

Rendszerarchitektúrák laboratórium

Tárgyfelelős: dr. Fehér Béla

Virtuális műszerek mérés sorozat

v 0.1

1. mérés: *LabVIEW alapismeretek (1.-3. fejezet)*
2. mérés: *Ismerkedés a CompactRIO rendszerrel. (4. fejezet)*
3. mérés: *Összetett önálló feladat (5. fejezet)*

BME MIT 2009.

© Scherer Balázs, dr. Tóth Csaba, Molnár Károly

1.	BEVEZETÉS.....	3
1.1	VIRTUÁLIS MŰSZEREK.....	3
1.2	A GRAFIKUS PROGRAMOZÁSRÓL	5
2.	ALAPISMERETEK	6
2.1	LABVIEW VI FUTTATÁSA, ALAPFOGALMAK	6
2.2	STÁTUS- ÉS ESZKÖZSOR	7
2.3	EGY EGYSZERŰ PÉLDAPROGRAM LÉTREHOZÁSA	8
2.4	VEZÉRLÉSI SZERKEZETEK (STRUCTURES) ÉS ADATTÍPUSOK.....	11
2.4.1	<i>Ciklusok</i>	11
2.4.2	<i>Sorrendi struktúrák</i>	14
2.4.3	<i>Választás (Case) struktúra</i>	15
2.5	EGYÉB FONTOS LABVIEW ELEMELK.....	15
3.	ISMERKEDÉS A VIRTUÁLIS MŰSZEREKEL	17
3.1	AZ NATIONAL INSTRUMENTS MŰSZERKÍNÁLATÁNAK RÖVID ÁTTEKINTÉSE	17
3.2	FÜGGVÉNYGENERÁTOR KÉSZÍTÉSE A PC HANGKÁTYA SEGÍTSÉGÉVEL.....	19
4.	ISMERKEDÉS A COMPACT-RIO RENDSZERREL	20
4.1	A COMPACTRIO FEJLESZTŐKÖRNYEZET	20
4.2	FEJLESZTÉS A COMPACTRIO EGYSÉGRE	22
4.3	SUGÁRZÁS MONITOROZÓ KÉSZÍTÉSE	32

1. BEVEZETÉS

1.1 Virtuális műszerek

- Műszerek** A gyors technológiai fejlődés ellenére az elektronikus műszerek funkcionális felépítése lényegében nem változott az elmúlt évtizedekben. Minden műszer blokkvázlata nagyon hasonló: mindegyiknek van bemeneti egysége, feldolgozó egysége és megjelenítő egysége. A műszerek fajtájától és az alkalmazott technológiától függően, természetesen, az egységek bonyolultsága nagyon változó. Egy egyszerű műszerben a feldolgozó lehet egy analóg átlagoló, egy bonyolultabban lehet például egy FFT-t számoló célhardver; a megjelenítő lehet egy mutató műszer, de lehet egy grafikus kijelző is.
- Adatgyűjtők/vezérlők** Ha a műszerek körét tágítjuk, és figyelembe vesszük az adatgyűjtő/vezérlő berendezéseket is, akkor a fenti blokkvázlatot csupán kimeneti/beavatkozó egységgel kell bővítenünk.
- „Műszermag”** A műszerek és az (egyszerűbb) adatgyűjtő/vezérlők lényegében csupán a bemeneti és kimeneti egységeikben különböznek, a feldolgozó és megjelenítő egységeik nagyon hasonlóak, vagyis van egy jelentős közös részük. Ezt a feldolgozó, megjelenítő közös részt „műszermagnak” is nevezhetjük.
- Egy univerzális „műszermag” felhasználásával sokkal gyorsabban és költséghatékonyabban tudunk műszereket, berendezéseket kifejleszteni, mint egyedi tervezéssel.
- „Mikroprocesszoros...”** A „műszermagok” univerzális megvalósítását a mikroprocesszorok megjelenése tette lehetővé. Az 1970-es évek végén és az 1980-as években készített műszerek neve gyakran így kezdődött: „Mikroprocesszoros...”, hasonlóan ahhoz, ahogy az 1960-as, 70-es években a „Tranzistoros...” jelzővel mutatták a készülékek korszerűségét. (Itt jegyezzük meg, hogy a gyakran használt „korszerű” szó nem a termék/rendszer minőségére utal, hanem csupán arra, hogy az a valami az adott korszak technikai színvonalának megfelelő. Ettől az még lehet jó is, de akár rossz is.)
- Beágyazott rendszerek** Lényegében a mikroprocesszorok tömeges elterjedésétől kezdve beszélhetünk beágyazott rendszerekről – noha ez a kifejezés csak jóval később, nálunk nagyjából 2000-tól terjedt el –, hiszen a mikroprocesszorok tették lehetővé, hogy komplex feldolgozó, vezérlő funkciókat építsünk a műszereinkbe (elfogadható áron), és így teljesüljön a beágyazott rendszerek definíciója: Beágyazott rendszerek nevezzük azokat a processzor alapú eszközöket, illetve az ezekből alkotott rendszereket, amelyek képesek a befogadó

fizikai/kémiai/biológiai környezetüket érzékelők (és beavatkozók) segítségével autonóm módon megfigyelni (és befolyásolni).

- Moduláris építkezés** A hardver tipizálásával, modulokra bontásával könnyen és gyorsan lehetett egyedi műszereket, készülékeket létrehozni. A hardvermodulokból összeállított berendezések testre szabása lényegében szoftverben történt. Megjegyzendő, hogy a hardver építőkockáihoz hasonlóan a szoftvert is fel lehetett bontani tipikus szoftvermodulokra, így a szoftverfejlesztés is nagyban felgyorsult.
- PC-s korszak** Az asztali számítógépek és különösen a PC-k megjelenése (nálunk az 1990-es évek elejétől) a készülékek konstrukcióját is alaposan megváltoztatta. A PC lényegében egy (viszonylag) olcsó, univerzális „műszermag”, komoly processzálási kapacitással, nagy memóriával és háttértárral, gazdag ember–gép kapcsolattal (klaviatúra, egér; grafikus kijelző, opcionálisan nyomtató) és többféle kommunikációs interfésszel (korábban csak soros, párhuzamos interfész, később Ethernet, USB stb.). Csupán a feladathoz illeszkedő I/O perifériákkal kell kiegészíteni, és kész a műszer vagy adatgyűjtő/vezérlő hardvere és alapszoftvere. A készülékfejlesztés nagyobbik része az alkalmazói szoftver elkészítését jelenti.
- Virtuális műszerek** Az 1990-es évektől kezdve a PC-alapú készülékfejlesztés nagyjából két irányvonalat követett. Az egyik főleg a hagyományos készülékgyártókra jellemző: megőrizték a szokásos készülékformát, kezelőfelületet, és a doboz belsejébe „elrejtették” a PC-t (speciális, ipari PC-alaplapot és perifériákat). A másik irányvonal standard asztali PC-t használ, de kiegészíti analóg és digitális I/O perifériákkal és alkalmas szoftverrel. Ez utóbbi megoldás vezetett a „virtuális műszer” fogalomhoz. Az univerzális „műszermagból” (PC-ből) és a kiegészítő, szintén univerzális perifériákból sokféle konkrét „műszer” alakítható ki, attól függően, hogy éppen milyen alkalmazói szoftver fut a „műszermagon”, vagyis a számítógépen. Ugyanazon a hardver- és alapszoftver-készleten – az alkalmazói szoftver cseréjével – sokféle „műszer” implementálható: multiméter, függvénygenerátor, oszcilloszkóp, spektrumanalizátor stb. Az így előálló „műszer” valódi abban az értelemben, hogy funkcionálisan megegyezik egy hasonló célműszerrel, de „virtuális” abban az értelemben, hogy a fizikai megjelenése lényegesen eltér azokétól.
- National Instruments** A „virtuális műszer” koncepció egyik vezető képviselője az amerikai National Instruments (NI). Az 1976-ban alapított cég analóg és digitális I/O perifériákat kezdett gyártani számítógépekhez majd PC-khez beépíthető kártya és külön bedobozolt egységek formájában. Az NI készülékeit – noha nem a legolcsóbbak – az iparban széles körben használják. Az NI egyik hardvergyártó bázisa Debrecenben van.
- LabVIEW** A National Instruments „virtuális műszer” koncepciója alkalmas szoftverrendszer nélkül aligha ért volna el átütő sikert. A moduláris és univerzális hardverelemekhez szintén moduláris és univerzális programozási nyelvre és környezetre van szükség. Azt is figyelembe kellett venniük, hogy a felhasználók jelentős része nem programozó, nem is villamosmérnök vagy éppen informatikus, tehát a nyelv és a

programozói/fejlesztői környezetnek kevés programozói előismeret birtokában is használhatónak kell lennie. Az NI többféle szoftverrendszert is kifejlesztett a hardvereihez; a legnépszerűbb a LabVIEW (Laboratory Virtual Instrument Engineering Workbench).

A LabVIEW adatfolyam alapú, magas szintű grafikus programozási nyelv és környezet, amely az elmúlt húsz év során jelentős fejlődésen ment keresztül. Jelenleg (2008) a 8.5.1 verziónál tartanak. Általános programozási nyelvként is használható, de igazi erősségét virtuális műszerek létrehozásakor mutatja meg, különösen NI perifériák alkalmazása esetén.

1.2 A grafikus programozásról

Szöveges programozás A villamosmérnök- és az informatikushallgatók általában szöveges programozási nyelveken keresztül ismerkednek meg a számítógépek programozásával; korábban a BASIC vagy a Pascal, ma inkább a C és a Java nyelv fogalmkészletét ismerik meg először, így a későbbiekben ezekhez viszonyítják a többi programozási nyelvet, környezetet is.

Grafikus programozás A LabVIEW egy grafikus programozási nyelv és fejlesztőkörnyezet, és mint ilyen, teljesen másfajta gondolkodásmódot kíván, mint a hagyományos procedurális nyelvek vagy az objektumorientált nyelvek. Természetesen a LabVIEW-n kívül más grafikus programozási nyelvek is léteznek, pl. Simulink (MathWorks), VEE (Agilent).

Adatfolyam-programozás A LabVIEW megalkotói előtt a műszerekre jellemző adatbeolvasás–feldolgozás–kijelzés tipikus feladatsor lebegett. A nyelv szerkezetét és elemeit ennek az adatfolyamnak a minél egyszerűbb leírhatósága határozta meg. Ebből következően a hagyományos procedurális programozási elvek és konstrukciók nem, vagy csak áttételesen, alkalmazhatók. Szintén nem alkalmazhatók közvetlenül az objektumorientált programozás fogalmai.

„Könnyű” használat Az adatfolyam alapú grafikus programozás – és így a LabVIEW is – a feladatok egy jelentős részére rendkívül jól illeszkedik, ezért az ilyen jellegű programok megírása kifejezetten könnyű, szinte élvezetes. Természetesen, azoknak az alkalmazásoknak is jelentős a száma, amelyekre csak nehézkesen, körülményesen használható a LabVIEW. Nagyon könnyen lehet példát találni olyan alkalmazásokra, ahol a LabVIEW segítségével néhány kattintás egy olyan program megírása, amelyet C-ben órák alatt sem lehetne elkészíteni, de ellenpéldákat is ugyanilyen könnyen találhatunk: ami pl. C-ben magától értetődően egyszerű, az a LabVIEW-ban bosszantóan nehézkes lehet, tehát mindkettőnek megvan a saját optimális alkalmazási köre. A LabVIEW sem csodaszer, alapos ismeretéhez ugyanolyan fáradságos út vezet, mint bármilyen más programozási nyelv elsajátításához. Noha néhány órai tanulás, gyakorlás után már tudunk használható programokat írni, de hónapok, évek múlva is lesz tanulnivalónk.

2. ALAPISMERETEK

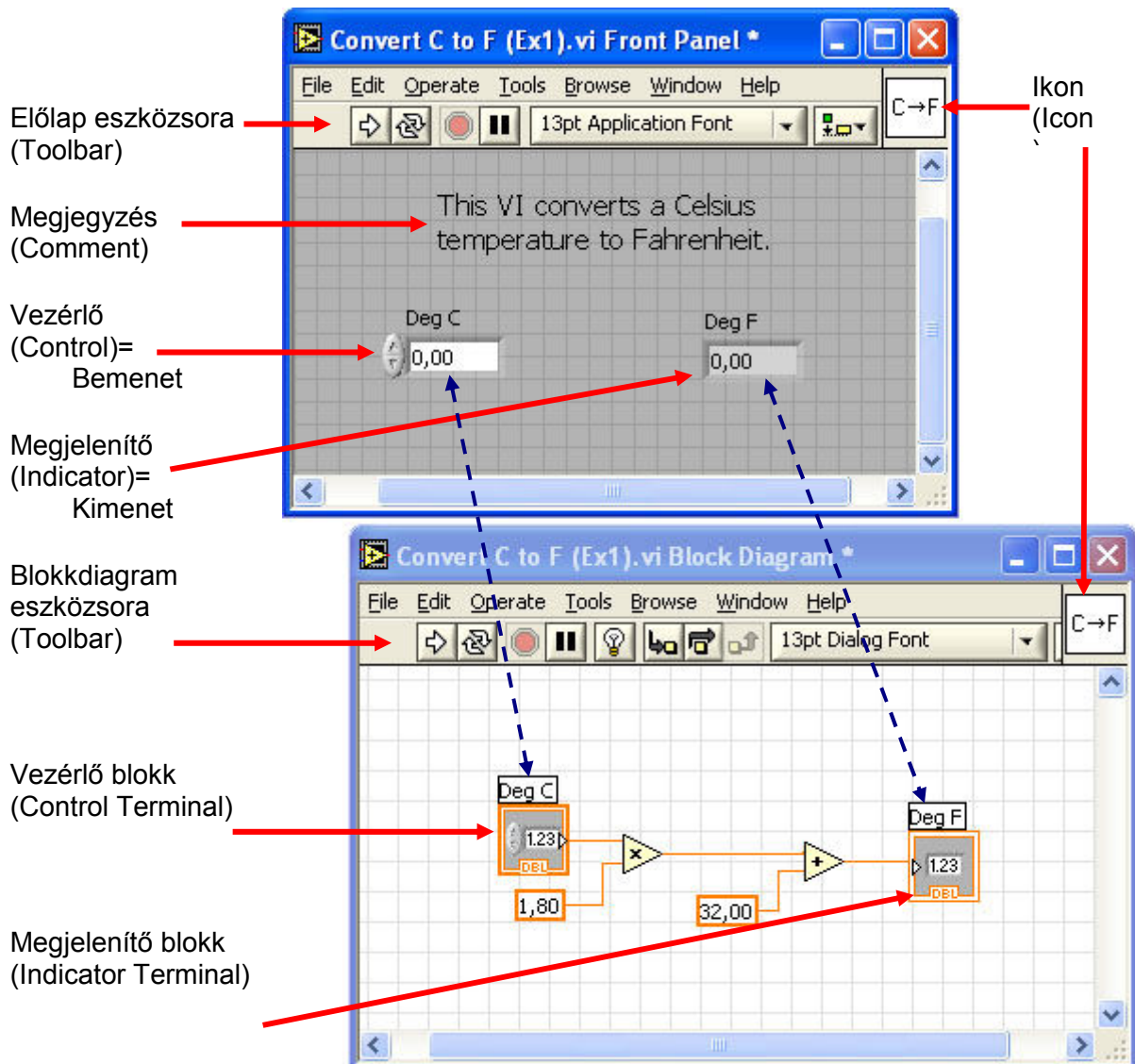
2.1 LabVIEW VI futtatása, alapfogalmak





A LabVIEW legfontosabb tulajdonságait egy egyszerű mintapéldán keresztül mutatjuk be.

Program = VI

A LabVIEW-ban a programokat „virtuális műszernek” (Virtual Instrument), vagyis VI-nek nevezik. Ha megnyitunk egy .vi kiterjesztésű VI fájlt a LabVIEW-val, akkor megjelenik a virtuális műszer előlapja (Front Panel). A **CTRL+E** billentyűkombináció leütésével (vagy a Window menüben Show Block Diagram), megjelenik a Block Diagram ablak is.

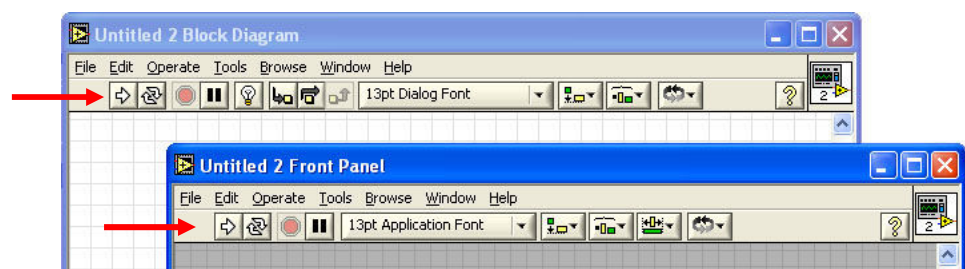
Az alábbiakban látható egy egyszerű VI Front Panel-je és Block Diagram-ja.












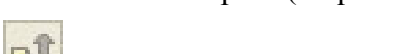




- Előlap Az előlapon a „virtuális műszer” (a program) ebben az esetben rendkívül egyszerű: egy Celsius fok értéket Fahrenheit fokra konvertál.
- Vezérlő, megjelenítő A minta VI-nak egy bemenete és egy kimenete van. A LabVIEW terminológiájában a bemenetet vezérlőnek (**Control**) nevezik, a kimenetet megjelenítőnek (**Indicator**). Ezen kívül vannak még konstansok (**Constant**), melyek értéke értelemszerűen nem változik a futás közben.
- Blokkdiagram A bemeneti és kimeneti egységek (vezérlő és megjelenítő blokkok; Control, Indicator) valamint a köztük kapcsolatot teremtő funkcionális egységek a blokkdiagramon össze vannak húzva. A húzolás jelenti a végrehajtás menetét, lényegében a programot. Minden bemeneti és kimeneti egységnek kétféle megjelenési formája van: a Front Panelen látható és a Block Diagrammon látható forma. (A fenti ábrán szaggatott kék nyilakkal jeleztük az összetartozó bemeneti és kimeneti egységeket.)
- Adatfolyam-programozás Minden egyes vezérlőtől (a bemenetektől) a huzalok vagy vonalak mentén egy-egy adatút alakul ki a megjelenítőig (kimenetekig). A blokkdiagram (program) végrehajtását az adatfolyam határozza meg. Általában balról jobbra haladva szokták felrajzolni az adatfolyamokat, de fontos tudni, hogy a végrehajtást nem a felrajzolás iránya szabja meg. **Egy adott funkcionális egység akkor hajtódik végre, ha az összes bemenetén rendelkezésre állnak az adatok.** A funkcionális egység a végrehajtás után az összes kimenetére kiadja a megfelelő adatot.
- Futtatás  Az eszközsorban a jobbra mutató vastag, üres nyílra kattintva indíthatjuk el a VI futtatását. (Ilyenkor a nyíl ikonja átalakul feketével kitöltött szaggatott nyíllá.).
- Folyamatos futtatás   Az első ikonra kattintva a VI folyamatosan futni fog, míg az előző, egyszerű futtatás esetén csak egyszer fut le. A futás ilyenkor a második, Stop gombbal állítható le.
- A futtatás animálása  Ha a blokkdiagramon az izzólámpa ikon ki van választva, az adatok áramlását mozgó korongocskák mutatják, és az értékek megjelennek a huzalokon. A programok működésének megértéséhez és hibakereséshez ez a funkció nagyon hasznos.

2.2 Státus- és eszközsor

A Block Diagramnak és a Front Panel-nek nagyon hasonló státus- és eszközsora van (toolbar).



A Block Diagram státus- és eszközsora a következő elemekből áll (balról jobbra, az eszközsor némiképp lehet LabView verzió függő):

-  Futtatás (Run) / fut / szintaktikai hiba
-  Folyamatos futtatás (Continuous Run)
-  Megszakítás (Abort Execution)
-  Szünet / Folytatás (Pause/Continue)
-  Végrehajtás szemléltetése (Execution Highlighting), debuggoláshoz
-  Huzalozás adatainak folyamatos elmentése a végrehajtás során, debuggolásnál használható. (Retain data values)
-  Belépés subVI-ba (Step Into)
-  SubVI átlépése (Step Over)
-  Kilépés subVI-ból (Step Out)
-  Szövegformázás (Text Settings)
-  Objektumok vonalhoz igazítása (Align Objects)
-  Objektumok közötti távolság beállítása (Distribute Objects)
-  Átrendezés (Reorder)
-  A kontextusfüggő súgó megjelenítése/elrejtése (Show/Hide Context Help Window)

Csak a Front Panel státus- és eszközsorában található:

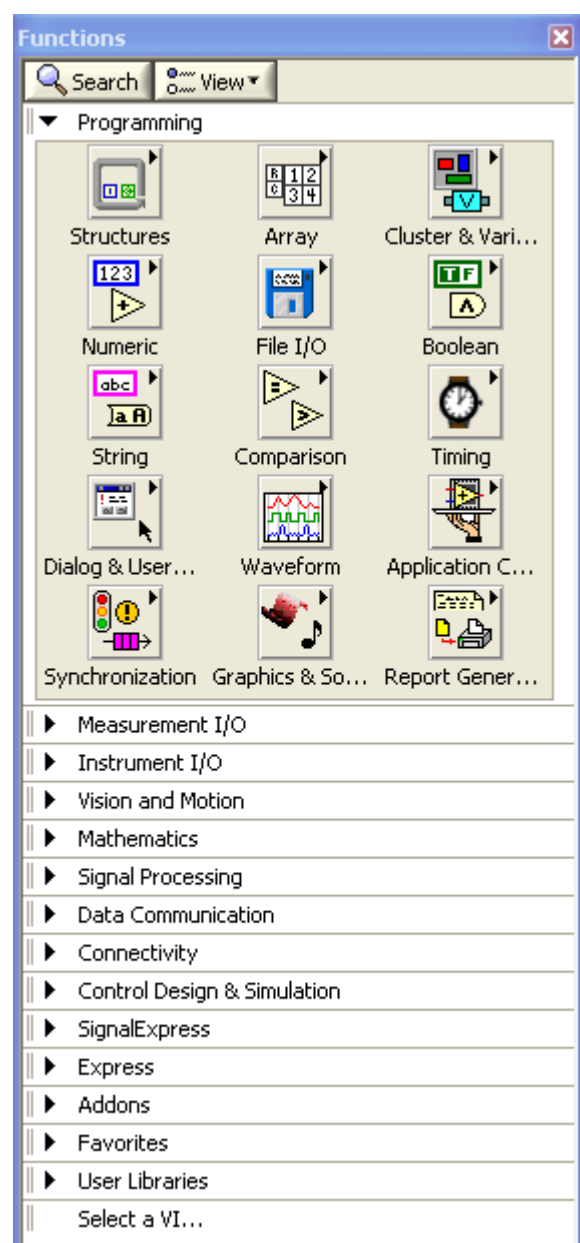
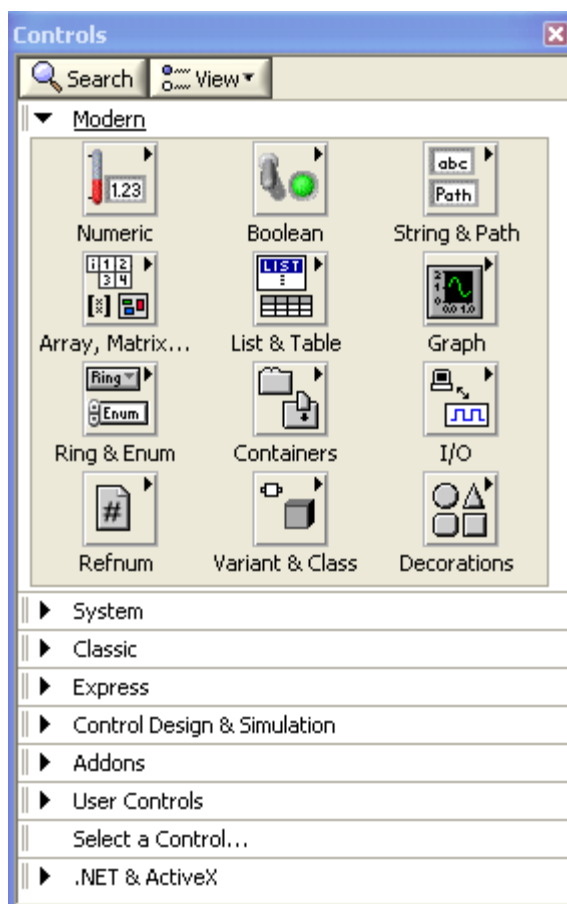
-  Objektum átméretezése (Resize Objects)

2.3 Egy egyszerű példaprogram létrehozása

A legalapvetőbb fogalmak megismerése után egy egyszerű példaprogram írásán keresztül mutatjuk be a LabVIEW további tulajdonságait. Létrehozuk a Celsius–Fahrenheit átalakítás párját, a Fahrenheit– Celsius konvertert. Az algoritmus nagyon egyszerű:

A Fahrenheitben adott értékből kivonunk 32-t, majd elosztjuk 1.8-al!

- Új VI megnyitása A LabVIEW indítása után válasszuk ki a New csoportból a Blank VI (üres VI) sort.
- Megjelenik egy üres Front Panel és a szintén üres Block Diagram (ha nem jelenne meg, hívjuk elő a CTRL+E leütésével).
- Paletták A VI-k írásához, futtatásához, hibakereséséhez, javításához szükséges eszközöket ún. palettákba szervezik (a paletták kinézete verzió és konfigurációfüggő, de az alap blokkok megegyeznek).
- Vezérlők (Controls) Az Front Panel-hez tartozó segédeszközöket a **Controls** paletta tartalmazza. Ezt úgy hívhatjuk elő, ha a Front Panel-re a jobb egérgombbal kattintunk.
- Funkciók (Functions) A Block Diagram eszközei a **Functions** palettában vannak. Ezt úgy hívhatjuk elő, ha a Block Diagramra kattintunk a jobb egérgombbal.



A VI-nk előlapja

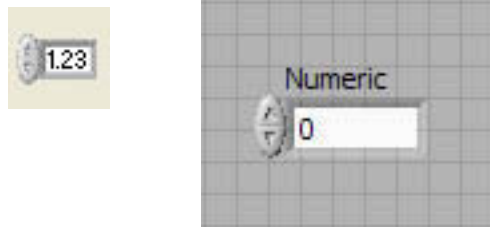
Először állítsuk elő a speciális virtuális műszerünk Front Panel-jét! Egy vezérlőnk (bemenetünk: Fahrenheit fok) és egy megjelenítőnk (kimenetünk: Celsius fok) van.

Változófogalom

Hagyományos programozás esetén két változóról beszélünk, az egyik tartalmazná a Fahrenheit, a másik pedig a Celsius értéket. Eddig szándékosan kerültük a változó elnevezést, mivel a LabVIEW-ban alapvetően nem változókkal, hanem adatforrásokkal (vezérlők, Controls) és megjelenítőkkal (Indicators), valamint adatfolyamokkal dolgozunk. (A LabVIEW-ban is vannak lokális és globális változók, de ezeket csak akkor használjuk, ha az eredeti adatfolyam-konceptióval nem tudjuk megoldani a feladatot.)

Control létrehozása

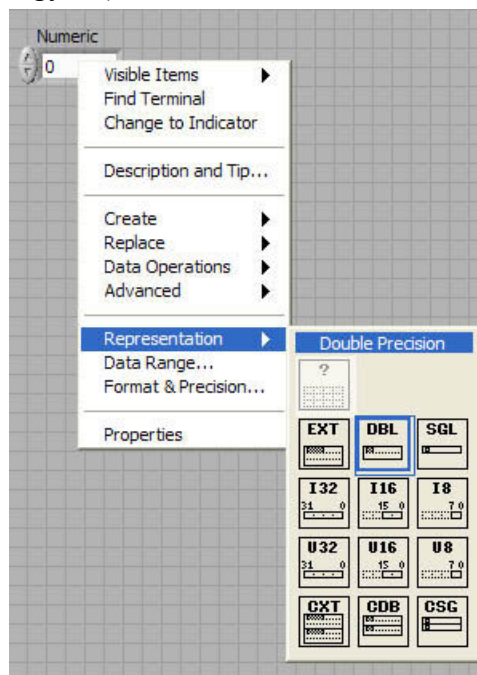
Az Front Panel Controls palettáján az egérrel kattintsunk a *NumCtrls* alpalettára (bal felső ikon), majd válasszuk ki a Num Ctrl ikont!










Kattintsunk a blokkdiagram ablakra, és nézzük meg, hogy az előlapra elhelyezett Numeric blokkunkhoz milyen szimbólum tartozik (Numeric, DBL)!

Adattípus

Váltunk vissza a Front Panel-re, kattintsunk a jobb egérgombbal a Numeric ikonra, és válasszuk ki a legördülő menüből a Representation pontot! Itt kékkel bekeretezve megjelenik a vezérlőnk (lényegében változónk) típusa (defaultként DBL, vagyis Double Precision). Ha szükséges, itt tudjuk megváltoztatni a változó típusát. (Most ne tegyük!)



- Indicator létrehozása  A Numeric Control mintájára helyezzük el az előlapon az indikátorunkat (Numeric Indic...)! Ez egy Numeric 2 feliratú lesz. Ezzel megvan az adatfolyamunk eleje és vége, most a közbülső részeket kell létrehoznunk.
- Nevek megváltoztatása A Numeric és a Numeric 2 semmitmondó neveket a blokkdiagramon változtassuk meg beszédeesebbekre: az előbbit írjuk át Fahrenheit-re, az utóbbit pedig Celsius-ra!
- Kivonás  Az algoritmus szerint a kezdeti értékből először ki kell vonni 32-t. Ehhez kell egy kivonó egység és egy 32 értékű konstans.
- Konstansok  Ugyanerről a Numeric palettáról tegyünk egy konstans értéket reprezentáló ikont a blokkdiagramra (Numeric Cons..., bal alsó sarok), és a default nulla értéket írjuk át 32-re!
- Ugyanezt egyszerűbben úgy tudjuk megtenni, hogyha a kivonó ikon adott bemenetére mutatunk az egérrel, majd a jobb gomb segítségével előhívott ablakból kiválasztjuk a Create/Constans menüpontot. Ez azért fontos, mert a későbbiekben így tudunk bonyolult konstansokat, de akár indikátorokat vagy controllokat is létrehozni egy mozdulattal. Ennek az az előnye, hogy ilyenkor mindig a megfelelő konstans jön létre, nem nekünk kell kiválasztani és létrehozni a típust (sok VI elég bonyolult struktúrákat vár bemenetként).*
- Osztás  A Numeric alpalettáról tegyünk egy osztás (Divide) műveleti egységet a blokkdiagram megfelelő helyére!
- Törött nyíl  Mielőtt nekilátnánk a huzalozásnak, figyeljük meg a blokkdiagram ablak státus/eszközsorában a bal oldali ikont (Run nyíl)! Egy kettétört, szürke nyilat fogunk látni. Ez jelzi, hogy a VI (program) szintaktikailag nem alkalmas futásra. Ha a program futásra kész, az ikon automatikusan átvált ép, világos nyílra.
- Futtatás  A rendszer összehuzalozása után kattintsunk a Run nyílra! Egy rövid villanás, és megjelenik a Celsius ablakban a konvertált érték (A program egyetlen egyszer fut le).
- Mentés  Mentsük el a programunkat Convert_F_to_C.vi néven!

2.4 Vezérlési szerkezetek (Structures) és adattípusok

2.4.1 Ciklusok

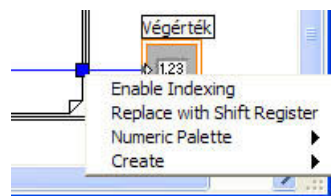
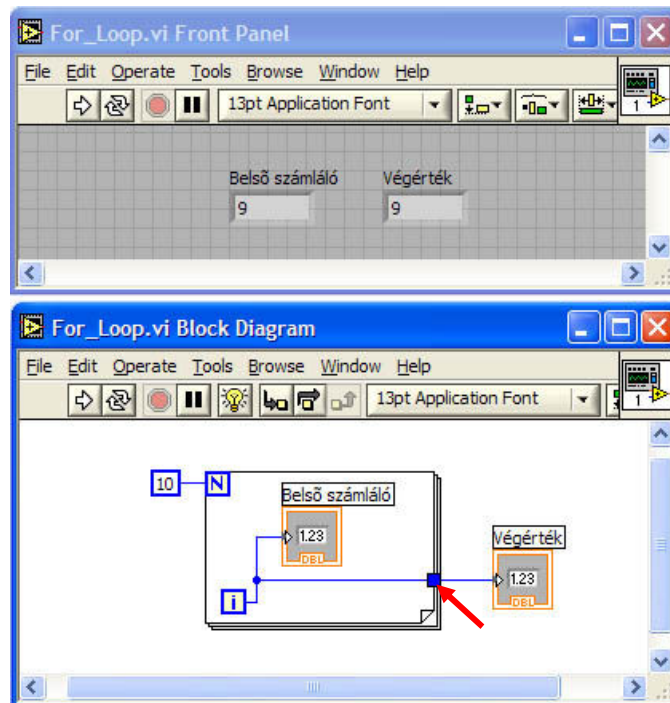
A Functions paletta Structures palettáján található a vezérlési szerkezetek. Ezek közül először a ciklusokkal foglalkozunk.

FOR ciklus

I Használatához ki kell választani a **For loop** ikont, majd körbe kell venni a ciklusba zárandó kódrészt. A For ciklusnak van ciklusszámlálója (i, Iteration Terminal), a ciklusok számát pedig az N bemeneti változó (Count Terminal) értéke határozza meg. A ciklus 0-tól N-1-ig fut.

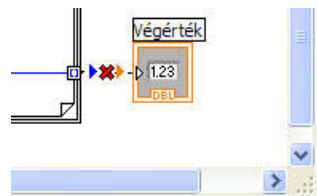
Tekintsük az alábbi mintaprogramot (FOR_Loop.vi):

A piros nyíllal jelölt kék négyzetnek az a feladata, hogy „kivezesse” az i ciklusváltozó végértékét a ciklusból. Ha a jobb egérgombbal rákattintunk a kék négyzetre, előugrik egy legördülő menü: Enable Indexing stb. menüpontokkal. Az Enable Indexing arra is utal, hogy most Disable állapotban van a kék négyzet (le van tiltva az automatikus indexelés, lásd alább), vagyis ebből az állapotból Enable-t tudjuk tenni. Auto index letiltva

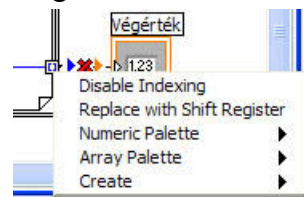


Auto index engedélyezve

Válasszuk ki az Enable Indexing sort! A következőt tapasztaljuk:



Megváltozik a négyzet kitöltése, és a Végérték-hez menő adatfolyam megszakad. Kattintsunk ismét a négyzetre!



Az Enable/Disable Indexing most már elárulja, hogy mi történik. A ciklus kétféleképpen vezethetjük ki az i -t: automatikus indexelés letiltásával vagy automatikus indexeléssel. Az előbbi esetben a kimenő adat egyetlen érték, az i végértéke lesz, míg az utóbbiban egy tömb, a ciklusváltozó ciklusonkénti értékével. A fenti példában a kapcsolat azért szakadt meg, mert egy tömböt nem lehet közvetlenül összekötni egy skalárral.

Ez az automatikus indexelés (vagy annak letiltása) általánosan is igaz, nemcsak a FOR ciklusra, és nemcsak a ciklusváltozóra működik.

Adattípusok

A FOR_Loop.vi az adattípusok szemléltetésére is alkalmas. A LabVIEW-ban jelentősége van a színeknek. A kék szín egész (integer) típust jelent (lásd i és N , valamint a 10 konstans), a barna dupla pontosságú lebegőpontost (double). Itt automatikus típuskonverzióra is látunk példát: az i -t probléma nélkül összekapcsolhatjuk a DBL változóval, a LabVIEW automatikusan elvégzi a szükséges konverziót. Természetesen, az a jobb megoldás, ha azonos típusokat kötünk össze, és mi végezzük el a típusdefiníciókat (Representation: DBL, I32, I16 stb.).

Tömbök (arrays)

A normál programozási nyelvekhez hasonlóan a LabVIEW-ban is lehetőségünk van tömböket létrehozni és kezelni.

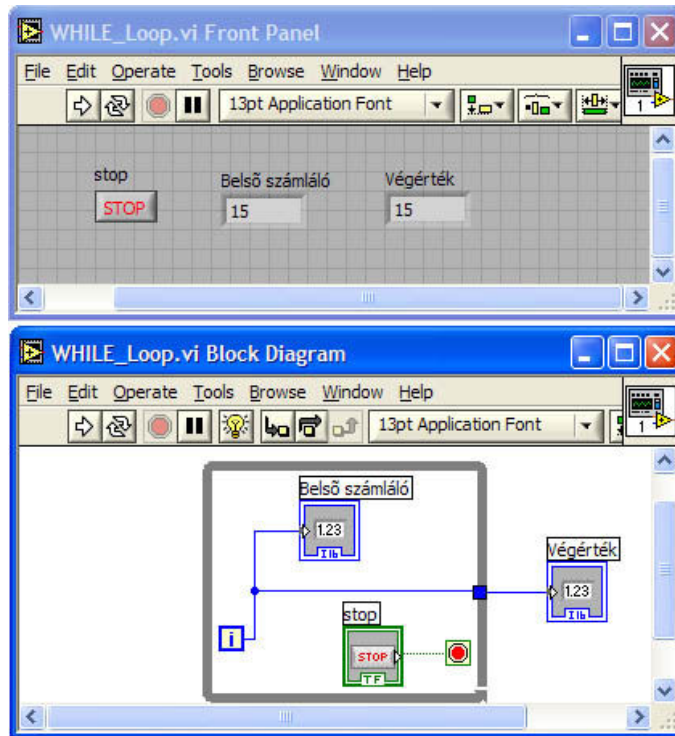
Fürtök (Clusters)

A fürt vagy köteg (cluster) adatokat összefogó adatstruktúra. Az adatok különböző típusúak is lehetnek. Hasonlít a C nyelv struct-jához. A gyakorlatban legtöbbször bonyolultabb VI-k konfiguráló konstansaiként fogjuk használni.

WHILE ciklus

Van ciklusszámlálója (i , Iteration Terminal). Legalább egyszer lefut. A ciklusok számát a ciklusfeltétel (Conditional Terminal) szabja meg. Az alábbi WHILE_Loop.vi mintapéldában a ciklus mindaddig végrehajtódik, amíg rá nem kattintunk a STOP gombra az előlapon. (Természetesen, minden program leállítható a státuszsorban levő Stop gombbal.)

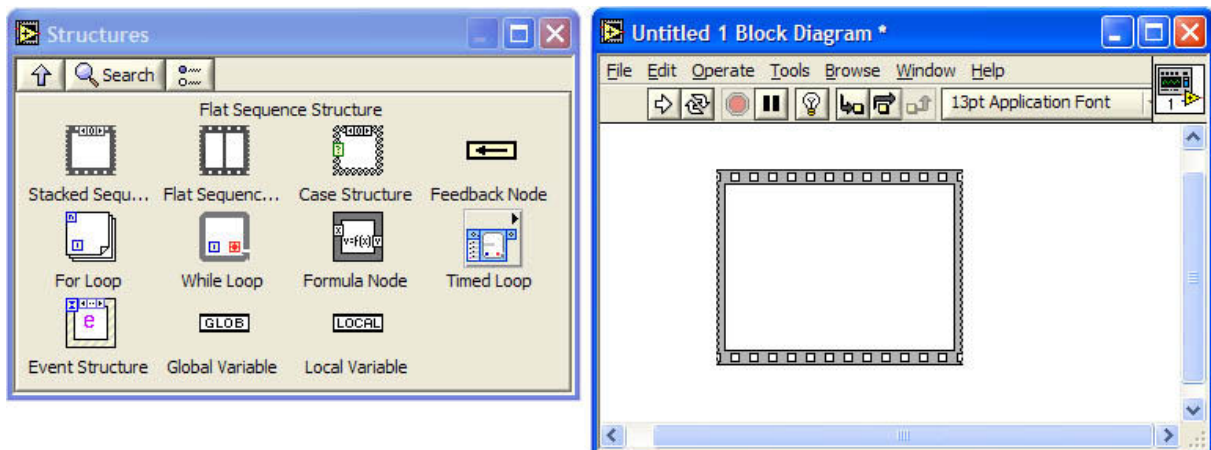
A while ciklus leállítására sokféle lehetőség van. A jobb egérgombbal a piros stop jelre kattintva a legördülő menüből kiválaszthatjuk a számunkra legmegfelelőbbet.



2.4.2 Sorrendi struktúrák

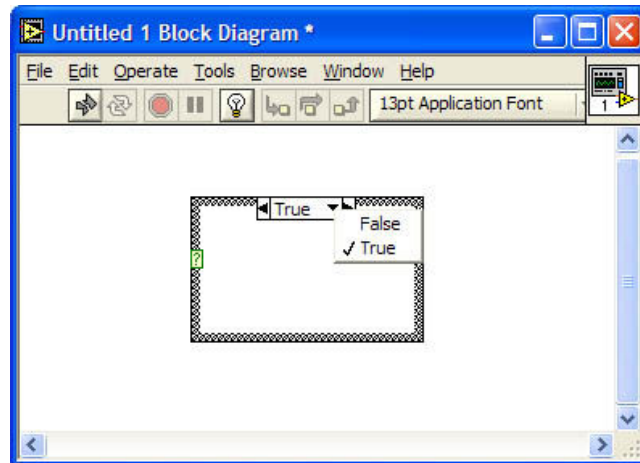
A sorrendi struktúrák alapesete a **Flat Sequence**. A Functions paletta Structures alpalettáján található, kinézete egy filmszalaghoz hasonló. Ikonjának kiválasztása után a blokkdiagrammon körbe kell venni a szekvenciálisan végrehajtandó programrészt, vagy a keret kijelölése után lehet belevonszolni a blokkokat!

A blokkokat szekvenciálisan, mint a filmszalag képkockáit, egymás után hajtja végre.



2.4.3 Választás (Case) struktúra

A szokásos switch-case struktúra stack-szerű megjelenítése. A keretek egymás mögé illeszkednek, akár a kártyalapok. Egyszerre csak egy „lap” látszik. Helye: **Functions / Structures / Case Structure**.

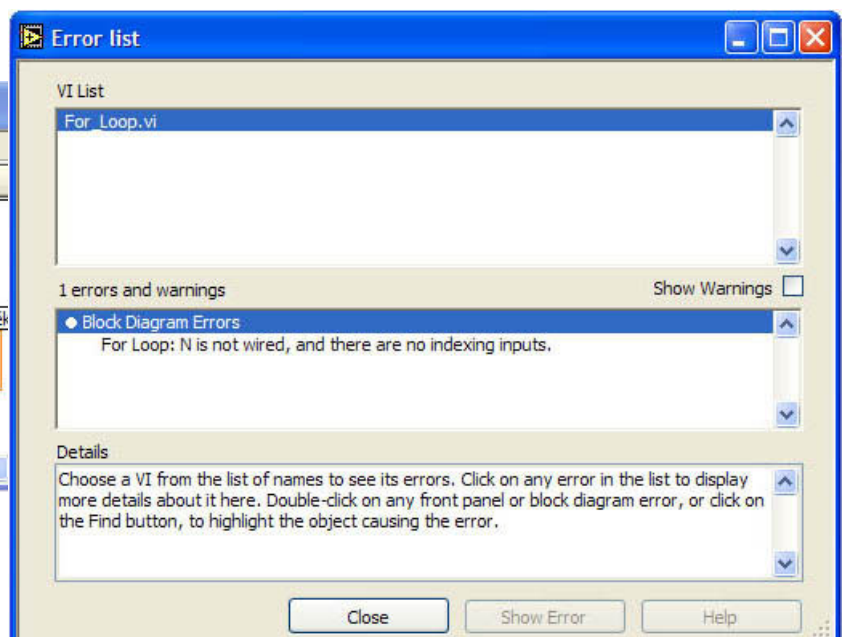
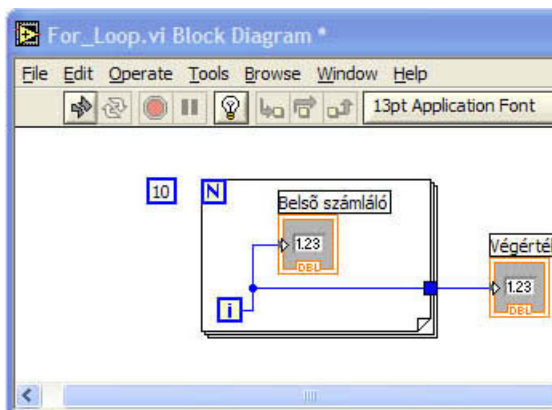


A keretre történő jobb-kattintással előjön a struktúrához tartozó legördülő menü. A legördülő menüben adhatunk új case eseteket a struktúrához, illetve távolíthatunk el nem használt eseteket. Fontos megjegyezni, hogy a Case struktúra által mutatott esetek (Case-ek) függnek a kiválasztó bemenet típusától. Alap esetben a Case struktúra logikai kiválasztó bemenettel rendelkezik, így két eset van definiálva a True és a False. Amennyiben a kiválasztó bemenet típusát megváltoztatjuk (pl. integer vagy string típusra), akkor az esetek is megváltozhatnak.

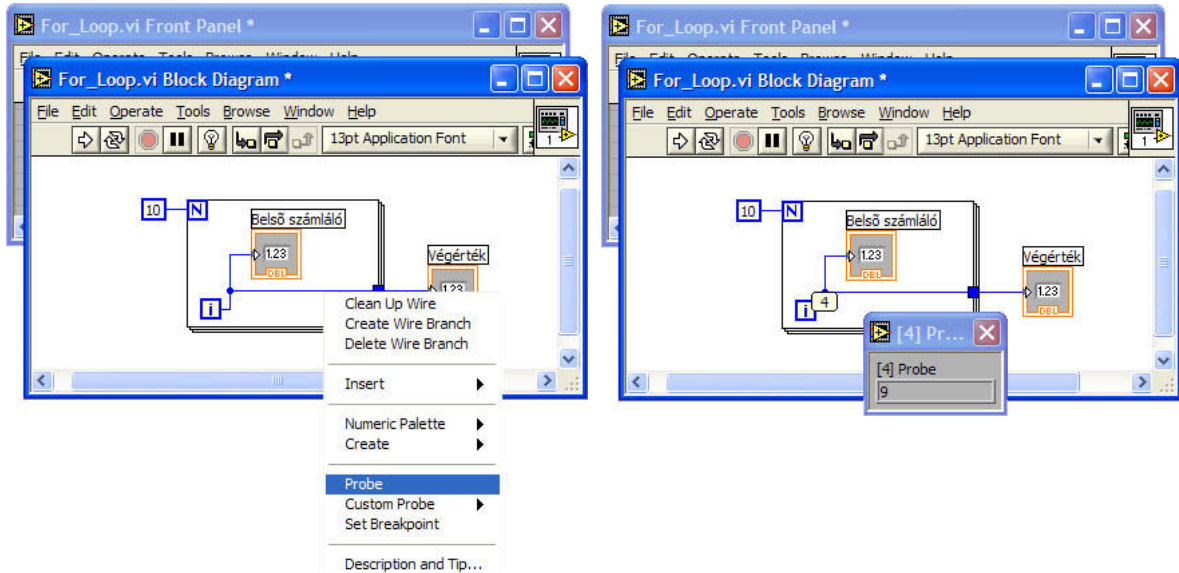
2.5 Egyéb fontos LabVIEW elemek


Hibakeresés

Kattintsunk a törött nyíl gombra! A megjelenő ablak kiírja a hibá(ka)t.



Mintavevő (Probe) Jobbkattintás a huzalra: megjelenik egy legördülő menü, kiválasztjuk a Probe-ot, amely futás közben kijelzi a huzalon „átáramló” adatot. (Kiválaszthatjuk a Mintavevő (Probe) eszközt is az Eszközök (Tools) palettából, majd kattintsunk a huzalra!)



Kontextusfüggő sűgő  Nagyon sokat segít a kontextusfüggő sűgő (**Context Help**). Ha rávisszük a kurzort valamelyik elemre, a hozzá tartozó rövid magyarázat azonnal megjelenik.

subVI A LabVIEW-ban a VI-k nem csak főprogramként használhatóak, hanem lehetőségünk van egy VI-ban egy másik VI-t, mint egyetlen blokkot futtatni.. Tehát minden VI-ból készíthetünk egy olyan egységet, amelyet egy másik VI-ba szubrutinszerűen, egyetlen blokkként beilleszthetünk. Ezeket nevezzük subVI-nak. A subVI-k esetében a bemenetek és kimenetek helyettesítik a Front panel kezelőfelületét. A subVI-k létrehozását egyszerűen kipróbálhatjuk úgy, hogy a blokk diagramban a subVI-ba zárnı szándékozott területet kijelöljünk, majd az Edit / Create Sub VI menüponttal létrehozunk belőle egy subVI-t. A subVI-kból könyvtárakat állíthatunk össze. A subVI-k használata egyszerűsíti és átláthatóbbá teszi a programjainkat, gyakorlatilag a LabVIEW könyvtári elemek többsége is mint subVI jelenik meg a számunkra.

3. ISMERKEDÉS A VIRTUÁLIS MŰSZEREKEL

3.1 Az National Instruments műszerkínálatának rövid áttekintése

Adatgyűjtő kártyák A legegyszerűbb módja, hogy egy PC-ből műszert készítsünk az, hogy valamilyen adatgyűjtő interfésszel egészítjük ki. Ez az adatgyűjtő interfész lehet a legegyszerűbb, minden gépen megtalálható hangkártya, vagy annál lényegesen több funkcionalitást nyújtó PCI, PCI Express buszra csatlakozó adatgyűjtő modul. A National Instruments igen széles eszközválasztékkal rendelkezik ilyen modulokból. Az általános célú adatgyűjtő eszközök 4-80 analóg bemenettel, 0-4 analóg kimenettel, 16 bites felbontással és 250 kHz-től 1,25MHz-es mintavételi frekvenciával rendelkeznek, de kaphatóak speciális modulok is, amelyek vagy a mintavételi frekvenciában, vagy más paraméterben lényegesen felülmúlják ezeket az általános eszközöket.



Az analóg adatgyűjtő kártyákon kívül más, speciális funkciókat ellátó modulok is találhatóak a National Instruments termékpalettájában (motor meghajtók, digitális I/O-k stb).

Műszerek Illesztése Amennyiben valamilyen oknál fogva a PC-s adatgyűjtő kártyák nem lennének megfelelőek, vagy elérhetők akkor általában valamilyen külső műszert kell a PC-hez csatolni. A hozzácsolás módját mindig az illesztendő eszköz határozza meg azzal, hogy milyen kommunikációs interfészt nyújt. Ez a legtöbb esetben Rs232, Rs485, CAN vagy Ethernet-et jelent, de sok műszer egy hagyományos műszerillesztő felületet a GPIB támogatja.

A GPIB busz Mivel a GPIB-ről az eddigi tanulmányok során nem igen esett szó, itt röviden bemutatjuk, bár a laborban nem fogjuk használni. A GPIB kommunikációs felületet az IEEE 488-as szabvány specifikálja és több, mint 30 éve használják az iparban. Sok műszer még manapság is támogatja ezt a felületet, például az alaplabor mérések során használt Agilent műszerek mindegyike rendelkezik GPIB interfésszel (Ez már azért sem meglepő, mert a szabvány alapját 1965-ben a HP fejlesztette ki HP-IB néven). A GPIB busz sebessége 1 MiB/sec ezt 8 adatvonalra és számtalan kiegészítő handshake vonal segítségével tudja elérni. A National Instruments egy kiegészítést készített ehhez a szabványhoz HS-488 néven, ami 8 MiB/s sebességre képes.

PXI

Sok ipari környezetben nem megengedett PC-k kihelyezése, ezért a szükség lehet egy kompaktabb ipari kivitelű rendszerre. A National Instruments ezekre a szituációkra fejlesztette ki a PXI rendszerét. A PXI (PCI eXtension for Instrumentation) gyakorlatilag egy az ipari követelményeknek megfelelő PC bázisú rendszer. A PCI busz ipari kiegészítése pedig a szinkronizációs problémák megoldását szolgálja. A kiegészítő jelek között szerepel közös órajel és trigger jelek. Szoftver tekintetében nincs lényegi eltérés a PXI rendszerek és a normál PC-s rendszerek között, mivel mindkettő PC-s alapon működik.



A PXI rendszerek előnye a szinkronizációban és az ipari kivitelben keresendő. Mint az ábrán is látható ugyanúgy modulárisan, adatgyűjtő kártyák beillesztésével lehet a PXI eszközökből mérőrendszert megvalósítani, mint az asztali PC-s esetben, csak itt a kivitel kompaktabb és jobban megfelel az ipari követelményeknek. A PXI jelenleg talán a legszélesebb körben használt ipari adatgyűjtő, vezérlő platform.

CompactDAQ

A National Instruments palettáján az USB egyre szélesebb körű elterjedésének köszönhetően a 2000-es évek közepén jelent meg a CompactDAQ sorozat, amely a PXI-os moduláris műszerépítési koncepciónak egy olcsó PC központú verziója.



Az USB-re csatlakozó alapsínhez itt is többfajta mérő beavatkozó modul illeszthető, így viszonylag olcsón (a PXI-hoz képest) létrehozva egy moduláris mérő, beavatkozó rendszert.

CompactRIO

Az USBs sorozat sikerét követően készült el az egy fokkal komplexebb feladatokra képes, a PXI-al majdnem megegyező

funkcionalitást nyújtani tudó CompactRIO (RIO: Reconfigurable I/O) sorozat, ahol a mérő modulokat befoglaló sín Real-Time PC-t és FPGA támogatást is tartalmazhat.



A CompactRIO sorozat tulajdonságait és alkalmazását későbbiekben részletesen bemutatjuk.

LabVIEW támogatás A fent bemutatott hardver megoldások csak úgy váltak jól használhatóvá, hogy a National Instruments igen magas szintű támogatást nyújt hozzájuk a LabVIEW illetve LabWindows/CVI környezetben. Ez a támogatás gyakorlatilag azt jelenti, hogy a cég által forgalmazott hardverek VI szinten elérhetőek, konfigurálhatóak és nagyon egyszerűen alkalmazhatóak, például *Data Acquisition Assistant (DAQ Assistant)* szoftvercsomag segítségével.

Alkalmazási terület A bemutatott rendszerek alkalmazási területe elsősorban ott keresendő, ahol a mérési funkciók változatosak, és egy standard műszer nem tudja kielégíteni azokat. Illetve még inkább azokon a területeken, ahol valamilyen együttműködés szükséges a különböző műszerek, érzékelők beavatkozók között. Tipikus alkalmazási terület valamilyen ipari automatizálási folyamat, például gyártósorok automatizálása, ahol minden lépésnél tesztelő méréseket kell végezni, és ezeket a méréseket folyamatosan dokumentálni kell az ISO-9000-es szabvány követelményei miatt. Ezekre a problématerületekre nem igen létezik más ennyire komplett eszközkészletű megoldás, így nagy valószínűséggel állíthatjuk, hogyha felkeresünk egy gyárat, akkor ott találkozni fogunk ezekkel a technológiákkal és eszközökkel.

3.2 Függvénygenerátor készítése a PC hangkátya segítségével

(1. mérés: *LabVIEW alapismeretek* záró feladata).

Cél A labor során alapvetően egy PC-t fogunk Virtuális műszerként használni. A PC hangkátyája lesz a műszer kártyánk a feladat specifikációja a következő:

Készítsen a hangkátya segítségével függvénygenerátort a PC-ből.

A függvénygenerátor fő funkciói a következők:

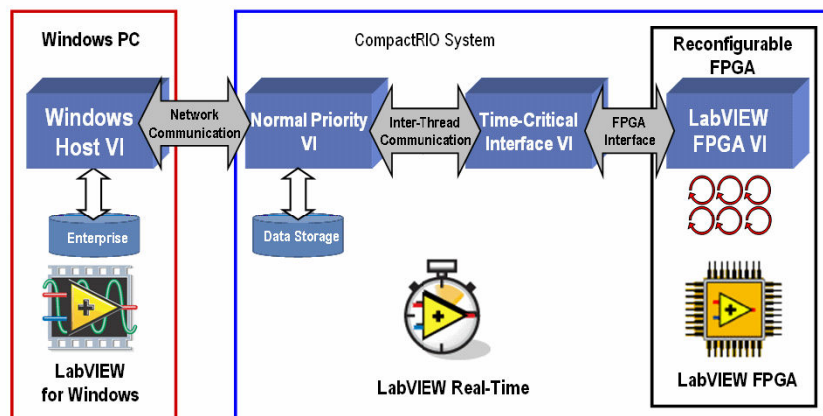
- a. Sinusjel, négyszögjel, háromszögjel közül kiválasztható hullámforma. A négyszögjelnél a kitöltési tényező megadható.
- b. Állítható frekvencia és amplitúdó (az amplitúdó részhez egy segéd konverziós konstans bevezetése szükséges, hogy tényleges fizikai paramétereket tudjuk állítani az előlapon).
- c. Kiment engedélyező nyomógomb.
- d. A kiadott hullámformát jelenítse meg a képernyőn

4. ISMERKEDÉS A COMPACT-RIO RENDSZERREL

4.1 A CompactRIO fejlesztőkörnyezet

CompactRIO

A CompactRIO segítségével a különböző mérési beavatkozási funkciókat a megfelelő modulok alapkeretbe való behelyezésével lehet végrehajtani. A rendszer érdekessége nem is a funkciójában, hanem konfigurációjának, programozásának újszerű módjában rejlik. Ugyanis az eszközhöz kötődően három külön helyre lehet programot fejleszteni, de mindegyik programot ugyanabban az integrált LabVIEW alapú környezetben (még ha az eszközkészlet picit változik is). Az első hardver közeli rész ahova a programot írhatjuk a mérőmodulokat összefogó alapkeretben elhelyezkedő FPGA. Erre az eszközre a hardver közeli feldolgozásokat lehet a LabVIEW FPGA modul segítségével leprogramozni (például digitális szűrés, időbélyegezés, pulzusszámolás stb.). A következő szint ahova lehetőségünk van saját programot helyezni az a szintén a CompactRIO hardverben elhelyezkedő Real-Time ipari PC. Erre a modulra a LabVIEW Real-Time modul segítségével tudunk konfigurációt készíteni, tipikusan az ide kerülő konfigurációk tartalmazzák a magasabb szintű vezérlést, például egy PID szabályozást, adattárolás stb.. Az utolsó absztrakciós szint, amit lehetőségünk van kezelni az egy távoli host gépen futó felügyeleti alkalmazás. Erre a helyre a normál LabVIEW-ban készült programok formájában helyezhetjük el az alkalmazásainkat, tipikusan ez a grafikus felhasználói interfész helye.



A CompactRIO egység programozásának hierarchiája

Bár ez a hármas tagoltság elsőre bonyolultnak tűnhet, a gyakorlatban igen egyszerűen használható egyetlen integrált fejlesztőrendszerből.

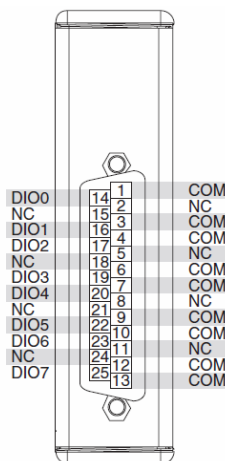
cRIO-9074

A mérések során a cRIO-9074-es CopmactRIO keretet fogjuk használni. Ez 8 mérő kártya fogadására képes, a hátlapba egy 2M ekvivalens kapu komplexitású Xilinx FPGA foglal helyet. A helyi real-time feldolgozást pedig egy 128MiB DRAM-al és 256MiB háttértárral rendelkező 400MHz-es vezérlő biztosítja. A keretbe beépítve két 10/100BASE-Tx Ethernet csatorna található, amelyek közül az 1. sorszámún kapcsolódhatunk a modulhoz fejlesztés közben. Valamint rendelkezik egy Rs232 csatlakozási ponttal is.



NI-9401

Az első mérési gyakorlatainkhoz a z NI-9401 8 csatornás TTL szintű, nagysebességű 100ns-es digitális I/O modult fogjuk használni.

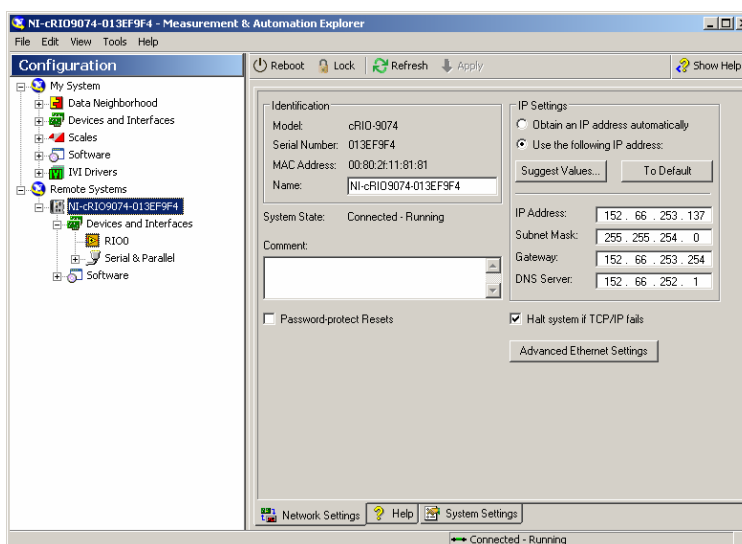


4.2 Fejlesztés a CompactRIO egységre

Kapcsolat felvétel



Első lépésként a CompactRIO modul használatához ellenőrizni kell a TCP/IP hálózati beállításokat, amelyek segítségével kapcsolatba lehet lépni az eszközzel. Erre a NI Measurement&Automation Explorer programját használjuk. Ez a programcsomag arra szolgál, hogy áttekinthessük, és szükség esetén módosítsuk a PC-khez hozzárendelt mérési adatgyűjtők és beavatkozók beállításait. Itt kerülnek felsorolásra az egyes PCI alapú mérő kártyák, GPIB buszon csatlakozó külső műszerek, valamint a távoli eszközök (*Remote Systems*) kategóriában az egyes CompactRIO eszközök. Ha szerencsénk van, akkor itt már a méréshez tartozó cRIO-9074-es modul szerepel a listában.



Ha nem akkor a *Remote Systems/Create New* menüpont segítségével adhatjuk hozzá a listához (automatikusan kerestessük meg, vagy adjuk meg a modul IP címét direktben). Amennyiben nem tudjuk a modul IP címét akkor lehetőségünk van azt kideríteni úgy, hogy a soros porton keresztül hozzácsatlakoztatjuk az eszközt a PC-hez, bekapcsoljuk az cRIO-9074 előlapján lévő *Console Out* switch-et és a sorosportra 9600baud sebességgel elküldött üzenetből megállapítjuk a két Ethernet csatlakozáshoz tartozó TCP/IP beállítást.

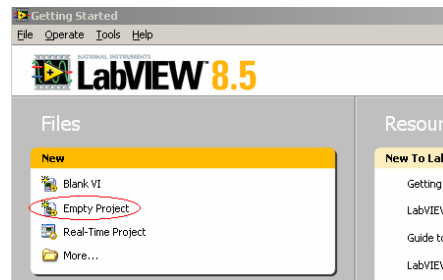
```
MAX system identification name: NI-cRIO9074-013EF9F4
Initializing network...
Device 1 - MAC address: 00:80:2F:11:81:81 - 152.66.253.137 (primary)
Device 2 - MAC address: 00:80:2F:11:81:82 - 192.168.9.2 (secondary)
```

NI Measurement&Automation Explorer segítségével átírhatjuk az eszköz hálózati beállításait. *(Azon ne lepődjünk meg, hogy itt nem jelennek meg a sínbe illesztett modulok és azok tulajdonságai, ezeket csak a LabVIEW projektekből lehet majd látni.)*

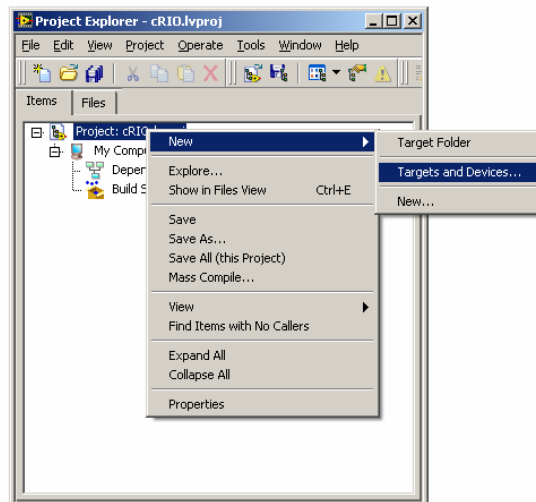
Az első project



Mint arról a bevezetőben már szóltunk a CompactRIO modulnál három különféle helyre is lehet programot írni. Az első mintaprojektünkben egy olyan környezetet fogunk varázsló nélkül létrehozni, ahol mindhárom szintre tudunk programot készíteni. Itt érdemes megjegyezni, hogy lehet akár a beépített varázslók segítségével is projecteket létrehozni, de így 0-ról indulva talán kicsit jobban áttekinthető a folyamat. Indítsuk el a *National Instruments LabVIEW 8.5.1*-et és válasszuk a *New/Empty Project* opciót.



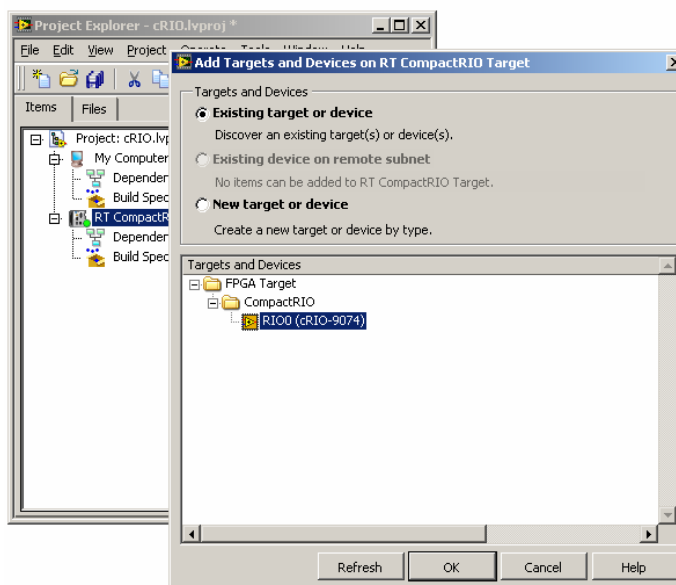
Ennek a hatására megjelenik a projekt ablak, ahol felsorolásra kerülnek az általunk használt mérőeszközök és programunk. Mivel a jelenleg egy üres mérőrendszer nélküli környezetet kaptunk, ezért a projecthez hozzá kell adnunk a CompactRIO eszközünket. Ezt a Projekten jobb egérgombot nyomva a *New/Target and Devices* menüvel tehetjük meg.



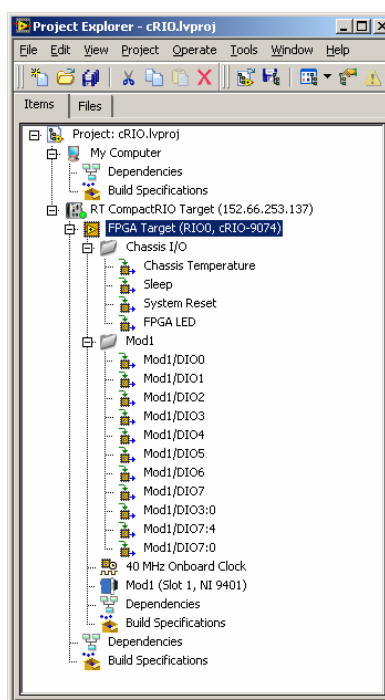
Lehetőségünk van már meglévő – ezeket automatikusan detektálja a LabVIEW – vagy egy általunk definiált új eszközt hozzáadni a rendszerhez (Nem mindig detektálja a rendszer még a Measurement&Automation Explorer-el megadott eszközöket sem. Ilyenkor új eszközként kell hozzáadnunk a modult, ami abból áll, hogy ki kell választanunk az eszköz típusát, ami cRIO-9074 és megadni az IP címét). Adjuk hozzá a mérésnél használt cRIO kártyát a rendszerhez.

Amennyiben a rendszer automatikusan nem detektálta a cRIO-9074-es eszközt, akkor szükség lehet arra is, hogy külön hozzáadjuk az FPGA target-et is projecthez (Az FPGA huzalozza össze a RealTime

PC-vel az egyes mérőkártyákat, ezért ez szükséges ahhoz, hogy lássuk az egyes mérő moduljainkat.) Ez hasonlóan történik, mint a projekt indításnál bemutatásra került, azzal a különbséggel, hogy itt a CompactRIO modulhoz adjuk hozzá az új *target*-et. Ha jól adtuk meg a cRIO modul IP címét és végrehajtottuk a *Connect* parancsot, akkor itt már automatikusan fel kell ismernie a rendszernek az FPGA targetet.



Az *FPGA target* felismerése után a rendszer felderíti a modulhoz csatlakoztatott mérőkártyákat. Megjelennek a project ablakban a magához az cRIO9074-es modulhoz tartozó mérő és beavatkozó csatornák (*chassis*), valamint az egyes mérőmodulok csatornái. Az ábrán a *Mod1* néven az NI9401 digitális I/O kártya csatornái láthatóak.

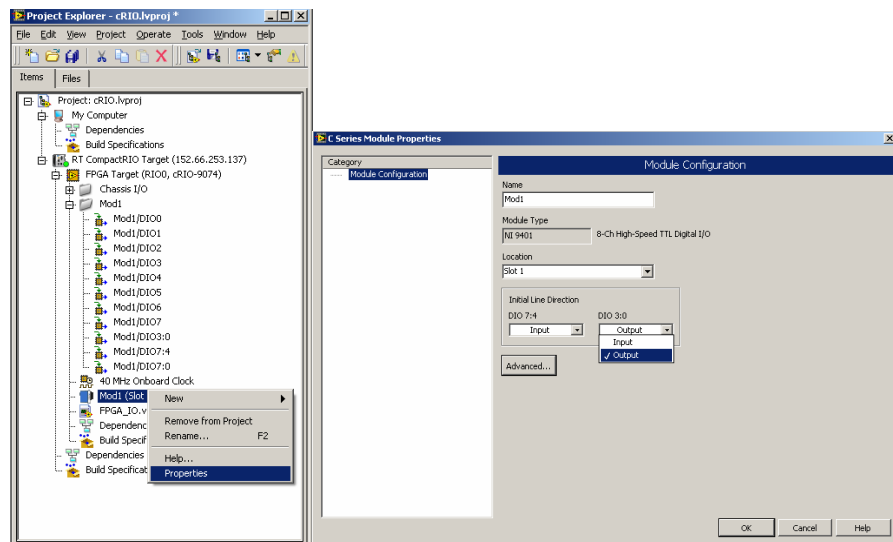


Ezekkel a lépésekkel gyakorlatilag a projekt környezetet előállítottuk.

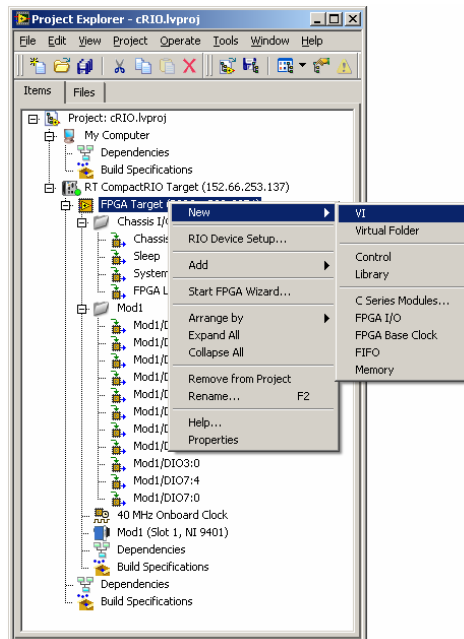
LED villogtatás



A következő példákban megnézzük, hogyan kell működésbe hozni a CompactRIO egyes mérő moduljait. Első lépésként konfiguráljuk a NI-9401 modul DIO0-DIO3 csatornáját kimenetnek. A project ablakban található Mod1 fizikai reprezentációjaként megjelenő ikon *Properties* menüpontjában (Ne felejtjük el utána elmenteni a projektet).

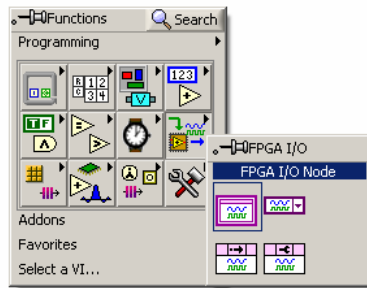


A vezérlés létrehozásához adjunk a projekthez hozzá egy új FPGA VI-t, ez fogja ellátni az egyes I/O lábak vezérlését, hiszen ez a legalsó hardware közeli réteg (Az *FPGA Target*-en jobb egérgomb, majd *New/VI*).

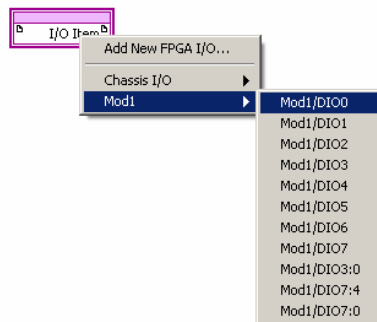


Az új VI rögtön szerkeszthetővé is válik. Rakjunk le a *Front panel*-re egy kapcsolót és a *Block diagram*mon rakjunk mellé egy *FPGA I/O*

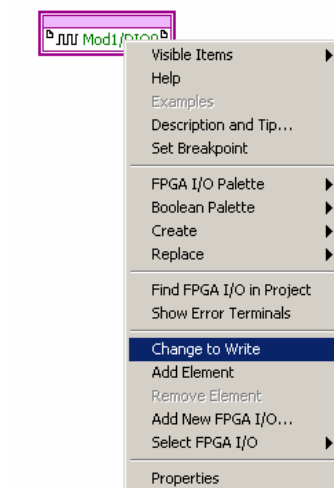
Node-ot, amely egy mérő kártya csatornát fog megszemélyesíteni (Ez megtalálható a *Block diagram* szerkesztő palettáján).



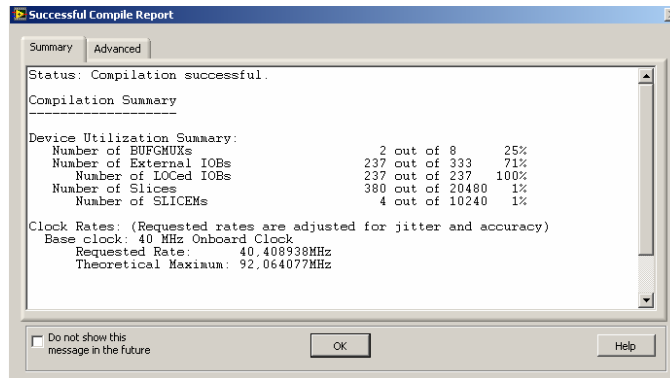
A lehelyezett *FPGA I/O Node*-ra kattintva hozzá tudjuk rendelni ezt a grafikus blokkot valamelyik tényleges fizikai csatornához.



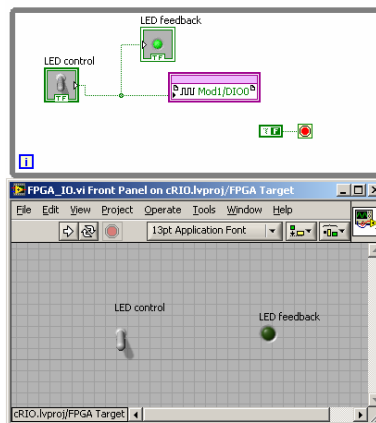
Rendeljük hozzá az NI-9401 modul DIO0-ás csatornájához. Mivel írni szeretnénk az adott lábra, váltsuk át az *I/O Node*-ot írhatóvá, a hozzá tartozó jobb egérgombbal elérhető menü *Change to Write* menüpontjának segítségével.



Huzalozzuk össze a blokkdiagramot, *(ne felejtjük az egész funkcionalitást egy végtelen ciklusba helyezni)*, majd indítsuk el a végrehajtását a szokásos módon. Ennek hatására generálódik a háttalban lévő FPGA-hoz tartozó huzalozási bitfile (Nyugodtan dőlünk hátra ez legalább 5 percig tart.). A programgenerálás végén megjelenik egy státuszriport, ami tájékoztat a fordítás eredményéről.

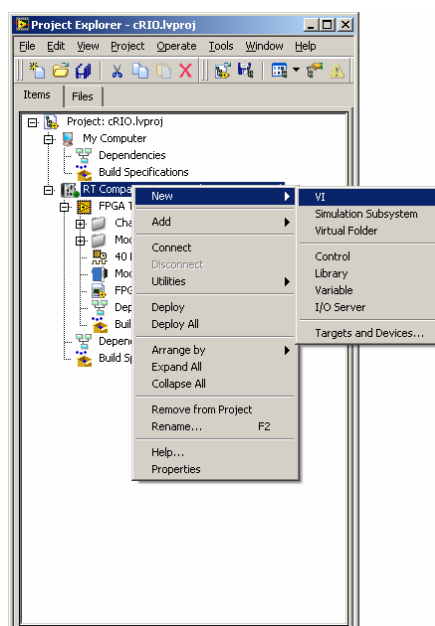


Az OK gombot aktiválva automatikusan elindul az FPGA VI és lehetőségünk van direktbe vezérelni az adott lábat a *Front Panel*-en keresztül, ezzel megvalósítva a LED villogtatás funkciót.



A RealTime PC

Következő lépésként bővítük a projektet úgy, hogy készítsünk programot a CompactRIO-ba integrált vezérlőre is. Adjunk egy VI-t hozzá a CompactRIO modul real-time vezérlőjéhez.



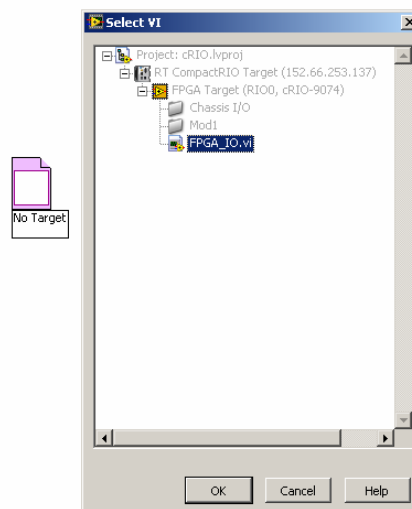
FPGA Interface

Ahhoz, hogy a CompactRIO real-time vezérlőjéből használni tudjuk az egyes I/O csatornákat először kapcsolatot kell teremtenünk a FPGA program és a real-time vezérlő programja között. Ez nem teljesen a hagyományos, VI-SubVI kapcsolaton keresztül működik, de annyi hasonlóság van benne, hogy ebben az esetben is az FPGA-hoz tartozó VI *Front Panel*-jére helyezett bemeneteket és kimeneteket tudjuk mintegy függvényhívási felületként kezelni.

Az FPGA-val való kapcsolat megteremtéséhez először meg kell neveznünk azt az FPGA VI-t, amit erről a magasabb absztrakciós szintről vezérelni kívánunk. Ezt a *Block diagram* palettájában található *FPGA Interface/Open FPGA VI Reference* blokkal tudjuk megtenni.

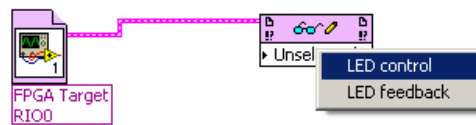


Az *Open FPGA VI Reference* VI lehelyezése után arra rákattintva ki tudjuk választani a projectben használt FPGA VI blokkot.

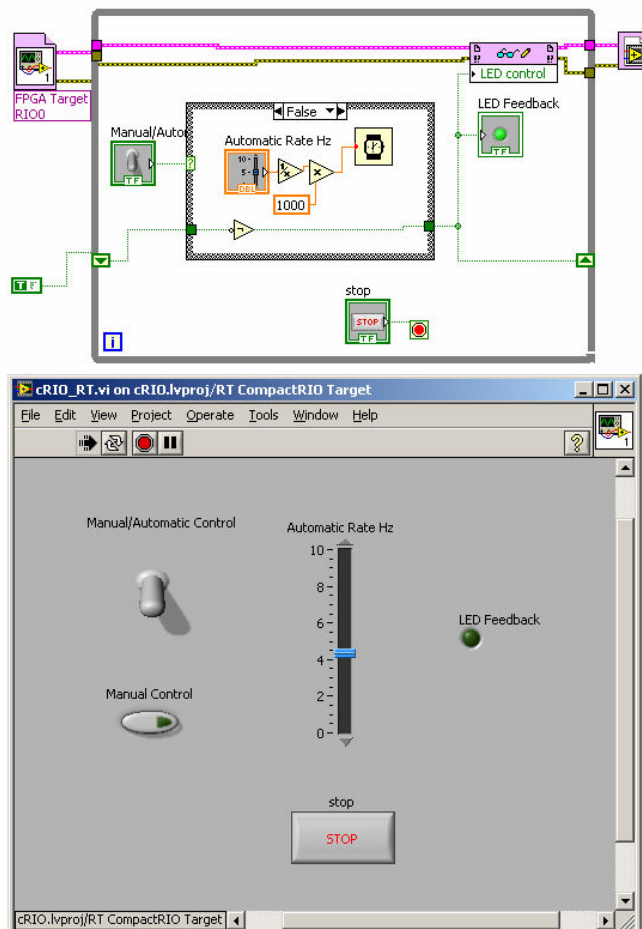


Következő lépésként a szintén *FPGA Interface* könyvtárban található *Read/Write Control* VI-val tudjuk az FPGA vezérlő VI *Front Panel*-jére helyezett bemeneteket és kimeneteket kezelni. Ezt lehelyezve majd a

referenciáját összehuzalozva az *Open FPGA VI Reference*-el ki tudjuk választani a számunkra szükséges be-/kimenetet.

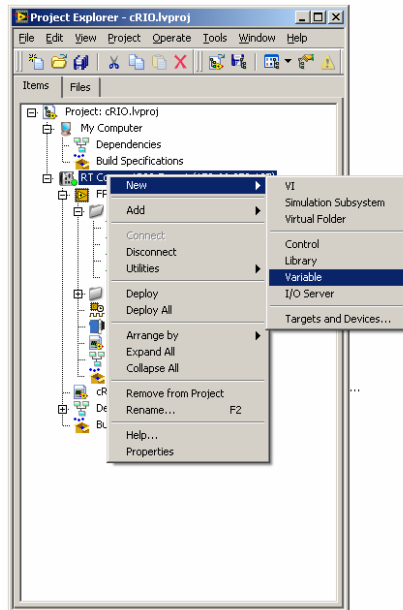


A magasabb szintű vezérlést demonstrálandó készítünk egy programot, amely vagy egy adott frekvenciával, vagy kézi vezérléssel villogtatja a LED-et. Ne felejtjük el a program végén *Close FPGA VI Reference*-el lezárni az FPGA interfészt.



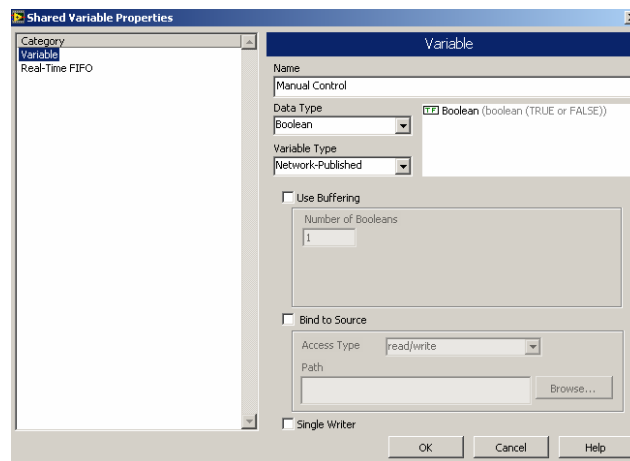
A kész VI-t elindítva, ugyanúgy, mint az FPGA-s esetben letöltésre kerül a CompactRIO egységre a program és elindul.

Felhasználói felület A terepre kihelyezhető modulra már átnéztük, hogy hogyan kell programot készíteni. A következő feladat, hogy megvizsgáljuk, hogy ez egy *Desktop* alkalmazással hogyan tud kapcsolatot tartani. Erre az ún. *Shared variable* módszer használható a legegyszerűbben. Ezzel gyakorlatilag megadjuk a real-time PC-n és az asztali PC-n futó alkalmazás számára is, hogy mely változókat használnak közösen, és a LabVIEW a két node közötti adatátvitellel automatikusan szinkronizálja a változók értékét a két helyen.

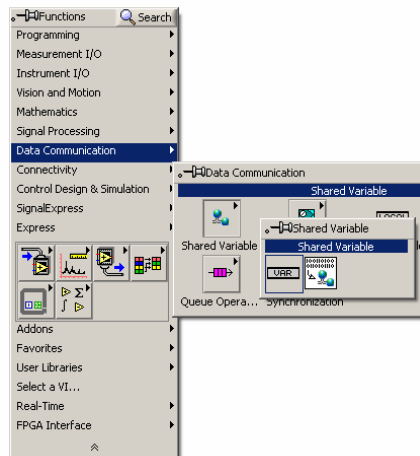


Shared variable

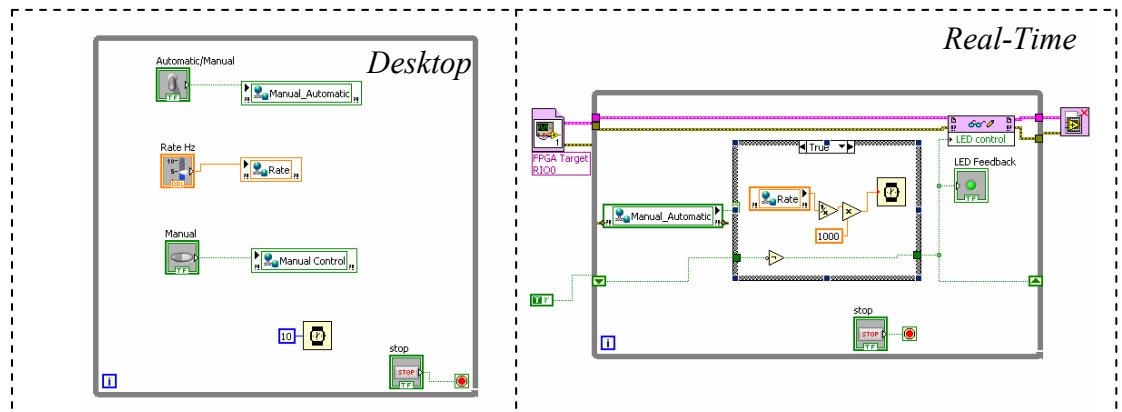
Adjunk hozzá a project CompactRIO részéhez három változót a real-time PC-n futó kezelőszervek (*Manual/Automatic Control switch, a Manual Control gomb és az Automatic Rate Hz*) kiváltásához



Hozzunk létre egy VI-t a *Desktop* részhez is a felhasználói felület számára. Adjuk hozzá az új VI *Front Panel*-jéhez a megfelelő kontrolokat. Hozzunk létre egy új osztott változót, a *Block diagram Data Communication/Shared Variable/Shared Variable VI*-jával.



A lerakott ikonra kattintva hozzá tudjuk rendelni a VI-t a projekt egyik osztott változójához. Azt, hogy a változóba írunk, vagy olvasunk a jobb egér gombbal elérhető *Change to Write* vagy *Change to Read* menüponttal tudjuk megadni. Ismételjük meg ezt a műveletet az összes többi változóra, valamint a CompactRIO VI-ra is.



A rendszer működtetéséhez minkét VI-t el kell indítani.

Tanácsok

Bonyolultabb rendszereknél célszerű hibavédelmet is alkalmazni, illetve a real-time rendszeren lévő VI leállítását megoldani a Desktop PC-ről. További hasznos tanács, hogy sok esetben célszerű az ún. ütemezett ciklusokat használni *Timed Loop*-mert ezek gyakorlatilag különálló szállakká képződnek le a végrehajtásnál, ezért garantáltan párhuzamosan működnek, illetve jól időzíthetőek. Jelen esetben például, ha a *Desktopon* futó VI-ból kiszedjük a késleltetést, akkor az akár a rendszer fagyásához is vezethet, mert annyira túltömi az osztott változós kommunikációt. Szintén érdemes megjegyezni, hogy a fejlesztés során akár a hierarchia egyes szintjeit át is ugorhatjuk, például nem kötelező a CompactRIO real-time vezérlőjéhez programot írni, lehet közvetlenül a desktop PC-ről vezérelni az FPGA VI-t.

4.3 Sugárzás monitorozó készítése

(2. mérés: *Ismerkedés a CompactRIO rendszerrel* záró feladata).

Cél

A BME tanreaktoránál a zónában elhelyezkedő sugárzásmérő fejek az általuk kiadott pulzusok gyakoriságával jelzik a sugárzás mértékét. Feladat, hogy a *Zóna* körül elhelyezkedő 4 mérőfejhez készítsen adatgyűjtő/feldolgozó egységet.

Az adatgyűjtő fő tulajdonságai a következők:

- a. A 4 csatornán érkező pulzusok minimális hossza 1 μ s, a pulzusok gyakorisága 1Hz és 100kHz között változhat.
- b. Mindegyik csatornához tartozó sugárzásérték jelezzük ki a Vezérlő panelen
- c. A RealTime modul önállóan vezéreljen 4 lámpát (LED) amely a villogási frekvenciája megegyezik a bejövő pulzusok frekvenciájával.
- d. A RealTime modul rögzítse a háttértárára a mérési eredményeket egy file-ba.