Cristian Foşalău

Introducere în instrumentaţia virtuală

Editura CERMI
Iaşi, 2010

CUPRINS

Prefață

Lucrarea de față reprezintă un manual pentru învățarea noțiunilor elementare legate de instrumentația virtuală, în particular pentru deprinderea cu utilizarea mediului de programare LabVIEW, produs al firmei National Instruments. Lucrarea se adresează în special studenților de la facultățile cu profil electric ce au incluse în programele de învățământ noțiuni de instrumentație virtuală, dar şi potențialilor utilizatori şi proiectanți de aplicații de sisteme de măsură automate şi distribuite.

Instrumentația virtuală reprezintă un concept introdus de aproximativ 20 de ani, fiind născut din dorința utilizării calculatorului pentru a construi un instrument de măsură. La ora actuală instrumentația virtuală câştigă tot mai mult teren în special pentru aplicații complexe, în care sunt necesare măsurări concomitente a mii de puncte, prelucrarea de cantități mari de informații de măsură şi accesul la rezultate de la distanță. Un instrument virtual este compus din două părți: un software dedicat care rulează pe o unitate de calcul, de preferat un calculator de proces, şi o secțiune hardware, cu rol de interfață între procesul de măsură şi unitatea de calcul.

Cartea de față realizează introducerea în partea de programare cu ajutorul limbajului LabVIEW, prezentând noțiunile de bază cu ajutorul cărora se poate dezvolta un program de instrumentație virtuală. Cartea nu tratează noțiuni de comunicare cu structurile hardware, acesta fiind subiectul unui al doilea volum.

In principiu, cu ajutorul LabVIEW se poate construi orice tip de aplicație, de la simple operații matematice, până la sisteme de comunicație sau baze de date complexe. LabVIEW posedă o serie de biblioteci care conțin funcții mai simple sau mai complexe, cu ajutorul cărora utilizatorul îşi poate construi propria aplicație, chiar cu un nivel relativ redus de cunoştințe de programare. Tendința producătorilor programului este de a veni în întâmpinarea unei game cât mai largi de utilizatori, pentru ca aceştia să-şi poată construi o aplicație care să le satisfacă cerințele în cât mai mare măsură.

Manualul este format din 10 capitole plus un capitol de probleme recapitulative, parcurgerea lui fiind suficientă pentru ca cititorul să acumuleze un nivel de cunoştințe de bază care să-i permită dezvoltarea unui program general în LabVIEW. Cartea conține o serie de exerciții rezolvate, în care utilizatorul este condus pas cu pas spre soluție. La finalul fiecărei secțiuni există şi câteva exerciții propuse, având scopul de a aprofunda noțiunile învățate şi de a dezvolta creativitatea cititorului.

Pentru completarea cunoştințelor, recomandăm şi consultarea exemplelor disponibile în LabVIEW precum şi a documentațiilor de pe pagina web a firmei National Instruments (www.ni.com) şi de pe pagina portalului *Connexions* http://cnx.org/content/#keyword/L/Labview.

Autorul

Prescurtări şi notații în text

PF – Panoul Frontal
DL – Diagrama de Legături
C – Control
I – Indicator
IV – Instrument Virtual
subIV – subrutină a unui Instrument Virtual
M – Mouse
MD – Click pe butonul din dreapta al mouse-ului
MS – Click pe butonul din stânga al mouse-ului
NP – Nod de proprietăți
RD – Registru de deplasare
VL – Variabilă locală
VG – Variabilă globală
LVM – Format de fişier text specific LabVIEW
TDM – Format de fişier binar specific LabVIEW
WG – Waveform Graph
WCh – Waveform Chart
XYG – XY Graph

 – Observație importantă

☞ - Indicație

Capitolul 1

INTRODUCERE ÎN INSTRUMENTAȚIA VIRTUALĂ

Generalități

In ultimii 20 de ani, instrumentația de măsură a evoluat atât în privința performanțelor, cât şi a flexibilității, în sensul înglobării a cât mai multe funcții de măsurare pe acelaşi dispozitiv. Dezvoltarea tehnicilor digitale şi în particular a computerelor, a permis creşterea numărului de puncte de măsurare simultană prin realizarea sistemelor de măsură complexe, supervizate de calculatoare. Dacă în generațiile mai vechi erau preponderente instrumentele de măsură analogice, controlate manual prin folosirea unor butoane de pe panoul frontal, în momentul de față, prin dezvoltarea conceptului de *instrument virtual*, instrumentele de măsură sunt programe de calculator care gestionează interfețele dintre procesul de măsurat şi computer în vederea achiziției de date Măsurătorile făcute cu aceste instrumente sunt efectuate automat, iar utilizatorul are posibilitatea de a adăuga funcțiuni noi prin program sau de a modifica modul de prezentare a rezultatelor.

Dezvoltarea instrumentelor digitale dotate cu interfețe de comunicare face posibil controlul instrumentelor de măsură reale nu numai manual, ci şi prin intermediul calculatorului. In generația actuală avem deci instrumentația de măsură programabilă prin computer. Utilizatorul îşi clădeşte un sistem cu mai multe instrumente, interfațate cu un computer PC, pe care le citeşte secvențial pentru a efectua măsurătorile cerute de aplicație.

La ora actuală se utilizează combinații de instrumente numerice programabile de sine stătătoare controlate de calculator, cu instrumente ce sunt incorporate în calculator prin utilizarea de cartele adiționale de achiziții de date şi software adecvat. Aceste sisteme oferă mai multă flexibilitate şi performanțe metrologice superioare datorită faptului că instrumentul este construit ca parte componentă a calculatorului, ceea ce face ca puterea de calcul şi de prezentare a datelor a computerului să poată fi folosite în operația de măsurare.

Cea mai mare parte a instrumentelor reale se limitează la îndeplinirea următoarelor funcții principale: culegerea datelor de măsură, analiza lor şi afişarea rezultatelor. Aceste funcții sunt implementate hardware în instrument, aşa încât ele,

odată stabilite, nu mai pot fi schimbate. Dacă se doresc funcţii suplimentare, este necesară modificarea structurii aparatului, cu consecinţe importante asupra costului. Aşa sunt construite multimetrele numerice, osciloscoapele numerice, generatoarele de semnal, aparatele de măsură a mărimilor neelectrice (termometre, barometre, anemometre, vitezometre, etc.), aparatele specializate pe transmisia semnalelor prin reţele, ş.a.m.d.

In principiu, un *instrument virtual* (IV) este un program de calculator în combinaţie cu un dispozitiv de achiziţii de date care simulează funcţiile unui instrument real, oferind performanţe comparabile cu acesta. De aici şi numele de *virtual.*

Structura unui instrument virtual

Structura unui IV este dată în figura 1.1. S1…Sn sunt senzori ce transformă mărimile de măsurat din proces în semnale electrice. Câteva exemple de senzori: termocuple, termorezistenţe, accelerometre piezoelectrice, mărci tensometrice, elemente cu ieşire în tensiune sau curent unificat, etc. Deoarece mărimile de ieşire din senzori nu sunt întotdeauna compatibile cu intrările interfeţei (de regulă tensiuni sau curenţi), este necesară intercalarea unui bloc de prelucrare primară a

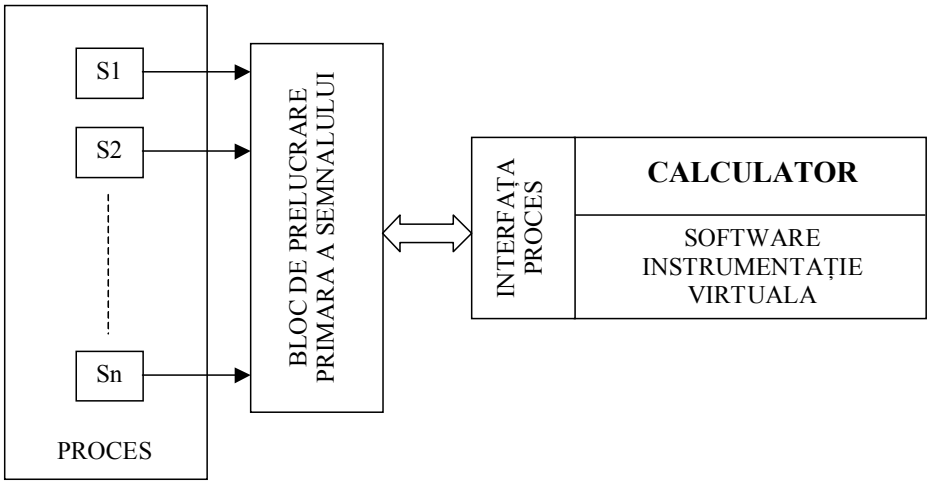


Figura 1.1

semnalului, ce conţine punţi de c.a sau c.c., punţi tensometrice, amplicatoare, filtre, elemente de izolare, multiplexoare, surse de semnal şi excitaţie a punţilor, circuite de compensare a joncţiunii reci, etc. Aceste elemente furnizează un semnal cu un raport semnal-zgomot scăzut, de tip tensiune sau curent, al cărui dependenţă de mărimea măsurată este foarte precis cunoscută.

Pentru a putea comunica cu procesul măsurat, calculatorul trebuie să fie dotat cu un dispozitiv hardware adiţional de interfaţare. Rolul acestui dispozitiv este de a transforma mărimea analogică de la intrare în cuvânt numeric pe baza conversiei analog-numerice. Pe lângă convertorul analog-numeric (CAN), o interfaţă mai poate conţine multiplexoare, amplificatoare şi circuite de eşantionare-memorare, aceste elemente nefiind însă obligatorii. Transmiterea cuvântului numeric către computer se poate face fie direct prin magistrala calculatorului (de cele mai multe ori prin protocolul DMA), fie prin interfeţele de comunicaţie serială şi paralelă. La ora actuală există o largă paletă de dispozitive capabile de a converti mărimile analogice de măsurat în semnale numerice. Câteva exemple ar fi: cartelele de achiziţii de date, digitizoarele, multimetrele incorporate, modulele individuale pentru măsurări distribuite, modulele de tip PXI, etc. Acestea sunt construite cu o autonomie mai mare sau mai mică faţă de calculator, însă toate au trăsătura comună de a converti semnalele analogice, din procesul real (în particular cele primite de la blocul de prelucrare primară a semnalelor), în semnale numerice, cu o acurateţe cât mai bună.

Odată depuse în memoria calculatorului, semnalele digitizate sunt gata de a fi prelucrate. Aici intră în funcţie programul de instrumentaţie virtuală, ce conduce şi supervizează toate funcţiile instrumentului: achiziţia, prelucrarea, stocarea, afişarea şi transmiterea la distanţă a informaţiei de măsură. Există pe piaţă mulţi producători de astfel de programe. La ora actuală se optează pentru programarea grafică, datorită accesibilităţii ridicate utilizatorilor mai puţin experimentaţi şi interfeţelor prietenoase de prezentare. Se încearcă similitudinea cât mai apropiată cu panourile frontale ale instrumentelor reale, beneficiind în plus de flexibilitate şi paleta largă de funcţii abordate.

Liderul de piaţă incontestabil în domeniul instrumentaţiei virtuale este firma National Instruments Ltd. din SUA (www.ni.com). Această companie pune la dispoziţia constructorilor de aplicaţii de sisteme de măsură o gamă uriaşă de dispozitive hardware dedicate măsurărilor distribuite, achiziţiei de date şi comunicaţiilor, precum şi de limbaje de programare specializate pe instrumentaţia virtuală. Cel mai cunoscut este limbajul LabVIEW, care este şi obiectul manualului de faţă.

Funcţiile instrumentelor virtuale

Spre deosebire de instrumentele reale, cele virtuale, fiind în principal bazate pe programe de calculator, prezintă un grad ridicat de flexibilitate, orice funcţie adiţională fiind foarte facil de implementat cu costuri minime, prin simple modificări de program. Funcţiile de bază ce pot fi dezvoltate în structura unui IV sunt:

- achiziţia automată a semnalelor obţinute de la senzorii de măsurare a mărimilor electrice şi neelectrice dintr-un proces, în conformitate cu

structura şi configuraţia aplicaţiei, stabilite de beneficiar;
- prelucrarea locală, parţială sau totală a informaţiei la punctul de măsură, cu ajutorul microsistemelor dedicate;
- stocarea datelor pe suport magnetic de tip disc dur, CD-ROM sau memorii portabile.
- transmiterea informaţiei de măsură prelucrate sau în stare brută la distanţă, la dispecer, prin reţea locală de calculatoare, Internet sau legătură fără fir (radio sau telefonie mobilă);
- afişarea şi prezentarea datelor pe monitor de calculator sub formă de interfeţe accesibile utilizatorului, în format grafic sau numeric;
- controlul automat al procesului ai cărui parametri sunt măsuraţi prin algoritmi numerici implementaţi pe calculator (pentru procese lente) sau procesoare numerice de semnal (pentru procese rapide)
- organizarea datelor, local sau la dispecer, în baze de date sau tabele de calculaţie, cu prelucrarea specifică a acestora (urmărirea unor indicatori de eficienţă, de calitate, de productivitate, etc.)

După cum se poate observa, funcţiile unui instrument virtual stau la baza unei palete mult mai largi de aplicaţii decât cele ale unui instrument real. Funcţiile unui instrument virtual nu trebuie să fie incluse toate în aceeaşi cutie, ca la instrumentul real. Calculatorul poate prelua o parte sau chiar toate aceste funcţii.

Avantajele şi dezavantajele instrumentaţiei virtuale

Instrumentaţia virtuală s-a dezvoltat în ultimele decenii ca urmare a progreselor tehnologiei digitale şi a dezvoltării calculatoarelor. In consecinţă, avantajele acestei tehnologii se răsfrâng şi asupra tehnicilor de măsurare. Printre aceste avantaje putem aminti:

- *expandabilitate* - posibilitatea achiziţiei unui număr mare de semnale de măsură prin utilizarea de multiplexoare pe intrările analogice;
- *precizie* - prelucrarea semnalelor pe cale numerică conferă precizii ridicate deoarece nu este afectată de toleranţele componentelor, temperatură, îmbătrânire, zgomote, etc. Singurele limitări de acest fel sunt date de partea analogică cuprinsă în senzori şi blocul de prelucrare primară. Se obţin astfel precizii net superioare aparatelor analogice şi comparabile cu cele ale instrumentelor reale numerice;
- *flexibilitate* - adăugarea unor funcţii noi, cu costuri minime, prin simple modificări de program;
- *stocarea informaţiei* măsurate în cantităţi foarte mari pe suporturile de memorie ale calculatorului, organizarea acesteia în baze de date şi prelucrare statistică;
- *transmiterea la distanţă* a datelor prin reţele de calculatoare, Internet, telefonie mobilă sau radio.

Înlocuirea instrumentelor de măsură reale din procesele industriale cu cele virtuale nu poate fi făcută însă în totalitate în momentul de față deoarece acestea din urmă prezintă totuşi unele neajunsuri care vor fi în viitor cu siguranță depăşite prin avansul tehnologiei. Aceste dezavantaje sunt date de:

- limitarea benzii de frecvență a semnalelor măsurate datorită limitărilor impuse de lanțul de măsurare şi în principal de CAN;
- costurile încă destul de ridicate.

In prezent există digitizoare ce lucrează cu frecvențe de până la 1000 megaeşantioane/secundă (Msamples/s), cu conversie pe 8 biți. Creşterea rezoluției convertoarelor şi deci a preciziei măsurării se poate face în detrimentul vitezei, deci cu limitarea benzii de frecvență. Cele mai bune performanțe la ora actuală (nivelul anului 2010) sunt atinse de convertoare pe 24 de biți, ce lucrează cu frecvențe de până la 100 MHz. Pentru prelucrarea informației de măsură în timp real sunt necesare însă calculatoare cu mare putere de calcul sau procesoare de semnal specializate de tip DSP. Aceste instrumente se limitează doar la afişarea semnalelor pe ecrane de tip osciloscop şi la realizarea unor calcule simple, de exemplu de aflare a valorilor maxime, medii, efective, a frecvenței şi perioadei.

Aplicații de instrumentație virtuală

Gama de aplicații ce utilizează instrumentația virtuală este extrem de vastă. Datorită avantajelor oferite de acest concept, aplicațiile pot fi extinse practic în orice domeniu al activității umane, evident acolo unde eficiența economică justifică costurile încă destul de ridicate. Printre aceste aplicații amintim:

- monitorizări complexe de procese industriale pentru mărimi lente, cu transmiterea la distanță a informației de măsură şi afişarea la dispecer sub formă numerică şi grafică;
- operații de frecvență ridicată, unde este necesară colectarea unui număr mare de date într-un timp scurt;
- operații repetitive ca testări şi calibrări automate şi experimente care rulează un număr mare de ori;
- operații la distanță şi în medii ostile, în locuri periculoase pentru prezența operatorului uman;
- conducere şi control numeric al proceselor prin algoritmi specifici;
- operații de precizie înaltă şi de lungă durată, dincolo de posibilitățile manuale, ca înregistrarea traiectoriei stelelor prin telescop.

Capitolul 2

PREZENTAREA MEDIULUI DE PROGRAMARE LABVIEW

LabVIEW este un mediu de programare grafic pentru instrumentaţie virtuală, produs şi dezvoltat de firma National Instruments din Austin, Texas, începând cu anul 1986. Acest mediu de programare s-a impus cu timpul datorită facilităţilor oferite pentru dezvoltarea de aplicaţii de măsurare, testare şi control cu ajutorul calculatorului, existând la ora actuală un număr foarte mare de integratori ce dezvoltă aplicaţii în acest limbaj pe întreg globul. LabVIEW este în principal dedicat construirii de instrumente virtuale, având funcţii specializate pe operaţiuni de achiziţie, prelucrare, afişare şi transmisie la distanţă a semnalelor de măsură, dar poate fi privit şi ca un limbaj de programare grafică de uz general, de sine stătător. LabVIEW este un mediu de programare foarte flexibil, dispunând de biblioteci puternice de funcţii dedicate. Astfel, se pot concepe şi dezvolta aplicaţii de la cele mai simple, ca de exemplu măsurarea şi afişarea unei temperaturi, până la programe sofisticate de testare on-line şi control industrial.

Programele dezvoltate în LabVIEW se numesc instrumente virtuale (*virtual instruments* sau *IV*-uri) şi prezintă extensia *.vi*. Aceste programe au rolul de a primi date de la utilizator sau de la interfeţele calculatorului cu procesul ai cărui parametri se măsoară, de a le prelucra şi apoi de a le afişa, stoca sau a le transmite la distanţă. Ideea constructorului limbajului este ca aceste IV-uri să semene cât mai bine din punctul de vedere al utilizatorului cu un instrument de măsură real, atât ca înfăţişare cât şi ca funcţii. LabVIEW, în cei peste 20 de ani de când există, a evoluat mult în complexitate. La ora actuală, limbajul posedă puternice capabilităţi de gestionare a resurselor calculatorului, în scopul optimizării puterii de calcul, ca şi un mare număr de toolkituri, seturi de biblioteci adiţionale dedicate, ce permit programatorului dezvoltarea de aplicaţii extrem de complexe.

Un IV conţine următoarele trei elemente principale:

1. Panoul frontal
2. Diagrama de legături
3. Pictograma cu conectorul

Sfaturi de început

Înainte de a începe lucrul, este bine să țineți cont de câteva sfaturi ce pot contribui la uşurarea activităților de exersare a programării în LabVIEW.

1. Cumpărați-vă un caiet studențesc în care veți nota la fiecare lecție operațiile pe care le executați, lucrurile noi pe care le învățați, greşelile pe care faceți, eventualele erori care apar la rularea instrumentelor, locurile unde salvați, posibile lucruri mai greu de înțeles şi care ar trebui reluate.
2. Să aveți la îndemână o memorie portabilă (stick de memorie) pe care să salvați tot ce lucrați la lecția respectivă.
3. Țineți cont că o problemă poate avea mai multe soluții. Găsiți o primă soluție, verificați că funcționează, găsiți-i punctele slabe şi încercați apoi să o optimizați sau să găsiți o altă soluție mai eficientă.
4. O aplicație LabVIEW este în fapt o aplicație de programare. In principiu, pentru a fi cât mai eficienți, ar fi bine să urmați următorii paşi:
   - Citiți cu atenție textul problemei.
   - Stabiliți clar funcțiile pe care trebuie să le îndeplinească instrumentul. Pe baza acestor funcții, împărțiți aplicația pe module (subIV-uri), care vă vor ajuta să organizați mai bine programul şi să folosiți aceeaşi funcție şi în alte aplicații.
   - Faceți analiza problemei şi desenați organigrama.
   - Faceți o analiză a controalelor şi indicatoarelor de care aveți nevoie şi concepeți panoul frontal. Grupați obiectele de pe panoul frontal după funcțiuni.
   - Pe parcursul dezvoltării programului, verificați din aproape în aproape funcționarea lui parțială.
   - Salvați variante intermediare ale programului. Dacă faceți o modificare în program, salvați varianta nouă sub alt nume.

Exercițiul 2.1

Scop

Familiarizarea cu elementele componente ale unui IV.

Mod de lucru

1. Lansați programul LabVIEW.
2. Deschideți fereastra de exemple din meniul *Help – Find Examples*.
3. Deschideți exemplul *Industry Applications – Analysis – Signal Generation and Processing.vi*.
4. Observați panoul frontal (PF) al instrumentului (figura2.1). Acest IV generează două semnale de formă şi frecvență la alegere, le aplică o fereastră, după care

semnalele sunt filtrate cu un filtre de asemenea la alegere. Pe PF se afişează desfăşurarea semnalelor în timp, cu şi fără fereastra aplicată, şi spectrul lor de putere (*Power Spectrum*). Pe indicatorul grafic al spectrului de putere există posibilitatea modificării frecvenței de tăiere a filtrului (*filter cutoff*).

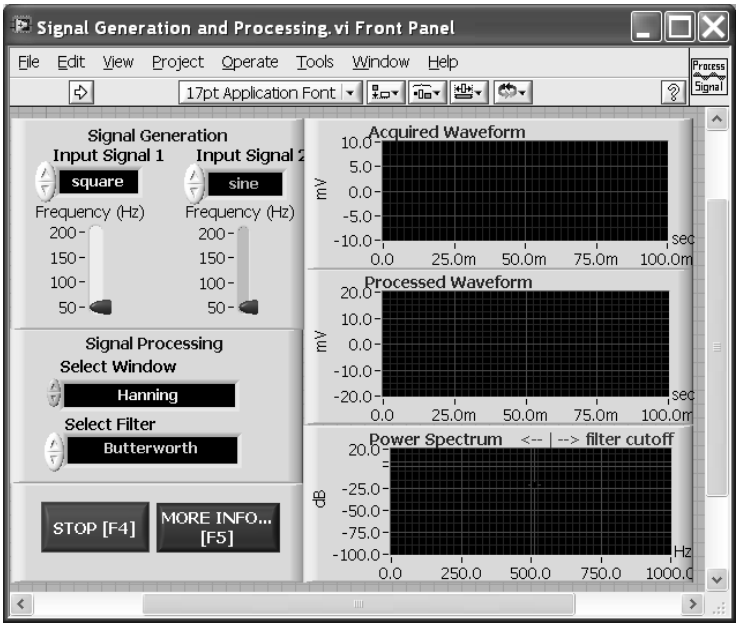


Figura 2.1

5. Rulați IV-ul de la butonul *Run* .
6. Modificați forma semnalelor, a ferestrei, tipul filtrului şi frecvența de tăiere, urmărind forma semnalelor înainte şi după procesare, precum şi spectrul lor de putere.
7. Deschideți diagrama de legături (DL) a IV-ului apăsând CTRL+E sau din submeniul *Window – Show Block Diagram* (figura 2.2). Acesta este codul sursă al programului în limbajul LabVIEW, ce materializează IV-ul.
8. Observați legăturile dintre nodurile grafice ale DL. De exemplu pictograma 

de pe DL este un nod sub forma unei funcții care calculează spectrul de putere al unui semnal. Această funcție este în principiu tot un IV, denumit în continuare *subIV*, care este apelat de către instrumentul principal sub forma unei subrutine.
9. Efectuați dublu click pe pictograma *Power*. In acest moment s-a deschis un nou IV, care este format de asemenea din PF şi DL. Dacă deschideți DL, veți observa că şi acest subIV conține o pictogramă.

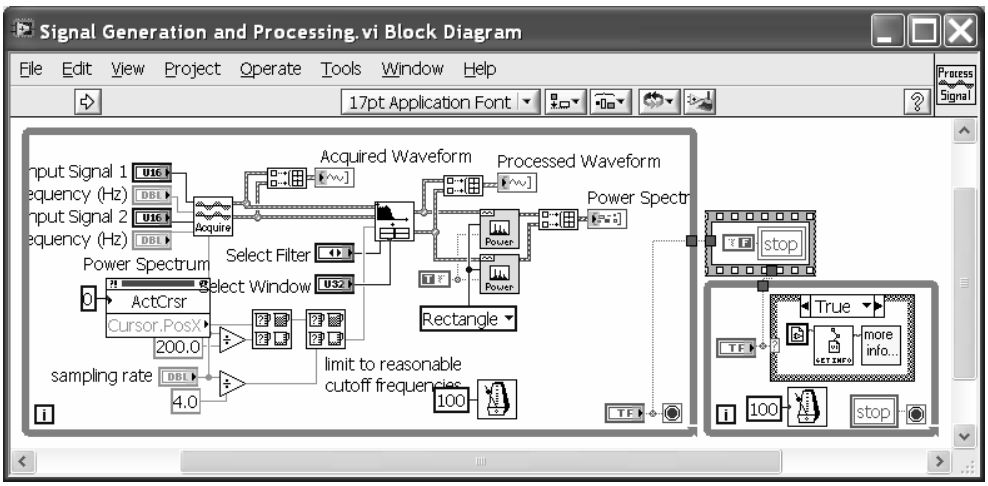


Figura 2.2

10. In continuare, dublu click pe această pictogramă deschide un nou subIV. Se constată astfel construcția modulară a IV-urilor în LabVIEW, un concept extrem de util atât din punct de vedere structural cât şi funcțional.
11. Reveniți pe PF a instrumentului principal, *Signal Generation and Processing.vi*.
12. Deschideți meniul *View – VI Hierarchy*. Observați în fereastra deschisă ierarhia întregului IV, care conține toate modulele din care acesta este format, împreună cu legăturile funcționale dintre ele.

Realizând un dublu click pe oricare din subIV-urile din ierarhie, se deschide PF al acestuia împreună cu diagrama sa de legături.

Există pe DL pictograme de funcții care nu materializează IV-uri. Aceste funcții sunt implementate sub formă de programe compilate direct în limbajul C, ale căror coduri nu pot fi vizualizate. Un exemplu este funcția ce calculează maximul şi minimul elementelor unui şir  , care se găseşte în subpaleta de funcții referitoare la matrici (*Array*). In principiu, majoritatea funcțiilor apelabile prin paletele de funcții sunt de acest gen.

13. Deschideți şi alte exemple din fereastra de exemple şi identificați elementele componente ale IV-urilor. Alte exemple interesante:
    Industry Applications – Analysis – Temperature System Demo.vi.
    Industry Applications – Process Control – Tank Simulation.vi.
    Industry Applications – Process Control – Control Mixer Process.vi.

Inițierea unei noi sesiuni de lucru

La lansarea programului LabVIEW, se deschide fereastra de start care ne permite să inițiem un nou IV sau un nou proiect (figura 2.3). Această fereastră conține două zone: zona fişierelor (*Files*) şi zona resurselor (*Resources*).

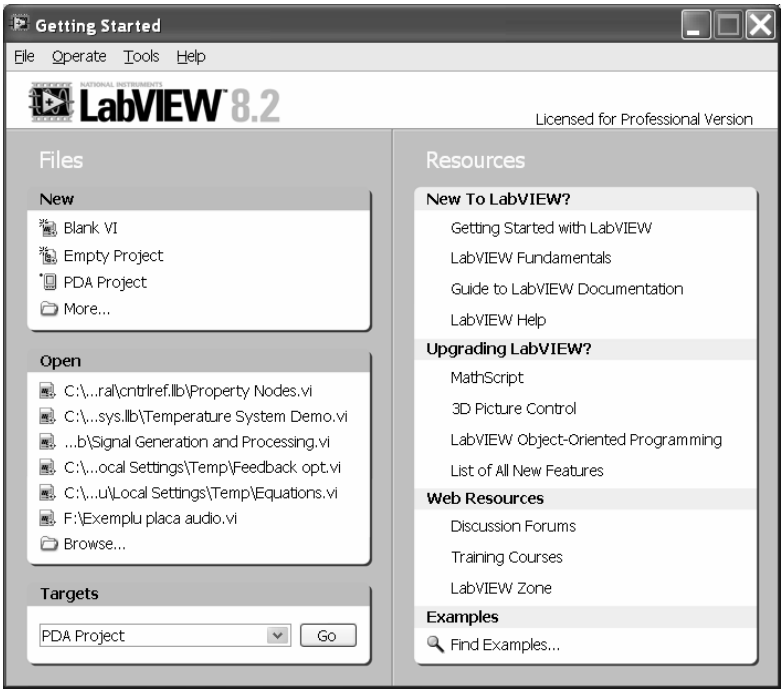


Figura 2.3

Zona fişierelor este împărțită, la rândul ei, în zona de inițiere a unor fişiere noi (*New*), zona de deschidere a unor fişiere deja create şi salvate anterior (*Open*) şi eventual zona fişierelor care vor fi încărcate în diferite dispozitive autonome pe care rulează LabVIEW, cum ar fi dispozitive Real-Time, PDA-uri sau plăci FPGA. Acestea se numesc în limba engleză *targets*, pentru utilizarea lor fiind necesară instalarea unor toolkit-uri corespunzătoare. Zona resurselor permite accesarea de documentații, cursuri şi informații legate de LabVIEW în principal prin Internet, găsirea de exemple, inițierea de colaborări cu utilizatori din întreaga lume prin intermediul forumurilor sau realizarea unor actualizări ale programului.

Această fereastră poate fi evitată la lansare dacă se bifează opțiunea *SkipGetting Started Window on launch* din meniul *Tools – Options – Environment*.

Diferența dintre un IV (*VI*) şi un proiect LabVIEW (*project*) este aceea că primul este reprezentat de un singur fişier cu extensia *.vi* care îndeplineşte toate funcțiile unui instrument virtual, pe când un proiect conține mai multe fişiere, unele

generate de LabVIEW, altele adăugate de către utilizator conținând resurse , fişiere de date, inițializări şi setări care folosesc la atingerea unui scop. In general proiectele se creează când aplicația este mai complexă, necesitând mai multe IV-uri pentru implementare sau când se doreşte ca aceasta să ruleze pe dispozitive speciale, altele decât computerul personal, cum ar fi dispozitivele de tip FPGA (Field Programmable Gate Array), PDA (Personal Digital Assistant) sau RT (Real-Time).

In cadrul acestui manual ne vom ocupa doar de dezvoltarea de aplicații de tip instrument virtual singular (*.vi*) ce vor rula doar pe computerul personal. Pentru informații referitoare la dezvoltarea de proiecte sau implementarea pe dispozitive speciale, consultați manualul de utilizare al toolkit-ului respectiv şi pagina web **www.ni.com**.

Panoul Frontal

PF reprezintă interfața utilizatorului cu instrumentul virtual. La deschiderea unui IV nou, panoul frontal este vid. Pe acesta se pot adăuga obiecte grafice care se numesc controale (C) şi indicatoare (I). C şi I sunt terminale interactive de intrare-ieşire a datelor. Orice control poate fi transformat în indicator şi invers, prin accesarea meniului pop-up al acestuia. Prin intermediul controalelor utilizatorul furnizează date instrumentului, în timp ce indicatoarele afişează informațiile procesate de instrument. Un exemplu de PF este dat în figura 2.4.

Paleta de controale şi indicatoare

Există o largă varietate de tipuri şi forme de controale şi indicatoare. Acestea se găsesc pe paleta de controale (*Controls Palette*). Accesarea acestei palete se face în următoarele moduri:

a) *View –Controls Palette*
b) *MD pe PF*

In cel de-al doilea mod, paleta poate fi fixată ca fereastră pe ecran prin MS pe pioneza din colțul din stânga sus a acestei ferestre. Altfel, paleta se închide după realizarea unui MS oriunde pe PF.

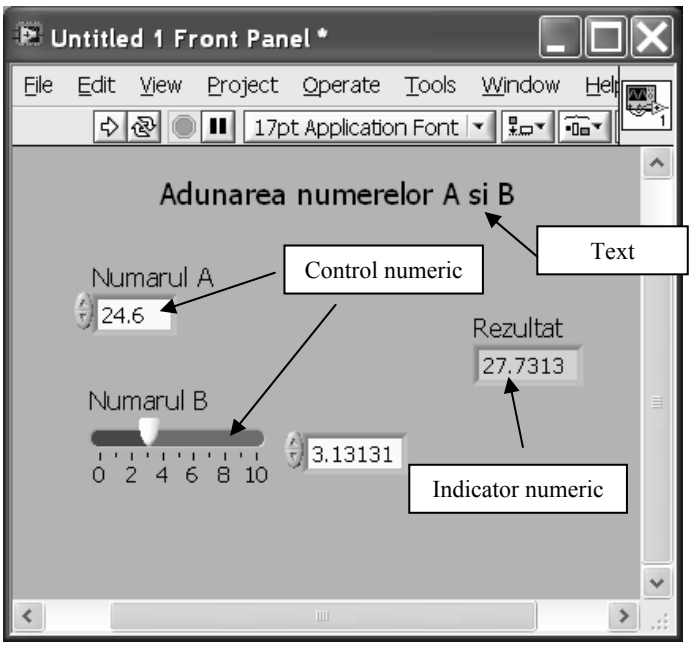


Figura 2.4

Exercițiul 2.2

Scop

Cunoaşterea paletei de controale de pe panoul frontal.

Mod de lucru

1. Deschideți un nou IV.
2. Deschideți paleta de controale.
3. Treceți în revistă tipurile principale de C şi I oferite de paletă din meniul *Modern* (care se deschide în mod implicit). Observați că acestea sunt grupate după tipurile de date pe care le vehiculează:
   - Numeric (Numerice)
   - Boolean (Booleene)
   - String & Path (Siruri de caractere şi căi)
   - Array & Cluster (Matrici şi clustere)
   - List & Table (Liste şi tabele)
   - Graph (Indicatoare grafice)
   - Ring & Enum (Control în inel şi enumerare)
   - Containers (Containere)
   - I/O (Controale de intrare-ieşire)

- Refnum (Număr de referință, un identificator unic atunci când se lucrează cu fişiere, directoare, dispozitive sau conexiuni la rețea)
- Variant & Class (Variante şi clase)
- Decorations (Decorări)

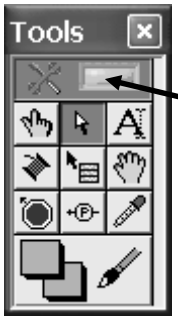
In afara controalelor din meniul *Modern*, există şi alte categorii de controale care, în principiu, au aceleaşi funcțiuni ca cele din meniul *Modern*, diferind de acestea doar prin aspect. Astfel sunt controalele din meniul *System* (care sunt preluate din paleta de obiecte a sistemului de operare), cele din meniul *Classic* (care sunt controale din versiunile anterioare lui LabVIEW 5.1), controale de tip *Express* sau controale definite de utilizator. In continuare vom face referire doar la controalele din meniul *Modern*.

Uneltele utilizate în LabVIEW

Unealta în LabVIEW este un mod de operare special al mouse-ului. Acestea se recunosc după forma prompterului. Uneltele de lucru se găsesc pe paleta de unelte, care este accesibilă atât pe PF cât şi pe DL. Deschiderea paletei de unelte se face cu comanda *View – Tools Palette*. Uneltele se schimbă automat în funcție de poziția prompterului pe un obiect. Uneltele se pot schimba şi manual, realizând un MS pe semnul grafic corespunzător uneltei dorite de pe paleta de unelte. Schimbarea manuală a uneltelor se face în următoarele moduri:

- MS pe semnul grafic de pe paletă corespunzător uneltei dorite
- cu tasta *Tab* (balans între uneltele principale 1), 2), 3) şi 10))
- cu tasta *Space*, balans între uneltele 1) şi 2) pe PF sau 2) şi 4) pe DL

Odată ce s-a optat pentru schimbarea manuală a uneltelor, opțiunea se menține pentru toată sesiunea de lucru. Dacă se doreşte reactivarea schimbării automate, se realizează MS pe *Automatic Tool Selection* de pe paletă, ca în figura 2.5.



Automatic Tool Selection
Balans între selectarea manuală şi selectarea automată a uneltelor

Figura 2.5

In ordinea de la stânga spre dreapta şi de sus în jos, uneltele au următoarele semnificații:

Tabelul 2.1

Nr.	Simbol	Denumire	Acțiune
1)		***Operare*** *(Operate value)*	schimbarea valorii unui C sau I sau selectarea textului dintr-un C
2)		***Selectare*** *(Position/size/select)*	selectare, poziționare, redimensionare
3)		***Editare*** *(Edit text)*	editare text şi creare etichete libere
4)		***Realizare legături*** *(Connect wire)*	realizarea legăturilor dintre obiectele de pe diagramă. Nu este activ pe PF decât atunci când se realizează legăturile dintre conectorul unui subIV şi obiectele de pe PF
5)		***Meniu shortcut*** *(Object Shortcut Meniu)*	deschiderea meniului pop-up al unui obiect (echivalent cu MD pe obiect)
6)		***Defilare*** *(Scroll Window)*	defilarea liberă a ferestrei principale (fără utilizarea scroll-bar-ului)
7)		***Setare întreruperi*** *(Set/Clear Breakpoint)*	selectarea unei întreruperi în IV pe funcții şi structuri (nu e activ pe PF)
8)		***Tester*** *(Probe Data)*	crearea unei probe de test pe un fir (nu e activ pe PF)
9)		***Preluare culoare*** *(Get Color)*	copierea unei culori
10)		***Selectare culoare*** *(Set color)*	paleta de culori

Introducerea datelor într-un control

Utilizatorul are posibilitatea să introducă date într-un control în mai multe moduri, în funcție de forma şi tipul controlului. Pentru controalele de tip numeric de pe PF, există următoarele posibilități:

a) prin manipularea cu unealta de operare a micilor săgeți din stânga valorii numerice a C. La apăsarea pe săgeata de sus sau pe cea de jos, valoarea lui C incrementează, respectiv decrementează valoarea numerică cu câte o unitate (figura 2.6);
b) prin scrierea cu uneltele 1) sau 2) a valorii dorite direct de la tastatură.

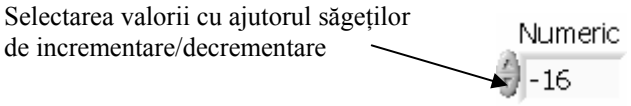


Figura 2.6

In cazul controalelor numerice de tip *Slide* sau *Knob*, se acționează cu unealta 1) asupra cursorului ca şi cum s-ar produce culisarea sau rotirea unui cursor de pe un buton real.

La controalele de acest tip, care nu au vizibile implicit valorile numerice înscrise, pentru introducerea datelor de la tastatură se procedează mai întâi la afişarea controlului de valori numerice (MD pe control şi se selectează din meniul pop-up *Visible Items – Digital Display*), după care se introduce valoarea în acest control cu uneltele 1) sau 2).

Afişarea temporară a valorii înscrise într-un control *Slide* se face cu MS pe cursorul controlului (figura 2.7)

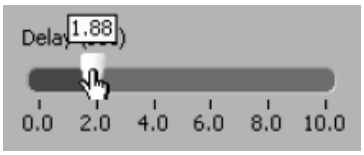


Figura 2.7

Redimensionarea unui control sau indicator

Orice obiect de pe PF poate fi redimensionat utilizând unealta 2). Modul de redimensionare se face în funcție de tipul controlului. De exemplu, pentru un C de tip numeric simplu, redimensionarea se face doar după o dimensiune (lungimea câmpului de date). Pentru un control de tip *Slide* sau *Knob*, ca şi pentru controalele booleene sau şir de caractere, redimensionarea se poate după două dimensiuni astfel:

- se apropie M de marginea obiectului fără a face clic, până când obiectul de redimensionat este înconjurat de o serie de puncte denumite noduri de redimensionare, corespunzătoare direcțiilor după care se face redimensionarea. Plasarea prompterului mouseului în aceste noduri face ca acesta să ia forma unei săgeți duble (figura 2.8).
- se ține apoi apăsat MS şi se redimensionează obiectul după dorință (drag & drop)

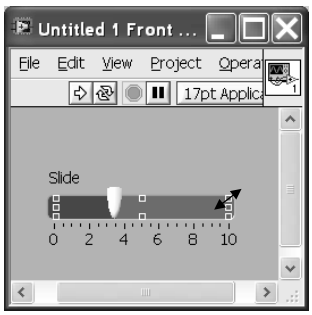


Figura 2.8

Mutarea, ştergerea şi copierea controalelor şi indicatoarelor

Dacă se doreşte mutarea unui obiect pe PF dintr-un loc în altul sau ştergerea sau copierea lui, obiectul trebuie mai întâi să fie selectat utilizând unealta 2). Selectarea se face realizând un MS pe obiect, caz în care obiectul apare încadrat într-un chenar punctat, ca în figura 2.9.

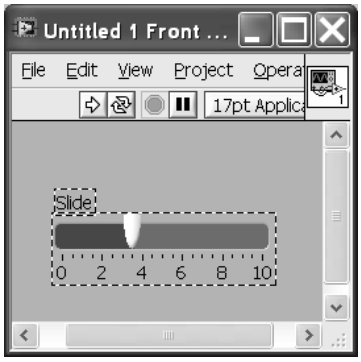


Figura 2.9

Mutarea se realizează prin simpla tragere cu M a obiectului pe suprafaţa PF.

Ştergerea se realizează prin apăsarea tastei *Delete*.

Copierea se face prin apăsarea tastei *Ctrl-C*. In acelaşi mod se face decuparea şi lipirea obiectelor cu tastele *Ctrl-X,* respectiv *Ctrl-V*.

Stergerea, copierea, decuparea şi lipirea unui obiect de pe PF se poate face şi de pe diagrama de legături (DL), acţionând asupra terminalului corespunzător.

Mutarea unui terminal pe DL însă nu implică şi mutarea obiectului corespunzător pe PF. De asemenea, dacă se realizează decuparea unui obiect pe DL, lipirea lui trebuie să se facă tot pe DL. Nu se poate face deci decuparea pe DL şi lipirea pe PF. Dacă se realizează copierea/decuparea unui obiect pe PF şi se încearcă lipirea

lui pe DL, acesta va apărea ca o constantă conținând valoarea numerică a obiectului respectiv din momentul copierii/decupării.

Cosmetizarea unui control sau indicator

Un C sau I poate fi cosmetizat prin schimbarea culorilor formelor principale şi a textului. Schimbarea culorilor formelor se face în modul următor:

- se selectează unealta 10) şi se poziționează pe forma care se doreşte a se colora
- cu MD se alege culoarea preferată din paleta de culori
- cu MS se fixează culoarea

Schimbarea culorii textului:

- se selectează unealta 3)
- se selectează tot textul de editat (valoare, etichete, caption, text independent)
  - se foloseşte butonul shortcut de setare a textului din bara de butoane de sub bara de meniuri.

Meniul *shortcut* al unui control sau indicator

Acest meniu se accesează fie cu unealta 5), fie, mai simplu, cu MD pe obiect. Fiecare obiect are meniul lui specific. O parte din opțiunile meniului sunt însă comune tuturor controalelor şi indicatoarelor. Acestea sunt (figura 2.10):

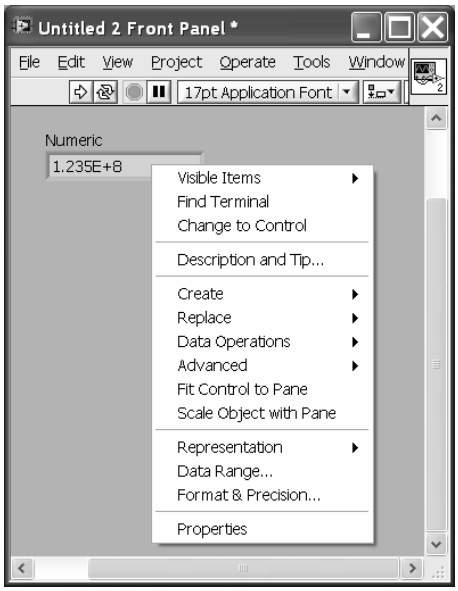


Figura 2.10

Visible Items

- *Label* – etichetă – un cuvânt de identificare a obiectului în diagrama de legături. Editarea etichetei se face cu unealta 3)
- *Caption* – un cuvânt care poate fi diferit de etichetă şi care însoţeşte obiectul pe PF. Poate fi o descriere mai lungă a obiectului şi nu apare pe DL. Ideea utilizării lui *caption* este de a nu încărca inutil DL, atunci când denumirea obiectului este prea lungă. Pentru identificarea terminalului pe DL se foloseşte eticheta.
- *Unit Label* – eticheta unității de măsură. Apare doar dacă se lucrează cu unități de măsură.
- *Radix* – formatul de scriere a numărului (zecimal, binar, octal). Este specific doar C şi I numerice.
- *Increment/Decrement* – (numai pentru C) afişarea sau nu pe PF a săgeților de incrementare/decrementare a conținutului controlului, aflate în stânga valorii numerice.

Find Terminal – Apăsând MS pe această opțiune, se deschide DL iar terminalul corespunzător clipeşte pentru identificarea lui.

Change to Indicator (Control) – transformă C în I şi invers.

Description and Tip – permite editarea descrierii obiectului în secțiunea de documentare a IV-ului.

Create

- *Local Variable* – creează automat o variabilă locală cu numele etichetei obiectului
- *Property Node* - creează automat un nod de proprietăți, pentru modificarea programatică a proprietăților obiectului
- *Reference* – creează automat o referință pentru obiectul pe care îl reprezintă
- *Invoke Node* - creează automat un nod de invocare atunci când se realizează acțiuni sau metode asupra unei aplicații sau a unui IV.

Replace – deschide paleta de controale pentru a permite înlocuirea obiectului cu un altul.

Data Operations

- *Reinitialize to Default Values* – reinițializează controlul cu valoarea implicită. Dacă nu se specifică altfel, valoarea implicită este 0 la controalele numerice, FALSE la cele booleene, şirul vid la şiruri de caractere, etc.
- *Make Current Value Default* – stabileşte ca implicită valoarea curentă
- *DataSocket Connection* – stabileşte o legătură de tip DataSocket cu obiectul
- *Cut, Copy, Paste Data* – decupează, copie sau lipeşte datele din şi în alt obiect

Advanced

- *Key Navigation* – atribuie o cheie pentru accesul de la tastatură a C în timpul

rulării. Dacă C este de tip numeric, prin tastarea cheii atribuite se permite schimbarea de la tastatură a valorii C. Dacă este de tip boolean, se basculează între TRUE şi FALSE.

- *Synchronous Display* – afişează valoarea obiectului sincron cu rularea instrumentului, la fiecare actualizare a lui. Se utilizează mai ales la indicatoare grafice, când se creează animaţie.
- *Customize* – deschide meniul de personalizare a obiectului. Dacă se doreşte ca un C sau I să arate altfel decât este cel din paletă (de ex. să fie mai mare, să scrie cu alte caractere, alte culori, etc.), înseamnă că se creează un C sau I personalizat. După editarea după dorinţă, acesta se salvează într-un fişier cu extensia *.ctl* care se depune în directorul *user.llb*. Obiectul poate fi adus în orice moment pe PF din paleta de controale ca orice alt obiect predefinit, utilizând submeniul *User Controls*.
- *Hide Control (Indicator)* – ascunde obiectul de pe PF, dar terminalul rămâne pe DL. Pentru ca acesta să fie din nou vizibil pe PF, se accesează în meniul pop-up de pe DL, *Show Control (Indicator).*
  - *Enabled State* – validează starea de activare a obiectului. Se referă numai la C.
  - *Enabled* – activ. Permite introducerea datelor de către operator.
  - *Disabled* – inactiv. Nu permite introducerea datelor.
  - *Disabled & Garyed* – inactiv şi reprezentat în tonuri de gri.

Fit Control to Pane – redimensionează obiectul după toate dimensiunile permise astfel încât acesta să se încadreze complet în panoul frontal.

Scale Object with Pane – redimensionează obiectul proporţional cu dimensiunile panoului frontal. Modificarea dimensiunilor PF duce la modificarea proporţională a obiectului.

In afara opţiunilor de mai sus, care sunt comune tuturor controalelor, există şi o serie de opţiuni specifice, dependente de tipul controlului. Mai jos sunt date opţiunile pentru controalele de tip numeric.

Representation – se deschide o paletă în care se specifică tipul de date numerice pe care le vehiculează obiectul (v. tipuri de date).

Data Range – deschide fereastra de proprietăţi a obiectului pe meniul *Data Range*, în care se specifică (figura 2.11):

1. Valoarea implicită a controlului. Cu această valoare se iniţializează controlul la încărcarea IV-ului. Valoarea implicită se poate schimba cu opţiunea *Make Current Value Default* din meniul shortcut al acestuia.
2. Tipul de date numerice vehiculat de control.
3. Valoarea minimă pe care o poate lua obiectul. Aceasta poate fi implicit valoarea dată de reprezentarea numărului pe octeţi în concordanţă cu tipul de date vehiculat, sau o valoare stabilită de utilizator. Dacă se introduce o valoare

mai mică decât valoarea minimă, există două posibilități (specificate în câmpurile din dreapta)

- valoarea introdusă să fie limitată la valoarea minimă (*Coerce*)
- valoarea introdusă să fie ignorată (*Ignore*)

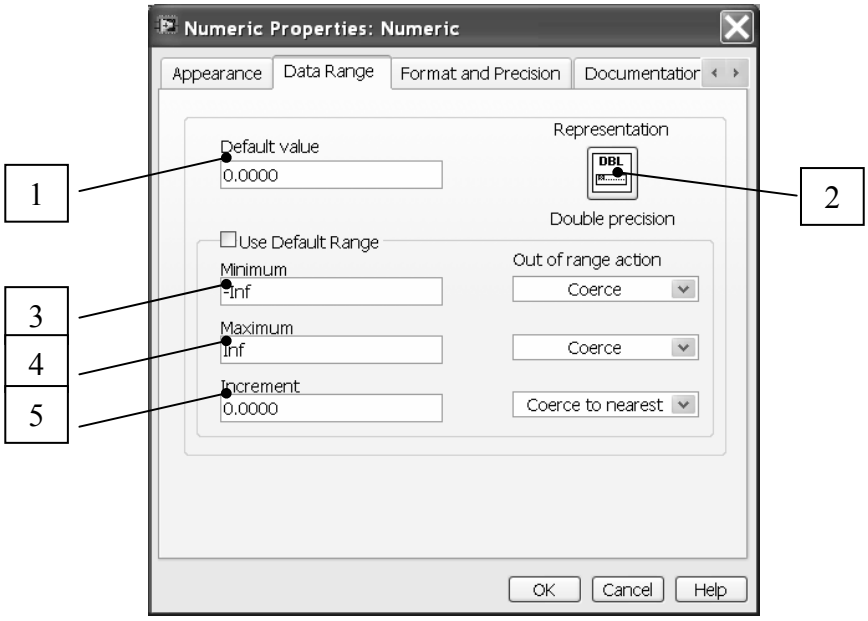


Figura 2.11

4. Valoarea maximă pe care o poate lua obiectul, cu aceleaşi observații ca la specificarea valorii minime.
5. Valoarea cu care C îşi incrementează sau îşi decrementează conținutul la apăsarea săgeților din stânga valorii numerice cu unealta de operare.

Format & Precision - stabileşte formatul şi precizia de afişare a numerelor pe displayul digital al obiectului. Formatul poate fi:

Floating point (virgulă mobilă), la care reprezentarea se face sub forma unui întreg urmat de un număr de zecimale, atâtea câte sunt specificate în câmpul *Digits*. Tipul preciziei (*Precision Type*) specificat în câmpul *Digits* poate fi:

- sub forma de număr de cifre după virgula zecimală (*Digits of Precision*)
- sub formă de număr de cifre semnificative (*Significant Digits*). Dacă numărul are mai multe cifre decât numărul de cifre semnificative specificat, valoarea lui este rotunjită sau trunchiată la numărul de cifre semnificative.

Scientific, la care reprezentarea se face sub forma unui număr în virgulă mobilă înmulțit cu 10 la o putere. Exemple: 1.24E+2, 2.579E-3.

SI notation, la care numărul este reprezentat sub formă de multipli sau submultipli ai unei unități de măsură în sistemul internațional. Exemple: 1.24m, 3.519M, 2.4n.

Automatic formatting, la care formatarea numărului se face automat fie în *floating point,*, fie în *scientific*, în funcție de mărimea lui.

Dacă numărul este întreg, pe lângă posibilitățile de mai sus, acesta mai poate fi reprezentat în *hexazecimal, octal* sau *binar*.

Conținutul unui control (indicator numeric) poate fi interpretat şi ca informație de timp, caz în care numărul este reprezentat în două moduri:

Absolute time, în care numărul, convertit la întreg, reprezintă numărul de secunde scurs de la data timpului universal, 1 ianuarie 1904 (01.01.1904), ora 2.00. In acest caz controlul (indicatorul) afişează numărul convertit în oră şi dată. Există posibilitatea de stabilire a formatului de afişare a orei şi a datei.

Relative time, în care numărul este convertit în ore, minute şi secunde scurse de la timpul 0.

a) Toate opțiunile de mai sus se referă doar la modul în care numărul este afişat pe displayul digital al controlului, şi nu la reprezentarea lui în calculator, care este dată de tipul de dată specificat în *Representation*.

b) C sau I din paleta de controale numerice de tip *slide, bar, knob, dial, meter, gauge, tank, thermometer* au posibilitatea de a afişa şi numeric valoarea introdusă, apelând în meniul shortcut *Visible Items – Digital Display*. Displayul digital prezintă un meniu shortcut diferit de cel al obiectului pe care îl reprezintă.

c) Controalele de la punctul b) mai prezintă în meniul shortcut o opțiune intitulată *Text Labels*. Apelând această opțiune, controlul devine automat de tip U32 (v. tipuri de date), fiecărei valori întregi atribuindu-i-se un text. Textele pot fi editate cu unealta 3).

d) In afara opțiunilor prezentate, fiecare C sau I are şi alte opțiuni specifice.

Properties este o opțiune comună tuturor C şi I, care reuneşte într-o singură fereastră cu mai multe Tab-uri, toate opțiunile din meniul shortcut al obiectului.

Butoane pentru comenzi rapide

Atât pe panoul frontal cât şi pe diagrama de legături, în partea de sus a ferestrei, sunt disponibile o serie de butoane prin care se furnizează comenzi programului sau se execută diverse operațiuni de editare texte sau grupare de obiecte. In figura 2.12 este dată bara butoane pentru panoul frontal. Acestea sunt:

1. ***Run*** (rularea instrumentului)

2. ***Run continuously*** (rulare continuă). După terminarea primei rulări a programului, aceasta se reia în mod automat până cât se apasă butonul *Abort execution*.
3. ***Abort execution*** (STOP de urgență). Opreşte imediat rularea programului, în orice fază de execuție s-ar afla.

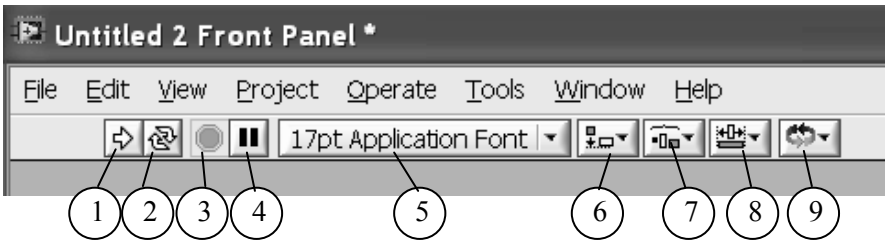


Figura 2.12

4. ***Pause*** (Pauză). Opreşte temporar instrumentul într-un nod. Se foloseşte pentru depanare, când se urmăreşte evoluția unor variabile. La apăsarea butonului, se trece automat pe diagrama de legături şi se indică prin încadrarea într-un chenar clipitor a nodului ce urmează a fi executat.
5. ***Text settings*** (Fixarea parametrilor de text). Se stabilesc tipurile caracterelor textelor, mărimea, stilul, alinierea, culoarea lor. Se selectează în prealabil textul după care se fac modificările dorite.
6. ***Align objects*** (Aliniază obiectele). Se realizează selectarea obiectelor ce se doresc a fi aliniate, după care se optează pentru aliniere relativ la marginile laterale sau centru.
7. ***Distribute objects*** (Distribuție obiecte). Se realizează amplasarea obiectelor într-o arie selectată în funcție de distanța dintre ele.
8. ***Resize objects*** (Redimensionare obiecte). Se realizează redimensionarea automată a obiectelor în funcție de distanța dintre ele.
9. ***Reorder*** (Rearanjare). Se pot realiza grupări/degrupări de obiecte, ancorarea în fereastră, aducerea în plan apropiat sau în plan îndepărtat a obiectelor suprapuse.

Acțiunea mecanică a controalelor booleene

Controalele de tip boolean arată şi se comportă pe panoul frontal ca nişte comutatoare electromecanice. Pentru a simula cât mai bine funcționalitatea acestora, ele pot fi configurate în funcție de modul cum îşi schimbă starea la apăsare. Există 6 tipuri de acțiuni mecanice, selectabile din meniul shortcut al controlului, opțiunea *Mechanical Action*.

Tabelul 2.2

	**Comutare la apăsare** (Switch When Pressed)	Schimbă valoarea controlului la apăsare cu unealta de operare. Acțiunea e similară cu cea a unui comutator de lumină. Citirea valorii comutatorului de către instrument nu îi afectează starea.
	**Comutare la eliberare** (Switch When Released)	Schimbă valoarea controlului atunci când se ia degetul de pe butonul mouse-ului (când se eliberează butonul). Citirea valorii comutatorului de către instrument nu îi afectează starea.
	**Comutare până la eliberare** (Switch Until Released)	Schimbă valoarea controlului la apăsare şi o reține atâta timp cât controlul este apăsat. La eliberare se revine la vechea valoare. Acțiunea e similară cu apăsarea pe un buton de sonerie. Citirea valorii comutatorului de către instrument nu îi afectează starea.
	**Zăvorâre la apăsare** (Latch When Pressed)	Schimbă valoarea controlului la apăsare cu unealta de operare şi reține noua valoare până la prima citire a controlului de către instrument, când se revine la valoarea inițială. Acțiunea este aceeaşi chiar dacă se ține sau nu apăsat butonul mouse-ului. Acțiunea e similară cu cea a unei siguranțe automate. Se utilizează în bucle WHILE pentru butonul de stop.
	**Zăvorâre la eliberare** (Latch When Released)	Schimbă valoarea controlului la eliberarea butonului mouse-ului. Valoarea este reținută până la prima citire a controlului de către instrument, când se revine la valoarea inițială.
	**Zăvorâre până la eliberare** (Latch Until Released)	Schimbă valoarea controlului la apăsare şi o reține până la prima citire sau până când se eliberează butonul mouse-ului, care operație se execută ultima.

Tipuri de date reprezentate în LabVIEW

Informația vehiculată în IV-urile construite în LabVIEW se prezintă sub forma unei largi varietăți de tipuri de date. Cele mai importante sunt datele numerice, dar şi alte tipuri cum ar fi booleenele, şirurile de caractere sau clusterele sunt de asemenea foarte utilizate. Mai jos sunt date tipurile de date numerice precum şi celelalte tipuri de date care sunt în mod obişnuit utilizate în construcția IV-urilor.

Tipuri de date numerice

Tipul de dată numerică se stabileşte din opțiunea *Representation* a meniului shortcut. La aducerea pe PF a unui C sau I, reprezentarea lui implicită este de tip

Real – dublă precizie. Paleta de tipuri din opțiunea *Representation* este dată în figura 2.13, iar semnificațiile fiecărui tip de dată sunt date în tabelul 2.3.

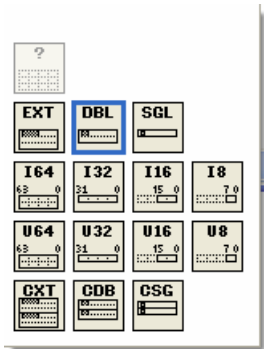


Figura 2.13

Tabelul 2.3

Simbol pe diagramă	Tip dată numerică	Reprezentare în memorie (nr. biți)	Nr. zecimale	Interval de reprezentare
EXT	Real Precizie extinsă	80 în Windows (depinde de platformă)	15 ÷ 33, depinde de platformă	Val. minimă pozitivă: 6,48E-4966 Val. maximă pozitivă: 1,19E+4932 Val. maximă negativă: - 1,19E-4932 Val. minimă negativă: - 6,48E-4966
DBL	Real Dublă precizie	64	15	Val. minimă pozitivă: 4,94E-324 Val. maximă pozitivă: 1,79E+308 Val. maximă negativă: - 1,79E+308 Val. minimă negativă: - 4,94E-324
SGL	Real Simplă precizie	32	6	Val. minimă pozitivă: 1,40E-45 Val. maximă pozitivă: 3,40E+38 Val. maximă negativă: - 3,40E+38 Val. minimă negativă: - 1,40E-45

I64	Intreg Quad cu semn	64	18	Val. minimă : -1E+19 Val. maximă : 1E+19
I32	Intreg Long cu semn	32	9	De la – 2.147.483.648 până la 2.147.483.647
I16	Intreg Word cu semn	16	4	De la -32,768 până la 32,767
I8	Intreg Byte cu semn	8	2 (cifre semnificative)	De la -128 până la 127
U64	Intreg Quad fără semn	64	19 (cifre semnificative)	De la 0 până la 2E+19
U32	Intreg Long fără semn	32	9 (cifre semnificative)	De la 0 până la 4,294,967,295
U16	Intreg Word fără semn	16	4 (cifre semnificative)	De la 0 până la 65,535
U8	Intreg Byte fără semn	8	2 (cifre semnificative)	De la 0 până la 255
CXT	Complex Precizie extinsă	256	15 ÷ 33	Partea reală şi partea imaginară la fel ca la real – precizie extinsă
CDB	Complex Dublă precizie	128	15	Partea reală şi partea imaginară la fel ca la real – dublă precizie
CSG	Complex Simplă precizie	64	6	Partea reală şi partea imaginară la fel ca la real – simplă precizie

Atunci când se leagă date numerice de reprezentări diferite la intrările aceleiaşi funcţii, LabVIEW furnizează rezultatul de tipul cel mai lung. Dacă ambele intrări au acelaşi număr de biţi dar sunt de reprezentări diferite (de ex. U32 şi I32), LabVIEW furnizează rezultatul fără semn. In exemplul din figura 2.14, adunarea dintre un număr real în reprezentare dublă precizie (8 octeţi) ai un număr întreg cu semn pe 4 octeţi I32 dă un rezultat corespunzător numărului mai lung, deci real dublă precizie. Această operaţie se numeşte *constrângere*, iar operandul constrâns este marcat printr-un punct la intrarea în funcţie.

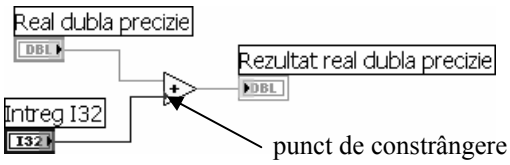


Figura 2.14

Alte tipuri de date în LabVIEW

Tabelul 2.4

Simbol pe diagramă	Tip dată	Culoare
	Boolean	Verde
	Sir de caractere (*string*)	Roz
	Enumerare	Albastru
	Vector sau matrice (*array*). Include între paranteze tipul de date al elementelor componente.	Culoarea tipului de dată al elementelor componente.
 	Ciorchine (*cluster*). Include mai multe tipuri de date. Simbolul este maro dacă datele sunt de acelaşi tip şi roz dacă datele sunt de tipuri diferite	Maro sau roz
	Cale (*path*)	Aqua
	Formă de undă (*waveform*). Include t_0 – momentul începerii achiziției, dt – incrementul şi vectorul de date.	Maro
	Număr de referință (*Reference number* sau *Refnum*)	Aqua
	Variantă. Include numele controlului sau indicatorului, informații despre tipul de date şi datele propriu-zise.	Mov
	Nume de intrare – ieşire (*I/O name*)	Mov
	Forme grafice (*Picture*)	Albastru

Exercițiul 2.3

Scop

Familiarizarea cu funcționalitatea uneltelor de pe panoul frontal, introducerea datelor într-un control, poziționarea, redimensionarea, cosmetizarea, meniul shortcut.

Mod de lucru

1. Deschideţi un nou IV.
2. Deschideți paleta de controale.

3. Aduceți pe PF primul control de tip numeric.
4. Redenumiți eticheta „Control numeric” utilizând unealta de editare text 3)

Observați că la aducerea controlului pe PF, se permite redenumirea etichetei. Dacă se face MS oriunde pe PF, se pierde acest drept, iar controlul rămâne cu eticheta implicită („Numeric”). Pentru redenumirea etichetei se foloseşte unealta 3).

5. Modificați fonturile valorii numerice a controlului utilizând bara de comenzi rapide *Text Settings* (figura 2.15), după următoarea schemă: size: 24, style: bold, color: roşu.

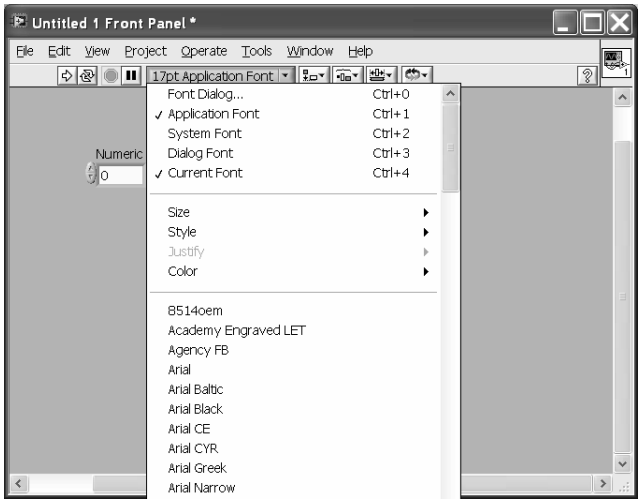


Figura 2.15

6. Experimentați modificarea valorii controlului utilizând săgețile de incrementare/decrementare.
7. Introduceți în control valoarea: 2.49216 de la tastatură în modul următor:
   - poziționați prompterul în interiorul displayului numeric al controlului. Dacă pe paleta de unelte, *Automatic Tools Selection* este activ (beculeţul verde este aprins), veți observa că prompterul se schimbă automat în unealta de editare text 3). Dacă selectarea uneltelor se face manual (*Automatic Tools Selection* este inactiv), selectați unealta 1) sau unealta 3).
   - Scrieți de la tastatură valoarea 2.49216 şi apoi apăsați tasta *Enter*.
8. Cu aceeaşi unealtă 3), poziționați prompterul după cifra 9.
9. Apăsați săgețile de defilare sus/jos de pe tastatură. Veți observa incrementarea/decrementarea valorii sutimilor. Realizați acelaşi lucru şi pentru modificarea miimilor, etc. Aceasta este o metodă de reglaj fin al unui control numeric.
10. Deschideți şi studiați meniul shortcut al controlului.
11. In opțiunea *Data Range*, debifați *Use Default Range*.
12. Scrieți la *Increment* valoarea 0.01 şi apăsați OK.

13. Manevrați săgețile de incrementare/decrementare ale controlului şi observați modul în care se schimbă valoarea acestuia.
14. Bifați din nou opțiunea *Use Default Range*.
15. Modificați precizia de afişare la două zecimale. Observați modul în care se face trunchierea numărului.
16. Experimentați reprezentarea numărului în formatele *scientific* şi *SI notation*, cu diferite zecimale şi cifre semnificative.
17. Introduceți de la tastatură numărul 2.5.
18. Stabiliți din opțiunea *Format and Precision* numărul de zecimale 0. Observați modul în care se face trunchierea.
19. Introduceți de la tastatură numărul 3.5 şi observați din nou cum se face trunchierea.

Ca regulă, dacă valoarea unei zecimale este exact 0,5 din valoarea poziției imediat anterioare, programul rotunjește întotdeauna la numărul par cel mai apropiat. De exemplu 7,5 este rotunjit la 8, iar 8,5 este rotunjit tot la 8.

20. Selectați controlul utilizând unealta de selectare 2).
21. Experimentați repoziționarea lui, ştergerea, decuparea, copierea.

In cazul în care s-a executat o comandă nedorită sau se doreşte la revenirea unei comenzi anterioare, există *Undo* în meniul *Edit* sau se tastează *Ctrl-Z.*

22. Experimentați redimensionarea controlului.
23. Înlocuiți controlul numeric de mai sus cu un control de tip *knob* utilizând opțiunea *Replace* din meniul shortcut.
24. Experimentați redimensionarea acestui control. Observați că se permite redimensionarea atât a butonului rotativ, cât şi a scalei.
25. Afişați displayul digital al controlului din meniul shortcut, opțiunea *Visible Items – Digital Display*.
26. Displayul digital are propriul meniu shortcut. Deschideți-l şi observați diferența față de meniul controlului propriu-zis.
27. Modificați valorile capetelor de scală între valorile -100 şi 100 cu ajutorul uneltei de editare a textului, 3).
28. Din meniul shortcut, experimentați opțiunile submeniului *Scale*.

Opțiunea *Format & Precision* de pe meniul shortcut a controlului este aceeaşi cu cea a displayului digital. Opțiunea *Format & Precision* a scalei se referă la modul cum se face inscripționarea diviziunilor semnificative doar ale scalei.

29. Adăugați controlului un nou ac indicator cu opțiunea *Add Needle* din meniul shortcut. Acum controlul seamănă cu un potențiometru dublu. Observați apariția celui de-al doilea display digital. Valorile furnizate de acest control sunt reunite într-un *cluster*.
30. Adăugați şi alte controale pe panoul frontal şi studiați-le meniul shortcut, posibilitățile de redimensionare, scalele, etc.

Diagrama de legături

Diagrama de legături (DL) reprezintă programul propriu-zis dezvoltat în mediul LabVIEW şi conține codul sursă al instrumentului virtual. Fiecare obiect de pe PF (C sau I) are un corespondent pe DL, care se numeşte *terminal*. Scrierea codului sursă constă în realizarea de legături sub formă grafică între aceste terminale şi diverse funcții, astfel încât IV-ul să satisfacă cerințele impuse.

De pe PF, deschiderea DL se face în următoarele moduri:

- din meniul *Window – Show Block Diagram*
- dublu clic pe orice C sau I de pe PF
- apăsarea tastei *Ctrl-E*
- de pe bara de programe

De pe DL, trecerea spre PF se face în modul următor:

- din meniul *Window – Show Front Panel*
- dublu clic pe orice terminal de pe DL
- apăsarea tastei *Ctrl-E*
- de pe bara de programe

O DL conține următoarele elemente:

- terminale
- noduri
- fire de legătură
- structuri

Terminalele sunt porturi de intrare-ieşire ce fac legătura dintre PF şi DL. Ele sunt corespondentele obiectelor de pe PF şi se reprezintă pe DL printr-un simbol care este în concordanță cu tipul de dată vehiculată de obiect. Terminalele pot fi şterse de pe DL doar odată cu obiectul de pe PF. Mutarea unui terminal pe DL nu conduce la mutarea obiectului corespunzător de pe PF. Terminalele cu conturul îngroşat corespund controalelor, iar cele cu conturul subțire corespund indicatoarelor de pe PF.

Nodurile sunt obiecte de pe DL care sunt caracterizate de un număr de intrări/ieşiri şi execută diverse operații şi funcții în timpul rulării IV-ului. Nodurile sunt analoge instrucțiunilor, funcțiilor şi subrutinelor din limbajele de programare bazate pe text.

Firele de legătură transferă datele între obiectele de pe DL. Fiecare fir are o singură sursă de date, dar oricât de mulți receptori. Firele au diferite culori, stiluri şi grosimi ce depind de tipurile de date vehiculate. Un fir rupt, ce nu poate transporta date, se reprezintă printr-o linie întreruptă. Un fir cu două surse de date este un fir rupt.

Structurile sunt reprezentări grafice ale instrucțiunilor de ciclare şi condiționare din programarea bazată pe text. Se utilizează pentru repetarea unor blocuri de cod sau pentru execuția unor coduri condiționat sau într-o anumită ordine.

Firele de legătură au diverse forme, grosimi şi culori în funcţie de tipul datelor vehiculate prin acestea. In tabelul 2.5 sunt date principalele tipuri de fire ce vehiculează date într-un IV.

Tabelul 2.5

	Scalar, boolean, enumerare, număr de referinţă. Culoarea depinde de tipul de dată
	Vector (matrice cu o dimensiune). Culoarea depinde de tipul elementelor componente.
	Matrice multidimensională. Culoarea depinde de tipul elementelor componente.
	Cluster, variantă
	Formă de undă
	Toate celelalte tipuri de date.

Meniul shortcut al terminalelor de pe DL

Pe DL, accesarea meniului shortcut se face în acelaşi mod ca şi pe PF. Există însă unele diferenţe între meniul obiectului de pe PF şi cel al terminalului corespunzător de pe DL. Câteva din acestea sunt:

- în submeniul *Visible Items* este accesibilă doar eticheta
- *Hide Control (Indicator)* face ca obiectul corespunzător să nu apară pe PF. Dacă obiectul este deja ascuns, opţiunea se transformă în *Show Control (Indicator)*.
- *Change to Constant* schimbă terminalul în constantă, care este vizibilă şi accesibilă doar pe DL. La accesarea acestei opţiuni, obiectul de pe PF dispare.
- *Create: Constant, Control* sau *Indicator* creează automat tipul de obiect corespunzător terminalului. Această opţiune este deosebit de utilă la construirea terminalelor corespunzătoare ieşirilor sau intrărilor unor funcţii la care nu se cunoaşte exact tipul de dată. Prin crearea terminalului se realizează automat şi legătura cu acesta. Dacă, de exemplu, la ieşirea unei funcţii se selectează *Create Control*, terminalul se creează, dar nu se realizează legătura deoarece firul ar avea două surse: ieşirea funcţiei şi controlul creat.
- *Create: Local Variable, Reference, Property Node, Invoke Node* creează automat o variabilă locală, o referinţă, un nod de proprietăţi sau un nod de invocare relativ la C/I în cauză.

Paleta de funcții

Paleta de funcții este o fereastră ce se deschide doar de pe DL. Aceasta conține operatori, funcții, noduri, structuri şi subIV-uri cu ajutorul cărora se construieşte programul în LabVIEW. Accesarea paletei de funcții se face în modurile următoare:

- *View – Show Functions Palette*
- *MD pe DL.*

In cel de-al doilea mod, paleta poate fi fixată ca fereastră pe ecran prin MS pe pioneza din stânga sus a acestei ferestre. Altfel, paleta se închide după realizarea unui MS oriunde pe DL. Paleta de funcții este prezentată în figura 2.16.

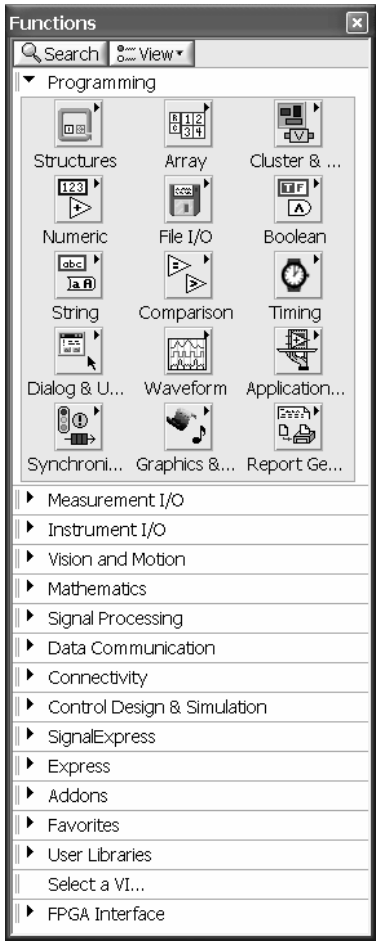


Figura 2.16

Funcţiile pe paletă sunt grupate după destinaţia lor. Cele mai utilizate sunt cele din subpaleta *Programming*, care se expandează în mod implicit. Celelalte sunt biblioteci care pot fi apelate de către utilizator la construirea de aplicaţii dedicate. De exemplu, dacă se realizează un IV necesar procesării unor semnale achiziţionate în prealabil, se utilizează funcţii din subpaleta *Signal Processing*. Dacă este necesar un aparat matematic complex, avem la dispoziţie biblioteca *Mathematics*. Multe din aceste subpalete nu sunt populate cu funcţii. Adăugarea de biblioteci acestor subpalete se face prin instalarea toolkit-urilor dedicate puse la dispoziţie de National Instruments. Obiectul cursului de faţă este studierea modului de apelare şi de utilizare a unor funcţii doar din subpaleta *Programming*.

Există o mare varietate de funcţii în LabVIEW. Diversitatea şi compexitatea acestora creşte pe măsură ce National Instruments scoate noi variante ale limbajului de programare.

Meniul shortcut al funcţiilor

Fiecare funcţie are disponibil un meniu shortcut, cu ajutorul căruia se stabilesc o serie de parametri, se pot vizualiza etichete (funcţia poate fi etichetată, la fel ca orice terminal), se poate apela helpul mare, se pot vizualiza exemple, se pot face descrieri ale funcţiei sau se pot stabili punct de întrerupere în vederea depanării programului. Aceste opţiuni sunt comune tuturor funcţiilor, în prima parte a meniului, prin opţiunile:

- *Visible Items*
- *Help*
- *Examples*
- *Description and Tip…*
- *Set Breakpoint*

Pe lângă acestea, mai există o serie de opţiuni specifice fiecărei funcţii. O opţiune comună mai este *Replace*, prin care se permite utilizatorului înlocuirea funcţiei cu o alta prin deschiderea automată a paletei de funcţii.

Meniul shortcut al funcţiei poate diferi de meniul shortcut al intrărilor şi ieşirilor. Intrările şi ieşirile conţin în meniu opţiunea *Create*, prin care utilizatorul poate crea automat tipul de dată vehiculat de intrarea/ieşirea respectivă.
Un caz particular de funcţii în reprezintă structurile, la care meniul shortcut se deschide prin realizarea unui MD pe marginea cadrului structurii.

Apelarea helpului în LabVIEW

LabVIEW prezintă un puternic sistem de helpuri prin care utilizatorul este ghidat şi îndrumat în activitatea de programare. In plus, utilizatorul are la dispoziţie un număr mare de exemple, grupate pe categorii de aplicaţii, în care sunt

exemplificate diverse situații de operare a unor funcții şi în care se pot găsi de multe ori soluții tehnice. Atât helpurile, cât şi exemplele, sunt disponibile în meniul principal *Help*, accesabil atât de pe PF cât şi pe DL. De remarcat faptul că National Instruments pune la dispoziție o bibliotecă a utilizatorilor accesibilă pe Internet denumită *LabVIEW Zone*, în care dezvoltatorii de instrumente virtuale pot găsi soluții la problemele lor, dar pot oferi ei înşişi soluții prin posibilitatea de a posta în această bibliotecă propriile IV-uri. Local, există două tipuri de helpuri ale programului:

- "helpul mic", apelabil din meniul principal *Help – Show Context Help*, prin care se deschide o fereastră permanentă în care se afişează informații sumare despre funcția de pe DL sau obiectul de pe PF. Informația este afişată doar la suprapunerea prompterului mouse-ului peste acel obiect sau funcție (fără a se face clic). De asemenea, helpul mic furnizează informații despre funcțiile din paleta de funcții la simpla accesare a acestora cu mouse-ul.
- "helpul mare", specific doar funcțiilor de pe DL, apelabil din meniul shortcut al funcției (MD pe funcție), opțiunea *Help*. „Helpul mare" se mai poate deschide şi apăsând pe linkul Detailed help din „helpul mic". Aici se dau informații mai detaliate privind funcția, inclusiv exemple sau aplicații.

C şi I de pe PF afişează în fereastra helpului mic textul care a fost introdus în opțiunea *Description and Tip* din meniul shortcut.

Exercițiul 2.4

Scop

Trecerea în revistă a funcțiilor principale din paleta de funcții.

Mod de lucru

1. Deschideți un nou IV.
2. Deschideți diagrama instrumentului.
3. Deschideți paleta de funcții şi fixați-o pe diagramă.
4. Treceți în revistă funcțiile din submeniul *Programming* (cele care se deschid pe paletă în mod implicit). Observați că funcțiile sunt grupate în subpalete, fiecare subpaletă conținând funcții specifice. Acestea sunt:
   - *Structures* conține funcții legate de structuri
   - *Array* conține funcții de prelucrare a matricilor. Aceste funcții nu sunt dedicate operațiilor cu matrici, care se găsesc în subpaleta *Mathematics – Linear Algebra.*
   - *Cluster & Variant* conține funcții de creare şi manipulare a clusterelor şi de conversie şi manipulare a datelor de tip variantă (*variant*).

- *Numeric* conține operații și funcții cu numere reale și complexe.
- *File I/O* conține funcții și subIV-uri de scriere și citire a datelor în și din fișiere.
- *Boolean* conține operații logice cu numere în sistem boolean.
- *String* este subpaleta ce conține funcții de manipulare a șirurilor de caractere.
- *Comparison* conține funcții de comparare a datelor de tip numeric, boolean, șir de caractere, matrici și cluster.
- *Timing* conține subIV-uri și funcții de determinare a vitezei de efectuare a unor operații în raport cu un timp de referință sau de citire a datei și orei calculatorului.
- *Dialog & User Interface* permite implementarea unor interfețe de comunicare cu utilizatorul în scopul introducerii interactive a datelor sau furnizării de instrucțiuni.
- *Waveform* conține subIV-uri și funcții de construire a datelor de tip formă de undă (*waveform*), ce cuprind eșantioane și atribute (informații despre canalele de achiziție, timp), precum și recuperarea acestor informații în scopul procesării.
- *Application Control* oferă posibilitatea controlului programatic al IV-urilor și al altor aplicații pe computerul local sau prin rețea.
- *Synchronization* conține funcții necesare sincronizării mai multor sarcini de efectuat în același timp, precum și transferului de date între aceste sarcini.
- *Graphics $ Sound* permite personalizarea afișării datelor sub formă grafică, importul și exportul informației din fișiere grafice precum și înregistrarea și redarea sunetului.
- *Report Generation* facilitează construirea de rapoarte.

5. Citiți helpul fiecărei subpalete.
6. Deschideți meniul *Mathematics* și treceți în revistă subpaletele și funcțiile componente.
7. Deschideți meniul *Signal Processing* și observați funcțiile dedicate procesării semnalelor.

Unele subpalete se găsesc în mai multe locuri în paleta de funcții. De exemplu, subpaleta de funcții numerice *Numeric* se găsește atât în subpaleta *Programming*, cât și în subpaleta *Mathematics*.

8. Găsiți și alte subpalete localizate în mai multe locuri în paleta de funcții.
9. Plasați pe DL operatorul *Add* din subpaleta *Numeric*.
10. Suprapuneți prompterul peste una din intrări. Dacă este activat selectorul automat de unelte, prompterul se va schimba în unealta de legături  . Observați cum intrarea este semnalizată prin clipire. Realizarea unui MS pe intrare cu unealta de legături duce la inițierea unui traseu de date.
11. Deschideți „helpul mic”, apoi „helpul mare” ale funcției *Add* și studiați-le.
12. Plasați pe DL funcția *Search and Replace String* din submeniul *String*.

13. Treceți prompterul peste această funcție. Observați în „helpul mic” pictograma funcției şi conectorul acesteia. Comparativ cu funcția *Add*, această funcție este mai complexă, având mai multe intrări şi ieşiri. Citiți în help destinația funcției şi modul de utilizare a intrărilor.
14. Treceți prompterul peste fiecare intrare şi ieşire. Observați cum acesta se schimbă în unealta de legături şi de asemenea apare denumirea fiecărei intrări.

Intrărilor ale căror nume sunt scrise cu aldine (bold) trebuie legate obligatoriu în diagramă, altfel programul dă eroare. Intrările care nu sunt scrise cu aldine pot sau nu să fie legate pe diagramă. Dacă sunt legate, ele vor utiliza datele care sunt disponibile prin legături. Dacă nu sunt legate, se vor utiliza valorile implicite (indicate între paranteze).

15. Realizați un dublu click pe funcția *Search and Replace String*. Observați că nu se întâmplă nimic. Aceasta înseamnă că funcția face parte din biblioteca internă a LabVIEW, iar codul ei nu poate fi vizualizat.
16. Plasați pe diagramă funcția *Sine Waveform* din subpaleta *Signal Processing – Waveform Generation*. Această funcție mai poate fi găsită şi în subpaleta *Waveform – Analog Waveform – Waveform Generation.*
17. Studiați helpul funcției.
18. Realizați dublu click pe această funcție. Observați că se deschide panoul frontal al acesteia, semn că funcția este reprezentată în bibliotecă printr-un subIV.
19. Deschideți DL şi realizați dublu click pe pictograma funcției *Sine Wave*  . S-a deschis un alt panou frontal.
20. Treceți pe DL şi observați existența nodului cod de interfață din figura 2.17. Acesta conține programul funcției scris într-un limbaj procedural (C++) şi compilat. Nodul cod de interfață are intrări şi ieşiri ca o funcție obişnuită, însă codul sursă nu poate fi cunoscut. Acesta este ultimul nivel al ierarhiei IV-ului.

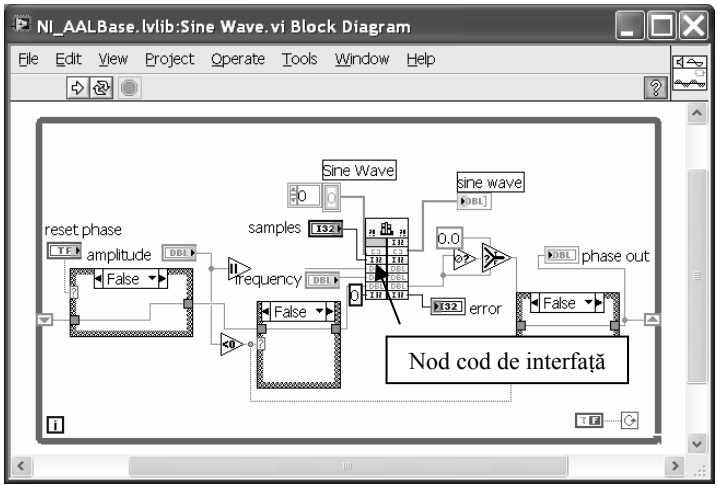


Figura 2.17

Funcția *Sine Wave* poate fi găsită direct în submeniul *Signal Processing – Signal Generation*. In general, sub multe funcții în bibliotecile LabVIEW din paleta de funcții care sunt construite sub formă de subIV-uri utilizând o serie de funcții elementare preluate fie din subpaleta *Programming*, fiind utilizând noduri cod de interfață. Tendința este de a realiza biblioteci cu funcții cât mai complexe şi cât mai uşor accesibile utilizatorilor chiar fără o pregătire de specialitate prea avansată. In acest scop există şi subpaleta de funcții *Express*, care conține o serie de funcții de bază destinate achiziției, generării, manipulării şi procesării semnalelor, precum şi pentru construirea unor IV-uri simple, la care introducerea datelor şi fixarea parametrilor se face prin ferestre interactive cu realizarea automată a unei mari părți din legături, astfel încât utilizatorul să obțină instrumentul fără prea mult efort de programare.

21. Adăugați şi verificați ierarhia până la nivelul codului de interfață şi pentru funcțiile:
    - *Signal Processing – Waveform Measurements – Basic Averaged DC-RMS.vi*
    - *Mathematics – Linear Algebra – Solve Linear Equation.vi*
    - *Measurement I/O – DAQmx Data Acquisition – DAQ Assistant*

Exercițiul 2.5

Scop

Construirea unui instrument virtual care să execute adunarea a două numere fixate pe două controale numerice de pe panoul frontal şi să afişeze rezultatul pe un indicator numeric.

Mod de lucru

1. Construiți pentru început un director nou pe discul dumneavoastră de lucru (denumiți-l de exemplu *Lucru Labview*) în care vom salva în continuare toate IV-urile pe care le vom crea.
2. Porniți programul LabVIEW şi deschideți un nou IV.
3. Plasați pe PF un control de tip *Numeric* şi unul de tip *Knob*.
4. Etichetați cele două controale cu etichetele „1”, respectiv „2”.
5. Editați *Caption* cu „Operator 1”, respectiv „Operator 2”, scris cu albastru, arial, 18 pt, bold.
6. Pentru controlul *Knob*, arătați şi indicatorul digital cu *Visible Items – Digital Display*.
7. Stabiliți pentru cele controale digitale precizia de afişare la 2 cifre zecimale.
8. Deschideți diagrama de legături. Plasați pe diagramă operatorul de adunare *Add* din subpaleta de funcții *Numeric*.
9. Realizați legăturile intrărilor funcției cu cei doi operatori în modul următor:

- Selectați unealta de legături . Dacă sunteți pe selectorul automat, atunci când plasați prompterul pe una din intrările funcție *Add*, acesta se va schimba automat în unealta de legături.
- Observați cum, aducând unealta peste intrările funcției, acestea clipesc şi în acelaşi timp apare eticheta lor implicită.
- Realizați un MS pe intrarea de sus.
- Mergeți cu M până în dreptul terminalului 1 şi realizați un nou MS.
- Schimbați direcția înspre terminalul 1 şi continuați traseul până pe terminalul 1.
- Faceți MS pe terminalul 1. In acest moment trebuie să aveți realizată legătura, care se va colora automat în culoarea portocalie.
- Faceți acelaşi lucru şi cu terminalul 2.

10. Vom crea acum automat indicatorul de ieşire. Plasați unealta de legături pe ieşirea funcției *Add*.
11. Faceți MD pe ieşire şi deschideți meniul shortcut.
12. Selectați în meniu *Create – Indicator*.
13. Schimbați eticheta indicatorului în „1+2”.
14. Treceți pe PF şi editați *Caption* cu „Rezultat adunare”.
15. Schimbați precizia de afişare a indicatorului numeric la 3 cifre zecimale.
16. Modificați textul displayului digital al indicatorului scris cu roşu, arial, 24 pt, bold, pe fond alb.

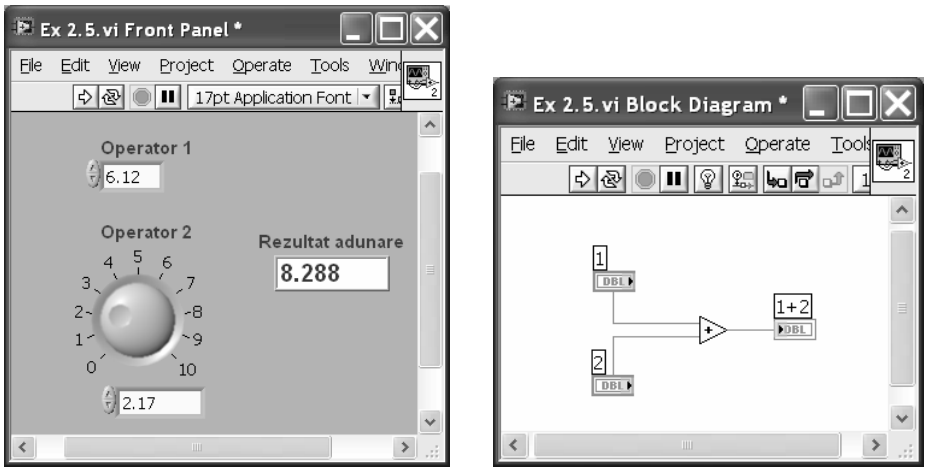


Figura 2.18

17. Treceți din nou pe DL şi deschideți helpul mic şi helpul mare pentru operatorul de adunare.
18. Schimbați unul din operatori din control în indicator. Firul de legătură s-a întrerupt deoarece a rămas fără sursă de date.

19. Mergeți cu M pe butonul *Run* al barei de comenzi rapide. Observați că acest buton este sub forma unei săgeți întrerupte, care semnifică faptul că programul are erori de sintaxă (*List errors*).

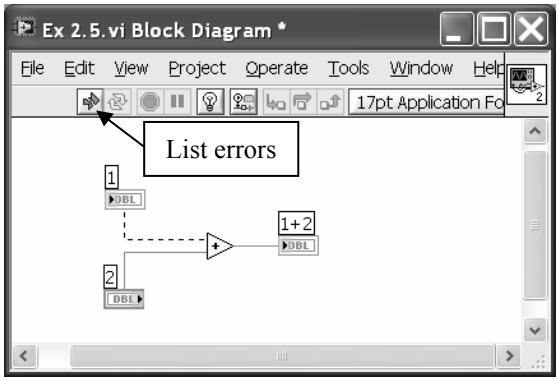


Figura 2.19

20. Apăsați acest buton. Studiați şi interpretați mesajele de eroare din fereastra deschisă (figura 2.20).

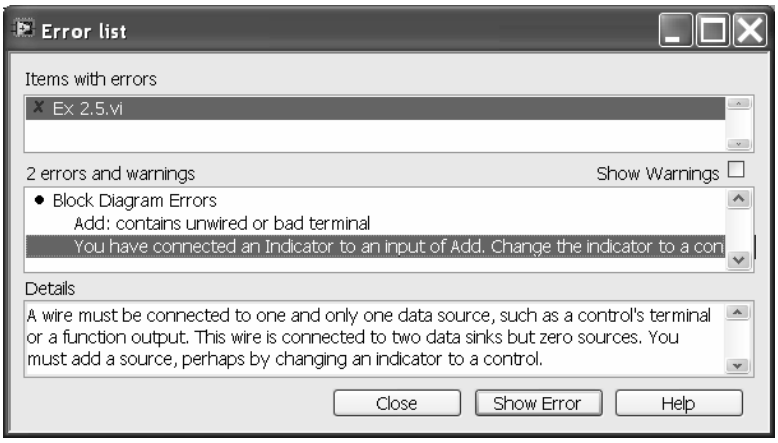


Figura 2.20

21. Schimbați din nou operatorul din indicator în control pentru a se reface legătura.
22. Treceți pe PF şi modificați valoarea controalelor.
23. Stabiliți ca implicite valorile controalelor din meniul *Edit – Make Current Values Default*.
24. Salvați instrumentul sub numele *Ex 2.5.vi*.
25. Porniți instrumentul de la butonul *Run* şi observați efectuarea operației.

26. Apăsați butonul *Run Continuously* şi manevrați controalele urmărind valoarea rezultatului.
27. Opriți rularea instrumentului de la butonul *Abort Execution*.
28. Adăugați pe DL operația de înmulțire.
29. Creați şi denumiți indicatorul operației de înmulțire ca „Rezultat inmultire”.
30. Rulați instrumentul şi urmăriți rezultatele.

Utilizarea testerelor

In vederea depanării, LabVIEW pune la dispoziție sonde de testare sau testere (*Probe* în limba engleză), care pot fi activate pe firele de legătură cu scopul a urmări valoarea datelor vehiculate prin fir la un moment dat. Testerele pot fi plasate pe orice fir de legătură realizând MD pe fir şi alegând opțiunea *Probe*, când se deschide o fereastră ce conține un indicator de tipul datelor vehiculate de fir. In timpul rulării instrumentului, datele pot fi observate pe acest indicator.

Testerele sunt numerotate automat în ordinea plasării lor. Ferestrele testerelor sunt disponibile atât pe PF, cât şi pe DL.

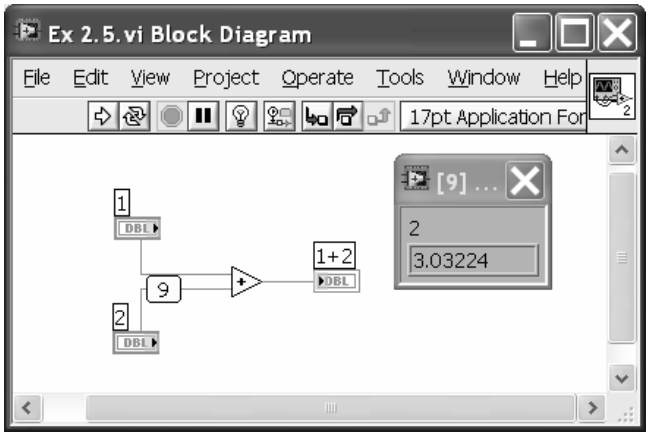


Figura 2.21

31. Plasați câte un tester pe fiecare fir de pe DL, rulați instrumentul şi observați cum se schimbă valorile.

Unități de măsură

LabVIEW oferă posibilitatea lucrului cu unități de măsură. Unitatea de măsură poate însoți datele asociate unui control sau indicator şi suferă aceleaşi transformări prin intermediul operațiilor şi funcțiilor ca şi datele însele. Unitățile de măsură

trebuie însă utilizate cu precauție, deoarece se pot face uşor erori datorită incompatibilității acestora la intrările şi/sau ieşirile funcțiilor.

Asocierea unității de măsură unui C sau I se face în modul următor:

- din meniul shortcut al C/I, se selectează *Visible Items – Unit Label*
- se tastează în dreptunghiul negru un caracter (m de exemplu)
- se face MD pe acest caracter şi se selectează *Build Unit String*.
- în cazul în care caracterul tastat coincide cu o unitate de măsură, se deschide o fereastră unde este subliniată unitatea de măsură specificată (în cazul nostru, metrul). Dacă caracterul tastat nu reprezintă o unitate de măsură, acesta este însoțit pe etichetă de un semn „ ? ”, iar fereastra care se deschide conține toate unitățile de măsură disponibile în LabVIEW, grupate pe tipuri de mărimi (figura 2.22).

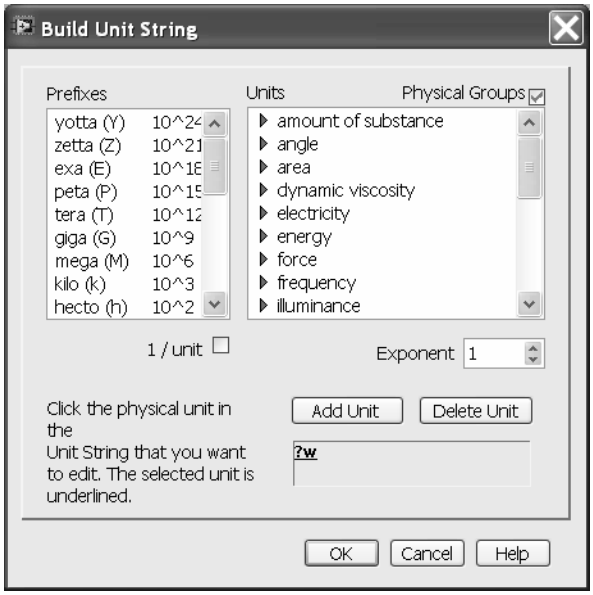


Figura 2.22

- în secțiunea *Units* se face dublu click pe mărimea dorită. Se vor expanda toate unitățile disponibile, atât cele din sistemul internațional de unități, cât şi cele derivate şi tolerate.
- se alege unitatea dorită (eventual se selectează şi un prefix, din secțiunea *Prefixes*).

Exercițiul 2.6

Scop

Conversia unei unități de măsură într-o altă unitate de măsură.

Mod de lucru

1. Deschideți un nou IV.
2. Plasați pe PF un control numeric şi un indicator numeric.
3. Faceți legătura dintre control şi indicator.
4. Asociați controlului unitatea de măsură de presiune *bar*, iar indicatorului unitatea de măsură *torr*, după metoda de mai sus.
5. Dați controlului valoarea 1.
6. Rulați instrumentul. Valoarea indicatorului arată câți torr reprezintă un bar.
7. Faceți după modelul de mai sus următoarele conversii: acre – m^2, calorie termică – joule, newton – dyne, foot – inch, gauss – oersted.

Unitatea m^2, care nu există în lista unităților, se scrie m^2.
Se pot face conversii numai între unitățile de măsură ale aceleiaşi mărimi fizice

Conceptul flux de date (*Data flow*)

Programele scrise în limbaje procedurale, bazate pe instrucțiuni text (Visual Basic, C/C++, Java, etc.), sunt executate secvențial, după conceptul „flux controlat", în care instrucțiunile se execută într-o ordine prestabilită prin program. Limbajul G, după care funcționează LabVIEW, foloseşte tehnica „fluxului de date" (*data flow*), în care datele şi efectuarea funcțiilor şi a nodurilor se realizează în paralel. Chiar în interiorul unui IV programul lucrează multitasking, în sensul că se pot executa mai multe funcții în acelaşi timp, cu condiția ca acestea să aibă toate datele disponibile le intrare. După acest concept, pot rula în acelaşi timp mai multe IV-uri. Aşadar, un nod care are disponibile datele la toate intrările, este efectuat indiferent de starea celorlalte noduri. Evident însă, dacă ieşirea unui nod reprezintă intrare pentru alt nod, cel de-al doilea nod va trebui să aştepte până ce este efectuat cel dinainte. Un model al conceptului „data flow" este ilustrat în figura 2.23. In figură se prezintă modul în care se realizează funcțiile *f* şi *g*, ambele de variabile A şi B, în cazul programării procedurale şi al programării în LabVIEW.

In cazul a) întâi se citesc datele A, apoi cele B, după care se efectuează funcția f(A,B), apoi funcția g(A,B). De remarcat că funcția g se efectuează abia după efectuarea funcției f, deşi datele de intrare sunt disponibile pentru ambele funcții. E posibil însă ca datele B să fie disponibile înaintea datelor A. Programul aşteaptă întâi datele A, apoi le citeşte pe cele B, ducând la pierdere de timp.

In cazul b), nu există o prioritate între intrările A şi B, fiecare dată fiind citită de îndată ce este disponibilă. Mai mult, funcțiile f şi g se efectuează independent una de cealaltă, de îndată ce datele de intrare sunt disponibile. In felul acesta se obține o economie substanțială de timp, esențială în aplicațiile în care operațiile trebuie să se succeadă cu viteză (ex.: achiziții de semnale de frecvență ridicată şi prelucrarea concomitentă a informației).

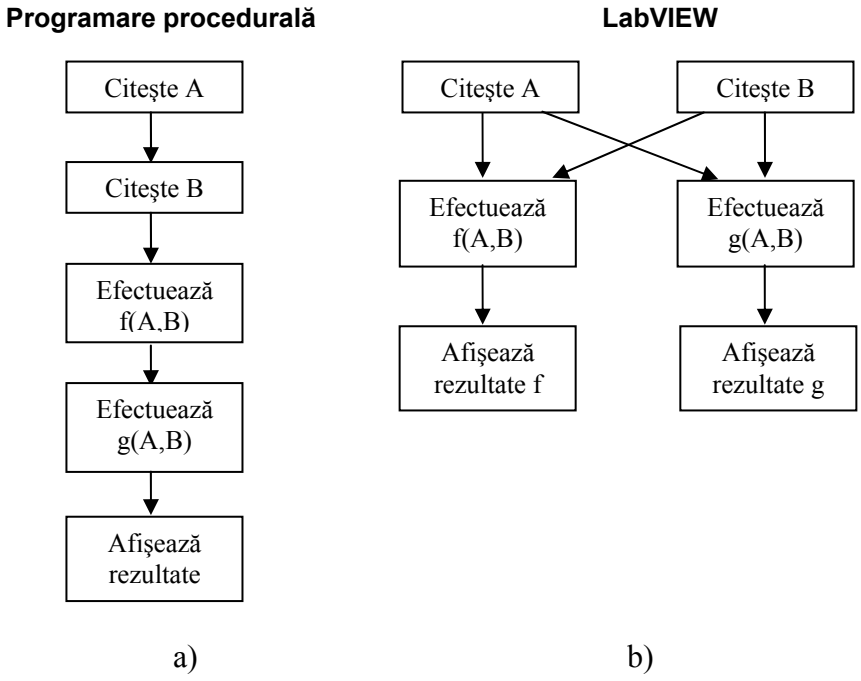


Figura 2.23

Alocarea şi managementul memoriei se face automat de către program, doar pentru datele care sunt în lucru. După ce datele cu care s-a lucrat nu mai sunt necesare, memoria se eliberează. In felul acesta se realizează şi o economie importantă de resurse.

Exercițiul 2.7

Scop

Înțelegerea conceptului „data flow”.

Mod de lucru

1. Încărcați instrumentul *Ex 2.5.vi*.
2. Adăugați instrumentului următoarele operații (figura 2.24):
   - scăderea din Operator 1 a unui număr aleatoriu cuprins între 0 şi 1 (funcția *Random Number (0,1)*, care se găseşte în subpaleta *Numeric*).
   - înmulțirea dintre Operator 2 şi sin(Operator 1). Funcția *sin* se găseşte în paleta de funcții în subpaleta *Mathematics – Elementary & Special Functions – Trigonometric*.

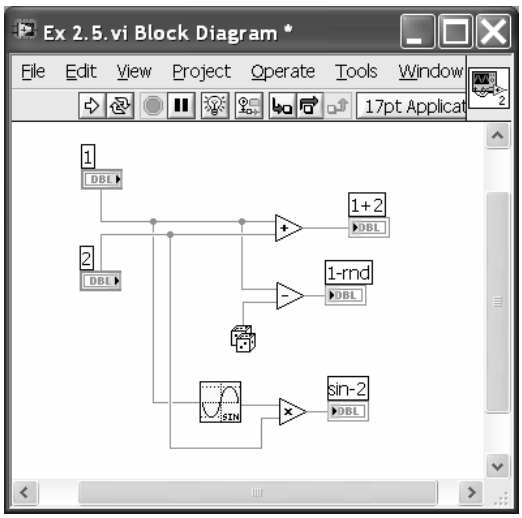


Figura 2.24

3. Pe DL, apăsați butonul *Highlight Execution*. Această comandă permite urmărirea animată a executării programului, cu evidențierea valorii datelor la ieşirile funcțiilor.
4. Rulați instrumentul şi urmăriți modul în care se efectuează funcțiile şi parcursul fluxului de date.

> Observați executarea în paralel a funcțiilor, independent unele de altele, de îndată ce sunt disponibile datele la intrare. Deoarece unele operații, cum ar fi înmulțirea sau calculul sinusului, sunt mai complexe şi necesită un timp de calcul mai îndelungat, momentele de afişare a rezultatelor sunt diferite. Efectuarea înmulțirii nu începe până când nu a fost realizată operația anterioară, calculul lui sin(operator 1).

5. Salvați instrumentul sub numele *Ex 2.6.vi* în modul următor:
   - Din meniul *File* selectați *Save As*.
   - In fereastra deschisă apăsați *Continue.*
   - In noua fereastră tastați numele noului fişier *Ex 2.6.vi*.
   - Apăsați *OK*.

Capitolul 3

REALIZAREA UNUI SUBIV

Una din trăsăturile cele mai importante ale mediului de programare LabVIEW este modularitatea, adică un IV creat cu un scop poate fi inclus într-un alt IV ca *subIV*, care la rândul lui poate şi el fi inclus într-un alt IV. Se creează astfel o structură ierarhică a IV-ului, mult mai uşor de urmărit şi de organizat. De asemenea, un mare avantaj al structurii modulare îl constituie faptul că un subIV poate fi apelat ca şi nod în oricâte alte IV-uri principale, toate rulând în acelaşi timp independent unele de altele.

Dacă din diagrama instrumentului principal se face un dublu-click pe pictograma unui subIV, se deschide panoul său frontal şi diagrama de legături. La rândul său DL a subIV-ului deschis poate conţine alte subIV-uri, care pot fi deschise în acelaşi mod. Pictograma subIV-ului se află în colţul din dreapta sus al PF sau a DL. SubIV-urile se realizează atunci când pe DL a unui IV sunt operaţii care se repetă, când acea operaţie este necesară şi în alte IV-uri, sau cu scopul unei organizări mai compacte a unei diagrame, în general foarte largi.

SubIV-urile din LabVIEW au drept corespondent subrutinele în programarea procedurală bazată pe text. In figura 3.1 este dată echivalenţa dintre subIV-uri şi subrutinele din programarea în C, pentru funcţia *mediere*.

Cod funcţie în C	**Apelarea în programul principal**

function mediere (in1, in2, aut)
{
out = (in1 + in2)/2.0;
}
	main
	{
	mediere(op1, op2, rezultat)
	}

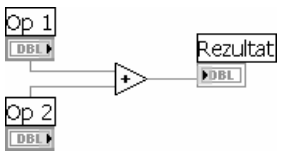
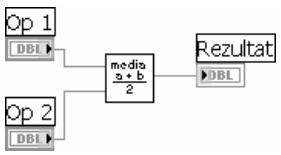
Diagrama bloc a subIV-ului	**Apelarea în DL a IV-ului principal**
	

Figura 3.1

Un subIV se realizează ca un IV oarecare, numai că pictogramei acestuia i se ataşează terminale de intrare-ieşire pentru includerea lui în alte diagrame. Etapele de realizare a subIV-ului vor fi ilustrate mai jos printr-un exemplu.

Exercițiul 3.1

Scop

Realizarea unui subIV care să efectueze adunarea şi înmulțirea a două numere aplicate la intrare.

Mod de lucru

Realizarea funcțională a subIV-ului

1. Deschideți un nou IV.
2. Construiți panoul frontal şi diagrama de legături ca în figura 3.2. Etichetați intrările cu *a* şi *b*, iar rezultatele adunării şi înmulțirii cu *suma*, respectiv *produs*.

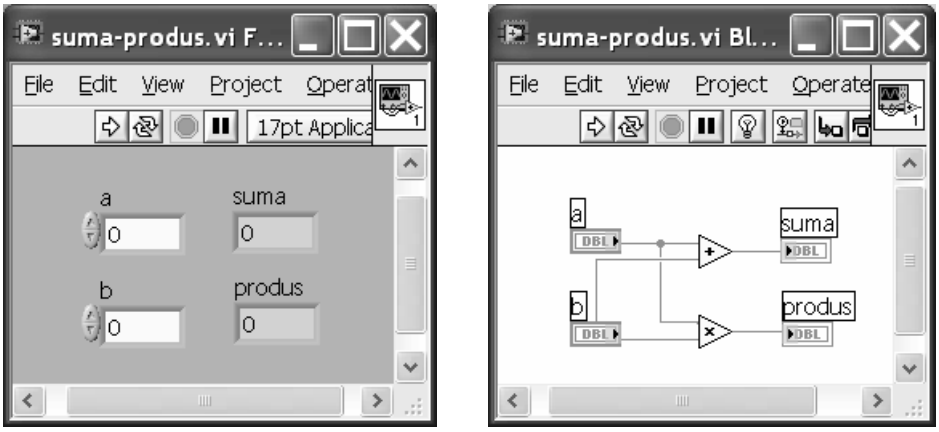


Figura 3.2

3. Dați valori controalelor numerice şi verificați funcționalitatea IV-ului.
4. Salvați IV-ul sub numele *Suma-produs.vi*.

Conectarea terminalelor pictogramei

5. Treceți pe PF.
6. Faceți MD pe pictograma din dreapta sus a PF şi selectați *Show Connector*.

Configurația de terminale care se deschide în mod implicit (fiecare pătrățel reprezintă un terminal) acoperă necesarul de intrări şi de ieşiri corespunzătoare

numărului de controale (considerate intrări) şi de indicatoare (considerate ieşiri) de pe PF al subIV-ului. De regulă, terminalele de intrare se stabilesc în partea stângă a pictogramei, iar cele de ieşire în dreapta, considerând fluxul de date circulând de la stânga spre dreapta diagramei. Terminalele care rămân nelegate nu sunt luate în considerare. Sunt posibile până la 28 de terminale, însă dacă se atribuie mai mult de 16 terminale unei pictograme, aceasta devine ilizibilă şi greu de utilizat.

7. Faceţi din nou MD pe pictograma cu terminale şi selectaţi *Patterns*.
8. Din întreaga tabelă cu opţiuni de configuraţii care se deschide, selectaţi configuraţia minimală cu 4 terminale (2 intrări şi 2 ieşiri)  .
9. Observaţi că aducând M peste pictogramă, acesta se transformă automat în unealta de legături (care este specifică doar realizării legăturilor de pe DL). Faceţi MS pe pătrăţelul din colţul din stânga sus a configuraţiei de terminale, apoi iarăşi MS pe controlul *a* (figura 3.3).

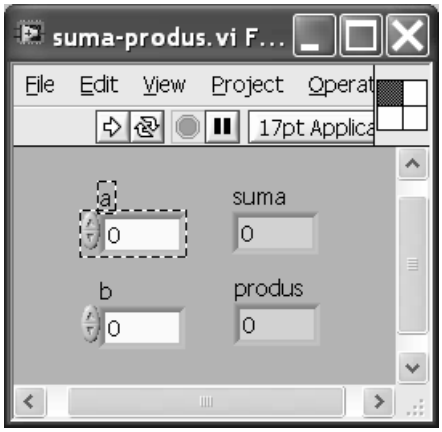


Figura 3.3

Observaţi că pătrăţelul a luat culoarea tipului de date vehiculate de controlul cu care a fost legat (în cazul nostru, portocaliu, pentru date numerice de tip dublă precizie).

10. Faceţi acelaşi lucru şi cu celelalte 3 terminale: *b* –stânga jos, *suma* – dreapta sus, *produs* – dreapta jos.

Editarea pictogramei

11. Faceţi MS pe pictogramă şi selectaţi *Show Icon*. Acum pictograma are un desen implicit la care se schimbă doar numărul din desen, care reprezintă al câtelea IV nou a fost deschis în sesiunea de lucru.
12. Faceţi din nou MS şi selectaţi *Edit Icon*. Se deschide un editor grafic rudimentar, care permite editarea imaginii pictogramei. Suprafaţa de editare este de 32 x 32 de pixeli. Se recomandă realizarea unui desen cât mai sugestiv,

care să indice utilizatorului funcționalitatea subIV-ului. Propunem editarea pictogramei cu desenul din figura 3.4.

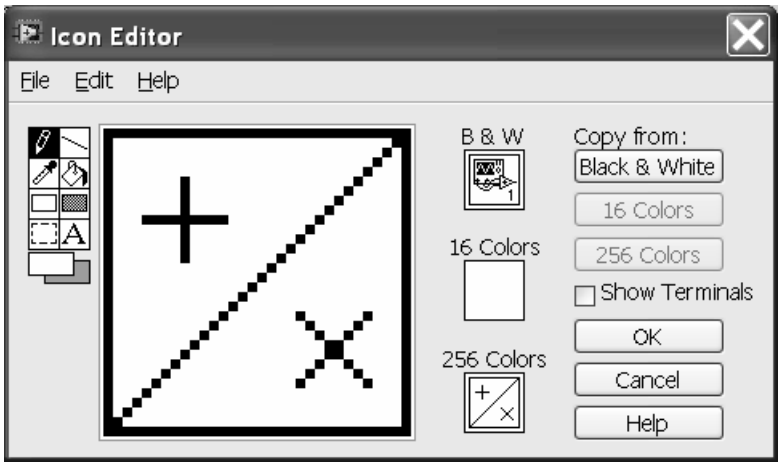


Figura 3.4

Deschiderea editorului pictogramei se poate face şi printr-un dublu click direct pe pictogramă.

Deşi chenarul, ca şi întregul desen implicit poate fi şters, se recomandă totuşi păstrarea unui chenar, deoarece la plasarea pe DL a subIV-ului, marginile pictogramei se estompează şi nu mai poate fi bine distins subIV-ul de fundalul alb al DL.

Editarea desenului pictogramei se mai poate face prin importarea unei imagini în format *.jpg* sau *.bmp*. Importarea se face prin tragerea cu M a imaginii din utilitarul *Explore* peste pictogramă. LabVIEW redimensionează automat imaginea la 32 x 32 pixeli.

13. Salvați din nou IV-ul.

Apelarea subIV-ului

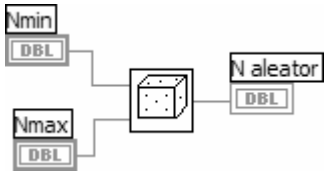
Un subIV odată creat şi salvat, el poate fi apelat ca oricare funcție din paleta de funcții, subpaleta *Select a VI…*

14. Deschideți un nou IV.
15. Aduceți subIV-ul *Suma-produs.vi* pe DL din paleta de funcții submeniul *Select a VI…* şi plasați-l pe diagramă. Programul deschide fereastra de dialog în directorul implicit de salvare a IV-urilor.
16. Observați pe pictogramă existența terminalelor cu denumirile aceleaşi cu etichetele controalelor şi indicatoarelor care le-au creat.
17. Creați în mod automat controalele şi indicatoarele corespunzătoare terminalelor prin *MD pe pictogramă – Create – Control(Indicator).*

18. Verificați funcționalitatea IV-ului.
19. Salvați IV-ul principal sub numele *Ex 3.1*.

Exerciții propuse

EP 3.1. Să se realizeze un subIV care să genereze la ieşire numere aleatoare cuprinse între două numere date, *Nmin* şi *Nmax* considerate ca intrări.



EP 3.2. Să se construiască un subIV care să calculeze modulul şi argumentul unui număr complex dat prin coordonatele sale carteziene (partea reală - a şi partea imaginară - b).

EP 3.3. Să se construiască un subIV care să calculeze câtul a două numere complexe.

Capitolul 4

OPERAȚII CU VECTORI, MATRICI ŞI CLUSTERE

Definirea matricilor pe panoul frontal

Matricea reuneşte într-o structură organizată, date de acelaşi tip. Parametrii definitorii ai unei matrici sunt dimensiunea şi elementele componente. O matrice cu o singură dimensiune se numeşte *vector*. O matrice poate avea maximum 2^{31}-1 elemente pe o dimensiune. Elementele matricii pot fi: numere, booleene, şiruri de caractere, clustere, căi şi forme de undă. Nu se admit matrici de matrici. Totuşi pot exista matrici de clustere unde elementele clusterului sunt tot matrici. Matricile sunt foarte utile la colectarea datelor obținute din achiziții de forme de undă şi la obținerea rezultatelor din bucle, unde la fiecare rulare a buclei se generează câte un element al matricii.

In limba engleză, cuvântul *array* desemnează în general o matrice cu n dimensiuni, în timp ce cuvântul *matrix* se referă la o matrice cu doar două dimensiuni. In limba română vom utiliza cuvântul *matrice* în general, ce înglobează atât *array* cât şi *matrix.*

Elementele unei matrici sunt ordonate. Fiecare element al matricii se accesează prin index. Indexul unei dimensiuni porneşte întotdeauna de la 0, care este primul element al dimensiunii şi merge până la *n-1*, unde *n* este numărul de element ale matricii. Pentru crearea unui control sau indicator de tip matrice, se accesează în paleta de controale subpaleta *Array, Matrix & Cluster.* De pe paletă, se plasează pe PF obiectul denumit *Array*. In acest moment avem pe PF cadrul matricii cu indexul, fără elemente. Implicit, dimensiunea este 1 (vector). Pentru adăugarea elementelor, se preia din paleta de controale tipul de element care ne interesează, de exemplu un numeric, după care se depune în controlul *array* în zona pătratului gri de lângă index. Veți observa că la depunere, pătratul va fi încadrat de punctele de redimensionare sau va fi încadrat într-un dreptunghi punctat (figura 4.1). După depunere, controlul rămâne de un gri estompat, semn că matricea nu are încă elemente definite, ci se cunoaşte numai tipul de dată ce caracterizează elementele. Atribuirea de valori elementelor matricii se stabileşte ca la un control obişnuit, element cu element.

Adăugarea unei noi dimensiuni unei matrici se poate face în următoarele două moduri:

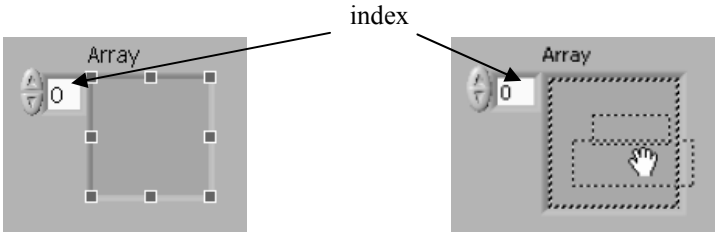


Figura 4.1

- se deschide meniul shortcut al indexului (MD pe index) şi se selectează *Add Dimension.*
- se apropie unealta 2) de index până când acesta apare încadrat între puncte de redimensionare, după care se ţine apăsat MS şi se trage în jos de punctul de la mijloc, latura de jos (figura 4.2).

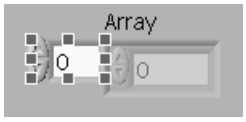


Figura 4.2

In figura 4.3 este prezentată o matrice cu două dimensiuni.

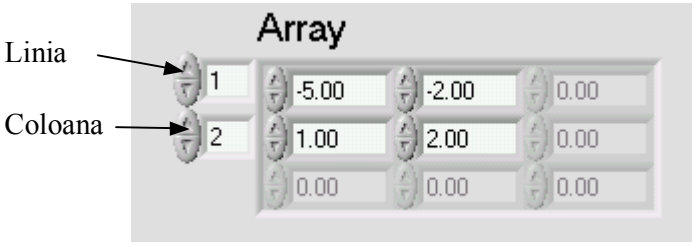


Figura 4.3

Primul index reprezintă numărul liniei pe care se află elementul afişat, iar cel de-al doilea index este numărul coloanei pe care se află elementul. Indexul porneşte întotdeauna de la 0. De exemplu, o matrice cu *m* linii şi *n* coloane va avea indexul ultimului element (*m-1, n-1*).

Pentru a vizualiza mai multe elemente odată pe PF, se apropie unealta 2) de marginea elementului până când apar punctele de redimensionare, după care se trage de punctul din mijloc de pe latura dreaptă (pentru desfăşurare pe orizontală) sau de punctul din mijloc de pe latura inferioară (pentru desfăşurare pe verticală) sau de colţul din dreapta jos (pentru desfăşurare în plan, la matrici cu două sau mai multe dimensiuni) Se pot astfel vizualiza oricâte linii şi coloane dorim şi ne

permite suprafața PF, inclusiv cele care nu au elemente definite. Indexul ne arată poziția elementului din stânga-sus. In exemplul din figura 4.3 este vorba despre elementul aflat pe linia cu index 1 şi coloana cu index 2.

In variantele mai noi ale LabVIEW, subpaleta de controale *Array, Matrix & Cluster* conține şi posibilitatea creării directe a unei matrici bidimensionale de numere reale sau complexe (*matrix*), la care elementele pot fi vizualizate şi utilizând bare de defilare (scrollbar).

Exercițiul 4.1

Scop

Crearea pe PF a controalelor şi indicatoarelor de tip matrice.

Mod de lucru

1. Deschideți un nou IV.
2. Plasați pe PF un control de tip *Array* din subpaleta *Array, Matrix & Cluster.*
3. Plasați în aria de definire a elementelor un control de tip *numeric*.
4. Desfăşurați pe orizontală, apoi pe verticală elementele vectorului.
5. Dați valori primelor 5 elemente ale vectorului astfel creat.
6. Manevrați indexul şi observați cum se modifică poziția primului element.
7. Treceți pe DL şi observați modelul terminalului de tip *Array*. Schimbați C în I şi invers.
8. Înlocuiți elementele de tip *numeric* cu elemente de tip *boolean, string, paleta de culori*. (Se utilizează opțiunea *Replace* din meniul shortcut. Paleta de culori este controlul *Framed color box* din subpaleta *Numeric*).
9. Reveniți la elemente de tip *numeric*.
10. Adăugați o nouă dimensiune vectorului.
11. Desfăşurați matricea cu 2 dimensiuni atât după linii, cât şi după coloane.
12. Dați valori primelor 5 elemente din linia a 2-a (linia cu index 1).
13. Manevrați indecşii şi observați poziționarea primului element afişat.
14. Adăugați o nouă dimensiune. Am obținut acum o matrice cu 3 dimensiuni, care seamănă cu o carte (cea de a treia dimensiune se numeşte filă).
15. Să se determine care dintre cei trei indecşi corespunde fiecărei dimensiuni.
16. Vidați matricea prin eliminarea tuturor elementelor din matrice cu opțiunea *Empty Array* din meniul shortcut deschis prin MD pe zona indecşilor, opțiunea *Data operations*.

Operații şi funcții cu matrici

Operațiile şi funcțiile cu matrici se găsesc în paleta de funcții, subpaleta *Array* (figura 4.4). Acestea sunt:

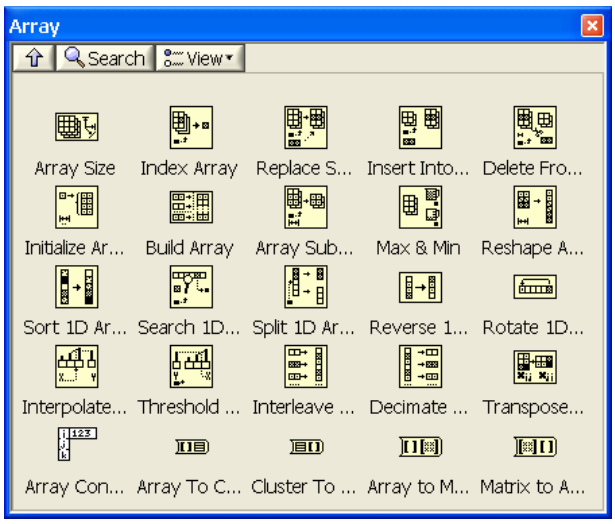


Figura 4.4

Tabelul 4.1

Nume funcție	Acțiune
Array Size	returnează numărul de elemente de pe fiecare dimensiune a unei matrici
Index Array	returnează valoarea elementului de la indexul specificat
Replace Array Subset	înlocuiește o submatrice a matricii inițiale cu o alta
Insert into Array	inserează un element sau o submatrice într-o matrice, începând de la un anumit index
Delete from Array	şterge un element sau o submatrice dintr-o matrice
Initialize Array	returnează o matrice n-dimensională în care fiecare element este inițializat cu o anumită valoare
Build Array	construiește o matrice pornind de la elementele sale
Array Subset	returnează o submatrice a matricii date pornind de la un anumit index
Array Max & Min	returnează valoarea elementului maxim şi cel minim dintr-o matrice, împreună cu indecşii corespunzători
Reshape Array	redimensionează matricea în funcție de numărul de elemente al fiecărei dimensiuni aplicat la intrare
Sort 1D Array	sortează elementele unui vector în ordinea ascendentă

Search 1D Array	caută un element într-un vector
Split 1D Array	divizează un vector în alți doi subvectori pornind de la un index
Reverse 1D Array	inversează ordinea elementelor unui vector
Rotate 1D Array	roteşte elementele unui vector
Interpolate 1D Array	returnează valoarea calculată prin interpolare dintre două elemente adiacente ale unui vector, corespunzătoare unui index fracționar. Interpolarea între elementele adiacente se face după o funcție liniară.
Threshold 1D Array	caută o pereche de elemente adiacente dintr-un vector astfel încât primul element să fie mai mic decât un prag dat şi al doilea element să fie mai mare decât pragul. Realizează interpolarea între cele două elemente şi returnează indexul fracționar căruia îi corespunde pragul.
Interleave 1D Arrays	creează un vector cu elementele întrețesute ale vectorilor de la intrare
Decimate 1D Array	divide un vector în subvectori cu elemente întrețesute
Transpose 2D Array	realizează transpusa unei matrici bidimensionale

Extragerea unei linii sau a unei coloane dintr-o matrice

Pentru extragerea unei linii sau a unei coloane dintr-o matrice se utilizează funcția *Index Array*, prin legarea la index doar a numărului liniei sau a coloanei care ne interesează, cealaltă intrare rămânând liberă. Dacă se leagă ambii indecşi, funcția returnează valoarea elementului de la indexul respectiv. Funcția *Index Array* este redimensionabilă, adică se pot efectua toate operațiile de mai sus pe aceeaşi funcție. Pentru aceasta, mergeți cu M pe funcție până apar punctele de redimensionare şi trageți de punctul din mijloc – latura de jos pentru adăugarea altor celule. In exemplul din figura 4.5, prima celulă furnizează elementele liniei cu indexul 1 (un vector), a doua celulă ne dă elementele coloanei cu indexul 2 (un vector), iar a treia celulă ne dă elementul de pe linia 4, coloana 6 (un scalar).

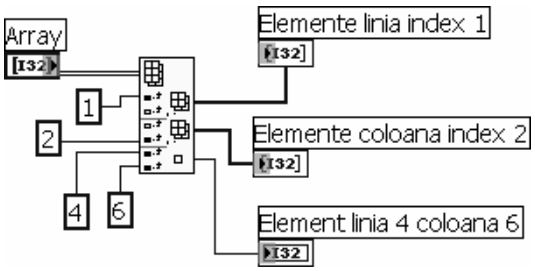


Figura 4.5

Dacă matricea de intrare are 3 dimensiuni, atunci fiecare celulă va avea 3 intrări (celula se redimensionează automat în funcție de dimensiunea matricii). Dacă se dă de exemplu valoarea 2 primei intrări, atunci funcția va furniza la ieşire o matrice bidimensională conținând elementele filei cu indexul 2 (figura 4.6).

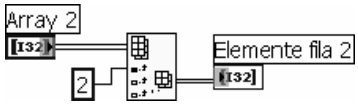


Figura 4.6

Particularități ale funcției *Build Array*

Cu ajutorul funcției *Build Array*, se pot construi matrici de oricâte dimensiuni pornind de la alte matrici sau elemente componente. Se pot adăuga intrări suplimentare prin opțiunea *Add Input* a meniului shortcut al unei intrări sau prin tragerea cu M a punctelor de redimensionare. Funcția *Build Array* prezintă următoarele particularități:

1) Dacă la intrarea unei funcții *Build Array* se leagă o matrice, la ieşire se obține o matrice cu o dimensiune în plus (figura 4.7).

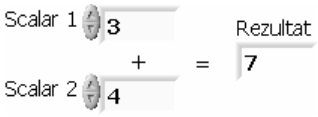
Figura 4.7

2) Dacă la intrările unei funcții *Build Array* se leagă mai multe matrici (vectori), sunt posibile următoarele:
   - dacă intrările sunt de aceeaşi dimensiune şi opțiunea *Concatenate Inputs* din meniul shortcut al funcției *Build Array* nu este validată, se obține la ieşire o matrice cu o dimensiune mai mult, la care intrările sunt considerate elemente (dacă se apropie unealta 4) de una din intrări, apare scris *Input element*).

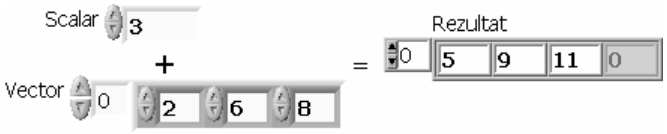
- dacă în meniul shortcut se specifică opțiunea *Concatenate Inputs*, se realizează o "alipire" sau concatenare a intrărilor, obținându-se o matrice cu aceeaşi dimensiune.

Polimorfism

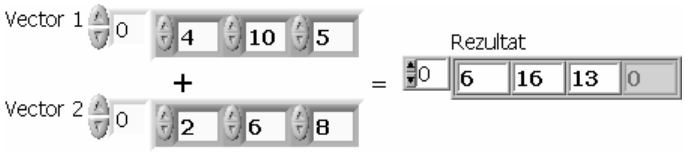
In LabVIEW, funcțiile numerice sunt polimorfe. Aceasta înseamnă că intrările în aceeaşi funcție numerică pot fi diferite ca structură. De exemplu, putem să adunăm un scalar cu un vector (matrice) sau doi vectori (matrici) între ele. Mai jos sunt date câteva situații de polimorfism.



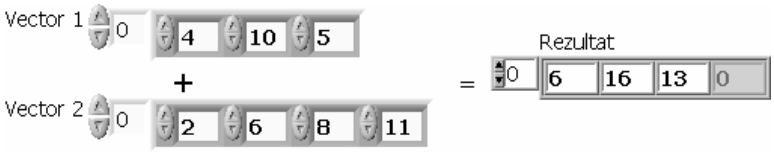
a) Scalar + scalar



b) Scalar + vector. Se adună scalarul la fiecare element al vectorului.



c) Vector + vector cu acelaşi număr de elemente. Se adună element cu element.



d) Vector + vector cu număr diferit de elemente. Se adună element cu element, însă rezultatul are numărul de elemente al vectorului mai mic.

Nu se pot aduna un vector cu o matrice bidimensională.

Inmulțirea a două matrici bidimensionale se face prin înmulțirea matricilor element cu element.

Exercițiul 4.2

Scop

Experimentarea funcțiilor cu matrici din subpaleta de funcții *Array*..

Mod de lucru

1. Deschideți un nou IV.
2. Construiți pe PF un vector numeric cu 5 elemente şi o matrice numerică cu 3 linii şi 4 coloane la care elementele au valori arbitrare.
3. Deschideți şi fixați paleta de funcții pe DL.
4. Citiți helpul mic al fiecărei funcții.
5. Utilizând ca intrări cele două matrici de pe PF şi indicatoare adecvate, experimentați următoarele funcții: *Array size, Index array, Insert into array, Initialize array, Array subset, Array Max & Min, Reshape Array, Sort 1D array, Search 1D array, Interleave 1D array, Decimate 1D array*.

Exercițiul 4.3

Scop

Experimentarea funcțiilor *Build Array* şi *Index Array*.

Mod de lucru

a)

1. Deschideți un nou IV.
2. Construiți pe PF doi vectori numerici, unul cu 4 elemente arbitrare iar celălalt cu 3 elemente.
3. Adăugați pe DL funcția *Build Array* şi legați la intrarea funcției primul vector.
4. Creați automat indicatorul de la ieşirea funcției.
5. Rulați instrumentul şi observați tipul de dată creat şi elementele componente.

b)

6. Adăugați o nouă funcție *Build Array* pe care o extindeți pentru două intrări.
7. Legați vectorii la cele două intrări.
8. Creați automat indicatorul de la ieşirea funcției.
9. Rulați instrumentul şi observați ieşirea. S-a obținut o matrice bidimensională la care fiecare linie este câte un vector de la intrare. Vectorul care are mai puține elemente se completează cu zerouri până la numărul de elemente al celui mai mare.

c)

10. Adăugați o nouă funcție *Build Array* căreia îi legați cei doi vectori la intrare.

11. Deschideți meniul shortcut al uneia din intrări şi bifați opțiunea *Concatenate Inputs*.
12. Creați automat indicatorul de la ieşire şi rulați instrumentul.
13. Observați că în acest caz ieşirea este tot un vector, obținut prin concatenarea celor doi.

d)

14. Adăugați funcția *Index Array* şi legați-i la intrare matricea bidimensională din cazul b).
15. Configurați funcția pentru extragerea elementelor primei linii, a celei de a doua coloane şi a elementului (1,3).
16. Legați la ieşiri indicatoare adecvate.
17. Rulați instrumentul şi observați rezultatele.
18. Salvați instrumentul sub numele *Ex 4.3.vi*.

Exercițiul 4.4

Scop

Experimentarea funcțiilor *Interpolate 1D Array* şi *Threshold 1D Array*.

Mod de lucru

Funcția ***Interpolate 1D Array*** calculează o valoare intermediară între două elemente adiacente ale unui vector corespunzător unui index ipotetic fracționar, după metoda interpolării liniare. De exemplu, în figura 4.8, indexului fracționar 1,8 al vectorului {1; 2,4; 3,2; 5; 5,4}, îi corespunde valoarea interpolată 3,04.

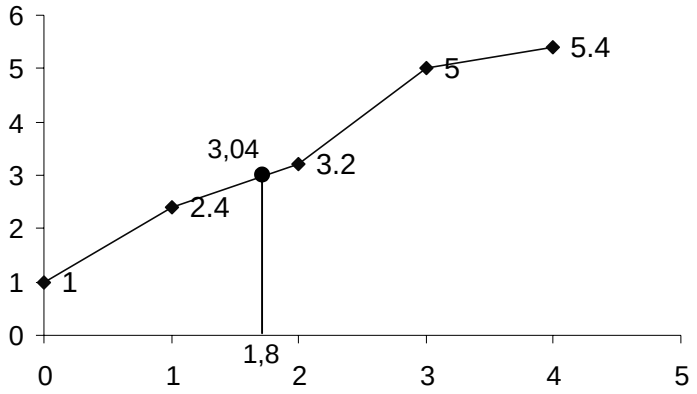


Figura 4.8

1. Deschideți un nou IV.

2. Construiţi vectorul cu elementele de mai sus.
3. Plasaţi pe DL funcţia *Interpolate 1D Array* din subpaleta *Array*.
4. Citiţi helpul funcţiei.
5. Legaţi vectorul la intrare şi creaţi automat un control pentru indexul fracţionar şi indicatorul de ieşire.
6. Rulaţi instrumentul dând diverse valori indexului fracţionar.

Funcţia ***Threshold 1D Array*** caută între două elemente adiacente ale unui vector indexul fracţionar corespunzător unui număr dat (threshold), cu condiţia ca numărul să fie mai mare decât primul element al perechii şi mai mic decât cel de-al doilea element. Dacă sunt mai multe perechi de elemente în vector îndeplinind condiţia de mai sus, este necesar să se stabilească un index de start de unde funcţia începe căutarea. Dacă se găseşte o pereche la care numărul este mai mic decât primul element şi mai mare decât cel de-al doilea, funcţia returnează valoarea indexului de start.

7. Plasaţi pe DL funcţia *Threshold 1D Array*.
8. Citiţi helpul funcţiei.
9. Legaţi la intrare vectorul construit, iar la intrarea *Threshold y* legaţi ieşirea funcţiei *Interpolate 1D Array*.
10. Creaţi un indicator la ieşire.
11. Rulaţi instrumentul şi daţi diverse valori indexului fracţionar de la intrarea funcţiei *Interpolate 1D Array*.
12. Daţi un index de valoare mai mare decât indexul maxim al vectorului. Observaţi rezultatele celor două funcţii.
13. Salvaţi instrumentul sub numele *Ex 4.4.vi*.

Definirea clusterelor pe panoul frontal

Clusterul este un tip de dată care reuneşte în aceeaşi structură elemente de tipuri diferite, comparativ cu matricea, care reunea într-o structură ordonată elemente de acelaşi tip. In cluster pot fi introduse orice tipuri de date. Toate trebuie însă să provină fie de la controale, fie de la indicatoare. Aşadar, clusterele sunt fie de tip „control”, fie de tip „indicator”. Atunci când se plasează un control sau un indicator într-un cadru de cluster, toate celelalte obiecte devin controale, respectiv indicatoare. Clusterul în LabVIEW este echivalent structurii (*struct*) din C. Pe DL datele conţinute într-un cluster se transportă printr-un singur fir. La destinaţie, datele se extrag din cluster, toate, sau numai cele necesare.

Clusterele se utilizează pentru:

- transportul datelor diferite pe distanţe mari pe DL utilizând un singur traseu de date.
- reducerea numărului de intrări şi/sau ieşiri dintr-un subVI.

Crearea unui cluster pe PF se face din submeniul *Array, Matrix & Cluster* de pe paleta de controale. La aducerea unui obiect de tip *Cluster* pe PF de pe paleta de controale, acesta este reprezentat printr-un cadru, care poate fi redimensionabil. In acest cadru se aduc, de pe paleta de controale, elementele ce vor fi conținute în cluster (figura 4.9).

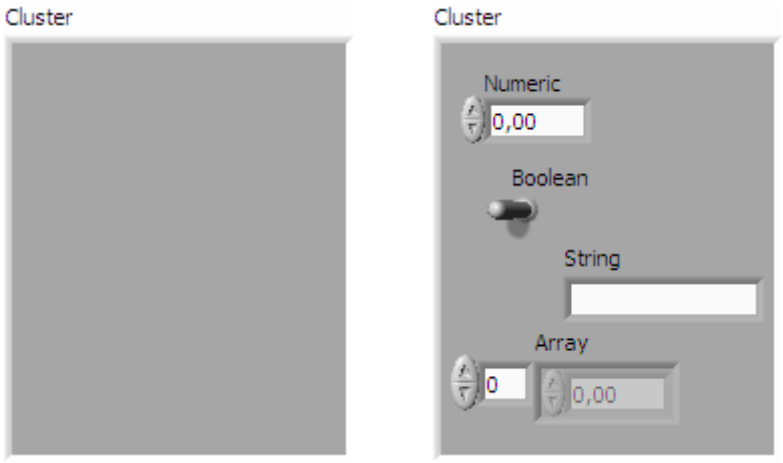


Figura 4.9

Meniul shortcut al clusterului se accesează prin MD exact pe marginea cadrului ce delimitează clusterul. Acesta oferă posibilitatea aranjării spațiale a obiectelor conținute în cadru prin opțiunea *Autosizing* cu una din cele 4 variante.

Elementele din cluster prezintă o ordine logică, dată de succesiunea cu care acestea au fost aduse în cadrul clusterului. Tot aceasta este şi ordinea în care clusterul va fi desfăcut în elemente componente cu ajutorul funcției *Unbundle*, pe DL. Se poate face reordonarea obiectelor în cluster utilizând opțiunea *Reorder Controls in Cluster* din meniul shortcut.

Operații şi funcții cu clustere

După cum am văzut mai sus, pentru ca informația să poată fi mai uşor transportată în interiorul unui IV sau între IV-uri, datele se împachetează în clustere, deoarece clusterul este tratat ca un singur obiect, mult mai uşor de manipulat. Utilizarea informației din cluster se poate face însă doar după ce acesta a fost despachetat. Funcțiile cu care se realizează despachetarea clusterului sunt: *Unbundle* şi *Unbundle by Name*, care se găsesc în paleta de funcții, subpaleta *Cluster & Variant.*

Funcția *Unbundle* realizează despachetarea tuturor elementelor din cluster, iar *Unbundle by Name* extrage din cluster doar elementul specificat prin etichetă. Pentru identificarea informației, la aplicarea clusterului funcției *Unbundle,* aceasta

se organizează după etichetele controalelor (indicatoarelor) care au contribuit la formarea clusterului, sau a denumirii terminalelor funcţiilor de la care provin datele. Funcţia *Unbundle by Name* permite specificarea din lista etichetelor a unuia sau a mai multor elemente ale clusterului. Numai pentru aceste elemente computerul alocă memorie la despachetare, comparativ cu funcţia *Unbundle* pentru care se alocă memorie pentru toatele elementele clusterului.

Reformarea clusterului sau realizarea unui cluster nou pe DL se face din elementele componente cu funcţiile *Bundle* şi *Bundle by Name*.

Exerciţiul 4.5

Scop

Experimentarea funcţiilor de manipulare a clusterelor.

Mod de lucru

1. Deschideţi un nou IV.
2. Plasaţi pe PF un cluster care să conţină următoarele elemente: un control numeric, două controale booleene şi un control de tip vector numeric, ca în figura 4.10.

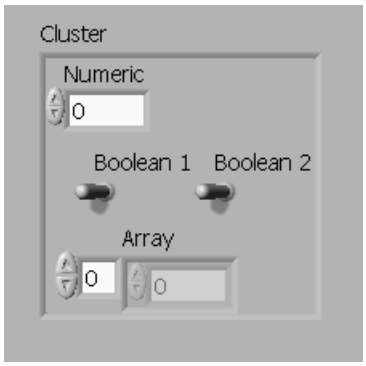


Figura 4.10

3. Reordonaţi elementele clusterului după următoarea formulă: **0** – Array, **1** – Boolean 1, **2** – Boolean 2, **3** – Numeric. Pentru aceasta, deschideţi meniul shortcut al clusterului (MD pe chenarul cadrului) şi selectaţi *Reorder Controls in Cluster*. Observaţi că PF se înnegreşte, iar fiecărui element al clusterului i se asociază o pereche de numere: numărul pe fond negru este noul număr de ordine, iar cel de pe fond alb este vechiul număr de ordine (figura 4.11). Numărul de ordine ce urmează a fi alocat unui element este cel scris în căsuţa din dreptul textului *Click to set to…*Alocarea acestui număr se face prin click MS pe elementul dorit, după care se trece automat la următorul număr, până când se alocă numere de ordine tuturor elementelor. La sfârşit se apasă butonul *Confirm*.

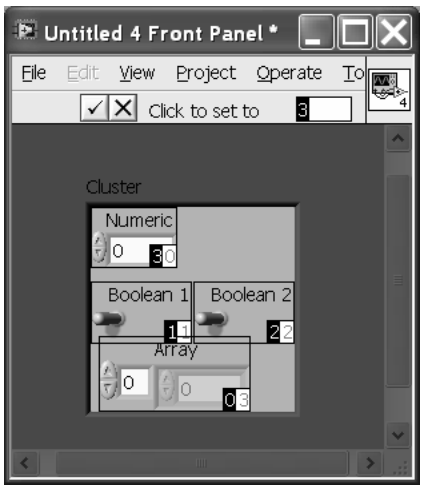


Figura 4.11

4. Schimbați controlul *Numeric* în indicator. Observați cum toate celelalte controale se transformă în indicatoare.
5. Transformați din nou clusterul în control.
6. Plasați pe DL o funcție *Unbundle* din paleta de funcții *Cluster & Variant*.
7. Legați clusterul la intrarea funcției *Unbubdle*. Observați cum în momentul legării, funcția se autodimensionează după câte elemente ale clusterul. De asemenea, ieşirile capătă culoarea tipului de date vehiculat. Ordinea de desfacere este cea stabilită la punctul 3 (figura 4.12 I).
8. Plasați o funcție *Bundle*, căreia îi legați la intrare *Array* şi *Boolean 2*. Creați automat la ieşire indicatorul corespunzător.
9. Creați la ieşirea funcției *Unbundle* corespunzătoare numericului, indicatorul potrivit.
10. Treceți pe PF, porniți instrumentul de la butonul *Run Continuously* şi manevrați controalele clusterului urmărind indicatoarele create.
11. Plasați o funcție *Unbundle by Name*, căreia îi legați clusterul la intrare. Veți observa că, implicit, ieşirea va fi cea corespunzătoare elementului cu numărul de ordine 0 din cluster (*Array* în cazul nostru).
12. Dorim acum să despachetăm doar pe *Boolean 1*. Pentru aceasta, deschideți meniul shortcut al ieşirii funcției *Unbundle by Name* şi, în opțiunea *Select Item,* bifați *Boolean 1* (figura 4.12 II).
13. Creați un indicator şi rulați instrumentul.
14. Daca se doreşte modificarea programatică a unuia sau mai multor elemente din cluster, se poate utiliza funcția *Bundle* în configurația din figura 4.12 III sau funcția *Bundle by Name* în configurația din figura 4.12 IV. In acest caz, *Numeric 2* modifică doar elementul *Numeric* din indicatoarele *output cluster 2* şi *output cluster 3*, suspendând controlul *Numeric* din clusterul inițial.

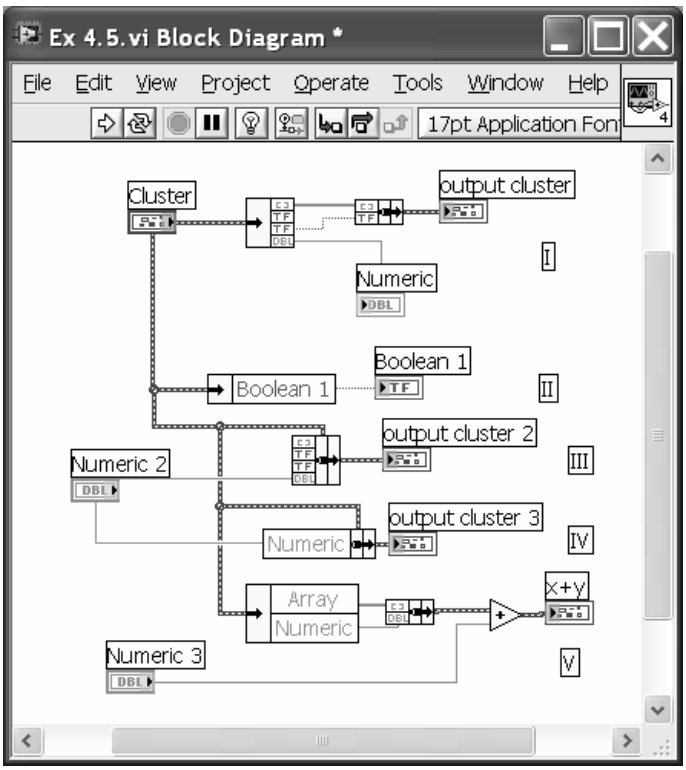


Figura 4.12

15. Construiți un nou cluster format numai din elementele *Array* şi *Numeric*, utilizând funcția *Bundle* (figura 4.12 V).
16. Plasați un operator de adunare şi legați una din intrări la noul cluster, iar pe cealaltă la un control scalar (*Numeric 3*). Creați un indicator automat la ieşire.
17. Porniți instrumentul şi dați diverse valori lui *Numeric 3*, observând ce se întâmplă cu valorile elementelor controlului cluster $x + y$.

După cum se observă, putem efectua operații între un scalar şi un cluster numeric. Se pot de asemenea efectua operații între o matrice şi un cluster numeric, ce conține scalari sau alte matrici, ca şi între două clustere numerice. Ca şi la matrici, aici se aplică *polimorfismul*, concept care permite adaptarea intrărilor unui operator sau ale unei funcții la tipul datelor aplicate la intrare.

Clustere de erori

Oricât de atent s-ar lucra şi oricât de experimentat ar fi proiectantul, nu se pot prevedea toate problemele care ar putea apărea în construcția unui instrument virtual. Dacă nu ar fi prevăzut un mecanism de verificare a erorilor, nu am putea şti decât că instrumentul nu funcționează, dar nu şi de ce. Sistemul de verificare a

erorilor disponibil în LabVIEW poate spune de ce şi unde apare o eroare. Acest sistem se adresează în special operațiilor de intrare/ieşire (I/O), cum ar fi: lucrul cu fişierele, comunicații cu instrumentele (serial, paralel, Ethernet), achiziții de date, etc. Sistemul de verificare a erorilor ne poate ajuta să identificăm următoarele probleme:

- inițializarea incorectă a comunicației sau scrierea improprie pe un dispozitiv extern;
- dispozitivul extern este defect sau nu lucrează corect;
- probleme cu sistemul de operare sau cu driverele.

Manipularea erorilor

Implicit, LabVIEW sesizează automat orice eroare care apare la rularea unui IV prin suspendarea execuției acestuia, identificarea subIV-ului sau funcției unde s-a produs eroarea şi afişarea unei ferestre de dialog. De exemplu, dacă o funcție de I/O depăşeşte limita de aşteptare, e posibil să nu se dorească oprirea instrumentului, ci reluarea ciclului de aşteptare. Acest lucru poate fi decis în diagrama instrumentului.

Funcțiile şi instrumentele din bibliotecile LabVIEW returnează erori în două moduri:

- sub forma unor coduri de eroare (specific funcțiilor);
- sub forma unor clustere (specific subIV-urilor).

Manipularea erorilor de către LabVIEW respectă principiul fluxului de date, în sensul că informațiile despre erori circulă de-a lungul lanțului de prelucrare, în paralel cu datele. Pe parcursul execuției, LabVIEW verifică existența erorilor în fiecare nod al diagramei. Dacă se detectează o eroare la un nod, execuția acestuia se suspendă, însă informația despre eroare (tipul erorii şi locul unde s-a produs) se propagă în continuare spre celelalte noduri din lanțul de prelucrare, până la final. Pentru aflarea acestor informații, se plasează după ultimul nod un indicator de tipul *Simple Error Handler*, care se găseşte în subpaleta de funcții *Dialog & User Interface*. In figura 4.13 este prezentat modul de propagare a informației despre erori într-un lanț de salvare în fişiere.

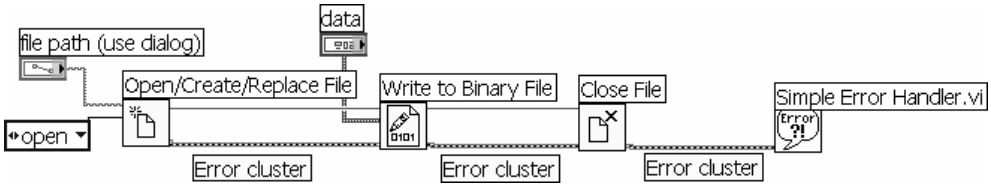
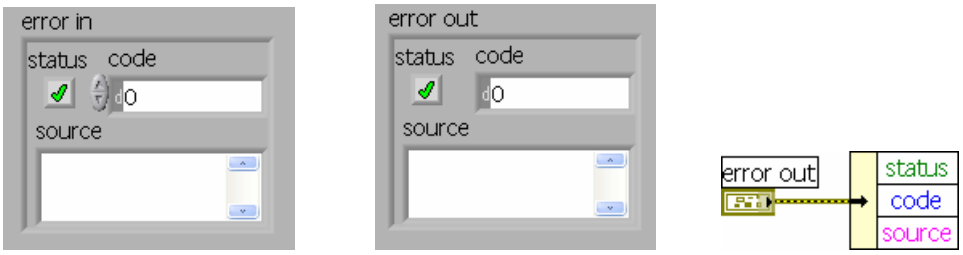


Figura 4.13

In figura 4.14 este dată componența unui cluster de erori. Acesta conține următoarele informații:



a) Control cluster de erori b) Indicator cluster de erori c) Elementele componente

Figura 4.14

- **status** este un boolean care devine *true* la apariția unei erori. Valoarea acestui boolean poate fi folosită pentru oprirea instrumentului într-o buclă WHILE.
- **code** este un întreg pe 32 de biți, care identifică eroarea printr-un număr. In absența vreunei erori, codul este zero. Dacă acest număr este diferit de zero, dar valoarea lui **status** este *false*, nu este o eroare propriu-zisă ci un avertisment.
- **source** este un text care identifică nodul unde s-a produs eroarea.

In cazul apariției unei erori, pe lângă informațiile date în clusterul de erori, e posibilă explicarea acesteia prin accesarea opțiunii *Explain Error* din meniul *Help*. Un cluster de erori poate fi legat direct la terminalul de condiționare al unei bucle WHILE în scopul opririi acesteia la apariția unei erori. Doar componenta **status** este luată în considerare de către terminal, aceasta având valoarea unui boolean. In acest caz, *Stop if True* şi *Continue if True* se schimbă în *Stop on Error* şi *Continue while Error*.

Capitolul 5

STRUCTURI

Structurile în LabVIEW sunt reprezentări grafice ale instrucţiunilor de ciclare şi condiţionare din limbajele bazate pe text. Ca orice alt nod, structurile prezintă terminale prin care acestea se interconectează cu alte noduri din diagrama instrumentului, sunt executate automat când toate datele sunt disponibile la intrări şi furnizează la ieşire rezultatul când execuţia structurii este completă.

Orice structură este formată dintr-un cadru redimensionabil ce conţine blocul de coduri de executat, în concordanţă cu regulile structurii. Secţiunea de coduri din interiorul unei structuri este denumită *subdiagramă*. Locurile prin care datele intră sau ies dintr-o structură se numesc *tuneluri*.

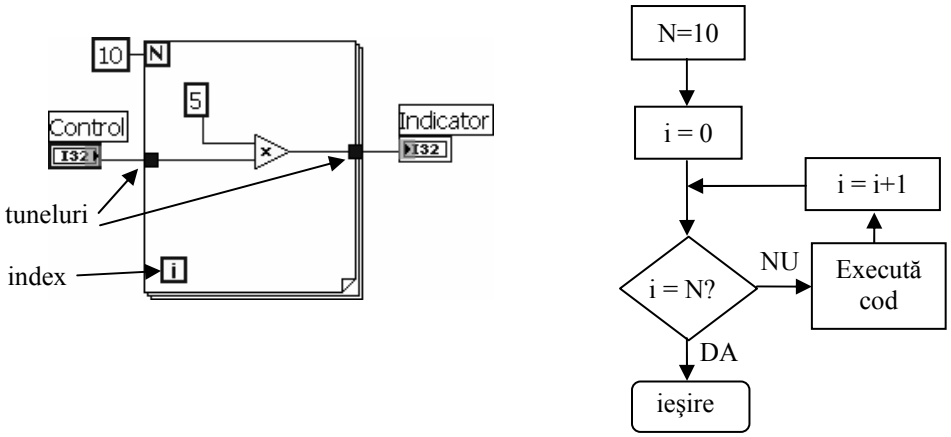
Structurile pot fi accesate din paleta de funcţii, submeniul *Structures*. Există 5 tipuri de structuri de bază:

1. Bucla FOR
2. Bucla WHILE
3. Structura CASE
4. Structura Secvenţă (SEQUENCE)
5. Nod de formule (FORMULA NODE)

Bucla FOR

Bucla FOR este o structură care execută codurile din interior de un anumit număr de ori, cât este valoarea numărului legat la terminalul N. In figura 5.1 sunt date a) reprezentarea buclei FOR în LabVIEW şi b) organigrama acesteia. Aducerea unei bucle FOR pe DL se face prin preluarea acesteia din subpaleta *Structures* şi dimensionarea corespunzătoare porţiunii de diagramă care se doreşte a fi inclusă în buclă. Se poate şi aduce întâi bucla pe DL, după care se construieşte în interior subdiagrama de executat. Terminalul i se numeşte *terminal de iteraţie*. Acesta ne dă în fiecare moment numărul iteraţiei curente. Prima iteraţie este întotdeauna 0. Atât *N* cât şi *i* sunt numere întregi de tip I32; aşadar, valoarea maximă a lui *N* poate fi 2^{31}-1. Dacă la *N* se leagă un număr real, LabVIEW constrânge acest număr la tip

I32 şi eventual îl limitează la 2^{31}-1. Dacă la *N* se leagă numărul 0, atunci bucla nu execută nici o iteraţie. Tunelurile, atât cel de intrare cât şi cel de ieşire, sunt sub forma unor dreptunghiuri pline de culoarea tipului de date vehiculat, daca intrarea sau ieşirea sunt scalari, sau sub forma unui dreptunghi în care se află două paranteze drepte, dacă intrarea sau ieşirea sunt vectori sau matrici. In acest caz avem de-a face cu *autoindexarea*, care va fi tratată în secţiunea următoare.



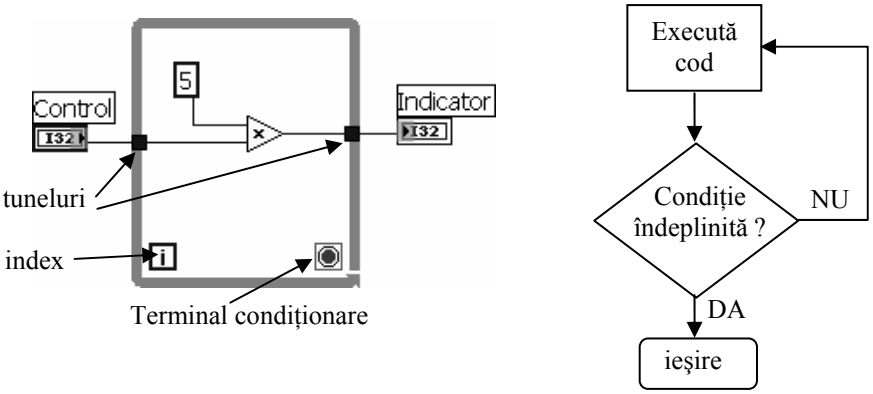
a) b)

Figura 5.1

Bucla WHILE

Bucla WHILE este o structură care execută o subdiagramă atâta timp cât este îndeplinită o condiţie. In figura 5.2 sunt date reprezentarea buclei WHILE în LabVIEW, respectiv diagrama structurii. Bucla WHILE este echivalentă instrucţiunii *do…while* din programarea convenţională bazată pe text. Din diagramă se observă că o buclă WHILE se execută cel puţin o dată, deoarece testarea condiţiei se face după execuţia codului. Condiţia poate fi îndeplinită dacă terminalul de condiţionare este TRUE (*Stop if TRUE*) sau FALSE (*Continue if TRUE*). Acest lucru poate fi schimbat din meniul shortcut al terminalului. Dacă terminalul nu este legat, LabVIEW dă un mesaj de eroare.

Terminalul de iteraţie *i* şi tunelurile au aceeaşi semnificaţie ca şi la bucla FOR. Condiţia de execuţie a buclei trebuie legată în interiorul acesteia, pentru a putea fi citită la fiecare rulare. Exemplul din figura 5.3 înseamnă "ciclare infinită", deoarece constanta TRUE este citită o dată înainte de a intra în buclă, după care bucla ciclează continuu întrucât condiţia nu mai este citită, aceasta rămânând continuu TRUE. Oprirea buclei se face din butonul *Abort Execution* de pe bară.



a) b)

Figura 5.2

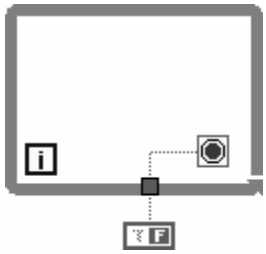


Figura 5.3

Este recomandabil ca toate instrumentele ce se construiesc să fie încadrate în întregime într-o buclă WHILE prevăzută la terminalul de condiționare cu un buton de STOP. Este sigur astfel că toate sarcinile începute în cadrul buclei sunt duse la sfârşit. Oprirea instrumentului trebuie să se facă obligatoriu de la acest buton. Dacă oprirea instrumentului se face de la butonul de stop de urgență (*Abort Execution*) din bară, pot exista sarcini neterminate cum ar fi: deschiderea şi neînchiderea unor fişiere, a unor legături cu rețeaua (serverul), alocarea unor resurse hardware fără a mai fi dezalocate, etc. In exemplul din figura 5.4, bucla va cicla de 3 ori sau până când se apasă butonul STOP.

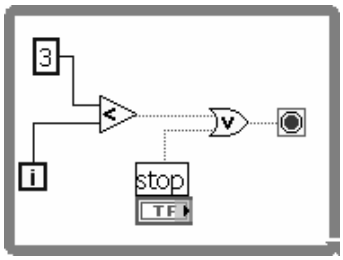


Figura 5.4

Autoindexarea

Autoindexarea este o noțiune specifică buclelor FOR şi WHILE. Dacă la intrarea unei bucle se aplică o matrice şi din meniul shortcut al tunelului de intrare se validează opțiunea *Enable Indexing*, atunci la fiecare iterație bucla ia în calcul, în ordine, câte un element din matricea respectivă. In exemplul din figura 5.5, la intrare avem un vector cu 6 elemente, este validată autoindexarea, iar la terminalul *N* este legat numărul 5.

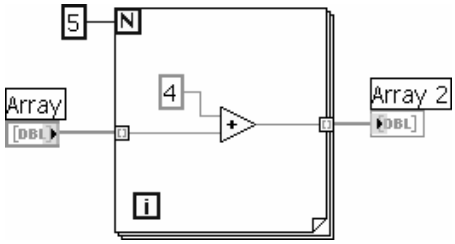


Figura 5.5

La prima iterație, bucla adună la elementul cu index 0 al vectorului numărul 4 şi depune rezultatul într-un vector la ieşire. La a doua iterație se adună cu 4 elementul cu index 1 al vectorului, ş.a.m.d. Dacă este validată autoindexarea tunelului de la ieşire, rezultatul va fi un vector care conține rezultatele fiecărei iterații. Bucla ciclează doar de 5 ori, chiar dacă vectorul are 6 elemente, deoarece terminalul *N* are valoare mai mică. In general, dacă la intrare sunt mai mulți vectori şi este validată autoindexarea, bucla va cicla de atâtea ori cât este minimul dintre numărul de elemente al vectorilor şi valoarea lui *N*. Deci:

$$\text{Număr rulări} = \min\{\text{size}(V1), \text{size}(V2), \ldots, N\} \qquad (5.1)$$

Dacă nu se validează autoindexarea la intrare, atunci bucla va cicla de 5 ori şi de fiecare dată va efectua adunarea întregului vector cu numărul 4.

Dacă nu se validează autoindexarea la ieşire, atunci rezultatul furnizat de buclă va fi cel al ultimei iterații.

` Dacă la intrare se află o matrice bidimensională, atunci la prima iterație se ia in calcul linia 0, la a doua iterație linia 1, ş.a.m.d. Dacă, din contra, se validează opțiunea *Disable Indexing*, atunci la fiecare iterație se ia în calcul toată matricea.

In cazul buclei WHILE, autoindexarea vectorului de intrare se realizează şi după terminarea elementelor acestuia, bucla oprindu-se doar atunci când nu mai este îndeplinită condiția de ciclare. După terminarea elementelor vectorului, se ia în calcul mai departe până la oprirea buclei valoarea implicită a ultimului element.

Regiştrii de deplasare (shift registers)

Regiştrii de deplasare (RD) sunt perechi de terminale ce se utilizează în buclele FOR şi WHILE pentru transferarea valorilor calculate de la o iterație la alta. RD sunt similari cu variabilele statice din programarea structurală.

Crearea unui RD se face din meniul shortcut al buclei (MD pe marginea structurii), selectare *Add Shift Register*. La terminalul din dreapta, RD stochează valoarea corespunzătoare iterației actuale, iar la terminalul din stânga este accesibilă valoarea corespunzătoare iterației imediat anterioare. RD se poate redimensiona în partea stângă pentru obținerea şi a valorilor altor iterații anterioare. Redimensionarea se face din meniul RD – *Add Element*, sau analog cu redimensionarea matricilor (figura 5.6).

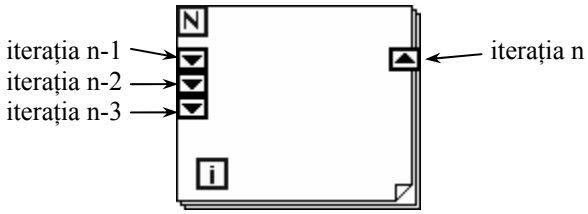


Figura 5.6

RD trebuie inițializați pe fiecare terminal din partea stângă. Aceasta se face prin legarea la aceste terminale a câte unei valori (constantă sau C) corespunzătoare tipului de date vehiculat de RD. Dacă RD nu se inițializează, atunci la prima iterație a buclei la pornirea instrumentului se ia în calcul valoarea stocată în RD la ultima rulare a buclei când instrumentul a fost oprit. Dacă nu s-au mai făcut rulări anterioare ale instrumentului, valoarea inițială va fi valoarea implicită a tipului de date vehiculate de RD.

IV-ul din figura 5.7 calculează media alunecătoare a şirului de numere generat de funcția *Random Number (0-1)*. Registrul de deplasare este inițializat cu 0.

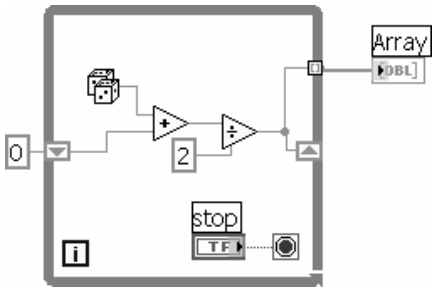


Figura 5.7

Structura CASE (Caz)

Această structură se utilizează pentru efectuarea unor subdiagrame numai în anumite cazuri. Este formată din mai multe subdiagrame suprapuse, fiecare din ele reprezentând un caz. La un moment dat este vizibil doar unul din cazuri (figura 5.8). Structura CASE este echivalentă instrucțiunii *if…then…else* din programarea text.

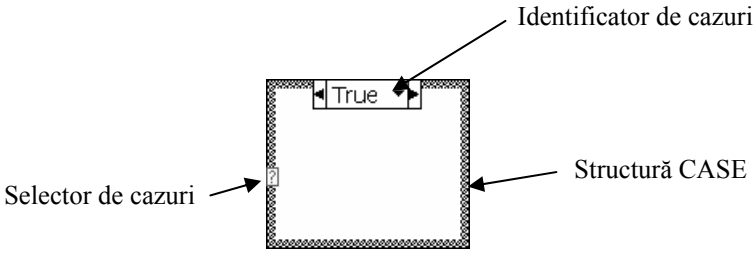


Figura 5.8

O structură CASE conține două elemente definitorii: *selectorul de cazuri* şi *identificatorul de cazuri*.

***Selectorul de cazuri* (SC)** este un terminal de al cărui valoare depinde care caz se execută la un moment dat. La acest terminal se pot lega date de tip boolean, numeric de tip întreg, şiruri de caractere sau controale de tip enumerare.

***Identificatorul de cazuri* (IC)** este o etichetă specifică fiecărui caz în parte, ce trebuie să aibă corespondent strict între valorile selectorului de cazuri. De ex., dacă la SC se leagă un boolean, etichetele IC vor fi *True* şi *False*. Dacă la SC se leagă un numeric, etichetele IC vor fi o parte din numerele de la SC. Dacă la SC se leagă şiruri de caractere, eticheta cazului va fi chiar şirul de caractere care-l comandă, scris între ghilimele.

Dacă la SC se leagă mai multe valori decât numărul de cazuri din structură, este necesară stabilirea unui caz implicit (default). De ex. dacă la IC avem cazurile 0, 1 şi 2 unde 0 este *default* şi de la SC se aplică 3, atunci structura execută cazul implicit, 0.

Etichetele din IC pot fi date şi sub formă relațională. De ex., ..10 reprezintă eticheta unui caz care se execută pentru toate numerele până la 10. 12.. reprezintă eticheta unui caz care se execută pentru toate numerele mai mari decât 12.

Pentru vizualizarea diagramei unui caz, se apasă pe săgețile stânga-dreapta ale IC pentru defilare între cazuri succesive sau se face click pe IC şi se alege din listă cazul dorit.

Tunelurile de intrare sunt terminale de intrare în CASE. Pot exista tuneluri de intrare utilizate de unele cazuri şi de altele nu.

Tunelurile de ieşire trebuie să primească date de la toate cadrele CASE din structură. Dacă există cel puţin un cadru care nu furnizează date tunelului, acesta este reprezentat printr-un dreptunghi alb şi programul dă eroare de sintaxă. In momentul în care tunelul de ieşire primeşte date de la toate cadrele, acesta capătă culoarea tipului de date pe care le transportă.
Se pot adăuga sau şterge cazuri din meniul shortcut al structurii.

Structura SEQUENCE (Secvență)

Această structură se foloseşte atunci când se doreşte execuția programului într-o anumită ordine preferențială, alta decât *fluxul de date* oferit de LabVIEW. Structura SECVENȚĂ se prezintă sub două forme: secvență plată (*flat sequence*) şi secvență suprapusă (*stacked sequence*). **Secvența plată**, exemplificată în figura 5.9, forțează executarea codurilor cadru cu cadru, de la stânga spre dreapta. In exemplul din figura 5.9 se va executa întâi operația de adunare, după care cea de înmulțire, însă rezultatele vor fi disponibile ambele în acelaşi timp, în cadrul al doilea. Trecerea dintr-un cadru în altul se face prin tuneluri. Se pot adăuga şi alte cadre din meniul shortcut, opțiunea *Add Frame After (Before)*. Secvența plată este utilă atunci când se doreşte efectuarea unor coduri într-o anumită ordine, dar şi urmărirea desfăşurată a acestor coduri. Dezavantajul secvențelor plate este că ocupă spațiu mare pe diagramă.

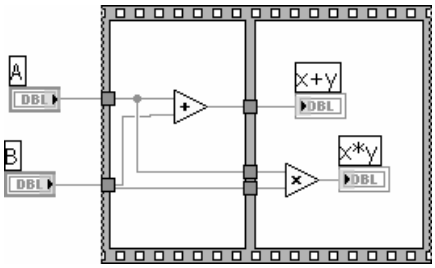
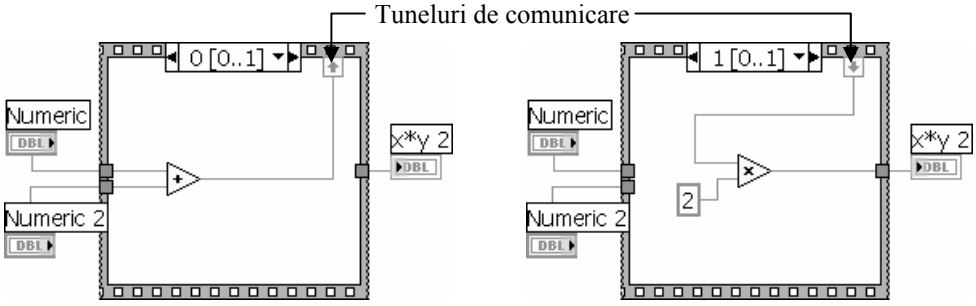


Figura 5.9

Secvența suprapusă dispune cadrele precum cărțile de joc, unul peste altul, în ordinea execuției. La un moment dat este vizibilă subdiagrama doar dintr-un singur cadru. Trecerea datelor dintr-un cadru în altul se poate face numai în sensul execuției secvenței, prin ***tuneluri de comunicare***, care se obțin din meniul shortcut, opțiunea *Add Sequence Local*. Săgeata tunelului specifică direcția datelor.

In figura 5.10 sunt prezentate două cadre succesive ale aceleiaşi secvențe suprapuse. In primul cadru se realizează adunarea a două numere, iar în cadrul al doilea rezultatul adunării, transportat din cadrul 1 prin tunelul de comunicare, este înmulțit cu 2. Rezultatul este disponibil numai după efectuarea întregii secvențe. Nici o informație dintr-o structură SECVENȚĂ nu este accesibilă decât după ce

structura a fost complet executată. Secvențele suprapuse sunt mai compacte decât cele plate, fiind utile atunci când spațiul alocat pe diagramă este limitat.



a) b)

Figura 5.10

Tunelurile de ieşire dintr-o structură SECVENȚĂ pot avea doar o singură sursă de date, spre deosebire de structura CASE, unde fiecare caz trebuie să trimită date tunelului. Ieşiri pot fi create din orice secvență, dar datele sunt accesibile la acea ieşire numai după terminarea de executat şi a ultimei secvențe din structură.

Nod de formule (Formula Node)

Nodul de formule este un tip de structură bazată pe limbajul text convențional cu care se realizează diverse operații în diagrama bloc a instrumentului sau chiar se rulează secțiuni de program. Cu această structură nu este necesară apelarea altor programe sau aplicații şi nici a operatorilor aritmetici din paleta de funcții. In cadrul acestei structuri se pot introduce sub formă de text expresii matematice folosind operatorii din limbajele text convenționale, dar se acceptă şi cuvinte specifice instrucțiunilor de ciclare şi condiționare din limbajul C (do, if, while, for, case). Nodurile de formule acceptă expresii formate din oricâte variabile. Instrucțiunile din nodurile de formule trebuie separate prin semnul „ ; ”. Comentariile se inserează la fel ca în C, utilizând semnele /* şi */.

Structurile de tip nod de formule sunt utile la calculul expresiilor ce conțin multe variabile sau sunt foarte complexe.

In figura 5.11 este prezentat un exemplu de calcul al funcției discrete:

$$z = 4 + A\sin(2\pi i / N) \tag{5.2}$$

Bucla FOR ciclează de atâtea ori cât este N , care reprezintă numărul de puncte în care este eşantionată sinusoida. i este indexul buclei, care joacă rol aici de variabilă discretă. Ieşirea din buclă se face cu autoindexare; vectorul obținut reprezintă chiar sinusoida digitizată.

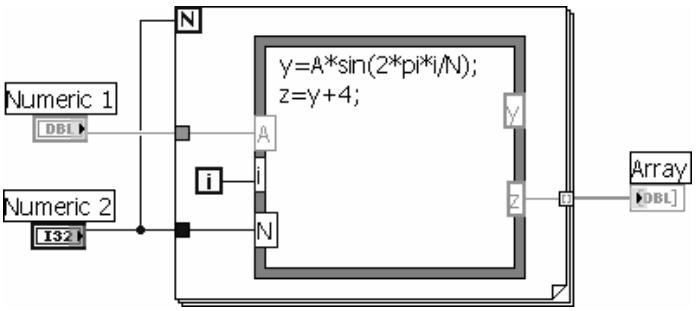


Figura 5.11

In interiorul unei structuri nod de formule sunt acceptate şi secvenţe de program în limbaj text. In figura 5.12 este dat un exemplu de astfel de cod, care are drept scop generarea a 100 de numere aleatoare..

```
int32;
float64 y(100);

for (i=1; i<100; i++)
y(i)=rand();

/*acest exemplu este un
generator de 100 numere
aleatoare*/
```

Figura 5.12

Reguli pentru variabile în noduri de formule

- Variabilele pot fi definite în structură (ca în figura 5.12) sau pot fi specificate prin intrări şi ieşiri.
- Adăugarea de intrări şi/sau ieşiri se face din meniul shortcut al structurii, opţiunea *Add Input (Output)*.
- Toate variabilele, inclusiv cele intermediare (cum este y din figura 5.11) trebuie definite. Variabilele intermediare se definesc ca ieşiri.
- Numele intrărilor şi ieşirilor trebuie să fie aceleaşi cu numele variabilelor pe care le definesc. Pot exista cuvinte cheie predefinite cum ar fi „pi”, care nu trebuie declarate.
- Pot exista intrări şi ieşiri cu acelaşi nume, dar nu pot exista două intrări (ieşiri) cu acelaşi nume.
- Intrările trebuie să aibă legate obligatoriu surse de date. Eticheta controlului de intrare nu trebuie să fie aceeaşi cu numele variabilei căreia îi furnizează date.
- Variabilele pot fi scalari reali (floating point), întregi sau matrici de numere.
- Variabilele nu pot avea unităţi de măsură.

- Lista sintaxei funcţiilor şi operatorilor utilizaţi în nodurile de formule se obţine prin apelarea helpului structurii, din meniul shortcut al acesteia (MD pe marginea cadrului), opţiunea *Help*.

Alte funcţii privind lucrul cu nodurile de formule se găsesc în paleta de funcţii, *Mathematics – Scripts & Formulas*..

Mai există o posibilitate de evaluare a unei expresii scrise în mod text, utilizând funcţia *Expression Node* din subpaleta *Numeric*. Această funcţie permite evaluarea unei expresii de o singură variabilă, scrisă după aceleaşi sintaxă ca şi la *Formula Node*. Variabila se notează cu *x* şi ia valorile sursei de date aplicate la intrare. In figura 5.13 este dat un exemplu de utilizare a funcţiei *Expression Node*.

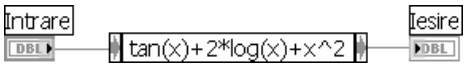


Figura 5.13

Exerciţiul 5.1

Scop

Studiul autoindexării.

Mod de lucru

1. Deschideţi un nou IV.
2. Plasaţi pe PF doi vectori numerici de tip control, unul conţinând 5 elemente, iar celălalt 3 elemente numere reale oarecare.
3. Plasaţi pe PF un vector numeric de tip indicator.
4. Construiţi diagrama din figura 5.14. Toate tunelurile vor avea validată autoindexarea.

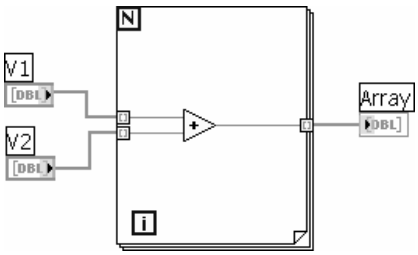


Figura 5.14

5. Pe DL apăsaţi butonul *Highlight Execution*.
6. Rulaţi instrumentul şi observaţi câte iteraţii se execută. IV-ul va executa 3 iteraţii, câte elemente are vectorul mai mic. Vectorul de ieşire conţine rezultatul obţinut prin adunarea, element cu element, a celor doi vectori de la intrare.

7. Invalidați autoindexarea tunelurilor de la intrare. De ce s-a întrerupt legătura la ieşire?
8. Invalidați şi autoindexarea tunelului de la ieşire.
9. Încercați să rulați instrumentul. Interpretați mesajul de eroare afişat.
10. Dați valoarea 4 terminalului *N* al buclei FOR.
11. Rulați acum instrumentul şi observați numărul de iterații şi rezultatul de la ieşire.
12. Legați ieşirea operatorului *Add* la un nou tunel, căruia îi validați autoindexarea.
13. Construiți automat indicatorul de ieşire. Ce tip de date reprezintă?
14. Rulați instrumentul şi interpretați rezultatul.
15. Validați din nou autoindexarea vectorului cu mai multe elemente.
16. Rulați IV-ul şi observați numărul de iterații efectuate şi rezultatul obținut. Se vor executa 4 rulări deoarece valoarea legată la *N* este mai mică decât numărul de elemente al vectorului autoindexat. Rezultatul este o matrice cu 4 linii şi 3 coloane, fiecare linie fiind obținută prin adunarea primelor 4 elemente ale vectorului mai mare cu elementele vectorului mai mic. Vectorul de la ieşire conține elementele ultimei linii a matricii.
17. Validați acum autoindexarea vectorului mai mic şi invalidați autoindexarea vectorului mai mare.
18. Rulați instrumentul şi interpretați rezultatele.
19. Salvați IV-ul sub numele *Ex 5.1.vi*

Exercițiul 5.2

Enunț

Să se construiască, utilizând buclele FOR şi WHILE, un vector care să conțină, în ordine, toate numerele naturale de la 1 la 100.

Mod de lucru

Ideea rezolvării acestei probleme este să se folosească indexul buclelor.

Utilizarea buclei FOR

1. Deschideți un nou IV.
2. Plasați o buclă FOR pe DL.
3. Legați la terminalul *N* numărul de elemente ale vectorului, în cazul nostru 100.
4. Deoarece indexul *i* porneşte de la 0, vom adăuga acestuia operatorul +1 (*increment*), care se găseşte în subpaleta de funcții numerice.
5. Scoateți ieşirea operatorului +1 din buclă prin autoindexare şi legați-o la un vector indicator.
6. Rulați instrumentul şi observați rezultatul. Verificați dacă indexului 99 al vectorului de ieşire îi corespunde numărul 100.

Utilizarea buclei WHILE

1. Deschideți un nou IV.
2. Plasați o buclă WHILE pe DL.
3. Legați operatorul *increment* la indexul buclei şi scoateți-i ieşirea prin autoindexare, la fel ca la bucla FOR.
4. Vom stabili acum condiția de oprire a buclei. Pentru aceasta vom testa la fiecare iterație dacă i+1 ≤ 100. Rezultatul îl legăm la terminalul de condiționare al buclei. Terminalul trebuie să funcționeze pe *Continue if True*.

Diagramele celor două instrumente sunt date în figurile 5.15 a) şi b).

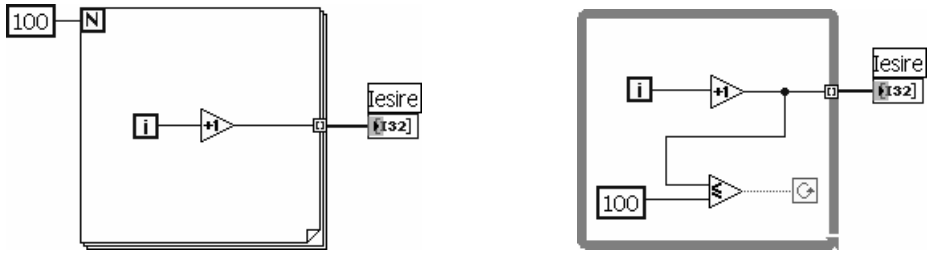


Figura 5.15

5. Salvați cele două IV-uri sub numele *Ex 5.2a.vi* şi *Ex 5.2b.vi*

Exerciții propuse

EP 5.1.
Să se construiască un vector care să conțină, în ordine, toate numerele întregi cuprinse între două numere date pe panoul frontal.

Exercițiul 5.3

Enunț

Să se construiască un IV care să genereze continuu numere întregi aleatoare cuprinse între -400 şi 600, până când se generează un număr care coincide cu un număr întreg dat pe PF. Să se afişeze pe PF valoarea numărului curent şi câte numere s-au generat până la coincidență.

Mod de lucru

Organigrama problemei este prezentată în figura 5.16.

1. Deschideți un nou IV.
2. Plasați pe PF un control numeric etichetat „Număr dat”, un indicator numeric etichetat „Număr curent” şi un indicator etichetat „Număr iterații”.
3. Pe DL plasați o buclă WHILE.
4. Plasați în buclă generatorul de numere aleatoare sub forma funcției *Random Number 0 – 1*, pe care o găsiți în subpaleta de funcții numerice.
5. Plasați operatorii de înmulțire şi scădere şi realizați legăturile conform organigramei.
6. Rotunjiți numărul obținut la cel mai apropiat întreg utilizând funcția *Round to Nearest* din subpaleta de funcții numerice.
7. Realizați comparația cu numărul dat cu o funcție din subpaleta *Comparison*.
8. Aplicați rezultatul comparației la terminalul de condiționare al buclei.
9. Scoateți valoarea lui *i* la ieşire printr-un operator *Increment* şi afişați rezultatul.

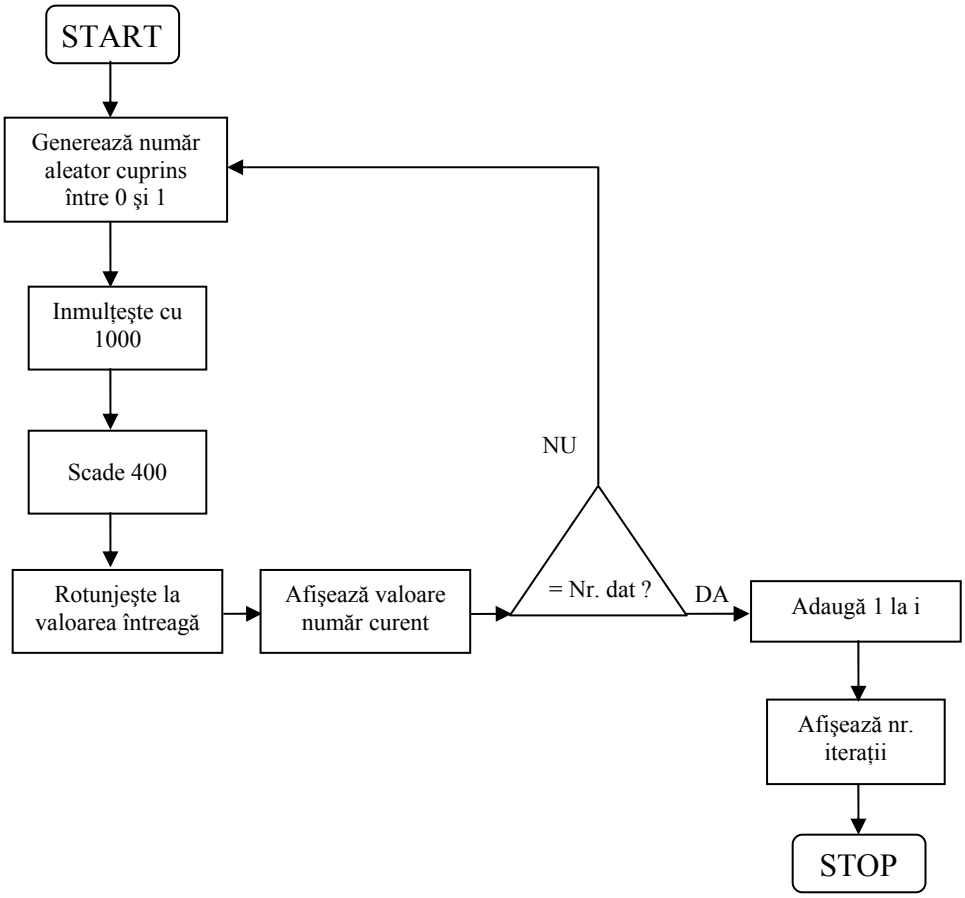


Figura 5.16

10. Încercați să rulați IV-ul. Dacă nu funcționează, observați eventualele erori şi depanați-l.
    In figura 5.17 este dată o posibilă soluție a problemei.

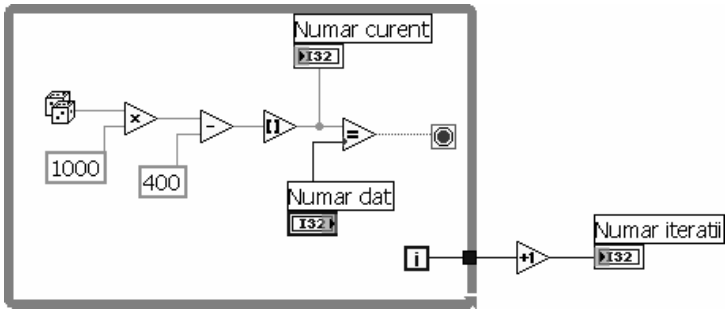


Figura 5.17

11. Salvați IV-ul sub numele *Ex 5.3.vi*.

Exercițiul 5.4

Enunț

Să se construiască un vector care să fie format din primele N elemente ale unei progresii aritmetice, fiind date de pe PF primul termen, *a0*, numărul de termeni, *N* şi rația *r*.

Mod de lucru

Progresia aritmetică este un şir de numere ai cărui termeni se obțin prin relația de recurență:

$$a_n = a_{n-1} + r \tag{5.3}$$

unde a_{n-1} şi a_n sunt doi termeni consecutivi, iar *r* este rația. Pentru inițierea iterației este necesar primul termen, a_0.

In acest exercițiu vom experimenta utilizarea regiştrilor de deplasare pentru impelentarea relațiilor de recurență în LabVIEW.

1. Plasați pe un nou IV un control numeric de tip double etichetat *a0*, un control de tip double etichetat *r*, un control de tip I32 etichetat *N* şi un indicator de tip array double etichetat *Progresie*.
2. Plasați pe DL o buclă FOR căreia îi legați terminalului *N* controlul cu eticheta *N*.

3. Adăugați un registru de deplasare din meniul shortcut al buclei, opțiunea *Add Shift Regsiteri.* Terminalul din stânga al registrului reprezintă termenul a_{n-1} din relația 5.3, iar terminalul din dreapta, termenul a_n.
4. Adăugați operatorul *Add* în buclă.
5. Legați la una din intrări controlul *r*, iar la cealaltă intrare terminalul din stânga al registrului de deplasare.
6. Legați ieşirea sumatorului la terminalul din dreapta al registrului.
7. Inițializați registrul de deplasare prin legarea controlului *a0* la terminalul din stânga, în afara buclei.
8. Scoateți rezultatul la ieşirea buclei prin autoindexare şi legați-l la indicatorul *Progresie*. Întrucât progresia trebuie să conțină primul termen, *a0*, este necesar ca rezultatul să fie preluat de pe firul ce leagă terminalul din stânga al registrului.
9. Plasați toată diagrama într-o buclă WHILE.
10. Creați automat un control de STOP din meniul shortcut al terminalului de condiționare opțiunea *Create Control*.
11. In figura este prezentată diagrama acestui IV.
12. Dați valori celor 3 controale şi rulați instrumentul. Verificați rezultatul pentru primii 5 termeni.
13. Salvați sub numele *Ex. 5.4.vi*.

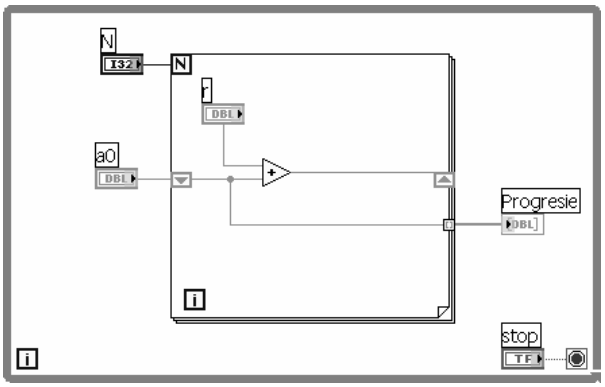


Figura 5.18

Exercițiul 5.5

Enunț

Să se construiască un IV care să afişeze primele N elemente ale unei progresii aritmetice sau geometrice, şi să calculeze suma sau produsul acestor termeni. Selectarea tipului progresiei şi tipului operației se face de pe panoul frontal. Se mai stabilesc pe PF primul termen al progresiei, *a0*, numărul de termeni, *N* şi rația *r*.

Mod de lucru

In acest exercițiu vom experimenta utilizarea structurii CASE pentru luare de decizii.

1. Deschideți un nou IV.
2. Plasați pe PF controalele *a0, N* şi *r*, precum şi indicatorul *Progresie*, la fel ca la exemplul 5.4.
3. Plasați pe PF un indicator numeric *double* etichetat *Rezultat*.
4. Plasați pe PF două controale de tip *Text ring* din subpaleta de controale *Ring & Enum*.
5. Editați controlul *Tip progresie* în modul următor:
   1. Din meniul shortcut al controlului, selectați *Edit Items*…
   2. In fereastra deschisă, editați textul *Aritmetică* pentru valoarea 0 şi *Geometrică* pentru valoarea 1, ca în figura 5.19. Pentru adăugarea unei noi linii, apăsați *Insert*.
   3. Apăsați OK.
   4. Redimensionați controlul după lungimea textului din interior.
6. După modelul de la punctul 5, editați controlul *Tip operație* cu cele două opțiuni, *Suma* şi *Produs*.

Cele două controale *Ring* sunt numerice de tip U16, după cum se poate observa de pe DL. Fiecărui element text îi corespunde un număr întreg. Pentru a vedea corespondența dintre elementul text şi numărul asociat, selectați din meniul shortcut *Visible Items – Digital display*.

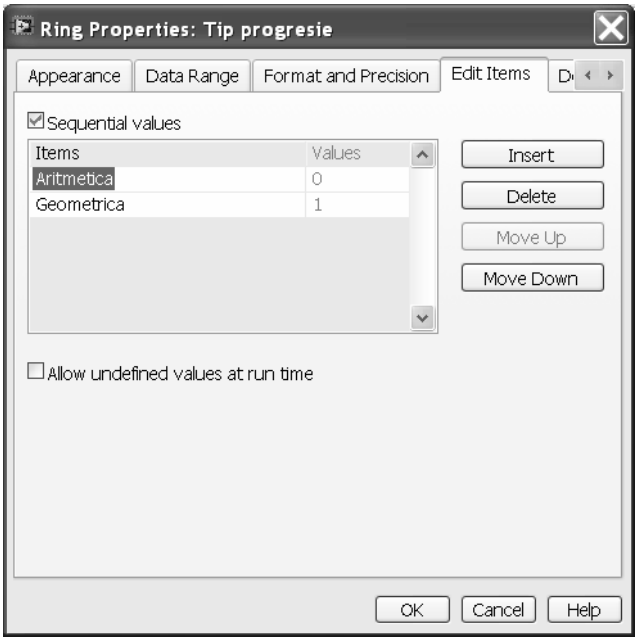


Figura 5.19

7. Realizaţi schema de construire a progresiei aritmetice, aceeaşi ca la exerciţiul 5.4.
   Vom completa acum diagrama pentru selectarea tipului progresiei. Vom folosi pentru aceasta o structură CASE. In primul cadru vom plasa operatorul de adunare, iar în cel de-al doilea, operatorul de înmulţire.
8. Din paleta de funcţii selectaţi o structură de tip CASE şi încadraţi în această structură operatorul *Add*.
9. Legaţi controlul *Tip progresie* la selectorul structurii. Observaţi cum eticheta de identificare a structurii CASE se schimbă din valoarea implicită „True” în „1”. Aşadar, pentru valoarea selectorului 1 (cazul 1), se va crea progresia aritmetică, ceea ce este în contradicţie cu textul elementului de pe selector (*Geometrică* este corespunzătoare valorii 1 a selectorului). Pentru a corecta acest lucru, în meniul shortcut al structurii CASE selectaţi *Make This Case 0, Default*. Cadrul cazului 0 trebuie să arate ca în figura 5.20.

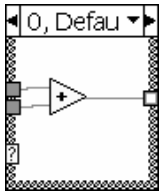


Figura 5.20

10. Treceţi pe cazul 1 şi plasaţi un operator *Multiply* în cadru. Realizaţi legăturile intrărilor şi ieşirii cu tunelurile corespunzătoare, aceleaşi ca la cazul 0.
11. Incadraţi toată diagrama într-o bulcă WHILE.
12. Creaţi automat un control de STOP din meniul shortcut al terminalului de condiţionare opţiunea *Create Control*.
13. Treceţi pe PF. Daţi valori lui N, a0 şi r şi rulaţi instrumentul. Manevraţi controlul ring şi verificaţi crearea celor două tipuri de progresii.
14. Adăugaţi o nouă structură CASE pe care o plasaţi după indicatorul *Progresie*, în interiorul structurii WHILE.
15. Legaţi controlul *Tip operatie* la selectorul structurii.
16. Plasaţi în cadrul 0 funcţia *Add Array Elements*, iar în cadrul 1 *Multiply Array Elements* din subpaleta de funcţii numerice.
17. Legaţi rezultatul *Progresie* la intrarea celor două funcţii; legaţi ieşirea acestora la indicatorul *Rezultat*.
18. Rulaţi instrumentul şi verificaţi funcţionarea celor două selectoare.
19. Înlocuiţi controalele de tip *Text Ring* cu controale de tip *Meniu Ring* şi *Enum*, pe care le găsiţi în subpaleta de controale *Ring & Enum*. Experimentaţi funcţionarea acestor controale.
    Diagrama instrumentului este prezentată în figura 5.21.

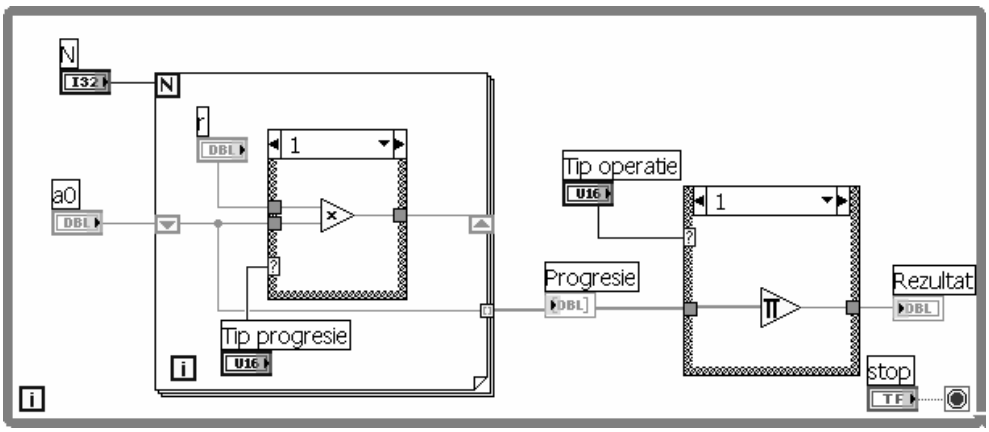


Figura 5.21

Exercițiul 5.6

Enunț

Să se construiască un IV care să implementeze un filtru cu răspuns infinit la impuls având ecuația cu diferențe următoarea relație recursivă:

$$y(n) = 1{,}3x(n) - x(n-1) + x(n-2) + 0{,}5y(n-1) - 1{,}1y(n-3) \qquad (5.4)$$

Să se determine răspunsul acestui filtru la un semnal de intrare format din 100 numere aleatoare cuprinse între -2 şi 2.
Semnalul de intrare se consideră cauzal, adică x(n) = 0 pentru $\forall$ n ≤ 0.

Mod de lucru

1. Deschideți un IV nou.
2. Plasați pe PF un control de tip vector numeric etichetat *x(n)* şi un indicator de tip vector numeric etichetat *y(n)*.
3. Vom genera pentru început semnalul de intrare, x(n). Pentru obținerea şirului de numere aleatoare cuprinse între -2 şi 2, vom utiliza generatorul de numere aleatoare *Random Number (0-1)* la care vom efectua următoarele operații:

$$\begin{aligned} &0 < x < 1 \;\mid\; \cdot 4 \\ &0 < x < 4 \;\mid\; -2 \\ &-2 < x < 2 \end{aligned} \qquad (5.5)$$

4. Realizați generatorul semnalului de intrare după diagrama din figura 5.22.

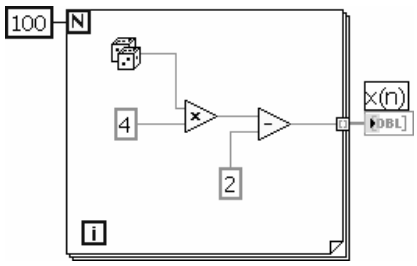


Figura 5.22

5. Pentru implementarea filtrului, avem nevoie de doi regiştri de deplasare: unul care furnizează valorile x(n-1) şi x(n-2), iar altul care ne dă valorile y(n-1) şi y(n-3). Implementarea relaţiei (5.4) se poate face în două moduri: prin realizarea operaţiilor utilizând operatori din subpaleta *Numeric* sau utilizând structura *Formula Node*. Cele două soluţii sunt prezentate în figurile 5.23 şi 5.24. Iniţializarea regiştrilor se face cu valoarea 0, în virtutea cauzalităţii semnalelor vehiculate.
   Se observă că soluţia ce utilizează *Formula Node* este mult mai convenabilă deoarece modificarea ecuaţiei cu diferenţe a filtrului se poate face foarte simplu, prin simpla modificare a relaţiei din cadrul formulei.

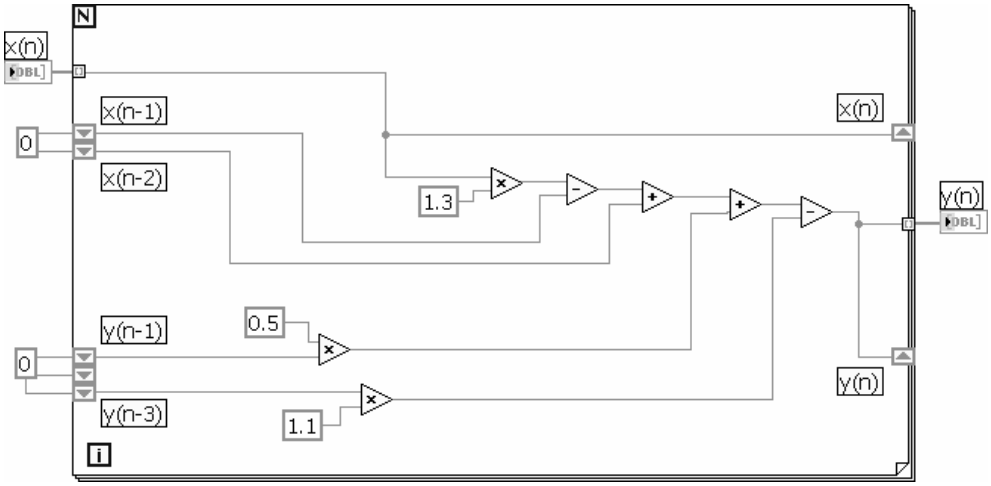


Figura 5.23

6. Construiţi cele două diagrame în două IV-uri diferite, pe care le salvaţi *Ex 5.6a.vi*, respectiv *Ex 5.6b.vi*.
7. Verificaţi funcţionarea lor.
8. Modificaţi limitele de variaţie a semnalului de intrare şi verificaţi din nou funcţionalitatea instrumentului.

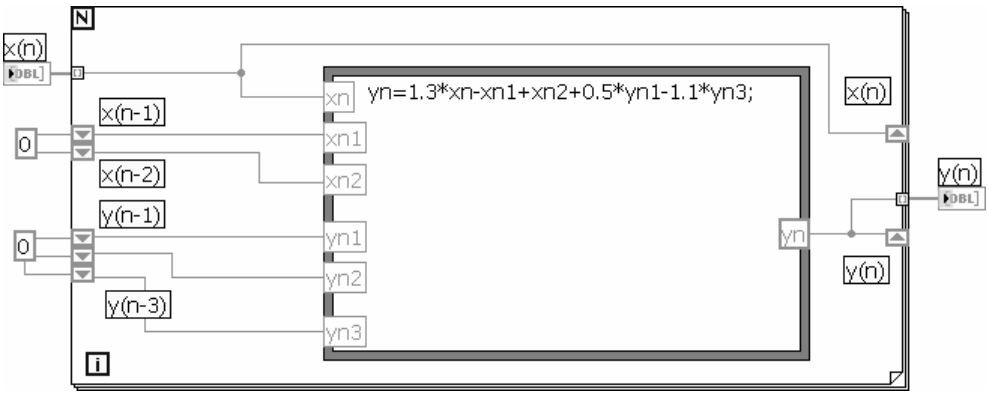


Figura 5.24

Exercițiul 5.7

Enunț

Să se construiască un vector care să fie format din N numere aleatoare cuprinse între două numere date, Nmin şi Nmax, fixate pe PF. Să se construiască apoi alți doi vectori, unul care să conțină toate numerele primului vector, mai mici decât un număr N1 din intervalul [Nmin, Nmax], iar celălalt numerele mai mari decât N1. Să se afişeze pe PF vectorul inițial şi cei doi vectori sortați.

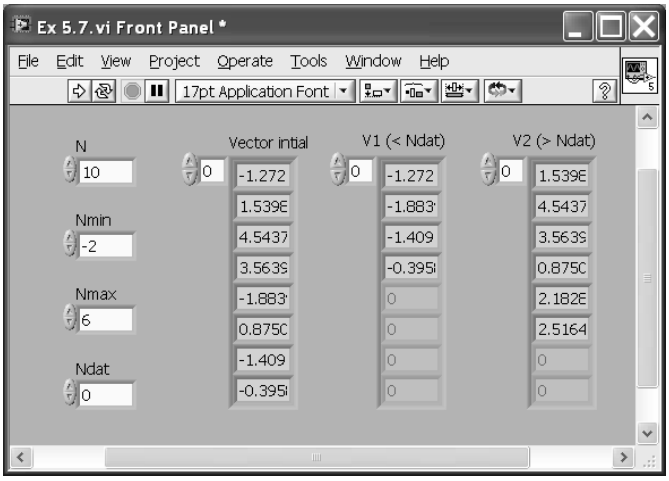


Figura 5.25

Mod de lucru

1. Deschideți un nou IV.

2. Plasați pe PF: un control numeric I32 etichetat *N*, două controale numerice de tip real double etichetate *Nmin* şi *Nmax*, un control numeric double etichetat *Ndat* şi trei indicatoare vector etichetate *Vector initial*, *V1 (< Ndat)* şi *V2 (>Ndat)* (figura 5.25).
3. Generarea vectorului inițial se va face utilizând funcția *Random Number (0 – 1)* din subpaleta de funcții numerice, pe baza următoarelor relații:

$$\begin{aligned} &0 < x < 1 \;\mid\; \cdot(N\max - N\min) \\ &0 < x < N\max - N\min \;\mid\; + N\min \\ &N\min < x < N\max \end{aligned} \tag{5.6}$$

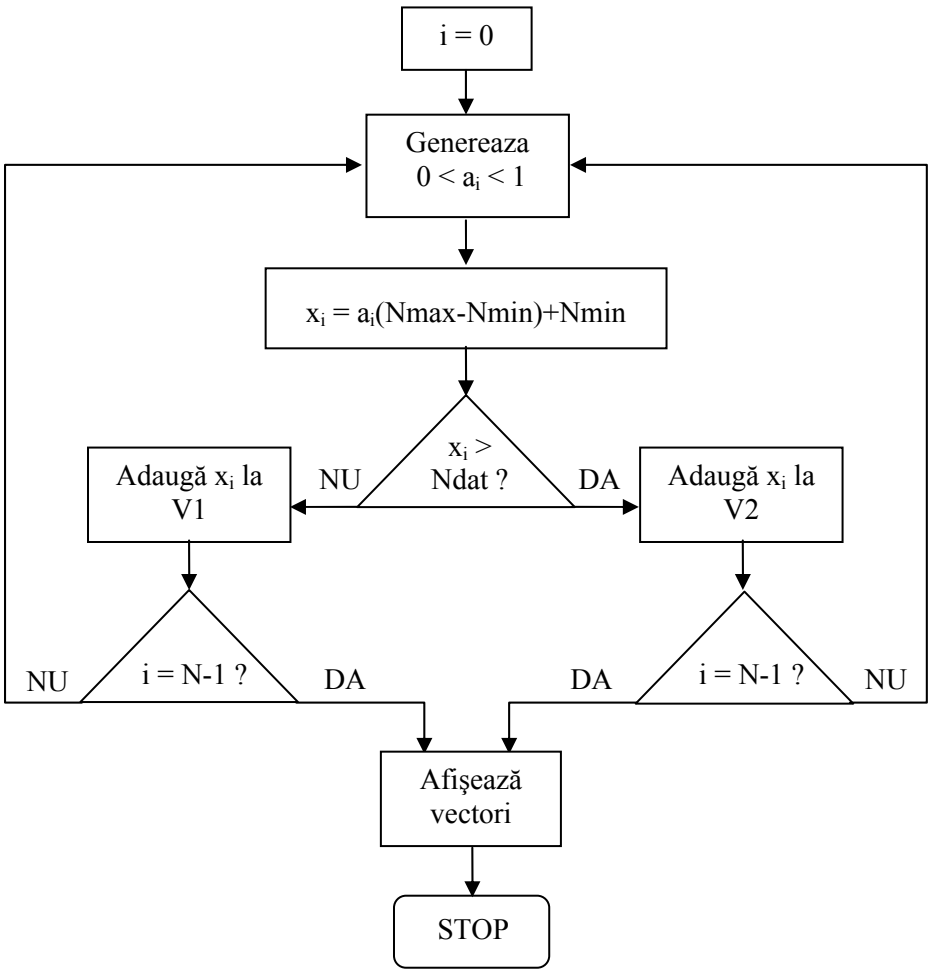


Figura 5.26

Intr-o buclă FOR care va rula de *N* ori, înmulțiți aşadar ieşirea funcției *Random Number (0 – 1)* cu valoarea (Nmax-Nmin), după care adunați rezultatul cu Nmin. Numărul obținut va fi cuprins între Nmin şi Nmax.

Pentru selectarea celor 2 vectori după poziția numerelor generate față de numărul dat, vom utiliza un algoritm bazat pe organigrama din figura 5.26.

Construirea vectorilor V1 şi V2 se face în urma comparației numărului generat cu numărul dat, al cărei rezultat selectează două cazuri: cazul FALSE corespunde construirii vectorului V1, iar cazul TRUE corespunde construirii vectorului V2. Construirea vectorilor se face element cu element, utilizând câte o funcție *Build Array* şi un registru de deplasare inițializat cu vectorul vid.

4. Construiți diagrama din figura 5.27. Inițializarea celor doi regiştri de deplasare cu vectorul vid se face printr-un MD pe terminalul din stânga şi selectare *Create Constant.*

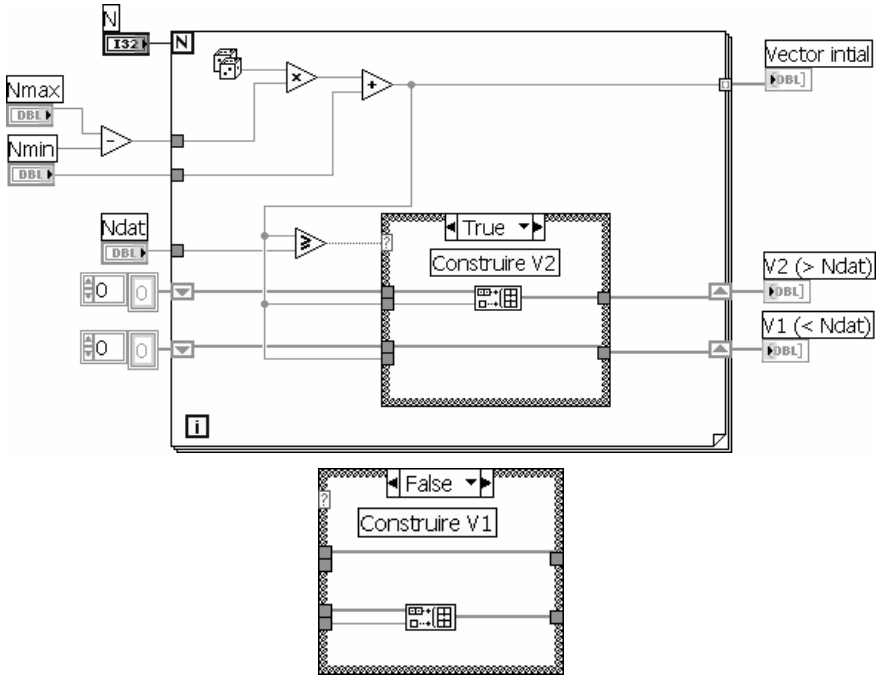


Figura 5.27

5. Salvați IV-ul sub numele *Ex 5.7.vi*.
6. Dați valori parametrilor *N, Nmin, Nmax, Ndat* şi rulați instrumentul. Verificați corectitudinea separării celor doi vectori din vectorul nițial.

Exercițiul 5.8

Enunț

Să se realizeze un IV care să contorizeze de câte ori s-a apăsat pe un buton OK de pe PF. Instrumentul să se poată opri în orice moment de la un buton de stop.

Mod de lucru

Se utilizează două bucle WHILE, una în interiorul alteia. Bucla exterioară contorizează apăsarea butonului OK, în timp ce în bucla interioară se aşteaptă apăsarea butonului, care condiționează ieşirea din această buclă. Ieşirea din ambele bucle este forțată de apăsarea butonului STOP. Organigrama problemei este dată în figura 5.28, iar diagrama de legături este dată în figura 5.29.

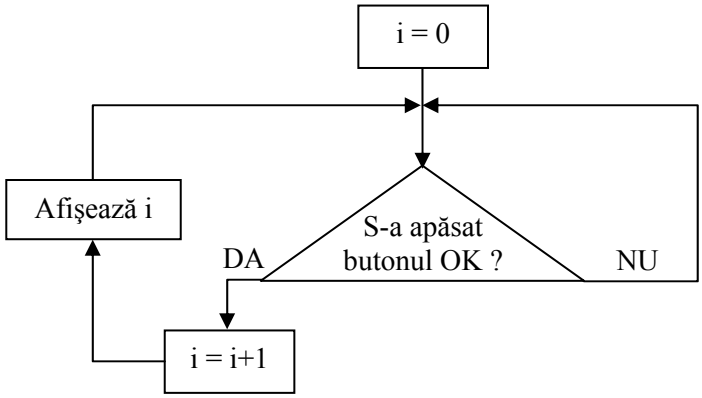


Figura 5.28`

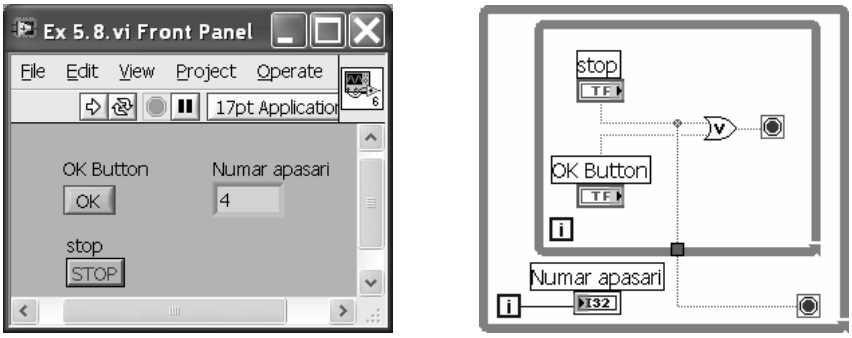


Figura 5.29

Exercițiul 5.9

Enunț

Să se realizeze un IV care să aprindă un indicator de tip *Color Box* pe culorile roşu, galben şi albastru în funcție de poziția unui comutator fixat pe panoul frontal.

Mod de lucru

1. Deschideți un nou IV.
2. Plasați pe PF un indicator de tip *Color Box* şi un control numeric de tip *Vertical Pointer Slide*.
3. Schimbați reprezentarea controlului în I32.
4. Deschideți meniul shortcut al controlului şi selectați *Text Labels*. In acest moment controlul are doar două poziții, *min* şi *max*, fiecărei poziții fiindu-i asociat un număr.
5. Deschideți meniul shortcut al controlului text 

şi selectați *Edit Items...*
6. Faceți un dublu click pe textul *min* şi scrieți cuvântul *rosu*.
7. Faceți un dublu click pe textul *max* şi scrieți cuvântul *galben*.
8. Faceți un click sub cuvântul *galben*. S-a adăugat o nouă linie. Adăugați acestei linii cuvântul *albastru*.
9. Apăsați OK. Observați cum controlul slide are acum trei poziții: *roşu, galben şi albastru.* Fiecărei poziții îi corespunde câte un număr: 0 pentru roşu, 1 pentru galben şi 2 pentru albastru (figura 5.30).

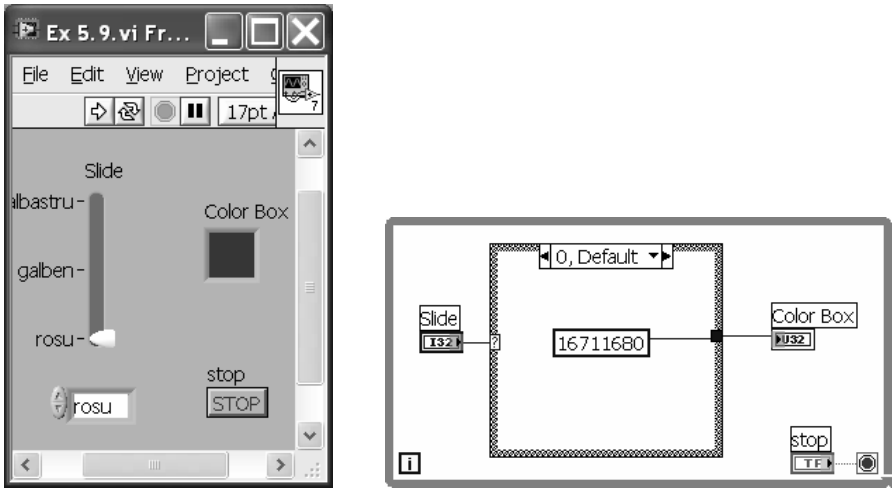


Figura 5.30

10. Indicatorul *Color Box* este un indicator de tip numeric, care afişează o culoare corespunzătoare unui anumit număr.
11. Plasați pe PF o structură CASE, căreia îi legați controlul slide la selector.

12. In cadrul 0 plasați o constantă numerică din subpaleta de funcții numerice. Dați acestei constante valoarea 16711680.
13. In cadrul 1 plasați o constantă căreia îi atribuiți valoarea 16773632.
14. Adăugați un nou caz după cazul 1. Plasați în cadrul 2 constanta 655580.
15. Scoateți toate cele 3 constante la ieşirea structurii CASE printr-un tunel comun, pe care-l legați la indicatorul *Color Box*.
16. Încadrați întreaga diagramă într-o buclă WHILE asistată de un buton de STOP.
17. Rulați instrumentul şi verificați funcționalitatea.

Exerciții propuse

EP 5.2.
Să se construiască un vector format din N numere aleatoare cuprinse între două numere N1 şi N2 fixate de pe panoul frontal. Să se ordoneze apoi aceste numere în ordine crescătoare sau descrescătoare, selectabil dintr-un control de pe panoul frontal.

EP 5.3.
a) Să se construiască un vector format din N numere întregi aleatoare cuprinse între două numere date, N_{min} şi N_{max} fixate de pe panoul frontal.
b) Să se determine apoi de câte ori un număr dat pe PF coincide cu elementele vectorului construit şi să se afişeze un vector cu indecşii elementelor găsite.
c) Să se înlocuiască elementele găsite cu valoarea lui N_{max}+1.

EP 5.4.
Să se construiască o matrice cu două linii, care să conțină:
- pe prima linie toate numerele pare cuprinse între două numere date
- pe a doua linie toate numerele impare cuprinse între două numere date.

EP 5.5.
Să se realizeze un instrument virtual care să calculeze şi să afişeze toți divizorii unui număr dat.

EP 5.6.
Să se realizeze un IV care să afişeze toate numerele prime până la un număr dat pe panoul frontal.

EP 5.7.
Să se construiască o matrice cu M linii şi N coloane (M şi N fixate de pe panoul frontal) ale cărei elemente sunt numere întregi aleatoare cuprinse între 0 şi 50.
Să se afişeze apoi de câte ori un număr dat pe PF se află printre elementele matricii de mai sus şi să se înlocuiască aceste elemente cu numărul 51.

EP 5.8.
Să se realizeze un IV care să elimine dintr-un vector dat pe PF un anumit element şi să afişeze vectorul rămas.

EP 5.9.
Fie semnalul digital ale cărui eşantioane sunt date în tabelul de mai jos. Ştim că eşantionarea s-a efectuat cu perioada T_0 = 1 ms. Să se realizeze un IV care să determine automat perioada unui semnal eşantionat aplicat la intrare şi să se exemplifice funcţionarea prin exemplul de semnal de mai jos.

☞ Se poate utiliza funcţia *Threshold 1D array* şi structuri adecvate.

0.2	2.2	4.2	6.2	4.2	2.2	0.2	-1.8	-3.8	-5.8	-3.8	-1.8	0.2	2.2	4.2	6.2

EP 5.10.
a) Să se realizeze un IV care să simuleze, utilizând generatorul de semnale aleatoare, aruncarea a două zaruri. Aruncarea se iniţiază de la un buton de pe PF.
b) Să se afişeze de câte ori s-a aruncat o anumită valoare a sumei zarurilor într-o sesiune de aruncări. Panoul frontal al instrumentului este dat în figura 5.31.

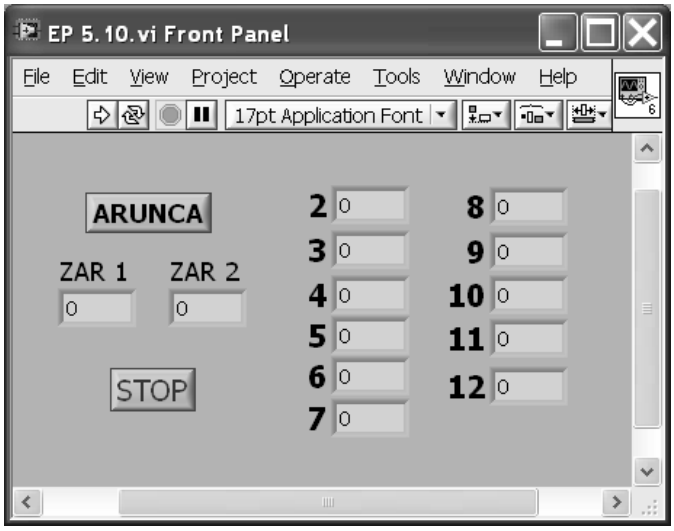


Figura 5.31

EP 5.11.
Să se realizeze un IV care să genereze un vector de N numere aleatoare întregi cuprinse între două numere date pe PF. Să se construiască apoi un alt vector care să fie format din produsele elementelor primului vector, luate două câte două. De exemplu, primul element al celui de-al doilea vector este produsul primului cu al doilea element al primului vector, al doilea element este produsul dintre al treilea şi al patrulea, ş.a.m.d.

ŞIRURI DE CARACTERE

Caracterele sunt tipuri de date stocate pe un singur octet, reprezentate în calculator prin valoarea codului lor ASCII (American Standard Code for Information Interchange).

Şirurile de caractere (*string*) reprezintă succesiuni de coduri ASCII. Ele se utilizează în special acolo unde este nevoie ca informaţiile să poată fi citite şi editate direct, utilizând programe de procesare a textelor. Conversia datelor de orice tip în şiruri de caractere se face de regulă pentru salvarea acestora în fişiere sau pentru afişarea de mesaje pe panoul frontal. Conversia şirurilor de caractere în alte tipuri de date se face la descifrarea unor mesaje furnizate de unele aparate de măsură, extragerea informaţiei de măsură. Controalele şi indicatoarele de tip *şir de caractere* pot fi accesate din subpaleta *String & Path*. In controale, introducerea datelor sub formă de text se introduce de la tastatură.

Reprezentarea binară a unui număr scris sub forma şirurilor de caractere este diferită de reprezentarea lui sub formă numerică. Operaţiile cu numere sub forma *string* sunt specifice acestor tipuri de date şi se găsesc în subpaleta de funcţii *String* şi nu *Numeric*.

Afişarea datelor pe controlul (indicatorul) de pe PF poate fi făcută în următoarele feluri, selectabil din meniul pop-up:

- **Normal Display** – arată caracterele aşa cum au fost introduse, fără a se evidenţia semnele speciale (Tab, CR, Space, etc.)
- **'\' Codes Display** - afişează şi codurile caracterelor speciale. Acestea sunt:

Tabelul 6.1

Cod	Semnificaţie
\00 - \FF	Valoarea în hexa a unui caracter. Trebuie să fie scris cu litere mari
\b	Backspace (ASCII BS, \08)
\f	Form feed (ASCII FF, \0C)
\n	Linefeed (ASCII LF, \0A)
\r	Carriage return (ASCII CR, \0D)

\t	Tab (ASCII HT, \09)
\s	Space (echivalent \20)
\\	Backslash (ASCII \, \5C)

- **Password Display - a**fişează în loc de caractere semnul *
- **Hex Display** - afişează codurile ASCII ale caracterelor în hexa. Ca şi **'\' Codes Display**, această afişare este utilă la depanare şi atunci când se comunică cu instrumente.

Funcţii cu şiruri de caractere

Aceste funcţii se găsesc în subpaleta *String* de pe paleta de funcţii. Vom prezenta în continuare două dintre funcţiile de bază, cu care se realizează conversia din tipul de date caracter în alte tipuri de date şi invers.

Funcţia Scan From String

Această funcţie scanează şirul de intrare şi-l converteşte în conformitate cu formatul stabilit la intrarea *Format String*. Pentru stabilirea formatului, se poate utiliza fereastra interactivă de dialog obţinută prin dublu clic pe funcţie sau din meniul shortcut al funcţiei, opţiunea *Edit Scan String*.

Datele sunt disponibile la ieşirile *Output (n)*, care iau forma tipului de dată stabilită la format. Dacă tipul de dată specificat prin format nu este găsit în şir, atunci la ieşire se afişează valorile implicite stabilite prin intrările *Default Value (n)*. Se pot stabili oricâte ieşiri prin redimensionarea numărului acestora (la fel ca la matrici).

Este necesar să se ştie dinainte formatul datei căutate.

Exemple

Tabelul 6.2

Input string	Format string	Default(s)	Output(s)	Remaining string
abc, xyz, 12.3+56i 7200	%3s, %s%f%2d	-	abc xyz 12.3+56i 7	00
Q+1.27E-3 tail	Q%f t	-	1.27E-3	ail
0123456789	%3d%3d	-	12 345	6789

X:9.860 Z:3.450	X:%fY:%f	100 (I32) 100.0 (DBL)	10 100.0	Z: 3450
set49.4.2	set%d	-	49	.4.2

Funcția Format Into String

Această funcție converteşte datele de la intrare în şir de caractere, în concordanță cu formatul specificat. Datele de intrare pot fi şiruri de caractere, numere, date de tip *enum*, căi şi amprente de timp (numărul de secunde scurs de la Timpul Universal, care este vineri, 1 ianuarie 1904, ora 0.00). Nu admite ca intrări matrici şi clustere. Pentru stabilirea formatului se poate utiliza fereastra interactivă de dialog obținută prin dublu clic pe funcție sau din meniul shortcut al funcției, opțiunea *Edit Scan String*. Se pot adăuga oricâte operații de conversie (intrări), prin scrierea acestora în directiva de format una după alta, fără spații. La ieşire, şirurile corespunzătoare se concatenează.

Input String este un şir inițial, care se concatenează cu rezultatul conversiilor.

Stabilirea formatului

Sintaxa formatului din funcțiile *Scan From String* şi *Format Into String* este următoarea:

[Str]%[-][0][lățime][.precizie]{u.m.}Conversie[Str]

unde:

Str – un şir oarecare care precede sau succede numărul

[-] – aliniere la stânga. Dacă lipseşte, alinierea se face la dreapta.

[0] – indică faptul că locurile libere se completează cu zerouri. Dacă lipseşte, completarea se face cu spații goale.

[lățime] – numărul minim de caractere alocate numărului

[.precizie] – numărul de cifre zecimale

{u.m.} – unitatea de măsură

Conversie este unul din caracterele:

f – format fracționar (ex. 12,345)
e – format ştiințific (ex. 1,234E1)
g – format fracționar/ştiințific
p – format în sistemul internațional de unități
d – format întreg zecimal (ex. 12)
x – întreg în hexazecimal (ex. B8)
o - întreg în octal (ex. 701)
b – întreg în binar (ex. 10110)

t – relativ la timp
T – amprentă de timp
s – string (ex. Abc)
% - caracter

Exemple

Formatul **%3.2f** semnifică:

3 = numărul minim de caractere alocat numărului. Numărul se aliniază la dreapta. Dacă avem **%-3.2f**, numărul se aliniază la stânga. Dacă avem **%03.2f**, locurile rămase libere se completează cu 0.

2 = numărul de cifre de după virgulă.

Tabelul 6.3

Format	Argument	Ieşire
score= %2d%%	87	score= 87%
level= \n%-7.2e V	0.03642	level= 3.64e-2 V
Name: %s, %s.	Smith John	Name: Smith, John.
Temp: %05.1f %s	96.793 Fahrenheit	Temp: 096.8 Fahrenheit
String: %10.5s.	Hello, World	String:_____Hello.
%5.3f	5.67 N	5.670 N
%5.3{mN}f	5.67 N	5670.000 mN
%5.3{kg}f	5.67 N	5.670 ?kg

Notă: Ultimele 3 exemple sunt pentru intrări însoţite de unităţi de măsură. La ultimul exemplu nu a fost recunoscută unitatea de măsură.

Există o serie de funcţii de conversie numeric – string şi invers în subpaleta *String/Number Conversion*. Acestea sunt funcţii mai simple, nu este necesară specificarea formatului, însă sunt dedicate unui anumit tip de număr. De exemplu funcţia *Number to Fractional String* converteşte un număr real într-un şir de caractere, căreia i se pot specifica lungimea şi precizia ca intrări în funcţie.

Funcţiile duale *Array to Spreadsheet String* şi *Spreadsheet String to Array*

Aceste funcţii realizează conversiile matricilor şi a vectorilor în formate de stringuri speciale (*Spreadsheet*) şi invers. Conform acestui format, elementele de pe o linie sunt separate între ele prin tabulatori, iar liniile sunt separate prin

Carriage return şi *Line Feed* (echivalent cu apăsarea tastei Enter). Se pot specifica totuşi şi alte semne de separaţie între elemente în afara tabulatorilor.

Acest tip de scriere a stringurilor este recunoscut şi de alte medii de programare sau programe utilizate: C, Excel, Mathcad, Spice, etc. La aceste funcţii, formatul se scrie la fel ca la funcţia *Format Into String*, însă nu se mai face în mod interactiv.

Exerciţiul 6.1

Scop

Studiul funcţiilor cu şiruri de caractere din subpaleta *String*.

Mod de lucru

1. Deschideţi un nou IV.
2. Plasaţi pe PF un control de tip *String* din subpaleta de controale *String & Paths*.
3. Introduceţi în control următorul text:
   ```
   Aceasta este prima linie.
   Aceasta este a doua linie.
   ```
4. Vizualizaţi textul în cele 4 moduri: *Normal, „/” Codes Display, Password Display, Hex Display*.
5. Utilizând vizualizarea în hexa, determinaţi codurile ASCII ale următoarele caractere: A, e, p, . , spaţiu (\s), sfârşit de linie (\n).
6. Treceţi pe DL şi deschideţi subpaleta de funcţii *String*.
7. Treceţi în revistă funcţiile disponibile şi citiţi-le helpul.
8. Experimentaţi funcţiile: *String length, Concatenate string, String subset, To upper case, To lower case, Replace substring, Search and replace string, Match pattern.*

Exerciţiul 6.2

Scop

Studiul funcţiei *Format Into String*..

Mod de lucru

1. Deschideţi un nou IV.
2. Plasaţi pe PF un control *numeric* şi un indicator *string*.
3. Introduceţi în control numărul 7,54673.
4. Aduceţi pe DL funcţia *Format Into String*.
5. Citiţi helpul funcţiei şi identificaţi intrările şi ieşirile.

6. Deschideți fereastra interactivă de stabilire a formatului (dublu click pe funcție sau deschideți meniul shortcut, opțiunea *Edit Format String*).
7. In secțiunea *Selected operation* modificați opțiunile disponibile şi observați cum se modifică formatul în fereastra *Corresponding Format String*.
8. Legați numericul la intrarea *Input 1* şi indicatorul *string* la ieşirea funcției.
9. Validați *Use minimum field width* (1) şi *Use maximum string length* (2).
10. In (1) scrieți cifra 2 şi în (2) scrieți cifra 3. Observați cum s-a modificat formatul.
11. Rulați instrumentul şi observați rezultatul.
12. In (1) scrieți cifra 5, iar în (2) scrieți cifra 1. Rulați şi observați rezultatul. In indicatorul *string*, numărului convertit i s-au rezervat 5 spații din care doar 3 sunt ocupate, celelalte două fiind lăsate libere. Şirul de caractere ce reprezintă numărul este aliniat la dreapta. Cele două spații libere pot fi observate dacă pe indicatorul *string* se afişează codurile caracterelor (opțiunea *„/" Codes Display*).
13. Deschideți din fereastra de editare a formatului şi în secțiunea *Options* selectați *Pad using Zeros*. Observați cum s-a modificat formatul.
14. Rulați şi observați rezultatul.
15. In fereastra de editare a formatului, în secțiunea *Options* selectați *Left justify*. Observați cum s-a modificat formatul.
16. Rulați şi observați rezultatul.
17. Adăugați unitatea de măsură Newton (N) controlului numeric. Pentru aceasta selectați *Visible Items – Unit Label*. In eticheta unității scrieți N.
18. Rulați din nou şi observați cum la numărul convertit în string se adaugă unitatea de măsură.
19. Se poate face conversia direct a unei unități de măsură în altă unitatea de măsură pentru aceeaşi mărime. Pentru aceasta se adaugă după cifra de precizie, între acolade, noua unitate de măsură. Poate fi şi un multiplu sau submultiplu al aceleiaşi unități ca la intrare.
20. Înlocuiți formatul existent cu formatul %5.3{mN}f.
21. Rulați şi observați în rezultat conversia N → mN.
22. Adăugați o nouă operație (apăsați butonul *Add New Operation*) căreia îi stabiliți un format binar. Observați că s-a creat o nouă intrare funcției. Creați automat un control numeric acestei noi intrări şi dați o valoare oarecare controlului.
23. Rulați şi observați în rezultat scrierea în binar a părții întregi a numărului.
24. Adăugați un nou argument cu format string. Observați cum intrarea nou creată ia culoarea specifică tipului de date din format.
25. Legați un control *string* la intrare (care să conțină şi numere) şi rulați.
26. Experimentați exemplele din tabelul 6.3.
27. Salvați instrumentul cu numele *Ex 6.2.vi*.

Exercițiul 6.3

Scop

Studiul funcției *Scan From String*.

Mod de lucru

1. Deschideți un nou IV.
2. Plasați pe PF un control *string*, un indicator *numeric* şi un indicator *string*.
3. Plasați pe DL funcția *Scan From String*.
4. Citiți helpul funcției şi identificați intrările şi ieşirile.
5. Experimentați exemplele din tabelul 6.2.

Exercițiul 6.4

Scop

Studiul funcției *Array to Spreadsheet String*.

Mod de lucru

1. Deschideți un nou IV.
2. Plasați pe PF un control de tip matrice 3 x 2 cu elementele diverse numere reale.
3. Aplicați matricea la intrarea unei funcții *Array to Spreadsheet String***.**
4. Experimentați cu diverse formate: %3.2f, %10.2f cu aliniere la stânga şi la dreapta, %d, %2d. Observați cum se realizează conversia pe un indicator de tip *string*.
5. Afişați pe indicator scrierea codată şi observați poziția spațiilor, taburilor, etc.
6. Experimentați şi cu alte semne de delimitare între elementele matricii în loc de *tab*.

Exercițiul 6.5

Enunț

Pe PF se plasează un control numeric. Să se afişeze pe un indicator de tip *string* următorul text:

```
Numarul afisat este ….
Textul a fost creat la data de: ……., ora ……
Autorul textului este (numele).
```

Mod de lucru

Rezolvarea problemei necesită concatenarea unor şiruri de caractere, obținute fie din texte scrise sub formă de constante, fie din conversia altor tipuri de date în

şiruri de caractere. Funcţia cea mai potrivită pentru construirea textelor prin concatenare este *Format Into String*, deoarece aceasta permite şi conversia directă a numerelor şi a altor tipuri de date în şiruri de caractere. O altă funcţie cu care se pot construi texte prin concatenare este *Concatenate Strings*.

1. Deschideţi un nou IV.
2. Plasaţi pe PF un control numeric.
3. Plasaţi pe DL funcţia *Format Into String.*
4. Realizaţi diagrama din figura 6.1. Trecerea de la o linie la alta se face cu constantele *Line Feed* sau *Carriage Return*. Data şi ora curentă se obţin direct în format *string* cu ajutorul funcţiei *Get Date/Time String* din subpaleta *Timing*.

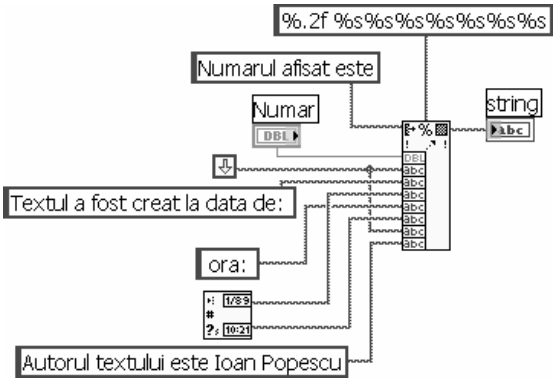


Figura 6.1

5. Rulaţi instrumentul şi verificaţi rezultatul.
6. Rezolvaţi aceeaşi problemă utilizând funcţia *Concatenate Strings.*
7. Salvaţi instrumentul sub numele *Ex 6.5.vi*.

Exerciţii propuse

EP 6.1.
Să se realizeze un IV care să caute într-un text dat un grup de litere şi să-l înlocuiască cu alt grup de litere.
Variantă: Înlocuirea grupului căutat să se facă cu un număr fixat pe PF.

EP 6.2.
Să se realizeze un instrument virtual care să extragă şi să afişeze într-un vector toate numerele reale dintr-un text dat.

EP 6.3.
Să se realizeze un IV care să extragă dintr-un text dat toate cuvintele care încep cu litera *s* şi să le afişeze într-un vector de stringuri.

Afişarea datelor în tabele

Un tabel este un indicator în care informaţia este ordonată pe linii şi coloane. Ca tip de dată, tabelul este o matrice cu 2 dimensiuni de date de tip şir de caractere.

Tabelul se găseşte în subpaleta de controale *List & Table*.

Tabelul este format din linii şi coloane, separate prin linii de delimitare. Liniile de delimitare pot fi făcute vizibile sau nu din *Visible Items*. Fiecare linie (coloană) poate conţine câte un cap de tabel (*header*). Inscripţionarea capurilor de tabel se face manual. *Headerul* este separat de restul datelor printr-o linie dublă. El poate fi de asemenea făcut vizibil sau nu din *Visible Items.*

Dacă tabelul conţine mai multe date decât suprafaţa vizibilă, sunt disponibile bare de defilare (scrollbar) atât pe verticală cât şi pe orizontală. Aceste bare pot fi ascunse din *Visible Items*.

Introducerea datelor în tabel se face cu unealta text. Datele de intrare (ieşire) în (din) tabel sunt matrici de stringuri. Conversia matrice de numere - matrice de stringuri se face cu funcţiile de conversie din subpaleta de funcţii *String/Number Conversion*.

Exerciţiul 6.6

Scop

Studiul tabelelor.

Mod de lucru

1. Deschideţi un nou IV.
2. Plasaţi pe PF un control de tip tabel. Schimbaţi controlul în indicator.
3. Afişaţi header-ul de coloane şi editaţi celule corespunzătoare primelor 3 coloane cu textele: *Coloana 1, Coloana 2, Coloana 3*.
4. Plasaţi o matrice numerică bidimensională cu 2 linii şi 3 coloane cu elemente numere oarecare.
5. Aplicaţi matricea tabelului prin intermediul funcţiei de conversie *Number To Fractional String* din subpaleta *String/Number Conversion*. Fixaţi precizia la 3 cifre după virgula zecimală.
6. Rulaţi instrumentul şi observaţi rezultatul.
7. Salvaţi sub numele *Ex 6.6.vi*.

Exerciţiul 6.7

Enunţ

Să se construiască un instrument virtual care să afişeze într-un tabel, pe coloane, valorile funcţiilor:

$$f1(x) = x^2 + \log x + 1$$
$$f2(x) = 2\sin(\pi x / 180)$$
$$f3(x) = x^3 + \sqrt{x}$$

pentru toate valorile lui x număr întreg cuprinse între 2 şi 15.
Headerul fiecărei coloane conține numele funcției corespunzătoare.

Mod de lucru

1. Deschideți un nou IV.
2. Plasați pe PF un tabel indicator.
3. Adăugați tabelului headerul de coloane.
4. Scrieți în header numele fiecărei funcții: f1, f2, f3.
5. Plasați pe DL o buclă FOR.
6. Construiți diagrama din figura 6.2. Numărul de rulări este 14 (numerele de la 2 la 15). S-a utilizat *Formula Node* pentru o mai uşoară scriere a relațiilor. Rezultatele sunt scoase la ieşirea buclei FOR prin autoindexare, după care se construieşte matricea cu toate cele 3 şiruri de rezultate utilizând *Build Array*. Conversia matricii numerice în matrice string se face cu funcția *Number To Fractional String* (funcția *Format Into String* nu acceptă matrici la intrare). A fost necesară transpunerea pentru ca şirurile cu rezultate să fie dispuse pe coloane.
7. Rulați instrumentul şi observați rezultatul.
8. Salvați sub numele *Ex 6.7.vi*.

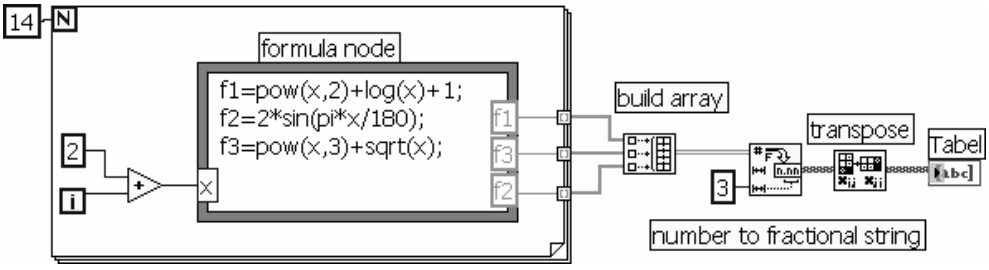


Figura 6.2

Exerciții propuse

EP 6.4.
Să se realizeze un IV care să ordoneze în ordine alfabetică, pe coloane, cuvintele scrise într-un tabel.

EP 6.5.
Să se realizeze un IV care să afişeze într-un tabel, pe M linii şi N coloane, numere întregi aleatoare cuprinse între două numere date, N1 şi N2. Să se găsească apoi

toate coincidențele numerelor din tabel cu un număr dat K şi să se afişeze numărul acestor coincidențe şi coordonatele elementelor găsite (nr.linie, nr.coloană).
De pe panoul frontal se fixează: M, N, N1, N2, K.

Capitolul 7

VARIABILE LOCALE ŞI GLOBALE

Variabilele locale (VL) sunt obiecte de pe DL care corespund unor controale sau indicatoare de pe PF, fiind identificate după eticheta acestora. Scrierea într-o variabilă locală are acelaşi rezultat ca şi introducerea datelor într-un terminal, chiar dacă acesta este control sau indicator. In fapt, variabila locală este o "oglindă" a zonei de memorie unde se află terminalul corespunzător. Cu variabila locală se poate citi acea zonă de memorie sau se poate scrie în ea. Aşadar, printr-o variabilă locală un control se poate utilizat fie ca intrare, fie ca ieşire de date.

Datele dintr-o variabilă locală pot fi utilizate doar în IV-ul în care a fost creată variabila. Putem avea oricâte variabile locale pentru un terminal. Acestea se pot utiliza în situaţii în care prezenţa terminalului este necesară în mai multe locuri odată (de exemplu în structuri). De asemenea, VL sunt utile atunci când se doreşte transportul datelor pe o diagramă complicată, fără a mai fi necesară realizarea legăturii prin fir.

Crearea unei variabile locale se face în două moduri:

1) Deschideţi meniul shortcut al terminalul al cărui variabilă locală doriţi să o creaţi, apoi selectaţi *Create – Local Variable*. Se creează astfel în mod automat variabila locală cu numele etichetei terminalului.
2) Accesaţi de pe DL subpaleta de funcţii *Structures – Local Variable.* Plasaţi variabila pe DL. Faceţi apoi MD pe semnul întrebării şi selectaţi *Select Item*. Se deschide o fereastră cu numele tuturor terminalelor care acceptă atribuirea de variabile locale. Alegeţi din această fereastră terminalul dorit.

Exerciţiul 7.1

Scop

Studiul variabilelor locale.

Mod de lucru

1. Deschideţi un nou IV.

2. Plasați pe PF un control numeric.
3. Creați pentru controlul numeric o variabilă locală.

In momentul creării, variabila locală este capabilă să primească date. Dacă mergeți cu prompterul pe variabilă, veți observa apariția unui terminal în partea stângă a ei, semn că aceasta primește date. Schimbarea variabilei locale din primitor în furnizor de date se face din meniul shortcut, opțiunea *Change To Read*.

4. Legați la VL un control.
5. Dați o valoare controlului inițial diferită de 0 şi rulați instrumentul. Observați că, deoarece controlul legat la VL conține valoarea 0, şi controlul reprezentat de VL primeşte valoarea 0.
6. Dați acum o valoare diferită de 0 controlului legat la VL. Rulați instrumentul. Observați cum controlul inițial ia valoarea VL.
7. Schimbați VL în variabilă de citire (*Change To Read*) şi controlul asociat ei în indicator. Refaceți legătura.
8. Dați o valoare controlului inițial şi rulați instrumentul. Observați cum indicatorul citeşte valoarea VL, deci a controlului pe care îl reprezintă.
9. Repetați studiul cu date de tip *boolean*, *string* şi *array*.
10. Salvați sub numele *Ex 7.1.vi*.

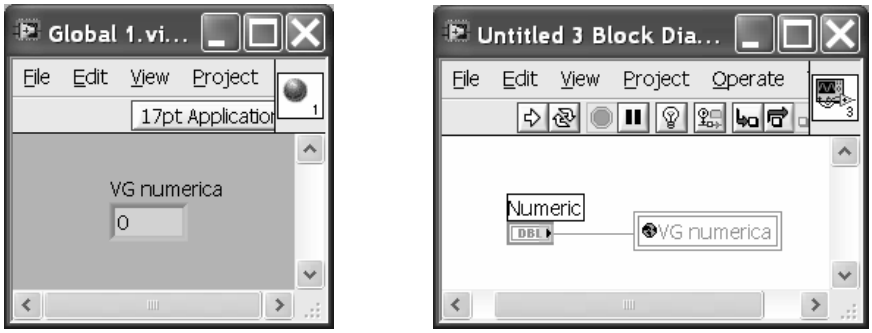
Variabilele globale (VG) sunt variabile stocate în nişte IV-uri care au doar panou frontal, numite containere de variabile globale. VG sunt stocate în nişte controale de pe PF al containerului, care păstrează valorile variabilelor utilizate în IV-urile care le apelează. Scopul creării acestor tipuri de variabile este de a utiliza datele create într-un IV în mod global şi în alte IV-uri.

Ca şi variabilele locale, cele globale sunt apelabile pe DL prin noduri specifice, care se găsesc în subpaleta de structuri – *Global Variable*. Realizarea unui dublu click pe VG de pe DL deschide PF al containerului. Aici se pot adăuga oricâte controale de pe paleta de controale, fiecare din acestea reprezentând o VG. Obligatoriu, controalele se vor eticheta, deoarece apelarea variabilelor se va face prin această etichetă. In figura 7.1a este prezentat containerul *Global1.vi* care conține variabila globală *VG numerică*. Această variabilă globală este apelată de către IV-ul din figura 7.1b, care este înscrisă prin controlul *Numeric*.

După plasarea unei VG pe DL şi definirea unui PF pentru ea, nodul este asociat acelui PF. Deoarece sunt mai multe variabile asociate aceluiaşi PF, trebuie selectat dintr-o listă numele variabilei căreia i se adresează. Această listă se afişează prin dublu click pe VG creată sau din meniul shortcut opțiunea *Select item*.

Intr-o VG se pot scrie sau din ea se pot citi date. Ca şi la VL, pentru citire se selectează din meniul shortcut *Change to Read*, iar pentru scriere se selectează *Change to Write*. VG pot fi citite sau scrise de către orice IV aflat în memorie la un moment dat. Trebuie avut însă grijă ca citirea să se facă atunci când datele sunt stabile, adică un alt IV să nu scrie atunci când VG este citită.

Plasarea aceleiaşi VG în alt IV se face prin apelarea ei ca orice subIV din subpaleta de funcții, cu opțiunea *Select a VI*.



a) b)

Figura 7.1

Exercițiul 7.2

Scop

Studiul variabilelor globale.

Mod de lucru

1. Deschideți un nou IV.
2. Plasați pe DL o variabilă globală din paleta de structuri, *Global Variable*.
3. Deschideți PF al VG prin dublu click pe ea sau din meniul shortcut opțiunea *Open Front Panel*. Acesta este un container de VG. Observați că în meniul *View* nu este validă deschiderea diagramei de legături.
4. Plasați pe PF al containerului un control numeric etichetat *VG numerica*, unul boolean etichetat *VG booleana* şi unul string etichetat *VG string*.
5. Salvați containerul sub numele implicit (*Global 1.vi*) în acelaşi folder unde salvați şi IV-urile. Lăsați PF al containerului deschis.
6. Treceți pe PF al IV-ului inițial. Plasați pe PF un control de tip numeric.
7. Deschideți meniul shortcut al variabilei globale. Selectați *Select Item*. Observați că aveți la dispoziție toate variabilele globale din containerul *Global1.vi*.
8. Selectați *VG numerica*.
9. Legați variabila la controlul numeric.
10. Micşorați fereastra astfel încât să aveți disponibile concomitent şi PF al instrumentului, şi PF al containerului *Global 1.vi*.
11. Rulați IV-ul cu *Run Continuously* şi dați valori controlului numeric. Observați modificarea concomitentă a valorii variabilei *VG numerica* de pe PF al containerului.

12. Opriți IV-ul de la butonul *Abort Execution*.
13. Adăugați pe DL a instrumentului variabilele globale *VG booleana* şi *VG string*. Pentru aceasta aveți două posibilități:
    a. copiați variabila *VG numerica*, apoi selectați variabila dorită cu *Select Item*.
    b. adăugați o nouă variabilă din subpaleta de funcții opțiunea *Select a VI...*, după care selectați containerul *Global1.vi* şi variabila dorită cu *Select Item*.

Adăugarea unei noi VG din subpaleta de structuri înseamnă crearea unui nou container, căruia trebuie să-i adăugați alte variabile globale.

14. Legați controale corespunzătoare variabilelor şi experimentați scrierea cu diverse valori.
15. Deschideți un nou IV.
16. Adăugați pe DL variabila globală *VG numerica* după procedeul de la pct. 13.
17. Schimbați variabila în citire, din meniul shortcut – *Change To Read*.
18. Creați automat variabilei un indicator.
19. Potriviți ferestrele celor două instrumente şi a containerului astfel încât acestea să fie afişate concomitent.
20. Porniți cele două instrumente şi modificați valoarea controlului numeric al primului instrument, urmărind valoarea indicatorului celui de-al doilea instrument şi valoarea variabilei globale de pe PF al containerului.
21. Salvați cele două instrumente sub numele *Ex 7.2a.vi* şi *Ex 7.2b.vi*.

Exercițiul 7.3

Enunț

Să se realizeze un IV care să calculeze şi să afişeze pe un indicator *string* întâi suma şi apoi produsul a două numere fixate de pe panoul frontal. Instrumentul funcționează după următorul protocol:

1. La pornirea instrumentului, se realizează suma numerelor şi se afişează pe un indicator mesajul: *Suma numerelor este:....*
2. La apăsarea unui buton OK de pe PF, se trece la efectuarea calculului produsului, după care se afişează mesajul: *Produsul numerelor este:....*
3. La apăsarea din nou a butonului OK se deschide o fereastră de dialog pe care se afişează textul: *Doriți un nou calcul?*
4. Dacă se doreşte un nou calcul, se apasă butonul DA, iar daca nu se doreşte, se apasă butonul NU.
5. Dacă se apasă butonul DA, se reia ciclul de la punctul 1, iar dacă se apasă butonul NU instrumentul se opreşte.

Mod de lucru

Organigrama instrumentului este prezentată în figura 7.2.

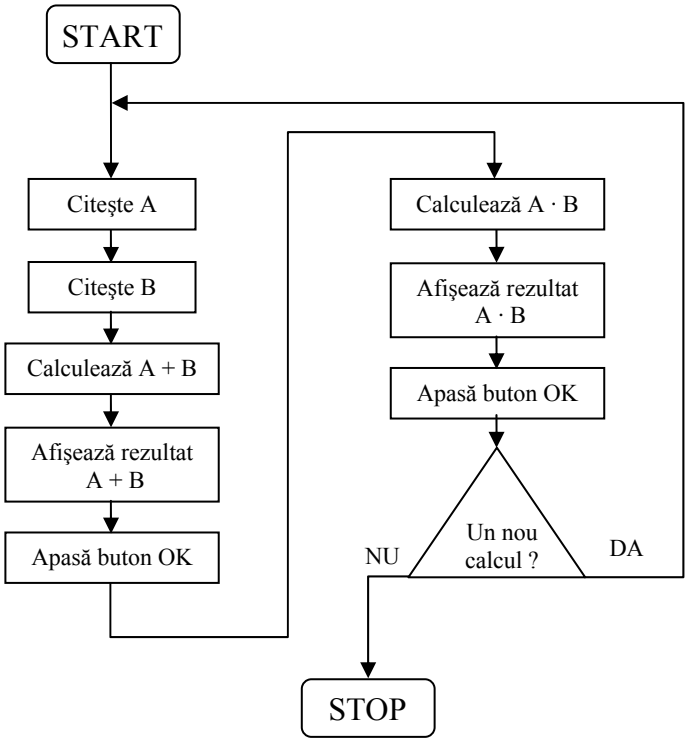


Figura 7.2

Acțiunile în cadrul instrumentului fiind secvențiale, vom utiliza o structură de tip secvență.

1. Construiți panoul frontal al IV-ului prin plasarea următoarelor elemente: două controale numerice etichetate *A* şi *B*, un indicator string etichetat *Mesaj* şi un buton boolean de tip *OK*.
2. Plasați pe DL o buclă WHILE.
3. In bucla WHILE plasați o structură SECVENȚĂ.
4. Introduceți în primul cadrul al secvenței operația de adunare şi afişarea mesajului, pe care îl realizați cu o funcție *Format Into String* (figura 7.3).
5. In cadrul al doilea vom introduce bucla de aşteptare pentru apăsarea butonului OK. Aceasta constă într-o buclă WHILE la a cărei terminal de condiționare se leagă butonul OK. Atâta timp cât butonul OK nu este apăsat, instrumentul stă în buclă, aşteptând apăsarea butonului OK.
6. Cadrul al 3-lea conține operația de efectuare a produsului (figura 7.4). Rezultatul produsului trebuie afişat în acelaşi indicator string, ca şi suma. Întrucât indicatorul a fost utilizat în primul cadru, acum vom utiliza o variabilă locală a sa, configurată în scriere.

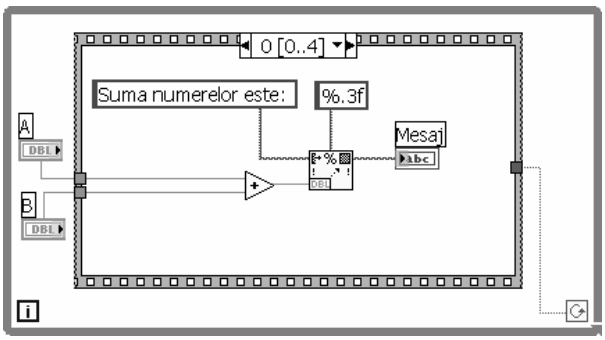


Figura 7.3

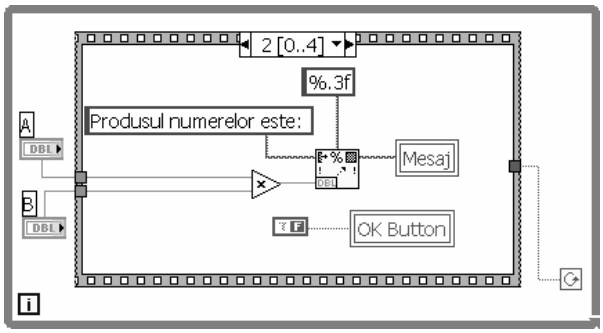


Figura 7.4

Deoarece butonul *OK* va avea nevoie şi el de o variabilă locală pentru cazul al 4-lea (când instrumentul trebuie să aştepte din nou apăsarea butonului *OK*) şi LabVIEW nu acceptă crearea de variabile locale pentru butoanele cu acţiune mecanică *latch*, vom stabili butonului *OK* acţiunea mecanică *Switch When Pressed* şi vom readuce în cadrul al treilea butonul la starea *false*. Schimbarea acţiunii mecanice a butonului se face din meniul său shortcut, opţiunea *Mechanical Action*.

7. Cadrul al 4-lea este identic cu cadrul al 2-lea, numai că butonul *OK* va fi reprezentat printr-o variabilă locală a sa (figura 7.5).

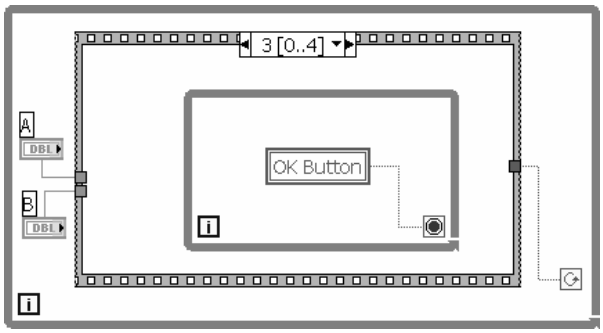


Figura 7.5

8. Pentru implementarea dialogului din cel de-al 5-lea cadru, vom utiliza funcția *Two Button Dialog* din subpaleta *Dialog & User Interface*, cum este prezentat în figura 7.6. Valoarea booleană furnizată la ieşirea acestei funcții o legăm la terminalul de condiționare al buclei mari. Este necesară readucerea pe *false* a butonului *OK*, pentru a-l pregăti pentru un nou calcul.

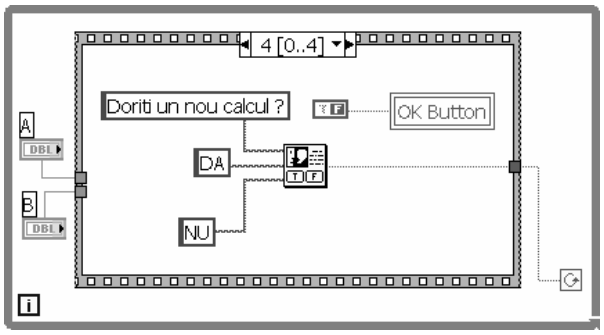


Figura 7.6

9. Rulați instrumentul şi verificați funcționarea.
10. Salvați sub numele *Ex 7.3.vi*.

Exerciții propuse

EP 7.1.
Să se realizeze un IV care are panoul frontal din figura 7.7 şi îndeplineşte următoarele funcții:

1. Generează numere aleatoare cuprinse între 10 şi 20, la intervale de timp egale cu 300 ms.
2. Dacă numărul generat depăşeşte valoarea Nmax, pe un indicator de tip *string* se afişează mesajul „S-a depăşit nivelul maxim" şi se aprinde LED-ul 1, celelalte fiind stinse.
3. Dacă numărul generat se află sub valoarea Nmin, pe indicatorul *string* se afişează mesajul „Sub nivelul minim" şi se aprinde LED-ul 2, celelalte fiind stinse.
4. Dacă valoarea numărului generat se află între Nmin şi Nmax, nu se afişează nici un mesaj şi se aprinde LED-ul 3, celelalte fiind stinse.

☞ Temporizarea de 300 ms se poate face utilizând funcția *Wait (ms)* din subpaleta de funcții *Timing*.

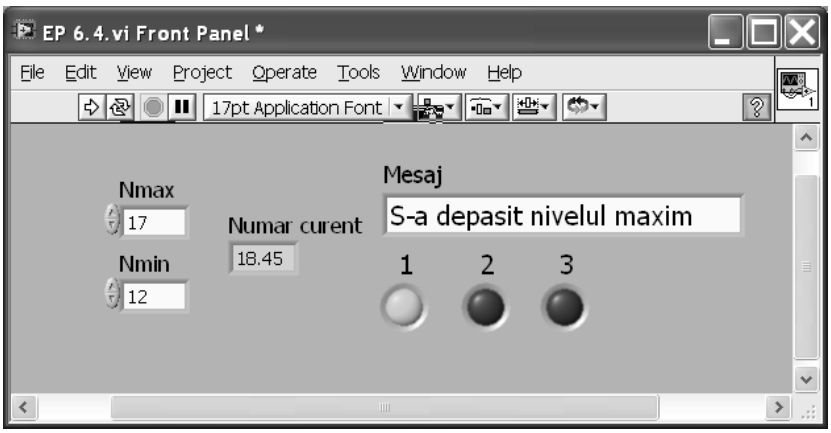


Figura 7.7

EP 7.2.

Să se realizeze un IV prin care să se introducă interactiv date într-un tabel în următoarea ordine: nume, prenume, anul nasterii, grupa sanguina, la apăsarea unui buton de pe PF.

La apăsarea butonului se deschid succesiv ferestre etichetate in care se cere introducerea datelor specificate. Fiecare fereastra are prevăzut un buton OK cu care se validează datele.

La terminarea unui rând de date se deschide o altă fereastră de dialog în care se întreabă dacă se doreşte continuarea cu un nou şir de date. Validarea se face cu două butoane: DA şi NU.

Datele sunt afişate în tabel în IVul principal pe linii. Coloanele au headere cu numele informației cerute: Nume, Prenume, etc.

Capitolul 8

NODURI DE PROPRIETĂȚI

Nodurile de proprietăți (*Property Node*) (NP) sunt noduri de pe DL pe care se pot modifica în mod programatic proprietăți ale unor obiecte LabVIEW. In acest capitol ne vom ocupa doar de nodurile de proprietăți ale controalelor şi indicatoarelor de pe PF.

Un NP pentru un control/indicator de pe PF se poate crea în două moduri:

- din meniul shortcut al controlului/indicatorului, se selectează *Create – Property Node*, după care se selectează proprietatea pe care dorim să o programăm. Prin această metodă se creează un NP strict pentru acel control, care poartă aceeaşi eticheta.
- se plasează pe DL un nod de proprietăți general din subpaleta de funcții *Application Control – Property Node*. Acestui NP trebuie să îi asociem un obiect printr-o referință. Pentru asocierea unui control/indicator de pe PF, acestuia i se va crea o referință din meniul său shortcut, opțiunea *Create – Reference*, referință ce va fi legată nodului de proprietăți la intrarea *Reference*.

Proprietățile care pot fi editate şi modificate diferă mult de la un obiect la altul. Fiecare control/indicator are proprietățile lui specifice. Pentru schimbarea unei proprietăți, este suficient să se facă un click pe proprietate, când se va deschide lista cu proprietățile disponibile. In helpul mic sunt date informații despre fiecare proprietate şi modul cum se foloseşte, prin simpla poziționare a mouse-ului pe acea proprietate. Se pot adăuga mai multe proprietăți unui NP, astfel: fie din meniul shortcut - *Add Element* fie prin extinderea dimensiunii întocmai ca la matrici, pornind de la punctul de redimensionare care apare la trecerea prompterului peste NP (figura 8.1).

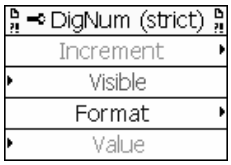


Figura 8.1

Proprietățile sunt executate în ordine, de sus în jos. Proprietățile pot fi scrise sau citite. Proprietățile care pot fi citite au o săgeată în partea dreaptă, iar cele care pot fi scrise au săgeata în partea stângă. In exemplul din figura 8.1, proprietățile *Increment* şi *Format* pot fi citite, iar proprietățile *Visible* şi *Value* pot fi înscrise. Trecerea de la modul citire la modul scriere şi invers se face din meniul shortcut *Change To Read (Change To Write)*.

Tipurile de date vehiculate de proprietăți diferă. De aceea, pentru scrierea sau citirea lor, se recomandă crearea automată a controalelor sau indicatoarelor corespunzătoare.

Exercițiul 8.1

Scop

Studiul nodurilor de proprietăți.

Mod de lucru

1. Deschideți un nou IV.
2. Plasați pe PF un control de tip *Numeric*.
3. Creați un nod de proprietăți pentru acest control, cu proprietatea *Visible*.
4. Creați un indicator pentru proprietatea *Visible*. Observați ce tip de dată s-a creat.
5. Rulați instrumentul şi observați indicatorul. Deoarece acesta se aprinde, înseamnă că controlul este vizibil.
6. Ştergeți indicatorul boolean.
7. Treceți proprietatea *Visible* în modul scriere.
8. Creați automat un control.
9. Rulați instrumentul cu *Run Continuously* şi apăsați pe controlul boolean. Observați apariția şi dispariția controlului numeric.
10. Adăugați şi alte proprietăți şi verificați funcționarea.
11. Adăugați pe PF şi alte tipuri de controale, creați-le noduri de proprietăți şi experimentați.

Exercițiul 8.2

Enunț

Să se realizeze un IV care să simuleze o sursă de tensiune continuă reglabilă şi un indicator de tip voltmetru, legat la această sursă.
Specificații tehnice:
- Domeniul de reglare al sursei este 0 – 100 V.

- Valoarea tensiunii se citeşte de pe un indicator de tip *meter* prevăzut cu comutator de game de măsură cu capetele de scală: 1 – 3 – 10 – 30 – 100 V.
- Indicatorul îşi schimbă automat capătul de scală în funcție de poziția comutatorului.
- Valorile cu care sunt inscripționate diviziunile semnificative ale scalei gradate au precizia de o zecimală pentru gamele de 1 V şi 3 V şi fără zecimale pe celelalte game.

Mod de lucru

1. Deschideți un nou IV.
2. Plasați pe PF un control numeric de tip *Knob*, din care vom regla tensiunea, un indicator numeric de tip *Meter* pe post de volmetru şi încă un control numeric de tip *Dial*, pe care-l vom configura ca selector de game pentru indicatorul *Meter,* ca în figura 8.2. Etichetați obiectele ca în figură.

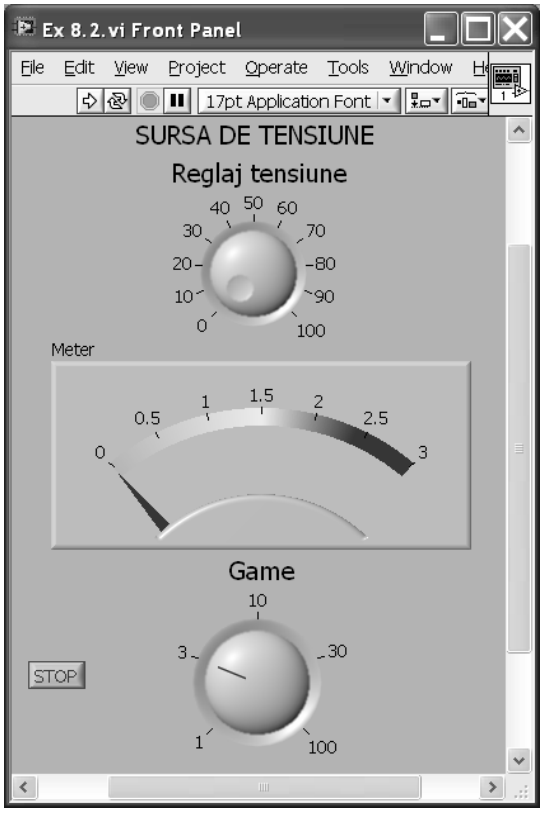


Figura 8.2

3. Stabiliți capătul de scală al controlului *Reglaj tensiune* la 100.
4. Treceți controlul *Game* în modul *Text Labels*, din meniul shortcut.

5. Deschideți fereastra de editare a textului din meniul shortcut al indicatorului text – *Edit Items*.
6. Scrieți în fereastră valorile capetelor de scală ale selectorul de game, ca în figura 8.3.
7. Plasați pe DL o buclă WHILE asistată de un buton STOP, în care vom încadra diagrama IV-ului.

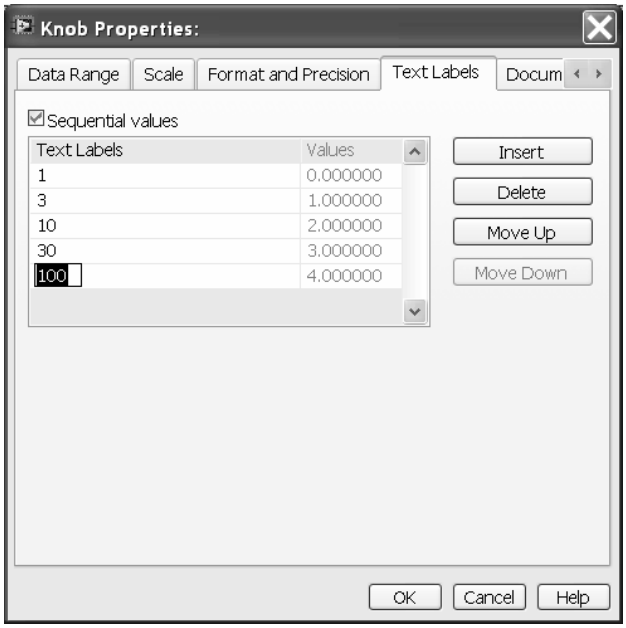


Figura 8.3

8. Faceți legătura dintre controlul *Reglaj tensiune* şi indicatorul *Meter* şi rulați instrumentul. Manevrați controlul. Observați necesitatea selectorului de game, deoarece pentru valori ale controlului mai mari decât 10 (cât este deocamdată capătul de scală al indicatorului *Meter*), acul indicator ajunge la capăt de scală.
9. Modificarea capetelor de scală al indicatorului *Meter* în funcție de poziția selectorului de game o vom face programatic, utilizând noduri de proprietăți. Creați un nod de proprietăți pentru indicatorul *Meter* în modul următor: *Create – Property Node – Scale – Range – Maximum.* (figura 8.4)

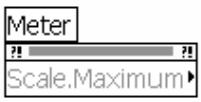


Figura 8.4

10. Treceți nodul în modul scriere (*Change To Write*).

11. In această proprietate vom înscrie valorile capetelor de scală, selectabile de către controlul *Game* printr-o structură de tip CASE. Fiecărui capăt de scală îi corespunde câte un cadru din structura CASE (figura 8.5).
12. Pentru rezolvarea ultimei condiții a problemei vom utiliza o proprietate prin care se stabileşte programatic formatul şi precizia afişajului. Pentru aceasta adăugați nodului de proprietăți al indicatorului *Meter* proprietățile *Scale – Format & Precision – Format* şi *Scale – Format & Precision – Precision*.
13. Legați la proprietatea *Format* valoarea 0 pentru toate cazurile (0 înseamnă reprezentarea numărului în sistem zecimal) şi la proprietatea *Precision* valorile 1 pentru cazurile corespunzătoare gamelor 1 şi 3 şi 0 pentru celelalte 3 cazuri.
14. Salvați instrumentul sub numele *Ex 8.2.vi*.
15. Rulați şi verificați funcționalitatea.

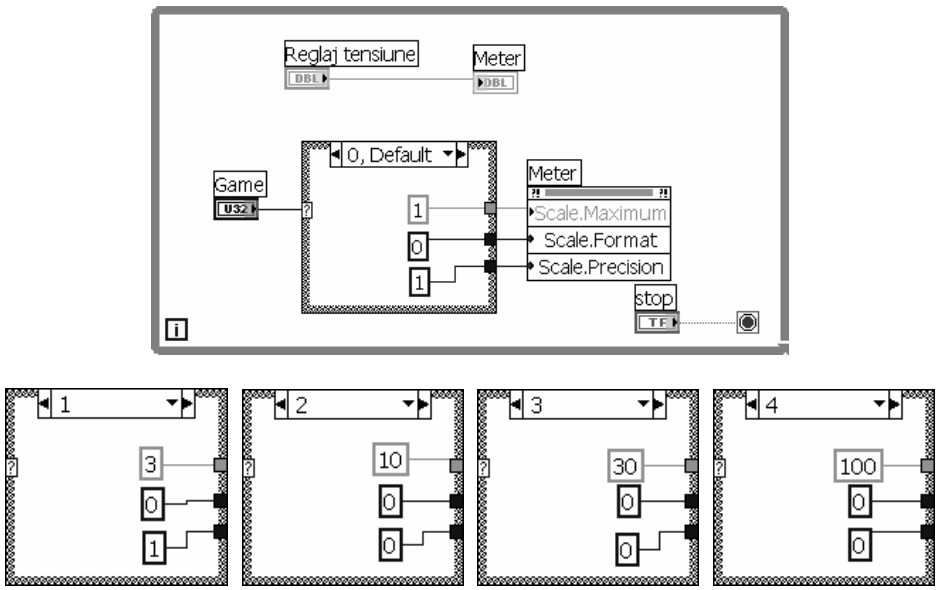


Figura 8.5

Capitolul 9

INDICATOARE GRAFICE

Indicatoarele grafice (IG) servesc la afişarea pe panoul frontal a curbelor, graficelor şi desenelor în mod bi şi tridimensional sau sub formă de paletă de culori. In LabVIEW există mai multe tipuri de indicatoare grafice în subpaleta de controale *Graph*. In cadrul acestui capitol ne vom ocupa doar de indicatoarele de tip *Waveform Graph, Waveform Chart* şi *XY Graph*.

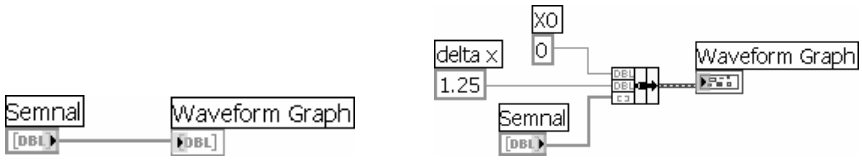
Waveform Graphs (WG)

Pe WG se pot afişa grafice de funcţii univariabile, y = f(x), la care valorile variabilei (punctele de pe abscisă) sunt egal distanţate (de exemplu un semnal eşantionat cu frecvenţă de eşantionare constantă).

Afişarea unui singur grafic pe un Waveform Graph

Graph-urile acceptă la intrare două tipuri de date:

a) *vectori numerici*, la care distanţa dintre punctele de pe abscisă este considerată 1 ($\Delta x = 1$) (figura 9.1a);
b) *cluster numeric* format din 3 elemente: valoarea iniţială a variabilei x_0, distanţa dintre două puncte adiacente de pe abscisă, Δx şi valorile funcţiei sub formă de vector (figura 9.1b).



a) b)

Figura 9.1

Afişarea de grafice multiple pe acelaşi Waveform Graph

Pentru afişarea mai multor grafice pe acelaşi indicator grafic, datele pot fi aduse la intrare sub forma:

a) unei matrici bidimensionale, în care fiecare linie reprezintă elementele câte unui grafic. Graficele au aceeaşi abscisă şi acelaşi număr de puncte, iar distanţa dintre punctele abscisei este $\Delta x = 1$, începând de la $x_0 = 0$. Este cazul afişarii formelor de undă achiziţionate cu o placă de achiziţii pe mai multe canale. In acest caz, punctele fiecărei forme de undă formează coloanele matricii şi este necesară o transpunere pentru afişarea corectă pe WG (figura 9.2).

Figura 9.2

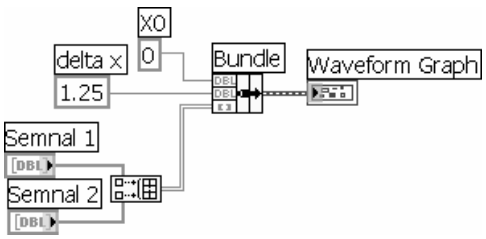
b) unor clustere, cu următoarele variante:

b1) cluster format din valoarea lui x_0, Δx şi matricea cu valorile semnalelor de afişat. In acest caz se pot modifica valoarea de început x_0 şi distanţa dintre puncte Δx, care sunt aceleaşi pentru toate graficele (figura 9.3a). Graficele trebuie să aibă acelaşi număr de puncte.

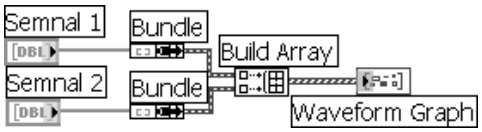
b2) vector de clustere, la care fiecare element este un cluster ce conţine vectorul cu valorile semnalului (figura 9.3b). In acest caz vectorii de intrare, deci graficele pot avea număr diferit de elemente, însă toate încep de la $x_0 = 0$ şi $\Delta x = 1$.

b3) cluster format din valorile lui x_0 şi Δx, comune pentru toate graficele, şi vectorul de clustere format ca la punctul b2) (figura 9.3c). Graficele pot avea număr diferit de elemente, dar toate au acelaşi x_0 şi Δx.

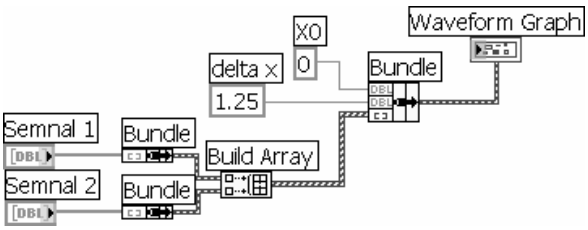
b4) vector de clustere construite ca la b1), ce reprezintă cazul cel mai general în care fiecare grafic are propriile x_0, Δx şi număr de elemente (figura 9.3d).



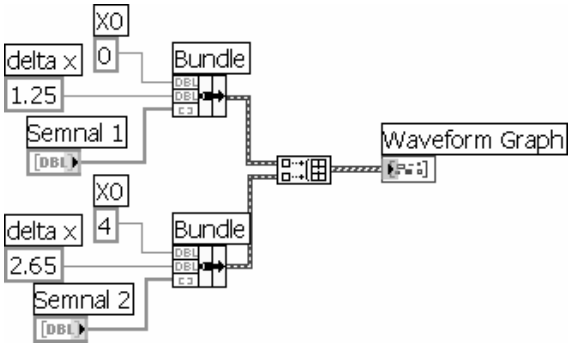
a)



b)



c)



d)

Figura 9.3

Waveform Charts (WCh)

WCh se comportă la fel ca şi WG, cu specificaţia că, pe lângă vectori sau matrici, aceste indicatoare acceptă la intrare şi scalari. Un scalar aplicat la intrare este afişat imediat pe grafic, permiţând astfel urmărirea în timp real a evoluţiei unui şir de date sau a unei forme de undă. In acest mod, WCh poate fi utilizat ca înregistrator virtual. Şirul de date va fi actualizat prin adăugare la datele afişate anterior. Distanţa dintre valorile absciselor a două puncte adiacente este $\Delta x = 1$. Datele anterioare sunt păstrate într-un buffer al cărui lungime poate fi prestabilită de către utilizator. Pe WCh pot fi vizualizate datele anterioare prin derularea acestui buffer. Pentru aceasta este disponibil un scroll-bar.

Afişarea unui singur grafic pe un Waveform Chart

Datele pot fi aduse la WCh fie sub formă de vector, ca la WG, când toate vor fi afişate odată, sau sub formă de scalar, când vor fi adăugate punct cu punct graficului anterior. De regulă, afişarea punct cu punct se face în cadrul unui proces

iterativ, care implică bucle FOR sau WHILE.

In exemplul din figura 9.4, pe WCh se afişează punct cu punct restul împărţirii indexului buclei WHILE la 10, rezultând un grafic sub formă de dinţi de ferăstrău. Punctele se acumulează în buffer şi pot fi vizualizate prin derularea bufferului cu ajutorul scroll-barului.

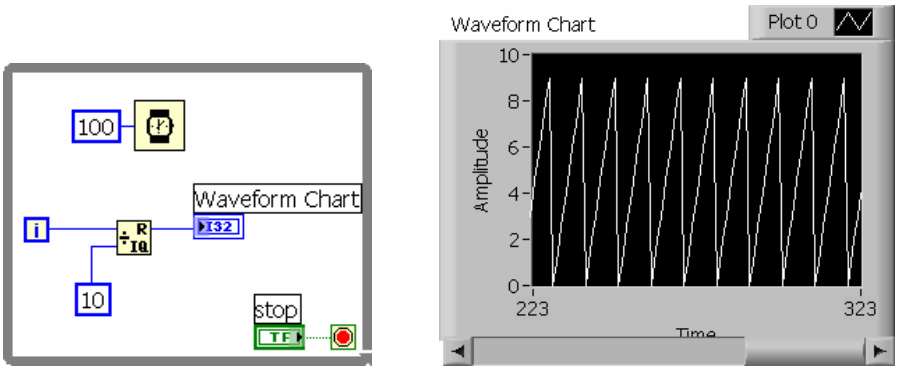


Figura 9.4

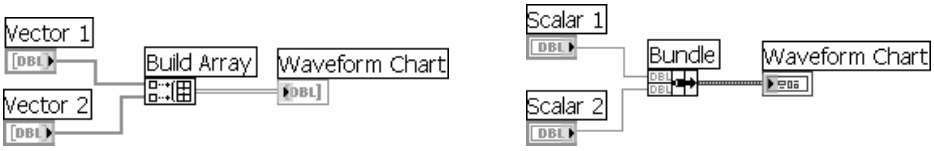
Mărimea bufferului poate fi ajustată din meniul shortcut, opţiunea *Chart History Length…* Aceasta este dată în numărul maxim de puncte înregistrate.

Conţinutul bufferului poate fi şters din meniul shortcut, opţiunea *Data Operations – Clear Chart*.

Afişarea de grafice multiple pe acelaşi Waveform Chart

Dacă graficele sunt sub formă de vectori care trebuie afişaţi concomitent pe acelaşi WCh, cu aceştia se construieşte utilizând funcţia *Build Array* o matrice bidimensională ce va fi aplicată indicatorului (figura 9.5a).

Dacă graficele sunt construite punct cu punct, scalarii fiecărui grafic se înmănunchează într-un cluster, după care se aplică indicatorului (figura 9.5b).



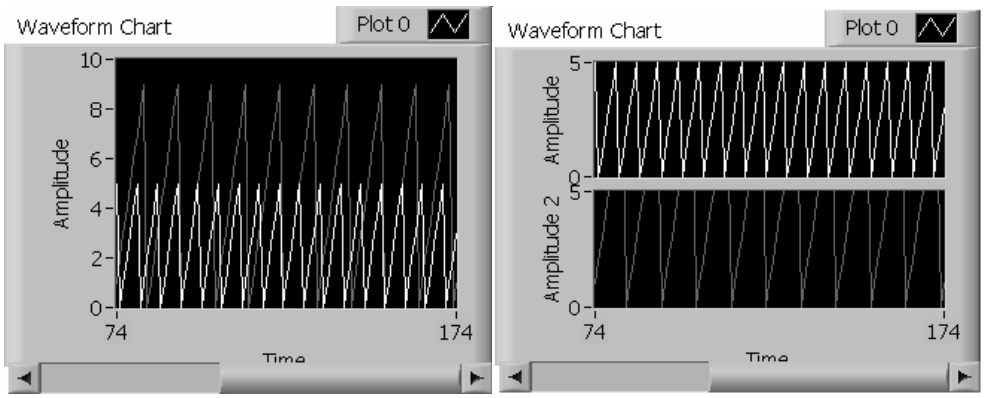
a) b)

Figura 9.5

In cazul în care se afişează mai multe grafice pe aceeaşi suprafaţă grafică a unui WCh, acestea pot fi: a) suprapuse sau b) stivuite.

Graficele suprapuse se reprezintă pe aceeaşi suprafaţă grafică, accesând opţiunea *Overlay Plots* din meniul shortcut al indicatorului (figura 9.6a). Graficele stivuite se reprezintă pe suprafeţe grafice diferite, accesându-se cu opţiunea *Stack Plots* din meniul shortcut al indicatorului (figura 9.6b).

Aceste două opţiuni sunt valabile numai când trasarea graficelor se face punct cu punct (la intrare se aplică clustere de scalari).



a) b)

Figura 9.6

Moduri de actualizare a graficelor pe un Waveform Chart

Actualizarea graficelor pe un WCh se poate face în 3 moduri, după cum se selectează din meniul shortcut, opţiunea *Update Mode.* Cele 3 moduri sunt disponibile doar la afişarea punct cu punct, în timp ce instrumentul funcţionează. Dacă instrumentul se opreşte, opţiunea dispare din meniul shortcut. Cele 3 moduri de actualizare sunt (figura 9.7)*:*

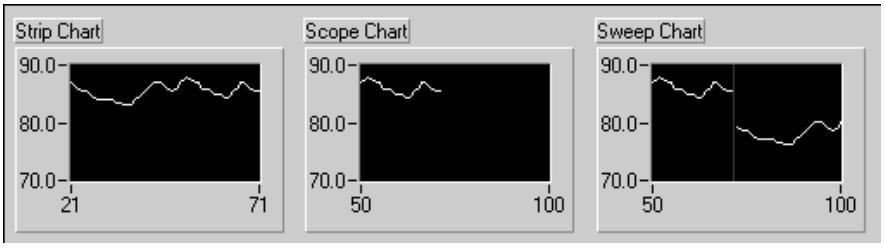


Figura 9.7

- ***Strip Chart*** – graficul se desfăşoară continuu de la stânga spre dreapta, pe măsură ce se adaugă noi puncte.

- ***Scope Chart*** – graficul este desenat progresiv, ca pe un osciloscop. După baleierea completă de la stânga la dreapta a ecranului, graficul este şters şi se reia afişarea. începând cu primul punct din stânga.
- ***Sweep Chart*** – permite vizualizarea şi a vechiului grafic (de partea dreaptă a unei linii verticale ce baleiază ecranul) şi a celui nou (de partea stângă a liniei), întocmai ca un EKG.

Schimbarea atributelor indicatoarelor grafice

Schimbarea atributelor indicatoarelor grafice Waveform Graph şi Waveform Chart se face utilizând opțiunile ce pot fi vizualizate sau ascunse din meniul shortcut, *Visible Items*. Aceste opțiuni deschid următoarele subpalete şi legende prin care se stabilesc atributele indicatoarelor:

- plot legend
- scale legend
- graph palette
- cursor palette (numai la graph)
- scrollbar (numai la chart)

1. Plot legend
Utilizând această opțiune, se pot modifica atributele graficelor legate de tipul şi forma punctelor, a liniilor, culoarea, modul de interpolare între puncte, etc. Denumirea graficului poate fi schimbată cu unealta text.

2. Scale legend
Se pot stabili atribute legate de scalele x şi y, cum ar fi: formatul şi precizia scalelor, dacă să fie vizibile sau nu, autoscalarea, forma, culoarea şi tipul gridurilor, etc. Denumirile scalelor se modifică cu unealta text.

3. Graph palette
Conține unelte de mărire sau micşorare (zoom), de manevrare a graficului pe toată suprafața şi de manevrare a cursoarelor. Cursoarele sunt disponibile numai la WG şi se definesc cu *Cursor palette.*

4. Cursor palette (numai la Waveform Graph)
De pe această paletă se stabilesc forma, dimensiunile, culoarea, modul de manevrare a cursoarelor şi se pot realiza măsurători pe grafic. Cursorul indică în orice moment coordonatele punctului în care se află.

5. Scrollbar (numai la Waveform Chart)
Această opțiune permite vizualizarea datelor anterioare stocate în bufferul WCh, atunci când acestea nu pot fi afişate în întregime pe suprafața grafică disponibilă.

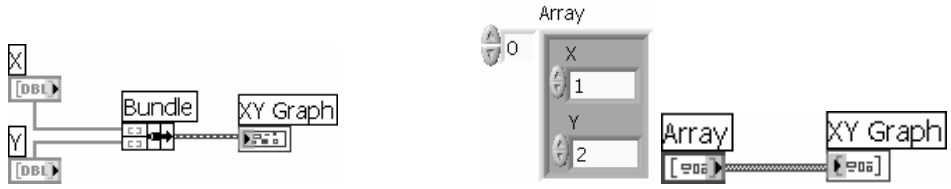
XY Graphs (XYG)

Aceste indicatoare sunt tot forme de graph-uri la care datele sunt specificate prin perechi de puncte ce reprezintă coordonatele carteziene din planul de afişare grafică. Se pot desena astfel orice forme geometrice plane, la care punctele nu trebuie să fie neapărat egal distanţate, ca la WG sau ca la WCh.

Afişarea unui singur grafic pe un XY Graph

Pentru afişarea unui singur grafic, un XYG aceeptă la intrare datele prezentate sub două forme:

a) un cluster cu două elemente: unul este vectorul valorilor lui X, iar celălalt vectorul valorilor lui Y (figura 9.8a). Cei doi vectori pot avea număr diferit de elemente, dar numărul de puncte afişate este dat de vectorul cu cele mai puţine elemente.
b) un vector de clustere, fiecare element al vectorului fiind câte un cluster ce conţine două controale numerice ce reprezintă coordonatele carteziene ale punctului (figura 9.8b).



a) b)

Figura 9.8

Afişarea de grafice multiple pe acelaşi XY Graph

Pentru afişarea mai multor grafice pe acelaşi XYG, acestea se grupează într-un vector după care se aplică indicatorului, ca în figura 9.9. Vectorii pot avea orice număr de elemente. Pentru fiecare grafic, numărul de elemente este dat de vectorul cel mai mic.

Exerciţiul 9.1

Enunţ

Să se genereze şi să se afişeze grafic doi vectori formaţi din valorile funcţiei sinus, respectiv cosinus, pe un număr de perioade selectate de pe panoul frontal. Pe PF se mai dau numărul total de puncte şi amplitudinea funcţiilor.

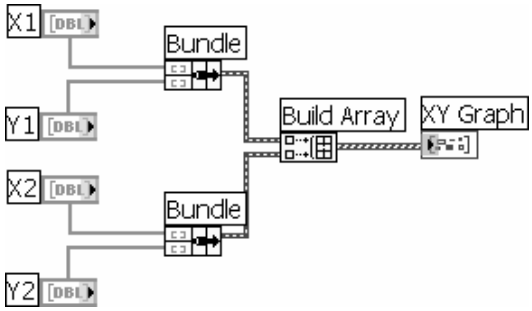


Figura 9.9

☞ Cele două funcţii eşantionate sunt date de următoarele relaţii:

$$y1(i) = A1\sin\left(2\pi\frac{N_{p1}}{N}i\right)$$
$$y2(i) = A2\cos\left(2\pi\frac{N_{p2}}{N}i\right) \qquad (9.1)$$

unde N este numărul total de puncte, Np1 şi Np2 reprezintă numărul de perioade, iar A1 şi A2 sunt amplitudinile funcţiilor sinus, respectiv cosinus.

Mod de lucru

1. Deschideţi un nou IV.
2. Plasaţi pe PF controalele de fixare a numărului de eşantioane (numeric I32), a numărului de perioade (dbl) şi a amplitudinilor celor două funcţii (dbl), precum şi un indicator de tip *Waveform Graph*, ca în figura 9.10.

Vom utiliza două moduri de a construi funcţiile *y1* şi *y2*: cu ajutorul nodurilor de formule şi utilizând generatorul de funcţii din biblioteca LabVIEW.

Utilizarea nodului de formule

Generarea funcţiilor se va face prin calculul valorilor *y1* şi *y2*, punct cu punct, în cadrul unei bucle FOR la care elementul de indexare *i* al formulelor este chiar indexul buclei.

3. Construiţi cele două funcţii cu ajutorul unei structuri *Formula Node*, ca în figura 9.11. Formele de undă se obţin prin autoindexarea la ieşirea buclei FOR a rezultatelor *y1* şi *y2*.

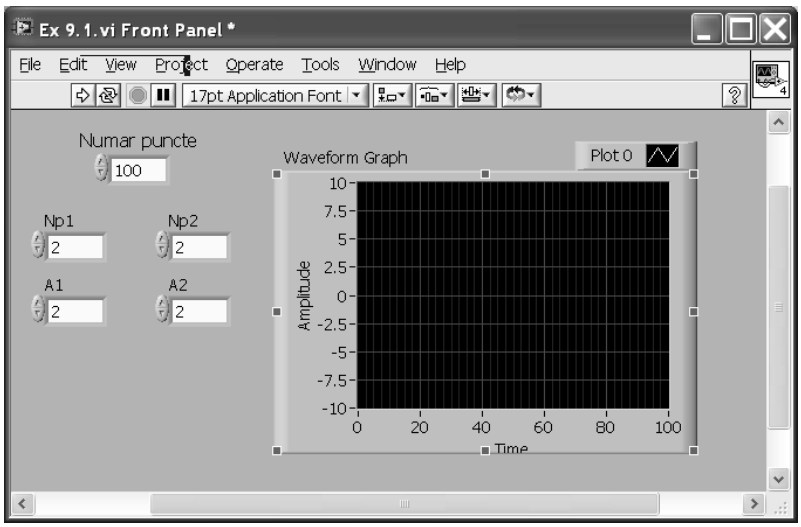


Figura 9.10

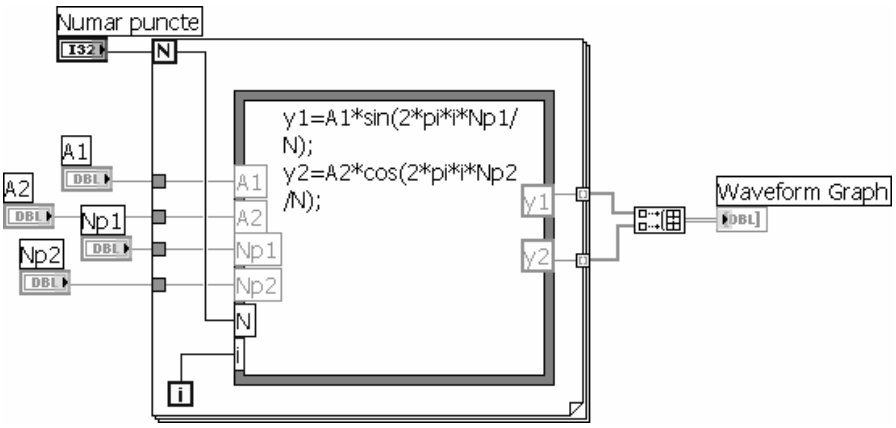


Figura 9.11

4. Daţi valori parametrilor de intrare şi rulaţi IV-ul. Veţi obţine pe indicatorul grafic cele două curbe, care au pe abscisă indexul fiecărui punct de pe grafic (în exemplul din figura 9.12 parametrii au fost următorii: N = 100, Np1 = 2, Np2 = 3, A1 = 2, A2 = 3).
5. Deschideţi meniul shortcut al indicatorului *Waveform Graph* şi studiaţi-i posibilităţile de modificare a atributelor din *Visible Items*.
6. Ne propunem acum ca, pe abscisă, în loc de indexul punctelor graficelor, să avem valori de unghi exprimate în grade.

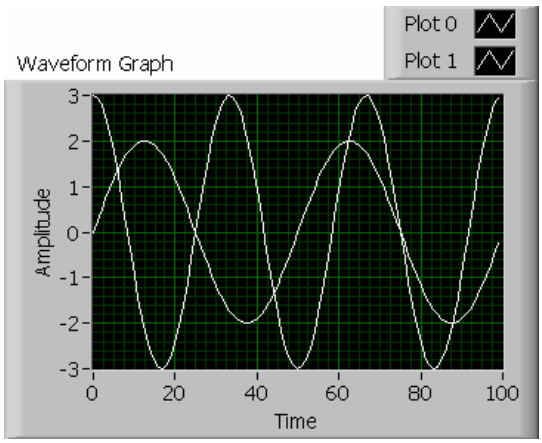


Figura 9.12

Pentru aceasta, vom utiliza modul b4) de la secțiunea **Afişarea de grafice multiple pe acelaşi Waveform Graph,** în care distanța dintre două valori adiacente de pe abscisă (cuanta de unghi), Δφ, este dată de transformarea în grade:

$$\begin{aligned} deltafi1 &= 360 * Np1 / N \\ deltafi2 &= 360 * Np2 / N \end{aligned} \qquad (9.2)$$

Cele două relații de mai sus ce definesc cuanta de unghi le calculăm de asemenea în cadrul nodului de formule. Diagrama finală a IV-ului este prezentată în figura 9.13.

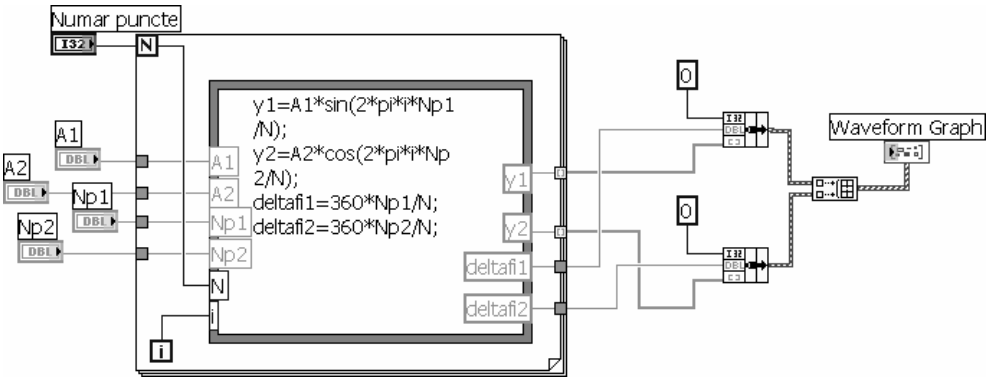


Figura 9.13

7. Rulați acum instrumentul şi observați rezultatul. Fiecare punct de pe grafic are ca abscisă un unghi, iar ca ordonată valoarea sinusului, respectiv a cosinusului corespunzător.

8. Modificați eticheta abscisei din *Time* în *Grade*. Folosiți pentru aceasta unealta *text.*
9. Deschideți legenda de cursoare (*Visible Items – Cursor Legend*) şi creați două cursoare (MD pe legendă – *Create Cursor – Single Plot*).
10. Mişcați cursoarele câte unul pe fiecare curbă (inițial cursoarele se găsesc în punctul de abscisă 0) şi urmăriți coordonatele punctelor.
11. Adăugați pe PF un *Waveform Chart*.
12. Realizați pe DL legăturile pentru afişarea concomitentă a ambelor grafice pe WCh, în afara buclei FOR (figura 9.14).

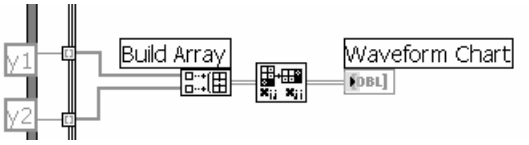


Figura 9.14

Observați necesitatea intercalării unei funcții de transpunere a matricii de intrare în WCh, întrucât WCh citeşte datele pe coloane, iar cele două funcții, *y1* şi *y2*, sunt dispuse după liniile matricii.

13. Rulați instrumentul de mai multe ori, cu diverse valori de intrare şi observați acumularea punctelor în istoricul WCh.
14. Adăugați scrollbarul (*Visible Items – X Scrollbar*) şi derulați informația anterioară.
15. Modificați lungimea bufferului de stocare istoric din meniul shortcut, *Chart History Kength*.
16. Mutați WCh în interiorul buclei FOR, pentru a-l utiliza cu funcția de înregistrator punct cu punct. Realizați legăturile conform figurii 9.15.

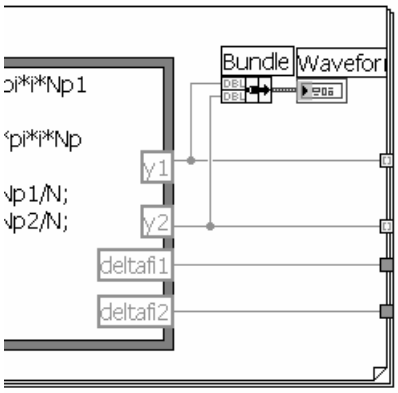


Figura 9.15

17. Adăugați în buclă un temporizator de tipul *Wait(ms)* din subpaleta de funcții *Time*. Fixați timpul de aşteptare la 300 ms.
18. Rulați instrumentul şi observați funcționarea.
19. Experimentați opțiunile *Stack Plots* şi *Overlay Plots* din meniul shortcut.
20. Utilizați cele 3 moduri de actualizare a graficelor, pe care le găsiți în opțiunea *Update Mode* din meniul shortcut. Opțiunea există doar când instrumentul funcționează.
21. Salvați IV-ul sub numele *Ex 9.1a.vi*.

Utilizarea generatorului de funcții

Aceleaşi grafice pot fi construite utilizând generatorul de funcții *Sine Wave.vi* din biblioteca LabVIEW. Acesta se află în subpaleta *Signal Processing – Signal Generation*. In figura 9.16 este prezentată pictograma funcției *Sine Wave*.

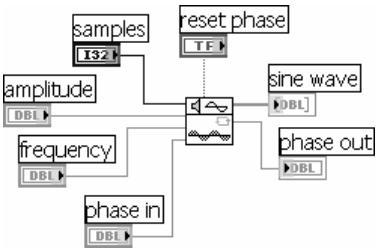


Figura 9.16

Semnificația terminalelor este următoarea:

Samples – numărul de eşantioane

Amplitude – amplitudinea formei de undă

Frequency – frecvența normalizată, dată de relația:

$$f_n = \frac{frecventa\ semnal}{frecventa\ de\ esantionare} = \frac{numar\ perioade}{numar\ total\ de\ esantioane} = \frac{1}{numar\ esantioane\ pe\ perioada} \tag{9.3}$$

Phase in – faza inițială în grade

Reset phase – determină faza inițială a formei de undă. Dacă *Reset phase* este *true*, atunci faza inițială este *phase in*. Dacă *Reset phase* este *false*, atunci faza inițială este setată la valoarea *phase aut* a ultimei rulări a funcției.

Phase out – faza ultimului eşantion al şirului, în grade.

Sine wave – vectorul de materializează forma de undă.

Nu este necesară includerea funcției într-o buclă, deoarece ea furnizează la ieşire direct forma de undă printr-un vector.

22. Utilizând aceleaşi date de intrare, construiţi formele de undă *y1* şi *y2* din ecuaţiile 9.1 cu ajutorul generatorul de funcţii de mai sus. O soluţie este dată în figura 9.17.

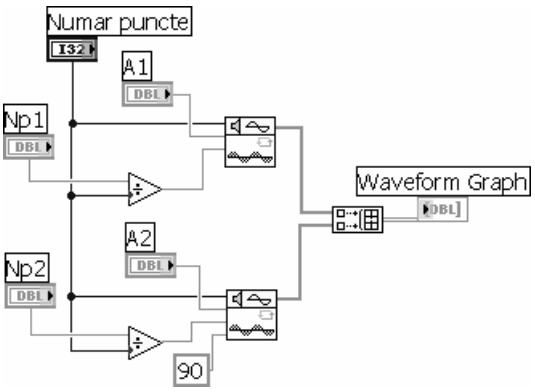


Figura 9.17

23. Experimentaţi afişarea pe abscisa WG a fazei fiecărui punct generat.
24. Inlocuiţi WG cu WCh. Observaţi că în acest caz nu se mai poate face înregistrarea punct cu punct a formei de undă.
25. Salvaţi sub titlul *Ex 9.1b.vi*.
26. Studiaţi şi alte funcţii din subpaleta *Signal Generation*, citiţi-le helpul şi experimentaţi-le funcţionarea.

Exerciţiul 9.2

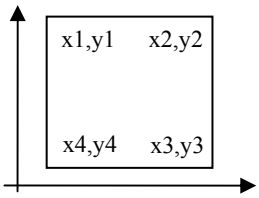
Enunţ

Să se realizeze un IV care să deseneze în mod programatic, pe un XY Graph, un pătrat căruia i se dau pe PF latura *L* şi coordonatele colţului din stânga sus, *x1* şi *y1* (figura 9.18).

Mod de lucru

Dacă (x1, y1) sunt coordonatele colţului din stânga sus şi L este latura pătratului, atunci celelalte coordonate se vor calcula cu relaţiile:

$$x2 = x1 + L \quad x3 = x2 \quad x4 = x1$$
$$y2 = y1 \quad y3 = y1 - L \quad y4 = y3 \qquad (9.4)$$



Astfel, vom crea un vector ce conţine cele 4 abscise, plus al cincilea element egal cu primul element, şi alte vector format din ordonatele vârfurilor, plus al cincilea element care este egal cu primul. Cel de-al cincilea element se adaugă deoarece, pe XYG, laturile pătratului se obţin prin unirea cu linii drepte a punctelor plasate pe

suprafața grafică, cel de-al cincilea punct fiind necesar pentru trasarea celei de a patra laturi.

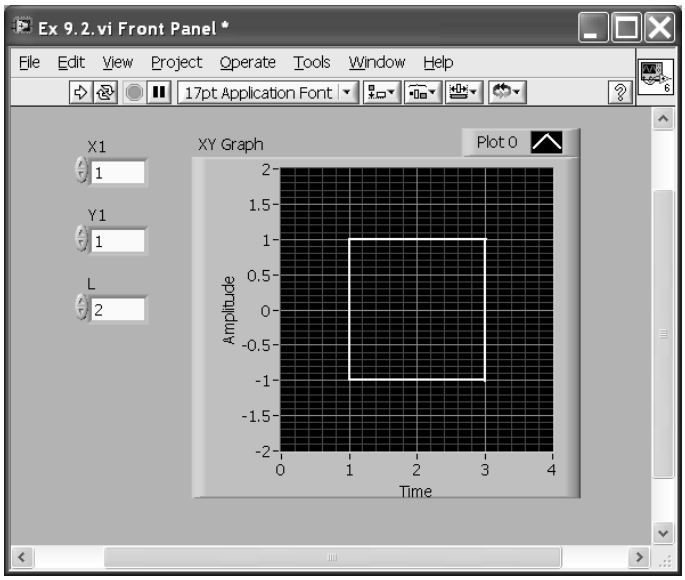


Figura 9.18

1. Utilizând un nod de formule, construiți cei doi vectori utilizând relațiile (9.4).
2. Dați valori coordonatelor colțului şi laturii şi rulați instrumentul.
3. Salvați IV-ul sub titlul *Ex 9.2.vi*.

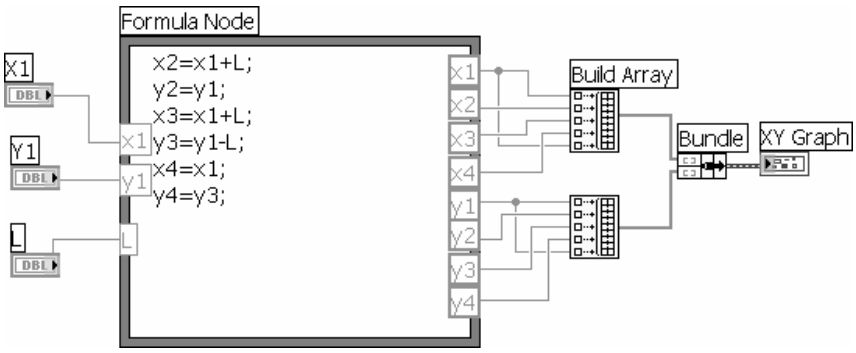


Figura 9.19

Figuri geometrice, precum şi desene complexe se pot realiza utilizând funcțiile specifice de desenare din subpaleta de funcții *Graphics & Sound – Picture Functions*. Desenarea se face pe un indicator grafic special al cărui suprafață este împărțită în pixeli, coordonatele obiectelor desenate fiind date în pixeli. Colțul din stânga – sus reprezintă originea.

Exerciții propuse

EP 9.1.
După modelul de la exercițiul 9.2, desenați pe suprafața unui WY Graph un triunghi echilateral, căruia i se dau de pe panoul frontal latura şi coordonatele unuia dintre vârfuri. Încercați să realizați triunghiul şi utilizând funcțiile din biblioteca *Graphics & Sound – Picture Functions*.

EP 9.2.
Să se realizeze un IV care să deseneze pe un WY Graph un cerc căruia i se specifică raza şi coordonatele centrului.

☞ Coordonatele unui punct de pe cerc sunt date de relațiile:

$$\begin{aligned} x &= x_0 + r\cos\alpha \\ y &= y_0 + r\sin\alpha \end{aligned} \tag{9.5}$$

EP 9.3.
Să se realizeze un instrument virtual care să îndeplinească următoarele cerințe:

1. Generează, la alegere, următoarele tipuri de funcții: sinusoidal, triunghiular, dreptunghiular, rampă, selectabil dintr-un control de pe PF. De pe PF se mai stabilesc: frecvența semnalului în Hz, frecvența de eşantionare în Hz, numărul de perioade, faza inițială, amplitudinea, offset-ul (componenta continuă). La funcția dreptunghiulară se va specifica şi factorul de umplere (duty cycle %).
2. Pe PF va fi un LED care se aprinde dacă nu este îndeplinită condiția Nyquist şi se afişează mesajul: "Conditia Nyquist nu este îndeplinită. Ajustați f sau fe."
3. Indicatorul grafic va avea pe abscisă faza, exprimată în grade (figura 9.20).

☞ Utilizați generatoarele din subpaleta *Signal Generation*.

☞ Pentru calculul parametrilor de intrare utilizați relațiile (9.3) şi relația:

$$Numarul\ total\ de\ esantionae = \frac{Numar\ perioade}{Frecventa\ normalizata} \tag{9.6}$$

EP 9.4.
Să se realizeze un instrument virtual care să calculeze şi să traseze pe acelaşi grafic funcțiile:

$$y = x^3 + mx^2 + 1;$$
$$y = ax + b$$

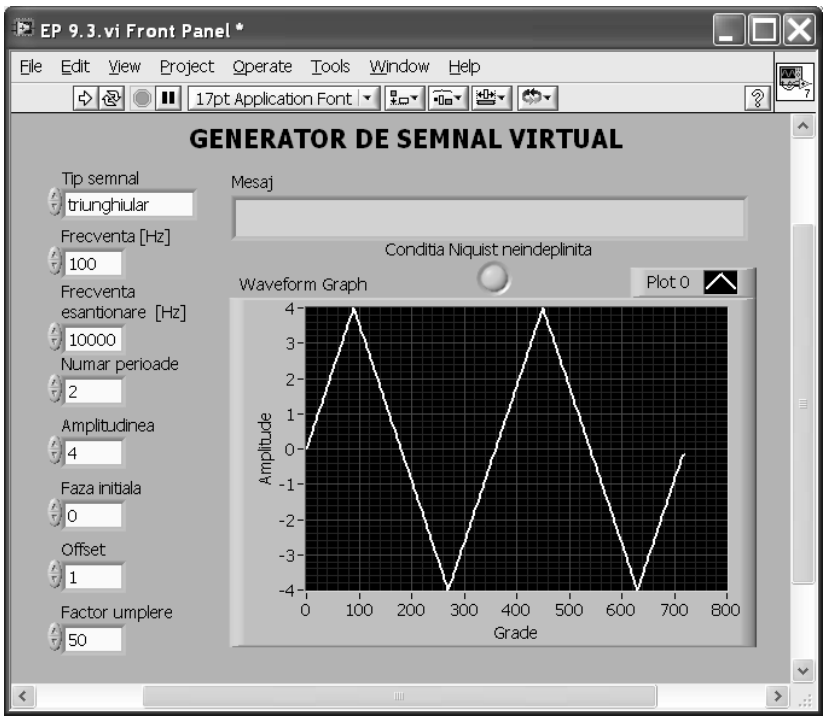


Figura 9.20

pentru *x* număr întreg cuprins între două valori, *Nmin* şi *Nmax*, stabilite de pe panoul frontal. De pe PF se mai stabilesc de asemenea valorile coeficienţilor *m, a* şi *b*. Trasarea graficelor se va face atât pe un WG cât şi pe un WCh.

EP 9.5.
Să se realizeze un IV care să rotească pe suprafaţa grafică a unui XY Graph un segment de dreaptă de lungime fixă în jurul unuia din capete. Se dau de pe PF: lungimea segmentului, coordonatele capătului fix, numărul de puncte al cercului după care se face rotaţia capătului liber, viteza de rotaţie şi sensul de rotaţie.

EP 9.6.
Aceeaşi problemă ca mai sus, numai că segmentul se roteşte în jurul mijlocului său.

EP 9.7.
Să se construiască un IV care să rotească un cub de latură dată în jurul axei sale verticale, cu o viteză dată. Aceeaşi problemă pentru rotirea în jurul unei diagonale.

Capitolul 10

SALVAREA DATELOR ÎN LABVIEW

Odată achiziționate şi prelucrate, datele obținute din măsurare trebuie stocate. Acest lucru se realizează utilizând funcțiile de lucru cu fişierele, cu ajutorul cărora se poate scrie sau citi în şi din fişiere. O sesiune de lucru cu fişiere implică următoarele operații principale:

1. Crearea sau deschiderea fişierului, în urma căreia LabVIEW furnizează un număr de referință (*refnum*) care va fi utilizat în operațiile ulterioare.
2. Scrierea în sau citirea din fişier.
3. Închiderea fişierului.

Funcțiile de lucru cu fişierele se găsesc în subpaleta *File I/O* . Pe lângă citirea şi scrierea fişierelor, în LabVIEW se pot crea, muta sau redenumi fişiere sau directoare, crea fişiere speciale de forma *spreadsheet* sau în format ASCII, scrie şi citi date în format binar, etc.

Datele pot fi stocate sau citite din fişiere în următoarele formate:

Formate generale

- Binar (Binary Byte Stream) – reprezintă cea mai compactă şi rapidă metodă de stocare a datelor. Datele trebuie convertite în prealabil în şiruri binare de caractere şi trebuie cunoscut exact ce tip de date au fost salvate pentru a putea fi citite ulterior.
- ASCII (ASCII byte stream) – se salvează când se doreşte utilizarea datelor în procesoare de text sau alte programe ce acceptă astfel de format (Matlab, C/C++, Microsoft Office, etc.). Pentru stocarea în astfel de format, datele trebuie să fie convertite ASCII.

Formate specifice

- LVM (LabVIEW measurement data file) este tot un format ASCII, însă fişierul creat prezintă un preambul conținând o serie de informații legate de parametrii măsurării, numele operatorul, data, ora, locul măsurării, etc. Citirea acestor fişiere, ca şi scrierea, se face cu funcții LabVIEW din subpaleta *File I/O*.

- TDM este un format binar conceput special pentru produsele National Instruments. Formatul constă în crearea a două fişiere separate: o secţiune XML conţinând atributele datelor înregistrate, şi un fişiere cu formele de undă în binar.
  In continuare ne vom ocupa doar de manipularea fişierelor în format ASCII.

Funcţii pentru lucrul cu fişierele

In subpaleta *File I/O* există două tipuri de funcţii pentru lucrul cu fişierele în LabVIEW: funcţii de nivel înalt şi funcţii de nivel scăzut.

Funcţii de nivel înalt

Funcţiile de nivel înalt sunt: *Write To Spreadsheet File*, *Read From Spreadsheet File*, *Write To Measurement File* şi *Read From Measurement File*.

Formatul *spreadsheet* se referă la modul în care sunt înscrise într-un fişier text elementele unui vector sau ale unei matrici cu două dimensiuni. Liniile matricii reprezintă linii în fişierul text, iar elementele, sub formă de şiruri de caractere, sunt despărţite printr-un *tab*. Liniile sunt despărţite între ele prin caracterul *end of line*.

- ***Write To Spreadsheet File*** converteşte un vector sau o matrice de numere în formatul text *spreadsheet*, pe care îl înscrie într-un fişier nou sau îl ataşează la un fişier existent. Funcţia poate face, la cerere, transpunerea matricii. Funcţia creează sau deschide automat fişierul, scrie în el, apoi îl închide. Dacă nu se specifică calea şi numele fişierului, se deschide automat o fereastră interactivă în care se stabilesc aceste date.
- ***Read From Spreadsheet File*** citeşte un anumit număr de linii dintr-un fişier *spreadsheet* începând cu un offset şi converteşte datele citite într-o matrice bidimensională. Funcţia deschide fişierul, citeşte datele şi la final îl închide.
- ***Write To Measurement File*** este o funcţie la care introducerea parametrilor se face printr-o fereastră interactivă. Acest gen de funcţii se numesc funcţii „expres". Funcţia scrie într-un fişier text de tip LVM sau binar de tip TDM. Din fereastră se pot seta modul de deschidere sau creare a fişierului, formatul, informaţii despre header, delimitatorul. La fel ca şi celelalte funcţii de nivel înalt, şi aceasta deschide (creează) fişierul, scrie în el şi apoi îl închide.
- ***Read From Measurement File*** citeşte din fişierele de tip LVM sau TDM. Fixarea parametrilor de citire se face de asemenea printr-o fereastră interactivă.

Nu este recomandabil să se folosească funcţiile de nivel înalt pentru scrieri/citiri repetitive, deoarece operaţiile de deschidere/închidere fişiere repetate solicită mult resursele calculatorului. Pentru aceasta este bine să se utilizeze funcţiile de nivel scăzut.

La operațiile cu fișiere este recomandabil să se încadreze funcțiile de scriere/citire sau chiar întreaga diagramă a instrumentului într-o buclă WHILE asistată de un buton de STOP, deoarece lanțul de operații *deschidere/creare fișier – scriere/citire – închidere fișier* trebuie să se efectueze complet. Dacă se utilizează butonul *Abort Execution* de pe bara de butoane rapide, instrumentul se oprește brusc în orice etapă a execuției și e posibil ca fișierele să rămână deschise.

Exercițiul 10.1

Enunț

a) Să se genereze și să se afișeze grafic, în mod continuu, doi vectori formați din valorile funcției sinus, respectiv cosinus, pe un număr de perioade selectate de pe panoul frontal. Pe PF se mai dau numărul total de puncte și amplitudinea funcțiilor.
b) La apăsarea unui buton SALVARE de pe PF, să se salveze cei doi vectori fie în format *spreadsheet*, fie în format *LVM*, la alegere dintr-un selector de pe PF. Salvarea se va face cu două cifre zecimale, pe coloane, iar la salvări repetate datele se vor adăuga la fișierul creat.
c) Instrumentul se oprește de la un buton de stop.

Mod de lucru

Panoul frontal al instrumentului este dat în figura 10.1.

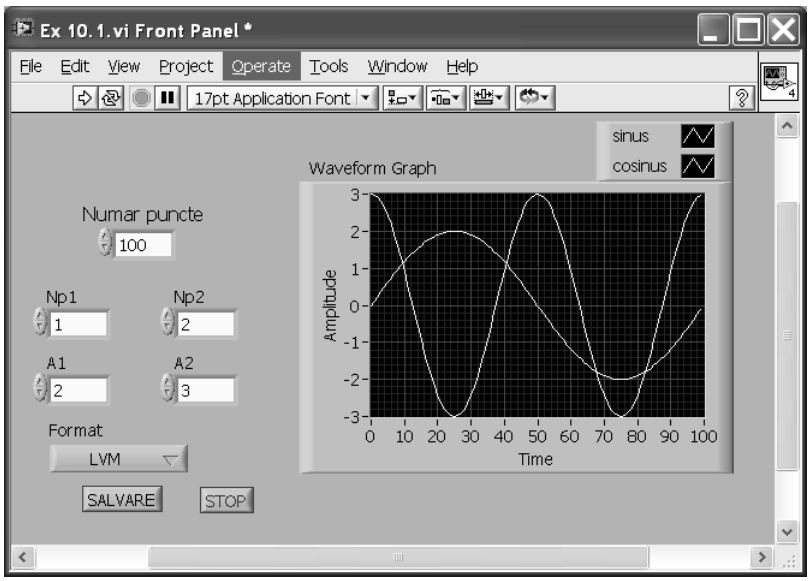
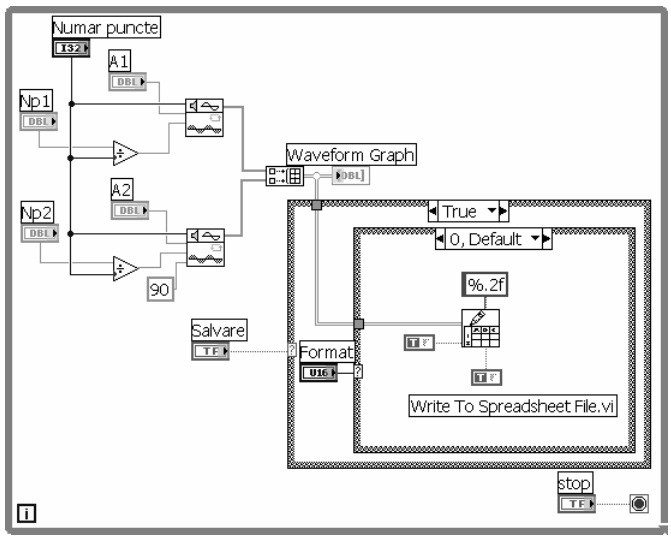
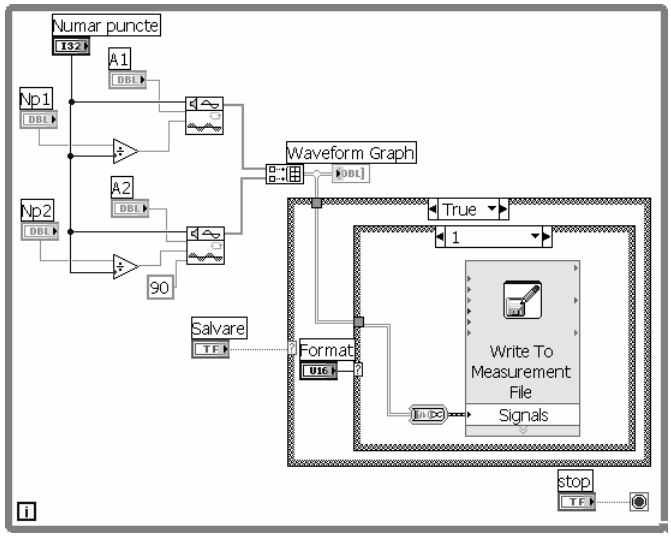


Figura 10.1

1. Deschideți un nou IV.
2. Plasați pe PF controalele şi indicatoarele necesare. Pentru selectorul de format puteți utiliza un control *Ring* sau *Enum*. Pentru butonul SALVARE utilizați un *OK Button*.
3. Generați cei doi vectori după modelul de la exemplul 9.1. Plasați toată diagrama într-o buclă WHILE.



a)



b)

Figura 10.2

4. Operaţiunea de salvare se include în cadrul *true* al unei structuri CASE, selectabilă prin butonul SALVARE. Selectarea celor două formate se face printr-o altă structură CASE, ca în figura 10.2.
5. Plasaţi în primul cadrul al structurii CASE aferente formatului, funcţia *Write To Spreadsheet File*.
6. Studiaţi helpul funcţiei şi determinaţi parametrii de intrare.
7. Realizaţi un dublu click pe funcţie şi studiaţi-i diagrama. Realizaţi iarăşi dublu click pe funcţia *Write Spreadsheet String.vi* şi observaţi lanţul *deschidere – scriere – închidere fişier* din componenţa acestei funcţii.
8. Stabiliţi formatul şi parametrii de intrare astfel încât să îndeplinească cerinţele problemei. Observaţi că este necesară o transpunere a matricii de intrare deoarece funcţia salvează fiecare vector pe linii.
9. Plasaţi pe cel de-al doilea cadru al structurii CASE aferente formatului, funcţia *Write To Measurement File*.
10. La depunerea funcţiei pe diagramă se deschide fereastra interactivă (funcţia este de tip „expres").
11. Studiaţi parametrii ferestrei prin apelarea helpului funcţiei.
12. Stabiliţi parametrii astfel încât să fie îndeplinite cerinţele problemei (figura 10.3). In secţiunea *File Name* scrieţi calea către directorul de lucru şi numele fişierului *Ex 10.1.lvm*. Dacă această secţiune rămâne necompletată, LabVIEW salvează automat în directorul special creat la instalare: *…\My Documents\LabVIEW Data*.

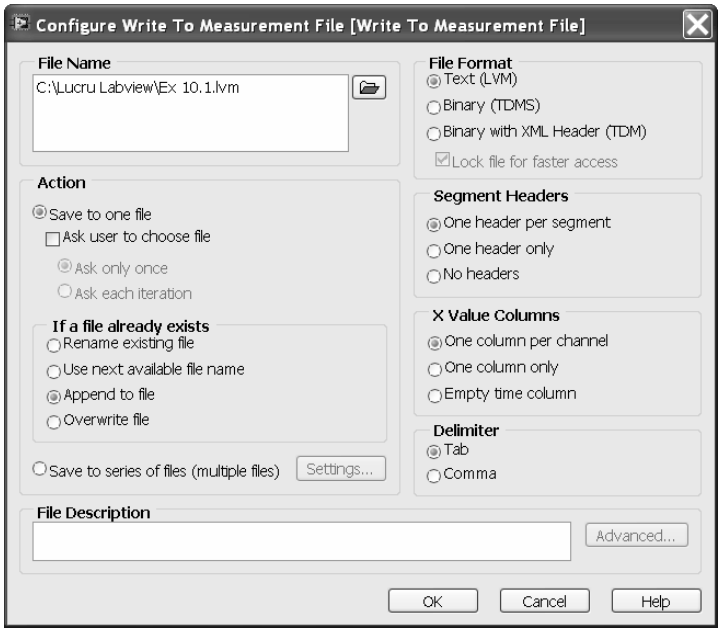


Figura 10.3

Funcția expres *Write To Measurement File* nu are posibilitatea de a stabili formatul conversiei număr – string. Formatul este fix, cu 6 cifre zecimale.

13. Legați semnalul la intrările funcțiilor.
14. Dați valori controalelor de intrare şi rulați instrumentul. Salvați de câte 3 ori consecutiv pentru fiecare format. Pentru formatul *spreadsheet*, scrieți în fereastra interactivă care se deschide numele fişierului *Ex 10.1.txt* pe calea directorului de lucru.
15. Deschideți cele două fişiere cu utilitarul *Wordpad* şi observați modul în care au fost salvate datele. Studiați headerul fişierului LVM. Faceți o comparație între ele. Observați că în cazul formatului *spreadsheet* nu există o delimitare între şirurile de date adăugate, ca la fişierul LVM.
16. Salvați IV-ul sub numele *Ex 10.1.vi*.

Exerciții propuse

EP 10.1.

a) Construiți un IV care să citească fişierele salvate la exercițiul 10.1, utilizând funcțiile de nivel înalt *Read From Spreadsheet File* şi *Read From Measurement File*.
b) Afişați vectorii citiți pe PF sub formă grafică şi într-un tabel, pe coloane.

EP 10.2.

Construiți un IV care să salveze cei doi vectori obținuți din funcțiile sinus şi cosinus în format *spreadsheet*; fişierul să conțină în capul fiecărei coloane numele funcției, iar la sfârşit să aibă o legendă cu detalii despre fişier conținând data şi ora la care acesta a fost creat.

```
SINUS    COSINUS
0.00     3.00
0.13     2.98
0.25     2.91
0.37     2.79
0.50     2.63
0.62     2.43
0.74     2.19
0.85     1.91
0.96     1.61
Detalii despre fisier:
Data: 11/12/2003
Ora: 12:23:35 PM
```

☞ Pentru realizarea acestui IV, utilizați pentru scriere funcția *Write To Text File* din subpaleta *File I/O*. Va trebui aşadar să convertiți întâi matricea de numere în format *spreadsheet* utilizând funcții din subpaleta *String*. Construiți apoi textul care va fi salvat cu ajutorul funcției *Concatenate Strings*.

EP 10.3.
Construiți un IV care să citească fişierul de la exercițiul EP 10.2.

Funcții de nivel scăzut

Funcțiile de nivel scăzut realizează câte una din sarcinile din lanțul de scriere/citire separat, adică există o funcție pentru deschidere/creare fişier, câte o funcție pentru scriere sau citire din fişier şi o funcție de închidere a fişierului. Aceste funcții se utilizează când este necesară salvarea repetată a datelor sau când este necesară adăugarea de noi atribute fişierului, care nu pot fi adăugate cu funcțiile de nivel înalt.

Funcțiile principale de nivel scăzut sunt: *Open/Create/Replace File, Close File* şi una din funcțiile de scriere/citire: *Write to Text File, Read from Text File, Write to Binary File, Read from Binary File, Format into File, Scqn from File.*

Open/Create/Replace File este prima funcție din lanțul de salvare, care creează, deschide sau înlocuieşte un fişier. Funcția returnează un număr de referință *Refnum*, care conține informații despre fişier şi care este trecut tuturor celorlalte funcții ulterioare. In acest fel, dacă se doreşte scrierea repetitivă în acelaşi fişier în cadrul unei bucle FOR sau WHILE, nu mai este necesară deschiderea şi închiderea de fiecare dată a fişierului, operații care solicită sistemul de operare şi îngreunează funcționarea calculatorului, ci se va introduce în buclă doar funcția aferentă scrierii. In figura 10.4 este dat un exemplu de adăugare a unor numere aleatoare, punct cu punct, la un vector.

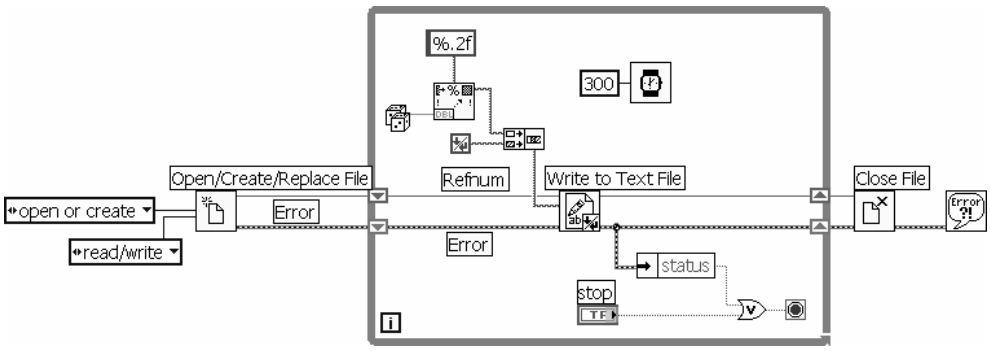


Figura 10.4

In exemplul de mai sus, funcția de scriere *Write to Text File* este plasată într-o buclă WHILE, primind printr-un registru de deplasare valoarea numărului de referință furnizat de funcția *Open/Create/Replace File*. Textul scris în fişier este un număr aleator cuprins între 0 şi 1 furnizat de generatorul *Random Number (0-1)*, apoi convertit în şir de caractere ca număr cu 2 zecimale urmat de un caracter special *End of Line,* care face ca scrierea vectorului să se facă pe coloană.

Un cluster de eroare creat de prima funcție este trecut celorlalte funcții, după care este aplicat unui indicator de tip *Simple Error Handler* din subpaleta *Dialog & User Interface*. Această legătură are rolul de a semnala pe parcursul rulării instrumentului eventualele erori care apar. Legarea în cascadă a lanțului de semnalizare a erorilor permite stabilirea cu precizie a locului şi a cauzei de apariție a erorii.

Bucla WHILE se opreşte la apăsarea unui buton de stop sau la apariția unei erori, caz în care statusul se schimbă din *false* în *true*.

Exerciții propuse

EP 10.4.
Utilizând exemplul din figura 10.4, să se realizeze un instrument virtual care să genereze şi să salveze, punct cu punct, eşantioanele a doi vectori construiți prin funcțiile sinus şi cosinus, în următoarele condiții:

- generarea eşantioanelor şi salvarea se face la intervale de timp stabilite pe panoul frontal.
- salvarea vectorilor în fişier se face pe coloane.
- pe PF se afişează cei doi vectori, punct cu punct pe un Waveform Chart.
- oprirea instrumentului se face în unul din următoarele cazuri: la apăsarea unui buton de STOP, în cazul apariției unei erori sau la atingerea unui număr de eşantioane prestabilit.

De pe PF se fixează:

- amplitudinile celor două funcții
- numărul de eşantioane după care se opreşte instrumentul
- intervalul de timp dintre două eşantioane
- calea fişierului de salvare

EP 10.5.
Aceeaşi problemă ca mai sus, dar la care se adaugă următoarele condiții:

- fiecare eşantion se salvează împreună cu ora la care a fost generat şi unghiul corespunzător
- la atingerea numărului de eşantioane prestabilit sau a unei mărimi a fişierului fixate de pe PF în octeți (byte), instrumentul nu se opreşte, ci continuă cu salvarea într-un nou fişier. Fişierele vor fi denumite automat, în ordine: F1, F2, F3, …
- La sfârşitul fiecărui fişier se va tipări un mesaj, după caz:

- „continua cu fişier …”
- „inregistrare oprita de la butonul STOP”
- „inregistrare oprita datorita unei erori”

Capitolul 11

PROBLEME RECAPITULATIVE

PR 1
Să se realizeze un instrument virtual care să calculeze răspunsul y(n) al filtrului numeric dat de ecuația:

$$y(n) = 0{,}25y(n-1) + 0{,}5y(n-2) + x(n) + 0{,}1x(n-1)$$

la o secvență de intrare x(n) formată din 100 numere aleatoare cuprinse între 50 şi 70. Filtrul este cauzal, adică toate valorile lui x(n) şi y(n) pentru n < 0 sunt nule.

☞ Se folosesc doi regiştri de deplasare : unul pentru construirea lui x(n), iar celălalt pentru y(n). Regiştrii se inițializează cu 0, deoarece filtrul este cauzal.

PR 2
Să se afle răspunsul filtrului de mai sus la o secvență de intrare x(n) formată din 10 perioade ale unui semnal sinusoidal cu amplitudinea 2, având 40 eşantioane pe perioadă.
Să se reprezinte ambele semnale pe acelaşi indicator grafic.

PR 3
- Să se realizeze un instrument virtual care să calculeze formula:

$$y = \lg x^2 + e^x + \sin \pi x + tg\, \pi x$$

  pentru 100 numere egal distanțate cuprinse între 3 şi 9.
- Să se reprezinte graficul lui y.
- Să se afişeze într-un tabel, pe o coloană, valorile lui x, iar pe altă coloană valorile lui y.

PR 4
- Să se reprezinte, pe acelaşi grafic, funcțiile :

$$y1(x) = x^3 - 5x + 3$$
$$y2(x) = 2\lg x + 1$$

pentru 200 de valori egal distanţate ale lui x cuprinse între 1 şi 3.

- Să se găsească, pe cale grafică, soluţia aproximativă a ecuaţiei :

$$y1(x) = y2(x)$$

☞ Se reprezintă pe acelaşi XY Graph funcţiile y1(x) şi y2(x) (calculate eventual cu *Formula Node*), după care se determină pe grafic, utilizând cursorul, punctele de intersecţie ale celor două curbe, ale căror abscise reprezeintă valoarile lui x pentru care y1(x) = y2(x).

PR 5

- Să se deseneze pe acelaşi XG Graph :
  - un romb căruia i se specifică coordonatele unuia din vârfuri, latura şi un unghi.
  - un segment de dreaptă orizontal, căruia i se specifică lungimea, iar unul din capete este pe romb.
- Să se deplaseze segmentul cu unul din capete pe rombul desenat, plan-paralel, cu o viteză fixată de pe panoul frontal. (*Plan-paralel* înseamnă că, în timpul deplasării, segmentul rămâne întotdeauna paralel cu o direcţie dată, orizontala de exemplu)

PR 6

- Să se construiască un IV care să afişeze, pe un Waveform Chart, la intervale de timp egale fixate de pe panoul frontal, punctele corespunzătoare funcţiei:

$$y(n) = \lg(n) + x^2 - 6$$

calculate pentru n = 1,2,3,….

- Când y(n) atinge sau depăşeşte o valoare maximă, *Ymax*, fixată de asemenea de pe panoul frontal, se afişează într-un indicator de tip string mesajul „S-a depasit limita maxima”, iar instrumentul se opreşte.
- Dacă instrumentul este oprit de către operator de la butonul STOP, se afişează mesajul: „Oprit de la butonul STOP”.

PR 7

- Să se construiască un şir y(n) format din 20 numere aleatoare cuprinse între două numere date, N_{min} şi N_{max}.
- Să se împartă şirul de mai sus în alte două subşiruri:
  - y1(n) care să conţină elementele lui y(n) cuprinse între două valori date, N_{1min} şi N_{1max}

- y2(n) care să conţină celelalte elemente ale lui y(n).

- Să se afişeze pe un indicator de tip string, pe panoul frontal, cele două şiruri, în următorul format:

```
Sirul y1(n) este:
   2.13 4.12 6.24 5.33……..
Sirul y2(n) este:
   9.44 1.16…….
Numarul de elemente al lui y1 este….
Numarul de elemente al lui y2 este……
```

- Să se salveze informaţia afişată mai sus într-un fişier ASCII.
- Să se construiască un IV care să citească fişierul de mai sus.

PR 8

- Să se construiască două progresii: una aritmetică (p.a.) şi una geometrică (p.g.) având câte 20 de termeni fiecare, cu raţia 3 şi primul termen 4.
- Să se construiască un şir format din elementele celor două progresii, în alternanţă, astfel: primul element din p.a., al doilea din p.g., al treilea din p.a, al patrulea din p.g., etc.
- Să se salveze cele două progresii într-un fişier ASCII, pe două coloane, în dreptul fiecărei coloane scriindu-se numele progresiei.
- Să se calculeze pentru şirul construit, următoarele valori: valoarea maximă, valoare minimă, valoarea medie, valoarea pătratică medie, deviaţia standard şi varianţa. Să se afişeze aceste valori într-un tabel, în următoarea formă:
-

Valoarea maxima	Valoarea minima	Valoarea medie	RMS	Dev. standard	Varianta
566	2	349	235.4	2.45	4.55

☞ Funcţiile pentru calculul valorilor cerute se găsesc în subpaleta: *Mathematics – Probability and Statistics*. Valoarea pătratică medie se numeşte în limba engleză *Root Mean Square* (RMS).

PR 9

- Să se construiască o matrice cu M linii şi N coloane, ale cărei elemente sunt numere aleatoare cuprinse între –10 şi 10 (M şi N se fixează de pe panoul frontal).
- Să se determine valoarea maximă şi valoarea minimă a matricii.
- Să se salveze matricea şi cele două valori într-un fişier ASCII cu următorul format:

```
Acest fişier a fost creat la data de ……., ora …….
```

Matrice M x N

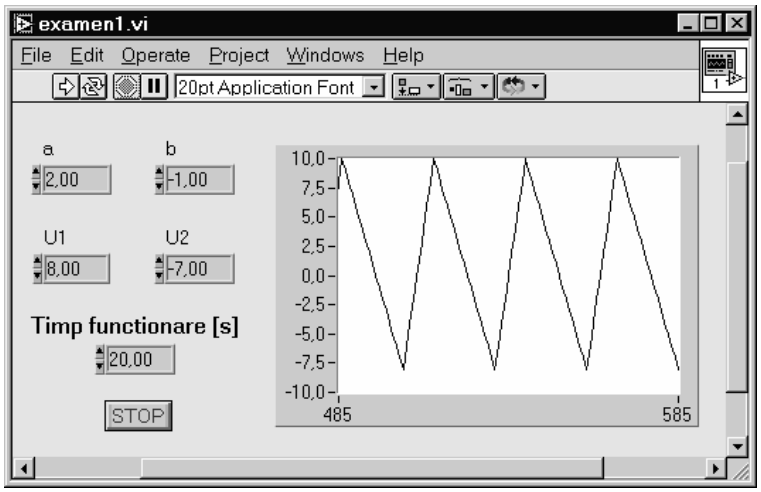
```
Valoarea maxima este ……
Valoarea minima este …….
```

PR 10

- Să se realizeze un instrument virtual care să genereze, la intervale de timp stabilite de pe panoul frontal, numere aleatoare cuprinse între 0 şi 10.
- Dacă numărul generat depăşeşte o limită superioară, se aprinde un LED verde şi se afişează mesajul "S-a depăşit limita superioară".
- Dacă numărul generat se află sub o limită inferioară, se aprinde un LED roşu şi se afişează mesajul "Numărul se află sub limita inferioară".
- Dacă numărul se află între cele două limite ambele LEDuri sunt stinse şi nu se afişează nici un mesaj.

PR 11

Să se realizeze un instrument virtual cu următorul panou frontal:



Instrumentul îndeplineşte următoarele cerinţe:

- Generează, cu indicare pe un indicator de tipul Waveform Chart, o undă sub forma dinților de fierăstrău, la care pantele să poată fi fixate de pe panoul frontal (controalele *a* şi *b*). De pe panoul frontal se fixează tensiunile maximă şi minimă, U1 şi U2, între care este încadrată forma de undă.
- Instrumentul funcționează continuu o perioadă de timp fixată de pe PF, după care face salvarea automată a datelor într-un fişier de tip "spreadsheet".
- Oprirea se face de la un buton de STOP.

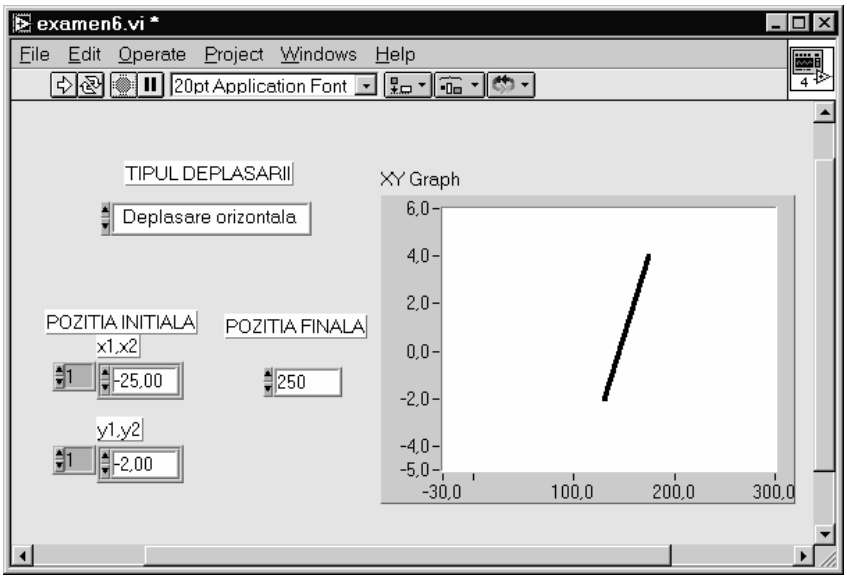
PR 12

Să se realizeze un instrument virtual care să deseneze pe un indicator de forma XY Graph un sfert de cerc cu centrul în origine şi de rază fixată de pe panoul frontal.
De pe un indicator de pe panoul frontal se selectează cadranul în care se face desenarea sfertului de cerc.
De pe panoul frontal se mai fixează, de asemenea, numărul de puncte din care este realizată curba.

PR 13

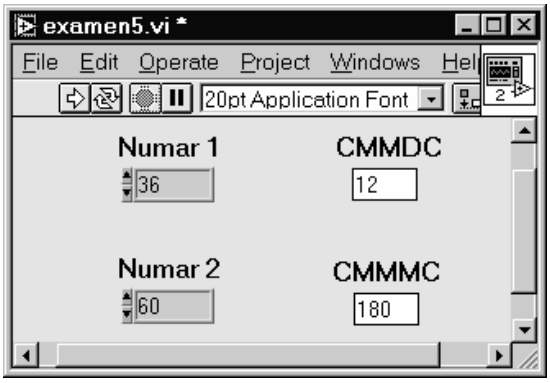
Să se realizeze un instrument virtual care să îndeplinească următoarele funcții:

- să deseneze pe un indicator XY Graph un segment de dreaptă ale cărui capete se dau prin coordonate carteziene de pe panoul frontal;
- să deplaseze acest segment plan-paralel pe orizontală sau pe verticală, la alegere dintr-un control de pe panoul frontal. Deplasarea se va face alternativ, între două puncte prestabilite.



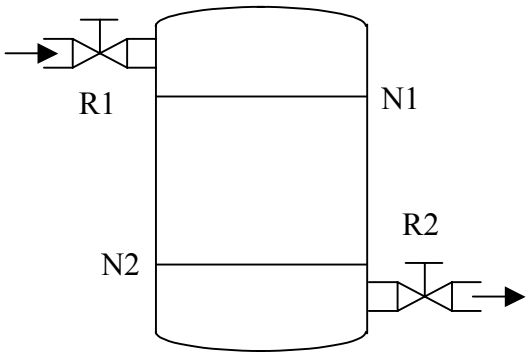
PR 14

Să se realizeze un instrument virtual care să calculeze cel mai mare divizor comun (cmmdc) şi cel mai mic multiplu comun (cmmmc) a două numere date.



PR 15

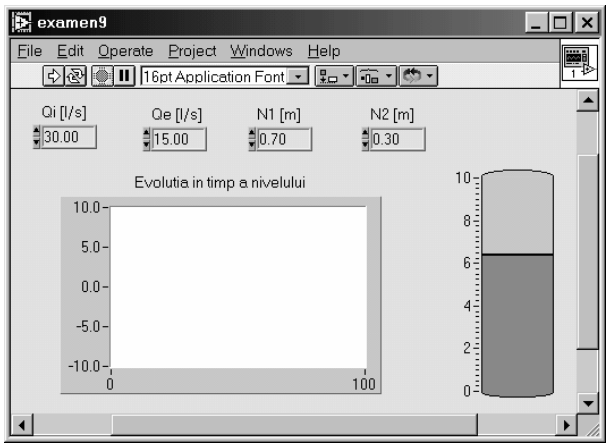
Să se simuleze în LabVIEW o instalație automată care să mențină nivelul de lichid dintr-un rezervor între două limite date, ştiind următoarele:



- Debitul consumului de lichid din rezervor este constant şi se fixează de pe panoul frontal, în l/s.
- Debitul de alimentare cu lichid a rezervorului este constant şi se fixează de pe panoul frontal, în l/s.
- Robinetul R2 este deschis permanent (consumul se face continuu).
- Robinetul de alimentare (R1) se deschide când nivelul ajunge la nivelul minim N2 şi se închide la nivelul maxim N1.
- Se va monitoriza, pe un Wave Chart variația în timp a nivelului.
- Se va indica pe un indicator de tip *tank* nivelul lichidului din rezervor în fiecare moment.
- Se va afişa pe un indicator de tip *string* starea la un moment dat a robinetului R1 (robinet deschis/robinet închis).

Indicații suplimentare:

1) La pornirea instrumentului, rezervorul este plin.
2) Se ştie că rezervorul este de formă cilindrică, cu aria bazei de 1 m^2 şi înălțimea de 1 m.



PR 16

- Să se construiască, fără a utiliza generatorul de funcții din biblioteca LabVIEW, un semnal triunghiular căruia i se specifică: nr. de puncte/perioadă, amplitudinea şi numărul de perioade generate.
- Să se calculeze valoarea medie şi cea efectivă a semnalului utilizând funcțiile Mean.vi şi RMS.vi din biblioteca *Mathematics – Probability & statistics.*
- Să se salveze semnalul generat, la apăsarea unui buton SALVARE, într-un fişier cu următorul format:

```
Nr. puncte/perioada:……..
Amplitudinea:……..
Nr. perioade:……
Media:……..
Valoare efectiva:……
Semnalul este:
…….(semnalul în format spreadsheet)……..
```

PR 17

Să se construiască un instrument virtual care să realizeze introducerea interactivă a datelor într-o bază de date având următoarele câmpuri:

- Numele
- Prenumele
- Grupa

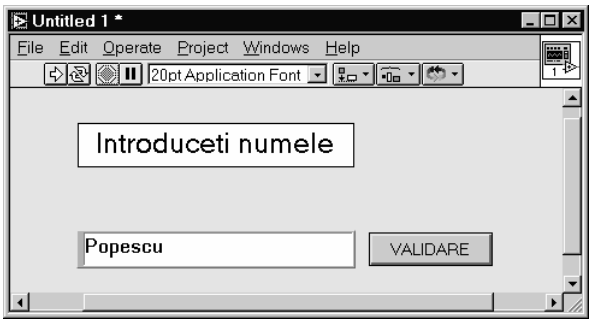
- Nota la examen

Introducerea datelor se va face interactiv, în modul următor:

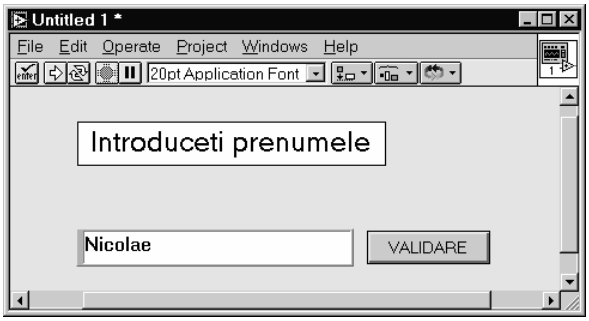
a) La pornirea instrumentului se deschide fereastra a), unde se introduce numele
b) La apăsarea tastei VALIDARE se deschide fereastra b), unde se introduce prenumele
c) La apăsarea tastei VALIDARE se deschide fereastra c), unde se introduce grupa
d) La apăsarea tastei VALIDARE se deschide fereastra d), unde se introduce nota
e) La apăsarea tastei VALIDARE se deschide fereastra e), ce conține un tabel cu cele 4 câmpuri.

Dacă se dorește o nouă înregistrare, se apasă tasta VALIDARE şi ciclul se repetă. Noile date se adaugă celor anterioare.

La oprirea instrumentului de la butonul STOP se realizează salvarea tabelului într-un fişier şi se opreşte funcționarea.



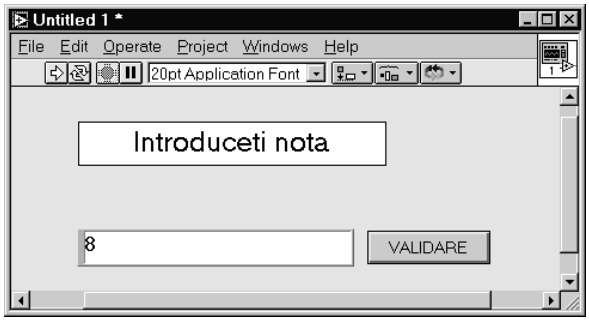
a)



b)


Untitled 1 *
File Edit Operate Project Windows Help
20pt Application Font
Introduceti grupa
6301
VALIDARE

c)


Untitled 1 *
File Edit Operate Project Windows Help
20pt Application Font
Introduceti nota
8
VALIDARE

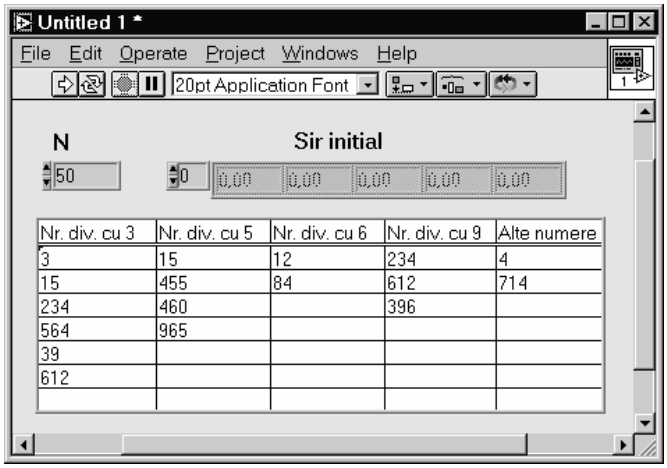
d)


Untitled 1 *
File Edit Operate Project Windows Help
20pt Application Font
Numele
Prenumele
Grupa
Nota
Popescu
Nicolae
6301
8
Doriti o noua inregistrare?
VALIDARE

e)

PR 18

- Să se genereze un şir format din N numere întregi aleatoare cuprinse între 1 şi 1000.
- Să se separe apoi şirul în alte 5 subşiruri, formate din:
  Subşir 1 – toate numerele din şir divizibile cu 3
  Subşir 2 – toate numerele din şir divizibile cu 5
  Subşir 3 – toate numerele din şir divizibile cu 6
  Subşir 4 – toate numerele din şir divizibile cu 9
  Subşir 5 – celelalte numere
- Să se afişeze aceste numere într-un tabel ca cel de pe panoul frontal de mai jos.



Nr. div. cu 3	Nr. div. cu 5	Nr. div. cu 6	Nr. div. cu 9	Alte numere
3	15	12	234	4
15	455	84	612	714
234	460		396	
564	965			
39				
612				

PR 19

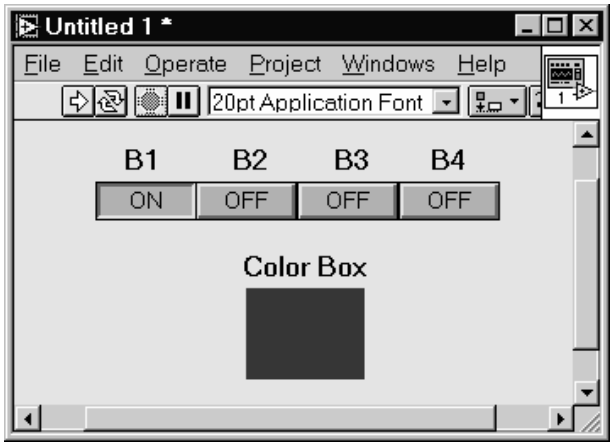
Să se construiască un instrument virtual care să găsească şi să afişeze primele N numere prime din şirul numerelor naturale, N fiind dat pe PF.

PR 20

Să se realizeze un instrument virtual care să determine şi să afişeze în ce zi a săptămânii a căzut o anumită dată din anul 2010, ştiind că 1 ianuarie 2010 a fost vineri.

Să se generalizeze problema pentru aflarea oricărei zile a secolului 21.

PR 21
Pe panoul frontal al unui instrument virtual se găsesc 4 butoane de tip boolean şi un *Framed Color Box*, ca în figura de mai jos:



Să se realizeze acest IV încât să funcţioneze în modul următor:
- la apăsarea butonului B1, Color Box se colorează roşu şi pâlpâie cu perioada de 200 ms.
- la apăsarea butonului B2, Color Box se colorează galben şi pâlpâie cu perioada de 400 ms.
- la apăsarea butonului B3, Color Box se colorează verde şi pâlpâie cu perioada de 600 ms.
- la apăsarea butonului B4, Color Box se colorează alternativ roşu-galben-verde cu perioada de 600 ms.
- la apăsarea unuia din butoane, toate celelalte revin în starea neapăsat.

PR 22
Să se construiască un instrument virtual care să găsească şi să afişeze toţi divizorii unui număr dat. Afişarea divizorilor se va face ca elemente ale unui vector şi într-un tabel.

Bibliografie

National Instruments, *LabVIEW Fundamentals*, August 2005.

National Instruments, *Getting Started with LabVIEW*, August 2006.

National Instruments, *LabVIEW Basics I Introduction Course Manual*, May 2006

National Instruments, *LabVIEW Basics II Development Course Manual*, September 2007 Edition

National Instruments, *LabVIEW Graphical Programming Course*, 2007, disponibil on-line la http://cnx.org/content/col10241/1.4/

J. Travis, J. Kring, *LabVIEW for Everyone: Graphical Programming Made Easy and Fun*, Third Edition, 2007