ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

ΣΧΕΔΙΑΣΜΟΣ ΠΡΑΓΜΑΤΙΚΗΣ ΓΕΝΝΗΤΡΙΑΣ ΤΥΧΑΙΩΝ ΑΡΙΘΜΩΝ

Διπλωματική Εργασία

ΧΡΥΣΟΣΤΟΜΟΣ ΚΕΦΑΛΟΠΟΥΛΟΣ

Επιβλέπων: ΓΕΩΡΓΙΟΣ ΣΤΑΜΟΥΛΗΣ

Σεπτέμβριος 2022

ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΠΟΛΥΤΕΧΝΙΚΗ ΣΧΟΛΗ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

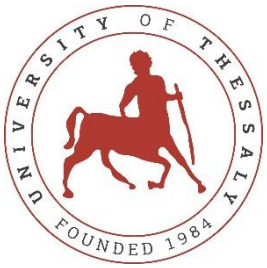
ΣΧΕΔΙΑΣΜΟΣ ΠΡΑΓΜΑΤΙΚΗΣ ΓΕΝΝΗΤΡΙΑΣ ΤΥΧΑΙΩΝ ΑΡΙΘΜΩΝ

Διπλωματική Εργασία

ΧΡΥΣΟΣΤΟΜΟΣ ΚΕΦΑΛΟΠΟΥΛΟΣ

Επιβλέπων: ΓΕΩΡΓΙΟΣ ΣΤΑΜΟΥΛΗΣ

Σεπτέμβριος 2022



UNIVERSITY OF THESSALY

SCHOOL OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

DESIGN OF TRUE RANDOM NUMBER GENERATOR

Diploma Thesis

CHRYSOSTOMOS KEFALOPOULOS

Supervisor: GEORGIOS STAMOULIS

September 2022

Εγκρίνεται από την Επιτροπή Εξέτασης:

Επιβλέπων	**Γεώργιος Σταμούλης** Καθηγητής, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας
Μέλος	**Νέστωρ Ευμορφόπουλος** Αναπληρωτής Καθηγητής, Τμήμα Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών, Πανεπιστήμιο Θεσσαλίας
Μέλος	**Αντώνιος Δαδαλιάρης** Π.Δ 407, Τμήμα Πληροφορικής και Τηλεπικοινωνιών, Πανεπιστήμιο Θεσσαλίας

ΥΠΕΥΘΥΝΗ ΔΗΛΩΣΗ ΠΕΡΙ ΑΚΑΔΗΜΑΪΚΗΣ ΔΕΟΝΤΟΛΟΓΙΑΣ ΚΑΙ ΠΝΕΥΜΑΤΙΚΩΝ ΔΙΚΑΙΩΜΑΤΩΝ

Με πλήρη επίγνωση των συνεπειών του νόμου περί πνευματικών δικαιωμάτων, δηλώνω ρητά ότι η παρούσα διπλωματική εργασία, καθώς και τα ηλεκτρονικά αρχεία και πηγαίοι κώδικες που αναπτύχθηκαν ή τροποποιήθηκαν στα πλαίσια αυτής της εργασίας, αποτελούν αποκλειστικά προϊόν προσωπικής μου εργασίας, δεν προσβάλλουν οποιασδήποτε μορφής δικαιώματα διανοητικής ιδιοκτησίας, προσωπικότητας και προσωπικών δεδομένων τρίτων, δεν περιέχουν έργα/εισφορές τρίτων για τα οποία απαιτείται άδεια των δημιουργών/δικαιούχων και δεν είναι προϊόν μερικής ή ολικής αντιγραφής, οι πηγές δε που χρησιμοποιήθηκαν περιορίζονται στις βιβλιογραφικές αναφορές και μόνον και πληρούν τους κανόνες της επιστημονικής παράθεσης. Τα σημεία όπου έχω χρησιμοποιήσει ιδέες, κείμενο, αρχεία ή/και πηγές άλλων συγγραφέων αναφέρονται ευδιάκριτα στο κείμενο με την κατάλληλη παραπομπή και η σχετική αναφορά περιλαμβάνεται στο τμήμα των βιβλιογραφικών αναφορών με πλήρη περιγραφή. Δηλώνω επίσης ότι τα αποτελέσματα της εργασίας δεν έχουν χρησιμοποιηθεί για την απόκτηση άλλου πτυχίου. Αναλαμβάνω πλήρως, ατομικά και προσωπικά, όλες τις νομικές και διοικητικές συνέπειες που δύναται να προκύψουν στην περίπτωση κατά την οποία αποδειχθεί, διαχρονικά, ότι η εργασία αυτή ή τμήμα της δεν μου ανήκει διότι είναι προϊόν λογοκλοπής.

Ο Δηλών

Χρυσόστομος Κεφαλόπουλος

DISCLAIMER ON ACADEMIC ETHICS AND INTELLECTUAL PROPERTY RIGHTS

Being fully aware of the implications of copyright laws, I expressly state that this diploma thesis, as well as the electronic files and source codes developed or modified in the course of this thesis, are solely the product of my personal work and do not infringe any rights of intellectual property, personality and personal data of third parties, do not contain work / contributions of third parties for which the permission of the authors / beneficiaries is required and are not a product of partial or complete plagiarism, while the sources used are limited to the bibliographic references only and meet the rules of scientific citing. The points where I have used ideas, text, files and / or sources of other authors are clearly mentioned in the text with the appropriate citation and the relevant complete reference is included in the bibliographic references section. I also declare that the results of the work have not been used to obtain another degree. I fully, individually and personally undertake all legal and administrative consequences that may arise in the event that it is proven, in the course of time, that this thesis or part of it does not belong to me because it is a product of plagiarism.

The Declarant

Chrisostomos Kefalopoulos

Ευχαριστίες

Για την ολοκλήρωση αυτής της διπλωματικής εργασίας και της πορείας μου στον πανεπιστημιακό χώρο, συνέλαβαν αρκετά άτομα τα οποία οφείλω να ευχαριστήσω. Αρχικά, τους καθηγητές του πανεπιστημίου Θεσσαλίας και κυρίως τον επιβλέποντα καθηγητή Σταμούλη Γεώργιο για τις επισημάνσεις του και την υποστήριξη του για την περάτωση αυτής της εργασίας . Τέλος , ένα ευχαριστώ στην οικογένεια μου για την αξία της γνώσης που μου υιοθέτησαν και την αμέριστη συμπαράσταση τους καθ' όλη την διάρκεια των σπουδών μου.

Περίληψη

Ανέκαθεν, οι τυχαίοι αριθμοί συνείσφεραν στην ζωή των ανθρώπων σε τύχερα παιχνίδια και αποφάσεις όπου χρειαζόταν εντιμότητα και δίκαιη αντιμετώπιση. Με την πάροδο της εξέλιξης της τεχνολογίας, η ασφάλεια των ηλεκτρονικών συναλλαγών, επικοινωνιών και δεδομένων βασίζεται σε γεννήτριες παραγωγής τυχαίων αριθμών. Τέτοιες γεννήτριες που μπορούν να παράγουν τυχαιότητα σε μεγάλα ποσά αριθμών ,είναι αναγκαίες σε τομείς όπως η κρυπτογραφία, η στατιστική δειγματοληψία, υπολογιστικές προσομοιώσεις και τα τυχερά παιχνίδια. Οι τρόποι παραγωγής είναι αρκετοί και αναλόγως με τον τρόπο παραγωγής μίας γεννήτριας, εκείνη μπορεί να κατηγοριοποιηθεί ως πραγματική ή ως ψευδής. Σκοπός αυτής της διπλωματικής εργασίας είναι να ξεκαθαρίσει τους τρόπους με τους οποίους μπορεί να σχεδιάσει κάποιος μια πραγματική γεννήτρια και να παρουσιάσει τον σχεδιάσμο μίας μέσω ενός κυκλώματος με μια πλακέτα ανοικτού κώδικα. Τέλος, παρουσιάζονται μερικές ιδέες για την εξέλιξη του σχεδίου.

Λέξεις-κλειδιά:

TRNG , PRNG , Θόρυβος από φυσικές διαδικασίες , LFSR , Noise-data whitening , post processing , Testing randomness , Arduino Nano

Abstract

All along, random numbers contributed in the lives of everyday people regarding games of chance and decisions where decency and fairness were necessary. Along with the evolution of technology, the safety of electronic transactions, communications and data is based on random number generators. Such generators that can produce randomness in large amounts of numbers are necessary in sectors such as cryptography, statistic sampling, computational simulations and games of chance. The methods of generation vary and depending on the method of its generation, a generator may be categorized as true random or pseudo random. The purpose of this diploma thesis is to clarify the methods based on which someone can design a true random generator and to present one's design via an open -source circuit board. Finally, some ideas on the further development of the design are presented.

Keywords: TRNG, PRNG, Noise from physical sources, LFSR , noise-data whitening , post processing, Testing randomness , Arduino Nano

Πίνακας Περιεχομένων

Κατάλογος εικόνων

Συντομογραφίες

βλπ βλέπε

κ.λπ. και λοιπά

κ.ο.κ και ούτω καθεξής

TRNG True Random Number Generator

PRNG Pseudo Random Number Generator

QRNG Quantum Random Number Generator

LFSR Linear Feedback Shift Register

Arduino Μονάδα μικροεπεξεργαστή

1 Random Number Generator

1.1 Εισαγωγή

Οι τυχαίοι αριθμοί χρησιμοποιούνται σε πολλούς τομείς της σύγχρονης ζωής. Η πιο απλή χρήση τους βρίσκεται στα τυχερά παιχνίδια ή και ακόμα στην απόφαση ποιος θα ξεκινήσει πρώτος έναν αγώνα. Ιστορικά, οι γεννήτριες τυχαίων αριθμών εφευρέθηκαν πολύ καιρό πριν την ύπαρξη γραπτών αριθμών και μηχανικών συσκευών για τον υπολογισμό τους. Πλέον, τα ηλεκτρονικά συστήματα, δεδομένα και οι τηλεπικοινωνίες παίζουν ένα πολύ μεγάλο ρόλο στην καθημερινότητα μας. Προσωπικές πληροφορίες, νέα και άλλα δεδομένα αποθηκεύονται και μεταφέρονται μέσω διαδικτύου όπου μπορεί να έχει πρόσβαση ο οποιοσδήποτε. Όσο περισσότερο αυξάνεται η χρήση ηλεκτρικών συσκευών, του διαδικτύου και ηλεκτρονικών συναλλάγων, αντίστοιχα τόσο αυξάνονται και οι κακόβουλοι χρήστες που πράττουν ηλεκτρονικά εγκλήματα διατηρώντας την ταυτότητα τους κρυφή. Αυτές οι συμπεριφορές έχουν δημιουργήσει την ανάγκη ανάπτυξης τεχνολογιών και μεθόδων ώστε να μην επιτρέπεται η μορφοποίηση και η πρόσβαση σε δεδομένα από μη εξουσιοδοτημένους χρήστες. Η εξασφάλιση ότι τα ηλεκτρονικά συστήματα και οι πληροφορίες του κάθε χρήστη βρίσκονται υπό προστασία είναι ύψιστης σημασίας, καθώς σε οποιαδήποτε άλλη περίπτωση δεν θα υπήρχε εμπιστοσύνη και λόγος χρήσης μεταξύ του χρήστη και της ηλεκτρονικής μεταφοράς δεδομένων και επικοινωνίας.

Ο τρόπος με τον οποιο επιτυγχάνεται αυτό είναι οι γεννήτριες παραγωγής τυχαίων αριθμών. Η διαδικασία αυτή είναι καθοριστική για την ασφάλεια της ιδιωτικότητας των δεδομένων του καθενός. Κύρια λειτουργία της κρυπτογράφησης είναι η κωδικοποίηση αυτών των δεδομένων ώστε να μην επιτρέπεται η πρόσβαση σε χρήστες που δεν κατέχουν την σωστή διαδικασία αποκωδικοποίησης. Η ικανότητα κρυπτογράφησης του κάθε μηχανισμού εξαρτάται από την τυχαιότητα των δυαδικών αριθμών που υπάρχουν μέσα σε αυτόν. Επομένως οι τυχαίοι αριθμοί που υπάρχουν μέσα σε ένα κρυπτογραφικό σύστημα κρίνουν και την ικανότητα αυτού του συστήματος να παραμένει ασφαλής σε επικείμενες επιθέσεις.

Επίσης πέρα από θέματα κυβερνοασφάλειας, κρυπτογράφησης και εφαρμογές σε τυχερά παιχνίδια, οι γεννήτριες παραγωγής τυχαίων αριθμών έχουν και άλλους τομείς

χρήσης. Τέτοιοι τομείς είναι οι μοντελοποιήσεις, υπολογιστικές προσομοιώσεις και η στατιστική δειγματοληψία.

1.2 Αντικείμενο Διπλωματικής

Ο κύριος στόχος της διπλωματικής αυτής είναι η ανάπτυξη ενός σχεδίου μίας πραγματικής γεννήτριας παραγωγής τυχαίων αριθμών. Επιπλέον αναλύεται το ιστορικό υπόβαθρο παραγωγής αριθμών, ο τρόπος κατηγοριοποίησης γεννητριών ανάλογα με τον τρόπο λειτουργίας τους και διάφορα σχέδια τα οποία αξιοποιούνται μέχρι σήμερα. Συγκεκριμένα, σκοπός της είναι να παρουσιάσει τον τρόπο με τον οποίο η εντροπία που μεταφράζεται ως θόρυβος σε ένα φυσικό σύστημα , συλλέγεται και επεξεργάζεται ώστε να μετατραπεί σε ένα αρχείο τυχαίων αριθμών που αποτελείται από ακολουθίες 0 και 1.

Τέλος η υλοποίηση του φυσικού συστήματος έγινε με την χρήση ενός ηλεκτρονικού κυκλώματος και η συλλογή μαζί με την επεξεργασία του θορύβου με την βοήθεια ενός μικροελεγκτή Arduino.

1.3 Δομή Ενοτήτων

Το περιεχόμενο της διπλωματικής συνοψίζεται ως εξής :

Στο Κεφάλαιο 2 γίνεται μία αναφορά στο ιστορικό υπόβαθρο των γεννητριών και η οι κατηγοροποιήσεις μεταξύ των διάφορων σχεδίων που υλοποιούνται.

Στο Κεφάλαιο 3 αναφέρονται διαφορετικά σχέδια και ο τρόπος με τον οποίο λειτουργεί το καθένα με σκοπό την συλλογή θορύβου.

Στο Κεφάλαιο 4 γίνεται η ανάλυση της λειτουργίας της γεννήτριας και των υλικών που χρησιμοποιήθηκαν για την κατασκευή της.

Στο Κεφάλαιο 5 παρουσιάζονται τα αποτελέσματα της γεννήτριας αφού έχουν περαστεί από δοκιμές που ελέγχουν την τυχαιότητα.

Στο Κεφάλαιο 6 αναλύονται οι τεχνικές λεπτομέρειες για τον προγραμματισμό του μικροελεγκτή και την χρήση λογισμικού του.

Στο Κεφάλαιο 7 αναφέρονται τα συμπεράσματα της διπλωματικής και ορισμένες μελλοντικές ιδέες για την βελτίωση του σχεδίου.

2 Ιστορικό Υπόβαθρο Γεννητριών Τυχαίων Αριθμών

2.1 Πρώιμη ανάγκη

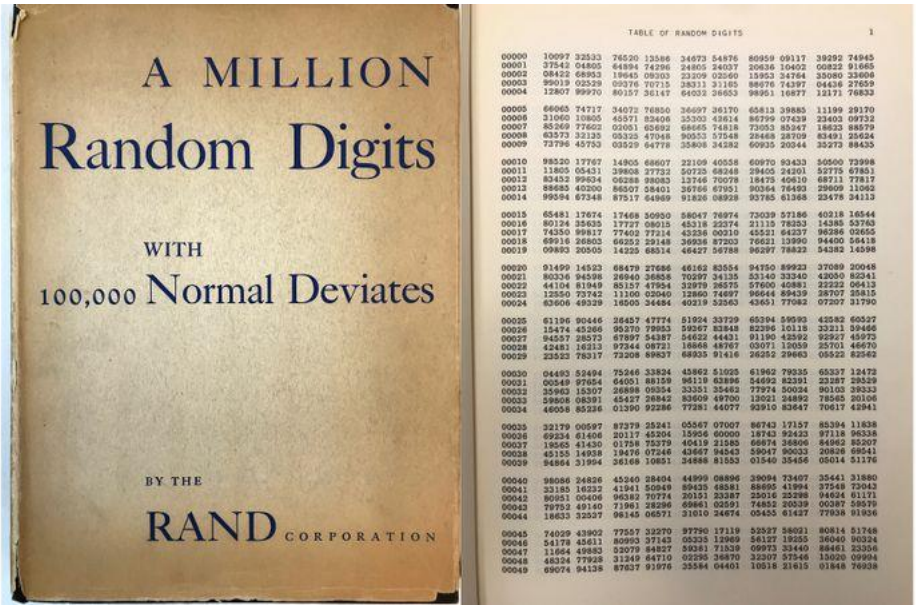
Αρχικά η ανάγκη για τυχαίους αριθμούς δημιουργήθηκε για την απόφαση μιας κατάστασης η οποία είχε δύο ή και περισσότερες πιθανές εκβάσεις. Από τους πιο απλούς και παλιούς μηχανισμούς παραγώγης τυχαίων αριθμών ήταν η αναστροφή ενός νομίσματος. Ωστόσο αυτή η τεχνική ήταν αξιοκρατική και χρήσιμη μόνο για καταστάσεις με δύο περιπτώσεις. Ο πιο γνωστός μηχανισμός που υπήρχε πολύ πριν από την γραφή αριθμών είναι το τίναγμα των ζαριών όπου και αύξησε την φήμη των τυχερών παιχνιδιών. Με το τίναγμα ενός κυβικού ζαριού μπορούσε κάποιος να τύχει μεταξύ έξι διαφορετικών αποτελεσμάτων. Το άθροισμα αυτών των αποτελεσμάτων μπόρει να τροποποιηθεί ή με την αλλαγή του σχήματος του ζαριού ή και με το τίναγμα πολλαπλών.

Με την πάροδο των χρόνων και λίγο πριν από την εφεύρεση των υπολογιστών, επιστήμονες της στατιστικής χρειαζόντουσαν μεγάλες ακολουθίες τυχαίων αριθμών για τυχαία δειγματοληψία και πειράματα τυχαιότητας. Έτσι, πολλοί από αυτούς δημιούργησαν δικούς τους πίνακες από τυχαίους αριθμούς που είχαν πάρει από δειγματοληψίες εκείνης της εποχής. Ένα τέτοιο παράδειγμα είναι ο L.H.C. Tippett [1] ο οποίος το 1927 πήρε τα στοιχεία μιας απογραφής του 1925 και κατασκεύασε έναν πίνακα με 46.100 αριθμούς, τους οποίους αποκαλούσε τυχαίους. Παρόμοιοι πίνακες και εκθέσεις με τυχαίους αριθμούς δημιουργήθηκαν από στατικολόγους της εποχής, αλλά αποδείχτηκε ότι οι αριθμοί δεν ήταν πραγματικά τυχαίοι καθώς ορισμένα ψηφία και ακολουθίες επαναλαμβανόνταν συχνότερα από άλλες.

Η πρώτη σημαντική απόπειρα κατασκευής μιας γεννήτριας παραγωγής τυχαίων αριθμών έγινε από τους Kendall M.G. και Babington-Smith (1938) [2] . Η συσκευή τους αποτελούταν από έναν δίσκο ο οποίος ήταν διαχωρισμένος σε δέκα ισόχωρους τομείς, όπου και ο κάθε τομέας αναπαριστούσε έναν αριθμό από το μηδέν εώς το εννιά. Ο δίσκος περιστρεφόταν με ταχύτητα περίπου 250 στροφές το λεπτό και υπήρχε μία λάμπα συνδεδεμένη με έναν πυκνωτή σε ένα ανοιχτό κύκλωμα που όταν έκλεινε, έδειχνε για μία στιγμή τον έναν από τους δέκα τομείς. Ένας άνθρωπος ένωνε το κύκλωμα περιοδικά ανά περίπου τρία με τέσσερα δευτερόλεπτα και ένας ακόμη σημείωνε τους αριθμούς που έδειχνε η λάμψη της λάμπας. Επίσης είχαν φτιάξει τέσσερις δοκιμές για τον έλεγχο της τοπικής τυχαιότητας σε πίνακες αριθμών, τονίζοντας ότι ενώ δεν είναι επαρκείς για την απόδειξη

ύπαρξης της τοπικής τυχαιότητας είναι άκρως απαραίτητες και χρήσιμες. Το πρώτο κριτήριο ονομαζόταν δοκιμασία συχνότητας και προαπαιτούσε όλα τα ψηφία που παρουσιάστηκαν στον πίνακα να εμφανίζονται περίπου τις ίδιες φορές. Επόμενη ήταν η σειριακή δοκιμασία όπου μετριέται η συχνότητα μίας ακολουθίας συνεχόμενων ψηφίων. Η τρίτη δοκιμασία λεγόταν πόκερ όπου υπολογιζόντουσαν οι συχνότητες ακολουθιών που αποτελούνται από 5 ψηφία και χρειαζόταν να συνάδουν με αντίστοιχες ακολουθίες. Τελευταία ήταν η δοκιμασία κενού όπου υπολογιζόταν η συχνότητα των αριθμών των θέσεων που υπήρχαν ανάμεσα σε διαδοχικές εμφανίσεις οποιουδήποτε ψηφίου. Έπειτα, οι συχνότητες που παρατηρήθηκαν, συγκρίθηκαν με τις αντίστοιχες θεωρητικές μέσω της κατανομής του x^2. Τα αποτελέσματα της γεννήτριας τους πέρασαν τις δοκιμασίες αλλά η μηχανή δεν ήταν πλήρως αυτοματοποιημένη και υπήρχε ανθρώπινη παρέμβαση στην λειτουργία της.

Με την εφεύρεση των ηλεκτρονικών υπολογιστών μπορούσε πλεόν κάποιος να πέρασει τους πίνακες τυχαίων αριθμών στο σύστημα του υπολογιστή του μέσω των λεγόμενων punched cards. Αυτά ήταν κομμάτια χάρτινων καρτών που αναπαριστούσαν ψηφιακές πληροφορίες ανάλογα με τα σημεία που ήταν τρυπημένα. Χρησιμοποιήθηκαν κυρίως για την εισαγωγή, εξαγωγή και αποθήκευση δεδομένων σε υπολογιστικά συστήματα του 20ού αιώνα. Μία ακόμη μηχανή που άλλαξε τον τρόπο με τον οποίο οι άνθρωποι φανταζόντουσαν την παραγωγή τυχαίων αριθμών ήταν υλοποίηση της RAND Corporation στην Σάντα Μόνικα το 1947 [3]. Ο βασικός σχεδιάσμος ήταν μια ηλεκτρική συσκευή που έκπεμπε περίπου 100000 παλμούς το δευτερόλεπτο και κάθε δευτερόλεπτο μετριώνταν ο αριθμός των παλμών. Ύστερα ο αριθμός αυτός υπολογιζόταν modulo 32 και 20 από τις 32 τιμές αποθηκευόνταν σαν δεκαδικά ψηφία. Μετά, αυτά τα ψηφία αναπαραστήθηκαν με τρύπες σε 20000 κάρτες με 50 ψηφία η κάθε κάρτα. Οι στατιστικές δοκιμές στα αποτελέσματα έδειξαν ότι η συχνότητα των περιττών αριθμών ήταν μεγαλύτερη από αυτήν των ζυγών. Για απάντηση σε αυτό, οι επικεφαλείς καθάρισαν τους αριθμούς προσθέτωντας σε κάθε ψηφίο modulo10 του αντίστοιχου ψηφίου της προηγούμενης κάρτας. Τα τελικά αποτελέσματα είχαν περάσει τις δοκιμασίες για τον έλεγχο της τοπικής τυχαιότητας του Kendall και Babington-Smith. Ήταν η πρώτη φορά όπου χρησιμοποιήθηκε ένας υπολογιστής και μια συσκευή που παρήγαγε φυσικό θόρυβο για την παραγωγή ενός πίνακα τυχαίων αριθμών με πλήρως αυτοματοποιημένο τρόπο.



Εικόνα 1: Βιβλίο της εταιρείας Rand με 100000 τυχαίους αριθμούς.

Μετά το πέρας της παραγωγής τους, αυτοί οι αριθμοί ήταν αποθηκευμένοι σε τρυπημένες κάρτες ή περασμένοι σε βιβλία με εκατοντάδες σελίδες το καθένα, όπως φαίνεται στην εικόνα ένα. Επιστήμονες, όπως δειγματολήπτες και φυσικοί που εκτελούσαν προσομοιώσεις και χρησιμοποιούσαν ηλεκτρονικούς υπολογιστές χρειάστηκαν τέτοιους πίνακες τυχαίων αριθμών. Ωστόσο, η ανάγνωση των αριθμών από τρυπημένες κάρτες και άλλων εξωτερικών συσκευών αποθήκευσης καθυστερούσε αρκετά και το μέγεθος της μνήμης των υπολογιστών ήταν πολύ μικρό για να αποθηκεύει τόσο μεγάλο όγκο αριθμών. Άρα , η ανάγκη για την αποθήκευση, την προσπέλαση και την στιγμιαία και γρήγορη δημιουργία τυχαίων αριθμών αυξανόταν όλο και περισσότερο.

Επομένως, οι δύο επικρατέστεροι τρόποι παραγωγής τυχαίων αριθμών στην στιγμή ήταν ή με την χρήση μιας φυσικής συσκευής η οποία θα παρήγαγε θόρυβο και η εντροπία αυτού θα μεταφραζόταν σε τυχαίους αριθμούς, ή με την χρήση ενός ντετερμινιστικού αλγορίθμου ο οποίος θα μιμούταν μια τυχαία ακουλουθία αριθμών, αλλά στην πραγματικότητα θα ήταν αναπαράξιμη. Η πρώτη τεχνική είναι γνωστή ως πραγματική γεννήτρια τυχαίων αριθμών (True Random Number Generator - TRNG) και η δεύτερη με την χρήση αλγορίθμου ως ψευδής γεννήτρια τυχαίων αριθμών (Pseudo Random Number Generator – PRNG).

2.2 TRNG – Πραγματική Γεννήτρια Τυχαίων Αριθμών

Μία πραγματική γεννήτρια τυχαίων αριθμών είναι μία ηλεκτρονική συσκευή η οποία παράγει μια ακολουθία τυχαίων αριθμών, οι οποίοι δεν ακολουθούν κάποια προσχεδιασμένη διάταξη. Η τελική ακουλουθία που παράγει ένα σχέδιο TRNG είναι στην φύση της τυχαία ακόμα και εάν ο σχεδιασμός της γεννήτριας και ο αρχικός σπόρος λειτουργίας είναι γνωστά. Ωστόστο, είναι αναγκαίο ο αρχικός σπόρος που εισάγεται στην συσκευή να είναι τυχαίος από την στιγμή που βασική πτυχή του TRNG είναι η απόλυτη τυχαιότητα και αξιοκρατία. Κύρια λειτουργία αυτών των συσκευών είναι η παρατήρηση και η μέτρηση μιας φυσικής κατάστασης. Αυτές λειτουργούν καταγράφοντας την εντροπία μιας ειδικά σχεδιασμένης και ελεγχόμενης φυσικής διαδικασίας. Πιο συγκεκριμένα, βασίζονται σε μικροσκοπικά φαινόμενα τα οποία παράγουν τυχαία σήματα θορύβου μικρού επιπέδου. Πλεόνασμα τέτοιων σχεδιασμών συσκευών αποτελεί το γεγονός ότι δεν είναι δύσκολη η συλλογή απρόβλεπτης τυχαιότητας μέσα από τον χαώδη κόσμο της φύσης. Τέτοιες πηγές φυσικού θορύβου είναι τα ατμοσφαιρικά σήματα, η θερμική ενέργεια, η απόκλιση της περιόδου του σήματος ενός ρολογιού, η ραδιενεργή αποσύνθεση και ακόμα και τα κβαντικά φαινόμενα.

Μία ακομα ονομασία που έχει δωθεί στις πραγματικές γεννήτριες τυχαίων αριθμών είναι το Hardware Random Number Generator, καθώς βασίζονται κυρίως σε υλικό για την παραγωγή τυχαιότητας και όχι αλγόριθμους και λογισμικό. Επιπλέον χαρακτηριστικό τους αποτέλει ότι είναι μη περιοδικές συσκευές και ότι μία ακουλουθία που έχει παραχθεί δεν μπορεί να επαναληφθεί κατα βούληση ακόμα και εάν ο αρχικός σπόρος είναι ο ίδιος.

2.2.1 Εφαρμογές των TRNG

Η πρωταρχική χρήση γεννητριών παραγωγής τυχαίων αριθμών υπήρξε στα τυχερά παιχνίδια, τον τζόγο, ακόμα και στην αποφάση μεταξύ δύο εκδοχών που απαιτείται αξιοκρατία. Με την εξέλιξη της τεχνολογίας, η ασφάλεια μεταξύ επικοινωνιών, συναλλαγών και μεταφοράς πληροφοριών έγινε απαραίτητη.

Βασικές χρήσεις των πραγματικών γεννητριών τυχαίων αριθμών είναι :

1. Τυχερά παιχνίδια, τζόγος : Χρήση συνήθως σε καζίνο και διαδικτυακά παιχνίδια τζόγου ώστε να αποφευχθεί ο χρήστης και

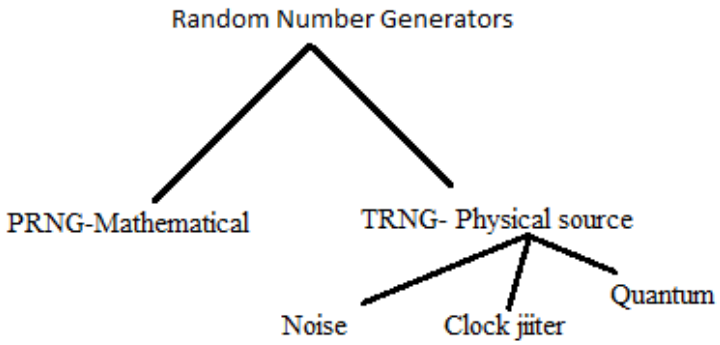
αντίστοιχα ο ιδιοκτήτης να εξαπατήσει το μηχάνημα με σκοπό την αύξηση των κερδών.

2. Κρυπτογραφία : Η ασφάλεια ενός δικτύου ή αρχείου βρίσκεται στην δύναμη του αλγορίθμου του και στην ικανότητα του επιτιθέμενου να αποσπάσει τον σπόρο και τις πληροφορίες του συστήματος.
3. Στατιστική δειγματοληψία : Για την επιλογή μεταξύ δειγμάτων για στατιστικές έρευνες και τα πιθανά αποτελέσματα μεγάλων δημοσκοπήσεων
4. Προσομοιώσεις : Για την μίμηση απρόβλεπτων θορύβων πραγματικού κόσμου που επηρεάζουν την συμπεριφορά ενός συστήματος
5. Chaffing and Winnowing : Είναι μια κρυπτογραφική τεχνική που χρησιμοποιείται για να σταλθούν με ασφάλεια δεδομένα μέσω ενός καναλιού που δεν προστατεύεται, χωρίς να υπάρχει κρυπτογράφηση των πληροφοριών.

2.3 PRNG – Ψευδής Γεννήτρια Τυχαίων Αριθμών

Μία ψευδής γεννήτρια τυχαίων αριθμών είναι συνήθως ένας αλγόριθμος ο οποίος παράγει μία φαινομενικά τυχαία ακολουθία αριθμών. Η ακουλουθία δυαδικών αριθμών που προκύπτει από το λογισμικό είναι μία ντετερμινιστική επεξεργασία του αρχικού σπόρου. Ονομάζεται ψευδής καθώς φαίνεται να είναι τυχαίο το αποτέλεσμα αλλά στην πραγματικότητα δεν είναι ακριβώς απρόβλεπτο. Εάν ο σπόρος γνωρίζεται η ακουλουθία των αριθμών μπορεί να αναπαραχθεί ξανά, σε αντίθεση με το TRNG. Ακόμη, σε αρκετές περιπτώσεις όταν γνωρίζεται η τελική ακουλουθία και ο αρχικός σπόρος είναι δυνατόν να αποκαλυφθεί και ο αλγόριθμος που χρησιμοποιήθηκε, καθώς η ακολουθία που προκύπτει από αυτόν είναι μία σειρά μαθηματικών πράξεων του σπόρου εκκίνησης. Επίσης αυτές οι γεννήτριες είναι περιοδικές, από την στιγμή που η τυχαιότητα τους περιορίζεται από την δυνατότητα του μαθηματικού μοντέλου να παράγει μια ακολουθία από τον αρχικό σπόρο. Η περίοδος του PRNG ορίζεται από το μέγιστο μήκος μη επαναλαμβανόμενων μοτίβων. Η συνήθης μέγιστη περίοδος που μπορεί να έχει ένα PRNG το οποίο στην αρχική του κατάσταση έχει n bits είναι $2^n - 1$. Ωστόσο, συνήθως είναι ακόμα μικρότερη από αυτό, καθώς εξαρτάται από τον αρχικό σπόρο και την υλοποίηση του αλγορίθμου.

Επίσης η φαινομενική τυχαιότητα ενός PRNG μπορεί να αυξηθεί προσθέτωντας περισσότερα bits από όσα χρειάζεται η συγκεκριμένη εφαρμογή που θα χρησιμοποιηθεί. Το πλεονέκτημα των PRNG σε σχέση με τα TRNG είναι ότι δεν απαιτούν υλικό και ειδικά σχεδιασμένο περιβάλλον. Επίσης η λειτουργία τους είναι γρηγορότερη και πιο αποδοτική όσον αναφορά το κόστος, καθώς βασίζονται σε λογισμικό μαθηματικών μοντέλων.



Χαρακτηριστικά	PRNG	TRNG
Μη ντετερμινιστικό	Όχι	Ναι
Απρόβλεπτο	Όχι	Ναι
Αναπαράξιμο	Ναι	Όχι
Περιοδικό	Ναι	Όχι

Εικόνα 2:Χαρακτηριστικά και διαφορές πραγματικών και ψευδών γεννητριών τυχαίων αριθμών.

2.3.1 Εφαρμογές των PRNG

Οι ψευδής γεννήτριες τυχαίων αριθμών δεν είναι οι καταλληλότερες για εφαρμογές που η τυχαιότητα δυαδικών αριθμών είναι το πιο επιθυμητό χαρακτηριστικό, όπως είναι η κρυπτογράφηση και η αποκρυπτογράφηση κλειδιών.

Βασικές χρήσεις γεννητριών τυχαίων αριθμών είναι :

1. Προσομειώσεις και μοντελοποιήσεις, όπου απαιτείται η επανειλλημένη αναπαραγωγή της ίδιας ακουλουθίας
2. Βαθμολόγηση σφαλμάτων, όπου εξετάζεται η συχνότητα ανίχνευσης ενός σφάλματος σε μια γραμμή παραγωγής ή και σε μία συσκευή
3. Στους ηλεκτρονικούς υπολογιστές και την ψηφιακή επικοινωνία του συστήματος τους, όπου είναι αδύνατο μία τέτοια ντετερμινιστική συσκεύη να παράγει τυχαίους αριθμούς
4. Σε ηλεκτρονικά παιχνίδια που δεν είναι κύριο ζητούμενο η ακριβής τυχαιότητα.
5. Monte Carlo εκτιμήσεις, οπού χρησιμοιποιούνται τυχαία δείγματα για να εκτιμηθούν τα πιθανά αποτελέσματα ενός αβέβαιου γεγονότος.

3 Παρόμοιες υλοποιήσεις RNG

3.1 Middle-Squared method – PRNG

Η μέθοδος των μεσσαίων τετραγώνων[4] ήταν από τα αρχικά PRNG και παρουσιάστηκε από τον John von Neumann το 1949. Λόγος δημιουργίας ήταν ότι η ταχύτητα παραγωγής αριθμών αυτής της γεννήτριας ήταν πολύ μεγαλύτερη από το να διαβάζει πραγματικά τυχαίους αριθμούς από τρυπημένες κάρτες, πράγμα το οποίο ήταν πρακτικό για το έργο που είχε με έναν από τους πρώτους υπολογιστές (ENIAC). Για να παράξει κάποιος μία ακουλουθία από n-ψηφία υποτιθέμενα τυχαία, πρέπει να πάρει σαν αρχική τιμή έναν n-ψηφία αριθμό και να τον τετραγωνίσει, ώστε να δημιουργεί ο $2n$-ψηφία αριθμός. Έπειτα, τα μεσσαία n-ψηφία του αποτέλεσματος είναι ο λεγόμενος τυχαίος αριθμός και εάν ήταν επιθυμητό μπορούσε να επαναληφθεί η διαδικασία. Εάν το αποτέλεσμα του τετραγωνισμού έχει λιγότερα από $2n$-ψηφία τότε ακουλουθείται από μηδενικά για αναπληρώσει. Για την μέθοδο αυτήν είναι σημαντικό ο n-ψηφία αριθμός να είναι ζυγός καθώς από το τετράγωνο των περιττών δεν θα υπάρχουν ακριβώς μεσσαία n-ψηφία να διαλεχτούν και θα αναγκαστούν να προστεθούν μηδενικά. Η περίοδος ενός n-ψηφία αριθμού δεν μπορεί να είναι μεγαλύτερη από 8^n.

Πλέον, αυτή η πρακτική θεωρείται κακή καθώς έχει μικρή περίοδο και συνήθως μετά από κάποιους κύκλους αρκέτοι από τους αρχικούς σπόρους καταλήγουν σε μηδέν ή σε προηγούμενο αριθμό της ακολουθίας. Ωστόσο, έθεσε τα θεμέλια και για άλλους παρόμοιους αλγορίθμους.

3.2 Linear Congruential Generator – PRNG

Μία γεννήτρια LCG είναι ένας αλγόριθμος που αποδίδει μια λεγόμενη τυχαία ακουλουθία αριθμών η οποία υπολογίζεται από μία τμηματική γραμμική εξίσωση. Η μέθοδος αυτή είναι από τα πιο παλιά και γνωστά PRNG.

Η γεννήτρια ορίζεται από την εξίσωση :

$$X_{n+1} = (aX_n + c)\ mod\ m \qquad (1)$$

Όπου X είναι η ακολουθία και αποτελείται από :

$m, 0 < m$ – Το modulus

$a, 0 < a < m$ – Ο συντελεστής

c, $0 \leq c < m$ – Η σταθερά

$X_0, 0 \leq X_0 < m$ – Ο αρχικός σπόρος

Όλες οι παραπάνω συνιστώσες δέχονται ακέραιες και σταθερές τιμές. Η εξίσωση λειτουργεί εκτελώντας ένα αναδρομικό μαθηματικό modulo βασισμένο στον αρχικό σπόρο (X_0), τον συντελεστή (a), την σταθερά (c) και το modulo (m).

Η συγκεκριμένη μέθοδος παρουσιάστηκε το 1958 από τους W. E. Thomson and A. Rotenberg οι οποίοι είχαν κάνει τροποποιήσεις στην αρχική μέθοδο του Derrick Lehmer [5]. Η κύρια διαφορά στην εξίσωση μεταξύ των δυο υλοποιήσεων ήταν ότι στην αρχική η σταθερά ήταν μηδενική $(c = 0)$. Σε αυτήν την περίπτωση η γεννήτρια ονομαζόταν Lehmer RNG ή επίσημα MCG (multiplicative congruential generator).

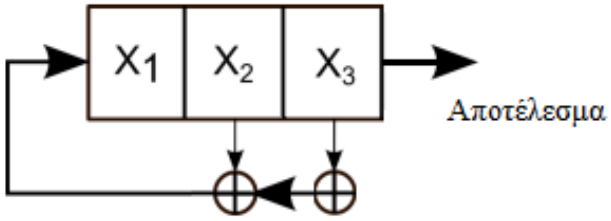
Με την πάροδο των χρόνων υπήρξαν πολλές διαφοροποιήσεις και μοντέλα γεννητριών LCG με σκοπό την επέκταση της περιόδου και την καλύτερη πιστοποίηση από τα τεστ τυχαιότητας της εποχής. Πολλές από τις υλοποιήσεις υποφέρανε από περιοδικότητα και συχνότητα εμφάνισης ακολουθιών. Ένα γνωστό παράδειγμα είναι η RANDU [6] της IBM που χρησιμοποιήθηκε αρκετά τις δεκατίες του 1960 και 1970 και συνεχίστηκε μεχρι και τα τέλη του 1990, ενώ παράλληλα έδινε χαμηλά αποτέλεσματα στο φασματικό τεστ.

3.3 Linear Feedback Shift Register – PRNG

Η LFSR είναι μία από τις πιο γνωστές και εύκολες μεθόδους κατασκευής PRNG. Κυρίως είναι ένα shift register του οποίου το bit εισαγωγής είναι μια γραμμική συνάρτηση του bit της προηγούμενης κατάστασης του.

Η πιο συνηθισμένη γραμμική συνάρτηση που χρησιμοποιείται είναι η exclusive-or (XOR), η οποία εκτελείται μαζί με συγκεκριμένες ολισθήσεις του αρχικού σπόρου. Επομένως, η LFSR συνήθως είναι ένα shift register του οποίου το bit εισαγωγής παράγεται από την XOR κάποιων άλλων bit του συνολικού shift register. Επειδή στην φύση του το PRNG είναι ντετερμινιστικό, η γραμμική συνάρτηση αναδρομής που θα έχει η LFSR είναι πολύ σημαντική ώστε να μην είναι μικρή η περίοδος. Η μέγιστη περίοδος που μπορεί να έχει μια LFSR είναι $2^n - 1$, όπου n είναι ο αριθμός εισαγωγής της συνάρτησης.

Στην εικόνα 3 δίνεται ένα παράδειγμα μίας απλής υλοποίησης με μια ολίσθηση προς τα δεξιά και μια XOR μεταξύ των δύο τελευταίων ψηφίων ώστε να δημιουργηθεί το νέο ψηφίο εισαγωγής.



Επανάληψη	Αποτέλεσμα	Last bit
Σπόρος	110	0
1	111	1
2	011	1
3	001	1
4	100	0
5	010	0
6	101	1
7	110	0

Εικόνα 3: Υλοποίηση απλής LFSR

Η μέγιστη περίοδος που επιτυγχάνεται είναι 7 επαναλήψεις πριν ξανά αρχίσει ο ίδιος κύκλος. Ως last bit έχει ορισθεί το τελευταίο ψηφίο του αποτελέσματος και η ακολουθία 01110010 είναι ο τυχαίος αριθμός της LFSR. Σε πιο περίπλοκα σχέδια αυτής της μεθόδου θα γινόταν να αυξηθούν τα ψηφία του αρχικού σπόρου και να υπήρχαν περισσότερες ολισθήσεις και πράξεις XOR μεταξύ των ψηφίων, ώστε να γίνει πιο πολύπλοκο να προβλεφτεί η ακουλουθία των last bit. Μία περιπλοκότερη υλοποίηση της LFSR χρησιμοποιείται στο σχέδιο αύτης της πτυχιακής εργασίας στα επόμενα κεφάλαια.

3.4 Clock jiiter – TRNG

Ένας ιδανικός ταλαντωτής έχει μια σταθερή περίοδο για την ολοκλήρωση της διαδικασίας του. Ωστόσο, εξαιτίας διάφορων θορύβων , το σήμα του ρολογιού δεν είναι ποτέ ακριβώς σταθερό και οι άκρες του μετακινούνται από την μόνιμη θέση τους. Τέτοιοι θόρυβοι είναι θερμικοί, τροφοδοσίας και παρεμβάσεις θορύβων από διπλανά κυκλώματα. Σε ενα ψηφιακό σύστημα το φαινόμενο της απόκρισης του πραγματικού ρολογιού από το ιδανικό ονομάζεται clock jitter. Ένα ιδανικό ρολόι ορίζεται απο την εξίσωση (1.1), όπου $t(n)$ είναι ο χρόνος της νιοστής περιόδου του ρολογιού και T η περίοδος του.

$$t(n) = n * T \tag{2}$$

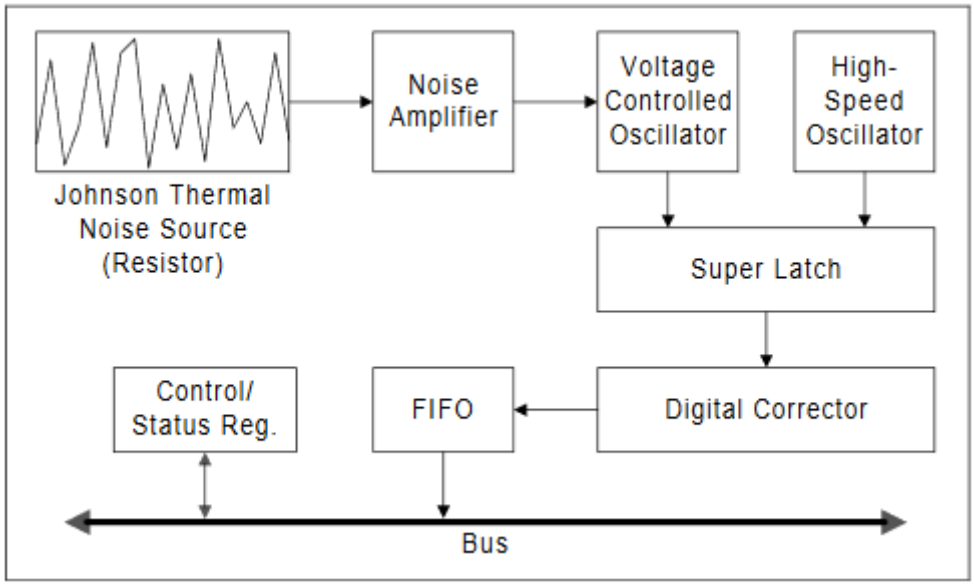
Ωστόσο, ο χρόνος ενός πραγματικού ρολογιού δεν είναι ακέραιο πολλαπλάσιο της περιόδου του, εξαιτίας του clock jitter.

Επίσης phase jitter ονομάζεται η διαφορά του χρόνου της νιοστής περιόδου του πραγματικού ρολογιού με τον χρόνο της αντίστοιχης περιόδου του ιδανικού ρολογιού

$$\delta_{\varphi}(n) = t_r(n) - n * T_{ref} \tag{3}$$

Το jitter ενός ταλαντωτή με μία πύλη ανιστροφέα(inverter) μπορεί να χρησιμοποιηθεί ως πηγή φυσικού θορύβου, αλλά επειδή ο θόρυβος είναι πολύ μικρός απαιτείται η αύξηση του jitter. Η λύση είναι ένας δακτύλιος ταλαντωτής (ring oscillator - free running oscillator) , ο οποίος φτιάχνεται ενώνοντας έναν περιττό αριθμό από inverter σε σχήμα δακτύλιου, όπου το κάθε inverter ανατροφοδοτεί το επόμενο στην σειρά. Το αποτέλεσμα του τελευταίου inverter τροφοδοτείται στο πρώτο inverter έτσι ώστε το τελικό αποτέλεσμα του κύκλου να είναι η λογική πύλη NOT του αρχικού σπόρου, για αυτό και δεν μπορεί να χρησιμοποιηθεί ζυγός αριθμός inverter καθώς το αρχικό και τελικό αποτέλεσμα των inverter θα ήταν το ίδιο δημιουργώντας έναν σταθερό χρόνο περιόδου. TRNG βασισμένα σε ring oscillator jitter είναι εύκολα ως προς την υλοποίηση αλλά αρκετά ευάλωτα σε επιθέσεις υψηλής τάσης και έχουν χαμηλή εντροπία [7].

Ένα ακόμα σχέδιο είναι με την χρήση δύο oscillator με διαφορετικές συχνότητες ο καθένας. Το 1999 η Intel δημιούργησε ένα TRNG το οποίο ήταν ένας δακτύλιος ταλαντωτής που έπαιρνε μετρήσεις θορύβων Johnson-Nyquist[8] που είναι παρών σε όλες τις αντιστάσεις. Στο συγκεκριμένα σχέδιο λαμβάνονται δείγματα θερμότητας στις αντιστάσεις, αυξάνοντας παράλληλα την τάση στο μικροκύκλωμα. Έπειτα υπάρχουν δύο free running oscillators, ένας ο οποίος είναι γρήγορος και ο άλλος αργός. Ο θερμικός θόρυβος χρησιμοποιείται για να προσαρμοστεί η συχνότητα του αργού ρολογιού. Αυτό το θερμικά μεταβαλλόμενο αργό ρολόι χρησιμοποιείται για να προκαλέσει μετρήσεις και να ξεκινήσει το γρήγορο ρολόι. Σε κάθε απρόβλεπτο χτύπημα του αργού ρολογιού, το μικροκύκλωμα λαμβάνει και το αποτέλεσμα του γρήγορου ρολογιού. Τα σήματα και από τους δύο ταλαντωτές τροφοδοτούνται σε έναν ψηφιακό διορθωτή. Η λειτουργία αυτού είναι να εναλλάσει εάν ο συνδυασμός των bit είναι (0,1) να παράγει το αποτέλεσμα 1, εάν είναι (1,0) να παράγει το 0 και τέλος να μην παράγει αποτέλεσμα στην περίπτωση που ο συνδυασμός είναι (0,0) ή (1,1).



Εικόνα 4: Η τεχνική υλοποιήση του αρχικού TRNG της Intel.

3.5 Chaos RNG – TRNG

Τις περισσότερες φορές οι φυσικοί θόρυβοι που επιλέγονται ως πηγές εισόδου για την παραγωγή τυχαίων αριθμών είναι ο θερμικός θόρυβος σε αντιστάσεις και πυκνωτές, και phase jitter των ταλαντωτών. Στην καθημερινότητα ωστόσο υπάρχουν πολλά συμβάντα τα οποία είναι δυνατόν να παρθούν ως φυσικές πηγές θορύβου. Υλοποιήσεις με τέτοιες πηγές χαρακτηρίζονται χαώδη RNG. Εάν συγκριθούν τα χαρακτηριστικά του χάους με τις δυο προηγούμενες μεθόδους ως προς τη μη περιοδικότητα, την τυχαιότητα και το ευρύ φάσμα διακρίνεται ότι οι ιδιότητες του είναι υποσχόμενες για την παραγωγή τυχαίων αριθμών. Είναι σημαντικό σε μία τέτοια υλοποίηση η πηγή της εντροπίας να είναι ένα μη γραμμικό και δυναμικό σύστημα το οποίο λειτουργεί μέσα σε μία μεγαλύτερη χαώδη κατάσταση. Τα συνήθης βήματα λειτουργίας είναι να γίνεται η περισυλλογή της εντροπίας από έναν δειγματολήπτη και έπειτα η εξαγωγή των στοιχείων αυτής που έχουν ατέλειες ώστε να μεγιστοποιηθεί η απόδοση του συστήματος. Τα TRNG που είναι βασισμένα στην χαώδη συμπεριφορά ενός συστήματος χωρίζονται σε δύο κατηγορίες, του συνεχούς και του διακριτού χρόνου. Τα συστήματα συνεχούς χρόνου βασίζονται σε διαφορικές εξισώσεις που είναι συσχετιζόμενες με τον ρυθμό αλλαγής των μεταβλητών του συστήματος στην τρέχουσα κατάσταση. Τέτοια σχέδια συνήθως στηρίζονται σε αναλογικά κυκλώματα και για αυτό απαιτούν μεγάλη κατανάλωση και έκταση. Από την άλλη οι γεννήτριες διακριτού χρόνου βασίζονται σε διαφορικές εξισώσεις που συσχετίζονται μονάχα στην τρέχουσα κατάσταση. Η λειτουργία τους απαιτεί ενα εξωτερικό ρολοί το οποίο ανάλογα την συχνότητα του θα κρίνεται η ταχύτητα παραγωγής της γεννήτριας. Επίσης αυτή η υλοποίηση μπορεί να κατασκευαστεί από ένα ψηφιακό κύκλωμα, οπότε συνήθως έχουν χαμηλή κατανάλωση και δεν χρειάζονται πολλά εξαρτήματα και

Ένα απλό παράδειγμα μίας τέτοιας τεχνικής είναι το LavaRND, σχεδιασμένο στις αρχές του 2000[9]. Είναι ένα Chaos RNG το οποίο εκμεταλλεύεται την συμπεριφορά φωτιστικών λαμπών λάβας. Ειδικότερα, μια φωτογραφία της λάμπας τραβιέται από μία κάμερα και έπειτα ,μέσω ενός CCD μικροκυκλώματος η φωτογραφία ψηφιοποιείται και μετατρέπεται σε 19,200 pixels. Η φωτεινότητα Y και οι τιμές των χρωμάτων UV παράγονται μέσω των pixels και οι τιμές της φωτεινότητας εισάγωνται πρώτα σε έναν αλγόριθμο εξαγωγής εντροπίας ώστε να εξαλειφθούν μη χαοτικά δεδομένα. Ύστερα τα υπόλοιπα εισάγονται σε έναν SHA-1 αλγόριθμο κατακερματισμού για να αυξηθεί περαιτέρω η τυχαιότητα της τελικής ακουλουθίας. Το LavaRND πέρασε το στατιστικό τεστ τυχαιότητας

NIST και παρήγαγε ακουλουθίες bit με ρυθμό 200kb/s. Το σχέδιο αυτό είναι ανοιχτού κώδικα και ο οποιοσδήποτε μπορεί να δοκιμάσει να το κατασκευάσει ή και να πειραματιστεί.

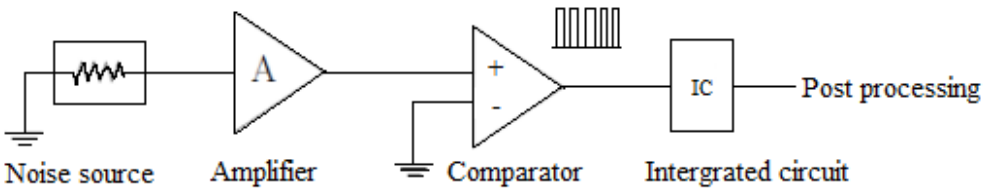
Η χαώδης συμπεριφορά εμπεριέχεται σε ένα φαινομενικά ντετερμινστικό σύστημα. Αυτό το χαώδη σύστημα είναι αρκετά ευαίσθητο στις αρχικές του συνθήκες, κάτι που σημαίνει ότι οι παραμικρές αλλαγές στις αρχικές μεταβλητές θα αλλάξουν τα αποτελέσματα κατά πολύ. Για αυτό και χρειάζεται αρκετή παραμετροποίηση και εξέταση ο σχεδιασμός ενός χαοτικού RNG.

3.6 Noise based – TRNG

Οι πρώτοι θόρυβοι που προσπάθησαν να καταγραφούν και να χρησιμοποιηθούν ως πηγές εντροπίας πέρα από τους τομείς της μηχανικής και της δειγματοληψίας προήλθαν από τα μικροκυκλώματα. Ο θόρυβος Johnson-Nyquist [8] είναι ηλεκτρονικός θόρυβος ο οποίος δημιουργείται από την θερμική καταπόνηση φορέων ηλεκτρικής ενέργειας(ως συνήθως τα ηλεκτρόνια) οταν βρίσκονται μέσα σε έναν ηλεκτρικό αγωγό στον οποίο περνάει τάση. Πιο συγκεκριμένα από την κίνηση των ηλεκτρονίων , η τάση στους ακροδέκτες μίας αντίστασης ξεκινάει να κυμαίνεται. Με την σταδιακή αύξηση της τάσης και την σύγκριση της με μία άλλη σταθερή τιμή αυτής επιτυγχάνεται η παραγωγή τυχαίων bit.

Ένας ακόμα τρόπος παραγωγής θορύβου είναι με την χρήση reversed biased διόδων Zener[10]. Αυτή η δίοδος είναι συνδεδεμένη ανάστροφα με την φορά του ρεύματος και σχεδιασμένη να επιτρέπει στο ρεύμα ενός κυκλώματος να κυλάει μόνο εάν η τάση του ρεύματος είναι μεγαλύτερη από αυτήν της διόδου. Το Zener φαινόμενο συμβαίνει σε διόδους που υπάρχει μεγαλύτερη συγκέντρωση φορέων ρεύματος και μικρή περιοχή εξάντλησης ανάμεσα στο n και p τομέα μίας διόδου, γεγονός που οδηγεί σε ένα ηλεκτρικό πεδίο υψηλής τάσης κατά μήκος της διασταύρωσης των τομέων. Η κάθε διόδος έχει μια προκαθορισμένη τάση κατάρρευσης συνήθως κοντά στα 5-6volt. Τα ισχυρά ηλεκτρικά πεδία επιτρέπουν την διοχέτευση ηλεκτρονίων κατά μήκος της περιοχής εξάντλησης ενός ημιαγωγού. Αυτό οδηγεί σε πολυάριθμους φορείς ελεύθερου φορτίου και με αυτήν την δημιουργία αυτών φορέων αυξάνεται το αντίστροφο ρεύμα. Οι κορυφές ρεύματος που παρουσιάζονται στο μήκος της διόδου είναι ένας πραγματικά τυχαίος θόρυβος. Ωστόσο απαιτείται ύστερα την συλλογή του να ενισχυθεί και επεξεργαστεί ώστε να απαλειφθούν

δεδομένα που μειώνουν την εντροπία και τέλος να μετατραπεί σε μία ακουλουθία από bit. Το φαινόμενο αυτό συμβαίνει συνήθως σε μία αντίστροφη πολωμένη δίοδο όταν η υπάρχουσα τάση είναι κάτω τον 5volt. Αντίστοιχο είναι και το φαινόμενο της φαινόμενο της χιονοστιβάδας στις διόδους Zener το οποίο συμβαίνει πάνω από τα 5volt και σε τομείς διόδων που έχουν μικρή συγκέντρωση φορέων ρεύματος και μεγάλη περιοχή εξάντλησης ανάμεσα στους τομείς.



Εικόνα 5: Αναπαράσταση μίας γεννήτριας βασισμένης σε θόρυβο κυκλώματος όπου ο θόρυβος ενισχύεται και έπειτα περνάει από έναν συγκριτή για να δώσει αποτέλεσμα 0 ή 1.

Παρόμοιο σχέδιο υπάρχει με την ανάστροφη πόλωση ενός διπολικού τρανζίστορ (base – emitter – collector) , όπου δεν ρέει ρεύμα. Όμως, εάν ξεπεραστεί η τάση κατάρρευσης (breakdown voltage) χωρίς να κινδυνεύει η ακεραιότητα των τρανζίστορ, ξεκινάει η παραγωγή τυχαίου αναλογικού θορύβου. Αυτή η τεχνική χρησιμοποιείται σε αυτήν την διπλωματική εργασία και θα αναλυθεί στα επόμενα κεφάλαια.

3.7 Radioactive Decay – Quantum RNG

Οι γεννήτριες που βαζίσονται σε πηγές οι οποίες δέχονται τον θόρυβο τους από την κβαντική μηχανική ονομάζονται QRNG. Τέτοια παραδείγματα είναι η ραδιενεργή αποσύνθεση υλικών[11] και πολλές πρακτικές οπτικής του φωτός, όπως την διαχωροποίηση ενός μοναδικού φωτονίου με την βοήθεια ενός beam-splitter[12] ή και τον χρόνο αφιξης φωτονίων.

Οι περισσότερες υλοποιήσεις τέτοιων σχεδίων έχουν την ίδια βασική ιδέα όπου ένα ραδιενεργό υλικό(Cobalt-60, Strontium-90, Cesium-137) εκπέμπει ιονίζουσα ραδιενέργεια και αυτή εντοπίζεται ή από έναν ημιαγωγικό αισθητήρα ή από έναν μετρητή Geiger-M̈üller.

Ο μετρητής αυτός αποτελείται από έναν γεμισμένο με αέρια θάλαμο και δύο ηλεκτρόδια με μεγάλη διαφόρα στην φόρτιση τους. Όταν ένα ιονίζων δείγμα εισέρχεται στον σωλήνα , δημιουργούνται ιόνια και ηλεκτρόνια τα οποία επιταγχύνουν στις πλευρές του σωλήνα. Ετσι τα ηλεκτρόνια αποκτούν αρκετή κινητική ενέργεια ώστε να ιονίσουν και άλλα άτομα αερίου δημιουργόντας μια αλυσιδωτή αντίδραση η οποία μπορεί να μετρηθεί ως τάση εξωτερικά του σωλήνα. Επομένως στον αισθητήρα υπάρχουν κάποιες σταθερές ενδειξής διαφοροποίησης του ιονίζων υλικού. Ο χρόνος μεταξύ αυτών των ραδιενεργών ενδείξεων είναι απρόβλεπτος. Μία διαδικτυακή υλοποίηση RNG ενός σέρβερ που ονομάζεται Hotbits [13] και είναι βασισμένη στην αποσύνθεση του Cesium-137 , συγκρίνει τα χρονικά διαστήματα μεταξύ τριών συνεχόμενων ενδείξεων ηλεκτρονίων. Ύστερα το τυχαίο bit προκύπτει από πιο χρονικό περιθώριο ήταν το μεγαλύτερο. Συνήθως τέτοιες πρακτικές είναι αργές σε σχέση με τα noise-based RNGs για αυτό και συνδέονται πολλαπλά συστήματα ώστε να μεγαλώσει η ποσότητα αποδοχής bit. Στο διαδίκτυο υπάρχουν σχετικές οδηγίες για κάποιον ο οποίος θελει να αναπαραστήσει αυτό το σχέδιο[13].

Εικόνα 6: Αριστέρα βρίσκεται το πρωτότυπο της Hotbit και δεξιά το ολοκληρωμένο μοντέλο.

4 Σχέδιο Πραγματικής Γεννήτριας Τυχαίων Αριθμών

4.1 Υλικό Συστήματος

Σε αυτό το κεφάλαιο θα αναλυθούν όλα τα υλικά που χρησιμοποιήθηκαν για την κατασκευή της γεννήτριας και η υλοποίηση της.

4.1.1 Μικροελεγκτής Arduino

Το 2005 ξεκίνησε ένα σχέδιο δημιουργίας μίας συσκευής για τον έλεγχο προγραμμάτων διαδραστικών σχεδίων από μαθητές με σκοπό μία πιο φθηνή υλοποίηση από τα άλλα πρωτότυπα συστήματα τα οποία ήταν διαθέσιμα εκείνη την περίοδο. Έτσι θα ήταν πιο εύκολο για φοιτήτες αλλά και χομπίστες να ασχοληθούν με την κατασκεύη και τον προγραμματισμό ηλεκτρονικών κυκλωμάτων.

Το Arduino είναι μία μονή πλακέτα ανοικτού κώδικα με ενσωματωμένο μικροελεγκτή και εισόδους και εξόδους για την επεξεργασία σήματος. Η πλακέτα αυτή μπορεί να προγραμματιστεί μέσω της γλώσσας Wiring (η οποία είναι η γλώσσα προγραμματισμού C++ και από ένα σύνολο απο βιβλιοθήκες που είναι και αύτες υλοποιημένες στην C++). Η σύνδεση μέσω υπολογιστή είναι εφικτή μέσω προγραμμάτων se Processing, Max/MSP, Pure Data και SuperCollider. Κύριο πλεονέκτημα του είναι ότι μπορεί να χρησιμοποιηθεί για την ανάπτυξη ανεξάρτητων διαδραστικών αντικειμένων και την λήψη δεδομένων από μία πληθώρα αισθητήρων έχοντας παράλληλα την δυνατότητα να ελέγχει ένα ολόκληρο κύκλωμα που αποτελείται από διακόπτες, αισθητήρες, φωτισμό, κινητήρες και άλλες φυσικές εξόδους. Το λογισμικό το οποίο εκτελείται στον υπολογιστή, και μέσω αυτού επιτρέπεται πρώτα η εγγραφή και μετά η αποστολή του κώδικα στην πλακέτα Arduino ονομάζεται ολοκληρωμένο περιβάλλον ανάπτυξης (IDE). Τα προγράμματα όπως προαναφέρθηκε είναι γραμμένα σε C++ ή και λίγες περιπτώσεις C. Το IDE παρέχει βιβλιοθήκες με την επεξήγηση μικρών σχεδίων τα οποία μπορούν να φανούν χρήσιμα στον χρήστη σε μία μεγαλύτερη υλοποίηση. Ο τρόπος λειτουργίας της πλακέτας

όταν έχει εγγραφεί ο κώδικας σε αυτήν είναι μέσω ενα προγράμματος κυκλικής εκτέλεσης όπου ο χρήστης πρέπει να ορίσει δύο μόνο λειτουργίες:

-setup(): Μια συνάρτηση που τρέχει μόνο μια φορά με σκοπό να αρχικοποιεί τις ρυθμίσεις.

-loop(): Μία συνάρτηση που καλείται αστάματητα μέχρι να απενεργοποιηθεί η πλακέτα ή να οριστεί από τον χρήστη.

Οι πλακέτες Arduino επιτρέπουν την χρήση τεχνολογίας shields(ασπίδας), οι οποίες είναι πλακέτες επεκτάσιμων κυκλωμάτων που συνδέονται στα pins της πλακέτας Arduino. Οι επεκτάσεις αυτές συνήθως προσθέτονται όταν το μοντέλο της υπάρχουσας πλακέτας του χρήστη δεν έχει κάποια δυνατότητα που του είναι απαραίτητη. Τέτοια παραδείγματα είναι η σύνδεση σε ένα δίκτυο μέσω Ethernet και Wifi, η οδήγηση ενός κινητήρα ή και η προσθήκη οθόνης πάνω στην πλακέτα για την παρουσίαση δεδομένων σε ζωντανό χρόνο.

Το πιο σημαντικό στην χρήση Arduino είναι το χαμηλό κόστος της πλακέτας και η εύκολη διεπαφή του χρήστη με το περιβάλλον προγραμματισμού με επεκτάσιμο υλικό για την υλοποίηση πολύπλοκων σχεδίων.

4.1.1.1 Arduino Nano

Στο παρόν σχέδιο χρησιμοποιήθηκε το Arduino Nano το οποίο βασίζεται στον ATmega328P, έναν 8-bit RISC μικροελεγκτή χρονισμένο στα 16MHz. Έχει ενσωματωμένη μνήμη τριών τύπων με αυτές να είναι :

- 2kb SRAM η οποία είναι η ωφέλιμη μνήμη ώστε να αποθηκεύουν τα προγράμματα μεταβλήτες, πίνακες και άλλες συνιστώσες κατά την διάρκεια λειτουργίας. Έχει παρόμοια λειτουργία με την RAM ενός υπολογιστή όπου εάν διακοπεί η παροχή ρεύματος ή γίνει επαναφορά στο Arduino αυτή χάνεται.
- 1kb EEPROM η οποία χρησιμοποιείται για εγγραφή και ανάγνωση δεδομένων από τα προγράμματα κατά την διάρκεια λειτουργίας του Arduino. Είναι η μνήμη όπου ο προγραμματιστής μπορεί να αποθηκεύεσει πληροφορίες για να υπάρχουν μακροπρόθεσμα. Σε αντίθεση με την SRAM όμως, σε απώλεια παροχής ρεύματος ή επαναφοράς του συστήματος δεν χάνει τα περιεχόμενα της.
- 32kb FLASH memory, από τα οποία τα 2kb είναι δεσμευμένα για τον bootloader του Arduino από τον κατασκευαστή του. Ο bootloader είναι η

διαδικασία της εκκίνησης του Arduino και τα υπόλοιπα 30kb μνήμης περιέχουν το σχέδιο προγραμματισμού της εκάστοτε λειτουργίας του μικροελεγκτή. Η μνήμη flash οπως και η EEPROM δεν χάνει τα περιεχόμενα της με απώλεια τροφοδίας ή επαναφοράς με σκοπό να μην χρειάζεται η επανάληψη εγγραφής του προγράμματος του χρηστή στο Arduino.

4.1.1.2 Εισόδοι – Εξόδοι Πλακέτας

Ο μικροελεγκτής ATmega328P υποστηρίζει σειριακή επικοινωνία η οποία προωθείται μέσα από έναν ελεγκτή Serial-over-USB ώστε να υπάρχει σύνδεση με υπολογιστή μέσω USB (type-B mini-USB). Η σύνδεση αυτή λειτουργεί αμφίδρομα με τον υπολογιστή πρώτα να μεταφέρει τα σχέδια προγραμμάτων στο Arduino και αντίστοιχα αυτό να επικοινωνεί με τον υπολογιστή μεταφέροντας τα δεδομένα που παράγει κατά την διάρκεια εκτέλεσης του προγράμματος.

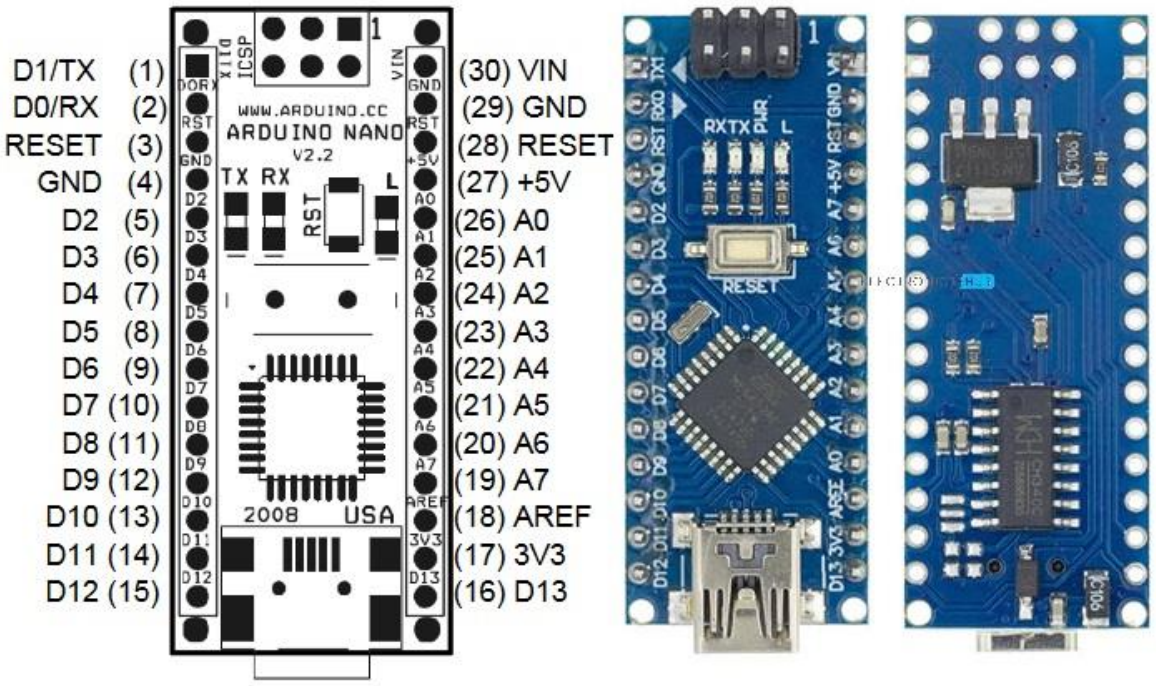
Στο Arduino Nano υπάρχουν συνολικά 30 ακροδέκτες εισόδου και εξόδου.Ο 14 από αυτές είναι οι ψηφιακές υποδοχές ακροδεκτών με την ένδειξη D0-D13, αριθμημένες από το 0 εώς το 13 οι οποίες μπορούν να λειτουργήσουν ως ψηφιακές εισόδοι και εξόδοι. Οι ακροδέκτες αυτοί λειτουργούν με μέγιστη τάση τα 5V και συνεχές ρεύμα τα 40mA. Επίσης μπορούν να ρυθμιστούν ως φηφιακοί εισόδοι μέσω από το πρόγραμμα δηλώνοντας την κατάσταση του ως HIGH(1) ή LOW(0). Αυτό βέβαια εξαρτάται και από το εάν έχει συνδεθεί στον ακροδέκτη συσκευή που του παρέχει σήμα ρεύματος.

Ωστόσο μερικές από αυτές τις 14 υποδοχές έχουν και δεύτερη λειτουργία :

- Οι ψηφιακοί ακροδέκτες 0 και 1 με ένδειξη RX και TX(αλλιώς και DO,D1) λειτουργούν ως μεταφορείς δεδομένων μέσω της σειριακής θυράς. Έτσι, όταν το πρόγραμμα στέλνει δεδομένα στην σειριακή, αυτά προωθούνται και στην θύρα USB μέσω του ελεγκτή Serial-Over-USB αλλά και ενδεχομένως στον ακροδέκτη D0 εάν υπάρχει και αλλή συσκευή για επικοινωνία. Για παράδειγμα μπορούν να ενωθούν με αυτούς τους δύο ακροδέκτες δύο διαφορετικά Arduino ώστε να επικοινωνούν μεταξύ τους. Όταν γίνεται αυτό και στο πρόγραμμα ενεργοποιείται το σειριακό interface, οι θύρες RX TX δεν μπορούν να χρησιμοιηθούν ως εισόδοι και εξόδοι.
- Οι ψηφιακοί ακροδέκτες 2 και 3 μπορούν να λειτουργούν ως εξωτερικοί διακόπτες. Δηλαδή γίνεται να ρυθμιστούν σαν ψηφιακοί εισόδοι στις οποίες εάν παρατηρηθεί κάποια αλλαγή στη ροή του προγράμματος τότε διακόπτεται η κανονική λειτουργία του συστήματος και εκτελείται μία συνάρτηση που έχει οριστεί για την συγκεκριμένη διακοπή.

- Οι ψηφιακοί ακροδέκτες 3, 5, 6, 9, 10, 11 έχουν την δυνατότητα λειτουργίας PWM (Pulse with Modulation) και μπορούν να χρησιμοποιηθούν ως ψευδοαναλογικές εξόδοι.

Από την άλλη πλευρά του Arduino υπάρχουν 8 αναλογικοί ακροδέκτες με την ένδειξη A0-A7, καθένας από αυτούς λειτουργεί ως αναλογική είσοδος με την βοήθεια του ADC (Analog to Digital Converter) που είναι ενσωματωμένος στον μικροελεγκτή. Πιο συγκεκριμένα κάθε μία από τις εισόδους μπορεί να δεχθεί μηδενική τάση εώς μία τάση αναφοράς που είναι ρυθμισμένη στα 5V. Ύστερα, μέσα από το πρόγραμμα μεταφράζεται η τιμή του ακροδέκτη με έναν ακέραιο αριθμό ανάλυσης 10-bit παρέχοντας 1024 διαφορετικές τιμές εισόδου. Με την ένδειξη στα 1023 να δείχνει ότι η τάση στο αντίστοιχο αναλογικό ακροδέκτη είναι 5V.Επίσης δίπλα στο A7 ακροδέκτη βρίσκεται το pin AREF (Analog-REFerence) το οποίο επιτρέπει στον χρήστη να τροφοδοτήσει το Arduino με μια συγκεκριμένη τάση από μία εξωτερική παροχή με σκοπό να προσδιορίζει την τάση αναφοράς που χρησιμοποιείται για αναλογική είσοδο.



Εικόνα 7: Σχέδιο Arduino Nano που χρησιμοποιήθηκε στην υλοποίηση.

4.1.1.3 Τροφοδοσία Μικροελεγκτή

Το Arduino Nano μπορεί να τροφοδοτηθεί με ρεύμα με υπολογιστή μέσω σύνδεσης USB (type-B mini-USB) η μέσω εξωτερικής τροφοδοσίας. Στην πλακέτα υπάρχει ο ακροδέκτης VIN, GND(ground) , +5V ΚΑΙ 3V3. Η ασφαλές τιμές τάσεις για την εξωτερική τροφοδοσία (VIN) κυμαίνονται από 7-12V και μπορεί να παρέχεται από μπαταριές, ένα κοινό μετασχηματιστή ή οποιαδήποτε άλλη πηγή συνεχούς ρεύματος. Η τάση λειτουργίας είναι τα 5V και μέσω του ακροδέκτη +5V και 3V3 μπορεί να υπάρχει παροχή και σε άλλα μέρη του κυκλώματος. Τέλος υπάρχουν και δύο ακροδέκτες RESET οι οποίοι είναι προγραμματιζόμενοι για την επαναφορά του συστήματος.

Η πλακέτα Arduino Nano χρησιμοποιήθηκε σε αυτήν την υλοποίηση καθώς ήταν επαρκής οι δυνατότητες τις για τις διαδικασίες που έπρεπε να εκτελέσει αλλά κύριως επειδή είναι πολύ μικρότερη από τις αντίστοιχες πλακέτες Arduino διαθέτοντας περισσότερο χώρο για το υπόλοιπο κύκλωμα.

4.1.2 Περιφερειακά Τεχνικά Υλικά

Τα περιφερειακά υλικά είναι απαραίτητα για την δημιουργία και την επεξεργασία του τυχαίου θορύβου και ύστερα την μεταφορά αυτού στις υποδοχές του μικροελεγκτή Arduino Nano με σκοπό να είναι επεξεργάσιμος από τον υπολογιστή. Τα υλικά για τον σχεδιασμό και την μελέτη της συγκεκριμένης διπλωματικής εργασίας με στόχο την δημιουργία μιας πραγματικής γεννήτριας τυχαίων αριθμών αναλύονται παρακάτω :

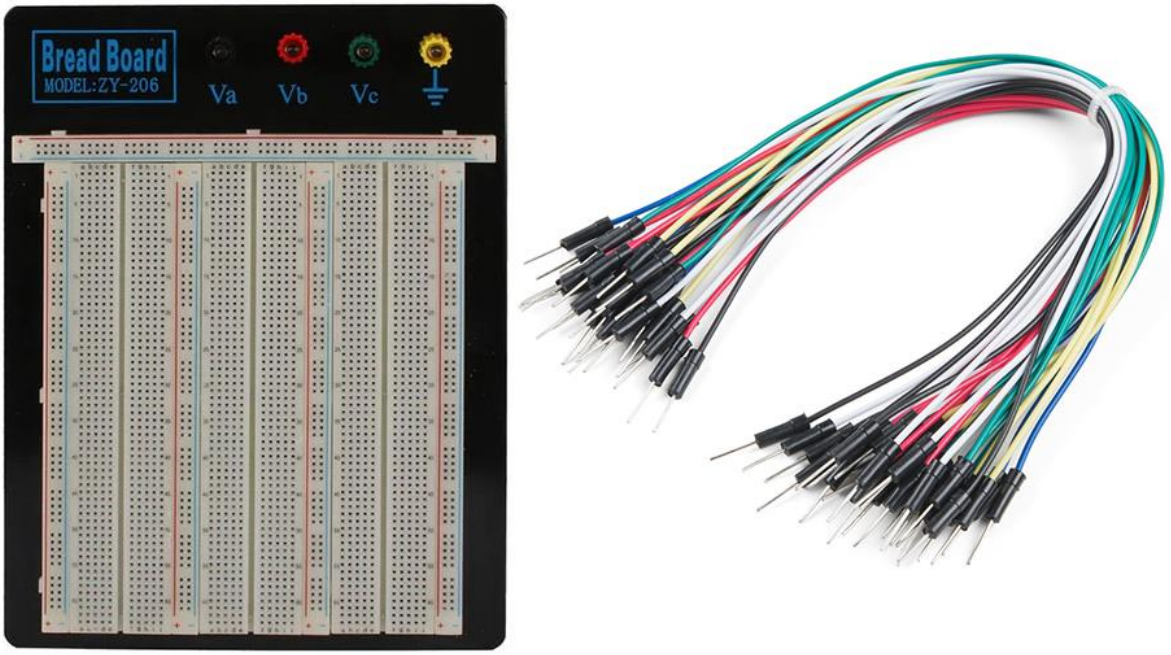
4.1.2.1 Breadboard

Ένα breadboard συχνά αναφερόμενο και ως μη συγκολλημενή πλάκα-πλακέτα ή και πλακέτα πρωτότυπου είναι μια πλακέτα που λειτουργεί ως βάση για την κατασκευή πρωτότυπων ηλεκτρονικών κυκλωμάτων. Κύριο χαρακτηριστικό της πλάκας αυτής είναι ότι δεν απαιτείται να γίνει συγκόλληση ώστε να επικοινωνήσουν τα επιμέρους υλικά που συνδέονται πάνω της. Αυτό επιτυγχάνεται έχοντας σειρές από ράγες χαλκού όπου η κάθε σειρά αποτελείται από 5 υποδοχές που επικοινωνούν μεταξύ τους εάν συνδεθούν. Οι αποστάσεις μεταξύ των υποδοχών είναι στα 2.54mm έτσι ώστε να κουμπώνουν ακριβώς πάνω τους μικροελεγκτές Arduino και ολοκληρωμένα κυκλώματα (IC). Μια πληθώρα

ηλεκτρονικών συστημάτων μπορούν να κατασκευάστουν ως πρωτότυπα για αρχή, από μικρά αναλογικά ή ψηφιακά κυκλώματα ή και ολόκληρες κεντρικές μονάδες επεξεργασίας. Στην εικόνα 8 το αντιστοιχό breadboard που χρησιμοποιήθηκε.

4.1.2.2 Jumper Wires

Αυτά είναι ένα ηλεκτρικό καλώδιο ή αρκέτα μαζί σε ενα καλώδιο τα οποία στους ακροδέκτες τους έχουν ένα κομμάτι σίδερο με σκοπό να ενώνονται στις ράγες των υποδοχών ενός breadboard χωρίς να χρειάζεται συγκόλληση. Κύρια λειτουργία τους είνα η ένωση του breadboard με το Arduino και άλλα υλικά.



Εικόνα 8: Στα αριστερά είναι το breadboard και στα δεξία τα JumperWires.

4.1.2.3 Αντιστάσεις - Trimpot

Οι αντιστάσεις είναι ηλεκτρονικό εξάρτημα το οποίο χρησιμοποιείται σε διάφορα κυκλώματα με σκοπό να περιορίσει την ροή του ρεύματος. Όσο μεγαλύτερη είναι η τιμή της αντίστασης του , τόσο περισσότερο αντιστέκεται και αντίστοιχα τόσο λιγότερο ηλεκτρικό ρεύμα θα ρέει μέσα από αυτό. Η μονάδα μέτρηση της αντίστασης είναι τα Ohm και η το κάθε ποσό αντίστασης χαρακτηρίζεται με κωδικούς χρώματος. Στην παρούσα διπλωματική εργασία έχουν χρησιμοποιηθεί αντιστάσεις των 1kΩ, 10kΩ, 2.2kΩ, 4.7kΩ και 1MΩ.

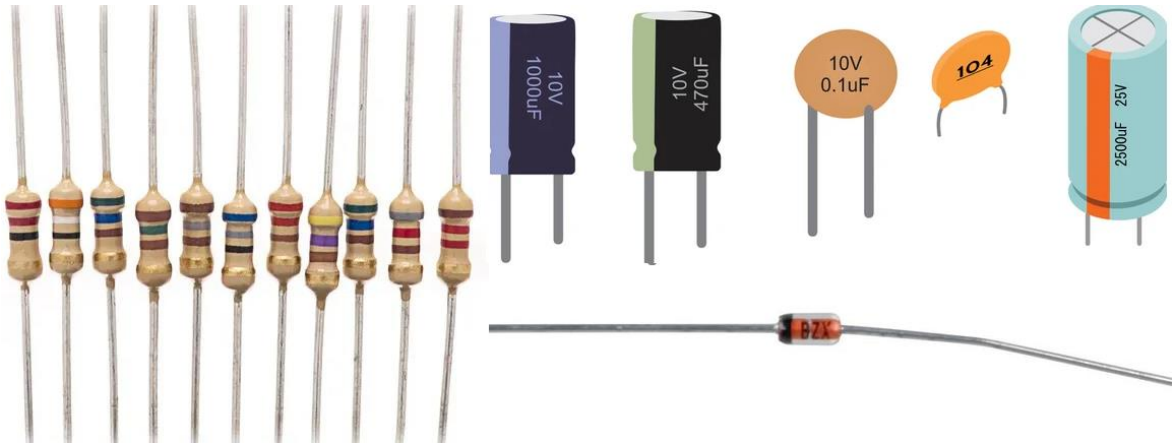
Παρομοίως τα trimpot (ποτενσιόμετρο) είναι μία αντίσταση η οποία έχει μεταβαλλόμενη τιμή και με το γύρισμα της κεφαλής της δεξιόστροφα ανεβαίνει η τιμή της αντίστασης αλλιώς κατεβαίνει. Στην παρούσα διπλωματική εργασία χρησιμοποιείται ένα trimpot τωμ 10kΩ(με μικρότερη τιμή τα 0Ω και μεγαλύτερη τα 10kΩ.

4.1.2.4 Πυκνωτές

Ο πυκνωτής είναι ένα ηλεκτρολογικό παθητικό στοιχείο το οποίο σκοπός του είναι να αποθηκεύει ενέργεια. Αυτή η ενέργεια αποθηκεύεται υπό την μορφή ηλεκτρικού φορτίου στους οπλισμούς του δημιουργώντας ένα ηλεκτρικό πεδίο. Το φαινόμενο να αποθηκεύεται ηλεκτρική ενέργεια υπό την μορφή ηλεκτρικού πεδίου σε ένα παθητικό στοιχείο ονομάζεται χωρητικότητα. Μονάδα μέτρησης της χωρητικότητας είναι το Farad και ισούται με την χωρητικότητα ενός πυκνωτή , ο οποίος έχει διαφορά δυναμικού ενός Volt ανάμεσα στους οπλισμούς του, όταν το ηλεκτρικό φορτίο σε έναν απο αυτούς είναι 1 Coulomb και ο απέναντι οπλισμός έχει ίσο και αντίθετο φορτίο. Κύριος λόγος που χρησιμοποιούνται οι πυκνωτές είναι για να υπάρχει αντίσταση στις μεταβολές τις τάσης και να κρατάει σταθερό το κύκλωμα ώστε να μην υπάρχουν αυξομοιώσεις. Στο σχέδιο της διπλωματικής αυτής χρησιμοποιήθηκαν πυκνωτές των 470μF, 10nF και 100nF.

4.1.2.5 Δίοδος Zener

Η δίοδος Zener είναι μία δίοδος μπορεί να λειτουργεί στην περιοχή κατάρρευσης (breakdown voltage), δηλαδή αντέχει να λειτουργεί υπό φορτίο τάσης στην οποία άλλες διόδοι κινδυνεύουν να καταστραφούν. Κύρια λειτουργία της είναι να κρατάει την τάση στο φορτίο του κυκλώματος σταθερή ανεξάρτητα από μεταβολές στην τάση της γραμμής και στην αντίσταση του φορτίου. Στο σχέδιο της γεννήτριας χρησιμοποιείται μία δίοδος 4.7V.



Εικόνα 9: Στα αριστερά φαίνονται αντιστάσεις διάφορων Ohm, πάνω δεξιά διαφορετικοί πυκνωτές και κάτω δεξιά η δίοδος Zener.

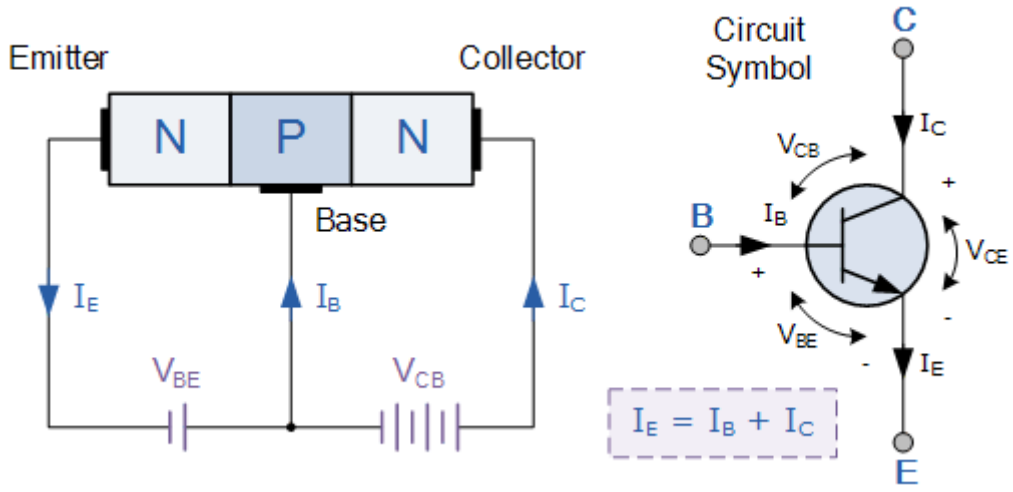
4.1.2.6 Τροφοδοσία

Για την τροφοδοσία από εξωτερική πηγή χρησιμοποιήθηκε ένα κοινός μετασχηματιστής συνεχούς τάσης ο οποίος έχει επιλογή διάφορων τιμών τάσεων. Η τιμή που χρησιμοποιήθηκαν σε αυτήν την διπλωματική βρίσκεται στα 12V, 1.5A και 18W.

4.1.2.7 Τρανζίστορ – Bipolar junction Transistor

Το τρανζίστορ είναι μία διάταξη ημιαγωγών που χρησιμοποιείται για την ενίσχυση και την εναλλαγή ηλεκτρονικών σημάτων ισχύος. Επίσης μπορεί να λειτουργήσει ως σταθεροποιητής τάσης, να διαμορφώσει συχνότητα και ως διακόπτης. Με άλλα λόγια είναι μία αντίσταση της οποίας η τιμή αλλάζει με βάση το σήμα εισόδου και η αλλαγή αντίστασης μπορεί να μετασχηματίζεται. Από εκεί προκύπτει και ο όρος transistor (transfer resistor). Ο όρος διπολικό τρανζίστορ αναφέρεται στην χρήση ηλεκτρονίων αλλά και οπές ηλεκτρονίων ως φορείς ρεύματος. Τα τρανζίστορ αποτελούνται από τρεις περιοχές οι οποίες είναι ο emitter(εκπομπός), base(βάση) και το collector(συλλέκτης). Το emitter είναι συνδεδεμένο στην αριστερή πλευρά του n-type(negative charged) σχεδιού και αντίστοιχα το collector είναι στην δεξιά πλευρά του n-type. Αντιθετώς, το base είναι συνδεδεμένο στο p-type(positive-charged). Επίσης υπάρχουν και δύο n-p τομείς με τον έναν να είναι ανάμεσα στο emitter και στο base και την άλλη στο collector-base. Το transistor που επιλέχθηκε για

την πτυχιακή αυτή έχει κωδικό 2N3904 και η διάταξη του είναι n-p-n. Στην εικόνα 10 φαίνεται η πως η ροή του ηλεκτρικού φορτίου περνάει από το τρανζίστορ.



Εικόνα 10: Σχέδιο αναπαράστασης n-p-n τρανζίστορ.

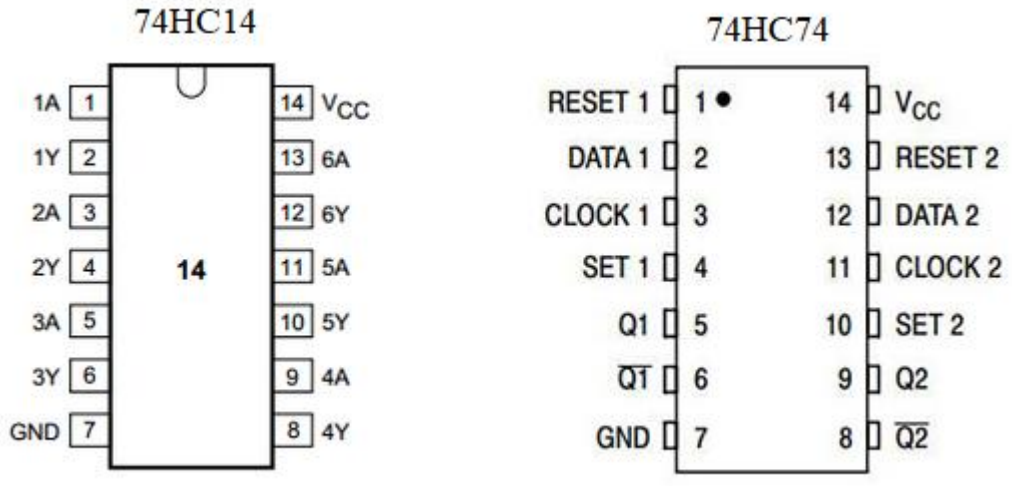
4.1.2.8 Integrated Circuit – Ολοκληρωμένα

Ένα IC είναι ένα ηλεκτρονικό κύκλωμα συνδεδεμένων λογικών πυλών κατασκευασμένο πάνω σε ένα ημιαγώγιμο υλικό όπως το πυρίτιο. Όταν το μέγεθος της κατασκευής είναι της κλίμακας των μικρομέτρων τότε ονομάζεται και μικροτσίπ. Το κάθε μικροτσίπ έχει διαφορετικό σκοπό και είναι βασισμένα στο να δέχονται δυαδικά δεδομένα και να τα επεξεργάζονται σύμφωνα με τις εντολές που έχουν αποθηκευεμένα στην μνήμη τους. Στην παρούσα διπλωματική χρησιμοποιήθηκαν δύο διαφορετικά IC , το ένα με κωδικό 74HC14 και το άλλο 74HC74.

Το ολοκληρωμένο 74HC14 είναι ένα Hex Schmitt – Trigger Inverter. Αποτελείται απο 14 ακροδέκτες των οποίων είναι η Vcc(πηγή τάσης), GND (γείωση) και μετά 6 υποδοχές σήμα εισόδου(1A) οι οποίες με την χρήση της λογικής πύλης NOT καταλήγουν σε 6 σήματα εξόδου(1Y). Ο λόγος χρήσης αυτού του ολοκληρωμένου είναι επειδή εναλλάσει αναλογικά σήματα κυματομορφών σε ψηφιακά σήματα (0, 1). Αυτό επιτυγχάνεται με την σύγκριση της τάσης στο σήμα εισόδου με την χρήση δύο τάσεων αναφοράς +Vt και την -Vt. Για παράδειγμα το σήμα εξόδου θα είναι χαμηλό (0) μόνο όταν στο σήμα εισόδου η τάση είναι μεγαλύτερη της τάσης αναφοράς +Vt. Στην περίπτωση που η τάση στο σήμα εισόδου είναι ανάμεσα στις δύο τάσεις αναφοράς +Vt και την -Vt τότε το

σήμα εξόδου θα παίρνει την αμέσως προηγούμενη τιμη που είχε. Όταν στο σήμα εισόδου η τάση θα πέσει κάτω από την -Vt τότε το σήμα εξόδου θα γίνει υψηλό (1) , μέχρι να ξαναφτάσει η τάση στην εισόδο στην +Vt και να γίνει 0. Με αυτόν τον τρόπο το σήμα μας ψηφοποιείται σε 0 ή 1 αλλά προσθέτεται εξάρτηση από προηγούμενες τιμές γιατί εάν η τασή στην είσοδο παραμένει ανάμεσα στην +Vt και την -Vt τότε υπάρχει αναδρομική σχέση μεταξύ των τιμών. Αυτή είναι η λειτουργία του 74HC14 με μία μεταβαλλόμενη τάση εισόδου.Τεχνικά χαρακτηριστικά υπάρχουν στο datasheet του ολοκλωμένου.

Το ολοκληρωμένο 74HC74 περιέχει δύο ανεξάρτητες D type positive-edge-triggered flip-flop με ασύγχρονο preset. Ο λόγος που χρησιμοποιείται αυτό το ολοκληρωμένο είναι για να απαλείψει την εξάρτηση που δημιουργήθηκε από τα συνεχή 0 ή 1 που προήλθαν από την ψηφιοποίηση του σήματος μέσω του 74HC14. Αυτό γίνεται θέτωντας την flip flop να παράγει το αποτέλεσμα Q μόνο όταν υπάρχει εναλλαγή τιμής στην είσοδο του ολοκληρωμένου. Έτσι η συχνότητα παραγωγής τυχαίων αριθμών περιορίζεται μόνο όταν συμβαίνει εναλλαγή τιμής στην έξοδο του 74HC14, αφήνοντας έξω τις τιμές οι οποίες προήλθαν από το feedback όταν η τάση στη είσοδο του 74HC14 ήταν σταθερή ή ήταν ανάμεσα στην +Vt και την -Vt. Παρακάτω δίνονται τα διαγράμματα των ολοκληρωμένων.



Εικόνα 11: Αριστερά βρίσκεται το σχεδιάγραμμα του 74HC14 και αντίστοιχα δεξιά του 74HC74.

4.2 Σχέδιο Πραγματικής Γεννήτριας Τυχαίων Αριθμών

4.2.1 Avalanche Noise-Breakdown

Το Avalanche breakdown είναι ένα φαινόμενο το οποίο μπορεί να συμβεί σε μονωτικά και ημιαγώγικα υλικά[14]. Στους ημιαγωγούς υπάρχουν δύο είδη φορέων ρεύματος, τα ελεύθερα ηλεκτρόνια και οι ηλεκτρικές οπές. Οι ηλεκτρικές οπές δημιουργούνται σε ένα άτομο ή σύμπλεγμα ατόμων όταν ένα ηλεκτρόνιο ξεφεύγει από τον χώρο αυτών. Εάν στον ημιαγωγό υπάρχει τάση που δημιουργεί ηλεκτρικό πεδίο τότε τα ηλεκτρόνια θα κινούνται με την θετική κατεύθυνση του ρεύματος ενώ οι ηλεκτρικές οπές αντίστροφα. Συνήθως αυτή η διαδικασία είναι απλή και γίνεται μέχρι να γίνει η μεταφόρα στις αντίστοιχες μεριές του ημιαγωγού. Ωστόσο, στην περίπτωση που το ανάστροφο ηλεκτρικό πεδίο είναι τόσο ισχυρό, τα ελεύθερα ηλεκτρόνια και οι ηλεκτρικές οπές αποκτούν αρκετή ταχύτητα ώστε να μπορέσουν να χτυπήσουν άλλους φορείς ρεύματος εκτοπίζοντας τους. Αυτά με την σειρά τους συγκρούονται δημιουργώντας και άλλους φορείς ρεύματος, πράγμα το οποίο αυξάνει το ρεύμα και δημιουργεί αυτήν την αλυσιδωτή διαδικασία. Ο τρόπος σύνδεσης υλικών είναι η με την ανάστροφη χρήση διόδων Zener ή με την ανάστροφή σύνδεση BJT n-p-n (Bipolar Junction Transistor). Ωστόσο πρέπει να δωθεί ιδιαίτερη σημασία στην τιμή της τάσης την οποία υποβάλλεται η συσκευή, στην οποία συμβαίνει το breakdown καθώς αυτή δεν είναι η εργαστασιακή της λειτουργία. Η υψηλή τιμή τάσης και ρεύματος που ρέει την συσκευή καθόλη την διάρκεια του breakdown, συνήθως καταλήγει σε μερική αύξηση της θερμοκρασίας αλλά είναι δυνατόν να καταστρέψει και το τρανζίστορ εάν πιεστεί στα άκρα.

Πιο συγκεκριμένα, οι τομείς των τρανζίστορ που συζητήθηκαν στο προηγούμενο κεφάλαιο(Emitter-base, collector-base) κατασκευάζονται σε μεγέθη που έχουν φορείς ρεύματος(highly-lightly doped), και με μία περιοχή(depletion region) οπού δεν υπάρχουν ελεύθερα ηλεκτρόνια για να μεταφέρουν ρεύμα και το μέγεθος της οποίας μπορεί να είναι αντίστοιχα μεγάλο ή μικρό. Το depletion region λειτουργεί ως ένα φράγμα που αντιτίθεται στην ροή των ηλεκτρονίων απο τον n-τομέα και ηλεκτρικών οπών από τον p-τομέα. Το φαινόμενο Avalanche Breakdown συμβαίνει σε τρανζίστορ[15] με lightly doped p-n τομείς όπου το depletion region είναι μεγάλο και η ανάστροφη τάση ξεπερνάει τα 5V. Το αποτέλεσμα που παράγεται από ένα τέτοιο τρανζιστορ είναι μία απρόβλεπτη ροή ρεύματος (θόρυβος) η οποία μπορεί να χαρακτηριστεί ως πηγή εντροπίας.

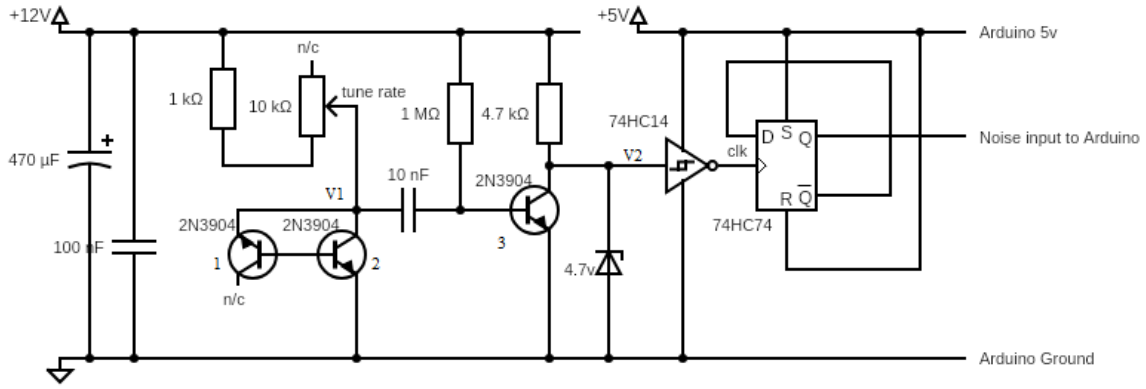
4.2.2 Υλοποίηση Κυκλώματος

Το βασικό σχέδιο του κυκλώματος βρίσκεται στην εικόνα 12. Η τροφοδοσία του κυκλώματος από τα αριστέρα γίνεται μέσω ενός κοινού φορτιστή με τιμή τάσης τα 12V. Στο κύκλωμα έχουν προστεθεί 2 πυκνωτές ένας των 100nF και ένας των 470μF για να αφαιρεθεί ο θόρυβος και πιθανές αυξομοιώσεις τάσης που μπορεί να προκληθούν από το φορτιστή των 12V. Ακριβώς μετά την αντίσταση των 1kΩ η τάση μετριέται στα 11.6V πριν την είσοδο της στο trimpot.

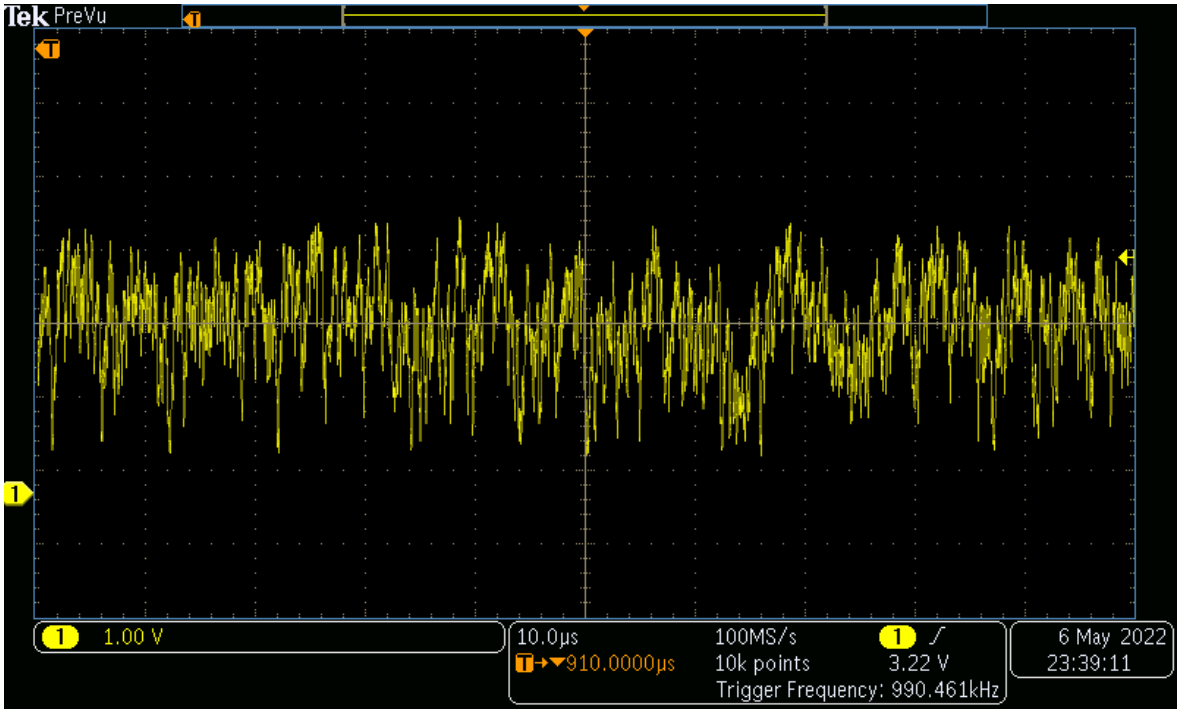
Η χρήση trimpot με τιμές 1kΩ-10kΩ είναι αποδοτική καθώς μπορούμε να ρυθμίσουμε την Vbr (τάση κατάρρευσης) που θα ρέει μέσα από το τρανζίστορ. Το τρανζίστορ που δημιουργεί τον θόρυβο είναι το νούμερο 1. Ο emitter του είναι αντίστροφα πολωμένος χρησιμοποιώντας την τάση που ρέει από το trimpot και ο collector του είναι εκτός του κυκλώματος. Η τιμή τάσης V1 όπως φαίνεται στο κύκλωμα είναι στα 8.7V ώστε να γίνεται η κατάρρευση του φράγματος και δημιουργείται το Avalanche noise. Από το datasheet(εγχειρίδιο πληροφοριών) του τρανζίστορ 2N304 το emitter-base breakdown voltage είναι στα 6v. Εάν η V1 είναι μικρότερη των 6V το κύκλωμα δεν άγει.

Τα τρανζίστορ 2 και 3 λειτουργούν με κύριο σκοπό να ενισχύσουν το σήμα που παράγει το 1 και επειδή είναι ενισχυτές υψηλού αναλογικού σήματος είναι σημαντικό να χρησιμοποιούνται μικρά jumper wires σε μήκος ώστε να αποφευχθεί η συλλογή ηλεκτρομαγνητικών πεδίων. Ο πυκνωτής των 10nF δεν επιτρέπει να αλλοιωθεί ο αναλογικός τομέας του κυκλώματος από διαταραχές του ψηφιακού τομέα. Οι αντιστάσεις των 1MΩ και 4.7kΩ μειώνουν την τάση στην είσοδο και στην έξοδο του τρίτου τρανζίστορ. Η V2 βρίσκεται στα 1.7V.

Στην εγκατάσταση της δίοδου Zener 4.7V η λωρίδα κατεύθυνσης ρεύματος πρέπει να δείχνει από την αντίθετη πλευρά της γείωσης, διαφορετικά από μία κανονική δίοδο. Η δίοδος λειτουργεί ως βαλβίδα που δεν επιτρέπει τιμές τάσεων ανώ των 4.7V να περάσουν στο ολοκληρωμένο 74HC14. Στην εικόνα 13 φαίνεται το τυχαίο και ενισχυμένο σήμα εξόδου από το τρανζίστορ 3 πριν περάσει στο 74HC14.



Εικόνα 12: Κύκλωμα υλοποίησης και επεξεργασίας θορύβου

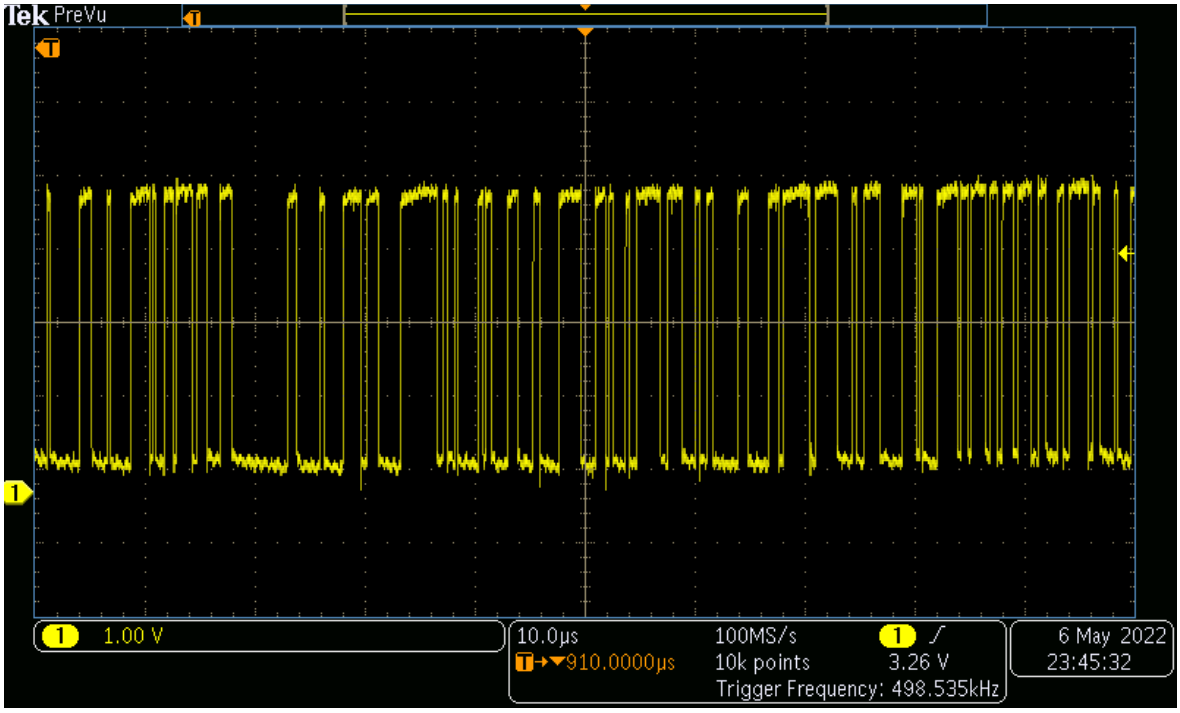


Εικόνα 13: Το τυχαίο σήμα που εξέρχεται από το τρανζίστορ 3 πριν εισαχθεί στο 74HC14.

Τα ολοκληρωμένα κυκλώματα (74HC14, 74HC74) που χρησιμοποιούνται για την επεξεργασία του θορύβου τροφοδοτούνται από την παροχή του Arduino Nano στα 5V. Το Arduino Nano τροφοδοτείται από την σύνδεση του ηλεκτρονικού υπολογιστή μέσω του καλωδίου type-B mini-USB.

Στο ολοκληρωμένο κύκλωμα 74HC14 χρησιμοποιείται μόνο ο πρώτος μετατροπέας (1A – 1Y) που μεταμορφώνει τις αναλογικές τιμές τάσεων σε μία ακουλουθία 0 και 1 όπως εξηγήθηκε προηγουμένως στο κεφάλαιο 4.1.2.8. Στην εικόνα 14 που παρατηρήθηκε με την

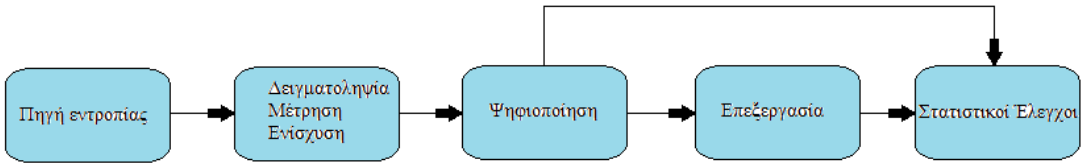
χρήση παλμογράφου φαίνεται ότι παρά την ψηφιοποίηση του σήματος που επιτυγχάνεται, υπάρχουν χρονικά περιθώρια όπου το σήμα παραμένει υψηλό ή χαμηλό χωρίς να υπάρχει ισορροπία στην συχνότητα εμφάνισης 0 και 1. Αυτό διορθώνεται με το ολοκληρωμένο 74HC74.



Εικόνα 14: Μετατροπή αναλογικού σήματος σε ψηφιακό με το ολοκληρωμένο 74HC14.

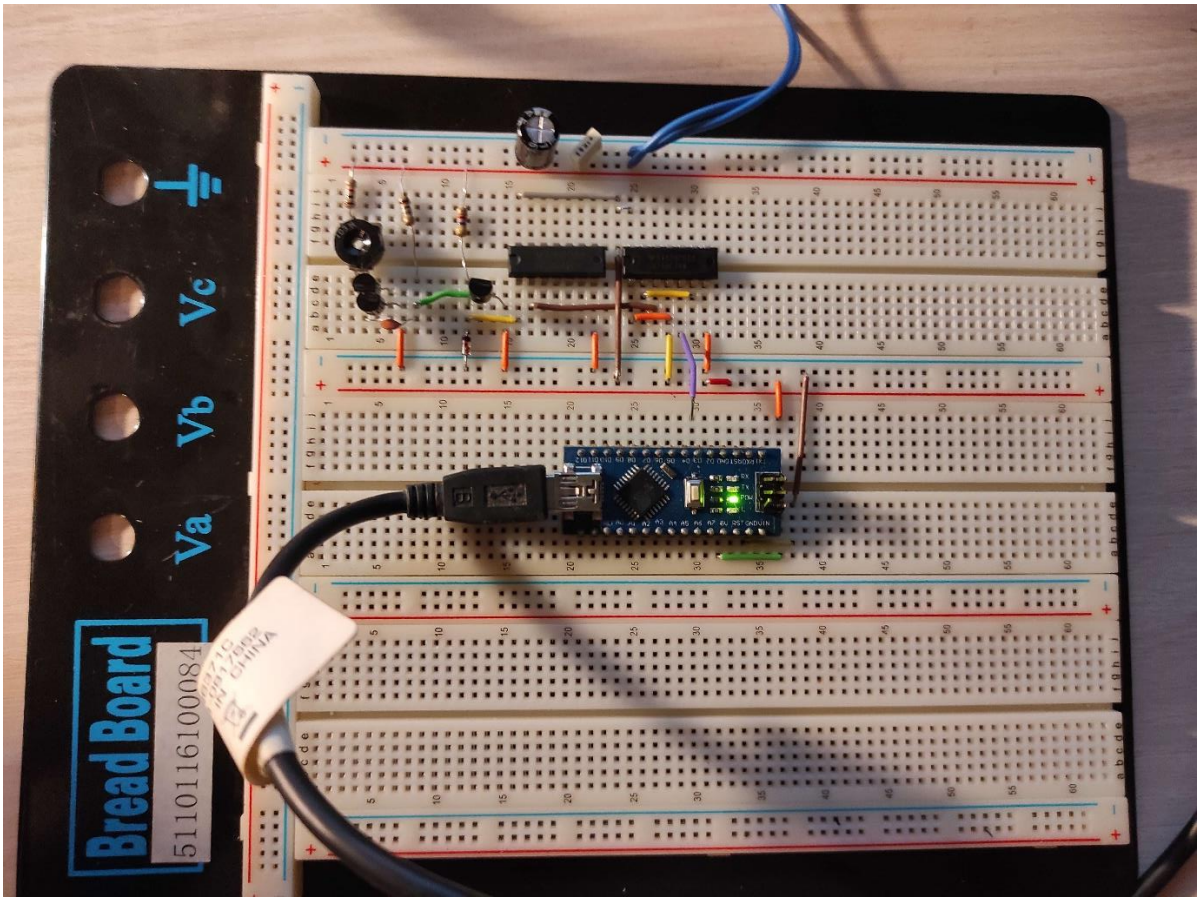
Έπειτα το ψηφιακό σήμα περνάει στο ολοκληρωμένο κύκλωμα 74HC74 όπου εκτελείται η D type flip-flop. Η συγκεκριμένη flip-flop δεν κάνει τίποτα μέχρι να λάβει την θετική άνοδο ενός σήματος στην υποδοχή του ρολογιού της. Έπειτα επεξεργάζεται την πληροφορία και αντιγράφει την κατάσταση της εισόδου της D στην έξοδο Q. Μετά αποθηκεύει την αντίθετη κατάσταση του D στο NOT Q. Το ολοκληρωμένο περιμένει το επόμενο σήμα ρολογιού κρατώντας παράλληλα τις δύο καταστάσεις εξόδου. Εάν το ρολόι μείνει στην ίδια κατάσταση τότε και η flip flop μένει ανενεργή μέχρι να γίνει αλλαγή στην είσοδο. Αυτό το κύκλωμα χρησιμοποιήθηκε κυρίως για να ισορροπηθεί η συχνότητα εμφάνισης 0 με την συχνότητα εμφάνισης 1 ώστε να μην υπάρχει συσχέτιση ή εξάρτηση του συστήματος από το ένα ή το άλλο.

Στην εικόνα 15 βλέπουμε τα βήματα για την δημιουργία μιας γεννήτριας τυχαίων αριθμών.



Εικόνα 15: Γενική διαδικασία παραγωγής πραγματικής γεννήτριας τυχαιών αριθμών.

Μετά την επεξεργασία του σήματος μέσω των ολοκληρομένων κυκλωμάτων η τελική ακολουθία bit εισάγεται σε μία από τις ψηφιακές του υποδοχές(D2-D13) του Arduino Nano.

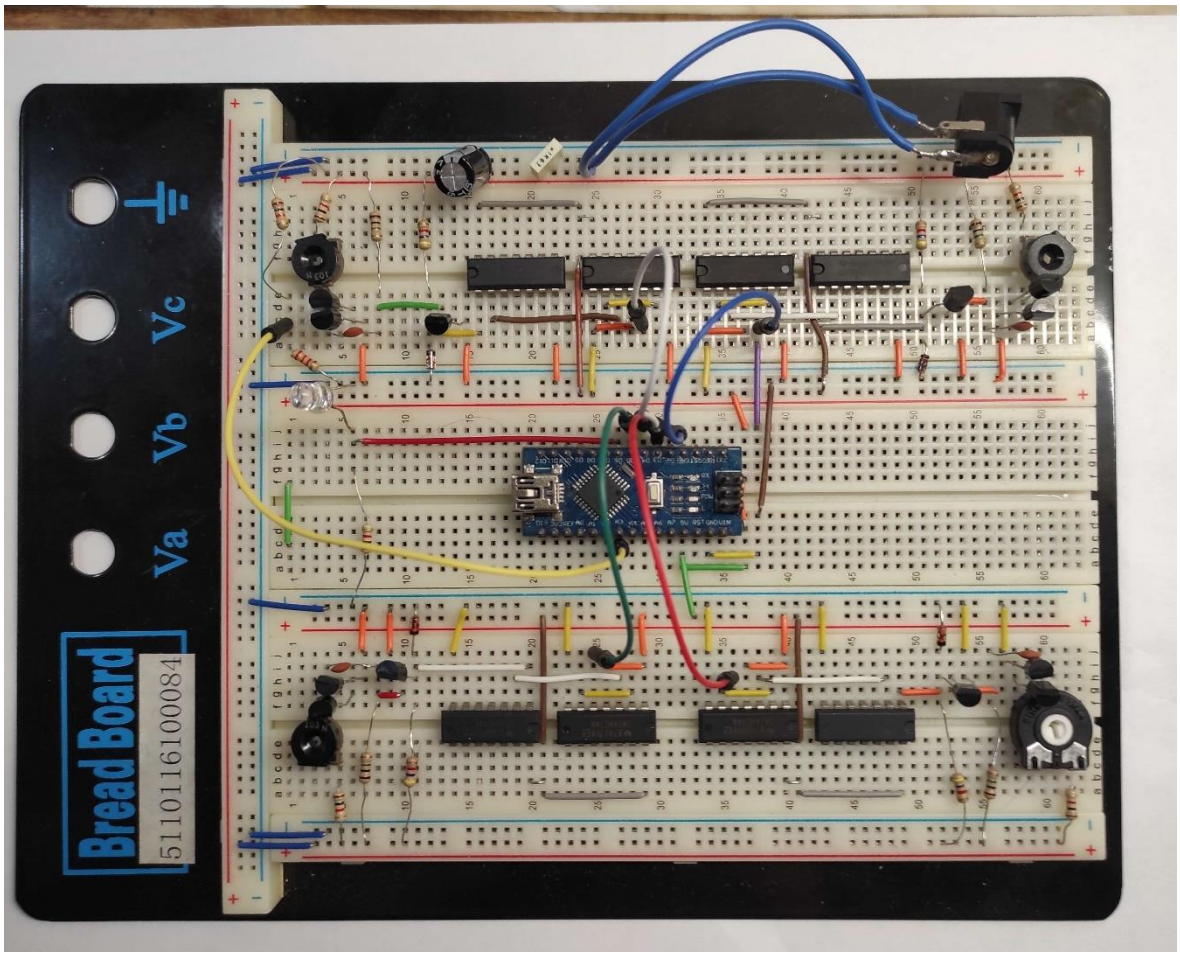


Εικόνα 16: Υλοποίηση κυκλώματος με ένα κανάλι ήχου.

Σε αυτό το κύκλωμα η εντροπία ήταν 8 bit στο byte για δεδομένα μεγαλύτερα των 100mb(πριν την επεξεργασία μέσω του Arduino IDE), ωστόσο η ποσότητα παραγωγής της ήταν μικρή. Για τον λόγο αυτό προστέθηκαν ακόμα 3 πανομοιότυπα κυκλώματα στο

breadboard ώστε να τετραπλασιαστεί η ποσότητα του θορύβου. Στην εικόνα 17 δίνεται το κύκλωμα ολοκληρωμένο χωρίς να βρίσκεται υπό ηλεκτρικό φορτίο.

Σε αυτό το κομμάτι βοήθησε το μέγεθος του Arduino Nano, καθώς με οποιαδήποτε άλλη μεγαλύτερη πλακέτα Arduino δεν θα χωρούσαν τα υπόλοιπα κυκλώματα θορύβου. Η εντροπία του συστήματος είναι στα 8 bits άνα byte και μεταφέρεται στο Arduino μέσω USB με ρυθμό στα 115200 bauds. Επειδή το σήμα είναι ψηφιακό (0, 1) αυτό σημαίνει ότι στέλνονται δεδομένα με ρυθμό περίπου 12kb/s.



Εικόνα 17: Υλοποίηση και των τεσσάρων πηγών και τροφοδοσία θορύβου μέσω των jumper wires στο Arduino.

4.2.3 Post processing

Οι γεννήτριες παραγωγής τυχαίων αριθμών δεν μπορούν να φτιαχτούν και να λειτουργήσουν σε επίπεδο τελειότητας. Post-processing (αλλιώς και data whitening) ονομάζεται η διαδικασία κατά την οποία αφού έχει παραχθεί το τυχαίο ψηφιακό σήμα

επεξεργάζεται με σκοπό να βελτιώσει την εντροπία και την αξιοπιστία του σήματος[16]. Υπάρχουν κυρίως δύο διαφορετικές μεθόδοι :

- Αλγοριθμικό post processing : Εισάγωντας το ψηφιακά δεδομένα σε έναν αλγόριθμο με σκοπό να βελτιώσει τις στατιστικές αποδόσεις και εξαρτήσεις που έχουν. Συνήθως συμβαίνει με την χρήση αλγορίθμων XOR.
- Κρυπτογραφικό post processing : Συμπίεση δεδομένων με την χρήση κρυπτογραφικών αλγορίθμων που βασίζονται στον κατακερματισμό. Σύνηθες τέτοιες τεχνικές είναι οι SHA hash functions[17].

Σε αυτό το σχέδιο post processing έχει γίνει με το ολοκληρωμένο 74HC74 που απάλειψε την πιθανότητα εμφανίσης περισσότερων άσσων από μηδενικά και το αντίστροφο μέσω της χρήσης D flip flop (positive edge). Αυτή η διαδικασία παρουσιάστηκε πρώτη φορά από τον Von Neumann[4], ωστόσο έτσι μειώνεται η ακουλουθία στο μίσο της γιατί κάθε ζευγάρι (0,0 και 1,1) δεν προστίθεται στην τελική ακολουθία.

Μία ακομά τεχνική που χρησιμοποιείται σε αυτό το σχέδιο είναι με το πέρασμα των δεδομένων από μία 8bit LFSR και ύστερα μία ακόμα 16bit LFSR. Τα δεδομένα πριν περαστούν στις LFSR έχουν ήδη 8bit εντροπίας στο byte για αυτό και δεν θεωρείται το σχέδιο ως PRNG. Έπισης οι LFSR πριν την έξοδο της ακουλουθίας ανατροφοδοτούνται με μία καθυστέρηση που παίρνει τυχαίες τιμές από το Avalanche effect του τρανζίστορ 1. Η τεχνική υλοποίηση αυτών παρουσιάζεται στο 6^{ο} κεφάλαιο.

Το post-processing μπορεί να βελτιώσει την εντροπία (ανά bit) στο αποτέλεσμα της γεννήτριας χωρίς όμως να έχει την δυνατότητα να παράγει εντροπία. Επίσης η χρήση τέτοιων διαδικασίων συνήθως μικραίνουν το αποτέλεσμα όσον αναφορά το μέγεθος των bit (όπως στην περιπτώση του σχεδίου αυτού) και καθυστερούν τον συνολικό έργο λόγω του χρόνου λειτουργίας τους. Σκόπος μίας ιδανικής γεννήτριας τυχαίων αριθμών είναι να μην χρειάστει να καθαρίσει τα δεδομένα της, ώστε να είναι ανεξάρτητη από το post-processing.

5 Αποτελέσματα Αξιολογήσεων

Ένα μεγάλο κομμάτι στην παραγωγή τυχαίων αριθμών είναι η αξιολόγηση τους. Μέσω δοκιμών μπορεί ο καθένας να ελέγξει που υστερούν τα δεδομένα όσον αναφορά την τυχαιότητα. Τα περισσότερα τεστ τυχαιότητας συγκρίνουν και δοκιμάζουν τις στατιστικές μετρήσεις μεγάλων ακουλουθιών αριθμών του τελικού αποτελέσματος. Κάποια τεστ είναι προσανατολισμένα περισσότερο για τον έλεγχο PRNG, άλλα TRNG και κάποια είναι γενικής φύσεως.

Σε αυτό το κεφάλαιο παραθέτονται τα αποτελέσματα τριών γνωστών τεστ τυχαιότητας. Το ENTTEST[18] και το RNGTEST είναι κατάλληλα για την δοκιμή ενός πρωτότυπου RNG καθώς είναι γρήγορα, παρέχουν σημαντικές πληροφορίες και είναι βασισμένα στο πρωτόκολλο NIST[19]. Το πιο εκτεταμένο τεστ που παρουσίαζεται είναι το DieHarder (A random number test suite)[20].

Για την δοκιμή των αποτελεσμάτων της γεννήτριας χρησιμοποιήθηκε ένα αρχείο των 4GB με όνομα bin4GB.

5.1 ENT TEST

Το ENT test εκτελεί 5 βασικά τεστ τυχαιότητας σε ακουλουθίες byte. Συνήθως χρησιμοποιείται για ψευδείς γεννήτριες που βασίζονται σε αλγόριθμους αλλά λειτουργεί το ίδιο και σε πραγματικές.

Εντροπία : Η πυκνότητα πληροφορίας ενός αρχείο μεταφρασμένη ως ένα αριθμό bit ανά byte. Όσο μεγαλύτερος είναι ο αριθμός τόσο μεγαλύτερη και η τυχαιότητα του αρχείου.

Κατανομή x^2 : η κατανομή υπολογίζεται για την πληθώρα των byte στο αρχείο ως έναν αριθμό και ένα ποσοστό του κάθε πόσο συχνά μία πραγματικά τυχαία ακουλουθία αριθμών θα ξεπερνούσε την τιμή αυτήν. Το ποσοστό αυτό μεταφράζεται στον βαθμό εάν η ακουλουθία μας είναι ύποπτη για μη τυχαιότητα. Οι τιμές που αποδεικνύουν την τυχαιότητα βρίσκοντα ανάμεσα στο ποσοστό 10% - 90%.

Arithmetic Mean value: Είναι το αποτέλεσμα προσθέτωντας όλων των byte του αρχείου και διαιρώντας με το μήκος του φακέλου(file length). Εάν η διαίρεση δεν τείνει στο αποτέλεσμα τιμής 127.5 τότε τα δεδομένα δεν είναι κοντά στο να είναι τυχαία.

Monte Carlo Value for Pi : Κάθε διαδεχομένη ακολουθία των 6 byte χρησιμοποιείται ως 24 bit X και Y συντεταγμένες σε ένα κύκλο. Εάν η απόσταση από το τυχαίο σημείο που ορίστηκε από τις συντεταγμένες είναι μικρότερο από την ακτίνα του κύκλου η ακουλουθία των 6 byte καταχωρείται ως hit. Το ποσοστό των hits χρησιμοποείται για να βρεθεί η τιμή του Pi. Για μεγάλες ακουλουθίες byte εάν η τιμή φτάνει στην τιμή του Pi τότε οι ακολουθίες είναι κοντά στο να είναι τυχαίες.

Serial Correlation Coefficient: Σε αυτό το τεστ μετριέται κατά πόσο είναι εξαρτημένο το κάθε byte από το προηγούμενο του. Για να είναι τυχαίες ακουλουθίες η ποσότητα που μετριέται πρέπει να τείνει στο μηδέν.

```
metaplita@metaplita-HP-Pro-3500-Series:~$ cat /home/metaplita/trngtest/bin4gb | ent
Entropy = 8.000000 bits per byte.

Optimum compression would reduce the size
of this 4500000000 byte file by 0 percent.

Chi square distribution for 4500000000 samples is 235.28, and randomly
would exceed this value 80.70 percent of the times.

Arithmetic mean value of data bytes is 127.4995 (127.5 = random).
Monte Carlo value for Pi is 3.141589408 (error 0.00 percent).
Serial correlation coefficient is -0.000021 (totally uncorrelated = 0.0).
```

Εικόνα 18: Αποτελέσματα Ent-test.

Η εντροπία του φακέλου δεν θα μπορούσε να ήταν καλύτερη και όλα τα τεστ έκριναν ότι το αρχείο είναι τυχαίο.

5.2 RNG TEST

Το τεστ αυτό δοκιμάζει κάθε φορά ακουλουθίες των 20000 bits , βασισμένο στο πρωτόκολλο FIPS 140-2[21]. Κάθε τέτοια ακολουθία υποβάλλεται σε 5 τεστ :

Monobit test: Σκοπός αυτού του τεστ είναι να κρίνει εάν η συχνότητα εμφάνισης 0 είναι ίση με την συχνότητα εμφάνισης 1.

Poker test: Η ακολουθία των 20000bit διαρείται σε 5000 μικρότερες από 4bit(nibble) η κάθεμια. Έπειτα μετράει τις τιμές που έχει η κάθεμια ακουλουθία των 4bit και συγκρίνει τις συχνότητες ανέμεσα στις τιμές.

Runs test: Ένα run ορίζεται η ακουλουθία συνεχόμενων bit είτε είναι 0 είτε 1. Οι φορές εμφάνισης και τα μεγέθη των ακουλουθιών ζυγίζονται για να κριθούν εάν είναι τυχαία.

Longest Run in a block: Ένα long run θεωρείται μια ακουλουθία ίδιων bit με μέγεθος μεγαλύτερο και ίσο των 26. Σε ένα δείγμα των 20000 bits εάν δεν υπάρχουν long run τότε το τεστ θεωρείται πετυχημένο.

Continuous run: Αυτό το τεστ ψάχνει να βρει ακουλουθίες των 32 bit και εάν βρει 2 με την μία να διαδέχεται την άλλη στο ίδιο μπλοκ τότε αποτυγχάνει.

```
metaplita@metaplita-HP-Pro-3500-Series:~$ cat /home/metaplita/trngtest/bin4gb | rngtest
rngtest 5
Copyright (c) 2004 by Henrique de Moraes Holschuh
This is free software; see the source for copying conditions.  There is NO warranty; no

rngtest: starting FIPS tests...
rngtest: entropy source drained
rngtest: bits received from input: 36000000000
rngtest: FIPS 140-2 successes: 1798637
rngtest: FIPS 140-2 failures: 1362
rngtest: FIPS 140-2(2001-10-10) Monobit: 181
rngtest: FIPS 140-2(2001-10-10) Poker: 167
rngtest: FIPS 140-2(2001-10-10) Runs: 519
rngtest: FIPS 140-2(2001-10-10) Long run: 506
rngtest: FIPS 140-2(2001-10-10) Continuous run: 1
rngtest: input channel speed: (min=198019801.980; avg=25597200802.330; max=0.000)bits/s
rngtest: FIPS tests speed: (min=28.899; avg=227.161; max=244.532)Mibits/s
rngtest: Program run time: 152631117 microseconds
```

Εικόνα 19: Αποτελέσματα RNG test.

Από την εικόνα 5.2 παρατηρείται ότι το αρχείο έχει 1798637 επιτυχίες και 1362 αποτυχίες. Αυτό δίνει ένα ποσοστό αποτυχίας 0.07%.

5.3 Diehard

Τα Diehard τεστ ήταν μία σειρά στατιστικών δοκιμών για να μετρήσουν την ποιότητα γεννήτριας παραγωγής τυχαίων αριθμών. Είχαν δημιουργηθεί από τον George Marsaglia[22] και πρωτοκυκλοφόρησαν το 1995 με την μορφή CD-ROM. Μετά από καιρό περάστηκαν στο διαδίκτυο μέσω του Robert G. Brown[20].

```
metaplita@metaplita-HP-Pro-3500-Series:~$ dieharder -g 201 -f /home/metaplita/trngtest/bin4gb -a
#=============================================================================#
#            dieharder version 3.31.1 Copyright 2003 Robert G. Brown          #
#=============================================================================#
   rng_name    |           filename             |rands/second|
 file_input_raw| /home/metaplita/trngtest/bin4gb|  3.84e+06  |
#=============================================================================#
        test_name   |ntup| tsamples |psamples|  p-value |Assessment
#=============================================================================#
   diehard_birthdays|   0|       100|     100|0.62987918|  PASSED
      diehard_operm5|   0|   1000000|     100|0.61330670|  PASSED
  diehard_rank_32x32|   0|     40000|     100|0.34467284|  PASSED
    diehard_rank_6x8|   0|    100000|     100|0.80155068|  PASSED
   diehard_bitstream|   0|   2097152|     100|0.90349132|  PASSED
        diehard_opso|   0|   2097152|     100|0.77228097|  PASSED
        diehard_oqso|   0|   2097152|     100|0.80448981|  PASSED
         diehard_dna|   0|   2097152|     100|0.59088744|  PASSED
diehard_count_1s_str|   0|    256000|     100|0.98065618|  PASSED
diehard_count_1s_byt|   0|    256000|     100|0.87462627|  PASSED
 diehard_parking_lot|   0|     12000|     100|0.24520972|  PASSED
    diehard_2dsphere|   2|      8000|     100|0.85926025|  PASSED
    diehard_3dsphere|   3|      4000|     100|0.19321178|  PASSED
     diehard_squeeze|   0|    100000|     100|0.42339142|  PASSED
        diehard_sums|   0|       100|     100|0.97430865|  PASSED
        diehard_runs|   0|    100000|     100|0.27377697|  PASSED
        diehard_runs|   0|    100000|     100|0.04999923|  PASSED
# The file file_input_raw was rewound 1 times
       diehard_craps|   0|    200000|     100|0.16813178|  PASSED
       diehard_craps|   0|    200000|     100|0.11272742|  PASSED
# The file file_input_raw was rewound 2 times
 marsaglia_tsang_gcd|   0|  10000000|     100|0.16845353|  PASSED
 marsaglia_tsang_gcd|   0|  10000000|     100|0.67536401|  PASSED
# The file file_input_raw was rewound 2 times
         sts_monobit|   1|    100000|     100|0.58417764|  PASSED
# The file file_input_raw was rewound 2 times
            sts_runs|   2|    100000|     100|0.10921032|  PASSED
# The file file_input_raw was rewound 2 times
          sts_serial|   1|    100000|     100|0.40930473|  PASSED
          sts_serial|   2|    100000|     100|0.84496681|  PASSED
          sts_serial|   3|    100000|     100|0.32408801|  PASSED
          sts_serial|   3|    100000|     100|0.26031222|  PASSED
          sts_serial|   4|    100000|     100|0.56559545|  PASSED
          sts_serial|   4|    100000|     100|0.46364322|  PASSED
          sts_serial|   5|    100000|     100|0.09612489|  PASSED
          sts_serial|   5|    100000|     100|0.52588129|  PASSED
          sts_serial|   6|    100000|     100|0.23872579|  PASSED
          sts_serial|   6|    100000|     100|0.82958757|  PASSED
          sts_serial|   7|    100000|     100|0.93692161|  PASSED
          sts_serial|   7|    100000|     100|0.73024782|  PASSED
          sts_serial|   8|    100000|     100|0.83136604|  PASSED
          sts_serial|   8|    100000|     100|0.45634183|  PASSED
          sts_serial|   9|    100000|     100|0.61967758|  PASSED
          sts_serial|   9|    100000|     100|0.96481644|  PASSED
          sts_serial|  10|    100000|     100|0.32728692|  PASSED
          sts_serial|  10|    100000|     100|0.05795085|  PASSED
```

Εικόνα 20: Diehard αποτελέσματα(1).

```
          sts_serial|  11|    100000|     100|0.66390756|  PASSED
          sts_serial|  11|    100000|     100|0.80958809|  PASSED
          sts_serial|  12|    100000|     100|0.61362998|  PASSED
          sts_serial|  12|    100000|     100|0.40262075|  PASSED
          sts_serial|  13|    100000|     100|0.03350914|  PASSED
          sts_serial|  13|    100000|     100|0.04770561|  PASSED
          sts_serial|  14|    100000|     100|0.06770893|  PASSED
          sts_serial|  14|    100000|     100|0.12207467|  PASSED
          sts_serial|  15|    100000|     100|0.04070591|  PASSED
          sts_serial|  15|    100000|     100|0.04499635|  PASSED
          sts_serial|  16|    100000|     100|0.00501344|  PASSED
          sts_serial|  16|    100000|     100|0.13144196|  PASSED
# The file file_input_raw was rewound 2 times
         rgb_bitdist|   1|    100000|     100|0.93308604|  PASSED
# The file file_input_raw was rewound 2 times
         rgb_bitdist|   2|    100000|     100|0.80843489|  PASSED
# The file file_input_raw was rewound 3 times
         rgb_bitdist|   3|    100000|     100|0.40262884|  PASSED
# The file file_input_raw was rewound 3 times
         rgb_bitdist|   4|    100000|     100|0.03589479|  PASSED
# The file file_input_raw was rewound 3 times
         rgb_bitdist|   5|    100000|     100|0.88747328|  PASSED
# The file file_input_raw was rewound 3 times
         rgb_bitdist|   6|    100000|     100|0.97454405|  PASSED
# The file file_input_raw was rewound 3 times
         rgb_bitdist|   7|    100000|     100|0.21972998|  PASSED
# The file file_input_raw was rewound 3 times
         rgb_bitdist|   8|    100000|     100|0.94820973|  PASSED
# The file file_input_raw was rewound 3 times
         rgb_bitdist|   9|    100000|     100|0.99389257|  PASSED
# The file file_input_raw was rewound 3 times
         rgb_bitdist|  10|    100000|     100|0.92103848|  PASSED
# The file file_input_raw was rewound 4 times
         rgb_bitdist|  11|    100000|     100|0.01463582|  PASSED
# The file file_input_raw was rewound 4 times
         rgb_bitdist|  12|    100000|     100|0.28830341|  PASSED
# The file file_input_raw was rewound 4 times
rgb_minimum_distance|   2|     10000|    1000|0.40372348|  PASSED
# The file file_input_raw was rewound 4 times
rgb_minimum_distance|   3|     10000|    1000|0.65086153|  PASSED
# The file file_input_raw was rewound 4 times
rgb_minimum_distance|   4|     10000|    1000|0.72855039|  PASSED
# The file file_input_raw was rewound 4 times
rgb_minimum_distance|   5|     10000|    1000|0.36602944|  PASSED
# The file file_input_raw was rewound 4 times
    rgb_permutations|   2|    100000|     100|0.99622051|   WEAK
# The file file_input_raw was rewound 4 times
    rgb_permutations|   3|    100000|     100|0.53591390|  PASSED
# The file file_input_raw was rewound 4 times
    rgb_permutations|   4|    100000|     100|0.97077333|  PASSED
# The file file_input_raw was rewound 4 times
    rgb_permutations|   5|    100000|     100|0.92706596|  PASSED
# The file file_input_raw was rewound 4 times
      rgb_lagged_sum|   0|   1000000|     100|0.96284268|  PASSED
```

Εικόνα 21: Αποτελέσματα Diehard (2).

```
# The file file_input_raw was rewound 4 times
      rgb_lagged_sum|   1|   1000000|     100|0.71225574|  PASSED
# The file file_input_raw was rewound 5 times
      rgb_lagged_sum|   2|   1000000|     100|0.99197824|  PASSED
# The file file_input_raw was rewound 5 times
      rgb_lagged_sum|   3|   1000000|     100|0.51268884|  PASSED
# The file file_input_raw was rewound 5 times
      rgb_lagged_sum|   4|   1000000|     100|0.58113800|  PASSED
# The file file_input_raw was rewound 6 times
      rgb_lagged_sum|   5|   1000000|     100|0.36721706|  PASSED
# The file file_input_raw was rewound 7 times
      rgb_lagged_sum|   6|   1000000|     100|0.30556508|  PASSED
# The file file_input_raw was rewound 7 times
      rgb_lagged_sum|   7|   1000000|     100|0.68221219|  PASSED
# The file file_input_raw was rewound 8 times
      rgb_lagged_sum|   8|   1000000|     100|0.50295269|  PASSED
# The file file_input_raw was rewound 9 times
      rgb_lagged_sum|   9|   1000000|     100|0.78222235|  PASSED
# The file file_input_raw was rewound 10 times
      rgb_lagged_sum|  10|   1000000|     100|0.88998792|  PASSED
# The file file_input_raw was rewound 11 times
      rgb_lagged_sum|  11|   1000000|     100|0.51932272|  PASSED
# The file file_input_raw was rewound 12 times
      rgb_lagged_sum|  12|   1000000|     100|0.56855276|  PASSED
# The file file_input_raw was rewound 13 times
      rgb_lagged_sum|  13|   1000000|     100|0.34047842|  PASSED
# The file file_input_raw was rewound 15 times
      rgb_lagged_sum|  14|   1000000|     100|0.22958426|  PASSED
# The file file_input_raw was rewound 16 times
      rgb_lagged_sum|  15|   1000000|     100|0.89621324|  PASSED
# The file file_input_raw was rewound 18 times
      rgb_lagged_sum|  16|   1000000|     100|0.98640069|  PASSED
# The file file_input_raw was rewound 19 times
      rgb_lagged_sum|  17|   1000000|     100|0.03283002|  PASSED
# The file file_input_raw was rewound 21 times
      rgb_lagged_sum|  18|   1000000|     100|0.37065596|  PASSED
# The file file_input_raw was rewound 23 times
      rgb_lagged_sum|  19|   1000000|     100|0.77339030|  PASSED
# The file file_input_raw was rewound 25 times
      rgb_lagged_sum|  20|   1000000|     100|0.74077771|  PASSED
# The file file_input_raw was rewound 27 times
      rgb_lagged_sum|  21|   1000000|     100|0.51039448|  PASSED
# The file file_input_raw was rewound 29 times
      rgb_lagged_sum|  22|   1000000|     100|0.83561307|  PASSED
# The file file_input_raw was rewound 31 times
      rgb_lagged_sum|  23|   1000000|     100|0.97153097|  PASSED
# The file file_input_raw was rewound 33 times
      rgb_lagged_sum|  24|   1000000|     100|0.13087572|  PASSED
# The file file_input_raw was rewound 35 times
      rgb_lagged_sum|  25|   1000000|     100|0.59764621|  PASSED
# The file file_input_raw was rewound 38 times
      rgb_lagged_sum|  26|   1000000|     100|0.19685308|  PASSED
# The file file_input_raw was rewound 40 times
      rgb_lagged_sum|  27|   1000000|     100|0.67179791|  PASSED
```

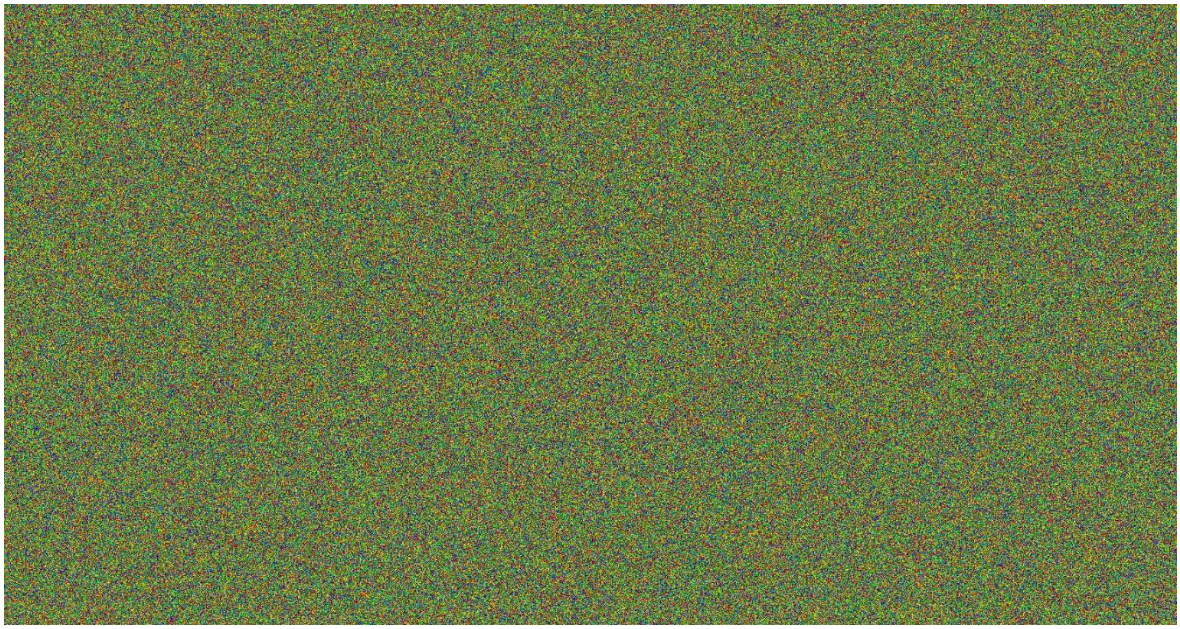
Εικόνα 22: Αποτελέσματα Diehard(3).

```
# The file file_input_raw was rewound 43 times
      rgb_lagged_sum|  28|    1000000|     100|0.40243669|  PASSED
# The file file_input_raw was rewound 45 times
      rgb_lagged_sum|  29|    1000000|     100|0.59219083|  PASSED
# The file file_input_raw was rewound 48 times
      rgb_lagged_sum|  30|    1000000|     100|0.64812182|  PASSED
# The file file_input_raw was rewound 51 times
      rgb_lagged_sum|  31|    1000000|     100|0.43113383|  PASSED
# The file file_input_raw was rewound 54 times
      rgb_lagged_sum|  32|    1000000|     100|0.49017049|  PASSED
# The file file_input_raw was rewound 54 times
     rgb_kstest_test|   0|      10000|    1000|0.96557453|  PASSED
# The file file_input_raw was rewound 54 times
     dab_bytedistrib|   0|   51200000|       1|0.51609709|  PASSED
# The file file_input_raw was rewound 54 times
             dab_dct| 256|      50000|       1|0.82994075|  PASSED
Preparing to run test 207.  ntuple = 0
# The file file_input_raw was rewound 54 times
        dab_filltree|  32|   15000000|       1|0.73759806|  PASSED
        dab_filltree|  32|   15000000|       1|0.64333676|  PASSED
Preparing to run test 208.  ntuple = 0
# The file file_input_raw was rewound 54 times
       dab_filltree2|   0|    5000000|       1|0.93007754|  PASSED
       dab_filltree2|   1|    5000000|       1|0.75828184|  PASSED
Preparing to run test 209.  ntuple = 0
# The file file_input_raw was rewound 54 times
        dab_monobit2|  12|   65000000|       1|0.05114150|  PASSED
```

Εικόνα 23: Αποτελέσματα Diehard(4).

Από τις εικόνες παρατηρείται ότι δεν υπήρχε κανένα FAILED τεστ, ένα μόνο WEAK που θεωρείται αποδεκτό και όλα τα υπόλοιπα ήταν επιτυχές. Επίσης είναι σημαντικό να γίνονται τα τεστ με μεγάλα αρχεία καθώς μεγαλώνουν την αξιοπιστία των αποτελεσμάτων.

Επιπλέον μέσω του λογισμικού Ubuntu δημουργήθηκε μία εικόνα bitmap από το αρχείο των 4GB, όπου κάθε pixel της εικόνας αναπαραστάται από ένα bit. Αυτό φυσικά δεν είναι κάποιο σοβάρο τεστ αλλά είναι αρκετό για να δείξει εάν κάτι δεν δουλεύει σωστά.



Εικόνα 24: Bitmap από το 4GB αρχείο.

6 Τεχνική Υλοποίηση

Στο κεφάλαιο αυτό θα αναλυθεί ο προγραμματισμός του Arduino Nano και η χρήση του λογισμικού IDE. Επίσης θα αναλυθούν και οι τεχνικές λεπτομέρειες που πραγματοποιήθηκαν για την επεξεργασία και την συλλογή του τυχαίου θορύβου από το κύκλωμα. Η υλοποίηση του σχεδίου και η εφαρμογή του πραγματοποιήθηκαν στο λειτουργικό Ubuntu 22.04 LTS ώστε να υπάρχει άμεση επικοινωνία μέσω του τερματικού.

6.1 ARDUINO IDE

Για την επεξεργασία του σήματος και τον προγραμματισμό του μικροελεγκτή χρησιμοποιήθηκε το ολοκληρωμένο περιββάλον αναπτυξής (IDE). Τα προγράμματα τα οποία γράφει ο χρήστης ονομάζονται σχέδια και περιλαμβάνονται σε ένα επεξεργαστή κειμένου(text editor). Κάτω από αυτό το χώρο κειμένου υπάρχει ένα μικρο τερματικό. Ο χρήστης μπορεί να επιβεβαιώσει τον κώδικα του για λάθη σύνταξης και τα μηνύματα λάθος παρουσιάζονται στο τερματικό.

Για να μπορέσουμε να περάσουμε τον κώδικα στην πλακέτα Arduino μας θα πρέπει πρώτα να τηρούνται κάποια κριτήρια:

- Ο κώδικας θα πρέπει να είναι συντακτικά σωστός και να μην εμφανίζεται μήνυμα λάθους.
- Η πλακέτα Arduino που χρησιμοποιούμε να είναι συνδεδεμένη μέσω USB στον υπολογιστή μας.
- Να διαλέξουμε μέσω του λογισμικού IDE την θύρα στην οποία συνδέουμε το Arduino. (πχ /dev/ttyACMx , /dev/ttyUSBx)
- Να διαλέξουμε το μοντέλο της πλακέτας το οποίο θα προγραμματίσουμε μέσω του λογισμικού

Μία ακόμα πολύ χρήσιμη λειτουργία είναι το Serial Monitor όπου μπορούμε να παρακολουθήσουμε πληροφορίες που στέλνει η πλακέτα Arduino στον υπολογιστή ενώ παράλληλα εκτελείται το πρόγραμμα μας. Έτσι δεν χρειάζεται να περιμένουμε τον χρόνο

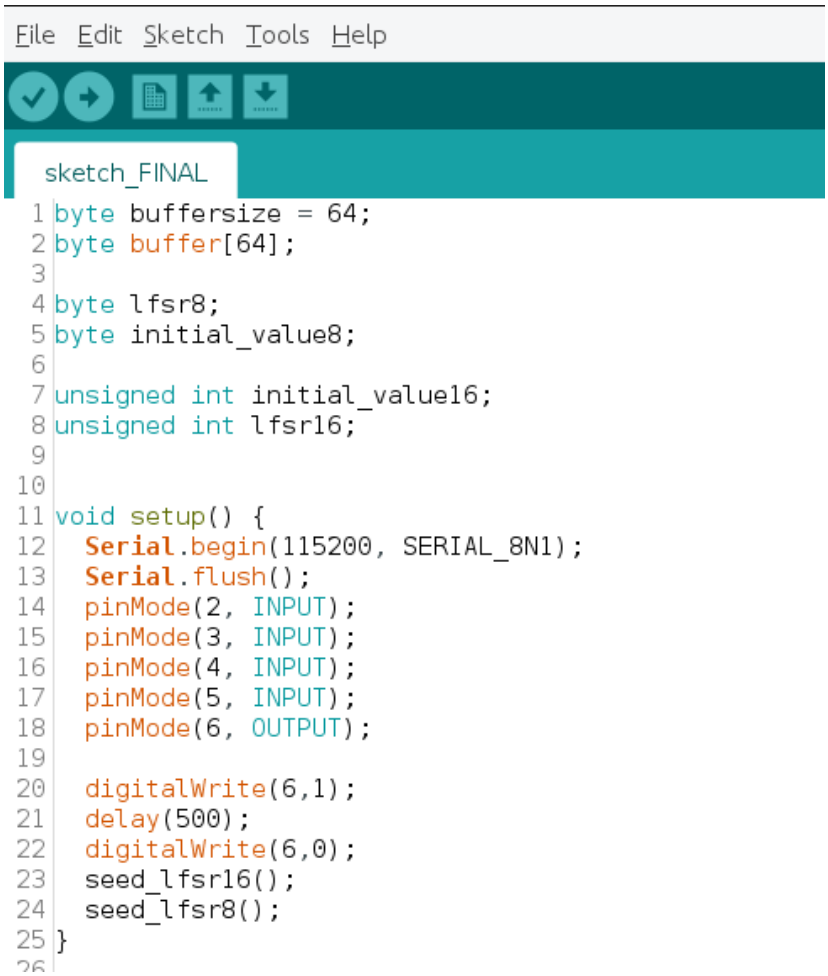
εκτέλεσης του προγράμματος. Αυτό είναι μία σημαντική βοήθεια όταν προσπαθούμε να παράγουμε μεγάλα αρχεία τυχαίων δεδομένων.

Τέλος περιέχει μία μπάρα εργαλειών η οποία αποτελείται από συνηθισμένες συναρτήσεις τις οποίες μπορούμε να πάρουμε έτοιμες και να τις χρησιμοποιήσουμε στο σχέδιο μας. Κάποια παραδείγματα είναι η ενεργοποίηση ενός led και η λύσεις μαθηματικών προβλημάτων.

6.2 Κώδικας Arduino Nano

6.2.1 Setup

Στο 4ο κεφάλαιο αναφέραμε ότι ο κώδικας που γράφουμε στο Arduino IDE χωρίζεται σε δύο συναρτήσεις τύπου void, την setup όπου γινόνται αρχικοποιήσεις για την λειτουργία της πλακέτας όσο αναφορά τις εισόδους και τις εξόδους της και την συνάρτηση βρόγχου.




```
byte buffersize = 64;
byte buffer[64];

byte lfsr8;
byte initial_value8;

unsigned int initial_value16;
unsigned int lfsr16;


void setup() {
  Serial.begin(115200, SERIAL_8N1);
  Serial.flush();
  pinMode(2, INPUT);
  pinMode(3, INPUT);
  pinMode(4, INPUT);
  pinMode(5, INPUT);
  pinMode(6, OUTPUT);

  digitalWrite(6,1);
  delay(500);
  digitalWrite(6,0);
  seed_lfsr16();
  seed_lfsr8();
}
```

Εικόνα 25: Αρχικοποίηση τιμών και λειτουργίας υποδοχών του μικροελεγκτή.

Στην εικόνα 25 βλέπουμε την δήλωση μεταβλητών και των εισόδων που χρησιμοποιεί το Arduino Nano για την συλλογή θορύβου. Η lfsr8 και το initial_value8 χρησιμοποιούνται για τις δυαδικές πράξεις μεταξύ bit έχοντας τελική τιμή 8bit. Αντίστοιχα την ίδια λειτουργία έχουν και οι lfsr16 και initial_value16 όμως με χωρητικότητα τα 16bit. Οι μεταβλητές αυτές χρησιμοποιούνται ύστερα για το post – processing των δεδομένων με την χρήση LFSR.

Το Serial.begin(115200, SERIAL_8N1) χρησιμοποιείται για να γεφυρώσει τον ρυθμό μεταφόρας δεδομένων και την σεριακή επικοινωνία. Το Serial.flush() περιμένει να σταλθούν τα δεδομένα που έχει μέσα του buffer επικοινωνίας του Serial over USB πρωτού την είσοδο στον βρόγχο.

Με το pinMode input αφήνουμε τις αντιστοιχες ψηφιακές υποδοχές του Arduino με αριθμο D2, D3, D4 και D5 να λειτουργήσουν ως εισόδοι. Το pinMode 6 λειτουργεί ως έξοδος και έχει συνδεθεί ένα led για την οπτική κατανόηση ότι το σύστημα δουλεύει. Το led ανάβει για μισό δευτερόλεπτο μέσω της τιμής 1 στην εντολή digitalWrite και κλείνει αντίστοιχα με το 0. Οι seed_lfsr θα εξηγήθούν μετέπειτα.

6.2.2 Συναρτήσεις

Η πρώτη συνάρτηση είναι η makenibble που δημιουργεί 4bit. Το delayMicroseconds είναι εντολή του IDE και καθυστερεί το πρόγραμμα για όσο χρόνο ορίσουμε.

```
57 byte makenibble() {
58   byte a = 0;
59   byte b = 0;
60   byte dly = 0;
61
62
63   dly = (PIND & 0b00111100) >> 2;
64   delayMicroseconds(dly + 2);
65
66
67   delayMicroseconds(10);
68   a = PIND & 0b00111100;
69   delayMicroseconds(10);
70   b = PIND & 0b00111100;
71   return (a ^ b);
72 }
```

Εικόνα 26: Συνάρτηση δημιουργίας 4bit.

ΤΟ PIND είναι ένα μεταβλητή του μικροελεγκτή η οποία παίρνει ως τιμή τις εισόδους των ψηφιακών υποδοχών από 0 εώς 7 (D0-D7). Το dly ουσιαστικά δημιουργείται από μία τυχαία τιμή που δέχεται το Arduino εκείνη την στιμγή και μετά γίνεται δεξία ολίσθηση κατά 2bit. Το dly αναπαραστείται από μία τιμή 4bit όπως καταλαβαίνουμε από την πράξη στην γραμμή 63. Μετά την ολίσθηση η μέγιστη τιμή καθυστέρησης που μπορεί να δεχτεί είναι το 15(1111) και η μικρότερη το 0 (0000).

Η τιμή του PIND αλλάζει συνεχώς από τις μετρήσεις των ψηφιακών υποδοχών D2, D3, D4 και D5. Διαλέγωντας μία τυχαία τιμή a και b με χρονοκαθυστέρηση, βγάζοντας την XOR τιμή τους αυξάνουμε την εντροπία.

Ο συνολικός χρόνος εκτέλεσης του ενός βρόγχου του προγράμματος κρίνεται με μία τυχαιότητα με πιθανές τιμές απο 22microsecond και 37microsecond επειδή δεν υπάρχουν άλλες καθυστερήσεις πέρα αυτές στην συνάρτηση makenibble.

Στην εικόνα 27 βλέπουμε την παραγωγή ενός αρχικού σπόρου για την τιμή της κάθε LFSR.

```
75 void seed_lfsr8() {
76   lfsr8 = 0;
77   do {
78     lfsr8 = lfsr8 | (makenibble() << 2);
79     lfsr8 = lfsr8 | (makenibble() >> 2);
80   } while (lfsr8 == 0);
81   initial_value8 = lfsr8;
82 }
83
84 void seed_lfsr16() {
85   lfsr16 = 0;
86   do {
87     lfsr16 = lfsr16 | makenibble();
88     lfsr16 = lfsr16 << 4;
89     lfsr16 = lfsr16 | makenibble();
90     lfsr16 = lfsr16 << 6;
91     lfsr16 = lfsr16 | (makenibble() << 2);
92     lfsr16 = lfsr16 | (makenibble() >> 2);
93   } while (lfsr16 == 0);
94   initial_value16 = lfsr16;
95 }
```

Εικόνα 27: Δημιουργία αρχικής τιμης lfsr μέσω της makenibble.

Η seedlfsr χρησιμοποιείται για να αρχικοποιηθεί η τιμή lfsr8 με τυχαίο θόρυβο από την makenibble. Η makenibble αποτελείται από 4bit. Εκτελούμε μία αριστερή ολίσθηση κατα δύο θέσεις και περνάμε την τιμή των 6bit στην lfsr8. Μετά εκτελώντας μία αριστερή ολίσθηση και bitwise OR με την προηγούμενη τιμή της lfsr8 δημιουργείται ένα byte. Αντίστοιχα η ίδια διαδικάσια συμβαίνει και στην lfsr16 με σκοπό να δημιουργηθούν 2byte με την χρήση τυχαίου θορύβου. Η συναρτήσεις τελειώνουν μόνο εάν οι μεταβλήτες δεν έχουν μηδενική τιμή, αλλίως επαναλαμβάνονται.

6.2.3 Loop

Εδώ είναι ο κώδικας βρόγχου που εκτελείται ασταμάτητα μέχρι να δημιουργηθεί το αρχειό δεδομένων τυχαίων bit, σύμφωνα με την ποσότητα που ζητήθηκε από τον προγραμματιστή.

```
27 void loop() {
28   for (int i = 0; i <= (buffersize - 1); i++) {
29     buffer[i] = (makenibble() >> 2);
30     buffer[i] = (buffer[i] | (makenibble() << 2));
31   }
32   for (int i = 0; i <= (buffersize - 1); i++) {
33     lfsr8 ^= lfsr8 >> 3;
34     lfsr8 ^= lfsr8 << 5;
35     lfsr8 ^= lfsr8 >> 7;
36     if (lfsr8 == initial_value8) {
37       seed_lfsr8();
38     }
39     buffer[i] = buffer[i] ^ lfsr8;
40   }
41   for (int i = 0; i <= (buffersize - 1); i++) {
42     lfsr16 ^= lfsr16 >> 7;
43     lfsr16 ^= lfsr16 << 9;
44     lfsr16 ^= lfsr16 >> 13;
45     if (lfsr16 == initial_value16) {
46       seed_lfsr16();
47     }
48     buffer[i] = buffer[i] ^ (lfsr16 & 0b11111111);
49   }
50
51   for (int i = 0; i <= (buffersize - 1); i++) {
52     Serial.write(buffer[i]);
53   }
54 }
```

Εικόνα 28: Post processing μέσω 2 LFSR, XOR με τον buffer και εξαγωγή στον υπολογιστή.

Στο πρώτο κομμάτι του βρόγχου ο κώδικας γεμίζει επανειλλημένα τον buffer μέσω της makenibble και εκτελεί μία 8bit-LFSR [23] στο lfsr8. Συγκρίνει την νέα τιμή του lfsr8 με την πάλια που πήρε από το initial_state8 και εάν είναι ίδιες ξαναεκτελεί την συνάρτηση seed_lfsr8. Έπειτα εκτελεί μία XOR μεταξύ του buffer και του lfsr8.

Στο δεύτερο κομμάτι γίνεται το ίδιο με το παραπάνω μέρος με την μόνη διαφορά ότι χρησιμοποείται 16bit-lfsr[23] και γίνεται XOR μόνο στα τελευταία 8 bit (όπως και με την lfsr8). Η εντολή πρόσθεσης bitwise AND μας βοηθάει να φιλτράρουμε τα τελευταία 8 bit όπως παρομοίως έγινε με την lfsr8

Πιο γενικευμένα η διαδικάσια σε έναν βρόγχο αφού έχουν γεμίσει οι LFSR με τυχαία δεδομένα είναι :

- Το γέμισμα του buffer με 64 byte τυχαίων δεδομένων
- Εκτέλεση XOR κάθε εισόδου του buffer με την 8bitLFSR
- Εκτέλεση XOR κάθε εισόδου του buffer με την 16bitLFSR
- Μεταφόρα buffer μέσω Serial-Over-USB

6.2.4 Λειτουργία μέσω εντολών

Για την λειτουργία του TRNG πρέπει να συνδέσουμε το Arduino στον υπολογιστή μέσω μίας θύρας USB2 Η USB3. Μετά το πέρασμα του κώδικα στο Arduino, πρέπει να προσαρμόσουμε την θύρα για την σωστή επικοινωνία με τον υπολογιστή. Αυτό γίνεται τρέχοντας στο τερματικό την εντολή :

```
stty raw -echo -ixoff -F /dev/ttyUSB0 speed 115200
```

Ο αριθμός της θύρας USB αλλάζει ανάλογα με το σε ποια από όλες συνδέσαμε το Arduino.

Για οποιάδηποτε εφαρμογή ή τεστ στην περίπτωση αυτή(Diehard) που θέλουμε να εγκαταστήσουμε εκτελούμε :

```
sudo apt update
sudo apt install dieharder
```

Για την εγγραφή δεδομένων 4GB σε ένα αρχείο εκτελούμε την εντολή :

```
head -c 4000000000 /dev/ttyUSB0 > /home/metaplita/trng/bin4gb
```

Για την απόρριψη φτωχών δεδομένων μέσω του rngtest εκτελούμε :

```
cat /dev/ttyUSB0 | rngtest -p > /home/metaplita/trng/bin4gb
```

Για την εκτελέση του ENT test :

```
cat /home/metaplita/trng/bin4gb | ent
```

Αντίστοιχα για το RNG test :

```
cat /home/metaplita/trng/bin4gb | rngtest
```

Για να ελέγξουμε το αρχείο μας με το Diehard εκτελούμε :

```
dieharder -g 201 -f /home/metaplita/trng/bin4gb
```

Συνίσταται όταν τεστάρουμε μικρά αρχεία να χρησιμοποιούμε το ENT και RNG τεστ. Εάν θέλουμε να ελέγξουμε με το Diehard θα ήταν σωστό να χρησιμοποιήσουμε αρχεία μεγαλύτερα των 500mb καθώς όσο λιγότερο το μέγεθος τόσα περισσότερα αποτυχημένα τεστ θα εμφανίζονται.

Ο κώδικας της εργασίας βρίσκεται στο GitHub.
URL (https://github.com/Ckefalopoulos/Project-Thesis)

7 Συμπεράσματα

Στο κεφάλαιο αυτό θα αναφέρουμε το τελικό αποτέλεσμα της διπλωματικής και πως αυτό ανταποκρίνεται στις αρχικές προσδοκίες μας. Επιπλέον , θα τεθούν προτάσεις για μελλοντική ανάπτυξη του σχεδίου με σκοπό την αύξηση της αποδόσης.

7.1 Σύνοψη και συμπεράσματα

Η διπλωματική εργασία αυτή είχε σκοπό την επεξήγηση διάφορων γεννητριών τυχαίων αριθμών και κυρίως την δημιουργία μιας πραγματικής γεννήτριας. Η παραγωγή τυχαίων αριθμών μπορεί να επιτευχθεί με διάφορους τρόπους που είτε αφορούν τη χρήση ενός κυκλώματος είτε την χρήση ενός αλγορίθμου που προσφέρει μία φαινομενική τυχαιότητα. Στην πρώτη περίπτωση υπάρχουν πολλαπλά φυσικά συστήματα από τα οποία μπορεί να αντληθεί εντροπία αλλά συνήθως τέτοια σχέδια ερχόνται με πολλές δυσκολίες. Τέτοιες είναι το κόστος ενέργειας, το περιβάλλον δέσμευσης και η μικρή πόσοτητα παραγωγής αριθμών σε σχέση με άλλα σχέδια(PRNG) που απαιτούν μόνο την χρήση ηλεκτρονικού υπολογιστή.

Το σχέδιο αυτής της διπλωματικής στην ουσία είναι η αξιοποίηση της εντροπίας ενός φαινομένου που δημιουργείται στους ηλεκτρονικούς ημιαγωγούς με σκοπό την συλλογή τυχαιότητας ως θορύβου. Μέσω της ανάστροφης πόλωσης ενός τρανζίστορ και το Avalanche effect που συμβαίνει στο emmiter-base του , είμαστε ικάνοι να παράγουμε πραγματικά τυχαίο θόρυβο. Με την χρήση ολοκληρωμένων και ενός Arduino Nano ο θόρυβος επεξεργάζεται και μεταφράζεται σε μία ακολουθία 0 και 1. Η δυσκολία της παραγωγής πραγματικά τυχαίων αριθμών έγκειται στον λόγο της ταχύτητας παραγωγής με την ποιότητα των ακολουθιών. Όσο πιο γρήγορα θα προσπαθούμε να παράγουμε τυχαίους αριθμούς τόσο πιο πολύ θα πέφτει και η τιμή της εντροπίας. Για αυτό και χρειάζεται αρκετή μελέτη με σκοπό να αξιοποιούμε την πηγή εντροπίας στο μέγιστο χωρίς να μεταβάλλεται η τυχαιότητα των αριθμών.

7.2 Μελλοντικές επεκτάσεις

Υπάρχουν αρκέτες λειτουργίες και παραμέτροι οι οποίες θα μπορούσαν να προστεθούν στο σχέδιο μελλοντικά. Κύριος στόχος για την βελτίωση του συστήματος είναι να γίνει η μεταφορά του κυκλώματος από το breadboard σε μία μητρική πλακέτα (stripboard) και ο εσωκλεισμός της σε ένα μεταλλικό πλαίσιο. Στο stripboard το κύκλωμα μικραίνει σε χώρο και επιτυγχάνεται η τέλεια επαφή μεταξύ των υλικών καθώς αυτά συγκολλούνται. Έτσι δεν θα υπαρχούν περιπτώσεις όπου η επαφή δεν θα είναι καλή. Το μεταλλικό πλαίσιο εγγυάται την αποβολή εξωτερικών πηγών θορύβου που μπορεί να μεταβάλλουν την αποδοτικότητα του σχεδίου. Τέτοια παραδείγματα έιναι η θερμοκρασία και η υγρασία. Επίσης η αγόρα ενός πιο αξιόπιστου τροφοδοτικού θα βοηθούσε στις μεταβολές τάσης.

Τέλος, στο κομμάτι του κώδικα θα ήταν πολύ σημαντική η προσθήκη ενός τεστ αυτοαξιολόγησης της τιμής των πηγών εντροπίας και της τροφοδοσίας. Με την μέτρηση της τάσης του τροφοδοτικού και τις τιμές παραγωγής των πηγών θορύβου σε ζωντανό χρόνο μπορούμε να εμφανίζουμε τους αντίστοιχους κωδικούς σφάλματος μέσω ενός led ή με την χρήση μηνυμάτων στο Serial.Monitor. Με αυτήν την λειτουργία θα έχουμε την δυνατότητα κάποιες δεδομένες χρονικές στιγμές που παρουσιάζεται σφάλμα να μην αποθηκεύουμε τιμές που παράγονται από την γεννήτρια.

Βιβλιογραφία

[1] Tippett, L. H. C. 1927. Random Sampling Numbers. Cambridge University Press.

[2] Kendall, M. G., and B. Babington-Smith. 1938. “Randomness and other random sampling numbers”. Journal of the Royal Statistical Society 101:146–166.

[3] Brown, G. W. 1949. “History of RAND’s Random Digits: Summary”. Technical Report P113, available at http://www.rand.org/pubs/papers/P113.html, RAND Corporation, Santa Monica, CA.

[4] J. von Neumann, “Various techniques used in connection with random digits,” J. Res. Natl. Bur. Stand., no. 3, pp. 36–38, 1951.

[5] Lehmer, Derrick H. 1951. "Mathematical methods in large-scale computing units". Proceedings of 2nd Symposium on Large-Scale Digital Calculating Machinery: 141–146.

[6] IBM, System/360 Scientific Subroutine Package, Version II, Programmer's Manual, H20-0205-1, 1967, p. 54.

[7] K. Yang, D. Blaauw, and D. Sylvester, “An All-Digital Edge Racing True Random Number Generator Robust Against PVT Variations,” in IEEE Journal of Solid-State Circuits, vol. 51, no. 4, pp. 1022-1031, April 2016

[8] H. Nyquist. Thermal agitation of electric charge in conductors. Phys. Rev., 32:110-113, 1928.

[9] “LavaRND,” www.lavarnd.org , 2000

[10] M. Stipcevic. Fast nondeterministic random bit generator based on weakly correlated physical events. Rev. Sci. Instrum., 75:4442-4449, 2004.

[11] GUDE, M. Concept for a high performance random number generator based on physical random phenomena. Frequenz 39, pp. 7-8 (1985), pp. 187–190.

[12] T. Jennewein, U. Achleitner, G. Weihs, H. Weinfurter, and A. Zeilinger, “A fast and compact quantum random number generator,” Rev. Sci. Instrum., vol. 71, p. 1675, 2000.

[13] J. Walker, “HotBits: A random number service,” www.fourmilab.ch/hotbits/.

[14] McKay, K. (1954). “Avalanche Breakdown in Silicon”. Physical Review , vol. 4, p.877-884.

[15] Roehr, William D. 1963, “High-speed switching transistor handbook”, pp. 286-313.

[16] P. Lacharme, 2008, "Post-processing functions for a biased physical random number generator", International Workshop on Fast Software Encryption, pp. 334-342.

[17] NIST Approved Algorithms for Secure Hashing, https://doi.org/10.6028/NIST.FIPS.180-4

[18] J. Walker, “ ENT : A Pseudorandom Number Sequence Test Program,”ww.fourmilab.ch/random/.

[19] A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications, Special publication 800- 22, May 2001, National Institute for Standards and Technology (NIST).

[20] Robert G. Brown, “ Dieharder : A Random Number Test Suite”, https://webhome.phy.duke.edu/~rgb/General/dieharder.php.

[21] R. J. Easter and C. French. Annex C: Approved Random Number Generators for FIPS PUB 140-2. Security Requirements for Cryptographic Modules. NIST, February 2012.

[22] Marsaglia, G. 1996. “DIEHARD: A Battery of Tests of Randomness”. https://web.archive.org/web/20160125103112/http://stat.fsu.edu/pub/diehard/

[23] Marsaglia, G. 2003. “Xorshift RNGs”. Journal of Statistical Software 8 (14): pp.1–6.