LabVIEW Real Time Test Environment for Education and Research Purpose

Imre Tóbel

evotech Ltd., Szabadság utca 117., Budaörs, 2040, Hungary, tobeli@evotech.hu

Abstract

In the field of control technology teaching and research, the development of test environments, which can be used to build and test a prototype of a complex control system, is very helpful. Simulation environments provide also a valuable solution for this. However, real-time interaction with the environment may also be a requirement for tests or studies for educational or research purposes. LabVIEW software and supported hardware components, as well as the LabVIEW Real-Time Module, make it easy to build a real-time test environment. This article presents a PC-based, cost-effective test environment that allows for real-time applications of laboratory-based controls for both educational and research purposes.

Keywords: RTOS; engineering education; test environment;

LabVIEW valósidejű tesztkörnyezet oktatási, kutatási célokra

Tóbel Imre

evotech Kft., Szabadság utca 117., Budaörs, 2040, Magyarország, tobeli@evotech.hu

Absztrakt

Irányítástechnika oktatásában, kutatási feladatok során elvégzendő vizsgálatok esetén nagy segítséget nyújt egy olyan fejlesztői tesztkörnyezet, amely segítségével akár egy komplex irányítórendszer prototípusa viszonylag egyszerű módon felépíthető és tesztelhető. A szimulációs környezetek erre adnak megoldást. Azonban az oktatási vagy kutatási célú tesztek, vizsgálatok során a környezettel történő valósidejű interakció megvalósítása is elvárás lehet. A LabVIEW szoftver és az általa támogatott hardver elemek valamint a LabVIEW Real-Time Module segítségével ilyen valósidejű tesztkörnyezet egyszerűen felépíthető. Jelen cikk egy PC alapú költséghatékony tesztkörnyezetet mutat be, mely segítségével valósidejű irányítások is vizsgálhatóak laboratóriumi körülmények között mind oktatási, mind pedig kutatási célokat figyelembe véve.

Kulcsszavak: valósidejű operációs rendszer; műszaki oktatás; tesztkörnyezet;

1. Bevezető

Az oktatásban és kutatási feladatokban a környezetből érkező információk feldolgozásának megértése, a jelenségek magyarázata szempontjából az egyes vizsgálatok elvégzését lehetővé tevő tesztkörnyezetek, mérő és elemzőrendszerek alkalmazása elengedhetetlen (Geda, 2011). A szimulációt a mérnöki gyakorlatban elterjedten használják ilyen vizsgálatok, elemzések,

fejlesztés, hangolás céljából számos alkalmazási területen, melyek egyes jelenségek pontosabb megértésével segíti ezen rendszerek továbbfejlesztését, viselkedésének pontosabb előrejelzését a műszaki gyakorlat eltérő területein (Kővári, 2009).

Manapság az ilyen jellegű tesztkörnyezetek összeállítása számítógép alapú rendszerek alkalmazásával valósul meg. A környezettel szoros kapcsolatban álló számítógép alapú rendszerek működésétől azt várjuk, hogy azok a bemeneti adatok beolvasása/események észlelése után, a számítógépen futó program segítségével feldolgozza azokat és a feldolgozás eredménye alapján meghatározza a szükséges kimeneteket, válaszokat. A környezetben lezajló folyamatok jeleinek változási sebességéhez, bekövetkezett események gyorsaságához kell illeszteni az az azzal kapcsolatban álló rendszer feldolgozási időzítését (File & Goedegebuure, 2003), hogy a folyamat változásaihoz képest a rendszer rövid időn belül/meghatározott időzítés szerint képes legyen a reagálásra. Az olyan számítógép alapú rendszereket, amelyek előírt időzítésnek megfelelően, a bemenet észlelésétől, annak feldolgozásától a kimenet előállításáig, valós idejű rendszereknek nevezzük (Kővári, 2010). Ezen rendszerek egyik elterjedt alkalmazása a HIL (hardwer-in-the-loop) (Grega, 1999) környezet építése, ahol a számítógép egy folyamat valós jeleit, működését képes szimulálni valós időben (Kővári, 2009). A HIL rendszerek fontos szerepet töltenek be a modern oktatási rendszerek területén is (Chen et al, 2017).

A cikkben a National Instruments LabVIEW fejlesztőkörnyezet által támogatott valósidejű rendszerek tesztelésére alkalmas személyi számítógép alapú rendszer kerül bemutatásra. A LabVIEW manapság már széles körűen alkalmazott fejlesztőkörnyezet, annak ellenére, hogy fejlesztése és alkalmazása eleinte elsősorban méréstechnikai területre irányult. Azonban manapság már számos alkalmazási területen, például szenzorhálózatokkal összefüggésben (Farkas et al, 2014), készítenek alkalmazásokat a LabVIEW által nyújtott grafikus programozás segítségével.

1.1. Valósidjű rendszerek

A valós idejű rendszerek működésében a reakcióidő nagyon fontos tényező. A valósidejű működés időzítésének feltételei az alábbiak lehetnek (Jancskárné, 2015):

- abszolút: amikor a feldolgozást egy meghatározott időpontokban/időpont körül/időpontig/időpont után kell végrehajtani;
- relatív: amikor a feldolgozást valamilyen esemény bekövetkezéséhez viszonyítva meghatározott időpontokban/időpont körül/időpontig/időpont után kell végrehajtani.

A valósidejű tesztkörnyezerekkel szemben általánosan támasztott követelmények az alábbiak szerint foglalhatók össze:

- előírt időzítés szerint működjön, vagyis minden feldolgozás, válasz determinisztikus legyen;
- egyidejű eseményeket is fel tudja dolgozni (párhuzamos feldolgozás);
- megbízható, biztonságos.

Mivel az általános célú operációs rendszerek különböző alkalmazások és folyamatok párhuzamos futtatására vannak optimalizálva, ezek jellemzően úgy működnek, hogy minden feladat kapjon valamennyi feldolgozási időt. Ennek eredményeképpen az alacsony prioritású feladatok is kapnak kisebb processzoridőt a futásra, vagyis megszakíthatják a magasabb prioritású alkalmazás futását. Ez biztosítja, hogy az alacsony prioritású feladatok is fussanak, ami nem mindig követi a programozó által meghatározott prioritások szerinti futtatást.

Ezzel ellentétben, a valós idejű operációs rendszerek sokkal pontosabban követik a programozók által meghatározott prioritásokat. A legtöbb valós idejű operációs rendszer, ha a magasabb prioritású feladat a processzor 100%-t használja, akkor nem fog futni az alacsonyabb prioritású feladat addig, ameddig a magasabb prioritású feladat be nem fejeződik. Ezért, a valós idejű rendszer tervezőinek kell ütemezni ezeket az alkalmazásokat, óvatosan figyelve arra, hogy a prioritások megfelelőek legyenek. Egy tipikus valós idejű alkalmazásban, a tervező időzítés alapján helyezi el a kritikus kódot (pl.: eseményjelző vagy vezérlő kód) igen magas prioritással. Egyéb kevésbé fontos kód, mint a lemezre történő naplózás vagy a hálózati kommunikáció alacsonyabb prioritású részben fut.

Az általános célú operációs rendszereknél a megszakítások kiszolgálása adott válasz válaszideje nem kritikus, addig a valós idejű rendszerek garantálják, hogy az összes megszakítást egy bizonyos maximális időn belül kiszolgálja.

Amennyiben olyan alkalmazást fejlesztése szükséges, melyek több feladatot hajtanak végre párhuzamosan, úgy az egyes szálak prioritásai beállíthatók.

Számítógép alapú folyamatirányító-rendszerek esetében például az ipari folyamatirányítás legáltalánosabb irányítóegysége, a PLC működése determinisztikus, ami azt garantálja, hogy a bemenetek beolvasása, a rajta futó program, a kimenetek beállítása meghatározott ciklusidőn belül megtörténik. Azonban a személyi számítógépeken futó operációs rendszerek általában nem determinisztikus működésűek, így ott a meghatározott időn belül elvégzett folyamatok nem garantálhatók. Azon rendszerek esetén, melyeknél a valósidejű működés

szükésges, ott valósidejű operációs rendszereket (RTOS: Real-Time Operating System) használnak, melyek kernele garantálja a determinisztikus működést. A valósidejű rendszerek két részre oszthatók, így megkülönböztetünk (National Instruments, 2015):

- „hard real-time”;
- „soft real-time” rendszereket.

A hard real-time rendszerek esetében feldolgozás során időtúllépés nem megengedett, ott a kritikus folyamatok adott idő alatt történő lefuttatása biztosított. Azonban a soft real-time rendszereknél a kritikus folyamatok csak magas prioritással futnak, ezért ezen megoldás esetén az időtúllépés meghatározott mértékben és gyakorisággal megengedett. Az időzítésben, vagyis egy folyamat lefutási idejében bekövetkező eltéréseket „Jitter”-nek nevezik, azonban a valósidejű operációs rendszerek arra vannak optimalizálva, hogy a „Jitter” jelenség minél kisebb legyen (National Instruments, 2018a).

A valósidejű operációs rendszerek esetén a valósidejű futtatás minél determinisztikusabbá tételéhez kiegészítő egységeket, funkciókat is tartalmaznak, mint például:

- watchdog - válaszidő figyelő, ha a programunk leáll, és így a watchdog számára adott időn belül választ nem tud küldeni, úgy a watchdog a rendszert automatikusan újraindítja;
- determinisztikus adat kommunikációt megvalósító programrészek és a valós idejű program részek között;
- több CPU maggal rendelkező rendszerek esetén segédprogramok az egyes CPU magok közötti terheléskiegyenlítés konfigurálásra;
- ciklus végrehajtásához szükséges időzítés szabályozása.

2. LabVIEW Real-Time module

A LabVIEW Real-Time modul a National Instruments cég által fejlesztett LabVIEW grafikus programozási környezet egy kiegészítő modulja. Segítségével a LabVIEW környezetben összeállított program valósidejű működést lehetővé tevő hardverre letölthető és futtatható. A LabVIEW Real-Time modul többféle valósidejű működést lehetővé tevő hardvert támogat, mint például a CompactRIO, CompactDAQ, PXI, képfeldolgozó rendszerek és a személyi számítógépek is. A LaBVIEW programozási környezetben elkészített program Ethernet kapcsolaton keresztül tölthető le a valósidejű futtatást lehetővé tevő hardverre, amit

LaBVIEW Real-Time Target-nak neveznek. A LabVIEW Real-Time Target-en futó valósidejű operációs rendszer biztosítja az alkalmazás futtatásának pontos időzítését.

2.1. Valósidejű futtatást lehetővé tevő támogatott hardverek

Az NI LabVIEW Real-Time modul többféle hardver eszközt támogat (National Instruments 2018b):

- PXI (kibővített PCI): egy szabványos ipari PXI platform ipari környezetre optimalizálva, valamint szinkronizálás céljából integrált időzítőt és trigger egységet tartalmaz, valamint beágyazott vezérlőt és I/O modulokat is;
- Ipari szabályzó: ipari kivitelű, viszonylag nagyteljesítményű processzor és egy PCI vagy PCIe foglalat is található I/O modulokkal történő bővíthetőség céljából;
- CompactRIO (kompakt újrakonfigurálható I/O): egy valósidejű programok futtatására alkalmas FPGA vezérlő mely I/O modulokat is tartalmaz. Az iparban alkalmazott PLC-hez hasonló kialakításúak, azonban nagyobb számítási teljesítményt tudnak.
- Single-Board RIO (Single-Board újrakonfigurálható I/O): egy áramköri lapon kialakított kompakt vezérlő egységek, melyek architektúrája megegyezik a CompactRIO rendszerrel.
- Real-Time Vision (intelligens kompakt kamera): ipari nagyteljesítményű képfeldolgozó szenzorokat tartalmaz és a valósidejű képfeldolgozási programok futtatására alkalmas.
- Ipari, asztali PC: akár általános célú ipari számítógép, vagy akár egy normál asztali számítógépet is használható, ha ezek az NI által meghatározott rendszerkövetelményeknek megfelelnek.

Laboratóriumi körülmények között oktatási és kutatási célokra nem feltétlenül van szükség az ipari környezetre optimalizált, robosztus házban elhelyezett és ennek megfelelően drága megoldások alkalmazására. Oktatási és kutatási célból a személyi számítógép alapú megoldások olyan költséghatékony alternatívát jelentenek, melyek a legtöbb esetben biztosítják a tesztkörnyezettől elvárt funkcionalitást költséghatékony megoldás mellett. Számítási teljesítmény szempontjából sem kell feltétlenül kompromisszumot kötni, mivel a személyi számítógépek terén elérhető erősebb processzorok is alkalmazhatók. A személyi számítógépek valósidejű hardverként történő alkalmazása esetén oda kell figyelni arra, hogy a LabVIEW Real-Time Module nem minden hardver eszközt támogat. A támogatott hardver

eszközökről részletes információ a National Instruments oldalán található (National Instruments, 2018c).

A költséghatékonyság, mint egyik fontos tényező számos más kutatásban is megjelenik. A Katona, J. et al (2016) cikkben szimulációs szoftvert alkalmaznak, elkerülve ezzel a robot megvásárlását, valamint a tesztkörnyezet fizikai megvalósítását, amíg Katona, J. et al (2016) tanulmányában egy olyan alacsony költségű mobilrobot megépítését ismerteti, amely további vizsgálatok és kutatások alapjául szolgálhat.

3. Valósidejű tesztkörnyezet megvalósítása

A LabVIEW-t futtató általános célú úgynevezett Host PC, amely az előzőek alapján lehet egy személyi számítógép is, Ethernet hálózaton keresztül kapcsolódik a valósidejű operációs rendszert futtató Target PC-hez. Amennyiben a megfelelő beállítások már megtörténtek, akkor a Host PC-n a LabVIEW fejlesztőkörnyezetben fejlesztett alkalmazás elindításakor az alkalmazás Etherneten kapcsolaton keresztül automatikusan áttöltődik a valós idejű hardverre, a Target PC-re, és ott a program elindul. A tesztkörnyezet előnye, hogy a LabVIEW hagyományos hibakeresési lehetőségei ebben az esetben is használhatók, mint például a „breakpoint”, annak ellenére is, hogy a program egy másik különálló valós idejű hardveren, a Target PC-n fut.

A tesztkörnyezet tehát a LabVIEW és a LabVIEW Real-Time Modul segítségével került megvalósításra és a rendszer két fő egységből áll:

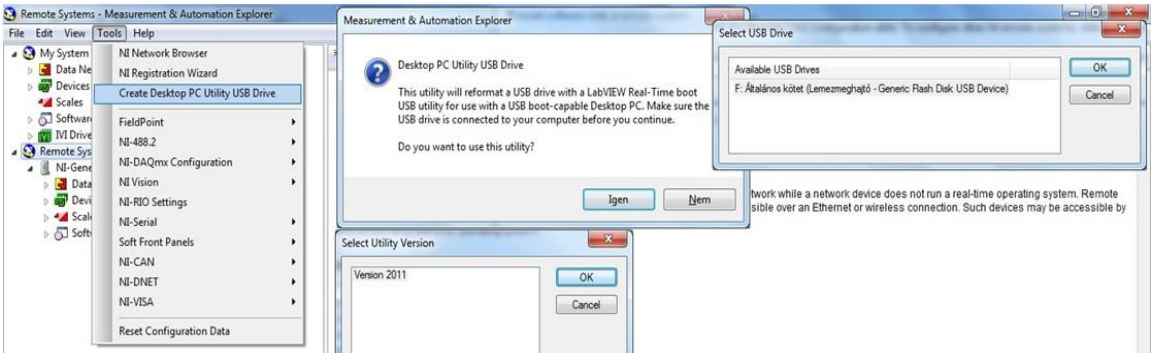
- Host PC: a szoftver fejlesztés és a futási eredmények megjelenítése, egy általános PC, vagy laptop;
- Target PC: a korábbiak alapján szintén lehet egy személyi számítógép amelyen valósidejű operációs rendszeren fut, és amely a be- és kimenetek illesztéséhez szükséges A/D, D/A átalakító egységet tartalmazza.

A Target PC és a Host PC közvetlenül Ethernet hálózaton keresztül kommunikál és azonos alhálózatban célszerű lenniük. A Target PC tartalmazza a külső jeleket fogadó, előállító A/D, D/A illesztőegységeket és a hozzájuk csatlakoztatható adaptereket. A Target PC-be egy a LabVIEW Real-Time Modul által támogatott hálózati kártya szükséges. A Target PC-re a LabVIEW Real-Time segítségével kell feltelepíteni a valósidejú működést biztosító operációs rendszert, melyre USB flash meghajtó is használható. A Hyper-Threading-et célszerű letiltani a BIOS-ban. A nem használt egységeket célszerű a BIOS-ban letiltani.

3.1. Host PC előkészítése

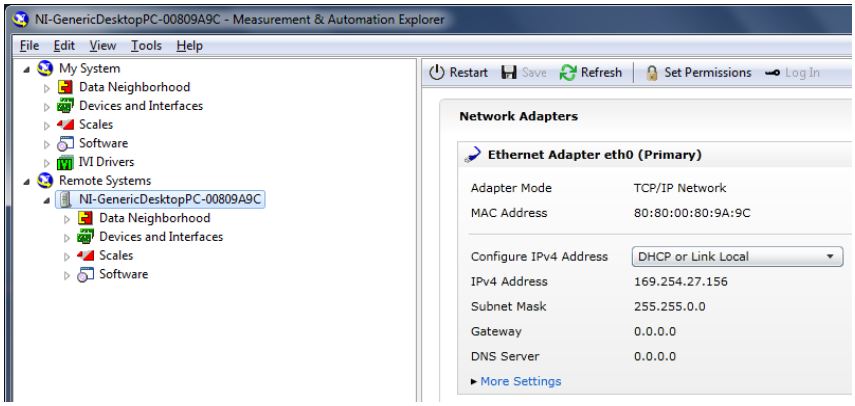
Első lépésben a Windows operációs rendszert futtató Host PC-re történt a LabVIEW programcsomag feltelepítése, mely magában foglalja a Measurement & Automation Explorer (MAX) alkalmazást, aminek segítségével telepíthetjük, illetve beállíthatjuk a Target PC-t.

A Target PC-n futó valósidejű futtatási környezet installálásához és indításához egy USB Drive szükséges, melynek létrehozását az 1. ábra mutatja.



1. ábra USD Drive létrehozása a Target PC telepítéséhez

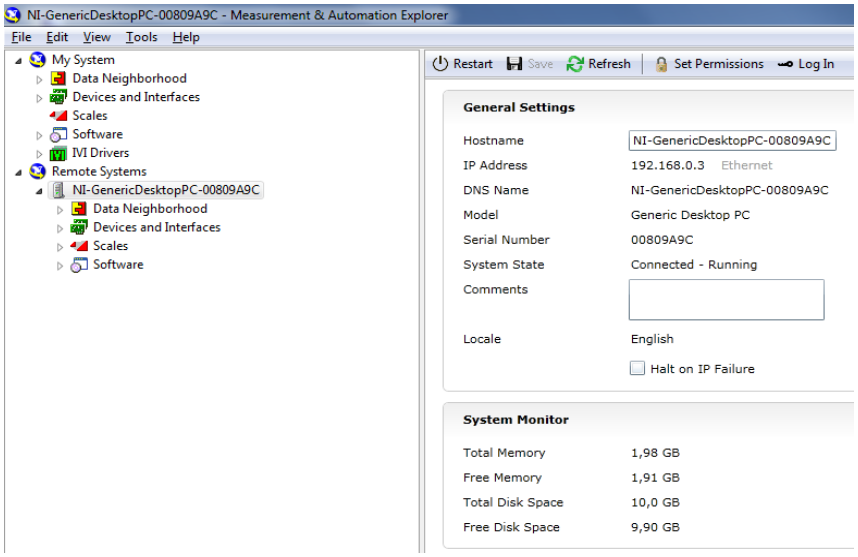
A Target PC első betöltése után a „Boot using software installed onthe hard disk” menüpontot választva a hálózati beállítások automatikusan megtörténtek. A Host PC-n DHCP szerver nem futott, ezért a Target PC automatikusan előre beállított IP címet nem kapott. Ahhoz, hogy a Target PC-hez a Host PC kapcsolódni tudjon, azonos alhálózatban kell lenniük ez a hálózati IP cím beállításokkal kell biztosítani. Amennyiben a kapcsolat létrejött, úgy a Host PC-n a MAX-ban a Remote Systems listában megjelenik a Target PC (2. ábra).



2. ábra Target PC beállítása a MAX-ban

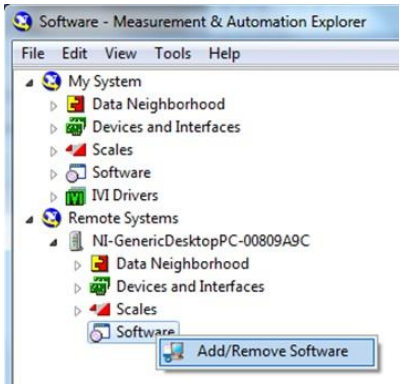
A Target PC számára 192.168.0.3 statikus IP cím, valamint 255.255.255.0 alhálózati maszk került beállításra (3. ábra). A beállításokat elmentve a Target PC újraindul. A Host PC IP

beállítása a 4.7. ábra szerint 192.168.0.1 IP címre és 255.255.255.0 alhálózati maszkra került módosításra, hogy ugyanabban az alhálózatban legyenek. Újraindítás után a Target PC állapota a Host PC-n a System Settings-ben lekérdezhető (3. ábra).



3. ábra Target PC lekérdezése

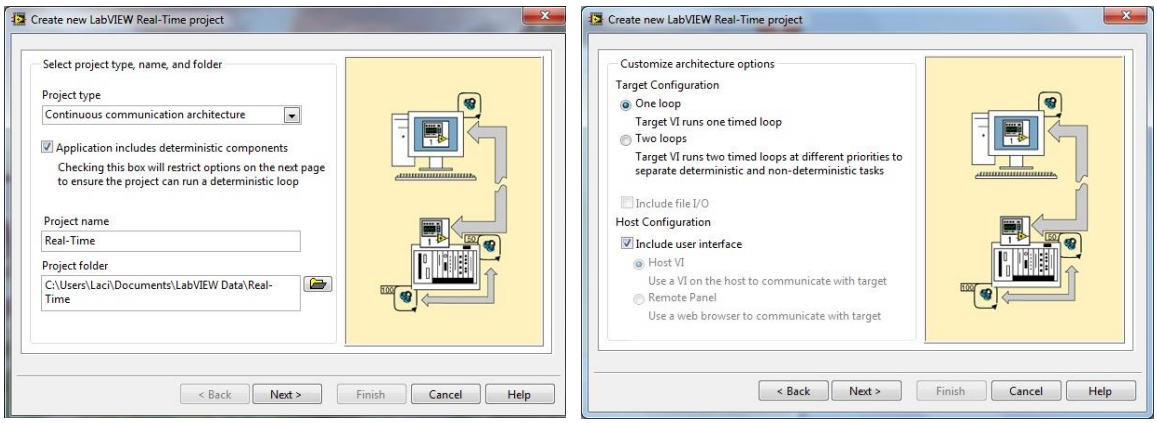
A Target PC számára a szükséges modulok a Software pontban kiválaszthatók és telepíthetők (4. ábra) a multifunkciós adatgyűjtő kártya és a hálózati kártyát tartalmazó illesztő programokkal együtt.



4. ábra Modulok telepítése a Target PC-re

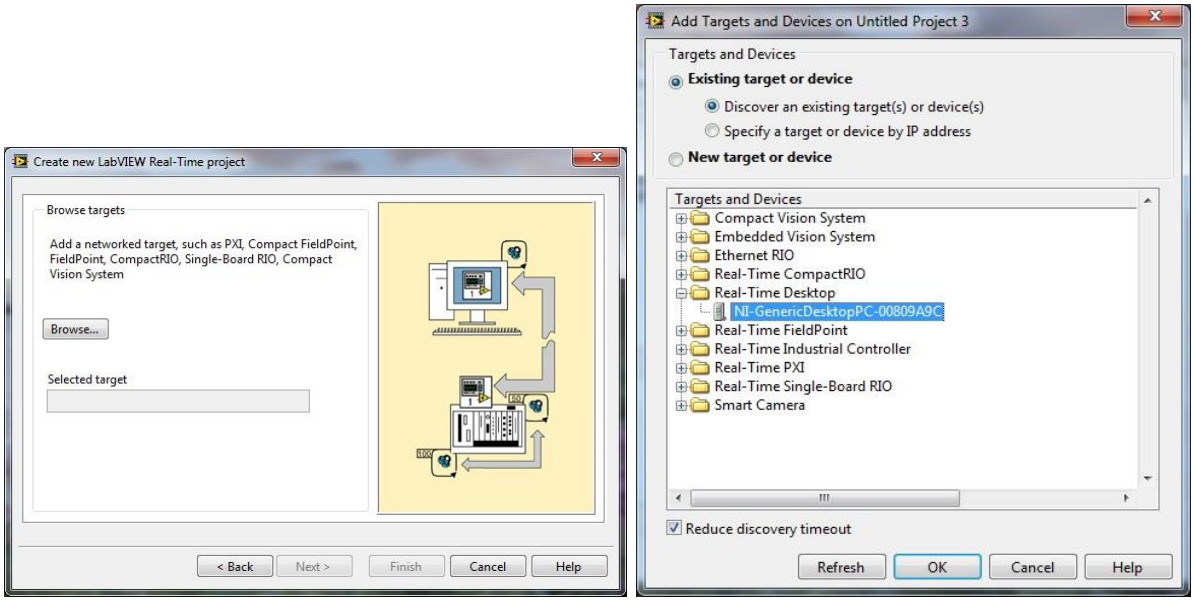
A modulok telepítése után a Target PC alkalmas arra, hogy Ethernet kapcsolaton a Host PC-ről programot tölthessünk rá és futtathassuk azt. Egy valósidejű alkalmazás elkészítéséhez és letöltéséhez első lépésben létre kell hozni egy valósidejű projektet, melyet a LabVIEW indítása után választhatunk ki. A következő lépés a projekt típusának megadása, valamint

hogy egy hurok vagy két időzített hurok fusson (5. ábra). A Target PC jeleinek vizsgálatára a Host PC-t használjuk, ezért a felhasználói felület a Host PC-n kerül kialakításra.



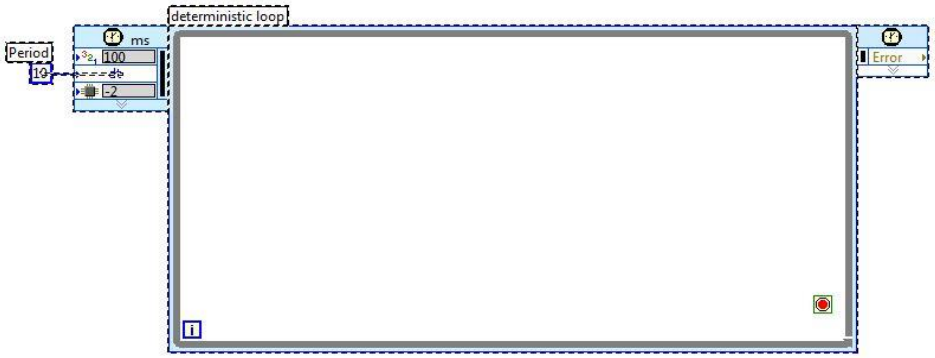
5. ábra Real-Time projekt paramétereinek magadása

A Target hozzáadása során ki kell választani a használt számítógépet, ami a Target PC lesz (6. ábra).



6. ábra Target PC megadása

Amennyiben ez siekresen megtörtént, úgy a LabVIEW automatikusan elkészíti a futtatási környezet létrehozásához szükséges file-okat. A 7. ábrán látható LabVIEW programrészlet a létrehozott valósidejű futtatást biztosító determinisztikus hurok blokkját mutatja.



7. ábra Valósidejű futtatást lehetővé tevő determinisztikus hurok

A determinisztikus hurokban megvalósított alkalmazást a Target PC-re letöltve, azt a valósidejű operációs rendszer valósidejű alkalmazásként fogja futtatni. LabVIEW programozási ismeretek birtokában összetett valósidejű alkalmazások is egyszerűen megvalósíthatók.

4. Összefoglalás

A cikk egy valósidejű rendszerek tesztelésére, vizsgálatára alkalmas költséghatékony környezetet mutat be, mely nagy segítséget nyújthat mind például a jelek és rendszerek, mérés és irányitástechnikai vagy egyéb ehhez kapcsolódó ismeretkörök jelenségeinek megismerésére, tanulmányozására (Ponce et al, 2012). Az oktatásban elterjedten szimulációkat alkalmaznak egyes jelenségek megfigyelésére, bizony paraméterek megváltozására bekövetkező hatások elemzésére, azonban a valós időben is megfigyelhető jelenségek életközelibb tapasztalatokat adhatnak a hallgatóknak (Wicks, 2009). Kutatási célokból, laboratóriumi körülmények között elvégzett vizsgálatok, tesztek esetében sem feltétlenül szükségesek igen drága, az ipari környezet hatásainak is ellenálló, robosztus egységek használata, ezeken a területeken is előnyös lehet a bemutatott személyi számítógép alapú valósidejű tesztkörnyezet kialakítása.

A bemutatott tesztkörnyezet különösen alkalmas lehet a valósidejű jelenségek vizsgálatára elemzésére, bemutatására, szemléltetésére, mely sokkal eredményesebb lehet a tanulás hatékonysága terén. A tesztkörnyezet olyan állapotok vizsgálatára is alkalmas, melyek egy valós rendszer esetében nem, vagy nehezen kivitelezhetők a berendezés túlterhelése, meghibásodása nélkül. Ezért a tesztkörnyezet által lehetővé tett jelenségek vizsgálata olyan

tapasztalati tudáshoz segítheti a tanulókat, melyek más módszerekkel nem vagy csak részben vizsgálhatók.

Irodalomjegyzék

Chen, Z., Chen, W., Liu, X., & Song, C. (2017). Development of an educational interactive hardware-in-the-loop missile guidance system simulator. Computer Applications in Engineering Education, 26(2), 341-355.

Farkas I. et al (2014). Wireless Sensor Network Protocol Developed for Microcontroller-based Wireless Sensor Units, and Data Processing with Visualization by LabVIEW. In Proceedings of the IEEE 12th International Symposium on Applied Machine Intelligence and Informatics, 95–98.

File, J., & Goedegebuure, L. (2003). Real-time systems: Reflections on higher education in the Czech Republic, Hungary, Poland and Slovenia. VUTIUM Press, Brno University of Technology.

Geda Gábor (2011). Modellezés és szimuláció az oktatásban. Educatio Kht.

Grega, W. (1999). Hardware-in-the-loop simulation and its application in control education. In FIE'99 Frontiers in Education. 29th Annual Frontiers in Education Conference. Designing the Future of Science and Engineering Education. Conference Proceedings (IEEE Cat. No. 99CH37011, Vol. 2, 12B6-7.

Jancskárné Anweiler Ildikó (2015). Számítógépvezérelt irányítások, Pécsi Tudományegyetem, Műszaki és Informatikai Kar, jegyzet.

Katona J. et al (2016). Speed control of Festo Robotino mobile robot using NeuroSky MindWave EEG headset based brain-computer interface. In *2016 7th IEEE international conference on cognitive infocommunications (CogInfoCom)*, 251-256.

Katona J. et al (2016). Cost-effective wifi controlled mobile robot. In *11th International Symposium on Applied Informatics and Related Areas (AIS 2016)* (pp. 28-31).

Kővári A. (2010). Real-Time Modeling of an Electro-hydraulic Servo System. In Computational Intelligence in Engineering, 301–311.

Kővári A (2009). Influence of cylinder leakage on dynamic behavior of electrohydraulic servo system. In SISY 2009 - 7th International Symposium on Intelligent Systems and Informatics 375–379.

Kővári A (2009). Hybrid Current Control Algorithm for Voltage Source Inverters. In 1st IEEE Eastern European Conference on the Engineering of Computer Based Systems, 65–70.

Kővári A (2009). Hardwer-in-the-Loop Testing of an Electrohydraulic Servo System. In 10th International Symposium of Hungarian Researchers on Computational Intelligence and Informatics, 631–642.

National Instruments (2018.01.15 a). What is a Real-Time Operating System (RTOS)? [Online] http://www.ni.com/white-paper/3938/en/

National Instruments (2018.01.15 b). Embedded Systems Hardware. [Online] http://www.ni.com/embedded-systems/products/hardware/

National Instruments (2018.01.15 c). Requirements for Desktop PCs as LabVIEW Real-Time Targets. [Online] http://www.ni.com/product-documentation/8239/en/

Ponce, P., Pacas, M., & Molina, A. (2012). Real time systems for teaching induction motor drives. In 2012 6th IEEE International Conference on E-Learning in Industrial Electronics (ICELIE), 65-73.

Wicks, C. (2009). Lessons learned: teaching real-time signal processing [DSP Education]. IEEE Signal Processing Magazine, 26(6), 181-185.