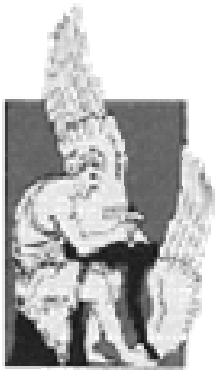


ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

Τμήμα Ηλεκτρονικών Μηχανικών και Μηχανικών Υπολογιστών

ΔΙΠΛΩΜΑΤΙΚΗ ΕΡΓΑΣΙΑ

Μελέτη Αξιοπιστίας Συστημάτων, Σχεδιασμός και Υλοποίηση Watchdog με Χρήση AVR για την Εποπτεία Συστήματος

Κώστας Γκρίτσης

Επιβλέπων Καθηγητής:
Καθηγητής Απόστολος Δόλλας

Εξεταστική επιτροπή:
Καθηγητής Απόστολος Δόλλας
Αναπληρωτής Καθηγητής Κων/νος Καλαϊτζάκης
Αναπληρωτής Καθηγητής Διονύσιος Πνευματικάτος

Αύγουστος 2003

Στη μνήμη του γιατρού Γιώργου Κτιστάκη.

Ευχαριστίες.

Θα ήθελα να ευχαριστήσω τον καθηγητή Απόστολο Δόλλα για την εμπιστοσύνη που μου έδειξε και για την πολύτιμη καθοδήγηση που μου παρείχε κατά την διάρκεια εκπόνησης της διπλωματικής μου εργασίας.

Επίσης θα ήθελα να ευχαριστήσω όλα τα μέλη του εργαστηρίου Μικροεπεξεργαστών και Υλικού: τον κο. Μάρκο Κιμιωνή για την φροντίδα και την βοήθεια, τους μεταπτυχιακούς και προπτυχιακούς φοιτητές για τις συμβουλές και την συμπαράσταση.

Τέλος θα ήθελα να ευχαριστήσω το backup tape autoloader της HP για το ασύγκριτο αίσθημα ασφάλειας που μου παρείχε όλα αυτά τα χρόνια…

Περιεχόμενα

1. Εισαγωγή.

1.1 Η σημασία της αξιοπιστίας των υπολογιστικών συστημάτων.

Ο ραγδαίος ρυθμός αύξησης της χρήσης των υπηρεσιών της πληροφορικής σε κάθε ανθρώπινη δραστηριότητα αποτελεί μια ιστορική πραγματικότητα η οποία συμπληρώνει πάνω από πέντε δεκαετίες εξελίξεων. Ταυτόχρονα, όλο και περισσότερα συστήματα των οποίων οι λειτουργίες και οι υπηρεσίες είναι κρίσιμες για την ασφάλεια αλλά και την πρόοδο του ανθρώπινου πολιτισμού, συμπεριλαμβάνουν ένα υποσύστημα πληροφορικής.

Έτσι οι καθημερινές λειτουργίες που συνδέονται με την ανθρώπινη ζωή αλλά ταυτόχρονα εμπλέκονται και με την χρήση ενός απλού έως εξαιρετικά σύνθετου υπολογιστικού συστήματος καθημερινά πολλαπλασιάζονται. Λειτουργίες που έχουν σχέση με την σταθεροποίηση των όρων της ασφάλειας είτε αυτή αφορά την εύρυθμη λειτουργία ενός πυρηνικού αντιδραστήρα, είτε την σωστή εναλλαγή των κυκλοφοριακών ρυθμίσεων σε έναν αυτοκινητόδρομο, είτε την προστασία των προσωπικών δεδομένων των πολιτών σε μια ηλεκτρονική συναλλαγή αποτελούν χαρακτηριστικά παραδείγματα περιπτώσεων όπου η αξιοπιστία των συστημάτων πληροφορικής είναι κρίσιμης σημασίας.

Πέραν όμως του θεμελιώδους αιτήματος για ασφάλεια, η άρρηκτη σύνδεση των οικονομικών και των διοικητικών λειτουργιών μιας επιχείρησης ή ενός οργανισμού με την υποδομή πληροφορικής του, αλλά ακόμη και η απ' ευθείας εξάρτηση της παραγωγικότητας ενός ατόμου με τον προσωπικό υπολογιστή του, δίνει την δυνατότητα να διατυπωθεί ο ισχυρισμός πως ο δείκτης αξιοπιστίας που χαρακτηρίζει τα πληροφοριακά συστήματά, τελικά επηρεάζει άμεσα τον ρυθμό ανάπτυξης, οικονομικής και πολιτιστικής της σύγχρονης κοινωνίας.

Η διείσδυση των υπολογιστικών συστημάτων στην καθημερινότητα μπορεί να προβληθεί με παραστατικό τρόπο από την υπηρεσία του ηλεκτρονικού ταχυδρομείου. Το ηλεκτρονικό ταχυδρομείο πλέον αποτελεί αναγκαία καθημερινή υπηρεσία τόσο

για τις ανάγκες επικοινωνίας στο επάγγελμα όσο και στις διαπροσωπικές σχέσεις. Χαρακτηριστικό είναι ότι σε έρευνα που έγινε μεταξύ των υπαλλήλων εταιριών [1] διαπιστώθηκε πως το 20% των ατόμων δυσανασχετεί αμέσως μόλις διαπιστώσει πως δεν λειτουργεί η υπηρεσία ηλεκτρονικού ταχυδρομείου, ενώ μια ολόκληρη ώρα σφάλματος στη λειτουργία της υπηρεσίας θα προκαλέσει την έντονη διαμαρτυρία του 80% του προσωπικού.

Επιβεβαιώνεται λοιπόν αυτό που λογικά προβλεπόταν από τον ρυθμό διείσδυσης των υπολογιστικών συστημάτων στην καθημερινή ζωή, ότι δηλαδή το αίτημα για αξιοπιστία και σταθερή λειτουργία των συστημάτων πληροφορικής δεν περιορίζεται πλέον μόνο σε εξειδικευμένες εφαρμογές και ειδικού τύπου συστήματα (dedicated systems) αλλά προβάλει ως έντονο πρόβλημα ακόμη και στις περιπτώσεις όπου τα «συνηθισμένα» συστήματα οικιακών υπολογιστών χρησιμοποιούνται για την πρόσβαση στην υπηρεσία του ηλεκτρονικού ταχυδρομείου.

1.2 Η αξιοπιστία των υπολογιστικών συστημάτων.

Η επιστήμη της αξιοπιστίας, της οποίας τα βασικά εργαλεία διαμορφώθηκαν στους χώρους της επιστήμης υλικών και της αξιοπιστίας των ηλεκτρονικών εξαρτημάτων καλείται πλέον να επαναδιατυπώσει την θεωρία της και να επανεξετάσει τις μεθοδολογίες που χρησιμοποιεί αφού η αλματώδης ανάπτυξη της πληροφορικής εισήγαγε νέες προκλήσεις στο πεδίο δράσης της.

Η ακριβής εκτίμηση της αξιοπιστίας των σύγχρονων υπολογιστικών συστημάτων είναι εξαιρετικά δύσκολη διαδικασία για δυο βασικούς λόγους. Ο πρώτος λόγος είναι ο μεγάλος βαθμός αλληλεξάρτησης μεταξύ του υλικού και του λογισμικού, τα οποία πλέον συναντώνται πάντοτε να αλληλεπιδρούν σε ένα ολοκληρωμένο σύστημα πληροφορικής. Ο δεύτερος λόγος είναι πως τα υποσυστήματα λογισμικού εξαιτίας της δυναμικής τους φύσης δεν επιδέχονται εύκολα την μαθηματική ανάλυση που απαιτείται για την διατύπωση ενός μοντέλου που θα περιγράφει την συμπεριφορά τους όσον αφορά την αξιοπιστία τους.

Κατά την προσπάθεια για την υλοποίηση ενός αξιόπιστου σύστηματος, είναι προφανής η υπόθεση πως οι δυο βασικοί άξονες εργασίας προς τους οποίους θα πρέπει να κατευθυνθεί ο μηχανικός είναι η επίτευξη αυξημένης αξιοπιστίας στο φυσικά χαρακτηριστικά και την ποιότητα του υλικού (hardware) και επιπλέον ο κατά το δυνατόν εκτενέστερος έλεγχος της ορθής λειτουργίας του λογισμικού (software). Οι λεπτομέρειες στην εφαρμογή των γενικών αρχών των δυο παραπάνω αξόνων θα περιγραφούν στη συνέχεια της παρούσας εργασίας, όμως θα πρέπει ευθύς εξαρχής να επισημανθεί στον αναγνώστη μια ιδιαίτερα σημαντική διαπίστωση. Υπάρχει ένας τρίτος άξονας εργασίας, ο οποίος είναι ισοδύναμος με τους δυο προηγούμενους. Πρόκειται για την ολοκλήρωση του υλικού με το λογισμικό. Η διεπαφή και η εύρυθμη συνεργασία του υλικού με το λογισμικό όχι μόνο αποτελούν έναν πολύ σημαντικό παράγοντα της αξιοπιστίας ενός συστήματος πληροφορικής αλλά επιπλέον όπως καταδεικνύει η εμπειρία, τα λάθη που εμφανίζονται σε αυτή την διεπαφή αποτελούν μια από τις κύριες αιτίες για την δυσλειτουργία ή το ολικό σφάλμα λειτουργίας ενός υπολογιστικού συστήματος.

1.3 Σκοπός της Διπλωματικής Εργασίας

Η παρούσα διπλωματική εργασία βασίζεται στις βασικές αρχές της επιστήμης της αξιοπιστίας ώστε να μελετηθεί και να κατασκευαστεί ένα σύστημα το οποίο αυξάνει την αξιοπιστία του προσωπικού υπολογιστή. Το ζητούμενο της αύξησης της αξιοπιστίας του προσωπικού υπολογιστή είναι ιδιαίτερα σημαντικό αφού οι σύγχρονες τάσεις της πληροφορικής τείνουν να απομακρυνθούν από τα συμπαγή συστήματα του υπερυπολογιστή (super computer). Αντιθέτως τα διανεμημένα συστήματα που αποτελούνται από πολλούς προσωπικούς υπολογιστές (network of PC’s) προβάλλονται συνέχεια ως η νέα εναλλακτική λύση για την επίτευξη υπολογιστικής ισχύος και αξιοπιστίας. Ο πλέον επικρατών εκπρόσωπος αυτής της τάσης είναι το grid computing, το οποίο είναι ένας τύπος παράλληλου και κατανεμημένου συστήματος το οποίο επιτρέπει την κοινή χρήση, επιλογή και συσσώρευση γεωγραφικά κατανεμημένων, αυτόνομων πόρων με τρόπο δυναμικό, κατά την διάρκεια του χρόνου εκτέλεσης και με ικανοποίηση περιορισμών όπως η διαθεσιμότητα, η απόδοση, το κόστος, και η ποιότητα υπηρεσιών.

Στην πρόταση που διατυπώνεται στην παρούσα διπλωματική, η βασική ιδέα είναι ότι ο προσωπικός υπολογιστής ελέγχεται από μία χαμηλού κόστους, ανεξάρτητη συσκευή με τρόπο ο οποίος διασφαλίζει την λειτουργικότητα και αξιοπιστία του συνολικού συστήματος. Η προτεινόμενη λύση θα τεκμηριωθεί πλήρως μέσα από τη μελέτη του προβλήματος αλλά και θα παρουσιασθεί σαν υλοποιημένο σύστημα.

Έτσι, η συνεισφορά της εργασίας αυτής τελικά δεν περιορίζεται στο ούτως ή άλλως αρκούντως σημαντικό επίπεδο του αναστοχασμού της προβληματικής της αξιοπιστίας του προσωπικού υπολογιστή σε σχέση με όλες τις σύγχρονες συνθήκες που διαμορφώνονται από τις τεχνολογίες και τα προϊόντα που περιβάλλουν το PC σήμερα, αλλά επιπλέον προχωρά στην υλοποίηση ενός συστήματος που συμβαδίζει και συνεργάζεται με τις τρέχουσες τεχνολογίες υλικού και λειτουργικού. Η λειτουργία του συστήματος που προτείνεται προσφέρει αύξηση της αξιοπιστίας μέσω της παρακολούθησης της ορθής λειτουργίας του προσωπικού υπολογιστή και της δυναμικής παρέμβασης για την διόρθωση προβλημάτων ή την ειδοποίηση για την εμφάνισή τους.

1.4 Δομή της διπλωματικής εργασίας.

Στο δεύτερο κεφάλαιο της διπλωματικής γίνεται μια εκτενής αναφορά στις τεχνολογίες γύρω από την αξιοπιστία πληροφοριακών συστημάτων. Φυσική συνέπεια της αναφοράς αυτής είναι η παράταξη συμπερασμάτων τα οποία οδηγούν στην σύνταξη της μεθοδολογίας, η οποία χρησιμοποιείται για να σχεδιαστεί το υπό υλοποίηση σύστημα. Με βάση αυτή ακριβώς τη μεθοδολογία, στο τρίτο κεφάλαιο γίνεται η μοντελοποίηση του συστήματος καθώς και η θεωρητική τεκμηρίωση των σχεδιαστικών επιλογών. Στο τέταρτο κεφάλαιο παρουσιάζεται η αρχιτεκτονική του συστήματος ενώ στο πέμπτο κεφάλαιο παρουσιάζεται η υλοποίηση του. Στο έκτο κεφάλαιο παρουσιάζονται τα αποτελέσματα από τα διάφορα πειράματα και σενάρια χρήσης του συστήματος. Τέλος στο έβδομο κεφάλαιο παρουσιάζονται τα συμπεράσματά της εργασίας και οι μελλοντικές επεκτάσεις που μπορεί να έχει το σύστημα.

2. Αξιοπιστία Υπολογιστικών Συστημάτων.

2.1 Εισαγωγή.

Η αξιοπιστία εν γένει είναι ένα πρόβλημα, του οποίου η λύση πάντοτε περιλαμβάνει δόσεις θεωρίας και τεχνικής, αφού σε αντίθεση με ίσως άλλα επιστημονικά προβλήματα η αξιοπιστία μετριέται σε πραγματικές συνθήκες. Αυτό σημαίνει πως ένα μοντέλο το οποίο σχεδιάζεται στο χαρτί και κάτω από τις ιδανικές συνθήκες, στον πραγματικό κόσμο θα πρέπει να αντεπεξέλθει σε αστάθμητους παράγοντες, οι οποίοι ενδεχόμενα δεν συνυπολογίστηκαν.

Έτσι δεν είναι δύσκολο να γίνει σαφές πως από την στιγμή που είναι αρκετά επίπονο να βρεθούν μηχανισμοί για την διασφάλιση της ποιότητας απλών μηχανικών εξαρτημάτων όπως π.χ. οι βίδες ή τα μπουλόνια, αφού πρέπει να συμπεριληφθούν πολλοί και διαφορετικοί παράγοντες πέραν της κατασκευαστικής διαδικασίας όπως το φορτίο χρήσης και οι συνθήκες αποθήκευσης, ακόμη περισσότερο στον ισχυρά δυναμικό κόσμο των υπολογιστικών συστημάτων, η δυσκολία αυτή πολλαπλασιάζεται. Ένα εργαλείο (όπως π.χ. ένα σύστημα ανύψωσης) μπορεί να αποτύχει λόγω αστοχίας στα υλικά του, λόγω κακών χειρισμών από τον άνθρωπο – χειριστή, λόγω παραβίασης των προδιαγραφών λειτουργίας (προσπάθεια ανύψωσης μεγαλύτερου βάρους από το επιτρεπτό), λόγω κακών συνθηκών λειτουργίας ή τέλος λόγω πλημμελούς συντήρησης. Ένα υπολογιστικό σύστημα μπορεί να αποτύχει για τους ίδιους ακριβώς λόγους –εάν είναι αποδεκτή η άποψη πως τα σφάλματα του λογισμικού δεν είναι τυχαία αλλά αποτελούν προβλήματα που εισήχθηκαν στην διαδικασία σχεδίασης- με μία ειδοποιό διαφορά. Τα υποσύστημα τα οποία αποτελούν τον υπολογιστή δεν έχουν σταθερή αλλά δυναμική σύνδεση μεταξύ τους. Αυτή η δυναμική σύνδεση και τα προβλήματα που δημιουργούνται στην αναεγκαθίδρυση της εισάγουν άλλον ένα βαθμό ελευθερίας στους τρόπους που μπορεί να οδηγηθεί στην αποτυχία ένα σύστημα υλικού / λογισμικού.

Η παραπάνω παρατήρηση οδηγεί συνεπώς σε μια διευρυμένη μεθοδολογική προσέγγιση. Η αξιοπιστία των υπολογιστικών συστημάτων για να διατυπωθεί με

συνέπεια πέραν από τις αναφορές της στις βασικές αρχές της κλασικής επιστήμης της αξιοπιστίας, θα πρέπει να εμπλουτιστεί με ιδέες και τεχνικές από τους σχεδιαστές υλικού και λογισμικού, ώστε να μπορούν να αντιμετωπιστούν στην πληρότητα τους οι νέες απαιτήσεις. Μάλιστα, μετατρέποντας το μειονέκτημα σε πλεονέκτημα, η διαπίστωση πως η δυναμική φύση των συστημάτων πληροφορικής προκαλεί τον υψηλό βαθμό αναξιοπιστίας των τελευταίων, οδηγεί στην αναγκαιότητα για την επινόηση αντίστοιχα δυναμικών μηχανισμών, οι οποίοι με δυναμικές παρεμβάσεις προσπαθούν να αυξήσουν την αξιοπιστία των συστημάτων που εποπτεύουν.

Εν συνεχεία θα παρατεθούν τα αναγκαία θεωρητικά στοιχεία από την θεωρία της αξιοπιστίας τα οποία χρησιμοποιούνται στον σχεδιασμό και την υλοποίηση του συστήματος που περιγράφεται στα πλαίσια της παρούσας διπλωματικής εργασίας. Ωστόσο είναι προφανές πως δεν επιχειρείται η πλήρης κάλυψη του θέματος της αξιοπιστίας, αφού αυτό θα ήταν ανέφικτο ούτως ή αλλιώς στα πλαίσια μιας διπλωματικής εργασίας, αλλά μόνο τα στοιχεία εκείνα που οδηγούν στην πλήρη τεκμηρίωση των σχεδιαστικών επιλογών και των μεθόδων υλοποίησης.

2.2 Μαθηματικοί Ορισμοί

Ο βασικός ορισμός της αξιοπιστίας ενός συστήματος απαιτεί να οριστεί η έννοια της αστοχίας.

Ορίζεται ότι ένα σύστημα **παρουσιάζει βλάβη** όταν εμφανίζει ένα πρόβλημα στη λειτουργία του έτσι ώστε η αντιμετώπιση του προβλήματος είναι αδύνατη ή χωρίς αξία για την επίτευξη του αρχικού σκοπού του συστήματος.

2.2.1 Αξιοπιστία

Σύμφωνα με τα παραπάνω ορίζεται **αξιοπιστία – R(t)** ενός συστήματος η πιθανότητα να μην εμφανίζει βλάβη στον χρόνο t:

$$P(\text{survival up to time t})=P(T>t) \equiv R(t)$$
$$P(\text{Ορθή λειτουργία μέχρι τον χρόνο t})=P(T>t) \equiv R(t) \quad (2.1)$$

Το t είναι ο χρόνος, το T είναι ο χρόνος που εμφανίζεται η αποτυχία ενώ τα P(T>t) και R(t) είναι πιθανότητες. Προφανώς όταν $t \to \infty$, $R(t) \to 0$ αφού η πιθανότητα της αποτυχίας αυξάνει με τον χρόνο λειτουργίας.

2.2.2 Αναξιοπιστία

Επιπλέον

$$P(\text{failure at t})=P(T \le t) \equiv Q(t)$$
$$P(\text{βλάβη στον χρόνο t})=P(T \le t) \equiv Q(t) \quad (2.2)$$

όπου Q(t) είναι η **συνάρτηση αναξιοπιστίας.**

Το t είναι ο χρόνος, το T είναι ο χρόνος που εμφανίζεται η αποτυχία ενώ το P(T<t) και Q(t) είναι πιθανότητες.

2.2.3 Πυκνότητα βλαβών.

Από τον ορισμό της συνάρτησης κατανομής μιας συνεχούς τυχαίας μεταβλητής, είναι προφανές ότι η Q(t) είναι πράγματι η συνάρτηση κατανομής για το T. Έτσι, η **συνάρτηση πυκνότητας βλαβών** f(t) μπορεί να υπολογισθεί:

$$f(t) = \frac{d}{dt} Q(t) \quad (2.3)$$

2.2.4 Ρυθμός βλαβών.

Η συνάρτηση **ρυθμός βλαβών** ορίζεται ως:

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{1}{\Delta t} \left[\text{probability of failure in } (t,t+\Delta t), \text{ given survival up to t}\right] \quad (2.4)$$

$$\lambda(t) = \lim_{\Delta t \to 0} \frac{1}{\Delta t} \left[\text{πιθανότητα βλάβης μέσα στο } (t,t+\Delta t), \text{ με δεδομένη την ορθή λειτουργία μέχρι τον χρόνο t}\right]$$

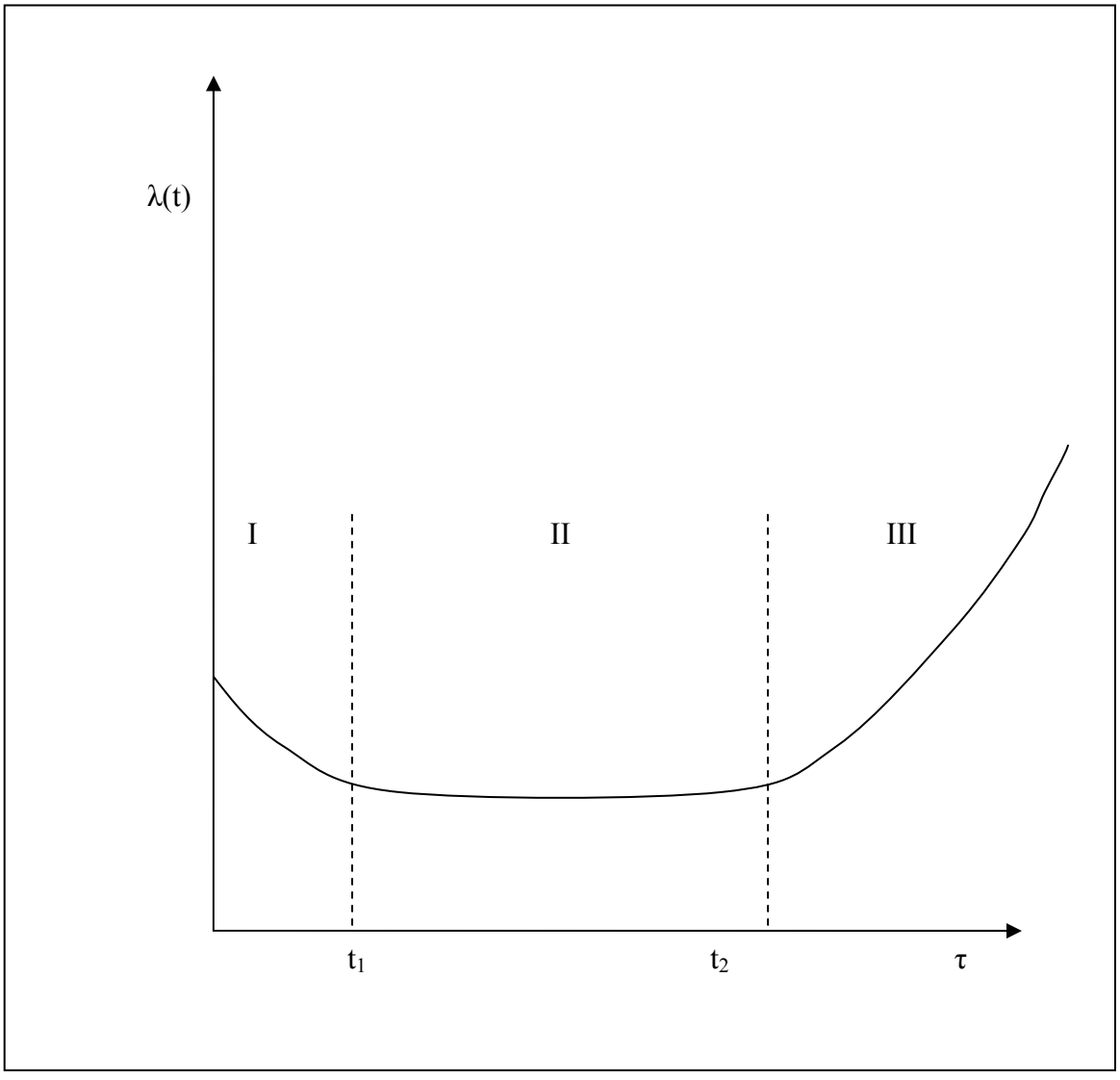
ενώ μπορεί να αποδειχθεί ότι:

$$\lambda(t) = \frac{f(t)}{R(t)} \tag{2.5}$$

Οι τέσσερις παραπάνω συναρτήσεις αποτελούν την ομάδα των βασικών συναρτήσεων που χρησιμοποιούνται στην ανάλυση αξιοπιστίας.

2.3 Καμπύλη “bathtub”.

Για ένα αρκετά μεγάλο και ομοιογενή πληθυσμό εξαρτημάτων ή συστημάτων, η πράξη έχει δείξει ότι ισχύει η μορφή της καμπύλης που παρουσιάζεται στο γράφημα 1 για την συνάρτηση ρυθμού βλαβών.



Γράφημα 1. Καμπύλη bathtub (Πηγή: [23])

Η συγκεκριμένη καμπύλη αναπαριστά τα διάφορα στάδια κατά την διάρκεια ζωής ενός συστήματος. Η πρώτη περιοχή (Ι) αντιστοιχεί στις πρώιμες βλάβες κατά την αρχική αποσφαλμάτωση (αλλιώς γνωστές και ως παιδική θνησιμότητα). Ο ρυθμός βλαβών μειώνεται όσο η αποσφαλμάτωση συνεχίζεται μέχρι που το σύστημα μπαίνει στην δεύτερη και ώριμη πλέον φάση της ζωής του με χαμηλότερο ρυθμό βλαβών και με σφάλματα που μπορούν να χαρακτηρισθούν τυχαία. Πρόκειται για την χρήσιμη περιοχή της ζωής του συστήματος. Τέλος, η τρίτη περιοχή αφορά την περιοχή κόπωσης του συστήματος όπου εμφανίζεται πλέον μια απότομη και σταθερή αύξηση στον ρυθμό βλαβών.

2.4 Αξιοπιστία των υπολογιστικών συστημάτων.

2.4.1 Εισαγωγικά στοιχεία

Οι βασικοί ορισμοί που μόλις παρουσιάστηκαν οφείλουν την ύπαρξή τους στον τομέα της επιστήμης της αξιοπιστίας που αναπτύχθηκε κυρίως γύρω από την επιστήμη των υλικών. Ο τομέας αυτός της αξιοπιστίας προσπαθεί κυρίως να μελετήσει την φυσική συμπεριφορά των υλικών, των εξαρτημάτων τα οποία κατασκευάζονται με τα υλικά και εν τέλει των συστημάτων, τα οποία αποτελούν διασύνδεση των εξαρτημάτων. Η μελέτη της φυσικής συμπεριφοράς χρησιμοποιεί εργαλεία κυρίως από την αναλυτική φυσική στην προσπάθειά της να συνάγει συμπεράσματα για την λειτουργική συμπεριφορά ενός συστήματος κάτω από ορισμένες συνθήκες (θερμοκρασία, υγρασία, ακτινοβολία, φόρτος εργασίας, κλπ.). Αναπτύσσονται μοντέλα όχι μόνο για την αξιοπιστία των εξαρτημάτων αλλά και για την αξιοπιστία που εμφανίζουν διαφορετικές συνδεσμολογίες μεταξύ των υλικών.

Παρόλο που το κομμάτι του υλικού ενός υπολογιστικού συστήματος ανήκει στην παραπάνω ταξινόμηση και συνεπώς μπορεί να επιδεχθεί την ανάλυση με το παραπάνω μοντέλο, τα υπολογιστικά συστήματα περιλαμβάνουν και το κομμάτι του λογισμικού, το οποίο όμως είναι αδύνατον να ενταχθεί στο παραπάνω μοντέλο για δυο λόγους.

Αφενός, αυτό είναι ανέφικτο εξαιτίας της δυναμικής φύσης των συστημάτων λογισμικού [3]. Παρόλο που το λογισμικό διαθέτει την ίδια εσωτερική ιεραρχική ταξινόμηση, δηλαδή αποτελείται από εξαρτήματα (modules), δηλαδή από υποσυστήματα που διασυνδέονται μεταξύ τους και εμφανίζονται ως συστήματα, η σύνθεση αυτών των συστημάτων είναι απολύτως δυναμική, σε αντίθεση με τα υλικά συστήματα των οποίων τα υποσυστήματα συνδέονται σε μια στατική τοπολογία. Έτσι στην πρώτη περίπτωση δεν είναι δυνατόν να γίνει μια καθολική μαθηματική μελέτη της τοπολογίας και σε αντίθεση με την δεύτερη περίπτωση, δεν μπορούν να εξαχθούν συμπεράσματα για την αξιοπιστία του συστήματος. Τουναντίον είναι τυπικό κατά την διάρκεια λειτουργίας ενός ηλεκτρονικού υπολογιστή να ενεργοποιηθούν και να τερματίσουν δεκάδες έως χιλιάδες διαφορετικές εφαρμογές (δηλαδή υποσυστήματα) χρησιμοποιώντας κοινούς πόρους (επεξεργαστής, μνήμη, περιφερειακές μονάδες, λειτουργικό σύστημα).

Επιπλέον όμως, στην κοινότητα της επιστήμης της αξιοπιστίας υπάρχει ακόμη ανοιχτό το θέμα σχετικά με το κατά πόσο υπάρχει η έννοια της βλάβης στα συστήματα λογισμικού. Συγκεκριμένα, προβάλλεται η θέση πως εφόσον το λογισμικό είναι ένα λογικό κατασκεύασμα, δηλαδή δεν έχει φυσική υπόσταση, τότε δεν τίθεται θέμα φυσικής βλάβης. Η όποια αστοχία εμφανίζει το λογισμικό κατά την λειτουργία του δεν είναι τίποτα άλλο παρά ένα λάθος που έγινε κατά την διάρκεια του σχεδιασμού και της καταγραφής των προδιαγραφών του.

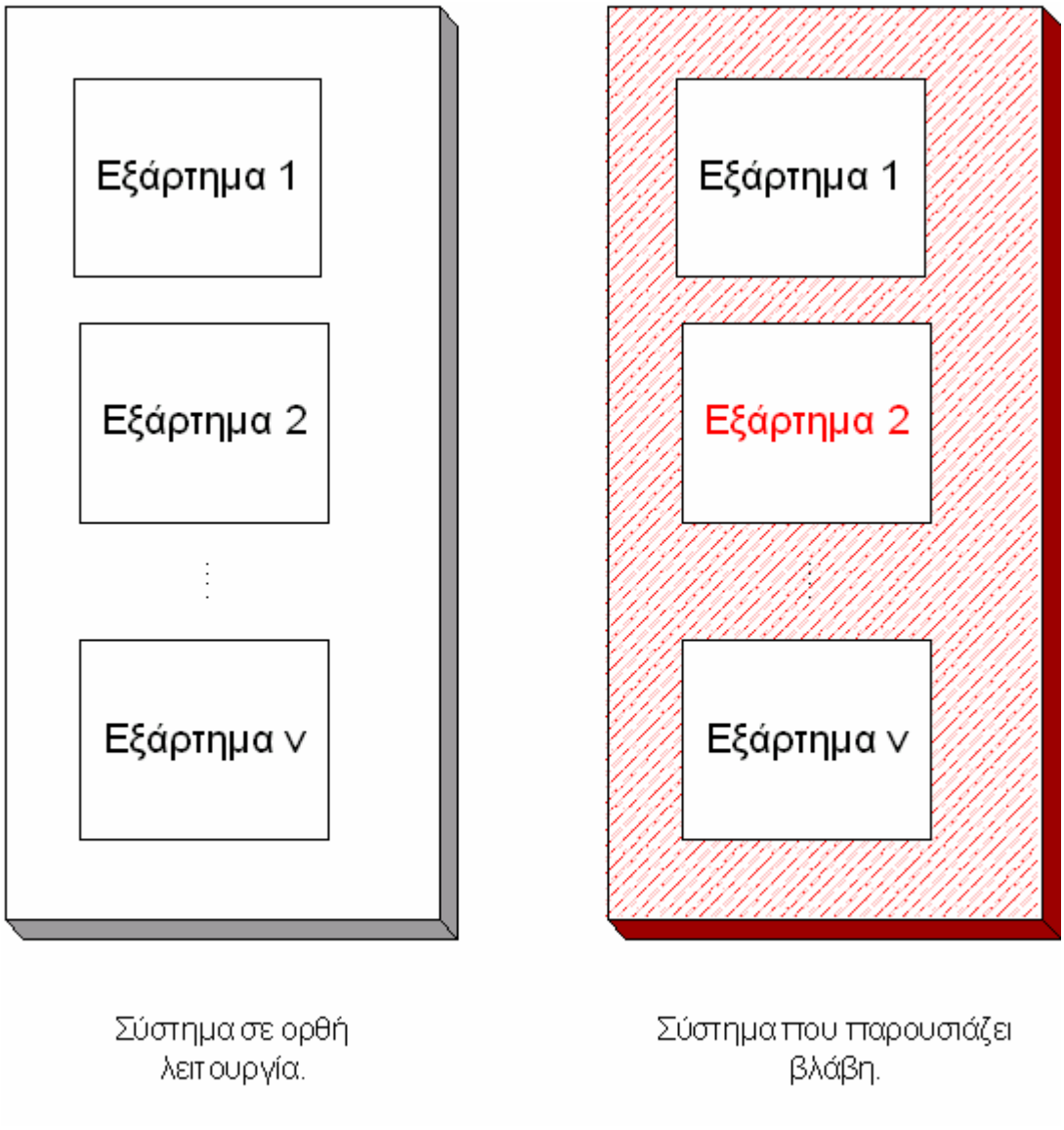
Στην προσπάθεια της επαναδιατύπωσης της θεωρίας της αξιοπιστίας, με σκοπό την αντιμετώπιση του παραπάνω προβλήματος, είναι προφανές πως η κατάληξη θα είναι ένα περισσότερο περιγραφικό παρά μαθηματικά αυστηρό μοντέλο. Κάτω από το πρίσμα των παραπάνω διατυπώσεων επιχειρείται η παρακάτω ταξινόμηση των αποτυχιών, προσανατολισμένη προς τα υπολογιστικά συστήματα.

Σφάλματα Αποτυχίας και Σχεδιαστικά Σφάλματα.

Μια αποτυχία εμφανίζεται όταν ένα σύστημα ή ένα εξάρτημα του δεν αποδίδει όπως αναμενόταν. Ένα παράδειγμα στο επίπεδο των εξαρτημάτων θα μπορούσε να αποτελέσει ένα βραχυκύκλωμα μεταξύ βάσης – εκπομπού μέσα σε ένα μεγάλο

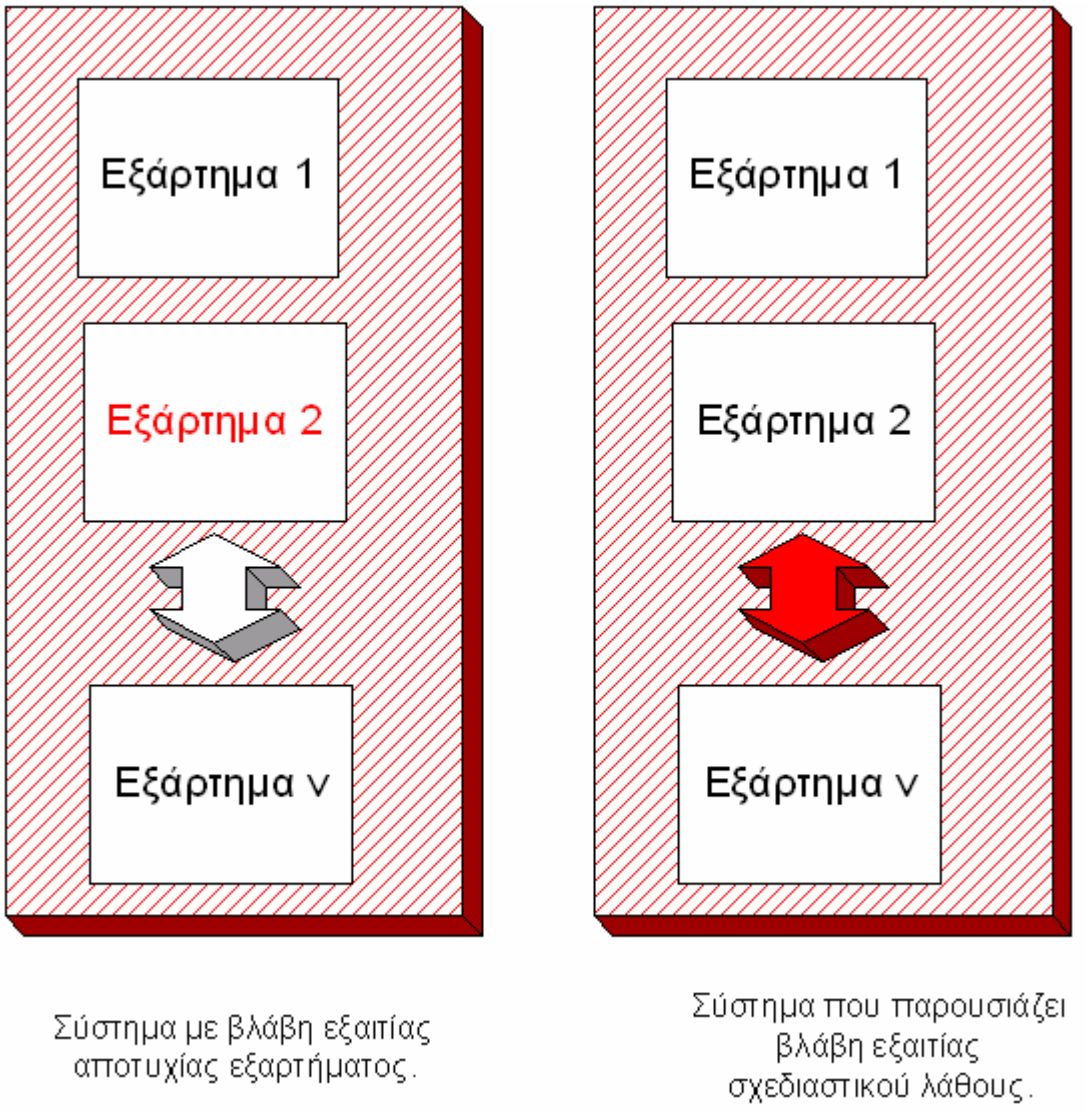
ολοκληρωμένο κύκλωμα ή μια ψυχρή κόλληση (δηλαδή μια κόλληση η οποία ανοίγει το κύκλωμα εξαιτίας μηχανικών κραδασμών).

Μία βλάβη μπορεί από την άλλη να είναι η εκδήλωση μιας αποτυχίας ενός εξαρτήματος ή μιας σχεδιαστικής αποτυχίας. Όπως φαίνεται στο γράφημα 2 η αποτυχία του εξαρτήματος 2 (το οποίο σημαίνουμε με κόκκινο χρώμα) έχει ως αποτέλεσμα να εμφανίζει βλάβη το σύστημα). Οι αποτυχίες των εξαρτημάτων μπορούν να προκληθούν είτε από εσωτερικά φυσικά φαινόμενα είτε από εξωτερικές περιβαλλοντικές επιδράσεις όπως π.χ. τα ηλεκτρομαγνητικά πεδία ή οι μεταβολές στην τροφοδοσία.



Γράφημα 2. Έκφραση αποτυχίας ενός εξαρτήματος ως σφάλμα του συστήματος.

Τα σχεδιαστικά σφάλματα μπορούν να χωριστούν σε δυο κλάσεις ή κατηγορίες. Η πρώτη κατηγορία των σχεδιαστικών σφαλμάτων προκαλείται από την χρήση εξαρτημάτων έξω από το πεδίο των προδιαγραφών τους. Προφανώς αυτά τα σφάλματα μπορούν να αποφευχθούν με τον προσεχτικό επανέλεγχο του σχεδίου. Στο γράφημα 3 παρατηρούμε πως το σύστημα παρουσιάζει βλάβη επειδή ακριβώς ένα από τα εξαρτήματα χρησιμοποιείται εκτός προδιαγραφών. Η δεύτερη κατηγορία αφορά τις περιπτώσεις όπου ο σχεδιαστής δεν λαμβάνει υπόψη του όλες τις λογικές καταστάσεις που μπορεί να βρεθεί το σύστημα κατά την διάρκεια της λειτουργίας του. Στο γράφημα 3 φαίνεται πως παρόλο που όλα τα εξαρτήματα του δεύτερου συστήματος λειτουργούν κανονικά, οι αλληλεπιδράσεις μεταξύ τους είναι λανθασμένες (κόκκινο αμφίδρομο βέλος).



Γράφημα 3. Κατηγορίες σχεδιαστικών λαθών.

Σε αυτού του είδους τα σφάλματα κατατάσσονται και τα σφάλματα λογισμικού. Συνεπώς, επειδή κάθε υπολογιστής συμπεριλαμβάνει λογισμικό στα υποσυστήματά του, αυτού του είδους τα σφάλματα αποτελούν έναν από τους σημαντικότερους παράγοντες για την αξιοπιστία των υπολογιστικών συστημάτων.

Πρέπει να σημειωθεί πως εδώ ακριβώς βρίσκεται η λεπτή διαχωριστή γραμμή των δυο διαφορετικών χώρων στους οποίους μπορεί να υποβόσκει η πιθανότητα σφάλματος για ένα υπολογιστικό σύστημα δηλαδή λογικό (σχεδιαστικό λάθος) ή αποτυχία υλικού.

Μόνιμα και Προσωρινά Σφάλματα.

Η σχέση των σφαλμάτων με τον χρόνο περιγράφεται από την διάκριση τους στα μόνιμα και στα προσωρινά. Τα μόνιμα σφάλματα θα μπορούσαν να είναι π.χ. βραχυκυκλώματα ή ανοιχτά κυκλώματα μέσα σε ένα εξάρτημα εξαιτίας φυσικών αποτυχιών.

Πίνακας 1. Κατηγορίες σφαλμάτων σε σχέση με την περιοδικότητα εμφάνισης.

<table>
<tr><td>Προσωρινά Σφάλματα</td><td><table><tr><td>Παροδικά</td><td>Διακύμανση της τροφοδοσίας</td></tr><tr><td>Διακεκομμένα</td><td>Χαλαρές συνδέσεις στα εξαρτήματα</td></tr></table></td></tr>
<tr><td>Μόνιμα Σφάλματα</td><td>Βραχυκυκλώματα / Ανοιχτά κυκλώματα.</td></tr>
</table>

Τα προσωρινά σφάλματα πάλι χωρίζονται σε παροδικά όπως π.χ. αυτά που προκαλούνται από τον βομβαρδισμό με σωματίδια άλφα ή από τις διακυμάνσεις της τροφοδοσίας. Τέτοια προσωρινά σφάλματα εμφανίζονται συχνά στις μνήμες. Ένας πιο τυπικός ορισμός αυτών των σφαλμάτων είναι πως δεν προκαλούνται από φυσική ζημιά στο υλικό του συστήματος. Η δεύτερη κατηγορία προσωρινών σφαλμάτων ονομάζονται διακεκομμένα σφάλματα. Αυτά είναι μεν προσωρινά αλλά επανεμφανίζονται με μη προβλέψιμο τρόπο. Προκαλούνται συνήθως από χαλαρές

συνδέσεις των υποσυστημάτων ή από την χρήση εξαρτημάτων στο όριο των προδιαγραφών τους. Συχνά τα διακεκομμένα σφάλματα μετά από πολλές επανεμφανίσεις μετατρέπονται σε μόνιμα.

Ενεργά και Μη Ενεργά Σφάλματα.

Τα σφάλματα χωρίζονται επίσης σε ενεργά και μη ενεργά.

Πίνακας 2. Κατηγορίες σφαλμάτων σε σχέση με τις συνέπειες της εμφάνισής τους στην λειτουργία του συστήματος.

Ενεργά Σφάλματα	Η εμφάνισή τους οδηγεί σε αποτυχία το σύστημα.
Ανενεργά Σφάλματα	Η εμφάνισή τους δεν επηρεάζει τελικά την λειτουργία του συστήματος.

2.4.2 Υλικό και Λογισμικό.

Πηγές για την δυσλειτουργία των συστημάτων προσωπικών υπολογιστών μπορούν να είναι αστοχίες του υλικού. Συχνά τα συστήματα που βασίζονται σε μικροεπεξεργαστές λειτουργούν κάτω από αντίξοες συνθήκες όπως αστάθεια και θόρυβος στη γραμμή τροφοδοσίας, χωροταξική γειτνίαση με ισχυρές πηγές ηλεκτρομαγνητικών παρεμβολών, υψηλές θερμοκρασίες και αυξημένα επίπεδα ακτινοβολίας. Όμως χάρις στην πρόοδο στην αξιοπιστία του υλικού τα τελευταία χρόνια, το κυρίως πρόβλημα είναι η κακή λειτουργία του λογισμικού.

Όπως διαπιστώνεται στα στατιστικά στοιχεία από την λειτουργία της συστοιχίας υπολογιστών (cluster) της μηχανής αναζήτησης Google, η λόγος του αριθμού αποτυχιών του λογισμικού ως προς τον αριθμό των αποτυχιών του υλικού είναι δέκα προς ένα.[5]

Οι βασικοί λόγοι για την αποτυχία του λογισμικού είναι σφάλματα στον σχεδιασμό και στην κατασκευή του κώδικα, τα οποία εκδηλώνονται ως δυσλειτουργία του λογισμικού κατά την διάρκεια της εκτέλεσης (program execution).

Η κλασική επιστήμη της αξιοπιστίας αντιμετωπίζει μακροσκοπικά το πρόβλημα μέσω των εργαλείων της στατιστικής και των πιθανοτήτων και συγκεκριμένα με την χρήση στοχαστικών διαδικασιών με κατανομές Poisson και όχι μόνον, προσπαθώντας να περιγράψει την ακολουθία εμφάνισης σφαλμάτων σε ένα σύστημα. Ενώ όμως μέσω της αναλυτικής φυσικής, η οποία προσπαθεί να μελετήσει τις φυσικές διεργασίες που οδηγούν / συμβάλλουν στην φθορά ενός μηχανικού εξαρτήματος [6] είναι δυνατόν να διατυπωθεί ένα πλήρες και σταθερό μοντέλο για τα φυσικά εξαρτήματα, στην περίπτωση του λογισμικού οι κλασικές μέθοδοι που αναφέρονται δεν μοντελοποιούν πλήρως –και τις περισσότερες φορές ούτε κατά μέρος- το πρόβλημα.

Όπως περιγράφεται αναλυτικά στο [3] όπου πραγματοποιήθηκε η καταγραφή και η στατιστική μελέτη των σφαλμάτων που εμφάνισε ένα μεγάλο, βιομηχανικό και πολύπλοκο σύστημα λογισμικού, αποδεικνύεται πως η ακολουθία των σφαλμάτων δεν είναι διαδικασία Poisson. Μια διαισθητική εξήγηση του φαινομένου είναι πως τα ενδεχόμενα (δηλαδή τα διαφορετικά σφάλματα που ελλοχεύουν μέσα στον κώδικα του συστήματος) δεν είναι ανεξάρτητα όπως υποθέτει η στατιστική ανάλυση. Η ανεξαρτησία καταργείται επειδή η φύση της σειριακής εκτέλεσης του κώδικα (sequential execution) είναι προφανές πως επιβάλλει την εκτέλεση άρα και διαπίστωση του πρώτου (σε σειρά) από δυο ενδεχόμενα σφάλματα τα οποία εμφανίζονται σε μια περιοχή κώδικα. Έτσι στην ν-οστή δοκιμή εκτέλεσης του λογισμικού όπου θα εμφανιστεί το σφάλμα κ, παρόλο που και ένα σφάλμα κ’ είναι παρόν όσον αφορά την στατιστική μελέτη, το κ’ αποκρύβεται αφού η εκτέλεση θα διακοπεί εξαιτίας του σφάλματος κ. Θα χρειαστεί μια επόμενη ν+1 εκτέλεση ώστε να αποκαλυφθεί το σφάλμα κ’. Η επανάληψη της εκτέλεσης είναι τελικά αυτή που θα εμποδίσει την ακολουθία να συμμορφωθεί προς μια διαδικασία Poisson.

Η παραπάνω θεώρηση δικαιολογείται και από τα συμπεράσματα στο [3] αφού τελικά και στο πείραμα οι εμφανίσεις όλων των σφαλμάτων δεν ακολουθούν διαδικασία Poisson για τον λόγο που μόλις περιγράψαμε. Είναι ενδιαφέρον όμως να σημειωθεί ότι η **πρώτη εμφάνιση** κάθε σφάλματος συμμορφώνεται προς μια διαδικασία Poisson. Η διαπίστωση αυτή αποτελεί μια δικαίωση προς τις ερευνητικές προσπάθειες που έγιναν προς την κατεύθυνση της αύξησης της αξιοπιστίας του λογισμικού με βάση τις κλασικές μεθόδους της μηχανικής της αξιοπιστίας, ταυτόχρονα όμως εξηγεί και το γιατί οι προσπάθειες αυτές δεν τελεσφόρησαν προς

μια ολική αντιμετώπιση του προβλήματος. Η προσπάθεια να αυξηθεί την αξιοπιστία των στατικών μερών ενός συστήματος λογισμικού δεν μπορεί να οδηγήσει στην ολική σταθερότητα του συστήματος γιατί θα υπάρχει πάντοτε ένα τμήμα της λειτουργίας που είναι δυναμικό.

Τέλος, στα συμπεράσματα του [3] διατυπώνεται μεταξύ άλλων το αίτημα της καταγραφής στατιστικών στοιχείων για την συμπεριφορά του λογισμικού και ειδικότερα στην αυτοματοποίηση αυτής της διαδικασίας (αίτημα το οποίο λαμβάνεται υπ' όψη στο σύστημα που υλοποιείται στα πλαίσια της παρούσας διπλωματικής εργασίας) ώστε να είναι δυνατή η ανάλυση των λόγων αποτυχίας του λογισμικού. Το αίτημα αυτό είναι απολύτως δικαιολογημένο από την στιγμή που εκ των πραγμάτων είναι αληθής η υπόθεση πως ο ρυθμός και οι ακολουθίες με τις οποίες αποτυγχάνει το λογισμικό δεν είναι τόσο προβλέψιμες όσο οι αντίστοιχες του υλικού. Έτσι η αντιμετώπιση του προβλήματος στα πλαίσια μιας στατικής μοντελοποίησης δεν προσφέρει τα επιθυμητά οφέλη. Γι' αυτό είναι υποχρεωτική η σχεδίαση δυναμικών – ενεργητικών συστημάτων τα οποία με την ανάλυση και παρέμβασή τους σε πραγματικό χρόνο βοηθούν τα συστήματα λογισμικού να αυξήσουν την αξιοπιστία τους.

Η εισαγωγή αυτών των εργαλείων μπορεί να οδηγήσει στην διάγνωση των σχεδιαστικών λαθών που οδηγούν στην αποτυχία του λογισμικού, οπότε όπως αναφέρεται και στο [4] η εισαγωγή των εννοιών και των εργαλείων του Reliability Engineering κυρίως στο στάδιο της ανάπτυξης θα επιφέρει την αύξηση της αξιοπιστίας του λογισμικού.

2.4.3 Δικτύωση συστημάτων και κατανεμημένα υπολογιστικά συστήματα.

Η δυνατότητα της σύνδεσης ηλεκτρονικών υπολογιστών μέσω δικτύου σήμερα πλέον είναι αυτονόητο χαρακτηριστικό κάθε συστήματος πληροφορικής. Όπως αναφέρεται και στην εισαγωγή του κεφαλαίου περί δικτύων του βιβλίου των Hennessy & Patterson [5], «όπως ένας σύγχρονος υπολογιστής χωρίς ιεραρχία μνήμης είναι *χαλασμένος* (broken) έτσι και ένας σύγχρονος υπολογιστής χωρίς πρόσβαση στο δίκτυο είναι εξίσου *χαλασμένος*». Με την παραπάνω υπερβολή οι συγγραφείς

προφανώς προσπαθούν να υπερτονίσουν πως σήμερα δεν νοείται εφαρμογή πληροφορικής εάν πρώτα απ' όλα δεν είναι δικτυακή εφαρμογή.

Οι συνέπειες της ολοκλήρωσης των υπολογιστικών συστημάτων με τα δίκτυα μάλιστα σήμερα ενισχύεται περισσότερο ακόμη από την ραγδαία ανάπτυξη των ασύρματων δικτύων (wireless networks) τα οποία προσθέτουν δυο ακόμη παραμέτρους στο θέμα:

- Η δικτύωση μπορεί να είναι μεταβαλλόμενη και δυναμική και δεν υπάρχει η σταθερότητα στην τοπολογία που επιβαλλόταν από τα ενσύρματα τοπικά δίκτυα.

- Η δικτύωση γίνεται πιο φτηνή με την χρήση πομποδεκτών που περικλείονται σε ένα ολοκληρωμένο κύκλωμα δίνοντας δυνατότητα για δικτύωση σε χαμηλού κόστους συστήματα μικροελεγκτών, ενσωματωμένων συστημάτων (embedded systems) κλπ. Η δουλειά που περιγράφεται στο [7] από την ομάδα του Berkeley αποτελεί ένα κλαστικό παράδειγμα.

Είναι προφανές πως το φαινόμενο του δικτύου «network effect» δηλαδή η αύξηση της συνολικής αξίας των συστημάτων εκθετικά ως προς τον αριθμό των δικτυωμένων συστημάτων δεν αφήνει ανεπηρέαστη και την αξιοπιστία. Μια ολόκληρη γενιά νέων αλγορίθμων και συστημάτων αμοιβαίας ή κεντρικής παρακολούθησης ορθής λειτουργίας έχουν αναπτυχθεί με βασικό σκοπό την εκμετάλλευση του φαινόμενου του δικτύου για τους σκοπούς της αξιοπιστίας. Αποκορύφωμα αυτών είναι οι τεχνολογίες της ομαδοποίησης (clustering), οι οποίες φτάνουν στο σημείο να προσφέρουν υπηρεσίες υπερυψηλής διαθεσιμότητας (τα λεγόμενα «five nines» δηλαδή ορθή λειτουργία του συστήματος στο 99,999% του χρόνου).

Είναι όμως αναγκαίο να επισημανθούν δυο θέματα τα οποία είναι ανοιχτά σχετικά με την σχέση αξιοπιστίας και δικτυωμένων υπολογιστικών συστημάτων, τα οποία περιγράφονται στις επόμενες παραγράφους, το πρόβλημα της απομόνωσης και το θέμα της «έξυπνης αξιοπιστίας».

Το πρόβλημα της απομόνωσης.

Η κυρίαρχη τάση του grid computing δηλαδή των κατανεμημένων συστημάτων που αποτελούνται από τεράστιο αριθμό κόμβων έχει επικεντρώσει το πρόβλημα της διαθεσιμότητας (availability) και αξιοπιστίας στην συνολική επίδοση ολόκληρου του κατανεμημένου συστήματος. Παρ' όλα αυτά δεν θα πρέπει να παραμεληθεί από την ερευνητική κοινότητα η ανάγκη για την υπεράσπιση και της ατομικής αξιοπιστίας του κάθε κόμβου που θα συμμετέχει στα μελλοντικά συστήματα πλέγματος (grids).

Με άλλα λόγια πρέπει να επισημανθεί πως πάντοτε είναι αναγκαίο να φροντίζουμε για την αξιοπιστία του κάθε κόμβου, αφού αυτός ο κόμβος υπάρχει η πιθανότητα να αποτελεί το σημείο πρόσβασης του τελικού χρήστη στο σύστημα. Όπως επισημαίνεται και στο άρθρο [2] του Daniel Begun ένα από τα σημεία στα οποία απαιτείται βελτίωση σχετικά με την εμπειρία που έχουν οι τελικοί χρήστες με τα σύγχρονα υπολογιστικά συστήματα είναι η έλλειψη αξιοπιστίας. Με δεδομένες τις οικονομικές συνέπειες που έχει η εμπέδωση της χρήσης των υπολογιστών στην κοινωνία είναι μόνον προς το όφελος γενικά της πληροφορικής η παροχή πιο αξιόπιστων συστημάτων έτσι ώστε αυτό να καταγράφεται στην συνείδηση του τελικού χρήστη. Δεν θα πρέπει λοιπόν οι ερευνητές, οι κατασκευαστές συστημάτων και η βιομηχανία, αφοσιωμένοι στην αναζήτηση του «άγιου δισκοπότηρου» της αξιοπιστίας του ολοκληρωμένου συστήματος να αγνοούν τους επιμέρους κόμβους. Άλλωστε η σωρευτική αποτυχία των κόμβων τελικά θα οδηγήσει με σιγουριά και στην αποτυχία του συστήματος.

«Έξυπνη» αξιοπιστία.

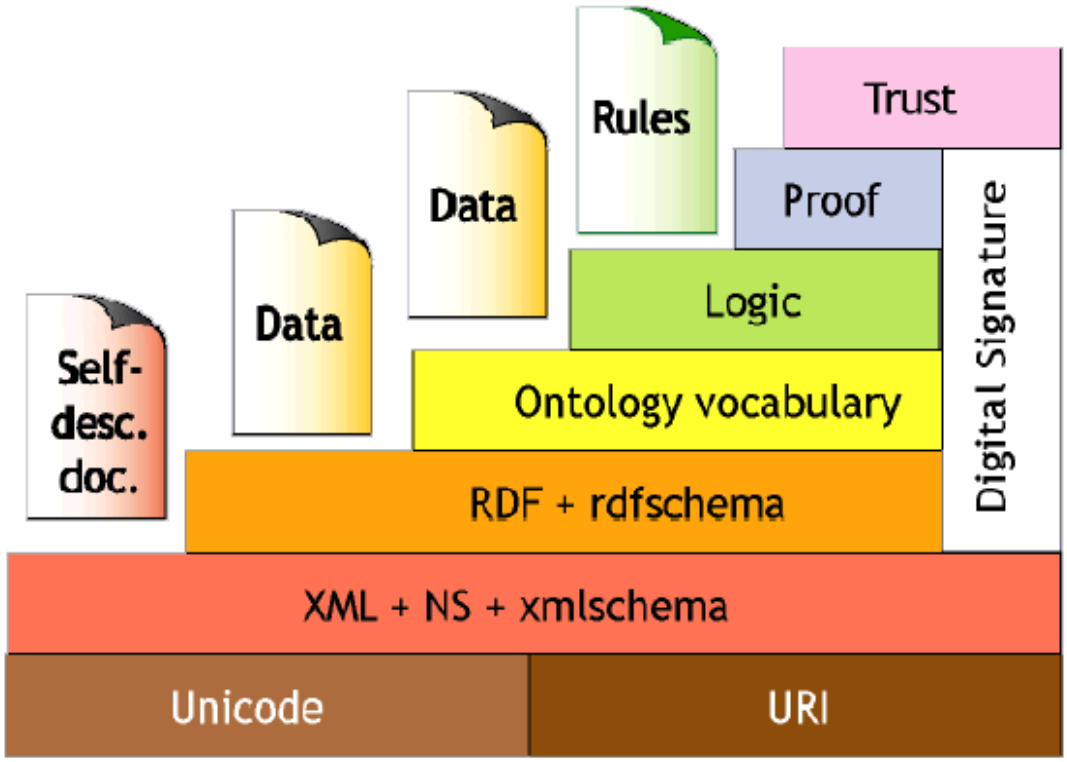
Έχοντας εντοπίσει τα σχεδιαστικά σφάλματα ενός συστήματος (βλέπε §2.4.1) ως έναν από τους βασικούς λόγους δυσλειτουργίας των συστημάτων πληροφορικής είναι αναμενόμενο να υπάρχουν προτάσεις προς ερευνητικές προσπάθειες για την αύξηση της αξιοπιστίας μέσω εργαλείων και τεχνικών οι οποίες προσπαθούν να θωρακίσουν τους σχεδιασμούς από αυτά τα λάθη. Είναι όμως κοινή διαπίστωση πως το μεγαλύτερο ποσοστό των λαθών αυτών οφείλεται σε αστοχίες στην σημασιολογία (semantic failure) που χρησιμοποιείται από τους διάφορους σχεδιαστές καθώς και στην έλλειψη αρκετά ισχυρών γλωσσών για την περιγραφή των λεπτών οντολογικών

διατυπώσεων σχετικά με τις ιδιότητες κάθε υποσυστήματος που συμμετέχει σε ένα υπολογιστικό σύστημα.

Εντελώς αναπάντεχα φαίνεται πως πιθανώς ο χώρος της επιστήμης της αξιοπιστίας, ο οποίος παραδοσιακά αποτελούσε κυρίαρχο χώρο των πιθανοτήτων και της στατιστικής να δεχθεί την εισβολή των διακριτών μαθηματικών και της προτασιακής λογικής. Ένα από τα οράματα του Semantic Web (μια πολύ καλή εισαγωγή μπορεί να βρει ο αναγνώστης στο άρθρο [8]) είναι το «αξιόπιστο web» (web of trust), δηλαδή η παροχή αξιόπιστων υπηρεσιών παγκόσμιου ιστού, οι οποίες έχουν ενσωματωμένες στη δομή τους όλους τους αναγκαίους μηχανισμούς για την παροχή σταθερών (και συνεπώς αξιόπιστων) υπηρεσιών σε όλα τα επίπεδα. Παρόλο που η αναφορά του μηχανισμού λειτουργίας του semantic web ξεφεύγει από τα όρια της παρούσας εργασίας, είναι αναγκαίο να γίνει αναφορά στο επίπεδο μεταξύ δεδομένων και κανόνων, δηλαδή στο επίπεδο των οντολογιών. Οι οντολογίες (web ontology language) αποτελούν ένα πλαίσιο σημασιολογικής περιγραφής υπηρεσιών και λειτουργιών, πάνω στις οποίες περιγραφές και χρησιμοποιώντας πρωτοβάθμια λογική μπορούν να εκτελεστούν λογικές συναγωγές και να ενοποιηθούν διάφορες λειτουργίες.

Η λειτουργία του συστήματος βασίζεται στην χρήση κοινών περιγραφών υπηρεσιών με την χρήση των τεχνολογιών του Resource Description Framework και της προτασιακής λογικής. Ενώ μέχρι σήμερα η σύνθεση των απλούστερων υπηρεσιών σε περισσότερο σύνθετες γίνονταν με ad hock τρόπο, μέσα στο πλαίσιο λειτουργίας των οντολογιών η σύνθεση αυτή θα υπακούει στην εφαρμογή λογικών κανόνων, δίνοντας στον συντονιστή των υπηρεσιών (service coordinator) ένα ισχυρό εργαλείο ελέγχου της συμβατότητας άρα και της δυνατότητας σταθερής συνεργασίας μεταξύ των συντιθέμενων υπηρεσιών.

Είναι προφανές πως η επιτυχία αυτού του μοντέλου θα αποτελέσει μια ιδιαίτερη σημαντική ενίσχυση της αξιοπιστίας των διαθέσιμων συστημάτων. Εφόσον θα είναι διαθέσιμο ένα μαθηματικό σύστημα το οποίο θα εγγυάται τον αυτόματο και ευσταθή χειρισμό της σημασιολογίας των συστημάτων λογισμικού, ο κύριος λόγος για τον οποίο το λογισμικό αποτυγχάνει θα καταπολεμηθεί στη ρίζα του. Κατά συνέπεια η συνολική αξιοπιστία των συστημάτων αναμένεται να βελτιωθεί σε σημαντικό βαθμό.



Γράφημα 4. The semantic web component stack.

Ωστόσο τα σύνθετα συστήματα πάντοτε θα βασίζονται στη σύνθεση των απλών στοιχειωδών συστημάτων, των οποίων η αποικοδόμηση θα οδηγήσει με βεβαιότητα σε ένα σύστημα υλικού λογισμικού, του οποίου την αξιοπιστία υποστηρίζει το υπό υλοποίηση σύστημα που παρουσιάζεται σε αυτή την εργασία.

2.5 Σχέση αξιοπιστίας και κόστους.

Ένα από τα σημαντικά χαρακτηριστικά κάθε συστήματος είναι το κόστος του. Συνήθως το κριτήριο με το οποίο γίνεται η επιλογή της αγοράς ενός συστήματος είναι αυτό του ελάχιστου κόστους με το οποίο καλύπτονται οι αναγκαίες προδιαγραφές του συστήματος. Προφανώς το κριτήριο αυτό επιτρέπει στις επιχειρήσεις να λειτουργούν με τον πλέον οικονομικό τρόπο.

Παρ' όλα αυτά υπάρχει συχνά μια λάθος προσέγγιση, διότι στον υπολογισμό του κόστους λαμβάνεται υπόψη μόνον το αρχικό κόστος, αγνοώντας το κόστος της

συντήρησης του συστήματος (ώστε αυτό να εξακολουθεί να καλύπτει τις αρχικές προδιαγραφές). Είναι βέβαια αυταπόδεικτο πως το κόστος συντήρησης και επισκευής εξαρτάται άμεσα από τα επίπεδα αξιοπιστίας του συστήματος. Όσο πιο αξιόπιστο είναι το σύστημα, τόσο πιο λίγο θα κοστίσει η συντήρηση ή / και η επισκευή του, ωστόσο το αρχικό κόστος θα έχει την τάση να είναι υψηλότερο.

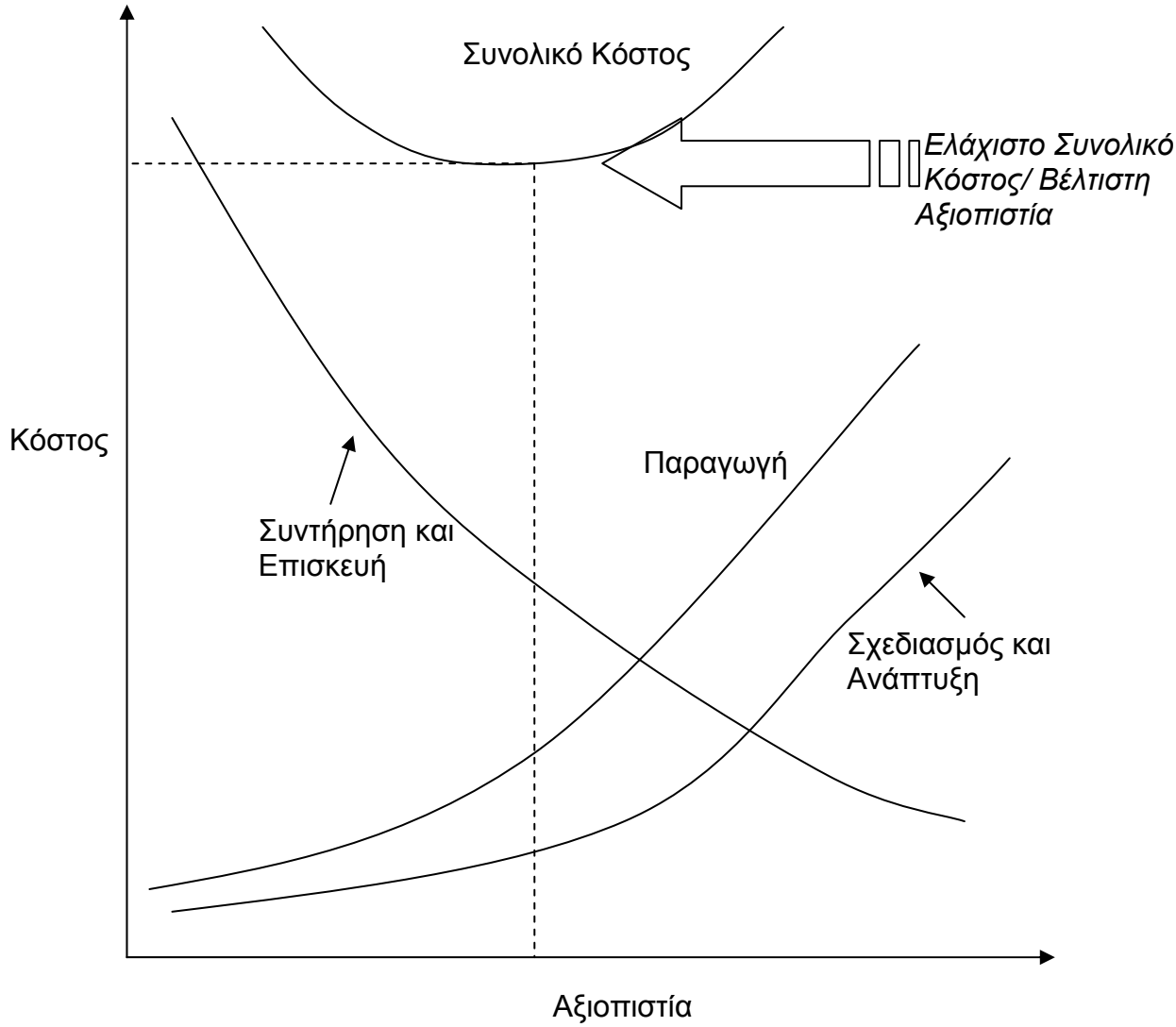
Το συνολικό κόστος ενός εξαρτήματος αποτελεί το άθροισμα τριών όρων που είναι το κόστος του σχεδιασμού, το κόστος της παραγωγής και το κόστος της συντήρησης. Όταν η αξιοπιστία ενός συστήματος αυξάνει, είναι προφανές πως θα υπάρχει αύξηση στα κόστη σχεδιασμού και παραγωγής, ενώ ταυτόχρονα το κόστος συντήρησης θα ελαττώνεται.

Οι παράγοντες για την αύξηση του κόστους σχεδιασμού και παραγωγής είναι η λεπτομερέστερη και περισσότερο αναλυτική διαδικασία ανάπτυξης με περισσότερα πειράματα και ελέγχους καθώς επίσης και η μεγαλύτερη σταθεροποίηση των συνθηκών παραγωγής (π.χ. μείωση των ανοχών στην αλυσίδα παραγωγής ή βελτίωση της ποιότητας των εξαρτημάτων).

Στο γράφημα εμφανίζεται η άθροιση των παραπάνω μεγεθών και η καμπύλη του τελικού κόστους, η οποία προκύπτει τελικά, επιπλέον χάρις σε αυτήν μπορεί να εξαχθεί η πλέον συμφέρουσα απόφαση για το σύστημα που θα επιλεχθεί προς αγορά.

Από την άλλη, η πολυπλοκότητα των συστημάτων δεν επιτρέπει τις περισσότερες φορές τον προσδιορισμό όλων των παραμέτρων ώστε να είναι δυνατή μια τόσο ακριβής περιγραφή ώστε να χαραχτεί με ακρίβεια η καμπύλη.

Ωστόσο είναι προφανές πως είναι υπαρκτή η θεωρητική βάση, η οποία τεκμηριώνει το επιχείρημα της αναγκαιότητας να αναζητείται η βέλτιστη ή η σχεδόν βέλτιστη λύση στην σχέση κόστος προς αξιοπιστία, αφού η απομάκρυνση από τον βέλτιστο λόγο είτε προς τα δεξιά είτε προς τα αριστερά θα οδηγήσει σε εξίσου επιζήμιο αποτέλεσμα.



Γράφημα 5. Συνάρτηση Κόστους Αξιοπιστίας. (Πηγή: [23])

3. Εποπτεία συστημάτων & μοντελοποίηση της προτεινόμενης λύσης.

3.1 Εισαγωγή.

Σύμφωνα με τις θεωρητικές βάσεις καθώς και τα ερευνητικά συμπεράσματα που παρατέθηκαν στο προηγούμενο κεφάλαιο και πάντοτε στην τροχιά της αύξησης της αξιοπιστίας των υπολογιστικών συστημάτων, σε ένα πρώτο επίπεδο επιβεβαιώνεται το σταθερό αίτημα για αξιοπιστία που εκπεφρασμένα ανακύπτει σε όλες τις τρέχουσες εξελίξεις στην έρευνα και διαπιστώνεται η ανάγκη για την σταθερή συμπεριφορά των συστημάτων ηλεκτρονικών υπολογιστών όσον αφορά την ανοχή σε σφάλματα και την αξιοπιστία. Είναι χαρακτηριστικό πως στην σημερινή φάση της έρευνας όπου η τεχνολογία των κατανεμημένων υπολογιστικών συστημάτων προβάλλεται από πολλούς ως το μελλοντικό σημείο αναφοράς για την επιστήμη της πληροφορικής, η αξιοπιστία και η ανοχή σε σφάλματα των κατανεμημένων υπολογιστικών συστημάτων αποτελεί ένα από τα βασικά στοιχεία που παρουσιάζονται ως πλεονεκτήματα και ένα από τα στοιχεία που κάνουν την ειδοποιό διαφορά των κατανεμημένων συστημάτων σε σχέση με τα κεντρικοποιημένα.

3.2 Δυναμικές μέθοδοι βελτίωσης αξιοπιστίας.

Οι διαδικασίες που μπορούν να ακολουθηθούν για την αύξηση της αξιοπιστίας ενός συστήματος, όπως περιγράφηκε στο προηγούμενο κεφάλαιο, αφορούν σε ένα μεγάλο μέρος τους ποιοτική αναβάθμιση της διαδικασίας σχεδιασμού (εκτενέστερες δοκιμές λειτουργίας, χρήση αναλυτικότερων μοντέλων σχεδιασμών) όπως και της διαδικασίας κατασκευής (βελτίωση ποιότητας εξαρτημάτων, βελτίωση συνθηκών κατασκευής). Όμως υπάρχει μία εξίσου ισχυρή συνιστώσα στην συνολική συνισταμένη της αξιοπιστίας και αυτή είναι η αστάθεια που υπεισέρχεται από την δυναμική λειτουργία του συστήματος. Είναι προφανές πως απαιτείται η εφαρμογή ενός δυναμικού μηχανισμού, ο οποίος θα είναι σε θέση να εποπτεύει το σύστημα του οποίου η αξιοπιστία είναι επιθυμητό να βελτιωθεί.

Μια ευρέως διαδεδομένη μέθοδος βελτίωσης της αξιοπιστίας είναι το clustering δηλαδή η δημιουργία εφεδρικών συστημάτων. Η γενική αρχή λειτουργίας του clustering είναι η συντήρηση δύο ή περισσότερων «δίδυμων» υπολογιστικών συστημάτων, τα οποία παρέχουν την ίδια υπηρεσία. Ένα από τα συστήματα αναλαμβάνει την παροχή των υπηρεσιών αλλά ταυτοχρόνως ενημερώνει σε κάθε βήμα του τα δίδυμα συστήματα για την πρόοδό του. Σε περίπτωση που διακοπεί η λειτουργία του λόγω σφάλματος, η παροχή των υπηρεσιών συνεχίζεται από κάποιο από τα πλεονάζοντα συστήματα (active / passive cluster operation), μέχρις ότου να ειδοποιηθεί ο διαχειριστής του συστήματος και να αναλάβει την επισκευή του συστήματος, ώστε αυτό να επανέλθει στην κανονική του λειτουργία. Προσφάτως εμφανίστηκαν και εμπορικά εφεδρικά συστήματα στα οποία δεν υπάρχει πλέον η έννοια του πλεονάζοντος συστήματος αλλά αντιθέτως όλα τα υπολογιστικά συστήματα συμμετέχουν εξίσου στην παροχή των υπηρεσιών (active / active cluster operation). Εφόσον κάποιο από τα συστήματα παρουσιάσει σφάλμα, τότε τα υπόλοιπα συστήματα συνεχίζουν να λειτουργούν κανονικά, εμφανίζοντας όμως μειωμένη απόδοση σε όρους ρυθμού εξυπηρέτησης πελατών από πριν.

Η τεχνική των πλεοναζόντων συστημάτων είναι συνεπής αλλά και ιδιαίτερα δαπανηρή, ώστε η εφαρμογή της να αποτελεί επιλογή μόνο στην περίπτωση κρίσιμων υπηρεσιών.

Όσον αφορά όμως τις εφαρμογές όπου οι απαιτήσεις της κρισιμότητας των υπηρεσιών είναι λιγότερο πιεστική, μπορεί να εφαρμοστεί η έννοια του φρουρού (watchdog), σύμφωνα με την οποία ένα σύστημα παρακολουθεί την ορθή λειτουργία ενός άλλου συστήματος με σκοπό την ενεργοποίηση διάφορων διαδικασιών για την αποτροπή ή επιδιόρθωση βλαβών. Προϋπόθεση της λειτουργίας αυτής είναι η καταρχήν λογική ανεξαρτησία λειτουργίας των δυο συστημάτων.

3.3 Φρουροί

Οι watchdog timers και οι watchdog επεξεργαστές χρησιμοποιούνται στην ανίχνευση λαθών στο υλικό ή / και στο λογισμικό. Εφόσον ένα λάθος ανιχνευθεί μία κατάλληλη διακοπή (interrupt) ή μια επανεκκίνηση (reset) του συστήματος θα επαναφέρει το σύστημα στην κανονική του λειτουργία. Μάλιστα, στην περίπτωση ενός συστήματος

με ανοχή στα σφάλματα η κατάλληλη διακοπή μπορεί να οδηγήσει στην αποσύνδεση του λανθάνοντος επεξεργαστή από το σύστημα και ενδεχομένως την ενεργοποίηση ενός εφεδρικού επεξεργαστή στη θέση του. Τα συστήματα watchdog timers και οι watchdog επεξεργαστές κατατάσσονται στις ταυτόχρονες τεχνικές ανίχνευσης λαθών (concurrent error detection).

3.3.1 Watchdog Timers

Ο watchdog timer που είναι η πιο απλή μέθοδος ανίχνευσης λαθών αποτελείται από έναν hardware timer ο οποίος μπορεί να βρίσκεται είτε εντός είτε εκτός του επεξεργαστή που φυλάσσει. Εάν ο timer αυτός μηδενιστεί τότε ενεργοποιείται η διακοπή που είναι υπεύθυνη για την διαδικασία ανάνηψης, η οποία τις περισσότερες φορές δεν είναι άλλη από την επανεκκίνηση του επεξεργαστή που φυλάσσει ο watchdog timer. Προφανώς, το πρόγραμμα που ελέγχει τον επεξεργαστή θα πρέπει να περιέχει στην κανονική ροή του εντολές που αρχικοποιούν σε τακτά χρονικά διαστήματα τον watchdog timer ώστε να αποφεύγεται ο μηδενισμός του.

Υπάρχουν τέσσερα είδη (όπως καταγράφονται στο [9]) watchdog timers:

(i) Μέρος του επεξεργαστή.
(ii) Ενσωματωμένα στο chipset του επεξεργαστή.
(iii) Ως κάρτα επέκτασης.
(iv) Ως συσκευή που επικοινωνεί με κάποιο κανάλι (σειριακή, παράλληλη).

Πρωτογενής δουλειά σε Watchdog Timers έγινε από τον Tim Williams, ο οποίος περιγράφει τα ηλεκτρικά χαρακτηριστικά του watchdog, όπως ο τρόπος που συνδέονται με το κύκλωμα και η μέθοδος του reset. Πρώτες αναφορές σε τέτοια κυκλώματα έκανε στο βιβλίο του [10].

Παραλλαγές και επεκτάσεις στην λειτουργία του Watchdog Timer έχουν προταθεί όπως στο [9] όπου η επιβεβαίωση της λειτουργίας του συστήματος γίνεται με την έξυπνη χρήση της συσκευής του πληκτρολογίου. Συγκεκριμένα θέτοντας στο σύστημα την δοκιμασία της απόκρισης σε μηνύματα για κάθε πάτημα του πλήκτρου caps lock, num lock, scroll lock, το CPU απαντάει με συγκεκριμένους κωδικούς για

την διαχείριση των ενδεικτικών led. Σε αυτή την παραλλαγή περίπου κάθε ένα λεπτό γίνεται μια στιγμιαία εναλλαγή σε ένα από τα πλήκτρα ώστε να διαπιστωθεί η απόκριση του CPU στην αλλαγή των led.

3.3.2 Watchdog Επεξεργαστές

Από την άλλη οι watchdog επεξεργαστές εποπτεύουν τις γραμμές εισόδου εξόδου του υπό παρακολούθηση επεξεργαστή και προσπαθούν να διαγνώσουν λάθη στις αποκρίσεις του σύμφωνα με τα δεδομένα που αυτός λαμβάνει και τις αποκρίσεις που στέλνει. Ένας από τους βασικούς λόγους που η έρευνα οδηγήθηκε στην έννοια του watchdog processor είναι η πιθανότητα ένας επεξεργαστής ο οποίος εποπτεύεται από έναν watchdog timer να βρεθεί σε μια τέτοια κατάσταση δυσλειτουργίας κατά την οποία ενώ δεν λειτουργεί εντός προδιαγραφών παρ 'όλα αυτά εκτελεί τις εντολές που είναι υπεύθυνες για την αρχικοποίηση του φρουρού, με αποτέλεσμα την εσφαλμένη αργία του τελευταίου.

3.4 Επεκτάσεις της τεχνικής των φρουρών.

Η χρήση των watchdog σχεδιασμών μελετάται ακόμη και σήμερα (σε αντίθεση με τους ισχυρισμούς κάποιων ότι πρόκειται για μια ώριμη τεχνολογία). Χαρακτηριστική είναι η πρόσφατη εργασία των Iben, Lakhia και Rubin [7] στην οποία μελετούν τον μηχανισμό watchdog είτε στην απλή του εκδοχή της παρακολούθησης ενός σύνθετου συστήματος από ένα απλό (δηλαδή στο μοντέλο που βασιζόμαστε στην υλοποίηση που παρουσιάζουμε στα πλαίσια αυτής της εργασίας), είτε στην επέκταση του μηχανισμού σε ασύρματα δίκτυα επεξεργαστικών μονάδων, όπου η μία παρακολουθεί την άλλη. Στα συμπεράσματα της εργασίας τους περιλαμβάνεται η παρατήρηση πως υπάρχει μεγάλο περιθώριο βελτίωσης και επέκτασης των μηχανισμών watchdog.

Επιπλέον είναι σκόπιμο να επισημανθεί μια διάσταση επικαιρότητας στο πρόβλημα που πραγματεύεται η παρούσα εργασία και αυτή δεν είναι άλλή απ' το γεγονός πως παρόλο που ο μηχανισμός του watchdog εμφανίζεται από κάποιους στην βιβλιογραφία ως μια ώριμη τεχνολογία, τελικά διαπιστώνεται πως η καθιέρωση της

χρήσης του είναι μια παγιωμένη πραγματικότητα σε ορισμένους τομείς (όπως π.χ. ο τομέας των γιατρικών μηχανημάτων) και όχι γενικευμένα όπως ίσως θα έπρεπε στην βιομηχανία της πληροφορικής. Μάλιστα από κάποιους περιγράφεται ως αναγκαιότητα η ευρύτερη χρήση των μηχανισμών watchdog [2] ως μια άμεσα εφαρμόσιμη λύση προς την αύξηση της αξιοπιστίας των συστημάτων οργάνωσης γραφείου σήμερα.

Η χρήση των hardware watchdog timers είναι θεμελιώδης και στα κατανεμημένα συστήματα για προβλήματα που αφορούν την αρχιτεκτονική συστημάτων με ανοχή στα σφάλματα. Συγκεκριμένα στο [11] περιγράφεται ο αλγόριθμος του Perfect Failure Detector, ο οποίος είναι ένας μηχανισμός που εγγυάται την σίγουρη ανίχνευση του σφάλματος λειτουργίας ενός συστήματος σε κατανεμημένα συστήματα με κοινό χρόνο. Στο παραπάνω μοντέλο κάθε επεξεργαστής που συμμετέχει στο κατανεμημένο σύστημα παρακολουθείται από έναν watchdog timer ο οποίος όμως δεν φροντίζει να εκκινήσει από την αρχή το σύστημα εφόσον αυτό εμφανίσει πρόβλημα αλλά να του προκαλέσει την τεχνητή διακοπή λειτουργίας εφόσον το επιβάλει το πρωτόκολλο του perfect failure detector.

Παρ' όλα αυτά όπως θα καταδειχθεί και στην σχετική έρευνα, η αξιοπιστία αυτή χτίζεται πάνω σε κλασικούς μηχανισμούς της τεχνολογίας αξιοπιστίας όπως ο μηχανισμός του watchdog.

Επάνω ακριβώς σε αυτή τη διαπίστωση και σε ένα δεύτερο επίπεδο στα πλαίσια της διπλωματικής αυτής εργασίας μελετάται, σχεδιάζεται και υλοποιείται ένα σύστημα watchdog βασισμένο σε μικροελεγκτή το οποίο χάρις ακριβώς στις δυνατότητες των σύγχρονων μικροελεγκτών δίνει μια νέα οπτική στις αρμοδιότητες και στην λειτουργικότητα που προσφέρει ένα watchdog σύστημα.

3.5 Ανάλυση Εναλλακτικών Λύσεων.

Μια λύση στο πρόβλημα κατά την οποία δύο ή περισσότεροι προσωπικοί υπολογιστές που επικοινωνούν μέσω δικτύου λειτουργούν ως φρουρός ο ένας στον άλλο είναι εφικτή και αρχικά φαίνεται να εμφανίζει την ίδια συμπεριφορά με το

σύστημα που υλοποιείται στην παρούσα εργασία. Παρόλα αυτά υπάρχουν εγγενείς αδυναμίες.

Στην περίπτωση που χρησιμοποιείται το δίκτυο επικοινωνιών για την ανταλλαγή των μηνυμάτων ελέγχου του φρουρού είναι δυνατόν σε περίπτωση που το δίκτυο είναι υπερφορτωμένο να υπολειτουργεί, δίνοντας την εσφαλμένη εντύπωση της μη απόκρισης του φρουρούμενου συστήματος με αποτέλεσμα να παρθεί εσφαλμένη απόφαση από τον φρουρό για την δυσλειτουργία του προσωπικού υπολογιστή. Είναι προφανές πως για να επιτευχθεί υψηλός δείκτης αξιοπιστίας της απόφασης το κανάλι επικοινωνίας πρέπει να παρέχει μεγαλύτερες εγγυήσεις από το παραδοσιακό κανάλι δικτύου.

Επιπλέον η σχετική απλότητα (σε σχέση με τον προσωπικό υπολογιστή) στην λειτουργία του μικροελεγκτή μειώνει την πιθανότητα σφάλματος στην εκτέλεση του φρουρού, η οποία από την άλλη ποτέ βέβαια δεν είναι μηδενική αφού και ο φρουρός είναι λογισμικό.

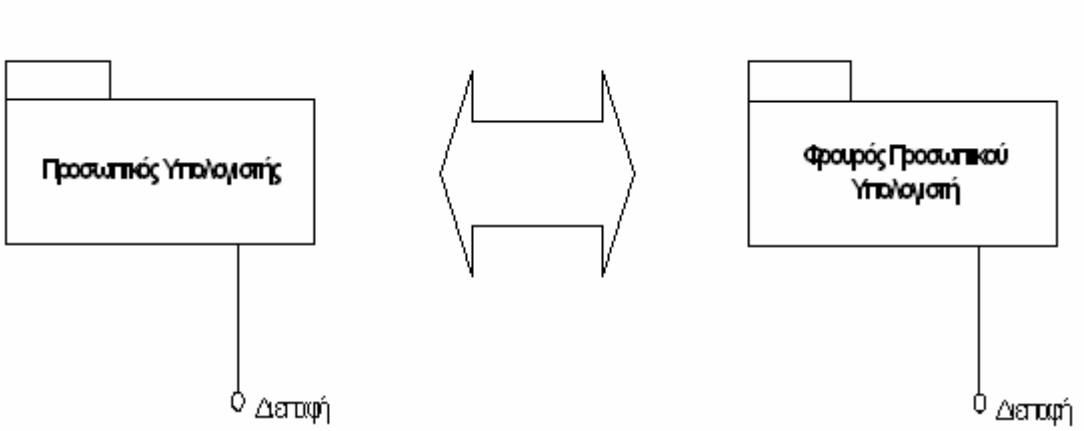
Τέλος, εξαιτίας της λογικής απομόνωσης του μικροελεγκτή σε σχέση με το σύστημα του προσωπικού υπολογιστή (αφού το πρωτόκολλο επικοινωνίας μεταφέρει μόνο δεδομένα ελέγχου) εγγυάται πως δεν υπάρχει περίπτωση αλλοίωσης της λειτουργίας του φρουρού από κακόβουλες ενέργειες όπως π.χ. οι υιοί ή οι δούρειοι ίπποι. Η αρχή λειτουργίας παρόμοιων προγραμμάτων είναι πως πάντα επιτίθεται σε όλους τους δικτυακούς γείτονες του συστήματος που καταφέρνουν να προσβάλλουν. Έτσι ο φρουρός, ο οποίος είναι λογικά και φυσικά απομονωμένος διαθέτει ένα σημαντικό πλεονέκτημα ασφάλειας.

3.5 Μοντελοποίηση του φρουρού προσωπικού υπολογιστή.

Η πρόταση που υλοποιείται για την βελτίωση της αξιοπιστίας του προσωπικού υπολογιστή συνίσταται σε ένα σύστημα υλικού-λογισμικού το οποίο εποπτεύει την λειτουργία ενός PC στα δυο παρακάτω επίπεδα:

- Παρακολούθηση λειτουργίας λογισμικού συστήματος.
- Παρακολούθηση λειτουργίας κρίσιμων παραμέτρων λειτουργίας συστήματος.

Το σύστημα αποτελείται από ένα υποσύστημα υλικού βασισμένο σε μικροελεγκτή, με κύριο στόχο τον περιορισμό του κόστους και από δυο ανεξάρτητα αλλά σε αλληλεπίδραση υποσυστήματα λογισμικού εκ των οποίων τα ένα φιλοξενείται στον υπό επόπτευση προσωπικό υπολογιστή, ενώ το δεύτερο στον μικροελεγκτή.



Γράφημα 6. Αναπαράσταση βασικής λειτουργικότητας.

Το σύστημα είναι καταρχήν φυσικά και λογικά διαχωρισμένο (detached) από τον υπολογιστή που επιβλέπει και η σύνδεση γίνεται μέσω σειριακής επικοινωνίας. Η αυτονομία τόσο στη λειτουργία όσο και στην τροφοδοσία, δίνει την μέγιστη αυτάρκεια στο σύστημά, αφού θα εξακολουθήσει να λειτουργεί ακόμη και σε πλήρη αποτυχία (π.χ. απώλεια τροφοδοσίας) του υπολογιστή υπό επίβλεψη.

Επιπλέον χάρις στις δυνατότητες των νέων μικροελεγκτών προστίθεται νέα λειτουργικότητα στο υποσύστημα του watchdog, ο οποίος έχοντας τη δυνατότητα να εκτελέσει πράξεις επικοινωνίας (π.χ. μέσω της οδήγησης μιας συσκευής modem) μπορεί να ειδοποιήσει για την εμφάνιση προβλημάτων. Έτσι η λειτουργικότητα του watchdog δεν περιορίζεται στην στατική διάσταση της επανεκκίνησης (reboot) του συστήματος σε περίπτωση σφάλματος αλλά επεκτείνεται.

Η επέκταση αυτή μάλιστα αποκτά ιδιαίτερη βαρύτητα όταν λόγω της σημασίας που έχει η δικτυακή σύνδεση στα σημερινά συστήματα, ο watchdog που υλοποιείται μπορεί εκτός των άλλων να στείλει σήματα ενημέρωσης ακόμη και για την περίπτωση απώλειας δικτύου εκτός βασικού καναλιού (out of band) ενημερώνοντας τους ενδιαφερόμενους πως η διακοπή της επικοινωνίας με τον υπολογιστή υπό

επίβλεψη δεν οφείλεται σε δυσλειτουργία του ίδιου αλλά σε πρόβλημα στο δίκτυο. Η εν λόγω επέκταση έρχεται να υποστηρίξει μια ακόμη διάσταση της του προβλήματος της επί μέρους αξιοπιστίας των peer to peer κατανεμημένων συστημάτων σε ένα λειτουργικό σύνολο όπως αναλύσαμε εκτενώς.

Το σύστημα πέραν της βασικής λειτουργικότητας του watchdog διαθέτει και τις παρακάτω δυνατότητες:

- Λογισμικό για την παρακολούθηση (monitoring) όλων των σημαντικών παραμέτρων του ηλεκτρονικού υπολογιστή.
- Ημερολόγιο (log) συμβάντων πάνω στο watchdog το οποίο μπορεί να ανακτηθεί μέσω της σειριακής επικοινωνίας ώστε να μπορεί να γίνει διάγνωση ακόμη και αν ο υπολογιστής υπό επίβλεψη είναι αδύνατον να επανέλθει σε λειτουργία. (Black box).
- Δυνατότητα οδήγησης modem ώστε να οδηγηθούν εξωτερικές συσκευές τηλεειδοποίησης (π.χ. ένα modem το οποίο επικοινωνεί με συγκεκριμένους τηλεφωνικούς αριθμούς) σχετικά με τα προβλήματα που εμφανίστηκαν.

4. Φρουρός Προσωπικού Υπολογιστή.

4.1 Περιγραφή Συστήματος.

Το σύστημα αποτελείται από τη μονάδα του φρουρού η οποία βασίζεται σε μικροελεγκτή και από τον δαίμονα ο οποίος τρέχει στον φρουρούμενο προσωπικό υπολογιστή.

Γράφημα 7. Σύστημα Φρουρού Προσωπικού Υπολογιστή

Τα βασικά χαρακτηριστικά της λειτουργικότητας του συστήματος παρουσιάζονται παρακάτω:

Έλεγχος της ορθής λειτουργίας του pc.

Εφόσον ενεργοποιηθεί η λειτουργία φρούρησης, ο φρουρός ελέγχει περιοδικά (με περίοδο που ορίζεται από τον χρήστη του συστήματος) εάν ο προσωπικός υπολογιστής ανταποκρίνεται. Σε περίπτωση μη απόκρισης του υπολογιστή και ανάλογα με τις ρυθμίσεις που έχουν γίνει ο φρουρός μπορεί είτε να προκαλέσει την επανεκκίνηση του προσωπικού υπολογιστή, είτε να στείλει μήνυμα ειδοποίησης μέσω της δευτερεύουσας σύνδεσης, είτε και τα δύο.

Αναφορά κατάστασης λειτουργίας του pc

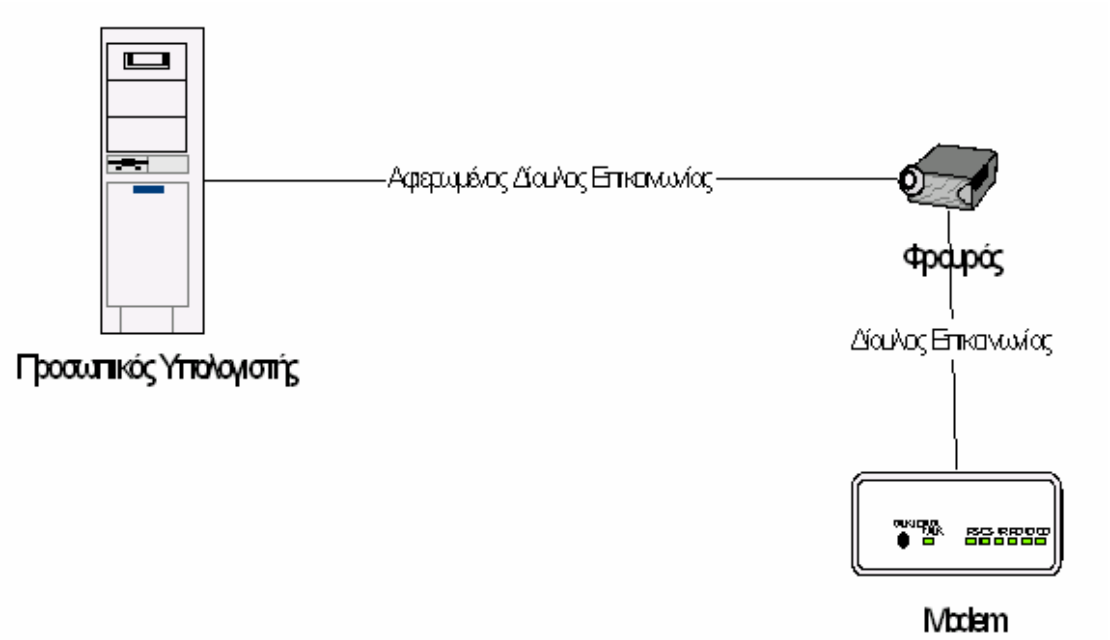
Ο φρουρός διαθέτει μηχανισμό ένδειξης που πληροφορεί τον χρήστη για την κατάσταση του προσωπικού υπολογιστή αλλά και προφανώς για την ορθή λειτουργία του φρουρού.

Ημερολόγιο συμβάντων

Ο φρουρός κρατάει στη μνήμη του ημερολόγιο, του οποίου το είδος των εγγραφών έχουν καθοριστεί από τις επιλογές του χρήστη. Το ημερολόγιο μπορεί να ανακτηθεί από τον ίδιο ή από άλλο υπολογιστή με σκοπό την διάγνωση των αιτιών της προβληματικής λειτουργίας του εποπτευόμενου προσωπικού υπολογιστή.

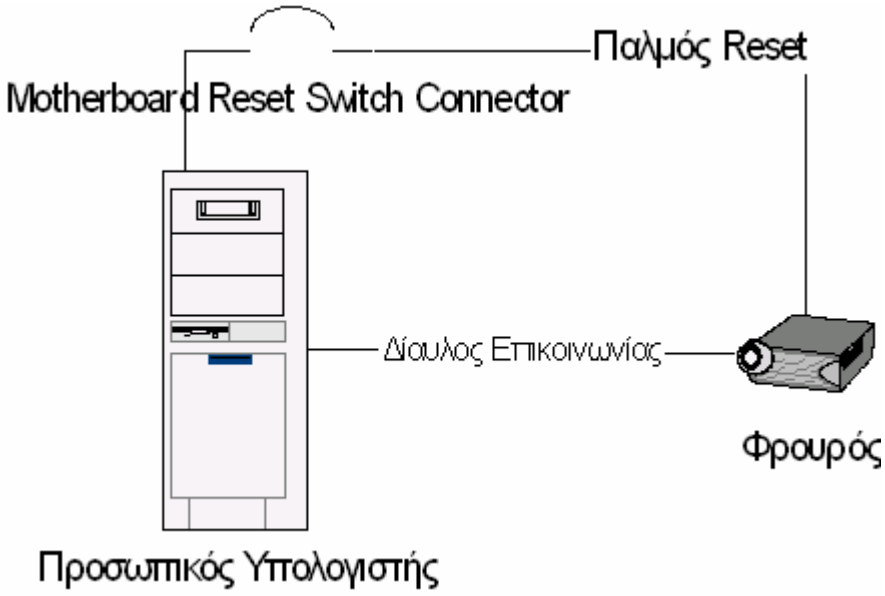
Λειτουργία Ειδοποίησης.

Η επικοινωνία του φρουρού με τον προσωπικού υπολογιστή γίνεται μέσω διαύλου επικοινωνίας, ο οποίος όμως είναι αφιερωμένος μόνο στην εξυπηρέτηση της επικοινωνίας μεταξύ φρουρού και εποπτευόμενου συστήματος. Για την λειτουργικότητα της εξωτερικής (δευτερεύουσας) από το σύστημα ειδοποίησης για προβλήματα, το σύστημα συνδέεται σε κάποια μονάδα επικοινωνίας π.χ. modem.



Γράφημα 8. Λειτουργία Ειδοποίησης out of band

Τέλος, για την περίπτωση λειτουργίας αυτόματης επανεκκίνησης του συστήματος προβλέπεται η παρακάτω συνδεσμολογία.



Γράφημα 9. Συνδεσμολογία επανεκκίνησης

Η λειτουργικότητα του φρουρού εμφανίζει τα παρακάτω πλεονεκτήματα:

1. Αύξηση της αξιοπιστίας του συστήματος με ιδιαίτερα χαμηλό κόστος (συγκριτικά πολύ μικρό με το κόστος του εποπτευόμενου συστήματος).
2. Υποστήριξη της δυνατότητας της ειδοποίησης ακόμη και σε περίπτωση ολικής αποτυχίας του προσωπικού υπολογιστή. Η αυτονομία του φρουρού αποτελεί συγκριτικό πλεονέκτημα σε σχέση με συστήματα, τα οποία εξαρτούν τη λειτουργικότητα της διασύνδεσης από την λειτουργία του προσωπικού υπολογιστή.
3. Διάθεση διαγνωστικών στοιχείων, τα οποία είναι προστατευμένα στη μνήμη του φρουρού και δεν επηρεάζονται από βλάβες στον προσωπικό υπολογιστή.

4.2 Καταστάσεις λειτουργίας.

Το σύστημα έχει διάφορες καταστάσεις λειτουργίας οι οποίες περιγράφονται στον πίνακα που ακολουθεί:

Πίνακας 3. Καταστάσεις Λειτουργίας

Όνομα	**Περιγραφή**
Παρακολούθηση λειτουργίας.	Το σύστημα εποπτεύει την λειτουργία του προσωπικού υπολογιστή και αναφέρει πιθανή δυσλειτουργία.
Παρακολούθηση λειτουργίας με ενεργοποιημένη την αυτόματη επανεκκίνηση.	Το σύστημα εποπτεύει την λειτουργία του προσωπικού υπολογιστή και αναφέρει πιθανή δυσλειτουργία. Σε περίπτωση μη απόκρισης του υπολογιστή το σύστημα προκαλεί την επανεκκίνηση του υπολογιστή.
Λειτουργία ημερολογίου.	Το σύστημα καταγράφει στοιχεία για την λειτουργία του προσωπικού υπολογιστή. (μπορεί να συνδυαστεί με μία από τις δυο προηγούμενες).
Λειτουργία αναφοράς.	Το σύστημα δίνει πρόσβαση στο ημερολόγιο της μνήμης για τις περιπτώσεις διάγνωσης βλαβών.

4.3 Βασικές αρχές λειτουργίας.

Η λειτουργία του συστήματός στηρίζεται στην υπόθεση πως εφόσον ο εποπτευόμενος προσωπικός υπολογιστής δουλεύει σωστά τότε μία από τις διεργασίες του (η οποία είναι η διεργασία που επικοινωνεί με τον φρουρό) θα απολαμβάνει επεξεργαστικό χρόνο ανά τακτικά χρονικά διαστήματα, ώστε να είναι ικανή να απαντά στους περιοδικούς ελέγχους που πραγματοποιεί ο φρουρός.

Τα παρακάτω στοιχεία από την αρχιτεκτονική των λειτουργικών συστημάτων Windows και Linux, ως τα πλέον αντιπροσωπευτικά στην χρήση τους σε προσωπικούς υπολογιστές παρατίθενται παρακάτω:

4.3.1 Λειτουργία των διακοπών στην αρχιτεκτονική του Windows™

Στα Windows και συγκεκριμένα στις βιβλιοθήκες των 32-bit εκδόσεων που περιλαμβάνουν τα Windows 2000, Windows XP και Windows 2003 προϊόντα ο γενικός μηχανισμός των interrupts βασίζεται στα Interrupt Request Levels (IRQLs).

Ένα IRQL μπορεί να πάρει τιμές από 0 έως 31. Κάθε IRQL αντιπροσωπεύει μια συγκεκριμένη προτεραιότητας πηγής διακοπών ώστε τελικά να διαμορφώνονται κατηγορίες πηγών όπως φαίνεται στον πίνακα 4 για την αρχιτεκτονική x86 της Intel.

Πίνακας 4. Windows Interrupt Request Level

31	High	Hardware interrupts
30	Power fail	
29	Inter-processor interrupt	
28	Clock	
27	Profile	
26	Device n	
.		
.		
.		
3	Device 1	
2	DPC / dispatch	Software interrupts
1	APC	
0	Passive	Normal thread execution

Έτσι κάθε πηγή διακοπών που δηλώνεται στο λειτουργικό σύστημα κατατάσσεται σε μία από τις παραπάνω κατηγορίες (οι οποίες σε άλλες αρχιτεκτονικές θα έχουν διαφορετική ταξινόμηση ανάλογα με τις σχεδιαστικές επιλογές του υλικού). Επιπλέον κάθε πηγή διακοπής λαμβάνει και μια εγγραφή στον πίνακα Interrupt Dispatch Table (IDT), ο οποίος αποτελεί τον χάρτη των διεργασιών εξυπηρέτησης διακοπών. Η κατηγορία 0 συμπεριλαμβάνει τις διακοπές του λογισμικού και τελικά σε αυτήν περιλαμβάνεται και η εξυπηρέτηση των διεργασιών του λογισμικού εφαρμογών (είτε αυτό βρίσκεται σε kernel mode είτε αυτό βρίσκεται σε user mode).

Το λειτουργικό σύστημα επιλέγει την σειρά εξυπηρέτησης των διακοπών με βάση τον παραπάνω πίνακα. Είναι προφανές πως όσο πιο χαμηλά βρίσκεται καταχωρημένη μια πηγή διακοπής, τόσο πιο ευαίσθητη είναι στην εμφάνιση δυσλειτουργιών αφού

προβλήματα στην κανονική ροή του συστήματος θα σημαίνουν την εμπλοκή των υψηλότερων πηγών διακοπών και συνεπώς την αδυναμία να εξυπηρετηθεί αυτή.

Συνεπώς η τοποθέτηση της διαγνωστικής διαδικασίας στην κατηγορία «0» εγγυάται πως οι υψηλότερες πηγές είχαν την ευκαιρία να διεκπεραιωθούν. Ο προσεκτικός όμως αναγνώστης θα παρατηρήσει πως αυτό δεν σημαίνει πως οι υψηλότερες πηγές θα διεκπεραιωθούν ορθά. Π.χ. μια διαδικασία write μπορεί να αποτύχει δίδοντας συνεχή write failures στο λειτουργικό σύστημα αλλά χωρίς να παγώσει η διαδικασία του dispatching. Γι' αυτό είναι αναγκαίο η λειτουργικότητα του φρουρού να ενισχυθεί με λογική διάγνωση της ορθής λειτουργίας των ζωτικών συστημάτων του υπολογιστή, όπως θα περιγραφεί παρακάτω.

Επιπλέον, εξαιτίας της αρχιτεκτονικής του λειτουργικού συστήματος των Windows, σύμφωνα με το οποίο η προτεραιότητα της I/O επικοινωνίας του επεξεργαστή βρίσκεται πάντοτε τουλάχιστον στο επίπεδο 3, η ίδια αυτή η πράξη της επικοινωνίας μεταξύ του φρουρού και της διεργασίας είναι διάγνωση της λειτουργίας του εποπτευόμενου συστήματος.

4.3.2 Λειτουργία των διακοπών στην αρχιτεκτονική του Linux ™

Το σύστημα διακοπών του Linux περιγράφεται συνοπτικά στον παρακάτω πίνακα:

Πίνακας 5. Intel Interrupt Table

Intel Vector Assigned #	Description	Linux Implementation
0..31	Exceptions Nonmaskable interrupts	Legacy
32..47	Maskable interrupts	Legacy
48.255	Software interrupts	128 (0x80) system calls system_call() kernel function

Ο αντίστοιχος πίνακας με τις διεργασίες που εξυπηρετούν τις διακοπές ονομάζεται Interrupt Descriptor Table.

Ο αλγόριθμος δρομολόγησης του linux βασίζεται στην διαμερισματοποίηση του χρόνου του επεξεργαστή σε περιόδους που ονομάζονται "epochs". Σε κάθε epoch, μια διεργασία έχει ένα συγκεκριμένο χρόνο επεξεργαστή του οποίου η διάρκεια υπολογίζεται –δυναμικά- όταν αρχίζει το epoch.

Πίνακας 6. Κλήσεις πυρήνα για την διαχείριση του χρονοπρογραμματισμού.

System Call	**Description**
nice( )	Change the priority of a conventional process.
getpriority( )	Get the maximum priority of a group of conventional processes.
setpriority( )	Set the priority of a group of conventional processes.
sched_getscheduler( )	Get the scheduling policy of a process.
sched_setscheduler( )	Set the scheduling policy and priority of a process.
sched_getparam( )	Get the scheduling priority of a process.
sched_setparam( )	Set the priority of a process.
sched_yield( )	Relinquish the processor voluntarily without blocking
sched_get_ priority_min( )	Get the minimum priority value for a policy.
sched_get_ priority_max( )	Get the maximum priority value for a policy.
sched_rr_get_interval( )	Get the time quantum value for the Round Robin policy.

Ο χρονοπρογραμματιστής (scheduler) του Linux επιλέγει την επόμενη διεργασία ανάλογα με την προτεραιότητα της. Υπάρχουν δυο ειδών προτεραιότητες:

Στατική Προτεραιότητα στην αρχιτεκτονική Linux ™

Μια στατική προτεραιότητα ανατίθεται στις διεργασίες από τον χρήστη – δημιουργό (creator owner) και παίρνει τιμές από 1 έως 99. Ο χρονοπρογραμματιστής ποτέ δεν αλλάζει τις τιμές των στατικών προτεραιοτήτων. Οι στατικές προτεραιότητες θα

πρέπει να ανατίθενται μόνο σε διεργασίες που είναι ευαίσθητες σε περιορισμούς πραγματικού χρόνου (real time processes).

Θα πρέπει να σημειωθεί εδώ πως παρόλο που οι παραπάνω διεργασίες αποκαλούνται διεργασίες πραγματικού χρόνου (real time processes) το Linux δεν μπορεί να χαρακτηριστεί λειτουργικό σύστημα πραγματικού χρόνου. Ειδικά μέχρι την έκδοση πυρήνα 2.2, η πλειοψηφία του κώδικα του πυρήνα ήταν non reentrant που σημαίνει πως δεν υπήρχε η δυνατότητα διακοπής της εκτέλεσης συγκεκριμένων διεργασιών του συστήματος χωρίς αυτές να τερματίσουν, αφαιρώντας έτσι την δυνατότητα για εξυπηρέτηση διακοπών σε πραγματικό χρόνο. Από την έκδοση 2.4 και μετά ο επανασχεδιασμός του πυρήνα με την χρήση της τεχνικής των “tasklets” περιόρισε το πρόβλημα κυρίως για να υποστηριχθεί η συμμετρική πολυεπεξεργασία (SMP, symmetric multiprocessing) χωρίς όμως και πάλι το Linux να κατατάσσεται στην κατηγορία των εγγυημένων λειτουργικών συστημάτων πραγματικού χρόνου (βλέπε [19] σελίδες 134-136) .

Δυναμική Προτεραιότητα στην αρχιτεκτονική Linux ™

Η δυναμική προτεραιότητα αναφέρεται στις κανονικές διεργασίες και εξαρτάται από διάφορους παράγοντες όπως ο χρόνος που έχει απομείνει στο τρέχον epoch, οι εναπομείνασες διεργασίες που περιμένουν εκτέλεση (blocked state), ο χρόνος που έχει ήδη ανατεθεί σε μια διεργασία στο τρέχον epoch κλπ. Προφανώς η χαμηλότερη στατική προτεραιότητα θα είναι πάντα μεγαλύτερη από την υψηλότερη δυναμική προτεραιότητα.

Προφανώς και στην αρχιτεκτονική του Linux η τοποθέτηση της διεργασίας διάγνωσης στον χώρο των δυναμικών προτεραιοτήτων αποτελεί την καλλίτερη λύση για την πλήρη παρακολούθηση της ομαλής λειτουργίας όλων των διεργασιών του συστήματος.

4.4 Διαγνωστικά δεδομένα.

Όπως αναφέρθηκε και προηγουμένως (§ 4.3.1) η επιβεβαίωση της λειτουργίας του επεξεργαστή δεν εγγυάται την ορθή λειτουργία του συστήματος. Γι' αυτό είναι αναγκαίο ο φρουρός να παρακολουθεί διάφορες μετρικές σχετικά με το σύστημα οι οποίες επιτρέπουν την διάγνωση τυχόν δυσλειτουργιών.

Εάν από αυτές τις μετρικές διαπιστωθεί βλάβη τότε ενεργοποιούνται οι δυνατότητες ανάνηψης ή / και ειδοποίησης που διαθέτει ο φρουρός με σκοπό την επαναφορά σε κανονική λειτουργία του συστήματος.

Τα βασικά διαγνωστικά δεδομένα περιγράφονται στον παρακάτω πίνακα:

Πίνακας 7. Διαγνωστικοί δείκτες.

Φόρτος Επεξεργαστή (%)	Ο φόρτος επεξεργαστή είναι το ποσοστό του χρόνου όπου ο επεξεργαστής εκτελεί ένα νήμα το οποίο δεν είναι άεργο. Ο δείκτης αυτός είναι ο κύριος δείκτης της δραστηριότητας του επεξεργαστή. Υπολογίζεται μετρώντας το χρόνο που ξοδεύει ο επεξεργαστής για να εκτελέσει το ειδικό νήμα με το όνομα "idle thread" και αφαιρώντας αυτόν τον χρόνο από το 100.
Χρόνος εξυπηρέτησης διακοπών υλικού (%)	Ο χρόνος εξυπηρέτησης διακοπών υλικού είναι το ποσοστό του χρόνου του επεξεργαστή όπου αυτός δέχεται και συνάμα εξυπηρετεί διακοπές υλικού. Ο δείκτης αυτός είναι ένας έμμεσος τρόπος να παρατηρηθεί ο φόρτος που προκαλείται στον επεξεργαστή από τις συσκευές που προκαλούν διακοπές όπως το ρολόι του συστήματος, οι γραμμές επικοινωνίας και η περιφερειακή αποθήκευση.
Διαθέσιμα Mbytes μνήμης.	Τα διαθέσιμα Mbytes μνήμης είναι το ποσό της φυσικής μνήμης που είναι διαθέσιμη στις διεργασίες που εκτελούνται στον επεξεργαστή σε Mbytes (Bytes / 1,048,576). Υπολογίζεται από το άθροισμα των μηδενισμένων, ελεύθερων και σε αναμονή (zeroed, free, stand by) λιστών μνήμης. Η ελεύθερη μνήμη είναι η μνήμη έτοιμη προς χρήση, η zeroed μνήμη είναι η μνήμη που έχει

	εκκαθαριστεί για λόγους ασφαλείας και η μνήμη αναμονής είναι η μνήμη που έχει μόλις αφαιρεθεί από το σύνολο εργασίας κάποιας διεργασίας (κατευθυνόμενη προς τον δίσκο για paging) αλλά διαθέσιμη για επαναφορά αν χρειαστεί.
Free Disk Space (%)	Ο μετρητής αυτός δείχνει τον διαθέσιμο ελεύθερο χώρο του δίσκου συστήματος.

Επιπλέον είναι αναγκαία και η διάγνωση της λειτουργίας της δικτυακής δραστηριότητα του συστήματος, ειδικά σε περιβάλλοντα κατανεμημένων συστημάτων.

Για την διάγνωση του δικτύου χρησιμοποιείται το πρωτόκολλο Internet Control Message Protocol προς την πύλη προς το δίκτυο του εποπτευόμενου συστήματος (default gateway). Το ICMP (βλέπε [21] για λεπτομέρειες) υλοποιήθηκε ως ένα εργαλείο για την διάγνωση και επιθεώρηση της λειτουργίας των δικτύων TCP/IP.

Τα βασικά στοιχεία που προσφέρει το ICMP είναι:
Έλεγχος συνδεσιμότητας.
Το ICMP, χρησιμοποιώντας έναν μηχανισμό challenge response δυο φάσεων, αποστέλλει ένα διαγνωστικό πακέτο στον επιθυμητό δικτυακό προορισμό. Εφόσον ο προορισμός είναι προσπελάσιμος και ενεργός είναι υποχρεωμένος να αποστείλει ένα πακέτο επιβεβαίωσης λειτουργίας πίσω στον αποστολέα.

Ο αποστολέας λαμβάνοντας την απόκριση του ICMP διασφαλίζει πως εκείνη τη στιγμή που λαμβάνεται η απόκριση υπάρχει ενεργό μονοπάτι προς τον προορισμό και πως ο προορισμός είναι ενεργός.

Ο έλεγχος συνδεσιμότητας είναι ο στοιχειωδέστερος διαγνωστικός έλεγχος που προσφέρει το ICMP.

Μέτρηση χρόνου απόκρισης.
Επιπλέον, το ICMP μπορεί να μετρήσει και τον χρόνο απόκρισης του προορισμού και πιο συγκεκριμένα τον round trip time, δηλαδή τον χρόνο που χρειάζεται ένα πακέτο

να αναχωρήσει από την πηγή, να φτάσει στον προορισμό και να επιστρέψει (ως επιβεβαίωση, acknowledgment) πίσω στην πηγή.

Η μέτρηση του χρόνου απόκρισης μπορεί να χρησιμέψει στην διάγνωση της ποιότητας της σύνδεσης μεταξύ πηγής και προορισμού. Αφενός υπάρχουν εφαρμογές οι οποίες έχουν συγκεκριμένες ανοχές στις χρονικές καθυστερήσεις της μεταγωγής των πακέτων. Οι διακυμάνσεις στις χρονικές καθυστερήσεις μπορούν να χρησιμοποιηθούν για να διαγνωστεί αστάθεια ή παροδικές βλάβες σε μια τηλεπικοινωνιακή ζεύξη.

Στην λύση που υλοποιείται, ο χρήστης του συστήματος μπορεί να ορίσει όρια στην χρονική καθυστέρηση του δικτύου ως προς κάποιο προορισμό, τα οποία όταν παραβιαστούν η σύνδεση χαρακτηρίζεται ως προβληματική.

Διαγνωστικά χαμηλού επιπέδου.
Το ICMP τέλος προσφέρει μια πληθώρα δικτυακών διαγνωστικών χαμηλού επιπέδου. Τα περισσότερα από αυτά χρησιμοποιούνται για τις ευαίσθητες ρυθμίσεις (fine tuning) των παραμέτρων λειτουργίας των τηλεπικοινωνιακών ζεύξεων που φιλοξενούν δίκτυα TCP/IP. Για τις ανάγκες όμως της διάγνωσης σφαλμάτων εξαιρετικά χρήσιμη είναι η μετρική του χρόνου εναπομένουσας ζωής του πακέτου (TTL: Time to Live). Η μετρική αυτή είναι ένας ακέραιος αριθμός, ο οποίος ανήκει στο πακέτο και διασχίζει το δίκτυο μαζί του. Κάθε φορά που το πακέτο διαπερνά έναν μεταγωγέα, το TTL πρέπει να μειωθεί τουλάχιστον κατά ένα. Αν και η βασική χρησιμότητα του TTL είναι η προστασία του δικτύου από ατέρμονα περιφερόμενα σε κύκλους πακέτα που θα πλημμύριζαν τα ενεργά στοιχεία δικτύου, η μέτρηση του TTL δίνει χρήσιμα στοιχεία για την μέτρηση των μηχανικών χαρακτηριστικών της σύνδεσης δικτύου.

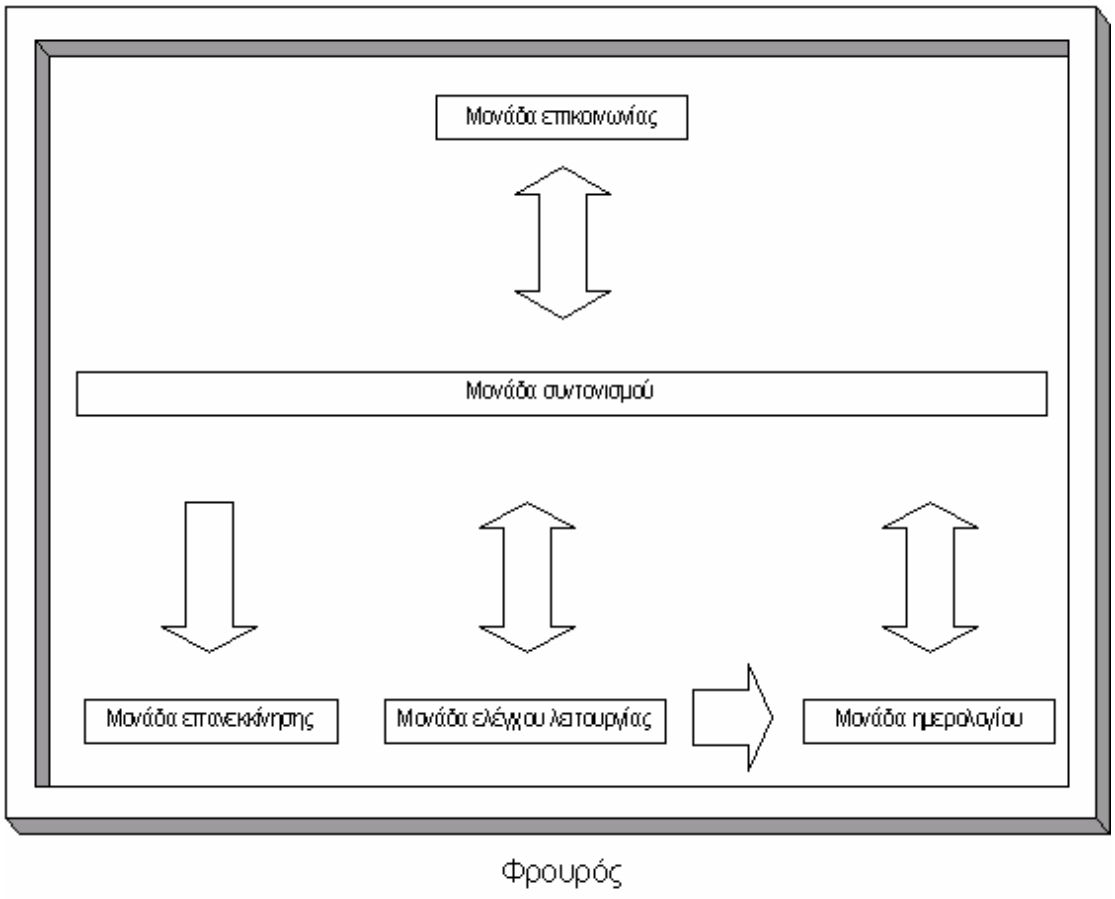
Στην λύση που υλοποιείται, το σύστημα ανιχνεύει την δυσλειτουργία της γραμμής εξαιτίας λήξης του TTL και ενημερώνει το χρήστη με κατάλληλο διαγνωστικό μήνυμα.

4.5 Λειτουργική Αποικοδόμηση του Λογισμικού.

Το λογισμικό του συστήματος διαχωρίζεται σε δυο περιοχές. Στο λογισμικό το οποίο τρέχει στον μικροελεγκτή καθώς και το λογισμικό το οποίο τρέχει πάνω στον προσωπικό υπολογιστή. Η επικοινωνία των δυο υποσυστημάτων λογισμικού γίνεται με την χρήση του πρωτοκόλλου που ορίζεται παρακάτω.

4.5.1 Λογισμικό Μικροελεγκτή.

Το λογισμικό του μικροελεγκτή έχει ως αρμοδιότητα την υλοποίηση των εργασιών που αντιστοιχούν στα υποσυστήματα που περιγράφονται στον πίνακα 3. Στο γράφημα 10 παρουσιάζεται η λειτουργική σύνδεση των εν λόγω μονάδων.



Γράφημα 10. Τοπολογία Λειτουργικών μονάδων.

Οι λειτουργικές μονάδες του συστήματος περιγράφονται παρακάτω:

Πίνακας 8. Λειτουργικές Μονάδες Συστήματος

Όνομα	**Περιγραφή**
Μονάδα επικοινωνίας	Είναι υπεύθυνη για την επικοινωνία με τον φρουρούμενο υπολογιστή καθώς και με τις τυχόν συνδεόμενες συσκευές για τον σκοπό της ειδοποίησης. Η μονάδα έχει την δυνατότητα να ελέγξει δυο κανάλια επικοινωνίας.
Μονάδα ελέγχου λειτουργίας.	Η μονάδα αυτή είναι υπεύθυνη για την επιβεβαίωση λειτουργίας του φρουρούμενου υπολογιστή. Η μονάδα έχει διαγνωστικούς μηχανισμούς για την έναρξη λειτουργίας του συστήματος καθώς και για την παρακολούθηση της σταθερής λειτουργίας του.
Μονάδα ημερολογίου	Το υποσύστημα της μονάδας ημερολογίου είναι υπεύθυνο για την διατήρηση του ημερολογίου συμβάντων.
Μονάδα επανεκκίνησης	Το υποσύστημα αυτό είναι υπεύθυνο για την επανεκκίνηση του προσωπικού υπολογιστή. Η μονάδα θα πρέπει να έχει τη δυνατότητα να παράγει τον κατάλληλο παλμό που προβλέπεται από το ACPI (Advanced Configuration and Power Interface) Specification.
Μονάδα ελέγχου	Το υποσύστημα αυτό είναι υπεύθυνο για την λειτουργία των υπόλοιπων μονάδων.

Όπως φαίνεται στο σχήμα 10 η μονάδα επικοινωνίας ανταλλάσσει δεδομένα με την μονάδα συντονισμού ώστε να επιτυγχάνεται η επικοινωνία με τον προσωπικό υπολογιστή. Η μονάδα συντονισμού μπορεί να ζητήσει από την μονάδα επανεκκίνησης να ενεργοποιηθεί. Επίσης η μονάδα συντονισμού ανταλλάσσει δεδομένα με τις μονάδες ελέγχου λειτουργίας και ημερολογίου. Τέλος η μονάδα

ελέγχου λειτουργίας έχει την δυνατότητα να δρομολογήσει για εγγραφή στην μονάδα ημερολογίου τα γεγονότα που θεωρεί απαραίτητο να καταγραφούν στο ημερολόγιο.

4.5.2 Λογισμικό Προσωπικού Υπολογιστή.

Το λογισμικό του προσωπικού υπολογιστή χωρίζεται στο λογισμικό διαχείρισης και το λογισμικό λειτουργίας.

Λογισμικό Διαχείρισης.

Μέσω του λογισμικού διαχείρισης δίνεται στον χρήστη η δυνατότητα να ορίσει τις διάφορες παραμέτρους λειτουργίας του συστήματος. Στον πίνακα 9 περιγράφονται οι λειτουργίες διαχείρισης που έχει στη διάθεσή του ο χρήστης:

Πίνακας 9. Παράμετροι Διαχείρισης Συστήματος.

Λειτουργία	**Πεδίο Τιμών**	**Περιγραφή**
Ενεργοποίηση Φρουρού	Ενεργή ή ανενεργή.	Ο φρουρός τίθεται σε λειτουργία.
Ενεργοποίηση Επανεκκίνησης.	Ενεργή ή ανενεργή.	Σε περίπτωση μη απόκρισης του προσωπικού υπολογιστή, ο φρουρός θα επαννεκινήσει τον προσωπικό υπολογιστή.
Μάσκα ελέγχου Επανεκκίνησης	0 έως 127	Καθορίζει τις διαγνωστικές πηγές που προκαλούν την επανεκκίνηση του συστήματος.
Ορισμός Αριθμού Προσπαθειών Επανεκκίνησης	1 έως 255	Ο αριθμός των διαδοχικών επανεκκινήσεων που θα πραγματοποιήσει το σύστημα σε περίπτωση επαναλαμβανόμενης αποτυχίας του συστήματος.
Ορισμός Χρονικού Ορίου Διαδοχικής Επανεκκίνησης	0 έως 65536	Ο αριθμός των δευτερολέπτων που διαρκεί το διάστημα ανάμεσα στο οποίο αν συμβούν δυο αποτυχίες του συστήματος, χαρακτηρίζονται διαδοχικές.

Ορισμός Χρόνου Αδρανούς Λειτουργίας κατά την επανεκκίνηση.	0 έως 65536	Ο αριθμός των δευτερολέπτων που παραμένει ανενεργός ο φρουρός μετά την αποστολή του παλμού επανεκκίνησης.
Ορισμός Περιόδου Ελέγχου Συστήματος	0 έως 65536	Ο χρόνος μεταξύ δυο διαδοχικών ελέγχων του συστήματος σε δέκατα του δευτερολέπτου.
Ορισμός Ομάδας Εντολών Οδήγησης Εξωτερικής Συσκευής.	Συμβολοσειρά	Συμβολοσειρά μεγέθους από 0 έως 120 bytes, η οποία θα αποσταλθεί στην εξωτερική συσκευή με σκοπό την ειδοποίηση του χρήστη.
Μάσκα ελέγχου Ειδοποίησης	0 έως 127	Καθορίζει τις διαγνωστικές πηγές που ενεργοποιούν την διαδικασία ειδοποίησης του χρήστη.
Ημερολόγιο Φόρτου Επεξεργαστή	Ενεργή ή ανενεργή.	Ενεργοποιείται η παρακολούθηση του φόρτου του επεξεργαστή.
Στάθμη καταγραφής φόρτου.	0 έως 100	Ποσοστό φόρτου επεξεργαστή το οποίο προκαλεί την εγγραφή συμβάντος στο ημερολόγιο.
Ημερολόγιο Χρόνου Εξυπηρέτησης Διακοπών	Ενεργή ή ανενεργή.	Ενεργοποιείται η παρακολούθηση του χρόνου εξυπηρέτησης διακοπών.
Στάθμη καταγραφής χρόνου εξυπηρέτησης διακοπών.	0 έως 100	Ποσοστό χρόνου που εξυπηρετούνται διακοπές, ο οποίος προκαλεί την εγγραφή συμβάντος στο ημερολόγιο.
Ημερολόγιο Mbytes Μνήμης	Ενεργή ή ανενεργή.	Ενεργοποιείται η παρακολούθηση των ελεύθερων MByte της μνήμης
Στάθμη καταγραφής Mbytes	0 έως 100	Αριθμός ελεύθερων Mbytes που προκαλούν την εγγραφή συμβάντος στο ημερολόγιο.
Ημερολόγιο Ελεύθερου Disk Space	Ενεργή ή ανενεργή.	Ενεργοποιείται η παρακολούθηση του ποσοστού ελεύθερου χώρου στο δίσκο.

Στάθμη Ελεύθερου Disk Space	0 έως 100	Ποσοστό ελεύθερου χώρου στο δίσκο που προκαλεί εγγραφή συμβάντος στο ημερολόγιο.
Ενεργοποίηση Παρακολούθησης Δικτύου.	Ενεργή ή Ανενεργή.	Παρακολούθηση συνδεσιμότητας με συγκεκριμένο κόμβο του δικτύου.
Όριο Χρονικής Καθυστέρησης Δικτύου.	0 έως 65536	Ο μέγιστος αριθμός msec χρονικής καθυστέρησης ώστε η σύνδεση να μην χαρακτηρίζεται προβληματική.
Έλεγχος Λήξης TTL.	Ενεργή ή Ανενεργή.	Παρακολούθηση επάρκειας TTL.

Επιπλέον της δυνατότητας του χειρισμού των παραπάνω παραμέτρων, το λογισμικό διαχείρισης επιτρέπει και την επικοινωνία με τον φρουρό για την ανάγνωση του ημερολογίου του (log bulk download).

Λογισμικό Λειτουργίας.

Το λογισμικό λειτουργίας (το οποίο στην γλώσσα των λειτουργικών συστημάτων ονομάζεται service στα Windows και daemon στο Linux) έχει την ευθύνη της διαγνωστικής λειτουργίας σε δυο επίπεδα.

Στο πρώτο επίπεδο ανταποκρίνεται στους ελέγχους του φρουρού, απαντώντας κάθε φορά στο πακέτο επιβεβαίωσης λειτουργίας που στέλνει ο φρουρός μέσω του αφιερωμένου καναλιού επικοινωνίας.

Στο δεύτερο επίπεδο φροντίζει για την πραγματοποίηση όλων των μετρήσεων που παράγουν τα βοηθητικά διαγνωστικά στοιχεία, ελέγχει εάν ικανοποιούνται τα όρια των παραμέτρων που έχει ορίσει ο χρήστης και ενημερώνει αντίστοιχα την μνήμη του ημερολογίου στον φρουρό.

4.6 Προδιαγραφές Λειτουργίας Συστήματος.

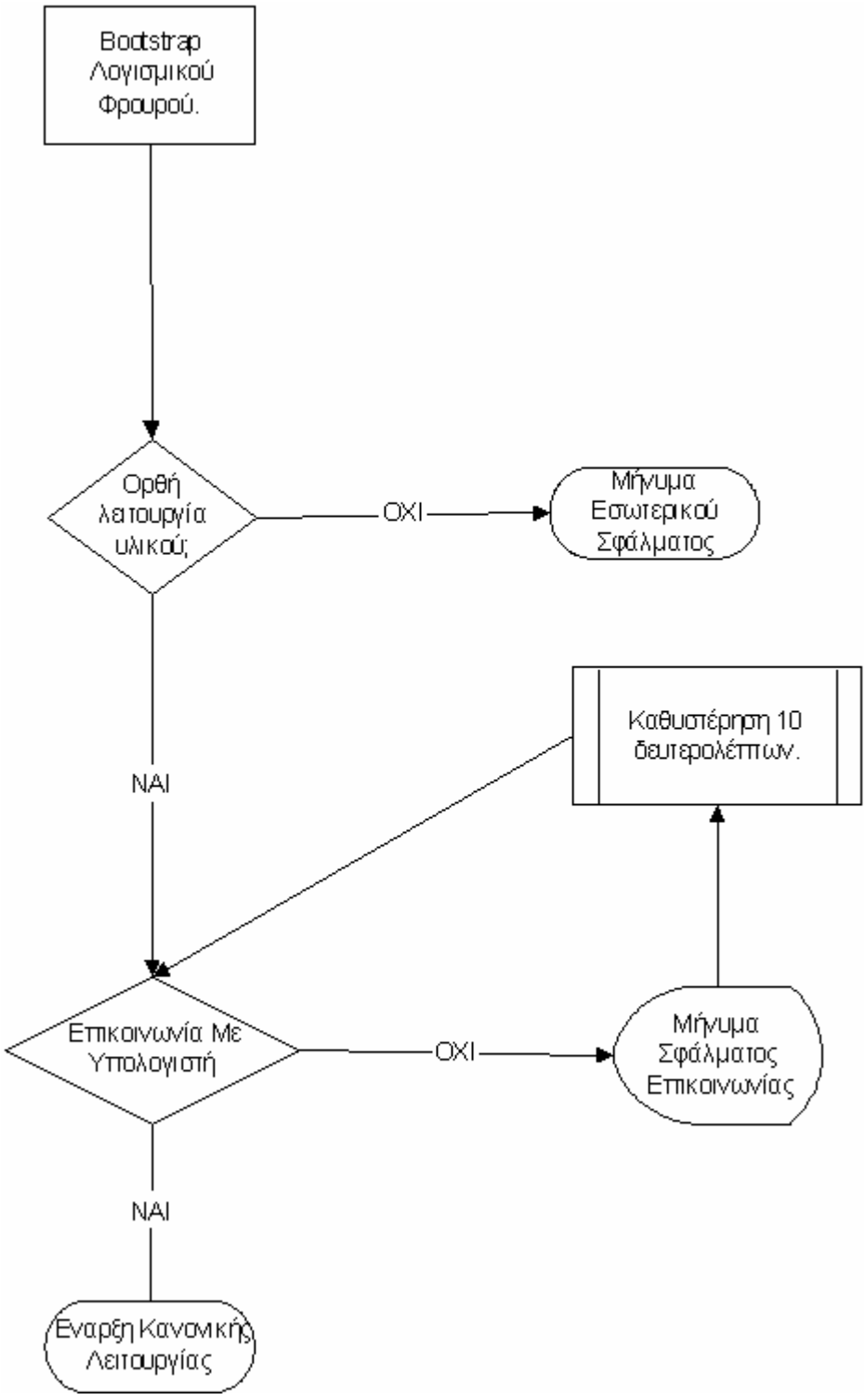
4.6.1 Διαγράμματα Λειτουργίας Λογισμικού

Όταν ο φρουρός τεθεί σε λειτουργία και αφού εκτελέσει τις διαγνωστικές λειτουργίες για την ορθή λειτουργία του, ξεκινάει την προσπάθεια της επικοινωνίας με τον προσωπικό υπολογιστή. (Γράφημα 11) Εάν ο προσωπικός υπολογιστής δεν είναι διαθέσιμος, γίνεται προσπάθεια επανασύνδεσης κάθε δέκα δευτερόλεπτα.

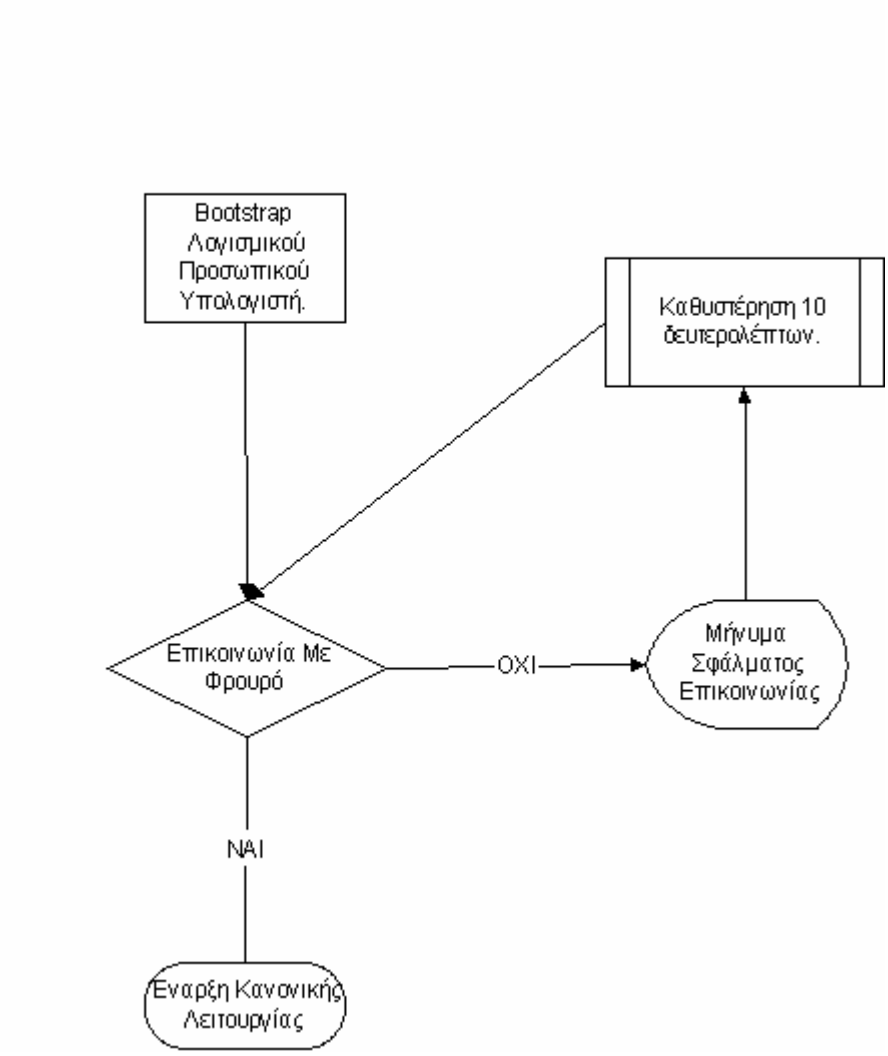
Ομοίως ο δαίμονας (Γράφημα 12) που τρέχει στον προσωπικό υπολογιστή αναμένει τη σύνδεση με τον φρουρό. Κάθε δέκα δευτερόλεπτα αναμονής της σύνδεσης, ο δαίμονας ενημερώνει με μήνυμα λάθους τον χρήστη πως η σύνδεση δεν είναι αποκατεστημένη.

Η κανονική έναρξη λειτουργίας του συστήματος (ώστε να μην εμφανιστεί μήνυμα λάθους στον χρήστη) προϋποθέτει πως πρώτα θα τεθεί σε λειτουργία ο φρουρός και μετά ο ηλεκτρονικός υπολογιστής. **Παρόλα αυτά όμως** από τα διαγράμματα που περιγράφηκαν είναι προφανές πως το σύστημα θα μπορέσει να ξεκινήσει τη λειτουργία του ακόμη και αν συμβεί το αντίθετο. Στην τελευταία περίπτωση, ο χρήστης του προσωπικού υπολογιστή θα λάβει κάποιο ή κάποια προειδοποιητικά μηνύματα έως ότου να αποκατασταθεί η επικοινωνία των δυο συστημάτων.

Εφόσον γίνει η έναρξη της κανονικής λειτουργίας του συστήματος τότε τον έλεγχο έχει ο δαίμονας που τρέχει στον προσωπικό υπολογιστή. Ο δαίμονας είναι υπεύθυνος για να μεταφέρει όλες τις κατάλληλες παραμέτρους στον φρουρό και προφανώς να ενεργοποιήσει την λειτουργία της επανεκκίνησης εφόσον αυτό χρειαστεί.



Γράφημα 11. Διάγραμα Έναρξης Λειτουργίας Φρουρού



Γράφημα 12. Διάγραμμα Έναρξης Λειτουργίας Δαίμονα

Η κεντρική λειτουργία του φρουρού περιγράφεται από το διάγραμμα ροής του γραφήματος 13. Στον παρακάτω πίνακα επεξήγονται τα σύμβολα του γραφήματος.

Τ_Ελέγχου	Μετράει το χρονικό διάστημα μεταξύ δυο επανελέγχων του συστήματος.
Μ_ΔΕ	Μετράει το χρονικό διάστημα μεταξύ δυο επανεκκινήσεων.
Διαδοχικές Επανεκκινήσεις	Κρατάει τον αριθμό των τελευταίων διαδοχικών επανεκκινήσεων.
Αριθμός Προσπαθειών	Παράμετρος του χρήστη σύμφωνα με τον πίνακα 5.

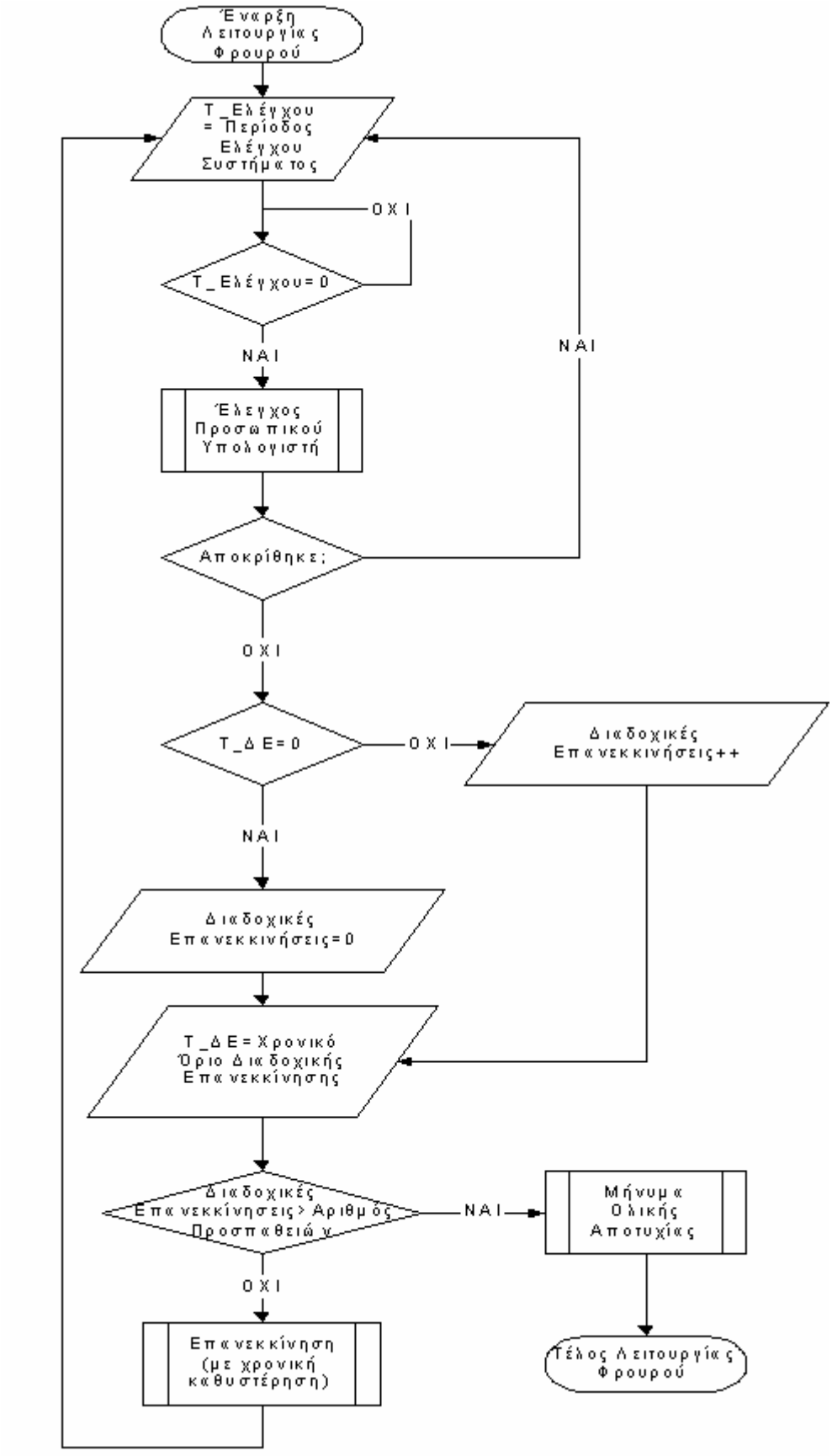
Περίοδος Ελέγχου Συστήματος	-//-
Χρονικό Όριο Διαδοχικών Επανεκκινήσεων	-//-

Το σύστημα ξεκινά τη λειτουργία του αρχικοποιώντας τις μεταβλητές Μ_ΔΕ και Διαδοχικές Επανεκκινήσεις στο μηδέν αφού με την ενεργοποίηση του συστήματος δεν έχει υπάρξει καμία προηγούμενη επανεκκίνηση.

Με την χρήση του χρονιστή Τ_Ελέγχου υλοποιείται η απαίτηση για έλεγχο του συστήματος σύμφωνα με την παράμετρο Περίοδος Ελέγχου Συστήματος που έχει ορίσει ο χρήστης. Αφού το συγκεκριμένο χρονικό διάστημα παρέλθει ο φρουρός ελέγχει τη λειτουργία του προσωπικού υπολογιστή. Εάν αυτός αποκριθεί τότε ξεκινάει ένας ακόμη κύκλος μέτρησης του Τ_Ελέγχου.

Σε αντίθετη περίπτωση ελέγχεται μέσω του Τ_ΔΕ εάν έχει παρέλθει το ικανό χρονικό διάστημα από την προηγούμενη επανεκκίνηση ώστε η επερχόμενη επανεκκίνηση να μην χαρακτηρισθεί διαδοχική. Ανάλογα με την περίπτωση ενημερώνεται και η τιμή της μεταβλητές «Διαδοχικές Επανεκκινήσεις». Ο Τ_ΔΕ αρχικοποιείται πάλι.

Η λειτουργία του φρουρού ολοκληρώνεται με τον έλεγχο των διαδοχικών επανεκκινήσεων. Εάν δεν έχει ξεπεραστεί το όριο που έχει τεθεί ως παράμετρος από τον χρήστη τότε πραγματοποιείται επανεκκίνηση του συστήματος και ο φρουρός επιστρέφει στην λειτουργία του Τ_Ελέγχου. Διαφορετικά καταγράφεται ολική αποτυχία του συστήματος και ο φρουρός τερματίζει τη λειτουργία του στο επίπεδο παρακολούθησης του συστήματος.



Γράφημα 13. Διάγραμμα Λειτουργίας Φρουρού

4.6.2 Πρωτόκολλα Επικοινωνίας.

Η επικοινωνία μεταξύ του φρουρού και του προσωπικού υπολογιστή γίνεται με το πρωτόκολλο που περιγράφεται στους παρακάτω πίνακες.

Στον πίνακα 10 περιγράφονται τα μηνύματα που ενεργοποιούνται από τον δαίμονα ή το λογισμικό διαχείρισης που βρίσκεται στον προσωπικό υπολογιστή.

Πίνακας 10. Πρωτόκολλο επικοινωνίας δαίμονα – φρουρού.

Όνομα	**Εντολή**	**Απόκριση**	**Περιγραφή**
Watchdog Disable	WDD	OK	Ο φρουρός δεν εποπτεύει το σύστημα.
Watchdog Enable	WDE	OK	Ο φρουρός εποπτεύει το σύστημα.
Write Log Entry	LW$3b$	OK	Εισαγωγή εγγραφής στο ημερολόγιο του φρουρού.
Write Log Entry	LW$3b$	FL	Η εισαγωγή εγγραφής στο ημερολόγιο απέτυχε λόγω πλήρους ημερολογίου. Ο δαίμονας πρέπει να φροντίσει για την διαγραφή των εγγραφών του ημερολογίου.
Flush Log	LFA	OK	Διαγραφή όλων των εγγραφών από το ημερολόγιο.
Retrieve Log	LRA	L	Φόρτωση στον δαίμονα όλων των εγγραφών του ημερολογίου.
Enable Notify	WNE	OK	Ενεργοποίηση ειδοποίησης.
Disable Notify	WND	OK	Απενεργοποίηση ειδοποίησης.
Set Notify String	WNSS	OK	Ορισμός συμβολοσειράς ελέγχου της ειδοποίησης.
Set Checking Intervals	WDI$2b$	OK	Ορισμός παραμέτρου *Περίοδος Ελέγχου Συστήματος*.
Set Consecutive	WDR$1b$	OK	Ορισμός παραμέτρου *Αριθμός*

Reboots			*Προσπαθειών Επανεκκίνησης*
Check Health	WDH	OK	Ο φρουρός λειτουργεί κανονικά.
Check Health	WDH	FL	Ο φρουρός παρουσιάζει σφάλμα λειτουργίας.
Maximum Delay Between Failures for Consecutive Mode	WTD$2b$	OK	Ορισμός παραμέτρου *Χρονικό όριο διαδοχικής επανεκκίνησης.*
Set Bootstrap Delay	WDB$2b$	OK	Ορισμός παραμέτρου *Χρόνος αδρανούς λειτουργίας κατά την επανεκκίνηση.*

Ο πίνακας 11 καταγράφει τα μηνύματα που αποστέλλονται από τον φρουρό προς τον δαίμονα. Τα μηνύματα αυτά εξυπηρετούν μόνο την διαδικασία του bootstrapping του συστήματος στις βαθμίδες αποκατάστασης επικοινωνίας και επιβεβαίωσης ορθής λειτουργίας.

Πίνακας 11. Πρωτόκολλο επικοινωνίας φρουρού - δαίμονα.

Όνομα	**Εντολή**	**Απόκριση**	**Περιγραφή**
Check Communication	PNG	OK	Επιβεβαίωση επικοινωνίας με τον προσωπικό υπολογιστή.
Check Availability	AYR	OK	Επιβεβαίωση λειτουργίας δαίμονα.
Check Availability	AYR	FL	Ο δαίμονας δεν είναι διαθέσιμος.

Τέλος παρατίθενται οι καταστάσεις του συστήματος για τις οποίες θα πρέπει να λαμβάνει γνώση ο χρήστης από τον φρουρό με την χρήση κάποιου κατάλληλου μέσου.

Πίνακας 12. Πίνακας μηνυμάτων προς τον χρήστη.

Μήνυμα.
Μονάδα φρουρού σε τροφοδοσία και πλήρως λειτουργική.
Εσωτερικό σφάλμα μονάδας φρουρού.
Σφάλμα επικοινωνίας με προσωπικό υπολογιστή.

Αποκατεστημένη επικοινωνίας με προσωπικό υπολογιστή.
Λειτουργία Επανεκκίνησης Ενεργοποιημένη.
Ο φρουρός έχει ήδη προκαλέσει επανεκκίνηση της μονάδας τουλάχιστον μια φορά.
Ολικό σφάλμα συστήματος. (Προκλήθηκαν όλες οι επιτρεπόμενες διαδοχικές επανεκκινήσεις.)

5. Υλοποίηση.

5.1 Βασική Περιγραφή

Το σύστημα που υλοποιήθηκε χρησιμοποιεί το πρωτόκολλο RS232 (EIA232) για την επικοινωνία του φρουρού με τον ηλεκτρονικό υπολογιστή αλλά και για την επικοινωνία με την συσκευή ειδοποίησης.

Συγκεκριμένα ο προσωπικός υπολογιστής συνδέεται μέσω της σειριακής του θύρας με την θύρα του φρουρού που φέρει την ονομασία "PC". Εάν είναι επιθυμητό να χρησιμοποιηθεί και κάποια συσκευή ειδοποίησης, αυτή θα συνδεθεί στην θύρα "COM" που διαθέτει ο φρουρός.

Ακόμη για την υλοποίηση της λειτουργίας της επανεκκίνησης, ο φρουρός θα πρέπει να συνδεθεί στον reset switch connector της μητρικής πλακέτας του προσωπικού υπολογιστή που εποπτεύει.

Για την εμφάνιση της διαγνωστικής πληροφορίας χρησιμοποιούνται τρία LED. Κάθε ένα από τα LED φέρει μια από τις παρακάτω ονομασίες:

- Power On LED
- On Line LED
- Watchdog Enabled LED

Αφού ολοκληρωθεί η σύνδεση του συστήματος, όλες οι διαχειριστικές εργασίες και οι εργασίες χειρισμού εξυπηρετούνται από το λογισμικό του συστήματος.

Για την άμεση ενημέρωση του χρήστη για την κατάσταση του συστήματος (health monitor) χρησιμοποιούνται τα ενδεικτικά led του συστήματος με τον συμβολισμό που περιγράφεται στον πίνακα 13.

Πίνακας 13. Καταστάσεις διαγνωστικών LED.

Κατάσταση LED	**Περιγραφή**
Powen On φωτοβολεί σταθερά	Ο φρουρός συνδέθηκε με την τροφοδοσία.
Power On φωτοβολεί διακεκομμένα	Ο φρουρός εμφάνισε εσωτερικό σφάλμα στην λειτουργία του.
On Line φωτοβολεί σταθερά	Ο φρουρός αποκατέστησε την επικοινωνία με τον προσωπικό υπολογιστή.
On Line φωτοβολεί διακεκομμένα	Ο φρουρός δεν έχει αποκαταστήσει την επικοινωνία με τον προσωπικό υπολογιστή και επαναλαμβάνει την προσπάθεια κάθε δέκα δευτερόλεπτα.
Watchdog Enabled φωτοβολεί σταθερά	Ο φρουρός έχει ενεργοποιηθεί από τον χρήστη. Το σύστημα είναι σε πλήρη λειτουργία.
Watchdog Enabled φωτοβολεί διακεκομμένα.	Ο φρουρός έχει ενεργοποιηθεί από τον χρήστη. Το σύστημα είναι σε πλήρη λειτουργία και έχει ήδη εξαναγκάσει σε επανεκκίνηση τον προσωπικό υπολογιστή τουλάχιστον μια φορά.
Και τα τρία LED φωτοβολούν διακεκομμένα.	Το σύστημα έκανε τον μέγιστο επιτρεπτό αριθμό διαδοχικών επανεκκινήσεων.

5.2 Περιγραφή Χαρακτηριστικών AVR ATmega161

Για την υλοποίηση του συστήματος χρησιμοποιήθηκε ο μικροελεγκτής της οικογένειας AVR 8-bit της εταιρίας ATMEL ATmega161.

Τα παρακάτω χαρακτηριστικά επιβεβαιώνουν την επάρκεια του μικροελεγκτή ATmega161 για τις απαιτήσεις του προβλήματος που αντιμετωπίζει η παρούσα εργασία.

Δυνατότητα Αυτό-προγραμματισμού.

Ο ATmega161 διαθέτει μηχανισμό αυτό-προγραμματισμού, μια λειτουργικότητα που επιτρέπει την εύκολη αναβάθμιση του λογισμικού στις νεότερες εκδόσεις.

Δυο Ενσωματωμένες UART Θύρες.

Η διάθεση των δυο UART θυρών εντός του ολοκληρωμένου προσφέρει ένα εξαιρετικά σημαντικό πλεονέκτημα αξιοπιστίας της σχεδίασης αφού το φυσικό επίπεδο της επικοινωνίας είναι on-chip εξαλείφοντας τα προβλήματα που εμφανίζονται από τις παρεμβολές κατά την οδήγηση εξωτερικών κυκλωμάτων επικοινωνίας από μικροελεγκτές. Αυτός είναι και ο λόγος για τον οποίο δεν επιλέχθηκε στην σχεδίαση η χρήση ενός μικρότερου επεξεργαστή AVR με μία μόνο μονάδα, στον οποίο θα συνδεόταν μια εξωτερική UART.

Επάρκεια Χρονιστών.

Ο ATmega161 διαθέτει επαρκείς 8-bit και 16-bit χρονιστές, οι οποίοι χρησιμοποιούνται για τις ανάγκες της σχεδίασης.

Εσωτερικός Watchdog Timer.

Ο ATmega161 διαθέτει εσωτερικό watchdog timer για την παρακολούθηση της λειτουργίας του συστήματος, προσφέροντας ακόμη μια ασφαλιστική δικλείδα στην εγγύηση της λειτουργίας του συστήματος.

Δυνατότητα Σύνδεσης με Εξωτερική Μνήμη.

Ο ATmega161 έχει τη δυνατότητα σύνδεσης με εξωτερική μνήμη που φτάνει το μέγεθος 63K x 8. Η μνήμη αυτή είναι ικανοποιητική για την λειτουργία του ημερολογίου προσφέροντας χωρητικότητα από έξι έως δέκα πέντε χιλιάδες εγγραφές αναλόγως με την λεπτομέρεια των εγγραφών.

Κατάσταση Εξοικονόμησης Ενέργειας.

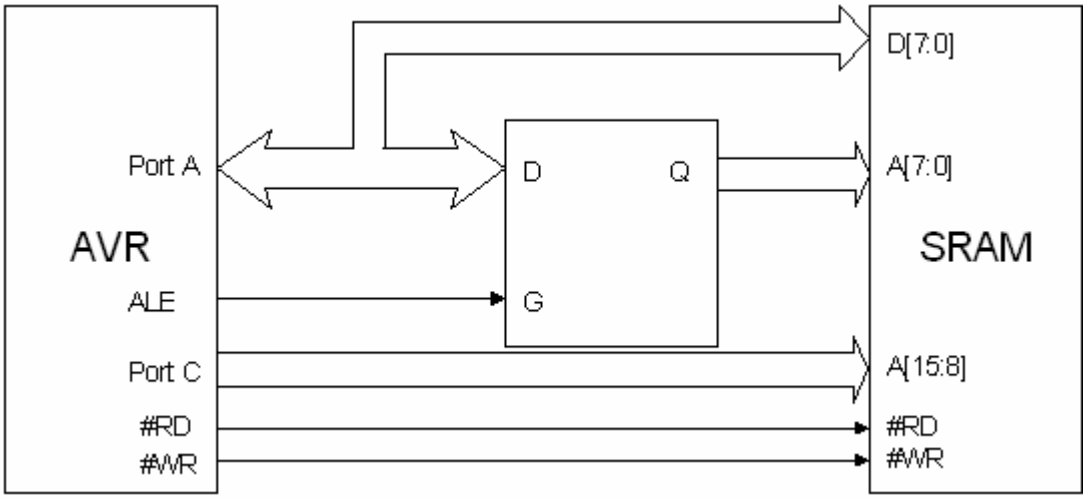
Ο ATmega161 έχει τη δυνατότητα λειτουργίας σε κατάσταση εξοικονόμησης ενέργειας (Sleep, Idle Mode). Η δυνατότητα αυτή στο χώρο της συγκεκριμένης υλοποίησης, που προτείνεται αποδεικνύεται εξαιρετικά οικονομική σε κατανάλωση ενέργειας.

5.3 Σχεδίαση Υλικού.

Το σχεδιάγραμμα του συστήματος παρουσιάζεται αναλυτικά στο σχήμα 15.

Χρήση της Εξωτερικής Μνήμης από τον ATmega161

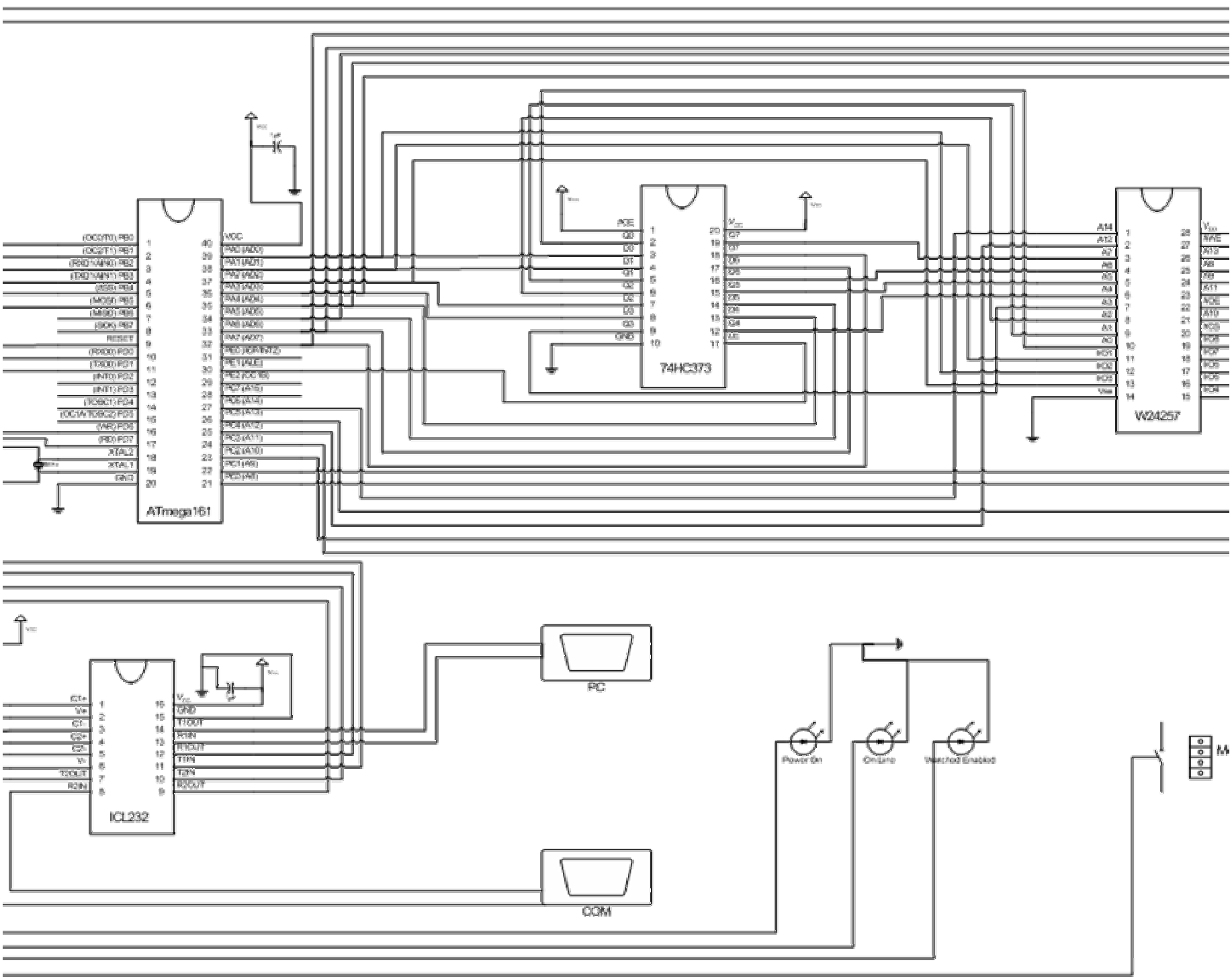
Για την επικοινωνία με την εξωτερική μνήμη ο ATmega161 χρησιμοποιεί δυο θύρες (Port A Και Port C) των 8 bit η κάθε μία. Το port C χρησιμοποιείται αποκλειστικά για τα bit 15:8 της διεύθυνσης ενώ το port A χρησιμοποιείται για την φόρτωση στη μνήμη των ύπολοιπων bit 7:0 διεύθυνσης και μετά για τα bit 7:0 δεδομένων. Για την διάκριση των δυο λειτουργιών του port A πληροφορεί η έξοδος ALE του AVR. Η σχηματική αναπαράσταση της επικοινωνίας του AVR με την εξωτερική μνήμη παρουσιάζεται στο γράφημα 14.



Γράφημα 14. Χρήση της εξωτερικής μνήμης από τον ATmega161.

Χρήση του UART.

Οι θύρες επικοινωνίας του AVR είναι Universal Asychronous Receiver Transmitter και λειτουργούν στην στάθμη της λειτουργίας του TTL. Η ενίσχυση του σήματος στην στάθμη των +/- 12V ώστε να συμμορφώνεται στο πρότυπο RS-232 γίνεται με την χρήση του ολοκληρωμένου ICL232 του οποίου το κύριο χαρακτηριστικό είναι οι ενσωματωμένοι μετατροπείς τάσης (οι οποίοι λειτουργούν χρησιμοποιώντας πηγή 5V). Έτσι δεν χρειάζεται η υλοποίηση ξεχωριστής γραμμής τροφοδοσίας στα 12V για τις ανάγκες του RS232.



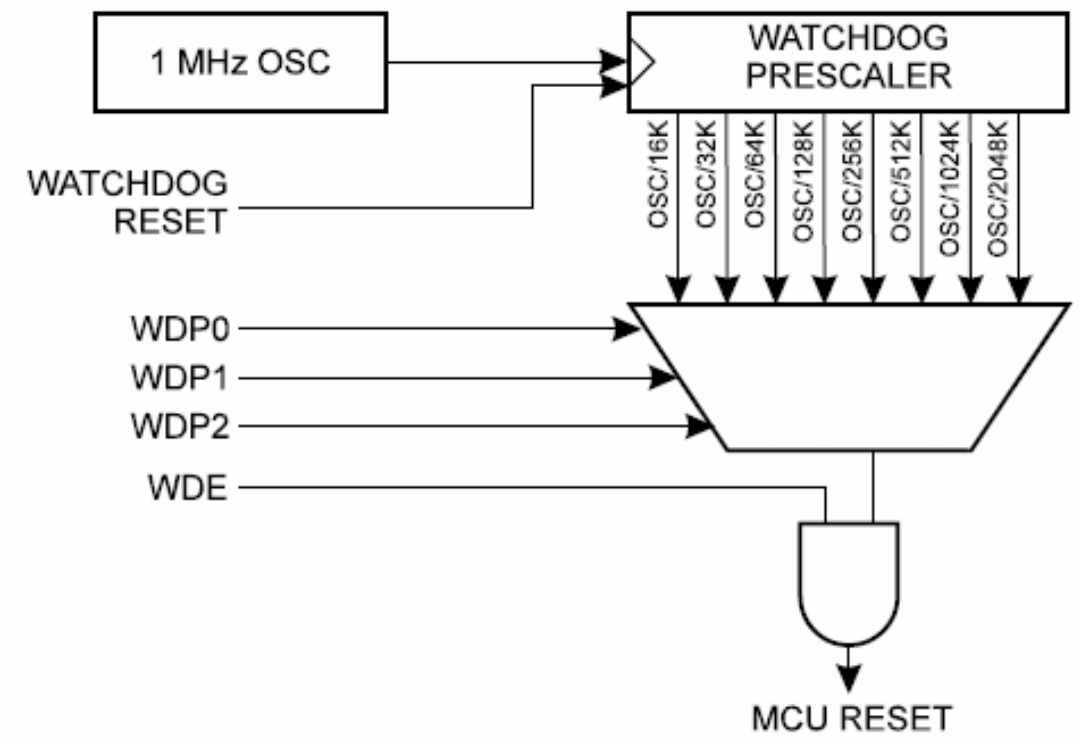
Γράφημα 15. Σχεδιάγραμμα κυκλώματος.

5.3 Υλοποίηση Υποσυστημάτων.

5.3.1 Διαγνωστικό ορθής λειτουργίας.

Για την εξασφάλιση της ορθής λειτουργίας του φρουρού ως ολοκληρωμένο σύστημα χρησιμοποιείται το εσωτερικό hardware watchdog που διαθέτει ο ATmega161. Η σχεδίαση χρησιμοποιεί το κύκλωμα αυτό ώστε να καλύπτεται η περίπτωση δυσλειτουργίας του ίδιου του φρουρού.

Το εσωτερικό hardware watchdog του ATmega161 έχει εξαιρετικά μεγάλη σταθερότητα στην λειτουργία διότι ο μηχανισμός του (σχήμα 16) είναι απλός άρα και ευσταθής. Επιπλέον ο ταλαντωτής αυτού του κυκλώματος είναι αυτόνομος και ξεχωριστός από την πηγή ταλάντωσης των υπόλοιπων συστημάτων του μικροελεγκτή. Ο ταλαντωτής είναι κατασκευασμένος εντός του ολοκληρωμένου του μικροελεγκτή εμφανίζοντας μεγάλη ευστάθεια στα φυσικά χαρακτηριστικά του.



Γράφημα 16. Λειτουργία Hardware Watchdog Timer.

Ακόμη η αξιοπιστία της λειτουργίας του hardware watchdog ενισχύεται και από το γεγονός πως το σύστημα προστατεύεται από κατά λάθος (ή τυχαία) απενεργοποίηση του watchdog με την ισχύ ενός ειδικού μηχανισμού απενεργοποίησης. Συγκεκριμένα η απενεργοποίηση του watchdog timer γίνεται εφόσον ακολουθηθεί μια συγκεκριμένη διαδικασία η οποία απαιτεί τον συγχρονισμό τριών διαφορετικών συμβάντων στο σύστημα, κάνοντας έτσι εξαιρετικά απίθανη την περίπτωση τυχαίας απενεργοποίησης του watchdog.

Εφόσον ο watchdog είναι ενεργοποιημένος, για να απενεργοποιηθεί θα πρέπει να συμβούν τα παρακάτω:

1. Στην ίδια εντολή (δηλαδή ταυτόχρονα) να ανατεθεί η τιμή 1 στα bit ελέγχου WDTOE (Watchdog Turn-Off Enable) και WDE (Watchdog Enable).
2. Εντός το πολύ τεσσάρων κύκλο, να ανατεθεί η τιμή 0 στο WDE.

Λειτουργία του εσωτερικού Watchdog.

Για να ενεργοποιηθεί ο εσωτερικός watchdog θα πρέπει να αποδοθεί η τιμή 1 στο Bit 3 WDE (Watchdog Enable) του καταχωρητή Watchdog Timer Control Register (WDTCR), ο οποίος ελέγχει την λειτουργία του watchdog.

Από τη στιγμή που ο watchdog ενεργοποιείται, θα πρέπει ο κώδικας που εκτελεί ο μικροελεγκτής να φροντίζει για την επαναφόρτωση του μετρητή χρησιμοποιώντας την εντολή WDR (Watchdog Reset). Διαφορετικά μόλις ο μετρητής φτάσει στο μηδέν θα προκληθεί reset του συστήματος. Τα bit 2..0 – WDP2 (Watchdog Timer Prescaler 2,1,0) επιλέγουν τον χρονισμό του Watchdog timer ώστε να ρυθμιστεί η μέγιστη περίοδος που θα πρέπει να γίνει μηδενισμός του watchdog.

Η εμφάνιση συμβάντος reset του μικροελεγκτή με πηγή το watchdog δηλώνεται από το σύστημα στον καταχωρητή MCU Status Register MCUSR. Συγκεκριμένα το bit 3- WDRF (Watchdog Reset Flag) παίρνει την τιμή 1.

Η υπορουτίνα reset του συστήματος πριν απ' όλα ελέγχει την τιμή του WDRF ώστε να διαγνώσει την πιθανή αποτυχία στην λειτουργία του συστήματος και να ειδοποιήσει τον χρήστη για την ύπαρξη του προβλήματος.

5.3.2 Ημερολόγιο Συμβάντων.

Για να γίνει η καλύτερη δυνατή εκμετάλλευση της μνήμης, η αποθήκευση των συμβάντων στο ημερολόγιο γίνεται με την χρήση χρονοσφραγίδων και σχετικών γεγονότων. Το ημερολόγιο ξεκινάει πάντα με μία χρονοσφραγίδα, την οποία ακολουθούν συμβάντα σχετικά ως προς αυτήν όσο η απόσταση των συμβάντων το επιτρέπουν. Όταν χρειαστεί να καταγραφεί ένα συμβάν το οποίο είναι πολύ μακριά από τη χρονοσφραγίδα, τότε κλείνει το σύνολο αυτό και εγγράφεται μια καινούργια χρονοσφραγίδα.

ΑΡΧΗ ΧΡΟΝΙΚΟΥ ΠΛΑΙΣΙΟΥ
ΧΡΟΝΟΣΦΡΑΓΙΔΑ
ΣΧΕΤΙΚΟ ΓΕΓΟΝΟΣ 1
ΣΧΕΤΙΚΟ ΓΕΓΟΝΟΣ 2
….
ΣΧΕΤΙΚΟ ΓΕΓΟΝΟΣ ν
ΤΕΛΟΣ ΧΡΟΝΙΚΟΥ ΠΛΑΙΣΙΟΥ
ΑΡΧΗ ΧΡΟΝΙΚΟΥ ΠΛΑΙΣΙΟΥ
ΧΡΟΝΟΣΦΡΑΓΙΔΑ
ΣΧΕΤΙΚΟ ΓΕΓΟΝΟΣ 1
ΣΧΕΤΙΚΟ ΓΕΓΟΝΟΣ 2
…

Σχήμα 17. Δομή Ημερολογίου.

Η μορφή των εγγραφών του ημερολογίου παρουσιάζεται στο σχήμα 18. Κάθε εγγραφή καταλαμβάνει 3 bytes στη μνήμη.

ΕΙΔΟΣ ΕΓΓΡΑΦΗΣ	ΤΙΜΗ	ΧΡΟΝΙΚΗ ΑΠΟΣΤΑΣΗ
5 bit	7 bit	12 bit

Σχήμα 18. Δομή Εγγραφής.

Τα είδη εγγραφής που χρησιμοποιήθηκαν στην παρούσα υλοποίηση είναι τα παρακάτω:

Πίνακας 14. Κωδικοί εγγραφών ημερολογίου.

Είδος Εγγραφής	Επεξήγηση
0	Εγγραφή ελέγχου ημερολογίου.
1	Φόρτος επεξεργαστή.
2	Χρόνος εξυπηρέτησης διακοπών.
3	Ελεύθερα Mbytes μνήμης.
4	Ελεύθερος χώρος δίσκου.
5	Σφάλμα δικτύου.
6-31	Διαθέσιμες για μελλοντικές υλοποιήσεις.

Πίνακας 15. Κατηγορίες εγγραφών ελέγχου ημερολογίου.

Τιμή	Επεξήγηση	Επεξηγήσεις
0	Αρχή χρονικού πλαισίου.	Ακολουθούν 6 byte με κωδικοποιημένη την ημερομηνία και την ώρα.
1	Τέλος χρονικού πλαισίου.	Πρέπει να ακολουθείται από αρχή χρονικού πλαισίου ή τέλος αρχείου (μνήμης).
2	Διακοπή επικοινωνίας.	Σημειώθηκε διακοπή επικοινωνίας με το εποπτευόμενο σύστημα.
3	Διακοπή συστήματος.	Σημειώθηκε επανεκκίνηση του εποπτευόμενου συστήματος λόγω ενεργοποίησης του φρουρού.

Περιγραφή Λειτουργίας Χρονοσφραγίδων.

Για την αποθήκευση της ημερομηνίας HH/MM/XXXX το σύστημα χρησιμοποιεί κωδικοποίηση που βασίζεται στην κατασκευή του αριθμού MMHHXXXX. Προφανώς ο παραπάνω αριθμός δεν μπορεί να ξεπεράσει την τιμή 12319999 η οποία μπορεί να αποθηκευθεί επαρκώς σε 24 bit.

Ομοίως για την αποθήκευση της ώρας με λεπτομέρεια δευτερολέπτου χρησιμοποιείται η μετατροπή της ώρας ΩΩ:ΛΛ:ΔΔ σε ακέραιο της μορφής ΩΩΛΛΔΔ.

Εν συνεχεία κάθε εγγραφή φέρει την χρονική απόσταση σε δευτερόλεπτα από την χρονοσφραγίδα.

5.3.3 Σύστημα λογισμικού.

Ορισμός Παραμέτρων.

Για τον ορισμό των παραμέτρων του συστήματος ο χρήστης διαμορφώνει τις τιμές του αρχείου ρυθμίσεων, του οποίου μια τυπική διαμόρφωση παρατίθεται αμέσως.

```
#Advanced PC Watchdog Configuration File
# (c) 2003 Technical University of Crete, Microcontroller & Hardware Laboratory
#

#Deamon enabled (1) / disabled (0)
WatchdogDeamonEnable=1

#Watchdog enabled / disabled
WatchdogEnable=1

#Maximum consecutive reboots [1..255]
WatchdogMaximumConsecutiveReboots=3
```

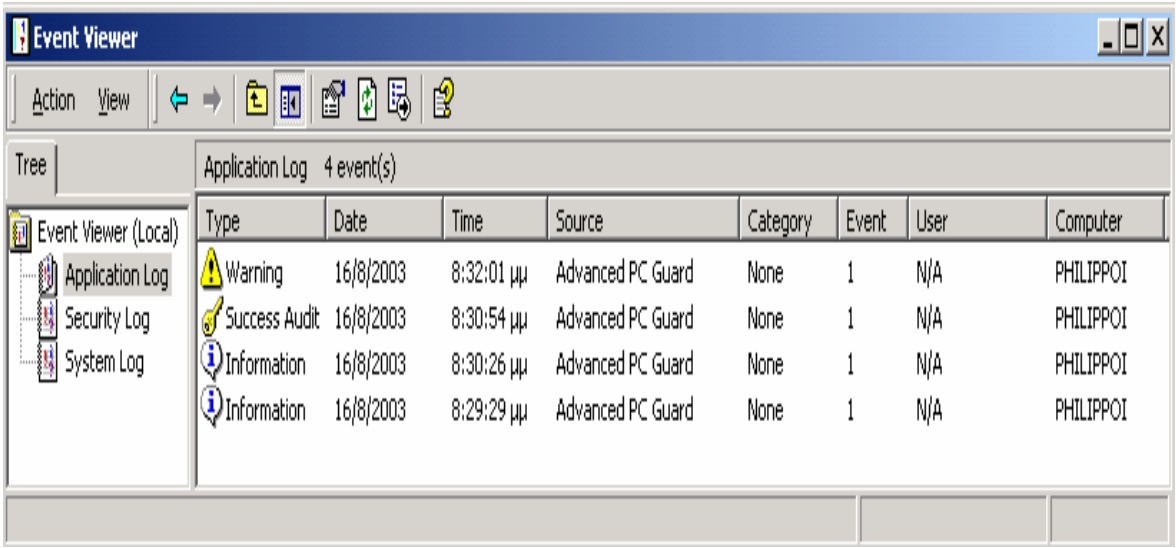
```
#Time in seconds between consecutive reboots [0..65536]
WatchdogConsecutiveSeconds=600

#Bootstrap delay in seconds [0..65536]
WatchdogBootstapDelay=300

#Period in between dog feeding in tenths of seconds [0..65536]
FeedMeInTenthsOfSeconds=100

#Notification String
ModemString=”ATA\nATH1\nATD2821037347;\nAT#VTS\nATH0\n”

#CPU Usage Threshold, Enable, Reboot, Notify
CPU=100,1,1,1

#Interrupt Time Usage Threshold, Enable, Reboot, Notify
INT=20,1,0,1

#Memory Free Mbytes Threshold, Enable, Reboot, Notify
MEM=40,1,0,1

#Boot disk space Mbytes Usage Threshold, Enable, Reboot, Notify
DISK=99,1,0,1

#Network Connection Delay Threshold, Enable, Reboot, Notify
NETDELAY=500,1,0,1

#Network connectivity Enabled, Reboot, Notify
NETAV=1,0,1

#Network TTL Status Enabled, Reboot, Notify
NETTTL=1,0,1
```

Ημερολόγιο Υπολογιστή.

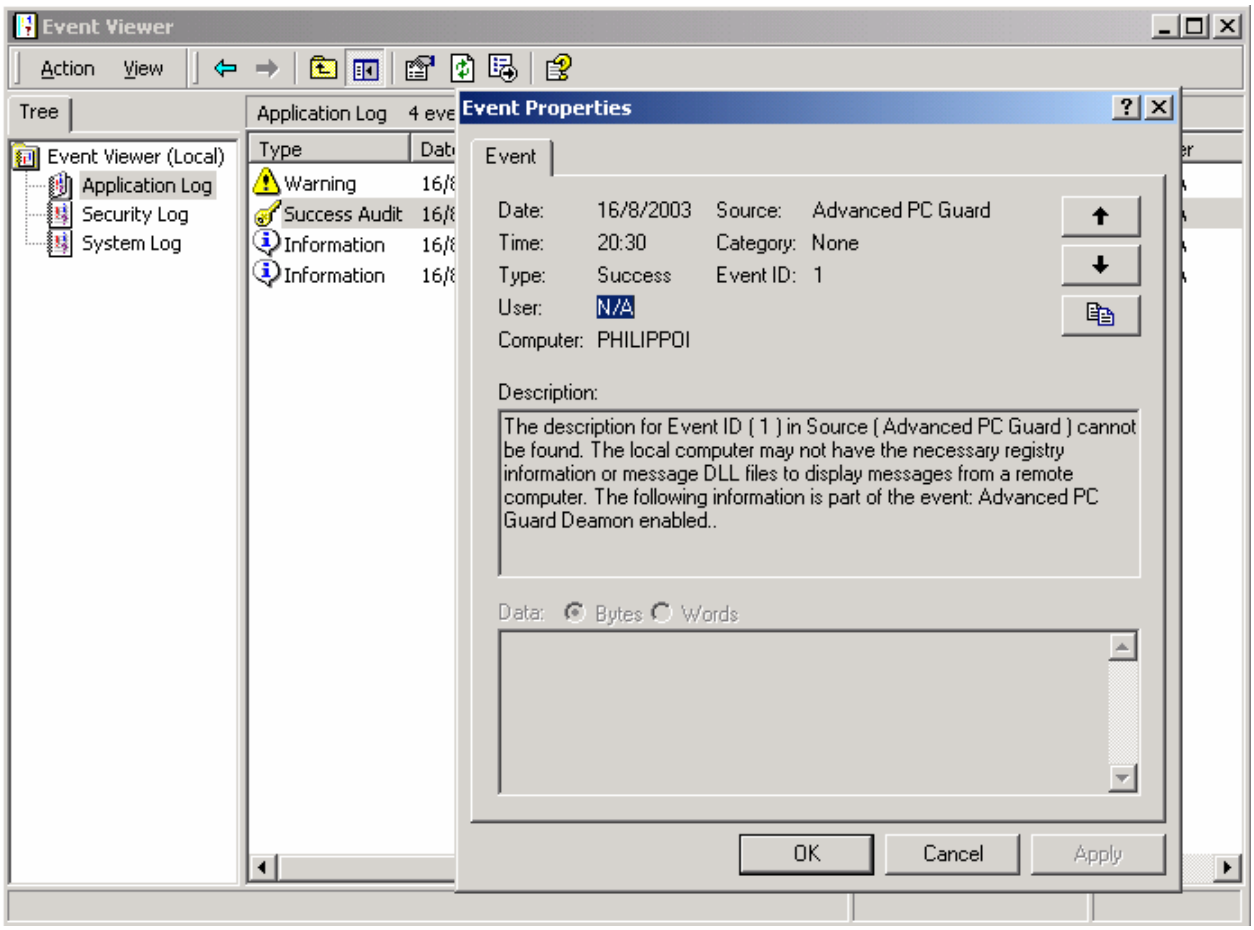
Τα αποτελέσματα της λειτουργίας του συστήματος εγγράφονται στο ημερολόγιο του συστήματος.

Windows.

Για το λειτουργικό σύστημα Windows, οι πράξεις του συστήματος καταχωρούνται στο ημερολόγιο εφαρμογών του λειτουργικού συστήματος (Application Event Viewer.). Για την εισαγωγή των εγγραφών χρησιμοποιείται η βιβλιοθήκη Win32:LogAPI.



Γράφημα 19. Υλοποίηση Event Log Messages στα Windows.



Γράφημα 20. Υλοποίηση event log εγγραφής στα Windows.

Linux.

Στο λειτουργικό σύστημα Linux, το ημερολόγιο υλοποιημένο ως ένα αρχείο κειμένου τοποθετείται στο directory /var/log.

6. Λειτουργία και Πειράματα.

6.1 Λειτουργικά Συστήματα και Crashes

6.1.1 Windows Crashes.

Οι λόγοι οι οποίοι θα οδηγήσουν το λειτουργικό σύστημα Windows σε απότομη διακοπή της λειτουργίας του περιγράφονται σύμφωνα με τα τεχνικά εγχειρίδια της Microsoft παρακάτω:

- Ένας οδηγός συσκευής ή μια συνάρτηση του λειτουργικού συστήματος που τρέχει σε kernel mode προκάλεσε μια εξαίρεση που το λειτουργικό δεν μπορεί να χειριστεί, όπως η παραβίαση πρόσβασης μνήμης.
- Μια κλήση σε συνάρτηση του πυρήνα προκάλεσε ενημέρωση του πίνακα χρονοπρογραμματισμού, και ενεργοποίησε ένα dispatcher object το οποίο δεν έχει λάβει signal, ενώ το IRQL (βλέπε §4.3.1) είναι στο DPC / dispatch επίπεδο.
- Εμφανίστηκε page fault σε μνήμη της οποίας το αντίγραφο είναι στο page file ενώ το επίπεδο διακοπών είναι στο DPC / dispatch επίπεδο.
- Ένας οδηγός συσκευής ή μια συνάρτηση του λειτουργικού συστήματος συνειδητά προκαλεί απότομη διακοπή λειτουργίας του συστήματος (καλώντας την συνάρτηση συστήματος KeBugCheckEx) επειδή ανίχνευσε συνθήκες που υποδηλώνουν πως η περαιτέρω λειτουργίας του συστήματος δεν μπορεί να συνεχιστεί χωρίς να υπάρχει κίνδυνος για καταστροφή δεδομένων.
- Εμφανίστηκε λάθος στο υλικό μέσω μια διακοπής που δεν επιδέχεται μάσκα (Non Maskable Interrupt).

Το λειτουργικό σύστημα Windows προσφέρει την δυνατότητα της εξαναγκασμένης απότομης διακοπής λειτουργίας με το πάτημα του πλήκτρου Control ταυτόχρονα μα το διπλό πάτημα του πλήκτρου Scroll Lock για λόγους πειραματισμού και ανάπτυξης.

Για να ενεργοποιηθεί η λειτουργικότητα αυτή θα πρέπει στο μητρώο (Windows Registry) να προστεθεί η εγγραφή με τύπο DWORD και περιγραφή CrashOnCtrlScroll στην θέση HKLM \SYSTEM \CurrentControlSet \Services \i8042prt \Parameters .

Όπως αναφέρθηκε και πιο πάνω επίσης απότομη διακοπή μπορεί να προκληθεί όταν μια διεργασία (η οποία όμως θα πρέπει να έχει εγκατασταθεί στο σύστημα ως οδηγός συσκευής στο επίπεδο του kernel και όχι στο επίπεδο του user) καλέσει την συνάρτηση KeBugCheckEx.

6.1.2 Linux Crashes.

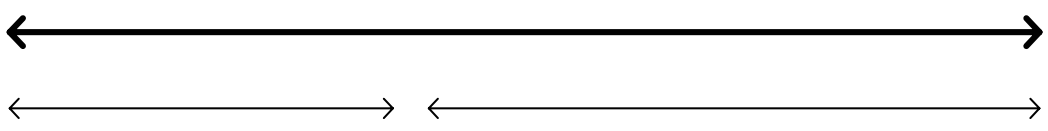
Η απότομη διακοπή στο Linux υλοποιείται από τη συνάρτηση του πυρήνα (/linux/kernel/panic.c). Η δήλωση της συνάρτησης είναι η void panic(const char * fmt).

Η κλήση της συνάρτησης panic οδηγεί σε απότομη διακοπή λειτουργίας του συστήματος.

6.2 Αύξηση της αξιοπιστίας.

Στο κεφάλαιο 2 της εργασίας παρουσιάστηκαν οι βασικές μαθηματικές εξισώσεις που περιγράφουν την αξιοπιστία των συστημάτων. Με σκοπό την μελέτη των επιδράσεων του δυναμικού συστήματος που υλοποιήθηκε καταστρώθηκε ο πίνακας 16 ο οποίος παρουσιάζει μια γενική εικόνα των απαιτήσεων της αξιοπιστίας σχετικά με την αδιάλειπτη (δηλαδή χωρίς διακοπή) λειτουργία.

Η σύνταξη του πίνακα γίνεται με βάση το σχήμα του γραφήματος 21.



Γράφημα 21. Πρακτικό σχήμα υπολογισμού χρόνου αδιάλειπτης λειτουργίας.

Έστω ένα σύστημα που ξεκινάει τη λειτουργία του με βλάβη, η οποία διαρκεί χρόνο x. Εφόσον το σύστημα θέλει να συμμορφωθεί προς δείκτη αξιοπιστίας k θα πρέπει να ισχύει η μαθηματική σχέση:

$$\frac{y}{x+y} = k \tag{6.1}$$

Ο πίνακας δείχνει την σχέση μεταξύ ημερών εκτός λειτουργίας και απαιτούμενης αδιάλειπτης λειτουργίας ώστε να ικανοποιείται uptime 99%, 99.9%, 99,99%, 99,999% αντιστοίχως.

Πίνακας 16. Υπολογισμός χρόνου αδιάλειπτης λειτουργίας σε συνάρτηση με χρόνο βλάβης για αντίστοιχα επίπεδα δείκτη αξιοπιστίας.

Χρόνος Βλάβης Σε Μέρες		**Χρόνος Αδιάλειπτης Λειτουργίας Σε Μέρες με αντίστοιχο Uptime**			
		99%	**99.9%**	**99.99%**	**99.999%**
1 sec	1,15741E-05	0,001145833	0,0115625	0,115729167	1,157395833
10 sec	0,000115741	0,011458333	0,115625	1,157291667	11,57395833
30 sec	0,000347222	0,034375	0,346875	3,471875	34,721875
1 min	0,000694444	0,06875	0,69375	6,94375	69,44375
5 min	0,003472222	0,34375	3,46875	34,71875	347,21875
30 min	0,020833333	2,0625	20,8125	208,3125	2083,3125
1 h	0,041666667	4,125	41,625	416,625	4166,625
2 h	0,083333333	8,25	83,25	833,25	8333,25
6 h	0,25	24,75	249,75	2499,75	24999,75
12 h	0,5	49,5	499,5	4999,5	49999,5
	1	99	999	9999	99999
	2	198	1998	19998	199998
	3	297	2997	29997	299997

Πίνακας 17. Υπολογισμός χρόνου αδιάλειπτης λειτουργίας σε συνάρτηση με χρόνο βλάβης για αντίστοιχα επίπεδα δείκτη αξιοπιστίας (2).

Χρόνος Βλάβης	Χρόνος Αδιάλειπτης Λειτουργίας με αντίστοιχο Uptime			
	99%	99.9%	99.99%	99.999%
1 sec	99 sec	17 min	2 h	28 h
10 sec	17 min	3 h	28 h	12 days
30 sec	49 min	8 h	4 days	1 month
1 min	99 min	16 h	7 days	2 months
5 min	8 h	4 days	35 days	11 months
30 min	2 days	20 days	7 months	6 years
1 h	4 days	41 days	13 months	11 years
2 h	8 days	84 days	27 months	23 years
6 h	24 days	9 months	7 years	70 years
12 h	49 days	1 years	13 years	1,3 cent
1 day	99 days	3 years	27 years	2,7 cent
2 days	7 months	6 years	55 years	5,5 cent
3 days	10 months	8 years	83 years	8,3 cent

Από τον παραπάνω πίνακα είναι προφανές πως η παρατεταμένη αποτυχία του συστήματος οδηγεί στην εκτίναξη του χρόνου συνεχούς και σταθερής λειτουργίας που θα ήταν αναγκαίος προς την αποκατάσταση της αξιοπιστίας του συστήματος. Η άμεση παρέμβαση του φρουρού για την επαναφορά του συστήματος, χωρίς την παρέμβαση του χρήστη έχει σαν αποτέλεσμα την αποφυγή του μακρού χρονικού διαστήματος σε κατάσταση βλάβης.

6.3 Διενέργεια Πειραμάτων.

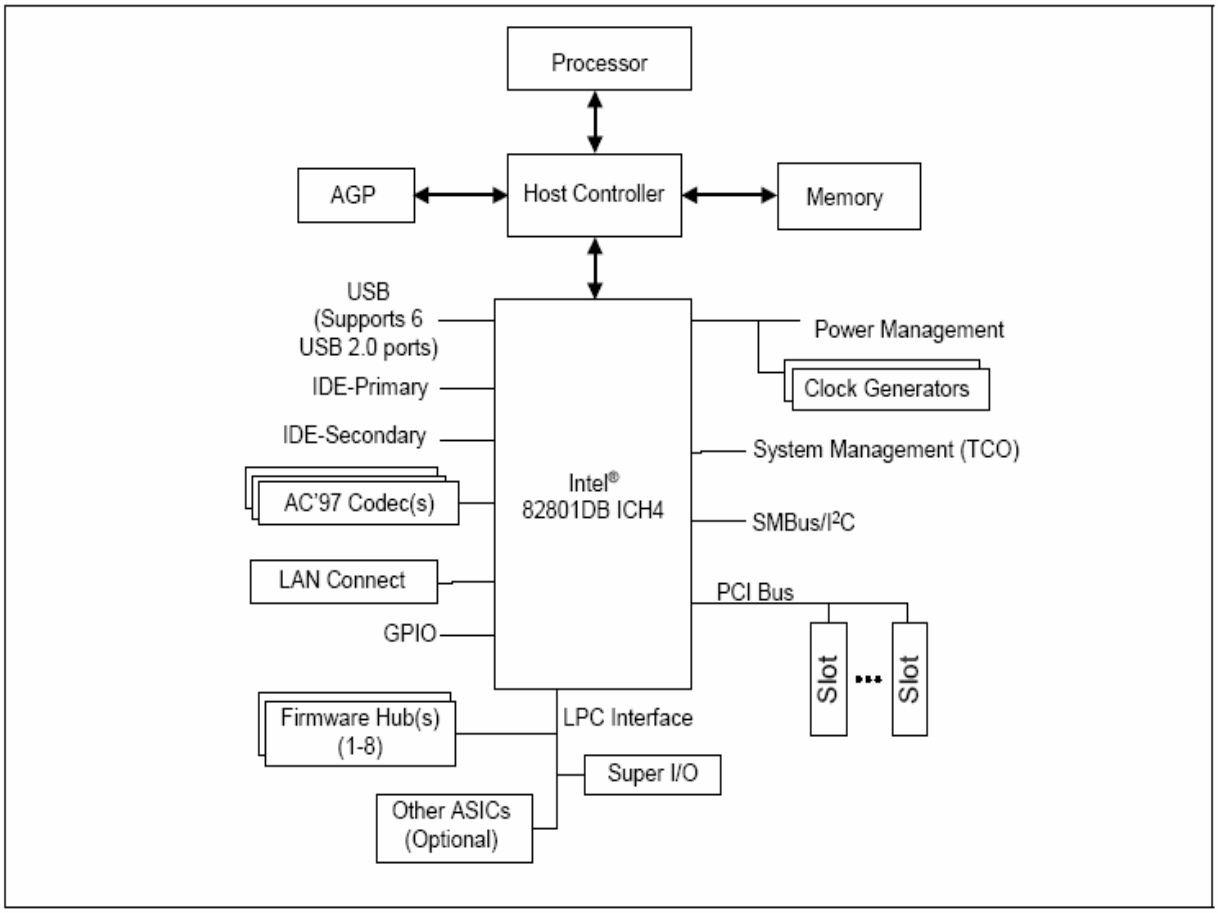
Για την διεξαγωγή των πειραμάτων χρησιμοποιήθηκε προσωπικός υπολογιστής με τα παρακάτω χαρακτηριστικά:

MotherBoard

Η μητρική πλακέτα του υπολογιστή ήταν της εταιρίας Gigabyte, μοντέλο GA-8IE η οποία χρησιμοποιεί το Intel 845E Chipset. Το συγκεκριμένο chipset αποτελείται από τον ελεγκτή μνήμης 82845 Memory Controller Hub και από τον ελεγκτή I/O 82801DB.

Το γράφημα 22 αφορά την λειτουργία του I845E προσανατολισμένου προς την μεριά του I/O controller hub. [27] Στο σχήμα διακρίνεται η σύνδεση του ICH4 μέσω σύνδεσης LPC (Low Pin Count) με το ολοκληρωμένο Super I/O ITE8702, το οποίο είναι υπεύθυνο για έλεγχο οδηγού μαλακού δίσκου, σειριακές πόρτες, ps2 ποντίκι, ps2 πληκτρολόγιο και παράλληλη σύνδεση (centronics).

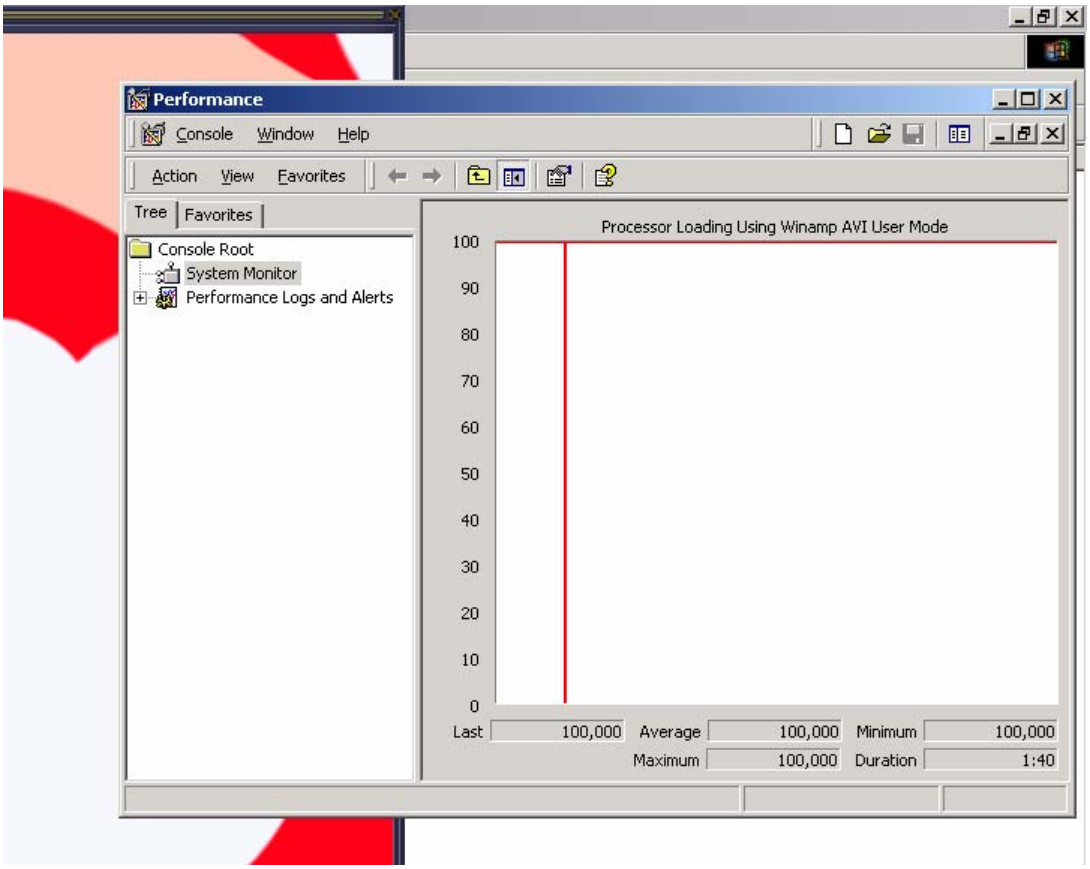
Ο επεξεργαστής του συστήματος ήταν Pentium 4 @ 2GHz, η μνήμη 512 Mbytes και ο δίσκος υποστήριζε το πρωτόκολλο ATA100.



Γράφημα 22. Διάγραμμα συνδέσεων E845 Intel Chipset.

6.3.1 Συμπεριφορά του Συστήματος με υπερφόρτωση του επεξεργαστή από διεργασία του χρήστη στα Windows.

Για την δοκιμή του συστήματος το πρώτο πείραμα που έγινε ήταν η εκτέλεση της εφαρμογής winamp με την επέκταση Advanced Visualization Studio σε περιβάλλον Windows. Η συγκεκριμένη εφαρμογή χρησιμοποιεί κοπιαστικά τον επεξεργαστή με σκοπό την ανασύνθεση εικόνων σε πραγματικό χρόνο.



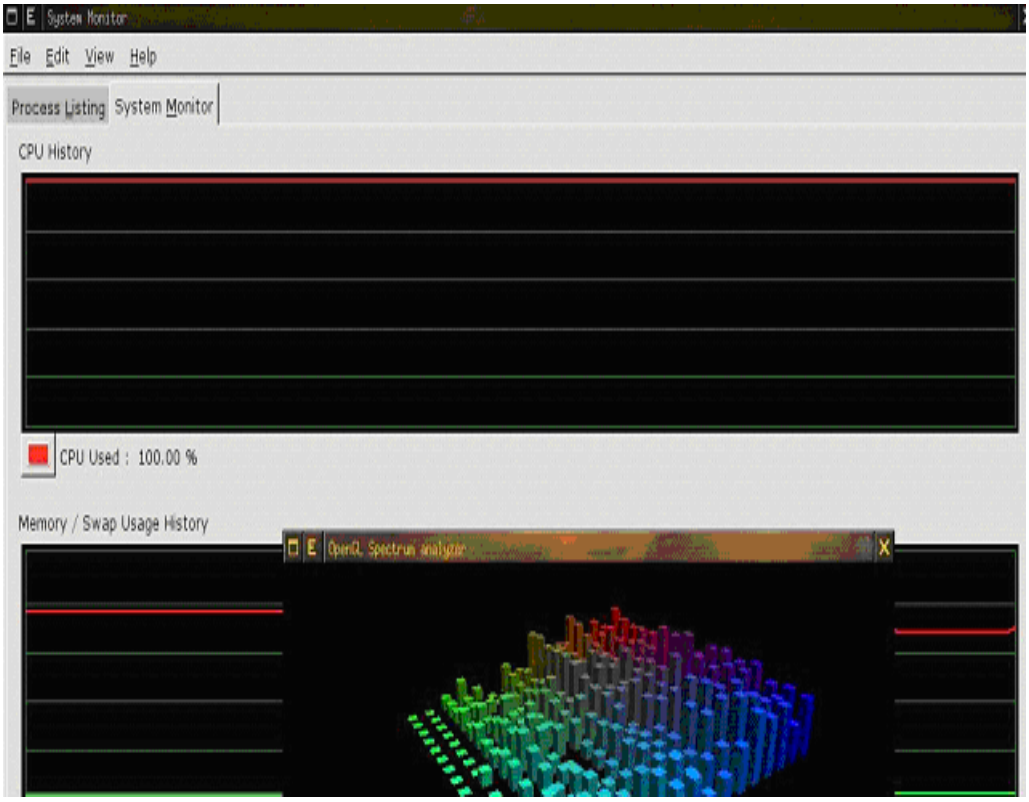
Γράφημα 23. Πείραμα κόρου επεξεργαστή στα Windows.

Η επικοινωνία μεταξύ φρουρού και προσωπικού υπολογιστή δεν εμφάνισε κανένα πρόβλημα είτε όταν η διεργασία του Winamp έτρεχε με προτεραιότητα normal είτε με προτεραιότητα realtime.

6.3.2 Συμπεριφορά του Συστήματος με υπερφόρτωση του επεξεργαστή από διεργασία του χρήστη στο Linux.

Το ίδιο πείραμα επαναλήφθηκε στο Linux. Η συμπεριφορά του συστήματος ήταν ευσταθέστατη χωρίς να παρατηρηθεί πρόβλημα στην επικοινωνία του δαίμονα με τον φρουρό.

Η συμπεριφορά αυτή εξηγείται από την προτεραιότητα που έχει η εξυπηρέτηση του I/O σε σχέση με τα επεξεργαστικά καθήκοντα υπολογισμών.



Γράφημα 24. Πείραμα κόρου επεξεργαστή στο Linux.

6.3.3 Συμπεριφορά του Συστήματος με υπερφόρτωση του συστήματος εικονικής μνήμης με σκοπό την αύξηση των διακοπών στα Windows.

Το δεύτερο πείραμα που εκτελέστηκε ήταν η εκτέλεση του παρακάτω κώδικα:

```
int main()
{
        char * flood;
         for (;;) {
                flood= (char    *)malloc(1000);
         }
        return 0; // Should never receive this exit code
}
```

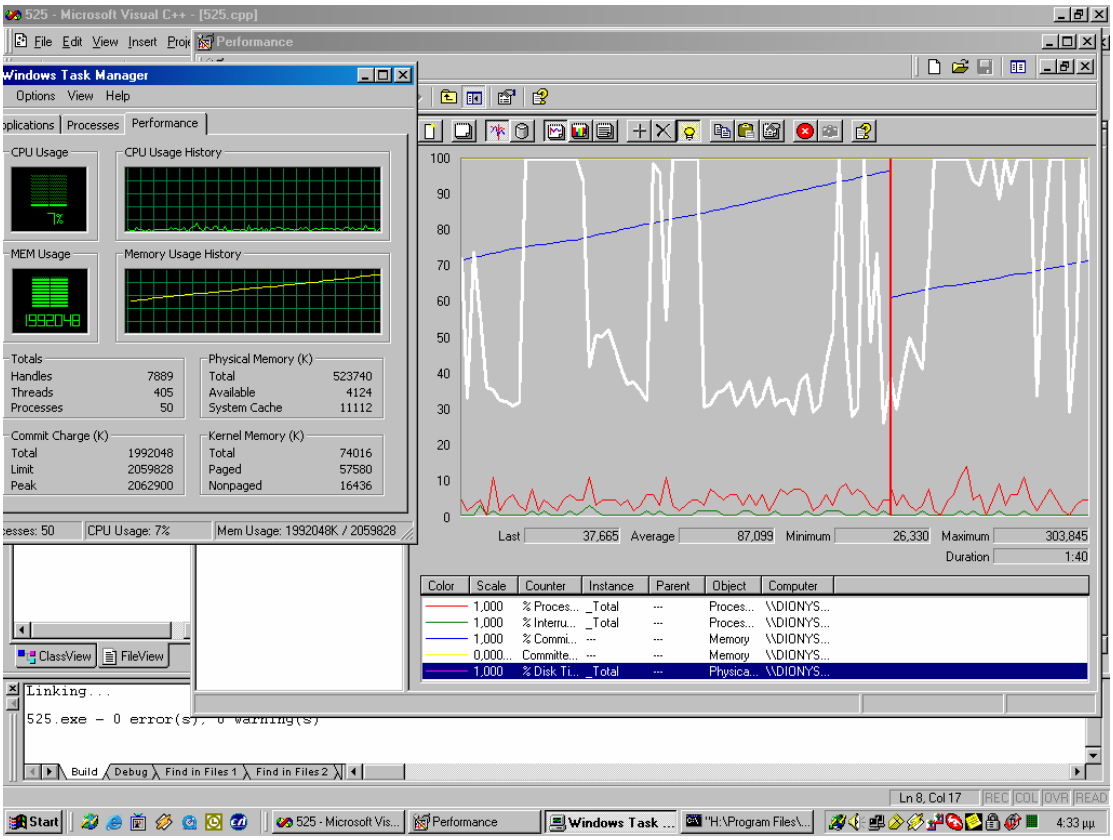
Γράφημα 25. Κώδικας δοκιμής διαχειριστή μνήμης.

Ο κώδικας έχει σκοπό την κατάληψη όλης της διαθέσιμης μνήμης του συστήματος, με σκοπό την υπερφόρτωση του διαχειριστή εικονικής μνήμης που θα έχει ως αποτέλεσμα την διερεύνηση της συμπεριφοράς του συστήματος α) σε σχέση με τις διεργασίες του πυρήνα όπως είναι ο memory manager και β) σε σχέση με τον αυξημένο φόρτο εξυπηρέτησης διακοπών που συνδέονται άμεσα με το κανάλι επικοινωνίας προσωπικού υπολογιστή και φρουρού.

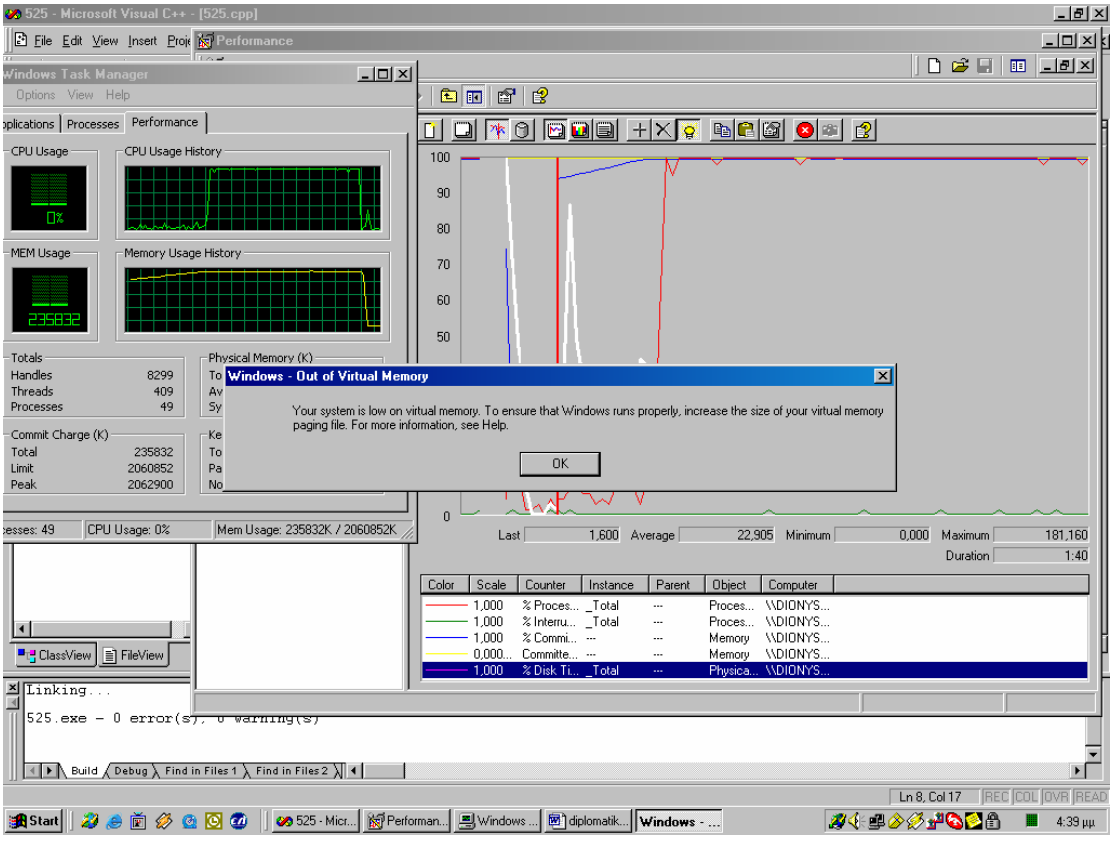
Το σύστημα του φρουρού πέρασε με απόλυτη επιτυχία και αυτό το πείραμα. Ο φρουρός δεν έχασε την επικοινωνία με το σύστημα.

Η συμπεριφορά του λειτουργικού συστήματος Windows περιγράφεται στο γράφημα 26. Με κόκκινο χρώμα φαίνεται ο φόρτος του επεξεργαστή (% CPU time), με πράσινο χρώμα ο φόρτος των διακοπών, με μπλε χρώμα η δεσμευμένη μνήμη (% memory used) και με λευκό χρώμα ο φόρτος του δίσκου (% disk usage).

Παρατηρήθηκε πως η εξαντλητική δέσμευση μνήμης δεν οδηγεί σε αστάθεια το σύστημα, αφού ο φόρτος του επεξεργαστή παραμένει κοντά στο δέκα της εκατό. Ο δίσκος, ο οποίος λειτουργεί μεταξύ του 50% και του 100% του χρόνου του προσπαθεί να ακολουθήσει σε δέσμευση την προφανώς γρηγορότερη μνήμη.



Γράφημα 26. Πείραμα υπερφόρτωσης διαχειριστή μνήμης στα Windows.



Γράφημα 27. Πείραμα υπερφόρτωσης διαχειριστή μνήμης στα Windows (2)

Το γεγονός πως ο δίσκος δεν οδηγείται συνέχεια στο 100% πρέπει να έχει σχέση με τον τρόπο που λειτουργεί ο δρομολογητής του δίσκου. Πιθανώς το σύστημα να αφήνει κενά ώστε να εξυπηρετηθούν και αιτήσεις δεδομένων και όχι page file. Πάντως, η καλή υγεία του επεξεργαστή εγγυάται την επικοινωνία με τον φρουρό.

Όταν γεμίσει ακόμη και το page file, η συμπεριφορά των Windows απέναντι στον κώδικα της δοκιμής, όπως φαίνεται και στο σχήμα 27 είναι ουδέτερη. Το λειτουργικό παύει να δίνει μνήμη στην διεργασία αφού αυτό είναι αδύνατο να συμβεί. Ταυτόχρονα εμφανίζεται το αντίστοιχο μήνυμα προς τον χρήστη σχετικά με την εξάντληση της εικονικής μνήμης.

Το πρόγραμμα εν τω μεταξύ επειδή δεν φροντίζει να κάνει έλεγχο λαθών εκφυλίζεται σε μια αέναη αποτυχημένη κλήση της συνάρτησης, γεγονός που οδηγεί τον φόρτο του επεξεργαστή να εκτιναχθεί στο 100%.

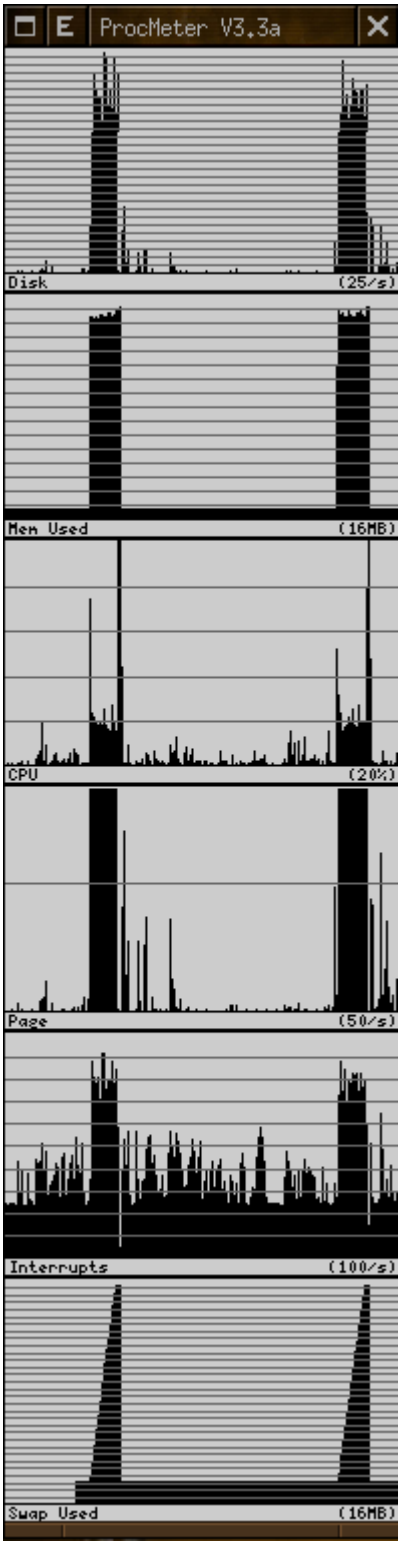
Ο τερματισμός του προγράμματος που φαίνεται στο τέλος του γραφήματος 27 οφείλεται σε παρέμβαση από τον χρήστη και όχι σε παρέμβαση από το λειτουργικό.

6.3.4 Συμπεριφορά του Συστήματος με υπερφόρτωση του συστήματος εικονικής μνήμης με σκοπό την αύξηση των διακοπών στο Linux.

Ο φρουρός πέτυχε στην δοκιμασία της υπερφόρτωσης του συστήματος εικονικής μνήμης και στο λειτουργικό σύστημα Linux χωρίς το σύστημα να οδηγηθεί σε λάθος συμπέρασμα για την ευστάθεια του προσωπικού υπολογιστή.

Η συμπεριφορά του κώδικα του σχήματος 25 στο λειτουργικό σύστημα Linux ήταν παρόμοια με αυτή που παρατηρήθηκε στο περιβάλλον των Windows. Όπως φαίνεται και στο σχήμα 28 η έναρξη της εκτέλεσης του προγράμματος εξελίχθηκε φυσιολογικά. Παρατηρείται ωστόσο μια μεγαλύτερη επιβάρυνση του επεξεργαστή για την εξυπηρέτηση των κλήσεων δέσμευσης μνήμης (γύρω στο 20% σε σχέση με το 10% στα windows). Και πάλι όμως η ευστάθεια του συστήματος δεν απειλείται στο παραμικρό ώστε να παρεμποδιστεί η έγκαιρη απάντηση στις αιτήσεις του φρουρού.

Μάλιστα το Linux προχωράει ένα βήμα περισσότερο στο θέμα της ασφάλειας, αφού η επαναλαμβανόμενη αποτυχημένη κλήση της malloc οδηγεί σε αυτόματο τερματισμό της διεργασίας του πειράματος. Έτσι ο επεξεργαστής φτάνει στο 100% του φόρτου για ελάχιστο χρονικό διάστημα ώστε να μην μπαίνει το σύστημα καν σε συνθήκες που θα μπορούσαν να προκαλέσουν εσφαλμένη απόφαση.



Γράφημα 28. Πείραμα υπερφόρτωσης διαχειριστή μνήμης στο Linux.

7. Συμπεράσματα και Επεκτάσεις.

7.1 Συμπεράσματα

Στα πλαίσια της παρούσας εργασίας υλοποιήθηκε ένα σύστημα υλικού – λογισμικού βασισμένο σε μικροελεγκτή με σκοπό την αύξηση της αξιοπιστίας του προσωπικού υπολογιστή.

Ο σχεδιασμός του συστήματος έγινε με γνώμονα την επίτευξη σταθερότητας και υψηλής αξιοπιστίας ώστε να μπορεί να υπάρχει το αναγκαίο προπέτασμα για την διαφύλαξη της αξιοπιστίας του υπό εποπτεία συστήματος. Γι' αυτό χρησιμοποιήθηκαν συμπαγείς και δοκιμασμένοι αλγόριθμοι για το λογισμικό ώστε να αποφευχθούν τυχόν κρυμμένα σφάλματα στον κώδικα.

Επιπλέον η αξιοπιστία του συστήματος προστατεύεται από πολλές ασφαλιστικές δικλίδες:

1. Διαγνωστικό σύστημα λειτουργίας μικροελεγκτή που υλοποιεί τον φρουρό.
2. Τοποθέτηση της διεργασίας του δαίμονα στην μικρότερη προτεραιότητα του λειτουργικού ώστε να ελαχιστοποιηθούν οι πιθανότητες εσφαλμένης διάγνωσης ορθής λειτουργίας του συστήματος.
3. Υποστήριξη της βασικής διαγνωστικής λειτουργίας με συμπληρωματικά διαγνωστικά στοιχεία.

Το υλοποιημένο σύστημα θα βοηθήσει στην αύξηση της αξιοπιστίας ενός υπολογιστικού συστήματος που βασίζεται σε προσωπικό υπολογιστή **άμεσα** φροντίζοντας για την επανεκκίνηση του προσωπικού υπολογιστή σε περίπτωση εμφάνισης σφάλματος.

Επιπλέον όμως ο φρουρός συνεισφέρει στην αξιοπιστία του υπολογιστικού συστήματος **έμμεσα** αφενός εκμηδενίζοντας τον κενό χρόνο έως ότου να γίνει χειροκίνητα αντιληπτή η αποτυχία του συστήματος αφού έχει την δυνατότητα ειδοποίησης με αυτόνομο από τον προσωπικό υπολογιστή τρόπο. Αφετέρου το ημερολόγιο συμβάντων του φρουρού μπορεί να προσφέρει πολύτιμα διαγνωστικά

στοιχεία τα οποία μπορούν να βοηθήσουν τον διαχειριστή να εντοπίσει τους λόγους της αστοχίας του συστήματος ή να τον αποτρέψουν από λάθη κατά την διαδικασία επισκευής του συστήματος.

7.2 Μελλοντικές Επεκτάσεις.

Οι μελλοντικές επεκτάσεις που προτείνονται για το σύστημα είναι οι παρακάτω:

Έλεγχος των οδηγών συσκευών επικοινωνίας.

Θα μπορούσε να μελετηθεί η δυνατότητα ελέγχου από τον δαίμονα του προσωπικού υπολογιστή του οδηγού της συσκευής επικοινωνίας (com port) ώστε να γίνεται απομόνωση του καναλιού επικοινωνίας σε επίπεδο οδηγού ώστε να αποφεύγονται οι παρεμβολές στο κανάλι λόγω σφαλμάτων από άλλες πράξεις εισόδου – εξόδου που πραγματοποιεί ο προσωπικός υπολογιστής.

Δυναμική Προσαρμογή της Περιόδου Ελέγχου με βάση την Πρόβλεψη Κόρου στον Επεξεργαστή.

Θα μπορούσε να μελετηθεί η δυνατότητα ειδοποίησης του φρουρού πως υπάρχει πρόβλεψη για υπερφόρτωση του επεξεργαστή στην άμεση χρονική περίοδο, ώστε ο φρουρός να είναι περισσότερο ελαστικός. Έτσι θα αποφεύγονται εσφαλμένες επανεκκινήσεις του συστήματος λόγω υπερφορτώσεις του επεξεργαστή, που δεν αποτελούν όμως δυσλειτουργία του συστήματος.

Ευρετήριο Γραφημάτων.

Ευρετήριο Πινάκων.

Βιβλιογραφικό Υλικό

Παγκόσμιος Ιστός.

[1] Ron Coates, *Study: E-mail woes worse than divorce,* http://zdnet.com.com/2100-1104-1013611.html, CNET Networks, June 5, 2003

[2] Daniel A. Begun, *The PC of the future, part one,* http://reviews.cnet.com/4520-6603_7-5021128-1.html?legacy=cnet, CNET Networks, March 6, 2002

[8] Tim Berners-Lee, *The Semantic Web,* http://www.sciam.com/article.cfm?articleID=00048144-10D2-1C70-84A9809EC588EF21, Scientific American, May 2001

Αναφορές.

[3] W.K. Ehrlich et al, *How Faults Cause Software Failures: Implications for Software Reliability Engineering,* Software Reliability Engineering, 1991. Proceedings., 1991 International Symposium on , 17-18 May 1991, p233-241

[4] Wilson D. Yates III et al., *Reliability Engineering As Applied to Software,* Reliability and Maintainability Symposium, 1990. Proceedings, Annual , 23-25 Jan. 1990, p425-429

[7] Hayley Iben, Ali Lakhia, Rachel Rubin, *Watchdog Designs for TinyOS Motes,* Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, May 16, 2002

[9] Roman Kochan et al., *Improved Watchdog Timer for Control the IBM PC Based Autonomous Computer Systems,* Modern Problems of Radio Engineering, Telecommunications and Computer Science, 2002. Proceedings of the International Conference , 18-23 Feb. 2002, p181-182

[10] Williams, T., *The care and correction of microprocessors* , EMC in High Integrity Digital Systems, IEE Colloquium on , 17 May 1991, p61-65

[11] Christof Fetzer, *Enforcing Perfect Failure Detection,* Distributed Computing Systems, 2001. 21st International Conference on. , 16-19 April 2001, p350-357

[12] Hassanein Amer et al., *Increasing the Reliability of the Motorola MC68HC11 in the Presence of Temporary Failures*, IEEE Melecon, 2002, p231-234

Βιβλιογραφία.

[5] Hennessy John & Patterson David, *Computer Architecture, A Quantitative Approach*

[6] Bunis Carl, *Design for reliability,* CRC Press LLC, 2000

[13] G.W.A. Dummer et al., *An Elementary Guide to Reliability,* Pergamon Press, 1977

[14] Ramakumar R, *Reliability Engineering,* The Electrical Engineering Handbook, CRC Press LLC, 2000

[165 Guy, C.G, *Computer Reliability,* The Electrical Engineering Handbook, CRC Press LLC, 2000

[16] Solomon Russinovich, *Inside Windows 2000, Third Edition,* Microsoft Press, 2000

[17] Brain Reeves, *Win32 System Services,* Microsoft Technologies Services, 1999

[18] Rubini, *Linux Device Drivers,* O' Reilly, 2002

[19] Bovet, *Understanding the Linux Kernel,* O' Reilly, 2000

[20] ATmega161, 8-bit AVR Microcontroller Data Book

[21] DARPA Internet Program, *RFC 792, Internet Control Message Protocol*, September 1981

[22] E. E. Lewis, *Introduction to Reliability Engineering*, John Wiley & Sons, 1996

[23] Patrick O' Connor, *Practical Reliability Engineering*, John Wiley & Sons, 1996

[24] Ι.Μ.Κοντολέοντος, *Ανάλυση και σχεδίαση αξιοπιστίας συστημάτων*, Εκδόσεις Αϊβαζή, 1996

[25] *Requirements for Hardware Watchdog Timers Supported by Microsoft Windows,* Microsoft™ Corporation, 2003

[26] Cisco Systems, RFC 3164, The BSD syslog Protocol

[27] Intel, *Intel® 82801DB I/O ControllerHub 4 (ICH4),Datasheet*, May 2002