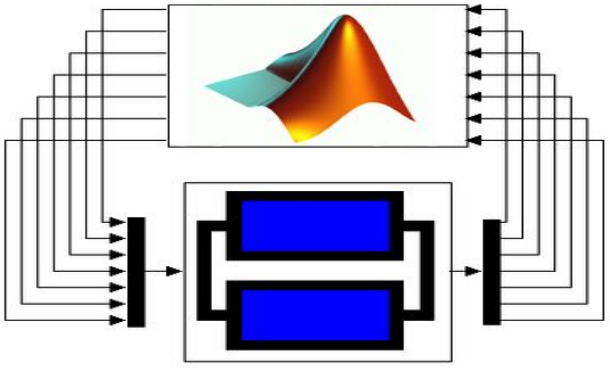
ΑΕΙ ΠΕΙΡΑΙΑ Τ.Τ.
ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΦΑΡΜΟΓΩΝ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ ΗΛΕΚΤΡΟΝΙΚΩΝ
ΥΠΟΛΟΓΙΣΤΙΚΩΝ ΣΥΣΤΗΜΑΤΩΝ Τ.Ε.

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΕΓΧΕΙΡΙΔΙΟ DYMOLA (MODELICA) vs. MATLAB

ΣΟΦΙΑ Γ. ΛΟΥΜΙΩΤΗ

ΕΙΣΗΓΗΤΡΙΑ : ΑΝΑΣΤΑΣΙΑ Ν. ΒΕΛΩΝΗ, ΚΑΘΗΓΗΤΡΙΑ
ΗΛΕΚΤΡΟΛΟΓΟΣ ΜΗΧ/ΚΟΣ Ε.Μ.Π.




Dymola

ΠΤΥΧΙΑΚΗ ΕΡΓΑΣΙΑ

ΕΓΧΕΙΡΙΔΙΟ DYMOLA (MODELICA) vs. MATLAB

ΣΟΦΙΑ Γ. ΛΟΥΜΙΩΤΗ

Α.Μ. 40816

ΕΙΣΗΓΗΤΡΙΑ :

ΑΝΑΣΤΑΣΙΑ Ν. ΒΕΛΩΝΗ

ΕΞΕΤΑΣΤΙΚΗ ΕΠΙΤΡΟΠΗ :

ΗΜΕΡΟΜΗΝΙΑ ΕΞΕΤΑΣΗΣ :

ΔΗΛΩΣΗ ΣΥΓΓΡΑΦΕΑ ΠΤΥΧΙΑΚΗΣ ΕΡΓΑΣΙΑΣ

Ο/Η κάτωθι υπογεγραμμένος/η ……...
του ……, με αριθμό μητρώου ……. φοιτητής/τρια του Τμήματος Μηχανικών Η/Υ Συστημάτων Τ.Ε. του Α.Ε.Ι Πειραιά Τ.Τ. πριν αναλάβω την εκπόνηση της Πτυχιακής Εργασίας μου, δηλώνω ότι ενημερώθηκα για τα παρακώτω :

«Η Πτυχιακή Εργασία (Π.Ε.) αποτελεί προϊόν πνευματικής ιδιοκτησίας τόσο του συγγραφέα, όσο και του Ιδρύματος και θα πρέπει να έχει μοναδικό χαρακτήρα και προτότυπο περιεχόμενο.

Απαγορεύεται αυστηρά οποιοδήποτε κομμάτι κειμένου της να εμφανίζεται αυτούσιο ή μεταφρασμένο από κάποια άλλη δημοσιευμένη πηγή. Κάθε τέτοια πράξη αποτελεί προϊόν λογοκλοπής και εγείρη θέμα Ηθικής Τάξης για τα πνευματικά δικαιώματα του άλλου συγγραφέα. Αποκλειστικός υπεύθυνος είναι ο συγγραφέας της Π.Ε., ο οποίος φέρει και την ευθύνη των συνεπειών, ποινικών και άλλων, αυτής της πράξης.

πέραν των όποιων ποινικών ευθυνών του συγγραφέα σε περίπτωση που το Ίδρυμα του έχει απονείμει Πτυχίο, αυτό ανακαλείται με απόφαση της Συνέλευσης του

Τμήματος. Η Συνέλευση του τμήματος με νέα αποφασή της, μετά από αίτηση του ενδιαφερομένου, του αναθέτει εκ νέου την εκπόνηση της Π.Ε. με άλλο θέμα και διαφορετικό επιβλέποντα καθηγητή. Η εκπόνηση της εν λόγω Π.Ε. πρέπει να ολοκληρωθεί εντός τουλάχιστον ενός ημερολογιακού 6μήνου από την ημερομηνία ανάθεσης της. Κατά τα λοιπά εφαρμόζονται τα προβλεπόμενα στο άρθρο 18, παρ. 5 του ισχύοντος Εσωτερικού Κανονισμού.»

ΕΥΧΑΡΙΣΤΙΕΣ

Η παρούσα πτυχιακή εργασία ολοκληρώθηκε μετά από πολύ δουλειά και αποτελεί το τελευταίο κομμάτι της ολοκλήρωσης των σπουδών μου στο τμήμα Μηχανικών Ηλεκτρονικών Υπολογιστικών Συστημάτων του ΑΕΙ Πειραιά Τ.Τ.

Την δουλειά και προσπάθειά μου υποστήριξε και καθοδήγησε η επιβλέπων καθηγήτρια κα. Αναστασία Βελώνη την οποία θα ήθελα να ευχαριστήσω θερμά.

Επίσης θα ήθελα να ευχαριστήσω ιδιαίτερα την οικογένειά μου για την υποστήριξη της κατά την διάρκεια των σπουδών μου.

ΠΕΡΙΛΗΨΗ

Η παρούσα πτυχιακή ασχολείται με την παρουσίαση του επιχειρισιακού μοντέλου προσομοίωσης DYMOLA, καθώς και με την αλληλεπιδρασή του με το περιβάλλον του MatLab.

Στις παρακάτω σελίδες περιγράφονται οι τρόποι για τη δημιουργία και

προσομοίωση ολοκληρωμένων μηχανικών, μηχανολογικών, υδραυλικών κ.α. προτύπων με την χρήση του DYMOLA. Αν και τα παραδείγματά μας είναι απλά, βοηθούν στην κατανόηση της χρησιμότητας του εργαλείου και την ευκολία που παρέχει στην υλοποιήση τόσο αυτών όσο και πιο πολύπλοκων συστημάτων. Αυτό υλοποιείται μέσω της πλούσιας βιβλιοθήκης δομικών στοιχείων που διαθέτει το DYMOLA και της δυνατότητας καταχώρησης του απαιτούμενου κώδικα σε κλάσεις της γλώσσας Modelica, στην οποία είναι βασισμένο το εργαλείο.

Ακόμα υπάρχει εκτενής αναφορά στο περιβάλλον του MatLab και ο τρόπος που αλληλεπιδρά με το DYMOLA. Γίνονται σαφής οι ομοιότητες, οι διαφορές αλλά και ο τρόπος που διαχειρίζεται το καθ' ένα τις εργασίες του.

Τέλος, υπάρχουν απλά παραδείγματα χρήσης του MatLab, αλλά και πιο πολύπλοκα που δημιουργούνται με την βοήθεια των εργαλείων που διαθέτει με τα οποία εξηγείται γιατί είναι πιο αποτελεσματικό σε σχέση με το DYMOLA σε συγκεκριμένες περιπτώσεις.

ABSTRACT

The present thesis concerns the presentation of the commercial modeling and simulation tool DYMOLA, as well its interaction with MatLab environment.

The following pages describe ways of modeling and simulate integrated mechanic, mechanical, hydraulic and etc. modelsusing DYMOLA. Allthought the examples described here are simple, they contribute on comprehension the tool's usage and convenience for modeling these kind of systems. DYMOLA's vast library of stractured components and insertion of Modelica programming code on classes are the main reason for the above.

Moreover, there is lenthly reference on MatLab environment and how it interacts with DYMOLA. Similarities, differences and the way each of them manage their tasks are being explained.

Finally, simple MatLab examples are described giving reader a more detailed view and also more complicated ones show the use of ToolBox Kit MatLab provides and why it can be more effective in certain things than DYMOLA.

ΠΕΡΙΕΧΟΜΕΝΑ

ΠΙΝΑΚΑΣ ΕΙΚΟΝΩΝ

ΚΑΤΑΛΟΓΟΣ ΠΙΝΑΚΩΝ

ΚΕΦΑΛΑΙΟ 1

1.1 ΕΙΣΑΓΩΓΗ

1.1.1 Περιγραφή του αντικειμένου της πτυχιακής εργασίας

Σκοπός αυτής της πτυχιακής είναι να εντάξει τον αναγνώστη στην χρήση του εργαλείου DYMOLA το οποίο χρησιμοποιείται σε διάφορους κλάδους επαγγελμάτων και ειδικότερα την σημασία χρήσης του από έναν Μηχανικό Η/Υ.

Το DYMOLA είναι ένα πρόγραμμα προσομοίωσης με το οποίο μπορούμε να δημιουργήσουμε μηχανικά, ηλεκτρικά, κ.α., μοντέλα συστημάτων πολύ γρήγορα και πολύ εύκολα και κατόπιν να τα προσομοιώσουμε σε real – time περιβάλλον. Ο λόγος που το DYMOLA μπορεί να τα δημιουργήσει είναι οι πλούσιες βιβλιοθήκες με δομικά στοιχεία, σε γλώσσα Modelica, που διαθέτει και ουσιαστικά το μόνο που έχει να κάνει ο χρήστης είναι να εισάγει τα απαραίτητα εικονίδια για την δημιουργία του προτύπου.

Χαρις στην ευελιξία που προσφέρει, το DYMOLA μπορεί να δημιουργήσει από πολύ εύκολα συστήματα, για παράδειγμα ένα σύστημα ανάδρασης, μέχρι πολύπλοκα και δύσκολα συστήματα όπως το σύστημα ενός αεροσκάφους. Επίσης το πρόγραμμα περιλαμβάνει έτοιμα συστήματα τα οποία μπορούν να διδάξουν και να βοηθήσουν στην περαιτέρω κατανόηση του όλου πράγματος του χρήστη. Έτσι το μόνο που χρειάζεται κάποιος είναι οι γνώσεις μηχανικού που ήδη διαθέτει και όχι κάτι πολύλποκο. Στις παρακάτω σελίδες δημιουργούμε βήμα – βήμα εύκολα μοντέλα – πρότυπα, αλλά και χρησιμοποιούμε τα υπάρχοντα πολύπλοκα συστήματα για παραπάνω εξοικίωση με το πρόγραμμα. Ακόμα γίνεται εκτενής αναφορά στη γλώσσα προγραμματισμού Modelica στην οποία είναι βασιμένο και χτισμένο το πρόγραμμα.

Επίσης σ’ αυτην την εργασία ασχολούμαστε και με το περιβάλλον εργασίας του MatLab και πως αυτό αλληλεπιδρά με το DYMOLA. Παρουσιάζονται ομοιότητες και διαφορές μεταξύ των δύο εργαλείων και γίνεται αναφορά στο πως αυτά μπορούν να συνδιαστουν και σε ποιους τομείς γίνεται καλύτερη χρήση του ενός και το άλλου.

Τέλος, παρουσιάζεται εκτενώς το περιββάλον εργασίας τοόσο του DYMOLA όσο και του MatLab. Όπως το DYMOLA έτσι και για το MatLab δημιουργούμε απλά παραδείγματα και με την βοήθεια των εργαλείοθηκών που αυτό μας παρέχει δημιουργούμε πιο περίπλοκα συστήματα και εξηγούμε γιατί είναι πιο αποτελεσματικά με την χρήση του MatLab σε σχέση με το DYMOLA.

ΚΕΦΑΛΑΙΟ 2

2.1. MODELICA LANGUAGE

2.1.1 ΕΙΣΑΓΩΓΗ:

«An object – oriented, equation based language to conveniently model complex physical systems containing mechanical, electrical, electronical, hydraulic, thermal, control and electrical power or process – oriented subcomponents».

Είναι μια αντικειμενοστραφής (object – oriented), βασισμένη σε εξισώσεις (equation based) γλώσσα που βοηθά στη μοντελοποίηση πολύπλοκων φυσικών συστημάτων, βασισμένων σε ¨στοιχεία¨ σχεδίασης (component – oriented modeling) που μπορούν να περιέχουν μηχανικά (mechanical), ηλεκτρικά (electrical), ηλεκτρονικά (electronical), υδραυλικά (hydraulic), θερμικά (thermal) υποστοιχεία, καθώς επίσης και υποστοιχεία ελέγχου (control) και ηλεκτρικής ενέργειας (electrical power) ή επίσης τεχνικά υποστοιχεία (process – oriented subcomponents).

2.1.2. ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Η γλώσσα δημιουργήθηκε το 1996 από επιστήμονες του χώρου της Πληροφορικής και της Μηχανικής. Από τότε συνεχίζει να εξελείσεται και η χρήση της αυξάνεται όλο και περισσότερο στο χώρο της βιομηχανίας για την σχεδίαση και ανάπτυξη μοντέλων. Πολλές εταιρίες κολλοσοί, όπως AUDI, BMW, SIEMENS κ.τ.λ, χρησιμοποιούν την MODELICA για την ανάπτυξη και κατασκευή οχημάτων επαρκούς ενέργειας, την βελτίωση των συστημάτων air-conditioning καθώς και για την παροχή εργοστασίων ενέργειας αντίστοιχα.

Υπάρχουν πολλά περιβάλλοντα προσομοίωσης βιομηχανικών ή μη αναγκών που χρησιμοποιούν την MODELICA όπως CATIA Systems, CyModelica, DYMOLA, OpenModelicaEdit κ.α.

Τα μοντέλα που δημιουργούνται με την χρήση της MODELICA μπορούν εύκολα να μετατραπούν σε μοντέλα του περιβάλλοντος SIMULINK χρησιμοποιώντας στοιχεία εξαγωγής του DYMOLA, MampleSim κ.α.. Σ’ αυτήν την πτυχιακή εργασία θα

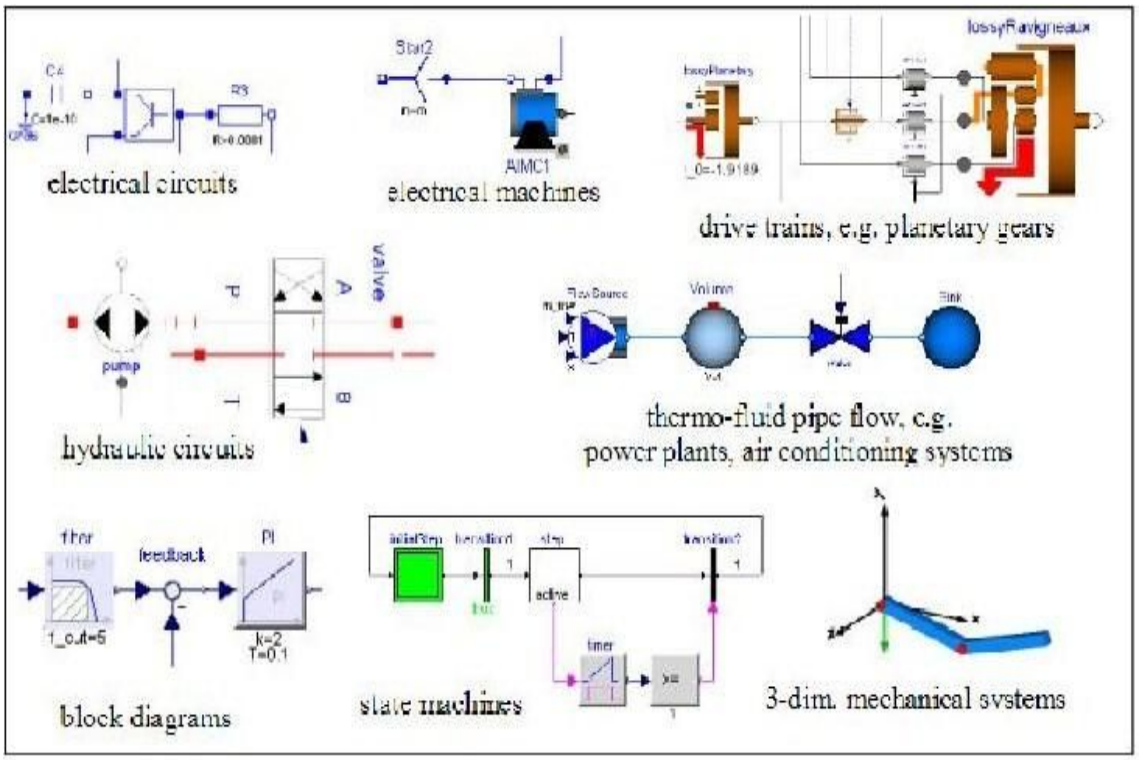
ασχοληθούμε με το περιβάλλον προσομοίωσης DYMOLA καθώς και την σύνδεση του με το MatLab.

2.1.3 ΓΕΝΙΚΑ:

Η Modelica διαθέτει βιβλιοθήκες πλούσιες από σετ μοντέλων και προτύπων έτοιμα προς χρήση. Είναι μια αντικειμενοστραφής γλώσσα μοντελοποίησης μεγάλων, πολύπλοκων και ετερογενών φυσικών συστημάτων που περιγράφει μαθηματικά μοντέλα μέσω αλγεβρικών και διαφορικών εξισώσεων. Οι εξισώσεις αυτές μπορούν να χρησιμοποιηθούν για τη διαμόρφωση των φυσικών φαινομένων.

Από την μεριά του χρήστη τα μοντέλα περιγράφονται με σχηματικά διαγράμματα (schematics) τα οποία ονομάζονται επίσης και διαγράμματα αντικειμένων (object diagrams) ή block διαγράμματα. Εμείς θ' αναφερόμαστε σ' αυτά ως block διαγράμματα. Τα διαγράμματα αυτά σχεδιάζονται με τη βοήθεια των διάφορων δομικών στοιχείων που βρίσκονται σε διαφορετικές κατηγορίες μέσα στις υπάρχουσες βιβλιοθήκες της Modelica.

Η δημιουργία και προσομοίωση ενός προτύπου είναι το σημαντικότερο κομμάτι για την υλοποίηση του πραγματικού μοντέλου. Για παράδειγμα θα πρέπει πρώτα να σχεδιαστούν τα τυποποιημένα δομικά στοιχεία (μηχανές, βαλβίδες, εξαρτήματα) με τις κατάλληλες προδιαγραφές. Ακριβώς η ίδια διαδικασία πρέπει να γίνει και για την υλοποίηση του φυσικού μοντέλου στον «πραγματικό» κόσμο. Γι' αυτό το λόγο είναι και το δυσκολότερο κομμάτι στο οποίο πρέπει να εργαστεί ένας μηχανικός. Όπως είναι φυσικό, για να είναι αποδοτικότερος στο τρόπο δουλείας του, πέρα από τις απαραίτητες βασικές γνώσεις του αντικειμένου το, θα πρέπει να έχει καλή γνώση των βιβλιοθηκών του προγράμματος, αλλά και να μπορεί να τις χειρίζεται με τον σωστό τρόπο.



Εικόνα 1. Παράδειγματα δομικών στοιχείων σε block διάγραμμα. Ένα σχηματικό διάγραμμα περιλαμβάνει συνδεόμενα στοιχεία όπως μια αντίσταση ή ένα υδραυλικό κύλινδρο. Ένα στοιχείο έχει ¨συνδετήρες¨ (connectors) – λέγονται και πόρτες (ports) - οι οποίοι περιγράφουν περιπτώσεις αλληλεπίδρασης, π.χ. ένα σήμα εισόδου (input signal).

Η Modelica είναι μια γλώσσα περιγραφής κειμένου που καθορίζει όλα τα μέρη ενός μοντέλου και χτίζει στοιχεία του μοντέλου σε βιβλιοθήκες, τα οποία καλούνται πακέτα (packages). Είναι μια μοντέρνα γλώσσα, χτισμένη σε μη αιτιολογική μοντελοποίηση με μαθηματικές εξισώσεις και αντικειμενοστραφή σχεδίαση. Με λίγα λόγια τα μοντέλα που χτίζονται με τη χρήση της μπορούν να χρησιμοποιηθούν ξανά και ξανά.

Τα σχηματικά διαγράμματα που χρησιμοποιούνται από τη Modelica για την δημιουργία προτύπων, αποτελούνται από συγκεκριμένους από κανόνες. Αυτοί οι κανόνες χρησιμοποιούνται για την μετάφραση οποιασδήποτε κλάσης σε μια ενιαία δομή.

Μια κλάση θα πρέπει να έχει επιπλέον ιδιότητες έτσι να παράγει ώστε ένα set από διαφορικές εξισώσεις (flat hybrid DAE). Τέτοιες κλάσεις ονομάζονται μοντέλα προσομοίωσης.

Η ενιαία δομή της Modelica καθορίζεται επίσης και γι' άλλες περιπτώσεις εκτός από μοντέλα προσομοίωσης όπως:

- περιλαμβάνει συναρτήσεις – έτσι μπορεί να χρησιμοποιηθεί ώστε να παρέχει αλγοριθμικά περιεχόμενα.
- περιλαμβάνει πακέτα – χρησιμοποιούνται ως μηχανική δομή.
- και μερικά μοντέλα – χρησιμοποιούνται ως μοντέλα βάσης (demos).

Τα παραπάνω επιτρέπουν τον καθορισμό της ορθότητας πριν το «κτίσιμο» του μοντέλου προσομοίωσης.

Η γλώσσα σχεδιάστηκε ώστε να διευκολύνει τον συμβολικό μετασχηματισμό των μοντέλων ως συνεχή ή στιγμιαία ενιαία δομή. Η MODELICA δεν καθορίζει πως προσομοιώνεται ένα μοντέλο. Καθορίζει όμως, το set των εξισώσεων του οποίου το αποτέλεσμα της προσομοίωσης θα πρέπει να ικανοποιεί όσο το δυνατόν καλύτερα. Υποστηρίζει διάφορους τύπους εξισώσεων, όπως κανονικές εξισώσεις, διαφορικές αλγεβρικές εξισώσεις (DAE) καθώς και γραφικές εξισώσεις (SIMULATION).

Στόχος της γλώσσας είναι να δημιουργεί ρεαλιστικά πρότυπα που θα χρησιμοποιηθούν στη δημιουργία πραγματικών συστημάτων. Γι’ αυτό το λόγο είναι δομημένη ώστε να επιτρέπει στα εργαλεία (δομικά στοιχεία) να παράγουν έναν αποδοτικό κώδικα χωρίς να χρειάζονται συνεχείς χειρωνακτικοί έλεγχοι για των εντοπισμό λαθών.

ΚΕΦΑΛΑΙΟ 3:

3.1. DYMOLA

3.2 ΕΙΣΑΓΩΓΗ:

Το DYMOLA είναι ένα πρόγραμμα προσομοίωσης (commercial modeling = επιχειρησιακό μοντέλο), το οποίο θεωρείται βασικό εργαλείο γι' ένα μηχανικό καθώς μπορεί να δημιουργεί πρότυπα χρησιμοποιώντας τις βιβλιοθήκες του – βασισμένες στη γλώσσα Modelica – πλούσιες σε δομικά στοιχεία.

Μεγαλύτερο πλεονέκτημα του, είναι η αυτόματη καταχώρηση του κώδικα (ιεραρχικά δομημένες κλάσεις σε γλώσσα Modelica) πίσω από τα δομικά στοιχεία που εισάγουμε από την βιβλιοθήκη για την δημιουργία του προτύπου μας. Επομένως είναι αρκετές οι μηχανολογικές γνώσεις και η εξοικείωση με τον browser τη βιβλιοθήκης για την δημιουργία του προτύπου. Ένα πρότυπο που δημιουργείται κάθε φορά απαρτίζεται από δομικά στοιχεία/εικονίδια, όπου συνδέονται μεταξύ τους και αποτελούν ένα λεπτομερές σύνολο, από μηχανολογικής, ηλεκτρολογικής, υδραυλικής άποψης ή και ένα σύνολο όλων αυτών.

Το DYMOLA είναι ένα φιλικό για τον χρήστη πρόγραμμα, το οποίο του δίνει την δυνατότητα ν' ασχολείται περισσότερο με την ουσία της δουλείας του και την βελτιστοποίηση των αποτελεσμάτων του, αποφεύγοντας χρονοβόρες και κουραστικές διαδικασίες όπως η συγγραφή πολύπλοκου και πολλες φορες μακροσκελή κώδικα.

3.2.1 ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ – ΓΕΝΙΚΑ:

Το DYMOLA πρωτοξεκίνησε το 1978 όταν ο Hilding Elmqvist παρουσίασε τη διδακτορική του εργασία πάνω σε μεθόδους μοντελοποίησης βασισμένες σε στοιχεία καθώς και προσομοίωση με την χρήση μη – αιτιατών εξισώσεων και «χειραγώγηση»

συμβολικών αυτόματων συστημάτων (automatic symbolic manipulation), σε μια φόρμα κατάστασης χώρου (state space form) κατάλληλη για αριθμητική επίλυση.

Στην εργασία του περιέγραφε μια γλώσσα μοντελοποίησης, την MODELICA(DYMOLA), όμως λόγω των περιορισμών στις δυνατότητες των υπολογιστών εκείνη την εποχή (μεγάλη κατανάλωση ενέργειας, μικρή χωρητικότητα μνήμης) δεν ήταν δυνατόν να χρησιμοποιηθεί σε μεγαλύτερη κλίμακα και ειδικότερα σε βιομηχανικό επίπεδο.

Με τα χρόνια οι δυνατότητες των υπολογιστών αυξήθηκαν, δίνοντας την δυνατότητα ανάπτυξης του DYMOLA. Το 1992 ο Elmqvist ίδρυσε την εταιρία DYNASIM. Μια καινούργια και πιο ανεπτυγμένη εκδοχή της γλώσσας αλλά και εργαλείων του DYMOLA έκανε την εμφάνιση της, παρέχοντας την ευκαιρία ανάπτυξης αντικειμενοστραφής προσέγγισης με την υποστήριξη διακριτών και υβριδικών προσομοιώσεων.

Όταν η σχεδίαση της γλώσσας προσομοίωσης MODELICA έφτασε σε ένα ικανοποιητικό επίπεδο, χρησιμοποιήθηκε για να ενισχύσει τα εργαλεία του DYMOLA και με την πάροδο των χρόνων έγινε η κυρίαρχη γλώσσα αυτού.

3.3. ΧΑΡΑΚΤΗΡΙΣΤΙΚΑ ΤΟΥ DYMOLA:

Το DYMOLA είναι ένα δυναμικό πρόγραμμα προσομοίωσης, κατάλληλο για την επεξεργασία διαφόρων ειδών φυσικών συστημάτων. Χρησιμοποιεί βιβλιοθήκες που υποστηρίζουν μεθοδολογία διαμόρφωσης βασισμένη στον προσανατολισμό και τις εξισώσεις του αντικειμένου.

Τα κύρια χαρακτηριστικά του DYMOLA είναι:

- Ο χειρισμός των σύνθετων προτύπων κατασκευής
- Η γρήγορη διαμόρφωση και προσομοίωση
- Καθορισμένα πρότυπα εργαλεία
- Επαφή με εργαλεία
- Τρισδιάστατη εικόνα σε πραγματικό χρόνο προσομοίωσης

3.4. Η ΑΡΧΙΤΕΚΤΟΝΙΚΗ ΤΟΥ DYMOLA

Το περιβάλλον εργασίας του DYMOLA έχει έναν ισχυρό γραφικό συντάκτη, για τη σύνθεση των προτύπων και όπως έχουμε ήδη αναφέρει και παραπάνω είναι βασισμένο στη χρήση της γλώσσας MODELICA. Επίσης, μπορεί να εισαγάγει κι άλλα αρχεία στοιχείων και γραφικές παραστάσεις. Ακόμα, περιέχει έναν συμβολικό μεταφραστή – για τις εξισώσεις του MODELICA - που παράγουν τον c κώδικα για την προσομοίωση.

Γενικά η δομή του DYMOLA ήταν παρεμφερής με ήδη προϋπάρχοντα περιβάλλοντα (OpenModelica, MathModelica) και αποτελείται από τα παρακάτω κύρια μέρη:

- Έναν μεταγλωττιστή (Modelica Compiler) για την βελτιστοποίηση και την μείωση του μεγέθους των εξισώσεων του συστήματος.
- Ένα σύστημα προσομοίωσης για την αριθμητική επίλυση υβριδικών DAE εξισώσεων.
- Ένα γραφικό user Interface
- Έναν text editor για την σύνταξη των μοντέλων του Modelica.

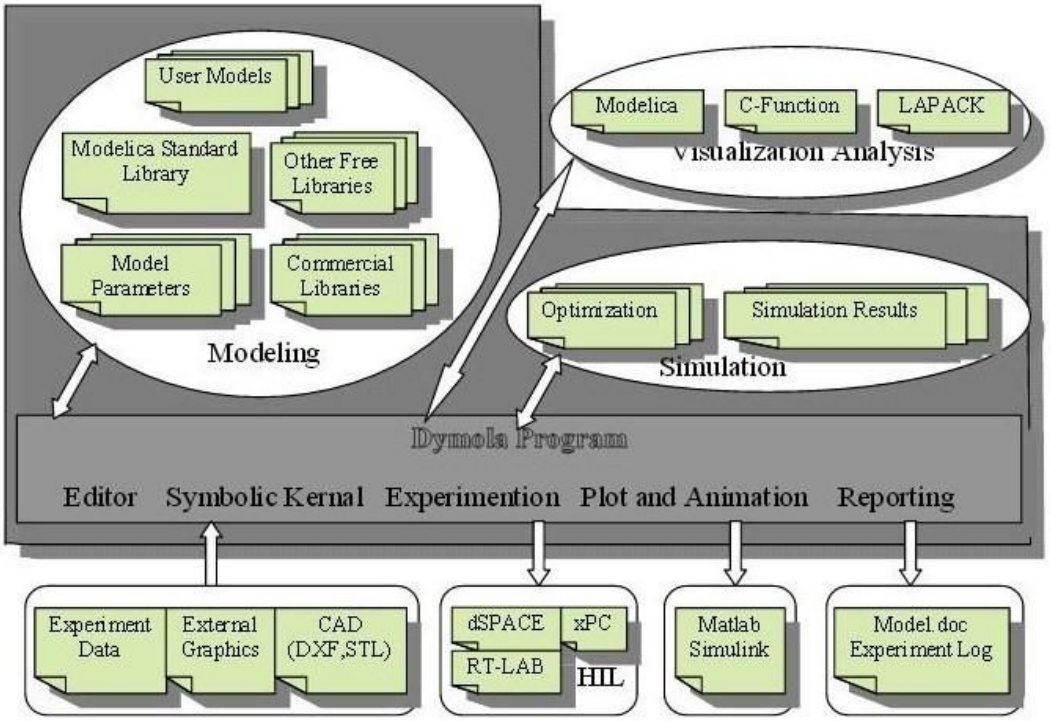
3.5. Η ΔΟΜΗ ΤΟΥ DYMOLA

Το DYMOLA είναι ένα πλήρως ολοκληρωμένο λογισμικό περιβάλλον το οποίο, όπως αναφέραμε, χρησιμοποιείται για μοντελοποίηση και προσομοίωση απλών ή πολύπλοκων φυσικών συστημάτων. Τα μοντέλα συνθέτονται με την χρήση των δομικών στοιχείων της κύριας βιβλιοθήκης (γλώσσα Modelica), καθώς επίσης και με άλλες γλώσσες ανοιχτού κώδικα ή εμπορικές.

Ο χρήστης μπορεί να φτιάξει τα δικά του μοντέλα, είτε συνδέοντας εξαρτήματα από τις ήδη υπάρχουσες βιβλιοθήκες είτε γράφοντας τις δικές του εξισώσεις. Επιπλέον έχει την δυνατότητα να χρησιμοποιήσει τον γραφικό συντάκτη μοντελοποίησης του προγράμματος και να σχεδιάσει ένα μοντέλο. Στο block διάγραμμα τοποθετεί τα εικονίδια που αντιπροσωπεύουν τα στοιχεία του μοντέλου, σχεδιάζοντας συνδέσεις και δίνοντας παραμέτρους τιμών στα «κουτιά» διαλόγου. Η βασισμένη σε εξισώσεις φύση της Modelica είναι ιδανική για χρήση επαναχρησιμοποιούμενων βιβλιοθηκών

σαν κι αυτές.

Το DYMOLA μετασχηματίζει μια επεξηγηματική, βασισμένη σε εξισώσεις, περιγραφή ενός προτύπου/μοντέλου σε αποδοτικό κώδικα προσομοίωσης. Χρησιμοποιεί άλγεβρα υπολογιστών για να χειρίζεται μεγάλα σετ από εξισώσεις. Επιπλέον το DYMOLA μπορεί να εξάγει κώδικα για προσομοίωση σε Simulink (MatLab). Έτσι έχουμε τη δυνατότητα το μοντέλο που έχουμε δημιουργήσει με το DYMOLA να το χρησιμοποιήσουμε ώστε να κάνουμε αριθμητικές προσομοιώσεις και



να δούμε πως δουλεύει το σύστημα, για παράδειγμα συναρτήσει του χρόνου, με τη βοήθεια του MatLab πάντα. Οπότε βλέπουμε ότι τα δυο εργαλεία μπορούν να συνδεθούν μεταξύ τους πάνω σ' ένα κοινό project κάνοντας το πιο αποτελεσματικό και αποφεύγοντας τα προβλήματα κατά τη σχεδίαση και τη μετέπειτα χρήση του σε πραγματικό χρόνο και περιβάλλον.

Εικόνα 2 Δομή του DYMOLA. Τα μοντέλα συνθέτονται με την χρήση της κύριας βιβλιοθήκης της Modelica, αλλά και την χρήση άλλων γλωσσών. Οι χρήστες μπορούν να γράψουν και τις δικές του εξισώσεις.

3.6. ΤΟ ΠΕΡΙΒΑΛΛΟΝ ΕΡΓΑΣΙΑΣ ΤΟΥ DYMOLA

Το DYMOLA έχει δυο παράθυρα εργασίας : το παράθυρο του block διαγράμματος (modeling) και το παράθυρο της προσομοίωσης (simulation).

Χρησιμοποιούμε το παράθυρο του block διαγράμματος (modeling) για να «χτίσουμε» το μοντέλο/πρότυπό μας, χρησιμοποιώντας τα στοιχεία που βρίσκονται στο διπλανό παράθυρο των βιβλιοθηκών της MODELICA. Το DYMOLA μας παρέχει κι έτοιμα πρότυπα παραδείγματα (demos) με τα οποία θ' ασχοληθούμε σε παρακάτω ενότητα αναλυτικά.

Με τη χρήση του παραθύρου προσομοίωσης (simulation), έχουμε την δυνατότητα να «τρέξουμε» το πρότυπο/μοντέλο μας και να δούμε τ' αποτελέσματα που μας δίνει σε real time συμπεριφορά. Έτσι μπορούμε να καταλάβουμε αν το πρότυπο μας συμπεριφέρεται όπως θέλουμε και στην ουσία αν είναι αποτελεσματικό και δουλεύει με τον επιθυμητό τρόπο.

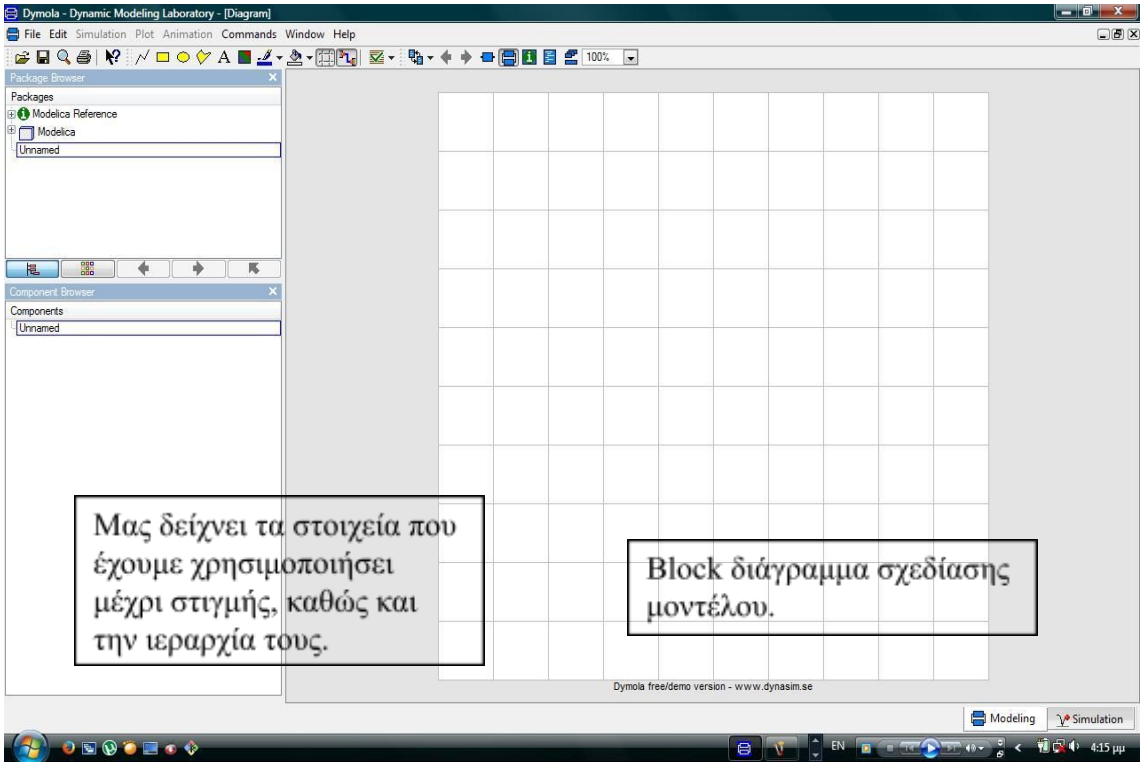
3.6.1. ΤΟ BLOCK DIAGRAM ΤΟΥ DYMOLA

Στην αριστερή μεριά του παραθύρου, Εικόνα 3., υπάρχουν τα browser πακέτα που περιέχουν τις βιβλιοθήκες με τα στοιχεία προτύπων της γλώσσας Modelica. Δεξιά έχουμε το block διάγραμμα στο οποίο τοποθετούμε τα στοιχεία ώστε να δημιουργήσουμε το επιθυμητό πρότυπο/μοντέλο. Ουσιαστικά είναι ο «καμβάς» που σχεδιάζουμε το μοντέλο μας. Το μόνο που χρειάζεται είναι να σύρουμε το εικονίδιο του στοιχείου από την βιβλιοθήκη στο block διάγραμμα.

Στο κάτω αριστερό μέρος της οθόνης μας φαίνονται όλα τα στοιχεία τα οποία έχουμε προσθέσει στο block διάγραμμα. Πατώντας επάνω σε οποιοδήποτε γίνεται ανάπτυξη των καρτελών η οποία ουσιαστικά δείχνει πως το κάθε στοιχείο συνδέεται με οποιοδήποτε άλλο, δηλαδή την ιεραρχία με την οποία είναι δομημένα ώστε να σχηματίζουν το σύστημα σαν σύνολο.

Τέλος, έχουμε την γραμμή εντολών με την οποία εκτελούμε διάφορες εργασίες άμεσα, όπως π.χ. να μεταβούμε στον κώδικα της Modelica που δημιουργείται αυτόματα όταν τοποθετήσουμε τα στοιχεία μας στο διάγραμμα. Αυτό άλλωστε είναι κι

ένα από τα μεγαλύτερα πλεονεκτήματα της χρήσης του DYMOLA. Επίσης υπάρχουν επιπλέον εικονίδια που μπορούμε να χρησιμοποιήσουμε είτε στη σχεδίαση προτύπων, είτε στην εναλλαγή μεταξύ παραθύρων αλλά και για να εκκινήσουμε την διαδικασία προσομοίωσης.



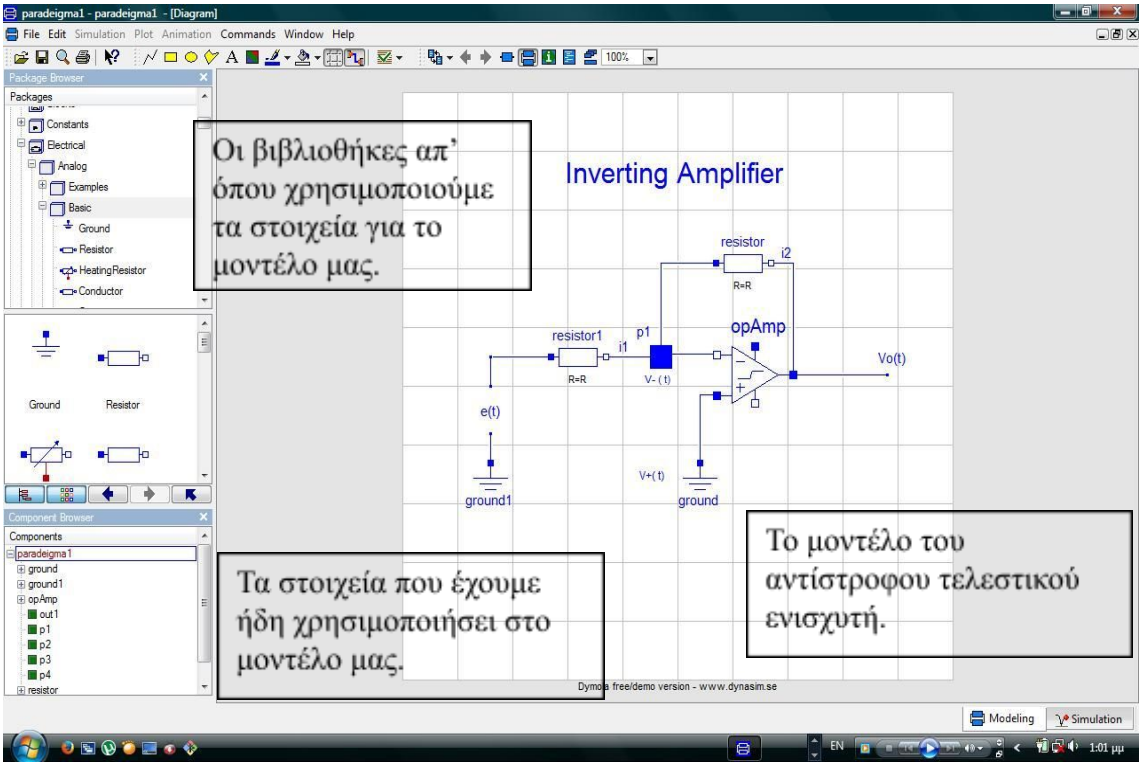
Εικόνα 3. Το παράθυρο εργασίας του DYMOLA για το block διαγραμμα.

3.6.1.1. ΑΠΛΟ ΠΑΡΑΔΕΙΓΜΑ ΔΗΜΙΟΥΡΓΙΑΣ ΜΟΝΤΕΛΟΥ – BLOCK ΔΙΑΓΡΑΜΜΑ

Παρακάτω παρουσιάζεται ένα απλό παράδειγμα δημιουργίας ενός αντίστροφου τελεστικού ενισχυτή με την βοήθεια των βιβλιοθηκών της Modelica. :

- από τις βιβλιοθήκες επιλέγουμε τα στοιχεία που χρειαζόμαστε για να δημιουργήσουμε το μοντέλο μας. Αυτά εμφανίζονται στο από κάτω παράθυρο και το μόνο που έχουμε να κάνουμε είναι να τα σύρουμε στο block διάγραμμα.

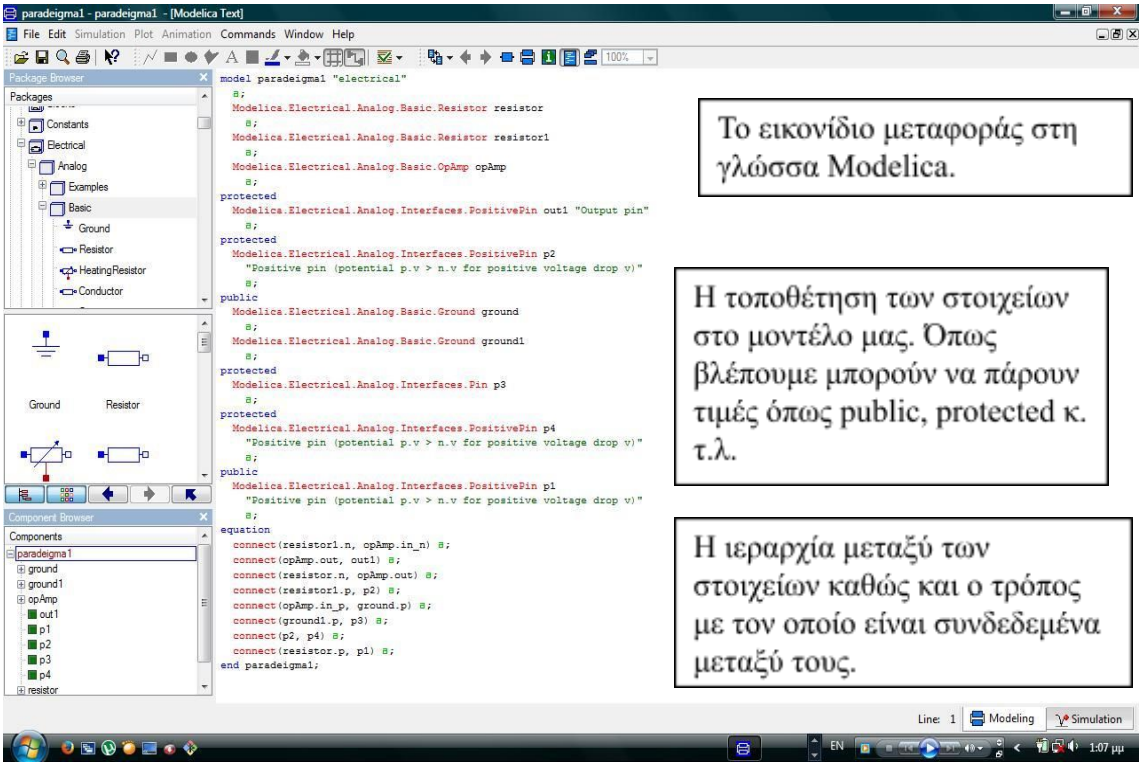
- σιγά – σιγά εισάγουμε τα στοιχεία και τα ενώνουμε καταλλήλως.
- κάτω αριστερά στην οθόνη εμφανίζονται όλα τα στοιχεία που έχουμε χρησιμοποιήσει καθώς επίσης και την ιεραρχία μεταξύ τους.



Εικόνα 4. Πρόχειρο μοντέλο αντίστροφου τελεστικού ενισχυτή.

3.6.1.2. ΑΠΛΟ ΠΑΡΑΔΕΙΓΜΑ ΔΗΜΙΟΥΡΓΙΑΣ ΜΟΝΤΕΛΟΥ - ΓΛΩΣΣΑ MODELICA

Επιλέγοντας από την γραμμή εργαλείων μπορούμε να επιλέξουμε να εμφανιστεί το πρόγραμμα σε γλώσσα Modelica του μοντέλου που έχουμε σχεδιάσει. Μ' αυτόν τον τρόπο μπορούμε να δούμε την δομή και την ιεραρχία με την οποία είναι δομημένα αυτά στο block διάγραμμα.

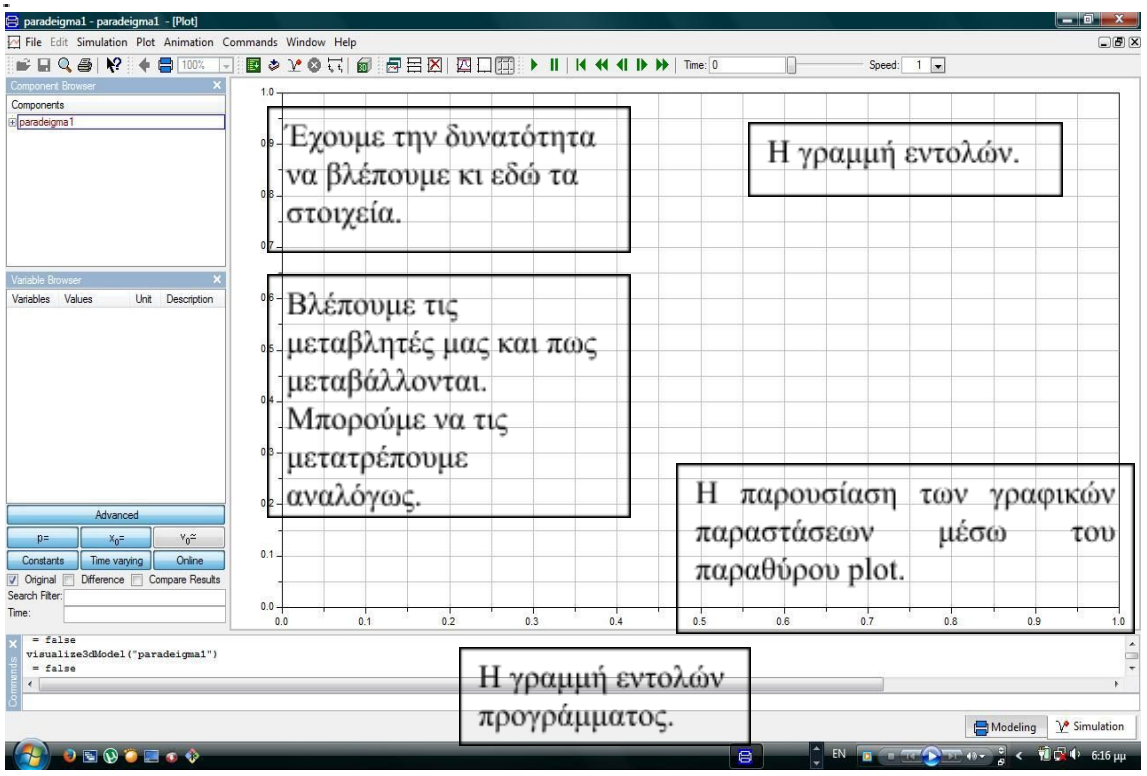


Εικόνα 5. Το μοντέλο σε γλώσσα προγραμματισμού Modelica.

Επίσης μπορούμε να πάρουμε τις κλάσεις με τα στοιχεία που έχουν δημιουργηθεί για το μοντέλο μας και να έχουμε μια πλήρη εικόνα για την λειτουργία του.

3.6.2. ΤΟ ΠΑΡΑΘΥΡΟ ΠΡΟΣΟΜΟΙΩΣΗΣ ΤΟΥ DYMOLA

Το μεγαλύτερο πλεονέκτημα από την χρήση του DYMOLA είναι η δυνατότητα που μας δίνεται να προσομοιώσουμε το μοντέλο μας και να έχουμε αποτελέσματα σε πραγματικό χρόνο. Έτσι έχουμε μια εικόνα πως αυτό θα συμπεριφερθεί στον «πραγματικό» κόσμο κι θα λειτουργεί με τον αναμενόμενο τρόπο. Αν όχι τι λάθη υπάρχουν και πως αυτά θα επιλυθούν.



Εικόνα 6. Το παράθυρο εργασίας προσομοίωσης του DYMOLA.

Από την γραμμή εντολών έχουμε τη δυνατότητα να επιλέξουμε εικονίδια για την εκτέλεση διαφορετικών εντολών, π.χ. διαφορετικές εργασίες του plot. Επίσης, μέσω αυτής μπορούμε να προσομοιώσουμε το μοντέλο μας και να μελετήσουμε την συμπεριφορά του επιλέγοντας το κουμπί της προσομοίωσης.

Σε περίπτωση που υπάρχει λάθος, μας εμφανίζεται μήνυμα λάθους στην γραμμή εντολών του προγράμματος.

3.6.3. ΠΡΟΣΟΜΟΙΩΣΗ ΕΝΟΣ ΥΠΑΡΧΟΝΤΟΣ ΠΡΟΤΥΠΟΥ/ΜΟΝΤΕΛΟΥ

Όπως αναφέραμε και στην αρχή, το DYMOLA μας δίνει την δυνατότητα να προσομοιώσουμε ένα ήδη δημιουργημένο μοντέλο ώστε ν' αποκτήσουμε μεγαλύτερη οικειότητα με το εργαλείο, καθώς και εξοικείωση με τον τρόπο λειτουργίας του. Επίσης η χρήση των υπαρχόντων προτύπων επιτρέπει στον χρήστη το πειραματισμό σε διάφορα επίπεδα, που μπορούν να βοηθήσουν στη δημιουργία ενός καινούργιου μοντέλου. Χρησιμοποιώντας ένα έτοιμο παράδειγμα θ' αναλύσουμε το block διάγραμμα καθώς και το παράθυρο προσομοίωσης του interface μας.

3.6.3.1. BLOCK ΔΙΑΓΡΑΜΜΑ – ΥΠΑΡΧΟΝ ΜΟΝΤΕΛΟ

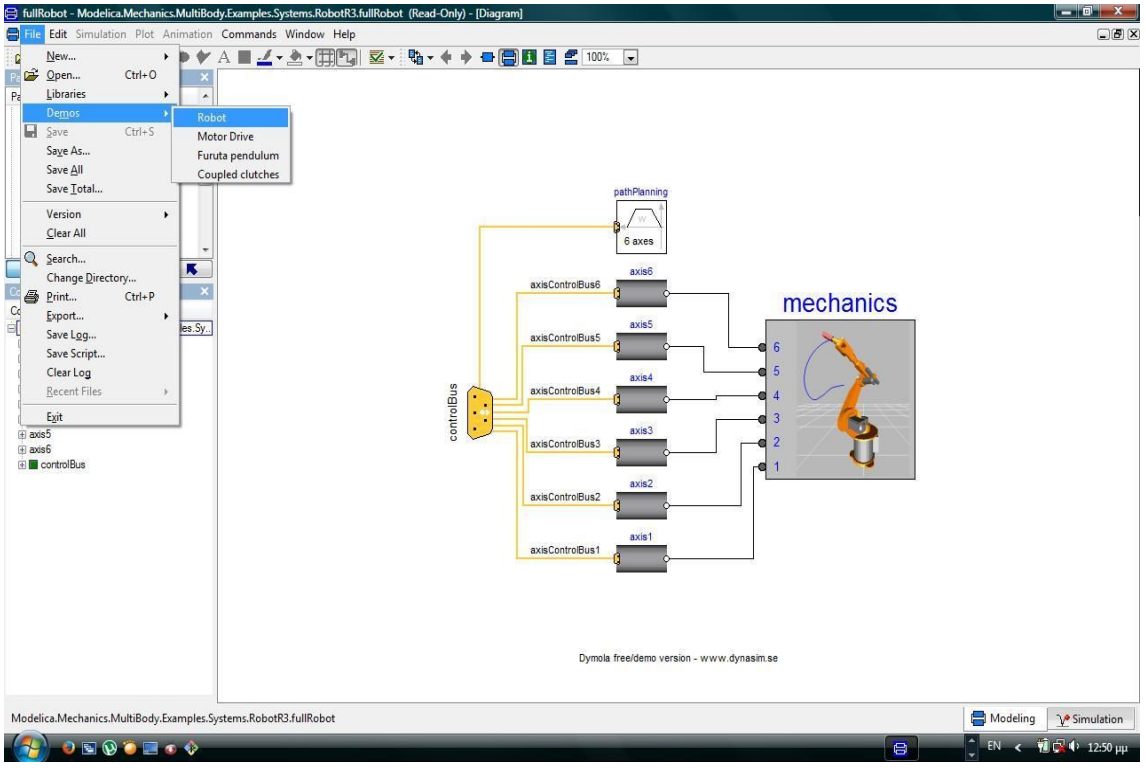
Ξεκινώντας στη γραμμή εργαλείων επιλέγουμε την καρτέλα File, μετά τα Demos και έπειτα ένα από τα υπάρχοντα πρότυπα. Όπως φαίνεται και στην Εικόνα 7 παρακάτω έχουμε τέσσερα έτοιμα μοντέλα προς χρήση, το ρομποτικό βραχίονα, τον κινητήρα, ένα εκκρεμές σε τρισδιάστατη μορφή και το ζεύγος συμπλέκτη – φρένων σε όχημα. Πατώντας οποιοδήποτε, αυτό εμφανίζεται στην οθόνη ως ένα ολοκληρωμένο σύστημα αποτελούμενο από δομικά στοιχεία δομημένα με ιεραρχική σειρά. Με το παραθύρο προσομοίωση «τρέχουμε» το μοντέλο μας. Έτσι βλέπουμε πως συμπεριφέρεται σαν σύνολο και μελετώντας τις γραφικές του παραστάσεις κατανοούμε τις παραμέτρους και τον τρόπο λειτουργίας του.

Έτσι έχουμε μια σφαιρική άποψη για συστήματα παρόμοιας φύσης και μπορούμε ν' αναπτύξουμε το δικό μας, τέτοιου είδους σύστημα, ευκολότερα. Ακόμα, μας δίνει μια ιδέα για τον τρόπο που θα πρέπει να συμπεριφέρεται το καινούργιο σύστημά μας και είναι πιο εύκολο να αναγνωρίσουμε τυχόν σφάλματα που θα μας

παρουσιαστούν.

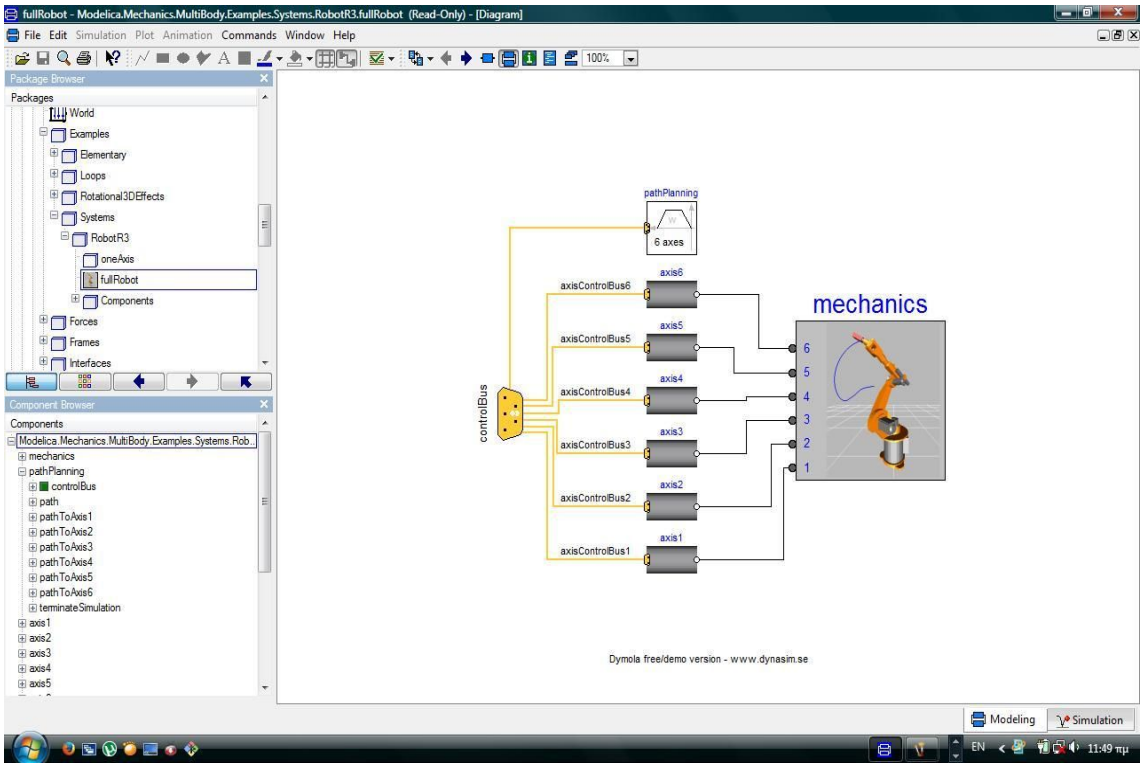
Στη συνέχεια θα επιλέξουμε ένα απ' αυτά τα πρότυπα και θα μελετήσουμε τη σύνθεση και το τι αντιπροσωπεύει το καθ' ένα στοιχείο που το αποτελεί, καθώς και τον τρόπο με τον οποίο λειτουργούν αυτά τα στοιχεία μεταξύ τους σαν δομή.

Για το παράδειγμά μας επιλέγουμε τον ρομποτικό βραχίονα (robot). Το παράθυρο του block διαγράμματός του εμφανίζεται στην παρακάτω εικόνα.



Εικόνα 7. Επιλογή ενός υπάρχοντος μοντέλου από την βιβλιοθήκη, του ρομποτικού βραχίονα και εμφάνιση του.

Στη παραπάνω εικόνα απεικονίζεται ένας πλήρως σχεδιασμένος ρομποτικός βραχίονας, έξι βαθμών ελευθερίας με ελεγκτές, μηχανολογικά μέρη, φρένα, ταχύτητες και μηχανικά στοιχεία. Μέσω της δυνατότητας προσομοίωσης που μας παρέχει το DYMOLA, βλέπουμε τον τρόπο λειτουργίας τέτοιων συστημάτων. Ακόμα καλύτερα, έχουμε την δυνατότητα να ελέγξουμε ένα προς ένα τα στοιχεία και τα μέρη του προτύπου και να δούμε πως αυτά συνδέονται μεταξύ τους καθώς και πως είναι δομημένα ιεραρχικά το ένα σε σχέση με το άλλο. Σ' αυτό βέβαια μας βοηθάνε και τα παράθυρα βιβλιοθηκών και πακέτων που μας παρέχει μόνο του το σύστημα.



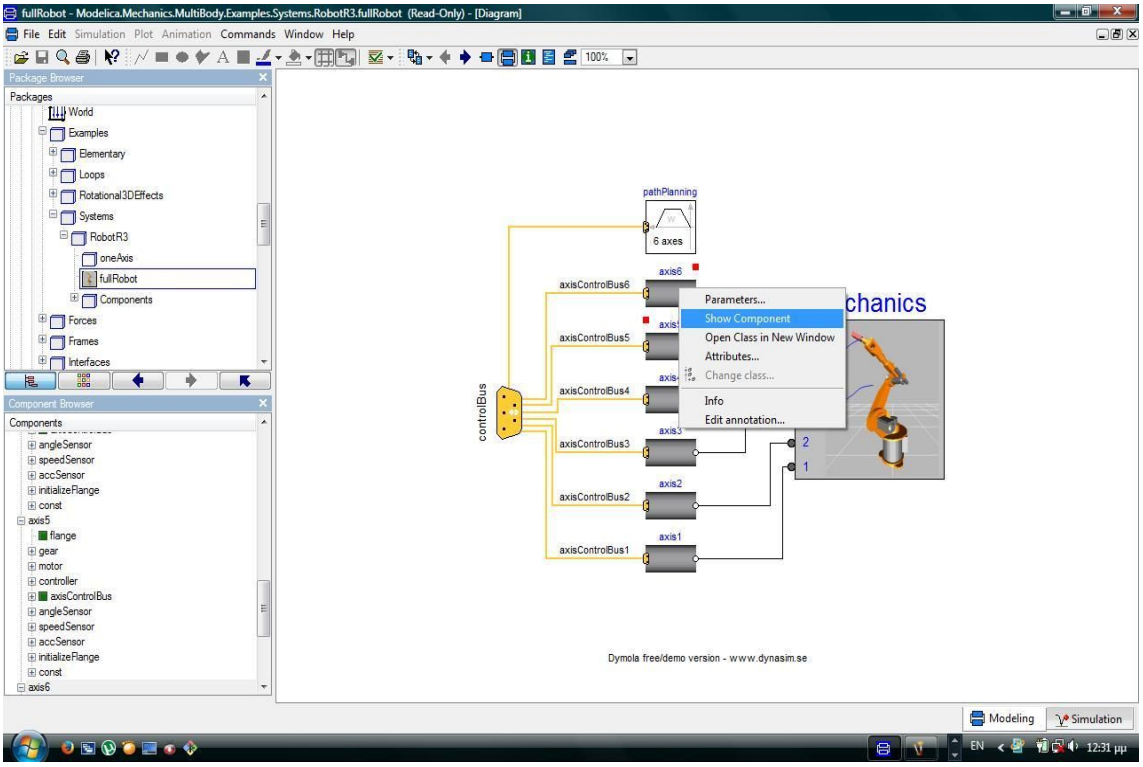
Εικόνα 8. Στο αριστερό μέρος της οθόνης του interface φαίνονται τα παράθυρα βιβλιοθηκών καθώς και η ιεραρχία μεταξύ των στοιχείων που αποτελούν το μοντέλο.

Ένα από τα πλεονεκτήματα του DYMOLA, είναι η αποτελεσματικότητα κι ευκολία με την οποία απεικονίζει πολύπλοκά συστήματα και έτσι μας βοηθά στην κατανόησή τους σε μικρότερο χρόνο. Το έτοιμο πρότυπο που εξετάζουμε ως παράδειγμα είναι ήδη ένα πολύπλοκο και δυσνόητο σύστημα. Παρ' όλ' αυτά εκτελώντας τις κατάλληλες εντολές ο χρήστης είναι σε θέση να εξετάσει τα στοιχεία (τα οποία είναι δομημένα σε blocks) που το αποτελούν ως σύνολο αλλά και το καθ' ένα στοιχείο ξεχωριστά.

Ο ρομποτικός βραχίονας που εξετάζουμε έχει έξι βαθμούς ελευθερίας, οι οποίοι

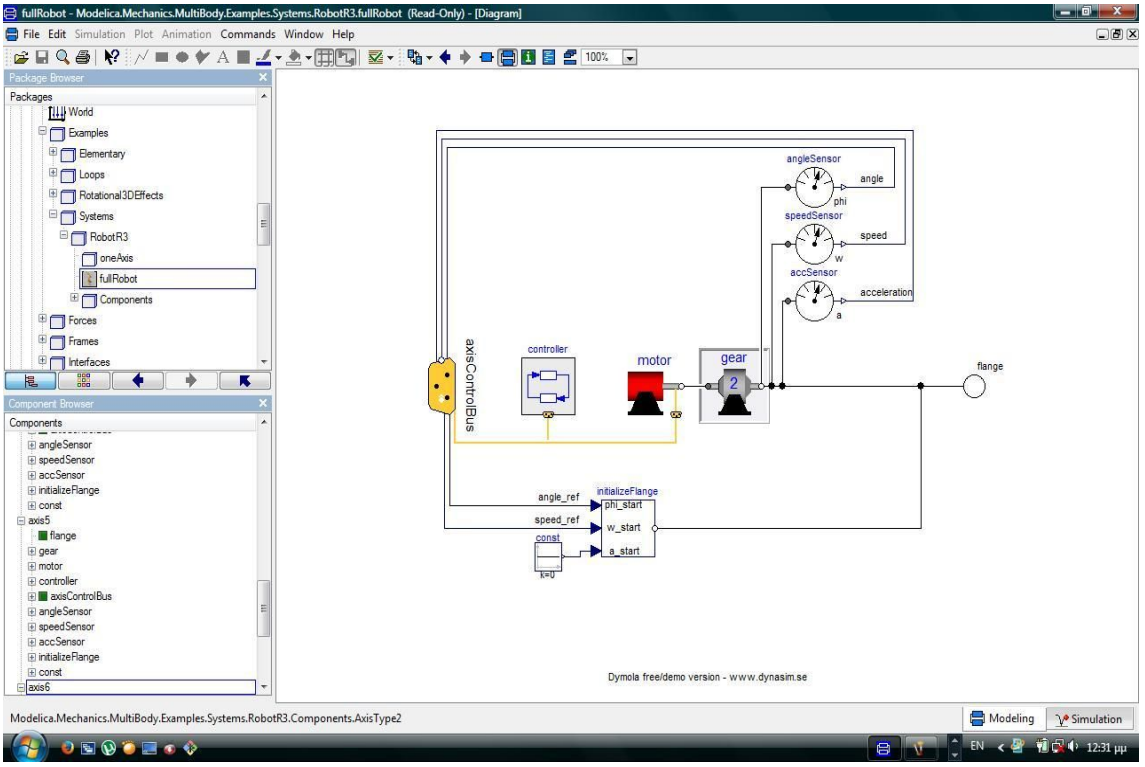
πρέπει να είναι σε αρμονία μεταξύ τους και να δουλεύουν ταυτόχρονα. Στο demo – πρότυπο αυτοί παρουσιάζονται με τα block στοιχεία axis. Έχουμε έξι τέτοια block στοιχεία που αντιστοιχούν σε κάθε βαθμό. Το DYMOLA μας επιτρέπει να δούμε πως είναι δομημένα αυτά τα block στοιχεία και από τι αποτελούνται.

Πατώντας πάνω σε οποιαδήποτε στοιχείο και κάνοντας δεξί κλικ εμφανίζεται ένα παράθυρο εργασίας. Απ' αυτό επιλέγουμε τη ετικέτα show component και μας



εμφανίζει την δομή του στοιχείου. Για παράδειγμα, θα επιλέξουμε το block στοιχείο axis6, το οποίο αντιστοιχεί στο βαθμό ελευθερίας περιστροφής της παλάμης του βραχίονα. Η διαδικασία επιλογής φαίνεται στις παρακάτω εικόνες.

Εικόνα 9. Επιλογή απεικόνισης των δομικών στοιχείων που αποτελούν το block στοιχείο axis6.



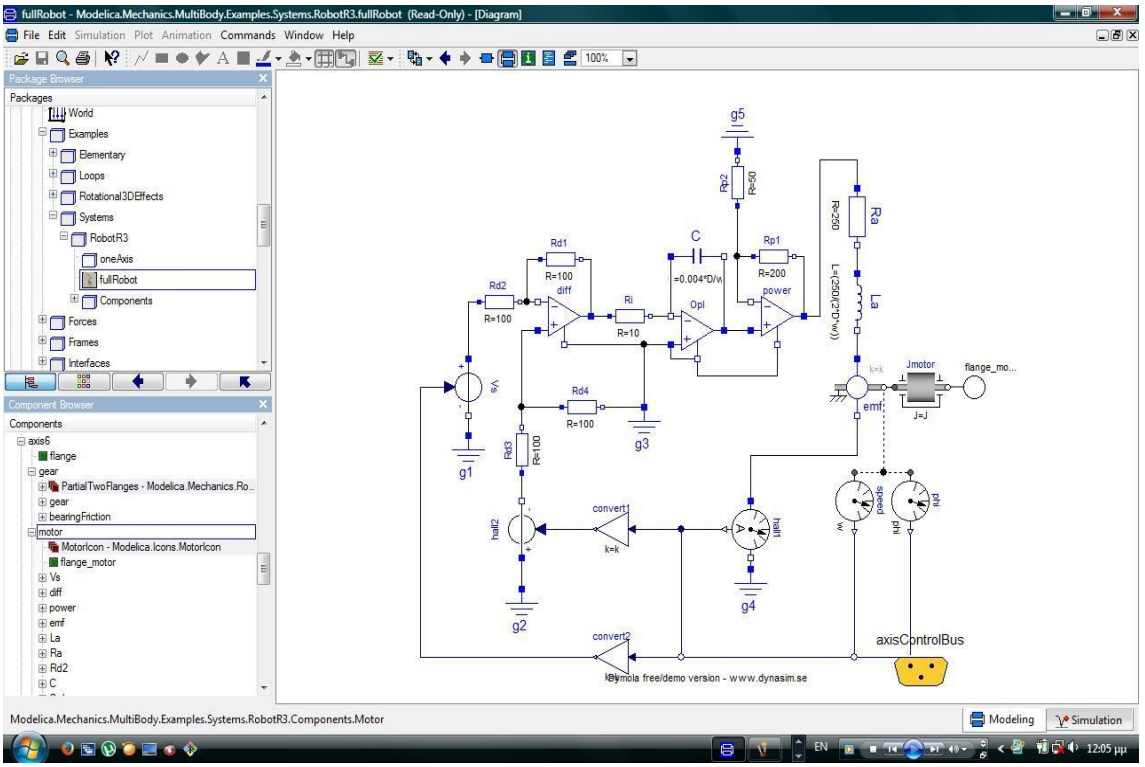
Εικόνα 10. Απεικόνιση των δομικών στοιχείων που αποτελούν το block στοιχείο axis6.

Το block στοιχείο axis6 αποτελεί τον έκτο βαθμό ελευθερίας του βραχίονα. Για να πετύχουμε την περιστροφή το στοιχείο αποτελείται από έναν ελεγκτή για τον καθορισμό και των έλεγχο της κίνησης. Επίσης έχουμε τον κινητήρα ο οποίος παράγει την κίνηση συμπεριλαμβανομένου ενός επιπλέον ελεγκτή, καθώς και του δοχείου ταχυτήτων το οποίο είναι υπεύθυνο για την ελαστικότητα της ταχύτητας των κινήσεων, καθώς και της τριβής του άξονα του βραχίονα. Τέλος, περιέχονται αισθητήρες που καθορίζουν τον τρόπο που πρέπει να κινηθεί η παλάμη του βραχίονα καθώς και την ταχύτητα αυτής.

Οι αισθητήρες χρησιμοποιούνται για να καθορίσουν την κίνηση στον βραχίονα.

Παρατηρώντας στην Εικόνα 10. βλέπουμε ότι είναι υπεύθυνοι για την γωνία, την ταχύτητα και την επιτάχυνση της κίνησης. Και οι τρεις συνδέονται στο δοχείο ταχυτήτων ώστε μέσω του ελεγκτή να είναι σωστή η λειτουργία τους, αλλά επίσης και μέσω του ControlBus να υπάρχει σύνδεση και αρμονία με τα υπόλοιπα block στοιχεία axis που αντιστοιχούν στις άλλες κινήσεις του βραχίονα.

Τα δομικά στοιχεία που αποτελούν το δομικό στοιχείο axis6, αποτελούνται κι αυτά από επιπλέον στοιχεία. Αυτό είναι λογικό, καθώς έχουμε να κάνουμε με πολύπλοκα συστήματα μηχανικής και ρομποτικής τα οποία αποτελούνται από πολλά εξαρτήματα εναρμονισμένα μεταξύ τους. Έτσι αν για παράδειγμα, πατήσουμε πάνω στο block διάγραμμα του axis6, και επιλέξουμε πάλι την ετικέτα show component για κάποιο από τα στοιχεία του θα μας εμφανίσει την δομή τους. Ακολουθώντας πάλι την



διαδικασία επιλέγουμε τη εμφάνιση της δομής του κινητήρα που περιλαμβάνεται στο διάγραμμα του axis6. Στην παρακάτω εικόνα, Εικόνα 11., μας εμφανίζεται το block διάγραμμα από τα στοιχεία που αποτελούν τον κινητήρα του έκτου βαθμού ελευθερίας του ρομποτικού βραχίονα.

Εικόνα 11. Απεικόνιση του block διαγράμματος των δομικών στοιχείων που αποτελούν το

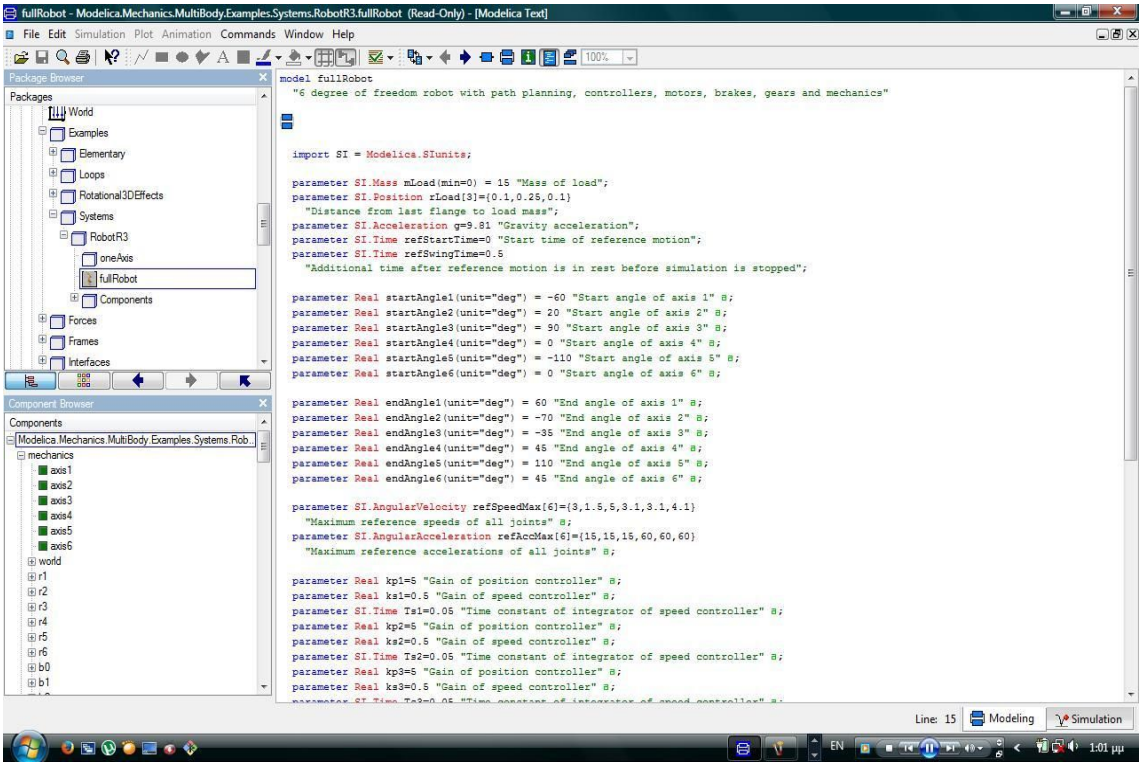
δομικό στοιχείο του κινητήρα για το στοιχείο axis6.

Παρατηρώντας την Εικόνα 11., βλέπουμε ότι ο κινητήρας αποτελείται από βασικά ηλεκτρονικά δομικά στοιχεία, όπως αντιστάσεις, διόδους, πυκνωτές κ.τ.λ.. Αυτά είναι συνδεδεμένα μεταξύ τους ώστε σε πραγματικό χρόνο η μηχανή θεωρητικά θα δουλεύει ανάλογα. Με βάση το παραπάνω παράδειγμα κατανοούμε πόσο εύχρηστο και σημαντικό εργαλείο είναι το DYMOLA, καθώς όχι μόνο μας επιτρέπει να μελετήσουμε το πρότυπο πριν την κατασκευή του, αλλά επίσης μας δείχνει τον θεωρητικό τρόπο που θα πρέπει να κατασκευαστεί ώστε να δουλεύει πρακτικά.

Όλα τα ρομποτικά, μηχανικά/μηχανολογικά συστήματα ξεκινάνε να χτίζονται χρησιμοποιώντας τα βασικά στοιχεία και όσο προχωράνε γίνονται όλο και πιο περίπλοκα. Η βασική προϋπόθεση όμως για έναν μηχανικό είναι να κατανοήσει την βάση του μοντέλου που χτίζει, πριν προχωρήσει σε επόμενο επίπεδο και σε πιο πολύπλοκη αρχιτεκτονική. Με το DYMOLA η παραπάνω προϋπόθεση ικανοποιείται, καθώς μπορούμε να δούμε την «αρχική» κατάσταση κάθε στοιχείου που αποτελεί το μοντέλο μας, χωρίς ουσιαστικά να το φτιάξουμε εμείς, αφού τα δομικά στοιχεία για τη δημιουργία προτύπων προϋπάρχουν ήδη στις βιβλιοθήκες της Modelica. Επίσης, εφόσον το εργαλείο μας επιτρέπει την χρήση έτοιμων προτύπων δεν χρειάζεται η κατασκευή κομματιών απαραίτητων για το δικό μας μοντέλο από την αρχή. Μπορούμε απλά να χρησιμοποιήσουμε τα μέρη που μας είναι απαραίτητα από τα υπάρχοντα μοντέλα. Έτσι γλυτώνουμε και χρόνο και κόστος τόσο στην δημιουργία του μοντέλου μας, όσο και στη κατασκευή του.

3.6.3.2. Η ΓΛΩΣΣΑ MODELICA - ΥΠΑΡΧΟΝ ΜΟΝΤΕΛΟ

Επιλέγοντας το εικονίδιο από την γραμμή εργαλείων μπορούμε να δούμε πως είναι δομημένο το block διάγραμμά μας στην γλώσσα Modelica αλλά και τις κλάσεις που συνθέτουν το πρότυπο, Εικόνα 12..



Εικόνα 12. Το πρότυπο του ρομποτικού βραχίονα γραμμένο σε γλώσσα Modelica, όπως εμφανίζεται στο παράθυρο του DYMOLA.

Στο παράθυρο της γλώσσας εμφανίζονται οι παράμετροι που έχουν δημιουργηθεί και χρησιμοποιηθεί κατά την κατασκευή του μοντέλου. Οι παράμετροι αυτοί αναφέρονται στα block δομικά στοιχεία που χρησιμοποιήθηκαν και έχουν διαφορετικό όνομα ανάλογα το block που αντιπροσωπεύουν αλλά και την χρήση τους. Σ' αυτές έχουν δοθεί τιμές, οι οποίες αντιστοιχούν στις τιμές που υπάρχουν στα αντίστοιχα block στοιχεία. Οι τιμές αυτές θα χρησιμοποιηθούν κατά την διάρκεια της προσομοίωσης, ώστε να γίνουν οι κατάλληλες πράξεις μέσω των εξισώσεων που συνδέουν τα δομικά στοιχεία για να λειτουργήσει το σύστημα και να παραχθούν τα επιθυμητά αποτελέσματα.

Στο τέλος κάθε παραμέτρου δίνεται μια ετικέτα με τ' όνομα περιγραφής, η οποία ουσιαστικά είναι η επεξήγηση για το τι προηγήθηκε. Η επεξήγηση βρίσκεται ανάμεσα σε εισαγωγικά ("…"). Μ' αυτόν τον τρόπο η Modelica περιγράφει ακόμα πιο αναλυτικά το τι έχει ήδη δημιουργηθεί σε περίπτωση που υπάρξει σύγχυση στη κατανόηση από μεριάς κάποιου χρήστη.

Όπως αναφέραμε οι παράμετροι της Modelica ουσιαστικά αντιστοιχούν στα block δομικά στοιχεία που αποτελούν το μοντέλο που απεικονίζεται στο block διάγραμμα. Η διαφορά τους σε σχέση με τα δομικά στοιχεία είναι ότι αυτές δημιουργούνται όταν τοποθετούνται τα στοιχεία και συνδέονται μεταξύ τους ώστε να δημιουργήσουν το μοντέλο. Όπως τα δομικά στοιχεία, έτσι κι αυτές είναι ιεραρχικά δομημένες μεταξύ τους και η μια έχει άμεση σύνδεση κι εξάρτηση με την άλλη. Η όλη αλληλεξάρτηση παρουσιάζεται με την μορφή κλάσεων.

Για παράδειγμα, παίρνουμε την παράμετρο που αντιστοιχεί στην επιτάχυνση της βαρύτητας:

parameter SI.Acceleration g=9.81 "Gravity acceleration";

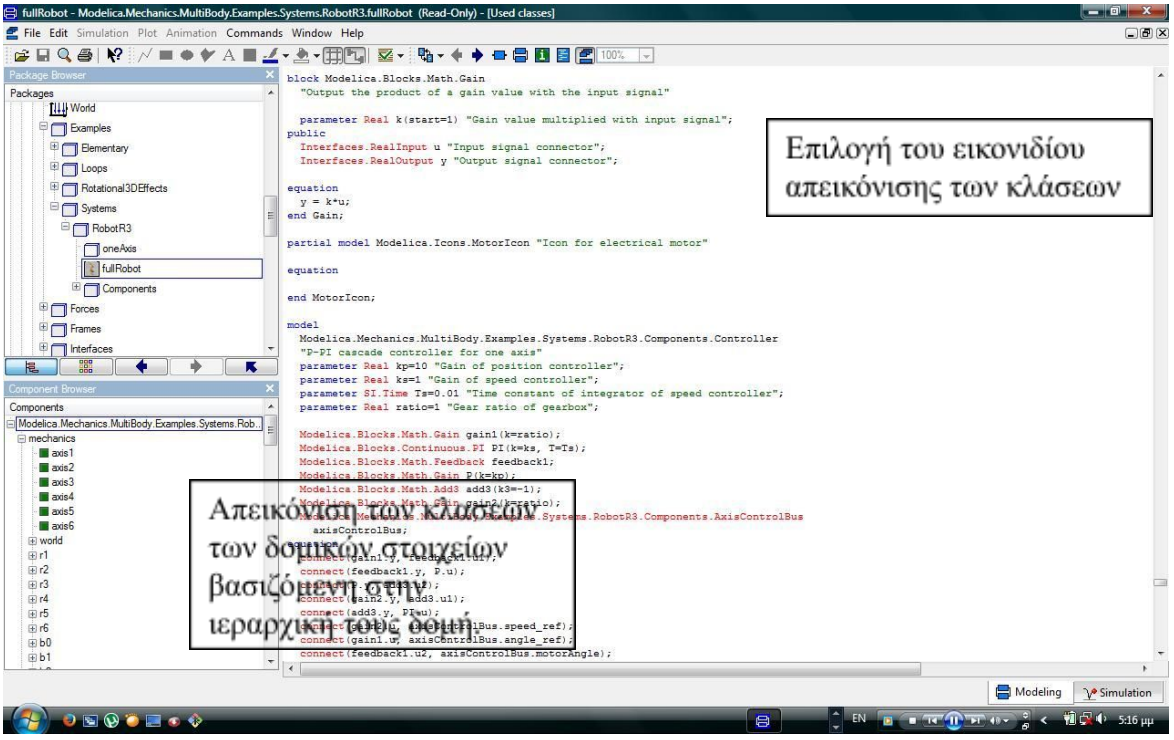
Εδώ δηλώνεται το block της βιβλιοθήκης στο οποίο αντιστοιχεί (SI.Acceleration), καθώς επίσης της έχει δοθεί η τιμή της, γνωστή από τους νόμους της βαρύτητας στη φυσική (g = 9.81). Η τιμή θα χρησιμοποιηθεί στις εξισώσεις που συνδέουν την συγκεκριμένη παράμετρο με τις υπόλοιπες για να μας δώσουν τον αποτέλεσμα στις ανάλογες εξισώσεις. Η ετικέτα που βρίσκεται ανάμεσα στα εισαγωγικά ("Gravity acceleration") μας επεξηγεί ουσιαστικά το τι δηλώνει η παράμετρος, δηλαδή την τιμή

της βαρύτητας. Η παράμετρος τερματίζει με το ερωτηματικό (; - semi colon στον προγραμματισμό).

3.6.3.3. ΟΙ ΚΛΑΣΕΙΣ ΤΗΣ ΓΛΩΣΣΑΣ MODELICA – ΥΠΑΡΧΟΝ ΜΟΝΤΕΛΟ

Από προηγούμενη ενότητα γνωρίζουμε ότι τα block δομικά στοιχεία είναι ιεραρχικά δομημένα μεταξύ τους. Επομένως και οι παράμετροι της γλώσσας Modelica είναι κι αυτές δομημένες με βάση την ιεραρχία των στοιχείων και επομένως συνδέονται και σχετίζονται άμεσα η μια με την άλλη. Η ιεραρχία κι η συσχέτιση αυτή εμφανίζεται με την μορφή κλάσεων στο DYMOLA.

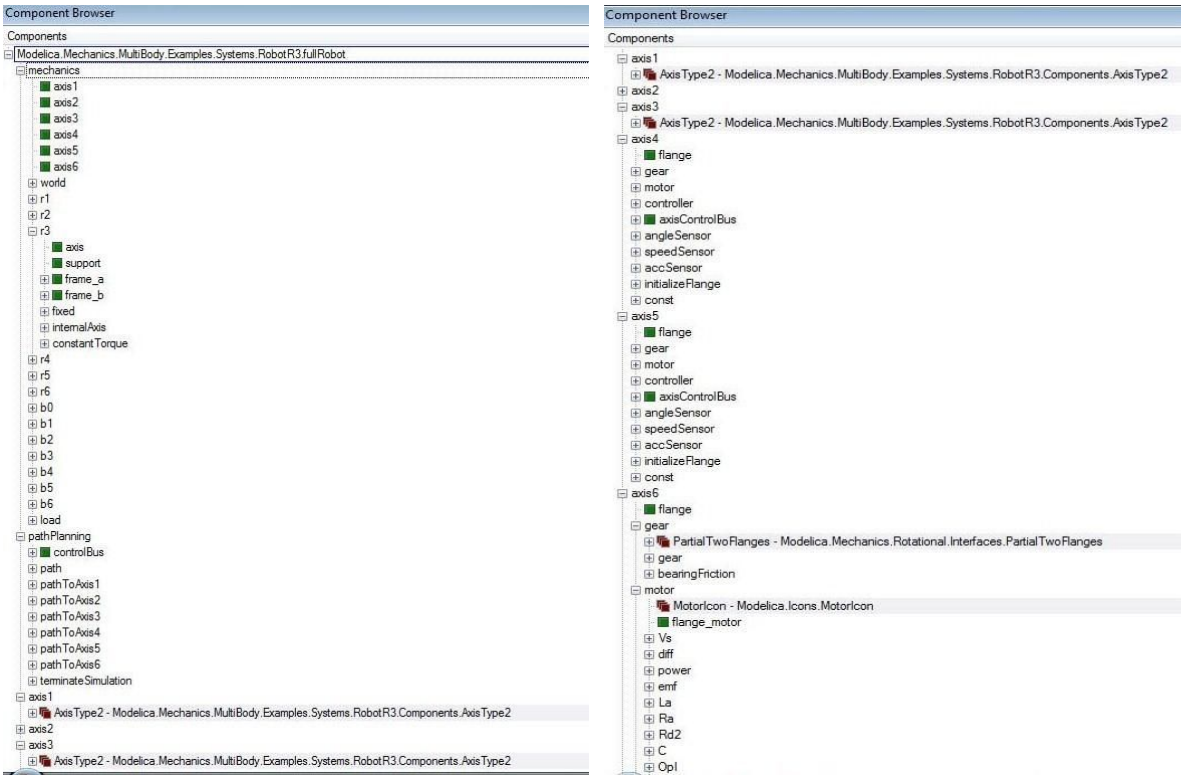
Από την γραμμή εργαλείων επιλέγουμε το εικονίδιο που μας εμφανίζει το παράθυρο με τις κλάσεις. Η διαδικασία απεικονίζεται στη παρακάτω εικόνα.



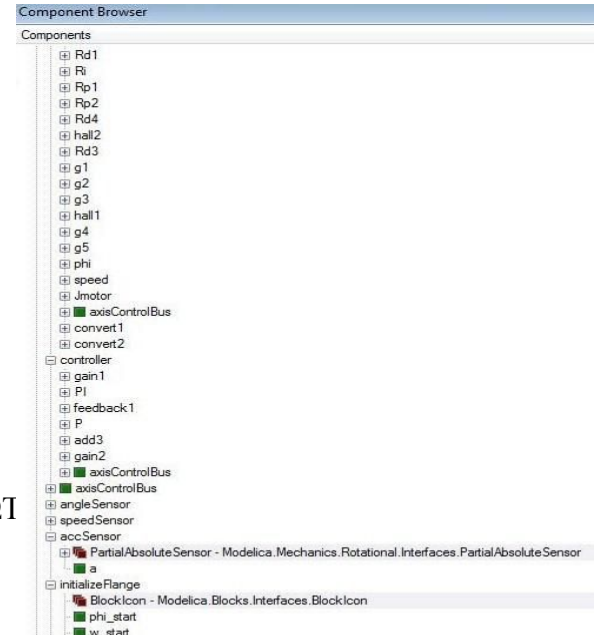
Εικόνα 13. Το παράθυρο απεικόνισης των κλάσεων της Modelica για το μοντέλο του ρομποτικού βραχίονα.

Οι κλάσεις φαίνονται και στο αριστερό μέρος της οθόνης στο browser των στοιχείων, με συμπυκνωμένη μορφη. Επίση στο παράθυρο αυτό απεικονίζονται τα

στοιχεία που έχουν χρησιμοποιηθεί, σε ποιο κομμάτι του μοντέλου υπάρχουν και που αντιστοιχούν. Επίσης ο τρόπος που συνδέονται αυτά μεταξύ τους μέσω των κλάσεων. Πιο συκγεκριμένα, μπορούμε να τα δουμε στην Εικόνα 14. Στην εικόνα εμφανίζονται τόσο τα ονόματα των στοιχείων, όσο και των κλάσεων που βρίσκονται. Η μεγαλύτερη κλάση είναι αυτή που αντιστοιχεί στο μοντέλο μας ως σύνολο – Modelica.Mechanics.MultiBody.Examples.Systems.RobotR3.fullRobot - και εμπεριέχει όλες τις υπόλοιπες. Με λίγα λόγια μπόρούμε να πούμε ότι οι κλάσεις είναι extend στην κεντρική μας κλάση. Οι κλάσεις συνδέονται μεταξύ τους ώστε να λειτουργεί το σύστημα.

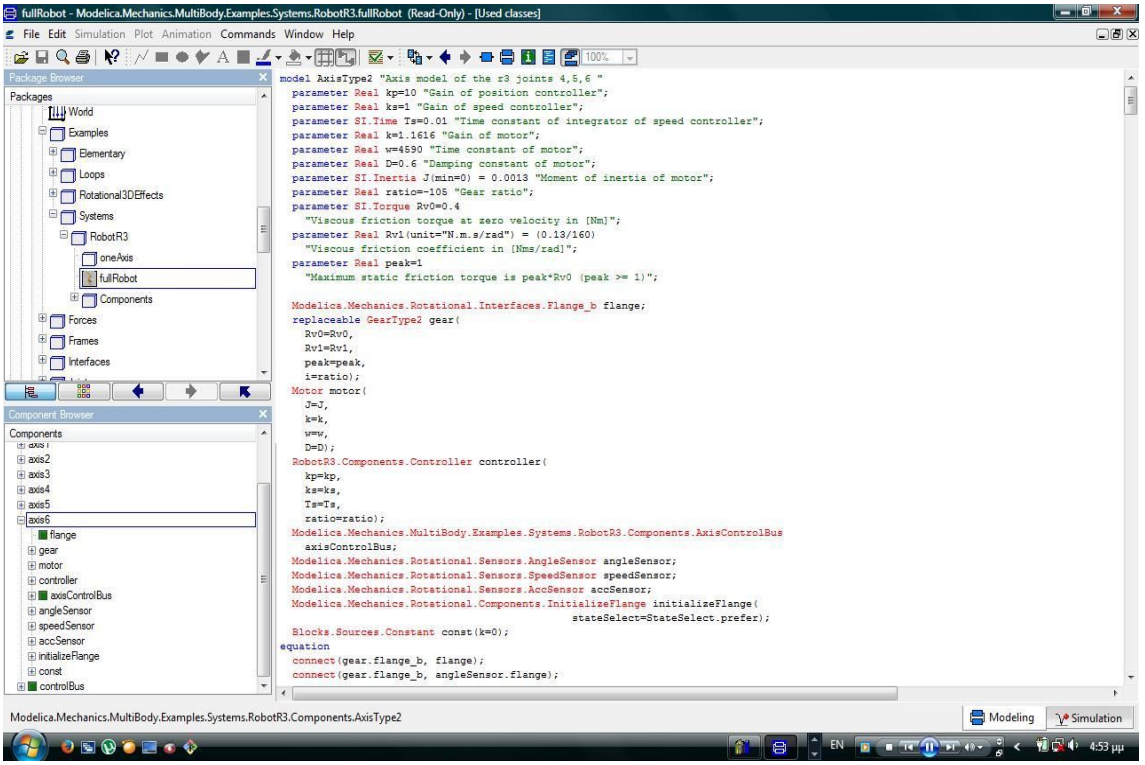


Εικόνα 14. Η ιεραρχική δομή των κλάσων του προτύπου όπως φαίνεται στο browser των στοιχείων.



Ένας εύκολος και γρήγορος τρόπος για να δούμε μια κλάση καθώς και τα

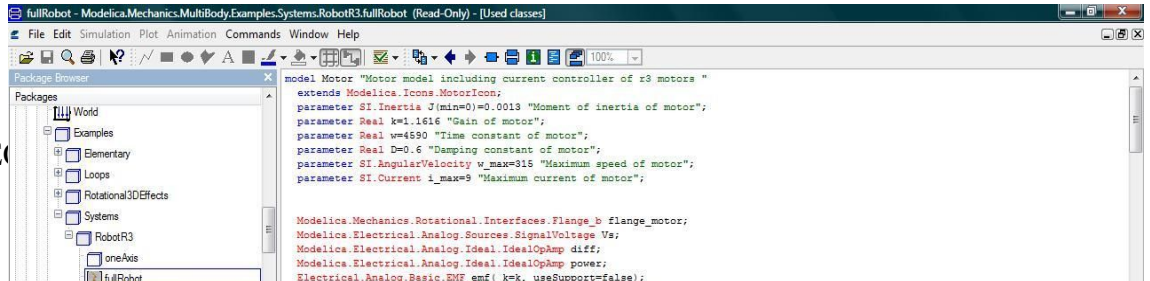
στοιχεία που την αποτελούν είναι να βρισκόμαστε στην καρτέλα των κλάσεων στο κεντρικό παράθυρο και να επιλέξουμε την κλάση που μας ενδιαφέρει πατώντας πάνω της στο browser των στοιχείων. Η διαδικασία φαίνεται στη παρακάτω εικόνα,



Εικόνα 15.

Εικόνα 15. Επιλογή της κλάσης που καθορίζει τον έκτο βαθμό ελευθερίας του ρομποτικού βραχίονα.

Στην εικόνα απεικονίζεται η κλάση που καθορίζει τον έκτο βαθμό ελευθερίας του ρομποτικού βραχίονα (axis6). Η κλάση περιλαμβάνει στοιχεία όπως οι αισθητήρες κίνησης, το κιβώτιο ταχυτήτων, ο κινητήρας κ.τ.λ.. Τα στοιχεία αυτά είναι υπεύθυνα για την κίνηση της παλάμης του κινητήρα, όπως τα εξηγήσαμε και στην Ενότητα 3.6.3.1.. Πατώντας επάνω στο δομικό στοιχείο axis6 μας εμφανίζεται η κλάση ολοκληρωμένη. Παρ' όλ' αυτά έχουμε την δυνατότητα να επιλέξουμε και να εμφανίσουμε την κλάση ενός εσωτερικού δομικού στοιχείου, για πάραδειγμα του κινητήρα, Εικόνα 16.



Εικόνα 16. Απεικόνιση της κλάσης του κινητήρα για το δομικό στοιχείο του axis6.

Στη συνέχεια θα εξετάσουμε πιο αναλυτικά ένα κομμάτι κώδικα του δομικού στοιχείου axis6 που παρουσιάστηκε στις προηγουμενες παραγράφους. Θα προσπαθήσουμε να εξηγήσουμε και να κατανοήσουμε τον τρόπο που λειτουργούν οι κλάσεις, αλλά και τη σημασία τους στη κατασκευή των μοντέλων.

Ξεκινώντας, επομένως, για το κομμάτι του κώδικα που έχουμε ως αναφορά η κλάση αρχίζει με το να θέτει τις παραμέτρους για την κίνηση του κινητήρα. Στις παραμέτρους αυτές δίνονται οι τιμές και το είδος τους. Για παράδειγμα, για την παράμετρο του κέρδους (k) του κινητήρα μας δίνεται η τιμή του και καθορίζεται με την λέξη Real, καθώς η τιμή αυτή αποτελεί καθαρό αριθμό. Τέλος, δίνεται η επεξήγηση της παραμέτρου, ώστε ο χρήστης να ξέρει που χρησιμοποιείται αυτή.

```
parameter SI.Inertia J(min=0)=0.0013 "Moment of inertia of motor";
parameter Real k=1.1616 "Gain of motor";
parameter Real w=4590 "Time constant of motor";
parameter Real D=0.6 "Damping constant of motor";
parameter SI.AngularVelocity w_max=315 "Maximum speed of motor";
parameter SI.Current i_max=9 "Maximum current of motor";
```

Στη συνέχεια προσθέτονται τα δομικά στοιχεία τα οποία χρησιμοποιούνται, καθώς επίσης τους δίνονται και οι τιμές τους. Κάθε δομικό στοιχείο έχει το όνομα της βιβλιοθήκης της οποίας ανήκει καθώς και των προεκτάσεων αυτής. Σε ορισμένα στοιχεία εμφανίζονται τιμές, οι οποίες έχουν δοθεί από εκείνον που φτιάχνει το μοντέλο – τον μηχανικό στη συγκεκριμένη περίπτωση – κατά τη διαδικασία δημιουργίας. Όπως έχουμε αναφέρει και σε προηγούμενα κεφάλαια, η γλώσσα προγραμματισμού (Modelica) δημιουργείται αυτόματα κατά τη διάρκεια του «χτισίματος» του μοντέλου στο block διάγραμμα. Εφόσον λοιπόν κατά αυτήν τη διαδικασία δίνονται τιμές σε στοιχεία, οι τιμές αυτές ενσωματώνονται και στις αντίστοιχες κλάσεις της γλώσσας Modelica. Το κομμάτι κώδικα που ακολουθεί συνεχίζει από το σημείο που σταματήσαμε παραπάνω εφαρμόζοντας όσα αναφέραμε στην παράγραφο οσον αφορά την ενσωμάτωση των τιμών στις κλάσεις.

```
Modelica.Mechanics.Rotational.Interfaces.Flange_b flange_motor;
```

```
  Modelica.Electrical.Analog.Sources.SignalVoltage Vs;
 Modelica.Electrical.Analog.Ideal.IdealOpAmp diff;
  Modelica.Electrical.Analog.Ideal.IdealOpAmp power;
  Electrical.Analog.Basic.EMF emf( k=k, useSupport=false);
Modelica.Electrical.Analog.Basic.Inductor La(L=(250/(2*D*w)));
  Modelica.Electrical.Analog.Basic.Resistor Ra(R=250);
  Modelica.Electrical.Analog.Basic.Resistor Rd2(R=100);
Modelica.Electrical.Analog.Basic.Capacitor C(C=0.004*D/w);
Modelica.Electrical.Analog.Ideal.IdealOpAmp Opl;
  Modelica.Electrical.Analog.Basic.Resistor Rd1(R=100);
  Modelica.Electrical.Analog.Basic.Resistor Ri(R=10);
Modelica.Electrical.Analog.Basic.Resistor Rp1(R=200);
 Modelica.Electrical.Analog.Basic.Resistor Rp2(R=50);
 Modelica.Electrical.Analog.Basic.Resistor Rd4(R=100);
  Modelica.Electrical.Analog.Sources.SignalVoltage hall2;
  Modelica.Electrical.Analog.Basic.Resistor Rd3(R=100);
  Modelica.Electrical.Analog.Basic.Ground g1;
 Modelica.Electrical.Analog.Basic.Ground g2;
  Modelica.Electrical.Analog.Basic.Ground g3;
  Modelica.Electrical.Analog.Sensors.CurrentSensor hall1;
 Modelica.Electrical.Analog.Basic.Ground g4;
Modelica.Electrical.Analog.Basic.Ground g5;
  Modelica.Mechanics.Rotational.Sensors.AngleSensor phi;
  Modelica.Mechanics.Rotational.Sensors.SpeedSensor speed;
Modelica.Mechanics.Rotational.Components.Inertia Jmotor(
                          J=J);
Modelica.Mechanics.MultiBody.Examples.Systems.RobotR3.Components.Axis
ControlBus
  axisControlBus;
  Blocks.Math.Gain convert1;
  Blocks.Math.Gain convert2;
```

Τέλος, θα πρέπει να υπάρξει μια σύνδεση όλων των παραμέτρων μεταξύ τους. Αυτό επιτυγχάνεται με τη δημιουργία μιας εξίσωσης και τη χρήση της παραμέτρου **connect()**. Μέσα σ' αυτήν γίνεται η σύνδεση όλων των παραπάνω στοιχείων μέσω

των κλάσεων και των εξισώσεων που τις συνδέουν. Αν δεν υπάρξει το συγκεκριμένο κομμάτι κώδικα, όταν ο χρήστης θα προσπαθήσει να προσομοιώσει το σύστημα θα εμφανίσει λάθη και εν τέλει δεν θα λειτουργήσει. Το κομμάτι κώδικα της κλασης τελειώνει με την εντολή **end Motor;**, που ουσιαστικά κλείνει την κλάση που αφορά αποκλειστικά το χτίσιμο του κινητήρα. Στον παρακάτω κώδικά βλέπουμε τις κλάσεις και τις μεταβλητές που έχουν δηλωθεί παραπάνω να εννώνονται όλες μεταξύ τους.

```
equation
        connect(La.n, emf.p);
        connect(Ra.n, La.p);
        connect(Rd2.n, diff.n1);
        connect(C.n, Opl.p2);
        connect(Opl.p2, power.p1);
        connect(Vs.p, Rd2.p);
       connect(diff.n1, Rd1.p);
       connect(Rd1.n, diff.p2);
       connect(diff.p2, Ri.p);
       connect(Ri.n, Opl.n1);
       connect(Opl.n1, C.p);
       connect(power.n1, Rp1.p);
       connect(power.p2, Rp1.n);
       connect(Rp1.p, Rp2.p);
       connect(power.p2, Ra.p);
       connect(Rd3.p, hall2.p);
        connect(Rd3.n, diff.p1);
       connect(Rd3.n, Rd4.p);
       connect(Vs.n, g1.p);
       connect(g2.p, hall2.n);
       connect(Rd4.n, g3.p);
       connect(g3.p, Opl.p1);
       connect(g5.p, Rp2.n);
       connect(emf.n, hall1.p);
       connect(hall1.n, g4.p);
       connect(emf.flange, phi.flange);
       connect(emf.flange, speed.flange);
```

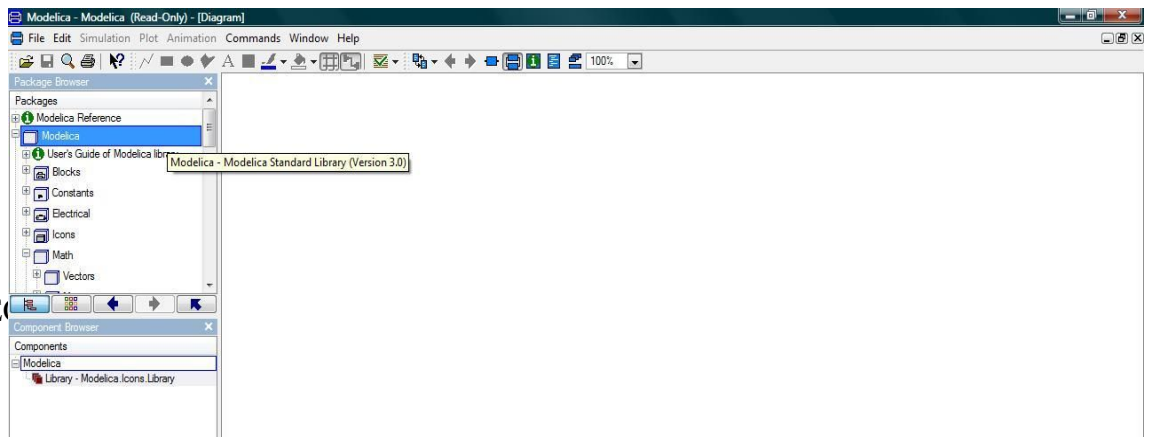
```
connect(Opl.n2, power.n2);
connect(Opl.p1, Opl.n2);
connect(Opl.p1, diff.n2);
connect(Jmotor.flange_b, flange_motor);
connect(phi.phi, axisControlBus.motorAngle);
connect(speed.w, axisControlBus.motorSpeed);
connect(hall1.i, axisControlBus.current);
connect(hall1.i, convert1.u);
connect(convert1.y, hall2.v);
connect(convert2.u, axisControlBus.current_ref);
connect(convert2.y, Vs.v);
connect(emf.flange, Jmotor.flange_a);
end Motor;
```

Όλο το παραπάνω παράδειγμα αποτελεί ένα πολύ μικρό κομμάτι κώδικα του συνολικού συστήματος. Το πλεονέκτημα όμως είναι ότι είτε μικρό είτε μεγάλο κομμάτι κώδικα, ο χρήστης δεν χρειάζεται να γράψει ούτε γραμμή. Όλος ο κώδικας θα δημιουργηθεί αυτόματα απλά «χτίζοντας» το πρότυπο. Αυτός είναι επόμενος κι ο λόγος που χαρακτηρίσαμε το DYMOLA ως ένα περιβάλλον εργασίας φιλικό προς τον χρήστη στην αρχή αυτής της εργασίας.

3.6.3.4. ΤΥΠΟΠΟΙΗΜΕΝΗ ΒΙΒΛΙΟΘΗΚΗ

Χρησιμοποιώντας τα δομικά στοιχεία από την τυποποιημένη βιβλιοθήκη Modelica, το πρότυπο ενισχύεται κι έχουμε την δυνατότητα να το επαναχρησιμοποιήσουμε για διαφορετικό σκοπό. Έτσι, δημιουργούμε διαφορετικά συστήματα με την παρόμοια περιγραφή.

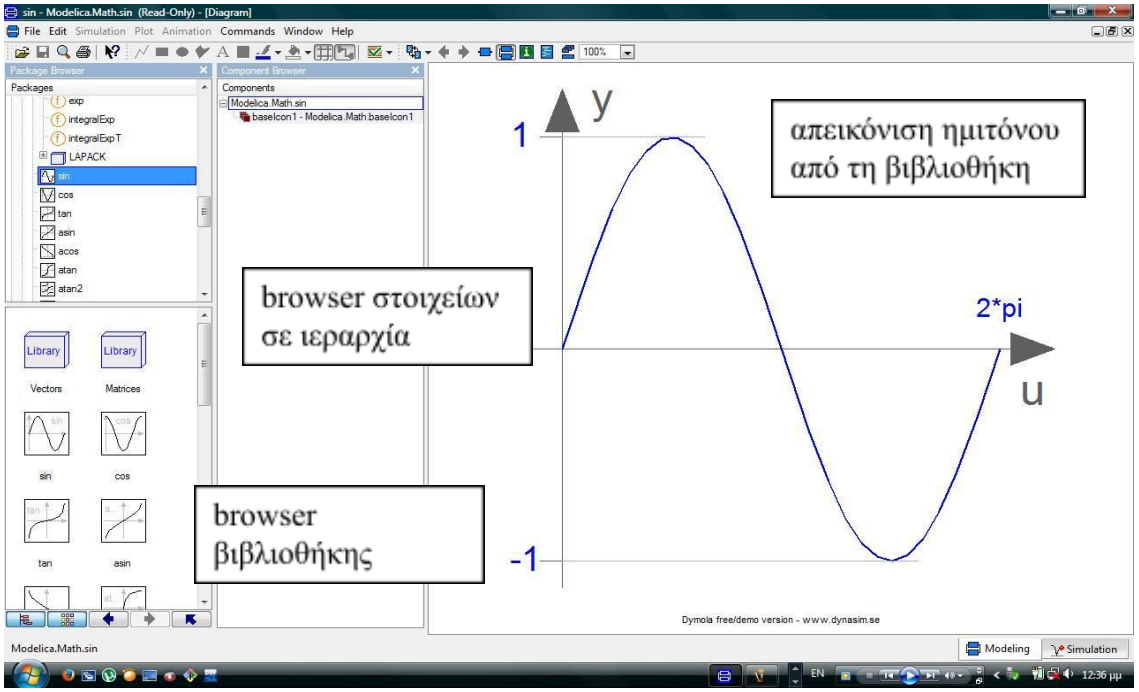
Τώρα θ' ανοίξουμε τη βιβλιοθήκη για να μελετήσουμε τα δομικά στοιχεία στα οποία έχουμε πρόσβαση καθώς και πως τα χρησιμοποιούμε. Κάνουμε διπλό κλικ στη Modelica στο παράθυρο του package browser, Εικόνα 17..



Εικόνα 17. Άνοιγμα τυποποιημένων βιβλιοθηκών Modelica.

Όπως φαίνεται στη παραπάνω εικόνα η τυποποιημένη βιβλιοθήκη είναι ιεραρχικά δομημένη σε υποβιβλιοθήκες. Αυτές οι υποβιβλιοθήκες περιλαμβάνουν τα block υποστοιχεία που είναι απαραίτητα για την δημιουργία ενός προτύπου. Το κάθε block υποστοιχείων αποτελείται από στοιχεία μιας συγκεκριμένης κατηγορίας. Παραδείγματος χάριν, έχουμε το block υποστοιχείων Constants, το οποίο περιέχει τις μαθηματικές σταθερές (π = 3,14), τις φυσικές σταθερές (G = 9,8) καθώς και τις εξαρτημένες σταθερές μηχανών. Ακόμα το block υποστοιχείων Electrical που περιέχει τα ηλεκτρικά και ηλεκτρονικά στοιχεία όπως η αντίσταση (resistor) και η δίοδος (diode). Το Math παρέχει την πρόσβαση στις μαθηματικές λειτουργίες όπως δείκτες (vectors), πίνακες (matrices) καθώς και εξισώσεις για μαθηματικές πράξεις όπως ημίτονο, συνημίτονο (sin, cos, tan) κ.τ.λ..

Μπορούμε ν' ανόξουμε ακόμα ένα παράθυρο για τις βιβλιοθήκες ώστε να έχουμε μεγαλύτερη ακρίβεια σ' αυτό που ψάχνουμε, Εικόνα 18.

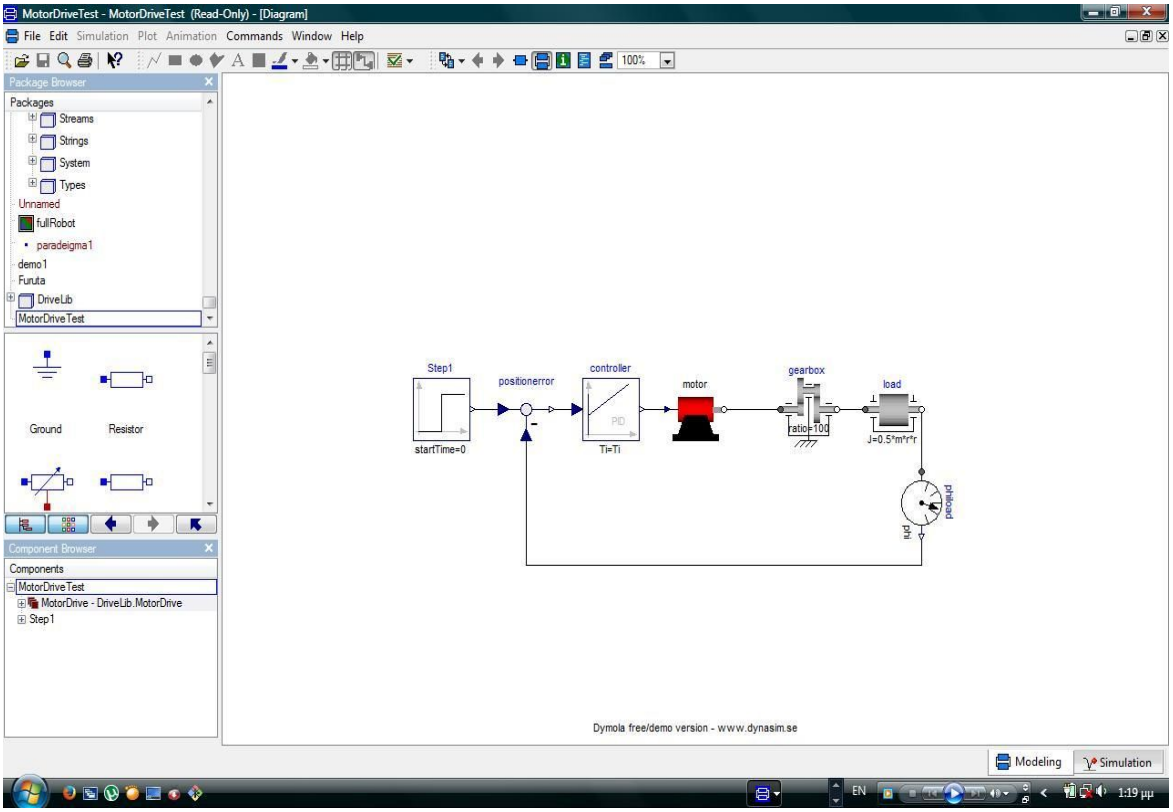


Εικόνα 18. Απεικόνιση παραθύρου βιβλιοθηκών (library browser).

Θα πάρουμε ως παράδειγμα το μοντέλο οδήγησης κινητήρα χρησιμοποιώντας συνεχη ηλεκτρική μηχανή μέσω κιβωτίου ταχυτήτων και χρησιμοποιώντας PID

controllers ως μέσο ελέγχου της μηχανής.

Στην παρακάτω εικόνα φαίνεται το παράθυρο του block διαγράμματος του προτύπου.



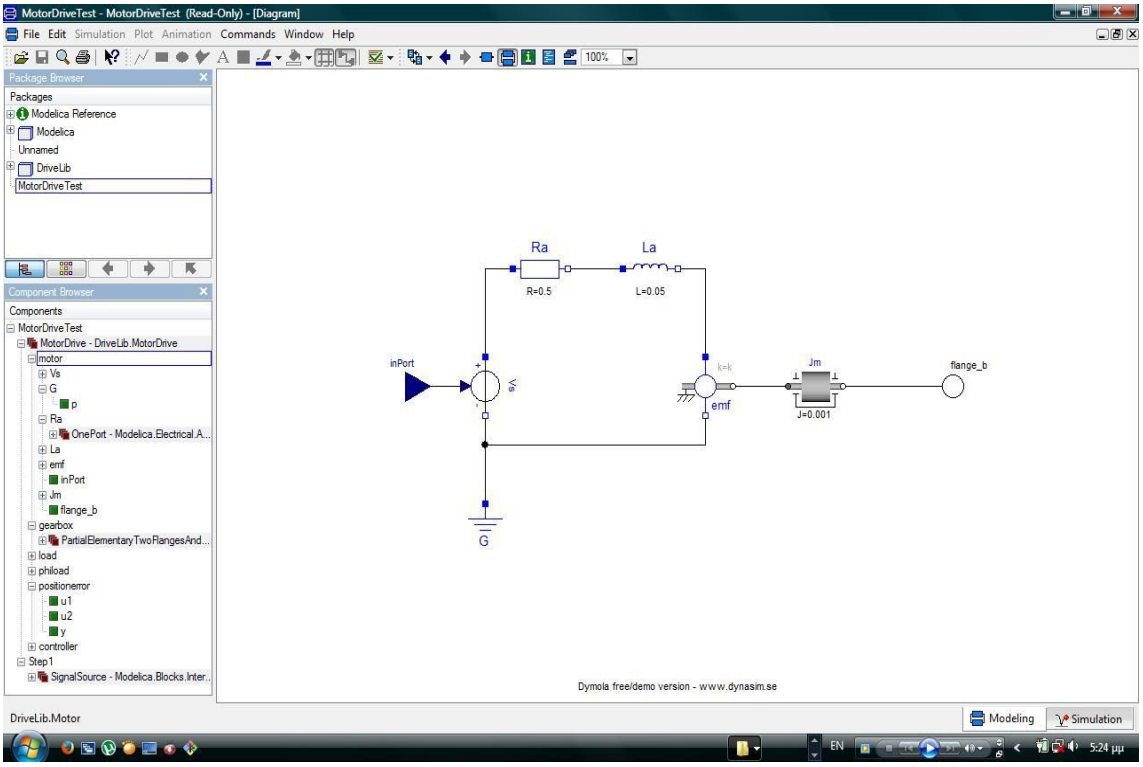
Εικόνα 19. Πρότυπο/Μοντέλο οδήγησης κινητήρα.

Βλέποντας την παραπάνω εικόνα έχουμε το μοντέλο στο οποίο έχουμε τα στοιχεία σε πλήρη διάταξη και ιεραρχία μεταξύ τους. Τα στοιχεία που το συνθέτουν το είναι τα εξής:

- το πρότυπο ξεκινάει δίνοντας μια βηματική διέγερση ως δύναμη στον κινητήρα.
- συνεχίζουμε μ' έναν ανιχνευτή λάθους της θέσης του κινητήρα. Αυτό μας βοηθάει να καταλάβουμε αν ο κινητήρας μας βρίσκεται εκεί που θέλουμε.
- ακόμα υπάρχει ένας PID controller, ελεγκτής μ' ανάδραση, ώστε να ελέγχεται η σωστή λειτουργία του κινητήρα και των αυτοματισμών που υπάρχουν στο σύστημα γενικά.
- έχουμε φυσικά τον κινητήρα, για τον οποίο έχει σχεδιαστεί το όλο σύστημα.

- το κουτί ταχυτήτων (gearbox), απαραίτητο στοιχείο για τον σωστό σχεδιασμό, σχεδίαση και λειτουργία του προτύπου και κατ' επέκταση του συστήματος.
- τα δυο τελευταία στοιχεία λειτουργούν σαν έξοδοι για τ' αποτελέσματα του συστήματος καθώς και γι' ανάδραση στον ελεγκτή και στο σύστημα εξ' ολοκλήρου.

Το DYMOLA μας δίνει την δυνατότητα να εμφανίσουμε την διάταξη κάθε στοιχείου που αποτελεί το πρότυπο/μοντέλο. Έτσι μπορούμε να σχηματίσουμε μια πιο ολοκληρωμένη άποψη του συστήματος μας (όπως κάναμε και σε προηγούμενη παράγραφο για τα εξαρτήματα του ρομποτικού βραχίονα).



Εικόνα 20. Απεικόνιση του εικονιδίου του κινητήρα καθώς και του κυκλώματος του.

Παραπάνω φαίνεται η συνδεσμολογία που απαιτείται για την δημιουργία του κινητήρα που υπάρχει στο αρχικό μεγάλο πρότυπο. Όπως και στην προηγούμενη ενότητα με τον ρομποτικό βραχίονα, έτσι κι εδώ εμφανίζουμε τα υποστοιχεία που αποτελούν κάθε στοιχείο του συστήματος. Στη συγκεκριμένη περίπτωση έχουμε εμφανίσει τα υποστοιχεία που συνθέτουν τον κινητήρα. Όσο πολύπλοκο ένα σύστημα και να είναι πάντα ξεκινάει στηριζόμενο σε βασικές αρχές και συνδεσμολογίες. Επομένως βλέπουμε τον κινητήρα ν' αποτελείται από αντιστάσεις,

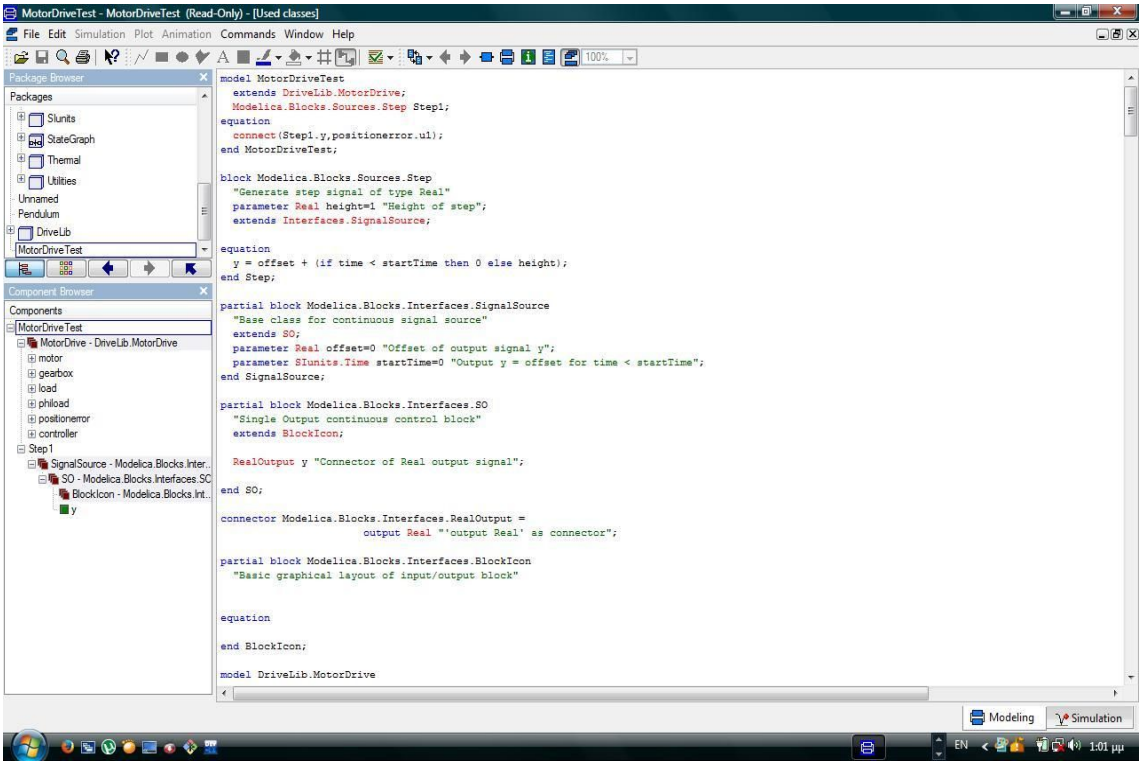
πηνία, κ.τ.λ., καθώς και τον τρόπο που είναι αυτά συνδεδεμένα μεταξύ τους.

Κοιτώντας το παράθυρο του browser των στοιχείων έχουμε την λίστα των στοιχείων του μοντέλου/προτύπου, καθώς και την ιεραρχία που αυτά είναι δομημένα μεταξύ τους.

Το παραπάνω τυποποιημένο πρότυπο λόγω της ευελιξίας του και της τυποποιημένης προσέγγισης του, μπορεί να χρησιμοποιηθεί σε πολλά διαφορετικά συστήματα που όμως έχουν παρόμοια φιλοσοφία στη σύνθεσή τους. Για παράδειγμα, μπορούμε να το χρησιμοποιήσουμε για την σχεδίαση κινητήρα που θα χρησιμοποιηθεί σε απλά οχήματα αλλά κάνοντας σαφώς ιο περίπλοκες τροποποιήσεις μπορεί να χρησιμοποιηθεί και για κατασκευή αεροσκαφών. Όπως γίνεται σαφές, τα συστήματα αυτά μπορεί να διαφέρουν στην πολυπλοκότητα, στη σχεδίαση και στον τρόπο υλοποίησης, όμως έχουν κοινό παρανομαστή λειτουργίας, την χρήση και την οδήγηση κινητήρων.

3.6.3.5 MODELICA LANGUAGE

Σ' αυτήν την ενότητα θα αναλύσουμε τον κώδικά της Modelica για τον κινητήρα. Παρακάτω φαίνεται ο κώδικας σε γλώσσα Modelica καθώς και οι κλάσεις που έχουν δημιουργηθεί αυτόματα για το έτοιμο παράδειγμα, Εικόνα 21..



Εικόνα 21. Ο κώδικας σε Modelica για τον κινητήρα.

Ρίχνοντας μια πιο προσεκτική ματιά στις κλάσεις του μοντέλου, μια – προς – μια, αντιλαμβανόμαστε τον τρόπο που δομούνται τα στοιχεία μέσα σ' αυτό.

Για παράδειγμα παίρνουμε την πρώτη κλάση:

```
model MotorDriveTest
  extends DriveLib.MotorDrive;
  Modelica.Blocks.Sources.Step Step1;
equation
  connect(Step1.y,positionerror.u1);
end MotorDriveTest;
```

- έχουμε τον ορισμό του μοντέλου: model MotorDriveTest, το οποίο επεκτείνεται με τη χρήση της βιβλιοθήκης: DriveLib.MotorDrive; και εφαρμόζει βηματική διέγερση ως είσοδο στο σύστημα.
- συνεχίζοντας έχουμε τη συνάρτηση της σύνδεσης της διέγερσης: Step1, με την είσοδο του ανιχνευτή positionerror.u1 και οδηγώντας στην έξοδο Step1.y στο επόμενο κομμάτι του συστήματος.

Με την ίδια λογική αναπτύσσονται κι όλες οι υπόλοιπες κλάσεις του παραδείγματός μας. Προχωρώντας, οι κλάσεις γίνονται πιο σύνθετες, καθώς χτίζονται οι συναρτήσεις μεταξύ των στοιχείων, δίνοντας για παράδειγμα τις παραμέτρους και τις τιμές αυτών αλλά και τη μεταξύ τους σύνδεση, για την ομαλή λειτουργία του μοντέλου.

3.6.4. ΔΗΜΙΟΥΡΓΙΑ ΜΟΝΤΕΛΩΝ ΜΕ ΤΗΝ ΒΟΗΘΕΙΑ ΤΟΥ DYMOLA

Μέχρι στιγμής έχουμε ασχοληθεί με τις επιφάνειες εργασείας του εργαλείου καθώς και με το τι μας παρέχει η καθεμία. Επίσης έχουμε δει την χρήση μοντέλων που ήδη υπάρχουν σ' αυτό και τον τρόπο που μπορούμε να τα χρησιμοποιήσουμε τόσο για εξοικίωση όσο και για το πώς κομμάτια τους θα μπορούσαν να χρησιμοποιηθούν σε καινούργιο project.

Στην ενότητα που ακολουθεί θα επικεντρωθούμε στη δημιουργία προτύπων και

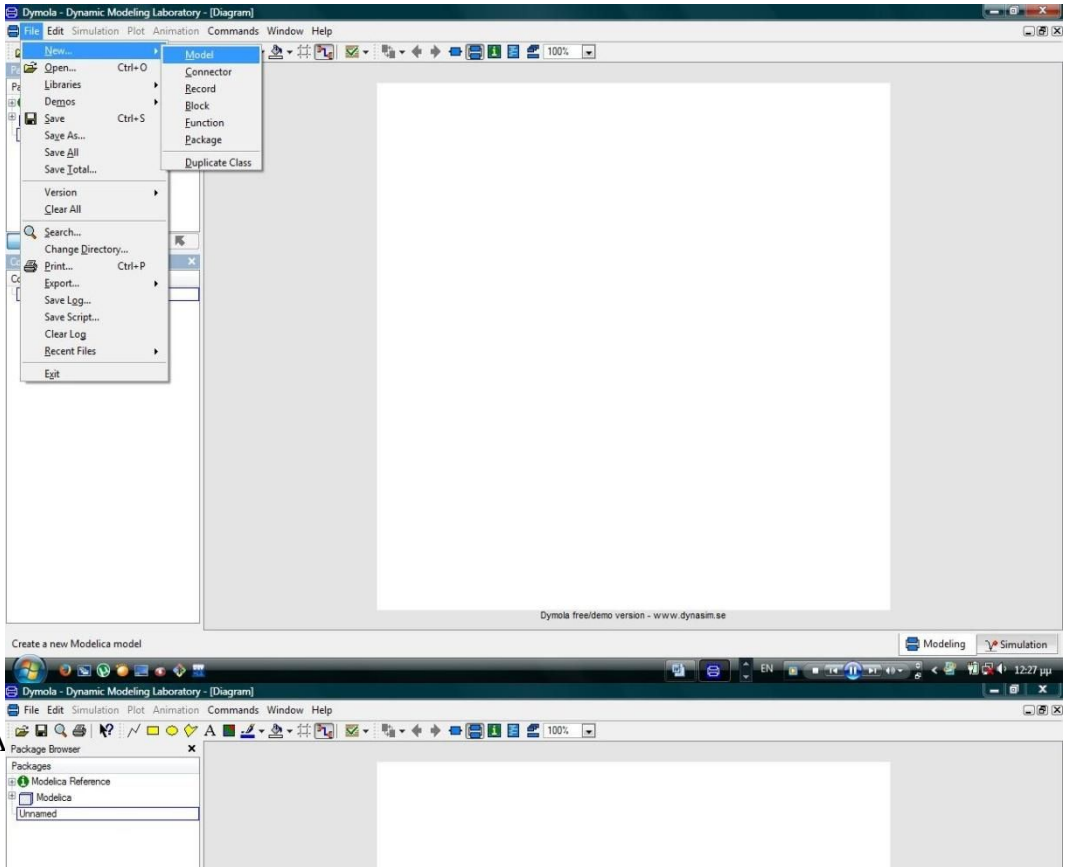
μοντέλων με την βοήθεια του DYMOLA και αργότερα θα συγκρίνουμε με το MatLab.

3.6.4.1.ΤΕΛΕΣΤΙΚΟΣ ΕΝΙΣΧΥΤΗΣ ΜΕ ΑΝΑΔΡΑΣΗ

Με την βοήθεια του DYMOLA θα δημιουργήσσουμε ένα μοντέλο τελεστικού ενισχυτή, στον οποίο θα περιλαμβάνεται συνδεσμολογία κλειστού βρόγχου και έτσι το σύστημά μας θα περιέχει ανάδραση.

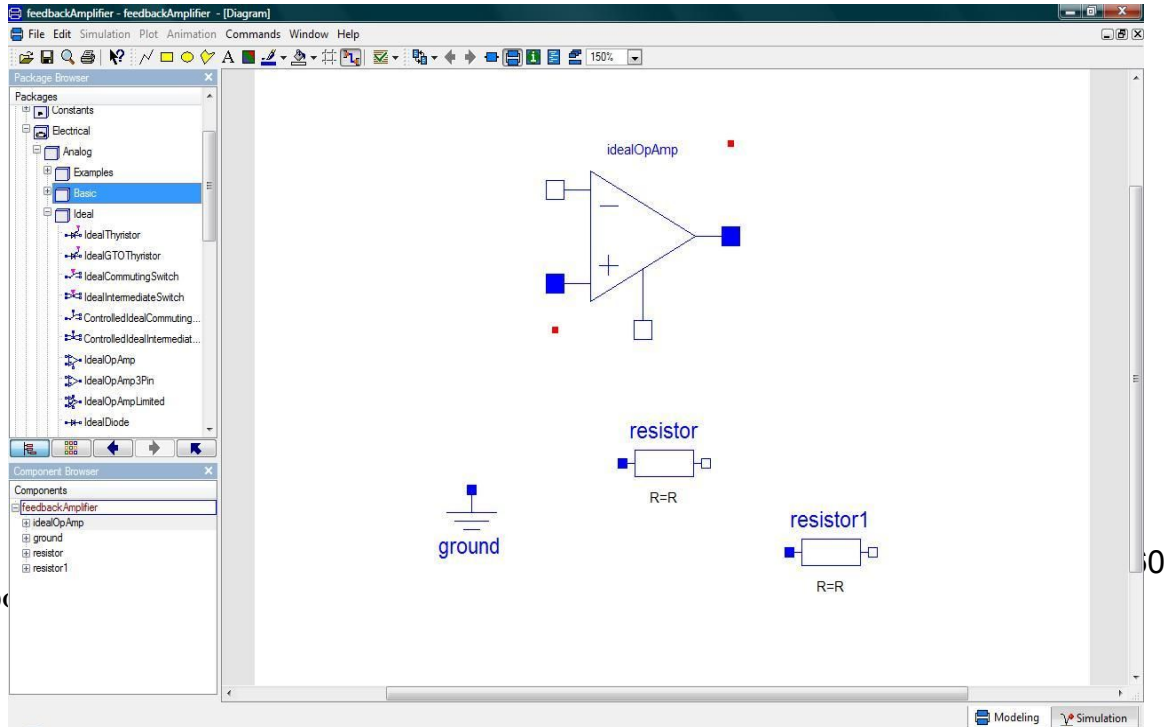
Το κύριο ηλεκτρικό στοιχείο του μοντέλου μας είναι ένας ιδανικός τελεστικός ενισχυτής. Σ' αυτόν στην συνέχεια προσθέτουμε ένα δίκτυο ωμικής αντίστασης, το οποίο αποτελείται από δύο αντιστάτες 10kOhm και 90kOhm. Το ζητούμενό μας εδώ είναι το συστημά μας να δίνει στην έξοδο του τελεστικού ενισχυτή κέρδος ρεύματος 10 (dc gain = 10). Αργότερα, θα προσθέσουμε στο σύστημα έναν πυκνωτή C ο οποίος ισούται με 2 pF. Ο πυκνωτής θα είναι τοποθετημένος σε παράλληλη διάταξη με τον αντιστάτη της ανάδρασης που είναι συνδεδεμένος με την έξοδο. Με αυτήν την συνδεσμολογία η ανάδραση οδηγεί σε αντιστάθμιση του συστήματος ώστε να έχουμε όσο το δυνατό τιμή κέρδους κοντά στην επιθυμητή.

Ξεκινώντας το DYMOLA, δημιουργούμε ένα καινούργιο μοντέλο το οποίο ονομάζουμε feedbackAmplifier. Από το menu File επιλέγουμε New → Model και δίνουμε όνομα στη καρτέλα που εμφανίζεται. Η διαδικασία φαίνεται στις παρακάτω εικόνα, Εικόνα 22..



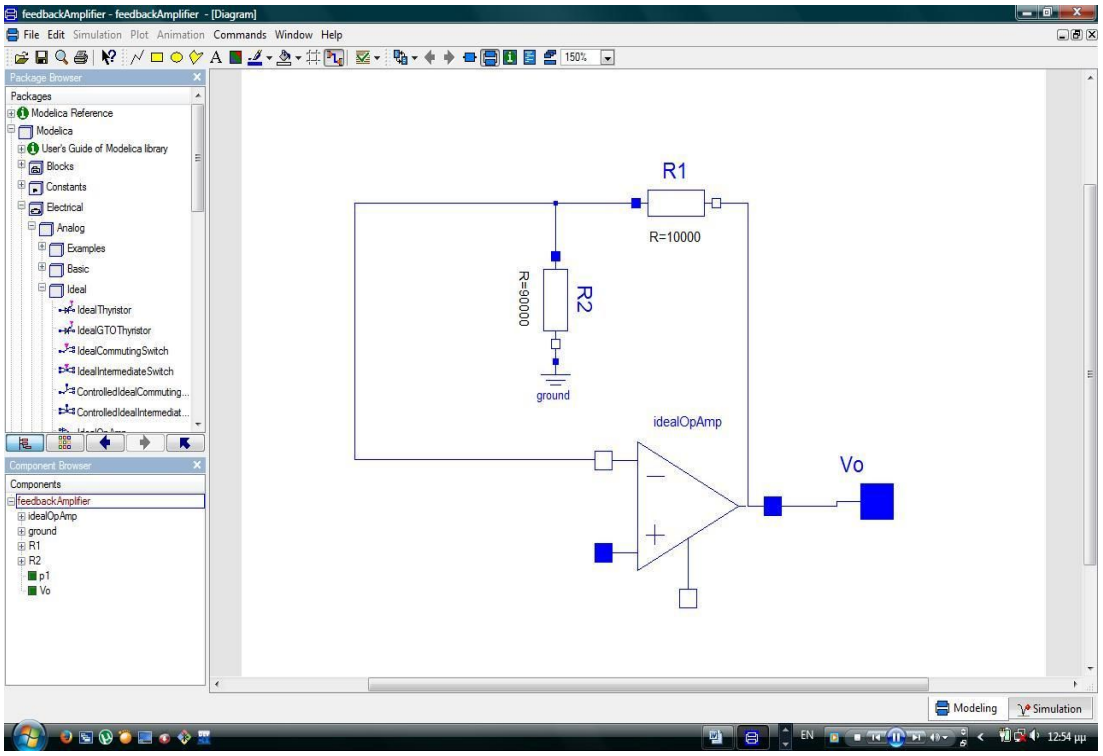
Εικόνα 22. Δημιουργία μοντέλου τελεστικού ενισχυτή με ανάδραση.

Στη συνέχεια δημιουργούμε το ζητούμενο μοντέλο. Για να γίνει αυτό, πρέπει να «σύρουμε» τα απαιτούμενα block στοιχεία από τις βιβλιοθήκες της Modelica στο block διάγραμμα σχεδίασης. Ανοίγουμε την βιβλιοθήκη Electrical → Analog → Basic και Ideal και επιλέγουμε τον ιδανικό τελεστικό ενισχυτή, δυο αντιστάσεις καθώς και γείωση.



Εικόνα 23. Επιλογή των στοιχείων του προτύπου από τις βιβλιοθήκες.

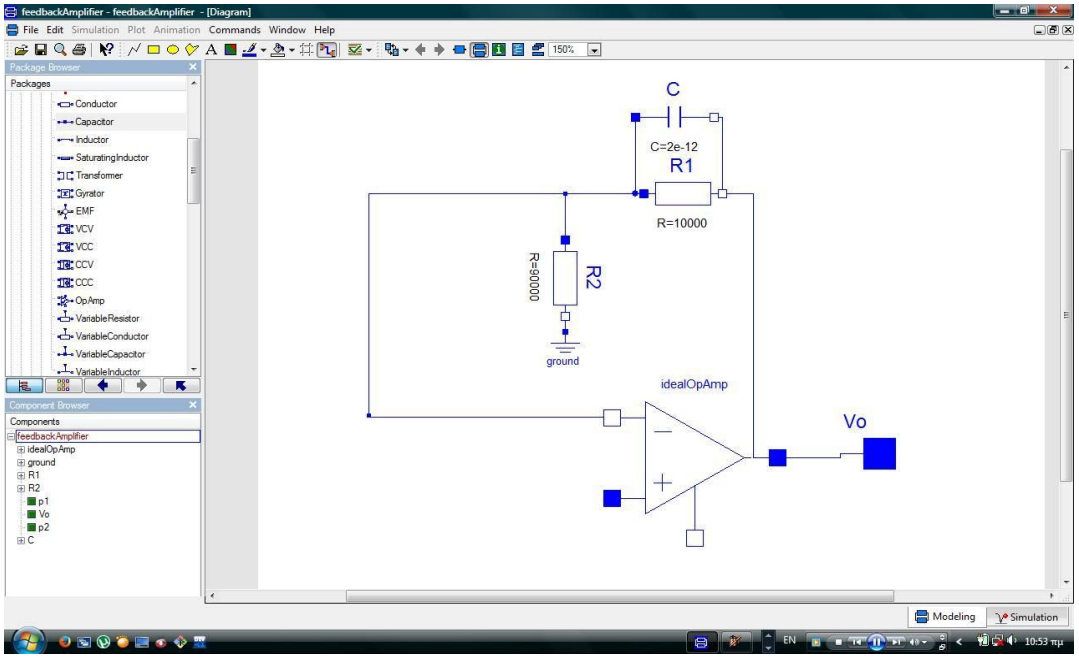
Στην συνέχεια ενώνουμε τα στοιχεία μεταξύ τους όπως φαίνεται στη παρακάτω εικόνα, Εικόνα 3.11.. Για να δουλεύει το συστημά μας κατά τρόπο επιθυμητό, οι τιμές που δίνονται στα στοιχεία είναι R1 = 10000kOhm και R2 = 90000kOhm



Εικόνα 24. Συνδεσμολογία τελεστικού με ανάδραση.

Στη συνδεσμολογία του κλειστού βρόγχου της ανάδρασης, προσθέτουμε έναν πυκνωτή και μετατρέπουμε το σύστημα σ' έναν τελεστικό ενισχυτή ανάδρασης με αντιστάθμιση. Ο πυκνωτής είναι συνδεδεμένος πάραλληλα με τον έναν από τους διαιρέτες τάσης της ανάδρασης και σε σειρά με τον άλλο και για να αποδίδει το

σύστημα θα πρέπει να έχει τιμή στα 2pF, Εικόνα 25..

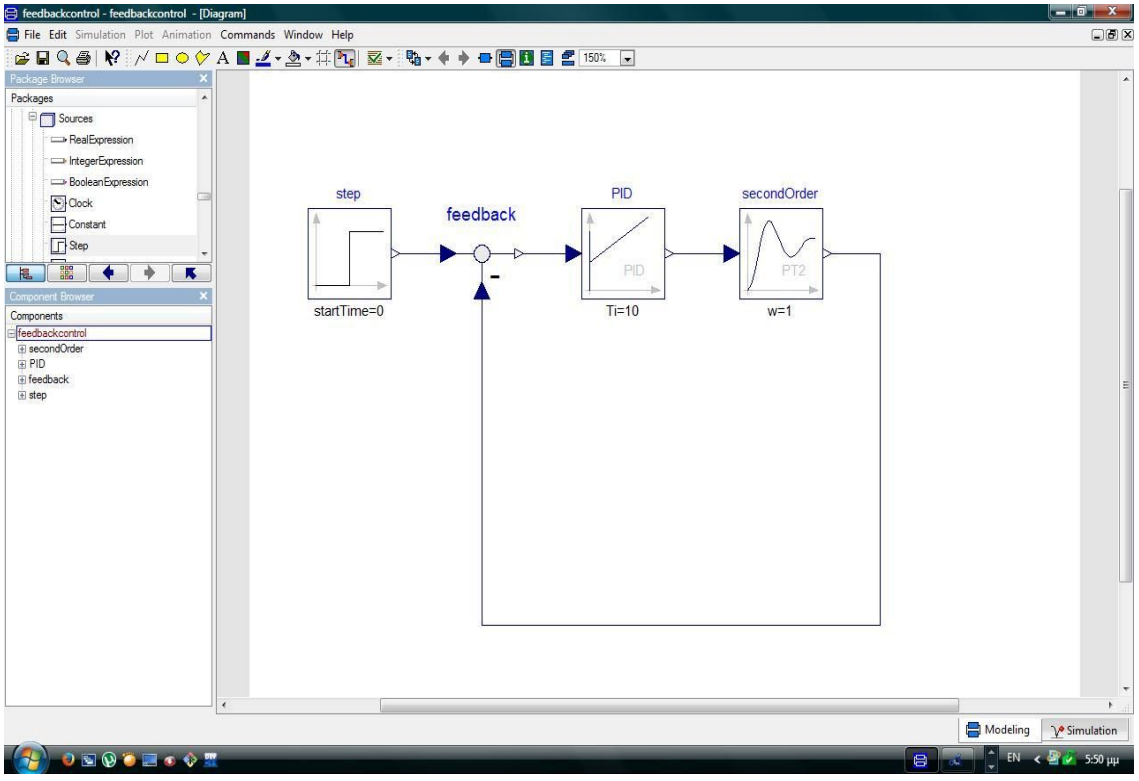


Εικόνα 25. Τελεστικός ενισχυτής με ανάδραση που οδηγεί σε αντιστάθμιση του συστήματος.

3.6.4.2. ΣΥΣΤΗΜΑ ΕΛΕΓΧΟΥ ΜΕ ΑΝΑΔΡΑΣΗ (FEEDBACK CONTROL SYSTEM)

- **Δημιουργία block διαγράμματος:**

Στην παρακάτω ενότητα θα σχεδιάσουμε ένα σύστημα ελέγχου με ανάδραση το οποίο θα αντιδρά στην οποιαδήποτε αλλαγή. Στην παρακάτω εικόνα, Εικόνα 26., φαίνεται το block διάγραμμα του συστήματος.



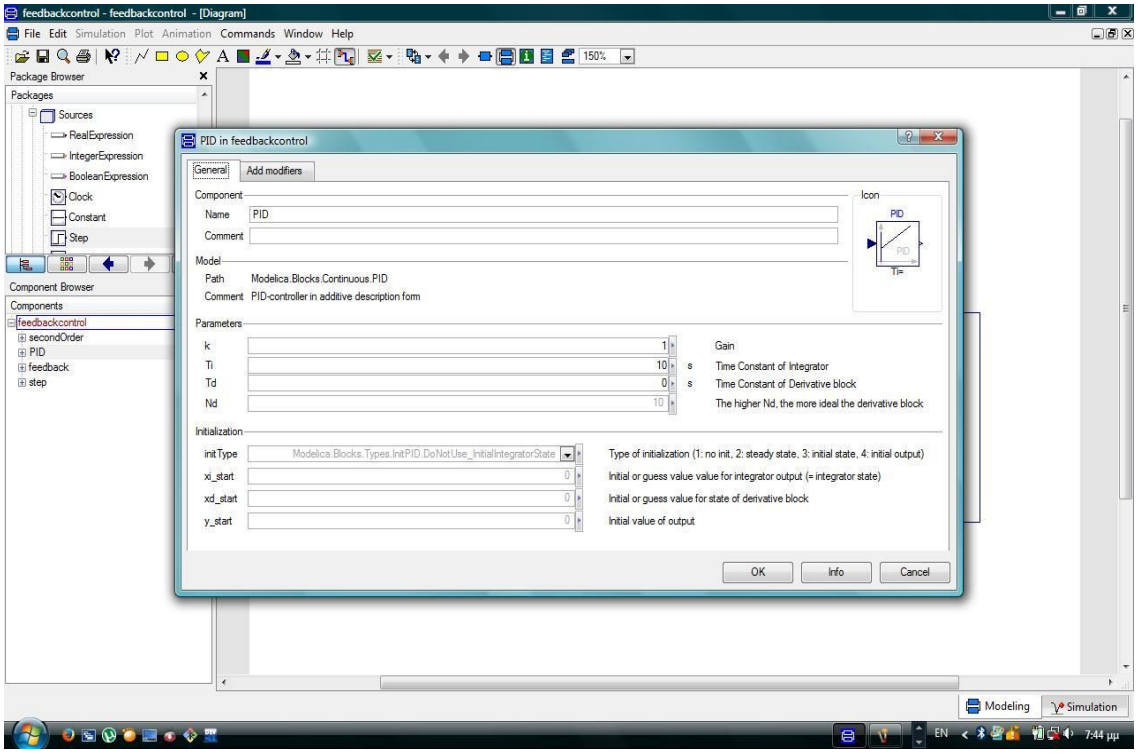
Εικόνα 26. Σύστημα ελέγχου με ανάδραση.

Για να φτάσουμε στο αποτέλεσμα που φαίνεται παραπάνω θα πρέπει να εισάγουμε από την βιβλιοθήκη τα δομικά στοιχεία που απαιτούνται. Επομένως, στο αριστερό παράθυρο πακέτων επιλέγουμε από την βιβλιοθήκη τη καρτέλα Blocks. Όλα τα δομικά στοιχεία για την σχεδίαση του μοντέλου μας βρίσκονται σ' αυτήν. Άρα έχοντα επιλέξει την καρτέλα Blocks, απ' αυτήν «σέρνουμε» στο block διάγραμμα τα εξής στοιχεία:

- Blocks → Sources → Step, εισαγωγή της βηματικής εισόδου.
- Blocks → Math → Feedback, εισαγωγή της ανάδρασης.
- Blocks → Continuous → PID, εισαγωγή του ελεγκτή
- Blocks → Continuous → SecondOrder, εισαγωγή του block δεύτερης τάξης ως έξοδος.

Αναλυτικότερα, στο block διάγραμμα η βηματική απόκριση δίνεται σαν είσοδος για τον ελεγκτή PID του συστήματος. Επίσης χρησιμοποιούμε και ένα block δεύτερης τάξης σαν έξοδο για άμεσα αποτελέσματα.

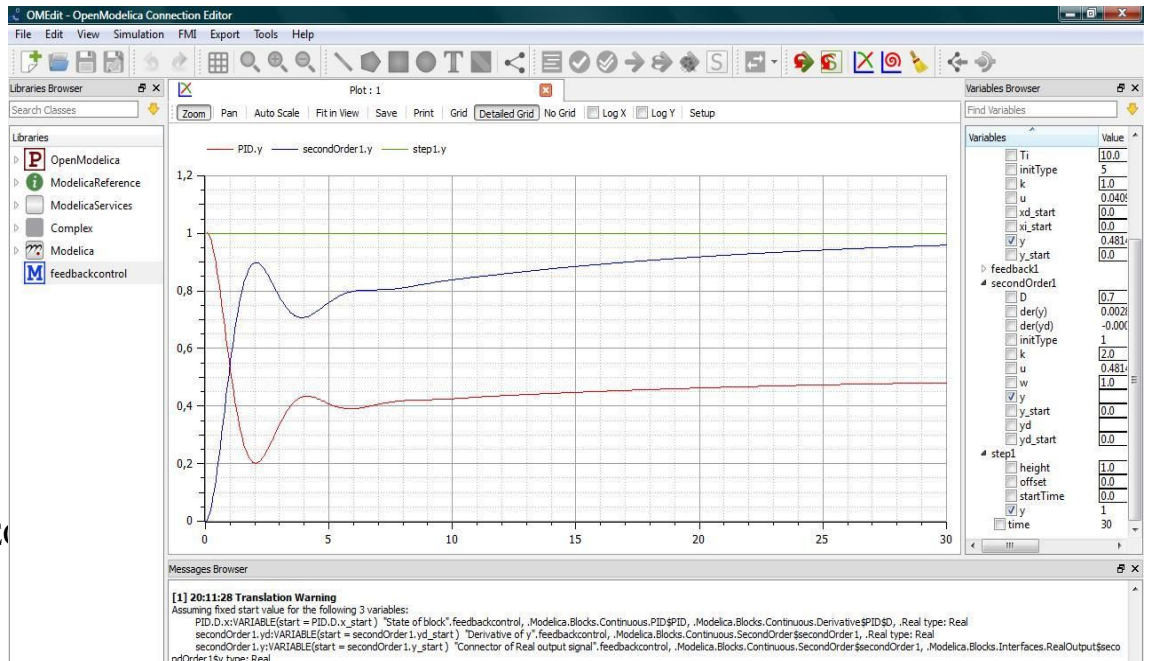
Αφού έχουμε εισάγει κι ενώσει τα στοιχεία δίνουμε τις απαραίτητες τιμές στις μεταβλητές του καθ' ενός κάνοντας διπλό κλικ πάνω στο δομικό στοιχείο. Για παράδειγμα, η Εικόνα 27., δείχνει τις τιμές των παραμέτρων που θέτουμε για τον PID ελεγκτή. Έχουμε ορίσει το κέρδος ίσο με 1 και τις τιμές των T_i και T_d που αντιστοιχούν στο χρόνο των σταθερών, ίσες με 10 και μηδέν αντίστοιχα.



Εικόνα 27. Τιμές των παραμέτρων του PID Controler.

- **Προσομοίωση του μοντέλου:**

Στη συνέχεια θα προσομοιώσουμε το μοντέλο που δημιουργήσαμε και θα μελετήσουμε πως συμπεριφέρεται στην έξοδο.



Εικόνα 28. Προσομοίωση μοντέλου.

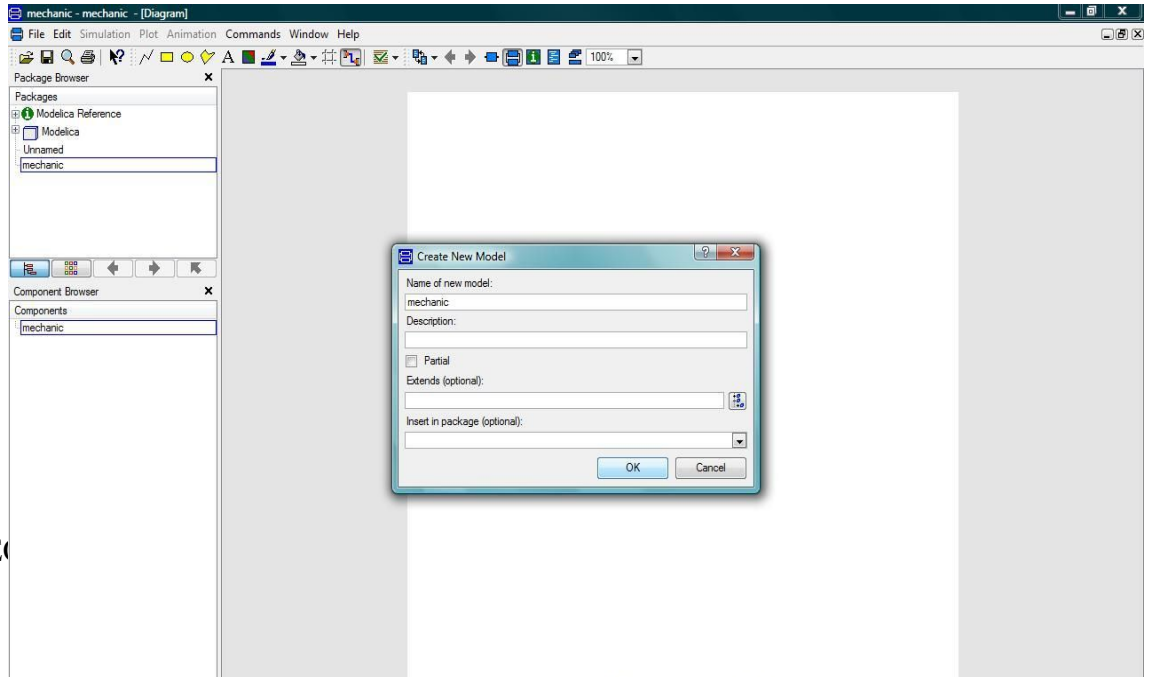
Στην παραπάνω εικόνα, Εικόνα 28., φαίνεται πως θα πρέπει να συμπεριφερθεί το σύστημά μας στον «πραγματικό» κόσμο. Η βηματική απόκριση είναι η είσοδος του συστήματος μας και το feedback κινείται με βάση αυτήν. Επειδή το σύστημά έχει ανάδραση η έξοδος του συστήματος ενισχύεται σε σχέση με την είσοδο αλλά και την έξοδο, καθώς ένα κομμάτι αυτής επιστρέφει στο σύστημα, ανατροφοδοτώντας το.

3.6.4.3. ΔΗΜΙΟΥΡΓΙΑ ΚΑΙ ΠΡΟΣΟΜΟΙΩΣΗ ΜΗΧΑΝΙΚΟΥ ΚΥΚΛΩΜΑΤΟΣ

- **ΔΗΜΙΟΥΡΓΙΑ ΠΡΟΤΥΠΟΥ (MODELING) :**

Θα σχεδιάσουμε και θα προσομοιώσουμε ένα μηχανικό κύκλωμα κινητήρα δύο μαζών αδράνειας, οι οποίες συνδέονται μ' έναν ομοαξονικό μοχλό και τροφοδοτούνται με μια τροχοπέδη η οποία ασκεί ροπή του ζεύγους δυνάμεων στην κάθε πλευρά κυλίνδρου ώστε να γυρίζει.

Αρχικά ξεκινάμε χτίζοντας το μοντέλο μας βήμα βήμα. Δημιουργούμε ένα καινούργιο πρότυπο τ' οποίο ονομάζουμε mechanic, Εικόνα 29..

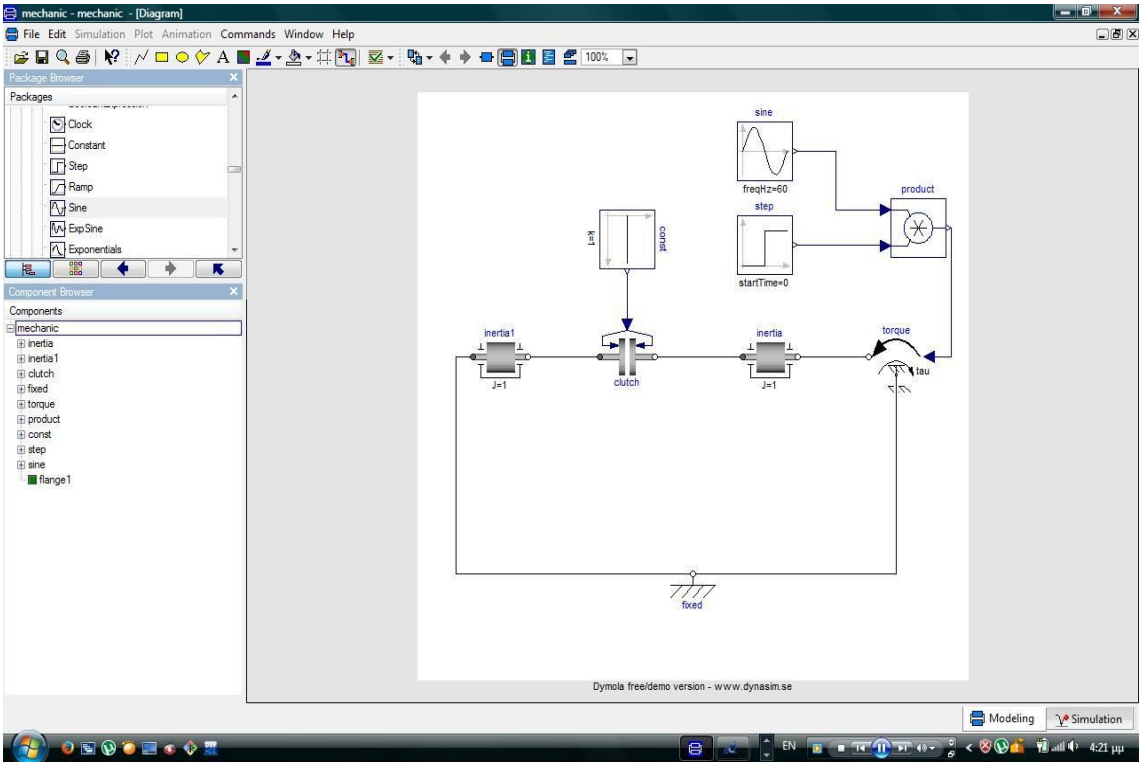


Εικόνα 29. Δημιουργία προτύπου mechanic.

Έχοντας δημιουργήσει το αρχείο μας, θα εισάγουμε τα απαραίτητα δομικά στοιχεία για την δημιουργία του προτύπου.

Ανοίγουμε την βιβλιοθήκη της modelica που βρίσκεται στ' αριστερά, στο παράθυρο των πακέτων. Από την καρτέλα mechanics επιλέγουμε Rotational → Components και εισάγουμε δύο φορές το εικονίδιο inertia (σώμα αδράνειας) και μια φορά το δομικό στοιχείο clutch (μοχλός σύνδεσης) καθώς και το εικονίδιο fixed (γείωση). Στη συνέχεια επιλέγουμε από την καρτέλα sources τα δομικά στοιχεία torque (ροπή δυνάμεων) και const (σταθερά). Τέλος από την καρτέλα Blocks → Math επιλέγουμε τα εικονίδια product (έξοδος), sine (σήμα ημιτόνου ως είσοδος) και step (βηματική είσοδος).

Επομένως τώρα που έχουμε εισάγει όλα τα δομικά στοιχεία που αποτελούν το μοντέλο μας, τα ενώνουμε μεταξύ τους ώστε να μπορέσουμε να το προσομοιώσουμε και να δούμε πως συμπεριφέρεται. Η Εικόνα 30. απεικονίζει το μοντέλο μας με τα στοιχεία συνδεμένα μεταξύ τους.



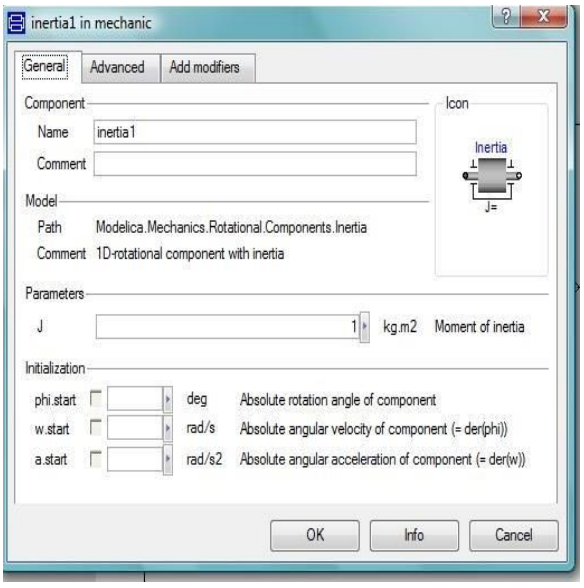
Εικόνα 30. Το μοντέλο του μηχανικού κυκλώματος σχεδιασμένο στο DYMOLA.

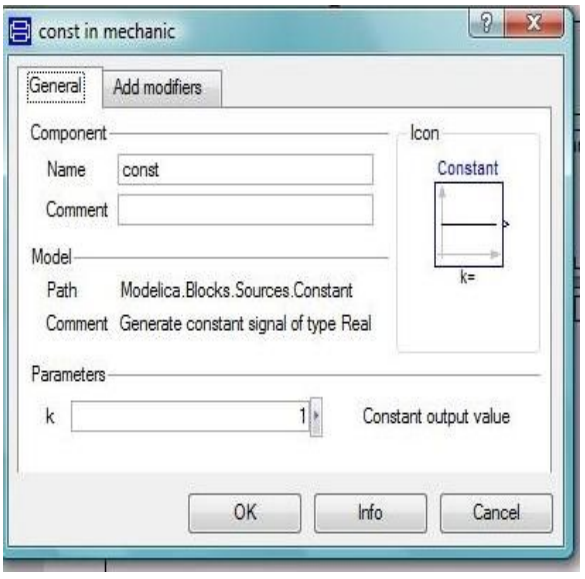
Όπως βλέπουμε παραπάνω έχουμε δυο κυλινδρικά σώματα αδράνειας (inertia 1&2) συνδεδεμένα μεταξύ τους μέσω ενός μοχλού (clutch).

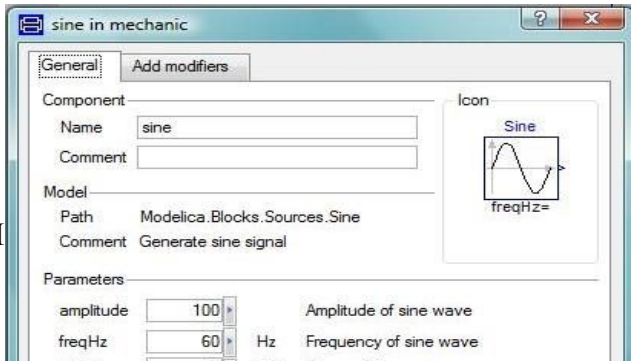
Σαν είσοδος δίνονται δύο σήματα, μια βηματική είσοδος η οποία πρακτικά λειτουργεί σαν παλμός και μια είσοδος ημιτόνου (sine) στην οποία δίνουμε συγκεκριμένο χρόνο εκκίνησης, πλάτος και διάρκεια. Οι δύο είσοδοι συνδιάζονται σε μια έξοδο μέσω του δομικού στοιχείου product, η οποία με την σειρά της δίνεται ως σήμα είσοδου στην ροπή (torque) του συστήματος ώστε ν' ασκηθεί δύναμη και να κινηθούν οι δυο κυλινδροι που αποτελούν τον κινητήρα.

Στη συνέχεια δίνουμε τις απαραίτητες τιμές στις εισόδους και την σταθερά του συστηματός μας. Για τα δύο σώματα της αδράνειας η παράμετρος ισούται με J = 1kg / m^2 και η οποία αντιστοιχία στην στιγμή της αδράνειας. Η τιμή της σταθεράς του μοχλού είναι k = 1. Τέλος, ορίζουμε τις τιμές για τα σήματα εισόδου sine και step. Η step όπως είπαμε είναι η βηματική είσοδος και η τιμή του παλμού ισούται με 1. Για το ημιτονοειδές σήμα έχουμε το πλάτος παλμού να είναι amplitude = 100 και την συχνότητα του freq = 60 Hz.

Για να δώσουμε τις αντίστοιχες τιμές στο καθ' ένα στοιχείο κάνουμε διπλό κλικ πάνω του. Στην εικόνα, Εικόνα 31., που ακολούθει φαίνονται οι τιμές των δομικών στοιχείων.







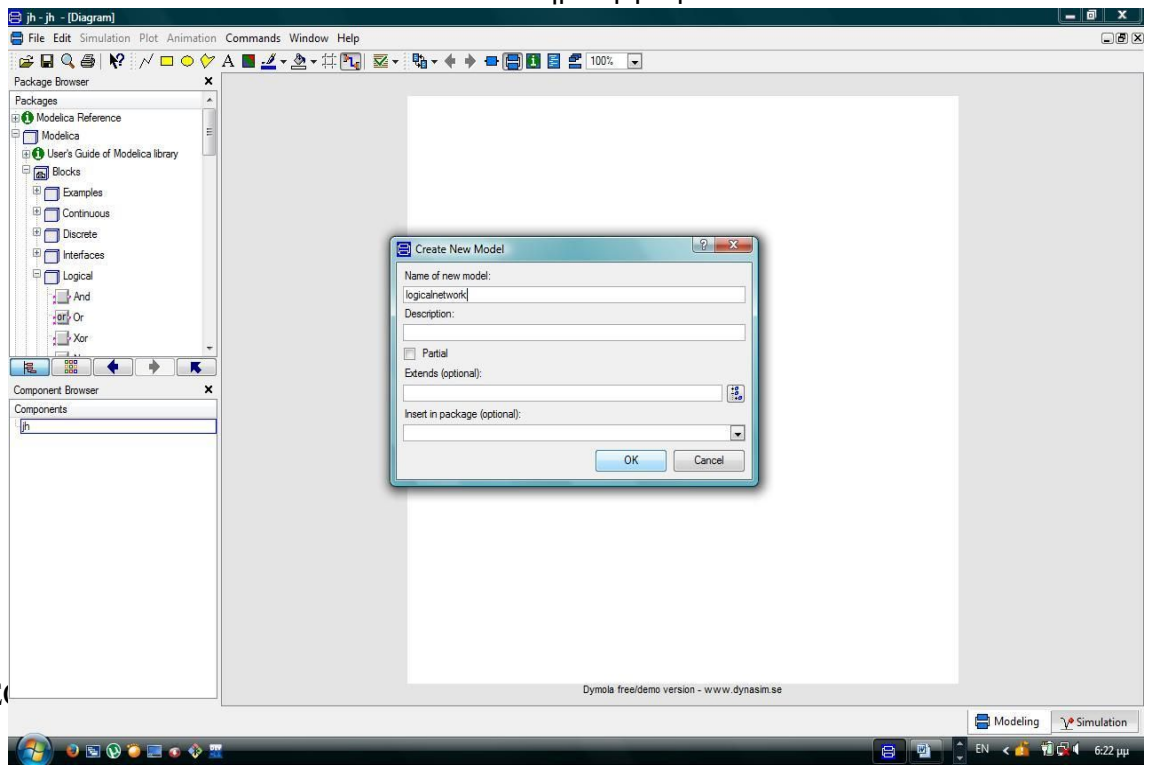
Εικόνα 31. Οι τιμές των παραμέτρων των δομικών στοιχειών του μοντελού όπως δηλώνονται στο DYMOLA.

3.6.4.4. ΔΗΜΙΟΥΡΓΙΑ ΛΟΓΙΚΟΥ ΚΥΚΛΩΜΑΤΟΣ (ΧΡΗΣΗ ΓΙΑ ΜΑΘΗΜΑΤΙΚΕΣ ΠΡΑΞΕΙΣ)

Το DYMOLA εκτός από πολύπλοκα μηχανικά, μηχανολογικά κ.α. συστήματα μπορεί να χρησιμοποιηθεί και για σχεδίαση και κατασκευή λογικών κυκλωμάτων τα οποία μπορούν να κάνουν μαθηματικές πράξεις Boolean μεταβλητών. Τέτοιου είδους κυκλώματα χρησιμοποιούνται στους υπολογιστές ως μετρητές στην μνήμη τους.

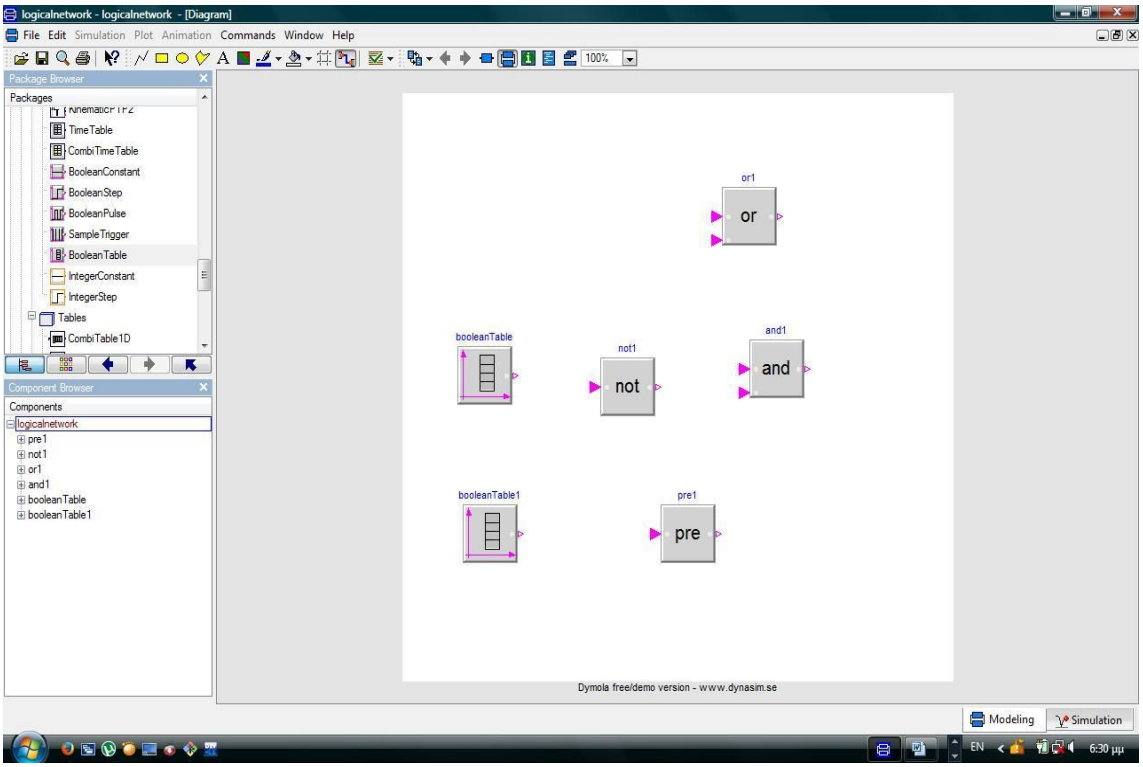
Ξεκινάμε, όπως πάντα, ορίζοντας ένα καινούργιο μοντέλο και δίνοντάς του όνομα. Ονομάζουμε το μοντέλο μας *logicalnetwork* και πατάμε ok ώστε να δημιουργηθεί.

Εικόνα 32. Δημιουργία μοντέλου.



Στη συνέχεια εισάγουμε τα δομικά στοιχεία που θ' αποτελέσουν το σύστημα μας. Αρχικά θα πρέπει να εισάγουμε τα στοιχεία που αντιστοιχούν στις Boolean εισόδους του συστήματος, δηλαδή τους δυο πίνακες Boolean, Table1 και Table2.

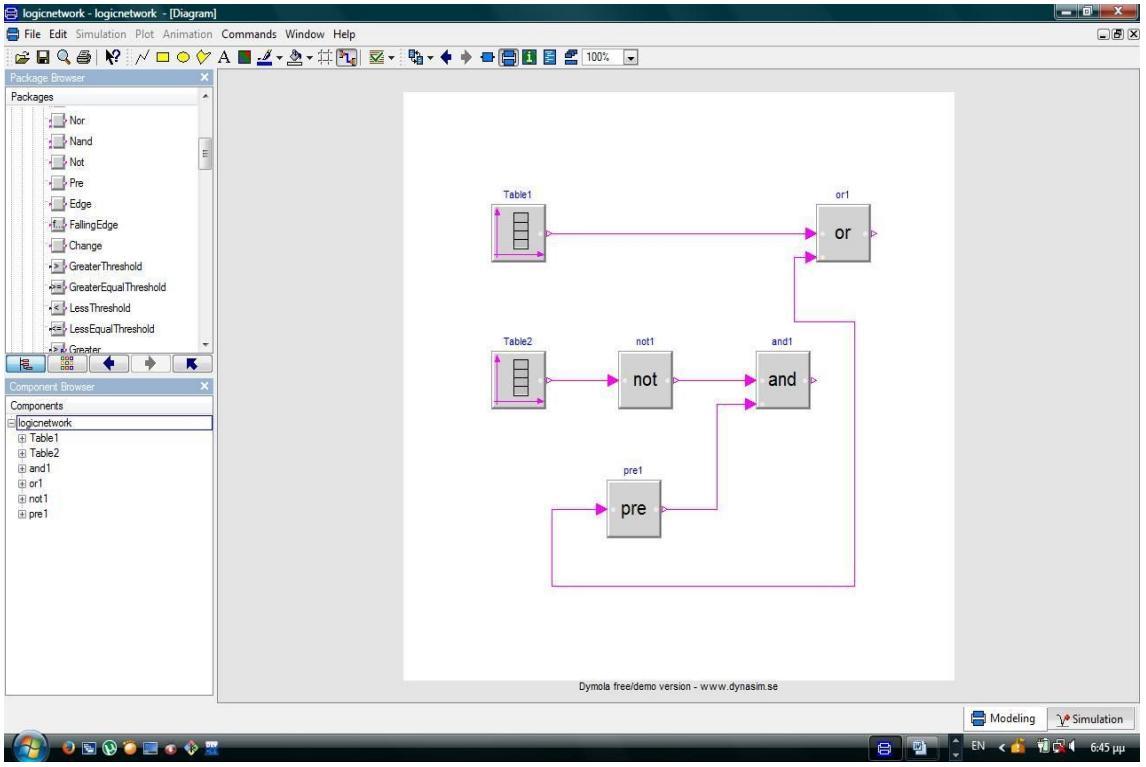
Επόμενο βήμα είναι να εισάγουμε τα δομικά στοιχεία των πράξεων. Στο παραδειγμά μας οι πράξεις που θα υλοποιούνται θα είναι οι and, or και not καθώς και το δομικό στοιχείο pre το οποίο διακόπτει συνεχόμενες loopες πράξεων για



συγκεκριμένο χρονικό διάστημα. Στην Εικόνα 33. που ακολουθεί φαίνονται τα δομικά στοιχεία που εισάγουμε για να υλοποιήσουμε το προτυπό μας.

Εικόνα 33. Τα δομικά στοιχεία που αποτελούν το σύστημα.

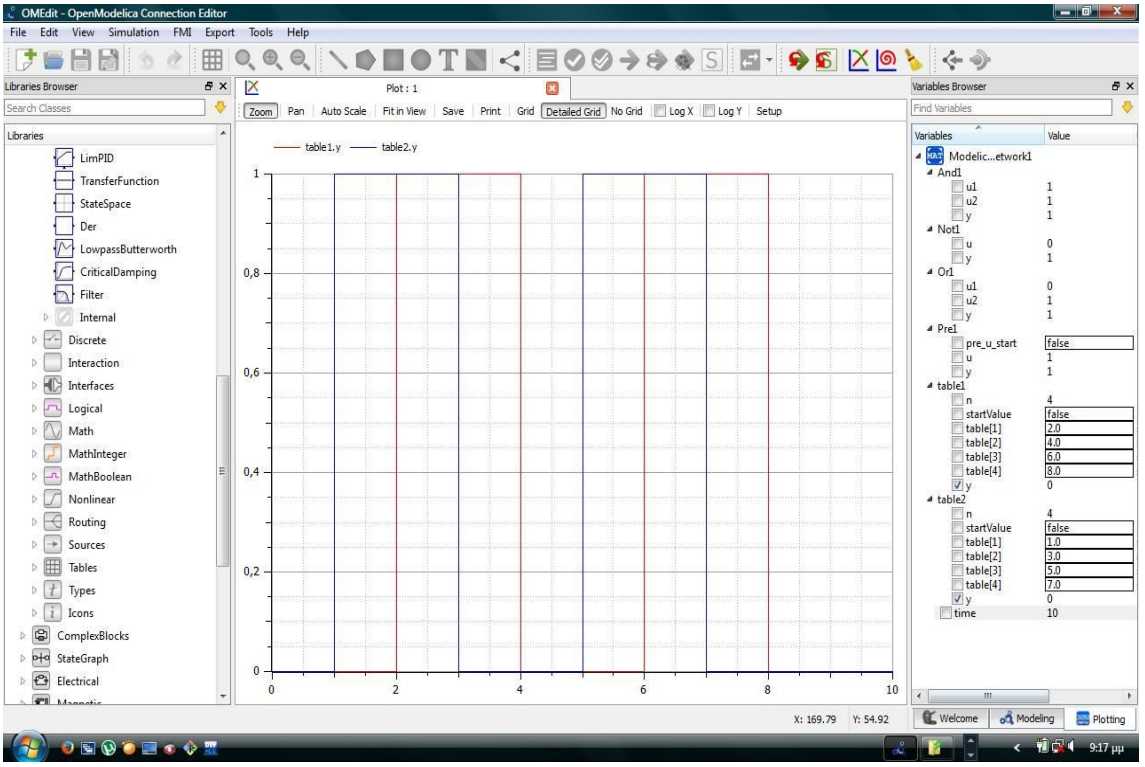
Στη συνέχεια συνδέουμε τα δομικά στοιχεία μεταξύ τους, Εικόνα 34.. Το Boolean του Table1 πρώτα αντιστρέφεται με την χρήση της πράξης not και ύστερα γίνεται η πρόσθεση and. Για το άλλο Boolean γίνεται η πράξη or. Και οι δύο είσοδοι περνάνε από την πράξη pre.



Εικόνα 34. Σύνδεση δομικών στοιχείων.

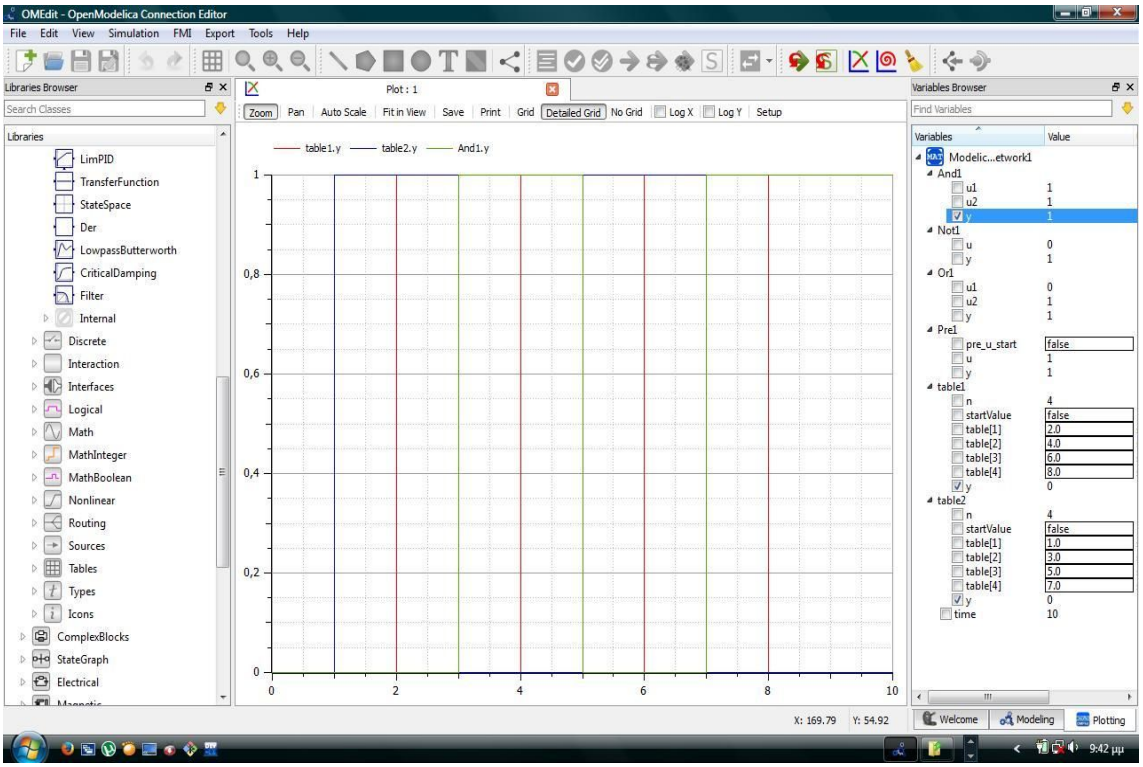
Τώρα είμαστε έτοιμοι να τρέξουμε το μοντέλο μας. Αν υπάρχει οποιοδήποτε σχεδιαστικό πρόβλημα κατά τη διάρκεια εκκίνησης της προσομοίωσης το DYMOLA θα εμφανίσει λεπτομερές μήνυμα λάθους ώστε ο χρήστης να το εντοπίσει γρήγορα και να το διορθώσει. Οι εικόνες που ακολουθούν μας δείχνουν τα' αποτελέσματα της προσομοίωσης του συστήματος.

Στην Εικόνα 35. φαίνονται οι έξοδοι των δυο Boolean εισόδων. Όπως φαίνεται και στην εικόνα και οι δυο παλμοι εξόδου έχουν ίδιο πλάτος αλλά διαφορετική δίαρκεια και καθορίζονται από τις πράξεις που γίνονται στο σύστημα.



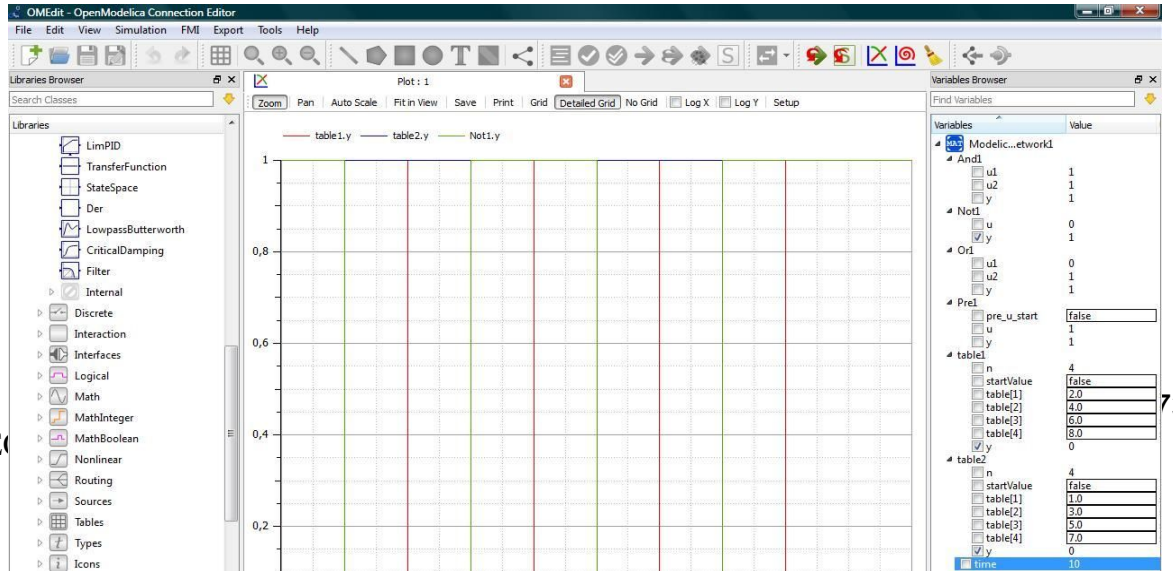
Εικόνα 35. Απεικόνιση των αποτελεσμάτων των table1 και table2 της προσομοίωσης του συστήματος.

Η επόμενη εικόνα Εικόνα 36., δείχνει την έξοδο των table1 και table2 καθώς και την έξοδο της πράξης and. Όπως φαίνεται και παρακάτω, το αποτέλεσμα της πράξης and είναι η «πρόσθεση» των table1 και table2. Έτσι ο παλμός της and έχει πλάτος ένα, αλλά η χρονική στιγμή που ξεκινάει είναι στο τρία.



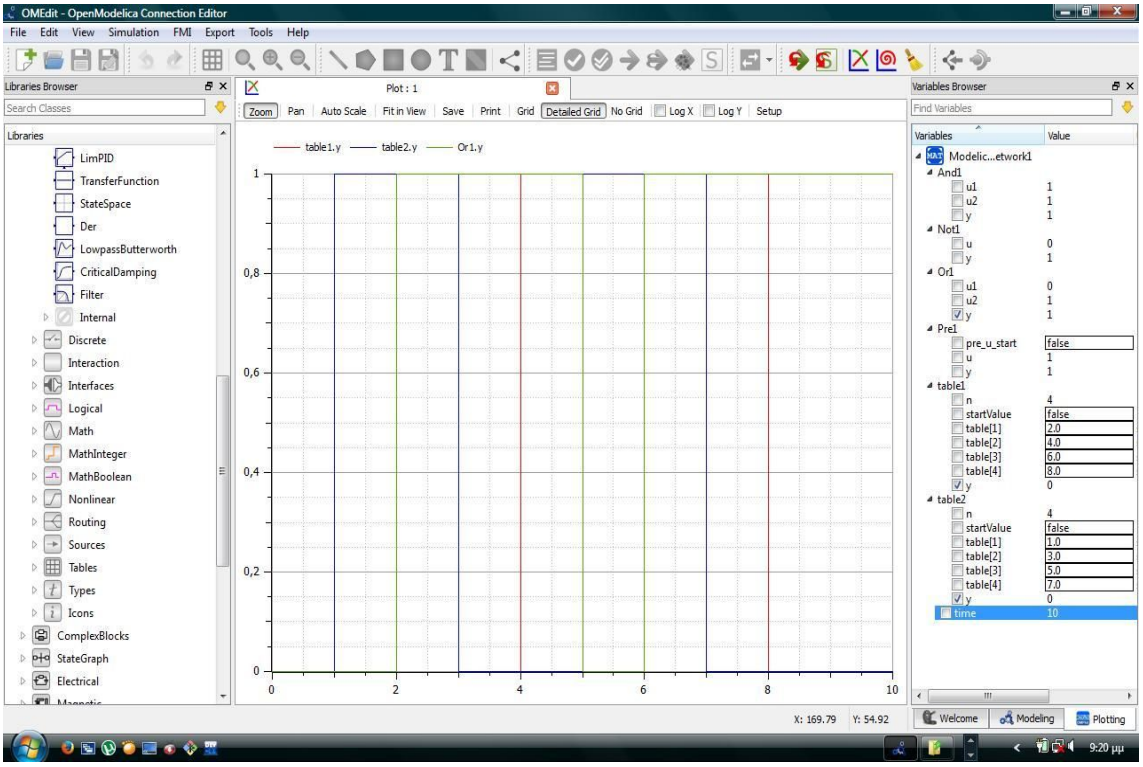
Εικόνα 36. Απεικόνιση των εξόδων table1, table2 και της πράξης and.

Στην επόμενη εικόνα, Εικόνα 37., φαίνεται η λογική πράξη not. Στην περίπτωση αυτή, η έξοδος που μας δίνεται στην προσομοίωση είναι η έξοδος του table2 ανεστραμένη.



Εικόνα 37. Απεικόνιση της εξόδου για την πράξη not.

Τελος, στην Εικόνα 38., απεικονίζεται η έξοδος των table1, table2 και της πράξης or. Ο παλμός εξόδου της or είναι το αποτέλεσμα της πράξης μεταξύ των table1 και table2, και όπως φαίνεται και στην εικόνα, διαφοροποιεί την διάρκεια και την φάση του παλμού.



Εικόνα 38. Απεικόνιση των table1 & table2 καθώς και της πράξης or.

ΚΕΦΑΛΑΙΟ 4:

4.1 MATLAB

4.1.1. ΕΙΣΑΓΩΓΗ

Το MatLab (MATrix LABoratory), είναι ένα ολοκληρωμένο προγραμματιστικό περιβάλλον το οποίο περιέχει την δικιά του γλώσσα προγραμματισμού, βιβλιοθήκες (libraries) καθώς και πολλές εργαλειοθήκες (toolboxes). Με το πέρασμα των χρόνων το MatLab έχει αναπτυχθεί αρκετά, ώστε να γίνει ένα ισχυρότατο εργαλείο το οποίο εκατομμύρια μηχανικοί και επιστήμονες χρησιμοποιούν παγκοσμίως για να αναλύουν, σχεδιάζουν και να προσομοιώνουν συστήματα, όπως στην αεροναυπηγική, στην ιατρική, στα τηλεπικοινωνιακά δίκτυα κ.α.. Χρησιμοποιείται επίσης στην εκμάθηση μηχανών, στην επεξεργασία σήματος, στην επεξεργασία εικόνας, στην επιστήμη της Πληροφορικής όπως ο προγραμματισμός, στις επικοινωνίες, στην χρήση της Πληροφορικής στα οικονομικά, στη ρομποτική, στη σχεδίαση συστημάτων ελέγχου και σε πάρα πολλούς άλλους τομείς.

Το MatLab χρησιμοποιεί μια υψηλού επιπέδου, βασισμένη σε πίνακες γλώσσα για να περιγράφει υπολογιστικά μαθηματικά και να μοντελοποιεί διάφορα συστήματα με την χρήση κώδικα εύκολου στην εκμάθηση. Είναι σχεδιασμένη έτσι ώστε ο χρήστης να μπορεί να χρησιμοποιεί γραφικά σύμβολα για να σχηματίζει μια σαφή εικόνα για τα δεδομένα που έχει. Διαθέτει βιβλιοθήκη μεγάλη σε ποικιλία εργαλειοθηκών, η οποία επιτρέπει την χρήση έτοιμων αλγορίθμων κάνοντας την δουλειά του χρήστη πολύ πιο εύκολη.

4.1.2. ΙΣΤΟΡΙΚΗ ΑΝΑΔΡΟΜΗ

Αν και σήμερα το MatLab είναι ένα πλήρως ολοκληρωμένο τεχνικό υπολογιστικό περιβάλλον προσομοίωσης, αρχικά ξεκίνησε ως ένα απλό πρόγραμμα «Εργαστηρίου Πινάκων – Matrix Laboratory». Οι J. H. Wilkinson, George Forsythe

και John Todd το 1950, ήταν οι πρώτοι επιστήμονες οι οποίοι ασχολήθηκαν με την ανάπτυξη ενός περιβάλλοντος για την επίλυση αριθμητικών υπολογισμών στην γραμμική άλγεβρα. Στα τέλη του 1970, ο Cleve Moler, πρόεδρος του τομέα της επιστήμης της Πληροφορικής στο Πανεπιστήμιο του Μεξικό, άρχισε ν' αναπτύσσει το MatLab χρησιμοποιώντας ως βάση την δουλειά των προηγούμενων ετών, καθώς και με την βοήθεια των πρωτοπόρων επιστημών.

Σκοπός του ήταν να σχεδιάσει ένα λογισμικό ώστε να δώσει στους φοιτητές του πρόσβαση σε λογισμικά βιβλιοθήκης για την εκτέλεση αριθμητικής γραμμικής άλγεβρας σε ψηφιακούς υπολογιστές, όπως το LINPACK και EISPACK, χωρίς να χρειαστεί να μάθουν FORTRAN. Σύντομα διαδόθηκε σε άλλα πανεπιστήμια και βρήκε ένθερμο κοινό στην κοινότητα των εφαρμοσμένων μαθηματικών.

Λόγω της δημοφιλίας αυτής, ο Moler συνεργάστηκε με τους μηχανικούς Jack Little και Steve Bangert οι οποίοι αναγνώρισαν την εμπορική δυνατότητα του προγράμματος, ξαναγράφοντας το MatLab σε C και παράλληλα ιδρύοντας το 1984 την MathWorks ώστε να συνεχίσουν την ανάπτυξη του.

Στην αρχή το MatLab χρησιμοποιήθηκε από ερευνητές και επαγγελματίες στον μηχανικό έλεγχο, ο οποίος ήταν και ο τομέας ειδικότητας του J. Little, αλλά γρήγορα αναπτύχθηκε και σε άλλα πεδία. Σήμερα χρησιμοποιείται σε πάρα πολλούς τομείς από την διδασκαλία γραμμικής άλγεβρας, στην Πληροφορική, Μηχανική και στα Συστήματα Ελέγχου και σε ακόμα περισσότερα γνωστικά πεδία.

4.2. ΧΡΗΣΗ ΤΟΥ MATLAB ΣΤΙΣ ΒΙΟΜΗΧΑΝΙΕΣ

Λόγω τις εύχρηστης λειτουργικότητας του και της ευρείας γκάμας από εργαλειοθήκες που διαθέτει, το MatLab χρησιμοποιείται σε πάρα πολλές βιομηχανίες. Τυπικό παράδειγμα είναι η χρήση του από εταιρίες αεροδιαστημικού σκοπού και εταιρίες αμυντικού τομέα. Τέτοιου είδους βιομηχανίες χρησιμοποιούν το MatLab και το Simulink ώστε να δημιουργούν πρότυπα συστήματα ελέγχου, επαναλαμβάνοντας συνεχώς δοκιμές βελτίωσης. Αυτοκινητοβιομηχανίες χρησιμοποιούν επίσης το MatLab για ν' αναπτύξουν συστήματα ελέγχου για την κατανάλωση των καυσίμων.

Οποιαδήποτε εταιρία που ασχολείται με τον τομέα των υπολογιστών, χρησιμοποιεί το MatLab για την δημιουργία πρότυπων αλγορίθμων. Άλλος ένας τομέας της Πληροφορικής που κάνει χρήση του, είναι η Ρομποτική. Έχοντας σαν πλεονέκτημα την ευκολία στην χρήση του καθώς και την hardware υποστήριξη, η δημιουργία ρομπότ υψηλού επιπέδου είναι πλέον μια πολύ απλουστευμένη διαδικασία.

Επιπλέον, πολλές οικονομικές εταιρίες χρησιμοποιούν το MatLab για να υπολογίσουν ή να βελτιστοποιήσουν τα επίπεδα κινδύνου. Το ίδιο το πρόγραμμα έχει εργαλειοθήκες οικονομικού περιεχομένου ώστε να κάνει αυτές τις αναλύσεις ακόμα πιο εύκολες. Ακόμα οποιαδήποτε εταιρία ή ερευνητικό κέντρο κάνει χρήση της ανάλυσης και παρουσίασης δεδομένων χρησιμοποιεί το MatLab για να κάνει την δουλεία της τόσο πιο εύκολή όσο και πιο έγκυρη.

4.3. ΠΛΕΟΝΕΚΤΗΜΑΤΑ ΧΡΗΣΗΣ ΤΟΥ MATLAB

Το MatLab αποτελεί ένα πολύ δημοφιλή και με μεγάλη γενικά αποδοχή προγραμματιστικό περιβάλλον καθώς μπορεί να χρησιμοποιηθεί σε πάρα πολλούς διαφορετικούς τομείς – εκπαιδευτικούς και βιομηχανικούς. Κυριότερο πλεονέκτημά του είναι η εκτέλεση πράξεων σε πίνακες, το οποίο χρησιμεύει στην επεξεργασία εικόνων καθώς και στην ανάλυση συστημάτων ελέγχου.

Επιπλέον, είναι πολύ εύκολη η ανάπτυξη κώδικα γραμμένο σε MatLab, καθώς διαθέτει μια ισχυρή, τεχνική γλώσσα, απ’ ότι για παράδειγμα σε γλώσσα προγραμματισμού C++. Ο κώδικας αυτός είναι συμπαγής και λιγότερος σε έκταση, επομένως ευκολότερος στην υλοποίηση. Για παράδειγμα, κώδικας 10 – 20 γραμμών σε C++, μπορεί να γραφτεί σε 1 με 2 γραμμές MatLab. Λαμβάνοντας αυτό υπόψιν, αντιλαμβανόμαστε την ευκολία που δίνει στην υλοποίηση προγραμμάτων μεγαλύτερου μεγέθους. Επίσης, η μεγάλη ποικιλία σε βιβλιοθήκες γλυτώνει τον χρήστη από την κατανάλωση άσκοπου χρόνου και ενέργειας στην εγκατάσταση και σύνθεση καινούργιων εργαλείων, ώστε να ικανοποιήσουν τις ανάγκες του για υλοποίηση του project. Αντίθετα, μ’ αυτόν τον τρόπο, έχει την δυνατότητα να δοκιμάζει και να εκτελεί καινούργιες ιδέες στη στιγμή.

Μέρος της δημοφιλίας του προγράμματος προέρχεται λόγω της μεγάλης

λειτουργικότητάς του, η οποία βασίζεται στην ευρεία βάση χρηστών του και την εκτενή καταγραφή και ανταλλαγή λύσεων και ιδεών μεταξύ τους μέσω του MathWorks. Όλο αυτό βοήθησε στην σχεδίαση και δημιουργία πολλών και διαφορετικών βιβλιοθηκών και εργαλείων ώστε να ικανοποιούν τις πολλές και ξεχωριστές ανάγκες των χρηστών. Παραδείγματος χάρη, πολλά projects έχουν άμεση και εύκολη λύση καθώς προηγούμενοι χρήστες δούλεψαν πάνω σ' αυτά οδηγώντας στην ανάπτυξη των κατάλληλων εργαλείων για τους επόμενους.

Τέλος, το MatLab υπερέχει στην καταγραφή των συναρτήσεων καθώς προσφέρει ποικίλους τρόπους απεικόνισης των δεδομένων τα οποία είναι φιλικά προς το χρήστη.

4.4. ΜΕΙΟΝΕΚΤΗΜΑΤΑ ΧΡΗΣΗΣ ΤΟΥ MATLAB

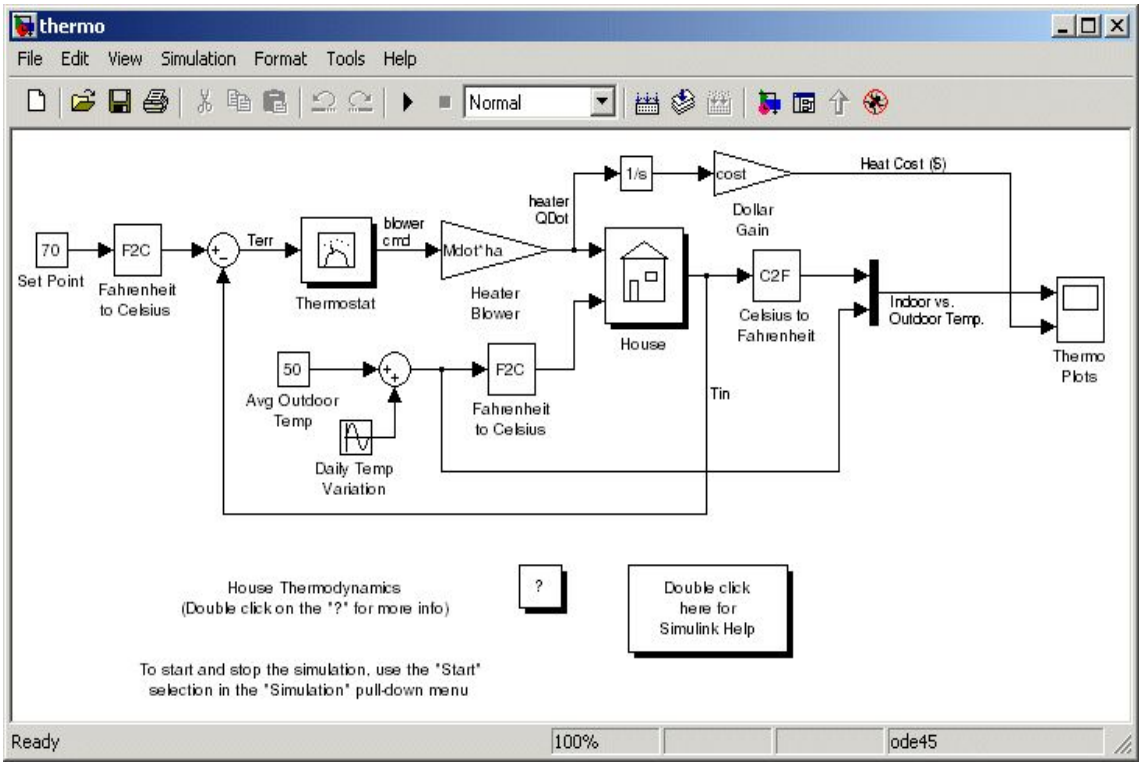
Πρώτο και κυριότερο μειονέκτημα του MatLab είναι το κόστος. Μια πλήρης έκδοση του προγράμματος κοστίζει πέντε με δέκα φορές περισσότερο από έναν μεταγλωττιστή C ή Fortran. Ωστόσο, η χρήση του μειώνει σε σημαντικά επίπεδα το χρόνο που χρειάζεται ένα συγκεκριμένο project για να ολοκληρωθεί. Ακόμα κι έτσι όμως η αγορά του από κάποιον κρίνεται πολύ ακριβή. Γι' αυτό, υπάρχει η έκδοση student η οποία είναι σχεδόν ίδια με την πλήρη έκδοση και σε πολύ χαμηλότερη τιμή.

Επίσης το MatLab χρησιμοποιεί μια επεξηγηματική γλώσσα και έτσι εκτελεί τις εντολές πιο αργά απ' ότι θα έκαναν γλώσσες που χρησιμοποιούν μεταγλωττιστές. Το πρόβλημα αυτό , παρ' όλα αυτά, λύνεται με το σωστό «χτίσιμο» του προγράμματος στο MatLab, αυξάνοντας μ' αυτόν τον τρόπο την ταχύτητα εκτέλεσης του κώδικα και επομένως και των εργασιών.

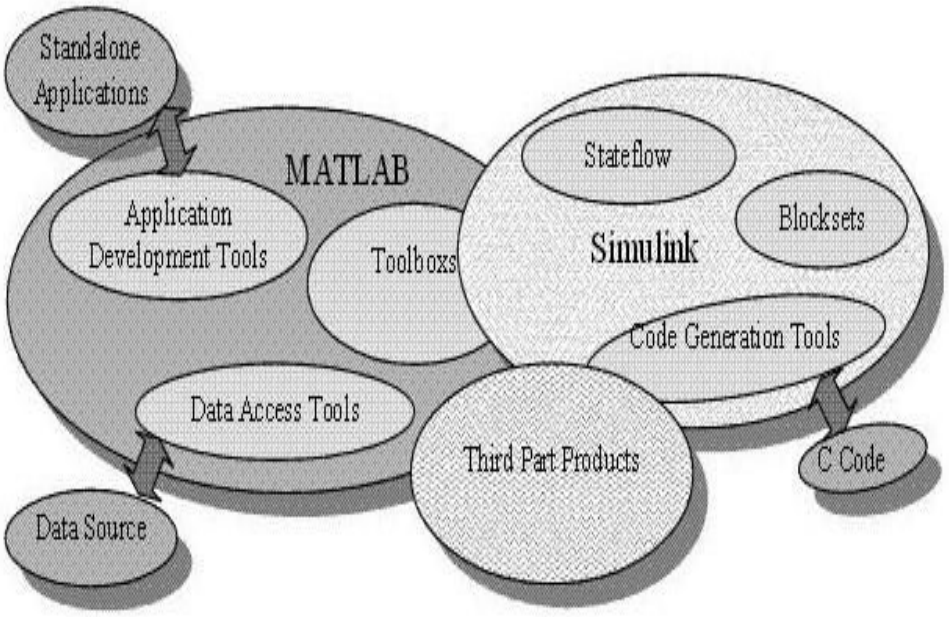
4.5 Η ΣΧΕΣΗ ΤΟΥ MATLAB ΜΕ ΤΟ SIMULINK

Το Simulink είναι ένα περιβάλλον block διαγράμματος που χρησιμοποιείται για προσομοίωση πολλαπλών επιπέδων (multidomain) καθώς και για μοντελοποίηση. Επιπλέον υποστηρίζει τη δημιουργία αυτόματου κώδικα και συνεχείς δοκιμές κι επαλήθευση συστημάτων.

Το Simulink παρέχει έναν γραφικό συντάκτη και προσαρμόσιμες block βιβλιοθήκες. Επίσης έχει την δυνατότητα να δίνει λύσεις σε μοντελοποίηση και προσομοίωση δυναμικών συστημάτων. Το περιβάλλον του Simulink είναι ενοποιημένο με το MatLab, δίνοντας του τη δυνατότητα να ενσωματώνει αλγόριθμους του τελευταίου σε μοντέλα και να εξάγει αποτελέσματα προσομοίωσης σ' αυτό για περαιτέρω ανάλυση.



Εικόνα 39. Ένα απλό παράδειγμα χρήσης του Simulink μέσω του MatLab. Τρέχουμε το demo που εμφανίζεται στην εικόνα πληκτρολογώντας thermo στο command window του MatLab. Το μοντέλο περιγράφει την θερμοδυναμική ενός σπιτιου.



Εικόνα 40. Η σχέση μεταξύ MatLab και Simulink. Το περιβάλλον του Simulink είναι ενοποιημένο με το MatLab.

4.6. DYMOLA vs. MATLAB

4.6.1. ΓΕΝΙΚΑ

Το DYMOLA είναι ένα εργαλείο που χρησιμοποιείται για μοντελοποίηση και προσομοίωση. Το MatLab από την άλλη μεριά είναι μια «εξειδικευμένη» γλώσσα και ένα τεχνικό, υπολογιστικό περιβάλλον, το οποίο προσφέρει μαθηματικά και γραφικά εργαλεία για ανάλυση δεδομένων, οπτικοποίηση και αλγοριθμική ανάπτυξη εφαρμογών.

4.6.2. ΣΥΓΚΡΙΣΗ ΜΕΤΑΞΥ DYMOLA & MATLAB

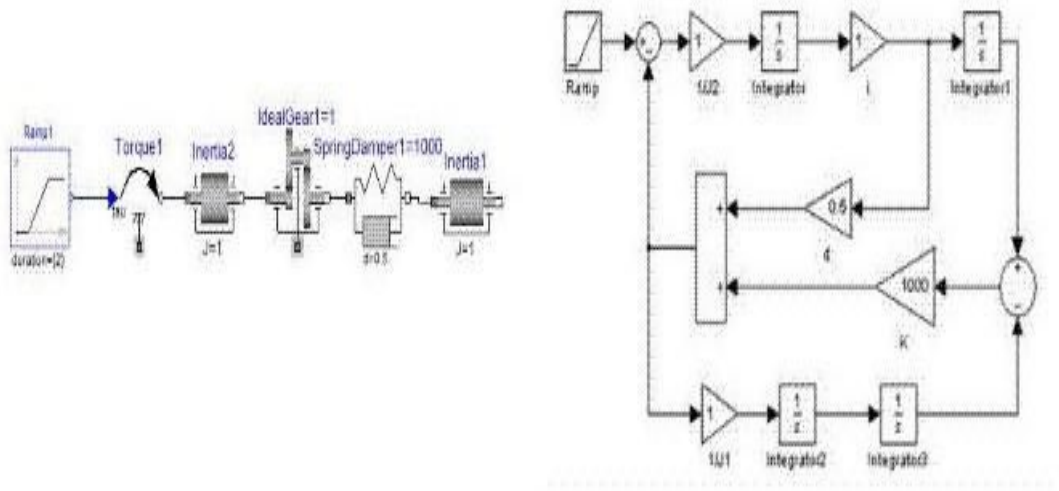
Το DYMOLA και το MatLab είναι δυο πάρα πολύ χρήσιμα και αρκετά παρόμοια εργαλεία, τα οποία παρέχουν εξαιρετική βοήθεια στους ανθρώπους που απασχολούνται τόσο στο χώρο της Πληροφορικής και της Μηχανικής Υπολογιστών γενικότερα, όσο και στους χώρους της Επιστήμης και της Εκπαίδευσης. Και τα δύο αυτά εργαλεία έχουν τα πλεονεκτήματα και μειονεκτήματα τους και η επιλογή χρήσης του ενός εκ των δυο είναι καθαρά επιλογή των απαιτήσεων που θέλει να εκπληρώσει ο χρήστης. Συγκρίνοντας τα, μας παρουσιάζονται πολλές ομοιότητες και διαφορές μεταξύ των δυο.

1. Τόσο το MatLab όσο και το DYMOLA μπορούν να δουλέψουν στις ίδιες «πλατφόρμες» όπως τα Windows, UNIX και Linux. Η διαφορά τους είναι στο χώρο της μνήμης τον οποίο απαιτεί το καθένα (DYMOLA → 150MB δίσκος 128MB RAM, MatLab → 600MB δίσκος, 256MB RAM μόνο για εγκατάσταση του χωρίς Simulink).

2. Δεν χρειάζονται κάποιο άλλο λογισμικό για να τρέχουν καθώς και τα δύο είναι επαρκή περιβάλλοντα για μοντελοποίηση και προσομοίωση.
3. Η προγραμματιστική τους γλώσσα διαφέρει. Το DYMOLA χρησιμοποιεί την γλώσσα MODELICA, η οποία είναι ιδανική για την μοντελοποίηση μεγάλων, πολύπλοκων και ετερογενών φυσικών συστημάτων. Το MatLab, από την άλλη μεριά χρησιμοποιεί M–files τα οποία είναι έτοιμα πρότυπα/προγράμματα που στηρίζονται στη γλώσσα MatLab και σε προγραμματισμό κώδικα σε C. Αυτό έχει σαν αποτέλεσμα να χρειάζεται περισσότερος χρόνος και εργασία για να «χτίσει» κάποιος πολύπλοκα φυσικά συστήματα.
4. Κληρονομικότητα στοιχείων κι επανεγγραφή. Η MODELICA είναι μια γλώσσα σχεδιασμένη έτσι ώστε να χτίζει βασικά στοιχεία τα οποία μπορούν να κληρονομηθούν (extended classes). Ο χρήστης έχει την δυνατότητα να χρησιμοποιεί ήδη υπάρχοντα στοιχεία ενώ καινούρια «χτίζονται». Στο MatLab δεν υπάρχει η ¨ιδέα¨ της κληρονομικότητας. Για παράδειγμα, αν χρησιμοποιούμε block διαγράμματα στο Simulink και κάποιος χρήστης θέλει να χρησιμοποιήσει συγκεκριμένα στοιχεία από κάποιο απ’ αυτά θα πρέπει να τα ξαναγράψει.
5. Επεκτασιμότητα. Και τα δύο περιβάλλοντα έχουν ανοικτή αρχιτεκτονική και μπορούν να επεκτείνουν τα εργαλεία τους.
6. Διεπαφή (Interface). Το MatLab και το DYMOLA αν κι έχουν πολύ διαφορετικό user Interface, ταυτόχρονα παρουσιάζουν και κάποιες ομοιότητες. Το DYMOLA υποστηρίζει χτίσιμο σε block διαγράμματα και ενώση εικονιδίων. Το MatLab χρησιμοποιεί κατά κύριο λόγο Command Window για να δίνει τις εντολές και να εκτελεί ενέργειες, γράφοντας κατά κύριο λόγω κώδικα σε γλώσσα MatLab. Όμως μέρος του περιβάλλοντος αποτελεί και το Simulink το οποίο υποστηρίζει χτίσιμο σε block διάγραμμα, παρόμοιας λογικής με το χτίσιμο σε block διάγραμμα του DYMOLA. Έτσι λοιπόν, αν και φαινομενικά τα δυο εργαλεία δείχνουν εντελώς διαφορετικά από τη μεριά διεπαφής τους με τον χρήστη, έχουν μεγάλη ομοιότητα.
7. Οπτικοποιήση αποτελεσμάτων προσομοίωσης. Στο MatLab υπάρχουν δυο τρόποι για να το πετύχουμε αυτό: με plotting και animation. Επομένως, μόνο τα

ολοκληρωμένα αποτελέσματα μπορούν ν' απεικονισθούν. Το DYMOLA είναι ένα εργαλείο που μας επιτρέπει να «χτίζουμε» σταδιακά τη δουλειά μας και μπορούμε να βλέπουμε αν τ' αποτελέσματα που μας δίνονται σταδιακά είναι τα επιθυμητά. Αν όχι, έχουμε την δυνατότητα να κάνουμε τροποποιήσεις και διορθώσεις.

8. Δυνατότητες ανάλυσης αποτελεσμάτων. Το MatLab είναι ένα πανίσχυρο εργαλείο για να υπολογίζει, να σχεδιάζει και ν' αναλύει δεδομένα. Αυτό μπορούμε να το πετύχουμε είτε χρησιμοποιώντας το MatLab Window Command, είτε το GUI interface. Το DYMOLA απ' τη μεριά του, δίνει τη δυνατότητα στο χρήστη να σχηματίζει μια σαφή εικόνα σχετικά με τ' αποτελέσματα αρκετά εύκολα αλλά γενικά θεωρείται ένα λιγότερο ισχυρό εργαλείο γι' ανάλυση και υπολογισμούς.
9. Σύγκριση παραγωγής κώδικα. Και τα δυο περιβάλλοντα μπορούν να χρησιμοποιήσουν κώδικα C για προσομοίωση μοντέλων. Οι χρήστες έχουν την δυνατότητα να γράψουν τον δικό τους κώδικα για να ελέγξουν το πρόγραμμα.
10. Διαφορά στη τιμή. Σε γενικές γραμμές το MatLab θεωρείται ακριβότερο απ' ότι το DYMOLA. Η διαφορά αυτή υπάρχει καθώς το MatLab περιέχει πάρα πολλές βιβλιοθήκες, οι οποίες είναι πολύ ακρίβες.



Dymola Model

Simulink Model

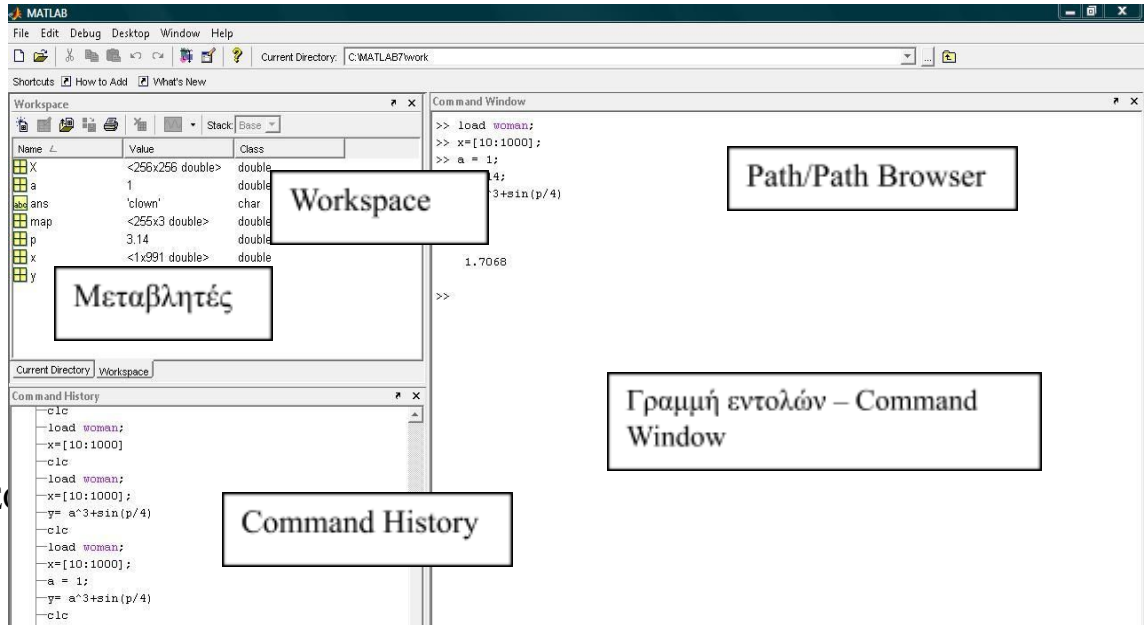
Εικόνα 41. Ίδιο μοντέλο «χτισμένο» σε DYMOLA & MATLAB/SIMULINK. Σύμφωνα με την εικόνα το DYMOLA χρησιμοποιείται για πολύπλοκα μοντέλα προσομοίωσης με πολλές εξισώσεις κυρίως πολλαπλών τομέων. Έτσι είναι ευκολότερο στην χρήση λόγω του αντικειμενοστραφούς προσανατολισμού. Όμως το MATLAB είναι ισχυρότερο εργαλείο στην διεξαγωγή υπολογισμών, ανάλυσης δεδομένων και παροχής αποτελεσμάτων.

4.7. ΤΟ ΠΕΡΙΒΑΛΛΟΝ ΕΡΓΑΣΙΑΣ ΤΟΥ MATLAB

Στο MatLab, μπορούμε να εργαστούμε με δυο διαφορετικούς τρόπους. Μπορεί να χρησιμοποιηθεί το *Command Window*, όπου ο χρήστης δίνει μεμονωμένες εντολές κάνοντας δοκιμές και παίρνοντας άμεσα αποτελέσματα. Ο άλλος τρόπος είναι η χρήση *scripts*, προγραμμάτων δηλαδή, τα οποία ο χρήστης γραφεί στο περιβάλλον του editor. Το MatLab έχει έτοιμα scripts που ο χρήστης μπορεί να χρησιμοποιήσει άμεσα.

Τα scripts αποτελούνται από σειρές εντολών τα οποία εκτελούνται είτε απευθείας από τον editor δίνοντας την εντολή *Run* στο Command Window, είτε γράφοντας τα' όνομα του. Αποθηκεύονται με την κατάληξη **.m** και η αποθήκευση γίνεται συνήθως στο αρχείο *work* μέσα στα *directories* του MatLab καθώς η τοποθεσία αυτή είναι προεπιλεγμένη στο περιβάλλον.

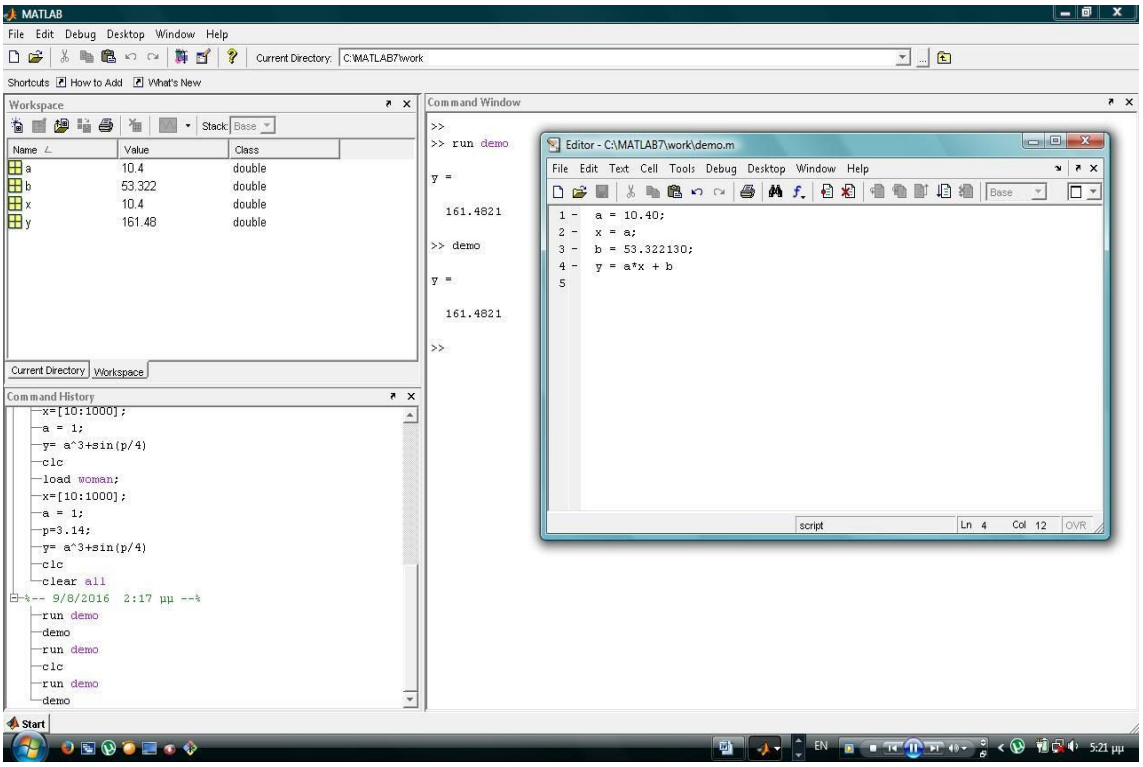
Ο χρήστης έχει την δυνατότητα ν' αλλάξει το *path* του directory είτε γράφοντας το στη γραμμή που εμφανίζεται στο πάνω μέρος της οθόνης, είτε μέσω *του Path Browser*. Σε κάθε περίπτωση τ' αρχεία θα πρέπει να είναι αποθηκεμένα σ' ένα και μόνο directory και αυτό να έχει οριστεί από το MatLab.



Εικόνα 42. Το περιβάλλον εργασίας του MatLab.

Το περιβάλλον εργασίας του MatLab περιλαμβάνει τα παράθυρα εργασίας τα οποία φαίνονται παραπάνω στο στην εικόνα 42. τα οποία αναλυτικά είναι:

- Πρώτο και κυριότερο είναι το Command Window, τ' οποίο είναι το παράθυρο εργασίας όπου περιέχεται η γραμμή εντολών. Σ' αυτό ο χρήστης πληκτρολογεί όλες τις εντολές που θέλει να εκτελέσει.
- Το Workspace είναι το παράθυρο που αποθηκεύονται όλες οι μεταβλητές και πίνακες που δημιουργεί ο χρήστης μέσω της γραμμής εντολών.
- Το Command History περιέχει το ιστορικό του περιβάλλοντος και κατ' επέκταση των χρηστών. Εκεί φαίνονται όλες οι εντολές που εκτελέστηκαν απ' όταν έγινε η εκκίνηση του προγράμματος, αλλά και οι εντολές που έδωσαν οι χρήστες την προηγούμενη φορά που χρησιμοποίησαν το MatLab. Μ' αυτόν τον τρόπο είναι εύκολο να ελέγχουν τις εντολές που εκτέλεσαν τις προηγούμενες φορές. Καλό θα ήταν όμως το Command History να διαγράφεται ώστε να μην καταναλώνει χώρο στην μνήμη.
- Στο επάνω μέρος της διεπαφής του MatLab φαίνεται η γραμμή του Path όπου αποθηκεύονται τ' αρχεία, καθώς και το κουμπί που οδηγεί στον Path Browser.
- Τέλος, εκτός απ' αυτά τα παράθυρα υπάρχει και ο editor που χρησιμοποιείται για την συγγραφή των προγραμμάτων και ο οποίος φαίνεται στην παρακάτω εικόνα.



Εικόνα 43. Ένα απλό παράδειγμα χρήσης του editor του MatLab. Ο χρήστης μπορεί να γράφει προγράμματα και να τα καλεί και να τα τρέχει ανά πάσα στιγμή μέσω του Command Window.

4.8 ΑΠΛΟ ΠΑΡΑΔΕΙΓΜΑ ΧΡΗΣΗΣ ΤΟΥ MATLAB

Το MatLab όπως αναφέρθηκε είναι ένα πολύ χρήσιμο εργαλείο το οποίο μπορεί να χρησιμοποιηθεί σε πολλούς διαφορετικούς επαγγελματικούς χώρους. Έτσι, όπως είναι φυσικό, είναι βασικό στη μελέτη και τη κατασκευή μοντέλων φυσικών συστημάτων που απασχολούν τον κλάδο των μηχανικών. Στην ενότητα που ακολουθεί θα δούμε ένα απλό αλλά βασικό παράδειγμα χρήσης του.

Ως μηχανικοί κατανοούμε πόσο σημαντικό είναι να είμαστε σε θέση να υπολογίζουμε την απόκριση συχνότητας ενός συστήματος, συνεχούς ή διακριτού χρόνου, καθώς αυτή μας δίνει πληροφορίες για την συμπεριφορά του συστήματος στην έξοδο ανάλογα την είσοδο που θα εφαρμόσουμε.

Η απόκριση συχνότητας H(Ω) αποτελεί την περιγραφή ενός συστήματος στο πεδίο της κυκλικής συχνότητας και δίνεται από τον τύπο.:

$$H(\Omega) = F\{h(t)\} = Y(\Omega) / X(\Omega)$$

ο οποίος είναι ο μετασχηματισμός Fourier της κρουστιής απόκρισης h(t) του συστήματος.

Με τη βοήθεια της απόκρισης συχνότητας μπορούμε να καθορίσουμε εάν το σύστημα μας θα μπορεί να λειτουργεί σαν φίλτρο και θα μηδενίζει το σήμα εισόδου σε κάποιο συγκεκριμένο διάστημα συχνοτήτων. Εκτός απ' τη κατηγορία των ιδανικών φίλτρων όπου το σήμα X(Ω) περνά αναλλοίωτο από το σύστημα υπάρχουν τέσσερις βασικές κατηγορίες φίλτρων.:

- *τα βαθυπερατα (low - pass) φίλτρα* : το σήμα περνάει αναλλοίωτο στις χαμηλές συχνότητες και αποκόπτεται στις υψηλές.
- *τα υψιπερατά (high - pass) φίλτρα* : κάνουν ακριβώς το αντίθετο από τα βαθυπερατα. Αποκόπτουν δηλαδή το σήμα στις χαμηλες συχνότητες και το περνάνε αναλλοίωτο στις υψηλές.
- *τα ζωνοπερατά (band - pass) φίλτρα* : το σήμα περνάει χωρίς παραμόρφωση για συγκεκριμένη ζώνη συχνοτήτων και αποκόπτεται στις υπόλοιπες.
- *τα ζώνης αποκοπής (band - stop) φίλτρα* : αυτά τέλος, αποκόπτουν το σήμα για μια συγκεκριμένη ζώνη συχνοτήτων και το περνάνε χωρίς παραμόρφωση από τις υπόλοιπες.

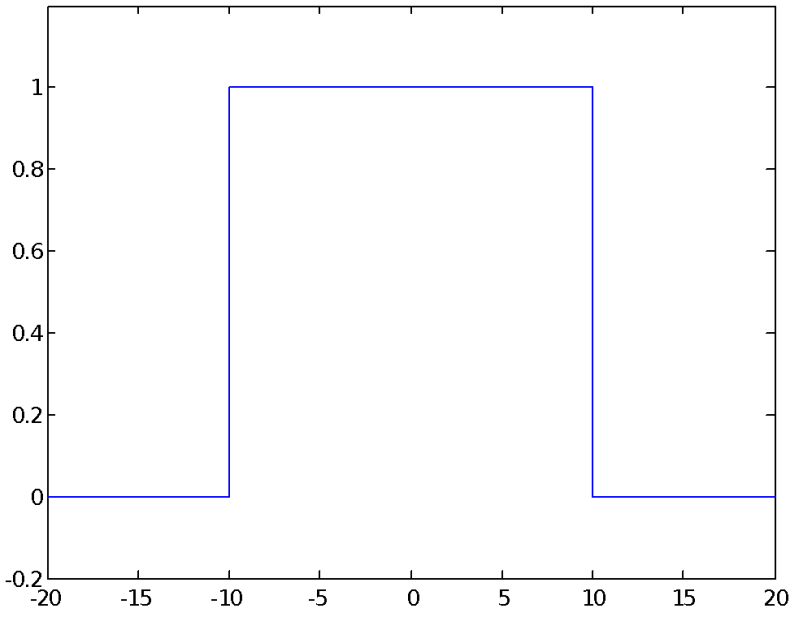
Επόμένως για να παραμείνει το σήμα χωρίς αλλοιώσεις κατά το περασμά του από ένα φίλτρο, θα πρέπει η απόκριση της συχνότητας του φίλτρου να είναι ένα στη ζώνη συχνοτήτων που επιθυμούμε το φίλτρο να παραμείνει αναλλοίωτο.

Στη συνέχεια θα σχεδιάσουμε με τη βοήθεια του MatLab την απόκριση συχνότητας ενός ιδανικού βαθυπερατού φίλτρου γραμμικής φάσης καθώς επίσης και την κρουστική απόκριση αυτού. Η απόκριση συχνότητας ενός τετοιου φίλτρου ορίζεται ως εξής:

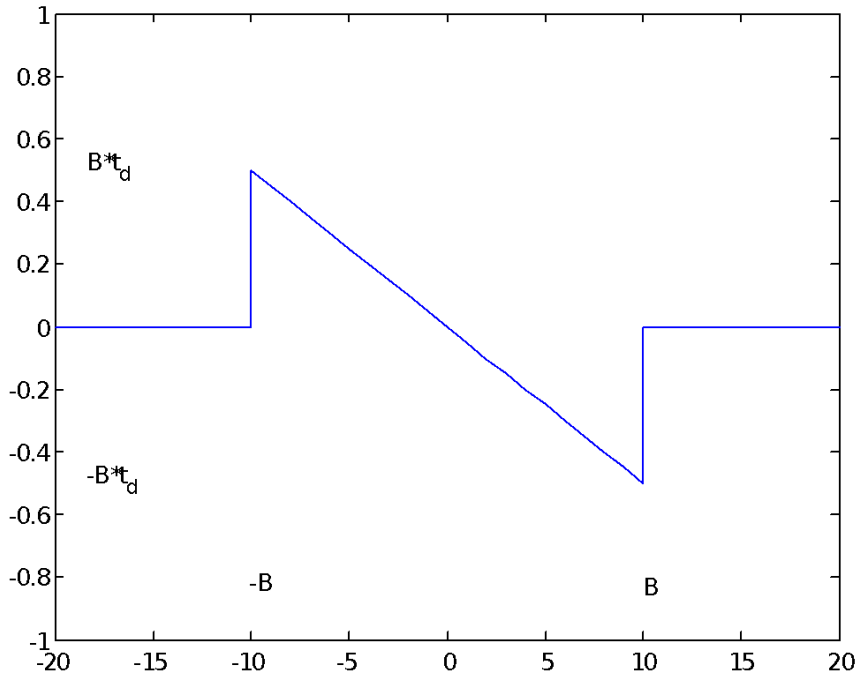
$$H(\Omega) = \begin{cases} e^{-j\Omega td}, & -B \le \Omega \le B \\ 0, & \Omega < -B,\ \Omega > B \end{cases}$$

Ο κώδικας που γράφουμε στο *Command Window* του MatLab για την δημιουργία του φίλτρου είναι ο παρακάτω:

```
>> B=10;
>> td=0.05;
>> w1=-20:-B;
>> H1=zeros(size(w1));
>> w2=-B:B;
>> H2=exp(-j*w2*td);
>> w3=B:20;
>> H3=zeros(size(w3));
>> w=[w1 w2 w3];
>> H=[H1 H2 H3];
>> plot(w,abs(H))
>> figure;
>> plot(w,angle(H))
>>
```



Εικόνα 44(α). Απεικόνιση του μέτρου απόκρισης της συχνότητας.

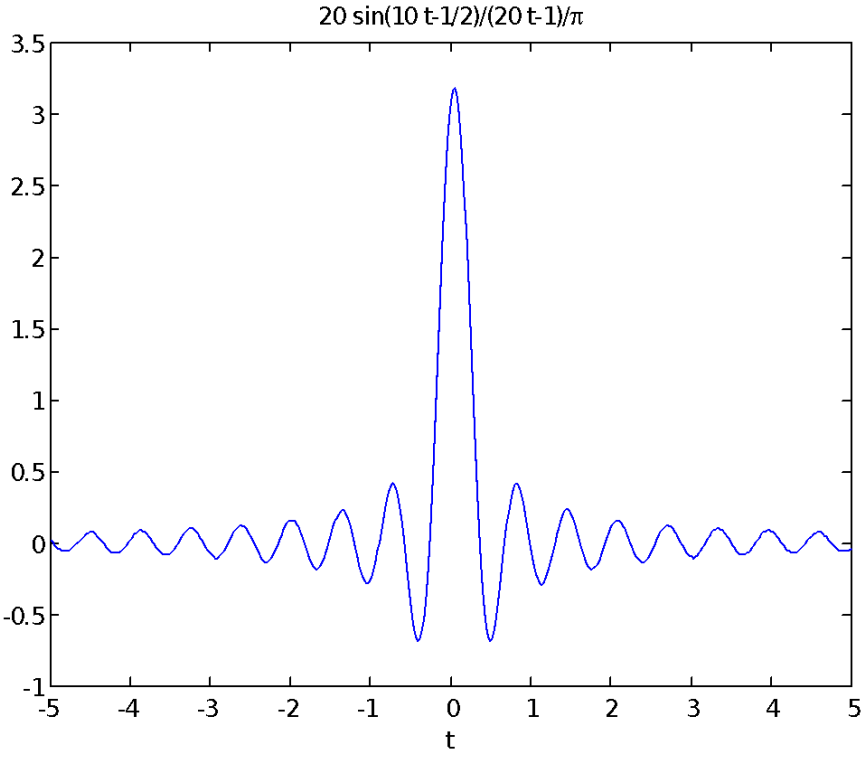


Εικόνα 44.(β). Απεικόνιση της φάσης απόκρισης της συχνότητας.

Ο παραπάνω κώδικας υπολογίζει και σχεδιάζει τις γραφικές παραστάσεις για το μέτρο και τη φάση απόκρισης της συχνότητας του φίλτρου. Οι γραφικές παραστάσεις εμφανίζονται με την εντολή *plot()*.

Τέλος, ο παρακάτω κώδικας υπολογίζει και εμφανίζει την γραφική παράσταση της κρουστικής απόκρισης του συστήματος.

```
>> syms t w
>> H=exp(-j*w*td)*(heaviside(w+B)-heaviside(w-B));
>> h=ifourier(H,t);
>> figure;
>> ezplot(h,[-5 5]);
>>
```



Εικόνα 44.(γ). Κρουστική απόκριση h(t).

Από τις γραφικές παραστάσεις που παίρνουμε με την βοήθεια του MatLab βλέπουμε ότι αφού αρχικά ορίσαμε την απόκριση συχνότητας του φίλτρου, στη συνέχεια σχεδιάσαμε το μέτρο και την φάση. Το μέτρο πράγματι είναι 1 (Εικόνα 44.(α)) και η φάση είναι Ω_{td} στο $-B \leq \Omega \leq B$ και η κλίση της ευθείας της φάσης είναι $-T_d$ (Εικόνα 44.(β).). Τέλος, στην Εικόνα 44.(γ). παίρνουμε την κρουστική απόκριση h(t) του φίλτρου η οποία είναι ένας παλμός κεντραρισμένος στο T_d.

Το παραπάνω παράδειγμα είναι ένα τυπικό δείγμα της χρησιμότητας του MatLab για έναν μηχανικό, κυρίως στο μαθηματικό κομμάτι της δουλειά του, καθώς ο υπολογισμός και η σχεδίαση της απόκρισης συχνότητας και της κρουστικής

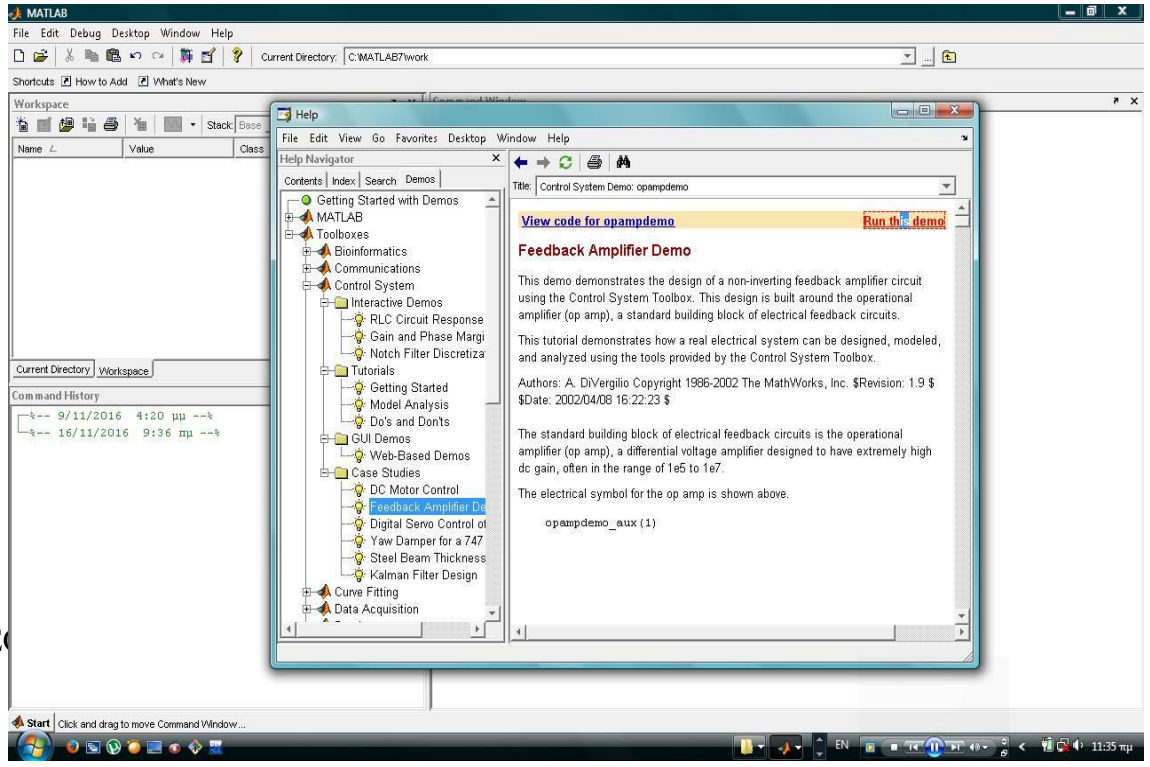
απόκρισης ενός βαθυπερατου φίλτρου στο χαρτί είναι χρονοβόρα και δύσκολη μαθηματική διαδικασία. Με τη χρήση του MatLab όμως γίνεται άμεσα και εύκολά και έχουμε μια σωστή απεικόνιση των δεδομένων

4.9. ΧΡΗΣΗ ΤΟΥ MATLAB TOOLBOX ΓΙΑ ΤΗΝ ΧΡΗΣΗ ΤΩΝ DEMOS

4.9.1 ΚΑΘΙΕΩΡΕΝΟ ΜΟΝΤΕΛΟ ΤΕΛΕΣΤΙΚΟΥ ΕΝΙΣΧΥΤΗ

Το MatLab, όπως και το DYMOLA, μας επιτρέπει να κάνουμε χρήση των ήδη δημιουργημένων μοντέλων που υπάρχουν, τόσο για εκμάθηση όσο και για να δούμε πως μπορούμε να βελτιώσουμε το δικό μας project. Στο MatLab βέβαια τα demos χρησιμοποιούνται λίγο διαφορετικά, καθώς μας προσφέρουν την γενική άποψη για το μοντέλο καθώς και οδηγίες για να το δημιουργήσουμε μόνοι μας.

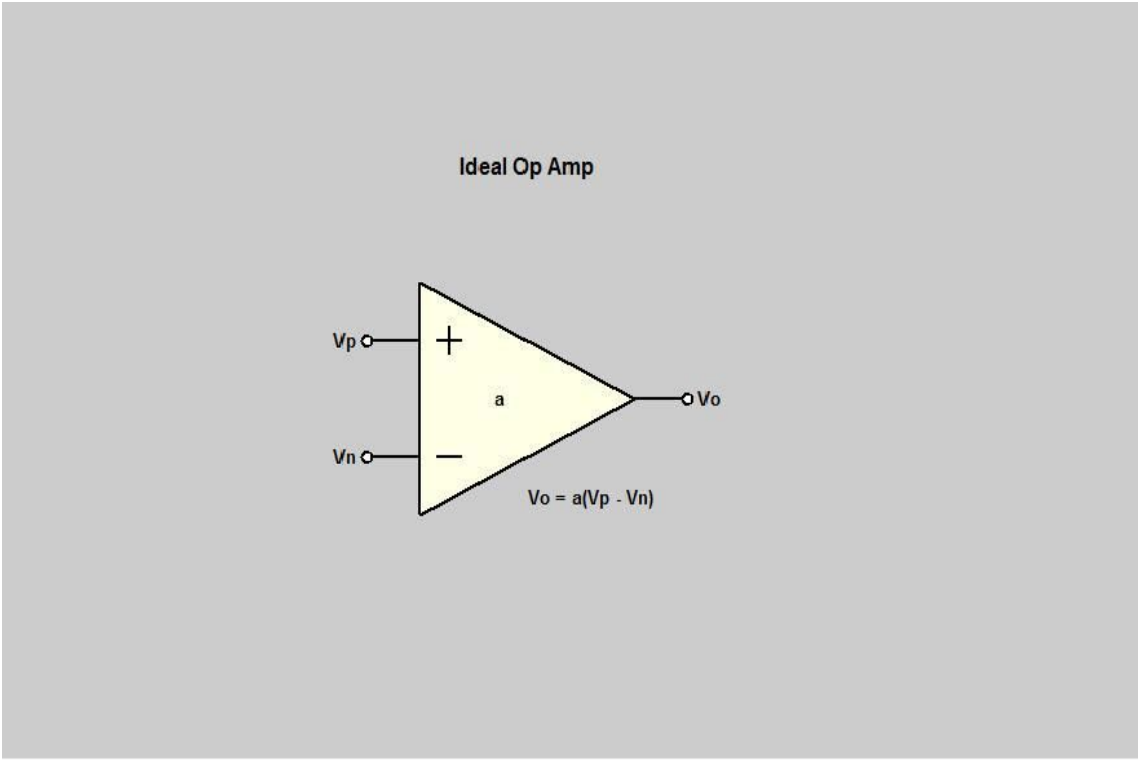
Θα χρησιμοποιήσουμε το demo που υπάρχει στο MatLab ώστε να δημιουργήσουμε έναν τελεστικό ενισχυτή με ανάδραση (Feedback Amplifier). Ξεκινώντας το MatLab επιλέγουμε από τη γραμμή εργαλειών των εικονίδιο της βοήθειας (help). Στο παράθυρο που ανοίγει επιλέγουμε την καρτέλα demos και έπειτα Toolboxes → Control Systems → Case Studies → Feedback Amplifier Demo. Η διαδικασία φαίνεται στην παρακάτω εικόνα, Εικόνα 45.



Εικόνα 45. Άνοιγμα των Demo στο MatLab.

Το demo παρουσιάζει τη σχεδίαση ενός μη – αντιστρέψιμου τελεστικού ενισχυτή με ανάδραση, χρησιμοποιώντας το Control System ToolBox του MatLab. Το συγκεκριμένο μοντέλο έχει δημιουργηθεί με βάση το καθιερωμένο μοντέλο του τελεστικού ενισχυτή (δημιουργία ηλεκτρικών κυκλωμάτων με ανάδραση). Με λίγα λόγια παρουσιάζει πως ένα πραγματικό ηλεκτρικό σύστημα μπορεί να σχεδιαστέι, μοντελοποιηθεί και αναλυθεί χρησιμοποιώντας τα εργαλεία του MatLab.

Το καθιερωμένο block μοντέλων ηλεκτρικών κυκλωμάτων με ανάδραση είναι ο standar τελεστικός ενισχυτής. Αυτός περιλαμβάνει ενισχυτή με διαφορική διαφορα δυναμικού, σχεδιασμένο έτσι ώστε να έχει εξαιρετικά υψηλό κέρδος dc ρεύματος. Αποτελείται από τη τάση στην είσοδο, V_p^+ και V_n^- , το κέρδος ρεύματος α και την τελική τάση εξόδου V_o. Η σχέση που μας δίνει την τάση εξόδου είναι: $V_o = \alpha (V_p - V_n)$. Στην παρακάτω εικόνα, Εικόνα 46., βλέπουμε το σχήμα του τελεστικού καθώς και τα χαρακτηριστικά του.



Εικόνα 46. Τελεστικός ενισχυτής

α = κέρδος
V_p = θετική τάση
V_n = αρνητική τάση
V_o = τελική τάση εξόδου

$$V_o = \alpha (V_p - V_n)$$

Θέλουμε την συνάρτηση μεταφοράς του συστήματός. Υποθέτουμε ότι ο τελεστικός ενισχυτής λειτουργεί σε γραμμική κατάσταση και όχι σε κατάσταση κορεσμού. Επομένως έχουμε μία εξίσωση που περιγράφει τη συνάρτηση μεταφοράς της ανοικτής ανάδρασης ως ένα γραμμικο σύστημα σταθερού χρόνου. Η συναρτηση μεταφοράς ανοικτού βρόγχου είναι:

$$\alpha(s) = \frac{a_0}{(1 + s / w_1)(1 + s / w_2)}$$

Εδώ υποθέτουμε τη χρήση ενός μη – ανταγωνιστικού τελεστικού ενισχυτή με δύο πόλους (με συχνότητες w_1, w_2) και κέρδος ρεύματος (dc gain) α_o.

Θα δημιουργήσουμε στο MatLab την συνάρτηση μεταφοράς. :

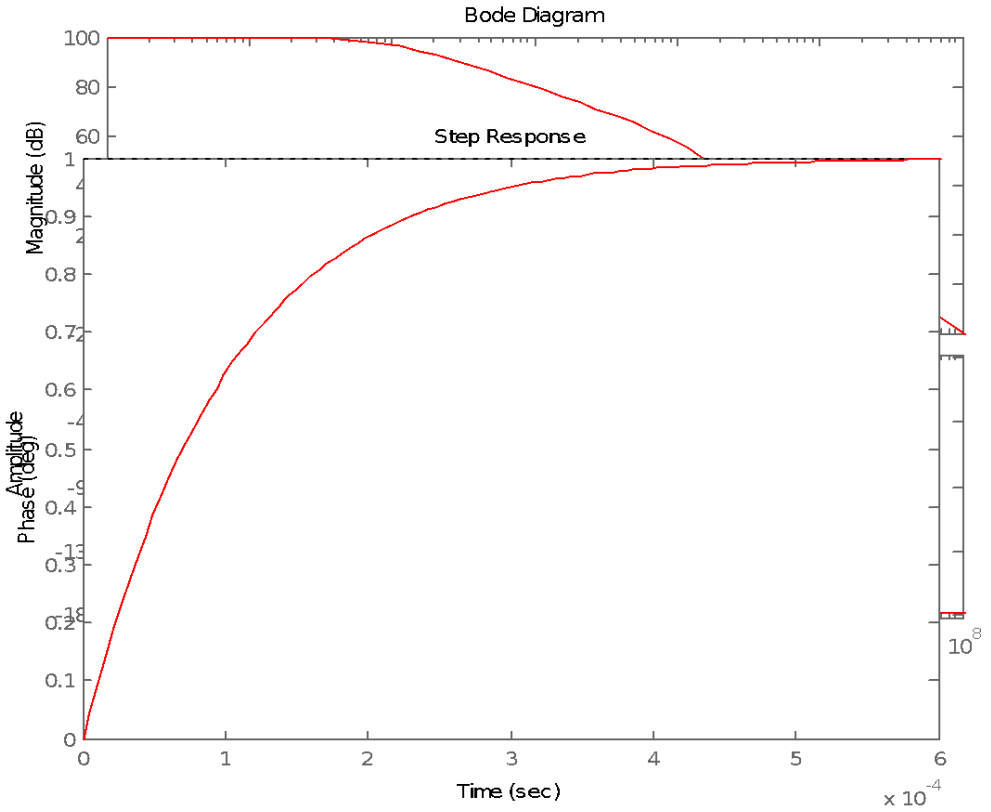
```
>> a0 = 10^5;
>> w1 = 10^4;
>> w2 = 10^6;
>> a = a0/(1+s/w1)/(1+s/w2)

Transfer function:
        100000
---------------------------
1e-010 s^2 + 0.000101 s + 1

>>
```

Αρχικά δώσαμε τιμές στους δύο πόλους της συχνότητας καθώς και στο dc κέρδος του τελεστικού. Έπειτα ορίσαμε μια μεταβλητή Laplace s, χτησιμοποιώντας την εντόλη tf (transfer function) και μ' αυτήν δημιουργήσαμε μια συνάρτηση μεταφοράς ανοικτής ανάδρασης α (s).

Μπορούμε να δούμε την απόκριση συχνότητας α(s) μέσω των διαγραμμάτων

BODE, καθώς επισής και την απόκριση του κέρδους α.

```
>> bode(a,'r')
>> figure;
>> a_norm = a/dcgain(a);
>> step(a_norm, 'r')
```

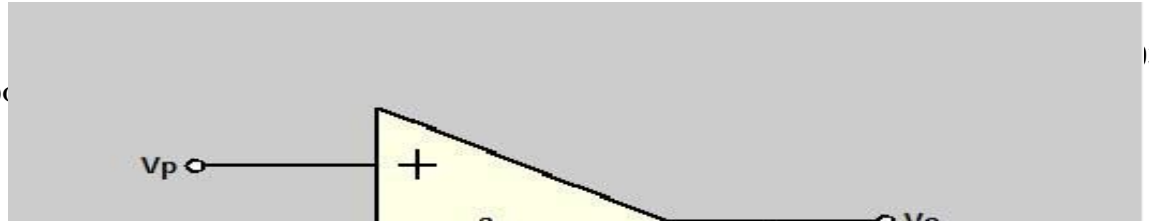


Εικόνα 47(α). Διαγράμματα BODE.

Εικόνα 47(β). Απόκριση Κέρδους.

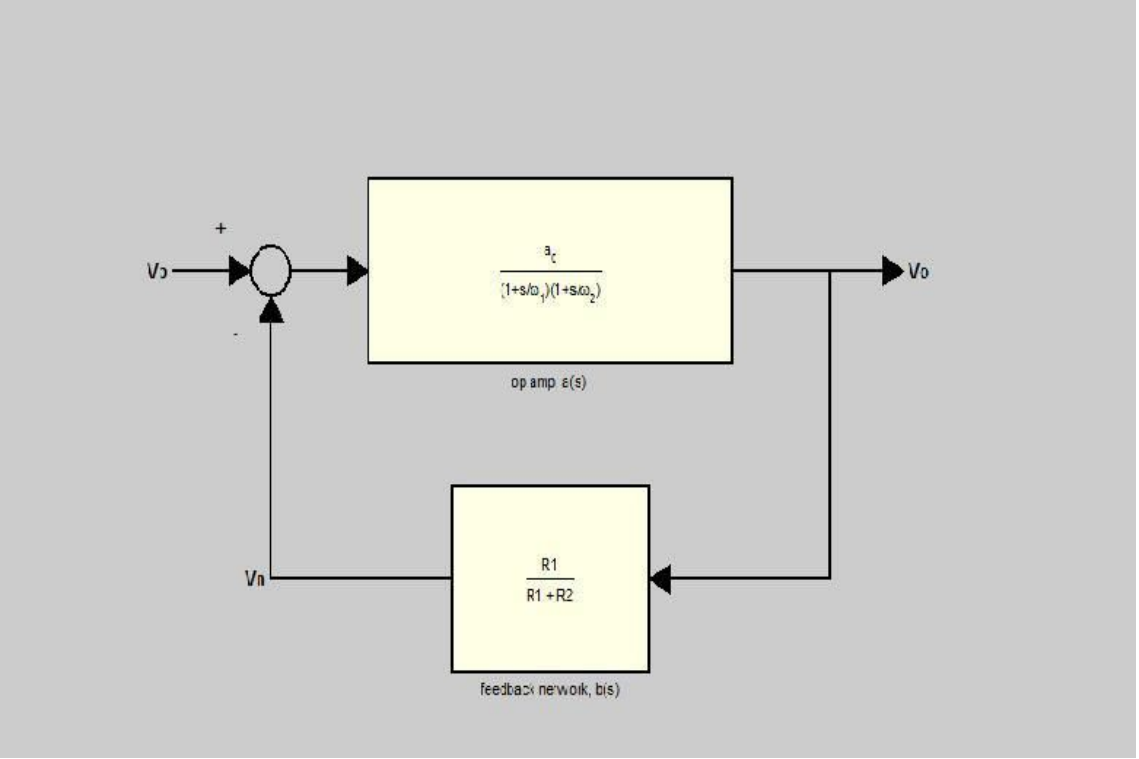
4.9.2. ΤΕΛΕΣΤΙΚΟΣ ΕΝΙΣΧΥΤΗΣ ΜΕ ΑΝΑΔΡΑΣΗ

Το παραπάνω σύστημα είναι η σχεδίαση του καθιερωμένου μοντέλου τελεστικού ενισχυτή. Τώρα θα το μετατρέψουμε σ' έναν τελεστικό ενισχυτή με ανάδραση. Για να το πετύχουμε αυτό, προσθέτουμε ένα ωμικό κύκλωμα ανάδρασης b(s). Το κύκλωμα του φαίνεται στην Εικόνα 48..



Εικόνα 48. Τελεστικός ενισχυτής με ανάδραση.

Το κύκλωμα ανάδρασης b(s) είναι απλά ένας διαιρέτης τάσης με είσοδο V_o και έξοδο V_n. Η συνάρτηση μεταφοράς για το b(s) είναι: $b = V_n / V_o = R1 / (R1 + R2)$. Επομένως τώρα μετατρέψαμε το συστημά μας σε σύστημα κλειστού βρόγχου όπως φαίνεται στο παρακάτω σχήμα, Εικόνα 49..



Εικόνα 49. Το σύστημα του τελεστικού ενισχυτή με κέρδος κλειστού βρόγχου.

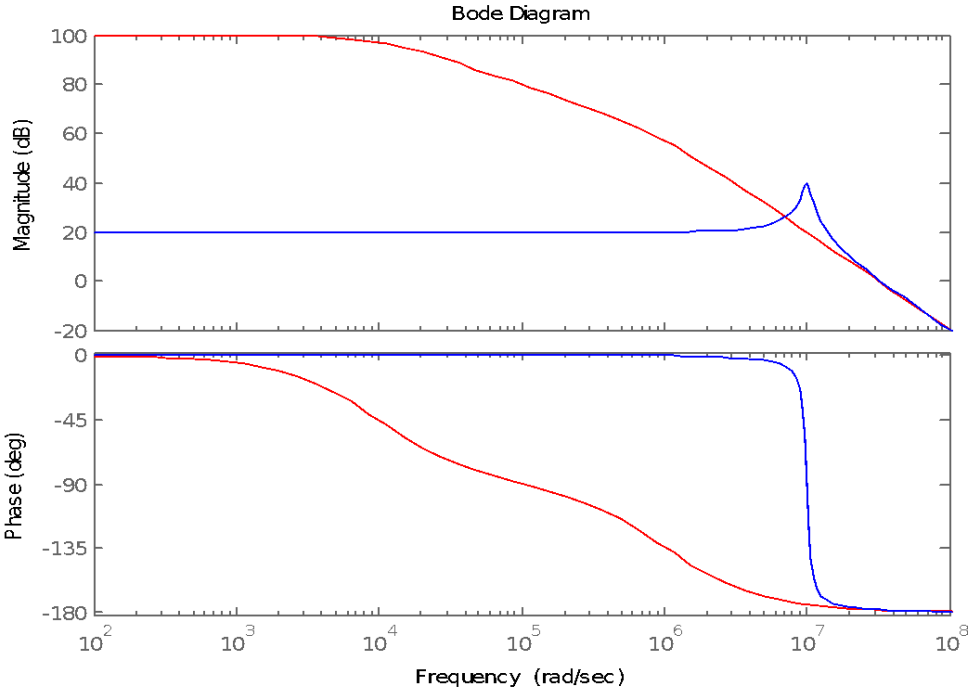
Θέλουμε έναν ενισχυτή με dc κέρδος να ισούται με 10 και R1 ίση με 10 kOhm. Επομένως οι εντολές του MatLab είναι οι παρακάτω. :

```
>> A0=10;
```

```
>> b=1/A0;
>> R1=10000;
>> R2=R1*(1/b-1)

R2 =

     90000

>>
```

Στη συνέχεια θα σχεδιάσουμε το σύστημα κλειστού βρόγχου με την εντολή feedback. Έτσι θα πάρουμε τις γραφικές παραστάσεις για τις αποκρίσεις συχνότητας και τα διαγράμματα BODE.

```
>> A=feedback(a,b);
>> figure;
>> bode(a,'r',A, 'b')
>>
```



Εικόνα 50. Διαγράμματα BODE ανοικτού και κλειστού βρόγχου.

Η χρήση αρνητικής ανάδρασης για να μειώσουμε το κέρδος χαμηλής συχνότητας (LF) οδήγησε στην αντίστοιχη άνοδο του εύρους ζώνης (bandwidth) του συστήματος. Η ανταλλαγή κέρδους / εύρους ζώνης είναι βασικό χαρακτηριστικό σε κυκλώματα τελεστικού ενισχυτή με ανάδραση. Εφόσον το κέρδος τώρα εξαρτάται από το κύκλωμα της ανάδρασης πρέπει να εξετάσουμε τη σχέση ευαισθησίας / απόκλισης του κέρδους σε σχέση με τον τελεστικό ενισχυτή ανοικτού βρόγχου. Η εντολές στο MatLab είναι οι ακόλουθες:

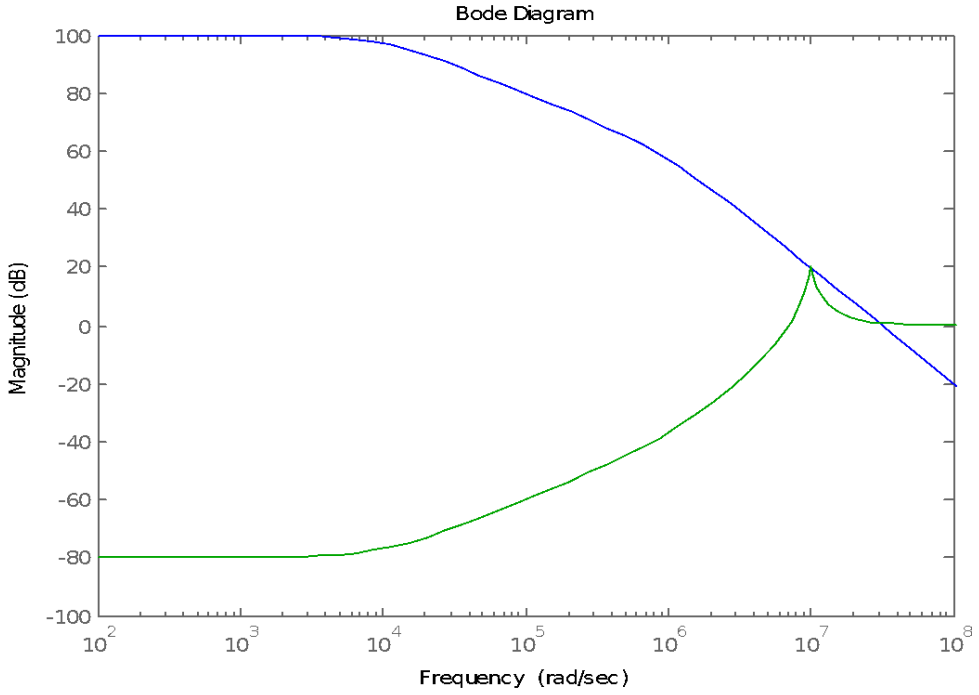
```
>> L = a*b

Transfer function:
          10000
--------------------------
1e-010 s^2 + 0.000101 s + 1

>>
```

[L(s) = a(s)b(s) → συνολικό κέρδος (ευαισθησία)]
[A(s) = το κέρδος κλειστού βρόγχου, α(s) = το κέρδος ανοικτού βρόγχου]
[Η Ευαισθησία του συστήματος δίνεται από τον τυπο: S = δA/A / δα/α = 1/ 1+α(s)b(s) = 1/1+L(s)].

Με λίγα λόγια, η ευαισθησία του συστήματος S(s), αναπαριστά την ευαισθησία του κέρδους κλειστού βρόγχου A(s) στις αποκλίσεις της συνάρτησης μεταφοράς ανοικτού βρόγχου. Η αντίστροφη σχέση μεταξύ S(s) και L(s) μας δίνει το κέρδος μη – ευαισθησίας. Στο MatLab έχουμε:

```
>> S=1/(1+L);
>> figure;
>> S=feedback(1,L);
```

```
>> bodemag(a, 'b', S, 'g')
>>
```



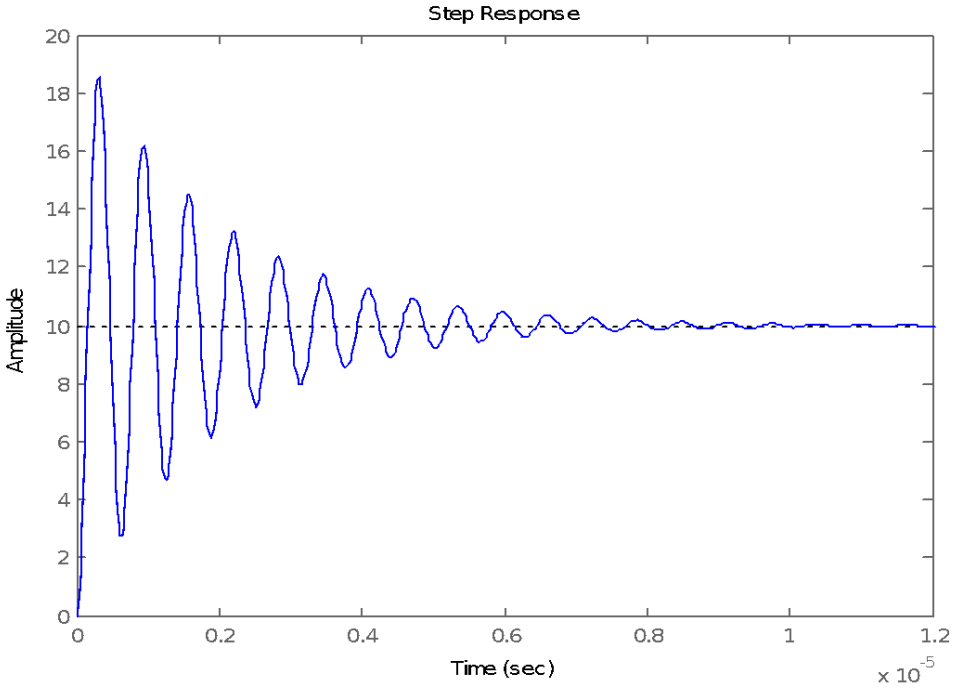
Εικόνα 51. Κέρδος μη – ευαισθησίας συστήματος.

Η πολύ μικρή χαμηλής συχνότητας ευαισθησίας (-80db) που φαίνεται στο σχήμα

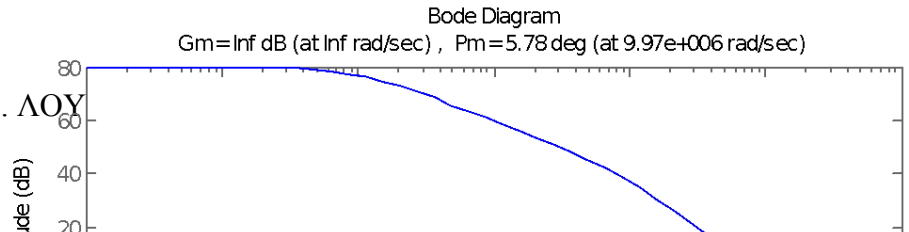
της παραπάνω εικόνας, Εικόνα 51, υποδηλώνει ότι το κέρδος κλειστού βρόγχου α(s) επηρεάζεται από τις αποκλίσεις του κέρδους κλειστού βρόγχου A(s). Τέτοιου είδους απόκλίσεις στο α(s) παρουσιάζεται συχνά λόγω τεχνικών αποκλίσεων, αλλαγή στη θερμοκρασία κ.τ.λ..

Με το MatLab έχουμε ακόμα την δυνατότητα να έχουμε την βηματική απόκριση του κέρδους ανοικτού βρόγχου σε σχέση με το κέρδος κλειστού βρόγχου α(s) – A(s). Ακόμα μπορούμε να εμφανίσουμε και το όριο σταθερότητας του συστήματος και έτσι να ξέρουμε μέχρι ποιο σημείο το συστημά μας θα συμπεριφέρεται με τον σωστό τρόπο.

```
>> step(A)
>> figure;
>> margin(L)
>>
```



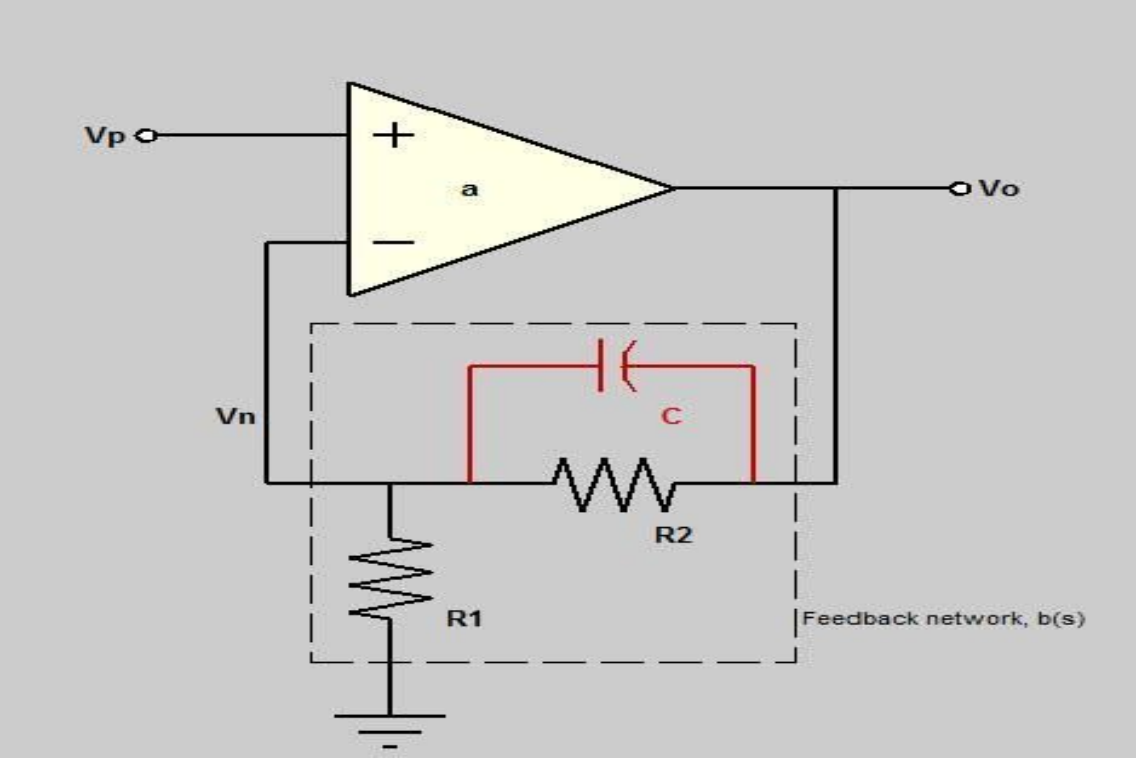
Εικόνα 52(α). Η βηματική απόκριση μεταξύ του κέρδους ανοικτού και κλειστού βρόγχου.



Εικόνα 52(β). Το όριο σταθερότητας του συστήματος μας.

4.9.3. ΤΕΛΕΣΤΙΚΟΣ ΕΝΙΣΧΥΤΗΣ ΜΕ ΑΝΑΔΡΑΣΗ – ΑΝΑΔΡΑΣΗ ΜΕ ΑΝΤΙΣΤΑΘΜΙΣΗ

Τώρα θα μετατρέψουμε το συστημά μας έτσι ώστε η ανάδραση που υπάρχει να οδηγεί σε αντιστάθμιση αυτού. Η συνδεσμολογία φαίνεται στη παρακάτω εικόνα, Εικόνα 53..

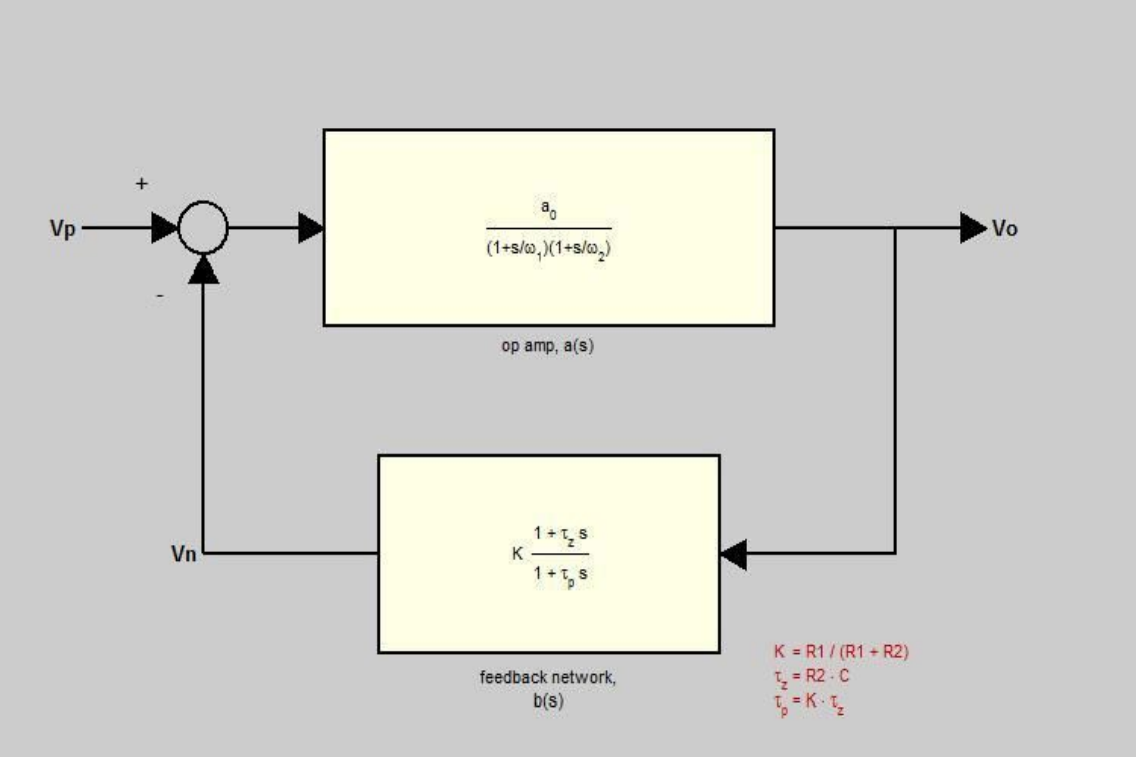


Εικόνα 53. Συνδεσμολογία τελεστικού ενισχυτή με ανάδραση που οδηγεί σε αντιστάθμιση του συστήματος.

Όπως βλέπουμε και στο σχήμα, το b(s) κομμάτι της συνδεσμολογίας το οποίο αντιστοιχούσε στη συνδεσμολογία της ανάδρασης που εξετάσαμε στην προηγούμενη ενότητα, έχει τροποποιηθεί καθώς έχει προστεθεί ένας πυκνωτής C παράλληλα με τον αντιστάτη της ανάδρασης R2. Η χωρητικότητα του πυκνωτή επιλέγεται ώστε να εισάγει μια φάση που να οδηγεί το b(s) κοντά στη συχνότητα αυξάνοντας το όριο φάσης του τελεστικού ενισχυτή.

Παρακατω απεικονίζεται το σχέδιο της καινούργιας συνάρτησης μεταφοράς του τελεστικού ενισχυτή με ανάδραση που οδηγεί σε αντιστάθμιση του συστήματος,

Εικόνα 54. Επίσης στο σχήμα φαίνονται και οι εξισώσεις αυτής.



Εικόνα 54. Καινούργια συνάρτηση μεταφοράς.

Μπορούμε να υπολογίσουμε στο περίπου την χωρητικότητα του πυκνωτή τοποθετώντας το μηδέν του b(s) σε 0db συχνότητας του L(s).:

```
>> [Gm,Pm,Wcg,Wcp]=margin(L);
>> C=1/(R2*Wcp)

C =

  1.1139e-012

>>
```

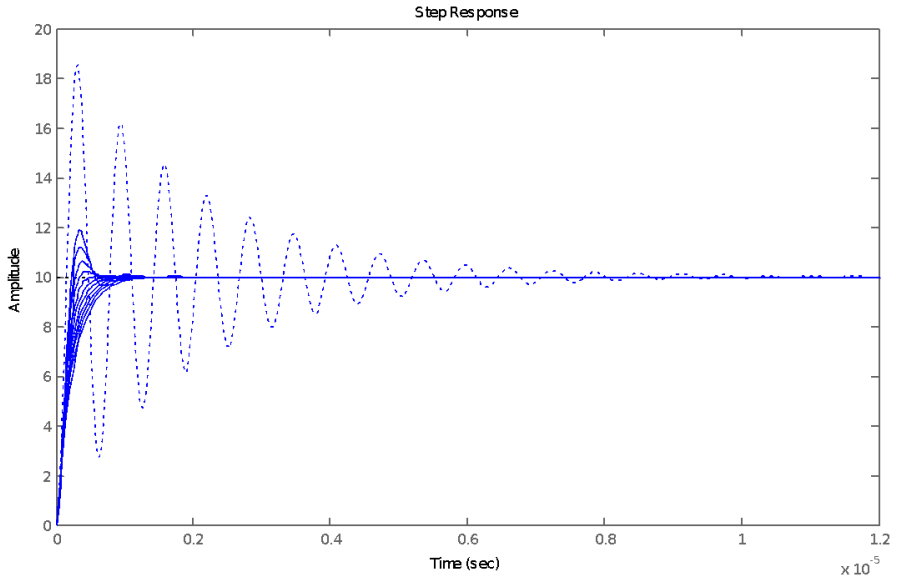
Για να μελετήσουμε την επίδραση του πυκνωτή στην απόκριση του ενισχυτή δημιουργούμε έναν πίνακα LTI μοντέλου για το b(s), ο οποίος θ' αναφέρεται σε αρκετές τιμές του C γύρω από την αρχική τιμή. Στο MatLab γράφουμε τις παρακάτω

εντολές.:

```
>> [Gm,Pm,Wcg,Wcp]=margin(L);
>> C=1/(R2*Wcp)

C =

  1.1139e-012

>>
>>
>> K=R1/(R1+R2);
>> C=[1:.2:3]*1e-12;
>> for n=1:length(C)
b_array(:,:,n)=tf([K*R2*C(n) K],[K*R2*C(n) 1]);
end
>>
>> A_array = feedback(a,b_array);
>> L_array = a*b_array;
>>
```

Με τις δύο τελευταίες εντολές δημιουργήσαμε τους LTI πίνακες για το A(s) και L(s). Μπορούμε να κάνουμε plot την βηματική απόκριση (step response) όλων των μοντέλων στο LTI πίνακα (A_array(s)) μαζί με A(s) χρησιμοποιώντας όπως και σε προηγούμενη περίπτωση την εντολή step του MatLab.:

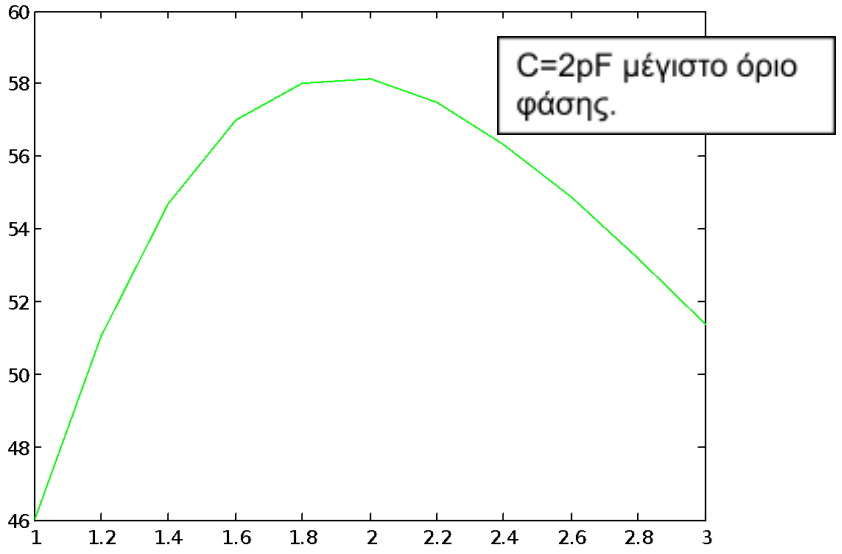
```
>> figure;
>> step(A, 'b:', A_array, 'b');
```



Εικόνα 55. Βηματική απόκριση όλων των μοντέλων του LTI πίνακα.

Βρίσκουμε τα όρια της φάσης (phase margins) για τον βρόγχο του κέρδους των πινάκων L_arrays, χρησιμοποιώντας την εντολή margin του MatLab. Επίσης μπορούν να γίνουν plot και σαν συνάρτηση C.:

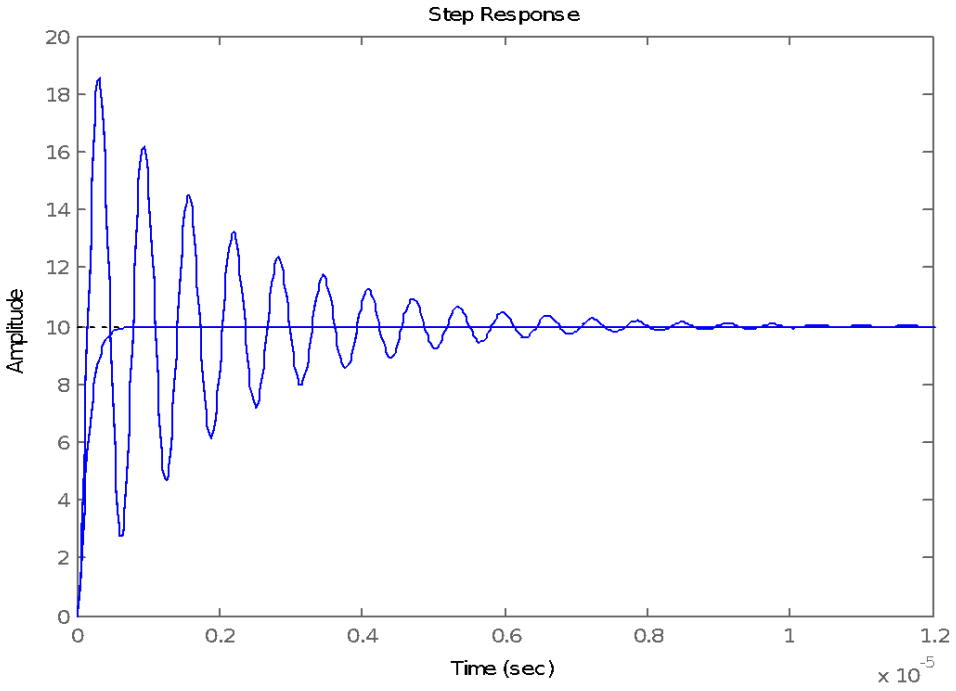
```
>> [Gm, Pm, Wcg, Wcp]=margin(L_array);
>> figure;
>> plot(C*1e12, Pm, 'g');
>>
```



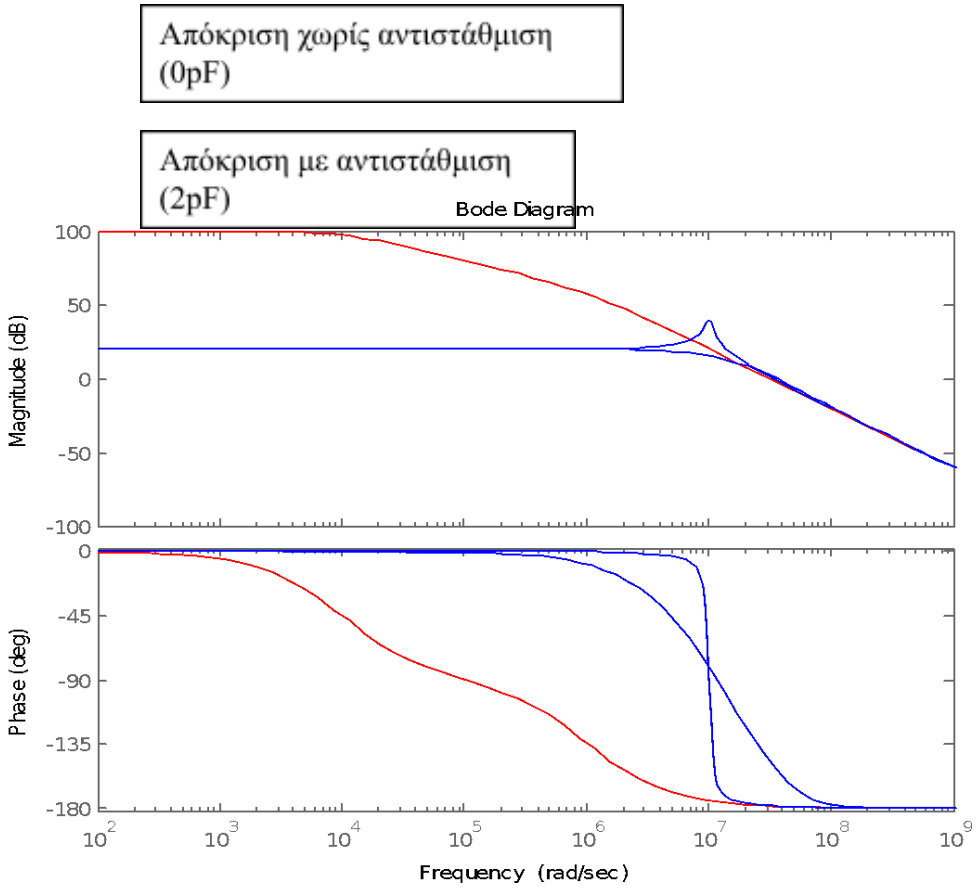
Εικόνα 56. Τα όρια της φάσης σαν συνάρτηση C.

Στην Εικόνα 56. βλέπουμε ότι τ' όριο της φάσης, τ' οποίο είναι 58deg., αποκτάται όταν ο πυκνωτής C φτάσει στα 2pF. Το μοντέλο που ανταποκρίνεται σ' αυτήν την τιμή είναι το έκτο μοντέλο στον πίνακα LTI, (b_arrays). Επομένως μπορούμε να κάνουμε plot στην βηματική απόκριση του συστήματος κλειστού βρόγχου για το συγκεκριμένο μοντέλο, διαλέγοντας index 6 από το LTI πίνακα A_array(s). Στη συνέχεια μπορούμε ν' απεικονίσουμε την απόκριση συχνότητας και για τα τρία μοντέλα (ανοικτού βρόγχου, κλειστού βρόγχου και εξισορροπημένου κλειστού βρογχου). Ακολουθούν οι εντολές στο MatLab.:

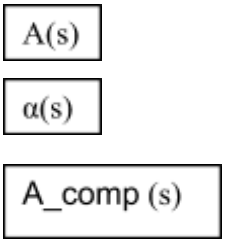
```
>> A_comp=A_array(:,:,6);
>> figure;
>> step(A, 'b', A_comp, 'b')
>> figure;
>> bode(a, 'r', A, 'b', A_comp, 'b')
```



Εικόνα 57. Βηματική Απόκριση του συστήματος κλειστού βρόγχου.



Εικόνα 58. Απόκριση συχνότητας και των 3 μοντέλων.



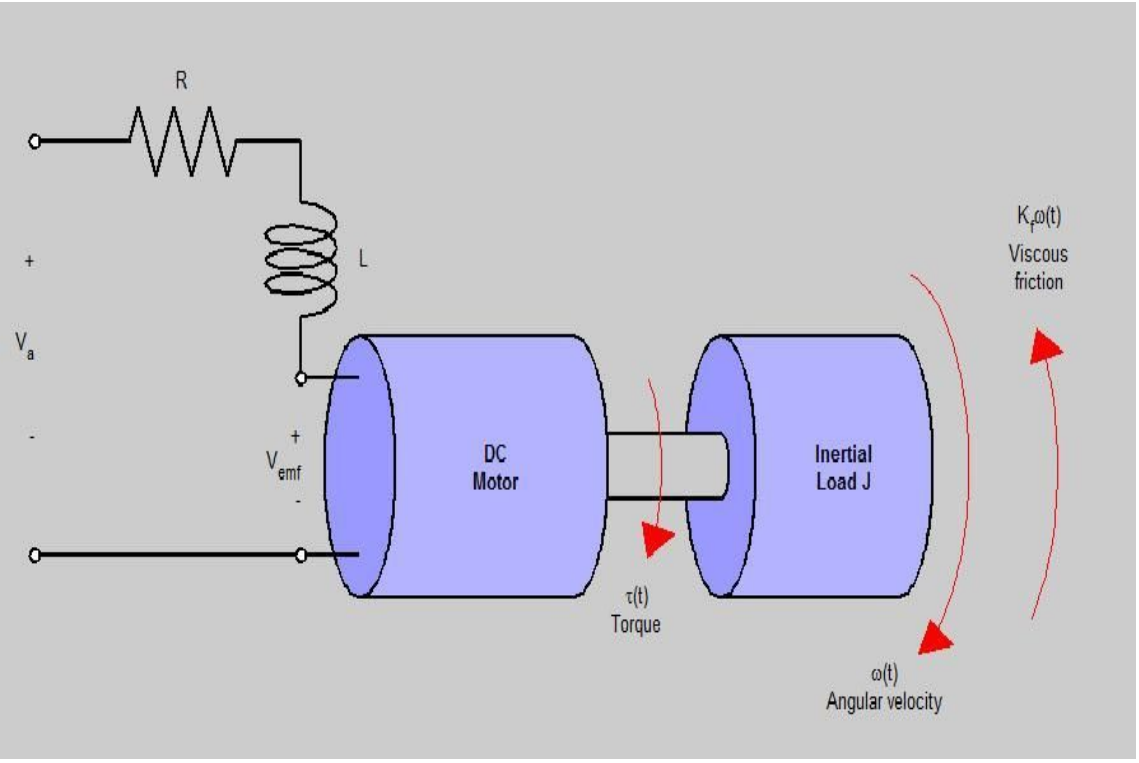
Ανακεφαλαιώνοντας, με την βοήθεια του MatLab , δημιουργήσαμε ένα δίκτυο ωμικής αντίστασης (R1, R2) το οποίο επιλέχθηκε για ν' αποδώσει κέρδος ίσο με 10 (20db) σ' έναν ευρείας ζώνης τελεστικό ενισχυτή. Στο σύστημα προστίθεται συνδεσμολογία κλειστού βρόγχου αντιστάθμισης. Επομένως ανάδραση οδηγεί σε αντιστάθμιση του συστήματος ώστε να χαμηλώσει τον βρόγχο κέρδους κοντά στη συχνότητα διαστάυρωσης. Η τιμή για τον πυκνωτή αντιστάθμισης C βελτιστοποιείται για να παρέχει το μεγαλύτερο όριο φάσης περίπου 58 βαθμών.

4.10. ΧΡΗΣΗ MATLAB ΓΙΑ ΤΗΝ ΟΔΗΓΗΣΗ ΚΙΝΗΤΗΡΑ ΧΩΡΙΣ ΕΥΑΙΣΘΗΣΙΑ

Στο προηγούμενο κεφάλαιο αναφερθήκαμε στην χρήση του DYMOLA για την οδήγηση κινητήρα. Το ίδιο το εργαλείο μας δίνει την δυνατότητα να προσομειώσουμε το μοντέλο και να δούμε πως δουλεύουν σε γενικές γραμμές τέτοιου είδους συστήματα. Την ίδια δυνατότητα μας δίνει όμως και το MatLab χρησιμοποιώντας τα toolboxes του, μπορούμε να τράξουμε ένα μοντέλο κινητήρα. Έτσι θα δούμε ομοιότητες και διαφορές μεταξύ των δύο, αλλά και ποιο απ' αυτά είναι κατάλληλο ανάλογα με την φύση της εργασίας που καλούμαστε να κάνουμε.

4.10.1 Μοντέλο Απόρριψης Ευαισθησίας Κινητήρα (Disturbance Rejection)

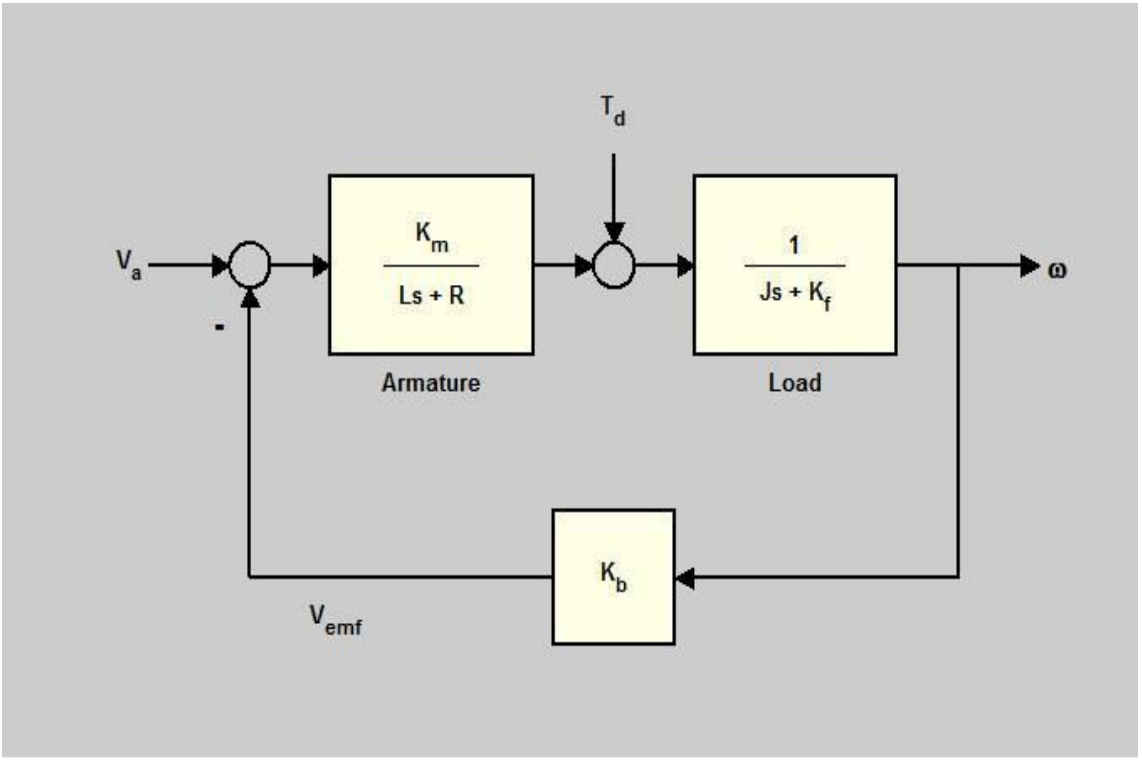
Στην Εικόνα 59. που ακολουθεί φαίνεται το μοντέλο ενός κινητήρα στον οποίο η διαφορά δυναμικού Va ελέγχει την γωνιακή ταχύτητα του ζεύγους του κινητήρα.



Εικόνα 59. Το μοντέλο του κινητήρα όπως παρουσιάζεται στο demo του MatLab.

Στο συγκεκριμένο demo το MatLab μας δείχνει πώς να υλοποιήσουμε τρείς διαφορετικές τεχνικές (feedrorward command, integral feedback control, LQR regulation), ώστε να μειώσουμε την ευαισθησία που εμφανίζεται στο κινητήρα μας λόγω της ροπής των δυνάμεων.

Πιο συγκεκριμένα, στο απλοποιημένο μοντέλο που παρουσιάζεται παραπάνω το ζεύγος δυνάμεων του κινητήρα δημιουργεί ροπη, Td, η οποία προκαλεί την ευαισθησία στο σύστημα. Πρέπει να μειώσουμε τις μεταβολές στην ταχύτητα που δημιουργούνται από την ροπή ώστε να εχουμε καλύτερη απόδοση και λιγότερη ευασθησία. Στην Εικόνα 60., φαίνεται το σύστημα ανάδρασης του μοντέλου μας.



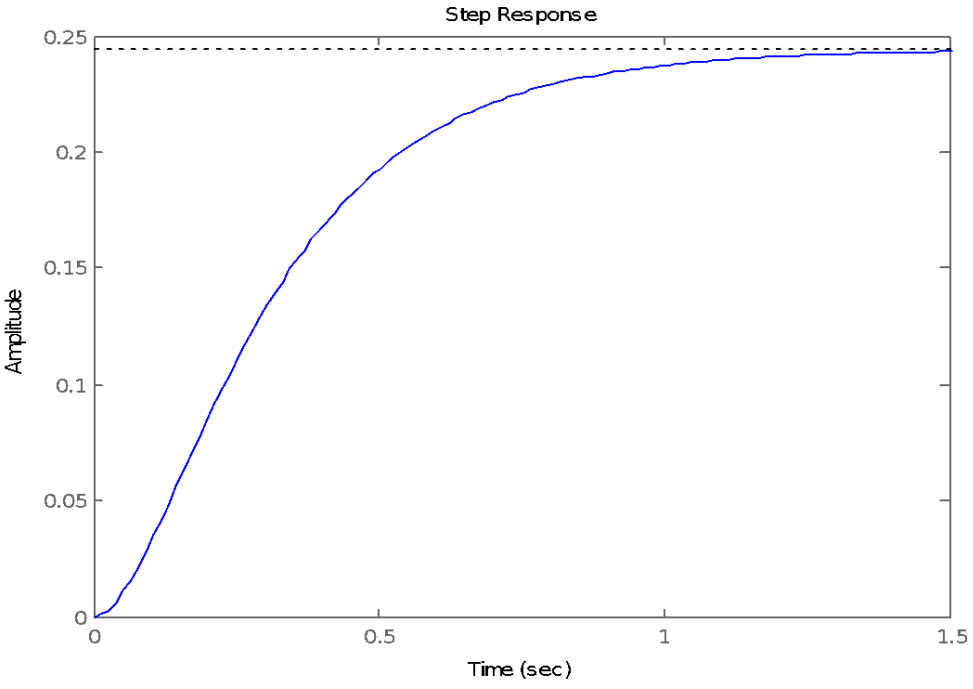
Εικόνα 60. Σύστημα ανάδρασης του κινητήρα.

Στο παραδειγμά μας οι φυσικές σταθερές είναι οι παρακάτω, οι οποίες δίνονται σαν είσοδοι στο MatLab. Επίσης, υπάρχουν δύο είσοδοι Va, Td και μια έξοδος W για την συνάρτηση μεταφοράς (transfer function). Ακολουθεί ο κώδικας:

```
>> R=2.0;
>> L=0.5;
```

```
>> Km = 0.1;
>> Kb = Km;
>> Kf=0.2;
>> J=0.02;
>> h1=tf(Km,[L R]);
>> h2=tf(1,[J Kf]);
>> dcm=ss(h2)*[h1,1];
>> dcm=feedback(dcm,Kb,1,1);
>> step(dcm(1))
```

Η τελευταία εντολή μας εμφανίζει την γραφική παράσταση της μεταβολής της γωνιακής ταχύτητας του κινητήρα σε μια βηματική αλλαγή.



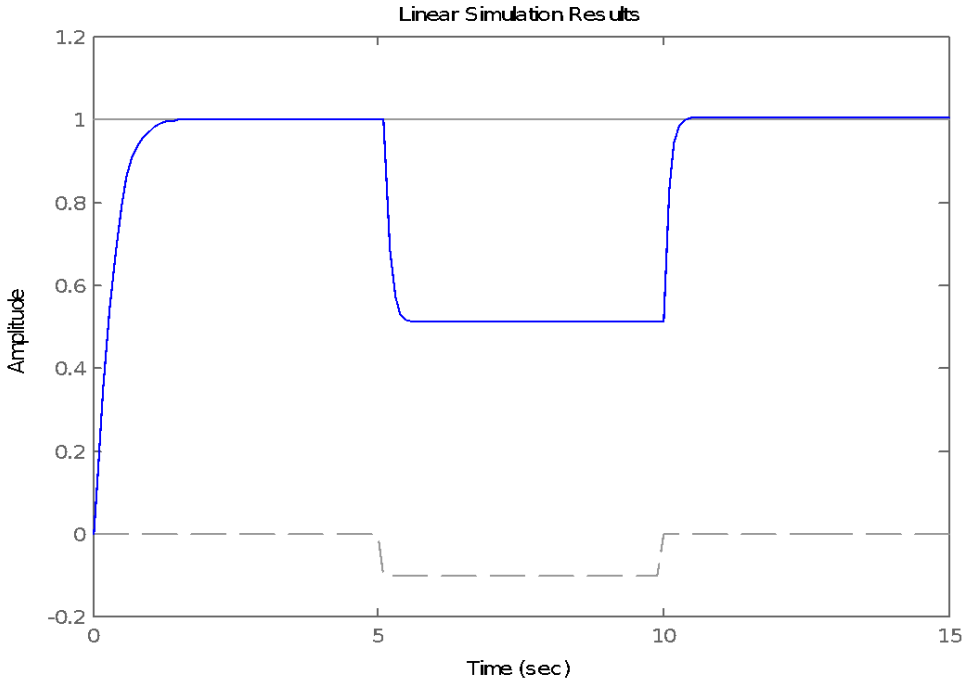
Εικόνα 61. Βηματική απόκριση για την γωνιακή ταχύτητα του κινητήρα.

- *Feedforward Control – Ελεγχος Πρώιμης Τροφοδοσίας*

Μπορούμε να χρησιμοποιήσουμε την τεχνική *feedforward control* ώστε να μετατρέψουμε την γωνιακή ταχύτητα του κινητήρα ω σε ταχύτητα με καθορισμένη

τιμή ω_ref. Το κέρδος της τροφοδοσίας Kff θα πρέπει να είναι ανταποδοτικό του κέρδους ρεύματος DC από την Va σε ω. Επόμένως, με τον παρακάτω κώδικα στο MatLab θα δούμε πως αυτή η τεχνική χειρίζεται την ευαισθησία του κινητήρα.:

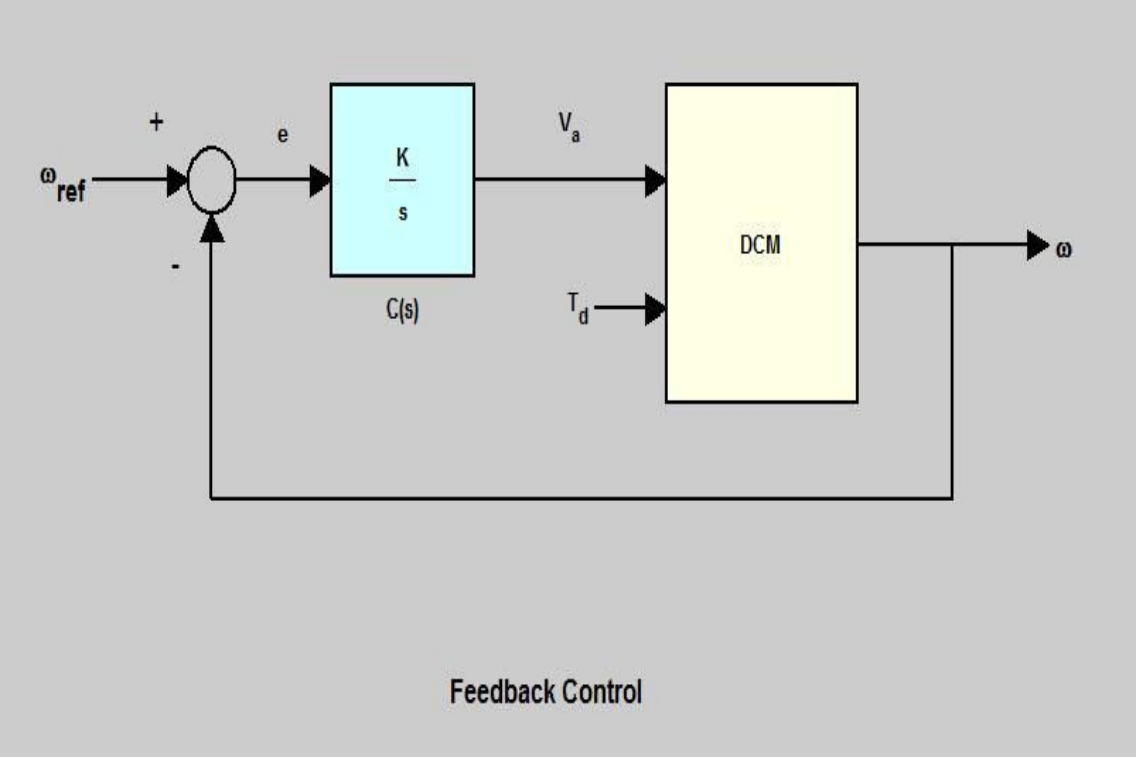
```
>> kff=1/dcgain(dcm(1))

kff =

    4.1000

>> t=0:0.1:15;
>> Td=-0.1*(t>5 & t<10);
>> u=[ones(size(t)); Td];
>> cl_ff=dcm*diag([kff,1]);
>> lsim(cl_ff,u,t)
>>
```



Εικόνα 62. Γραμμική απεικόνιση της ευαισθησίας του συστήματος.

Η παραπάνω γραφική παράσταση μας βοηθάει να κατανοήσουμε πως η feedforward τεχνική χειρίζεται την ευαισθησία. Έτσι προσομοιώνουμε την απόκριση σαν μια βηματική εντολή ωref = 1, με ευαισθησία Td = - 0,1 Nm ανάμεσα στο χρονικόδιάστημα t = 5 και t = 10 sec. Όπως φαίνεται και από την Εικόνα 62., η feedforward τεχνική δεν μπορεί να χειριστεί την ευαισθησία στο σύστημα του κινητήρα.

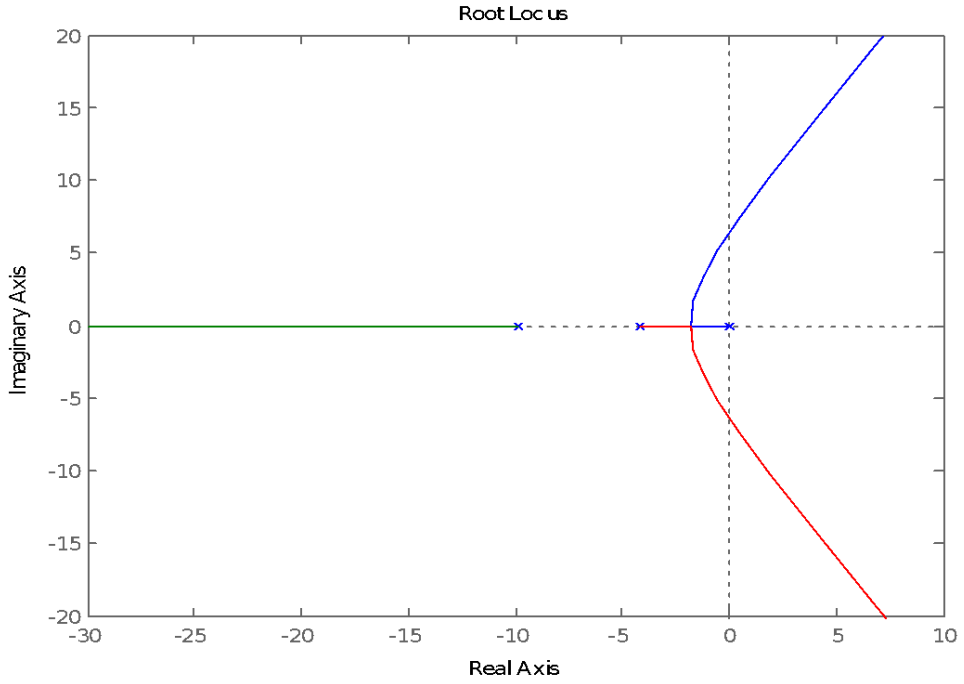
Θα προσπαθήσουμε να εφαρμόσουμε μια δομή ελέγχου ανάδρασης στο σύστημά μας, ώστε να έχουμε λάθος μηδενικής κατάστασης χρησιμοποιώντας ολοκλήρωμα ελέγχου. Το block ανάδρασης φαίνεται στην παρακάτω εικόνα, Εικόνα 63..



Εικόνα 63. Block ανάδρασης λάθους μηδενικής κατάστασης.

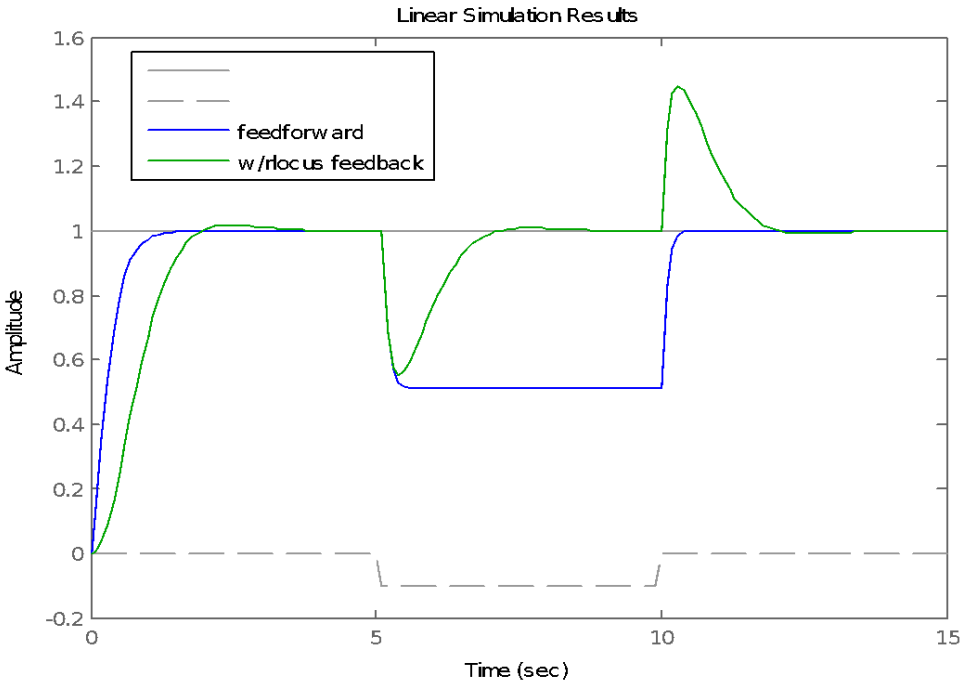
Το λάθος μηδενικής κατάστασης στην ανάδραση ελέχου δίνεται από τον τύπο C(s) =K / s. Για να καθορίσουμε το κέρδος K, χρησιμοποιούμε την εντολή rlocus, η οποία εφαρμόζεται στον ανοικτό βρόγχο 1/s * transfer(Va - ω). Έτσι στο MatLab έχουμε:

```
>> rlocus(tf(1,[1 0])*dcm(1))
```



Εικόνα 64. Καθορισμός του κέρδους με την εντολή rlocus.

```
>> K=5;
>> C=tf(K, [1 0]);
>> cl_rloc = feedback(dcm*append(C,1),1,1,1);
>> figure;
>> lsim(cl_ff,cl_rloc,u,t)
>>
```

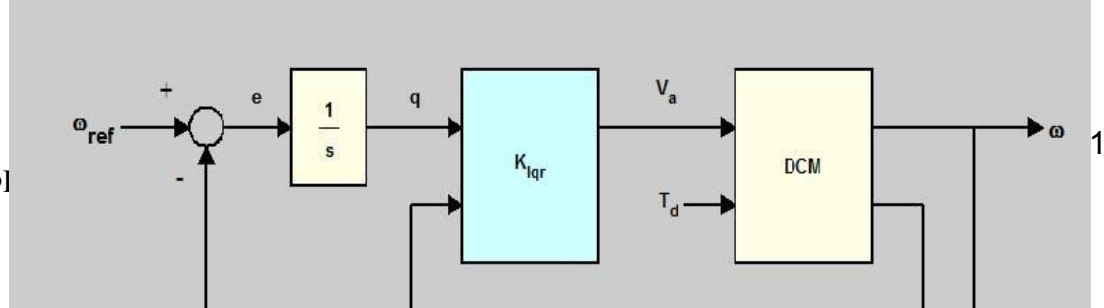


Εικόνα 65. Τα αποτελέσματα που δίνει η κάθε μια τεχνική στην ίδια γραφική.

Όπως βλέπουμε και από την γραφική απεικόνιση της Εικόνας 65, η τεχνική της εντολής rlocus δίνει καλύτερα αποτελέσματα στο ν' απορρίπτει τις διαταραχές που δημιουργούνται από τον φόρτο της ροπής των δυνάμεων που ασκεί ο κινητήρας.

- *Linear Quadratic Regulator (LQR) - Τεχνική Γραμμικού Ρυθμιστή 2ου Βαθμού*

Η προηγούμενη τεχνική βελτίωσε το σύστημα μας στο πως δέχεται και διαχειρίζεται την ευαισθησία αλλά όχι σε ικανοποιητικό βαθμο. Έτσι λοιπόν σχεδιάζουμε έναν γραμμικό ρυθμιστή δευτέρου βαθμού για την δομή ανάδρασης. Το σχήμα του φαίνεται στην παρακάτω εικόνα, Εικόνα 66..



Εικόνα 66. Σύστημα LQR αναδρασης.

Σε σχέση με το σχήμα ανάδρασης της προηγούμενης τεχνικής, εδώ έχουμε μια επιπλέον συνάρτηση κόστους (K_{iqr}) καθώς και στο ολοκλήρωμα λάθους η LQR χρησιμοποιεί ένα επιπλέον διάνυσμα x = (i,ω) για να δημιουργήσει την τάση V_a .

$Va = K1 * \omega + K2 * \omega/s + K3 * i$

i είναι το ρεύμα του οπλισμού του κινητήρα

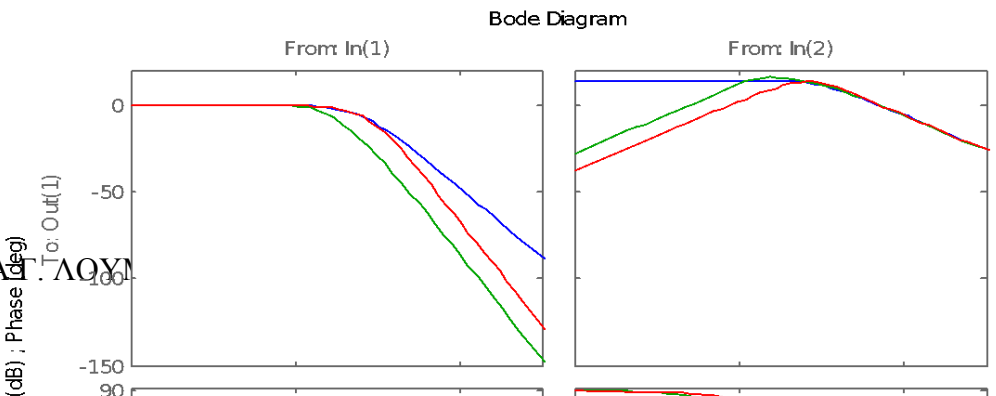
LQR συνάρτηση κόστους:

$$\int_0^{\infty} (20\, q(t)^2 + \omega(t)^2 + 0.01\, V_a(t)^2)dt$$

όπου $q(s) = \omega(s) / s$

Πίνακας 1.3. Συναρτήσεις της ανάδρασης.

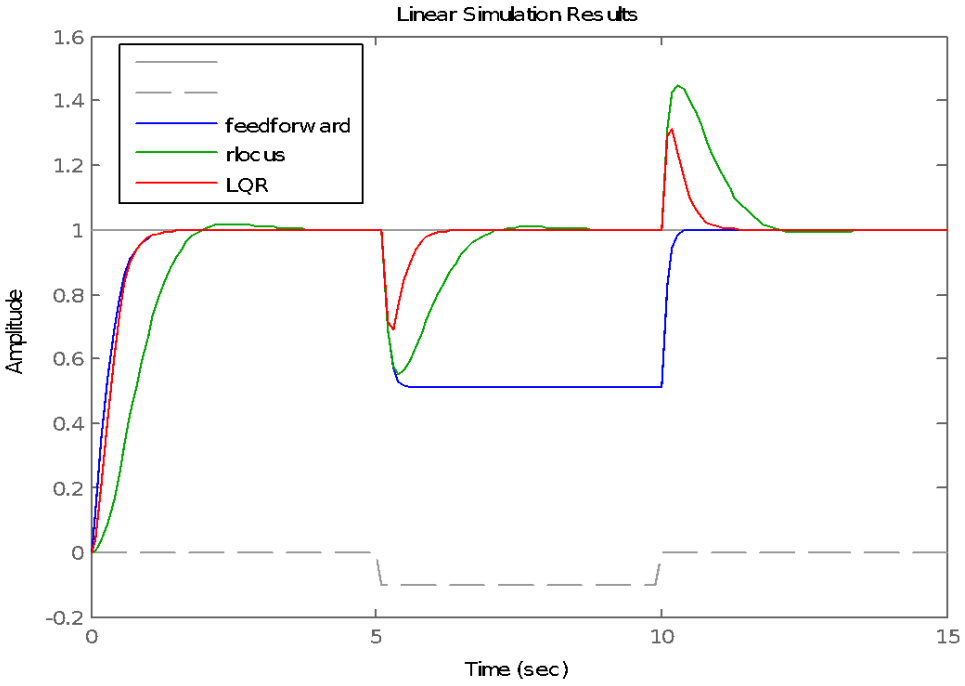
```
>> dc_aqmg = [1; tf(1, [1 0])]*dcm(1);
>> figure;
>> k_lqr = lqry(dc_aqmg, [1 0;0 20], 0.01);
>> P = augstate(dcm);
>> C = k_lqr *append(tf(1,[1 0]),1,1);
>> OL = P*append(C,1);
>> CL = feedback(OL, eye(3), 1:3, 1:3);
>> cl_lqr = CL(1, [1 4]);
>> bode(cl_ff,cl_rloc,cl_lqr)
>>
```



Εικόνα 67. Τα Bode διαγράμματα και για τις τρεις τεχνικές.

Συγκρίνουμε τ' αποτελέσματα και των τριών τεχνικών μαζί απεικονιζοντάς τα στο ίδιο plot. Παρακάτω ο κώδικάς στο MatLab.:

```
>> figure;
>> lsim(cl_ff,cl_rloc,cl_lqr,u,t)
>>
```



Εικόνα 68. Το plot με τις τρεις τεχνικές μαζί.

Από την Εικόνα 68. βλέπουμε ότι η τεχνική LQR – λόγω των επιπλέον βαθμών ελευθερίας – αποδίδει καλύτερα στο ν' απορρίπτει τις διαταραχές που δημιουργούνται στο σύστημα του κινητήρα λόγω της ροπής των δυνάμεων που ασκείται σ' αυτόν.

ΚΕΦΑΛΑΙΟ 5

5.1 ΓΕΝΙΚΟ ΣΥΜΠΕΡΑΣΜΑ

Έχοντας εξετάσει το πώς λειτουργεί ένα μοντέλο κινητήρα χρησιμοποιώντας το DYMOLA και το MatLab μπορούμε να βγάλουμε ένα γενικότερο συμπέρασμα για το πώς αποδίδει το ένα ή το άλλο εργαλείο καθώς και ποιό χρησιμοποιείται καλύτερα και που. Όπως έχουμε αναφέρει και προηγουμένως, η χρήση του ενός ή του άλλου εξαρτάται αποκλειστικά από τον χρήστη και την δουλειά που αυτός θέλει να κάνει.

Σε γενικές γραμμές όμως, το DYMOLA είναι πιο εύχρηστο στην σχεδίαση του μοντέλου με βάση τις πραγματικές προδιαγραφές. Για παράδειγμα, το μοντέλο του κινητήρα δημιουργήσαμε με την βοήθεια του DYMOLA θα αντιπροσωπεύει τον πραγματικό κινητήρα που θα δημιουργηθεί για να χρησιμοποιηθεί στο project μας, π.χ. σε κάποιο όχημα.

Ειδικότερα, το DYMOLA προσφέρει αντικειμενοστραφή μοντελοποίηση και την δυνατότητα επαναχρησιμοποίηση μοντέλων της βιβλιοθήκης. Το μεγαλύτερο

πλεονεκτημά που δίνει σ' έναν χρήστη, ειδικά σ' έναν μηχανικό, είναι η δημιουργία και προσομοίωση σε παραγματικό χρόνο φυσικών μοντέλων με μεγάλη ευκολία.

Συνθέτει γραφικά μοντέλα με την χρήση των εικονιδίων – δομικά στοιχεία – της βιβλιοθήκης του. Επίσης, οι βιβλιοθήκες που περιέχουν τις κλάσεις των μοντέλων εφαρμόζουν κληρονομικότητα σ' αυτές (ιεραρχική δομή). Το DYMOLA κάνει επεξεργασία συμβολικών εξισώσεων, αλλά παρ' όλα αυτά ο χρήστης δεν χρειάζεται να μετατρέψει τις εξισώσεις σε εντολές εκχώρησης. Έχουμε επίσης την δυνατότητα 3-D απεικόνισης των μοντέλων μας.

Το DYMOLA, επομένως, είναι ένα εργαλείο ανάπτυξης και προσομοίωσης μοντέλων με εύκολο τρόπο. Παρά τις άπειρες δυνατότητες που προσφέρει στον χρήστη για την σχεδίαση τέτοιων συστημάτων, δεν ενδύκνυται για την μαθηματική απεικόνιση και κατανόηση των μοντέλων. Παρ' όλα αυτά μπορεί να μετατρέψει τα φυσικά μοντέλα που ο χρήστης έχει σχεδιάση μ' αυτό σε κώδικα MatLab και να πάρει τις μαθηματικές απαντήσεις που ψάχνει.

Και φτάνουμε στο δεύτερο εργαλείο με το οποίο ασχοληθήκαμε στην παρούσα πτυχιακή εργασία. Το MatLab μας βοηθάει να κατανοήσουμε την μαθηματική και φυσική λειτουργία του μοντέλου μας. Χρησιμοποιείται παγκοσμίως από μηχανικούς και επιστήμονες για ν' αναλύει και να σχεδιάζει συστήματα κυρίως σε μαθηματικό επίπεδο. Με το MatLab μπορούμε να υπολογίσουμε εύκολα και σίγουρα τι τιμή θα πρέπει να έχει για παράδειγμα το κέρδος της ροπής των δυνάμεων ενός κινητήρα ή την τιμή του ρεύματος.

Ακόμα, με το MatLab μπορούμε εύκολα να υπολογίσουμε τις συναρτήσεις μεταφοράς ανοικτού και κλειστού βρόγχου με βάση των οποίων θα πρέπει να λειτουργεί ο κινητήρας μας. Όπως γνωρίζουμε ο υπολογισμός τέτοιων συναρτήσεων είναι πολύ σημαντικός στο αντικείμενο εργασίας ενός μηχανικού αλλά ορισμένες φορές δύσκολος να γίνει στο χαρτί. Όπως είναι φανερό θα πρέπει να γίνεται χωρίς λάθη και ευτυχώς για μας το MatLab εξασφαλίζει την λύση τέτοιων μαθηματικών υπολογισμών άμεσα και με τον σωστό τρόπο.

Σε γενικές γραμμές λοιπόν το MatLab είναι ένα εργαλείο σχεδιασμένο με τέτοιο τρόπο ώστε να λύνει μηχανικά και επιστημονικά προβλημματα μαθηματικού φάσματος πολύ γρήγορα κι εύκολα.

Έχει γλώσσα βασισμένη σε πίνακες η οποία υποστηρίζει έναν εύκολο τρόπο ανάπτυξης υπολογιστικών μαθηματικών. Με το MatLab ο χρήστης έχει την δυνατότητα ανάπτυξης γραφικών για την απεικόνιση και αναπαράσταση δεδομένων.

Το MatLab μπορούμε να πούμε ότι είναι το εργαλείο που «μιλάει» μαθηματικά και χρησιμποιείται σε πολλούς διάφορους επαγγελματικούς κλάδους, από την Μηχανική και την Ρομποτική μέχρι την Οικονομία και την Εκπάιδευση. Με αυτό είναι εύκολη η χρήση της γραμμικής άλγεβρας μέσω υπολογιστή και επομένως η ανάπτυξη μαθηματικού κώδικα.

Επομένως καταλήγουμε στο συμπέρασμα και τα δυο είναι πολύ εύχρηστα και σημαντικά εργαλεία για έναν Μηχανικό Η/Υ, και όχι μόνο, τα οποία του δίνουν άπειρες δυνατότητες ανάπτυξης, κατανόησης και προσομοίωσης μοντέλων φυσικών και όχι μονο συστημάτων.

Το DYMOLA και το MatLab έχουν πολλά κοινά σημεία και δυνατότητες τα οποία αναπτύξαμε στις παραπάνω σελίδες αλλά και σημαντικές διαφορές που τα καθιστούν διαφορετικά και απαραίτητα για διαφορετικούς λόγους. Λόγω των ομοιοτήτων τους δίνουν την δυνατότητα στον χρήστη να εργαστεί ταυτόχρονα και με τα δυο πάνω στο ίδιο project. Όμως το καθένα πηγαίνει ένα βήμα παραπάνω δίνοντας επιπλέον πληροφορίες και κάνοντας λεπτομερή επεξεργασία σε συγκεκριμένα κομμάτια του μοντέλου. Η δουλειά του χρήστη γίνεται πιο εύκολή και έχει μεγαλύτερη ακρίβεια και ευελιξία στο πως θα επεξεργαστεί το σύστημα του. Έτσι ώστε δίνουν μια άμμεση και ρεαλιστική προοπτική για το πώς θα «χτιστει» ένα μοντέλο όσο το δυνατόν πιο γρήγορα αλλά το κυριότερο πιο κοντά στο πραγματικό πρότυπο που εν τέλει είναι και το ζητούμενο.

ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] https://www.modelica.org/association

[2] https://www.modelica.org/documents/ModelicaSpec33Revision1.pdf

[3]

https://www.openmodelica.org/images/docs/tutorials/modelicatutorialfritzson.pdf

[4] "Principles of Object Oriented Modeling and Simulation with Modelica 2.1", Peter Fritzson, Wiley-IEEE Press, 2004

[5] "Introduction to Modeling and Simulation of Technical and Physical Systems with Modelica", Peter Fritzson, Wiley-IEEE Press, September 2011

[6] www.openmodelica.org

[7]
http://www.mathworks.com/company/newsletters/articles/the-origins-of-matlab.html

[8] "Σήματα & Συστήματα με MATLAB", Αλέξης Παλαμιδης – Αναστασία Βελώνη, Σύγχρονη Εκδοτική, Αθήνα 2008

[9] http://www.matlabtips.com/what-is-matlab-where-how-and-when-to-use-it/

[10] http://www.mathworks.com/products/matlab/

[11] "MATLAB Programming for engineers, Fifth Edition", Stephen J. Chapman, Cengage Learning, Australia 2015

[12] http://www-rohan.sdsu.edu/doc/matlab/toolbox/simulink/ug/quick_start.html

[13] http://www.mathworks.com/products/simulink/

[14]
https://www.doria.fi/bitstream/handle/10024/98924/kote_tutkimusraportti_52.pdf?sequence=2

[15] Hilding Elmqvist, Dag Brück, Sven Erik Mattsson, Hans Olsson and

Martin Otter, Dymola for Multi-Engineering Modeling and Simulation, Paper presented at the 2nd International Modelica Conference, March 18 – 19, 2002, Germany

[16] http://www.Modelica.org/Conference2002/papers.shtml

[17] Comparison Study of two competing models of an all mechanical power transmission system, research 52, Professor Heikki Martikka, M.Sc Ming Ye, Lappeenranta University of Technology Department of Mechanical Engineering, Finland, 2004

[18] Σήματα, Συστηματα και Κυκλώματα συνεχούς χρόνου, Ηρακλής Γ. Δημόπουλος, Εκδόσεις Unitext, Δεύτερη Έκδοση, 2009

[19] Authors: A. DiVergilio Copyright 1986-2002 The MathWorks, Inc.,Revision: 1.9,

Date: 2002/04/08 16:22:23

[20] http://book.xogeny.com/behavior/equations/mechanical/

[21] Συστήματα Αυτόματου Ελέγχου, Ανάλυση & Προσομοίωση, Αναστασία Ν. Βελώνη, Εκδόσεις Τζίολα, 2011.

[22] Σχεδίαση Ολοκληρωμένων Συστημάτων CMOS VLSI τέταρτη έκδοση, Neil H. E. Weste, David M. Harris, Παπασωτηρίου Εκδόσεις, 2011

[23] Ψηφιακά Συστήματα Αυτομάτου Ελέγχου, Αναστασία Ν. Βελώνη, Αυτοέκδοση, 2014

[24] Hybrid System Modeling using Modelica anad DYMOLA with applications, Yau Hei Chan, University of Wisconsin, Madisson, 2005

[25] Principals of Object – Oriented Modeling and Simulation with Modelica 3.3 – A Cyber – Physical Aproach, Second Edition, Peter Fritzson, IEEE Press, Wiley