Innovative Teaching Approaches in development of Software Designed Instrumentation and its application in real-time systems

Praktikum iz merno-akvizicionih sistema

Korišćenje petlji i struktura za odlučivanje

Co-funded by the Erasmus+ Programme of the European Union

Innovative Teaching Approaches in development of Software Designed Instrumentation and its application in real-time systems

Faculty of Technical Sciences

Ss. Cyril and Methodius University
Faculty of Electrical Engineering and Information Technologies

Zagreb University of Applied Sciences

School of Electrical Engineering
University of Belgrade

Faculty of Physics
Warsaw University of Technology

Co-funded by the Erasmus+ Programme of the European Union

Predgovor

„Praktikum iz merno-akvizicionih sistema“ je udžbenik nastao u okviru rada na projektu Erasmus+ KA2 2018-1-RS01-KA203-000432 strateška partnerstva u okviru visokog obrazovanja: „**Innovative Teaching Approaches in development of Software Designed Instrumentation and its application in real-time systems (ITASDI)**“, 2018-2019, http://itasdi.uns.ac.rs/. Udžbenik je prevashodno namenjen studentima druge godine Elektrotehničkog fakulteta Univerziteta u Beogradu koji prate kurs pod istoimenim nazivom „Praktikum iz merno-akvizicionih sistema“ (šifra predmeta 13E053PMS), ali i drugima koji žele da steknu osnovna znanja i praktična iskustva u dizajniranju merno-akvizicionih sistema.

U uvodnom delu knjige je objašnjen koncept sofverski dizajnirane instrumentacije, a potom se čitaoci uvode u konkretne realizacije i primene ovakvih sistema kroz detaljno („korak po korak“) objašnjene primere. Rešenja svih primera su dostupna na sajtu ITASDI projekta. Osim elementarnih zadataka, knjiga sadrži i tzv. „zadatke za samostalni rad“ koji imaju za cilj povezivanje stečenog osnovnog znanja i razvoj veština dizajniranja složenijih aplikacija. Predlozi rešenja autora „zadataka za samostalni rad“ su takođe dostupna na sajtu ITASDI projekta.

U prvom delu knjige (Lekcije 1-9) autori su ilustrovali specifičnosti i primenu komercijalnog *LabVIEW* (*National Instruments*, SAD) softverskog okruženja u akviziciji signala i slike. U ovom delu su objašnjene osnove *data flow* principa programiranja, načini debagovanja aplikacija, primene različitih vrsta petlji i različitih struktura podataka, ilustrovane su mogućnosti modularnog programiranja i različiti pristupi upisa podataka u datoteke. Osim osnova *LabVIEW* programiranja, u ovom delu su objašnjene i napredne tehnike paralelnog programiranja i događajem vođenog programiranja kao i način kreiranja distribucije *LabVIEW* aplikacija.

U drugom delu knjige (Lekcije 10-14) su objašnjeni primeri korišćenja „otvorenog“ harvera (eng. *open source hardware*) i softvera „otvorenog“ kôda (eng. *open source software*) za sofverski dizajniranu instrumentaciju. U ovom delu su ilustrovane primene *Arduino* mikrokontrolera i kompatibilnih senzora kroz primere sistema koji koriste njegove analogne i digitalne portove kao i serijsku komunikaciju. Za kreiranje grafičkog korisničkog interfejsa u zadacima je odabrana *PyQt* biblioteka *Python* programskog jezika.

Na ovakav koncept knjige koja se jednim delom odnosi na primene komercijalno dostupnog sotvera, a drugim delom na primenu „otvorenih“ platformi su u velikoj meri uticali Akademik Dejan Popović, redovni profesor u penziji Elektrotehničkog fakulteta Univerziteta u Beogradu (na prvi deo) i dr Predrag Pejović, redovni profesor Elektrotehničkog fakulteta Univerziteta u Beogradu (na drugi deo).

Autori se zahvaljuju univerzitetskim partnerskim institucijama koje su učestvovale na ITASDI projektu (Fakultetu tehničkih nauka - Univerzitet u Novom Sadu, Tehničkom veleučilištu u Zagrebu – Univerzitet za primenjene nauke, Fakultetu za elektrotehniku i informacione tehnologije – Univerzitet Sv. Kiril i Metodije u Skoplju, Fakultetu za fiziku Univerziteta za tehnologije u Varšavi) na saradnji tokom

pripreme ovog udžbenika, kao i na saradnji tokom organizacije takmičenja *Balkan Open Competition in Software-designed Instrumentation*, http://blsc.etf.rs/.

Autori se posebno zahvaljuju recenzentima udžbenika, dr Mirjani Simić-Pejović, vanrednom profesoru Elektrotehničkog fakulteta Univerziteta u Beogradu i dr Nenadu Miljiću, vanrednom profesoru na Mašinskom fakultetu Univerziteta u Beogradu na strpljenju i podršci pri pisanju ovog udžbenika.

Beograd, oktobar 2019. *Autori*

Softverski dizajnirana instrumentacija

Razvoj računarskih tehnologija u poslednjih pola veka značajno je uticao na promenu uloge softvera u sistemima za testiranje i merenje. Tradicionalni instrumenti, namenski dizajnirani za merenje određenih fizičkih veličina, su 70-ih godina prošlog veka bili potisnuti programabilnim instrumentima sa GPIB (*General Purpose Interface Bus*) komunikacijom. Desetak godina kasnije su se pojavili i prvi merni računarski sistemi koji su bili bazirani na standardnom personalnom kompjuteru i imali u sebe ugrađene akvizicione kartice sa GPIB interfejsom pri čemu je način rada računarski-baziranog mernog sistema u potpunosti bio definisan softverom na računaru. Ovakvim konceptom je omogućeno da se paleta tradicionalnih instrumenata, fizički prisutnih na laboratorijskom stolu, zameni jednim personalnim računarom sa akvizicionom karticom i paletom softvera dizajniranih za specifične namene. Dalji razvoj pre svega u oblasti integrisanih tehnologija, uticao je na poboljšanje performansi akvizicionih kartica i njihovih interfejsa (*PCI, PCMCIA, RS 232, RS 485, USB, PXI, FireWire, Bluetooth, Ethernet, LAN, CAN,* I^2C itd.), ali je koncept „softverom definisane instrumentacije“, „softverski dizajnirana instrumentacija“ ili „virtuelne instrumentacije“ u kojoj je softver krucijalan deo merne instrumentacije ostao nepromenjen do danas.

Standardna arhitektura modernog mernog sistema obuhvata:

1. senzore koji mernu veličinu pretvaraju u električni signal (npr. termistori za merenje temperature, fotootpornici za merenje nivoa osvetljenosti, kapacitivni pretvarači za merenje pomeraja i sl.)
2. kola za kondicioniranje (pojačavači, filtri, kola za korekciju i sl.) koja omogućavaju da se električni signal sa izlaza senzora prilagodi tako da se performanse akvizicione kartice maksimalno iskoriste
3. akviziciona kartica za diskretizaciju signala u skladu sa teoremom odabiranja (frekvencija odabiranja može biti hardverski ili softverski kontrolisana)
4. memorija računara gde se po FIFO (*First In First Out*) principu smeštaju i čitaju odbirci signala
5. računar sa instaliranim drajverima za akvizicionu karticu i softverom koji omogućava:
    a. interfejs ka korisniku mernog sistema,
    b. obradu i vizuelizaciju podataka,
    c. snimanje, tzv. logovanje podataka,
    d. dalji prenos podataka (lokalna mreža, Internet).

Pri projektovanju savremenih merno-akvizicionih sistema, a u skladu sa prirodom merne veličine i namenom dizajniranog sistema, potrebno je voditi računa o sledećim koracima:

1. izbor senzora prema sledećim kriterijumima:
    a. karakteristikama senzora (osetljivost, ulazni opseg, stabilnost, ponovljivost i reproducibilnost, rezolucija, linearnost, histerezis, vreme odziva, frekvencijski odziv)

   b. ekonomskim faktorima (cena, dostupnost, vreme života)
   c. faktorima okruženja (opseg temperature, vlažnost, korozija, zaštita od prekoračenja opsega, osetljivost na elektromagnetnu interferenciju, čvrstina, veličina, potrošnja energije, mogućnost samokalibracije)
2. projektovanje kondicioniranja signala prema sledećim kriterijumima:
   a. prilagođenje propusnog opsega i pojačanja/slabljenja frekvencijskom i amplitudskom opsegu signala koji se meri i karakteristikama akvizicione kartice
   b. eliminacija artefakta (usled drifta u električnim kolima, interferencije, magnetske indukcije)
   c. niskošumni zahtevi
   d. mala merna nesigurnost
   e. mala potrošnja kola
3. izbor akvizicione kartice ili mikrokontrolera sa analogno-digitalnim (A/D) konvertorom prema sledećim kriterijumima:
   a. broj ulaza/izlaza i tip ulaza/izlaza (analogni/digitalni)
   b. paralelni A/D konvertori (po jedan za svaki od analognih ulaza) ili multipleksiranje analognih kanala na jednom A/D konvertoru
   c. referenciranje ulaznih kanala (diferencijalno merenje, merenje u odnosu na masu ili u odnosu neku drugu referentnu tačku)
   d. brzina i rezolucija analogno-digitalne konverzije
4. izbor računara u skladu sa zahtevima koja se odnose na merenja u realnom vremenu
5. izbor softverskog okruženja za dizajniranje instrumenta prema sledećim kriterijumima:
   a. isplativost cene softverskog okruženja (primena softvera otvorenog kôda, primena komercijalnih softvera)
   b. pouzdanost okruženja
   c. potreba korisnika da instrument radi pod određenim operativnim sistemom
   d. cena održavanja i nadgradnje aplikacije
   e. cena distribucije aplikacije.

U narednim poglavljima će biti detaljno opisani primeri jednostavnih merno-akvizicionih sistema koji koriste: 1) NI USB A/D karticu i komercijalno *LabVIEW* softversko okruženje i 2) hardver otvorenog kôda (*Arduino* mikrokontroler) i *Python* softver otvorenog kôda.

DEO I

Primena LabVIEW softverskog alata u merno-akvizicionim sistemima

Lekcija 1 – *LabVIEW* okruženje

Cilj

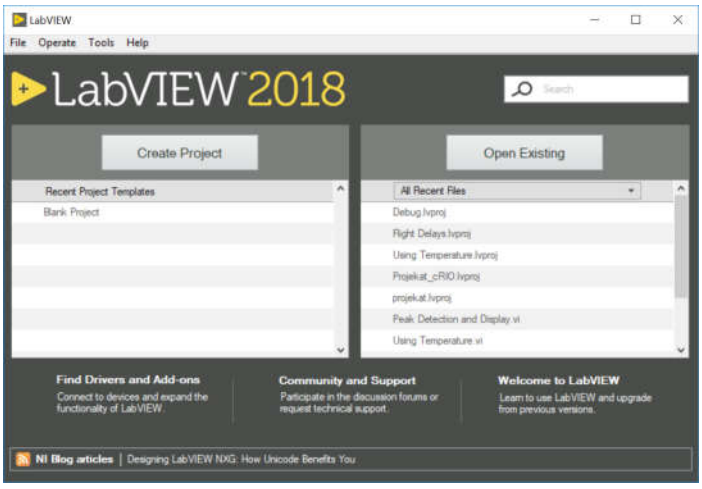
Cilj lekcije je da studente upozna sa:

- pokretanjem programa u *LabVIEW* okruženju
- osnovnim komponentama i alatima *LabVIEW* okruženja
- tehnikama debagovanja u *LabVIEW* okruženju
- kreiranjem projektne datoteke.

1.1 Pokretanje LabVIEW programa

LabVIEW programi imaju ekstenziju **.vi** i nazivaju se **V**irtuelni **I**nstrumenti (VI) jer svojim interfejsom i mogućnostima primene podsećaju na klasične instrumente (osciloskope, multimetre, kontrolere itd.).

LabVIEW softver se pokreće izborom opcije: ***Start»All programs»National Instruments»LabVIEW 2018 (32-bit)*** ili klikom na ***LabVIEW*** prečicu na *Desktop*-u. Inicijalni NI LabVIEW prozor je prikazan na Sl. 1.1.1.

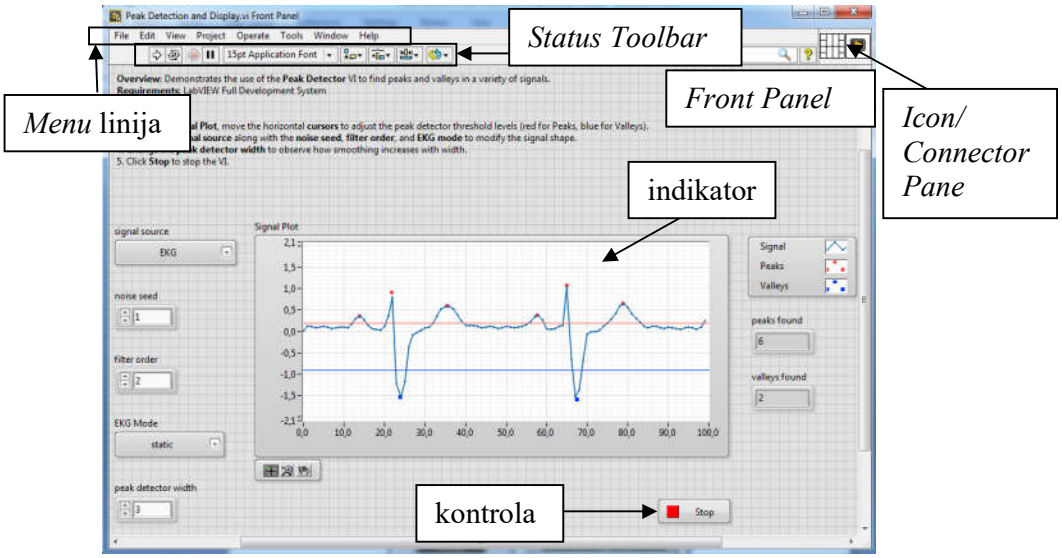


Sl. 1.1.1. *LabVIEW* inicijalni prozor

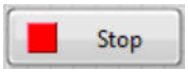
Pokrenuti primer *Peak Detection and Display.vi* na sledeći način:

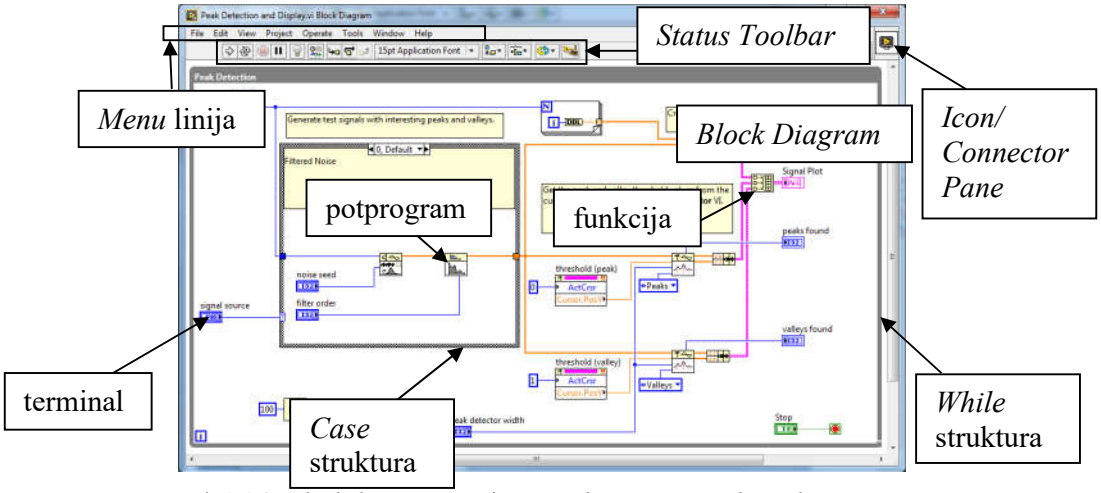
1. U prozoru na Sl. 1.1. selektovati opciju ***Help... »Find Examples***.
2. Selektovati ***Analysis, Signal Processing and Mathematics***, potom ***Signal Processing***, i na kraju ***Peak Detection and Display.vi.*** Pojaviće se prozor kao na Sl. 1.2. Ovaj prozor predstavlja *Front panel*, tj. korisnički interfejs za pozvani program. Na korisničkom interfejsu se nalaze: 1) **kontrole**

(promenljive koje omogućavaju korisniku da zada vrednosti na korisničkom interfejsu) i 2) **indikatori** (promenljive na kojima se prikazuju informacije i koje korisnik ne može da menja). Na *Front Panel*-u uočiti gde se nalaze *Menu* linija i *Status Toolbar*, Sl. 1.1.2.



Sl. 1.1.2 Front panel za primer *Peak Detection and Display.vi*

3. Pritiskom na ***Run*** dugme  pokrenuti program. Menjati vrednosti **kontrola** na interfejsu (*signal source*, *noise seed*, *filter order*, *EKG Mode*, *peak detector width*) i posmatrati promene na grafičkom **indikatoru** (*Signal Plot*, *peaks found*, *valleys found*).
4. Pritiskom na dugme ***Stop***  na *Front Panel*-u zaustaviti program.
5. Pokrenuti ponovo izvršavanje programa pritiskom na ***Run*** dugme  . Potom nasilno prekinuti izvršavanje programa pritiskom na dugme ***Abort Execution***  .
6. Otvoriti prozor *Block diagram* izborom ***Window»Show Block Diagram***, Sl. 1.1.3. Na *Block Diagram*-u uočiti gde se nalaze *Menu* linija i *Status Toolbar*, Sl. 1.1.3.



Sl. 1.1.3. *Block diagram* za primer *Peak Detection and Display.vi*

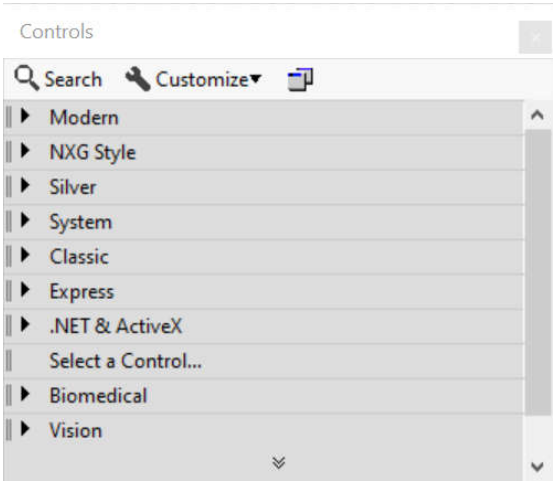
7. Uočiti da *Block diagram* sadrži terminale (tj. promenljive), funkcije, potprograme, strukture prikazane na Sl. 1.3. Uočiti da su terminali i žice različitih boja (npr. plave boje su celobrojne promenljive, roze boje su nizovne promenljive, zelene boje su logičke promenljive itd.). Uočiti i da se debljina žica razlikuje (tanke linije odgovaraju skalarnim promenljivama, deblje linije odgovaraju nizovima i matricama).
8. Zatvoriti program bez čuvanja eventualnih izmena.

NAPOMENA: Svaki *LabVIEW* VI sadrži tri osnovna dela: 1) ***Front panel*** (korisnički interfejs), 2) ***Block diagram*** (programski kôd) i 3) ***Icon/Connector pane*** (ikona i konektor) – gornji desni ćošak *Front Panel*-a i *Block Diagram*-a - omogućava da se program koristi kao potprogram u nekom drugom VI-u (primena je detaljno objašnjena u Lekciji 5).

1.2 Kreiranje i debagovanje programa

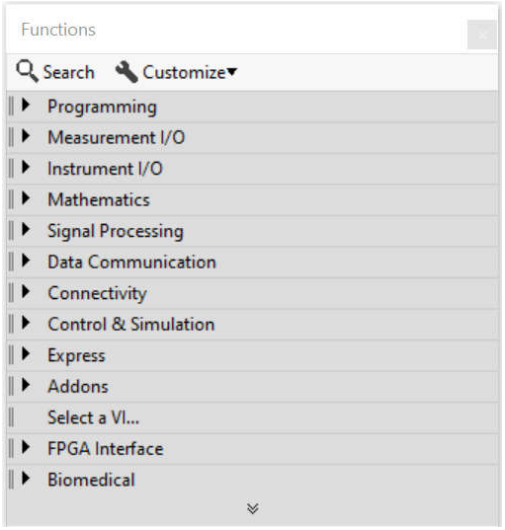
Zadatak 1.2.1.
Kreirati program koji vrednost numeričke kontrole prikazuje na grafičkom indikatoru. *Front panel* (korisnički interfejs) programa obojiti po želji. Omogućiti da se pri postavljanju miša iznad numeričke kontrole pojavi natpis „Temperatura“, kao i da se u dokumentacije numeričke kontrole pojavi objašnjenje „Ova promenljiva oznacava temperaturu“.

1. Otvoriti nov **.vi** program izborom opcije ***File»New VI***. Pojaviće se dva prozora: *Front Panel* i *Block Diagram*. Ova dva prozora se mogu istovremeno posmatrati izborom opcije ***Window»Tile Up and Down*** ili izborom opcije ***Window»Tile Left and Right***.
2. Otvoriti paletu ***Controls*** selekcijom ***View»Controls Palette*** na *Front Panel*-u, Sl. 1.2.1.



Sl. 1.2.1. Paleta *Controls*

3. Otvoriti paletu ***Functions*** selekcijom ***View»Functions Palette*** u *Block Diagram*-u, Sl. 1.2.2.



Sl. 1.2.2. Paleta *Functions*

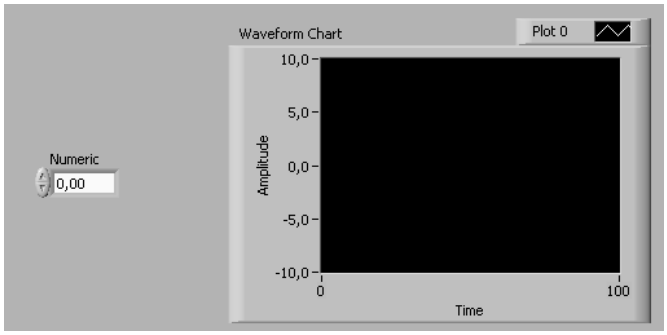
4. Otvoriti paletu ***Tools*** selekcijom ***View»Tools Palette*** ili na *Front Panel*-u ili u *Block Diagram*-u, Sl. 1.2.3. Ova paleta služi za manipulaciju elementima u oba prozora.

Sl. 1.2.3. Paleta *Tools*

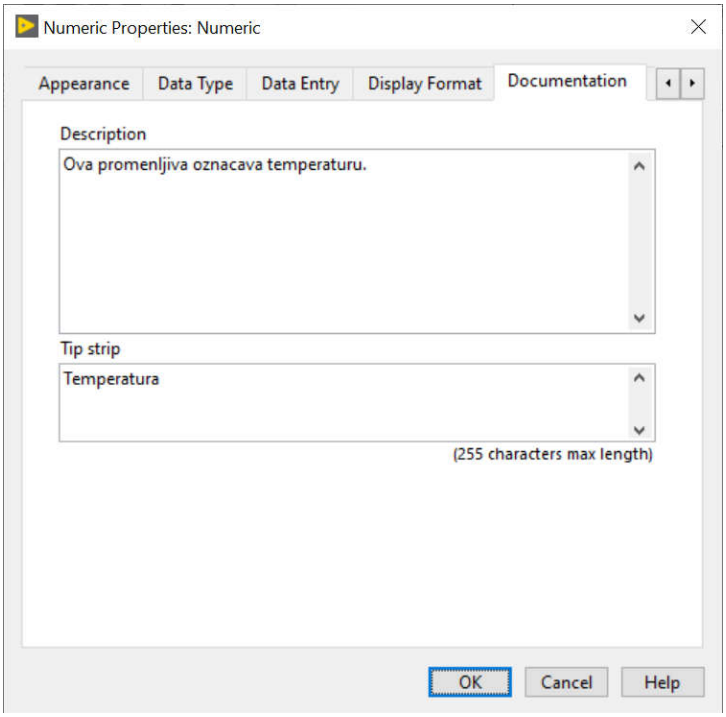
NAPOMENA: Uočiti da je paleta ***Tools*** dostupna i na *Front Panel*-u i u *Block Diagram*-u za razliku od paleta ***Controls*** i ***Functions*** koje su dostupne samo u jednom od ova dva prozora.

5. Iz ***Controls»Modern*** palete izabrati kontrolu ***Numeric»Numeric Control*** i postaviti je na *Front panel*. U *Block Diagram*-u će se pojaviti odgovarajući terminal.
6. Iz ***Controls»Modern*** palete izabrati indikator ***Graph»Waveform Chart*** i postaviti ga na *Front panel*, Sl. 1.2.4. U *Block Diagram*-u će se pojaviti odgovarajući terminal.
7. Desnim mišem kliknuti na *Waveform Chart* u *Font panel*-u ili *Block Diagram*-u. Izabrati opciju *Properties* i uočiti opcije (*Appearance*, *Data Type*, *Data Entry*, *Display Format*, *Documentation*) koje okruženje nudi za podešavanje izgleda ovog indikatora (prikaz, tip podataka, ograničenja promenljive, prikaz formata, dokumentacije, respektivno). U sekciji *Documentation* uneti natpise kao na

Sl. 1.2.5. Nakon unosa natpisa pritisnuti CTRL+H kako bi se prikazao „prozor za pomoć“



Sl. 1.2.4 Izgled *Front Panel*-a sa numeričkom kontrolom *Numeric* i grafičkim indikatorom *Waveform Chart*



Sl. 1.2.5. Podešavanje natpisa numeričke kontrole

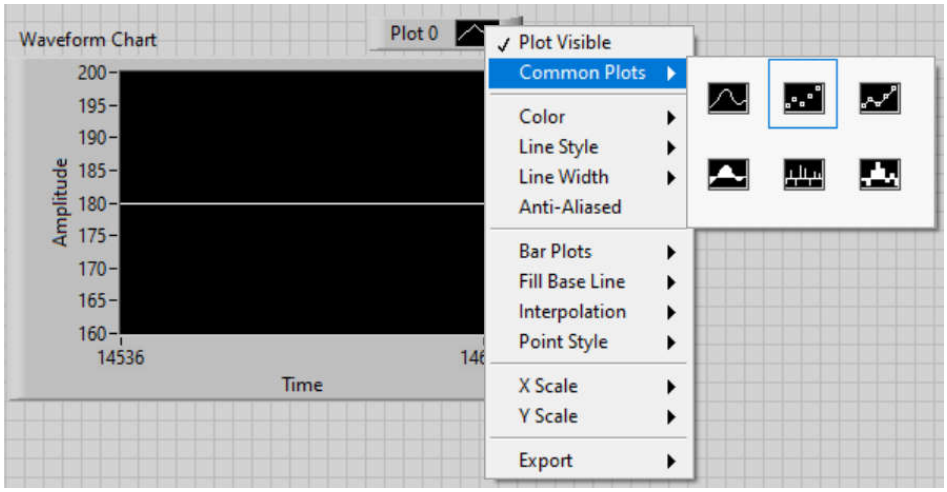
NAPOMENA1: Svaka kontrola i svaki indikator poseduju opcije podešavanja koje se pozivaju desnim klikom miša u *Front panel*-u, tj. *Block Diagram*-u. Programsko podešavanje opcija kontrola i indikatora će biti detaljno objašnjeno u Lekciji 3.4.

NAPOMENA2: Svaka promenljiva, funkcija i potprogram ima mogućnost prikaza „prozora za pomoć“ u kome se nalazi opis. Za funkcije i potprograme se u ovom prozoru nalazi i raspored ulaznih i izlazlanih promenljivih.

8. Iz *Tools* paleti izabrati žicu tj. ***Wiring Tool***  i povezati izvučenu kontrolu i indikator, Sl. 1.2.6. Uočiti da kontrola ima malu crnu strelicu sa desne strane terminala, a da indikator ima malu crnu strelicu sa leve strane terminala što sugeriše na „čitanje“ i „upisivanje“ podataka, respektivno. Uočiti da je boja terminala i žica za numeričku kontrolu i indikator narandžasta (narandžasta boja je rezervisana za *floating-point* tip podataka).

Sl. 1.2.6. Izgled *Block Diagram*-a sa numeričkom kontrolom *Numeric* i grafičkim indikatorom *Waveform Chart*

7. Pritiskom na ***Continuous Run Button*** , program će ući u kontinualan môd izvršavanja, sve dok se ne pritisne dugme ***Abort Execution*** . Dok se program izvršava menjati vrednost kontrole pomoću ***Operating Tool***-a i pratiti promenu na indikatoru. Promeniti način prikaza tačaka na grafiku sa „linije“ na „tačkice“ desnim klikom miša na *Plot* opciju *Waveform Chart*-a kao na Sl. 1.2.7. **NAPOMENA**: pri „linijskom“ prikazivanju vrednosti na grafiku, program spaja linijom diskretne vrednosti.

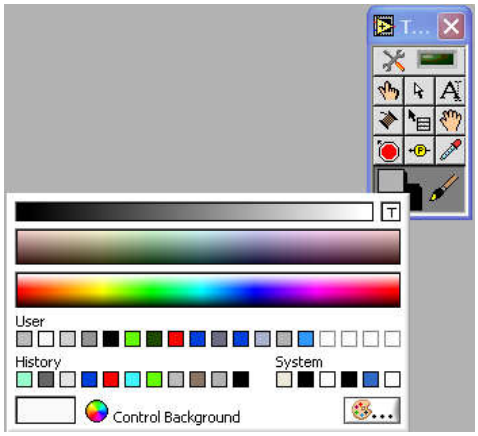


Sl. 1.2.7. Promena načina prikaza vrednosti na grafiku sa „linije“ na „tačkice“

8. Iz *Tools* palete izabrati ***Color Copy Tool*** i kliknuti crnu površinu *Waveform Chart* indikatora na *Front Panel*-u. Odgovarajuće boje (siva za okvir *Waveform Chart*-a i crna za površinu grafika) će se pojaviti u ***Coloring Tool***-u palete *Tools*.
9. Iz *Tools* palete izabrati ***Coloring Tool*** i po želji izmeniti boje interfejsa na sledeći način: 1) na jedno od dva polja ***Coloring Tool***-a za boju kliknuti levim tasterom miša, 2) iz ponuđene palete boja izabrati željenu, Sl. 1.2.8, 3) levim tasterom miša kliknuti na površinu koju želite drugačije da obojite.
10. Sačuvati program kao *kontrola_indikator.vi*.

NAPOMENA 1: Opcija u paleti boja *Coloring Tool*-a, Sl. 1.2.8. služi za postavljanje transparentnosti.

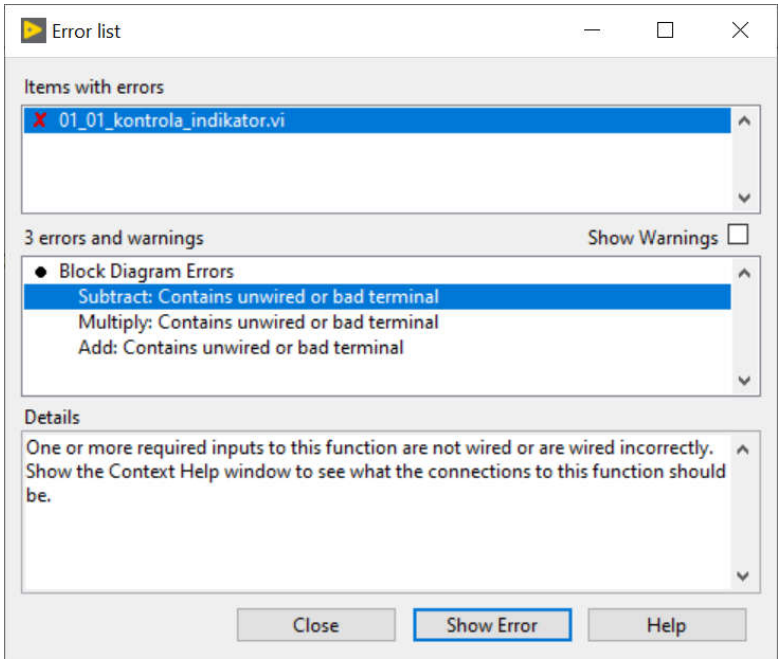
NAPOMENA 2: Selekcija i promena veličine elemenata se vrši pomoću alatke *Positioning/Resizing Tool* . Brisanje elementa ili žice se vrši tako što se element ili žica selektuju, a potom pritisne taster *Delete* na tastaturi. Automatska selekcija alatki u *Tools* paleti se podešava klikom na opciju *Automatic Selection Tool* u *Tools* paleti.



Sl. 1.2.8. Selekcija boje u *Tools* paleti

Zadatak 1.2.2.
Modifikovati program iz prethodnog zadatka tako da na osnovu zadate temperature T i vrednosti konstanti R_0, T_0 i k izračunava vrednost $R=R_0(1+k(T-T_0))$, a rezultat prikazuje na grafičkom indikatoru.

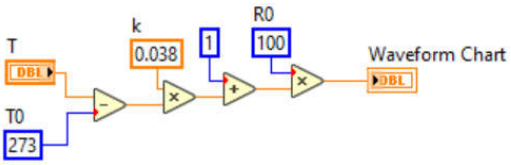
1. Obrisati žicu koja povezuje *Numeric* kontrolu i *Waveform Chart* indikator tako što se najpre pomoću alatke *Positioning/Resizing Tool* selektuje žica, a potom pritisne dugme *Delete*.
2. U paleti ***Functions»Programming»Numeric*** pronaći funkcije *Add*, *Subtract* i *Multiply*, kao i numeričku konstantu *Numeric Constant*. Postaviti ih na *Block Diagram*. Uočiti da se u *Status Toolbar*-u pojavila tzv. slomljena strelica koja ukazuje na grešku u kôdu. Levim tasterom miša kliknuti na slomljenu strelicu. Pojaviće se prozor sa listom grešaka kao na Sl. 1.2.9.



Sl. 1.2.9. Lista grešaka

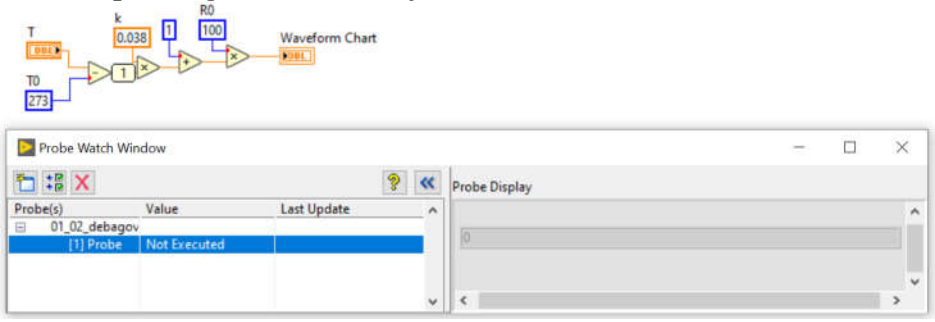
3. Pogledati „prozore za pomoć“ za navedene elemente pomoću CTRL+H opcije. Povezati programski kôd kao na Sl. 1.2.10. Pomoću alatke *Labeling Tool* uneti vrednosti konstanti u *Block Diagram*-u kao i nazive labela *T*, *T0*, *k*, *R0*.

NAPOMENA: Pritiskom na *CTRL+Space* je moguće pozvati meni za brzo traženje funkcije prema njenom imenu (*Quick Drop* tj. brzo dohvatanje).



Sl. 1.2.10.

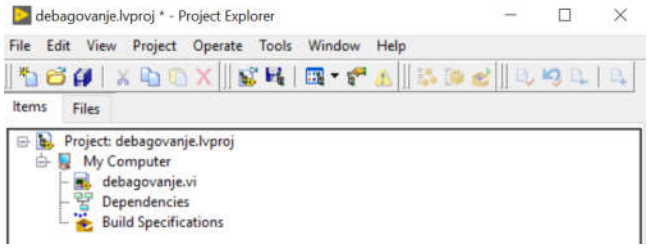
4. Pritiskom na opciju *Clean Up Diagram* u *Status Toolbar*-u *Block Diagram*-a obezbediti automatsko sređivanje programskog kôda. **NAPOMENA**: moguće je selektovati deo programskog kôda i samo na njega primeniti opciju „sređivanja“.
5. Pokrenuti izvršavanje programa pritiskom na dugme ***Continuous Run Button***. Uključiti opciju *Highlight Execution* i uočiti kako se u *Block Diagram*-u sada duž žica pojavljuju međurezultati. **NAPOMENA**: *LabVIEW* programiranje spada u kategoriju *dataflow* programiranja, tj. *LabVIEW* funkcije i potprogrami počinju da se izvršavaju tek kada su sve ulazne promenljive funkcije ili potprograma dostupne.
6. Iz *Tools* palete izabrati alatku *Probe Data* i postaviti je na žicu iza oduzimanja vrednosti *T* i *T0*. Pojaviće se *Probe Watch* prozor kao na Sl.1.2.11. Posmatrati kako se vrednost na izlazu funkcije za oduzimanje menja dok se na *Front Panel*-u pomoću ***Operating Tool***-a menja vrednost za *T*.



Sl. 1.2.11. *Probe Watch* prozor

7. Zaustaviti izvršavanje programa pritiskom na dugme ***Abort Execution***.
8. Uključiti opciju *Retain Wire Values* u *Status Toolbar*-u *Block Diagram*-a. Pokrenuti izvršavanje programa pritiskom na dugme ***Continuous Run Button***. Promeniti vrednost numeričke kontrole *T*. Zaustaviti izvršavanje programa pritiskom na dugme ***Abort Execution***. Iz *Tools* palete izabrati alatku *Probe Data* i postaviti po jednu na izlazu svake od funkcija u programu. Uočiti da je zahvaljući uključenju opcije *Retain Wire Values* sada moguće pratiti poslednje izračunate vrednosti na izlazu svake od funkcija i nakon završetka rada programa.

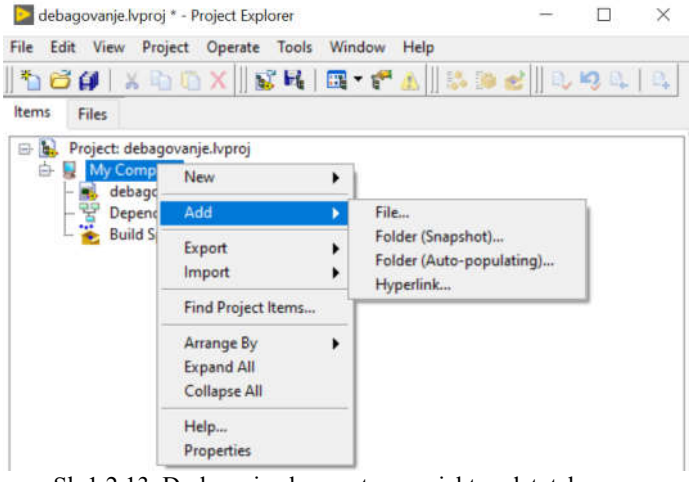
9. Iz *Tools* palete izabrati alatku *Breakpoint* . Postaviti je na proizvoljnu žicu u *Block Diagram*-u. Pokrenuti izvršavanje programa pritiskom na dugme ***Continuous Run Button*** . Primetiti kako će izvršavanje kôda biti pauzirano na mestu gde je „tačka prekida“ postavljena. Pritiskom na opciju *Continue* u *Status Toolbar*-u *Block Diagram*-a nastaviti izvršavanje programa.
10. Okruženje omogućava tzv. „korak po korak“ debagovanje. Za pokretanje ove vrste testiranja izabrati opciju *Start Single Stepping* u *Status Toolbar*-u *Block Diagram*-a. Pomoću *Step Into* i *Step Over* opcija je moguće „ući u“ funkciju i „završiti“ izvršavanje pojedinih funkcija respektivno, a pomoću opcije *Finish Block Diagram* se završava izvršavanje programa. Isprobati navedene opcije i završiti izvršavanje programa.
11. Sačuvati program kao novi program *debagovanje.vi*.
12. U praksi, rešenja obično sadrže osim glavnom programa i niz potprograma i prateće dokumentacije. Zbog toga većina softverskih okruženja, uključujući i *LabVIEW*, nudi čuvanje rešenja u vidu tzv. projektnih datoteka (eng. *project file*). Da bi se trenutni program *debagovanje.vi* sačuvao u okviru projektne datoteke izabrati opciju ***File»Create Project...»Blank Project»Finish*** i potvrditi da se trenutno otvoreni *debagovanje.vi* doda („Add“) u kreirani projekat. Pojaviće se projektna datoteka kao na Sl. 1.2.12.



Sl. 1.2.12. Projektna datoteka

13. Sačuvati projektu datoteku kao *debagovanje.lvproj*.

NAPOMENA: U projektu datoteku je moguće dodati fajlove, hiperlinkove ili kreirati direktorijume izborom opcije ***MyComputer»Add***, Sl. 1.2.13. Direktorijumi u okviru projekta mogu biti virtuelni (*Snapshot*) ili stvarni (*Auto-populating*), tj. mogu da realno ne postoje ili da realno postoje na hard disku, respektivno.



Sl. 1.2.13. Dodavanje elemenata u projektnu datoteku