Programmera
i
LabVIEW

Av
Lars Bengtsson (1994)

Under bearbetning av
Lars Hellberg (januari 2007)

Version 1.5
januari 2007

Förord.

Detta är ett kompendium som skrevs av Lars Bengtsson för bl.a. GU kursen FY0270 under tidigt 1990 tal. Den visade sig också passa bra när den projektbaserade kursen Elektrisk Mätteknik B startade 1994 och det har tjänat oss väl under dessa drygt 10 år. Jag har emellertid efterhand funnit att vissa delar har föråldrats mer än acceptabelt. I synnerhet har den del som behandlar instrument-kommunikation och GPIB skapat mer problem än nytta under de sista åren. Jag har därför äntligen kommit till skott och påbörjat moderniseringen genom att skriva om Del 3 GPIB m.m.

De övriga delarna fungerar bra men kommer också att få en översyn snarast.

Göteborg 24/1 - 2007

Lars Hellberg

Förord.

Detta är ett kompendium som skrevs av Lars Bengtsson för bl.a. GU kursen FY0270 under tidigt 1990 tal. Den visade sig också passa bra när den projektbaserade kursen Elektrisk Mätteknik B startade 1994 och det har tjänat oss väl under dessa drygt 10 år. Jag har emellertid efterhand funnit att vissa delar har föråldrats mer än acceptabelt. I synnerhet har den del som behandlar instrument-kommunikation och GPIB skapat mer problem än nytta under de sista åren. Jag har därför äntligen kommit till skott och påbörjat moderniseringen genom att skriva om Del 3 GPIB m.m.

De övriga delarna fungerar bra men kommer också att få en översyn snarast.

Göteborg 24/1 - 2007

Lars Hellberg

Lars Bengtsson
Fysiska Institutionen
Chalmers Tekniska Högskola/Göteborgs Universitet

1. Starta LabView.

På DD-Linux, gå till applikationsfönstret och skriv:

labview

2. LabViews olika fönster.

När LabView startat öppnar sig två nya fönster på skärmen. Det fönster som ligger "överst" är det aktiva fönstret. Du aktiverar det undre fönstret genom att "klicka" på det.

Det fönster som låg överst från början kallas **Panelfönstret** och det undre kallas för **Diagramfönstret**.

I Panelfönstret lägger du själv in de switchar, tryckknappar displayer, indikatorer etc., som du vill ha. För varje komponent som du lägger in i Panelfönstret kommer LabView själv att placera en symbol i Diagramfönstret som representerar komponenten.

Principen är följande: Först lägger man in de komponenter man behöver i Panelfönstret. Därefter "byter" man till Diagramfönstret. I Diagramfönstret drar man de ledningar som behövs för att samman koppla de olika komponenterna. I Panelfönstret kan man också göra både enkla och mycket avancerad databehandling på de olika signalerna.

Lars Bengtsson
Fysiska Institutionen
Chalmers Tekniska Högskola/Göteborgs Universitet

3. Första exemplet.

För att klargöra begreppen för dig ska vi göra ett enkelt exempel. Vår uppgift är föjande:

På panelen ska finnas två rattar och en analogvisare. ratten är graderad från 0 till 10. När du "vrider" på en av rattarna ska den analoga visaren visa summan av vad de båda rattarna visar.

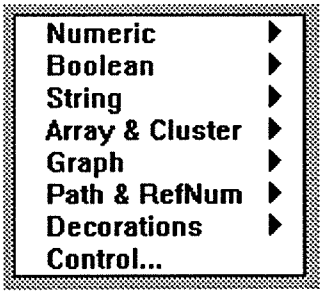
Uppgiften utförs på följande sätt:

1) Se till så att Panelfönstret är aktivt. Om du inte vet vilket fönster som är aktivt så kan du "dra ner" **Windows** menyn. Överst i Windows menyn står "Show Diagram" om Panelfönstret är aktivt och "Show Panel" om Diagramfönstret är aktivt. Du kan alltså byta fönster endera genom att peka på översta raden i Windows menyn eller genom att bara placera musen i det fönster du vill aktivera och sedan klicka där.

2) Längst upp i Panelfönstret ser du ett antal symboler. Den första symbolen är en pil:

och den andra symbolen kan vara endera en pil eller en penna. Om den är en pil, betyder det att du är i "kör-läge" (execute mode) och om det är en penna betyder det att du är i "editerings-läge" (edit mode). Det är bara i editerings-läget som du kan ändra och lägga in komponenter i panelfönstret. Se därför till att du är i editeringsläget genom att klicka på denna symbol tills pennan ligger överst. (Klicka sedan också eventuellt på den fjärde symbolen så att muscursorn är en pil.)

3) Placera nu musen inuti Panelfönstret och håll höger musknapp nedtryckt (OBS! **Håll** nedtryckt!) Då ser nu följande meny:



Medan du håller höger musknapp nertryckt drar du nu ner tills raden med "Numeric" lyser i inverterade färger. Håll fortfarande höger musknapp nedtryckt. Nu öppnar sig

en ny meny till höger. Flytta dig runt i den nya menyn (musknappen nedtryckt). Efterhand som du flyttar dig ser du längst ner vad varje komponent heter. Flytta dig runt i menyn tills du hittar den komponent som heter "Knob". När du hittat den släpper du upp höger musknapp. Då dyker ratten upp i Panelfönstret.

LabView erbjuder dig nu omedelbart ett tillfälle att sätta ett namn på komponenten. Namnet man sätter på komponenten kallas för komponentens **label**.

Skriv "Knob 1" och tryck på ENTER. OBS! Du ska inte trycka på den "vanliga" ENTER-tangenten utan på ENTER-tangenten som finns vid det numeriska tangentbordet.

Gör nu på samma sätt en gång till för att ta fram ytterligare en ratt och kalla den för "Knob 2".

Blir det trångt i Panelfönstret kan du förstora fönstret genom att endera "dra" i ett hörn eller klicka på upp-pilen i högra hörnet.

Nu återstår bara att ta fram den analoga visaren som ska visa summan av de båda vridrattarnas inställning.

4) Håll nere höger musknapp, gå in under "Numeric" menyn och leta upp den komponent som kallas "Meter" och släpp upp musknappen. Analogvisaren dyker nu upp i Panelfönstret. Sätt labeln "Summan" på denna komponent.

Nu har vi alla komponenter vi behöver. Genom att peka på en komponent och samtidigt hålla vänster musknapp nedtryckt kan du flytta komponenterna i fönstret. Flytta gärna komponenterna så att du tycker att de "ligger bra" i fönstret.

Nu återstår för oss att binda ihop komponenterna till en fungerande enhet. Detta gör vi i Diagramfönstret.

5) Aktivera Diagramfönstret genom att göra "Show Diagram" i **Windows**menyn.

I Diagramfönstret ser du nu de tre symboler som representerar komponenterna vi placerat i Panelfönstret. Symbolernas färg (orange) talar om för oss att de komponenter vi valt kan representera flyttal (dvs decimaltal). DBL anger vilket talområde som tillåts. Viktigt att observera är följande detalj:

Symbolen för "Knob 1" och Knob 2" anges med en **fet** ram, medan symbolen för "Summan" komponenten representeras av en tunn ram.

Detta beror på att "Knob 1" och "Knob 2" är "utgångar" som ger ifrån sig något (ett decimaltal mellan 0 och 10 i vårt fall). Alla komponenter som "ger ifrån sig något" är utgångar och kallas i LabView för CONTROLLERS.

Motsatsen till en CONTROLLER är en komponent som "tar emot" något och är således en ingång. En ingång kallas i LabView för en INDICATOR.

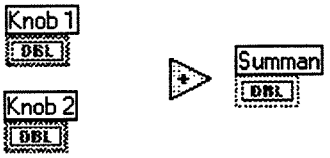
Symbolen för en CONTROLL - komponent anges med fet stil och en INDICATOR - komponent anges med tunn stil i Diagramfönstret.

Lars Bengtsson
Fysiska Institutionen
Chalmers Tekniska Högskola/Göteborgs Universitet

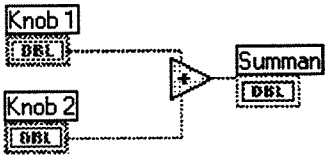
För att klara av vår uppgift behöver vi göra lite "databehandling". Vi vill ju att den analoga visaren ska visa summan av vad de båda rattarna visar. Vi behöver en additionsoperator.

6) Placera muscursorn någonstans i Diagramfönstret och håll ner höger musknapp. Välj **Arithmetics** i den meny som dyker upp. I **Arithmetics** menyn väljer du symbolen för additionsfunktionen:

Släpp upp höger musknapp. Nu hamnar additionssymbolen i Diagramfönstret. Flytta om symbolerna i Diagramfönstret så att de ligger så här:



Byt nu muscursor genom att klicka på trådrullen längts upp i Diagramfönstret. Placera spetsen på den nya muscursorn på symbolen för "Knob 1" så att denna börjar blinka. Håll ner vänster musknapp och dra sedan en tråd från "Knob 1" till **övre vänstra hörnet** på additionssymbolen. När du träffat "rätt" börjar övre vänstra hörnet att blinka på additionssymbolen. Släpp då vänster mustangent. Dra på samma sätt en tråd från "Knob 2" till **nedre vänstra hörnet** på additionssymbolen och från **spetsen** på additions symbolen till "Summan" symbolen. Slutresultatet ska se ut så här:



Nu är vi klara i Diagramfönstret. Byt nu till Panelfönstret. Nu ska vi köra programmet. Programmet exekverar du genom att först byta från editeringsläget till exekverings-läget. Detta gör du genom att klicka på pennan så att pilen istället hamnar överst:

När du nu flyttar muscursorn i Panelfönstret har cursorn förvandlats till en hand. Detta betyder att du nu kan ändra inställningen på komponenterna i Panelfönstret. Flytta

handen till den gula randen på en av vridrattarna och ändra rattens läge genom att hålla in vänster musknapp samtidigt som du "vrider" ratten. Ändra den andra ratten på samma sätt. Summan av rattarnas inställning syns ännu inte på analogvisaren. För att göra **en** programexekvering klickar du en gång på pilen (längst till vänster) i symbolmenyn längst upp i Panelfönstret. Kontrollera att analogvisaren visar summan av rattarnas inställning.

Kunde du inte köra programmet! I så fall kan det bero på att pilen i symbolmenyn är "bruten". Om den inte är hel, beror det på att man gjort något fel i ledningsdragningen i Diagramfönstret. Det absolut vanligaste felet brukar vara att man råkat dra en liten ledning någonstans som inte har någon anslutning. Det här felet är så vanligt att LabView har ett speciellt korrigeringskommando för att ta bort alla dina "dåliga" ledningar. Detta kommando heter "Remove Bad Wires" och ligger under **Edit**menyn. Kortkommandot är Ctrl+B. Lär dig detta kortkommando. Det kommer du att få stor nytta av. Alltså: Om programmet inte gick att köra, vilket indikeras med en bruten pil i symbolmenyn, tryck då först på Ctrl+B. Om pilen nu "lagades" kan du köra programmet så som anges ovan. Är pilen fortfarande bruten, be då assistenten hjälpa dig att kontrollera dina ledningsdragningar i Diagramfönstret.

Så som vi körde programmet ovan med endast en exekvering kallas "Single execute mode". Det finns också en "Continous execute mode". I denna exekveringsmod körs programmet kontinuerligt, dvs om och om igen. Du kan ändra till denna mod genom att klicka på

i symbolmenyn. Byt till "Continous execute mode" och ändra rattarnas inställning. Nu ser du att visaren uppdateras omedelbart när du ändrar rattarnas inställning.

När man gått in i den kontinuerliga exekveringsmoden får man inte glömma stänga av den också! Detta gör man genom att klicka på

4. Korrigeringar.

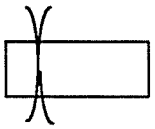
Ett problem med kopplingen ovan är att rattarna och analogvisaren båda är graderade från 0 till 10. Om summan av rattarnas läge överskrider 10 kan alltså inte visar

instrumentet visa rätt summa eftersom dess max värde är 10. Vi ska nu ändra visar instrumentets skala så att den istället går från 0 till 20.

Byt till editerings läge i Panelfönstret (dvs pennan överst). Klicka på symbolen

Lars Bengtsson
Fysiska Institutionen
Chalmers Tekniska Högskola/Göteborgs Universitet

längst upp i Panelmenyn. Muscursorn får då utseendet



Placera cursorn på siffran 10.0 i analog visaren och dubbelklicka så att siffrorna visas i vitt mot svart bakgrund. Skriv sedan "20" och tryck ENTER (OBS! ENTER vid numeriska tangentbordet). Nu ändras visar skalan automatiskt så att maxvärdet blir 20.

Byt nu till exekveringsläget (pilen överst, pennan underst) och kör programmet igen. Du behöver inte göra någon ändring i Diagramfönstret. Nu kommer analogvisaren att visa rätt värde för alla inställningar på rattarna.

5. Lite finesser.

Låt Panelfönstret vara aktivt och stoppa programkörningen om den är igång. Gå över i editeringsläge (pennan överst). Klicka på pilen (fjärde symbolen i Panel menyn) så att muscursorn blir en pil. Om du inte gjort det tidigare, så maximera nu Panelfönstret genom att klicka på "upp pilen" längst upp i Panelfönstrets högra hörn. Nu är vi klara att göra lite ändringar i vår panel så att den ser lite snyggare ut. Det första vi ska göra är att ta bort de digitala displayer som LabView automatiskt förser alla numeriska komponenter med.

Peka med musen på en av rattarna. Håll ner höger musknapp. Då öppnar sig en ny meny. Dra ner tills **Show** "tänds". I den nya meny som dyker upp är raden "Digital Display" förbockad. Genom att välja denna rad försvinner bocktecknet och därmaed också den digitala displayen. Släpp höger musknapp. Som du ser försvinner nu den digitala displayen till ratten. Gör likadant på den andra ratten och på den analoga visaren.

Om du pekar med muspilen i ena hörnet på analog visaren ändrar muscursorn utseende. Den ändrar sig till ett "dubbel hörn":

Det betyder att du hittat hörnet på figuren och då kan du också ändra storleken på figuren genom att samtidigt som du håller ner vänster musknapp så "drar" du i figuren. Ändra storleken på analogvisaren och rattarna genom att dra i deras hörn på det här sättet. Gör dem ganska stora och placera dem på lämpliga platser i fönstret. Utnyttja hela fönstret.

Nu kommer det roligaste, nämligen att färglägga komponenterna.

Klicka på symbolen för penseln längst upp i Panelfönstrets symbolmeny. Muscursorn byter då form och blir en pensel. Placera den nya cursorn i analogvisaren och håll ner höger musknapp. Då öppnar sig en färgpalett. När du flyttar dig runt i färgpaletten ändrar sig komponenten till den färg du pekar på i paletten. Välj en färg du gillar och släpp upp musknappen. Gör på samma sätt för att färglägga rattarna.

Observera två saker vid färgläggningen:

1) Du kan färglägga även små enskilda detaljer i en komponent. T. ex. kan du färglägga visaren i analog visaren i en egen färg genom att peka på visaren med penseln och hålla ner höger musknapp. Andra detaljer som kan färgläggas är t. ex. ramen kring komponenterna, skalan och komponenternas label.

2) Även bakgrunden kan färgläggas!

6. Hjälpfönstret.

En mycket användbar finess i LabView är ett tredje fönster som kallas **Help Window**. Vi kommer här att kalla det endera Help Window eller Hjälp fönstret.

Aktivera Diagramfönstret och låt muscursorn vara en pil.

Gå nu in under **Windows**menyn och välj "Show Help Window". Nu dyker ett tredje fönster upp som kallas **Help**. Placera muscursorn på additionssymbolen. I Helpfönstret dyker då additionssymbolen upp samt detaljerad information om dess in- och utgångar. Som du ser finns två ingångar som kallas **x** och **y** och en utgång som kallas **x+y**.

Förutom **vilka** in- och utgångar som finns får man också information om **var** in- och utgångarna är placerade, vilket är viktigt för att man ska kunna ansluta trådarna på rätt plats vid tråddragningen.

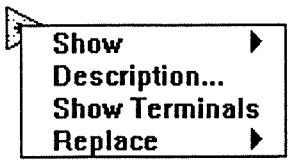
Om man har fler funktionssymboler i Diagramfönstret, väljer man vilken funktions-symbol som ska visas i Hjälp fönstret genom att bara placera muscursorn över funktionssymbolen. Den kommer då automatiskt att dyka upp i Hjäpfönstret.

7. Editering i Diagramfönstret.

Antag att vi nu istället vill att analog visaren ska visa **produkten** av rattarnas inställningar istället för summan. Då måste vi byta ut additionssymbolen i Diagramfönstret mot en multiplikationsfunktion.

Aktivera Diagramfönstret. Låt muscursorn vara en pil. Placera cursorn på additions-symbolen och håll ner höger musknapp. Då dyker följande meny upp:

Lars Bengtsson
Fysiska Institutionen
Chalmers Tekniska Högskola/Göteborgs Universitet



Dra ner till Replace. Då dyker ytterligare en meny upp. Där väljer du "Arithmetic" och får upp menyn med alla aritmetiska funktioner. Välj multiplikationsfunktionen:

och släpp upp höger musknapp. Nu har additionssymbolen i Diagrammet bytts ut mot en multiplikationssymbol.

Aktivera nu Panelfönstret. Innan du kör programmet ändrar du skalan på analog visaren så att den går till 100 istället för till 20.

Kör nu programmet och observera att analogvisaren visar produkten av rattarnas inställning.

Övning 1: Byt ut analogvisaren mot "Vertical Fill Slide" och kör programmet.

Övning 2: Byt ut funktionsoperatorn mot ett subtraktionstecken så att skillnaden mellan rattarnas inställning visas. Välj en lämplig skala.

De program vi har gjort i övningarna och exemplet ovan kallas för ett VI, och uttalas "viaj". Det är en förkortning för engelska **Virtual Instrument**. LabView hjälper oss alltså att skapa "virtuella" bilder av verkliga instrument. Mer om begreppet VI:s senare.

I övning 3 nedan ska du få konstruera ett eget VI från början. Först måste vi "rensa" skärmen. Gå in under **File** menyn och välj **Close**. Svara sedan **No** på frågan i fall du vill spara det VI du just har skapat. Nu försvinner Panelfönstret och Diagramfönstret och istället kommer ett mindre **LabView**fönster upp med tre alternativ. Välj **New VI**. Nu kan du göra övning 3 nedan.

Övning 3: Du ska ha följande **Controls**: 1 st. **Vertical Pointer Slide** och 1 st. **Horizontal Pointer Slide**. Du ska ha följande **Indicators**: 1 st. **Tank** och 1 st. **Gauge**. De båda Controllernas inställningar ska multipliceras. Tank-indikatorn ska visa inversen av produkten och Gauge-indikatorn ska visa roten ur produkten. Skala om

indikatorerna om det behövs. Tips: Du behöver tre operatorer i Diagramfönstret för att klara uppgiften: Multiply, Square root och Reciprocal. Färglägg gärna Panelen. Visa assistenten din lösning innan du går vidare.

8. Exempel 2: Constants.

Vi ska ta ytterligare ett enkelt exempel för att illustrera användandet av numeriska konstanter. Vi ska också se hur man kan ändra en indikator till en control och vice versa samt hur man byter talområde på valda variabler.

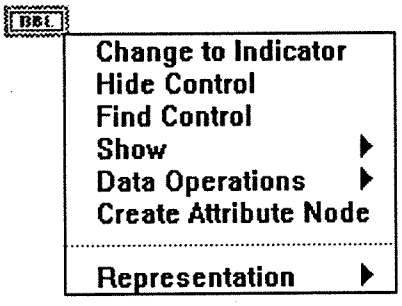
Gör **Close** i **File** menyn och välj **No** när du blir tillfrågad om du vill spara ditt program. Välj sedan **New VI** i det LabView fönster som dyker upp.

Låt Panelfönstret vara aktivt och håll ner höger musknapp. Välj en **Digital Control** och en **Digital Indicator** i **Numeric** menyn.

Vår avsikt är nu att Indikatorn ska visa "nians" multiplikationstabell. Det värde som indikatorn visar ska multipliceras med nio och visas i indikatorn.

Aktivera Diagramfönstret. Placera Controllerns symbol (den som har fetare ram) till vänster i fönstret och Indikatorns symbol till höger. Se till så att du är i editeringsläget (pennan överst) och välj pilen som muscursor. Som du ser är både Indikator- och Controller symbolen orange färgade. Det betyder att de kan anta decimalvärden. Detta behöver vi inte utan vi ska ändra detta så att de bara kan anta heltals värden.

Placera muscursorn på Controlsymbolen och håll ner höger musknapp så att du får upp följande meny:



Gå in under Representations och välj den symbol som heter Byte. Släpp upp höger musknapp. Symbolen du pekade på byter nu färg och blir blå och får utseendet

"I8" betyder Integer (=heltal) med 2 upphöjt till 8 olika värden. Det betyder att vi valt att vår indikator endast kan anta heltalsvärden från -128 till +127. Låt också Indikator-symbolen bli en heltalssymbol, men välj här istället **Word** storleken, dvs en symbol som istället för I8 visar I16.

Lars Bengtsson
Fysiska Institutionen
Chalmers Tekniska Högskola/Göteborgs Universitet

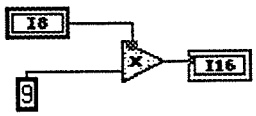
Fråga 1: Vilka heltals värden kan indikatorn nu anta? .....

Fråga 2: Varför räcker det inte med "I8" -storlek för vår indikator? .....

.....

.....

Håll ner höger musknapp och välj **Multiply**operatorn i **Arithmetic** menyn. Placera Operatorsymbolen mellan contollern och Indikatorn. Håll ner höger musknapp igen och gå nu in i menyn som heter **Structs & Constants**. Välj där **Numeric Constant** och släpp upp höger musknapp. Skriv "9" och tryck på ENTER (vid numeriska tangent bordet). Placera sedan nian lämpligt i Diagramfönstret och dra anslutningar mellan komponenter och operatorer så att det ser ut ungefär så här:



Byt till Panelfönstret och gå in i exekveringsläget (pilen överst). Kontrollera att pilen längst till vänster i symbolmenyn är "hel". Är den inte hel så gör Ctrl+B (=Remove Bad Wires). Är den fortfarande inte hel så gå tillbaks till Diagramfönstret och se om du själv kan hitta felet i din koppling. Hittar du inte felet så tillkalla assistenten.

När pilen är hel aktiverar du Continous Execute Mode genom att Klicka på

Muscursorn förvandlas som förut till ett pekfinger. Du kan nu ändra inställningen på Controllern genom att med pekfingret klicka på upp- och nerpilarna på Controllern. Kontrollera att indikatorn visar nians multiplikationstabell.

Stoppa program körningen genom att klicka på

Nu ska vi demonstrera lite finesser och kortkommandon som hjälper dig att snabba upp programskrivningen. Ha Panelfönstret aktivt, byt till editeringsläge och välj pilen som muscursor. Klicka på Indikatorsymbolen i Panelfönstret (Digitalindikatorn som du tog fram nyss). Som du ser blir det då en streckad ram runt symbolen, som talar om att symbolen är "vald". Tryck på "Delete" på tangentbordet så att den försvinner.

För att ta fram en digital indikator igen kan man naturligtvis göra som tidigare, dvs i

Numeric menyn. Här kommer en alternativ metod (som kanske inte är en bättre metod just i det här exemplet):

Klicka på den återstående komponenten, som är en Digital Control, så att en streckad ram uppstår kring komponenten. Håll ner CTRL-tangenten, placera muscursorn på komponenten, håll ner vänster musknapp och dra fram en ny komponent. Du har nu fått en kopia av den digitala controllern. Flytta den så att den ligger ungefär där Indikatorn låg tidigare.

Nu var det ju förstås inte en Digital Control vi ville ville ha utan en Digital Indicator. Enkelt! Sätt muscursorn på den nya Controllern och håll ner höger musknapp. Välj sedan **Change to Indicator** i den meny som dyker upp.

Byt till Diagramfönstret. Ta bort de gamla streckade ledningarna med CTRL+B och dra en ny ledning till den nya Indikatorn. Klart! Testkör programmet igen.

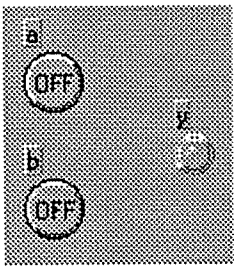
9. Exempel 3: Booleska variabler och operatorer.

Vi ska göra ett enkelt exempel på med **booleska variabler**, dvs variabler som endast kan anta två värden (sant eller falskt). Dessa variabler kallas också för **logiska variabler**. Vi ska skapa ett Panelfönster med tre komponenter: Två stycken logiska controllers och en logisk indikator. Indikatorn ska visa OR-funktionen av de båda controlleringångarna.

Aktivera Panelfönstret, gör **Close** i **File** menyn, spara inte ditt gamla VI och välj sedan **New VI**.

Håll ner höger musknapp och välj **Labelled Round Button** i **Boolean** menyn. Skriv "a" som dess Label och tryck ENTER. Hämta på samma sätt ytterligare en rund tryck-knapp och kalla denna för "b".

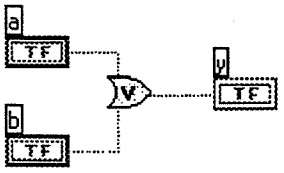
Håll ner höger musknapp igen och i **Boolean**menyn hämtar du nu LED, vilket är en digital indikator. Kalla den "y" och tryck ENTER. Din Panel ska nu se ut ungefär så här:



Alla logiska indikatorer och controller på Panelen är från början i sitt "falsk" läge. Vi vill att LEDindikatorn ska tändas röd vid "sant". För att åstadkomma detta måste vi först ändra LEDindikatorns läge till "sant" och sedan färga den röd.

Ändra muscursorn till en hand (eller ett pekfinger) och klicka en gång på LED:n. Nu visas dess "sant"-sida. Byt muscursor igen till en pensel, placera penslen på LED:n och håll ner höger musknapp. Välj en röd färg.

Byt till Diagramfönstret och byt till pilcursorn. Håll ner höger musknapp, gå in i **Arithmetic**menyn och välj OR-operatorn. Placera komponenterna lämpligt och dra ledningar så att du får ett Diagramfönster som nedan:



Byt tillbaks till Panelfönstret. Byt till kontinuerligt exekveringsläge. "Tryck" på knapparna och observera LED-lampans variationer.

Fråga 3: För vilka inställningar på Controller-knapparna "lyser" LED lampan? .....

.....

.....

Övning 4: Ändra så att LED-lampan istället visar NAND-funktionen av de båda Controlleringångarna. OBS! Glöm inte att stoppa programkörningen innan du börjar ändra i Diagramfönstret.

10. Textsträngar och strängoperationer.

En av LabView kanske allra bästa egenskaper är dess förmåga att hantera textsträngar på ett enkelt men kraftfullt sätt. Anledningen till detta är förstås att språket utvecklades av National Instrument i avsikt att förenkla kommunikationen med mätinstrument försedda med GPIB-interface, dvs sådana instrument som kan kontrolleras från PC med hjälp av kommandon skrivna i klartext.

Med ytterligare ett exempel ska vi illustrera användandet av strängar och de två vanligaste strängoperationerna, nämligen "Pick Line & Append" och "Concatenate Strings". Vi förklarar dessa i detalj nedan.

Vår uppgift är följande:
I Panelen ska finnas två stycken **Textringar**. En Textring är en Control som innehåller ett antal textrader och till varje textrad hör ett radnummer. Man väljer textrad genom att klicka på en upp- eller nerpil som finns till varje Textring. Ut från Controllern kommer en heltalssiffra, vilket helt enkelt är radnumret till den textrad man har valt i Textringen.

Den ena textringen ska ha innehållet

Volvo
Saab
Ford
Audi
Nissan
Fiat

och den andra textringen ska innehålla

bästa
sämsta
värsta
risigaste
roligaste
tråkigaste

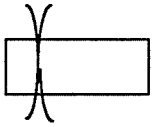
Sen ska vi ha en **String Indicator** som visar en mening som beror på vilka ord vi valt i textringarna. T. ex.

Volvo är den bästa bil jag kört.

Gör Close i Filemenyn som vanligt utan att spara ditt program. Gör sedan New VI så att du får en "fräsch" skärm att jobba med.

Låt Panelfönstret vara aktivt. Håll ner höger musknapp och välj **Textring** i **Numeric** menyn. Kalla dess Label för **bil** och tryck ENTER. Byt muscursor genom att klicka på

Då blir cursorn till en



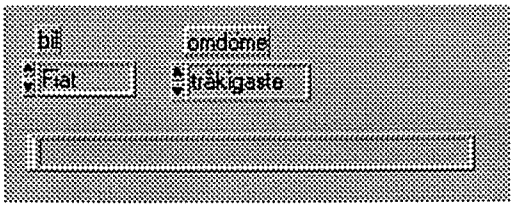
Placera denna cursor i Textringen du just tog fram och klicka en gång med vänster musknapp. Skriv **Volvo** och tryck ENTER. Placera muscursorn på textringen och håll

ner höger musknapp. Välj **Add Item After** i den meny som dyker upp. Nu kommer nästa rad i Textringen (som ännu är tom) att dyka upp i Textringens fönster. Skriv in **Saab** och tryck ENTER.

Gör på samma sätt för att skriva in **Ford**, **Audi**, **Nissan** och **Fiat**. Skapa sedan en ny Textring vars Label du kallar för **omdöme**. I denna textrings rader skriver du i tur och ordning in **bästa**, **sämsta**, **värsta**, **risigaste**, **roligaste** och **tråkigaste**. OBS. Ordet "tråkigaste" får inte helt plats i fönstret så du får "ta tag" i fönstrets ena hörn och göra fönstret lite längre.

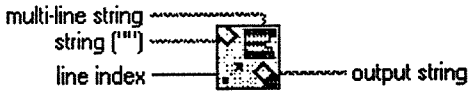
Lars Bengtsson
Fysiska Institutionen
Chalmers Tekniska Högskola/Göteborgs Universitet

Placera muscursorn i Panelfönstret igen. Håll ner höger musknapp och hämta en **String Indicator** i **String**menyn. Ta tag i ena hörnet på Indikatorfönstret och dra ut det så att du får en Panel som ser ut ungefär som nedan:



Byt nu till Diagramfönstret. Börja med att placera symbolerna i Diagramfönstret så här:

Lämna mycket "space" ovanför symbolerna. Aktivera **Help Window**. Den första strängoperator vi ska hämta heter **Pick Line & Append**. Håll ner höger musknapp och hämta denna operator i **String** menyn. I Hjälpfönstret kommer då följande symbol upp:



Pick Line & Append

Operatorn fungerar på följande sätt: "Multi-line string" är en kolumn med ord. Det översta ordet har radnummer 0, nästa har radnummer 1 osv. Värdet på ingången "line index" talar om vilken rad som ska sammanfogas **efter** den text som kommer in på "String" ingången. "String" ingången kan också lämnas öppen (= tom sträng). Det som kommer ut på utgången "output string" är alltså ingångssträngen sammanfogad med en av raderna i "multi-line" strängen. Vilken rad som sammanfogas med ingångssträngen bestäms av värdet på ingången "line index". Man ska alltså "Välja en rad och sammanfoga" vilket på engelska blir "Pick Line & Append".

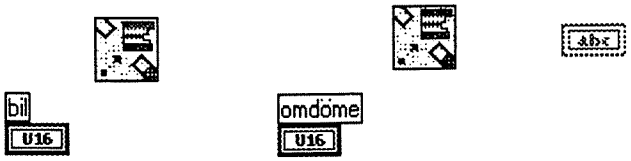
Observera att:

1) Strängvariabler har rosa färg.

2) "line index" ingången ska ha en heltalsvariabel.

Ta fram ytterligare en "Pick Line & Append"-operator och placera dem som nedan:

Lars Bengtsson
Fysiska Institutionen
Chalmers Tekniska Högskola/Göteborgs Universitet



Håll ner höger musknapp igen och ta fram en **String Constant** i **Structs & Constant**-menyn.

I det nedanstående ska du observera följande: RETURN betyder att du trycker på Enter-tangenten vid det alfabetiska tangentbordet och ENTER betyder att du trycker på Enter-tangenten vid det numeriska tangentbordet.

I String Constanten som du nyss hämtade skriver du följande:

Volvo	RETURN
Saab	RETURN
Ford	RETURN
Audi	RETURN
Nissan	RETURN
Fiat	ENTER

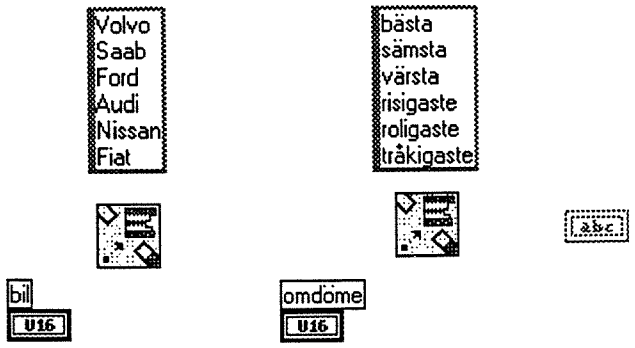
Placera den här kolumnen med ord ovanför den vänstra **Pick Line & Append**-operatorn i ditt Diagramfönster. Den här ordkolumnen blir vår "multi-line string" till den vänstra **Pick Line & Append**operatorn.

Skapa på samma sätt en ordkolumn med innehållet

bästa
sämsta
värsta
risigaste
roligaste
tråkigaste

och placera denna ovanför den högra **Pick Line & Append** operatorn. Ditt Diagramfönster ska nu se ut ungefär så här:

Lars Bengtsson
Fysiska Institutionen
Chalmers Tekniska Högskola/Göteborgs Universitet



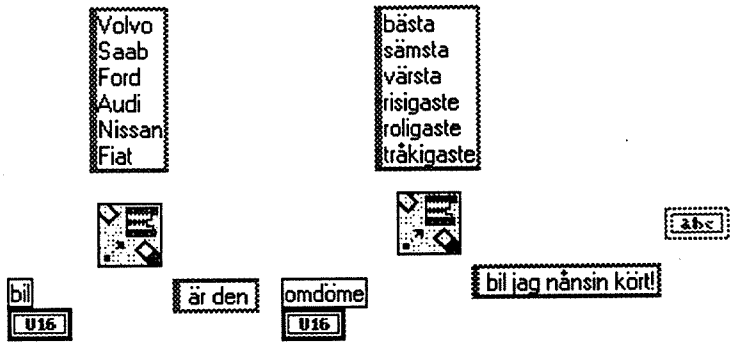
Ta fram ytterligare två String Constant. I den första skriver du bara en rad som lyder

är den

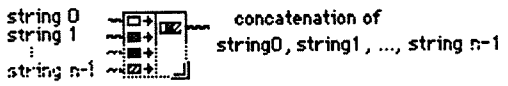
Inled och avsluta med ett mellanslag! I den andra skriver du

bil jag nånsin kört!

Inled här med ett mellanslag. Placera dessa sträng konstanter i Diagramfönstret ungefär som nedan:



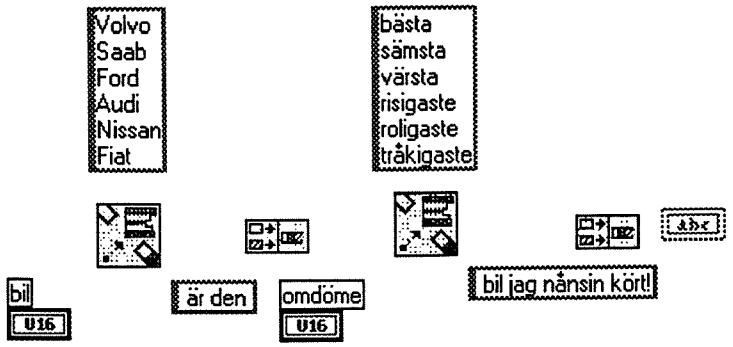
Nu behöver vi bara ett verktyg till innan vi kan börja ledningsdragningen. Vi behöver ett sätt att "knyta ihop" de strängar vi skapat. **Concatenate Strings**operatorn i **String**-menyn löser problemet åt oss. Plocka fram denna operator och placera muscursorn på symbolen så att den dyker upp i Hjälp fönstret:



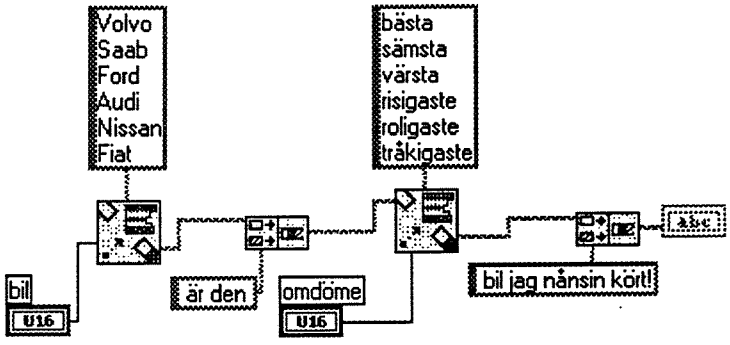
Concatenate Strings

Concatenate Stringsoperatorn binder ihop två strängar till en sträng. Den sträng som matas in överst läggs först i den hopbundna strängen. Operatorn kan även binda ihop fler än två strängar. Man kan nämligen utöka antalet ingångar på operatorn genom att "dra" i dess ena hörn!

Du behöver två Concatenate Strings operatorer till vår uppgift. Placera dem enligt figuren nedan:



Nu är det dags för tråddragningen. Nedan ser du den färdiga kopplingen.



Fundera ett ögonblick på hur kopplingen kommer att fungera och varför den fungerar. Byt sedan till Panelfönstret och kör programmet.

Gör **Close** i **File** menyn och **New VI** innan du gör övning 5 nedan.

Övning 5: I Panelfönstret ska du ha en Text ring och en String Indicator. I Text ringen ska följande ord finnas

toppen!
skitkul!
underbart!
fantastiskt!

I String Indikatorn ska man kunna se en mening som beror på vilket ord man väljer i Text ringen:

Lars Bengtsson
Fysiska Institutionen
Chalmers Tekniska Högskola/Göteborgs Universitet

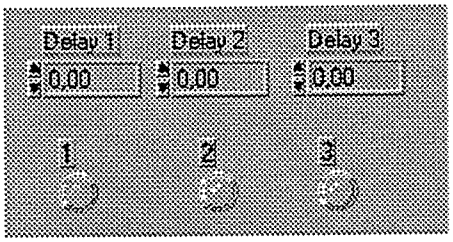
LabView programmering är toppen!

11. Sekvens rutor och tidsfunktioner.

Vi ska avsluta den här labben med ett exempel som demonstrerar användandet av **sekvensrutor** i LabView. I samband med detta ska vi presentera en tidsfunktion och användandet av en logisk konstant.

I Panelfönstret ska vi ha tre stycken **Digital Controls** och tre stycken **Round LED**. Vi ska skriva ett program som tänder LEDindikatorerna med en viss fördröjning. Fördröjningen för varje LED anges av respektive Digital Control.

Rensa skärmen så som du gjort tidigare. Börja i Panelfönstret. Ta fram tre stycken **Digital Controls** från **Numeric**menyn och tre stycken **Rund LED** ur **Boolean**menyn. Sätt lämpliga Labels på komponenterna så att du kan identifiera dem i Diagramfönstret. Placera dem i Panelfönstret ungefär som nedan.



Byt nu till Diagramfönstret. Börja med att ändra representationen på Digital Controllerssymbolerna så att dessa blir "I8" istället för "DBL". Flytta sedan undan alla symboler och lägg dem alla tills vidare i vänstra kanten av Diagramfönstret.

Håll ner höger musknapp och gå in i menyn **Structs & Constants**. Där hämtar du den symbol som heter **Sequence**. Symbolen som nu hamnar i Diagramfönstret ser ut som en filmruta (åtminstone ska man tolka den så).

Så här fungerar sekvensrutorna:

Längst upp i sekvensrutan ser du ett nummer, nummer 0 i vårt fall. Sätt muscursorn på 0:an och håll ner höger musknapp. Välj **Add Frame After** i den meny som dyker upp. Nu ändrades numret längst upp till en etta. Vad som egentligen hände var att du adderade till ytterligare en sekvensruta efter den med nummer noll. Genom att klicka på pilarna längst upp i sekvens rutan kan du få fram rutan med nummer noll. Finessen men sekvens rutor är att de kommandon och operatorer som du skriver i ruta noll kommer att utföras **först**, **sedan** kommer allt i sekvens ruta ett att utföras. Man kan naturligtvis ha fler än två sekvens rutor.

Gör **Add Frame After** fyra gånger till så att den sista rutan har nummer 5.

"Ta tag" i sekvensrutans hörn och gör den lite större, ca 10x10 cm.

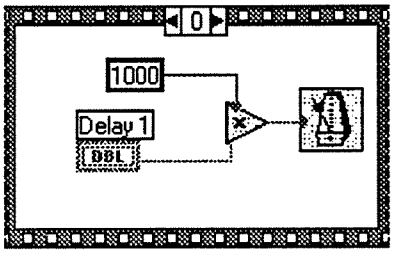
För att lösa uppgiften med att tända LED indikatorerna med olika tidsfördröjning ska vi utnyttja dessa sekvensrutor. I första rutan (ruta 0) skapar vi en fördröjning, i andra rutan tänder vi LED nummer 1, i tredje rutan har vi en ny fördröjning, i fjärde tänder vi andra LED:n osv. Längden på varje fördröjning bestäms av tillhörande **Digital Indicator**.

Låt sekvens ruta noll ligga "överst". Placera muscursorn i sekvens rutan och håll ner höger musknapp. Gå in i menyn **Dialog & Date/Time** och hämta **Wait Until Next ms Multiple**. Figuren dyker då upp i sekvensrutan och i Hjälpfönstret ser du följande figur:

Wait Until Next ms Multiple

Till vänster kommer ett heltal in. Sedan kommer programkörningen att stoppas i så många ms som talet anger. Programmet fördröjs. Talet 1000 in på vänster sidan kommer att ge en fördröjning på 1 sekund. Utgången kommer vi inte att använda i vårt program.

Eftersom det är en multipel av millisekunder som är inparameter till tidsfördröjningen, betyder det att vi måste slå in ett ganska stort tal för att få en märkbar fördröjning. För att slippa slå in ett stort tal kan vi istället mata in ett litet tal och sedan i **Diagramfönstret** multiplicera talet med 1000. I Sekvensruta nummer 0 ska du då ha fyra saker: Symbolen för den **Digitala Control** som ska bestämma fördröjningen till första LED:n, en konstant som är lika med 1000, en multiplikationsoperator samt tidsfunktionen ovan. Försök att hitta alla dessa saker själv och placera dem i ruta noll. Ledningsdragningen listar du väl ut själv? Ungefär så här bör det se ut:

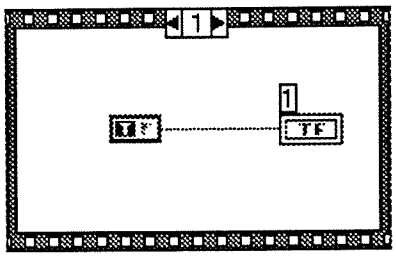


Klicka på den högra pilen så att sekvens ruta 1 kommer upp. När vi kommer till den här filmrutan har fördröjningen i ruta 0 klarats av och nu är det dags att tända första LED:n. För att "tända" en logisk indikator (vilket LED:n ju är) behövs en boolesk konstant. Placera muscursorn i ruta 1 och håll ner höger musknapp. Gå in under **Structs & Constants** och hämta **Boolean Constant** som ser ut så här:

En logisk konstant kan anta två olika värden "True" eller "False". Efter som den högra delen av symbolen, den med bokstaven "F" i, är grön, betyder det att vår konstant har värdet "False". Eftersom den här konstanten ska "tända" LED:n måste den ha värdet "True". Byt muscursor så att cursorn blir en hand. Placera sedan handen på bokstaven "T" i symbolen och klicka en gång så att bokstaven "T" istället blir grön:

Lars Bengtsson
Fysiska Institutionen
Chalmers Tekniska Högskola/Göteborgs Universitet

Placera även symbolen för LED 1 i rutan och dra en tråd mellan konstanten och LED:n:



I ruta 2 placerar du den **Digitala Control** som ska bestämma fördröjningen innan LED 2 ska tändas. I övrigt ser ruta 2 ut exakt som ruta 0. Ruta 3 ser ut exakt som ruta 1 fast här ska förstås LED 2 tändas. Vad som ska vara i rutorna 4 och 5 listar du ut själv, eller hur?

När du är klar med ledningsdragningen i **Diagramfönstret** går du tillbaks till **Panelen**. Byt till exekveringsläge men starta inte programmet än. När du är i exekveringsläget blir cursorn en hand. Klicka på pilarna som hör till varje **Digital Control** så att delay1 blir tre sekunder, delay2 blir 2 sekunder och delay3 blir en sekund.

Kör nu programmet **en** gång genom att klicka på pilen längst till vänster i panel menyn (inte **Continous execute** alltså).

Fungerar programmet så ska LED:erna tändas en i taget med allt mindre fördröjning.

Fråga 4: Efter att du kört programmet en gång så är alla dioderna tända. Att köra programmet ytterligare en gång vore alltså meningslöst eftersom de redan är tända. Anta att vi vid nästa programkörning vill få LED:erna att släckas med samma fördröjning som de tänds. Vilka ändringar måste du då göra i Diagramfönstret?

.....

.....

.....

Övning 6: Gör de nödvändiga ändringarna du föreslagit i fråga 4 och kör programmet.

12. Förslag till fler roliga övningar.

Har du tid över och vill träna mer på LabView finnas fler övningar i kompendiet

"Mäta och styra med persondatorer, övningar"

av Johan Boman, Ove Eriksson, Morgan Gustafsson och Thomas Nilsson. Följande övningar behandlar de avsnitt vi gått igenom ovan:

1.1, 1.2, 1.3, 1.8, 1.11, 3.6.

Lars Bengtsson
Fysiska Institutionen
Göteborgs Universitet/
Chalmers Tekniska Högskola

September 1994

Programmera i LabView.

2. Talomvandlingar, loopar och mer om texthantering.

Innehåll: Vi lär oss hur man omvandlar från en talrepresentation till en annan, vi tittar på for- och whileloopen, casesatsen, grafisk presentation och mer om hantering av textsträngar tillsammans med olika talrepresentationer.

Keywords: Conversions, For-loop, While-loop, Case, Text String, XY-Graph

---

Namn: .....

Handledare: .....

Laborationen utförd den .....
Datum

Laborationen godkänd den ..... av .....
Datum Handledares underskrift

1. Först lite om talkonvertering.

Ett tal kan på datorspråk representeras på en mängd olika sätt. Det kan vara ett heltal eller flyttal, det kan vara decimalt, binärt eller hexadecimalt. Det kan också behandlas som en textsträng. Den viktigaste frågan i talrepresentation är om talet är lagrat som en textsträng eller som ett "riktigt" tal. Vi tar ett exempel. Det decimala talet "7" ska lagras i datorn. Om detta lagras som ett tal kommer det någonstans i datorns minne att finnas en minnescell med det binära innehållet "0000 0111" (vi förutsätter byte-orienterat minne).

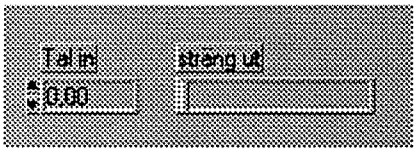
Ett annat sätt är att lagra talet som ett text tecken. I så fall lagras istället talets ASCII-kod. ASCII-koden för tecknet "7" är 37hex. I det här fallet finns det en minnescell någonstans som har innehållet "0011 0111".

Ska talet användas för matematiska operationer är det första alternativet att föredra eftersom det inte går att göra några matematiska operationer på "texttecken".

Vid hantering av mätdata i samband med GPIB-bussen kommer vi ofta att ha behov av att omvandla ett tal från textsträngar till "vanliga" tal. GPIB bussen talar ju "klarspråk", vilket bl. a. innebär att mätdata skickas i form av ASCII-kodade text-strängar. Vill man därför utföra några matematiska operationer på mätdata så måste dessa först omvandlas till "vanliga" tal.

Den första funktionen vi ska titta på heter **To Decimal**.

Starta LabView, om du inte redan har gjort det och se till att du har ett "rent" Panel-fönster att starta i. I Panelfönstret placerar du en **Digital Control** och en **String Indicator**. (Kommer du inte ihåg hur man hämtade dessa kan du titta tillbaks i första labben.) I Controllerns label skriver du "Tal in" och i Indicatorns label skriver du "Sträng ut":



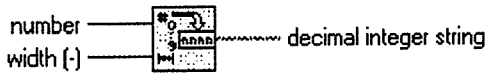
Vi vill i den här uppgiften att det tal vi skriver in på controllern ska visas i sträng-indicatorn. Normalt sett ska ju inte det här gå. Ett tal kan ju inte vara en sträng, eller hur? I LabView finns det dock funktioner som omvandlar ett tal till en sträng.

Byt till Diagramfönstret och aktivera Hjälpfönstret. Ändra först talrepresen-tationen för Controllern från "Double Precision" till "Word". (Håll ner höger musknapp på ikonen och gå in under **Representation**menyn.)

Som du ser är färgen på controllern blå och färgen på indicatorn är rosa. Dessa kan man alltså inte koppla ihop. (Pröva att dra en tråd från controllern till indicatorn, får du se.)

Placera cursorn någonstans i Diagramfönstret (inte på någon ikon) och håll ner höger musknapp. I menyn **Strings** hämtar du funktionen **To Decimal** och placerar denna

mellan talcontrollern och stringindicatorn. Symbolen för **To Decimal** dyker upp i Hjäpfönstret:



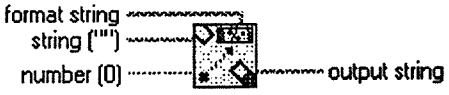
To Decimal

På "number" ingången kommer talet som ska omvandlas in. "Width" ingången kan lämnas öppen. På utgången "Decimal integer string" kommer resultatet av omvandlingen, dvs den omvandlade strängen. Observera att "number" ingången är blå och "Decimal integer string" utgången är rosa. Placera den mellan controllersymbolen och indicatorsymbolen och dra ledningar som nedan:

Kontrollera att pilen längst till vänster i menyn längst upp i fönstret är hel. (Om inte, tryck CTRL+B. Det kan vara "dåligt" dragna ledningar som bryter pilen.)

Byt till Panelfönstret när pilen är hel. Skriv in ett tal i talcontrollern, t. ex. 345, och kör programmet. 345 dyker nu upp i sträng indikatorn.

Vi ska nu passa på att presentera ytterligare en strängfunktion, nämligen **Format & Append**. Symbolen ser ut så här:



Format & Append

Med **Format & Append** kan man ta ett tal (representerat som ett tal, inte sträng) och omvandla det till en textsträng samtidigt som man lägger till lite text tillsammans med talet. Talet som ska omvandlas kommer in på "number" ingången. Man kan bestämma om talet ska presenteras som ett heltal eller flyttal, hexadecimalt eller annan bas. Detta bestäms av **formatsträngen**.

Syntaxen för formatsträngen ser ut så här (något förenklat!):

{textsträng}%Konverteringstecken{textsträng}

Textsträngarna som kan omge konverteringstecknet är valfria och kan uteslutas. Observera %-tecknet som talar om för kompilatorn var konverteringstecknet börjar. Följande konverteringstecken finns:

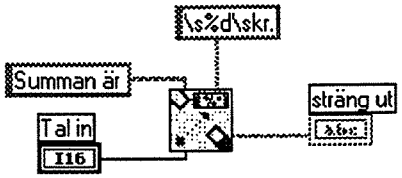
- d omvandling till decimalt heltal
- x omvandling till hexadecimalt heltal

o	omvandling till oktalt heltal
f	omvandling till flyttal
e,g	omvandling till flyttal uttryckt som en tiopotens

Dessutom kan format strängen innehålla "osynliga" tecken, t. ex. Carriage Return (r) och Space (s). Dessa tecken ska förgås av ett "backslash" - tecken i format strängen, dvs ett "\".

stringingången kan lämnas öppen eller knytas till en textsträng. Denna textsträng kommer i så fall att hamna före den fomaterade strängen.

Vi tar ett exempel. Anta att vi har en **Format & Append** funktion med följande utseende:

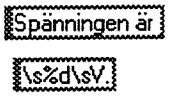


Anta att talet i "number" ingången är "17". Ovanstående kommer då att resultera i texten

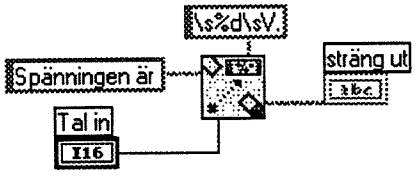
Summan är 17 kr.

Den första delen, *Summan är*, kommer från "string" ingången. Formatsträngen säger sen att nästa tecken ska vara ett mellanslag (ett "Space") eftersom den inleds med "\s". "d":et efter procenttecknet betyder att talet in på numberingången ska göras om till ett heltal (representerat som text!). Sen kommer ytterligare ett mellanslag, "\s", och sen kommer texten *kr.* .

Byt till **Diagramfönstret**. Låt cursorn vara en pil. Placera cursorn på **To Decimal** operatorn och håll ner höger musknapp. Medan du håller ner höger musknapp drar du ner till **Replace**. Då öppnar sig den vanliga **Diagramfönster** menyn. Gå in under **String**menyn och hämta **Format & Append** funktionen. Släpp upp höger musknapp. som du ser byttes den gamla operatorn ut mot den nya **Format & Append**. Tryck CTRL+B en gång för att få bort alla ledningsdragningar som eventuellt inte passar till den nya operatorn. Skapa två strängkonstanter som ser ut så här:



Dra sedan ledningarna i Diagramfönstret så här:



Om pilen i vänstra hörnet är hel kan du gå till Panelfönstret och köra programmet. Skriv in ett tal i numeric controllern och kör. I string indikatorn ska du då se t. ex.:

Spänningen är 20 V.

Övning 1: Skriv ett VI som med hjälp av en **Format & Append** funktion ger en utskrift i två rader. Du ska konstruera en formatsträng som ger utskriften

Räntan är
1,234E+1 %

Allt ska åstadkommas i formatsträngen, du ska alltså inte ha någon sträng på **string** ingången. Räntesatsen ska vara kontrollerbar via en **numeric control** i **Panelfönstret** och resultatet ska presenteras i en två raders **textindikator**. Tips: Det "osynliga" tecknet för ny rad heter \n och för att skriva ut ett procent tecken via format strängen måste detta dubbelskrivas för att kompilatorn inte ska blanda ihop detta med början på konverteringstecknet.
Observera att talet ska presenteras i tiopotensform.

Fråga 1: Vilken format sträng kom du fram till i övning 1? .....

.....

***Övning 2:** Exemplet ovan som ger utskriften *Spänningen är 20 V*, kan också lösas med hjälp av **To Decimal**, två strängkonstanter och **concatenate strings**. Hur? Visa din lösning för assistenten.

Gå in i **Diagramfönstret** igen och håll ner höger musknapp. Om du går in i **String**-menyn så ser du att det finns många fler strängoperatorer som vi ännu inte har berört. T. ex. finns motsatsen till **To Decimal**, dvs en funktion som tar ett tal representerat som en sträng och omvandlar till ett "riktigt" tal. Denna funktionen heter naturligtvis **From Decimal**. Motsvarande funktioner finns också för oktala och hexadecimal tal. Av de övriga har vi i den här labben berört **Format & Append** och i labb 1 använde vi **Pick Line & Append**. De övriga ska vi inte beröra med exempel utan dessa förklaras vid behov senare.

Vi lämnar tills vidare tal/sträng hanteringen.

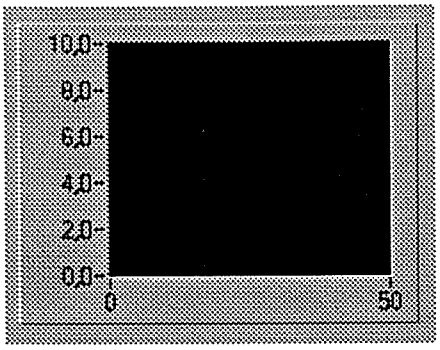
2. Grafisk presentation av mätvärden.

Ofta vill man visa inte bara det nuvarandet värdet av en funktion eller mätprocess, utan man vill också se hur värdet har sett ut tidigare och hur det varierat i tiden. För detta erbjuder LabView en funktion som heter **Waveform Chart**. Denna väljer man som en

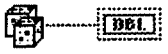
indikator i Panelfönstret under menyn **Graphs**. I **Diagramfönstret** ska den behandlas som en indikator. Vi ska här med ett enkelt exempel visa hur den fungerar. Vi ska använda den i nästa avsnitt, som handlar om loopar, där den kommer mera till sin rätt.

Gå över i Panelfönstret och rensa fönstret som vanligt utan att spara ditt gamla VI.

Håll ner höger musknapp och hämta indikatorn **Waveform Chart** i **Graphmenyn**.



Ändra Y-skalan så att den går från 0 till 1,0. Byt sen till Diagramfönstret. Placera Chartsymbolen i **Diagramfönstrets** högra halva. Håll ner höger musknapp och hämta funktionen **Random Number (0-1)** i **Arithmetic**menyn. Denna funktion genererar ett slumptal mellan 0 och 1. Knyt slumptalsgeneratorn till Chartsymbolen:



Byt till Panelfönstret och exekveringsläge. Klicka upprepade gånger på exekverings-pilen längst till vänster i menyn längst upp. För varje gång programmet exekveras presenteras värdet i grafen.

För att visa lite finesser med Waveform Graph ska du göra följande:

1) Byt till editeringsläge och låt cursorn vara en pil. Placera cursorn i graf-fönstret och håll ner höger musknapp. Gå in under menyn **Data operations** och gör **Clear Chart** så att grafen "töms".

2) Placera cursorn i grafen igen och håll ner höger musknapp. Den här gången går du in under menyn **Show** och väljer **Legend**. En figur dyker då upp till höger om grafen som representerar graflinjen. Nu kan du "editera" linjen.

3) Placera cursorn i legendfönstret och välj en röd färg i **Color**menyn.

4) Placera cursorn i legendfönstret igen. Håll ner höger musknapp och gå in i menyn **Point Style** och välj en punktform.

5) Gå in i legendmenyn igen och gå in under **Interpolation** och välj **None**.

Byt till exekveringsläge och kör programmet som förut. Nu kommer en röd punkt att dyka upp i graffönstret för varje gång du kör programmet.

Vad händer när du kört programmet 50 gånger, dvs när du är utanför fönstrets "räckvidd"? Fönstret rullar på och visar alltid de 50 senaste värdena. Det finns två andra sätt att visa data.

Byt till editeringsläge och håll ner höger musknapp. Gå in under menyn **Data operations** och sen **Update modes**. Byt mode till **Sweep chart** och kör programmet igen.

Obs! Naturligtrvis kan du göra graffönstret större genom att "dra" i ena hörnet.

Vi ska också titta på hur man kan visa två grafer samtidigt.

Som du ser har linjen i legendfönstret ett nummer, nummer 0. Låt muscursorn vara en pil. "Ta tag" i legend fönstrets nedre högra hörn och "dra" ner fönstret så att ytterligare en linje kommer fram. Den nya linjen är vit och har nummer 1. Nu har grafen möjlighet att displaya två insignaler.

Frågan är bara hur man matar in två insignaler till den. Vi återkommer strax till detta problem.

Byt till Diagramfönstret. Vi ska låta den andra signalen in till grafen vara en konstant, dvs den ska bara ge ett rakt streck i grafen.
Hämta en **Numeric Constant** under **Structs & Constants** menyn. Ge den värdet 0.7 och placera den strax under symbolen för slumptalsgeneratorn.

Nu kommer vi till det stora problemet, nämligen hur vi ska få grafen att förstå att vi vill att den ska visa två grafer samtidigt. Det bästa hade väl varit om man helt enkelt hade kunnat dra ytterligare en tråd från den nya konstanten och in på samma ikon som slumptalsgeneratorn går in på.

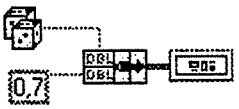
Så enkelt är det naturligtvis inte!

För att lösa problemet måste man använda sig av arrayer och dessa måste "bundlas" ihop och sen ska dessa "hopbundlade" arrayer matas in till grafikonen.

Det ligger utanför det här kompendiets ambitioner att förklara arraybegreppet. Vi ska bara lära oss hur det praktiskt går till.

Låt muscursorn vara en pil i **Diagramfönstret**. Placera muscursorn på tråden mellan tärningarna och grafikonen. Håll ner höger musknapp och välj **Insert** i den meny som dyker upp. Gå sen in i **Array & Clusters** menyn och välj **Bundle** operatorn. Släpp upp höger musknapp. Tråden från tärningarna går nu via **Bundle** operatorn till grafikonen. Samtidigt bytte grafikonen utseende. Det nya utseendet betyder kort sagt att den nu kan ta emot arrayer (=vektorer). Arrayer anges med brun färg i LabView.

"Ta tag" i **Bundle**operatorns nedre högra hörn och dra ner den så att ytterligare en ingång dyker upp. Dra sen en tråd från konstanten du tog fram tidigare till den nya ingången på **Bundle**operatorn. Ungefär så här ska det se ut:

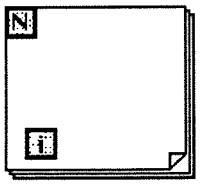


Gå över till Panelfönstret och testkör programmet om pilen är hel. Kör en exekvering i taget. Kontinuerlig exekvering går lite för snabbt för att man ska kunna se vad som händer.

Övning 3: Skapa ett VI som i en graf samtidigt visar värdet från tre stycken numeric controllers som finns i Panelfönstret. Graferna ska ha olika färg och "Line" interpolerade.

3. For loopen.

Under menyn **Structs & Constants** i Diagramfönsret finns **For Loopen** vars symbol ser ut så här:



Allt som finns inuti For Loopens ruta kommer att upprepas N stycken gånger. Till N knyts ett tal som kan vara både heltal eller flyttal, men flyttal kommer att avrundas till heltal. "i" är loopräknaren som talar om hur många varv loopen har gått.

Två saker ska man tänka på när man använder **For Loopar**:

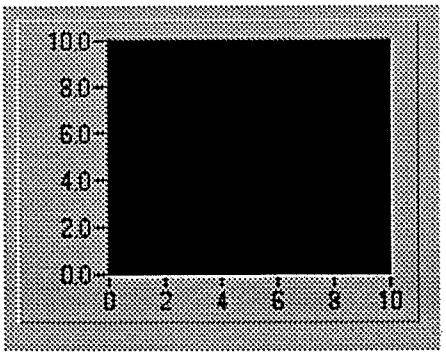
1) "i" är en lokal variabel och kan bara användas inuti For Loopen.

2) Ingenting "kommer ut" ur For Loopen förrän loopen har genomgått samtliga varv. Det innebär att om ett värde (t. ex. sin(i)) beräknas för i=0 till i=4, så kommer dessa fem resultat samtliga att komma ut som en array när loopen gått alla sina varv.

Observera också att om N=10 kommer variabeln "i" att gå från 0 till 9.

Vi ska åskådliggöra For loopens funktion med ett exempel. Gör först **Close** och **New VI** som vanligt för att få rena fönster.

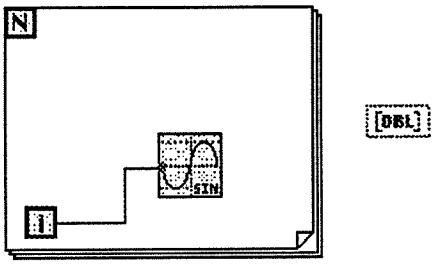
I Panelfönstret plockar du fram ett **Waveform Graph** under **Graph**menyn:



Byt sen till Diagramfönstret. Ta fram en **For loop** från **Structs & Constants**menyn. Gör **For loopen** större genom att "dra" i ett hörn. **Waveform Graph**symbolen ska vara utanför **For loopen**. Vi ska visa med det här exemplet dels att ingenting kommer ut ur For loopen förrän den gått samtliga varv i loopen och dels att när data väl kommer ut så kommer de ut som en array.

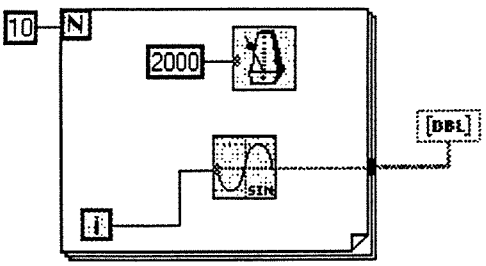
För att visa detta ska vi beräkna sinus av loopräknaren (dvs den lokala variablen i) tio gånger med en sekunds mellanrum. Resultatet ska presenteras på Waveform Graph som ligger utanför For loopen. Efter som den ligger utanför For loopen så kommer ingenting att synas i grafen förrän alla tio värdena har beräknats (efter 10 * 2 = 20 sekunder).

Inuti For loopen placerar du först en **sine** funktion som du hämtar under **Trig&Log**-menyn. Till sinusfunktionens ingång drar du loopräknarvariabeln i:



Hämta en **Numerisk konstant**, sätt den till tio och knyt den till **N**. Därmed har vi bestämt att **For Loopen** ska genomlöpas tio gånger. Dra en tråd från sinusfunktionens utgång till grafsymbolen utanför **For Loopen**. Observera att tråden är "tunn" inuti **For Loopen** och "tjock" utanför For Loopen. Det betyder att så länge vi befinner oss innanför **For Loopen** är det enstaka tal vi hanterar och **utanför For Loopen** är det en array med tal.

För att åstadkomma en fördröjning mellan beräkningen av varje värde lägger vi in en **Wait Until Next ms Multiple** och knyter till dess ingång en numerisk konstant med värdet 2000. Funktionen hittar du under **Dialog & Date/Time**. Så här ska den färdiga kopplingen se ut.



Om pilen är hel går du till Panelfönstret och kör programmet. OBS! Kör **inte** kontinuerlig exekvering utan bara en exekvering. Observera att ingenting händer förrän alla värden är beräknade, vilket tar ca 20 sekunder. Därefter kommer alla värden ut på en gång.

Fråga 2: Anta att man istället skulle vilja ha en omedelbar uppdatering i grafen, dvs varje beräknat värde ska visas direkt i grafen. Hur ska man bära sig åt för att åstadkomma detta? Räcker det med att bara flytta in graf symbolen inuti For Loopen? (Pröva gärna!)

.....

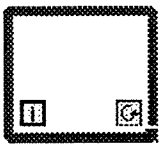
.....

.....

Övning 4: Skapa ett VI som beräknar naturliga logaritmen av talen 0, 0.5, 1.0, 1.5, ..., 49.0, 49.5, 50.0. Talen ska skrivas ut på en graf omedelbart efter att de beräknats.

4. While Loopen.

En annan loop som är mycket användbar är **While Loopen**. Har du programmerat i t. ex. Pascal tidigare känner du igen den här loopen. **While Loopen** utför något så länge ett logisk villkor är uppfyllt. While Loopen finns även den under Structs & Constants och ser ut så här:



Den lokala variablen "i" är en loop räknare precis som i **For Loopen**. Symbolen längst ner till höger är omringad av en grön ram och är därför en boolesk funktion. Till denna knyts en Boolesk variabel och tillståndet hos denna variabel kommer att avgöra om While Loopen kommer att göra fler varv. Observera att tillståndet på den booleska

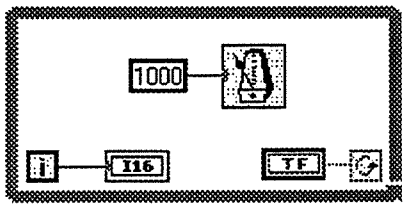
variablen kommer att läsas av **först** och beroende på dess tillstånd bestäms om nästa loop ska påbörjas eller om loopen ska avslutas.

Vi tar förstås ett exempel:

Rensa skärmen. I Panel fönstret lägger du in en **Boolesk Control** (med text på) och en **digital indikator**. Indikatorns talrepresentation ska vara **integer word**.

Avsikten med vårt program är att den digitala indikatorn ska räknas upp med ett varje sekund så länge som den booleska controllern är sann.
Gå till Diagramfönstret. Ta fram en **While Loop** och gör den lite större, ca 5x10 cm. Placera både den digitala indikatorn och den booleska controllern inuti **While Loopen**. Dra en tråd från loop räknaren "i" till indikatorn och dra en tråd från den booleska controllern till symbolen i **While Loopens** nedre högra hörn.

Hämta en **Wait Until Next ms Multiple** ur **Dialog & Date/Time menyn**. Knyt en numerisk konstant (=1000) till tidsfunktionens ingång. Så här ska det se ut ungefär:



Kontrollera att pilen är hel och byt sen till Panelfönstret. Byt till exekveringsläge men kör inte än. Klicka först en gång på den booleska controllern så att den blir "ON". Det betyder att villkoret för att while loopen ska fortsätta är uppfyllt när vi startar programmet.

Kör nu programmet, inte kontinuerligt, utan en gång. Programmet kommer nu att öka värdet i digitala indikatorn tills du klickar på controllerknappen så att den blir "OFF".

Fråga 3: Anta att tryckknappen från början varit "OFF". Hur många varv hade då **While Loopen** gått och vilket värde hade den digitala indikatorn stått på efter programmet?

.....

.....

.....

Övning 5: Gör ett VI som i panelfönstret har en **Tank indikator** vars högsta värde är 10000. Dessutom ska du ha en **Digital controll** vars tal representation ska vara **integer word**. I controllern skrivs ett värde in, t. ex. 2400, sen när programmet körs

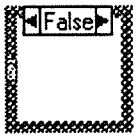
ska tanken så snabbt som möjligt fyllas upp till detta värde och sen ska programmet avslutas.

5. Case satsen.

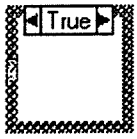
Case satsen känner du också igen om du programmerat i ett högnivåspråk som Pascal. Till **Case** satsen kommer en eller flera inparametrar som ska behandlas på något sätt. **Case** satsen erbjuder olika "behandlings alternativ" beroende på värdet hos **case variabeln**.

Case variablen kan vara heltal eller boolesk. I **LabView** är **case variablen** boolesk default, men kan lätt ändras till heltal.

Case satsen finns under **Structs & Constants** och ser ut så här:



Precis som med **For-** och **While Loopen** kan du "dra" i hörnen för att göra den större eller mindre. Längst upp i **Case** fönstret finns upp- och ner stegningspilar. Om du klickar en gång på en av pilarna kommer **Case** fönstret att se ut så här:

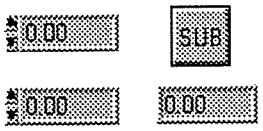


I fönstrets vänsterkant finns en liten ruta med grön ram, som är ingången **för case variabeln**. Till denna ingång knyts en logisk variabel. Om variablen är logisk ett kommer de operationer som du skrivit in i "TRUE" rutan att utföras annars kommer "FALSE" operationerna att utföras.

Rensa **LabView** fönstrena och lägg in en boolesk control, två digitala controller och en digital indikator i Panelfönstret. Välj en boolesk control med text på. När den booleska controllern är intryckt ska det stå "ADD" på knappen och när den inte är intryckt ska det stå "SUB" på den.

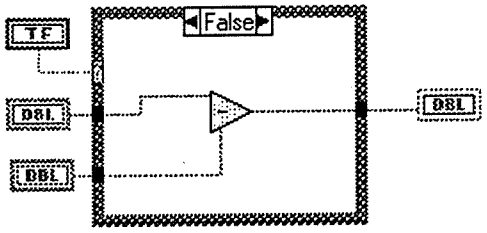
Vi ska göra ett VI där summan av eller differensen mellan de båda digitala controllerna kommer att stå i indikatorn. Om tryckknappen är intryckt är det summan som visas, om den inte är intryckt är det differensen som står i indikatorn.

Så här ska Panelfönstret se ut:



Byt till Diagramfönstret. Ta fram en **Case sats** och förstora den till ca 10x10 cm. Den booleska controllern och de båda digitala controllerna placerar du till vänster om **Case satsen** och den digitala indikatorn till höger om **Case satsen**.

Placera cursorn i **Case satsens** "FALSE" ruta och ta fram en subtraktionsoperator. Dra trådar från de digital controllerna utanför **Case satsen** till ingångarna på subtaktionsoperatorn inuti **Case satsen**. Dra sen en tråd från subtraktionsoperatorn utgång till den digitala indikatorn utanför **Case satsen**. Dra sen också en tråd från den booleska controllern till **case variabelns** ingång. Så här ska det se ut:



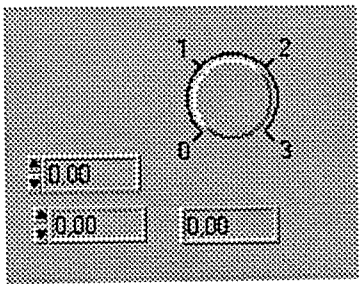
Klicka sedan på en av pilarna längst upp i Case fönstret så att "TRUE" rutan kommer fram. Observera i "TRUE" rutan att de ingångar du gjorde i "FALSE" rutan finns markerade även i den här rutan.

Placera en Addoperator i "TRUE" fönstret. Nu behöver du bara dra trådar från de markerade ingångarna i fönstrets vänstra kant till Addoperatorns ingångar. Samma sak med utgången; den behöver bara dra till den vita rutan i höger kanten som markerar utgången.

Kontrollera att pilen är hel och byt sen till Panelfönstret. Byt först till exekveringsläge. Innan du kör programmet matar du in två tal i de digitala controllerna. Kör sen programmet kontinuerligt och observera indikatorns värde när du trycker på ADD/SUB knappen.

Vi ska nu utöka vår Case sats så att vi även kan se produkten och kvoten mellan våra två inparametrar.

Byt till editeringsläge i Panelfönstret och placera cursorn på tryckknappen. Håll ner höger musknapp och välj **Replace** i menyn. Gå sen in under **Numeric** och välj **Dial**. En ratt dyker nu upp i Panelfönstret istället för tryckknappen. Ändra först dess talrepresentation till integer byte och låt sen dess max värde vara 3. Den ska alltså bara ge fyra stycken heltals värden; 0, 1, 2 och 3. Ta också bort den Digitala display som visas tillsammans med ratten. Hur du tar bort den? Det listar du ut själv.



Byt sen till Diagramfönstret. Observera att **Case variablen** har blivit blå och att det inte står "TRUE" eller "FALSE" längst upp i **Case satsens** fönster utan "1" eller "0". Det betyder att det nu är ett heltal som bestämmer vilken operation som ska utföras på inparametrarna. Eftersom det bara finns två olika logiska variabler kan man bara ha två case satser om case variablen är en logisk variabel. Eftersom vi nu istället har ett heltal som case variabel kan vi i princip har hur många case satser som helst (naturliga heltal finns det ju gott om, eller hur?).

Klicka på pilarna längst upp i **Case** fönstret så att ruta nummer "1" ligger uppe. För att lägga till ytterligare **Case satser** placerar du muscursorn på ettan och håller ner höger musknapp och väljer **Add Case After** i den meny som dyker upp. En ny, tom ruta dyker upp med nummer "2". I den här rutan placerar du en **Multiply**operator och drar lämpliga trådar till dess in- och utgångar. Gör på samma sätt för att lägga till en fjärde ruta med nummer "3" som innehåller en divisionsoperator.

När pilen är hel byter du till Panelfönstret och kör programmet. Kör programmet kontinuerligt och observera vad som händer när du vrider på vrid ratten.

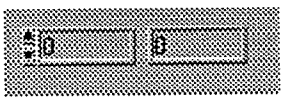
Övning 6: I Panelfönstret ska du ha en strängindikator och en numerisk control. Controllerns tal representation ska vara Byte integer och dess max värde ska vara +10 och dess min värde 0. (Detta sätter du i menyn Data Range...) Om värdet i den digitala controllern är "0" ska sträng indikatorn visa "Nolla", om värdet är "1" ska indikatorn visa "Etta" osv. Utnyttja en Case sats med elva case rutor för att lösa problemet.

6. Skift register.

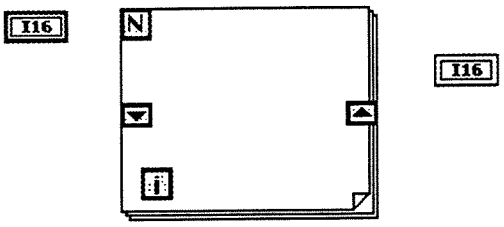
Vi ska avsluta labben med att visa en mycket användbar finess i samband med loopar, nämligen skiftregistret. Med hjälp av skiftregistret kan man inuti en loop få tillgång till värden som beräknats i en tidigare loop.

Vi ska i exemplet nedan beräkna summan av de N första heltalen, där N kan varieras via en Digital Control i Panelen och summan presenteras i en Digital Indicator.

Placera en **Digital Indicator** och en **Digital Control** i Panelfönstret. Låt båda ha tal-representationen **word integer**.



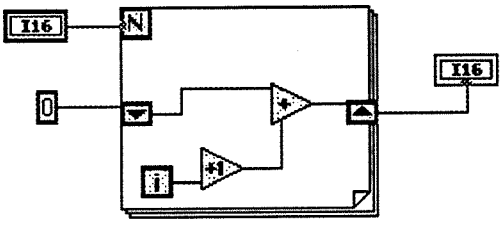
Byt till **Diagramfönstret**. Hämta en **For Loop** och gör den ca 10x10 cm stor. Placera controllern till vänster om **For Loopen** och indicatorn till höger om den. Sätt muscursorn på **For Loopens** högra ram kant och håll ner höger mus knapp. **Välj Add Shift Register** i menyn. Till **For Loopen** adderas nu en upp- och en ner pil på höger resp. vänster ram kant:



Uppilen på höger ramkant representerar det nya värdet och nerpilen representerar det värde som beräknades i **For** rutan i loopen innan.

Vi vill summera N stycken värden. I princip gör vi så här: Vi använder loopräknaren "i" och summerar dess värde varje varv. I varje varv får loopräknaren "i" och resultatet av summeringen i föregående varv gå till skiftregistrets upppil. Det betyder att detta värde finns tillgängligt i nerpilen till vänster vid nästa loopvarv. Låter det krångligt? Fullfölj exemplet nedan så kanske det klarnar. Be annars assitenten att förklara hur skiftregistret fungerar. Skiftregistret är ett mycket kraftfullt verktyg när man arbetar med loopar.

Lägg in en **Add**operator och en **Increment**operator inuti **For loopen**. Lägg slutligen en konstant med värdet noll till vänster om **For loopen**. Den får utgöra skiftregistrets initialvärde. Dra sen trådar enligt figuren nedan.



Byt till Panelfönstret om pilen är hel och provkör programmet. Mata in N i controllerfönstret och exekvera. Försäkra dig om att det stämmer.

Fråga 4: Varför måste man ha en Increment funktion mellan loop räknaren "i" och additions operatorn?

.....

.....

Fråga 5: Anta att N har fått värdet 20. När For loopen kommer till sitt fjärde varv, dvs i=3, vilket värde har då skiftregistrets nerpil i vänster kanten och vilket värde har upppilen just när fjärde varvet har avslutats?

.....

.....

Fråga 6: Varför måste skiftregistret ha ett initialvärde? Vad händer om du tar bort konstanten "0" och ändå kör programmet?

.....

.....

.....

Observera följande:

1) Även om indikatorn är ansluten till skiftregistrets uppil, som i exemplet ovan, så kommer ändå ingenting ut ur **For loopen** förrän hela **For loopen** är avslutad.

2) I vissa fall kan det vara önskbart att ha tillgång till ännu "äldre" beräkningar än bara det som gjordes i loopen närmast innan. Detta kan man råda bot på genom att placera cursorn på skiftregistrets nerpil i vänsterkanten och hålla ner höger musknapp. I den meny som dyker upp väljer man **Add Element**. Då dyker en ny nerpil upp under den tidigare. Den nya nerpilen representerar det värde som beräknades två loopar tidigare!

3) Även **while loopen** har möjlighet att utnyttja skift register.

Övning 7: Gör ett VI som beräknar fakulteten av ett tal N, dvs N!. N matas in via en digital control i Panelfönstret och fakulteten ska visas i en digital indikator. du ska lösas uppgiften med hjälp av skiftregister.

7. Mer övningar.

Följande övningar i kompendiet "Mäta och styra med persondatorer, Övningar" av Boman, Eriksson, Gustafsson och Eriksson, behandlar de avsnitt vi gått igenom ovan:

2.4-2.8, 3.4-3.7

Programmera i LabVIEW

Del 3

Instrumentkommunikation med GPIB och Filhantering.

Lars Bengtssons ursprungliga text har omarbetats och innehållet moderniserats av Lars Hellberg i januari 2007.

1. GPIB-bussen.

GPIB-bussen kallas ofta också för HPIB-, IEEE-, eller IEC-bussen. Alla dessa namn avser samma buss. GPIB-bussen är en bussstandard, ursprungligen utvecklad av Hewlwtt-Packard i avsikt att samtidigt kunna styra flera instrument via en kontrollenhet. Varje instrument på bussen har en unik adress och kommunikationen på bussen sker i "klarspråk" (nåja!), dvs m.h.a. ASCII-strängar. Såväl data som kommandon och adresser skickas i ASCII-format.

Spridningen av GPIB växte snabbt och blev redan på 70-talet antaget som IEEEstandard. Standarden definierar allt från mekanisk utformning av kontakter till formatet på de elektriska signalerna på bussen. Sedan den första standarden antogs har en revidering av standarden gjorts. Den gamla standarden kallar vi här för "Traditional GPIB" och den nya för "488.2". Vi kommer i labben att blanda den nya och den gamla stadarden. För att den nya standarden ska kunna användas krävs att instrumentet som ansluts har försetts med den nya standarden.

Kontrollenheten utgörs idag så gott som alltid av en persondator, PC eller MAC, försedd med ett GPIB-interface.

2. GPIB och LabView.

Eftersom kommunikationen mellan instrumenten och kontrollenheten på GPIB-bussen sker i ASCII-format (dvs med textsträngar) betyder det att LabView, med sina kraftfulla strängfunktioner, lämpar sig utmärkt som kommandospråk i en GPIBkontroller. Detta är naturligtvis ingen tillfällighet eftersom det var just med tanke på GPIB-bussen som LabView utvecklades.

3. De vanligaste GPIB-kommandona.

Om du håller ner höger musknapp när du befinner dig i Diagramfönstret, finns en rad som heter **488** i **Instrument I/O** menyn som dyker upp. Om du väljer denna rad dyker ytterligare en meny upp där du kan välja mellan den gamla och den nya standarden:

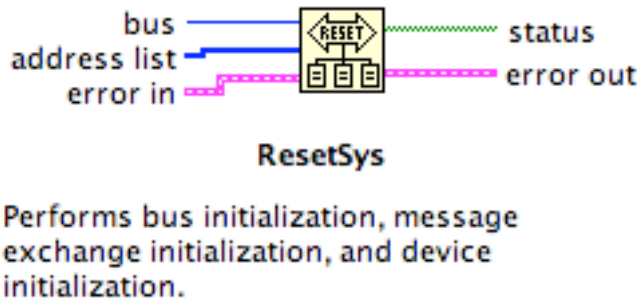
”Traditional” GPIB
488.2

Stannar du i **488** ser du ca tio kommandon. Här kommer vi att använda två av dessa kommandon:

GPIB Write.vi
GPIB Read.vi

Från **488.2**-menyn kommer vi endast att använda **ResetSys.vi**-kommandot. Med hjälp av dessa tre kommandon kan vi faktiskt göra väldigt mycket.

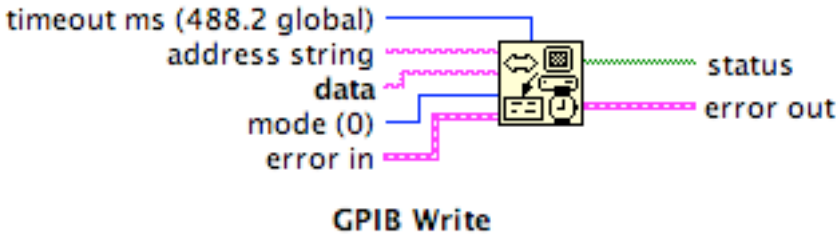
Här följer en kort beskrivning av dessa tre kommandon:



Det här kommandot initierar GPIB-bussen och instrumenten anslutna till bussen. Observera att en kontrollenhet (datorn) kan ha mer än ett GPIB interface vilket i praktiken betyder att en kontrollenhet kan styra mer än en GPIB-buss. När man exekverar **ResetSys**-kommandot måste man därför ange vilken buss man avser. Finns det bara en buss har den nummer 0 och behöver inte anges..

Till ingången **adress list** ska adresserna anges till de instrument som ska initieras. Observera att denna ingång är en arrayingång! (Finns det bara ett instrument på bussen klara man sig oftast med funktionen **DevClear.vi** i stället)

De två utgångarna **status** och **GPIB error** är typiska för alla GPIB-kommandon i LabView. **GPIB error** är en utgång av integer typ som anger vilket fel som uppstått (om ett fel uppstått). Ett fel har uppstått om bit 15 i statusarrayen är "true". De andra bitarna i statusarrayen anger olika tillstånd på bussen som t.ex. Listner Active, Talker Active, LockOut State, TimeOut State etc. Vi kommer inte att använda oss av dessa utgångar här.



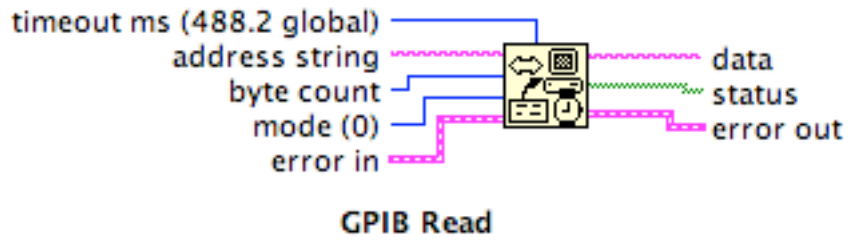
Detta kommando används för att sända kommandon till instrumentet. På adressingången anger man instrumentets GPIBadress och observera att detta är en textingång.

data är också en textingång. Textsträngen ska innehålla de kommandon och data som man vill skicka till instrumentet. Kommandosyntaxen anges i instrumentets manual.

På modeingången ska man ange hur Write-kommandot ska avslutas. Det finns tre olika alternativ för att avsluta kommandosträngen (EOI är en särskild ledning i GPIB-sladden):

0: Skicka EOI efter sista tecknet (Default). Oftast OK.
1: CR + EOI
2: LF + EOI
3: CR+LF+EOI

Dessa tre möjligheter täcker behovet för de flesta instrument. Vilken avslutning man använder beror förstås på vad instrumentets manual anger.



Reads **byte count** number of bytes from the GPIB device at **address string.**

adress string ingången har samma funktion som för Write-kommandot.

mode anger vilket EOS-tecken som förväntas (End Of String) enligt tabellen nedan

0: Inget EOS. Read avslutas med byte count eller EOI
1: EOS är CR. Read avslutas med CK byte count eller EOI
2: EOS är LF. Read avslutas med LF, byte count eller EOI

På **byte count** ingången anger man hur många bytes som ska tas emot. Mottagningen avslutas när detta antal tecken har mottagits eller när ett avslutningstecken detekterats. Här väljer man normalt ett tal som är mtcket större än det antal bytes man förväntar sig som svar, t.ex. 1000

data string utgången innehåller ett antal data bytes från GPIB instrumentet. Denna utgång är en strängutgång och resultatet kan se ut så här:

"VDC 3.049E-3"

Ovanstående är "klarspråk" för den som är van vid att läsa GPIB-språket men kan verka kryptiskt i början. Strängen kommer från en voltmeter som ombetts mäta sin spänning. "VDC" betyder likspänning och "3.049E-3" betyder 0.003049. Spänningen på voltmetern var alltså 3.049 mV (DC).

Typiskt för en mottagen datasträng är att man måste vidta två åtgärder för att extrahera de data man vill ha:

1) Man måste "skala bort bokstäverna" i textsträngen så att bara "siffrorna" återstår.

2) Konvertera den återstående textsträngen med "siffror" till ett tal.

Fråga 1: Försök att erinra dig vad du gjort i de tidigare labbarna. Kan du föreslå någon lämplig strängfunktion som skulle kunna "skala bort" bokstäverna "VDC" i strängen ovan?

……

Fråga 2: Föreslå någon funktion i LabView som omvandlar de återstående "siffrorna" i strängen till ett "tal".

……

4. Noname System Multimeter MM1234 (ett påhittat instrument)

För att kunna kommunicera med multimetern via kontrollenheten måste man först och främst veta vilken adress instrumentet har. På äldre instrument ställs adressen in "hårdvarumässigt" med hjälp av små binärswitchar. På moderna instrument kan man ställa in adressen "mjukvarumässigt" i en meny på displayen. Vårt instrument är av den senare modellen. På frontpanelen finns en knapp som heter "CHECK" (för andra instrument. Tryck in den tre gånger. Då står det på displayen t. ex.:

Addr 10

Detta är instrumentets adress på GPIB-bussen. Man kan nu också ändra adressen om man vill. När man tryckt in "CHECK" knappen motsvarar den undre knappraden siffrorna 0-9 (blå). Med hjälp av dessa slår man in den nya adressen som kan ligga mellan 0 och 30. När du antecknat adressen trycker du på "END" knappen så att instrumentet återigen visar aktuellt mätvärde.

Vidare måste man studera instrumentets manual för att lära sig vad det kan göra och hur syntaxen ser ut. Detta är ofta både svårt och tråkigt.

Vårt instrument kan mäta totalt sex olika storheter, nämligen lik- och växelspänning, lik- och växelström, resistans samt temperatur. Först måste instrumentet ställas in så att det mäter den storhet man vill mäta. Detta gör man genom att skicka ett kommando till instrumentet. Vilka kommandon som används ser du nedan:

	Kommando
Likspänning	VDC
Växelspänning	VAC
Resistans	RTW
Likström	mC
Växelström	IAC
Temperatur	TDC

Man kan också ange vilket mätområde man vill att instrumentet ska mäta inom. T. ex.

kommer kommandot "VDC 300" att ställa instrumentet på likspänningsmätning upp till 300 V. Anger man inget område är det automatisk inställning som gäller (AUTO). Vid en förvald inställning går mätningen snabbare.

OBS! Om man skickar kommandot som det står ovan så beordrar man instrumentet att utföra mätningen och visa resultatet på instrumentets display. Om man vill läsa ut resultatet till datorn med GPIB måste man addera ett frågetecken till kommandot, t.ex. VDC?

Nu har vi nämligen alla byggstenar vi behöver för att göra första exemplet.

5. Första exemplet.

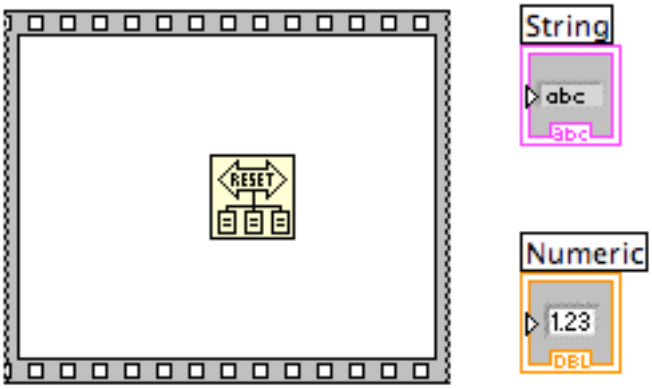
Vi börjar med ett mycket enkelt exempel där vi skall mäta växelspänningsmätning.

Programmet kommer att bestå av fyra stycken sekvensrutor. I ruta nummer 0 initieras instrumentet. I ruta nummer 1 sänds ett kommando till multimetern så att den ställs in för växelspänningsmätning. I ruta nummer 2 triggas instrumentet till att ta ett mätvärde och i sista rutan sker lite textbehandling innan mätvärdet presenteras.

Se till så att du har ett nytt, rent Panelfönster. I Panelfönstret lägger du in dels en **string indicator** och en **numeric indicator.** "Dra ut" **string indicatorn** så att den blir ungefär dubbelt så lång som från början:

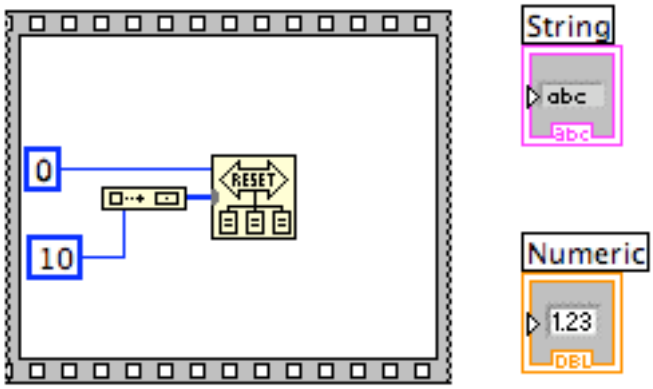
Byt till Diagramfönstret. Lägg undan symbolerna för de två indikatorerna i vänstra kanten tills vidare. Aktivera också Hjälpfönstret.

Ta fram en sekvensruta och gör den lite större. Placera en **ResetSys** inuti rutan (som hämtas i **488.2**menyn):



I rutan placerar du två konstanter den ena med värdet "0" och den andra med instrumentets adress som du antecknade tidigare. Observera att adress ingången måste vara en array!

Hämta därför operatorn **Build array** i menyn **Array&Cluster**. Gör operatorn "endimensionell" genom att dra in ena hörnet om det behövs. Till dess ingång drar du adresskonstanten och dess utgång drar du till **adress**ingången på **ResetSys**. Till **bus**ingången drar du konstanten "0":



Det här är allt vi ska göra i ruta 0. I och med att den här rutan exekveras är GPIB bussen och instrumentet (med adress 10) initierade. Vi går till nästa ruta!

Placera muscursorn på "0":an längst upp i sekvensrutan och håll ner höger musknapp och välj **Add Frame After**. Då kommer nästa sekvensruta upp med nummer 1. I den här rutan ska vi skicka ett kommando till multimetern så att den ställs in för mätning av växelspänning.

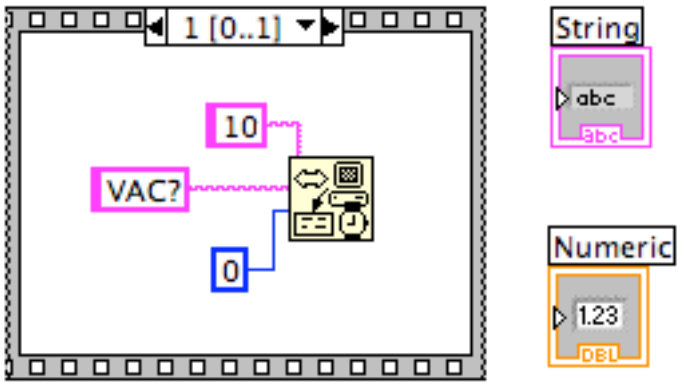
Eftersom vi ska skicka ett kommando till multimetern är det kommandot **GPIB-Write** vi ska använda.

Placera muspilen i sekvensruta 1. Håll ner höger musknapp och hämta operatorn **GPIB-Write** i Traditionalmenyn.

Till **adress**-ingången knyter du en strängkonstant som innehåller instrumentets adress.

Till **mode**-ingången knyter du en "0":a (vilket är default-värdet).

data string ska vara en strängkonstant innehållande kommandot som betyder "växelspänning", dvs "VAC?". Så här ska det se ut: 1



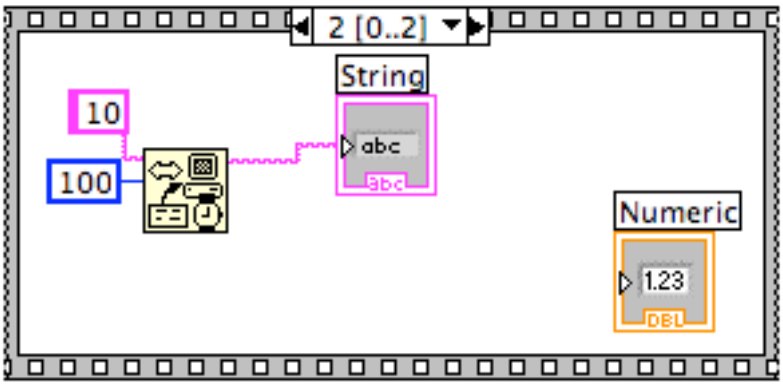
Nu är vi färdiga i ruta 1. Placera muscursorn på "1":an längst upp i rutan och håll ner höger

tangent. Välj **Add Frame After** som du gjorde förut för att lägga till ytterligare en sekvensruta. Den nya rutan ska ha nummer 2.

I ruta 2 ska vi ta emot det mätvärde som multimetern registrerade i ruta 1. Med kommandot **GPIB-Read** beordras instrumentet att skicka sitt mätvärde till datorn. Placera cursorpilen i ruta 2 och hämta **GPIB-Read**-kommandot (du vet var, eller hur?)

address-ingången ska ha samma värde som tidigare. **mode**-ingången lämnar vi nu öppen. Till **count**ingången knyter du en konstant = 100. Gör sekvensrutan lite större och aktivera Hjälpfönstret om du inte gjort det tidigare. Flytta in strängindikatorn och digitalindikatorn i sekvensrutan.

GPIB-Readoperatorns **data string**utgång kan du dra direkt till strängindikatorn (det är ju en text som kommer från instrumentet, eller hur?).



På den digitala indikatorn ska vi så småningom visa bara mätdata, d.v.s. ett tal utan vidhängande text. För att veta vilka textoperationer vi måste göra för att åstadkomma detta måste vi först köra programmet en gång och studera exakt hur textsträngen från instrumentet ser ut!

Kontrollera att "pilen är hel" och byt sen till Panelfönstret. Kör programmet en gång, dvs inte kontinuerlig exekvering. Studera samtidigt displayen på multimetern och observera hur den "blinkar till" när den mäter.

Det kan hända att strängindikatorn är för kort för att ta emot hela textsträngen från instrumentet. Dra för säkerhets skull ut den så att du är säker på att ingenting döljer sig i dess högra ände. Texten i fönstret kan se ut så här:

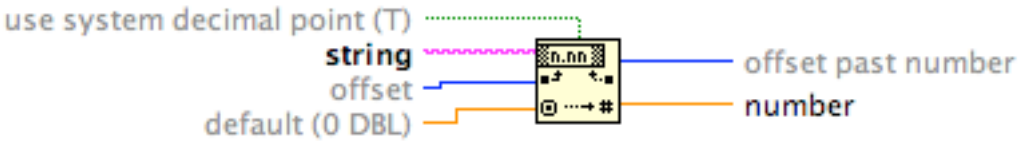
VAC 09,0346E-03

Detta är alltså en textsträng. Vår nästa uppgift blir att extrahera mätdata ur denna sträng. Mätdata utgörs av talet 0,0090346. Observera att det är talet 0,0090346 och inte texten 0,0090346 vi är ute efter!

Två saker behöver vi göra med textsträngen. Vi måste först skala bort den inledande texten "VAC" och sen omvandla resterande text till ett tal.

I LabView finns en standardfunktion för båda dessa behov. Skrivsättet med 10-potens kallas i LabViewspråk för "exponential", "fractional" eller "engineering". Det finns en

funktion i LabView som omvandlar en text skriven på det här sättet till motsvarande tal. Den heter förstås **From Exponential/Fract/Eng:**

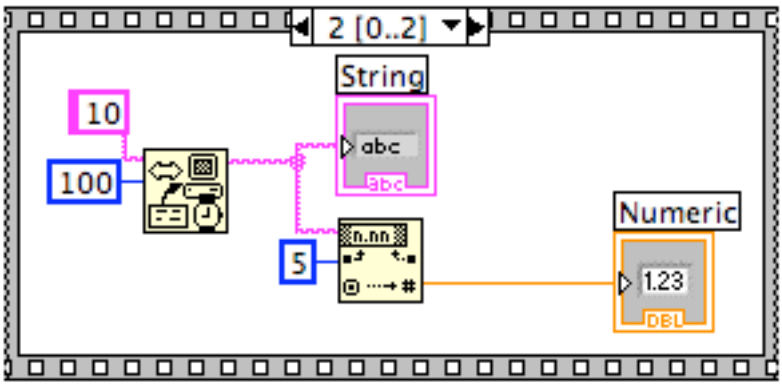


Fract/Exp String To Number

Interprets the characters 0 through 9, plus, minus, e, E, and the decimal point (usually period) in **string** starting at **offset** as a floating-point number in engineering notation, exponential, or fractional format and returns it in **number**.

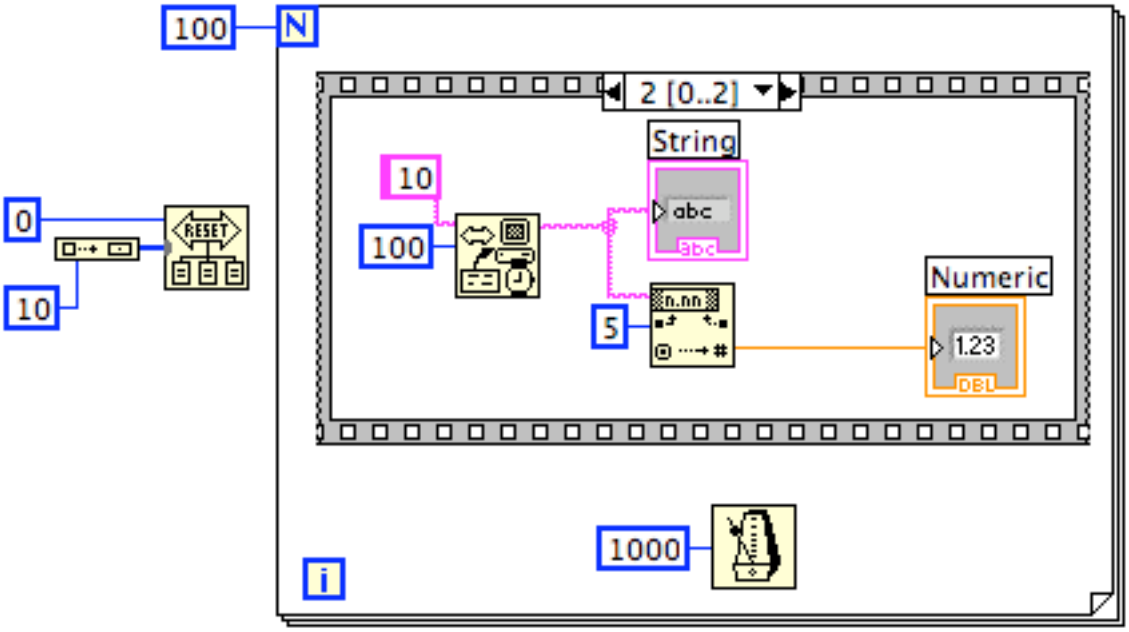
Funktionen tittar på den inkommande textsträngen (= **string**) och räknar ett visst antal tecken (= **offset**). Efter offset ska "texttalet" komma som den ska omvandla till ett tal. Texttalet ska vara skrivet på tiopotens eller decimalform och resultatet av omvandlingen kommer ut på **number**utgången.

Lägg in den här funktionen i sekvensruta 2 och dra en tråd från data stringutgången på **GPIB-Read** operatorn till ingången på **From Exponential/Fract/Eng**. Utgången kopplar du till den digitala indikatorn:



Klart! Byt till Panelfönstret. Kör programmet och observera digitalindikatorn.

Behåll programmet ovan och gå in i Panelen. Byt ut den digitala indikatorn (Högerklicka på indikatorn och välj **Replace**) mot en **Waveform Chart** med **Autoscale** på X- och Y-axeln. Byt sen till Diagramfönstret. Öppna en **For Loop** och placera hela ditt gamla program inuti **For Loopen**. Inuti **For Loopen** lägger du också en **Wait Until Next ms Multiple** och knyter en konstant = 1000 till dess ingång. Till **For Loopens N**-värde knyter du siffran 100. Obs! Det finns ingen anledning att reseta systemet före varje mätning. Funktionen **ResetSys.vi** kan man därför lägga utanför **For Loopen**. Den kommer då att utföras precis en gång innan **For Loopen** exekveras. Flytta ut den ur sekvensruta 0 och ta sen bort ruta 0.



Programmet kommer att ta ett mätvärde varje sekund och hundra värden kommer att registreras. Värdena presenteras i grafen på panelen. Kör programmet en gång, d.v.s. inte i kontinuerlig. "Mätningen" kommer att ta 100 sekunder.

Gör inte Close ännu. Vi ska strax utöka programmet ovan, så behåll programmet ett tag till!

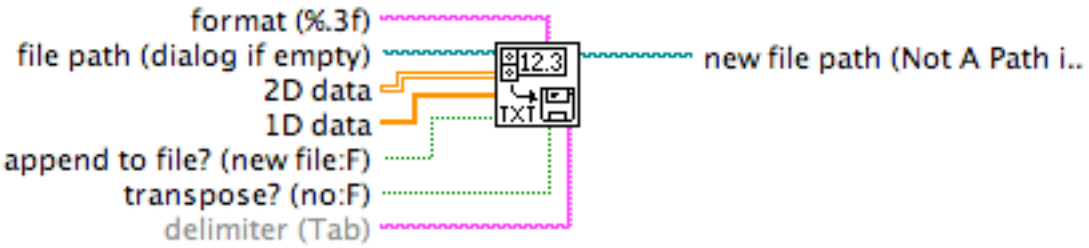
6. Lagra data.

Vid datainsamling måste man naturligtvis lagra sina data någonstans, t.ex. på hårddisken eller ett USB-minne. Man säger att man lagrar data som en fil. LabView har färdiga funktioner för filhantering

LabView kan inte bara skriva till en fil, man kan också bestämma hur datamängden skrivs till filen.

Kanske vill man senare presentera sina data i en snygg rapport. Rapporten skriver man i ett ordbehandlingsprogram (t. ex. WORD) och kurvan ritar man i ett spreadsheetprogram (t. ex. EXCEL). Ett spreadsheetprogram är ett program som har mycket kraftfulla verktyg för att presentera kurvor utgående från en datamängd lagrad i någon matrisform, som i sin enklaste form är en enda kolumn med data.

Den breda användningen av spreadsheetprogram har konstuktörema av LabView tagit fasta på. I LabView finns en funktion som omvandlar en array med tal till spreadsheetformat och lagrar det nya formatet i en fil. Filnamnet anger man via en dialogruta. Funktionen heter **Write To Spreadsheet File. Notera att** denna funktion endast hanterar numersika textsträngar och inte generella textsträngar.



Macintosh HD:...EW 7.1:vi.lib:Utility:file.llb:Write To Spreadsheet File.vi

Converts a 2D or 1D array of single-precision numbers to a text string and writes the string to a new byte stream file or appends the string to an existing file.

I formatsträngen anger man hur data ska lagras (jämför med funktionen (Format & Append). Defaultvärdet (dvs det värde som används om inget annat anges av användaren) är

%.3f

och betyder att data lagras i flyttalsform med tre decimaler (men dock som text!).

Till ingången som heter **file path (dialog if empty)** knyter man namnet till den fil där man vill lagra data. Ett alternativ är att inte ange någonting alls här. Då kommer det upp en dialogruta på skärmen när man kör programmet där man får ange filnamnet.

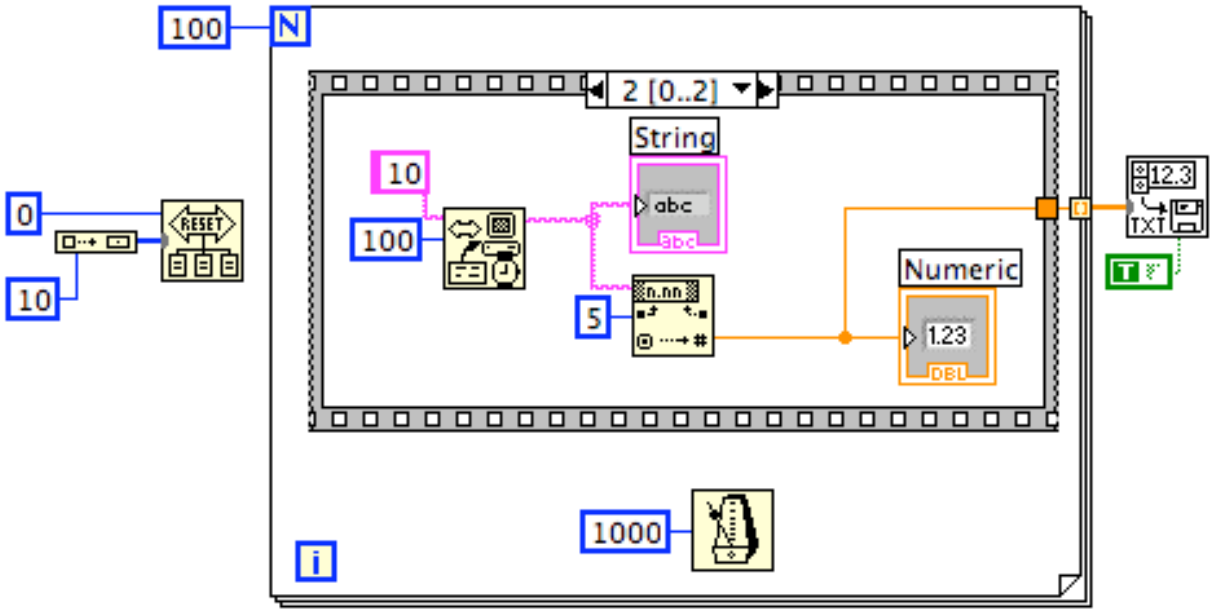
Data man vill spara knyts endera till **2D data** eller **1D data** ingången beroende på om datamängden lagrats i en endimensionell eller tvådimensionell array.

Man kan också lägga till den nya datamängden till slutet på en redan befintlig fil genom att knyta en boolesk konstant "TRUE" till ingången **append to file? (new file:F).** Default skapas dock en ny fil.

Den sista ingången, som heter **transpose?**, är mycket användbar. En datamängd som lagras i en array i LabView och som skrivs direkt till en datafil kommer att lagras utan "CR"-tecken (Carriage Return) mellan varje element. Dataelementen kommer då att lagras i en enda lång rad. De flesta spreadsheetprogram vill dock hellre läsa in datamängden i kolumner. Genom att välja att transponera så kommer data att lagras i kolumner. (Jämför matristransponering inom matematiken!)

Vi ska använda funktionen **Write To Spreadsheet File** för att lagra våra mätdata som vi får från voltmetern. Hämta **Write To Spreadsheet File** funktionen och placera den utanför **For-Loopen**. Funktionen hittar du under Filemenyn.

Till transpose? ingången knyter du en boolesk konstant som är "TRUE". Mätdataarrayen som kommer ut från **For-Loopen** knyter du till **1D data**ingången:



Ni kan simulera denna del genom att ta bort hela sekvensstrukturen samt reset-funktionen och i stället tråda loopräkaren filtunneln.

Förhoppningsvis har ni nu skapat en datafil som innehåller era mätdata. För att kontrollera att filen finns och innehåller "rätt" data kan man öppna filen i ett annat program som kan läsa textfiler.

7. Kommentar: Alternativ programmeringsmetod.

Sekvensstrukturen som använts ovan nyttjas för att styra exekveringsordningen. Man måste ju t.ex. först skicka ett kommando innan man kan läsa in svaret. Det är inget funktionellt fel att använda sekvensstrukturen med den ger dålig överblick över programmet. Det är oftast bättre att använda data flödet i Errorklustret för att styra exekveringsordningen. Se exemplet nedan.