

Környezeti paraméterek mérése és kiértékelése

TDK-dolgozat

2014



Zsupányi Krisztián (ZSKSAAG.KEFO)

zsupi1987@gmail.com

Kecskeméti Főiskola,

GAMF Kar Informatika Tanszék

Konzulens:

Drenyovszki Rajmund (Főiskolai tanársegéd)

drenyovszki.rajmund@gamf.kefo.hu



TARTALOMJEGYZÉK

1	Tartalmi összefoglaló.....	2
2	Bevezetés	3
2.1	Smart Grid.....	4
2.2	Fogyasztói oldal befolyásolása, CAC algoritmus	5
2.3	Környezeti paraméterek felhasználása becslésre.....	6
3	Méréshez készített hardver elemek.....	8
3.1	Mérőrendszer felépítése	8
3.2	A megépített mérőegység.....	10
3.3	Adatkoncentrálor központi modul	19
4	Mérést végző beágyazott szoftver	21
4.1	Mérőegység szoftvere	21
4.2	Adatkoncentrátor szoftvere	25
4.3	PC oldali webszerver, adatbázis és weboldal.....	28
5	Mérés megtervezése és menete	33
5.1	Mérések helyszíne	33
5.2	Gyűjtött adatok	33
6	Jelenlét becslése a mozgásérzékelő adataiból	37
6.1	Megoldási javaslatunk, becslés jóságának mérése	37
7	Smart* adatbázis feldolgozása, idősorok közötti összefüggések feltárása (jelenlétből fogyasztás becslése)	41
7.1	Smart* adatbázis feldolgozása	41
8	Eredmények ÉRTÉKELÉSE, TERVEK.....	44
9	Irodalomjegyzék	45
10	MELLÉKLET	47

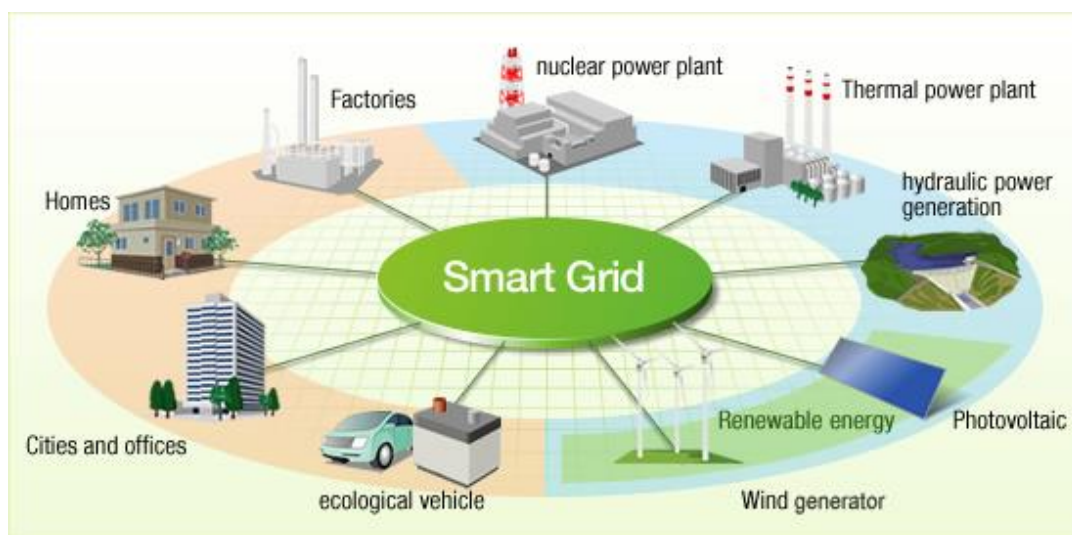
1 TARTALMI ÖSSZEFOGLALÓ

TDK dolgozatomban az újgenerációs villamos termelő és elosztó hálózatokhoz (a szakirodalomban: Smart Grid) kapcsolódó egyik fontos problémával foglalkozom, nevezetesen az új mérési eljárások által nyújtott többlet információ felhasználásának lehetőségeivel. A villamos hálózat korszerűsítésének egyik eleme az, hogy most már háztartások, sőt készülékek szintjén vannak mérések. Rendelkezésünkre fog állni információ a fogyasztásokról és egyéb, készülékekre jellemző paraméterekről, de elképzelhető az is, hogy ezek mellett a környezeti (fizikai) mennyiségeket is folyamatosan mérjük. A belső és külső hőmérséklet, páratartalom, fényviszonyok és legkézenfekvőbb módon a jelenlét szoros összefüggésben vannak a fogyasztással. Az összefüggések feltárásához és elemzéséhez először is megbízható, részletes mérési adatokra lenne szükségünk, majd ezeket felhasználva módszereket, algoritmusokat lehet kifejleszteni a gyakorlati alkalmazásokhoz.

Dolgozatomban először a probléma általános tárgyalása olvasható, a többlet információ Smart Grid-ben történő felhasználására javaslatot adunk, majd ezt követik egy ilyen célra kifejlesztett mérőrendszer megtervezéséről, felépítéséről, szoftveréről és beüzemeléséről szóló fejezetek. A mérőrendszer bemutatása után szó lesz a mérések menetéről, a gyűjtött adatokról, végül azok felhasználásáról (jelenlét becslése a mozgásérzékelő adatiból). A dolgozat végül az idősorok közötti összefüggések feltárásának módszereivel foglalkozik.

2 BEVEZETÉS

Társadalmunk növekvő energiaigénye, azon belül is az egyre fokozódó elektromos áramigény egy sor kihívás elé állítja az energia szektor szereplőit. Az utóbbi években sok újfajta elektromos eszköz jelent meg a háztartásokban, melyek fogyasztása nagymértékben véletlenszerű. Gondoljunk csak az egyre nagyobb ütemben terjedő (és ez a terjedés vélhetőleg exponenciális jellegű) elektromos autókra, amiknek energiaigénye és töltés közbeni teljesítményfelvétele meghalad szinte minden mai háztartási eszközét. A töltés megkezdése és időtartama kiszámíthatatlan, hiszen függ a felhasználóktól. Az új fogyasztókon kívül, erősödő igény Európában, de a világban is, a megújuló energiaforrások használata a hagyományos (fosszilis) energiahordozók helyett. Míg a hagyományos olaj-, szén- és gáztüzelésű valamint nukleáris erőművek jól tervezhetőek, a zöld energiaforrások, mint szél és a nap erősen függenek az időjárástól, tehát előre nehezen becsülhető a segítségükkel termelt energia mennyisége (és időtartama). A villamos energia hálózat egyik legfontosabb feladata, hogy fenntartsa az áramtermelés és fogyasztás közötti kényes egyensúlyt. A szolgáltatók előre betervezett energiatermeléssel számolják a költségeiket, túlermelés esetén az energia elvész (pl. a Dunát fűtjük vele), előre be nem tervezett extra fogyasztás esetén pedig gyorsan bekapcsolható, de drága energiatermelőket kell használniuk (pl. olajgenerátorok).



1. ábra Smart grid illusztráció

2.1 Smart Grid

A villamos hálózat egyensúlyának fenntartása egyre nehezebben megoldható feladat lesz már a közeljövőben is, amit a hagyományos villamos energiatermelő és elosztó rendszer nem tud hatékonyan elvégezni. Politikai és társadalmi nyomás miatt a növekvő fogyasztást sem lehet újabb szennyező (szén, olaj) vagy éppen veszélyes üzemű (atomenergia) erőművek telepítésével kielégíteni. A kutatók, politikusok és szolgáltatók már egy ideje felismerték az újgenerációs villamos energia rendszer felépítésének szükségességét, aminek összefoglalásaként a "Smart grid" elnevezés terjedt el. A kifejezés (magyar fordítása kb. "okos hálózat") főleg az információ- és kommunikációs technológiák intelligens felhasználására utal, ami magában foglalja a kétirányú kommunikáción túl a kétirányú energiaáramlás biztosítását is. Az új hálózat létrehozásának célja az energiateljesítmény minőségének, hatékonyságának növelése, a megújuló energiahordozók integrálása és a rendszer megbízhatóságának fokozása (1. ábra).

A megújuló energiaforrások és új fogyasztók (elektromos autók) bizonytalan viselkedését a végpontokon elhelyezett okos mérőeszközök által biztosított információ és annak algoritmikus feldolgozása tudja ellensúlyozni. A központi irányításon túl szükséges a kommunikációs csatornákkal összekötött beágyazott számítógépeken futtatott újfajta irányító programok alkalmazása. A fogyasztói oldal befolyásolásával mód nyílik a csúcspotasztás elhalasztására, valamint alacsony hálózati kihasználtság esetén extra fogyasztások bekapcsolására, ami lehet például elektromos autók töltése (2. ábra).



2. ábra Elektromos autók töltése parkolóházban

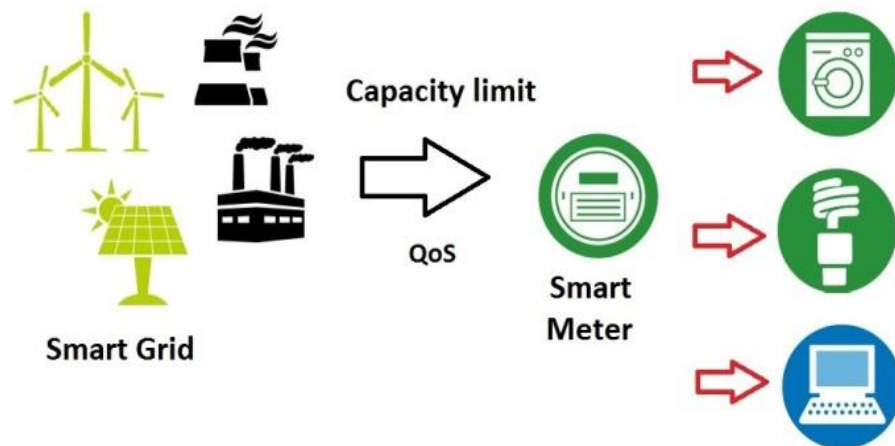
2.2 Fogasztói oldal befolyásolása, CAC algoritmus

A fogyasztói oldal befolyásolása korlátozott keretek között már régóta alkalmazott stratégia a villamos energia hálózatban. Újabb megvalósításai (1. táblázat) sikerrel vethetőek be több jelentkező probléma megoldására [1].

1. táblázat

Megoldás	Cél	Beavatkozás szintje	Időlépték
Közvetlen vezérlés	Fogyasztás formálása, csúcsfogyasztás levágása, fogyasztás elhalasztása, alulfogyasztás kitöltése, a hálózat stabilitásának érdekében a termelés és fogyasztás egyensúlyba hozása.	Készülékszintű	~ perc
Valósídejű árazás		Fogyasztószintű	~ óra
Fogyasztás átütemezése		Városszintű	~ egy nap

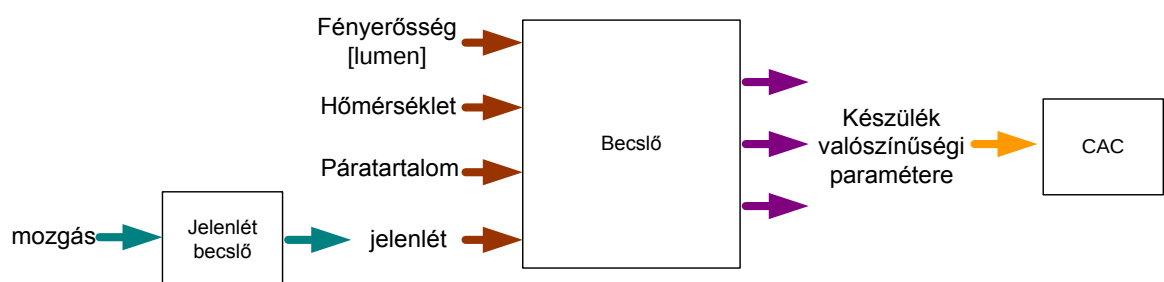
Mi* a készülékszínten megvalósítható közvetlen vezérléssel foglalkozunk, melyre megoldási javaslatunk a Consumption Admission Control elnevezésű algoritmikus eljárás [2] (*A Kecskeméti Főiskolán zajló Smart Grid kutatásokat végző csoport munkájába kapcsolódtam be.). A CAC algoritmus működésekor a Smart Grid hálózathól érkezik (3. ábra) a kapacitáshatár (Capacity limit) és annak betartására vállalt valószínűségi érték (QoS- Quality of Service). Ezeket a paramétereket használja fel a Smart Meter-ben futó algoritmus és megpróbálja a készülékek kapcsolgatásával alakítani azok fogyasztási valószínűségét (agregált valószínűség eloszlását). Lényege, hogy alkalmasan megválasztott készülékek üzemeltetését más időpontra lehet halasztani, ezzel befolyásolva a fogyasztás összértékét és időbeli alakulását. Ehhez szükség van a végpontokon „okos” eszközökre, melyek vezérlési utasításokat tudnak fogadni. Ezen kívül felhasználjuk a fogyasztások készülékszintű mérését, továbbá környezeti paraméterek regisztrálását.



3. ábra A CAC algoritmusban használt paraméterek

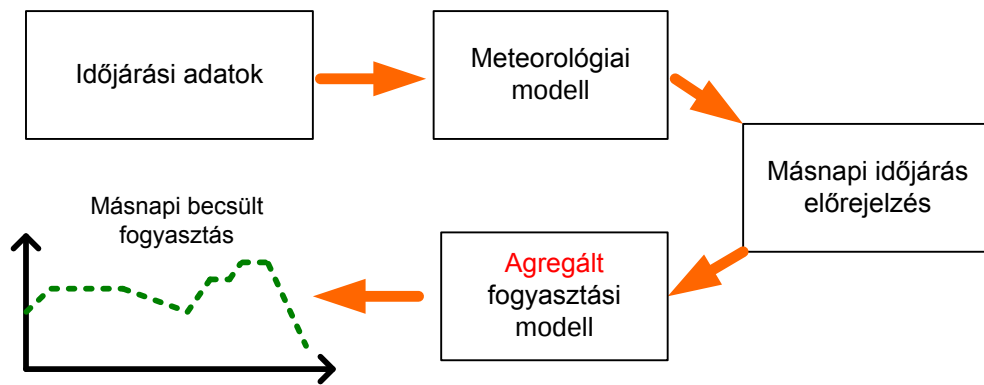
2.3 Környezeti paraméterek felhasználása becslésre

Egyik fontos kutatási terület annak kiderítése is, hogy az összegyűjtött többlet információval mit lehet kezdeni. Elképzelésünk szerint a környezeti és egyéb paraméterek (jelenlét, külső és belső fényerősség/hőmérséklet) felhasználhatóak a készülékek fogyasztásának becslésére (4. ábra), ami felhasználható algoritmusunk javítására is. A CAC algoritmus működésének egyik kulcsfontosságú eleme továbbá a benne szereplő készülékek fogyasztásának modellezése, hiszen ezek alapvetően befolyásolják a becslő módszereket, amikből a összfogyasztás is jósolható (6. ábra).



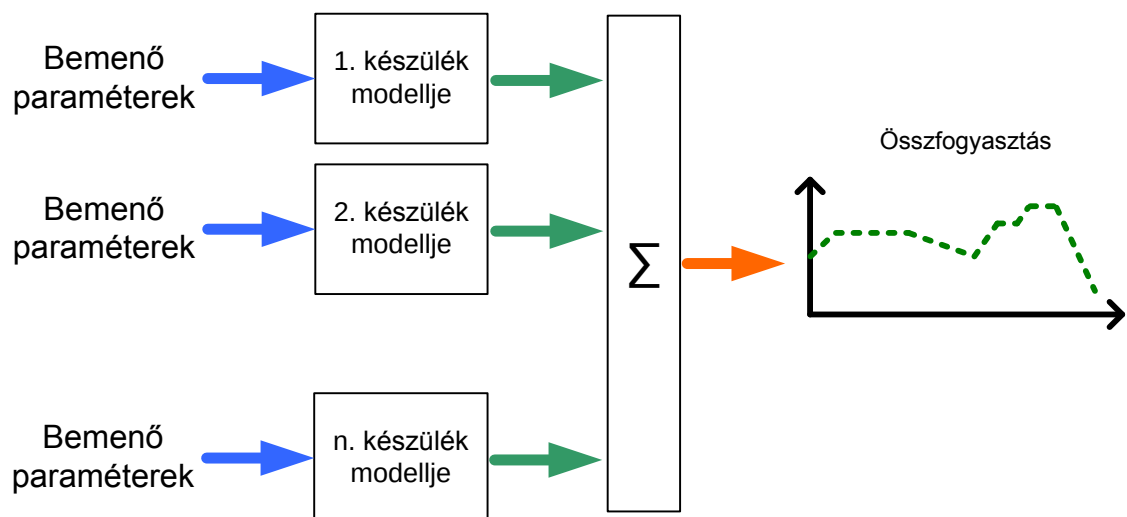
4. ábra Becslés elve

Becslésre jelenleg is vannak eljárások, például országos vagy regionális szinten az időjárási adatokat fel tudják használni fogyasztás előre megbecslésére [5. ábra].



5. ábra Másnapi fogyasztás előrejelzése meteorológiai adatokból

Ezzel szemben a mi megközelítésünk fordított, hiszen alacsony szinten próbáljuk meg megbecsülni az egyes készülékek fogyasztását az elérhető mérési adatokból, legyenek azok mozgásérzékelő vagy környezeti paramétereket mérő egységek jelei. Az egyik legegyszerűbb lehetőségként az időpontból és a jelenlétből a készülék bekapcsolás valószínűségét kíséreljük megbecsülni. Ehhez szükségünk van mérési adatokra. A Smart Grid szakirodalmat tanulmányozva gyorsan kiderül, hogy elérhetőek ugyan háztartások mérési adatai (REDD data set [3], Smart* data set [4], ECO data set [5]), azonban egyetlen adatbázis sincs, ami ideálisan felhasználható a vizsgálatainkhoz. Így mivel nincsenek ilyen mérési eredmények, saját méréseket kell végeznünk, amihez először is szükségünk van egy mérőrendszerre.



6. ábra Összfogyasztás becslése a környezeti paraméterek felhasználásával

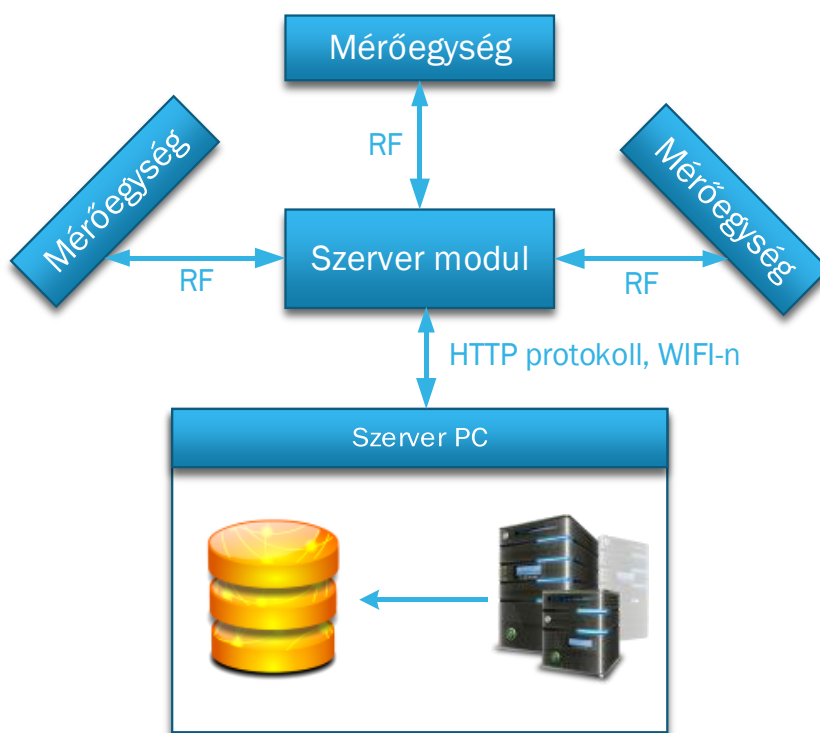
3 MÉRÉSHEZ KÉSZÍTETT HARDVER ELEMEEK

A környezeti paraméterek mérésére számos eszköz áll rendelkezésünkre. A megvásárolható megoldások általában nem tartalmazzak minden, számunkra fontos érzékelőt, továbbá az egyszerűbb eszközök esetén gyakran előfordul, hogy a mérési adatokat csak megjeleníteni képesek, tárolni és pl. vezeték nélküli kommunikációs csatornán továbbítani nem. A harmadik ok, ami miatt egyedi hardver építése mellett döntöttünk, a megvásárolható eszközök magas ára.

A rendszert több szempont alapján terveztük meg. Fontos volt számunkra a könnyű bővíthetőség, stabilitás és a vezeték nélküli kommunikációs lehetőségek kihasználása. További fontos szempont, hogy a létező megoldásoknál jóval olcsóbb eszközt készíthessünk el.

3.1 Mérőrendszer felépítése

A mérőrendszer egy kulcsfontosságú tulajdonsága a könnyű bővíthetőség, ezért a mérőrendszer felépítése szerver-kliens kialakítást kapott.



7. ábra A mérőrendszer felépítése

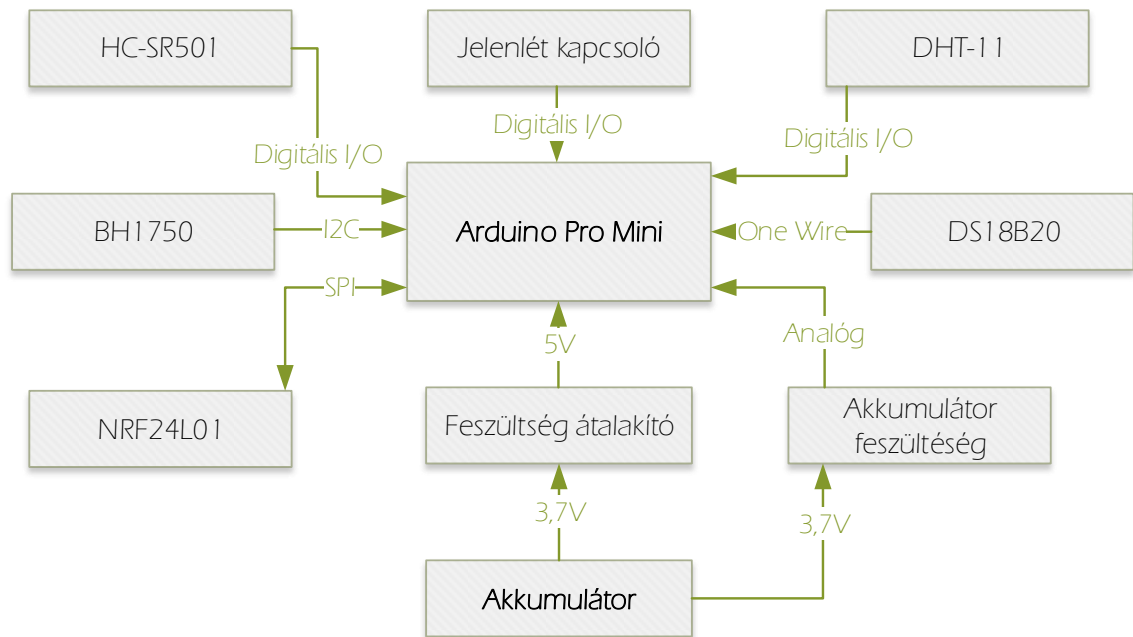
Az architektúra (7. ábra) három rétegre bontható, ahol az első szintet a mérőegységek képviselik. A mérőegységek feladata a szenzorokkal való kapcsolattartás, a szenzoradatok kiolvasása és továbbítása. A középső réteg a szerver modul, ami lényegében egy hídnak tekinthető, így csak adattovábbítást végez. A harmadik réteg maga az adattároló szerver és a hozzá kapcsolódó interface szolgáltatás. Az adattároló szerver egy személyi számítógépen fut, és lehetőséget ad az adatok vizuális megjelenítésére és kimentésére is.

A rétegek közötti kapcsolattartást vezeték nélküli kommunikáció biztosítja. A mérőmodulok RF (rádió frekvenciás) adó-vevővel vannak ellátva. Az átjátszó híd modul két vezeték nélküli protokoll-t valósít meg, egyik az előbb említett RF, a másik pedig a WIFI szabvány szerinti. Felvetődhet a gondolat, hogy az egyes mérőmodulok miért nem rendelkeznek rögtön WIFI kommunikációval. Ennek oka az, hogy az RF adó-vevők aránylag könnyen használhatóak és olcsók. A WIFI kommunikációval is rendelkező adatkoncentrátor egység tartja a kapcsolatot a központi adatrögzítő PC-vel, így feladata a mérőmodulokról RF-en fogadott adatok továbbküldése a PC felé, ahol eltárolásra kerülnek.

Fontos azonban megjegyezni, hogy a WIFI hálózaton lévő PC és híd modul között HTTP protokollon keresztül történik az adattovábbítás így nem igényel további beállítást a router-ben, ha a rendszert egy meglévő hálózaton működtetjük. Az adattárolást végző szerver PC-n így egy webkiszolgálóra és egy MySQL adatbázisra van szükségünk. Az adatbázis előnye hogy php-ban készített weboldal segítségével könnyen be- és kiolvashatók a mérési adatok.

3.2 A megépített mérőegység

A mérőegység egy általánosan elterjedt mikrovezérlő panel (Arduino Mini Pro) köré épített szenzorokat és vezeték nélküli kommunikációs modult foglal magában egy nyákra integrálva (8. ábra). A rendszer könnyű telepítési tulajdonsága, hogy egy 3,7V feszültségű akkumulátor biztosítja a panel számára szükséges tápellátást.



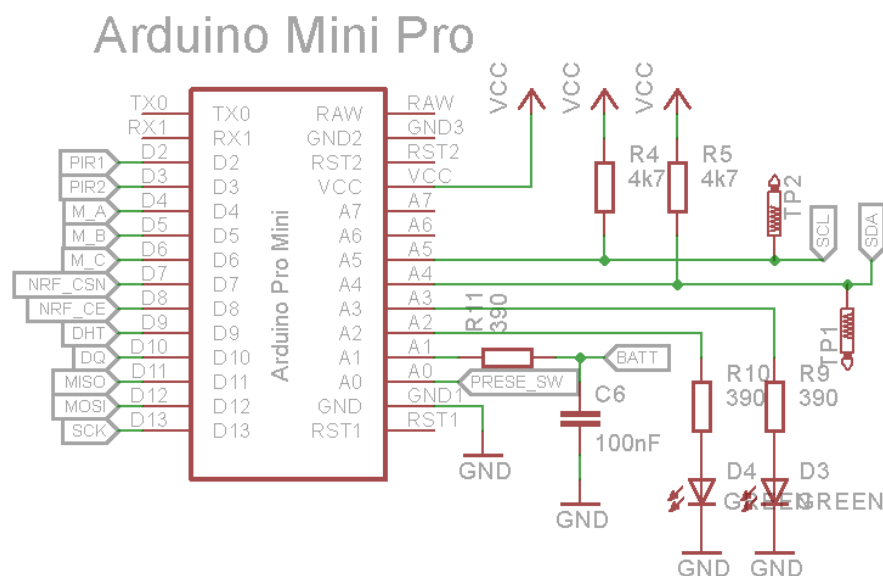
8. ábra A mérőegység blokk diagramja

Az Arduino Mini Pro használatának előnye a nyílt hardverben és szoftverben rejlik, továbbá, hogy az Arduino támogatottsága hatalmassá vált az utóbbi években, mind a hobbi mind a komolyabb fejlesztők körében.

2. táblázat A fejlesztőeszköz főbb tulajdonságai

Mikrovezérlő	ATmega328P
Működési feszültség	5V
Tápfeszültség	5 - 12 V
Digitális I/O	14 (ebből 6 PWM kimenet)
Analóg bemenet	8
Flash memória	16 KB (ebből 2 KB a bootloaderre)
SRAM / EEPROM	1 KB / 512 bytes
Órajel	16 MHz

A fejlesztőrendszeren elhelyezett Atmel 328P mikrovezérlő rendelkezik digitális és analóg portokkal is (további paramétereket ld. a 2. táblázatban), így analóg és digitális jelek kezelésére is kiváló választás. A mikrovezérlős modulra csatlakozókon keresztül illeszthetők a szenzorok (9. ábra). Továbbá a mérőmodulon elhelyezésre kerültek a szenzorokhoz szükséges illesztő alkatrészek, így például az I2C busz felhúzó ellenállásai és az akkumulátor feszültségmérő stabilizáló kondenzátora. A PIR szenzorok állapotát két LED jelzi.



9. ábra A szenzorok bekötése a mikrovezérlőbe

Felhasznált szenzorok

A 8. ábra blokk diagramján látható, hogy a rendszer több szenzor fogadására képes, ezért a továbbiakban a szenzorok tulajdonságaival és beépítésével foglalkozunk. A rendszer különlegessége, hogy lehetőséget ad több egyforma szenzor egyidejű csatlakoztatására. Ez tipikusan a fényérzékelő szenzorra igaz. Ugyan a gyártó lehetőséget ad az I2C szolgáló cím kiválasztására, de ez maximum két variáció lehet. A rendszert 4 darab fényérzékelő szenzorral szeretnénk működtetni, ezért egy kiegészítő áramkörre van szükség, így a továbbiakban erre is részletesebben kitérünk.

DHT-11 páratartalom mérő

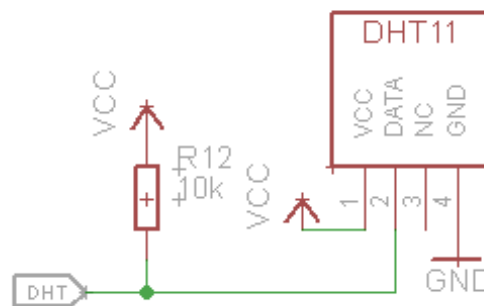
A DHT11 digitális hőmérséklet és páratartalom mérő szenzor 4 érintkezős tokozásban érhető el. 3.3V - 5.5V közötti tápfeszültségről üzemeltethető. A szenzor saját, egy-vezetékű protokollal rendelkezik (10. ábra).

Az érzékelő kapacitív páratartalom mérővel és termisztorral rendelkezik. A mért adatok lekéréséhez nincs szükség analóg bemenetre, ugyanis a szenzor egy darab digitális porton képes kommunikálni a mikrovezérlővel. A szenzor két másodpercenként biztosít új mérésre lehetőséget.

További paraméterek:

- 5V táp és adat I/O
- 2.5mA max. áramfogyasztás a mérési konverzió ideje alatt
- Páratartalom mérés: 20-80% relatív érték, 5%-os pontosság
- Hőmérsékletmérés: 0-50°C, $\pm 2^\circ\text{C}$ pontosság
- 1 Hz mintavételezés
- Méret: 15.5mm x 12mm x 5.5mm
- Csatlakozás: 4 vezeték 0.1" lábtávolság

A DHT 11 szenzor az adatvezeték mellett további VCC és GND lábbakkal rendelkezik, amelyek bekötése elengedhetetlen a helyes működéshez.



10. ábra A DHT-11 szenzor illesztése a mikrovezérlőhöz

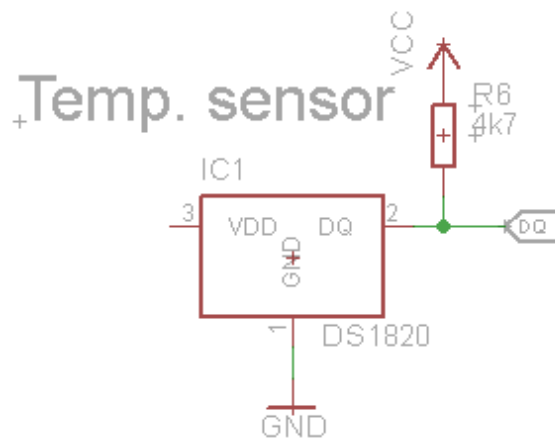
DS18B20 hőmérséklet mérő

A DS18B20 digitális hőmérő mérési pontossága 9 bit és 12bit között állítható. A kommunikáció egy vezetékes One Wire buszon történik. Minden DS18B20 szenzor egy egyedi 64 bites szériaszámmal rendelkezik, így ugyanarra a buszra fűzhető fel több hőmérő [6].

Fontosabb tulajdonságok:

- 64 bites egyedi szériaszám
- 3V – 5V-os tápellátás
- Tág mérési tartomány -55°C és $+125^{\circ}\text{C}$ között
- $\pm 0.5^{\circ}\text{C}$ pontosság -10°C és $+85^{\circ}\text{C}$ között
- 9 bit és 12 bit között állítható felbontás
- Konvertálási idő 12 bites Word típusba 750ms alatt

A szenzor hardveres illesztéséhez egy-egy további felhúzó ellenállásra van szükség az adat- és tápvezetékre csatlakoztatva. A OneWire busz különlegessége, hogy működéséhez csak egy GND és egy táp vezetékre van szükség (11. ábra). A kommunikáció a tápvezetéken történik, melyen „parazita táp”-ként az eszköz energiaellátása is biztosított.



11. ábra Hőmérséklet érzékelő szenzor bekötése

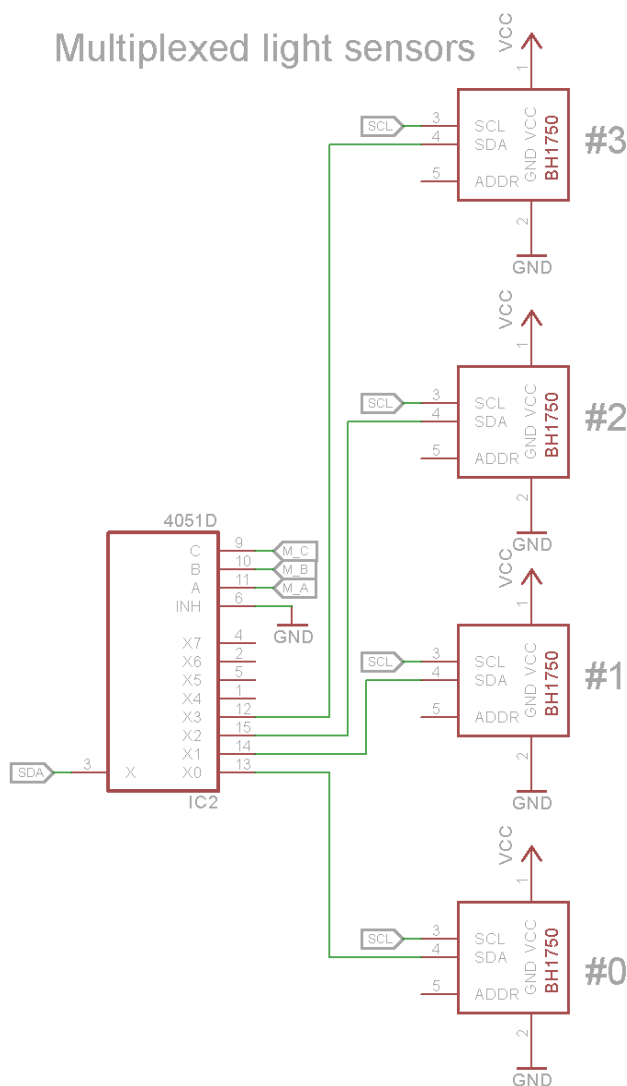
BH1750 fényérzékelő

A BH1750FVI egy környezeti fénymérő egység I2C interface-el. A szenzor nagy felbontással, 1-65535lx között képes mérni [7].

Tulajdonságok:

- I2C buszon történő kommunikáció
- Illeszkedik az emberi szem spektrumához
- Megvilágítás-digitális átalakító
- Nagy felbontás
- 50Hz-60Hz szűrése
- Független a fényforrásoktól
- Két választható I2C cím
- Infrafény zavaró hatása kicsi

A fényérzékelés fontos szerepet játszik a mérésben, ezért a fényviszonyoknak több helyen is mérhetőnek kell lenniük. A mérőmodult ezért 4 fényszenzorral lehet felszerelni az adott méréshez igazodva. A megtervezett panel csatlakozókkal van ellátva, így a szenzorok akár távolra is elvezethetők vezeték segítségével, tehát mérhető akár a külső és belső fényerő egyaránt.

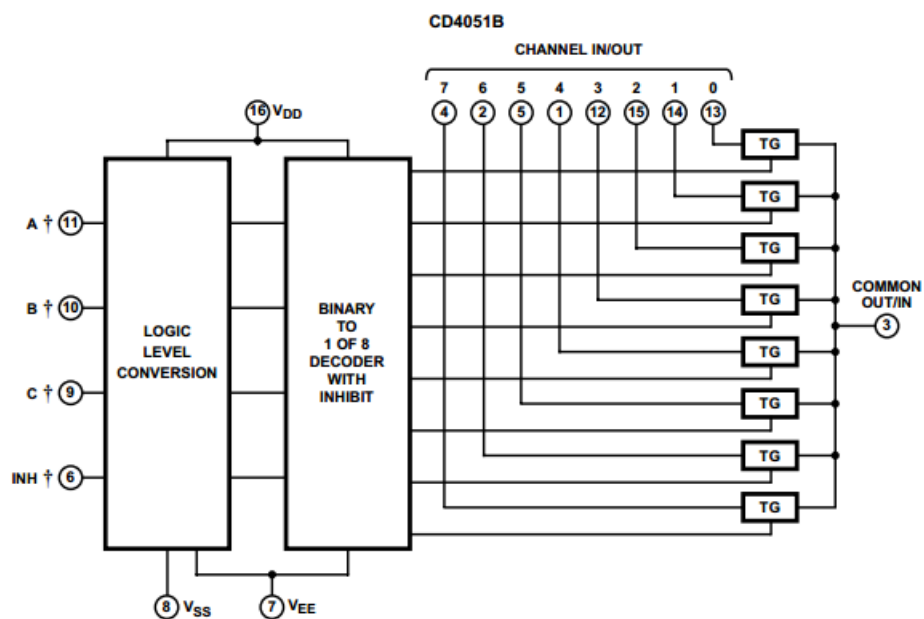


1. ábra Multiplexelés megvalósítása

A hardveres kialakítást a felsorolt okok miatt multiplexelve alakítottuk ki (12. ábra). A szenzor I2C buszon keresztül kommunikál a mikrovezérlővel. Az I2C busz két vezetékkel használ (SDA és SCL). Az SCL vezeték adja a busz közös órajelét, míg az SDA vezetéken a kérés és válasz adatok mozognak az SCL vezetékhöz szinkronizálva. Az I2C buszt úgy alakították ki, hogy az adott kommunikációval rendelkező szenzorok egy buszra

illeszthetők és egy címmel elérhetők legyenek. A szenzor címét a gyártók alakítják ki és a hardverbe kódolják. I2C esetén tipikusan 3 bites cím használható, azonban a kiválasztott fénymérő szenzor csak az utolsó címbit módosítását engedi, tehát maximum két egyedi cím alakítható ki egy buszra, ami egyben korlátozást is felvet.

A probléma forrása az, hogy a mikrovezérlő egy I2C busszal rendelkezik és az adott szenzorból csak kettő helyezhető el egy buszra. Négy szenzor felhasználása esetén el kell kerülnünk a címütközést. A címütközés megoldása lehet egy szoftveres I2C busz létrehozása, ami a processzor terhelést növeli és nehezíti a többi kód ütemezését. Ezért a jelen projektben hardveres megoldást választottunk. Az I2C vezeték adat (SDA) lábának multiplexelése ugyanis könnyen megoldja a felmerült problémát. A multiplexelésnek köszönhetően a szenzorok akár ugyanolyan címmel is rendelkezhetnek, ezzel is megkönnyítve a programozást.



2. ábra A multiplexer belső felépítése

A multiplexeléshez 4051-es analóg multiplexer IC-t választottuk ki. A multiplexeren három vezérlőláb segítségével választható ki, hogy a beérkező maximum 8 vezetékből melyiket csatoljuk a közös vezetékre. Az IC további előnye hogy a kommunikáció kétirányú, így az I2C SDA vezetéket teljes értékűen multiplexelhetjük anélkül, hogy foglalkoznánk az adatáramlás irányával.

A 4051 típusú IC-t felhasználva (13. ábra) a mikrovezérlő felül egy, a szenzor oldalon 8 adatvezeték érhető el. Az I2C órajel (SCL) vezetéket nem szükséges multiplexelni ugyanis

az órajel beérkezése nem okoz problémát a szenzornak, mivel csak a kiválasztott szenzorhoz érkezik be érvényes adat is az órajel mellé.

HC-SR501 mozgásérzékelő

A HC-SR501 mozgásérzékelésére használható infra modul. A TTL szintű feszültség kimenetének és áramfelvételének köszönhetően könnyen illeszthető tipikusan használt mikrovezérlőkhöz.

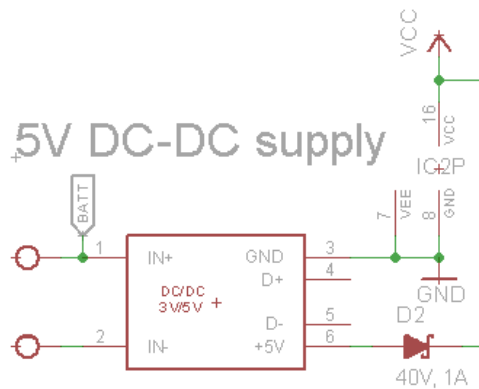
Fontosabb paraméterek:

- Dimenziók: 3.2cm x 2.4cm x 1.8cm
- Infra érzékelő vezérlő panellel
- Állítható érzékenység és szint tartás
- Működési feszültség: DC 4.5V- 20V
- Áramfelvétel: <60uA
- Feszültség kimenet: 3.3V TTL kimenet
- Érzékelési távolság: 3 – 7m
- Érzékelési tartomány: <140°
- Működési hőm: -20-+80°C

A passzív infravörös érzékelő (röviden: PIR) a tárgyak infravörös kisugárzását méri, tehát passzív működéséből adódóan nem bocsájt ki magából fényt a méréshez. A szenzor méri az adott helység infravörös kisugárzását, ha ez a mennyiség változik az alapállapothoz képest, akkor jelzést küld. A két trimmer potenciométer az érzékenység és az visszaadott jel időbeni nagyságát szabályozza.

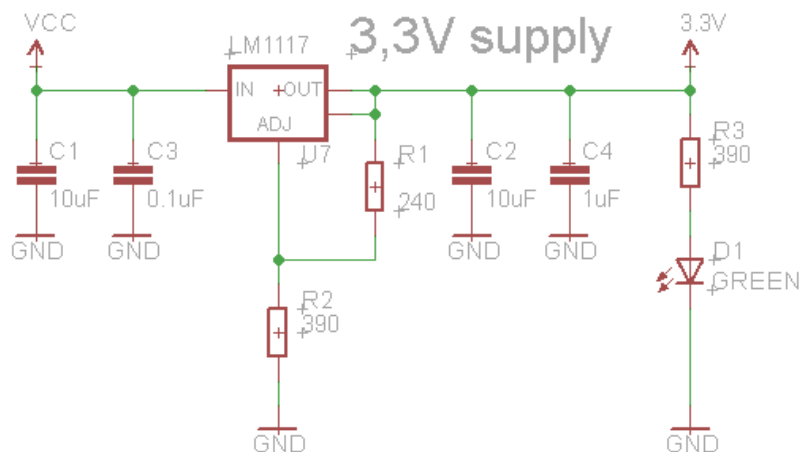
Tápellátás megvalósítása

A mérőrendszerhez tervezett mérőegység a korábban már bemutatott szenzorokat csatlakoztatja a modulon található AVR alapú Arduino Pro Mini-hez. A mikrovezérlő 5V-os tápellátást igényel, ehhez stabilizált feszültséget kell kialakítani.



3. ábra Tápellátás a kapcsolási rajzon

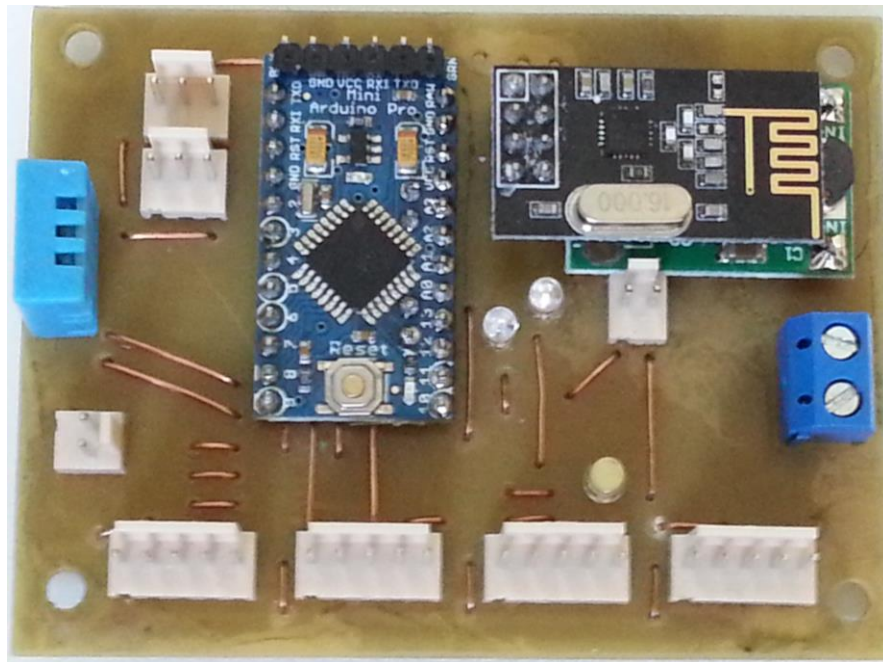
A mérőmodul egyik fő előnye, hogy vezeték nélküli kapcsolattal rendelkezik, így a tápellátást is akkumulátorral történik. Az általunk használt Li-Ion akkumulátor 3,7V feszültségű. Ahhoz, hogy egy cella csatlakoztatható legyen a mikrovezérlőhöz, egy „step-up” konverterre van szükség. Az általunk használt feszültség konverter (14. ábra) biztosítja a stabil tápellátást a mérőmodul számára. A kapcsolást kiegészítettük egy védődiódával is, hogy a visszatáplálástól megkíméljük.



4. ábra 3,3V-os tápellátás kapcsolása

A mikrovezérlővel ellentétben az RF modul 3,3V-os, ezért egy 3,3V feszültségű stabilizált tápellátást építettünk ki a panelen (15. ábra). A kapcsoláshoz az LM1117 típusú feszültség regulátort használtuk és a kimeneten egy LED-et is elhelyeztünk, amely a tápfeszültség megléte esetén világít. A kimenetre kapcsolt különböző értékű kondenzátorok a kimeneti jel simítását végzik ugyanis a kommunikáció modul nagyon érzékeny a tápellátás ingadozására.

A megépített mérőmodul a 16. ábrán látható.



5. ábra Mérőpanel felülnézetből

3.3 Adatkoncentráló központi modul

Az adatkoncentráló egység feladata a mérőmodulok adatainak fogadása és tárolása. A fogadó oldalon egy RF adó-vevő található. Az adattárolást WIFI kapcsolat segítségével PC-n végezzük. A WIFI modul megfelel a mai b\g\n szabványoknak. A két vezeték nélküli egység közti kapcsolatot szintén egy Atmel AVR alapú Arduino Nano hozza létre.

NRF24L01 adó-vevő

Az nRF24L01 egy chipbe épített 2.4GHz-es adóvevő, beágyazott kommunikációs protokollal. A tervezés során az ultra alacsony fogyasztást tartotta szem előtt a gyártó. A beépítéshez néhány külső passzív alkatrész szükséges [8].

- 2.4GHz-es frekvencia
- Maximum adatátviteli sebesség 2Mbps
- Ultra alacsony fogyasztás
- 11.3mA TX 0dBm kimenő teljesítmény esetén
- 12.3mA RX 2Mbps adatátviteli sebesség esetén
- 900nA kikapcsolt állapotban
- 22μA készenléti állapotba
- Beépített feszültség stabilizátor
- 1.9 - 3.6V közötti tápfeszültség
- Automatikus csomagkezelés
- 6 adatcsatornás MultiCeiver™ kommunikáció
- ±60ppm 16MHz kristály
- 5V toleráns bemenetek

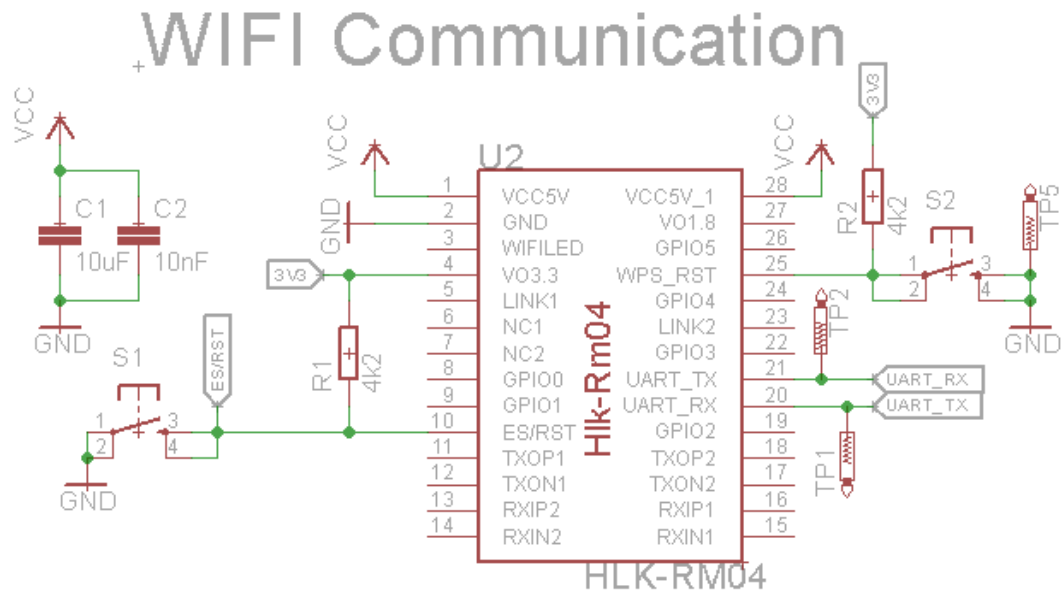
HLK-RM04 WIFI modul

A HLK RM04 egy olcsó beágyazott UART-Ethernet-WIFI modul. Az eszköz nagyon sok WIFI szabványt és autentikációt támogat. További előnye, hogy kliens, hozzáférési pont és router módba is konfigurálható [9].

A WIFI tulajdonságai:

- 2.4GHz-es 802.11 b/g/n kompatibilis
- WIFI kliens, AP, Router mód
- WPS/WDS támogatás
- 1200-500000bps adatátviteli sebesség
- WEP/WPA-PSK/WPA2-PSK autentikáció támogatás
- PPPOE/TCP/UDP/DDNS/DHCP/DNS/HTTP támogatás
- AT+ parancs támogatás
- Webes vagy Serial konfigurálási lehetőség

A modul 5V-os tápellátással rendelkezik, de a bemeneti lábak 3,3V-os jelszinten működnek. A mikrovezérlőként használt Arduino Nano 5V-os rendszerű így az adatbuszra feszültségosztó ellenállásokat helyeztünk, így illesztve egymáshoz a két jelszintet. Az Wifi mellé két gomb is elhelyezésre került, amikkel alapállapotba lehet visszaállítani az eszközt. Jelen projektben a modul vezeték nélküli képességét szeretnénk kihasználni így csak az UART kommunikáció kötöttük be, a vezetékes Ethernet kommunikációt mellőztük.



17. ábra A Wifi modul bekötési rajza

4 MÉRÉST VÉGZŐ BEÁGYAZOTT SZOFTVER

A megtervezett mérőrendszer szoftveres kiépítési is több rétegre bontható. A mérő és adatkoncentrálor egységen egy-egy beágyazott szoftver fut. A PC oldali adatmegjelenítő és tároló egység pedig egy PHP-MySQL alapú webszerver. A szoftveres kommunikáció mindenhol vezeték nélküli. A mérő egységek RF-en míg a PC oldalra már WIFI kommunikáció segítségével érkeznek az adatok.

4.1 Mérőegység szoftvere

A mérőegység a korábban említett Arduino platformon fut, melynek erőssége a könnyű és gyors programozhatóság. Az eszköz C++ alapú programozási nyelvet támogat.

A szoftver „Setup” részében történik meg az eszközök felkészítése kommunikációra és mérésre. Az szenzorok inicializálása után egy indító üzenet jelzi a mérőegység rendelkezésre állását, ezt az üzenetet RF kapcsolaton keresztül továbbítja.

```
void setup()
{
    Serial.begin(115200);

    pinMode(PIR_PIN, INPUT);
    pinMode(PRE_PIN, INPUT);
    pinMode(PIRLEDPIN, OUTPUT);

    lightMeter.begin();
    dht.setup(DHT_PIN); // data pin 2

    setupRF();

    transmitter.txPL("START|System running...\n");
    transmitter.send(FAST);

    sendtimer.setInterval(1000, sendResult);
    measuretimer.setInterval(1000, measure);
}
```

18. ábra A mérőegység inicializálása

A mérőegység működését két szoftveresen definiált időzítő látja el melyeket a „setup” végén láthatunk. A „measuretimer” időzítő végzi el az egyes szenzorok mérési eredményeinek begyűjtését. A „sendtimer” időzítő pedig az utolsó mérési adatokat továbbítja a központi adatgyűjtő szerver felé.

Az időzítőben definiált függvények 1 másodpercenként futnak le, így a modul minden másodpercben végez mérést, azonban egyes szenzorok esetében nem szükséges a túl gyakori mintavételezés.

```
void measure()
{
    if(DEBUG) Serial.println("Measure");

    pirState = digitalRead(PIR_PIN); // read input value
    digitalWrite(PIRLEDPIN,pirState);

    presence_sw = digitalRead(PRE_PIN); // read input value

    int battery = analogRead(1);
    battVolt = (battery / 1023.0) * Vcc; // only correct if Vcc = 5.0 volts

    lux = lightMeter.readLightLevel();

    sensors.requestTemperatures();
    dsTemp = sensors.getTempCByIndex(0);

    humidity = dht.getHumidity();
    dhttemp = dht.getTemperature();
}
```

19. ábra Egy mérési ciklus

A mérés során többféle adat is begyűjtésre kerül, ezek: mozgásérzékelő állapota, jelenlét kapcsoló állapota, akkumulátor feszültsége, fényérzékelő modul mérése, két fajta szenzor hőmérsékleti mérése, páratartalom. Az egyes érzékelő adatok globális változókba kerülnek ahonnan küldéskor kiolvashatók.

```

int i = 0;
void sendResult()
{
    i++;
    if(DEBUG)
    {
        transmitter.txPL("#="+ String(DATABASE_ID) + "&");
        transmitter.send(FAST);

        if(i % 60 != 0)
        {
            transmitter.txPL("presw="+ String(presence_sw) + "&");
            transmitter.send(FAST);
            transmitter.txPL("lx="+ String(lux) + "&");
            transmitter.send(FAST);
            transmitter.txPL("pir="+ String(pirState) + "\n");
            transmitter.send(FAST);
        }
        else
        {
        }
    }
}

```

20. ábra Mérési adatok továbbítása

Az adatok továbbítása a „sendResult” függvény hívásakor történik meg. A küldés során nem minden adatot küldünk másodpercenként, ugyanis a mérés során vannak nagy időállandójú paraméterek is. A szoftver minden 60. küldésnél elküldi további szenzorok adatait is.

3. táblázat Szenzor adatok neve és küldési gyakorisága

Szenzor	Küldési gyakoriság
Jelenlét kapcsoló (presw)	Másodpercenként
Fényerő (lx)	Másodpercenként
Mozgásérzékelő (pir)	Másodpercenként
Akkumulátor feszültség (bat)	Percenként
DS hőmérséklet (dst)	Percenként
DHT hőmérséklet (dhht)	Percenként
Páratartalom (dhth)	Percenként

A szenzoradatok mellett a modul előre beállított azonosítószáma is továbbításra kerül, az adatok könnyen azonosíthatósága érdekében. A mérési adatokból és az ID-ből elkészített szöveges üzenet a következő felépítésű:

#=1&lx=123&presw=1&bat=3.45&dst=25.625&dhtt=27&dhth=40&pir=1

Az adatsor felépítésének előnye hogy HTTP POST kérések összeállításánál rögtön felhasználó, tehát nem igényel további „string” műveleteket.

4.2 Adatkoncentrátor szoftvere

Az adatkoncentráló egység szoftvere fogadja a mérőegység által küldött mérési adatokat. Az adatok WIFI hálózaton keresztül egy PC futó webszerverhez érkeznek. A webszerver szabványos HTTP protokollt használ.

A modulon található WIFI egység rendelkezik egy böngészőből is elérhető konfigurációs felülettel. Ezen a felületen állíthatjuk be, hogy a UART portjára érkező adatok milyen IP címre és portra kerüljenek továbbításra.

The screenshot shows the configuration interface for the HLK-RM04 Serial2Net Settings. The interface is in English and includes a sidebar with navigation options: HLK-RM04, Serial2Net Settings, Advance Settings, Serial2Net UART 2 Settings, and Administration. The main content area displays the following settings:

Current	Updated
NetMode:	WIFI(AP)-SERIAL
SSID:	SG-Network
Encrypt Type:	WPA2 AES
Password:	00112233
IP Address:	192.168.11.254
Subnet Mask:	255.255.255.0
Serial Configure:	115200,8,n,1
Serial Framing Lenth:	64
Serial Framing Timeout:	10 milliseconds
Network Mode:	client
Remote Server Domain/IP:	192.168.11.1
Locale/Remote Port Number:	80
Network Protocol:	tcp
Network Timeout:	10 seconds

6. ábra WIFI modul konfigurációs felülete

A WIFI eszköz beállításainál a WIFI AP funkciót használjuk, így nincs szükség külön routerre, hogy a számítógépek egy hálózatba kerüljenek. Továbbá a WIFI modul kliensként kapcsolódik egy távoli IP címhez amennyiben az UART porton adat érkezik. A távoli IP cím esetünkben a webszervert futtató számítógép IP címe kell lennie.

Az általunk használt fontosabb beállítások:

- NetMode: WIFI(AP) – SERIAL
- SSID: SG-Network
- Serial Configure: 115200,8,n,1
- Remote Server(Domain/IP): 192.168.11.1
- Remote Port Number: 80

Az Arduinon futó szoftver

A mikrovezérlőn futó szoftver első lépésben konfigurálja a WIFI és RF kommunikáció adatcsatornáit. A modulon létrehoztunk egy szoftveres soros portot is, ami hibakeresési lehetőségeket biztosít számunkra. A WIFI modult az előzőleg is használt 115200 bps-re konfiguráltuk. Végül az RF kapcsolat beállítását végeztük el.

```
SoftwareSerial mySerial(2, 3); // RX, TX

nRF24L01p transmitter(7, 8); //CSN, CE
String message;

void setup() {
  mySerial.begin(9600);
  mySerial.println("Hello, world?");
  Serial.begin(115200);
  setupRF();
}

void setupRF()
{
  SPI.begin();
  SPI.setBitOrder(MSBFIRST);
  transmitter.channel(13);
  transmitter.RXaddress("SG-Net");
  transmitter.TXaddress("SG-Net");
  transmitter.init();
}
```

7. ábra Adatkoncentrátor inicializálása

Az RF adóvevőhöz először inicializálnunk kell az SPI buszt, ezután állítható be a csatorna száma és az adó által használt két cím.

```
String readed = "";
transmitter.read();
transmitter.rxPL(readed);
message += readed;
if(message.endsWith("\n") && !message.startsWith("#"))
{
  message="";
}
else if(message.endsWith("\n") && message.startsWith("#"))
{
  mySerial.print(message);
  postData(message);
  message="";
}
}
```

8. ábra Mérési adatok fogadása

A mikrovezérlős program futás során fogadja az UART porton érkező adatokat egészen addig, amíg sortörés karakter nem érkezik. Az üzeneteket az RF adóvevő 32 byte-os csomagkorlátja miatt nem egy stringként kapjuk meg, így a beérkező csomagok összefűzéssel tárolódnak a memóriában. A sortörés karakter jelzi az üzenet végét, és

amennyiben üzenetnek megvan az indító # karaktere is az üzenet teljesnek számít, továbbítható a webszerver felé.

Egy üzenet beolvasásának végén az üzenet feldolgozzuk a postData függvény segítségével.

```
void postData(String SensorData)
{
  /*
  Serial.println("POST /SG_Logger/sensordatabulk HTTP/1.1");
  Serial.println("Host: 192.168.11.1");
  Serial.println("User-Agent: Arduino/1.0");
  Serial.println("Connection: close");
  Serial.println("Content-Type: application/x-www-form-urlencoded");
  Serial.print("Content-Length: ");
  Serial.println(SensorData.length());
  Serial.println();
  Serial.print(SensorData);
  Serial.println();
  */
}
```

9. ábra Fogadott mérési adatok POST kéréssé alakítása

Az üzenetek feldolgozása egy HTTP Post kérés összeállítását jelenti. A kérésnek két része van a tartalom és egy fejléc. A fejléc estünkben nagyrészt statikus szöveg, a következőképp értelmezhető:

- POST /SG_Logger/trunk/sensordata/1 HTTP/1.1 //Kérés elérési útja
- Host: 192.168.1.254 //Kérés IP-címe
- User-Agent: Arduino/1.0 //Kérést kezdeményző IP cím
- Content-Type: application/x-www-form-urlencoded; //Tartalom típusa
- Content-Length: 27 //Tartalom hossza
- //Üres sor
- SEN=DS temp sensor&VAL=1234 //Kérés tartalma

Az üzenetben található tartalom rész mindig az aktuális mérési adatsort tartalmazza, és elküldése után nem várunk vissza üzenetet a webszervertől.

4.3 PC oldali webszerver, adatbázis és weboldal

A PC oldali szoftver kialakítást Apache-PHP-MYSql csomagokkal valósítottuk meg. A webszerveren futó szoftvert PHP nyelven írtuk meg. A PHP szoftver elkészítéséhez egy micro framework-ot használtunk, amely a Fat-Free Framework(röviden: F3) nevet viseli. A Fat-Free Framework egy egyszerűen használható mikro keretrendszer robusztus web alkalmazások készítéséhez. Az keretrendszer fejlett URL routing lehetőségekkel bír, emellett beépített támogatás található több népszerű adatbázishoz, illetve nyelvtámogatás készítéséhez is.

SG Logger weboldal

Az elkészített weboldal két fő fájlból áll az index.php-ből, és egy config.ini fájlból. Az F3 segítségével először az adatbázis kapcsolatot hozzuk létre. A létrehozott kapcsolatot egy munkamenet változóban tároljuk, elérhetővé teszi az adatbázis kapcsolatot függvényeken belül is.

```
$db = new DB\SQL(
    'mysql:host=' . $f3->get('DB_host') . ';port=' . $f3->get('DB_port') . ';dbname=' . $f3->get('DB_name'),
    $f3->get('DB_user'),
    $f3->get('DB_pass')
);

new Session();
$f3->set('DB', $db);
```

10. ábra MySQL adatbázis inicializálása

Az adatbázis inicializálásánál használt változókat a keretrendszer a config.ini fájlból nyeri, így nem kell közvetlen a kódban megadnunk ezeket az adatokat.

```
$f3->map('/serverinfo', 'sg_serverinfo');
$f3->map('/sensor', 'sg_sensor');
$f3->map('/sensor/@sensorid', 'sg_sensorbyid');
$f3->map('/sensordatabulk', 'sg_sensordatabulk');
$f3->map('/sensordata/@deviceid', 'sg_sensordata');
$f3->map('/device', 'sg_device');
$f3->map('/device/@deviceid', 'sg_devicebyid');
$f3->map('/user', 'sg_user');
$f3->map('/user/@userid', 'sg_userbyid');
```

26. ábra Fat-Free Framework inicializálása

Az adatbázis kapcsolat felépítése után definiáltuk az egyes „routing”-okat ahol kihasználtuk a keretrendszer OOP támogatását. A keretrendszer „routing” lehetőségei

segítségével könnyebben értelmezhető és használható URL címeket kialakítani az alkalmazásban. Ez a funkció a szabványos REST API-k elkészítéséhez nyújt segítséget. A REST API egy szerver alkalmazás, amely távoli kliensek kéréseit teljesíti. A REST szerver minden PHP objektumnál a HTTP protokoll által definiált GET, POST, PUT, DELETE funkciókat valósítja meg. A kéréstípusok megszabják a beérkezett adatokkal végzendő műveletet.

4. táblázat REST URL-ek funkciói

Útvonal	POST létrehozás	GET olvasás	PUT frissítés	DELETE törlés
/sensor	Szenzor létrehozása	Szenzorok lekérése	Szenzorok frissítése	Szenzorok törlése
/sensor/@sensorid	-	Szenzor lekérdezése	Szenzor frissítése	Szenzor törlése
/sensordatabulk	Tömeges szenzor adat beírása	-	-	-
/sensordata/@deviceid	Szenzor adat beírás	-	-	-
/device	Eszköz létrehozása	Eszközők lekérdezése	Eszközők frissítése	Eszközők törlése
/device/@deviceid	-	Eszköz lekérdezése	Eszköz frissítése	Eszköz törlése
/user	Felhasználó létrehozása	Felhasználók lekérdezése	Felhasználók frissítése	Felhasználók törlése
/user/@userid	-	Felhasználó olvasása	Felhasználó frissítése	Felhasználó törlése

Az egyes URL címekhez külön osztályokat rendeltünk. Az osztályokban a kéréseknek megfelelő tagfüggvényeket hoztunk létre, az F3 támogatásával. A létrehozott tagfüggvényekben deklarálnunk kell az egyes kéréstípusok feladatát.

```

class sg_sensor {
    function get($f3) {
        ...
    }
    function post($f3) {
        ...
    }
    function put() {
        ...
    }
    function delete() {
        ...
    }
}

```

27. ábra REST PHP objektum felépítése

Az általunk létrehozott rendszer MySQL adatbázisban tárolja az adatokat így a keretrendszerbe épített adatbázis támogatás előnyünkre vált. A framework támogatja a SQL parancsokkal való művelet végrehajtást, de emellett „Mapper”-ek segítségével is tudunk műveleteket végrehajtani. A „Mapper”-ek objektumokként kezelik az adatbázis tábláit, így az OOP eszközeivel érhetjük el őket.

```

$sensordata = new DB\SQL\Mapper($db, 'sg_sensordata');
$sensordata->sg_sensor_id = $one_sens->id;
$sensordata->sg_device_id = $deviceid;
$sensordata->value = $sensorvalue;
$sensordata->save();
echo "OK";

```

11. ábra Új adat mentése adattáblába

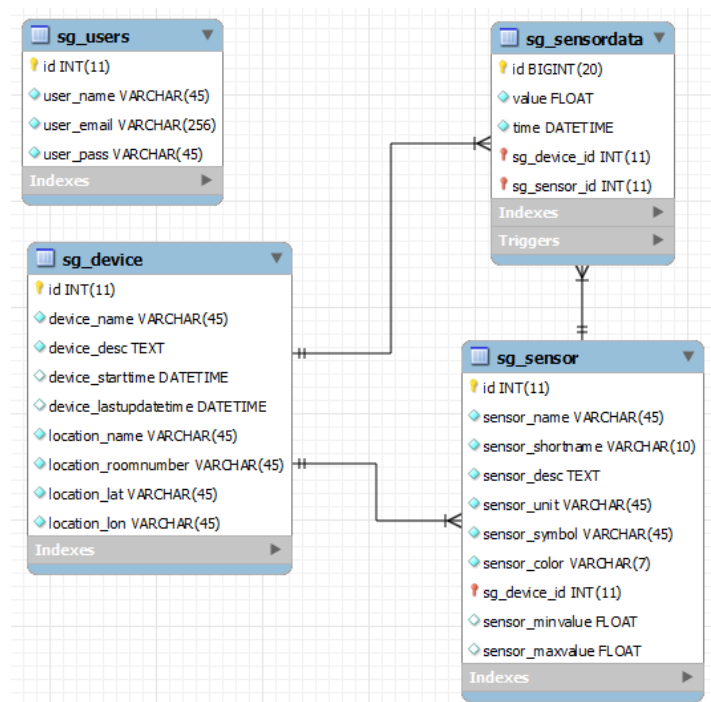
A kliens eszközök több úton is frissíthetik a szenzorok adatit. Lehetséges egy szenzor és több szenzor egyidejű frissítése is. Az általunk preferált módszer a több szenzor egyidejű frissítése (/sensordatabulk), így gyorsabb művelet végrehajtást érhetünk el, ezzel csökkentve a szerver terhelését.

PC oldali adatbázis és a weboldal felépítése

A kifejlesztett webalkalmazáshoz MySql adatbázis is tartozik. Az adatbázis kialakítása során törekedtünk az egyszerű felépítésre.

Az adatbázisban 4 tábla található:

- sg_users: A webes felülethez tartozó felhasználókat tárolja
- sg_device: A kliens eszközök adattáblája
- sg_sensordata: A szenzorok mérési adatainak táblája
- sg_sensor: Az eszközökhöz tartozó szenzorok adattáblája



29. ábra Adatbázis felépítése

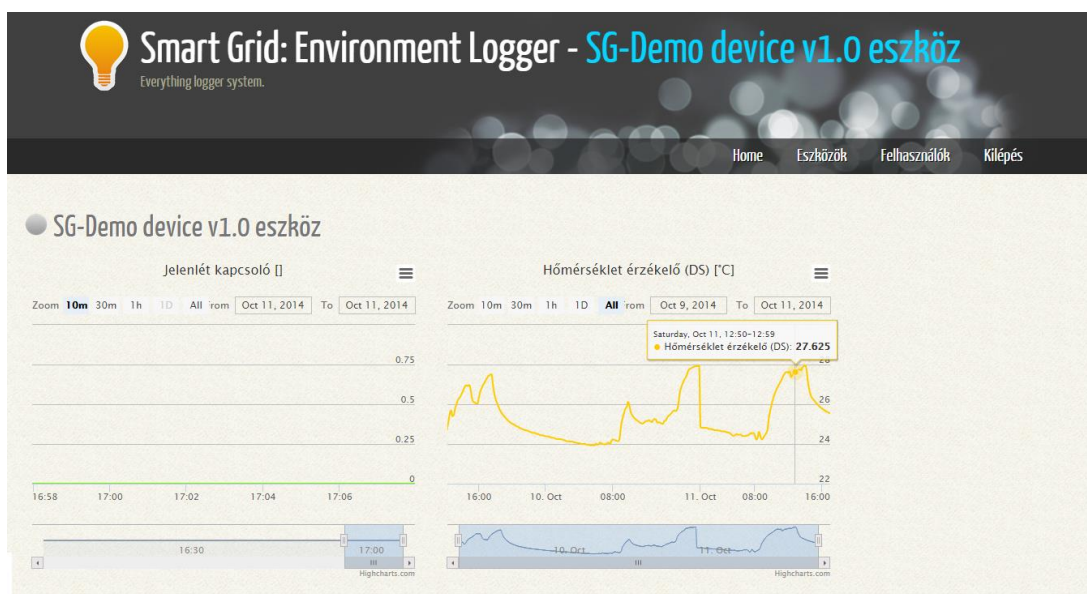
Az eszközök alap információ mellett lehetőség van a mérés helyének eltárolására is. Továbbá a megjelenítő grafikonok görbéinek színét és minimum maximum értékét is lehetőségünk van testre szabni.

Az általunk elkészített weboldal nem csak kapcsolatot tart az eszközökkel, hanem azok menedzselésére is lehetőséget biztosít. A felhasználóknak joga van eszközöket felvenni, és az egyes eszközökhöz szenzorokat definiálni. A későbbiekben e szenzorok aktuális értékeit frissíthetjük a kliens eszközökről.



12. ábra Az eszközök megjelenítése a weboldalon

A weboldal felső részén található menük segítségével is tudunk navigálni a weboldalon. Az eszközök menü alatt találhatjuk meg a rendszerben regisztrált eszközöket. Az oldal lehetőséget ad új eszközök felvételére is. A felvett eszközökről néhány fontosabb információt láthatunk ezen az oldalon, mint például: a mérőegység helye, utolsó frissítés ideje, az eszköz neve.



13. ábra A mérési adatok megjelenítő felülete

Az eszközök menüben bármelyik eszközre kattintva tekinthetjük meg az adott eszközhöz tartozó szenzorok mérési adatait. A weboldal minden szenzort külön grafikonban jelenít meg. Az oldalon található megjelenítők külső gyártó által készített jQuery pluginek. A grafikonok szabadon testre szabhatók, például módosítható a megjelenített időtartam továbbá lehetőséget adnak nyomtatásra vagy fájlba mentésre.

5 MÉRÉS MEGTERVEZÉSE ÉS MENETE

A méréseink megtervezéséhez figyelembe kellett vennünk a lehetőségeinket. Jelenleg irodai környezetben gondolkodunk, a Főiskolán telepített egységekkel.

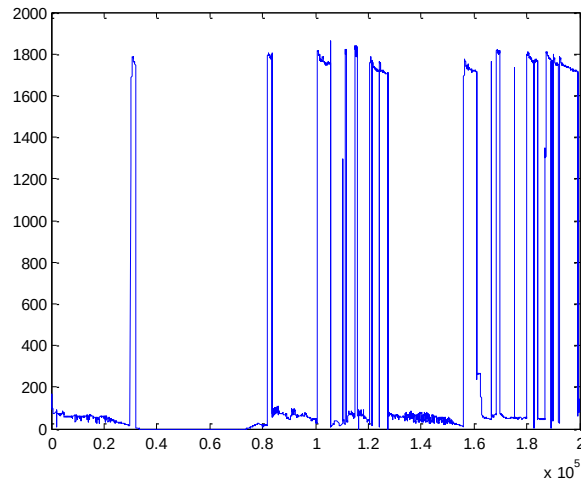
5.1 Mérések helyszíne

A környezeti paraméterek méréséhez a megtervezett modulból többet terveztünk elhelyezni a Kecskeméti Főiskola GAMF Karán, az Informatika Tanszék épületében. Több iroda/labor felszerelése a mérőegységekkel, kiegészítve fogyasztásmérő egységekkel, lehetőséget ad számunkra több helyszín adatainak naplózását és későbbi feldolgozását. Elsődleges célunk az volt, hogy a mozgásérzékelőt kiegészítsük egy kapcsolóval, ami a jelenlétet jelzi, ezzel annotálva az idősorunkat. Ezeken túl a belső fényerősség, páratartalom és kétféle hőmérséklet adatot rögzítünk folyamatosan egy irodában. A külső fizikai paraméterek mérésére kültéren elhelyezhető modult tervezünk telepíteni (fényerősség, hőmérséklet, páratartalom, légköri nyomás). A külső modulnak természetesen vízállónak kell lennie, de egyúttal lehetővé kell tennie a valós hőmérséklet (árnyékban) és páratartalom folyamatos mérését. Ez elsősorban modul dobozolásában jelent kihívást.

5.2 Gyűjtött adatok

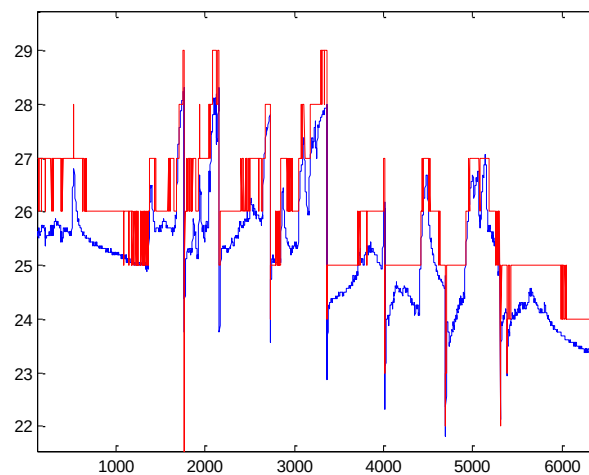
A mérési adatok a MySQL adatbázisból kiexportálhatóak pl. csv állományba, lehetővé téve a további feldolgozásukat MATLAB környezetben.

A fényérzékelő mérési adatai a 32. ábrán láthatóak (részlet a gyűjtött mérésből). A közvetlenül a lámpatest alatt elhelyezett szenzor felhasználható a lámpa be- és kikapcsoltságának érzékelésére, ami egyértelműen megadja a lámpa felvett teljesítményét is egyben. Tehát nincs szükség az általában csak nehezen hozzáférhető lámpacsatlakozásokra telepített fogyasztásmérőre.



32. ábra Fényérzékelő mérési adatok

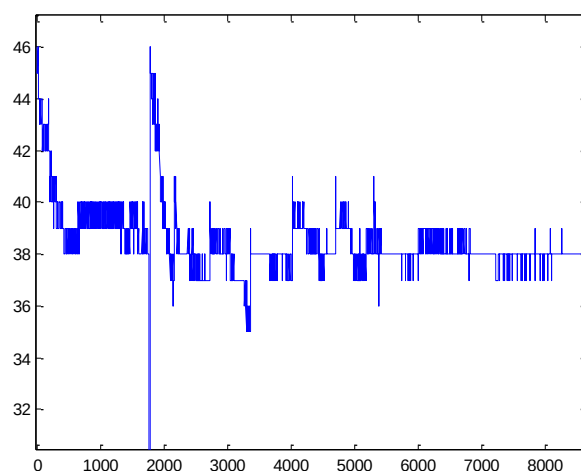
A belső hőmérséklet gyűjtésére kétféle hőmérőt telepítettünk. A 33. ábrán jól látható, hogy a két szenzor különböző felbontással tudja a hőmérsékletet visszaadni. A szenzorok ára között gyakran igen nagy különbség lehet, a pontosabbak drágábbak, de egyes feladatokhoz az olcsóbb, kevésbé pontos eszközök is elegendőek. Annak kiderítése, hogy milyen pontos mérésekre, és egybe milyen drága eszközökre van szükség például a hőmérséklet érzékelésére, egy újabb kutatási téma lehet.



33. ábra Kétféle hőmérsékletérzékelő adatai (piros: DHT-11, kék: DS18B20)

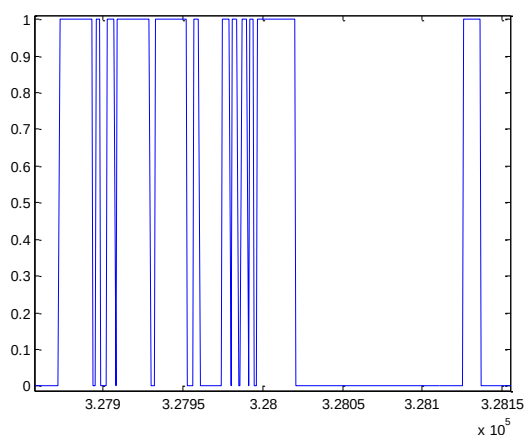
Jelenleg a belső páratartalom mérése is megtörténik, bár a fogyasztási adatok közvetlen összefüggésére még nem látunk lehetőséget. Más alkalmazásokban, például a szellőztetési igény megállapítására természetesen közvetlenül felhasználhatóak a mérési adatok, és

méréseinknek ez is egy célja, hogy másoknak lehetőséget adjunk egyrészt az adatok felhasználására, másrészt új alkalmazási területek feltárására.

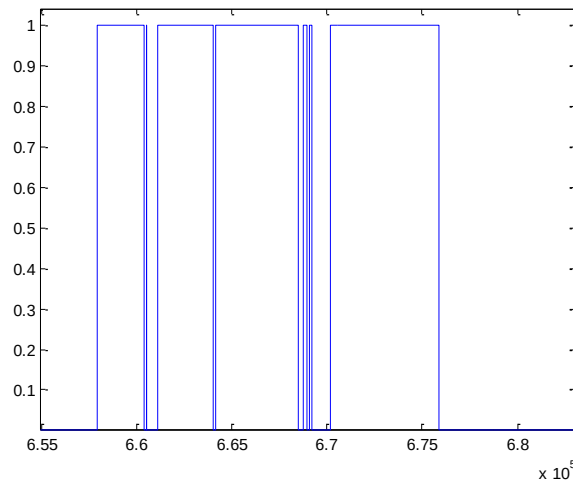


34. ábra Páratartalom érzékelő adatai

A következő két érzékelő jele összefügg egymással (35. és 36. ábra), hiszen a mozgásérzékelő adataiból szeretnénk majd megbecsülni (ld. következő fejezet) a jelenlétet. Mindkét adatsorról elmondható, hogy bináris (0,1 érték) jellegű. A jelenlét kapcsoló az irodába belépve, jobb oldalt található feltűnő helyen. A mérések során az irodában előforduló személyekkel megbeszéltük, hogy belépéskor kapcsolják be (ha még nem lenne), utolsónak elhagyva az irodát pedig kapcsolják ki azt. A mozgásérzékelő esetében annak érzékenysége és telepítési helye is befolyásolja a kapott adatsorokat, de ezeknek a befolyásoló tényezőknek a pontos vizsgálatával jelenleg nem foglalkozunk.

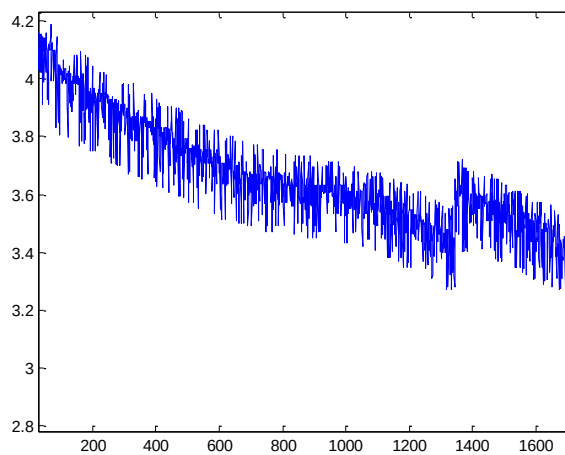


35. ábra Mozgásérzékelő mérési adatai



36. Jelenlét kapcsoló mérési adatai

Végül bemutatunk a tápfeszültség mérési adataira egy példát (37. ábra). Erre nyilvánvalóan azért van szükség, mert egyes esetekben (pl. kültéri egység) a mérő modulnak akkumulátorról kell működnie, és a lemerülést (feszültség drasztikus csökkenése) érzékelni kell.



37. ábra Akkumulátor feszültség mérési adatai

6 JELENLÉT BECSLÉSE A MOZGÁSÉRZÉKELŐ ADATAIBÓL

Eddigi mérési adatainkat egyik fontos részfeladatunk megoldására kíséreltük meg felhasználni. Ez a jelenlét megbecslése a mozgásérzékelő adataiból, ugyanis a kettő közötti összefüggés nem egyértelmű. A PIR szenzor nagyobb mértékű mozgásra jelez egy rövid időtartamra, így annak megállapítása, hogy éppen valaki (vagy valakik) jelen vannak adott esetben nem könnyű feladat. Gondoljunk csak ülőmunkára, alvásra vagy egyéb olyan tevékenységre, amik nem járnak nagymértékű mozgással.

6.1 Megoldási javaslatunk, becslés jóságának mérése

A feladat megoldására egy egyszerű megoldást javasolunk: a mozgásérzékelő jelzését egy meghatározott ideig tartva becsüljük a jelenlétet. Természetesen a tartási idő befolyásolja a becslés jóságát, amire egy mérési elvet dolgoztunk ki a saját mozgás és jelenlét (az annotációhoz használt kapcsoló jele) mérési adatainkra alapozva. A becslés jóságát a következő formulával határoztuk meg:

$$L = \frac{TP + TN}{TP + TN + FP + FN}$$

, ahol a becsült és a valós (annotált) jelenlét összevetéséből

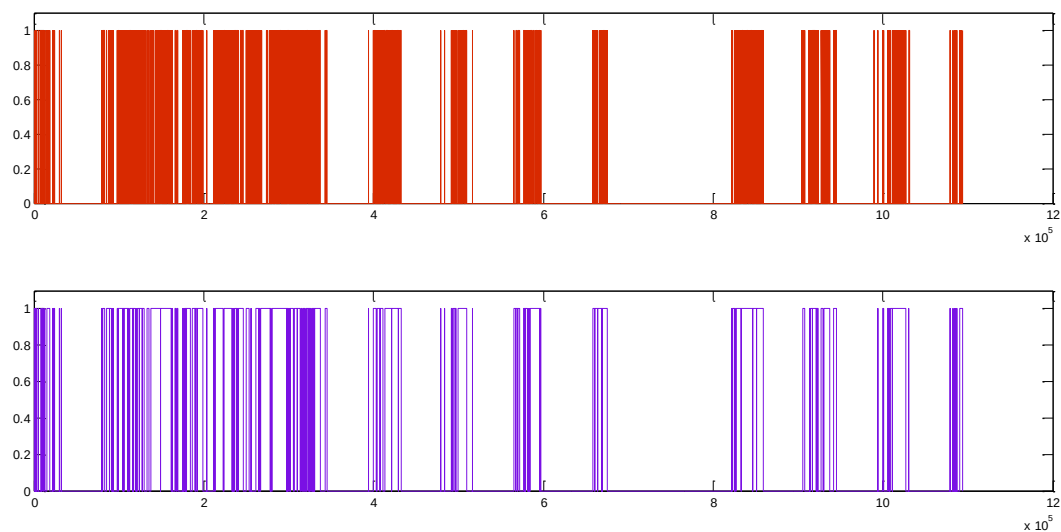
TP – True Positive: jelenlét igaz, a becslés is igazat ad,

TN – True negative: a jelenlét hamis, a becslés is hamisat ad,

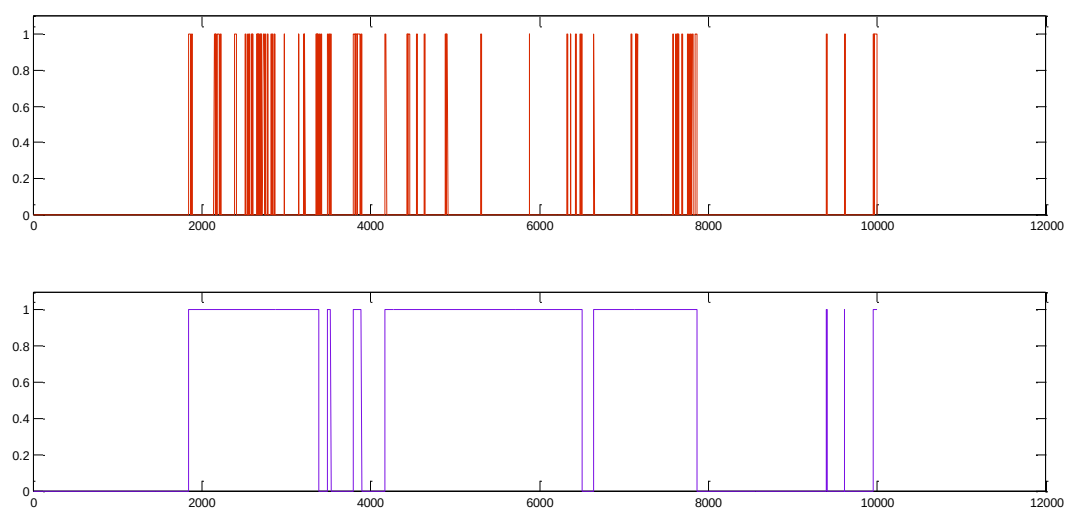
FP – False positive: a jelenlét hamis, de a becslés igazat ad,

FN – False negative: a jelenlét igaz, de a becslés hamisat ad.

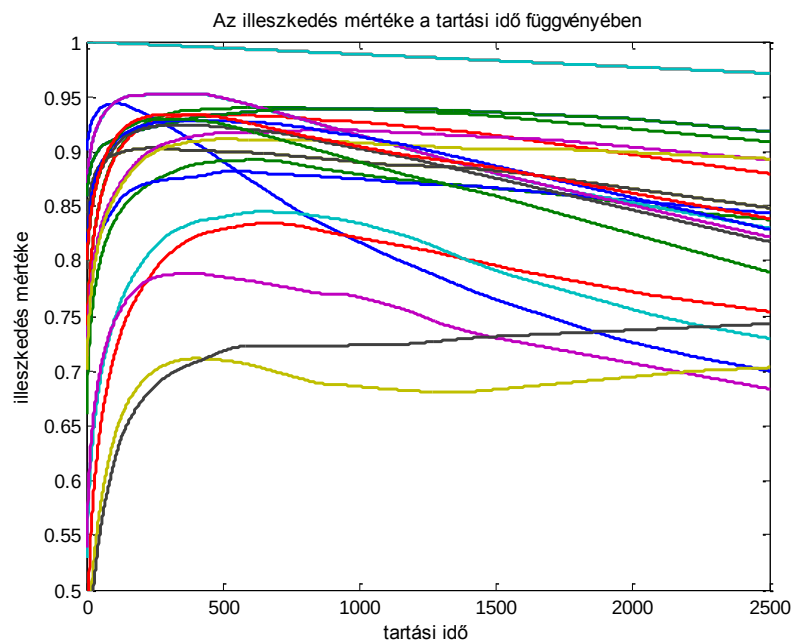
A 15 napos mérési sorozat (38,39. ábrák) alapján először egynapos (86400 másodperc) időablakokat képeztünk, fél napos átfedésekkel. Az időablakok alapján a tartási időt 0-2500 másodperces tartományban beállítva kiszámítottuk az összes esetben a becslés jóságát (40. ábra). A maximális illeszkedés (függvény maximuma) jelenti a legjobb becslést.



38. ábra 15 napos mérési sorozat; felső ábrán a mozgásérzékelő jele, alsón a jelenlét jelzésére használt kapcsoló jele

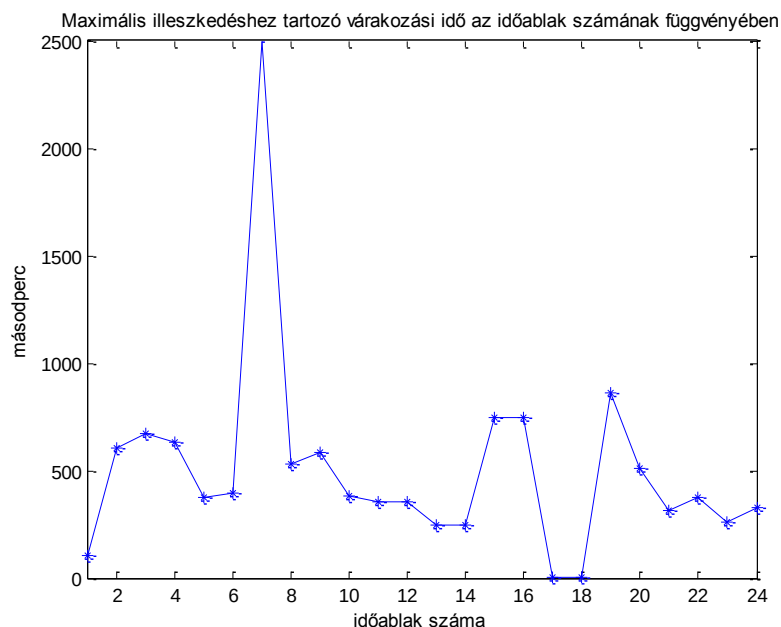


39. ábra Mozgásérzékelő és jelenlét jelei rövidebb időszakaszra

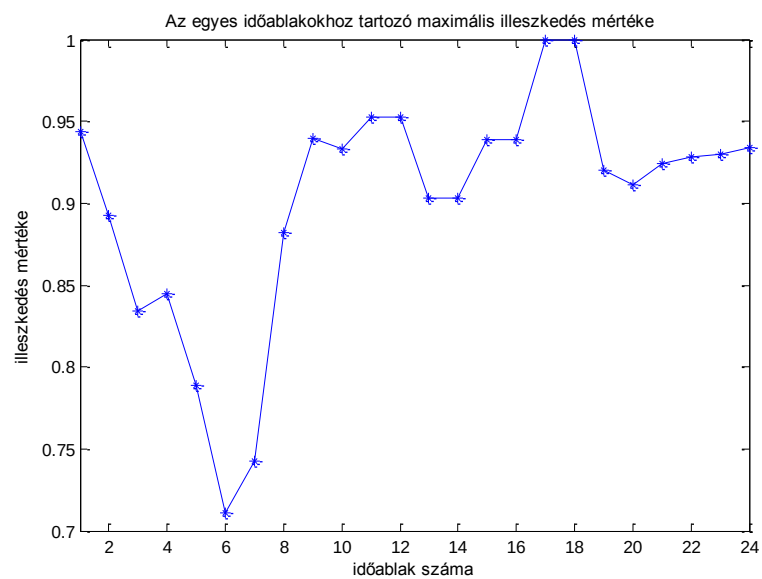


40. ábra Az illeszkedés mértéke a tartási idő függvényében

Az összes időablakhoz külön meghatározva a maximális illeszkedést megkapjuk a maximális illeszkedéshez tartozó várakozási időket (41. ábra). Ennek átlagát választva a becslésünk optimálisnak tekinthető az összes időablakra véve. Az egyes időablakokhoz tartozó maximális illeszkedés mértéke a 42. ábrán látható.



41. ábra Maximális illeszkedéshez tartozó várakozási idők (átlag: 506,25 másodperc)



42. ábra Az egyes időablakokhoz tartozó maximális illeszkedés mértéke

7 SMART* ADATBÁZIS FELDOLGOZÁSA, IDŐSOROK KÖZÖTTI ÖSSZEFÜGGÉSEK FELTÁRÁSA (JELENLÉTBŐL FOGYASZTÁS BECSLÉSE)

A saját tervezésű és építésű mérőrendszerünk telepítése jelenleg is folyamatban van. Mivel saját adatok még nem állnak rendelkezésre a jelenlét és fogyasztás viszonylatában, egy szabadon hozzáférhető adatbázisból indultunk ki, hogy feltárjuk az egyes idősorok közötti összefüggéseket.

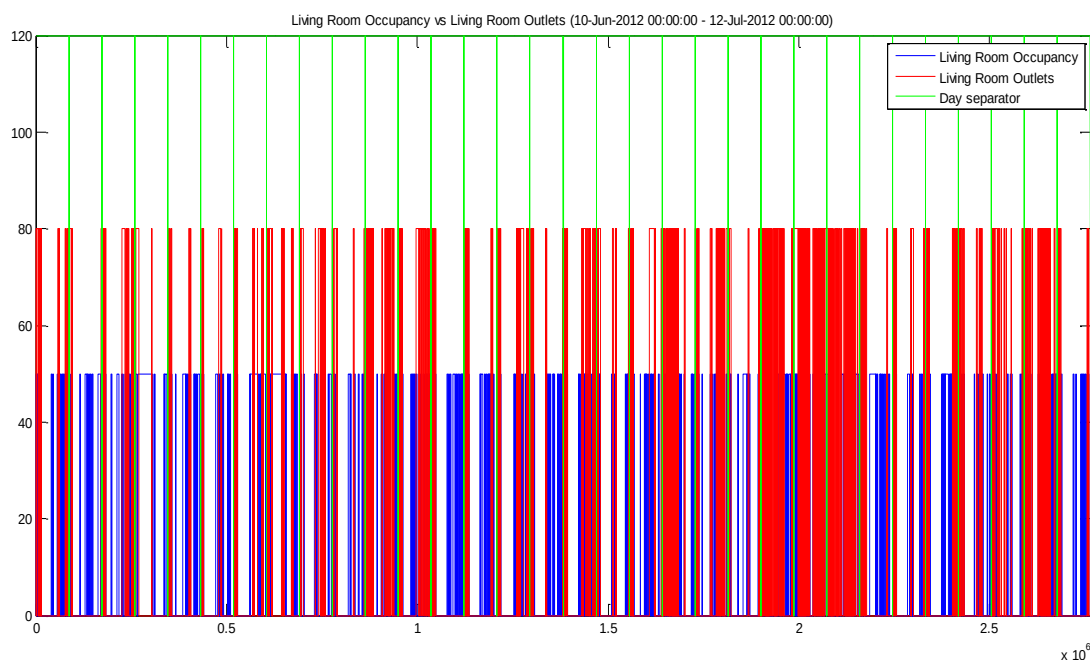
7.1 Smart* adatbázis feldolgozása

A Smart* adatbázist [4] használtuk fel, bár nem teljesen felel meg céljainknak. Ugyan részletes, készülékszintű mérési adatokat tartalmaz, sajnos jelenlét helyett csupán mozgásérzékelők adataira támaszkodhatunk, illetve az adatbázis elég sok hibával rendelkezik. A feldolgozás során derült ki például, hogy az összesen három hónapos idősorokban, több mint egy hónapra nincs mozgásadat gyűjtve, továbbá a konyhai mozgásdetektor adatai használhatatlanok (0-24 órában folyamatos mozgást jelez). A Smart* csv állományok formájában tartalmazza napi bontásban a fogyasztási, mozgás és környezeti paraméterek adatsorait. A feldolgozáshoz MATLAB-ot használtunk és a következő adat előkészítő és adattisztító lépéseket végeztük el:

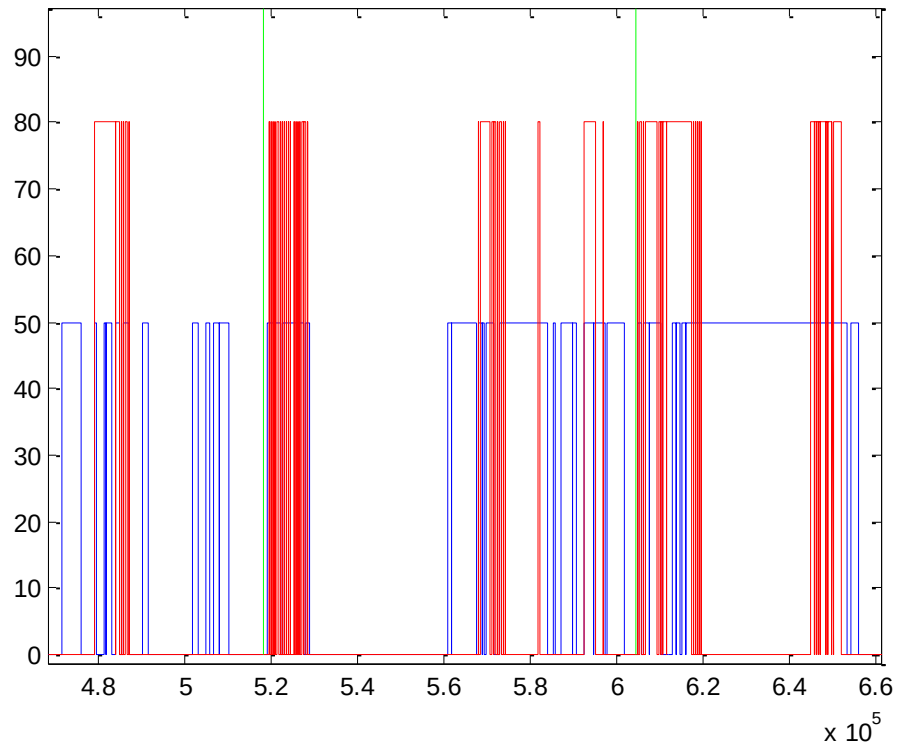
- A csv állományok beolvasása, összefűzése,
- Készülék- és helyiségszintű leválogatás,
- Időbélyegek folytonosságának, sorrendjének ellenőrzése,
- Időbélyeg szerinti sorba rendezés,
- Hiányzó adatok pótlása (fogyasztási idősorok esetén rövid kihagyásokra lineáris interpoláció, hosszabbakra nulla beszúrása; mozgásadatoknál előző érték ismétlése).

Az elő feldolgozás után következhetett az idősorok párba állítása függés szerint, amiknek a kiválasztása egyrészt az alapján történt, hogy milyen adatok állnak rendelkezésre, másrészt logikus módon egyáltalán mely párok esetén lehetséges függőség. A kiválasztott idősorok esetében ezután meg kellett állapítani a bekapcsolási valószínűségüket egy-egy időszakban. Az időszak ebben az esetben napon belül egy-egy időszakot jelent, pl. fél vagy egész órát. A bekapcsoltság valószínűségét az összes elérhető napra véve átlagoltuk, kiszámítottuk a szórását és legnagyobb, ill. legkisebb értékeit.

Egy példa látható a következő két ábrán, amiknek előállítása MATLAB-ban történt és átskálázást hajtottunk rajta végre (a mozgásérzékelő értéke 0-1, míg a fogyasztás jóval nagyobb lehet). Kékkel a nappaliban a jelenlétet, pirossal eszközök együttes fogyasztását mutatja, a zöld elválasztó függőleges vonalak pedig a naphatárokat mutatják. (További párok tekinthetők meg a mellékletben.)

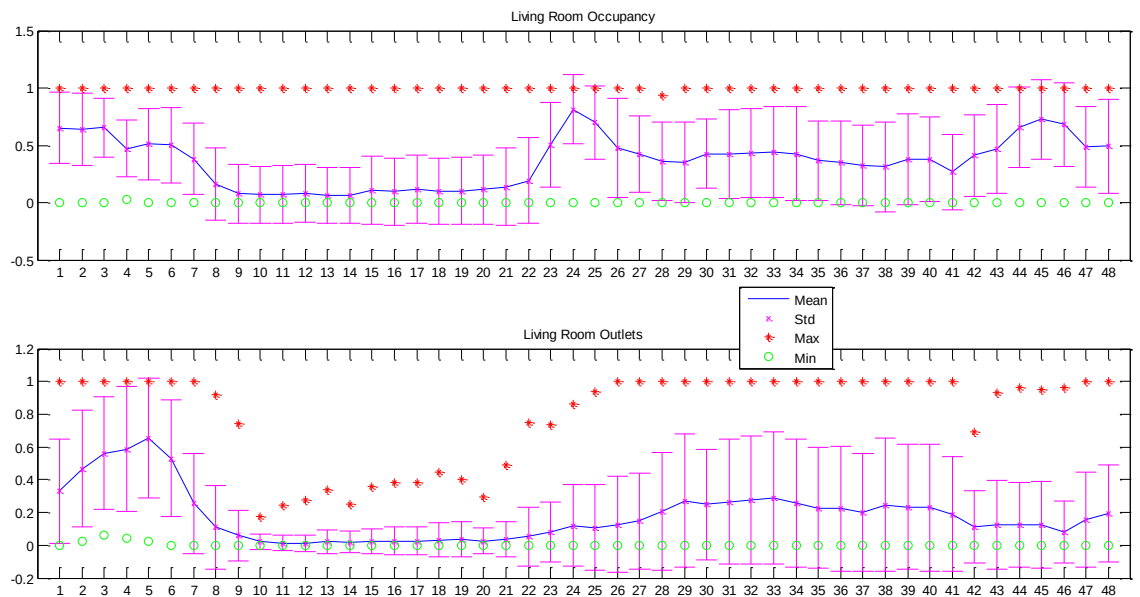


43. ábra Nappaliban jelenlét és fogyasztás



44. ábra Nappaliban jelenlét és fogyasztás (rövidebb időtartam)

A második ábrarészletre tekintve (44. ábra), a feltételezésünk szerinti összefüggés (ti. a jelenlét és a fogyasztás közötti összefüggés) létezik. Az időfüggő statisztikai paramétereket a következő ábra mutatja.



45. ábra Időfüggő statisztikai paraméterek

8 EREDMÉNYEK ÉRTÉKELÉSE, TERVEK

A TDK dolgozatban egy környezeti paraméterek mérésére alkalmas eszközről, azzal történt mérések leírásáról, valamint az adatok felhasználásáról olvashattunk, de konkrétan megfogalmazva a következő eredmények születtek:

- Elkészült egy hardver és szoftver együttes környezeti paraméterek mérésére; a rendszer fogyasztásmérők adatait is képes fogadja,
- A mérések megkezdődtek, egy irodában hőmérséklet, belső fényviszony, páratartalom és annotált mozgás adatok gyűltek több héten keresztül,
- A mért adatokból ki lett dolgozva egy eljárás a mozgásból a jelenlét becslésére,
- A jelenlét és az eszközök közötti fogyasztás közötti összefüggés felderítése elkezdődött, de jelenleg még más adatbázissal (Smart* data set) történik a vizsgálódás.

A kutatási témát szándékunkban áll folytatni, jelenleg is a mérőrendszer további részeinek telepítésével, az adatok pontosabb feldolgozásával foglalkozunk. Az összefüggések feltárása részben megtörtént, terveink szerint a modellt neurális hálózattal fogjuk meghatározni. A neurális hálózat bemenetei a jelenlét időablaka és az időablakra meghatározott bekapcsolási valószínűség, míg a kimenete a fogyasztás valószínűsége az adott időablakban. Az összefüggések nem lineárisak, de a neurális hálózat éppen abban erős, hogy nemlineáris kapcsolatokat is modellezni tud megfelelő tanítási eljárást és tanítóhalmazt alkalmazva.

9 IRODALOMJEGYZÉK

[1] Rajmund Drenyovszki, Lorant Kovacs, Andras Olah, David Tisza, Kalman Tornai, Zeno Domotor: “A Survey on Demand Side Management Techniques in Smart Grids“, Proceedings of Factory Automation, pp. 13-18, May 21-22, Veszprém, Hungary, 2013., ISBN: 978-615-5044-80-9

[2] Lorant Kovacs, Janos Levendovszky, Andras Olah, **Rajmund Drenyovszki**, David Tisza, Kalman Tornai and Istvan Pinter: “A probabilistic approach for admission control of smart appliances in Smart Grids“, Proceedings of 5th International Scientific and Expert Conference TEAM 2013, pp. 18-21, November 4-6, Presov, Slovakia, 2013.

[3] J. Zico Kolter and Matthew J. Johnson. REDD: A public data set for energy disaggregation research. In proceedings of the SustKDD workshop on Data Mining Applications in Sustainability, 2011.

[4] Sean Barker, Aditya Mishra, David Irwin, Emmanuel Cecchet, Prashant Shenoy, and Jeannie Albrecht: ”Smart*: An Open Data Set and Tools for Enabling Research in Sustainable Homes”, Proceedings of the 2012 Workshop on Data Mining Applications in Sustainability (SustKDD 2012), Beijing, China, August 2012.

[5] Christian Beckel, Wilhelm Kleiminger, Romano Cicchetti, Thorsten Staake, and Silvia Santini: ”The ECO Data Set and the Performance of Non-Intrusive Load Monitoring Algorithms”, Proceedings of the 1st ACM International Conference on Embedded Systems for Energy-Efficient Buildings (BuildSys 2014). Memphis, TN, USA. ACM, November 2014.

[6] M. Integrated, „Maxim Integrated,” Maxim Integrated, 22 április 2008. [Online]. Available: <http://datasheets.maximintegrated.com/en/ds/DS18B20.pdf>. [Hozzáférés dátuma: 22 szeptember 2014].

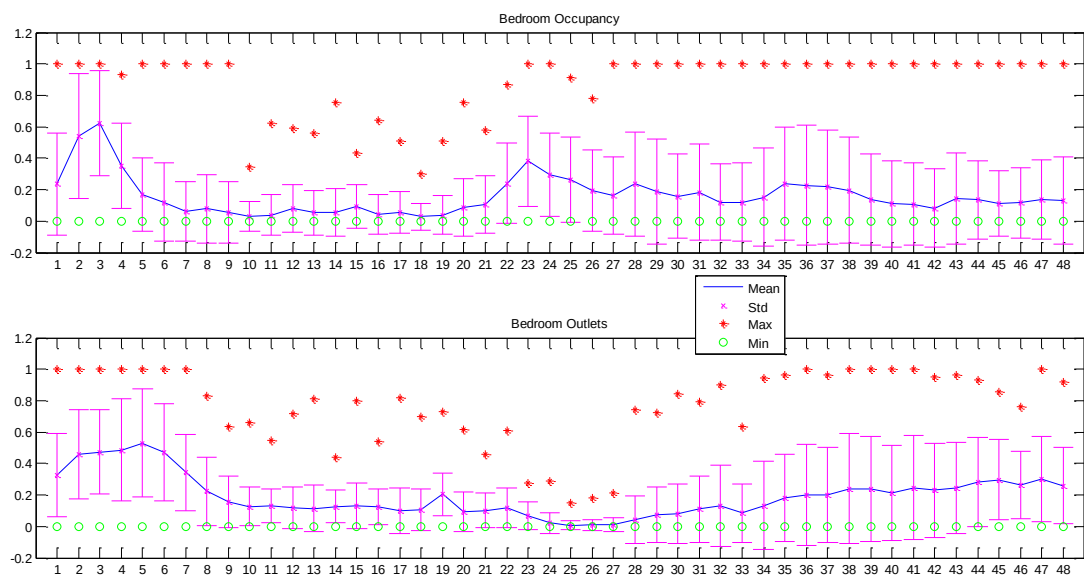
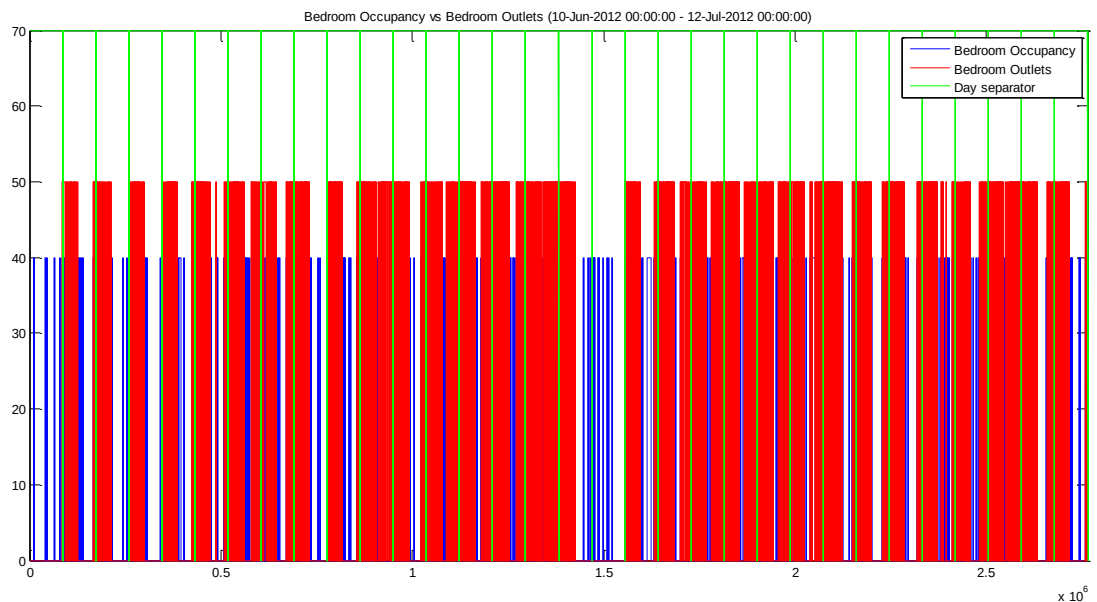
[7] R. Semiconductor, „ROHM Semiconductor,” ROHM Semiconductor, november 2011. [Online]. Available: <http://rohms.rohm.com/en/products/databook/datasheet/ic/sensor/light/bh1750fvi-e.pdf>. [Hozzáférés dátuma: 22 szeptember 2014].

[8] „Nordic Semiconductor,” Nordic Semiconductor, Július 2007. [Online]. Available: http://www.nordicsemi.com/eng/content/download/2730/34105/file/nRF24L01_Product_Specification_v2_0.pdf. [Hozzáférés dátuma: 13 Október 2014].

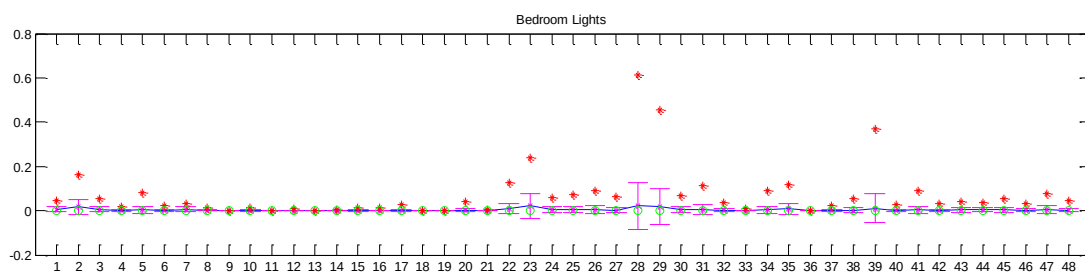
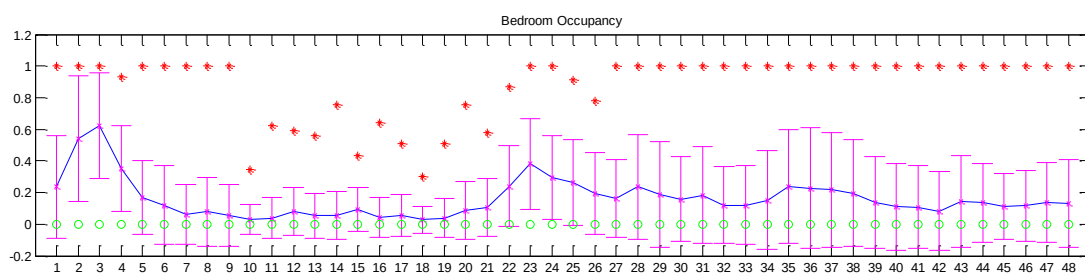
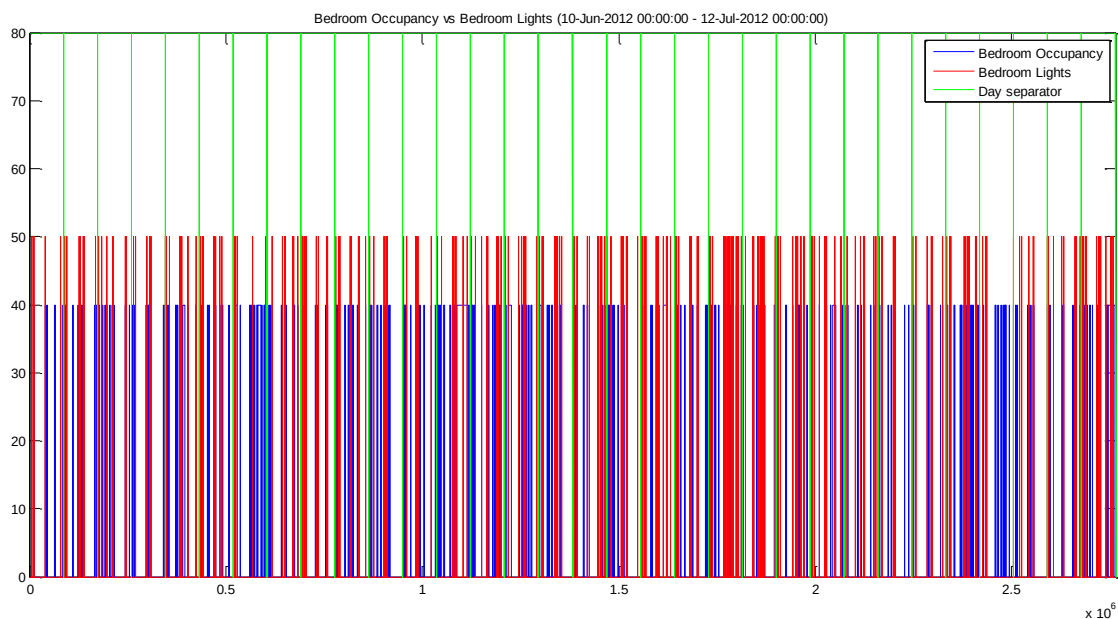
[9] HK-Link, „HK-Link,” HK-Link, 20 November 2012. [Online]. Available: <http://www.hlktech.net/inc/lib/download/download.php?DIId=20>. [Hozzáférés dátuma: 13 Október 2014].

10 MELLÉKLET

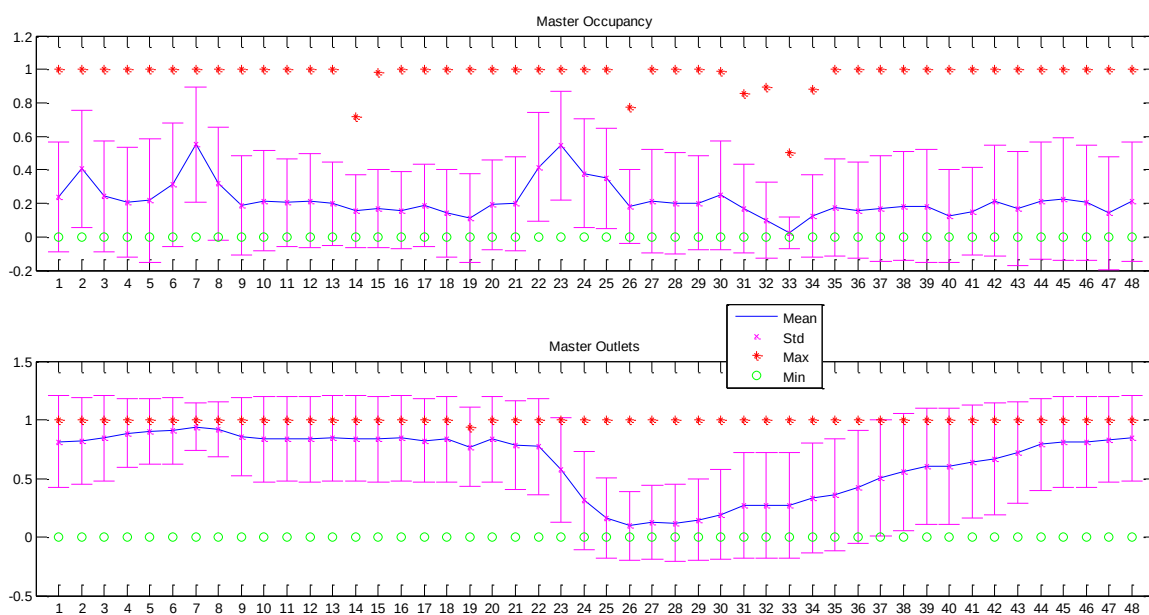
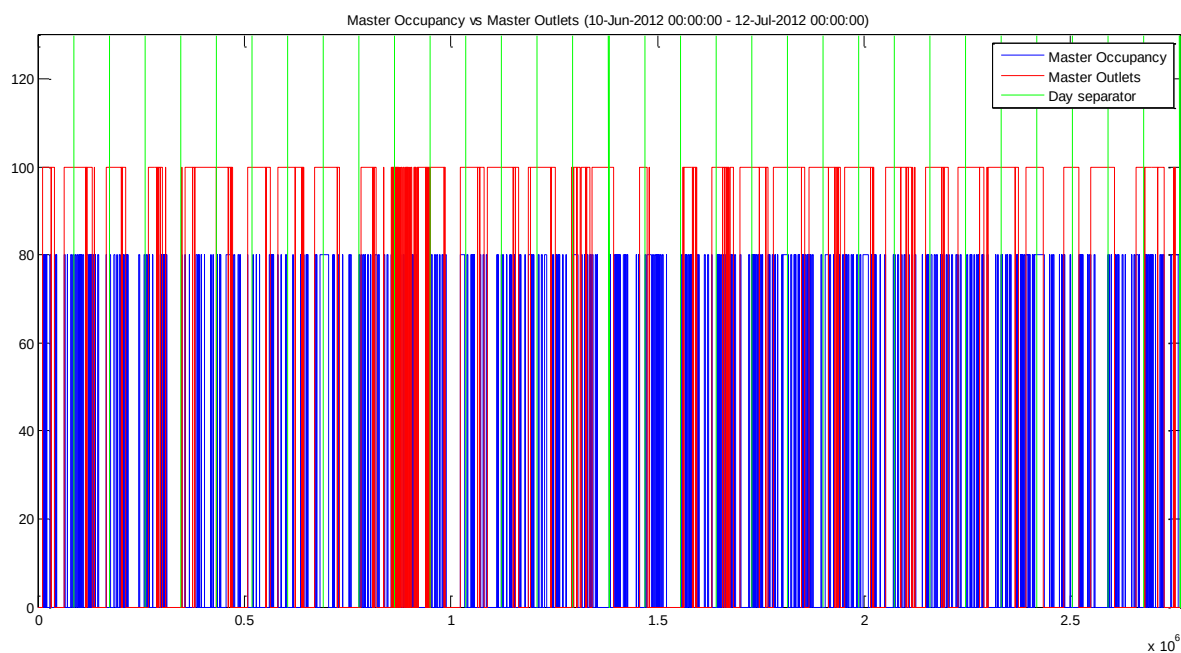
Hálószoja jelenlét, hálósobai fogyasztók



Hálószoza jelenlét, hálósobai lámpák



Nagyszoba jelenlét, nagyszobai lámpák



Mikrohullámú sütő, konyhai lámpák

