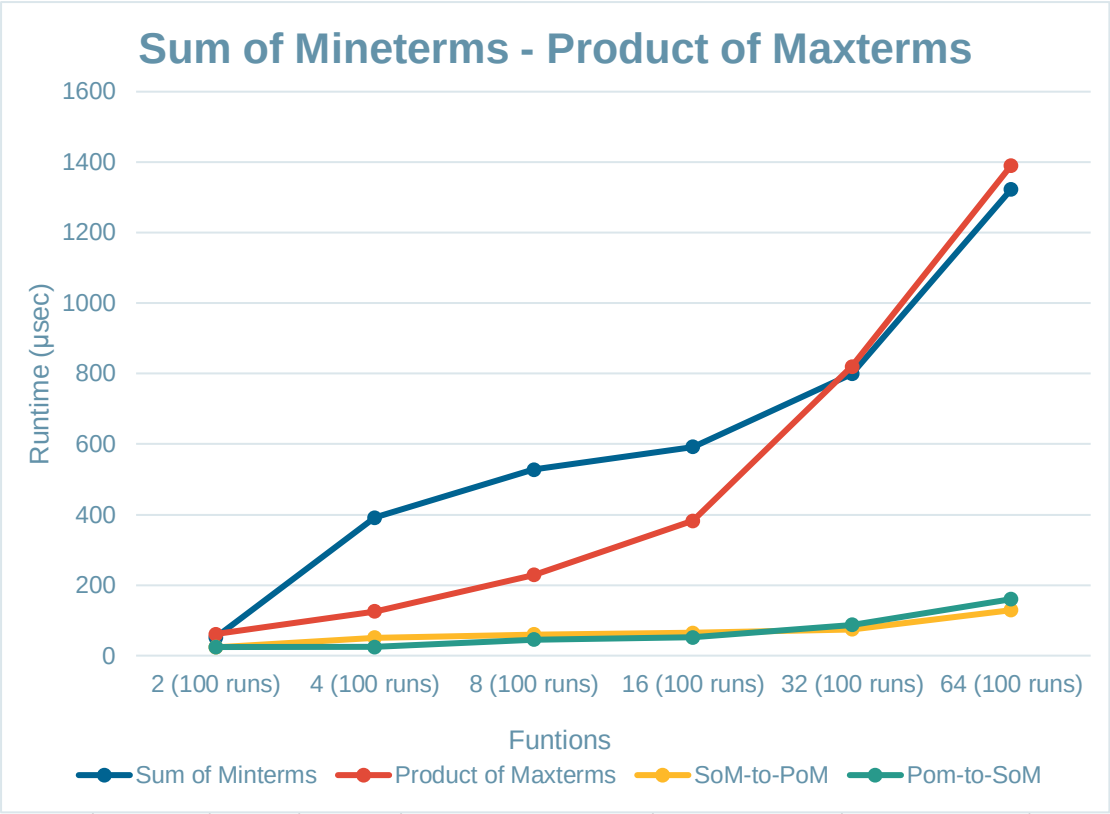
και η Product of Maxterms όπου είναι και αναμενόμενο δίοτι είναι πιο πολύπλοκες στη συγγραφή τους.



Εικόνα 12. Μέσοι χρόνοι εκτέλεσης των προβαλλόμενων συναρτήσεων για τυχαίες εισόδους διαφόρων μεγεθών.



Εικόνα 13. Μέσοι χρόνοι εκτέλεσης των προβαλλόμενων συναρτήσεων για τυχαίες εισόδους διαφόρων μεγεθών.

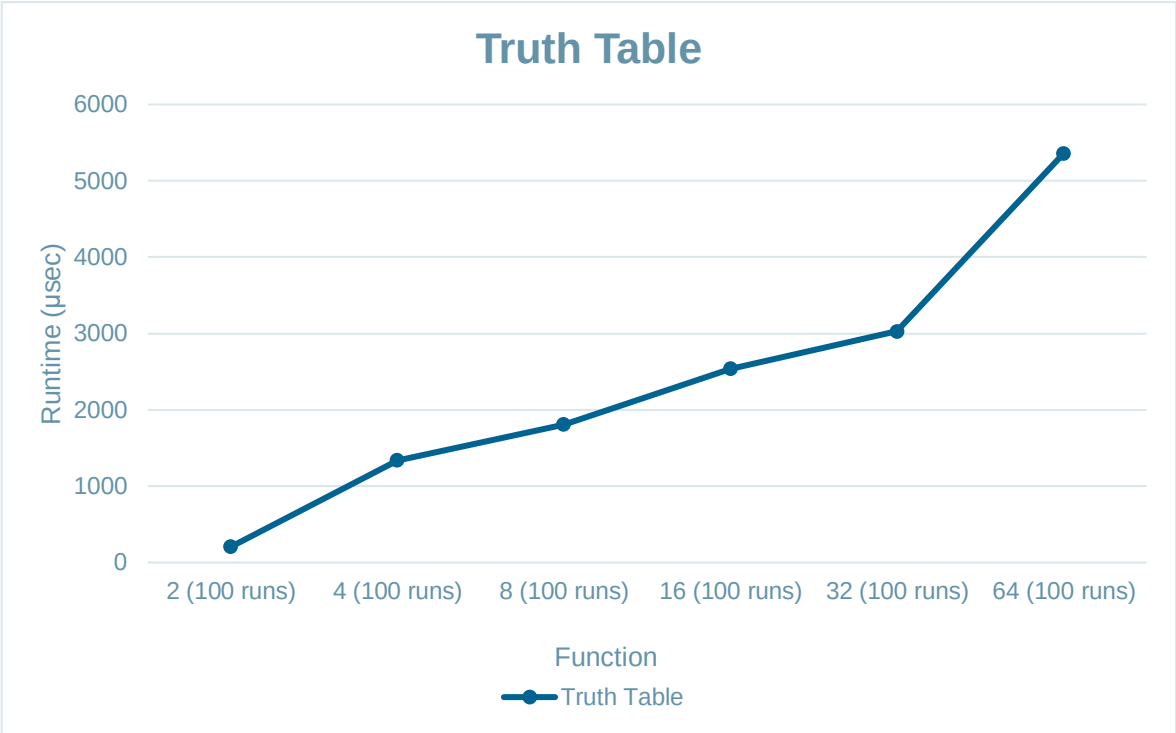


Εικόνα 14. Μέσοι χρόνοι εκτέλεσης των προβαλλόμενων συναρτήσεων για τυχαίες εισόδους διαφόρων μεγεθών.

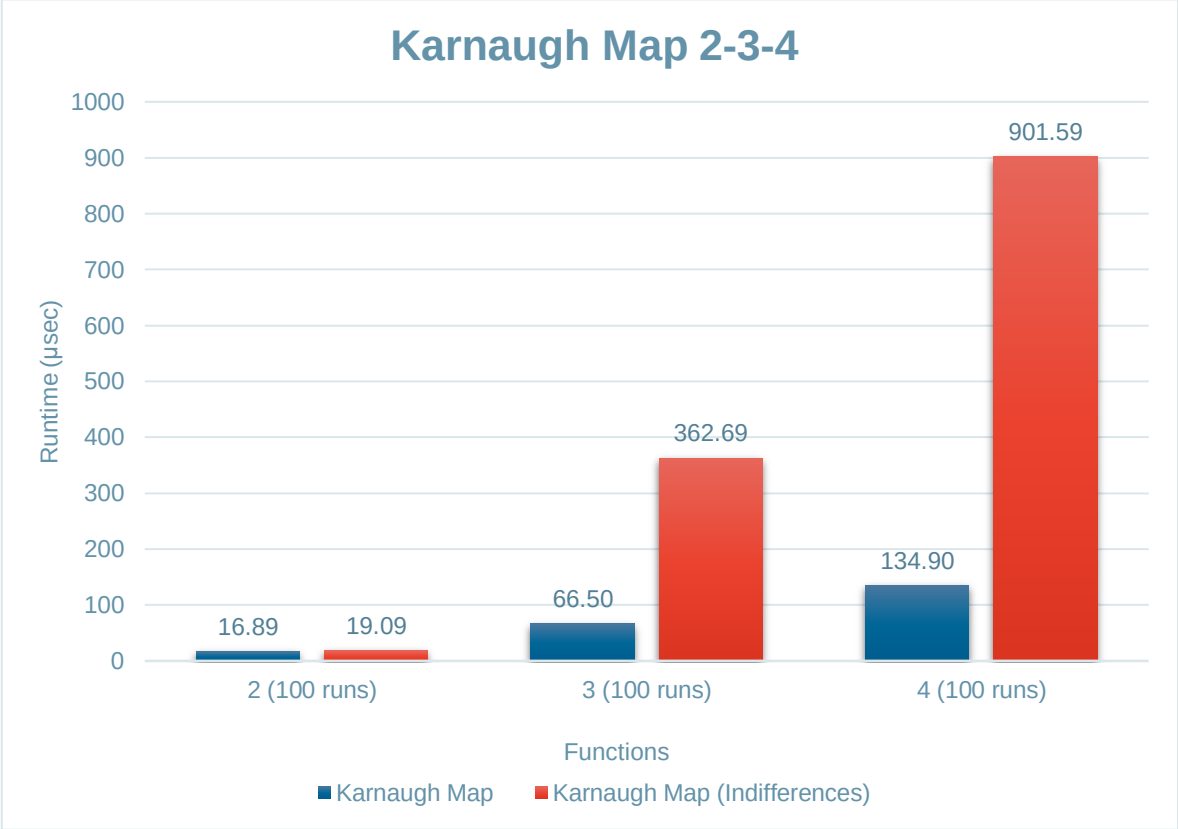
Στην Εικόνα 15 αναπαρίσταται διαγραμματικά ο μέσος χρόνος εκτέλεσης της Truth Table, για τυχαίες εισόδους για 2-4-8-16-32-64 bits από 100 τρεξίματα. Παρατηρούμε μια αυξητική τάση το οποίο είναι αναμενόμενο όταν αυξάνονται τα δεδομένα που λαμβάνει σαν είσοδο.

Στην Εικόνα 16 αναπαρίσταται διαγραμματικά ο μέσος χρόνος εκτέλεσης των συναρτήσεων Karnaugh για 2, 3, 4 μεταβλητές καθώς και για Karnaugh με καταστάσεις αδιαφορίας, για τυχαίες εισόδους από 100 τρεξίματα. Παρατηρούμε ότι οι συναρτήσεις Karnaugh για 2 μεταβλητές παρουσιάζουν τους χαμηλότερους χρόνους το οποίο είναι αναμενόμενο όπως και για 4 μεταβλητές να έχει τον υψηλότερο χρόνο εκτέλεσης. Παράλληλα βλέπουμε ότι για τις ίδιες εισόδους οι συναρτήσεις με τις καταστάσεις αδιαφορίας έχουν μεγαλύτερο χρόνο εκτέλεσης το οποίο είναι λογικό καθώς είναι πιο πολύπλοκες.

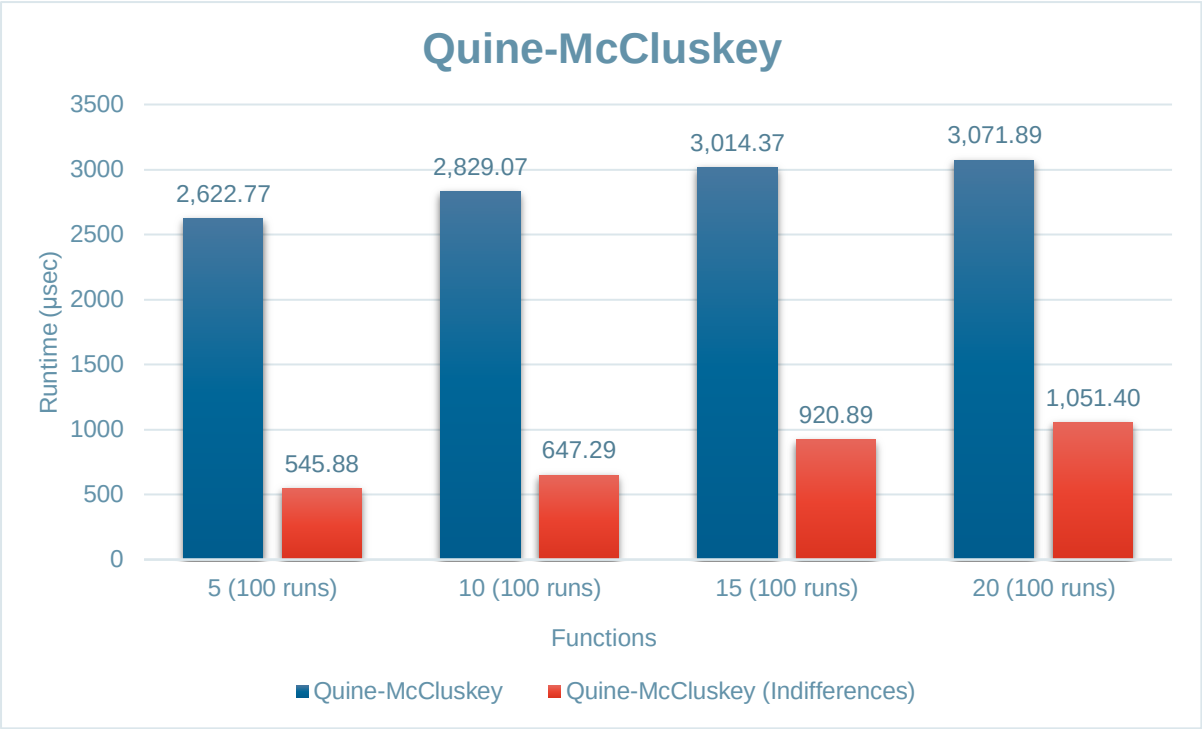
Στην Εικόνα 17 φαίνεται διαγραμματικά ο μέσος χρόνος εκτέλεσης των συναρτήσεων Quine-McCluskey για 5-10-15-20 τυχαίες εισόδους από 100 τρεξίματα. Παρατηρούμε ότι η συνάρτηση Quine-McCluskey καθώς και η Quine-McCluskey(Indifferences) έχουν σταδιακή αύξηση στον χρόνο εκτέλεσής τους όσο αυξάνονται και οι εισόδοι. Ανάμεσα στις 2 συναρτήσεις η πρώτη εμφανίζει αρκετά μεγαλύτερους χρόνους εκτέλεσης, το οποίο οφείλεται στην συγγραφή του κώδικα.



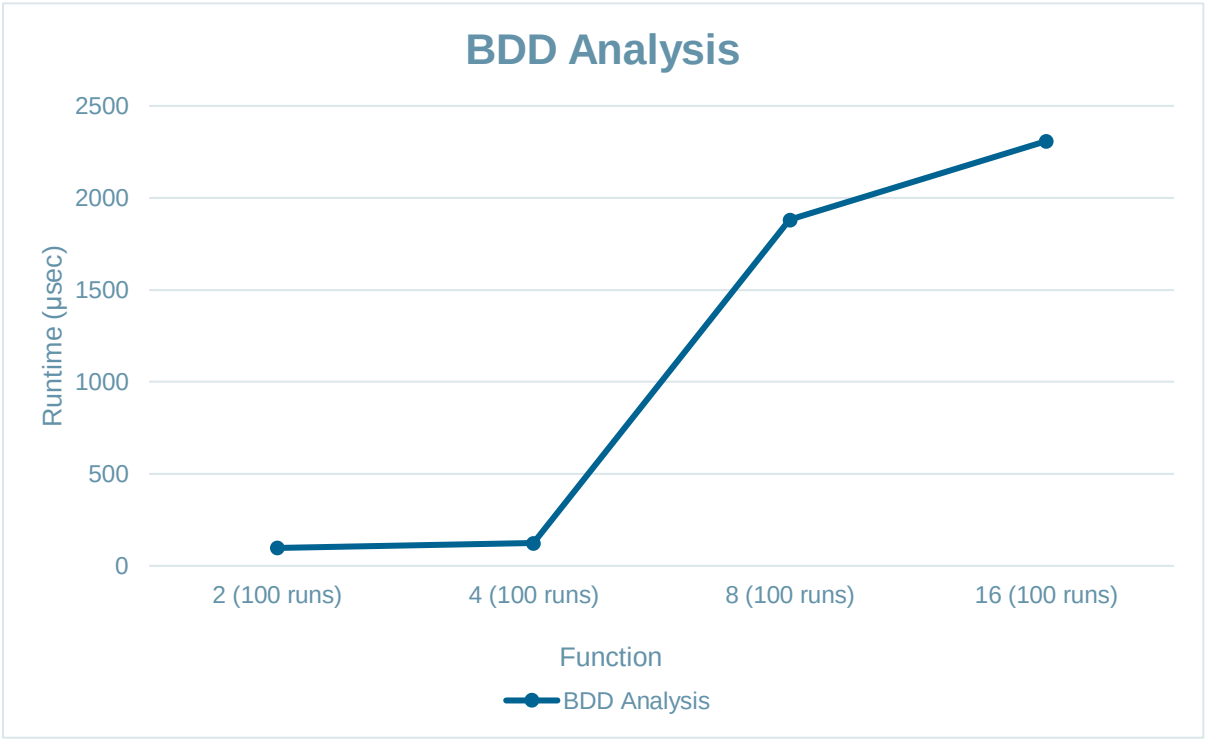
Εικόνα 15. Μέσος χρόνος εκτέλεσης της Truth Table για τυχαίες εισόδους διαφόρων μεγεθών.



Εικόνα 16. Μέσοι χρόνοι εκτέλεσης των αναγραφόμενων συναρτήσεων Karnaugh.



Εικόνα 17. Μέσοι χρόνοι εκτέλεσης των συναρτήσεων Quine-McCluskey.



Εικόνα 18. Μέσος χρόνος εκτέλεσης της συνάρτησης BDD Analysis για διάφορα μεγέθη εισόδων.

Στην Εικόνα 18 φαίνεται διαγραμματικά ο μέσος χρόνος εκτέλεσης της συνάρτησης BDD Analysis για 2-4-8-16 τυχαίες εισόδους από 100 τρεξίματα. Παρατηρούμε ότι η συνάρτηση εμφανίζει αύξηση στον χρόνο εκτέλεσης όσο τα δεδομένα που δέχεται μεγαλώνουν, το οποίο είναι αναμενόμενο.

6.2 Συμπεράσματα & Μελλοντικές Προεκτάσεις

Στην παρούσα εργασία παρουσιάστηκε μια νέα βιβλιοθήκη, σκοπός δημιουργίας της οποίας αποτελεί η επίλυση προβλημάτων συνδυαστικής λογικής. Τέτοιου είδους προβλήματα αποτελούν κυρίαρχο μέρος του γνωστικού αντικειμένου της Λογικής / Ψηφιακής Σχεδίασης, το οποίο κατά βάσην ασχολείται με την μελέτη ψηφιακών κυκλωμάτων. Δεδομένου ότι αυτά αποτελούν βασικό μέρος της επιστήμης των υπολογιστών αναλογίζεται κανείς την σημαντικότητα της μελέτης και κατανοήσης του συγκεκριμένου αντικειμένου. Η επίλυση προβλημάτων συνδυαστικής λογικής δίνει την δυνατότητα να ασχοληθεί κανείς με την Λογική Σχεδίαση ενώ σε συνδυασμό με την γλώσσα Python, που επιλέχθηκε για την συγγραφή της βιβλιοθήκης δίνει στον οποιονδήποτε ένα πλεόκτημα στην εκμάθηση προγραμματισμού.

Η Python αποτελεί μια εύκολη στην εκμάθισή της γλώσσα, ενώ η απλότητα χρήσης της και ευκολία εκμάθισής της την καθιστά ιδανική για την ανάπτυξη της ζητούμενης βιβλιοθήκης. Αυτό το γεγονός υποστηρίζεται από το ότι ο χρόνος υλοποίησής της ήταν σύντομος και δεν συναντήθηκαν σημαντικά εμπόδια κατά την συγγραφή της.

Οι δοκιμές που έγιναν ήταν σημαντικές για την διασφάλιση της ομαλής λειτουργίας και ορθότητας των αποτελεσμάτων των συναρτήσεων. Ο χρόνος εκτέλεσης των συναρτήσεων της βιβλιοθήκης είναι ιδανικός με μέσο όρο εκτέλεσης της τάξης ~ μsec. Επίσης, τα περισσότερα προγράμματα που περιέχει δεν είναι μεγάλης πολυπλοκότητας με αποτέλεσμα οι χρόνοι απόκρισης των συναρτήσεων να είναι ιδανικοί χωρίς να υπάρχουν αστάθειες στις εκτελέσεις τους.

Η βιβλιοθήκη δύναται να προσφερθεί δωρεάν σε τρίτους που πιθανόν την χρειαστούν. Έχει δυνατότητα επέκτασης από όποιον το επιθυμεί ώστε να μπορεί να συμπεριλάβει περισσότερα ζητήματα της Λογικής Σχεδίασης. Οι προσθήκες που μπορούν να γίνουν είναι απεριόριστες, και δυνητικά μπορούν να φτάσουν μέχρι σε σημείο ώστε να επιλυθούν όλα τα ζητήματα συνδυαστικής λογικής. Η χρήση της είναι αρκετά εύκολη, ενώ μπορεί να χρησιπομοιηθεί και από άτομα χωρίς βαθείς γνώσεις προγραμματισμού.

Βιβλιογραφία

1. Συλλογικό έργο Παπαοδυσσεύς, Κώστας Έξαρχος, Μιχαήλ Αραμπατζής, Δημήτριος Γιαννόπουλος, Φώτιος. *Λογική σχεδίαση ψηφιακών συστημάτων*. s.l. : Τζιόλα, 2018.

2. Morris Mano, Michael Ciletti. *Ψηφιακά Σχεδίαση*. s.l. : Παπασωτηρίου, 2018.

3. WAKERLY, JOHN F. *ΨΗΦΙΑΚΗ ΣΧΕΔΙΑΣΗ : ΑΡΧΕΣ ΚΑΙ ΠΡΑΚΤΙΚΕΣ* . s.l. : Κλειδάριθμος, 2019.

4. Guttag, John V. *Υπολογισμοί και προγραμματισμός με την Python*. s.l. : Κλειδάριθμος, 2015.

5. [Ηλεκτρονικό] 2. https://el.wikipedia.org/wiki/%CE%86%CE%BB%CE%B3%CE%B5%CE%B2%CF%81%CE%B1_%CE%9C%CF%80%CE%BF%CF%85%CE%BB.

6. [Ηλεκτρονικό] https://el.wikipedia.org/wiki/%CE%9B%CE%BF%CE%B3%CE%B9%CE%BA%CE%AE_%CF%80%CF%8D%CE%BB%CE%B7.

7. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Karnaugh_map.

8. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Quine%E2%80%93McCluskey_algorithm.

9. [Ηλεκτρονικό] https://docplayer.gr/5647964-Arithmitika-systimata-kodikes.html.

10. [Ηλεκτρονικό] http://www.ece.upatras.gr/images/Attachments/biblia/psifiaki_logiki/%CE%A8%CE%B7%CF%86%CE%B9%CE%B1%CE%BA%CE%AE_%CF%83%CF%87%CE%B5%CE%B4%CE%AF%CE%B1%CF%83%CE%B7_%CE%B1%CF%81%CF%87%CE%AD%CF%82_%CE%BA%CE%B1%CE%B9_%CF%80%CF%81%CE%B1%CE%BA%CF%84%CE%B9%CE%BA%CE.

11. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Binary-coded_decimal.

12. [Ηλεκτρονικό] http://www.eng.ucy.ac.cy/mmichael/courses/ECE210/lab_lecture_seq%20%5BCompatibility%20Mode%5D.pdf.

13. [Ηλεκτρονικό] http://arch.icte.uowm.gr/docs/Python_Programming_Full_Book_Dasygenis_Terzidou.pdf.

14. [Ηλεκτρονικό] https://el.wikipedia.org/wiki/Python.

15. [Ηλεκτρονικό] https://pythoninsider.blogspot.com/2021/10/python-3100-is-available.html.

16. [Ηλεκτρονικό] https://trio.dev/blog/companies-using-python.

17. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Pandas_(software).

18. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/NumPy.

19. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/SciPy.

20. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Matplotlib.

21. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/Scikit-learn.

22. [Ηλεκτρονικό] https://en.wikipedia.org/wiki/IPython.

23. [Ηλεκτρονικό] https://www.anaconda.com/.

24. [Ηλεκτρονικό] https://pypi.org/project/LogicPy/.

25. [Ηλεκτρονικό] https://pypi.org/project/logic-py/.

26. [Ηλεκτρονικό] https://pypi.org/project/logic-py/.
27. [Ηλεκτρονικό] https://pypi.org/project/truth-table-generator/.
28. [Ηλεκτρονικό] https://pypi.org/project/quine-mccluskey/.
29. https://el.wikipedia.org/wiki/%CE%86%CE%BB%CE%B3%CE%B5%CE%B2%CF%81%CE%B1_%CE%9C%CF%80%CE%BF%CF%85%CE%BB.
30. [Ηλεκτρονικό] https://docplayer.gr/3774644-Syndyastika-kyklomata.html.
31. [Ηλεκτρονικό] https://el.wikipedia.org/wiki/Python.
32. [Ηλεκτρονικό] https://el.wikitechpro.com/571109-combinatory-logic-OYTJBS#In_computing.