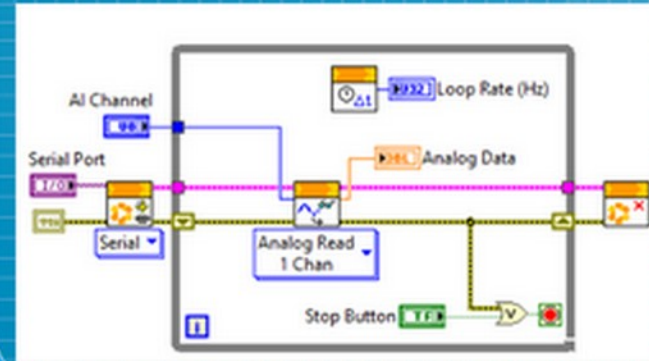
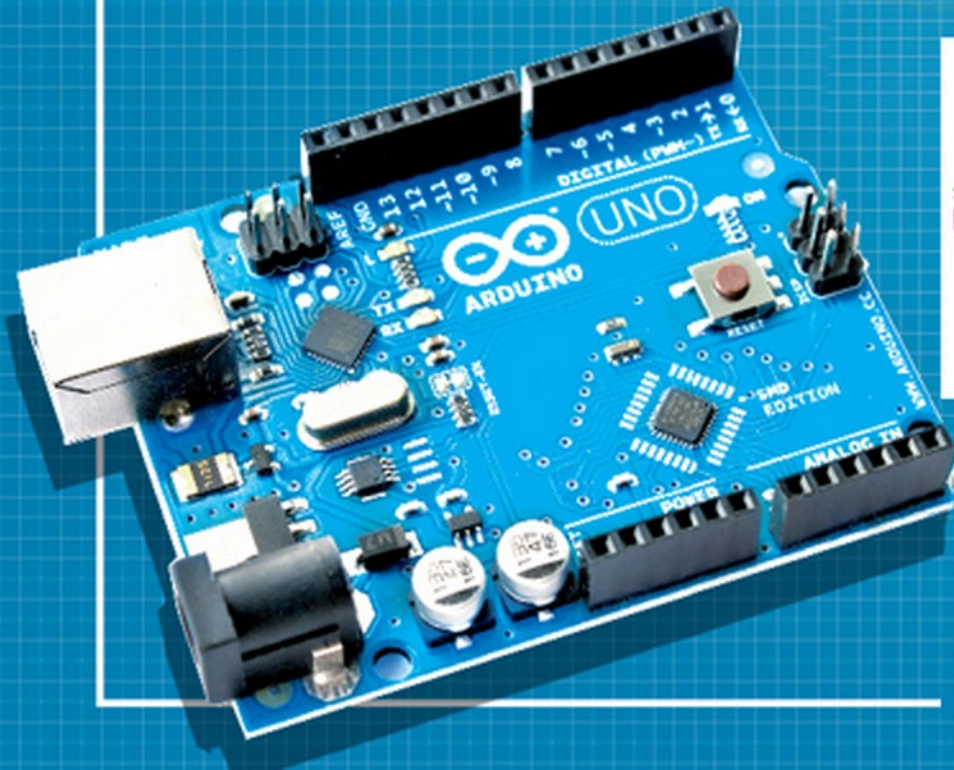


12. LabVIEW + LINX + Arduino - 2. rész

LabVIEW for Arduino

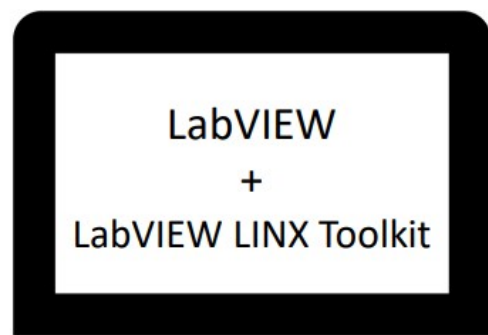


LabVIEW
MakerHub

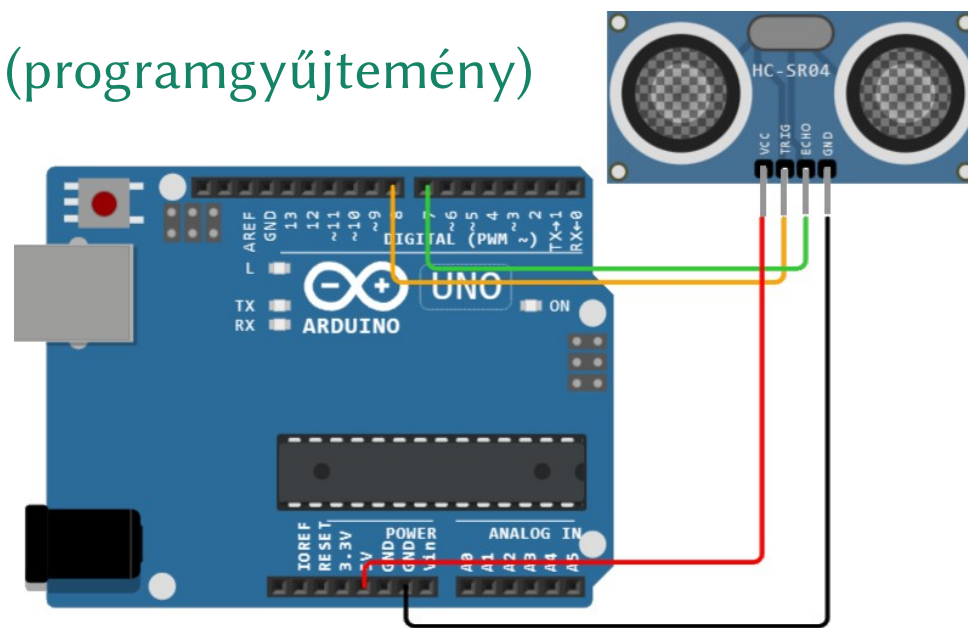
Felhasznált és ajánlott irodalom

- NI: [Getting Started with Arduino and LabVIEW Community Edition](#)
- NI: [LabVIEW Documentation](#)
- Szabó Norbert: [LabVIEW bevezető](#)
- Jáger Attila: [LabVIEW alapismeretek: 1. fejezet, 2. fejezet, 3. fejezet](#)
- Friedl Gergely: [LabVIEW segédlet](#)
- Hans-Petter Halvorsen: [LabVIEW LINX and Arduino](#)
- SIN Consulting: [LabVIEW-Basics](#) (programgyűjtemény)

PC



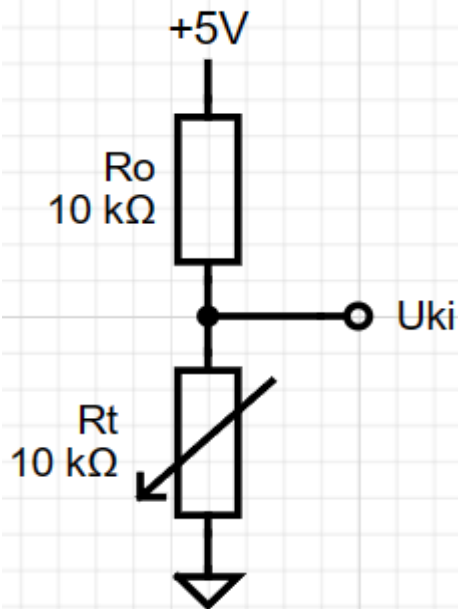
USB cable Type A-B



Hőmérés termisztorral

- **A termisztor** olyan ellenállás, amelynek a hőmérséklet függvényében megváltozik az ellenállása
- Hőmérésre többnyire negatív hőfoktényezőjű (NTK) ellenállást használunk, melynek hőfüggése nem lineáris
- **Steinhart-Hart egyenlet:**
$$\frac{1}{T_K} = A + B \cdot \ln(R_t) + C \cdot (\ln(R_t))^3$$

ahol A, B, C konstansok: $A=0.001129148$,
 $B=0.000234125$, $C=8.76741E-08$

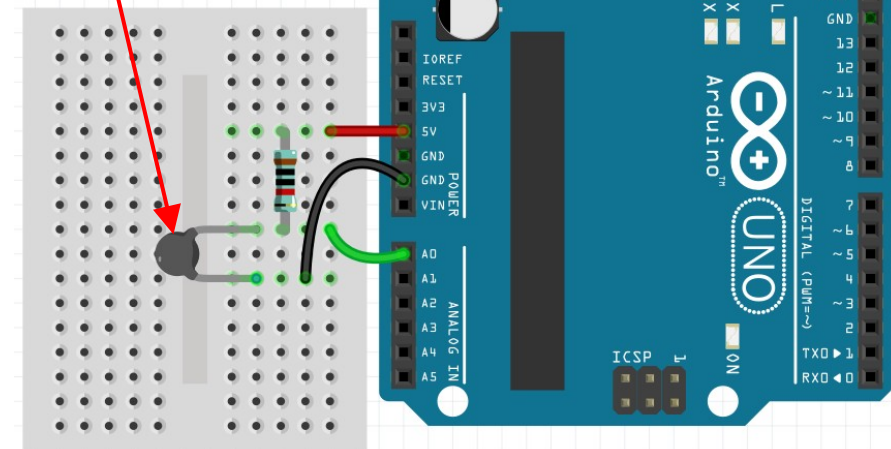


$$U_{ki} = \frac{5V \cdot R_t}{R_t + R_0}$$

↓

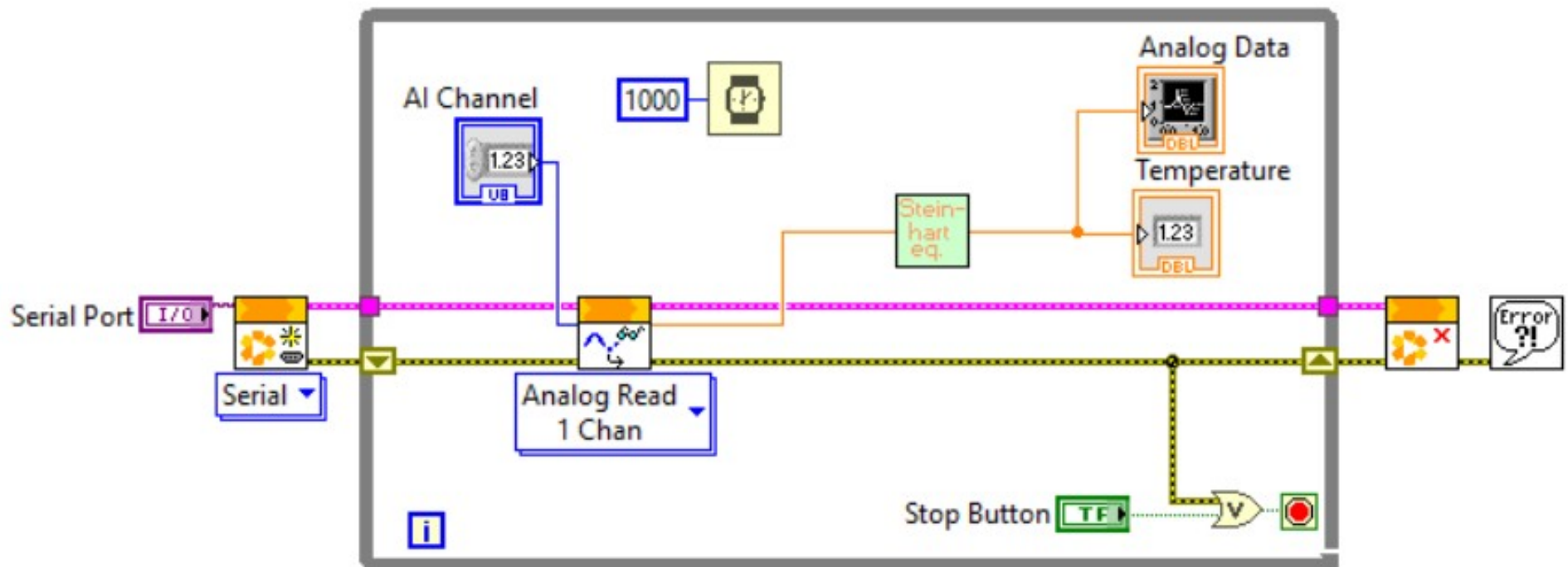
$$R_t = \frac{U_{ki} \cdot R_0}{5V - U_{ki}}$$

Termisztor



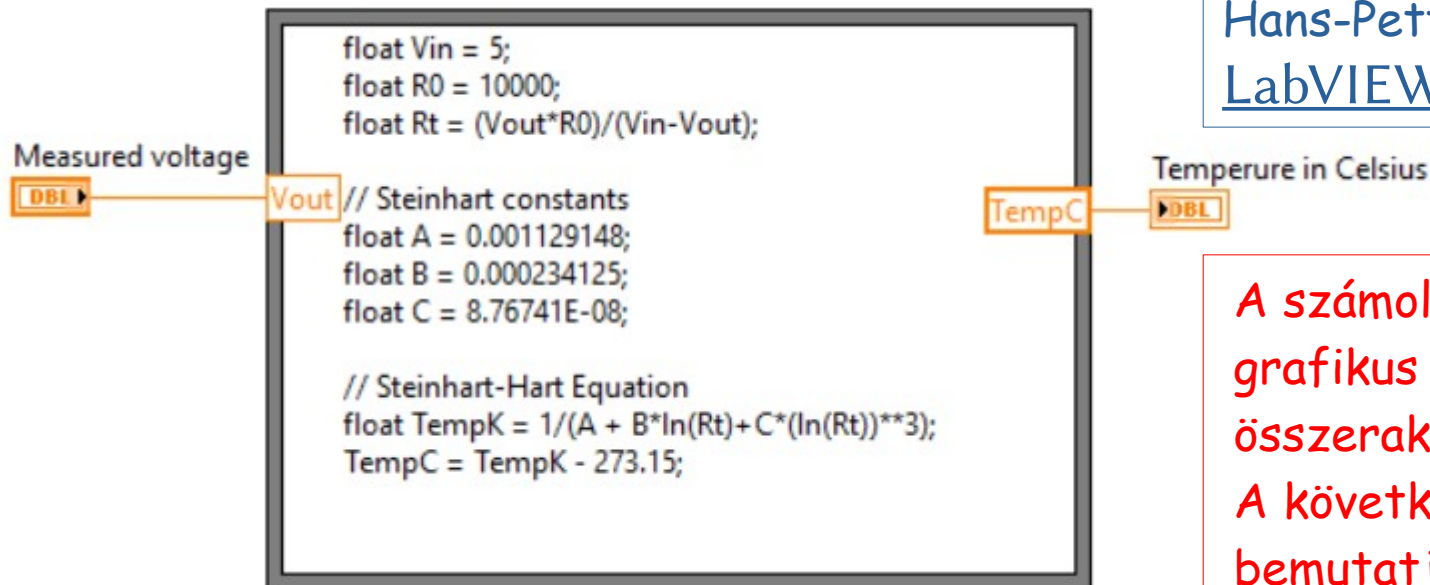
LINX – Thermistor Read (modified).vi

- Az alábbi programot nulláról kiindulva is „összelegőzhatjuk” az előző előadás útmutatása szerint, de kényelmesebb a **LabVIEW LINX** mintaprogramjai közül a [LINX - Thermistor Read.vi](#)-ből kiindulni
- A program egyetlen analóg csatornát kezel, s a kiolvasott értékből kiszámított hőmérsékletet numerikusan és grafikusan megjeleníti
- A **Steinhart-Hart egyenletet** tartalmazó blokkot most egy újrahasznosítható, ún. *subVI* formájában készítjük el.



Steinhart-hart.vi

- Néha kényelmesebb a számolást szövegesen leírni, erre szolgál a **Structures** lapon a **Formula Node** elem
- A **File** → **New VI** menüpont új lapot nyit, itt helyezzünk el egy **Formula Node**-ot és írjuk bele az alábbi sorokat!
- A jobb egérgombbal a keretre kattintva **Add Input** és **Add Output** segítségével készíthetünk kimeneti és bemeneti változókat, amelyekhez **Create Control** és **Create Indicator**-ral rendeljük duplapontos számbeírót, illetve kijelzőt!

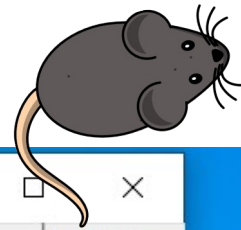


Forrás:
Hans-Petter Halvorsen:
[LabVIEW LINX and Arduino](#)

A számolást természetesen grafikus programozással is összerakhattuk volna. A következő programban majd bemutatjuk, hogy hogyan...

Steinhart-hart.vi – kivezetések hozzárendelése

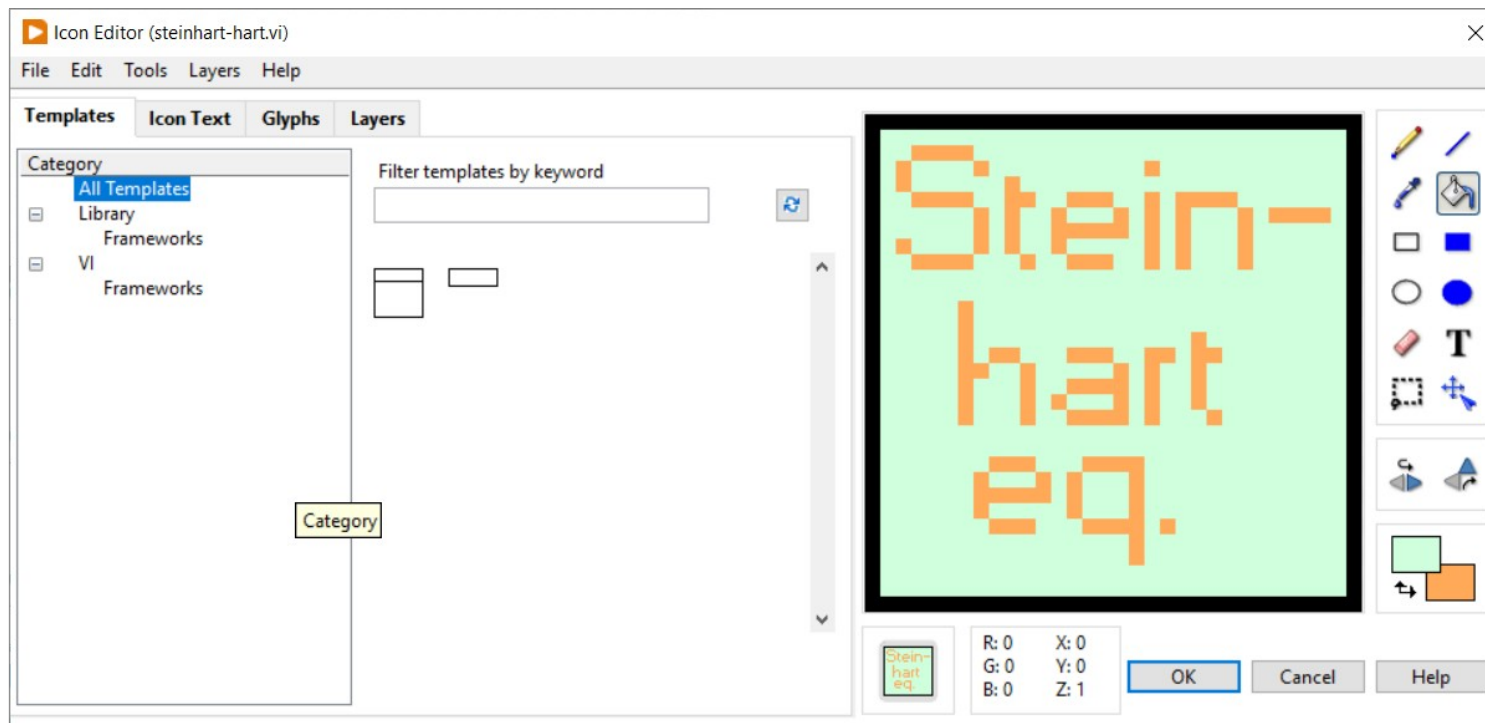
- A front panel jobb felső sarkában a jobb egérgombbal a bal oldali ikonra kattintva a **Pattern** menüből válasszuk ki azt az alakzatot, amelynek egy bemenete és egy kimenete van (itt kézzel keretezve látszik)
- Ugyanezen ikonra bal gombbal kattintva rendelhetjük hozzá egyenként a be- és kimenetekhez az előlapi elemeket (azokra is bal gombbal kattintva)



The screenshot displays the LabVIEW software interface for a project named 'steinhart-hart.vi Front Panel *'. The menu bar includes 'File', 'Edit', 'View', 'Project', 'Operate', 'Tools', 'Window', and 'Help'. The toolbar shows various icons for navigation and editing, with the font set to '15pt Calibri'. The main workspace contains two numeric indicators: 'Measured voltage' and 'Temperure in Celsius', both showing the value '0'. A context menu is open over the workspace, listing options such as 'VI Properties', 'Set Icon to VI Name', 'Find All Instances', 'Add Terminal', 'Remove Terminal', 'Patterns', 'Rotate 90 Degrees', 'Flip Horizontal', 'Flip Vertical', 'Disconnect All Terminals', 'Disconnect This Terminal', and 'This Connection Is'. The 'Patterns' option is selected, and a grid of terminal icons is displayed to the right. One icon, representing a terminal with one input and one output, is highlighted with a blue border.

Steinhart-hart.vi – ikon hozzárendelés

- A front panel jobb felső sarkában a jobb egérgombbal a jobboldali ikonra kattintva, a felbukkanó menüből válasszuk az **Edit Icon** menüpontot!
- Ha a szerkesztéssel végeztünk, kattintsunk az OK gombra, majd **mentsük el** az így elkészített modult egy **.vi** állományba!
- Az elmentett **subVI** modult blokkszerkesztő módban jobbkattintás után a **Select a VI** menüpontban kereshetjük meg és csatolhatjuk be programjainkba



LINX – Thermistor Read (modified).vi

LINX - Thermistor Read (modified).vi

File Edit View Project Operate Tools Window Help

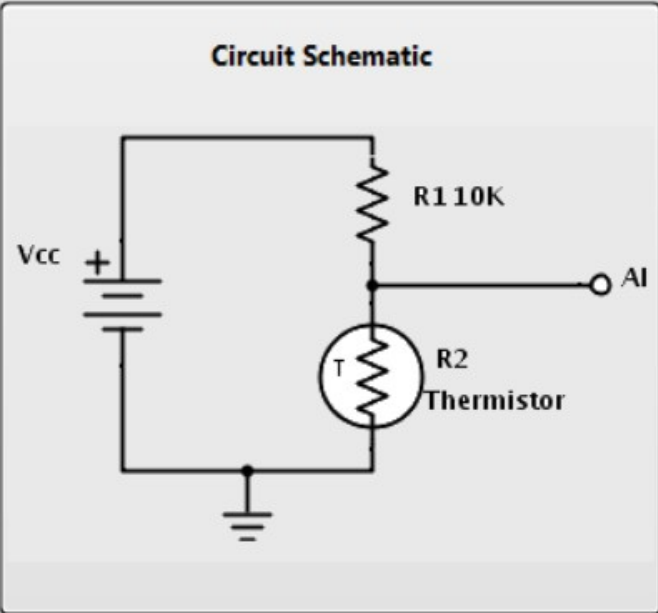
LINX - Thermistor Read Example (modified)

This example demonstrates how to read from a thermistor attached to an analog input

Instructions

1. Select the **COM Port** associated with the LINX Device.
2. Select the **Analog Channel** to read.
3. Click the **Run Arrow**.

Circuit Schematic



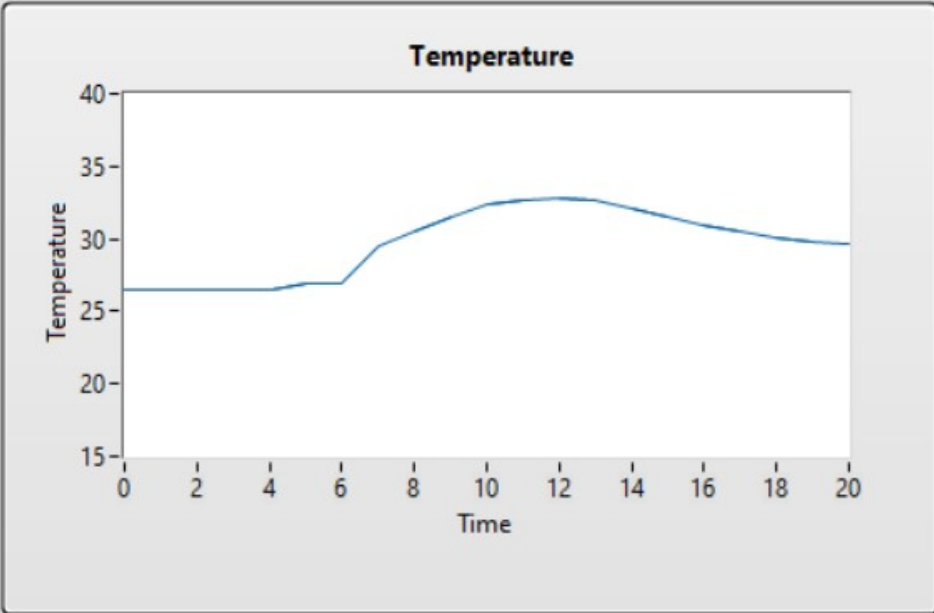
LINX Device Settings

Serial Port: COM4

AI Channel: 0

Temperature: 29,618

Temperature

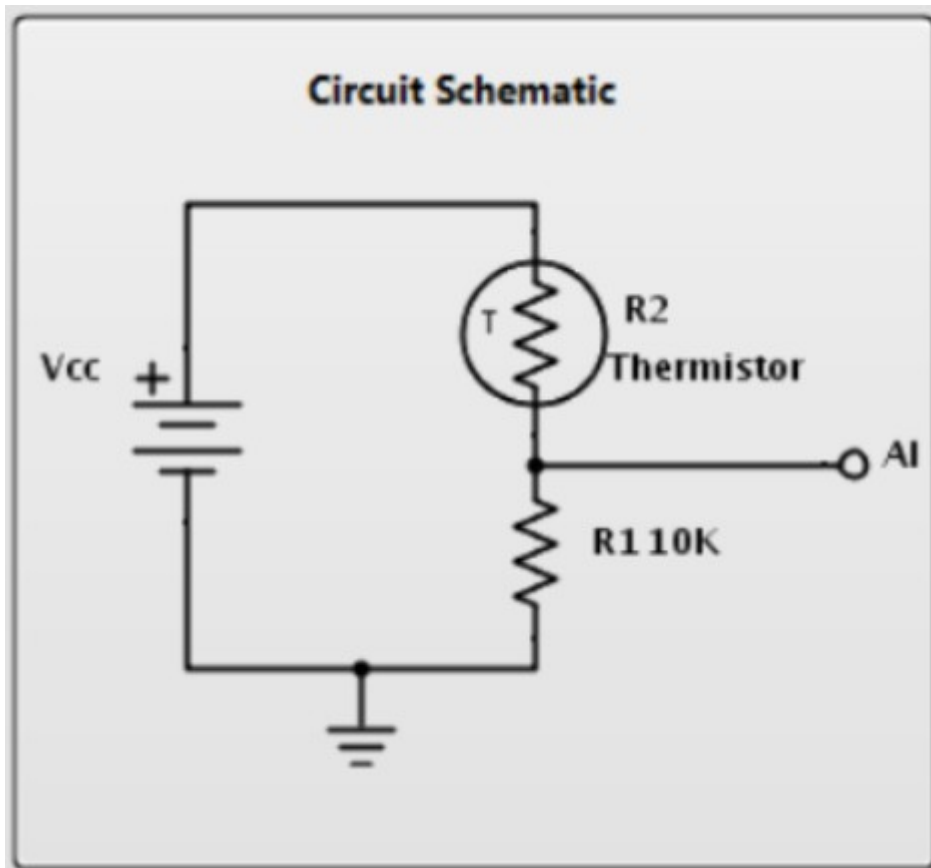


Time	Temperature
0	26.5
2	26.5
4	26.5
6	27.0
8	30.0
10	32.5
12	33.0
14	32.0
16	31.0
18	30.0
20	29.5

Stop

LINX-Thermistor-Read.vi

- Ez a „gyári” mintapélda fordítva, a feszültségosztó felső tagjában alkalmazza a termisztort, ennek megfelelően a feszültségmérés után egy kicsit más képlettel tudjuk meghatározni az ismeretlen R_t értékét
- Fentiekén túl a program egy apró, ám kellemetlen hibát tartalmaz, amit ki kell javítanunk



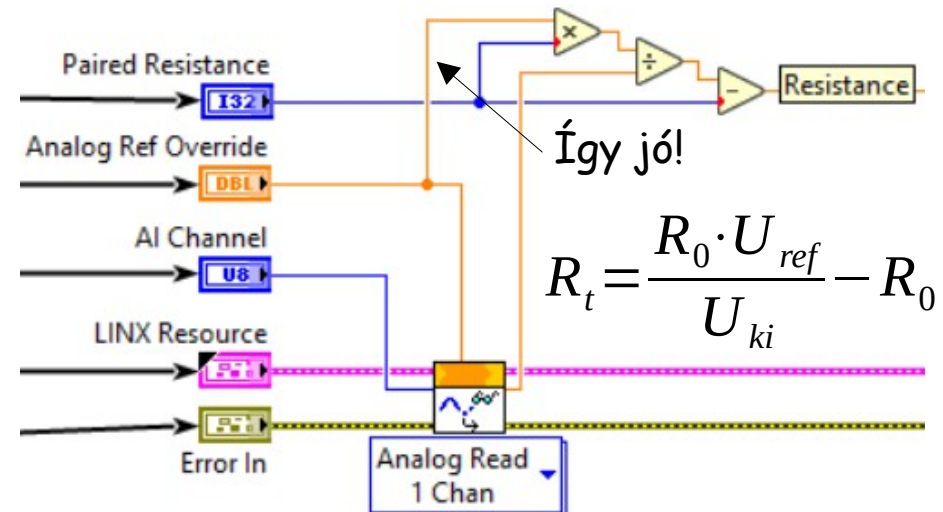
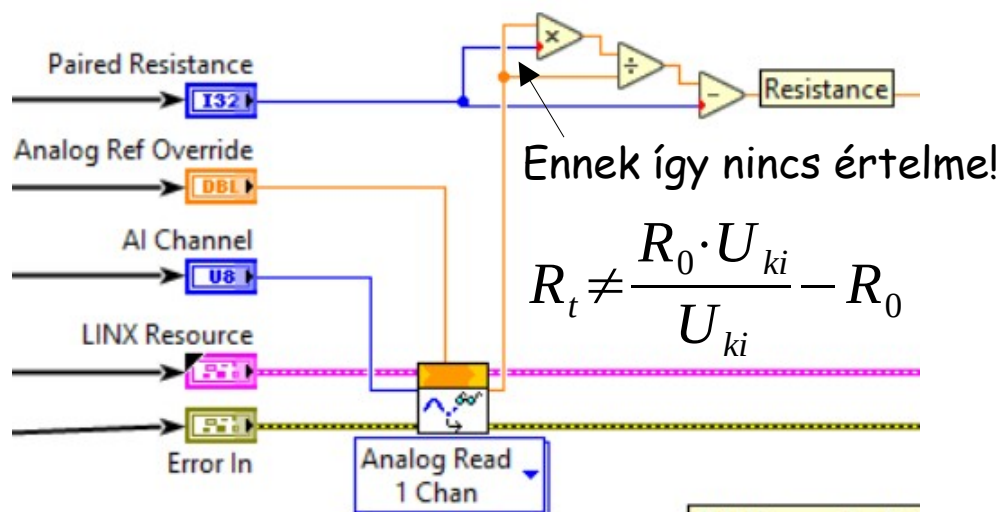
$$U_{ki} = \frac{U_{be} \cdot R_0}{R_t + R_0}$$



$$R_t = \frac{U_{be} \cdot R_0}{U_{ki}} - R_0$$

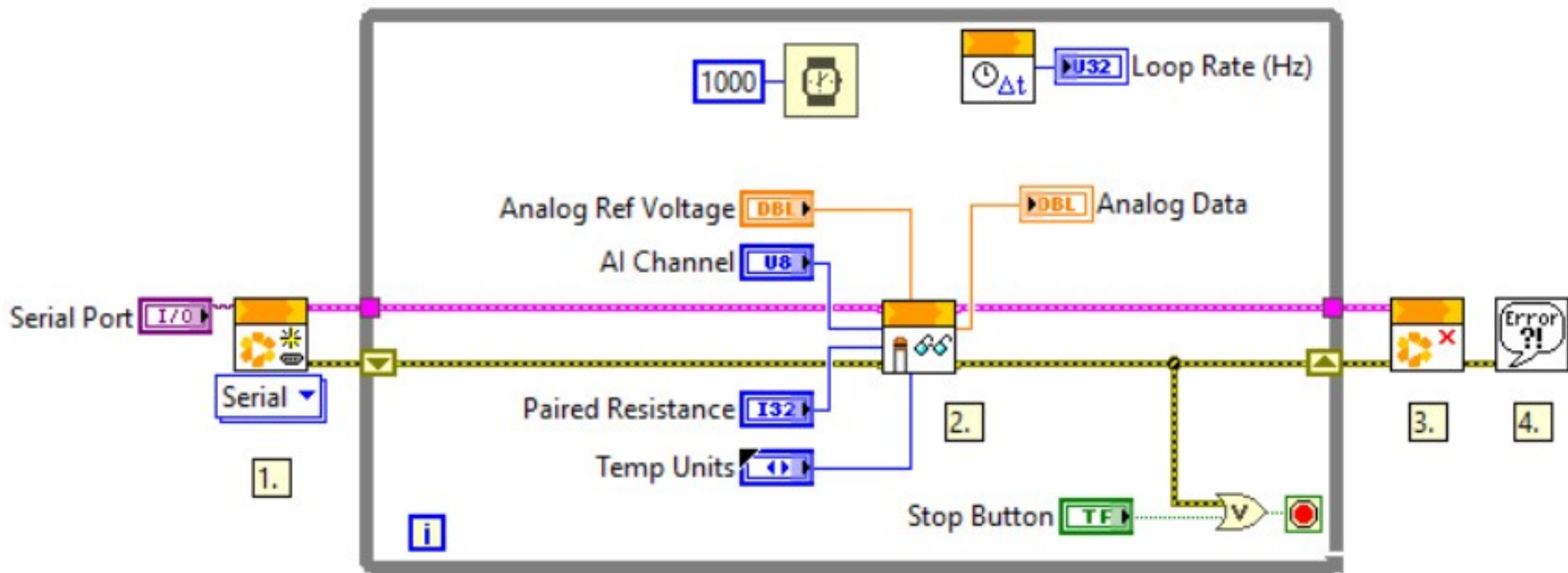
Mi a gond a gyári mintapéldával?

- A **LINX - Thermistor Read.vi** mintaprogrammal (ezt használtam fel a kiinduláshoz az előző programnál) két problémám volt:
- A programban szereplő **Thermistor.vi** modul nem jelenik meg a **Hobbyist/Sensors/Temp** lapon, csak a LabVIEW telepítési helyén a **vi.lib\MakerHub\LINX\Public\Sensors\Temperature** mappában található meg (a **Select a VI** menüpontnál lehet importálni)
- A másik probléma: a fenti **subVI** modulban van egy elkötés, ami miatt rosszul végzi a számolást. Szerencsére a hiba könnyen kijavítható, ha rájövünk, hogy *mire is gondolhatott a költő...*



LINUX-Thermistor-Read-Example.vi

- A gyári mintaprogram blogdiagramját megnyitva, középen láthatjuk a **Thermistor.vi** nevű *subVI* modult, amit ki kell javítani (duplakattintással tudjuk megnyitni)



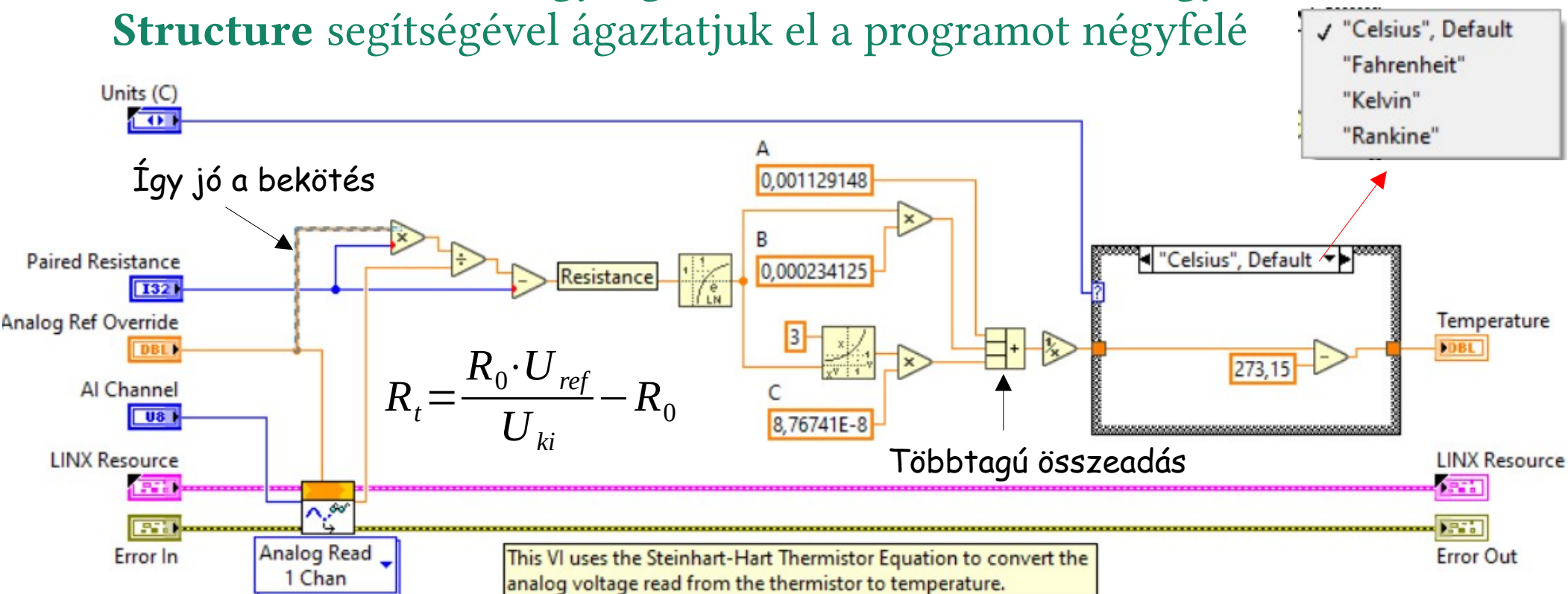
1. Open a connection to the LINX device.
2. Read from the thermistor attached to the specified analog channel.
3. Close the connection to the LINX device.
4. Handle Errors

LINX-Thermistor-Read-Example.vi

- A **Thermistor.vi** modul kiolvassa egy ADC mérés eredményét, majd a mért feszültségből kiszámolja a termisztor ellenállását (R_t)
- R_t értékéből pedig kiszámoljuk a Kelvin fokokban mért hőmérsékletet

$$\frac{1}{T_K} = A + B \cdot \ln(R_t) + C \cdot (\ln(R_t))^3$$

- A kiválasztott mértékegységre történő átszámításhoz egy **Case Structure** segítségével ágaztatjuk el a programot négyfelé



LINX-Thermistor-Read-Example.vi



LINX - Thermistor Read Example

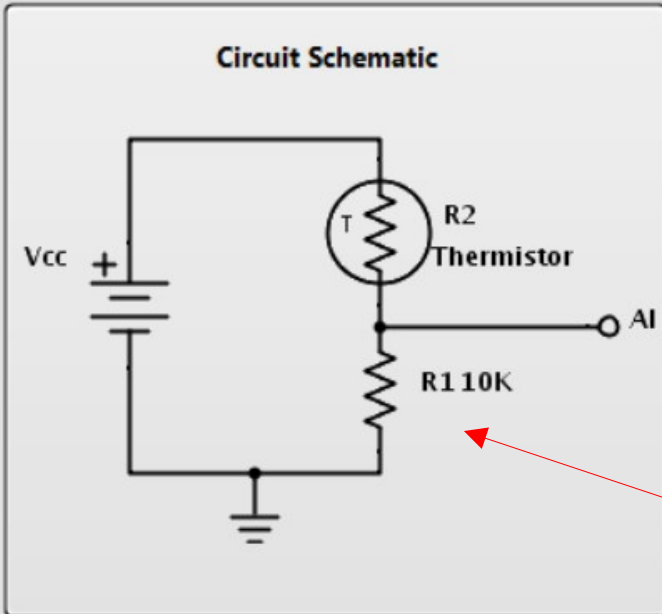
This example demonstrates how to read from a thermistor attached to an analog input

Instructions

1. Select the **COM Port** associated with the LINX Device.
2. Select the **Analog Channel** to read.
3. Click the **Run Arrow**.

Temp Units

Loop Rate (Hz)



LINX Device Settings

Serial Port

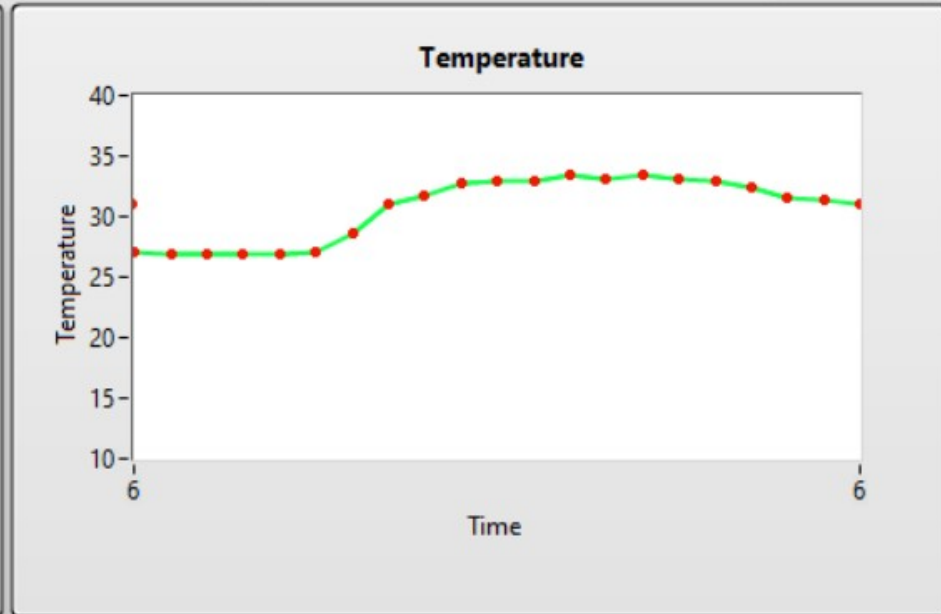
Analog Ref Voltage

Vcc

AI Channel

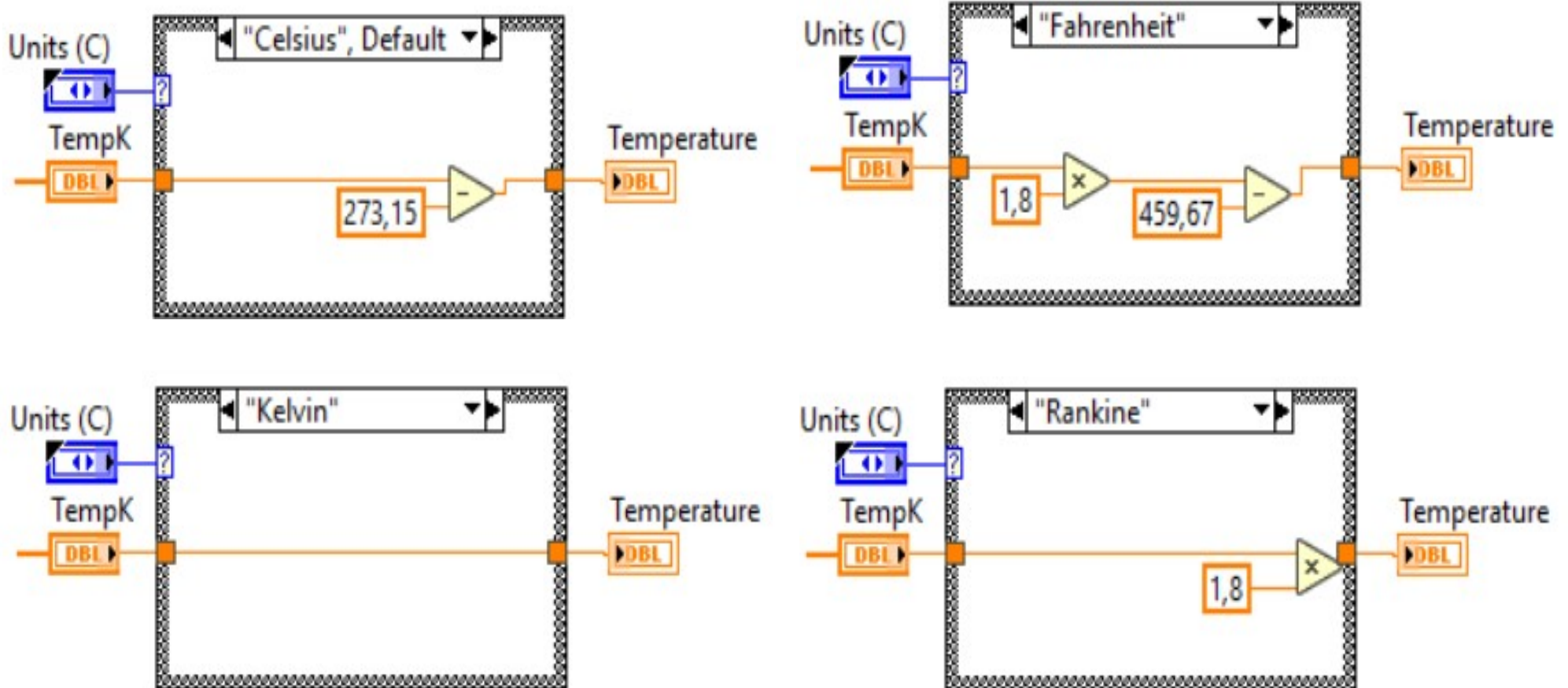
A0

Paired Resistance



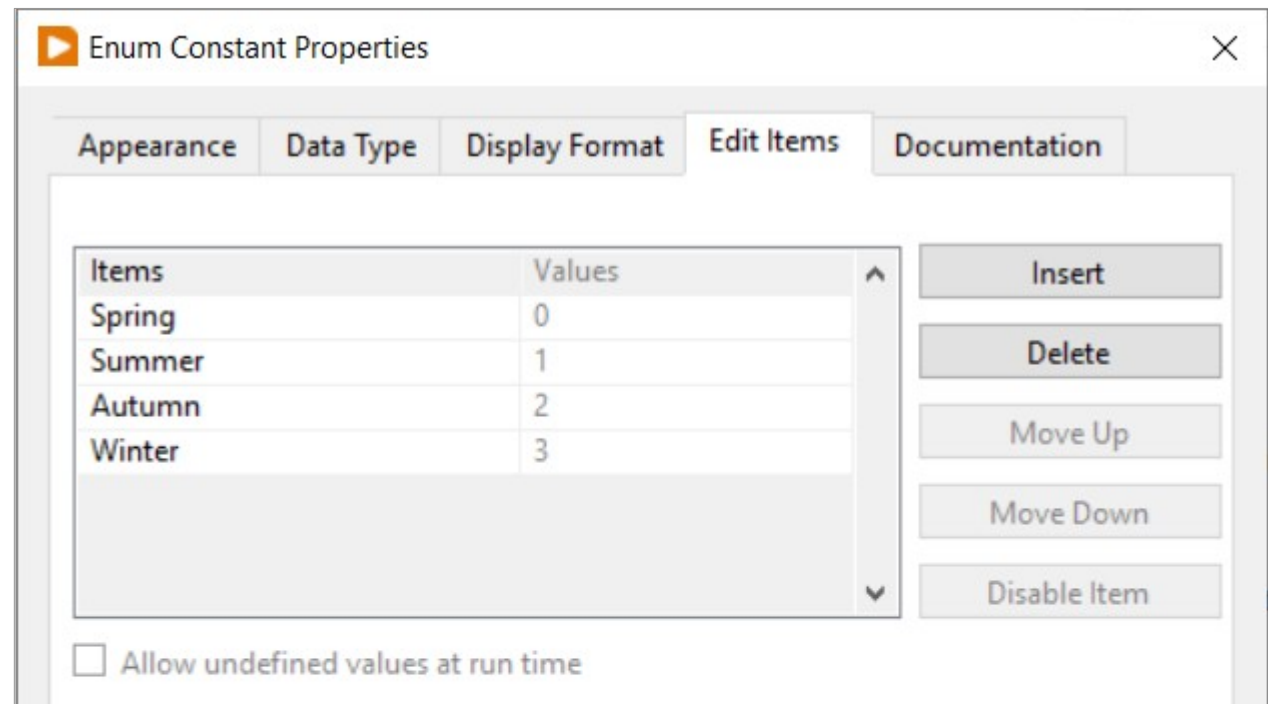
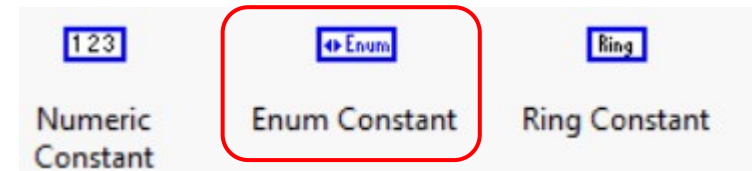
Hogyan működik a mértékegység váltás?

- Az előlapi mértékegység-választó egy *enum* típusú változót kezel, melynek értékétől függően egy **Case struktúrában** négyfelé ágazik a program és átszámítja az eredetileg Kelvin fokokban meghatározott hőmérsékletet a kért egységekre



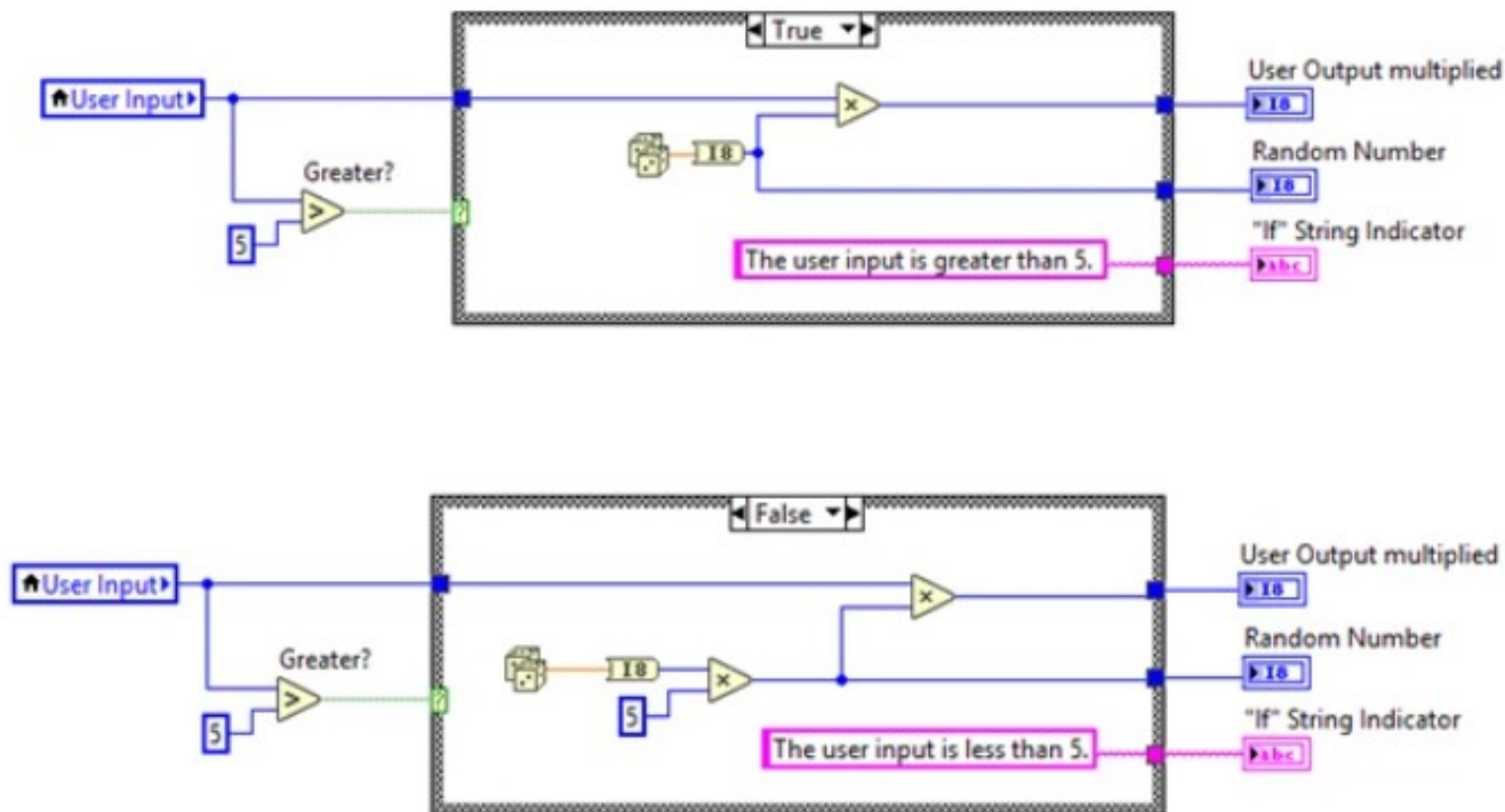
Enum konstans és változó

- Az Enum (felsorolás) típusú konstans a **Numeric** palettán található
- Létrehozás után az **Edit Items** menüpontban vagy a **Properties** menüben szerkeszthetjük
- Lényegében nevekből és hozzárendelt sor-számokból áll
- A **Change to Control** menüpontban tehetjük vezérlővé, hogy az előlapon is beállítható legyen



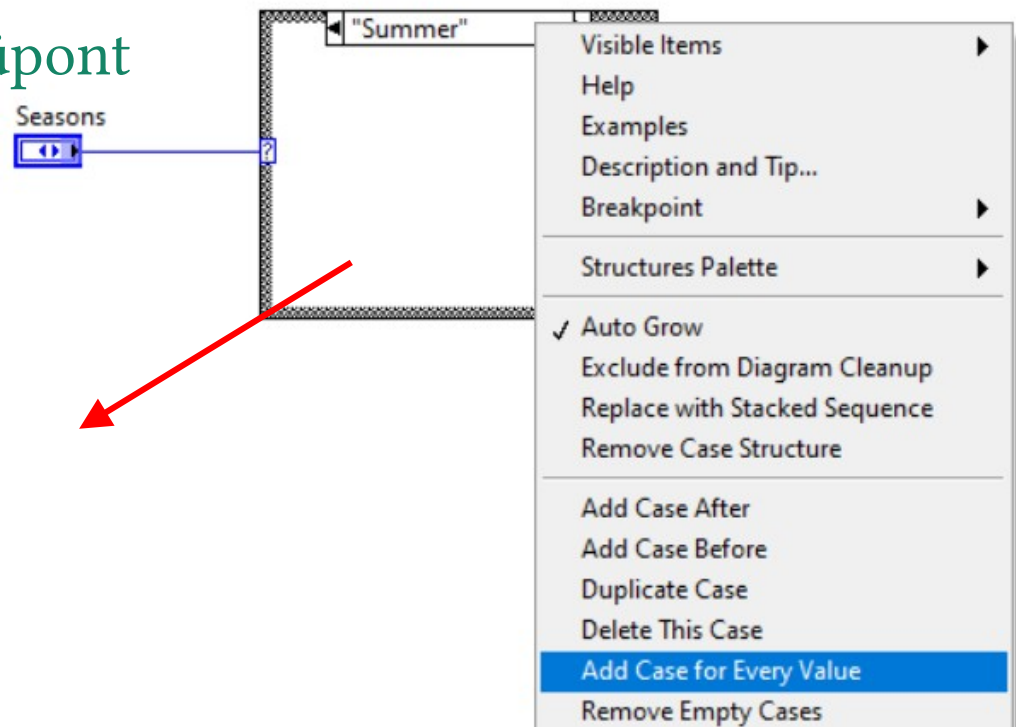
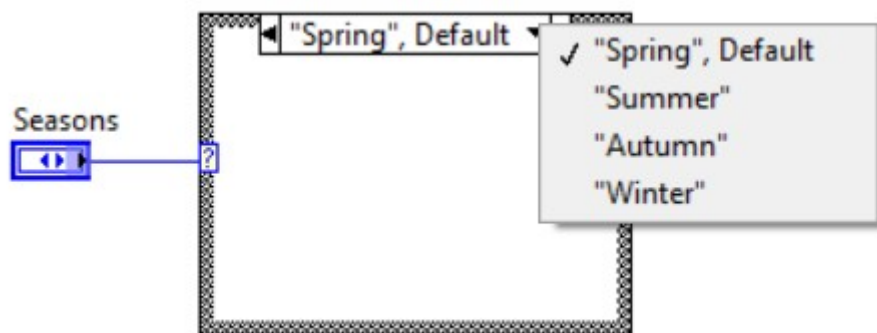
Az alapértelmezett Case struktúra

- A LabVIEW Case struktúrája (a *Structures* palettán található) a C nyelvi **if** és **switch** utasítások megfelelője. Alapértelmezetten csak két állapotot kezel, mint az **if** utasítás
- A True és False bemenethez tartozó aldiagramokat a felső esetválasztó beállítása után látjuk, illetve szerkeszthetjük



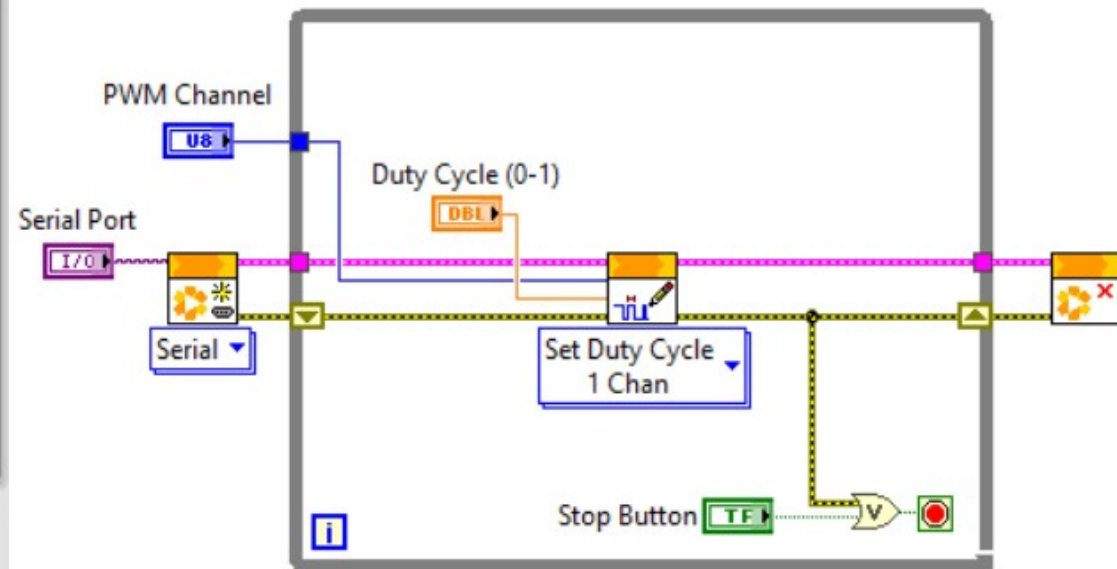
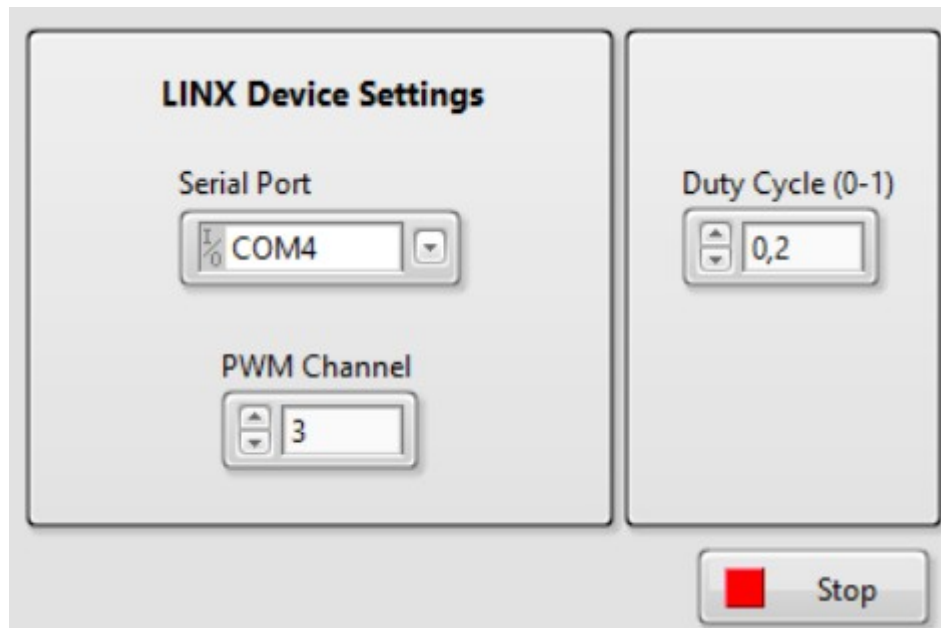
A kiterjesztett Case struktúra

- A LabVIEW Case struktúrája több esetre is kiterjeszthető, ha más típusú bemenő adatot rendelünk hozzá (pl. enum vagy numerikus)
- Például az előzőekben bemutatott Seasons nevű enum típusú vezérlőt kössük egy Case struktúra bemenetére
- A kiterjesztéshez a struktúra tetején látható esetválasztóra jobb gombbal kattintva használjuk az **Add Case...** kezdetű parancsokat
- A legkényelmesebb megoldás az **Add Case for Every Value** menüpont választása, amely létrehozza az összes szükséges esetet



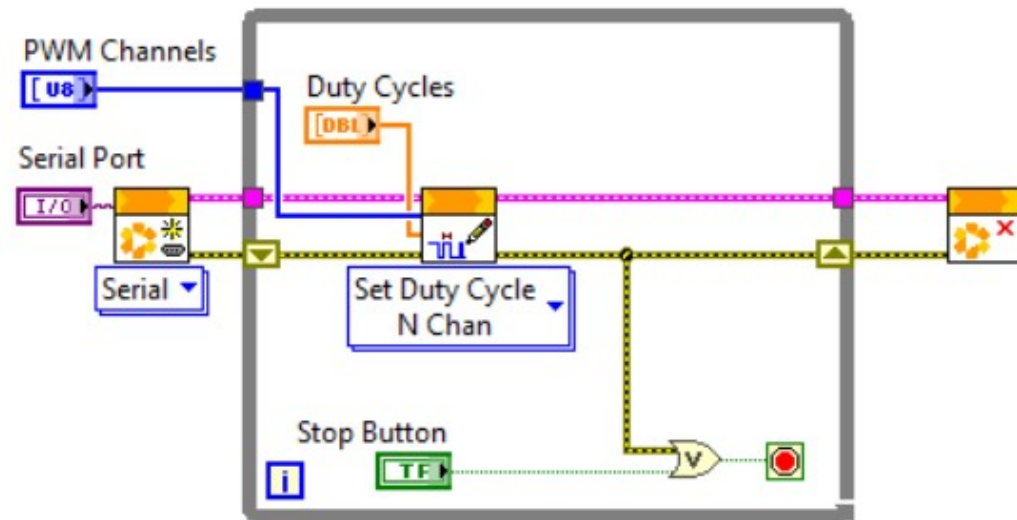
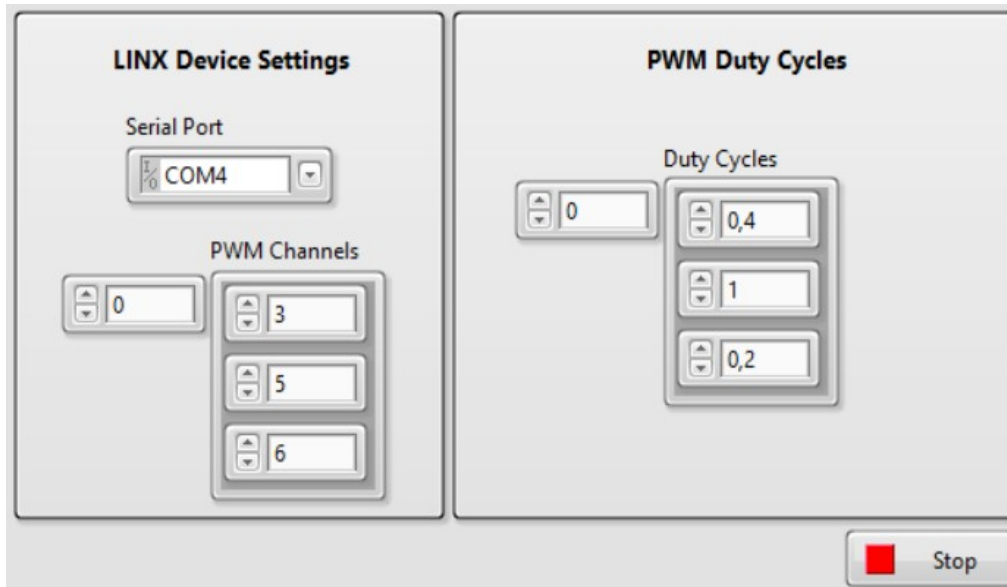
PWM – impulzusszélesség moduláció

- A PWM használatához csak egy **PWM Set Duty Cycle.vi** modult kell elhelyezni a programunkban
- Megjegyzés: ne higgyünk a dokumentációnak, nem 0..255 közötti érték adható meg kitöltésnek, hanem 0..1 közötti lebegőpontos szám!
- A **LINUX - PWM 1 Channel.vi** „gyári” mintapélda a **PWM Set Duty Cycle.vi** modul egycsatornás módú használatát mutatja be
A két bemenő paraméter ekkor a PWM-képes kivezetés sorszáma (U8) és a kitöltési tényező (0-1 közötti lebegőpontos szám)



Több PWM csatorna együttes kezelése

- A **PWM Set Duty Cycle.vi** modul N csatornás módja abban különbözik, hogy itt a PWM csatornák és a kitöltési tényezők megadása 1 dimenziós tömbökkel történik (ezt jelzi a blokkdiagramon a vastagabb vonal)
- A **LINUX - PWM N Channel.vi** „gyári” mintapélda a **PWM Set Duty Cycle.vi** modul N-csatornás módú használatát mutatja be
A bemenő paramétereket a tömbelemek rovatába írhatjuk be
- A tömbök előtt álló indexelőt ne babráljuk el, maradjon nulla!



LINX - Set RGB LED Color.vi

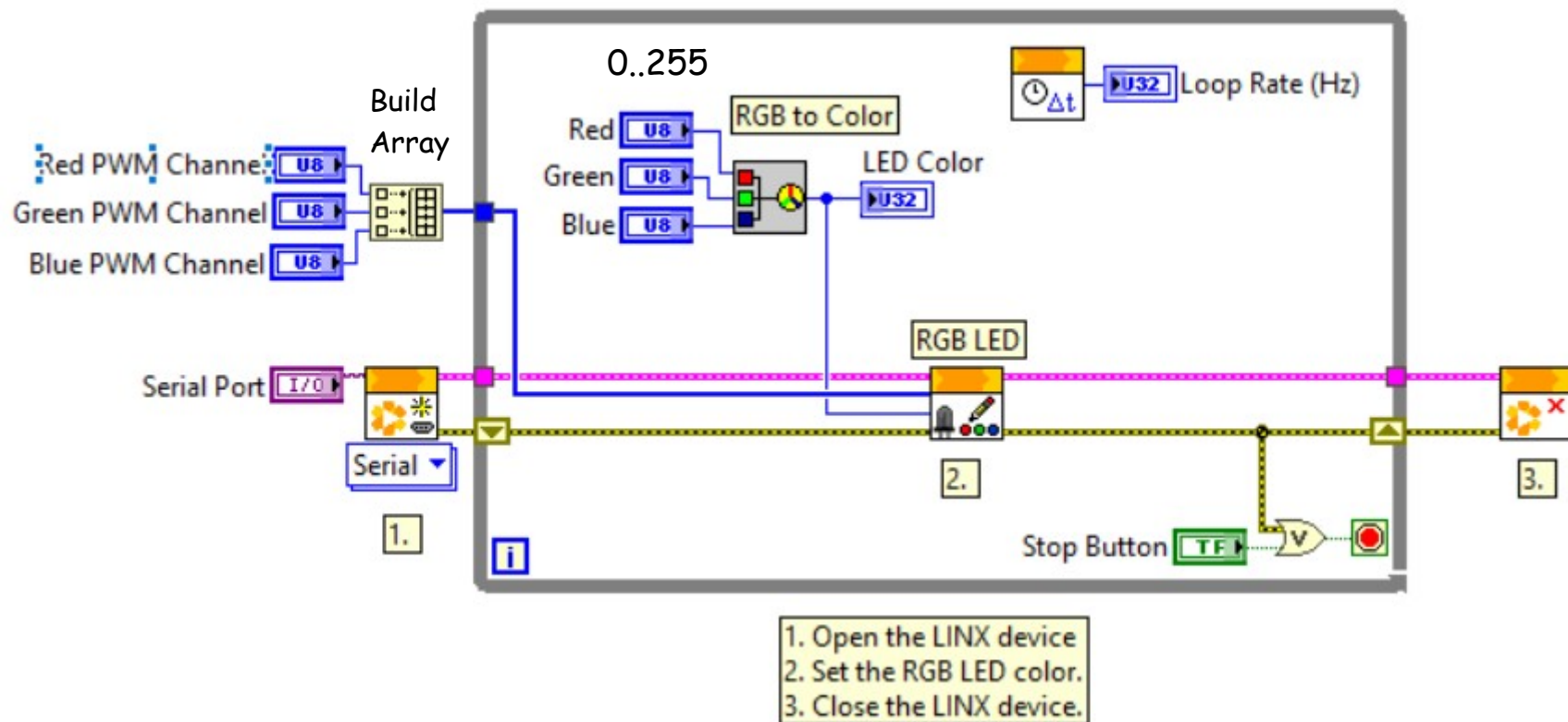
- Az N-csatornás PWM alkalmazására egy látványos példa az RGB LED vezérlése, a „gyári” program azonban apró javításra szorul

The screenshot displays the LabVIEW interface for the 'LINX - Set RGB LED Color.vi' program. The interface is divided into three main sections:

- Circuit Schematic:** Shows a wiring diagram for an RGB LED. The Blue PWM Channel is connected to the Blue LED (B) through a 220Ω resistor (R1). The Green PWM Channel is connected to the Green LED (G) through a 220Ω resistor (R2). The Red PWM Channel is connected to the Red LED (R) through a 220Ω resistor (R3). All LEDs are connected to a common ground (GND).
- LINX Device Settings:** Configures the communication with the LINX device. The Serial Port is set to COM4. The Red PWM Channel is set to 3, the Green PWM Channel to 5, and the Blue PWM Channel to 6.
- RGB LED Color:** Features three vertical sliders for Red, Green, and Blue intensity, each ranging from 0 to 255. The Red slider is at approximately 50, the Green at 200, and the Blue at 200. A circular LED Color preview shows a cyan color. A 'Loop Rate (Hz)' control is set to 192. A 'Stop' button is located at the bottom right.

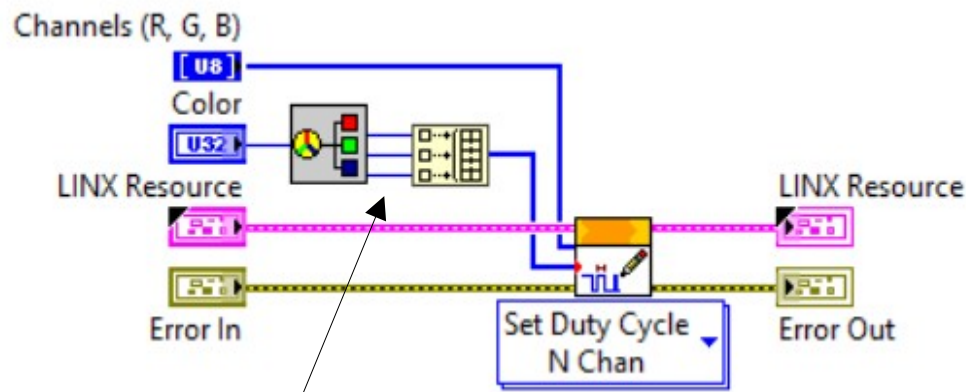
LINUX - Set RGB LED Color.vi

- A program három csúszka (0..255) adatait összesíti egy színváltozóban, majd az **RGB LED.vi** modul segítségével kikeveri a színt a három megadott PWM csatorna segítségével (a három csatornára egy közös katódú RGB LED három anódját kell csatlakoztatnunk, a közös katódot pedig a földre kell kötni)



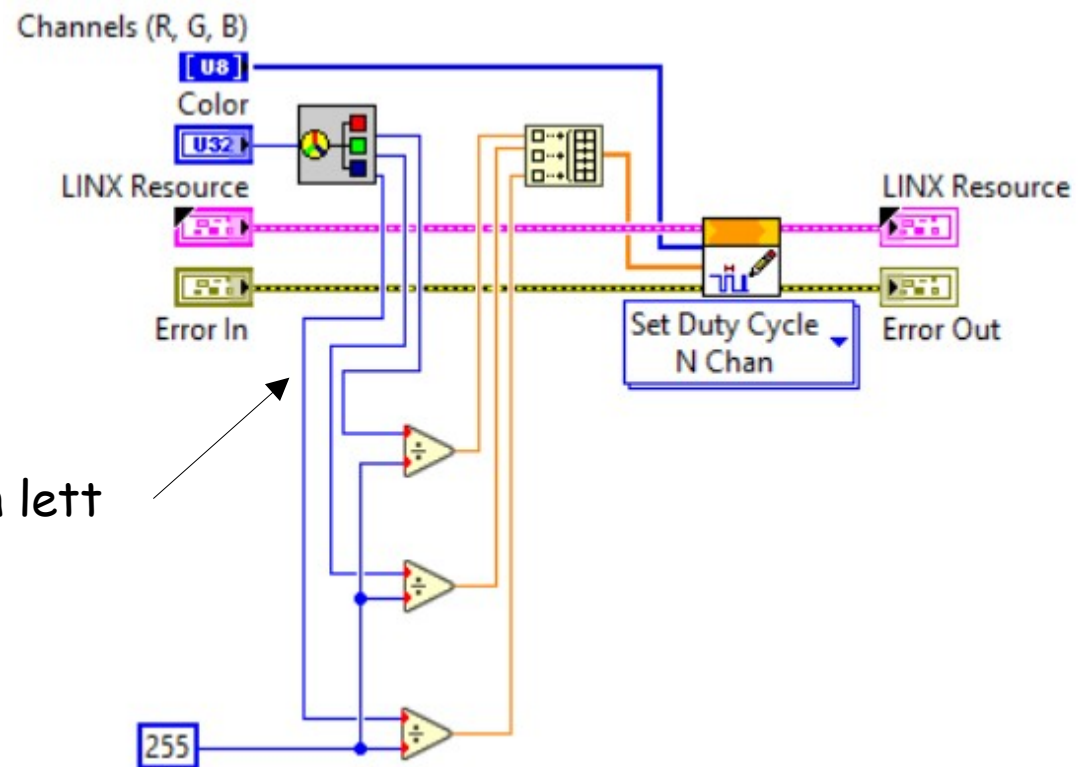
RGB LED.vi

- Apró probléma, hogy a PWM csatornák kitöltési tényezőjét 0..1 közé eső lebegőpontos számmal kell megadni, nem pedig 0..255 közötti egész számokkal, ahogy a programunk akarja...
- A LabVIEW `vi.lib\MakerHub\LINX\Public\Sensors\Lights\` mappájában nyissuk meg az **RGB LED.vi** állományt és javítsuk ki, majd mentjük el! (csak el kell osztani mindegyik színekomponenst 255-tel)



Ilyen volt

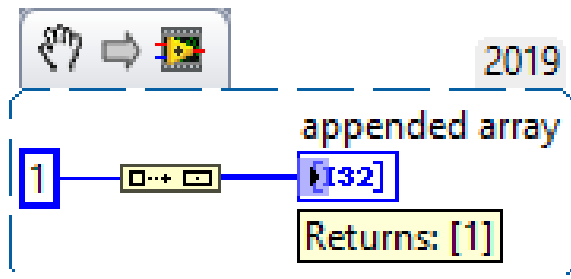
N db. PWM csatorna
kitöltésének beállítása



Ilyen lett

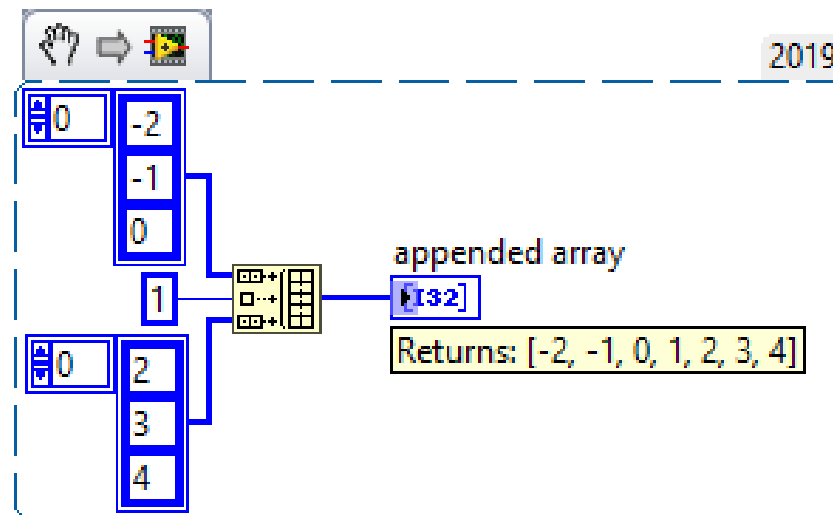
Mit csinál a Build Array elem?

- Skalár elemeket vagy tömböket fűz össze egy tömbbé
- Példa: egy elemű tömb létrehozása



A modult bal egérgombbal alul vagy felül lehet „megfogni” és kiterjeszteni

- Másik példa: két tömb és egy skalár összefűzése



LINX - Ultrasonic_modified.vi

- A LINX-Ultrasonic mintapéldát átdolgoztuk HC-SR04 szenzorhoz (az eredeti példa a GH311 szenzorhoz készült)

Ultrasonic Example (modified)

This example demonstrates how to measure distance using an HC-SR04 ultrasonic sensor.

Instructions

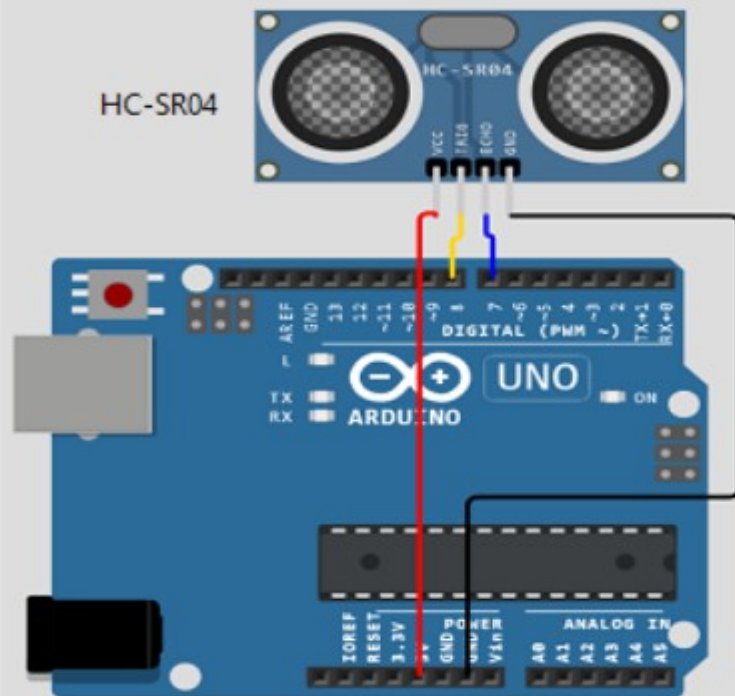
1. Select the **COM Port** associated with the LINX Device.
2. Select the **Echo Digital Input** channel connected to the ultrasonic sensor.
3. Select the **Trig Digital output** channel connected to the ultrasonic sensor.
4. Click the **Run Arrow**.

Loop Rate (Hz)

101

Stop

Circuit Schematic



LINX Device Settings

Serial Port

COM4

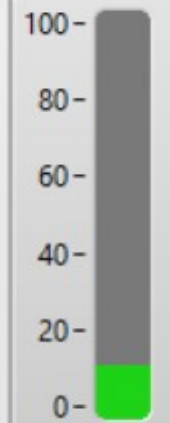
Trig pin

8

Echo pin

7

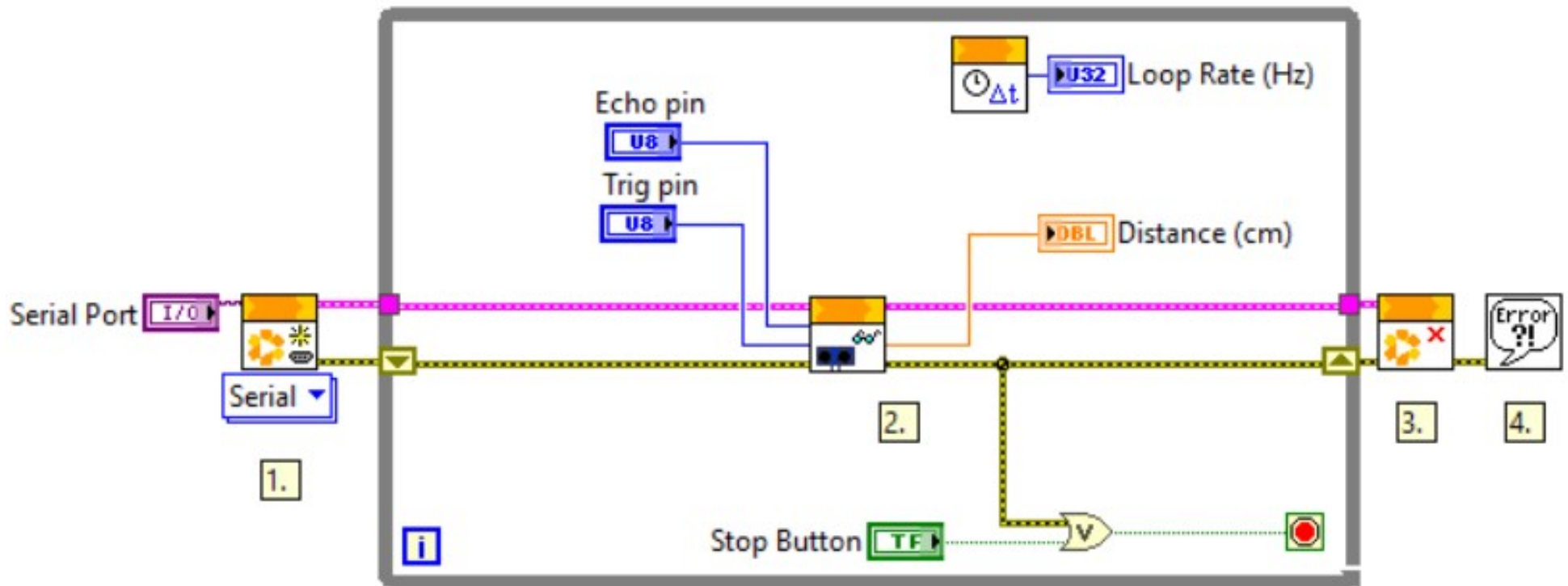
Distance (cm)



10,566

LINX - Ultrasonic_modified.vi

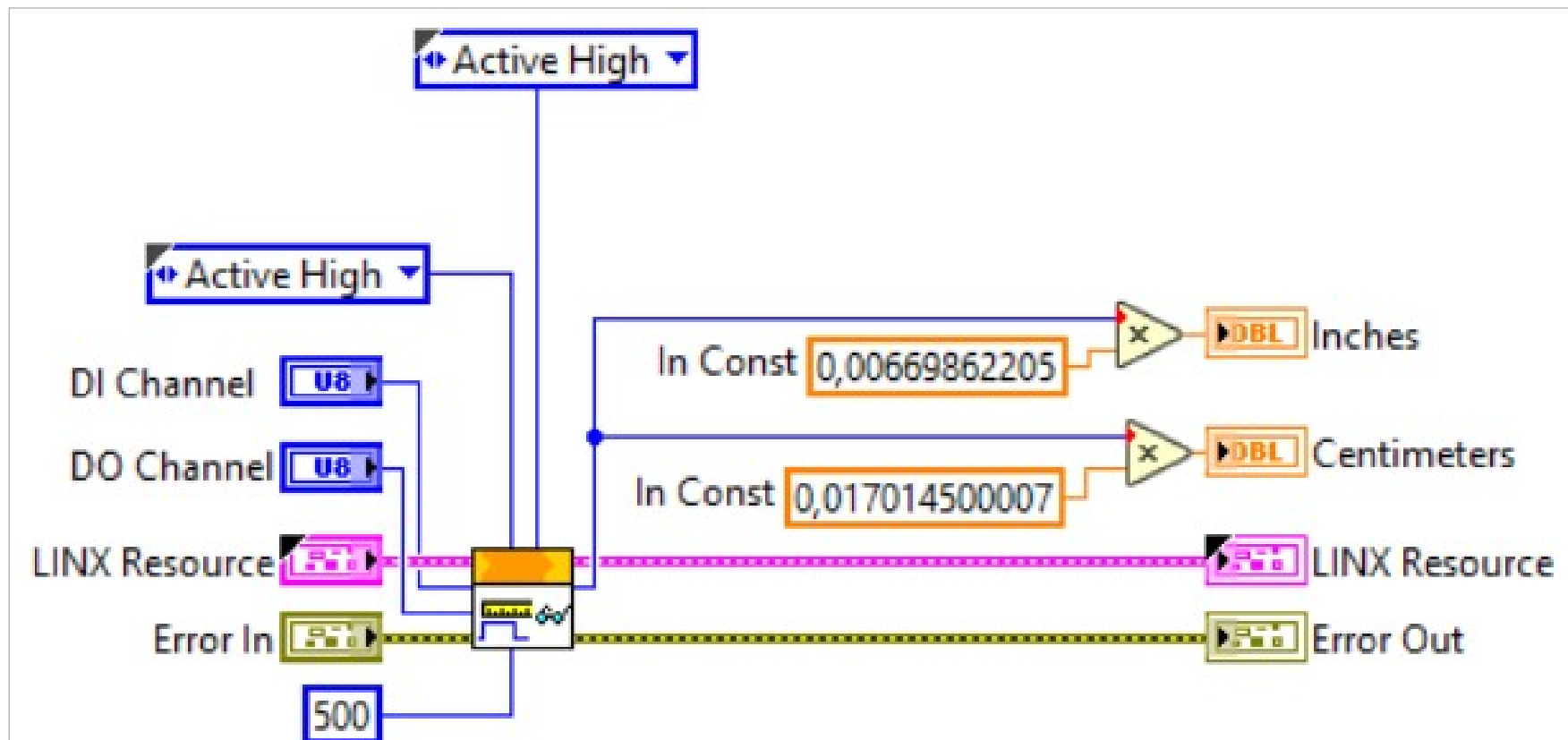
- A program blokkdiagramjában egy **Ultrasonic Read.vi** modult találunk, amelynek szétválasztott Echo és Trigger kivezetéseket kezel
- Az **Ultrasonic Read.vi** modult duplakattintással nyissuk meg!



1. Open a connection to the LINX device.
2. Read the value from the ultrasonic sensor connected to the specified digital input channel.
3. Close the connection to the LINX device.
4. Handle Errors

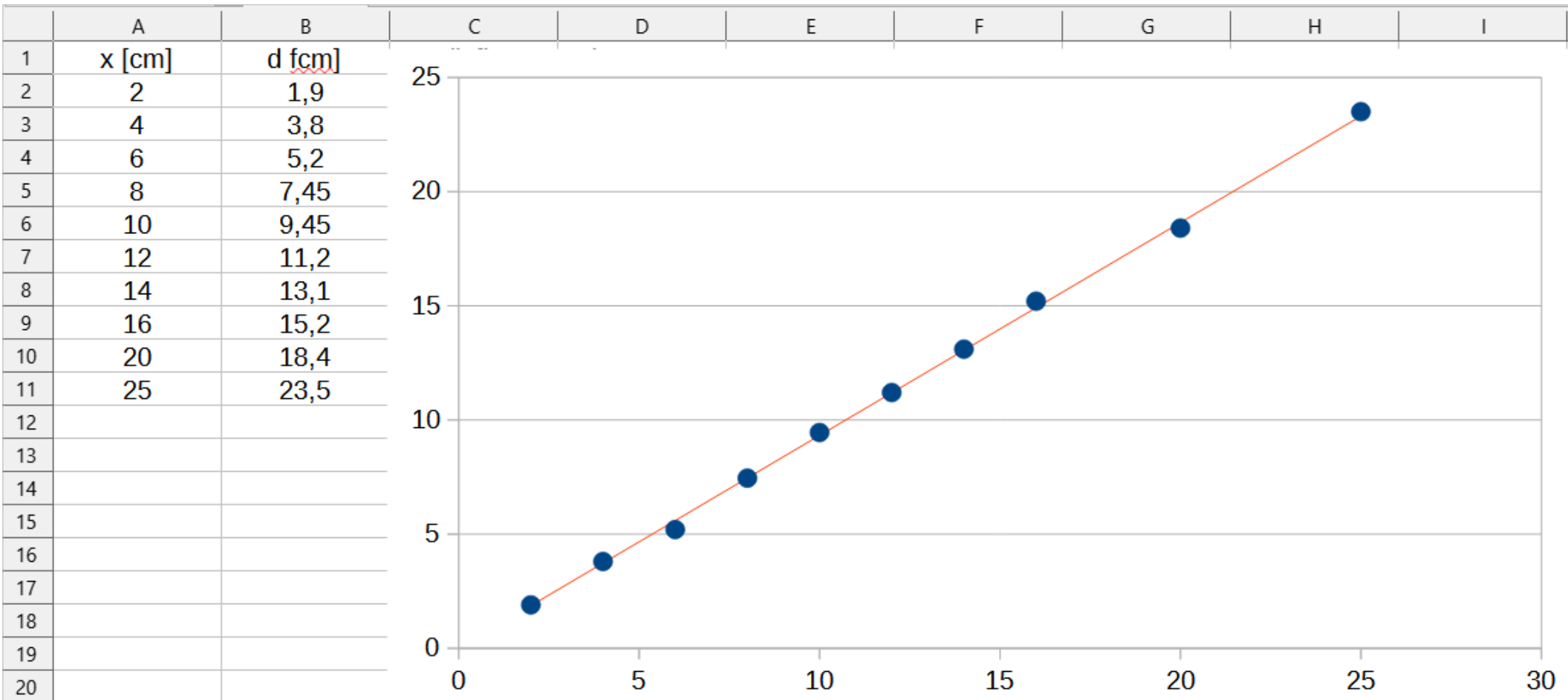
Az Ultrasonic Read.vi modul

- A DO csatorna küldi ki a Trig jelet, DI pedig az Echo jelet fogadja
- **Megjegyzés:** Fontos, hogy a triggerjel és a fogadott válaszjel is **Active High** módba legyen állítva!
- A válaszjelet inch-, ill. cm-re átszámítva kapjuk meg a kimeneteken



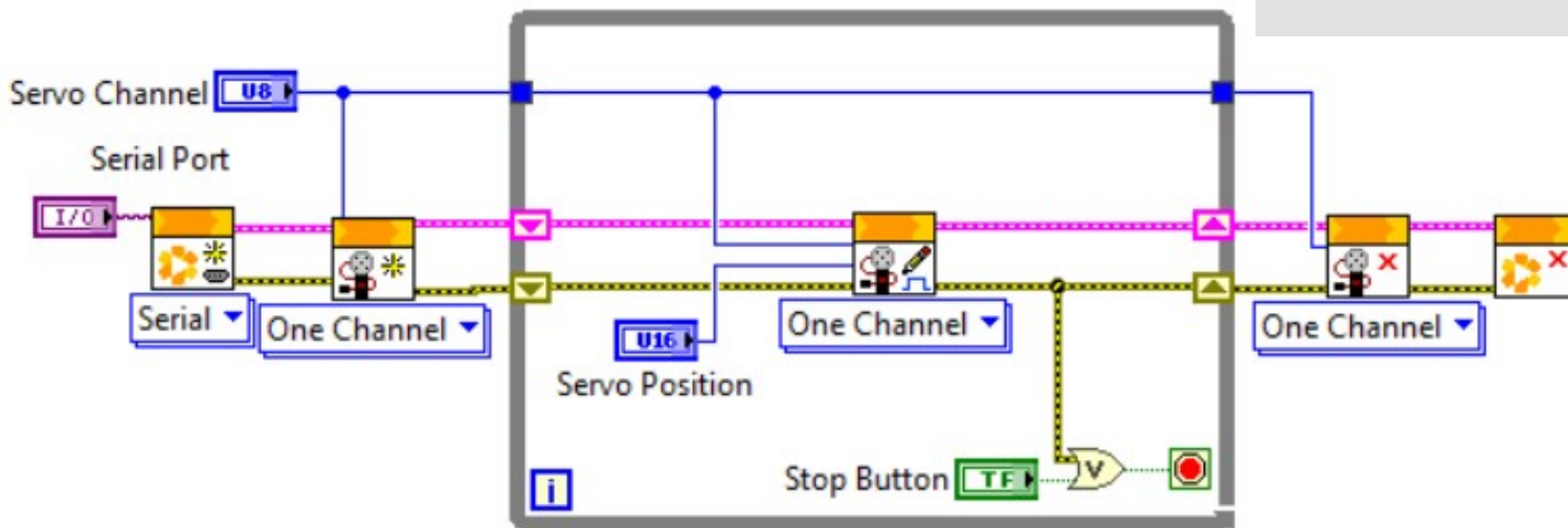
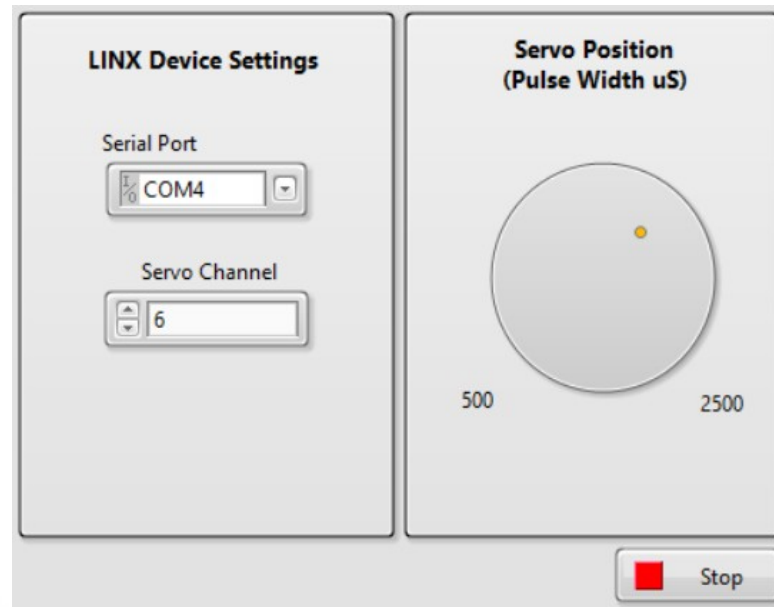
A távolságmérés kalibrálása

- Az **x** oszlop a vonalzóval mért, a **d** oszlop pedig az ultrahangos szenzorral mért távolság ($T \approx 25 \text{ }^\circ\text{C}$, rel. Hum. $\approx 44 \%$)



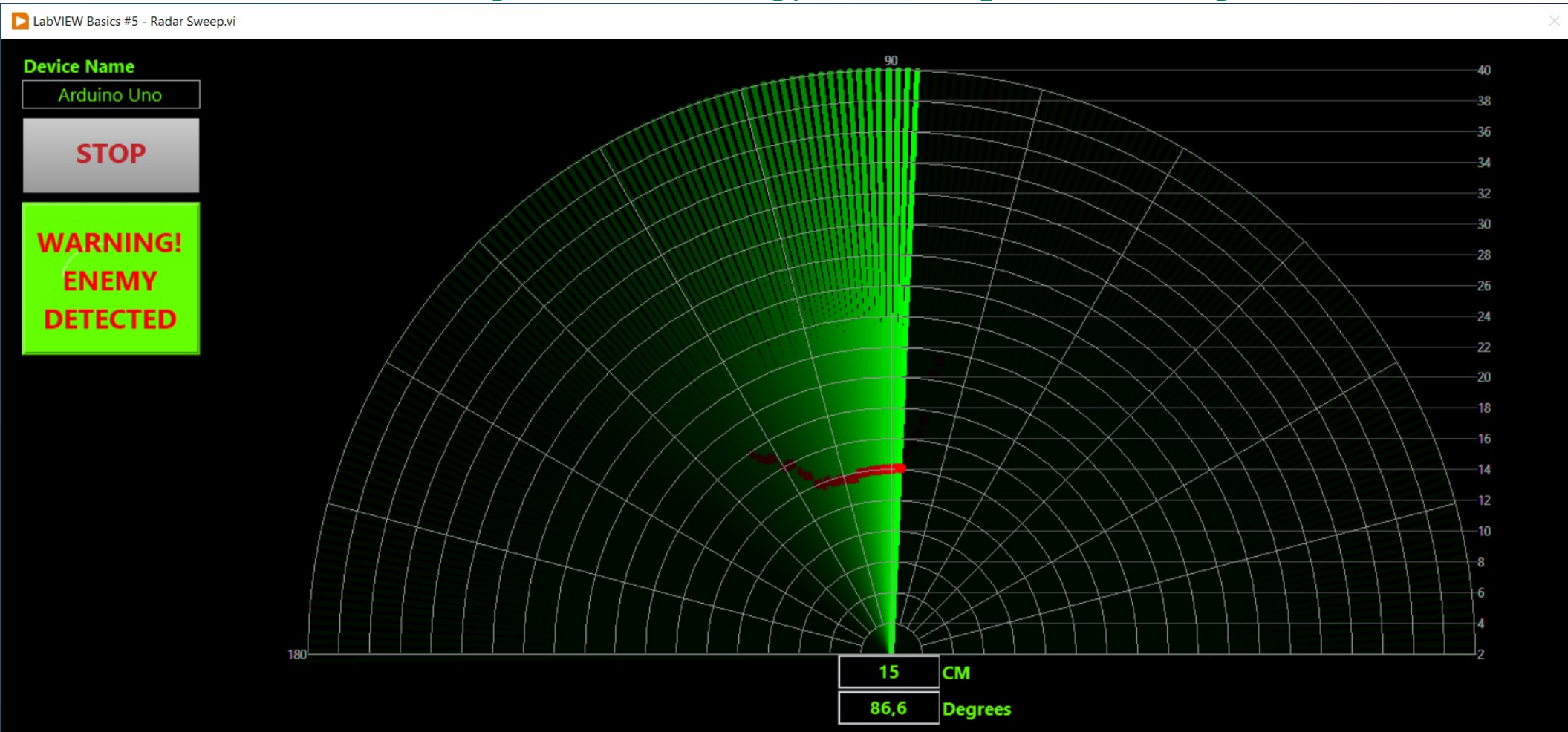
LINX – Servo 1 Channel.vi

- A PWM-hez hasonló a szervó motorok vezérlése, ennél is van lehetőség egy, illetve több csatorna együttes vezérlésére
- A LINX – Servo 1 Channel.vi nevű ”gyári” példaprogram az egycsatornás működést mutatja be



Ultrahangos „radar”

- Egy LabVIEW+Arduino mintaprogram gyűjteményben található ultrahangos szonár példaprogramot a LabVIEW Basics #6 - SONAR Part 3 videóban részletesen is ismertetik. A programban egy **HC-SR04** ultrahangos szenzor méri a távolságot, miközben egy szervó a pásztázást végzi



Ultrahangos „radar”

- A program blokkdiagramját megnyitva tudjuk beállítani a **COM** portot, az ultrahangos szenzor **Echo** és **Trig** lábaihoz rendelt kivezetéseket és a szervó vezérlésre szolgáló kimenetet (**Servo channel**)
- A programot itt teljes terjedelmében nem tudjuk bemutatni

