ΤΕΧΝΟΛΟΓΙΚΟ ΕΚΠΑΙΔΕΥΤΙΚΟ ΙΔΡΥΜΑ
ΜΕΣΟΛΟΓΓΙΟΥ

ΤΜΗΜΑ ΤΗΛΕΠΙΚΟΙΝΩΝΙΑΚΩΝ ΣΥΣΤΗΜΑΤΩΝ &
ΔΙΚΤΥΩΝ

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΠΡΟΓΡΑΜΜΑ ΠΡΟΣΟΜΟΙΩΣΗΣ

" HARD REAL TIME " ΣΥΣΤΗΜΑΤΩΝ

ΟΝΟΜΑΤΕΠΩΝΥΜΟ:ΕΛΕΟΝΟΡΑ ΚΡΟΜΕΝΤΟΠΟΥΛΟΥ

ΕΠΙΒΛΕΠΩΝ: ΜΑΡΙΑΤΟΣ ΕΥΑΓΓΕΛΙΝΟΣ, Επιστημονικός συνεργάτης

ΝΑΥΠΑΚΤΟΣ 2010

Εγκρίθηκε από την τριμελή εξεταστική επιτροπή

Ναύπακτος, Ημερομηνία .....

ΕΠΙΤΡΟΠΗ ΑΞΙΟΛΟΓΗΣΗΣ

1. Ευαγγελινός Μαριάτος
2. .....
3. .....

ΠΙΝΑΚΑΣ ΠΕΡΙΕΧΟΜΕΝΩΝ

Ευχαριστίες

Τελειώνοντας την παρούσα διπλωματική εργασία θα ήθελα να απευθύνω τις ευχαριστίες μου στα άτομα που με την βοήθειά τους έκαναν δυνατή την εκπόνησή της.

Κατ' αρχάς, θα ήθελα να ευχαριστήσω θερμά τον υπεύθυνο καθηγητή της πτυχιακής εργασίας μου κο. Ευαγγελινό Μαριάτο για την επίβλεψη της εργασίας, τις συμβουλές του και για τη δυνατότητα που μου παρείχε ν' ασχοληθώ με ένα ενδιαφέρον θέμα και να διεβρύνω τον κύκλο των γνώσεών μου.

Επίσης, θα ήθελα να ευχαριστήσω την μητέρα μου που χωρίς την ψυχική και υλική βοήθειά της και την αμέριστη συμπαράστασή της δεν θα ήταν δυνατό να ολοκληρώσω τις προπτυχιακές μου σπουδές.

Όπως επίσης τον φίλο και συνεργάτη Θωμά που συνέβαλε τα μέγιστα στην εκπόνηση αυτής της πτυχιακής εργασίας.

Αφιερωμένο στην γιαγιάκα μου.

ΠΕΡΙΛΗΨΗ ΠΤΥΧΙΑΚΗΣ

Με την προσομοίωση μπορούν να αποφευχθούν πολλά λάθη που θα μπορούσαν να δημιουργηθούν στην εφαρμογή ενός πολύπλοκου συστήματος .Όπως είναι τα hard real-time συστήματα. Στην πτυχιακή αυτή το κύριο θέμα είναι να δημιουργηθεί ένα λογισμικό το οποίο θα προσομοιώνει ένα σύστημα που θα αποτελείτε από διαφορές διεργασίες οι οποίες εκτελούνται μετά την είσοδο δεδομένων και θα έχει σαν αποτέλεσμα την δημιουργία ενός διαγράμματος χρόνων-διεργασιών της προσομοίωσης .

Οι διεργασίες έχουν η καθεμία την δική της προτεραιότητα την οποία δηλώνει ο χρήστης. Έτσι όταν εκτελείτε μια διεργασία και η τιμή του αισθητήρα δώσει το σήμα ότι πρέπει να εκτελεστεί μια άλλη διεργασία μέσα σε ένα εύλογο χρονικό διάστημα θα πρέπει να ξεκινήσει η άλλη διεργασία. Το λογισμικό μας το οποίο κληθήκαμε να δημιουργήσουμε δίνει σε μία εικόνα τους χρόνους και την σειρά εκτέλεσης αυτών των διεργασιών.

Στην συγκεκριμένη πτυχιακή οι τιμές δίδονται από τον χρήστη μέσω web εφαρμογής. Δηλαδή δίνονται ηλεκτρονικά και καταχωρούνται σε μια βάση δεδομένων. Στην συνέχεια οι τιμές αποθηκεύονται σε csv αρχεία τα οποία διαβάζει ο κώδικας της C++. Κατόπιν αφού εκτελεστεί ο κώδικας εμφανίζει τα αποτελέσματα της αλληλεπίδρασης των διεργασιών ως μια εικόνα bmp (Bitmap).

SUMMARY OF DIPLOMA ESSAY

With simulation a lot of errors can be avoided, that could be created in the application of complicated systems, as are the hard real-time systems. In this Diploma Essay the main subject is to create a software which simulates a system that will constitute from different tasks that are executed afterwards the entry of data and will have as result the creation of tasks time diagram .

Each task has its own priority declared by the user. Thus when a task is executing and the value of a sensor triggers the execution of another task in a legitimate time interval, then the task must be executed. Our software gives us in a picture the time values and order of execution of tasks.

In this essay the values are given by the user via web application. That is to say they are given in a digital manner and registered in a database. Then the values are stored in csv files which the code of C++, reads. Then after the code is executed it presents the results of interaction of tasks as a picture bmp (Bitmap).

ΣΥΝΤΟΜΗ ΠΕΡΙΓΡΑΦΗ ΤΗΣ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Σύμφωνα με την έρευνα που έχει γίνει σε σχέση με την προσομοίωση των "hard real time" συστημάτων, χρησιμοποιήθηκαν ως θεωρητική βάση για την εκπόνηση της πτυχιακής εργασίας οι ακόλουθες εργασίες, οι οποίες περιγράφουν την προσομοίωση "hard real time" σε διάφορες περιπτώσεις:

I. Simulating Hardware, Software and Electromechanical Parts Using Communicating Simulators.

NIKOS C. PETRELLIS ,ALEXIS N. BIRBAS, MICHAEL K. BIRBAS, EVANGELINOS P. MARIATOS AND GEORGE D. PAPADOPOULOS

University of Patras, Electrical Engineering and Computer Technology Department, Applied Electronics (Laboratory, Rio Campus 26500, Patras Greece-Received November 25, 1996; Revised October 2, 1997)

II. A Simple Multi-Tasking Simulator

Adriaan de Beer1 Colin Fidge2

1School of Information Technology & Electrical Engineering

2Software Verification Research Centre

The University of Queensland, Australia

III. STRESS Simulator for Hard Real-time Systems

N. C. Audsley, A. Burns, M. F Richardson and A. J. Wellings Real-Time Systems Research Group, Department of Computer Science, University of York, Heslington, York, YO1 5DD, UK (email: Andy.WellingsKminster.york.ac.uk)

ΚΕΦΑΛΑΙΟ 1

1.1 ΕΙΣΑΓΩΓΗ

Όλο και περισσότερο, τα συστήματα ηλεκτρονικών υπολογιστών χρησιμοποιούνται στις εφαρμογές όπου μια αποτυχία του συστήματος μπορεί να οδηγήσει στη απώλεια ζωής ή μια καταστροφική ζημιά. Συνεπώς, η ανάγκη έχει προκύψει για μια θεωρητική βάση και μια πρακτική μεθοδολογία από τις οποίες το σωστό σχέδιο και η κατασκευή τέτοιων συστημάτων μπορούν να επιτευχθούν.

Αυτά τα συστήματα αναφέρονται συχνά ως "hard-real time" συστήματα, όπου το "real-time" απεικονίζουν το γεγονός ότι πρέπει άμεσα να αλληλεπιδράσουν με ένα μεταβαλλόμενο φυσικό περιβάλλον και τα "hard" συστήματα αναφέρονται στο γεγονός ότι τουλάχιστον μερικές λειτουργίες συστημάτων πρέπει να εκτελεσθούν μέσα στους συγκεκριμένους περιορισμούς χρονισμού.

1.2 Η ΥΠΟΔΟΜΗ

Τα hard-real time συστήματα αντιμετωπίζονται γενικά συμπεριλαμβάνοντας ένα σύνολο εργασιών που όταν προκαλούνται ή «ελευθερώνονται» πρέπει να ολοκληρωθεί η εκτέλεσή τους πριν από μια καθορισμένη προθεσμία. Οι εργασίες αυτές μπορούν να «απελευθερωθούν» με έναν από τρεις γενικούς τρόπους. Οι περιοδικές εργασίες «απελευθερώνονται» σε τακτά χρονικά διαστήματα. Οι σποραδικές εργασίες «απελευθερώνονται» σε μεταβλητά διαστήματα, αν και υπάρχει ένας ελάχιστος χρόνος μεταξύ των διαδοχικών «απελευθερώσεων» της διεργασίας. Οι μη περιοδικές εργασίες μπορούν να «απελευθερωθούν» οποιαδήποτε στιγμή, χωρίς τον ελάχιστο χρόνο μεταξύ των διαδοχικών απελευθερώσεων.

ΚΕΦΑΛΑΙΟ 2

2.1 ΠΡΟΣΟΜΟΙΩΝΟΝΤΑΣ ΥΛΙΚΟ,ΛΟΓΙΣΜΙΚΟ ΚΑΙ ΗΛΕΚΤΡΟΜΗΧΑΝΙΚΑ ΜΕΡΗ ΜΕ ΤΗΝ ΧΡΗΣΗ ΕΠΙΚΟΙΝΩΝΟΥΝΤΩΝ ΠΡΟΣΟΜΟΙΩΤΩΝ

Μία εργασία στην οποία βασίστηκε αυτή η πτυχιακή είναι ο σχεδιασμός κυκλωμάτων ενσωματωμένου επεξεργαστή για τον έλεγχο ηλεκτρομηχανικών συστημάτων σε ένα ενοποιημένο περιβάλλον , παρέχει πολλά πλεονεκτήματα, όπως μειωμένο σχεδιασμό και χρόνο εκσφαλμάτωσης και κατά συνέπεια μικρότερο χρόνο για να βγεί στην αγορά.

Σε αυτή την εργασία παρουσιάζεται μια περίληψη μιας μεθόδου σχεδιασμού η όποια επικεντρώνεται στην ταυτόχρονη προσομοίωση διαφόρων ετερογενών μερών.

Ένας πολύ σημαντικός τομέας βιομηχανικών εφαρμογών είναι ο έλεγχος διαφόρων ηλεκτρομηχανικών συστημάτων , από καταναλωτικά προϊόντα μέχρι διαστημικά σκάφη. Τα συστήματα αυτά συνήθως αποτελούντα από διάφορα μέρη, όπως τα ψηφιακά μέρη, το υλικολογισμικό του επεξεργαστή και τα ηλεκτρομηχανικά μέρη που ελέγχονται από τα ψηφιακά.

Η συνήθης πρακτική μέχρι πρόσφατα είναι η ξεχωριστός σχεδιασμός των διαφορετικών μερών , από αντίστοιχες σχεδιαστικές ομάδες μηχανικών.

Εξαιτίας αυτής της προσέγγισης οι σχεδιαστές δεν μπορούν να έχουν μια ολοκληρωμένη άποψη της συμπεριφοράς όλου του συστήματος μιας και γενικά δεν είναι πολύ διαδεδομένα ενοποιημένα περιβάλλοντα τα όποια μπορούν να υποστηρίξουν την συνολική προσομοίωση ετερογενών συστημάτων.

Έτσι σχεδιάζονται τα επιμέρους τμήματα και κατόπιν ενώνονται και γίνονται δοκιμές για την αλληλεπίδρασή τους. Αν παρουσιαστεί κάποιο πρόβλημα , πρέπει να ξανασχεδιαστούν.

Η αναγκαιότητα μιας πιο αποδοτικής προσέγγισης είναι εμφανής. Θα δώσει στους σχεδιαστές την δυνατότητα να αξιολογήσουν την συμπεριφορά του συστήματος πριν την κατασκευή του. Με τον τρόπο αυτό σχεδιαστικά λάθη θα γίνουν αντιληπτά σε πρώιμο στάδιο και οι αλλαγές που θα απαιτηθούν θα έχουν το ελάχιστο κόστος.

Προς αυτή την κατεύθυνση δημιουργήθηκε ένα ενοποιημένο περιβάλλον το όποιο επιτρέπει τον ταυτόχρονο σχεδιασμό και επαλήθευση των επιμέρους διαφορετικών μερών του συστήματος.

Έτσι χρησιμοποιήθηκαν κάποιες κλάσεις της C++ σε πρώτο επίπεδο .Κατόπιν γίνεται χρήση πιο λεπτομερών τμημάτων κώδικα C τα όποια αποτελούν το κομμάτι του λογισμικού του συστήματος και το VHDL (*VHSIC hardware description language*, VHSIC: *very-high-speed integrated circuit*) το ψηφιακό τμήμα του συστήματος.

Η βασική ιδέα είναι να χρησιμοποιηθούν οι σύνδεσμοι της γλώσσας C των επιμέρους προσομοιωτών για να γίνει δυνατή η επαρκής επικοινωνία και η ανταλλαγή δεδομένων μεταξύ των προσομοιωμένων μερών του συστήματος.

Για τον συγχρονισμό των διαφόρων επιμέρους προσομοιωτών , αυτά τα εργαλεία είναι συνδεδεμένα σε μορφή Master/Slave.

Μεθοδολογία για τον σχεδιασμό του συστήματος

Η γενική μεθοδολογία φαίνεται στο σχήμα 2.1. Η παρατήρηση του πραγματικού κόσμου ακολουθείται από μια επίσημη ανάλυση των απαιτήσεων του συστήματος και ένας υψηλού επίπεδου προσδιορισμός των λειτουργιών του συστήματος.

Κατόπιν η μοντελοποίηση του μηχανικού μέρους πραγματοποιείται με την κατάλληλη μέθοδο μου μας υποδεικνύει ο προσομοιωτής PSS (Physical System Simulator) που χρησιμοποιείται .

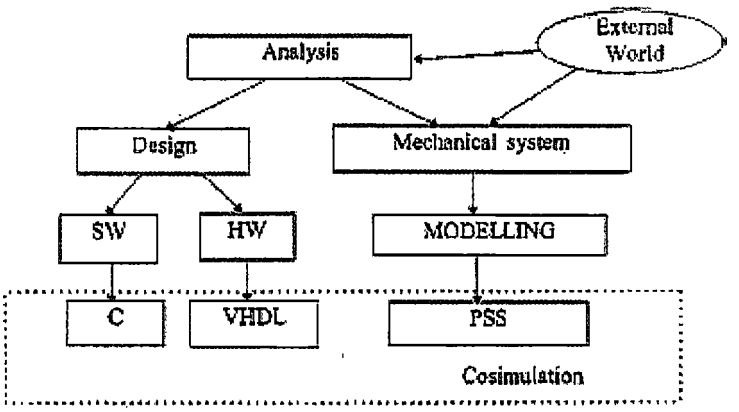
Ο σχεδιασμός του ψηφιακού μέρους μαζί με το λογισμικό ή το υλικολογισμικό το όποιο θα εκτελείται, βασίζεται σε αντικειμενοστραφείς τεχνικές. Παρόλο που αυτές οι τεχνικές χρησιμοποιούνται εκτενώς στην τεχνολογία των υπολογιστών, κυρίως για την περιγραφή των μερών λογισμικού, τις επεκτείναμε και για την περιγραφή του υλικού.

Σε αυτό το υψηλό επίπεδο της περιγραφής, είναι φυσικό ότι η κατανομή των τμημάτων των συστημάτων σε υλικό και λογισμικό δεν έχει καθοριστεί ακόμα.

Τα αποτελέσματα προσομοίωσης μαζί με κάποιες τεχνικές διαχωρισμού υλικού/λογισμικού, βοηθούν το σχεδιαστή για να καθορίσουν τον τρόπο με τον οποίο αυτές οι λειτουργίες μπορούν τελικά να εφαρμοστούν σύμφωνα με τους διαθέσιμους πόρους υλικού, το κόστος και τα όρια ταχύτητας.

Κατόπιν τα μέρη λογισμικού και υλικού καθαρίζονται σε C και συντάσσονται σε VHDL αντίστοιχα, με έναν αυτοματοποιημένο τρόπο. Τα τμήματα που προκύπτουν από κώδικα C και τα μέρη VHDL μπορούν να προσομοιωθούν μαζί και μαζί με το PSS όπως περιγράφουμε σε αυτό το έγγραφο. Κατά αυτόν τον τρόπο, ακριβέστερα αποτελέσματα μπορούν να

επιτευχθούν και ο σχεδιαστής μπορεί να έχει μια επισκόπηση για το πώς λειτουργεί ολόκληρο το σύστημα.



Σχήμα 2.1

Γενικά χαρακτηριστικά προτεινόμενου περιβάλλοντος προσομοίωσης

Τα γενικά χαρακτηριστικά των αναπτυγμένων διεπαφών, που χρησιμοποιούνται για την προσομοίωση, είναι τα ακόλουθα:

(α) Ο προσομοιωτής VHDL, το PSS και το πρόγραμμα εφαρμογής που οργανώνονται σε UNIX περιβάλλον.

(β) Υποτίθεται ότι ήταν ευδιάκριτες διαδικασίες.

(γ) Μπορούν να τρέξουν σε διαφορετικές μηχανές και η εσωτερική μεταξύ των διαδικασιών επικοινωνία εκτελείται χρησιμοποιώντας τις υποδοχές ρευμάτων που επιλέχτηκαν λόγω της αυξανόμενης ταχύτητας και της αξιοπιστίας τους.

Τα RPCs ήταν θεωρημένο επίσης ως εναλλακτική λύση αλλά δεν χρησιμοποιήθηκαν στην παρούσα έκδοση επειδή δεν είναι ακόμα πλήρως τυποποιημένοι και είναι αρκετά πιο αργοί από τις υποδοχές.

(δ) Συστήνεται η ύπαρξη ενός ενιαίου καναλιού επικοινωνίας υποδοχών μεταξύ των δύο προσομοιωτών καθώς επίσης και μεταξύ του προσομοιωτή VHDL και του προγράμματος εφαρμογής προκειμένου να αποφευχθούν τα διάφορα προβλήματα συγχρονισμού που μπορούν να προκύψουν εάν οι πολλαπλές συνδέσεις ανοίξουν.

(ε) Η διεπαφή της γλώσσας C έχει την δυνατότητα να χρησιμοποιεί διάφορους προσομοιωτές και όλος ο κώδικας διεπαφών γράφεται επίσης σε C με εξαίρεση μερικά προσαρμοστικά συστατικά που περιγράφονται σε VHDL. Αυτοί μπορούν να χρησιμοποιηθούν στην πλευρά Synopsys για τη διασύνδεση των τμημάτων των διεπαφών του υπόλοιπο ψηφιακού κυκλώματος.

(ζ) Synopsys και PSS συνεργάζονται σε μια διαμόρφωση Master/Slave. Αυτό σημαίνει ότι το πρόγραμμα C που περιγράφει το λογισμικό, είναι η αρχική κύρια ενότητα.

Όποτε το αποκλειστικό υλικό πρέπει να προσεγγιστεί, ο προσομοιωτής VHDL εκκινεί και η διαδικασία της C, εμποδίζεται έως ότου ο προσομοιωτής VHDL εκτελέσει την απαραίτητη δράση. Επιπλέον, εάν επιλεχτεί η αξία μιας φυσικής παραμέτρου συστημάτων, η ταυτότητα παραμέτρου στέλνεται στο PSS από τον προσομοιωτή VHDL, μαζί με timestamp (χρονική σήμανση).

Συνοψίζοντας βλέπουμε ότι η κοινή προσομοίωση των ετερογενών μερών ενός συστήματος που αποτελείται από τα ηλεκτρομηχανικά, ψηφιακά μέρη και τα προγράμματα εφαρμογής γίνεται εφικτή επιτρέποντας στους σχεδιαστές τέτοιων συστημάτων για να τους ελέγξει γρήγορα με το ελάχιστο κόστος. Η διεπαφή μεταξύ προσομοιωτών VHDL και προσομοιωτών ηλεκτρομηχανικοί συστημάτων χαρακτηρίζονται από την αποδοτικότητα και τη φορητότητα και καλύπτουν μια ευρεία σειρά των περιπτώσεων προσομοίωσης.

ΚΕΦΑΛΑΙΟ 3

3.1 ΕΝΑΣ ΑΠΛΟΣ ΠΡΟΣΟΜΕΙΩΤΗΣ ΠΟΛΛΑΠΛΩΝ ΔΙΕΡΓΑΣΙΩΝ

Η δυναμική συμπεριφορά πολυδιεργαστικών συστημάτων είναι πολύπλοκη και δύσκολη στην κατανόηση. Στην εργασία αυτή εξηγείτε πώς κατασκευάζεται ένας απλός προσομοιωτής πολλαπλών διεργασιών, χρησιμοποιώντας το υπολογιστικό μοντέλο της θεωρίας χρονοπρογραμματισμού. Το αποτέλεσμα είναι ένας προσομοιωτής που περιγράφει γραφικά τις συμπεριφορές των συστημάτων.

Τα σύγχρονα συστήματα βασίζονται πάρα πολύ σε λογισμικό πολλαπλών διεργασιών. Έτσι λοιπόν οι προγραμματιστές τέτοιου λογισμικού πρέπει να έχουν μια λεπτομερή κατανόηση του τρόπου συμπεριφοράς του λογισμικού τους κατά την εκτέλεσή τους.

Ο σκοπός μας λοιπόν είναι η κατασκευή ενός εργαλείου, που θα μπορεί να απεικονίσει την συμπεριφορά ενός πολυδιεργαστικού συστήματος σε μια εύκολα κατανοητή μορφή.

3.2 ΘΕΩΡΙΑ ΧΡΟΝΟΠΡΟΓΡΑΜΜΑΤΙΣΜΟΥ

Η θεωρία χρονοπρογραμματισμού σταθερής προτεραιότητας βασίζεται σε ένα συγκεκριμένο υπολογιστικό μοντέλο ενός συστήματος πολλαπλών διεργασιών. Σε αυτό το μοντέλο υπάρχουν, ένα στατικό σύνολο ταυτόχρονων διεργασιών , που βρίσκονται όλες στον ίδιο επεξεργαστή. Ένας αλγόριθμος χρονοπρογραμματισμού διαμοιράζει υπολογιστικό χρόνο στις διεργασίες σε διακριτά τμήματα.

Κάθε διεργασία έχει μια στατική βασική προτεραιότητα, παρόλο που μπορεί να πάρει προσωρινά και μια υψηλότερη ενεργή προτεραιότητα. Κατά τον χρόνο εκτέλεσης κάθε διεργασία απαιτεί έναν άπειρο αριθμό εκτελέσεων.

Ο χρόνος κατά τον οποίο η διεργασία είναι έτοιμη προς εκτέλεση, ονομάζεται *χρόνος άφιξης*. Ο χρόνος κατά το οποίο η διεργασία πραγματικά εκκινεί, ονομάζεται *χρόνος εκκίνησης*. Αυτός ο χρόνος μπορεί να διαφέρει σημαντικά από τον χρόνο άφιξης.

Ο χρόνος κατά τον οποίο μια διεργασία τελειώνει την εκτέλεση της, ονομάζεται *χρόνος λήξης*.

Για κάθε διεργασία ο προγραμματιστής καθορίζει μια χρονική προθεσμία μέσα στην οποία η διεργασία πρέπει να έχει εκτελεστεί, και μετράται σε σχέση με τον χρόνο άφιξης.

Οι προγραμματιστές επίσης χρησιμοποιούν μια άλλη παράμετρο , τον υπολογιστικό χρόνο χειρότερου σεναρίου.

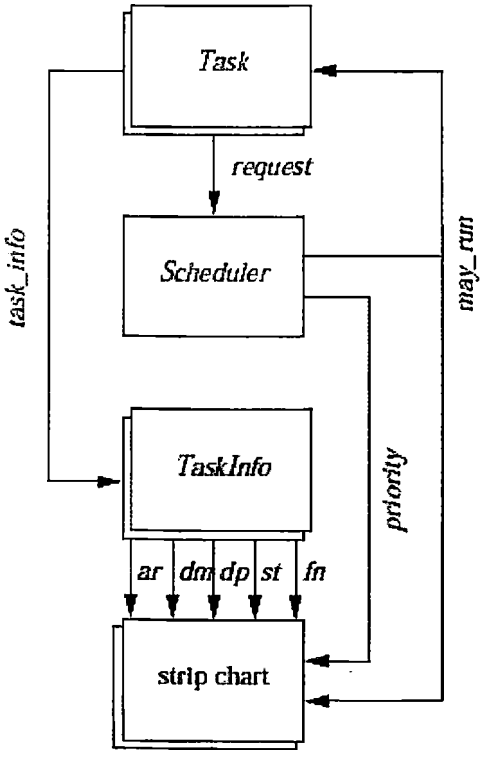
Η θεωρία παρέχει συγκεκριμένες δοκιμές χρονοπρογραμματισμού, οι οποίες επιτρέπουν στους προγραμματιστές να προβλέπουν αν ο σχεδιασμός ενός πολυδιεργαστικού συστήματος, θα τηρήσει τις προθεσμίες ή όχι.

Ο προσομοιωτής πολλαπλών διεργασιών

Ο προσομοιωτής που κατασκευάζουμε, βασίζεται στο περιβάλλον Foresight Toolkit.

Είναι ένα περιβάλλον σχεδιασμού και ανάλυσης πολύπλοκων συστημάτων. Παρέχει μια ευρεία γκάμα από εργαλεία για την κατασκευή εκτελέσιμων μοντέλων , την προσημείωση τους και την ανάλυση της συμπεριφοράς τους .

Η πρόκληση ήταν να παρουσιάσουμε το υπολογιστικό μοντέλο της θεωρίας χρονοπρογραμματισμού σε γλώσσα διαγραμματικής μορφής του περιβάλλοντος Foresight, όπως φαίνεται και στο σχήμα 3.1.



Σχήμα 3.1

Παράδειγμα

Προλαμβανόμενος εν αντιθέσει με μη προλαμβανόμενο χρονοπρογραμματισμό

Ας πούμε ότι έχουμε ένα σύνολο 3 διεργασιών με χαρακτηριστικά όπως φαίνονται στον πίνακα 3.1.

Task no.	Base priority	Period T	Comp. time C	Dead-line D	Offset O
1	1 (low)	10	4	8	0
2	2 (med)	8	3	4	2
3	3 (high)	7	2	3	1

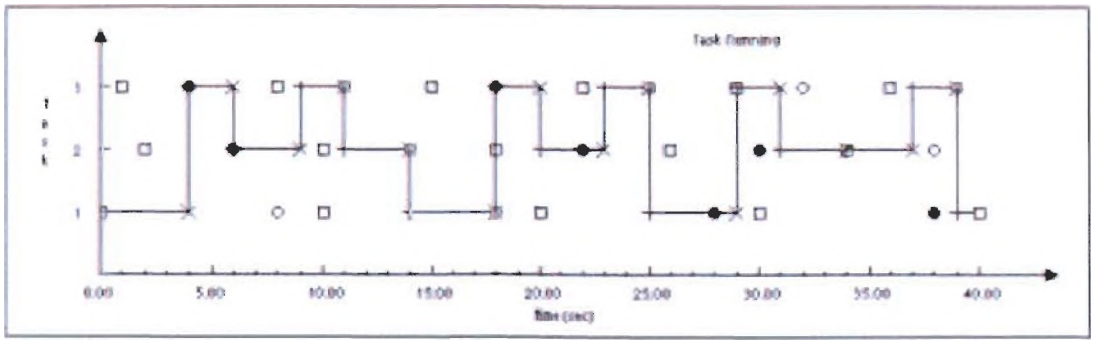
Πίνακας 3.1

Στο παράδειγμά μας η διεργασία 3 έχει την υψηλότερη προτεραιότητα και χρειάζεται μέχρι 2 δευτερόλεπτα υπολογιστικού χρόνου κάθε 7 δευτερόλεπτα.

Κάθε εκτέλεση της διεργασίας 3 απαιτείται να τελειώσει μέσα σε 3 δευτερόλεπτα από τον χρόνο άφιξης της. Μια αρχική μετατόπιση σημαίνει ότι η διεργασία 3 φτάνει στον χρόνο 1, η δεύτερη εκτέλεση στον χρόνο 8 και ούτω καθεξής. Ομοίως και για τις άλλες δύο διεργασίες.

Το σχήμα 3,2 μας δείχνει την προσομοίωση των εκτελούμενων διεργασιών , με την προϋπόθεση ότι έχει χρησιμοποιηθεί μη προλαμβάνουσα πολιτική χρονοπρογραμματισμού. Μιας και δεν υπάρχει αρχική μετατόπιση , η χαμηλότερης προτεραιότητας διεργασία 1 φτάνει στον χρόνο 0 και αρχίζει να εκτελείται αμέσως. Παρόλο που οι εκτέλεσης των υψηλότερης προτεραιότητας διεργασιών 3 και 2 φτάνουν στον χρόνο 1 αντίστοιχα, η μη προλαμβάνουσα πολιτική χρονοπρογραμματισμού σημαίνει ότι πρέπει να περιμένουν μέχρι η διεργασία 1 να τελειώσει.

Όταν αυτό γίνεται , στον χρόνο 4, η υψηλότερης προτεραιότητας διεργασία που αναμένει, δηλαδή η 3, αρχίζει να εκτελείται. Παρόλα αυτά αποδεικνύεται πολύ αργά, διότι ο χρόνος προθεσμίας της διεργασίας 3 ήταν στον χρόνο4(3 δευτερόλεπτα μετά την άφιξή της). Έτσι η προθεσμία είχε εκπνεύσει όπως φαίνεται και στο σχήμα 3.2.

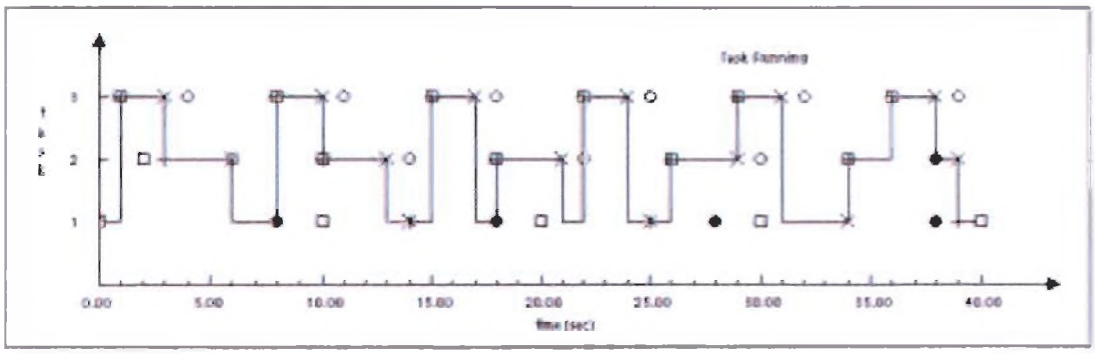


Σχήμα 3.2

Άμεσα λοιπόν βλέπουμε ότι ο μη προλαμβάνων χρονοπρογραμματισμός έχει ένα μεγάλο μειονέκτημα.

Δεν έχει τρόπο να αποτρέψει χαμηλής προτεραιότητας διεργασίες να οδηγήσουν υψηλότερης προτεραιότητας διεργασίες στο να αποτύχουν.

Με την χρήση προλαμβάνουσας πολιτική χρονοπρογραμματισμού βλέπουμε ότι έχουμε πολύ καλύτερη συμπεριφορά του συστήματός μας σε σχέση με την τήρηση των προθεσμιών όπως φαίνεται και στο σχήμα 3.3.



Σχήμα 3.3

ΣΥΜΠΕΡΑΣΜΑ

Είδαμε πώς ένας προσομοιωτής πολλαπλών διεργασιών γρήγορα και εύκολα κατασκευάστηκε χρησιμοποιώντας το υπολογιστικό μοντέλο της θεωρίας χρονοπρογραμματισμού σε έναν γενικής χρήσης προσομοιωτή

ΚΕΦΑΛΑΙΟ 4

4.1 ΣΧΕΤΙΚΑ ΜΕ ΤΟ STRESS

Το σύστημα STRESS, που είναι η εργασία στην οποία κυρίως βασίστηκε η πτυχιακή εργασία, είναι μια συλλογή από εργαλεία που προσομοιώνει τη συμπεριφορά των hard real-time συστημάτων. Προορίζεται πρώτιστα ως μέσο με το οποίο διάφοροι αλγόριθμοι σχεδιασμού και διαχείρισης των πόρων μπορούν να αξιολογηθούν και να χρησιμοποιηθούν για να μελετήσουν τη γενική συμπεριφορά των εφαρμογών και των "hard-real time " συστημάτων.

Το περιβάλλον STRESS περιλαμβάνει διάφορα χωριστά εργαλεία συμβάλλοντας στην προσομοίωση των hard real-time συστημάτων, που χτίζεται επάνω σε μια ενιαία γλώσσα προσομοίωσης. Αυτή η γλώσσα σχεδιάζεται συγκεκριμένα όσον αφορά τα hard real-time συστήματα, για να επιτρέψει την έκφραση της περιγραφής "hardware", το λογισμικό πυρήνων και τα προγράμματα εφαρμογών.

Το περιβάλλον STRESS αποτελείται από τις παρακάτω γενικές ενότητες:

1. Η διεπαφή του περιβάλλοντος STRESS
2. Ανάλυση του εφικτού
3. Προσομοιωτής εκτέλεσης
4. Εργαλείο απεικόνισης

Στο περιβάλλον STRESS βασικές έννοιες που χρησιμοποιούνται είναι οι παρακάτω:

I. **Διεργασίες.**

Μια διεργασία είναι η βασική μονάδα ενός συστήματος αλληλεπιδρώντων υπολογισμών. Οι διεργασίες χωρίζονται σε τρεις βασικούς τύπους: περιοδικές, μη περιοδικές και σποραδικές. Οι περιοδικές διεργασίες ενσωματώνονται στις λέξεις κλειδιά periodic και endper (ομοίως απεριοδικές διεργασίες σε aperiodic... endape και σποραδικές διεργασίες σε sporadic... endspo). Ανάλογα με τον τύπο διεργασίας, διάφορα χαρακτηριστικά μπορούν να τεθούν (μερικά των οποίων είναι προαιρετικά). Παραδείγματος χάριν, οι περιοδικές διεργασίες απαιτούν τις τιμές, περίοδος(period) και προθεσμία (deadline) για να τεθούν σε λειτουργία.

II. Σηματοφόροι.

Οι σηματοφόροι παρέχουν μια βασική μέθοδο να δημιουργείται αμοιβαίος αποκλεισμός μεταξύ διεργασιών στον ίδιο επεξεργαστή ή τον κόμβο. Οι σηματοφόροι που βρίσκονται σε έναν κόμβο μπορούν να προσπελαστούν από οποιοδήποτε επεξεργαστή στον κόμβο. Ένας σηματοφόρος σε έναν επεξεργαστή μπορεί μόνο να προσεγγιστεί από εκείνο τον επεξεργαστή.

III. Αντικείμενα και μέθοδοι.

Τα αντικείμενα παρέχουν την κοινή επικοινωνία μνήμης μεταξύ των διεργασιών μέσα σε έναν ενιαίο κόμβο. Ένα αντικείμενο δεν μπορεί να προσπελαστεί από μια διεργασία που τρέχει σε έναν διαφορετικό κόμβο. Τα αντικείμενα προσπελαύνονται μέσω των μεθόδων, και μπορούν να περιέχουν τοπικές μεταβλητές και τοπικούς σηματοφόρους.

4.2 ΣΥΜΠΕΡΑΣΜΑ ΣΧΕΤΙΚΑ ΜΕ ΤΟ ΠΕΡΙΒΑΛΛΟΝ STRESS

Το περιβάλλον STRESS είναι εύλογα ώριμο και χρησιμοποιείται αυτήν την περίοδο από πολλά όργανα ακαδημαϊκής και βιομηχανικής έρευνας. Ο προσομοιωτής STRESS παρέχει ένα ευέλικτο περιβάλλον στο οποίο οι διάφοροι αλγόριθμοι χρονοπρογραμματισμού και διαχείρισης πόρων μπορούν να αξιολογηθούν και να δοκιμαστούν. Παρέχει επίσης μέσα αξιολόγησης και δοκιμής συνόλων διεργασιών και την εφαρμογή αντίστοιχων πυρήνων που αντιστοιχούν σε πραγματικές εφαρμογές.

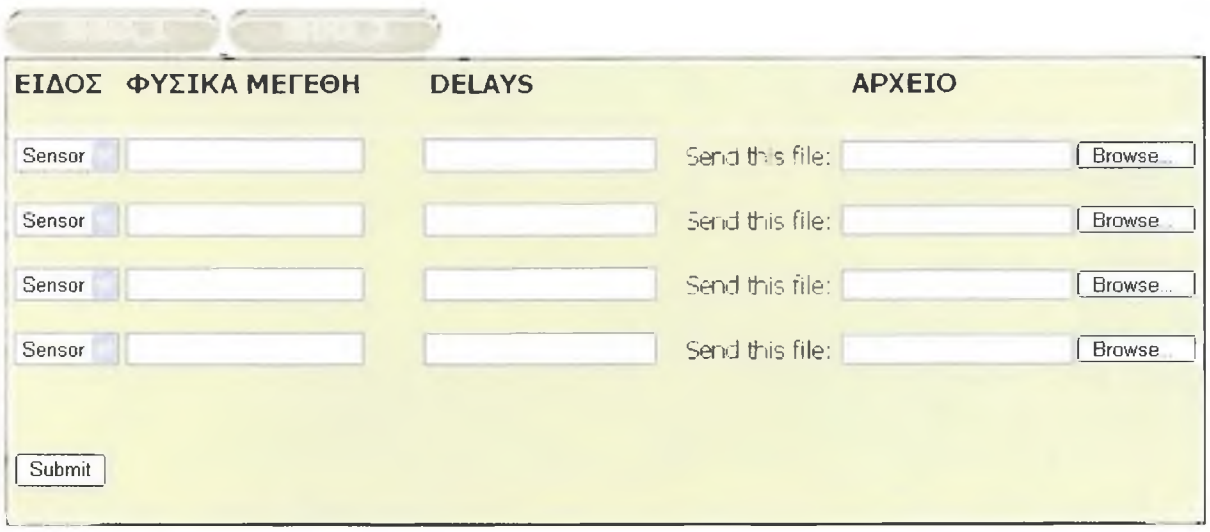
Τέλος, αναμένεται ότι το περιβάλλον STRESS θα επεκταθεί για να επιτρέψει την παραγωγή κώδικα (ή τη μετάφραση σε μια άλλη υψηλού επιπέδου γλώσσα, π.χ. C++) για να εκτελεί εφαρμογές και πυρήνες επάνω σε μια υπό ανάπτυξη δοκιμή .

ΚΕΦΑΛΑΙΟ 5

5.1 ΔΗΜΙΟΥΡΓΙΑ ΤΗΣ ΠΡΩΤΗΣ ΙΣΤΟΣΕΛΙΔΑΣ (WEB PAGE STEP ONE)

Στόχος της πτυχιακής εργασίας είναι η δημιουργία ενός προγράμματος γραμμένου σε υψηλού επιπέδου Γλώσσας που θα προσομοιώνει κάποιες διεργασίες και θα έχει ως αποτέλεσμα την προβολή των χρόνων εκτέλεσης τους. Το πρόγραμμά μας όπως είναι λογικό έχει κάποιες εισόδους (παίρνει δεδομένα από τον χρήστη). Αυτά τα δεδομένα ο χρήστης όπως έχουμε πε και παραπάνω τα εισάγει στο σύστημα μέσω μιας ιστοσελίδας.

Η πρώτη σελίδα του ιστότοπου αφορά τις τιμές των αισθητήρων. Εδώ γίνεται η καταχώρηση των ονομάτων των φυσικών μεγεθών, που είναι τα ονόματα των τιμών που μετράει ο κάθε αισθητήρας. Επίσης δηλώνεται από τον χρήστη η καθυστέρηση που έχει ο κάθε αισθητήρας δηλαδή ο χρόνος που καταναλώνει κατά την λήψη και την μεταφορά της πληροφορίας. Αυτό όπως φαίνεται και στο σχήμα (5.1.1) ονομάζεται "DELAY".Στην συνέχεια ο χρήστης ανεβάζει ένα αρχείο με τυχαίες τιμές (που μπορούν να καταγράψουν οι αισθητήρες) οι οποίες είναι διαθέσιμες να χρησιμοποιηθούν στην προσομοίωση. Το αρχείο που θα ανεβάσει θα πρέπει να έχει το ίδιο όνομα (όνομα.txt) με το όνομα του φυσικού μεγέθους.

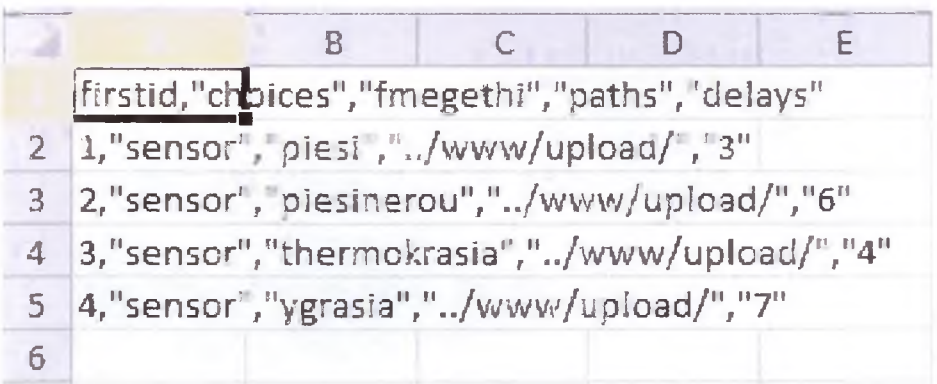


Σχήμα 5.1.1: Ιστότοπος->Βήμα πρώτο.

Πίσω από αυτό έχει δημιουργηθεί μια βάση δεδομένων (MySQL) στην οποία αποθηκεύονται τα δεδομένα που προαναφέραμε. Η ιστοσελίδα συνδέεται με την βάση δεδομένων μέσω της γλώσσας PHP. Η γλώσσα PHP καλεί το κάθε δεδομένο από την φόρμα της ιστοσελίδας και το στέλνει στην βάση δεδομένων.

Τα δεδομένα που αποθηκεύονται στην βάση δεδομένων για να διαβαστούν από τον κώδικα θα πρέπει να είναι σε διαφορετική μορφή. Έτσι η PHP δίνει την εντολή αφού αποθηκευτούν στην βάση δεδομένων οι καταχωρήσεις του χρήστη να δημιουργηθεί ένα CSV αρχείο

σχήμα (5.1.2) μέσα στο οποίο γράφονται τα δεδομένα.



Σχήμα 5.1.2: Ιστότοπος-> Πρώτο αρχείο CSV.

5.2 ΚΩΔΙΚΑΣ ΓΛΩΣΣΑΣ PHP (STEP_ONE)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Step one</title>
</head>
<body>
<?php
//Orismos arxikwn metavlitwn
  $fys1 = $_POST['fys1'];
  $fys2 = $_POST['fys2'];
  $fys3 = $_POST['fys3'];
  $fys4 = $_POST['fys4'];
  $ch1 = $_POST['ch1'];
  $ch2 = $_POST['ch2'];
  $ch3 = $_POST['ch3'];
  $ch4 = $_POST['ch4'];
  $de1 = $_POST['de1'];
  $de2 = $_POST['de2'];
  $de3 = $_POST['de3'];
  $de1 = $_POST['de4'];
?>
<?php
```

```
set_time_limit(0);
move_uploaded_file ($_FILES['userfile1'] ['tmp_name'],
    "../www/upload/{$_FILES['userfile1'] ['name']}");
echo ($_FILES['userfile1']['size']);
echo ("<br>");
echo ($_FILES['userfile1']['type']);
echo ("<br>");
echo ($_FILES['userfile1']['name']);
echo ("<br>");
echo ($_FILES['userfile1']['tmp_name']);
echo ("<br>");
set_time_limit(0);
move_uploaded_file ($_FILES['userfile2'] ['tmp_name'],
    "../www/upload/{$_FILES['userfile2'] ['name']}");
echo ($_FILES['userfile2']['size']);
echo ("<br>");
echo ($_FILES['userfile2']['type']);
echo ("<br>");
echo ($_FILES['userfile2']['name']);
echo ("<br>");
echo ($_FILES['userfile2']['tmp_name']);
echo ("<br>");
set_time_limit(0);
move_uploaded_file ($_FILES['userfile3'] ['tmp_name'],
    "../www/upload/{$_FILES['userfile3'] ['name']}");
echo ($_FILES['userfile3']['size']);
```

```
echo ("<br>");
echo ($_FILES['userfile3']['type']);
echo ("<br>");
echo ($_FILES['userfile3']['name']);
echo ("<br>");
echo ($_FILES['userfile3']['tmp_name']);
echo ("<br>");
set_time_limit(0);
move_uploaded_file ($_FILES['userfile4'] ['tmp_name'],
    "../www/upload/{$_FILES['userfile4'] ['name']}");
echo ($_FILES['userfile4']['size']);
echo ("<br>");
echo ($_FILES['userfile4']['type']);
echo ("<br>");
echo ($_FILES['userfile4']['name']);
echo ("<br>");
echo ($_FILES['userfile4']['tmp_name']);
echo ("<br>");
set_time_limit(0);
?>
<?php
$uploadDir = '../www/upload/';
/* dhmiourgia $filePath1 */
$filePath1 = $uploadDir;
/* dhmiourgia $filePath2 */
$filePath2 = $uploadDir;
```

```
/* dhmiourgia $filePath3 */
$filePath3 = $uploadDir;
/* dhmiourgia $filePath4 */
$filePath4 = $uploadDir;
//syndesi me tin vasi
$con = mysql_connect("localhost","root","1234");
if (!$con)
 {
 die('Could not connect: ' . mysql_error());
 }
mysql_select_db("systhma", $con);
$sql="INSERT INTO first (firstid,choices,fmegethi,paths,delays)
VALUES
('1','$_POST[ch1]','$_POST[fys1]','$filePath1','$_POST[de1]')";

if (!mysql_query($sql,$con))
 {
 die('Error: ' . mysql_error());
 }
echo "1 record added";
mysql_close($con)
?>

<?php
$con = mysql_connect("localhost","root","1234");
if (!$con)
```

```
 {
 die('Could not connect: ' . mysql_error());
 }
mysql_select_db("systhma", $con);
$sql="INSERT INTO first (firstid,choices,fmegethi,paths,delays)
VALUES
('2','$_POST[ch2]','$_POST[fys2]','$filePath2','$_POST[de2]')";
if (!mysql_query($sql,$con))
 {
 die('Error: ' . mysql_error());
 }
echo "1 record added";
mysql_close($con)
?>
<?php
$con = mysql_connect("localhost","root","1234");
if (!$con)
 {
 die('Could not connect: ' . mysql_error());
 }
mysql_select_db("systhma", $con);
$sql="INSERT INTO first (firstid,choices,fmegethi,paths,delays)
VALUES
('3','$_POST[ch3]','$_POST[fys3]','$filePath3','$_POST[de3]')";
if (!mysql_query($sql,$con))
 {
```

```
 die('Error: ' . mysql_error());
 }
echo "1 record added";
mysql_close($con)
?>
<?php
$con = mysql_connect("localhost","root","1234");
if (!$con)
 {
 die('Could not connect: ' . mysql_error());
 }
mysql_select_db("systhma", $con);
$sql="INSERT INTO first (firstid,choices,fmegethi,paths,delays)
VALUES
('4','$_POST[ch4]','$_POST[fys4]','$filePath4','$_POST[de4]')";
if (!mysql_query($sql,$con))
 {
 die('Error: ' . mysql_error());
 }
echo "1 record added";
mysql_close($con)
?>

<?php
$server="localhost";
$db="systhma";
```

```
$table="first";
$login="root";
$password="1234";
$filename="export.csv";
mysql_connect($server, $login, $password);
mysql_select_db($db);
$fp = fopen($filename, "w");
$res = mysql_query("SELECT * FROM $table");
// fetch a row and write the column names out to the file
$row = mysql_fetch_assoc($res);
$line = "";
$comma = "";
foreach($row as $name => $value) {
   $line .= $comma . "" . str_replace("", """, $name) . "";
   $comma = ",";
}
$line .= "\n";
fputs($fp, $line);
// remove the result pointer back to the start
mysql_data_seek($res, 0);
$row = mysql_fetch_assoc($res);
   $line = "";
   $comma = "";
   foreach($row as $value) {
      $line .= $comma . "" . str_replace("", """, $value) . "";
      $comma = ",";
```

```
    }
    $line .= "\n";
    fputs($fp, $line);
$row = mysql_fetch_assoc($res);
    $line = "";
    $comma = "";
    foreach($row as $value) {
        $line .= $comma . "" . str_replace("", """, $value) . "";
        $comma = ",";
    }
    $line .= "\n";
    fputs($fp, $line);
$row = mysql_fetch_assoc($res);
$line = "";
    $comma = "";
    foreach($row as $value) {
        $line .= $comma . "" . str_replace("", """, $value) . "";
        $comma = ",";
    }
    $line .= "\n";
    fputs($fp, $line);
$row = mysql_fetch_assoc($res);
$line = "";
    $comma = "";
    foreach($row as $value) {
        $line .= $comma . "" . str_replace("", """, $value) . "";
```

```
        $comma = ",";
    }
    $line .= "";
    fputs($fp, $line);
fclose($fp);
?>

<?php
$server="localhost";
$db="systhma";
$table="first";
$login="root";
$password="1234";
$db1=mysql_connect($server, $login, $password);
mysql_select_db($db);
$query="DELETE FROM $table";
$result=mysql_query($query);
mysql_close($db1);
?>
</body>
</html>
```

5.3 ΚΩΔΙΚΑΣ ΓΛΩΣΣΑΣ HTML (STEP_ONE)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-7" />
<title>Step One</title>
</head>

<body style="font-family:Tahoma">
<a href="index.html"> <img  src="step_1.jpg" border="0" /> </a>
<a href="step_2.html"> <img  src="step_2.jpg" border="0" /> </a>
<a href="result.hta"> <img src="result.jpg" border="0" /></a></p>
<br />
<table bgcolor="e8f6a9" border="2" bordercolor="#104406">
<tr bordercolor="#FFFFFF">
<td bordercolor="#FFFFFF"> <H3>ΕΙΔΟΣ   ΦΥΣΙΚΑ ΜΕΓΕΘΗ
        DELAYS
        
     
   
                         APXEIO</H3>
 </td>
</tr>
<tr>
<td bordercolor="#FFFFFF">
```

```
<form action="upload1.php" method="post" enctype="multipart/form-data" >
<select name="ch1">
<option value="sensor">Sensor</option>

</select>
<input type="text" name="fys1" maxlength="50"
/>       
<input type="text" name="de1" maxlength="50" />  
<input type="hidden" name="MAX_FILE_SIZE" value="10000">
   Send this file:
   <input name="userfile1" type="file">
    <br />
<select name="ch2">
<option value="sensor">Sensor</option>
</select>
<input type="text" name="fys2" maxlength="50"
/>       
<input type="text" name="de2" maxlength="50" />  
<input type="hidden" name="MAX_FILE_SIZE" value="10000">
   Send this file:
   <input name="userfile2" type="file">
    <br />
<select name="ch3">
<option value="sensor">Sensor</option>
</select>
<input type="text" name="fys3" maxlength="50"
/>       
```

```
<input type="text" name="de3" maxlength="50" />  
<input type="hidden" name="MAX_FILE_SIZE" value="10000">
   Send this file:
   <input name="userfile3" type="file">
    <br />
<select name="ch4">
<option value="sensor">Sensor</option>
</select>
<input type="text" name="fys4" maxlength="50"
/>       
<input type="text" name="de4" maxlength="50" />  
<input type="hidden" name="MAX_FILE_SIZE" value="10000">
   Send this file:
   <input name="userfile4" type="file">
 <br />
 <br />
 <br />
<input name="upload" type="submit" value="Submit" />
</form>
</td>
</tr>
</table>
</body>
</html>
```

5.4 ΚΩΔΙΚΑΣ ΓΛΩΣΣΑΣ MySQL (STEP ONE)

ΣΥΝΔΕΣΗ ΣΤΗΝ MySQL :

```
mysql –h localhost -u root -p
enter password: 1234
```

ΔΗΜΙΟΥΡΓΙΑ ΤΗΣ ΒΑΣΗΣ ΔΕΔΟΜΕΝΩΝ:

```
create database systhma;
```

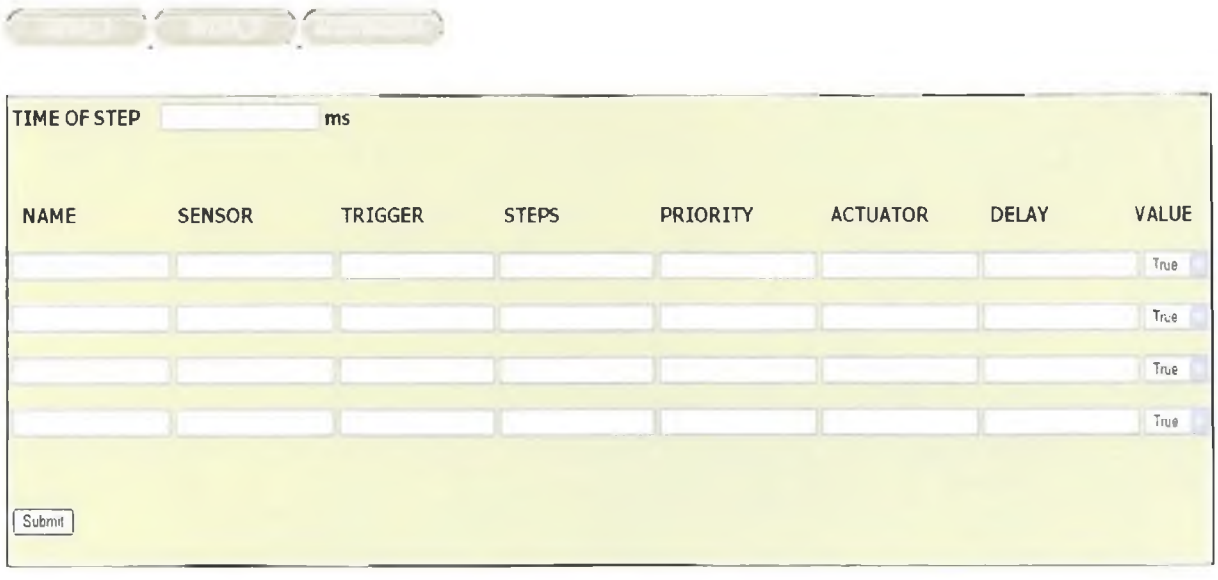
ΕΝΤΟΛΕΣ ΓΙΑ ΤΗΝ ΔΗΜΙΟΥΡΓΙΑ "TABLE"

```
use systhma;
create table first
(
firstid char(2) not null,
choices char(30) not null,
fmegethi char(50) not null,
delays char(50) not null,
paths varchar(60) not null
)
;
```

ΚΕΦΑΛΑΙΟ 6

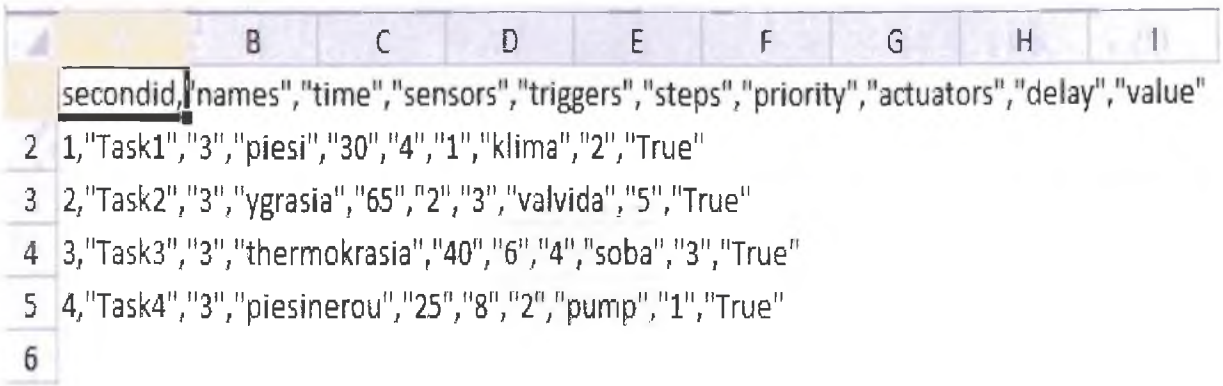
6.1 ΔΗΜΙΟΥΡΓΙΑ ΤΗΣ WEB ΙΣΤΟΣΕΛΙΔΑΣ (STEP TWO)

Στο δεύτερο βήμα ο χρήστης συνεχίζει να δηλώνει δεδομένα απαραίτητα για την εκτέλεση του προγράμματος. Πρώτα καταχωρεί τον χρόνο που διαρκεί το κάθε βήμα (Time of step) μιας διεργασίας σε milisecond (ms). Στην συνέχεια καταχωρεί το όνομα της διεργασίας (π.χ Task1) και το όνομα του αισθητήρα. Η επόμενη καταχώρηση είναι το "Trigger" το οποίο είναι η τιμή που όταν την υπερβεί η τιμή του αισθητήρα, ενεργοποιείται ο ενεργοποιητής (actuator). Ύστερα ο χρήστης δηλώνει των αριθμό των βημάτων του "actuator" και την προτεραιότητα "priority" που δίνει στην κάθε διεργασία. Επόμενη δήλωση είναι το όνομα που δίνει ο χρήστης στον "actuator" και την καθυστέρηση που έχει μέχρι την ενεργοποίηση του. Τέλος όπως φαίνεται και στο σχήμα 3 ο χρήστης δηλώνει τον σηματοφόρο "semaphore" με όνομα "value" και τιμές "true" or "false". Ο σηματοφόρος δίνει το σήμα έναρξης μίας διεργασίας. Όλα τα παραπάνω απεικονίζονται αναλυτικά στο σχήμα (6.1.1) .



Σχήμα 6.1.1: Ιστότοπος-> Βήμα δεύτερο.

Όπως και στο πρώτο βήμα τα αρχεία αποθηκεύονται σε έναν δεύτερο πίνακα της βάσης δεδομένων MySQL ,τα οποία στέλνονται εκεί μέσω της PHP. Επίσης με την εντολή που υπάρχει μέσα στην PHP τα δεδομένα αποθηκεύονται σε ένα αρχείο CSV (σχήμα 6.1.2) από όπου και διαβάζονται από τον κώδικα.



	B	C	D	E	F	G	H	I
	secondid,"names","time","sensors","triggers","steps","priority","actuators","delay","value"							
2	1,"Task1","3","piesi","30","4","1","klima","2","True"							
3	2,"Task2","3","ygrasia","65","2","3","valvida","5","True"							
4	3,"Task3","3","thermokrasia","40","6","4","soba","3","True"							
5	4,"Task4","3","piesinerou","25","8","2","pump","1","True"							
6								

Σχήμα 6.1.2: Ιστότοπος-> Δεύτερο αρχείο CSV.

6.2 ΚΩΔΙΚΑΣ ΓΛΩΣΣΑΣ HTML (STEP_TWO)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-7" />
<title>Second Step</title>
</head>
<body style="font-family:Tahoma">
// Δημιουργία μενού ιστοσελίδας
<a href="index.html"> <img  src="step_1.jpg" border="0" /> </a>
<a href="step_2.html"> <img  src="step_2.jpg" border="0" /> </a>
<a href="result.html"> <img src="result.jpg" border="0" /></a></p>
<br />
//Δημιουργία  πίνακα
<table bgcolor="e8f6a9" border="2" bordercolor="#104406">
<tr bordercolor="#FFFFFF">
<td bordercolor="#FFFFFF">
// Δημιουργία φόρμας εισαγωγής δεδομένων
<form action="upload2.php" method="post" enctype="multipart/form-data" >
<h3> TIME OF STEP   
 <input type="text" name="time" maxlength="25" /> ms</h3>
 <br />
 <br />
<h3>   NAME
               
```

```
      SENSOR
      
         TRIGGER
      
        STEPS
       
           PRIORITY
            &nbs
p; ACTUATOR
           DELAY&nb
sp;            &n
bsp;   VALUE </h3>
<input type="text" name="name1" maxlength="50" />
<input type="text" name="s1" maxlength="50" />
<input type="text" name="tr1" maxlength="50" />
<input type="text" name="st1" maxlength="50" />
<input type="text" name="pr1" maxlength="50" />
<input type="text" name="act1" maxlength="50" />
<input type="text" name="del1" maxlength="50" />
<select name="val1">
<option value="True"> True</option>
<option value="False"> False</option>
</select>
<br />
<input type="text" name="name2" maxlength="50" />
<input type="text" name="s2" maxlength="50" />
<input type="text" name="tr2" maxlength="50" />
<input type="text" name="st2" maxlength="50" />
<input type="text" name="pr2" maxlength="50" />
```

```
<input type="text" name="act2" maxlength="50" />
<input type="text" name="del2" maxlength="50" />
<select name="val2">
<option value="True"> True</option>
<option value="False"> False</option>
</select>
<br />
<input type="text" name="name3" maxlength="50" />
<input type="text" name="s3" maxlength="50" />
<input type="text" name="tr3" maxlength="50" />
<input type="text" name="st3" maxlength="50" />
<input type="text" name="pr3" maxlength="50" />
<input type="text" name="act3" maxlength="50" />
<input type="text" name="del3" maxlength="50" />
<select name="val3">
<option value="True"> True</option>
<option value="False"> False</option>
</select>
<br />
<input type="text" name="name4" maxlength="50" />
<input type="text" name="s4" maxlength="50" />
<input type="text" name="tr4" maxlength="50" />
<input type="text" name="st4" maxlength="50" />
<input type="text" name="pr4" maxlength="50" />
<input type="text" name="act4" maxlength="50" />
<input type="text" name="del4" maxlength="50" />
```

```
<select name="val4">
<option value="True"> True</option>
<option value="False"> False</option>
</select>
 <br />
  <br />
  <br />
 <input name="upload" type="submit" value="Submit" />
</form>
</td>
</tr>
</table>
</body>
</html>
```

6.3 ΚΩΔΙΚΑΣ ΓΛΩΣΣΑΣ PHP (STEP_TWO)

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>Untitled Document</title>
</head>
<body>

<?php
//Ορισμός αρχικών μεταβλητών
$time = $_POST['time'];
$onoma1 = $_POST['name1'];
$onoma2 = $_POST['name2'];
$onoma3 = $_POST['name3'];
$onoma4 = $_POST['name4'];
$s1 = $_POST['s1'];
$s2 = $_POST['s2'];
$s3 = $_POST['s3'];
$s4 = $_POST['s4'];
$tr1 = $_POST['tr1'];
$tr2 = $_POST['tr2'];
$tr3 = $_POST['tr3'];
$tr4 = $_POST['tr4'];
```

```
$st1 = $_POST['st1'];

$st2 = $_POST['st2'];

$st3 = $_POST['st3'];

$st4 = $_POST['st4'];

$pr1 = $_POST['pr1'];

$pr2 = $_POST['pr2'];

$pr3 = $_POST['pr3'];

$pr4 = $_POST['pr4'];

$act1 = $_POST['act1'];

$act2 = $_POST['act2'];

$act3 = $_POST['act3'];

$act4 = $_POST['act4'];

$va1 = $_POST['val1'];

$va2 = $_POST['val2'];

$va3 = $_POST['val3'];

$va4 = $_POST['val4'];

?>
```

```
<?php

// Σύνδεση με βάση δεδομένων και καταχώρηση των τιμών των πεδίων

$con = mysql_connect("localhost","root","1234");

if (!$con)

  {
```

```
 die('Could not connect: ' . mysql_error());
 }
mysql_select_db("systhma", $con);
$sql="INSERT INTO second
(secondid,names,time,sensors,triggers,steps,priority,actuators,delay,value)
VALUES
('1','$_POST[name1]','$_POST[time]','$_POST[s1]','$_POST[tr1]','$_POST[st1]','$_POST[pr
1]','$_POST[act1]','$_POST[del1]','$_POST[val1]')";
if (!mysql_query($sql,$con))
 {
 die('Error: ' . mysql_error());
 }
echo "1 record added";
mysql_close($con)
?>


<?php
$con = mysql_connect("localhost","root","1234");
if (!$con)
 {
 die('Could not connect: ' . mysql_error());
 }
mysql_select_db("systhma", $con);
$sql="INSERT INTO second
(secondid,names,time,sensors,triggers,steps,priority,actuators,delay,value)
VALUES
```

```
('2','$_POST[name2]','$_POST[time]','$_POST[s2]','$_POST[tr2]','$_POST[st2]','$_POST[pr
2]','$_POST[act2]','$_POST[del2]','$_POST[val2]')";
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
echo "1 record added";
mysql_close($con)
?>


<?php
$con = mysql_connect("localhost","root","1234");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }
mysql_select_db("systhma", $con);
$sql="INSERT INTO second
(secondid,names,time,sensors,triggers,steps,priority,actuators,delay,value)
VALUES
('3','$_POST[name3]','$_POST[time]','$_POST[s3]','$_POST[tr3]','$_POST[st3]','$_POST[pr
3]','$_POST[act3]','$_POST[del3]','$_POST[val3]')";
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
echo "1 record added";
```

```
mysql_close($con)
?>
<?php
$con = mysql_connect("localhost","root","1234");
if (!$con)
  {
  die('Could not connect: ' . mysql_error());
  }
mysql_select_db("systhma", $con);
$sql="INSERT INTO second
(secondid,names,time,sensors,triggers,steps,priority,actuators,delay,value)
VALUES
('4','$_POST[name4]','$_POST[time]','$_POST[s4]','$_POST[tr4]','$_POST[st4]','$_POST[pr
4]','$_POST[act4]','$_POST[del4]','$_POST[val4]')";
if (!mysql_query($sql,$con))
  {
  die('Error: ' . mysql_error());
  }
echo "1 record added";
mysql_close($con)
?>
```

```
<?php
// Σύνδεση με βάση δεδομένων
$server="localhost";
$db="systhma";
```

```
$table="second";
$login="root";
$password="1234";
$filename="export2.csv";
mysql_connect($server, $login, $password);
mysql_select_db($db);
$fp = fopen($filename, "w");
$res = mysql_query("SELECT * FROM $table");
// Διάβασμα πίνακα και αποθήκευση των ονομάτων των πεδίων  σε αρχείο
$row = mysql_fetch_assoc($res);
$line = "";
$comma = "";
foreach($row as $name => $value) {
   $line .= $comma . "" . str_replace("", """, $name) . "";
   $comma = ",";
}
$line .= "\n";
fputs($fp, $line);



// Επαναφορά του δείκτη αποτελέσματος (result pointer) στην αρχική του τιμή.
mysql_data_seek($res, 0);
// Δημιουργία βρόχου που διαβάζει τα δεδομένα
  $row = mysql_fetch_assoc($res);
   $line = "";
   $comma = "";
```

```
    foreach($row as $value) {
        $line .= $comma . '"' . str_replace('"', '""', $value) . '"';
        $comma = ",";
    }
    $line .= "\n";
    fputs($fp, $line);
$row = mysql_fetch_assoc($res);
    $line = "";
    $comma = "";
    foreach($row as $value) {
        $line .= $comma . '"' . str_replace('"', '""', $value) . '"';
        $comma = ",";
    }
    $line .= "\n";
    fputs($fp, $line);
$row = mysql_fetch_assoc($res);
    $line = "";
    $comma = "";
    foreach($row as $value) {
        $line .= $comma . '"' . str_replace('"', '""', $value) . '"';
        $comma = ",";
    }
    $line .= "\n";
    fputs($fp, $line);

$row = mysql_fetch_assoc($res);
```

```
    $line = "";
    $comma = "";
    foreach($row as $value) {
        $line .= $comma . '"' . str_replace('"', '""', $value) . '"';
        $comma = ",";
    }
    $line .= "";
    fputs($fp, $line);
fclose($fp);
?>

<?php
// Σύνδεση με πίνακα και καθαρισμός των τιμών του
$server="localhost";
$db="systhma";
$table="second";
$login="root";
$password="1234";
$db1=mysql_connect($server, $login, $password);
mysql_select_db($db);
$query="DELETE FROM $table";
$result=mysql_query($query);
mysql_close($db1);
?>
</body>
</html>
```

6.4 ΚΩΔΙΚΑΣ ΓΛΩΣΣΑΣ MySQL (STEP_TWO)

ΣΥΝΔΕΣΗ ΣΤΗΝ MySQL :

```
mysql –h localhost -u root -p
enter password: 1234
```

ΕΝΤΟΛΕΣ ΓΙΑ ΤΗΝ ΔΗΜΙΟΥΡΓΙΑ "TABLE"

```
use systhma;
create table second
(
secondid char(2) not null,
names char(30) not null,
time char(5) not null,
sensors char(50) not null,
triggers char(5) not null,
steps char(4) not null,
priority char(1) not null,
actuators char(50) not null,
delay char(5) not null,
value char(10) not null
)
;
```

ΚΕΦΑΛΑΙΟ 7

7.1 ΚΩΔΙΚΑΣ ΕΦΑΡΜΟΓΗΣ ΠΡΟΣΟΜΟΙΩΣΗΣ (ΚΩΔΙΚΑΣ ΣΕ ΓΛΩΣΣΑ C++)

Η πτυχιακή εργασία αυτή βασίζεται κυρίως πάνω στο γενικό πλάνο του περιβάλλοντος "STRESS" που έχουμε προαναφέρει. Εδώ όμως βλέπουμε μια διαφοροποίηση όσον αφορά το πρακτικό επίπεδο διότι προστέθηκε κάτι πολύ βασικό. Αυτό είναι ο προγραμματισμός των εφαρμογών σε κώδικα C++. Δηλαδή ο προγραμματισμός των αλληλεπιδράσεων των διεργασιών που συμμετέχουν στο σύστημα.

Ο βασικός στόχος του προγραμματισμού είναι να εκτελούνται τέσσερις (4) διεργασίες με ορισμένη σειρά προτεραιότητας εκτέλεσης (την οποία ορίζει ο χρήστης). Ανάλογα τις τυχαίες τιμές (τις οποίες ανεβάζει σε ένα αρχείο κειμένου ο χρήστης σε μορφή txt) που θα παίρνουν οι αισθητήρες (sensors) θα ενεργοποιούνται ή όχι οι ενεργοποιητές "actuators".

Καθοριστικό ρόλο στην διαδικασία ενεργοποίησης και απενεργοποίησης των "actuators" παίζουν δύο δείκτες. Ο πρώτος λέγεται "trigger" και δείχνει την τιμή-κατώφλι που παίρνει ο ένας αισθητήρας, έτσι ώστε να ενεργοποιηθεί ο αντίστοιχος "actuator".Δηλαδή αν η τιμή που παίρνει ο αισθητήρας είναι μεγαλύτερος της τιμής, που έχει το "trigger" τότε ανάλογα ενεργοποιείται ο "actuator". Ο οποίος με την σειρά του διαβάζει έναν άλλο δείκτη τον σηματοφόρο "value" και ανάλογα με το αν είναι "true" ή "false" ενεργοποιεί ή απενεργοποιεί ένα "actuator".Ο "actuator" ενεργοποιείται ή απενεργοποιείται μέχρι ο αισθητήρας "sensor" να καταγράψει μια τιμή μεγαλύτερη ή ίση με αυτήν της τιμής του "trigger".

Όπως σε όλους τους κώδικες προγραμματισμού δηλώνονται στην αρχή οι βιβλιοθήκες που θα χρησιμοποιηθούν. Αμέσως μετά δηλώνονται οι παράμετροι και οι μεταβλητές. Στην συνέχεια υπάρχει ένα κομμάτι το οποίο δημιουργεί μια εικόνα (512 * 312). Το 512 δημιουργείται ανάλογα με τα βήματα της κάθε διεργασίας και το 312 ανάλογα με τον αριθμό των διεργασιών.

Ο χρήστης όπως αναφέρεται και παραπάνω ανεβάζει έναν αριθμό υποθετικών τιμών για κάθε αισθητήρα. Αυτές τις τιμές τις διαβάζει ο κώδικας και τις προσομοιώνει. Στην συνέχεια διαβάζει τις τιμές που δίνει ο χρήστης στο δεύτερο βήμα του "web site".

Συνοπτικά ο κώδικας αποτελείται από τα παρακάτω τμήματα:

Στο πρώτο τμήμα του προγράμματος δηλώνονται οι παράμετροι της προσομοίωσης.

Δηλαδή :

i. πόσο αργά θα τρέχει η προσομοίωση
ii. πόσα βήματα θα διαρκέσει η κάθε διεργασία
iii. πόσα msec αντιστοιχούν σε κάθε βήμα
iv. πόσες διεργασίες μπορούμε να έχουμε

Στο επόμενο τμήμα του προγράμματος χρησιμοποιώντας κάποιες συναρτήσεις χρόνου της C, δημιουργείται μια αναλογική καθυστέρηση. Κατόπιν προετοιμάζεται η εικόνα εξόδου σε ένα πίνακα .Στην συνέχεια διαβάζονται οι τιμές από τα αρχεία csv του δεύτερου βήματος (step_two) και του πρώτου βήματος (step_one) δηλ. τα φυσικά μεγέθη του "web_site".

Σε αυτό το σημείο διαβάζονται οι τιμές του δείκτη "trigger" και δημιουργούνται τα γεγονότα (events). Δημιουργείται το διάγραμμα χρόνου-διεργασιών και γράφεται σε αρχείο εικόνας τύπου bmp (output.bmp).

Σενάριο 1°

Δημιουργήσαμε διαφορετικά σενάρια (δηλ. με διαφορετικές τιμές στην θέση των triggers) ώστε να καταλάβουμε την πραγματική χρήση αυτής της εφαρμογής και του κώδικα. Ως δειγματοληψία έχουμε κάποιες φυσικές τιμές που θα έχουν ως είσοδο οι αισθητήρες π.χ.:

ΠΙΕΣΗ		ΘΕΡΜΟΚΡΑΣΙΑ		ΥΓΡΑΣΙΑ		ΠΙΕΣΗ ΝΕΡΟΥ	
0.1	14	0.1	24	0.1	15	0.1	15
0.2	30	0.2	36	0.2	20	0.2	20
0.3	42	0.3	40	0.3	30	0.3	30
0.4	56	0.4	52	0.4	65	0.4	65
0.5	62	0.5	55	0.5	90	0.5	90
0.6	68	0.6	67	0.6	85	0.6	85
0.7	75	0.7	75	0.7	96	0.7	96

Πίνακας 7.1

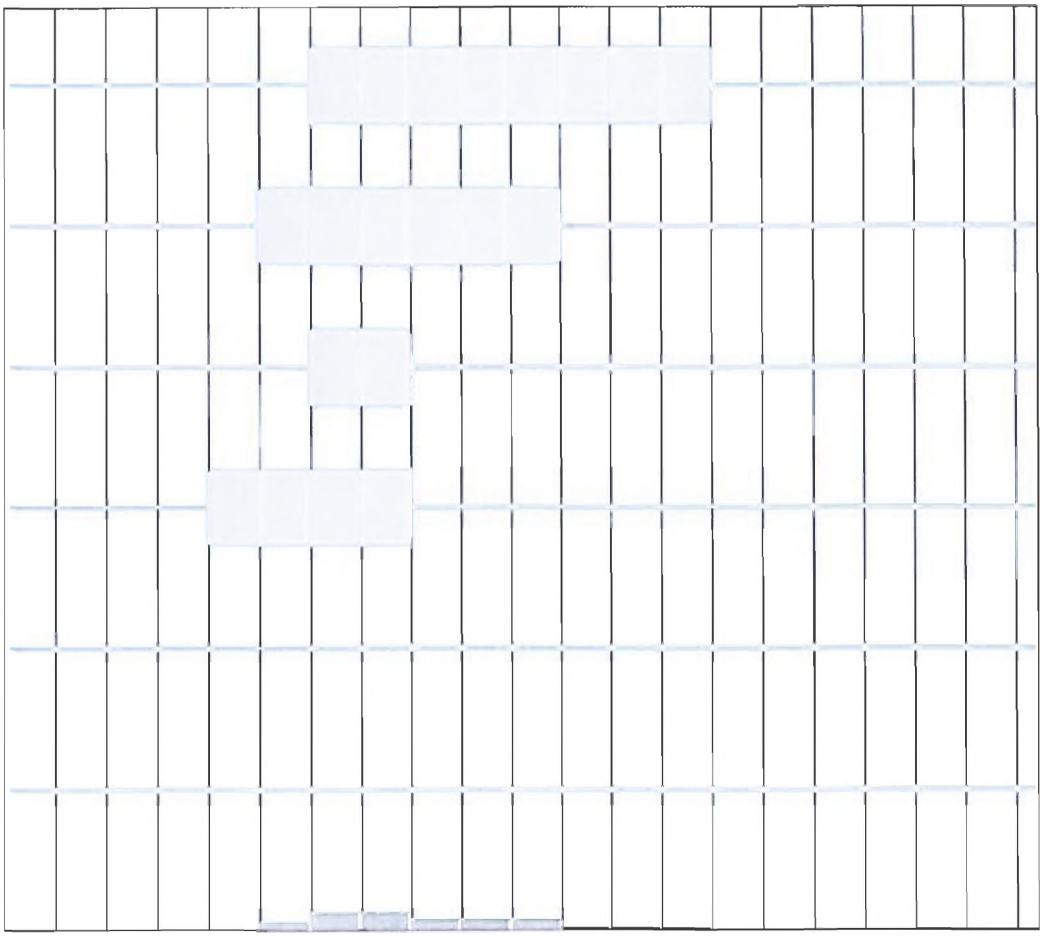
Στην συνέχεια ακολουθούμε το πρώτο βήμα του ιστότοπου και συμπληρώνουμε ως εξής :

```
"firstid","choices","fmegethi","paths","delays"
"1","sensor","piesi","../www/upload/","3"
"2","sensor","piesinerou","../www/upload/","6"
"3","sensor","thermokrasia","../www/upload/","4"
"4","sensor","ygrasia","../www/upload/","7"
```

Κατόπιν συμπληρώνουμε και το δεύτερο βήμα του ιστότοπου:

```
"secondid","names","time","sensors","triggers","steps","priority","actuators","delay","value"
  "1",      "Task1", "3",  "piesi",   "30",      "4",      "1",     "klima",     "2",   "True"
  "2",      "Task2", "3", "ygrasia",  "65",      "2",      "3",   "valvida",     "5",   "True"
  "3",      "Task3", "3","thermokrasia","40",    "6",      "4",     "soba",      "3",   "True"
  "4",      "Task4", "3", "piesinerou", "25",    "8",      "2",     "pump",      "1",   "True"
```

Τρέχουμε την προσομοίωση με το πρόγραμμα της C και παίρνουμε τα αποτελέσματα σε μορφή εικόνας όπως φαίνεται στο παρακάτω σχήμα. (Σχήμα 7.1)



Σχήμα 7.1

Στο παραπάνω σχήμα φαίνονται να εκτελούνται και οι τέσσερις 4 διεργασίες του συστήματος. Οι μπλε κάθετες στήλες δείχνουν το ποσοστό χρήσης της Κεντρικής Μονάδας Επεξεργασίας (CPU) του συστήματος όπως αυτό προσομοιώνεται. Ανάλογα με την τιμή του trigger που έχει δοθεί στην κάθε διεργασία έχουμε παίρνουμε από το σύστημα την ανάλογη χρονική στιγμή που ξεκινάει η κάθε διεργασία. Κάθε κάθετη στήλη συμβολίζει ένα κύκλο ρολογιού της CPU.

Σενάριο 2°

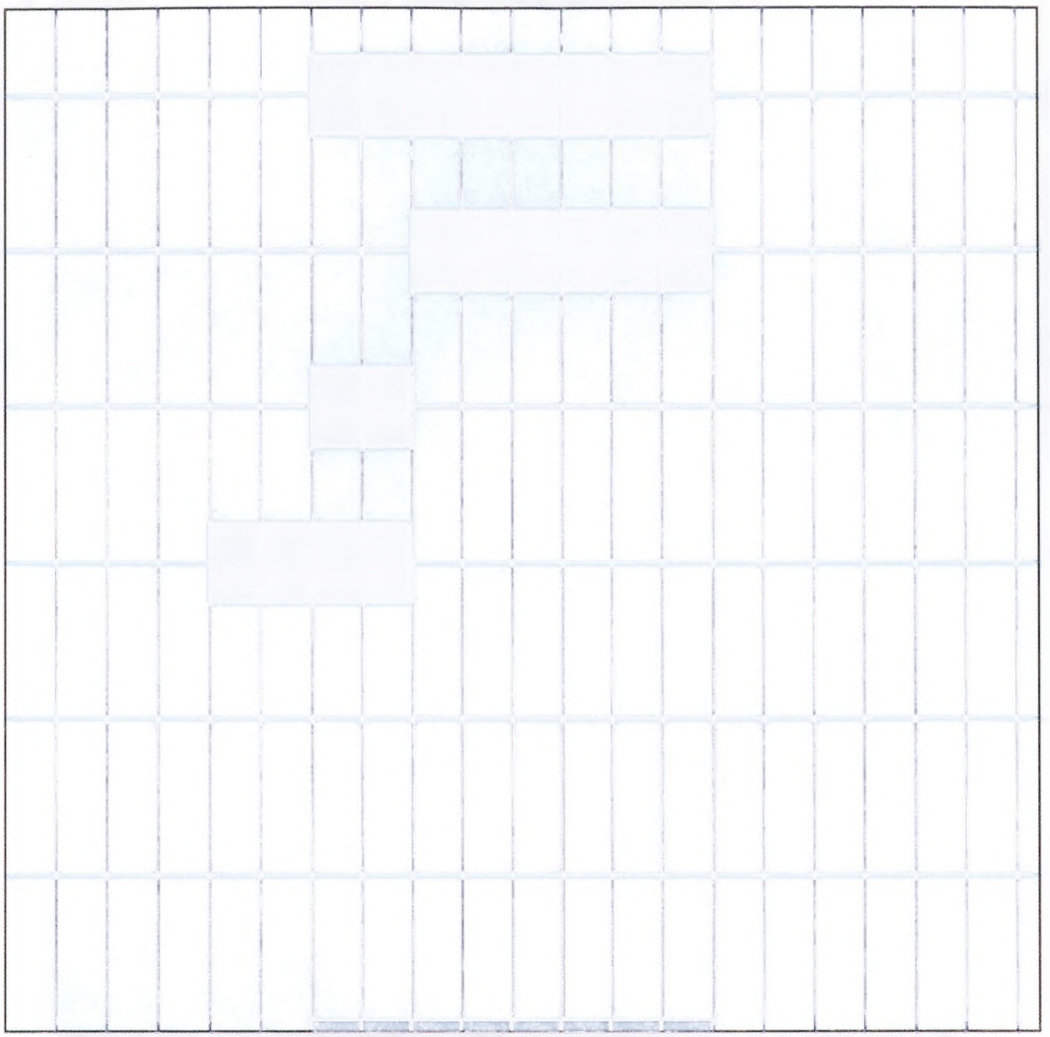
Στο δεύτερο σενάριο θα χρησιμοποιήσουμε διαφορετική δειγματοληψία τιμών σε κάποια διεργασία. Συγκεκριμένα θα δώσουμε άλλες τιμές στον αισθητήρα της θερμοκρασίας . (Πίνακας 7.2)

Ενώ τις άλλα φυσικά μεγέθη θα παραμείνουν ως έχουν. Όπως επίσης το βήμα ένα και δύο του ιστότοπου.

ΠΙΕΣΗ		ΘΕΡΜΟΚΡΑΣΙΑ		ΥΓΡΑΣΙΑ		ΠΙΕΣΗ ΝΕΡΟΥ	
0.1	14	6.1	28	0.1	15	0.1	15
0.2	30	6.2	28	0.2	29	0.2	20
0.3	42	0.3	28	0.3	30	0.3	30
0.4	56	0.4	30	0.4	65	0.4	65
0.5	62	0.5	30	0.5	90	0.5	90
0.6	68	0.6	35	0.6	85	0.6	85
0.7	75	0.7	42	0.7	96	0.7	96

Πίνακας 7.2

Τρέχουμε την προσομοίωση με το πρόγραμμα της C και παίρνουμε τα αποτελέσματα σε μορφή εικόνας όπως φαίνεται στο παρακάτω σχήμα. (Σχήμα 7.2)



Σχήμα 7.2

Στο παραπάνω σχήμα φαίνονται να εκτελούνται και οι τέσσερις 4 διεργασίες του συστήματος. Εδώ βλέπουμε να μετατοπίζεται το σήμα της διεργασίας (θερμοκρασία) προς τα δεξιά. Γιατί σύμφωνα με την νέα δειγματοληψία η διεργασία (θερμοκρασία) ξεκινάει τρεις χρονικές περιόδους πιο μετά.

ΚΕΦΑΛΑΙΟ 8

8.1 ΣΥΜΠΕΡΑΣΜΑΤΑ- ΠΡΟΤΑΣΕΙΣ

Καταλήγοντας δημιουργήσαμε μια διαδικτυακή εφαρμογή , η οποία μας δίνει την δυνατότητα ο κάθε χρήστης να μπορεί να εισάγει τα δικά του δεδομένα τα οποία τα επεξεργάζεται ένας κώδικας γραμμένος σε C++. Στην συνέχεια εξάγει τα αποτελέσματα για τους χρόνους των διεργασιών. Τα αποτελέσματα που απορρέουν από τον κώδικα μας δείχνουν αν εκτελούνται οι διεργασίες μέσα στα χρονικά πλαίσια που θέλει ο κάθε χρήστης. Τα αποτελέσματα παρουσιάζονται σε μορφή διαγράμματος ώστε να είναι πιο εύκολη η επισκόπηση του συστήματος πραγματικού χρόνου.

Η παραπάνω διαδικτυακή εφαρμογή θα μπορούσε να βελτιωθεί δίνοντας την δυνατότητα στον χρήστη να επιλέγει τον αριθμό των φυσικών μεγεθών, που θα προσομοιώνει το σύστημα μας. Κατ' αυτόν τον τρόπο οι ιστοσελίδες μπορεί να είναι δυναμικές και όχι στατικές όπως στην συγκεκριμένη πτυχιακή εργασία. Δηλαδή να διαμορφώνεται ο αριθμός των γραμμών της φόρμας εισαγωγής ανάλογα με τον αριθμό αισθητήρων που καθορίζει ο χρήστης. Κατ' επέκταση θα πρέπει να αναδιαμορφωθεί ο κώδικας καταχώρησης των στοιχείων στην βάση δεδομένων και ο κώδικας προγραμματισμού της προσομοίωσης.

Ένα άλλο σημείο που θα μπορούσαμε να σταθούμε είναι η χρήση μιας γλώσσας προγραμματισμού πιο φιλική σε περιβάλλον Web (π.χ. Java). Όπως είναι γνωστό υπάρχουν διάφοροι περιηγητές που ο καθένας έχει και τους δικούς του περιορισμούς όσον αφορά την εκτέλεση εξωτερικών προγραμμάτων (περιορισμοί ασφαλείας). Στην συγκεκριμένη πτυχιακή εργασία χρησιμοποιήθηκε ο περιηγητής "Internet Explorer", λόγω κάποιον περιορισμών που προέκυψαν από την χρήση της C++ ως γλώσσα προγραμματισμού του προγράμματος προσομοίωσης. Η χρήση μιας άλλης γλώσσας προγραμματισμού όπως αναφέρθηκε πιο πάνω θα μπορούσε να ενσωματώσει τον κώδικα προσομοίωσης μέσα στο περιβάλλον Web (ιστοσελίδες) και όχι να είναι εξωτερικό πρόγραμμα όπως είναι τώρα. Αυτό θα είχε σαν αποτέλεσμα η εφαρμογή μας να είχε χαρακτηριστεί σαν "cross platform" (ανεξάρτητη πλατφόρμα εκτέλεσης).

ΒΙΒΛΙΟΓΡΑΦΙΑ

I. The C++ Programming Language 3e , Bjarne Stroustrup ,
Addison-Wesley , 1997
ISBN: 0-201-88954-4

II. PHP and MySQL Web Development (4th Edition) (Paperback)
Luke Welling (Author), Laura Thomson (Author)

III. Mastering MySQL 4
Ian Gilfillan. Sybex, December 2002

IV. Ανάπτυξη Web Εφαρμογών με PHP και MySQL
Luke Welling, Laura Thomson
Εκδόσεις Μ.Γκιούρδας, Τρίτη Έκδοση
ISBN:960512357-6

V. Simulating Hardware, Software and Electromechanical Parts Using Communicating Simulators.
NIKOS C. PETRELLIS ,ALEXIS N. BIRBAS, MICHAEL K. BIRBAS, EVANGELINOS P. MARIATOS AND GEORGE D. PAPADOPOULOS
University of Patras, Electrical Engineering and Computer Technology Department, Applied Electronics(Laboratory, Rio Campus 26500, Patras Greece-Received November 25, 1996; Revised October 2, 1997)

VI. A Simple Multi-Tasking Simulator
Adriaan de Beer1 Colin Fidge2
1School of Information Technology & Electrical Engineering
2Software Verification Research Centre
The University of Queensland, Australia

VII. STRESS Simulator for Hard Real-time Systems
N. C. Audsley, A. Burns, M. F Richardson and A. J. Wellings Real-Time Systems Research Group, Department of Computer Science, University of York, Heslington, York, YO1 5DD, UK (email: Andy.WellingsKminster.york.ac.uk)

INTERNET LINKS

PHP TUTORIAL

http://www.w3schools.com/php/default.asp

ΣΧΕΤΙΚΑ ΜΕ ΤΗΝ ΔΗΜΙΟΥΡΓΙΑ CSV FILE->

http://www.studiolounge.net/2009/04/03/php-mysql-csv/

ΕΓΚΑΤΑΣΤΑΣΗ ΤΟΥ APACHE,MySQL,PHP->

http://www.wampserver.com/en/

PHP multiple files upload

http://www.phpeasystep.com/phptu/2.html

GLASSY BUTTONS

http://www.glassybuttons.com/buttonmill/glassy.php

UPLOADING FILES TO SERVER

http://www.php-mysql-tutorial.com/wikis/php-tutorial/uploading-files-to-the-server-using-php.aspx

PHP TUTORIAL

http://www.tizag.com/phpT/fileupload.php

PHP TUTORIAL

http://www.htmlgoodies.com/beyond/php/article.php/3472551

Update Table While Insert Data To Another Table

http://www.bigresource.com/PHP-Update-table-while-insert-data-to-another-table-ZBsBaqQA.html

Retrieve Data From a MySQL Database

http://www.php-mysql-tutorial.com/wikis/mysql-tutorials/retrieve-data-from-a-mysql-database.aspx

PHP - File Write: fwrite Function

Create a CSV file from MySQL with PHP

C++ Tutorial

C Programming tutorials

C++ Examples

ΛΟΓΙΣΜΙΚΟ

- Adobe Dreamweaver
- MySQL Database
- PHP 5
- Apache 2 Web Server
- Dev C++ Compiler
- Microsoft Notepad

ΠΑΡΑΡΤΗΜΑ Α

Κώδικας C++

```
#include <stdlib.h>   /* Δηλώσεις των βιβλιοθηκών που θα χρησιμοποιηθούν */
#include <stdio.h>
#include <time.h>
#include <string.h>


#define SLOWFACTOR 100     /* Πόσο αργά θα τρέχει η προσομοίωση */
#define TOTAL_RUN_STEPS 20   /* Πόσα βήματα θα διαρκέσει η προσομοίωση */
#define SIMSTEP 0.1              /* Πόσα millisec θα διαρκέσει το κάθε βήμα */
#define NUMBER_OF_TASKS 4  /*Πόσες διεργασίες μπορούμε να έχουμε */

int timer;

void tic() {
   timer=clock();
}
/*
 Το clock είναι συνάρτηση της C που επιστρέφει πόσα millisec έχουν περάσει από την αρχή.
 Με το timer εδώ παίρνουμε το χρόνο που πέρασε από την προηγούμενη φορά (από το tick).
 Αυτές οι δυο συναρτήσεις έχουν να κάνουν με τη δημιουργία μιας αναλογικής καθυστέρησης
 κι όχι με την ουσία της προσομοίωσης.
 */
int toc() {
```

```
    int tdif;
    tdif = clock()-timer;
    return tdif;
}

unsigned char pix[512][512];  // this is global

int makebmp(char* fnm);


int main(int argc, char** argv) {

    FILE* fo;



    char system_name[100];   /* Όνομα του συστήματος */
    char tmps[100];              /* Προσωρινή μεταβλητή string */
    int task_number;           /* Τρέχων αριθμός διεργασιών */
    int running;               /* Κατά την διάρκεια που τρέχει η προσομοίωση */
    int i;
    int step;                  /* Βήμα του λειτουργικού συστήματος */
    int task;                  /* Τρέχουσα διεργασία */
    int cpu_load;              /* Το ποσοστό φόρτωσης της CPU (1-100) */
    char task_name[NUMBER_OF_TASKS][100];    /* Όνομα της διεργασίας */
    int task_run[NUMBER_OF_TASKS];   /*Αν εκτελείτε η διεργασία */
```

```
int task_load[NUMBER_OF_TASKS];  /*Εκτέλεση της διεργασίας */
float task_start[NUMBER_OF_TASKS]; /*Εδώ ξεκινάει η διεργασία */
float task_clock[NUMBER_OF_TASKS];
int  task_steps[NUMBER_OF_TASKS]; /* Οι διεργασίες χρησιμοποιούν βήματα */
char task_sensor[NUMBER_OF_TASKS][100];  /* Attached sensor */
int task_trigger[NUMBER_OF_TASKS];  /* Επίπεδο trigger */
int task_event[NUMBER_OF_TASKS];  /* Αριθμός γεγονότων */
char task_actuator[NUMBER_OF_TASKS][100]; /*Ενεργοποιητές κινήσεων */
int task_outev[NUMBER_OF_TASKS];  /*Δημιουργεί γεγονότα */
int task_actdelay[NUMBER_OF_TASKS]; /*Καθυστέρηση έναρξης ενεργοποιητή */
char task_actvalue[NUMBER_OF_TASKS][100]; /*Semaphore του ενεργοποιητή */

int event_number; /* Πόσα γεγονότα έχουμε */
char event_name[20][100]; /*Όνομα του γεγονότος */
float event_start[20];      /* Χρόνος γεγονότος */
float event_end[20];  /*Πότε τελειώνει το γεγονός */
int event[20];       /*Ο αριθμός id των γεγονότων */

char cmd;

event_number=0;

   for (i=0;i<100;i++) event[i]=0;

   for (i=0;i<NUMBER_OF_TASKS;i++){
      task_run[i]=0;
```

```
        task_steps[i]=0;
        task_start[i]=0.0;
        task_load[i]=0;
        task_event[i]=-1;
        task_trigger[i]=0;
    }


// Prepare the Image in an array


int x,y;

int onestep = 512/TOTAL_RUN_STEPS;
int taskheight = 312/NUMBER_OF_TASKS;


for (x=0;x<512;x++)
for (y=0;y<512;y++)
{
    if (x%onestep==0)
        pix[x][y]=10; //red
    if (y%taskheight==0)
        pix[x][y]=40; //green
}
```

```
//READ INPUT FILES ----------

int ec;

FILE* eventfile;
FILE* taskfile;

char sa[30]; int sai;
char sb[30];
char sc[30];
char sd[30];
char se[30];
char sf[30];
char sg[30];
char sh[30];
char sj[30];
char sk[30];
```

/* Όλα αυτά είναι προσωρινές μεταβλητές τύπου string για να διαβάζουμε τα αρχεία και στη συνεχεία μεταφέρονται στις ανάλογες μεταβλητές του προγράμματος. */

```
    printf("--------------------------------\n\n");


    if ((taskfile=fopen("export2.csv","rt"))==NULL) {
        printf("ERROR: expected file named EXPORT.CSV !\n");
        system("pause");
        exit(1);
    }


fscanf(taskfile,"%[^,],%[^,],%[^,],%[^,],%[^,],%[^,],%[^,],%[^,],%[^,],%s",sa,sb,sc,sd,se,sf,s
g,sh,sj,sk);
    printf("%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\n",sa,sb,sc,sd,se,sf,sg,sh,sj,sk);


    i=0;
    // read rows


    while(!feof(taskfile))
    {


fscanf(taskfile,"%[^,],%[^,],%[^,],%[^,],%[^,],%[^,],%[^,],%[^,],%[^,],%s",sa,sb,sc,sd,se,sf,s
g,sh,sj,sk);
     printf("%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\t%s\n",sa,sb,sc,sd,se,sf,sg,sh,sj,sk);


     sai = atoi(&sa[2]); //printf(" ##  ID: %s _____ %d\n",sa,sai);
/*
```

Η συνάρτηση atoi μετατρέπει ένα string σε ακέραιο, π.χ. Το "23" το κάνει 23 και μπορούμε να το δώσουμε σε μια μεταβλητή τύπου int.

```
*/
      strcpy(task_name[i],sb);
      task_start[i]=0.0;
      task_clock[i]=(float)atoi(&sc[1]); //printf(" ## CLOCK: %s _____
%f\n",sc,task_clock[i]);
      task_steps[i]=atoi(&sf[1]); //printf(" ## STEPS: %s _____ %d\n",sf,task_steps[i]);
      task_load[i]=atoi(&sg[1]); //printf(" ## LOAD: %s _____ %d\n",sg,task_load[i]);
      strcpy(task_sensor[i],sd);
      task_trigger[i]=atoi(&se[1]);
      strcpy(task_actuator[i],sh);
      task_actdelay[i]=atoi(&sj[1]);
      strcpy(task_actvalue[i],sk);

      task_outev[i]=0;

      for(ec=1;ec<strlen(task_sensor[i])-1;ec++)
        event_name[event_number][ec-1]=task_sensor[i][ec];

      event_name[event_number][ec-1]='\0';

      event_start[event_number]=(float)task_trigger[i];
      event_end[event_number]=0.0;
      printf("\nEVENT[%d]=%s with Trigger at
%f",event_number,event_name[event_number],event_start[event_number]);
      task_event[i]=event_number;
      event_number++;
```

```
printf("\nTask No:%d (id:%d) Name: %s Clock: %2.2f Start: %2.2f\nTrigger when
sensor: %s is at: %d (event %d)\nSteps: %d Priority: %d\nSets actuator: %s after %d at
%s\n",
        i,
        sai,
        task_name[i],
        task_clock[i],
        task_start[i],
        task_sensor[i], // sensor
        task_trigger[i], // trigger
        task_event[i], //event
        task_steps[i], // steps
        task_load[i], // priority
        task_actuator[i], // actuator
        task_actdelay[i], // delay
        task_actvalue[i]  // value
        );
 i++;

 }

 for (task = 0; task <i; task++)  printf("TASK(%d)  STEPS=%d\n",task,task_steps[task]);

 fclose(taskfile);

 printf("------ read now the physical sensors to generfate events ---\n");
```

```
    if ((eventfile=fopen("export.csv","rt"))==NULL) {
    printf("ERROR: expected file named EXPORT.CSV !\n");
    system("pause");
    exit(1);
}

// read header


fscanf(eventfile,"%[^,],%[^,],%[^,],%[^,],%s",sa,sb,sc,sd,se);
printf("%s\t%s\t%s\t%s\t%s\n",sa,sb,sc,sd,se);

// read rows
while(!feof(eventfile))
{
 // read the entry that points to the sensor
 fscanf(eventfile,"%[^,],%[^,],%[^,],%[^,],%s",sa,sb,sc,sd,se);
 printf("%s\t%s\t%s\t%s\t%s\n",sa,sb,sc,sd,se);

 // read the sensor file
 char sfname[100], fi[100], pa[100];
 for (i=1;i<strlen(sc)-1;i++) fi[i-1]=sc[i]; fi[i-1]='\0';
 for (i=1;i<strlen(sd)-1;i++) pa[i-1]=sd[i]; pa[i-1]='\0';
 sprintf(sfname,"%s%s.txt",pa,fi);
 for (i=0;i<strlen(sfname);i++)
    if (sfname[i]=='/') sfname[i]='\\';
```

```
  printf("File Name: %s\n",sfname);


   FILE *sf;
/*
   Ορίζουμε ένα Αρχείο (ένα αρχείο για τη C είναι ένας δείκτης
σε μια μεταβλητή τύπου FILE)
*/
   if ((sf=fopen(sfname,"rt"))==NULL)
   {
      printf("Cannot open file!\n");


  system("pause");
      exit(1);
   }


   float f;
   int d;


   while(!feof(sf))
   {
      fscanf(sf,"%f %d",&f,&d);
      //printf("              At time: %f  Sensor gives: %d --- ",f,d);
      for(ec=0;ec<event_number+1;ec++)
      {
         if (strcmp(fi,event_name[ec])==0   )
         {
```

```
        //printf("found it as event trigger: %d\n",ec);
        if (((float)d>event_start[ec])&&(event_end[ec]==0.0))
        {
           printf("  At time: %f  Sensor gives: %d --- ",f,d);
           printf("BINGO! ");
           event_start[ec]=f;  printf("Start of Task: %d  at %f\n",ec,event_start[ec]);
           event_end[ec]=999.999;
        }
      }
   }
}

fclose(sf);
}

fclose(eventfile);


printf("Now we have to prepare events for the tasks....\n");


// ------------------------------------

float simtime;
```

```
running=0;
step=0;


// SAVE RESULTS:
fo=fopen("OUT.TXT","wt");

// Application Loop: Runs until end of simulation time
// ------------------------------------------------------------------
// ------------------------------------------------------------------
while(running==0) {


tic(); while(toc()<(int)(SLOWFACTOR*SIMSTEP)); // Create a delay to make it look
real...
step++; if (step>TOTAL_RUN_STEPS) running=1;

simtime= (float)step*SIMSTEP;

cpu_load=0;

//system("pause"); //step-by-step

printf("SIMTIME=%2.2f STEP=%3d :",simtime,step);
fprintf(fo,"Time: %02.03f  Step: %04d",simtime,step);
```

```
for (i=0;i<event_number;i++)
{
   if ((event_start[i]<simtime))
   {
      event[i]=1;
   }
   if (event[i]>0) printf("#");
   else printf("-");
}
printf(" ");


for (task=0;task<NUMBER_OF_TASKS;task++)
{

   if (
         ((event[task_event[task]]==1)||(task_event[task]==-1))
         &&(task_start[task]<simtime)
         &&(task_steps[task]>0)
         &&(task_load[task]+cpu_load<100)
     )
   {
      task_run[task]=task_steps[task];
      task_steps[task]--;
      if ((task_steps[task]==0)&&(task_outev[task]>0))
      {
```

```
        event[task_outev[task]-1]=1;
      }
      cpu_load=cpu_load+task_load[task];

   }
   else
   {
      task_run[task]=0;
   }

   printf(" %3d ",task_run[task]);
   fprintf(fo," %03d ",task_run[task]);

   for (x=step*onestep;x<(step+1)*onestep-1;x++)
   for (y=(task+3)*taskheight-20;y<(task+3)*taskheight+20;y++)
   {
      if (task_run[task]>0)
      pix[x][y]=20; //green
   }
}

printf(" CPU_Load=%4d\n",cpu_load);
fprintf(fo," CPU: %03d \n",cpu_load);
```

```
      for (x=step*onestep;x<(step+1)*onestep-1;x++)
      for (y=1;y<110;y++)
      {
        if (y<cpu_load)
            pix[x][y]=50; //green
      }


      } // end of application loop
      // ---------------------------------------------------------------------
      // ---------------------------------------------------------------------

      fclose(fo);


   makebmp("input.bmp");


   //system("pause");
   return (EXIT_SUCCESS);
}


int makebmp(char* fnm)
{
```

```
// Write the file

      char fname[20];

strcpy(fname,fnm);

 FILE *fp;
 FILE *fo;

 unsigned long cnt,im_row,im_col;
 unsigned char tc,tr,tg,tb;
 unsigned int w1,w2,width,h1,h2,height,offset,bpp;
 unsigned int padd,i;

       unsigned char tmpim[1228800];


if (  !(fo=fopen("output.bmp","wb")) ) {
      printf("Cannot create output file...\n");
      system("pause");
};

 if ( !(fp=fopen(fname,"rb"))) {
   printf("Unable to load BMP file: %s...\n",fname);
   return(1);
 } else {
```

```
cnt = 0;
im_row=0;
im_col=0;
    while (!feof(fp)) {
        tc = fgetc(fp) & 0x0ff;

   fputc(tc,fo);

   if (cnt==10) offset=tc;
   if (cnt==18) w1=tc;
   if (cnt==19) w2=tc;
   if (cnt==22) h1=tc;
   if (cnt==23) h2=tc;
   if ((cnt==30)&&(tc>0)) printf("ERROR!! Cannot Handle BMP files with
Compression!!!\n");
   if (cnt==28) bpp=tc;
   if (cnt==53) break;

   cnt++;
}

width  = w1+w2*256;
height = h1+h2*256;
padd = 0;
//printf("W:%d, H:%d   Image Data Starts at:%d with %d bits/pixel
\n",width,height,offset,bpp);
```

```
for (im_row=0;im_row<height;im_row++)
{
   for (im_col=0;im_col<width;im_col++)
   {
      tr = fgetc(fp) & 0x0ff;
      tmpim[(im_col*height+height-im_row-1)*3+2] = tr;

      //if ((im_col%32==0)&&(im_row%32==0)) printf("%02u",tc/10);

      tg = fgetc(fp) & 0x0ff;
      tmpim[(im_col*height+height-im_row-1)*3+1] = tg;

      //if ((im_col%32==0)&&(im_row%32==0)) printf("%02u",tc/10);

      tb = fgetc(fp) & 0x0ff;
      tmpim[(im_col*height+height-im_row-1)*3+0] = tb;

      //if ((im_col%32==0)&&(im_row%32==0)) printf("%02u-",tc/10);

      tb = 0; tr = 0; tg = 0;

      if (pix[im_col][im_row]>0)
      {

       if (pix[im_col][im_row]%10==0)
       {
```

```
      tr = 255; pix[im_col][im_row] = pix[im_col][im_row]-10;
    }
    if (pix[im_col][im_row]%20==0)
    {
      tg = 255; pix[im_col][im_row] = pix[im_col][im_row]-20;
    }
    if (pix[im_col][im_row]==40)
    {
      tb = 255;
    }
    }

    tr = 255-tr;
    tb = 255-tb;
    tg = 255-tg;

    fputc(tb,fo);
    fputc(tg,fo);
    fputc(tr,fo);



}
//if (im_row%32==0) printf("\n");

for (i=0;i<padd;i++) {
```

```
        tc = fgetc(fp) & 0x0ff;
        fputc(tc,fo);
        }
           }
    }

        fclose(fp);

    fclose(fo);

    return(0);

}
```