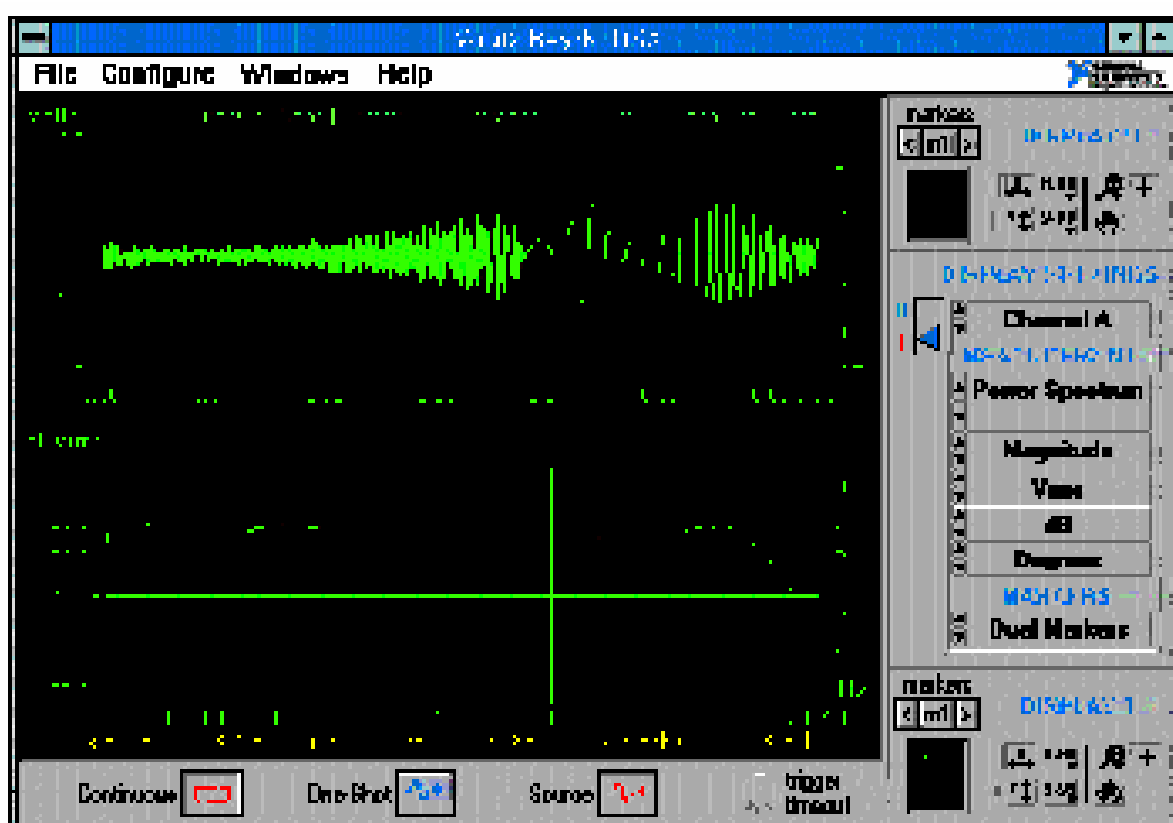


ELEKTROTEHNIČKI FAKULTET SARAJEVO

ADNAN SALIHBEGOVIĆ

SPECIJALNA MJERENJA



SEPTEMBAR 2003

SADRŽAJ

Poglavlje	Naziv	Strana
1	Virtualna instrumentacija -VI	2
2	Uvod u LabVIEW	10
3	Kreiranje subVI-eva	26
4	Konture i chartovi	40
5	Polja, klasteri i grafovi	67
6	Slučaj, strukture sekvence i čvor formule	77
7	Stringovi i File I/O	91
8	Prikupljanje podataka i kontrola instrumenta	102
9	Razvoj LabVIEW instrument drajvera	112
10	Pregled LabVIEW instrument drajvera	155
11	Primjeri virtualnih instrumenata	168
12	Start upravljanja instrumentom sa interaktivnom rutinom i instrument Wizardom	187
13	Primjeri korištenja LabVIEW-a I DAQ modula u sintezi virtualnih instrumenata	194
14	Povezivanje sa signalima iz mjernog okružaja	226
15	Karakteristike DAQ modula	257
16	Korištenje brze Fourierove transformacije (FFT) sa HP 54600 osciloskopima	316
17	FFT laboratorijski eksperimenti sa HP 54600 osciloskopima	334
18	Prikupljanje i obrada podataka korištenjem IOTech modula DAQBoard 200/A	359

POGLAVLJE 1

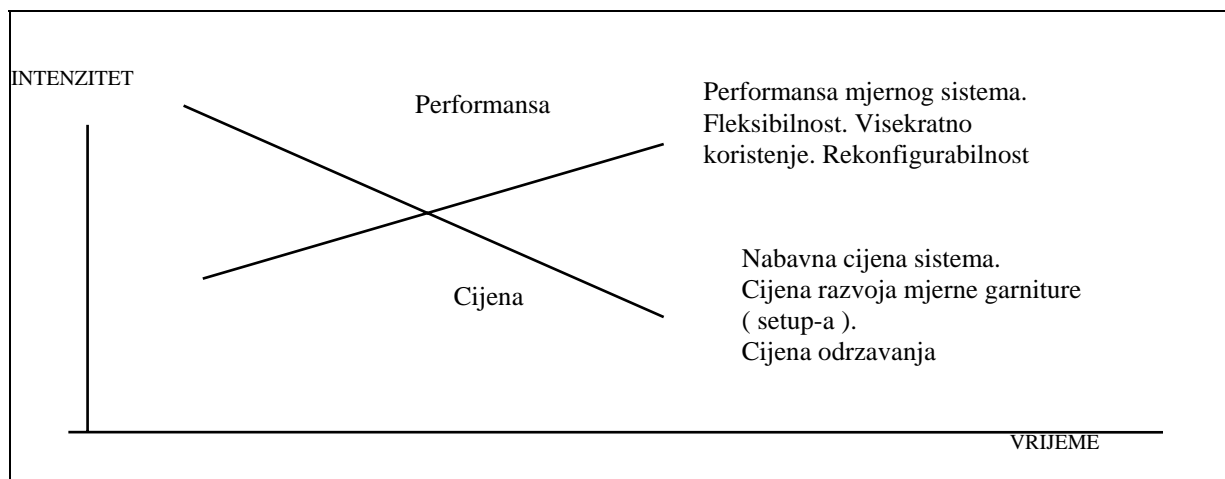
VIRTUALNA INSTRUMENTACIJA - VI

U 1986 američka kompanija National Instruments je uvela LabVIEW 1.0 sa ciljem da obezbjedi software-ski alat koji bi naoružao inženjere da razvijaju specifične za aplikaciju (customized) sisteme, na isti način na koji je spreadsheet omogućio svima onima koji su u businessu da analiziraju finansijske podatke.

NI je na neki način pionir revolucije u virtualnoj instrumentaciji, revolucije koja mjenja način korištenja instrumentacije i kod testiranja, mjerenja i automatizacije procesa , smanjujući značajno troškove a bez zrtvovanja tačnosti i kvaliteta mjerenja.

Uvodeći koncept **konfigurabilne virtualne instrumentacije**, podesivi hardware i software nudi jedan ogroman nivo različitih primjena u mjerenju i testiranju sa istim setom hardware-a , koji je najčešće PC baziran i standardizirane arhitekture.

Slijedeći diagram grafički zorno ilustrira prednosti ovakve korisnički definisane virtualne instrumentacije :



Tradicionalni mjerni instrument (multimetar, osciloskop, generator signala , analizator frekventnog spektra , itd.) je autonomni uređaj sa svojim mogućnostima ulaza i izlaza signala koji se na njega priključuju i fiksnim karakteristikama kako se uređaj opslužuje i koristi od strane korisnika : tj. fiksnim tkz. user interface-om , tj rasporedom i oblikom tastera , prekidača , izbornika i preklopnika opsega , buksni (klima), za priključak signala i ostalog.

Unutar uređaja postoje specijalni elektronski sklopovi i moduli, uključujući A/D i D/A konvertore, kola za kondicioniranje signala, mikroprocesore, memorijske blokove, interne bas konvertore realnih signala koji se priključuju na mjerni uređaj, analiziraju ih i prikazuju ih korisniku na raznim oblicima elemenata za prikazivanje signala kao sto su: analogne skale, led diode, displeji, bar graf displej diode itd.

Proizvođač mjernog instrumenta definise sve funkcionalnosti instrumenta i korisnik ih ne moze mjenjati.

Virtualna instrumentacija (VI) se bazira na otvorenoj arhitekturi standardnih tipova industrijskih PC računara da bi obezbjedila i ponudila korisniku procesiranje signala, memoriju za njihovo pohranjivanje i prikazivanje na ekranu na uređajima koji su danas univerzalno korišteni i prisutni.

Jeftini DAQ (data acquisition - prikupljanje podataka) moduli te GPIB (general purpose instrumentation bus - instrumentalni bus opšte namjene) te VXI (prosirenje - ekstenzija Motorolinog standardnog VME basa za izgradnju modularnih bus baziranih mikroprocesorskih sistema otvorene arhitekture), interfejsni moduli koji se plaguju u otvorene standardizirane basove, omogućuju ovim PC baziranim mjernim instrumentima kontakt sa vanjskim signalima tj. takozvane **front end** mogućnosti.

Zbog otvorene arhitekture PC-eva i radnih stanica (workstation , uobičajeni naziv za PC uključen u lokalnu mrežu računara), funkcionalnost virtualne instrumentacije definise korisnik pa prema tome ona postaje i podesiva i rekonfigurabilna.

PREDNOSTI KORISNIČKI DEFINISANE VIRTUALNE INSTRUMENTACIJE

Fundamentalni koncepti virtualne instrumentacije direktno se prevode u bazične prednosti za korisnika.

To je korisnik, a ne proizvođač instrumenta, koji definise funkcionalnost mjernog instrumenta.

Dakle, pristupačna, fleksibilna mjerna instrumentacija se pojavljuje kao rezultat ogromnih ulaganja u tehnoloske pomake u industriji računara, i mi kao korisnici mjernih uređaja za različite namjene u istraživanju i industriji od toga ogromno profitiramo.

Software je ključna supstanca u onome sto se zove virtualna instrumentacija.

Aplikacioni software naoruzava korisnika sa alatima neophodnim da gradi virtualne instrumente i prosirojuje njihovu funkcionalnost obezbjeđujući njihovu povezivost sa ogromnim mogućnostima PC-eva, radnih stanica i njihovog bogatog izbora aplikacija. Ne samo da su nabavne cijene virtualnih instrumenata niske zbog ogromne konkurencije na trzistu PC računara koja dramatično snizava njihove cijene, nego su takodjer troskovi razvoja i održavanja značajno smanjeni korištenjem :

- softwareskih i hardwareskih platformi sa otvorenom arhitekturom
- alata raspolozivih u svakom trenutku (off-the- shelf, tj. sa police)
- visekratno iskoristivih programskih modula

Software sa bibliotekama koje su projektovane za mjernu instrumentaciju, kao sto su biblioteke drivera (softwareskih modula pod standardnim operativnim sistemom koji povezuju operativni sistem i software sa hardware-om mjernog instrumenta) za konkretne tipove mjernih instrumenata, minimiziraju vrijeme razvoja virtualnog instrumenta.

Programski kod koji je intuitivan i automatski generise svoju dokumentaciju koja objasnjava kako i sta radi software, sa sirokim spektrom raspolozivih korisnih

programa (utilities) za evaluaciju i trazenje i korekciju gresaka (tj. troubleshooting) , takodjer osigurava niske troskove odrzavanja ovakve virtualne instrumentacije.

TEHNOLOSKI USPJESI U DIGITALIZACIJI

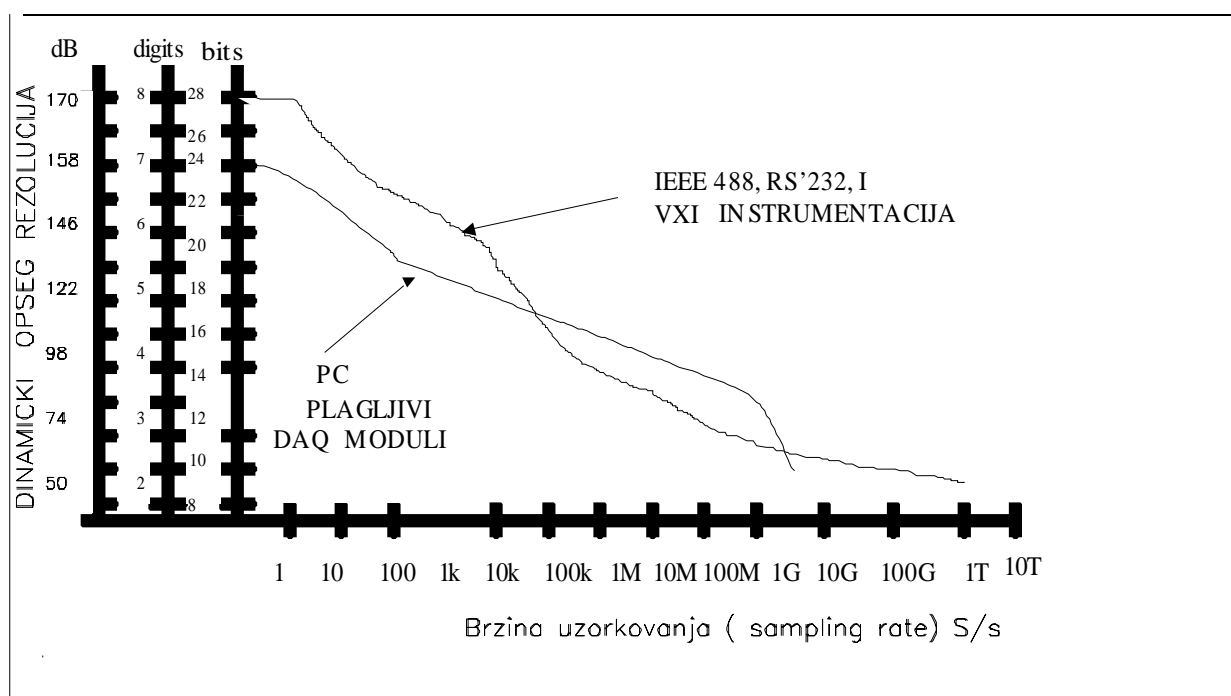
Mogućnosti digitalizatora (elektronskih sklopova koji vrse uzorkovanje i pretvaranje u digitalnu formu analognih signala), naročito kod uplagljivih (plug-in) DAQ modula dramatično rastu i prosiruju se.

Uplagljivi DAQ moduli imaju jasne mogućnosti širokog spektra klasičnih mjernih instrumenata i mogu se koristiti za izgradnju vrlo ekonomičnih i sofisticiranih virtualnih instrumenata.

GPIB (ili ponegdje jos koristen termin HPIB prema Hewlett Packard-u koji ga je prvi definisao i uveo) i VXI instrumenti nastavljaju i dalje da predvode savremenu tehnologiju mjerne instrumentacije.

Virtualni instrumenti kombinuju uplagljive DAQ module sa GPIB, VXI , serijskim i industrijskim interfejsima i komunikacionim protokolima u jedinstvenu, konzistentnu programsku paradigmu (tj. sintagmu odnosno izraz koji to znači - virtualni instrument), i to od početka do kraja, tako da korisnik moze dizajnirati mjerni uređaj koji je optimalan i po cijeni koštanja i po njegovoj performansi.

Funkcionalnost sistema je odredjena sa vrstama programskih obrada koje se provode nad uzorkovanim signalima , a ne kao prije sa funkcionalnošću instrumenta sa kojim se vrši mjerenje.



Slijedeća slika ilustrira granice danasnjeg stanja tehnologije digitalizatora koji na direktan način opredjeljuju i oblast primjene virtualne instrumentacije u odnosu na klasične mjerne instrumente fiksne funkcionalnosti i namjene.

OPCIJE KOD VIRTUALNE INSTRUMENTACIJE

Danas, zavisno od proizvođača, postoji širok izbor software-skih paketa namjenjenih realizaciji virtualne instrumentacije.

NI kao pionir u oblasti danas ima dva softwareska paketa za ove funkcije.

Prvi je LabVIEW koji se sastoji od kompajliranih (prevedenih iz visoko nivovskog grafičkog programskog jezika G u masinske, izvrsne module) modula.

Za programere koji koriste programski jezik C i C++ , **NI** je razvio programski paket koji je nazvao **LabWindows/CVI**, interaktivni C programski okruzaj koji nudi performansu kompajlera sa lakoćom koju ima programski jezik Basic kao interpreter. Nadalje, **NI** nudi programski paket **Measure** za rad sa Microsoft Excel okruzajem, kao i **Component Works** za Microsoft Visual Basic programski okruzaj.

Takodjer, **NI** nudi i gotove programske pakete virtualne instrumentacije za specificiranu funkcionalnost, koja se moze odmah početi koristiti, samo je potreban, nakon instalisanja software-a, pritisak na mouse taster.

KRATAK PREGLED STANDARDA U OBLASTI VI

IEEE 488

U 1965, Hewlett-Packard je dizajnirao tkz. HP-IB (Hewlett-Packard Interface bus) da bi povezao njegovu familiju programabilne instrumentacije sa njihovim specifičnim kontrolerima. Zbog velike brzine prenosa podataka na ovome basu, ovaj interfejs je vrlo brzo postao popularan i prihvaćen i od drugih proizvođača. Kasnije je prihvaćen i kao IEEE standard 488-1975, da bi kasnije bio doradjivan i pojavio se kao ANSI/IEEE 488.1 - 1987.

Dalja poboljšanja ovoga standarda su rezultirala u objavi dva nova prijedloga koja su se pojavila. Prvi je bio ANSI/IEEE 488.2 - 1987 koji je poboljšao prethodni standard definisuci preciznije kako kontroleri i instrumenti komuniciraju preko ovoga basa.

U 1990, standardne komande za programabilne instrumente (Standard Commands for Programmable Instruments - SCPI) definisali su strukturu u standardu IEEE 488.2 i uvele jedinstveni skup komandi koji se koristi sa svakim SCPI instrumentom.

NI je prosirio korištenje IEEE 488 basa na računare koje su proizvodile i druge firme osim Hewlett-Packard-a. Zato se i pojavio naziv **GPIB (General purpose interface bus)** koji se danas česće koristi nego HP-IB.

Standardizirajući hardware za mjernu instrumentaciju na ovome basu, **NI** je razvio i čitavu seriju driverskog software-a za različite tipove mjernih instrumenata koji se preko ovoga basa povezuju sa PC-em. Ova familija drivera dobila je oznaku NI-488 i postala je de facto industriski standard.

VXI bus

VXI bas je vrlo brzo razvijajuća platforma za mjernu instrumentaciju. Prvi put je uveden u 1987, od kada je dozivio ogroman razvoj i ekspanziju kao i prihvatanje od velikog broja proizvođača tako da je definisan i kao IEEE standard 1155 - 1994 godine.

Danas vise od 300 proizvođača proizvode vise od 1200 komercijalnih VXI proizvoda.

VXI se koristi u širokom spektru tradicionalnih mjernih i test uradjaja ali takodjer i kod takozvanih ATE (automatic testing equipment) aplikacija.

On takodjer dozivljava ogroman razvoj kao platforma za prikupljanje i analizu signala kod istrazivanja ali i u upravljanju u okviru industrijskih aplikacija.

Mada se jos uvijek pojavljuju čisti VXI sistemi u dijelu akvizicije i interfejsa sa procesom, sve česće se mogu sresti aplikacije kod kojih se VXI integrisse zajedno sa tradicionalnim GPIB instrumentima i/ili plagljivim DAQ modulima.

VXI koristi tkz. mainframe sasiju sa maksimalno do 13 slotova (mjesta za plagovanje kartica) da se moze modularno graditi zeljeni broj ulazno/izlaznih kanala veze sa procesom.

Kod proizvođača postoji veliki broj ovih instrumenata i veličina mainframe (sasije). Posto je VXI baziran na svjetski rasprostranjenom i prihvaćenom VME basu, moguće je koristiti VME module u VXI sasijama i sistemima.

VXI bas (backplane) uključuje 32 -bitni VME kompjuterski bas kao i instrumentacioni bas visokih performansi za precizni tajming i sinhronizaciju izmedju različitih komponenti mjernog instrumenta.

PLAGLJIVI MODULI ZA PRIKUPLJANJE PODATAKA

Plagljivi DAQ moduli, koji su najbrže rastući segment instrumentacije, predstavljaju srce virtualne instrumentacije. Plagljivi DAQ moduli su raspoloživi za mnoge tipove PC računara koji koriste standardne baseve kao sto su:

PCI, EISA, ISA, PC card (ranije PCMCIA) i Macintosh NuBus.

Ovi moduli (plaglive kartice) se pojavljuju u raznim kombinacijama analognih, digitalnih i tajming ulaza i izlaza.

Mnogi moduli imaju programabilne mogućnosti rada po kanalu glede brzine uzorkovanja (samplovanja) i konverzije, nezavisno pojačanje za svaki kanal, kao i različite opcije trigerovanja kao sto su: pre, post, usporeno (delayed), analogno i digitalno trigerovanje, čineći ove module vrlo fleksibilnim za najrazličitije primjene.

Kondicioniranje signala na ulaznom modulu (front-end) moze biti koristeno da smanji troskove uvođenja u sistem različitih tipova i nivoa signala koji se sreću u praksi.

Tako postoji široka lepeza modula za kondicioniranje signala za termoelemente, RTD (mjerjenje temperature promjenom otpora), naponske i strujne ulaze, visokonivovske signale, kao i digitalne ulazne i izlazne signale većeg energetskog nivoa.

Da bi se kontrolisali DAQ hardwareški moduli sa software-skim drajverima (drivers) , mogu se programirati ovi drajveri za različite Operativne sisteme kao sto su Windows NT, WIN 95, 98 , Windows2000 ili pak Mac OS, i DOS .

Softwareski paketi kao LabVIEW, LabWindows/CVI , Measure, ComponentWorks, mogu pojednostaviti razvoj kompletnih data akvizicionih sistema koristenih u različitim računarski baziranim sistemima uključujući i virtualnu instrumentaciju.

INDUSTRIJSKE KOMUNIKACIJE

Pod pojmom industrijske komunikacije podrazumjeva se široka lepeza hardwareških i software-ških proizvoda i protokola koji se koriste sa uređajima koristenim u primjenama automatizacija u industriji.

Mnogi uređaji koji se koriste u automatizaciji u industriji koriste RS-485 /422 ili RS-232 serijski interfejs za komunikacije.

Elementi kao što su: da li se koristi puna (duplex) ili polu-duplex komunikacija kao i koji softwareski protokol, određuju hardwaresko i softwaresko rješenje koje će se koristiti.

Većina proizvođača koriste za serijsku komunikaciju specifični ASCII komandni set koji na hardwareskom nivou koristi standardne serijske portove koji se mogu naći na svim PC-evima (COM1 , COM2, COM3, COM4).

Uređaji koji su raspoloživi sa ovako jednostavnim protokolima uključuju:

data loggere (prikupljanje i arhiviranje podataka iz procesa), digitalno bazirane regulatore, module za složeniju obradu signala (napr. kalorimetri) i slično.

Standardizacija protokola na ovom nivou nije mnogo odmakla izuzev možda MODBUS protokola kojeg koriste mnogi proizvođači PLC uređaja (programmable logic controllers- programabilni logički kontroleri).

Medjutim, danas se koriste i mnogo sofisticiranije industrijske mreže i protokoli za prenos podataka između pojedinih djelova distribuiranih sistema upravljanja. Dosta takvih protokola su specifični za proizvođače industrijskih računarski baziranih sistema (napr. Honeywell - Hiway , ABB Masterbus, Allen Bradley Datahiway (DH) + , itd.), mada se može uočiti i standardizacija oko jednog mreznog protokola kao što je to Ethernet i na nižem nivou Foundation fieldbus (FF) I Siemens podržavani Profibus.

DRIVERSKI SOFTWARE

NI ima široku lepezu software-skih driverskih modula za različite hardwareske module kao:

NI-DAQ za DAQ i SCXI module, NI-488.2 za GPIB hardware, NI-VXI/VISA za VXI hardware, NI-DSP za hardware na bazi DSP (digital signal processor) hardwarea, te NI-FBUS; koji su portabilni (prenosivi dakle upotrebljivi) na više platformi različitih Operativnih sistema.

Svaki od ovih "device drivera" (softwareskih modula za upravljanje hardwareskim modulom za akviziciju ili obradu) je dizajniran da maksimizira programsku fleksibilnost i propusnost za podatke.

Nadalje, svaki od ovih drivera je dizajniran za zajednički aplikacioni programski interfejs (API - application programming interface), tako da bez obzira na Operativni sistem , aplikacioni programi kreirani na bazi ovih drivera su portabilni.

Ovi driverski softwareski moduli daju korisniku mogućnost da koriste jedanput razvijene programske module u jezicima kao što su C, C++, Basic, i de facto programskim okruženjima kao što su LabVIEW i LabWindows/CVI na kompjuterima sa različitim Operativnim sistemima.

Za uređaje koji su bazirani na registrima za prenos podataka između modula i PC računara kao što su plugljivi DAQ moduli, driverski softwareski moduli otklanjaju svu onu kompleksnost koja se sreće kod programiranja na nivou registara .

SOFTWARE ZA OBRADU I VIZUELIZACIJU

Kod virtualne instrumentacije se ostvaruje tkz. in-line analiza podataka koji se prikupljaju preko ulaznih modula.

Korisniku koji konfigurise neku funkciju virtualnog instrumenta stoje na raspolaganju funkcionalni moduli koji mu omogućuju da filtrira signal da bi eventualno odstranio superponirane smetnje, koriguje podatke deformisane eventualno neispravnim uredjajem za primarno prikupljanje i obradu, ili da kompenzira za uticaje okruzenja kao sto su temperatura i vlaznost.

Korisnik moze dobiti dodatne korisne podatke o signalu kojeg mjeri i analizira koristeći analizatorske funkcije da bi uklopio (fitting) podatke u neki analitički oblik krive, transformise podatke u frekventni domen ili izvrši statističku obradu nad podacima.

Softwareski paketi kao sto su LabVIEW i LabWindows/CVI kao i ostali pomenuti, uključuju biblioteke za izvršenje svih ovih pobrojanih analitičkih tehnika ali i mnogo vise od toga.

Funkcije uključuju tehnike za procesiranje signala, uklapanje krivih (curve fitting), numeričku analizu, kao i statističku analizu.

LabVIEW i LabWindows/CVI su kompletna softwareska razvojna okruzenja sa jednako moćnim alatima za akviziciju, in-line analizu, kao i prezentaciju rezultata obrada i analiza.

Prezentaciona grafika u paketima LabVIEW i LabWindows/CVI je projektovana za real-time performanse, tako da korisnik moze posmatrati i analizirati podatke u istom trenutku kada ide i njihovo prikupljanje.

Sa korisnički (custom) konfigurisanim grafovima i prezentacionim prozorima i prikazima, korisnik moze prikazati podatke u vise različitih formata da bi mogao da ih prati i analizira u obliku koji mu daje najvise informacija.

LabVIEW

Nakon sto je LabVIEW postao raspoloziv u 1986 godini, grafičko programiranje (i objektno orjetirano programiranje) je izraslo od alternativne programske metode u de facto industrijski standard.

Sa LabVIEW korisnik moze brzo dizajnirati svoj specifični korisnički interfejs i grafički asamblirati svoj program kao blok dijagram.

Razvojna paradigma (inačica) je idealna za one koji realizuju razvoj mjerne instrumentacije za bilo koju vrstu mjerenja.

Ove (1997) NI je uveo **LabVIEW Professional Development Suite** (kompletni razvojni paket) koji se sastoji od kolekcije novih razvojnih alata, programskih uputa kao i standardnih dokumenata dizajniranih za profesionalne programske timove koji grade velike aplikacije sa LabVIEW.

Sa LabVIEW korisnik moze asamblirati eksperimente koji imaju intuitivni grafički korisnički interfejs, prikuplja signale sa plug-in modulima za prikupljanje podataka, i kontrolise GPIB, VXI i serijski povezanu instrumentaciju.

LabVIEW biblioteka drivera za instrumente olaksava programske napore sa visoko-nivovskim intuitivnim funkcijama, umjesto nisko-nivovskih komandi instrumentu.

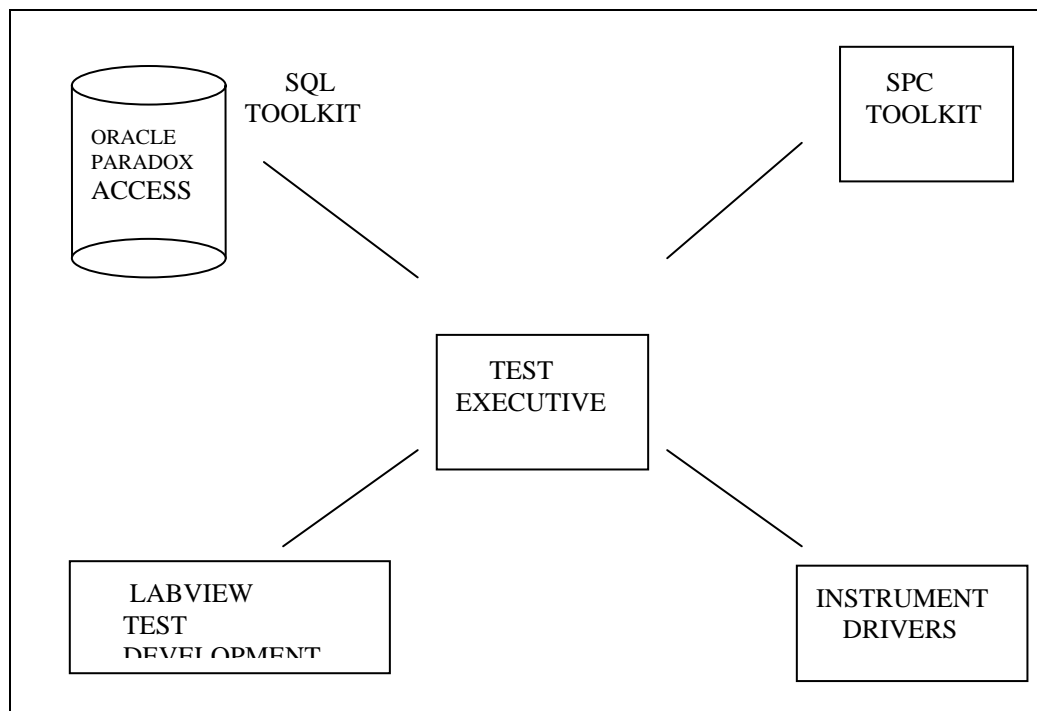
U ovoj biblioteci NI nudi više od 600 drivera za GPIB, VXI i serijsku instrumentaciju, pokrivajući sve važnije proizvođače mjerne instrumentacije danas u svijetu.

Sa **Test Executive** programskim dijelom, korisnik može da dizajnira interaktivno sve izvrsne test sekvence i na taj način organizuje izvođenje pojedinih dijelova programskih cjelina u okviru cjelokupnog projekta mjerenja i analize rezultata.

SQL toolkit - je direktna veza Labview-a sa više od 30 lokalnih ili udaljenih baza podataka. Sa database bibliotekom u SQL (standard query language - standardni jezik za pretraživanje baza podataka), korisnik može povezati svoj mjerni ili test program da bi pohranio podatke u neku od baza ili pak download-ovao (napunio) iz baze test i konfiguracione parametre.

SPC (statistical process control) toolkit - sa ovim alatom korisnik može da analizira svoje rezultate dobijene mjerenjem, sa ciljem da provede statističku analizu rezultata. Ove rezultate onda može koristiti da poboljša upravljanje procesom da bi podigao njegov kvalitet.

Slijedeća slika pokazuje strukturu LabVIEW paketa i odnose između pojedinih dijelova.



POGLAVLJE 2

UVOD U LABVIEW

LABVIEW

Labview je programska razvojna aplikacija, slicna mnogim razvojnim sistemima na bazi C ili Basic-a ili pak NI LabWindows/CVI razvojnom sistemu. Ipak, LabVIEW se razlikuje od pobrojanih aplikacija u jednom vaznom pogledu. Drugi programski sistemi (izuzev VB -Visual Basic-a) koriste tekst bazirane jezike da generisu linije koda, dok LabVIEW koristi graficki programski jezik **G**, da kreira programe u obliku blok dijagrama.

Korisnik moze koristiti LabVIEW sa malo programerskog iskustva. LabVIEW koristi terminologiju, ikone, i ideje bliske naučnicima i inženjerima i oslanja se na graficke simbole radije nego na tekstualni jezik da bi opisao programske akcije.

LabVIEW sadrzi obimne biblioteke funkcija i subrutina za vecinu programskih zahtjeva koji se mogu pojaviti. Za operative sisteme kao sto su Windows, Macintosh i Sun, LabVIEW sadrzi aplikaciono specificne biblioteke za akviziciju podataka i VXI kontrolu instrumenata. LabVIEW takodjer sadrzi aplikaciono specificne biblioteke za GPIB i serijsku vezu kontrole instrumenta, analizu podataka, predstavljanje podataka, i njihovo pohranjivanje. LabVIEW ukljucuje konvencionalne programske razvojne alate, tako da korisnik koji razvija aplikaciju moze postaviti breakpointe (**tačke loma**, tj. mjesta u programu gdje ce se njegovo izvršenje zaustaviti i cekati na akciju korisnika), animacija izvršenja programa da bi se vidjelo kako podaci prolaze kroz program, i korak po korak izvršenje programa (single step), sa ciljem da bi se olaksao debugging (trazenje i otklanjanje programskih gresaka) kao i razvoj programa.

Kako LabVIEW radi

LabVIEW ukljucuje biblioteke funkcija i razvojnih alata specificno dizajniranih za upravljanje instrumentima. LabVIEW programi se nazivaju **virtualni instrumenti** (VI-evi), jer njihov izgled i rad imitira stvarne instrumente. Ipak, ovi moduli su sa funkcijama konvencionalnih programskih jezika. VI-evi imaju i interaktivni korisnicki interfejs i ekvivalent izvornom kodu, i prihvataju parametre od VI-eva na visem nivou.

Slijede opisi ovih triju karakteristika VI-eva:

- ◆ VI-evi sadrže jedan interaktivni korisnicki interfejs, koji se naziva prednjim panelom (**front panel**), posto on simulira panel fizičkog instrumenta. Prednji panel moze sadrzavati dugmad, tastere, grafove, i druge kontrolne i indikacione elemente. Korisnik unosi podatke koristeći tastaturu i mis (mouse) a zatim posmatra rezultate na ekranu.
- ◆ VI-evi primaju instrukcije iz **blok dijagrama**, koji korisnik gradi u **G** jeziku. Blok dijagram obezbjedjuje graficko rjesenje za programski problem. Blok dijagram sadrzi izvorni kod za VI.

- ◆ VI-evi koriste hijerarhijsku i modularnu strukturu. Mogu se koristiti kao top-level programi (najgornjeg nivoa), ili kao podprogrami unutar drugih programa ili podprograma. VI unutar druge VI se zove subVI (pod VI). **Ikona i konektorski panel VI** djeluju kao graficka parametarska lista tako da preko nje drugi VI-evi mogu prenjeti podatke u nju kao subVI.

Sa ovim karakteristikama, LabVIEW promovira i priklanja se konceptu modularnog programiranja. Dizajner dijeli aplikaciju u niz taskova (zadataka) koje on moze dalje dijeliti sve dok komplicirana aplikacija ne postane niz jednostavnih podtaskova (podzadataka). Dizajner gradi VI da bi izvršio svaki podzadatak a nakon toga kombinira ove Vi-eve na drugom blok dijagramu da bi ostvario veci zadatak. Konačno, VI na najvisem nivou (top level) sadrzi skup subVI-eva koji predstavljaju aplikacione funkcije.











Posto se svaka subVI moze izvršiti sama za sebe, nezavisno od ostatka aplikacije, debugiranje (trazenje gresaka u programu) je mnogo lakse.

Nadalje, mnogi subVI niskog nivoa često izvršavaju zadatak koji je zajednicki za vise aplikacija, tako da se moze razviti specijalizirani skup subVI-eva pogodan za aplikacije koje dizajner namjerava razviti.

Paleta alata

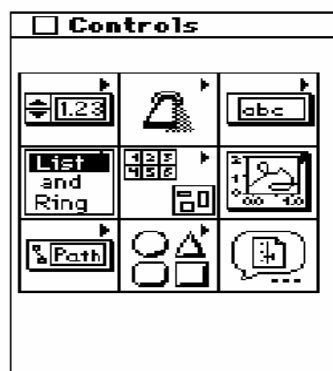
LabVIEW koristi plutajucu (floating) paletu alata (**Tools**), koji se koriste za editiranje i debugiranje VI-eva. Koristite <Tab> taster da bi se prelazilo sa alata na alat na paleti. Ako je Tools paleta zatvorena, izabrati **Windows >> Show Tools Palette** da bi se prikazala paleta na ekranu. Ova paleta je prikazana na slijedejoj slici:



	- Operating tool	Postavlja Controls i Functions paletu na prednji panel i blok dijagram.
	- Positioning tool	Pozicionira, mjenja velicinu i vrši selekciju objekata
	- Labeling tool	Editira tekst i kreira slobodne labele
	- Wiring tool	Ožičava objekte medjusobno u blok dijagramu
	- Object pop-up menu tool	Poziva pop-up (iskačuci) meni za objekat
	- Scroll tool	Skrolira(klizi)kroz prozor(window) bez koristenja scrollbarova (bočnih indikatora u prozoru)
	- Breakpoint tool	Postavlja breakpointe na VI-eve , funkcije, loops(konture),sekvence i cases (programske module)
	- Probe tool	Kreira test tačke na zicama .
	Color copy tool	Kopira boje za pasting(kopiranje)
	Color tool	Postavlja boje u prednjem (foreground) planu i pozadini

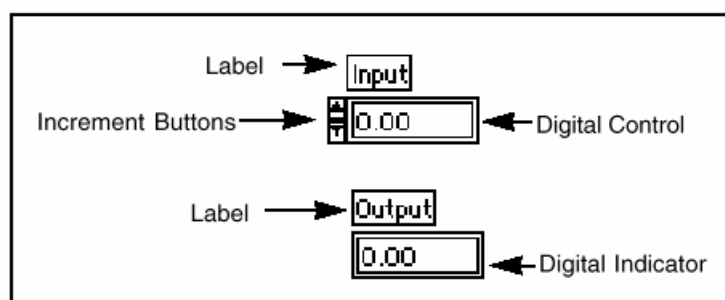
Kontrolna paleta

Kontrolna (**Controls**) paleta se sastoji od graficke , plutajuće palete i automatski se otvara kada se lansira LabVIEW. Dizajner koristi ovu paletu da bi postavio kontrolne i indikacione elemente na prednji panel VI. Svaka ikona na najvisem nivou (top-level) sadrzi subpalette. Ako **Controls** paleta nije vidljiva, moze se otvoriti selekcijom **Windows>>Show Cotrols Pallette** sa menija prednjeg panela. Moze se takodjer izbaciti (pop-up) na neki otvorenu površinu prednjeg panela da bi se pristupilo privremenoj kopiji **Controls** palete. Slijedeca slika prikazuje top nivo **Controls** palete :



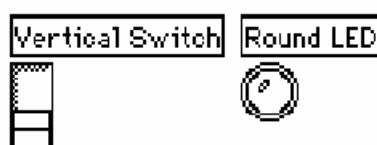
Kontrolni i indikacioni elementi

Dizajner može koristiti numeričke kontrolne elemente da unese numeričke iznose, a numeričke indikatore koristi za display numeričkih iznosa. Dva najčešće korištena numerička objekta su : *digital control* (digitalni kontrolni element) i *digital indicator* .




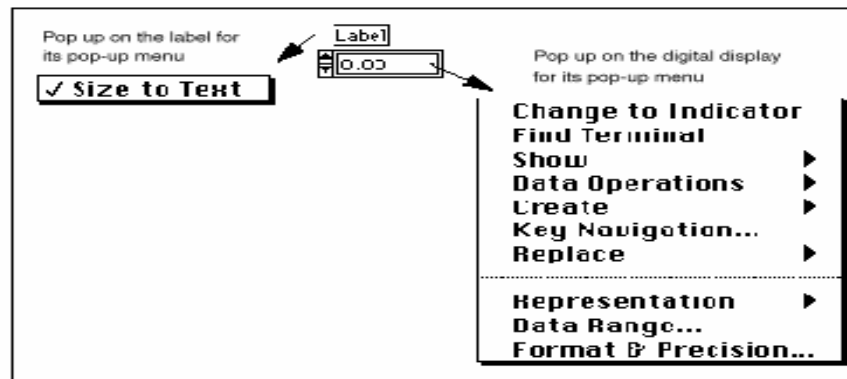
Bulovi (Boolean) kontrolni i indikacioni elementi

Dizajner ih koristi za unosenje i prikazivanje (display) Bulovih (true/false) vrijednosti (logičke vrijednosti tačno/netačno). Bulovi objekti simuliraju prekidače, tastere i LED diode (indikacione diode). Najčešće korišteni Bulovi objekti su vertikalni prekidač (*vertical switch*) i okrugla LED dioda .



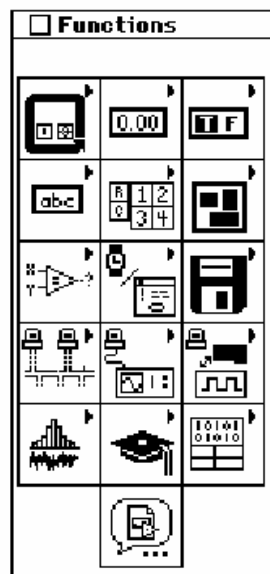
Konfigurisanje kontrolnih i idikacionih elemenata

 Dizajner može konfigurirati skoro sve kontrolne i indikacione elemente koristeći opcije iz njihovih pop-up menija. Iskakanje (popping-up) se ostvaruje klikom miša na individualne komponente kontrolnih i indikacionih display menija za konfigurisanje ovih komponenata. Jedan lak način da se pristupi pop-up meniju je da se klikne sa Object pop-up menu tool , (koji je pokazan na lijevoj strani ovog paragrafa), na bilo koji objekat koji ima pop-up menu. Slijedeca slika ilustrira ovaj metod displaya za *digital control*.



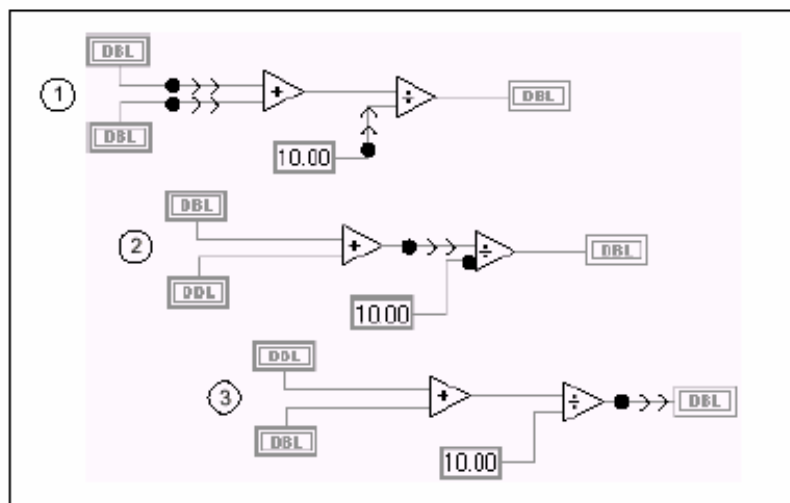
Funkcionalna paleta

Functions paleta se sastoji od grafičke, plutajuće (floating) palete koja se automatski otvara kada se predje u blok dijagram *mode rada*. Dizajner je koristi da bi postavio čvorove (nodes) (kao što su konstante, indikatori, VI-evi, itd.) na blok dijagram VI. Svaka top-level ikona sadrži subpalette. Ako **Functions** paleta nije vidljiva, može se pozvati birajući **Windows»Show Functions Palette** iz blok dijagram menija. Može se također pozvati pop-up načinom u bilo koju slobodnu površinu blok dijagrama da bi se pristupilo privremenoj kopiji **Functions** palete.



TOK PODATAKA

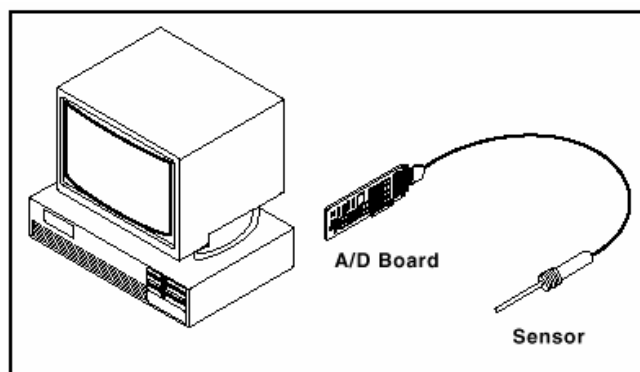
Labview u izvršenju VI-eva slijedi model toka podataka (dataflow) za izvršenje programa. Blok dijagram se sastoji od čvorova (nodes) kao što su VI-evi, strukture, i terminali prednjeg panela. Ovi čvorovi su povezani sa *zicama* (wires), koje definiraju tok podataka kroz program. Izvršenje čvora se ostvaruje kada su njegovi ulazi raspoloživi. Kada čvor završi svoje izvršenje, on stavlja na raspolaganje sve svoje izračunate izlaze slijedecem cvoru u redoslijedu toka podataka, ako što se vidi na slijedećoj slici.



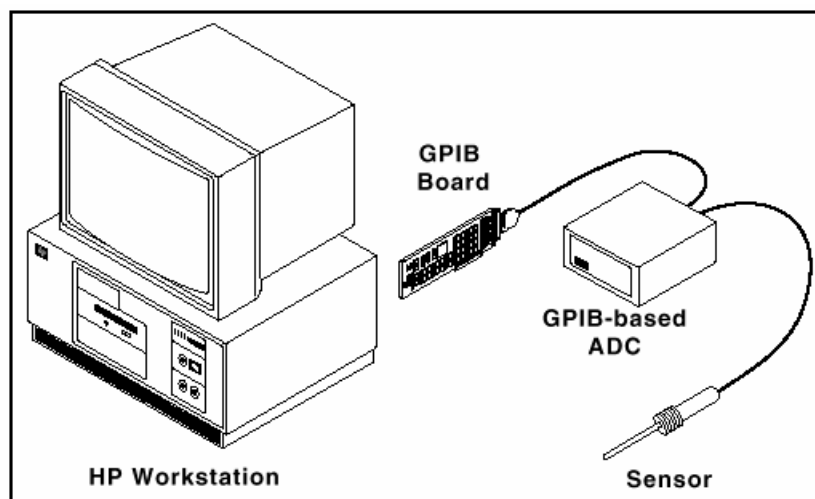
GRADNJA VI

CILJ

Izgraditi VI koja simulira prikupljanje očitavanja temperature. Mi ćemo koristiti **Demo Voltage Read VI** (VI za očitavanje napona) da bi mjerili napon, a zatim ga pomnožiti sa 100.0 da bi konvertovali napon u temperaturu. (u stepenima F). Zamislamo da imamo senzor (termoelement) ili transmiter temperature koji pretvara temperaturu u napon. Mi ćemo povezati senzor na analogno/digitalni pretvarač (ADC), kako je to pokazano na slijedećoj slici. Ovaj ADC pretvarač će pretvoriti napon u digitalni podatak.



Senzor bi također mogao biti spojen na ADC koji je povezan sa PC računarom preko GPIB, kako je to prikazano na narednoj ilustraciji. Ovdje on također radi pretvaranje napona u digitalni podatak.



Prednji panel

1. Otvoriti novi prednji panel selekcionisuci **File>>New** ili birajuci **New VI** taster u dijalog boxu.
2. Ako **Controls** paleta nije vidljiva, izabrati **Windows>> Show Controls Palette** da bi se pojavila paleta. Moguce je takodjer pristupiti Controls paleti klikčuci misom u otvorenom polju. Da bi se izbacila (pop-up) ova paleta treba kliknuti na desni taster misa.



3. Izabrati termometarski indikator iz **Controls>> Numeric**, i staviti ga na prednji panel vukuci indikator na panel.



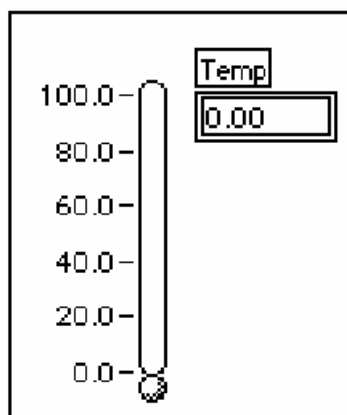
4. Otkucati Temp unutar boxa za tekst labele i zatim kliknuti na Enter taster na toolbaru.

Napomena: Ako se klikne van tekst boksa bez unosenja teksta, labela isčezava. Labela se ponovo može pokazati klikčuci na control paletu i birajuci Show >> Label.



5. Reskalirati termometarski kontrolni element da pokaze temperaturu izmedju 0.0 i 100.

- a) Koristeci alat za labeliranje (označavanje), dvaput uzastopno kliknuti na 10.0 na termometarskoj skali da se naglasi (highlight).
- b) Ukucati 100.0 u skalu i kliknuti mis bilo gdje van display prozora. LabVIEW automatski skalira medju-inkremente. Kontrolni element temperature treba sada da izgleda otprilike kao na slijedejoj slici:



BLOK DIJAGRAM

1. Otvoriti blok dijagram birajući **Windows>> Show Diagram**. Selektirati blok dijagram objekte koji se razmatraju u daljnjem tekstu detaljnije, iz **Functions** palete. Za svaki objekat koji zelite da umetnete, izabrati ikonu a zatim objekat iz top-nivoa palete, ili izabrati objekat iz odgovarajuće subpalete. Kada se pozicionira mis na blok dijagram, LabVIEW će prikazati okvir (outline) objekta.

Ako **Functions** paleta nije vidljiva, izabrati **Windows>> Show Functions Palette** da bi se prikazala paleta. Ovoj paleti se također može pristupiti poping-up (klikćuci) u otvorenoj oblasti blok dijagrama.

Postaviti svaki od slijedecih objekata na blok dijagram.



VI pod nazivom **Demo voltage read** (čitaj napon iz **Functions>>Tutorial** direktorija) simulira očitavanje napona sa DAQ plug-in modula.

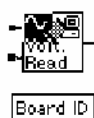


Multiply funkcija (iz **Functions>>Numeric**). U ovom primjeru ova funkcija će pomnožiti napon koji dolazi iz Demo Voltage read VI modula sa 100.0.



Numerička konstanta (iz **Functions>>Numeric**). Potrebne su dvije numeričke konstante: jedna za skal faktor od 100, a druga za konstantu uređaja. Za prvu numeričku konstantu, unjeti 100.0 kada se konstanta pojavi prvi put na blok dijagramu.

2. Kreirati drugu numeričku konstantu koristeći kraticu (shortcut) da bi se automatski kreirala i ožičila konstanta za Demo Voltage Read VI modul.



- a) Koristeći alat za ožičenje (wiring tool), kliknuti (pop-up) na ulaz označen *Board ID* na Demo Voltage Read VI-u i izabrati **Create Constant** iz pop-up menija. Ova opcija automatski kreira numeričku konstantu i ožičava je sa Demo Voltage Read VI.

- b) Ukucati 1 kada se konstanta pojavi prvi put na blok dijagramu. Ovo mijenja default vrijednost sa nule na jedinicu.

Napomena: Nije potrebno promjeniti na Labeling tool (alat za označavanje), da bi se unjela vrijednost, jer kurzor je već unutra i čeka na unosenje teksta.

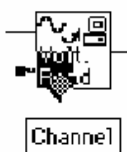
- a) Kliknuti na konstantu i izabrati **Show>>Label** . Koristeci alat za labeliranje, promjeniti default labelu (*Board ID*) na *Device*.

U ovom primjeru, dvije numeričke vrijednosti predstavljaju konstantu 100.0 i uređaj za multiply funkciju.

abc

3. Postaviti string konstantu (**Functions>>String**) na blok dijagram.

4. Koristeci alat za ožičenje (wiring tool), kliknuti na ulaz oznacen *Channel*, na donjem lijevom uglu **Demo Voltage Read VI** i izabrati **Create Constant** iz pop-up menija. Ova opcija automatski kreira string konstantu i ožičava je sa *Demo Voltage Read VI*.

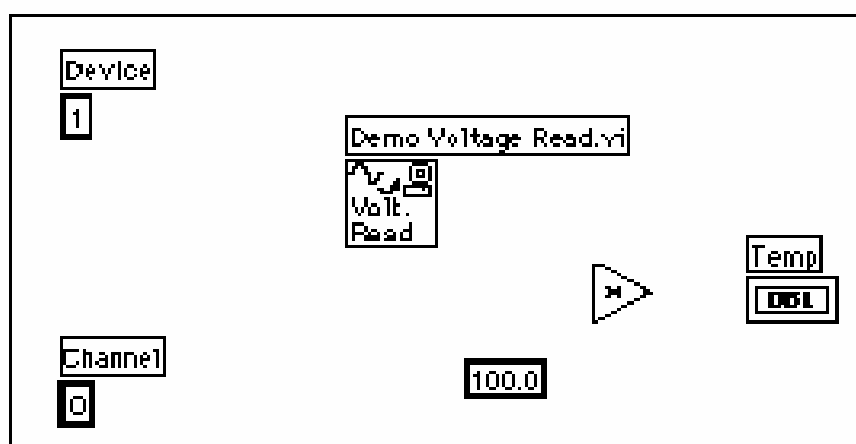


5. Ukucati 0 kada se konstanta pojavi po prvi put na blok dijagramu. Kliknuti na konstantu i izabrati **Show>>Label**. Primjetite da u ovom trenutku, *Channel* se pojavljuje kao default labela tako da nema potrebe da je mjenjate.

U ovom primjeru koristi se string konstanta da bi se predstavio broj kanala.

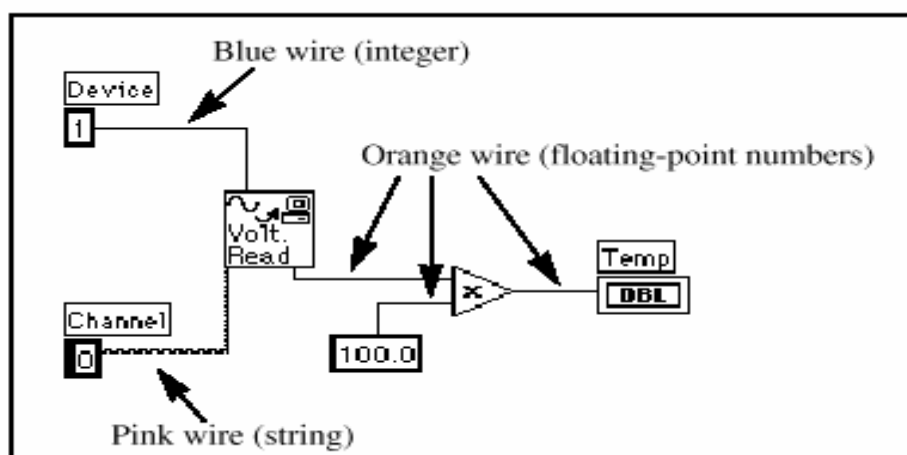
*Napomena : Kontrolni elementi, konstante i indikatori se mogu kreirati i ožičiti sa većinom funkcija. Ako ove opcije nisu na raspolaganju za neku specifičnu funkciju, **Create Control**, **Create Constant** i **Create Indicator** su blokirane (disabled) na pop-up meniju. Više informacija o ovim alatima bice dato kasnije u ovom poglavlju.*

Dakle, vi bi trebali da imate na vasem blok dijagramu **izvučene** (pulled down) sve objekte kao sto se to vidi na slijedećoj slici:



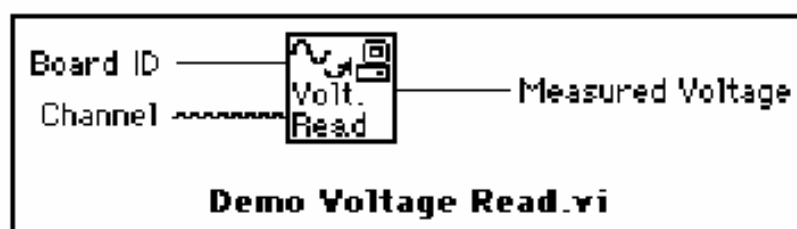
6. Koristeci alat za ožičenje , ožičiti preostale objekte kako ce biti kasnije objasnjeno u poglavlju o Tehnikama Ožičenja (Wiring Techniques)

*Napomena : Da bi se pomicali objekti unutar blok dijagrama, kliknuti na alat za pozicioniranje (Positioning tool) u **Tools** paleti.*



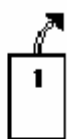
U LabVIEW zice su kodirane boje zavisno od vrste podataka koje svaka zica nosi. Plave zice nose integere, narandjaste zice nose floating-point brojeve (sa pokretnim zarezom), zelene zice nose Bulove , a ružičaste (pink) zice nose stringove (niz karaktera).

Help prozor se može pozvati birajući **Help>> Show help**. Postavljajući bilo koji od editirajućih alata na node, prikazuju se ulazi i izlazi te funkcije u help prozoru. Dok prelazite sa alatom za editiranje preko VI ikone, LabVIEW naglasava terminale (priključke) za ožičenje i u blok dijagramu kao i u help prozoru. Kada počnete ožičavanje vaših dijagrama, ovo treptajuće (flashing highlight) naglasavanje vam može pomoći da povežete ulaze i izlaze na odgovarajuće priključke (terminale).



Demo Voltage read VI simulira očitavanje napona na kanalu 0 plug-in ploče obezbjeđujući vjestacki (simulirano) podatke za **Measured Voltage** izlaz. Ovi podaci predstavljaju realnu temperaturu podjeljenu sa 100. VI nakon toga mnozi napon sa 100.0 da bi konvertovala **ocitani** napon u $^{\circ}\text{F}$.

TEHNIKE OZIČENJA



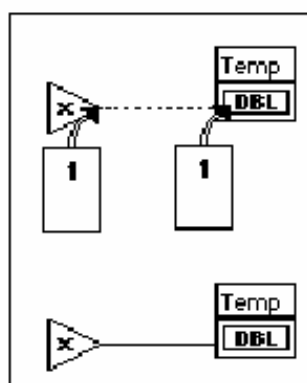
U prikazima ozičenja u ovoj sekciji, strijela na kraju ovog simbola misa pokazuje gdje treba kliknuti a broj pokazan na misu indicira koliko puta kliknutu taster misa.

Hot spot (vruća tačka) alata je vrh namotanog segmenta zice.



Da bi se ozičilo sa jednog terminala do drugog, kliknuti alat za ozičenje (wiring tool) na prvom terminalu, zatim prebaciti alat na drugi terminal i kliknuti na drugi terminal. Pri tome nije bitno na kojem se terminalu prvo pocinje.

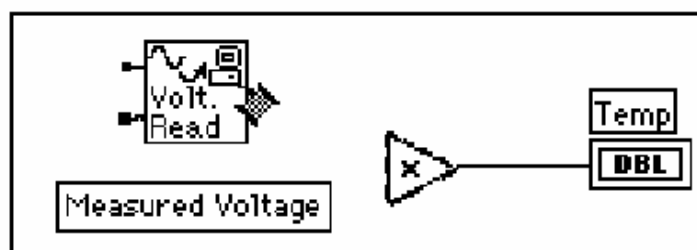
Kada je alat za ozičenje na terminalu, zona terminala blinka (treperi), da indicira da će klik povezati zicu na taj terminal. Ne smije se držati pritisnutim taster misa dok se prebacije alat ozičenja sa jednog terminala na drugi. Zica se može saviti na taj način da se mis pokreće okomito na tekuci pravac kretanja. Da bi se napravilo više pregiba na zici, potrebno je kliknuti na taster misa. Da se promjeni pravac zice, potrebno je pritisnuti taster (space bar) rastavnice. Kliknite na taster misa, da bi se **prikovala** zica za podlogu i zatim pokrenite mis u okomitom pravcu.



NATPISNE TABLICE (STRIPS)



Kada se pokreće alat ozičenja preko terminala čvora, iskace (pop-up) natpisna tablica za terminal. Tablica se sastoji od malih zutih tablica sa tekstom na kojima piše ime svakog terminala. Ove natpisne tablice trebaju pomoći u ozičenju terminala. Slijedeca ilustracija pokazuje tablicu (Measured Voltage - mjereni napon) koja se pojavljuje kada se postavi alat za ozičenje iznad izlaza Demo Voltage Read VI.



Napomena : Kada se postavi alat ožičenja iznad čvora (noda), LabVIEW pokazuje izdanke zica koje pokazuju svaki ulaz i izlaz. Izdanak zice ima tačku na kraju ako je riječ o ulazu u čvor.

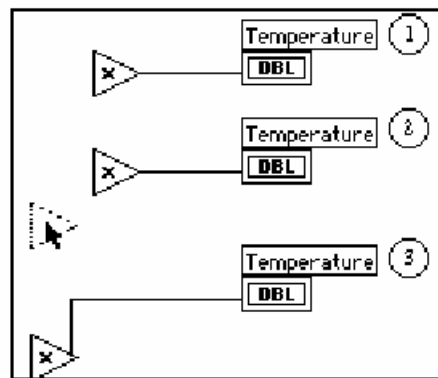
POKAZIVANJE TERMINALA

Vrlo je važno ožičiti korektno terminale funkcije. Može se prikazati ikona konektora da bi se olaksala korekcija ožičenja. Da bi se ovo učinilo, kliknite na funkciju i ponovno izaberite **Show>>Terminals**.

RASTEZANJE ZICE

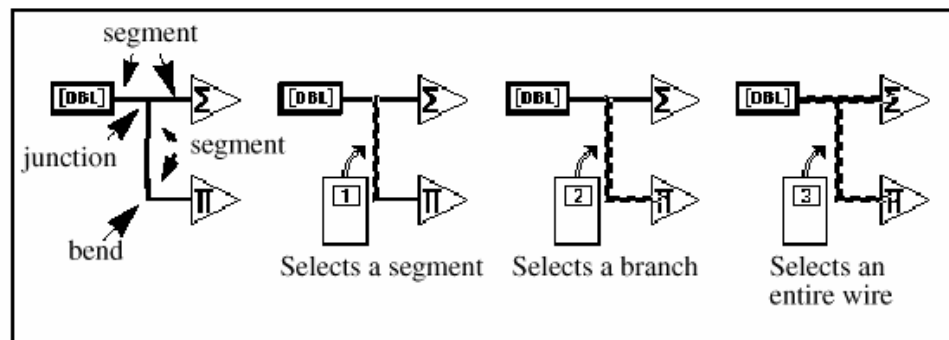


Moguće je pokretati ožičene objekte individualno ili u grupi, vukuci (dragging) selektirane objekte na novu lokaciju sa alatom za pozicioniranje (positioning tool).




IZBOR I BRISANJE ZICA

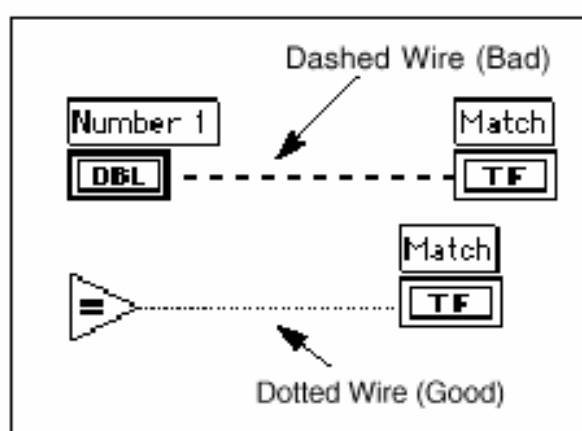
Moguće je napraviti grešku kod ožičenja čvorova. Ako se to desi, selektirati zicu koja se želi pobrisati a zatim pritisnuti taster **< Delete >**. Segment zice je horizontalni ili vertikalni odsječak zice. Tačka gdje se sreću tri ili četiri segmenta zice se naziva *junction* (spojiste). Grana zice (*wire branch*) sadrži sve segmente zice od jednog spojista do drugog, od terminala do slijedećeg spojista, ili od jednog terminala do drugog ako nema spojista između. Segment zice se selektira klikanjem na njega sa alatom za pozicioniranje. Klikanjem dva puta uzastopno vrši se selekcija grane, a trostruko klikanje selektira cijavu zicu.



LOSE ZICE (BAD WIRES)

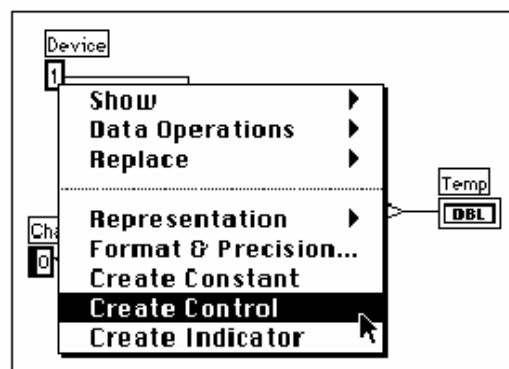
-
- ili
-  Crtkana linija predstavlja losu zicu. Losa zica se moze pojaviti zbog niza razloga , kao naprimjer povezivanje izlaza dva kontrolna elementa, pak povezujuci terminal izvora (source) sa terminalom destinacije kada tipovi podataka se ne slazu (napr. povezujuci numerički sa Bulovim podatkom). Dizajner moze otkloniti losu zicu klikom na nju sa alatom za pozicioniranje i pritiskom na <Delete> taster. **Birajući Edit>Remove Bad wires** brise sve lose zice u blok dijagramu. Ovo je vrlo korisna brza popravka (fix) koja se moze pokusati ako VI odbija da se izvrsava.

Napomena: Ne treba brkati , crtkanu liniju sa tačkastom linijom. Tačkasta linija predstavlja Bulov tip podatka, kao sto to pokazuje slijedeca slika :



KREIRANJE I OZIČENJE KONTROLNIH ELEMENATA, KONSTANTI I INDIKATORA

Za terminale koji djeluju kao ulazi na blok dijagramu, LabVIEW ima dvije osobine koje se mogu koristiti da se kreira i ozici kontrolni element ili konstanta. Ovim osobinama se moze pristupiti klikom (pop-up) na terminal a zatim biranjem **Create Control** ili **Create Constant**. LabVIEW automatski kreira i ozičava kontrolni ili tip konstante na terminalni ulaz. Slijedeca slika pokazuje primjer pop-up menija.



Za terminal koji djeluje kao izlaz blok dijagrama, može se izabrati **Create Indicator** osobinu da bi se kreirao a zatim ozicio indikator na terminalu. Pristupa se ovoj opciji klikom na terminal i birajući **Create Indicator** . LabVIEW automatski kreira i ozičava korektni tip indikatora na izlazu terminala.

*Napomena: Jedanput kada je izabran **Create Indicator**, mora se preključiti na prednji panel i koristiti alat za pozicioniranje da se izabere i pobriše indikator.*

IZVRSENJE VI

1. Da bi se učinio prednji panel aktivnim prozorom treba kliknuti na njegovu zaglavlje (title bar) ili biranjem **Windows>>Show Panel**. Takodjer je dovoljno kliknuti bilo gdje unutar prozora.



2. Izvršiti VI klikanjem na Run taster na toolbaru prednjeg panela.

3. Kliknuti na continuous run taster na toolbaru.



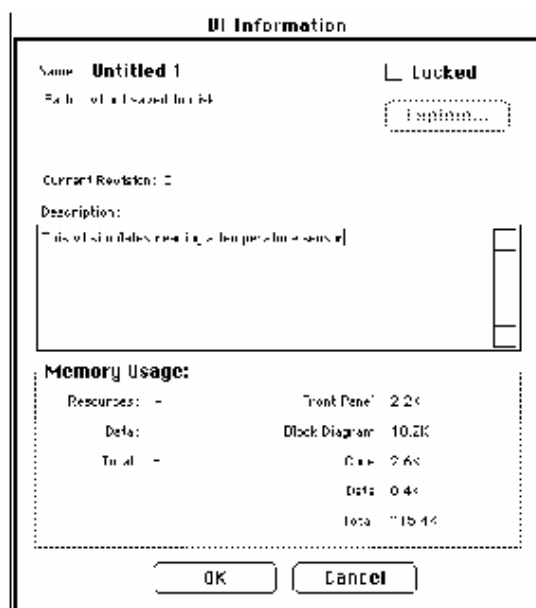
4. Kliknuti na continuous run taster opet da bi se deselektirao. Nakon toga VI kompletira izvršenje i isključuje se.

Napomena : Continuous run taster nije preferirani metod da bi se ponovilo izvršenje koda blok dijagrama. Dizajner treba koristiti konturnu (loop) strukturu. Ovo će se dalje analizirati u Poglavlju 3 , Konture i chartovi (Loops and charts).

DOKUMENTIRANJE VI MODULA

Dizajner može dokumentirati VI birajući **Windows>>Show VI Info** ... Nakon toga može ukucati opis VI u VI informacioni dijalog boks. Nakon toga može pozvati opis ponovno selektirajući **Windows>>Show VI Info**....

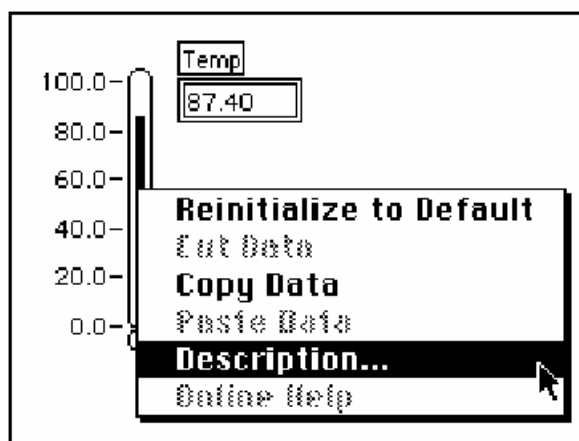
1. Dokumentirajte VI. Izaberite **Windows>>Show VI Info** ... Ukucajte opis za VI , kao što je pokazano na slijedećoj slici, a zatim kliknite na **OK**.



Može se gledati opis objekata na prednjem panelu (ili njihovih odgovarajućih terminala na blok dijagramu) klikom na objekat i izborom **Data Operations>>Description...**

Napomena: Ne može se mijenjati opis dok se izvrsava VI.

Slijedeca slika je jedan primjer pop-up menija koji se pojavljuje dok se izvrsava VI. Nije moguće dodati ili promijeniti opis dok se izvrsava modul VI, ali je moguće posmatrati prethodno unesene informacije.

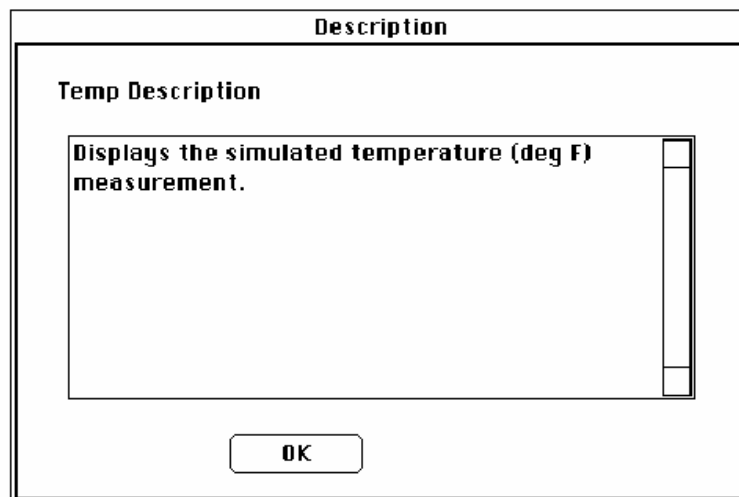


2. Dokumentovati termometarski indikator.

a) Na prednjem panelu, pop-up termometarski indikator i izabrati **Data Operations>>Description...**

b) Ukucati opis za indikator, kako je to pokazano na narednoj ilustraciji, a zatim kliknuti na **OK**.

3. Pokazati opis koji je kreiran ponovno, kliknuvsi na termometarski indikator i izabiruci **Data Operations>> Description...**



POHRANJIVANJE (SAVING) I PUNJENJE (LOADING) VI MODULA.

Kao i sa drugim aplikacijama, moguće je pohraniti VI u datoteku (file), u regularni direktorij (folder). U LabVIEW moguće je također pohraniti više VI-eva u jedan file koji se naziva VI biblioteka (VI library). **Tutorial.lib** biblioteka je jedan primjer VI biblioteke.

Ipak pohranjivanje VI-eva u individualne file-ove je mnogo efektivnije jer je moguće kopirati, promjeniti ime (rename) i pobrisati (delete) file-ove lakše nego kada se koristi VI biblioteka.

Pohranite VI koji si kreirao u VI biblioteku.

1. Izabrati **File>>Save As...**
2. Dati ime VI i pohraniti je kao mywork.llb.
3. Ukucati My Thermometer.vi u dijalog boks.
4. Kliknuti na **OK**.
5. Zatvoriti VI birajući File>>Close.

POGLAVLJE 3

KREIRANJE SUBVI-EVA

Razumjevanje hijerarhije

Jedan od ključeva za kreiranje LabVIEW aplikacija je razumjevanje i korištenje hijerarhijske prirode VI-eva. Nakon kreiranja VI, dizajner je može koristiti kao subVI u blok dijagramu VI-eva višeg nivoa. Zbog toga, subVI je analogna subrutini u C jeziku. Isto kao što nema ograničenja u broju subrutina koje se mogu koristiti u C programu, tako isto nema ograničenja u broju subVI-eva koje se mogu koristiti u LabVIEW programu. Dizajner može također pozivati subVI unutar druge subVI.

Kada se kreira aplikacija, počinje se sa top-level VI (na najvišem nivou) i definišu ulazi i izlazi za aplikaciju. Nakon toga, konstruišu se subVI-i da izvršavaju neophodne operacije nad podacima kako one protiču kroz blok dijagram. Ako blok dijagram ima veliki broj ikona, potrebno ih je grupisati u VI-eva nižeg nivoa, da bi se održala jednostavnost blok dijagrama. Ovaj modularni pristup čini aplikaciju jednostavnom za debugiranje (testiranje programa), razumjevanje i održavanje.

KREIRANJE SUBVI-EVA

CILJ Kreirati ikonu i konektor za **My Thermometer VI** koju smo kreirali u poglavlju 1 i koristiti tu VI kao subVI.

Da bi se koristila VI kao subVI, mora se kreirati ikona da bi prikazivala tu VI u blok dijagramu VI, kao i konektor pona na koji se mogu povezati ulazi i izlazi.

IKONA



Kreirati ikonu koja predstavlja VI u blok dijagramu druge VI.

Ikona može biti slikovna predstava namjene VI, ili pak može biti tekstualni opis VI ili njenih terminala.

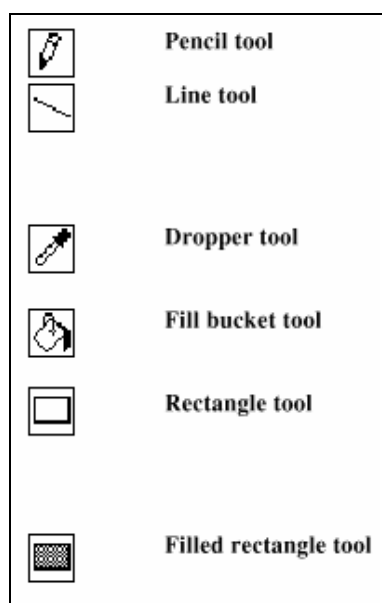
1. Ako VI **My Thermometer** nije otvorena, otvoriti je sa izborom **File>>Open...** ili klikanjem na **Open VI** taster u dijalog boks. Otvoriti mywork.llb u Windows-ima; ova se biblioteka može naći u privremenom (temporary) direktoriju ili u direktoriju windows\temp.

2. Izabrati **My Thermometer VI** iz mywork.llb

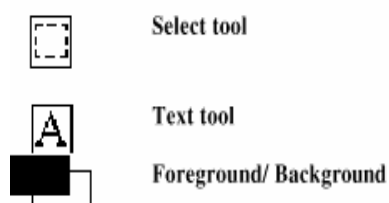
3. Pozvati **Icon Editor** kliknuvši na površini ikone u gornjem desnom uglu prednjeg panela i birajući **Edit Icon**. Kao shortcut (kratica) moguće je također dvostruko kliknuti na površini ikone da bi se ona editirala.



Alati i tasteri Editora ikona



- Crta i brise piksel po piksel
- Crta pravu liniju. Pritisnuti <Shift> a zatim vucite ovaj alat da bi se nacrtala horizontalna, vertikalna i dijagonalna linija.
- Kopira boju prednjeg plana (foreground) iz nekog elementa u ikonu.
- Puni oznacenu zonu sa bojom prednjeg plana (foreground).
- Crta rektangularne granice u boji prednjeg plana (pp). Dvaput kliknuti na ovaj alat da se uokviri ikona u boji prednjeg plana(pp).
- Crta kvadraticni okvir obrubljen bojom pp i ispunjen sa bojom zadnjeg plana (zp). Ako se dvaputa klikne, uokvirava se ikona u pp boju i puni sa zp bojom.



Selektira zonu oko ikone da bi se mogla kretati, klonirati (kopirati), ili drugo. Unosi tekst u dizajn ikone.

Prikazuje tekuce boje pp i zp. Kliknuti na svaku da se dobije paleta boja iz koje se onda moze izabrati nova boja ako se zeli promjeniti.

Tasteri desno od ekrana za editiranje izvrsavaju slijedece funkcije;

Undo Ponistava posljednu operaciju editiranja

OK Pohranjuje crtez kao VI ikonu i vraca se na prednji panel

Cancel Vraca se na prednji panel bez pohranjivanja promjena



4. Brise default ikonu.
 - a) Sa select alatom , izabrati unutarnju sekciju default ikone, pokazane sa lijeve strane.
 - b) Pritisnuti <Delete> taster da bi se izbrisala unutrasnjost default ikone.

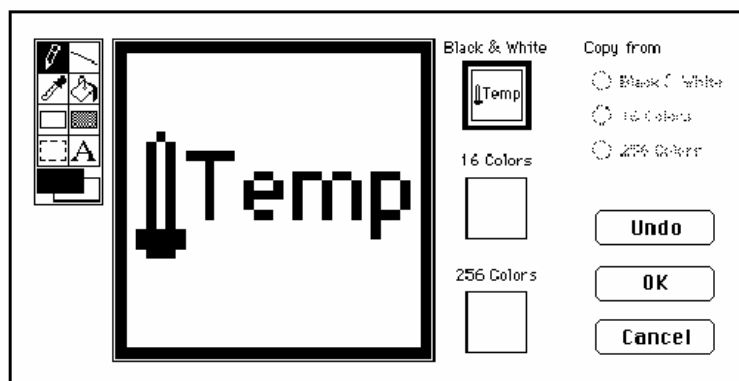


5. Nacrtati termometar sa alatom olovke.



6. Kreirati tekst sa Tekst alatom. Da bi se promjenio tip slova (font) teksta, dvaput kliknite na Text alat. Eksperimentirati sa editorom da bi se naucili alati.

Vasa ikona treba sada izgledati otprilike kao na slijedejoj slici:

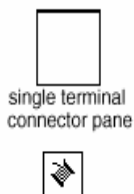


7. Zatvoriti Icon editor kliknuvši na **OK** kada je završite. Nova ikona će se pojaviti na ikonskom polju (pane) u gornjem desnom uglu prednjeg panela.

KONEKTOR

Sada se može kreirati konektor.

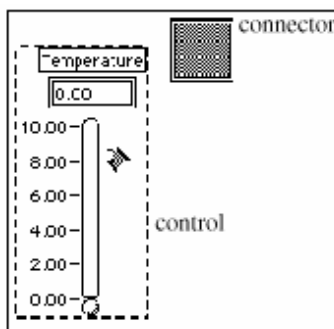
1. Definirati konektorski terminal kliknuvši na ikonsko polje (pane) na prednjoj ploči i birajući **Show Connector**, kako to pokazuje slijedeca ilustracija.



Posto LabVIEW izabire oblik terminala baziran na broju kontrolnih elemenata i indikatora na prednjem panelu, u ovom slučaju postoji samo jedan terminal - indikator termometra.

2. Doznacite terminal za termometar.
 - a) Kliknite na terminal u konektoru. Kurzor se automatski mjenja u alat za ozicanje, a terminal će pocrniti.

- b) Kliknuti na termometarski indikator. Kretanjem okvira crtkane linije, obuhvata se indikator kako to pokazuje slijedeca ilustracija:



Ako se klikne na otvorenu površinu prednjeg plana, crtkana linija iscezava i selektirani terminal posivi (dims), indicirajući na ovaj način da ste doznacili indikator terminalu. Ako je terminal bijel, niste napravili spoj korektno. Ponovite prethodne korake ako je neophodno.

3. Pohraniti VI birajući **File>>Save**. VI je sada kompletan i spreman za korištenje kao subVI u drugim VI-evima. Ikona predstavlja VI u blok dijagramu pozivajuće VI. Konektor (sa jednim terminalom) daje na izlazu temperaturu.

Napomena: Konektor specificira ulaze i izlaze u VI kada ga koristite kao subVI. Potrebno je imati na umu da kontrolni elementi prednjeg panela mogu biti korišteni samo kao ulazi: indikatori prednjeg panela mogu biti samo kao izlazi.

4. Zatvoriti VI birajući **File>>Close**.

KORISTENJE VI KAO SUBVI

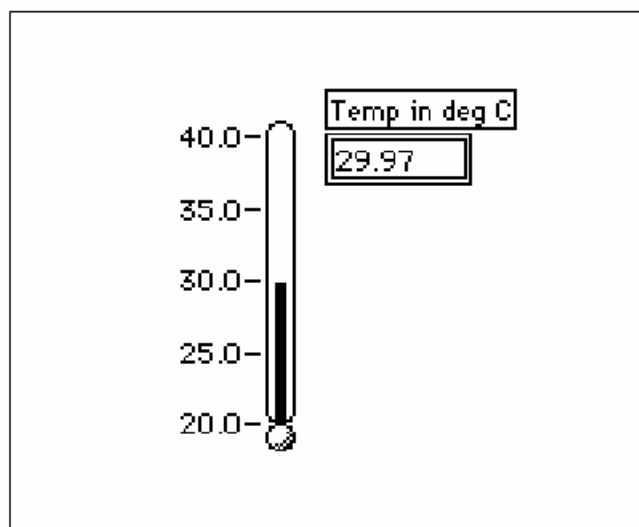
Moguće je koristiti bilo koju VI koja ima ikonu i konektor kao subVI u drugoj VI. U blok dijagramu, izvršite izbor VI-eva koje ćete koristiti kao subVI-eve izabравsi ih u **Functions>Select VI**. Izabравsi ovu opciju pojavljuje se file dijalog boks, iz kojeg možete izabrati bilo koju VI u sistemu. Ako otvorite VI koja nema ikonu i konektor, bijeli prazan kvadratni boks se pojavljuje u blok dijagramu pozivajuće VI. Nije moguće označiti ka ovome cvoru.

SubVI je analogna subrutini. SubVI cvor (ikona/konektor) je analogan pozivu subrutine. SubVI cvor ne predstavlja samu subVI, kao što nije ni call statement u subrutini (instrukcija poziva), nije sama subrutina. Blok dijagram koji sadrži nekoliko identičnih subVI cvorova poziva istu subVI nekoliko puta.

CILJ Izgraditi VI koju će koristiti **My Thermometer** VI kao subVI.

My Thermometer VI koji dizajniramo isporučuje temperaturu u stepenima Fahrenheit-a. Ovu temperaturu ćemo dalje konvertovati u stepene Celzijusa.

PREDNJI PLAN



1. Otvoriti novi prednji panel birajući **File>>New** ili klikom na New VI taster u dijalog boksu.
2. Izabrati termometar iz **Controls>>Numeric**. Ukucati **Temp in deg C** da bi se oznacila. Ako ste kliknuli van termometra prije ukucavanja labele, ona ce isceznuti. Da bi se pokazala labele (natpisnica) ponovno, kliknuti na termometar i izabrati **Show>>Label** a zatim ukucati vasu labele.



3. Promjeniti opseg termometra da se prilagode vrijednosti temperature. Sa operativnim alatom, dvaputa kliknite na donju granicu, ukucati 20, a zatim pritisnuti <Enter> na tastaturi. Nema potrebe ukucavati decimalnu tacku ili nule poslije nje. LabVIEW ih dodaje automatski. Slicno, promjeniti gornju granicu termometra na 40 i pritisnuti <Enter>. LabVIEW automatski podesi medjuvrijednosti.

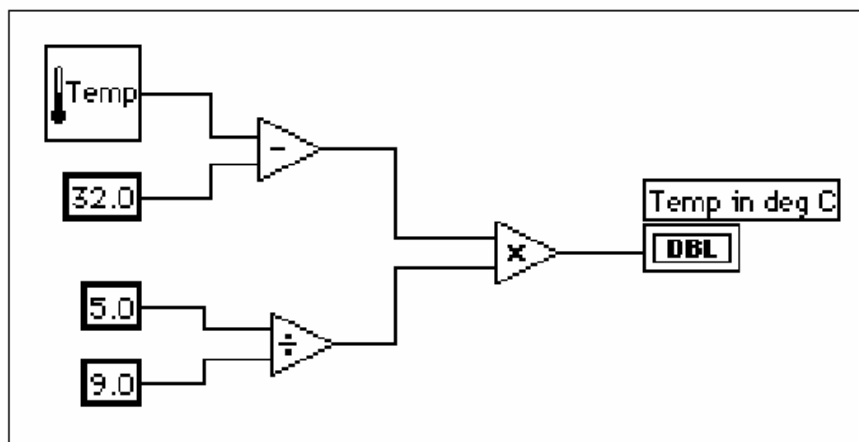
Svaki put kada se kreira novi kontrolni element ili indikator, LabVIEW automatski kreira odgovarajuci terminal u blok dijagramu. Simboli terminala sugeriraju tip podatka kontrolnog ili indikacionog elementa. Naprimjer, DBL terminal predstavlja dvostruku preciznost, TF je Bulov terminal, I16 terminal predstavlja regularni 16-bitni integer (cijeli broj); a ABC predstavlja string (niz karaktera).

BLOK DIJAGRAM

1. Izabrati **Windows>>Show Diagram**.

2. Klinuti (pop-up) u slobodnom prostoru blok dijagrama i izabrati **Functions>>Select a VI** Pojavljuje se dijalog boks. Locirati i otvoriti mywork.llb biblioteku. (u Win95/98 , mozete naci ovu biblioteku u temporary (privremenom) direktoriju windows\temp. Dvapat kliknuti na **My Thermometer VI** ili ga oznacite i kliknite na **Open** u dijalog boks. LabVIEW postavlja **My thermometer VI** na blok dijagram.

3. Dodati druge objekte u blok dijagram kako je to pokazano na slijedejoj ilustraciji:



1.23

Numericka konstanta (**Functions>>Numeric**).Dodati tri numericke konstante na blok dijagram. Doznaciti vrijednosti 32.0, 5.0, i 9.0 za konstante koristeći alat za labeliranje.

NAPOMENA: Vi mozete zakljuciti o tipu konstante po njenoj boji. Plave numericke konstante su integeri (cijeli brojevi), a narandjaste konstante su brojevi dvostruke preciznosti. LabVIEW automatski konvertuje brojeve u odgovarajuci format kada je to potrebno.

NAPOMENA: Podsjetite se mogucnosti da kliknete na funkciju i izaberete *Create Constant* da se automatski kreira i ozici korektna konstanta sa funkcijom.



Funkcija oduzimanja (Functions>>Numeric) oduzimaju 32 od vrijednosti u Fahrenheitima za konverziju u stepene Celzijusove.



Funkcija dijeljenja (Functions>>Numeric) racuna vrijednost od 5/9 za konverziju temperature.



Funkcija mnozenja (Functions>>Numeric) vraca vrijednost u stepenima Celzijusa iz procesa konverzije.

4. Provesti ozicenje objekata u blok dijagramu kako je to ranije demonstrirano.

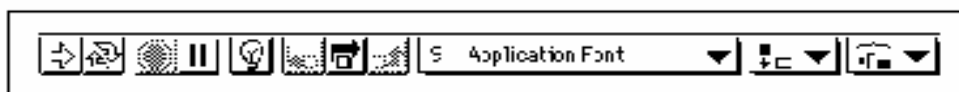
NAPOMENA: Izlomljena linija izmedju ikone termometra i terminala **Temp in deg C** moze indicirati da ste doznacili subVI konektorski terminal nekorektno sa indikatorom na prednjem panelu. Nakon modifikacije subVI, mozda ce biti potrebno izabrati **Relink to subVI** iz pop-up menija ikone. Ako je potrebno izaberite **Edit>>Remove bad Wires**.



5. Vratiti se na prednji panel i kliknuti na run taster u toolbaru.

BLOK DIJAGRAM TOOLBAR

Blok dijagram sadrzi dodatne opcije na toolbaru prednjeg panela, kako se vidi iz slijedece slike:



Blok dijagram toolbar sadrzi slijedece tastere koje dizajner moze koristiti za debugiranje VI (otklanjanje programskih gresaka).



Hilite execute button

Pokazuje podatke kako prelaze preko zica.



Step into button

Stepuje u konture (loops), subVI, itd.



Step over button

Pocinje step korak po korak, stepuje preko konture, subVI, itd.



Step out button

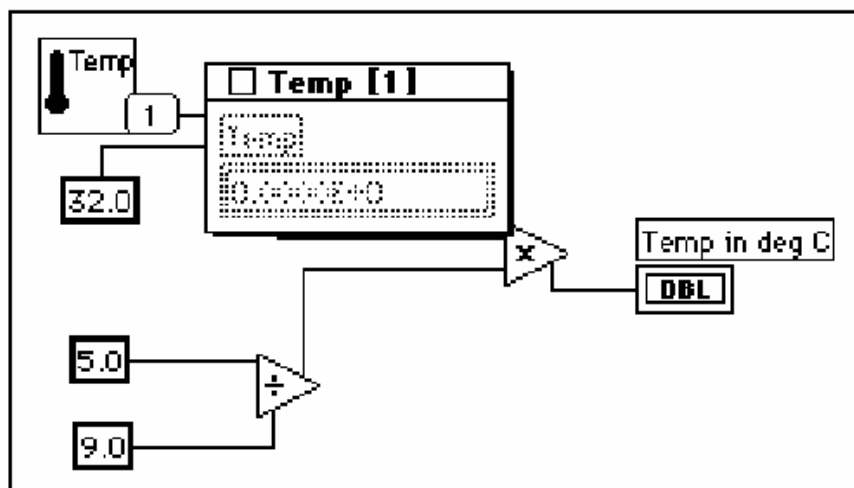
Kompletira izvršenje kontura, VI-eva, blok dijagrama, itd.

TEHNIKE DEBAGIRANJA

Termometar treba prikazati vrijednost u selektiranom opsegu. Predpostavimo da zelimo vidjeti vrijednost u Fahrenheitima radi poredjenja i debugiranja. LabVIEW sadrzi neke alate koji mogu pomoci u ovome. U obom primjeru, ispitacemo tester (probe) i izvršenje programa.

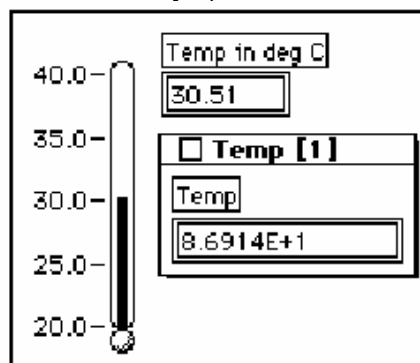
1. Izabrati **Windows>>Show Diagram**.
2. Izabrati alat za test (probe) iz **Tools** palete. Kliknuti sa testerom na vrijednost temperature (zicu) koja dolazi iz **My thermometer** subVI. Tester prozor ce iskociti sa naslovom **temp 1** kao i zuti urez koji pokazuje broj testera, kao sto se to vidi na slijedecej slici. Prozor testera se takodjer pojavljuje na prednjem panelu.





3. Vratiti se na prednji panel. Pomaci testerski prozor tako da se mogu vidjeti i vrijednost testera i termometra kako je to pokazano na slijedećoj ilustraciji. Izvršiti VI (run). Temperatura u stepenima Fahrenheita ce se pojaviti u prozoru testera.

*NAPOMENA: Vrijednosti temperature koje se pojavljuju na ekranu mogu biti različite nego sto se vidi na ovoj slici. Ukoliko je tako, pogledati u Poglavlju 3 **Konture i chartovi** (Loops and charts) u sekciju Numeric Conversion (numericka konverzija).*



4. Zatvoriti tester prozor, klikom na zatvoreni boks na vrhu titlebara testerskog prozora.

Druga korisna debaging tehnika je ispitivanje toka podataka u blok dijagramu koristeći LabVIEW karakteristiku naglasavanja izvršenja.



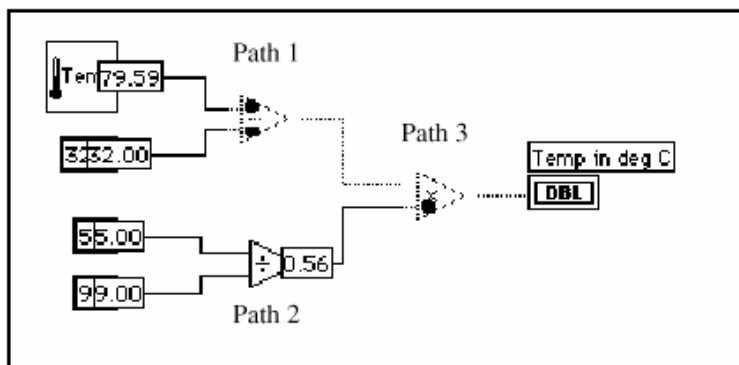
5. Vratiti se na blok dijagram VI izvršivši **Windows>>Show Diagram**.



6. Početi naglasavanje izvršenja klikom na **hilite execute** taster u toolbaru, koji je pokazan na lijevoj strani. Nakon toga ovaj taster ce se promjeniti u osvjetljenu sijalicu (vidi lijevo).

7. Kliknuti na taster izvršenja da bi se izvršila VI, i primjetiti da naglasavanje izvršenja animira izvršenje VI blok dijagrama. Pokretne lampice predstavljaju tok podataka kroz VI. Također primjetiti da se vrijednosti podataka pojavljuju na zicama i prikazuju vrijednosti sadržane u zicama u tom trenutku, kako je to

pokazano na slijedecem blok dijagramu, kao da smo upravo dohvatili testerom zicu.



Primjetimo redosljed u kojem se izvrsavaju razliciti cvorovi u LabVIEW. U konvencionalnim tekst baziranim programskim jezicima, programski iskazi se izvrsavaju u redosljedju u kojem se pojavljuju. LabVIEW, pak, koristi data flow programiranje, kod kojeg cvor se izvrsava kada je podatak na raspolaganju na svim ulazima cvora, a ne u redosljedju **top-to-bottom** (odozgo - na- dole) ili sa lijeva na desno.

Prethodna slika pokazuje da se LabVIEW moze izvrsavati istovremeno (multitask - viseprograma), izmedju staza 1 i 2 jer medju njima nema medjuzavisnih podataka, tj. nista u stazi 1 ne zavisi od podataka u stazi 2 i nista u stazi 2 ne zavisi od podataka u stazi 1. Staza 3 se mora izvrsiti posljednja posto funkcija mnozenja je zavisna od funkcija oduzimanja i djeljenja.

Naglasavanje izvrsenja je koristan alat za ispitivanje prirode toka podataka u LabVIEW.

Vi mozete koristiti taster za korak po korak (single step) izvrsenje programa ako zelite vise kontrole nad procesom debugiranja.



8. Pocnite single stepping klikom na **step over** taster na toolbaru. Klikom na ovaj taster prikazace se prva izvrsiva sekvenca u VI programu. Nakon sto LabVIEW kompletira ovaj dio sekvence, on ce naglasiti slijedeci dio koji ce se izvrsiti u VI-u.



9. Predjite (step over) preko funkcije djeljenja klikom na **step over** taster (na lijevoj strani). Klikom na ovaj taster, izvrsice se funkcija djeljenja. Nakon sto LabVIEW kompletira ovaj dio sekvence, on ce naglasiti slijedeci dio koji ce izvrsiti u VI.



10. Udjite u (step into) **My Thermometer** subVI, tako sto cete kliknuti na **step into** taster na toolbaru. Klikanjem otvorice se prednji panel i blok dijagram od razvijene subVI. Vi sada mozete izabrati da ili idete korak po korak (single step) kroz subVI ili da je cijelu izvrsite.



11. Zavrсите izvrsenje blok dijagrama klikanjem na iskoracni (**step out**) taster na toolbaru. Klikanjem na ovaj taster kompletiraju se sve preostale sekvence u blok dijagramu. Nakon sto LabVIEW kompletira ovaj dio sekvence, on naglasava slijedeci dio koji ce se izvrsiti u VI. Mozete takodjer zadržati taster misa nakon klikanja na **step out** taster da bi pristupili pop-up meniju. Na ovom pop-up

meniju, mozete izabrati koliko daleko ce se izvorsavati VI prije nego pauzira (pausing). Slijedeca slika ilustruje vase opcije za završno izvršenje u pop-up meniju **step out** tastera.



12. Izabrati **File>>Save as** i pohraniti VI u mywork.llb. Dati ime VI kao **Using My Thermometer.vi** i nakon toga zatvori file.

OTVARANJE, RAD SA, I PROMJENE SUBVI-EVA.

Mozete otvoriti VI koja je koristena kao subVI iz blok dijagrama pozivajuće VI. Otvarate blok dijagram subVI na taj način što dvaput kliknete na subVI ikonu ili izabiruci **Project>>This VI's SubVIs**. Nakon toga otvarate blok dijagram selektirajući **Windows>>Show Diagram**.

Bilo koja promjena napravljena na subVI mjenja samo verziju u memoriji sve dok ne pohranite (save) tu subVI. Zapazite da promjene uticu na sve pozive te subVI a ne samo na cvor koji je koristen da se otvori VI.

HIJERARHIJSKI PROZOR

Koristite Hijerarhijski prozor (**Project>>Show VI Hierarchy**) da bi se vizelizirao prikaz medjuzavisnosti VI-eva, obezbjedjujuci informaciju o VI pozivaocima subVI-eva (callers) i samim subVI-evima. Ovaj prozor sadrzi toolbar koji mozete koristiti da konfigurisete nekoliko tipova postavnih vrijednosti za prikazane elemente. Slijedeca ilustracija pokazuje primjer toolbara VI hijerarhije.



Dizajneri mogu koristiti tastere na toolbaru za Hijerarhiju prozora ili pak **VIEW** meni, ili pop-up na prazni prostor u prozoru da bi se pristupilo slijedecim opcijama:

**Redraw**

inzira cvorove nakon sukcesivnih zahvata na hijerarhijskim cvorovima ako je potrebno minimizirati presjecanje linija i maksimizirati simetricnost. Ako fokusni cvor postoji, vi mozete proci kroz prozor tako da prvi korjen koji pokazuje subVI je vidljiv.

**Switch to vertical layout**

ira cvorove odozgo na dole (top-to-bottom) , stavljacuci korjene na vrh.

**Switch to horizontal layout**

ira cvorove sa lijeva na desno , postavljajuci korjene na lijevu stranu.

**Include/Exclude VIs in VI libraries**

ipa hijerarhiski graf da ukljuci ili iskljuci VI-eva u VI bibliotekama.

**Include/Exclude global variables**

ipa hijerarhijski graf da ukljuci ili iskljuci globalne varijable.

**Include/Exclude typedefs**

lapa hijerarhijski graf da ukljuci ili iskljuci typedefs.

Dodatno, **View** meni i pop-up meniji ukljucuju **Show all Vis** i **Full VI path i Label** opcije kojima ne mozete pristupiti na toolbaru.



Dok pomicete operativni alat preko objekata u hijerarhijskom prozoru, LabVIEW prikazuje ime VI ispod VI ikone.

Koristiti <Tab> taster da se preklopi izmedju prozorskih alata za pozicioniranje i prolazanje (scrolling) . Ova osobina je korisna za prebacivanje cvorova iz hijerarhijskog prozora u blok dijagram.

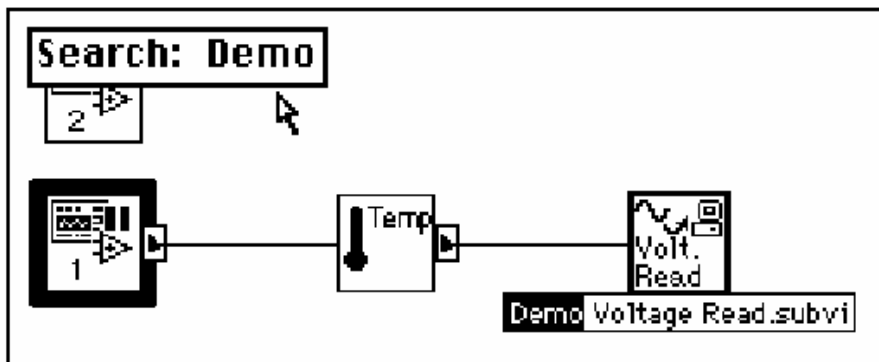
Vi mozete vuci (drag) VI ili subVI cvor u blok dijagram ili kopirati ga na clipboard (windows medjumemoriju za kopiranje medju windows aplikacijama) klikanjem na cvor. <Shift> klik na VI ili subVI cvor da bi se selektiralo vise objekata za kopiranje u druge blok dijagrame ili prednje panele. Dvaputa klikanje na VI ili subVI otvara prednji panel tog cvora.

Svaki VI koji sadrzi subVI ima jedan taster sa strelicom uz VI koji se moze koristiti da se pokaze ili sakrije subVI. Klikanjem na crveni taster sa strelicom ili dvaput klikanjem na samu VI otvara se subVI od te VI. Taster sa crnom strelicom na VI cvoru znaci da su sve subVI prikazane. Vi takodjer mozete pop-up na VI ili subVI cvor da bi se pristupilo meniju sa opcijama, kao sto je pokazivanje ili sakrivanje VI-eva , otvaranje VI ili subVI prednjeg panela, editiranje ikone za VI, itd.

HIJERARHIJA TRAZENJA

Mozete takodjer pretrazivati trenutno vidljive cvorove u Hijerarhijskom prozoru poimenicno. Vi inicirate trazenje ukucavanjem imena cvora, bilo gdje na

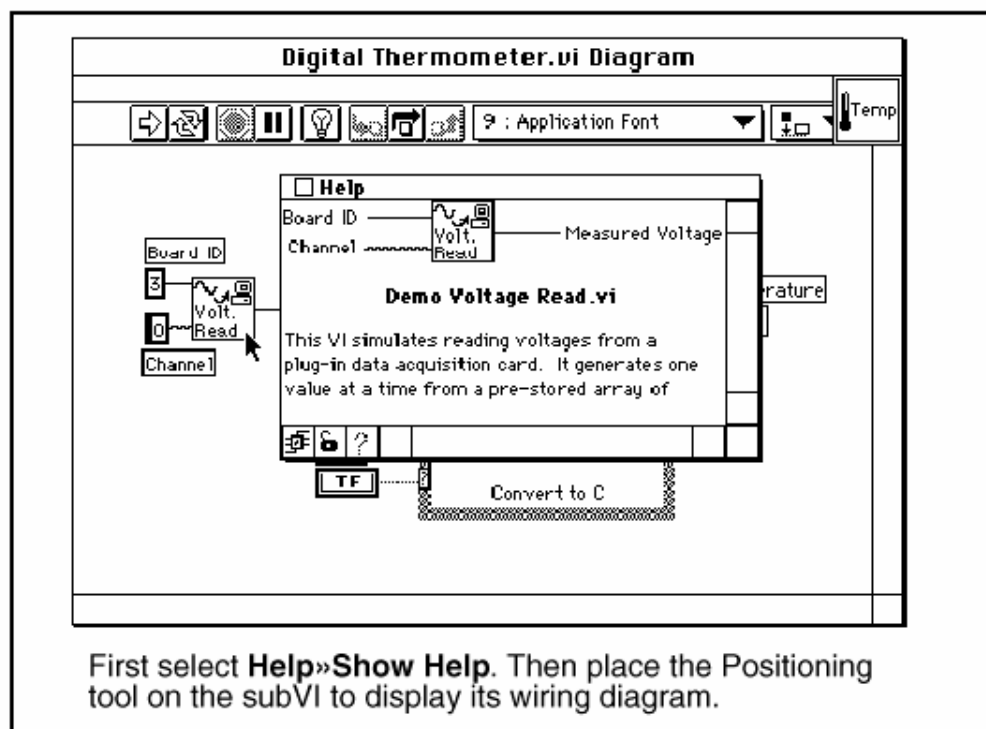
prozoru. Dok vi ukucavate tekst, pojavljuje se prozor za trazenje (search window), koji prikazuje tekst dok ga vi ukucavate i istovremeno pretrazuje kroz hijerarhiju. Slijedeca slika pokazuje hijerarhiju pretrazivanja:



Nakon nalazenja korektnog cvora, mozete pritisnuti < Enter> da trazi slijedeci cvor koji se slaze sa stringom (nizom karaktera) koji trazimo, ili mozete pritisnuti <Shift-Enter> da ponovo nadje prethodni cvor koji se slaze sa stringom trazenja.

ONLINE HELP ZA SUBVI CVOROVE

Kada postavite jedan od alata na subVI cvor, **Help** prozor pokazuje ikonu za subVI sa zicama prikacnim za svaki terminal. Slijedeca ilustracija pokazuje primjer online help-a. Ovo je digitalni termometar VI iz **Functions>>Tutorial**. Termometarski primjer VI kojeg kreirate takodjer sadrzi tekst koji ste ukucali u VI informacioni dijalog boks.



SIMPLE/COMPLEX HELP POGLED

U help prozoru, mozete specificirati da li zelite da prikazete jednostavan ili kompleksni pogled za objekte blok dijagrama.

NAPOMENA: Kada otvorite Help prozor, LabVIEW automatski default-ira ka jednostavnom help prozoru.

U jednostavnom help prozoru, LabVIEW prikazuje samo zahtjevane i preporucene ulaze za VI-eve i funkcije. U kompleksnom help pogledu, LabVIEW prikazuje zahtjevane, preporucene, i opcione ulaze za VI-eve i funkcije. Takodjer prikazuje puno ime i direktorijsku lokaciju (path) za VI. Da bi se pristupilo jednostavnom help pogledu, pritisnuti Simple/Complex Diagram help preklopnik, ili izabrati **Help>>Simple Diagram Help**. Slijedeca ilustracija pokazuje obadva izgleda Simple/Complex Diagram help preklopnika.

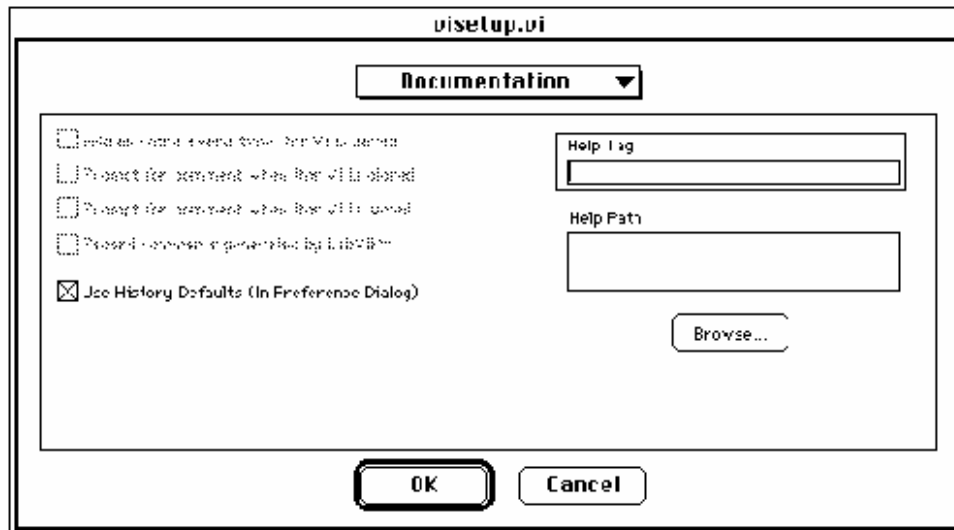


U help prozoru, zahtjevani ulazi se pojavljuju u masnom (**bold**) tekstu, preporuceni ulazi se pojavljuju u punom obicnom tekstu, a opcioni ulazi se pojavljuju u sivom tekstu. Kada dizajnirate svoju VI, mozete specificirati koji su ulazi zahtjevani, preporuceni ili opcioni, putem klicanja (pop-up) na neki ulaz ili izlaz na konektorskom simbolu (pane) i selektirajuci korektnu opciju iz submenija **This Connection is** .

VEZE SA ONLINE HELP FILE-OVIMA



U help prozoru, mozete kliknuti na online help taster da bi pristupili LabVIEW online help kao i help file-ovima koje ste kreirali koristeci help kompajler (compiler - program za prevodjenje razvijenog source kode programa u masinski - izvrsni kod). Ako zelite da kreirate vas vlastiti help file, morate specificirati vezu (link) sa help file-om, klicanjem na simbol polje (pane) ikone i izabiruci **VI Setup** . Kada se VI setup dijalog boks otvori, izabrati **Documenation** iz ring kontrole na vrhu boksa, i nakon toga unjeti stazu (path) help filea u Help Path boks. Slijedeca slika pokazuje opcije koje se pojavljuju u VI Setup dijalog boks.



Selektirajte **Browse** da pridružite help file kao i temu da pridružite sa vasim VI.

POGLAVLJE 4

KONTURE I CHARTOVI (LOOPS AND CHARTS)

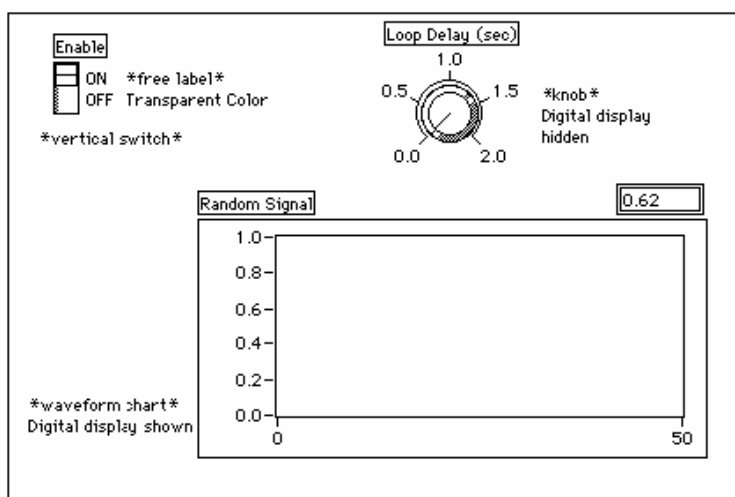
Uvod

Strukture kontrolisu tok podataka u VI. LabVIEW ima cetiri strukture: while konturu (**while loop** , tj. vrti se **dok** je neki uslov isponjen), For konturu (**for loop**), case strukturu (**case structure**) i sekvencijalnu strukturu (**sequence structure**). Ovo poglavlje ce uvesti strukture while i for kontura zajedno sa chart i sift registrima (shift register). Case i sekventne strukture ce biti objasnjene u poglavlju 5.

KORISTENJE WHILE KONTURA I CHARTOVA

CILJ Koristice se while konture i chartovi za prikupljanje i prikazivanje podataka u realnom vremenu.

Izgradicemo VI koja generira slucajne podatke i prikazuje ih na chartu. Kontrolni element dugmeta na prednjem panelu ce podesavati brzinu konture izmedju 0 i 2 sekunde a prekidač ce zaustaviti VI. Naucicemo kako promjeniti mehanicku akciju prekidača tako da nema potrebe da se ukljuci svaki put kada se izvrsava VI. Pocnimo od prednjeg panela datog na narednoj slici:



1. Otvoriti novi prednji panel izabiruci **File>>New** ili klikanjem na New VI taster u dijalog boks.



2. Postaviti vertikalni prekidač (**Controls>>Boolean**) na prednji panel. Labelirati (oznaciti) prekidač sa **Enable** . Mi cemo koristiti ovaj prekidač da zaustavimo prikupljanje podataka.



3. Koristiti alat za labeliranje da se kreira slobodna labela za ON i OFF. Mozete kreirati ove natpise klikanjem na labeling alat a zatim na prednji panel i ukucavajući tekst labele. Koristiti alat za boju da se ucini ivica labele nevidljivom. (klikanjem na alat za boju i izborom T(transparent) u donjem lijevom uglu palete).



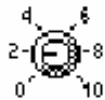
4. Postaviti graf valnog oblika (**Controls>>Graph**) na prednji panel. Oznaciti graf sa **Random Signal**. Chart ce prikazati slucajne podatke u realnom vremenu.

NAPOMENA: Budite sigurni da ste izabrali **waveform chart** a ne **waveform graph**. U Graph peleti, **waveform chart** se pojavljuje najblize lijevoj strani palete.

5. Kliknuti na chart i izabrati **Show>>Digital Display**. Digitalni prikaz pokazuje posljednje vrijednosti.



6. Koristeci alat za oznacavanje (labeling), dvaput kliknuti na 10.0 u chartu, ukucati 1.0, i zatim kliknuti van zone labeliranja. Klik unosi vrijednost. Mozete takodjer pritisnuti na taster <Enter>.



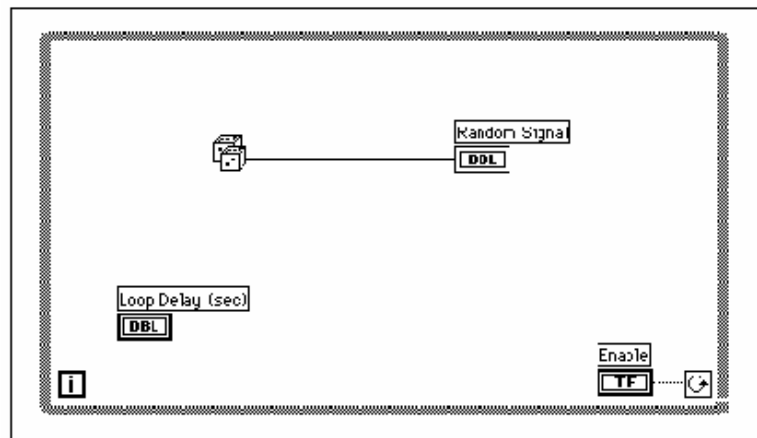
7. Postaviti dugme (Controls>>Numeric) na prednji panel. Oznaciti dugme sa **Loop**

Delay (sec- kasnjenje konture). Ovo dugme ce kontrolisati vrijeme za while konturu u ovom primjeru. Kliknuti (pop-up) na dugme i izvršiti deselkciju (ukidanje selekcije) **Show>>Digital Display** da se sakrije digitalni display koji se po default vrijednosti pokazuje.

8. Koristeci alat za labeliranje, dvaput kliknuti na 10.0 na skalu oko dugmeta, ukicati 2.0 i kliknuti van oblasti labele da bi se unijela nova vrijednost.



BLOK DIJAGRAM

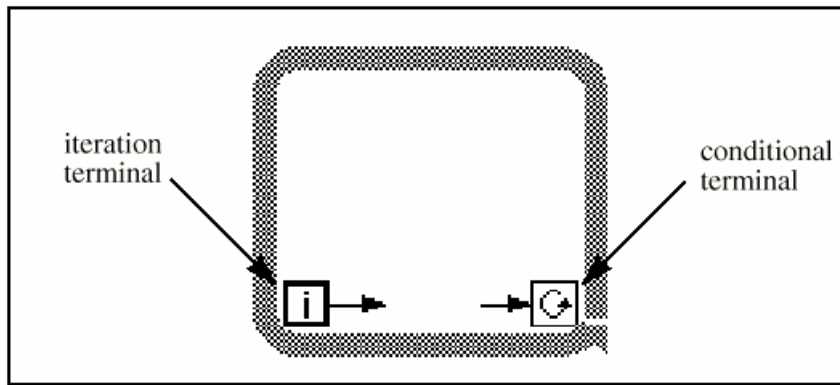


1. Otvoriti blok dijagram

2. Postaviti **While loop** u blok dijagram birajuci je iz **Functions>>Structures**. While kontura je boks koji moze promjeniti velicinu (resizable) koji nije trenutacno spusten na dijagram. Umjesto toga dizajner ima mogucnost da promjeni velicinu i poziciju bloka. Da bi se to uradilo, kliknuti u zonu iznad i na lijevo od svih terminala. Nastaviti drzanje tastera misa, i izvlaciti kvadraticni boks oko terminala. While kontura se nakon toga kreira na specificiranoj lokaciji i velicini.



While kontura, pokazana na slijedejoj ilustraciji, je boks (promjenljive velicine - resizable), kojeg koristite da izvršite dijagram unutar njega, sve dok Bulova vrijednost koja se prenosi preko uvjetnog terminala (conditional terminal - ulazni terminal) je FALSE (netacna). VI ispituje uvjetni terminal na **kraju** svake iteracije, **zbog toga While kontura ce se uvijek izvršiti barem jedanput**. Iteracioni terminal je izlazni numericki terminal koji sadrzi broj puta koliko se kontura vec izvršila. Ipak, brojac iteracija uvijek starta od nule, tako da ako kontura se izvrši jedanput, iteracioni terminal ce na izlazu pokazati 0.



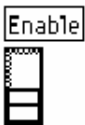
While kontura je ekvivalentna slijedecem pseudo kodu:

Do

Execute Diagram Inside the Loop (ovim se postavlja uslov kruzenja u konturi)
While Condition is TRUE (dok je uslov istinit).



3. Izaberite funkciju slucajnog broja (0-1) iz **Functions>>Numeric**.
4. Ozicite dijagram kako je to pokazano na prvoj slici odjela *Blok Diagram*, spajajuci funkciju slucajnog broja (0 - 1) i preklopnik Enable (omoguci) sa terminalom uslova za While konturu. Ostaviti terminal kasnjenja u konturi za sada neozicenim.
5. Vratiti se na prednji panel i ukljuciti vertikalni prekidac klikom na njega sa operativnim alatom. Izvrsiti VI.

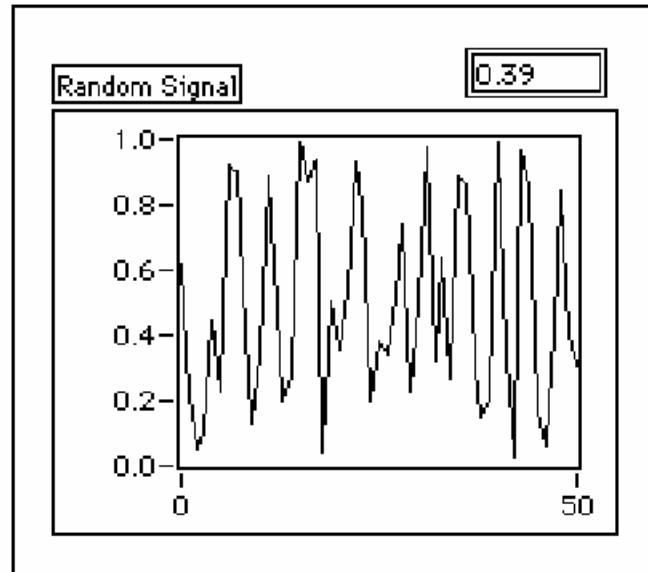


While kontura je beskonacna struktura konture. Dijagram unutar svojih granica izvrsava se sve dok je prekidac ukljucen (TRUE), i dijagram nastavlja da generira slucajne brojeve u prikazuje ih na chartu.

6. Da se zaustavi kontura, kliknite na vertikalni prekidac. Iskljucujuci prekidac (off), salje se vrijednost FALSE ka terminalu uslova konture, i zaustavlja kontura.
7. Chart ima bafer prikaza (dio memorije za privremeno pohranjivanje podataka) koji zadrzava odredjeni broj tacaka i kada su iscezle sa vidljivog dijela ekrana. Chart-u se moze dodati scrollbar (dodatci na bocnim stranama prozora za scrolling - klizanje slike) na taj nacin da se klikne na chart i izabere **Show>>Scrollbar**. Mozete koristiti alat za pozicioniranje da se podesi velicina i pozicija scrollbara.

Da bi se klizalo (skrolovalo) kroz chart, kliknite i zadržite pritisnutim taster misa na bilo kojoj streli scrollbara (gore ili dole).

Da se ocisti bafer prikaza i resetuje chart, kliknite (pop-up) na chart i izaberite **Data Operations>>Clear Chart**.



*NAPOMENA: Default velicina display bafera je 1024 tacke. Mozete povecati ili smanjiti ovu velicinu bafera klikom na chart i birajuci **Chart History Length**.*

DODAVANJE TAJMINGA

Kada izvrsavate VI, While kontura se izvrsava sto je brze moguće. Ponekad, pak zelite uzeti podatke u izvjesnim intervalima, kao naprimjer jedanput u sekundi ili minuti.

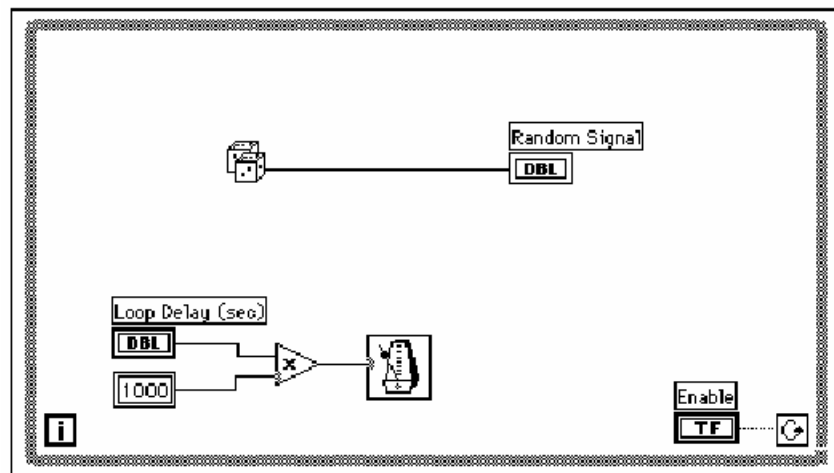
LabVIEW timing funkcije izrazavaju vrijeme u milisekundama (ms), mada Operativni sistem mozda nije u stanju da odrzava ovaj nivo timing tacnosti. Slijedeca lista sadrzi upute za odredjivanje tacnosti LabVIEW timing funkcija na vasem sistemu.

(**OS Windows 3.1**) Timer ima default rezoluciju 55 ms. Mozete konfigurisati LabVIEW da ima 1 ms rezoluciju selektirajuci **Edit>>Preferences** selektirajuci **Performance and disk** iz Paths ringa, i skidajuci chek znak sa **Use default Timer** checkbox.

LabVIEW ne koristi 1 ms rezoluciju kao default jer to predstavlja mnogo vece opterecenje za vas Operativni sistem.

(**OS Windows 95/98/NT**). Timer ima rezoluciju od 1 ms. Medjutim, ovo zavisi od hardwarea, jer na sporijim sistemima 80486, rezolucija timing-a ce biti ispod ove rezolucije

Vi mozete kontrolisati timing konture koristeci **Wait until Next ms Multiple** funkciju (**Functions>>Time & Dialog**). Ova funkcija obezbjedjuje da niti jedna iteracija nije kraca od specificiranog broja milisekundi.



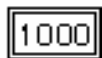
1. Modificirati VI da bi se generisao novi slucajni broj u vremenskom intervalu definiranom dugmetom, kao sto je pokazano na prethodnom dijagramu.



Cekati do slijedece ms Multiple funkcije (**Functions>>Time & Dialog**). U ovom primjeru, vi mnozite terminal dugmeta sa 1000 da bi pretvorili vrijednost na dugmetu izrazenu u sekundama u milisekunde. Koristiti ovu vrijednost kao ulaz u funkciju **Wait Until Next ms Multiple**.



Funkcija mnozenja (Multiply iz **Functions>>Numeric**). U ovom primjeru, funkcija mnozenja mnozi vrijednost na dugmetu sa 1000 da pretvori sekunde u milisekunde.

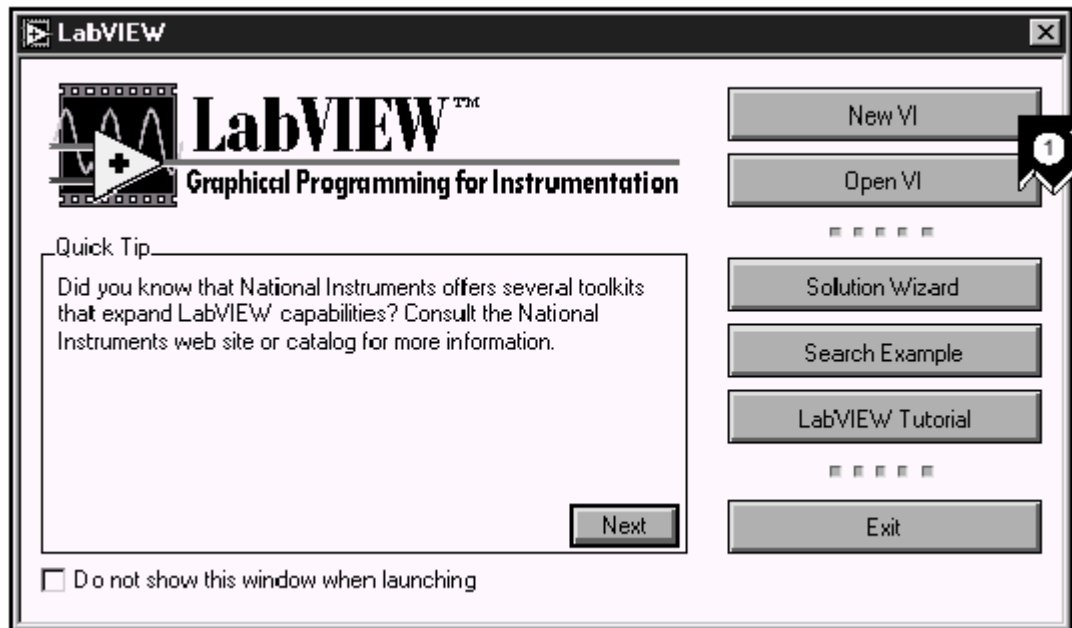


Numericka konstanta (**Functions>>Numeric**). Numericka konstanta sadri konstantu sa kojom morate pomnoziti vrijednost dugmeta da dobijete vrijednost u milisekundama. Tako napr. ako dugme ima vrijednost 1.0, kontura se izvrsava jedanput svakih 1000 ms (jedanput u sekundi).

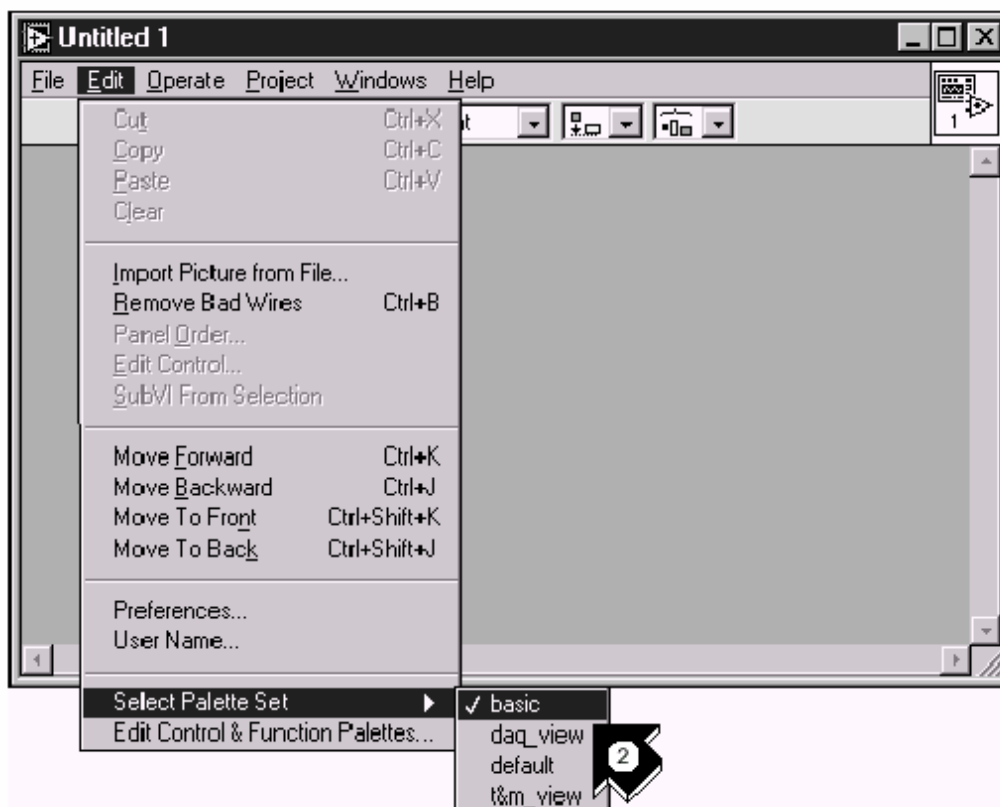
2. Izvrsiti VI. Rotirati dugme da se dobiju razlicite vrijednosti za broj sekundi.
3. Pohraniti i zatvoriti VI u mywork.llb. Nazovite file **My Random Signal.vi**.

DETALJNI OPIS GENERIRANJA VI

Na slijedecem promjeru koji je slican prethodnom prikazacemo detaljniji postupak pri generiranju VI u Labview.

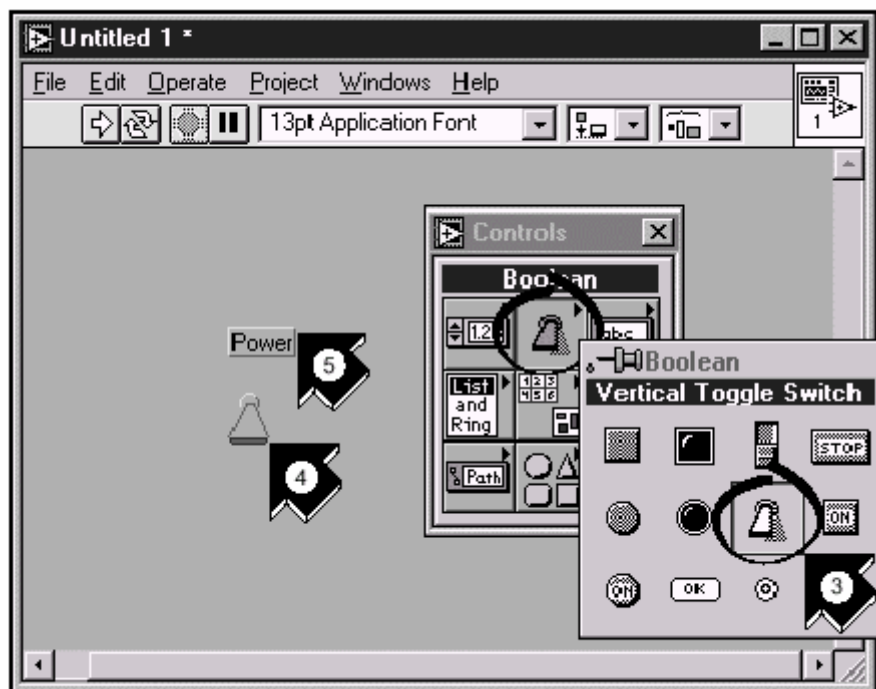


1. Kreirati novu VI selektirajući New VI u Labview dialog boxu kako se to vidi sa gornje slike.



2. Na novom prednjem panelu, izabrati Edit>>Select Palette Set>>Basic

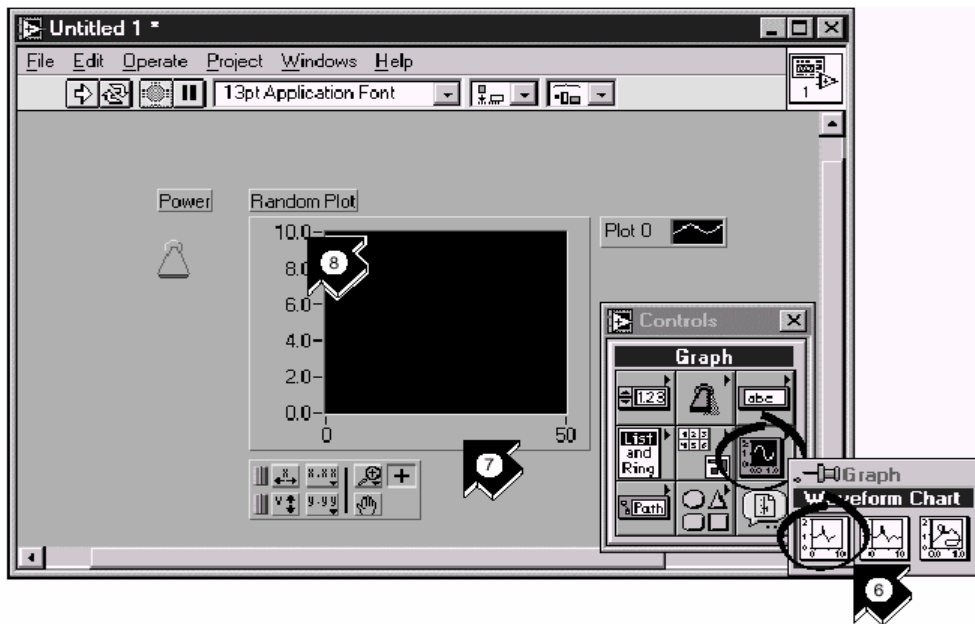
Bazicni set pallete koji je selektiran za ovu aktivnost je mali podskup iz Labview-ovih biblioteka.



- Selektirati **Vertical Toggle switch** iz subpalette **Controls>>Boolean**.
- Pomaci pointer u polje prednjeg panela i kliknuti da postavite preklopni prekidač (toggle switch) na prednji panel.
- Ukucajte **Power** (Napajanje) u labelu za preklopni prekidač. Ako labela iscezne, izabrati **Show label** iz iskacuceg (pop-up) menija na prekidaču. Da bi pristupili ovom meniju, kliknite na desni taster misa.



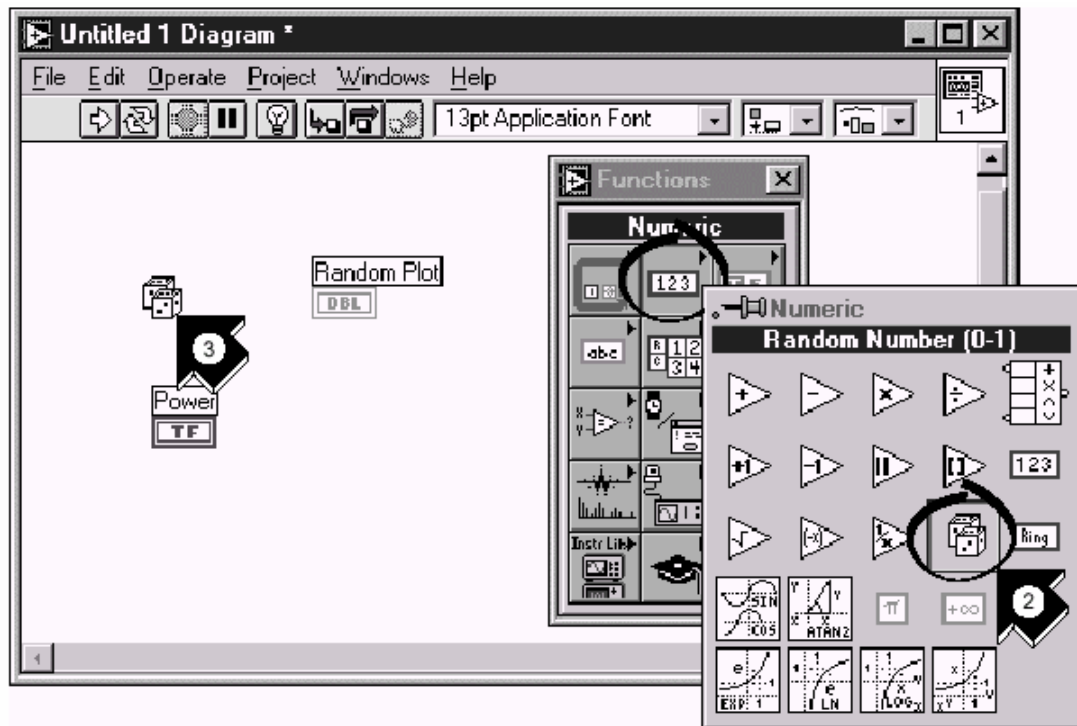
Upustvo (tip). Da reorganizujete ili promjenite velicinu objekata na prednjem panelu ili zica, korisitte **Positioning tool** iz tools palete.



- Kreirajte **waveform chart** selektirajući **Controls>>Graph>>Waveform chart**. Ovaj chart će, kao što smo već rekli, isctavati podatke tacku po tacku.
- Postavite chart na prednji panel i labelirajte ga **Random plot**.
- Da bi promjenili skalu na chartu valnog oblika, selektirajte **Operating tool** iz **tools palette**. Ako tools paleta nije vidljiva, selektirajte **Windows>>Show Tools palette**. Dvaputa kliknite na **10.0** na Y osi indikatora i ukucajte **1.0** da bi promjenili skalu.

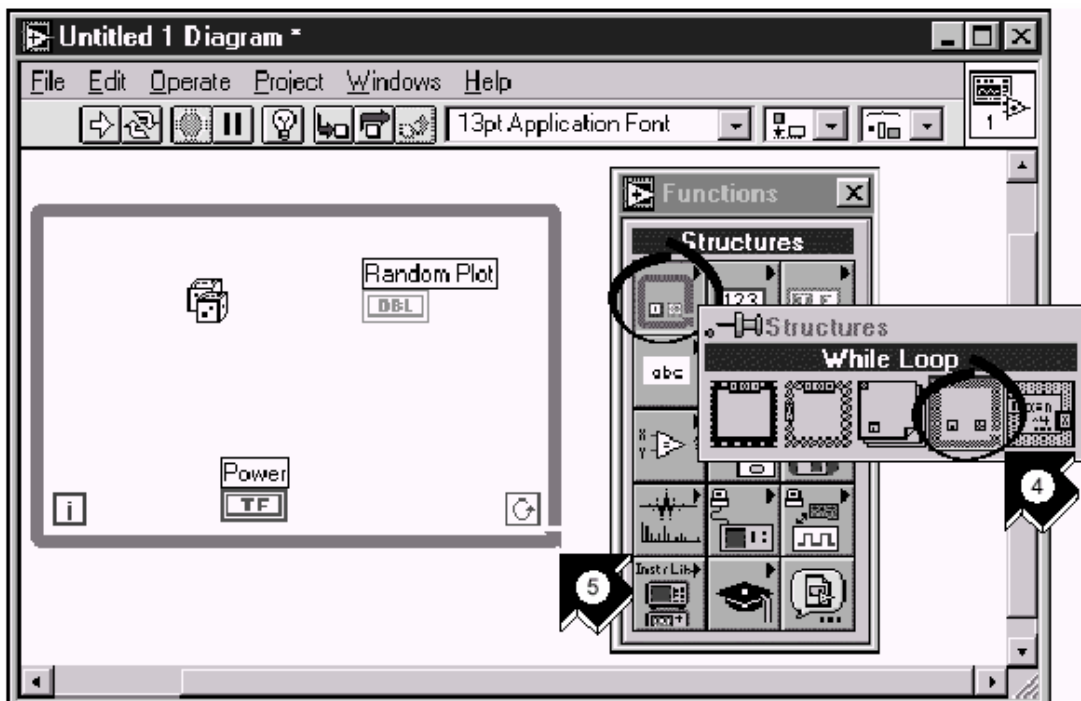
Sada možemo pristupiti drugom dijelu, kreiranju izvornog koda (source code) u blok dijagramu naše VI.

1. Podjimo u blok dijagram izabiruci **Show Diagram** iz Windows menija ili klikajući u prozoru blok dijagrama. Dva terminala na blok dijagramu korespondiraju sa preklopnim prekidačem za napajanje (power toggle switch) i crtezu charta valnog oblika koje smo postavili na prednji panel.



Selektirajte **Random number (0-1)** iz **Functions>>Numeric** subpalete. Ako ova funkcionalna paleta nije vidljiva, selektirajte **Windows>>Show Functions Palette**. Za ovu aktivnost, vi koristite generator slučajnih brojeva da generirate ulazne podatke za vasu VI.

Postavite random number function na blok dijagram. Ova VI će generirati slučajne brojeve u opsegu od 0-1.



4



conditional terminal

Op iz **Functions>>Structures** subpalete. While kontura ce izvorsiti sav kod unutar njenih granica , sve dok je vrijednost na njenom uslovnom terminalu (conditional termina) TRUE (logicki istina). Kada se vrijednost promjeni na FALSE (logicki lazna) , while kontura ce prestati da se izvrsava I program izlazi iz nje.

- Postavite vas pointer u poziciju na blok dijagramu gdje zelite da ankerisete (fiksirate) gornji lijevi ugao bloka konture. Vucite konturu dijagonalno da ukljucite funkciju slucajnog broja, prekidac napajanja I chart slucajnih brojeva.

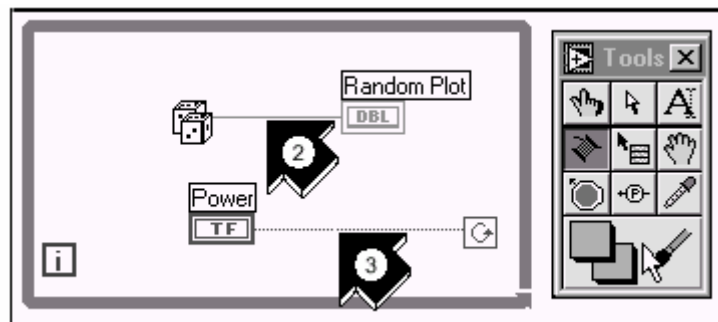
OZICITE I IZVRSITE VASU VI

Mi sada treba da dodamo ozicenje koje ce definisati tok podataka u VI. Nakon sto kompletiramo VI, mi cemo je izvorsiti sa prenjeg panela da vidimo plot u chartu valnog oblika.



Wiring tool

1. Selektirajmo Wiring tool iz palete alata. Ako paleta alata (tools) nije vidljiva, selektirati je sa **Windows>>Show Tools palette**.



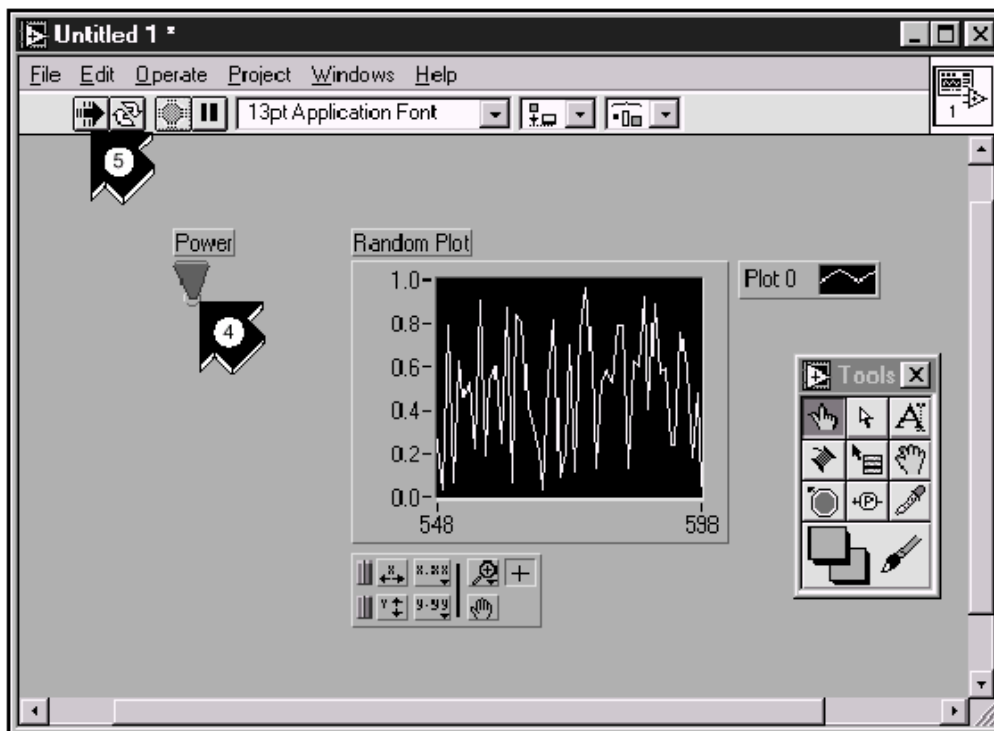
2. Oziciti funkciju slucajnog broja sa terminalom charta. Da bi kreirali zicu, kliknimo na funkciju slucajnog broja, povucimo pointer ka chartu I kliknimo ponovno da zavravimo zicu.

Upustvo (tip) : Kada pozicioniramo alat za ozicenje iznad terminala, terminal blinka I pojavljuje se **tip strip** koji labelira terminal . Kada korektan terminal pocinje da blinka, kliknuti da se ozici do ili od tog terminala. Ako primjetite crnu isprekidanu liniju (losa linija), selektirajte **Edit>>Remove Bad Wires**.



conditional terminal

icite prekidac napajanja sa terminalom uslova while konture.



Operating tool

Kliknite na prekidač napajanja da dodaj u TRUE poziciju.



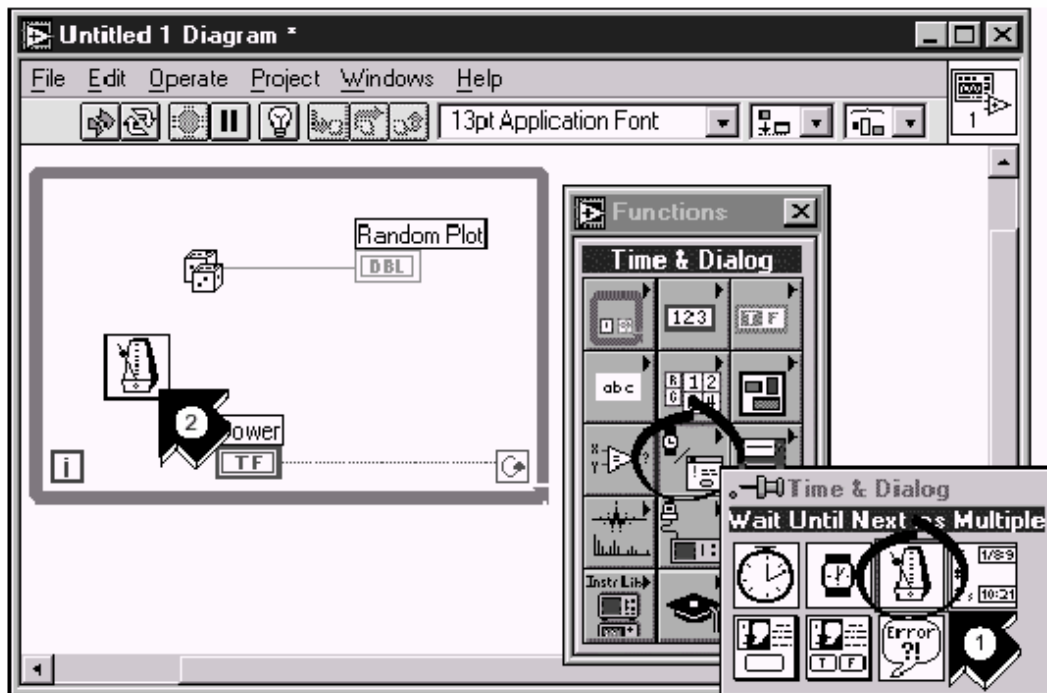
Run

Run taster na toolbaru da se izvrši VI.

6. Da se zaustavi VI, kliknite na prekidač napajanja da predje u FALSE poziciju. Posto se While kontura izvrsava sve dok je ulaz na terminal uslova TRUE, promjena vrijednosti preklopnog prekidača napajanja na FALSE zaustavice izvršenje konture.

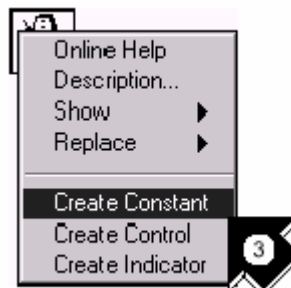
DODAVANJE TIMINGA U VI

Mozemo dodati vremenski definisano kasnjenje u VI da bi se tacke sporije isctavale na plotu charta valnog oblika.

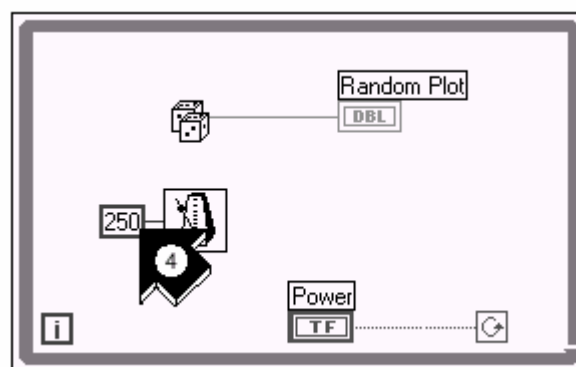


Predjite u blok dijagram. Selektirajte **Wait Until Next ms Multiple** iz subpalette **Functions>>Time & Dialog**.

Postavite funkciju Wait Until Next ms Multiple unutar While konture.



Iz pop-up menija na lijevoj strani funkcije Wait until Next ms Multiple, izabrati **Create Constant**. Kada pristupite pop-up meniju funkcije, pazite da postavite pointer na lijevu stranu funkcije. U suprotnom, konstanta koju kreirate neće biti ožičena sa funkcijom.



Ukucajte 250 u ms multipl kontrolni element da bi dobili 250 ms kasnjenje izmedju generiranja dvaju susjednih tacki u plotu charta.



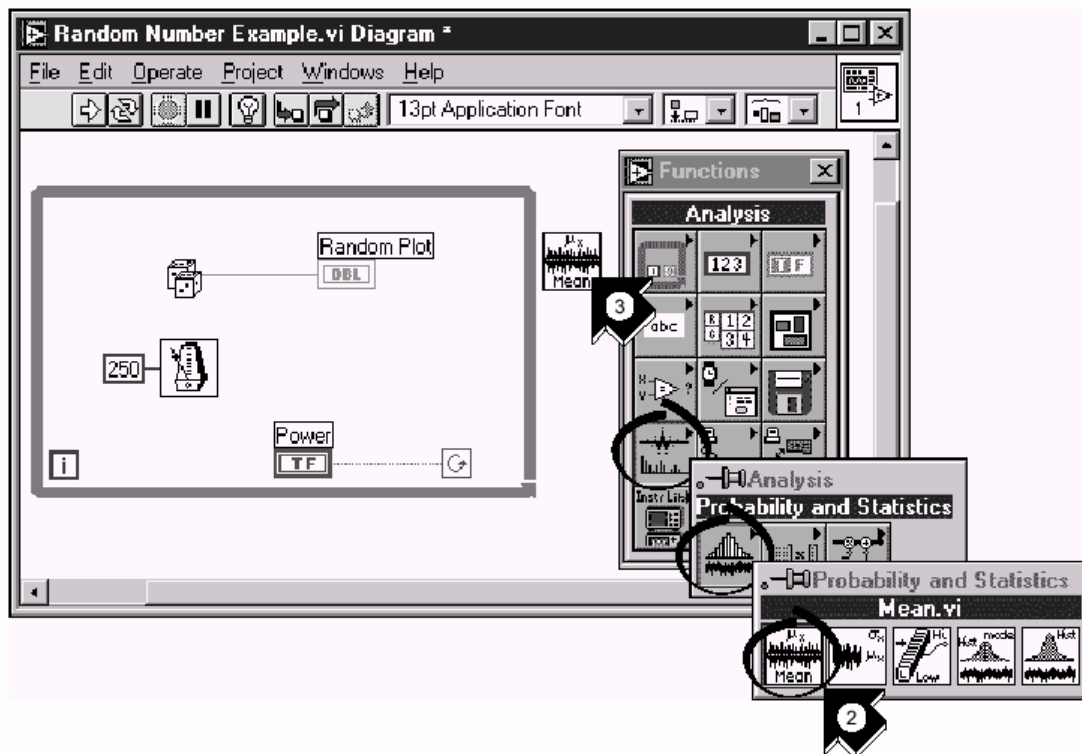
Run

Otidjite na prednji panel, kliknite na prekidač napajanja da predje u TRUE položaj I izvršite VI da vidite efekat kasnjenja. Kliknite zatim ponovo na prekidač napajanja da predje u FALSE položaj I zaustavi izvršenje VI. Pohranite ovaj program kao Random Number example.vi u nekom od vaših subdirektorija.

DODAVANJE ANALIZE I FILE I/O FUNKCIJA

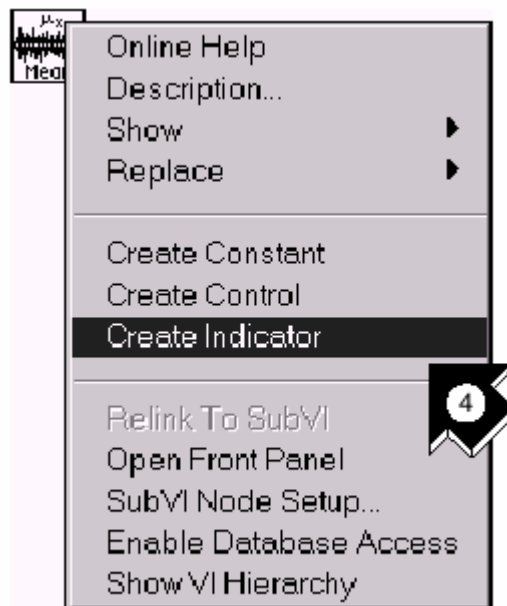
U daljoj razradi nase VI mozemo se odluciti da usrednjimo slucajne vrijednosti koje smo prikupili kao I da ih pohranimo u spreadsheet file.

- Predjimo u prozor blok dijagrama ove nase VI.

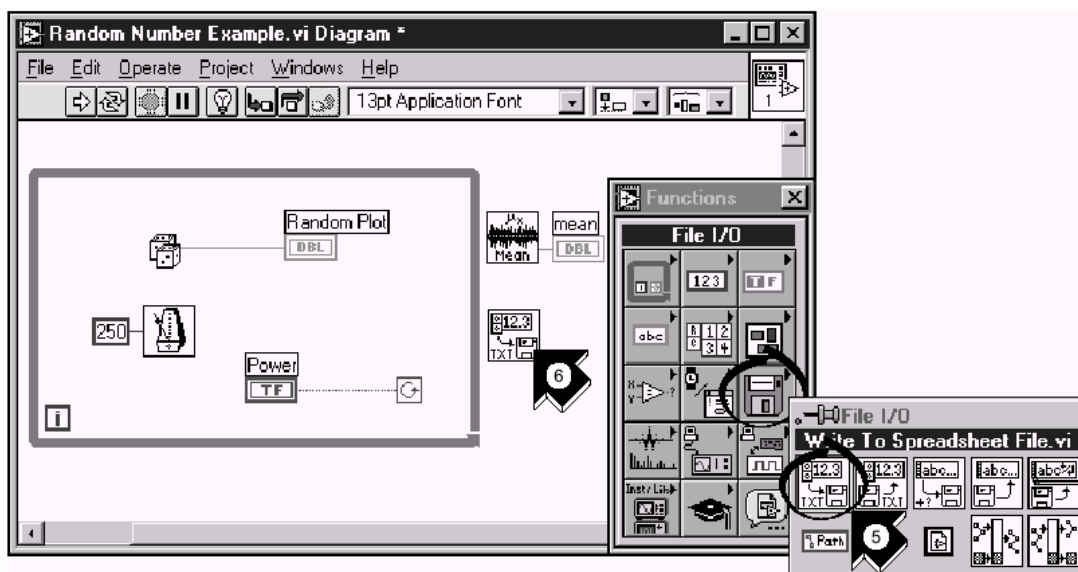


2. Selektirajte Mean.vi iz subpalette **Functions>>Analysis>>Probability and Statistics**.

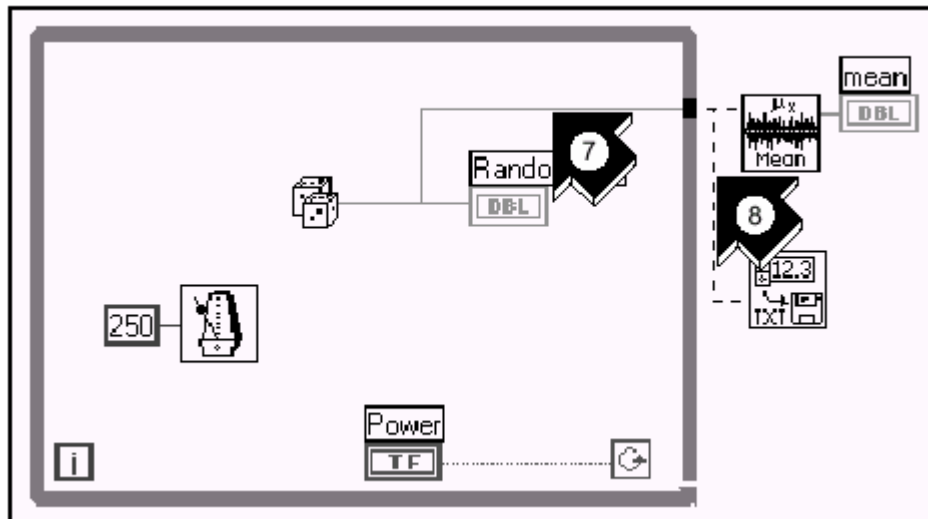
3. Postavite Mean VI u blok dijagram van While konture.



Iz pop-up menija na gornjem desnom uglu Mean VI (VI srednje vrijednosti), izaberite **Create Indicator**. Ovo ce kreirati numericki indikator na prednjem panelu koji ce prikazati srednju vrijednost slucajnih podataka.



- Izaberite **Write to Spreadsheet File.vi** iz **Functions>>File I/O** subpalete.
- Postavite Write to Spreadsheet File VI u blok dijagram izvan While konture.



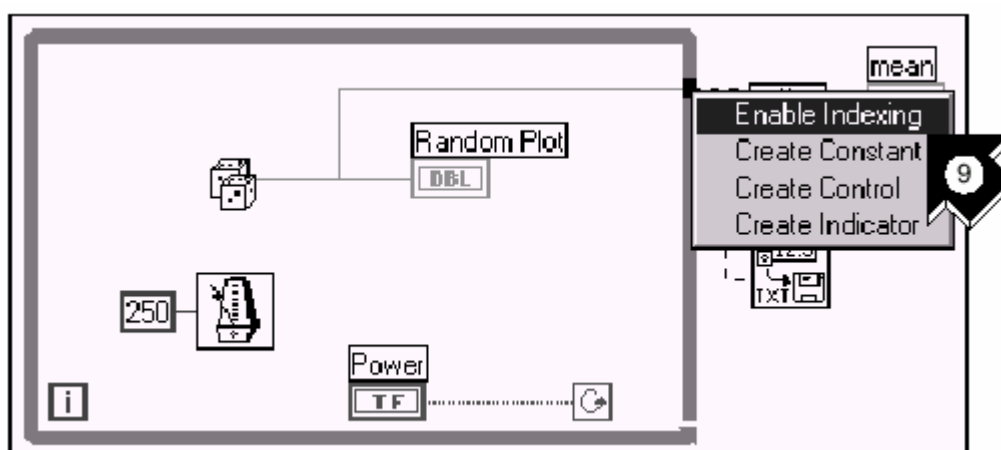
Wiring tool

7. Koristeci alat za ozicenje (Wiring tool), oziciti funkciju slucajnog broja na X ulazni terminal od Mean VI. Kreirajte ovu granu ozicenja izvlaceci je iz postojeceg segmenta zice.

Upustvo :Segment zice blinka kada je alat za ozicenje korektno pozicioniran da ucvrsti novu zicu iz postojeceg segmenta.

Alat za ozicenje vam omogucava da vidite vrh strip labele za terminale na cvorovima u blok dijagramu.

- Kreirajte jos jednu granu zice iz grane kreirane u prethodnom koraku. Ozicite ovu novu granu ka **1D data** ulazu od Write to Spreadsheet File VI. Vi koristite 1D data ulaz zato sto While kontura kreira jedno-dimenzionalni niz podataka iz generiranih slucajnih brojeva.



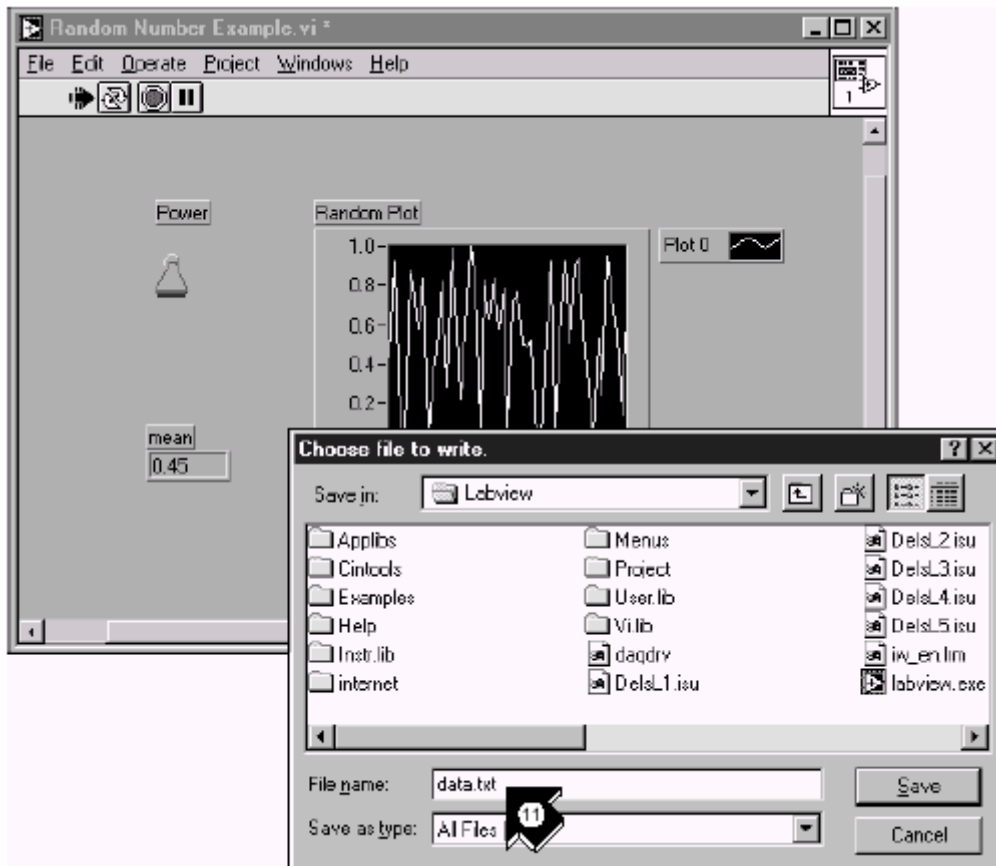
Crni **tunnel** (tunnel) na While konturi je terminal izlaza podataka iz konture. Iz pop-up menija na crnom tunelu, izaberite **Enable Indexing**. Crtkane zice ce se promjeniti u pune narandjaste zice. Enable indexing dozvoljava While konturi do

prikuplja podatke I prenosi ih ka Mean VI kao set podataka, nakon okoncanja izvrsenja konture.



Operating tool

- Predjite na prednji panel. Koristeci Operating alat, kliknite na preklopnik napajanja da predje u TRUE poziciju I izvorsite VI.



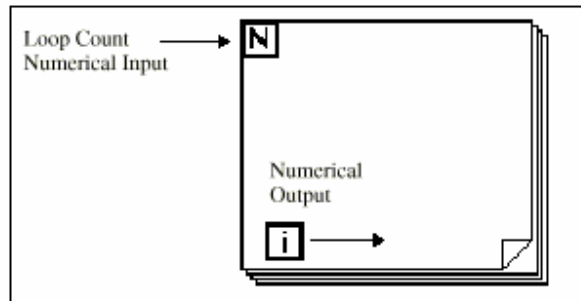
11. Kada iskljucite napajanje, vidjecete srednu vrijednost vasih podataka kao I dijalog boks za file koji ce vas pitati za ime file-a u koji ce pohraniti slucajne vrijednosti. Ukucajte ime napr. Data.txt I kliknite na **Save**.

Upustvo: *Srednja vrijednost se neće pojaviti sve dok traje prikupljanje podataka, kada ste kliknuli na preklopnik napajanja da predje u FALSE položaj.*

12. Koristite bilo koji editor teksta (napr. Notepad or Wordpad) da otvorite data.txt file I pogledate podatke u njemu.

Opaska: *Mozete naci rjesenje ove VI u direktoriju Labview/vi.lab/tutorial.lib/Random Number Example Solution.vi.*

FOR KONTURA



Postaviti For konturu na blok dijagram selektirajući to iz **Functions>>Structures**. For kontura (vidjeti ilustraciju) je boks promjenljive velicine (resizable), kao i While kontura. Kao i While kontura, ne stavlja se odmah na dijagram. Umjesto toga, mala ikona koja predstavlja For konturu se pojavljuje u blok dijagramu, i dizajner ima mogućnost da definiše velicinu i poziciju. Da bi to učinili, prvo kliknite na oblast iznad i lijevo od svih terminala. Dok držite taster misa pritisnut, izvlacite kvadratni okvir oko terminala koje želite postaviti unutar For konture. Kada otpustite dugme misa, LabVIEW kreira For konturu velicine i pozicije koje ste izbrali.

For kontura izvršava dijagram unutar njegovih granica unaprijed određeni broj puta. For kontura ima dva terminala:



Ovo je **count terminal** (ulazni terminal). Ovaj terminal specificira broj puta koliko će kontura biti izvršena.



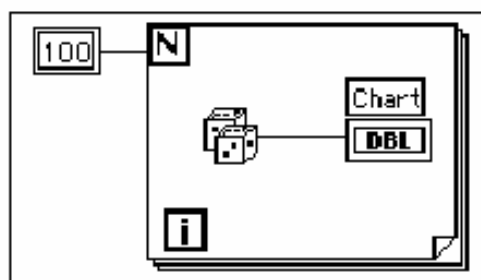
iteration terminal (izlazni terminal). Iteracioni terminal sadrži broj puta koliko je kontura bila izvršena.

For kontura je ekvivalentna slijedecem pseudo-kodu:

For i= 0 to N-1

Execute Diagram inside The Loop (izvrši dijagram unutar konture)

Primjer na slijedećoj slici pokazuje For konturu koja generira 1000 slučajnih brojeva i prikazuje ih kao tačke na chartu.

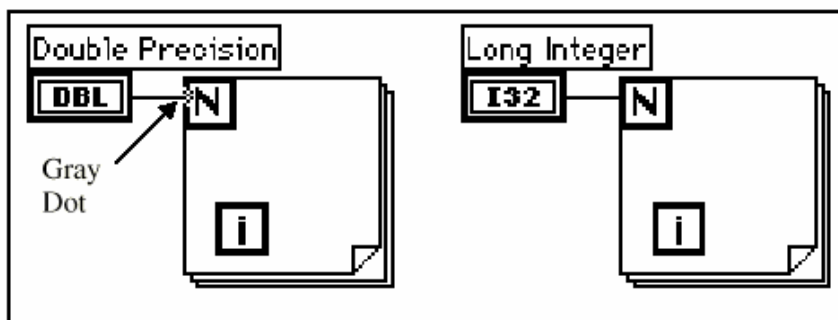


NUMERICKA KONVERZIJA

Do sada, svi kontrolni i indikatorski elementi koje smo koristili bili su sa dvostrukom preciznoscu, brojevi sa pokretnim zarezom (floating point), predstavljeni sa 32 bita. LabVIEW moze pak predstavljati numericke vrijednosti kao integere (byte, word, ili long integer), ili brojeve sa pokretnim zarezom (floating point single, double, ili extended- precision). Default predstavljanje za numericnu vrijednost je double-precision (dvostruka preciznost), pokretni zarez (floating point).

Ako ozicite zajedno dva terminala koji su razlicitih tipova podataka, LabVIEW pretvara jedan od terminala u isti oblik predstavljanja kao i drugi. Kao podsjetnik, LabVIEW ce postaviti sivu tacku koja se naziva koercivna tacka (coercion dot), na terminal gdje se konverzija desava.

- N** Naprimjer, posmatrajmo count terminal for konture. Predstava terminala je kao long integer. Ako ozicimo sa ovim terminalom broj koji je dvostruke preciznosti pokretnog zareza, LabVIEW ce pretvoriti broj u dugi integer. Primjetimo sivu tacku na count terminalu na prvoj For konturi.

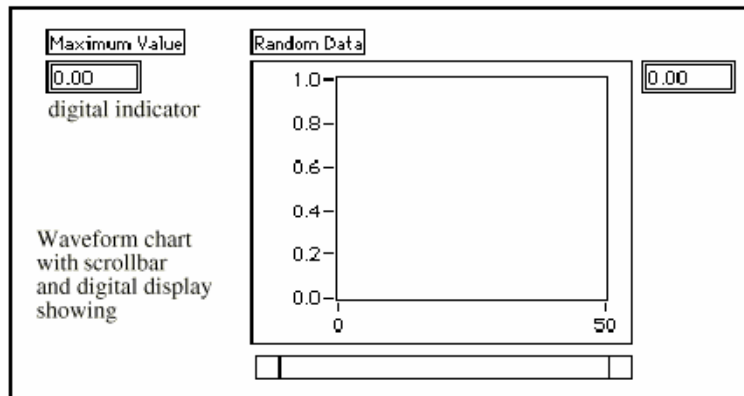


NAPOMENA: Kada VI pretvara floating point brojeve u integere (cjelobrojne), zaokruzava ih na najblizu cijelu vrijednost. Ako je broj tacno na polovini izmedju dva cijela broja, zaokruzuje se na najblizi parni cijeli broj. Naprimjer, VI ce zaokruziti 6.5 na 6, ali 7.5 na 8. Ovo je IEEE standard za ocitavanje cijelih brojeva. (vidjeti IEEE Standard 754 za detalje).

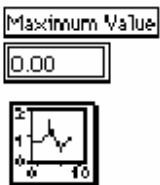
KORISTENJE FOR KONTURE

CILJ Koristiti For konturu i sift registre da se izracuna maksimalna vrijednost u nizu slucajnih brojeva. Vi cete koristiti For konturu (N=100), umjesto While konture.

PREDNJI PANEL

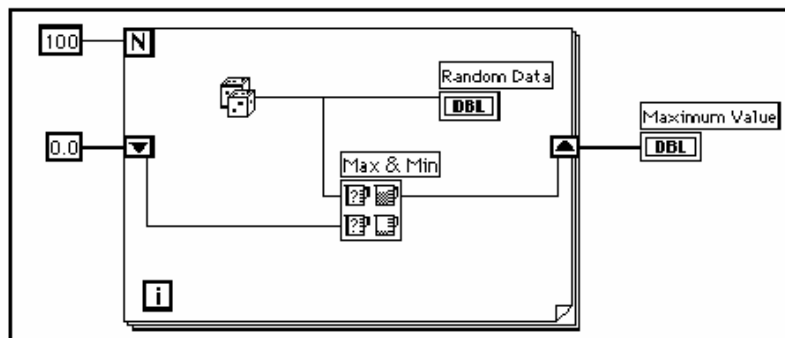


1. Otvoriti novi prednji panel i dodati objekte pokazane na prethodnoj ilustraciji.



- Postaviti digitalni indikator na prednji panel i labelirati ga sa "Maximum Value".
- Postaviti chart valnog oblika na prednji panel i dati mu ime "Random data". Promjeniti skalu charta u opseg 0.0 do 1.0.
- Kliknuti (pop-up) na chart i izabrati Show>>Scrollbar i **Show>>Digital Display**. Pop-up i onemogućiti **Show>>Palette** opciju ako je selektirana.

BLOK DIJAGRAM



- Otvoriti blok dijagram
- Dodati For konturu (Functions>>Structures).
- Dodati shift registar bilo poping-up ili kliknuvsi desnim tasterom misa na lijevu ili desnu ivicu For konture i izabiruci **Add Shift Register**.
- Dodati druge objekte na blok dijagram.



Funkcija slucajnih brojeva (0-1) (**Function>>Numeric**) generise slucajne podatke.



Numericka konstanta (Functions>>**Numeric**). For kontura treba da zna koliko iteracija treba da napravi. U ovom slucaju, izvrsava se For kontura 100 puta.



Numericka konstanta (Functions>>**Numeric**). Vi postavljate pocetnu vrijednost sift registra na nulu za ovaj primjer koji razvijamo, jer znate da je izlaz generatora slucajnih brojeva izmedju 0.0 i 1.0.

Nesto morate znati unaprijed i o podacima koje prikupljate da bi mogli inicijalizirati sift registre. Na primjer, ako inicijalizirate sift registar na 1.0, tada je ta vrijednost vec veca nego sve ocekivane vrijednosti podataka, i bice uvijek maksimalna vrijednost. Ako ne inicijalizirate sift registar tada ce sadrzavati maksimalnu vrijednost prethodnog izvrsenja VI. Zbog toga, mozete dobiti maksimalnu izlaznu vrijednost koja nije u relaciji sa tekucim skupom prikupljenih podataka.



Funkcija Max i Min (Functions>>**Comparison**) uzima od dva numericka ulaza i izlaza maksimalnu vrijednost u gornjem desnom uglu a minimalnu od dvije u donjem desnom uglu. Posto smo mi u ovom primjeru zainteresovani samo za maksimalnu vrijednost, ozicicemo samo maksimalni izlaz i ignorirat cemo minimalni izlaz.

5. Oziciti terminale kako je to pokazano. Ako bi terminal maksimalne vrijednosti bio unutar For konture, vidjeli bi kako se kontinualno azurira, sadrzi samo posljednji izracunati maksimum.

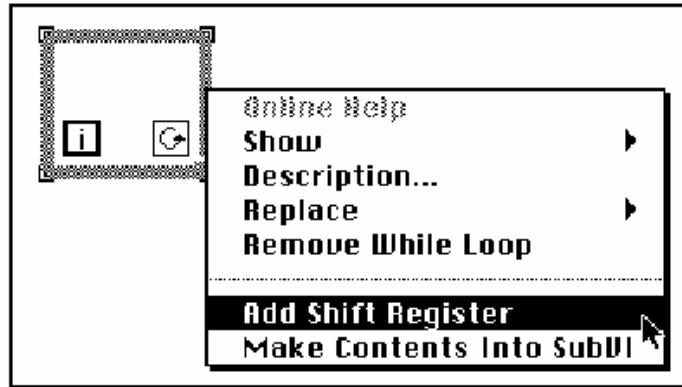
NAPOMENA: Azuriranje indikatora svaki put kada kontura iterira (ponovno izvrsava), moze oduzimati mnogo vremena i treba ga izbjegavati da bi se povecala brzina izvrsenja.

6. Izvrsite VI.

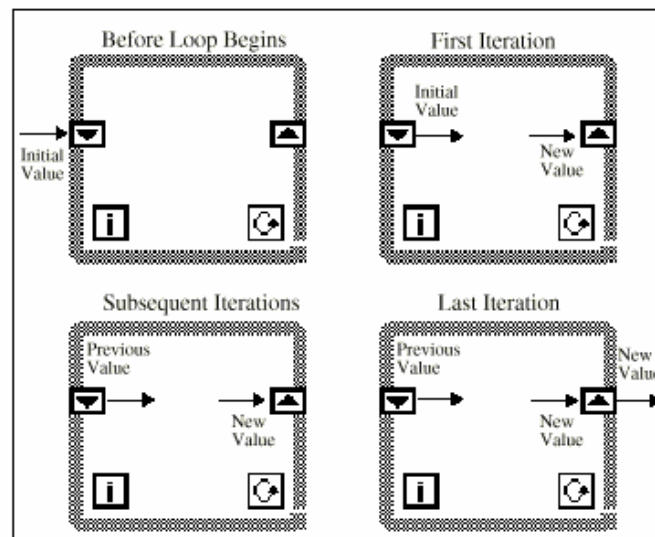
7. Pohranite VI. Dati ime VI **My Calculate Max.vi**

SIFT REGISTRARI

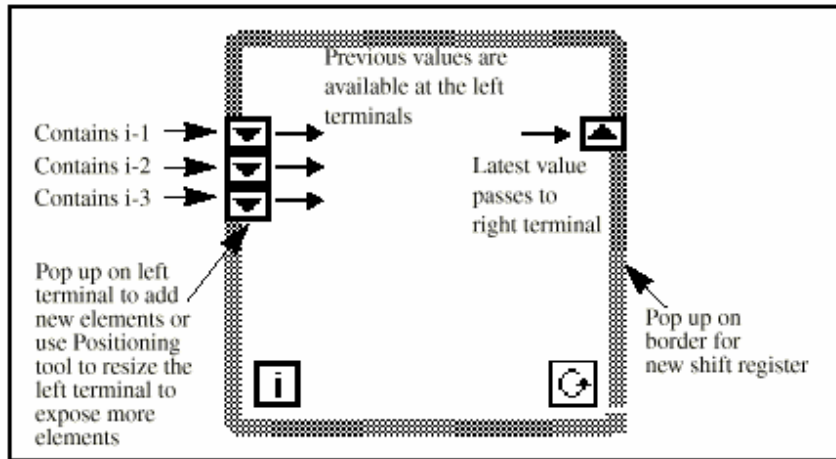
Sift registri (raspolozivi za While i For konture), prenose vrijednosti iz jedne konture iteracije u drugu. Kreirate sift registar klikanjem na lijevu ili desnu ivicu konture i izabiruci **Add Shift Register**



Sift registar sadrzi par terminala direktno suprostavljenih jedan drugome na vertikalnim ivicama granica konture. Desni terminal pohranjuje podatke nakon kompletiranja iteracije. Podatak se pomjera na kraju iteracije i pojavljuje se na lijevom terminalu na pocetku slijedece iteracije (vidjeti slijedecu ilustraciju). Sift registar moze sadrzavati podatak bilo kojeg tipa: numericki, Bulov, string (niz alfanumerickih karaktera), itd. Sift registar se automatski adaptira na tip podatka prvog objekta kojeg ozicite sa sift registrom.



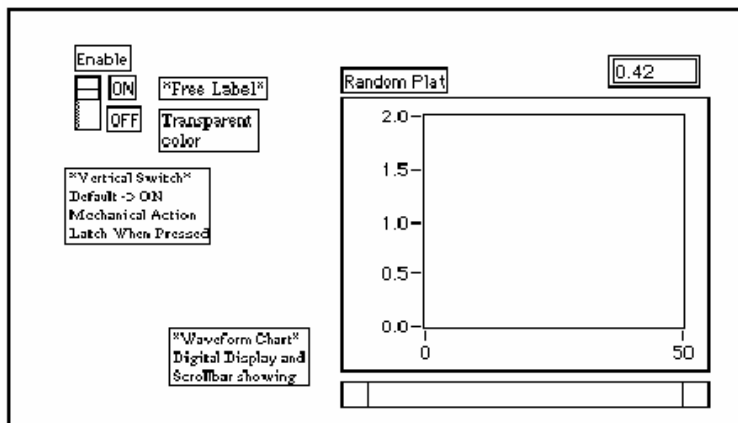
Mozete konfigurirati sift registar da memorira vrijednosti iz nekoliko prethodnih iteracija. Ova osobina je korisna za usrednjavanje tacaka podataka. Vi kreirate dodatne terminale da pristupite vrijednostima iz prethodnih iteracija klikanjem na lijevi ili desni terminal i izbiruci **Add Element**. Naprimjer, ako sift registar sadrzi tri elementa u lijevom terminalu, vi mozete pristupiti vrijednostima posljednje tri iteracije.



KORISTENJE SIFT REGISTARA

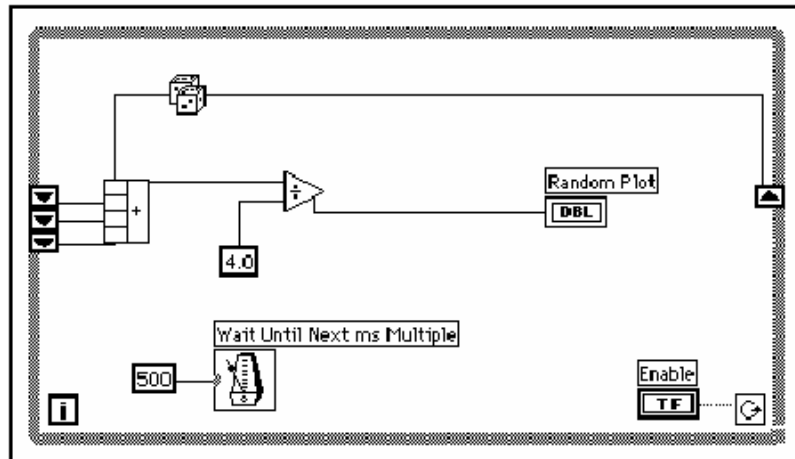
CILJ Izgradicemo VI koja ce prikazati dva slucajna zapisa na chartu. Dva zapisa ce se sastojati od slucajnih zapisa i tekuceg prosjeka posljednje cetiri tacke slucajnog plota.

PREDNJI PANEL



1. Otvoriti novi prednji panel i kreirati prednji panel pokazan na prethodnoj ilustraciji.
2. Nakon sto dodate chart valnog oblika na prednji panel, promjeniti skalu na opseg od 0.0 do 2.0.
3. Nakon dodavanja vertikalnog prekidaoca, kliknite na taster na prednjem panelu i izaberite **Mechanical Action>>Latch When pressed** i postaviti ON stanje kao default izabiruci **Operate>>Make Current Values Default**.

BLOK DIJAGRAM



1. Dodati While konturu (Functions>>**Structures**) u blok dijagram i kreirati sift registar.



Kliknuti na lijevu ili desnu ivicu While konture i izabrati **Add Shift Register**.

Dodati jedan ekstra element klikanjem na lijevi terminal sift registra i birajući **Add Element**. Dodati treci element na isti nacin kao i drugi.

2. Izgraditi Blok dijagram pokazan na prethodnoj slici.



Funkcija slucajnih brojeva (0 - 1) (Functions>>Numeric) generira svjeze podatke.



Ovo je kompaundna (objedinjavajuca) aritmeticka funkcija. (Functions>>**Numeric**). U ovom primjeru, kompaundna aritmeticka funkcija vraca sumu slucajnih brojeva iz dvije iteracije. Da se doda vise ulaza, kliknuti na jedan ulaz i izabrati Add Input iz pop-up menija.



Funkcija dijeljenja (Functions>>**Numeric**). U ovom primjeru, funkcija djeljenja vraca prosjek cetiri posljednja slucajna broja.



Numericka konstanta (Functions>>Numeric). Za vrijeme svake iteracije While konture, funkcija slucajnog broja (0-1) generira jednu slucajnu vrijednost. Vi dodaje ovu vrijednost sa posljednje tri vrijednosti pohranjene u lijevom terminalu sift registra. Funkcija slucajnog broja (0-1) dijeli rezultata sa 4 da bi nasla prosjecnu vrijednost (tekuca + prethodne tri). Prosjek se nakon toga prikazuje na chartu valnog oblika.



Cekati da slijedeca Funkcija multipla ms (**Functions>>Time & Dialog**), obezbjedi da svaka iteracija konture se pojavi ne brze nego milisekundni ulaz. Ulaz je 500 ms za ovaj primjer. Ako kliknete na ikonu i izaberete **Show>>label**, labela **Wait Until next ms Multiple** ce se pojaviti.



3. Pop-up na ulaz funkcije **Wait Until Next ms Multiple** i izaberite **Create Constant**. Pojavice se numericka konstanta i automatski se ozicava sa funkcijom.

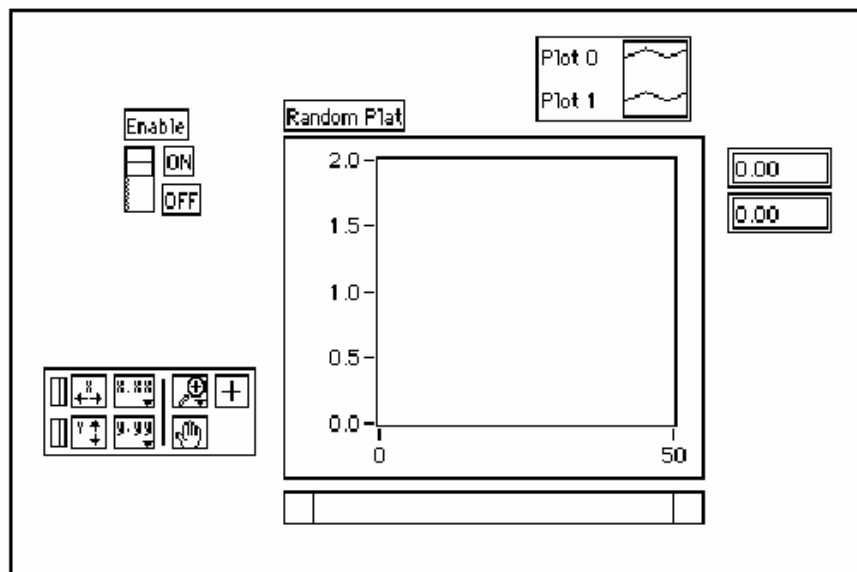
4. Ukucajte 500 u labelu. Numericka konstanta ozicena sa **Wait Until Next ms Multiple** funkcijom specificira da se ceka 500 ms. Dakle kontura se izvrsava jedanput svake pola sekunde.

2. Izvršite VI. VI će pokazati dva zapisa na chartu. Zapisi se preklapaju. Tj. oni imaju istu vertikalnu osu. Pokušajte izvršiti VI sa naglasavanjem izvršenja da bi vidjeli podatke u sift registrima.

Nezaboravite da isključite taster za naglasavanje izvršenja, na toolbaru, kada završite, tako da se VI može izvršavati punom brzinom.

SPECIJALIZACIJA (CUSTOMIZING) CHARTOVA

Možete specijalizirati chartove tako da zadovoljavaju vaše specifične zahtjeve kod prikazivanja podataka ili da prikazu više informacija. Karakteristike koje stoje na raspolaganju uključuju: scrollbar, legendu, paletu, i digitalni displej.



Na chartu, digitalni displej je omogućen. Primjetimo da poseban digitalni displej postoji za svaki zapis na chartu.

1. Ako je scrollbar prisutan, sakrijte ga klikanjem na chart i deselektiranjem **Show>>ScrollBar**.
2. Specijalizirajte (customise) Y osu.





Koristiti alat za labeliranje dvaput kliknuvši na 2.0 u Y skali. Ukucati 1.2 i pritisnuti <Enter>.

Ponovno koristeći alat za labeliranje, kliknite na drugi broj od dna Y ose. Promijeniti ovaj broj na 0.2, 0.5 ili neku drugu vrijednost različitu od tekuće vrijednosti. Ovaj broj određuje numeričko rastojanje podjele na Y osi.



NAPOMENA: Velicina charta ima direktnog efekta na prikaz podjele osa. Povećati veličinu charta ako imate problema da specijalizirate za vaše potrebe ose charta.

3. Pokazati legendu klikom na chart i birajući show>>legend. Pomaknuti legendu na zeljeno mjesto.




-  Vi mozete postaviti legendu bilo gdje relativno u odnosu na chart. Razvucite legendu da ukljuci dva zapisa koristeci kurzor za promjenu velicine (resizing). Alat za pozicioniranje se mjenja u kurzor za promjenu velicine da indicira da mozete promjeniti velicinu legende. Promjeniti 0 u **Current value** (tekuca vrijednost), dvaput kliknuvsi na labelu sa alatom za labeliranje i ukucavanjem novog teksta. Vi takodjer mozete promjeniti zapis 1 u **Running Avg** (tekuci prosjek) na isti nacin. Ako tekst iscezne, povecati tekst boks legende (resizing) mjenjajuci velicinu iz lijevog ugla legende sa kurzorom za (resizing). Mozete postaviti stil linije za crtanje zapisa i stil tacke putem klicanja na zapis u legendi. Mozete postaviti sirinu linije zapisa klicanjem na zapis u legendi. Koristeci ovaj meni, mozete promjeniti default vrijednosti linije na vrijednost koja je veca od 1 piksela. Mozete takodjer izabrati sirinu linije kurzora (koja nije vidljiva ali se moze naci na printerskom ispisu ukoliko ga printer podrzava).

 Ako je vas monitor u boji mozete definisati boje pozadine (background), tragova ili stila tacke, klicanjem sa alatom za boju. Izabrati boju koju zelite iz palete boja.

4. Prikazite paletu charta klicanjem na chart i izbiruci **Show>>Palette**.

  Sa paletom mozete modificirati prikaz charta i za vrijeme dok se VI izvrsava. Mozete resetovati chart, skalirati X i Y osu, i promjeniti format prikaza u svakom trenutku. Mozete takodjer scroll (klizati) da gledate i druge dijelove ili zumirati u zone grafa ili charta. Kao i kod legende, mozete postaviti paletu bilo gdje relativno u odnosu na chart.

5. Izvrsite VI. Dok VI se izvrsava, koristite tastere iz palete da modificirate chart.

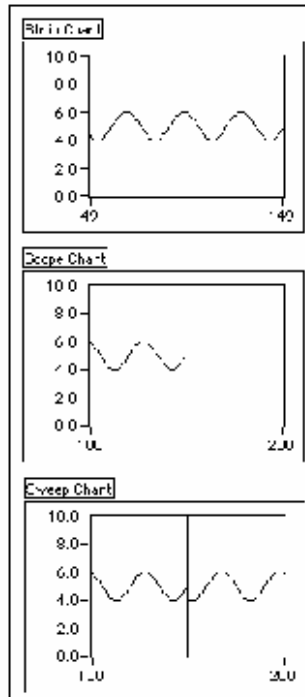
   Mozete koristiti X i Y tastere da reskalirate x i Y ose, respektivno. Ako zelite da graf autoskalira kontinualno bilo koju od dvije ose, kliknite na prekidač zakljucavanja (lock switch), lijevo od svakog tastera da zakljucate na autoskaliranje.

Vi mozete koristiti i druge tastere da modificirate preciznost teksta na osi, ili da kontrolisete operativni mod za chart. Eksperimentirajte sa ovim tasterima da istrazite njihovu funkciju.

NAPOMENA: Modificiranje formata teksta osa cesto zahtjeva vise fizickog prostora nego sto je pocetno bilo odvojeno za osu. Ako promjenite osu, tekst moze postati veci od maksimalne velicine koju valni oblik moze korektno prikazati. Da bi se ovo korigovalo, koristiti kurzor za promjenu velicine da se smanji displaj prostor charta.

RAZLICITI MODOVI CHARTA

Slijedeca ilustracija pokazuje tri opcije prikaza charta koji su na raspolaganju u **Data Operations>>Update Mode**: strip chart, scope chart i sweep chart. Default mode je strip chart. (Ako VI se jos uvijek izvrsava, **Data Operations** submeni je pop-up meni za chart.)



Strip chart mode klizajući prikaz je sličan pisacu sa trakom. Kako VI prima svaku novu vrijednost, iscrtava vrijednost na desnoj margini, i pomjera stare vrijednosti u lijevo.

1. Provjerite da se VI još izvršava, kliknite na chart, i izaberite **Data Operations>>Update Mode>>Scope Chart**.

Scope chart mode ima ponavljajući zapis (retracing display), sličan onome na osciloskopu. Kako VI prima svaku novu vrijednost, iscrtava vrijednost desno od posljednje vrijednosti. Kada plot (zapis) dostigne desnu ivicu zone iscrtavanja, VI briše plot i počinje ponovno iscrtavanje od lijeve ivice. Scope chart je značajno brži nego strip chart jer nije opterećen dodatnim procesiranjem kao kad postoji klizanje (scrolling) charta.

2. Provjeriti da VI se još uvijek izvršava, kliknuti na chart i izabrati **Data>>Operations>>Update Mode>>Sweep Chart**.

Sweep chart mode djeluje na sličan način kao i scope chart, ali ne postaje prazan (blank) kada podaci dodju do desne ivice. Umjesto toga, pokretna vertikalna linija označava početak novog podatka i kreće se preko displeja kako VI dodaje nove podatke.

3. Zaustaviti VI, i pohraniti je. Nazovite je My **Random Average.vi**

POGLAVLJE 5

POLJA (ARRAYS), KLASTERI (CLUSTERS) I GRAFOVI

POLJA (ARRAYS)

Polje se sastoji od skupa elemenata podataka koji su istog tipa. Polje ima jednu ili više dimenzija i do $2^{31} - 1$ elemenata po jednoj dimenziji (ukoliko naravno ima dovoljno memorije). Polja u LabVIEW mogu biti bilo kojeg tipa (izuzev tipa polje, chart ili graf). Pristupa se svakom elementu polja preko njegovog indeksa. Indeks je u opsegu 0 do n-1, gdje n je broj elemenata u polju. Slijedece jednodimenzionalno polje numerickih vrijednosti ilustrira ovu strukturu. Primjetimo da prvi element ima indeks 0, drugi element indeks 1, i tako dalje.

index	0	1	2	3	4	5	6	7	8	9
10-element array	1.2	3.2	8.2	8.0	4.8	5.1	6.0	1.0	2.5	1.7

KONTROLNA POLJA, KONSTANTE I INDIKATORI

Vi kreirate kontrolna polja, konstante i indikatore na prednjem panelu ili na blok dijagramu kombiniranjem skoličke polja (array shell) sa numerickim, Bulovim, stringom ili klasterom (cluster - grozd, svezan). Vec smo rekli da element polja ne moze biti drugo polje , chart ili graf.

GRAF

Graficki indikator se sastoji od dvo-dimenzionalnog prikaza jednog ili vise polja koji se zovu plots (zapisi). LabVIEW ima tri tipa grafova:

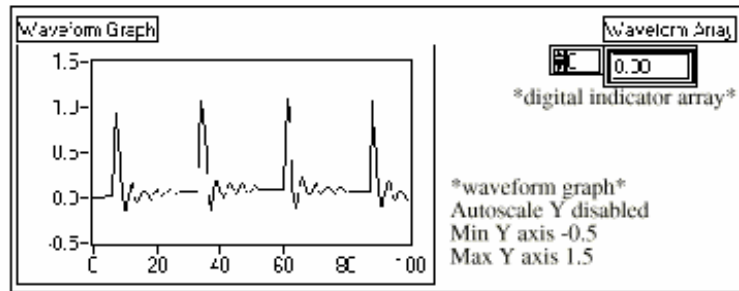
XY grafovi, grafovi valnog oblika (**waveform graphs**) i grafove intenziteta (**intensity graphs**).

Razlika izmedju grafa i charta (vec ranije diskutirana u trecem poglavlju, Konture i Chartovi), je u tome da graf iscrtava podatke kao blok , dok chart iscrtava podatke tacka po tacka ili polje po polje.

KREIRANJE POLJA SA AUTO-INDEKSIRANJEM

CILJ Kreirati polje koristeći auto-indeksiranje **For konture** i iscrtati polje podataka u grafu valnog oblika (waveform graph).

Mi cemo izgraditi VI koja generira polje koristeći **Generate Waveform** VI i iscrtava polje u grafu valnog oblika. Mi cemo takodjer modificirati VI da iscrtava visestrukie ispise (plotove).

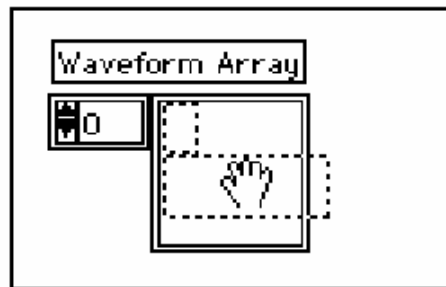


. Otvoriti novi prednji panel.

2. Postaviti skoljku polja (**array shell**) iz **Controls>> Array & Cluster** u prednji panel. Labelirati skoljku polja kao **Waveform array**.

3. Postaviti digitalni indikator iz **Controls>>Numeric** unutar display elementa skoljke polja, kako to pokazuje slijedeca slika. Ovaj indikator prikazuje sadrzaj polja.

1.23



Kako je vec prethodno receno, graf indikator je dvo-dimenzionalni prikaz jednog ili vise polja podataka koji se nazivaju zapisima (plots). LabVIEW ima tri tipa grafova: XY grafovi, grafovi valnog oblika (waveform graphs) i grafovi intenziteta (intensity graphs).

4. Postaviti graf valnog oblika iz **Controls>>Graph** na prednji panel. Labelirati graf **Waveform Graph**.

Graf valnog oblika iscrtava polja sa uniformno (ravnomjerno) rasporedjenim tackama, kao prikupljeni valni oblici koji se menjaju u vremenu.

5. Uvecati graf vucenjem ugla sa kurzorom za promjenu velicine (resizing cursor).

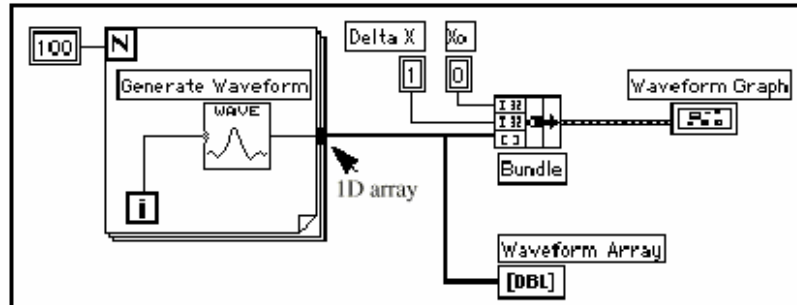
Po defaultu, grafovi autoskaliraju njihove ulaze. To jest, oni automatski podesavaju X i Y granicne vrijednosti skala da bi prikazali cjelokupan skup ulaznih podataka.

6. Onemoguciti autoskaliranje klikanjem na graf i deselektiranjem **Y Scale>>Autoscale Y**.



- Modificirati granice Y ose na taj način što ćete dvaput kliknuti na granice skale sa alatom za labeliranje i unoseći nove brojeve. Promijeniti minimum Y ose na -0.5 i maksimum na 1.5.

BLOK DIJAGRAM



- Izgraditi blok dijagram prikazan na prethodnoj slici.



VI za Generaciju valnog oblika (**Generate Waveform**) može se naći u (**Functions>>Tutorials**), će vratiti jednu tacku valnog oblika. VI zahtjeva skalarni indeksirani ulaz, tako da je potrebno oziciti terminal broja iteracija u konturi sa ovim ulazom. Klikanjem na VI i izborom **Show>>Label** prikazace rijec **Generiraj valni oblik** u labeli.

Primjetimo da zica od **Generate Waveform** VI postaje deblja kako se mijenja u polje na ivici konture.

For kontura automatski akumulira polja na svojoj ivici. Ovo se naziva autoindeksiranje (*auto-indexing*). U ovom slucaju, numericka konstanta ozicena na numericki ulaz za brojac konture For konture, kreira 100 elementno polje (indeksirano 0 - 99).



Funkcija uvezivanja (Bundle function) (u **Functions>>Cluster**) asambliira komponente iscrtavanja (plot) u klaster (svezanj - grozd). Potrebno je da promjenite ikonu Bundle funkcije prije nego što ćete je oziciti na propisan način. Postaviti alat za pozicioniranje na donji desni ugao ikone. Alat se transformise u kurzor za izmjenu velicine (resizing), pokazan na lijevoj strani.



Kada se alat promjeni, kliknuti i vuci sve dotle dok se ne pojavi i treci terminal. Sada mozete nastaviti sa ozicavanjem blok dijagrama kao što je pokazano na prvoj slici ovog Poglavlja.

Klaster se sastoji iz tipa podataka koji može sadržavati elemente podataka različitih tipova. Klaster u blok dijagramu kojeg mi gradimo u ovom primjeru, grupira povezane elemente podataka iz različitih mjesta na dijagramu, reducirajući zapetljavanje i umrsivanje zica. Kada koristite klaster, subVI-i zahtjevaju manje terminala za spajanje. Klaster je inače analogan **record-u** u Pascal programskom jeziku, ili **struct** u C jeziku. Klaster se može zamisliti kao svezanj zica (slično telefonskom kابلu). Svaka zica u kابلu bi predstavljala različit element klastera. Komponente uključuju inicijalnu vrijednost za X (0), delta (inkrementalnu) vrijednost za X (1), kao i Y polje (podaci valnog oblika, koje obezbjedjuju numericke konstante na blok dijagramu). U LabVIEW treba koristiti Bundle funkciju da se asambliira klaster.

NAPOMENA: *Provjeriti da se generisu oni tipovi podataka koje ce prihvatiti grafovi i chartovi.*

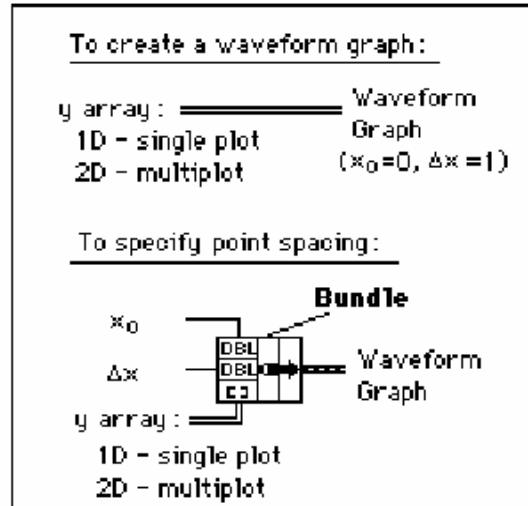
U procesu gradjenja ovog blok dijagrama, provjeriti tipove podataka kroz slijedece korake:



Otvoriti Help prozor birajuci **Help>>Show Help**

Pomjeriti alat za ozicenje preko terminala grafa.

- Provjeriti informaciju o tipu podatka koja se pojavljuje u Help prozoru. Kao primjer pogledati slijedecu sliku:



100

Numericka konstanta (**Functions>>Numeric**). Tri numericke konstante postavljaju slijedece vrijednosti: broj iteracija For konture, pocetnu X vrijednost i inkremment (delta) X vrijednost. Primjetimo da mozemo kliknuti na terminal za brojac For konture , prikazan na lijevoj strani , i izabrati Create Constant da automatski dodamo i ozicimo numericu konstantu za taj terminal.

N

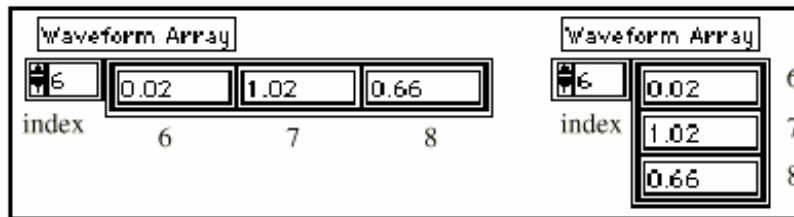
Svaka iteracija For konture generise jednu tacku u valnom obliku koju VI pohranjuje u polje za valni oblik automatski kreirano na ivici konture. Nakon sto kontura okonca svoje izvršenje, Bundle funkcija uvezuje pocetnu vrijednost X (X₀), delta vrijednost za X i polje za crtanje grafa.

2. Vratiti se na prednji panel i izvršiti VI. VI iscrtava autoindeksirano polje valnog oblika na grafu. Pocetna vrijednost X je 0 a delta X vrijednost je 1.
3. Promjeniti delta X vrijednost na 0.5 a pocetnu vrijednost na 20. Ponovno izvršiti VI.

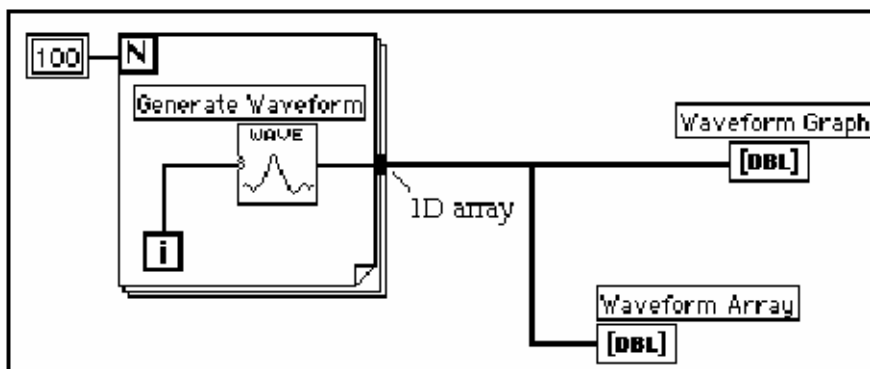
Primjetite da graf sada prikazuje istih 100 tacaka podataka sa pocetnom vrijednoscu od 20 i sa delta X = 0.5 za svaku tacku (pogledati X osu). U testu sa vremenskom kontrolom, ovaj graf ce korespondirati sa 50 sekundi podataka, pocevsi sa 20 sekundi. Eksperimentirajte sa nekoliko kombinacija za pocetne i delta X vrijednosti.

4. Mozete gledati bilo koji element u polju unoseci indeks tog elementa u indeksni displej. Ako unesete broj veci nego sto je velicina polja, displej blijedi (dim) , indicirajuci na taj nacin da niste definisali vrijednost za taj indeks.

Ako zelite da gledate vise od jednog podatka, mozete povecati indikator polja. Postaviti alat za pozicioniranje na donji desni ugao polja. Alat se transformise u kurzor za promjenu velicine (resizing), pokazan na lijevoj strani. Kada se alat promjeni, vucite na desno ili pravo dole. Polje sada prikazuje nekoliko elemenata u rastucem redoslijedu indeksa, pocevsi sa elementom koji korespondira specificiranom indeksu kako to pokazuje slijedeca ilustracija:



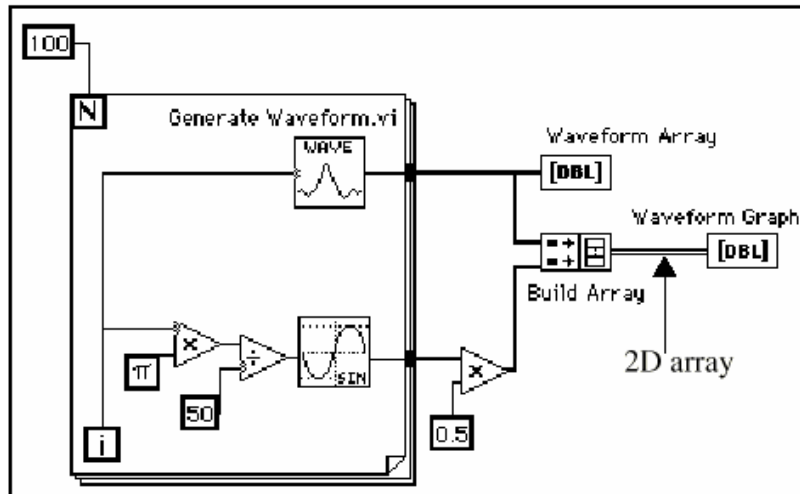
U prethodnom blok dijagramu, vi ste specificirali pocetno X i delta X vrijednost za valni oblik. Cesto, pak, pocetna vrijednost X je nula a delta X vrijednost je 1. U ovim situacijama, vi mozete oziciti polje valnog oblika direktno sa terminalom grafa valnog oblika, kako je to vidljivo na slijedejoj slici:



5. Vratiti se na blok dijagram. Obrisati Bundle funkciju i numericke konstante ozicene sa njom. Da bi se obrisala funkcija i konstante, izabrati funkciju i konstante sa alatom za pozicioniranje a onda pritisnuti <Delete>. Izabrati **Edit>>Remove Bad Wires**. Završiti ozicenje blok dijagrama kako je to pokazano na prethodnoj slici.
6. Izvršiti VI. Primjetite da VI iscrta valni oblik sa pocetnom vrijednoscu $X = 0$ i delta $X = 1$.

VISESTRUKI GRAFOVI

Vi mozete kreirati visestruke grafove valnog oblika gradeci polje od tipa podataka koji se normalno salju u jednostruki graf.



1. Nastaviti graditi blok dijagram kako je pokazano na prethodnom dijagramu.



Funkcija sinusa iz (Functions>>Numeric>>Trigonometric). U ovom primjeru, vi koristite ovu funkciju u For konturi da igradite polje tacaka koje predstavljaju jedan ciklus sinusoide.



Funkcija gradnje polja (**Build array function** u Functions>>Array). U ovom primjeru vi koristite ovu funkciju da kreirate tacnu strukturu podataka da bi isctrali dva polja na grafu valnog oblika, koji je u ovom slucaju dvo-dimenzionalno polje. Povecati funkciju za gradnju polja (Build array) tako da kreirate dva ulaza vucenjem ugla sa alatom za pozicioniranje.



Pi- π (konstanta (Functions>>Numeric>>Additional Numeric Constant).

Podsjetimo se da mozemo naci funkcije mnozenja i djeljenja (Multiply & Divide) u Functions>>Numeric.

2. Prebacite se na prednji panel. Izvrsite VI.

Primjetite da se dva valna oblika isctravaju na istom grafu valnog oblika. Pocetna vrijednost X default-ira na 0 a vrijednost za delta X na 1, za obadva skupa podataka.

NAPOMENA: Mozete promjeniti izgled zapisa (plota) na grafu klikanjem na legendu za specifican zapis. Naprimjer, mozete promjeniti sa linijskog grafa na bar graf, izabiruci **Common Plots>>Bar Graph**.

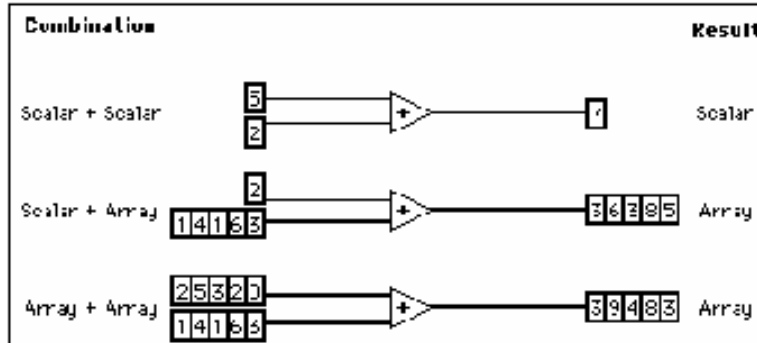
3. Pohranite i zatvorite VI. Nazovite je **My Graph Waveform Arrays.vi**.

Pohranite je u direktorij (folder) **mywork.llb**.

POLIMORFIZAM (VISEOBLIKOVITOST)

Polimorfizam je sposobnost funkcije da se prilagodi ulaznim podacima razlicitih tipova, dimenzija ili nacina predstavljanja. Vecina LabVIEW funkcija su polimorfne. Prethodni blok dijagram je jedan primjer polimorfizma. Primjetimo da koristimo funkciju mnozenja (Multiply) u dvije lokacije, unutar i van For konture.

Unutar For konture, funkcija mnozi dvije skalarne vrijednosti; van For konture, funkcija mnozi polje sa skalarnom vrijednoscu. Slijedeci primjer pokazuje neke od polimorfni kombinacija funkcije sabiranja (add).



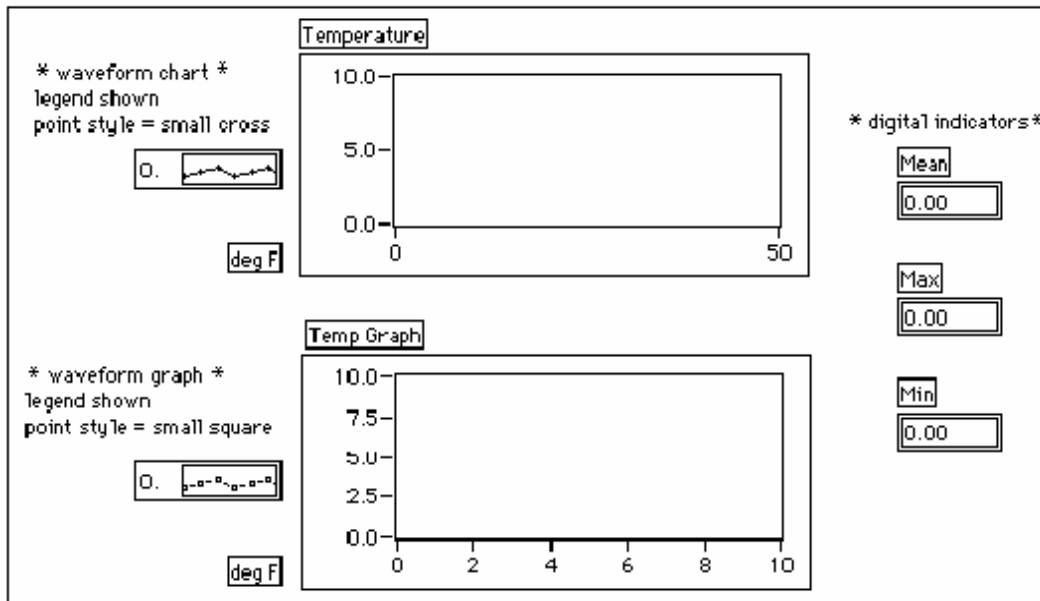
U prvoj kombinaciji, dva skalara se sabiraju, i njihov rezultat je skalar. U drugoj kombinaciji, skalar se dodaje na svaki elemenat polja, i rezultat je polje. U trecoj kombinaciji, svaki element polja se dodaje odgovarajucem elementu drugog polja. Mozete koristiti i druge kombinacije, kao sto su klasteri numerickih vrijednosti, polje klastera, itd.

Ovi principi se mogu primjeniti i na druge LabVIEW funkcije i tipove podataka. LabVIEW funkcije mogu biti polimorficne do razlicitog nivoa. Neke funkcije mogu prihvatiti numericke i Bulove ulaze, druge mogu prihvatiti kombinaciju bilo kojih tipova podataka.

KORISTENJE VI ZA GRAF I ANALIZE

CILJ Izgradicemo VI koja mjeri temperaturu svakih 0.25 sekundi u periodu 10 sekundi. Za vrijeme akvizicije (prikupljanja podataka), VI prikazuje mjerenja u realnom vremenu na strip chartu. Nakon kompletiranja akvizicije, VI iscrta podatke na grafu i izracunava srednju vrijednost, maksimalnu i minimalnu temperaturu.

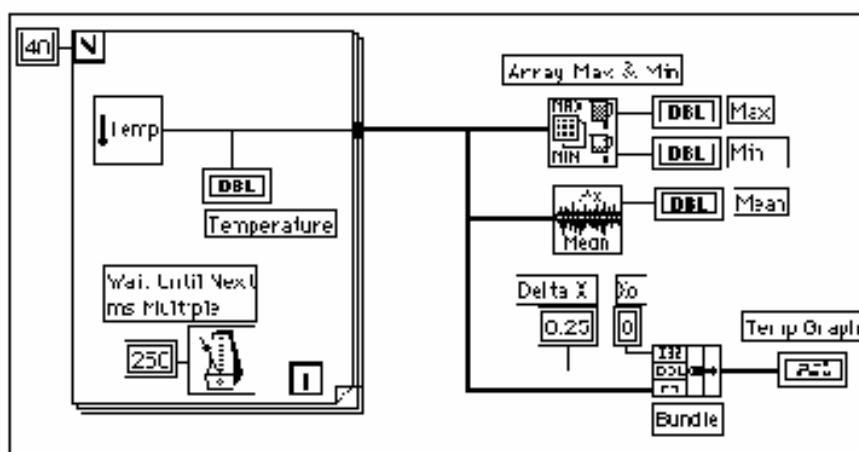
PREDNJI PANEL



1. Otvoriti novi prednji panel i izgraditi prednji panel kakav je prikazan na prethodnoj slici. Mozete modificirati stil tacaka chart i graf valnog oblika, klikanjem na njihove legende.

Chart valnog oblika temperature prikazuje temperaturu onako kako je prikupljena. Nakon prikupljanja (akvizicije), VI iscrtava podatke u temp Grafu. Digitalni indikatori: Srednja vrijednost (Mean), Max i Min prikazuju srednju vrijednost, maksimalnu i minimalnu temperaturu.

BLOK DIJAGRAM



1. Izgraditi blok dijagram pokazan na prethodnoj slici, koristeći slijedeće elemente:



VI Digitalnog termometra (**Functions>>Tutorial**), ili mozete koristiti VI koju smo izgradili u drugom Poglavlju birajuci **Functions>>Select a VI**. i izbiruci **My Thermometer VI**. Ova funkcija nakon izvršenja vraća jedno temperaturno mjerenje.



Cekati dok se ne izvrši funkcija **Wait until next ms multiple function** (**Functions>>Time & Dialog**). U ovom primjeru, ova funkcija obezbjeđuje da se For kontura izvrši svakih .25 sec (250 ms).



Numericka konstanta (**Functions>>Numeric**). Vi takodjer mozete kliknuti na **Wait Until next ms Multiple function** i izabrati **Create Constant** da automatski kreira i ozici numericku konstantu.



Funkcija Maximuma i Minimuma polja (**Functions>>Array**). U ovom primjeru ova funkcija vraća maksimalnu i minimalnu temperaturu izmjerenu za vrijeme akvizicije.



Funkcija srednje vrijednosti (mean) VI (**Functions >>Analysis >>Probability and Statistics**) vraća srednju vrijednost mjerenja temperature.



Bundle funkcija (Funkcija uvezivanja, iz **Functions>>Cluster**) asamblira komponente ispisa (plota) u klaster (grozd, svezan). Komponente uključuju početnu vrijednost za X (0), delta vrijednost za X (0.25), kao i polje Y vrijednosti (temperaturni podaci). Koristiti alat za pozicioniranje da se promjeni velicina funkcije vukuci jedan od uglova.

For kontura se izvrsava 40 puta. Funkcija **Wait Until Next ms Multiple** uzrokuje da svaka iteracija se desava poslije 250 ms. VI pohranjuje temperaturna mjerenja u polje kreirano na ivici For konture (auto-indeksirano). Nakon sto se kompletira izvršenje For konture, polje prenosi podatke razlicitim cvorovima.

Polje Max & Min funkcije vraća maksimalnu i minimalnu temperaturu. Srednja VI vraća prosjek vrijednosti temperaturnih mjerenja.

Kompletna VI ce povezati polje podataka sa pocetnom vrijednoscu X (0) i delta X vrijednoscu (0.25). VI zahtjeva delta X = 0.25 tako da bi mogla da iscrtava tacke temperature svakih 0.25 sekundi na grafu valnog oblika.

2. Vratite se na prednji panel i izvršite VI.
3. Pohranite VI u mywork.llb kao **My Temperature Analysis.vi**.

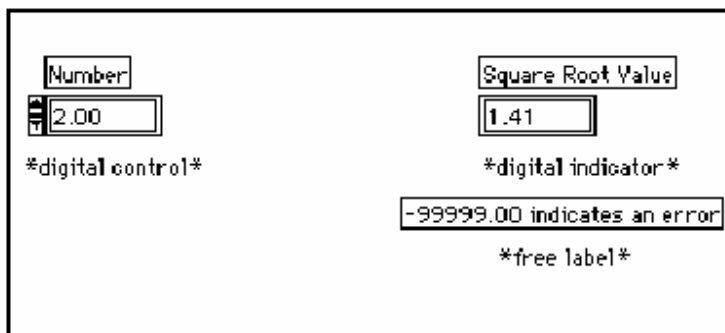
POGLAVLJE 6

SLUCAJ (CASE), STRUKTURA SEKVENCE I CVOR FORMULE

KORISTENJE CASE STRUKTURE

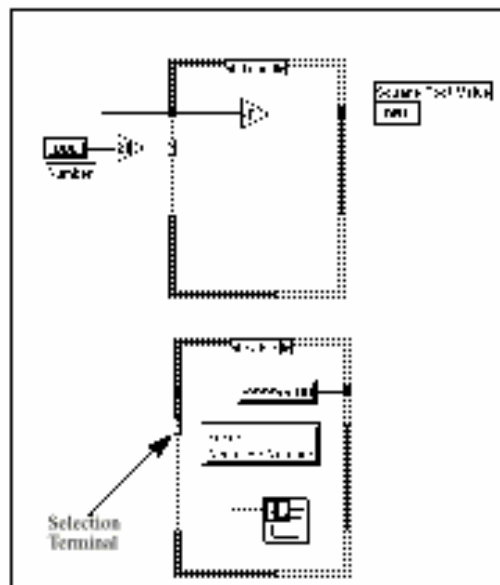
Izgradicemo VI koja provjerava broj da ispita da li je pozitivan. Ako je broj pozitivan VI izracunava kvadratni korjen iz broja, inace, funkcija VI ce vratiti znak greske.

PREDNJI PANEL



1. Otvoriti novi prednji panel i izgraditi prednji panel kakav je prikazan na gornjoj slici.

Kontrolni element Broj obezbjeduje brojcanu vrijednost. Indikator vrijednosti kvadratnog korjena prikazuje vrijednost kvadratnog korjena tog broja. Slobodna labela djeluje kao pogodna lokacija za opasku Korisnika.

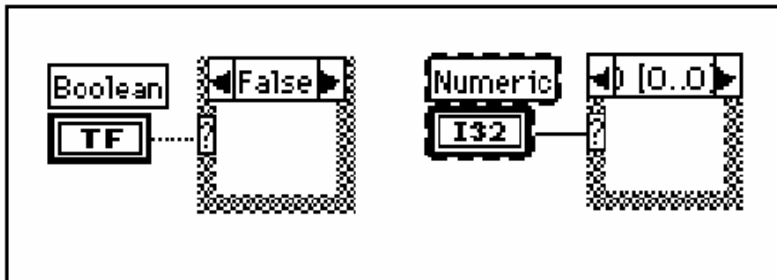




1. Otvorite Blok dijagram

2. Postaviti Case strukturu (**Functions>>Structures**) u blok dijagram. Uvecati Case strukturu vukuci jedan ugao sa kurzorom za promjenu velicine.

Po defaultu, Case struktura je Bulov tip podatka i ima samo dva slucaja (cases True (tacan) i False (lazan)). Bulova Case struktura je analogna sa **if-then-else** iskazom u tekst baziranim programskim jezicima. Automatski se mjenja u numericku vrijednost ako se selekcionni terminal poveze sa numerickim kontrolnim elementom.

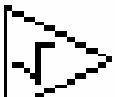


Moguće je prikazati samo jednu case funkciju u svakom trenutku. Da bi se promijenili slucajevi, kliknite na strelice na vrhu Case strukture.

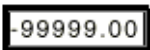
3. Izabrati druge objekte blok dijagrama i ozicite ih kako je to pokazano u ilustraciji blok dijagrama.



Funkcija **Greater or Equal To 0?** (veće ili jednako 0?) iz (**Functions>>Comparison**). U ovom primjeru, funkcija određuje da li je ulazni broj negativan. Funkcija će vratiti TRUE ako je ulaz broja veći ili jednak sa 0.



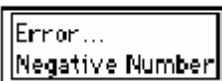
Funkcija kvadratnog korijena (Square root, **Functions>> Numeric**). U ovom primjeru funkcija vraća kvadratni korijen ulaznog broja.



Numericka konstanta. (**Functions>>Numeric**).



Taster Dijalog funkcija (**Functions>>Time & Dialog**). U ovom primjeru, funkcija prikazuje dijalog boks koji sadrži poruku **Error.. Negative number** - greska .. negativan broj).



Konstanta String (Niz) (**Functions>>String**). Unesite tekst unutar boksa sa alatom za labeliranje.

U ovom primjeru, VI izvršava bilo True case (slučaj) ili False case. Ako je broj veći ili jednak 0, VI izvršava True case i vraća kvadratni korijen broja. False case izbacuje na izlaz -99999.00 i prikazuje dijalog boks sa porukom **Error...Negative Number**.

NAPOMENA: Moramo definisati izlazni tunel (output tunnel), za svaki od slucajeva. Kada kreiramo izlazni tunel u jednom slucaju, tuneli ce se pojaviti na istoj poziciji i u svim drugim slucajevima. Neoziceni tuneli sa pojavljuju kao bijeli kvadratni boksovi.

*Provjerite da ozicite sa izlaznim tunelom za svaki neoziceni slucaj (case), klikanjem na sam tunel svaki put. U ovom primjeru, vi doznacujete vrijednost izlaznom tunelu u False case jer True case ima izlazni tunel. Ako ne zelite da doznacite vrijednost izlazu u svim **case-ovima** , morate postaviti indikator na taj case ili koristiti globalnu ili lokalnu varijablu.*

4. Vratiti se na prednji panel i izvršiti VI. Pokusati sa brojem vecim od nule kao i sa brojem manjim od nule mjenjajuci vrijednost u digitalnom kontrolnom elementu kojeg smo labelirali **Number**. Primjetimo da kada mjenjamo vrijednost digitalnog elementa na negativan broj, LabVIEW prikazuje poruku greske, koju smo uspostavili za False case u case strukturi.
5. Pohraniti i zatvoriti VI. Imenovati je My **Square Root.vi**.

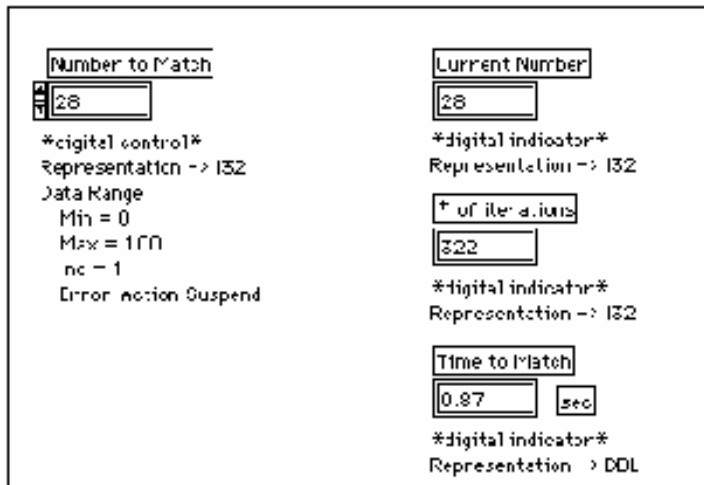
VI LOGIKA

```
if (Number >= 0) then
Square Root Value = SQRT(Number)
else
Square Root Value = -99999.00
Display Message "Error...Negative Number"
end if
```

KORISTENJE STRUKTURE SEKVENCE

Izgradicemo VI koja racuna vrijeme potrebno da se generise slucajni broj koji se podudara sa datim brojem.

Prednji Panel



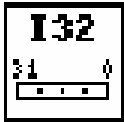
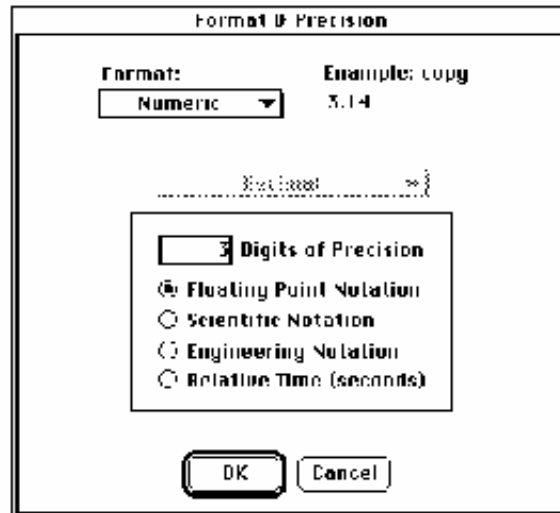
1. Otvoriti novi prednji panel i izgraditi novi prednji panel kakav je pokazan na slijedećoj ilustraciji. Budite sigurni da modificirate kontrolne elemente i inikatore kako je to opisano u tekstu koji slijedi nakon ilustracije.

Kontrolni element **Number to Match** sadrži broj sa kojim zelimo da se upari (match). Indikator tekuceg broja (Current Number), prikazuje tekuci slucajni broj. Indikator sa oznakom **# of iterations**, prikazuje broj iteracija prije uparenja (match). Indikator **Time to Match** pokazuje koliko je sekundi trebalo da se pronadje uparujuci broj.

Modifikacija formata numericke vrijednosti

Po defaultu, LabVIEW prikazuje vrijednosti u numerickim kontrolnim elementima u decimalnoj notaciji sa dva decimalna mjesta (naprimjer 3.14). Vi mozete koristiti **Format & Precision.** opciju pop-up menija indikatora ili kontrolnog elementa da promjenite preciznost prikaza ili da prikazete vrijednosti u tkz. naucnoj ili inzenjerskoj notaciji. Mozete takodjer koristiti **Format & Precision.** opciju da definisete formate vremena i datuma za numericke elemente.

1. Promjeniti preciznost na indikatoru **Time to Match.**
 - a) Kliknuti na **Time to Match** digitalni indikator i **izabrati Format & Precision.** Morate biti u prednjem panelu da bi mogli pristupiti meniju.
 - b) Unjeti vrijednost 3 za **Digits of Precision** i kliknuti na OK.



7. Promjeniti prikazivanje digitalnog kontrolnog elementa i dva digitalna indikatora na **long integer**.

1. Kliknite na digitalni kontrolni element **Number to match** i izaberite **Representation>>Long**.

- b) Ponoviti prethodni korak za **Current Number** kao i za **# of iterations** digitalne indikatore.

Postavljanje opsega podataka (Data range)

Sa **Data Range** opcijom vi mozete sprijeciti korisnika vaseg programa da postavi vrijednost kontrolnog elementa ili indikatora van datog opsega ili inkrementa promjene vrijednosti. Vase opcije su: da ignorise vrijednost van opsega, prisili je da ostane unutar dopustenog opsega, ili da zaustavi izvršenje VI. Simbol greske opsega pojaviće se umjesto tastera za izvršenje, na toolbaru, kada greska opsega zaustavi izvršenje. Također, tamni okvir će oznaciti kontrolni element koji je van opsega.



1. Postaviti opseg podataka između 0 i 100 sa inkrementom 1.

a) Kliknuti na **Time to Match** indikator i izabrati **Data Range**.

b) Unjeti tekst u dijalog boks, kako je pokazano na narednoj ilustraciji i kliknuti na **OK**.

Representation

132
31 0

Long

If Value is Out of Range:
Suspend ▼

Minimum 0

Maximum 100

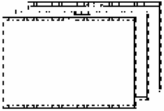
Increment 1

Default 0

Use Default Values

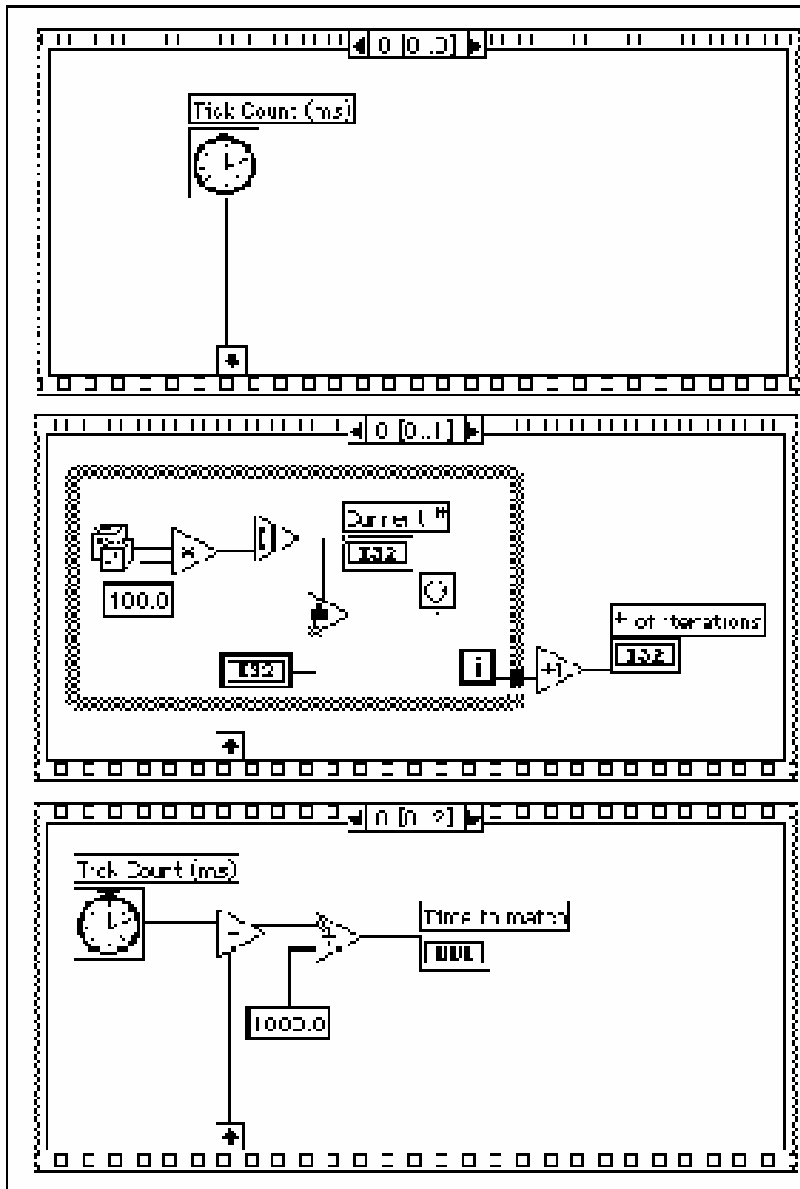
OK Cancel

1. Otvoriti blok dijagram
2. Postaviti Strukturu sekvence (**Sequence structure, Functions>>Structures**) u blok dijagram.



Struktura sekvence, koja izgleda kao okvir filma, izvršava sekvencijalno blok dijagram. U konvencionalnim programskim jezicima, programski iskazi se izvršavaju u redoslijedu u kojem se pojavljuju. U **data flow** programiranju, cvor se izvršava kada su podaci raspoloživi na svim ulazima cvora, mada je zbog toga potrebno izvršiti jedan cvor prije drugoga. LabVIEW koristi sekvencijalnu strukturu kao metod da kontrolise redoslijed u kojem se cvor izvršava.

Blok dijagram

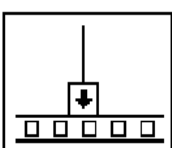


LabVIEW postavlja dijagram kojeg LabVIEW prvog izvrsava unutar granica Okvira 0, zatim postavlja dijagram kojeg izvrsava drugog unutar granica Okvira 1, i tako dalje. Kao i u slucaju Case strukture, samo jedan Okvir je vidljiv u svakom trenutku vremena.

3. Uvecati strukturu vucenjem jednog ugla sa kurzorom za promjenu velicine.

4. Kreirati novi okvir (frame), klikanjem na granici okvira i izabiruci

Add Frame After. Ponoviti ovaj korak da se kreira Okvir 2.



Okvir 0 na prethodnoj slici sadrzi mali boks sa strelom unutar njega. Taj blok je lokalna varijabla sekvence koja prenosi podatke izmedju okvira sekventne strukture. Vi mozete kreirati lokalne sekvence na ivici okvira. Podaci oziceni do okvira lokalne sekvence, ce niti na raspolaganju u narednim okvirima. Medjutim, ne mozete pristupiti podacima u okvirima koji prethode okviru u kojem ste kreirali lokalnu sekvencu.

5. Kreirati lokalnu sekvencu klikanjem na donjoj granici Okvira 0 i birajući **Add Sequence Local**.

Lokalna sekvenca se pojavljuje kao prazni kvadrat. Strelica unutar kvadrata se automatski pojavljuje kada ozicite funkciju do do lokalne sekvence.

6. Završiti blok dijagram kako je to pokazano na prvoj slici u poglavlju o blok dijagramu.

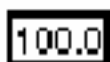


Funkcija brojac otkucaja (Tick count (ms) iz **Functions>>Time & Dialog**). Vraca broj milisekundi koje su prosle nakon ukljucenja napajanja. Za ovaj primjer koji razvijamo trebace nam dvije funkcije **Tick Count**.

Funkcija Slučajnog broja (0 - 1) (**Functions>>Numeric**) . Vraca slučajni broj između 0 i 1.



Funkcija mnozenja (Multiply, iz **Functions>>Numeric**). U ovom primjeru , funkcija mnozi slučajni broj sa 100. Drugim rijecima, funkcija vraca slučajni broj između 0.0 i 100.0.



Funkcija numericke konstante (Functions>>Numeric). U ovom primjeru numericka konstanta predstavlja maksimalni broj koji moze biti pomnozen.



Funkcija **Zaokruzi na najblizu vrijednost** (Round to nearest, iz **Functions>>Numeric**). U ovom primjeru, funkcija zaokružuje slučajni broj između 0 i 100 na najblizi cijeli broj.



Funkcija **Nije jednako?** (Not equal?, iz **Functions>> Comparison**) . U ovom primjeru funkcija poredi slučajni broj sa brojem specificiranim na prednjem panelu i vraca TRUE ako brojevi nisu jednaki. U suprotnom, ova funkcija vraca FALSE.

Funkcija inkrementa (Functions>>Numeric). U ovom primjeru, funkcija povećava brojac While konture za 1.

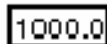


Funkcija oduzimanja (Functions>>Numeric). U ovom primjeru, funkcija vraca vrijeme (u milisekundama), proteklo između Okvira 2 i Okvira 0.



Funkcija djeljenja (Functions>>Numeric).U ovome primjeru, funkcija dijeli broj milisekundi koji je prosao sa 1000 da bi pretvorila broj u sekunde.

Numericka konstanta (**Functions>>Numeric**). U ovome primjeru, funkcija pretvara broj iz milisekundi u sekunde.



U Okviru 0, funkcija Broj otkucaja (Tick Count), vraca tekuće vrijeme u milisekundama. Ova funkcija je ozicena sa lokalnom sekvencom, gdje je vrijednost na raspolaganju u narednim okvirima. U Okviru 1, VI izvrsava While konturu sve dok specificirani broj se ne upari sa slučajnim brojem (0 - 1) koji ta funkcija vrati. U Okviru 2 , funkcija **Tick Count (ms)** vraca novo vrijeme u milisekundama. VI oduzima staro vrijeme (preneseno iz Okvira 0 kroz lokalnu sekvencu) od novog vremena da izracuna isteklo vrijeme

7. Vratiti se na prednji panel i unjeti broj unutar kontrolnog elementa **Number to Match** i izvršiti VI.

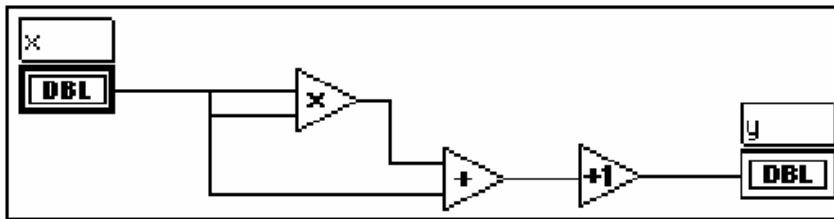
8. Pohraniti VI i zatvoriti je. Nazvati je **My time to match.vi**.

CVOR FORMULE (FORMULA NODE)

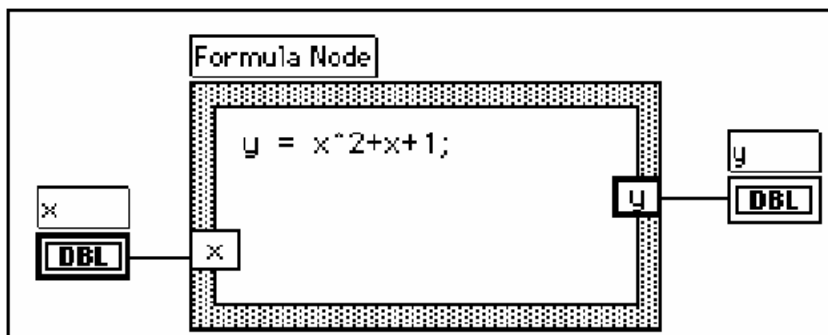
Cvor formule je boks promjenljive velicine koji se moze koristiti da se unese formula direktno u blok dijagram. Postavite cvor formule na blok dijagram izabiruci ga iz **Functions>>Structures**. Ova karakteristika je korisna kada jednačina ima mnogo varijabli ili je na drugi nacin komplikovana. Na primjer, posmatrajmo jednačinu:

$$y = x^2 + x + 1.$$

Ako implementiramo ovu jednačinu koristeći regularne LabVIEW aritmetičke funkcije, blok dijagram izgleda kao na slijedećoj slici :



Mozemo implementirati istu jednačinu koristeći Cvor formule (Formula Node), kako je prikazano na slijedećoj slici:



Sa cvorom formule, mozemo direktno unjeti komplikovanu formulu ili formule, umjesto kreiranja subsekcija blok dijagrama. Formule unosite sa alatom za labeliranje. Kreirajte ulazne i izlazne terminale Cvora formule (Formula node), klikanjem na ivici cvora i izborom **Add Input (Add Output)**. Ukucajte ime varijable u boks. Varijable su osjetljive i prave razliku izmedju velikih i malih slova. Unesite formulu ili formule unutar boksa. Svaki iskaz formule mora se završiti sa polukolonom (;).

Operatori i funkcije koje su na raspolaganju unutar cvora formule su izlistani u Help prozoru za **Formula Node** kako je to prikazano na slijedećoj ilustraciji. Polukolona zaključuje svaku liniju iskaza.

Formula Node operators, lowest precedence first:	
=	assignment
? :	conditional
&&	logical
== != > < >= <=	relational
+ - * / ^	arithmetic
+ - !	unary
Formula Node functions:	
abs acos acosh asin asinh atan atanh ceil	
cos cosh cot csc exp expm1 floor getexp getman	
int intrz ln lnpi log log2 max min mod rand	
rem sec sign sin sinc sinh sqrt tan tanh	

Slijedeci primjer pokazuje kako mozete izvrstiti uslovno doznacavanje unutar cvora formule.

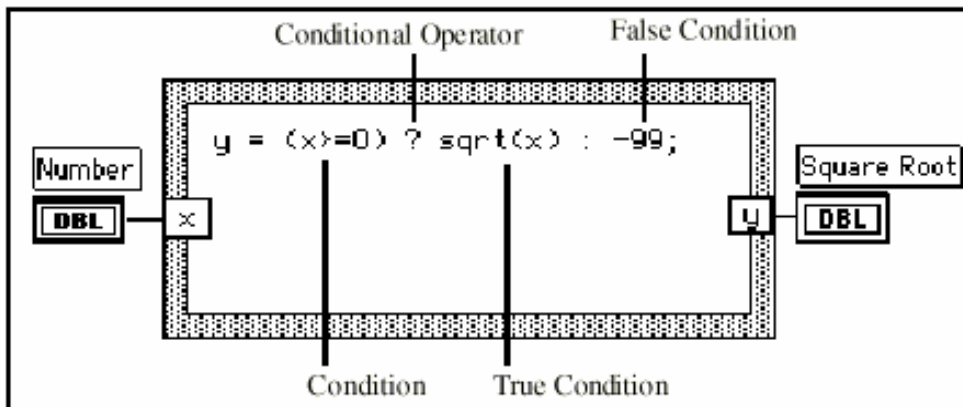
Posmatrajmo slijedeci fragment koda koji izracunava kvadratni korijen od x ako je $x > 0$, i doznacava rezultat varijabli y. Ako je $x < 0$, kod doznacava vrijednost -99 varijabli y.

```

if (x >= 0) then
  y = sqrt(x)
else
  y = -99
end if

```

Vi mozete implementirati fragment koda koristeci cvor formule, kako je to prikazano na slijedecem dijagramu:



KORISTENJE CVORA FORMULE

CILJ Izgradicemo VI koja ce koristiti cvor formule da izracuna slijedece jednacine:

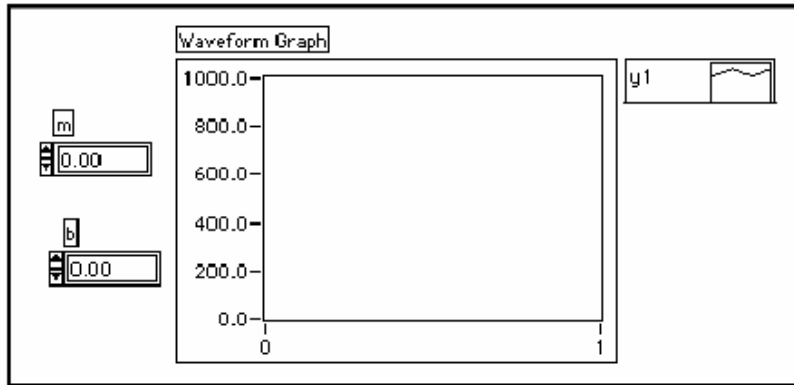
$$y_1 = x^3 - x^2 + 5$$

$$y_2 = m * x + b$$

gdje je opseg x od 0 do 10.

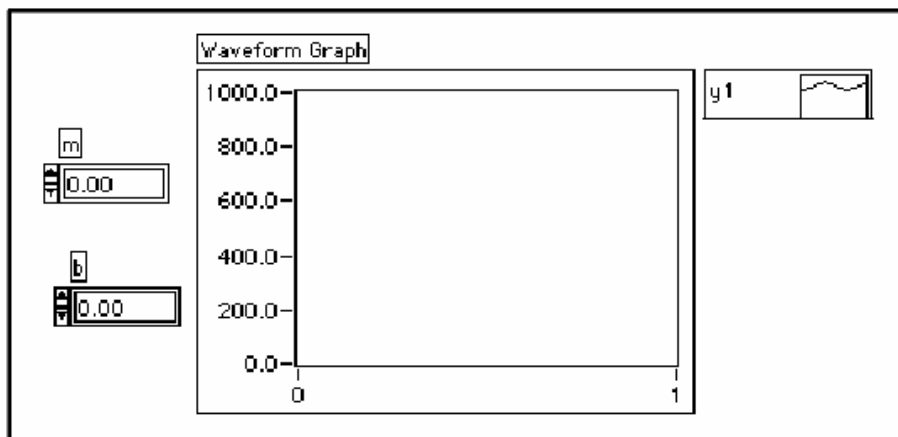
Koristicemo samo jedan cvor formule za obadvije jednacine, i iscrtacemo grafove rezultata na istom grafu.

Prednji Panel



1. Otvoriti novi prednji panel i izgraditi prednji panel pokazan na prethodnoj ilustraciji. Indikator grafa valnog oblika prikazuje zapise jednacina. VI koristi dva digitalna kontrolna elementa da unese vrijednosti za m i b .

Blok dijagram



- . Izgraditi blok dijagram prikazan na prethodnoj ilustraciji.
- !. Postaviti For konturu (Functions>>Structures) u blok dijagram i povuci za uglove da se uveca kontura.





Cvor formule (Functions>>**Structures**). Sa ovim cvorom, mozete direktno unositi formulu (formule). Kreirajte terminale sa tri ulaza klikanjem na ivicu i izborom **Add Input**. Kreiranje izlaznog terminala ostvarujete izborom **Add Output** iz pop-up menija.


Kada kreirate ulazni ili izlazni terminal morate dati ime varijable. Ime varijable mora se upotpunosti podudarati sa onim koje koristite unutar formule. Imena su osjetljiva na velika i mala slova (razlikuju ih). To znaci da ako koristite mala slova

u imenovanju terminala, vi morate koristiti mala slova u imenu varijable unutar formule. Mozete unjeti imena varijabli i samu formulu sa alatom za labeliranje.

Napomena: Mada imena varijabli nisu limitirana u duzini, budite svijesni da dugacka imena uzimaju znacajan dio prostora dijagrama. Polukolona (;) zavrsvava iskaz formule.

 Numericka konstanta (Functions>>Numeric). Vi mozete takodjer kliknuti na terminal brojaca i izabrati **Create Constant** da automatski kreirate i ozicite numericku konstantu. Numericka konstanta specificira broj iteracija For konture. Ako je opseg x od 0 do 10 ukjucujuci 10, treba da ozicite 11sa terminalom brojaca.

 Posto iteracioni terminal broji od 0 do 10, koristite ga da kontrolisete vrijednost X u cvoru formule.

 Blok za gradnju polja (Build Array (Functions>>Array)), postavlja dva ulaza u polje u obliku visezapisnog grafa (multiplot graph). Kreirajte dva ulazna terminala koristeći kurzor za promjenu velicine, vukuci za jedan od uglova.

3. Vratiti se na prednji panel i izvršiti VI sa razlicitim vrijednostima za m i b .
4. Pohraniti i zatvoriti VI. Imenujte VI ***My equations.vi***.


POGLAVLJE 7

STRINGOVI I FILE I/O

STRINGOVI

String (niz karaktera), je skup ASCII karaktera. Mozete koristiti stringove za mnogo vise uloga nego samo jednostavne tekstualne poruke. Kod formiranja instrumentalnih VI-eva, mozemo ih koristiti da prenesemo numericke podatke kao niz karaktera a zatim konvertovati ove nizove u brojeve. Pohranjivanje numerickih podataka na disk moze takodjer ukljucivati stringove. Da bi se pohranili brojevi u ASCII file, moramo prvo konvertovati brojeve u stringove prije nego sto zapisemo brojeve u file na disku.

Kreiranje string kontrolnih i indikacionih elemenata

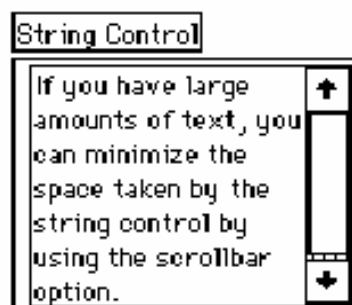
- C  olne i indikacione elemente za string, prikazane na lijevoj strani, mozemo naci u **Controls>>String & Table**. Mozemo unjeti ili promjeniti tekst unutar string kontrolnog elementa koristeći Operativni ili alat za labeliranje.



Povecati string kontrolne elemente i indikatore vukuci uglove sa alatom za pozicioniranje.

STRINGOVI I FILE I/O

Ako zelite da minimizirate prostor koji zauzima string kontrolni ili indikacioni element na prednjem panelu, izaberite **Show>>Scrollbar**. Ako je ova opcija zasjenjena (dimmed, tj. nedostupna), morate povecati vertikalnu dimenziju prozora da je ucinite raspolozivom.

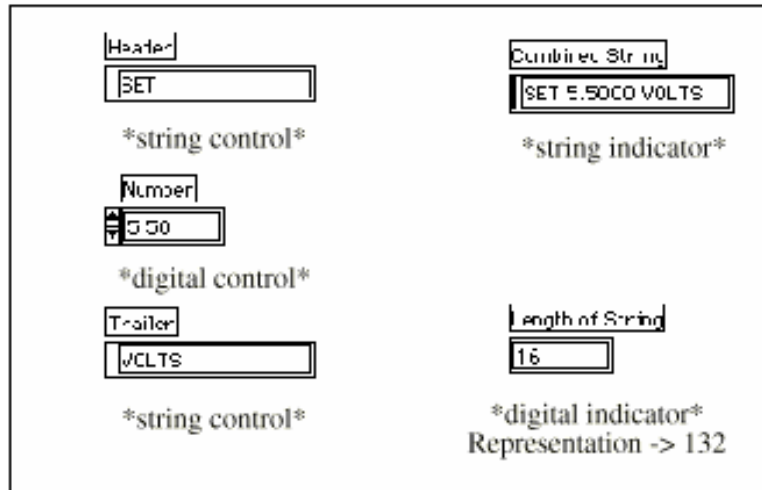


KORISTENJE STRING FUNKCIJA

CILJ LabVIEW ima mnogo funkcija da manipulira stringovima. Naci cete ove funkcije u **Functions>>String**. Izgradicemo VI koja ce konvertovati broj u string i

spajati jedan string sa drugim (concatenate), da se formira jedinstveni izlazni string. VI ce takodjer odrediti duzinu izlaznog stringa.

Prednji Panel

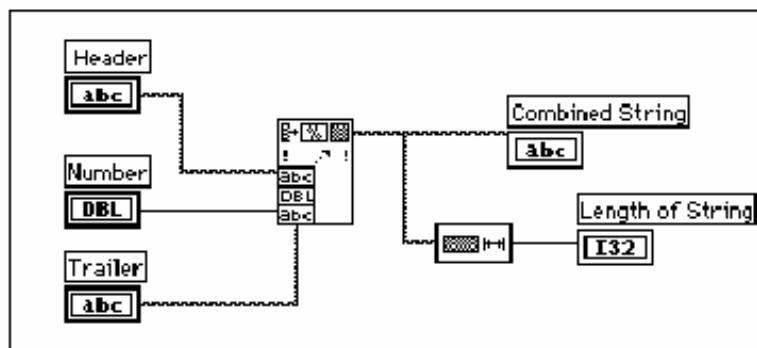


1. Otvoriti novi prednji panel i izgraditi prednji panel pokazan na prethodnoj ilustraciji. Budite sigurni da modificirate kontrolne i indikativne elemente kako je to prikazano na ilustraciji.

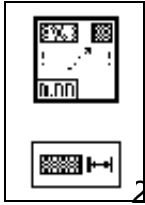
Dva kontrolna elementa za string i digitalni kontrolni element mogu biti kombinirani u jedinstveni izlazni string i prikazani u indikatoru stringa. Digitalni indikator prikazuje duzinu stringa.

Izlaz kombiniranog stringa u ovome primjeru ima slican format sa kombiniranim stringovima koji se koriste kod GPIB (IEEE 488) i serijskih (RS-232 ili RS-422) instrumentata. Pogledati kasnije u poglavlju 8 Akvizicija **podataka i kontrola instrumentata**, da saznate vise o stringovima koji se koriste za komande instrumentima.

Blok Dijagram



1. Izgraditi blok dijagram pokazan na prethodnoj ilustraciji.



Funkcija **Formatiraj u string (Functions>>String)** spaja (concatenate) i formatira članove stringa i stringove i jedinstveni izlazni string. Koristite kurzor za promjenu velicine na ikoni da dodate tri ulaza za argumente.

Funkcija **duzine stringa (Functions>>String)** vraća broj karaktera u spojenom stringu.

2. Izvršite VI. Primjetimo da funkcija **Format Into String** povezuje dva string kontrolna elementa u jedinstveni izlazni string.
3. Pohraniti VI kao **My Build String.vi**. Vi ćete koristiti ovu VI u narednom primjeru.

File I/O

LabVIEW file I/O funkcije (**Functions>>File I/O**) su snazan i fleksibilan set alata za rad sa file-ovima. Osim citanja i pisanja podataka, LabVIEW file I/O funkcije pomjeraju i mjenjaju imena file-ova i direktorija (foldera), kreiraju spreadsheet (tabelarne) tipove file-ova citljivog ASCII teksta, i zapisuju podatke u binarnoj formi radi brzine i kompaktnosti.

Mozete pohraniti i pronaci i učitati podatke iz file-ova u tri različita formata.

- ASCII byte stream (ASCII niz byte-ova). Trebate pohraniti podatke u ASCII formatu kada im želite pristupiti iz drugog softwareskog paketa, kao npr. Word procesora ili nekog spreadsheet programa. Da bi pohranili podatke na ovaj način, morate pretvoriti sve podatke u ASCII stringove.
 - Datalog file-ovi. Ovi file-ovi su u binarnom formatu kojima samo LabVIEW može pristupiti. Datalog file-ovi su slični database file-ovima, jer možete pohraniti nekoliko različitih tipova podataka u jedan (log) zapis u file-u.
 - Binarni byte stream. Ovi file-ovi su najkompaktniji i najbrzi metod pohranjivanja podataka. Morate pretvoriti podatke u binarni string format i morate znati tačno koje tipove podataka koristite da pohranite i povratite podatke u i iz file-ova.
- Ovo poglavlje će analizirati ASCII byte stream file-ove jer je to najčešći file format za podatke.

File I/O funkcije

Najveći broj file I/O operacija uključuje tri osnovna koraka: otvaranje postojećeg file-a ili kreiranje novog file-a; pisanje u ili citanje iz file-a; i zatvaranje file-a. Zbog toga, LabVIEW sadrži mnoge korisne VI-eve u **Functions>>File I/O**. Ovo Poglavlje opisuje devet visoko-nivovskih korisnih alata (utilities). Ove korisne funkcije su izgrađene nad VI-evima srednjeg nivoa koje inkorporiraju provjeru gresaka kao i funkcije za file I/O (ulazno-izlazne manipulacije).

Mozemo također postaviti razdjelnicu (delimiter) ili string razdjelnicu, kao što su tab-ovi (razdjelnica fiksne duzine najčešće 4 prazna polja), zarezi itd. u spreadsheet-u. Ovo nam uštedjuje napor razdjeljivanja (parsing) spreadsheet-a ako smo koristili razdjelnicu različitu od tab-a, da uspostavimo polja u spreadsheet-u.



VI **Upisi karaktere u file** (Write Characters to File), upisuje string karaktera u novi byte stream file ili dodaje string već postojećem file-u. VI otvara ili kreira file, upisuje podatke i zatim zatvara file.



VI **Citaj karaktere iz file-a** (Read Characters from File) čita specificirani broj karaktera iz byte stream file-a počevši od specificiranog karaktera (offset). Ova VI otvara file ispred i zatvara ga nakon toga.



VI **Citaj linije iz file-a** (Read Lines From File), cita specificirani broj linija iz byte stream file-a počevši od specificiranog offseta (pomaka). Ova VI otvara file prije toga i zatvara ga nakon toga.



VI **Upisi u spreadsheet file** (Write To Spreadsheet File), pretvara 1D (jednodimenzionalna) ili 2D (dvodimenzionalna) polja brojeva jednostruke preciznosti za string tekst i upisuje tekst u novi byte stream file ili priključuje string postojećem file-u. Mi možemo također, kao opciju transponirati podatke (transpose). Ova VI otvara ili kreira file prije toga i zatvara ga nakon operacije. Možemo koristiti ovu VI da kreiramo tekst file-ove koje mogu citati većina spreadsheet programa.



VI **Citaj iz spreadsheet file-a** (Read from Spreadsheet File), cita specificirani broj linija ili redova iz numerickog tekst file-a, počevši od specificiranog offset karaktera, i pretvarajući podatke u 2D, polje brojeva jednostruke preciznosti. Možemo opciono transponovati polje. Možemo koristiti ovu VI da citamo spreadsheet file-ove u tekst formatu.

Upisivanje u Spreadsheet file

Jedna vrlo česta aplikacija za pohranjivanje podataka u file je da formatira tekst file tako da ga možete otvoriti u spreadsheet-u. U većini spreadsheet-ova tabovi razdjeljuju kolone a EOL (End of line, kraj linije teksta) karakteri razdjeljuju redove, kako je to pokazano na narednoj slici:

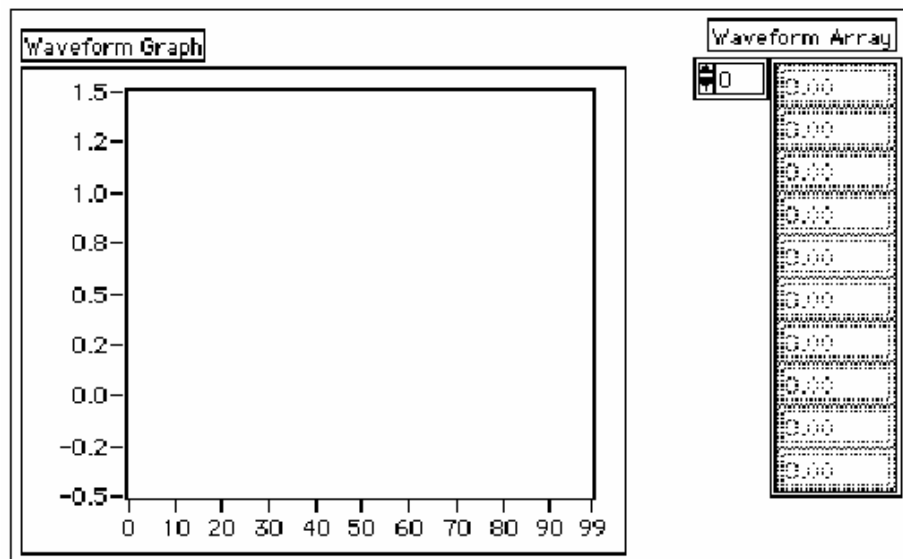
0.00	0.4258	☛ = Tab
1.00	0.3073	☛ = Line Separator
2.00	0.9453	
3.00	0.9640	
4.00	0.9517	

Otvarajući file koristeći spreadsheet program daje slijedecu tabelu:

	A	B	C
1	0	0.4258	
2	1	0.3073	
3	2	0.9453	
4	3	0.964	
5	4	0.9517	
6			

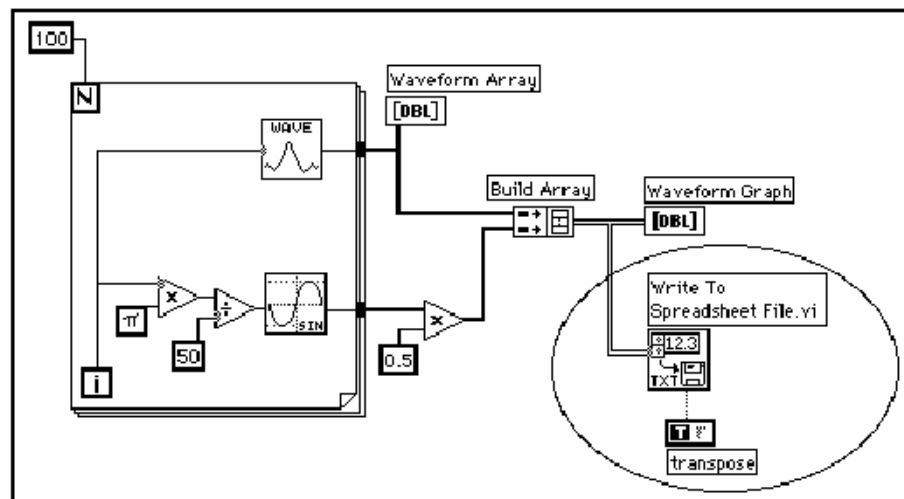
CILJ Mi ćemo modificirati postojeću VI da koristimo file I/O funkciju tako da možemo pohraniti podatke u novi file u ASCII formatu. Kasnije, možemo pristupiti ovom file-u iz spreadsheet aplikacije.

Prednji Panel



Otvoriti **My Graph Waveform Arrays.vi** koju smo izgradili u četvrtom poglavlju. Ako se sjećamo, ova VI generira dva skupa podataka u poljima i iscrtava ih na grafu. Modificiracemo ovu VI da upisemo dva polja podataka u file gdje svaka kolona sadrži po jedno polje podataka.

Blok Dijagram



- Otvorite blok dijagram od **My Graph Waveforms Arrays** i modificirajte VI dodavajući blok dijagram funkcije koje su bile smjestene unutar ovalne kruznice, kako je to prikazano na prethodnoj slici.



Vi **Upisi u spreadsheet file** (Write To Spreadsheet File, u **Functions>>File I/O**) pretvara dvodimenzionalno polje u spreadsheet string i upisuje ga u file. Ako niste specificirali ime staze (path) gdje upisati file, tada će iskociti dijalog boks i zatražiti

da unesete ime file-a. VI **Upisi u spreadsheet file** upisuje bilo 1-dimenzionalno ili 2-dimenzionalno polje u file. Postoji mi imamo 2D polje podataka u ovome primjeru, ne moramo da ozicavamo u 1D ulaz. Sa ovom VI, mozemo koristiti spreadsheet razdjelnicu (delimiter) ili string delimitera, kao sto su tab-ovi ili zarezi; u nasim podacima.

Bulova konstanta (Functions>>Boolean) kontrolira da li ili ne LabVIEW transponira 2D polje prije nego sto ga upise u file. Da bi se promjenila vrijednost na TRUE kliknite na konstantu sa Operativnim alatom. U ovom slucaju, mi zelimo da podatci budu transponovani, jer su polja podataka specifičnih redova (tj. svaki red dvo-dimenzionalnog polja je polje podataka). Postoji svaka kolona spreadsheet file-a sadrzi polje podataka, 2D polje se mora prvo transponovati.

3. Vratite se na prednji panel i izvršite VI. Nakon sto su polja podataka generirana, file dijalog boks ce traziti ime file-a za novi file kojeg kreiramo. Unesite ime file-a i kliknite na **OK**.

*Napomena: Ne pokusavajte da unosite podatke u VI biblioteke, kao sto je **mywork.llb**. Ukoliko ovo pokusate to moze rezultirati u prepisivanju preko vase biblioteke i gubitku vaseg prethodnog rada.*

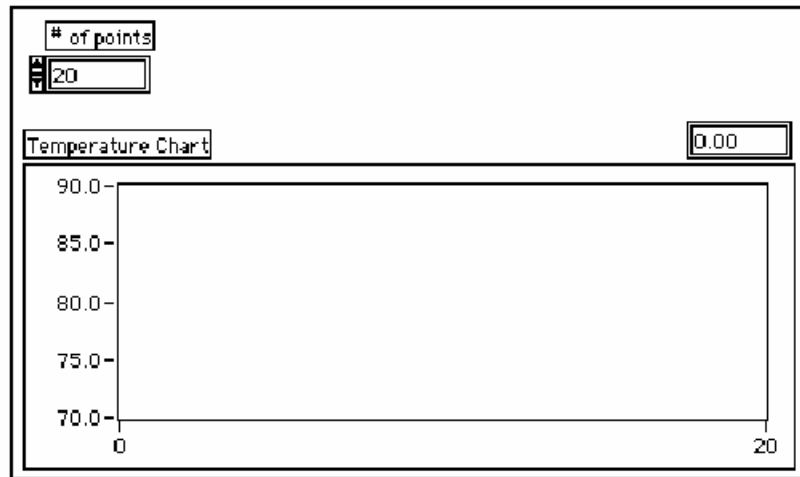
4. Pohranite VI, nazovite je **My Waveform Arrays to File.vi** i zatvorite VI.
5. Vi sada mozete koristiti spreadsheet software ili editor teksta da otvorite i pogledate u file koji ste upravo krierali. Trebate vidjeti dvije kolone od 100 elemenata.

U ovome primjeru, podatci nisu bili pretvoreni ili upisani u file sve dok sva polja podataka nisu bila prikupljena. Ako prikupljamo velike bafere podataka ili bi zelili da upisujemo vrijednosti podataka na disk u trenutku kada su generisani, tada moramo koristiti razlicitu VI za File I/O.

Dodavanje podataka u file

CILJ Kreiracemo VI da pridodamo temperaturne podatke u file u ASCII formatu. Ova VI koristi For konturu da generira temperaturne vrijednosti i pohrani ih u file. Za vrijeme svake iteracije, mi cemo vrsiti pretvaranje podataka u string, dodati zarez (comma) kao rezdjelni karakter (delimiter), i pridodati (append) string file-u.

Prednji Panel



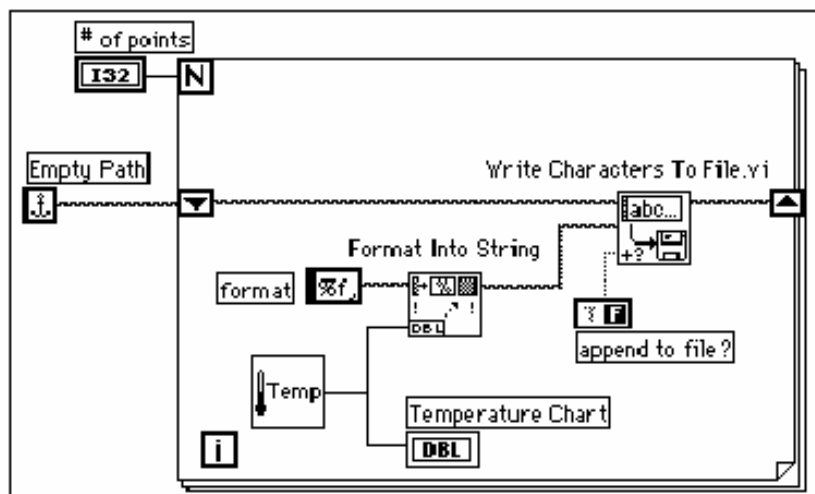
1. Otvoriti novi prednji panel i postaviti objekte kako je to pokazano na prethodnoj slici.

Prednji panel sadrzi digitalni kontrolni element i chart valnog oblika. Izaberite Show>>Digital Display. Kontrolni element **# of points** specificira koliko mnogo temperaturnih vrijednosti treba prikupiti i upisati u file. Chart prikazuje krivu temperature. Reskalirati y osu charta za opseg 70.0 do 90.0, i reskalirati x osu za opseg 0 do 20.

2. Kliknuti (pop-up) na **# of points** digitalni kontrolni element i izabrati Representation>>Long.



Blok Dijagram



1. Otvoriti blok dijagram

2. Dodati For konturu i uvecati je. Ova VI generira niz temperaturnih vrijednosti specificiranih sa **# of Points** kontrolnim elementom.
3. Dodati sift registar u konturu klikanjem na ivicu konture. Ovaj sift registar sadrzi ime staze (path name) do file-a.
4. Završiti sa ozicjenjem objekata.



VI **Isprazni konstantu staze** (Empty path constant ,iz **Functions>>File I/O >>File Constants**) . Ova funkcija inicijalizira sift registar tako da prvi put kada pokusate da upisete vrijednost u file, staza je prazna. File dijalog boks ce traziti da unesete ime file-a.



VI **My Thermometer** koju smo izgradili u Poglavlju 2, (**Functions>>Select a VI..**) ili VI **Digital Thermometer** (**Functions>>Tutorial**) vraća simuliranu vrijednost temperaturnog mjerenja iz senzora temperature.



Funkcija **Formatiraj u string** (**Functions>>String**) pretvara temperaturno mjerenje (broj) u string i dodaje mu zarez koji ga slijedi .



String konstanta (Functions>>String). Ovaj format specificira da mi zelimo pretvoriti broj u format stringa sa frakcijom u string sa zarezom.

VI **Upisi karaktere u file** (Write Characters To File , u **Functions>>File I/O**) upisuje string karaktera u file.



Bulova konstanta (Functions>>Boolean) postavlja **append to file?** ulaz od **Write Characters To File** Vi -a na vrijednost TRUE tako da nove vrijednosti temperature su pridodate u izabrani file svaki put kada kontura se ponovno izvrsava (iterira). Kliknite sa Operativnim alatom na konstantu da bi postavili njenu vrijednost na TRUE.

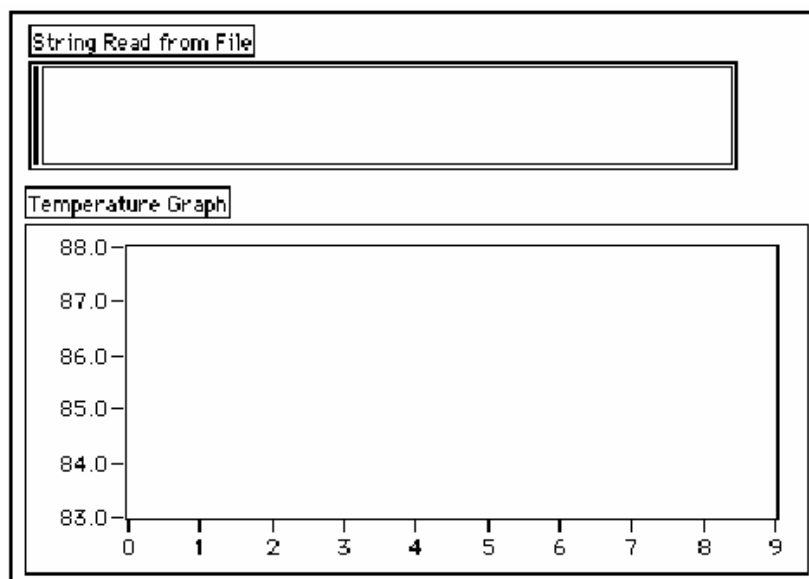


5. Vratite se na prednji panel i izvršite VI sa **# of points** postavljeno na 20. File dijalog boks ce vam zatražiti da unesete ime filea. Kada unesete ime file-a, VI pocinje upisivanje temperaturnih vrijednosti u taj file kako se generise svaka tacka.
6. Pohranite VI, imenujte je **My Write Temperature to File.vi**, i zatvorite VI.
7. Koristite bilo koji software za procesiranje teksta kao naprimjer Write za Windows-e , da otvorite taj file podataka i posmatrate njegov sadrzaj. Treba da ste dobili file koji sadrzi 20 vrijednosti podataka (sa preciznoscu od 3 mjesta nakon decimalne tacke) , razdjeljenih medjusobno sa zarezima.

Citanje Podataka iz File-a

CILJ Kreiracemo VI koja cita file podataka koje smo upisali u prethodnom primjeru, i prikazati te podatke na grafu valnog oblika. Moramo citati podatke u istom formatu u kojem smo ih pohranili. Zbog toga, posto smo prvobitno pohranili podatke u ASCII formatu koristeći string tip podataka, moramo ih citati u obliku stringa podataka sa jednom od Vi-eva za file I/O.

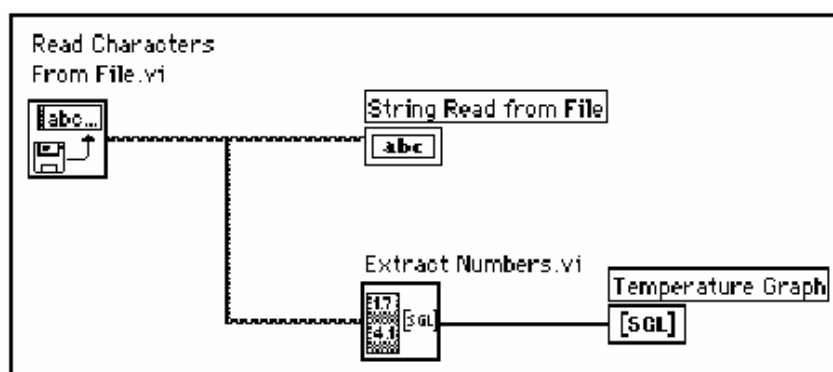
Prednji Panel



1. Otvoriti novi prednji panel i izgraditi prednji panel pokazan na prethodnoj ilustraciji.

Prednji panel sadrži string indikator i graf valnog oblika. Indikator **String Read from File** prikazuje temperaturne podatke razdjeljene sa zarezima iz file-a kojeg smo upisali u posljednjem primjeru kojeg smo uradili. Graf prikazuje temperaturnu krivu.

Blok Dijagram



1. Izgradite blok dijagram kakav je prikazan na prethodnoj slici.



VI **Citaj karaktere iz file-a** (Read Characters From File, iz **Functions>>File I/O**) čita podatke iz file-a i izdaje informacije u string. Ako ime staze (path) nije specificirano, file dijalog boks će tražiti da unesete ime file-a. U ovome primjeru, ne treba da određujete broj karaktera koje treba pročitati, jer ima manje karaktera u file-u od default vrijednosti (512).

Morate znati kako su podatci pohranjeni u file da bi ih iscitati iz file-a. Ako znate koliko je dugacak file, mozete koristiti VI **Citaj karaktere if file-a** (Read Characters From File), da odredite poznati broj karaktera koje treba pročitati.



VI **Izvuci brojeve** (Extract Numbers , iz **Functions>>Tutorials**) uzima ASCII string koji sadrzi brojeve odjeljene sa zarezima, LF (line feed, nova linija), ili drugim ne-numericnim karakterima i pretvara ih u polje numerickih vrijednosti.

2. Vratite se na prednji panel i izvršite VI. Izaberite file podataka koji ste upravo upisali na disk kada vam to zatrazi file dijalog boks. Trebate vidjeti iste vrijednosti podataka prikazane u grafu kao sto ste ih vidjeli u primjeru VI **My Write temperature to File** .
3. Pohranite VI, nazovite je **My Temperature from File.vi** , i zatvorite VI.

POGLAVLJE 8

PRIKUPLJANJE PODATAKA I KONTROLA INSTRUMENATA

KORISTENJE LABVIEW ZA PRIKUPLJANJE PODATAKA

Jedna od najdragocijenijih karakteristika LabVIEW-a je njegova sposobnost da prikuplja podatke iz skoro bilo kojeg izvora. LabVIEW sadrži VI-eve za kontrolu slijedećeg:

- Plug-in modula za prikupljanje podataka (pod Windows, Macintosh i Sun Operativnim sistemima)
- GPIB (IEEE 488) instrumenta
- Instrumenta sa serijskim kanalom veze (portom)
- VXI instrumenta (pod Windows, Macintosh i Sun Operativnim sistemima)
- PXI instrumenata (PCI instrument extension bus)

Ovi VI-evi koriste National Instruments (NI), standardne industrijske software-ske drajvere da obezbjede kompletnu kontrolu korisnikovog hardware-a za prikupljanje podataka i kontrolu instrumenata.

Plug-in moduli za akviziciju podataka

National Instruments (NI) proizvodi sve komponente koji su potrebne za gradnju kompletnih akvizicionih sistema i mjernih instalacija. Plug-in moduli su na raspolaganju za standardne buseve mikroprocesorskih PC konfiguracija kao što su: PC/AT, EISA, PCI, IBM PS/2 MicroChannel, Macintosh NuBus, Macintosh LC/LCII , i SPARCstation Sbus računari.

Ovi moduli imaju različite kombinacije analognih, digitalnih, i vremenskih ulaza i izlaza. Možemo koristiti front-end (prednji kraj) SCXI multipleksere sa kondicioniranjem signala da ekonomično povećamo broj analognih ulaznih kanala. Široka skupina modula za kondicioniranje signala za termoelemente, termootpore (RTD - resistance temperature detectors, otporni detektori temperature), naponski i strujni ulazi , i visoko strujni digitalni ulazi i izlazi kompletiraju ovu liniju hardware-skih modula za akviziciju signala.

VISA Biblioteka

VISA je jedinstvena biblioteka interfejsa za kontrolu GPIB, VXI, i drugih tipova instrumenata. Koristeći VISA funkcije, možemo konstruirati jedinstveni VI kao instrument drajver, koja kontrolira specifičan tip instrumenta, preko nekoliko različitih I/O medija. String se prenosi na VISA **Open** funkciju da bi se izvršila selekcija tipa I/O koji će se koristiti za komunikaciju sa instrumentom.

Jedanput kada se otvori sesija sa instrumentom, VISA funkcije, kao što su VISA **Read** i VISA **Write**, izvršavaju I/O aktivnosti instrumenta na generički način.

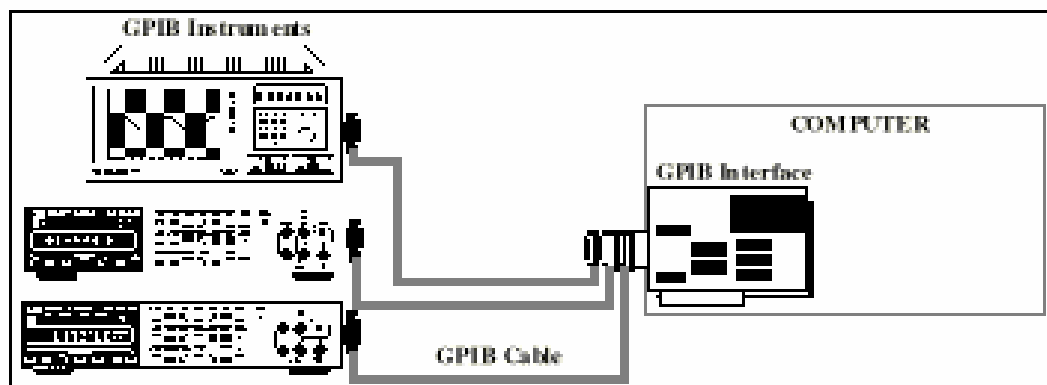
Dakle, program nije vezan ni za jednu specifičnu GPIB ili VXI funkciju. VISA instrument drajver se smatra interfejsom koji je nezavistan i može se koristiti u nekoliko različitih sistema.

Instrument drajveri koji koriste VISA funkcije, hvataju (capture) aktivnosti specifične za dati instrument a ne za komunikacioni medij. Ovo otvara više mogućnosti za višestruko korištenje instrument drajvera u nizu različitih programa.

GPIB BUS

Interfejsni bus opšte namjene (General Purpose Interface Bus- GPIB) , takodjer nazvan i IEEE 488, je metod komuniciranja sa autonomnim instrumentima, kao što su multimetri, osciloskopi, analizatori, itd. **NI** proizvodi mnoštvo prizvoda za kontrolu instrumenata sa GPIB busom.

Najdirektniji metod je da se instalira plug-in GPIB kartica u PC računar i poveže instrument direktno sa ovim modulom koristeći GPIB kabl.



LabVIEW GPIB funkcije kontrolišu NI GPIB interfejse. LabVIEW koristi NI, NI-488.2 standardni software koji dolazi zajedno sa GPIB interfejsom.

GPIB biblioteka (Functions>>Instrument I/O) sadrži i tradicionalne GPIB funkcije kao i 488.2 funkcije. GPIB 488.2 funkcije dodaju IEEE 488.2 kompatibilnost LabVIEW-u. Ove funkcije implementiraju pozive koje IEEE 488.2 specificira i slične su rutinama u okviru NI -488.2 software-a.

GPIB instrumenti nude test I proizvodnim inženjerima najširu selekciju proizvođača instrumenata, od onih opšte namjene do specijaliziranih instrumenata.

Kontroleri(controllers), talkers I Listeners

Da bi se odredilo koji uređaj ima aktivnu kontrolu nad basom, uređaji u okviru GPIB basa se kategoriziraju kao Kontroleri , talkers (oni koji govore tj. šalju podatke) I listeners (tj. oni koji slušaju odnosno primaju podatke). Svaki uređaj na GPIB basu ima jedinstvenu GPIB primarnu adresu između 0 i 30.

Kontroler definira komunikacione linkove, odgovara uređajima na njihove zahtjeve za servisima, šalje GPIB komande, i prenosi ili prima kontrolu nad basom.

Talker su instruirani od strane kontrolora da govore ,i postave podatke na GPIB. U svakom datom trenutku vremena , samo jedan uređaj može biti adresiran da govori na basu.

Listeneri su adresirani od strane Kontrolora da slušaju i čitaju podatke sa GPIB basa. Više uređaja istovremeno može biti određeno od strane kontrolora da sluša (prima podatke).

Hardwareška specifikacija GPIB basa.

GPIB je digitalni , paralelni bas sa 24 linije. Sastoji se od osam linija podataka (data lines DIO 1-8), 5 linija za upravljanje basom (EOI, IFC, SRQ, ATN, REN), tri handshake (sinhronizacione) linije (DAV, NRFD, NDAC),i osam linija umašenja.

GPIB koristi dakle 8 bitni paralen i odnosno bajt serijski, asinhroni prenos podataka. Ovo znači da su cijeli bajti sekvencijalno handshakirani duž basa sa brzinom koju određuje najsporiji učesnik u prenosu podataka.

Pošto je jedinica podatka na GPIB basu bajt (osam bita), poruke koje se prenose su vrlo često kodirane kao stringovi ASCII karaktera.

Dodatne električne specifikacije dozvoljavaju podacima da se prenose preko GPIB-ja sa maksimalnom brzinom od 1 MB/sec, pošto je GPIB linijski transmisioni sistem. Ove specifikacije su:

- maksimalna udaljenost od 4 m između bilo koja dva uređaja I prosječna udaljenost od 2 m na cijeloj dužini basa.
- Maksimalna dužina kabla od 20 m.
- Maksimalno 15 uređaja se može spojiti na svaki bas sa najmanje 2-3 uređaja kao aktivno (tj. napajano iz mreže a ne sa basa).

Ako korisnik prekorači bilo koji od ovih limita, biće potreban dodatni hardware da produži dužinu basa ili poveća broj dozvoljenih uređaja.

Veća brzina u prenosu podataka može se postići sa HS488 (high speed 488) uređajima I kontrolorima što predstavlja jedno proširenje GPIB specifikacija I podržano je od strane kontrolera koje proizvodi **NI**:

Napomena : Kod korištenja GPIB basa I komunikacije izmedju instrumenata na ovom basu, ako je moguće, treba koristiti VISA funkcije radije nego GPIB zbog VISA raznovrsnosti.

VXI BUS (VME proširenje za Instrumentaciju).

VXI bus je brzo razvijajuća platforma za instrumentacione sisteme. VXI koristi mainframe šasiju sa maksimalno trinaest slotova (mjesto za plaganje kartica) da primi modularne plug-in kartice. Veliki niz raznovrsnih instrumenata i dimenzija mainframe-a (računarskog dijela modularnog sistema), su na raspolaganju brojnim korisnicima i sistem integratorima. Moguće je koristiti instrumente različitih veličina u okviru istog mainframe-a. Može se ostvariti kontrola VXI mainframe-ova na nekoliko različitih načina.

VXI definira standardni komunikacioni protokol ka ovim šasijama sa modularnim plajljivim karticama. Putem ovog interfejsa korisnik može koristiti ASCII komande da kontrolira instrumente, slično kao i kod GPIB.

VXI bas specifikacija je proširenje specifikacije VME bas specifikacije (definirane sa IEEE 1014 specifikacijom). Kao elektromehanički superset od VME basa, VXI bas koristi iste konektore na zadnjoj bas ploči (backplane) kao i VME bas, istu veličinu ploča i iste signale koji su definirani i u specifikacijama VME basa.

VXI bas dodaje dvije nove veličine štampanih ploča , mjenja širinu modula u racku, I definira neke dodatne signale na backplanu (zadnjoj bas ploči).

VXI hardwareške komponente

VXI sistem se sastoji od racka (mainframea) ,kontrolera, instrumenata, i kablova. VXI mainframe je šasija, rack ili kavez (cage), koji sadrži napojnu jedinicu, sistem hladjenja, zadnju štampanu ploču (backplane), i elemente za fizičku montažu VXI bas modula. Mainframei se pojavljuju u četiri veličine (A,B,C i D) koje korespondiraju sa najvećim pločama koje se mogu uplagovati u mainframe.

VXI konfiguracije

Korisnik može koristiti VXI na vrlo različite načine. Može ih integrirati u sistem zajedno sa drugim GPIB instrumentima, ili može izgraditi sistem koristeći samo VXI instrumente. Svaka konfiguracija sistema ima slijedeće jedinstvene prednosti:

Ugradjeni (embedded) kontroleri

- visoku performansu I malu veličinu
 - direktni pristup ka VXI bas brzom odzivu na interapt.
- ❖ MXI (multisystem extension interface) imaju:
- embedded performansu sa standardnim desktop računarima
 - koriste daljinski PC da kontrolišu VXI sistem
 - MITE/DMA 23 Mbyte/s transfer blokova

❖ GPIB u VXI translatorima :

- :kontroliraju VXI mainframe sa IEEE 488

Prva konfiguracija uključuje kustomizirani VXI kompjuter direktno unutar mainframea. Koristeći ovu konfiguraciju, korisnik može koristiti sve prednosti visokih performansi VXI pošto računar može da direktno komunicira sa VXI backplaneom (zadnjim basom).

Druga gore navedena konfiguracija kombinuje beneficije u performansi kustomiziranog embedded računara sa fleksibinošću desktop računara opšte namjene.

Sa ovom konfiguracijom , korisnik može koristiti MXI bas link velike brzine, da se poveže sa vanjskim kompjuterom direktno preko VXI backplanea.

Treća konfiguracija se sastoji od jednog ili više VXI mainframea linkovanih na vanjski kompjuter putem GPIB. Korisnik može koristiti ovu konfiguraciju da postepeno integriira VXI u postojeći GPIB sistem I programira VXI instrumente koristeći postojeći GPIB software.

LabVIEW ima VXI VI-eve za visoko-nivovsku i nisko-nivovsku kontrolu VXI sistema. Mi pristupamo ovim VI-evima preko **Functions>>Instrument I/O >> Visa**.

PXI modularna instrumentacija

Novi modularni instrumentacioni sistem baziran na PCI proširenju za instrumentaciju (**PCI eXtension for Instrumentation**) pruža PC bazirani mjerni sistem visokih performansi.

PXI je kompletno kompatibilan sa CompactPCI i inkorporira napredne osobine tajminga i trigerovanja koje su bile i kod VXI. PXI popunjava prazninu izmedju desktop PC-jeva niske cijene i skupih VXI i GPIB rješenja kombinujući industrijske standarde Windowsa, PCI, CompactPCI I VXI.

Korisnik dizajnira PXI sistem selektirajući sve, uključujući kontroler (embedded Pentium 3 i 4 klasu kao i periferale), šasiju, i module.

PXI moduli mogu biti : analogno digitalni, digitalno analogni, digitalni I/O, multifunkcionalni ulazno/izlazni moduli, za prikupljanje podatka, slike, kontrolu kretanja, te instrumenti kao osciloskopi, multimetri, serijski analizatori podataka, kao I drugi specifični uređaji.

Komunikacija Serijskim portovima

Serijska komunikacija je popularno sredstvo prenosa podataka između računara i perifernih uređaja kao što su štampači, ploteri, ili programabilni instrumenti. Serijska komunikacija koristi transmiter da pošalje podatke, jedan bit u svakom trenutku vremena, preko jedne jedine komunikacione linije do prijemnog modula (receiver). Ovaj metod komunikacije je vrlo čest kod prenosa podataka malim brzinama na relativno veće distance. Naprimjer, serijski podatci se prenose modemima, ili preko standardnih telefonskih linija.

Serijska komunikacija je bila vrlo popularna jer većina PC kompjutera je imala jedan ili dva serijska porta. Ograničenje serijske komunikacije, je u tome što serijski port može komunicirati sa samo jednim uređajem. Da bi se to moglo ostvariti sa više uređaja moramo koristiti modul sa višestrukim serijskim portovima ili serijski multiplekserski boks.

Prije nego što počnemo koristiti LabVIEW za serijsku komunikaciju, moramo prvo osigurati da je instrument korektno spojen sa računarom. Za Windows-e, moramo također obezbjediti da nema konflikata među interapt nivoima. Jedan od načina da se ovo osigura je da se koristi softwareski terminal za opšti terminal kao što je Microsoft Windows terminal ili *Zterm* program.

Korisnik mora specificirati parametre za serijsku komunikaciju: brzinu u Baud-ima (bits/sec) za prenos, broj bita podataka (data bits) sa kojima je kodirana riječ, vrijednost za bit pariteta (parity), kao i broj stop bita.

Svaki preneseni karakter se pakuje u okvir karaktera koji se sastoji od jednog start bita i nakon njega slijede bitovi podataka.

Bit starta dakle signalizira početak svakog okvira za karakter.

Bitovi podataka se prenose "upside down and backwards", što znači da se koristi invertirana logika a redoslijed prenosa je od bita najmanje važnosti (LSB- least significant bit) ka bitu najveće važnosti (MSB- most significant bit).

Da bi se interpretirali biti u okviru karaktera, moramo ih čitati sa desna na lijevo, i čitati jedan (1) kada je napon negativan a nula (0) kada je napon pozitivan.

Opcioni bit pariteta slijedi nakon bitova podataka u okviru karaktera. Bit pariteta , ako je prisutan, također će slijediti inverznu logiku. Ovaj bit je uključen kao jednostavan način za provjeravanje grešaka u prenosu. Korisnik unaprijed specificira da li je paritet prenosa paran (even) ili neparan (odd). Ako izabere da paritet bude neparan (odd), transmiter serijskog prenosa će postaviti bit pariteta na takav način da napravi neparan broj 1-inica među bitovima podataka i bitom pariteta.

Posljednji dio okvira karaktera se sastoji od 1, 1.5 ili 2 stop bita. Ovi bitovi su uvijek predstavljeni sa negativnim naponom. Ako se ne prenosi više karaktera, linija će ostati na negativnom (MARK) nivou.

Prenos slijedećeg okvira karaktera , ako ga ima, počće sa start bitom sa pozitivnim (SPACE) naponom.

Brzina prenosa podataka serijskom komunikacijom

Brzina prenosa u serijskoj komunikaciji se može sračunati u karakterima po sekundi za date komunikacione parametre , dijeleći brzinu prenosa u Baudima sa brojem bita u okviru jednog okvira karaktera (character frame).

Standardi kod serijske komunikacije

Postoji mnogo standarda za serijsku komunikaciju. Najčešći su slijedeći standardi:

- RS-232 (ANSI/EIA-232) se koristi za mnoge namjene, kao što je priključenje miša, štampača, modema, kao i industrijske instrumentacije. Zahvaljujući poboljšanjima u kvalitetu kola za drajving linije (line drivers) il kablova, aplikacije često povećavaju performansu RS-232 i iznad distance i brzine koje su standardom specificirane. RS-232 je ograničen na tačka-tačka (point -to-point) konekciju izmedju PC serijskih portova i uređaja.

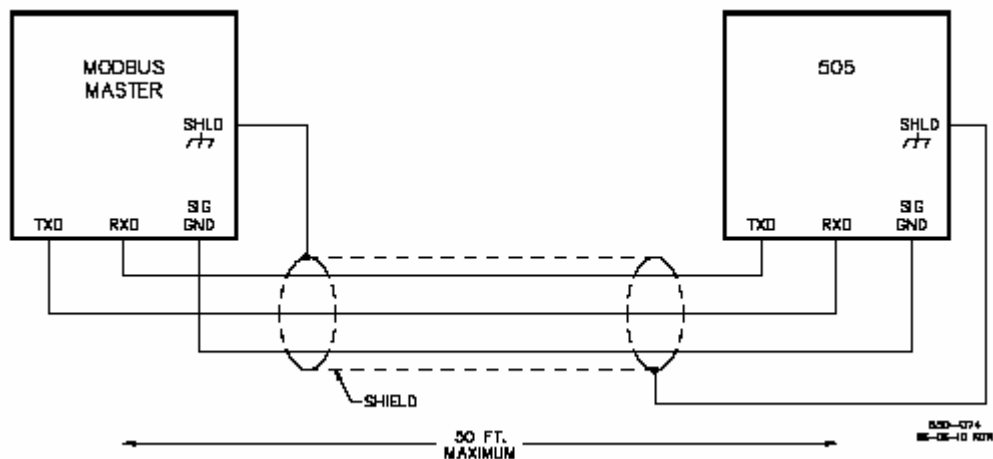


Figure 3-10. Typical RS-232 Communications

- RS-422 (AIA RS-422 Standard) koristi diferencijalni električni signal, za razliku od nebalansiranog jednostrukog (single ended) signala referenciranog prema masi kod RS-232. Diferencijalni prenos, koji koristi po dvije linije za prijem I prenos podataka , rezultira u većoj otpornosti na šum , i mnogo dužim distancama prenosa, u poredjenju sa RS-232. Ove dvije osobine , tj. veća otpornost na šumove i daljina prenosa su dvije velike prednosti u industrijskom ambijentu korištenja.

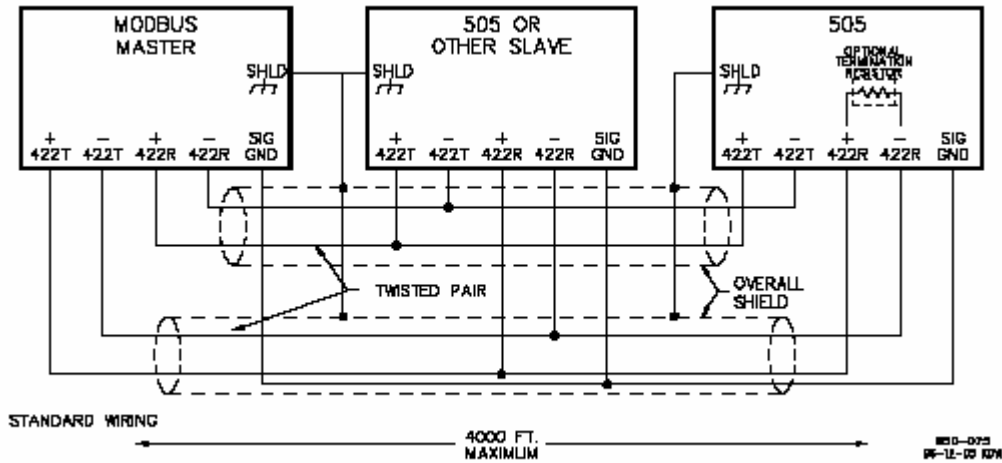


Figure 3-11. Typical RS-422 Communications

- RS-485 (EIA-485) standard, je poboljšanje u odnosu na RS-422 standard, jer dozvoljava spajanje više uređaja (do 32) na jedan port, i definiira električne karakteristike koje su potrebne da se obezbijedi adekvatni naponi signala u uslovima maksimalnog opterećenja. Sa ovom unaprijedjenom mogućnošću priključka više portova zajedno (multidrop), korisnik može formirati mrežu uređaja koji su povezani na jedan RS-485 serijski port.

Imunitet na šum ili multidrop sposobnost čine RS-485 serijskom komunikacijom koja će najčešće biti korištena u industrijskim komunikacijama, kada se zahtjeva da mnogo uređaja bude povezano međusobno na serijskoj komunikaciji kao što su PC-jevi, I/O uređaji za prikupljanje podataka iz procesa, MMI (HMI uređaji za interfejs između operatora i procesa koji vodi i nadzire (man (human) – machine interface), itd.

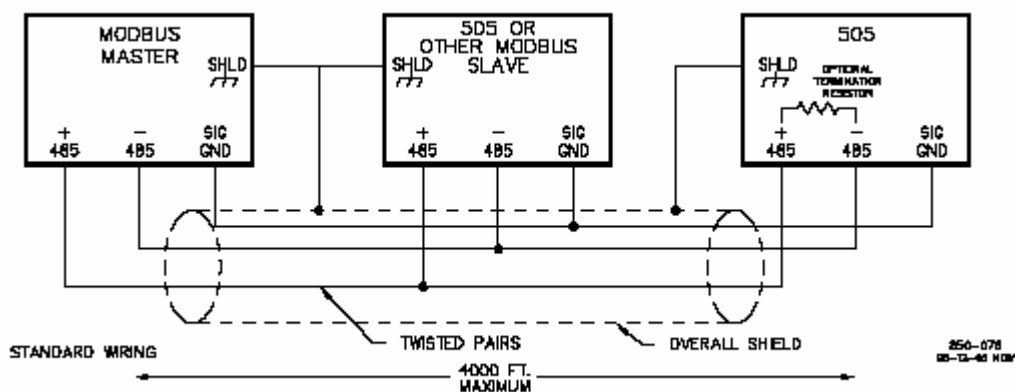


Figure 3-12. Typical RS-485 Communications

Kada mi spajamo neki instrument sa PC-jem koristeći serijsku komunikaciju, moramo obezbijediti korektan kabel za ovo spajanje koji će zavistiti da li je uređaj DCE (data communication equipment) ili DTE (data terminal equipment), što će opredjeliti da li je potrebno ukrštanje između pinova 2 (TXD – transmit) i 3 (RXD – receive). Nadalje trebamo postaviti na PC-ju i instrumentu usaglašene komunikacione parametre kao

što su : brzina u baudima (baud rate), broj data bita, stop bita, paritet, i kontrola toka podataka (hadshaking for flow control).

Jedanput kad je uspostavljena komunikacija sa instrumentom, mi smo spremni da koristimo LabVIEW-ov VI za serijski port, lociran u **Functions>>Instrument I/O >>Serial**.

LabVIEW sadrzi pet VI-a za serijsku komunikaciju i to:

Serial Port Init, Serial Port Write, Serial Port Read, Bytes at Serial port, i Serial Port Break.

Instrumenti na bazi računara

Instrumenti na bazi računara su realizovani na nekoliko platformi uključujući PCMCIA (PC) za laptop (notebook računare) , PCI za desktop računare i PXI.

Instrumenti na bazi računara su jedan od primjera virtuelnih instrumenata koji se sastoje od PC baziranog instrumentalnog modula, računara i aplikacionog softwarea. Tradicionalni instrumenti su samostalni (stand alone) uređaji , kod kojih je funkcionalnost instrumenta zatvorena (encapsulated) unutar crne kutije (“black box”).

Pošto tehnologije uređaja za digitalizaciju i minijaturizaciju se nastavljaju razvijati, danas su vrlo popularni PC kartice (PCMCIA) kao instrumentalni adapteri koji , zajedno sa laptopom , daju istu funkcionalnost kao i samostalni uređaji.

Instrumenti na bazi računara mogu koristiti prednosti snage procesiranja PC, velike i ekspanzirajuće memorije, displeja velike rezolucije, i mrežne povezljivosti sa intra i internetom, za lagani prenos podatka i rezultata mjerenja.

Korisnik može mjeriti napone, struje, otpornosti, koristeći ovakve instrumente, ili da proširi mogućnost virtuelnog instrumenta putem korištenja aplikacionog softwarea. Nadalje, on može kreirati data logger (registrator podataka) za automatsko prikupljanje i analizu podataka. Na taj način izvještaji o mjerenjima se mogu generirati u hodu (on the fly).

Sa aplikacionim softwareom, korisnik može kastomizirati mogućnosti virtuelne instrumentacije, da riješi mnoge mjerne probleme. Koristeći jeftinu PC tehnologiju, korisnik ima ekonomičnu alternativu nego da kupuje , nove skupe I fiksne funkcije, samostalne mjerne uređaje I instrumente.

Instrument Drajveri

Instrument drajver je software koji kontroliše dati specifični instrument.

LabVIEW je skoro idealno prilagodjen za kreiranje instrument drajvera. LabVIEW prednji panel može simulirati rad prednjeg panela realnog fizičkog instrumenta. Blok dijagram može slati neophodne komande instrumentu da izvršava operaciju koju specificira prednji panel. Kada završimo sa gradnjom instrument drajvera, nema više potrebe da se memoriraju komande za kontrolu instrumenta. Dovoljno je samo specificirati ulaz na prednjem panelu. Medjutim mala je vrijednost samo u

tome da se ima software-ski panel koji kontrolira instrument. Prava vrijednost je u tome da možemo koristiti instrument drajver kao subVI u sprezi sa drugim subVI-evima u većoj VI da kontrolira cjelokupni sistem.

LabVIEW ima biblioteku od preko 700 instrument drajvera za GPIB, serijsku, CAMAC , VXI (za Windows, Macintosh i Sun) i PXI instrumente. Pošto postoji veliki broj vrlo različitih instrumenata, nemoguće je demonstrirati tehnike kreiranja drajvera za sve tipove instrumenata; međutim, za instrumente bazirane na porukama svi drajveri grade komandni string i šalju ka instrumentu da izvrši operaciju koju specificira simulirani prednji panel.

Komandni string se sastoji od komandi specifičnih za tip uređaja (obično u ASCII), koje daljinski kontroliraju instrument. Zbog toga, instrument drajveri sadrže više funkcija manipulacije sa stringovima nego specifične interfejsne komande.

POGLAVLJE 9

Razvoj Labview instrument drajvera

Labview, grafički programski jezik koji je prvi uveo koncept virtualne instrumentacije, je tehnološka novina u rukama naučnika i inženjera sada već oko petnaest godina. Kako je rasla popularnost Labview-a, tako se povećavao i broj instrument drajvera - softwareskih modula projektovanih da upravljaju programabilnim instrumentima. Da bi pomogao razvoju ovih drajvera NI je kreirao standarde za strukturu ovih drajvera, management uređaja, instrumentni I/O dio (ulazno/izlazni interfejs), kao i način izvještavanja o greškama koje se mogu pojaviti.

Uvod

Instrument drajver je skup softwareskih rutina koje kontrolišu programabilni instrument. Svaka rutina korespondira nekoj programskoj operaciji kao:

konfigurisanje, očitavanje sa, pisanje na, kao i trigeriranje instrumenta.

Instrumentalni drajveri pojednostavljuju upravljanje instrumentom i reduciraju vrijeme za pisanje test programa, eliminisući potrebu da se uči programski protokol za svaki instrument. Biblioteka LabVIEW-a sadrži drajvere za instrumente za niz programabilnih instrumenata, uključujući GPIB, VXI, PXI, RS-232/422 I CAMAC instrumente.

Labview instrument drajveri obično komuniciraju sa instrumentima koristeći funkcije : "Virtual Instrument Software Architecture – VISA" (softwareska arhitektura virtuelnih instrumenata). VISA je protokol koji se koristi kada PC govori sa instrumentima.

Korisnik može koristiti VISA za mnoge različite tipove instrumenata , kao što je GPIB, serijski, VXI I PXI. Kada korisnik jedanput savlada kako da komunicira sa jednim tipom instrumenata koristeći VISA, nema potrebe da uči različit način komunikacije za drugi tip instrumenta.

On će trebati da nauči o specifičnim komandama za novi tip instrumenta , ali metodi kako se komande šalju i primaju od instrumenta se ne trebaju ponovo učiti i oni se ne mjenjaju.

Kad korisnik počinje da razvija aplikaciju za kontrolu instrumenta sa Labview-om, on ima opciju da koristi instrument drajver ili da komunicira direktno koristeći VISA.

LabVIEW software je idealan alat za razvoj drajvera za instrumente jer je pristup programiranju intuitivan. LabVIEW softwareska rutina, koja se naziva virtuelnim instrumentom (VI), se sastoji od:

- prednje ploče (front panel)
- blok dijagrama
- ikone/ konektora

Prednja ploča, analogna fizičkoj instrumentalnoj ploči instrumenta, je interaktivni korisnički interfejs VI. Na panelu, kontrolni i indikatorski elementi grafički predstavljaju ulaze i izlaze VI.

Blok dijagram, analog električnom kolu ožičenja instrumenta, je izvorni kod VI. Dodatno, blok dijagram, koji se sastoji od izvršnih blokova povezanih sa "žicama" toka podataka, pokazuje funkcionalnost VI.

Ikona/konektor je programski interfejs VI. Sastoji se od grafičke predstave VI (ikona) kao i od definicije ulaznih i izlaznih priključaka za VI (konektora). Kada je potrebno pozvati ili izvršiti VI iz nekog drugog VI modula, tj. kada je submodul kojeg poziva drugi VI modul, tada je potrebno postaviti kopiju prvog VI-evog ikone/konektora u blok dijagram VI modula koji ga poziva.

Informacije između dva VI modula se prenose kroz konektorske priključke.

LabVIEW instrumentalni drajveri su vrlo kvalitetni alati koji pojednostavljaju upravljanje instrumentima. Postoje VI za instrumentalne drajvere sadrže visoko nivojske funkcije sa intuitivnim prednjim panelima, korisnik može brzo istestirati i verificirati mogućnosti instrumenta bez da zna specifičnu sintaksu za taj instrument.

Tako je moguće skanirati blok dijagrame da bi se utvrdile relevantne osobine koje posjeduje, programska struktura, funkcionalnost, tok podataka (data flow), koje bi inače bile sakrivene i teško uočljive u programu koji je isključivo tekstualno struktuiran (tj. pisan samo kao niz komandi i iskaza).

Sto je najvažnije, korisnik može lako kreirati VI test sisteme programski povezujući instrument drajver VI-eve u okviru blok dijagrama.

Neke uobičajene zablude

LabVIEW instrumentalni drajver VI nije živi interaktivni prednji panel za instrument. Mada VI-jevi instrumentalnih drajvera mogu se interaktivno izvršavati, oni ne sadrže kontinualne konture koje očitavaju ulazne postavljene parametre i šalju instrumentalne komande u odzivu na real-time dinamičke ulazne komande korisnika.

Ustvari, oni (drajveri) očitavaju kontrolne komande sa prednjeg panela, formatiraju i šalju komandne stringove, očitavaju odzive na instrumentalne upite (poslate instrumentu), te ažuriraju indikatore u okviru prednjeg panela, jedanput u okviru izvršenja VI modula.

Ovo je vrlo važan koncept, da bi jedan VI instrumentalni drajver radio programabilno, on ne smije biti konstruiran tako da se zahtjeva interaktivni Operatorski ulaz da bi se modul izvršavao. Na taj način se izbjegava korištenje file ili tekst dijaloga ili nekog VI setup programa koji bi iskočio (pop-up) i zatražio od korisnika da unese neku vrijednost.

Naravno da ima aplikacija koje će upravo zahtjevati ovakav tip interaktivnog panela. Naprimjer, korisnik kao dizajner VI može željeti da budući korisnik unese postavne vrijednosti i parametre podešenja instrumenta koristeći neki **menu** bazirani interfejs.

Kako će se moći dobiti ovako ponašanje programa VI ako on nije dizajniran da radi na ovaj način?

Sa LabVIEW instrument drajverom postoji mnoštvo opcija za optimizaciju VI za interaktivno korištenje. Za brzo testiranje, korisnik može lako prisiliti VI da se izvršava u kontinualnoj konturi na taj način sto će pozvati njen prednji panel (pop-up) i kliknuti na taster za **continuous run**, i na taj način čineći da VI djeluje kao **soft** prednji panel koji kontroliše instrument u realnom vremenu.

Kod primjena, korisnik ima izbor mogućnosti modifikacije instrument drajvera VI putem omogućenja (enabling) pop-up akcije i dodavajući konture i uslove u dijagramu, ili korisnik može graditi visoko-nivovsku VI koja sadrži željeni interaktivni interfejs i poziva instrumentalni drajver VI u odgovarajućem trenutku u dijagramu.

Striktno interaktivno korištenje ne daje neke dodatne vrijednosti VI, korisnik može jednostavno pritiskati tastere na prednjoj ploči instrumenta umjesto da izvršava VI modul.

Medjutim, podsjetimo se da, kada gradimo LabVIEW instrumentalni drajver, on mora biti u stanju da radi i programabilno ali i interaktivno.

Ne treba koristiti dijaloge ili neki drugi način da bi se zatražilo od korisnika da unese ulaznu vrijednost.

Kod konfigurisanja, korisnik treba uvijek da **ožiči** sve kontrolne i indikacione elemente sa konektorom, kao i da poveže sve ulazne i izlazne vrijednosti preko ovih žica.

LabVIEW instrumentalni drajver nije ograničen samo na upravljanje jednim instrumentom. Mnogi korisnici žele da kontrolišu nekoliko identičnih instrumenata istovremeno, koristeći isti instrument drajver. Da li je ovo moguće?

Naravno, ukoliko je instrument drajver korektno dizajniran. VI instrumentalni drajveri, kao i svi ostali VI-evi u okviru LabVIEW, su serijski višekratno koristivi. To znači da LabVIEW normalno ne može koristiti višekratne pozive iste VI jedne preko drugih (interleave), ali može koristiti ponovno isti VI na serijski način. Zbog toga, postavljajući kontrolu adrese na prednju ploču instrument drajver-a VI-a i prenoseći ka ovom kontrolnom elementu različite adrese, korisnik može koristiti višekratno VI u svojoj aplikaciji da bi kontrolisao više od jednog instrumenta.

Da bi jedan instrument drajver VI bio višekratno korišten (ili multi-instance), podatci koji su sadržani unutar modula ne smiju biti djeljeni izmedju više instanci istog VI. Normalno, ovo nije problem jer LabVIEW omogućava serijsko višekratno korištenje VI, ulazne podatke konzumira (troši) VI a izlazni podatci se generišu sa svakim izvršenjem VI.

Medjutim, problem višekratnog korištenja postoji samo u slučaju kada se globalni podatci održavaju u okviru VI.

Naprimjer, globalni VI-i su po definiciji oni VI-i čiji se podaci dijele sa drugim dijelovima programa. Nadalje, neinicijalizirani šift registri, između izvršenja VI-eva također pohranjuju podatke koji mogu biti dijeljeni između višekratnih instanci VI. Instrument drajveri koji koriste ovaj mehanizam pohranjivanja ne mogu biti multi-instance drajveri jer podaci pohranjeni u VI od strane jednog instrumenta mogu biti iščitani ili prepisani od strane drugog instrumenta za koji ti podaci nisu namjenjeni.

U nekim instancama (rješenjima), memorija za globalne podatke izgleda kao jedina alternativa. Napr. ako korisnik postavi neke specifične parametre podešenja za jedan VI, ali su mu potrebni podaci ovih podešenja za drugi VI da bi odredio tok akcije koja slijedi, on može biti podtaknut da koristi globalni VI da u njega pohrani podatke prvog VI da bi ih čitao iz drugog VI. Nažalost, čineći ovo on ograničava drajver da kontroliše samo jedan instrument.

Kada je god to moguće, treba koristiti jedan instrument upit (query) unutar funkcije u pitanju, da bi se odredilo kako je instrument konfigurisan. Ako zahtjev za brzinom rada ili mogućnosti instrumenta ovo ne dozvoljavaju, postavne vrijednosti se mogu poslati kroz **žicu** do kontrolnih elemenata na prednjem panelu. Primjetimo da u ovom slučaju kontrolni element ustvari ne konfigurise postavne vrijednosti, nego samo o tome izvještava VI dijagrame tako da oni mogu odlučiti koju granu dijagrama da izvrše. Ovo je brže nego ispitujući (query) instrument, ali dodaje jedan dodatni ulaz na VI instrumentu i zahtjeva od korisnika da ga propisno konfigurise.

Konačno, korisnik može konstruisati standardni globalni VI modul koji se može koristiti od drugih instanci instrument drajvera koristeći neiskorištene šift registre podataka unutar svakog elementa koji korespondira sa jedinstvenom instrument adresom.

Instrument drajveri kod **NI** instrumentalne biblioteke su multi-instance drajveri. Ako se želi kreirati instrument drajver koji bi bio uvršten u ovu biblioteku, on mora biti također multi-instance drajver. Nadalje, prednji panel VI instrumenta mora uključiti handle (**ručku** za povezivanje sa programom) za kontrolu instrumenta, kao i da **ožiči** ovaj handle (ručku) sa svim I/O podmodulima VI u dijagramu.

Nadalje, svi neinicijalizirani šift registri i globalni VI trebaju biti uklonjeni iz drajvera. Na taj način će se eliminisati elementi za pohranjivanje podataka unutar drajvera i učiniti modul višekratno koristivim za više nego jedan instrument.

Model unutarnjeg dizajna instrument drajvera.

Moderni GPIB test instrumenti se karakterišu sa sve većim brojem funkcija, modova (načina) rada, i kontrolnih elemenata kojima operiše korisnik. Mikroprocesorski bazirani operativni sistemi često pojednostavljaju korištenje ovih instrumenata, predstavljajući korisniku standardni set menija koji pokazuju samo opcije koje mu stoje na raspolaganju za taj način rada ili neku specifičnu funkciju. Nasuprot ovome, VXI bus instrumenti, koji mogu imati isti ili i veći nivo kompleksnosti, obično ne sadrže elemente prednje ploče za kontrolu ili prikaz vrijednosti. Uprkos definisanju standardnog komandnog jezika kao i standarda

kao što su SCPI i IEEE 488.2, projektovanje i razvoj software-a za upravljanje modernim instrumentima i test uređajima je vremenski vrlo intenzivno i neizvjesno po konačnom rezultatu.

Zato je **NI**, da bi pomogao korisnike LabVIEW-a, razvio biblioteke instrument drajvera za popularne instrumente. Svaki instrument drajver ima jedan ili više VI organiziranih u modularnu hijerarhiju koja sadrži ne samo visoko nivovske aplikacione VI-eve, nego takodjer i VI komponente instrument drajvera.

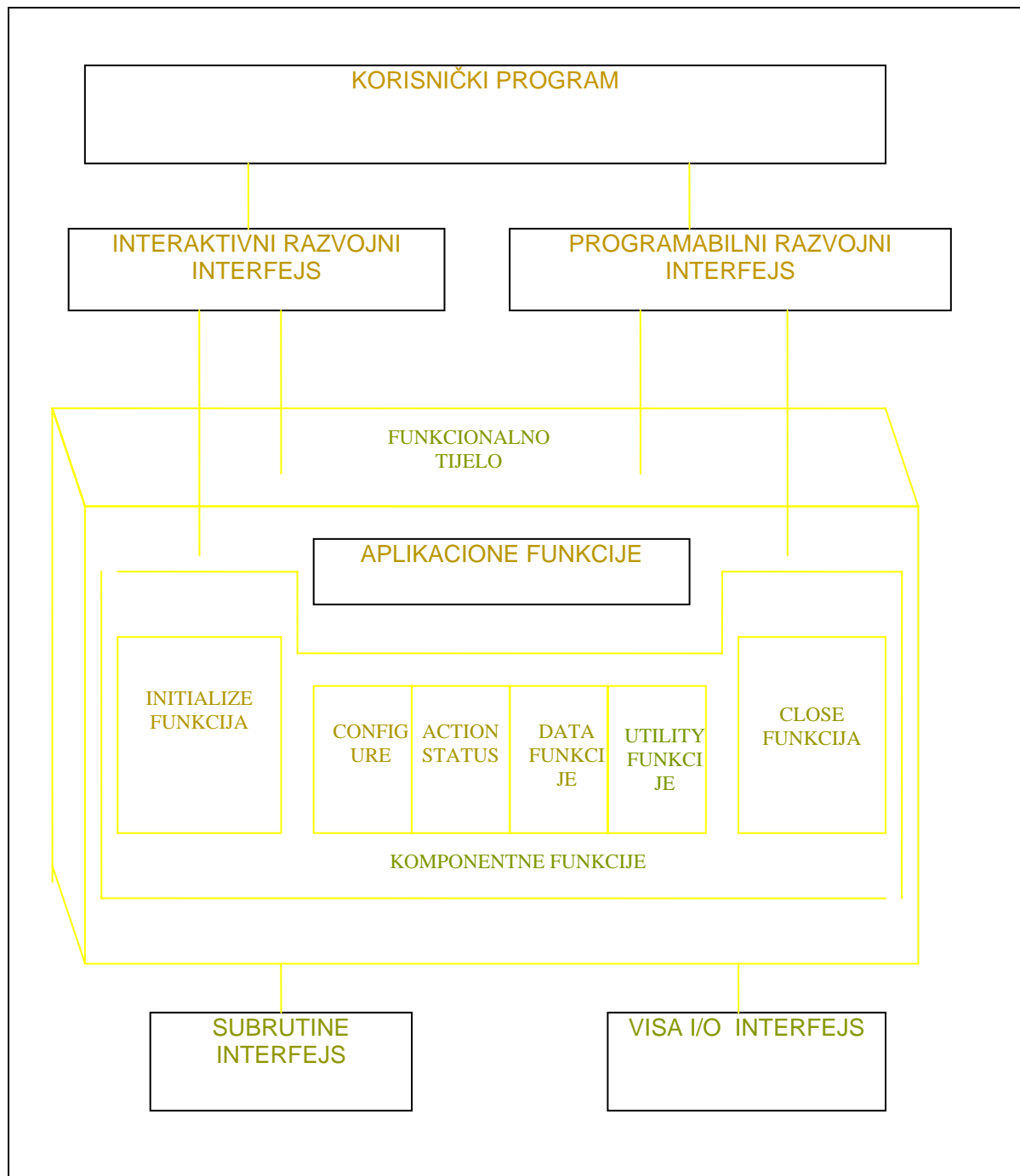
Interni dizajn model LabVIEW instrument drajvera je pokazan na narednoj slici 1, definise organizaciju funkcionalnog tijela drajvera. Ovaj model je važan za one koji razvijaju instrument drajvere jer je on osnova na kojoj počivaju principi razvoja modula. On je takodjer važan i za one koji su krajnji korisnici modula, jer su svi LabVIEW instrument drajveri organizovani u skladu sa ovim modelom. Jedanput kada se usvoji ovaj model, on se može koristiti na nizu drugih instrument drajvera.

Funkcionalno tijelo LabVIEW instrument drajvera se sastoji od dvije glavne kategorije VI-eva. Prva kategorija je skup komponentnih VI-eva, koji su individualni softwareski moduli od kojih svaki kontroliše specifični dio funkcionalnosti instrumenta. Druga kategorija su skup visoko-nivovskih aplikacionih VI-eva koji pokazuju kako kombinirati komponentne VI-eve da bi se izvršile bazične test i mjerne operacije sa instrumentom.

Interni dizajn model LabVIEW instrument drajvera se bazira na provjerenoj tehnologiji. Sa ovim modelom, korisnik ima neophodnu granularnost (rezoluciju) da upravlja korektno instrumentom unutar svoje softwareske aplikacije.

Tako je napr. moguće inicijalizirati sve jedanput kod starta, konfigurisati multiple instrumente, a zatim trigerovati simulatano nekoliko instrumenata.

Kao drugi primjer, moguće je inicijalizirati I konfigurisati instrument jedanput, a zatim trigerovati I očitavati sa instrumenta više puta.



Slika 1

Instrument drajver aplikacioni Vi-jovi

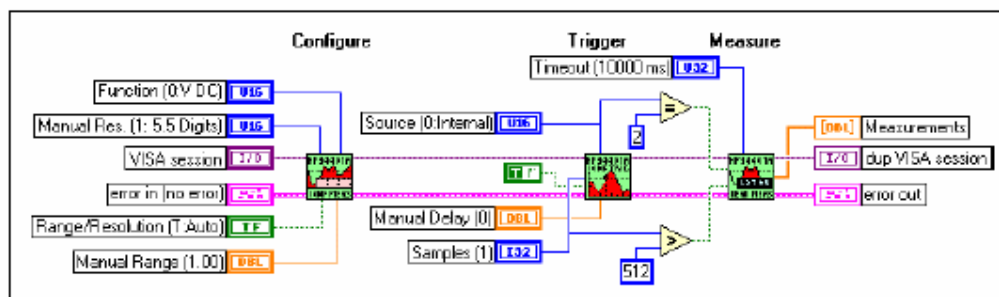
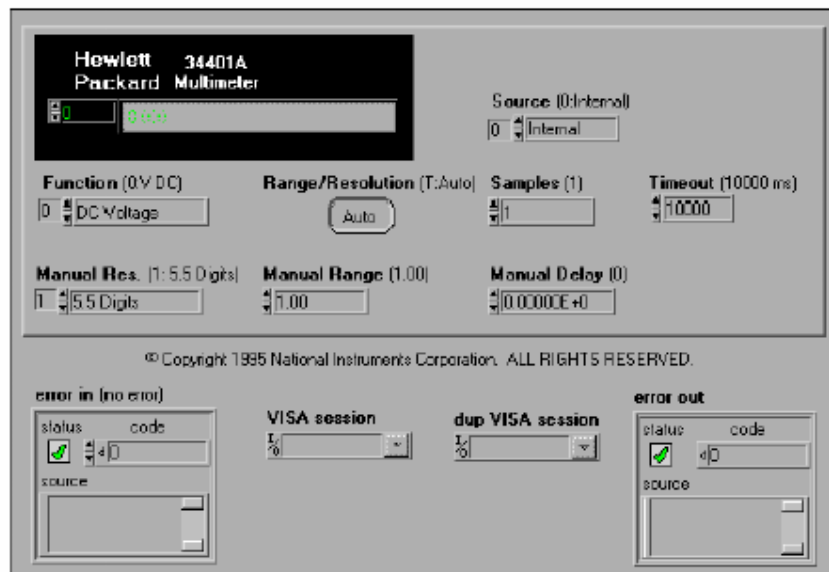
Aplikacioni VI se nalaze na najvišem nivou hijerarhije instrumentalnih drajvera. Ovi visoko-nivovski VI su pisani u LabVIEW blok dijagram izvornom kodu i kontrolišu najčešće korištene instrument konfiguracije i mjerenja pozivajući odgovarajuće VI-eve komponentnog nivoa. One demonstriraju visoko-nivovsku

test i mjernu funkcionalnost konfigurisući instrument za najčešće korišteni način rada, trigerovanja i uzimanja očitanih vrijednosti instrumenta.

Posto su aplikacioni VI-evi standardni VI-evi sa ikonama i konektorskim poljem (pane), oni se mogu pozivati iz bilo koje visoko-nivovske aplikacije, zahtjevajući jedinstven, mjerenju orjentirani interfejs sa instrumentom.

Za mnoge korisnike, aplikacioni VI-evi su jedini VI potrebni za kontrolu instrumenta.

Slika br. 2 pokazuje HP34401A instrument idemonstrira prednju ploču jednog aplikacionog VI.



Slika br. 2 Primjer HP34401A instrumenta

Initialize VI, je prva koju VI instrument drajvera poziva, uspostavlja komunikaciju sa instrumentom. Dodatno, može izvršiti bilo koje akcije neophodne da postavi instrument u stanje priključenog napajanja ili bilo koje drugo specifično stanje. Općenito, Initialize VI treba biti pozvana samo jedanput na početku aplikacionog programa.

Konfiguracione VI su kolekcija softwareskih rutina koje konfiguriraju instrument da bi izvršavao željenu operaciju. Mogu postojati mnogobrojne konfiguracione VI, zavisno od specifičnog tipa instrumenta. Nakon što su ove VI pozvane, instrument je spreman da izvrši mjerenja i stimulira sistem (ako generira neke valne oblike ili signale).

Action/status kategorija sadrži dva tipa VI-jeva. Action VI iniciraju ili završavaju test i mjerne operacije. Ove operacije mogu uključiti generiranje trigera ili generiranje stimulus signala.

Ove VI su različite od konfiguracionih VI pošto one ne mjenjaju setinge instrumenta, nego samo naredjuju instrumentu da izvrši neku akciju na bazi njegove tekuće konfiguracije.

Statusne VI dobijaju tekuće stanje instrumenta ili status operacija koje čekaju na izvršenje.

Data VI prenose podatke ka ili od instrumenta. Primjeri za ove su čitanje mjerene vrijednosti valnog oblika sa mjernog instrumenta i VI za downloadovanje valnih oblika ili digitalnih paterna ka instrumentu koji će ih generirati.

Utility VI izvršavaju niz operacija koje su pomoćne za najčešće korištene VI-jeve instrument drajvera. Ove VI uključuju većinu instrument drajver template Vi-jeva (mustre), kao što su :

reset, self-test, revision, error query, error message

a mogu uključiti i druge kastomizirane instrument drajver VI-jeve koje izvršavaju operacije kao što su :

kalibracija, pohranjivanje (storage), i recall setupa.

Close VI terminira softwaresku konekciju sa instrumentom i oslobadja resurse sistema. Općenito, Close VI treba biti samo jedanput pozvana na kraju aplikacionog programa i kada se završava komunikacija sa instrumentom.

Korisnik treba da obezbjedi da za svaki uspješni poziv (call) za Initialize VI postoji i uparena sa njim Close VI.

U protivnom će se nepotrebno zadržavati resursi sistema i neće biti vraćani Operativnom sistemu.

Vrste instrument drajvera

Postoje različite vrste instrument drajvera. Razlika medju njima nije toliko u tome kako ih koristimo , nego kako su implementirani. Tri tipa koja postoje su:

- LabView instrument drajveri
- VXI plug &play instrument drajveri
- IVI drajveri (interchangeable virtual instruments)

Redoslijed koji je naveden ilustrira i njihovu evoluciju kako su se razvijali u nekoliko posljednjih godina, mada su sve tri vrste drajvera i danas još uvijek raspoložive.

LabView drajveri se tako zovu jer su pisani upotpunosti sa LabView funkcijama. Ostale dvije vrste se skoro uvijek pišu u C progr. jeziku i imaju VI "wrappers" (omotnice) oko svakog C funkcionalnog calla. LabView instrument drajveri se lakše modifikuju I debugiraju nego druge vrste drajvera, i oni se lako konvertuju sa jedne hardwareske platforme PC-ja za drugu.

Ipak, mnogi od ovih drajvera su stariji i nisu u saglasnosti sa standardnim interfejsima.

Naprimjer, funkcije u drajveru za jedan tip multimetra mogu biti potpuno različite u drajveru za drugi tip multimetra.

LabView biblioteka instrument drajvera sadrži instrument drajvere za različite programabilne instrumente koji koriste GPIB, VXI ili serijske interfejse. Korisnik može koristiti ove drajvere onakve kakvi su. Međutim, pošto se ovi drajveri distribuiraju sa blok dijagram source kodom, korisnik ih može kastomizirati za svoje specifične aplikacije.

VXI plug&play drajveri su pisani u C donjeli su određenu konzistentnost u instrument drajvere, ali nisu pokrili neke aplikacije, kao što je testiranje uređaja kod proizvodnje.

IVI Fondacija je formirana da donese još više standardizacije u oblast gradnje instrument drajvera. Ovaj put su definirani za C jezik instrument specifični programski interfejsi.

To znači da korisnik može pisati program koji može raditi sa nekoliko različitih tipova osciloskopa od različitih proizvođača, bez da je potreban specijalni kod za svaki model u aplikaciji.

Da bi se prešlo sa jednog modela na drugi zahtjeva samo promjenu u konfiguraciji.

IVI Fondacija takodjer se posvetila i drugim problemima kao simulacija nedostajućih instrumenata i performansa.

Pošto su IVI drajveri na bazi C, oni imaju iste probleme kao i VXI plug&play drajveri kada je u pitanju njihovo korištenje u LabView. Naime u slučaju potrebe za modifikacijom, mora se koristiti C bazirani razvojni okružaj kao što je LabWindows/CVI, a nakon toga konvertovati drajver ponovo za korištenje u labView.

Ime resursa / Deskriptor instrumenta

Prije nego što možemo komunicirati sa instrumentom, moramo otvoriti komunikacioni link ka instrumentu sa Initialize VI u sklopu instrument drajvera.

Nakon završetka komunikacije sa instrumentom, možemo pozvati Close VI, i sve reference i resursi biće zatvoreni. Ako ne pozovemo Close VI, sve reference će se zatvoriti kada zatvorimo LabView.

Kada inicijaliziramo instrument, treba da znamo ime resursa (resource name) ili deskriptor instrumenta (Instrument Descriptor).

- *Resource* – VISA alias ime ili IVI logičko ime
- *Instrument descriptor* – Tačno ime i lokacija resursa u formatu

Interface Type [board index] :: Address:: INSTR.

Naprimjer, GPIB0 :: 2:: INSTR. je instrument deskriptor kada se koristi prva GPIB ploča da komunicira sa instrumentom sa adresom uređaja 2.

Treba koristiti Measurement & Automation explorer da se odrdi koji su resursi i adrese instrumenata raspoložive.

Korisnik može specificirati VISA Alias za ime resursa ili dekriptor instrumenta u VI instrument drajvera.

Error In/Error out klasteri

Manipulacije sa greškama unutar instrument drajver VI-jeva su slične kao i kod ostalih I/O u LabView. Svaki instrument drajver VI sadrži **Error In** i **Error Out** terminale za prenošenje klastera greške sa jedne Vi na drugu.

Klaster greške sadrži Boolean flag koji indicira da li se greška pojavila, broj za kod greške, i string koji sadrži lokaciju VI gdje se greška prvo pojavila.

Svaki instrument drajver je napisan tako da kada se greška pojavi , VI se ne izvršava. Informacija o grešci se prenosi na slijedeći VI , preko **Error Out** terminala.

Verifikacija komunikacije sa instrumentom

Da bi se verificirala komunikacija sa instrumentom i testirao tipični programatski rad instrumenta , potrebno je prvo koristiti **Getting Started** VI za specifičan instrument. U opštem slučaju, izuzev adrese instrumenta, sve ostale default vrijednosti će biti adekvatne, i ne treba ih mjenjati. Nakon postavljanja adrese i izvršenja VI, treba provjeriti da su sa instrumenta dobijeni neki razumni odgovori i da greška nije izvještena u klasteru greške. Najčešći razlozi za neuspjeh ove VI su:

- NI-VISA nije instaliran.
- Adresa instrumenta nije korektna
- Instrument drajver ne podržava tačan tip i model instrumenta kojeg mi koristimo.

Nakon verifikacije bazne komunikacije sa instrumentom koristeći Getting Started VI, korisnik će vjerovatno htjeti da kastomizira kontrolu instrumenta prema svojim potrebama.

Verifikacija VISA komunikacije

Ako se pokaže da VISA ne radi u okviru LabViewa , uključujući i instrument drajvere, potrebno je početi sa Vi-jem “ Find resource VI”. Ovaj VI se izvršava samostalno bez bilo kojeg drugog VISA VI-ja u blok dijagramu. Provesti debugiranje sa ovom VI dok se ne ustanovi razlog za prethodnu grešku.

Općenito o VISA

VISA je standardni I/O aplikacioni programski interfejs (application programming interface –API) , za programiranje instrumentacije. VISA može kontrolirati VXI, GPIB, PXI i serijske instrumente, šaljući odgovarajuće drajverske callove , zavisno od tipa instrumenta koji se koristi.

Tipovi poziva

(message –based communication versus register –based communication calls)

GPIB, serijski kao i neki VXI instrumenti koriste komunikaciju baziranu na poruci (message based communication). Message-based instrumentacija se programira sa visoko nivovskim ASCII stringovima karaktera. Instrument ima lokalni procesor koji parsira komandni string i setuje odgovarajuće registar bite da se izvrše željene funkcije.

Message-based instrumentacija se lako programira. Da bi se još više olakšalo, SCPI (standard commands for programmable instrumentation) standardizira ASCII komandne stringove koji se koriste za programiranje instrumenata. Svi SCPI instrumenti sa definiranom funkcijom se programiraju sa istim komandama. Umjesto da uči različite komandne poruke za svaki tip instrumenta od svakog specifičnog proizvođača, korisnik treba da nauči samo jedan set komandi. Najčešće message-based funkcije su

VISA read, VISA Write, VISA Assert trigger, VISA Clear, VISA Read STB

PXI i mnogi VXI instrumenti koriste komunikaciju baziranu na registrima (register based). Register –based instrumenti se programiraju na niskom nivou koristeći binarnu informaciju koja se direktno upisuje u kontrolne registre instrumenta. Prednost ovoga tipa komunikacije je brzina, pošto instrument ne mora više da parsira komandne stringove i konvertuje informaciju na nivo programiranja za registre.

Register-based instrumenti komuniciraju doslovno na nivou direktnih hardwareskih manipulacija nad registrima.

Najčešće register-based funkcije su :

VISA In, VISA Out, VISA Move In, VISA Move Out

Pisanje jednostavne VISA Aplikacije

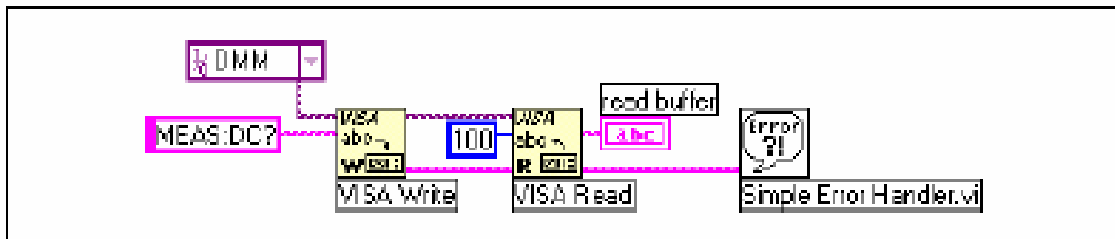
Za najjednostavnije instrumentalne aplikacije, potrebne su samo dvije VISA funkcije:

VISA Write i VISA Read

Primjer pokazan na narednoj slici je vrlo jednostavan, samo jedan VISA Write call i jedan VISA Read call. Instrument je specificiran koristeći “VISA Resource Name” konstantu. VISA Write funkcija će provjeriti da vidi da li je referenca već uspostavljena sa specificiranim instrumentom. Ako ne postoji referenca, referenca će biti automatski otvorena. Nakon toga, string MEAS:DC? se šalje ka instrumentu.

Kada se iščitava sa instrumenta, korisnik može jednostavno ožičiti “VISA Resource Name” izlaz od VISA Write funkcije ka VISA Read funkciji da bi se specificirao željeni instrument. Nakon toga korisnik može procesirati i prikazati vraćeni izlaz od VISA Read funkcije ako je to potrebno za njegovo mjerenje.

Nakon VISA Read slijedi “Simple Error Handler” VI da bi procesirala bilo koje greške koje bi se mogle pojaviti sa VISA funkcijama.



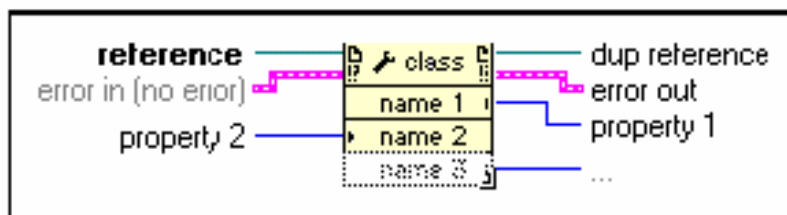
Slika br. 3 VISA primjer

Korištenje VISA osobina

VISA resursi imaju razne vrste osobina (atributa) sa vrijednostima koje mogu biti ili očitavane ili setovane u programu. U nastavku biće opisane neke od ovih osobina i kako ih koristiti.

Korištenje čvora osobina (property node)

Čvorovi osobina se koriste da čitaju ili setuju vrijednosti VISA osobina. Čvor osobine je pokazan na narednoj slici br. 4 :



Slika br. 4 Čvor osobina

Opaska : Čvor osobina je generički node koji također može biti korišten da se postave osobine ActiveX i VI servera.

Nakon postavljanja čvora osobina na blok dijagram, treba ožičiti VISA Sesiju sa referentnim ulaznim terminalom čvora osobina.

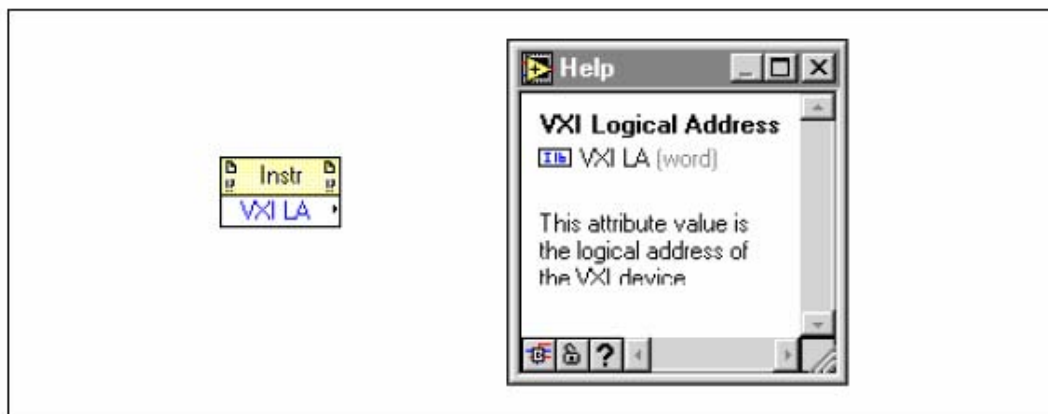
Čvor osobina sadržava jednostruki terminal osobine , kada se inicijalno postavi na blok dijagram. Međutim, može se povećati da sadrži onoliko terminala koliko je potrebno. Početni terminal na VISA čvoru osobina je read terminal. Ovo znači da vrijednost izabrane osobine u tom terminalu će biti read (čitanje). Ovo je indicirano sa malom strelicom koja pokazuje na desno na desnoj ivici terminala. Mnogi terminali mogu individualno biti promjenjeni sa read terminala na write terminal, klikajući sa desnim tasterom na osobinu koju želimo da promijenimo.

Opaska: Neke osobine su read only ili write only. Njihove vrijednosti se ne mogu postaviti.

Da bi se izabrala osobina u svakom terminalu čvora osobina, kliknuti na terminal proprety noda. Ovo će dati listu svih mogućih osobina koje se mogu setovati u programu. Broj različitih osobina pokazan u izborniku za selekciju itema u okviru VISA Proprety čvora i može biti limitiran promjenom VISA klase čvora osobina.

Da bi se promjenila VISA klasa , kliknuti desnim tasterom na VISA proprety čvor i izabrati “ **VISA Class**” . Nekoliko različitih klasa se može izabrati pod ovom opcijom , osim default INSTR klase, koja uključuje sve moguće VISA osobine. Ove klase ograničavaju prikazane osobine na one koje su važeće za izabranu klasu , umjesto za sve VISA osobine. Jedanput kada je sesija spojena na **Session** ulazni terminal čvora osobina, VISA Class je postavljena na klasu pridruženu sa tom sesijom.

Inicijalno, VISA osobine će izgledati korisniku unekoliko neobične. Koristiti Help mogućnosti LabView da se dobiju podatci o ovim osobinama i klasama, kao što je pokazano na narednoj slici br. 5 za VXI :



Slika br. 5

Postoje dva osnovna tipa VISA osobina: globalne osobine i lokalne osobine. Globalne osobine su specifične za resurs, dok lokalne osobine su specifične za sesiju. Naprimjer, VXI LA osobina je globalna osobina. Ona vrijedi za sve sesije koje su otvorene za taj resurs. Lokalna osobina je osobina koja može biti različita za individualne sesije specifičnog resursa. Primjer lokalne osobine je vrijednost za **timeout**. Neke od zajedničkih osobina za svaki tip resursa su pokazane u slijedećim listama:

Serial

Serial Baud rate – baud brzina za serijski port

Serial Data Bits - broj data bita korišten kod serijskog prenosa

Serial Parity - paritet korišten kod serijskog prenosa

Serial Stop Bits – broj stop bita korišten za serisjku transmisiju.

GPIB

GPIB Readdressing – specificira da li uređaj treba biti readresiran prije bilo kakve write operacije

GPIB Unaddressing - specificira da li uređaj treba biti deadresiran nakon operacije čitanja ili pisanja

VXI

Mainframe Logical Address – najniža logička adresa uređaja u istoj šasiji sa resursom

Manufacturer Identification – ID broj proizvođača iz konfiguracionih registara uređaja .

Model Code - kod modela uređaja iz konfiguracionih registara uređaja

Slot - slot u šasiji u koji je uplagovan uređaj.

VXI Logical Address – logička adresa uređaja

VXI memory Address Space - VXI adresni prostor korišten od strane resursa

VX Memory Address Base - veličina memorijskog regiona korištenog od strane resursa .

Postoji još mnogo drugih osobina osim ovih koje su izlistane. Postoje također osobine koje nisu specifične za dati tip interfejsa. **Timeout** osobina , koje je vrijeme koje se koristi u message –baziranim I/O operacijama, je jedan primjer takve osobine.(koristiti Help u okviru LabView da se dobije više informacija).

Korištenje VISA događaja (events)

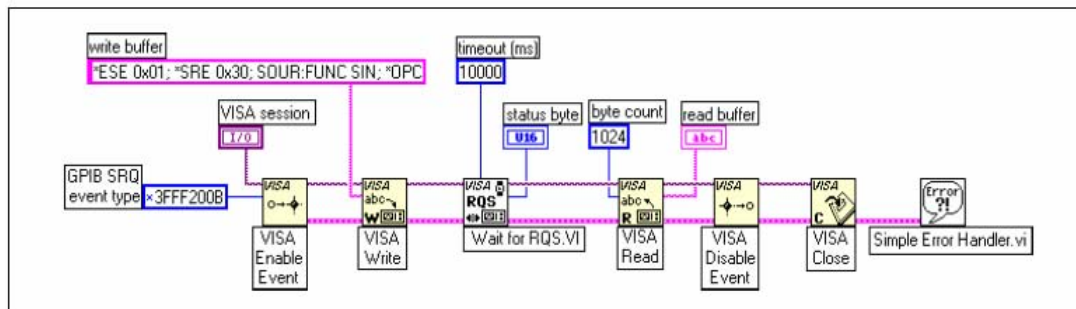
Događaj je sredstvo VISA komunikacije između resursa i njegove aplikacije. To je način da resurs obavjesti aplikaciju da se neki uslov pojavio koji zahtjeva akciju od strane aplikacije.

Primjeri različitih događaja su uključeni u narednim sekcijama.

Tipovi događaja

Primjer manipulacije sa GPIB SRQ Events

Naredna slika pokazuje blok dijagram kako rukovati GPIB Service Request (SRQ) događaje sa VISA.



Slika br. 6 Blok dijagram SRQ Events

VI omogućuje service request događaje a nakon toga upisuje komandni string na instrument. Od instrumenta se očekuje da odgovori sa SRQ kada je procesirao string. VI " Wait on Event Async" čeka do 10 sekundi za SRQ događaj da se pojavi.

Nakon što se SRQ pojavio, očitava se status bajt instrumenta sa "Read Status Byte " VI. Statusni bajt mora biti očitana nakon što se "GPIB SRQ events" desio, pošto kasniji SRQ događaji mogu da ne budu korektno primljeni. Konačno, odziv je očitana sa instrumenta i prikazan. "Wait on Event Async" je različit od regularnog "Wait on Event" VI, po tome što kontinualno zove "Wait on Event" sa timeoutom od 0 , da bi provjeravao (poll) na događaj.

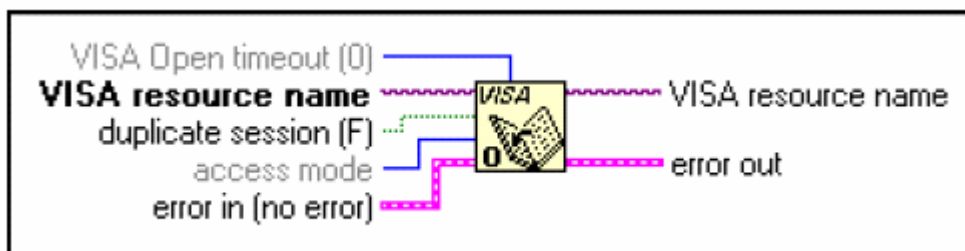
Ovo oslobadja vrijeme za druge paralelne segmente programa da se izvršavaju dok se čeka na događaj.

Napredne VISA funkcije

Otvaranje VISA sesije

Kao što je već prethodno diskutirano, kada aplikacija poziva VISA Read i/ili VISA Write, LabView provjerava da vidi da li je referenca već otvorena za specificirani instrument. Ako je referenca već otvorena, VISA call će koristiti tu referencu. Ako nema otvorene reference, VISA automatski otvara novu referencu. Korisnik može izabrati da eksplicitno otvori reference za korisnikov instrument koristeći VIS Open.

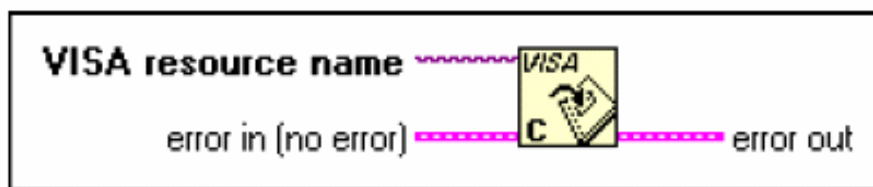
VISA Open funkcija je pokazana na slijedećoj slici br. 7 :



Slika br. 7 VI Open funkcija

Zatvaranje VISA sesije

Otvorena sesija sa VISA resursom koristi sistemske resurse unutar kompjutera. Da bi se korektno okončao VISA program, sve otvorene VISA sesije trebaju biti zatvorene. Da bi se ovo učinilo, koristiti VISA Close VI, koja je pokazana na narednoj slici br. 8 :



Slika br. 8 VISA Close VI

Ulaz VISA sesije ka VISA Close VI je sesija koja se treba zatvoriti. Ova sesija izvorno dolazi od terminala izlazne sesije na "VISA Open" VI , ili sa bilo koje druge VISA VI. Ako sesija nije zatvorena kada se VI izvršila , ostaće otvorena.

Opaska:

Ako je VI abortirana kada se VI debugirala, VISA sesija nije automatski zatvorena . Korisnik može koristiti " Open VISA Session Monitor" VI , koja je na raspolaganju u direktoriju vi.lib\utility, da asistira pri zatvaranju takvih sesija.

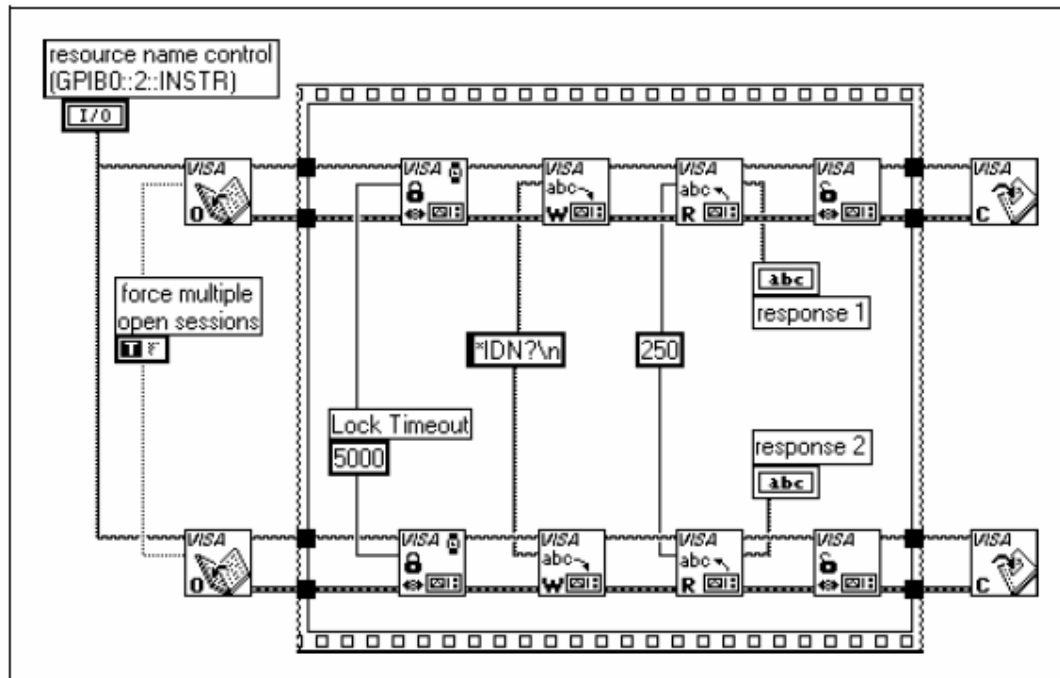
Locking (Zaključavanje)

VISA uvodi zaključavanje (locks) za kontrolu pristupa resursima. Sa VISA, aplikacije mogu simultano otvoriti višestruke sesije ka istom resursu i mogu pristupiti resursu istovremeno putem ovih različitih sesija.

U nekim slučajevima, aplikacije koje pristupaju resursu, moraju ograničiti druge sesije od pristupa tom resursu. Naprimjer, jedna aplikacija može trebati izvršenje write i read operacije u jednom koraku, tako da ni jedna druga aplikacija se ne može početi izvršavati između write i read operacija.

Aplikacija može zaključati resurs prije pozivanja write operacije i otključati ga nakon read operacije, da bi ih izvršila u jednom koraku.

VISA mehanizam zaključavanja traži arbitažu pristupa resursima na individualnoj bazi. Ako sesija zaključa resurs, operacije koje su pozvane od drugih sesija su servisirane ili vraćene sa greškom zaključanog resursa, zavisno od operacije i tipa zaključavanja koji se koristi.



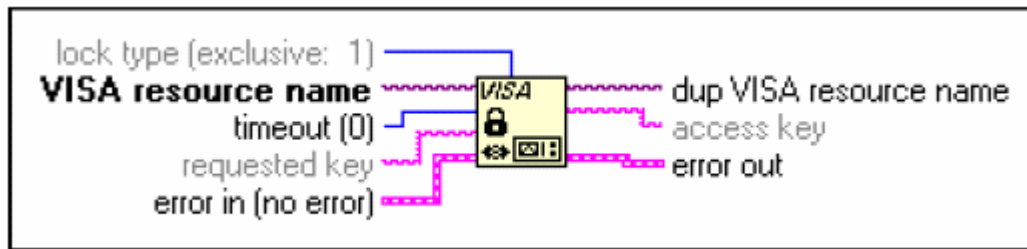
Slika br. 9 VISA Lock Async VI

“VISA Lock Async “ VI pokazana na prethodnoj slici, i raspoloživa u *Functions>>Instrument I/O>>VISA Advanced* paleti , otvara dvije sesije ka istom resursu i izvršava upit (query) , na svakoj od njih. Ovaj primjer koristi zaključavanje (lock) , da garantira da write/read parovi se dešavaju u očekivanom redoslijedu i ne prekrivaju se. Lock je oslobođen nakon što je write/read sekvenca kompletirana, i time dozvoljavajući da se nastavi i druga staza izvršavanja programa unutar sesije.

Nema garancije o tome koja će sesija prva primiti lock mehanizam. Zaključavanje je korisno u slučajevima kada više od jedne aplikacije može pristupiti istom resursu, ili gdje multipli moduli mogu otvoriti multiple sesije ka istom resursu čak i u okviru iste aplikacije.

Djeljena zaključavanja (shared locking)

Mogu postojati slučajevi kada korisnik želi da zaključa pristup resursu ali selektivno dijeli ovaj pristup. Naredna slika pokazuje **Lock Vi** sa Complex Helpom.



Slika br. 10 Ikona VISA Lock funkcije

Lock tip defaultira na “exclusive”, ali korisnik može ga setovati da bude dijeljen (shared). Nakon toga može ožičiti string ka **requested key**, da bude potreban password za drugu aplikaciju da bi pristupila resursu.

Medjutim, VI doznačava jedinicu u **access key**, ako to korisnik ne zatraži. Nakon toga se može koristiti ovaj ključ da se pristupi zaključanom resursu.

Tehnike manipulacije stringovima

Do sada smo vidjeli da većina problema LabView instrument drajvera može biti riješena bez modificiranja koda instrument drajvera. Medjutim, u nekim situacijama, modifikacija koda je nužna. Ova sekcija će opisati neke fundamentalne metode komunikacije sa instrumentom, i uvešće neke često korištene funkcije u LabView instrumentalnim drajverima.

Kako instrumenti komuniciraju

Podsjetimo se da su dva glavna tipa komunikacije sa instrumentom *message-based* i *register-based*.

Ne postoji standard za register-based komunikaciju sa instrumentom. Svaki uređaj radi nezavisno, i manual za taj instrument je najbolji izvor da vidimo kako ćemo ga isprogramirati.

Gradnja Stringova

Kada komuniciramo sa message-based instrumentom, moramo formatirati i izgraditi korektne komandne stringove, jer instrument neće korektno odraditi šta od njega zahtjevamo niti vratiti odgovor.

Tipično, komandni string je kombinacija tekst i numeričkih vrijednosti. Pošto instrumenti zahtjevaju da cijeli komandni string bude tekst, korisnik mora pronaći način da konvertuje numeričke vrijednosti u tekst i pridoda ih na ostatak komandnog stringa.

Funkcija “Format into String” se može koristiti da se izgrade komadni stringovi koji su potrebni da ih pošaljemo instrumentu. Ova funkcija dozvoljava da se uzme jedan inicijalni string i apendiraju drugi stringovi ili numerički tipovi podataka na njega.

Odklanjanje zaglavlja (removing headers)

Ova sekcija opisuje kako ćemo čitati informaciju koja je vraćena od message-based instrumenta. Instrument manual daje opis koja zaglavlja (header) i repove (trailer) možemo očekivati kod prenosa podataka.

Većina instrumenata vraća podatke sa dodatnim informacijama pridodatim bilo kao header ili trailer. Zaglavlje (header) obično sadrži informaciju kao što je broj tačaka podataka koji se šalje od instrumenta, ili settinge instrumenta.

U nekim slučajevima, repna (trailer) informacija sadrži jedinice mjerenja ili neke druge setinge (postavljenja) instrumenta, na kraju stringa podataka. Korisnik mora prvo otkloniti ove header i trailer informacije , prije nego ih može prikazati ili analizirati vraćene podatke od instrumenta.

Razmatrajmo primjer pokazan na prethodnoj slici na kojoj string sadrži 6 bajtni header (zaglavlje), tačke podataka, i 2 bajtni rep (trailer). Korisnik može koristiti Funkciju "String Subset" , koja je na raspolaganju u paleti **Functions>>String** da otkloni header kao što je to gore pokazano.

Subset stringa će vratiti substring od ulaznog stringa počevši od ofseta i zadržavajući broj karaktera koji je definiran parametrom : length (dužina). Parametri length i offset moraju biti skalari. Offset od 6 će otkloniti header od CURVE<SPACE>, a length od stringa je totalna dužina stringa odgovora instrumenta, umanjena za dužinu headera.

Opaska: Za stringove, vrijednosti ofseta počinju od nule, kao i indeksi kod polja (arrays) koji počinju od nule.

Prenos valnih oblika (waveforms transfers)

Pored informacija u headeru i traileru, podatci od instrumenta mogu biti vraćeni i u raznim formatima. Manual za instrument opisuje koji formati su na raspolaganju i kako korisnik može konvertovati i svaki od njih u upotrebljive podatke. Formatu koje ćemo diskutirati u nastavku uključuju :

ASCII, 1-byte binarni i 2 -byte binarni format.

ASCII valni oblici (waveforms)

Ako se podatci sa instrumenta šalju u ASCII formatu, korisnik ih može posmatrati kao string karaktera. Medjutim, ako je potrebna numerička manipulacija nad podacima, korisnik mora prvo konvertovati string podatke u numeričke podatke. Kao primjer, posmatrajmo valni oblik sastavljen od 1024 tačke, i svaka tačka ima vrijednost izmedju 0 i 255.

Koristeći ASCII kodiranje, trebaće nam maksimalno 4 bajta da predstavimo podatak, (maksimalno 3 bajta za numeričku vrijednost , i 1 bajt za koma separator). Dakle biće nam potrebno maksimalno 4096 bajta (4 * 1024) , plus header i trailer bajti da bi predstavili valnioblik kao ASCII string.

Drugi primjer je pokazan na prethodnoj slici sa otklonjenom header informacijom i podacima :

- 10.2 , 18.3, 8.91, 1.0, 5.5.

Možemo koristiti VI “Extract Numbers” da konvertujemo ASCII string u numeričku array varijablu. Extract Numbers VI je primjer VI koja je na raspolaganju u helpu *Search Examples*.

Ova VI nalazi sve brojeve u datom ASCII stringu i stavlja ih u **Single Precision Array** brojeva. Ne-numerički rezdjelivači (delimiter) kao naprimjer koma (,) kolona (:), itd. se predpostavljaju i svi formati koji su gore pomenuti su prepoznati. Bilo koji karakteri na početku ASCII stringa su ignorirani, tako da nema potrebe da skidamo (strip off) informaciju headera, kada koristimo “Extract numbers” VI. Gornji primjer pokazuje kako možemo izvaditi (extract) pet vrijednosti iz stringa i smjetiti ih u array brojeva.

Sada ove numeričke vrijednosti možemo iscrtavati ili ih koristiti u bilo kojim algoritmima numaričke analize.

1 BYTE Binary Waveform

Neki instrumenti nemaju opciju slanja podataka u ASCII formatu, ili iz razloga performansi, svi podatci za valni oblik se šalju u binarnoj formi. Ne postoje standardni binarni formati, tako da je potrebno tačno naći kako su podatci pohranjeni koristeći manual za instrument.

Jedan čest binarni format je je 1-byte binary. Sa ovim tipom kodiranja podataka, svaka pojedinačna vrijednost je konvertovana u 8 bitnu binarnu vrijednost prije nego se pošalje.

Kada čitamo 1-bajtnu binarnu podatke sa basa, on se dobija kao string karaktera. Medjutim, karakteri se ne pojavljuju da imaju ikakvu vezu sa očekivanim podacima. Binarni brojevi se interpretiraju kao ASCII karakter vrijednosti i odgovarajući karakteri se prikazuju. Neki primjeri su pokazani u gornjoj tabeli.

Ako je vrijednost od 65 poslata kao jedan podatak, mi bi očitali karakter **A** sa basa. Promjetimo da za vrijednost od 13 , nema predstavljivog na printeru ASCII karaktera, 13 je nevidljivi carriage return (vrćanja na početak reda) kontrolni karakter.

Korisnik može prikazati ove nevidljive karaktere u string indikatoru u LabView , ako bude izabrao display sa: ‘\’ **Codes Display**. Naprimjer, string indikator u vrhu pokazan na gornjoj slici , prikazuje vrijednosti u default **Normal Display** i mi ne možemo vidjeti treći karakter.

Medjutim, ako kliknemo sa desnim tasterom na taj string indikator, i izaberemo ‘\’ **Codes Display**., iz menija, vidjećemo carriage return karakter kao” \r “ , kao što je pokazano u drugom string indikatoru.

Omogućavajući ‘\’ **Codes Display** na LabView stringu, korisnik može sada vidjeti karaktere koji su prije možda bili nevidljivi. Medjutim, moramo još uvijek

konvertovati binarni string u numerički array da bi iscrtali ili proveli matematske operacije nad podacima.

Predpostavimo da instrument šalje binarni string koji sadrži 1024 1-bajtno binarno kodirane vrijednosti kao što je gore pokazano. Taj valni oblik bi zahtijevao samo 1024 bajta plus informacije za header ili trailer. Koristeći binarno kodiranje, mi ćemo trebati samo 1 bajt da predstavimo vrijednost podatka, pod pretpostavkom da je svaka vrijednost predstavljena kao 8 bitna cjelobrojna vrijednost bez znaka (unsigned 8 bit integer).

Konvertujući binarni string u numeričku varijablu polja (array), je malo kompleksniji nego konvertujući ASCII string. Moramo prvo otkloniti svu header i trailer informaciju, koristeći funkciju "String Subset" koju smo ranije opisali. Nakon toga treba konvertovati preostali data string u array varijablu integera, koristeći funkciju "String to Byte Array", koja je na raspolaganju u paleti :

Functions>>String>>String/Array/Path Conversion.

Opaska: Koristeći binarne podatke, bolje je izvaditi podatke koristeći veličinu podataka nego tražiti prvi karakter iz trailer informacije, pošto je moguće da "search character" može također biti sadržan kao dio binarnih vrijednosti.

2-Byte Binary Waveforms

Treći format podataka je 2-byte binary. Kada su podatci u 2 bajtnom binarnom formatu, onda je binarno kodiran i poslat kao ASCII karakter kao i kod 1 –bajt binarnog formata.

Medjutim, 16 bitni podatci (ili dva ASCII karaktera) predstavljaju svaku pojedinačnu vrijednost podatka. Mada ovaj format koristi dva puta više memorijskog prostora nego 1-bajtni binarni podatci, još je uvijek efikasnije pakovanje nego kod ASCII formatiziranih podataka.

Kao primjer, posmatrajmo osciloskop koji prenosi podatke valnog oblika u binarnoj notaciji. Za ovaj primjer, valni oblik se sastoji od 1024 tačke podataka gdje svaka vrijednost je 2 bajtni integer sa predznakom. (2 byte signed integer). Prema tome, cijeli valni oblik će zahtijevati 2048 bajta plus 5 bajta headera i 2 bajta trailera.

Uklonimo 5 bajtni header i uzmimo narednih 2048 bajta. Zatim koristeći **Type Cast** funkciju, koja je na raspolaganju u paleti :

Functions>>Advanced>>Data Manipulation

možemo konvertovati string valnog oblika u varijablu polja sa 16 bitnim integerima.

Redoslijed bajta (Byte Order)

Kada se podatci prenose u 2-byte binarnom formatu, važno je znati redoslijed bajta koje primamo od instrumenta. Dvo bajtna kombinacija **qH** ima odgovarajuću cjelobrojnu vrijednost 29.000, ali i suprotnom redoslijedu bajta tj. **Hq** ta vrijednost će biti 18.545.

Ako primimo najveći bajt prvi, moramo obrnuti redoslijed bajta prije konverzije u cjelobrojnu vrijednost. Posmatrajmo ponovno gornji primjer. Ovi 2 bajtni podatci

valnog oblika imaju istu veličinu i sadrže iste informacije u headeru i traileru, ali su podatci poslani sa najvećim bajtom kao prvim.

Da bi dobili korektne vrijednosti i iscrtani valni oblik, možemo koristiti funkciju "Swap Bytes", koja je na raspolaganju u paleti:

Functions>>Advanced>>Data Manipulation

da bi obrnuli viši bajt i niži bajt za svaki pojedinačni podatak.

Instrument drajver komponentni VI-evi

Aplikacioni VI se grade od skupa niže-nivovskih instrument drajver funkcija koje se zovu komponentni VI-evi. Za razliku od aplikacionih VI-eva (koji predstavljaju samo podskup instrumentalnih karakteristika), komponentni VI-evi su modularno organizirani i sadrže sveukupnu konfiguraciju instrumenata i mjerne mogućnosti. Komponentni VI-evi se mogu podijeliti u šest kategorija:

initialize, configuration, action/status, data, utility i close.

Svi LabVIEW instrument drajveri trebaju imati jednu **initialize** VI. To je prva instrument drajver VI koja se poziva, i ona uspostavlja komunikaciju sa instrumentom. Ona također može izvršiti i ID (identity) zahtjev kao i postaviti instrument bilo u default stanje uključenog napajanja ili u neko drugo specifično stanje.

Konfiguracione VI su skup softwareskih rutina koje konfiguriraju instrument da bi izvršavao željenu operaciju. Obično postoji više konfiguracionih VI, zavisno od kompleksnosti instrumenta. Nakon što su ove VI pozvane, instrument je spreman da uzme mjerenja ili pak da stimuliše sistem (ako je riječ o programabilnom izvoru signala).

Action/status kategorija VI sadrži dva tipa VI-eva. **Action** VI prouzrokuju da instrument inicira ili okonča operaciju testa/mjerenja kao što je naprimjer iniciranje trigerskog sistema ili generiranje stimulnog (podsticajnog) signala. Ovi VI se razlikuju od konfiguracionih VI jer oni ne mjenjaju postavljene vrijednosti instrumenta, nego samo naredjuju instrumentu da izvrši akciju zavisno od trenutne konfiguracije instrumenta.

Status VI dobijaju tekući status (stanje) instrumenta ili pak status operacija koje čekaju izvršenje. Specifične rutine u ovoj kategoriji kao i stvarne operacije koje one izvršavaju su ostavljene razvojnom inženjeru da odluči, ali se obično kreiraju prema potrebi kako to zahtjevaju druge funkcije.

Data VI (VI-evi podataka) uključuju VI-eva za prenos podataka ka i od instrumenta. Primjeri ovih su VI za očitavanje mjerene vrijednosti ili valnog oblika sa mjernog instrumenta, VI-evi za downloading (slanje sa PC na kojem se izvršava VI ka programabilnom instrumentu) valnih oblika ili digitalnih uzoraka ka mjernom uređaju itd. Specifične rutine u ovoj kategoriji zavise od instrumenta i ostavljene su na izbor inženjeru koji razvija instrument drajver.

Utility VI mogu izvršiti niz operacija koje su pomoćne za većinu najčešće korištenih instrument drajver VI-eva. Ove VI uključuju većinu template (osnovnih uzoraka) instrument drajver VI-eva kao što su :

reset, samo-testiranje, revizija, upit kod greške (error query),

a mogu uključiti i druge standardne rutine kao što su kalibracija ili pohranjivanje ili pozivanje setup-a instrumenta.

Svi LabVIEW instrument drajveri trebaju uključiti close VI. Ova VI završava softwaresku vezu sa instrumentom i de-alocira (oslobadja) resurse sistema (memorija, itd.).

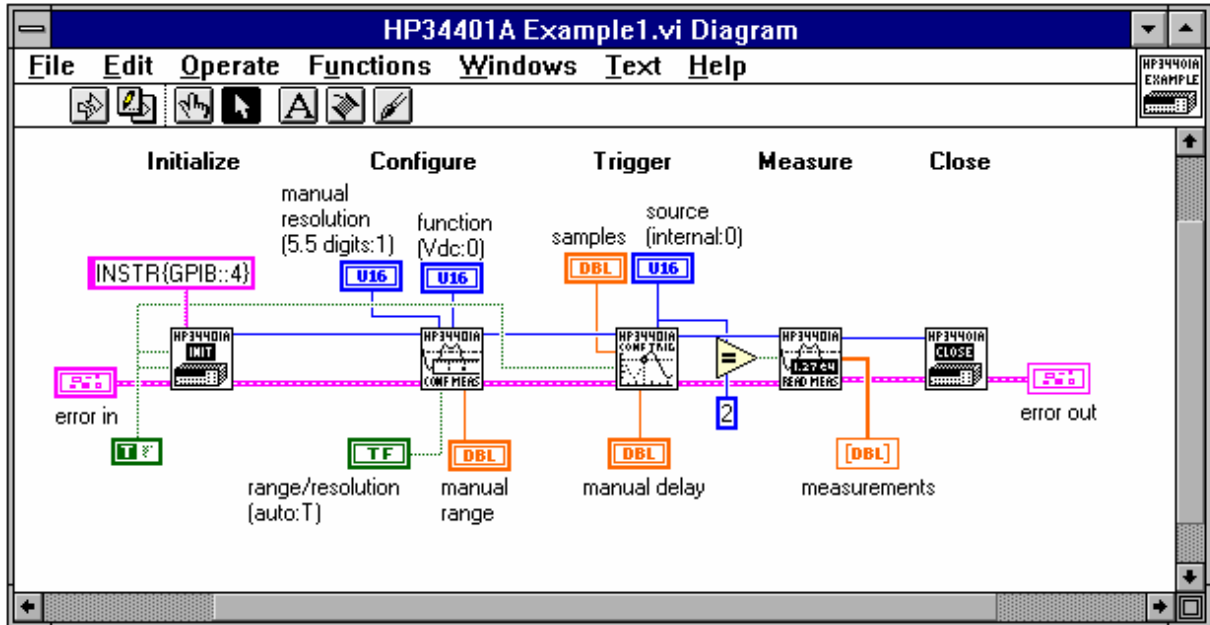
Svaka od ovih kategorija sa izuzetkom initialize i close, sadrži nekoliko modularnih VI. Najkritičniji dio posla kod razvoja instrument drajvera je kod početnog dizajna i organizacije komponentnih VI-eva. Specifične rutine u svakoj kategoriji mogu se pojaviti bilo kao template (uzorci) VI ili kao specifični VI specificirani od razvojnog inženjera. LabVIEW instrument drajver uzorci sadrže običajene VI kao sto su:

initialize, close, reset, self-test, revision query, i error-query,

koje korisnik može modifikovati i koristiti u njegovom instrument drajveru. Ovi VI su opisani detaljnije kasnije. Preostali VI, koji se nazivaju VI koje specificira razvojni inženjer, izvršavaju konkretne operacije sa instrumentom, kako ih je specificirao razvojni inženjer. Specifične kategorije komponentnih VI koje treba uključiti u drajver zavise od jedinstvenih mogućnosti instrumenta.

Koristeći interni dizajn model kako je naprijed opisano, korisnik može lako kombinirati instrument drajver VI-eve da kreira aplikacione programe. U slučaju kada uključeni aplikacioni VI nije optimiziran za specifičnu aplikaciju, korisnik može kreirati nove virtuelne instrumente kombinujući komponentne VI prema potrebi. Nadalje, on može optimizirati komponentne VI dodavanjem ili oduzimanjem kontrolnih elemenata sa panela i modifikujući dijagrame.

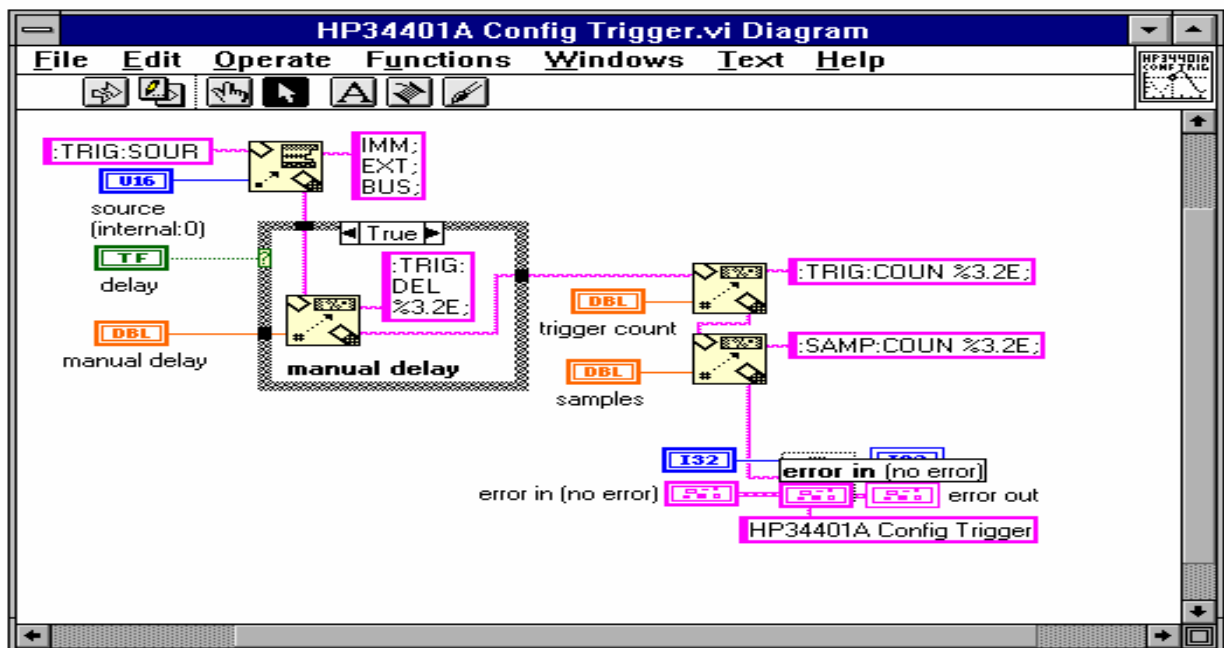
Naredna slika br. 11 pokazuje kako instrument drajver VI za HP 34401A DMM se programski koriste u dijagramu aplikacije VI HP34401A.



Slika br. 11 SubVI-i u Dijagramu primjera HP34401A

U blok dijagramu, instrument drajver komponente VI-a, ugradjenih u LabVIEW primitive kao i u VISA Transition biblioteku VI-a, izvršavaju funkcije kao što su: management uređaja, standardni I/O instrumenta i obradu grešaka.

Kao što se vidi sa naredne slike br. 4, niz komandi se ožičava u VISA Transition biblioteku VI, **Instr Write**. Ova VI izvršava neophodni I/O, provjerava za greške, i ažurira odgovarajuće indikatore grešaka. Ovi VI će detaljnije biti analizirani u poglavlju o VISA Transition biblioteka VI-eva.



Slika br. 12 Konfiguracioni triger dijagram za HP34401A

Idealni instrument drajver radi upravo ono što korisnik treba, ni više ni manje od toga. Kako se povećava broj programabilnih funkcija instrumenta, tako raste i potreba za dizajnom modularnog instrument drajvera, pošto sve kontrolne komponente ne mogu biti postavljene na samo jedan panel. S druge pak strane, instrument drajveri koji bi sadržavali stotine VI od kojih svaki sadrži samo po jednu komandu instrumentu, ne pomažu značajnije korisniku, koji nema iskustva sa instrumentalnim pravilima vezanim za redosljed komandi i njihovu interakciju. Modularni dizajn pojednostavljuje zadatak upravljanja instrumentom a također pojednostavljuje probleme oko modifikovanja VI-eva da bi se zadovoljili specifični zahtjevi.

Gradnja Instrument drajver VI-eva

Idealno, dizajner bi trebao da definise ukupnu strukturu drajvera prije nego sto počne da gradi VI-eve za određene funkcije. U stvarnosti, pak, struktura se radja u hodu kako dizajner projektuje, testira i modifikira VI-eve.

Preporuka za njega je da skicira ukupnu hijerarhiju, zatim gradi jednu po jednu VI, imajući uvijek na umu kako svaka od njih se uklapa u ukupnu hijerarhiju. On treba da odredi da li VI izvrsava svoju funkciju, i kako interaktira sa drugim VI-evima u okviru drajvera kojeg razvija.

Pogrešno je kruto se držati početne hijerarhije, ako se pokaže da ista nije praktična.

Nadalje, on treba da testira VI-eve koje je razvio zajedno sa instrumentom, da bi odredio da li hijerarhija ili VI-evi unutar hijerarhije zahtjevaju modifikaciju.

Kroz proces gradnje VI-eva, eksperimentiranja sa instrumentom i modifikacija, razvije se konzistentna struktura za drajver.

Da bi se pojednostavio zadatak kreiranja instrument drajvera, razmotrimo korištenje bottom-up pristupa dizajnu (odozdo - na gore).

Počnimo sa familijarizacijom (upoznavanjem) sa radom instrumenta. Zatim isčitajmo pažljivo manual za rad sa instrumentom. Naučimo kako koristiti instrument interaktivno, prije nego što počnemo bilo kakvo programiranje. Koristimo instrument u konkretnim šemama povezivanja (set-upa) da bi se steklo praktično iskustvo. Donesimo odluku koje kontrolne funkcije instrumenta su najpovoljnije za programabilno korištenje.

Nakon što smo naučili da koristimo interaktivno instrument, potrebno je da nastavimo sa analizom programabilnih funkcija instrumenta. Proletimo kroz skup instrukcija da bi ustanovili koje komande i funkcije se mogu programirati.

Počnimo razvoj strukture drajvera gledajući koje komande su korištene da bi se izvršila neka pojedinačna funkcija ili aktivnost sa instrumentom.

Modularni drajver će sadržavati individualne subVI-eve za svaku od funkcija. Tabela 1 izlistava HP34401A instrument drajver VI-eve i odgovarajuće sekcije manuala (priručnika) za njega.

Tabela 1 Poredjenje Sekcija u priručniku sa VI hijerarhijom

Virtualni instrument HP34401A Initialize	Sekcija u priručniku Input/output konfiguracija *IDN? *RST
HP34401A Config Measurement	Mjerna konfiguracija AC filter Autozero Funkcija Ulazni otpor Vrijeme integracije Opseg (range) Rezolucija
HP34401A Config Trigger	Operacije trigerovanja Očitavanje hold praga Broj uzoraka po trigeru Trigger delay (kasnjenje trigera) Izvor trigera
HP34401A Config Math	Matematske operacije Matematska funkcija Matematski registri
HP34401A Read Measurement	Očitavanje mjerenja Koristenje "Init" i "Fetch"
HP34401A System Controls	Operacije sistema Rad beepera (zvučno upozorenje) Display načini rada i prikazivanja

Prednja ploča (front panel) instrumenta

Svaki VI u instrument drajveru treba sadržavati prednji panel koji grupiše sve neophodne komande da bi izvršio funkcije VI. Naprimjer, **Configure measurement** VI bi sadržavala samo one komande neophodne da se postavi instrument u mjernu konfiguraciju; on ne bi trigerovao instrument niti konfigurirao bilo koju drugu karakteristiku instrumenta. Dizajner treba konstruirati dodatne VI da bi izvršio ove funkcije sa svakim prednjim panelom koji sadrži potrebne komande. U ovom stilu, rad kompleksnih instrumenata sa nizom opcija i načina rada se značajno pojednostavljuje zbog modularnosti VI-eva.

Odluka o tome koje komande postaviti na prednji panel VI je najveći izazov sa kojim se suočava dizajner instrument drajvera. Srećom, informacija o organizaciji panela je često raspoloživa u priručnicima o instrumentu ili sa samog instrumenta za koji se dizajnira VI.

Često, u dobro organizovanim priručnicima može se naći grupiranje komandi u sekcijama kao što su: konfigurisanje, trigerovanje (okidanje, start mjerenja), očitavanje mjerenja, što sve može poslužiti kao model za hijerarhiju drajvera.

Nadalje, dizajner može analizirati kako je Proizvodjač instrumenta grupirao komande na prednjem panelu instrumenta, i pri tome svaka od kontrolnih grupa se može lijepo prevesti u drajverske subVI module.

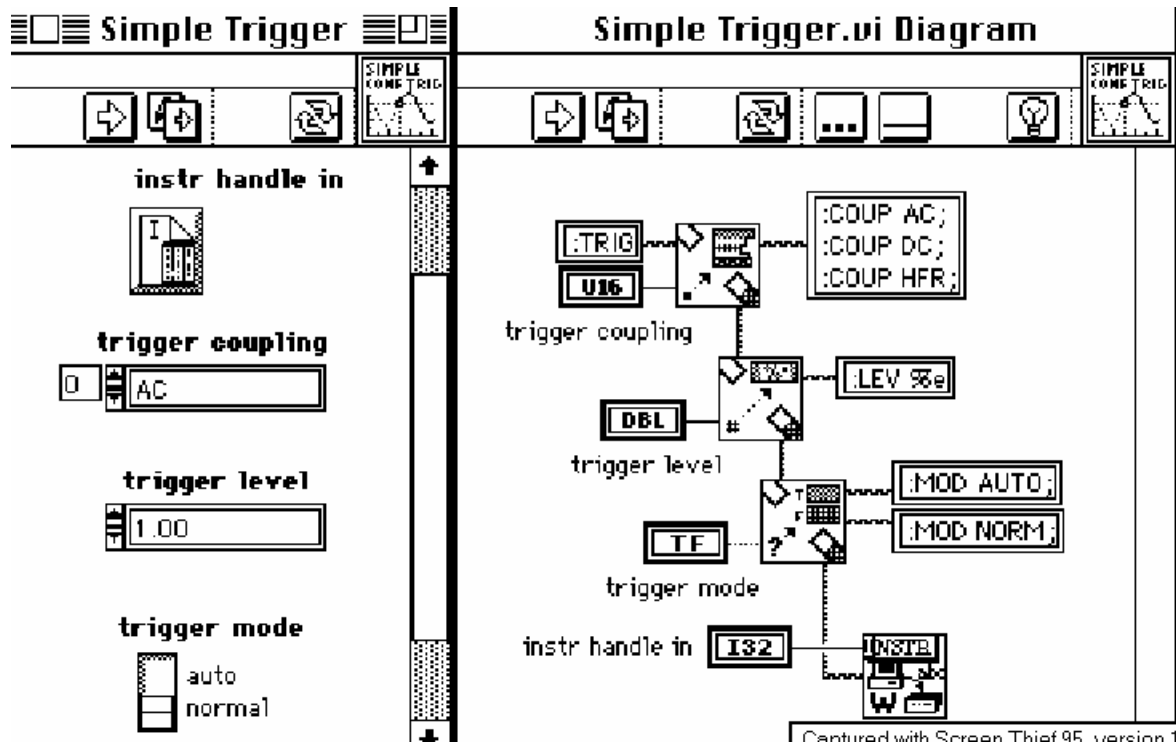
Jedanput kada je dizajner odlučio koje komande da objedini da bi formirao jednu subVI instrument drajvera, on mora odlučiti o tome koji stilovi unosenja komandi najbolje predstavljaju komande instrumenta za koji razvija VI. Tipično, komande instrumenta se mogu kategorizirati u četiri tipa stilova:

Bulov, digitalni, numerički, tekst-prsten numerički ili string (niz).

Naprimjer, bilo koja komanda instrumenta koja ima dvije opcije (napr. **TRIG MODE: Auto| NORMAL**) može se predstaviti na prednjem panelu kao Bulov prekidač (boolean switch). U ovom slučaju, dizajner bi označio prekidač kao *trigger mode* i dodao bi slobodnu oznaku (labelu) pokazujući opcije: auto ili normal. Za komande koje imaju diskretan broj opcija (kao što su **TRIG: COUP: AC | DC | HFREJ**), dizajner treba koristiti tekst-prsten radije nego digitalni numerički stil, pošto prstenasti tekst označava svaku numeričku vrijednost sa komandom koju ona predstavlja.

Svaka komanda koja zahtjeva numerički parametar čija vrijednost varira u širokom opsegu i ne može biti bolje predstavljena prstenom može biti predstavljena digitalnim numeričkim stilom. Konačno, komande koje zahtjevaju ASCII karaktere (kao što je napr. ime instrumenta), mogu biti predstavljene na prednjem panelu sa string ulaznom komandom. Dakle ova četiri tipa komandi: **Bulov, numerički, tekst prsten i string**, su sve ono što je potrebno dizajneru da predstavi većinu komandi instrumenata, na prednjem panelu VI.

Nadalje, postoje blok dijagram string ikone specijalno dizajnirane za korištenje sa ovim komandama koje pojednostavljaju formatiranje stringa i dodaju instrument komande u komandne poruke, kao što će to biti kasnije detaljnije analizirano u poglavlju o Blok dijagramu. Na narednoj slici br. 13 pokazane su ove komande i čvorovi dijagrama su korišteni da bi se kreirala jednostavna trigger konfiguracija VI.



Slika br. 13 Jednostavni triger VI koji koristi standardne kontrolne i string čvorove

Osim komandi potrebnih da bi radio instrument, prednji panel mora imati takodjer slijedeće zahjevane komande:

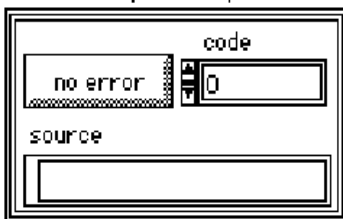
instr handle in



ins
ide
will
cor

instr handle in (ulazna tačka instrumenta) ulaz je jedinstveni identifikator za I/O sesiju uređaja (izuzev za **initialize VI**). Ona identificira uređaj sa kojim će VI komunicirati i prenosi sve neophodne konfiguracione informacije koje su potrebne da bi se izvršio I/O.

error in (no error)

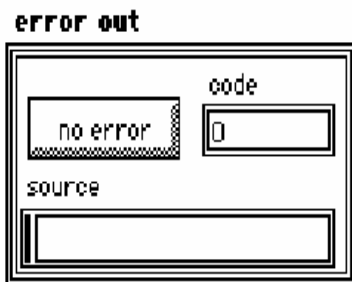


error in opisuje uslove postojanja greške koji se pojavljuju prije nego se ova VI izvrši. Default ulaz ovog skupa je no error (nema greske). Greška u skupu sadrži slijedeće parametre:

status je TRUE (tačan) ako se greška pojavila. Ako je status TRUE ova VI ne vrši nikakvu operaciju. Umjesto toga, šalje vrijednost greške u klasteru direktno u klaster **error out**.

code je kod greške pridružen greški. Vrijednost 0 znači da nema greške. (pogledati kasnije u sekciji ERROR Kodovi za opis mogućih kodova greški).

source je ime VI koja je proizvela gresku.



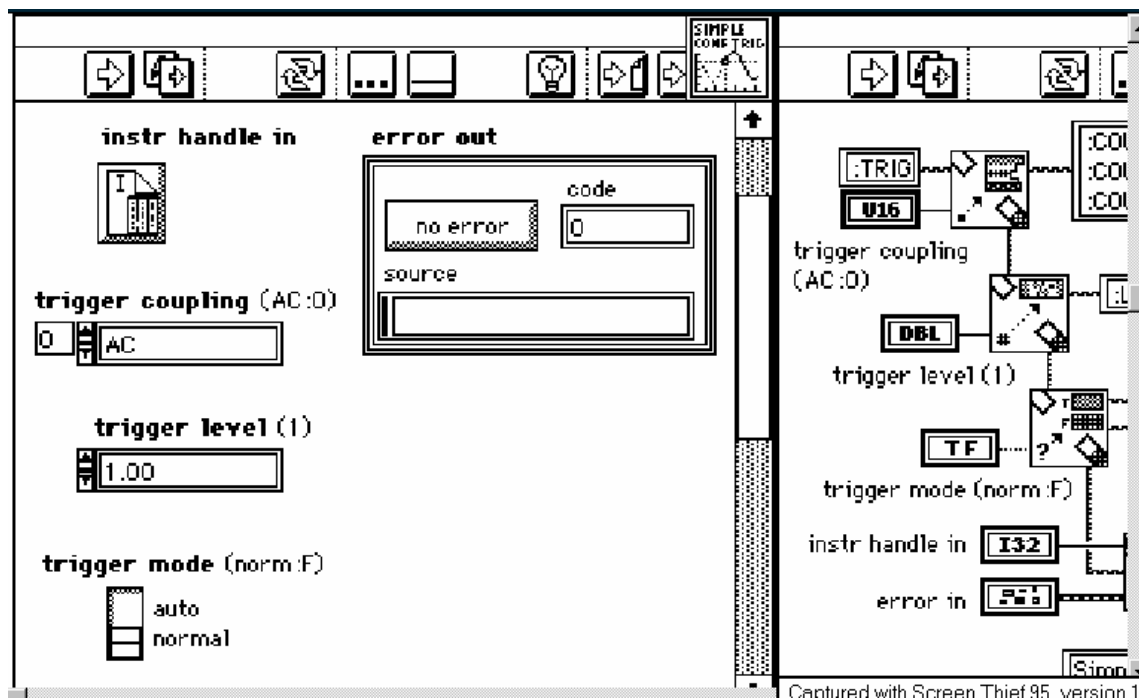
je klaster koji sadrži informaciju o grešci. Ako je greška sadržana u grešci, ovo se direktno prenosi u **error out**. Ako **error in** ne pokazuje nikakvu grešku, VI se normalno izvršava i **error out** opisuje bilo koju grešku koju je mogao generisati VI.

Kada se dizajniraju prednji paneli, treba koristiti slijedeće preporuke da bi se obezbjedila uniformnost sa drugim LabVIEW VI prednjim panelima. Prvo, koristi default fontove (tip slova) za sve labele (natpise) jer taj font koristi LabVIEW.

Takodjer treba koristiti **bold** tekst da bi se označile primarne ili važne komande, a obični (nenaglaseni) tekst za sekundarne komande. U većini slučajeva, sve komande instrumentalnog drajvera su primarne i zahtjevaju bold tekst. Takodjer potrebno je koristiti mala slova u svim riječima, izuzev skraćenice i akronimi koji zahtjevaju velika slova (napr. ID ili GPIB). Default informacija treba biti stavljena u zagrade u imenu komande tako da se default može vidjeti u help prozoru kada se ožičava unutar VI.

Naprimjer dizajner treba označiti funkcionalni prstenasti komandni selektor čija je default pozicija DC volti kao nulta **funkcija** (DCV:0), a Bulov selektor čija je pozicija true što indicira automatski **mode** (auto: T); (promjetimo da je default informacija pisana sa običnim tekstom).

Potrebno je postaviti ručku (handle) instrumenta u gornji lijevi ugao, a **error out** klaster u gornji ili donji desni ugao (zavisno da li smo se odlučili da uključimo instrumentov **handle out** indikator). Pošto greška u komandi nije dizajnirana da se koristi kao interaktivni ulaz, potrebno je postaviti van ekrana ili sakriti van vidnog polja. Popuniti sve kontrolne opise kako je to specificirano u Online help poglavlju ovog opisa. Naredna slika br. 6 pokazuje jednostavni triggerski VI nakon modifikacije da bi se zadovoljile preporuke u stilu. Dakle ovako treba izgledati prednji panel kako se to vidi na slici br. 14



Slika br. 14 Jednostavni triger VI koji slijedi preporuke za stil

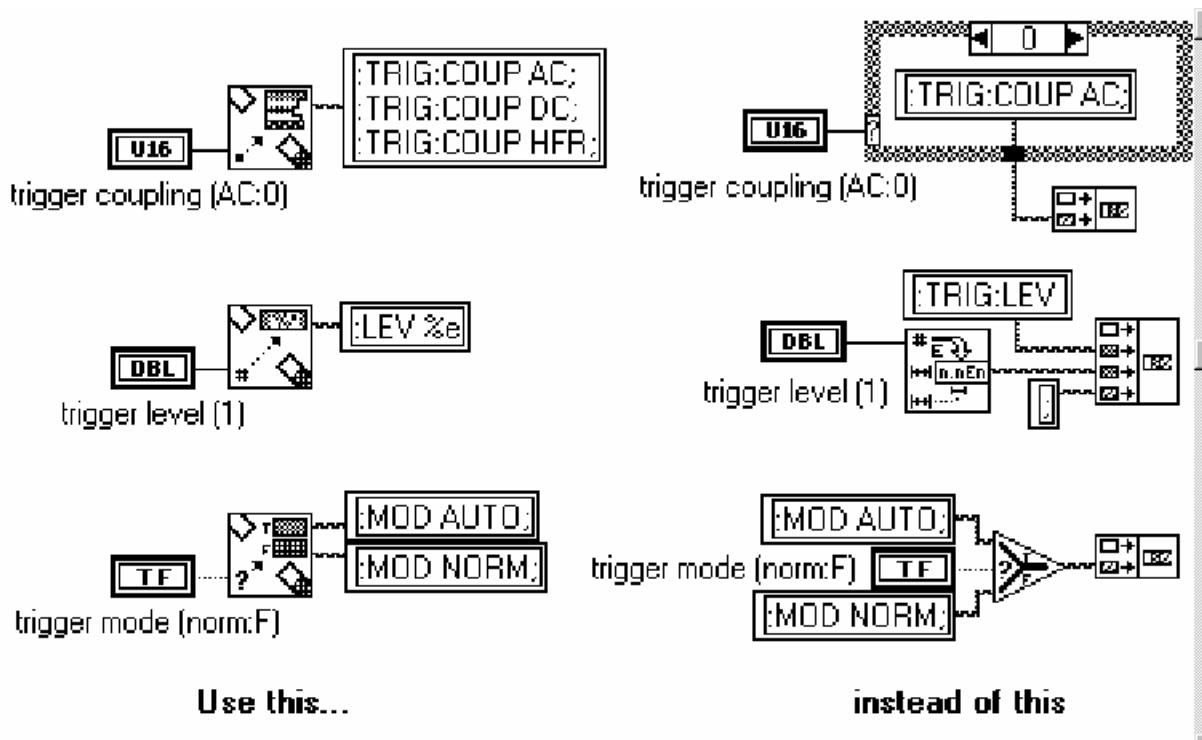
Jos jedna stvar koju treba razmatrati kod dizajniranja prednjih panela je korištenje opcije **Data Range** (opseg podataka). Pošto mnogi instrumenti će se blokirati ili pogrešno indicirati (pa čak se mogu i oštetiti) ako prime parametre koji su van opsega, dizajner mora primjeniti strategiju da detektuje i modifikuje ili pak izvjesti o pogrešnim postavnim vrijednostima (settings) prije nego sto su poslati instrumentu. Ovo može biti uradjeno na više načina, jedan od njih koristi ugradjenu osobinu u LabVIEW za **opseg podataka za parametre prednjeg panela**. Svaka komanda za upravljanje prednjim panelom ima opcije za uspostavljanje opsega podataka i akcija koje slijede. Najkorisnija je **coerce** koja automatski mjenja podatke u komandi koja je van opsega tako da je koriguje da bude unutar opsega.

Mada je **Data Range** jednostavan način da se spriječe mnoge greške opsega, **NI** sugeriše da se ne koristi zbog niza inherentnih ograničenja. Mada koercija spriječava greške opsega, nije baš lako vidljivo kada se koercija pojavi. Ovo može rezultirati u situaciju u kojoj se korisnik čudi zašto postavne vrijednosti instrumenta ne odgovaraju onima koje je on postavio u komandama za njih koje je unjeo. Drugo ograničenje je u tome što ovo provjeravanje ne djeluje u slučajevima kada se vrijednosti ne povećavaju linearno, kao sto je naprimjer postavljanje vremenske baze osciloskopa. Ovi tipovi kontrolnih elemenata neće dobiti zadovoljavajući nivo menagementa grešaka od funkcije **Data Range** jer imaju samo jednu inkrementalnu vrijednost. Dizajner može prevazići ovo ograničenje korištenjem tehnika blok dijagrama da bi se otkrile, korigovale i izvjestile greške opsega, ili pak ispitivanjem repova greški (queues) da bi se dobila informacija o greškama opsega. (vidjeti o ovom startegijama u sekciji **Error reporting** (Izveštavanje o greškama).

Blok dijagram

Nakon dizajniranja vašeg prednjeg panela, slijedeći korak je kreiranje dijagrama koji će izvršavati funkciju zahtjevanu od strane VI. Kao što je već rečeno, svaki tip prednjeg panela kontrolnog elementa ima odgovarajuću blok dijagram string funkciju koja pojednostavljuje zadatak gradnje komandnih stringova. Umjesto da ožičite Bulov kontrolni element do **Select** čvora i birajući string konstantu koja će biti poslata u **Concatenate Strings** čvor, koristite **Select & Append**. Ovaj čvor selektira odgovarajući string i pridružuje ga komandnom stringu u jednom koraku. Na isti način, koristite **Format & Append** da formatirate i povežete jednostavne numeričke vrijednosti, radije nego da koristite jedan od **To Decimal** ili **To Exponential** tip konverzionih čvorova u sprezi sa **Concatenate Strings** čvorom. Ponovno, **Format & Append** kombinira funkcionalnost odvojene konverzije i pridruženih čvorova i pojednostavljuje značajno blok dijagram. Za tekst ringove (prstenove), koristite **Select & Append**, a za string ulaze koristite **Concatenate Strings**.

Dijagram na slijedećoj slici br. 15 pokazuje preferirane metode za gradnju komandnih stringova.



Slika br. 15 Preferirane tehnike za gradnju stringova

Pažljivo analizirajte tok upravljanja dok gradite vaše dijagrame. LabVIEW ne izvršava program u očekivanom smjeru: sa lijeva na desno, odozgo - na dole. Postoji veliki nivo zavisnosti medju podacima koji automatski određuje redoslijed izvršenja programa; dodajte vještačku zavisnost medju podacima kada je to god moguće. Možete koristiti in/out klasterne greške da ulančite medjusobno I/O (input/output, ulazno/izlazne) funkcije.

Na taj način ćete definisati redoslijed izvršenja programa bez da koristite strukture **case** ili **sekvence** kako je to pokazano u HP34401A primjeru VI dijagrama pokazanog na Slici 3. Sekventne strukture, osim što sakrivaju dijelove dijagrama, su također efikasne u kontroli redoslijeda izvršenja programa. Bez obzira koji metod koristite, potrudite se da jasno definišete tok upravljanja izvršenjem programa i ne oslanjajte se na pretpostavke koja će se grana dijagrama prva izvršiti.

Čak i sa propisno definisanim redoslijedom izvršenja programa, nećete moći uvijek znati koliko je vremena potrebno instrumentu da odgovori na komande. Problemi timinga, se mogu pojaviti ako VI instrument drajvera pokuša da pošalje komande instrumentu dok je on zaposlen izvršenjem prethodno poslatih komandi. Često, nove komande će biti ignorirane. Upiti (queries) instrumentu predstavljaju drugi potencijalni problem. Nakon komandovanja uređaju da pošalje podatke, može proći izvjesno vrijeme prije nego što su podatci na raspolaganju. Ako pokušate da ih iščitavate za vrijeme ovoga perioda, podatci mogu biti nekorektni (corrupted), može se pojaviti **time-out** (isteklo vrijeme), ili instrument može otkazati. Ovi interni timing problemi mogu biti prevaziđeni na nekoliko načina:

Organizujte da instrument generira interapt da signalizira da je spreman da prihvati nove komande ili da su podatci raspoloživi. Instrument vam možda može dozvoliti da postavite bite u maski registra zahtjeva za servis, da konfigurirate specifične SRQ događaje. Ako je tako, možete koristiti **Wait for RQS** VI da suspendirate (zakočite) izvršenje VI instrument drajvera sve dok uređaj ne pokaže da je spreman da produži.

Nadalje, koristite informaciju o statusu da odredite da li je uređaj spreman. Možete pitati (query) mnoštvo instrumenata o njihovom stanju, i dekodirati ovu informaciju da odredite da li željeni uslov egzistira prije nego nastavite program. Unesite odgovarajuća vremenska kašnjenja za instrumente koji mogu generirati interapte indicirajući da su spremni za novu komandu. Pošto većina instrumenata ima ulazne bafere, obično je moguće poslati string koji sadrži nekoliko komandi instrumentu. Individualne komande se procesiraju od strane instrumenta jedna po jedna, na serijski način. Ponekad, instrument zahtjeva nekoliko trenutaka da završi izvršenje komandi u svom baferu prije nego što je spreman da prihvati nove komande ili odgovori na upit. Koristite samo **Wait (ms)** funkciju da uspostavite vremensko kašnjenje kada instrument ne može biti konfiguriran da generira servisni zahtjev kada je spreman, a informacija o njegovom statusu nije na raspolaganju.

Zajedno sa ovim pitanjima internog timinga, morate također razmatrati interakciju komponentnih VI-eva u drajveru. Ako jedan komponentni VI napusti instrument u pogrešnom stanju, druga komponenta VI može da radi pogrešno. Dodatno, mogu se pojaviti i timing problemi slični onim koje smo već opisali, ako jedna komponentna VI šalje komande instrumentu dok je on zauzet izvršavajući komande poslate od neke druge komponentne VI. Tehnike ranije pomenute su korisne u prevazilaženju ovih problema.

Koristite adekvatni stil ožičenja da poboljšate izgled dijagrama i lakoću njihovog razumjevanja. Ne pretrpavajte dijagram; ostavite dovoljno prostora za labela i žice. Ne zatrpavajte žice velikim brojem kontura (loops), slučajeva (cases),

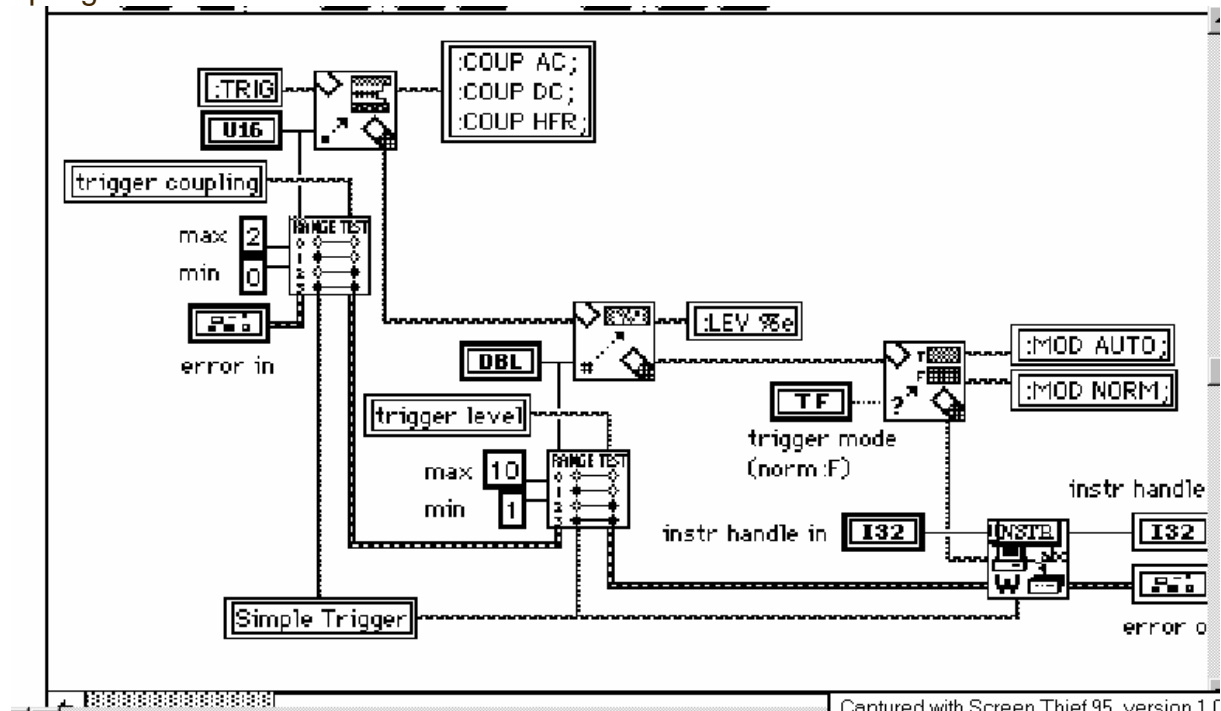
labela, ili drugim tipovima dijagramskih objekata. Također, reducirajte broj previjanja žica, poravnavajući terminale podataka kada je to god moguće. Možete koristiti kurzorske tastere da pomjerate objekte i po jedan piksel rezolucije ako je to potrebno. Koristite **Align** i **Distribute** u Edit meniju da dodate simetriju i prave linije u vašem dijagramu. Također, dodajte tekstualne labela svakom Okviru Case i Sekventnih struktura.

Možete labelirati duge žice i kompleksne operacije kada je to potrebno da povećate razumljivost. Labelirajte kontrolne i indikatorske čvorove sa normalnim tekstom, ali koristite masni (**bold**) tekst da istakne vaše slobodno stojeće komentare.

Morate biti spremni da izadjete na kraj i sa neočekivanim vrijednostima koje korisnik može unjeti u kontrolne elemente. Kao što je ranije istaknuto, element na prednjom panelu **Data Range** je otporan na mnoge greške opsega, ali postoje okolnosti koje ne dozvoljavaju njegovo korištenje.

U ovom slučaju, možete ili izgraditi subVI da testirate opsege i izvjestite klaster greske o greškama opsega, ili možete izgraditi blok dijagram rutine da prisile (coerce) podatak unutar opsega.

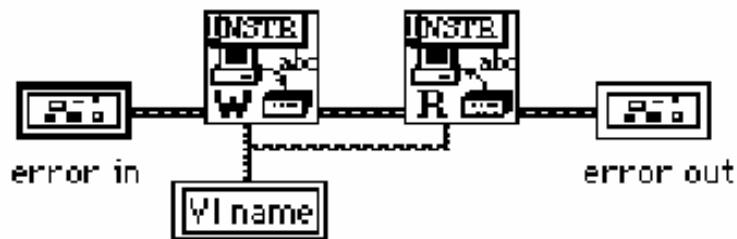
Nadalje, morate često uzimati u obzir da ostvarite kontrolne interakcije, koristeći **case** iskaze da izmjene blok dijagram rutine bazirane na postavljenoj vrijednosti nekog specifičnog kontrolnog elementa. Idealno, trebate biti u mogućnosti da izvršavate VI sa bilo kojom kombinacijom postavljenih vrijednosti (settings), bez da instrument otkáže u normalnom radu. Ako to nije slučaj, trebate dodati više elemenata za detekciju i korekciju u vaše blok dijagrame. Budite svjesni, ipak, da programirana provjera opsega, mada vjerovatno pojednostavljuje korištenje VI-eva, također komplikuje razvoj VI. Nadalje, programirana provjera opsega može lako udvostručiti veličinu vaših VI-eva i dodati penale na brzinu izvršenja. Slika br. 16 pokazuje promjene unjete u jednostavnu trigersku VI da programski provjeri opsege numeričkih ulaza.



Slika br. 16 Jednostavna trigerska VI koja koristi progr. Testiranje opsega

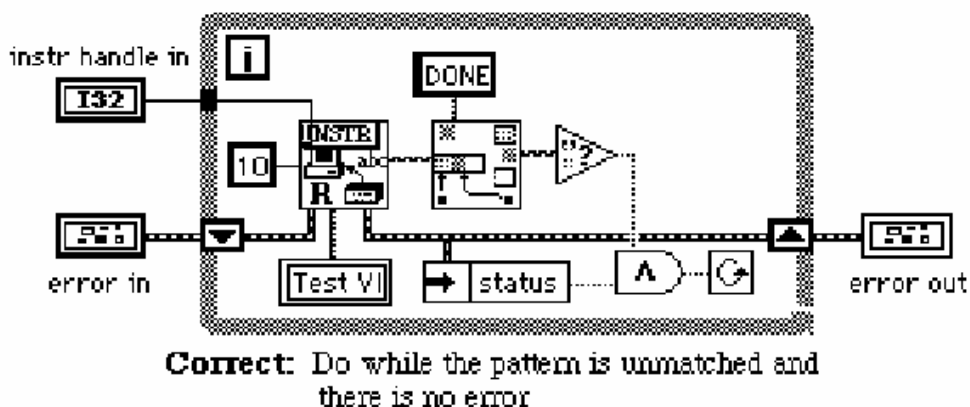
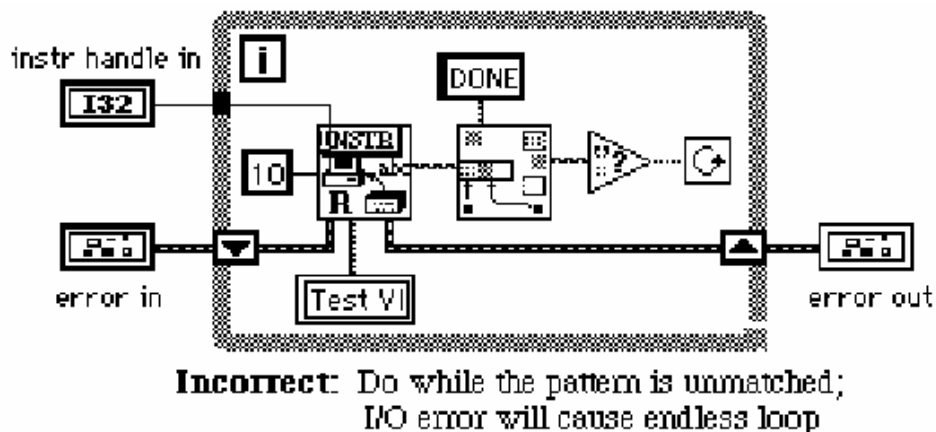
Postoji nekoliko pitanja koje morate analizirati kada koristite klaster grešaka da izvijestite o greškama. Ako se greška pošalje u **error in** terminal vašeg VI, I/O funkcije sadržane u VISA Transition biblioteci automatski prekidaju komunikaciju sa instrumentom da bi prenijeli poruku o grešci kroz vašu VI.

Ukoliko vi sami ne planirate da dodate vaše poruke grešaka ka **error out** klasteru, ne zahtjeva se nikakva druga akcija izuzev ožičenja konstante izvornog stringa sa I/O VI-evima i ožičenja **error in** kontrolnog čvora sa prvom I/O VI, ožičenja žice greške kroz slijedeće I/O VI-eve, i ožičenja **error out** indikatorskog čvora sa posljednjom I/O VI, kao što je to pokazano na narednoj slici 17.



Slika br. 17 Povezivanje Error I/O žice za propagaciju greške

Ako, sa druge strane, planirate da izvijestite o opsegu ili specifičnim greškama instrumenta ka izlazu klastera greške vaše VI, budite sigurni da ne prepisete bilo koju postojeću informaciju o grešci koja može već biti sadržana u klasteru greške. Ovo možete ostvariti na taj način da postavite vašu rutinu za osvježanje (update) u jedan **case** iskaz i izvršavajući ga samo ako nema postojeće greske. Opšta ideja je da se prenese prva greška do VI najvišeg nivoa, a ne posljednju grešku koja ustvari može biti posljedica prve. Takodjer, budite sigurni da postoji mogućnosti iskakanja iz while kontura ako se pojavi greška. Ovo je naročito važno ako koristite while konturu koja sadrži bilo kakve I/O rutine a uslovni test konture zavisi od rezultata I/O. Ako postoji greška, I/O rutine automatski isključuju VI i možemo se naći zarobljeni u beskonačnoj konturi. Zbog toga, uvijek testirajte Bulov klaster greške, u sprezi sa vasim uslovom normalnog završetka konture , da odredite kada završiti konturu, kako je to prikazano na narednoj slici br. 18.



Slika br. 18 Iskakanje iz While konture na pojavu greške

Alternativa programskom provjeravanju grešaka je da koristi ugradjene mogućnosti instrumenata da izvještavaju o greškama. Ova tehnika često daje bolje informacije o greškama korisniku sa minimalnim troškom za jedan dodatni upit (query). Radije nego da intenzivno koristi programsko provjeravanje grešaka, sugeriše se da se koriste upiti o greškama kada je to god moguće, a kao rezerva programsko provjeravanje za greške u situacijama kada upiti o greškama neće biti dovoljni.

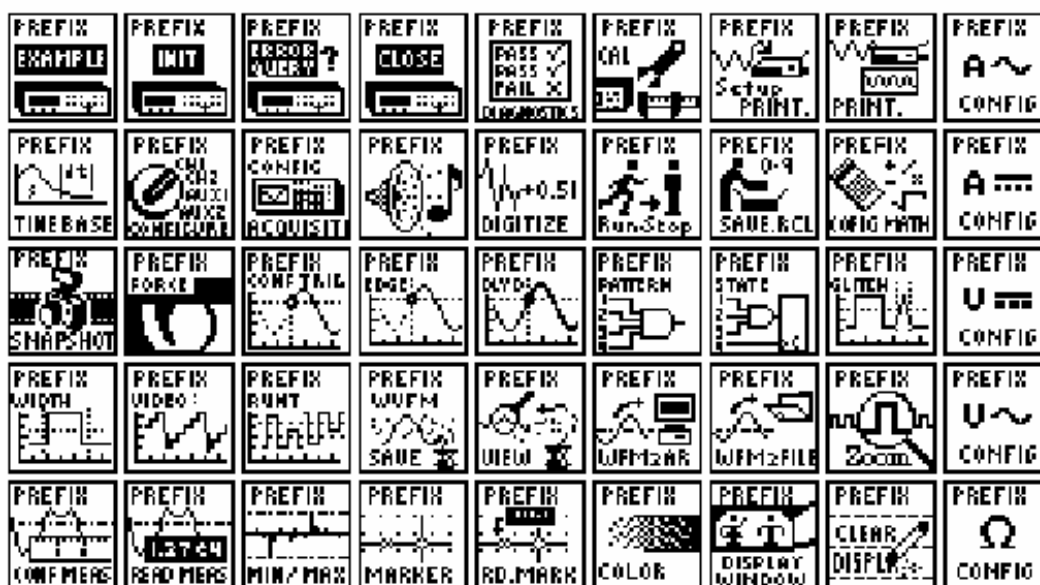
Ikona /Konektor

Postoji nekoliko pravila koja se odnose na konektorsko polje (pane) i ikonu svake VI. Tako je potrebno da rezervišete gornji lijevi dio terminala za **instr handle in** (tačku - ručku priključenja) a gornji desni dio terminala za **instr handle out** (opciono). Rezervirajte donji lijevi terminal za grešku u klasteru a donji desni terminal za **error out** klaster da bi pojednostavili ožičenje do odgovarajućih **error in** terminala. Prihvatljivo je izabrati konektorski patern koji ima dodatne terminale u slučaju da imate neke nepredvidjene dodatke u kontrolnim i indikacionim elementima u vašim instrument drajver VI-evima.

Ovo vam omogućuje da ne morate da mjenjate patern i zamjenjujete sve instance poziva u modificirane subVI-eve. Postavite ulaze na lijevoj strani a izlaze na desnu stranu kad god je to moguće, da bi obezbjedili tok podataka (data-flow) sa lijeva na desno u blok dijagramu.

Koristite ikone koje imaju prepoznatljivu grafičku vizuru za svaku VI. Naprimjer, možete **posuditi** ikone od sličnih funkcija u drugim instrument drajverima. Budite sigurni da uključite tekst u ikonu koja sadrži model instrumenta koji je kontrolisan od strane VI.

Ako niste u stanju da kreirate ikonu da izrazite funkciju VI, možete koristiti tekst. Primjeri ikona pokazanih na narednoj slici 19, mogu se naći u file-u **insticon.lib**.



Slika br. 19 Primjeri ikona

Važna razmatranja

Aplikacione VI

Aplikacione VI demonstriraju čestu upotrebu instrumenta i pokazuju kako komponentni VI-evi se programski koriste da izvrše zadatak. Naprimjer, aplikaciona VI osciloskopa, će konfigurisati vertikalna i horizontalna pojačala, triggerovati instrument, prikupiti valni oblik, i izvjestiti o greškama.

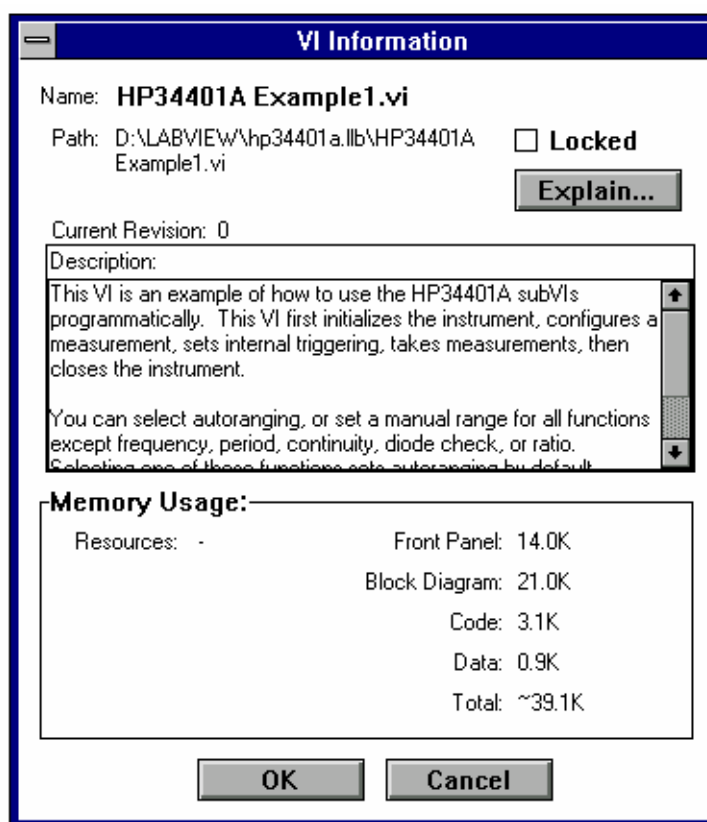
Ne pokušavajte da učinite da vaši test VI-evi koje razvijate, izvršavaju svaku moguću funkciju koju nadjete u instrument drajver komponentnoj VI. Umjesto toga, koncentrirajte se na gradnju jednostavnih, kvalitetnih test primjera.

Nemojte koristiti **initialize** i **close** VI unutar aplikacione VI, jer će je to učiniti manje upotrebljivom u aplikacijama višeg nivoa.

Online help informacije

Da pomognete korisniku, morate uključiti help za svaki instrument drajver. LabVIEW ima dva tipa help mehanizama koje korisnik može koristiti:

General Description help (Opšti opis), koji je na raspolaganju iz opisnog boks informativnog prozora kada korisnik izabere **Get Info...** iz **File** menija (pogledajte narednu sliku br. 20). Ovaj dijalog boks treba sadržavati opšti opis VI instrument drajver-a, uključujući i sva pravila korištenja kontrolnih elemenata, ili VI interakciju koja treba biti prezentirana korisniku.

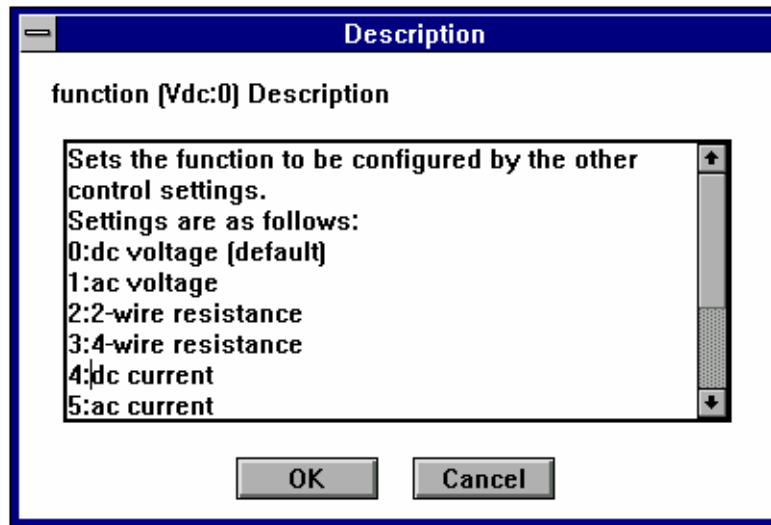


Slika 20 . GET INFO Opcija iz File menija

Takodjer, obezbjedite opis individualnih kontrolnih i indikatorskih elemenata. Help za ove elemente je najčešće korištena informacija od strane korisnika. *Front panel object help* se dobije izabirući **Data Operations>Description.** iz pop-up menija objekata (Slika br. 21).

Help informacija sadrži ime i opis parametra i njegov korektan opseg kao i default vrijednost. Budite sigurni da uključite informaciju koja pokazuje indeks brojeve i odgovarajuće postavne vrijednosti za sve ring i klizne (slide) kontrolne elemente, postavne vrijednosti koje korespondiraju True/False pozicijama na Bulovim kontrolnim elementima, kao i informacije o opsezima za numeričke kontrolne elemente. Vi takodjer trebate primjetiti i sve relevantne informacije koje se tiču

kontrolne interakcije u blokovima za opis svakog kontrolnog elementa koji može biti uključen u interakciju.



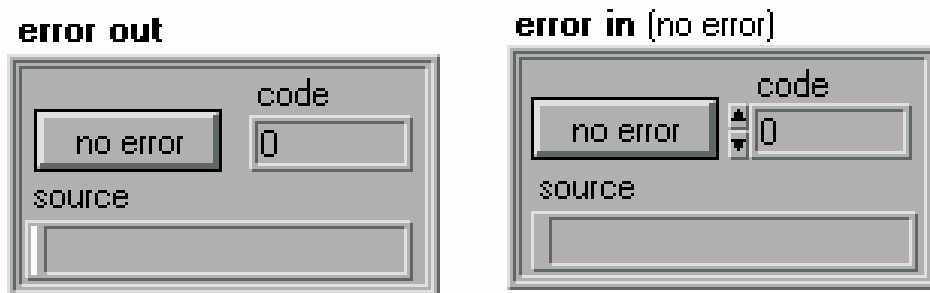
Slika br. 21 Prednji panel helpa za objekat iz Description... Option

Možete takodjer pomoći korisniku, postavljajući slobodne labele na prednji panel i u blok dijagram. Naročito u blok dijagramu, trebate pokazati sve labele terminala (u punom tekstu), a ivice učiniti transparentnim. Postavite slobodne labele u **case** iskaze i sekventne okvire koristeći masni (bold) tekst. Ovo čini da komentari se ističu što čini program lakšim za razumjevanje i modifikaciju.

Izveštavanje grešaka

LabVIEW instrument drajveri koriste National Instruments standard za izveštavanje o greškama baziran na korištenju klastera da izvjesti o svim greškama. (Slika 22).

Unutar klastera, Bulov indikator greške, numerički kod greške, i string indikator izvora greške izvještavaju ako postoji greška, specifični uvjet greške, kao i izvor (ime) VI u kojoj se pojavila greška (dodatni komentari mogu takodjer biti uključeni). Svaka VI instrument drajver-a ima **error in** i **error out** terminal definisan na njenom konektorskom polju (pane), korespodentno u donjem lijevom i donjem desnom terminalu. Ožičavajući **error out** klaster jedne VI sa **error in** klasterom druge VI, možete prenjeti informaciju o grešci kroz čitav vaš instrument drajver koja će propagirati sve do VI najvišeg nivoa (top level) u vašoj LabVIEW aplikaciji.



Slika 22 Klasteri I/O greške

Sekundarna korist od greške ulaz/izlaz je u tome što je zavisnost medju podacima dodata VI-evima koji inače nisu zavisni od podataka, i na taj način unosimo sredstva pomoću kojih možemo specificirati redoslijed izvršenja i van tradicionalnih sekventnih struktura (vidjeti slike 3 i 9). Nakon što je klaster greške prošao kroz sve instrument drajver VI-eve, može se poslati ka opštoj VI za manipulaciju greškom (General Error Handler VI), ili jednostavnoj VI za manipulaciju greške (Simple Handler VI) [distribuiranoj sa programom LabVIEW]. Ova VI interpretira kodove grešaka i prikazuje poruke korisniku.

Blok dijagram VISA transition VI-eva sadrži test greške (Bulova varijabla) da odredi da li je VI koja se prethodno izvršila generirala grešku. Ako je greška detektovana, **case** iskaz koji sadrži operativne elemente VI, preklapa isključujući time rad VI i prenoseći informaciju o grešci ka **error out** klasteru.

Ako greška nije detektovana, VI se normalno izvršava i određuje da li je generirala grešku. Ako jeste, informacija o grešci se prenosi do **error out** klastera, a ako nije **error in** informacija se propušta na izlaz. Koristeći se ovom tehnikom, prva greška trigeruje naredne VI-eve da se ne izvršavaju (ili neku drugu akciju koju korisnik definiše). Nakon toga kod greške i izvor greške propagiraju ka prednjem panelu vršnog nivoa (top level). Dodatno, upozorenja (kodovi greške i izvorne poruke sa Bulovom greškom postavljenom na False), će proći kroz VI bez da trigeruje akcije greške.

Tabela 2 izlistava kodove grešaka rezerviranih za instrument drajvere. VISA transition VI-evi interno generiraju kodove grešaka označene sa *; možete koristiti preostale kodove u svojim instrument drajverima ako su vam potrebni.

Tabela 2. Kodovi grešaka Instrument drajvera-a

Kod	Značenje	Generiran od
0	Nema greške: poziv je bio uspješan	
-1200	Nevalidna sintaksa stringa	VISA

-1201	Greška u traženju instrumenata	VISA
-1202	Nije mogao da inicijalizira interfejs ili instrument	VISA
-1205	Nevalidna ručka (handle) instrumenta	VISA
-1210	Parametar van opsega	Instrument Driver
-1215	Greška pri zatvaranju instrumenta	VISA
-1218	Greška pri dobijanju atributa	VISA
-1219	Greška pri postavljanju atributa	VISA
-1220	Nije mogao da otvori instrument	VISA
-1223	Upit radi identifikacije instrumenta nije uspjelo	Instrument Driver
-1224	Greška pri brisanju memorije instrumenta	VISA
-1225	greška pri trigerovanju instrumenta	VISA
-1226	Greška u pozivanju (izboru) instrumenta	VISA
-1228	Greška pri pisanju u instrument iz file-a	Instrument Driver
-1229	Greška pri čitanju iz instrumenta u file	Instrument Driver
-1230	Greška pri pisanju u instrument	VISA
-1231	Greška pri čitanju iz instrumenta	VISA
-1232	Instrument nije inicijaliziran	VISA
-1234	Greška pri postavljanju instrumenta u lokalni mod	VISA
-1236	Greška pri interpretaciji odziva instrumenta	Instrument Driver
-1239	Greška pri konfiguriranju timeout-a	Instrument Driver
-1240	Instrument je dospjelo u timeout	Instrument Driver
-1250	Greška pri mapiranju VXI adresa	VISA
-1251	Greška pri odmapiranju VXI adrese	VISA
-1252	Greška pri pristupu VXI adresi	VISA
-1260	Funkcija nije podržana od strane kontrolera	VISA
-1300	Greške specifične za tip instrumenta	Instrument Driver

Kao što je već rečeno u prethodnim sekcijama, postoje najmanje tri metoda za manipulisanje sa greškama. Najjednostavniji metod, provjeravanjem opsega podataka koji se unose na prednji panel, može otkriti osnovne greške opsega i provesti koerciju (prisiljavanje) da ostanu unutar opsega. Ovaj metod ne koristi klastere I/O grešaka, pa se niti ne može poduzeti programirana akcija u slučaju pojave grešaka opsega.

Drugi metod, manipulacija greškama putem blok dijagrama, može biti korišten u nizu situacija uključujući greške opsega, konflikti medju postavnim vrijednostima, kao i neke druge, ali zahtjeva da vi definirate programski i rutine za detekciju kao i rutine za korekciju grešaka i njihovo izvještavanje.

Ovo može značajno povećati veličinu i kompleksnost vaših VI-eva, ali isto tako može značajno da olakša njihovo korištenje.

Treći metod, upiti o postojanju grešaka instrumenta (instrument error queries), oslanjaju se na instrument da on detektira i korigira slučajevne grešaka i da o tome izvjesti LabVIEW. Ovaj metod upita o grešci, pojednostavljuje razvoj VI kao i njeno korištenje, ako se korektno implementira, ali ovo je moguće samo sa onim instrumentima koji imaju ove mogućnosti samo detekcije grešaka.

Kako je definisano SCPI standardom mnogi instrumenti imaju red čekanja koji pohranjuje greške i događaje u redosljedu kako su detektovani. Ovaj red čekanja je tipa FIFO (first in,first out), tj. Prvi u bafer, prvi iz bafera(reda). U slučaju

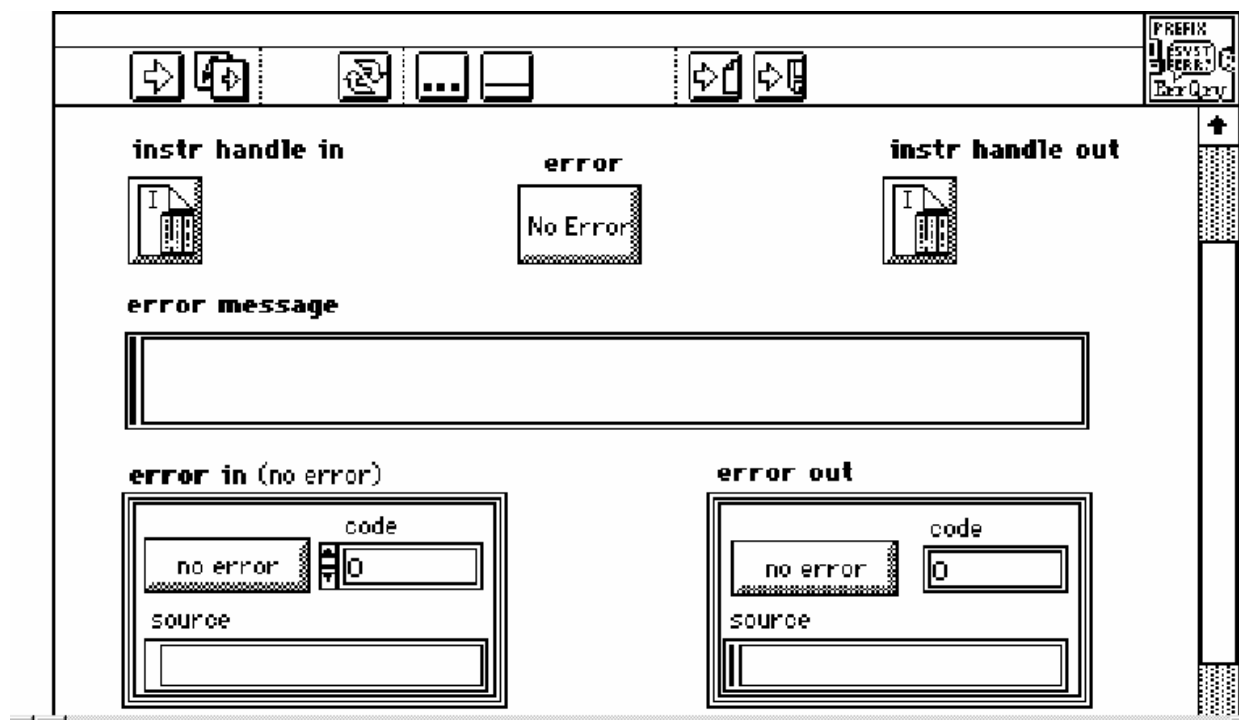
preplavljanja reda, zadržavaju se najstariji greške/dogadjaji, dok najsvježiji greške/dogadjaji se zamjenjuju sa porukom o prelivu reda (queue overflow). SCPI standard definiše neke opšte tipove grešaka, uključivo komandne greške, greške izvršenja, greške specifične za uređaje, i greške upita.

Svaka greška se pohranjuje u rep, sa jedinstvenim brojem za grešku/dogadjaj, opcionim deskriptorom, i opcionom informacijom koja je zavisna od tipa instrumenta. Izdavanjem komande: SYST: ERR? , SCPI instrumenti vraćaju jedan ulaz iz repa, koji može biti greška, upozorenje o prelivu, ili poruka, 0, tj. Nema greške.

U vašoj aplikacionoj VI sa instrument drajverom, možete koristiti ovaj rep čekanja da detektirate i izvestite o greškama instrumenata, kroz upite instrumenta nakon slanja komandi. Očito, upiti instrumentu o greškama dodaje na vrijeme izvršenja VI, ali ova tehnika je povoljna zbog toga jer djeluje kao mehanizam za detekciju specifičnih grešaka vezanih za instrument.

File za uzorke (templates) instrument drajver-a sadrži dvije verzije VI sa SCPI čitanjem grešaka, koji se može također koristiti kao uzorak kod gradnje VI-eva. VI **PREFIX Error Query** je najkorisnija sa SCPI instrumentima pošto ona čisti (flushes) bafer instrumenta za greške, i detektira prisustvo bilo kojih poruka o greškama.

Ako se otkriju greške, **PREFIX Error Query** VI će ažurirati klaster greške sa kodom -1300, i postaviti u izvornu poruku ime VI izvršavajući upit o grešci, kao i informaciju o grešci koju je vratio instrument. VI-evi za manipulisanje sa greškama u LabVIEW koriste kod greške -1300 kao greška specifična za dati tip instrumenta, i generira odgovarajuću poruku greške. Prednji panel VI je pokazan na narednoj Slici br. 23 .



Slika Br. 23 PREFIX Error Query (multiple) VI

Da bi se koristila ova VI, postavite je bilo gdje u vašoj aplikaciji gdje želite da pitate instrumente o postojanju grešaka. Za vrijeme početnog razvoja, možete želiti postaviti ovu VI u svaku komponentnu VI instrument drajvera da odredite da li vaša VI generira instrumentalne greške koje mogu biti izbjegnute sa boljim algoritmom. Sklonite ih iz finalne verzije da bi optimizirali drajver. U vašoj aplikacionoj VI, pozovite VI za upit na grešku, nakon što su se izvršile komponentne VI. Pošto SCPI instrumenti baferuju greške u rep čekanja, VI za upit na grešku (error query) biće u stanju da izvjestite o svim greškama koje su se pojavile kroz aplikaciju. Ako je informacija o grešci koju je vratio instrument dovoljno detaljna da se može tačno odrediti šta je krenulo loše u instrument drajveru, nema potrebe da dodajete programirano ispitivanje grešaka u vašim dijagramima, koristite mogućnosti instrumenta za ovo.

Ukoliko pak, vraćena informacija o grešci je kriptična (kodirana) ili isuviše opšta da bi bila od praktične vrijednosti, morate dodati više mehanizama za provjeravanje i korekciju grešaka u vašem VI-u, prije nego one dosegnu do instrumenta. Ako želite da informirate korisnike vašeg drajvera o uslovima grešaka u instrumentu, VI za upit grešaka (error query) je još jedan alat koji može biti upotrebljen da se ostvari ovaj cilj.

Biblioteke za podršku drajverima

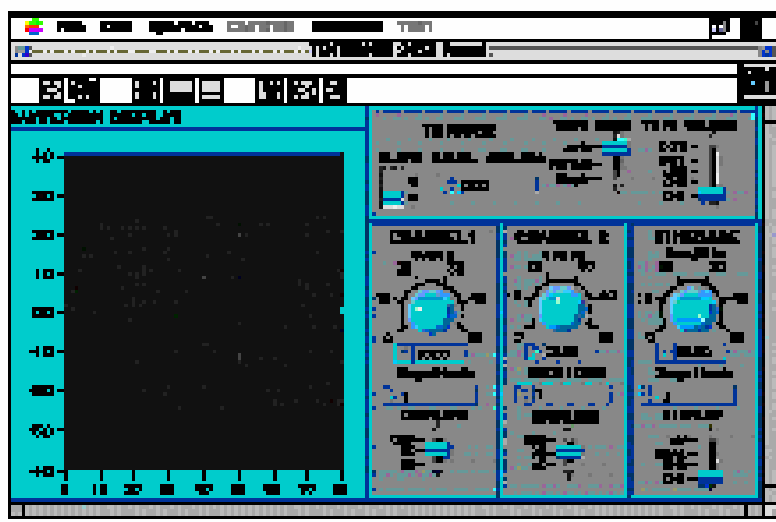
Uključene se verzijom LabVIEW 3.1 , biblioteke za podršku drajverima sadrže alate koji vam pomažu u razvoju drajvera za vaš instrument. Biblioteke sadrže uzorke (templates) VI-eva koje onda služe kao polazna tačka za kreiranje vaših vlastitih drajvera, komunikacione VI koje izvršavaju I/O instrumenta, biblioteke ikona, kao i podržavajući file-ovi. Glavni elementi interesa su VISA Transition biblioteka kao i VI uzoraka instrument drajvera.

POGLAVLJE 10

PREGLED LabVIEW INSTRUMENT DRIVERI

GPIB, VXI, Serijski i CAMAC LabVIEW Instrument driveri

Biblioteka LabVIEW Instrument drivera sadrži drivere za više od 500 GPIB, VXI, i serijskih instrumenata od više od 45 različitih proizvođača instrumenata. LabVIEW paket instrument drivera se sastoji od skupa standardnih VI-eva. Svaka VI korespondira nekoj programskoj operaciji kao što je konfiguriranje, očitavanje sa, upisivanje na, i trigerovanje instrumenta. LabVIEW instrument driveri reduciraju vrijeme potrebno za razvoj i pojednostavljaju kontrolu instrumenta eliminirajući potrebu za učenjem kompleksnih, nisko-nivovskih programskih protokola za svaki instrument posebno.



Driveri su pisani koristeći grafički programski jezik nazvan LabVIEW, tako da korisnik može ispitivati i modifikovati izvorni kod (source code) blok dijagrama drivera. Driveri, koji se distribuiraju sa izvornim kodom su ustvari lake za korištenje ikone koje se mogu uključiti i kombinirati u test programima višeg nivoa.

Razvojni alati za Instrument drivere

NI je kreirao specijalne razvojne alate za razvoj instrument drivera, da bi olaksao razvoj standardnih instrument drivera.

LabVIEW Instrument driver uzorci (templates) su osnova za sve instrument drivere i ubrzavaju razvoj VXI plug&play (P&P) kompatibilnih instrument drivera. Uzorci imaju jednostavnu fleksibilnu strukturu i skup zajedničkih VI-eva kao što su:

initialize, close, self-test, reset, error query, itd.

, za korištenje pri razvoju drivera. Svaki ima instrukcije za modifikovanje prema specifičnom tipu instrumenta.

VISA transition biblioteka je podskup skupa VISA (Virtual Instrument Software Architecture) baziranih VI-eva, obezbjedjujuci funkcionalnost koja je potrebna za povezivanje sa GPIB, MXI i VXI kontrolerima.

Kompletna LabVIEW biblioteka instrument drivera je na raspolaganju kod **NI** na CD.

Ova biblioteka sadrzi instrument drivere za slijedece instrumente:

MODEL NUMBER	INSTRMENT TYPE
Gould 270	Waveform Processor
Option for Gould 407x/9x	Waveform Processor
Gould 1602	Digitizing Oscilloscope
Gould 1604	Digitizing Oscilloscope
Gould 475	Digitizing Oscilloscope
Gould 4062	Digitizing Oscilloscope
Gould 4064	Digitizing Oscilloscope
Gould 4066	Digitizing Oscilloscope
Gould 4068	Digitizing Oscilloscope
Gould 4072	Digitizing Oscilloscope
Gould 4074	Digitizing Oscilloscope
Gould 4090	Digitizing Oscilloscope
Gould 4092	Digitizing Oscilloscope
Gould 4094	Digitizing Oscilloscope
Hameg 408	Oscilloscope
Hewlett-Packard 5180A	Digitizer
Hewlett-Packard 54100A	Digitizing Oscilloscope
Hewlett-Packard 54110D	Digitizing Oscilloscope
Hewlett-Packard 54111D	1 Gsa/s 2-Channel digitizing oscilloscope
Hewlett-Packard 54120A	Digitizing Oscilloscope
Hewlett-Packard 54200A	Digitizing Oscilloscope
Hewlett-Packard 54501A	10 Msa/s 2-channel digitizing oscillos.
Hewlett-Packard 54502A	400 Msa/s 2-channel digitizing oscilloscope
Hewlett-Packard 54503A	20 Msa/s digitizing oscilloscope
Hewlett-Packard 54505B	300 MHz 2-channel digitizing oscillos.
Hewlett-Packard 54506B	300 MHz 4-channel digitizing oscillos.
Hewlett-Packard 54510A	Digitizing Oscilloscope
Hewlett-Packard 54510B	300 MHz 2-channel digitizing oscillos.
Hewlett-Packard 54512B	300 MHz 4-channel digitizing oscillos.
Hewlett-Packard 54520A	Digitizing Oscilloscope
Hewlett-Packard 54522A	Digitizing Oscilloscope
Hewlett-Packard 54540A	Digitizing Oscilloscope
MODEL NUMBER	INSTRMENT TYPE
Hewlett-Packard 54542A	2 Gsa/s Oscilloscope
Hewlett-Packard 54600A	Digitizing Oscilloscope
Hewlett-Packard 54601A	Digitizing Oscilloscope

Hewlett-Packard 54602B	Digitizing Oscilloscope
Hewlett-Packard 54610B	Digitizing Oscilloscope
Hewlett-Packard 54657	Measurement/Storage module
Hewlett-Packard 54720A	Digitizing Oscilloscope
Iwatsu DS6121	Digitizing Oscilloscope
Iwatsu DS6612	Digitizing Oscilloscope
LeCroy LS-140	Digitizing Oscilloscope
LeCroy 7200	Digitizing Oscilloscope
LeCroy 9310	Digitizing Oscilloscope
LeCroy 9314	Digitizing Oscilloscope
LeCroy 9400	Digitizing Oscilloscope
LeCroy 9420	Digitizing Oscilloscope
LeCroy 9450	Digitizing Oscilloscope
Nicolet 320	Digitizing Oscilloscope
Nicolet 400	Series Oscilloscope
Nicolet 4094 A/B	Digitizing Oscilloscope
Philips PM 3350A 100 MS/s	Digitizing Oscilloscope
Philips PM 3382	Digitizing Oscilloscope
Philips PM 3384	Digitizing Oscilloscope
Philips PM 3392	Digitizing Oscilloscope
Philips PM 3394	Digitizing Oscilloscope
Tektronix DSA 600	Digitizing Analyzer
Tektronix RTD 710	Waveform digitizer
Tektronix 720	Digitizer
Tektronix SCD 1000	Digitizer
Tektronix SCD 5000	Digitizer
Tektronix TDS 31050 MHz	Digitizing Oscilloscope
Tektronix TDS 320 100 MHz	Digitizing Oscilloscope
Tektronix TDS 350 200 MHz	Digitizing Oscilloscope
Tektronix TDS 420 150 MHz	Digitizing Oscilloscope
Tektronix TDS 460350 MHz	Digitizing Oscilloscope
Tektronix TDS 520 500 MHz	Digitizing Oscilloscope
Tektronix TDS 540 500 MHz	Digitizing Oscilloscope
Tektronix TDS 640 500 MHz	Digitizing Oscilloscope
Tektronix TDS 644 500 MHz	Digitizing Oscilloscope
Tektronix TDS 820	6 Ghz Digitizing Oscilloscope
Tektronix 7D20	Digitizer
Tektronix 390AD	Digitizer
Tektronix 2212	60 Mhz digital storage oscilloscope
Tektronix 2230	100 Mhz dual time base digitizing oscilloscope
Tektronix 2252	100 Mhz oscilloscope
MODEL NUMBER	INSTRUMENT TYPE
Tektronix 2430A	100 MS/s 150 MHz digitizing oscilloscope
Tektronix 2432	Digitizing Oscilloscope
Tektronix 2440	500 MS/s 300 MHz digitizing

	oscilloscope
Tektronix 2465A	Analog oscilloscope tektronix 7612D digitizer
Tektronix 7854	700 MHz waveform processing oscilloscope
Tektronix 7912AD/HB	Digitizer
Tektronix 11400	Digitizing Oscilloscope
Tektronix 11800	Digitizing Oscilloscope
INSTRUMENTS DRIVERS - METERS	
B&K 2977	Phase meter
Fluke 45	Digital multimeter
Fluke 8502A	Digital multimeter
Fluke 8505A	Digital multimeter
Fluke 8506A	Digital multimeter
Fluke 8520A	Digital multimeter
Fluke 8540A	Digital multimeter
Fluke 8840A	Digital multimeter
Fluke 8842A	Digital multimeter
Giga-tronics 8541	Digital Power meter
Giga-tronics 8542	Digital Power meter
Hewlett-Packard 437B	Power meter
Hewlett-Packard 438A	Dual channel Power meter
Hewlett-Packard 3437A	High speed DC voltmeter
Hewlett-Packard 3456A	Digital multimeter
Hewlett-Packard 3457A	Digital multimeter
Hewlett-Packard 3458A	Digital multimeter
Hewlett-Packard 3478A	Digital multimeter
Hewlett-Packard 4192	LCR meter
Hewlett-Packard 4263A	LCR meter
Hewlett-Packard 4275A	LCR meter
Hewlett-Packard 4276A	LCZ meter
Hewlett-Packard 8452A	Spectrophptometer
Hewlett-Packard 34401A	Digital multimeter
Infratek 305	Wattmeter
Keithley 192	Digital multimeter
Keithley 195A	Digital multimeter
Keithley 196	Digital multimeter
Keithley 197A	Digital multimeter
Keithley 199	System Multimeter/Scanner
Keithley 485	Picoammeter
Keithley 580	Micro ohmmeter
MODEL NUMBER	INSTRUMENT TYPE
Keithley 617	Electrometer
Keithley 2000	Digital Multimeter
Keithley 2001	System Multimeter/Scanner
Keithley 2002	Digital Multimeter
Keithley 6517	High resistance meter/elector

Molelectron JD2000	Joulemeter/Ratiometer
Molelectron 4001	Laser energy meter
Newport Corp. 835	Optical power meter
Philips PM 2534	System DMM
Philips PM 6304	LCR meter
Prema 4000	Digital Multimeter
Prema 5000	Digital Multimeter
Prema 6000	Digital Multimeter
Prema 6031	Digital Multimeter
Rohde & Schwarz URE3	RMS/Peak Voltmeter
Siemens B3220	Digital Multimeter
Solartron 7061	Digital Multimeter
Solartron 7061	Digital Multimeter
Solartron 7081	Digital Multimeter
Solartron 7151	Digital Multimeter
Stanford Research 715	LCR meter
Stanford Research 720	LCR meter
Tektronix DM 5010	Digital Multimeter
Tektronix DM 5110	Digital Multimeter
Wandel & Goltermann P JM-4S	Jitter meter
Wandel & Goltermann SF60	Error and Jitter meter
INSTRUMENT DRIVERS – POWER SUPPLIES	
HAMEG 8142	Programmable power supply
Hewlett-Packard 6030A	Single output system power supply
Hewlett-Packard 6031A	Power supply
Hewlett-Packard 6032A	Power supply
Hewlett-Packard 6033A	Power supply
Hewlett-Packard 6038A	Power supply
Hewlett-Packard 6050A	System DC electronic load maiframe
Hewlett-Packard 6051A	System DC electronic load maiframe
Hewlett-Packard 6624A	Quad-output system DC power supply
Hewlett-Packard 6625A	Precision dual-output DC power supply
Hewlett-Packard 6626A	Precision dual-output DC power supply
Hewlett-Packard 6628A	Precision dual-output DC power supply
Hewlett-Packard 6629A	Precision dual-output DC power supply
Hewlett-Packard 6632A	Single-output DC power supply
Hewlett-Packard 6633A	Single-output DC power supply
Hewlett-Packard 6634A	Single-output DC power supply
Hewlett-Packard 59501B	Power supply Programmer
Keithley 213	Voltage source
MODEL NUMBER	INSTRUMENT TYPE
Keithley 220	Current source
Keithley 228	Voltage/current source

Keithley 230	Voltage source
Kepeco BOP488	Power supply
Kepeco SN488	Power supply programmer
Kepeco SNR488	Power supply programmer
Philips PM 2811	Single output power supply
Philips PM 2812	Dual output power supply
Philips PM 2813	Triple output power supply
Philips PM 2831	Programmable power supply
Philips PM 2832	Programmable power supply
Tektronix PS 2510G	Power supply
Tektronix PS 2511G	Power supply
Tektronix PS 2520G	Power supply
Tektronix PS 2521G	Power supply
Tektronix PS 5010	Power supply
Xantrex XKW Series	Power supply
INSTRUMENT DRIVERS – SOURCES	
Anritsu-Wiltron 6700	SeriesSwept Frequency Synthesizer
Anritsu-Wiltron 68000B	SeriesSynthesized Sweep generator
Anritsu-Wiltron 68100B	SeriesSynthesized Sweep generator
Anritsu-Wiltron 68200B	SeriesSynthesized Sweep generator
Anritsu-Wiltron 68300B	SeriesSynthesized Sweep generator
B&K 1049	Sine generator /noise generator
B&K 1051	Sine generator
Giga-troncis 7200	Series synthesized Microwave sweeper
Hameg 8130	Programmable function generator
Hewlett-Packard 3314A	Function Generator
Hewlett-Packard 3325A	Synthesized Function/Sweep generator
Hewlett-Packard 4140B	Voltage source
Hewlett-Packard 8116A	Pulse/Function Generator
Hewlett-Packard 8341	Synthesized sweeper
Hewlett-Packard 8643A	Signal generator
Hewlett-Packard 8647A	Synthesized signal Generator
Hewlett-Packard 8656B	Economy RF Signal Generator
Hewlett-Packard 8657A/B	Economy RF Signal Generator
Hewlett-Packard 8663A	Frequency synthesizer
Hewlett-Packard 8770	Arbitrary Waveform Synthesizer
Hewlett-Packard 8904A	Function Synthesizer/Generator
Hewlett-Packard 33120A	Function/Arbitrary waveform generator
LeCroy 9100	Arbitrary Function Generator
Philips PM 5193	Function Generator
Rohde & Schwarz	SMXSignal Generator
Stanford Research DG535PG	Digital pulse and delay generator
Stanford Research DS335	3 MHz Synthesized function generator
MODEL NUMBER	INSTRUMENT TYPE
Stanford Research DS340	15 MHz Synthesized function

	generator
Stanford Research DS345	30 MHz Synthesized function generator
Tabor 8550	Function generator
Tabor 8551	Function generator
Tektronix AFG 5101/5501	Prog. Arb./Function Generator
Tektronix FG 5010	Function Generator
Tektronix HFS 9003	Stimulu System
Wavetek 23	12 MHz synthesized function generator
Wavetek 75	Arbitrary wavefom generator
Wavetek 271	12 MHz pulse/function generator
Wavetek 295	Arbitrary wavefom generator
Wavetek 395	Arbitrary wavefom generator
INSTRUMENT DRIVERS - ANALYSERS	
Anritsu-Wiltron 2602A	RF Spectrum Analyser
Anritsu-Wiltron 37200A	Series Vector network analyser
Audio Precision SYS-22G	Sysem One Audio Analyser
Audio Precision SYS-222G	Sysem One Audio Analyser
Audio Precision SYS-322G	Sysem One Audio Analyser
EG&G 5208	Lock-in Analyser
Hewlett-Packard 3561A	Dynamic Signal Analyser
Hewlett-Packard 3577B	Network Analyser
Hewlett-Packard 4145B	Semiconductor Parameter Analyser
Hewlett-Packard 4155A	Semiconductor Parameter Analyser
Hewlett-Packard 4156A	Semiconductor Parameter Analyser
Hewlett-Packard 8510B	Network Analyser
Hewlett-Packard 8566B	Spectrum Analyser
Hewlett-Packard 8591E	Spectrum Analyser
Hewlett-Packard 8720	Network Analyser
Hewlett-Packard 8753	Network Analyser
Hewlett-Packard 8901A	Modulation Analyser
Hewlett-Packard 8903A	Audio Analyser
Hewlett-Packard 8903B	Audio Analyser
Newport Corp.	SuperCavity Optical Spectrum analyser
Norma-Goerz D6100	Wide Band Power Analyser
Schlumberger 1253	Gain-phase analyser
Tektronix AA 5001	Distrotion Analyser
Tektronix CSA 803	Communication Signal Analyser
Tektronix CTS 710	SONET Analyser
Tektronix 2754P	Network Analyser
Tektronix 2782	Spectrum Analyser
Tektronix 2784	Spectrum Analyser
Tektronix 2792	Spectrum Analyser
Tektronix 2794	Spectrum Analyser
Tektronix 2795	Spectrum Analyser

Tektronix 2797	Spectrum Analyser
MODEL NUMBER	INSTRUMENT TYPE
Tektronix 3001	Logic Analyser
Tektronix 3002	Logic Analyser
TTC Firebid 6000	Communication analyser
Voltech PM1200	Power analyser
Voltech PM3000A	Universal Power analyser
INTRUMENT DRIVERS – COUNTER/TIMERS	
Advantest R5373P	Microwav Counter/Timer
EIP Microwave 575B	Counter
EIP Microwave 578B	Counter
Hewlett-Packard 5316A	Universal Counter
Hewlett-Packard 5334A/B	Universal Counter
Hewlett-Packard 5335A	Counter
Hewlett-Packard 5342A	Microwave Counter
Hewlett-Packard 5345A	Counter
Hewlett-Packard 53131A	225 MHz Universal Counter
Hewlett-Packard 53132A	225 MHz Universal Counter
Hewlett-Packard 53181ARF	Frequency Counter
Philips PM 6666	Frequency Counter
Philips PM 6680	Programmable timer/Counter
Racal-Dana 1992	Counter
Racal-Dana 1994	Universal Counter
Stanford Research 400	Gated Photon Counter
Tektronix DC 5004	Counter/Timer
Tektronix DC 5009	Counter/Timer
Tektronix DC 5010	Counter/Timer
INSTRUMENT DRIVERS - CALIBRATORS	
EDC 520A	Calibrator
Fluke 5101B	Calibrator
Fluke 5440B	Calibrator
Precision Filters 6201C	Calibration Oscillator
Tektronix CG 5001/551AP	Calibration Oscillator
INSTRUMENT DRIVERS – MISCELLANEOUS	
Aerotech Unidex 11/12	Motion Controller
Aerotech Unidex 100	Motion Controller
Aerotech Unidex 400	Motion Controller
Audio Precision ATS-1	Audio Test System
EG&G 5210	Lock-In Amplifier
Fluke 2240C	Datalogger
Fluke 2620A	Data Acquisition Unit
HBM DMC	Digital Amplifier system
HBM MGC	Measurement Amplifier system
Hewlett-Packard 3421A	Data Acquisition/Control unit
Hewlett-Packard 3488A	Switch/Control unit
Hewlett-Packard 3497A	Data Acquisition unit
Hewlett-Packard 3852A	Data Acquisition system

MODEL NUMBER	INSTRUMENT TYPE
Hewlett-Packard 44701A	5-1/2 Digit voltmeter
Hewlett-Packard 44702A	High speed voltmeter
Hewlett-Packard 44705A	20 channel relay mux
Hewlett-Packard 44706A	60 channel single ended relay mux
Hewlett-Packard 44709A	20 channel FET mux
Hewlett-Packard 44713A	24 channel FET mux
Hewlett-Packard 44715A	5 channel counter/totaliser
Hewlett-Packard 44721A	16-ch digital input
Hewlett-Packard 44722A	8 channel AC digital input
Hewlett-Packard 44724A	8 channel relay actuator
Hewlett-Packard 44726A	2 channel arbitrary waveform DAC
Hewlett-Packard 44727A	4 channel voltage DAC
Hewlett-Packard 44728A	8 channel relay actuator
Hewlett-Packard 44730A	4 channel track hold mux
Hewlett-Packard 44732A	4 channel 120 Ohm strain gage mux
Hewlett-Packard 44733A	4 channel 350 Ohm strain gage mux
Hewlett-Packard 7470A	Digital plotter
Hewlett-Packard 7475A	Digital plotter
Hewlett-Packard 7550A	Digital plotter
Hewlett-Packard 8902A	Measuring receiver
Hewlett-Packard 8920	RF Communication test set
Hewlett-Packard 8958A	Cellular radio interface
Hewlett-Packard 11713A	Attenuator/Switch driver
Keithley 236	Source measure unit
Keithley 237	High voltage source measure unit
Keithley 238	High current source measure unit
Keithley 705	Scanner
Keithley 707	Switching matrix mainframe
Keithley 7001	Switch system
Keithley 7002	Switch system
Lakeshore 330	Auto-tuning temperature controller
Leybold Inficon PG3	Vacuum gauge controller
Measurement Group 2000	A/D converter
Neff system 470	Data acquisition system
Newport Corp. PMC200-P	Motion system
Newport Corp. PM500	Motion system
Precision filters 6201	Filter
Racal-Dana 1250	Switch system
Rod-L M30	Amp ground tester
Rod-L M100DC	Hipot tester
Rod-L M150AC	Hipot tester
Rod-L M300RT	Resistance tester
Rod-L M1088	GPIB Interface
Standard Research 245	Computer Interface/NIM Crate
Standard Research 250	Gated Integ. & Boxcar Averager
MODEL NUMBER	INSTRUMENT TYPE

Standard Research 510	Lock-in Amplifier
Standard Research 530	Lock-in Amplifier
Standard Research 630	Thermocouple monitor
Standard Research SR810	Lock-in Amplifier
Standard Research SR830	Lock-in Amplifier
Standard Research 850 DSP	Lock-in Amplifier
Teac RD-200T	PCM data recorder
Tektronix SG 5010	Low distortion oscillator
Tektronix SI 5010	Scanner
Tektronix 370A	Curve tracer
Tektronix 371A	Curve tracer
Vaisala QL 150	Sesor collector
Tektronix THS 710	Hand-held scope/OMM
Tektronix THS 720	Hand-held scope/OMM
LABVIEW VXI INSTRUMENT DRIVERS	
AD Data systems 230122	96-channel switch
Cal-Av 9100	Trigger Delay
Cytec CY/CX	Coaxial matrix
Cytec CY/8X8	Switch matrix
Cytec CY/48X48	Switch matrix
Cytec CY/32KC	Switch module
Cytec CY/128	Multiplexer
Datron Wavetek 1362	6.5 digit multimeter
Datron Wavetek 1362S	6.5 digit multimeter
EIP Microwave 1140A	Pulsed microwave synthesizer
EIP Microwave 1141A	Pulsed microwave synthesizer
EIP Microwave 1141A	Pulsed microwave synthesizer
EIP Microwave 1142A	Pulsed microwave synthesizer
EIP Microwave 1230A	Microwave counters
EIP Microwave 1231A	Microwave counters
Giga-tronics 58542	Peak and CW Power meter
Hewlett-Packard E1326A	5.5 digit multimeter
Hewlett-Packard E1328A	4 channel D/A converter
Hewlett-Packard E1330A/B	Quad 8-bit digital I/O
Hewlett-Packard E1333A	3 channel universal counter timer
Hewlett-Packard E1340A	Arbitrary function generator
Hewlett-Packard E1345A	16-channel relay multiplexer
Hewlett-Packard E1465A	16 by 16 relay matrix
Hewlett-Packard E1466A	4 by 64 relay matrix
Hewlett-Packard E1467A	8 by 32 relay matrix
Hewlett-Packard E1468A	8 by 8 matrix switch
Hewlett-Packard E1469A	4 by 16 matrix switch
Hewlett-Packard E1472A	50 ohm RF multiplexer
Hewlett-Packard E1473A	50 ohm RF multiplexer expander
Hewlett-Packard E1474A	75 ohm RF multiplexer
MODEL NUMBER	INSTRMENT TYPE
Hewlett-Packard E1475A	75 ohm RF multiplexer expander

Hewlett-Packard E1476A	64 channel thermocouple relay multiplexer
Hewlett-Packard E1740A	150 MHz time interval analyser card interface technology
Interface Technology IO100	Digital I/O module
Kinetic Systems V215	32-channel , 16 bit A/D converter
North Atlantic 5388	Synchro/resolver processor
Racal-Dana 1260-12	Relay actuator
Racal-Dana 1260-13	Signal switching/relay actuator module
Racal-Dana 1260-20	Power switching module
Racal-Dana 1260-30	multiplexer
Racal-Dana 1260-35	High-density signal
Racal-Dana 1260-40	Matrix
Racal-Dana 1260-45	High-density switch matrix
Racal-Dana 1260-54	1 GHZ Terminated RF Multiplexer
Racal-Dana 1260-64	18 GHz Microwave switch
Racal-Dana 1277 Series	Switching modules
Racal-Dana 2251	Universal Counter/Timer
Racal-Dana 2351	Time Interval Analyser
Schlumberger SI 1270	Frequency response analyser
Talon BE-64	Bus emulator/word generator
Tektronix VX4223	160 MHz universal counter/timer
Tektronix VX4234	Digital multimeter
Tektronix VX4236	6.5 digit Multimeter
Tektronix VX4240	Waveform digitiser/analyser
Tektronix VX4250	Waveform tester
Tektronix VX4286	32 channel analog/digital input module
Tektronix VX4332	40 channel 2-wire relay scanner master
Tektronix VX4334	24 channel 4-wire relay scanner master
Tektronix VX4353	32 channel SPST relay switch
Tektronix VX4355	24 channel DPST relay switch
Tektronix VX4356	20 channel DPDT relay switch
Tektronix VX4357	32 channel SPDT relay switch
Tektronix VX4363	32 channel SPST relay switch
Tektronix VX4365	24 channel DPST/SPDT relay switch
Tektronix VX4366	20 channel DPDT relay switch
Tektronix VX4367	32 channel SPDT relay switch
Tektronix VX4372	48 channel 2-wire reed relay scanner slave
Tektronix VX4374	24 channel 4-wire reed relay scanner slave
Tektronix VX4385	Matrix switch
MODEL NUMBER	INSTRUMENT TYPE
Tektronix VX4730	12 channel , 16 bit D/A converter
Tektronix VX4750	Function generator

Tektronix VX4790	Arbitrary waveform generator
Tektronix VX4801	40 line isolated digital I/O
Tektronix VX4802	80 line digital I/O
Tektronix/CDS 73A-256	12 channel 16 bit D/A converter
Tektronix/CDS 73A-270	Arbitrary pulse pattern generator
Tektronix/CDS 73A-308	Relay and high voltage logic driver
Tektronix/CDS 73A-332	40 channel 2-wire reed relay scanner master
Tektronix/CDS 73A-334	24 channel 4-wire reed relay scanner master
Tektronix/CDS 73A-342	Dual programmable resistance
Tektronix/CDS 73A-353	32 channel SPST relay switch
Tektronix/CDS 73A-355	24 channel DPST relay switch
Tektronix/CDS 73A-356	20 channel DPDT relay switch
Tektronix/CDS 73A-372	48 channel 2-wire reed relay scanner slave
Tektronix/CDS 73A-374	24 channel 4-wire reed relay scanner slave
Tektronix/CDS 73A-411	40-line isolated digital I/O
Tektronix/CDS 73A-412	80-line digital I/O
Tektronix/CDS 73A-453	1-channel MIL-STD-1553A/B bus simulator
Tektronix/CDS 73A-455	2-channel MIL-STD-1553A/B bus simulator
Wavetek 137520 MHz	Arbitrary function generator
Wavetek 139550 MHz	Arbitrary function generator
LABVIEW SERIAL AND CAMAC INSTRUMENT DRIVERS	
SERIAL INSTRUMENT DRIVERS	
Aerotech Unidex 100	Motion controller
Analog devices 6B series	Signal conditioning I/O modules
Analog devices μ Mac 6000	Modular I/O processor
Eurotherm 808/847	Digital controller
Fluke PM99	Scope/meter
Fluke 97	Scope/meter
Fluke 2625A Hydra	Data acquisition unit
HBM DMC	Digital Amplifier system
HBM MGC	Measuring amplifier system
Mettler PM4600	Balance
Ohaus GT Series	Balance
Spex CD2A	Spectrometer Drive system
Tektronix 222	Digitizing oscilloscope
CAMAC INSTRUMENT DRIVERS	
BiRa 5301	ADC
MODEL NUMBER	INSTRUMENT TYPE
DSP technology 2010/2012	Digitizer
JoergerS12	Scaler
Joerger TR Series	Digitizer

Jorway 60A	Input register
KineticSystems 3074	Output register
KineticSystems 3075	Output register
KineticSystems 3112	DAC
KineticSystems 3361	Stepper motor controller
KineticSystems 3514	ADC
KineticSystems 3516	ADC
KineticSystems 3525	Temperature Monitor
KineticSystems 3988	GPIB Crate controller
KineticSystems 4010	Transient recorder
KineticSystems 4020	Transient digitizer
LeCroy 2228A	TDC
LeCroy 2249SG	ADC
LeCroy 2256AS	Digitizer
LeCroy 4434	scaler
LeCroy 6810	Digitizer
LeCroy 8210	Digitizer
LeCroy 8232	ADC
LeCroy 8901/8901A	GPIB Crate controller
LeCroy TR8818	Transient recorder
LeCroy TR8837F	Transient recorder
LRS 2550B	Scaler

POGLAVLJE 11

PRIMJERI VIRTUALNIH INSTRUMENATA

Virtualni instrumenti su sinergicka kombinacija DAQ proizvoda (modula za prikupljanje podataka), software-a i PC-eva.

U prvom dijelu ovog poglavlja, analiziracemo slijedecih pet virtualnih instrumenata:

- osciloskop (VirtualBench - Scope)
- dinamicki analizator signala (VirtualBench - DSA)
- funkcionalni generator (VirtualBench - FG)
- digitalni multimetar (VirtualBench - DMM)
- data logger (pohranjivanje podataka) (VirtualBench - Logger)

Fleksibilnost ovakvih programabilnih instrumenata ogleda se u tome da ne samo sto pet razlicitih tipova instrumenata mogu biti konfigurisani i realizovani na istom hardware-u PC-ja, nego takodjer ti razliciti instrumenti mogu biti realizovani simultano tako da PC sa DAQ modulima moze istovremeno ostvarivati vise instrumentalnih funkcionalnosti.

Prednji paneli ce po izgledu biti sto slicniji njihovim fizickim uzorima tako da ovaj korisnicki interfejs bude prepoznatljiv za koristenje.

Instrumenti ce se tako konfigurisati da odmah nakon ukljucenja se loaduje (puni) program koji ce pripremiti PC za funkcije instrumenta tako da se korisniku doima kao da mu je to originalna funkcija i namjena.

Razmotricemo i specificnost koristenja portabilnog hardware-a (koji se kao i dedicated instrument moze ponjeti na test i mjernu lokaciju koja nije uvijek u laboratoriji), kao sto su notebooks i laptop portabilni PC-evi sa njihovim plug-in modulima tipa PCMCIA (sada PC cards) DAQCards za I/O module ili DAQ proizvodima koji se povezuju sa PC-em preko paralelnog porta.

Tradicionalni instrumenti imaju tri bazicne komponente: akviziciju i kontrolu te akvizicije, analizu podataka, i prezentaciju podataka. Savremeni trendovi u razvoju notebook PC-eva su donjeli kompjutere koji prevazilaze mogucnosti procesiranja i prikazivanja mnogih instrumenata. Tehnologija danasnjih plug-in modula za prikupljanje podataka (DAQ), PC kartica (ranije nazivanim PCMCIA kartice za notebook-ove), kao i uredjaji koji se plaguju na paralelni port PC-ja su uporedljivi sa kontrolnim i akvizicionim mogucnostima tradicionalnih instrumenata.

Koristenje VirtualBench (virtualnog mjernog kompleta) ce biti skoro identicno nacinu na koji koristimo standardne autonomne mjerne instrumente, ali prednosti pocinju u procesiranju izmjerenih vrijednosti, nacinu prikazivanja, kao i u mogucnostima pohranjivanja rezultata mjerenja kada koristimo PC bazirani mjerni uredjaj.

VirtualBench instrumenti pune i pohranjuju podatke i valne oblike na disk u istom obliku zapisa u kojem mogu biti korišteni od strane popularnih spreadsheet programa kao i procesora teksta. Mogućnosti generisanja izvještaja o mjerenju također su dodatna mogućnost virtualnog instrumenta sa mogućnostima dodavanja vremenskih podataka (kada je mjerenje obavljeno), komentara, imena ispitivaca itd. Nadalje, ovi valni oblici signala mogu biti istampani/iscrtni na stampacu/ploteru spojenom na PC.

Nadalje manuali za instrument i help file-ovi su online, instalisani na PC-ju prema tome lako pristupacni korisniku virtualnog instrumenta u svakom trenutku vremena.

Sistemske zahtjevi

Prema preporukama Proizvođača software-a **NI** zahtjevi na PC na kojem se izvršava program virtualnog instrumenta (VirtualBench) su relativno skromni i sastoje se u slijedećem :

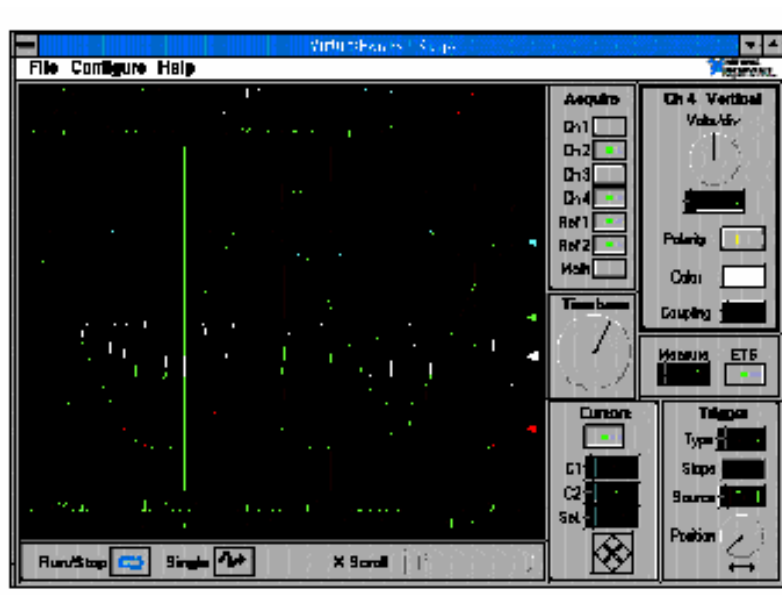
Windows95/98 (ili Windows 3.1) Operativni sistem, 12 MB RAM, 20 MB hard disk prostora.

VirtualBench software-ski paket od **NI** sadrži dakle slijedeće programske module : VirtualBench-Scope, VirtualBench-DSA, VirtualBench-FG, VirtualBench-DMM, VirtualBench-Logger.

VirtualBench-Scope

Akvizicija signala sa visokom tačnošću

VirtualBench-Scope nudi real-time brzine uzorkovanja (sampliranja) do 1.3 MHz (ili do 20 MHz koristeći tehniku uzorkovanja ekvivalentnog vremena), sa propusnim opsegom (bandwidth) do 1.6 MHz, i vertikalnom rezolucijom do 16 bita (zavisno od DAQ hardwarea koji se koristi sa VirtualBench-Scope-om). VirtualBench-Scope ima četiri kanala sa opsegom osjetljivosti od 1 mV/div do 10 V/div, vremensku bazu u opsegu 10 ns/div do 100 ms/div, i dužinu zapisa od 500 do 50.000 tacaka.



Fleksibilno trigerovanje

VirtualBench-Scope trigeruje i digitalne i analogne signale sa bilo kojim tipom DAQ modula, cak i u slucajevima kada modul nema analogno trigerovanje ugradjeno u hardware modula. Vi mozete podesiti triger point da kaptira (ulovi) signal prije i nakon tacke trigerovanja. Trigeri mogu biti pozitivnog ili negativnog nagiba, i rastuce ili opadajuce ivice.

Automatske matematske i funkcije analize

VirtualBench-Scope prevazilazi osnovne matematske funkcije sabiranja, oduzimanja, i mnozenja i nudi 17 dodatnih real-time mogucnosti valnih oblika, ukljucujuci VDC, VRMS, VMAX, VMIN, $V_{\text{peak-to-peak}}$, VPulse-Top, VPulse-Base, amplituda, prebacaj (overshoot), podbacaj (undershoot), frekvenca, period, kasnjenje impulsa, vrijeme porasta (risetime), vrijeme pada (falltime), sirina impulsa (pulsewidth), i grupna brzina (slew rate). Sa kurzorima VirtualBench-Scope-a, mozemo mjeriti razliku izmedju napona ili vremena u dvije tacke prikupljenog (akviziranog) i pohranjenog valnog oblika.

Selekcija hardwarea

Mada VirtualBench-Scope funkcioniра sa mnogim DAQ modulima sa analognim ulazom, moduli E serije sa analognim trigerovanjem nude beneficije sampliranja sa ekvivalentnim vremenom i boljeg vremena smirenja (settling) kada se skanira vise kanala. Jeftini DAQ moduli, kao Lab-PC+ nude nizi propusni opseg (bandwidth), 12 bitnu vertikalnu rezoluciju. Za portabilne aplikacije (sa notebookom) PCMCIA DAQCard i moduli za paralelni port DAQpads su idealni.

Opis karakteristika (sa DAQ modulom AT-MIO-16E-1)

- DC do 1.6 MHz propusni opseg
- 20 MHz brzina uzorkovanja
- 12 bitna vertikalna rezolucija
- 4 kanala

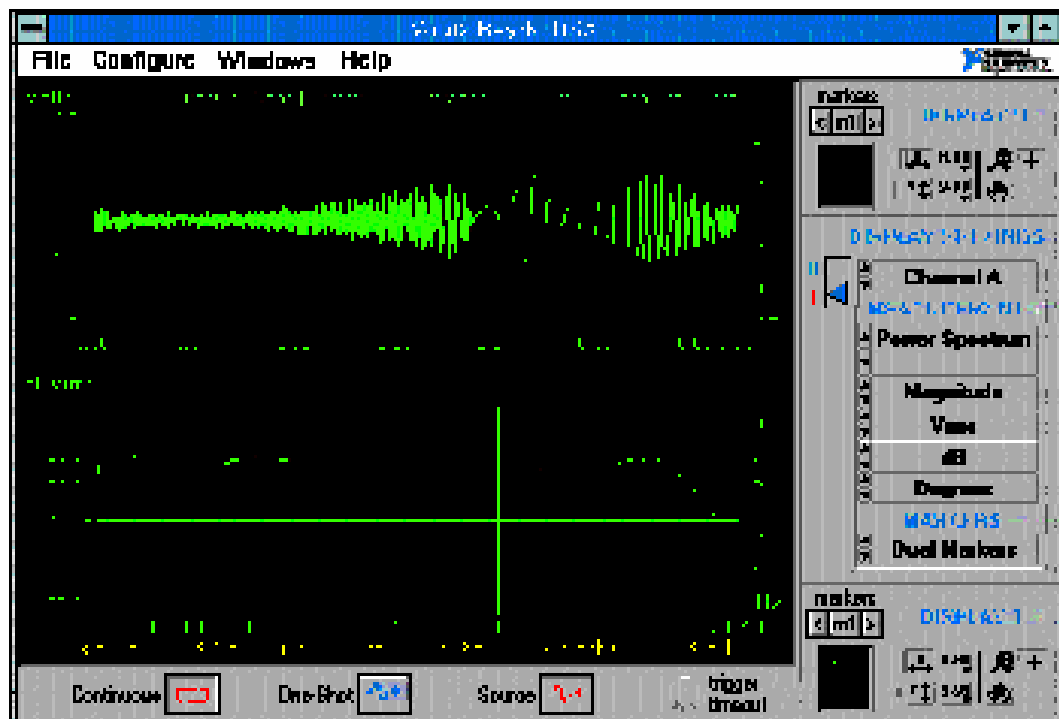
- kurzori za Δt i Δy mjerenje
- automatska statisticka mjerenja
- matematske funkcije
- prikazi (displays) do 16 miliona boja
- pohranjivanje zapisa do 50,000 tacaka
- 4 trigerska moda
- pohrani/pozovi (save/recall) konfiguracione podatke
- pohrani/pozovi (save/recall) valne oblike
- sposobnost iscrtavanja valnih oblika kao i stampanja konfiguracionih podataka

Virtualni dinamicki analizator signala za Windows 95/3.1

VirtualBench-DSA

Opis

Dinamicki analizatori signala koriste digitalno sampliranje i tehnike brze Fourier-ove transformaciju (FFT), da analiziraju nisko-frekventne signale u oblastima kao sto su akustika, sum (noise), vibracije i govor. VirtualBench-DSA je fleksibilni, FFT bazirani analizator za provodjenje vremenske, spektralne i analize signala u mrezama..



Karakteristike (sa DAQ modulom AT-DSP 2200 i AT-A2150)

- DC do 20 KHz propusni opseg
- 51.2 KS/s (hiljada samplova/sec) brzina samplinga
- 93 dB dinamički opseg sa zaštitom od aliasinga
- precizni 16 bitni ADC sa simultanim samplingom (uzorkovanjem)
- glatkost amplitude (amplitude flatness) do ± 0.025 dB
- do dva analogna ulazna kanala
- jedan precizni analogni izlaz
- 850 linija frekventna rezolucija (0.025 dB propusni opseg); 920 linija frekventna rezolucija (3 dB p. opseg)
- kurzori za Δt i Δy mjerenje
- mjeri THD , sadržaje harmonika, frekventni odziv, spektar snage (power spectrum), amplitudni spektar, koherentnost, i kros spektar
- do 16 miliona boja na displayu
- dužina zapisa do 2048 tacaka
- analogni i digitalni modovi triggerovanja
- pohrani/pozovi (save/recall) konfiguracione podatke
- valni oblici se pohranjuju na disk
- sposobnost iscrtavanja valnih oblika kao i stampanja konfiguracionih podataka

Spektralna mjerenja velike brzine

VirtualBench-DSA mjeri totalno harmonicko iskrivljenje (THD - total harmonic distortion), sadržaj harmonika, frekventni odziv, spektar snage, amplitudni spektar, koherentnost, i kros spektar. Ova mjerenja se prikazuju u jedinicama V, V^2 , V_{rms}^2 , V_{rms} , V/Hz, V^2/Hz , V_{rms}^2/Hz , dBVrms, dBVpk, dBm, dB, dBVrms/Hz, stepeni, radijani, sekunde, i V_{rms}/Hz zavisno od mjerenja i same aplikacije. Ugradjene funkcije **prozora** (windowinga) uključuju Hanning, ravni vrh (Flat top), Hamming, Blackman Harris, Blackman i Extract Blackman da bi se samnjili efekti **curenja spektra** (spectral leakage).

Analiza u vremenskom domenu

VirtualBench-DSA prikazuje vremenske funkcije kao nisko-frekventni osciloskop radi posmatranja signala simultano i u vremenskom i frekventnom domenu. Kurzori olaksavaju ekstrakciju ključnih parametara performanse kontrolnog sistema, kao što je prebacaj (overshoot) , vrijeme porasta (rise time), vrijeme smirenja (settling time), i kasnjene (delay time).

Odziv na stimulus (podsticaj)

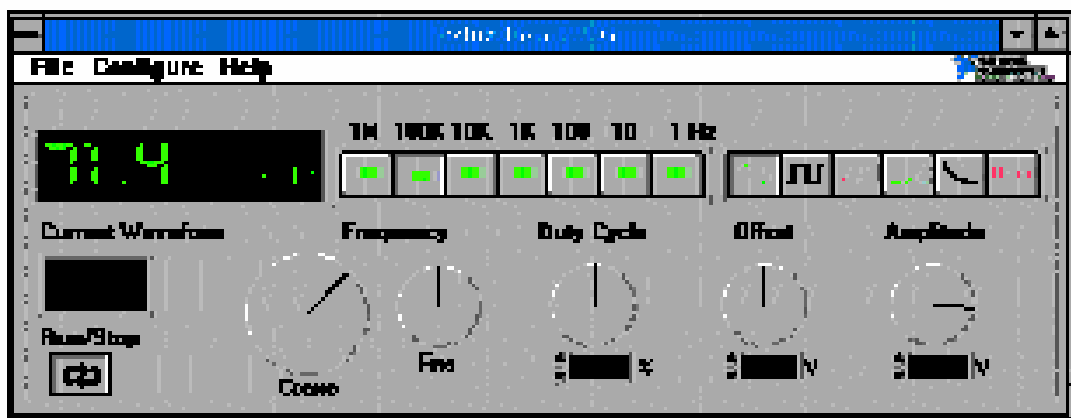
Moguće je generirati valne oblike iz VirtualBench-DSA za analizu mreže koristeći izlazni izvor. Na raspolaganju su : pseudoslučajni sum, fiksna sinusoida i chirp.

VIRTUALNI GENERATOR FUNKCIJA I MULTIMETAR

VirtualBench-FG

Opis

VirtualBench-FG je niskofrekventni generator funkcija za generiranje i sintezu stimulus signala.



Karakteristike (sa DAQ modulom AT-DSP 2200)

- sinusni, pravougaoni, trouglasti, pila, i valni oblik definisan od strane korisnika
- 2 Hz do 20 kHz opseg frekvencije sinusnog signala
- preciznost 16-bitna rezolucija
- glatkost amplitude (amplitude flatness) do ± 0.025 dB
- podesivost duty ciklusa (odnosa pozitivne i negativne poluperiode)
- pohrani/pozovi (save/recall) konfiguracione podatke
- sposobnost iscrtavanja valnih oblika kao i stampanja konfiguracionih podataka

Generiranje signala

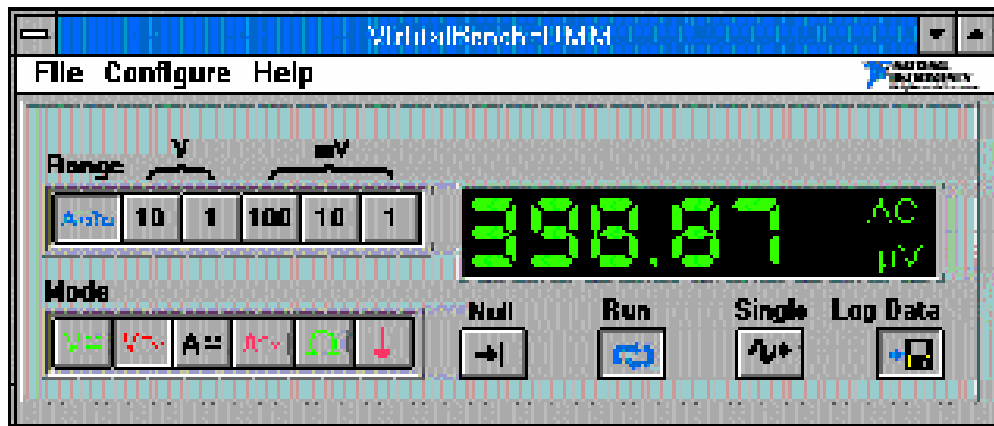
VirtualBench-FG ima unaprijed definisane valne oblike za generiranje sinusa, pravougaonog, trouglastog, eksponencijalnog sa porastom i opadanjem pile, valnog oblika.

Dodatno, VirtualBench-FG generira valne oblike koje definira korisnik do duzine od 2048 tacaka, ukljucujuci valne oblike pohranjene na disk koristeći VirtualBench-Scope ili VirtualBench-DSA, ili valne oblike definisane sa poljem (array) kreiranim spreadsheetom ili procesorom teksta. Kontrolni elementi na prednjem panelu omogućavaju podesenje frekvencije, ofseta, amplitude, i duty ciklusa valnog oblika.

VIRTUALNI MULTIMETAR I FUNKCIONALNI GENERATOR

VirtualBench-DMM

VirtualBench-DMM je jeftina virtualna verzija multimetra za prikupljanje podataka, laboratorijska mjerenja i razvoj.



Karakteristike (sa DAQ modulom AT-MIO-16XE-50)

- 4 digita multimetar
- VDC, VAC, IAC, IDC, i temperatura
- automatsko biranje opsega (autoranging)
- podaci mjerenja se pohranjuju na disk
- pohrani/pozovi (save/recall) konfiguracione podatke
- sposobnost iscrtavanja valnih oblika kao i stampanja konfiguracionih podataka

Mjerne mogucnosti

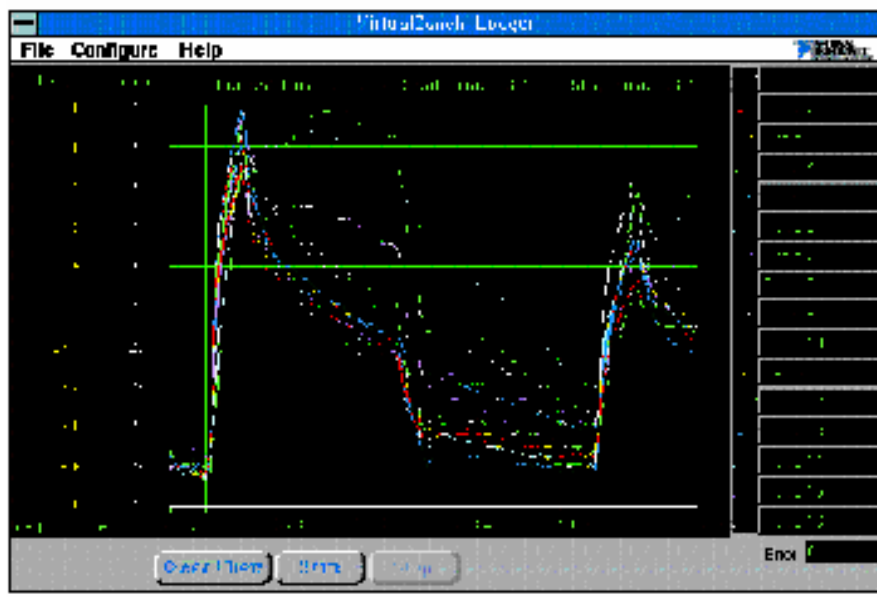
VirtualBench-DMM mjeri DC i AC napon, DC i AC struju, otpor i temperaturu. Ulazni opsezi su bilo sa automatskim izborom ili ih bira korisnik u opsegu od 1 mV do 10V. Temperaturna mjerenja se ostvaruju sa termoelementima, termistorima ili IC senzorima. Sa jednostavnim klikom na taster misa korisnik pohranjuje podatke mjerenja na disk radi kasnije analize.

DATA LOGGER

VirtualBench-Logger

Opis

VirtualBench-logger je visekanalni pohranjivac podataka (data logger) za namjene pohranjivanja i prikazivanja nisko-frekventnih signala kao sto su temperatura, pritisak i napon.



Karakteristike (sa DAQ modulom AT-MIO-16XE-50 sa SCXI)

- 384 kanala data logger/recorder
- do 16 kanala istovremeno prikazanih kao trend (zapis na traci pisaca : strip-u)
- do 16 miliona boja
- rezolucija 16 bita
- opseg do 250 V
- 250 Vrms izolacioni napon medju kanalima
- 10 kHz ulazni propusni opseg (bandwidth)
- izabirljivi hardwareski i softwareski filteri za sum
- mogucnosti zapisivanja podataka o temperaturi , naponu, struji
- skaliranje u inzenjerskim jedinicama
- y osa sa dvije skale
- podaci se pohranjuju na disk
- pohrani/pozovi (save/recall) konfiguracione podatke
- sposobnost iscrtavanja valnih oblika kao i stampanja konfiguracionih podataka

Mogucnosti mjerenja

VirtualBench-logger prikuplja do 384 kanala podataka i pri tom prikazuje 16 kanala. Sa menijima, mozete konfigurirati parametre svakog kanala ukljucujuci ime kanala, konverzione faktore ($Mx + B$), mjerne jedinice, opseg prikaza, i skale na prikazima.

Ugradjeni konverzioni faktori za B, E, J, K, N, R, S i T tipova termoelemenata, i RTD-ova pojednostavljaju mjerenja temperature. Mjerenja sa termoelementima koriste polinomijalne jednacine za linearizaciju i automatski izvrsavaju kompenzaciju hladnog kraja termoelementa. Automatsko skaliranje struje je na raspolaganju kod mjerenja strujnih signala. Softwareski izabirljivi filter za sum na 50 ili 60 Hz pomaze prikupljanju podataka u ambijentima velikog suma i

dopunjuje se sa filterima za sum koje posjeduju i neki DAQ hardwareski moduli. Dvostruka y osa omogućava monitorovanje signala razlicite prirode i opsega na istom prikazu.

PRIMJERI VI MODULA

UVOD

U primjerima koji slijede demonstrirace se snaga LabVIEW-a kao grafickog programskog jezika sa kojim gradimo virtualne instrumente (VI) umjesto pisanja programa. Kreiranje korisnickog interfejsa (user interfaces) u vidu prednjeg panela je brzo i lako, dajuci mogucnost interaktivne kontrole softwareskog sistema.

Specificiranje funkcionalnosti se ostvaruje putem, za inzenjere vrlo intuitivnog i prirodnog postupka, asambliranjem blok dijagrama. U okviru LabVIEW-a ovako asambliran blok dijagram je stvarni program, tako da se izbjegava vremenski dosta intenzivan napor pretvaranja ideja izlozenih u vidu blok dijagrama u kripticni izvorni kod (source code) nekog programskog jezika.

Na prednji panel VI koja se dizajnira, korisnik smjesta kontrolne i indikacione elemente za sistem, birajuci objekte iz menija za kontrolne elemente (Controls). Ovi objekti ukljucuju numericke displeje, mjerne instrumente, termometre, LED diode, grafove, chartove itd. Nakon razmjestaja kontrolnih elemenata na prednjem panelu, korisnik ih moze aranzirati i prilagoditi (customize) prema korisnickom intefejsu koji njemu odgovara.

Kada je VI kompletna, prednji panel ce se koristiti za kontrolu instrumenta, dok se VI izvrsava, klikanjem na prekidac, pomjeranjem kliznog prekidača (slide switch), zumiranjem u graf, ili unosjenjem vrijednosti pomocu tastature. Pri tome, panel je neprekidno osvjezavan i animiran programom koji se izvrsava, dajuci korisniku real-time feedback (odziv u realnom vremenu), iz VI koju je razvio.

Programiranje VI je dakle kroz konstruiranje blok dijagrama bez potrebe da dizajner brine o mnogim sintaktickim detaljima konvencionalnih programskih jezika. Korisnik bira objekte (ikone) iz funkcionalnih menija i povezuje ih sa zicama da bi prenio podatke od jednog bloka do drugoga. Ovi blokovi su u opsegu od jednostavnih aritmetickih funkcija do naprednih rutina za akviziciju i analizu podataka. Blokovi mogu takodjer ukljucivati operacije sa mrezom i file I/O operacije kojima se pohranjuju ili uzimaju podaci u ASCII, binarnom i spreadsheet formatu. Korisnik moze takodjer pozivati programe pisane u drugim programskim jezicima direktno iz blok dijagrama. Kod LabVIEW -a blok dijagram je izvorni kod (source code).

LabVIEW ima modularni dizajn, to znaci da korisnik moze otvoriti i izvrsavati mnoge blokove kao samostalne (stand-alone) VI-eve.

Dizajner moze funkcionalno rastaviti softwareski sistem u subkomponente tj. SubVI-eve. On moze interaktivno testirati ove subVI i koristiti ih kao ikone da izgradi sofisticirane slojeve VI-eva. Kreirajuci ikonu za VI koju razvija i koristeci je u blok dijagramu druge VI, dizajner moze sakriti kompleksnost dijagrama nizih nivoa, zadrzavajuci pri tome pristup medju vrijednostima tih nivoa.

Kao kompletan programski jezik, LabVIEW nudi programske strukture kao što su For i While konture, i Case iskazi za sekvencijalne, repetitivne i operacije granjanja. Ove strukture se pojavljuju kao graficke granice koje okružuju ikone koje one kontrolisu.

LabVIEW koristi metod programiranja poznat kao dataflow (tok podataka), koji dizajnera oslobadja od linearne arhitekture tekst baziranih programskih jezika. Posto je redoslijed izvršenja u LabVIEW odredjen sa tokom podataka izmedju blokova a ne redoslijedom sekvencijalnih linija teksta programskog koda, dizajner moze kreirati dijagrame koji imaju simultane operacije i na taj nacin ostvariti multitasking. Dakle LabVIEW je multitasking sistem koji izvrsava visestruke staze izvršenja programskih dijelova (execution threads), visestruke VI , zajedno sa drugim aplikacijama van VI modula.

U mnogim aplikacijama brzina izvršenja je kriticna. LabVIEW je jedini graficki programski sistem sa kompajlerom koji generira optimizirani kod sa brzinama izvršenja usporedivim sa kompajliranim programima pisanim u C i C++.

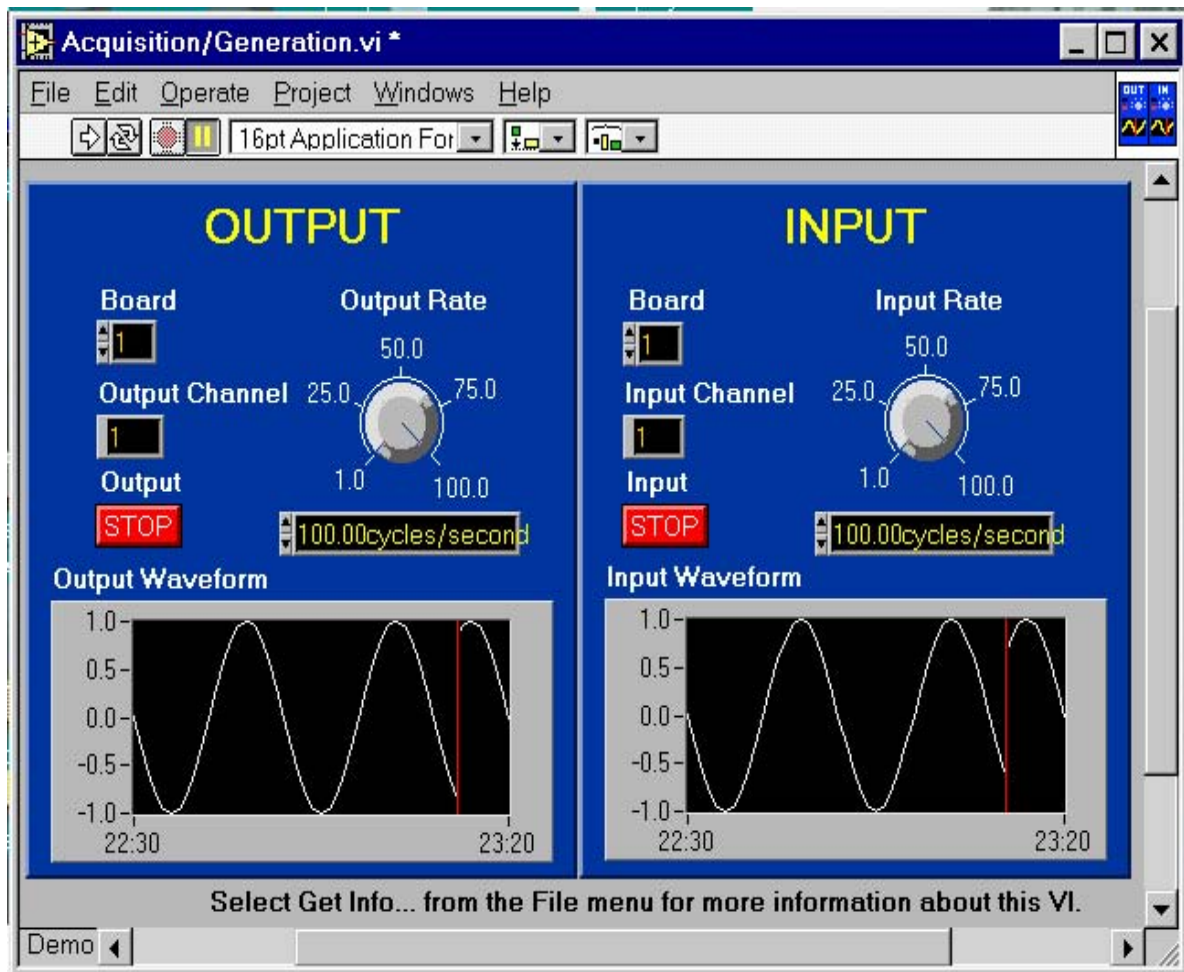
Za razliku od drugih programskih jezika koji vas prisiljavaju da prvo kompilirate izvorni kod da bi nasli gresku, LabVIEW odmah indicira nekorekne spojeve zica, i izlistava probleme u prozoru gresaka (Error window). Nadalje da olaksa debaging (trazenje i korekciju programskih gresaka), LabVIEW vrsi naglasavanje toka izvršenja programa, izvršenje korak po korak (single step), probe za zice (wire probe) za ocitavanje vrijednosti podataka koji teku kroz zice, i tacke prekida programa (breakpoint), tako da dizajner moze pratiti i nadzirati tok izvršenja programa kroz blok dijagram.

MODUL AKVIZICIJE/GENERIRANJA SIGNALA

IME VI: *Acquisition/Generation.vi*

Ova VI pokazuje multitasking (vise programskih cjelina-taskova se simultano izvrsava) mogucnosti LabVIEW-a.

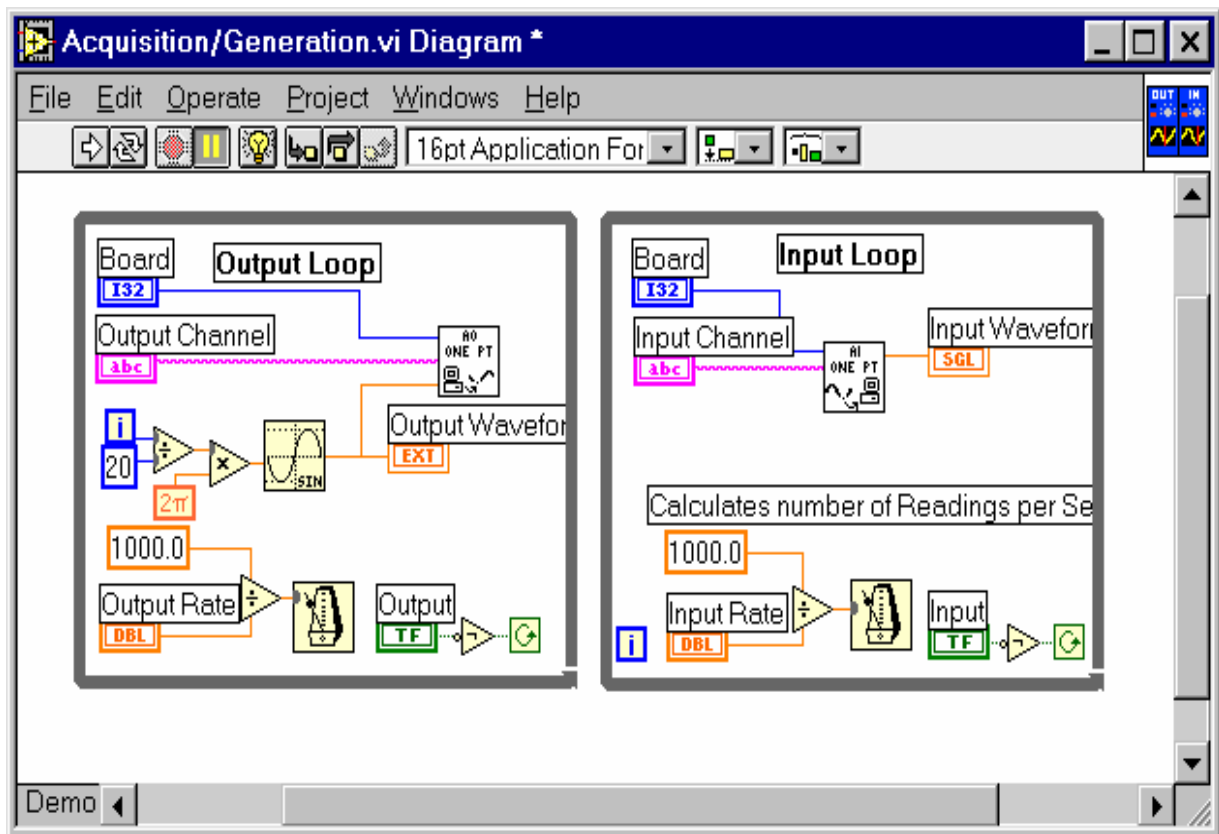
Jedna kontura izvrsava simulaciju analognog izlaza. Brzina sa kojom se generise izlaz moze biti kontrolisana sa **Output rate** dugmetom na lijevom dijelu prednjeg panela (vidjeti narednu sliku).



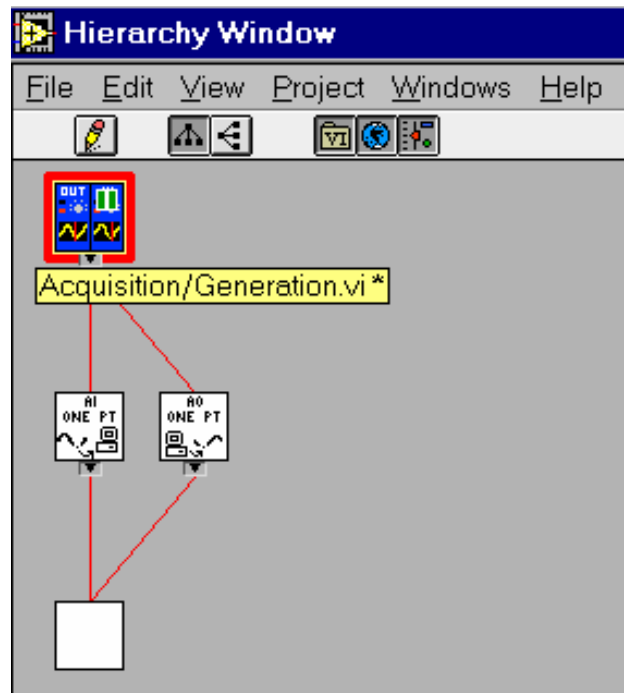
Druga kontura nezavisno izvrsava simulaciju analognog ulaza, ucitavajući vrijednosti koje su generirane u izlaznoj konturi. Brzina sa kojom se ucitavaju vrijednosti se moze kontrolisati pomocu **Input rate** dugmeta na desnom dijelu prednjeg panela. Primjetimo da mozemo nezavisno zaustaviti svaku konturu. Ako zaustavimo ulaznu konturu izvršenja, vrijednosti se nece ucitavati i strip chart (emulacija pisaca sa pokretnom papirnom trakom) se vise nece azurirati. Istovremeno, izlazna kontura ce nastaviti da se izvrsava.

Ako zaustavimo izlaznu konturu, nece se generirati nove vrijednosti, tako da ulazni zapis na chartu nece vise prikazivati sinusni valni oblik. Umjesto, prikazat ce se ravna linija.

Blok dijagram ove VI sa prikazanim prednjim panelom na prethodnoj slici, dat je na narednoj slici:

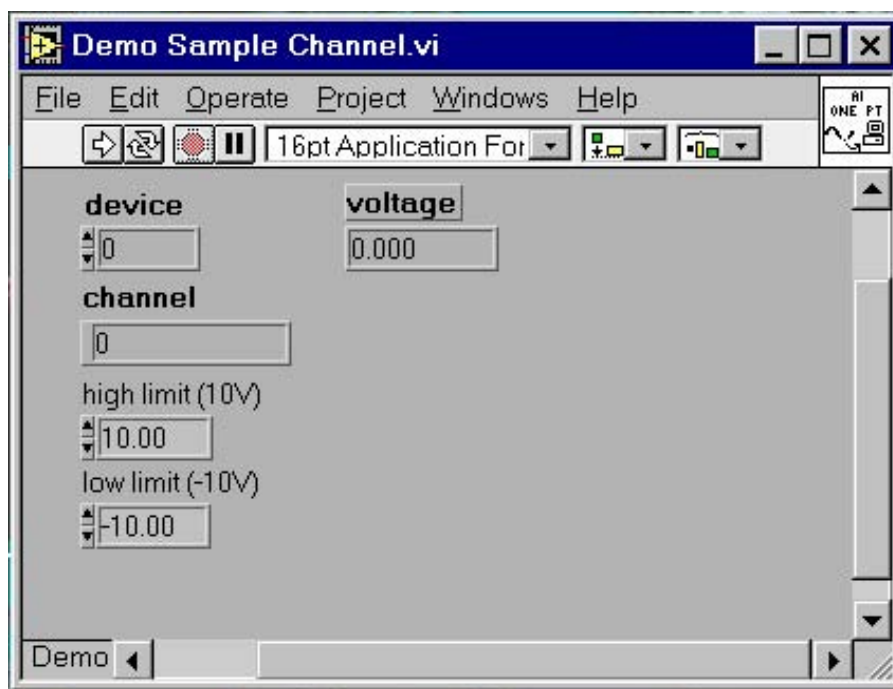


Hijerarhija u strukturi ove VI je data na narednoj slici u takozvanom hijerarhijskom prozoru (*Hierarchy window*):

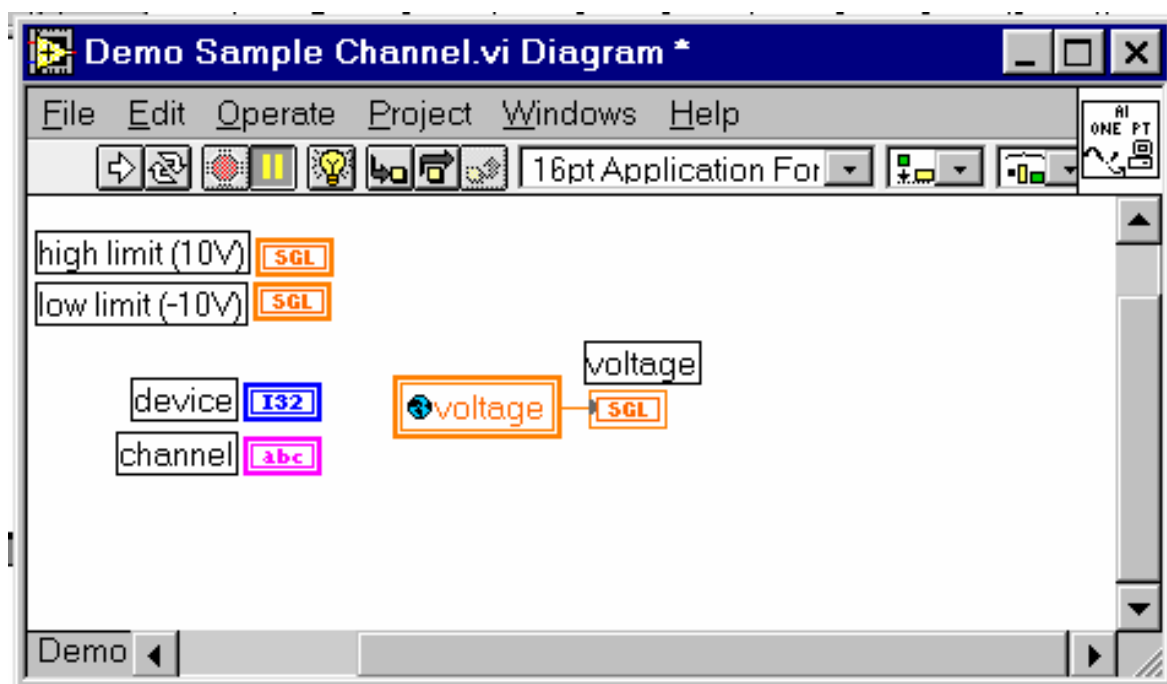


Kao što se vidi sa prethodne slike na najvišem nivou hijerarhije je Acquisition/Generation VI, koja poziva dvije subVI: Demo Sample Channel.vi (AI One Point) i Demo Update Channel.vi (AO One Point). Sa svoje strane ove subVI pozivaju svoju subVI: Global voltage.vi.

Demo Sample Channel.vi simulira mjerene signale priključene na specificirane kanale i vraća mjerena u polje (array) napona ili binarnih vrijednosti. Prednji panel ove subVI je prikazan na narednoj slici:

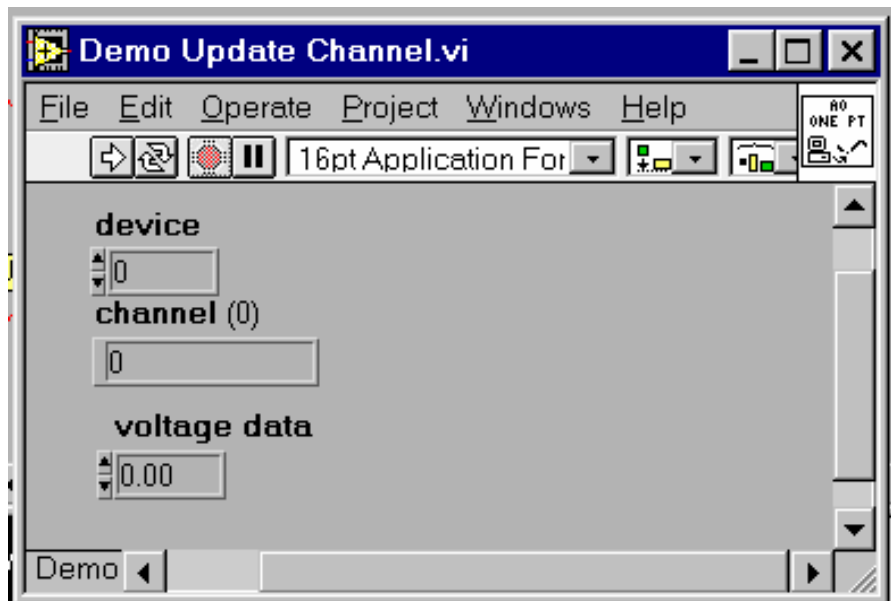


a blok dijagram ove subVI na slijedećoj :

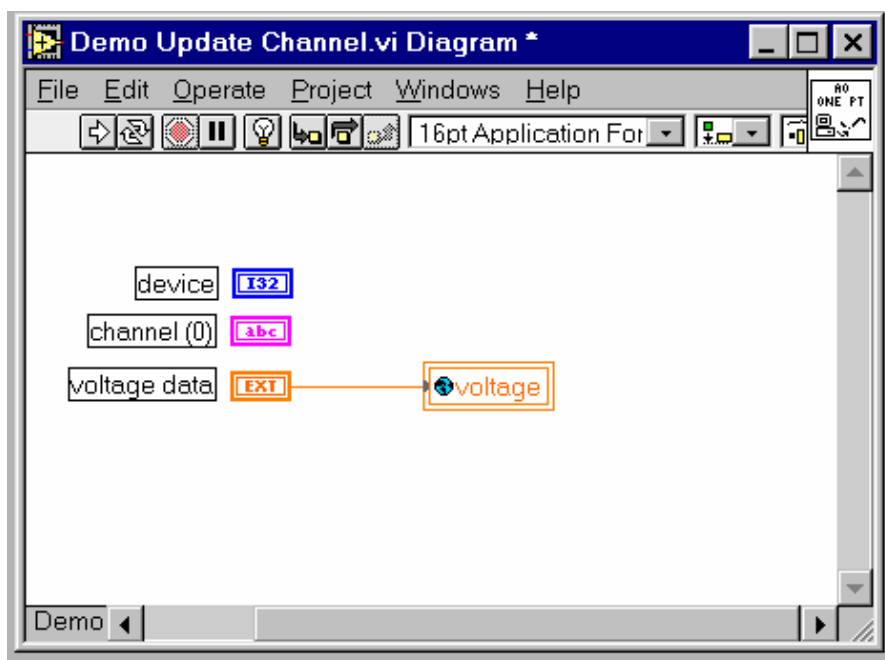


Demo Update Channel.vi je bazirana na **AO Write One Update** koja upisuje vrijednost napona na svaki od specificiranih analognih izlaznih kanala.

Prednji panel ove VI je prikazan na slijedećoj slici:



a blok dijagram ove subVI na slijedećoj :



AO Write One Update se koristi da izvrši trenutacni update (azuriranje) grupe od jednog ili više kanala. Ako se smjesti ova VI u konturu da bi upisivala više od jedne vrijednosti na istu grupu kanala, ozicite terminal za iteraciju konture sa VI iteracionim ulazom. Kod iteracije 0, VI poziva **AO Config** da konfigurira kanale grupe i hardware, zatim poziva **AO Single update** da upise napon na izlazne kanale. Pri narednim iteracijama VI poziva samo **AO Single Update**, izbjegavajući nepotrebno opterećenje programa sa konfiguriranjem. Ako

pozovemo **AO Single Update** samo jedanput, da upise jednu vrijednost na svaki kanal, tada je moguće ostaviti iteracioni ulaz neozicenim. Default vrijednost iteracije od 0, kaže da VI treba da izvrši konfiguriranje prije nego što upise bilo kakve podatke.

Prije nego što bilo šta učini, **AO Write One Update** provjerava ulazni klaster greske da odredi da li se greska već pojavila. Ako jeste, VI ne čini ništa, nego prenosi informaciju o gresci neizmjenjenu na **error out**. Kada se koristi **AO Write One Update u While** konturi, treba zaustaviti konturu ako je Bulov status u **error out** klasteru TRUE. Može se koristiti i **General Error handler VI** da dekodira informaciju o gresci i obavjesti korisnika.

Ulazne vrijednosti u ovaj modul su:

([i32]) device: broj plug-in akvizicionih modula (mora korisnik specificirati)

([string]) channels: specificira broj analognih izlaznih kanala za grupu i task. Ne može se doznaciti isti kanal više od jednoj grupi. Default je kanal 0.

([sgl]) voltage data: jedno dimenzionalno polje koje sadrži podatke koji će biti upisani u bafer u jedinicama [V], jedna vrijednost za svaki kanal dat u polju kanala.

([cluster]) limit setting: polje klastera, u kojem svaki element polja specificira granicne postavne vrijednosti za svaki kanal.

Svaki klaster sadrži slijedeće parametre:

([sgl]) high limit: jednak iznosu referentnog napona. Ovo je maksimalni napon koji DAC može proizvesti.

([sgl]) low limit: je bilo 0.0V (ako je riječ o unipolarnom opsegu) ili je vrijednost jednaka ali suprotnog znaka gornjoj granici (high limit) što implicira da je bipolarni opseg.

([u16]) reference source:

0: ne mijenjati vrijednost referentnog izvora

1: Interni

2: Eksterni

(i32) iteration: kontrolira kada **AO Write One Update** izvršava inicijalizaciju. Ako je iteracija 0, VI konfigurira grupu kanala i hardware, a nakon toga upisuje podatke.

Ako je iteracija veća od nule, **AO Write One Update** pretpostavlja da je konfiguracija već izvršena i samo upisuje podatke. Obično se ovaj ulaz ozicava sa terminalom iteracije While konture.

(cluster) error in: klaster koji opisuje bilo koji uslov greske prije izvršenja ove VI. Default ulaz za ovaj klaster je **no error**. Ako se greska već pojavila, ova VI ne čini ništa, izuzev prenošenja informacija sa **taskID in** i **error in** na **taskID out** i **error out**. Greska u klasteru sadrži slijedeće parametre:

(bool) status: kod greske pridružen gresci. Vrijednost 0 znači nema greske, negativna vrijednost je neka greska a pozitivna vrijednost je upozorenje.

(string) source: indikacija o tome gdje se greska pojavila, obično je to ime VI.

Izlazne vrijednosti:

(*cluster*) *error out*: klaster koji sadrži informaciju o gresci. Ako error in indicira postojanje greske, **error out** sadrži istu informaciju o gresci. Inace, opisuje status greske ove VI.

Koristite **General Error Handler** VI da ispitete gresku na kraju vasesg blok dijagrama.

Error out klaster sadrži slijedece parametre:

(*bool*) *status*: TRUE ako se greska pojavila.

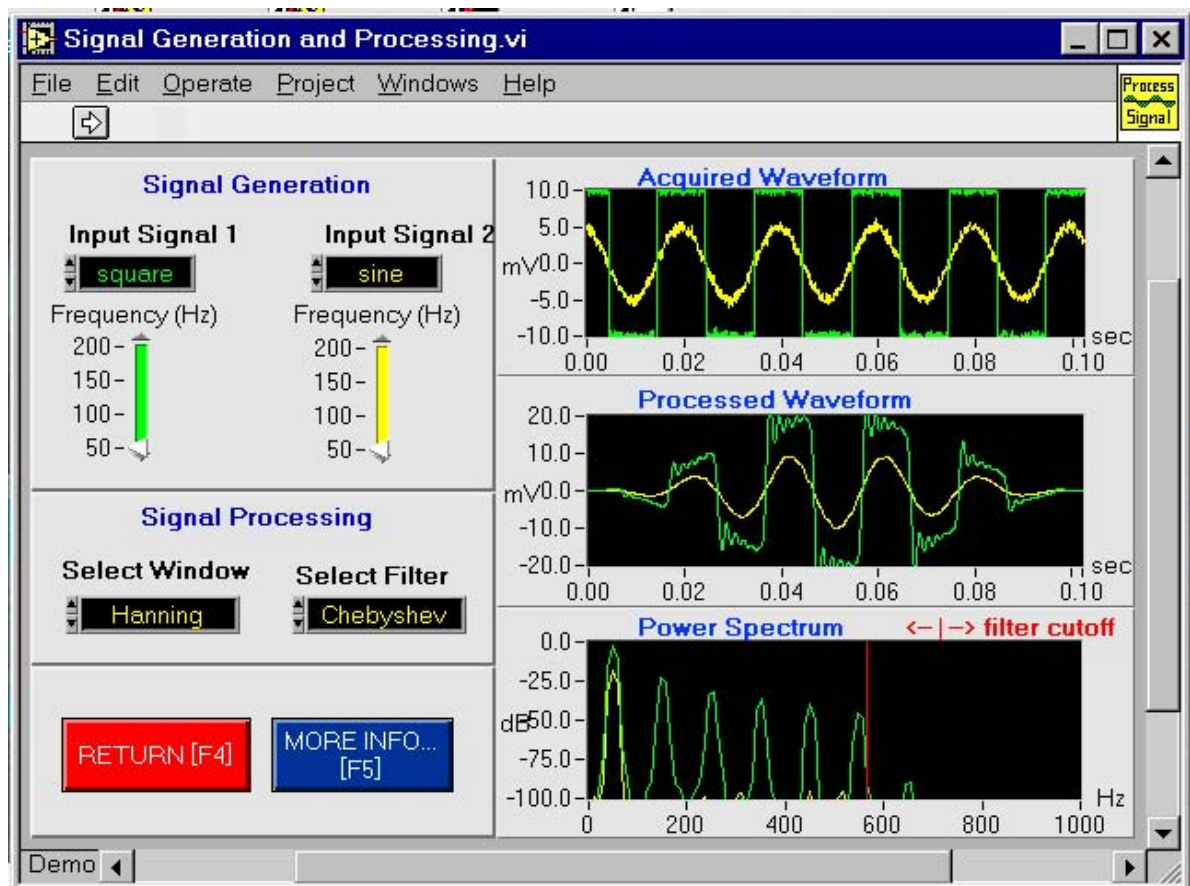
(*i32*) *code*: kod greske pridruzen gresci. Vrijednost 0 znaci nema greske, negativna vrijednost je neka greska a pozitivna vrijednost je upozorenje.

(*string*) *source*: indikacija o tome gdje se greska pojavila, obicno je to ime VI od Izlazne vrijednosti:

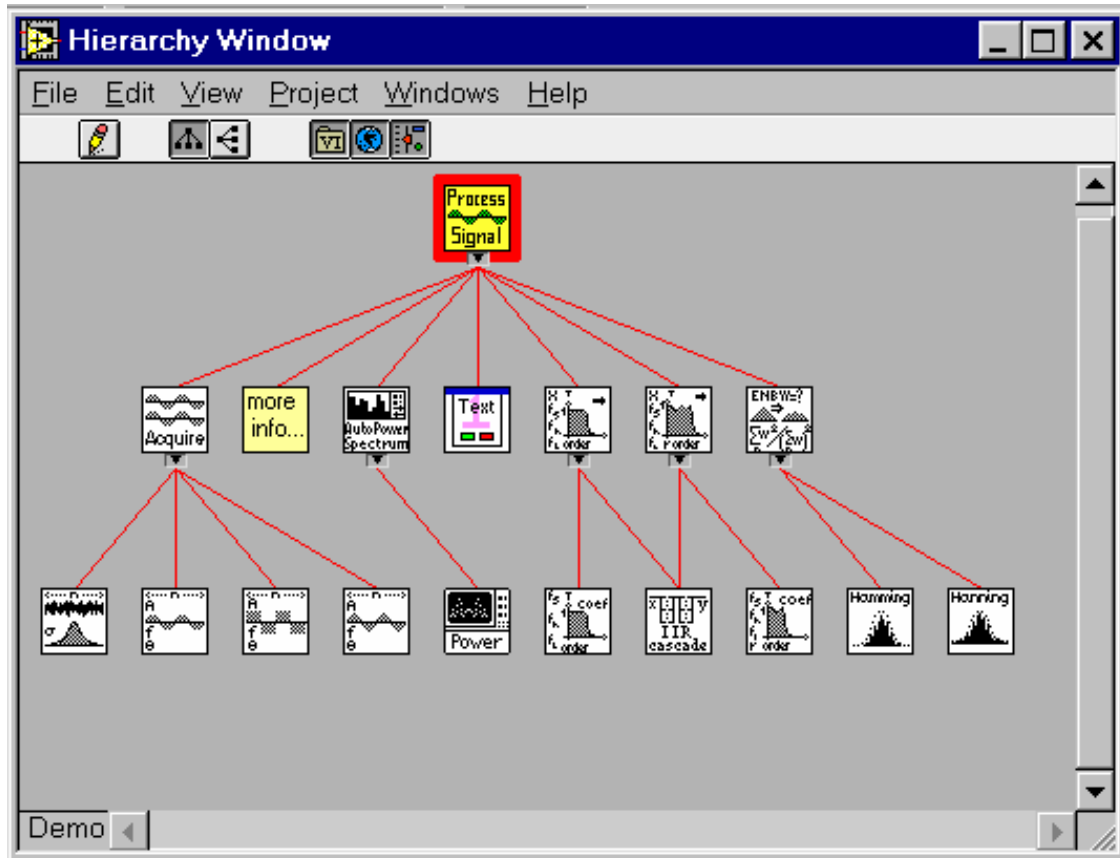
MODUL GENERIRANJA I PROCESIRANJA SIGNALA

IME VI : Signal Generation and Processing.vi

Prednji panel ove VI prokazan je na narednoj slici:



Hijerarhijski dijagram za VI i subVI-eve ukljucene u ovu VI je ilustriran na slijedećoj slici:



Dakle, osnovna VI - Signal Generation and processing poziva slijedeće subVI:

- Acquire Signal.vi
- More Information.vi
- Auto power Spectrum.vi
- Read Text info (one file).vi
- Butterworth Filter.vi
- Chebyshev Filter.vi
- Scaled Windows.vi

Ove subVI, pozivaju svoje subVI (drugi nivo) i to:

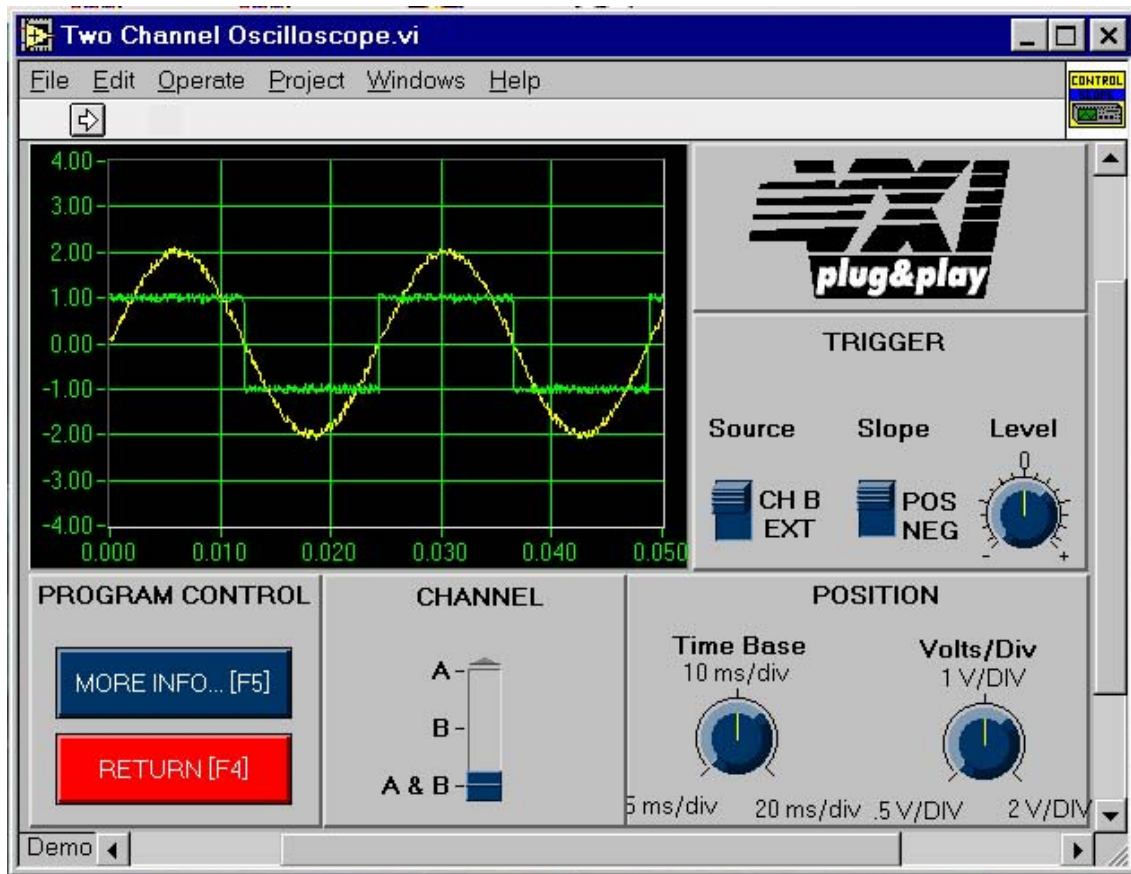
- Acquire Signal.vi poziva:
 - ◊ Gaussian White Noise.vi
 - ◊ Sine Wave.vi
 - ◊ Square Wave.vi
 - ◊ Triangle Wave.vi
 - ◊ Power Spectrum.vi
- Butterworth Filter.vi poziva :
 - ◊ Butterworth Coefficients.vi
 - ◊ IIR cascade Filter.vi
- Chebyshev Filter.vi poziva :

- ◇ IIR cascade Filter.vi
- ◇ Chebyshev Coefficients.vi
- Scaled Windows.vi poziva:
 - ◇ Hamming Window.vi
 - ◇ Hanning Window.vi

DVOKANALNI OSCILOSKOP

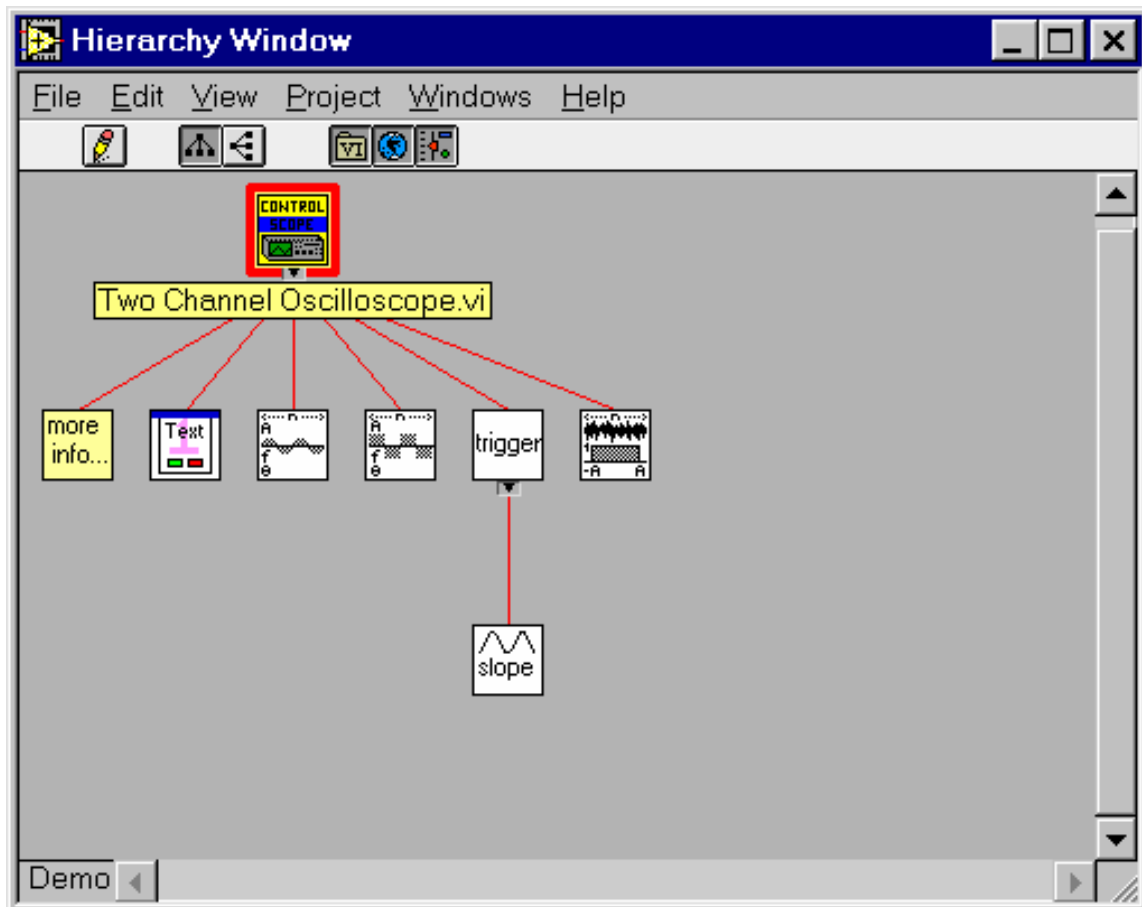
IME VI : Two Channel Oscilloscope.vi

Ova VI je virtualna verzija klasičnog dvokanalnog osciloskopa, čija je mimika standardnog osciloskopa toliko vjerna, da naredna slika koja pokazuje prednji panel VI može da djeluje zbujujuće:



Ovaj virtualni osciloskop demonstrira fleksibilnost LabVIEW-a za primjene u kontroli mjernih instrumenata. Koristeći jednostavne elemente kao: grafove, prekidače, dugmad, itd. možemo jednostavno kreirati vrlo ekonomične virtualne instrumente, koje možemo lako modifikovati prema specifičnim potrebama.

Hijerarhijski prozor za ovu Vi otkriva nam visenivovsku strukturu koja uključuje VI i dva nivoa subVI kako se to vidi na narednoj slici:



Na prvom nivou submodula nalaze se slijedeće subVI:

- More information.vi
- Read Text Info (one file).vi
- Sine Wave.vi
- Square Wave.vi
- Trigger.vi
- Uniform White noise.vi
- Slope.vi

Na drugom subnivou VI, Trigger.vi poziva Slope.vi modul.

POGLAVLJE 12

Start upravljanja instrumentom sa interaktivnom rutinom i LabVIEW Instrument Wizardom

Uvod

Jedna od najvećih frustracija kada počinjemo rad sa GPIB basom je verifikacija da mi doista možemo komunicirati sa GPIB spojenim instrumentima. Ponekad možemo provesti sate i sate listajući kroz debele priručnike i nivoe menija na instrumentu, pokušavajući da nađemo informaciju o GPIB adresi kao i da li je GPIB omogućen (enabled). Često i sam pokušaj da dobijemo odziv GPIB instrumenta na upit za identifikaciju (identification query), postaje prava avantura. Da bi smanjio ovu frustraciju kod povezivanja i rada sa GPIB baziranim instrumentima, NI pruža niz alata za produktivnost koji pomazu da se ovaj važan start-up postupak rada sa GPIB instrumentima ubrza tako da se mjerni inženjer može usredsrediti na važniji aspekt njegove mjerne aplikacije- gradnji test setupa. U nastavku, opisacemo različite alate i tehnike koje se mogu koristiti da se brzo i lako prodje početna faza startanja i povezivanja GPIB instrumenata.

Primjeri kako se koriste ove interaktivni pomazuci (utility) programi kao i LabVIEW Instrument Wizard su u verziji 5.0 (posljednja verzija u Septembru 1999 je verzija 5.1 , vidjeti na Web siteu NI: www.natinst.com)

KORISTENJE INTERAKTIVNE RUTINE ZA START

Interaktivna rutina, poznata kao Interface Bus Interactive Control (IBIC), je standardni softwareski pomazuci program (utility), koji je uključen sa NI proizvodima za GPIB kontroler. Koristeci tastaturu PC, ovaj pomazuci program u razvoju i debugiranju se može koristiti za interaktivnu komunikaciju (read, write, serial poll, etc) sa GPIB instrumentima. Sa ovim interaktivnim kontrolnim utility programom može se značajno ubrzati razvoj aplikacije uceci kako se može automatizirati proces mjerenja, otkriti GPIB problemi, i izbjeći problemi sa neispravnim instrumentima. Za Windows platforme, ova interaktivna kontrolna podrška se isporučuje od NI kompletna sa on-line help datotekom, koja opisuje NI-488 funkcije i NI-488.2 rutine, sintakse, kodove greski, i statusne varijable koje pružaju debugging informacije da bi rjesili problem kada se pojavi.

Tipicno, prvi izazov u radu sa GPIB instrumentom je odredjivanje njegove GPIB adrese. Brz način da se ovo ostvari je da se koristi interaktivna kontrolna podrška. Pri tome dvije komande su posebno korisne:

findlstn i ibln

BRZO ODEREDJIVANJE GPIB ADRESA

Komanda **nadji sve slusaoc** (**find all listeners**), **findlstn**, je 488.2 komanda koja pretražuje GPIB mrežu i prvo odredi broj instrumenata konfigurisanih kao slusaoci (listeners) koji su aktivni na basu; zatim nalazi GPIB adresu sa kojim je svaki slusaoc konfiguriran. Komanda **findlstn** može otkriti i 488.1 i 488.2 uredjaje. Sintaksa za ovu komandu je slijedeca:

findlstn *address_list limit_number*

gdje je:

address_list = moguće adrese za priključene instrumente, u granicama 1-30, odjeljene zarezom (,).

limit_number = maksimalni broj instrumenata koje treba tražiti

Da bi instrumenti bili prepoznati oni moraju biti uključeni na napajanje, i spojeni na GPIB kontroler preko spojnog kabla. Ako postoji dva ili više instrumenata na basu, možemo isključiti privremeno sve ostale izuzev jednog čiju adresu određujemo. Izolujući na ovaj način jedan po jedan instrument na basu i svaki put izvrsavajući **findlstn** komandu, možemo brzo odrediti adresu svakog instrumenta. Možemo također pristupiti i tako da izvršimo **findlstn** komandu jedanput, a zatim da pitamo svaki instrument za njegov ID (identifikacioni) string da bi uparili svaki instrument sa njegovom adresom. Upit za ID (ID query) će biti diskutiran u nastavku ovog teksta.

Slijedi primjer kako se koristi **findlstn** komanda. Predpostavimo da imamo jedan instrument na basu sa adresom 5. Možemo koristiti slijedecu sekvencu komandi sa interaktivnim kontrolnim programom podrške. Masna slova predstavljaju prompt programa, a statusni kod je izostavljen jer ćemo predpostaviti da se sve komande uspješno izvrsavaju. Komande se završavaju sa pritiskom na **Enter** taster. Komentari u zagradama objasnjavaju svaki korak:

:

(Ovo je prompt kada prvi put udjete u interaktivnu kontrolnu podršku)

: set 488.2

(Ova komanda aktivira 488.2 mod rada)

488.2 (0): sendifc

(Ova komanda šalje komandu za čiscenje interfejsa, i stavlja modul 0 kao kontroler koji je aktivan. Čiscenje interfejsa (interface clear – ifc) se šalje da bi se inicijalizirao sistem. U svakom trenutku vremena, postoji samo jedan aktivan kontroler u dužnosti (CIC – controller in charge) na basu. CIC koordinira komunikaciju na basu tako da uređaji govore (talk) ili slusaju (listen), kako to odgovara cjelini test setupa.

488.2 (0): findlstn

enter address list: 1, 2, 3, 4, 5, 6, 7, 8

(moguće adrese za instrumente)

enter 'limit' number: 1

(maksimalni broj instrumenata koji se treba tražiti na basu)

Odziv koji ćemo dobiti je slijedeci. **Count** daje broj instrumenata koji slusaju na basu a **listeners** daje adresu instrumenta:

count: 1

(broj slusaoca koji je nadjen je 1)

listeners: 5
(sa adresom 5)

USPOSTAVLJANJE KOMUNIKACIJE SA INSTRUMENTIMA

Nakon odredjivanja GPIB adresa instrumenata, lako je uspostaviti komunikaciju da bi verificirali da mozemo slati i primiti podatke ka i od instrumenta. Posto je vecina instrumenata u saglasnosti sa 488.2 standardom, mozemo poslati upit (query) ka instrumentu da utvrdimo njegovu identifikaciju, saljuci mu "*IDN?". Instrument ce najcesce odgovoriti sa imenom Proizvodjaca, nazivom modela, i razlicitim alfanumerickim karakteristikama koje Proizvodjac koristi da bi pratio revizije firmwarea (promiziranog/epromiziranog softwarea u instrumentu). Da bi komunicirali sa nasim zamisljenim instrumentom na adresi 5, treba da unesemo slijedecu sekvencu komandi. Primjetimo da **ibdev** je funkcija koja se koristi da se otvori sesija sa instrumentom.

:

(Ovaj prompt se pojavi kada prvi put udjemo u interaktivnu kontrolnu rutinu)

: ibfind gpib0
(Postavi module 0 online i otvori sesiju)

gpib0: ibsic
(Ova komanda salje komandu za ciscenje interfejsa, i postavlja modul 0 u ulogu zaduzenog kontrolora - CIC)

gpib0: ibdev
enter board index: 0 (unesi indeks modula)
enter primary address: 5 (unesi primarnu adresu)
enter secondary address: 0 (unesi sekundarnu adresu)
timeout: 13 (vrjeme cekanja na odbrojavanje)
enter **EOI on last byte** flag: 1
enter end-of-string mode/byte: 0
(ova komanda otvara sesiju sa uredjajem na adresi 5)

ud0: ibwrt *IDN?
(ud0 je novi prompt za nivo uredjaja da bi govorili direktno sa nasim uredjajem. Poslacemo ka 488.2 instrumentu upit *IDN? da bi ga upitali za identifikaciju. Ukoliko je rijec o 488.1 uredjaju moracemo konsultovati njegovu dokumentaciju da bi vidjeli koja je to komanda koja ce nam u odgovoru obezbjediti podatke o instrumentu. Tipicno, 488.1 uredjaji se ne odzivaju na *IDN?.)

ud0: ibrd 100
(Iscitaj 100 bajta iz instrumenta – u ovom slucaju ovih 100 bajta ce sadrzavati identifikacioni string instrumenta)

JEDNOSTAVAN NACIN ZA TRAZENJE GRESAKA NA INSTRUMENTU, KABLOVIMA I NAPAJANJU.

Cesto, zbog održavanja ili promjene konfiguracije, instrumenti se isključuju sa napajanja i spojni kablovi se odvajaju sa njih. Pri ponovnom uspostavljanju setupa i povratka instrumenata pod napon, ceste su greske da se ne povezu svi kablovi i konektori, ili da se svi instrumenti ne stave pod napon. Alternativno, neki od instrumenata mogu i da otkazu u toku rada sa mjenim setapom (garniturom). Komanda **ibln** je jedna pogodna funkcija za verifikaciju da su svi instrumenti iz setupa jos **zivi** na GPIB basu. Jednostavno ovom komandom verifikiramo da odredjeni instrument slusa na basu na svojoj odredjenoj adresi. Ako to nije slucaj onda nam je to znak da moramo da provjerimo kablove, napajanje, i da utvrdimo da li je uredjaj ispravan.

Sintaksa za **ibln** komandu je slijedeca:

ibln *primary_address secondary_address*

primary_address = primarna GPIB adresa

secondary_address = sekundarna GPIB adresa

Ova funkcija moze otkriti i 488.1 i 488.2 instrumente. Takodjer, posto vecina instrumenata ne koristi sekundarnu adresu, vrijednost 0 se koristi da specificira sekundarnu adresu.

Kao primjer, predpostavimo da da imamo instrument na adresi 10. Koristili bi slijedecu sekvencu komandi radi verifikacije da je instrument jos ziv i prisutan na basu:

:

(Ovo je prompt kod prvog ulaska u interaktivnu kontrolnu rutinu)

:ibfind gpib0

(postavi modul 0 online i otvori sesiju)

gpib0: ibsic

(ova komanda salje naredbu za ciscenje interfejsa-interface clear, i postavlja modul 0 u ulogu kontrolora na duznosti - controller in charge ; CIC)

gpib0: ibln 10 0

(provjeri da li instrument slusa na adresi 10)

Ako je uredjaj ziv **ziv** na basu, dobicemo slijedeci odziv:

Listen: TRUE

Ako postoji neki problem, dobicemo:

Listen: FALSE

sto nas obavjestava da moramo provjeriti za lose spojene ili odpojene kablove, prekide u napajanju, ili otkaz instrumenta.

Osnovne funkcije i gore opisani koncepti mogu izgledati odvec jednostavni, ipak oni mogu biti od ogromne pomoci kada treba da pronadjemo i otklonimo greske u

mjernom setapu sa GPIB instrumentima. Ovaj alat nam omogućava da se maksimalno usredsredimo na nasu mjernu aplikaciju i izbjegnemo trošenje isuvise vremena u otkrivanju i otklanjanju komunikacionih problema medju nasim povezanim instrumentima.

LabVIEW INSTRUMENT WIZARD AUTOMATIZIRA INTERAKTIVNU KONTROLU INSTRUMENTATA

Nakon utvrđivanja da je komunikacija medju GPIB instrumentima na basu korektna i radi isparavno, potrebno nam je da brzo predjemo iz ovog interaktivnog moda rada u programing mod tako da mozemo direktno poceti pisati nase test rutine bez mukotrpnog prelaznog procesa iz jednog moda rada u drugi.

Instrument Wizard u okviru LabVIEW 5.0 na taj nacin ntegrira i i automatizira, unutar LabVIEW-a neke od startup komandi i funkcija o kojima smo diskutirali u prethodnom tekstu. Sa ovim instrument Wizardom stedimo na vremenu u razvoju aplikacija kontrole instrumenata, posto LabVIEW automatski identificira prikljucene instrumente, instalira odgovarajuce biblioteke instrument drajvera potrebnih za programiranje, i zatim pokrece aplikacioni primjer koji ce verificirati komunikaciju. Aplikacioni primjer se generira koristeći instalirani instrumentalni drajver, tako da nam to moze poslužiti kao uzor (template) od kojeg mozemo poceti gradnju nase specificne test aplikacije.

Slijedi primjer kako mozemo koristiti Instrument wizard da startamo nas specificni GPIB sistem. U ovom primjeru dodacemo jos dva nepoznata uredjaja u sistem.

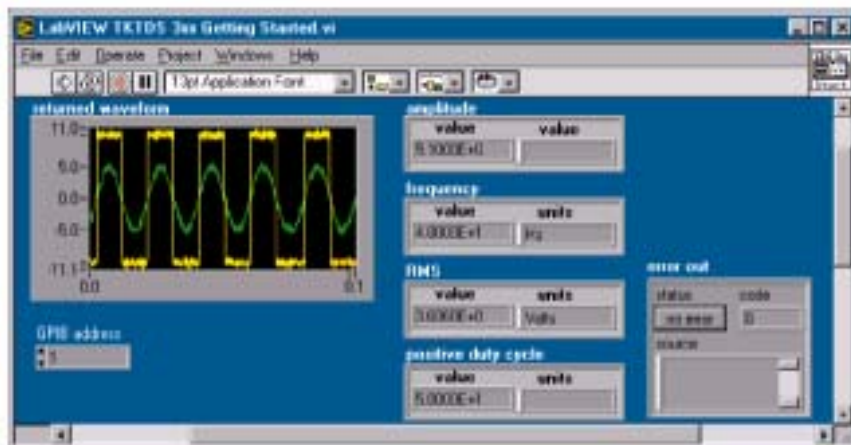
SKANIRANJE SISTEMA ZA PRISUSTVO INSTRUMENTATA

Nakon lansiranja instrument wizarada, LabVIEW ce skanirati sistem da utvrdi prisustvo GPIB instrumenata kao i VXI, RS-232 te instrumenata prikljucenih direktno na kompjuterski bas. Na narednoj slici br. 1 LabVIEW je otkrio 2 GPIB uredjaja na adresama 9 i 22. LabVIEW identificira nove GPIB objekte u sistemu kao uredjaj, izuzev ako mi koristimo **Identify Device** taster da bi poslali upit uredjaju za dodatnu specificnu informaciju. Po defaultu, **Identify Device** salje **ISDN?** upit ka instrumentu, ali imamo opciju da promjenimo string i da posaljemo instrumentu neku drugu komandu. LabVIEW nakon toga iscitava odziv instrumenta i azurira labelu drveta instrumenta da unese ovu primljenu informaciju. U ovom slucaju, identifikovali smo dva nova instrumenta: Tektronix-ov osciloskop (TekTDS3xx) na adresi 9 i Hewlett-Packard-ov (HP) DMM (HP34401A) na adresi 22.

Instrument drajveri su mocni softwreski alati koji kombiniraju granularni, nisko-nivovski string komandi potrebnih za konfigurisanje i mjerenje u visoko-nivovske funkcije kao: **initialize, reset, i read** valne oblike signala. Biblioteke instrument drajvera cine kontrolu instrumenata laksom i jednostavnijom i pomazu nam da se koncentriramo na sama mjerenja , razvoj test setupa kao i donosenje odluka vezanih za ciljeve mjerenja, umjesto da se iscrpljujemo oko detalja kako razviti instrument drajver software. **NI** obezbjedjuje kompletnu selekciju najpopularnijih instrument drajvera koje korisnici mogu zahtijevati. Za vise detalja potraziti na Internetu na adresi www.natinst.com/idnet

OTVARANJE PRIMJERA RADI VERIFIKACIJE KOMUNIKACIJE

Finalni korak da bi kompletirali sistem je da verificiramo komunikaciju sa instrumentom. Kada kliknemo na taster "Open Example" (vidjeti sliku br. 2), LabVIEW lansira pripremljeni primjer koji ce komunicirati sa nasim instrumentom, koji ce verificirati da mi mozemo i u stvarnosti uzimati mjerenja sa instrumenta. Na slici br. 3, vidljivo je da smo lansirali pripremljenu aplikaciju koja cita podatke valnog oblika sa Tektronix-ovog osciloskopa i verificira dvosmjernu komunikaciju sa instrumentom.



Slika br. 3 Verifikacija komunikacije sa instrumentom koristenjem aplikacionog primjera

Kao sto se moze vidjeti, LabVIEW instrument Wizard je dizajniran tako da vodi korisnika kroz proces dodavanja instrumenta u mjerni sistem. Automatizirajuci detekciju, upit, i instaliranje instrument drajvera, mi mozemo brzo uspostaviti mjerni setup i zatim se koncentrirati na izgradnju same mjerne aplikacije sto nam je primarnni cilj.

POGLAVLJE 13

PRIMJERI KORIŠTENJA LABVIEW-A I DAQ MODULA U SINTEZI INSTRUMENTATA

Uvod

U ovom poglavlju ćemo opisati DAQScope 5102 kao tipičnu konfiguraciju hardwarea **NI** (DAQ modula) i LABView-a instaliranog na PC-ju za sintezu mjernog instrumenta. DAQScope PCI-5102, DAQScope AT-5102 i DAQScope DAQCard-5102 uređaji su analogni ulazni moduli za PCI (peripheral component interface), AT (ili ISA) i PCMCIA (danas poznatiji kao PC card) baze PC desktop računara odnosno notebooka.

Ovi analogni ulazni moduli, zajedno sa LABView softwareom realizuju virtuelni instrument u ulozi digitajzera i osciloskopa.

U nastavku teksta mi ćemo koristiti DAQScope 5102 kao generički naziv koji će označavati jedan ili više tipova tj. : PCI-5102, AT-5102 i DAQCard-5102 uređaja. Zajedničke karakteristike 5102 uređaja su:

- dva analogni ulazna kanala rezolucije 8 bita
- Brzina uzorkovanja od 1 kS/s do 20 MS/s (uzoraka/sec) za sampling u realnom vremenu; 1 GS/s za slučajno uzorkovanje sa preklapanjem (RIS – random interleaved sampling)
- 15 MHz propusni opseg analognog ulaza
- analogni triggerski kanal sa softwareski selektabilnim nivoom, nagibom, i histerezisom
- dva digitalna trigera
- softwareski selektabilno AC/DC kuplovanje
- Trigerovanje sistemske integracije u realnom vremenu (RTSI real-time system integration) kod modela PCI-5102 i AT-5102
- memorija na DAQ ulaznom modulu za 663.000 uzoraka.

Sva tri modela 5102 slijede industrijski standard za PnP (plug and play) specifikacije na svim platformama PC računara sa jednim od tri pomenuta basa, i nude transparentnu integraciju sa komplementarnim sistemima. Ako naša aplikacija PC baziranog osciloskopa zahtjeva više od dva kanala za akviziciju analognih podataka, možemo sinhronizirati više ovakvih uređaja na svim platformama (PCI, ISA, PC card), koristeći RTSI bas da trigeruje na uređajima koji koriste RTSI bas, ili digitalne trigere na I/O konektoru.

Da bi se poboljšala vremenska rezolucija za repetitivne signale, može se koristiti RIS na DAQScope 5102. Ovaj metod uzorkovanja dozvoljava nam da pogledamo pretrigerske podatke i dostignemo efektivnu brzinu uzorkovanja i do 1 GS/s, tj. 50 puta više od real-time brzine uzorkovanja uređaja.

Detaljnije specifikacije DAQScope 5102 uređaja su slijedeće:

Ulazne karakteristike

Broj ulaznih kanala	2 jednostruka, simultano uzorkovana 1Mohm +/-1% u paraleli sa 30pF+/-15pF (CH0, CH1, CH2)
ADC rezolucija	8 bita, 1 u 256
Maximalna brzina uzorkovanja	
Interna	20 MS/s po svakom kanalu u real- time modu rada
Vanjskii clock uzorkovanja	20 MS/s
Minimalno low ili high vrijeme	24 ns
RIS mode	1 GS/s
Minimalna brzina uzorkovanja	1 KS/s (interno/eksterno)
Maksimalni ulazni opseg	+/- 500 V sa 100X sondom (pojaćanje 1). +/- 50 V sa 10X sondom (pojaćanje 1). +/- 5 V sa 1X sondom (pojaćanje 1). (CH0,CH1, TRIG)
Opsezi ulaznih signala (CH0, CH1) (bez prigusenja sonidi)	+/- 5 V kod pojaćanja 1 +/- 1 V kod pojaćanja 5 +/- 0.25 V kod pojaćanja 20 +/- 50 mV kod pojaćanja 100
Ulazno kuplovanje selektabilno	AC ili DC, software
Prenaponska zastita	+/- 42 V kod ukljućenja ili iskljućenja (bez vanjskog prigusenja) CH0, CH1, TRIG
Dubina FIFO memorije na modulu	663.000 uzoraka
Prenos podataka	programirani I/O podrzan na svim verzijama; direktno u memoriju burst transfer sa PCI bus kontrolom od strane PCI-5102 modula.

Prenosne karakteristike

Relativna tacnost	+/- 1 LSB tipicno, +/-1.8 LSB max
Diferencijalna nelinearnost	+/-0.3 LSB tipicno, +/-0.5LSB max
Bez gubitka informacije	8 bita garantirano
Greska ofseta (nakon kalibracije)	+/-1.5 LSB max
Greska pojaćanja (nakon kalibracije)	+/-1% max.

Dinamicke karakteristike**Opseg**

Mali nivo signala (-3 dB)	15 MHz tipicno
Veliki nivo signala(2% THD)	10 MHz tipicno
Nisko frekventni cut-off za AC kuplovanje	11 Hz (1.1 Hz sa sondom x10)

Smirenje za step u citavom opsegu, do +/-1% punog opsega 50 ns tipicno

Sum sistema Crosstalk(medjusum)	0.5 LSB rms tipicno -60 dB
-------------------------------------	-------------------------------

S/H karakteristike

Medjukanalno iskrivljenje(interchannel skew)	1 ns
Aperturno podrhtavanje (aperture jitter)	1 ns rms

Stabilnost

Preporuceno vrijeme zagrijavanja	15 min.
Ofset temperaturni koeficijent	(1 mV/°C)/pojacane + 30 μV/°C
Temperaturni koeficient pojacanja	50 ppm/°C

Trigeri**Analogni triger**

Izvor	CH0, CH1, TRIG
Nivo	+/-Pune skale za CH0 i CH1; +/-5V za TRIG softwareski selektabilno
Nagib selektabilno	pozitivni ili negativni,softwareski
Rezolucija	8 bita, 1 od 256
Histerezis	softwareski selektabilno, do punog opsega
Opseg MHz	15

Digitalni trigeri (PFI1 i PFI2)

Kompatibilnost	TTL/CMOS
Odziv	rastuca ili opadajuca ivica; softwareski selektabilno
Sirina impulsa	10 ns min

DC karakteristike unutar radnog opsega

Symbol	Parameter	Conditions	Min	Max
V_{IH}	Input HIGH voltage	—	2.0 V	$V_{CC} + 0.5$ V
V_{IL}	Input LOW voltage	—	-0.5	0.8 V
V_{OH}	Output HIGH voltage	$I_{OH} = -4$ mA $I_{OH} = -16$ mA $I_{OH} = -10$ μ A	3.7 V 2.4 V $V_{CC} - 0.1$ V	—
V_{OL}	Output LOW voltage	$I_{OL} = 16$ mA $I_{OL} = 10$ μ A	—	0.45 V 0.1 V
C_I	Input capacitance (nominal)	—	—	10 pF
I_{OS}	Output short circuit current ¹	$V_O = GND$ $V_O = V_{CC}$	-15 mA 40 mA	-120 mA 210 mA
¹ Only one output at a time; duration should not exceed 30 s.				

RTSI (PCI-1502, AT-5102)

Broj trigerskih linija

7 I/O

Clock linija

1

Fizicke karakteristike

PCMCIA (PC Card)

Tip II

Dimenzije

PCI-5102

10.67 sa 17.45 cm

AT-5102 (za ISA bus)

10.67 sa 17.45 cm

SOFTWARESKE PROGRAMSKE OPCIJE

Postoji nekoliko opcija na raspolaganju kod izbora programske podrške za ovaj konkretni hardwareski modul, ali i za bilo koje **NI** DAQ module. Mozemo koristiti LabVIEW, LabWindows/CVI, Components Works, Measure, ili Virtualbench.

VirtualBench je ansambl Vi-eva koji nam omogućava da koristimo DAQ module kao sto cinimo i sa autonomnim instrumentima, i pritom dodatno imamo na raspolaganju dodatno procesiranje, mogucnosti displeya na monitoru PC-ija kao i njegove mogucnosti pohranjivanja prikupljenih podataka. VirtualBench instrumenti pune i pohranjuju valne oblike na disk u istom obliku koji se koristi i kod popularnih spreadsheet programa (napr. Excel ili Paradox) ili procesora teksta (napr. Word). Detaljniji opis ovoga paketa bio je dat u jednom od ranijih poglavlja.

ComponentWorks sadrzi alate za prikupljanje podataka kao i upravljanje instrumentima, koji su izgradjeni na NI-DAQ drajverskom softwareu. ComponentWorks pruza visoko-nivovski programski interfejs za izgradnju

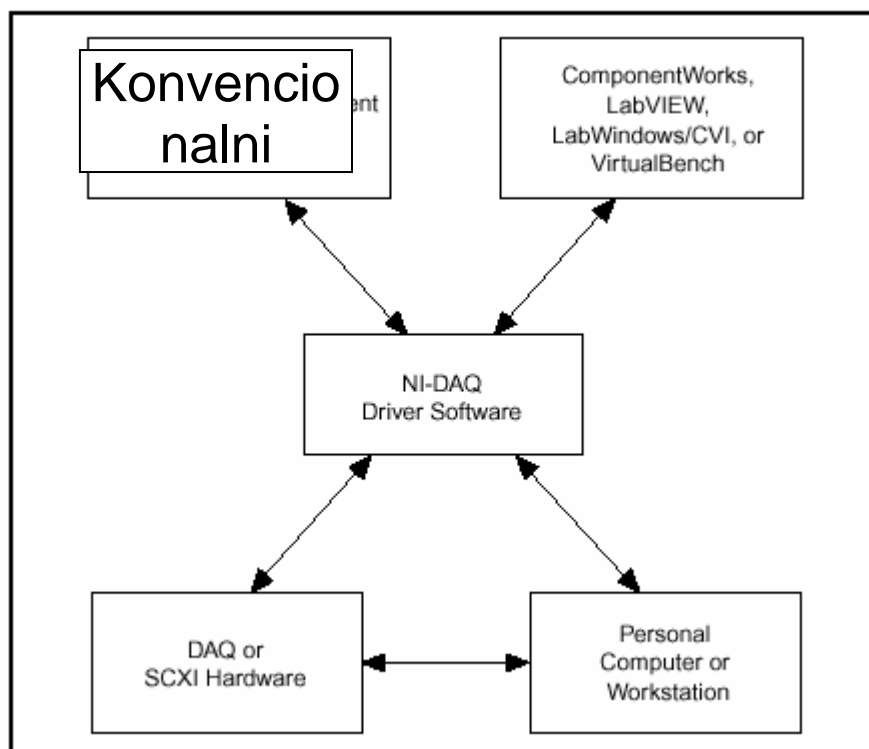
virtualnih instrumenata sa Visual Basicom, Visual C++, Borland Delphi, i Microsoft Internet Explorer (IE). Sa ComponentWorks, mogu se koristiti svi konfiguracioni alati, utilitiji (korisni programi) za management resursa kao i interaktivne kontrolne korisne rutine koje su uključene sa NI-DAQ.

Measure je softwareski dodatak (add-on) za Microsoft Excel za prikupljanje podataka i kontrolu instrumenata. Sa Measure, mi mozemo prikupljati podatke direktno sa uplagljivih DAQ modula, GPIB instrumenata, ili serijskih (RS-232) uređaja. Measure ima lagane dijalog prozore za konfigurisanje mjerenja sa instrumentima. Prikupljeni podatci se smjestaju direktno u Excel polja na radnim listama (worksheet), sa kojih je dalje moguće izvršiti analizu i generirati izvjestaje, koristeći mogućnosti Excela.

NI-DAQ DRAJVERSKI SOFTWARE

NI-DAQ drajverski software je uključen sa svakim od isporučenih NI DAQ hardware modula. NI-DAQ ima ekstenzivnu biblioteku funkcija koje mozemo pozivati iz svakog standardnog programskog okruženja (programming environment).

Bez obzira da li koristimo konvencionalne programske jezike, LabVIEW, LabWINDOWS/CVI, VirtualBench ili ComponentWorks, naša aplikacija uvijek koristi NI-DAQ drajverski software, kao što je to prikazano na slijedećoj slici 12-1:



Slika 12-1 Odnos između programskog okruženja, NI-DAQ i hardwarea kojeg koristimo

Uobicajeni pomocni pribor koji ide uz modul DAQScope 5102, najcesce ukljucuje:

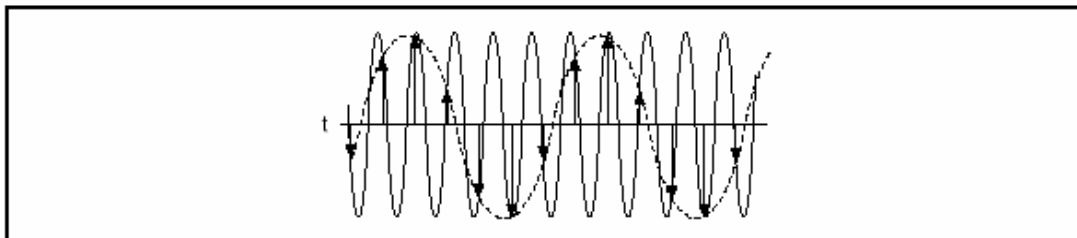
- TPI9258 100x sondu sa dodatcima za visoko naponske primjene
- MB200 SMB muski/muski kabel za master/slave timing i trigerovanje
- SMB300 SMB muski/aligatorski klip kabel za vanjsko trigerovanje
- RTSI bus kabel

DAQScope 5102 je upotpunosti softwareski konfigurabilna uplagljiva kartica, sa punim P&P (plug and play) mogucnostima. Informacija o konfiguraciji je pohranjena u permanentnoj/programabilnoj memoriji (kao napr EEPROM). PnP servis testira uredjaj nakon uplagovanja u jedan od tri busa za koji se proizvodi, iscitava pohranjenu informaciju o modulu, i arbitrira alokaciju resursa za : baznu adresu (base address), nivo interapta (interrupt level) i kanal direktnog pristupa memoriji(DMA- direct memory access). Nakon doznacavanja ovih resursa, operativni sistem ce omogućiti DAQ uredjaju da se aktivira i radi.

Opis rada digitajzera

Da bi razumjeli rad digitajzera, pogledacemo ukratko kako Nyquist-ov teorem utice na analogni propusni opseg i brzinu uzorkovanja.

Nyquist-ov teorem kaze da signal mora biti uzorkovan najmanje dva puta vecom brzinom od propusnog opsega da bi korektno rekonstruirali valni oblik; inace, visoko frekventni sadrzaj ce se aliasirati(predstaviti drugacije) sa frekvencijom unutar spektra koji je od interesa(propusnog opsega). **Alias** je lazna komponenta na nizoj frekvenciji koja ce se pojaviti u uzorkovanim podacima prikupljenim na nizoj brzini uzorkovanja od potrebne. Naredna slika 12.2 pokazuje 5 MHz sinusni valni oblik uzorkovan sa 6MS/s ADC konvertorom. Tackasta linija oznacava aliasirani signal registrovan od strane ADC kod te brzine uzorkovanja.



Slika 12.2 Sinusni signal demonstrira Nyquist frekvenciju.

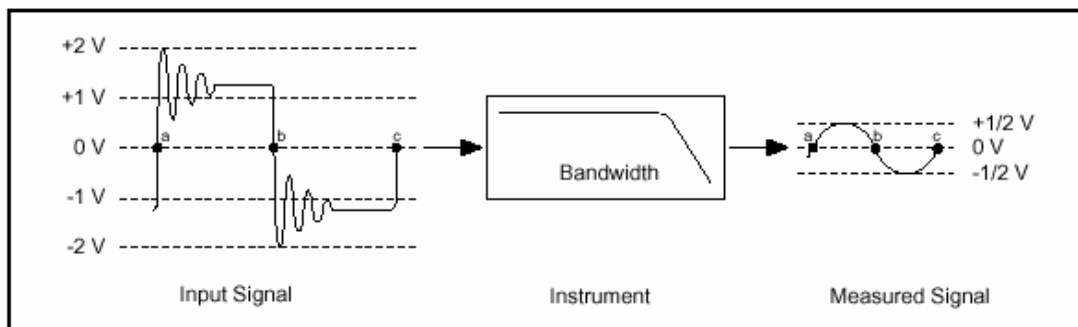
Kako se vidi sa ove slike, 5MHz frekvencija sinusnog signala se aliasira i pogresno pojavljuje kao da je to 1 MHz sinusni signal. Da bi se sprijecilo aliasiranje unutar propusnog opsega (passband), niskopropusni filter ogranicava frekventni sadrzaj ulaznog signala za sve frekvencije iznad Nyquistove.

Analogni propusni opseg

Analogni propusni opseg opisuje frekventni opseg (u Hz) unutar kojeg signal moze biti korektno uzorkovan (digitiziran). Ovo ograncenje je odredjeno sa

inherentnim frekventnim odzivom ulaznog puta signala (path)- od vrha sonde do ulaza u ADC pretvarac, koji uzrokuje gubitak amplitude i faze ulazne informacije. Analogni propusni opseg je frekvencija kod koje je izmjerena amplituda 3 dB manja od stvarne amplitude signala dovedenog na ulaz. Ovaj gubitak amplitude se pojavljuje i kod vrlo niskih frekvencija ukoliko je signal AC kuplovan kao i kod vrlo visokih frekvencija bez obzira na vrstu kuplovanja.

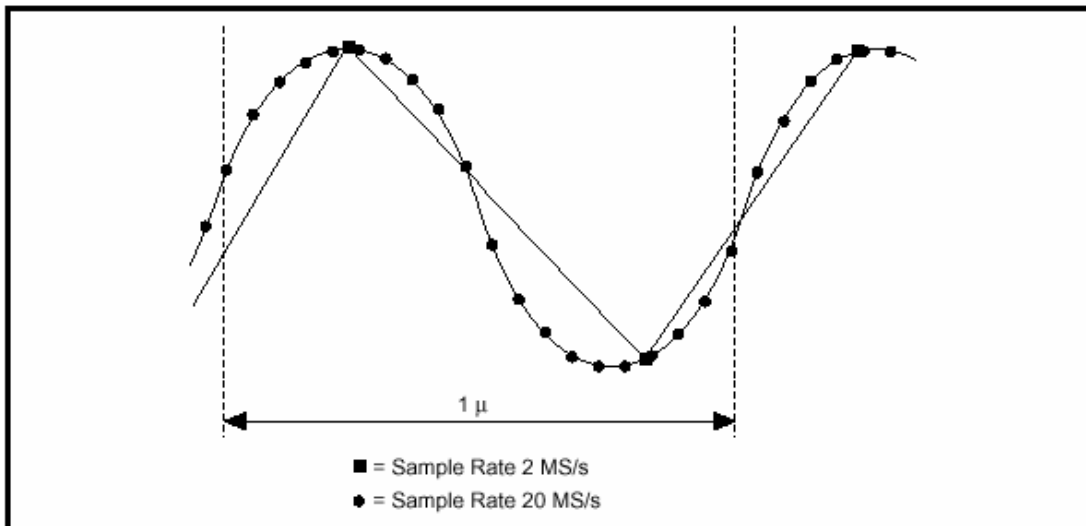
Kada je signal DC kuplovan, propusni opseg pojačala će se produžiti sve do DC napona. Naredna slika 12-3 ilustrira efekat analognog propusnog opsega na visoko frekventni signal. Kao rezultat, imamo gubitak visoko frekventnih komponenta i amplitude originalnog signala u toku prolaska signala kroz instrument:



Slika 12.3 Analogni propusni opseg

Brzina uzorkovanja

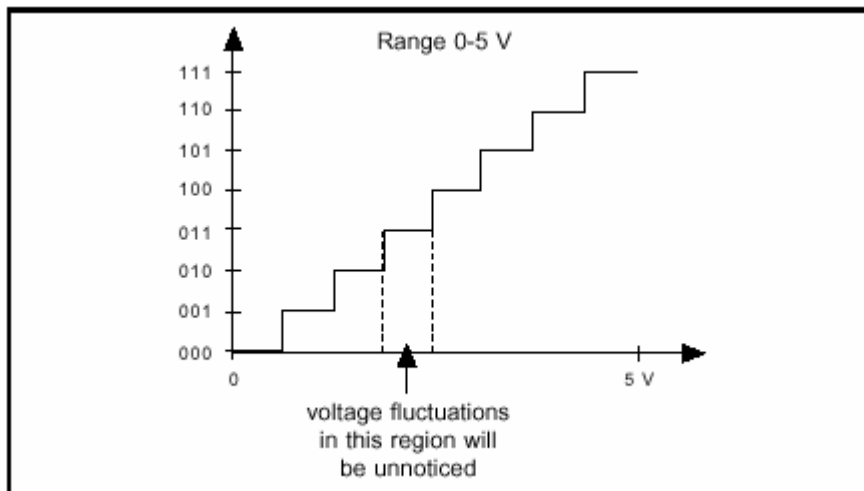
Brzina uzorkovanja je ona brzina pri kojoj je signal uzorkovan i digitaliziran (pretvoren u digitalni oblik) od strane ADC pretvaraca. U skladu sa Nyquist-ovim teoremom, visa brzina uzorkovanja proizvodi tacno mjerenje signala na visoj frekvenciji ako je analogni propusni opseg dovoljno sirok da dozvoli signalu da prodje kroz instrument do samplera i ADC pretvaraca bez prigusenja. Veca brzina uzorkovanja takodjer hvata (capture) vise detalja o valnom obliku na ulazu instrumenta. Slika 12-4 prikazuje 1 MHz sinusni signal uzorkovan sa 2MS/s ADC pretvaracem kao i sa 20 MS/s ADC-om. Brzi ADC digitalizira 20 tacaka po ciklusu ulaznog signala, u poredjenju sa 2 tacke po ciklusu sa sporijim ADC-om. U ovom primjeru vidimo kako veca brzina uzorkovanja tacnije hvata valni oblik kao i frekvenciju ulaznog signala.



Slika 12-4 Uzorkovanje sinusnog signala frekvencije 1 MHz

Vertikalna osjetljivost

Vertikalna osjetljivost opisuje najmanju promjenu ulaznog signala koju digitajzer može uhvatiti. Ovo ograničenje je uzorkovano time što jedan određeni digitalni napon (tj. iznos napona u digitalnom obliku), predstavlja opseg analognih vrijednosti. Zbog toga je moguće da vrlo male promjene signala na ulazu ne budu primjećene na ADC. Ovaj parametar zavisi od opsega ulaznog signala (range), pojačanja ulaznog pojačala i rezolucije ADC pretvaraca. Specificira se u jedinicama Volt/LSB. Naredna slika 12-5 pokazuje prenosnu funkciju od 3-bitnog ADC sa vertikalnim opsegom od 5 V i sa vertikalnom osjetljivošću od 5/8 V/LSB.



Slika 12-5 Prenosna funkcija za 3-bitni ADC

Rezolucija ADC-a

ADC rezolucija ograničava tačnost mjerenja. Ukoliko je rezolucija veća (tj. broj bita ADC-a), utoliko je tačnije mjerenje. 8 bitni ADC dijeli vertikalni opseg ulaznog pojačala u 256 diskretnih nivoa. Sa vertikalnim opsegom od 10V, 8 bitni ADC ne

može razlučiti razlike signala ispod 39 mV. 12 bitni ADC sa 4096 diskretnih nivoa može razlučiti naponske razlike ulaznog signala i do 2.4 mV.

Duzina zapisa

Duzina zapisa se odnosi na velicinu memorije koja je odvojena za pohranjivanje digitaliziranih uzoraka za postprocesiranje ili displej signala na ekranu. Kod digitajzera, duzina zapisa ogranicava maksimalno trajanje single-shot (jedno okidne) akvizicije. Naprimjer, sa baferom od 1000 uzoraka i brzinom uzorkovanja od 20 MHz, trajanje akvizicije je 50 μ s (broj tacaka pomnozen sa vremenom akvizicije/tacki, tj. 1000 x 50 ns). Sa baferom od 100000 uzoraka i brzinom uzorkovanja od 20 MHz, duzna trajanja akvizicije (prikupljanja) je 5 ms (100000 x 50 ns). Posto DAQScope 5102 ima velicinu bafera od 663.000 uzoraka, to znaci da je duzina akvizicije da se popuni citava memorija= 33.1 ms kod brzine uzorkovanja od 20 MS/s.

Duzina zapisa DAQScope PCI-5102 za jednookidne (single-shot) akvizicije je ogranicena kolicinom memorije koji je na raspolaganju u PC-ju, posto brzina prenosa podataka na PCI basu je veca nego brzina akvizicije samog DAQScope 5102 modula.

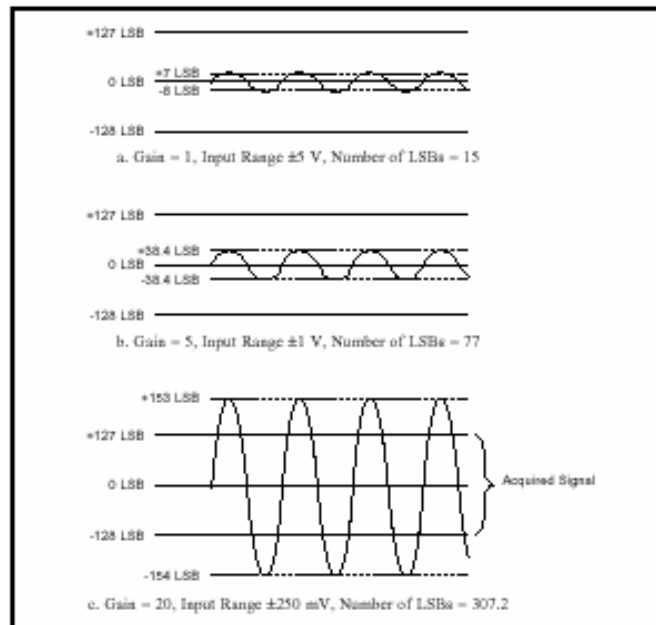
Trigerske opcije

Jedan od najvećih izazova da se realizuje mjerenje je uspješno okidanje (triggerovanje) trenutka pocetka akvizicije kod momenta koji nam je interesantan. Posto vecina digitajzera velike brzine ustvari zapisuje signal samo u malom segmentu ukupnog vremena, moguće je vrlo lako promasiti onaj dio toka signala koji zelimo zapisati, ako je trenutak triggerovanja pogresno postavljen. DAQScope 5102 je opremljen sa sofisticiranim triggerskim opcijama, kao sto su: pragovi triggerovanja, programabilne vrijednosti histerezisa, triggerski hold-off (zadrška), dvonivovsko triggerovanje na ulaznim kanalima kao i na posebnom triggerskom kanalu. DAQScope 5102 takodjer ima dva digitalna triggera koji pruzaju vise fleksibilnosti kod triggerovanja, dozvoljavajuci nam da spojimo TTL/CMOS digitalni signal da bi triggerovali pocetak akvizicije.

Realizacija tacnih mjerenja

Za tacna mjerenja potrebno je koristiti korektna podesenja kada se koristi DAQScope 5102 za prikupljanje podataka. Poznavanje karakteristika signala koji se akvizira pomaze da se izaberu korektna podesenja. Ove karakteristike ukljucuju izmedju ostalih:

* **peak-to-peak** (od vrha do vrha) vrijednost signala. Ovaj parametar, u jedinicama Volta, odrazava maksimalnu promjenu u signalu. Ako je V – vrijednost signala u bilo kojem trenutku vremena, tada $V_{pk-to-pk} = V_{max} - V_{min}$. Peak-to-peak vrijednost utice na vertikalnu osjetljivost ili pojacanje ulaznog pojacala. Ako nam ova vrijednost nije poznata, treba poceti sa najmanjim pojacanjem (kod maksimalnog ulaznog opsega) i povecavati pojacanje sve dok se valni oblik na ulazu ne digitizira sa maksimalnim dinamicnim opsegom bez odsjecanja dijela signala. Na narednoj slici 12-6 je pokazano da je pojacanje od 5 najbolje podesenje da se digitizira 300 mV, 1 MHz sinusni signal bez odsjecanja signala.



Slika 12-6 Dinamicki opseg 8 bitnog ADC sa tri podesenja pojačanja.

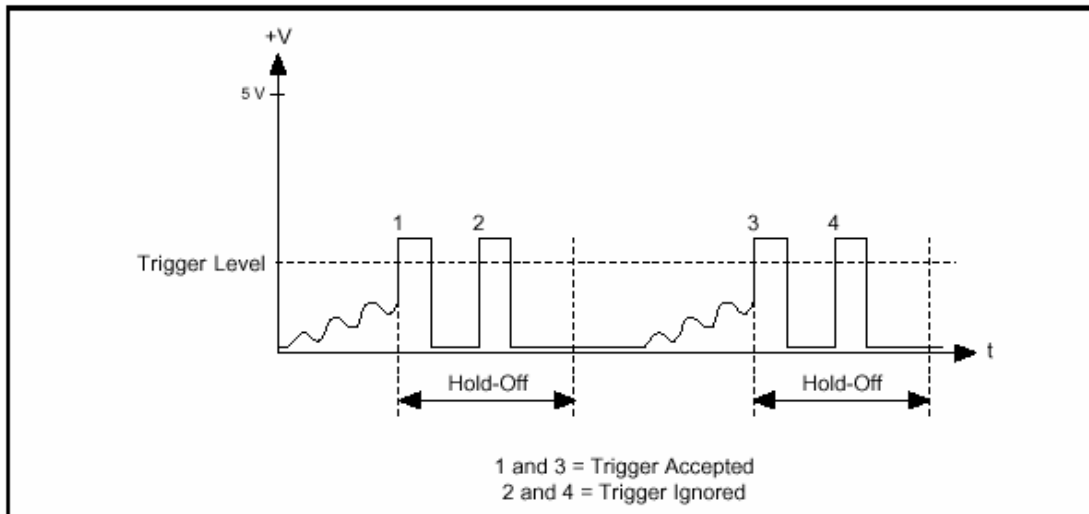
* **Impedansa izvora** - Vecina digitajzera i osciloskopa sa digitalnom memorijom (DSO- digital storage oscilloscopes) imaju ulazni otpor od $1\text{M}\Omega$ unutar propusnog opsega i sa 1x sondom, i $10\text{M}\Omega$ ulaznu impedansu sa 10x sondom. Ako je impedansa izvora prevelika signal ce biti prigusen na ulazu u pojačalo i mjerenje ce biti netacno. Ako je impedansa izvora nepoznata ali sa pretpostavkom da je velika, treba promjeniti prigusenje na sondi i prikupiti podatke sa ulaza. Ako 10x mjerenje rezultira u pojačanju amplitude, mjerenje ponovno moze biti netacno. Da bi se ovo korigovalo, pokusajmo smanjiti impedansu izvora pomocu baferovanja (vidjeti kasnije za objasnjenje).

Pored ulazne impedanse, svi digitajzeri, DSO-ovi, i sonde predstavljaju neki ulazni kapacitet u paraleli sa otporom. Ovaj kapacitet moze interferirati sa mjerenjem na isti nacin kao sto to cini i prethodno opisana otpornost. Moguce je reducirati ovaj kapacitet koristeći sondu sa prigusenjem (10x ili 100x) ili pak aktivnu sondu.

* **Ulazna frekvencija** – Ako je brzina uzorkovanja manja od 2 puta najveća frekventna komponenta u ulaznom signalu, frekventne komponente iznad polovine brzine uzorkovanja ce se alijasirati u propusnom opsegu na nizim frekvencijama, koje necemo moci razlikovati od ostalih frekvenci u propusnom opsegu. Ako je najveća frekvencija ulaznog signala nepoznata, treba poceti sa maksimalnom brzinom uzorkovanja da se predupredi alijasiranje, a zatim smanjivati brzinu uzorkovanja digitajzera sve dok displej ne pokaze ili dovoljno ciklusa valnog oblika ili informaciju koja nam je potrebna.

* **Opsti oblik signala** – Neke signale je jednostavno prikupiti koristeći standardne metode trigerovanja. Nekoliko iteracija sa nivoima trigerovanja dace stacionaran displej. Ovakav metod je uspjesan za sinusoidalne, trouglaste, kvadraticne i pila valne oblike. Neki od slozenijih valnih oblika , kao sto je slucaj sa nepravilnom povorkom impulsa, i tranzijenata mogu se

pokazati mnogo težim da se ulove. Slika 12-7 pokazuje primjer teskog trigerovanja povorke impulsa.



Slika 12-7 Tezak oblik signala za trigerovanje

Idealno, trigerski događaj se treba pojaviti kod uslova 1, ali ponekad instrument može trigerovati na uslov 2, posto signal presjeca nivo trigerovanja. Moguće je riješiti ovaj problem bez korištenja komplikovanih tehnika procesiranja signala koristeći *trigger hold-off* mogućnost, koja nam omogućava da specificiramo vrijeme nakon trigerskog događaja za koje ćemo ignorirati dodatne trigere koji bi se pojavili unutar ovog vremena. Sa adekvatnim vremenom zadrške (hold-off), valni oblik sa slike 12-7 se može adekvatno uloviti (capture), odbacujući uslove 2 i 4.

* **Ulazno kuplovanje** - Možemo konfigurirati ulazne kanale na DAQScope 5102 da budu DC ili AC kuplovani. DC kuplovanje dozvoljava DC i nisko frekventnim komponentama signala da prolaze kroz uređaj bez prigusenja. Nasuprot, AC kuplovanje odstranjuje DC ofsete i prigusuje nisko frekventne komponente signala. Ova osobina može biti iskoristena da se zumira na AC signale sa velikim DC ofsetom, kao što je naprimjer sum preklapanja na 12 V napojnom nivou.

Sonda i njeni efekti na valni oblik

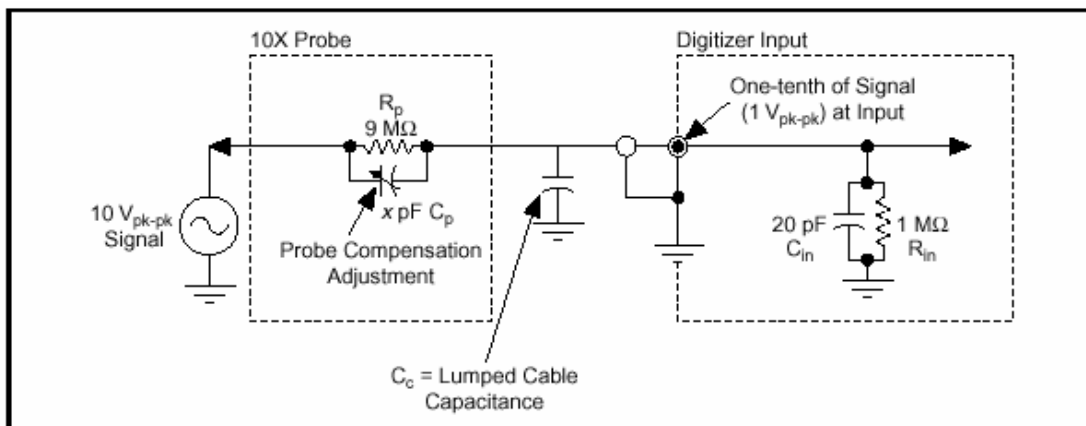
Signali putuju sa vrha sonde do ulaznog pojačala a nakon toga se uzorkuju i digitaliziraju od strane ADC. Staza koju signal prolazi cini sondu jednom važnom komponentom cijelog sistema mjerenja koja može značajno ugroziti tačnost mjerenja. Sonda može potencijalno uticati i na amplitudu i fazu izmjenjenog signala, a signal može pokupiti i dodatni sum sa sonde na svom putu do ulaznog pojačala. Postoji nekoliko tipova sondi kao što su:

pasivne, aktivne i strujne sonde.

Pasivna sonda

Pasivna sonda je najčešći korišten oblik sonde za osciloskope. Pasivne sonde su specificirane propusnim opsegom (ili vremenom porasta, rise time), odnosom

prigusenja (attenuation ratio), opsegom kompenzacije (compensation range), i aspektima mehanickog dizajna. Sonde sa prigusenjem , 10x ili 100x, imaju podesivi kapacitet koji moze reducirati kapacitivne efekte na ulazu. Mogucnost da ponisti ili minimizira efektivnu kapacitivnost na ulazu poboljsava propusni opseg sonde i vrijeme porasta. Slika 12-8 pokazuje tipicnu 10x sondu. Potrebno je da podesimo podesivi kapacitet, C_p , da dobijemo gladak frekventni odziv. C_p je kapacitet sonde (probe), R_p je otpor sonde, C_{in} je ulazna kapacitivnost , R_{in} je ulazna otpornost.



Slika 12-8 . Tipicna sonda 10x

Analiticki, dobijanje glatkog frekventnog odziva znaci :

$$R_{in} / (R_{in} + R_p) = C_p / (C_p + C_{in} + C_c)$$

Moze se pokazati da :

$$R_{in} (C_{in} + C_c) = C_p R_p ;$$

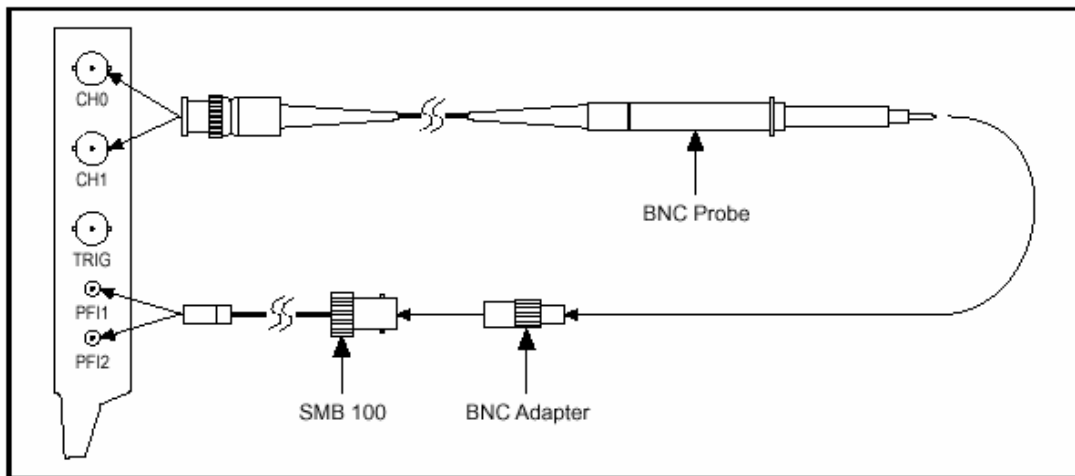
ili da je vremenska konstanta sonde jednaka vremenskoj konstanti ulaza digitajzera.

Kako kompenzirati sondu

Podesavanje podesivog kapaciteta sonde tako da dobijemo gladki frekventni odziv se naziva *kompenzacija sonde*. Na DAQScope 5102, moze se izabrati 0-5 V, 1 kHz niz impulsa kao referenca za ulaze PFI1 i PFI2. Pogledati sliku 12-9 u ovome opisu koraka da se kompenzira sonda:

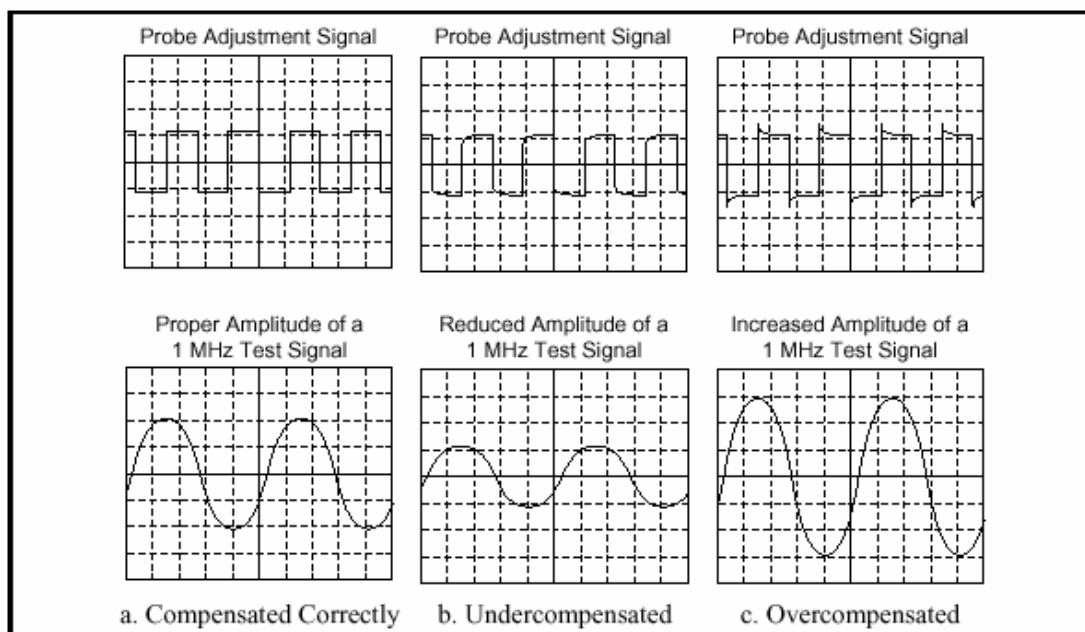
- * Spojiti BNC kraj sonde na ulazni kanal, bilo CH0 ili CH1.
- * Priklijuciti BNC adapter (iz pribora sonde) na vrh sonde.
- * Spojiti SMB100 kompenzacioni kabel sonde na jednu od PFI linija.
- * Priklijuciti sondu sa BNC adapterom na BNC zenski kraj SMB100 kabla.
- * Omoguciti kompenzacioni signal sonde na PFI liniji koju smo izabrali u koraku 3.
- * Digitizirati podatke na ulaznom kanalu.

* Podesiti podesivi kapacitet tako da valni oblik izgleda kao povorka cetvrtki bez izvitoperenja, sto je god bolje moguće.



Slika 12-9 Spajanje kompenzacionog kabla sonde

Kao sto je pokazano na slici 12-10, podkompenzirana sonda prigusuje signale veće frekvencije, dok prekompenzirana sonda pojačava veće frekvencije. Potrebno je često kalibrirati sondu da se obezbjedilo tačno mjerenje sa DAQScope 5102.



Slika 12-10 Poredjenja kompenzacije sonde

Aktivne i strujne sonde

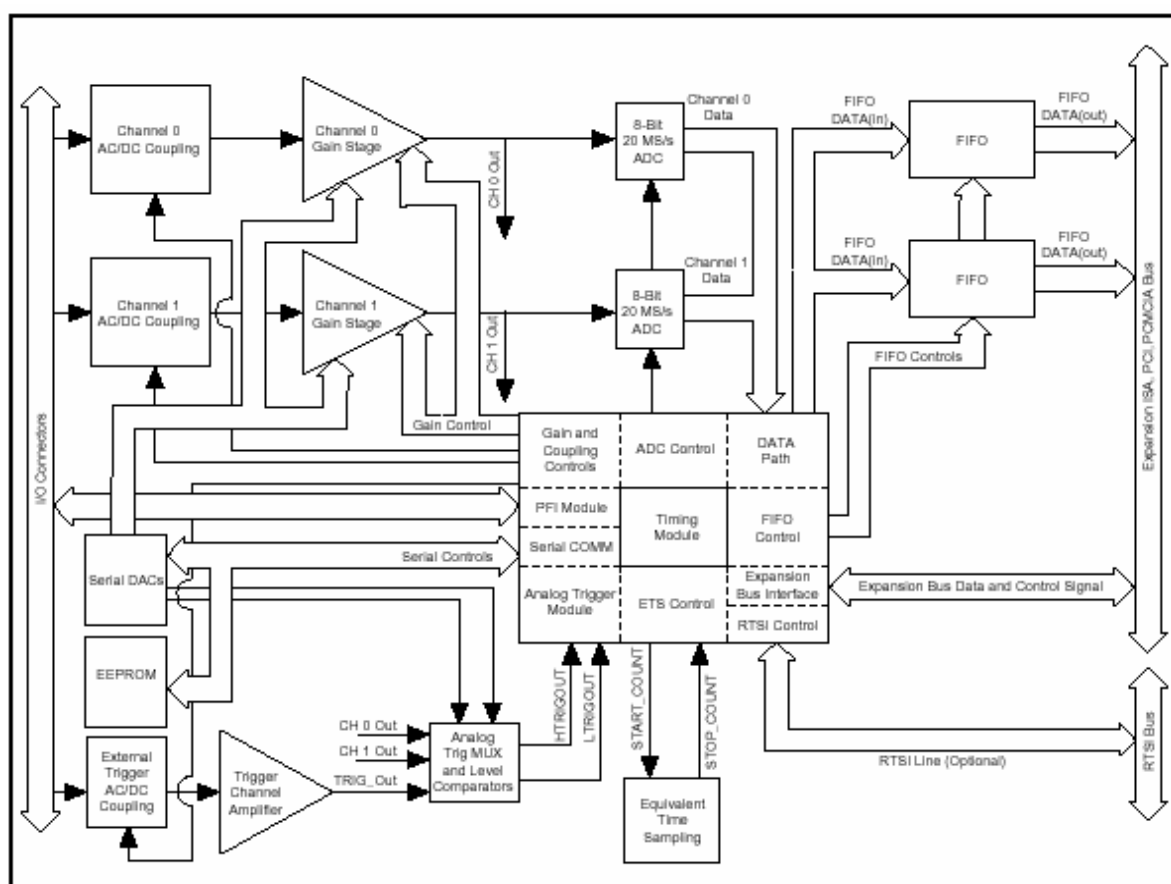
Mozemo takodjer koristiti aktivne sonde kao i strujne sonde sa digitajzerima i DSO.

Aktivne sonde kao sto su diferencijalne i sonde sa FET (field-effect transistor) tranzistorom, sadrze aktivno kolo u samoj sondi da bi odbacili sum i pojaicali signal. FET sonde su korisne za mjerenja nisko voltnih signala na visokim frekvencijama a diferencijalne sonde su poznate po svom visokom CMRR (common mode rejection ratio) i neuzemljnoj referentnoj tocki.

Umjesto koristenja serijskog otpora u konturi mjerenja struje, strujne sonde mjere AC i/ili DC struju koja tece u provodniku magnetskim putem (indukcija). Ovo nepostojanje serijskog otpora za preslikavanje struje u napon prouzrokuje vrlo malu interferenciju sa kolom u kojem se mjere strujni signali.

Hardware DAQScope 5102 modula

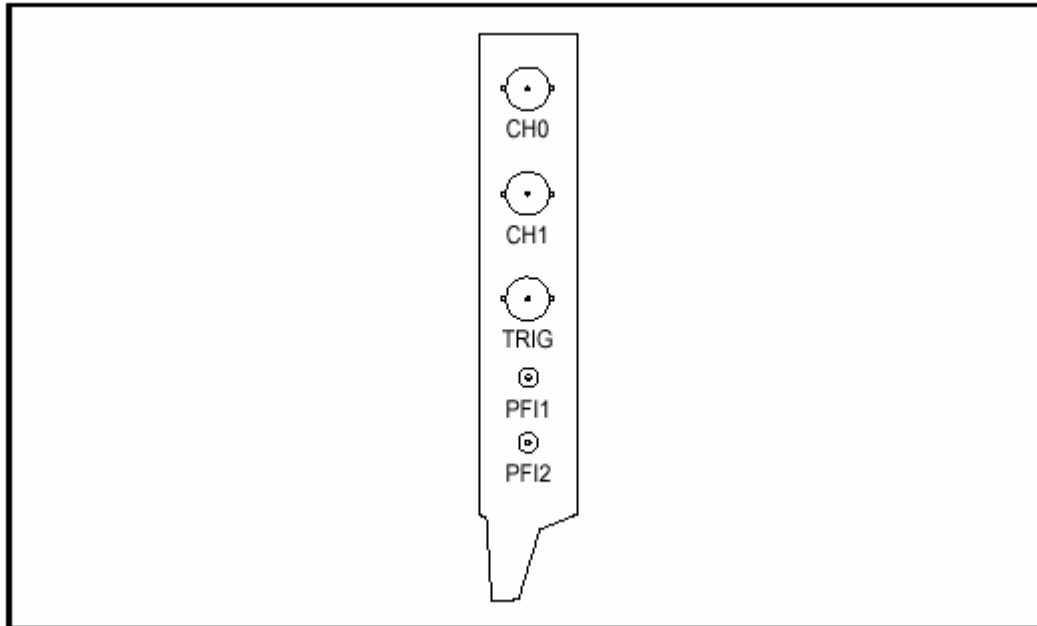
Naredna slika 12-11 pokazuje blok dijagram za DAQScope 5102 module.



Slika 12-11 Blok dijagram DAQScope 5102

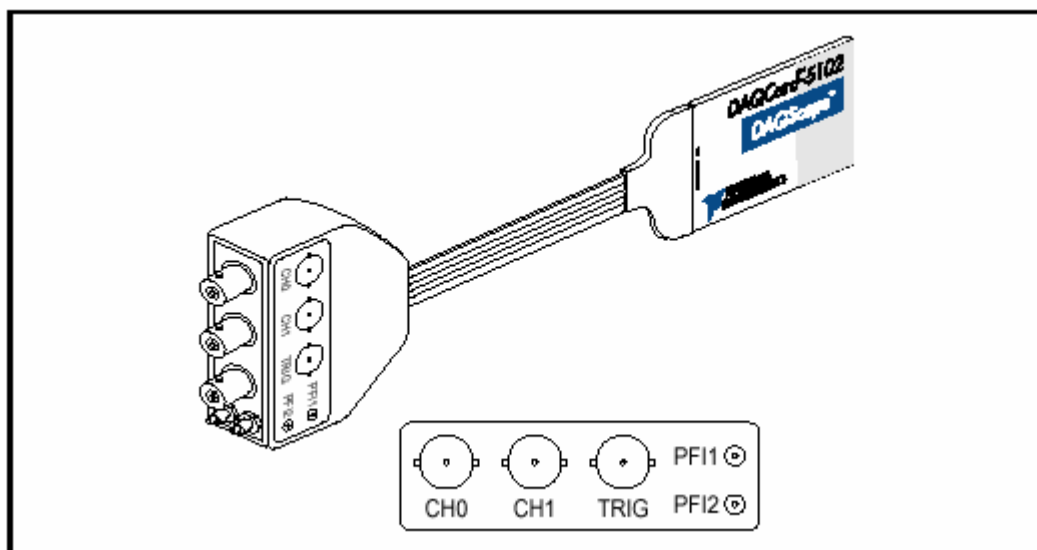
I/O konektor

DAQScope 5102 modul ima dva standardna BNC zenska konektora za CH0 i CH1 analogne ulaze, jedan standardni BNC zenski konektor za TRIG kanal, i dva standardna SMB zenska konektora za signale visenamjenskog digitalnog tajmiranja i trigerovanja, PFI1 i PFI2. Model PCI-5102 i AT-5102 daju direktne BNC prikljuke na prednjoj pločici modula, kako se to može vidjeti na narednoj slici 12-12.



Slika 12-12 PCI-5102 i AT-5102 modul sa konektorima na prednjoj ploči

Za model sa notebook-om DAQCard-5102 treba koristiti kablovski ansambl za ova spajanja koje isporučuje NI sa modulom, koji je prikazan na slijedećoj slici:



Slika 12-13 DAQCard-5102 konektor

Spajanja signala

Slijedeća tabela T12-1 daje opis I/O konektora

Signal	Opis
CH0, CH1	Digitalizira podatke i trigeruje akvizicije
TRIG	Koristi se kod vanjskog trigerovanja
PFI1, PFI2	Softwareski konfigurabilni digitalni trigeri ili digitalni izlazi

Mozemo koristiti CH0 i CH1 da digitaliziramo ulazne signale kao i da trigerujemo prikupljanje. Treba koristiti TRIG kanal samo za vanjsko trigerovanje; podatci na TRIG ulazu se nemogu digitalizirati. PFI1 i PFI2 su digitalni signali koje mozemo koristiti za tajming vremenski kriticnih aplikacija. Kada se koriste kao ulazi, PFI linije mogu trigerovati akviziciju, i/ili dozvoliti spoj sa vanjskim clockom. U izlaznom modu, PFI linije mogu izbaciti na izlaz Start Trigger, Stop Trigger, Scan Clock, i End of Acquisition signale, kao i analogni trigerski izlaz (ATC_OUT), frekventni izlaz, kao i informaciju o TTL niskom i visokom nivou. Imena signala i njihov opis variraju sa modom akvizicije koji se koristi.

Analogni ulaz

Dva analogna ulazna kanala su referencirani na zajednicku masu u bipolarnom modu. Ova podesenja su fiksna; zbog toga niti referentna tacka niti polarnost ulaznih signala se ne moze mjenjati. To znaci da ne mozemo koristiti CH0 ili CH1 za diferencijalna mjerenja ili da mjerimo floating signale(bez reference), izuzev da oduzmemo dva digitalna valna oblika u software-u. Za tacna mjerenja , treba obezbjediti da se mjerni signal na ulazu referencira na istu masu kao i sam DAQScope 5102, na taj nacin sto cemo prikljuciti krokodil stipaljku za uzemljenje na sondi, za signalnu masu. Naredna tabela T12-2 pokazuje ulazne opsege koji su na raspolaganju na kanalima CH0 i CH1.

Pojacanje	Input range		
	1X Probe	10X Probe	100X Probe
1	±5 V (default setting)	±50 V	±500 V
5	±1 V	±10 V	±100 V
20	±0.25 V	±2.5 V	±25 V
100	±50 mV	±0.5 V	±5 V

Tabela T12-2 Ulazni opsezi kanala CH0 i CH1

Napomena: Oznake 10x i 100x oznacavaju prigusenje , ne pojacanje. Naprimjer, sa 100x sondom i pojacanjem 1, ako mjerimo 400 V signal, DAQScope 5102 ce primiti 4 V (400/100=4V) na ulaznom konektoru.

TRIG kanal ima fiksni ulazni opseg od $\pm 5V$. Sve verzije DAQScope 5102 startaju sa default vrijednoscu pojacanja od 1, i time dozvoljavaju najveći mogući ulazni opseg. Vrijednosti opsega TRIG kanala su iste kao one sa pojacanjem od 1 u tabeli T`12-2.

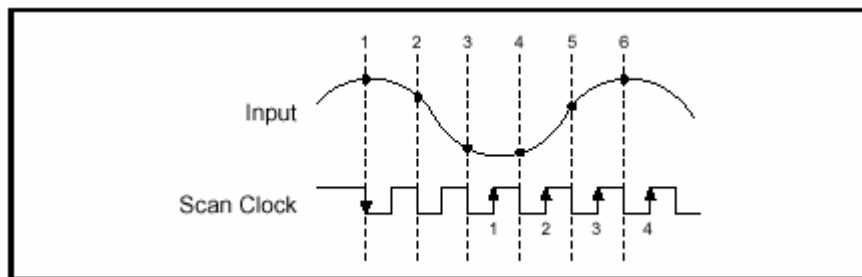
Kanali CH0, CH1 i TRIG imaju softwareski programabilnu selekciju kuplovanja izmedju AC i DC. Koristicemo AC kuplovanje (prikljucenje) kada AC signal na ulazu sadrzi veliku DC komponentu. Bez AC kuplovanja bilo bi tesko posmatrati detalje AC komponenti sa velikim DC ofsetom i malom AC komponentom, kao sto je naprimjer sum kod switching DC napojnih jedinica. Ako omogucimo AC kuplovanje, otklanjamo veliki DC ofset sa ulaznog pojacala i pojacavamo samo AC komponentu. Ova tehnika omogucava efektivno koristenje dinamicnog opsega , da bi digitizirali samo signal koji nam je od interesa. Nisko frekventni ugao kod AC kuplovanog kola je frekvencije koja je ispod one kod koje su signali priguseni za najmanje 3 dB. Nisko frekventni ugao je 11 Hz sa 1x sondom, 1.1 Hz sa 10x sondom, i 0.11 Hz sa 100x sondom.

Kada se mjenja nacin prikljucenja (kuplovanja) ulaznog signala kod DAQScope 5102, ulaznom stepenu je potrebno neko konacno vrijeme da se setluje (smiri), kao sto se to vidi sa slijedece tabele T12-3

Akcija	Vrijeme smirenja
preklapanje sa AC kuplovanja na DC	0.5 ms
Preklapanje sa DC kuplovanja na AC	
1x sonda	15 ms
10x sonda	150 ms
100x sonda	1.5 ms

ADC pipeline kasnjenje

ADC pretvarac na DAQScope 5102 je pipelined flash konvertor sa maksimalnom brzinom konverzije od 20MS/s. Pipeline arhitektura namece 2.5 scan clock ciklus kasnjenje , da bi pretvorila analogni signal u digitalnu vrijednost , kao sto je to prikazano na narednoj slici:



Slika 12-14 Kasnjenje scan clocka

U odnosu na Scan clock signal, digitalna vrijednost koja odgovara prvoj konverziji (prva opadajuca ivica scan clock signala), izbacuje izlaz sinhrono sa trecim rastucom ivicom scan clock signala.

Koristeci pipeline arhitekturu se takodjer uvodi niza granica na brzinu uzorkovanja. Tako naprimjer za DAQScope 5102 tacnost pocinje opadati ispod 1 kS/s.

DAQScope 5102 je dizajniran da se automatski podesava za pipeline kasnjenje kada se koristi interni scan clock. Ako koristimo vanjski scan clock moramo obezbjediti nezavisni clock da bi osigurali pouzdan rad.

Modovi akvizicije

DAQScope 5102 podrzava dva akviziciona moda: posttrigerska akvizicija i pretrigerska akvizicija.

Posttrigerska akvizicija

U posttrigerskom akvizicionom modu, hardware prikuplja uzorke nakon sto se scan triger signal pojavio. kada se desi triger, ulazni signal se digitalizira i zeljeni broj uzoraka se pohranjuje u memoriju na samom modulu. Tabela T12-4 pokazuje minimalni i maksimalni broj uzoraka koje 5102 uredjaj moze prikupiti.

Number of Channels	PCI-5102		AT-5102 and DAQCard-5102	
	Min	Max	Min	Max
One	360	16,777,088*	360	663,000
Two	180	16,777,088*	180	331,500

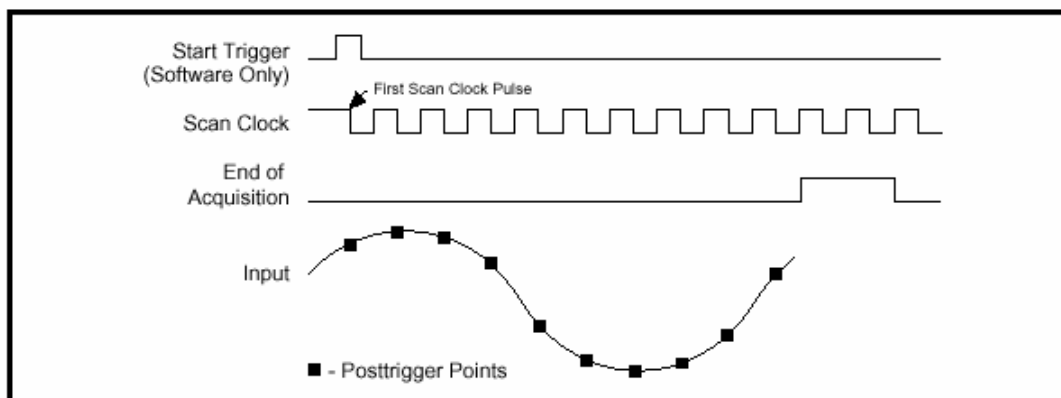
* Zavisi od raspolozive memorije u PC-ju

Tabela T12-4 Moguci broj uzoraka za posttrigerski scan

Kod modula AT-5102 i DAQCard-5102 , prenos podataka sa modula u glavnu memoriju PC se vrši nakon zavrsetka akvizicije, ogranicavajući na taj način broj uzoraka na onaj koji moze stati u memoriju modula.

Kod modula PCI-5102, podaci se mogu vrlo brzo prebacivati sa modula u host memoriju PC-ja, dok je akvizicija jos u toku. Modul je realiziran sa **NI** MITE ASIC (application specific integrated circuit) kolom, tako da kontrolise PCI bus i prebacuje prikupljene podatke na obadva kanala u PC memoriju u realnom vremenu bez gubitka podataka. Ova tehnologija omogucava akviziciju vise nego 663.000 uzoraka koliki je kapacitet memorije na modulu. Ova mogucnost prosirojuje broj uzoraka koji se moze skanirati u jednom shotu na 16 miliona uzoraka.

Slijedeca slika 12-15 pokazuje timing signale ukljucene u posttrigersku akviziciju. U ovom primjeru, hardware je programiran tako da prikuplja 10 posttriger uzoraka. Posttriger akvizicioni mod se inace koristi samo za one-shot (jednookidne) softwareski trigerovane akvizicije.



Slika 12-15 Posttrigger akvizicija

Pretrigerska akvizicija

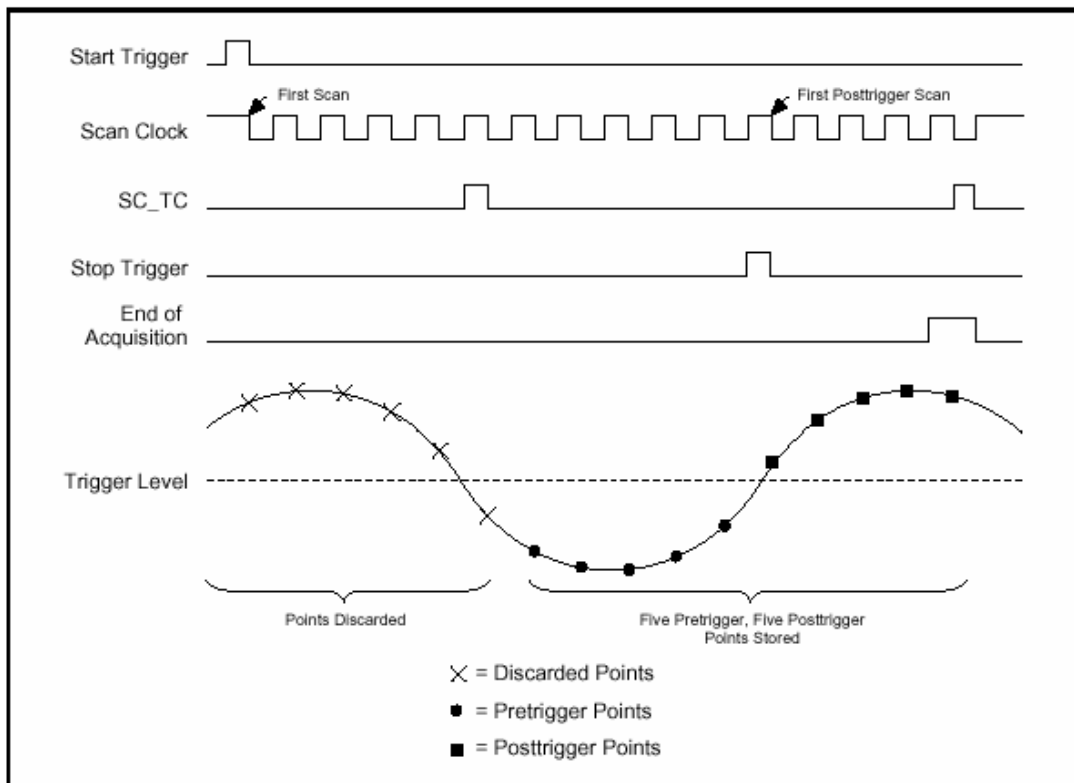
Kod moda pretrigerske akvizicije uređaj prikuplja određeni broj uzoraka, koji se naziva broj pretrigger uzoraka (pretrigger scan count), **prije** nego što se trigger pojavi. Nakon zadovoljenja zahtjeva broja pretrigger uzoraka, hardware nastavlja da prikuplja podatke i pohranjuje ih u prstenasti bafer implementiran u memoriji na samom modulu. Velicina ovog prstenastog bafera jednaka je broju pretrigger uzoraka. Kada se trigger pojavi, hardware prikupi i pohrani broj postrigger uzoraka koji je zadat i završi akviziciju. Tabela T12-5 pokazuje minimalni i maksimalni broj uzoraka koji je raspoloživ za DAQScope 5102 u pretrigerskom modu rada.

Broj kanala	PCI-5102		AT-5102 and DAQCard-5102	
	Min	Max	Min	Max
One				
Pretriggered scans	360	663,000	360	663,000 - (the number of posttriggered scans)
Posttriggered scans	10	16,777,088*	10	663,000 - (the number of pretriggered scans)
Two				
Pretriggered scans	180	331,500	180	331,500 - (the number of posttriggered scans)
Posttriggered scans	5	16,777,088*	5	331,500 - (the number of pretriggered scans)

* zavisi od raspoložive memorije

Tabela T12-5 Moguci broj uzoraka kod pretrigger moda rada

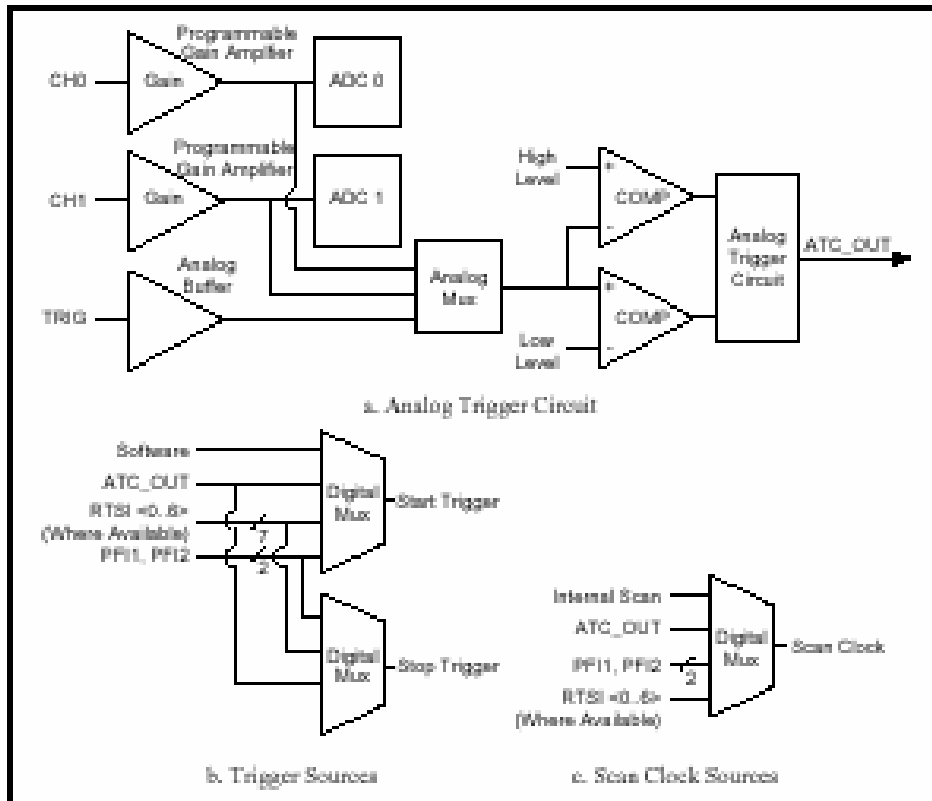
Naredna slika br. 12-16 pokazuje relevantne timing signale za tipični pretrigger mod akvizicije. Ilustracija prikazuje 5 pretrigger i 5 postrigger uzoraka. Kao trigger se koristi analogni trigering iznad postavljenog nivoa.



Slika 12-16 Pretrigerski mod akvizicije

Izvori trigerovanja

Scan clock, Start Trigger, i Stop trigger signali se mogu generirati kroz software ili su obezbjedjeni izvana kao digitalni ili analogni triger signali na jednom od ulaza (CH0, CH1) ili TRIG kanalu. Slika 12-17 pokazuje razlicite izvore trigerera. Dodatno, Scan Clock je na raspolaganju iz izvora (brojac) interno ka DAQScope 5102.

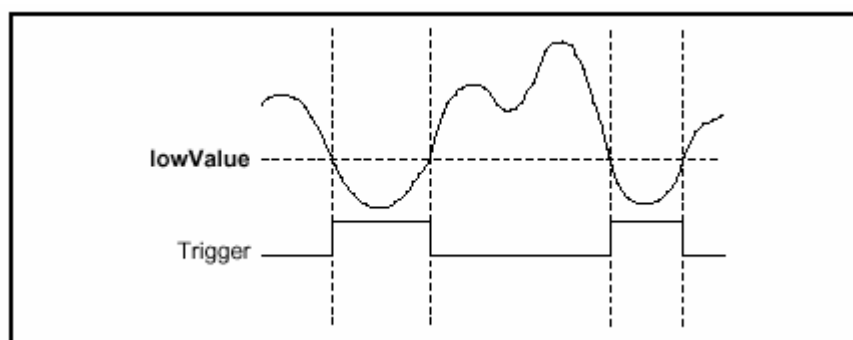


Slika 12-17 Izvori Scan clock, Start Trigger, i Stop Trigger signala

Kolo analognog trigerera

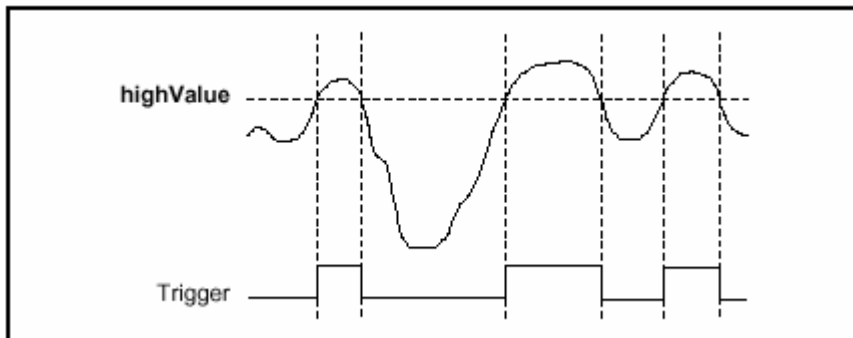
DAQScope 5102 sadrži sofisticirano kolo analognog trigerera koje prihvata Boolove izlaze iz komparatora nivoa i donosi inteligentne odluke o trigeru. Na raspolaganju je 5 analog trigering modova rada, kao što je prikazano na slikama 12-18 do 12-22. Moguće je postaviti u softveru nezavisno vrijednosti **low value** i **high value** (donju i gornju vrijednost).

U low level analognom trigerskom modu rada, triger se generise kada vrijednost signala je ispod **low value**. **High value** se ne koristi.



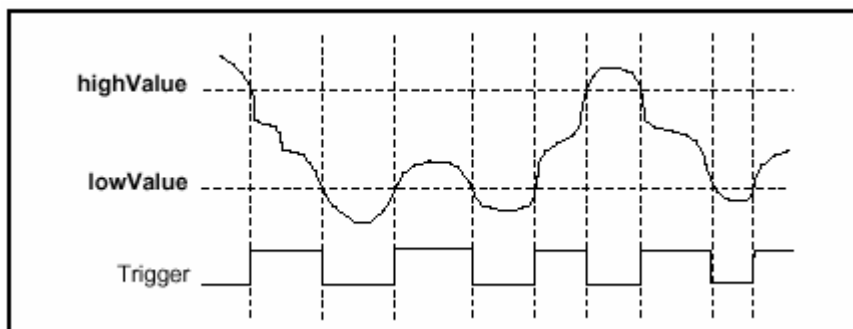
Slika 12-18 Ispod donje vrijednosti (low value) analogni trigering mode

Kod iznad gornjeg nivoa analognog trigerskog moda, triger se generise kada signalna vrijednost je veca od vise vrijednosti (**high value**). Low value se ne koristi.



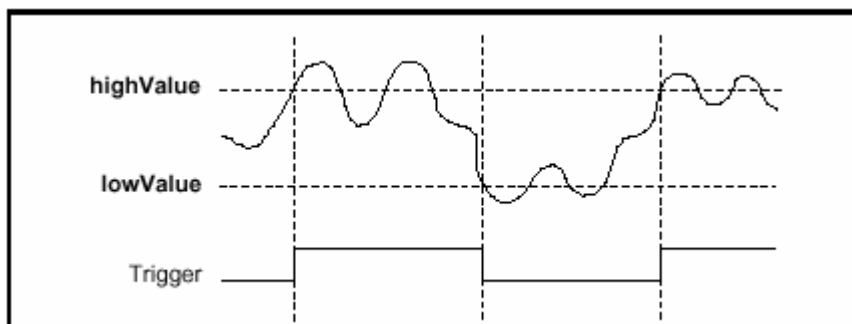
Slika 12-19 Iznad **high value** analogni trigerski mod

Kod analognog trigerskog moda , unutar regiona, triger se generira kada je vrijednost signala izmedju **low value** i **high value**.



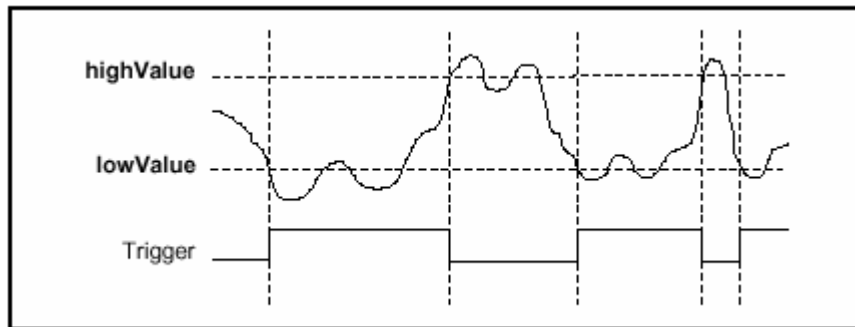
Slika 12-20 Analogni trigerski mod unutar regiona

Kod gornjeg – histerezis analognog triger mod, triger se generira kada je vrijednost signala veca od **high value** , sa histerezisom specificiranim sa **low value**.



Slika 12-21 Gornji histerezis analogni trigerski mod

Kod donjeg – histeresis analognog trigerskog moda, triger se generira kada je vrijednost signala niza od **low value**, sa histeresisom specificiranim sa **high value**.

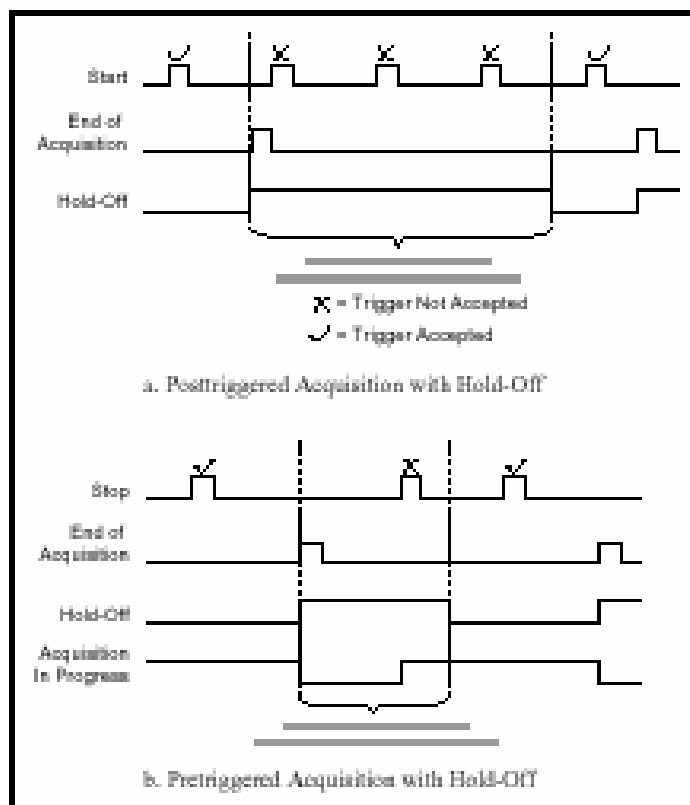


Slika 12-22 Donji histerezis analogni trigerski mod

Trigerska zadrška (hold-off)

Trigerski hold-off je obezbjedjen u hardwareu koristeći 24-bitni brojcaš nadole (down counter), clockiran (vremenski sinhronizovan) sa 2.5 MHz internom vremenskom bazom. Sa ovom konfiguracijom moguće je selektirati hardware-sku hold-off vrijednost od 800 ns do 6.71 s u inkrementima od 400ns.

Kada je akvizicija u toku, brojcaš je napunjen sa digitalnom vrijednošću koja odgovara željenom hold-off vremenu. Signal End of Acquisition (kraj akvizicije) trigeruje brojcaš da bi startao odbrojavanje nadole. Prije nego što brojcaš dostigne svoj konacni iznos (TC), svi trigeri se odbacuju u hardwareu. Kod trenutka TC, hold-off brojcaš se ponovo puni sa hold-off vrijednosti i čeka na End of Acquisition da ponovi opisani proces. Slika 12-23 pokazuje timing dijagram signala kada je hold-off omogućen.

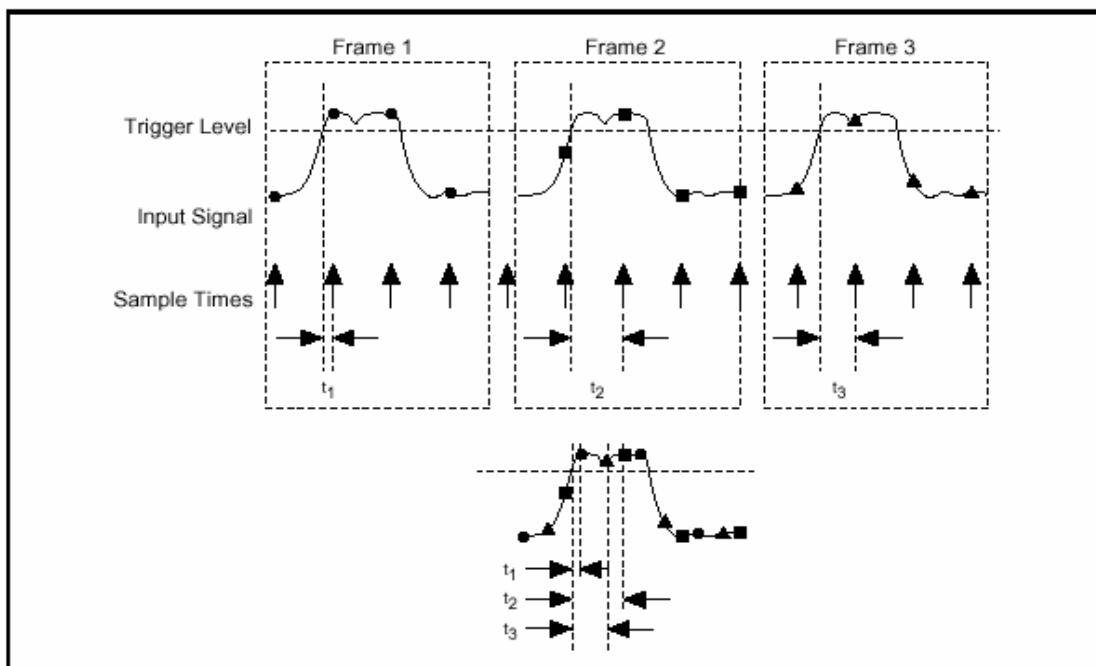


Slika 12- 23 Pretriger i Posttriger akvizicije sa hold-off

Slučajno uzorkovanje sa preklapanjem (Random interleaved sampling -- RIS)

Slučajno uzorkovanje sa preklapanjem (RIS) je oblik analogan ekvivalentnom vremenu uzorkovanja (ETS- equivalent time sampling), koji dozvoljava prikupljanje pretrigerovanih uzoraka. ETS-om se naziva bilo koji metod koristen za uzorkovanje signala na takav nacin da prividna brzina uzorkovanja je veca od stvarne brzine uzorkovanja. ETS se postize uzorkovanjem razlicitih tacaka duz valnog oblika kod svakog pojavljivanja trigeru, a zatim rekonstrukcija valnog oblika iz podataka prikupljenih za vrijeme mnogo ciklusa.

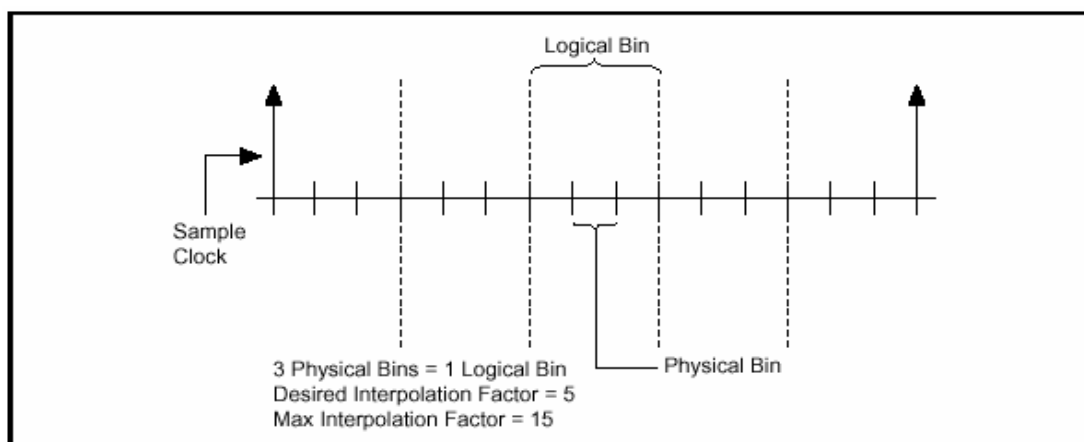
Kod RIS metoda stizanje trenutka trigerovanja valnog oblika pojavljuje se u nekom trenutku vremena koje je slucajno rasporedjeno izmedju dva trenutka uzorkovanja. Ovo vrijeme izmedju trigeru i slijedeceg trenutka uzorkovanja se mjeri, i ovo mjerenje dozvoljava rekonstrukciju valnog oblika. Slika 12-24 pokazuje tri pojavljivanja valnog oblika. U okviru 1, tackaste vrijednosti se uzorkuju, i triger se pojavljuje u vremenu t_1 prije slijedeceg uzorkovanja. U okviru 2, uzorkuju se kvadraticne tacke, i triger se pojavljuje u vremenu t_2 prije slijedeceg uzorkovanja. U okviru 3, trouglaste tacke se uzorkuju, i triger se pojavljuje u vremenu t_3 prije slijedeceg uzorkovanja. Sa poznavanjem ova tri vremena t_1 , t_2 i t_3 , mozemo rekonstruirati valni oblik signala kao da je bio uzorkovan sa vecom brzinom uzorkovanja, kao sto je to vidljivo sa naredne slike.



Slika 12-24 Rekonstrukcija valnog oblika sa RIS metodom

Mjerenje vremena je ostvareno sa vremenom do digitalnog pretvaraca (TDC – time to digital converter). Rezolucija TDC je broj fizikalnih cjelina (korpi) na koje TDC može kvantizirati vrijeme stizanja trigera. Ova rezolucija treba biti nekoliko puta veća nego maksimalni željeni interpolacioni faktor, koji je maksimalni broj logičkih cjelina sa kojim mi želimo da trigersko vrijeme dolaska bude kvantizirano. Veća rezolucija obezbjeđuje da se izlaz TDC ponovno kvantizira na željeni interpolacioni faktor, sve izlazne vrijednosti će imati otprilike istu vjerovatnoću da se pojave, tj. sve logičke cjeline će sadržavati aproksimativno isti broj fizikalnih cjelina.

Na primjer, pretpostavimo da je maksimalni interpolacioni faktor 5. Ako TDC može da izbaci na izlaz vrijednost 0-15, tada svaka logička cjelina će sadržavati tri fizičke cjeline, kao što je prikazano na slici 12-25.



Slika 12-25 Relacija između interpolacionog faktora, logičkih cjelina (korpi-bins) i fizičkih cjelina (korpi – bins)

Maksimalni interpolacioni faktor na DAQScope 5102 je 50, sto daje maksimalnu ETS brzinu od 1 GS/s. Pri ovoj brzini, odnos logickih cjelina nasprema fizickim cjelinama je aproksimativno 1:9.

Da bi se rekonstruirao valni oblik sa RIS, potrebno je da znamo RIS OFFSET, koji predstavlja minimalnu vrijednost koju TDC moze vratiti, a opseg vrijednosti, RIS GAIN (pojacanje), je $\max(\text{TDC vrijednost}) - \min(\text{TDC vrijednost})$.

RIS GAIN se koristi da se odredi broj fizickih cjelina po logickoj cjelini za zeljeni interpolacioni faktor.

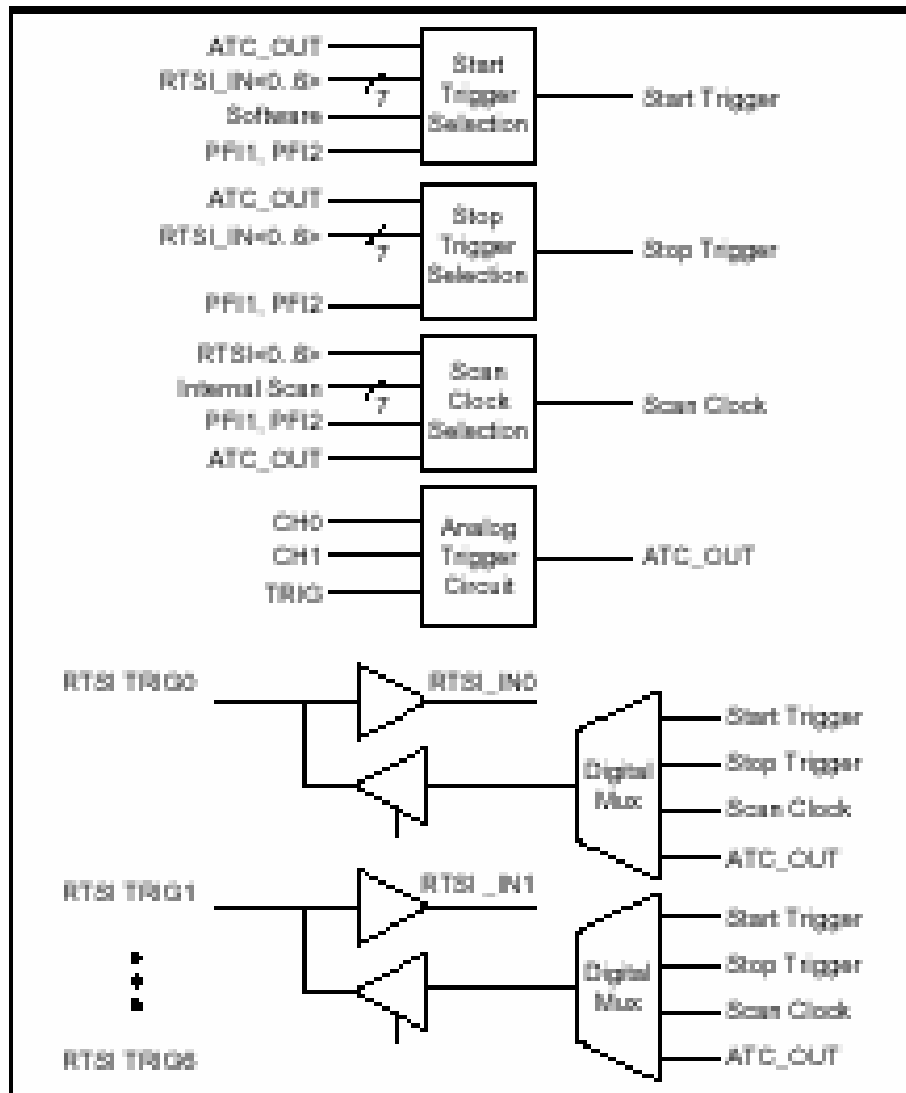
Napomena: ETS i RIS mogu raditi samo sa repetitivnim signalima.

RTSI bus trigger i clock linije

PCI – 5102 i AT – 5102

RTSI bas (koji nije prisutan kod verzije za notebook tj. DAQCard-5102), dozvoljava **NI** modulima da sinhronizuju timing i triggerovanje kod visestrukih modula. RTSI bas ima 7 bidirekcionalnih triggerskih linija i jednu biderkcionalnu clock signal liniju.

Mi mozemo programirati bilo koju od sedam triggerskih linija kao ulazne, da obezbjedimo Start Trigger, Stop Trigger, i Scan Clock signale ciji je izvor na Master modulu. Slicno, mozemo programirati master modul da izbacii na izlaz svoje interne signale: Start Trigger, Stop Trigger, i Scan Clock i ATC_OUT , na bilo koje od 7 triggerskih linija, kao sto je to pokazano na narednoj slici 12-26.



Slika 12-26 RTSI bas trigerske linije

RTSI bas clock linija je specijalna clock linija na RTSI basu koja može samo da nosi vremensku bazu sa master modula na slave modul. Da bi postigli najmanji jitter (podhrtavanje) između mjerenja na različitim modulima, potrebno je konfigurirati da slave moduli koriste RTSI bas clock iz master modula.

PFI linije

Sve verzije DAQScope 5102 uređaja imaju dvije programabilne digitalne ulazno/izlazne linije, PFI1 i PFI2, koje se mogu koristiti za vanjski timing i trigerovanje ili da se izbace na izlaz različiti signali na modulu. Smjer ovih linija se individualno izabire da bude ulaz ili izlaz.

PFI linije kao ulazi

PFI1 i PFI2 se mogu izabrati kao ulazi za Start Trigger, Stop Trigger, i Scan Clock signale.

PFI linije kao izlazi

PFI1 i PFI2 se mogu izabrati da izbace na izlaz slijedece digitalne signale:

- * **Start Trigger** – Ovaj signal je sinhroniziran sa 20 MHz vremenskom bazom. Kada je start uslov ispunjen, bilo kroz software ili analogni ili digitalni trigger, Start Trigger ce porasti na visu vrijednost (logicki 1) za 100 ns (tj. za 2 clock perioda 20 MHz-ne vremenske baze) i vratiti se natrag u svoje prethodno stanje.
- * **Stop Trigger** – Ovaj signal je sinhroniziran sa 20 MHz vremenskom bazom. Kada je start uslov ispunjen, bilo kroz software ili analogni ili digitalni trigger, Stop Trigger ce porasti na visu vrijednost (logicki 1) za 100 ns (tj. za 2 clock perioda 20 MHz vremenske baze) i vratiti se natrag u svoje pocetno stanje.
- * **Scan clock** – Ovaj signal je takodjer clock za ADC koji predstavlja brzinu kod koje je ulaz uzorkovan. Default stanje ovog signala je visok (high).
- * **End of Acquisition** – Ovaj signal je interno generiran da indicira unutarnjim djelovima modula da je akvizicija zavrшена. End of Acquisition sinhronizovan sa sa Scan clock , ostaje high za vrijeme od 2 Scan clock perioda na kraju akvizicije. Ovaj signal se moze upotrebiti da trigeruje vanska kola za vremenski kriticne aplikacije.
- * **ATC_OUT** – Ovaj signal je digitalni izlaz iz kola analognog trigeru na DAQScope 5102. Frekvencija i duty ciklus (odnos pozitivne nasprema negativne poluperiode), ovoga signala zavisi od trigerskog kanala , donje i gornje vrijednosti trigerskog nivoa, polariteta i trigerskog moda.
- * **Frekventni izlaz** – Ovaj signal je digitalna povorka impulsa sa programabilnom frekvencijom. Najcesca primjena frekventnog izlaza je da obezbjedi signal za kompenzaciju sonde. Moguce je izabrati dvije vremenske baze da se generira ova frekvencija , tj.:

7.16 MHz (asinhrono sa 20 MHz internom vremenskom bazom)

1.25 MHz (sinhrono sa 20 MHz internom vremenskom bazom)

DAQScope 5102 koristi 16 bitni brojca da programski izabere frekvenciju na izlazu. Frekvencija povorka impulsa kao funkcija vrijednosti brojaca moze biti izrazena kao:

Frekvencija= vremenska baza / koeficijent djeljenja
gdje:

koeficijent djeljenja = 3...65.535

Alternativno, da se izracuna koeficijent djeljenja za datu frekvenciju, relacija je :

koeficijent djeljenja = vremenska baza/frekvencija

Naprimjer, da se generira povorka impulsa od 1kHz, koja je uobicajena za ranije opisanu proceduru kalibracije, potrebno je izabrati slijedece parametre:

vremenska baza= 1,25 MHz

koeficijent djeljenja= 1,250

8. **Low** – Ovak TTL niski nivo signala je referenciran prema potencijalu uzemljenja PC. Ovo je signal logickog niskog nivoa. Medjutim ne treba ga koristiti kao GND za kolo.

* **High** – Ovo je TTL visoki nivo logickog signala, referenciran prema potencijalu uzemljenja PC-ja. Medjutim ne treba ga koristiti kao VCC tacku za 5 V napajanje logickog kola.

Master/Slave rad osciloscopa

Mozemo koristiti dva ili vise DAQScope 5102 modula u jednom sistemu da povecamo broj kanala za nasu primjenu, sinhronizirajuci ove module preko RTSI basa ili preko I/O konektora.

Moguće je takodjer konfigurirati sistem u kojem PCI-5102 ili AT-5102 je master modul koji kontrolira mjesavinu od PCI-5102 i AT-5102 slave modula. Za ovo nam je potreban RTSI bas kabel da ostvari ovu sinhronizaciju kako slijedi:

* Spojiti master modul sa slave modulom preko RTSI konektora. I kabel i konektor su upareni (keyed) tako da postoji samo jedan nacin njihovog medjusobnog spajanja.

* Obezbjediti da niti jedan drugi modul u sistemu nije konfiguriran da izbacii na izlaz svoju internu vremensku bazu na RTSI bas clock liniju.

* Programirati master modul da izbacuje na izlaz svoju internu vremensku bazu na RTSI bas clock liniju.

* Programirati master modul da izbacii na izlaz svoje Scan Clock i Stop trigger signale na nekoristene RTSI bas triger linije.

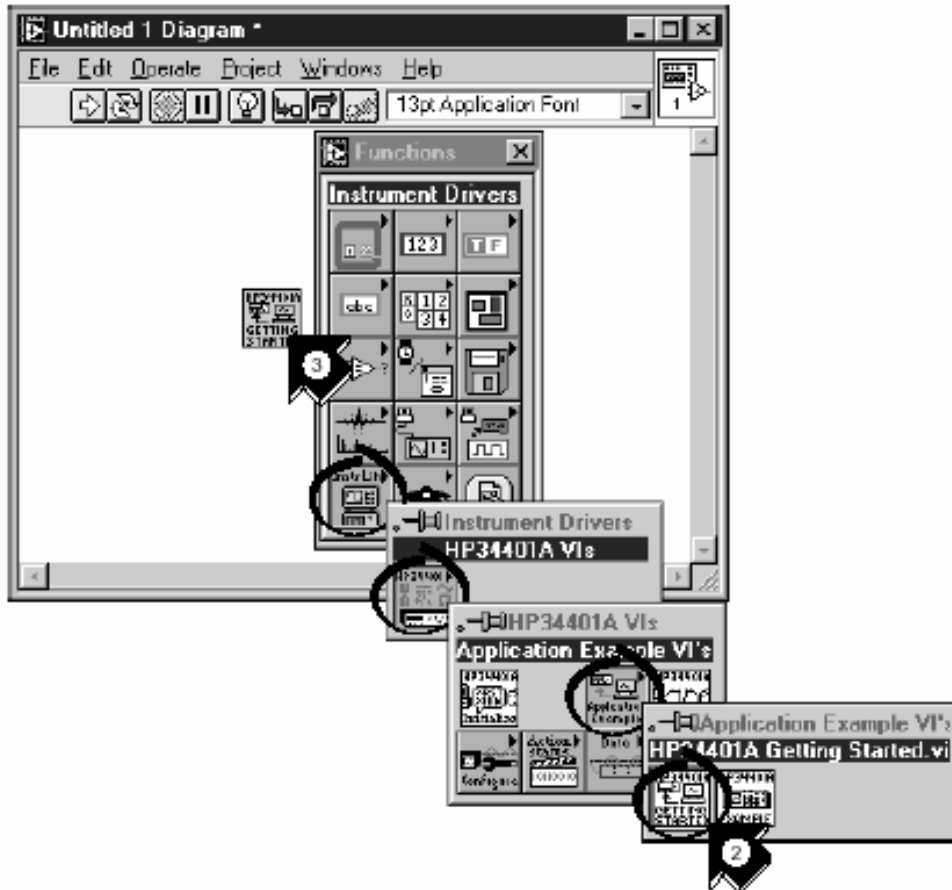
* Programirati slave modul da koristi RTSI bas clock kao svoju glavnu vremensku bazu.

* Programirati slave module da koriste vanjski RTSI Scan Clock i Stop Trigger sa RTSI bas triger linija, izabranih u koraku 4

PRIMJER INSTALIRANJA INSTRUMENT DRIVERA

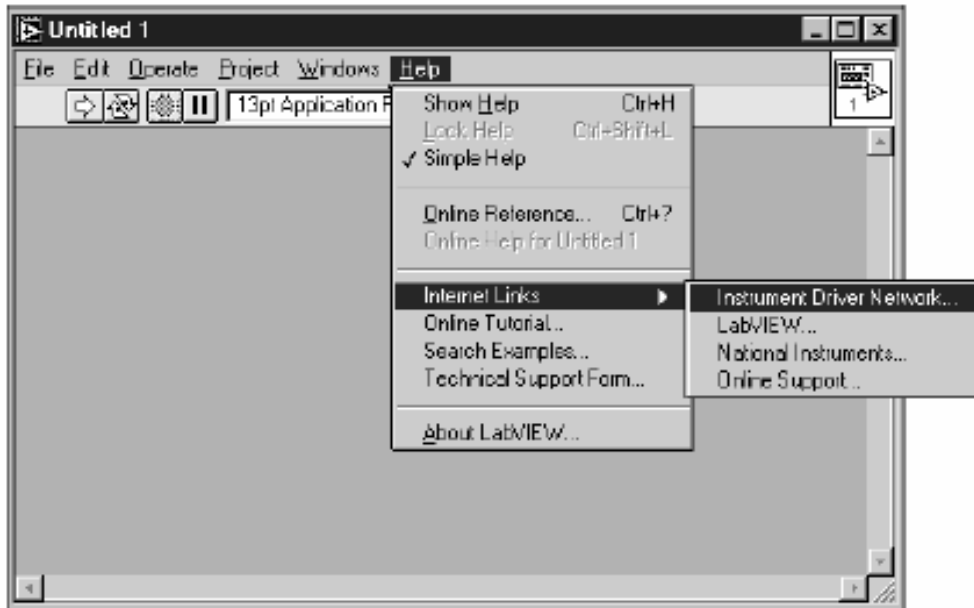
U nastavku ovog poglavlja pokazacemo nacin instaliranja jednog instrument drajvera iz obimne biblioteke **NI** za skoro svaki instrument poznatijih proizvođača mjernih instrumenata. U ovom primjeru bit ce to instrument drajver za Hewlett Packard-ov instrument HP34401A: Digitalni multimeter spojen preko GPIB kontrolera sa PC-jem

1. Kreirajmo novu VI za ovaj instrument i predjimo u blok dijagram :



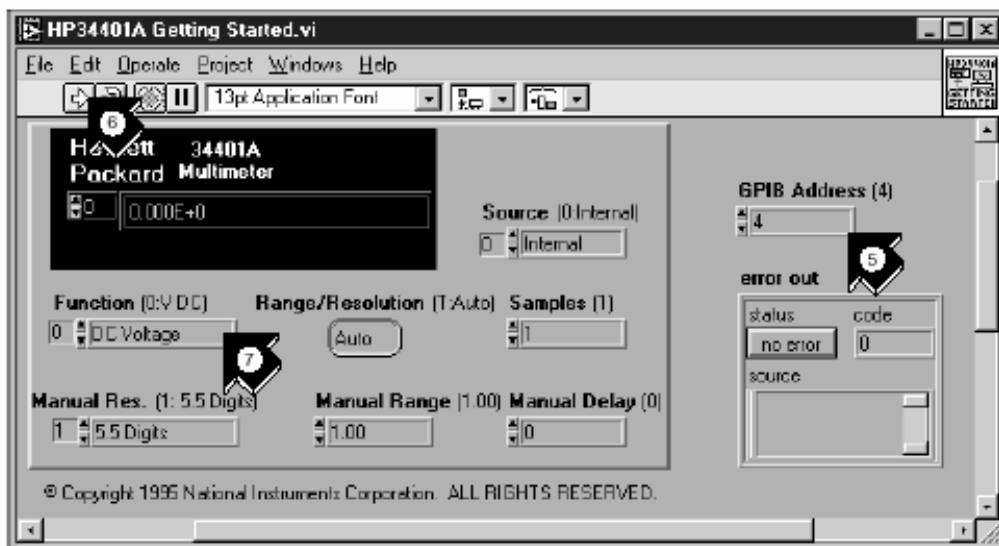
2. Izaberimo **Functions>> Instrument Drivers>> HP34401A Vis>> Application Example VI's >> HP34401A getting Started.vi**, ili izaberimo ime nekog drugog instrumenta kojeg zelimo povezati sa PC i nadjimo njegovu **Getting Started.vi** .

Ako ne mozemo naci nas instrument kojeg zelimo povezati, treba provjeriti u LabVIEW CD ili *Instrument Driver Library* CD i instalirati drajvere za nas zeljeni instrument u subdirektorij *instr.lib* u LabVIEW direktoriju.



3. Postaviti VI na blok dijagram

4. Dva puta kliknuti na HP34401A Getting started VI da bi dobili prednji panel i blok dijagram. Ova VI kontrolise HP34401A digitalni multimetar HP. Ova Getting started VI je jedan primjer visoko nivovske programske rutine koja poziva instrument drajversku subVI da bi upravljala instrumentom.



5. Na prednjem panelu ćemo provjeriti GPIB adresu od HP34401A getting started VI. default adresa je 4.



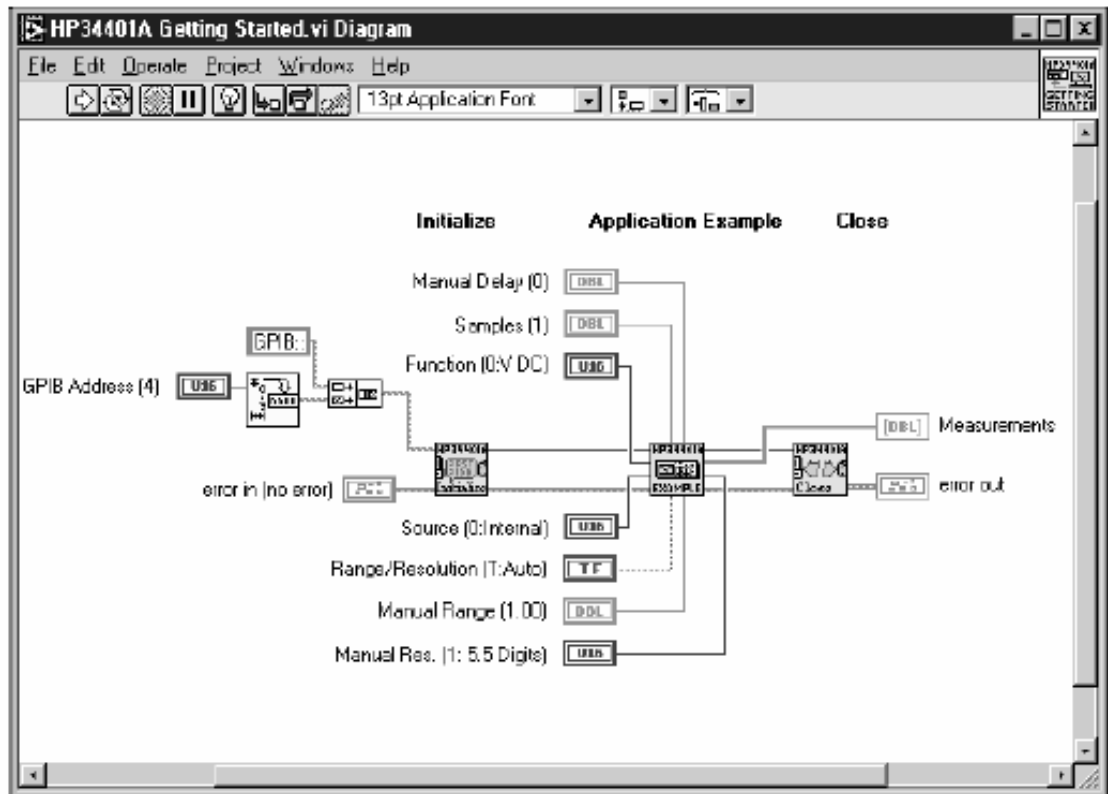
Run

6. Izvršimo (run) VI da bi mjerili naprimjer DC napone



Operating tool

7. Koristeći **Operating** alat, promijeniti **Function** ulaz i izvršiti VI ponovo da se mogu vidjeti i druga mjerenja



8. Analizirajmo blok dijagram. Vidimo da se **HP34401A Initialize** subVI prva poziva, zatim subVI za kontrolu instrumenta i na kraju sa završava sa **HP 34401A Close** subVI.

POGLAVLJE 14

POVEZIVANJE PC SA SIGNALIMA IZ MJERNOG OKRUŽAJA

Prije nego što kompjuterski bazirani VI instrument može mjeriti fizički signal iz vanjskog okružaja, senzor ili transdjuser mora konvertovati taj signal u električni signal bilo naponski ili strujni.

Plagljivi DAQ (data acquisition) modul često podrazumjeva kompletan DAQ sistem, mada je on ustvari samo jedna komponenta sistema.

Ovi uređaji se tipično spajaju direktno na interni bus PC računara putem plagljivog slota (ISA, EISA, PCI). Sa DAQ uređajima, hardware konvertuje ulazni signal u digitalni signal koji se šalje računaru. DAQ uređaj ne izračunava finalno mjerenje. Ovaj zadatak se prepušta softwareu koji je rezidentan na računaru.

Isti uređaj može izvršiti mnoštvo mjerenja jednostavno mjenjajući softwaresku aplikaciju koja čita podatke. Dakle, osim upravljanja, mjerenja i prikazivanja podataka, korisnikova aplikacija za računarski bazirani DAQ sistem takodjer ima ulogu firmwarea (tj. ugrađenog , u PROM ili EPROM softwarea) , potrebnog da se procesiraju podatci i izračunaju rezultati. Ovo najčešće egzistira kod instrumenata specijalne namjene, pri mjerenju i obradi odredjenih funkcija.

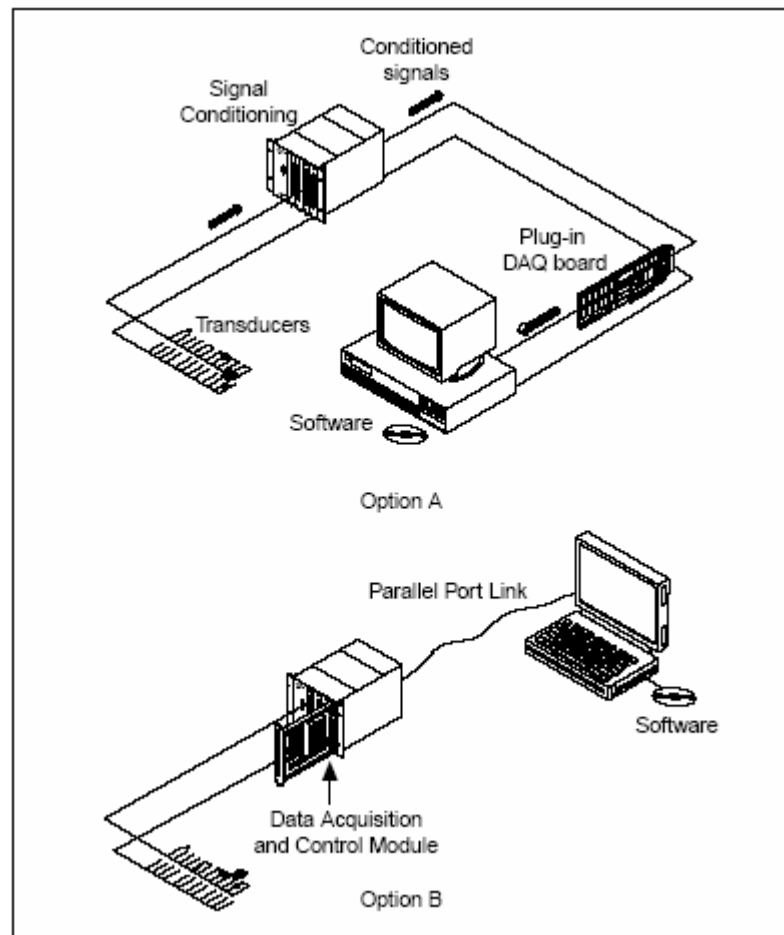
Kako kompjuteri komuniciraju sa DAQ uređajima

Za razliku od samostalnih mjernih instrumenata, nismo najčešće u mogućnosti da direktno priključimo signale na DAQ modul. U takvim slučajevima koristimo pomoćne module (accessories), da kondicioniramo signale, prije nego što ih plug-in DAQ modul konvertuje u digitalnu formu. Software kontrolira DAQ sistem prikupljajući sirove podatke, analizirajući podatke i prikazujući rezultate.

Naredna slika prikazuje dvije opcije DAQ sistema. U opciji A, plagljivi DAQ uređaj je rezidentan u računaru. U opciji B, DAQ uređaj je vanjski.

Sa vanjskom karticom, možemo graditi DAQ sisteme koristeći PC računare koji nemaju raspoloživih plug-in slotova, kao što je slučaj sa nekim laptopima (notebooks).

Kompjuter i DAQ modul komuniciraju putem različitih baseva, kao što je paralelni bus, serijski, ili Ethernet. Ovi sistemi se onda praktično pretvaraju u daljinske (remote), akvizicione sisteme.



Komponente DAQ sistema

Treća opcija, koja nije pokazana na prethodnoj slici, koristi PCMCIA (PC card), bas koji se nalazi na većini laptopa. PCMCIA DAQ uređaj se pluguje u računar, i signali se spajaju na ploču , kao i u sličaju opcije A. Ovo omogućava gradnju portabilnog, kompaktnog DAQ sistema.

Uloga softwarea

Računar prima sirove podatke. Software preuzima ove podatke i prikazuje ih u formi koju korisnik može razumjeti. Software manipulira podacima tako da se mogu pojaviti ili kao prikaz na grafu ili chartu, ili u izvještaju. Software također kontrolira DAQ sistem, informišući DAQ uređaj kada da prikuplja podatke, kao i sa kojih kanala da prikuplja podatke.

Tipično, DAQ software uključuje drajvere i aplikacioni software. Drajveri su specifični za svaki DAQ uređaj ili tip uređaja i uključuju set komandi koje uređaj prihvaća. Aplikacioni software (kao naprimjer i LabView), šalje komande na drajvere, da bi naprimjer prikupljali podatke sa termoelementa, i vratili mu te podatke, i nakon toga ih analizira i prikazuje prikupljene podatke.

LabView uključuje skup VI koje dozvoljavaju korisniku da konfigurira, prikuplja podatke sa, i šalje ih na DAQ uređaje. Ovo poštudjuje korisnika od toga da on sam mora pisati ove programe.

LabView DAQ VI šalju callove na NI-DAQ aplikacioni programski interfejs (API). NI-DAQ API sadrže alate i bazne funkcije koje interfejsiraju sa DAQ hardwareom.

Primjeri mjerenja sa DAQ modulima

Primjeri koje ćemo pokazati u ovom poglavlju pokazuju kako praviti mjerenja koristeći multifunkcionalni ulazno –izlazni tip DAQ uređaja (MIO – multifunctional I/O). Ovi primjeri su bazirani na korištenju DAQ VI-jeva.

Kako mjeriti istosmjerni (DC) napon

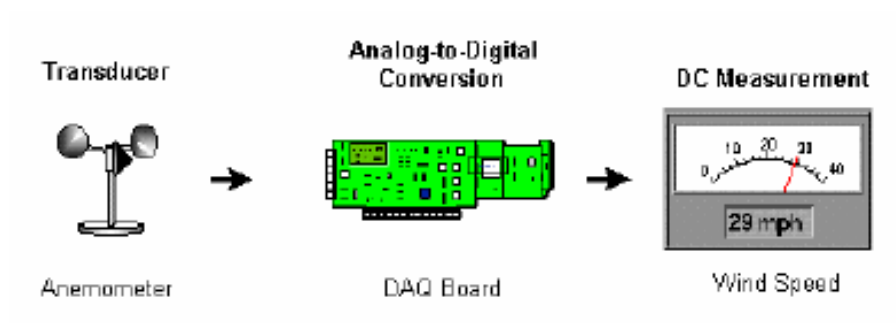
Uobičajeni tipovi DC signala koje treba mjeriti u inženejrskoj praksi su : signali sa transmitera pritiska , temperature, nivoa, sile, itd. (svi standardni 4-20 mA signali konvertovani u naponski signal 1- 5 V).

Da bi poboljšali tačnost većine mjerenja, koristićemo kondicioniranje signala. Ovo uključuje manipuliranje sa signalom koristeći i hardware i software. Uobičajene metode softwareskog kondicioniranja uključuju usrednjavanje, filtriranje, i linearizaciju.

Uobičajene hardwareske metode kondicioniranja uključuju pojačanje signala, kompenzaciju hladnog kraja (za termoelemente), pobudjivanje senzora (RTD mjerenje temeperature, ili napajanje Wheastonovog mosta sa strain gauge sensorima) , te filtriranje.

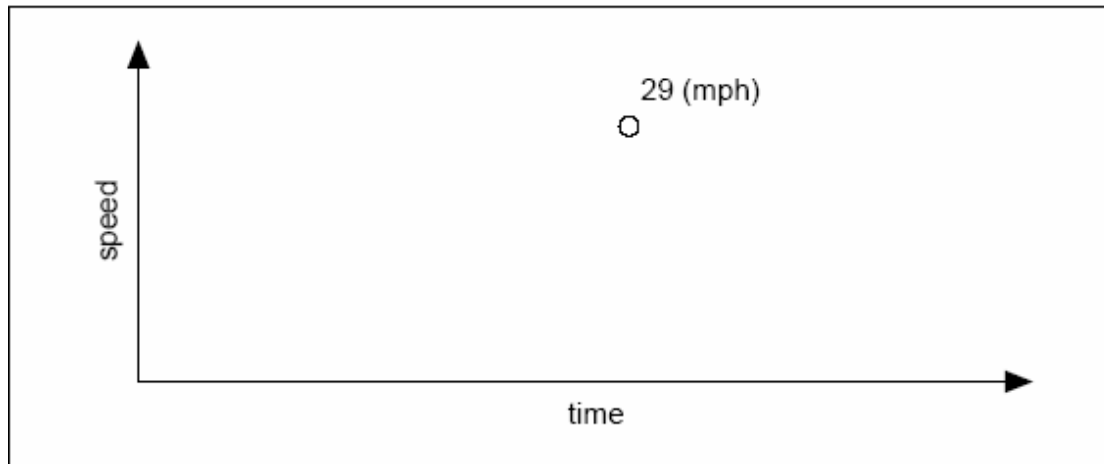
Primjer akvizicije jedne analogne vrijednosti

Naredna slika pokazuje jednostavnu akviziciju koristeći anemometar da mjeri brzinu vjetra:



Jednostavni akvizicioni sistem

U ovom primjeru, korisnik priključuje samo jednu mjernu tačku analognog mjerenja čiji valni oblik izgleda kao na narednoj slici :

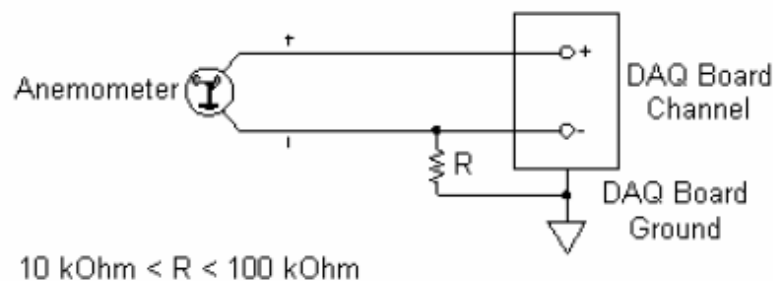


Brzina vjetra

Naredna slika pokazuje tipični dijagram ožičenja za anemometar sa izlaznim opsegom signala od 0 do 10 V, koji korespondira brzini vjetra od 0 do 200 mph. To znači da u softwearu, trebaćemo skalirati podatke po slijedećoj formuli :

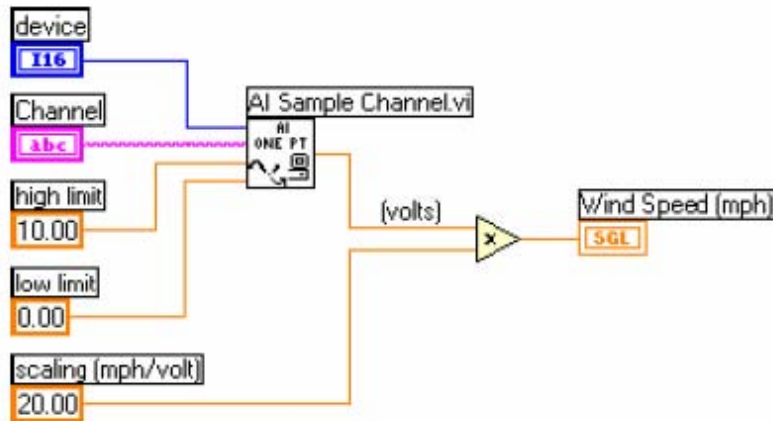
$$\text{Očitanje anemometra (V) } \times 20 \text{ (mph/V) } = \text{brzina vjetra (mph)}$$

Primjetimo korištenje otpora R, pošto anemometar obično nije uzemljeni izvor signala nego tkz. plivajući (floating signal source). Ako je pak anemometarski transmitter već uzemljen, korištenje otpora R bi rezultiralo u pogrešnom očitaniu (o ovome vidjeti kasnije).



Ožičenje anamometra

Naredna slika pokazuje blok dijagram koji je potreban za mjerenje brzine vjetra u LV. U ovom dijagramu, **device** je broj koji je doznačen za plug-in DAQ uređaj za vrijeme njenog konfigurisanja. **Channel** je analogni ulazni kanal na koji je anemometar ožičen. **High limit** i **low limit** vrijednosti pokazuju očekivani opseg signala. Ovaj opseg određuje iznos pojačanja kojeg će DAQ uređaj primjeniti. **AI Sample Channel** je DAQ subVI koji prikuplja jednu vrijednost analognog signala, u ovom slučaju sirovi napon. Vrijednost skaliranja od 20 mph/V je unjeta da se skalira ulazni napon u opsegu od 0 do 10 V u inženjerskoj jedinici opsega o do 200 mph , prema gornjoj formuli:



Mjerenje napona i skaliranje u brzinu vjetra

Mi možemo pojednostaviti ovaj blok dijagram koristeći DAQ imenovane kanale (**DAQ Named Channels –DNC**).

Naredna slika pokazuje LabView dijagram potreban da bi se mjerila brzina vjetra koristeći DAQ imenovani kanal . Ovo pojednostavljuje blok dijagram, pošto DAQ imenovani kanal ima informaciju o uređaju , kanalu, pojačanju i jednačini za skaliranje. Ponovno, **AI Sample Channel** prikuplja jednu vrijednost analognog signala, ali u ovom slučaju VI vraća odmah brzinu vjetra u inženjerskoj jedinici.

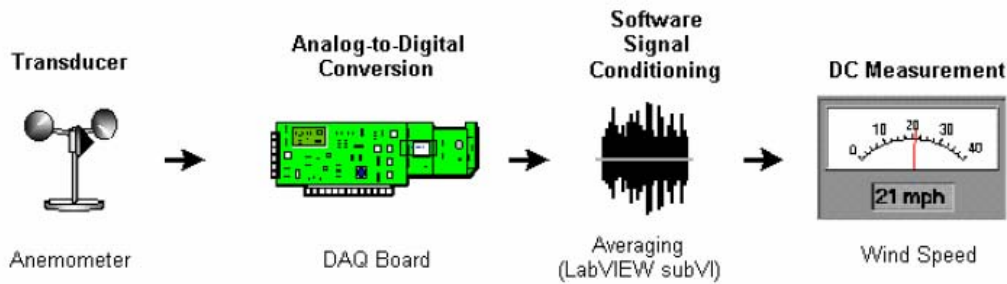


Mjerenje brzine vjetra koristeći DAQ imenovani kanal

Primjer usrednjavanja skaniranih vrijednosti

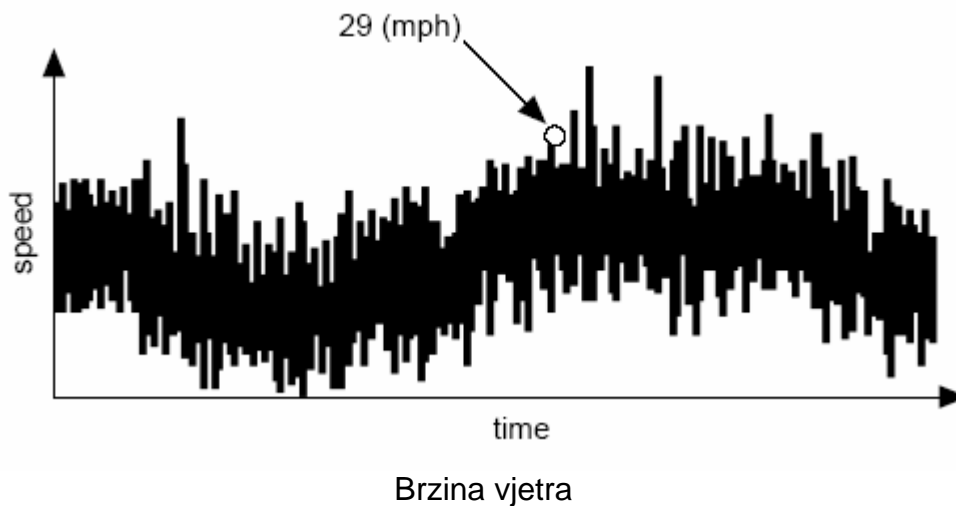
Jedan od najkorisnijih oblika, i jednostavnih za korištenje, kondicioniranja signala je usrednjavanje podataka u softwareu. Usrednjavanje može dati korisnija očitavanja u slučajevima kada se signal brzo mjenja , ili postoji šum u prenosu signala mjerenja.

Naredna slika pokazuje akvizicioni sistem za mjerenje brzine vjetra sa dodatkom softwareskog usrednjavanja.



Naredna slika pokazuje kako stvarna brzina vjetra može da izgleda u vremenu. Zbog udara vjetra, brzina vjetra izgleda vrlo šumovito.

Zato bi još bolji prikaz bio kada bi uzeli srednju brzinu na kraćem intervalu vremena.

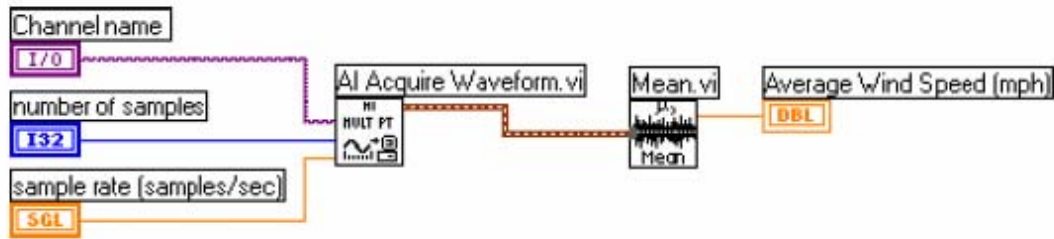


Pošto mi usrednjavamo u softwareu, hardwaresko ožičenje se neće mjenjati. Blok dijagram na slijedećoj slici pokazuje software za mjerenje srednje brzine vjetra ako koristimo imenovani DAQ kanal (DNC – DAQ named channel). Kako smo već rekli , DNC memorira informaciju o uređaju, kanalu , pojačanju i skaliranju.

Primjetimo da DAQ subVI u ovom primjeru se razlikuje od prethodnog slučaja akvizicije jedne tačke jer on sada akvizira valni oblik . Parametri **number of samples** (broj smplova) i **sample rate** (brzina sampliranja) definiraju valni oblik prikupljenih podataka.

Naprimjer, ako postavimo **number of samples** = 1000 i **sample rate** = 500 (smplova / sec) , biće potrebno 2 sekunde da se prikupi 1000 tačaka.

Valni oblik podataka od **AI Acquire Waveform** je ožičen na **Mean** subVI. Ovaj Vi će dati na svom izlazu usrednjenu brzinu vjetra u trajanju od 2 sekunde.



Prosječna brzina vjetra sa DNC

Jedan čest razlog usrednjavanja je i da se eliminiraju šumovi od 50 ili 60 Hz iz napona napajanja mreže. Pošto je šum napona napajanja sinusoidalni, usrednjavanje na jednom periodu će dati nultu srednju vrijednost.

Ako koristimo brzinu skeniranja koja je cjelobrojni multipl perioda šuma, šum iz napajanja će biti eliminiran. Jedan primjer koji radi i za šumove i 50 i 60 Hz je da skaniramo sa 300 skanova u sekundi, i onda uređujemo sa 30 tačaka. Primjetimo da je 300 cjelobrojni multipl i od 50 i 60. Jedan period za 50 Hz šum je $300/50 = 6$ tačaka. Jedan period za 60 Hz šum je $300/60 = 5$ tačaka. Usrednjavanje 30 tačaka je cjelobrojni multipl od obadva perioda, tako da možemo obezbjediti da usrednjavamo cijele periode.

Kako mjeriti naizmjenični (AC) napon

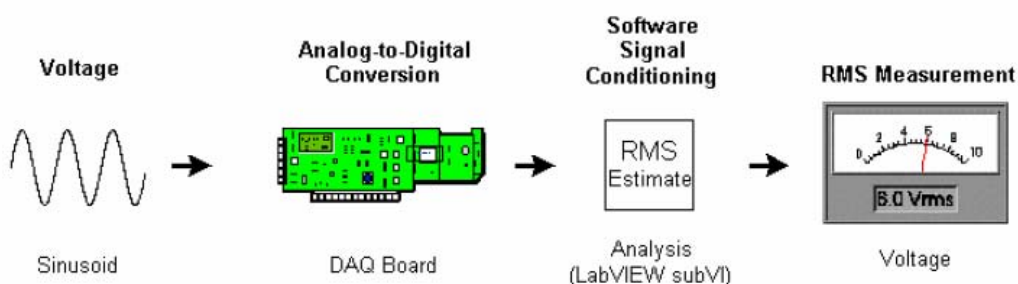
Ako želimo koristiti relacije koje vrijede za DC (istosmjerna) kola i kod mjerenja naizmjeničnog napona , onda je potrebn i izračunati njegovu efektivnu vrijednost V_{rms} (root mean square) tj. srednje kvadratnu vrijednost. Sa RMS , snaga koja je prenesena u električnom kolu na otporniku će biti :

$$P = \frac{V^2}{R}$$

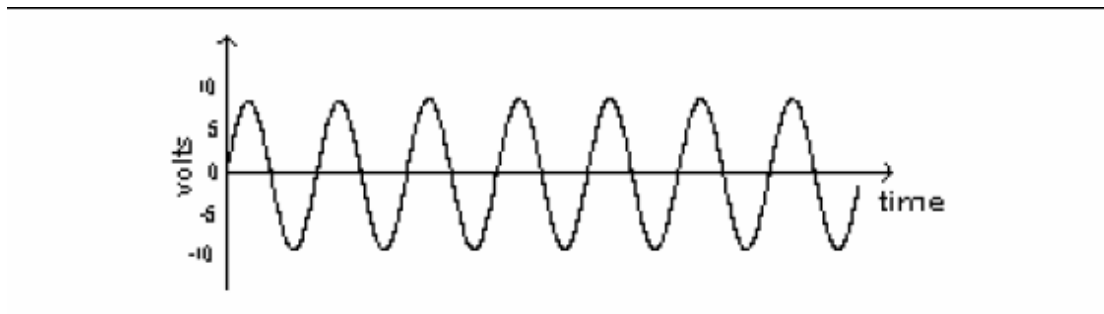
gdje je: $V = V_{rms}$

$$V_{rms} = \frac{V_{peak}}{\sqrt{2}}$$

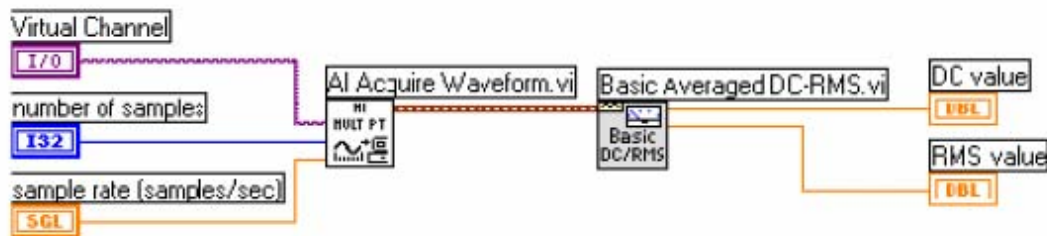
LabView omogućava lako mjerenje srednje kvadratne vrijednosti. Naredna slika pokazuje akvizicioni sistem za mjerenje V_{rms} :



Naredna slika pokazuje kako može izgledati stvarni valni oblik naizmjeničnog signala:

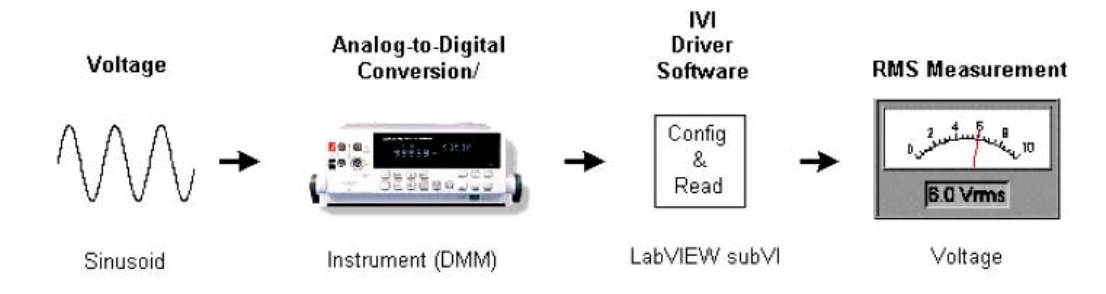


Blok dijagram na narednoj slici pokazuje software za mjerenje V_{rms} ako koristimo DNC .

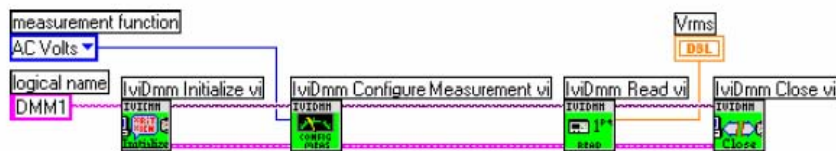


DAQ subVI **AI Acquire Waveform** prikuplja valni oblik signala. **Number of samples** i **sample rate** definiraju valni oblik. **Basic Averaged DC-RMS** Vi pružima valni oblik i procjenjuje RMS i DC komponente. Za sinusoidalni valni oblik simetričan oko nule, ova subVI će vratiti V_{rms} . Ako je sinusoidalni valni oblik ofsetiran od nule, **DC value** će vratiti vrijednost ovog DC offseta, a **RMS value** će vratiti V_{rms} kao da je valni oblik sinusoide bio simetričan oko nule.

Jedna prednost korištenja **Basic Averaged DC-RMS** VI je da može napraviti dobru procjenu sa najmanjom količinom podataka. U skladu sa Nyquistovim kriterijem, moramo prikupljati sa brzinom koja je najmanje dva puta veća od brzine signala koji se prikuplja, da bi dobili pouzdane frekventne podatke. Međutim, V_{rms} ne zavisi od frekvencije prikupljanja. On više zavisi od oblika signala. Tipično, da bi dobili dobar osjećaj o valnom obliku, moramo prikupljati sa brzinom koja je 5 do 10 puta veća od brzine valnog oblika. Prednost korištenja **Basic Averaged DC-RMS** VI je da ona daju dobru procjenu čak i kada akviziramo sa brzinom koja je samo tri puta veća od frekvencije valnog oblika. Isto mjerenje AC napona može se realizovati i sa instrumentom. Naredna slika pokazuje akvizicioni sistem za ovo mjerenje. U ovom slučaju, koristi se nezavistan mjerni uređaj.



Naredna slika pokazuje blok dijagram za mjerenje V_{rms} koristeći IVI klasu drajverskih VI. U ovom primjeru, instrument se prvo inicijalizira koristeći lokalno ime da se kreira sesija. Nakon toga, instrument se konfigurira za željeno mjerenje, u ovom slučaju AC Volts. Nakon konfigurisanja, uzimaju se očitavanja mjerenja. Na kraju sesija se zatvara.



Mjerenje V_{rms} koristeći instrument

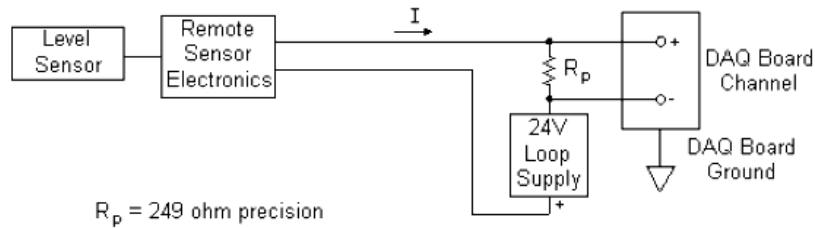
Kako mjeriti strujni signal

U narednom primjeru mi koristimo transducer sa standardnim strujnim signalom 4-20 mA da mjerimo nivo u posudi. Slika pokazuje akvizicioni sistem koji može biti korišten da ovo mjeri.



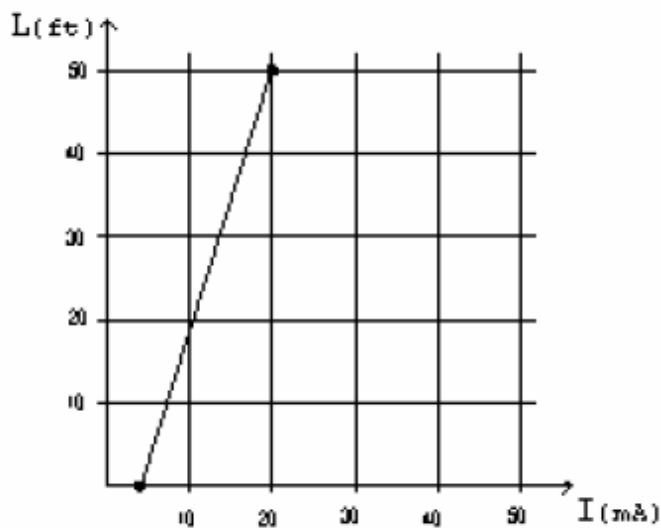
Akvizicioni sistem za mjerenje strujnog signala

Pošto MIO tip DAQ uređaja kod NI ne može direktno mjeriti strujni signal, napon se očitava sa preciznog otpornika koji je uključen u strujnu konturu. . Naredna slika ovo pokazuje:



Pošto je strujni signal 4-20 mA a R_p je 249 Ω , V je u opsegu od 0.996 V do 4.98 V, što je unutar opsega DAQ uređaja.

U ovom primjeru uzećemo da je fizikalna veličina nivoa u opsegu 0 do 50 feeta (stopa), pa se može opisati na slijedećem grafu , gdje je L nivo a I je strujni signal :



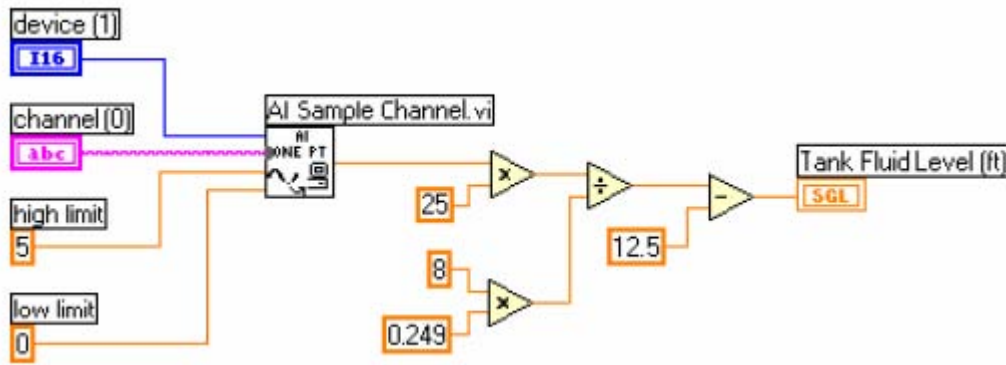
$$L = \frac{25}{8} I - \frac{25}{2}$$

Linearna relacija između nivoa u tanku i strujnog signala.

Koristeći Ohmov zakon, možemo uspostaviti slijedeću relaciju:

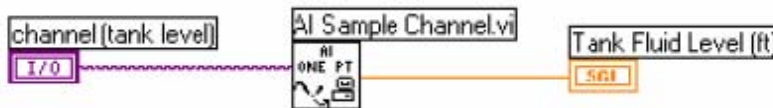
$$L = \frac{25 \cdot V}{8 \cdot 0.249} - \frac{25}{2}$$

Gornja jednačina može biti implementirana na blok dijagramu kao na slijedećoj slici:



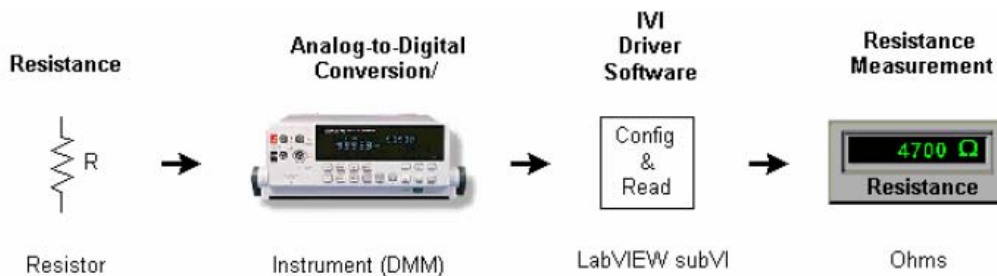
Mjerenje nivoa fluida sa DAQ modulom sa DNC

Alternativno, DNC konfigurisan u okviru DAQ Channel Wizarda koji se koristi kod konfigurisanja DAQ modula, može takodjer preuzeti ovo skaliranje. U tom slučaju , LabView dijagram se pojednostavljuje kao na slijedećoj slici:

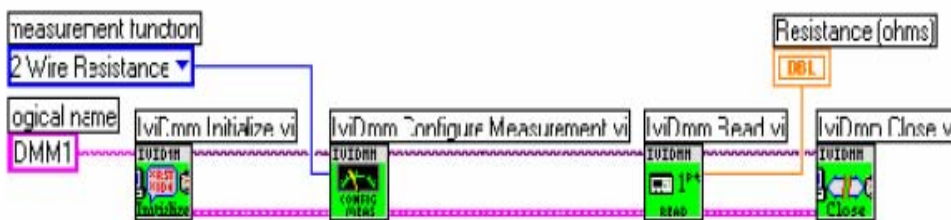


Kako mjeriti otpornost

Otpornost može biti mjerena koristeći jedan od DMM instrumenata (kao naprimjer NI tip NI 4050 ili NI 4060) , kako je prikazano na slijedećoj slici:



Naredna slika pokazuje LabView dijagram za mjerenje otpornosti koristeći IVI klasu drajverskih VI. Primjetimo sličnost sa mjerenjem V_{rms} . Razlika je da je funkcija mjerenja sada promjenjena na mjerenje otpornosti dvožično.

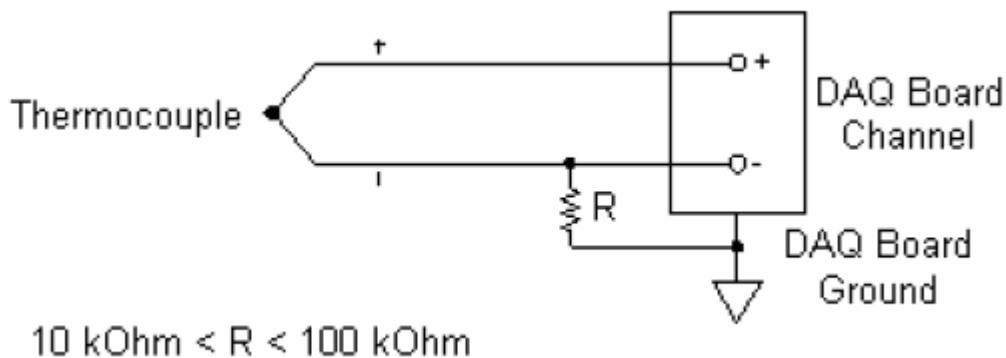


Mjerenje temperature

Mjerenje temperature sa termoelementom se realizuje prema slijedećoj slici:

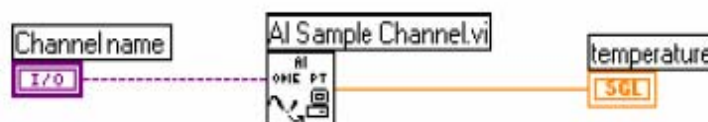


Naredna slika pokazuje tipični dijagram ožičenja za termoelement. Primjetimo da otpor R se koristi samo u slučaju kada termoelement nije umašen u bilo kojoj drugoj tački. Ako bi naprimjer vrh termoelementa bio umašen (zbog bržeg odziva), korištenje R bi formiralo konturu umašenja, što bi dovelo do pogrešnih očitavanja.



Ožičenje termoelementa

Naredna slika pokazuje blok dijagram koji je potreban za mjerenje temperature ako koristimo DNC (DAQ imenovani kanal). U ovom slučaju, ponovo DNC preuzima sve uključivo pojačanje, linearizaciju, i kompenzaciju hladnog kraja.



Mjerenje temperature koristeći DNC

Ako ne želimo da koristimo DNC da mjerimo temperature, moramo napisati VI koja će odrediti pojačanje potrebno za mjerni opseg temperature, očitava napon sa termoelementa, očitava napon kompenzacije hladnog kraja, i konvertuje sve ove informacije u temperaturu.

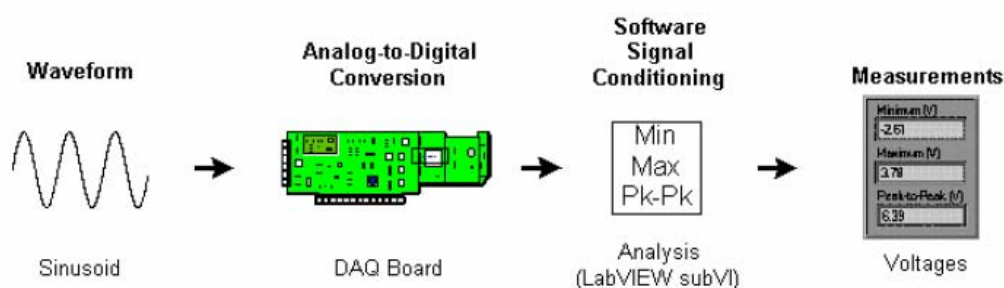
Pogledati i VI pod nazivom **Single Point Thermocouple Measurement** VI lociranu u **examples\daq\ solution\transduc.llb** da bi vidjeli kako se mjeri temperatura koristeći termoelement.

Primjeri osciloskopskih mjerenja

U ovom poglavlju ćemo razmatrati kako da načinimo mjerenja koja su tipična kod osciloskopa. Primjeri pokazuju kako koristiti MIO tip DAQ uređaja ili instrumenta da se realizuju ova mjerenja.

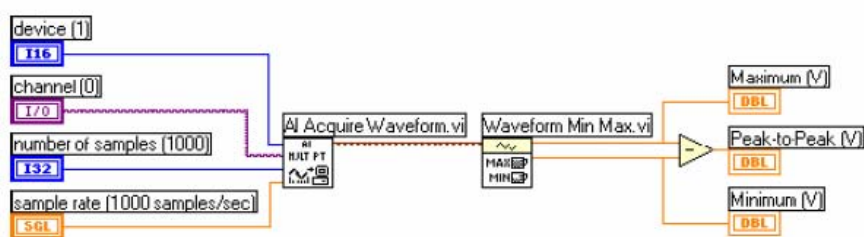
Kako mjeriti maximum, minimum i peak-to-peak napone

Ovaj primjer pretpostavlja da imamo neki tip signala koji se mjenja u vremenu. Naredna slika pokazuje kako bi mjerni sistem mogao izgledati:



Akvizicioni sistem za mjerenje max, min i peak-to-peak vrijednosti

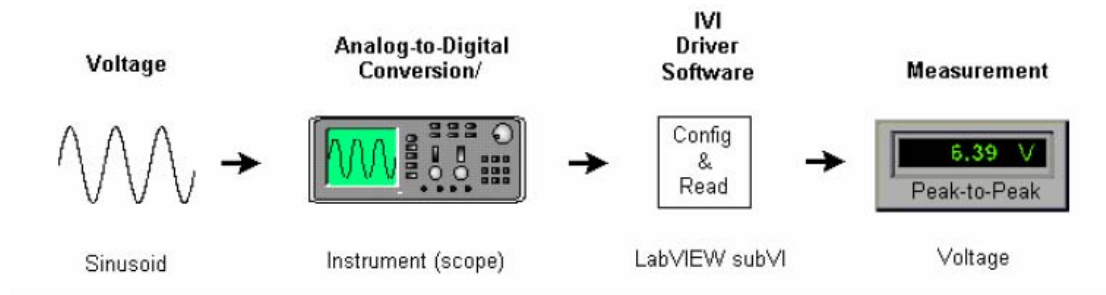
Za ovo mjerenje, signal je repetitivan, ali ne mora biti u redosljedu da očitava maximum, minimum a zatim peak-to-peak vrijednost. Naredna slika pokazuje LabView blok dijagram za ova mjerenja:



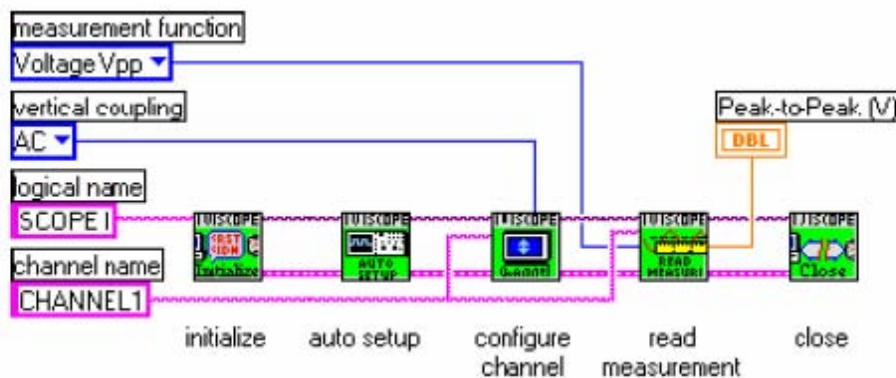
AI Acquire Waveform VI se poziva da skanira podatke sa jednog kanala DAQ uređaja. Prikupljeni valni oblik se prenosi na **Waveform Min Max** VI, koji vraća minimalnu i maksimalnu vrijednost valnog oblika.

Razlika ove dvije vrijednosti je peak-to-peak vrijednost napona.

Peak-to-peak mjerenje napona može također biti realizovano pomoću mjernog instrumenta. Slika koja slijedi pokazuje akvizicioni sistem za ovo mjerenje. U ovom slučaju, pokazan je osciloskop kao samostalni uređaj (stand alone). Alternativno, ovo može biti i instrumentalna ploča koja se plagije direktno u PC.



Slika koja slijedi pokazuje LabView dijagram koji mjeri peak-to-peak napon koristeći IVI klasu drajverskih VI.



Ovaj primjer koristi slijedeće VI u redoslijedu :

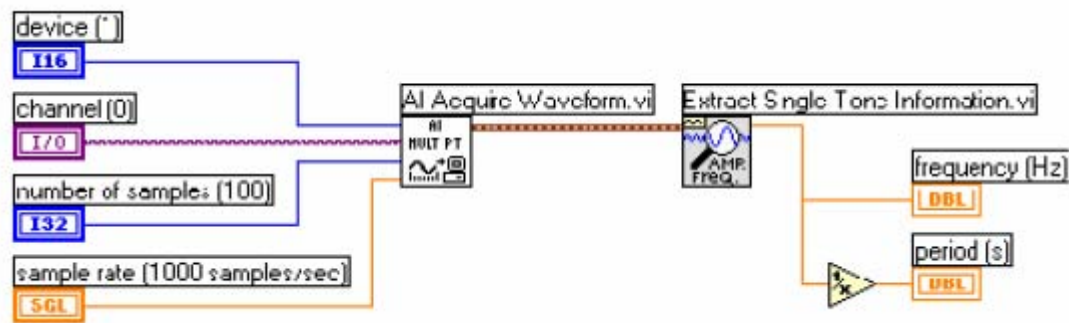
1. **IviScope Initialize** VI inicijalizira osciloskope i kreira sesiju.
2. **IviScope Auto Setup [AS]** VI osjeća ulazni signal i automatski konfiguriše mnoge setinge instrumenta.
3. **IviScope Configure Channel** VI postavlja kuplovanje na AC. Ovo eliminira DC komponentu iz signala.
4. **IviScope Read Waveform Measurement [WM]** VI očitava peak-to-peak napon ,
5. **IviScope Close** VI zatvara sesiju i oslobadja resurse.

Kako mjeriti frekvenciju i period repetitivnog signala

Primjer mjerenja frekvencije i perioda

Za ovaj primjer, treba da imamo repetitivni signal. Naš mjerni sistem je sličan onom za mjerenje minimuma, maximuma i peak-to-peak vrijednosti samo što je na instrumentu izabrano mjerenje frekvencije. Da bi dobili razumne rezultate moramo voditi računa o Nyquistovom kriteriju, To znači da ako želimo da mjerimo signal frekvencije od 100 Hz, treba nam brzina sampliranja od najmanje 200 S/s. U praksi, očekuje se da brzina sampliranja bude 5 do 10 puta veća od očekivane frekvencije koja se mjeri.

Naredna slika pokazuje blok dijagram koji će realizovati ovo mjerenje. Jedanput kada se frekvencija odredi, onda se period signala izračuna jednostavno kao inverzna vrijednost frekvencije.

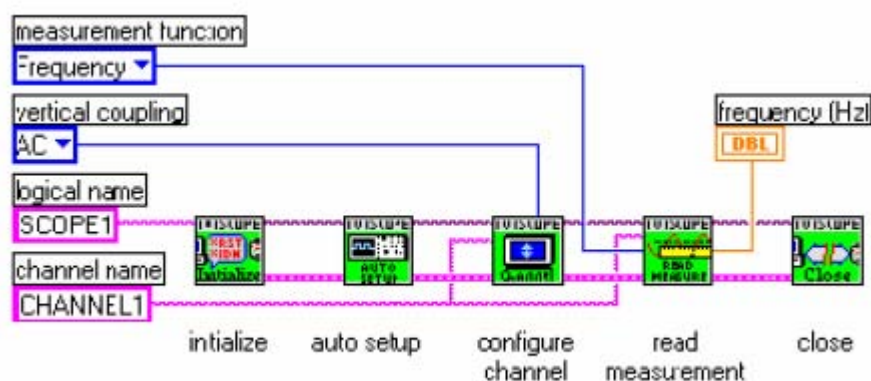


Osim brzine sampliranja, treba da odredimo i broj samplova koje treba prikupiti. U opštem slučaju, što više to bolje, ali imaju dvije stvari koje treba uzeti u obzir. Prvo, minimalno tri ciklusa moraju biti akvizirana. To znači da u slučaju 100 Hz, ako je brzina sampliranja 500 S/s , potrebno je da prikupimo najmanje 15 tačaka. Ovo je zbog toga što sampliramo sa oko 5 puta većom brzinom od frekvencije signala. To znači da da sampliramo oko 5 tačaka po ciklusu signala. Dakle, pošto trebamo 3 ciklusa, treba nam $3 \times 5 = 15$ samplova.

Drugo, broj tačaka koje prikupljamo određuje broj frekventnih “korpi” (“bins”) u koje podatci upadaju. Sa više korpi , frekvencija koju mjerimo može se uklopiti u jednu korpu umjesto u nekoliko. Veličina svake korpe je brzina samplovanja podijeljena sa brojem prikupljenih tačaka. Ako sampliramo sa sa 500 S/s , i prikupimo 100 tačaka, imaćemo korpe u intervalu od 5 Hz.

VI **Extract Single Tone** koja se koristi u ovom primjeru, koristi podatke iz tri dominantne korpe da odredi frekvenciju. Pravilo palca (ad hoc) , je da sampliramo 5 do 10 puta brže od frekvencije signala, i da prikupimo 10 ili više ciklusa.

Frekvencija se takodjer može mjeriti i pomoću instrumenta. Naredna slika pokazuje blok dijagram za ovo mjerenje.



Primjer mjerenja frekvencije i perioda sa filtriranjem

Frekventne komponente ispod Nyquistove frekvencije će se pojaviti tamo gdje i treba na njihovim vrijednostima. Frekventne komponente iznad Nyquistove

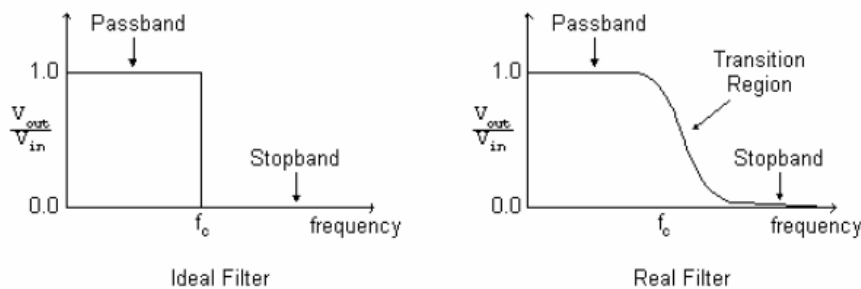
frekvencije pojaviće se aliasirane između 0 i Nyquistove frekvencije. Aliasirana komponenta je apsolutna vrijednost razlike između aktuelne komponente i najbližeg cjelobrojnog multipla brzine sampliranja.

Na primjer, ako imamo signal sa komponentom na 800 Hz, i sampliramo sa 500 S/s, ta komponenta će se pojaviti aliasirana na :

$$|800 - (2 \cdot 500)| = 200\text{Hz}$$

Jedan način da eliminiramo aliasirane komponente je da koristimo analogni hardware filter prije digitalizacije i analize za sadržaj frekvencije. Ako želimo da ovo filtriranje realizujemo u softwearu, moramo prvo samplirati pri brzini koja je dovoljno velika da korektno predstavi najveću frekventnu komponentu koja je sadržana u signalu. U ovom primjeru, sa najvećom komponentom od 800 Hz, minimalna brzina sampliranja treba biti 1600 Hz. U praksi, treba koristiti brzinu sampliranja od 5 do 10 puta veću od 800 Hz.

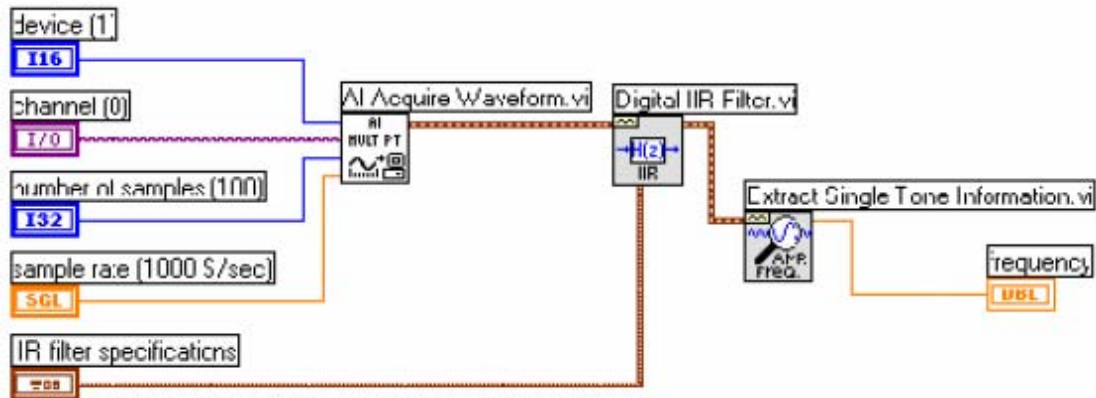
Predpostavimo sada da je frekvencija koju pokušavamo da mjerimo je oko 100 Hz. Možemo koristiti niskopropusni Butterworth-ov filter sa frekvencijom odsjecanja (cutoff frequency – f_c) setovanom na 250 Hz. Ovo će filtrirati frekvencije iznad 250 Hz i propustiti frekvencije ispod 250 Hz. Naredna slika pokazuje ovaj nisko propusni filter.



Realni filter je ono što možemo realno postići sa Butterworthovim filterom. Pojas propuštanja je gdje je V_{out}/V_{in} blisko 1.

Stop opseg (band) je gdje je V_{out}/V_{in} blisko 0. Između je tranzitni region , gdje su frekvencije postepeno prigušuju.

Naredna slika pokazuje blok dijagram filtera prije mjerenja frekvencije. Primjetimo **Digital IIR Filter** VI i IIR filterski specifikaciju.



Mjerenje frekvencije nakon filtriranja

Slika koja slijedi pokazuje kako izgledaju specifikacije IIR filtera na prednjem panelu. Ovdje ćemo i izabrati dizajn parametre za naš filter.

IIR filter specifications

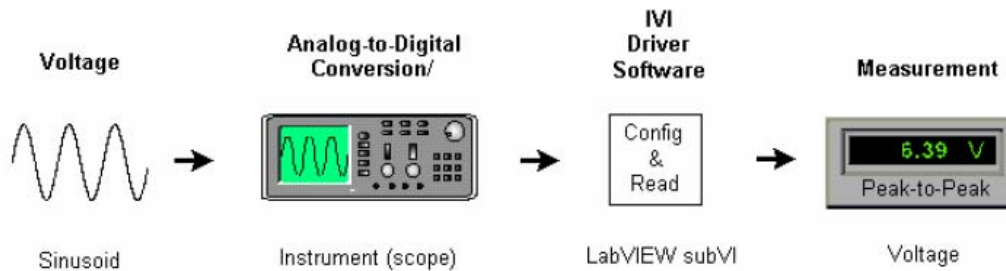
The screenshot shows the 'IIR filter specifications' dialog box with the following settings:

- Filter topology: Butterworth
- Order: 5
- Filter type: Lowpass
- Lower cut-off frequency (Hz): 250.00
- Upper cut-off frequency (Hz): 16.34k
- Passband ripple (dB peak-peak): 25.00m
- Stopband attenuation (dB): 92.00

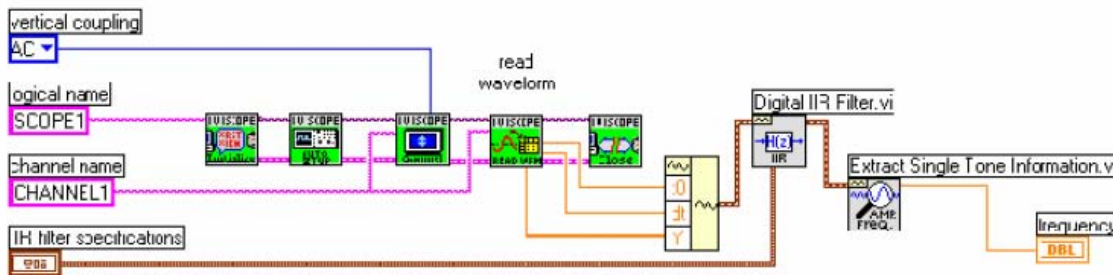
Prednji panel specifikacija IIR filtera

U ovom primjeru, niskopropusni Butterworth filter petog reda se koristi sa frekvencijom odsjecanja (cutoff frequency) , od 250 Hz. Ovaj red određuje kako će biti strm nagib u regionu tranzicije. Veći red daje strmije nagibe. Medjutim , niži red smanjuje i vrijeme računanja i grešku. U ovom našem primjeru, **Upper cut-off frequency** (frekvencija gornjeg odsjecanja) , **Passband ripple** (ripl u pojasu propuštanja) i **Stopband attenuation** (gušenje u nepropusnom opsegu) , kao ulazi su ignorisani.

Frekvencija sa softwareskim filtriranjem se može mjeriti koristeći i instrument. Instrumentalni setup je kao na slijedećoj slici :



Naredna slika pokazuje blok dijagram za ovo mjerenje. Primjetimo da se koristi **IVIScope Read Waveform** VI da bi se očitao array podataka. Izlazi iz ove subVI su ugradjeni u valni oblik podataka. Korištenje **Digital IIR Filter** VI i **Extract Single Tone Information** VI se koriste prema ranijoj diskusiji da bi se odredila frekvencija.



Mjerenje frekvencije nakon filtriranja koristeći instrument

Kako naći DAQ VI u LabView

Paleta **Functions >> Data Acquisition** sadrži 6 subpaleta, koje sadrže različite klase DAQ VI. Ove DAQ VI su klasifikovane kako slijedi:

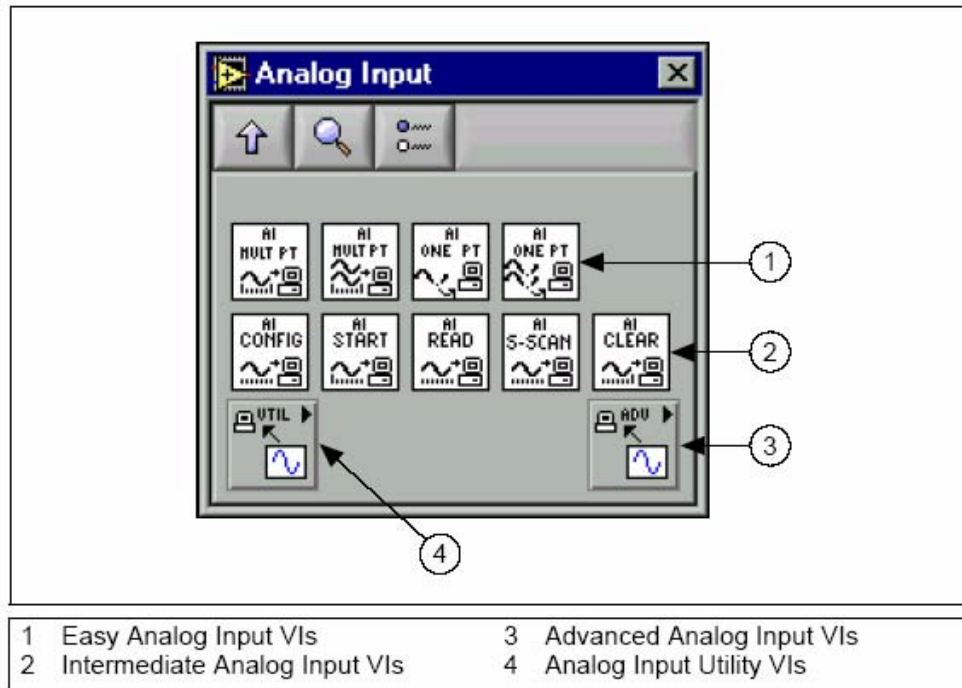
- VI Analognih ulaza
- VI Analognih izlaza
- VI Digitalni I/O
- VI Brojača (Counters)
- VI kalibracija i konfiguracija
- VI za kondicioniranje signala

Organizacija DAQ VI

Većina DAQ VI subpaleta organizuju VI-jeve u različite nivoe u skladu sa njihovim funkcionalnostima. Tako možemo naći slijedeća 4 nivoa DAQ Vi-jeva unutar sublapeta:

- Easy Vi (lagane VI)
- Intermediate Vi (srednje VI)
- Utility Vi (pomoćne VI)
- Advanced Vi (napredne VI)

Naredna slika pokazuje primjer DAQ subpalette koja sadrži sve ove nivoe :



Organizacija palete VI Analognih ulaza

Easy (lagane) VI

Easy VI izvršavaju jednostavne DAQ operacije i tipično su smještene u prvom redu palete DAQ VI. Možemo izvršavati ove VI sa prednjeg panela ili ih koristiti kao subVI u drugim aplikacijama.

Potrebna je samo jedna Easy VI da bi se izvršila osnovna DAQ operacija. Za razliku od srednjih (intermediate) i naprednih (advanced) Vi-jeva, lagane VI automatski upozoravaju korisnika o greškama sa dijalog boksom, što omogućava korisniku da zaustavi izvršenje VI ili ignorisanje greške.

Lagane VI su obično sastoje od srednjih (intermediate) VI, koje su opet sa svoje strane komponovane od advanced VI. Easy VI obezbjedjuju bazni interfejs samo sa najčešće korištenim ulazima i izlazima. Za kompleksnije primjene, treba koristiti intermediate i advanced VI, da bi se ostvarilo više funkcionalnosti i bolja performansa.

Intermediate (srednje) VI

Srednje VI imaju više hardwareske funkcionalnosti i efikasnosti u razvoju aplikacija nego Easy VI. Srednje VI sadrže grupe Advanced VI, ali koriste manje parametara i nemaju neke od naprednijih mogućnosti.

Srednje VI daju korisniku više kontrole pri rukovanju sa greškama nego Easy VI. Sa svakom VI, moguće je provjeravati greške i prenositi klaster greške na druge VI.

Opaska : Većina primjera koji će biti navedeni u nastavku biće bazirana na Intermediate VI.

Utility (Pomoćne) Vi

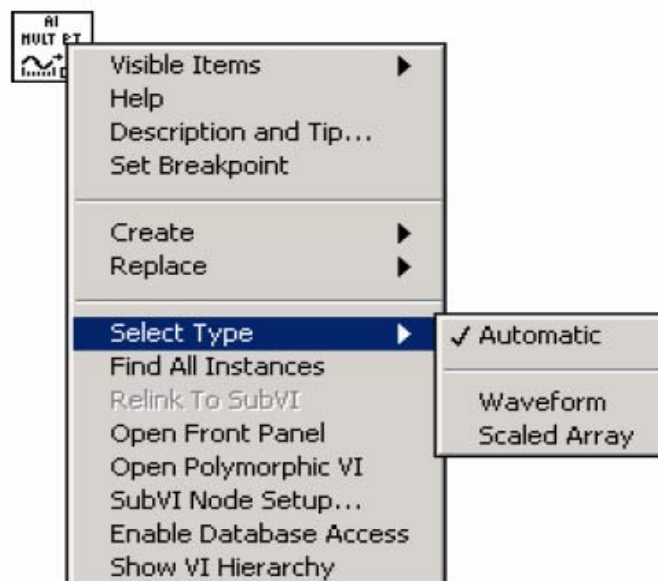
Pomoćne VI, koje se mogu naći u mnogim subpaletama DAQ paleta, su također Vi srednjeg (intermediate) nivoa , i prema tome imaju više hardwareske funkcionalnosti i efikasnosti u razvoju aplikacija nego Easy VI.

Advanced (Napredne) VI

Napredne VI su najniži nivo interfjsa ka DAQ drajverima. Vrlo malo aplikacija zahtjeva korištenje naprednih VI. Napredne VI daju na izlazu najveći obim statusne informacije od DAQ drajvera. Treba koristiti napredne VI, kada Easy VI ili srednje VI nemaju ulaze koji su potrebni da bi se kontrolisala neka nestandardna DAQ funkcija.

Polimorfične DAQ VI

Neke od DAQ VI su polimorfične. Ovo znači da one prihvataju ili vraćaju podatke u različitim formatima. Na primjer, Easy Analog Input VI može vratiti podatke bilo u obliku valnog oblika ili kao varijablu polja (array) sa skaliranim vrijednostima. Po defaultu, Polymorphic Analog Input VI vraćaju podatke kao valni oblik. Da bi se promjenio tip podatka koji se šalje iz VI nakon izvršenja, treba kliknuti desnim tasterom na VI ikonu i izabrati **Select Type** iz menija koji će iskočiti, kao što je to pokazano na slijedećoj slici:



Easy VI analognog izlaza može prihvatiti podatke bilo u obliku valnog oblika (waveform), ili kao polje skaliranih vrijednosti. Polimorfna VI analognog izlaza se adaptira na tip podatka koji je spojen na nju.

Konvencija za VI parametre

U svakoj LabView DAQ VI prednjem panelu ili u prozoru **Context Help** , način kako se pojavljuje kontrolne ili indikatorske labele, označavaju važnost tog parametra. Ako su prikazani masnim slovima (**bold**) oni se zahtjevaju da budu ožičeni na blok dijagramu da bi aplikacija mogla da se izvršava. Ako se imena parametara pojavljuju u normalnom tekstu (plain) , ona su opciona i nije nužno ih ožičiti u blok dijagramu da bi se program izvršavao.

Korisnik će vrlo rijetko trebati da koristi parametre koji su opcioni (plain) i još se pojavljuju u uglastim zagradama [] .

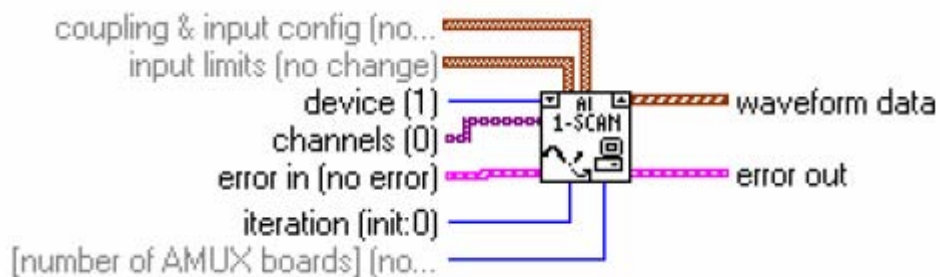
Default vrijednosti se pojavljuju u malim zagradama () sa desne strane imena parametara.

Naredna slika ilustrira ovu **Context Help** konvenciju parametara za VI **AI Read One Scan**. Kao što se vidi sa slike, moramo ožičiti slijedeće ulazne parametre : **device** (ako ne koristimo ime kanala tj. DNC) , **channels**, **error in**, i **iteration** , te slijedeće izlazne parametre :

waveform data i **error out**.

Da bi prenijeli informaciju o grešci sa jedne VI na drugu, trebamo spojiti **error out** cluster tekućeg VI na **error in** klaster slijedećeg VI.

Ulazni parametri : **coupling & input config**, **input limits**, **output units** i **number of AMUX boards** su opcioni.



Konvencija o default i tekućim vrijednostima

Da bi koristili DAQ VI, moramo znati razliku između default ulaza, default setinga, i tekućeg ulaza. Default ulaz je default vrijednost kontrolnog elementa na prednjem panelu. Ako ne ožičimo ulaz na terminal VI, tada default ulaz za taj terminal će se poslati na drajver. (vrijednost u () zagradama u **Context help** prozoru). Default seting je default vrijednost parametra koja je registrirana u drajveru.

Tekuća (current) vrijednost setinga je vrijednost kontrolnog parametra u svakom trenutku vremena. Tekuća vrijednost kontrolnog parametra je njegova default vrijednost sve dok se ne promjeni vrijednost tog kontrolnog parametra.

U mnogim slučajevima kontrolni ulaz defaultira na neku unaprijed definisanu vrijednost (najčešće 0).

Naprimjer, default ulaz za parametar može biti : **do not change current setting** (ne mjenjati tekući seting), a tekući seting može biti : **no AMUX-64T boards** (nema AMUX64T ploča) .

Ako korisnik promjeni vrijednost takvog parametra, nova vrijednost postaje tekući seting.

Kontrola valnog oblika (waveform)

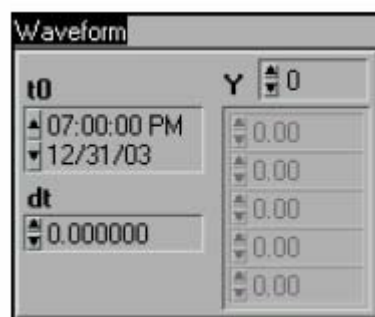
LabView predstavlja valni oblik sa kontrolnim parametrom po defaultu. Jednodimanzionalna varijabla polja (1D array) od valnih oblika , se koristi da predstavi višestruke valne oblike.

VI-jevi, funkcije, i objekti na prednjoj ploči koji se koriste za gradnju Vi-jeva za akviziciju, analizu i displej analognih mjerenja, prihvataju i vraćaju valne oblike (default) po defaultu.

Kontrolni element valnih oblika sadrži podatke pridružene sa jednostrukim valnim oblikom, uključujući i vrijednosti podataka i informacije o vremenu.

Kontrolni element valnog oblika prenosi komponente valnih oblika na VI funkcije koje koristimo da gradimo mjerne aplikacije. Ovaj element se može kastomizirati da ima mnoge pojavne oblike. DAQ VI koriste dvije najčešće oblike pojavljivanja, jednu da odrazi valne oblike sa po jednom tačkom akvizicije, i druge da odraze valne oblike sa više tačaka.

Naredna slika pokazuje kako se pojavljuju ovi kontrolni elementi valnih oblika:



Komponente valnog oblika

Kontrolni element valnog oblika je specijalni klaster (grozd) komponenata koje uključuju samo informacije uniformno samplirane od valnog oblika u vremenskom domenu.

Vrijeme starta (početka) to

Startno vrijeme je vrijeme početka prve tačke valnog oblika. Koristiti ovo vrijeme da se sinhronizuju zapisi kod slučaja gradnje muti kanalnog grafa , ili da se odredi koliko je kašnjenje izmedju pojedinih valnih oblika. Ovo vrijeme se ne koristi kod VI analognih izlaza.

Delta t (dt)

Delta t je vrijeme izmedju sukcesivnih vremenskih tačaka u valnom obliku. Ova vrijednost se ne koristi kod VI analognih izlaza.

Podatci valnog oblika (waveform data) – Y

Podatci valnog oblika su 1-no dimenzionalna varijabla polja (1D array) , sa podacima u obliku brojeva dvostruke preciznosti da bi predstavili valni oblik. Općenito, broj vrijednosti podataka u polju direktno korespondira sa brojem uzoraka koji je uzet sa akvizicionog uređaja.

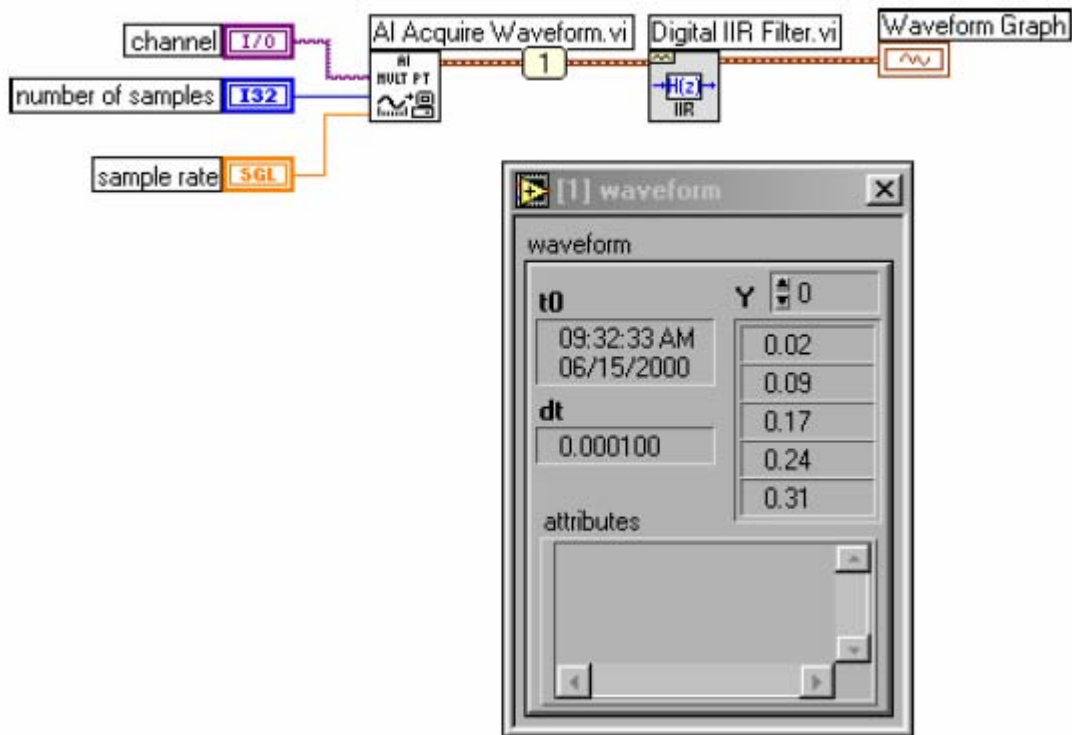
Atributi

Komponenta atributa može biti korištena da bi sadržavala informaciju o valnom obliku. Korisnik može postaviti attribute sa funkcijom : **Set Waveform Attribute** i očitavati attribute sa funkcijom : **Get Waveform Attribute**.

Korištenje kontrolnog elementa valnog oblika

Postoji niz LabView VI-jeva i primitiva koje prihvataju, rade na i vraćaju valne oblike (waveforms). Nadalje, korisnik može povezati žice kontrole valnog oblika direktno na mnoge LabView kontrolne elemente, uključujući grafove, chartove, numeričke, i numeričke array kontrolne elemente.

Na narednoj slici je pokazan blok dijagram koji prikuplja valni oblik sa kanala na uređaju akvizicionog modula, i šalje ga kroz Butterworthov filter i iscrtava rezultirajući valni oblik na grafu.



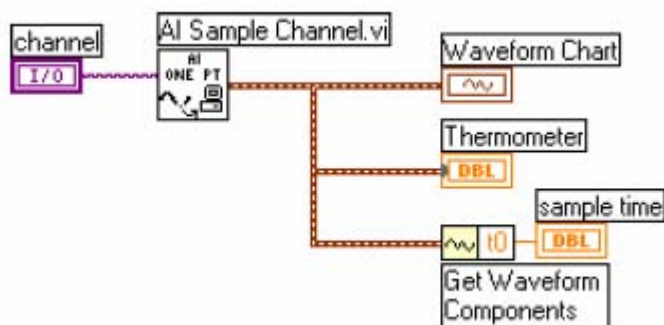
Korištenje tipa podatka valnog oblika

AI Acquire Waveform VI uzima određeni broj uzoraka sa kanala sa specificiranom brzinom uzorkovanja i VI vraća valni oblik. Test proba prikazuje komponente valnog oblika : vrijeme kada je počela akvizicija (**t0**), vrijeme između dvije sukcesivne tačke (**dt**), i podatke akviziranog valnog oblika (**Y**). FIR filter valnog oblika VI , prihvata polje valnih oblika i automatski filtrira podatke (**Y**) svakog valnog oblika. Nakon toga graf iscrtava i prikazuje valni oblik.

Može se koristiti i kontrolni element valnog oblika sa akvizicijom pojedinačnih tačaka kao što je pokazano na narednoj slici. VI **AI Sample Channel** uzima pojedinačne sample sa kanala i vraća valne oblike sa po jednom tačkom. Valni oblik će sadržavati vrijednost koja je očitana sa kanala i vrijeme kada je kanal očitano.

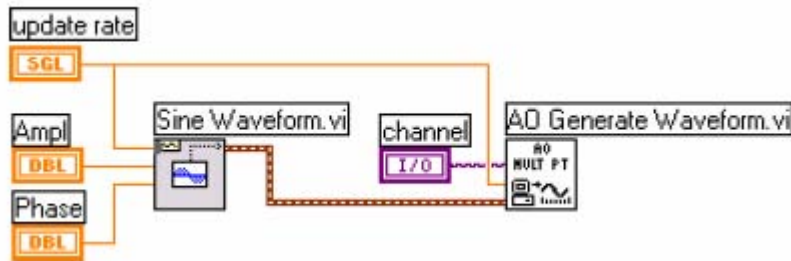
Chart i kontrolni element temperature prihvataju valni oblik i prikazuju podatke.

Funkcija **Get Waveform Components** se koristi da izvuče startno vrijeme iz valnog oblika.



Primjer akvizicije sa pojedinačnim tačkama

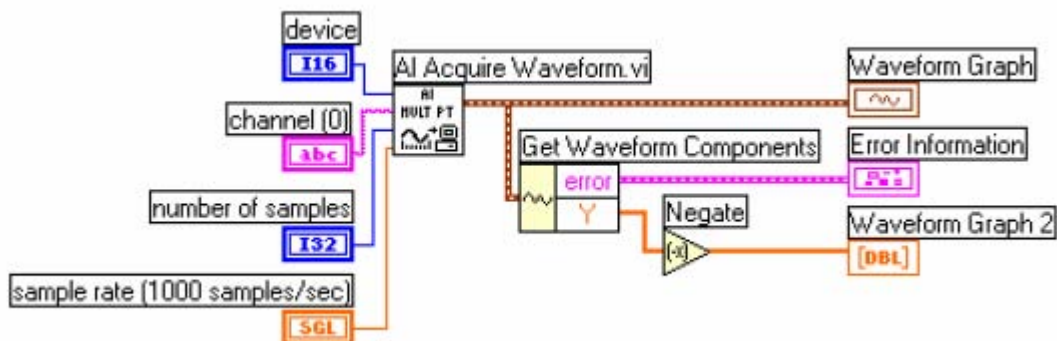
Kontrolni element valnog oblika se može koristiti sa analognim izlazom kao što je pokazan na narednoj slici. VI **Sine Waveform** generira sinusni valni oblik. VI **AO Generate Waveform** šalje valni oblik na uređaj.



Korištenje kontrolnog elementa valnog oblika sa analognim izlazom

Ekstrakcija komponenti valnog oblika

Treba koristiti Funkciju: **Get Waveform Components**, da se izvade komponente valnog oblika kojeg generiramo. VI na narednoj slici koristi ovu funkciju da izvadi podatke valnog oblika. Funkcija negiranja (Negate) negira podatke valnog oblika i iscrtava ih u grafu.

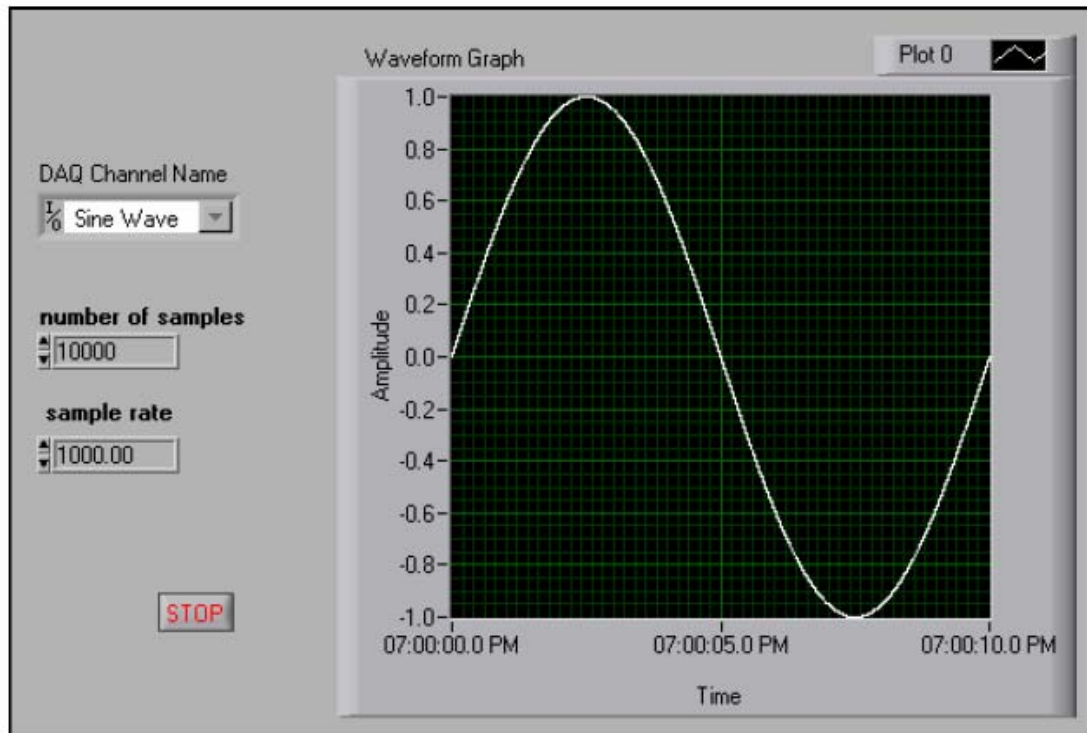


Podatci valnog oblika na prednjem panelu

Na prednjem panelu, treba koristiti **Waveform** kontrolni element, koji je raspoloživ u paleti **Controls>>I/O**, ili **waveform Graph**, koji je raspoloživ na **Controls>>Graph** paleti, da se prikažu podatci valnog oblika.

Koristiti Waveform kontrolni element da manipuliramo sa **t0**, **dt** i **Y** komponentama valnog oblika ili ako hoćemo da ih prikažemo preko indikatora. Kada ožičimo valni oblik na graf, **t0** komponenta je inicijalna vrijednost na x osi. Broj skanova koji je prikupljen i komponenta dt određuju naredne vrijednosti na x osi.

Vi na narednoj slici kontinualno prikuplja 10.000 skanova sa brzinom od 1000 skanova u sekundi, i skaniranje je počelo u 7:00 p.m. Korisnik može kontrolisati **dt** valnog oblika.



Valni oblik Y je iscrtan na graf. Trigersko vrijeme t_0 je 7:00:00 PM, dt je 1 milisekunda, tako da 10.000 skanov vrijednosti će biti distribuirano u intervalu 10 sekundi, sa posljednjom vrijednošću iscrtanom u 7:00:10 PM.

Adresiranje kanala, porta i brojača

VI analognih ulaza i izlaza imaju parametar kanala (**channels**), gdje možemo specificirati kanale sa kojih VI čita ili na koje piše. VI digitalnih ulaza i izlaza ima sličan parametar koji se zove lista digitalnog kanala (**digital channel list**), a ekvivalentna vrijednost se zove lista brojača (**counter list**) za brojačku VI.

Opaska: U daljem nastavku sve ove tri grupe parametara tj. **channels**, **digital channels** i **counter list**, ćemo nazivati sa **channels** (kanali).

Svaki kanal kojeg specificiramo u parametru **channels** postaje član grupe. Za svaku grupu, možemo prikupljati ili generirati podatke na kanalima koji su izlistani u grupi.

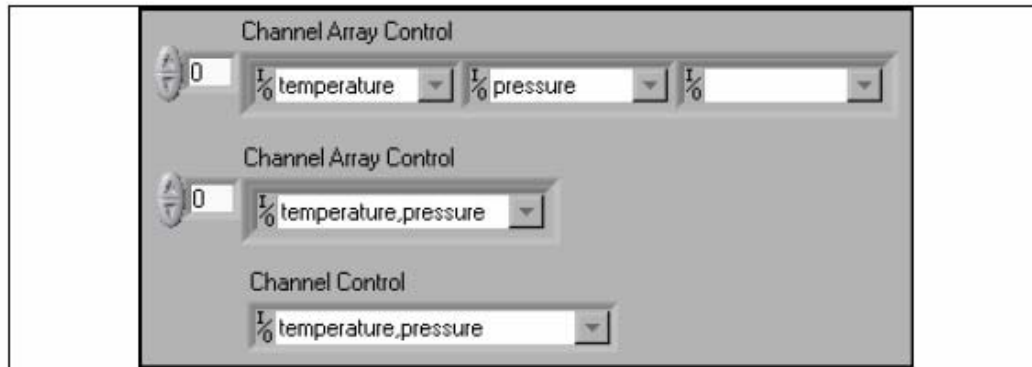
VI skaniraju (za vrijeme akvizicije) ili ažuriraju (za vrijeme generisanja), ove kanale u redoslijedu u kojem su navedeni u listi.

Da bi se izbrisala grupa, treba poslati prazan **channels** parametar i broj grupe ka VI, ili doznačiti novi **channels** parametar za grupu. Korisnik može mijenjati grupe samo na nivou Advanced VI.

Parametar **channels** kod Analognih i digitalnih VI je DNC (DAQ name channel) kontrolni element.

Adresiranje po imenu kanala (channel name addressing)

Ako koristimo DAQ Channels Wizard da konfiguriramo analogne i digitalne kanale, možemo adresirati ove kanale po imenu iz **channels** parametra u LabView. Channels može biti array varijabla stringova, ili u slučaju Easy VI, skalarni string kontrolni element, kao što je pokazano na narednoj slici.



Kontrolni elementi channels

Ako imamo **channels** array varijablu , možemo koristiti po jedan ulaz kanala za svaki element polja, specificirati cijelu listu u jednom elementu, ili koristiti bilo koju kombinaciju ova dva metoda. Ako unesemo u **channels** višestruka imena kanala , moramo konfigurirati sve kanale u listi za isti DAQ uređaj. Ako konfiguriramo kanale sa imenima : temperatura i pritisak, i obadva su mjerena sa istim DAQ uređajem.

Možemo specificirati listu kanala u jednom elementu razdvajajući ih sa zarezima, naprimjer: *temperatura, pritisak*.

Ako konfiguriramo kanale sa imenima : temp1, temp2, temp3, možemo specificirati opseg kanala razdvajajući ih sa kolonom, naprimjer: temp1: temp3.

Kod specificiranja imena kanala, spelovanje i razmaci su važni, ali veliko ili malo slovo nije važno (nije case sensitive).

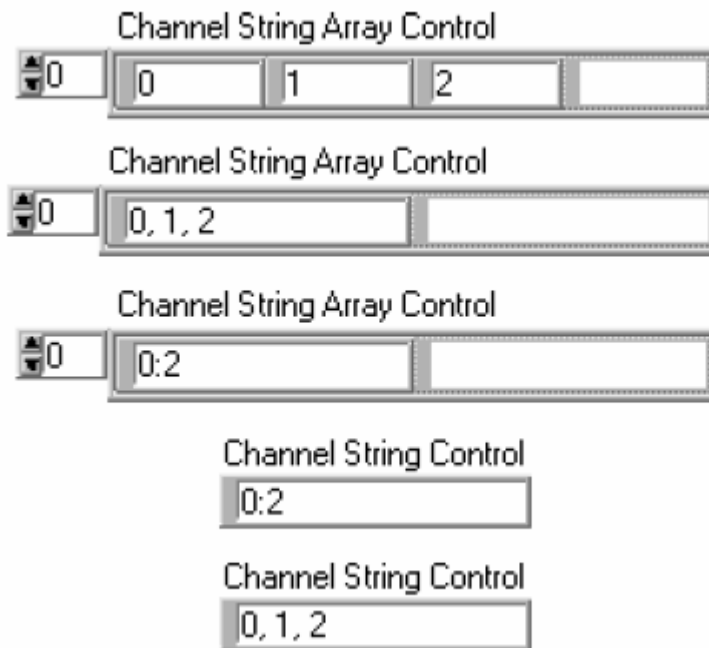
Kada koristimo imena kanala, nema potrebe da ožičavamo **device**, **input limits** ili **input config** ulazne parametre.

LabView će ignorirati **device** ulaz kada se koriste imena kanala.

Adresiranje broja kanala (channel number addressing)

Ako ne koristimo imena kanala da bi adresirali kanale, možemo ih adresirati sa brojem kanala u **channels** parametru. **Channels** može biti bilo koja varijabla polja sa string elementima , ili kod Easy VI, skalar string kontrolni element. Ako imamo **channels** polje, možemo koristiti jedan ulaz kanala po svakom elementu polja, cijelu listu u jednom elementu ili kombinaciju prva dva.

Napimjer ako imamo kanale 0,1 i 2 možemo specificirati listu kanala, u jednom elementu, odvajajući individualne kanale sa zarezom, : 0,1,2 . Ili kao opseg 0 : 2. Naredna slika pokazuje nekoliko načina kako možemo adresirati kanale 0,1i 2.



Setinzi za granične vrijednosti.

Setinzi za granične vrijednosti su minimalne i maksimalne vrijednosti koje analogni signali mogu poprimiti. Par graničnih vrijednosti može biti jedinstven za svaki analogni ulaz i/ ili izlaz.

Svaki par graničnih vrijednosti formira klaster. Limiti za analogne izlaze imaju i trećeg člana a to je referentni napon. Zbog jednostavnosti, LabView definiše ove granične vrijednosti kao par vrijednosti. LabView koristi polje ovih klastera da doznači granične vrijednosti kanala u channel **string** polju.

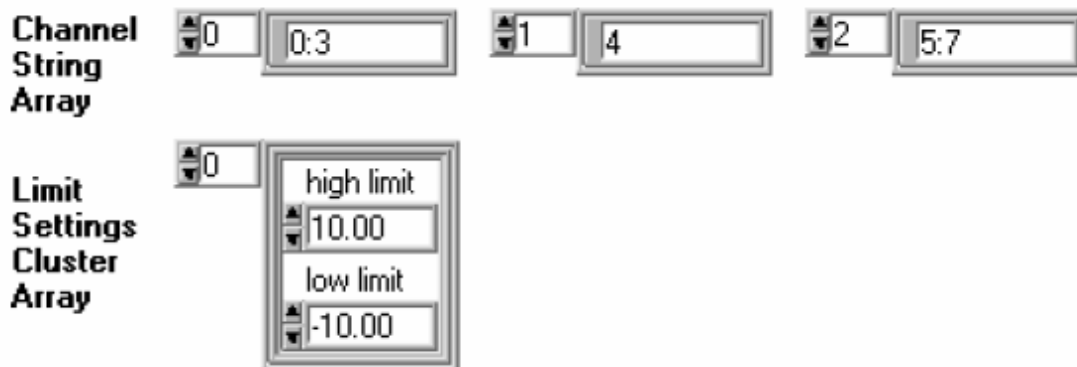
Ako ne koristimo DAQ Channel Wizard , tada je default jedinica za setinge graničnih vrijednosti Volt, mada jedinica primjenjena za ove granične vrijednosti može biti : Volti, struja, otpor, frekvencija, itd.

LabView koristi polje stringova da specificira koji kanali pripadaju grupi. LabView takodjer doznačuje sve kanale izlistane u **channels** polju i iste setinge u odgovarajućem **limit settings** klaster polju. Naredna slika ilustrira jedan ovakav slučaj:



U ovom primjeru, kanalima 0,1, 2 i 3 su doznačene granične vrijednosti 10.00 do -10.00, kanal 4 ima granice od 5.00 do -5.00. Kanali 5,6 i 7 imaju granične setinge od 1.00 do 0.00.

Ako **limit settings** klaster polje ima manje elemenata nego **channel** string polje, LabView doznačava svima preostalim kanalima granične vrijednosti sadržane u zadnjem ulazu u **limit settings** klaster polju. Naredna slika ilustrira ovaj slučaj:



U ovom primjeru, kanali 0,1,2 i 3 imaju granice od 10.00 do -10.00. Pošto ima još preostalih kanala, ali **limit settings** klaster polje je iscrpljeno. Zbog toga preostalim kanalima (4,5,6 i 7), se također doznačuju vrijednosti od 10.00 do -10.00.

Drugi DAQ VI parametri

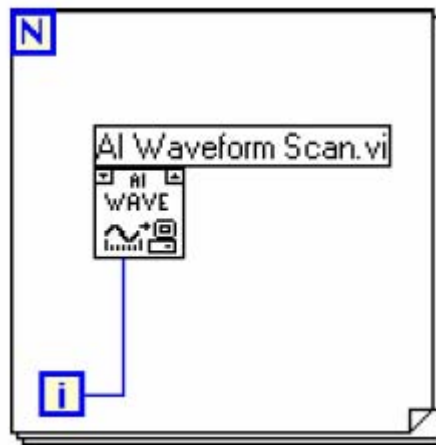
Ulaz **device** na VI-jevima za analogne I/O , digitalne I/O i brojače , specificira broj DAQ konfiguracionog softwera koji je doznačen za taj DAQ uređaj. Software doznačava jedinstveni broj za svaki DAQ uređaj. Parametar **device** obično se pojavljuje kao ulaz u konfiguracionu VI. Drugi česti parametar

konfiguracione VI je **task ID** , koji doznačava specifičnu I/O operaciju koji se koristi kroz čitav tok programa.

Neko DAQ Vi-jevi izvršavaju samo konfiguraciju uređaja ili I/O operaciju, dok drugi izvršavaju obadvoje, tj. i konfiguraciju i operaciju. Neke od ovih VI koje realizuju obadvije funkcije imaju ulaz **iteration**.

Ako VI ima **iteration** ulaz setovan na 0, LabView konfigurira DAQ uređaj a nakon toga izvršava specifičnu I/O operaciju.

Tipično, korisnik treba da ožiči **iteration** ulaz na iteracioni terminal konture kao što je pokazano na slijedećoj slici:



Kada ožičimo **iteration** ulaz na ovaj način , tada je uređaj konfigurisan samo za prvu I/O operaciju. Naredne I/O operacije će korsititi postojeću konfiguraciju.

Organizacija analognih podataka

Ako prikupljamo podatke sa više od jednog kanala višestruko, podatci se mogu vratiti kao polje valnih oblika. Svaki valni oblik predstavlja odvojeni kanal u polju valnih oblika.

Podatci mogu takodjer biti vraćeni kao dvodimenzionalno polje (2D). Objasnice organizaciju analognih podataka u obliku 2D polja.

Ako kreiramo 2D polje i labeliramo selektore indeksa ja prednjem panelu LabView-a, tada će polje izgledati kao na slijedećoj slici:



Promjer 2D varijable polja

Dva boksa sa lijeve strane su indeksni selektori za red i kolonu polja. Gornji indeks selektira red a donji indeks selektira kolonu.

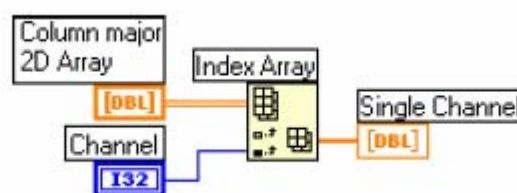
VI Analognih ulaza organiziraju svoje podatke po kolonama. Svaka kolona sadrži podatak sa jednog kanala, tako da selektirajući kolonu, mi selektiramo i kanal. Selektirajući red mi selektiramo skaniranu vrijednosti podataka u datom trenutku po svim kanalima. Ovaj način uredjenja se naziva uredjenje po koloni (column major order). Ako bi labelirali selektore indeksa za 2D polje sa uredjenjem po koloni, ona bi izgledala kao na slijedećoj slici:

scan	▲▼ 0	sc0, ch0	sc0, ch1	sc0, ch2	sc0, ch3
channel	▲▼ 0	sc1, ch0	sc1, ch1	sc1, ch2	sc1, ch3
		sc2, ch0	sc2, ch1	sc2, ch2	sc2, ch3
		sc3, ch0	sc3, ch1	sc3, ch2	sc3, ch3

Da bi iscrtali 2D polje sa uredjenjem po koloni, moramo konfigurirati valni oblik grafa ili charta da tretira podatke kao transponovane uključivanjem ove opcije u pop-up meniju za graf.

Opaska: Ova opcija je zamagljena (dimmed), sve dok ne ožičimo 2D polja sa grafom. Da bi konvertovali podatke sa uredjenjem po redu, treba koristiti funkciju **Transpose 2D array** koja je raspoloživa u **Functions>>Array** paleti. Možemo također transponovati podatke u polju za graf , klikajući desnim tasterom na graf i izabirući **Transpose Array** iz pop-up menija.

Da bi izvadili jedan kanal iz 2D polja uredjenog po koloni, treba koristiti funkciju **Index Array** , koja je raspoloživa u **Functions>>Array** paleti. Selektiramo kolonu (tj. kanal) na taj način što ćemo ožičiti selekciju sa indeksnim ulazom na dnu lijevo, i funkcija **Index Array** će proizvesti cijelu kolonu podataka kao 1D polje kao što je pokazano na narednoj slici:

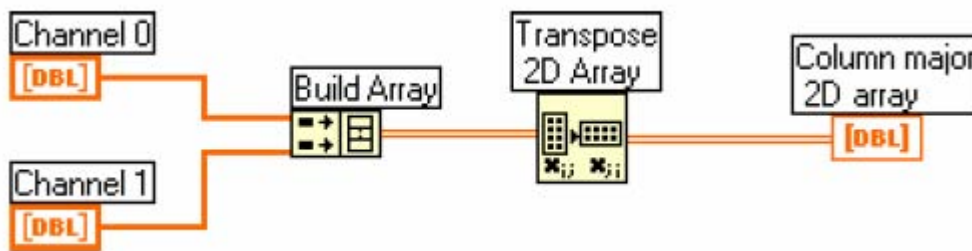


Baferi analognih izlaza koji sadrže podatke za više od jednog kanala su također 2D polja uredjena po koloni. Da bi se kreirala takva varijabla polja, potrebno je prvo urediti izlaz iz svakog izlaznog kanala kao 1D polje. Nakon toga treba selektirati funkciju **Build Array** na paleti **Functions>>Array & Cluster**. Treba dodati ulaznih terminala (redova) koliko je potrebno tj. koliko imamo kanala podataka.

Zatim je potrebno ožičiti svako 1D polje na terminal **Build Array** da se kombinuju ova polja u jedinstven red 2D polje sa uredjenjem po redu (major row).

Nakon toga možemo koristiti funkciju **Transpose 2D array** da konvertujemo polje uredjeno po redu u polje uredjeno po koloni.

Ovako uredjeno polje je sada spremno da se upiše na **AO Write VI**, kao što se vidi sa slijedeće slike:

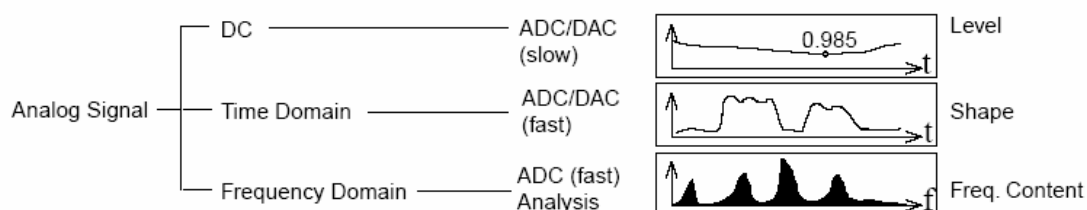


POGLAVLJE 15

KARAKTERISTIKE DAQ MODULA

Analogni ulazi

Analogni signali mogu biti grupisani u tri kategorije: DC, vremenski domen i frekventni domen. Naredna slika ilustrira koji tipovi signala korespondiraju kojim od pobrojanih kategorija:



Tipovi analognih signala

Korisnik mora definirati još nekoliko karakteristika signala, prije nego što može početi njihovo mjerenje. Na primjer:

- u odnosu na šta se signal referencira?
- Kako brzo signal varira u vremenu

Možemo tretirati DC signal kao formu vremenskog signala. Kod sporo promjenljivog signala, možemo uzeti samo jednu tačku za mjerenje. Međutim, neki DC signali mogu imati šum, pa je potrebno prikupiti više od jedne tačke a onda usrednjiti da se eliminira uticaj šuma.

Za signale u vremenskom i frekventnom domenu, mi ćemo prikupiti više tačaka podataka sa brzim vremenom skaniranja. Brže sampliranje prikuplja više tačaka u datom intervalu vremena, i zbog toga često daje bolju predstavu originalnog signala, nego sporije uzorkovanje,

Brzina uzorkovanja koju koristimo zavisi od tipova osobina koje pokušavamo da nadjemo u valnom obliku signala koji uzorkujemo. Naprimjer, ako pokušavamo da otkrijemo kratki impuls u vremenskom domenu, moramo uzorkovati dovoljno brzo da ne propustimo impuls. Vrijeme između sukcesivnih skaniranja mora biti manje od perioda impulsa. Ako uz to želimo da izmjerimo i vrijeme porasta impulsa, moramo samplirati sa još većom brzinom, koja zavisi od toga kako brzo impuls raste.

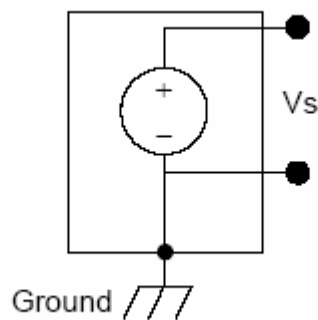
Ako mjerimo frekventne karakteristike valnog oblika, često ne moramo uzorkovati tako brzo kao kod mjerenja u vremenskom domenu. U skladu sa Nyquistovim kriterijem, moramo uzorkovati sa brzinom koja je veća od dvostruke maksimalne

frekvencije komponente u signalu da bi dobili tačnu informaciji o frekvenciji signala kojeg mjerimo. Frekvencija koja je jednaka polovini brzine samplovanja se naziva Nyquistova frekvencija.

Signali se pojavljuju u dva oblika : referencirani i ne referncirani izvori signala. Češće, referencirani izvori signala se nazivaju uzemljeni signali, a ne referencirani signali se nazivaju plivajući (floating) signali.

Uzemljeni izvori signala

Uzemljeni izvori signala imaju napone koji se referenciraju u odnosu na zemlju, ili uzemljenje objekta.

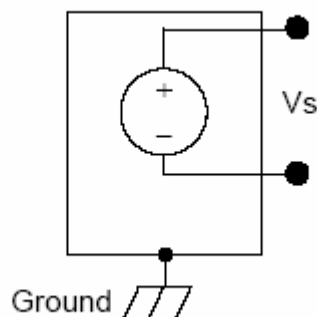


Uzemljeni izvori signala

Plivajući (floating) izvori signala

Plivajući izvori signala sadrže signal, kao naprimjer napon, koji nije povezan sa apsolutnom referencom, kao što je uzemljenje ili masa. Ovakvi primjeri su, baterije, termoelementi, transformatori, izolaciona pojačala, i instrumenti koji imaju plivajuće izlaze.

Naredna slika pokazuje ovakav tip izvora signala:



Plivajući izvor signala

Izbor mjernog sistema

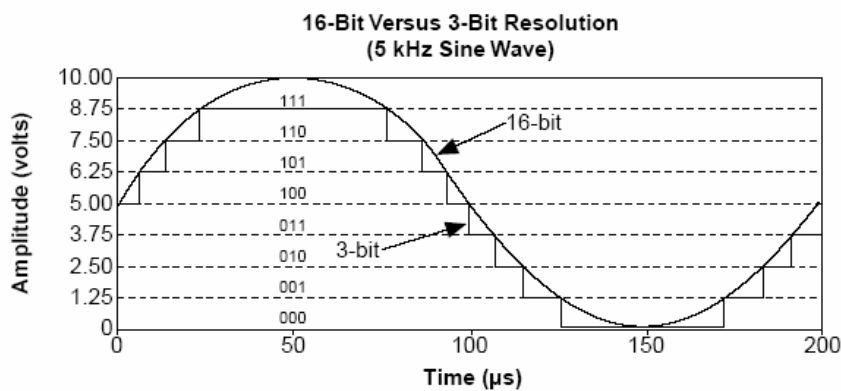
Neka od pitanja koja treba riješiti prije izbora DAQ mjernog dijela sistema su :
 Rezolucija AD konvertora, opseg uređaja , i opseg signala.

Rezolucija

Broj bita koji se koristi da predstavi analogni signal određuje rezoluciju AD konvertora. Ukoliko je veća rezolucija, veći je broj podjela u koji DAQ modul može izdijeliti ADC opseg, i time je manja promjena u signalu koja se može otkriti. 3 bitni ADC dijeli opseg u $2^3 = 8$ dijelova.

Binarni kod između 000 i 111 , predstavlja ovu podjelu. Naredna slika pokazuje sinusni valni oblik dobijen sa 3 bitnim ADC-om.

Povećavajući rezoluciju na 16 bita, broj podjela ADC raste sa 8 na 65,536 (2^{16}). ADC može sada dati vrlo tačnu predstavu analognog signala.

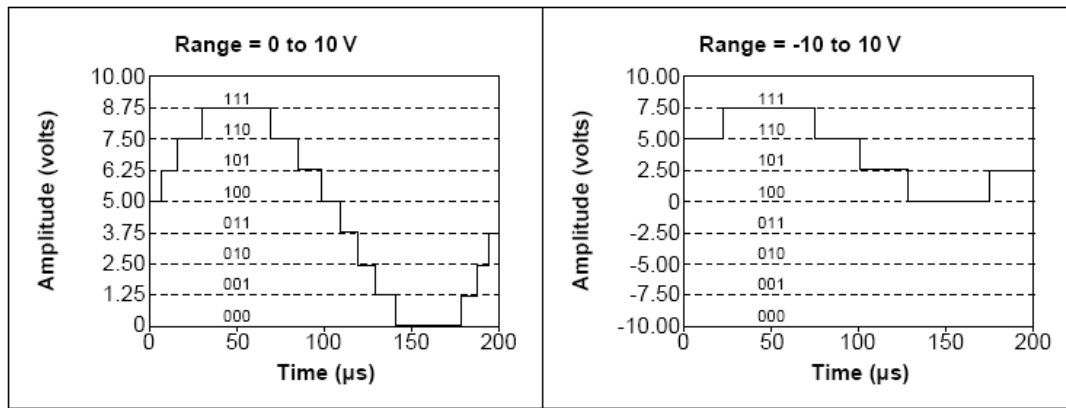


Efekti rezolucije na preciznost ADC-a

Opseg uređaja (device range)

Opseg se odnosi na minimalni i maksimalni nivo signala koji ADC može digitizirati. Mnogi DAQ uređaji imaju selektabilne opsege, tako da možemo da uparimo ADC opseg sa opsegom signala da bi se iskoristila najbolja rezolucija .

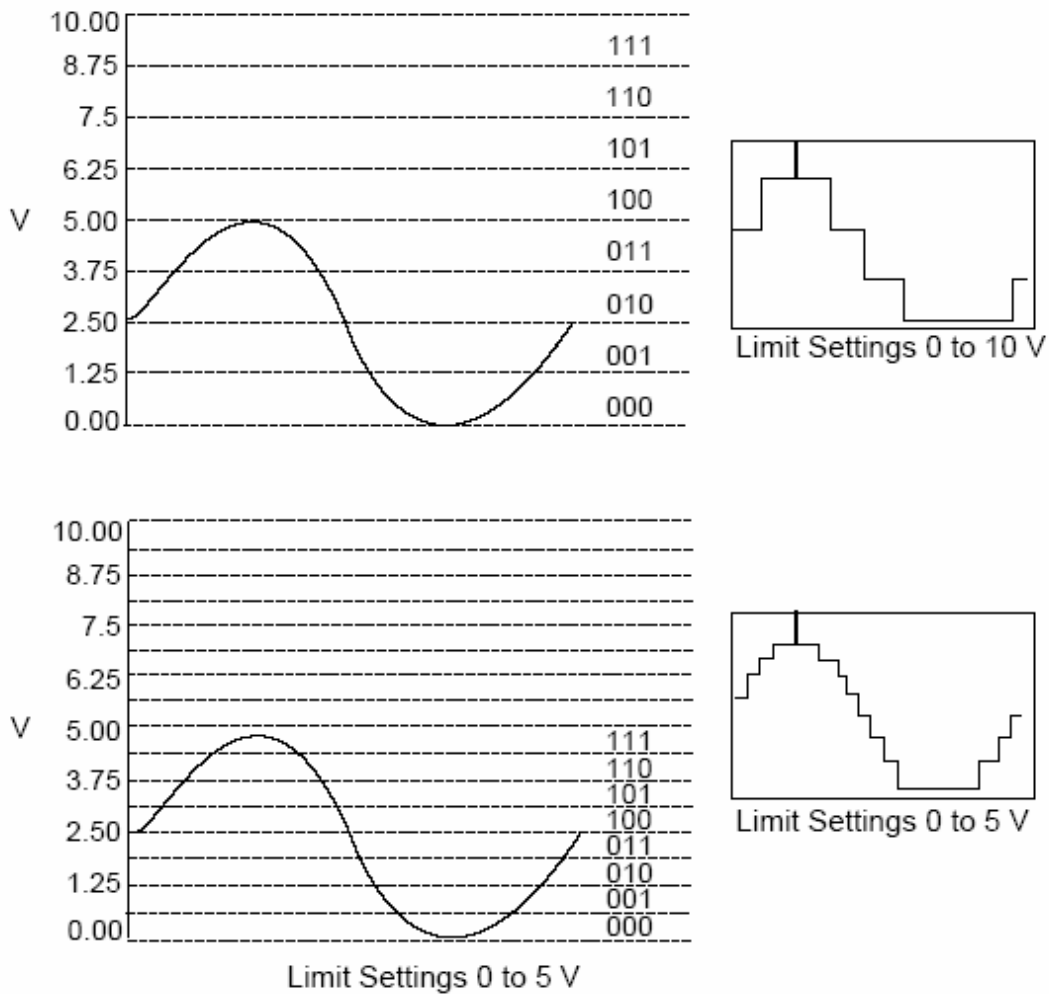
Naprimjer, na narednoj slici, 3 bitni ADC, pokazan na lijevom dijelu slike, ima 8 digitalnih podjela, u opsegu od 0 do 10 V. Ako izaberemo opseg od -10.00 do 10.00 V, kao što je pokazano na desnom dijelu slike, isti ADC sada dijeli opseg od 20 V u osam podjela. Najmanji detektabilni napon se sada povećava sa 1.25 V na 2.5 V, i imamo mnogo manje tačnu predstavu signala.



Efekti opsega na ADC preciznost.

Postavne vrijednosti granica signala

Granične vrijednosti su maksimalne i minimalne vrijednosti signala kojeg mjerimo. Preciznije granične vrijednosti dozvoljavaju da ADC koristi više digitalnih podjela da predstavi signal. Naredna slika pokazuje primjer ove tvrdnje:



Koristeći 3 bitni ADC i opseg uredjaja od 0.00 do 10.00 V, slika pokazuje efekat postavljanja graničnih vrijednosti između 0 i 5 V i 0 i 10 V. Sa granicama od 0 do 10 V, ADC koristi samo 4 od 8 digitalnih podjela u konverziji. Ali, ako koristimo granice od 0 do 5 V, ADC sada ima pristup do svih 8 digitalnih segmenata. Ovo će učiniti digitalnu predstavu signala tačnijom.

Razmatranja o selekciji opsega analognog ulaza

Rezolucija i opseg uredjaja DAQ modula određuju namanju detektabilnu promjenu u ulaznom signalu. Mi možemo izračunati najmanju detektabilnu promjenu, koja se zove širina koda (code width), koristeći slijedeću formulu:

$$\text{Širina koda} = \frac{\text{opseg uredjaja}}{2^{\text{rezoluciju}}}$$

Naprimjer, 12 bitni DAQ uredjaj sa 0 do 10 V ulaznim opsegom detektuje 2.4 mV promjenu, dok isti uredjaj sa -10 do 10 V ulaznog opsega detektuje promjenu od 4.8 mV.

$$\frac{\text{opseg uredjaja}}{2^{\text{rezoluciju}}} = \frac{10}{2^{12}} = 2.4 \text{ mV}$$

$$\frac{\text{opseg uredjaja}}{2^{\text{rezoluciju}}} = \frac{20}{2^{12}} = 4.8 \text{ mV}$$

AD konvertor visoke rezolucije obezbjeđuje manju širinu koda za dati opseg signala koji se konvertuje.

$$\frac{\text{opseg uredjaja}}{2^{\text{rezoluciju}}} = \frac{10}{2^{16}} = 0.15 \text{ mV}$$

$$\frac{\text{opseg uredjaja}}{2^{\text{rezoluciju}}} = \frac{20}{2^{16}} = 0.3 \text{ mV}$$

Što je manja širina koda, biće tačnije mjerenje.

Mi moramo također da znamo da li je naš signal jednopolaran ili bipolaran. Jednopolarni signali su signali čiji je opseg od 0 do pozitivne vrijednosti (naprimjer, 0 do 5 V).

Bipolarni signali su signali čiji opseg je od negativne do pozitivne vrijednosti (naprimjer -5 do 5 V).

Da bi se postigla manja širina koda, kada je signal unipolaran, treba specificirati da je opseg uredjaja unipolaran. Ako je opseg signala manji od opsega uredjaja, treba postaviti granične vrijednosti koje tačnije odražavaju opseg signala. Naredna tabela pokazuje kako širina koda od 12 bita DAQ uredjaja varira sa opsegom uredjaja i graničnim vrijednostima, pošto granične vrijednosti automatski podese pojačanje na uredjaju.

Preciznost mjerenja za različite opsege uredjaja i granične vrijednosti
(12 bitni A/D konvertor)

Device Voltage Range	Limit Settings	Precision ¹
0 to 10 V	0 to 10 V	2.44 mV
	0 to 5 V	1.22 mV
	0 to 2.5 V	610 μ V
	0 to 1.25 V	305 μ V
	0 to 1 V	244 μ V
	0 to 0.1 V	24.4 μ V
	0 to 20 mV	4.88 μ V
-5 to 5 V	-5 to 5 V	2.44 mV
	-2.5 to 2.5 V	1.22 mV
	-1.25 to 1.25 V	610 μ V
	-0.625 to 0.625 V	305 μ V
	-0.5 to 0.5 V	244 μ V
	-50 to 50 mV	24.4 μ V
	-10 to 10 mV	4.88 μ V
-10 to 10 V	-10 to 10 V	4.88 mV
	-5 to 5 V	2.44 mV
	-2.5 to 2.5 V	1.22 mV
	-1.25 to 1.25 V	610 μ V
	-1 to 1 V	488 μ V
	-0.1 to 0.1 V	48.8 μ V
	-20 to 20 mV	9.76 μ V
¹ The value of 1 Least Significant Bit (LSB) of the 12-bit ADC. In other words, the voltage increment corresponding to a change of 1 count in the ADC 12-bit count.		

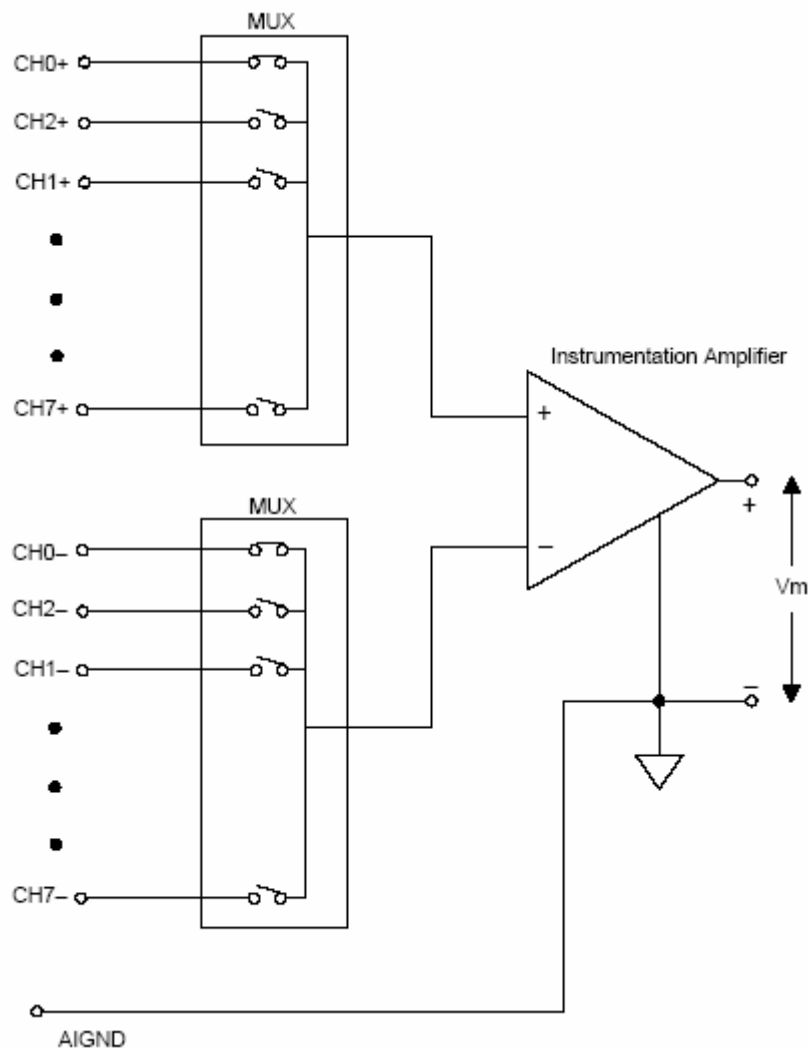
Sada kada znamo koja vrsta ADC se treba koristiti i koje granične vrijednosti treba koristiti za signal koji se konvertuje, možemo ga spojiti za mjerenje. Kod većine DAQ uredjaja postoji tri načina konfigurisanja uredjaja da konvertuje signal:

- Diferencijalni,
- referencirani jednostruki (referenced single ended)

- nereferecirani jednostruki (NRSE)

Diferencijalni mjerni sistem

Kod diferencijalnog mjernog sistema, nema potrebe da se povezuje bilo koji ulaz na fiksnu referencu, kao što je uzemljenje ili masa signala. DAQ uređaji sa instrumentalnim pojačalima se mogu konfigurirati kao diferencijalni mjerni sistem. Naredna slika prikazuje 8 kanalni diferencijalni mjerni sistem koji se koristi kod MIO serije DAQ modula. Analogni multiplekseri povećavaju broj mjernih kanala dok se još uvijek koristi samo jedno instrumentalno pojačalo. Za ovaj uređaj, pin sa oznakom AIGND (zemlja za analogni ulaz), je uzemljenje mjernog sistema.



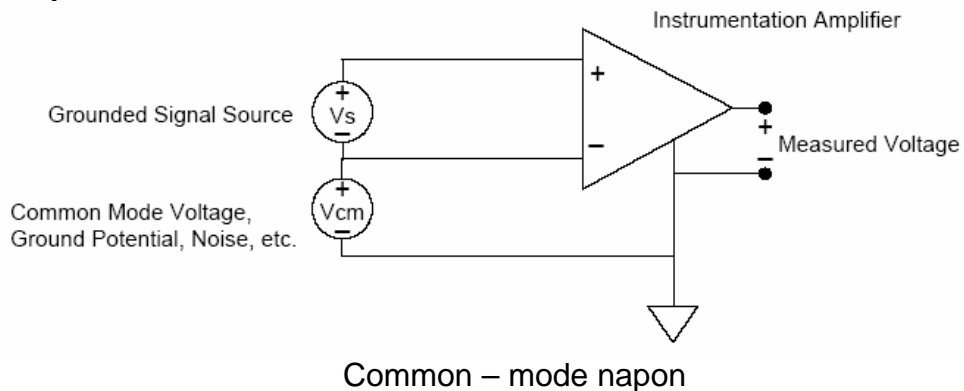
8 kanalni diferencijalni mjerni sistem

Općenito, diferencijalni mjerni sistem se preferira pošto on odbacuje ne samo greške indukovane sa konturom umašenja (ground loop induced), nego takodjer šum koji se kupi iz okoline. Treba koristiti diferencijalni mjerni sistem kada svi ulazni signali ispunjavaju slijedeće kriterije:

- nisko nivovski signali (naprimjer , manje od 1 V)

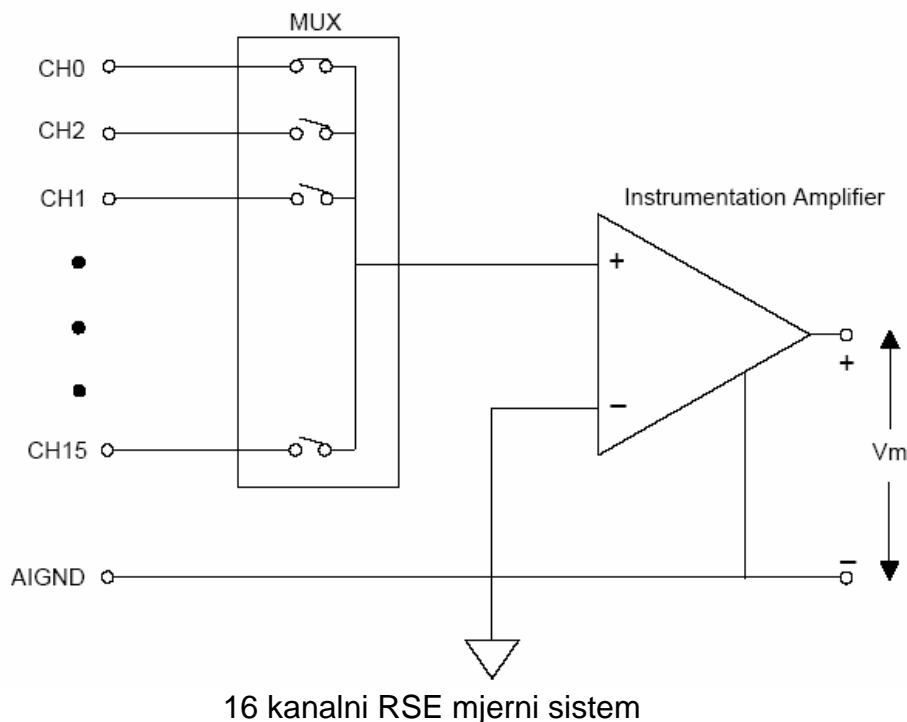
- dugi ili neširmovani kablovi i ožičenja koji prolaze kroz okruženje sa mnogo šumova.
- Svaki od ulaznih signala zahtjevaju odvojenu referentnu tačku prema masi ili povratnom signalu

Idealni sistem diferencijalnog mjerenja očitava potencijalnu razliku između njegova dva terminala, pozitivnog (+) i negativnog (-) ulaza. Svaki napon koji je prisutan na ulazima instrumentalnog pojačala u odnosu na masu pojačala se naziva *common-mode* napon. Idealni diferencijalni mjerni sistem potpuno odbacuje (tj. ne mjeri), napone common-moda, kao što je to pokazano na narednoj slici:



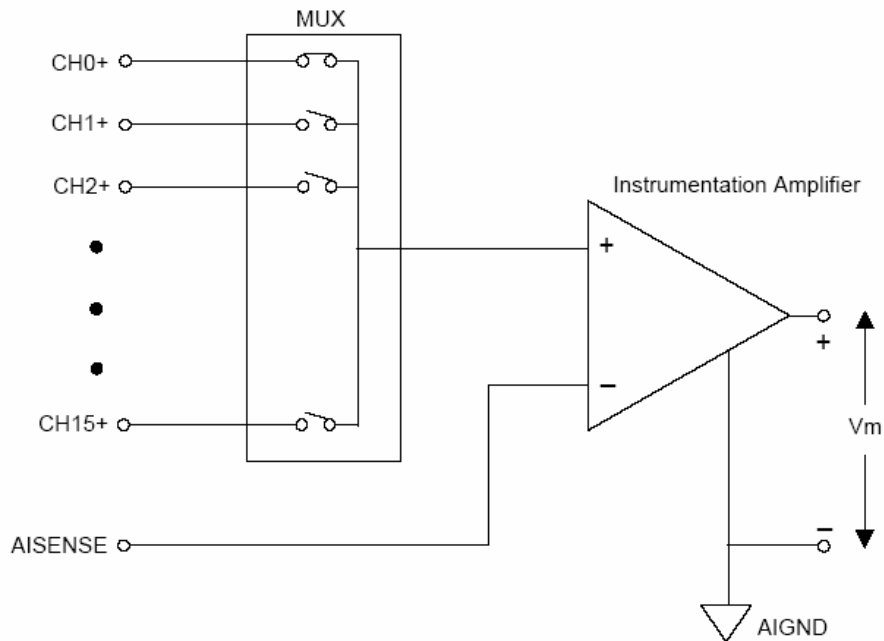
Referencirani jednostruki (single ended) mjerni sistem (RSE)

RSE mjerni sistem se koristi da mjeri plivajući signal, pošto on umašuje signal u odnosu na uzemljenje. Naredna slika pokazuje 16 kanalni RSE mjerni sistem. Treba koristiti ovaj mjerni sistem samo onda kada trebamo jednostruki (single ended -SE) a naš uređaj ne radi sa NRSE mjerenjem.



Nereferencirani jednostruki (SE) mjerni sistem (NRSE)

DAQ uređaji često koriste varijantu RSE mjerne tehnike poznatu kao NRSE mjerni sistem. Kod NRSE mjernog sistema, sva mjerenja se vrše u odnosu na zajedničku referencu, pošto su svi ulazni signali već umašeni. Naredna slika prikazuje NRSE mjerni sistem gdje AISENSE je zajednička referenca za mjerenje a AIGND je sistemska masa. Svi signali moraju djeliti zajedničku referencu u AISENSE.



16 kanalni NRSE mjerni sistem

Općenito, diferencijalni mjerni sistem se preferira pošto, kako smo već rekli, odbacuje ne samo greške zbog kontura uzemljenja, nego i zbog odbacivanja šuma iz okoline. Sa druge strane, jednostruke konfiguracije dozvoljavaju dva puta više mjernih kanala i prihvatljive su kada amplituda induciranih grešaka je manja od zahtjevane tačnosti podataka.

Korisnik može koristiti jednostruka mjerenja kada svi ulazi ispunjavaju slijedeće kriterije:

- visoko nivovski signali (normalno iznad 1 V)
- kratki ili propisno oklopljeni (širmovani) kablovi odnosno ožičenje se koristi koje prolazi kroz sredinu bez mnogo šuma.
- Svi signali mogu djeliti zajedničku referencu na izvoru signala

Adresiranje kanala s AMUX-64T

AMUX-64T vanjski multiplekser proširuje broj analognih ulaznih signala koje DAQ uredjaj može da mjeri. Korisnik može priključiti 1, 2, ili 4 AMUX-64T modula na DAQ uredjaj. Broj AMUX modula se postavlja u okviru konfiguracije uredjaja sa Measurement & Automation Explorerom. Svaka 4 kanala na AMUX modulu su multipleksirana na jedan kanal DAQ uredjaja.

U LabView , svaki kanal sa DAQ pločice korespondira sa 4 AMUX kanala na svakom AMUX-u. Naprimjer, sa jednim AMUX-64T , kanalni string 0:1 prikuplja podatke sa AMUX kanala od 0 do 7, itd.

Korisnik može prikupljati podatke sa jednog AMUX-64T kanala koristeći kanalni string **Amy!x** , gdje **x** definira broj kanala a **y** definira broj željenog AMUX-a.

(tako je y=1 ako imamo samo jedan AMUX konfigurisan). Naprimjer AM3!8 će vratiti kanal 8 na trećem konfigurisanom AMUX-64T.

Akvizicija jednostruke vrijednosti (single point)

U nastavku ćemo pokazati kako prikupiti samo jedan podatak sa jednog kanala, a zatim po jedan sa nekoliko kanala.

Jednokanalni, jednostruki analogni ulaz

Jednokanalni, jedno tačkasti analogni ulaz je trenutačna , nebaferovana operacija. Drugim riječima software očitava jednu vrijednost sa ulaznog kanala i odmah vrati vrijednost. Ova operacija ne zahtjeva nikakvo baferovanje niti tajmiranje.

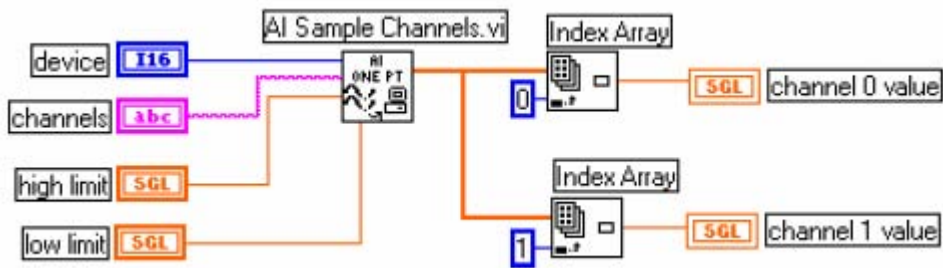
Kao primjer ove akvizicije pogledati VI **AI Sample Channel** u **Functions>>Data Acquisition>>Analog input** paleti.

Primjer ove akvizicije je dat u **examples\daq\analogin\analogin.llb** u primjeru **Acquire 1 Point from 1 Channel** VI .

Višekanalni, jednostruki analogni ulaz

Kod višestrukih kanala , sa jednotačnim očitanjem, LabView vraća vrijednosti nekoliko kanala odjedanput. DAQ uredjaj izvršava skeniranje kroz sve specificirane kanale i vraća njihove vrijednosti kada završi.

Easy I/O VI, **AI Sample Channels**, prikuplja pojedinačne vrijednosti više kanala. AI Sample Channels VI izvršava jednu A/D konverziju na specificiranim kanalima i vraća skalirane vrijednosti u valnom obliku. Očekivani opseg ovih signala , specificiran sa **high limits** i **low limits** unosima kod konfigurisanja, će se koristiti kod ovih kanala. Naredna slika pokazuje kako prikupiti signale sa više kanala koristeći ovu VI.



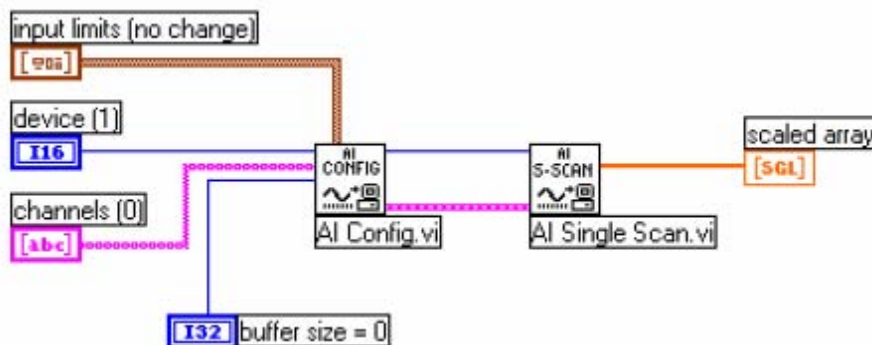
Prikupljanje napona sa više kanala

Easy Analog Input VI-jevi imaju nekoliko prednosti. Treba nam samo jedna ikona u blok dijagramu da bi se izvršio task. Easy VI zahtjevaju samo nekoliko baznih ulaza, i imaju ugrađenu provjeru grešaka. Međutim sa njima ne možemo realizovati neke detaljnije funkcije kao što je trigerovanje ili skaniranje na intervalu. Dodatno, ove VI uvijek se rekonfigurišu kod startanja, što može usporiti vrijeme procesiranja.

Kada trebamo više efikasnosti i brzine, treba koristiti srednje (Intermediate) VI, koje konfiguriraju akviziciju samo jedanput, a onda kontinualno prikupljaju podatke bez rekonfigurisanja.

Naredna slika pokazuje simplifikirani blok dijagram za nebaferovane aplikacije. LabView poziva **AI Config** VI, koja konfigurira kanale, selektira **input limits** (**high limit** i **low limit** ulaze kod Easy VI-jeva) i generira **taskID**.

Program prenosi **taskID** i klaster greške na **AI Single Scan** VI, koja vraća podatke u polju (po jednu tačku za svaki specificirani kanal).



Korištenje srednje VI za osnovne ne baferovane aplikacije

Pogledati primjer : **Cont Acq&Chart (immediate)** VI u **Functions>>Data Acquisition>>Analog input** da se vidi kako programirati AI Config i AI Single Scan Vi-jeve da izvršavaju niz jednostrukih skanova koristeći softwaresko tajmiranje (while loop) , i procesiranje svakog skana.

Da bi pozvali AI Config VI samo jedanput treba je staviti van While konture u programu. AI Config VI konfigurira kanale , selektira high i low granice, generiše taskID .

Nakon toga AI Config VI prenosi taskID i klaster greške u While konturu. LabView poziva AI Single Scan VI da izvede skan i prenese povratne podatke na My Single-Scan Processing VI. Sa ovom VI, korisnik može programirati bilo koje potrebe procesiranja, Ova VI nakon toga prenosi podatke kroz Build Array funkciju i na chart valnog oblika radi displeja na prednjem panelu.

Ovaj primjer koristi softwareski tajmiranu (pomoću metronoma) akviziciju. Sistemski sat može biti prekinut od strane interakcije sa korisnikom, tako da ako nam nije potrebna precizna brzina akvizicije, treba koristiti ovo softwaresko tajmiranje akvizicije.

Korištenje analognih ulazno/izlaznih kontrolnih kontura

Kada želimo da izbacimo analogne izlaze nakon što smo primili neke podatke sa analognih ulaza, treba koristiti analogne ulazno/izlazne kontrolne konture. Sa kontrolnim konturama , ovaj proces se stalno ponavlja.

Jednostruki analogni ulaz i izlaz VI, podržava nekoliko analognih I/O kontrolnih kontura odjednom pošto možemo prikupljati analogne ulaze od nekoliko različitih kanala u jednom skanu i upisivati sve analogne izlaze sa jednim ažuriranjem. Korisnik izvršava jedan poziv analognog ulaza, procesira vrijednosti analognih izlaza za svaki kanal, i onda izvršava jednostruki poziv analognog izlaza da ažurira sve analogne izlazne kanale.

Korištenje softwareski tajmirane analogne I/O kontrolne konture

Sa softwareski tajmiranim analognim kontrolnim konturama brzina analogne akvizicije i posljedično brzina kontrolnih kontura su kontrolisani sa softwareskim tajmerom kao što je naprimjer: **Wait Until Next ms Multiplier** tajmerom. Akvizicija se izvršava za vrijeme svake iteracije konture kada AI Single Scan AI se poziva i kontrolna kontura se izvršava jedanput za svaki interval vremena. Tajming konture može biti prekinut sa bilo kakvom interakcijom korisnika, što znači da brzina akvizicije nije konstantna i konzistentna kao što bi bila sa hardwareski tajmiranom upravljačkom konturom. Ovakav način softwareskog tajmiranja se može koristiti kada nije potrebna precizna brzina akvizicije.

Osim interakcije sa korisnikom, i veliki broj prednjih panela sa velikim grafičkim indikatorima kao što su chartovi i grafovi utiču na brzinu kontrolnih kontura . Osvježavanje ekrana monitora interaptira sistemski sat , koji kontroliše brzinu kontura. Zbog toga treba držati broj ovih chartova i grafova na minimumu kada koristimo softwareski tajmirane upravljačke konture.

Pogledati **Analog IO Control Loop (immed) VI** , u **examples\daq\analog_io\analog_io.llb** , kao primjer softwareski tajmiranih upravljačkih kontura. U ovom primjeru AI Read One Scan VI konfigurira DAQ uređaj da prikuplja podatke iz analognog ulaza sa kanala 0 i 1. Nakon toga program izvršava računanja nad podacima i izbacije rezultate na izlaze preko analognih izlaznih kanala 0 i 1. Pošto je iteracioni terminal spojen sa sa AI Read One Scan i AO Write One Update VI-jevima, aplikacija konfigurira DAQ uređaj za analogni ulaz i izlaz samo na prvoj iteraciji.

Brzina konture kao i brzina akvizicije je specificirana sa **loop rate**. Razlog zašto **actual loop period** je važan je zbog toga što interakcija sa korisnikom će uticati na brzinu konture i akvizicije.

Pogledati takodjer i **examples\daq\solution\control.llb** za još primjera sa konturama upravljanja.

Korištenje hardwareski tajmiranih analognih I/O kontrolnih kontura

Za precizniji tajming kontrolnih kontura akvizicije i precizniju brzinu skaniranja analognih ulaza, treba koristiti hardwareski tajmirane kontrolne konture.

Pogledati u primjere : **Analog IO Control loop (hw timed)** VI u direktoriju : **examples\daq\analog_io\analog_io.llb** za primjer hardwareski tajmirane , ne baferovane kontrolne konture.

Sa hardwareski tajmiranim kontrolnim konturama, akvizicija neće biti prekidana interakcijom sa korisnikom. Kod ovakvog tajminga, analogni ulaz se automatski stavlja u FIFO bafer DAQ uređaja sa intervalom koji je određen sa brzinom skaniranja analognog ulaza. Korisnik može sinhronizovati svoj dijagram kontrolne konture sa ovom preciznom brzinom analognog skaniranja ulaza, time što se stalno pozivati **AI Single Scan VI** da iščitava najstarije podatke iz FIFO bafera.

AI Single Scan VI će se vratiti čim slijedeći skan je prikupljen od strane DAQ uređaja. Ako se pohrani više od jednog skana u FIFO bafer DAQ uređaja, kada se pozove AI Single Scan VI, tada LabView nije bio sposoban da se nosi sa brzinom akvizicije. Korisnik može ovo detektovati tako što će nadzirati podatke zaostale u izlazu u AI Single Scan VI.

Ovo znači da softwareski overhead spriječava nas da se nosimo sa brzinom hardwareski tajmirane brzine konture. Kada se ovo desi, Boolov indikator **loop too slow** će biti setovan TRUE u **Analog IO Control loop (hw timed)** VI.

U ovom blok dijagramu, AI Config VI konfigurira uređaj da prikuplja kanale 0 i 1. Aplikacija ne koristi bafer kreiran u CPU memoriji, nego koristi FIFO bafer DAQ uređaja. **Input limits** će uticati na očekivani opseg ulaznih signala.

AI Start VI opčinje analognu akviziciju sa parametrom **loop rate** (scan brzinom) . Pri prvoj iteraciji , AI Single Scan VI čita najnovije podatke u FIFO baferu.

Ako više od jedne vrijednosti je pohranjeno u FIFO bafer DAQ uređaja kada ga iščitavamo, aplikacija nije bila u stanju da se drži sa brzinom akvizicije u upravljačkoj konturi, i mi smo propustili jedan interval kontrolne konture. Ovo će voditi ka pojavi greške , što će završiti konturu i akviziciju.

Nakon što aplikacija kompletira analognu akviziciju i generaciju, AI Clear VI čisti task analognog ulaza.

Blok dijagram Analog IO Control Loop (hw timed) VI takodjer uključuje chart valnog oblika u kontrolnoj konturi. Ovo takodjer smanjuje maksimalnu brzinu upravljačke konture, što možemo eliminirati ako otklonimo ovaj grafički indikator.

Mi možemo dodati još procesiranja u analognu I/O konturu na taj način što ćemo staviti analogni ulaz, kalkulacije u konturi, i analogni izlaz u prvi okvir sekvence unutar konture, a dodatno procesiranje u naredne okvire sekvence.

Poboljšanje performanse upravljačke konture

Postoje neka razmatranja koja treba provesti ako planiramo da imamo i druge VI-jeve koji se izvršavaju paralelno sa hardwareski tajmiranom upravljačkom konturom. Kada pozovemo AI Single Scan VI u hardwareski tajmiranoj kontrolnoj konturi, VI čeka sve do sljedećeg skana prije nego što se vrati sa podacima, što znači da CPU čeka unutar NI-DAQ drajvera, sve dok se skan ne prikupi.

Kao posljedica ovoga, ako pokušamo da izvršavamo druge LabView VI-jeve, oni mogu da se izvršavaju sporije ili sa prekidima. Korisnik može reducirati ovaj problem sa uvođenjem softwareskog kašnjenja, sa Wait (ms) VI, na kraju konture nakon što upišemo vrijednosti analognih izlaza. Sada druge LabView VI-jevi i konture se mogu izvršiti za vrijeme ovog vremena.

Druga dobra tehnika je da se polira (izabira) analogni ulaz bez čekanja u drajveru. U tom slučaju možemo postaviti AI Single Scan VI **time limit in sec** na 0.

Nakon toga, VI čita FIFO bafer DAQ uređaja i vraća se odmah, bez obzira da li je sljedeći skan prikupljen. AI Single Scan VI **scaled data** izlazna varijabla polja je prazna, ako skan nije bio još prikupljen. Polirati za analogni ulaz koristeći Wait (ms) ili Wait Until Next ms Multiple zajedno sa AI Single Scan VI u While konturi unutar dijagrama upravljačke konture.

Postaviti vrijeme čekanja manje od intervala kontrolne konture (najmanje upola). Ako **scaled data** izlazno polje nije prazno izaći iz konture polinga, šaljući **scaled data** polje i izvršavajući ostatak dijagrama upravljačke konture.

Ovaj metod ne vraća podatke čim je skan bio prikupljen, kao u prethodnom primjeru, nego daje dovoljno vremena za druge VI-jeve da se izvrše. Ovaj metod je dobra tehnika za balansiranje opterećenja CPU između nekoliko kontura i VI-jeva koji se izvršavaju paralelno.

Pogledati primjere u **examples\daq\solution\control.llb** za analizu opisanog.

Baferovana akvizicija valnih oblika

Jedan način da se prikupljaju višestruki podatci sa jednog ili više kanala je da se koriste nebaferovani metodi koji su ranije opisani u ovom poglavlju, na repetitivan način.

Međutim, prikupljanje jedne po jedne tačke sa jednog ili više kanala jedno za drugim je vrlo neefikasno i vremenski konzumirajuće. Također sa ovom metodom akvizicije, mi nemamo tačnu kontrolu nad vremenom između svakog sampla i kanala. Zbog toga možemo koristiti bafer podataka u memoriji računara da efikasnije prikupljamo podatke.

Ako želimo da prikupimo više od jednog očitavanja sa više od jednog kanala, potrebno je prikupljati podatke kao valne oblike. Postoje dvije tehnike baferovane akvizicije valnih oblika koje možemo koristiti, zavisno od toga šta želimo činiti sa podacima nakon akvizicije:

- jednostavna baferovana akvizicija,
- akvizicija sa kružnm baferom.

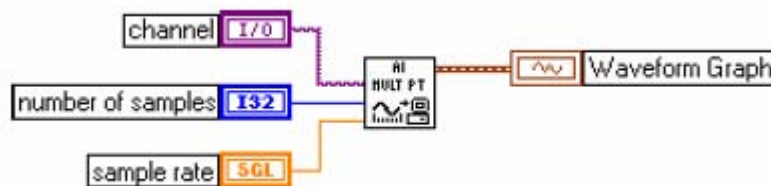
U nastavku ćemo objasniti baferovanu akviziciju valnih oblika i pokazati ove dvije tehnike.

Korištenje jednostavnih bafera za prikupljanje valnih oblika sa DAQ ulaznim VI

Sa baferovanim I/O , LabView prenosi podatke uzete u vremenskim intervalima od DAQ uređaja na bafer podataka u memoriji. U okviru VI, mi moramo specificirati broj uzoraka koje treba uzeti i broj kanala sa kojih LabView će uzimati sampleve. Iz ovih informacija, LabView alokira bafer u memoriji da pohrani broj tačaka podataka jednak broju uzoraka po kanalu pomnožen sa brojem kanala. Kako se akvizicija podataka nastavlja bafer se puni sa podacima. Medjutim , podatci neće biti dostupni sve dok LabView ne prikupi sve sampleve. Jedanput kada je akvizicija završena, podatci u baferu mogu biti analizirani , pohranjeni na disk ili prikazani na ekranu u okviru VI.

Prikupljanje jednog valnog oblika

Najlakši način da se prikupi jednostruki valni oblik sa jednog kanala je da se koristi **AI Acquire Waveform** VI, kao što je pokazano na narednoj slici:



Korištenje ovog VI zahtjeva da korisnik specificira uređaj i kanal, broj uzoraka i brzinu uzorkovanja .

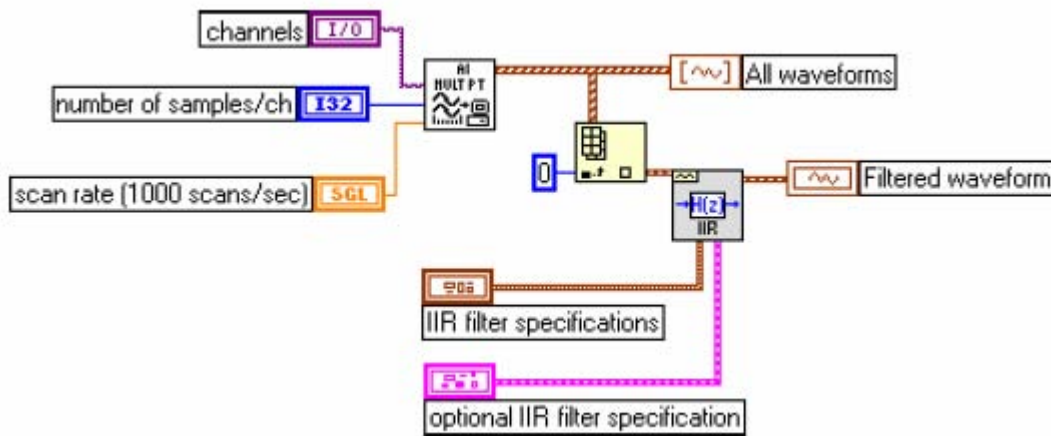
Mi možemo programatski postaviti **gain** postavljajući **high limit** i **low limit**. Koristeći samo minimalni set ulaza čini programiranje ove VI lakšim, ali nedostaju neke naprednije mogućnosti kao što je setovanje trigera.

Prikupljanje višestrukih valnih oblika

Mi možemo prikupiti više od jednog valnog oblika sa drugom Easy Analog Input Vi, i to AI Acquire Waveforms. Ova VI također ima minimalan skup ulaza, ali dozvoljava da se očita više od jednog kanala i vraća polje valnih oblika sa svim kanalima koje je očitala.

Da bi se pristupilo ili kontrolisao pojedinačni valni oblik, treba indeksirati polje valnih oblika sa funkcijom Indeksa polja (Index Array function) , ili koristiti ulazno indeksiranje na For ili While konturi.

VI na narednoj slici prikuplja valne oblike sa višestrukih kanala i crta valne oblike na grafu. Dodatno, funkcija indeksa polja, pristupa prvom valnom obliku u polju i šalje ga na filter, koji šalje valni oblik na drugi graf.



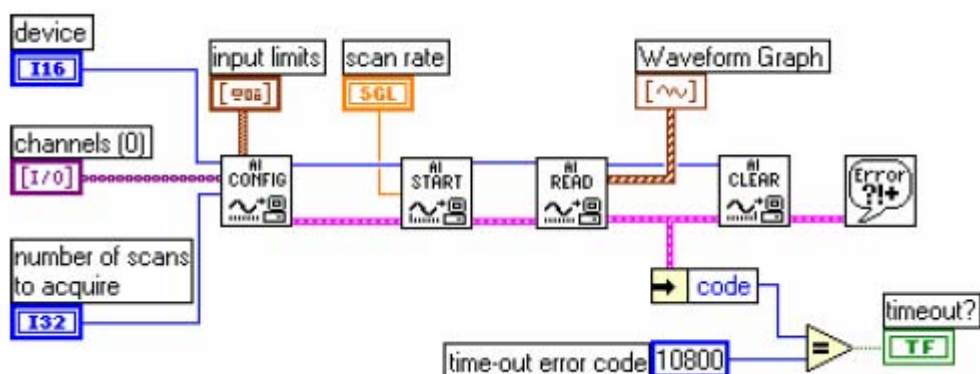
Prikupljanje i crtanje višestrukih valnih oblika i filtriranje jednog valnog oblika

Channels ulaz za AI Acquire Waveform VI, ima pull down meni , gdje možemo izabrati kanal iz liste konfigurisanih imena kanala. Mi možemo takodjer postaviti **high limit i low limit** ulaze za sve kanale na istu vrijednost. Kao i druge Easy VI, mi ne možemo koristiti nikakve napredne (advanced) programske osobine sa ovom VI.

Možemo takodjer prikupiti višestruke valne oblike koristeći srednje (Intermediate) VI. Srednje VI obezbjeduju više kontrole nad procesom akvizicije , kao naprimjer da možemo da čitamo bilo koji dio bafera.

Primjer je pokazan na narednoj slici gdje je Acquire N Scans VI, koja se može naći u **examples\daq\analogin\analogin.11b**.

Sa srednjim VI za analogne ulaze, moramo ožičiti **taskID** da bi identificirali DAQ operaciju da obezbjedimo da se VI-jevi izvršavaju u korektnom redoslijedu.



Korištenje srednje VI za prikupljanje višestrukih valnih oblika

Sa ovom VI, moćemo ne samo konfigurirati trigerovanje, kuplovanje, akviziciju, tajming, vadenje podataka, i dodatni hardware, nego takodjer kontrolisati kada se svaki korak akvizicionog procesa pojavi.

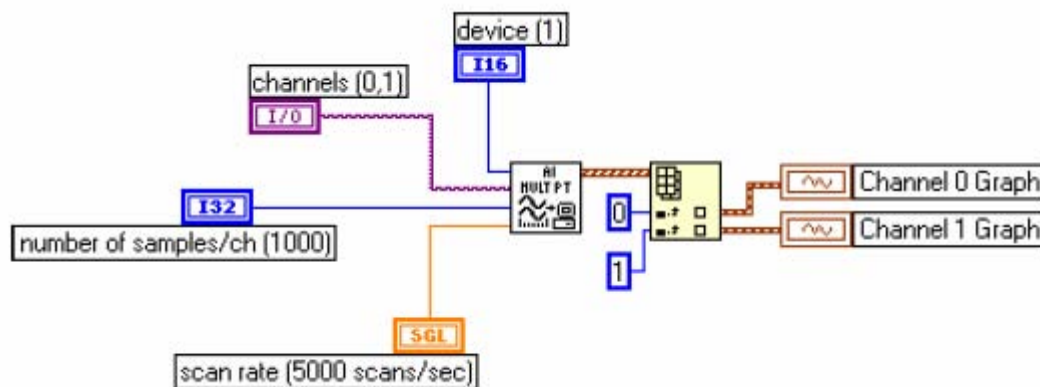
Sa AI Config VI, mi možemo konfigurirati različite parametre akvizicije, kao što su kanali koje treba očitati i veličinu bafera koju treba koristiti. U AI Start VI, korisnik specificira parametre korištene u programu da starta akviziciju, kao što je broj skanova koji će prikupiti, brzinu pri kojoj VI uzima podatke, i setinge triger. U AI Read VI, korisnik specificira parametre da izvadi podatke iz bafera akviziranih podataka.

Zatim, aplikacija poziva AI Clear VI, da dealocira sve bafere i druge resurse korištene u akviziciji time što će obezvrjediti (invalidate) **taskID**. Ako se pojavi greška u bilo kojoj od Vi-jeva, program prenosi grešku kroz preostale VI do Simple Error Handler VI, koja će u dijalogu obavjestiti korisnika o grešci.

Za mnoge DAQ uređaje , isti ADC sampluje mnoge kanale umjesto samo jednog. Maksimalna brzina sampliranja po kanalu je maksimalna brzina uređaja podjeljena sa brojem kanala.

Primjeri Jednostavno baferovane akvizicije sa iscrtavanjem

Naredna slika pokazuje kako se može koristiti AI Acquire Waveform VI da se prikupe dva valna oblika na kanalima 0 i 1 , i onda prikažu valni oblici na posebnim grafovima. Ovaj tip VI je koristan za poredjenje dva ili više valnih oblika ili za analizu kako signal izgleda prije i poslije prolaska kroz sistem. U ovom primjeru, 1000 skanova na kanalu 0 i 1 su uzeti pri brzini od 5000 skanova u sekundi. **Actual scan period** izlaz se pokazuje u aktuelnoj vremenskoj bazi na X osi grafova.



Primjer jednostavnog bafera analognog ulaza

Pogledati primjer u VI **Acquire N Scans example** u **examples \daq\analogin\analogin.llb**.

Analogni ulaz sa jednostavnim baferom i višestrukim startovima

U nekim slučajevima korisnik može željeti da prikupi kontinualni blok podataka , kao što je slučaj kod osciloskopskih aplikacija. U tom slučaju, možemo uzeti samo specificiran broj uzoraka kao snapshot od toga kako ulaz periodično izgleda.

Kao primjer možemo koristiti **Acquire N-Multi-Start** VI u `examples \daq\analogin\analogin.llb`.

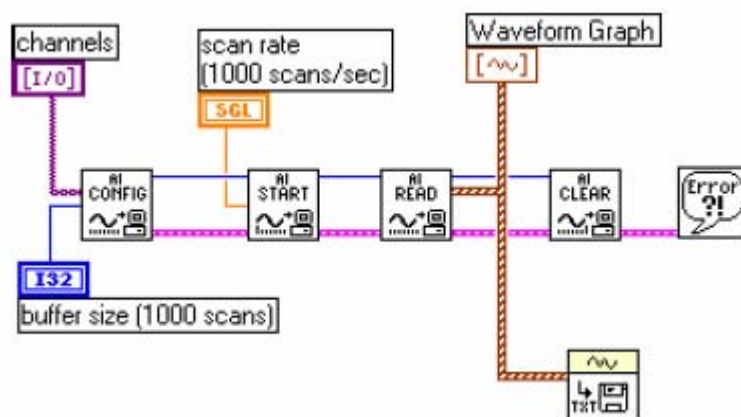
Ovaj primjer je sličan standardnom jednostavnom baferu za analogni ulaz, stim što sada i AI Start i AI Read VI-jevi su u While Loop, što znači da program uzima broj uzoraka svaki put kada While kontura iterira.

Jednostavno baferovani analogni ulaz sa upisivanjem podataka u spreadsheet fajl.

Ako želimo da upisujemo prikupljene podatke u fajl, postoje mnogo formata fajlova u kojima možemo pohraniti podatke. Spreadsheet format fajla se koristi najčešće pošto ga možemo očitavati koristeći većinu aplikacija spreadsheetova za kasnije iscrtavanje u grafu i analizu. U LabView mi možemo koristiti VI-jeve da pošaljemo podatke u fajl u spreadsheet formatu ili iščitati ih iz takvog fajla.

Možemo naći ove Vi-jeve u **Functions>>File I/O** paleti i u **Functions>>Waveform>>Waveform File I/O** paleti .

VI koji se koristi u ovom primjeru je **Export Waveforms to Spreadsheet File** VI , koji je pokazan na narednoj slici:



Upisivanje u spreadsheet fajl nakon akvizicije

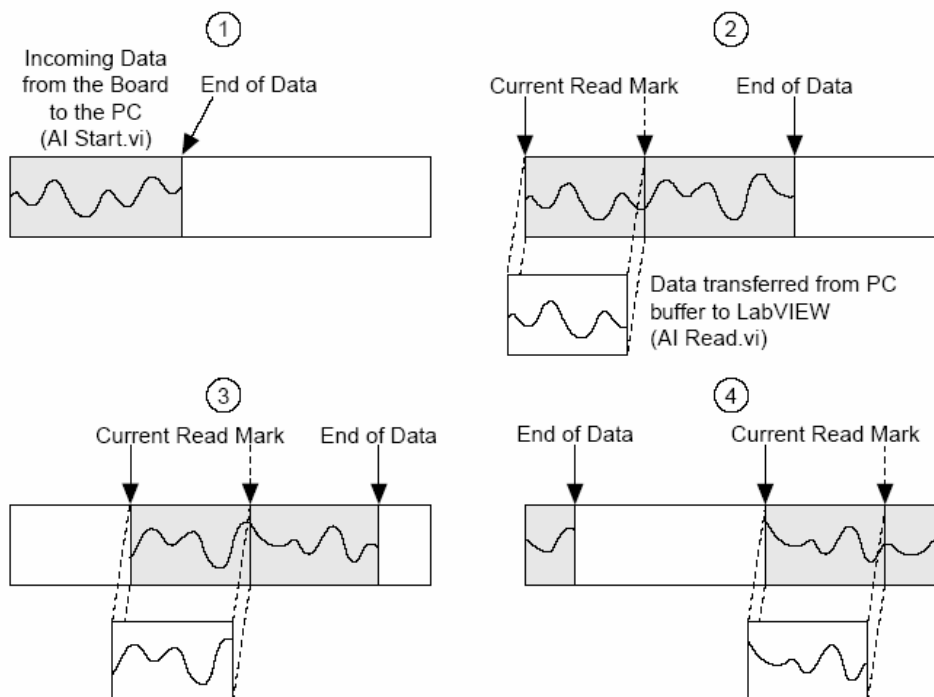
U ovom primjeru , srednja (Intermediate) VI analognog ulaza prikuplja polje valnih oblika , grafuje podatke, i kreira spreadsheet fajl koji sadrži podatke.

Korištenje kružnog bafera za pristup podacima za vrijeme akvizicije

Možemo primjeniti jednostavne tehnike baferovanja u mnogim DAQ aplikacijama, ali postoje neke aplikacije gdje ove tehnike nisu adekvatne. Ako želimo da gledamo, procesiramo, ili logujemo dijelove podatka kako su prikupljeni, ne treba koristiti ove jednostavne tehnike baferovanja.

Za ove tipove aplikacija, treba uspostaviti kružni bafer, da bi pohranili prikupljene podatke u memoriju. Naredna slika pokazuje kako kružni bafer radi. Djelovi podataka se iščitavaju iz bafera dok se bafer puni.

Koristeći kružni bafer, mi možemo setovati uređaj da kontinualno prikuplja podatke u pozadini (background) dok LabView vadi prikupljene podatke.



Kako kružni bafer radi

Kružni bafer se razlikuje od jednostavnog bafera samo po tome kako LabView postavlja podatke u njega i vadi ih iz njega. Kružni bafer se puni sa podacima , isto kao i jednostavni bafer. Međutim, kada dodje do kraja bafera, on se vraća na početak i puni ponovo isti bafer. To znači da se podatci mogu čitati kontinualno u kompjutersku memoriju, ali samo definirana količina memorije može biti korištena.

VI mora da vadi podatke iz bafera u blokovima, iz jedne lokacije u baferu, dok podatci ulaze u bafer u drugoj lokaciji, tako da nepročitani podatci se ne prepisuju sa novim podacima. Pošto je potrebno održavati bafer, možemo koristiti samo srednje i napredne VI-jeve za ovaj tip akvizicije podataka.

Dok kružni bafer radi dobro u mnogim aplikacijama, postoje dva moguća problema koja se mogu pojaviti sa ovim tipom akvizicije: VI može pokušati da vadi podatke iz bafera brže nego što se oni prikupljaju, ili VI ne vadi podatke iz bafera dovoljno brzo prije nego što LabView prepíše podatke u baferu sa novim. Kada VI pokušava da čita podatke iz bafera koji još nisu bili ni prikupljeni, LabView će čekati za podatke koje naša VI zahtjeva da se prikupe , i onda će vratiti te podatke.

Ako naša VI ne čita podatke iz kružnog bafera dovoljno brzo, VI šalje natrag grešku, saopštavajući korisniku da neki podatci mogu biti prepisani i izgubljeni.

Kontinualno prikupljanje podataka sa višestrukih kanala

Mi možemo prikupljati kontinualno vremenski samplirane podatke sa jednog ili više kanala sa srednjim VI. Pogledati primjer : **Acquire & Process N Scans VI** u **examples \daq\analogin\analogin1.llb**.

U ovom primjeru imamo ulaze za setovanje kanala, veličine kružnog bafera, brzine skeniranja, broj smplova koje treba izvaditi iz kružnog bafera.

Ova VI defaultira u **input buffer size** od 2000 smplova i **1000 number of scans to read at a time**, što znači da VI čita u polovini bafera podataka , dok VI puni drugu polovinu bafera sa novim podacima.

Ako ne vadimo podatke iz kružnog bafera dovoljno brzo, podatci koje nismo pročitali biće prepisani sa novim podacima. Mi možemo riješiti ovaj problem sa podešenjem jednog od slijedećih parametara:

Input buffer size, scan rate, ili number of scans to read at a time.

Ako naš program prepíše preko podataka podatke u baferu, tada podatci dolaze u bafer brže nego što ih naša VI može čitati i sve prethodne baferovane podatke, i Labview vraća kod greške. – **10846 overWriteWrror**.

Ako povećamo veličinu bafera tako da mu treba duže vremena da se napuni, naša VI će imati više vremena da pročita podatke iz njega.

Ako usporimo **scan rate** , mi reduciramo brzinu pri kojoj se bafer puni, i to takodjer daje našem programu više vremena da izvadi podatke. Mi možemo takodjer povećati **number of scans to read at a time**. Ovo vadi više podataka iz bafera svaki put i efektivno reducira broj puta koliko treba da pristupimo baferu prije nego on postane pun. Treba provjeriti izlazni **scan backlog** da bi vidjeli koliko vrijednosti podataka ostaje u kružnom baferu nakon čitanja. Pošto se koriste srednje VI, mi možemo takodjer kontrolisati i druge parametre kao što je trigerovanje, kuplovanje, i dodatni hardware.

Asinhrona kontinualna akvizicija koristeći DAQ pojavljivanja (occurences)

Glavna prednost prikupljanja podataka kao što je opisano u prethodnoj sekciji je u slobodi manipuliranja sa podacima izmedju poziva ka AI Read VI.

Jedino ograničenje je u tome što je akvizicija sinhrona. To znači da , jednaput kada pozovemo AI Read VI, mi ne možemo izvršiti bili kakav drugi task sve dok AI read VI ne povrati prikupljene podatke. Ako je naš DAQ uređaj još uvijek zaposlen sa skupljanjem podataka, mi ćemo morati čekati ne radeći ništa dok se ne završi VI.

Na jednoj višestaznoj (multithreaded) platformi kao što je Windows, ovo ograničenje može biti prevaziđeno sa alokacijom dodatnih threadova , ili mjenjajući preferirani izvršni sistem djelova naše aplikacije.

Druga alternativa je da koristimo asinhronu akviziciju. Mi možemo prikupiti asinhrono i kontinualno podatke sa više kanala koristeći iste srednje DAQ VI , dodavajući DAQ pojavljivanja (occurences).

Pogledati VI **Cont Acq&Chart (Async Occurence)** VI u **examples \daq\analogin\analogin1.llb** kao primjer asinhronu akvizicije.

Ovaj primjer koristi **DAQ Occurence Config VI** i **Wait on Occurence** funkciju da kontroliše očitavanja. Prvi DAQ Occurence Config VI setuje DAQ Event.

U ovom primjeru **DAQ Event** će postaviti pojavljivanje svaki put kada broj skanova koji je prikupljen je jednak vrijednosti opšte vrijednosti **A**, gdje je **A** opšta vrijednost **number of scans to read at a time**.

Unutar While Loop, Wait on Occurence funkcija spava u pozadini, sve dok izabrani **DAQ Event** ne desi.

Primjetimo da **timed out** izlaz iz Wait on Occurence funkcije je ožičen na selekcionni terminal Case strukture koja uključuje AI Read VI. Ovo znači da AI Read nije pozivan sve dok **number of scans to read at a time** nije bio prikupljen. Rezultat je da While Loop efektivno stavljena u stanje spavanja, pošto mi ne pokušavamo da pročitate podatke sve dok ne znamo da su prikupljeni. Ovo oslobadja thread izvršenja da može da radi ostale taskove dok mi čekamo na DAQ Event (dogadjaj).

Ako DAQ Occurence istekne vremenski (time out), tada će izlazna vrijednost za timed-out izlaz biti TRUE, i AI Read neće biti nikada pozvan.

Kada je akvizicija kompletirana, DAQ pojavljivanje (occurrence) se poziva opet da počisti sva pojavljivanja.

Primjer analognog ulaza sa kružnim baferom

Jedina razlika izmedju aplikacija sa jednostavnim baferima i aplikacija sa kružnim baferom u blok dijagramu je vrijednost **number of scans to acquire** ulaza kod AI Start VI, i mi moramo pozivati AI Read VI uzastopno da bi izvadili podatke. Ove promjene se mogu primjeniti na mnoge primjere u prethodnoj sekciji sa jednostavnim baferom.

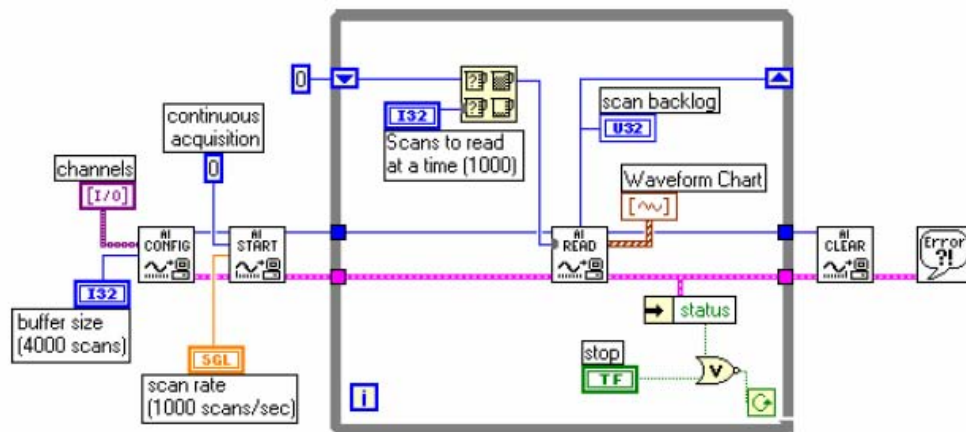
Analogni ulaz sa Baznim kružnim baferom

Naredna slika pokazuje primjer VI koja donosi podatke iz kanala pri brzini od 1000 smplova /sec, u bafer koji može držati 4000 smplova. Ovaj tip primjera može biti prikladan ako hoćemo da gledamo podatke iz kanala u dužem periodu vremena , ali ne možemo da pohranimo sve podatke odjednom u memoriju.

AI Config VI postavlja specifikaciju kanala i veličinu bafera , zatim AI Start VI inilijalizira prikupljanje podataka u pozadini (background) , pri specificiranoj brzini.

Unutar While Loop, AI Read VI ponovljivo čita blokove podataka iz bafera veličine jednake ili 1000 skanova ili veličine od **scan backlog** – zavisi šta je veće .

VI ovo realizuje koristeći Max&Min funkciju da odredi koja je veća od dvije vrijednosti. Korisnik nema potrebe da koristi Max& Min funkciju na ovaj način da bi aplikacija radila , ali ova funkcija pomaže da kontrolišemo veličinu od **scan backloga**, koji ustvari označava koliko smplova je preostalo u baferu. Ova VI kontinualno čita i prikazuje podatke iz kanala 0 sve dok se ne pojavi greška ili dok ne kliknemo na **Stop** taster.



Analogni ulaz sa bazičnim kružnim baferom koji koristi srednje VI.

Primjeri drugih analognih ulaza sa kružnim baferom

Slijedeća lista u direktoriju **examples\daq\ analogin\analogin.llb** i u direktoriju **examples\daq\ analogin\strmdisk.llb** opisuje mnoge VI-jeve analognih ulaza sa kružnim baferom kao što su :

- **Cont Acq& Chart (buffered) VI** – demonstrira analogni ulaz sa kružnim baferom slično prethodnom primjeru , ali ova VI uključuje i druge ulaze na prednjem panelu.
- **Cont Acq& Graph (buffered) VI** – je slična sa prethodnom izuzev što VI prikazuje podatke u grafu valnog oblika.
- **Cont Acq to File (binary) VI** - prikuplja podatke putem analognog ulaza sa kružnim baferom i pohranjuje ih u specificirani fajl kao binarne podatke. Ovaj proces se često naziva "streaming na disk".
- **Cont Acq to File (scaled) VI** - je slična prethodnoj binarnoj VI , sa izuzetkom da ova Vi piše prikupljene podatke u fajl kao skalirane naponske signale a ne kao binarne vrijednosti.
- **Cont Acq to Spreadsheet File VI** - kontinualno očitava podatke koje LabView prikuplja u kružni bafer i pohranjuje ove podatke u specificirani fajl u spreadsheet formatu. Korisnik može gledati podatke pohranjene u spreadsheet fajl pomoću ove VI u bilo kojoj spreadsheet aplikaciji.

Kontrola akvizicije sa trigerima

Akvizicija pojedinačnih vrijednosti i valnih oblika opisana u prethodnom odjelu, starta u slučajnim trenucima vremena , relativno u odnosu na podatke. Medjutim, ponekada mi treba da postavimo start akvizicije u tačno određenom trenutku vremena. Jedan primjer za ovo ako bi željeli da testiramo odziv nekog uređaja

kojeg testiramo na impulsnu ulaz. Ovaj impulsni ulaz se onda može također koristiti da kaže DAQ uređaju da starta prikupljanje podataka. Bez ovog ulaza, mi moramo startati prikupljanje prije nego što primijenimo test impuls.

Ovo je neefikasno korištenje memorije računara i prostora na disku, pošto moramo alocirati i koristiti više memorije nego što je neophodno. Jedanput, podaci koje trebamo za analizu bit će negdje na početku bafera, drugi put na kraju.

Mi možemo startati akviziciju na bazi uslova ili stanja analognog ili digitalnog signala koristeći tehniku **trigerovanja**.

U opštem slučaju, triger je bilo koji događaj koji starta prikupljanje podataka. Postoje dva osnovna tipa trigerovanja:

Hardwareski i softwareski trigering.

U okviru LabView mi možemo koristiti softwareski trigering da startamo akviziciju ili koristiti vanjski uređaj da izvrši hardwaresko trigerovanje.

Hardwaresko trigerovanje

Hardwaresko trigerovanje dozvoljava nam da postavimo vrijeme starta akvizicije i prikupljamo podatke u poznatom trenutku vremena, relativno u odnosu na trigerski signal. Vanjski uređaji proizvode ove hardwareske trigerske signale. U LabView, mi specificiramo uslove trigerovanja koji moraju biti dostignuti prije nego što akvizicija započne. Kada su uslovi ispunjeni, akvizicija počinje.

Mi možemo također analizirati podatke prije trigera.

Postoje dva tipa hardwareskih trigera: **digitalni i analogni**.

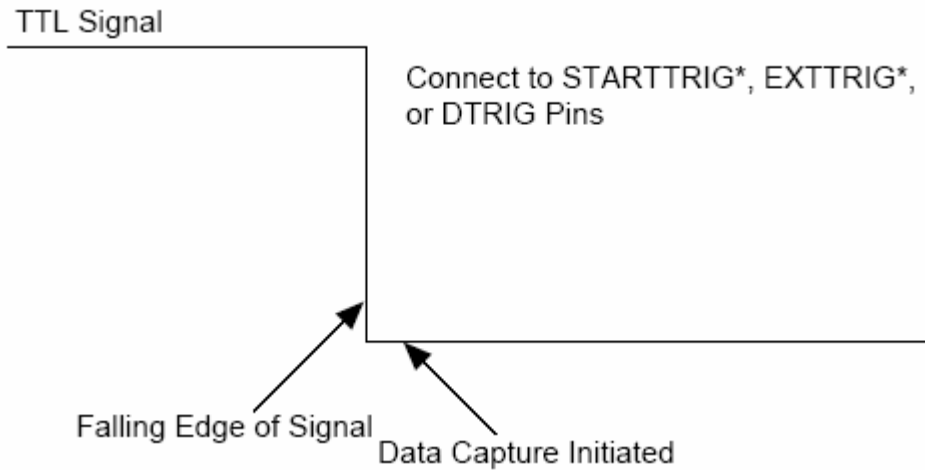
Digitalno trigerovanje

Digitalni triger je obično TTL signal koji ima dva diskretna nivoa: niski i visoki.

Kod promjene sa jedne vrijednosti na drugu, kreira se digitalna ivica. Postoje dva tipa ivica: **rastuća i opadajuća**. Mi možemo postaviti da analogna akvizicija starta kao rezultat rastuće ili opadajuće ivice od digitalnog trigerskog signala.

Na narednoj slici, akvizicija počinje nakon opadajuće ivice digitalnog trigerskog signala. Obično, digitalni trigerski signali su povezani na :

STARTTRIG*, EXTTRIG*, DTRIG, EXT TRIG IN, ili PFI pinove DAQ uređaja. Ako želimo da znamo koji pin ima naš uređaj, treba provjeriti u njegovom manualu. Pinovi STARTTRIG* i EXTTRIG* koji imaju zvjezdicu nakon njihovog imena, prihvataju opadajuću ivicu signala kao triger.

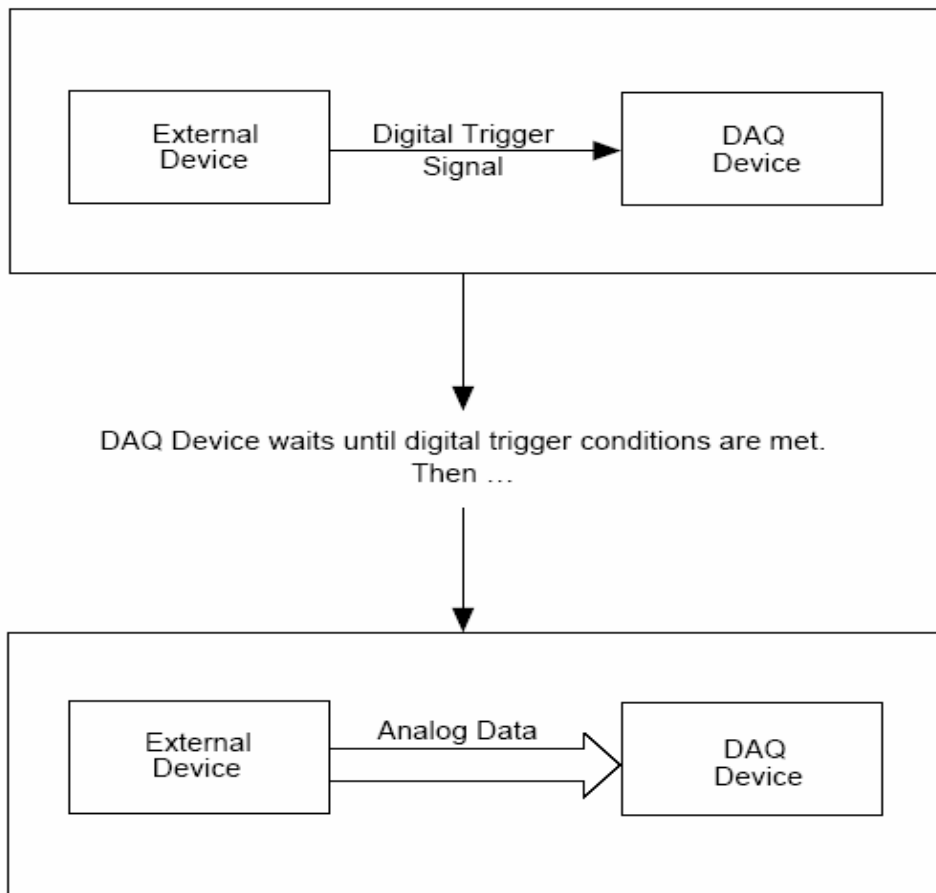


Dijagram digitalnog triger

Naredna slika pokazuje kako digitalni triger radi za prikupljanje podataka poslije trigerovanja (post triggered). U ovom primjeru, vanjski uređaj šalje triger, ili TTL signal , na DAQ uređaj. Čim DAQ uređaj primi signal, i trigerski uslovi su zadovoljeni, uređaj će početi sa prikupljanjem podataka.

Naprimjer sa NI DAQ uređajem NI 406X , impulsi za start triger se mogu generisati eksterno ili interno. Slijedeći izvori se mogu koristiti:

- softwareski start triger
- externi triger



Primjeri sa digitalnim trigerovanjem

Pogledati VI **Acquire N Scans Digital Trig** u `examples\daq\analogin\analogin.llb` kao primjer digitalnog trigerovanja.

Ova VI koristi srednje (intermediate VI) da izvrši baferovanu akviziciju, gdje LabView pohranjuje podatke u memorijski bafer za vrijeme akvizicije. Nakon što je akvizicija kompletna, VI vadi podatke iz memorijskog bafera i prikazuje ih na ekranu.

Mi moramo konfigurirati uređaj sa uslovima pri kojima će startati akviziciju.

Naprimjer, mi možemo izabrati **choose trigger type** da bude Boolean i postavljen na **START OR STOP TRIGGER**. Izabrati **START & STOP TRIGGER** samo onda kada imamo dva trigera: start i stop.

Nadalje, ako koristimo DAQ uređaj sa PFI linijama, (naprimjer E seriju NI DAQ modula), mi možemo specificirati uslove trigerskog signala u **trigger channel** kontrolnom elementu u okviru **analog chann & level** klastera.

Mi možemo prikupiti podatke i prije i poslije digitalnog trigerskog signala. Ako **pretrigger scans** je veći od 0, uređaj će prikupljati podatke prije nego što su trigerski uslovi zadovoljeni. On će zatim oduzeti vrijednost **pretrigger scans** od **number of scans to acquire** da odredi broj skanova koje će prikupiti nakon što su trigerski uslovi zadovoljeni.

Ako je **pretrigger scans** 0, tada ćemo prikupiti **number of scans to acquire** nakon što su trigerski uslovi zadovoljeni.

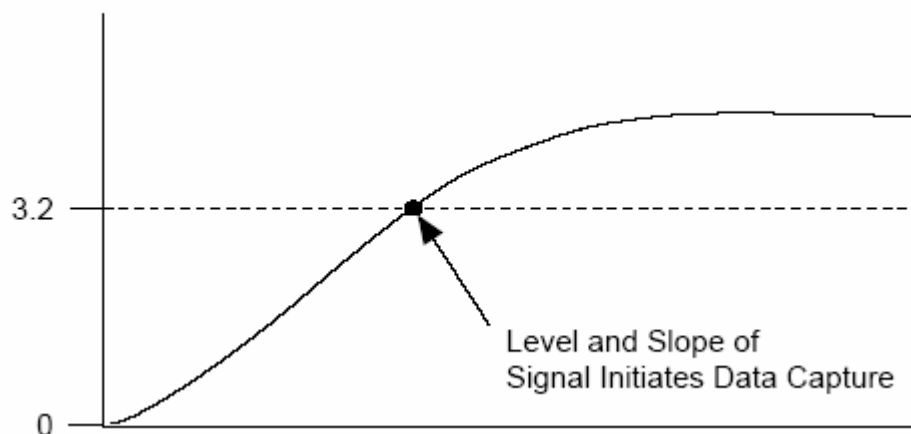
Prije nego što starta prikupljanje podataka, korisnik mora specificirati **trigger edge** ulaz , tj. da li se akvizicija trigeruje na rastuću ili opadajuću ivicu digitalnog trigerskog signala. On takodjer može specificirati vrijednost za **time limit**, tj. maksimalno vrijeme koje će Vi čekati za triger i zahtjevane podatke.

Primjer **Acquire N Scans Digital Trig VI** , drži podatke u memorijskom baferu sve dok uređaj ne kompletira akviziciju. Broj podataka koje treba prikupiti mora biti relativno mali da bi mogao da stane u memoriju. Ova VI gleda u i procesira podatke samo nakon završene akvizicije.

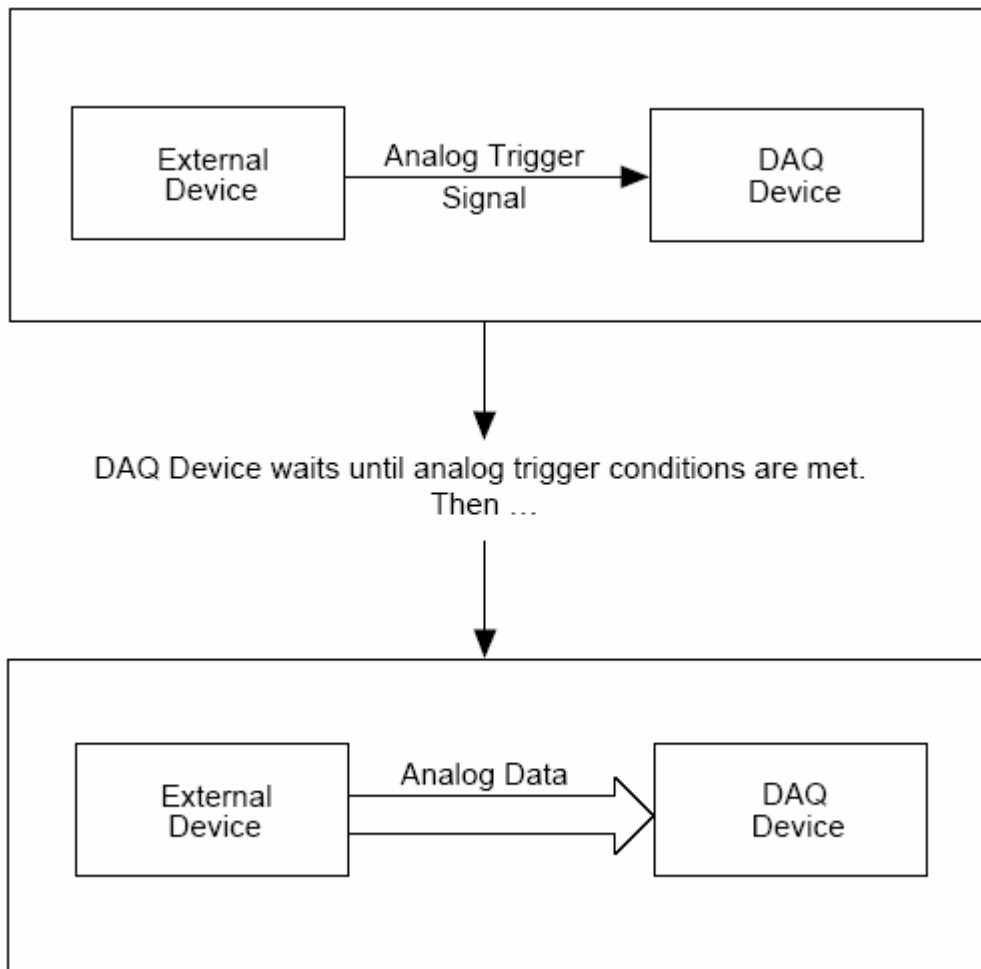
Analogno trigerovanje

Treba spojiti analogni trigerski signal na analogne ulazne kanale, na iste one kanale gdje spajamo i analogne ulaze. DAQ uređaj nadzire analogni trigerski kanal sve dok trigerski uslovi nisu zadovoljeni. Korisnik konfigurira DAQ uređaj da čeka na određeni uslov analognog ulaznog signala, kao što je naprimjer nivo signala ili nagib (rastući ili opadajući). Jedanput kada uređaj identificira uslov trigerovanja , on starta akviziciju.

Na narednoj slici analogni triger je setovan da starta akviziciju na rastući nagib signala, kada on dostigne iznos od 3.2 V.



Slijedeća slika ilustrira analogni triger za post-trigersku akviziciju koristeći vremensku liniju (timeline). Korisnik konfigurira DAQ hardware u LabView da počne uzimati podatke kada je dolazni signal na rastućoj ivici i kada amplituda dostigne 3.2 V.



Analogno trigerovanje DAQ uređaja

Primjeri sa analognim trigerovanjem

Pogledati primjer **Acquire N Scans Analog hardware Trig VI**, u **examples\daq\analogin\analogin-1lb**, za primjer analognog trigerovanja u Labview. Ova VI koristi srednje (intermediate VI-jeve) da izvrši baferovanu akviziciju, gdje se pohranjuju podatci za vrijeme trajanja akvizicije. Nakon što se akvizicija kompletira, VI vadi sve podatke iz memorijskog bafera i prikazuje ih.

U LabView, mi možemo prikupiti podatke i prije i nakon analognog trigerskog signala. Ako **pretrigger scans** je veće od 0, uređaj će prikupljati podatke prije trigerskog uslova. Zatim će oduzeti vrijednost **pretrigger scans** od **number of scans to acquire** da bi odredio broj skanova koje će prikupiti nakon što su trigerski uslovi zadovoljeni. Ako **pretrigger scans** je 0, tada **number of scans to acquire** je prikupljen nakon što su trigerski uslovi ispunjeni.

Potrebno je kompletirati slijedeće korake prije nego što počne prikupljanje podataka:

1. Specificirati u ulazu **trigger slope** , da li trigerovati akviziciju na rastuću ili opadajuću ivicu analognog trigerskog signala.
2. Unjeti **trigger channel** koji će se koristiti za spajanje analognog trigerskog signala.
3. Specificirati **trigger level** na trigerskom signalu potreban da počne akvizicija.

Nakon što korisnik specificira kanal na koji se dovodi trigerski signal, LabView čeka dok se specificirani uslovi ne steknu da starta baferovanu akviziciju.

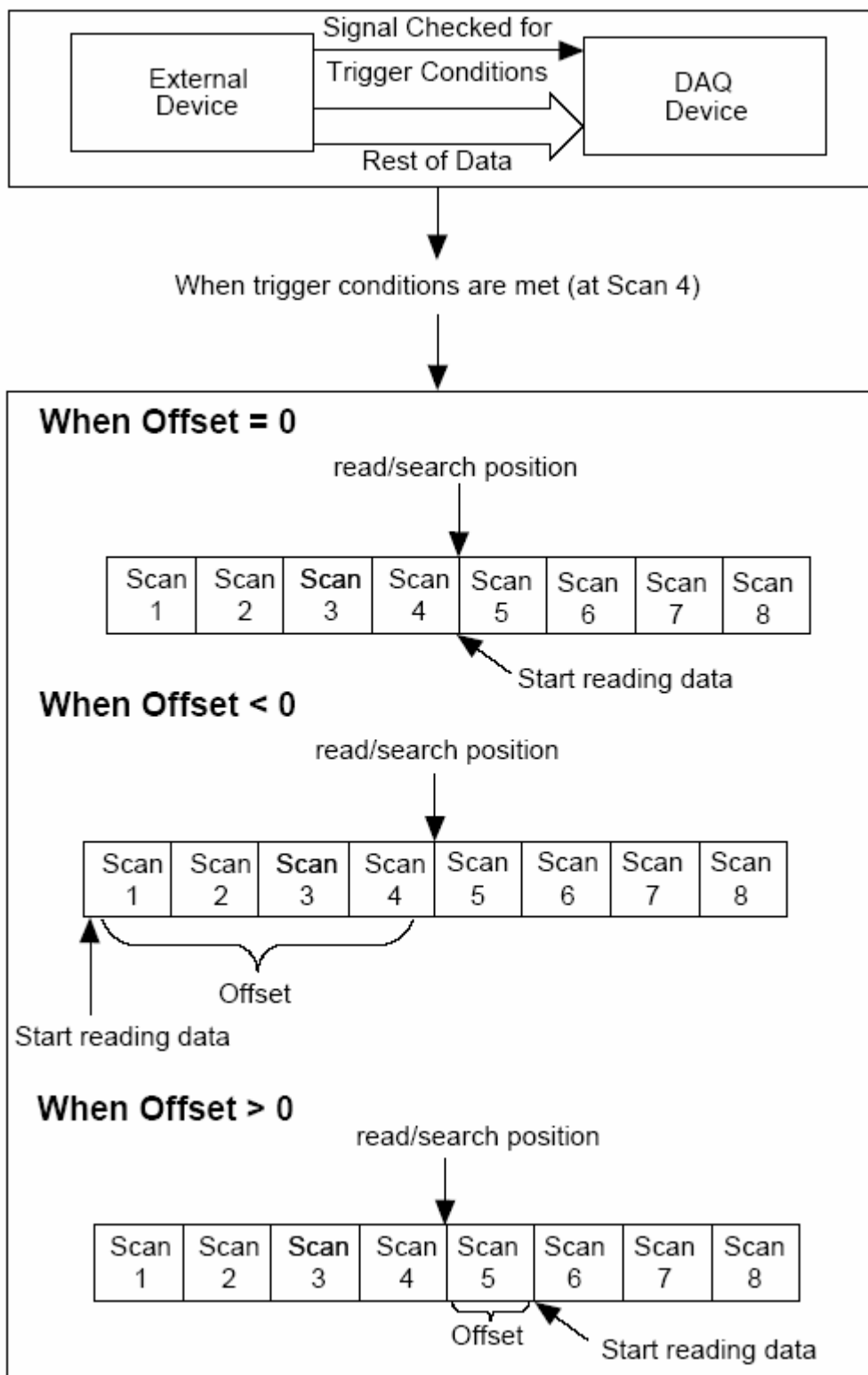
Pogledati primjer **Acquire & proc. N Scans -Trig VI** u **examples\daq\analogin\analogin-llb**, kako realizovati ako želimo da vidimo procesne informacije za vrijeme akvizicije.

Pogledati primjer **Acquire N-Multi-Analog Hardware Trig VI** u **examples\daq\analogin\analogin-llb** ako želimo da imamo višestruke analogne trigerske signale koji će startati višestruku akviziciju.

Softwaresko trigerovanje

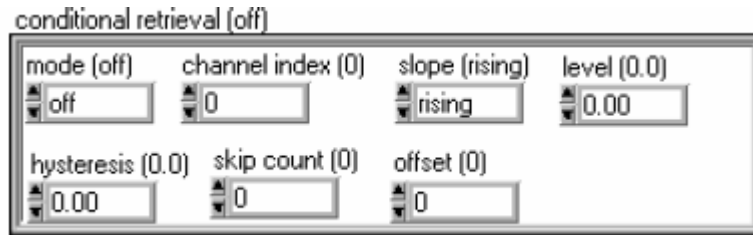
Kod softwareskog trigerovanja, mi možemo simulirati analogni triger koristeći software. Ova forma trigerovanja se često koristi u situacijama gdje hardwareski trigeri nisu raspoloživi. Drugo ime za softwareske trigerske signale, specifično analogne signale je uslovno prikupljanje (*conditional retrieval*). Sa uslovnim prikupljanjem, korisnik setuje DAQ uređaj da prikuplja podatke, ali uređaj ne vraća podatke u LabView sve dok podatci ne zadovolje uslove prikupljanja. LabView skanira ulazne podatke i izvršava poredjenje sa uslovima, ali ne pohranjuje podatke sve dok se ne zadovolje specifikacije.

Naredna slika pokazuje timeline događaja koji se tipično dešavaju kada se izvršava uslovno prikupljanje.



Read-search pozicioni poenter prolazi kroz bafer sve dok ne nadje lokaciju skanova gdje podatci zadovoljavaju uslove prikupljanja. Offset indicira lokaciju skana od koje Vi počinje čitanje podataka relativno u odnosu na read/search poziciju. Negativni offset indicira da mi trebamo pretriggerske podatke (tj. podatke prije uslova prikupljanja). Ako je offset veći od nule, tada mi trebamo posttriggerske podatke.

Klaster Uslovnog prikupljanja (conditional retrieval) iz AI Read Vi, specificira uslove analognog signala za prikupljanje, kao što je pokazano na narednoj slici:



AI Read VI klaster uslovnog prikupljanja

Kada prikupljamo podatke sa uslovnim prikupljanjem, podatci se pohranjuju u memorijski bafer, slično kao i kod aplikacija sa hardwarekim trigerovanjem. Nakon što VI starta , podatci se smještaju u bafer. Jedanput kada su uslovi za prikupljanje zadovoljeni, AI Read Vi pretražuje bafer za željenu informaciju. Kao i kod hardwareskog trigerovanja, korisnik specificira analogni kanal trigerskog signala, specificirajući **channel index**, tj. broj indeksa koji korespondira relativnom redoslijedu pojedinačnih kanala u listi kanala. Korisnik takodjer specificira **slope** (opadajući ili rastući) i **level** od trigerskog signala.

AI Read VI počinje traženje za uslove prikupljanja u baferu, na poziciji read/search. **Offset** vrijednost u klasteru **conditional retrieval** je gdje korisnik specificira skan lokacije od kojih VI počinje čitanje podataka relativno u odnosu na read/search poziciju. **Skip count** ulaz omogućava korisniku da specificira broj puta koliko trigerski uslovi su ispumjeni i propušteni prije nego se podatci akvizicije vrata LabView. **Hysteresis** ulaz kontroliše opseg koji se korsiti za uslove prikupljanja. Ovo je korisno kada ulazni signali imaju šuma koji može da neželjeno trigeruje početak akvizicije. Jedanput kada su **slope i level** uslovi na **channel index**-u zadoovljeni , read/search pozicija indicira lokaciju gdje su slovi prikupljanja zadovoljeni.

Primjeri sa uslovnim prikupljanjem

Acquire N Scans Analog Software Trig Vi primjer, koji se nalazi u **examples\daq\analogin\analogin-llb**, koristi srednje Vi-jeve.

Glavna razlika ovog softwareskog trigerovanja i primjera sa hardwarekim trigerovanjem je korištenje **conditional retrieval** ulaza za AI Read Vi. Korisnik setuje **trigger channel**, **trigger slope**, i **trigger level** na isti način za obadva metoda trigerovanja. Vrijednost **pretrigger scans** se negira i spaja na offset vrijednost od conditional retrieval klastera od AI Read Vi. Kada su uslovi trigerovanja zadovoljeni, VI vraća zahtjevani broj skanova.

Omogućavanje vanjskom izvoru da kontroliše brzinu akvizicije

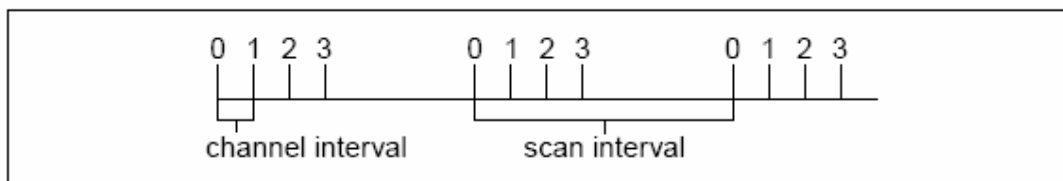
Tipično, DAQ uređaj koristi interne brojače da odredi brzinu prikupljanja podataka, ali ponekad korisnik može željeti da prikuplja podatke sa brzinom koja je sinhrona sa nekim posebnim signalima u sistemu. Naprimjer, možemo čitati temperaturne kanale svaki put kada se impuls pojavi, koji predstavlja signal da je pritisak prešao određenu vrijednost. U ovom slučaju, interni brojači su nedovoljni za ove potrebe. Potrebno je kontrolisati brzinu akvizicije sa nekim eksternim signalom.

Možemo porediti skanove na kanalima uzimajući snapshot napona na analognim ulaznim kanalima. Ako postavimo brzinu skaniranja na 10 skanova u sekundi, tada mi uzimamo 10 snapshotova svake sekunde svih kanala u listi kanala. U ovom slučaju, interni sat (scan clock) setuje brzinu skaniranja, koja kontroliše vremenski interval između skanova.

Nadalje, kod većine DAQ uređaja (izuzev kod onih koji sampliraju simultano), prelazak sa jednog kanala na slijedeći odnosno sa jednog sampla na drugi, zavisi od brzine kanalnog sata (channel clock). Dakle kanalni sat je sat koji kontroliše vremenski interval između individualnih samplova unutar skana, što znači da kanalni sat (channel clock) ide brže nego skan sat. (scan clock).

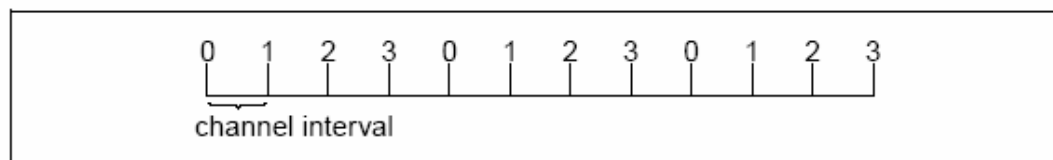
Što je brži kanalni sat, to su uži razmaci u vremenu u smplovima kanala unutar svakog skana, kao što se vidi na narednoj slici:

Opaska : Kod DAQ uređaja koji imaju i scan i kanalni sat, smanjenje brzine skaniranja ne mjenja brzinu kanalnog sata.



Kanalni i skan intervali koristeći kanalni klok

Neki DAQ uređaji nemaju skan satove, nego koriste skaniranje u krug (round-robin scanning). Naredna slika pokazuje primjer round-robin skaniranja.



Round-Robin skaniranje koristeći kanalni sat

Uređaji koji uvijek izvršavaju round-robin skaniranje uključuju , ali nisu ograničeni na slijedeće :

- PC-LPM-16
- PC-LPM-16PnP
- PC-516

- DAQCard-500
- DAQCard-516
- DAQCard-700
- Lab-LC
- NI 4060

Kada nema scan sata, kanalni sat se koristi da preklapa izmedju svakog kanala u jednakim intervalima vremena. Isto kašnjenje postoji izmedju svih uzoraka po kanalu, kao i izmedju posljednjeg kanala u skanu, i prvog kanala u slijedećem skanu. Kod DAQ modula sa skan i kanalnim satovima, round-robin skaniranje se pojavljuje ako onemogućimo skan sat na taj način što setujemo brzinu skaniranja na 0 i koristimo **interchannel delay** od AI Config VI , da kontrolišemo brzinu akvizicije.

LabView je scan-clock orijentisani program. Drugim riječima, kada izaberemo brzinu skaniranja, LabView automatski izabere kanalni sat za korisnika. LabView izabire najveću brzinu kanalnog sata koja dozvoljava adekvatno vrijeme smirenja (settling time) za ADC.

LabView dodaje 10 μ s na medjukanalno kašnjenje da kompenzira za faktore koji nisu uzeti u obzir.

Korisnik može setovati svoju brzinu po kanalu (channel clock rate) sa interchannel delay ulazom u AI Config VI, koja poziva Advanced AI Clock Config VI da konfigurira kanalni sat. Najjednostavniji metod da se postavi medjukanalno kašnjenje je da se postepeno povećava kašnjenje, ili period sata, sve dok se ne počnu pojavljivati konzistentni podaci sa podacima od prethodnih setinga kašnjenja.

Mi možemo utvrditi koliko je medjukanalno kašnjenje na taj način da izvršavamo AI Clock Config VI, za kanalni sat bez specificiranja frekvencije.

Vanjska kontrola kanalnog sata

Postoje situacije kada korisnik ima potrebu da eksterno kontroliše kanalni sat. Kanalna brzina je ona brzina kod koje se realizuje analogna konverzija. Na primjer, predpostavimo da trebamo da znamo nivo jakosti signala na ulazu, svaki put kada infracrveni senzor pošalje impuls. Većina DAQ uređaja ima EXTCONV* pin ili PFI pin na I/O konektoru da bi se omogućilo korisniku da priključi svoj kanalni sat. Kod NI 406x serije DAQ uređaja, treba koristiti EXTRIG ulazni pin. Ovaj vanjski signal mora biti TTL nivo signala.

Pogledati primjer **Acquire N Scans -ExtChanClk VI** , u **examples\daq\analogin\analogin-11b** gdje se koristi akvizicija sa vanjski kontrolisanim kanalnim satom. Ova VI uključuje AI Clock Config VI i vanjski izvor sata koji je spojen na I/O konektor.

Korisnik može omogućiti vanjske konverzije pozivajući napredni VI AI Clock Config. Ovaj VI se poziva od strane AI Config VI, koji normalno setuje interno kanalno kašnjenje automatski ili ručno, sa **interchannel delay** kontrolnim parametrom.

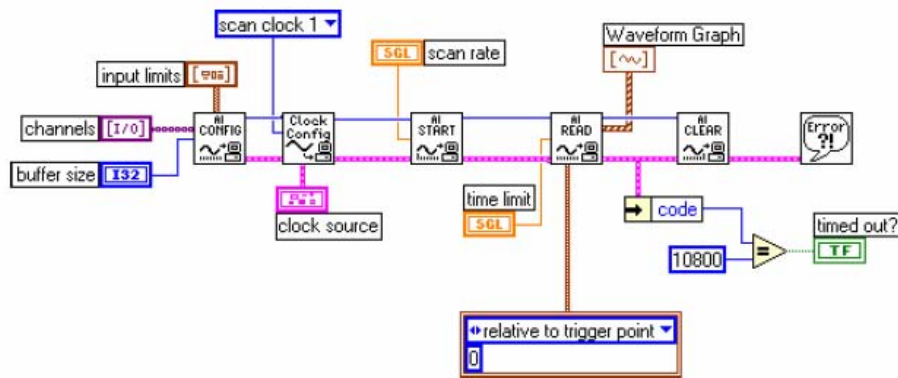
Medjutim, pozivajući AI Clock Config VI nakon AI Config VI resetuje kanalni sat tako da on dolazi iz vanjskog izvora za vanjsku konverziju.

Primjetimo takodjer da je scan sat setovan na 0 da se onemogući, dozvoljavajući kanalnom satu da kontroliše brzinu akvizicije.

Na većini uređaja, eksterna konverzija se javlja na opadajućoj ivici EXTCONV* linije. Konsultovati hardwareski manual za tajming dijagrame. Na uređajima sa PFI linijama (kao što su NI moduli E serije), može se postaviti **Clock Source Code** ulaz od AI Clock Config VI na PFI pin sa bilo opadajućom ili rastućom ivicom ili koristeći default PFI2/Convert* pin gdje konverzije se javljaju na opadajuću ivicu.

Pošto LabView određuje dužinu vremena prije nego AI Read VI tajmira na bazi **interchannel delay** i **scan clock rate**, možda će biti potrebno forsirati vremenski limit za AI Read VI, kao što je to pokazano u primjeru Acquire N Scans-ExtChanClk VI.

Naredna slika pokazuje primjer korištenja vanjskog scan sata da se izvrši baferovana akvizicija.



Prikupljanje podataka sa vanjskim scan satom

Vanjska kontrola scan sata

Vanjska kontrola scan sata može biti korisnija nego vanjska kontrola kanalnog sata (channel clock), ako sampliramo višestruke kanale, ali može biti malo teže da se odredi gdje ga priključiti, pošto nema na I/O konektoru pina sa labelom **ExtScanClock**, kao što ima pin s labelom **EXTCONV***.

Opaska: Neki MIO moduli NI, imaju izlaz na I/O konektoru koji je labeliran kao SCANCLK, koji se koristi za vanjsko multipleksiranje i nije analogni ulaz scan clocka. Ovo ne može biti korišteno kao ulaz.

Odgovarajući pin na koji se može dovesti vanjski scan clock može se naći u narednoj tabeli:

Ulazni pinovi za Vanjski scan clock

Uredjaj	Ulazni pin za externi scan clock
Svi uredjaji E serije (NI)	Svaki PFI pin (default : PF17/ STARTSCAN)
Lab.PC+ , 1200 uredjaji (NI)	OUT B1

Opaska : Neki uredjaji nemaju interne scan clockove i zbog toga ne podržavaju ni vanjske scan clockove.

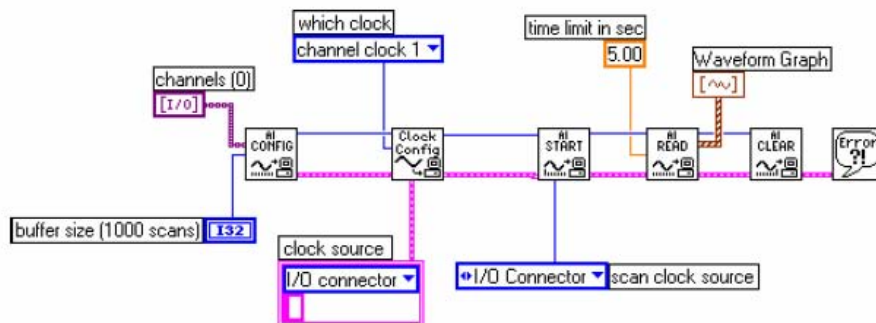
Nakon povezivanja vanjskog sata na korektan pin, treba setovati vanjski scan clock i u softwareu. Pogledati u primjeru Acquire N Scans-ExtScanClk Vi, u **examples\daq\analogin\analogin-11b**, da se vidi kako setovati za vanjski clock u softwareu.

Dvije napredne VI, AI Clock Config VI i AI Control VI , se koriste umjesto srednje AI Start VI. Ovo dozvoljava pristup do **clock source** ulaza. Ovo je neophodno pošto dozvoljava pristup do **clock source string**, koji se koristi da identificira PFI pin koji će se koristiti za scan clock za module E Serije.

Clock source takodjer uključuje clock source code (na prednjem panelu modula) , koji setuje I/O konektor. Parametar 0.0 ožičen na ulaz Clock Config VI će onemogućiti interni clock.

Vanjska kontrola scan i kanalnih satova

Korisnik može kontrolisati simultano scan i kanalne satove. Medjutim, potrebno je osigurati da se koristi odgovarajući tajming. Naredna slika demonstrira kako treba setovati aplikaciju da kontroliše obadva sata.



Simultano kontrolisanje scan i kanalnog sata

POGLAVLJE 16

KORIŠTENJE BRZE FOURIEROVE TRANSFORMACIJE (FFT) SA HP 54600 OSCILOSKOPIMA

HP 54600 serija osciloskopa sa HP 54657A ili HP 54658A instaliranim measurement storage modulom, imaju sposobnost izvršenja analize u frekventnom domenu na valnom obliku iz vremenskog domena, koristejem brze FFT transformacije (Fast Fourier transformation). Ovaj tekst daje kratak pregled Fourierove teorije, i jedinstveno ponašanje FFT.

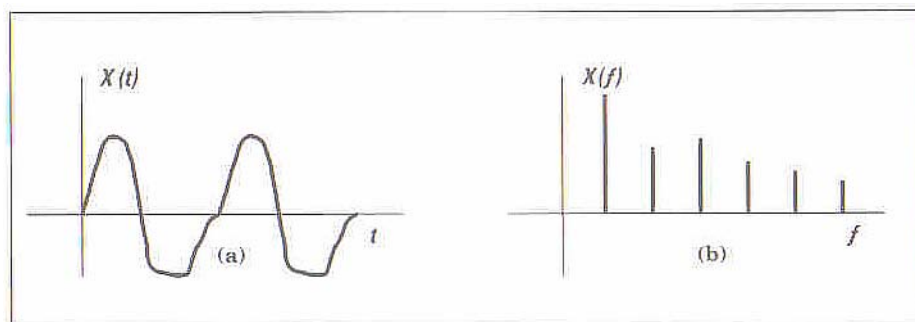
Fourierova teorija

Normalno, kada se signal mjeri sa osciloskomom ili VI, on se vizualizira u vremenskom domenu (slika 1 a).

Ali kada je frekventni sadržaj signala od interesa, ima smisla posmatrati signal u frekventnom domenu. U frekventnom domenu , vertikalna osa je napon kao i u vremenskom domenu, ali horizontalna osa je frekvencija (vidjeti sliku 1b).

Displej u frekventnom domenu pokazuje koliko mnogo energije signala je prisutno na svakoj frekvenciji.

Za jednostavni signal kao što je to sinusni, displej u frekventnom domenu ne daje mnogo više dopunske informacije. Medjutim, za kompleksnije signale, frekventni sadržaj je teško otkriti u vremenskom domenu i frekventni domen daje mnogo korisniji pogled na signal.

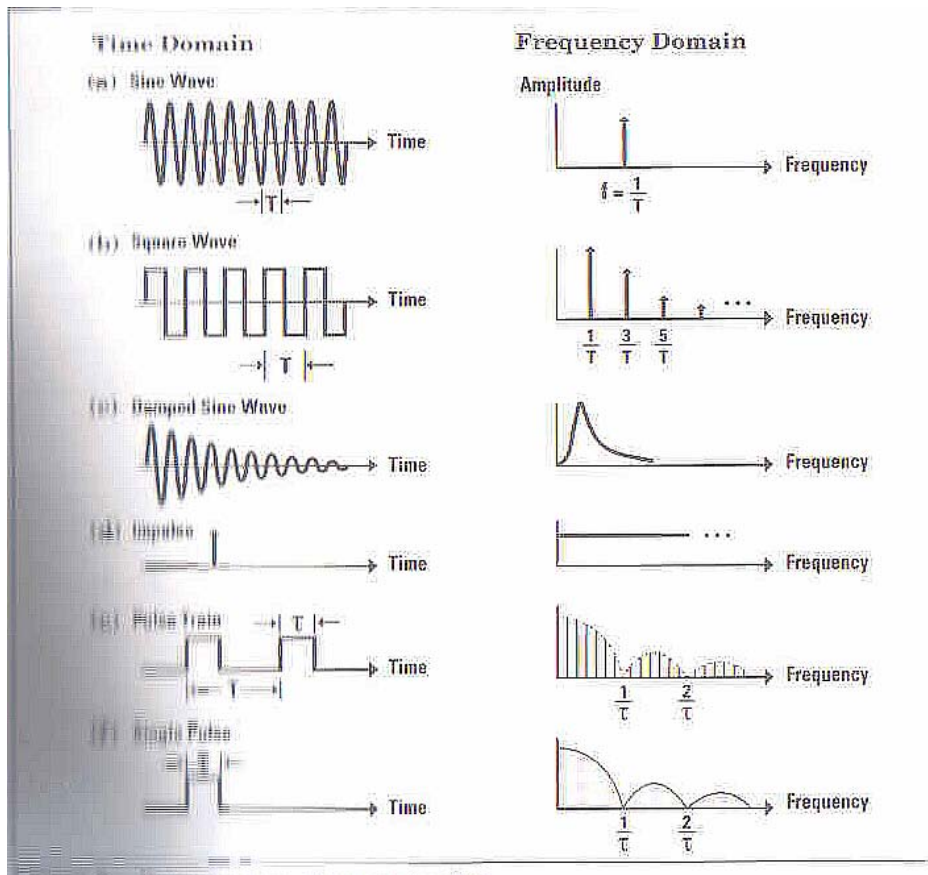


Slika 1

Fourierova teorija (uključujući i Fourierovu seriju i Fourierovu transformaciju) uspostavlja matematsku relaciju izmedju vremenskog domena i frekventnog domena. Fourierova transformacija je data sa :

$$V(f) = \int_{-\infty}^{\infty} v(t) e^{-j2\pi ft} dt$$

Neki tipični signali predstavljeni u vremenskom i frekventnom domenu su pokazani na narednoj slici br. 2:



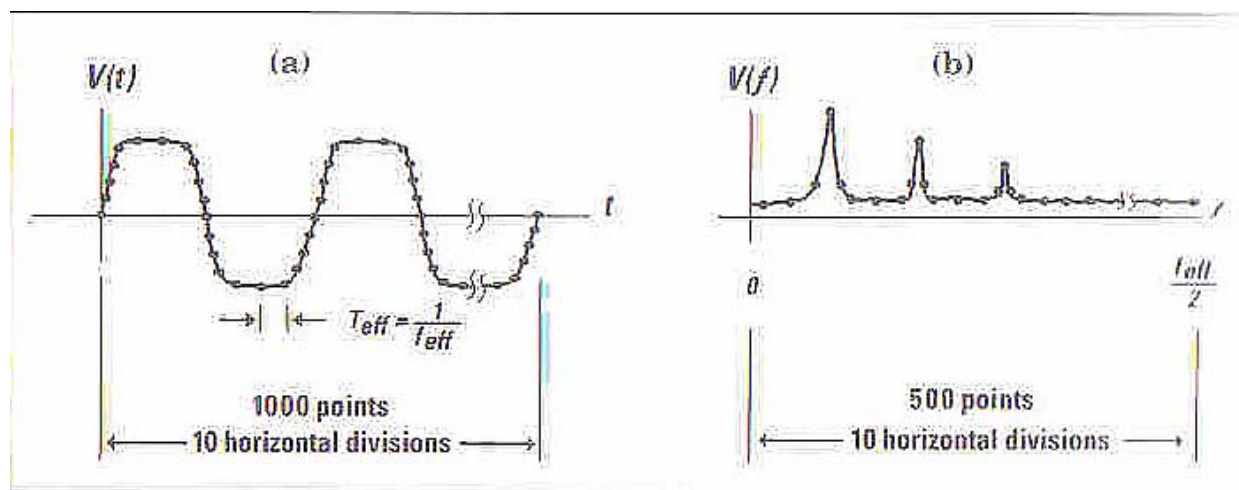
Primjeri frekventnog spektra
Slika br. 2

Brza Fourierova transformacija

Diskretna (ili digitizirana) verzija Fourierove transformacije se naziva diskretna Fourierova transformacija (DFT – discrete Fourier transform). Ova transformacija uzima digitizirane podatke iz vremenskog domena, i računa njihovu predstavu u frekventnom domenu. DFT nam omogućava da izračunamo predstavu u frekventnom domenu signala iz realnog vremenskog domena.

HP 54600 osciloscope sa modulom za mjerenje i pohranjivanje (measurement/storage module) kao i VI, koristi algoritam FFT (fast Fourier transform), da izračuna DFT. FFT i DFT proizvode isti rezultat pa se najčešće obadva nazivaju istim imenom FFT.

HP 54600 serija osciloscopa normalno digitizira valni oblik u vremenskom domenu i pohranjuje ga kao zapis od 4000 tačaka. FFT funkcija koristi 1000 od ovih tačaka (tj. svaku četvrtu tačku) da proizvede 500 tačaka prikaza u frekventnom domenu. Ovaj displej u frekventnom domenu se prostire od frekvencije 0 do $f_{\text{eff}}/2$ gdje f_{eff} je efektivna brzina uzorkovanja vremenskog rekorda (vidjeti slijedeću sliku 3 a).



Slika 3

- a) – Valni oblik u vremenskom domenu , uzorkovan
 b) – rezultirajući zapis u frekventnom domenu, koristeći FFT

Efektivna brzina uzorkovanja je recipročna vrijednost vremena između uzoraka i zavisi od vrijeme/podjela (time/div) podešenja na osciloskopu. Za HP 54600 seriju osciloskopa, efektivna brzina uzorkovanja je data sa :

$$f_{\text{eff}} = \frac{\text{record length}}{10 * \text{time/div}} = \frac{1000}{10 * \text{time/div}} = \frac{100}{\text{time/div}}$$

Primjetimo da efektivna brzina uzorkovanja za FFT može biti mnogo veće nego što je maksimalna brzina uzorkovanja za osciloskop. Tako npr. maksimalna brzina uzorkovanja za ovaj tip osciloskopa je 20 MHz , ali tehnika poznata kao **random repetitive samlong technique**, postavlja uzorke tako precizno u vremenu tako da brzina uzorkovanja koju vidi FFT može biti i do 20 GHz.

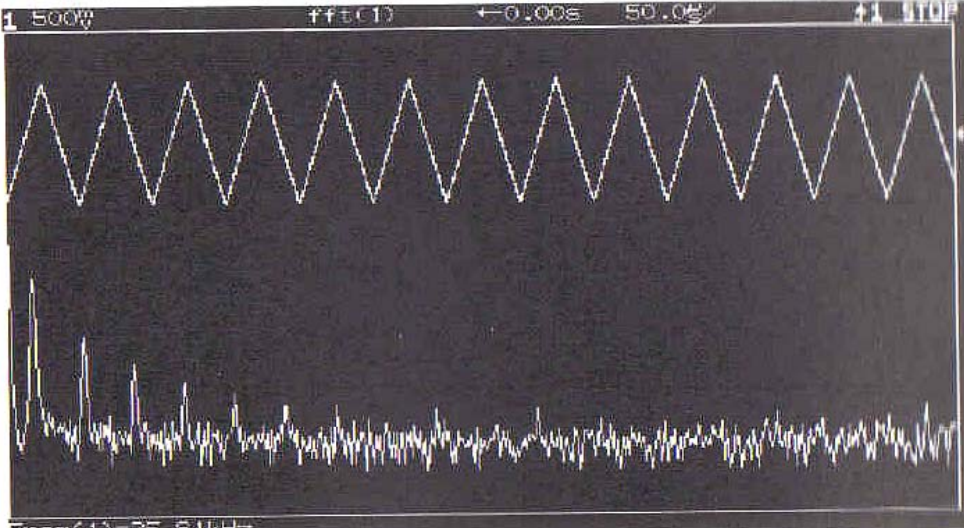
Default displej frekventnog domena pokriva normalni frekventni opseg od 0 do $f_{\text{eff}}/2$.

Aliasing (udvajanje)

Frekvencija $f = f_{\text{eff}}/2$ se također naziva i frekvencija previjanja (folding frequency). Frekvencije koje bi se normalno pojavile iznad $f_{\text{eff}}/2$ (i dakle van korisnog opsega FFT), se previjaju natrag u prikaz frekventnog domena. Ove nepoželjne komponente se zovu aliases (udvajajuće), pošto se one greškom pojavljuju kao aliasi od druge frekvencije.

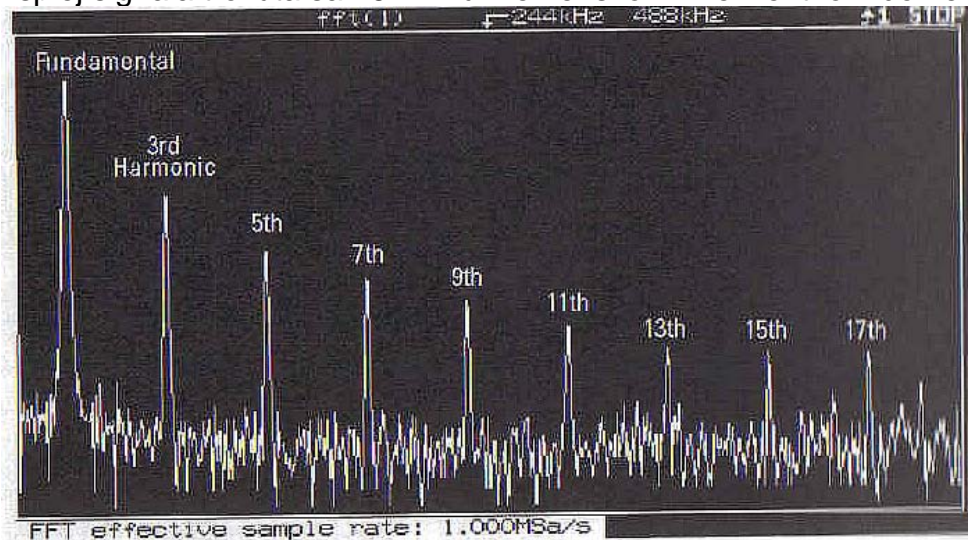
Aliasing će se izbjeći ako je efektivna brzina sampliranja veća od dvostrukog opsega signala koji se mjeri.

Frekventni sadržaj trouglastog valnog oblika uključuje osnovnu frekvenciju i veliki broj neparnih harmonika sa svakim novim harmonikom manjim u amplitudi od prethodnog. Na slici 4a, 26 kHz trouglasti valni oblik je prikazan u vremenskom domenu i u frekventnom domenu. Slika 4b pokazuje samo frekventni domen. Krajnja lijeva linija je osnovna frekvencija. Slijedeća značajna spektralna linija je treći harmonik. Nakon toga je peti harmonik, itd. Promjetimo da su veći harmonici sa manjom amplitudom i 17-i harmonik se tek nazire iznad FFT nivoa šuma. Frekvencija 17-og harmonika je $17 \times 26 \text{ kHz} = 442 \text{ kHz}$, što je unutar previjajuće frekvencije od $f_{\text{eff}}/2$ (500 kSa/s) na slici 4b. Zbog toga nikakav aliasing se neće javiti.



Slika 4a

Displej signala trokuta sa 26 kHz u vremenskom i frekventnom domenu



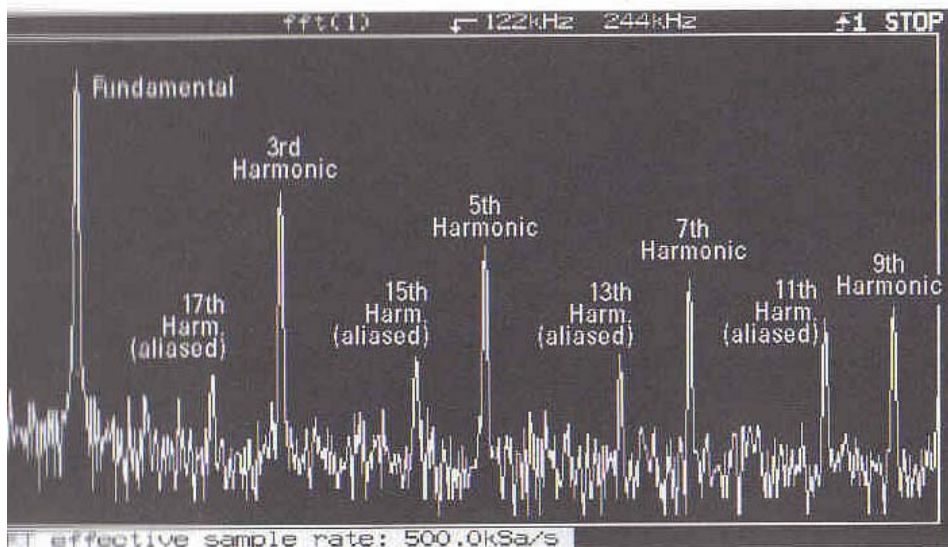
Slika 4b

frekventni spektar trouglastog signala

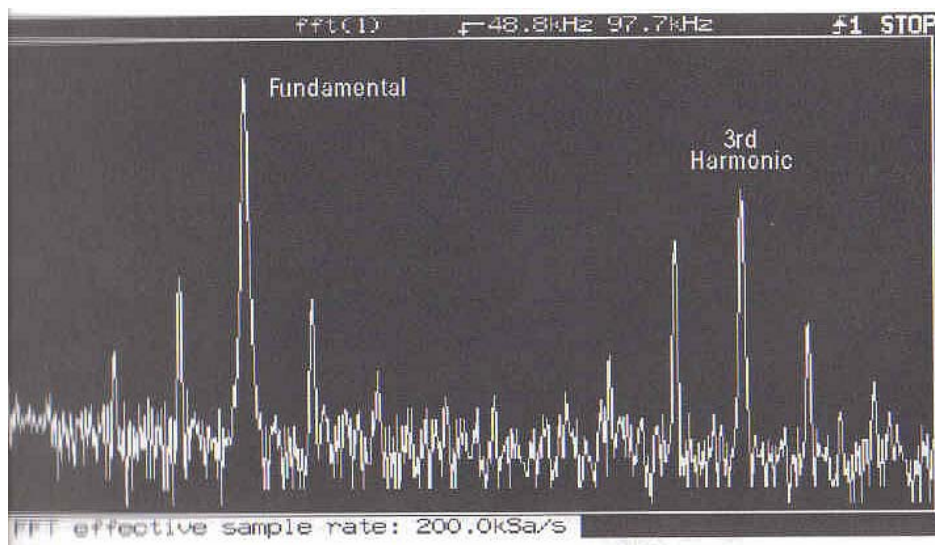
Slika 4c pokazuje FFT od istog valnog oblika sa time/div podešenim na 500 kSa/sec i frekvencijom previjanja od 250 kSa/sec.

Sada će gornji harmonici trouglastog valnog oblika prevazići frekvenciju previjanja i pojaviti se kao aliasi na displeju. Slika 4d pokazuje FFT istog trouglastog

signala, ali sa još nižom efektivnom brzinom sampliranja od samo 200 kSa/sec., i frekvencijom previjanja od 100 kSa/sec. Ovaj frekventni plot će biti značajno aliasiran.



Slika 4c



Slika 4d

Sa još nižom efektivnom brzinom samplovanja, samo osnovni i treći harmonik nisu aliasirani.

Često su efekti aliasiranja očigledni, naročito ako korisnik ima naku predstavu o frekventnom sadržaju signala kojeg analizira. Tako se spektralne linije mogu pojaviti i na mjestima gdje ne egzistiraju frekventne komponente.

Aliasirane frekventne komponente često mogu zavarati i nisu poželjne u mjerenjima. Signali koji su ograničenog frekventnog sadržaja, mogu se posmatrati bez aliasinga ukoliko osiguramo da je efektivna brzina uzorkovanja dovoljno velika.

Ako signal nije inherentno ograničenog frekventnog domena, niskopropusni filter se može koristiti da ograniči frekventni sadržaj signala koji se mjeri. (vidjeti sliku 5).

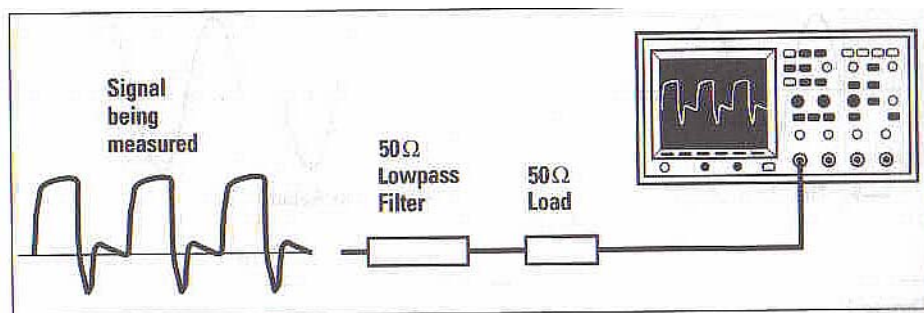


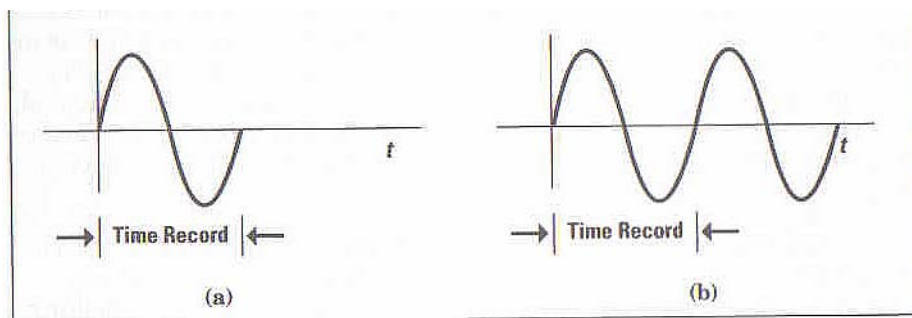
Figure 5

A lowpass filter can be used to band limit the signal, avoiding aliasing.

Slika 5

Curenje (leakage)

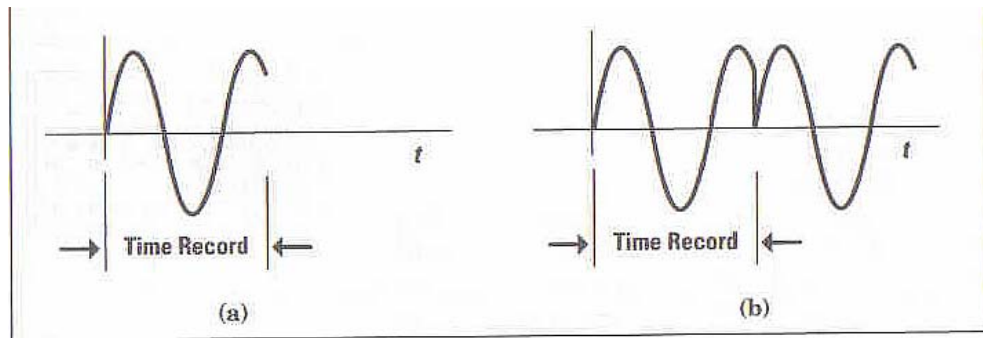
FFT radi na konačnoj dužini vremenskog intervala sa ciljem da se procjeni Fourierova transformacija, koja uzima vremenski interval nad beskonačnim intervalom. FFT radi nad rekordom sa konačnim vremenom, ali ima efekat ponavljanja ovog konačnog rekorda , nad cijelim beskonačnim intervalom vremena (vidjeti sliku 6). Sa valnim oblikom kao na slici 6a, zapis podataka u konačnom vremenu predstavlja stvarni valni oblik vrlo dobro, tako da će FFT rezultat aproksimirati vrlo dobro Fourierov integral.



Slika 6

- a) valni oblik koji tačno fituje jedan vremenski zapis
- b) kada se replicira , tranzijenti neće biti uvedeni

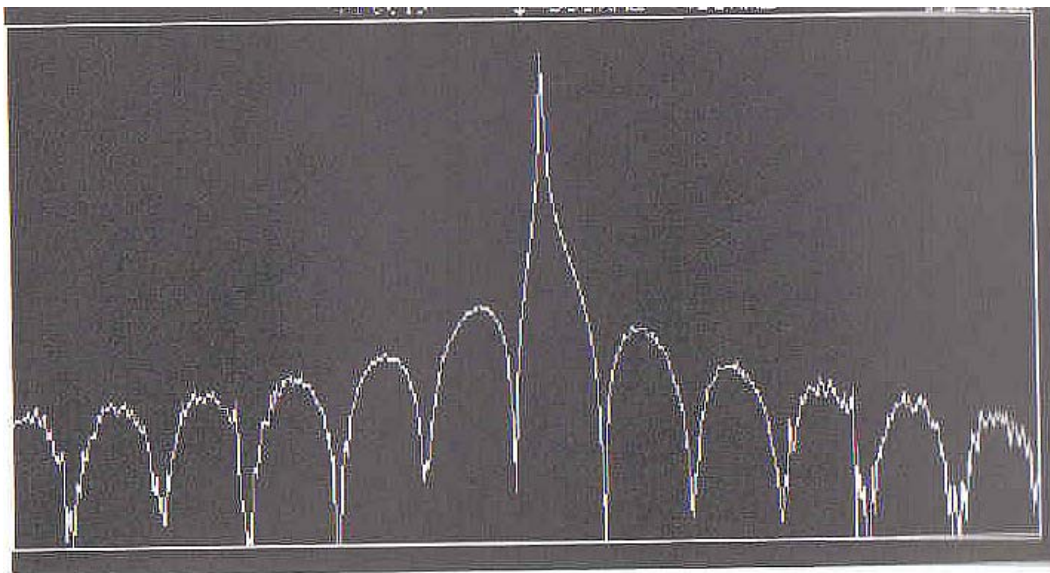
Medjutim , oblik i faza valnog oblika mogu biti takvi da se unese tranzijent kada se valni oblik replicira za sva ostala vremena izvan osnovnog intervala FFT transformacije, kao što je to pokazano na narednoj slici 7. U ovom slučaju FFT spektar nije dobra aproksimacija za Fourierovu transformaciju.



Slika 7

- a) Valni oblik koji ne fituje tačno u vremenski zapis
 b) kada se replicira, uvode se vrlo oštri tranzijenti, prouzrokujući curenje (leakage) u frekventnom domenu.

Ovaj efekat poznat kao curenje (LEAKAGE), je vrlo uočljiv u frekventnom domenu. Tranzijent uzrokuje da spektralna linija (koja bi trebala biti tanka i male debljine) se raširuje kako se to vidi na slijedećoj slici br. 8.

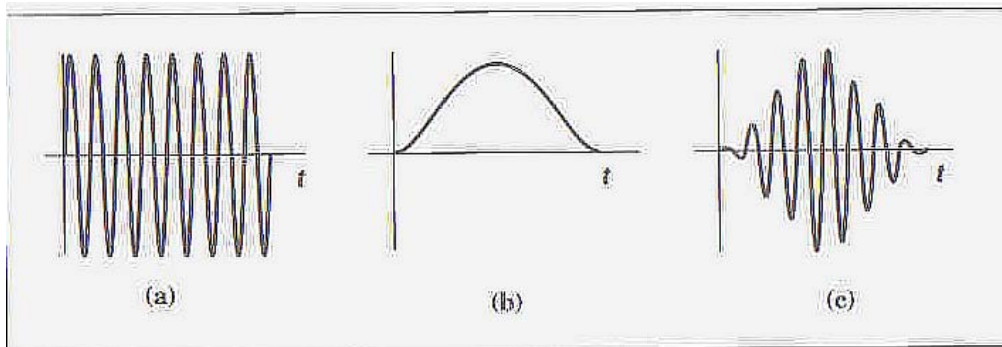


Slika br. 8

Riješenje za problem curenja je da prisilimo valni oblik da ide na nulu na krajevima intervala vremenskog zapisa (recorda), tako da neće postojati tranzijenti kada se replicira vremenski rekord. Ovo se postiže množeći vremenski zapis sa WINDOW funkcijom (funkcijom prozora). Naravno, prozor modificira vremenski zapis i proizvešće i svoj vlastiti efekat u frekventnom domenu. Za korektno dizajniran prozor, efekat u frekventnom domenu je značajno poboljšanje u odnosu na to kada se prozor uopšte ne koristi.

Četiri funkcije prozora su na raspolaganju u HP 54600 seriji osciloskopa: Hanning, Flattop, Rectangular (četvrtasti) i eksponencijalni.

Hanningov prozor daje glatki prelaz do nule kako se približuju krajevi zapisa signala. Slika 9a pokazuje sinusoidu u vremenskom domenu, dok Slika 9b pokazuje Hanningov prozor koji će biti primjenjen na podatke vremenskog domena. Uprozoreni zapis u vremenskom domenu je pokazan na slici 9c. Mada se ukupni izgled signala u vremenskom domenu promjenio, frekventni sadržaj ostaje u suštini nepromjenjen. Spektralna linija pridružena sa sinusoidom se malo raspršuje u frekventnom domenu kao što je to pokazano na slici 10. (Slika 10 je raširena po frekventnoj osi da jasno pokaže oblik prozora u frekventnom domenu).

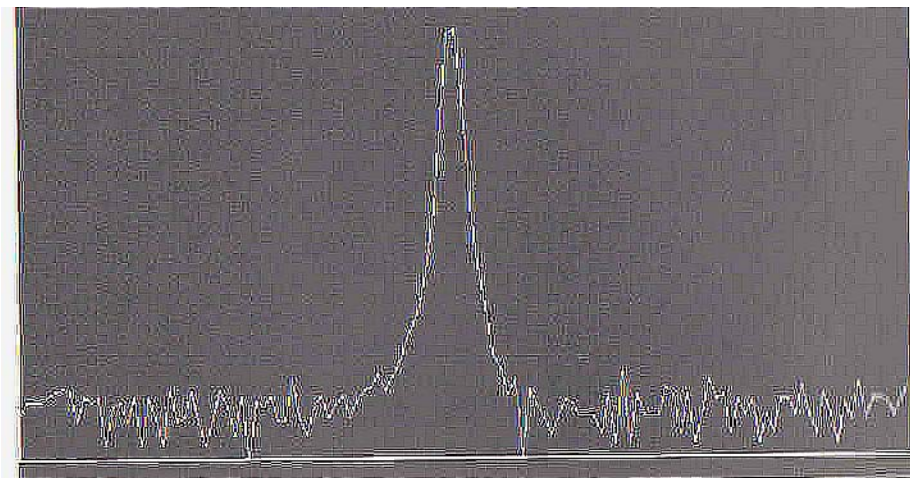


Slika 9

- a) originalni vremenski zapis
- b) Hanningov prozor
- c) Uprozoren vremenski zapis

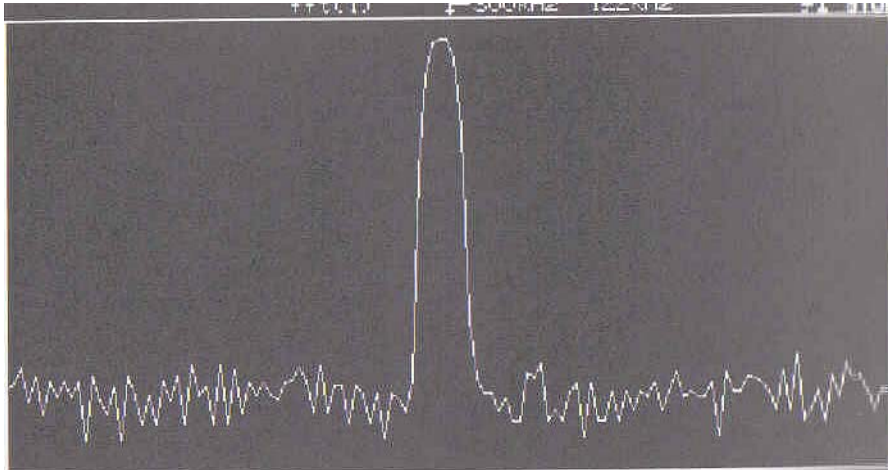
Oblik prozora je kompromis između amplitudne tačnosti i frekventne rezolucije. Hanningov prozor, u poredjenju sa drugim tipovima prozora, obezbjedjuje dobru frekventnu rezoluciju po cijenu nešto manje tačnosti amplitude.

FLATTOP (ravni vrh) prozor, ima deblju (i ravniju) karakteristiku u ftekvntnom domenu, kao što je to pokazano na slici 11. (ovdje ponovo slika je ekspanđirana po frekventnoj osi da jasno pokaže efekat prozora). Ravniji vrh na spektralnoj liniji u frekventnom domenu proizvodi poboljšanu amplitudnu tačnost, ali po cijenu lošije frekventne razlučivosti (kada se poredi sa Hanningovim prozorom).



Slika 10

Hanningov prozor ima relativno uzak oblik u frekventnom domenu



Slika 11

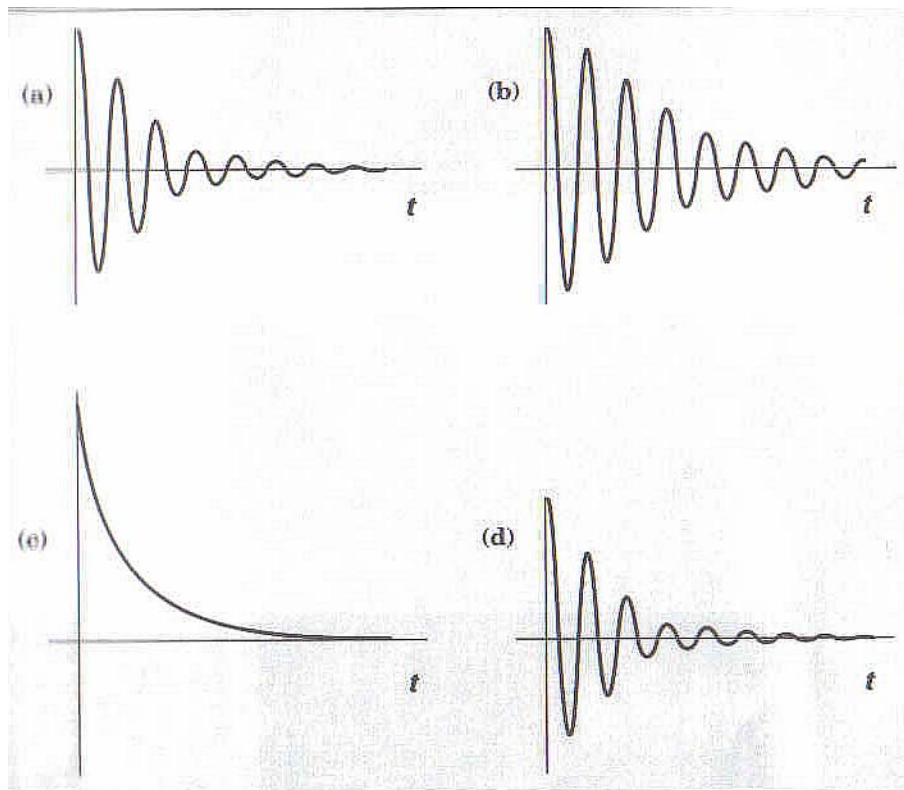
Prozor sa ravnim vrhom ima širi zaravnjeni oblik u frekventnom domenu

Četvrtasti prozor (takodjer pogrešno nazivan i uniformni prozor), je teško reći da je uopšte neki prozor, svi samplovi ostaju nepromjenjeni nakon propuštanja. Mada uniformni prozor ima potencijala za ozbiljne probleme curenja, u nekim slučajevima valni oblik ima u vremenskom zapisu istu vrijednost na oba kraja zapisa, i na taj način eliminira tranzijente koje FFT uvodi.

Takvi valni oblici se nazivaju **samo upozorenje (SELF WINDOWING)**. Valni oblici kao sinusni prolom (burst), impulsi, i opadajuće sinusoide mogu biti samouprozoravajući. (self-windowing).

Tipični tranzijentni odziv je pokazan na narednoj slici 12a. Kako je prikazano, valni oblik je samo-upozoravajući, pošto iščezava unutar dužine vremena zapisa, smanjujući probleme curenja.

Ako valni oblik ne disipira unutar vremenskog zapisa (kao na promjer na slici 12b), tada neka vrsta prozora treba biti korištena. Ako se na valni oblik primjeni prozor tipa Hanninga, početni dio vremenskog zapisa će biti prisiljen da počinje od nule. Ovo je tačna lokacija gdje se nalazi veći dio energije tranzijenta, tako da prozor kao što je uniformni bi bio neadekvatan.



Slika 12

- a) Tranzijentni odziv koji je samouprozoravajući
- b) tranzijentni odziv koji zahtjeva uprozorenje
- c) Eksponencijalni prozor
- d) Uprozoren tranzijentni odziv

Prozor sa opadajućim ekponencijalnim odzivom je koristan u ovakvoj situaciji. Početni dio valnog oblika nije poremećen, ali zadnji dio je prisiljen da ide na nulu. Još uvijek može postojati tranzijent na početku vremenskog zapisa, ali ovaj tranzijent nije uveden od strane FFT. To je ustvari tranzijent koji je rezultat mjerenja. Slika 12c pokazuje eksponencijalni prozor a slika 12d pokazuje rezultirajuću funkciju u vremenskom domenu kada se eksponencijalni prozor primjeni na sliku 12b. Eksponencijalni prozor je neodgovarajući za mjerenje bilo čega drugog osim tranzijentnih valnih oblika.

Selekcija prozora.

Većina mjerenja će zahtjevati korištenje prozora kao što je Hanningov ili prozor sa ravnim vrhom (flattop). Ovo su adekvatni prozori za tipična mjerenja u svrhu spektralne analize. Izabirući između ova dva prozora je kompromis između rezolucije frekvencije i tačnosti amplitude. Koristeći vremenski domen da objasnimo zašto se curenje javlja, treba se sada prebaciti u razmišljanje u frekventnom domenu. Što je uži propusni opseg domenskog filtera prozora frekvencija, to analizator može bolje da razluči između dvije spektralne linije na bliskom odstojanju.

Istovremeno, amplituda spektralne linije biće manje izvjesna. Suprotno, ukoliko je frekventni domen filtera širi i ravniji, mjerenje amplitude će biti tačnije i naravno, frekventna razlučivost će biti smanjena.

Izabirući između takve dvije funkcije prozora je ustvari izbor između oblika filtera u frekventnom domenu.

Pravougli i eksponencijalni prozori trebaju biti posmatrani kao prozori za specijalne situacije.

Pravougaoni prozor se koristi tamo gdje može biti garantirano da neće biti efekata curenja. Eksponencijalni prozor treba koristiti tamo gdje je ulazni signal tranzijentni signal.

Rad sa HP54600 serijom osciloskopa (HP 54645A) i modulom za mjerenje i pohranjivanje (HP 54657A)

Dodajući modul za mjerenje i pohranjivanje (measurement + storage) na osciloskop HP 54645A dodaje osciloskopu dodatne matematske mogućnosti obrade signala uključujući i FFT. Ova se funkcija pojavljuje se na softkey meniju sa +/- tasterom (matematski taster).

Postoje dvije matematske funkcije **F1** i **F2**. FFT funkcija je raspoloživa u **F2**. Funkcija **F2** može koristiti funkciju **F1** kao operand, dozvoljavajući da FFT bude izvršena na rezultatu **F1**. Ako postavimo funkciju F1 na Ch1-Ch2, a postavimo F2 na FFT od F1, dobiće se FFT od diferencijalnog mjerenja Ch1-Ch2.

Osciloskop može pokazati valni oblik u vremenskom domenu i spektar u frekventnom domenu istovremeno ili individualno. Normalno, tačke sampliranja nisu međusobno spojene. Za najbolji displej u frekventnom domenu, tačke sampliranja trebaju biti spojene sa linijama (vektorima).

Ovo može biti postignuto sa isključenjem svih kanala u vremenskom domenu, i uključenjem samo FFT funkcije. Alternativno, vektorski ON/OFF soft tasteri (na display meniju), mogu se uključiti na ON i pritisnuti STOP taster.

Bilo koja od ovih akcija dozvoljava uzorcima u frekventnom domenu da se spoje sa vektorima što uzrokuje da se displej pojavi da više liči na displej analizatora spektra.

Vertikalna osa FFT displeja je logaritamska , prikazana u dBV (decibeli relativni u odnosu na 1 V RMS).

$$\text{dBV} = 20 \log (V_{\text{RMS}})$$

Dakle, 1 V RMS sinusoide (2.8 V_{pp}) , će biti 0 dBV na FFT displeju.

Kada FFT je uključena, kurzori mogu čitati amplitudu u dBV i frekvenciju u Hz. (izvor kurzora mora biti postavljen na F2).

Pritišćući na taster FIND PEAKS , dobijemo lagano automatsko postavljanje kurzora na dvije najveće spektralne linije.

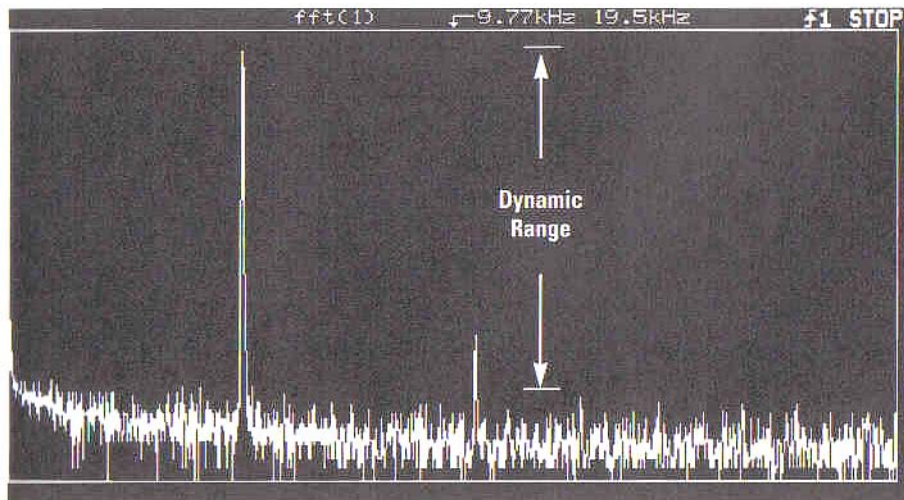
Frekventni opseg FFT je inherentno vezan na dugme **time/div**. Pod FFT menijem (selektiran sa +/- tasterom), postoje još dodatni soft tasteri za centralnu frekvenciju i frekventni opseg. Podešenjem dugmeta time/div. modificira se frekventni opseg, ali startna frekvencija (krajnja lijeva na displeju) ostaje ista. Centralna frekvencija i taster za frekventni opseg se koriste za zumiranje, na specifične frekvencije od interesa.

Ako je valni oblik odsječen (clipped) na ekranu, FFT se računa na valnom obliku koji je deformiran što će proizvesti distorziju i u frekventnom domenu (u obliku pogrešnih frekventnih komponenti).

Da bi se izbjegla ova vrsta problema volt/div. dugme na osciloskopu mora biti tako postavljeno da je čitav valni oblik postavljen i vidljiv na ekranu.

Dinamički opseg

Dinamički opseg mjerenja u frekventnom domenu je razlika (u decibelima) između najvećeg signala i najmanjeg signala koji može biti pouzdano izmjeren u isto vrijeme (vidjeti narednu sliku br. 13). Najveći mjerljivi signal je puna skala (8 podjela skale) valnog oblika u vremenskom domenu.



Slika 13.

Dinamički opseg mjerenja u frekventnom domenu je razlika između najvećih i najmanjih signala koji se mogu simultano mjeriti

Prag šuma mjerenja određuje najmanji signal koji može biti pouzdano izmjeren. Svaki signal ispod šuma ne može biti izdvojen. Distorzije i artefakti (izobličenja) sampliranja mogu također ograničiti dinamički opseg.

Tipični dinamički opseg FFT mjerenja je 50 dB. Korištenje usrednjavanja općenito povećava dinamički opseg mjerenja time što snižava prag šuma i reducira odzive usljed sampliranja.

Sampliranje

HP 54600 serija osciloskopa ima maksimalnu brzinu sampliranja od 200 Msa/sec. Jedno okidni opseg (singleshot bandwidth) u vremenskom domenu je specificiran na 1/10 brzine sampliranja, obezbjeđujući time najmanje 10 sampl tačaka po periodu kod najviše frekvencije. Pouzdani jednookidni mod rada sa aktivnom FFT funkcijom, će se javiti kod tim/div podešenim na 20 μ sec/div ili sporije. (na bržim podešenjima od ovoga tj. manjim time/div podešenjima puni zapis u vremenu, dovoljan broj tačaka ne bi mogao biti ostvaren po jednom trigerskom događaju, što ima za rezultat loše FFT mjerenje i računanje).

FFT rad na 20 $\mu\text{sec/div}$ rezultiraće u maksimalnom displeju frekvencije od 5 MHz, što definira korisnu jednodimenzionalnu mogućnost FFT.

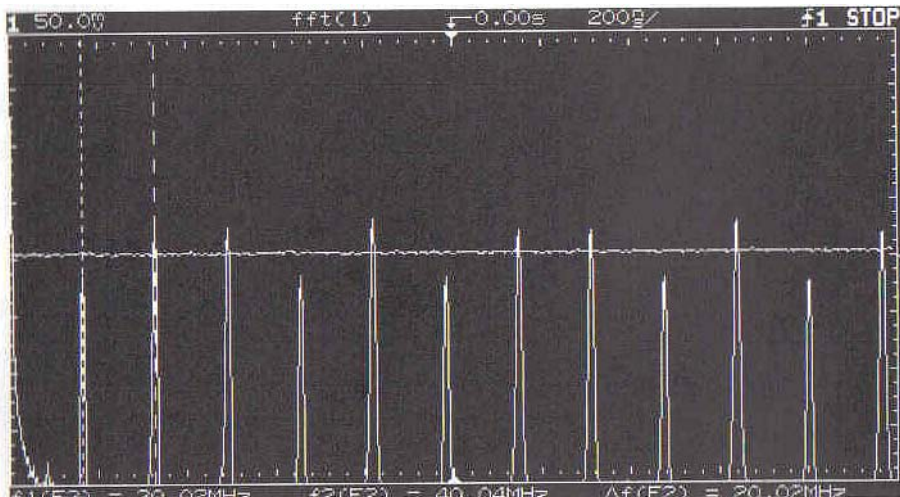
Repetitivna (ponavljajuća) tehnika sampliranja dozvoljava osciloskopu da digitizira valne oblike sa mnogo većom frekvencijom od 5 MHz, ukoliko su valni oblici ponavljajući. FFT funkcija tačno računa frekventni sadržaj ponavljajućih valnih oblika sve do propusnog opsega osciloskopa.

Na podešenjima time/div bržim od 20 $\mu\text{sec/div}$, artefakti procesa sampliranja koji nisu vidljivi u vremenskom domenu, mogu se pokazati u frekventnom domenu. Ovi artefakti se mogu pojaviti kao spektralne linije na cjelobrojnim harmonicima brzine sampliranja ili kao intermodulacija između sata sampliranja i ulazne frekvencije definirano kao:

$$f = n f_s \pm f_{in}$$

gdje je : $f_s = 20 \text{ MHz}$, sat sampliranja
 f_{in} frekvencija ulaznog signala
 n - bilo koji cijeli broj

Nadalje, harmonici od 20 MHz sata sampliranja se mogu pojaviti blizu kraja skale za vrijeme kada su valni oblici u vremenskom domenu samo parcijalno prikupljeni- uzorkovani (vidjeti na slici 14). Ovo je zbog toga što valni oblik ima nedostajuće tačke sampliranja koje se pojavljuju na intervalima koji su u relaciji sa brzinom sampliranja. Nakon što je valni oblik u potpunosti prikupljen, iskrzani (spurious) odzivi zbog sata saplovanja će biti mnogo manji i obično će iščeznuti. Intermodulacija između sata sampliranja i ulazne frekvencije će tipično biti veća od 50 dB ispod pune skale. Korištenje usrednjavanja smanjuje nivo ovih odziva.

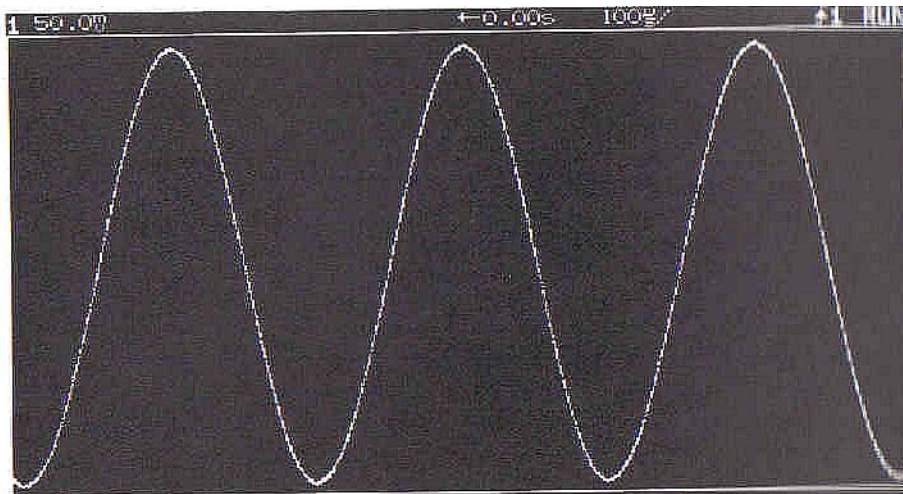


Slika 14

Harmonijska distorzija u sinusnom signalu

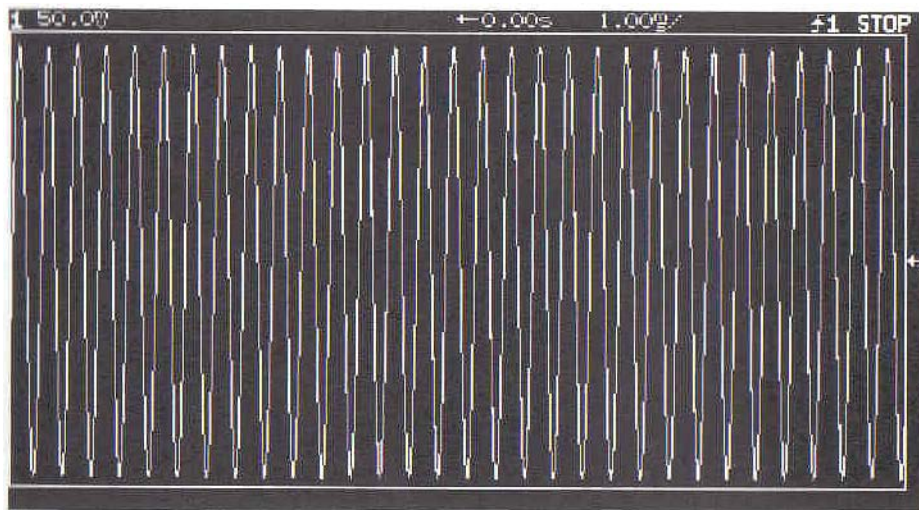
Sinusni valni oblici koji nisu idealno oblikovani u vremenskom domenu generiraju harmonike u frekventnom domenu. Ova harmonijska distorzija se pojavljuje na cjelobrojnim multiplima osnovne frekvencije sinusoidalnog signala. Posmatranje

ove distorzije u vremenskom domenu je obično vrlo teško , ukoliko se ne radi o zaista velikoj distorziji. Medjutim, u frekventnom domenu, ovi harmonici su vrlo vidljivi.



Slika 15a

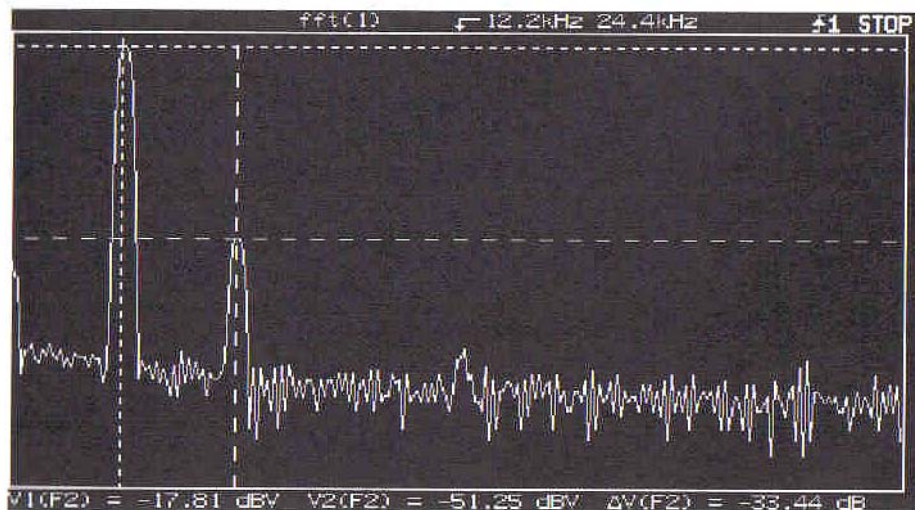
Slika 15a pokazuje sinusni valni oblik koji ima harmonijsku distorziju koja nije vidljiva u vremenskom domenu. Medjutim, FFT funkcija može vrlo lako odrediti količinu harmonijske distorzije. Slika 15a pokazuje dobru time/div selekciju za posmatranje signala u vremenskom domenu. Ako se primjeni FFT koristeći ovo time/div podešenje, spektralna linija bi se pojavila na krajnjem llijevom dijelu displeja frekventnog doemena. na slici 15b je pokazana sporija time/div skala , koja će sniziti efektivnu FFT brzinu samlovanja, rezultirajući u boljoj frekventnoj rezoluciji i boljoj separaciji osnovnog i viših harmonika valnog oblika sinusoide.



Slika 15b

Smanjujući time/div setting na vremenskom dugmetu osciloskopa, komprimiramo vremenski domen sinusoide i poboljšavamo frekventnu razlučivost

Nadalje , možemo koristiti dugmad na HP 54600 osciloskopu da centriramo frekvenciju i izaberemo frekventni opseg da bi zumirali u željeni dio spektra kao što je pokazano na slici 15c.



Slika 15c

FFT displej pokazuje harmonijsku distorziju prisutnu u sinusnom valnom obliku

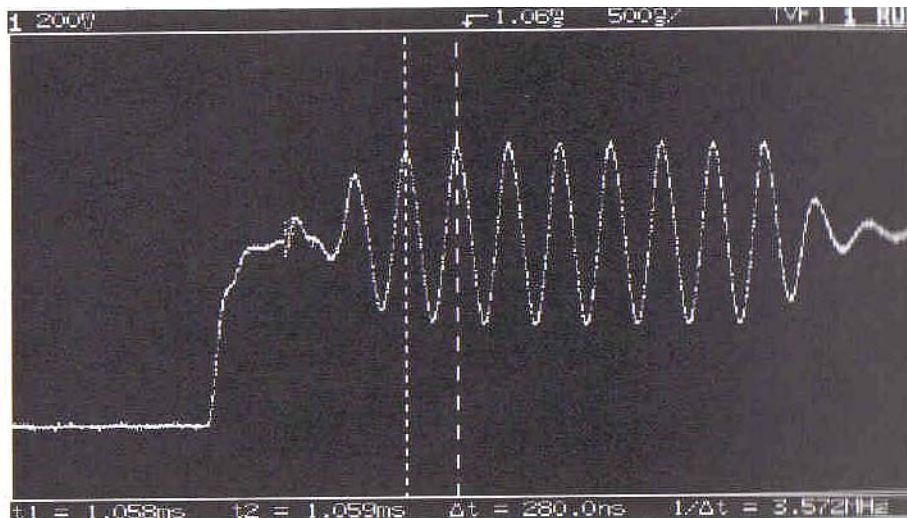
Primjetimo da je flattop prozor korišten u ovom mjerenju, rezultirajući u širim spektralnim linijama, ali korektnijim mjerenjima amplitude.

Softwareski taster "nadj vrhove" (Find Peaks) je korišten da postavi kurzore na dvije najveće spektralne linije, dajući absolutna mjerenja osnovne i drugog harmonika. Kurzor takodjer očitava i u relativnim jedinicama, indicirajući da je drugi harmonik 33 dB ispod osnovnog.

Video distorzija praska boja (colorburst)

Spektralni slučaj mjerenja harmonijske distorzije u sinusnom signalu se može naći kod video aplikacija. Tako naprimjer 3.58 MHz pod-nosilac boje (color sub-carrier) frekvencija je utisnuta (embedded) u NTSC kompozitni video signal i ima izvjestan nivo harmonijske distorzije udružene sa frkvencijom pod-nosioca.

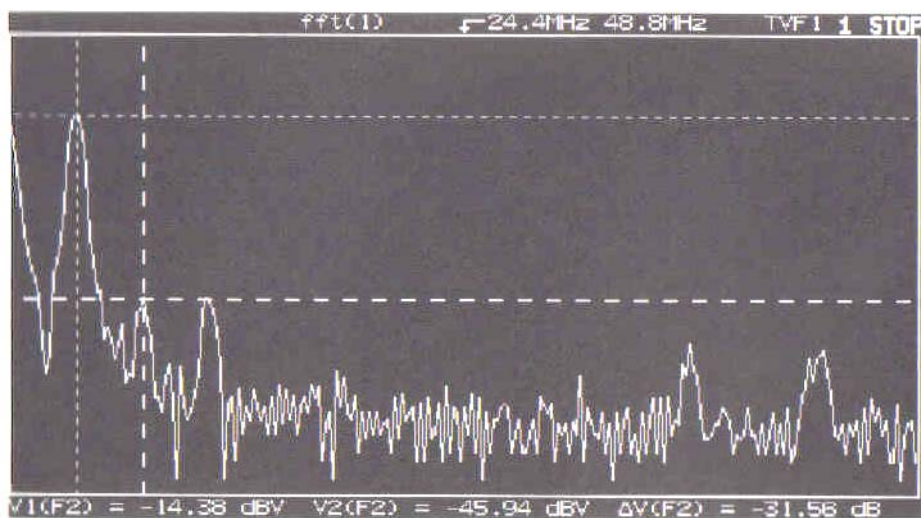
Da bi se mjerio samo ovaj signal, dugmad **time/div** i **delay** (kašnjenje) na osciloskopu se koriste da se zumira na pod-nosilac boje u vremenskom domenu. (slika 16a).



Slika 16a

FFT funkcija pokazuje harmonijski sadržaj podnosioca na slici 16b.

Ukoliko se ne bi dugmad za izbor time/div i kašnjenja na osciloskopu koristila da se zumira na željeni podnosilac, cijeli video signal (sa mnogim komponentama frekvencija) bi se pojavio u displeju frekventnog domena. Ove frekventne komponente bi pomutile podnosioca boje i njegove harmonike.



Slika 16b

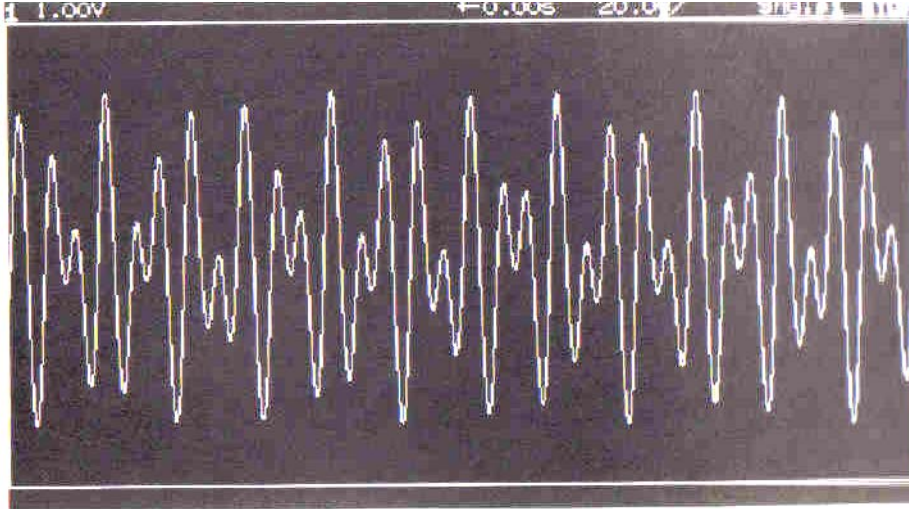
FFT funkcija pokazuje da je harmonijski sadržaj podnosioca boje više od 31 dB ispod podnosioca

Dvo tonska frekventna identifikacija

Drugi primjer korištenja FFT funkcije je da se identificiraju frekventne komponente koje je teško posmatrati u vremenskom domenu. Primjer za ovakav valni oblik je dvo tonski signal pokazan na slici 17a. Dva neharmonijski povezana sinusna valna oblika su nestabilna kada se gledaju u vremenskom domenu, tako da je

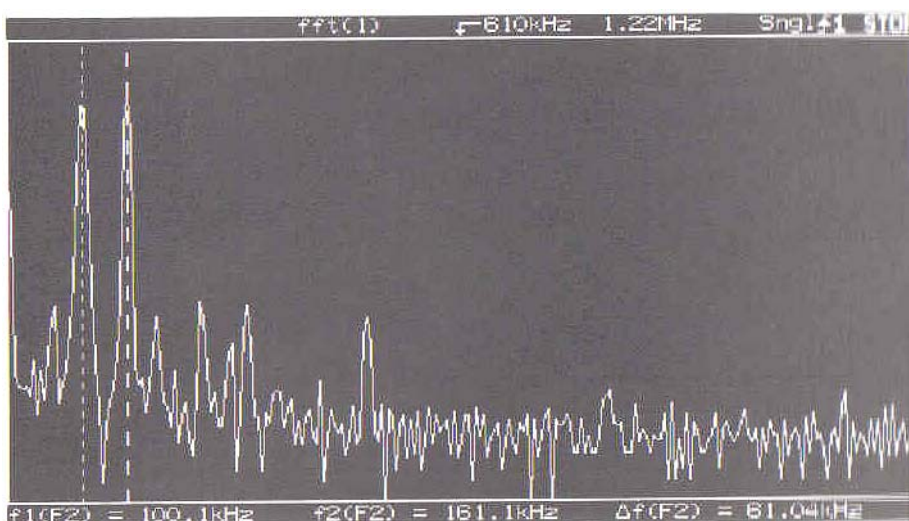
slika 17a , jedookidajući snimak njihovog valnog oblika a ne repetitivni displej. Neke procjene frekvencije tonova bi mogli pokušati provesti u vremenskom domenu ali vrlo teško.

Identificirajući frekvencije za više od dva tona bi bilo praktično nemoguće.



Slika 17a Dvo tonski signal u vremenskom domenu
FFT funkcija separira dva tona i prikazuje ih u frekventnom domenu (slika 17b).

Dvije najveće spektralne linije (sa kurzorima postavljenim na njihovim vrhovima), se lako mjere kao 100 kHz i 161 kHz. Primjetimo da su spektralne linije relativno tanke zbog toga što je korišten Hanningov prozor. Ovaj prozor, koji optimizira frekventnu razlučivost, je adekvatan za ovo mjerenje, pošto se mjeri frekventni sadržaj signala. Pošto su dva neharmonijski vezana sinusna valna oblika nestabilna (nerepetitivna), u vremenskom domenu, mjerenje mora biti učinjeno u jednostrukoj akviziciji, i korektno je samo za **time/div** postavljenje na 20 μsec ili sporije.



Slika 17b
FFT displej identificira frekvencije dva tona.

Preporuke za FFT mjerenja

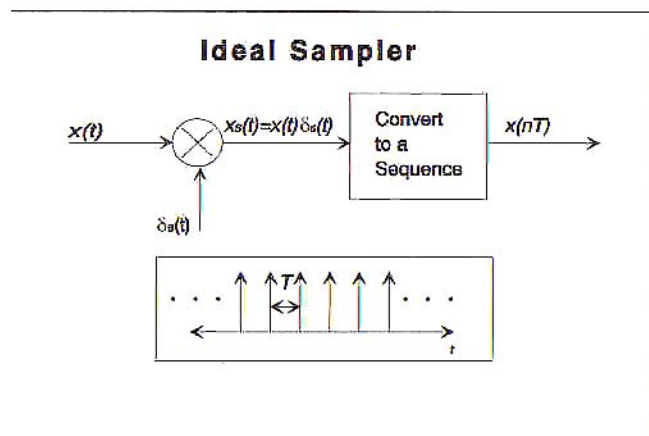
- Izabrati FFT efektivnu brzinu samplinga veću od dvostrukog frekventnog opsega signala
- Za najbolju frekventnu razlučivost koristiti HANNINGOV prozor
- Za najbolju amplitudnu tačnost koristiti FLATTOP prozor
- Za najbolji displej u frekventnom domenu treba isključiti kanal (displej u vremenskom domenu) ili pritisnuti STOP taster.
- Provjeriti da vremenski valni oblik nije klipovan (priljepljen) na displej kada se koristi FFT funkcija
- Koristiti time/div postavljenje od 20 μ sec/sec ili sporije za jednostruka (single-shot) mjerenja.

POGLAVLJE 17

FFT LABORATORIJSKI EKSPERIMENTI SA
HP 54600 OSCILOSKOPIMA

OSNOVE TEORIJE SAMPLIRANJA

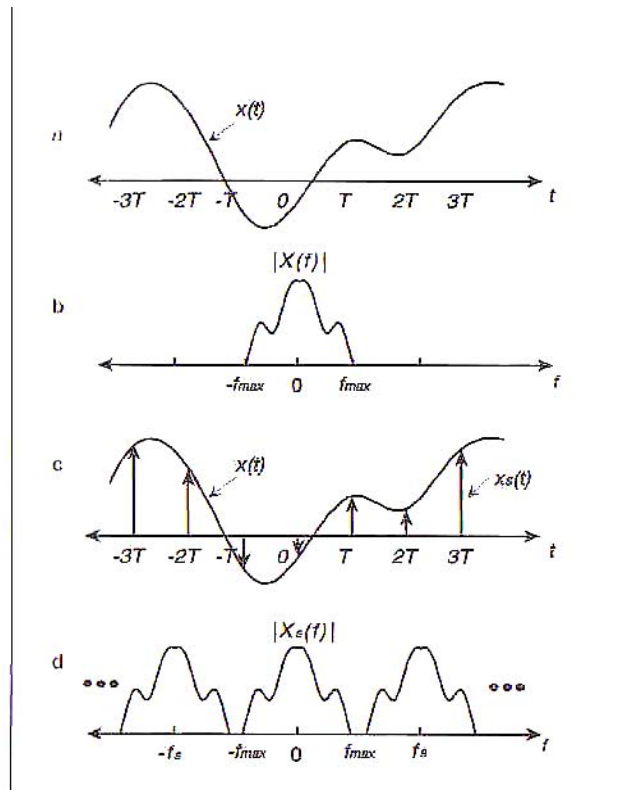
Teorija samplovanja obezbjeđuje matematske osnove za analizu kontinualnih vremenskih signala sa metodama digitalnog procesiranja signala (**DSP**).
Idealni sampler pokazan na narednoj slici 1.1 je važan konceptualni alat za integraciju vremenskih ko**ntinualnih i diskretnih domena**.



Slika 1.1

*Idealni sampler je konceptualni alat za uspostavljanaje veze izmedju kontinualne **Fourierove transformacije i diskretne (DFT)**. Ulaz u idealni sampler je $x(t)$ a izlaz se sastoji od uzorkovanih vrijednosti $x(nT)$. "Analogni" samplirani signal, $x_s(t)$, i njegova Fourierova transformacija $X_s(f)$, su pokazani na slici 1.2*

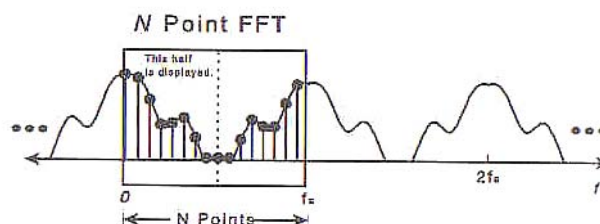
Slika 1.2 ilustrira kako idealni sampler (uzorkivač), može biti korišten da se objasni veza izmedju **Fourierove transformacije i DFT**. Najvažniji zaključak koji se može izvući iz slike 1.2 je da proces uzorkovanja rezultira u **periodičnoj Fourierovoj transformaciji** kao što je ilustrirano u djelu (d) na slici 1.2.



Slika 1.2

- a) originalni vremenski kontinualni signal $x(t)$
- b) Fourierova transformacija od $x(t)$
- c) Signal $x_s(t)$. Ovaj "konceptualni" signal se sastoji od niza impulsa gdje je svaki impuls težine $x(nT)$ i raspoređen za T sekundi jedan od drugog. Pošto je $x_s(t)$ tehnički vremenski kontinualni signal, on ima Fourierovu transformaciju $X_s(f)$, koja je ilustrirana u dijelu (d).
- d) Primjetimo da $X_s(f)$ je formirano prvo množeći $X(f)$ sa konstantnom vrijednošću $1/T$ i onda ponavljajući $X(f)/T$ na intervalima razmještenim za $f_s=1/T$ međusobno, (f_s je efektivna brzina uzorkovanja). Ako brzina uzorkovanja nije dovoljno velika, tada f_s neće biti dovoljno veliko da obezbjedi da se replike $X(f)/T$ ne preklapaju. Pojaviće se aliasing kada se replike od $X(f)/T$ preklapaju.

Za signale konačnog trajanja, DFT se dobije uzimajući N jednako raspoređenih uzoraka ove periodične Fourierove transformacije (vidjeti sliku 1.3).



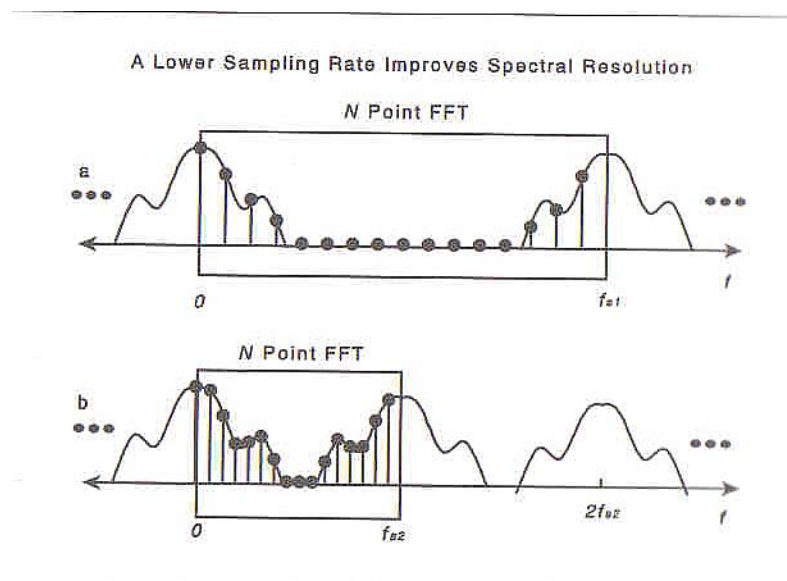
Slika 1.3

DFT, od diskretnog signala $x(nT)$ konačnog trajanja, se dobije samplirajući spektar $x_s(t)$ (slika 1.1). Ovi frekventni uzorci su jednako raspoređeni unutar frekventnog opsega 0 do f_s Hz. Može biti pokazano da za realne signale, DFT će uvijek posjedovati "presavijajuću" (folding), simetriju, koja je ilustrirana na gornjoj slici. Zbog toga je uobičajeno da se samo prikaže prva polovina (dio na lijevo od crtkane linije) kao FFT. I HP 54600 sa modulom HP 54657A slijedi ovu konvenciju tako da je maksimalni frekventni opseg za FFT displej $f_s/2$.

Vidjeli smo dakle, da DFT proizvodi N jednako raspoređenih uzoraka od Fourierove transformacije originalnog signala. Drugim riječima, spektar originalnog signala $x(t)$ je uzorkovan kod frekventnih vrijednosti

$$f_k = \frac{k}{NT} \text{ Hz for } k = 0, 1, 2, \dots, N - 1 .$$

Rezolucija frekvencije od $1/NT$, Hz je najbolje što je moguće očekivati da se postigne sa N tačaka FFT, pošto svaki frekventni uzorak je smješten za $1/NT$ jedan od drugoga. Na bazi gornjeg izraza može se vidjeti da povećavajući T (smanjujući brzinu samplovanja) vodi ka poboljšanoj frekventnoj razlučivosti. Međutim, da bi se izbjegao aliasing, brzina uzorkovanja ne smije biti reducirana ispod Nyquistove brzine ulaznog signala. Slika 1.4 grafički ilustrira efekat koji brzina uzorkovanja ima na spektralnu rezoluciju.



Slika 1.4

Slika grafički ilustrira efekat koji brzina uzorkovanja ima na spektralnu rezoluciju. Djelovi (a) i (b) prikazuju frekventne uzorke dobijene koristeći N tačaka FFT i brzinu uzorkovanja od f_{s1} i f_{s2} . Primjetimo da brzina uzorkovanja korištena u dijelu (a) je dva puta manja od ona na dijelu (b). Smanjenjem brzine uzorkovanja (a da još uvijek izbjegnemo aliasing), mi dobijemo više uzoraka u "interesantnom" regionu spektra. Primjetimo da dio (a) ima manje umetnutih uzoraka spektra nego što ima dio (b).

Efekat prozora

Pošto DFT zahtjeva ulazni signal konačne dužine, jedan tekući signal mora biti skraćen prije nego se primjeni FFT računanje. Proces skraćivanja se postiže sa preklapanjem ulazne sekvence $x[n]$ sa prozorom konačne dužine $w[n]$ i vršeci tačka po tačka množenje kao što je to pokazano na slici 1.5.

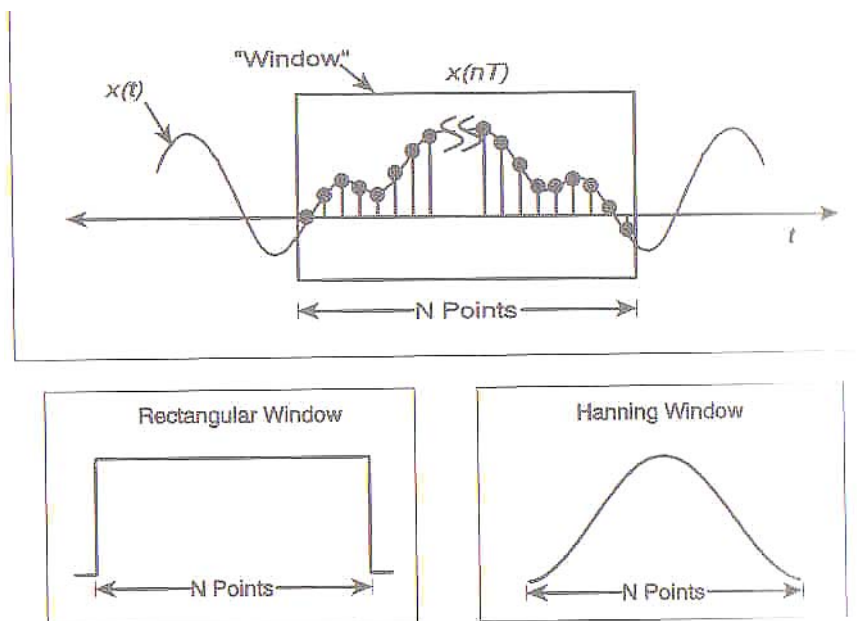
Dakle, FFT se računa na signalu $y[n]$ koji se dobije kao :

$$y[n] = x[n] w[n]$$

Procesi upozoravanja uvode gubitke u spektralnoj rezoluciji i efekat koji je poznat kao spektralno curenje (leakage). Općenito, izbor funkcije prozora uključuje kompromis između ova dva efekta. To jest, prozor sa boljom frekventnom rezolucijom, u opštem slučaju, nisu dobri sa aspekta curenja spektra i obratno.

Čitav niz opcija prozora je na raspolaganju za DSP aplikacije. U eksperimentima koje ćemo provesti, koristimo pravougaoni prozor, Hanningov, prozor ravnog vrha (flattop) i esponencijalni prozor.

Pravougaoni i Hanningov prozor su prikazani na slici 1.5.



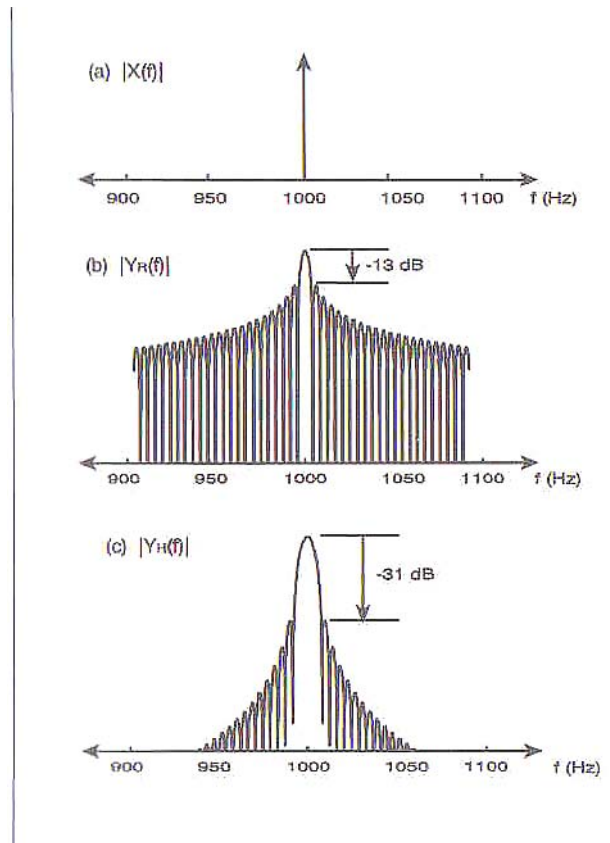
Slika 1.5

Tekući kontinualni signal je konvertovan u konačan broj uzoraka sa korištenjem prozora. Svaki uzorak u boksu gore pokazanom je pomnožen (tačka po tačka) sa funkcijom prozora.

Efekte upozorenja su dobro ilustrirani na primjeru sinusoidalnog signala. Slika 1.6 dio (a) prikazuje Fourierovu transformaciju (amplituda > 0, $f_0=1000$), sinusoidalnog signala. Naravno, spektralni sadržaj sinusoide je predstavljen impulsnom funkcijom koncentriranom kod osnovne frekvencije sinusoide f_0 . Da bi

se izvršila FFT analiza , mi ćemo prvo pomnožiti ulazni signal sa funkcijom prozora.

Podsjetimo se da Fourierova transformacija dva signala pomnožena u vremenskom domenu, je data sa konvolucijom transformacije svakog signala u frekventnom domenu. U opštem slučaju, konvolucija ima "izgladjujući" i "raspršavajući" (smoothing and spreading) efekat, koji rezultira u spektralnom curenju i gubitku u rezoluciji. Slika 1.6 ilustrira efekte uprozorenja u frekventnom domenu.



Slika 1.6

- a) Fourierova transformacija (amplitude) sinusoide
- b) 1024 tačke DFT , koristeći četvrtasti prozor. Primjetimo veliko curenje spektra za ovaj prozor. Medjutim, pošto spektralna rezolucija može biti posmatrana da je funkcija od širine glavnog brijega (main lobe), ovaj tip prozora može ponekad biti korišten da razriješi problem tijesno raspoređenih frekventnih komponenata.
- c) 1024 tačke DFT sa Hanningovim prozorom. Izbjegavajući naglo prekidanje signala u vremenskom domenu na ivicama prozora, Gibbsov efekat je reduciran i kao rezultat imamo smanjeno curenje spektra. Širina glavnog brijega (lobe) je veća, što znači da je spektralna rezolucija manja. Hanningov prozor je popularan zato što postiže dobar bilans između curenja spektra i rezolucije.

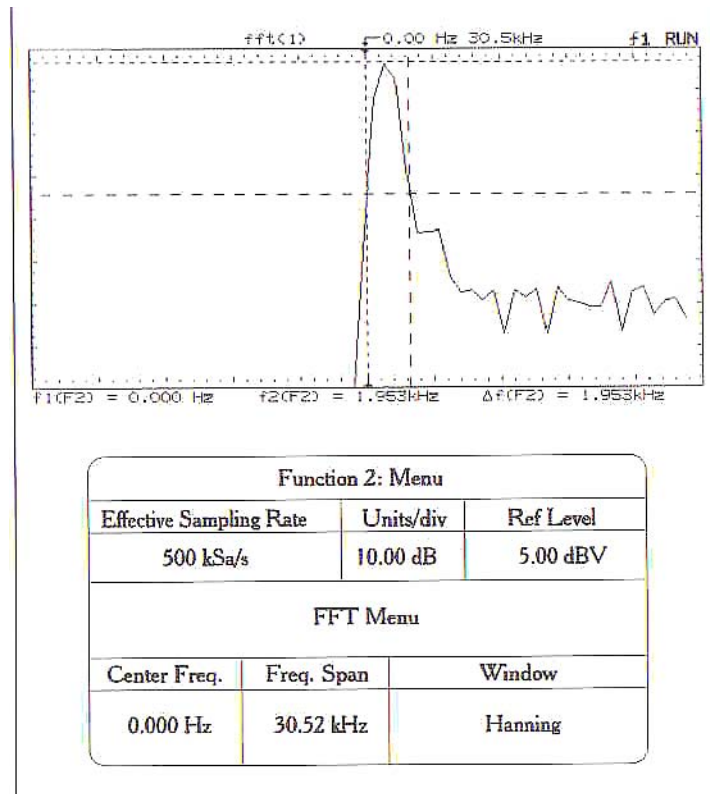
Eksperiment br. 1

Brzina sampliranja, frekventna rezolucija, i curenje spektra za sinusoidalni ulazni signal

Ovaj jednostavni eksperiment ilustrira relaciju izmedju efektivne brzine sampliranja i rezultirajuće frekventne razlučivosti za spektralnu analizu koristeći FFT. Osobine curenja spektra kod pravougaonog i Hanningovog prozora su takodjer demonstrirane. Poredjenje može biti napravljeno izmedju teoretskih rezultatata za FFT analizu sinusoide i eksperimentalnih rezultatata dobijenih koristeći FFT modul.

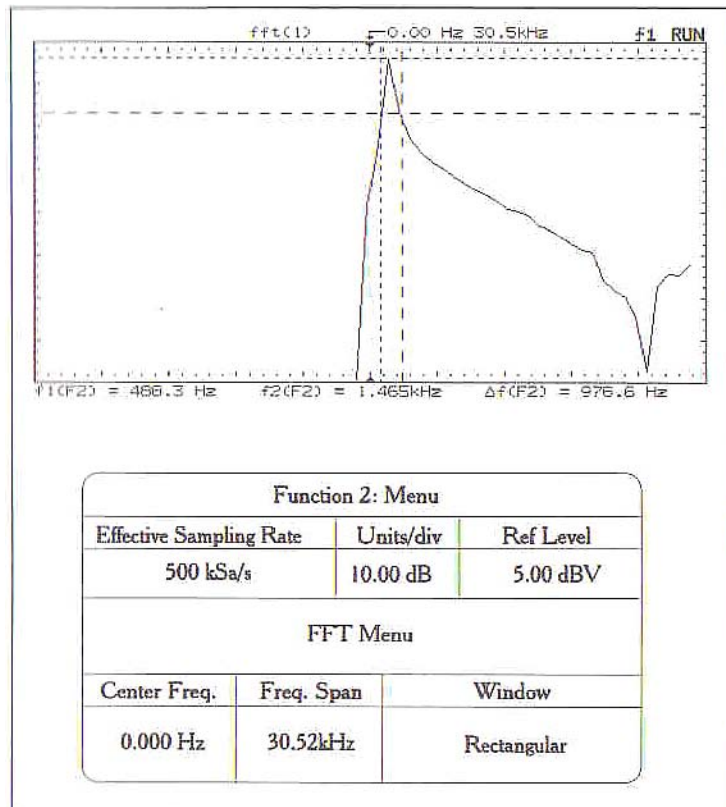
Procedura eksperimenta

1. Spojiti $3.5 V_{pp}$ sinusoidalni ulazni signal na kanal 1 osciloskopa HP 54645A. Podesiti frekvenciju sinusoide na približno 1 kHz. Primjetimo da trening kit HP 54654 A obezbjeđuje ovaj sinusni signal na pinu 12.
2. Koristiti *Autoscale* da se displejira valni oblik u vremenskom domenu. Mjeriti frekvenciju sinusoide selektirajući prvo *Time* a onda birajući *Freq* selekciju na Time Measurement dijelu Menija.
3. Poslije ovoga otići na *Function* meni pritišćući +/- taster. Pod "Function 2" dijelom menija selektirati *On*. Da bi prikazali vektorski displej FFT u realnom vremenu, selektirati 1 na prednjem displeju instrumenta a onda pritisnuti 1 na ekranskom meniju. Podesiti FFT meni podešenja na vrijednosti pokazane na slici 2.1 .
4. Koristiti *Cursors* i *Find Peaks* automatizovanog mjerenja da se izmjeri osnovna frekvencija sinusoide. Primjetimo da je širina glavnog loba Hanningovog prozora otprilike 2 kHz, (koristiti *Cursors* meni da se izmjeri širina glavnog loba). Nakon toga, promjeniti na pravougaoni prozor kao što je pokazano na slici 2.2. Primjetimo da je sada širina glavnog loba reducirana na oko 1 kHz. Medjutim, curenje spektra je značajno povećano sa ovim prozorom.



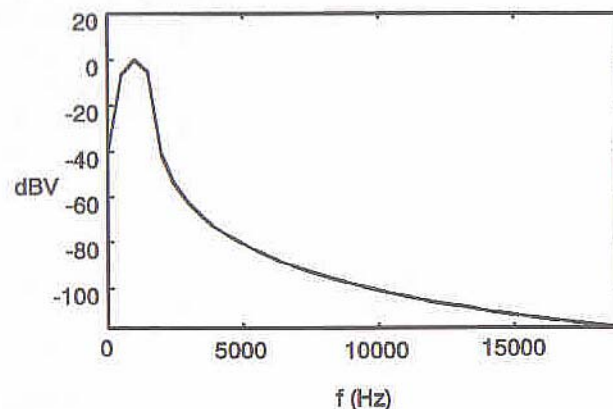
Slika 21.

1024 DFT od sinusoidalnog signala amplitude $3.5 V_{pp}$ i frekvencije 1 kHz, samplovanog sa 500 kSa/sec, koristeći Hanningov prozor. Širina glavnog loba se mjeri postavljanjem V_1 kurzora na vrh loba a V_2 kurzora na 31 dB ispod V_1 . Kao što je pokazano na slici, f_1 i f_2 kurzori su nakon toga podešeni na tačke gdje V_2 kurzor i glavni lob se presijecaju.



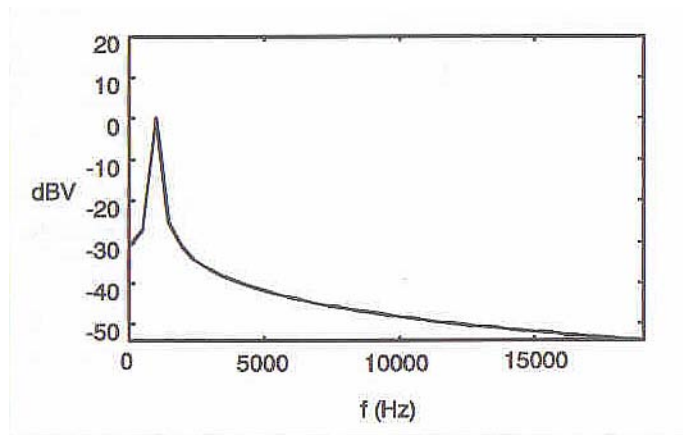
Slika 2.2

1024 DFT od sinusoidalnog signala amplitude $3.5 V_{pp}$ i frekvencije 1 kHz, samplovanog sa 500 kSa/sec, koristeći pravougaoni prozor. Širina glavnog loba se mjeri postavljanjem V_1 kurzora na vrh loba a V_2 kurzora na 13 dB ispod V_1 . Kao što je pokazano na slici, f_1 i f_2 kurzori su nakon toga podešeni na tačke gdje V_2 kurzor i glavni lob se presijecaju



Slika 2.3

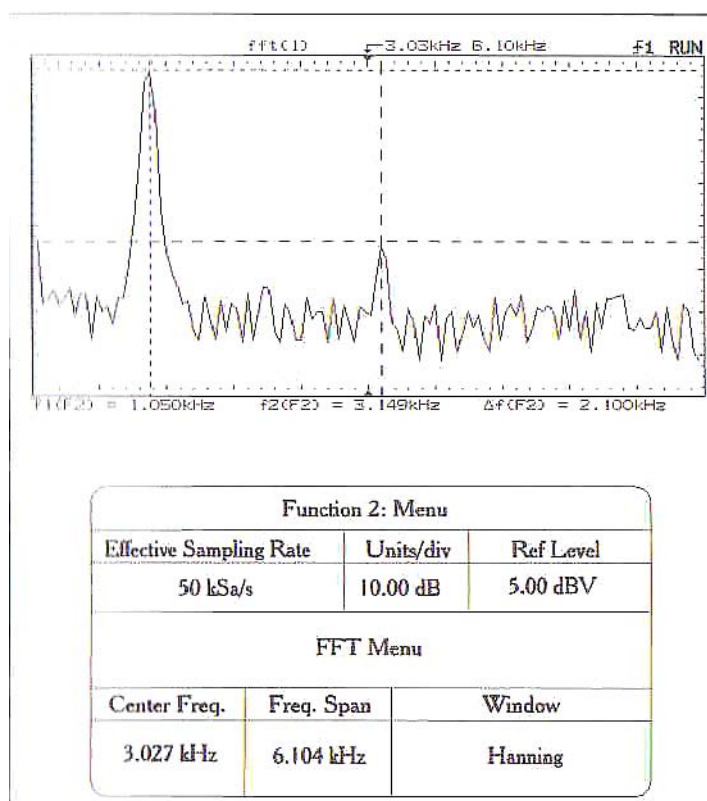
"Teoretska" 1024 tačke DFT od sinusoide amplitude 1 V RMS i frekvencije od 1 kHz samplovana sa 500 kSa/sec koristeći Hanningov prozor.



Slika 2.4

Isto kao i slika 2.3 samo što je korišten **pravougaoni** prozor.

5. Koristeći **Tim/Div** dugme smanjiti efektivnu brzinu sampliranja na **50 kSa/sec**. Podesiti podešenja FFT menija na ona specificirana na slici 2.5. Primjetimo da je širina glavnog loba reducirana na oko **200 Hz** za **Hanningov prozor** što indicira poboljšanu frekventnu razlučivost. Nakon toga izaberi **pravougaoni** prozor iz FFT menija i primjetiti da je sada širina glavnog loba oko **100 Hz**. Ponovno, curenje spektra je značajno veće.

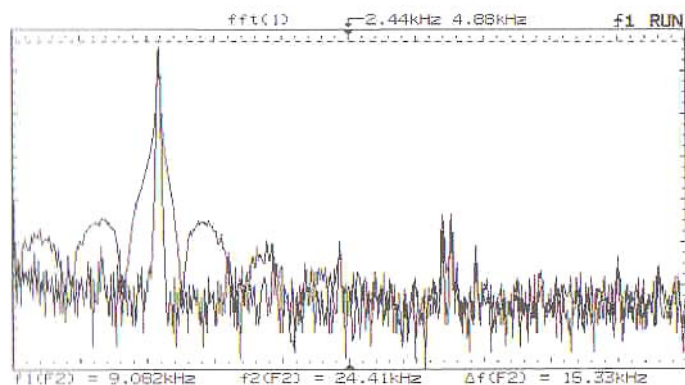


Slika 2.5

1024 DFT od sinusoidalnog signala amplitude $3.5 V_{pp}$ i frekvencije 1 kHz, samplovanog sa 50 kSa/sec, koristeći Hanningov prozor. Signal sinusoide sa trening kita HP 54654A je korišten kao ulazni signal. Ovaj signal ima namjerno uvedenu harmonijsku distorziju koja je evidentna na gornjoj slici. Jedan od

vertikalnih kurzora je na osnovnoj frekvenciji dok je drugi lociran na trećem harmoniku.

6. Ponoviti prethodni korak koristeći efektivnu brzinu uzorkovanja od 10 kSa/sec. Koristiti sliku 2.6 za pomoć u selekciji FFT podešenja u meniju. Primjetimo da je širina glavnog loba sada oko 20 Hz za Hanningov prozor i oko 10 Hz za pravougaoni prozor.



Function 2: Menu		
Effective Sampling Rate	Units/div	Ref Level
10 kSa/s	10.00 dB	5.00 dBV
FFT Menu		
Center Freq.	Freq. Span	Window
2.441 kHz	4.883 kHz	Hanning, Rectangular

Slika 2.6

1024 DFT od sinusoidalnog signala amplitude $3.5 V_{pp}$ i frekvencije 1 kHz, samplovanog sa 10 kSa/sec. Osciloskop prikazuje preklapanje spektra signala i kod korištenja Hanningov i pravougaonog prozora. Ovaj displej je kreiran prvo selektirajući postavljena kao u boksu a onda pohranjujući svaki trag ("trace"), koristeći Trace tastere osciloskopa. Uključujući obadva tracea, dobićemo displej kao na slici.

Pitanja

1. Treba li efektivna brzina sampliranja biti povećana ili smanjena da bi se povećala frekventna razlučivost FFT?
2. Da li postoji granica mogućnosti spektralne razlučivosti za fiksni 1024 tačke FFT?
3. Da li Hanningov prozor ispoljava više ili manje curenja spektra u poredjenju sa pravougaonim prozorom?

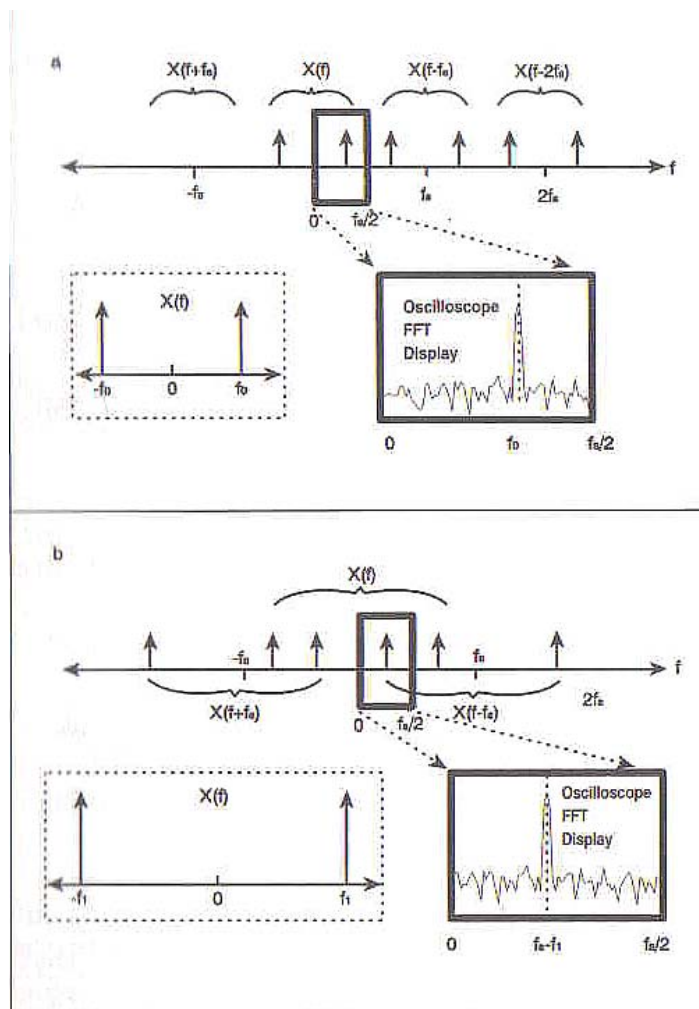
Zaključci

1. Spektralna rezolucija FFT se poboljšava smanjujući efektivnu brzinu sampliranja koristeći *Time/Div* dugme.
2. Efektivna brzina sampliranja treba biti veća od Nyquistove brzine ulaznog signala da se izbjegne alising.
3. Pravougaoni prozor ispoljava veći stepen curenja spektra od Hanningovog prozora. Loše osobine curenja spektra kod pravougaonog prozora pomraćuju njegove mogućnosti dobre spektralne razlučivosti.

Ekspерiment 2

Aliasing

Cilj ovog eksperimenta je da se demonstrira da se **aliasing pojavljuje ako efektivna** brzina sampliranja je ispod Nyquistove brzine za dati ulazni signal. Koristićemo ponovno sinusoidalni ulaz za ovaj eksperiment kao i u prvom primjeru. Teoretska osnova za ovaj eksperiment je ilustrirana na slici 3.1



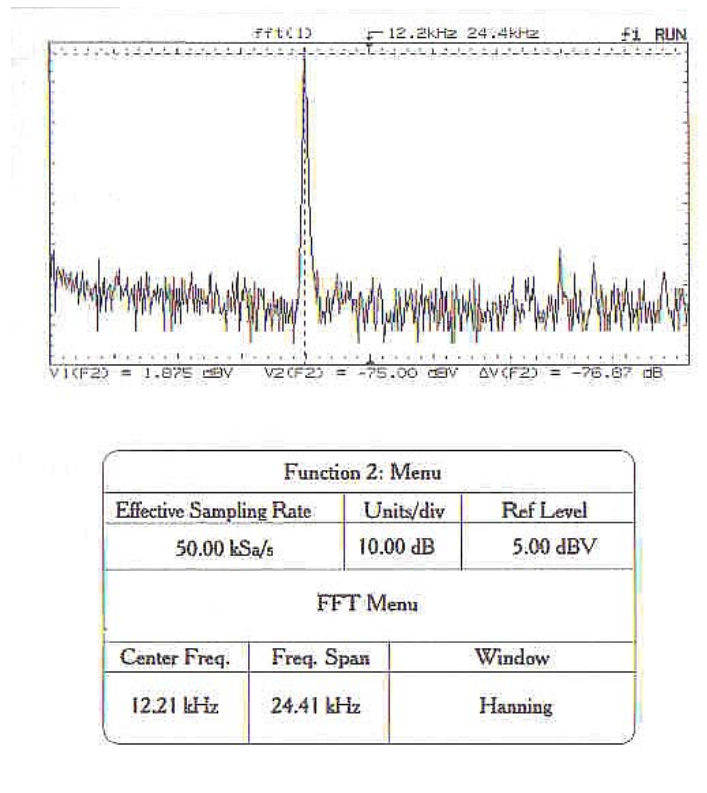
Slika 3.1

- a) U prvom dijelu slike, sinusoida se uzorkuje sa brzinom koja je veća nego Nyquistova frekvencija sinusoida. Spektar sinusoida je označen sa $X(f)$ i pokazan je u boks crtkanom linijom. Sampliranje signala će rezultirati u periodičnom ponavljanju originalnog spektra kao što je gore pokazano. Boks crtan punom debelom linijom predstavlja dio spektra koji prikazuje osciloskop. Primjetimo da maksimalna frekvencija koja može biti prikazana na osciloskopu je $f_s/2$. Pošto se pojedinačne replike ne preklapaju (tj. $f_0 < f_s/2$), spektar ulaznog signala je korektno predstavljen.
- b) U drugoj polovini slike, sinusoida je podsamplirana što rezultira u alisingu. Kada se originalni spektar periodično replicira, dio spektra se ne vidi zbog toga što $X(f)$ se ne pojavljuje u unutar frekventnog opsega displeja osciloskopa. Međutim, dio spektra kao rezultat $X(f-f_s)$ se pojavljuje unutar frekventnog opsega displeja. Zbog alisinga displej osciloskopa ne prikazuje korektno spektar signala.

Procedura

1. Spojiti signal generator na kanal 1 osciloskopa. Izabrati $3.5 V_{pp}$ sinusoidalni signal sa osnovnom frekvencijom od oko 10 kHz. Koristiti taster **Autoscale** da se prikaže valni oblik u vremenskom domenu.
2. Sada pritisnuti +/- taster a onda selektirati **On** ulaz na Function 2 Meniju. Nakon toga pritisnuti 1 taster na prednjem panelu i izabrati **Off** na meniju 1. Koristiti **Time/Div** kontrolno dugme da se izabere efektivna brzina sampliranja od 50 kSa/sec. Pogledati na narednu sliku 3.2 da se vidi kako podesiti ostala podešenja u FFT meniju.
3. Koristeći kontrolnu dugmad na generatoru sinusoida, postepeno povećavati frekvenciju sinusoida do oko 24 kHz, dozvoljavajući FFT displeju da se stabilizira u nekoliko tačaka tokom povećanja frekvencije. Treba da vidimo kako se vrh loba FFT displeja pomjera prema desno kako se povećava frekvencija sinusoida.
4. Nastaviti sa laganim povećanjem osnovne frekvencije sinusoida. Alising će se početi pojavljivati kada frekvencija sinusoida predje 25 kHz. Kako frekvencija prelazi opseg od 25 do 50 kHz, glavni lobe se pomjera prema lijevoj strani displeja. Kada frekvencija sinusoida nastavi da se povećava od 50 do 75 kHz , glavni lob će se ponovo pomjerati prema desnom kraju FFT displeja.
5. Postaviti frekvenciju sinusoida na 40 kHz. Koristiti **Cursors** i **Find Peaks** funkcije da izmjerite vršnu frekvenciju prikazanu na FFT. Zbog alisinga, FFT će pogrešno indicirati da se peak javlja kod oko 10 kHz.

6. Konačno, ponoviti korak 5 , stim što ovaj put treba promjeniti efektivnu brzinu sampliranja na 100 kSa/sec. Pošto je efektivna brzina sampliranja veća od Nyquistove brzine, spektar je korektno aproksimiran.



Slika 3.2
1024 DFT sinusoide 3.5 V_{pp} i frekvencije 10 kHz.

Pitanja

- Pošto spektralna rezolucija FFT je ograničena efektivnom brzinom sampliranja i pošto moramo samplirati sa brzinom iznad Nyquistove da izbjegnemo aliasing, da li je moguće poboljšati spektralnu rezoluciju sa dodavanjem nula na početku seta (zero padding).
- Ako se 120 kHz sinusoida sampluje sa 50 kSa/sec, kod koje će se frekvencije pojavljivati na FFT displeju aliasovana komponenta na 100 kHz.
- Da li na aliasing utiče izbor tipa prozora?

Zaključci

Frekventni opseg FFT displeja je od 0 do $f_s/2$ Hz. Sve frekventne komponente ulaznog signala koje su veće od $f_s/2$ pojavljivat će se kao aliasovane i na nižim frekvencijama na FFT displeju.

Ekperiment 3

Frekventna analiza periodičnog signala

Ovaj eksperiment demonstrira korištenje FFT za analizu spektralnog sadržaja pravougaonog i trouglastog signala. Poređićemo teoretske i eksperimentalne tehnike i diskutirati kompromise između frekventne rezolucije i aliasinga.

Teorija

Fourierova transformacija pravougaonog valnog oblika je u uskoj korelaciji sa njenim opisom preko Fourierove serije. Ustvari, Fourierova transformacija bilo kojeg periodičnog signala je data sa:

$$X(\omega) = \sum_{k=-\infty}^{\infty} 2\pi a_k \delta(\omega - k\omega_0)$$

gdje ω_0 (rad/s) je osnovna frekvencija periodičnog valnog oblika, a a_k su eksponencijalni oblici koeficijenata Fourierove serije.

Fourierova analiza pravougaonog valnog oblika (vrh peak = 1, duty cycle = 50 %), otkriva da su amplitude koeficijenata serije date sa :

$$|a_k| = \left| \frac{\sin(\pi k / 2)}{k\pi} \right|$$

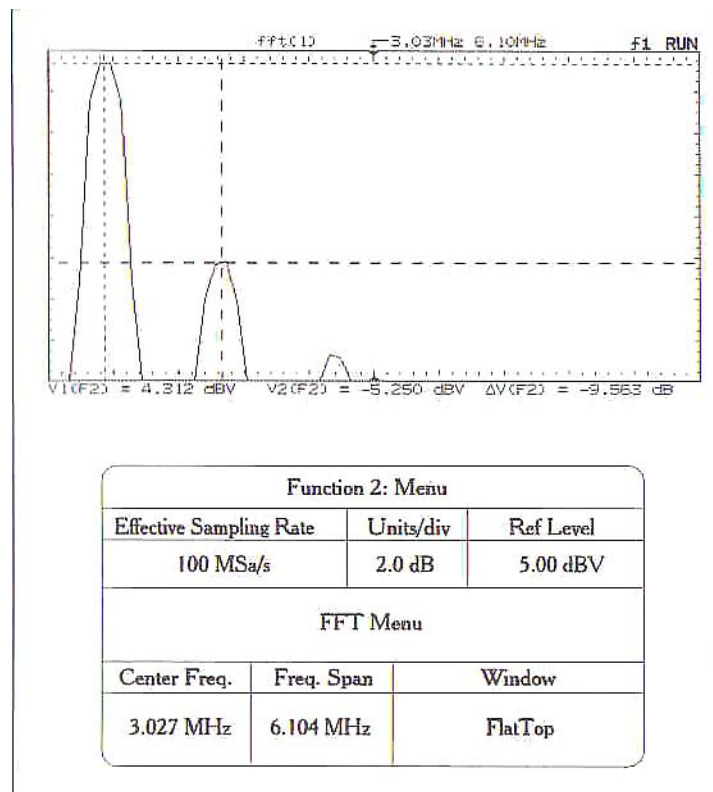
Tabela 4.1 prikazuje koeficijente (amplitude) Fourierove serije, za pravougaoni i trouglasti valni oblik. Ova tabela je korisna da se odredi efektivna brzina sampliranja da se može provesti FFT analiza. Na primjer, mi možemo željeti da zanemarimo sve spektralne komponente poslije 9-og harmonika. Pod ovom pretpostavkom, mi možemo izabrati da brzina sampliranja bude veća od $18 \omega_0$. Međutim, moramo imati u vidu da će se viši harmonici aliasirati, i da moramo biti pažljivi da ne izvodimo pogrešne zaključke iz ove analize.

Square Wave		Triangle Wave	
Harmonic	Magnitude (dB)	Harmonic	Magnitude (dB)
1	-9.943	1	-4.842
3	-19.485	3	-26.924
5	-23.922	5	-35.788
7	-26.845	7	-41.618
9	-29.028	9	-45.963
11	-30.771	11	-49.423
13	-23.222	13	-52.295

Tabela 4.1
Eksponencijalni oblik koeficijenata Fourierove serije

Procedura

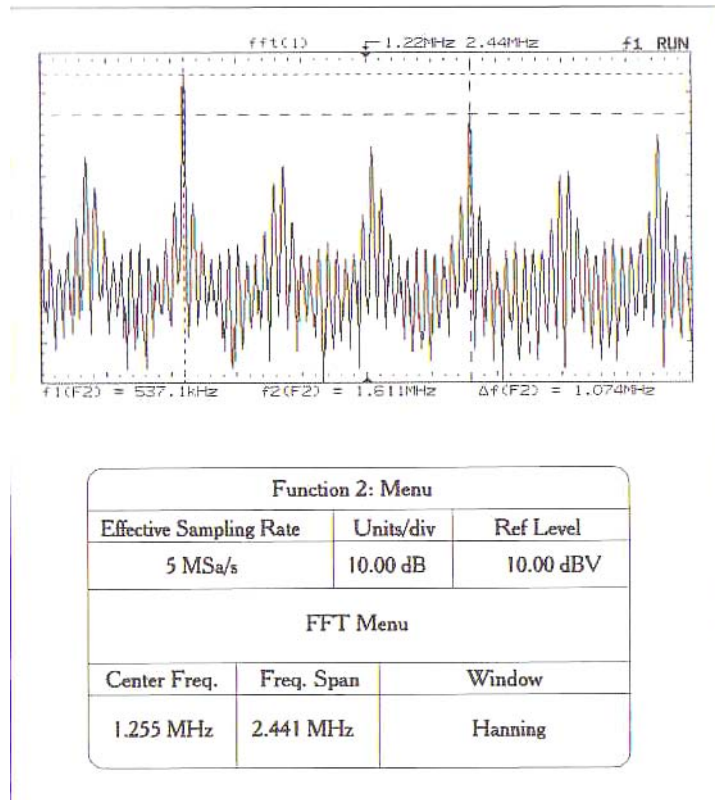
1. Koristiti bilo HP 54654 training kit (pin 2) ili generator funkcija da se dobije 500 kHz pravougaoni valni oblik ($2 V_p$). Koristiti *Autoscale* da se prikaže valni oblik u vremenskom domenu.
2. Aktivirati "*Functions*" meni selektirajući +/- taster. Selektirati *On* pod dijelom *Function 2* displeja. Zatim selektirati taster 1 na prednjem panelu i isključiti kanal 1. U ovoj tački , efektivna brzina sampliranja je 200 Msa/sec , i mnogi od harmonika pravougaonog valnog oblika su prikazani. Koristiti narednu sliku 4.1 za pomoć u selekciji menija za FFT.



Slika 4.1

DFT u 1024 tačke amplitude od $2 V_p$ 500 kHz pravougaonog valnog oblika je prikazana, koristeći Hanningov prozor i efektivnu brzinu sampliranja od 200 Msa/sec.

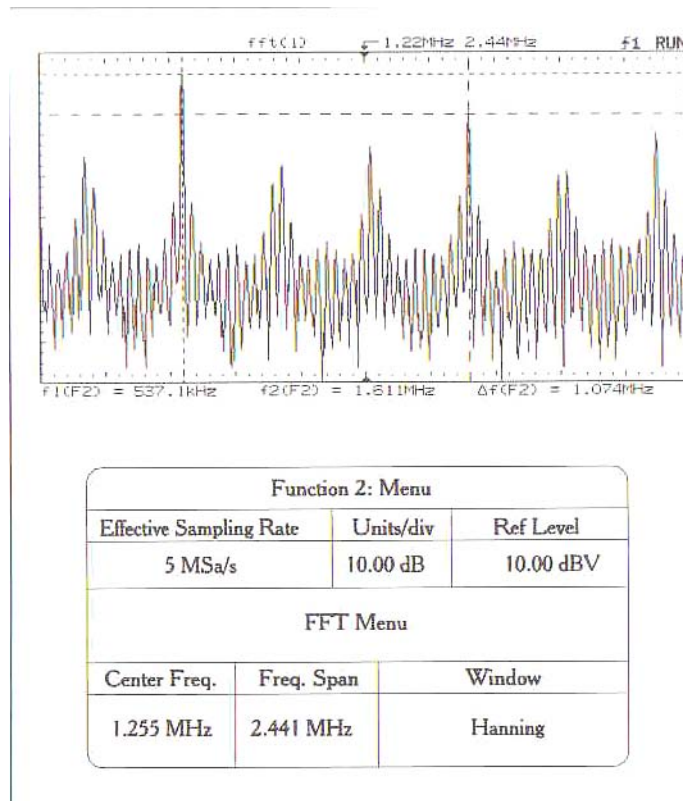
3. Sada koristiti *Time/Div* dugme da se smanji brzina sampliranja na 100 Msa/sec. Koristiti sliku 4.2 kao pomoć da se odrede postavljeno za FFT menije. *Cursors* i *Find Peaks* dugmad mogu biti korištena da se odrede relativne amplitudne razlike između vrhova. Poredite ove rezultate sa teoretskim vrijednostima dobijenim oduzimanjem vrijednosti amplituda u tabeli 4.1. Naprimjer, Tabela 4.1 pokazuje da ima 9.54 dB razlika između amplitude prvog i trećeg harmonika pravougaonog valnog oblika. Slika 4.2 je izabrana tako da 'zumira' na prva tri harmonika. Primjetimo da flattop prozor daje najtačnije mjerenje relativne amplitude između harmonika.



Slika 4.2

Postavljena FFT menija su izabrana tako da izoluju prva tri harmonika pravougaonog valnog oblika. Korzori su korišteni da izmjere razliku relativnih amplituda (u dB) izmedju harmonika

- Ovaj korak će pokazati efekte aliasinga. Koristiti *Tim/Div* dugme da se izabere efektivna brzina samplovanja od 5 Msa/sec. Koristiti sliku 4.3 za pomoć oko izbora postavljenja za FFT menije. U ovom displeju dva najviša vrha predstavljaju 1-i i 3-i harmonik pravouglog valnog oblika. Ipak, kako je pokazano i na slici, viši harmonici su aliasirani i pojavljuju se kao komponente na nižim frekvencijama u displeju FFT.
- Ponoviti gornje korake za 1 V_p 500 kHz trouglasti valni oblik.



Slika 4.3

1024 tačke DFT (amplitude $2 V_p$), 500 kHz pravougaoni valni oblik sampliran sa 5 Msa/sec. Kurzori pokazuju prvi i treći harmonik. Ostali vrhovi su od većih harmonika koji pošto su aliasirani se pojavljuju na nižim frekvencijama.

Pitanja

1. Da li je nužno imati stabilan displej u vremenskom domenu da bi se analizirao frekventni sadržaj signala?
2. Da li je prethodno znanje o frekventnom opsegu signala potrebno?
3. Da li je još uvijek moguće dobiti korisnu informaciju iz FFT displeja kada se komponente signala aliasiraju.

Zaključci

1. Flattop prozor je najefikasniji za mjerenje relativnih amplituda između različitih frekvencija harmonika.
2. Nekorektan izbor efektivne brzine sampliranja može rezultirati u aliasingu u frekventnom domenu tako da se viši harmonici pojavljuju na nižim frekvencijama u FFT displeju.

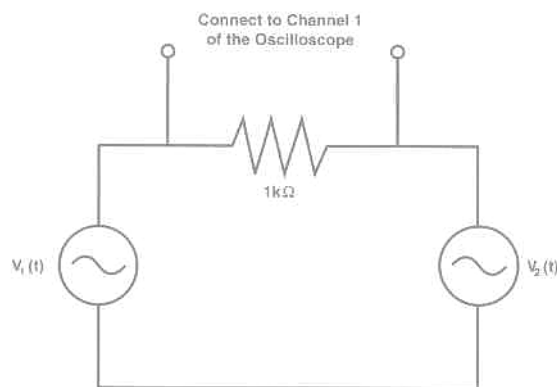
Ekspерiment 4

Suma sinusoida

Ovaj експеримент pokazuje kako FFT može biti korištena da se analizira spektralni sadržaj signala koji se sastoji od sume dvije sinusoidе. Slična analiza koristeći tehnike vremenaskog domena bila bi vrlo teško izvodljiva na osciloskopu.

Procedura

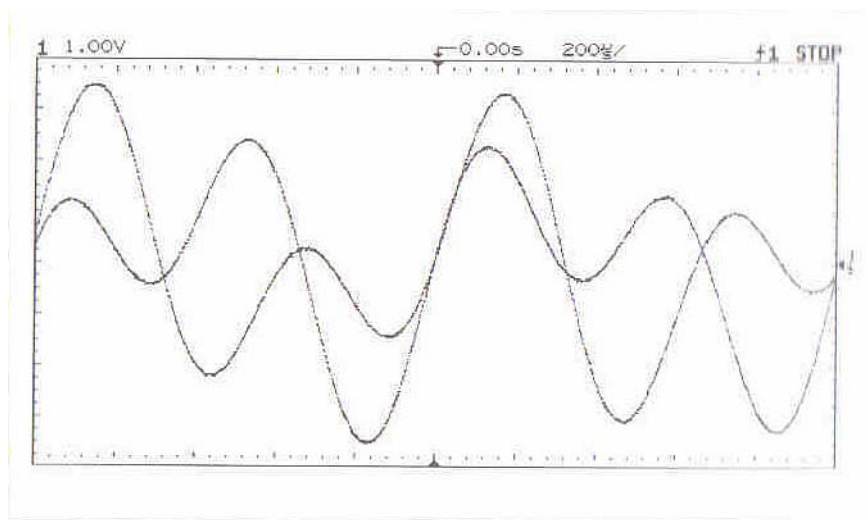
1. Koristiti dva izvora signala i otpornik da se napravi kolo pokazano na slici 5.1.
2. Koristiti osciloskop da se verificira da v_1 i v_2 valni oblici imaju korektna podešenja amplituda i frekvencija.



Slika 5.1

Naponski izvor $v_1(t)$ je $3.5 V_{pp}$, 1 kHz sinusoida, a izvor napona $v_2(t)$ je inicijalno postavljen na $3.5 V_{pp}$, 2 kHz sinusoida.

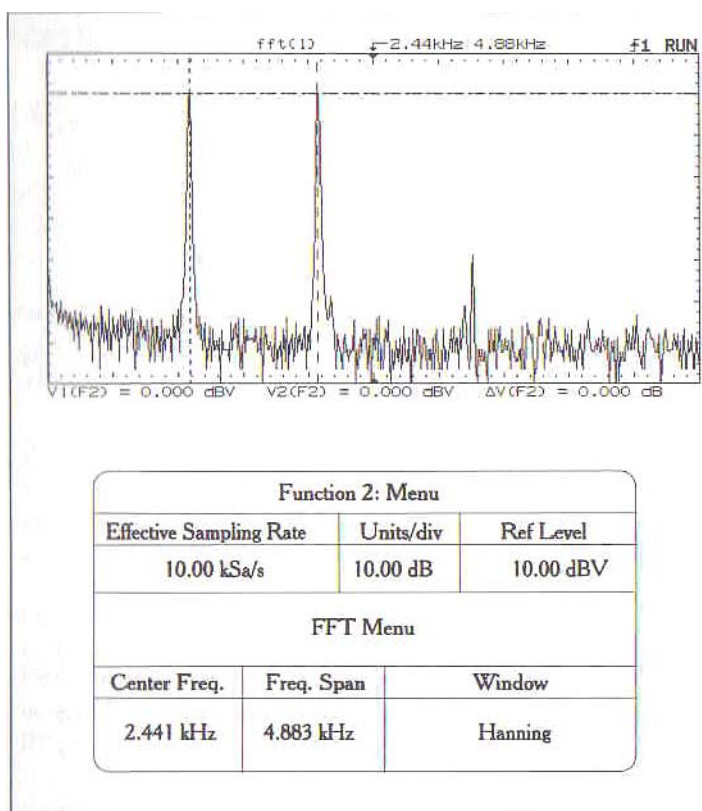
3. Koristiti *Autoscale* da se dobije displej u vremenskom domenu napona na krajevima otpornika. Pošto je napon na otporniku suma dvije sinusoidе, i pošto one imaju različite osnovne frekvencije, (koje nisu potpuno harmonijski u relaciji), rezultirajući displej u vremenskom domenu je nestabilan. Koristiti *Run* i *Stop* dugmad da bi se napravio 'snapshot' (okidni snimak) tragova osciloskopa. Ovo je ilustrirano na slici 5.2.



Slika 5.2

Displej u vremenskom domenu napona na otporniku.

4. Kao što je već opisano u prethodnim eksperimentima, uključiti FFT displej i isključiti vremenski displej na kanalu 1. Koristiti podešenja pokazana na slici 5.3 da se pokaže spektralni sadržaj ulaznog signala. Dugmad *Cursors* se mogu koristiti da se izmjeri lokacija frekvencije svake sinusoide. Interesantno je takodjer uočiti efekat promjene efektivne brzine sampliranja. Koristiti dugme *Time/Div* da se polako povećava brzina sampliranja, dozvoljavajući FFT displeju da se stabilizira poslije svakog povećanja brzine. Primjetimo da postaje sve teže razlikovati dvije sinusoide kako povećavamo brzinu sampliranja.



Slika 5.3

1024 tačke DFT signala koji predstavlja sumu dvije sinusoida. Svaka sinusoida ima napon od $3.5 V_{pp}$. Frekvencije sinusoida su 1 i 2 kHz respektivno.

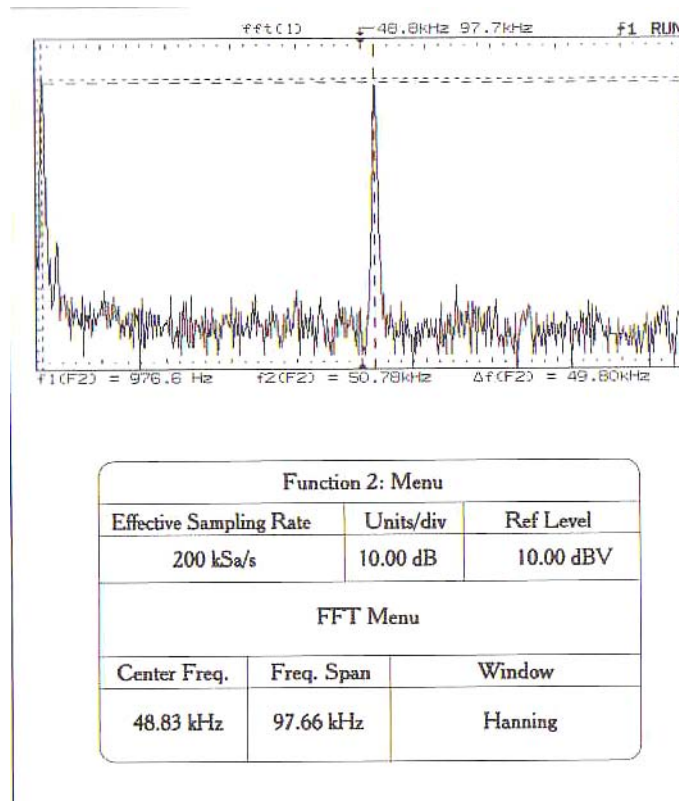
5. Podesiti frekvenciju izvora v_2 na 50 kHz. Koristiti *Autoscale* da se prikaže valni oblik u vremenskom domenu. Rezultirajući displej nije stabilan.
6. Uključiti FFT displej i isključiti vremenski displej na kanalu 1. Koristiti sliku 5.4 da se podese podešenja FFT menija. Na slici 5.4, obadvije frekventne komponente se lako razlikuju.

Pitanja

1. Kako prisustvo komponenti viših frekvencija utiče na spektralnu rezoluciju za analizu uskopojasnih komponenti signala?
2. Na bazi rezultata eksperimenta, zašto bi bilo teško koristiti FFT module da se analizira frekventni opseg 'tipičnog' amplitudno moduliranog signala?

Zaključci

1. **Stabilan valni oblik u vremenskom domenu nije potreban** kada se koristi HP 54657A FFT modul za izvođenje analize u frekventnom domenu, sve dok vremenska baza je sporija od $50 \mu s / div$.
2. Može biti teško razlučiti frekventne komponente uskog propusnog opsega kada ulazni signal sadrži komponente visoke frekvencije. Efektivna brzina sampliranja mora biti veća da bi se izbjegao aliasing, i zbog toga, rezultirajuća spektralna rezolucija je loša.



Slika 5.4

1024 tačke DFT signala koji se sastoji od dvije sinusoide sa osnovnim frekvencijama od 1 kHz i 20 kHz.

Eksperiment 5

Poredjenje funkcija FFT prozora

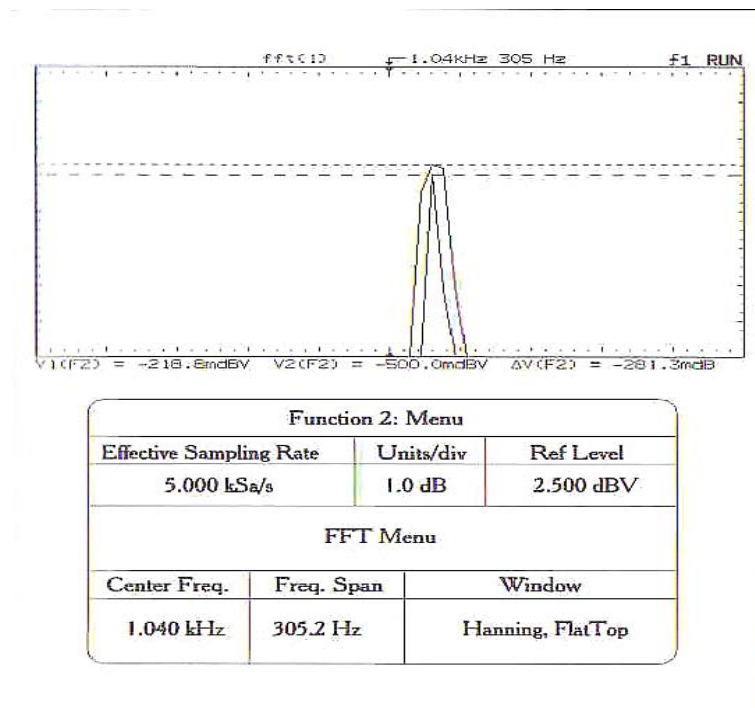
U ovom kratkom eksperimentu koristićemo Hanningov, pravougaoni i Flattop prozor da analiziramo frekventni sadržaj jednostavnih signala. Eksperiment poredi spektralne rezolucije i mogućnosti mjerenja spektralnih amplituda raznih prozora.

Procedura

1. Spojiti i prikazati 1V (RMS) sinusoidu sa osnovnom frekvencijom od 1 kHz. Ovaj eksperiment će uključiti mjerenje apsolutne amplitude sinusoide u frekventnom domenu. Zbog toga, 1 V (RMS), podešenje treba biti što je moguće tačnije. Za brzo čitanje RMS vrijednosti, pritisnuti taster VOLTS na osciloskopu i očitati RMS vrijednost.

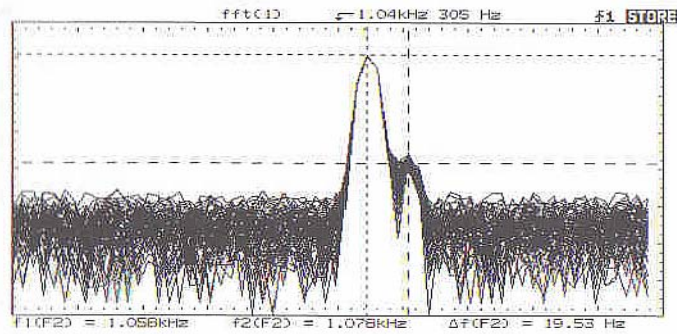
2. Uključiti FFT displej i isključiti displej u vremenskom domenu na kanalu 1. Koristiti podešenja pokazana na slici 6.1 da se izolira osnovna frekvencija sinusoide. Pošto je apsolutna amplituda skale FFT u dBV, koja je referencirana na sinusoidu od 1 V (RMS), teoretski vrh amplitude 'perfektno' sinusoide treba biti na 0 dBV. Koristiti Hanningov, pravougaoni i Flattop prozor da se izmjeri vrh amplitude. Koji od tri prozora će dati najbolje rezultate?
3. Za slijedeći dio ovog eksperimenta, spojiti jednostavno kolo iz prethodnog eksperimenta pokazano na slici 5.1. Izabrati v_1 da bude $3.5 V_{pp}$, 1 kHz sinusoida. Izabrati v_2 da bude frekvencije koja je otprilike 20 Hz veća od v_1 , i prigušiti amplitudu v_2 za otprilike 30 dB. Koristiti podešenja pokazana na slici 6.1, 6.2 i 6.3 da se demonstriraju osobine frekventne razlučivosti sva tri prozora.

Sekvenca preklapanja FFT tragova se dobije koristeći *Auto-store* osobinu osciloskopa. Primjetimo da prikazivanjem nekoliko preklapajućih FFT tragova, moguće je vrlo jasno identificirati obadvije harmonijske komponente. Također, primjetimo da Hanningov prozor je najefikasniji u identifikaciji frekventnih komponenata niskog energetskog nivoa.



Slika 6.1

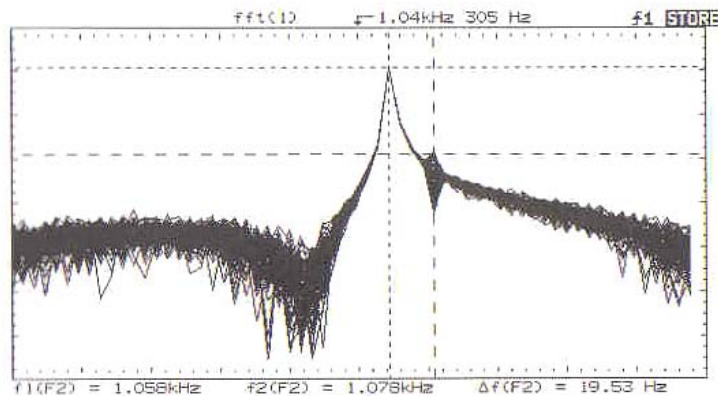
1024 tačke DFT sinusoide amplitude 1 V (RMS), sa osnovnom frekvencijom 1 kHz. Trace mogućnosti osciloskopa su korištene da se simultano prikažu rezultati Hanningovog i Flattop prozora.



Function 2: Menu		
Effective Sampling Rate	Units/div	Ref Level
5 kSa/s	10.00 dB	10.00 dBV
FFT Menu		
Center Freq.	Freq. Span	Window
1.045 kHz	305.2 kHz	Hanning

Slika 6.2

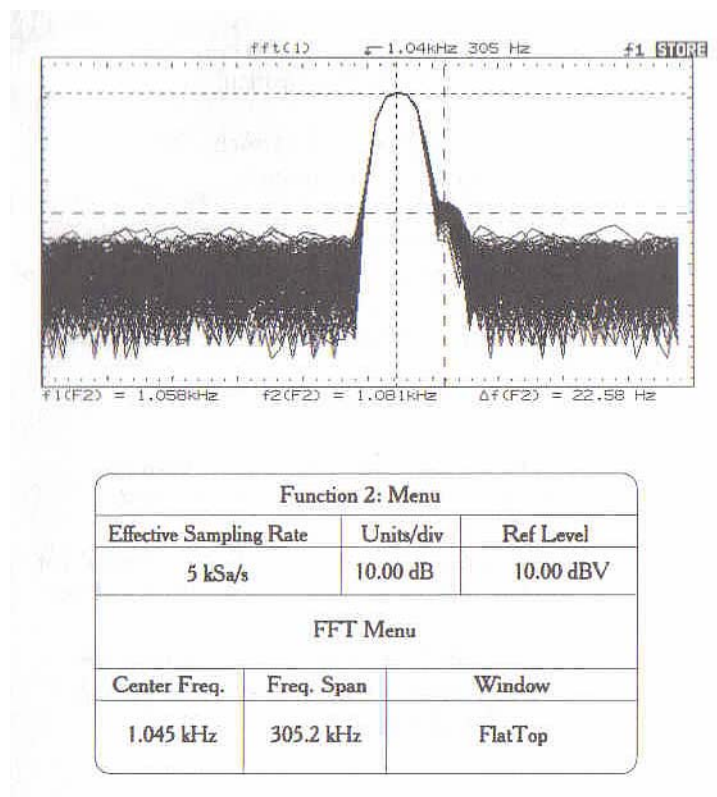
1024 tačke DFT sume dvije sinusoide koristeći Hanningov prozor. Dvije komponente su pomjerene za oko 20 Hz i sinusoida na nižoj frekvenciji je prigušena za oko 30 dB. *Autostore* osobina osciloskopa je korištena da se prikaže sekvenca preklapajućih FFT tragova.



Function 2: Menu		
Effective Sampling Rate	Units/div	Ref Level
5 kSa/s	10.00 dB	10.00 dBV
FFT Menu		
Center Freq.	Freq. Span	Window
1.045 kHz	305.2 kHz	Rectangular

Slika 6.3

Isto kao i slika 6.2 samo što se koristi pravougaoni prozor



Slika 6.4

Isto kao i slika 6.2 samo sa Flattop prozorom

Pitanja

1. Rangirajte tri prozora korištena u ovom eksperimentu u terminima njihove efektivnosti u tačnom mjerenju spektralnih amplituda.
2. Rangirajte tri prozora korištena u ovom eksperimentu u terminima njihove sposobnosti da razluče između blisko postavljenih frekventnih komponenata.
3. Periodogram usrednjavanja je uobičajena DSP tehnika koja se koristi u spektralnoj analizi. Periodogram se dobije kvadriranjem FFT amplitude dijela vremenskog prozora ulaznog signala. Diskutirajte relaciju između periodograma usrednjavanja i displeja dobijenih na slikama 6.2 do 6.4

Zaključci

1. FFT analizator je sposoban da razluči relativno blisko postavljene frekventne komponente, čak i kada je jedna komponenta prigušena.
2. Koristeći Auto-Store mogućnosti osciloskopa, moguće je koristiti 'vizuelno' usrednjavanje da se izdvoje efekti šuma i podrhtavanja samplera (sampling jitter).
3. Hanningov prozor je najefektivniji prozor za razriješenje frekventnih lokacija blisko postavljenih sinusoida, dok Flattop prozor je najefektivniji za provodjenje mjerenja amplitude.
4. RMS vrijednost valnog oblika nije ista kao vrijednost glavnog loba. RMS vrijednost je afektirana glavnim lobom, cjelokupnim harmonijskim sadržajem i internim šumom osciloskopa.

POGLAVLJE 18

PRIKUPLJANJE I OBRADA PODATAKA NA PC-u KORIŠTENJEM IOTECH MODULA DAQBOARD/200A, DBK15 I DBK20

Nastavak prezentacije DAQ baziranih sistema za prikupljanje i obradu podataka u okviru LabView-a biće posvećen slijedećim IOtech proizvodima:

- modul DaqBoard/200A
- interfejsni moduli DBK15 i DBK20

IOtech kompanija dizajnira i proizvodi PC-bazirane module za prikupljanje podataka i mjerne instrumente. Njihovi proizvodi se koriste u širokom rasponu testnih aplikacija i služe kao mjerni setovi u industrijama uključujući automobilsku, zračnu, kemijsku, komunikacijsku, elektronsku i mnoge druge.

1. HARDVERSKI OSNOVI MODULA FAMILIJE DAQBOARD [ISA-SLOT]



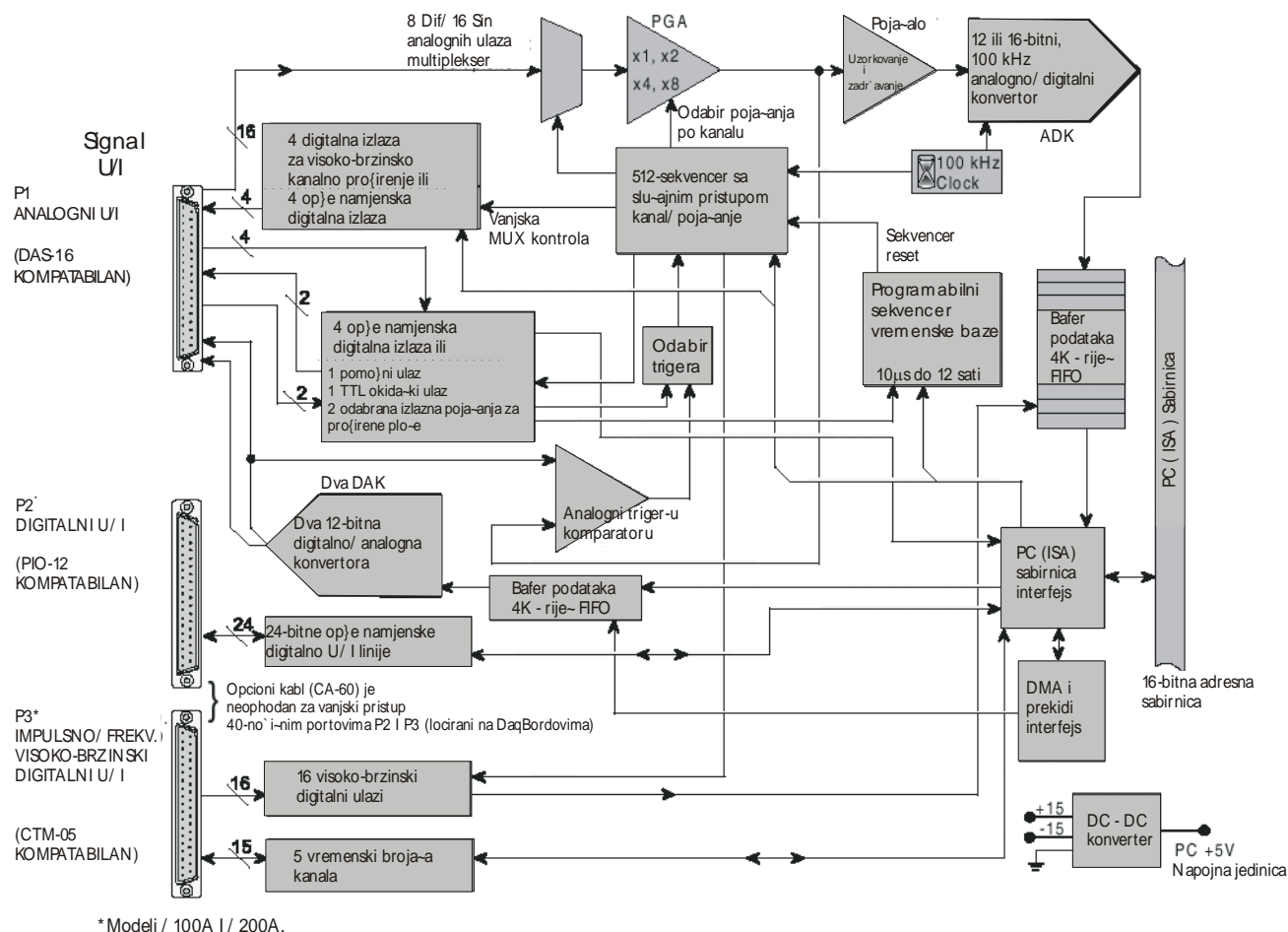
SLIKA 1.1. DAQBOARD

DaqBoard/100, /112A, /200A, i /216A su 100 kHz ISA – slot moduli koji obezbjeđuju analognu-digitalnu konverziju sa širokim izborom prilagođavanja signala, proširenje signala i softversku podršku od bilo kog PC-baziranog sistema za dobijanje informacija.

Moduli familije DaqBoard [ISA-slot] obezbjeđuju širok spektar ugrađenih analognih i digitalnih ulaza/izlaza. Svi moduli DaqBoard obezbjeđuju 16 analognih ulaza (proširivo do 256), 2 analogna izlaza i 4 digitalna ulaza i izlaza.

RAZLIČITI MODULI DAQBOARD IMAJU 12-BITNU ILI 16-BITNU REZOLUCIJU:

- 12-BITNI MODULI UKLJUČUJU DAQBOARD/100A I /112A.
- 16-BITNI MODULI UKLJUČUJU DAQBOARD/200A I /216A

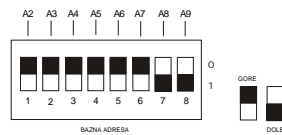


Slika 1.2. Blok dijagram za DaqBoard modele /100A, /112A, /200A, /216A

Moduli familije DaqBoard [ISA-tip] mogu skanirati kanale frekvencijom od 100kHz, i to im omogućava da akviziraju različite tipove mjernih davača od termopara do mjerača deformacija. Imaju širok izbor trigerovanja i obezbjeđuju programabilno kašnjenje od 10μs do 10 sati. Za svaki kanal obezbjeđuje se programabilno pojačanje, i na vrlo jednostavan način vrši se konfigurisanje prema željenim zahtjevima.

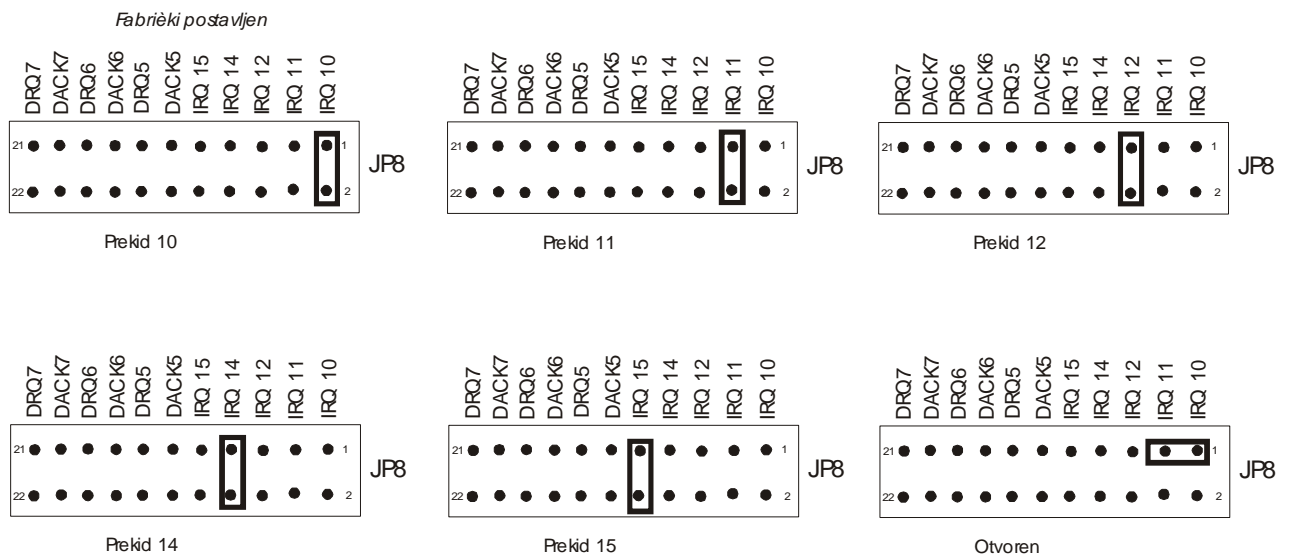
INSTALACIJOM MODULA DAQBOARD U RAČUNAR MORA SE PRILAGODITI UNUTRAŠNJI (INTERNI) SISTEM KONFIGURACIJE. UNUTRAŠNJA KONFIGURACIJA SAS TOJI SE OD POSTAVLJANJA PREKIDAČA I KRATKOSPOJNIKA TAKO DA ODGOVARAJU ŽELJENOM MODU RADA:

- BAZNA ADRESA

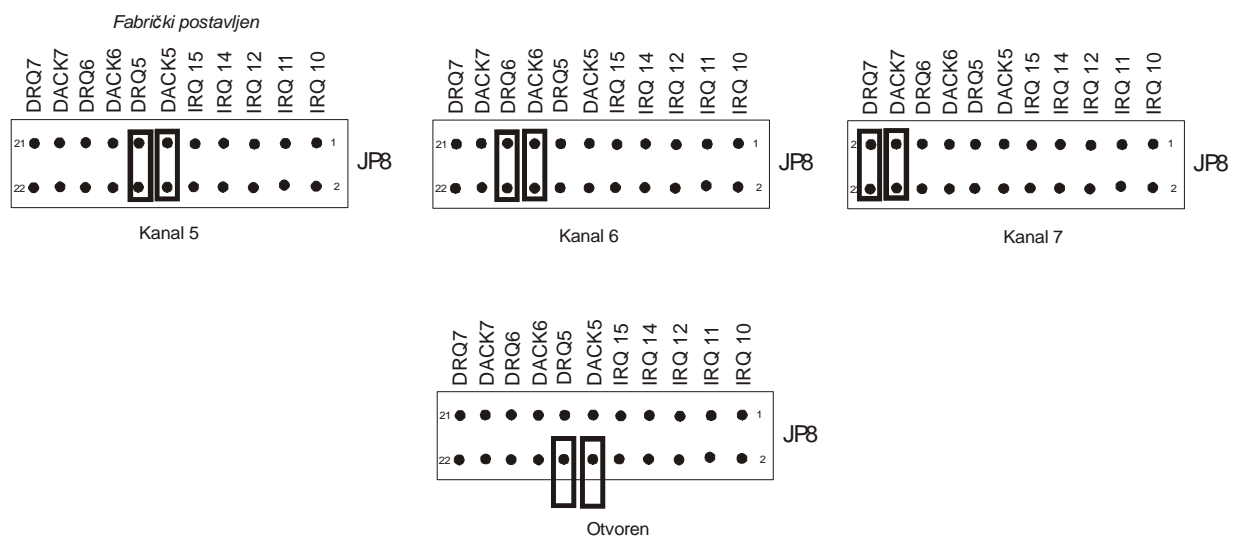


Slika 1.6. Fabrički postavljena vrijednost bazne adrese DaqBoard-a

- PREKID I DMA

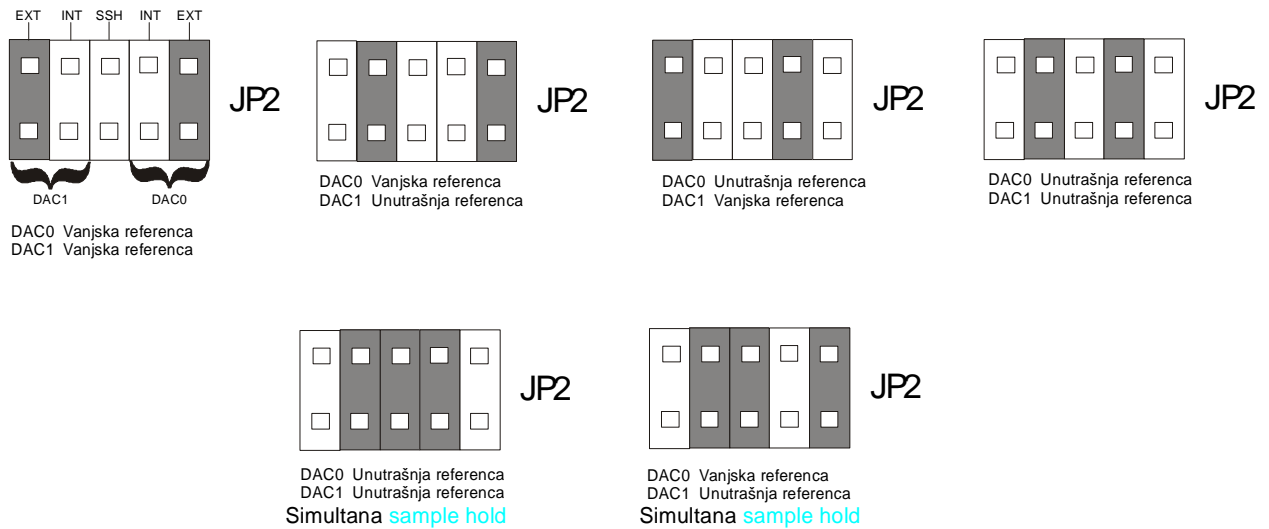


Slika 1.8. DaqBoard prekidi



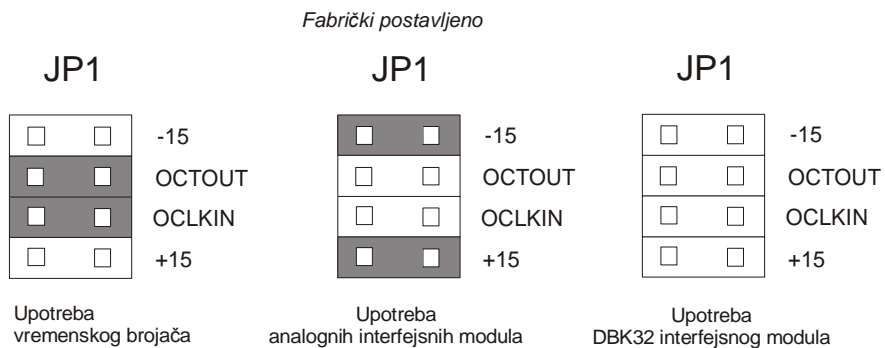
Slika 1.9. DaqBoard DMA postavka

• DAC



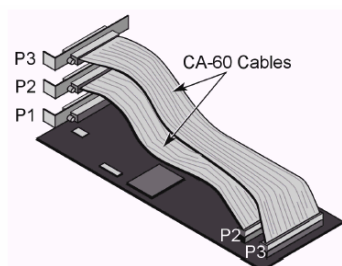
SLIKA 1.10. POSTAVKA DAC REFERENCE

• VANJSKO ANALOGNO PROŠIRENO NAPAJANJE

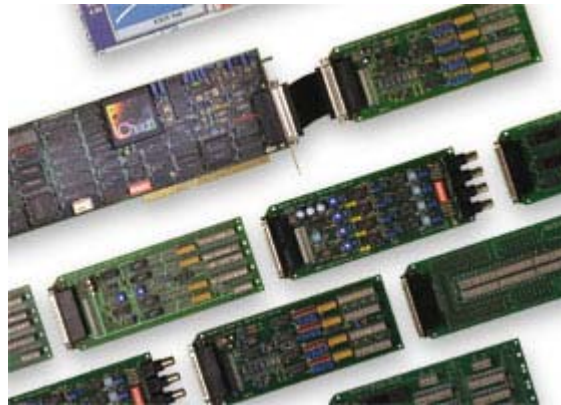


Slika 1.11. Postavka vanjskog analognog proširenog napajanja

MODULI DAQBOARD PRIHVATAJU SVE ANALOGNE I DIGITALNE ULAZNO/IZLAZNE SIGNALNE PREKO JEDNOG STANDARDNOG U/I KONEKTORA P1 I DVA 40-IGLIČASTA KONEKTORA (P2 I P3) NA PLOČI MODULA.



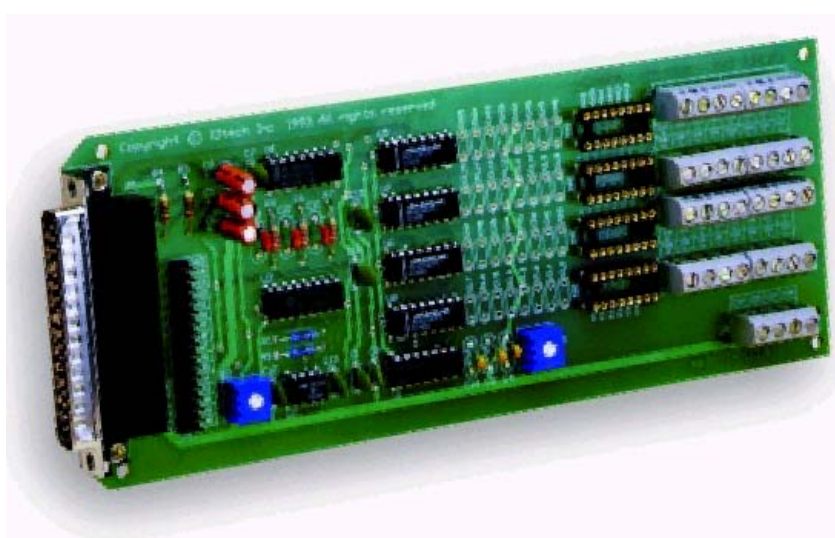
2. INTERFEJSNI MODULI DBK



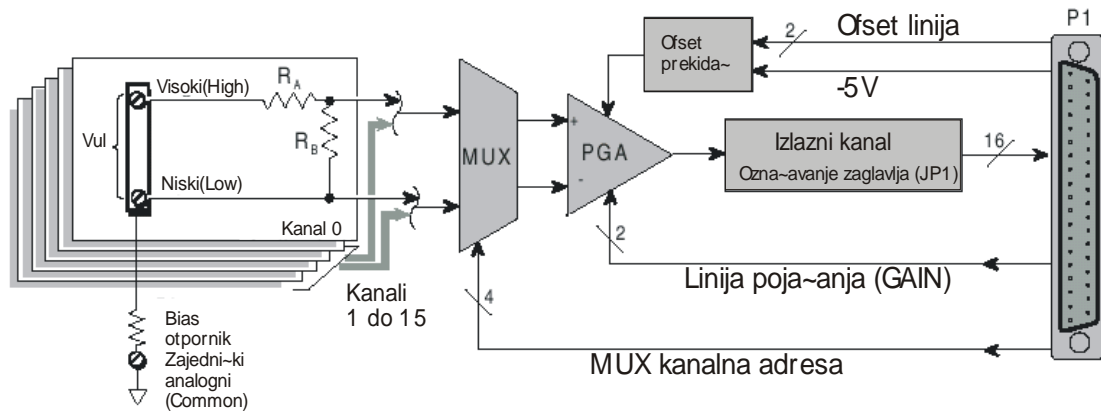
Slika 2.1. Interfejsni moduli

DBK serija interfejsnih modula za obradu signala su dizajnirani za upotrebu sa DaqBoard ISA-tip baziranim pločama i DaqBook portabl sistemima za prikupljanje podataka. DBK serija može biti korištena sa pločama baziranim na ISA-sabirnicama od proizvođača kao Keithly/MetraByte, Computer Boards, Avantech i Omega.

DBK serija se sastoji od četiri elementa: interfejsnih modula za obradu signala, pakovanje, napajanje i moduli za obradu signala sa visokim kapacitetom. Univerzalno strujno/naponski ulazni interfejsni modul DBK15 koristi 16 kanalni multiplexer (MUX) i programabilni ulazni pojačavač (PGA). Ako je konfigurisan sa odgovarajućim otpornicima, DBK15 može mjeriti napon do 30V istosmjerne struje (30 VDC) ili struje do 20mA. Ulazni pojačavač interfejsnog modula DBK15 je softverski programabilan za x1 ili x2 pojačanje po kanalu. Na DaqBoard se može konektovati kako jedan tako i više modula DBK15, maksimalno šesnaest.

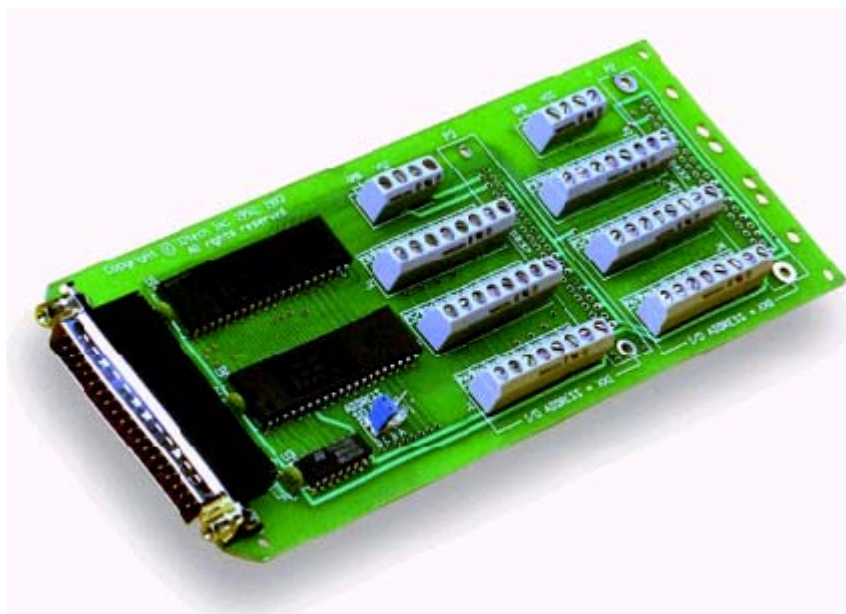


Slika 2.3. DBK15

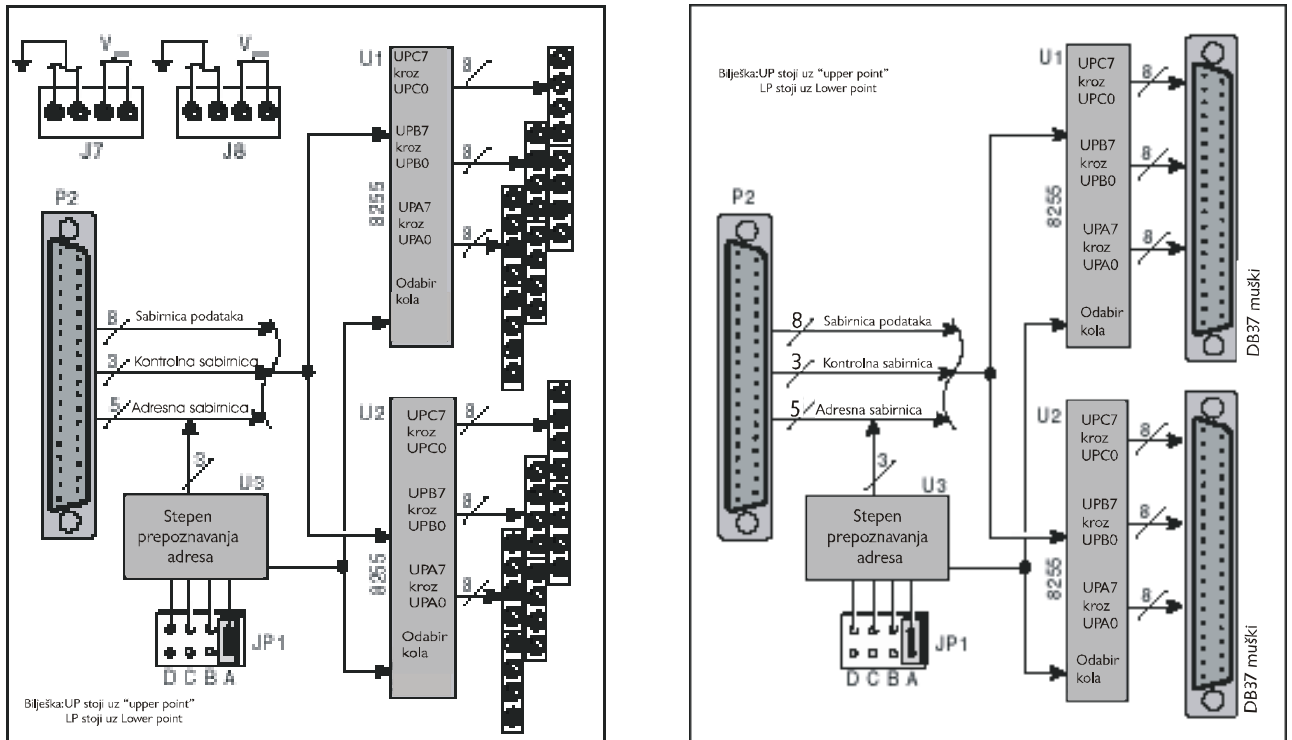


Slika 2.4. Blok dijagram DBK15

DBK20 i DBK21 su opšte namjenski, digitalni ulazno/izlazni moduli koji mogu povećati broj raspoloživih digitalnih ulazno/izlaznih (U/I) linija za 48 po modulu. DBK20 koristi konektore sa pod vijak, dok DBK21 koristi DB37 konektore. Oba modula se povezuju na P2 digitalni ulazno/izlazni (U/I) port sa glavnom jedinicom (DaqBoard) preko interfejsnog kabla (CA-37-x). Do četiri interfejsna modula DBK20 ili DBK21 mogu biti spojena na maksimum 192 linije (4x48).

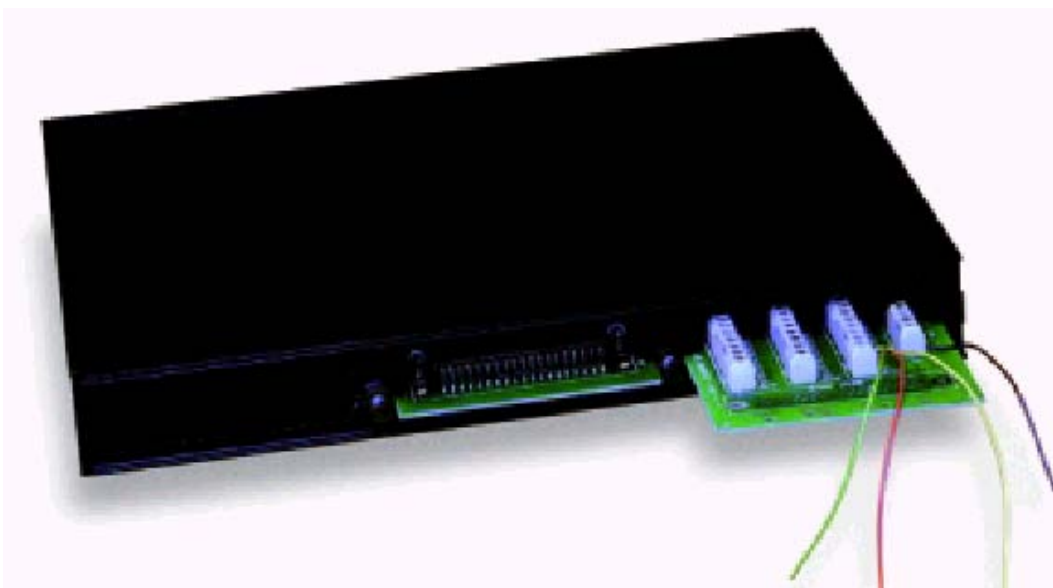


Slika 2.9. DBK20

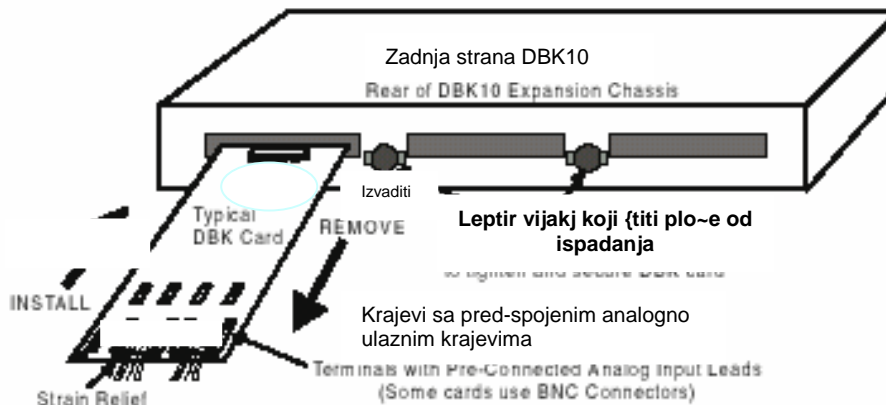


Slika 2.10. Blok dijagram interfejsnih modula DBK20 i DBK21

DBK10 je metalna kutija za proširenje koja prihvata do tri interfejsna modula. Više kutija može lako da se postavi tako da prihvati bilo koji broj interfejsnih modula.



Slika 2.13. DBK10



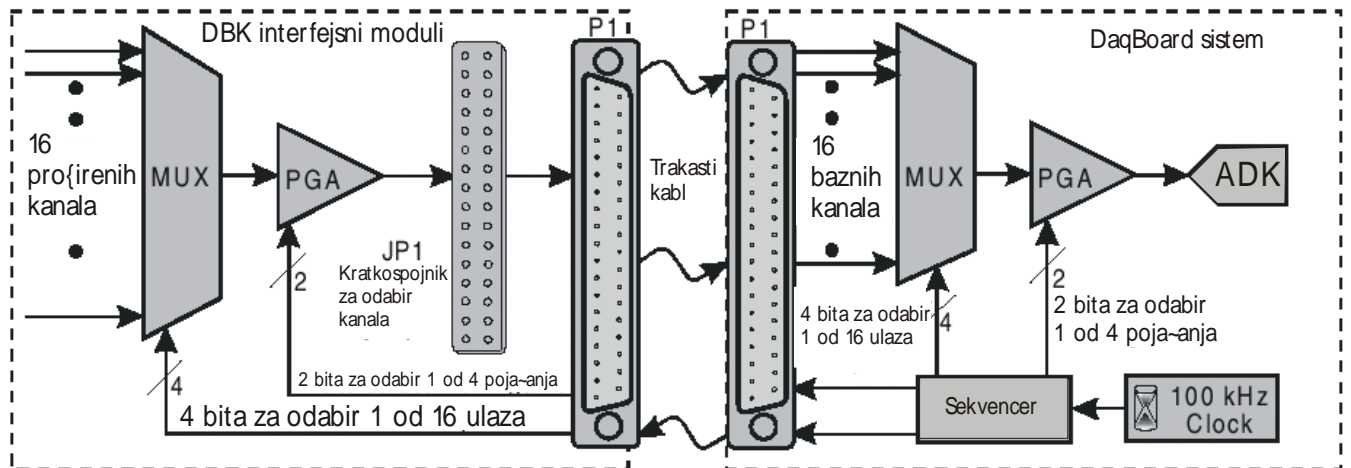
Slika 2.14. Instalacija interfejsnih modula DBK u DBK10 kutiju

3. OBRADA SIGNALA U DAQ SISTEMIMA

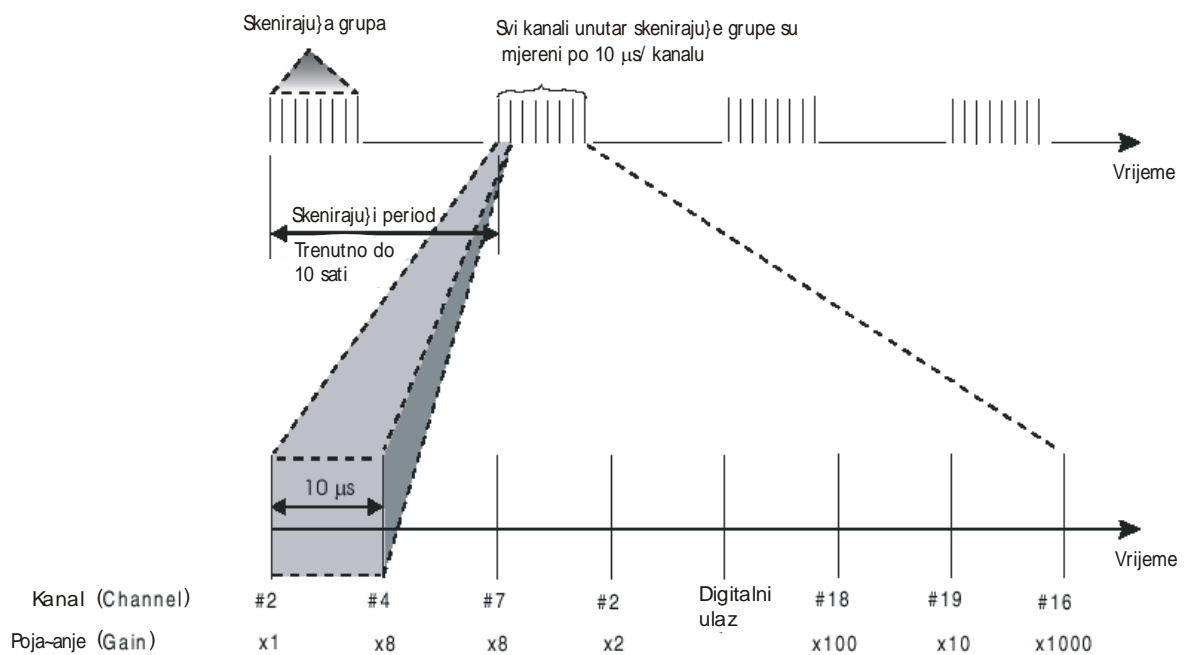
Sistem za prikupljanje (akviziciju) podataka koji ima modernu koncepciju sastoji se od: multipleksera, analogno-digitalnog konvertora, digitalno-analognog pretvarača (D/A), kola za uzorkovanje (*Sample/Hold*), pojačavača, brojača/tajmera (vremenskih kola), i drugih specijalnih kola.

Jedna od najvažnijih karakteristika sistema za akviziciju podataka za personalne računare je ta, što sva ta specijalna kola integrišu u jedan kompatibilan i kompaktan sistem.

Interfejsni moduli mogu povećati broj analogno ulaznih linija, od 16 baznih kanala do 256 ulaznih kanala. Svaki interfejsni modul DBK obezbjeđuje jedinstveni izlaz koji mora biti usmjeren na jedan od 16 baznih kanala. Takođe DaqBoard-ov sekvencer, pored toga što kontroliše DaqBoard također kontroliše programabilne osobine na interfejsnim modulima. Ova arhitektura osigurava isto uzorkovanje od 10 μ s kako za vanjske tako i za unutrašnje kanale.

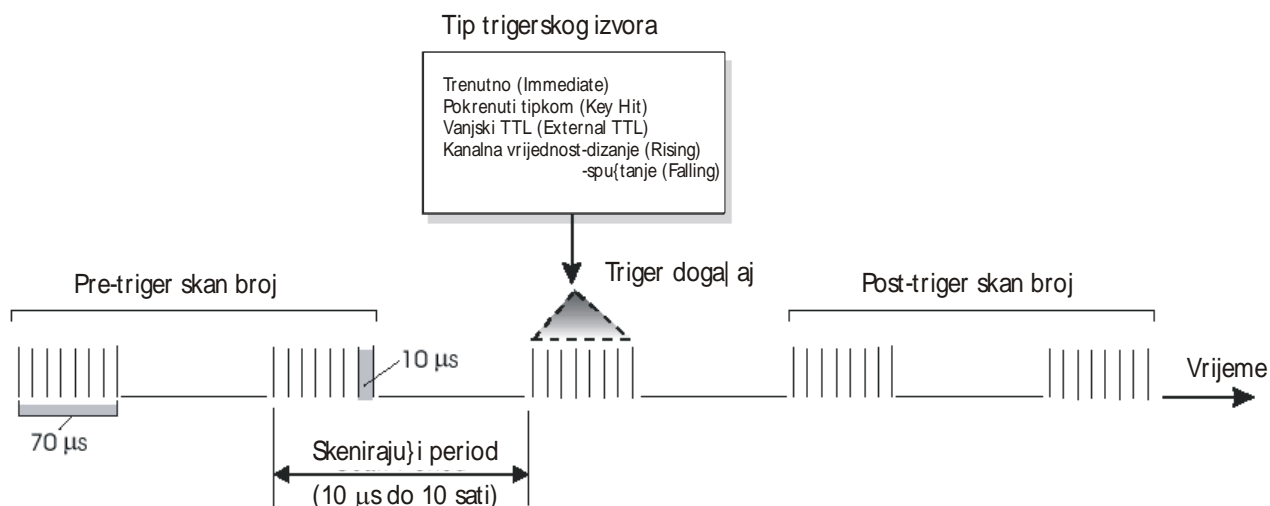


Slika 3.4. Funkcionalni dijelovi Daq sistema



Slika 3.7. Dijagram skaniranja

Ciklus prikupljanja se kontroliše trigerovanjem. Zahtjev trigerovanja se određuje po prirodi mjerenja i količini potrebnih podataka.

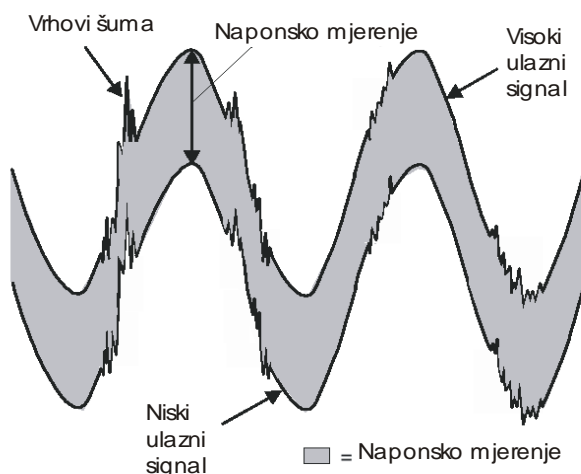


Slika 3.8. Parametri triggerovanja

Brojačko-vremenska kola u DaqBoard-u se koriste za brojanje digitalnih događaja, sinhronizaciju digitalnih impulsa i generisanje pravogaunih oblika i impulsa.

Korištenjem ulazne izolacije imaju se prednosti kao što su: zaštita kola, smanjenje šuma i potiskivanje zajedničkog napona (CMV).

Diferencijalni režimi sa osnovom uzemljenja, šentiranja i plivajući otklanjaju greške koje mogu da se jave pri mjerenju.



Slika 3.10. Zajednički naponski režim: poništavanje šuma sa diferencijalnim mjerenjem

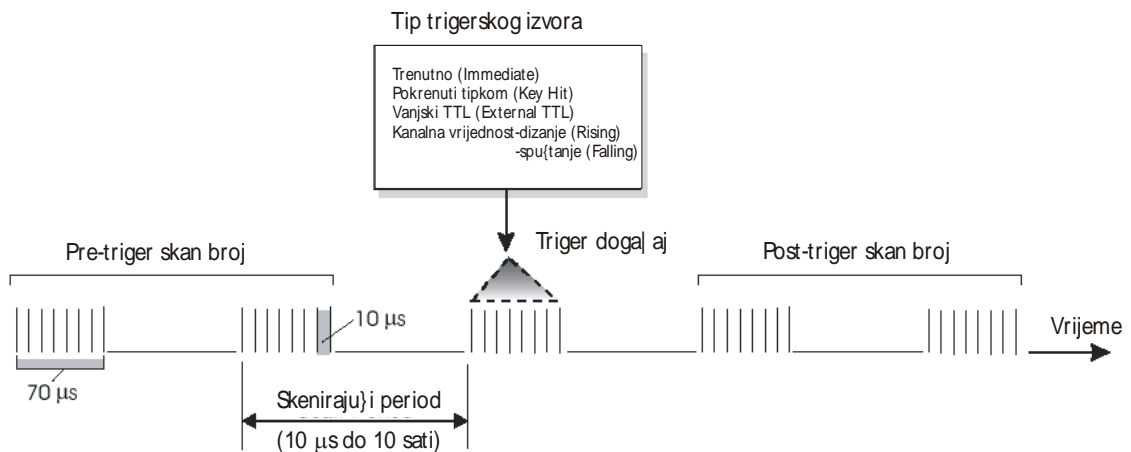
Brzina skaniranja (scan rate)

Najveći dio analognih signala dešava se u vremenskoj bazi DaqBoard sata. Skanirajući period je vremensko trajanje između uzastopnih (sukcesivnih) skanova. Inverzno, skanirajuća stopa ili skanirajuća frekvencija je broj skanova u

vremenskom intervalu, obično izraženim u skanovima u sekundi. Kanali u skanu su uvijek uzorkovani u istom periodu od $10\mu\text{s}$ (100kHz brzina), ili $5\mu\text{s}$ (200kHz brzina). Uopšteno frekvencija uzorkovanja mora biti veća od dvostruke najveće frekvencije signala, da se spriječe greške.

Trigerovanje (Triggering)

Trigerovanje kontroliše ciklus prikupljanja. Kad je sistem spreman, potreban je triger da se prikupе podaci. Tipično, specificirana su tri parametra za prikupljanje podataka: pred trigerski broj, poslije trigerski skan broj i izvor trigera. Mora se odrediti zahtjev triggeriranja, baziran na prirodi mjerenja i količini podataka potrebnih za zadovoljenje sistemskog cilja.



Parametri triggerovanja

- Pre-triger skan broj specificira broj skanova koji treba da se skupe prije tačke triggerovanja. Ako je pred triger skan broj veći od nule, sistem će kontinuirano prikupljati podatke dok se triger ne zadovolji. Ako se ne zahtjevaju pred trigerski skanovi, sistem ostaje nepromjenjen do trigera, onda on skuplja post-trigerske skanove dok se ne završi.
- Post-triger skan broj specificira broj skanova koji treba da se skupe poslije tačke triggerovanja. Poslije trigera post-trigerski skanovi bit će prikupljeni kao što je isprogramirano i onda će se sistem zaustaviti.
- Izvor trigera može biti softverska komanda, vanjski TTL ulaz itd.

Brojačko/vremenske funkcije (Counter/Timer)

U praksi postoje primjene koje, pored ostalog, zahtjevaju brojanje, vremensko usklađivanje događaja (tajming) i mjerenje frekvencije. Postoje i primjene koje zahtjevaju da se uređaji uključuju i isključuju u tačno određenim trenucima, ili za određeni vremenski period. Ove funkcije se ostvaruju pomoću kola koja nazivamo COUNTER/TIMER (C/T). Sistemski C/T su optimizovani za primjene koje daju impulse, mjere frekvenciju i generišu vremensku bazu.

DaqBook/100/200/260 i DaqBoard/100A/200A/2000 imaju brojačko/vremenske funkcije na P3 interfejsu. Ove funkcije se implementiraju na 9513 čipu i pristupa im se preko DaqView-a i programiranja. Sa odgovarajućim programiranjem pet kanala 9513 može biti nezavisno konfigurisano u različite operacione režime.

Nožice na P3 konektoru vremenskog brojača		
Nožica	Ime signala	Objašnjenje
12	CTR 5 GATE	Brojač 5 gate (9513 čip)
13	CTR 5 IN	Brojač 5 ulaz (9513 čip)
14	CTR 4 GATE	Brojač 4 gate (9513 čip)
15	CTR 4 IN	Brojač 4 ulaz (9513 čip)
16	CTR 3 GATE	Brojač 3 gate (9513 čip)
17	CTR 3 IN	Brojač 3 ulaz (9513 čip)
18	CTR 2 GATE	Brojač 2 gate (9513 čip)
19	CTR 2 IN	Brojač 2 ulaz (9513 čip)
30	OSC. OUT	Oscilator izlaz (9513 čip)
31	CTR 5 OUT	Brojač 5 izlaz (9513 čip)
32	CTR 4 OUT	Brojač 4 izlaz (9513 čip)
33	CTR 3 OUT	Brojač 3 izlaz (9513 čip)
34	CTR 2 OUT	Brojač 2 izlaz (9513 čip)
35	CTR 1 OUT	Brojač 1 izlaz (9513 čip)
36	CTR 1 IN	Brojač 1 ulaz (9513 čip)
37	CTR 1 GATE	Brojač 1 gate (9513 čip)

Tablica 3.2 Raspored nožica P3 za brojačke kanale.

Svaki kanal ima tri linije.

- IN – digitalni ulaz koji inkrementira brojač i obezbjeđuje vremensku bazu za operacije brojača.
- GATE – digitalni ulaz koji uključuje ili isključuje counter/tajmer.
- OUT – izlaz digitalnih četvrtki i impulsa

Ulazna izolacija

Za korištenje ulazne izolacije imaju se slijedeće prednosti :

zaštita elektronike od vanjskog sistema, smanjenje šuma i potiskivanje zajedničkog napona (CMV).

- Zaštita kola:

Ulazna izolacija razdvaja izvor signala od kola koja mogu biti oštećena signalom. Naponi veći od 10V mogu oštetiti kola. Visoko naponski signali zato trebaju biti izolovani od naponskog nivoa elektronike.

- Smanjenje šuma:

Izolacija eliminiše petlje uzemljenja za visoko pojačavačke sisteme koji imaju više jedinica. Šasije za svaki uređaj mogu biti na potencijalu zemlje, s tim da se malo razlikuju od drugih uređaja. Ove irelevantne za mjerenje struje i vrhovi koji se pojavljuju zbog indukcije mogu biti čuvani van mjernih kola.

- Potiskivanje zajedničkog CMV ("sinfaznog") napona (Common-mode voltage)

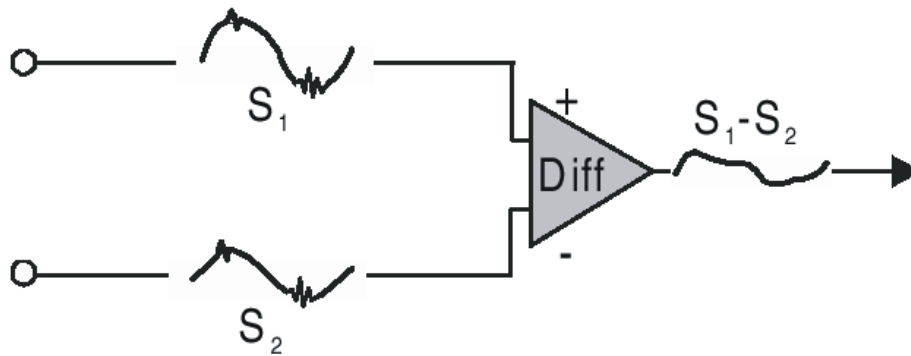
Postoji ograničenje u naponu koji diferencijalni pojačivač može podnijeti između zemlje i ulaza pojačivača. Na sreću diferencijalni pojačivač odbacuje visoki zajednički naponski režim signala. Visoki zajednički naponski režim (High common-mode voltage) i vrhovi šuma (špicevi) su odbačeni u fazi signala u amplitudi i frekvenciji.

Režimi signala

Daq jedinica radi u dva režima, nesimetrični (single-ended-SE) i diferencijalni. Daq jedinice mogu prihvatiti 8 diferencijalnih ili 16 ulaznih analognih nesimetričnih signala. Postavke podrazumjevanih vrijednosti interfejsnih modula DBK i Daq modula koriste 16 analognih nesimetričnih ulaza. Neki interfejsni moduli DBK koriste diferencijalne ulaze za mjerne davače ali izlazi interfejsnih modula DBK u Daq su uvijek nesimetrični. Za DaqBook/100,/112,/120, postavke kratkospojnika određuju signalne režime.

Nesimetrični režim je fabrički postavljena (default) konfiguracija. Za DaqBoard i DaqPC-karticu biranje između diferencijalnog i nesimetričnog režima je softverski u programu. Nesimetrični režim se odnosi na napon koji se mjeri između jedne signalne linije i zajedničkog napona uzemljenja. Mjereni napon može biti djeljen sa drugim kanalima.

Prednost nesimetričnog režima nad diferencijalnim je ta da on obezbjeđuje veći broj kanala 16 u odnosu na 8. Diferencijalni režim se odnosi na režim u kom se napon mjeri između dvije signalne linije. Prednost korištenja diferencijalnog ulaza je da on smanjuje greške indukcije, šuma koji nastaje zbog struje uzemljenja. Na narednoj slici dat je primjer kako se šum redukuje i isključuje kada se koristi diferencijalni režim



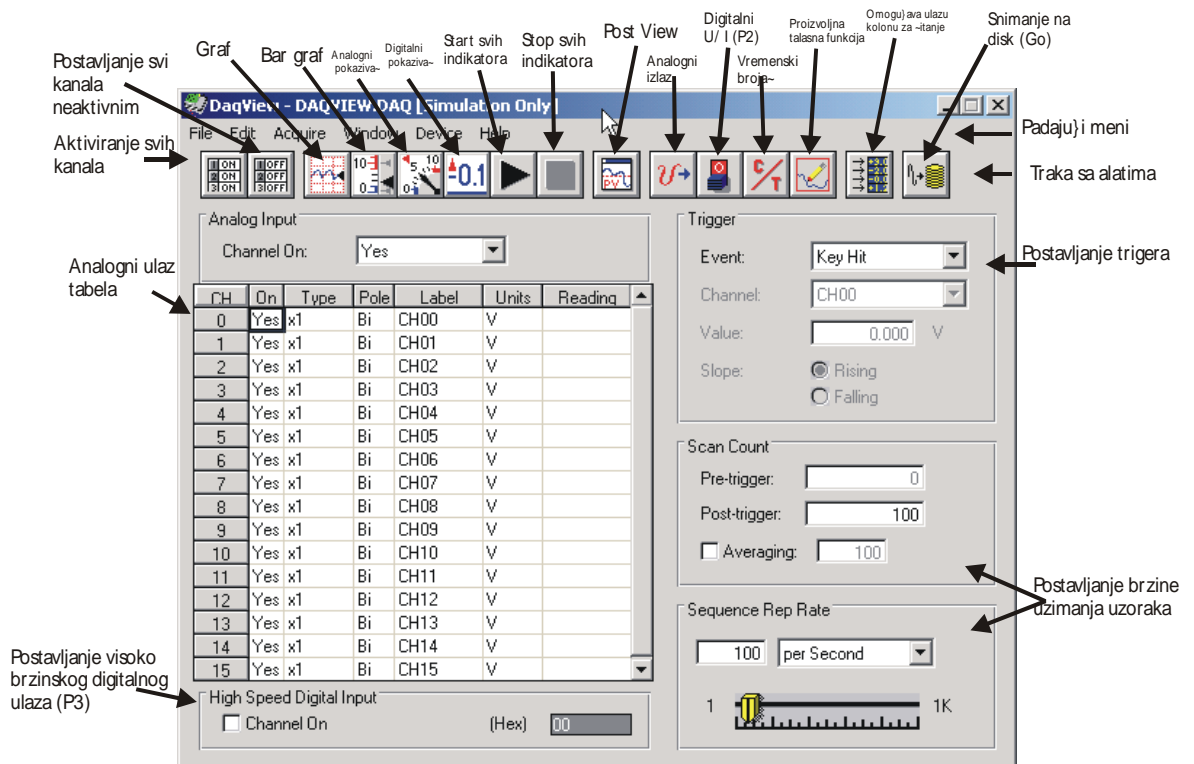
Signal S_2 se oduzima od signala S_1 i tako se dobija izlazni signal. Vrhovi šuma sa istim polaritetom, fazom i amplitudom u svakom ulaznom signalu se poništavaju, rezultujući istim diferencijalnim signalom. Na slici su S_1 i S_2 signali prikazani u fazi međutim ako su ovi signali izvan faze, šum u svakoj liniji bi imao istu amplitudu, fazu i polaritet i zbog tog razloga ipak bi bio poništen.

PROGRAM ZA PRIKUPLJANJE PODATAKA - DAQ VIEW

Daq View je program za prikupljanje podataka zasnovan na Windowsima, za rad DaqBook, DaqBoard-a ili DaqPCMCIA sa DBK interfejsnim modulima.

Daq View omogućava:

- Postavljanje sistemskih parametara (biranje kanala, pojačanja, tipovi mjernih davača –transducera, itd.) za dobijanje podataka.
- Pohranjivanje podataka na disk i prenos podataka u tabele i baze podataka.
- Konfiguracija i rad sa DBK interfejsnim modulima.
- Konfigurisanje vremenskih brojača za mjerenje frekvencije ili generisanje povorke impulsa.
- Korištenje dva analogna izlaza, uključujući generator funkcija za module Daq Board.
- Korištenje digitalnog U/I (za module DaqBoard sa digitalnim U/I).
- Aktiviranjem Post View-a mogu da se prate talasni oblici za maksimalno 16 kanala.



Slika 4.5. Glavni prozor DaqView-a

Startanje Daq View-a

Program mora biti pravilno instalisan, ako je potrebno pogledati poglavlje instalacije. Računarski zahtjevi uključuju:

- 386 PC/AT ili više

- 8 MB RAM-a
- Win. 3.x , Windows 9.x , Windows NT, Windows 2000
- IOtech hardver za prikupljanje podataka
- (za Daq ViewXL) Excel 5.0 ili više

Da bi pokrenuli aplikaciju potreban je klik na DaqView čija je lokacija u Start ⇒ Programs ⇒ DaqX Software programskoj grupi. Zavisno od postojećih datoteka DaqView startat će na jedan od tri načina:

- Automatsko biranje uređaja i puno podešavanje konfiguracije.
- Automatsko biranje uređaja i podrazumjevano podešavanje konfiguracije
- Ručno biranje uređaj i podrazumjevano podešavanje konfiguracije.

Kratak pregled DaqView-a

U ovom pregledu DaqView-a pokazat će se neke od njegovih osnovnih karakteristika.

- Otvoriti DaqView kao što je gore objašnjeno. Ako DaqBoard i DBK jedinice za kondicioniranje signala nisu spojene, odabrati *Simulated Instrument* kao uređaj.
- Ako je odabran režim *Simulated Instrument* , kolona čitanja analogno ulazne tabele (Spreadsheet) pokazaće simulirane podatke.
- Ako je hardver konektovan očitavanja će izraziti aktuelni signal.
- Sa alatne trake (Toolbar), odabrati dugme Bar Graph Meters, Analog Meters ili Digital Meters i onda trugaoni start indikator na traci alata novog prozora. Dolazeći podaci će se prikazati u formatu koji se odabrao. Poslednja stavka (ikona) na traci alata "meters" prozora omogućava da se postavi broj kanala koji će se prikazivati.
- Sa DaqView-e alatne trake glavnog prozora odabrati *Charts* (treća ikona s lijeva), i *DaqView Chanel Display* prozor će se pojaviti. Ako je neaktivan prvo se mora uključiti željeni kanal u padajućem meniju lijevo od grafa (chart). Za startovanje i zaustavljanje prikaza podataka su START i STOP dugmadi. Poslije zaustavljanja (STOP dugme) može se resetovati broj kanala koji se prikazuje.
- Kao uvod u sistemsku konfiguraciju, izabrati uređaj sa padajućeg menija *Device (Select Device)* i onda konfigurisati hardverske postavke (*Configure Hardware Settings*). Na lijevoj strani prozora *Configure Hardware Settings* , odabrati padajući meni bilo kog kanala da bi se otkrila opcija vanjske konekcije. Pored direktne signalne konekcije, mogu se birati i DBK interfejsni moduli. Desna strana prozora postavlja digitalnu konekciju ako je Daq opremljen sa njom.

Analogno ulazna tabela (Spreadsheet)

Analogno ulazna tabela (lijeva polovina glavnog prozora) prikazuje analogno ulazne kanale i dozvoljava nam da ih konfiguriramo. Svaki red pokazuje jedan kanal i njegovu konfiguraciju. Broj redova može da varira, ali svaki red ima sedam kolona. Neke kolone omogućavaju da blokovi ćelija budu promjenjeni u isto vrijeme (klikanjem miša na zaglavlje kolone može se označiti cijela kolona). Druge kolone omogućavaju izmjene samo pojedinačnih ćelija.

Slijedeća tabela sumira funkcije svake kolone.

Kolona	Upotreba i opis
CH	Ova kolona prikazuje kanalni broj (nemože se mijenjati iz ovog prozora). Ovaj broj uključuje broj glavnog kanala i broj ploče za proširenje i kanala (ako se koristi). Kanali za proširenje su konfigurisani koristeći prozor Hardware Configuration opisan kasnije u ovom poglavlju.
On	Ova kolona omogućava vam da izaberete da li će podatak biti priman sa tog kanala. Kad su selektirani ćelija ili blok ćelija u ovom kanalu pojavit će se kutija za označavanje koja omogućuje da se kanal uključi sa Yes ili isključi sa No. Duplo klikanje ćelije u ovoj koloni će prebaciti status kanala. Edit meni omogućava da se svi kanali naprave aktivni ili neaktivni.
Type	Ova kolona omogućava da se postavi pojačanje ili tip ulaza svakog kanala. Pojačanja i tipovi variraju za razne opcione kartice. Blok ćelija u ovoj koloni može biti označena za više kanala sa istim tipom opcione kartice. Dvostruko klikanje na ćeliju će označiti sljedeće dostupno pojačanje ili tip.
Pole	Ova kolona pokazuje polarizaciju kanala (unipolarni ili bipolarni) za svaki kanal. Polarizacija može biti programirana ovdje, za svaki kanal posebno kad se koristi Daq Book/200,/216 i bilo koji Daq Board ili DBK15 univerzalna strujno/naponska kartica. Kad se koristi bilo koji drugi Daq ova kolona se postavlja u Hardwer Setup window. Za selektovane ćelije koje mogu biti promjenjene prozor za označavanje će prikazati Uni ili Bi . Dvostruko klikanje na ćeliju će prebaciti polaritet. Ako hardver ne može programirati polaritet meni za označavanje neće biti prikazan.
Label	Ova kolona sadrži opisno ime za ulazni kanal. Inicijalna labela je broj kanala, ali ona može biti zamjenjena u bilo kojih osam znakova. Ova labela se koristi prilikom označavanja kanala u analognom trigeru i listi dijagrama.
Units	Ova kolona vam omogućuje da mijenjate tehničke jedinice svakog kanala i primjenjujete linearne jednačine na Daq–ovim podacima. Kad je ćelija ili blok ćelija u ovoj koloni označen, analogni ulazni prozor prikazuje ulazne opcije u padajućem meniju. Označavanje $mX+b$ omogućava vam da definišete m i b i labelu tehničke jedinice. Onda će biti prikazane u Units (jedinice) koloni i $mX+b$ će biti primjenjeno na

	<p>očitavanja iz Daq-a prije nego što je očitavanje prikazano na ekranu ili zapisana na disku. X u ovoj jednačini je napon pročitano od daq-a (ili stepeni celzijusa za DBK14). Na primjer ako je Daq kanal konfigurisan kao bipolarni sa dobitkom 1 (x1), podrazumjevani napon bit će $\pm 5V$. Ovo korenspondira vrijednostima jedan (1) za m i nula (0) za b i jedinica u voltima. V može biti zamjenjeno u milivolte postavljajući m na 1000 i Units na milivolte (mV). Units kolona može biti korištena za softversku kalibraciju Daq-a. To se radi čitanjem poznatih ulaza na dvije različite tačke ulaznog naponskog opsega (uobičajeno na 0 i punoj skali) i rješavajući jednačinu $y=mX+b$. Napon pune skale koji se mijenja prema pojačanju kanala je 5V/pojačanje za bipolarne kanale i 10V/pojačanje za unipolarne kanale.</p>
Reading	<p>Ova kolona prikazuje Daq-ova analogna ulazna očitavanja. Ona ne može biti promjenjena od strane korisnika i uključuje se označavanjem "Enable Input Reading" pod <i>Acquisition</i> menijem ili selektiranjem <i>Start/Stop All indicators</i> pod <i>Window</i> menijem. Ova kolona će prikazati očitavanja onoliko brzo koliko to računar omogućuje. Tabela ne može biti promjenjena dok je uključena <i>input reading</i> kolona</p>
<p>Obavijest: Pored analognih ulaza, postoji 16-bitni digitalni ulazni kanal kome se može pristupiti sa daq-ovog konektora P3 ovaj visoko brzinski ulaz koji se čita istom brzinom kao bilo koji analogni ulaz. Može se uključiti ili isključiti klikanjem na <u>checkbox</u> ispod tabele analognih ulaza.</p>	

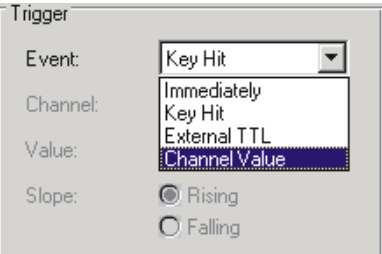
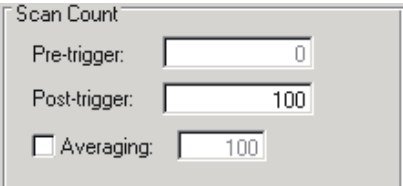
Parametri analogne ulazne tablice na Daq View – ovom glavnom prozoru

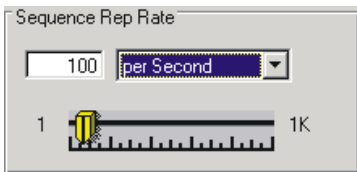
Konfiguracija prikupljanja podataka

Odmah do analogno ulazne tabele (spreadsheet) na desnoj strani glavnog prozora nalazi se sekcija za konfigurisanje prikupljanja. Tri glavna dijela sekcije uključuju setup parametre za triger, za skaniranje i uzorkovanje. Ove postavke bit će korištene kad se startuje prikupljanje na disk, selektirajući opciju «GO» (posljednja stavka u alatnoj traci-toolbaru). Kad je triger zadovoljen skanovi se prikupljaju na selektovanoj frekvenciji skaniranja i smještaju na disk u željenu datoteku.

Napomena: Visoko brzinski digitalni ulazni kanal je na donjem lijevom dijelu ekrana i može biti korišten sa daqovima koji ga podržavaju

Parametri	Opcije upotrebe ili opis
-----------	--------------------------

<p>Trigger Event (dogadjaj na koji triger okida)</p>	<p>Selektira izvor trigerera. Četiri moguća izvora trigerera su:</p> <p><i>Immediately(trenutno)</i>-izvrša trigerovanje trenutno</p> <p><i>Key Hit(pritisni tipku)</i>-vrši prikupljanje i čeka da korisnik pritisne tipku</p> <p><i>External TTL (vanjski TTL)</i>-Čeka padajući ili rastući vrh na pinu 25 konektora P1</p> <p><i>Channel Value(vrijednost kanala)</i>-Prati nivo selektovanog kanala;izvršava trigerovanje kad je parametar zadovoljen</p>	
<p>Trigger Slope (kosina okidača)</p>	<p>Specifikuje da li triger reaguje na rastuću ili padajuću kosinu na vanjskom TTL</p>	
<p>Analog Trigger Level Setup (postavka nivoa analognog trigerera)</p>	<p>Postavlja bazu trigerera na specificiranu vrijednost signala na odabranom kanalu</p>	
<p>Scan Count (brojanje skanova)</p>	<p>Broj skanova može biti u opsegu od 1 do 10.000.000. Skan uključuje sve kanale koji su označeni kao «ON» u analognoj ulaznoj konfiguracionoj listi. Pred – trigerski skan je broj skanova koji se dobije prije izvršavanja trigerovanja (dostupno samo za Channel Value triger).</p> <p>Napomena: Ako je pred trigerski skan veći od nule, pred trigerski i post trigerski uzorci nemogu premašiti 32508. Ako je pred trigerski skan jednak nuli, post trigerski skan ne može premašiti 10000000.</p>	


<p>Averaging (usrednjavanje)</p>	<p>Ovaj checkbox omogućava da usrednjavanje analogno ulaznog podatka bude uključeno ili isključeno. Usrednjavanje može da se koristi za povećavanje efektivne tačnosti signala uz koji je prisutan i šum. Usrednjavanje će povećati frekvenciju skeniranja i broj skanova ali opažena frekvencija i broj skanova (koji su postavljeni od DaqView-a) ne mijenjaju se.</p>	
<p>Sequence Rep Rate (brzina skeniranja)</p>	<p>Frekvencija skeniranja može biti postavljena u sekundama, milisekundama, minutima ili satima pomoću combo boxa. Pomjeranje klizača ili direktno unošenje u numeričko polje mijenja frekvenciju skeniranja. Maksimalna skenirajuća frekvencija zavisi od broja kanala koji su uključeni i od toga da li je uključeno Averaging (usrednjavanje). Uključivanje više kanala ili usrednjavanje će smanjiti maksimalnu frekvenciju skeniranja</p>	
<p>Napomena: Kad jednom počne prikupljanje podataka ovi parametri se ne mogu više mijenjati.</p>		

Opcije za konfiguraciju prikupljanja podataka u DaqView-ovom glavnom prozoru

Padajući meniji u DaqView-u

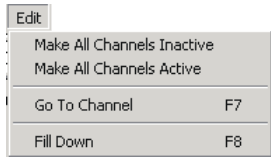


Neke (ali ne sve) opcije sa padajućih menija mogu biti pozvane i sa alatne trake. Svaka stavka na alatnoj traci ima sopstvenu ikonu. Postavljanje kurzora na ikonu i klikanje miša uključuje alatku ili otvara korenspodirajući prozor.

File

Stavke sa File menija i opis		
	<p>New</p>	<p>Postavlja sve parametre na njihove startne inicijalne postavke</p>

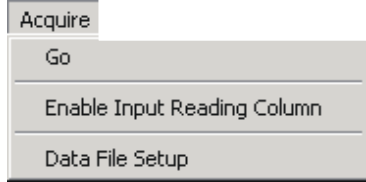


	Open	Postavlja parametre kao što je precizirano u specificiranoj setap datoteci.
	Save	Snima postojeću konfiguraciju da kasnije opet može da se pozove (piše preko postojeće verzije)
	Save As	Snima postojeću konfiguraciju da kasnije opet može da se pozove;pita da li da piše preko orginalne verzije ili da snimi u drugu datoteku.
	Convert Binary ASCII	Konvertuje prethodno dobijenu binarnu datoteku u ASCII koji može biti čitan raznim programima za analizu.
	Convert Binary to PostView Binary	Konvertuje prethodno dobijenu binarnu datoteku u binarnu datoteku koja može biti čitana programom PostView.
	Exit	Napušta DaqView program

Edit

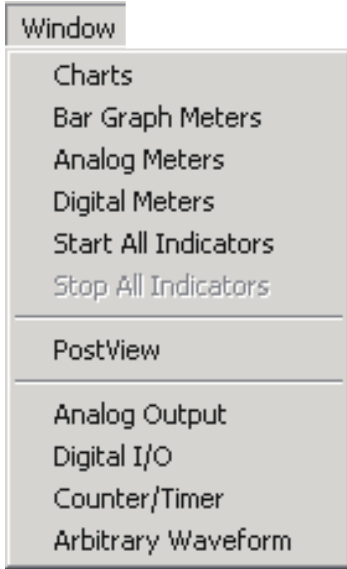











Edit meni stavke i opisi		
	 Make All Channels Inactive	Ova komanda stavlja NO u ON polje na svim kanalima. Za skeniranje samo nekoliko kanala lakše je isključiti sve kanale i uključiti samo one željene.
	 Make All Channels Active	Ova komanda stavlja YES u ON polje na svim kanalima
	Go To channel- F7	Ova komanda izbacuje zahtjev za unošenje broja kanala i prenosi se na taj kanal. Ovo je zgodno za hardverske konfiguracije sa stotinama kanala
	Fill Down-F8	Za mnogostruke ćelije odabrane u koloni ova komanda kopira vrijednost ćelije na vrhu u sve one koje se nalaze ispod

Acquire

Acquire meni stavke i opisi

	 Go	<p>Ova komanda priprema hardver za prihvatanje na disk kad je trigger zadovoljen, prikupljanje počinje. Sve interaktivne U/I kontrole su isključene kad je sistem pripremljen za prikupljanje. Parametri prikupljanja nemogu se mijenjati dok traje prikupljanje (snimanje).</p>
	 Enable Input Reading Column	<p>Ova komanda čita analogne ulaze i stavlja numeričke vrijednosti u tabelu u Reading koloni. Ako je Reading kolona uključena ova komanda je isključuje.</p>
	Data File Setup	<p>Ova sekcija određuje ime i tip datoteke koja će postojati poslije prikupljanja. Ime datoteke može biti direktno otkucano ili Browse Files (pretražuj datoteku) dugme može biti pritisnuto da se otvori dijalog boks za selekciju datoteka (selektovana datoteka će biti postavljena direktno u datoteka «name» polje). Za vrijeme prihvatanja bit će stvoren prosta binarna datoteka i <u>updated</u>, dok se podaci čitaju. Poslije prihvatanja ASCII tekst datoteka i Post View binarna datoteka mogu biti stvoreni ako su njihovi Check Box uključeni. Obje ove datoteke mogu biti čitane Post View-om. Ako sirovi binary check box (ček box za prostu binarnu datoteku) nije uključen, prosta binarna datoteka bit će izbrisana poslije stvaranja Post View ili ASCII datoteke. Box mnogostruka destinacija (Auto Re-Arm) usmjerava DaqView da stvori uzastopne datoteke za duge tokove podataka, kao što je potrebno. Ako je Check box <Validate File Overwrite> uključen to će prouzrokovati da DaqView potvrdi <u>overwriting</u> postojećih binarnih ASCII ili U/I datoteka.</p>

Window

Window meni stavke i opisi		
	 Charts	Prikazuje prozor sa dijagramima.
	 Bar Graph Meters	Pokazuje dolazeće podatke na pokazivaču sa trakama.
	 Analog Meters	Prikazuje dolazeće podatke na analognoj skali
	 Digital Meters	Pokazuje dolazeće podatke numerički
	 Start All Indikators	Počinje prikaz podataka u Reading koloni i bilo koji otvoreni dijagram ili pokazivač(analogni,digitalni)
	 Stop All Indikators	Zaustavlja prikaz podataka u Reading koloni i bilo koji otvoreni dijagram ili pokazivač.
	 Post View	Startuje PostView aplikaciju.
	 Analog Output	Prikazuje prozor analognog izlaza.
	 Digital I/O	Prikazuje digitalni I/O prozor.
	 Counter/Timer	Prikazuje prozor.
 Arbitrary Waveform	Prikazuje prozor proizvoljnog talasnog oblika.	

Device

Device meni stavke i opisi		
	Select DaqBook	Prikazuje dijalog box za unošenje printer porta (LPT1 doLPT4), nivo prekida(interrupt) (3-7) i port protokol (<u>osmobitski standard</u> , četverobitni standard i nekoliko EPP opcija). Napomena: Brzi EPP i SMC666 portni protokoli, interaptni nivoi uključuju 3-7, 10, 11, 12, 14, 15.

	DaqBoard	Prikazuje dijalog box za unošenje baznih adresa (0300 inicijalno) i nivoa interapta (10,11,12,14,15).
	Daq PCMCIA	Prikazuje dijalog box za unošenje baznih adresa (0330default) i nivoa interapta(5,6,7,10,11,12,14,15). Napomena: <i>DaqView pokušava da počne novu komunikaciju sa selektovanim uređajem. Ako je hardver nađen otvara se glavni prozor. Ako hardver nije nađen korisnik se upozorava da izvrši rekonfiguraciju i ponovo pokuša. Ako hardver nije nađen izađi iz DaqView-a i pokreni Daq test program</i>
	Simulated Device	Otvora DaqView <u>sesiju</u> ali ne pokušava da komunicira sa hardverom. Umjesto toga aplikacija simulira interakciju između softvera i hardvera. Ako je DaqView spojen sa pravim hardverom ova komanda će zatvoriti tu sesiju.
	Configure Hardware Settings	Otvora prozor za konfiguraciju hardvera koji omogućava korisniku da kaže softveru kako je hardver postavljen. Setup sekcije uključuju analogne ulazne kartice DBK, digitalne interfejsne module, ulazni polaritet, A/D signal Referenc i D/A vanjska Referenca.

Help

Help meni obezbjeđuje On-line pristup help datoteci. *About DaqView* daje broj verzije tekućeg softvera koji se koristi.