KAUNO TECHNOLOGIJOS UNIVERSITETAS

INFORMATIKOS FAKULTETAS

KOMPIUTERIŲ KATEDRA

Gediminas Bačkys

Mechatroninių sistemų modelių sudarymas ir tyrimas

Magistro darbas

Darbo vadovas doc. V. Petrauskas

Kaunas
2003

Turinys

1. ĮVADAS

Pasaulis įžengė į poindustrinę plėtros stadiją. ES yra užsibrėžusi sukurti žinių visuomenę per dešimtį metų, kad galėtų atgauti technologinio ir ekonominio pirmavimo pasaulyje pozicijas. Siekis sukurti žinių visuomenę ir tuo pagrindu spartinti pažangą visose gyvenimo srityse tampa pripažintu ir Lietuvos prioritetu. Šiuo metu turbūt nėra kito tokio svarbaus ir neatidėliotino žingsnio plėtojant žinių visuomenę kaip Aukštųjų technologijų plėtros (ATP) programa; ji numatyta Lietuvos mokslo ir technologijų baltojoje knygoje bei ūkio ilgalaikės plėtros strategijoje (18).

Mechatroninės sistemos, sintezuojančios mechaninius, elektromechaninius, elektroninius, kontrolės ir valdymo elementus, yra daugelio technologinių įrenginių ir kitų aukštos pridėtosios vertės gaminių pagrindas. Bene ryškiausias tokių gaminių pavyzdys yra įvairios pavaros, jutikliai ir valdikliai, sukurti panaudojant „protingas“ medžiagas, pasižyminčias adaptyvumu ir reaguojančias į aplinką, turinčias lengvai valdomus parametrus (naujos medžiagos yra vienas iš ES technologinės plėtros prioritetų). Pastaruoju metu mechatronikos koncepcija pradėjo aprėpti ir platesnį sudėtingų gaminių kūrimo ir realizavimo problemų ratą, integruodama ankstyvoje gaminių projektavimo fazėje bendras dizainerių, technologų, gamybininkų, rinkotyros ir reklamos specialistų pastangas, taip sudarydama galimybes formuotis virtualiajai gamybai ir vienalaikei inžinerijai.

Siekiama plėtoti mechatronikos tyrimus ir veiksmingai juos naudoti Lietuvos pramonės konkurencingumui didinti - suvienijus verslo bei mokslo ir studijų institucijų pastangas sukurti aukštųjų technologijų gaminių, kuriuos pajėgtų gaminti šalies įmonės, aktyvinti bevielio valdymo tyrimus ir algoritmų bei technologijų kūrimo darbus, išplėtoti biomedicininės inžinerijos tyrimus ir sukurti aukštųjų technologijų gaminių žmogaus sveikatinimui, pasiekti „protingų“ medžiagų ir mikroelektromechaninių sistemų panaudojimo praktinę išeigą, padėti verslo sektoriui iš esmės patobulinti technologijas ir jų valdymą, padėti pamatus klasterių kūrimuisi šalies ūkyje, valstybei juos racionaliai remiant, gerinti mechatronikos specialistų rengimo ir tobulinimo sąlygas.

Beveik visose šalies pramonės šakose yra tarptautiniu mastu konkurencingų įmonių, gaminančių aukštosiomis technologijomis grįstus produktus, priskirtinus mechatronikos sričiai. Paminėtinos AB ir UAB „KTU-Festo PAC“, „Katra“, “Brown Sharpe Precizika”, „Vingis“, „Elsis“, “Sportinė aviacija”, “GTV“, „Ekranas“, „Vingriai“, „Medelkom“, „Tempera“. Dar daugiau yra įmonių, mechatronines technologijas naudojančių „tradiciniams“ produktams gaminti. Pavyzdžiui, mechatroninės sistemos kreipiamųjų sistemų gamybai, derinimui ir linijų valdymui „Vilniaus Vingio“ įmonėje, mechatroninės sistemos kineskopų gamybos ir derinimo procesuose Panevėžio „Ekrane“, technologinių procesų valdymo įranga bendrovėse „Lifosa“, „Achema“, „Snaigė“ ir kt. Tačiau su mechatronika susijusių pramonės šakų produktyvumas Lietuvoje yra menkas: vieno

darbuotojo sukuriama pridėtinė vertė yra 3 500 - 7 000 JAV dolerių, tuo tarpu pažangių šalių vidurkis yra apie 50 000 dolerių. Siekiant atlaikyti rinkos konkurencinį spaudimą, būtina didinti įmonių produktyvumą, kooperuotais ištekliais padėti atlikti joms reikalingus tyrimus ir parengti investicijų atėjimą.

Mechatroninių sistemų tyrimais ir kūrimu Lietuvoje užsiima KTU, VGTU, VDU, ŠU, LEI, PFI bei kitos mokslo ir studijų institucijos. Vykdoma daug šalies ir užsienio užsakomųjų darbų (tris ketvirtadalius visų užsakomųjų darbų šalyje atlieka KTU), vykdomi Europos Sąjungos ir kitų tarptautinių programų projektai. KTU, be modeliavimo, dinamikos, tikslumo, patikimumo ir kitų tyrimų gali pasiūlyti naujas technologijas ir gaminius.

Mechatronikos specialistai rengiami KTU ir VGTU. Šiuo metu KTU pagrindinėse studijose ir magistrantūroje pagal mechatronikos studijų programas rengiama arti 30 studentų, o pagal gimininga mechatronikai valdymo technologijų, automatikos ir valdymo, procesų ir sistemų valdymo programas – dar beveik 200 studentų; sąsajų su mechatronika turi informacinių technologijų, mechanikos, telekomunikacijų ir kitos studijų programos. Šios srities specialistus pradeda ruošti ir kolegijos, bei profesinio rengimo centrai. Projekto autoriui teko dalyvauti darbo grupės, rengiančios V lygio mechatronikos specialisto standartą, veikloje.

Ruošiant įvairaus lygio specialistus mechatroninių sistemų aptarnavimui susiduriama su problemomis. Pirmiausiai trūksta pačios laboratorijų įrangos. Programuojamų loginių valdiklių įvairovė gana didelė, kiekvienas jų dažniausiai turi savo firminį programinį paketą ir mokymo įstaigos nepajėgios visa tai įsigyti. Net ir turint valdiklį bei programą reikalingas vykdyklis. Automatinio valdymo sistemose ir robototechnikoje vis plačiau naudojama pneumatika. Darbai prie laboratorinių stendų su pneumatiniais ir elektropneumatiniais vykdykliais yra labai vaizdūs. Problema – brangūs pneumoelementai ir oro paruošimo sistema pneumatikai. Magistro darbe tiriamos galimybės realią fizinę įrangą pakeisti kompiuterinėmis modeliavimo programomis. Pagrindinė projekto įdėja yra ištirti galimybę sukurti programinę įrangą, kurios pagalba galima modeliuoti mechatronines sistemas. Programuojamo loginio valdiklio programinę įrangą sujungti su kuriama programa ir vietoje realaus vykdymo įtaiso panaudoti jo modelį. Duotai įdėjai realizuoti buvo paruoštas projektas:

- Atlikta esamos programinės įrangos analizė
- Išnagrinėti funkciniai ir nefunkciniai reikalavimai
- Išaiškinti pagrindiniai modelių kūrimo principai
- Sudaryta programinės įrangos struktūra
- Sukurta vartotojo sąsaja
- Atlikta klaidų ir defektų analizė
- Sudaryta programinės įrangos ir vartotojo dokumentacija

2. ANALITINĖ DALIS

2.1. Modeliavimo metodų analizė

Pagrindinis magistrinio darbo uždavinys sukurti mechatroninės sistemos modeliavimo programinės įrangos projektą ir ištirti galimybes programuojamų loginių valdiklių programiniais paketais valdyti kuriamus sistemų modelius. Didelės valdymo sistemos bei šiuolaikinės tchnologijos yra įgyvendinamos, naudojant hibridines valdymo sistemas, kuriose tolydiniai procesai yra tarpusavyje susieti diskretiniais priežastingumo ryšiais.

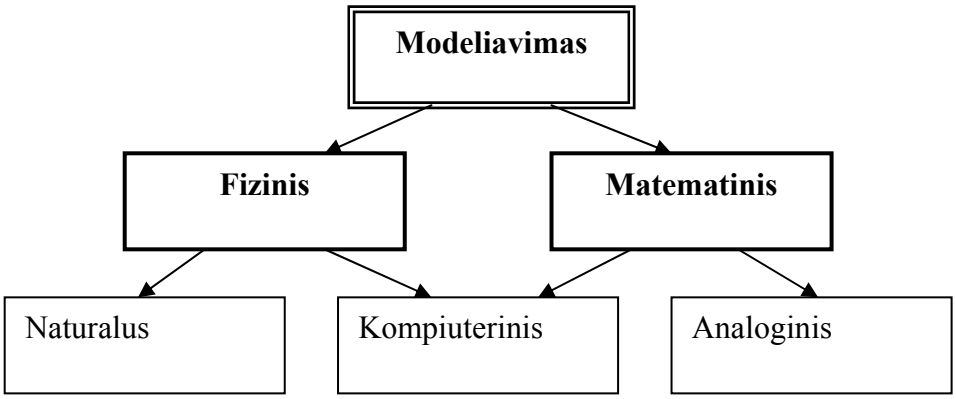
Tokių sistemų prigimtis ir realizavimo būdai gali būti labai įvairūs: nuo puslaidininkinių (tranzistorinių ar tiristorinių) energijos keitiklių iki lanksčios automatizuotos gamybos linijų ar sudėtingų agregatinių "žmogus - mašina" sistemų. Iš čia ir išplaukia naujų modeliavimo metodų, atspindinčių hibridinę realių sistemų struktūrą, poreikis (3)

Modeliavimas – kūrimas modelių, kad būtų galima išnagrinėti arba ištirti objektus, procesus, reiškinius. Tai tikrovės analizavimo būdas reikalingas objektyviems dėsningumams tirti. Modeliavimo objektas, kuris betarpiškai domina tyrinėtoją, vadinamas originalu. Jis yra pakeičiamas kitu objektu, kuris yra vadinamas modeliu. Tai analogas (pakaitalas) realiai egzistuojančių arba įsivaizduojamų objektų, procesų, reiškinių. Modelis naudojamas tada, kai apie originalą reikia gauti tokių duomenų, kurių arba sunku,arba iš viso neįmanoma gauti betarpiškai jį tiriant. Kad modelis atliktų savo funkcijas, jis turi būti panašus į originalą pagrindinėmis tyrinėtoją dominančiomis savybėmis ir bent viena savybe skirtis, ir jeigu modelis skiriasi būtent ta savybe, kuri trukdo betarpiškai tirti objektą, tyrinėtojas turi galimybę apeiti tą kliūtį. Tam pačiam objektui (procesui, reiškiniui) gali būti sukurta labai daug modelių. Pasirinkto modelio tipas priklauso nuo tikslo – kam kuriamas šis modelis, ir nuo to, kokiais metodais (būdais) renkama informacija apie modelio prototipą.

Modeliu gali būti ne tik realūs objektai, bet ir simboliniai modeliai (pvz. atomo modelis), lygtys aprašančios objekto reakciją, taipogi yra matematinis modelis.

Kurti panašias viena į kitą sistemas leidžia vienodi materialūs procesai, kurie vyksta skirtingos fizikinės struktūros sistemose. Tokį panašumą parodo matematinių išraiškų tapatybe.

Dabartiniu metu yra daug modeliavimo metodų. Plačiausiai paplitusius modeliavimo metodus galima klasifikuoti kaip parodyta 1 pav.. Yra skiriami du pagrindinai modeliavimo būdai: fizinis ir matematinis. Fizinio modeliavimo esmė tame, kad originalas yra tiriamas modelio pagalba, kuris yra panašus į originalą. Šis panašumas yra pagrįstas (originale ir modelyje vykstančių fizinių procesų tapatybe) tuo,kad originale ir modelyje vykstantys procesai turi tą patį fizinių procesų pasireiškimą, panašumas yra išlaikomas geometriniuose masteliuose, laike ir panašiai.



1 pav . Modeliavimo kryptys

Esminis fizinio modeliavimo privalumas yra tas, kad tiriamas reiškinys pasireiškia visapusiškai ir tokiu būdu susidaro galimybė pakankamu tikslumu gauti žinias apie mus dominantį. originalą. Pagrindiniu šio metodo trūkumu yra modelio sukūrimo problema ir su ja surišti sunkumai. Fizinis modeliavimas dažniausiai naudojamas hidrodinamikoje, aerodinamikoje, elektros mašinų pramonėje ir panašiose srityse, tiriant sudėtingus reiškinius, kuriuos tirti kokiu nors kitu, paprastesniu metodu, nėra galimybės. Tai galima pasakyti apie procesus neturinčius pakankamai tikslaus, o tuo pačiu ir paprasto matematinio aprašymo. Galima pateikti pavyzdį iš elektros mašinų pramonės. Čia susiduriame su dviem skirtingais atvejais: labai didelės elektros mašinos ir labai mažos elektros mašinos. Labai mažos elektros mašinos (kvarcinio laikrodžio elektros variklis), modeliuojamos didesne mašina, kuriai tirti galime panaudoti jutiklius, kurių prie originalo paprasčiausiai negalime prijungti. Labai didelės mašinos (hidroelektrinių turbogeneratoriai) negali būti tiriamos, dėl energetinių sąnaudų bei kainos. Toks generatorius yra gaminamas didelio žmonių kolektyvo ir kelis mėnesius. Tuo tarpu tirti objektą ekstremaliuose rėžimuose yra per brangu ar iš viso neįmanoma. Tuomet gaminamas mažesnis generatorius, net iki 100 kartų sumažintas, ir su juo atliekami visi norimi bandymai, nebijant , kad jis gali būti ir sugadintas. Visais minėtai atvejais yra laikomasi tam tikrų modelio sudarymo taisyklių, kad modeliai būtų adekvatūs originalui visomis savo savybėmis, ir skiriasi pagrindine modeliavimui trukdančia savybe.

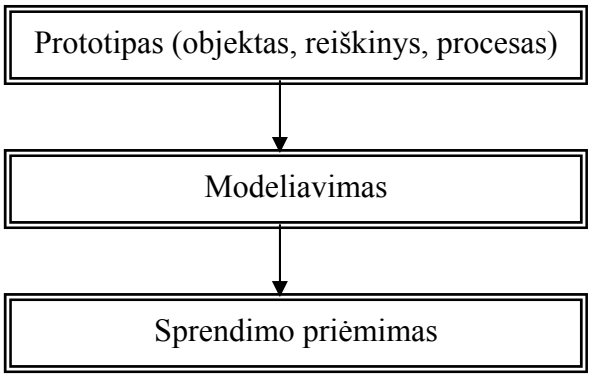
Tais atvejais, kai reiškiniai yra pakankamai tiksliai aprašyti matematiškai, jų tyrimui tikslinga taikyti matematinį modeliavimą (3).

Modeliavimas iš principo yra pasyvus tyrimo metodas, sudarantis galimybę pasižiūrėti į tiriamos sistemos funkcionavimą, esant vienokiems ar kitokiems techniniams, scheminiams ir pan., sprendimams. Tačiau visuomet kelia ir kels susidomėjimą galimybė nagrinėjamą sistemą suderinti maksimaliai pageidaujamam variantui. Kitaip sakant, kad tiriamąją sistemą galėtume optimizuoti. Optimizavimas gali būti dvejopas:

Pirmas būdas, kuomet keičiama tiriamos sistemos struktūra, t.y. sistemoje pakinta elementų bei detalių kiekis ir sujungimo būdas (reguliatorių tipas, komutavimas, elementai, kai pakinta matematinės išraiškos, o ne vien tik koeficientų reikšmės), kitaip tariant sistema savo fizine sudėtimi tampa kitokia. Toks optimizavimas dažnai vadinamas struktūriniu optimizavimu.

Antras būdas, kitaip dar vadinamas parametrine optimizacija, kai keičiami esamų elementų parametrai, kitaip tariant, tik matematinių išraiškų koeficientai, ir tokiu būdu pati sistema fiziškai sudėtimi nepakinta, o pakinta tik jos funkcionavimo savybės.

Modeliavimas tampa ne tik pasižiūrėjimo ar analizės įrankiu, o galingu instrumentu, leidžiančiu iš anksto suderinti įrenginio ar objekto parametrus, sutrumpinant derinimo laiką realiame objekte, tuo pačiu sutaupyti darbo bei piniginius resursus. Modeliuojant šiuolaikiniais modeliavimo įrankiais (programiniais paketais) modeliavimas pagal struktūrinę schemą yra pats aiškiausias ir informatyviausias. Labai dažnai struktūrinė schema arba prototipas nuo modelio schemos savo išvaizda ir išdėstymu nesiskiria, turi beveik tokią pačią konfigūraciją. Sudarant modelio schemą, joje turi būti įvertinti visi signalai ir kintamieji .



2 pav. Modeliavimo etapai

Modeliavimo pradžia – objekto, proceso, reiškinio prototipas. Modeliavimo pabaiga – sprendimo priėmimas: arba sukuriamas naujas objektas, kurio modelį mes nagrinėjome, arba pageriname jau egzistuojantį objektą, arba gauname apie šį objektą papildomų žinių.

2.2. Modeliavimo programų analizė

Dabartiniu metu egzistuoja gana daug kompiuterinės programinės įrangos, kurios paskirtis modeliuoti fizinius , cheminius, socialinius, gamybinius ir kitų tipų procesus bei reiškinius. Toliau yra atliekama programų, kurios gali būti naudojamos automatinio valdymo sistemų parametrams ir būsenoms modeliuoti analizė ir palyginimas. Pasirinktos tik kelios iš daugelio programų, plačiausiai naudojamos kolegijų ir universitetų studentų mokymui. Šių programų pasirinkimas bei taikymo

sritys yra subjiektyvios ir priklauso nuo dėstytojo kvalifikacijos. Ne visos jos yra lygiavertės, tačiau yra mokymo sričių kuriuose vietoje programos su didele funkcijų gausa galima panaudoti gana paprastas modeliavimo sistemas. Tuo labiau, kad galingos programinės įrangos funkcijos mažai kada pilnai panaudojamos. Kita vertus, specialistas mokantis naudotis standartine įranga, pvz MPLAB įranga visada sugebės pritaikyti savo tikslams ir nestandartines programas.

MATLAB

Techninių skaičiavimų aplinka, skirta greitos veikos sistemų matematiniam apdorojimui ir vizualizacijai. MATLAB apima skaitmeninę analizę, veiksmus su matricomis ir masyvais, signalų apdorojimą ir grafiką. Programa turi ir operacinės sistemos funkcijų: galima operuoti bylomis ir katalogais (6). Pagrindinės MATLAB taikymo sritys yra:

- Matematika ir skaičiavimai.
- Algoritmų sudarymas.
- Imitacinis modeliavimas ir maketavimas.
- Duomenų analizė ir procesų vizualizacija.
- Mokslinė ir inžinerinė grafika.
- Taikomųjų programų kūrimas, įskaitant ir vartotojo sąsają.

Viena iš MATLAB paketo sudedamųjų dalių yra SIMULINK programavimo kalba. Jos pagalba galima kurti ir modeliuoti sistemų struktūras. Tam tikslui naudojama struktūrinių blokų biblioteka.
Privalumai:

- plačiai taikoma mokymo įstaigose;
- daug pavyzdžių ir matlab bylų literatūroje bei internete;
- platus funkcijų ir veiksmų asortimentas;
- galimybė modeliuoti ir derinti inžinerines sistemas;

Trūkumai:

- programa komercinė, reikia pirkti licenziją;
- naudojama speciali programavimo kalba;
- nėra grafinių modelių bibliotekos;

CENTAURUS

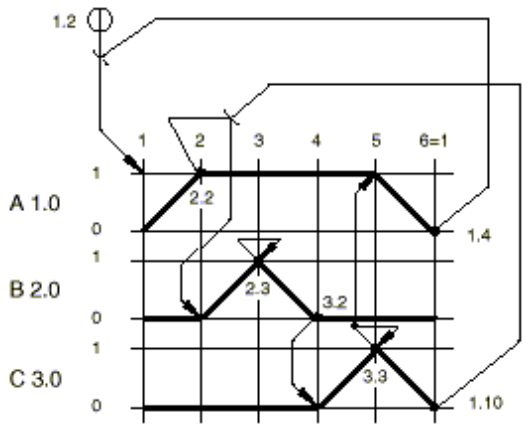
Kauno technologijos universiteto specialistų sukurta programinė įranga, skirta heterogeninių (mišrios struktūros) ir netiesinių valdymo sistemų analizei bei sintezei. Tai specializuota programinė įranga leidžianti modeliuoti sistemas turinčias algoritminę (loginę) dalį aprašomą Petri tinklo elementais, ir analoginę dalį, vaizduojamą struktūrine schema. Analoginė dalis taip vadinama tik sąlyginai, kadangi joje yra galimi ne tik analoginiai elementai, apjungiantys netiesines diferencialines lygtis, bet ir netiesiniai, diskretiniai ar specializuoti elementai. Tokiu būdu

heterogeninės sistemos modelyje yra abi sistemos: ir Petri tinklas ir analoginė dalis. Modeliuoti galima trijų rūšių modelius: tik Petri tinklą, tik analoginį modelį ar abu kartu.

Modelis įvedamas atitinkamais elementais, o analoginė dalis funkciniais blokais. Funkciniai blokai naudojami valdymo sistemos analoginio modelio kūrimui. Blokų visuma leidžia modeliuoti: tiesines, netiesines, diskretines ir tolydines sistemas, statinius ir dinaminius elementus, aritmetines ir logines operacijas bei specialius elementus. Visi šie paminėti elementai ir blokai sudaro atitinkamas bibliotekas. Petri tinklo elementai naudojami kurti loginius modelius. Yra trys tradiciniai Petri tinklo elementai – pozicija, perdava ir ryšiai, bei elementas vadinamas "procesu". "Procesas" vaizduoja pereinamuosius procesus vykstančius valdymo sistemos analoginėje dalyje ir turi įtaką algoritminei modelio daliai, ko nėra tradiciniuose Petri tinklų modeliuose. Iš Petri tinklų įvairovės bene plačiausiai valdymo sistemoms modeliuoti naudojami laikiniai spalvotieji ir klasikiniai tinklai. Tokia modelio struktūra įgalina modeliuoti dideles gamybines sistemas, kurių elementai tarpusavyje surišti telekomunikacijos elementais.

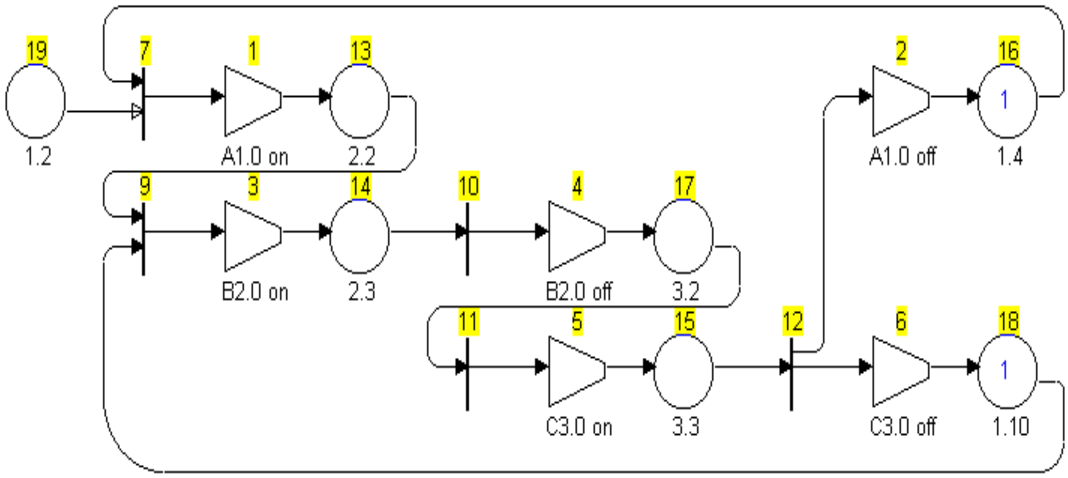
Valdymo sistemos modeliavimas dažnai turi aiškų tikslą – parašyti skaitmeninio valdiklio programą. Siekdamas unifikuoti įvairių valdiklių programavimą tarptautinis elektrotechnikos komitetas (IEC) priėmė standartus IEC 60848 (pirmoji redakcija 1988 m. ir antroji - 1999 metais) ir IEC 61131-3 (1992 m.), kuriais apibūdinamas valdiklių programavimas pagrįstas GRAFCET kalba. Šios kalbos grafinis pavidalas ir yra Petri tinklas, kurio žetonai yra loginio tipo. GRAFCET tinkle yra daug elementų, kurie būtini valdiklio programai parašyti, tačiau visai nereikalingi, kai analizuojamas algoritmo veikimas. Be to, ryšiai su valdymo objektais neturi grafinės išraiškos. Todėl modeliavimas taikant aukščiau minėtas Petri tinklo atmainas yra vaizdesnis, lengviau suvokiamas..

Petri tinklu ypač patogu modeliuoti programuojamame valdiklyje funkcionuojančios valdymo programos algoritmą, jį ne tik modeliuoti bet ir derinti, ir ypač ryšyje su funkcionuojančiu valdymo objektu. Toliau pateikiamas tokio modelio sudarymo pavyzdys CENTAURUS programiniu paketu.



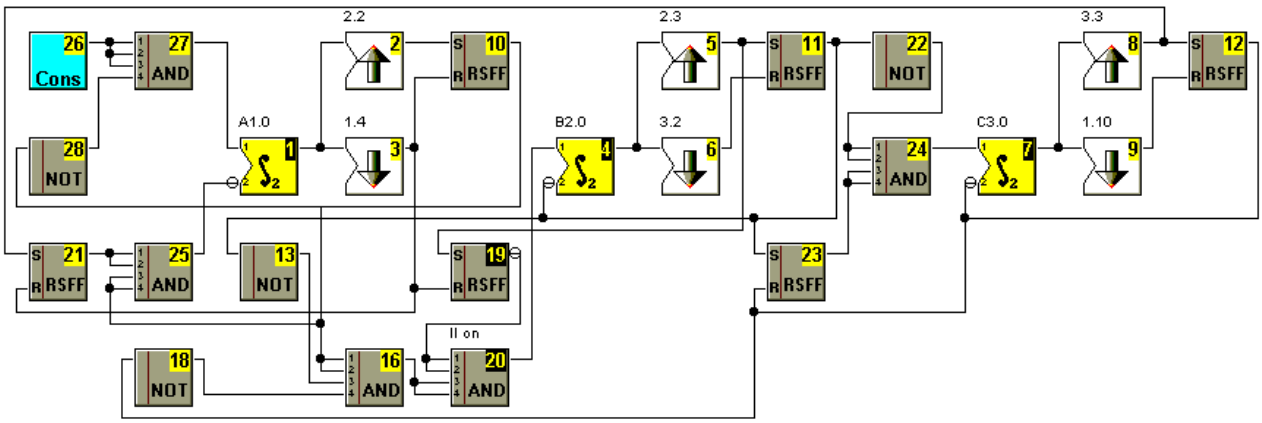
3 pav. Modeliuojamos pneumatinės sistemos judesio diagrama

Diagramoje pateikti trijų pneumatinių cilindrų (A 1.0; B 2.0; C3.0) būsenos tam tikrais laiko momentais. Nulinė būsena vaizduoja neišstumtą stumoklį, o vienetinė – pilnai išstumtą. Apie būsenų pokyčius sistemą informuoja jutikliai. Pneumatinė sistema paleidžiama “Start” mygtuku ir cilindrų stumokliams atlikus visus judesius, vėl grįžtama į pradinę būseną.



4 pav. Pneumatinio įtaiso Petri tinklo modelis

Petri tinklo modelyje trapecijų blokais nurodyti cilindrų stumoklių išstūmimo ir įtraukimo signalai, o apskritimais – jų būsenos po šių signalų. Naudojant Petri tinklo modelį gana paprasta sudaryti sistemos valdymo funkciinę schemą.



5 pav. Pneumatinio įtaiso modelio funkcinė schema.

Ypatinga paketo dialoginės sistemos savybė, kad duomenų, reikalingų skaičiavimams, įvedimas yra labai paprastas. Vartotojas turi tiktai pasirinkti vieną iš leidžiamų kelių ar teisingai atsakinėti į pateikiamus klausimus. Kur būtina, programų paketas pats nustato atsakymų seką.

- Programos privalumai:
  - Nedidelė apimtis (3 Mb).
  - Lietuviškos kalbos aplinka.

- Paprasta vartotojo aplinka.
- Duomenys įvedami dialogo režime.
- Galimybė naudoti įvairius modeliavimo metodus.

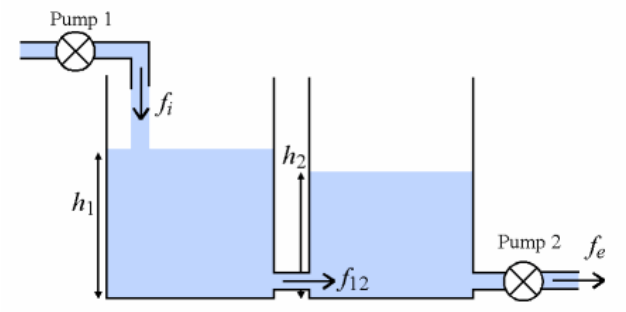
Trūkumai:

- Nėra duomenų bazės.
- Nėra simuliavimo režimo.

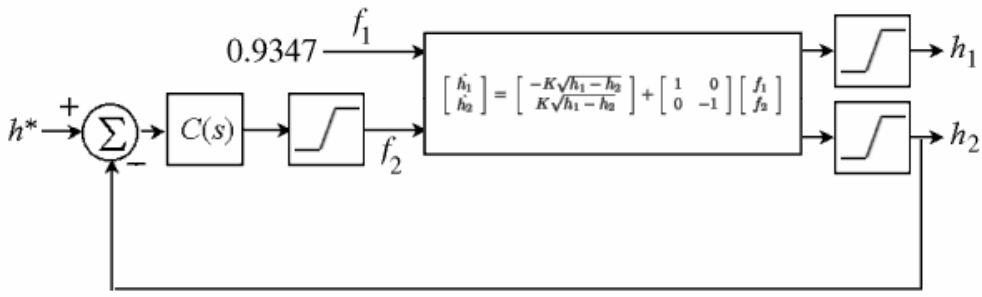
Control system design

http://cds.newcastle.edu.au/control/index.html

Australijos Niukastlo universiteto interaktyvi internetinė svetainė (25). Joje yra pateikiama teorinė procesų kontrolės ir valdymo medžiaga, momomosios skaidrės ir uždaviniai. Virtualioje laboratorijoje yra aštuonių uždavinių modeliai, kuriuos galima stebėti simuliaciniame režime. Galima paminėti tokius modelius kaip plieno valcavimo staklės, pH kontrolės stendas ir kiti. Modeliai dirba Java programiniame pakete. Didelis programos privalumas – galimybė atsisiųsti MPLAB programos bylos ir procesus tirti MPLAB SIMULINK pakete. Toliau yra pateikiamas vienas iš šioje internetinėje svetainėje esančių modelių.



6 pav. Talpų užpildymo lygių kontrolės modelis



7 pav. Modelio valdymo sistema

Privalumai:

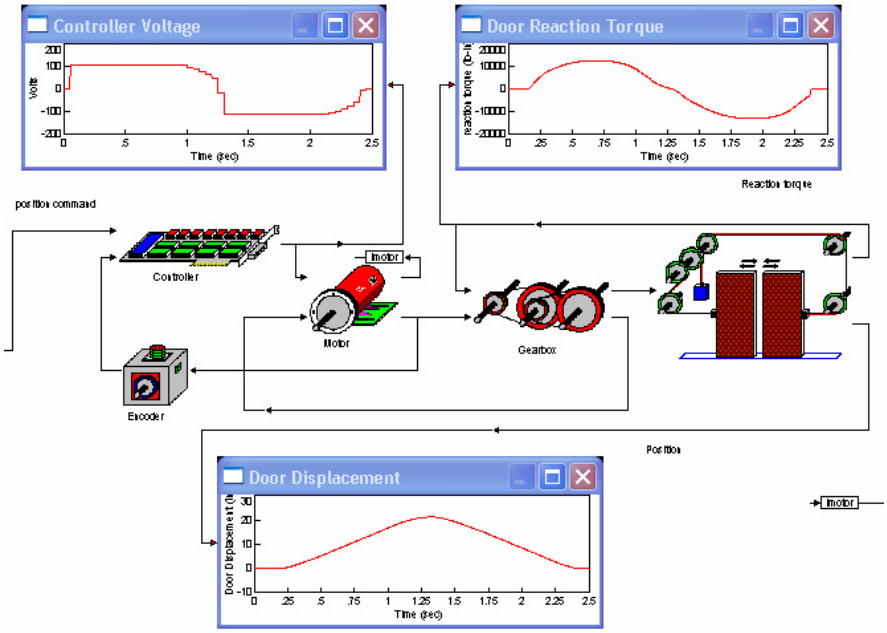
- galimybė tirti valdymo procesų parametrus;
- didelė MPLAB bylų duomenų bazė;
- pateikta teorinė mokymo medžiaga , skaidrės, matematinės procesų lygtys;

Trūkumai:

- modeliai veikia tik esant internetiniam ryšiui;
- nėra galimybės kurti savus modelius ;

VisSim

Visual Solutions, Inc. firmos gaminys (17). VisSim Wiewer programinė įranga turi didelę elementų biblioteką, funkcijas, loginę aritmetiką . Sukompiliuoja MPLAB funkcijas . Animuoja valdymo elementus. Leidžia simuliuoti funkcines ir blokines schemas, keisti jų parametrus.



8 pav. Durų uždarymo mechanizmo modelis ir pereinamosios charakteristikos

GLG Toolkit:

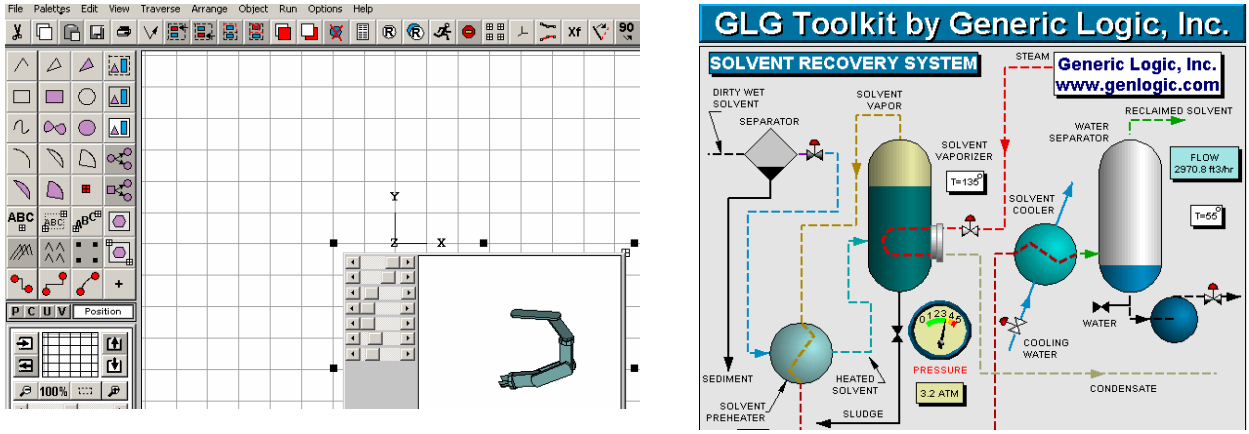
www.genlogic.com

Generic Logic firmos Glg Toolkit programinė įranga skirta kurti ir animuoti informacines sistemas, vizualizavimo programas , stebėti standartinius gamybinius procesus ir kurti naujus, adekvačius realioms sistemoms (24).

Programinis paketas taip pat skirtas dinaminiams procesams modeliuoti. Palaiko C/++, Java kalbas ir Active X komponentus Windows ir Unix operacinėse sistemose. Yra išleistos trys GLG Toolkit versijos: Basic, Professional ir Enterprise. Versijos skiriasi įrankių ir grafikos bibliotekomis. Jos leidžia formuoti 2D ir 3D grafikus procesų charakteristikų tyrimui, procesų kontrolės simbolius, matavimo prietaisus, aviacinės technikos prietaisus ir indikatorius bei kitus grafinius vaizdus. Taip pat galima susikurti ir savo grafinius objektus. Naudojamos **(EAPI)** taikomojo programavimo technologijos , leidžiančios dinamiškai keisti modelio grafiką iš kitų programinių paketų.

Grafinis Glg Tooltip redaktorius yra labai patogus vartotojui bei savo galimybėmis nenusileidžia kitiems specializuotiems visualizacijos paketams . Sukurtą grafinį vaizdą galima

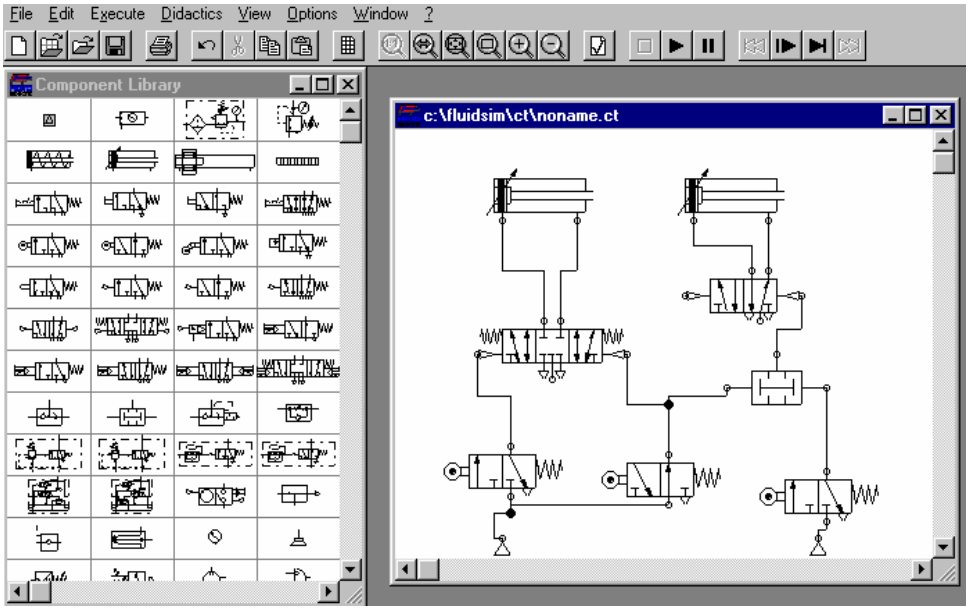
keisti iš 2D dimensijos į 3D, keisti mąstelį, peržiūros kampą pagal tris koordinačių ašis . Darbo prieduose pateikiamas modelio programos kodo pavyzdys Java kalboje.



9 pav. Grafinis redaktorius Glg Builder

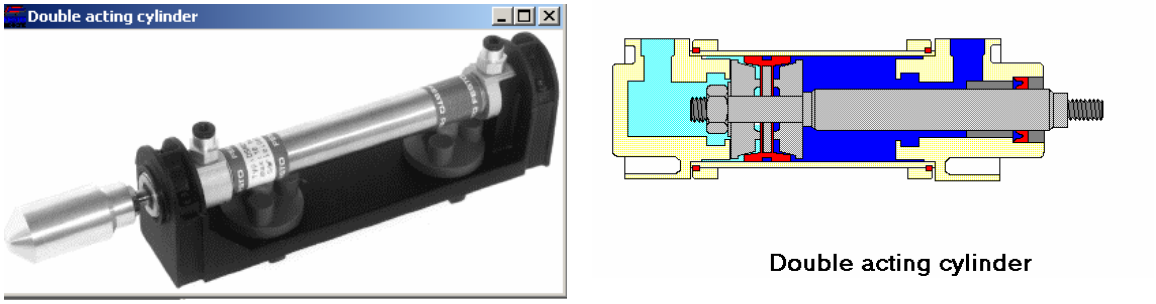
FluidSim

Mokomųjų darbų (WorkBench) programinės įrangos grupei priskiriama programa – pneumatikos ir elektropneumatikos įrenginių darbo modeliavimui. Jame yra elementai kurie dažniausiai naudojami mechatroninėse sistemose: jutikliai, vykdykliai, judikliai ir t.t. Tai pneumatiniai cilindrai, skirstytuvai mygtukai, elektromagnetinės relės. Schema renkama iš programos bibliotekoje esamų elementų. Yra galimybė pilnai modeliuoti sistemos veikimą, siekiant atlikti tam tikrą pneumatinių cilindrų judesių seką pagal duotas judesio lygtis ar laikines diagramas.



10 pav. Pneumatinės sistemos modelio pavyzdys

Dar vienas programos privalumas – galima gauti išsamią informaciją apie kiekvieną naudojamą elementą: pradedant nuotrauka, baigiant veikimo principu.



11 pav. Pneumatinio komponento nuotrauka ir pjūvis

Privalumai:

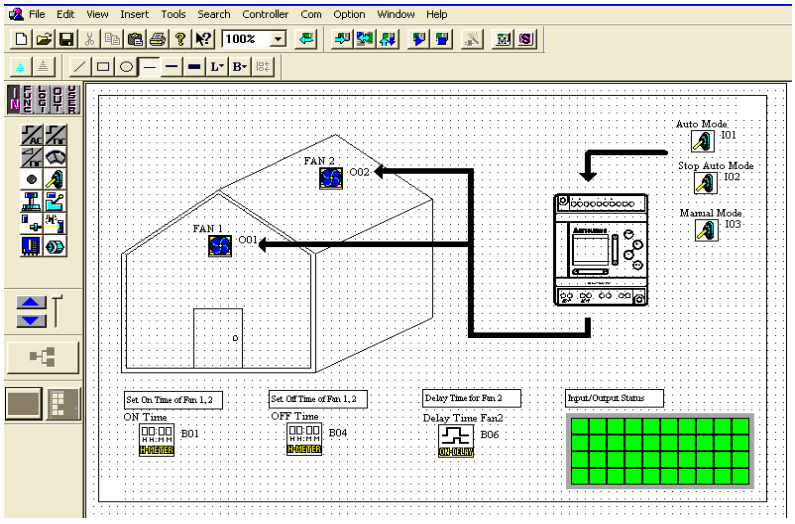
- programa nemokama;
- gausi elementų ir schemų pavyzdžių biblioteka;
- pateikiamas elemento vaizdas ir veikimo principas;
- modeliuojamas sujungtos schemos veikimas;

Trūkumai:

- nėra elektroninių komponentų;
- negalima modeliuoti pereinamųjų procesų;

AL-PCS/WIN

Tai Mitsubihi firmos Alpha valdiklių programavimo - vykdymo įtaisų modeliavimo įranga. Atliekant valdiklių programavimą galima vaizdžiai pademonstruoti programos veikimą grafinėje terpėje. Programos grafinis eskizas tai paprastas piešimo langas. Šis eskizas yra naudinga grafinio programos veikimo vaizdavimo priemonė. Jį leidžia demonstruoti įrenginius, prijungtus prie valdiklio įėjimų – išėjimų grafinėje vartotojo sumodeliuotoje terpėje (23).



12 pav. Programos eskizo langas

Privalumai:

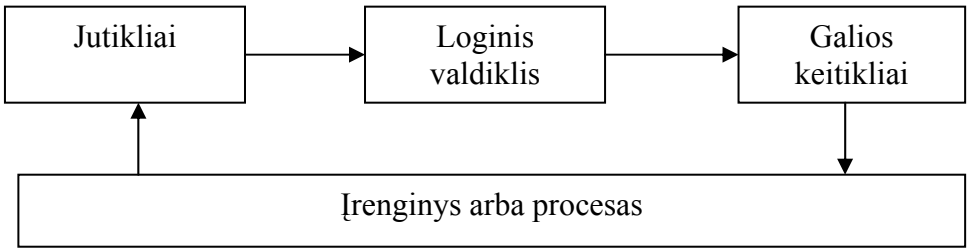
- leidžia braižyti diagramas naudojantis piešimo įrankių juosta;
- galima imituoti įrenginių prijungimą prie valdiklio;
- yra galimybė programuoti valdiklius ir bandyti simuliacijos režimu;
- leidžia imituoti ir tikrinti loginę programą , be prijungto valdiklio;

Trūkumai:

- modeliuojama grafika yra statiška;
- palaikoma tik viena valdiklių programavimo kalba;

Atlikus procesų ir sistemų modeliavimo programų analizę galima teigti, kad įrangos kūrėjai naudoja gana įvairius metodus savo tikslams pasiekti. Kai kuriose programose yra labai stipri matematinė bazė. Kitose daugiau yra remiamasi grafinėmis bibliotekomis ir funkciniais blokais. Daugumoje iš jų modeliuojamas procesas yra derinamas keičiant koeficientus standartiniuose duomenų įvedimo languose. Gaunamas rezultatas stebimas pateiktose diagramose arba grafikuose. Tai leidžia gana greitai gauti norimą rezultatą nesigilinant į proceso matematiką.

Magistriniame darbe keliami modeliavimo uždaviniai yra skirti imituoti loginių valdymo sistemų įtaisus . Jų valdymo algoritmus galima aprašyti loginės arba Bulio algebros dėsniais. Loginio valdymo įtaisą sudaro diskretiniai jutikliai, programuojami loginiai valdikliai (PLV), galios keitikliai ir valdomas įrenginys



13 pav. Loginio valdymo įtaiso struktūra

Pasirenkamas programuojamas loginis valdiklis ir pagal duotą judesio lygtį arba diagramą sudaroma procesą valdanti programa. Ši programa užkraunama į realiai egzistuojantį PLV. Visi kiti sistemos elementai , jutikliai, galios keitikliai ir procesas imituojami programiškai. Darbe tiriama galimybė komunikuoti vartotojo programai su valdikliu ir keisti jo būsenas. Jutikliams, galios keitikliams ir pačiam procesui modeliuoti yra pasirenkamas FluidSim programinės įrangos modelių variantas. Jis leidžia akivaizdžiai stebėti programos darbą ir įsitikinti jos teisingumu.

3. PROJEKTINĖ DALIS

3.1 Programinės įrangos projektavimas

Kuriant programas susijusias su PLV patartina naudoti modulinio projektavimo metodą (1). Ši metodika leidžia sudėtingą uždavinį skaidyti į smulkesnius. Jei programą projektuoja projektuotojų grupė, kiekvienam galima pateikti vieną ar kelis sprendžiamo uždavinio modulius. Kiekvienas modulis šiuo atveju turėtų būti užbaigta programa. Norint užtikrinti modulių užbaigtumą, kiekviename jų nustatoma kurie duomenys patenka ir kurie grįžta programos pabaigoje. Paprastai viršutinio lygio moduliai atlieka bendresnes funkcijas ir naudoja žemesnių lygių modulius. Modulinio projektavimo privalumai – užtikrinamas programos suprantamumas, efektyvumas, patikimumas, lankstumas. Šis metodas dažnai naudojamas sudarant programuojamų loginių valdiklių programas.

Programinės įrangos išbaigtumas gali būti nustatomas pagal galimybių išbaigtumo modelį (CMM -Capability Maturity Model) (20) kuriame išskiriami penki išbaigtumo lygiai:

1. Pradinis lygmuo. Programavimo procesas charakterizuojamas kaip chaotiškas. Nustatoma ir vykdoma keletas procesų, kurių baigtis priklauso nuo individualių programuotojo pastangų. Tai nepatyrusio programuotojo lygmuo kuris mano, kad programos patikimumą galima užtikrinti bandymais. Šiame lygmenyje nėra programos kūrimo proceso valdymo.
2. Atkartojimo lygmuo. Programavimo procesas dokumentuojamas, sudaromas darbų grafikas, parenkami programavimo metodai ir technika. Lygmenyje vykdomas reikalavimų valdymas, programos kokybės analizė, planuojamas ir nuolat stebimas projektavimo procesas.
3. Nustatymo lygmuo. Vykdomam projektui nustatomas tinkamiausias procesas ir atatinkama programinė įranga. Vyksta darbo grupės koordinacija, apmokymas.
4. Valdymo lygmuo. Įvertinamas programavimo procesas ir gauto produkto kokybė. Abi sudedamosios dalys kiekybiškai įverinamos. Vyksta kokybinis ir kiekybinis projekto valdymas.
5. Optimizavimo lygmuo. Naudojant kokybinį grįžtamą ryšį atliekamas proceso ir technologijų modernizavimas, klaidų ir defektų prevencija.

Valdant programinės įrangos projektavimo procesą, patartina naudotis ir programų metrika (software metrics) arba programinės įrangos kokybės sistema (SQS) (20). Ji sprendžia kokybės užtikrinimo kūrimo procese ir kokybės palaikymo eksploatacijos metu uždavinius.

a) standartizavimas

b) recenzavimas
c) defektų analizė
d) konfigūracijos valdymas
e) apsaugos priemonės
f) dokumentavimas
g) apmokymas
h) pardavimo valdymas

Sudarant , mechatroninės sistemos, valdomos PLV, programos kūrimo proceso algoritmą, skiriami keli etapai. Pirmame, formavimo etape nustatomi funkciniai ir nefunkciniai reikalavimai, apibrėžiamos charakteristikos. Toliau yra sudaromos konkrečios programos specifikacijos. Jos gali būti vidinės ir išorinės. Į išorines specifikacijas įeina programos vardas, įėjimo taškų vardai, programos funkcijos ir t.t. Vidinėse nurodomi valdymo signalai, darbo laiko diagramos, signalų trukmės, programiniai ryšiai. Vadovaujantis išorinėmis specifikacijomis sukuriama tarpusavyje susijusių modulių visuma, o pats modulių darbas aprašomas vidinėmis specifikacijomis. Projektavimo etape programa išskaidoma į modulius ir detalizuojami modulių algoritmai. Ketvirtas etapas yra programavimas arba kodavimas. Šiame etape suprojektuota programa užrašoma viena iš pasirinktų arba viena iš galimų programavimo kalbų. Nors tai vienas iš svarbiausių programinės įrangos kūrimo etapų, bet kokybišką ir patikimą programą galima sukurti tik kokybiškai ir korektiškai ją suprojektavus. Programavimo etape galima pasiūlyti laikytis šių rekomendacijų:

- reikėtų stengtis kurti paprastą o ne sudėtingą programą
- nesustoti prie pirmo programos varianto
- nevengti perrašyti nevykusios programos vietos
- įsitikinti ar duomenys ir signalai atitinka priimtus apribojimus
- užtikrinti kad ir neteisingai įvesti duomenys bus atpažinti
- kuo rečiau programoje naudoti perėjimo operatorius
- kuo dažniau naudoti komentarus
- pirma gauti teisingą programą, o tik po to ją optimizuoti

3.2 **Reikalavimų specifikavimas**

3.2.1. Reikalavimų analizė

Programinės įrangos kūrimo proceso metu numatomi veiksmai, kurių tikslas yra kurti ir prižiūrėti programinę įrangą. Šie veiksmai gali būti suskirtyti laiko atžvilgiu į keturias dalis:

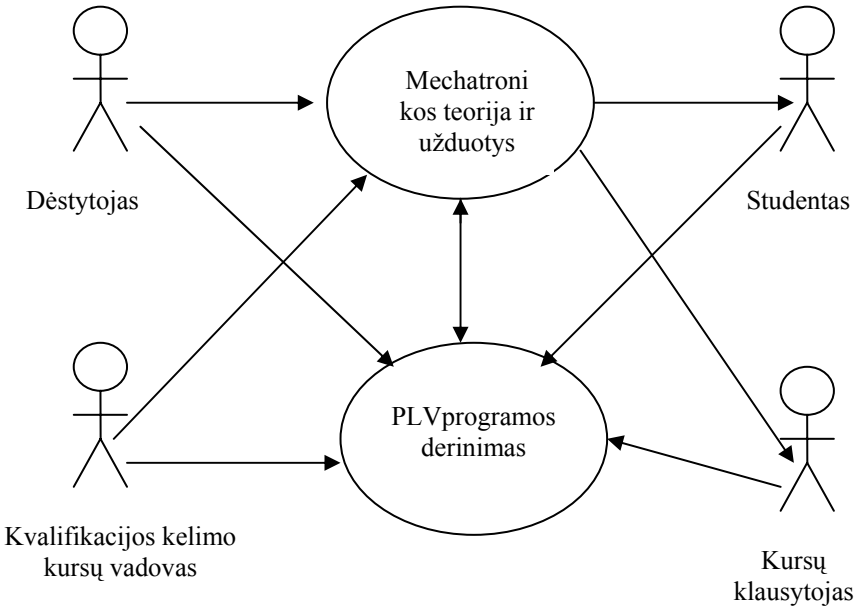
- reikalavimų specifikacijos sudarymas;
- programinės įrangos sistemos projektavimas, gamyba;
- tikrinimas, ar programinė įranga atitinka vartotojo poreikius;
- programinės įrangos eksploatavimas ir keitimas priklausomai nuo poreikių pasikeitimo;

Kokybiška programinė įranga turėtų teikti vartotojui reikiamą funkcionalumą ir greitaeigiškumą, turi būti tinkama eksploatavimui, kelti pasitikėjimą ir būti efektyvi bei tinkama naudojimui konkrečiam vartotojui. Programinė įranga turi vystytis, kad atitiktų besikeičiančius poreikius. Ji taip pat turi būti parašyta ir dokumentuota taip, kad galėtų būti keičiama be didelių išlaidų. Programinė įranga turėtų tikti operacinei sistemai ir neturėtų bereikalingai naudoti jos resursų.

Kuriant projektą būtina nustatyti funkcinius ir nefunkcinius reikalavimus būsimai programinei įrangai. Reikalavimai turi skelbti ką sistema turi daryti, o projektas turi apibūdinti, kaip tai padaryti. Tiriant galimybes, analizuojant reikalavimus, nustatoma ar tikslinga kurti sistemą. Teisingai nustatyti reikalavimus gali tik programuotojas gerai žinantis tiriamą sritį. Teisingumą galima patikrinti kuriant programos prototipus. Projekte numatoma kurti prototipus reikalavimų išgavimui ir vartotojo sąsajos funkcijų nustatymui.

Pateikiamas magistrinis darbas yra skirtas naudoti Panevėžio kolegijoje ir viešojoje įstaigoje „ Panevėžio mechatronikos centras“. Šių institucijų pageidavimu ir yra kuriamas projektas „ Mechatroninių sistemų modelių sudarymas ir tyrimas“. Kuriama programa bus naudojama sudentų mokymui, pramonės įmonių darbuotojų, besimokančių kvalifikacijos kėlimo kursuose, mokymuisi. Projektas gali būti naudojamas įvairių mechatronikos sudedamųjų dalių teorijos studijavimui, programavimo pagrindų įsisavinimui, taip pat praktiniams ir tiriamiesiems darbams atlikti. Projektavimo darbų pradžia – 2002 m. spalio 10 d., pabaiga – 2003 m. gruodžio 20 d. Trukmė – 14 mėnesių. Kuriant projektą naudojama tik legali programinė įranga ir kai kurios projekto dalys be paildomai įdiegtų programų dirbti negalės. Kaip matoma iš atliktos analizės, tu analogiškų programų nėra daug , ypač lietuvių kalba. Yra atskiros programos ir aprašymai *.pdf formatu, kurių panaudojimo galimybės yra ribotos. Kai kurios valdiklių programos turi simuliavimo ir derinimo paketus, kurių pagalba galima patikrinti sukurtus darbo algoritmus. Projektuojama įranga skirta tik mokymo tikslams, todėl pelno gauti nenumatoma. Taip pat nėra skaičiuojamos ir būsimos projekto kūrimo išlaidos.

Kuriamos programinės įrangos vartotojo reikalavimams pagrįsti sudaroma panaudojimo atvejų diagrama:



14 pav. Modeliavimo sistemos USE CASE diagrama

Kaip matoma iš diagramos projekte sprendžiami du pagrindiniai uždaviniai: teorinės medžiagos ir praktinių užduočių pateikimo bei programuojamų loginių valdiklių programų sudarymo ir derinimo.

Sistemai keliamų uždavinių analizė

Lentelė 1

Uždavinys	Aprašas	Pastabos
Mechatronikos teorija ir užduotys	Projekte numatomas atskiras teorijos modulis. Dėstytojas nurodo kokią teorinę disciplinos dalį reikėtų pasiskaityti , prieš atliekant praktinius darbus. Teorinėje dalyje pateikiamos žinios, susijusios su naudojamais programiniais paketais bei atskiromis mechatronikos disciplinomis. Pageidautina, kad studentai turėtų galimybę studijuoti naudodamiesi internetiniu tinklu. Teorinės žinios tikrinamos tęstų pagalba. Praktinės užduotys gali būti pateiktos su sprendimo pavyzdžiais, arba be jų. Galimybė dėstytojui keisti ir atnaujinti užduotis.	Galima naudoti teksto ir internetinių puslapių kūrimo redaktorius, pagalbos bylas.

PLV programų derinimas	Susipažinus su teorine dalimi toliau atliekamas praktinis darbas. Jame gali būti numatytas įrangos pasirinkimas, PLV programavimo kalbos tipai. Sudarius programą pagal pateiktą judesio diagramą, sistemoje turi būti galimybė ją išbandyti nenaudojant realios mechaninės, pneumatinės ar elektrinės įrangos.	PLV programų simuliatoriai galėtų dirbti savarankiškai arba lygiagrečiai su realia įranga.

Sprendžiant juos laikomasi modulinio projektavimo principų. Pasirenkami tokie programinės įrangos moduliai:

- teorijos
- praktinių užduočių
- distancinio mokymo (internetinis)
- pneumatinių sistemų modeliavimo
- PLV programų derinimo

3.2.2. **Funkciniai reikalavimai**

Altlikus vartotojo reikalavimų sistemai analizę ir nustačius uždavinius, kuriuos projektuojama programinė įranga turi spręsti, keliami funkciniai reikalavimai. Sistema turi užtikrinti šias paslaugas:

- pateikti mechatroninių sistemų sudedamųjų dalių teorinę medžiagą
- pateikti praktinių darbų užduotis ir pavyzdžius jiems atlikti
- tikrinti vartotojo žinių lygį testų pagalba
- programuoti valdiklius standartinėmis programavimo kalbomis
- tikrinti sukurtas programas panaudojant programinius mechatroninių sistemų modelius
- modeliuoti sistemų darbą programiniu būdu.

Programos darbo metu turi būti numatyta galimybė atlikti reikiamos informacijos paiešką naudojantis personalinio kompiuterio katalogais ir interneto nuorodomis. Sukurti programų modeliai turėtų būti spausdinami tiesiogiai arba kopijuojant ir keliant juos į tekstinius redaktorius. Pageidautina, kad analizės rezultatus būtu galima palyginti naudojant ne tik tekstinę , bet ir grafinę formas. Valdymo sistemų reguliatorių modeliai gali būti pateikiami matematinėmis išraiškomis. .

Programos vartotojas sistemos modeliavimo tikslams įveda duomenis iš pateikto sąrašo arba iš grafinių vaizdų bibliotekos. Grafiniai elementai leidžia sudaryti vykdymo mechanizmų modelius priklausomai nuo naudojamos valdiklio programos. Bibliotekoje turėtų būti šių įrenginių modeliai:

- pneumatiniai cilindrai
- elektropneumatiniai skirstytuvai
- elektros varikliai
- temperatūros reguliatoriai
- indikatoriai
- produktų talpos
- įvairių tipų jutikliai
- elektromechaniniai jungtukai
- kiti grafiniai elementai

Taip pat gali būti galutinės grafinės schemos kurias pasirenka vartotojas modeliuodamas valdiklių darbą. Galutiniu modeliavimo rezultatu turi būti reali valdiklio programa, atitinkanti užduotas darbo diagramas arba būsenas. Užkrovus programą į valdiklį ir prijungus jį prie realios mechatroninės sistemos ji turi atlikti tuos pačius judesius kaip ir modelis. Darbo rezultatas – modeliavimo būdu sukurta ir suderinta programa, veikianti su realia automatinio valdymo sistema. Šiame magistriniame darbe pasirinktas iš anksto sudarytų modelių variantas. Tai apsprendė poreikis ištirti PLV ir kuriamos programos suderinamumą bei galimybė pateikti studentams pneumatinių sistemų judesio diagramas ir vertinti programavimo uždavinių sprendimo teisingumą. Projekto vystymo stadijoje numatoma kurti programą su laisvai pasirenkamų komponentų bibliotekomis.

3.2.3. Nefunkciniai reikalavimai

Nefunkciniai reikalavimai apriboja kuriamą sistemą arba kūrimo procesą. Atlikus šiuo metu esamos panašios programinės įrangos pavyzdžius, galima nustatyti kuriamos programos reikalavimus kompiuterių aparatūrinei ir programinei įrangai. Aparatūrinė įranga turi atitikti Visual Basic 6.0 ir Internet Explorer 6.0 programinei įrangai keliamus reikalavimus. Projekto kūrimui naudojami IBM šeimos personaliniai kompiuteriai, kuriuose gali būti įdiegta Windows 9x, Windows 2000 arba Windows XP operacinės sistemos, Visual Studio programinis paketas, Festo valdiklių programavimo paketai FST 4, Isagraf, FluidSim, Mitsubishi valdiklių programa Alpha Programming ir kitos reikalingos programos. Projektui taip pat reikalingi Festo firmos programuojamieji loginiai valdikliai ir jų sąsajos.

1. Planuojama techninė įranga - > 450 MHz dažnio procesorius, 20 GB kietas diskas, 128 MB operatyvinė atmintis, 16 MB vaizdo plokštė.
2. Operacinė sistema -Windows 9x, NT, 2000.
3. PLV tipas: FESTO FST – 20, FST -34, FST – 160.
4. Planuojama programos apimtis < 40 megabaitų.

3.2.4. Srities reikalavimai

Programinės įrangos procese būtina numatyti ir specialius, srities , reikalavimus. Kuriamas projektas numatomas naudoti realiu laiku veikiančiose kritinėse sistemose – programuojamuose loginiuose valdikliuose. Valdant technologinius procesus ir sistemas nepatikima programinė įranga arba neteisingi jos parametrai gali sukelti avarijas. Projekto darbo kokybei ir patikimumui keliami reikalavimai yra susiję su konkrečiais vykdymo mechanizmais. Mažesni reikalavimai gali būti keliami lėtesnėms, pneumatinėms mechatroninėms sistemoms ir griežtesni sistemoms su elektriniais ir elektroniniais vykdymo įtaisais.

Numatomi tokie programinės įrangos reikalavimai suderinti pagal taikymo sritį:

- Sistemos reakcijos laikas į poveikį - < 20 ms.
- Vartotojo įvedamų duomenų sritis – griežtai apibrėžta.
- Sutrikimų dažnis - <0,02.
- Atsistatymo laikas – nustatomas pagal PLV technines sąlygas.

3.2.5. Reikalavimai vartotojo sąsajai

Sistemos vartotojas dažnai sprendžia apie sistemą iš jos sąsajos, o ne iš sistemos funkcionalumo. Blogai suprojektuota sąsaja gali būti vartotojo klaidų priežastimi. Prasta vartotojo sąsajos architektūra yra pagrindinė priežastis, kodėl daugelis programinės įrangos sistemų yra nenaudojamos. Daugelis vartotojų sąveikauja su sistemomis per grafinę sąsają, nors, kai kuriais atvejais dar naudojamos ir tekstu pagrįstos sąsajos. Grafinėse sąsajose naudojami langai, ikonos, meniu ir įrankių juostos ir kiti elementai.

Grafinės sąsajos privalumai:

- Lengvai išmokstama ir vartojama.
- Vartotojas gali greitai pereiti (persijungti) nuo vieno darbo prie kito ir gali sąveikauti su keletu skirtingu programų vienu metu.
- Greita, pilno ekrano sąveika yra įmanoma naudojant momentinį priėjimą prie bet kurios ekrano vietos.

Trūkumai:

- Projektavimas orientuotas į konkretų vartotoją - turi būti užtikrintas programuotojo ir vartotojo ryšys.
- Būtina kurti sąsajų prototipus
- Projektuotojai turi žinoti žmonių fizinius ir mentalinius apribojimus ir turi suprasti, kad žmonės daro klaidas.

Projektuojant vartotojo sąsajas būtina laikytis šių principų:

- vartotojo pažinimas;
- nuoseklumas;
- minimalus nustebimas;
- atstatomumas;
- vartotojų skirtingumas;

Projektui parenkama grafinė sąsaja. Ji neturi būti labai smulkmeniška ir perkrauta nereikalingais elementais. Vartotojas turi galimybę persijungti nuo vieno programos modulio prie kito. Komandos parenkamos standartinės, kad vartotojas iš anksto galėtų numanyti jų darbo rezultatus. Numatyta galimybė atšaukti vartotojo veiksmus. Programos vartotojai bus studentai ir inžinerinis – techninis personalas, kurio kompiuterinis raštingumas yra gana aukšto lygio, todėl sąsajoje galima naudoti specialius kompiuterinius terminus ir standartines ikonas. Panaudota meniu pasirinkimo ir tiesioginio manipuliavimo sąsajų derinys . Nestandartinės komandos išskirtos panaudojant spalvų sistemą. Vartotojas galės keisti grafinės sąsajos šriftus ir spalvas savo nuožiūra. Atsiradus problemoms turi būti numatyta galimybė iškviesti pagalbos bylas . Pranešimai apie klaidas turi būti labiau teigiami nei neigiami ir gerai suprantami vartotojui.

Vartotojo sąsaja programuojama Visual Basic kalba naudojant lietuviškus terminus, tačiau nenaudojant lietuviškų raidžių. Tai leis išvengti ženklų atpažinimo klaidų sistemose, kuriose nėra įdiegti reikiami šriftai. Sąsajos valdymo elementų pavadinimai suderinti su lietuviškais kompiuterinėje technikoje vartojamais terminais. Kadangi programa skirta dirbti realiu laiku, kartu su PLV operacinėmis sistemomis, projekto realizavimo metu gali tekti atlikti daug eksperimentų , keičiant vartotojos sąsajos elementus ir kodą. Vartotojo sąsajos pavyzdys pateikiamas priede Nr. 4.

3.3 Realaus laiko sistemų programavimas

Projektuojama sistema yra priskiriama realaus laiko sistemai. Tokios sistemos stebi ir valdo savo aplinką ir yra neišvengiamai susijusios su aparatūrine įranga – jutikliais ir judikliais. Laikas tokioms sistemoms yra esminis kriterijus. Realaus laiko sistemos privalo reaguoti per tam tikrą laiką. Realaus laiko sistema yra programinės įrangos sistema, kurios teisingas funkcionavimas

priklauso nuo gaunamų rezultatų ir laiko, per kurį šie rezultatai gaunami. Jei per nustatytą laiką rezultatai nėra gaunami, tai tokia programa nefunkcionuoja. Realaus laiko sistemos paprastai yra realizuojamos kaip tarpusavyje sąveikaujantys procesai, kuriuos valdo realaus laiko vykdiklis. Projektuojant tokias sistemas būtina:

- Nustatyti apdorojamus poveikius ir kokios turi būti reakcijos į juos.
- Kiekvienam poveikiui ir reakcijai nustatyti laiko apribojimus.
- Apjungti poveikio ir reakcijos apdorojimą į lygiagrečius procesus, susijusius su kiekviena poveikio ir reakcijos klase.
- Suprojektuoti algoritmus kiekvienos poveikio ir reakcijos klasės apdorojimui su nustatytais laiko apribojimais.
- Suprojektuoti planavimo sistemą, kuri užtikrintų, kad procesų vykdymo pradžios laikas būtų parinktas taip, jog juos būtų spėta užbaigti iki kito proceso pradžios.
- Apjungti realaus laiko vykdiklius, programą ir operacinę sistemą

Realaus laiko sistemoms modeliuoti gali būti naudojami baigtiniai automatai arba UML diagramos. Tokio baigtinio automato schema yra panagrinėta aprašant CENTAURUS programinę įrangą, todėl šiame skyriuje ji neanalizuojama.

3.4 Programavimo kalbos parinkimas

Programos vartotojui nėra labai svarbu kokia kalba yra parašyta programa, svarbu kad ji atliktų numatytas funkcijas ir būtu patogi vartojimui. Galima šiek tiek palyginti šiuo metu naudojamas objektinio programavimo kalbas (15).

Programuotojai naudoja Microsoft Visual C++ arba jos modifikacijas. Pagrindinis šios kalbos privalumas yra labai nedidelės talpos EXE byla, kurios dydį galima lyginti su Asemblerio exe byla. C++ exe failą generuoja vidinis statinis kompiliatorius ir jo paleidimui nereikalingos papildomos (runtime) bibliotekos. Jei programa yra kraunama per internetą, jos apimtis yra vienas iš svarbesnių rodiklių. Visual C++ kalba yra rašomos DLL bylos ir Active X komponentai. Kadangi sukurtų komponentų kodas yra dvejetainis, darbo greitis yra didelis. Dar vienas C++ privalumas yra bylų pernešamumas į kitas platformas, pav. programos sukurtos Windows OS gali būti lengvai transformuojamos į Linux ar BeOS operacinių sistemų aplinką.

Be abejo C++ turi ir trūkumų. Kadangi ją daugiausiai naudoja profesonalūs programuotojai labai mažai yra nemokamų bibliotekų ir komponentų. Pats programavimo kalbos mokymasis yra sunkus, nors literatūros pastaruoju metu šia tema daugėja.

Kita šiuo metu sparčiai populiarėjanti programavimo kalba yra Borland Delfi. Jos populiarumą apsprendžia dar ir tai kad komandų kodas primena Paskalio kalbos kodą, kurio

programavimo pagrindai yra dėstomi vidurinėse mokyklose. Kaip ir C++ Delfi kalboje sukurtos programos yra kompiliuojamos į exe bylas nenaudojant papildomų bibliotekų. Paleidžiamųjų bylų apimtis viršija analogiškų programų, sukurtų C++ apimtį, bet paprastai nera labai didelės. Tuo Delfi programos skiriasi nuo pagrindinio savo konkurento Visual Basic . Pagrindiniai sunkumai programuojant Delfi kalba yra rezidentinių programų kūrimas. Šių programų talpa ribojama keliais šimtais kilobaitų. Kurti Active X komponentus Delfi kalba yra beprasmiška: kompiliatoriaus generuojama bylų talpa yra didelė ir dėl to komponentų darbo greitis yra mažas.

Programos Visual Component Library (VCL) yra labai plati ir beveik visi jie platinami nemokamai. Be to beveik visų komponentų programų kodai yra su paaiškinimais , todėl juos naudoti neprofesionaliam programuotojui nėra labai sudėtinga. Atviras kodas leidžia juos modifikuoti arba naudoti tik kodo fragmentus. Dar viena gera ypatybė – komponentus išleidžia įvairios firmos, todėl jei yra klaidų , galima atsisiųsti kito gamintojo produktą.

Microsoft Visual Basic yra viena iš senesnių objektinio programavimo kalbų (7). Kaip ir kitos ji pasižymi savo privalumais ir trūkumais. VBA (Visual Basic for Aplication) naudojama Microsoft Office paketo valdymui. Dauguma Office programų turi makrokomandas, programuojamas VB kalba. Todėl gana paprasta yra kurti biuro programų papildymus ir juos testuoti. Kitas privalumas – kodas gana panašus į Basic kalbos operatorius todėl tiems, kurie kažkada vietoje Paskalio mokėsi Beisik‘ą, programuoti nesunku.

Tačiau trūkumų yra daugiau nei privalumų. Gal būt iš dalies dėl to, kad praeitis šiai programai pridėjo didelį balastą – papildomas dll bibliotekas. Sukūrus bet kokį exe failą nei viename kompiuteryje jis nepasileis, jei ten nėra msvbvm50.dll arba msvbvm60.dll bylų (priklausomai nuo Visual Basic versijos). Bylų talpa yra daugiau nei megabaitas ir senesnėse Windows versijose jų nėra. Tik nesenai Microsoft koorporacija pradėjo įdiegti šias dll bylas į operacinių sistemų sudėtį. Jei kuriamos programos talpinamos internete aišku galima papildomai įdėti ir bibliotekas, tačiau tada programos talpa gali išaugti kelis kartus.

Kitas trūkumas yra Active X komponentų panaudojimas programose. Šie komponentai paprastai sukurti konkrečiai Windows versijai. Perėjus į kitą platformą gerai dirbanti programa gali strigti. Taip pat atsiranda konfliktai tarp seno ir naujo tipo Active X komponentų. Klaidų taisyti negalima, nes kodas yra uždaras. Pačių komponentų apimtis irgi yra gana didelė. Pavyzdžiui įdėjus į programą Windows Commom Controls , nepriklausomai kas iš šio komplekto bus naudojama, programą papildo 650 kilobaitų. Naudojant OLE technologijas darbui su biuro programų produktais – dar keli šimtai kilobaitų. Windows 2000, XP sistemoms yra sukurti Visual Basic NET produktai, kurių talpa jau siekia keliasdešimt megabaitų.

Apžvelgus programavimo kalbų paketus galima daryti išvadą, kad mažiausios apimties produktai gaunami naudojant Visual C++ kalbą. (neskaitant Asemblerio, kuriuo programuoti moka

nedaugelis ir be to būtinas geras procesorių architektūros žinojimas). Sekanti po Visual C++ būtų Delfi. Ir tik kuriant programas, kurių apimtis nėra kritinis rodiklis, galima naudoti Visual Basic. Dar liko nepaminėta Java programavimo kalba, kuri šio metu sparčiai vystosi ir pagal galimybes lygiuojasi su C++.

Magistrinis darbas pagal apimtį yra didelis. Jame yra numatytas teorinės medžiagos ir praktinių užduočių pateikimas, naudojant .doc formatą. VB programavimo kalba leidžia naudoti OLE komponentus Misrosoft Offise programų atidarymui (16). VB dinaminių bibliotekų apimtis lyginant su pačia programa yra santykinai nedidelė. Visual Basic kalba paprasta projektuoti vartotojo sąsajas. Projekto autoriui teko mokytis Basic ir Visual Basic programavimo kalbas. Dėl šių ir kitų priežasčių projektavimui ir buvo pasirinkta Visual Basic programavimo kalba.

3.5 Programinės įrangos architektūra

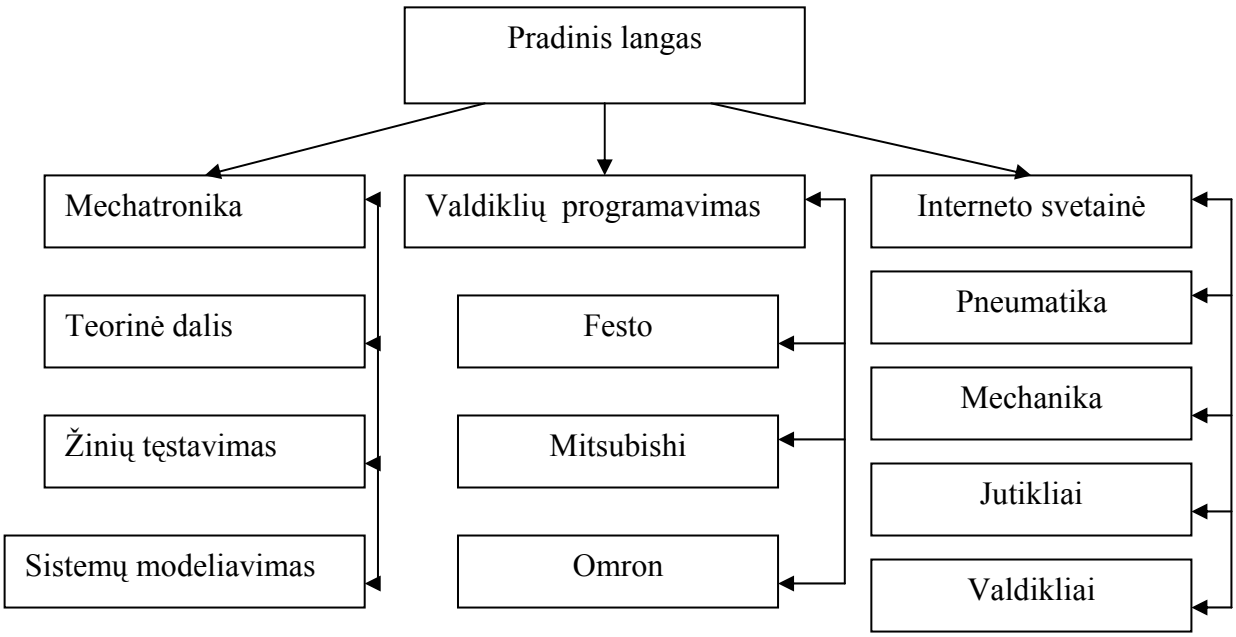
Projekto alternatyvos gali būti tokios: internetinis puslapis su nuorodomis į reikiamas programas ir autonominė programa apjungianti visus reikiamus komponentus panaudojant atskirus darbalaukius. Kadangi darbas yra skirtas gana siauram vartotojų ratui, priimtinesnis yra antrasis variantas, tai yra instaliuojama programa, su su visais jai reikalingais komponentais. Projektą sudaro kelios atskiros dalys (moduliai), kurias galima kurti nepriklausomai viena nuo kitos. Šias dalis galima suskirstyti į dvi grupes:

1. Teorinė medžiaga ir praktinės užduotys.
2. Sistemų modeliavimo programa.

. Teorines žinias yra patogiausia pateikti naudojant HTML formatą. Ši programavimo kalba leidžia greitai pasiekti norimą žinių šaltinį, suskirstyti duomenis pagal temas, sukurti nuorodas į papildomą literatūrą. Alternatyva būtų panaudoti programines Help bylas sukurtas panaudojant AnetHelpTools, RoboHelp Office ir kitus programinius paketus. Kuriant programą Visual Basic paketu patartina naudoti Help Composer , nes ji leidžia kurti What's This? užklausas tiesiai .exe bylose. Dar vienas pasirinkimas – teorinės žinios .pdf formatu. Šios alternatyvos privalumas – galimybė apriboti vartotojui kopijuoti ir spausdinti tekstus bei grafinę medžiagą.

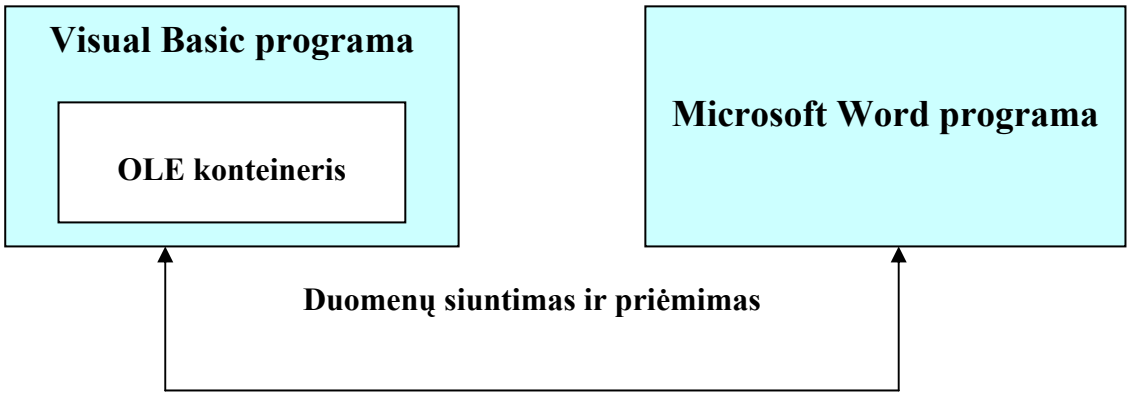
Projekto struktūra pateikiama toliau schemoje. Jį sudaro 16 Visual Basic kalbos formų ir du programiniai moduliai. Kiekviena forma veikia savarankiškai ir atlieka tam tikrą projekte numatytą funkciją.

.



15 pav. Projekto programinės įrangos struktūrinė schema

Teorijos ir praktinių užduočių modulis. Magistriniame darbe šis modulis yra projektuojamas atsižvelgiant į teorinių žinių , reikalingų darbui su programuojamų loginių valdiklių programomis , būtinumą. Studentai turi įsisavinti gana daug PLV programavimų paketų. Teorinėje dalyje pateikiami nurodymai, dėl darbo su Festo, Mitsubishi ir Omron firmų valdiklių programine įranga. Darbinėje programoje naudojant OLE konteinerį (16) yra įkeliamos Word‘o bylos, į kurias VB programa nukreipia ir, jei reikia, atidaro. Toks medžiagos pateikimas užtikrina greitą jos paiešką ir turinio keitimą. Toks pat metodas panaudotas ir praktinių užduočių pateikimui. Toliau pateikiamas OLE technologijų algoritmas.

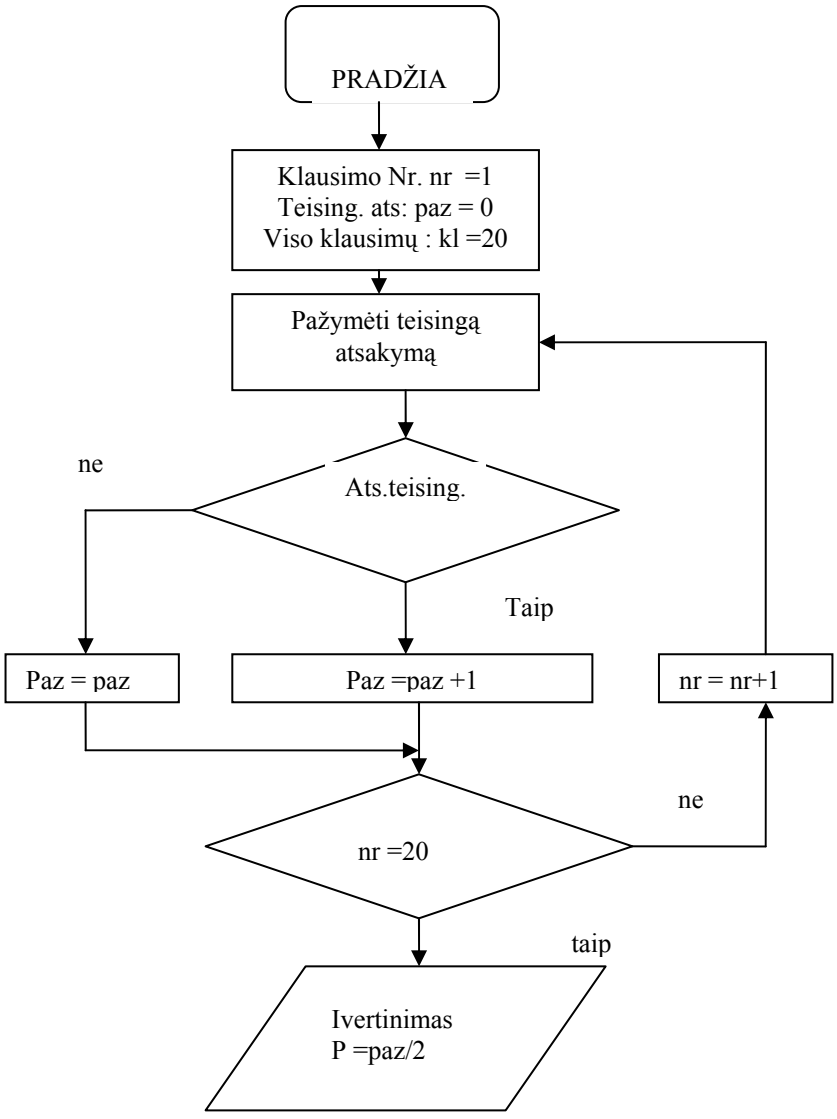


16 pav. Objektų ryšio schema naudojant OLE technologijas

Teorinių žinių tikrinimo testai yra skirti savarankiškai besimokantiems pasitikrinti , kaip yra įsisavinta teorinė medžiaga. Šiuo metu yra sukurta gana daug testavimo programų, tame tarpe ir lietuvių kalboje. Testai nėra pagrindinis šio projekto tikslas, todėl naudojamas tiesioginis „klausimas – atsakymo variantai“ algoritmas be papildomų bylų. Modulį sudaro pagrindinė programa, kurios teksto languose pateikiami klausimai ir galimi jų atsakymai. Studentas pažymi jo nuomone teisingą atsakymą ir tuo pačiu iškviečia sekantį klausimą. Atsakymų teisingumą tikrina paprogramė. Joje taip pat atliekamas balų skaičiavimas. Pilnai atsakius į viso testo klausimus, paprogramė pateikia įvertinimo pažymį. Testo algoritmas pateikiamas žemiau.

Privalumai: paprastas programavimas;
testo duomenys nepasiekiami vartotojui;

Trūkumai: negalima keisti klausimų ir atsakymų;
testas netinka žinių įsisavinimo savianalizei;
nėra grafinės informacijos pateikimo galimybės;



17 pav. Žinių tikrinimo testo algoritmas

Distancinio mokymo modulis yra skirtas studentams susipažinti su teorine medžiaga ir kai kurių mechatronikos disciplinų praktinių darbų užduotimis. Iš principo tai interneto tinklapis, kuriuo naudodamasis, studentas arba kursų klausytojas gali geriau pasiruošti studijoms. Modulis sudarytas iš pradinio lango, kuriame pasirenkama mechatroninės sistemos teorinė sritis. Pradinis langas yra kurtas CorelDraw programa pasirinkus atitinkamus grafinius vaizdus. Kiekviena nuoroda pateikia vartotojui teorinę medžiagą, o kai kur ir praktines užduotis. Kai kurioms esamoms nuorodoms puslapiai dar yra kūrimo stadijoje. Tolesnis puslapių programavimas atliekamas Microsoft ProntPage , Namo WebEditor bei DreamWeaver MX programomis. Kadangi medžiaga juose bus pastoviai atnaujinama, projekte yra pateikiamas tik šiuo metu Panevėžio kolegijos serveryje www.panko.lt/mechas/ esantis variantas.

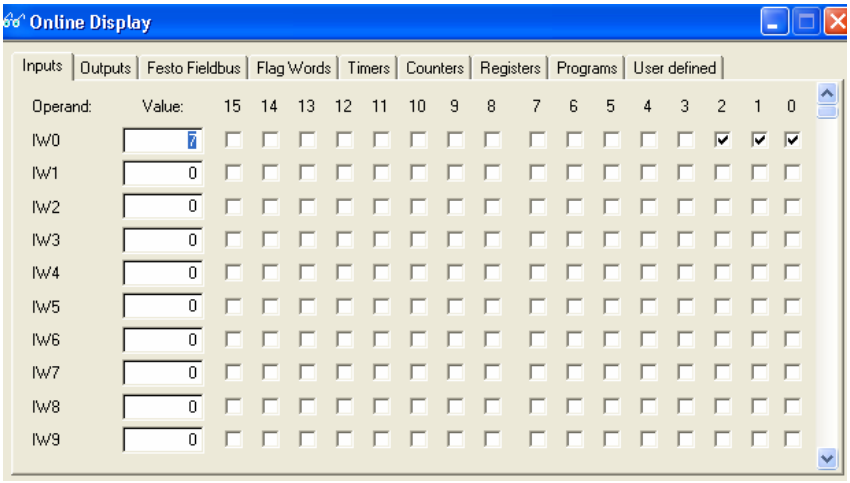
18 pav. Internetinės svetainės pradinis puslapis

. Į mechatronikos svetainę galima patekti per nuorodą programinėje įrangoje arba tiesiai iš kolegijos pradinio puslapio. Tinklapyje taip pat pateikiamos nuorodos į kitus su mechatroninėmis sistemomis interneto adresus. Panevėžio kolegija kartu su kitomis mokymo įstaigomis dalyvauja Phare projekte, kurio tikslas sukurti bendro naudojimo internetinę svetainę mechatronikos disciplinų studijoms. Projekte pateiktas internetinis modulis gali būti šios svetainės sudėtine dalimi.

Pasirinkus valdiklių programavimo dalį, galima susipažinti su valdiklių programų ypatumais ir programavimo metodika. Visual Basic programavimo kalboje yra panaudotos OLE technologijos, įgalinančios iškviesti ir paleisti bet kokią kitą programą esančią kompiuteryje. Tam tikslui į projektą įdedamas nuorodos kodas. Atidaromas tekstinis dokumentas Word aplinkoje arba valdiklių programavimo paketo metodikos aprašymas.

3.6 Sistemos modeliavimo įrangos kūrimas

Kuriant programą , turinčią galimybę imituoti realių sistemų veiklą buvo pasirinktas Festo firmos valdiklių programavimo paketas Beck Automation Office FST 4. Ši programa naudoja tekstinę kalbą STL . (struktūrizuotas tekstas) Tai, mano manymu, labai lengvai įsisavinama programavimo kalba, turinti daug galimybių. Tačiau pagrindinis FST – 4 trūkumas – nėra simuliacinio režimo . Prijungus valdiklį galima tik stebėti programos operatorių darbą linijos (on line) režime. Duomenys gaunami grafiniame lange arba programos teksto lange.



19 pav. PLV būsenų stebėjimo langas

Grafiniame lange galima pasirinkti stebėti valdiklių įėjimus, išėjimus, laiko reles, skaitiklius ir t.t. Projekte tyrinėjama galimybė panaudoti šiuos duomenis modelio, sukurto Visual Basic programa, valdymui. FST-4 bylos , į kurias įrašomi iš valdiklio gaunami ir atgal siunčiami duomenys yra užkoduoti. Informacijos iš jų paimti, bei jas keisti nėra galimybės. Sprendimo imti duomenis iš grafinio lango taip pat atsisakyta.

PLV yra jungiamas prie kompiuterio nuoseklaus (COM) prievado. Duomenys iš kompiuterio į valdiklį perduodami nuosekliai bitas po bito. „Gaudyti“ atskirus bitus nėra prasmės, nes informacija koduojama baituose. Kompiuteris ir valdiklis naudoja buferį, į kurį kaupiama nuosekli informacija . Iš buferio yra nuskaitomas lygiagretus kodas ir perduodamas į atmintį tolesniam apdorojimui. VB programoje naudojamų nuoseklių prievadų nustatymas pateikiamas lentelėje 2.

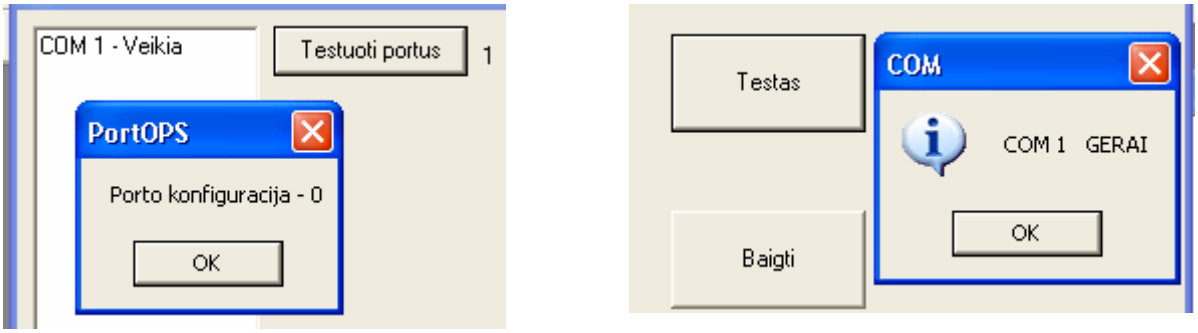
Nuoseklaus kompiuterio prievado valdymo komandos

Lentelė. 2

Savybės	**Pastabos**
CommPort	Nustato valdomo nuoseklaus porto numerį
Settings	Nustato duomenų perdavimo greitį, klaidų taisymo savybes, bitų skaičių

	viename pakete.
PortOpen	Atidaro arba uždaro nuoseklų portą
Input	Išveda informaciją iš imtuvo buferio.
Output	Įrašo informaciją į porto siųstuvo buferį
InputLen	Nustatomas imtuvo buferio ilgis

Naudojantis šiomis savybėmis pirmiausiai buvo sukurtas sąsajų tikrinimo testas. Naudojami du testo variantai, rodantys ar veikia nuoseklios kompiuterio sąsajos. Pirmame iš jų nurodoma konfiguracija, tai yra standartinis perdavimo greitis, 8 bitų buferis ir vienas „stop“ bitas. Antrame teste nustatoma ar yra prie porto prijungtas valdiklis. Testavimo programų kodai pateikiami projekto priede 2.

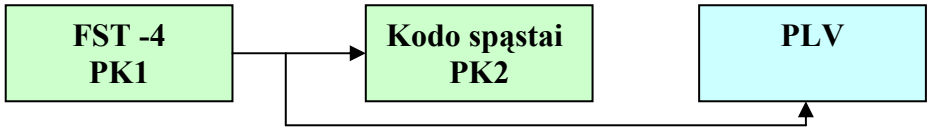


20 pav. COM sąsajos testavimo programų langai

VB programa ima informaciją apie portų veiklą saveikaudama su operacinės sistemos bylų biblioteka (kernel. dll).

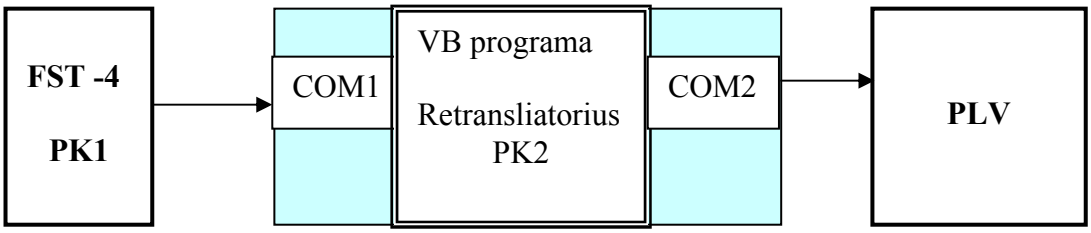
Sekantis projektavimo etapas – išsiaiškinti kokią informaciją PLV programa siunčia į valdiklį. Tam tikslui sukurtas VB prototipas imituojantis valdiklio imtuvą. Du kompiuteriai sujungiami panaudojus COM kabelį, tai yra organizuojamas „nul-modemas“. Viename kompiuteryje įdiegiama FST – 4 programa, o kitame VB kalba sukurta kodo „gaudymo“ (spąstų) programėlė. Įjungus valdiklio paieškos komandą „Login“ imtuvo programa teksto lange išduoda priimtą kodą. Tyrinėjant nustatyta , kad užklausos metu FST – 4 siunčia skaičių rinkinį **76, 63, 13**. Panaudojus ASCII kodų lentelę (19) matoma, kad tai ženklai **L, ? , Enter** išreišti dešimtainėje skaičiavimo sistemoje.

Tyrimas su dviem kompiuteriais neleido gauti detalesnės informacijos, nes valdikliui neatsakius, gaunamas pranešimas, jog prisijungti nepavyko. Sudaroma schema iš trijų komponentų: dviejų kompiuterių ir valdiklio. Panaudojamas lygiagretus nuoseklių sąsajų sujungimas elektriškai.



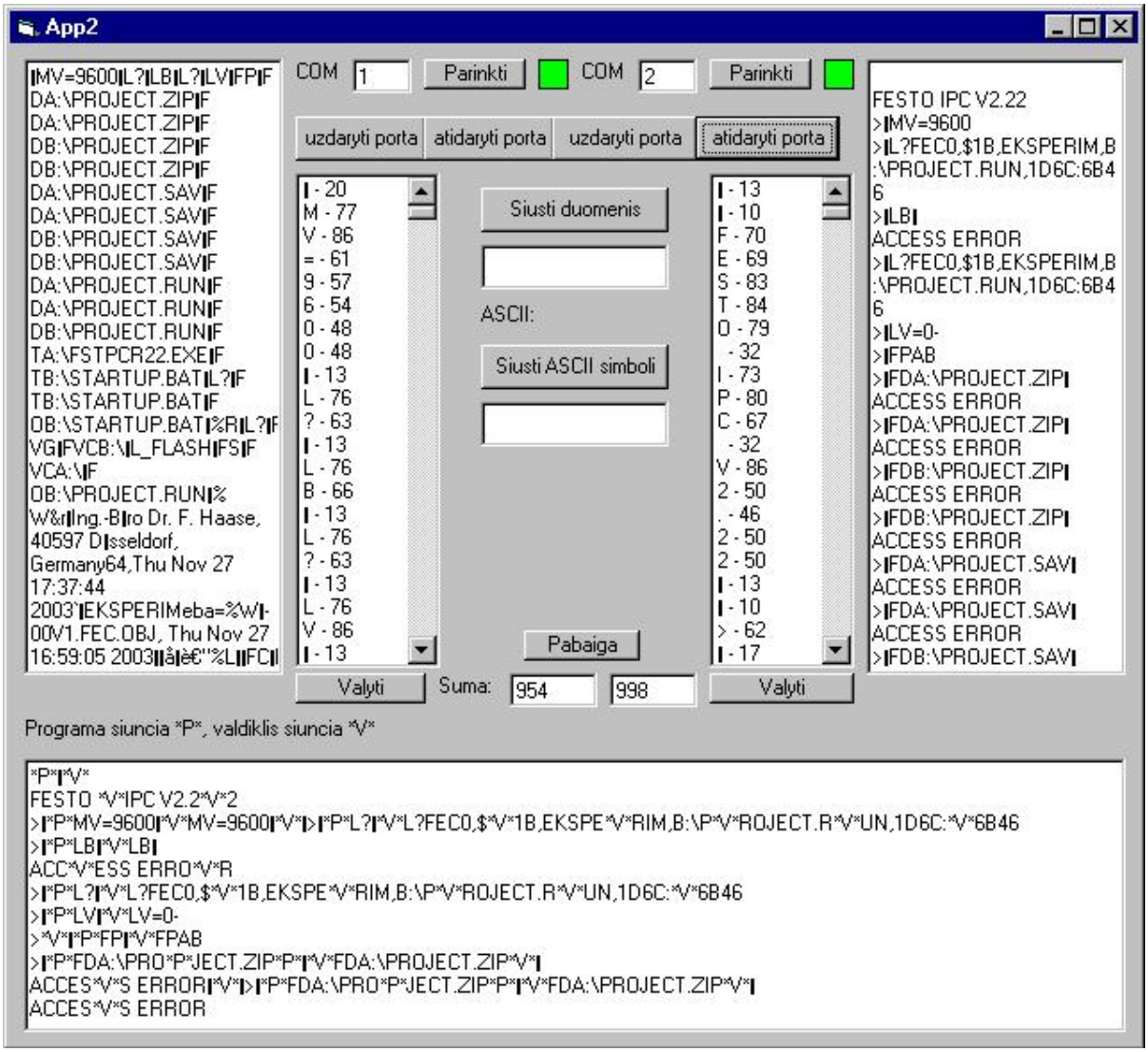
21 pav. Įrangos jungimas nustatant programinius kodus

Šis bandymas nebuvo sėkmingas, nes aparatūrinis nuoseklios sąsajos sujungimas pakeičia elektrinius signalų parametrus. Teko pasinaudoti programiniu jungimu. Kompiuteris su PLV programa jungiamas prie tarpinio kompiuterio COM1, o valdiklis – prie COM2. Sukuriama VB programėlė retransliuojanti informaciją iš COM1 į COM2 ir atvirkščiai. Taip pat visa informacija yra išvedama į tarpinio kompiuterio programos langus tekstiniame pavidale. Šis bandymas pasitvirtino. Programos lange galima stebėti kokius kodus perduoda FST – 4 į valdiklį ir kaip valdiklis reaguoja į šiuos kodus.

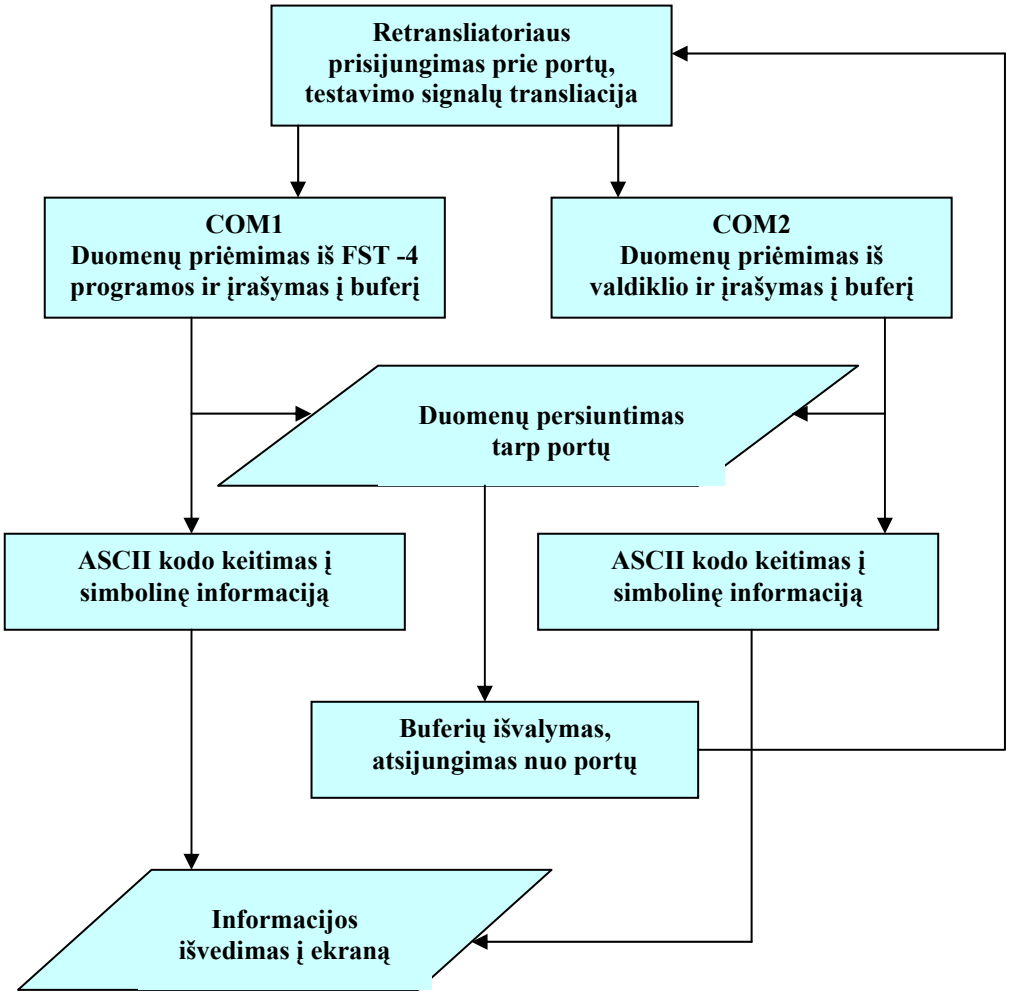


22 pav. Įrangos jungimas siunčiamų ir priimamų kodų nustatymui

Papildomai į tarpinį kompiuterį reikia įdiegti msvbvm60.dll ir mscomm32.ocx bylas.Toliau pateiktame retransliatoriaus lango pavyzdyje yra gautų rezultatų tekstai. Kairėje pusėje yra langas kuriame matoma siunčiama programa, dešinėje – ką siunčia valdiklis. Papildomai rodoma ir ASCII lentelės šešioliktainis kodas.. Apatinėje juostoje bendras „dialogas“. Kad būtų galimybė atskirti kodus naudojamos papildomos raidės. *P* - programos kodo informacija o *V*- valdiklio kodas. Retransliatoriaus programos kodas pateikiamas prieduose (priedas 3). Retransliatoriaus sąsajos langas ir darbo algoritmas yra 23, 24 paveikslėliuose..



23 pav. Retransliatoriaus lango vaizdas



24 pav. FST-4 – PLV programų retransliatoriaus darbo algoritmas.

Daba,r esant galimybei stebėti informaciją, pereinama į sekantį etapą – nustatyti kaip yra valdomi valdiklio įėjimai ir išėjimai darbo režime, kai programa yra užkrauta. Sukuriama paprasta programėlė STL kalboje, kurioje du jutikliai (mygtukai) valdo du išėjimus. Paspaudus mygtuką išėjimas įjungiamas, atleidus – išjungiamas.

IF input1 THEN SET output1

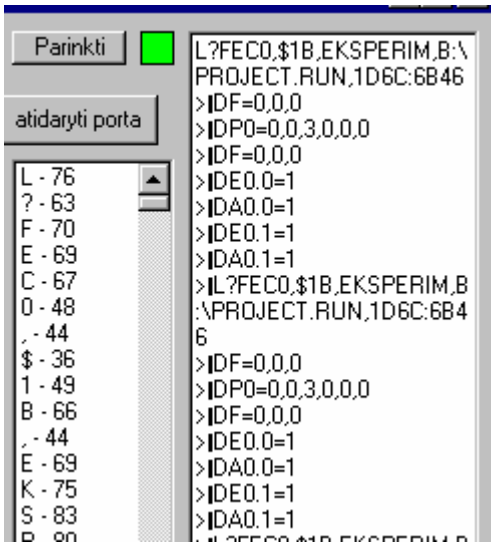
IF N input1 THEN RESET output1

IF input2 THEN SET output2

IF N input2 THEN RESET output2

Įjungiamas PLV darbo stebėjimo „on line“ režimas. Spaudžiant ir atleidžiant mygtukus valdiklio stende, stebima kaip keičiasi programinio monitoriaus informacija. Jau valdiklio pajungimo ir krovimo metu yra pastebėta, kad siunčiama ir priimama logiškai suprantama informacija, pav. frazės ACCESS ERRORR arba PROJECT .RUN. Tikėtina, kad linijos režimu dirbantis valdiklis, pasikeitus jo būsenai, turėtų siųsti su įėjimais ir išėjimais surištą informaciją.

Bandymais nustatyta, kad paspaudus mygtuką , valdiklis siunčia būseną pakeitusio įėjimo absoliutinį adresą ir jo reikšmę. Analogiškai keičiasi ir išėjimo būsena. Vadinasi jeigu VB modeliavimo programai pavyktų pakeisti įėjimo būseną programiškai, išėjimo signalas irgi patektų į programą. Bandymo rezultatai buvo teigiami.



25 pav. **V**aldiklio pranešimai apie būsenų pokyčius

Eilutė DE0.0=1 reiškia kad PLV įėjimų nulinio žodžio 0 bitas lygus vienetui, tai yra jutiklis, prijungtas prie šio įėjimo, yra įjungtas. DA rodo valdiklio išėjimų būseną. Atlikus tyrimus bei pasinaudojus FST-4 pagalbos bylomis nustatytos komandos ir raktiniai žodžiai perduodami į valdiklį ir iš jo. Pagal gautus duomenis sudarytas komandų interpretatorius (CI).

Prisijungimo prie valdiklio tvarka

Lentelė 3

Nr.	Programos siunčiami signalai	Valdiklio atsakymai	Pastabos
1	#20	#13#10FESTO IPC V2.2#13#10>#17	Valdiklis atsiunčia savo pavadinimą ir versijos numerį.
2	MV=9600#13	MV=9600#13	Programos ir valdiklio duomenų perdavimo greičio nustatymai.
		#10>#17	

Norint kokius nors duomenis gauti „masiškai“ visus iš karto, po komandos rašomas simbolis minusas „-“. Pvz., Jei norime gauti visas 16 išėjimo būsenų reikia rašyti komandą pvz., DA0.0-.

DC4 (Ctrl+T arba ASCII #20) – prisiregistravimas ir branduolio versijos gavimas.

Paleidus šią komandą, visų kitų komandų veikimas būna nutraukiamas. FESTO valdiklis atsako „FESTO IPC V2.xx“ , (xx – yra branduolio versijos Nr.).

X – ši komanda atlaisvina nuoseklųjį portą, baigia sesiją (po šios komandos norint siųsti komandas teks vėl prisijungti).

DA [<VN>**.**]**<**ŽN>**.**<BN> – Parodo išėjimo bito būseną. VN-valdiklio nr., ŽN – žodžio nr. BN – bito nr.

(Pvz.: komanda DA0.0 ir tarkim atsakymas DA0.0=0, DA0.x – x būna nuo 0...15)

DAW [VN>**.**]<ŽN> – Parodo įšėjimo žodį.

(Pvz.: komanda DAW0 atsakymas DAW=4, reiškia yra suveikęs trečias išėjimas, tai tas pats kas atsakymas DA0.3=1. Jei atsakymas DAW=3, aktyvūs yra išėjimai 0.0 ir 0.1 ir t.t.)

DD – Parodo kokie yra vaizdavimo nustatymai dešimtainio ar šešioliktainio skaičiaus.

– DD=D – dešimtainis be ženklo;

– DD=S – dešimtainis su ženklu;

– DD=H – šešioliktainis skaičius.

(Pvz.: komanda DD ir tarkim atsakymas DD=S)

DE [<VN>**.**]<ŽN>**.**<BN> – Parodo įėjimo bito būseną.

(Pvz.: komanda DE0.0 ir tarkim atsakymas DE0.0=1 – tai reiškia įėjimas veikia)

DEW [VN>**.**]<ŽN> – Parodo įėjimo žodį.

(Pvz.: komanda DEW0 atsakymas DEW=3, reiškia yra suveikę du įėjimai 1 ir 2, tai tas pats kas atsakymai DE0.0=1 ir DE0.1=1)

DM <ŽN>**.**<BN> – Parodo vėliavėlės (flag) bito būseną.

(Pvz.: komanda DF0.0 tarkim atsakymas DF0.0=1 – tai reiškia atmintis įjungta)

DF – Parodo klaidos būseną. (Pvz.: komanda DF atsakymas DF=0,0,0 reiškia klaidų nėra)

DP <PN> – Parodo programos būseną.

(Pvz.: komanda DP0 grąžina tokį rezultatą – DP0=0,0,3,1,0,0)

Pirmas skaičius - modulio tipas STL=0; LDR/FUP=1; C=2;

Antras skaičius - atminties ribos – visada 0;

Trečias - programos būsena 0 – neaktyvi; 2 – aktyvi sustabdyta; 3 – aktyvi veikianti;

Ketvirtas STL kalboje aktyvaus žingsnio Nr., LD kalboje aktyvios šakos Nr. Likę du skaičiai yra iškviesto modulio Nr. ir atitinkamai žingsnis (step).

DR <RN> – Parodo registro būseną; (Pvz.: komanda DR0 atsakymas DR0=–16689)

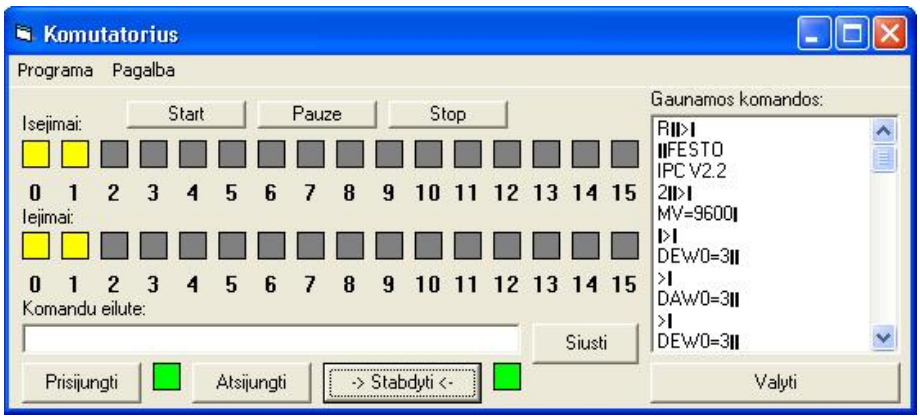
L? – Parodo sisteminius duomenis t.y. kontrolerio tipą (pvz.: HC1x ar FECx – x yra kontrolerio Nr.), šešioliktainį būsenos kodą, projekto vardą, projekto bylą (dažniausiai C:\PROJECT.RUN), aptarnavimo struktūros adresą. Pvz., šiuo atveju paleidus L? komandą valdiklis atsako:
L?FEC0,$1B,EKSPERIM,B:\PROJECT.RUN,1D6C:6B46 Šešioliktainis būsenos kodas pvz., gali turėti tokias reikšmes:

DV - Parodo esamą duomenų perdavimo greitį. (Pvz.: komanda DV rezultatas DV=9600)

MV - Parenka duomenų perdavimo greitį.

Gali būti tokie greičiai: „MV=1200“, „MV=2400“, „MV=4800“, „MV=9600“, „MV=19200“, „MV=38400“ ar „MV=56000“. Duomenų perdavimo greitis gali būti nurodomas, kad ir 2 simboliais, pavyzdžiui, „MV=96“ bus tas pats kas „MV=9600“.

Šių tyrimų rezultate sukurtas dar vienas modeliavimo programos prototipas, kurio paskirtis tiesiogiai valdyti FESTO valdiklį. Sukūrus programą STL kalboje ir užkrovus ją į valdiklį, FST -4 programinė įranga atjungiama. Valdiklis lieka prijungtas prie kompiuterio tiesiogiai, nebenaudojant retransliatoriaus. Tolesnis PLV darbas stebimas ir valdomas naudojant žemiau pateiktą programinį langą. Mygtukas “start” paleidžia valdiklyje esančią programą, “Stop” ją sustabdo. Komandų eilutėje galima rašyti norimą komandą ir ji bus perduota į PLV. Įėjimų ir išėjimų indikatoriai rodo jų būsenas.

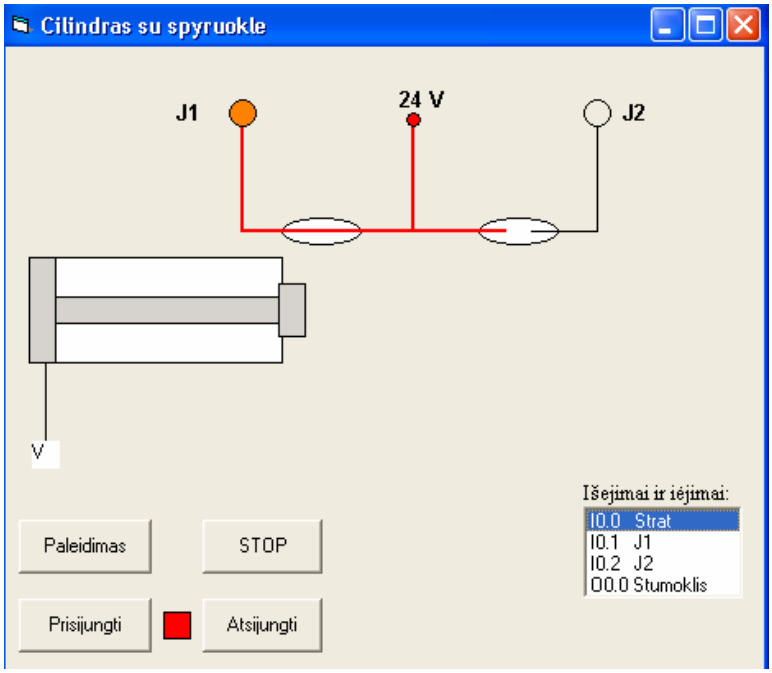


26 pav. Modeliavimo programos prototipas

Programos pagalba galima įjungti bei išjungti valdiklių įėjimus ir išėjimus kreipiantis į konkrečius absoliutinius adresus. Įėjimai modifikuojami komandos ME 0.0=x pagalba, kur x =1 arba 0 Jei valdiklis yra prijungtas prie vykdymo įrenginių ir jutiklių, šis sąsajos prototipas atlieka vizualizacijos funkcijas. Pagal tai , kokia programa yra vykdoma, keičiasi indikatorių būsenos.

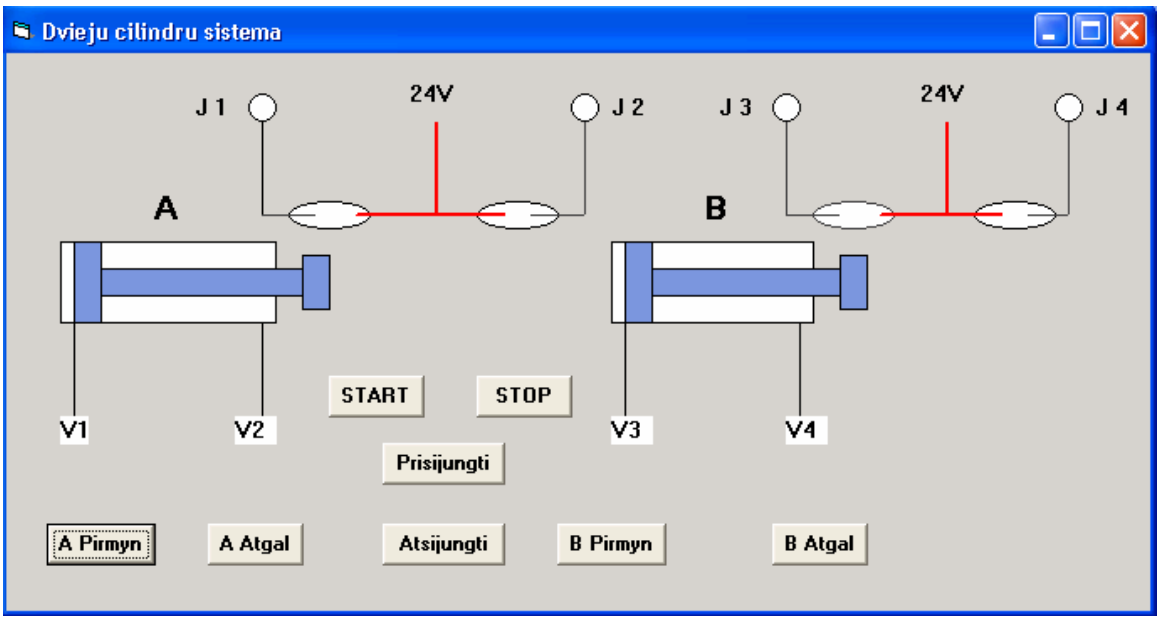
3.7. Elektropneumatinių sistemų modelių tyrimas

Tolesnis valdiklių darbo modeliavimo tyrimas jau yra susietas su technologinių procesų grafinių vaizdų valdymu (5). Kaip minėta anksčiau, daugelyje vykdyklių naudojami pneumatiniai cilindrai. Jų pagrindu Visual Basic aplinkoje imituojami judikliai. Kaip jutikliai kai kuriose schemose naudojami herkoninių kontaktų grafiniai vaizdai. Sukuriamos kelios grafinės aplinkos, imituojančios monostabilius ir bistabilius cilindrus.



27 pav. Monostabilaus pneumatinio cilindro modelis

Prisijungus prie valdiklio iš modelio sąsajos perduodami ir gaunami signalai, kuriuos VB programa dekoduoja ir paverčia grafinio vaizdo valdymo komandomis. Mygtukas “Paleidimas” imituoja “Start” mygtuko darbą FST - 4 programoje. Ši sąsaja leidžia modeliuoti tik nedidelės apimties STL kalbos programėles. Gali būti naudojami trys jutikliai ir vienas vykdiklis. Sudėtingesnėms programoms modeliuoti yra skirta dviejų bistabilių cilindrų ir keturių jutiklių sistema. Čia galima imituoti pagal įvairias judesio diagramas sudarytų programų veikimą, tikrinti jų algoritmus, modeliuoti laiko reles ir skaitiklius.



28 pav. Dviejų cilindrų elektropneumatinis modelis.

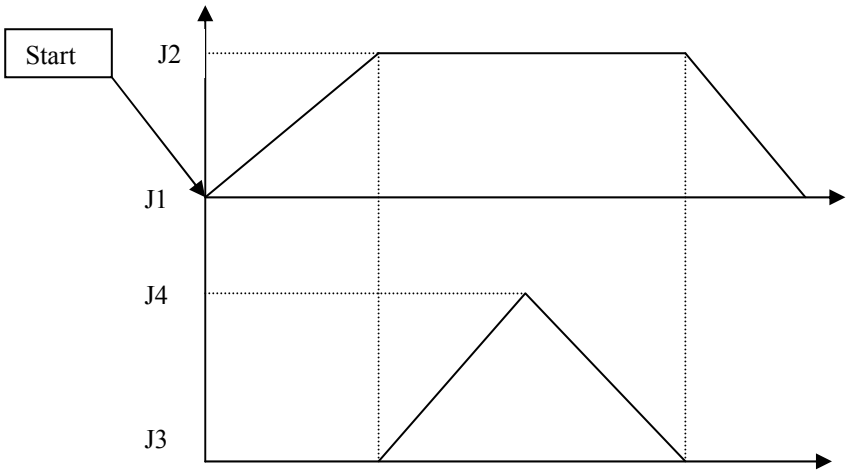
Programos lange yra du pneumatiniai cilindrai, kurie gali išstumti arba įtraukti stumoklius, priklausomai nuo signalų V1 – V4, imituojančių cilindrų oro magistrales. Su cilindrų stumokliais grafiškai yra surišti herkoniniai jutikliai J1 – J4. Šie jutikliai atitinkamai veikia, jei stumokliai yra galinėse padėtyse. Modelį galima valdyti rankiniu būdu, naudojant lange esančius mygtukus, arba programiškai. Projekto programinė įranga, PLV programavimo kalba FST – 4 ir programuojamas loginis valdiklis FST –FEC -34 susieti sekančiai:

Programų tarpusavio kodų sąryšis

Lentelė 4

VB programinis modelis	FST – 4 programa	PLV kodas
V1	Išėjimas O0.0; Aplus	DA0.0
V2	Išėjimas O0.1; Aminus	DA0.1
V3	Išėjimas O0.2; Bplus	DA0.2
V4	Išėjimas O0.3; Bminus	DA0.3
Start	Įėjimas I0.0	DE0.0 ME0.0=x
J1	Įėjimas I0.0	DE0.1 ME0.1=x
J2	Įėjimas I0.0	DE0.2 ME0.2=x
J3	Įėjimas I0.0	DE0.3 ME0.3=x
J4	Įėjimas I0.0	DE0.4 ME0.4=x

Keičiant bet kurį iš pateiktos lentelės stulpelio elementą programose arba valdiklyje keičiasi ir kiti du kodai, esantys toje pat eilutėje. Belieka modelyje nustatyti reikiamas objektų savybes, kurios priklausys nuo šių kodų. Tyrimams pasirenkama pneumatinės sistemos, sudarytos iš dviejų cilindrų, judesio diagrama, sudaromos loginės lygtys, programa ir atliekami bandymai.

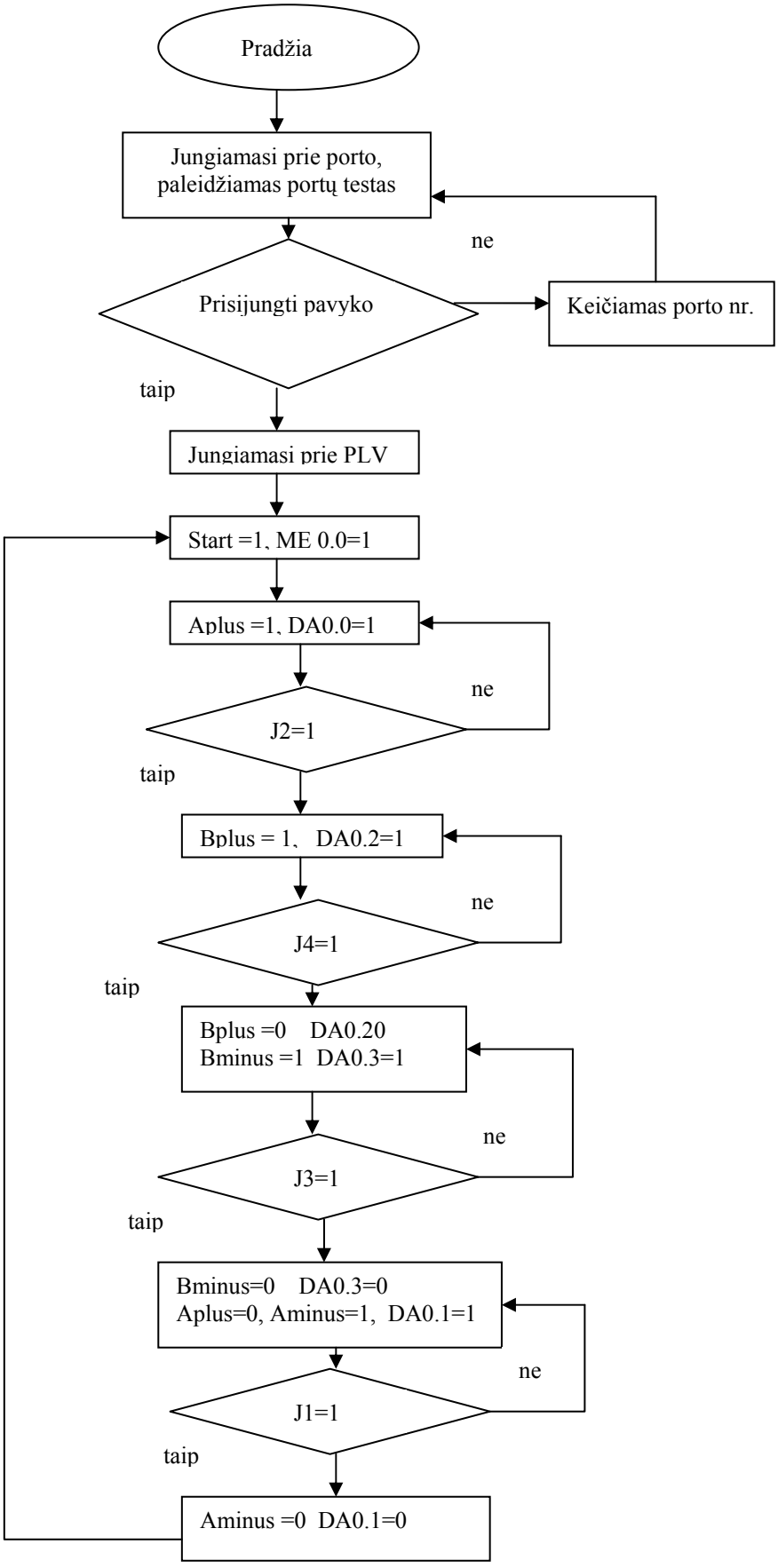


29 pav. Tiriamo modelio judesio diagrama.

Aplus = Start*J1*j3; kur: Aplus – pirmo cilindro stumoklio išstūmimas;
Aminus = J2*Minv*J3; Aminus – pirmo cilindro stumoklio grąžinimas;
Bplus = J2*J3; Bplus - antro cilindro stumoklio išstūmimas;
Bminus = J4*J2; Bminus - antro cilindro stumoklio grąžinimas;
M = J2*J4inv; M – papildomas programinės atminties elementas;

Programa STL kalboje gali būti sudaryta dviem būdais: lygiagrečiu kodu ir nuoseklaus vygdymo kodu. Antru atveju naudojami žingsniai, todėl nereikalingas atminties elementas. Pasirenkamas žingsninis programavimo būdas.

```
Step 1
IF Start AND J1 AND J3
 THEN SET Aplus
Step 2
IF J2 AND J3
THEN SET Bplus
Step 3
IF J4 AND J2
THEN RESET Bplus
     SET Bminus
     Step 4
IF J3 AND J2
THEN RESET Aplus
     SET Aminus
     Step 5
IF J1 AND J3
THEN RESET Aminus
JMP TO 1
```



30 pav. Sistemos darbo algoritmas

STL programos tekstas yra surenkama FST-4 redaktoriaus lange. Patikrinamos sintaksės ir loginės klaidos, jungiamasi prie PLV ir į jo operatyvinę atmintį užkraunamas programos kodas. Jei valdiklis yra prijungtas prie realios pneumatinės sistemos galima patikrinti, ar cilindrų stumokliai atlieka judesius pagal duotą laiko diagramą. FST-4 programa atjungiama nuo valdiklio ir prijungiama modeliavimo programa. Prisijungimą galima patikrinti naudojant porto tikrinimo testą iš pagrindinio vartotojo sąsajos lango. Paspaudus "Start" mygtuką siunčiamas signalas modifikuoti valdiklio įėjimo O0.0 būseną iš 0 į 1.Tuo pat metu valdiklis siunčia **A** stumoklio išstumimo signalą DA0.0=1, kuris įjungia modelio grafikos valdymą. Kai **A** stūmoklis pasiekia J2 jutiklį ir jis „suveikia", valdikliui siunčiamas signalas apie būsenos pokytį. PLV vėl siunčia į VB programą kodą B cilindro stūmoklio įjungimui. Tokiu būdu modelio grafika yra valdoma pagal į valdiklį užkrautą programą.

3.8. Programos tikrinimas ir derinimas

Programos derinimo etapas užima apie keturiasdešimt procentų laiko ir jo tikslas likviduoti programos klaidas ir netikslumus. Derinimas užtikrina, kad programinės įrangos sistemos atitiktų vartotojų poreikius. Programinės įrangos tikrinimas parodo ar teisingai kuriamas produktas . Derinimo metu yra nustatoma ar sukurta programa vykdo specifikacijose nustatytus reikalavimus. Taip pat yra atskleidžiamos sistemos klaidos ir vertinama ar sukurta programa yra darbinga. Tikrinimo metu sistema yra bandoma su testiniais duomenimis ir stebima jos darbinė elgsena. Sukurti testavimo programas kurios rodo visas klaidas yra neįmanoma. Programos tikrinimas gali būti statinis ir dinaminis. Statinis tikrinimas naudojamas visų projektavimo etapų kūrimo metu, o dinaminis dažniausiai tik tada, kai naudojami programų prototipai, nes tik esant veikiančiam programos moduliui jį galima išbandyti dinamiškai.

Nuo pasirinkto tikrinimo būdo priklauso ar bus nustatytos esamos klaidos. Jei tikrinant klaidų nerandama, tai dar nereiškia kad jų iš tikrųjų nėra. Tikrinimas ir testavimas turi parodyti, kad ši programinė įranga yra tinkama skirtam tikslui. Nustačius defektą reikėtų įvertinti jo daromą žalą. Kai kurios klaidos ir defektai gali būti aptikti, tačiau jei jie netrukdo normaliam programinės įrangos funkcionavimui ir bus pasiektas reikiamas tikrumo (pasitikėjimo) lygis tokia įranga laikoma tinkama. Jei defekto taisymui bus skiriami dideli kaštai, o defektuota sistema atlieka savo funkcijas, tokios klaidos gali būti paliktos. Dalis defektų gali būti aptinkama pradiniuose programos derinimo etapuose. Paprastai tęstavimo metu yra tik nustatoma ar yra defektų, o derinimo metu randama tiksli defekto vieta ir, jei tai įmanoma, jis taisomas.

Tikrinimo ir tęstavimo darbus būtina planuoti visam projekto periodui. Jei tokie darbai bus palikti galutiniam programos variantui, gali atsirasti situacija, kai vieni defektai slepia kitus. Taip

pat gali tekti keisti patį programos algoritmą. Planavimas padidina kaštus ankstyvame programinės įrangos kūrimo etape, tačiau mažina juos vėlesniuose. Klaidų požiūriu tikrinimo našumas yra nuo 50 iki 100 kodo eilučių per valandą. Tačiau tai subjektyvus lyginimas ir šitos ribos gali smarkiai kisti , priklausomai nuo programuotojo kvalifikacijos ir programavimo kalbos.

Klaidos pagal požymius gali būti skirstomos į klases. Skirstymo požymiai pateikti lentelėje 5.

Klaidų požymių lentelė

Lentelė 5

Klaidų klasė	Požymiai
Duomenų klaidos	Kintamieji panaudoti prieš inicializaciją. Kintamieji aprašyti bet niekada nepanaudoti. Kintamieji aprašyti dukart, bet nepanaudoti tarp aprašymų. Galimi masyvo ribų pažeidimai. Nedeklaruoti kintamieji
Kontrolės klaidos	Nepasiekiamas kodas. Nesąlyginės šakos į ciklus
Įvedimo/ išvedimo klaidos	Kintamieji išvedami dukart be tarpinio priskyrimo
Sąsajos klaidos	Parametrų tipų nesutapimai. Parametrų kiekio nesutapimai. Funkcijų rezultatų nepanaudojimas. Funkcijos ir proceūros be kreipinių.
Saugojimo valdymo klaidos	Žymekliai be ryšio. Žymeklių aritmetika.

Kuriant programinę įrangą vis plačiau naudojamas bedefektinis būdas. Jis remiasi ne defektų aptikimo ir šalinimo, o bedefektine filosofija (14). Ją sudaro:

- nuoseklus programinės įrangos kūrimas;
- formali specifikacija;
- statinis tikrinimas, naudojant korektiškumo argumentus;
- statistinis testavimas, nustatant programos patikimumą;

Bedefektinės įrangos kūrimui yra sudaromos atskiros grupės:

1. Specifikacijų grupė. Atsakinga už sistemos specifikacijos kūrimą ir palaikymą

2. Kūrimo grupė. Atsakinga už programinės įrangos kūrimą ir tikrinimą. Programinė įranga nėra vykdoma ir net nekompiliuojama šio proceso metu.

3. Sertifikacijos grupė. Atsakinga už programinės įrangos statistinių testų kūrimą. Naudojamas patikimumo augimo modelis, norint nustatyti, kada patikimumas yra tinkamas.

Pagrindinis bedefektinio PĮ būdo trūkumas – reikalinga aukšta programuotojų kvalifikacija.

Projekto programinio paketo testavimo tikslas- išbandyti sistemos galimybes esant ribinėms situacijoms. Testavimui numatomas naudoti „Baltos dėžės“ principas. Ribiniai atvejai yra gaunami iš pačios programos. Taip yra nustatomi defektai, kurie pasireiškia dėl programos ir valdiklių programavimo paketų kodų neatitikimo. Taisant defektus programoje turi būti numatomi apribojimai, palydimi klaidų pranešimais. Testuojant sąsają yra patikrinama ar teisingai veikia manipuliatoriai, mygtukai, meniu pasirinkimų komandos, ar nėra konfliktų tarp atskirų programos modulių, ar jie gali veikti savarankiškai.

Projekte , naudojant Visual Basic programavimo kalbą yra numatomos situacijos, kai gali įvykti klaidos ir tų klaidų apdorojimo operacijos. Atidarant dokumentines bylas, gali atsitikti taip , kad pasikeitė vardas, katalogas ir t. t. Naudojamas klaidos aptikimo operatorius *On Errorr GoTo* . Sudaroma paprogramė , kuri išveda pranešimą vartotojui arba nukreipia į kitą programos modulį. Taip pat klaidų aptikimui panaudoti operatoriai *Resume, Resume Next*, *Err.Raise Number* ir kitos priemonės. Programavimo metu naudojamasi metodika:

- ✓ naudoti metodus, kurie leidžia aptikti kodo defektus;
- ✓ pranešimus apie klaidas pageidautina iškviesti iš atskiros bylos;
- ✓ derinimo metu klaida turi pertraukti programos kodo darbą;

Derinant programinę įrangą buvo išaiškinti atvejai, kai sistemai nepavyksta prisijungti prie nuoseklios sąsajos, tačiau pranešimo apie klaidą nebuvo ir vartotojas negaudavo informacijos, kodėl modelis neveikia. Taip pat nustatyta stuacija, kai vienu metu prie to paties porto bando prisijungti modeliavimo programa ir FST-4 programa. Susidaro resursų panaudojimo konfliktas ir gaunamas operacinės sistemos pranešimas apie klaidą. Programiškai šios klaidos ištaisyti nepavyko, todėl apie tai būtina vartotoją informuoti sudarant dokumentaciją. Kita esminė klaida vyksta modifikuojant PLV įėjimus. Perduodant komandą MEW.B=x , įėjimo būsena pasikeičia tik vienam PLV ciklui. Valdiklis gali nespėti sureaguoti ir programos darbas sutrinka. Sudarant modelius reikėtų numatyti daugkartinį šios komandos perdavimą arba operuoti su valdiklio registrais ar vėliavėlėmis. Tada programinė STL eilutė būtų tokia:

IF J1 **OR R1** THEN SET Aplus arba IF J1 THEN **SET R1**; **IF R1** THEN SET Aplus;

kur R1 yra registro būsena.

Aišku šis pakeitimas keičia programavimo algoritmą ir jis tinka tik derinimo etapuose.

4. VARTOTOJO DOKUMENTACIJA

4.1. Funkcinis aprašymas

Programinė įranga sąlyginiu pavadinimu „MECHAS“ yra skirta mechatronikos disciplinas studijuojantiems studentams, kvalifikacijos kėlimo kursų klausytojams. Ji leidžia gilinti teorines ir praktines žinias, susipažinti su valdiklių programavimo paketais. Programą sudaro keli moduliai, kurie atlieka šias funkcijas:

a) pateikia darbo su FESTO, OMRON, MITSUBISHI valdikliais programiniais paketais metodiką;
b) pateikia pneumatinių, elektropneumatinių sistemų modeliavimo ir PLV programavimo praktines užduotis;
c) leidžia patikrinti studentų žinias testo pagalba;
d) iškviečia pagal nuorodas mechatronikos internetinį tinklapį, esantį Panevėžio kolegijos serveryje, bei pateikia nuorodas į kitus gimininguss tinklapius;
e) užkrauna ir paleidžia pneumatikos ir elektropneumatikos modeliavimo programą FluidSim;
f) pateikia elektropneumatinius modelius, kuriuos galima valdyti FST-4 programinės įrangos pagalba;
g) leidžia vizualizuoti procesus , kuriuos valdo FESTO FST-20, FST – 34, FST - 160 valdikliai;
h) Suteikia galimybę derinti STL kalba parašytas programas.

Šios funkcijos įgalina taikyti ši programinį produktą ne tik mokymo tikslams bet ir tiriamiesiems darbams, pvz. norint dekoduoti informaciją, kompiuterio siunčiamą ir priimamą per nuoseklią sąsają, kurti vizualizavimo programas kitų tipų valdikliams.

4.2. Sistemos vadovas

Mechatroninių sistemų modeliavimo programa, priklausomai nuo jos varianto, paleidžiama iš kataloge „ MECHAS“ esančios .exe bylos arba, jei buvo naudojamas instaliacinis paketas, per „Start – Programs – Mechas“ nuorodą. Pradiniame lange pateikiama informacija apie programos paskirtį. Paspaudus mygtuką „Testi“ pereinama į modulių pasirinkimo langą. Jame galimi šie pasirinkimai: internetinis puslapis, valdiklių programavimo paketai, pneumatinių sistemų modeliavimas, mechatroninių sistemų modeliavimas. Pirmu atveju, jei yra interneto ryšys, pagal

nuorodą patenkama į Panevėžio kolegijos serveryje esančia svetainę su teorine ir praktine mechatroninių sistemų medžiaga. Nesant ryšio, bus gautas atitinkamas klaidos pranešimas. Pasirinkus valdiklių programavimo paketus , toliau jie tikslinami pagal firmas – gamintojas ir atidaromi dokumentai Word'o formatu. Čia galima susipažinti su šių paketų programavimo metodika. Jei atsiranda pranešimas, kad dokumento nėra, arba jo negalima atidaryti, galimas atvejis, kad programinė įranga įdiegta ne į **C** diską, arba nesukurtas katalogas. Plačiau apie tai rašoma sistemos instaliavimo dokumente. Iškvietus pneumatinių sistemų modeliavimą yra paleidžiama programiniame pakete esanti mokomoji programa FluidSim, leidžianti kurti ir derinti pneumatines ir elektropneumatines sistemas. Paskutinis pasirinkimo lange yra mechatroninių sistemų modeliavimas. Pereinama į dar vieną pasirinkimo langą, kuriame yra teorinė dalis, praktinės užduotys, žinių tikrinimo testas ir sistemų modeliavimas. Pirmi du variantai, panaudojant programų sąryšio technologijas, atidaro atitinkamus dokumentus. Čia pranešimai apie klaidas analogiški prieš tai minėtiems. Teste pateikiama 20 klausimų su atsakymų pasirinkimo variantais. Pasirinkus sistemų modeliavimą, patenkama į pagrindinį langą. Jame yra sistemos modelių meniu su veikiančiais modeliais. Šie modeliai gali veikti savarankiškai, valdomi rankiniu būdu, arba STL programos derinimo režimu. Tam tikslui prie kompiuterio turi būti pajungtas FESTO FST -20 arba FST – 34 valdiklis su įdiegta programa. Vartotojas turi žinoti, kiek cilindrų valdoma programoje ir pagal tai parinkti reikiamą modelį. Esant tinkamai programai ir teisingam prisijungimui prie nuoseklaus porto grafinio modelio cilindrų stūmoklių judesiai atitiks realios PLV valdomos sistemos judesių diagramą. Visais kitais atvejais yra pateikiami pranešimai apie klaidas.

Iš bet kurio pasirinkimo lango ar programos galima grįžti atgal arba baigti darbą.

4.3. Programos instaliavimas

Projekto programinė įranga yra pateikiama dviem variantais: atviru Visual Basic kodu su paleidžiamąja „Mechas.exe" byla ir instaliacinis paketas su Setup byla. Pirmu atveju tai katalogas „Mechas" kurį reikia įkelti tiesiai į C diską. Jei bus pasirinktas kitas diskas arba papildomas katalogas, netsidarys dokumentai į kuriuos yra sukurtos nuorodos. Sistemos administratorius turi galimybę VB formų kode pakeisti nuorodas. Programoje taip pat yra FluidSim katalogas, kuriame randasi pneumatinių sistemų modeliavimo paketas. Diegimo metu reikia įvertinti tai, kad programa užima 40 MB diske. Iš jų – 16 MB FluidSim, apie 20 MB dokumentai. Programos ryšiui su nuosekliais portais būtina įdiegti VB bibliotekų bylas: msvbvm60.dll ir mscomm32.ocx. šios bylos yra „Mechas" kataloge, tačiau patartina jas perašyti į Windows OS System katalogą.

Naudojant instaliacinį Setup paketą visi komponentai įkeliami automatiškai, tačiau programos kodas administratoriui tampa neprieinamas.

Reikalavimai techninei kompiuterio įrangai minimalūs:

Prosesorius - > 400 Mhz

Operacinė sistema - Win 9x arba Win 2000 (XP)

Operatyvinė atmintis - >128 MB

Vaizdo plokštė - > 16 MB

4.4. Sistemos administratoriaus vadovas

Programinė įranga modeliavimo režime turi dirbti kartu su programuojamu loginiu valdikliu. Programa pritaikyta darbui tik su STL programavimo kalba, kurią palaiko Beck Automation Office FST – 4 paketas. Ši programa yra parduodama kartu su FESTO FST -20, 34, 160 valdikliais ir instaliuojama į kompiuterį. DOS operacinėje sistemoje veikiantis paketas FST-FEC irgi turintis STL kalbą, netinka, nes skiriasi duomenų struktūra. Valdiklis prie nuoseklaus COM porto yra pajungiamas standartiniu kabeliu su DB – 9S jungtimi. Kitas kabelio galas baigiasi RJ – 45 jungtimi, kuri jungiama į valdiklį. Neturint standartinio firmos, valdiklio gamintojos, kabelio jį galima pasigaminti pagal PLV instrukcijoje pateikiamą schemą. Programoje numatyta naudoti COM1 arba COM2 sąsajas. Jos pasirenkamos ir tikrinamos programos lango „Programiniai modeliai“ meniu operatoriumi „Prievadai“. Patartina naudoti tik tą nuoseklią sąsają, su kuria stabiliai veikia FST-4 programinė įranga.

Programa yra skirta darbui su vienu valdikliu, turinčiu 12 įėjimų ir 8 išėjimus. Jei sistemos ar proceso valdymui to nepakanka, valdiklius galima jungti nuosekliai. Tačiau šiuo atveju teks keisti komandų interpretatoriaus nustatymus.

5. PROGRAMOS KOKYBĖS ĮVERTINIMAS

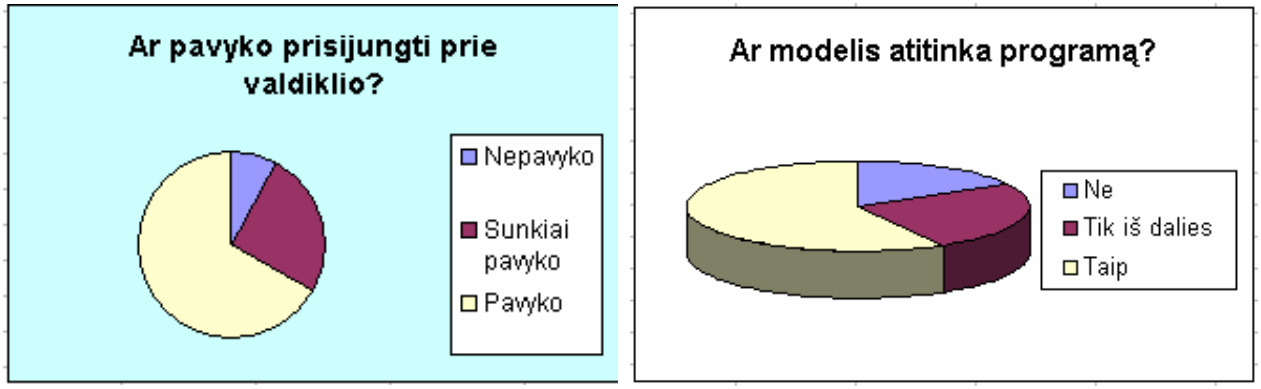
Programinės įrangos kokybei vertinti buvo atlikti tyrimai Panevėžio kolegijos programuojamųjų valdiklių laboratorijoje. Programa „Mechas“ buvo įdiegta į šešias kompiuterizuotas valdiklių programavimo darbo vietas. Kompiuteriuose taip pat instaliuota PLV programavimo įranga Beck Automation Ofice FST – 4. Buvo naudojami keturi FESTO FST – 20 ir du FESTO FST -34 valdikliai. Dvylikos studentų grupei pateiktos teorinės - praktinės užduotys, kurias reikėjo atlikti per nustatytą laiką ir apklausos anketos (priedas Nr.1). Visi studentai buvo susipažine su PLV programavimo kalbomis, tačiau pirmą kartą dirbo su tiriama programa. Laboratorijoje tirta programinės įrangos sistemų modeliavimo dalis. Programinės įrangos vertinimo užduotyse buvo tokie punktai:

1. Susipažinti su teorine medžiaga, pateikiama internetiniame tinklapyje.
2. Susipažinti su PLV programavimo paketų aprašymais.
3. Atlikti žinių patikrinimo testo užduotis.
4. Paleisti FluidSim programą ir sumodeliuoti elektropneumatinę schemą.
5. Sukurti programą STL kalba pagal pateiktą judesio diagramą.
6. Užkrauti programą į valdiklį, prijungtą prie pneumatinio stendo ir patikrinti stūmoklių judesių algoritmo teisingumą.
7. Prijungti tiriamos programinės įrangos dviejų cilindrų modelį prie valdiklio.
8. Patikrinti modelių grafikos veikimo atitikimą valdiklyje užkrautai programai.

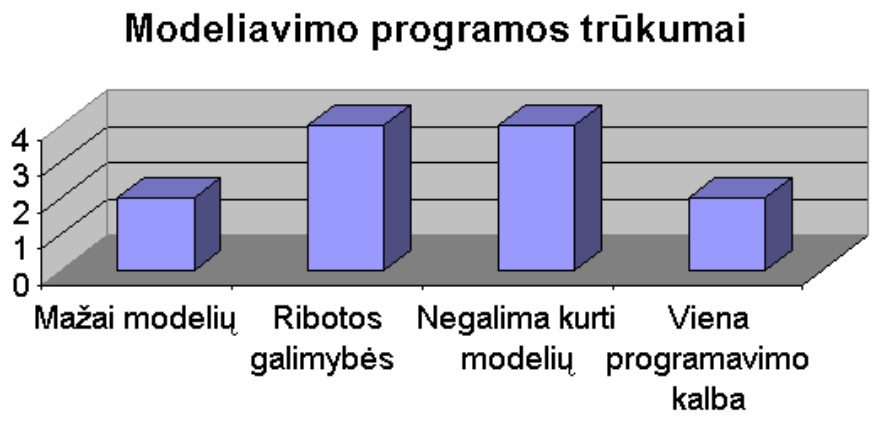
Pasibaigus nustatytam laikui, studentų, dirbusiu su programa, grupei pasiūlyta atsakyti į anketos klausimus. Anketa yra pateikta darbo prieduose. Galima apibendrinti ir pakomentuoti atsakymus į kai kuriuos klausimus.

Dauguma iš respondentų (8 iš 12) internetinio tinklapio medžiagą vertino gerai. 3 – patenkinamai ir 1 – labai gerai. 70% studentų tvirtino, kad programoje pateikiama metodinė medžiaga labai naudinga, likusiems – naudinga. Į klausimą dėl apklausos testo grafikos ir valdymo galimybių 50% studentų atsakė, kad tai juos tenkina, 20% - netenkina ir 30% - tenkina tik iš dalies. Galima daryti išvadą, kad šią programos funkciją dar reikia tobulinti. Tolesnio pokalbio metu studentai reiškė pageidavimus panaudoti teste laiko apribojimus ir įvesti savikontrolės režimą, kai rodomi teisingi atsakymai.

Visi apklausiamieji suprato, kaip pasirenkami modeliai, ir kaip jie veikia. Kiti programinės įrangos kokybės tyrimo apklausos būdu rezultatai yra pateikiami grafinėse diagramose.



31 pav. Programos tyrimo apklausos būdu rezultatai



32 pav. Studentų nuomonė apie pagrindinius programos trūkumus

Kaip matome iš apklausos rezultatų, programa turi gana daug trūkumų, kuriuos toliau vystant projektą būtina šalinti.

Atlikus projektuojamos programinės įrangos testavimą ir jos kokybės bandymus, galima daryti išvadą, jog sukurta programa atitinka funkcinius reikalavimus. Programos kokybę galima vertinti patenkinamai.

6. IŠVADOS

1. Kompiuterinis modeliavimas leidžia kurti ir optimizuoti sistemas bei procesus, nenaudojant brangių realių technologinių įrenginių.
2. Esamos sistemų ir procesų modeliavimo programos naudoja matematinius algoritmus arba funkcinius blokus ir rezultatas dažniausiai pateikiamas grafine forma.
3. Magistriniame darbe, panaudojus programų inžinerijos metodus , apibrėžti funkciniai, nefunkciniai ir srities reikalavimai kuriamai sistemai.
4. Projektavimo metu sukurta metodika PLV ir programinės įrangos tarpusavio sąveikos kodų analizei ir dešifravimui.
5. Darbe pateikiamų modelių pagalba galima tikrinti ir derinti STL kalba parašytas valdiklių programas.
6. Tyrimo metu sukurti ir patikrinti metodai gali būti panaudoti kitų firmų valdiklių programų kodų analizei.
7. Visual Basic programavimo kalba gali būti sėkmingai taikoma kuriant technologinių procesų vizualizavimo ir modeliavimo paketus.

7. LITERATŪRA

1. Deksnys V, Jamstramskas V. Įterptinės sistemos. Kaunas, Technologija, 2000.
2. Balžekas K, Rimkevičius V. Procesų loginio valdymo sistemų analizė ir sintezė. Kaunas, Technologija.2003.
3. Butkienė S, Simutis R, Tekorius T. Procesų modeliavimas. Kaunas, Technologija, 2002.
4. Starkus B. Visual Basic 6 jūsų kompiuteryje. Kaunas. Smaltija, 2000.
5. Aleksa V. Vykdymo įtaisai.Kaunas, Technologija, 2001.
6. Adomavičius J., Dulinskienė T. ir kiti. Informatika 2. Uždavinių paruošimas sprendimui Matlab ir Mathcad aplinkoje. Kaunas, Technologija, 2003.
7. Ostreika A . Programavimo Visual Basic pagrindai. Kaunas, Technologija, 2003
8. Šulcas V.Visual Basic 6 gramatika. Kaunas, Smaltija, 2003.
9. Viržonis D. Omron valdiklių programavimas. Kaunas, Technologija, 2003.
10. Bliesener R., Ebel F., vert. į l.k. Geleževičius V. Programuojamieji loginiai valdikliai. Festo AG & Co Esslingen, 1995.
11. Mark F. Russo. Automating science and engineering laboratories with Visual Basic, 2002
12. Necsulescu D. Mechatronics. New Jersey, Prentice Hall, 2002.
13. Dorf R.C. Bishop R.H. Modern control systems. New York, 1998.
14. John W. Horch Practical quide to software quality management. Artech House. 1999.
15. Žarkov S. Profesionalnaja razrabotka i prodviženije program.Sank-Peterburg, BXV-Peterburg , 2002.
16. Sergejev V. Visual Basic 6.0. Sank-Peterburg, BXV-Peterburg, 2003.
17. Dinaminių sistemų modeliavimas ir simuliavimas. Interaktyvus, [žiur. 2003.09.12.] Prieiga per internetą www.vissim.com
18. Eduardas Vilkas. Nelengva aukštųjų technologijų plėtra . Iš žur. *Veidas* [interaktyvus] 2002.10.31 Nr. 44 [žiurėta 2003. 05.16]. prieiga per internetą : http:// www.veidas.lt/
19. Skaitmeninių klaviatūros kodų lentelė. [žiūrėta 2003.09.15] Prieiga per internetą: www.asciitable.com
20. Programinės įrangos kokybės vertinimo sistema.[žiur. 2002.12.10], www.analog.com/dsp
21. Mechatronikos disciplinų mokomoji medžiaga[žiur.2003.05.12], www.festo.lt
22. Virtuali pneumatinių sistemų laboratorija [žiur.2002.09.23] , www.festo.com
23. Mitsubichi valdiklių programinės įrangos aprašymas [žiur.2003.06.10], http://global.mitsubishielectric.com/
24. Procesu vizualizavimo interaktyvus tinklapis [žiur.2003.10.15], http://www.genlogic.com/index.html
25. Procesų valdymo mokomasis interaktyvus tinklapis, http://csd.newcastle.edu.au/control/

8. Summary

There are many computer modeling systems, which are widely used in the world. Mechatronic is one of the practical range for modeling. Systems as MATLAB, CENTAURUS, VisSim and other are well known. Virtual programms, as CDS (newcastle university) are used for modeling on the Internet. Unfortunately, these systems are quite complicated and expensive.

So, naturally there comes demand on more simple and cheaper systems. The main objective of this project is to create low complicated mechatronics models creating system, who may change real pneumatics equipments used for education.

There were evolved an idea to caught and to decode data, which is sent from computer serial port to programmer logical controller (PLC) and received from it. There was explored the opportunity to imitate PLC control signals by computer programs. Broadcasting program was created by using Visual Basic programming language. This program helps to decode any data, which is trasmited over COM port. This relay let us to create command interpreter , which gives to vizualizate pneumatics stand works for FST – 4 and FESTO FST – 34 controlers. Two or three cilindrs pneumatics systems are being modeled by the imitation of sensors work. This idea could be used at other types of PLC and programs.

There are also teorical and practical education materials in project. These materials are publicated as Word documents or as internet page. This Project was tested and it will be used at Panevezys college.

9. TERMINŲ IR SANTRUMPŲ ŽODYNAS

Pavadinimas	Paaiškinimas
Modelis	Analogas (pakaitalas) realiai egzistuojančių arba įsivaizduojamų objektų, procesų, reiškinių., tapatus pasirinktu struktūros lygmeniu arba pasirinktomis funkcijomis.
Modeliavimas	egzistuojančių arba kuriamų (projektuojamų) objektų tyrimas, naudojantis jų modeliais; ko nors iš plastiškos medžiagos formavimas.
VB	Visual Basic programavimo kalba. (Projekte naudojama VB 6.0 versija)
PLV	Programuojamas loginis valdiklis
STL	Tekstinė programuojamųjų loginių valdiklių programavimo kalba
FESTO	Pneumatininius, hidraulinius, elektroninius mechatroninių sistemų komponentus, PLV gaminanti ir parduodanti firma
FST - 4	Beck Automation Office firmos valdiklių programavimo programinis paketas.
COM	Personalinio kompiuterio nuosekli sąsaja išoriniams įrenginiams pajungti
EAPI	**(**Extended Applications Programmer's Interface). Programinės įrangos kūrimo ir valdymo priemonės.
OLE	(Object Linking and Embedding) Objektų surišimas ir įdiegimas naudojamas VB programose , iškviečiant ir valdant kitų programų objektus.

10. PRIEDAI

Programinės įrangos „Mechas“ kokybės tyrimo Priedas Nr1.

ANKETA

Ši apklausa yra vykdoma, tikslu nustatyti Jums pateiktos programos savybes ir kokybę. Atsakykite į žemiau pateikiamus klausimus, pažymėdami labiausiai Jūsų nuomone tinkamą atsakymą. Dėkojame už atsakymus.

1. Kaip vertinate interneto tinklapio dizainą, pateikiamos medžiagos kokybę ir suprantamumą? .
   Blogai;
   Patenkinamai;
   Gerai;
   Labai gerai
2. Ar Jums buvo naudinga PLV programavimo paketų metodinė medžiaga, pateikta programoje?
   Nenaudinga
   Mažai naudinga
   Naudinga
   Labai naudinga
3. Ar Jus tenkina žinių tikrinimo testo grafinis apipavidalinimas ir valdymo galimybės?
   Netenkina
   Tenkina iš dalies
   Pilnai tenkina
4. Ar Jūs supratote programinių modelių pasirinkimo eigą ir jų rankinio valdymo galimybes?
   Nesupratau
   Sunkiai supratau
   Supratau
5. Ar pavyko programinę įrangą pajungti prie valdiklio?
   Nepavyko
   Sunkiai pavyko
   Pavyko
6. Ar Jūsų sukurta PLV programa atitiko judesio diagramą, dirbant prie pneumatinio stendo?
   Ne
   Atitiko tik iš dalies
   Pilnai atitiko
7. Ar programos modelio judesys atitiko užprogramuotą judesio diagramą?
   Ne
   Tik iš dalies
   Taip
8. Koks didžiausias modeliavimo programos trūkumas?
   Mažas modelių pasirinkimas
   Ribotos modelių funkcinės galimybės
   Nėra galimybės patiems kurti modelius
   Naudojama tik viena programavimo kalba
9. Kaip vertinate programos sąsają, valdymo elementų funkcionalumą?
   Blogai
   Patenkinamai
   Gerai
10. Ar pasitaikė programos darbe klaidų, kurių nesupratote arba kurios turėjo įtakos darbo rezultatams?
    Ne
    Taip

COM portų testavimo programų kodai

Forma

```
 Sub Command1_Click()
   Dim i As Integer
   For i = 1 To 4
      If COMAvailable(i) Then
         MsgBox "COM " & i & "   GERAI  ", vbInformation
      Else
         MsgBox "COM " & i & "  NEPAJUNGTAS  ", vbExclamation
      End If
   Next
End Sub
Private Sub Command2_Click()
End
End Sub
```

Modulis

```
Option Explicit
'API nuorodos
Public Declare Function CreateFile Lib "kernel32.dll" Alias "CreateFileA" (ByVal lpFileName As String, ByVal dwDesiredAccess As Long, ByVal dwShareMode As Long, lpSecurityAttributes As SECURITY_ATTRIBUTES, ByVal dwCreationDisposition As Long, ByVal dwFlagsAndAttributes As Long, ByVal hTemplateFile As Long) As Long
Public Declare Function CloseHandle Lib "kernel32.dll" (ByVal hObject As Long) As Long

'API struktura
Public Type SECURITY_ATTRIBUTES
   nLength As Long
   lpSecurityDescriptor As Long
   bInheritHandle As Long
End Type
'API konstantos
Public Const FILE_SHARE_READ = &H1
Public Const FILE_SHARE_WRITE = &H2
Public Const OPEN_EXISTING = 3
Public Const FILE_ATTRIBUTE_NORMAL = &H80

'Grazina TRUE jeigu  COM pajungtas, FALSE jeigu COM nepajungtas
Public Function COMAvailable(COMNum As Integer) As Boolean
   Dim hCOM As Long
   Dim ret As Long
```

```
    Dim sec As SECURITY_ATTRIBUTES
    'Bandoma atidaryti  COM porta
    hCOM = CreateFile("COM" & COMNum & "", 0, FILE_SHARE_READ + FILE_SHARE_WRITE,
sec, OPEN_EXISTING, FILE_ATTRIBUTE_NORMAL, 0)
    If hCOM = -1 Then
        COMAvailable = False
    Else
        COMAvailable = True
        'Uzdaromas COM portas
        ret = CloseHandle(hCOM)
    End If
End Function
```

Priedas Nr.3.

Portų retransliatoriaus programos kodas

```
Private Sub Command1_Click()
MSComm1.CommPort = Text1.Text
End Sub
Private Sub Command10_Click()
MSComm1.CommPort = Text1.Text
MSComm1.PortOpen = False
Shape15.BackColor = RGB(255, 0, 0)
End Sub
Private Sub Command11_Click()
MSComm1.CommPort = Text1.Text
MSComm1.PortOpen = True
Shape15.BackColor = RGB(0, 255, 0)
End Sub
Private Sub Command15_Click()
List1.Clear
Text3.Text = List1.ListCount
End Sub
Private Sub Command2_Click()
Text6.Text = ""
List1.Clear
Text3.Text = List1.ListCount
End Sub

Private Sub Command23_Click()
MSComm2.CommPort = Text8.Text
```

```
End Sub
Private Sub Command25_Click()
MSComm2.CommPort = Text1.Text
MSComm2.PortOpen = False
Shape16.BackColor = RGB(255, 0, 0)
End Sub

Private Sub Command26_Click()
MSComm2.CommPort = Text8.Text
MSComm2.PortOpen = True
Shape16.BackColor = RGB(0, 255, 0)
End Sub

Private Sub Command29_Click()
List2.Clear
Text7.Text = ""
End Sub
Private Sub Command3_Click()
MSComm1.Output = Text4.Text
End Sub

Private Sub Command7_Click()
MSComm1.Output = Chr$(Asc(Text2.Text))
End Sub

Private Sub Command9_Click()
If MSComm1.PortOpen Then MSComm1.PortOpen = False
If MSComm1.PortOpen Then MSComm2.PortOpen = False
End
End Sub
  Private Sub Form_Load()
    Form1.Caption = "App2"
     With MSComm1
        .CommPort = 1
        .Handshaking = 2 - comRTS
       .RThreshold = 1
       .RTSEnable = True
       .Settings = "9600,n,8,1"
       .SThreshold = 1
      '.PortOpen = True
       .EOFEnable = True
    End With
```

```
    With MSComm2
     .CommPort = 2
     .Handshaking = 2 - comRTS
    .RThreshold = 1
    .RTSEnable = True
    .Settings = "9600,n,8,1"
    .SThreshold = 1
    '.PortOpen = True
    .EOFEnable = True
  End With
End Sub

Private Sub Form_Unload(Cancel As Integer)
  MSComm1.PortOpen = False
End Sub

Private Sub gaunu2(Tekstas As String)
MSComm1.Output = Tekstas 'siunciame i COM1
For f = 1 To Len(Tekstas)
TK = Left(Tekstas, f)
simb = Right(TK, 1)
List2.AddItem simb & " - " & Asc(simb)
Next f
Text7.Text = Text7.Text & Tekstas
Text2.Text = Text2.Text & "*V*" & Tekstas
Text9.Text = List2.ListCount
End Sub
Private Sub Gaunu(Tekstas As String)
Dim f As Integer
Dim TK As String
Dim simb As String
MSComm2.Output = Tekstas 'siunciame i COM2
For f = 1 To Len(Tekstas)
TK = Left(Tekstas, f)
simb = Right(TK, 1)
List1.AddItem simb & " - " & Asc(simb)
Next f
Text6.Text = Text6.Text & Tekstas
Text2.Text = Text2.Text & "*P*" & Tekstas
Text3.Text = List1.ListCount
End Sub
```

```
Private Sub MSComm2_OnComm()
Select Case MSComm2.CommEvent
   Case comEvReceive ' Received RThreshold # of chars.
            gaunu2 (MSComm2.Input)
   End Select
End Sub
Private Sub MSComm1_OnComm()
Select Case MSComm1.CommEvent
      Case comEvReceive ' Received RThreshold # of chars.
            Gaunu (MSComm1.Input)
   End Select
End Sub
```

Priedas Nr.4

Komutatoriaus programos kodas

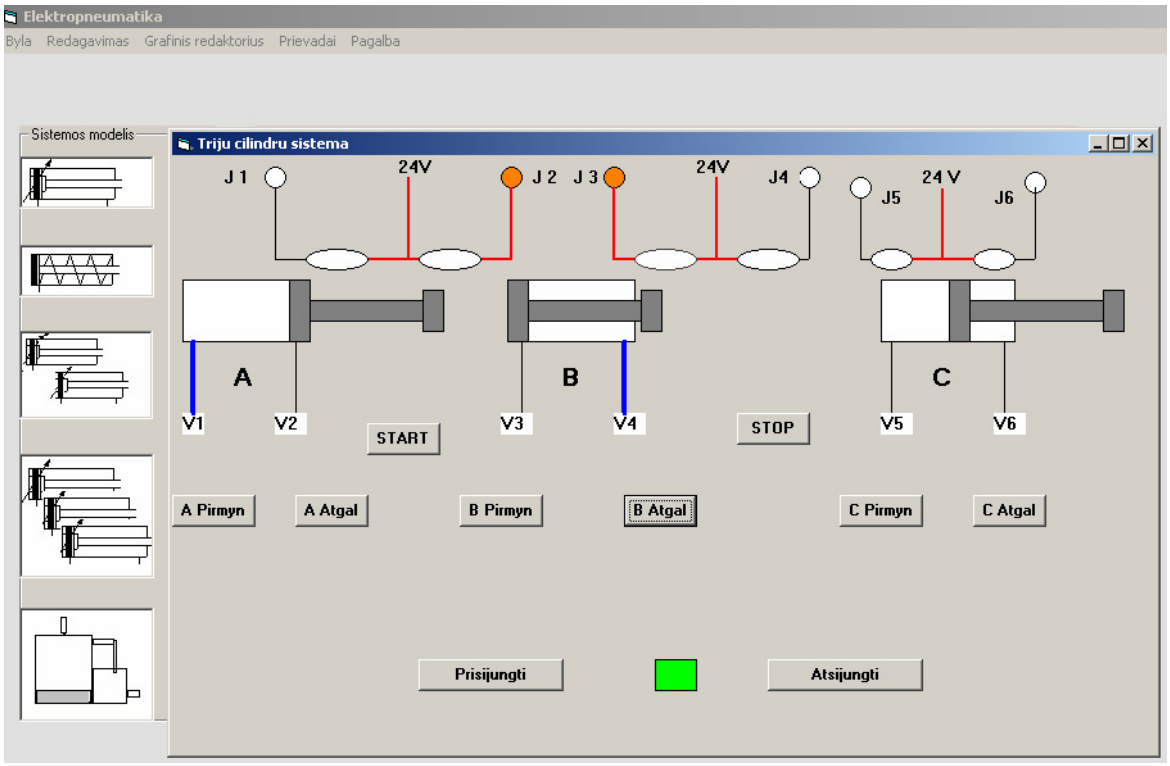
```
Dim Operacija, Eile As Byte
Dim Prisijungimas, Stebejimas As Boolean

Private Sub Apie_Click()
MsgBox "2003 (c) Programa skirta susisiekti su valdikliu"
End Sub

Private Sub Logoff()
Operacija = 2                    'Parenkamas operacijos Nr. skirtas skaitymui
MSComm1.Output = "x" & Chr$(13)     'Siunciamas signalas "atsijungti"
Shape3.BackColor = &HFF&            'Ijungiama indikacine lempute
End Sub

Private Sub Login()
Operacija = 1                    'Parenkamas operacijos Nr. skirtas skaitymui
MSComm1.Output = Chr$(20)          'Siunciama "control device 4" CI kalbos ijungimui
Shape3.BackColor = &HFF00&          'Ijungiama indikacine lempute
End Sub

Private Sub Command1_Click()
Login                         'Paleidziama prisijungimo procedura
End Sub

Private Sub Command2_Click()
Stebejimas = False                'Isjungiame stebejimo rezima
Command5.Caption = "-> Stebeti <-"  'Keiciamas mygtuko tekstas
```

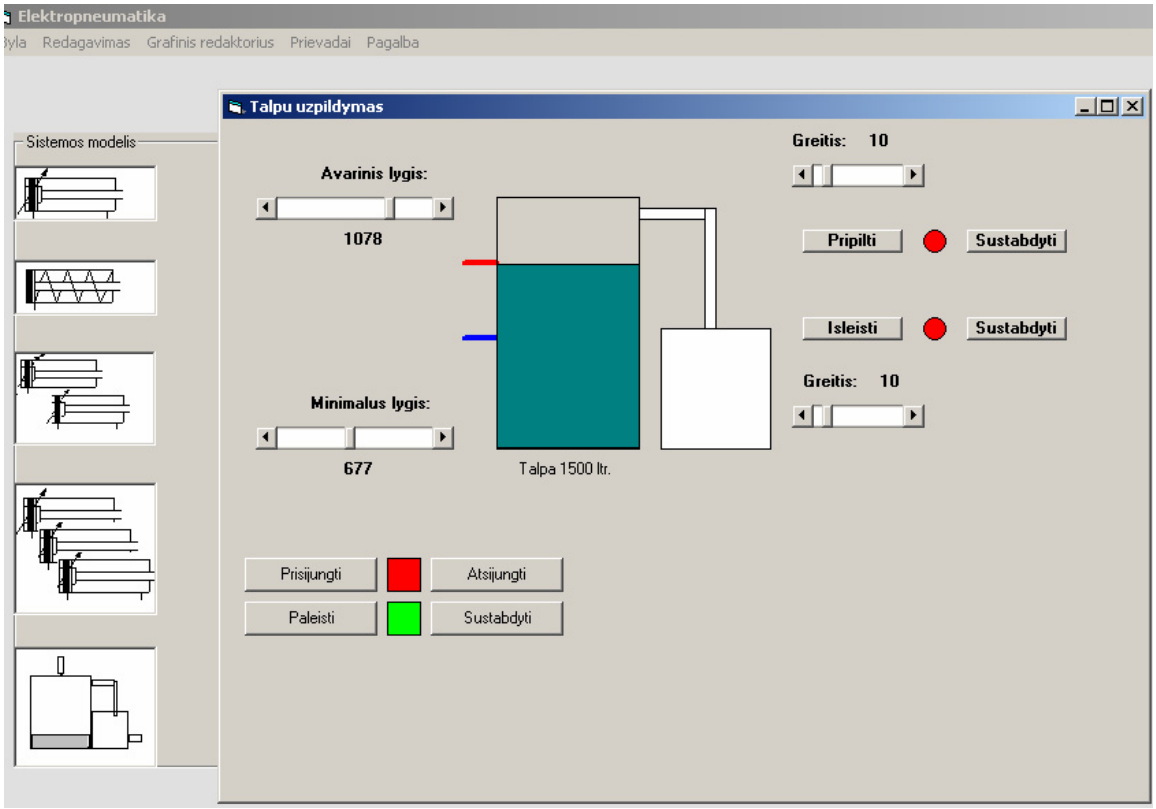
```
Shape4.BackColor = &HFF&          'Ijungiama indikacine lempute
Logoff                            'Paleidziama atsijungimo procedura
End Sub

Private Sub Command3_Click()
MSComm1.Output = "B" & Chr$(13)        'Pristabdo valdiklyje programa
End Sub

Private Sub Command4_Click()
If Prisijungimas Then                  'Jei prisijungta prie valdiklio...
Operacija = 3                          'Parenkamas operacijos Nr. skirtas skaitymui
MSComm1.Output = Text1.Text & Chr$(13)  'Issiusti vartotojo komanda
End If
End Sub

Private Sub Command5_Click()
If Stebejimas Then                     'Jei ijungtas stebejimas, tai...
Stebejimas = False                     'stebejimas isjungiamas;
Command5.Caption = "-> Stebeti <-"         'pakeiciamas mygtuko uzrasas
Shape4.BackColor = &HFF&                   'isjungiama indikacine lempute
Operacija = 0                          'Duomenu skaitymo operacija isjungiama
Else                                   'Jei neijungtas stebejimas, tai...
Stebejimas = True                      'stebejimas ijungiamas
Login                                  'prisijungiama prie valdiklio
Command5.Caption = "-> Stabdyti <-"        'pakeiciamas mygtuko uzrasas
Shape4.BackColor = &HFF00&                 'ijungiama indikacine lempute
End If
End Sub

Private Sub Command6_Click()
MSComm1.Output = "R" & Chr$(13) 'Paleidzia valdiklyje programa
End Sub

Private Sub Command7_Click()
MSComm1.Output = "S" & Chr$(13) 'Sustabdo valdiklyje programa
End Sub

Private Sub Form_Load()
Operacija = 0                  'Duomenu skaitymo operacija isjungiama
Prisijungimas = False          'Prie valdiklio neprisijungta
Stebejimas = False             'Isjungiamas automatinis stebejimas
Eile = 0                       'Parenkama eile taip, kad pirma teiraujamasi isejimu
```

```
 With MSComm1                'Su MSComm1 daryti
   .CommPort = 1             'Parenkamas COM1 portas
   .Handshaking = 2 - comRTS
   .RThreshold = 1
   .RTSEnable = True
   .Settings = "9600,n,8,1"    'Nustatomi porto parametrai
   .SThreshold = 1
   .PortOpen = True          'Ijungiamas portas
   .EOFEnable = True
 End With
End Sub

Private Sub Greitis()
Operacija = 4                    'Parenkamas operacijos Nr. skirtas skaitymui
MSComm1.Output = "MV=9600" & Chr$(13)   'Siunciama komanda greiciu suderinimui
End Sub

Private Sub Priimti(Tekstas As String)
Dim Ieskoti As String
Ieskoti = ">" & Chr$(17)        'Ieskoti bus komandos pabaigos teksto eilute
Select Case Operacija            'Priklausomai nuo operacijos Nr.
   Case 1                        'Prisijungimas prie valdiklio
   Prisijungimas = True              'Ijungiamas prisijungimas
   If InStr(Tekstas, Ieskoti) <> 0 Then Greitis    'Jei gaunamas pabaigos simbolis
   Case 2                        'Atsijungimas nuo valdiklio
   Prisijungimas = False             'Isjungiamas prisijungimas
 ' Case 3                        'Vartotojo operacija (nieko nedaryti)
   Case 4                        'Duomenu greitis
   If InStr(Tekstas, Ieskoti) <> 0 Then Operacija = 0
   Case 5                        'Uzklausos stebejimui
   If InStr(Tekstas, Ieskoti) = 0 Then stebetiF (Tekstas)
 ' Case Else                      'Kitais atvejais nieko nedaryti
End Select                      '*******
List1.AddItem Tekstas            'Visus gautus duomenis atvaizduoti sarase
End Sub

Private Sub MSComm1_OnComm()
   Dim InBuff As String
   Select Case MSComm1.CommEvent       'Is MSComm1 saraso...
 ' Galimos kai kurios klaidos
     Case comEventOverrun            'Jei duomenys prazuvo
         MsgBox "Klaida #01 Duomenys prazuvo"
```

```
    Case comEventRxOver             'Jei perpildytas priemimo buferis
        MsgBox "Klaida #02, Perpildytas priemimo buferis."
    Case comEventTxFull             'Jei pilnas siuntimo buferis
        MsgBox "Klaida #03, Pilnas siuntimo buferis."
' Galimi kai kurie ivykiai
    Case comEvReceive    'Jei buvo atsiustas koks nors signalas i porta, tai...
        Priimti (MSComm1.Input)     'Signala priimti
    End Select
End Sub

Private Sub UzgesintiLemputes()
Dim FN As Integer                'Papildomas kintamasis ciklui
For FN = 0 To 15                 'Nuo 0 iki 15 daryti...
If Eile = 0 Then                 'Jei buvo tikrinti isejimai, tai...
Shape1(FN).BackColor = &H808080          'Isjungti FN 1 eiles lempute
Else                             'Jei buvo tikrinti iejimai, tai...
Shape2(FN).BackColor = &H808080          'Isjungti FN 2 eiles lempute
End If                           '*******
Next FN                          '*******
End Sub

Private Sub stebetiF(EileS As String)
Dim Indik, FN As Long
Dim Tarp As String
Indik = Val(Right(EileS, 3)) 'Is tekstines eilutes duomenu pasiimame skaiciu
UzgesintiLemputes          'Uzgesina visas lemputes

If Eile = 0 Then             'Jei tikrinami buvo isejimai, tai...
Select Case Indik                'Priklausomai nuo desimtainio zodzio reiksmes
Case 1
Shape1(0).BackColor = &HFFFF&          'Ijungiama 1 lempute
Case 2
Shape1(1).BackColor = &HFFFF&          'Ijungiama 2 lempute
Case 3
Shape1(0).BackColor = &HFFFF&          'Ijungiama 1 lempute
Shape1(1).BackColor = &HFFFF&          'Ijungiama 2 lempute
End Select                       '*******
End If                     '*******

'Kol kas tikrinami tik 2 iejimai ir 2 isejimai

If Eile = 1 Then             'Jei tikrinami buvo iejimai, tai...
```

```
Select Case Indik                'Priklausomai nuo desimtainio zodzio reiksmes
Case 1
Shape2(0).BackColor = &HFFFF&          'Ijungiama 1 lempute
Case 2
Shape2(1).BackColor = &HFFFF&          'Ijungiama 2 lempute
Case 3
Shape2(0).BackColor = &HFFFF&          'Ijungiama 1 lempute
Shape2(1).BackColor = &HFFFF&          'Ijungiama 2 lempute
End Select                       '*******
End If                     '*******
End Sub

Private Sub Timer1_Timer()
If Stebejimas = True Then              'Jeigu ijungtas stebejimas, tai...
If Eile = 0 Then                       'Jei reikia tikrinti isejimus O
MSComm1.Output = "DAW0" & Chr$(13)           'Tikrinti isejimus I
Eile = 1                               'Toliau tikrinti iejimus I
Else                                 'Jei reikia tikrinti iejimus I
MSComm1.Output = "DEW0" & Chr$(13)           'Tikrinti iejimus I
Eile = 0                               'Toliau tikrinti isejimus O
End If                                '*******
Operacija = 5                          'Stebejimo operacijos NR. 5
End If                           '*******
End Sub

Private Sub Valymas_Click()
List1.Clear                      'Isvalomas sarasas
End Sub

Private Sub Pabaiga_Click()
MSComm1.PortOpen = False               'Uzdaromas portas
End
End Sub
```


Elektropneumatika
Byla Redagavimas Grafinis redaktorius Prievadai Pagalba
Sistemos modelis
Triju cilindru sistema
J 1
24V
J 2
J 3
24V
J4
J5
24 V
J6
A
B
C
V1
V2
V3
V4
V5
V6
START
STOP
A Pirmyn
A Atgal
B Pirmyn
B Atgal
C Pirmyn
C Atgal
Prisijungti
Atsijungti


Elektropneumatika
Byla Redagavimas Grafinis redaktorius Prievadai Pagalba
Sistemos modelis
Talpu uzpildymas
Greitis: 10
Avarinis lygis:
1078
Pripilti
Sustabdyti
Isleisti
Sustabdyti
Greitis: 10
Minimalus lygis:
677
Talpa 1500 ltr.
Prisijungti
Atsijungti
Paleisti
Sustabdyti