

---

**Univerzitet u Sarajevu**  
**ELEKTROTEHNIČKI FAKULTET U SARAJEVU**

Mr Samim Konjicija, dipl. ing. el.

**PRAKTIKUM AUTOMATIKE**  
**I INFORMATIKE**

**(Skripta)**

Sarajevo, januar 2007. godine

---

---

# SADRŽAJ

|                                                   |           |
|---------------------------------------------------|-----------|
| <b>UVOD</b> .....                                 | <b>7</b>  |
| <b>1. Matlab/Simulink</b> .....                   | <b>9</b>  |
| 1.1. Uvod.....                                    | 9         |
| 1.2. Rad sa Matlabom.....                         | 10        |
| 1.3. Operatori.....                               | 10        |
| 1.4. Komande i funkcije.....                      | 11        |
| 1.5. Matrice.....                                 | 12        |
| 1.6. Stringovi.....                               | 15        |
| 1.7. Strukture.....                               | 15        |
| 1.8. Grafika.....                                 | 16        |
| 1.8.1. 2D grafikoni.....                          | 17        |
| 1.8.2. 3D grafikoni.....                          | 19        |
| 1.8.3. Prikaz više grafika na jednom prozoru..... | 22        |
| 1.9. Programiranje.....                           | 23        |
| 1.9.1. M-fajlovi.....                             | 23        |
| 1.9.2. Skripte.....                               | 23        |
| 1.9.3. Funkcije.....                              | 24        |
| 1.9.4. Kontrola toka izvršavanja m-fajla.....     | 25        |
| 1.9.5. Programiranje grafičkog interfejsa.....    | 28        |
| 1.10. Control System Toolbox.....                 | 30        |
| 1.10.1. Predstavljanje sistema.....               | 30        |
| 1.10.2. Analiza karakteristika sistema.....       | 32        |
| 1.10.3. SISO Design Tool.....                     | 34        |
| 1.11. Simulink.....                               | 36        |
| 1.11.1. Rad sa Simulinkom.....                    | 37        |
| 1.11.2. Podešavanje parametara.....               | 38        |
| <b>2. LabView</b> .....                           | <b>43</b> |
| 2.1. Uvod.....                                    | 43        |
| 2.2. Korišćenje LabVIEW-a.....                    | 43        |
| 2.2.1. Virtualni instrument (VI).....             | 44        |
| 2.2.2. Dizajniranje virtualnog instrumenta.....   | 45        |
| 2.2.3. Osnovne programske strukture.....          | 48        |
| 2.2.4. Matematički izrazi.....                    | 50        |
| 2.2.5. Vremenske funkcije.....                    | 51        |
| 2.2.6. Nizovi.....                                | 52        |

---

|                                                           |           |
|-----------------------------------------------------------|-----------|
| 2.2.7. Grafici.....                                       | 52        |
| 2.2.8. Pohrana podataka.....                              | 53        |
| 2.2.9. Komunikacija sa hardverom.....                     | 54        |
| 2.2.10. Korištenje virtualnog instrumenta.....            | 55        |
| <b>3. Automatsko upravljanje.....</b>                     | <b>57</b> |
| 3.1. Uvod.....                                            | 57        |
| 3.2. Osnovna terminologija.....                           | 57        |
| 3.2.1. Zadana vrijednost.....                             | 58        |
| 3.2.2. Regulirana veličina.....                           | 58        |
| 3.2.3. Manipulativna veličina.....                        | 58        |
| 3.2.4. Objekat upravljanja.....                           | 59        |
| 3.3. Sistem.....                                          | 59        |
| 3.4. Upravljanje u otvorenoj i zatvorenoj sprezi.....     | 61        |
| 3.5. Osnovni pojmovi teorije automatskog upravljanja..... | 63        |
| 3.5.1. Regulirana veličina.....                           | 63        |
| 3.5.2. Manipulativna veličina.....                        | 64        |
| 3.5.3. Poremećaj.....                                     | 64        |
| 3.5.4. Zadana vrijednost.....                             | 64        |
| 3.5.5. Regulaciona greška.....                            | 65        |
| 3.5.6. Upravljački zakon.....                             | 65        |
| 3.5.7. Regulator.....                                     | 65        |
| 3.5.8. Izvršni organ i pogon izvršnog organa.....         | 66        |
| 3.5.9. Mjerni član.....                                   | 66        |
| 3.5.10. Regulaciona kontura.....                          | 66        |
| 3.6. Objekti upravljanja.....                             | 67        |
| 3.6.1. Vremenski odziv sistema.....                       | 67        |
| 3.6.2. Dinamičke karakteristike sistema.....              | 69        |
| 3.6.3. Statičke karakteristike sistema.....               | 70        |
| 3.7. Regulator.....                                       | 71        |
| 3.7.1. Tipovi regulatora.....                             | 72        |
| 3.7.2. Regulator sa proporcionalnim dejstvom.....         | 73        |
| 3.7.3. Regulator sa integralnim dejstvom.....             | 75        |
| 3.7.4. Regulator sa derivativnim dejstvom.....            | 76        |
| 3.7.5. PI regulator.....                                  | 78        |
| 3.7.6. PD regulator.....                                  | 79        |
| 3.7.7. PID regulator.....                                 | 80        |
| 3.7.8. Tehnička izvedba regulatora.....                   | 81        |
| <b>4. Komponente sistema automatskog upravljanja.....</b> | <b>83</b> |
| 4.1. Uvod.....                                            | 83        |
| 4.2. Optoelektronski mjerač protoka.....                  | 83        |
| 4.2.1. Princip rada.....                                  | 84        |

---

|                                                                                            |            |
|--------------------------------------------------------------------------------------------|------------|
| 4.2.2. Karakteristike i način priključivanja.....                                          | 84         |
| 4.3. Otpornički senzor temperature Pt 100.....                                             | 85         |
| 4.2.1. Princip rada.....                                                                   | 85         |
| 4.4. Ultrazvučni mjerač udaljenosti.....                                                   | 86         |
| <b>5. Uputstva za laboratorijsku opremu.....</b>                                           | <b>87</b>  |
| 5.1. Uvod.....                                                                             | 87         |
| 5.2. Bürkert 1110 Digital Industrial Controller.....                                       | 87         |
| 5.2.1. Struktura sa negativnom povratnom spregom i konstantnom<br>zadanom vrijednošću..... | 90         |
| 5.2.2. Struktura sa dodatnom regulacijom u otvorenom.....                                  | 92         |
| 5.2.3. Slijedna regulacija.....                                                            | 92         |
| 5.2.4. Regulacija odnosa.....                                                              | 95         |
| 5.2.5. Kaskadna regulacija.....                                                            | 95         |
| 5.3. Meilhaus Electronic 1208-FS modul za akviziciju podataka.....                         | 102        |
| 5.3.1. Načini prikupljanja vrijednosti analognih signala.....                              | 103        |
| 5.3.2. Povezivanje modula.....                                                             | 104        |
| 5.4. Moeller Easy 512-DC-RC Programmable Logic Controller.....                             | 107        |
| 5.4.1. Povezivanje modula.....                                                             | 110        |
| 5.4.2. Rad sa PLC-om i programiranje.....                                                  | 111        |
| 5.4.3. Razvojni alat EASY-SOFT-BASIC.....                                                  | 113        |
| <b>L i t e r a t u r a.....</b>                                                            | <b>119</b> |

---

---

## UVOD

Okruženje u kome živi moderni čovjek je ispunjeno različitim uređajima i postrojenjima koja više ili manje autonomno ispunjavaju funkcije za koje su sagrađena. U domaćinstvu i uopće oblasti stanovanja je veliki broj takvih uređaja i postrojenja, od jednostavnih (pegla, bojler, frižider, mašine za pranje veša i posuđa i sl.), preko velikog broja uređaja potrošačke elektronike (televizor, DVD, video i sl.), sve do kompleksnih klima uređaja, liftova i cjelokupnih sistema za upravljanje zgradama. U industriji su automatska postrojenja još i važnija, jer je bez njih odvijanje složenih procesa gotovo nemoguće. Samo neki primjeri takvih postrojenja su različiti parni kotlovi, turbine, generatori, proizvodne mašine, sve do cjelokupnih sistema kao što su elektroenergetski, sistem za distribuciju gasa, toplote i vode. Osnovna zajednička karakteristika svih pomenutih uređaja i postrojenja je postojanje jedne ili više komponenti koje omogućavaju automatsko odvijanje procesa ili obavljanje funkcija. Te komponente su sistemi automatskog upravljanja.

Teorijski osnovi sistema automatskog upravljanja se izučavaju u nizu predmeta redovnog studija. Pri tome je pristup analizi i sintezi sistema automatskog upravljanja apstrahovan od tehnologije njihove realizacije, najčešće iskazan uz korištenje odgovarajućeg matematičkog aparata. Jedan dio problematike praktične primjene teorije automatskog upravljanja se pokriva u okviru laboratorijskih vježbi.

Kurs Praktikum automatike i informatike je osmišljen tako da pokrije eksperimentalnu realizaciju principa upravljanja koji se izučavaju u okviru teorijskih kurseva na Odsjeku za automatiku i elektroniku tokom prve i druge godine studija.

U okviru kursa Praktikum automatike i informatike studenti bi trebali da:

- ovladaju vještinama planiranja i provođenja eksperimenta sa metodološkog aspekta,

- 
- steknu praksu i ovladaju korištenjem elektroničkih i električnih mjernih instrumenata,
  - konsolidiraju znanja koja stiču na teorijskim kursevima i razviju vještinu primjene teorijskih znanja za rješavanje praktičnih problema,
  - eksperimentalno verificiraju principe i zakone koji se izučavaju na teorijskim kursevima,
  - ovladaju inženjerskim vještinama, CAD i simulacionim alatima.

Eksperiment predstavlja fundamentalni pristup poimanju sistema i objekata, kao osnovi inženjerskih vještina. Planiranje i priprema eksperimenta, te njegovo provođenje i analiza rezultata treba da pomognu da student organizira znanje na način koji će mu omogućiti da ga kasnije koristi u praksi, pri rješavanju realnih problema na koje nalazi.

Bilo kakav eksperimentalni rad je praktično nezamisliv bez odgovarajućih mjernih instrumenata, uređaja i softvera. Da bi se ova oprema koristila, neophodno je razviti vještinu snalaženja u korisničkim uputstvima, koja često znaju biti vrlo obimna.

Kursevi redovnog studija na Odsjeku za automatiku i elektroniku pružaju studentu uvid u teorijske aspekte automatskog upravljanja i omogućavaju mu da ovlada neophodnim formalnim aparatom za tretiranje problema upravljanja. U okviru ovog kursa, kroz praktičnu ilustraciju principa i metoda koji se obrađuju u teoriji, student treba da stekne samopouzdanje i shvati da teorijska podloga predstavlja moćan alat za rješavanje kompleksnih i ne uvijek jasno definiranih problema kakvi se susreću u praksi. Za rješavanje takvih problema upravo je razvijanje inženjerskih vještina, čemu bi trebao dati doprinos i ovaj kurs, od izuzetnog značaja.

U skladu sa postavljenim ciljem, skripta za Praktikum automatike i informatike bi trebala dati početnu informaciju o sadržaju kursa, kratak pregled osnovnih tema iz teorije automatskog upravljanja, kao i osnovne informacije za korištenje opreme i softvera koji će se upotrebljavati tokom trajanja kursa.



---

# 1. Matlab/Simulink

## 1.1. Uvod

Matlab predstavlja vrlo razvijen skup alata za računanje (matrice, kompleksni brojevi, simbolička matematika), vizualiziranje (2D i 3D), modeliranje, simulaciju i programiranje. Karakterizira ga izrazita modularnost, otvorenost i nadogradivost. Ime mu je izvedeno od *matrix laboratory*, čime je naglašeno da Matlab sa svim podacima operiše kao sa matricama. Samo osnovne funkcije su ugrađene u jezgro Matlab, a sve ostalo je realizirano uglavnom pomoću m-fajlova, koji predstavljaju source fajlove programa realiziranih u Matlabu. Jezgru Matlab se različite funkcionalnosti dodaju pomoću tzv. toolboxova, koji predstavljaju zasebne skupine funkcija (programa) namijenjenih za rješavanje problema iz neke oblasti (tako imamo npr. Optimization Toolbox, Neural Network Toolbox, Fuzzy Logic Toolbox, Control System Toolbox, Signal Processing Toolbox, Statistics Toolbox itd.).

Matlab je u osnovi interaktivna okolina, i komunikacija sa korisnikom se odvija unošenjem komandi na promptu u komandnom prozoru, premda se većina radnji može obaviti i putem grafičkog interfejsa (GUI). Osim ovakvog načina rada, omogućeno je i pisanje programa, koje Matlab onda interpretira i izvršava. Source fajlove Matlab (m-fajlove), koji predstavljaju ASCII fajlove, je moguće i kompajlirati i time ubrzati njihovo izvršavanje. Podržano je kompajliranje u interni izvršni kod, klasični izvršni kod u obliku biblioteka i aplikacija, te prevođenje m-fajlova u C i C++ kod. Takođe, Matlab je putem Matlab API funkcija moguće koristiti i iz C, Fortran ili programa u drugim jezicima.

Nova verzija Matlab se pojavljuje svakih par godina, uz dosta poboljšanja u odnosu na prethodne verzije.

---

## 1.2. Rad sa Matlabom

Nakon pokretanja Matlaba, otvara se komandni prozor. Osnovni vid komunikacije korisnika sa Matlabom je putem tastature i komandnog prompta. Korisnik na komandnom promptu može unijeti komandu ili izraz, koji će se izvršiti nakon pritiska na tipku `<Enter>`.

Svi podaci sa kojima Matlab operiše su matrice, a skalar je samo specijalan slučaj matrice dimenzija  $1 \times 1$ . Iako Matlab posjeduje 15 predefiniраниh tipova podataka, najčešće nije potrebno voditi računa o tipu podatka jer funkcije Matlaba automatski obavljaju neophodne konverzije. Neki od predefiniраниh tipova su *double* (osnovni tip), *single*, *int8*, *int16*, *int32*, *int64*, *uint8*, *uint16*, *uint32*, *uint64*, *char* i sl. Matrice prije upotrebe nije potrebno deklarirati i dimenzionirati. Takođe, nije potrebno deklarirati ni tip varijabli, a svaka varijabla se kreira kada joj se pridruži vrijednost. Znak za pridruživanje je `=`, a identifikator može biti sve što počinje slovom i ne sadrži razmake i specijalne znake, s tim da se velika i mala slova razlikuju. Napomenimo da Matlab može raditi i sa stringovima, a oni se ograničavaju apostrofom `'`. Postoji skup predefiniраниh varijabli Matlaba, od kojih su neke navedene u tabeli 1.1.

Dozvoljeno je pridruživanje drugih vrijednosti predefiniраниm varijablama i funkcijama, ali tada se njihovo podrazumijevano značenje prekriva i ovo se ne preporučuje jer može dovesti do neispravnog rada funkcija koje koriste ove varijable i funkcije.

## 1.3. Operatori

Osnovni operatori u Matlabu su `+`, `-`, `*`, `/`, `\`, `^`, `'`, `:`. Treba napomenuti da su svi operatori matricni, što je vrlo važno naročito za operatore `*`, `/`, `\`, `^`. Ukoliko nam je potrebno da koristimo operator kao array operator (element-element), onda prije operatora stavljamo tačku (tako npr. matricno množenje je `*`, a množenje elementa sa elementom je `.*`). U tabeli 1.2. je dat pregled matricnih i array operatora, uz

---

pojašnjenje svakog od njih (A, B i C su matrice).

Zagrade (i) mijenjaju redosljed primjene operatora.

| <b>Varijabla</b>      | <b>Vrijednost</b>                                                                                    |
|-----------------------|------------------------------------------------------------------------------------------------------|
| <i>ans</i>            | podrazumijevana varijabla Matlaba, ukoliko nije zadata pri pozivu                                    |
| <i>pi</i>             | 3.14159285358979...                                                                                  |
| <i>i</i> ili <i>j</i> | imaginarna jedinica, $\sqrt{-1}$                                                                     |
| <i>eps</i>            | preciznost floating point formata, $2^{-52}$                                                         |
| <i>Inf</i>            | beskonačno, vrijednost veća od opsega floating point formata, $\infty$                               |
| <i>NaN</i>            | nije broj, rezultat izraza koji nemaju definirano matematičko značenje, npr. $0/0$ , $Inf/Inf$ , ... |
| <i>realmax</i>        | najveći broj u floating point formatu, $(2-\text{eps}) 2^{1024}$                                     |
| <i>realmin</i>        | najmanji pozitivan broj u floating point formatu, $2^{-1022}$                                        |
| <i>intmax</i>         | najveći 32-bitni cijeli broj                                                                         |
| <i>intmin</i>         | najmanji 32-bitni cijeli broj                                                                        |

Tabela 1.1. Predefinirane varijable Matlaba

## 1.4. Komande i funkcije

Na promptu se, osim izraza, mogu zadati komande Matlaba ili se pozivaju funkcije (sa ili bez argumenata). Neke od osnovnih komandi i funkcija Matlaba su:

*help* [*ime funkcije ili toolbox-a*] – prikazuje help za funkciju, toolbox ili osnovni help

*lookfor* riječ – traži sve helpove koji sadrže riječ

*dir* ili *ls* – sadržaj aktivnog direktorija

*pwd* – aktivni direktorij

*cd* – promjena aktivnog direktorija (*cd ime\_dir*, *cd ..*, *cd puna\_staza*, *cd*)

*mkdir* – kreiranje direktorija

*rmdir* – uklanjanje direktorija

*delete ime\_fajla* – briše fajl

*type ime\_fajla* – prikazuje sadržaj ASCII fajla

---

*diary ime\_fajla/off* – otvara/zatvara fajl u koji se zapisuje cjelokupna sesija  
*load ime\_fajla [varijabla ili lista varijabli]* – učitava varijable iz *.txt* ili *.mat* fajla  
*save ime\_fajla [varijabla ili lista varijabli]* – snima varijable u *.txt* ili *.mat* fajl  
*who, whos* – prikazuje varijable koje postoje u workspace-u  
*clear [varijabla ili lista varijabli]* – briše jednu, više ili sve varijable koje postoje u workspace-u  
*which funkcija* – ispisuje punu stazu funkcije sa imenom *funkcija* koja će biti izvršena

Jasno je da osim navedenih, postoji još veliki broj korisnih funkcija i komandi, a pomoć za njihovo korištenje je moguće naći korištenjem *help* sistema Matlaba. Kao osnovno, *help* daje pregled toolboxa koji su instalirani, sa kratkim opisom svakog od njih, *help general* daje pregled osnovnih komandi, *help elfun* pregled elementarnih matematičkih funkcija, *help elmat* pregled elementarnih matricnih funkcija i sl.

Matlab podržava komandnu i funkcijsku sintaksu i u nekim slučajevima su obadvije dozvoljene. Tako npr. *cd ime\_direktorija* i *cd('ime\_direktorija')* imaju isto značenje. Mi ćemo u takvim slučajevima preferirati funkcijsku sintaksu.

## 1.5. Matrice

Već je pomenuto da je matrica osnovni način predstavljanja podataka u Matlabu. Podaci matrice se ograđuju sa [ i ]. Elementi reda matrice se razdvajaju razmakom (<Space>) ili zarezom, a redovi sa <Enter> ili sa ; . Matrice se pojavljuju kao sastavni dio izraza u Matlabu. Tako npr. za unos matrice imamo:

$$A=[1, 2, 3; 4, 5, 6; 7, 8, 9]$$

ili

$$A=[1 2 3  
4 5 6  
7 8 9]$$

Nakon pritiska na tipku <Enter> u gornjim primjerima će Matlab dati vrijednost varijable A:

| Matrični operator | Djelovanje                                                                                                                                                                               | Array operator | Djelovanje                                                                                                                                                                                                       |
|-------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| +                 | Sabiranje matrica element sa elementom, odnosno $C = A + B$ znači da je svaki $c_{ij} = a_{ij} + b_{ij}$<br>Uslov: Dimenzije matrica moraju biti identične                               |                |                                                                                                                                                                                                                  |
| -                 | Oduzimanje matrica element od elementa, odnosno $C = A - B$ znači da je svaki $c_{ij} = a_{ij} - b_{ij}$<br>Uslov: Dimenzije matrica moraju biti identične                               |                |                                                                                                                                                                                                                  |
| '                 | Transponovanje matrice, odnosno $A = B'$ znači da je svaki $a_{ij} = b_{ji}$                                                                                                             |                |                                                                                                                                                                                                                  |
| *                 | Matrično množenje<br>Uslov: Broj redova prvog operanda mora biti jednak broju kolona drugog operanda                                                                                     | .*             | Množenje svakog elementa matrice prvog sa odgovarajućim elementom matrice drugog operanda, odnosno $C = A .* B$ znači da je svaki $c_{ij} = a_{ij} * b_{ij}$<br>Uslov: Dimenzije matrica moraju biti identične   |
| /                 | Matrično dijeljenje s desna, odnosno $C = A / B = A * inv(B)$<br>(Množenje s desna matrice koja predstavlja prvi operand sa inverznom matricom koja predstavlja drugi operand)           | ./             | Dijeljenje svakog elementa matrice prvog sa odgovarajućim elementom matrice drugog operanda, odnosno $C = A ./ B$ znači da je svaki $c_{ij} = a_{ij} / b_{ij}$<br>Uslov: Dimenzije matrica moraju biti identične |
| \                 | Matrično dijeljenje s lijeva, odnosno $C = B \setminus A = inv(B) * A$<br>(Množenje s lijeva matrice koja predstavlja prvi operand sa inverznom matricom koja predstavlja drugi operand) |                |                                                                                                                                                                                                                  |
| ^                 | Stepenovanje matrice, odnosno matrično množenje matrice sa sobom:<br>$A ^ 3 = A * A * A$<br>Uslov: Matrica mora biti kvadratna                                                           | .^             | Stepenovanje svakog elementa matrice operanda, odnosno $B = A .^ 3$ znači da je svaki $b_{ij} = a_{ij} ^ 3$                                                                                                      |

Tabela 1.2. Osnovni operatori u Matlabu

---


$$A = \begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \\ 7 & 8 & 9 \end{bmatrix}$$

Postoji još jedna funkcija znaka `;`. Naime, ukoliko bilo koji red završimo sa `;` izostaće prikaz nakon pritiska na `<Enter>`. Dakle, u Matlabu nije nužno da se svaki red ili izraz završi sa znakom `;` nego samo ukoliko nije potreban ispis rezultata izraza.

Transponovanu matricu dobijamo primjenom operatora transponovanja `'`, kao `A'`.

Neke od osnovnih funkcija za rad sa matricama su:

- `det(A)` – vraća determinantu matrice `A`
- `inv(A)` – vraća inverznu matricu matrici `A`
- `sum(A)` – vraća vektor-red čiji su elementi sume kolona matrice `A`
- `diag(A)` – vraća vektor-kolonu koja sadrži elemente glavne dijagonale matrice
- `poly(A)` – vraća vektor-kolonu koja sadrži koeficijente karakterističnog polinoma matrice `A`, tj. `det(λE-A)`
- `size(A)` – vraća vektor red čiji su elementi dimenzije matrice `A`
- `size(A,1)`, `size(A,2)` – vraća broj redova, odnosno kolona matrice `A`
- `finite(A)` – vraća matricu koja sadrži samo konačne elemente matrice `A`
- `zeros(m,n)` – vraća matricu popunjenu nulama dimenzije `m x n`
- `ones(m,n)` – vraća matricu popunjenu jedinicama dimenzije `m x n`
- `eye(n)` – vraća jediničnu matricu dimenzija `m x n`
- `flipr(A)`, `flipud(A)` – obrće matricu `A` sa lijeva na desno, odnosno odozgo na dole
- `rot90(A)` – rotira matricu za  $90^\circ$  u smjeru suprotnom od kazaljke na satu

Ako želimo adresirati dio ili samo element matrice, koristimo subskripte. Tako npr. element  $i$ -tog reda i  $j$ -te kolone matrice `A` se označava sa `A(i,j)`. Takođe, sve elemente matrice je moguće adresirati jednim subskriptom, pri čemu se podrazumjeva da su svi redovi matrice poredani jedan iza drugoga, npr. `A(k)`. Operatorom `:` zadajemo opseg, tako npr. `1:10` znači sve cijele brojeve od 1-10. Moguće je zadati i korak, pa npr. `0:0.1:1` označava brojeve 0, 0.1, 0.2, ..., 0.9, 1. Korak može biti i negativan. Ovo možemo koristiti i kao indekse u matricama, tako npr. `A(1:k,j)` označava sve elemente matrice `A` koji se nalaze u  $j$ -toj koloni i

---

redovima 1 do  $k$ . Dozvoljeno je i korištenje koraka za indekse u matrici, ali tada korak mora biti cijeli broj. Ako koristimo samo operator  $:$  u indeksu matrice, onda on označava sve vrijednosti indeksa, tako npr.  $A(:,j)$  vraća  $j$ -tu vektor-kolonu matrice  $A$ . Kao indeks se može koristiti i riječ *end* i ona označava zadnju kolonu ili red.

Vrlo je važno znati da u Matlab podržava jedan vid generalizacije pojma matrice, u smislu da dimenzionalnost matrice može biti i veća od 2. Većina funkcija Matlaba prihvata i ovakve višedimenzionalne matrice. Tako npr.  $ones(3,3,4)$  vraća četiri matrice dimenzija  $3 \times 3$  popunjene jedinicama.

Takođe, osim pojma matrice koja je sastavljena od elemenata istog tipa, Matlab podržava i tzv. polje ćelija (*cell array*), koje pruža više slobode u pogledu tipa elemenata koje sadržava (pošto npr. jedna ćelija može sadržavati broj, druga string i sl.). Za definiranje polja ćelija se koriste zagrade  $\{ \}$ . Primjer jednog polja ćelija je:

$$A = \{3, 'tekst', ones(4,2)\}$$

## 1.6. Stringovi

Matlab posjeduje i veliki broj funkcija za rad sa stringovima. Neke od njih su:

*eval(string)* – evaluira string *string* i izvodi ga kao komandu Matlaba

*feval(string,x1,x2,...)* – evaluira string, pri čemu mu se prosljeđuju vrijednosti  $x1, x2, \dots$

*strcat(S1, S2, ..., Sn)* – spaja stringove  $S1, S2, \dots, Sn$  i vraća rezultirajući string

*strcmp(S1, S2)* – poredi stringove i vraća 1 ako su isti, a 0 ako nisu

*num2str(X)* – konvertuje matricu  $X$  u string

*str2num(S)* – konvertuje string  $S$  u broj

*int2str(X)* – zaokružuje elemente matrice  $X$  na cijele brojeve i konvertuje je u string

## 1.7. Strukture

Struktura predstavlja vrlo koristan tip podataka u Matlabu, koji omogućava da se uz jedan identifikator veže više polja, od kojih svako polje može sadržavati različit tip

---

podatka. Strukturi se pristupa preko identifikatora varijable, a poljima preko identifikatora varijable i identifikatora polja: *id\_varijable.id\_polja*.

Da bi se kreirala struktura, dovoljno je pridružiti vrijednost polju:

$$\textit{korisnik.ime} = \textit{'Ime i Prezime'}$$

Ovim će biti kreirana varijabla *korisnik* tipa strukture, sa poljem *ime*. Pridruživanjem vrijednosti novim poljima se strukturi dodaju polja:

$$\textit{korisnik.telefon} = 12345678$$

Matlab dozvoljava vrlo jednostavan rad i sa poljima struktura, korištenjem indeksa, pri čemu vrijedi sve što je rečeno u vezi sa matricama. Tako npr. varijabla *korisnik* može biti vektor-red struktura, i da bi se pristupilo vrijednosti polja odgovarajućeg elementa ovog vektor reda, potrebno je koristiti indeks, npr. *korisnik(23).ime*.

## **1.8. Grafika**

Matlab ima vrlo razvijene mogućnosti za vizualizaciju podataka i jednostavno prikazivanje 2D i 3D grafikona. Obradićemo jedan dio tih mogućnosti.

Grafikoni se prikazuju u zasebnom grafičkom (*figure*) prozoru. Nakon poziva neke od grafičkih funkcija se ovaj prozor automatski otvara i sve dalje grafičke funkcije se odnose na njega. Možemo otvoriti i novi prozor bilo putem menija bilo komandom *figure*. Ukoliko je otvoreno više grafičkih prozora, samo jedan od njih može biti aktivan (grafičke funkcije se odnose na njega). Da bi grafički prozor postao aktivan, potrebno je kliknuti na njega. Drugi način je korištenje redosljednog broja grafičkog prozora u pozivu funkcije *figure*, tako npr. *figure(3)* aktivira grafički prozor čiji je redosljedni broj 3.



---

Neke od osnovnih komandi za rukovanje prikazom u grafičkom prozoru su:

*bold on/off* – uključuje/isključuje zadržavanje prikaza  
*axis on/off/ auto/ square/ equal* – rukovanje prikazom osa  
*axis([xmin xmax ymin ymax zmin zmax])* – definira opsege osa  
*grid on/off* – uključuje/isključuje prikaz koordinatne mreže  
*title('tekst naslova')* – naslov grafikona  
*text(x,y,'tekst')* – prikazuje *tekst* na poziciji *x,y*  
*xlabel('oznaka x ose')* – oznaka *x* ose  
*ylabel('oznaka y ose')* – oznaka *y* ose  
*legend('1. grafika','2. grafika','3. grafika',0)* – ispisivanje legende na grafiku  
*subplot(m,n,k)* – omogućava crtanje više grafika na jednom grafičkom prozoru

Od verzije 5.3 se većina ovih operacija može uraditi i putem menija i pop-up menija u okviru grafičkog prozora, ali ove funkcije omogućavaju da se grafik formatira u okviru m-fajla.

### 1.8.1. 2D grafikoni

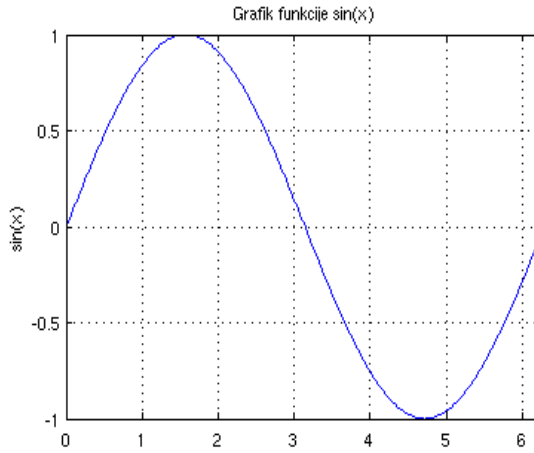
Za prikaz 2D grafikona se koristi funkcija *plot*. Tako npr. *plot(y)* prikazuje grafikon vrijednosti elemenata vektora *y* u odnosu na indeks vektora, a *plot(x,y)* prikazuje grafikon vrijednosti vektora *y* u odnosu na vrijednosti argumenta iz vektora *x*. Sljedeći niz komandi prikazuje dijagram funkcije sinus:

```
t=0:pi/100:2*pi;  
y=sin(t);  
plot(t,y)
```

Međutim, ovakav prikaz je najčešće potrebno dodatno urediti. U nastavku je naveden niz komandi koje uključuju prikaz koordinatne mreže, postavljaju prikaz u granice od 0 do  $2\pi$  po osi *x* i od -1 do 1 po osi *y*, ispisuju labele uz ose i postavljaju naslov na grafik:

```
grid on  
axis([0 2*pi -1 1])  
xlabel('x')  
ylabel('sin(x)')  
title('Grafik funkcije sin(x)')
```

Slika 1.1. prikazuje ovako uređen grafik.



Slika 1.1. Grafik funkcije jedne promjenljive

Da bi se odredila vrsta i boja linije grafika, koristi se stil definiran stringom koji se prosljeđuje kao treći argument pri pozivu funkcije: `plot(t,y,'r')`. Neki od često korištenih stilova su navedeni u tabeli 1.3.

| <b>Boja</b>   |                 | <b>Vrsta linije</b> |                                 |
|---------------|-----------------|---------------------|---------------------------------|
| <b>Simbol</b> | <b>Značenje</b> | <b>Simbol</b>       | <b>Značenje</b>                 |
| r             | crvena          | --                  | isprekidana                     |
| g             | zelena          | -.                  | crta-tačka                      |
| b             | plava           | :                   | tačke                           |
| c             | cijan           | .                   | diskretne vrijednosti tačkama   |
| m             | magenta         | o                   | diskretni vrijednosti kružićima |
| y             | žuta            | x                   | diskretne vrijednosti x-ovima   |
| k             | crna            | +                   | diskretne vrijednosti plusevima |

Tabela 1.3. Stilovi prikaza grafika

Moguće je jednim pozivom funkcije `plot` prikazati i više grafikona, koristeći sintaksu `plot(t1,y1,s1,t2,y2,s2,t3,y3,s3...)`.

---

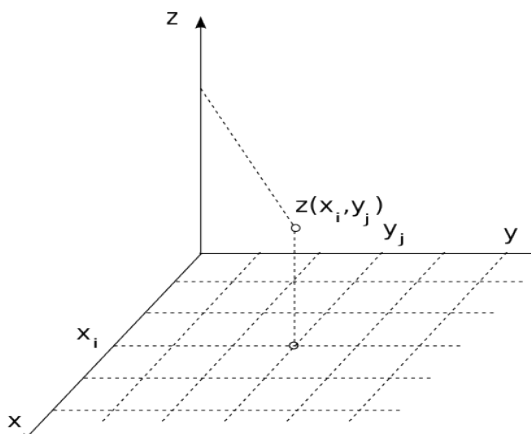
Treba napomenuti da svaki novi poziv funkcije *plot* briše prethodni grafik. Ukoliko je potrebno da se novi grafik doda na prethodne, potrebno je aktivirati zadržavanje grafika pozivom komande *hold on*. Zadržavanje se isključuje pozivom komande *hold off*.

Još neke važne funkcije za crtanje 2D grafika se mogu naći sa *help graph2d*.

### 1.8.2. 3D grafikoni

Matlab omogućava crtanje površi definiranih funkcijama dvije promjenljive, te crtanje parametarskih krivih. Da bismo prikazali grafik funkcije dvije promjenljive, potrebno je proći nekoliko koraka. Matlab definira površinu  $z=z(x,y)$  na mreži tačaka u  $x$ - $y$  ravni. Za crtanje 3D grafikona je potrebno prvo generisati tačke ravni  $x$ - $y$  (mrežu) kao matricu parova  $[X,Y]$ , gdje su  $X$  i  $Y$  matrice vrijednosti sa ose  $x$  i  $y$  čiji elementi predstavljaju koordinate tačaka ravni za koje će se crtati 3D grafikon (slika 1.2.). Da bismo generisali ovu matricu koristimo funkciju *meshgrid*:

$$[X,Y]=meshgrid(x,y);$$



Slika 1.2. Formiranje grafika funkcije dvije promjenljive

Nakon toga je potrebno izračunati vrijednosti funkcije  $Z=f(X,Y)$  za sve parove iz ovih matrica. Pri tome treba voditi računa o operatorima-potrebno je koristiti

---

operatore za polja, a ne matične operatore, npr.

$$Z=X.^2 + Y.^2;$$

Sada se pozove neka od funkcija za prikaz površi, npr. *mesh(X,Y,Z)*. Postoji niz funkcija za prikaz različitih vrsta 3D grafikona i sve one imaju vrlo sličnu sintaksu. Neke od njih su:

*mesh(X,Y,Z)*

*meshc(X,Y,Z)*

*meshl(X,Y,Z)*

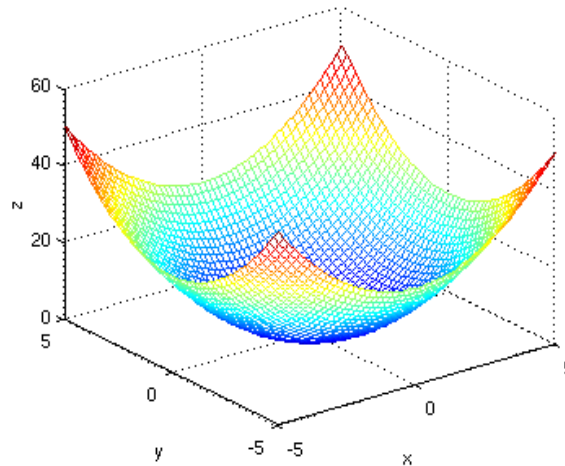
*surf(X,Y,Z)*

*surfc(X,Y,Z)*

*surfl(X,Y,Z)*

*contour(X,Y,Z,n)*

*contour3(X,Y,Z,n)*



Slika 1.3. Grafik funkcije  $z=x^2+y^2$

Osim grafika funkcija jedne i dvije promjenljive, u Matlabu je moguće crtati i parametarske krive. Za crtanje 3D parametarskih krivih se koristi funkcija *plot3*, a da bi se nacrtala neka parametarska kriva, neophodno je generisati odgovarajuće vektore  $x$ ,  $y$  i  $z$ , koji sadrže koordinate tačaka parametarske krive u 3D prostoru.

Kao primjer, nacrtajmo parametarsku krivu za koju se koordinata  $z$  mijenja od 0 do 1, a na tom opsegu vrijednosti tačke krive linearno pređu puni krug od ose  $x$  do ose  $x$ , na konstantnoj udaljenosti 2 od koordinatnog početka. Za slobodni parametar

---

uzmimo vrijednost  $z$  pa ćemo generisati vektor vrijednosti ove koordinate u traženom opsegu:

$$z = 0:0.05:1;$$

Ovu krivu je najjednostavnije opisati u cilindričnom koordinatnom sistemu, pa je poluprečnik konstantan, a ugao se mijenja od 0 (za  $z = 0$ ) do  $\pi$  (za  $z = 1$ ), odnosno :

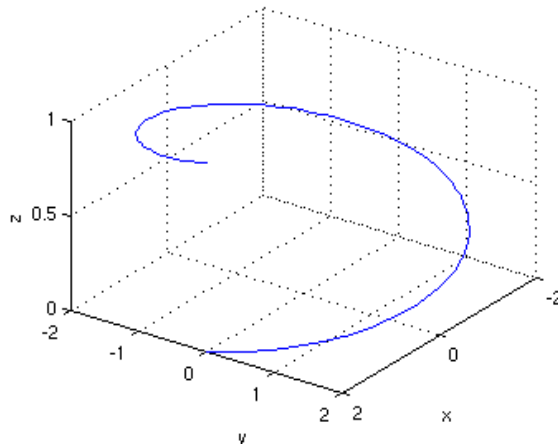
$$r = 2;$$
$$fi = 2 * pi * z;$$

Obzirom da grafičke funkcije Matlaba crtaju tačke čije su koordinate predstavljene u pravouglom koordinatnom sistemu, potrebno je izvršiti konverziju koordinata iz cilindričnog koordinatnog sistema:

$$x = r * cos(fi);$$
$$y = r * sin(fi);$$

Sada možemo nacrtati traženu parametarsku krivu (slika 1.4.):

$$plot3(x,y,z)$$



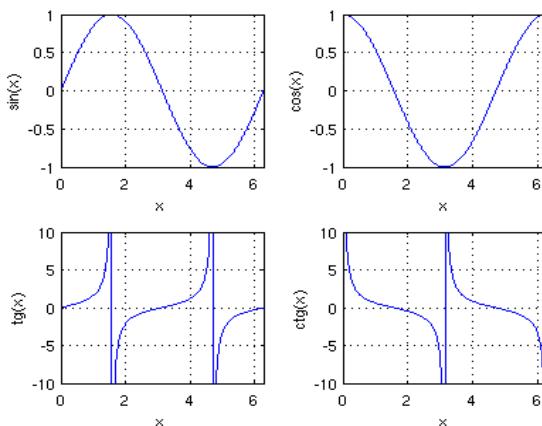
Slika 1.4. Grafik parametarske krive

---

### 1.8.3. Prikaz više grafika na jednom prozoru

Do sada je svaki grafik zauzimao cijeli grafički prozor. Međutim, Matlab omogućava da se na jednom grafičkom prozoru prikaže više grafika, koristeći funkciju *subplot*. Ovi grafici se organiziraju u redove i kolone, što se prosljeđuje funkciji *subplot* u okviru prva dva argumenta, dok treći argument predstavlja indeks grafika (gledano sa lijeva na desno i odozgo na dole) na koji će se odnositi sve grafičke funkcije koje slijede. Kao primjer, nacrtajmo na novom grafičkom prozoru četiri grafika, koji predstavljaju osnovne trigonometrijske funkcije u opsegu od 0 do  $2\pi$  (slika 1.5.):

```
figure
x = 0:0.01:2*pi;
y1=sin(x);
y2=cos(x);
y3=tan(x);
y4=tan(x).^(-1);
subplot(2,2,1),plot(x,y1), grid on, xlabel('x'), ylabel('sin(x)'), axis([0,2*pi,-1,1])
subplot(2,2,2),plot(x,y2), grid on, xlabel('x'), ylabel('cos(x)'), axis([0,2*pi,-1,1])
subplot(2,2,3),plot(x,y3), grid on, xlabel('x'), ylabel('tg(x)'), axis([0,2*pi,-10,10])
subplot(2,2,4),plot(x,y4), grid on, xlabel('x'), ylabel('ctg(x)'), axis([0,2*pi,-10,10])
```



Slika 1.5. Prikaz više grafika na jednom grafičkom prozoru

Još neke važne funkcije za crtanje 3D grafika se mogu naći sa *help graph3d*.

---

## **1.9. Programiranje**

Pored svega do sada navedenog, Matlab predstavlja i razvojno okruženje sa moćnim programskim jezikom. Matlab je u osnovi interpreter, ali posjeduje i dodatni kompajler (*mcc*) koji je u stanju kod pisan u Matlabu prevesti u interni izvršni kod, C/C++ kod, a uz korištenje vanjskog C++ kompajlera i u izvršni kod koji se može izvršavati nezavisno od Matlab okruženja.

### *1.9.1. M-fajlovi*

M-fajlovi su ASCII fajlovi (sa ekstenzijom *.m*) koji sadrže niz komandi i funkcija Matlaba. Ove funkcije i komande je moguće izvršiti pozivom imena m-fajla, nakon čega Matlab interpretira sadržaj fajla. Dakle, m-fajlovi su source fajlovi programa za Matlab. Da bi se mogao izvršiti kod u nekom m-fajlu, potrebno je da se taj m-fajl nalazi u aktivnom direktoriju Matlaba, ili u direktoriju koji je upisan u staze koje Matlab pretražuje.

Postoje dvije različite vrste m-fajlova, koje se bitno razlikuju po svojstvima, a to su skripte i funkcije.

### *1.9.2. Skripte*

Skripte predstavljaju m-fajlove koji sadrže samo niz naredbi Matlaba koje se izvršavaju pozivom imena fajla. Skripte ne prihvataju ulazne argumente i ne vraćaju izlazne vrijednosti. Koriste varijable iz workspacea Matlaba. Mogu stvarati nove varijable koje nakon završetka izvršavanja m-fajla ostaju dostupne u workspaceu. U okviru skripte je moguće realizirati sve što je moguće uraditi i sa komandnog prompta Matlaba.

---

### 1.9.3. Funkcije

Funkcije predstavljaju m-fajlove koji imaju posebno zaglavlje sa deklaracijom imena funkcije, ulaznih argumenata i vrijednosti koje funkcija vraća.

```
function izl_var=ime_funkcije(ul_var1,ul_var2,...)
```

gdje su:

*izl\_var* – varijabla koju funkcija vraća (ili više njih-matrica)

*ime\_funkcije* – ime funkcije koje odgovara imenu m-fajla

*ul\_var1, ul\_var2, ...* – argumenti koji se prenose u funkciju pri pozivu

Nakon deklaracije funkcije obično (ali ne obavezno) slijedi komentar, u kome se navodi opis funkcije, sintaksa, opis argumenata i vrijednosti koje funkcija vraća. Kada se na komandnom promptu pozove *help ime\_funkcije*, biće prikazan sadržaj ovog komentara. Ovo omogućava vrlo jednostavno dokumentiranje funkcija u Matlabu. U Matlabu je znak za komentar % i sve što slijedi nakon ovog znaka do kraja reda se pri izvršavanju koda zanemaruje.

Ime pod kojim je funkcija vidljiva izvana je ime m-fajla, bez obzira na ime navedeno u deklaraciji funkcije. U okviru m-fajla je moguće deklarirati i više funkcija, ali su sve osim osnovne (prve) vidljive i dostupne samo lokalno iz m-fajla u kome se nalaze. Tijelo funkcije se završava sa krajem fajla, ili sa početkom deklaracije naredne funkcije. Iz funkcije se izlazi nakon njenog završetka (kraj fajla ili deklaracija naredne funkcije) ili nakon nailaska na komandu *return*.

Matlab dozvoljava da se prilikom poziva funkciji ne proslijede svi argumenti, ali implementacija funkcije tada mora voditi računa o postavljanju podrazumijevanih vrijednosti nedostajućih argumenata, ukoliko je to neophodno. Ukoliko je potrebno pri pozivu funkcije preskočiti neki argument koji se neće koristiti, tada se koristi prazna matrica [].

Za određivanje koji ulazni argumenti su proslijeđeni pri pozivu funkcije se koriste funkcije *nargin* i *isempty*:



---

```
if (nargin == broj) | isempty(arg)
    postavka default vrijednosti argumenta arg
end
```

Isto tako, Matlab dozvoljava da se od funkcije ne preuzmu sve vrijednosti koje ona vraća. Ukoliko je broj varijabli koje preuzimaju ove vrijednosti manji, vraća se samo onoliko vrijednosti koliko ima varijabli, i to s lijeva na desno. Da bi se za takav slučaj uticalo na to koje vrijednosti će funkcija vratiti, može se koristiti funkcija *nargout*, o čemu se više detalja može naći u dokumentaciji.

Funkcija stvara svoj zasebni prostor varijabli lokalnog doseg a i trajanja, a iz prostora varijabli pozivajuće funkcije/workspacea se prenose samo ulazni argumenti, pri čemu su sve njihove promjene lokalnog karaktera. Funkcija može vratiti jedino vrijednosti varijabli koje su u deklaraciji navedene kao izlazne. Ukoliko je potrebno, moguće je varijable deklarirati i kao globalne:

```
global ime_varijable1 ime_varijable2 ...
```

Deklarisanje globalnih varijabli je potrebno izvršiti prije njihove prve upotrebe. Varijablu je potrebno proglasiti globalnom i u workspace-u, koliko je potrebno da bude dostupna i sa komandnog prompta.

Treba napomenuti da Matlab dozvoljava da se kao argumenti pri pozivu funkcije prenose, odnosno da funkcija vraća i kompleksne tipove podataka (matrice, stringove, strukture, polja ćelija, uključujući i objekte). Iako u Matlabu postoji mogućnost korištenja pointera i adresa, njihovo korištenje nije nužno.

#### 1.9.4. Kontrola toka izvršavanja m-fajla

Kontrola toka izvršavanja m-fajla je vrlo važna, obzirom da je bez toga nemoguće implementirati bilo kakav algoritam. U okviru programskog jezika Matlab-a postoje sve standardne strukture za kontrolu toka izvršavanja i u nastavku će biti kratko predstavljene.

---

## 1) *if* struktura

Sintaksa:

```
if uslov1
    kod1
elseif uslov2
    kod2
else
    kod3
end
```

Jasno je da se dio sa *else* i *elseif* mogu izostaviti.

U stvaranju vrijednosti uslova *uslov1* i *uslov2* se mogu koristiti izrazi sa relacionim operatorima (<, >, <=, >=, ==, ~=), kao i logički operatori & (and), | (or), xor, not. Treba voditi računa da relacioni izrazi sa matricama ne vraćaju proste logičke vrijednosti, nego matrice sa 0 i 1, ovisno o mjestu na kome su odnosi zadovoljeni odnosno nisu zadovoljeni. Zbog toga je potrebno u uslovima koristiti funkcije Matlaba, ako su argumenti izraza matrice:

*isequal(A,B)* – daje 1 ili 0 ovisno o tome da li su svi elementi A jednaki elementima B ili nisu  
*isempty(A)* – daje 1 ako je matrica A prazna  
*all(A)* – daje 1 ako su svi elementi A nenulti  
*any(A)* – daje 1 ako je barem jedan element A nenulti.

## 2) *switch* struktura

Izvršava grupu komandi, ovisno o vrijednosti varijable ili izraza.

Sintaksa:

```
switch varijabla
    case vrijednost1
        kod1
    case vrijednost2
        kod2
```

---

```
        .  
        .  
        .  
        otherwise  
            kod  
    end
```

Kod Matlaba se iz *switch* strukture automatski izlazi čim se jedan od uslova ispuni (za razliku od C-a).

### 3) *for* petlja

Omogućava izvršavanje dijela koda tačno određen broj puta.

Sintaksa:

```
    for n=1:10  
        kod  
    end
```

Dozvoljeno je postaviti proizvoljnu početnu, krajnju i vrijednost koraka, kao i eksplicitno navođenje elemenata vektora.

### 4) *while* struktura

Omogućava izvršavanje dijela koda sve dok je određeni uslov ispunjen.

Sintaksa:

```
    while uslov  
        kod  
    end
```

### 5) Komande za promjenu toka izvršavanja petlje.

Matlab posjeduje i komandu *break* koja omogućava izlazak iz *for* ili *while* petlje prije ispunjavanja uslova za izlazak.

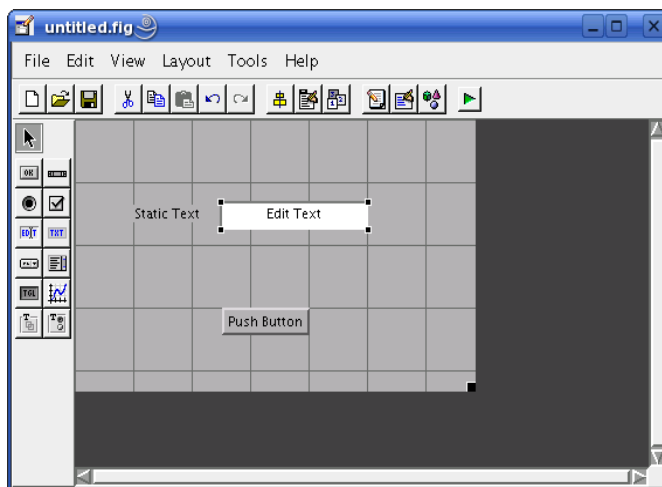
Isto tako, postoji i komanda *continue* koja prije završetka iteracije *for* ili *while* petlje vodi u sljedeću iteraciju.

---

### 1.9.5. Programiranje grafičkog interfejsa

Iako se interakcija korisnika sa Matlabom odvija preko komandnog prompta, Matlab omogućava i gradnju kompleksnog grafičkog interfejsa.

Grafički interfejs za neku aplikaciju pisanu u Matlabu je najjednostavnije razviti korištenjem alata *guide* (slika 1.6.).



Slika 1.6. Osnovni prozor alata *guide* za razvoj grafičkog interfejsa

Rad sa je sličan radu sa drugim alatima za razvoj GUI, i razvoj forme se sastoji u pozicioniranju vizualnih kontrola, koje se nalaze na traci sa lijeve strane, na željeno mjesto na formi. Svojstva svake kontrole je moguće podesiti korištenjem Property Inspector-a (slika 1.7.).

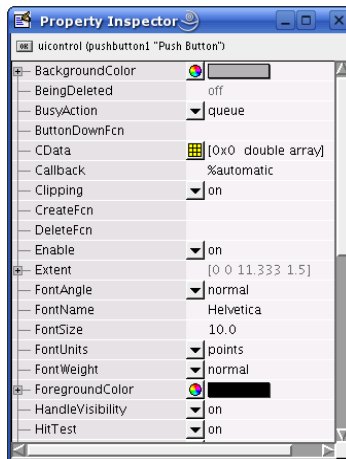
Matlab svaku formu pohranjuje u zaseban grafički fajl (fig-fajl), a programski kod koji inicijalizira formu, te implementira akcije vizuelnih kontrola se nalazi u m-fajlu sa istim imenom.

Da bi neki događaj sa vizuelnom kontrolom prouzrokovao željenu akciju, neophodno je napisati kod koji će se izvršiti kada se desi navedeni događaj. Ovaj kod se nalazi u pomenutom m-fajlu forme, a događaji kojima se može pridružiti

---

akcija su:

- aktiviranje vizualne kontrole (*CallBack*)
- kreiranje vizualne kontrole (*CreateFcn*)
- uništavanje vizualne kontrole (*DeleteFcn*)
- klik na dugme miša (*ButtonDownFcn*)
- pritisak tastera (*KeyPressFcn*)



Slika 1.7. Podešavanje svojstava vizualne kontrole

Do ovih funkcija se najlakše dolazi klikom na desno dugme miša sa pointerom na vizualnoj kontroli i izborom odgovarajućeg događaja u okviru *View Callback* u pop-up meniju.

Dvije osnovne funkcije koje omogućavaju očitavanje i postavljanje vrijednosti svojstava vizualne kontrole su *get* i *set*. Kao prvi argument ovim funkcijama se proslijeđuje identifikator kontrole (*handle*), a drugi argument je string sa imenom svojstva koje se očitava ili čija vrijednost se postavlja. Funkcija *set* ima i treći argument, koji predstavlja vrijednost svojstva koja se želi postaviti. Alat *guide* prilikom kreiranja forme kreira strukturu *handles* koja omogućava jednostavan pristup identifikatorima svih vizualnih kontrola na formi. Tako npr. *handles.pushbutton1* predstavlja identifikator vizualne kontrole čije svojstvo *Tag* ima vrijednost *'pushbutton1'*.

---

Kao primjer, da bi se svojstvo *String* vizualne kontrole *text1* postavilo na vrijednost 10, potrebno je izvršiti:

```
set(handles.text1,'String',int2str(10))
```

Ovdje treba obratiti pažnju da je broj 10 pretvoren u string korištenjem funkcije *int2str*.

Slično, da bi se u varijablu *var* smjestio broj koji je unesen u vizualnu kontrolu *edit1*, potrebno je izvršiti:

```
var = str2num(get(handles.edit1,'String'))
```

Obratiti pažnju da je izvršena konverzija stringa koji je pročitao u broj, korištenjem funkcije *str2num*.

Detaljnije upute za razvoj grafičkog interfejsa u Matlabu se mogu naći u dokumentaciji.

## **1.10. Control System Toolbox**

Kako je već navedeno, Matlab posjeduje veliki broj toolboxa koji se fokusiraju na različite problemske oblasti i implementiraju funkcije specifične za datu oblast. U okviru Control System Toolbox-a se nalaze praktično sve bitne funkcije za rješavanje problema u oblasti upravljanja i regulacije kontinualnih i diskretnih sistema. U nastavku će kratko biti predstavljen dio ovog toolboxa.

Control System Toolbox predstavlja kolekciju algoritama koji implementiraju najvažnije metode analize, modeliranja i projektovanja sistema upravljanja. Za dio ovih metoda postoji i grafički interfejs radi jednostavnijeg korištenja.

### *1.10.1. Predstavljanje sistema*

Objekti i elementi sistema upravljanja se mogu predstaviti u formi prenosne funkcije, faktorizirane prenosne funkcije preko nula, polova i statičkog pojačanja, te

u prostoru stanja. Moguć je rad sa kontinualnim i diskretnim sistemima, te transformacije između različitih načina predstavljanja sistema. Nako što se objekat predstavi u nekoj od pomenutih formi, na jednostavan način se može odrediti vremenski ili frekventni odziv, te geometrijsko mjesto korjena.

Neka je prenosna funkcija sistema data kao:

$$G(s) = \frac{s+0,5}{s^2+0,2 \cdot s+0,1}$$

Ovaj sistem se u Matlabu može opisati navedenom prenosnom funkcijom korištenjem funkcije *tf*:

$$G = tf([1,0.5],[1,0.2,0.1])$$

Dakle, kao argumenti funkciji *tf* se prosljeđuju vektori koeficijenata polinoma u brojniku i nazivniku, u opadajućem redoslijedu po *s*.

Slično, opis istog sistema u faktoriziranoj formi se može dobiti korištenjem funkcije *zpke*:

$$G = zpke(-0.5,[-0.1+0.3*i,-0.1-0.3*i],1)$$

U prostoru stanja se isti sistem može predstaviti definiranjem matrica A, B, C i D koje opisuju sistem u formi:

$$\begin{aligned} \dot{x} &= A \cdot x + B \cdot u & A &= \begin{bmatrix} -0,2 & -0,2 \\ 0,5 & 0 \end{bmatrix} \\ y &= C \cdot x + D \cdot u & B &= \begin{bmatrix} 2 \\ 0 \end{bmatrix} \\ & & C &= [0,5 \quad 0,5] \\ & & D &= 0 \end{aligned}$$

korištenjem funkcije *ss* :

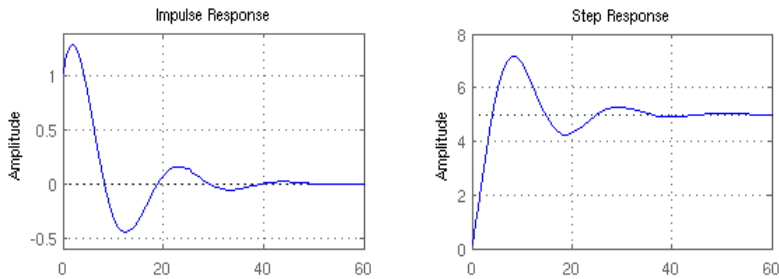
$$\begin{aligned} A &= [-0.2, -0.2; 0.5, 0] \\ B &= [2; 0] \\ C &= [0.5, 0.5] \\ D &= 0 \\ G &= ss(A, B, C, D) \end{aligned}$$

Konverziju iz jednog načina predstavljanja u drugi je moguće izvršiti vrlo lako, koristeći već navedene funkcije. Kao primjer, da bismo predstavu sistema u formi

prenosne funkcije transformisali u predstavu istog sistema u prostoru stanja, dovoljno je pozvati funkciju  $ss(G)$ , gdje je  $G$  opis sistema prenosnom funkcijom.

### 1.10.2. Analiza karakteristika sistema

Odziv pomenutog sistema na step ulaz i impulsni ulaz se može dobiti korištenjem funkcija *step* i *impulse*. Kao prvi argument je potrebno navesti sistem (u bilo kojoj od gore pomenutih formi), a kao drugi argument se može navesti krajnje vrijeme do kojeg se analizira ovaj odziv. Ukoliko se ovo vrijeme ne zada, biće određeno automatski. Ukoliko se pri pozivu ovih funkcija ne navedu varijable u koje će se smjestiti vrijednost odziva i vrijeme, odziv će biti nacrtan (slika 1.8.). U protivnom će se u ove varijable pohraniti vrijednosti odziva u datim trenucima, npr.  $[y,t]=step(G)$ .



Slika 1.8. Odzivi sistema: a) impulsni odziv b) odziv na step signal

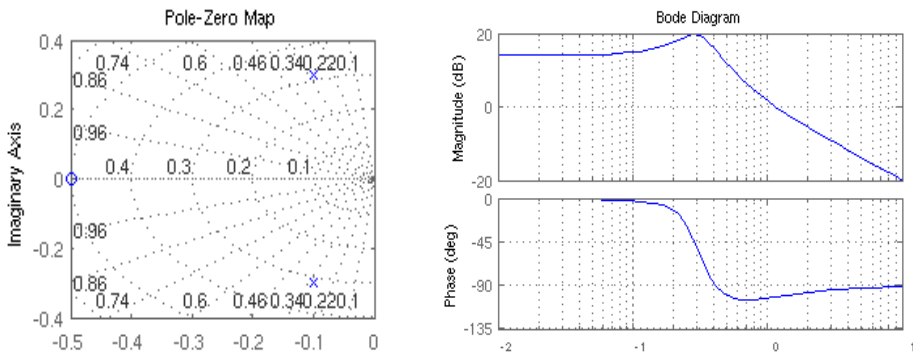
Primjenom funkcija  $pzmap(G)$  možemo nacrtati nule i polove prenosne funkcije, a pomoću  $bode(G)$  Bodeov dijagram za dati sistem (slika 1.9.).

Slično, pomoću funkcija  $nyquist(G)$  i  $nichols(G)$  se mogu dobiti Nyquistov i Nicholsov dijagram za dati sistem (slika 1.10.).

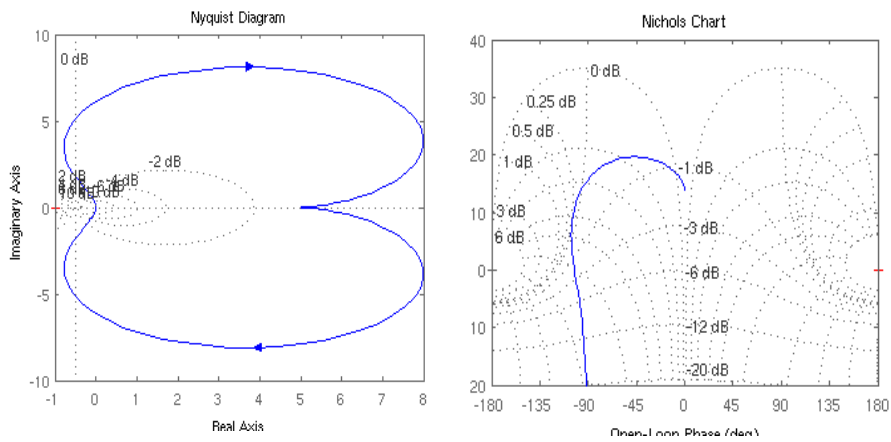
Ako imamo sistem sa negativnom povratnom spregom kao na slici 1.11., pomoću funkcije *feedback* možemo dobiti njegovu prenosnu funkciju:

$$W=feedback(G1,G2)$$

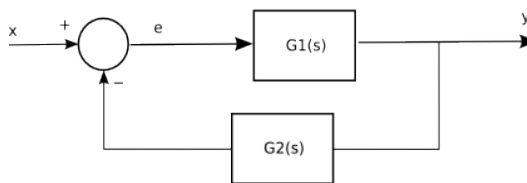




Slika 1.9. a) Polovi i nule prenosne funkcije b) Bodeov dijagram



Slika 1.10. a) Nyquistov dijagram b) Nicholsov dijagram



Slika 1.11. Sistem sa negativnom povratnom spregom

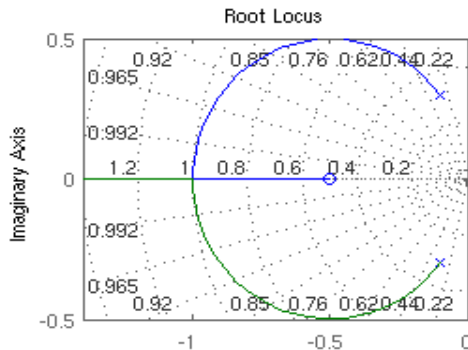
Ukoliko je potrebno odrediti prenosnu funkciju sistema sa jediničnom povratnom spregom u čijoj grani se nalazi blok sa gore datom prenosnom funkcijom  $G(s)$ , to možemo uraditi pozivom funkcije *feedback* :

$$W = \text{feedback}(G, 1)$$

Funkcija  $rlocus(G)$  crta geometrijsko mjesto korjena za dati sistem sa jediničnom negativnom povratnom spregom (slika 1.12.).

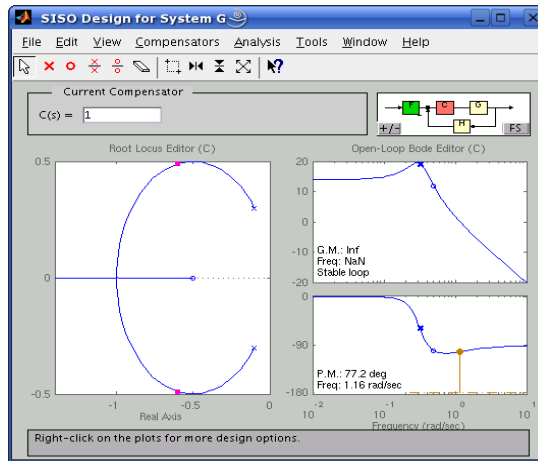
### 1.10.3. SISO Design Tool

Još jedan vrlo koristan alat predstavlja SISO Design Tool (slika 1.13.). To je grafički interfejs koji omogućava korištenje geometrijskog mjesta korjena, Bodeovih dijagrama i Nicholsovog dijagrama za interaktivno projektovanje kompenzatora. SISO Design Tool se pokreće sa komandnog prompta pozivom funkcije *sisotool*.

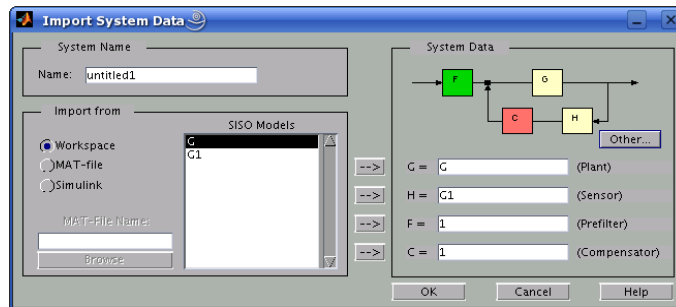


Slika 1.12. Geometrijsko mjesto korjena

Sistem upravljanja je predstavljen u gornjem desnom uglu forme alata. Objekat je predstavljen blokom  $G$ , a blok u grani povratne sprege blokom  $H$ . Kompenzator i filter ulaznog signala su predstavljeni blokovima  $C$  i  $F$ . Prenosne funkcije blokova se mogu importovati izborom menija *File* → *Import*, nakon čega se otvara dijalog (slika 1.14.) u kome se klikom na dugme *<Other>* može izabrati struktura sistema upravljanja. Za svaki od blokova sistema upravljanja se može importovati željena prenosna funkcija.



Slika 1.13. SISO Design Tool – alat za projektovanje kompenzatora za sisteme sa jednim ulazom i jednim izlazom



Slika 1.14. Dijalog za izbor strukture sistema i import prenosnih funkcija blokova

Procedura projektovanja kompenzatora se sastoji u interaktivnom pomjeranju nula i polova funkcije prenosa sistema sa zatvorenim povratnom spregom, te dodavanju nula i polova, čiji se efekti mogu odmah vidjeti na prikazanim dijagramima. Sa Bodeovog dijagrama se odmah mogu očitati i rezerve stabilnosti, dok se sa prenosne funkcije kompenzatora može očitati potrebno pojačanje.

Osim interaktivnog prikaza karakteristika sistema, analizu sistema je moguće izvršiti i u vremenskom kao i u frekventnom području, izborom željene opcije iz menija *Analysis*.

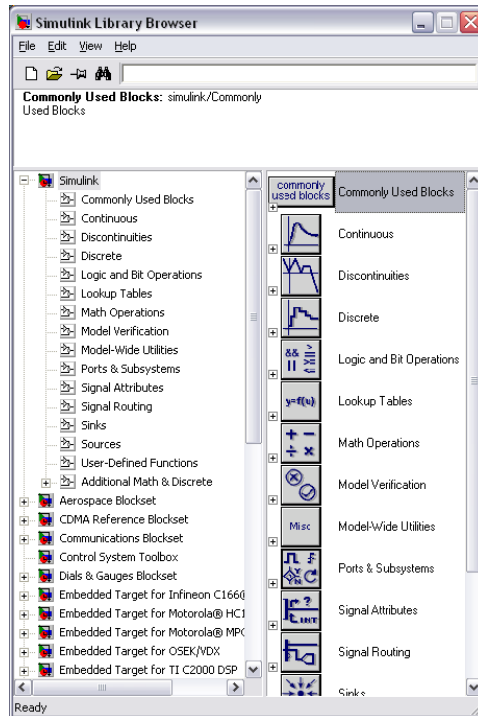
---

Detaljnije uputstvo za korištenje ovog alata, kao i primjeri korištenja, mogu se naći u dokumentaciji Control System Toolbox-a.

### **1.11. Simulink**

Simulink predstavlja softverski paket, sastavni dio okruženja Matlab/Simulink, namijenjen za modeliranje, simulaciju i analizu ponašanja dinamičkih sistema. Simulink omogućava modeliranje linearnih i nelinearnih sistema, koji su kontinualni po vremenu, diskretni po vremenu, ili sadrže i kontinualne i diskretne dijelove. Model sistema razvijen u Simulinku se sastoji od blokova, od kojih svaki realizira određenu funkcionalnost sistema. Blokovi su međusobno povezani na način koji odražava njihovu međusobnu razmjenu signala.

Simulink se pokreće unošenjem naredbe *simulink* na komandnom promptu Matlaba, ili klikom na dugme <Simulink>, koje se nalazi na toolbaru Matlaba. Nakon pokretanja, biće prikazan preglednik biblioteke blokova (slika 1.15.). Blokovi su organizirani u grupe, na osnovu funkcije koju obavljaju. Tako su npr. svi blokovi koji generišu neki signal, odnosno koji imaju samo izlaz (konstanta, generator signala, step signal, itd.) svrstani u grupu “*Sources*”. Svi blokovi koji prihvataju signal radi prikazivanja ili pohranjivanja, odnosno koji imaju samo ulaz (osciloskop, displej, pohranjivanje u varijablu Matlaba, pohranjivanje u fajl, X-Y grafik i sl.) su svrstani u grupu “*Sinks*”. Na sličnoj osnovi su grupisani blokovi u grupe “*Continuous*”, “*Discrete*”, “*Math Operations*” i sl. Takođe, svaki toolbox podržan od strane Simulinka ima svoju grupu (*blockset*), unutar koga se nalaze blokovi koji realiziraju funkcije toolboxa. Ukoliko nije poznato u kojoj grupi se nalazi neki blok, moguće ga je potražiti unošenjem njegovog imena ili nekog pojma vezanog za funkciju koju implementira u polje za traženje u gornjem dijelu prozora preglednika biblioteke blokova.



Slika 1.15. Preglednik biblioteke blokova

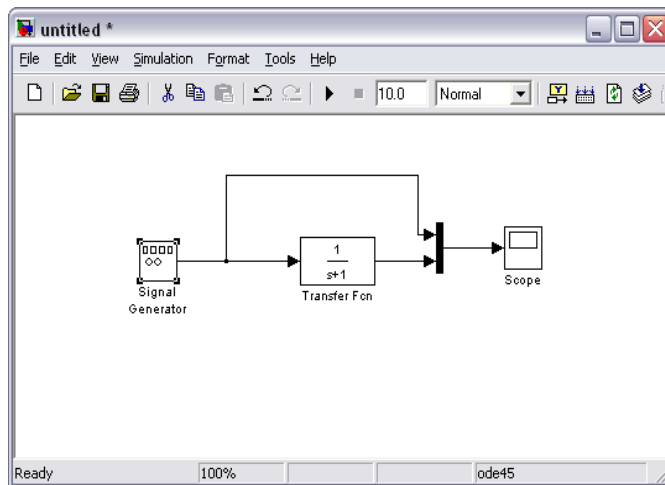
### 1.11.1. Rad sa Simulinkom

Simulink posjeduje vrlo razvijen grafički interfejs, tako da je gradnja modela vrlo jednostavna. Da bi se simuliralo ponašanje nekog sistema potrebno je proći sljedeće korake:

- Nacrtati model sistema, koji se sastoji od blokova i veza između blokova,
- Podesiti parametre svih blokova koji ih posjeduju,
- Podesiti parametre izvođenja simulacije,
- Pokrenuti simulaciju.

Da bi se nacrtao model sistema, potrebno je otvoriti novi model, klikom na novi dokument na toolbaru preglednika biblioteke blokova, ili izborom menija File->New->Model.

Blokovi se na novi model postavljaju povlačenjem iz preglednika biblioteke blokova (drag-and-drop). Pri radu sa blokovima koji se nalaze u modelu su podržane sve standardne opcije (Select, Copy, Cut, Paste). Veze između blokova se ostvaruju povlačenjem linije od ulaza (ili izlaza) jednog do izlaza (ili ulaza) drugog bloka. Ulaz nekog bloka je moguće povezati i sa linijom koja već povezuje druge blokove. Na slici 1.16. je prikazan izgled jednostavnog modela.



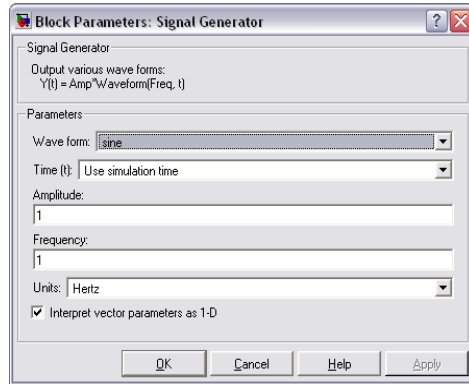
Slika 1.16. Model sistema

### 1.11.2. Podešavanje parametara

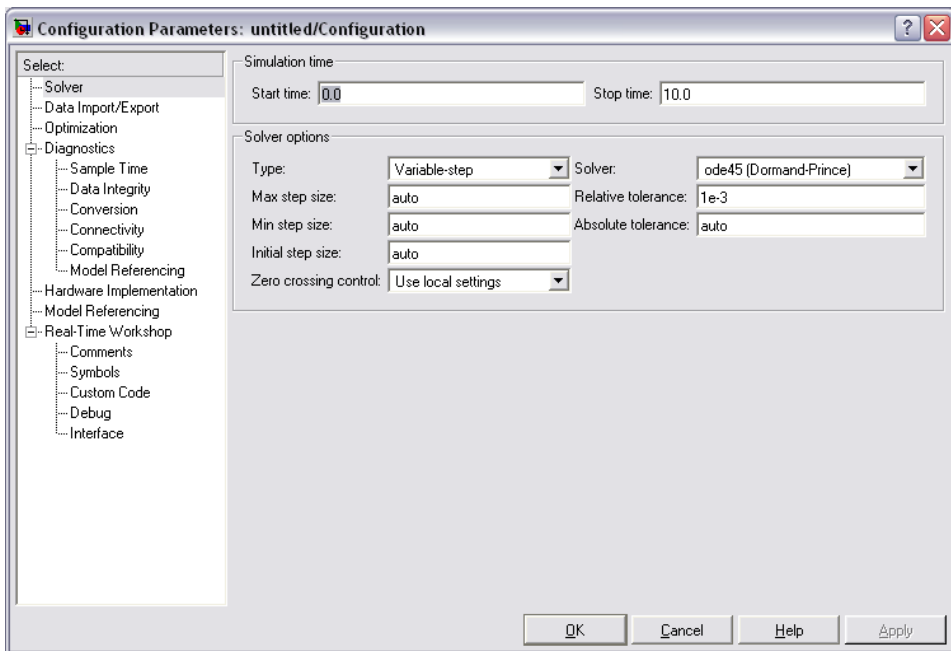
Treba napomenuti da većina blokova u Simulinku posjeduje parametre koji se mogu mijenjati. Tako npr. virtualni multiplekser (Mux) kao parametar ima broj ulaza, generator signala kao parametre ima tip signala, amplitudu i frekvenciju i sl. Da bi se otvorio dijalog za podešavanje parametara bloka (slika 1.17.) potrebno je dvostruko kliknuti na dati blok.

Osim parametara blokova, neophodno je podesiti i parametre izvođenja simulacije, kao što su vrijeme početka (*Start time*), vrijeme završetka simulacije (*Stop time*), tip i vrstu solvera koji će se koristiti za numeričku integraciju, veličinu koraka simulacije i sl. Ovi parametri se podešavaju izborom menija Simulation->Configuration

parameters (slika 1.18).



Slika 1.17. Dijalog za podešavanje parametara bloka



Slika 1.18. Dijalog za podešavanje parametara izvođenja simulacije

Najbitniji parametri čiju vrijednost treba postaviti su vrijeme završetka simulacije, te tip i vrsta solvera.

Vrijeme završetka simulacije (*Stop time*) predstavlja trajanje simulacije u sekundama

---

simulacionog vremena. Vrlo važno je napomenuti da Simulink ponašanje sistema simulira ne u realnom, nego u simulacionom vremenu. To znači da trajanje jedne sekunde simulacionog vremena može biti puno kraće, ali i puno duže od stvarnog trajanja jedne sekunde. Brzina izvođenja simulacije ovisi prije svega o kompleksnosti modela sistema čije ponašanje se simulira.

Međutim, treba reći da Simulink zajedno sa Real-Time Workshop-om predstavlja vrlo moćno okruženje za razvoj aplikacija za rad u realnom vremenu. Ove aplikacije se mogu izvršavati na nekom specifičnom hardveru za koji postoji podrška u okviru Real-Time Workshop-a (za koji postoji Real-Time Target), ili na PC računaru (xPC Target i Real-Time Windows Target). U tom slučaju Simulink predstavlja interfejs prema aplikaciji koja se odvija u realnom vremenu.

Postoje dva osnovna tipa solvera u Simulinku, a to su solveri sa varijabilnim korakom i solveri sa fiksnim korakom integracije.

Solveri sa fiksnim korakom integracije izračunavaju stanje modela u diskretnim trenucima vremena, pri čemu je prirast vremena uvijek konstantan. Tipičan predstavnik ovih solvera je Runge-Kutta. Pri tome Simulink može automatski odrediti potreban korak integracije (kada je vrijednost parametra Fixed-step size postavljena na *auto*), ili se vrijednost koraka integracije može zadati eksplicitno.

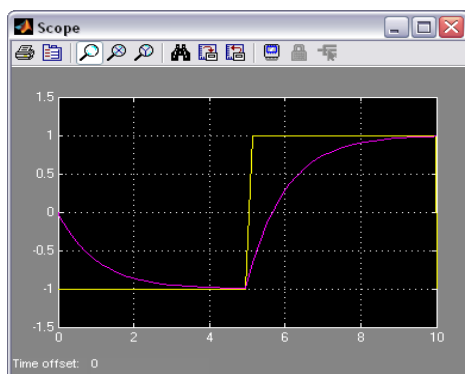
Za razliku od solvera sa fiksnim korakom integracije, solveri sa varijabilnim stepenom integracije mijenjaju veličinu koraka integracije unutar granica određenih vrijednostima parametara *Max step size* i *Min step size*, ovisnosti o kompleksnosti stanja sistema u datom trenutku simulacionog vremena. Ovi solveri su daleko efikasniji, obzirom da su u stanju povećati korak integracije kada to dopušta model, skraćujući time bitno trajanje izvršavanja simulacije.

Treba napomenuti da je u najvećem broju slučajeva sasvim prihvatljivo ostaviti postavke solvera na inicijalnim vrijednostima.



---

Nakon što je završen model sistema, potrebno je pokrenuti simulaciju. To se radi izborom menija Simulation->Start ili klikom na dugme <Start> na toolbaru modela. Nakon završetka izvođenja simulacije, rezultati simulacije se mogu pogledati i dalje procesirati. Na slici 1.19. je prikazan izlaz simulacije modela sa slike 1.16. na osciloskopu, koji se dobije dvostrukim klikom na blok Scope.



Slika 1.19. Rezultat simulacije sistema sa slike 1.16.

---

---

## **2. LabView**

### ***2.1. Uvod***

LabVIEW (Laboratory Virtual Instrument Engineering Workbench) predstavlja razvojno okruženje zasnovano na grafičkom programskom jeziku G. LabVIEW podržava komunikaciju sa hardverom korištenjem različitih interfejsa, kao što su GPIB, VXI, PXI, RS-232, RS-485, te uključuje podršku za različite kartice za akviziciju podataka. LabVIEW takođe posjeduje ugrađene biblioteke za podršku različitim softverskim standardima, kao što su TCP/IP i ActiveX. Posjeduje veliki broj biblioteka za prikupljanje, analizu, prezentaciju i pohranjivanje podataka, te sve standardne alate neophodne za razvoj. Pomoću LabVIEW okruženja se mogu kreirati i izvršne datoteke.

Osnovna prednost LabVIEW-a, osim pomenute podrške za različit hardver, se sastoji u tome što za razvoj aplikacija nije potrebno veliko iskustvo u programiranju.

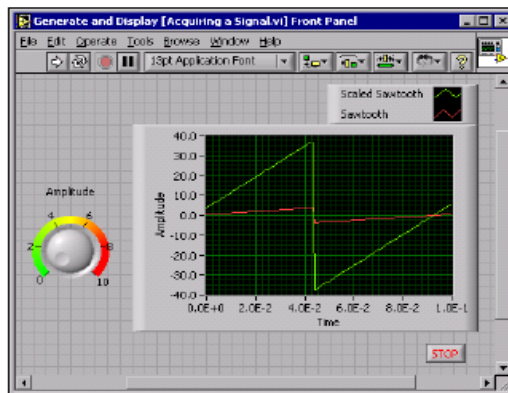
### ***2.2. Korištenje LabVIEW-a***

Programi u LabVIEW-u se nazivaju virtualnim instrumentima (VI), pošto njihov izgled i rad oponašaju fizičke instrumente, kao što su osciloskop ili univerzalni instrument. Kako je već rečeno, LabVIEW posjeduje različite alate za prikupljanje, analizu, prikaz i pohranjivanje podataka, te alate za debugiranje i praćenje toka izvršavanja koda.

---

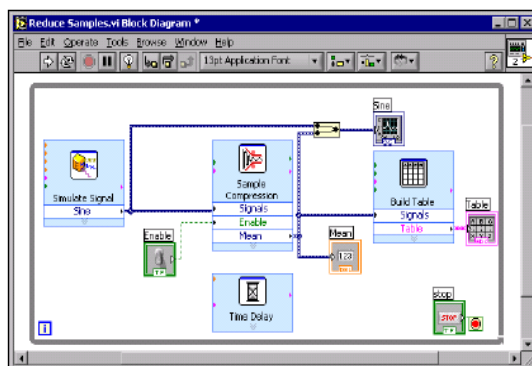
### 2.2.1. Virtualni instrument (VI)

Virtualni instrument se sastoji od dvije osnovne komponente, korisničkog interfejsa i blok dijagrama. Interaktivni korisnički interfejs virtuelnog instrumenta naziva se prednji panel (*front panel*) jer oponaša izgled panela fizičkog instrumenta (slika 2.1.). Na prednjem panelu se mogu nalaziti različiti simulira prednji panel fizičkog instrumenta. Prednji panel može sadržati različite kontrole (elemente korisničkog interfejsa). Kontrole predstavljaju ulazne uređaje, kao što su dugmad, potencimetri, klizači i slično. Indikatori su izlazni uređaji, kao npr. grafik, LED, različiti displeji i sl. Interakcija korisnika sa interfejsom virtualnog instrumenta se odvija putem miša i tastature kao ulaznih uređaja, te monitora kao izlaznog uređaja.



Slika 2.1. Korisnički interfejs aplikacije u LabVIEW-u

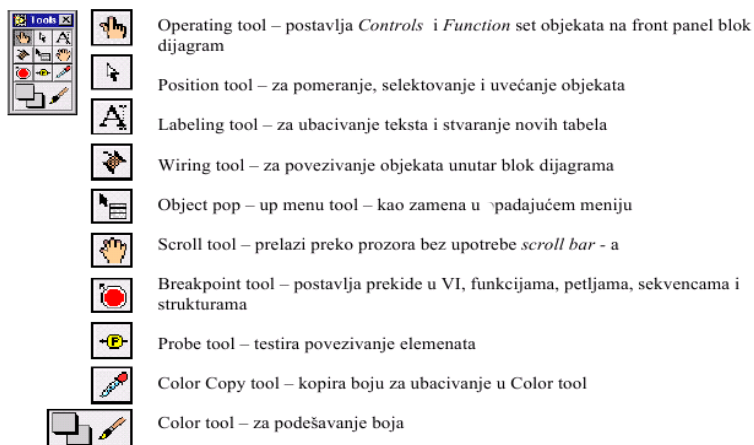
Blok dijagram (*block diagram*) (slika 2.3.) predstavlja grafičku realizaciju funkcije virtualnog instrumenta, odnosno izvorni kod virtualnog instrumenta. Blok dijagram očitava stanja i vrijednosti kontrola sa korisničkog interfejsa, realizira algoritam koji procesira ove podatke, te prezentira izlazne vrijednosti virtualnog instrumenta. Osim korisničkog interfejsa, virtualni instrument može koristeći različite protokole, podatke primiti i od drugih virtualnih instrumenata koji se nalaze na višem hijerarhijskom nivou, te pozivati druge virtualne instrumente i prosljeđivati im vrijednosti argumenata.



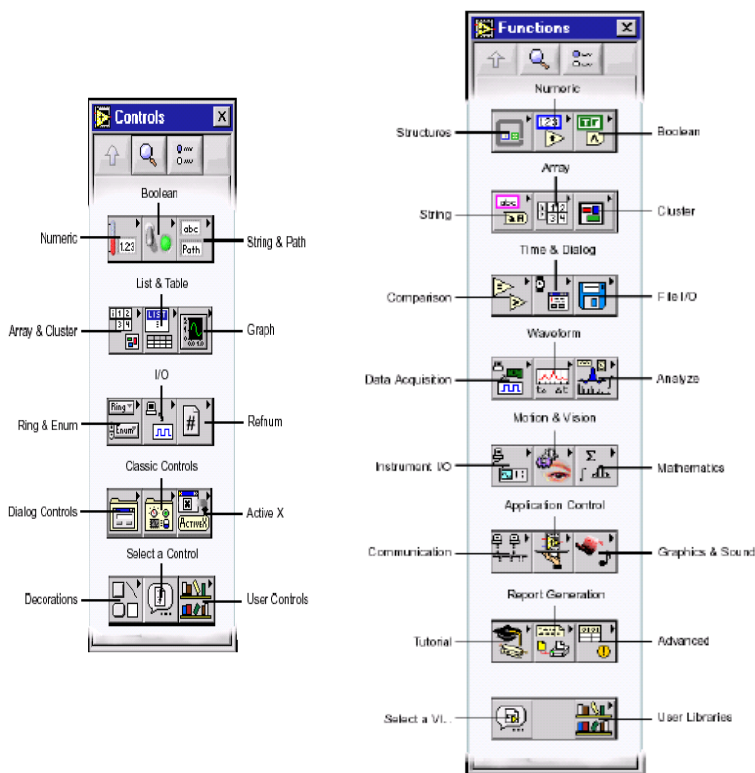
Slika 2.2. Blok dijagram koji realizira algoritam virtualnog instrumenta

### 2.2.2. Dizajniranje virtualnog instrumenta

Razvoj aplikacije u LabVIEW-u se sastoji u gradnji korisničkog interfejsa (front panela) VI, koji sadrži sve potrebne kontrole i indikatore. Nakon završetka projektovanja korisničkog interfejsa, potrebno je aplikaciji dodati kod koji upravlja objektima na front panelu, u formi blok dijagrama. Razvojno okruženje LabView-a posjeduje grafičke alate za realizaciju ovih aktivnosti. Osnovne alatke razvojnog okruženja LabView-a su predstavljene na slikama 2.3. i 2.4..



Slika 2.3. Alatke za rukovanje objektima na front panelu i blok dijagramu



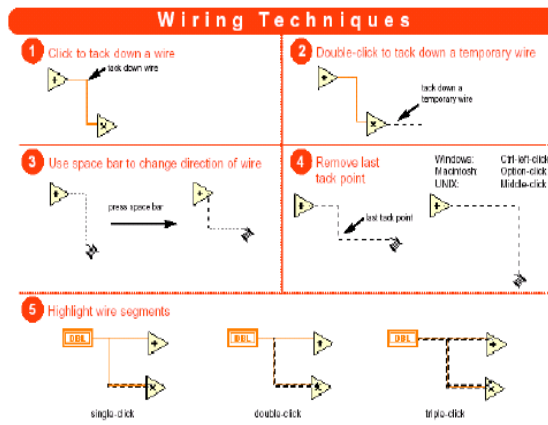
Slika 2.4. Kontrolne za gradnju front panela i funkcije za gradnju blok dijagrama

Alatke sa trake Tools (slika 2.3.) omogućavaju obavljanje osnovnih radnji sa korisničkim interfejsom i blok dijagramom, kao što su: dodavanje kontrole korisničkom interfejsu; promjena pozicije ili veličine kontrole na korisničkom interfejsu, odnosno bloka na dijagramu; dodavanje labele; postavljanje prekidne tačke i sl. Izbor odgovarajućeg alata sa ove trake se vrši klikom pointera miša.

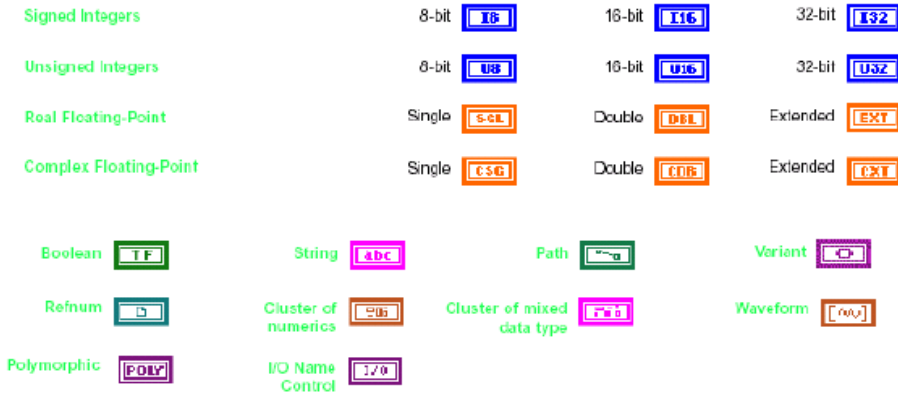
Traka sa kontrolama za gradnju front panela omogućava postavljanje željene kontrole na grafički interfejs, dok traka sa funkcijama omogućava postavljanje na dijagram bloka koji realizira neku funkciju (slika 2.4.).

Povezivanje objekata u blok dijagramu se vrši preko Wiring tool. Osnovne akcije za povezivanje su predstavljene na slici 2.5. Svaki objekat iz front panela ima svoj lik u blok dijagramu pri čemu je označen i oblik informacije (podatka) koji je naznačen

bojom (slika 2.6).



Slika 2.5. Osnovne akcije za pozvezivanje blokova u blok dijagramu



Slika 2.6. Osnovni tipovi podataka u LabVIEW i njihovo označavanje

Biblioteke virtuelnih instrumenata za obradu i analizu prikupljenih ili simuliranih signala kojima raspolaže LabVIEW daju mogućnosti razvoja programa za veoma širok domen primjena od upravljanja statističkim procesima do digitalne obrade signala. U okviru skupa funkcija za analizu podataka na raspolaganju stoje sljedeće grupe VI:

- 
- Vjerovatnoća i statistika. Statističke funkcije za srednju vrijednost ili standardnu statističku devijaciju skupa podataka kao i statističku funkciju vjerovatnoće i analizu varijanse.
  - Linearna algebra. Algebarske funkcije sa realnim i kompleksnim brojevima, vektorima i matricama.
  - Niz. Numeričke operacije sa jednodimenzionalnim i dvodimenzionalnim nizovima.
  - Generisanje signala. Generisanje digitalnog i analognog signala različitih talasnih oblika.
  - Digitalna obrada signala. Transformacija u frekvencijskom domenu, analiza u frekvencijskom domenu, analiza u vremenskom domenu i druge transformacije.
  - Digitalno fitriranje. IIR, FIR i nelinearna funkcija digitalnog filtriranja.
  - Smoothing Windows. Data windowing.
  - Interpolacija krivih.
  - Spektralna analiza i mjerenja. Funkcije koje su orijentisane ka mjerenju spektra jednog bočnog opsega (single-sided band, SSB), određivanju vršne snage i učestanosti.
  - Dodatne numeričke metode kao što su fitovanje funkcija, numerička integracija i detekcija vršne vrijednosti i sl.

### 2.2.3. Osnovne programske strukture

Blok dijagram u okviru virtualnog instrumenta može sadržavati i osnovne strukture za kontrolu toka izvršavanja algoritma, pa će one u nastavku biti kratko predstavljene. Ove strukture se izvršavaju automatski kada se na ulazu pojave podaci. Kada se izvrše operacije u okviru strukture, podaci se šalju na izlaz. Ulazni i izlazni priključci na okviru strukture, koji se generišu automatski kada veza presječe okvir strukture, nazivaju se tuneli (*tunnels*). Pored tunela, strukture imaju i



---

druge priključke.

Blokovi koji realiziraju dvije osnovne vrste petlji (FOR i WHILE) su predstavljeni na slici 2.7. FOR Loop omogućava da se dijagram koji je obuhvaćen ovom strukturom izvrši određen broj puta. Broj ponavljanja petlje označen je sa N i može se zadavati direktno, povezivanjem vrijednosti koja se nalazi izvan petlje sa priključkom N, ili implicitno, preko autoindeksiranja. Priključak iteracije (*iteration terminal*) sadrži realni broj tekuće iteracije. WHILE Loop predstavlja uslovnu programsku strukturu koja obezbjeđuje da se poddijagram obuhvaćen ovom strukturom izvodi sve dok je određeni uslov ispunjen. Struktura sadrži priključak za uslov (*conditional terminal*) i priključak rednog broja iteracije. Povezivanjem binarnog broja sa priključkom za uslov vrši se upravljanje petljom i na kraju svake iteracije provjerava se vrijednost broja koji je povezan sa terminalom za uslov i ako je ona jednaka logičkoj jedinici izvršava se sljedeća iteracija. Priključak rednog broja iteracije ima istu funkciju kao kod FOR Loop programske strukture. Obje programske petlje mogu koristiti priključke za privremeno pamćenje podataka prethodne iteracije koji će biti korišteni u narednoj iteraciji.



Slika 2.7. Osnovne petlje: FOR Loop i WHILE Loop

Case i Sequence programske strukture (slika 2.8.) mogu sadržavati više poddijagrama konfigurisanih u više nivoa, pri čemu je samo jedan vidljiv u datom trenutku.



Slika 2.8. CASE i SEQUENCE programske strukture

Case struktura sadrži dva ili više poddijagrama (*cases*), od kojih se samo jedan izvršava kada se izvršava struktura. Izbor poddijagrama koji će se izvršavati vrši se preko priključka koji se naziva selektor (*selector*). Ako je sa selektorom povezan

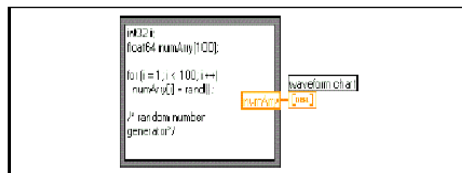
---

binarni broj, struktura sadrži dva poddijagrama koji odgovaraju FALSE i TRUE stanjima. Povezivanjem cijelog broja na selektor može se vršiti izbor poddijagrama u opsegu od 0 do  $2^{15}-1$ .

Sekvencijalna struktura, čiji izgled podsjeća na komadić filmske trake, sadrži jedan ili više poddijagrama ili ramova koji se izvršavaju sekvencijalno, po utvrđenom redu. Podaci se šalju iz strukture tek po izvršavanju posljednjeg poddijagrama. Sekvencijalna struktura se koristi za upravljanje redosljedom izvršavanja čvorova kada je on značajan, a nije uslovljen tokom podataka. Ovakav slučaj nije rijedak, a kao primjer može poslužiti upravljanje instrumentacijom preko IEEE-488 interfejsa, tj. prema programabilnom instrumentu se najprije pošalje komanda, a zatim izvrši očitavanje podataka. Ukoliko se ne bi koristile sekvencijalne strukture, moglo bi prvo doći do očitavanja podataka iz instrumenta, pa tek potom slanje komande, što bi dovelo do greške u radu.

#### 2.2.4. Matematički izrazi

U okviru blok dijagrama mogu se koristiti matematički izrazi koji se upisuju u uokviren pravougaonik koji se naziva čvor formule (*Formula Node*), slika 2.9.



Slika 2.9. Blok sa matematičkim izrazom

Čvor formule sadrži jednu ili više matematičkih funkcija ili formula i za pisanje formula koristi se BNF (*Backus-Naur Form*) notacija uobičajena za pisanje matematičkih izraza i kod klasičnih programskih jezika.

---

Po obodu programske strukture za formulu dodaju se priključci u koje se upisuju ulazne i izlazne promenljive. LabVIEW raspolaže velikim brojem funkcija koje se mogu koristiti u izrazima (aritmetičke, trigonometrijske, logaritamске i druge funkcije).

### 2.2.5. Vremenske funkcije

Vremenske funkcije (slika 2.10) omogućavaju očitavanje tekućeg vremena, proteklog vremena ili deaktiviranje neke operacije na određeno vrijeme. Kod većine vremenskih funkcija u LabVIEW osnovna jedinica za vrijeme je milisekunda.



Slika 2.10. Osnovne vremenske funkcije: (a) Tick Count (ms); (b) Wait (ms);

(c) Wait Until Next ms Multiple; (d) Get Data/Time String

Funkcija Tick Count (ms) daje podatak o vremenu, izraženo u milisekundama, koje je prošlo od momenta uključenja. Maksimalna dužina vremenskog intervala iznosi  $2^{32}-1$ .

Funkcija Wait (ms) zadržava izvršavanje funkcije za specificirani broj ms. Funkcija daje podatak o sadržaju milisekundnog tajmera po isteku zadatog intervala čekanja.

Brzinom izvršavanja programskih struktura For Loop/While Loop može se upravljati preko funkcija Wait Until Next ms Multiple. Period izvršavanja programske petlje definiira numerička konstanta koja se povezuje sa ikonom Wait Until Next ms Multiple i brojno je jednak vrijednosti konstante izražene u milisekundama.

Funkcija Get Data/Time String daje, u obliku odgovarajućih nizova, podatke o datumu i tekućem vremenu specificiranom preko broja sekundi proteklih od 12:00

---

GMT 1.januara 1904. god. Na raspolaganju stoje length vremenske funkcije Get Data/Time in Seconds koja daje podatak o proteklom vremenu u sekundama, Data/Time to SecondsTS. Konverziju datuma u proteklo vrijeme length Seconds To Data/Time za konverziju proteklog vremena izraženog u sekundama u tekući datum. Kod svih ovih funkcija kao referentno vrijeme uzima se 12:00 GMT, 1.januar 1904. god.

### 2.2.6. Nizovi

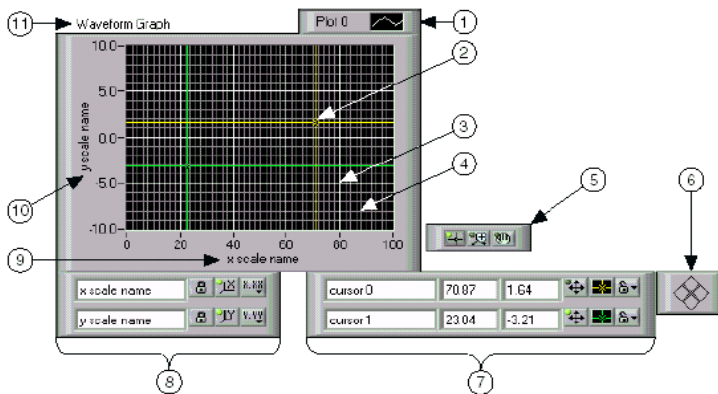
Nizovi ASCII karaktera (*ASCII strings*) su vrlo važni i imaju više namjena u akviziciji podataka. Numerički podaci se najčešće prenose kao nizovi karaktera koji se zatim ponovo konvertuju u brojne vrednosti. Pri memorisanju podataka brojevi se najpre konvertuju u nizove ASCII karaktera koji formiraju ASCII fajl. Za komunikaciju sa programabilnom instrumentacijom preko serijskog interfejsa RS-232 ili instrumentacionog paralelnog interfejsa IEEE-488 koriste se komande u vidu niza ASCII karaktera. LabVIEW raspolaže velikim brojem funkcija za rad sa nizovima uključujući i one koji vrše konverziju nizova u brojeve i obrnuto. Funkcija za upravljanje nizom (*string control*) i funkcija za unošenje broja povezuju se sa funkcijom za formatiranje niza Format Into String koja povezuje nizove sa ulaza, formatira brojeve a zatim generiše jedan niz na izlazu. Broj karaktera u formiranom nizu može se odrediti korištenjem funkcije za određivanje dužine niza String Length.

### 2.2.7. Grafici

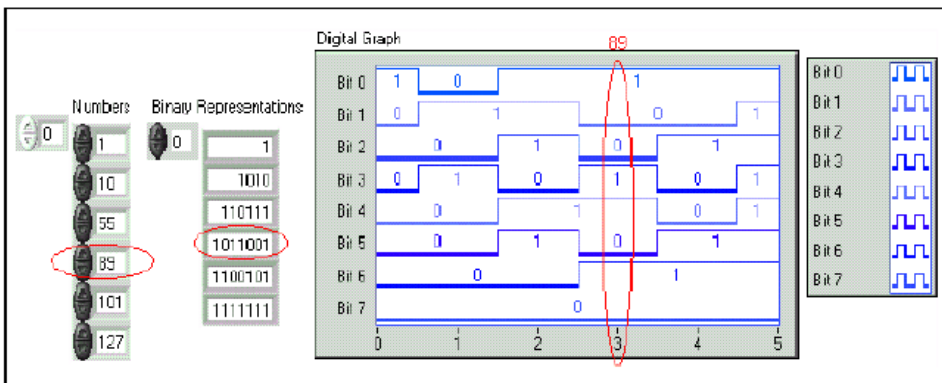
Osim prikaza tekstualnih poruka, u akviziciji podataka često se traži oblik podataka u vidu grafika. Na slici 2.11. je prikazan je oblik grafa, dok je na slici 2.12. dat prikaz digitalnog grafa.

Osnovni elementi na grafičkom prikazu su: 1- legenda plotu, 2-cursor, 3-oznaka grida, 4-mini oznaka grida, 4-paleta grafa, 6- kursor za pomjeranje, 7-legenda grafa,

8- legenda skale, 9-X osa, 10-Y osa, 11- oznaka grafa.



Slika 2.11. Grafički prikaz kontinualnog signala



Slika 2.11. Grafički prikaz digitalnog signala

### 2.2.8. Pohrana podataka

LabVIEW poseduje skup funkcija File I/O namenjenih za rad sa fajlovima, gdje pored funkcija za pamćenja ili očitavanja podataka, na raspolaganju stoje i druge funkcije kao što su kreiranje fajla tipa tabele, memorisanje podataka u binarnom formatu i sl. Podaci koji se memorišu mogu biti u jednom od sledećih formata:

1. Niz bajta. Podaci, čija je osnovna jedinica bajt, mogu biti istog ili različitog tipa. Ovaj format pruža mogućnost da se podaci koriste i u drugom softverskom

---

paketu, npr. u programu za rad sa tabelama ili u programu za obradu teksta.

2. Datalog fajl. Podaci su u binarnom formatu i ovakve fajlove može koristiti samo LabVIEW.

3. Niz binarnih bajta. Odlikuje se velikom kompaktnošću i brzinom memorisanja podataka. Podaci se najprije konvertuju u binarni niz i mora se znati koji tip podataka se memoriše, odnosno očitava.

### *2.2.9. Komunikacija sa hardverom*

Jedna od najvažnijih funkcija programskog paketa LabVIEW je upravljanje akvizicionim karticama za direktno povezivanje na ISA ili PCI sabirnicu personalnog računara. Ove kartice obuhvataju različite kombinacije analognih i digitalnih ulaza i izlaza. U okviru funkcije za akviziciju podataka nalaze se klase VI za akviziciju analognih signala Analog Input VIs, generisanje analognih signala Analog Output VIs, akviziciju i generisanje digitalnih signala Digital I/O VIs, brojanje impulsa i generisanje vremenskih signala Counter VIs, kalibraciju i konfiguraciju akviziciono-upravljačke kartice Calibration and Configuration VIs i virtuelni instrumenti za prilagođenje i obradu signala Signal Conditioning VIs. U okviru svake klase virtuelnih instrumenata, nalazi se veći broj VI za specifične primjene, od najjednostavnijih do najsloženijih. Ovi VI se koriste u blok dijagramu za realizaciju određenih funkcija akvizicije ili upravljanja.

Izbor akviziciono-upravljačke kartice povezan je sa konkretnim zahtjevima određenog zadatka. Danas, veliki broj proizvođača hardvera uz svoje proizvode nudi VI ili razvojni alat za izradu VI za LabVIEW, čime se znatno proširuje izbor hardvera koji se može koristiti sa ovim programskim paketom.

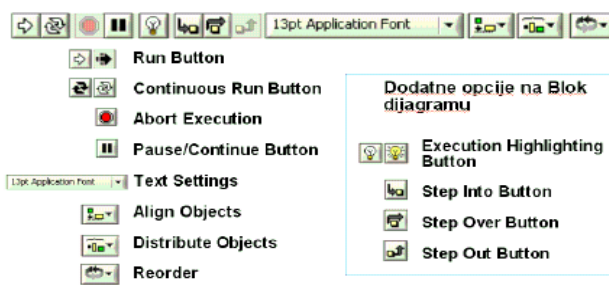
Preko instrumentacionih drajvera ostvaruje se komunikacija sa programabilnom instrumentacijom. Instrumentacioni drajver predstavlja skup virtuelnih instrumenata preko kojih se komunicira sa instrumentom korišćenjem standardnih VISA (Virtual Instrument Software Architecture) I/O funkcija. Drajver ili VI

---

predstavlja komandu visokog nivoa, kao što je Read DC Voltage VI za digitalni multimetar, ili Configure Time Axis VI za digitalni osciloskop. Pozvani drajver automatski šalje komande u vidu nizova karaktera koje odgovaraju određenom instrumentu. LabVIEW raspolaže velikim brojem drajvera za upravljanje instrumentacijom na bazi IEEE-488, RS-232 interfejsa i VXI instrumentacije. Drajveri za veći broj programabilnih instrumenata nalaze se u VISA biblioteci i dostupni su preko instrumentacionih I/O interfejsa, ali postoji i mogućnost izrade instrumentacionog drajvera za specifičnu primenu. Treba napomenuti da su raspoloživi instrumentacioni drajveri isporučeni sa blok-dijagramom, koji korisnik može editovati odnosno prilagoditi specifičnim zahtjevima.

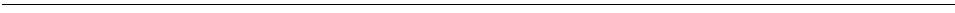
### 2.2.10. Korištenje virtualnog instrumenta

Upravljanje aplikacijom u LabVIEW-u, odnosno upravljanje virtualnim instrumentom se vrši korištenjem Status Toolbar-a (slika 2.12.), koji se nalazi na front panelu i na blok dijagramu.



Slika 2.12. Status Toolbar

Aplikacija se pokreće klikom na Run Button. Nakon pokretanja Run Button mijenja izgled, indicirajući na taj način da je virtualni instrument aktivan. Virtualni instrument se može pauzirati i ponovo pokrenuti.





---

## 3. Automatsko upravljanje

### 3.1. Uvod

U ovom poglavlju je dat uvod u automatsko upravljanje. Pojašnjena je terminologija i ovisnosti između elemenata sistema upravljanja, te ukazano na vezu između fizičke realizacije sistema automatskog upravljanja.

### 3.2. Osnovna terminologija

Različite fizikalne veličine, kao npr. pritisak, temperatura ili protok, se u okviru mašina ili postrojenja moraju držati na određenim vrijednostima. Ove vrijednosti se ne smiju promijeniti ni u slučaju djelovanja različitih poremećaja. Upravo ovakvi problemi se rješavaju realizacijom automatskog upravljanja.

Da bi se neka veličina koju je potrebno regulirati mogla dovesti na regulator kao električna veličina, neophodno je izvršiti njenu konverziju na odgovarajući način. Nakon toga se ova veličina može u okviru regulatora uporediti sa zadanom vrijednošću, ili zadanom promjenom. Rezultat ovog poređenja ukazuje na to kakvo je djelovanje neophodno da bi se ostvario postavljeni cilj.

Da bi se izvršilo pogodno djelovanje na postrojenje, neophodno je da postoji neki uređaj koji aktivno utiče na vrijednost posmatrane veličine u postrojenju (npr. grijač). Pored toga, neophodno je poznavati karakteristike ponašanja postrojenja.

Teorija automatskog upravljanja pokušava da pronađe međuovisnosti koje vrijede bez obzira na tehnologiju i prirodu procesa koji se odvijaju u postrojenjima. Ove međuovisnosti se izučavaju u okviru teorijskih predmeta iz oblasti automatskog upravljanja, a dosta informacija o njima se može pronaći u literaturi, koja ih tretira uz upotrebu matematičkog aparata. U nastavku ovog poglavlja će osnove

---

automatskog upravljanja biti predstavljene na takav način da se jednostavno uspostavi veza između fizičke realizacije sistema upravljanja i teorijske podloge koja omogućava njihovu analizu i projektovanje.

### *3.2.1. Zadana vrijednost*

Automatsko upravljanje se odvija tako da se regulirana veličina održava na potrebnoj vrijednosti, ili da se mijenja na zahtijevani način. Veličina koja definira pomenutu vrijednost ili promjenu se naziva zadana vrijednost.

### *3.2.2. Regulirana veličina*

Fizikalna veličina mašine ili postrojenja čija vrijednost treba da prati zadanu vrijednost se naziva regulirana veličina. Regulirana veličina može biti:

- pritisak u pneumatskom rezervoaru
- pritisak hidraulične prese
- temperatura galvanske kupke
- protok rashladne tečnosti kroz izmjenjivač toplote
- koncentracija neke hemikalije u hemijskom reaktoru
- brzina obrtanja ose u mašini sa električnim motorom
- pozicija radne alatke robotske ruke i sl.

### *3.2.3. Manipulativna veličina*

Na vrijednost regulirane veličine se u svakom postrojenju može uticati na određeni način. Na taj način se vrijednost regulirane veličine može dovesti na zadanu vrijednost. Fizikalna veličina preko koje se može uticijecati na reguliranu veličinu ili neka njena mjera (npr. njena električna predstava) se naziva manipulativnom veličinom. Manipulativna veličina može biti npr.:

- pozicija ispusnog ventila na pneumatskom rezervoaru

- 
- pozicija hidrauličkog ventila
  - napon na grijanju u galvanskoj kupki
  - pozicija zatvarača na cjevovodu za rashladnu tečnost
  - pozicija ventila na dovodnim cjevovodima u hemijskom reaktoru
  - napon armature istosmjernog električnog motora.

#### *3.2.4. Objekat upravljanja*

Između zadane veličine i regulirane veličine se nalazi komplikovani sklop uređaja. Ovi uređaji su međusobno uvezani kroz fizikalne međuovisnosti zadane i regulirane veličine. Dio te međuovisnosti koji je od interesa za upravljanje predstavlja objekat upravljanja. Ova međuovisnost se može iskazati odgovarajućim formalnim opisom (matematičkim izrazima, grafičkim prikazom i sl.).

### **3.3. Sistem**

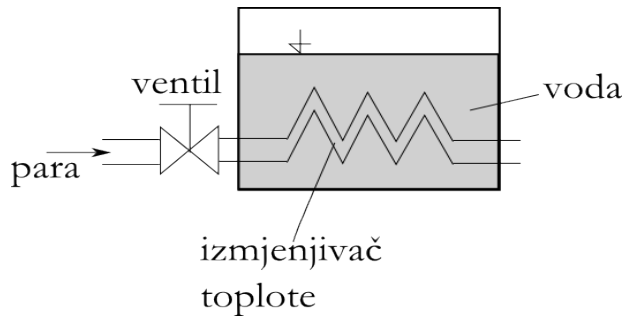
Sistem predstavlja dio postrojenja koji ima najmanje jednu ulaznu veličinu i jednu izlaznu veličinu. Ponašanje sistema se odražava u vezi između ulazne i izlazne veličine. Ovo ponašanje, odnosno veza između ulaznih i izlaznih veličina, se može opisati primjenom matematičkog aparata, na osnovu fizikalnih zakona. Isto tako, karakteristike ponašanja sistema se mogu odgonetnuti eksperimentalno. U automatskom upravljanju se sistemi najčešće predstavljaju grafički, pomoću blokova u okviru kojih je obuhvaćena veza između ulaznih i izlaznih veličina. Na slici 3.1. je predstavljen sistem sa jednim ulazom i jednim izlazom.



Slika 3.1. Blokovska predstava sistema

Kao primjer sistema posmatrajmo rezervoar u kome se nalazi voda, čija temperatura se mora održavati na konstantnoj vrijednosti. Voda se zagrijava

pomoću pare koja protiče kroz cijev protočnog izmjenjivača toplote. Protok pare se može regulirati pomoću ventila na ulazu u rezervoar. U ovom slučaju se objekat upravljanja nalazi između pozicije ventila za regulaciju protoka pare i temperature vode u rezervoaru. Drugim riječima, ovaj rezervoar sa izmjenjivačem toplote i pripadnim cjevovodima predstavlja sistem čija izlazna veličina je “temperatura vode”, a ulazna veličina “pozicija ventila” (slika 3.2.).



Slika 3.2. Rezervoar kao sistem

Unutar ovog sistema se odvijaju različiti procesi:

- pozicija ventila određuje protok pare kroz izmjenjivač toplote
- protok pare određuje količinu toplote koja će biti predata vodi
- temperatura vode raste kada je dovedena količina toplote veća od količine toplote koja se izgubi kroz zidove
- svi pomenuti procesi zajedno predstavljaju traženu ovisnost izlazne veličine sistema o ulaznoj.

Modeliranje sistema sa ulazima i izlazima i njegovo predstavljanje blokom ima prednost koja se ogleda u tome da se gube detalji tehničke izvedbe (uređaji, njihove veze i sl.), a da se postiže cjelovit pregled tehnološkog procesa. Uskoro ćemo vidjeti da objekti upravljanja potpuno različite fizikalne prirode imaju identično ponašanje, te se sa stanovišta automatskog upravljanja mogu i posmatrati na identičan način.

---

### ***3.4. Upravljanje u otvorenoj i zatvorenoj sprezi***

Nakon što je pojašnjen pojam sistema, nedostaje još nekoliko ustaljenih pojmova iz automatskog upravljanja. Poželjno je praviti razliku između pojmova upravljanja u otvorenoj i zatvorenoj sprezi.

Upravljanje u otvorenoj sprezi predstavlja djelovanje na sistem kod koga jedna ili više ulaznih veličina na osnovu zakonitosti datog sistema utiču na druge, izlazne veličine. (standard DIN 19226).

Osnovna karakteristika upravljanja je otvoreno djelovanje, tj. izlazne veličine ni na koji način ne utiču na ulazne veličine.

Kao primjer, posmatrajmo zapreminski protok koji se može podešavati pomoću ventila. Kod konstantnog pritiska prije ventila ovaj protok ovisi direktno o poziciji ventila. Do ove međuovisnosti se može doći kroz primjenu fizikalnih zakona ili eksperimentalno. Možemo uočiti da ventil predstavlja sistem, čija je izlazna veličina volumenski protok, a ulazna veličina pozicija.

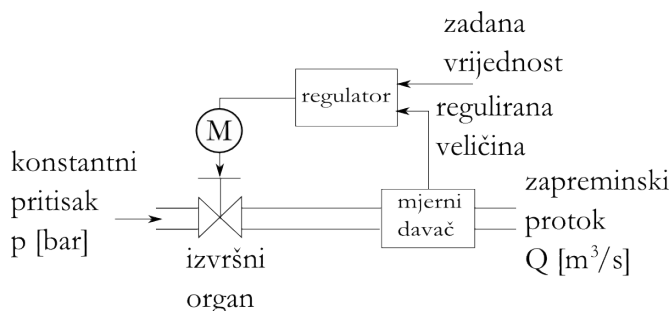
Ovakvim sistemom se može upravljati postavljanjem ventila u datu poziciju. Time se može postići željena vrijednost zapreminskog protoka.

Upravljanje u zatvorenoj sprezi je takvo djelovanje na sistem, kod koga se vrijednost regulirane veličine vraća i poredi sa zadanom vrijednošću. U ovisnosti o rezultatu poređenja, ulazna veličina u sistem djeluje tako da se izlazna veličina održi na zadanoj vrijednosti bez obzira na poremećaje. Ovakvim povratnim djelovanjem se dobija zatvorena kontura upravljanja. (standard DIN 19226).

Neka je potrebno da zapreminski protok iz prethodnog primjera, kao izlazna veličina sistema, prati zadane promjene. Radi toga je najprije potrebno izmjeriti vrijednost ovog protoka, i pretvoriti je u električni signal. Ovaj signal se onda vodi u regulator, gdje se poredi sa zadanom vrijednošću. Poređenje se vrši oduzimanjem izmjerene vrijednosti od zadane vrijednosti. Rezultat poređenja predstavlja

regulacionu grešku (regulaciono odstupanje), na osnovu kojega regulator određuje vrijednost ulazne veličine u sistem s ciljem otklanjanja postojećeg odstupanja od zadane vrijednosti.

Da bi se ventil postavio na poziciju koja je određena na osnovu regulacione greške, neophodno je da postoji neki električni motor kojim se može upravljati ili elektromagnet sa proporcionalnim djelovanjem. Pomoću njih će manipulativna veličina (pozicija ventila) biti postavljena na potrebnu vrijednost. Ova komponenta sistema automatskog upravljanja koja omogućuje postavljanje ulazne veličine sistema na potrebnu vrijednost se naziva izvršnim organom. Cijeli opisani sistem je predstavljen na slici 3.3.



Slika 3.3. Upravljanje u zatvorenoj sprezi

Regulator daje izvršnom organu regulacioni signal (regulaciono dejstvo) u ovisnosti o regulacionoj greški. Ukoliko je izmjerena vrijednost regulirane veličine veća od zadane vrijednosti, regulator će preko izvršnog organa pritisnuti ventil i time smanjiti vrijednost zapreminskog protoka. U suprotnom slučaju, ukoliko je izmjerena vrijednost manja od zadane vrijednosti, regulator će izvršnom organu dati signal koji će dovesti do otvaranja ventila, a time i povećanja vrijednosti protoka.

Međutim, regulirana veličina najčešće ne može idealno pratiti zadanu vrijednost:

- Ukoliko se djelovanje prečesto i prejako mijenja, to će imati pretjeran utjecaj na izlaznu veličinu sistema. Kao posljedica se mogu javiti oscilacije vrijednosti izlazne veličine.

- 
- Ukoliko je djelovanje presporo i slabo, izlazna veličina će samo grubo pratiti zadanu vrijednost, ili neće uopće biti u stanju da je prati.

Uz to, različiti sistemi odnosno različiti objekti upravljanja traže i različite načine upravljanja. Tako npr. sistemi koji imaju velika kašnjenja između djelovanja na ulazu i efekta tog djelovanja na izlazu moraju biti upravljani vrlo pažljivo. Iz toga proizlaze osnovni problemi i zadaci koje rješavaju stručnjaci iz oblasti automatskog upravljanja.

Da bi se kreiralo upravljanje nekom veličinom unutar postrojenja, neophodno je ispuniti sljedeće:

- utvrditi manipulativne veličine,
- odrediti ponašanje objekta upravljanja,
- naći odgovarajući zakon upravljanja,
- izabrati odgovarajući davač vrijednosti reguliranog signala i izvršni organ.

### ***3.5. Osnovni pojmovi teorije automatskog upravljanja***

Nakon što je na primjeru upravljanja zapreminskim protokom pojašnjena razlika između upravljanja u otvorenoj i zatvorenoj sprezi, te predstavljanja osnovnog principa upravljanja i osnovnih pojmova koji se susreću u upravljanju, sada će ti pojmovi biti tačnije i detaljnije definirani.

#### ***3.5.1. Regulirana veličina***

Cilj upravljanja je da se neka veličina održi na zadanoj vrijednosti, ili da prati zadanu promjenu. Ovu veličinu ćemo zvati reguliranom veličinom i označiti sa  $x$ . Na primjeru upravljanja zapreminskim protokom, regulirana veličina je bila zapreminski protok.

---

### 3.5.2. Manipulativna veličina

Da bi bilo moguće upravljanje nekom veličinom (reguliranom veličinom  $x$ ), neophodno je da na procesu ili postrojenju postoji mogućnost uticaja na vrijednost regulirane veličine  $x$ . Veličina preko koje se može utjecati na vrijednost regulirane veličine se naziva manipulativnom veličinom. Manipulativnu veličinu ćemo označiti sa  $m$ . U pomenutom primjer manipulativna veličina je bila pozicija ventila, odnosno struja pokretanja elektromagneta koji pogoni ventil.

### 3.5.3. Poremećaj

Na svaki sistem, pa tako i na sistem upravljanja, djeluju poremećaji. Postojanje poremećaja je najčešće osnovni razlog gradnje sistema upravljanja i regulacije. Tako u prethodnom primjeru, kada iz nekog razloga dođe do promjene pritiska ispred ventila, da bi se zapreminski protok održao na zadanoj vrijednosti neophodno je promijeniti poziciju ventila. Ovi nekontrolirani utjecaji se nazivaju poremećajima. Poremećaj označavamo sa  $z$ .

Objekat upravljanja je dio regulirane mašine ili postrojenja kod koje vrijednosti regulirane veličine moraju održati zadane vrijednosti. Objekat upravljanja se može predstaviti kao sistem kod kojega izlaznu veličinu predstavlja regulirana veličina, a ulaznu veličinu manipulativna veličina. Za pomenuti primjer upravljanja zapreminskim protokom smo vidjeli da je sistem predstavljao cjevovod sa regulacionim ventilom.

### 3.5.4. Zadana vrijednost

Zadana vrijednost predstavlja veličinu koju treba pratiti regulirana veličina i ona daje željenu vrijednost regulirane veličine. Zadana vrijednost može biti konstantna u vremenu, ili se može mijenjati, a označićemo je sa  $x_r$ .



---

### 3.5.5. Regulatorna greška

Regulatornom greškom nazivamo razliku između zadane vrijednosti i vrijednosti regulirane veličine:

$$e = x_z - x$$

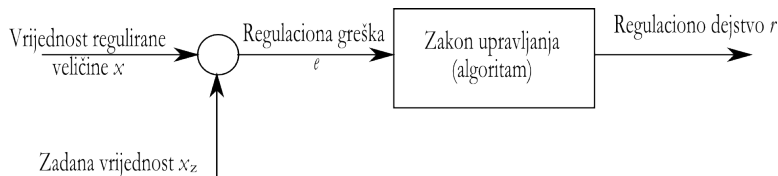
Regulatorna greška predstavlja osnovni signal na osnovu kojega se određuje vrijednost manipulativne veličine.

### 3.5.6. Upravljački zakon

Upravljački zakon predstavlja način na koji regulatorna kontura reagira na poremećaj. Određivanje upravljačkog zakona predstavlja jedan od osnovnih zadataka sinteze upravljanja.

### 3.5.7. Regulator

Regulator ima zadatak da održava vrijednost regulirane veličine što je moguće bliže zadanoj vrijednosti, bez obzira na promjene zadane vrijednosti i djelovanje poremećaja. U regulatoru se na osnovu regulatorne greške  $e$  i u skladu sa upravljačkim zakonom određuje vrijednost regulatornog dejstva (slika 3.4.). Regulatorno dejstvo preko izvršnog organa daje vrijednost manipulativne veličine, koja utiče na vrijednost regulirane veličine.



Slika 3.4. Princip rada regulatora

---

### *3.5.8. Izvršni organ i pogon izvršnog organa*

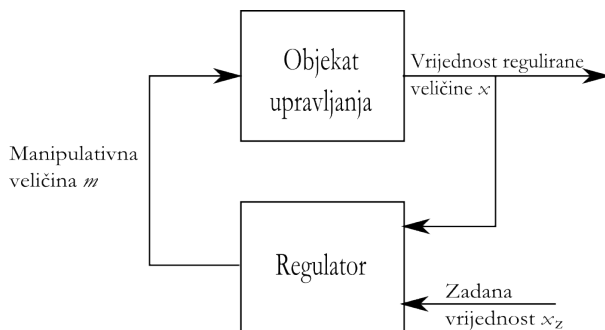
Izvršni organ mijenja manipulativnu veličinu i time utiče na vrijednost regulirane veličine. Obzirom da regulaciono dejstvo (signal) koje na svom izlazu daje regulator najčešće nije u stanju direktno preko izvršnog organa djelovati na manipulativnu veličinu (bilo zbog toga što su prirode manipulativne veličine i regulacionog dejstva različite, bilo da regulaciono dejstvo energetski nije u stanju da utiče na sistem ili proces), izvršni organ posjeduje određeni pogon preko koga se mijenja manipulativna veličina. Tako npr. regulaciono dejstvo preko određenog pojačala može djelovati na motor, koji predstavlja pogon ventila kao izvršnog organa. Ovaj motor mijenja poziciju ventila, koja predstavlja manipulativnu veličinu.

### *3.5.9. Mjerni član*

Regulirana veličina i ulazna veličina u regulator najčešće predstavljaju različite fizikalne veličine. Tako je u prethodnom primjeru regulirana veličina bila zapreminski protok ( $\text{m}^3/\text{s}$ ), dok je ulazna veličina u regulator najčešće neki standardni električni (ili eventualno pneumatski) signal. Postoje različiti standardni električni naponski (0 do 5 V, 0 do 10 V, -5 do 5 V, -10 do 10 V), strujni (0 do 20 mA, 4 do 20 mA) i frekventni signali. Da bi se fizikalna veličina koja predstavlja reguliranu veličinu mogla dovesti na ulaz regulatora, potrebno je izvršiti njenu konverziju u neki od standardnih signala. Ova konverzija je osnovna funkcija mjernog člana (mjernog davača, senzora). Tek nakon konverzije regulator je u stanju procesirati informaciju o vrijednosti regulirane veličine.

### *3.5.10. Regulaciona kontura*

Regulaciona kontura sadrži sve komponente koje omogućavaju zatvoreni krug djelovanja (slika 3.6.). Na slici nisu prikazani izvršni organ i mjerni član.



Slika 3.5. Blokovska predstava regulacione konture

### 3.6. Objekti upravljanja

Objekat upravljanja je dio mašine ili procesa u okviru kojeg se treba vrijednost regulirane veličine dovesti na zadanu vrijednost, bez obzira na promjene zadane vrijednosti ili utjecaje poremećaja. Ulazi u objekat upravljanja su manipulativna veličina i poremećaji. Obzirom da poremećaji najčešće nisu poznati, jedan od zadataka upravljanja je generisanje takve vrijednosti manipulativne veličine da se dobije željena vrijednost regulirane veličine, bez obzira na djelovanje nepoznatih poremećaja.

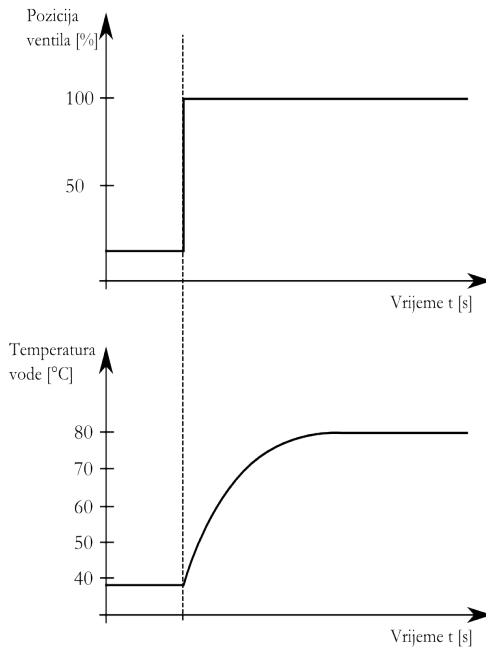
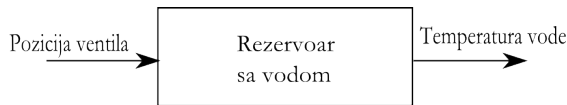
Da bi se odredio i podesio odgovarajući regulator za neki objekat upravljanja, neophodno je poznavati karakteristike objekta upravljanja. Pri tome za sintezu upravljanja nije od interesa interna tehnološka realizacija objekta upravljanja, nego isključivo sistemske karakteristike njegovog ponašanja.

#### 3.6.1. Vremenski odziv sistema

Karakteristike ponašanja sistema u vremenu su od naročitog interesa za sintezu upravljanja. Ove karakteristike ponašanja se nazivaju dinamičkim karakteristikama sistema. Preciznije, dinamičke karakteristike predstavljaju vremensku promjenu izlaznih veličina (reguliranih veličina) pri promjeni ulaznih (prije svega manipulativnih) veličina.

Pri tome treba imati na umu da svaki sistem ima specifično dinamičko ponašanje.

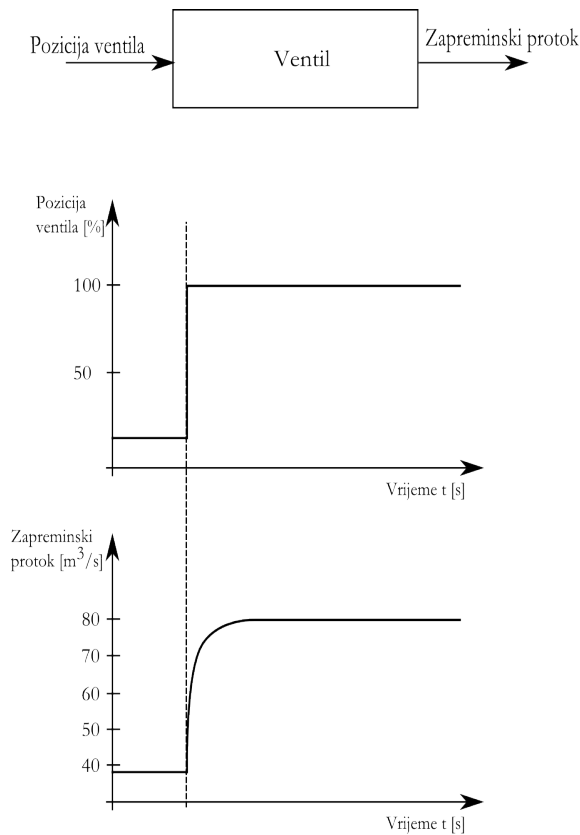
Kao primjer, kod pomenutog izmjenjivača toplote sa slike 3.2. promjena pozicije ventila na dovodnom cjevovodu za paru nije kao posljedicu imala momentalnu promjenu temperature vode u rezervoaru. Drugim riječima, temperatura vode kao izlazna veličina nije trenutno slijedila promjene zadane vrijednost, nego je zbog određenog toplotnog kapaciteta tečnosti u rezervoaru nakon skokovite promjene pozicije ventila trebalo proteći neko vrijeme dok temperatura tečnosti ne dospije na novu stacionarnu vrijednost. Dakle, osnovna karakteristika dinamičkog ponašanja ovakvog sistema je postojanje kašnjenja (slika 3.6.).



Slika 3.6. Vremenski odziv rezervoara sa vodom

Na slici se vidi da vrijeme potrebno da bi temperatura vode u rezervoaru dostigla novu stacionarnu vrijednost nije zanemarivo.

Slično dinamičko ponašanje ima i sistem iz drugog primjera, gdje se pozicijom ventila upravljalo zapreminskim protokom gasa (slika 3.7.). Međutim, iako su osnovne karakteristike dinamičkog ponašanja ovog sistema identične, vrijeme potrebno za postizanje nove stacionarne vrijednosti je daleko kraće.

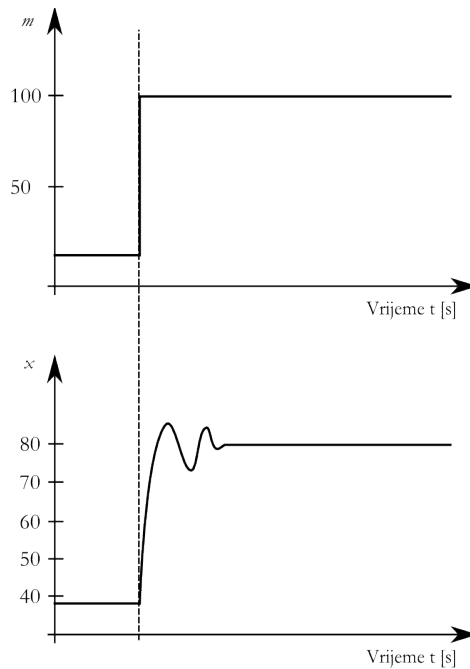


Slika 3.7. Vremenski odziv ventila

### 3.6.2. Dinamičke karakteristike sistema

U oba prethodna primjera je očigledno da smo posmatrali vremenski odziv izlazne veličine sistema kada se na njegov ulaz dovede skokovita promjena ulazne veličine. Ovo je vrlo često korištena metoda za ispitivanje i predstavljanje dinamičkog

ponašanja sistema. Ovakav vremenski odziv sistema se naziva odskočnim odzivom i karakterizira ponašanje svakog sistema. Na osnovu odskočnog odziva je moguće doći i do matematičkog opisa sistema. Još jedan primjer odskočnog odziva sistema je dat na slici 3.8. Vidi se da sistem novo stacionarno stanje postiže oscilirajući prigušeno neko vrijeme.



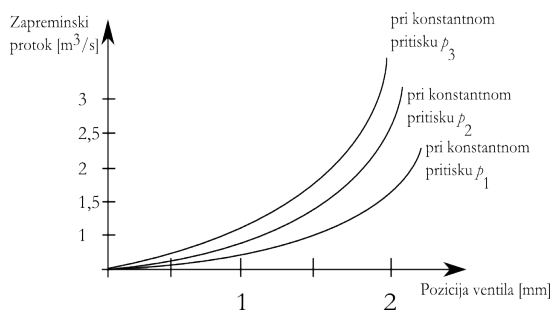
Slika 3.8. Prigušeni oscilatorni odziv sistema

### 3.6.3. Statičke karakteristike sistema

Drugi vrlo važan vid predstavljanja ponašanja sistema su statičke karakteristike sistema, koje predstavljaju ponašanje sistema u stacionarnom stanju. Sistem se nalazi u stacionarnom stanju onda kada se ni jedna veličina ne mijenja u vremenu. Ukoliko nema promjena ulaznih veličina, sistem će ostati u stacionarnom stanju beskonačno dugo.

I u stacionarnom stanju postoji ovisnost izlaznih veličina sistema o ulaznim veličinama i ova ovisnost se predstavlja statičkim karakteristikama sistema. Na slici

3.9. je predstavljena statička karakteristika ventila za regulaciju zapreminskog protoka. Na ovoj karakteristici se može uočiti ovisnost zapreminskog protoka o poziciji ventila.



Slika 3.9. Statičke karakteristike ventila

Pri tome, različite vrijednosti pritiska ispred ventila daju i različite ovisnosti zapreminskog protoka o poziciji ventila. Radi toga se za neki sistem najčešće daje ili određuje familija statičkih karakteristika, pri čemu je neki parametar (kao što je na slici 3.9. bio slučaj sa pritiskom) uzimao vrijednosti iz diskretnog skupa.

### 3.7. Regulator

U prethodnom odjeljku je bilo riječi o objektu upravljanja kao dijelu postrojenja, koje primjenom tehnika upravljanja treba biti stavljen pod kontrolu. Pri tome regulator, kao dio regulacione konture, igra ključnu ulogu.

Regulator predstavlja komponentu regulacione konture koja na osnovu poređenja zadane vrijednosti i mjerene vrijednosti regulirane veličine određuje regulaciono djelovanje i njime djeluje na sistem kojim se upravlja. U prethodnim odjeljcima je jasno pokazano da objekti upravljanja (sistemi) mogu imati vrlo različita ponašanja. Tako postoje brzi objekti (sistemi), objekti (sistemi) sa velikim transportnim kašnjenjem, kao i objekti (sistemi) sa dinamičkim kašnjenjem kod kojih nije moguća trenutna promjena regulirane veličine.

---

Na svaki od ovih objekata se mora djelovati na različit način da bi se postigli ciljevi upravljanja. Radi toga i postoje različiti tipovi regulatora, koji djeluju na različit način. Jedan od osnovnih zadataka projektanta sistema upravljanja se sastoji u određivanju optimalnog zakona upravljanja za dati objekt upravljanja.

### *3.7.1. Tipovi regulatora*

Regulatori se mogu ponašati na različit način, odnosno na različit način mogu davati regulaciono dejstvo kao izlazni signal na osnovu regulacione greške kao ulaznog signala. Postoje dvije osnovne klase regulatora:

- regulatori sa kontinualnom vrijednošću izlaza

Regulatori sa kontinualnom vrijednošću izlaza na svom izlazu daju kontinualno promjenljiv signal, ovisan o regulacionoj greški.

- regulatori sa diskretnim vrijednostima izlaza

Kod regulatora sa diskretnim vrijednostima izlaza se kao izlazni signal mogu pojaviti samo vrijednosti iz konačnog skupa unaprijed određenih vrijednosti. Najpoznatiji regulatori iz ove klase su dvopoložajni regulatori, koji na svom izlazu daju samo regulaciona dejstva “uključeno” i “isključeno”. Primjer ovakvog regulatora je termostat kakav se susreće u pegli, bojleru, grijalici i sl. Kada je regulaciona greška pozitivna, odnosno kada je zadana vrijednost temperature veća od trenutne vrijednosti temperature kao regulirane veličine, termostat daje regulaciono dejstvo “uključeno”, čime se dovodi struja na grijač, što prouzrokuje povećanje temperature. Kada je regulaciona greška negativna, odnosno kada je trenutna vrijednost temperature premašila zadanu vrijednost temperature, termostat daje regulaciono dejstvo “isključeno”, odnosno prekida se strujno kolo grijača. Radi toga, zbog gubitaka toplote, temperatura postepeno opada.



---

I dok se dvopoložajni regulatori jako često susreću u aparatima koji se koriste u širokoj potrošnji, regulatori koji se koriste u složenijim sistemima, mašinama, te procesima različite prirode, su u najvećem broju slučajeva kontinualni. Upravo ovi regulatori se nalaze u fokusu teorije automatskog upravljanja i u nastavku će biti pojašnjeni osnovni tipovi kontinualnih regulatora.

Svaki objekat upravljanja posjeduje svoj karakterističan vremenski odziv. Ovaj vremenski odziv je praktično određen konstrukcijom mašine, postrojenja ili tehnologijom procesa, i najčešće ga nije moguće modificirati. Vremenski odziv nekog sistema se može odrediti eksperimentalno ili teorijskom analizom. Regulator takođe predstavlja dinamički sistem, koji ima svoj karakteristični vremenski odziv. Međutim, za razliku od vremenskog odziva sistema kojim se upravlja, vremenski odziv regulatora određuje projektant sistema automatskog upravljanja i to na taj način da sistem u cjelini zadovolji tražene performanse. Drugim riječima, vremenski odziv cjelokupnog sistema regulacije treba ispuniti postavljene zahtjeve.

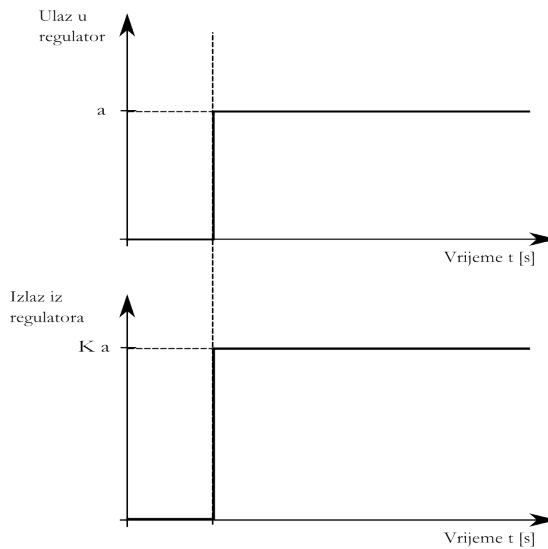
Regulaciono dejstvo klasičnih kontinualnih regulatora se sastoji od tri osnovne komponente:

- proporcionalnog djelovanja (P-djelovanja),
- integralnog djelovanja (I-djelovanja),
- diferencijalnog djelovanja (D-djelovanja).

Ove tri osnovne komponente djelovanja regulatora proizlaze iz načina na koji se određuje regulaciono dejstvo, na osnovu ulaznog signala u regulator.

### *3.7.2. Regulator sa proporcionalnim dejstvom*

Kod proporcionalnog regulatora je vrijednost regulacionog dejstva proporcionalna vrijednosti ulaznog signala u regulator (slika 3.10.). Kada je regulaciona greška velika, i regulaciono dejstvo će biti veliko. U protivnom, kada je regulaciona greška mala, i regulaciono dejstvo će biti malo.



Slika 3.10. Vremenski odziv proporcionalnog regulatora na step ulazni signal

Na slici se može uočiti da vremenski odziv proporcionalnog regulatora idealno prati promjene ulaznog signala u regulator (regulacione greške):

$$r(t) = \begin{cases} K \cdot a, & t \geq t_0 \\ 0, & t < t_0 \end{cases}$$

Osnovni parametar proporcionalnog regulatora je pojačanje regulatora  $K$ , koje se može izračunati kao količnik vrijednosti izlaznog i ulaznog signala regulatora.

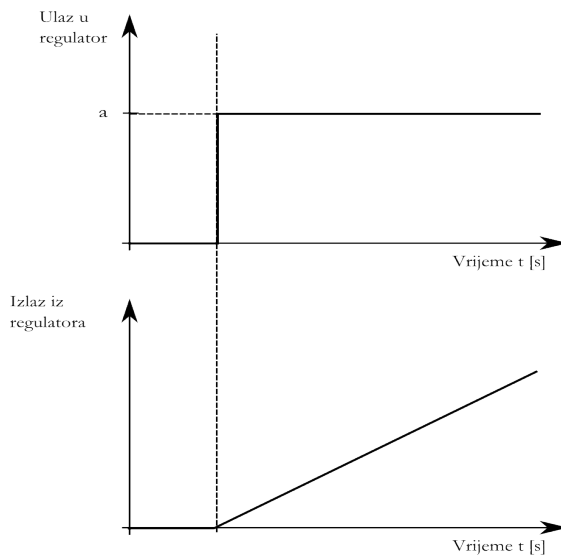
Dobra strana proporcionalnog regulatora je njegova jednostavna realizacija i jednostavno podešavanje, obzirom da posjeduje samo jedan parametar. Međutim, loša strana proporcionalnog regulatora je to što on daje regulaciono dejstvo samo u slučaju da postoji nenulta vrijednost regulacione greške. Radi toga nije moguće postići da sistem ima regulacionu grešku jednaku nuli ukoliko se koristi samo proporcionalni regulator, jer u tom slučaju regulator ne daje nikakvo regulaciono dejstvo, te će vrijednost regulirane veličine neizbježno pod utjecajem poremećaja otklizati sa zadane vrijednosti. Dakle, ukoliko se koristi samo proporcionalni regulator, uvijek će postojati neka preostala vrijednost regulacione greške koju ovaj

---

regulator nije u stanju ukloniti.

### 3.7.3. Regulator sa integralnim dejstvom

Ukoliko regulator ima integralno dejstvo, njegovo regulaciono djelovanje će predstavljati signal regulacione greške sumiran tokom vremena (odnosno integrirana regulaciona greška) (slika 3.11.). To znači da za kratkotrajna odstupanja od zadane vrijednosti integralni regulator neće dati znatno regulaciono dejstvo. Međutim, ukoliko regulaciona greška traje duže vremena, regulaciono dejstvo će postepeno porasti i neće iščeznuti sve dok regulaciona greška ne opadne na nultu vrijednost.



Slika 3.11. Vremenski odziv integralnog regulatora na step ulazni signal

Sa slike se vidi da kod integralnog regulatora vrijednost regulacionog dejstva nije proporcionalna regulacionoj greški, nego je brzina promjene regulacionog dejstva proporcionalna regulacionoj greški. Tako, ako je do skokovite promjene regulacione greške došlo u trenutku  $t_0$ , vremenska promjena regulacionog dejstva se može odrediti kao:

---

$$r(t) = \frac{1}{T_i} \cdot \int_{t_0}^{\infty} a \, dt \quad ,$$

odnosno prenosna funkcija I- regulatora je data sa:

$$G_r(s) = \frac{1}{T_i \cdot s}$$

Regulator sa integralnim dejstvom je pogodan za uklanjanje preostale regulacione greške. Ako se regulaciona greška jako promijeni, regulaciono dejstvo počinje rasti velikom brzinom. Kao posljedica, regulaciona greška počinje postepeno opadati, što dovodi i do postepenog smanjivanja regulacionog dejstva, sve do postizanja nulte vrijednosti regulacione greške.

Međutim, treba imati na umu da regulator gotovo nikada ne posjeduje samo integralno dejstvo. Ovo dejstvo je nepogodno za veliki broj objekata upravljanja koji posjeduju velika kašnjenja. Usljed toga regulator sa integralnim dejstvom ovakve objekte može dovesti u stanje osciliranja.

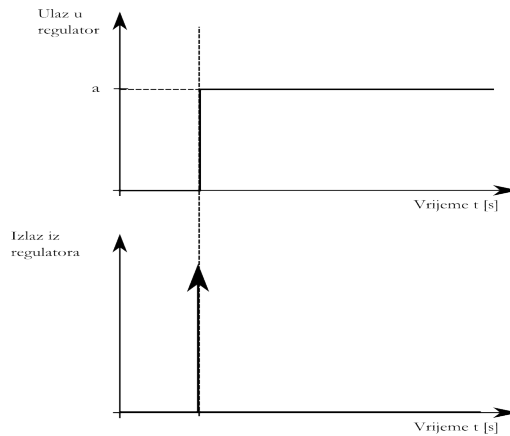
#### *3.7.4. Regulator sa derivativnim dejstvom*

Regulator sa derivativnim dejstvom se koristi za uklanjanje brzo promjenljivih vrijednosti regulacione greške. Kod ovog regulatora se prati brzina promjene regulacione greške, i na osnovu nje se određuje vrijednost regulacionog dejstva. Drugim riječima, signal regulacione greške se diferencira. Ukoliko se signal regulacione greške brzo mijenja, regulator sa derivativnim dejstvom će trenutno dati veliku vrijednost regulacionog dejstva. Vremenski odziv idealnog regulatora sa derivativnim dejstvom je dat na slici 3.12.

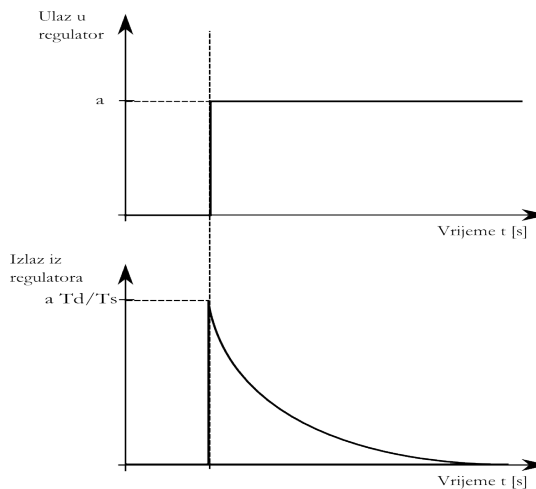
Na slici je predstavljen odziv idealnog diferencijatora na step ulazni signal. Već iz prirode prikazanog odziva se vidi da idealni diferencijator ne postoji, nego se derivativno dejstvo uvodi preko realnog diferencijatora. Na slici 3.13. je predstavljen odziv realnog derivativnog regulatora, sa koje se vidi da je amplituda skoka vrijednosti izlaza regulatora konačna i ovisna o parametrima regulatora, te

postepeno eksponencijalno opada prema nuli. Vremenski odziv realnog diferencijatora na step ulazni signal i njegova prenosna funkcija su:

$$r(t) = \frac{T_d}{T_s} \cdot a \cdot e^{-\frac{t}{T_d}}, \quad G_r(s) = \frac{T_d \cdot s}{T_s \cdot s + 1}$$



Slika 3.12. Vremenski odziv idealnog derivativnog regulatora na step ulazni signal



Slika 3.13. Vremenski odziv realnog derivativnog regulatora na step ulazni signal

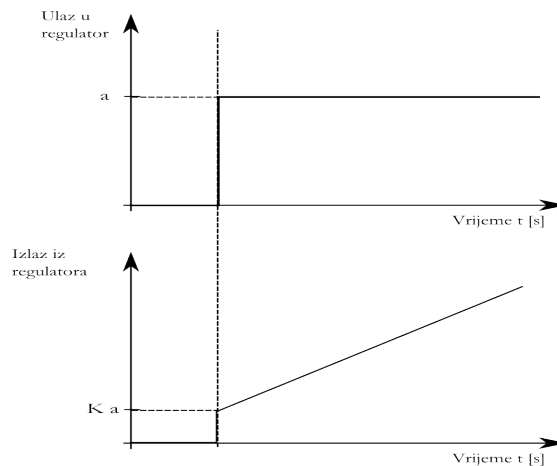
Međutim, primjena regulatora koji posjeduje samo derivativno dejstvo nema smisla, jer regulaciono djelovanje postoji samo u slučaju da postoji promjena vrijednosti regulacione greške. Znači da ovaj regulator ne bi davao nikakvo regulaciono dejstvo ukoliko bi regulaciona vrijednost imala nenultu, ali konstantnu vrijednost. Zato se

ovakvo dejstvo pojavljuje isključivo u kombinaciji sa nekim od gore navedenih (proporcionalnim, integralnim ili u kombinaciji sa oba).

### 3.7.5. PI regulator

PI regulator predstavlja regulator sa kombiniranim proporcionalnim i integralnim djelovanjem. Na taj način se koriste dobre osobine oba ova regulatora: brzo reagiranje na pojavu regulacione greške, te uklanjanje preostale vrijednosti regulacione greške. PI regulator se koristi u velikom broju slučajeva, a osim pojačanja proporcionalnog djelovanja, posjeduje još jedan parametar, koji određuje intenzitet djelovanja integralne komponente. Vremenski odziv PI regulatora je predstavljen na slici 3.14., a izraz na osnovu kojega formira regulaciono dejstvo i njegova prenosna funkcija su najčešće dati u formi:

$$r(t) = K \cdot \left( e(t) + \frac{1}{T_i} \cdot \int_{t_0}^{\infty} e(t) dt \right), \quad G_r(s) = K \cdot \left( 1 + \frac{1}{T_i \cdot s} \right)$$



Slika 3.14. Vremenski odziv PI regulatora na step ulazni signal

Može se reći da parametar  $T_i$  određuje koliko je PI regulator brži od čistog integralnog regulatora. Što je  $T_i$  manje, integralna komponenta u regulacionom dejstvu raste brže. Suprotno, što je  $T_i$  veće, integralna komponenta je manja i za

---

$T_i \rightarrow \infty$  ponašanje PI regulatora se približava ponašanju čistog P regulatora.

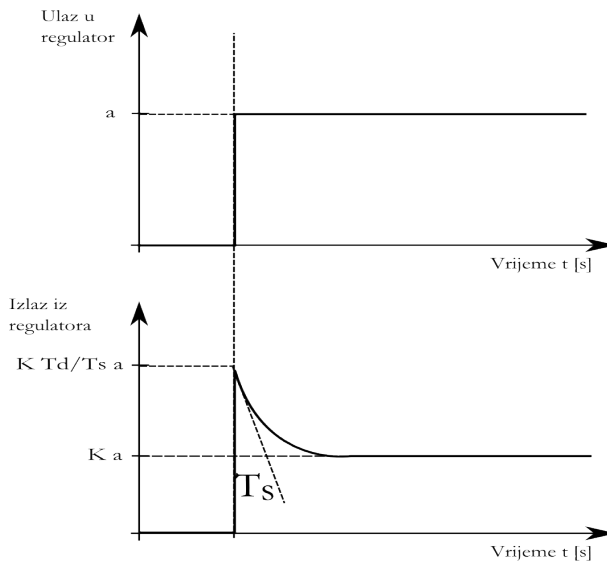
Djelovanje PI regulatora je veće što je vrijednost proporcionalnog pojačanja  $K$  veća, a vrijednost parametra  $T_i$  manja. Međutim, ukoliko se P i I dejstvo regulatora previše povećaju, može doći do pojave oscilacija regulirane veličine, odnosno do pojave nestabilnosti. Za svaku regulacionu konturu su uslovi pojave oscilatornog ponašanja (nestabilnosti) različiti i potrebno ih je odrediti, da bi se parametri regulatora mogli podesiti na vrijednosti koje garantuju kvalitetno ponašanje konture.

### 3.7.6. PD regulator

PD regulator kombinira proporcionalno i derivativno djelovanje. Pri tome proporcionalno djelovanje omogućava da regulator daje regulaciono dejstvo u skladu sa veličinom regulacione greške. S druge strane, derivativno djelovanje omogućava da regulator reagira na promjenu regulacione greške, te da regulaciono dejstvo koje će utjecati da regulaciona greška ne poprimi velike vrijednosti. U praktičnoj primjeni se ne koristi čisto derivativno dejstvo (tzv. idealni diferencijator), nego njegova modifikacija (tzv. realni diferencijator). Za ovo postoje razlozi tehničke (konstruktivne) i praktične prirode. Regulaciono dejstvo PD regulatora za step signal na ulazu i njegova prenosna funkcija su najčešće date u formi:

$$r(t) = K \cdot \left( a + \frac{T_d}{T_s} \cdot a \cdot e^{-\frac{t}{T_s}} \right), \quad G_r(s) = K \cdot \left( 1 + \frac{T_d \cdot s}{T_s \cdot s + 1} \right)$$

Vremenski odziv PD regulatora na step ulazni signal je dat na slici 3.15. Treba napomenuti da se PD regulator ne susreće često u upotrebi iz dva osnovna razloga. Prvi razlog je što nije u stanju ukloniti preostalu vrijednost regulacione greške, a drugi razlog je da postoji opasnost derivativna komponenta u regulacionom dejstvu, ukoliko pažljivo određena, sistem dovede do nestabilnosti.



Slika 3.15. Vremenski odziv PD regulatora na step ulazni signal

### 3.7.7. PID regulator

PID regulator kombinira sva do sada pomenuta dejstva u okviru jedinstvenog regulatora. Vremenski odziv PID regulatora je predstavljen na slici 3.16., a regulaciono dejstvo za step signal na ulazu se formira najčešće na osnovu izraza:

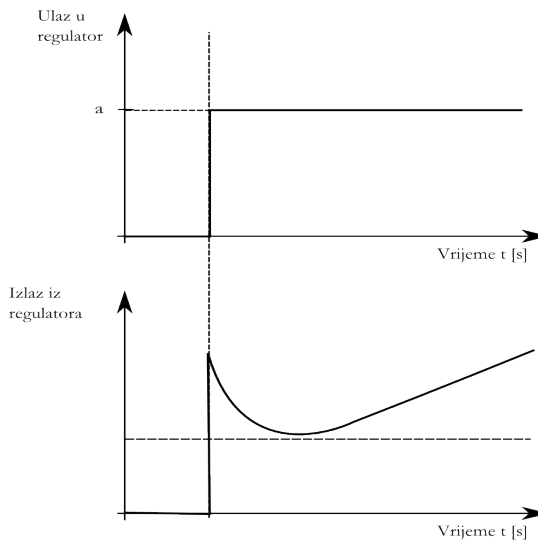
$$r(t) = K \cdot \left( a + \frac{1}{T_i} \cdot \int_{t_0}^{\infty} a \, dt + \frac{T_d}{T_s} \cdot a \cdot e^{-\frac{t}{T_s}} \right),$$

a njegova prenosna funkcija je:

$$G_r(s) = K \cdot \left( 1 + \frac{1}{T_i \cdot s} + \frac{T_d \cdot s}{T_s \cdot s + 1} \right)$$

PID regulator predstavlja standardni industrijski regulator, čiji parametri su pojačanje  $K$ , te parametri integralnog i derivativnog člana  $T_i$ ,  $T_d$  i  $T_s$ . Podešavanje PID regulatora se sastoji u određivanju vrijednosti ovih parametara za koje se postiže optimalno ponašanje regulacione konture kao cjeline.





Slika 3.16. Vremenski odziv PID regulatora na step ulazni signal

### 3.7.8. Tehnička izvedba regulatora

U prethodnim odjeljcima su navedene osobine osnovnih tipova regulatora, te data njihova formalna predstava. Tehnička realizacija datih izraza može biti različita: mehanička, pneumatska, električna, elektronička i sl. Iako se i danas mogu susresti mehaničke i pneumatske izvedbe regulatora, može se slobodno reći da u savremenom automatskom upravljanju jedino električne i elektroničke izvedbe regulatora igraju značajnu ulogu.

Električni/elektronički regulatori rade sa električnim ulaznim i izlaznim signalima. Radi toga je neophodno da se u regulacionoj konturi koristi odgovarajući mjerni pretvarač fizikalne veličine u električnu, te izvršni organ za pretvaranje električnog regulacionog signala u manipulativnu veličinu.

Da bi se pojednostavilo povezivanje mjernih pretvarača i izvršnih organa različitih proizvođača sa industrijskim regulatorima, definirane su standardne vrijednosti strujnih i naponskih signala, što je dato u tabeli 3.1.

| <i>Vrsta signala</i> | <i>Opseg vrijednosti</i>                              |
|----------------------|-------------------------------------------------------|
| Strujni              | 0 ... 20 mA<br>4 ... 20 mA                            |
| Naponski             | 0 ... 5 V<br>0 ... 10 V<br>-5 ... 5 V<br>-10 ... 10 V |

Tabela 3.1. Standardni električni signali

Unutar regulatora, signali se mogu procesirati na različite načine. Standardni kontinualni industrijski regulatori su najčešće elektroničke izvedbe, uz upotrebu operacionih pojačavača. Moderni industrijski regulatori koji se susreću danas su gotovo isključivo realizirani kao mikroračunarski sistemi na bazi mikroprocesora.

Osnovna razlika između analogne i digitalne realizacije regulatora je sljedeća:

- Kod regulatora na bazi operacionih pojačavača se naponski ili strujni signal direktno prevodi u odgovarajuću formu regulacionog signala.
- Kod digitalnih regulatora se analogni ulazni signali pretvaraju u digitalne signale. Digitalni signali se procesiraju primjenom odgovarajućeg algoritma i formira se digitalna forma regulacionog dejstva. Ovaj digitalni signal se zatim na odgovarajući način pretvara u analogni signal koji predstavlja regulaciono dejstvo.

Iako su analogna i digitalna tehnologija realizacije sa stanovišta teorije bitno različite, ne postoje razlike u izvedbama regulatora sa praktične strane.

---

## 4. Komponente sistema automatskog upravljanja

### 4.1. Uvod

U ovom poglavlju će ukratko biti predstavljene neke od komponenata sistema automatskog upravljanja koji će se koristiti i razrađivati u okviru kursa.

### 4.2. Optoelektronski mjerač protoka

U okviru laboratorijskog modela FESTO Didactic DD 3100 koji se koristi za eksperimente u okviru kursa se za mjerenje protoka upotrebljava optoelektronski mjerač protoka IR-Opflow. IR-Opflow je turbinski mjerač na bazi Peltonove turbine, što ga čini vrlo tačnim linearnim uređajem. Ovaj tip mjerača protoka je vrlo raširen u upotrebi u medicini, farmaciji, hemijskoj industriji, proizvodnji papira, poluprovodnika, biotehnologiji itd.

Izgled optoelektronskog mjerača protoka je dat na slici 4.1.



Slika 4.1. Optoelektronski mjerač protoka IR-Opflow

---

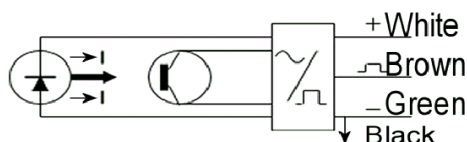
#### 4.2.1. Princip rada

Fluid protiče kroz mjerlač nailazeći prvo kroz helikoidnu mlaznicu, koja prouzrokuje spiralan rotirajući tok. Nakon toga fluid udara lopatice rotora turbine i prouzrokuje njeno okretanje. Rotor se okreće u struji fluida, bez fizičkog kontakta sa statičkim dijelom turbine, tako da se izbjegava otpor trenja. Isto tako, obzirom da rotor nije fiksiran za statički dio turbine ležajevima, nema habanja što produžava životni vijek turbine. Infracrveni optoelektronski predajnik i prijemnik su fiksirani unutar tijela mjerlača, zajedno sa parom minijaturnih elektronskih sklopova za stabilizaciju napona. Okretanje turbine prouzrokuje prekide linije prostiranja infracrvene svjetlosti između predajnika i prijemnika, što se na izlazu mjerlača manifestira pojavom slijeda impulsa čija frekvencija ovisi o brzini obrtanja turbine. Obzirom da brzina obrtanja turbine ovisi o protoku fluida, frekvencija impulsa je direktno proporcionalna protoku.

#### 4.2.2. Karakteristike i način priključivanja

Na tržištu se može naći više različitih tipova IR-Opflow mjerlača protoka, ovisno o opsegu vrijednosti protoka koji mogu mjeriti. IR-Opflow Type 3, koji se nalazi na laboratorijskom modelu DD 3100, može mjeriti protoke od 0,5 do 15 l/min. Pri tome se na izlaznim kontaktima dobija 3200 impulsa po litru fluida. Za navedeni opseg vrijednosti protoka, frekvencija izlaznih impulsa se kreće od 26,66 Hz do 800 Hz. Linearnost karakteristike mjerlača se kreće oko 1%.

Električni dijagram za povezivanje IR-Opflow mjerlača protoka je dat na slici 4.2. Mjerlač se napaja sa 5-24 V=, uz maksimalno opterećenje od 30 mA.

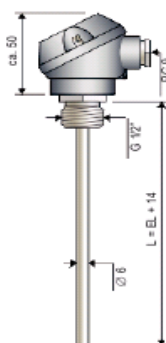


Slika 4.2. Optoelektronski mjerlač protoka IR-Opflow

---

### 4.3. Otpornički senzor temperature Pt 100

Senzor temperature Pt 100 predstavlja standardni industrijski senzor temperature. Omogućava vrlo tačno mjerenje temperature, na osnovu ovisnosti otpora platine o temperaturi. Na temperaturi  $0^{\circ}\text{C}$  ima nominalni otpor od  $100\ \Omega$ , a omogućava mjerenje temperatura u opsegu od  $-254,3^{\circ}\text{C}$ , pa sve do  $+1100^{\circ}\text{C}$ . Tačnost senzora (koja može biti manja od  $0,1^{\circ}\text{C}$ ) ovisi o materijalu od kojeg je senzor napravljen, ali i o metodi mjerenja koja se koristi. Otpornički senzori na bazi platine su vrlo dugotrajni, otporni na koroziju i karakteristika im je vrlo stabilna tokom eksploatacije. Izgled jedne vrste senzora Pt 100 je dat na slici 4.3.



Slika 4.3. Pt 100 senzor temperature

#### 4.2.1. Princip rada

Princip mjerenja temperature senzorom Pt 100 se zasniva na činjenici da se električni otpor platine mijenja proporcionalno temperaturi. U prosjeku, promjena otpora sa temperaturom iznosi  $0,35\ \Omega/\text{K}$  ( $0,39\ \Omega/\text{K}$  na  $0^{\circ}\text{C}$ ) i nije linearna. Nelinearnost se može izbjeći korištenjem odgovarajuće mjerne metode.

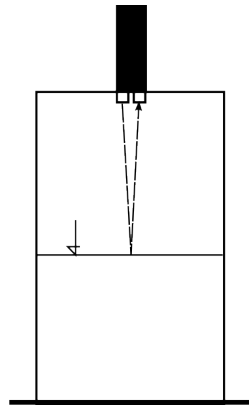
Promjena otpora senzora Pt 100 se mjeri elektronički i pretvara se u pojačani signal proporcionalan temperaturi. Za mjerenje otpora se koriste različite metode, najčešće na bazi Wheatstone-ovog mosta. Obzirom da je promjena električnog otpora od  $0,35\ \Omega/\text{K}$  vrlo mala, u obzir se mora uzeti i otpor provodnika kojim je

---

senzor povezan sa ostalim dijelom mjernog kruga.

#### **4.4. Ultrazvučni mjerač udaljenosti**

U okviru laboratorijskog modela DD 3100 se za mjerenje nivoa vode u rezervoaru koristi ultrazvučni mjerač udaljenosti. Princip rada ovog mjerača se zasniva na mjerenju vremena potrebnog da u sredini ispunjenoj vazduhom na atmosferskom pritisku ultrazvučni impuls emitovan od strane ultrazvučnog predajnika stigne do prepreke, odbije se i dođe nazad do ultrazvučnog prijemnika koji je montiran u isto kućište sa ultrazvučnim predajnikom (slika 4.4.).



Slika 4.4. Ultrazvučni mjerač nivoa

U okviru mjerača se nalazi mjerni pretvarač koji vrijeme putovanja ultrazvuka od predajnika do prijemnika pretvara u standardni strujni signal 4-20 mA.

---

## 5. Uputstva za laboratorijsku opremu

### 5.1. Uvod

U okviru ovog poglavlja će biti data kratka uputstva za korištenje i osnovne informacije za laboratorijsku opremu koja je na raspolaganju za izvođenje praktične nastave u okviru predmeta. Uputstva su namijenjena samo kao polazna informacija, a detaljnija informacija se može naći u zvaničnim korisničkim uputstvima opreme.

### 5.2. Bürkert 1110 Digital Industrial Controller

Bürkert 1110 Digital Industrial Controller predstavlja digitalni industrijski regulator je namijenjen za rješavanje zadataka iz oblasti procesnog upravljanja. Omogućava realizaciju PID zakona upravljanja, u okviru mikroprocesorskog uređaja nove generacije. Ovaj regulator kao ulazne i izlazne signale može koristiti bilo standardne strujne i naponske signale, ili frekventne signale, što omogućava direktno povezivanje davača i izvršnih organa. Posjeduje takođe i relejne izlaze, te dodatne izlaze za indikaciju greške, kao i binarne ulaze i izlaze, što omogućava realiziranje dodatnih funkcija. Za povezivanje regulatora sa drugim uređajima u sistemu upravljanja se može koristiti ugrađeni RS-232 ili RS-485/PROFIBUS interfejs.

Uređaj je jednostavan za korištenje i posjeduje LCD displej na kome je moguće jednostavno očitati bitne informacije.

Korištenjem sistema menija, operator može izvršiti sljedeće operacije:

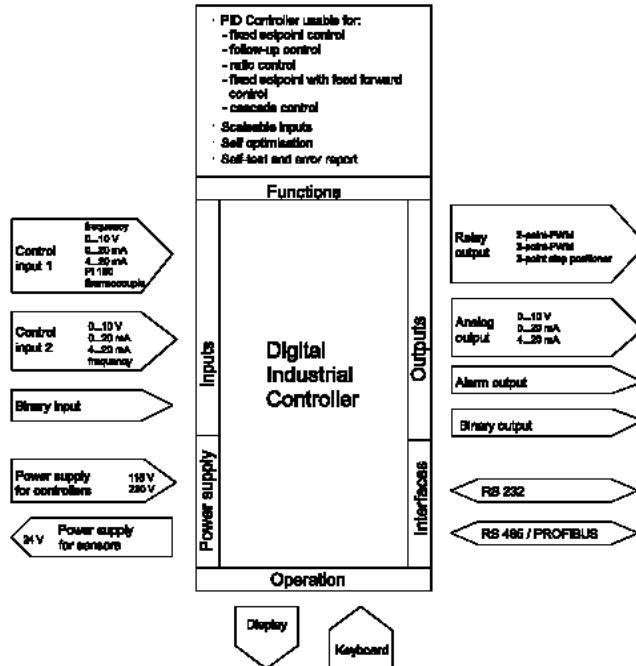
- Konfigurisanje strukture regulatora,
- Definiranje vrijednosti parametara regulatora,
- Ručne intervencije na procesu (ručno upravljanje).

Svi podaci se pohranjuju u EEPROM i raspoloživi su i nakon prekida napajanja uređaja.

Uređaj omogućava postavljanje različitih nivoa pristupa funkcijama regulatora (korištenjem korisničkih kodova), radi sprječavanja neovlaštenog pristupa. Nepromjenljivi master kod omogućava pristup svim funkcijama uređaja.

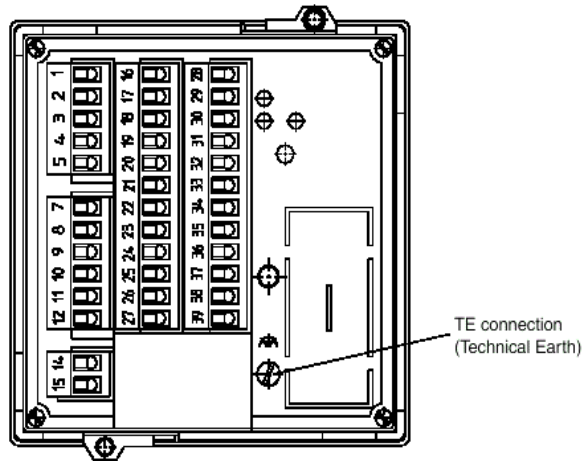
U okviru regulatora su implementirani algoritmi optimizacije, koji omogućavaju samopodešavanje i adaptaciju regulatora, tako da se vrijednosti regulatora mogu automatski prilagoditi procesu.

Struktura regulatora je predstavljena na slici 5.1. Na slici 5.2 je predstavljena zadnja strana uređaja sa kontaktima za povezivanje, a na slici 5.3. je prikaz rasporeda i funkcija ovih kontakata.

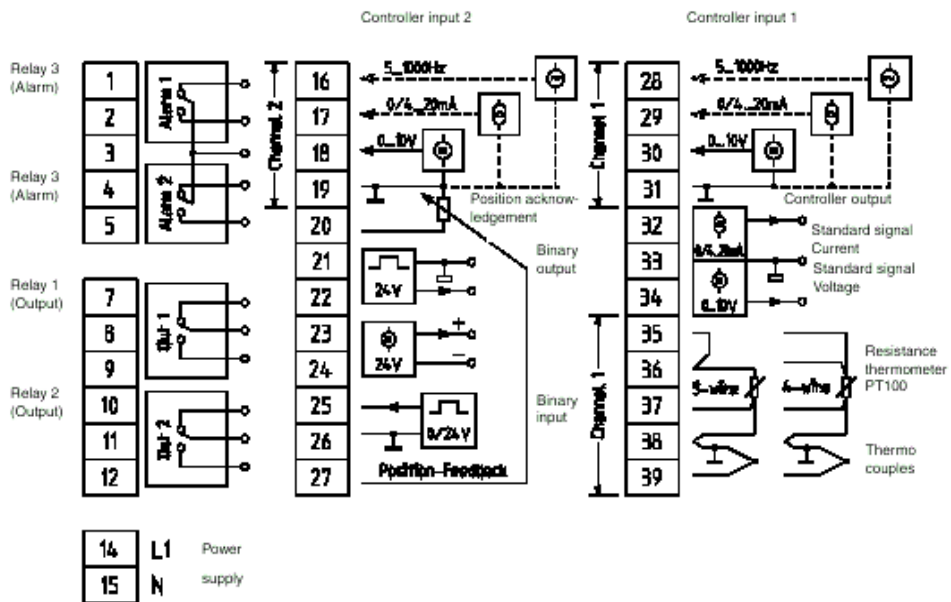


Slika 5.1. Struktura regulatora Bürkert 1110





Slika 5.2. Zadnja strana uređaja i kontakti za povezivanje



Slika 5.3. Raspored i funkcija kontakata za povezivanje

Bürkert 1110 Digital Industrial Controller se može koristiti za realizaciju:

- strukture sa jednom povratnom spregom i konstantnom zadanom vrijednošću,
- strukture sa bez povratne sprege (*feed forward*) i konstantnom zadanom

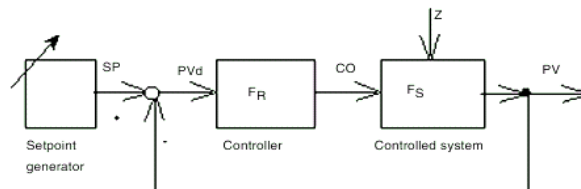
---

vrijednošću,

- slijedne regulacije (sa promjenljivom zadanom vrijednošću),
- regulacije odnosa veličina,
- kaskadne regulacije.

### 5.2.1. Struktura sa negativnom povratnom spregom i konstantnom zadanom vrijednošću

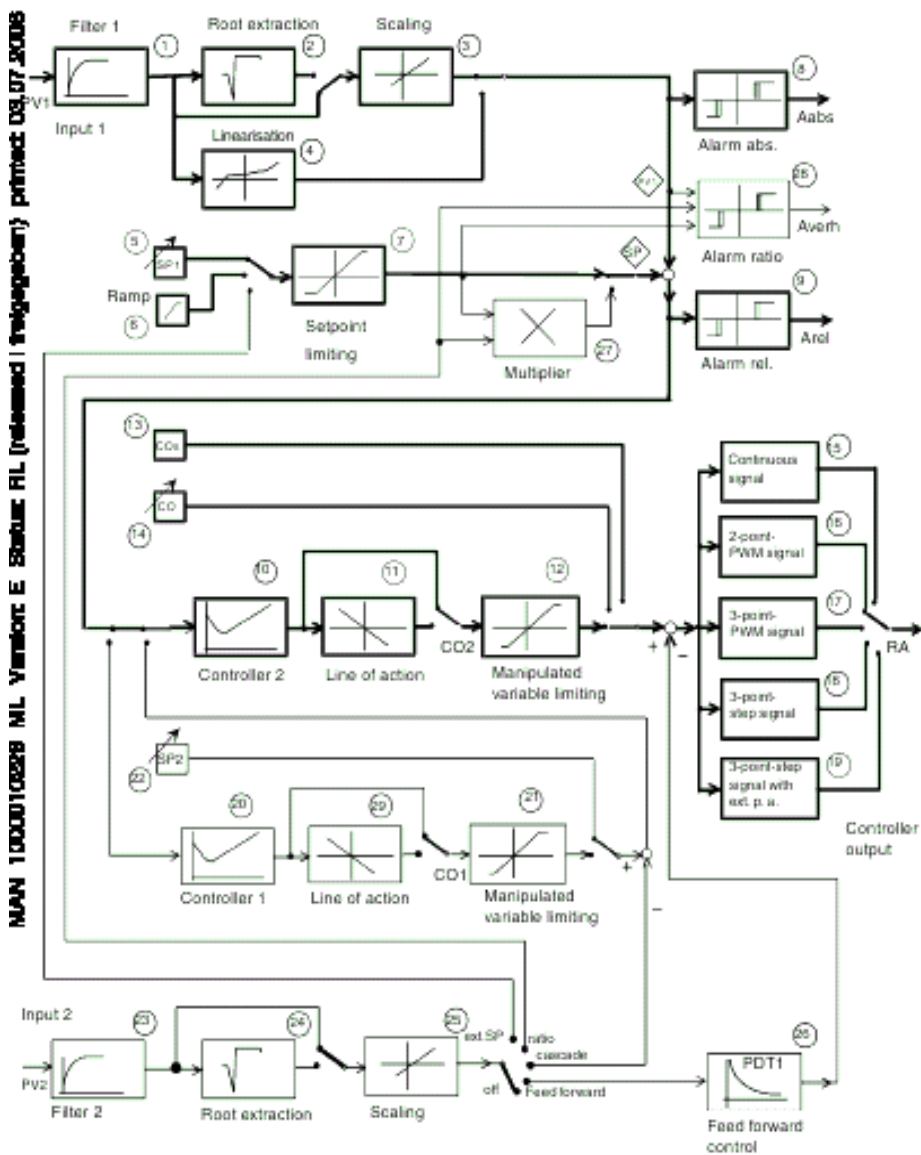
Struktura sa negativnom povratnom spregom i konstantnom zadanom vrijednošću se koristi kada je potrebno održati reguliranu veličinu (npr. temperaturu) na konstantnoj vrijednosti. Regulirana veličina (PV - *Process Variable*) se mjeri i njena vrijednost se upoređuje sa zadanom vrijednošću (SP – *Set Point*). Ova struktura je principijelno predstavljena na slici 5.4., a njeno podešenje na detaljnoj strukturi regulatora je predstavljeno na slici 5.5.



Slika 5.4. Struktura sa negativnom povratnom spregom

Ukoliko pod utjecajem smetnje  $Z$  dođe do odstupanja regulirane veličine PV od zadane vrijednosti SP, regulator  $F_R$  će na osnovu regulacionog odstupanja (regulacione greške)  $PVd = SP - PV$  generisati vrijednost manipulativne varijable CO takvu da djeluje u smislu otklanjanja odstupanja.

Kao primjer ovakvog problema upravljanja može poslužiti problem regulacije temperature u prostoriji. Cilj regulacije je da se kompenziraju poremećaji koji uzrokuju odstupanje sobne temperature od željene vrijednosti, kao što su gubici toplote kroz zidove, te strujanje zraka zbog otvaranja prozora i vrata. Regulator kontroliše dotok toplote, tako da se sobna temperatura održi na zadanoj vrijednosti.



Slika 5.5. Detaljna struktura regulatora sa negativnom povratnom spregom i konstantnom zadanom vrijednošću

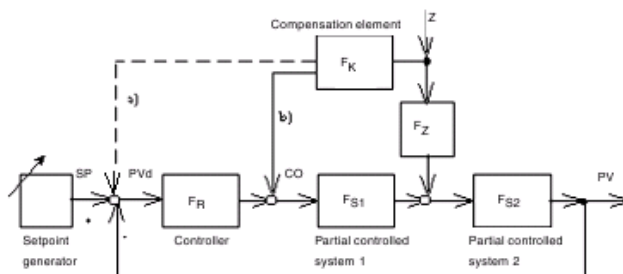
Zakon na osnovu kojega regulator određuje vrijednost signala CO je najčešće PID, čiji parametri su određeni tako da regulator proizvede željeno dejstvo. U ovoj strukturi se od dva raspoloživa PID regulatora u okviru uređaja (*Controller 1* i

---

Controller 2) koristi PID 2.

### 5.2.2. Struktura sa dodatnom regulacijom u otvorenom

U slučajevima kada je poremećaj mjerljiv, kvalitet odziva strukture sa negativnom povratnom spregom se može znatno poboljšati uvođenjem dodatne regulacije u otvorenom (*feedforward*) (slika 5.6.). Signal poremećaja se vodi preko kompenzatora  $F_K$ , koji ima PD djelovanje, uz inerciju. P komponenta djeluje proporcionalno vrijednosti poremećaja, a D komponenta djeluje kao realni diferencijator, na osnovu brzine promjene vrijednosti poremećaja.



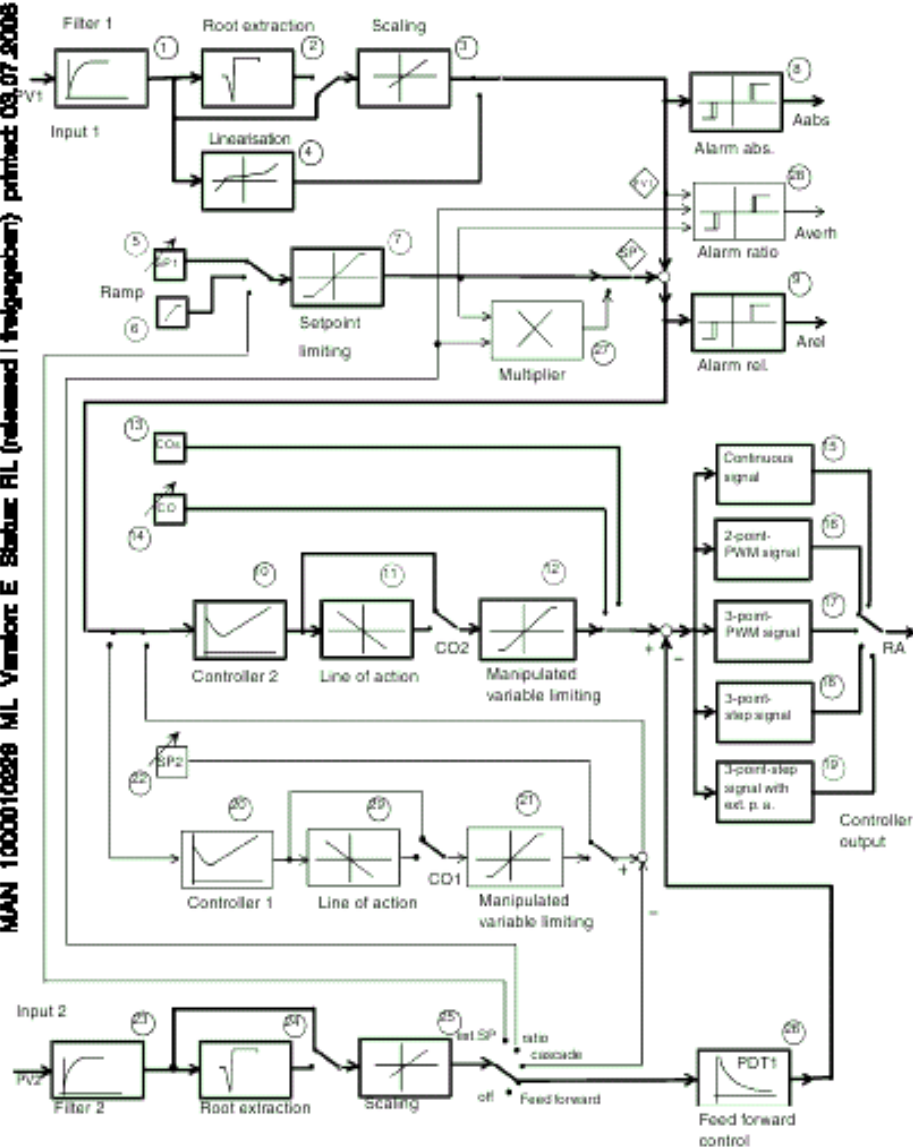
Slika 5.6. Struktura sa dodatnom regulacijom u otvorenom po poremećaju

Detaljna struktura je predstavljena na slici 5.7.

Primjer ovakvog problema regulacije je regulacija nivoa vode u parnom kotlu. Nivo se mjeri i na osnovu odstupanja od željene vrijednosti nivoa se određuje protok u dovodnom cjevovodu. Protok pare u izlaznom cjevovodu predstavlja poremećaj koji je mjerljiv i može se koristiti za kompenzaciju u otvorenom.

### 5.2.3. Slijedna regulacija

Kod slijedne regulacije je potrebno da regulirana veličina PV1 što je moguće bolje prati promjene zadane vrijednosti PV1, koja se sada dovodi u regulator izvana (slika 5.8. Podešenje regulatora u ovakvoj strukturi mora biti takvo da se postigne zadovoljavajući odziv na sve promjene zadane veličine, pri čemu je vrijeme

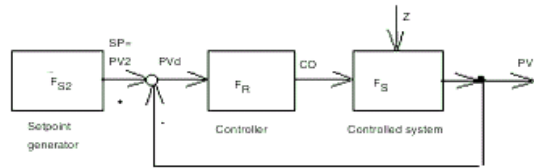


Slika 5.7. Detaljna struktura regulatora sa dodatnom regulacijom u otvorenom po poremećaju

smirivanja prelaznog procesa unutar prihvatljivih granica.

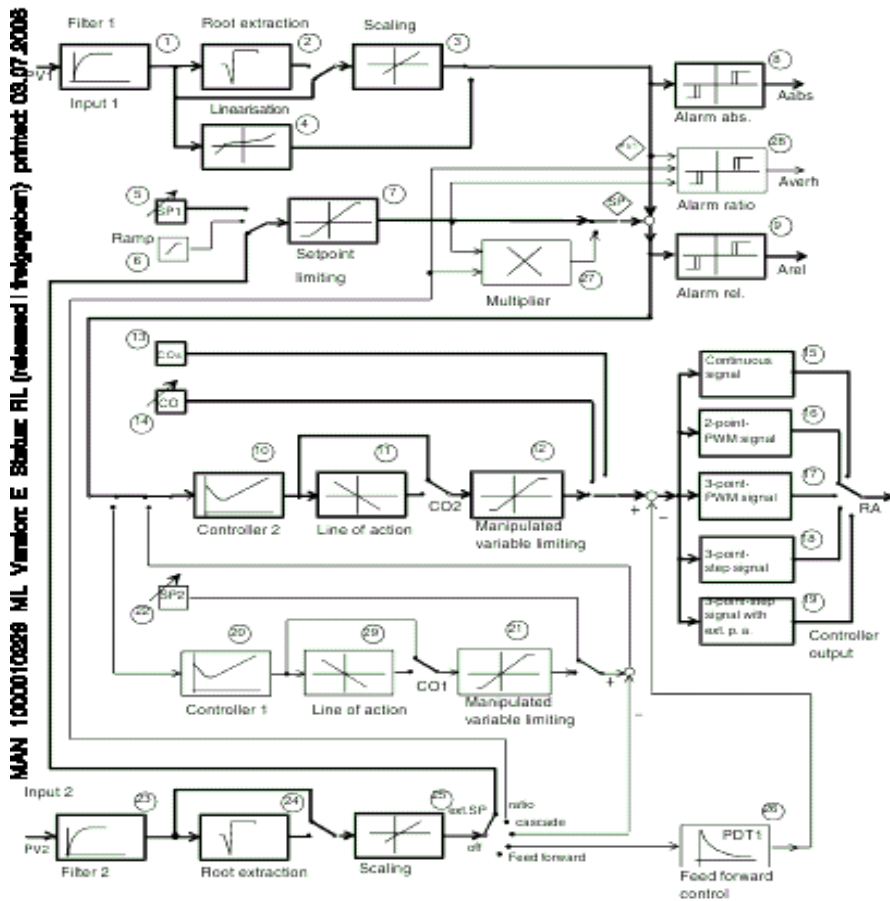
Primjer ovakvog problema regulacije može biti upravljanje pozicijom električnog

motora.



Slika 5.8. Slijedna regulacija

Detaljna struktura regulatora za slijednu regulaciju je predstavljena na slici 5.9.



Slika 5.9. Detaljna struktura regulatora za slijednu regulaciju

#### 5.2.4. Regulacija odnosa

Regulacija odnosa predstavlja specijalan slučaj slijedne regulacije, gdje je zadatak da regulirana veličina (PV1) prati promjene druge procesne veličine (PV2) sa zadanim odnosom:

$$SP_r = \frac{PV1}{PV2}$$

gdje je:

SP<sub>r</sub>: koeficijent odnosa veličina

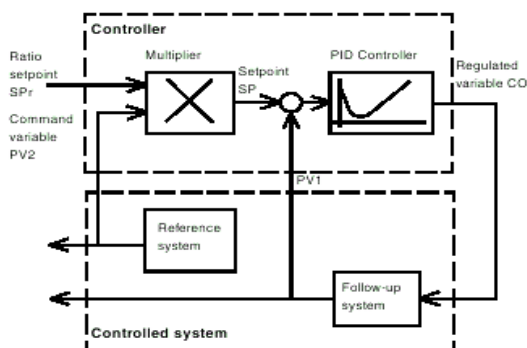
PV1: zavisna (regulirana) veličina

PV2: nezavisna veličina

Ovaj izraz omogućava da se interno odredi zadana vrijednost:

$$PV1_{set} = PV2 * SP_r$$

$$SP = X2 * SP_r$$



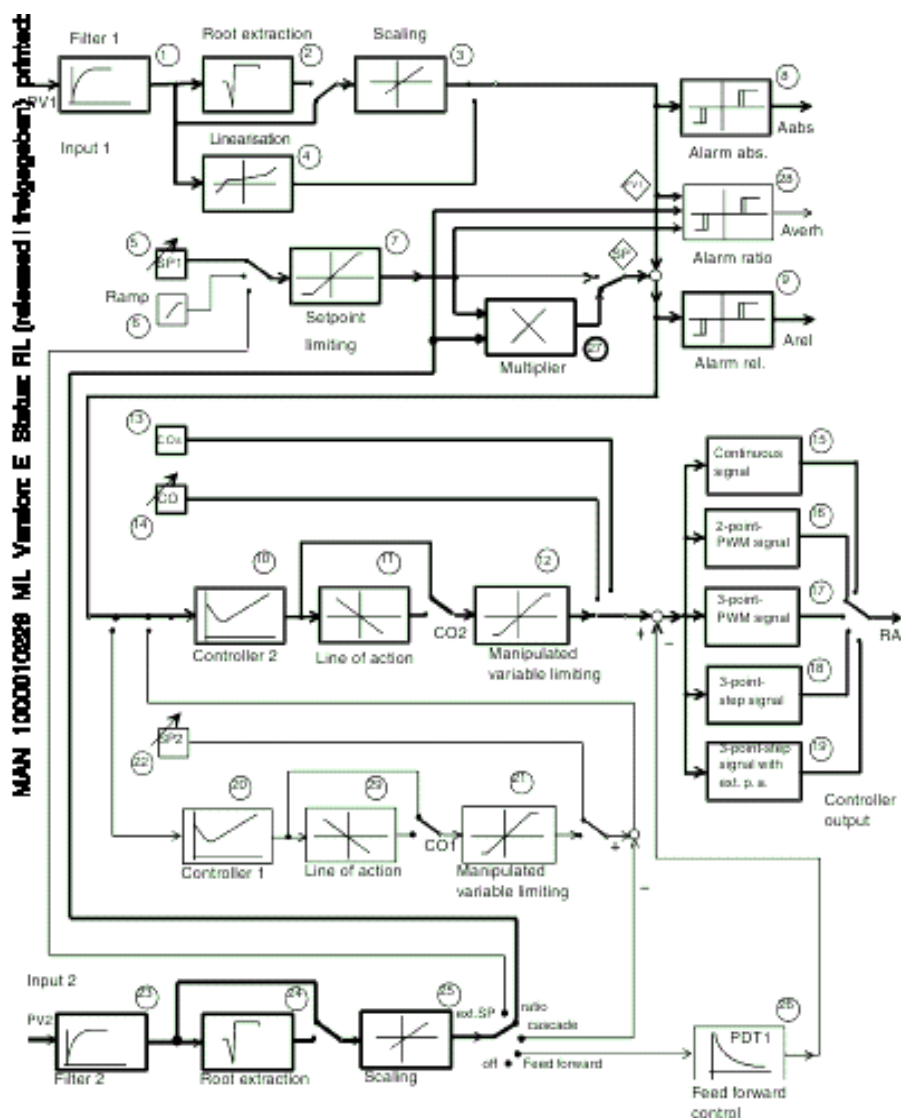
Slika 5.10. Struktura za regulaciju odnosa veličina

Detaljna struktura regulatora za regulaciju odnosa je predstavljena na slici 5.11.

#### 5.2.5. Kaskadna regulacija

Kod kaskadne regulacije dvije upravljačke konture su povezane tako da jedna upravljačka kontura (glavna kontura) sadrži drugu (pomoćnu) upravljačku konturu

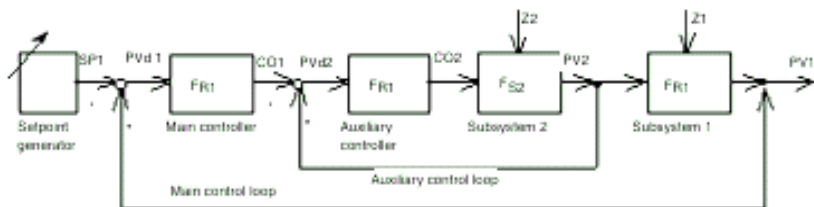
(slika 5.12).



Slika 5.11. Detaljna struktura regulatora za regulaciju odnosa veličina

Sistem upravljanja se sastoji od dva podsistema, FS1 i FS2. Regulirana veličina PV1 se mjeri na izlazu glavne konture FS1, a pomoćna regulirana veličina se mjeri na izlazu sistema FS2. Pomoćna regulaciona kontura se sastoji od regulatora FR2 i podsistema FS2. Izlaz regulatora glavne konture FR1 (CO1) predstavlja zadanu



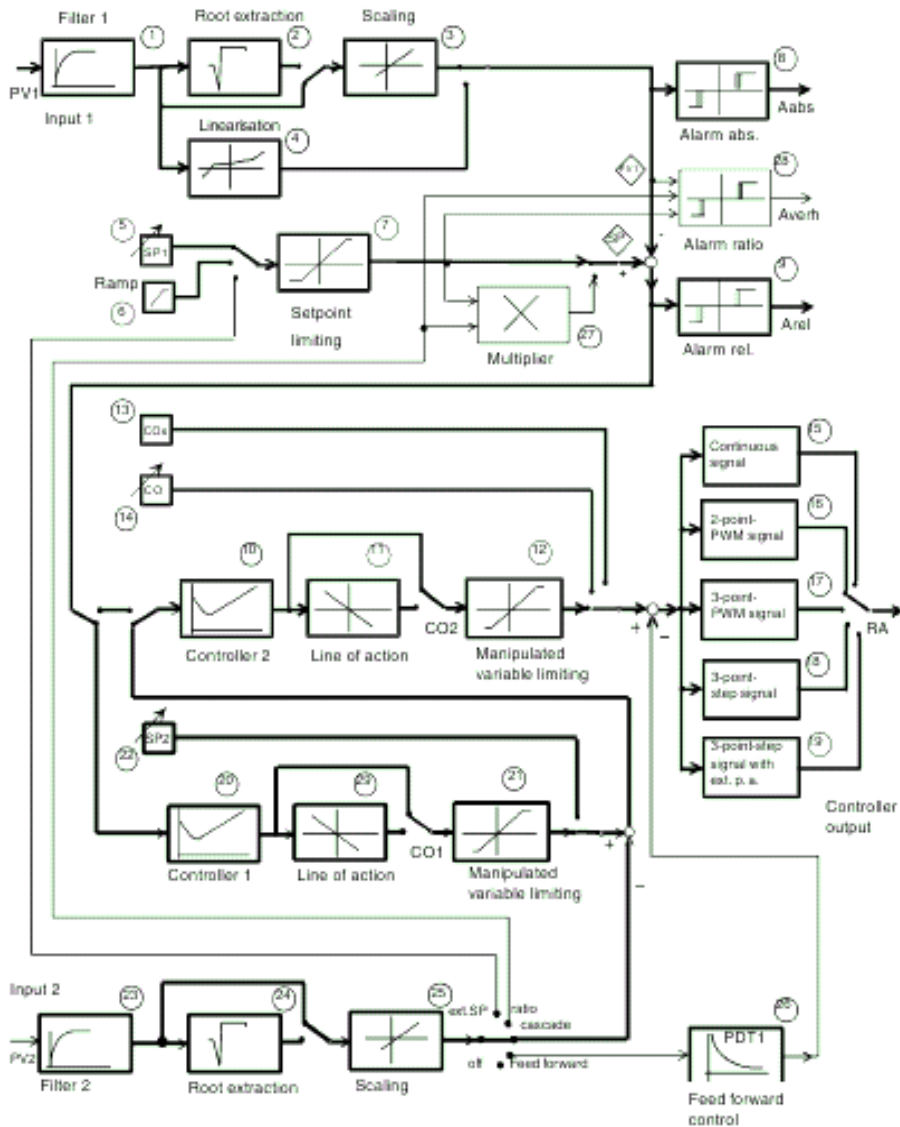


Slika 5.12. Kaskadna regulacija

vrijednost za pomoćnu regulacionu konturu. Zadana vrijednost za glavnu regulacionu konturu se može dovoditi izvana, ili može biti interno generirana kao konstantna zadana vrijednost. Pomoćna regulaciona kontura mora imati brži odziv od glavne regulacione konture, a njen osnovni zadatak je da otklanja poremećaje  $Z_2$  koji djeluju na podsistem  $FS_2$ . Poremećaji koji djeluju na podsistem  $FS_1$  se otklanjaju od strane glavne regulacione konture.

Kao primjer ovakve regulacije se može navesti regulacija temperature u rezervoaru grijanom parom. Brza pomoćna regulaciona kontura je kontura za regulaciju protoka pare, a glavna regulaciona kontura je kontura za regulaciju temperature.

Detaljna struktura regulatora za kaskadnu regulaciju je predstavljena na slici 5.13.

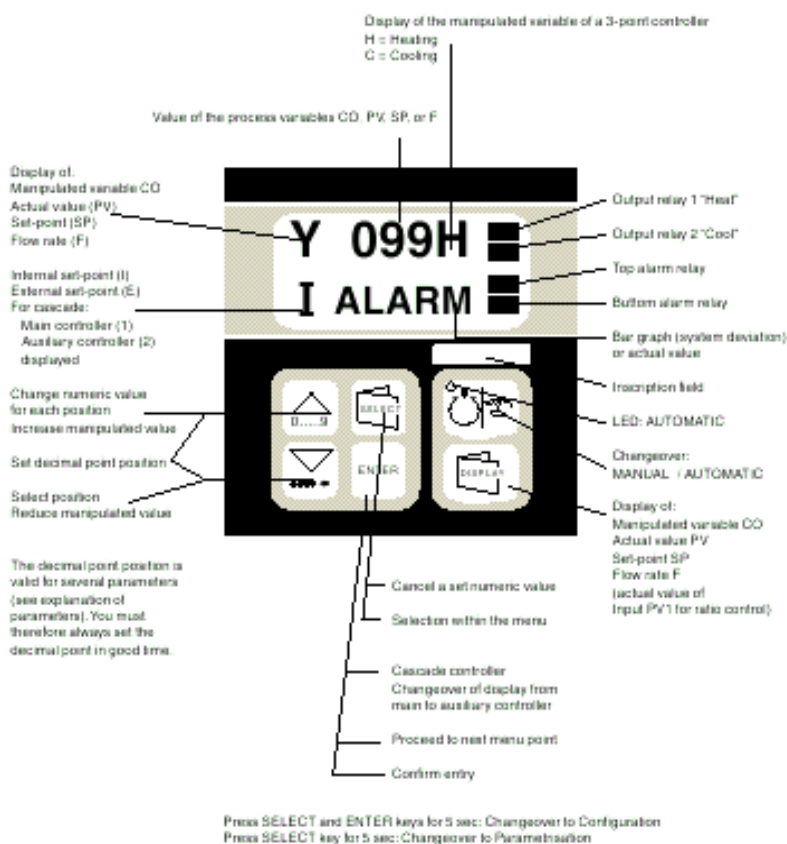


Slika 5.13. Detaljna struktura regulatora za kaskadnu regulaciju

## 5.2.6. Rad sa uređajem

Regulator posjeduje dva osnovna načina rada: ručni (MANUAL) i automatski (AUTOMATIC). Nakon uključjenja napajanja, regulator će se naći u načinu rada u kojem je bio prilikom isključenja napajanja.

Izgled prednje ploče uređaja je predstavljen na slici 5.14.



Slika 5.14. Izgled prednje ploče uređaja

U donjem dijelu se nalazi 6 tastera, čija funkcija ovisi o trenutnom nivou. U gornjem dijelu se nalazi LCD sa dvije linije od po 8 karaktera. Izgled displeja takođe ovisi o nivou. U tabeli 5.1. je pojašnjena funkcija svakog od ovih tastera.







| <b>Red. br.</b> | <b>Taster</b>                                                                                           | <b>Funkcija tastera</b>                                                                                                                                                                                                                                                                                                                                      |
|-----------------|---------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| 1.              | <br>MANUAL/AUTO key    | Promjena načina rada između MANUAL i AUTOMATIC.<br>AUTOMATIC način rada indicira LED na tasteru.                                                                                                                                                                                                                                                             |
| 2.              | <br>DISPLAY key        | Promjena procesne varijable koja se prikazuje u gornjem dijelu displeja:<br>SP: zadana vrijednost<br>PV: mjerena vrijednost regulirane veličine PV1 (ili vrijednost odnosa)<br>CO: upravljačka (manipulativna) varijabla (Ch i Cc za 3-značni PWM signal)<br>F: vrijednost protoka (mjerena vrijednost ulazne varijable PV1 za upravljanje odnosom veličina) |
| 3.              | <br>SELECT key         | Aktiviranje nivoa podešavanja parametara pritiskom u trajanju od 5 sekundi                                                                                                                                                                                                                                                                                   |
| 4.              | <br>ENTER key         | <ul style="list-style-type: none"> <li>• Aktiviranje nivoa konfiguracije strukture regulatora istovremenim pritiskom ovog tastera i SELECT tastera u trajanju od 5 sekundi</li> <li>• Potvrda vrijednosti</li> </ul>                                                                                                                                         |
| 5.              | <br>„Down arrow“ key | <ul style="list-style-type: none"> <li>• Izbor cifre</li> <li>• Smanjenje vrijednosti upravljačke varijable (u MANUAL načinu rada), tj. smanjenje napona ili struje (kada se koriste standardni signali) odnosno širine impulsa PWM signala</li> </ul>                                                                                                       |
| 6.              | <br>„Up arrow“ key   | <ul style="list-style-type: none"> <li>• Modificiranje numeričke vrijednosti</li> <li>• Povećavanje vrijednosti upravljačke varijable (u MANUAL načinu rada), tj. povećanje napona ili struje (kada se koriste standardni signali) odnosno širine impulsa PWM signala</li> </ul>                                                                             |

Tabela 5.1. Funkcije tastera uređaja

---

Postoje tri osnovna nivoa rada:

- **Konfiguracija strukture regulatora**

U okviru ovog nivoa se vrši izbor odgovarajuće strukture regulatora i prilagođavanje ulaza i izlaza davačima (senzorima) i izvršnim organima (aktuatorima). Za vrijeme konfigurisanja regulator mora biti u MANUAL modu. Nakon završetka konfigurisanja, regulator se vraća u način rada u kome je bio prije ulaska u nivo konfiguracije.

Nivo konfiguracije se aktivira istovremenim pritiskom na tastere 3. i 4. u trajanju od 5 sekundi.

- **Podešavanje parametara regulatora**

U okviru ovog nivoa se unose parametri za izabranu strukturu regulatora. Dozvoljene su samo izmjene parametara, a ne i strukture ili prilagođenja davačima i izvršnim organima. Za vrijeme podešavanja parametara regulator ostaje u načinu rada u kome je bio. Ukoliko u okviru od 30 sekundi ne bude pritisnut ni jedan taster, regulator napušta ovaj nivo, pri čemu se sve izmjene parametara snimaju.

Nivo podešavanja parametara regulatora se aktivira pritiskom na taster 3. u trajanju od 5 sekundi.

- **Izvršni nivo**

Ovaj nivo predstavlja osnovni nivo rada regulatora, kada on izvršava upravljački algoritam. Promjena između MANUAL i AUTOMATIC načina rada se vrši pritiskom na taster 1., pri čemu se trenutni način rada indicira LED, koja svijetli kada se regulator nalazi u AUTOMATIC načinu rada. Vrijednost zadane vrijednosti i regulirane veličine, te upravljačke veličine se mogu prikazati na LCD. Zadana

---

vrijednost se može postavljati i u MANUAL i u AUTOMATIC načinu rada. Ukoliko se izabere AUTO funkcija, postavljanje zadane vrijednosti aktivira proces samooptimizacije. Vrijednost upravljačke varijable se može zadavati samo u MANUAL načinu rada.

Izbor veličine koja se prikazuje se vrši pritiskom na taster 2. Prikaz veličina ovisi o izabranoj strukturi, što je detaljno opisano u korisničkom uputstvu uređaja. Kod strukture sa negativnom povratnom vezom i konstantnom zadanom vrijednošću je u gornjem dijelu displeja moguće prikazati zadanu vrijednost (SP), trenutnu vrijednost regulirane veličine (PV), vrijednost upravljačke (manipulativne) varijable (Co). Kada je na displeju prikazana upravljačka varijabla, pomoću tastera 5. i tastera 6. je moguće mijenjati njenu vrijednost. Tasterom 5. se bira pozicija vrijednosti koja se želi mijenjati, a tasterom 6. se mijenja vrijednost cifre. Nakon podešavanja vrijednosti je potrebno potvrditi unos pritiskom na taster 4.

### ***5.3. Meilhaus Electronic 1208-FS modul za akviziciju podataka***

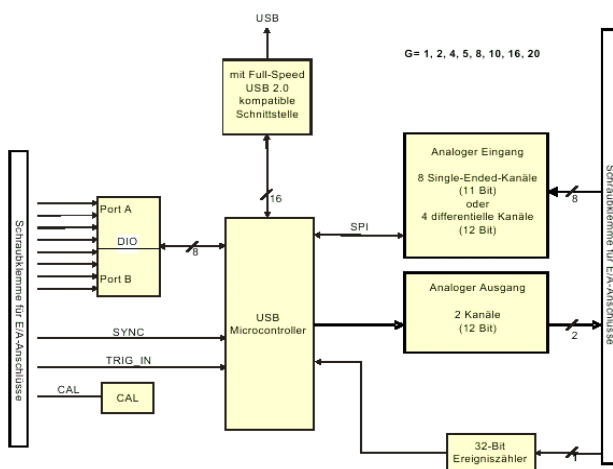
Meilhaus Electronic ME-RedLab 1208-FS (slika 5.15.) je modul za akviziciju podataka, koji se sa PC računarem povezuje preko USB 2.0 interfejsa. Modul posjeduje 8 analognih ulaza, dva analogna izlaza rezolucije 12 bita, 16 digitalnih ulazno/izlaznih linija i jedan eksterni 32-bitni brojački ulaz. Uređaj se napaja preko USB porta računara, tako da mu nije neophodan vanjski izvor napajanja. Modul predstavlja kopiju modula MCC 1208-FS (Measurement Computing Corporation), koji je u potpunosti podržan u Data Acquisition Toolbox-u Matlaba. Takođe, uz modul dolaze i drajveri za LabVIEW.

Analogni ulazi modula imaju rezoluciju od 11 bita kada su povezani na masu, odnosno 12 bita kada se koriste kao diferencijalni ulazi. 16 digitalnih linija se mogu konfigurisati kao dva 8-bitna ulazna ili izlazna porta. 32-bitni brojački ulaz omogućava brojanje signala TTL naponskog nivoa. Modul posjeduje linije za sinhronizaciju.



Slika 5.15. Izgled modula ME-RedLab 1208-FS

Blokovska struktura modula ME-Redlab 1208-FS je prikazana na slici 5.16.



Slika 5.16. Blokovska struktura modula ME-RedLab 1208-FS

### 5.3.1. Načini prikupljanja vrijednosti analognih signala

ME-RedLab 1208-FS može prikupljati analogne signale na dva načina: softverski taktovano i sa kontinuiranim očitavanjem.

Kod softverski taktovanog očitavanja je A/D konvertor pod kontrolom softvera. Softver je zadužen za pokretanje konverzije, nakon čijeg završetka se vrijednost prenosi na računar. Maksimalna brzina očitavanja vrijednosti na analognom ulazu

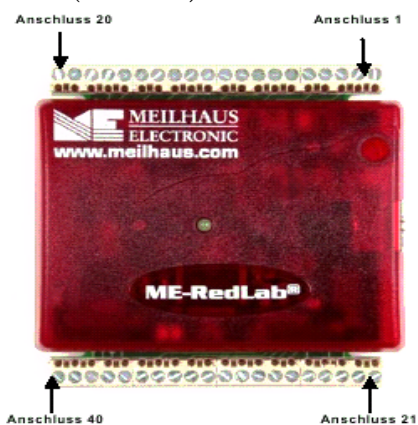
---

ovisi o sistemu.

Kod kontinuiranog očitavanja vrijednosti na analognom ulazu A/D konvertor neprekidno očitava vrijednosti ulaznih signala, koje se nakon završene konverzije pohranjuju u FIFO (First-In-First-Out) bafer. Na ovaj način se mogu očitavati vrijednosti sa do osam analognih ulaza. Maksimalna brzina očitavanja je 50000 uzoraka u sekundi, gledano za sve kanale. Ovo znači da će, u slučaju da se očitava vrijednost samo sa jednog kanala, brzina uzorkovanja ovog ulaza biti 50000 uzoraka u sekundi. U slučaju da se očitavaju dva kanala, brzina očitavanja svakog od njih će biti 25000 uzoraka u sekundi, za slučaj očitavanja četiri kanala brzina očitavanja svakog pojedinačno će biti 12500 uzoraka u sekundi itd. Ovaj način rada se može aktivirati bilo softverskom komandom, ili preko vanjskog hardverskog trigeru.

### 5.3.2. Povezivanje modula

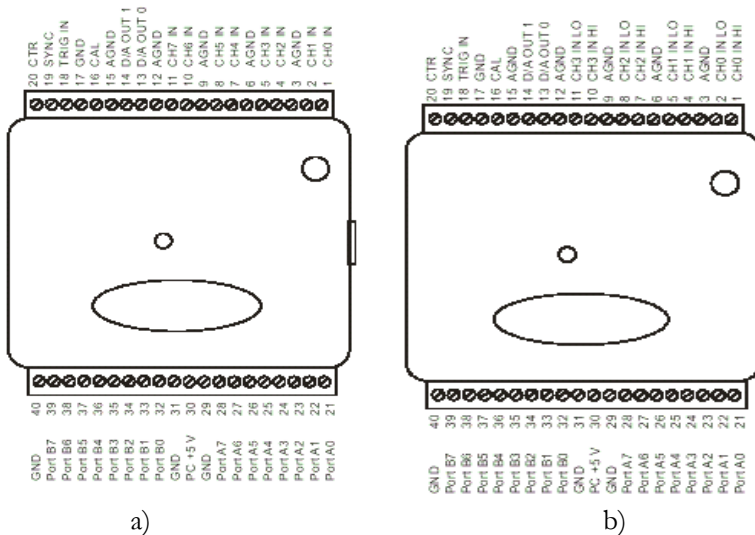
Modul ME-RedLab 1208-FS posjeduje LED koja indicira status uređaja. Ukoliko LED svijetli neprekidno, modul je priključen na napajanje i USB sabirnicu PC računara. Ako LED treperi, u toku je prenos podataka između PC računara i modula. Modul se sa vanjskim svijetom povezuje preko dvije klemne, koje se nalaze na njegovim bočnim stranama (slika 5.17.).



Slika 5.17. Klemne za povezivanje modula ME-RedLab 1208-FS



Slika 5.18. prikazuje oznake pojedinačnih priključaka za slučaj uzemljenih (5.18.a) i diferencijalnih (5.18.b) ulaza.



Slika 5.17. Raspored priključaka kod:

a) uzemljenog i b) diferencijalnog priključivanja analognih ulaza

Oznake priključaka su pojašnjene u tabeli 5.2.

| <b>Oznake</b>                                  | <b>Značenje</b>                                         |
|------------------------------------------------|---------------------------------------------------------|
| CH0 IN – CH7 IN                                | Analogni ulazi 0-7, uzemljeni način priključivanja      |
| CH0 IN HI – CH3 IN HI<br>CH0 IN LO – CH3 IN LO | Analogni ulazi 0-3, diferencijalni način priključivanja |
| D/A OUT 0, D/A OUT 1                           | Analogni izlazi 0 i 1                                   |
| TRIG_IN                                        | Ulaz vanjskog okidačkog signala                         |
| SYNC                                           | Priključak za sinhroniziranje više modula               |
| CAL                                            | Priključak za kalibriranje                              |
| AGND                                           | Priključci mase analognog signala                       |
| GND                                            | Masa modula                                             |
| CTR                                            | Brojački ulaz                                           |
| Port A0-A7, Port B0-B7                         | Digitalni ulazi/izlazi                                  |
| PC+5V                                          | 5V, Napon napajanja sa USB porta                        |

Tabela 5.2. Oznake priključaka na modulu

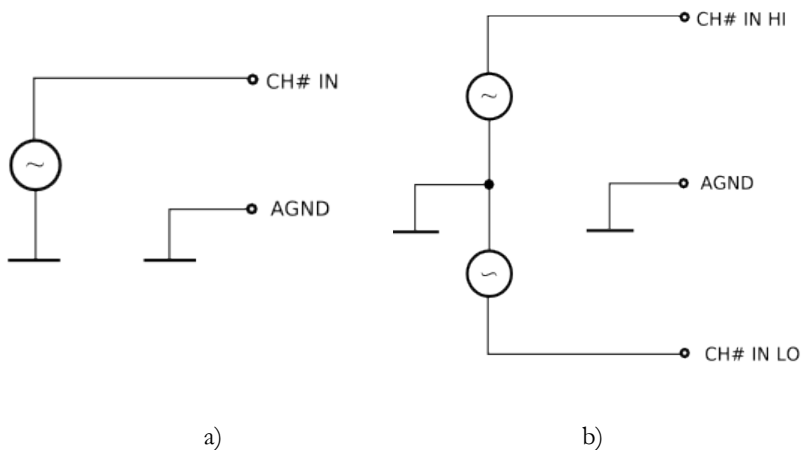
---

- **Analogni ulazi**

Kako je već napomenuto, analogni ulazi se mogu priključiti ili uzemljeno (jednostrano) ili diferencijalno.

Kod uzemljenog (jednostranog) načina priključivanja, na raspolaganju je 8 kanala. U ovom slučaju se mjereni signal dovodi na ulaz CH# IN, a drugi kraj izvora mjenog signala se povezuje na zajedničku masu analognog signala (AGND) (slika 5.18.a) . U ovom slučaju je opseg vrijednosti ulaznog signala  $\pm 10$  V, dok ostali opsezi nisu podržani.

U slučaju diferencijalnog načina priključivanja analognog signala, na raspolaganju su četiri analogna kanala. Vrijednost ulaznog signala se mjeri u odnosu na CH# IN LO. Na priključak CH# IN HI se dovodi mjereni signal, na CH# IN LO referentni signal, a AGND se povezuje sa masom (slika 5.18.b). Opsezi vrijednosti ulaznog analognog signala su:  $\pm 20$  V,  $\pm 10$  V,  $\pm 5$  V,  $\pm 4$  V,  $\pm 2,5$  V,  $\pm 2,0$  V, 1,25 V i  $\pm 1,0$  V. Osim toga, diferencijalni ulazi posjeduju ulazno pojačalo niskog nivoa šuma sa pojačanjem do 20, što je moguće programirati. Uslov da bi ovo pojačalo, odnosno diferencijalni kanali ostali u linearnom režimu rada je da se svi analogni signali nalaze u opsegu -10V do +20V u odnosu na masu.



Slika 5.18. a) Uzemljeno i b) diferencijalno priključivanje analognih signala

---

- **Analogni izlazi**

Modul ME-RedLab 1208-FS posjeduje dva analogna izlaza, koji mogu dati do 10000 uzoraka u sekundi ukupno, odnosno do 5000 uzoraka po kanalu. Izlazni napon se kreće u opsegu od 0 do 4,096 V, uz rezoluciju od 1 mV po bitu.

- **Digitalni ulazi/izlazi**

Na modulu se nalaze priključci za 16 digitalnih signala, koji se mogu softverski konfigurirati kao ulazi ili izlazi TTL nivoa.

- **Brojački ulaz**

CTR priključak predstavlja brojački ulaz, preko koga se može dovesti signal na interni 32-bitni brojač modula. Ovaj signal treba biti TTL nivoa, a stanje brojača se mijenja na promjenu sa 0 na 5 V. Brojač je u stanju brojati signale frekvencije do 1 MHz.

Pojašnjenje ostalih priključaka (SYNC, CAL) se može naći u dokumentaciji koja dolazi uz modul.

Tabele 5.3. i 5.4. daju pregled svih priključaka modula u dva načina priključivanja analognih ulaza.

#### ***5.4. Moeller Easy 512-DC-RC Programmable Logic Controller***

Easy 512-DC-RC je elektronički upravljački uređaj koji omogućava realizaciju logičkih funkcija, uz dodatak tajmera i brojača. Koristi se za realizaciju logičkog upravljanja u različitim aplikacijama, od upravljanja u zgradama i fabrikama, do upravljanja mašinama. Za implementaciju algoritma se koristi opis ljestvičastim dijagramom (*ladder diagram*), pri čemu se svaki element dijagrama može unijeti putem displeja i tastera na uređaju. Osim toga, dijagrami se mogu kreirati i na PC računaru, te onda prebaciti na PLC. Pored programiranja korištenjem ljestvičastog

dijagrama, PLC je moguće programirati i koristeći EASY-SOFT-BASIC razvojni alat.

Neke od mogućih realizacija dijagrama su:

- Serijska i paralelna veza kontakata i njihovih negiranih vrijednosti,
- Veza izlaznih kontakata i markera,
- Korištenje izlaza kao releja, impulsnih releja ili latch-eva,
- Korištenje višefunkcijskih vremenskih releja,
- Korištenje brojača na više i niže,
- Brojanje impulsa visoke frekvencije,
- Measure frequencies
- Procesiranje analognih ulaza,
- Prikaz proizvoljnog teksta sa varijablama, unošenje zadanih vrijednosti,
- Korištenje sata,
- Brojanje radnih sati,
- Praćenje izvršavanja dijagrama,
- Snimanje, učitavanje i zaštita dijagrama lozinkom.

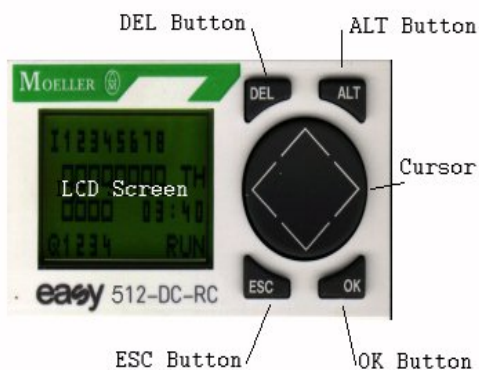
| Klemme | Signalname | Klemme | Signalname |
|--------|------------|--------|------------|
| 1      | CH0 IN     | 21     | Port A0    |
| 2      | CH1 IN     | 22     | Port A1    |
| 3      | AGND       | 23     | Port A2    |
| 4      | CH2 IN     | 24     | Port A3    |
| 5      | CH3 IN     | 25     | Port A4    |
| 6      | AGND       | 26     | Port A5    |
| 7      | CH4 IN     | 27     | Port A6    |
| 8      | CH5 IN     | 28     | Port A7    |
| 9      | AGND       | 29     | GND        |
| 10     | CH6 IN     | 30     | PC+5V      |
| 11     | CH7 IN     | 31     | GND        |
| 12     | AGND       | 32     | Port B0    |
| 13     | D/A OUT 0  | 33     | Port B1    |
| 14     | D/A OUT 1  | 34     | Port B2    |
| 15     | AGND       | 35     | Port B3    |
| 16     | CAL        | 36     | Port B4    |
| 17     | GND        | 37     | Port B5    |
| 18     | TRIG IN    | 38     | Port B6    |
| 19     | SYNC       | 39     | Port B7    |
| 20     | CTR        | 40     | GND        |

Tabela 5.3. Oznake svih priključaka na modulu kod uzemljenog priključivanja analognog signala

| Klemme | Signalname | Klemme | Signalname |
|--------|------------|--------|------------|
| 1      | CH0 IN HI  | 21     | Port A0    |
| 2      | CH0 IN LO  | 22     | Port A1    |
| 3      | AGND       | 23     | Port A2    |
| 4      | CH1 IN HI  | 24     | Port A3    |
| 5      | CH1 IN LO  | 25     | Port A4    |
| 6      | AGND       | 26     | Port A5    |
| 7      | CH2 IN HI  | 27     | Port A6    |
| 8      | CH2 IN LO  | 28     | Port A7    |
| 9      | AGND       | 29     | GND        |
| 10     | CH3 IN HI  | 30     | PC+5V      |
| 11     | CH3 IN LO  | 31     | GND        |
| 12     | AGND       | 32     | Port B0    |
| 13     | D/A OUT 0  | 33     | Port B1    |
| 14     | D/A OUT 1  | 34     | Port B2    |
| 15     | AGND       | 35     | Port B3    |
| 16     | CAL        | 36     | Port B4    |
| 17     | GND        | 37     | Port B5    |
| 18     | TRIG IN    | 38     | Port B6    |
| 19     | SYNC       | 39     | Port B7    |
| 20     | CTR        | 40     | GND        |

Tabela 5.4. Oznake svih priključaka na modulu kod diferencijalnog priključivanja analognog signala

Izgled PLC-a je prikazan na slici 5.19.



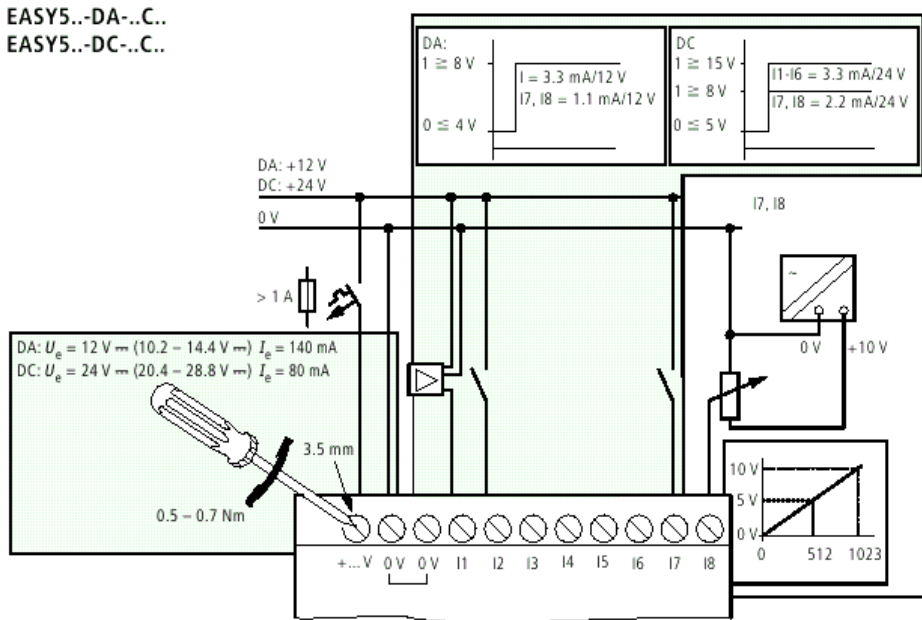
Slika 5.19. Izgled PLC-a Moeller Easy 512-DC-RC

Na prednjoj strani PLC-a se nalaze 4 tastera, DEL, ALT, ESC i OK, koji se koriste za programiranje i rad sa uređajem. Veliki okrugli taster se koristi za kretanje po menijima i dijagramima, a aktivna su mu četiri ruba (gore, dole, lijevo i desno). Taster OK se koristi za izbor označene funkcije menija.

Ovaj PLC se napaja sa 24 V= (za napajanje se može koristiti napojna jedinica Easy 200-POW), a posjeduje šest digitalnih ulaza (I1-I6) i dva 10-bitna analogna ulaza (I7, I8), te četiri relejna izlaza.

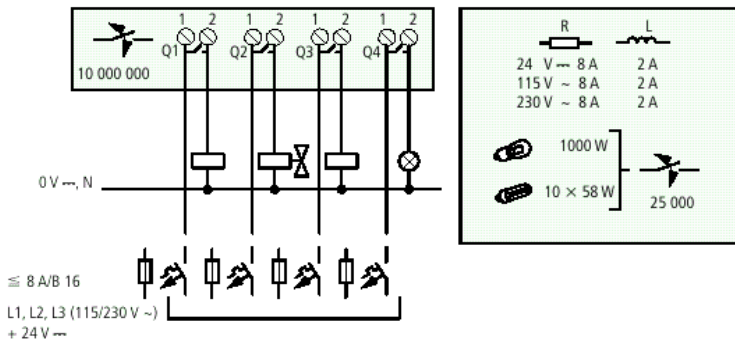
### 5.4.1. Povezivanje modula

Način povezivanja digitalnih i analognih ulaza je prikazan na slici 5.20., a način povezivanja relejnih izlaza na slici 5.21.



Slika 5.20. Povezivanje digitalnih i analognih ulaza PLC-a Moeller Easy 512-DC-RC

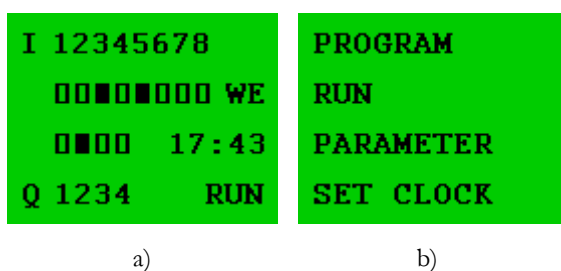
**EASY5...-R..**



Slika 5.20. Povezivanje relejnih izlaza PLC-a Moeller Easy 512-DC-RC

#### 5.4.2. Rad sa PLC-om i programiranje

Na slici 5.21.a) je prikazan izgled displeja u izvršnom modu, pri čemu se mogu vidjeti stanja ulaza (I) i izlaza (Q) PLC-a. Na pomenutoj slici se vidi da su ulazi 3 i 5 aktivirani (nalaze se na naponskom nivou 24V), a relej 2 je zatvoren. "RUN" pokazuje da uređaj trenutno izvršava program. Na modelima koji posjeduju sat je prikazan i dan (WE), te vrijeme. Pritiskom na taster OK se dobija prikaz glavnog menija (slika 5.21.b).



Slika 5.21. a) Izgled displeja u izvršnom modu b) prikaz glavnog menija

Korištenjem kursorskog tastera se bira akcija, a trenutno izabrana akcija je naznačena treptanjem. Značenje akcija glavnog menija je dato u tabeli 5.5. Akcija se bira pritiskom tastera OK.

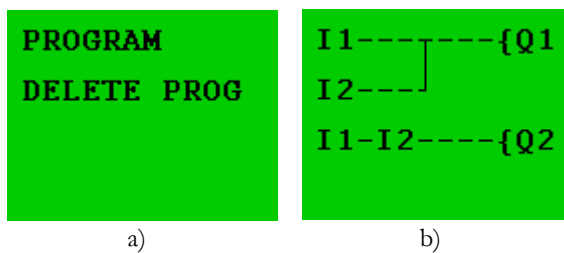
| <i>Akcija</i> | <i>Značenje</i>                                   |
|---------------|---------------------------------------------------|
| PROGRAM       | Meni PROGRAM, Programiranje PLC-a                 |
| RUN           | Pokretanje/zaustavljanje programa                 |
| PARAMETER     | Postavljanje vrijednosti brojača, tajmera itd.    |
| SET CLOCK     | Podešavanje sata (na modelima koji posjeduju sat) |

Tabela 5.5. Značenje akcija glavnog menija

Slika 5.22.a) prikazuje sadržaj menija PROGRAM, koji ima akcije PROGRAM za unošenje dijagrama, odnosno DELETE za brisanje dijagrama. Na slici 5.22.b) je kao primjer dat prikaz jednostavnog dijagrama koji realizira funkcije:

$$Q1 = I1 \vee I2$$

$$Q2 = I3 \wedge I4$$



Slika 5.22. a) Sadržaj menija PROGRAM b) Izgled jednostavnog dijagrama

Sa desne strane se nalaze varijable kojima se pridružuje vrijednost, dok je preostali dio rezervisan za ulaze, markere, tajmere i ožičenje. Svaku zasebnu granu dijagrama je moguće realizirati sa do tri kontakta.

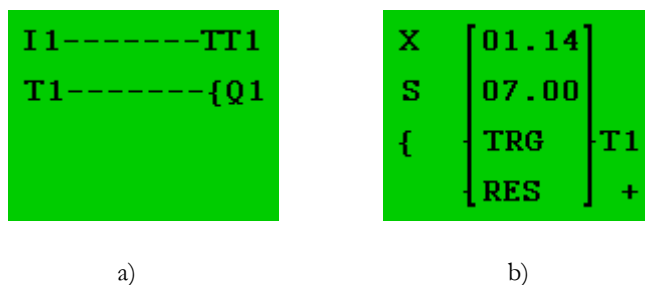
Da bi se unio dijagram sa slike, potrebno je uraditi sljedeće:

- Pritiskom na taster OK izabrati akciju PROGRAM
- Pomoću kursorskog tastera se postaviti u gornji lijevi ugao displeja i pritisnuti taster OK, nakon čega će se na mjestu kursora pojaviti oznaka I1
- Preći u desni gornji ugao displeja i pritisnuti taster OK, nakon čega će se na mjestu kursora pojaviti oznaka Q1
- Postaviti se za jedno mjesto desno od I1 i pritisnuti taster ALT da bi se aktiviralo ožičavanje, te kursorskim tasterom povezati I1 i Q1
- Pritisnuti taster ALT da bi se isključilo ožičavanje i postaviti se na početak drugog reda
- Pritisnuti taster OK, nakon čega će se na mjestu kursora pojaviti I1, te pritisnuti taster OK još jednom, te kursorski taster na gore da bi se na mjestu kursora dobila oznaka I2
- Pritisnuti taster ALT, te pomoću kursorskog tastera povezati I2 sa vezom u prvom redu
- Na opisani način realizirati drugu granu dijagrama
- Tasterom ESC se vratiti na glavni meni

Da bi se umjesto nekog kontakta koristila njegova negirana vrijednost, potrebno je



pritisnuti taster ALT. Ukoliko je potrebno promijeniti tip kontakta ili izlaza, nakon pritiska na taster OK i pojavljivanja oznake I1, koristiti kursori taster na gore. Tako npr. da bi se u dijagram ubacio tajmer T1, dovoljno je pritisnuti jednom kursori taster na gore, te taster OK za potvrđivanje. Na slici 5.23.a) je prikazan dijagram koji se aktivira kada ulazi I1 dospije na napon 24V, a izlaz tajmera se vodi na izlaz Q1. Nakon toga će se automatski pojaviti displej za postavljanje vrijednosti parametara tajmera (slika 5.23.b).



Slika 5.23. a) Upotreba tajmera u dijagramu b) Podešavanje parametara tajmera

Simbol gore lijevo označava tip tajmera (on-delay, off-delay, itd.-vidjeti uputstvo), pri čemu “X” označava on-delay tajmer (koji sa log 0 na log 1 prelazi nakon isteka vremena kašnjenja). “S” označava sekunde, a vrijednosti 1.14 i 7.00 predstavljaju proteklo vrijeme i postavljeno vrijeme tajmera.

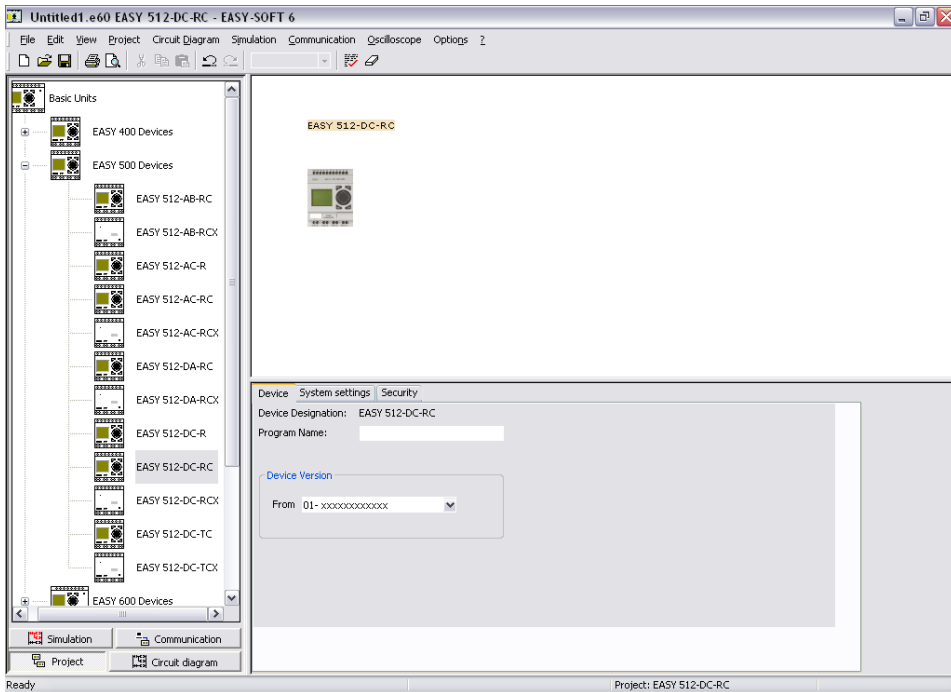
Detaljne upute za rad sa PLC-om Moeller 512-DC-RC se mogu pronaći u korisničkom uputstvu.

### 5.4.3. Razvojni alat EASY-SOFT-BASIC

Vrlo korisno pomagalo za rad sa Moeller PLC-ima predstavlja razvojni alat EASY-SOFT-BASIC, koji omogućava razvoj i simuliranje izvršavanja programa na Moellerovim PLC-ima, te prebacivanje programa sa PC-a na PLC i sa PLC-a na PC.

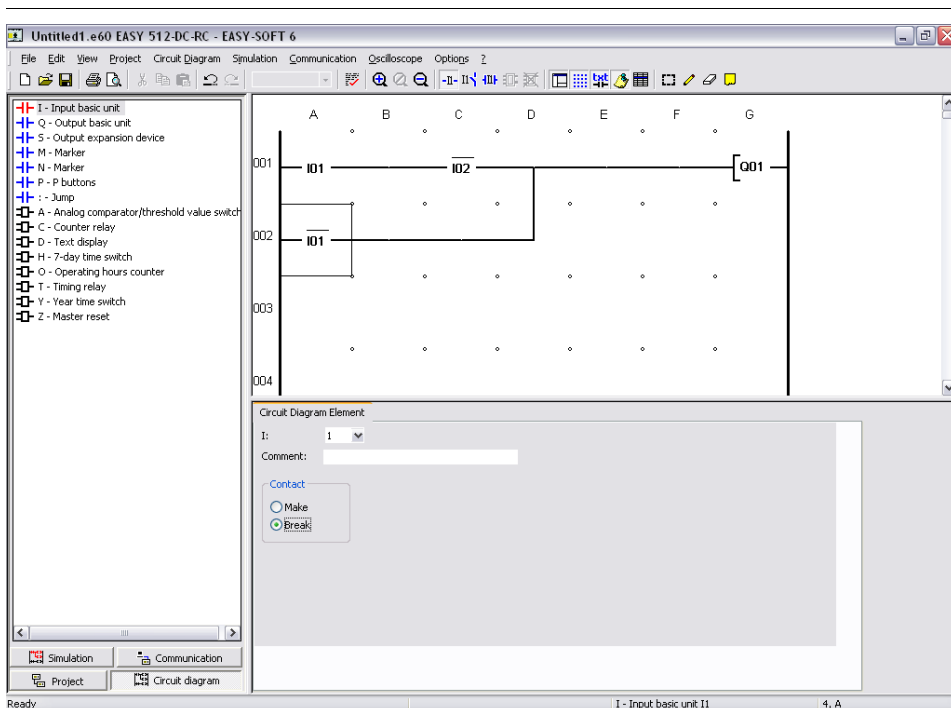
Osnovni prozor razvojnog alata je prikazan na slici 5.24. U lijevom dijelu prozora je moguće izabrati tip PLC-a koji se koristi za realizaciju projekta. Željeni tip PLC-a se koristeći drag-and-drop prevuče u gornji desni dio prozora, kao što je prikazano na slici. U donjem lijevom uglu se nalaze dugmad za izbor akcije:

- Project – detalji o projektu
- Circuit diagram – realizacija programa
- Simulation – simuliranje izvršavanja programa
- Connection – komuniciranje sa PLC-om (prebacivanje programa sa PLC-a i na PLC).



Slika 5.24. Izgled glavnog prozora EASY-SOFT-BASIC razvojnog alata

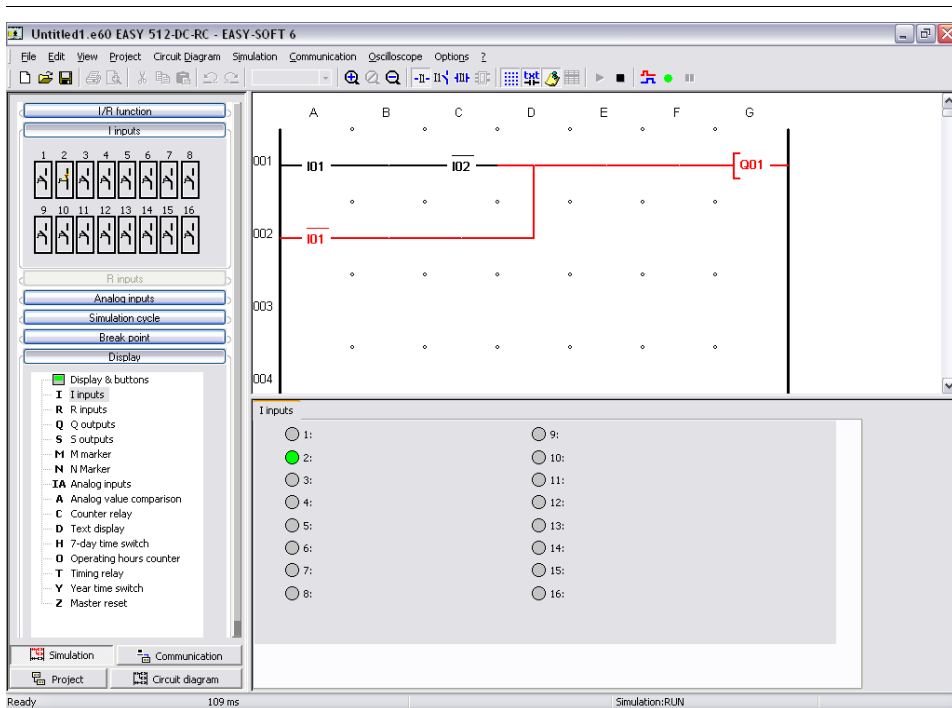
Sve ove akcije su dostupne i preko menija. Izgled forme za crtanje dijagrama (odnosno razvijanje programa) je prikazan na slici 5.25. U lijevom dijelu prozora se nalaze elementi dijagrama (ulazi, izlazi, markeri, tajmeri, brojači), dok je dijagram prikazan u gornjem desnom dijelu prozora. Elementi se koristeći drag-and-drop prevlače na željeno mjesto na dijagramu. Pri tome se za svaki element u donjem desnom dijelu prozora mogu podesiti parametri (broj ulaza, parametri tajmera i slično).



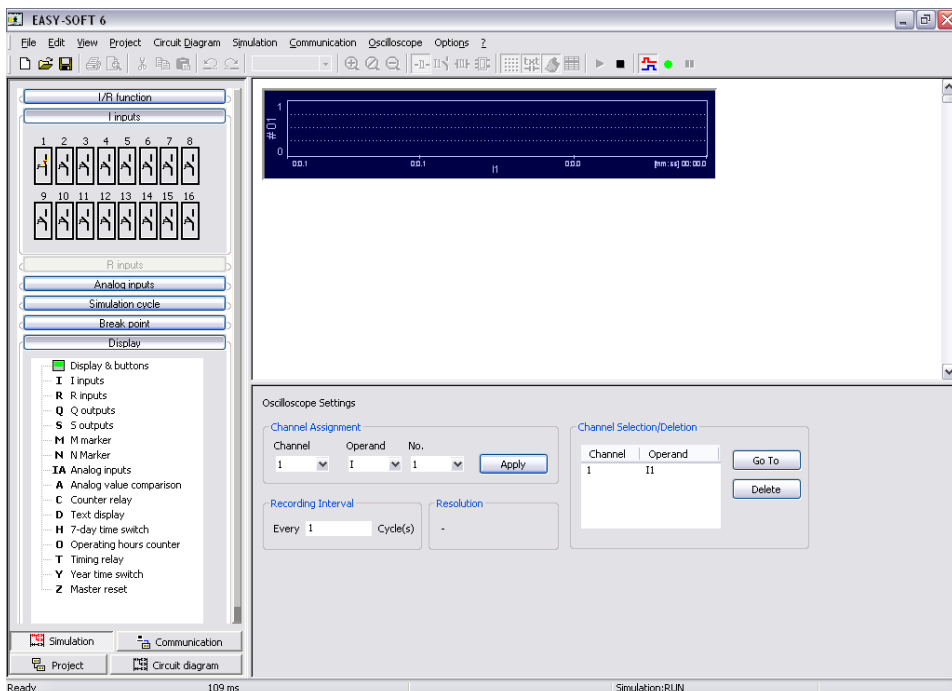
Slika 5.24. Izgled glavnog prozora EASY-SOFT-BASIC razvojnog alata

Na slici 5.25. je prikazan izgled prozora tokom odvijanja simulacije izvršavanja programa. U gornjem desnom dijelu prozora se vidi dijagram, pri čemu je na dijagramu crvenom bojom označen tok logičkog signala. U lijevom dijelu prozora se nalazi niz dugmadi, koja otvaraju različite kontrole za interakciju sa odvijanjem simulacije. Tako npr. klikom na dugme <Display> se otvaraju kontrole za izbor veličina čije vrijednosti će biti prikazivane u donjem desnom dijelu prozora. Na slici se vidi da su za prikaz odabrani ulazni kontakti, te su u donjem desnom dijelu prozora prikazani statusi raspoloživih ulaza PLC-a. Isto tako, dugme <Inputs> otvara kontrole za stanja ulaza, gdje je klikom na odgovarajući ulaz moguće promijeniti njegovo stanje. Slično vrijedi za ostale raspoložive kontrole. Na toolbar-u se nalaze kontrole za pokretanje i zaustavljanje simulacije.

Stanje veličina simulacije je, osim pomoću virtualnih LED i drugih indikatora, moguće prikazati i na virtualnom osciloskopu, što je prikazano na slici 5.26.

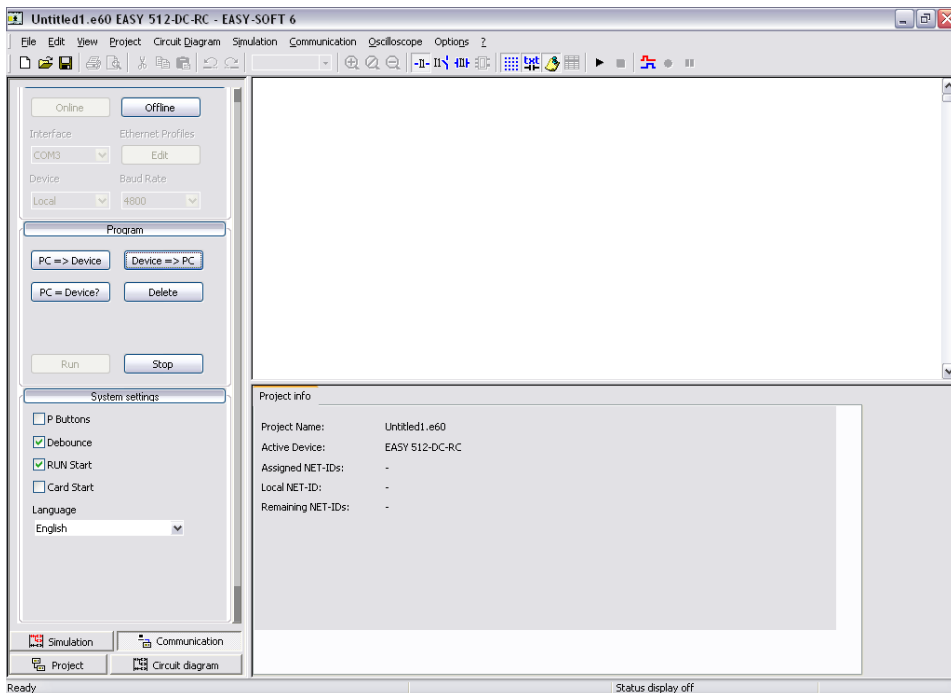


Slika 5.25. Simulacija izvršavanja programa



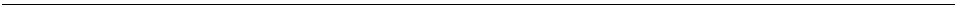
Slika 5.26. Prikaz vrijednosti veličina na virtualnom osciloskopu

Vrlo važnu mogućnost razvojnog alata EASY-SOFT-BASIC predstavlja mogućnost da se preko serijske veze ili preko mrežne konekcije razvijeni program prebaci na PLC, ili da se program sa PLC prebaci u razvojni alat (slika 5.27.).



Slika 5.27. Komunikacija između PC računara i PLC

U gornjem lijevom dijelu prozora se podešavaju parametri komunikacije. U srednjem lijevom dijelu prozora se odabire aktivnost, dok se u donjem lijevom dijelu podešavaju sistemski parametri PLC-a.



---

## L i t e r a t u r a

- [1] Hebibović, Mujo: “Teorija automatskog upravljanja”, Elektrotehnički fakultet u Sarajevu, Sarajevo, 2003.
- [2] Reuter, Manfred, Zacher, Serge: “Regelungstechnik für Ingenieure”, Viewegs Fachbücher der Technik, Wiesbaden, 2004.
- [3] Wendt, Wolfgang, Lutz, Holger: “Taschenbuch der Regelungstechnik”, Verlag Harri Deutsch, 3. Auflage, Frankfurt am Main, 2000.
- [4] The Mathworks, Inc.: “Getting Started in Matlab”, Product documentation, The Mathworks Inc., 2005
- [5] National Instruments: “Getting Started with LabVIEW”, Product documentation, April 2003 Edition, 2003
- [6] National Instruments: “LabVIEW Quick Start Guide”, Product documentation, February 1999 Edition, 1999
- [7] Bückert Fluid Control Systeme: “Digital Industrial Controller Type 1110, Operating Instructions”, Product documentation, Bückert Werke GmbH, 2002
- [8] Meilhaus Electronic: “ME-RedLab 1208FS, Bedienungsanleitung”, Product documentation, Meilhaus Electronic, 2006
- [9] Moeller: “Installation Instructions”, Product documentation, Moeller GmbH, 2004
- [10] Wehner, Fred: “Programming the Klockner Moeller EASY mini-PLC”, Klockner Moeller EASY512 & EASY719: Programming instructions for the mini-PLC
- [11] Bischoff, H., Hofmann, D., Terzi, E. v.: “Process Control System, Regelung von Temperatur, Durchfluß und Füllstand, Arbeitsbuch”, Festo Didactic KG, Esslingen, 1997
- [12] B+B Termotechnik: “Pt 100 Sensors, Technical Notes”, B+B Termotechnik GmbH, Donaueschingen, Germany
- [13] B+B Termotechnik: “Probes for the HVAC Industry”, B+B Termotechnik GmbH, Donaueschingen, Germany
- [14] Tecflow International: “Specifications of the IR-Opflow Turbine Flowmeter”, Tecflow International, Wijehen, The Netherlands
- [15] JLC International: “IR-Opflow Flow Meters”, JLC International, Inc., New Britain

---

Za pripremu skripte korišten je open source software:

- SuSe Linux 10.0
- OpenOffice.org Writer 2.0.0.1
- Inkscape 0.44
- The Gimp 2.2.8
- KPDF 0.4.2