



Debreceni Egyetem

Műszaki Kar

Villamosmérnöki és Mechatronikai Tanszék

Orvosbiológiai érzékelők alkalmazása mobil eszközön

Készítette: **Máté Balázs mechatronikai mérnök hallgató**

Belső konzulens: Dr. Tóth János

Tanszékvezető: Dr. Husi Géza

Debrecen

2014. november

Tartalomjegyzék

Bevezetés.....	3
1. A felhasznált Vernier eszközök.....	4
1.1. Kézi erőmérő.....	4
1.2. Spirométer.....	5
1.3. Oxigénérzékelő.....	6
1.4. Felületi hőmérsékletérzékelő.....	6
2. A Mindstorms NXT bemutatása.....	8
3. A mérőrendszer felépítése.....	10
3.1. NXT – PC kommunikáció.....	10
4. A programozáshoz felhasznált NXT blokkok áttekintése.....	12
5. A kézi erőmérő programja.....	16
5.1. Slave program.....	16
5.2. Master program.....	18
6. A spirométerhez írt program.....	23
6.1. Slave program.....	23
6.2. Master program.....	26
7. Az oxigénszenzor programja.....	28
7.1. Slave program.....	28
7.2. Master program.....	30
8. A felületi hőszenzorra írt programok.....	32
8.1. Slave program.....	32
8.2. Master program.....	34
9. Futtatható állományok készítése.....	37
10. Fejlesztési lehetőségek.....	38
11. Összegzés.....	39
Irodalomjegyzék.....	40
Ábrajegyzék.....	42

BEVEZETÉS

Az orvostudomány napjainkban az egyik legprofesszionálisabb és legpontosabb mérőrendszereket felvonultató ágazat. A pontosság és a megbízhatóság mellett hasonlóan fontos szempont az eszközök hordozhatósága is. A mérésadatgyűjtő rendszerek, a vezérlés-, szabályozástechnika fejlődése valamint a modern robotok megjelenése lehetőséget nyújt komplex, mobil rendszerek létrehozásához. Ezekkel az eszközökkel orvosi jelenlét, beavatkozás nélkül is fel lehet az egészséget, fizikai állapotot térképezni.

Az automatizált mérőrendszerek képesek információt nyújtani a vizsgált személyről, a megszerzett információkat tárolni, a letárolt adatokat diagnosztizálni és az eredményeket továbbítani. Ezáltal áthidalhatóak a térbeli, környezeti, időbeli akadályok, kiszélesítve ezzel az egészségfelmérés eszközeit. Az orvostechnikai mobileszközök segítségével növelni lehet a rugalmasságot és a hatékonyságot.

Célom egy olyan mérőrendszer létrehozása, amely alkalmas különböző külső szenzorok befogadására, melyek segítségével különböző fizikai állapotokat lehet megmérni, a méréseket letárolni, valamint a tárolt adatokat összehasonlítani referenciaértékekkel. Ezzel lehetőség nyílik ugyanannak a személynek a fizikai állapotának hosszú távú megfigyelésére, sőt akár több személy állapotának összehasonlítására is.

A mérőrendszer alapját az NI ELVIS II platform jelentette, majd az ott elért eredményeimet a LEGO MINDSTORMS NXT mobil eszközön kívántam megvalósítani. Az eddig elért sikereket felhasználva a fő célom a vezetékek nélküli kommunikáció megoldása mobil készülék segítségével, amely képes a mérőeszköz irányítására, valamint adatmentésére is.

1. A FELHASZNÁLT VERNIER ESZKÖZÖK

A Vernier egy 1981-ben alapított cég egyik fő profilja az érzékelők gyártása. Több mint 50 különböző szenzorból álló termékpalettájukból a Medical Engineering Kit tartalmazhat. Ez az oktatási célú csomag a Debreceni Egyetem Műszaki Kar hallgatói számára önálló labortevékenység végzése céljából rendelkezésre áll.

A TDK dolgozatomhoz ezeket az eszközöket használtam fel:

- Kézi erőmérő,
- Spirométer,
- Oxigénérzékelő,
- Felületi hőmérsékletérzékelő.

Az erőmérő és a spirométer az orvosdiagnosztikában is elterjedt eszközök, amelyek segítségével szakaszos mérésre van lehetőség, azaz nem szolgáltatnak folyamatosan adatokat, hanem egy előre meghatározott futási idő alatt mérnek, és ez alapján állhat elő egy mérési eredmény, „diagnózis”.

Ezzel szemben az oxigén és a hőmérsékletérzékelőt úgy használtam fel, hogy a folyamatos működést helyeztem előtérbe.

1.1.KÉZI ERŐMÉRŐ

A kézi erőmérő egy nyúlásmérő-bélyegen alapuló érzékelő. Alkalmazható markolás és ujjszorítás (csípés) erejének meghatározására, valamint az izomfáradás tanulmányozására is.



1. ábra Kézi erőmérő [1]

A külső mechanikai erő által a bélyegen bekövetkező alakváltozás, azaz keresztmetszetcsökkenés következtében nő az ellenállása. Az erőmérő egy Wheatstone hídkapcsoláson keresztül ezt erősíti és feszültséggé alakítja, ami a kimeneten mérhető. Kalibrálási beállítástól függően Newtonban, kg-ban és fontban tudja megjeleníteni a mért értékeket.

Pontossága (x) $\pm 0,6$ N és a maximális felbontása 13 bit (n). A Newtonba való átváltáshoz 175,416-szoros erősítés (A) és -19,295 offset szükséges. Ennek megfelelően a legkisebb (dF_{min}) eltérés, amit érzékel:

$$dF_{min} = \frac{U \times A}{2^n} = 0,1071 \text{ N} \quad (1)$$

A maximális tartomány, amiben képes mérni az eszköz 0-850 N, de csak 0-600 N tartományban alkalmas üzemszerű működésre. [2]

1.2.SPIROMÉTER

A spirométer egy légzést elemző szenzor, amely a belélegzett és kifújott levegő áramlását méri liter/sec-ben. Alkalmazható légzési mintázatok tanulmányozására, tüdőkapacitás mérésére és terheléses vizsgálatokra (FEV,VC) is.

A Piston Medical szerint: „A spirométer a funkcionális légzésdiagnosztika legfontosabb eszköze, nélkülözhetetlen a légzőrendszeri megbetegedések korai felismerésében.”



2. ábra Spirométer [3]

A „fej” középső részén membránként elhelyezett háló miatt, erőhatás (jelen esetben fújás és szívás) miatt nyomáskülönbség alakul ki a két oldalon. Mindkét oldalon egy-egy cső az ottani nyomásértéket elvezeti egy differenciál-nyomásérzékelőhöz, ami piezorezisztív módon feszültséget állít elő.

A kimenő feszültségből meghatározott kalibrálás után liter/sec-os értékek állnak elő. Mivel az eszköz jelleggörbéje lineáris, ezért az $y = mx + b$ alakú függvény alapján beállítható a kívánt léptékezés. A leírásban szereplő értékek alapján:

$$Y \left(\frac{L}{s} \right) = 7,1869 \frac{\left(\frac{L}{s} \right)}{V} * x(V) - 17,9672 \left(\frac{L}{s} \right) \quad (2)$$

Az eszköz működéséhez 5 V DC feszültség (U) szükséges. A névleges kimenet 60 $\mu\text{V}/(\text{L/s})$ és $\pm 10 \text{ L/s}$ a mérési tartománya. [4]

1.3.OXIGÉNÉRZÉKELŐ

Az oxigénérzékelő képes mérni az oxigénszintet a különböző biológiai és kémiai vizsgálatokban, de akár a szabad levegőn is. Alkalmazható az emberi légzés vizsgálatára, valamint egy szoba levegő-minőségének a meghatározására is.



3. ábra Oxigénérzékelő [5]

Ez az eszköz 0 és 27 % között képes mérni az O_2 szintet, egy elektrokémiai cella segítségével. A cella ólom anódot és arany katódot tartalmaz elektrolitba merítve. Az érkező oxigén molekulák reakcióba kerülnek az arany katóddal, ez pedig áramot generál, aminek a nagysága arányos az oxigén koncentrációjával. Ebből az áramból létrejövő arányos feszültséget lehet megmérni a kimeneten. [6]

A 0-27 % mérési tartományon, normál légköri nyomáson $\pm 1 \%$ a pontosság. Felbontása 12 bit, ami 0,01 %-os lépéseket jelent. Légköri nyomásváltozás esetében szükséges kalibrálni a szenzort a következő összefüggés alapján:

$$V_{out} = V_{out}(standard) \times \left(\frac{P}{1013} \right) \quad (3)$$

1.4.FELÜLETI HŐMÉRSÉKLETÉRZÉKELŐ

Ez a hőszensor kis hőmennyiségek érzékelésére, valamint flexibilis mérésekre lett tervezve. Egy olyan termisztort tartalmaz, ami miatt kifejezetten gyorsan reagál a hőingásokra. Felhasználási módjai között szerepel pl. bőrfelület mérése, légzési hőmérsékletmérés, hőtranszfer mérés.



4. ábra Hőmérsékletérzékelő [7]

Ez a szonda egy 20k Ω -os NTC termisztort használ. A termisztor miatt a hőmérséklete és az ellenállás változás között nem lineáris a kapcsolat. A legjobb illeszkedést a Steinhart-Hart (4) összefüggéssel lehet elérni.

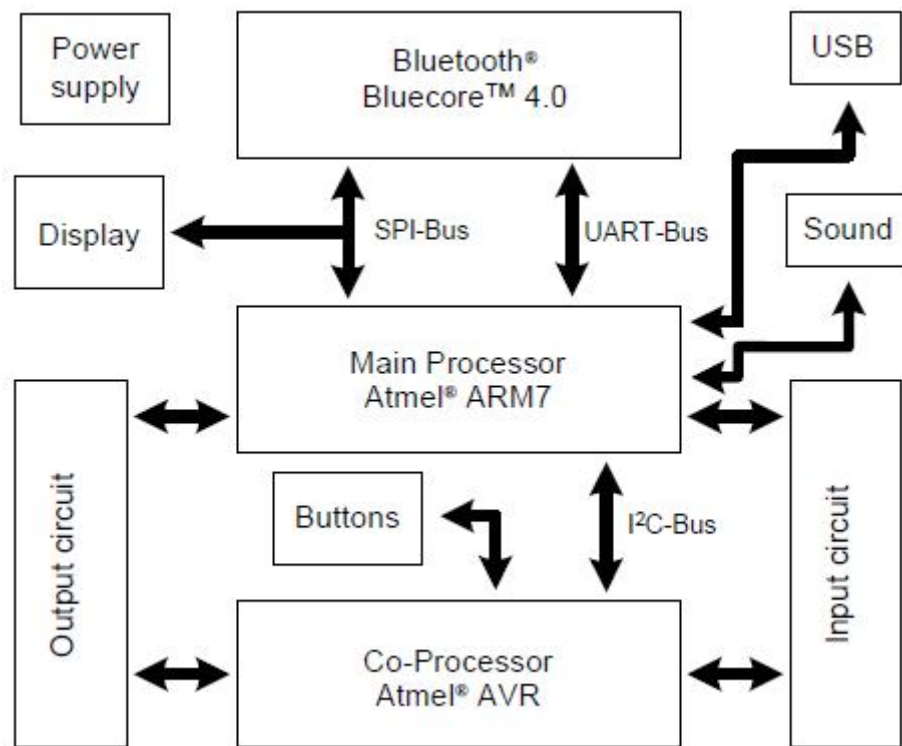
$$T = [K_0 + K_1(\ln 1000R) + K_2(\ln 1000R)^3]^{-1} - 273,15 \quad (4)$$

Itt a hőmérsékletet (T) Celsius fokban van megadva, R a mért ellenállás k Ω -ban, $K_0=1,02119 \cdot 10^{-3}$, $K_1=2,22468 \cdot 10^{-4}$, $K_2=1,33342 \cdot 10^{-7}$ értékű konstansok.

Az eszköz 150 °C-ig képes mérni anélkül, hogy károsodna, a mérési tartománya -25 °C és +125°C között van. Pontossága $\pm 0,2$ °C fagypontnál és $\pm 0,5$ °C forráspontnál (víz). Válaszideje 50 másodperc szélcsendben, 20 másodperc szellőben, tehát ez egy lassú szenzor. [8]

2. A MINDSTORMS NXT BEMUTATÁSA

Az NXT a LEGO és a National Instruments (NI) cégek által oktatási célra szánt eszköz, amit többféle szenzorral és aktuátorral lehet felszerelni, valamint az összeépített LEGO elemekből megvalósított berendezés programozás segítségével intelligenciával ruházható fel.



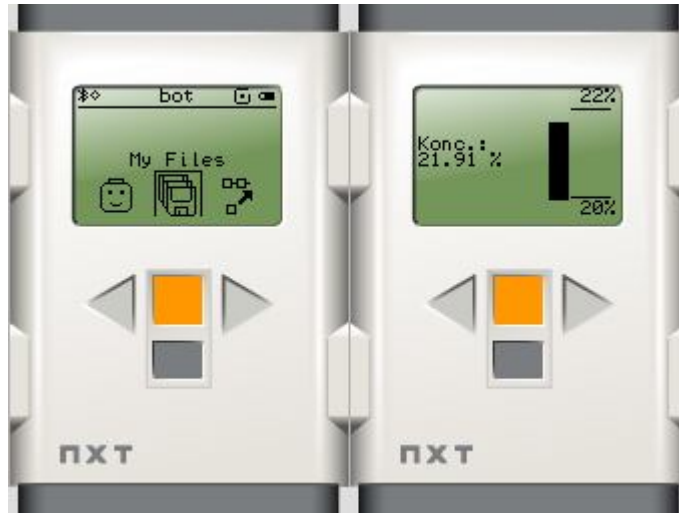
5. ábra NXT belső felépítése [9]

A központi feldolgozó egység egy 32 bites ARM processzor, ami 48 MHz-en üzemel és 64 kB RAM-mal, 256 kB FLASH-sel rendelkezik. Ez egy 8 bites AVR segédvezérlővel van kiegészítve, órajele 8 MHz, 512 bájt RAM-ot és 4 kB FLASH-t tartalmaz. Ezt a koncepciót gyakran használják kisméretű eszközökben a megfelelő gyorsaság érdekében.

A CPU-hoz SPI és UART buszon keresztül csatlakozik a Bluetooth egység, ami egy CSR Bluecore 4.0 chip egy külső 8 Mbytes FLASH-el. Az adatküldés az UART-on keresztül történik, a kommunikációs sebesség 460,8 Kbit/s.

A kijelző egy pontmátrix LCD kijelző, felbontása 100*64 pixel. A kijelzővezérlő egy UltraChip 1601, ami SPI interfészen keresztül kapcsolódik a CPU-hoz. Az SPI sebessége 2 MHz, amivel 17 ms-os képalkotási idő érhető el.

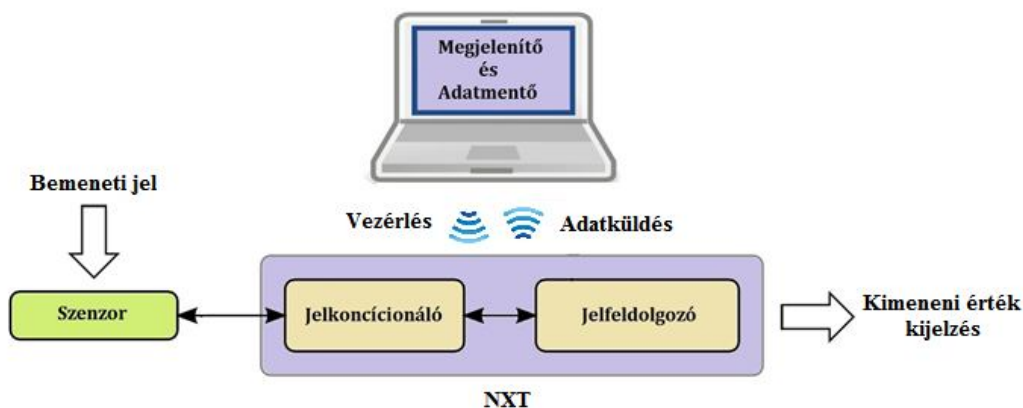
Bemeneti portokból 4 darab található (1-2-3-4), melyek analóg és digitális szenzorok fogadására is alkalmasak. A 4-es bemeneti port RS485-ös vezérlőhöz kapcsolódik, ami biztosítja a nagysebességű kommunikációt. Kimeneti portokból 3 darab van, amikhez külső eszközöket, például motorokat lehet csatlakoztatni. [9]



6. ábra Az NXT képernyője bekapcsoláskor (bal) és egy saját program futásakor (jobb)

3. A MÉRŐRENDSZER FELÉPÍTÉSE

A szenzorok jó működéséhez és a kapott eredmények megjelenítéséhez, adatmentéséhez saját fejlesztésű mérőprogramot kellett megvalósítanom.



7. ábra A mérőrendszer felépítése

A szenzorok BTA csatlakozóval szereltek, így az NXT-vel közvetlen nem kapcsolható össze. Erre a LEGO által forgalmazott speciális Vernier szenzorokhoz való adaptert használok.

NXT SENSOR
ADAPTER

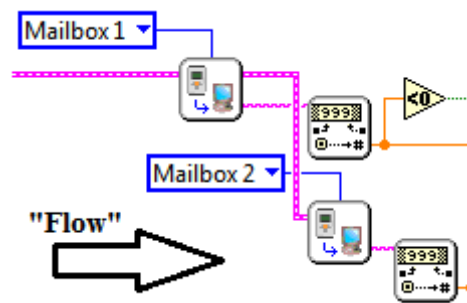


8. ábra Vernier szenzor adapter [10]

3.1. NXT – PC KOMMUNIKÁCIÓ

A számítógép és NXT közötti kommunikációhoz és programozáshoz a LabVIEW szoftvert használtam fel, mivel tanulmányaim alatt többször is találkoztam ezzel a fejlesztő környezettel. A LabVIEW szoftver NI magas szintű, grafikus programozói környezete. Segítségével úgynevezett virtuális műszereket (VI - Virtual Instrument) lehet fejleszteni. Minden VI két részből áll: egy előlapi panelből, ahol a bemenetek, kimenetek és grafikus elemek helyezhetők el és egy blokkdiagramból, amelyen a különböző elemek közti kapcsolatok, matematikai, logikai műveletek, ciklusok stb. építhetők fel. A programozás érdekessége, hogy az úgynevezett dataflow elvet követi,

azaz egy csomópont (node) akkor hajtódik végre, amikor az összes (kötelező) bemenetére adat érkezik. A végrehajtás után az eredmény továbbhalad a következő csomópont bemenetéig. [11]



9. ábra Általam készített blokkdiagram részlete

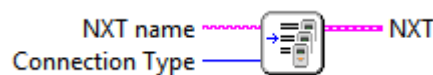
Az általam használt LabVIEW for LEGO MINDSTORMS 2010 egy teljes értékű LabVIEW környezet, amibe integrálva vannak az NXT-hez speciálisan megírt blokkok. Képes a számítógépen megírt programokat gombnyomásra áttölteni az NXT-re, külön projektkezelő része van és több beépülő applikációt is tartalmaz, amelyek különböző célfeladatokra használhatók, például:

- NXT Terminal – Firmware frissítés és fájlok kezelése,
- NXT Data Viewer – Az NXT egyedi formátumú adatfájljainak olvasására szolgál,
- NXT Remote Display – Élőkép az NXT kijelzőjéről, vezérlési lehetőség,
- NXT Sensor Viewer – Egyszerű interfész az érzékelők gyorsstesztjéhez.

A LEGO MINDSTORMS 2010 szoftverrel létre lehet hozni számítógépre írt Master VI-t és NXT-re írt Slave VI-t is. Míg a Masterben megírt program a gépen fut, a Slave az NXT-n. Így lehetőség nyílik a PC-ről történő irányításra, illetve a kétirányú adatcserére. További előnye hogy a Slave programot nem szükséges túlbonyolítani, mivel a főszámítást végezheti a számítógép. Hasonlóan nem szükséges az adatokat letárolni az NXT szegényes méretű memóriájába, azok egyből küldhetők a számítógépre és ott tárolhatóak. Ezzel tehermentesíthető az NXT-t és elkerülhetőek a túlterhelésből adódó akadályok, hibák és a tárhely túlszordulása.

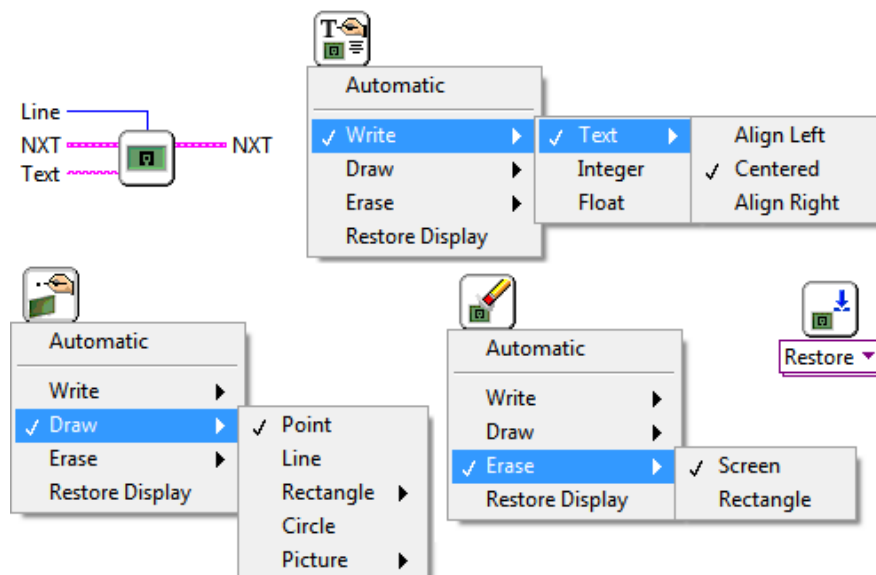
4. A PROGRAMOZÁSHOZ FELHASZNÁLT NXT BLOKKOK ÁTTEKINTÉSE

A megírt programjaimban sok helyen felhasználtam a beépített, NXT specifikus subVI-okat amiket a következőkben be fogok mutatni. Ezek a subVI-ok olyan előre megírt programrészletek, mint a C-ben vagy Pascalban a függvények, tehát a bemeneti változók (függvényargumentumok) alapján kimeneti értékkel szolgál. A LabVIEW subVI-ok egyik fő tulajdonsága, hogy több kimeneti node-ja lehet. [12]



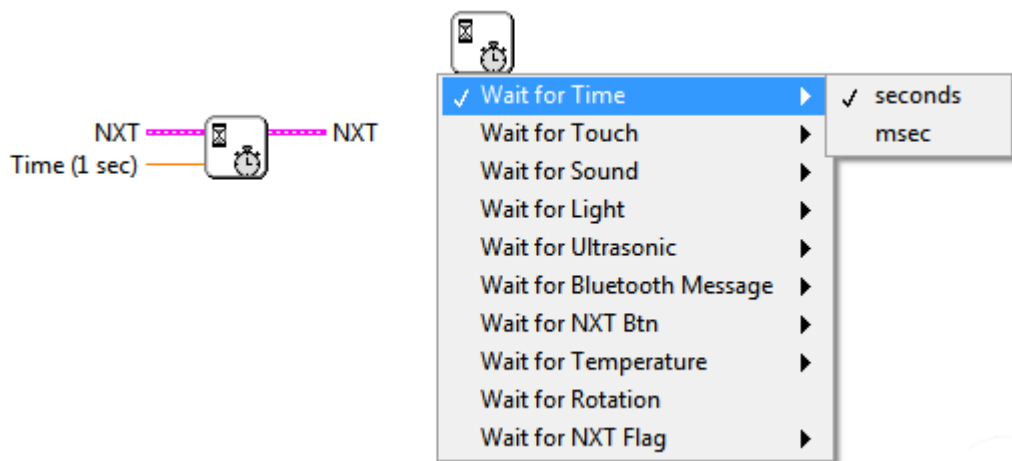
10. ábra Specify NXT blokk

A **Specify NXT** blokk kapcsolatot teremt egy NXT eszközzel USB-n vagy Bluetooth-on keresztül. Két bemenete van, az NXT name string típusú, a céleszköz nevét kéri amivel kommunikálni akar. A Connection type a kapcsolat típusa, ami kétféle lehet, USB illetve Bluetooth. Az általam megírt programokban a „bot” nevet használtam valamint először az USB-t a stabilitás és energiatakarékosság céljából, majd később a mobilitás miatt a Bluetooth-t. Egyetlen kimenete van, az NXT, ami a programbeli működésért felelős. Csak hasonló blokkok NXT termináljához csatlakoztatható, így a dataflow elvnek köszönhetően egyfajta szekvenciát valósít meg.



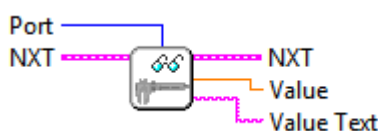
11. ábra Display blokkcsoport

A **Display Control** blokk többféle feladatra használható. Beállítható hogy írjon a kijelzőre szöveget, egész, vagy tizedes számot. Ilyenkor 3 bemenete van az NXT, a kiírandó adat (Text, Number), valamint a Line, ami a sort adja meg (1-től 8-ig) az eszköz kijelzőjén. Beállítható az is hogy rajzoljon a kijelzőre, ilyenkor az adatbemenet helyett koordinátá(ka)t kell megadni. Az Erase-t kiválasztva törölhető a kijelző egy része, vagy a teljes tartalma, a Restore-ral pedig visszaállítható eredeti állapotába a kijelző. Az itt felsorolt parancsoknak a kimenete az NXT.



12. ábra Wait for blokkcsoport

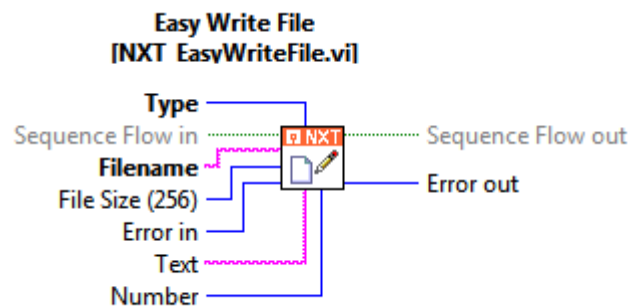
A **Wait for** is sok lehetőséget tartalmaz. A várakozás végét jelentheti az, hogy az előre megadott idő lejár, meghatározott porton keresztül a szenzor által érzékelt jel meghalad egy küszöbértéket vagy a felhasználó megnyomott egy gombot, stb. Programjaim írása során a Wait for Time-ot használtam, ahol bemenetként meg kell adni az időt, valamint a Wait for NXT Btn-t, ahol a bemenet az NXT egyik gombja. A kimenet itt egységesen az NXT.



13. ábra Vernier Read blokk

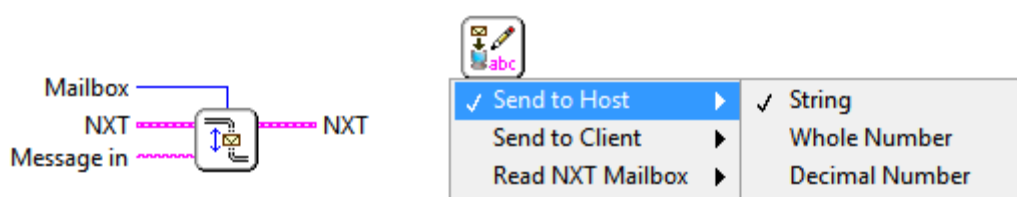
A **Vernier Read** azt a célt szolgálja, hogy a csatlakoztatott Vernier szenzorok által mért értékeket beolvassa. Több mint 30 féle szenzort támogat az NXT, ezeket nem minden esetben szükséges kalibrálni. Két bemenete van, a már ismert NXT és a Port ahol az NXT négy bemeneti csatornája közül kell választani. Első kimenete az NXT, a második

a Value, amin keresztül lebegőpontos számként, a harmadik a Value Text, ahol szöveggént küldi tovább az adatokat.



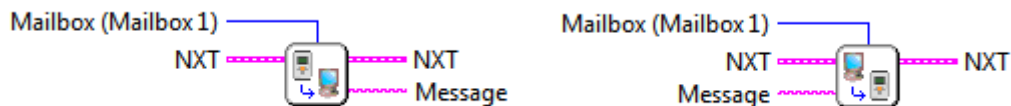
14. ábra Easy Write File natív blokk

Az **Easy Write File** egy natív I/O blokk, ami egy másik lehetőség az NXT programozására, jól használhatóak – bizonyos korlátok között – az alap NXT eszköztárral. Ezzel a blokkal az NXT memóriájában lehet létrehozni egy fájlt, majd beleírni és ezután írni lehet bele, valamint egy meglévő fájlhoz lehet írni további adatokat. Ha az előre megadott méretet túllépi, automatikusan növekszik a fájl mérete. Type bemenetén lehet megadni, hogy számot vagy szöveget tartalmazzon, Filename-nél a nevet lehet megadni meg (itt célszerű a kiterjesztésével együtt beírni), Text és Number bemenetei a fájlba írt adatokat jelentik. Error in/out node-ok a fájlírási és fájlkezelési hibákra vonatkoznak. Sequence Flow in/out node-ok hasonlóan az NXT node-hoz, a program futását szabályozzák, sajnos ezek nem kompatibilisek egymással.



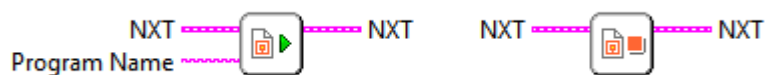
15. ábra Communication Control blokk

A **Communication Control** segítségével adatokat tudunk küldeni más eszközre, illetve olvasni lehet az NXT „postaládáját” ahová a máshonnan küldött adatok érkeznek. A Host, ami létrehozza a kapcsolatot, lehet NXT és PC is a „fogadó” Client viszont csak NXT lehet. A bemenetek NXT, Message, ami a kimenő/érkező üzenet és a Mailbox azaz az érkező vagy kimenő adatok postaládája, ebből összesen 10 van.



16. ábra Read from NXT és Write to NXT blokkok

A fenti két blokk a Direct Only csoportba tartozik, ami azt jelenti, hogy csak direkt vezérlésnél alkalmazhatók. A direkt vezérlés esetén a számítógépről futó programmal lehet kezelni és irányítani az NXT-t, mind USB mind Bluetooth kommunikáción keresztül. Ez esetben szükséges használni a fentebb említett Specify NXT blokkot. A baloldali blokk a **Read from NXT Mailbox**, ezzel lehet a számítógépre küldött adatokat olvasni. A jobboldali, **Write to NXT Mailbox** blokkal a számítógépről lehet egy string típusú adatot az NXT-re küldeni. Mindkettő esetén hasonló ki- és bemenetek vannak, de olvasásnál a Message output oldalon, írásnál input oldalon van értelemszerűen.



17. ábra Run és Stop Program blokkok

Szintén a Direct Only csoportba tartozik a **Run Program** (bal), amivel egy NXT-n levő programot lehet elindítani, és a **Stop Program** (jobb), amivel az éppen futó NXT programot lehet megállítani.

5. A KÉZI ERŐMÉRŐ PROGRAMJA

Ebben a programban egy Bluetooth kommunikációt valósítottam meg az NXT és a számítógép között. Az NXT-hez hozzákapcsoltam a kézi erőmérőt és az azzal mért értékeket folyamatosan a számítógépre küldtem, ahol tárolásra kerültek. Biztonsági okokból az aktuális mérés eredményét lementem az NXT memóriájába egy txt fájlba, de a belső tárhely mérete miatt, a következő mérés megkezdése előtt törlésre kerül.

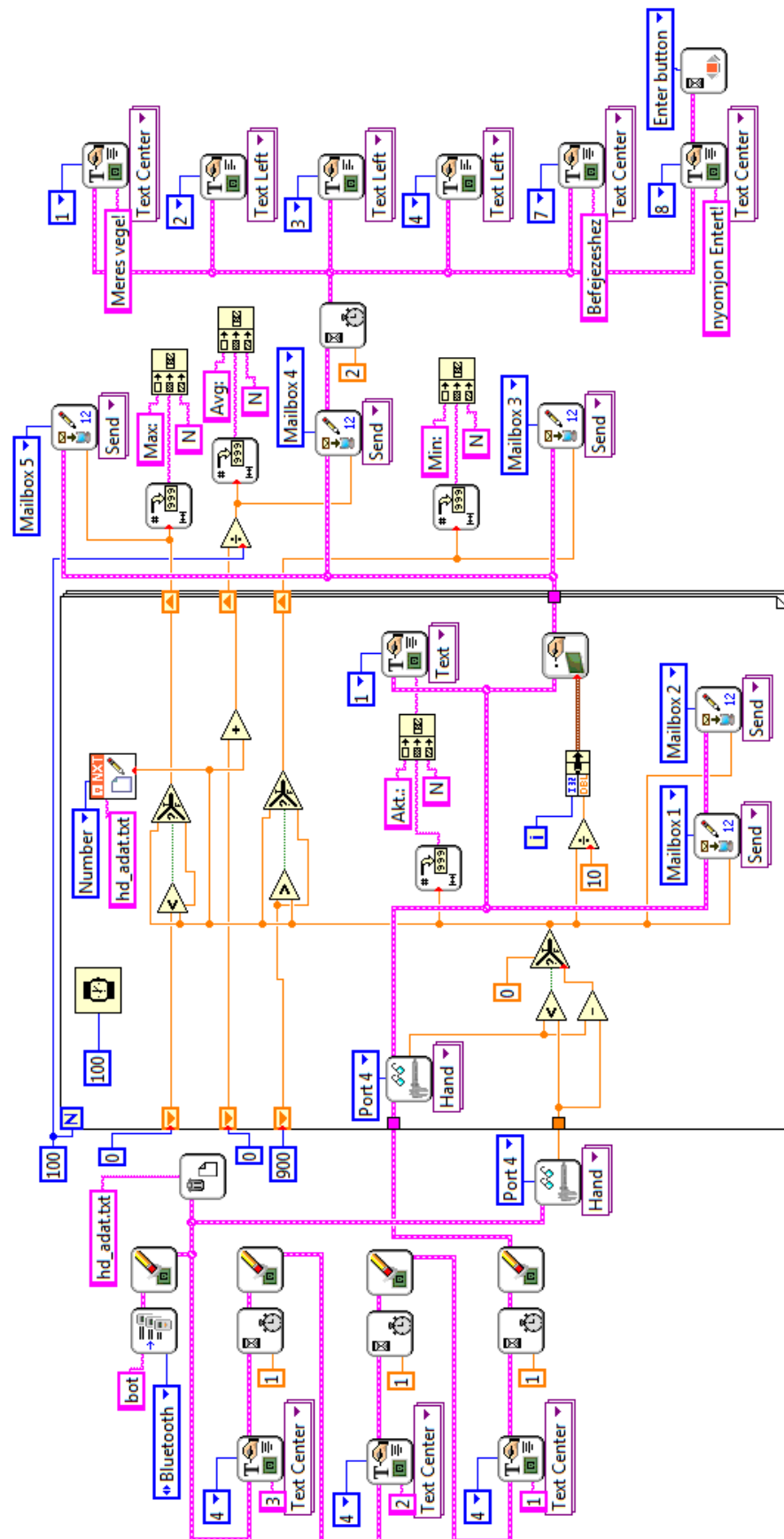
A kézi erőmérés elsődleges feladata az izomzat diagnosztizálása, de felhasználható pl. a fiatalok – idősok közötti differencia meghatározása. Ezen kívül különböző érdekes összefüggések, pl. a tenyérméret és a szorítás erőssége közötti párhuzam vizsgálata is lehetséges.

5.1.SLAVE PROGRAM

Az indításnál visszaszámol a program, közben sípszóval figyelmeztet a kezdésre. Mérés közben a látható az aktuális mért erőt valamint annak változását grafikus formában. A mérés végeztével az NXT tájékoztat a maximális, minimális és átlagos erő értékekről és rövid szöveges formában értékeli a teljesítményt:

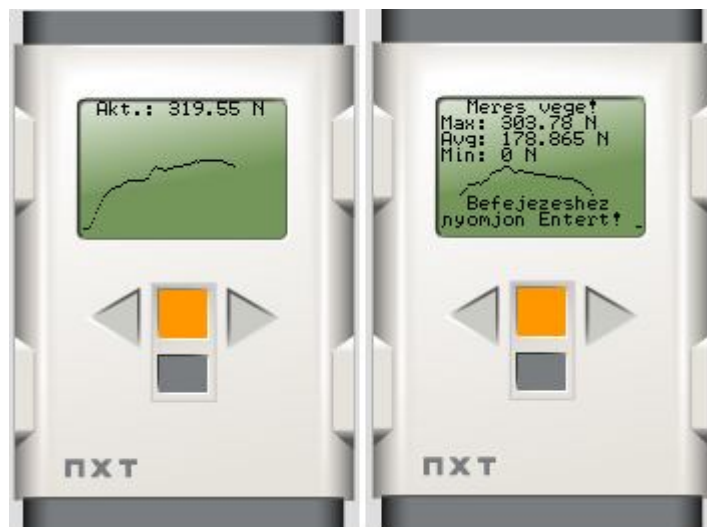
- 200 N alatti átlagérték – gyenge teljesítmény
- 200 és 400 N közötti átlagérték – átlagos teljesítmény
- 400 N feletti átlagérték – erős teljesítmény

Ezt a skálát egy 1986-os American Journal of Occupation Therapy cikk alapján határoztam meg. [13]



18. ábra Kézi erőmérő - slave program

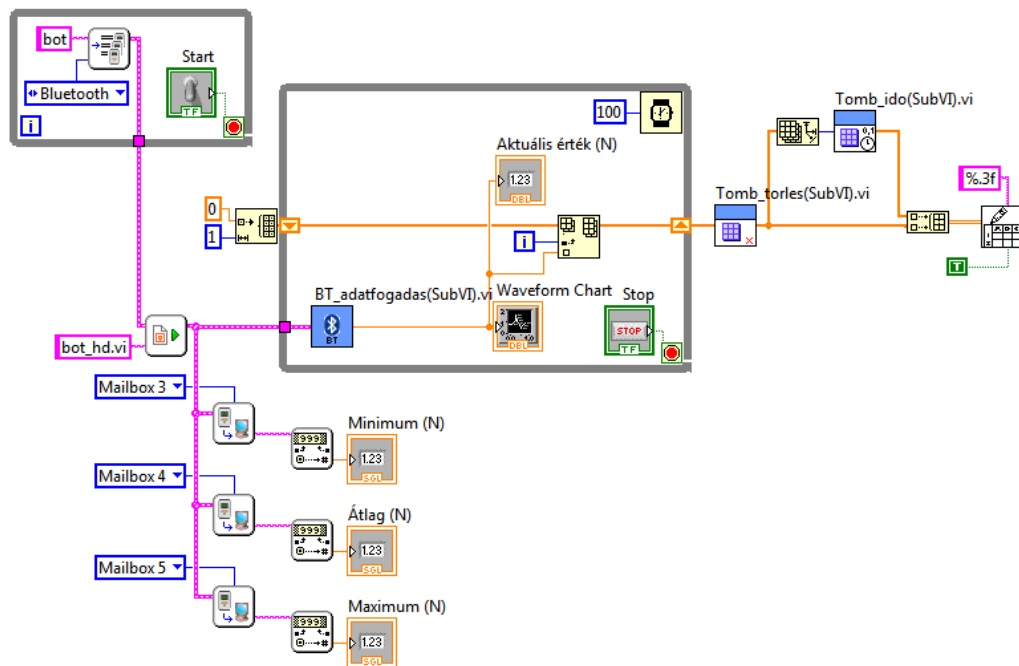
A program korai fázisában gyakran fordult elő, hogy negatív értékről indult a mérés. Ez az erőmérő használatából és a hőmérsékletből változásából adódik, mivel a szenzorban levő nyúlásmérő-bélyeg deformációt szenved és nem tér vissza az „alapállapotába”. Az offszet hiba kiküszöbölésére a szoftveres módot láttam megfelelőnek. A program indulásának pillanatában egy kezdőértéket olvas be a program, amit kinulláz. Ezt követően a mérés során ehhez a relatív nullához képest mér a program. Természetesen, a program nem engedi meg, hogy ezután ismét negatív érték kerüljön be, azokat is nullaként nyugtázza.



19. ábra A slave program futás közben az NXT-n

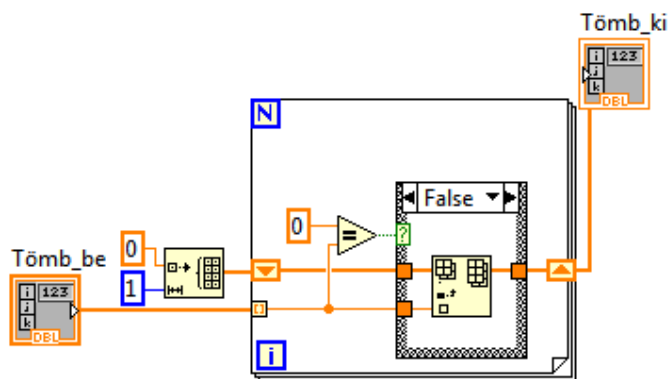
5.2. MASTER PROGRAM

Ez a program vezérli az 5. fejezetben ismertetett, NXT-re írt programot.



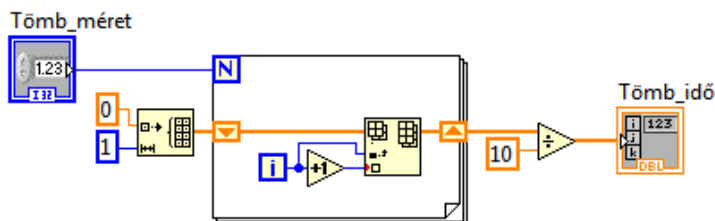
18. ábra Kézi erőmérő - master program

Ahogy a 20. ábrán látható, a program Start kapcsolóval indítható, Stop gombbal bármikor leállítható. Indítást követően a számítógép elindítja az NXT-n található kézi erőmérő slave programját. Ezután a kapott adatokat 100ms-ként olvassa be és tárolja le, közben real-time tájékoztat minket az aktuális erő értékéről, valamint ezt ki is jelzi egy grafikonon. Miután az NXT befejezte a mérést láthatóak a minimum, átlagos és maximum értékek. Ilyenkor le lehet állítani a programot. Mielőtt a program leállna, elvégzi a kapott adatok feldolgozását, kondicionálását. A program futása elején, amíg az NXT nem gyűjtött adatot, folyamatosan 0-t nyugtáz, hasonlóan így tesz miután befejezte a mérést és még nem állt le. A nulla értékek törlése a Tomb_torles subVI-ban történik. Mivel időbeli lefolyású volt a mérés, hozzá kell rendelni egy olyan tömböt, ami ezeket az időértékeket tartalmazza. Ez a Tomb_ido subVI-ban történik. Végül a program a kétdimenziós tömböt egy fájlba menti el, melynek helyét és nevét beírható a felugró ablakban.



19. ábra Tomb_torles subVI

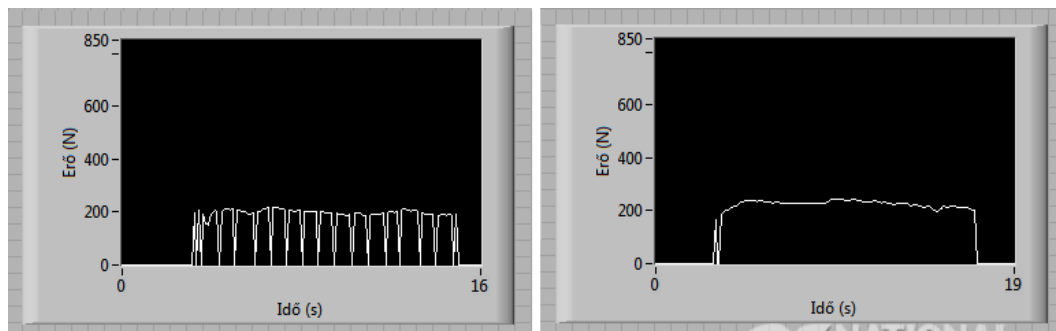
A Tomb_torles alprogram működése a következő. A beérkező tömb első elemét összehasonlítja a 0-val, és ha nem egyenlő, akkor egy új tömb első helyére másolja. Ha egyenlő, akkor nem történik másolás. jöhet a soron következő elem. Mindez egy for ciklusba van ágyazva és a bemeneti tömb indexelt átjárón keresztül van bekötve. Ez azt jelenti, hogy a ciklus a tömbön végigérve nem fut tovább.



20. ábra Tomb_ido subVI

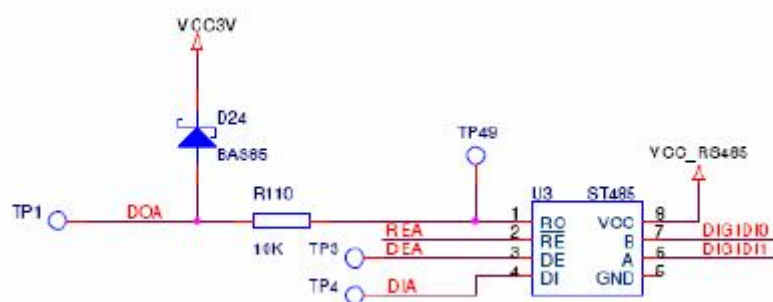
A Tomb_ido alprogram bemenete egy tömb mérete, ami meghatározza, hogy a for ciklus hányszor fut le. A program inicializál egy új tömböt és minden helyre a ciklus számlálójánál eggyel nagyobb számot ír. Az eggyel növelésre azért van szükség, mert a számláló 0-tól n-1-ig számol. Mivel az adatok 100ms-ként érkeznek, az elemeket 10-zel elosztva elkészül a megfelelő tömb.

Méréseket futtatva abba a hibába ütköztem, hogy az átküldött adatok között 0 értékek szerepeltek, egy bizonyos mintázatot követve. Ez a PC-n futó program grafikonján jelent meg, a 23. ábrán látható.

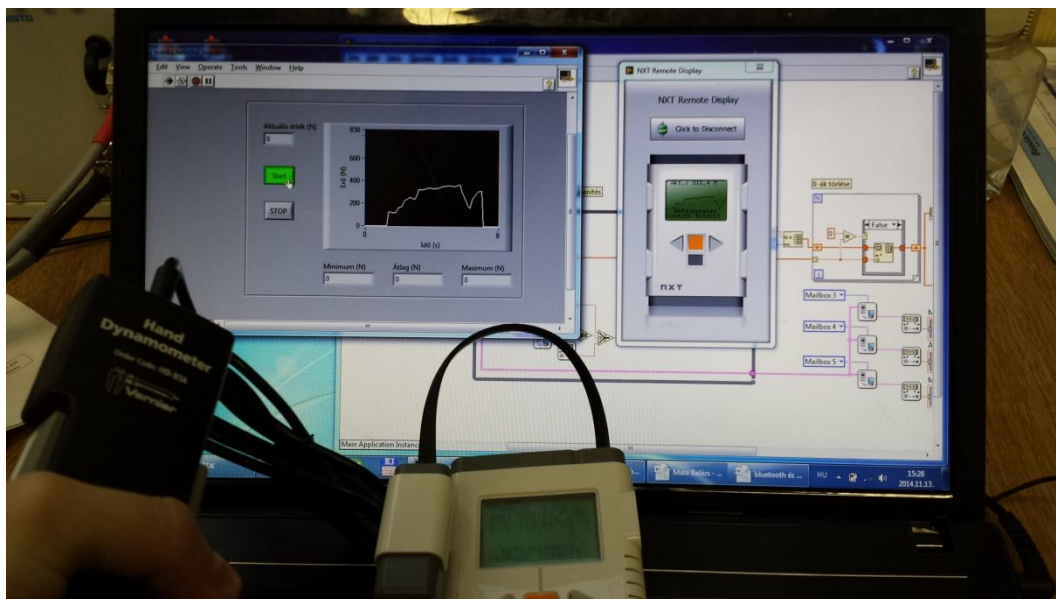


21. ábra Beérkezett adatok a hiba kijavítása előtt és után

USB-n keresztüli kommunikációnál ez nem jelentkezett, Bluetooth-on igen, tehát nem volt elég stabil a kapcsolat, bizonyos időközönként a küldött információ elvesződött. A probléma megoldására szoftver oldalon sorosan kötöttem be még egy küldő fogadó részt, egy másik mailbox-ra. Így ha az egyik csatornán nem érkezik meg az adat a szinkron-kisés miatt, a másikon nagy valószínűséggel megérkezik. Hardveres oldalon az NXT négy bemenete közül a 4-es jelű speciális, nagysebességű kommunikációra alkalmas port. Egy RS485-ös chip van beszerelve az alap áramkörbe, ami közvetlenül csatlakozik a főprocesszorhoz, onnan az adat SPI-buszon át, kerül a Bluetooth-hoz küldésre. Összességében ezzel a megoldással sikerült megnövelni a sebességet és biztosítani az adatáramlást, hogy a korrigálás utáni mérésekbe nem kerültek kihagyott elemek.



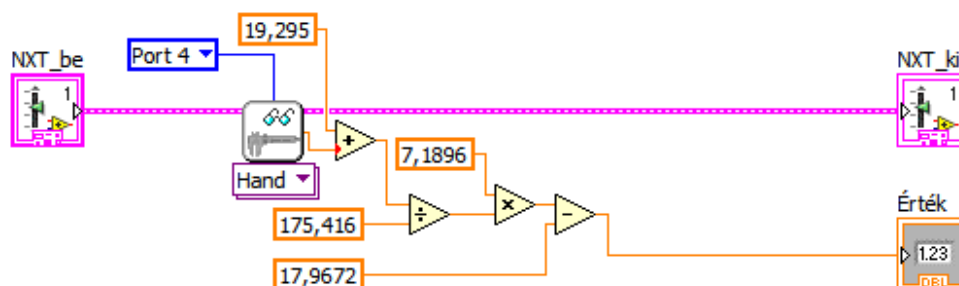
22. ábra NXT 4-es portjának áramköre [7]



23. ábra Az erőmérő program futás közben

6. A SPIROMÉTERHEZ ÍRT PROGRAM

A program megírása előtt az volt az alapvető probléma, hogy az NXT eszköztárában nem volt megfelelő blokk, ami a szenzor értékeit helyesen olvassa ki. A legegyszerűbb megoldásnak az bizonyult, hogy egy másik érzékelőhöz való blokkot átalakítottam. A kézi erőmérőt használtam alapként, mivel ugyan úgy feszültség alapján érzékel és a 0-5 Voltos tartomány is megegyezik. A könnyebb érthetőség és átláthatóság miatt egy subVI-t készítettem, amit egyedi ki/bemenetekkel és ikonnal láttam el.



24. ábra Spirométer subVI

Ahogy a fenti ábrán látható a subVI-ban a bejövő jelet átalakítom feszültség-értékké, a kézi erőmérő kalibrációs értékeivel. Ezután a spirométer kalibrációs értékeivel kimeneti értéket számolok, ami már a légzési sebesség L/s-ban kifejezett értéke.

Az általam használt spirométernek a légzésdiagnosztikában is használatos eszközökhöz hasonló tulajdonságai vannak, tehát egy megfelelő programmal kiválthatja azok használatát. A programozásnál a főbb paraméterek meghatározására törekedtem. Az elképzelésem a dohányzó, nem dohányzó, ex-dohányzó személyek összehasonlítása.

6.1.SLAVE PROGRAM

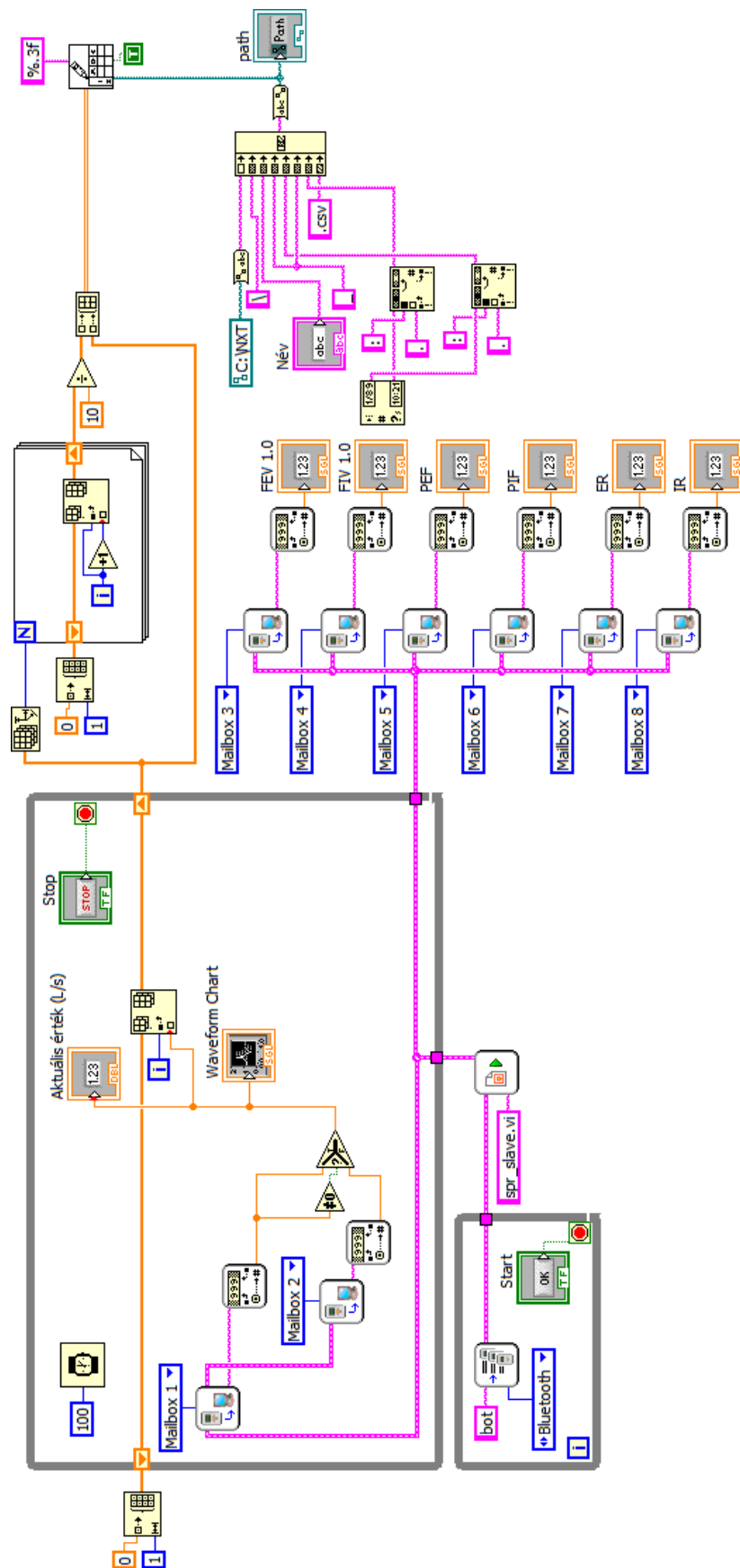
Az NXT téglára készített program elkészítésénél a kézi erőmérő programjára támaszkodtam, felépítésük emiatt hasonló. Lényeges különbség viszont az, hogy az adatgyűjtés több „szálon” fut. Az egyik ciklusba a nyugalmi 0 L/s-os értéktől különböző bemenet hatására lép be a program. Itt másodpercenként 10 adat kerül mérésre, rögzítésre és küldésre, továbbá a program dinamikusan vizsgálta a maximális (PEF) és minimális (PIF) értékeket, amik jellemző értékek a légzésdiagnosztikában. A további két ciklus a forszírozott értékek meghatározását végzi el (FEV1.0 és FIV1.0). A legelső

ciklusban pedig a kilégzési és belégzési tartalékokat (ER és IR térfogatot) méri a program.

Bizonyos feltételekkel a program alkalmas további értékek mérésére is, amivel meghatározható például a légzési térfogat (RV_T), vitálkapacitás (VC) és a tüdőkapacitás (TLC), fontos azonban, hogy a vitálkapacitás mérésénél figyelembe kell venni a vizsgált személy alkati sajátosságait kifejező korrekciós tényezőit is. Ezek táblázatokból leolvashatók.

6.2.MASTER PROGRAM

A számítógépen futó program fő feladata az NXT-ről érkező adatok mentése és rendszerezése. Felépítése és működése hasonló a kézi erőmérő master programjához. Indítása gombnyomásra történik, a mért adatok folyamatosan egy tömbbe kerülnek. Az NXT által meghatározott adatok a lefutás után kerülnek kiírásra. A fájlt a „C” meghajtó NXT nevű mappájába menti a program egyedi azonosítóval, amiben szerepel a név, és a dátum, könnyítve ezzel az esetleges adathalmaz rendszerezését és a későbbi visszakeresést.



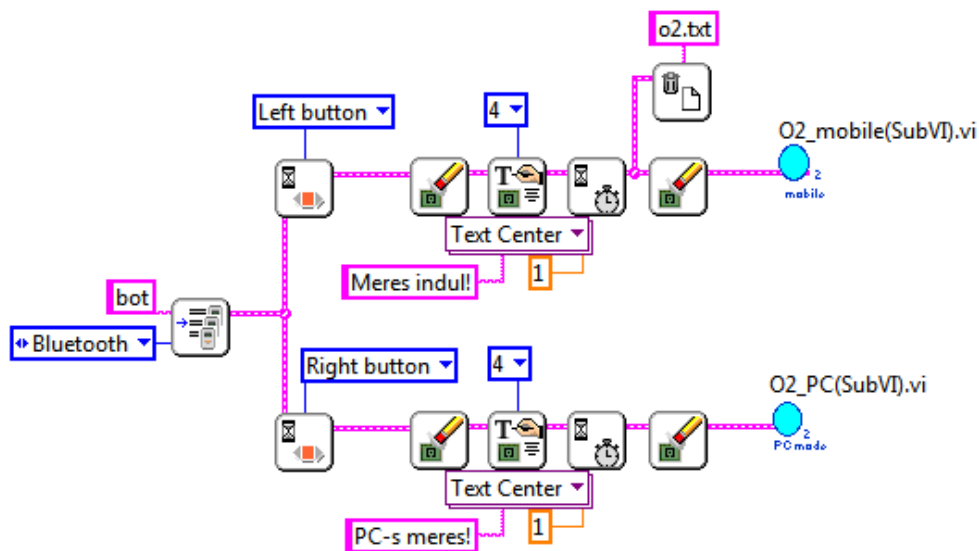
26. ábra Spirométer master programja

7. AZ OXIGÉNSZENZOR PROGRAMJA

Az oxigénszenzornak a levegő minőségének ellenőrzését adtam fő feladatként, de úgy írtam meg a programot, hogy a későbbiekben alkalmas legyen a spirométerrel együtt működni. Egy arra alkalmas „T” csatlakozóval a spirométerhez tudjuk kapcsolni az eszközt, így alkalmassá válik a be- és kilégzett levegő oxigéntartalmának meghatározására.

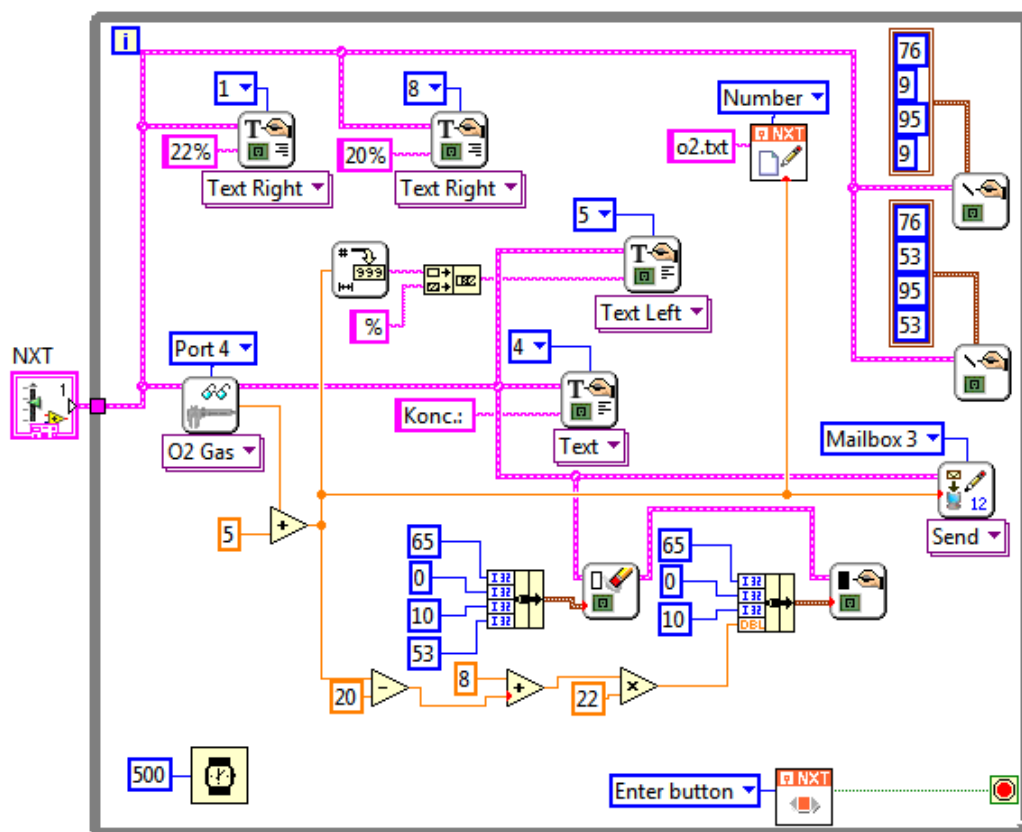
7.1.SLAVE PROGRAM

Ez a program az NXT-n két feladatot is elláthat. Működhet önálló mérőeszközként, ilyenkor az NXT Enter gombjával lehet elindítani és leállítani, a mért adatok hasonlóan tárolódnak az eszközben, mint a kézi erőmérő programjában. Másik esetben működhet a PC-vel, ebben az esetben az NXT Enter gombjára nyomva lehet mérni, majd ezt küldjük át a számítógépre tárolásra. A második esetben a mérés többször is elvégezhető ugyanabban a programfutásban, de minden gombnyomáskor csak egy pillanatnyi értéket mér le az érzékelő.



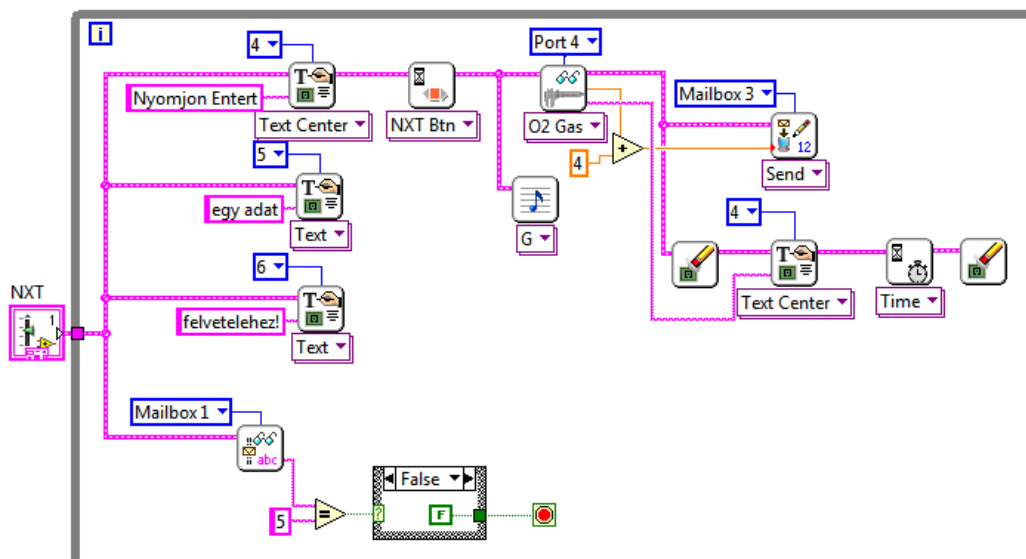
27. ábra Oxigénszenzor - slave program

A program egy választófelület, ahol a bal gomb megnyomásával a PC nélküli mérés indul el, a jobb gombbal pedig a PC-s. Az áttekinthetőség miatt ezt a két lehetőséget egy-egy subVI-ban írtam meg.



28. ábra PC nélküli mérés alprogramja

A 30. ábrán az O2_mobile subVI látható. A programban nagy hangsúly van az NXT kijelzőjén, az adatokat szám formátumban is kijelezi, valamint egy sáv szemlélteti ezt. Mivel általános esetben 21% körüli a levegő oxigéntartalma, ezért ennek az értéknek a környezetében akartam grafikus módon ábrázolni a kisebb változásokat. Több mérést végezve a szabadlevegőn, szobában, mély és magas pontokon, fénymásoló közvetlen közelében és egyéb helyeken, ezekből a mérésekből a ± 1 %-os környezetet találtam ideálisnak.

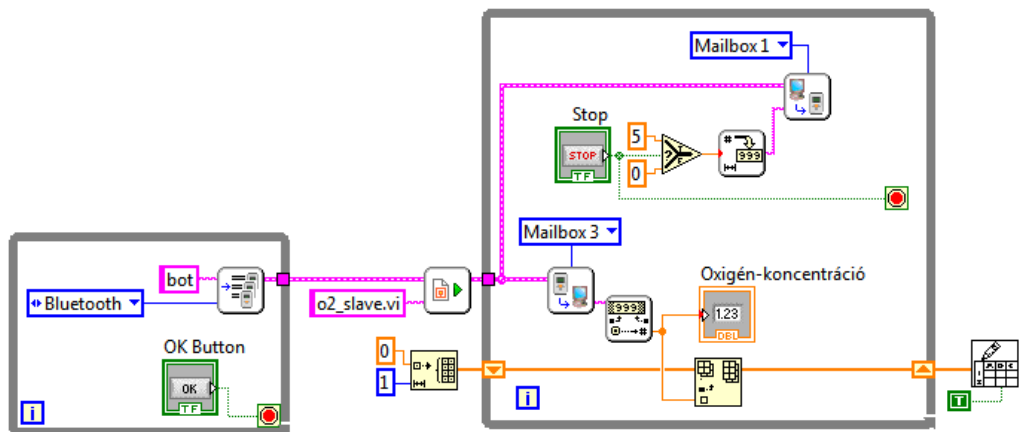


29. ábra PC-s alprogram blokkdiagramja

A 31. ábrán látható O2_PC subVI egy egyszerűbb folyamatot ír le. Itt az egyes mérések pillanatában hangjelzést is ad az NXT, majd az értéket 1 másodperc ki is jelzi a kijelző. A futás sajnos bármeddig nem tarthat, az „i” ciklusváltozó értéke elérheti a határértékét, ilyenkor a program hibával leáll. Mivel az „i” 2147483647 érték lehet maximálisan, ezért a gyakorlatban ezzel az eseménnyel felesleges számolni.

7.2.MASTER PROGRAM

A master itt csak egy kisebb részt szolgál ki, így a blokkdiagramja nem bonyolult. Az indítást követően az OK gombbal indítható az adatgyűjtés. A program az NXT-n elindítja a slave programot, itt ki kell választani a PC-s mérést jobb gombbal, ezután a számítógép egy tömbbe menti az érkezett adatokat. A programban megoldott az, hogy amíg nem érkezik az NXT-től adat, addig ne növelje a tömböt, így a végén egyszerűbben fájlba írható. Mindkét eszközön futó program egyszerre leállítható a Stop gombra kattintva.



30. ábra Az oxigénérzékelő master programja



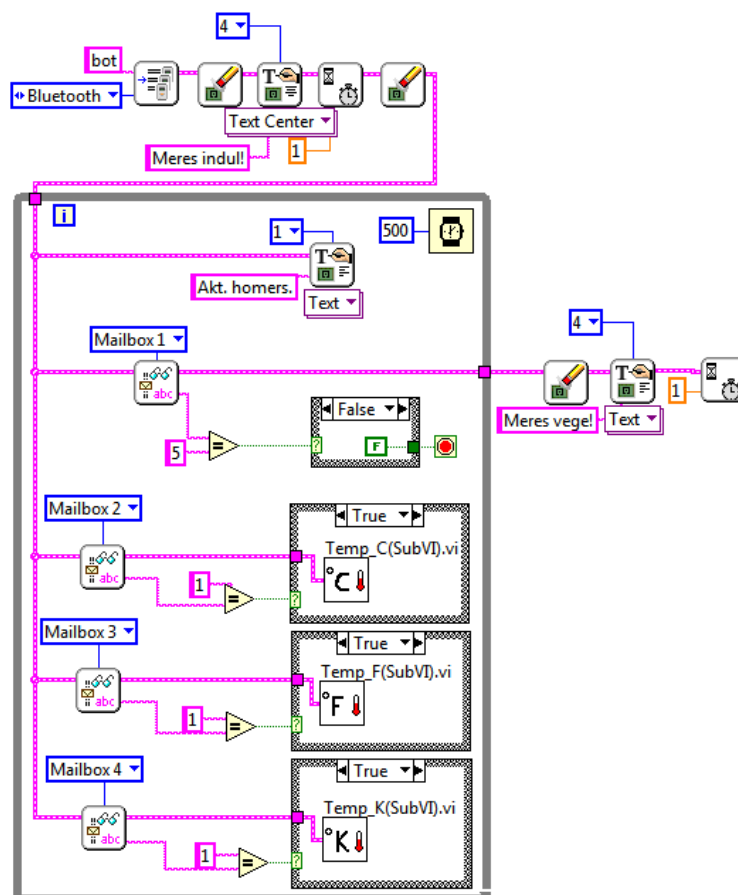
31. ábra Az oxigénmérő működés közben

8. A FELÜLETI HŐSZENZORRA ÍRT PROGRAMOK

Ez az eszköz tökéletesen alkalmas testhőmérséklet mérésére, emiatt a végleges programot ehhez igazítottam. Elsőként megbizonyosodtam arról, hogy a hagyományos „higanyos” hőmérőhöz képest mennyire pontos, de nem tapasztaltam szignifikáns eltérést és ezen kívül az alkalmazott eszköznek két előnyét is felfedeztem. Az egyik a másfélszeres gyorsaság a higanyos és az elektromos hőmérőkhöz képest, a másik pedig a tetszőleges pontosság beállítása.

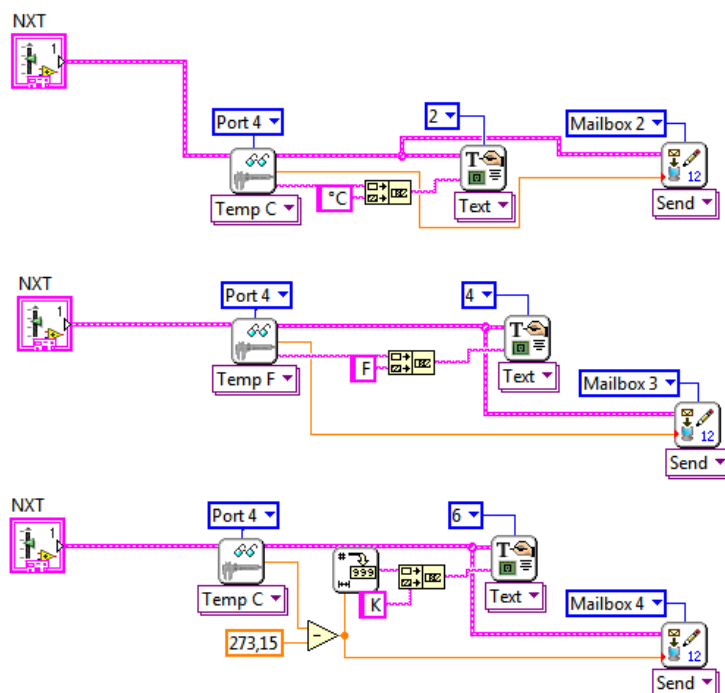
8.1.SLAVE PROGRAM

Ez az érzékelő gyorsnak számít a hőszenzorok között, de még így is relatíve sok időbe kerül a beállítás. Függetlenül a környezeti hatásoktól (pl. a levegő áramlása) minimum 20 maximum 50 másodperc szükséges ahhoz, hogy a valós értékre álljon be. Az érzékelő Celsius és Fahrenheit skálában is képes mérni, a programban mindkettőt használtam, valamint a Celsiusból Kelvint is számítottam. Mindhárom módszerre külön subVI-t írtam, így átláthatóbb az anyaprogram. Az NXT kijelzőjén az aktuális érték megjelenítése tetszőlegesen állítható a számítógépről. Ezt a *case* struktúra alkalmazásával értem el. A választó bemenet a szerint változik, hogy milyen üzenetet kap bluetoothon keresztül. Ha 0-t, akkor a *case* struktúra *false* esete aktív, ha 1-et, akkor a *true*, ahová a saját subVI-jaimat tettem be. A PC-ről történő indítás, leállítás és az adatküldés itt is megoldott.



32. ábra Hőszenzor - slave program

A slave programban található subVI-ok nagyon hasonlóak egymáshoz, pár apró eltérés van közöttük. A Temp_C Celsiusban mér, a Temp_F Fahrenheitben, ezeknél csak az erre a célra alkalmas blokkokat használtam. A Temp_K alprogram Kelvin skáláját a Celsius skála 273,15-ös eltolásával hoztam létre.



33. ábra Az egyes hőmérséklet skálák alprogramjai

8.2.MASTER PROGRAM

Az indítás és leállítás az előbbi programoknál használt módon működik. A blokkdiagramon látható, hogy a ciklusban három egymással párhuzamosan kötött egység van, amiknek a programozása ugyan azt a sémát követi. Egy kapcsolót ON állásba kapcsolva a master program 1-et küld üzenetként a slave-nek és az így érkező adatokat a master egy egy tömbbe menti. A tömb felépítése az alábbi kimentett táblázaton látszik. A program indulásnál létrehozza a fejléct, majd azon értékek, amikhez tartozó kapcsoló fel van kapcsolva elmentésre kerülnek, illetve egy hőmérő grafikusán jeleníti meg. Lekapcsolt állapotban lévő csatorna nem szolgáltat adattal, a mentett fájlban üres cellaként jelenik meg, ez látható az 1. táblázatban.

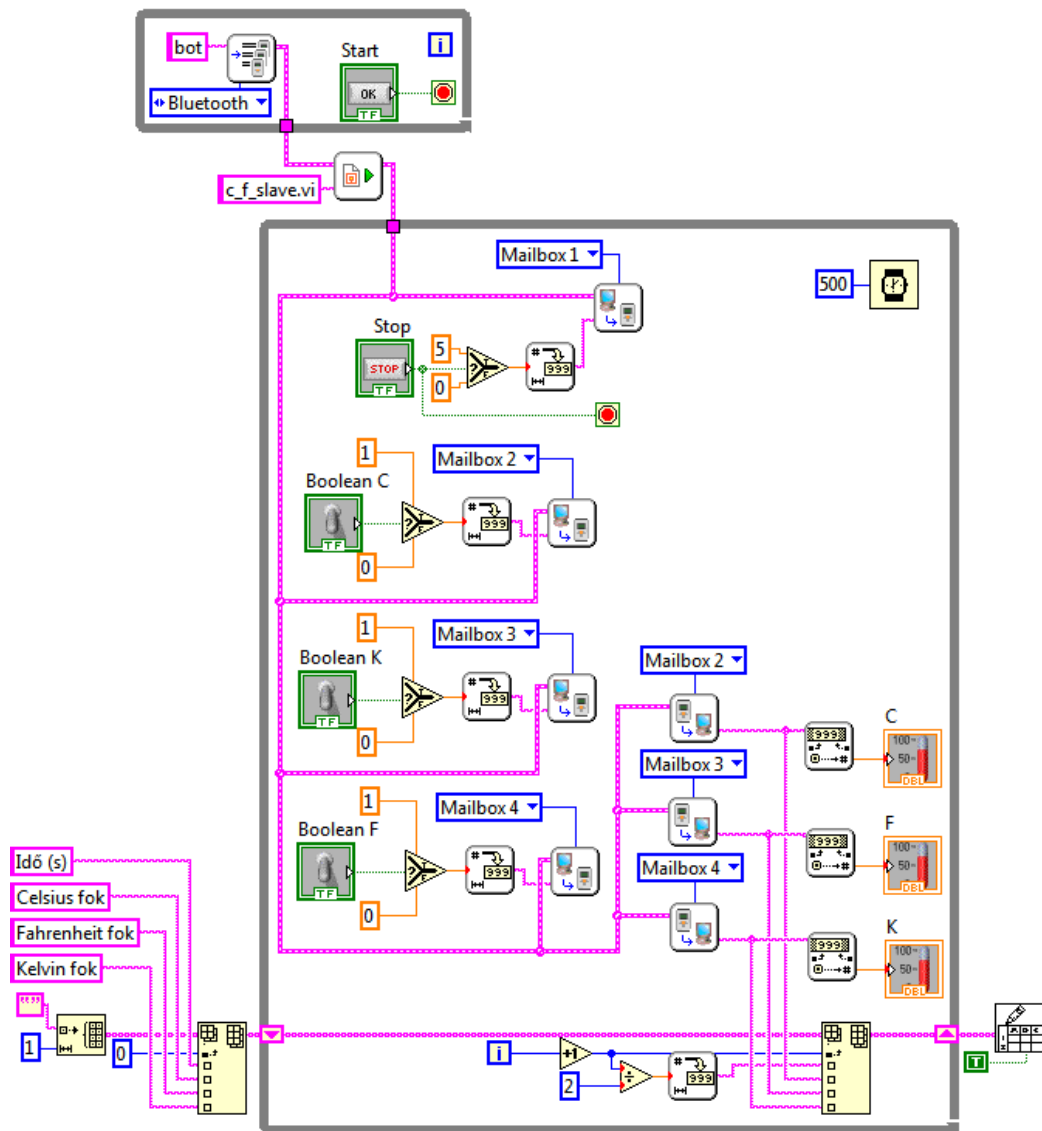
1. táblázat Hőszenzor programjával kimentett adatok

Idő (s)	Celsius fok	Fahrenheit fok	Kelvin fok
0,5	37,895		
1	37,895		
1,5	37,368		
2	37,158	98,884	

2,5	36,425	97,565	
3	35,904	96,627	309,054
3,5	34,765	94,577	307,915
4	32,92	91,256	306,07
4,5		84,159	302,127



34. ábra Hőszenzorral történő mérés

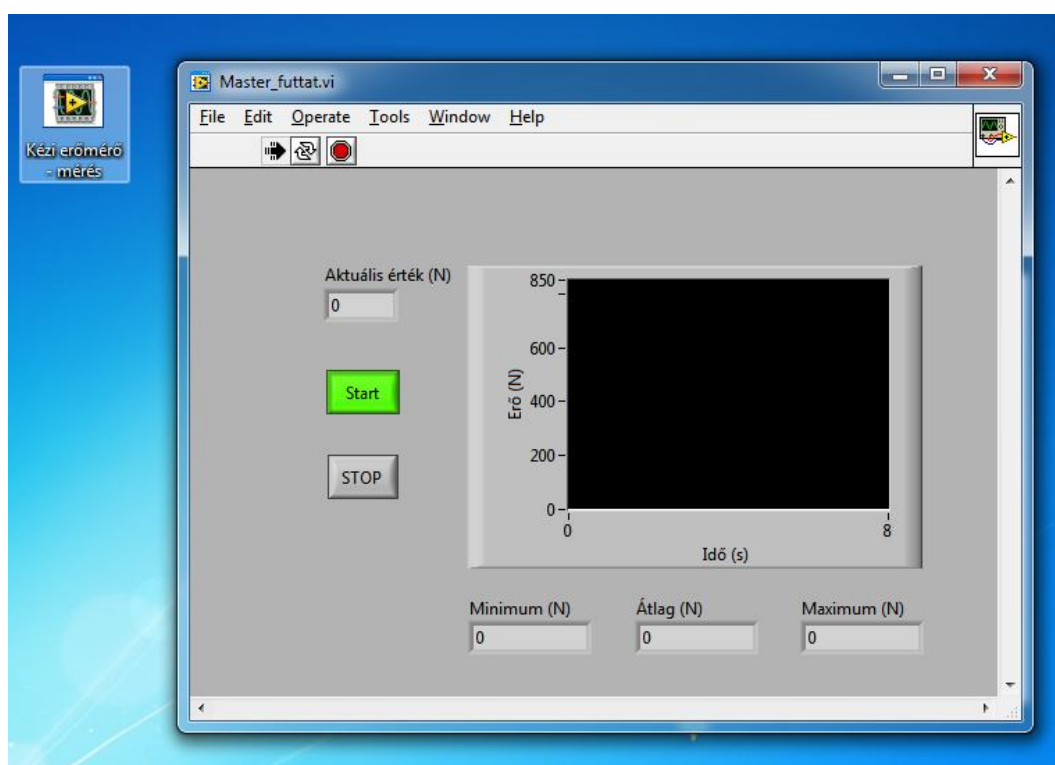


35. ábra Hőszenzor - master program

9. FUTTATHATÓ ÁLLOMÁNYOK KÉSZÍTÉSE

A LabVIEW-ban lehetőség van a kész projektek front paneljeiből futtatható .exe állományt készíteni. Ez számos előnnyel jár. Elsősorban a méréshez ezután nincs szükség a LabVIEW fejlesztőkörnyezetére. A megfelelő Run-Time Engine-t feltelepítve megfelelően futtathatóak az így elkészített fájlok. További előny, hogy a program egyetlen fájlból áll, amit könnyebb kezelni és a felhasználók számára könnyebb duplán kattintva egy ikonon elkezdni a mérést, mint projekteket megnyitni és keresni benne a megfelelő VI-t, majd futtatni a fejlesztőkörnyezetben. [14]

Az általam készített .exe fájlok sikeres futtatásához a 2010-es, 32-bites SP1-es Run-Time Engine alkalmas.



36. ábra Kézi erőmérő futtatható állománya

10. FEJLESZTÉSI LEHETŐSÉGEK

A Bluetooth kapcsolat működésével illetve a kommunikáció gyorsaságával eleinte problémáim akadtak, valamint az NXT tárhelyének a kis mérete miatt, a programokat kompaktabb formában szükséges az eszközre tölteni, amivel várhatóan a Bluetooth kommunikáció is javulna.

A dolgozat elkészítése alatt egy darab Vernier - NXT adapter állt rendelkezésemre. További adapterek beszerzése után az eszköz alkalmas lenne párhuzamosan mérni akár 4 szenzorral is, ami komplex elemzésre is lehetőséget adna.

Mivel az NXT csomag tartalmaz motorokat is, amiket ráépítve a téglára irányításra alkalmas eszközzé alakítható. A LabVIEW-ban könnyen implementálható rá távvezérlő program, mellyel a nehezen vagy egyáltalán nem megközelíthető helyeken is lehetne mérni.

A PC kommunikáció esetében nagy előny, hogy egyszerre több NXT-hez is lehet kapcsolódni illetve bizonyos hierarchiák is kiépíthetőek az NXT-k között.

11. ÖSSZEFOGLALÁS

Nagy kihívás volt számomra, hogy TDK dolgozatomban ezzel a témával foglalkozhattam. Mind a szenzorok újszerű, mobilalapú mérésének kidolgozása, mind pedig a kapott eredmények rendszerbe való behelyezése izgalmas feladat volt.

Dolgozatomban bemutattam a különböző Vernier szenzorokat és az NXT működését és teljesítettem a mobil platformra épülő mérőrendszer összeállítását. Összekötve az eszközöket feltérképeztem a Bluetooth kommunikáció nyújtotta lehetőségeket. Rávilágítottam a PC-NXT közti vezeték nélküli kommunikáció egyes nehézségeire és kísérletet tettem azok megoldására. Megmutattam, hogy az egyes szenzorok előre kalibrált állapot ellenére hajlamosak elállítódni, ezeket próbáltam minél hatékonyabban kiküszöbölni.

4 különböző szenzorral, külön-külön programot írtam, törekedve a különböző megoldásokra.

Megvalósítottam a vezeték nélküli adatmentést is, melynek segítségével a mért adatokat könnyebben, egyszerűbben fel lehet dolgozni.

Az általam megvalósított mérőrendszer működését a gyakorlatban videóra rögzítettem, amit a Youtube videómegosztó portálon közzétettem, így lehetőség van megtekinteni a következő linkeken:

- Oxigénérzékelővel szerelt NXT mérőrendszer
<https://www.youtube.com/watch?v=FItkmbaCj44>
- Vernier hőszennzorral szerelt NXT mérőrendszer
<https://www.youtube.com/watch?v=IQefob6CqP8>

IRODALOMJEGYZÉK

[1] http://www.vernier.com/images/cache/product.hd-bta._physiology._hero.002.590.332.jpg

Letöltve: 2014. október 5.

[2] Vernier Hand Dynamometer manual

<http://www.vernier.com/files/manuals/hd-bta.pdf>

Letöltve: 2014. október 5.

[3] http://www.dynatech.de/data/bilder/SEN_SPR_GG.jpg

Letöltve: 2014. október 5.

[4] Vernier Spirometer manual

<http://www2.vernier.com/booklets/spr-bta.pdf>

Letöltve: 2014. október 5.

[5] http://www.vernier.com/images/cache/product.hd-bta._physiology._hero.002.590.332.jpg

Letöltve: 2014. október 5.

[6] Vernier O₂ Gas Sensor manual

<http://www.vernier.com/files/manuals/o2-bta.pdf>

Letöltve: 2014. október 5.

[7] http://www.vernier.com/images/cache/product.hd-bta._physiology._hero.002.590.332.jpg

Letöltve: 2014. október 5.

[8] Vernier Surface Temperature Sensor manual

<http://www.vernier.com/files/manuals/sts-bta.pdf>

Letöltve: 2014. október 5.

[9] LEGO MINDSTORMS NXT Hardware Developer Kit

<http://cache.lego.com/upload/contentTemplating/Mindstorms2SupportFilesDownloads/otherfiles/download8CFD37F17F7EFCDC412AE7CEBF245C6A.zip>

Letöltve: 2014. október 13.

[10] http://www.vernier.com/images/cache/bta-nxt_home.896.279.jpg

Letöltve: 2014. október 5.

[11] LabVIEW Online Mérésadatgyűjtési Szeminárium

2014. április 3.

[12] LabVIEW Toolkit for LEGO MINDSTORMS NXT Programming Guide

<http://zone.ni.com/devzone/cda/tut/p/id/4435>

Letöltve: 2014. október 13.

[13] Grip and Pinch Strength: Norms for 6- to 19-Year-Olds

http://www.researchgate.net/publication/19385043_Grip_and_pinch_strength_norms_for_6-to_19-year-olds

Letöltve: 2014. november 3.

[14] LabVIEW és Mérésadatgyűjtés Gyakorlati Szeminárium – Feladatgyűjtemény

National Instruments, 2014.

[15] Harsányi Gábor - Érzékelők az orvosi biológiában

Budapesti Műszaki Egyetem, 2005.

[16] Sipeky Attila: Grafikus Programozás LabVIEW-ban

Pécsi Tudományos Egyetem, 2003.

[17] xenon.bibl.u-szeged.hu/~vidaa/holi/02/elettan/tetelek/27.pdf

Letöltve: 2014. március 2.

[18] LabVIEW alapismeretek

<http://www.sze.hu/~jager/LabVIEW/tananyagok/>

Letöltve: 2014. november 3.

[19] Oxigénérzékelővel szerelt NXT mérőrendszer

<https://www.youtube.com/watch?v=FItkmbaCj44>

[20] Vernier hőszenzorral szerelt NXT mérőrendszer

<https://www.youtube.com/watch?v=IQefob6CqP8>

ÁBRAJEGYZÉK

1. ábra Kézi erőmérő	4
2. ábra Spirométer	5
3. ábra Oxigénérzékelő	6
4. ábra Hőmérsékletérzékelő	7
5. ábra NXT belső felépítése	8
6. ábra Az NXT képernyője	9
7. ábra A mérőrendszer felépítése.....	10
8. ábra Vernier szenzor adapter	10
9. ábra Általam készített blokkdiagram részlete.....	11
10. ábra Specify NXT blokk.....	12
11. ábra Display blokkcsoport	12
12. ábra Wait for blokkcsoport.....	13
13. ábra Vernier Read blokk.....	13
14. ábra Easy Write File natív blokk.....	14
15. ábra Communication Control blokk.....	14
16. ábra Read from NXT és Write to NXT blokkok	15
17. ábra Run és Stop Program blokkok.....	15
18. ábra Kézi erőmérő - slave program	17
19. ábra A slave program futás közben az NXT-n	18
20. ábra Kézi erőmérő - master program	19
21. ábra Tomb_torles subVI	20
22. ábra Tomb_ido subVI	20
23. ábra Beérkezett adatok a hiba kijavítása előtt és után.....	21
24. ábra NXT 4-es portjának áramköre	21
25. ábra Az erőmérő program futás közben.....	22

26. ábra Spirométer subVI	23
27. ábra Spirométer slave programja	25
28. ábra Spirométer master programja.....	27
29. ábra Oxigénszenzor - slave program	28
30. ábra PC nélküli mérés alprogramja	29
31. ábra PC-s alprogram blokkdiagramja.....	30
32. ábra Az oxigénérzékelő master programja.....	31
33. ábra Az oxigénmérő működés közben	31
34. ábra Hőszenzor - slave program	33
35. ábra Az egyes hőmérséklet skálák alprogramjai.....	34
2. táblázat Hőszenzor programjával kimentett adatok.....	34
36. ábra Hőszenzorral történő mérés.....	35
37. ábra Hőszenzor - master program	36
38. ábra Kézi erőmérő futtatható állománya.....	37

Köszönetnyilvánítás

Szeretném megköszönni Dr. Tóth János Tanár Úrnak a segítséget és a támogatást, hogy dolgozatom elkészülhessen.