



ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

KANDÓ KÁLMÁN VILLAMOSMÉRNÖKI KAR
MŰSZERTÉCHNIKAI ÉS AUTOMATIZÁLÁSI INTÉZET



SZAKDOLGOZAT



OE-KVK
2021.

Majoros Márk
T008263/F112904/K



ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

Óbudai Egyetem
Kandó Kálmán Villamosmérnöki Kar
Műszertechnikai és Automatizálási Intézet

SZAKDOLGOZAT FELADATLAP

Hallgató neve: Majoros Márk

Szaktervezési száma: SZD21060114202796

Törzskönyvi száma: T008263/F112904/K

Neptun kódja: VS5UMB

Szak: villamosmérnöki (magyar nyelven)

Specializáció: Műszer-automatika specializáció

A dolgozat címe: Smart City szenzor adatgyűjtő és adatelemző szoftverének fejlesztése

A dolgozat címe angolul: Data acquisition and data processing software development for Smart City sensors

A feladat részletezése: Feladat:

1. Mikrokontroller programozása. Analóg és digitális szenzorok jelének kiolvasása, feldolgozása, adatcsomagok előállítása. Az adatcsomagok továbbítása NB IoT adatkommunikációs csatornán egy adatbázisba.

2. Az adatbázisban tárolt információ feldolgozása, megjelenítése, elemzése.

Intézményi konzulens neve: Dr. Balázs László

A kiadott téma elévülési határideje: 2024. június 30.

Beadási határidő: 2021. 12. 15.

A szaktervezési: Nem titkos.

Kiadva: Budapest, 2021. 11. 09.



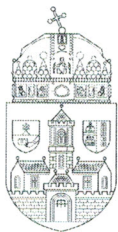
Intézetigazgató

A dolgozatot beadásra alkalmasnak találom:

Balázs László

belső konzulens

külső konzulens



ÓBUDAI EGYETEM
ÓBUDA UNIVERSITY

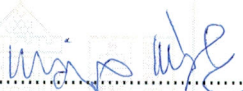
KANDÓ KÁLMÁN VILLAMOSMÉRNÖKI KAR
MŰSZERTÉCHNIKAI ÉS AUTOMATIZÁLÁSI INTÉZET



HALLGATÓI NYILATKOZAT

Alulírott hallgató kijelentem, hogy a szakdolgozat saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült szakdolgozatban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja, a titkosításra vonatkozó esetleges megkötések mellett.

Budapest, 2021.november 30.



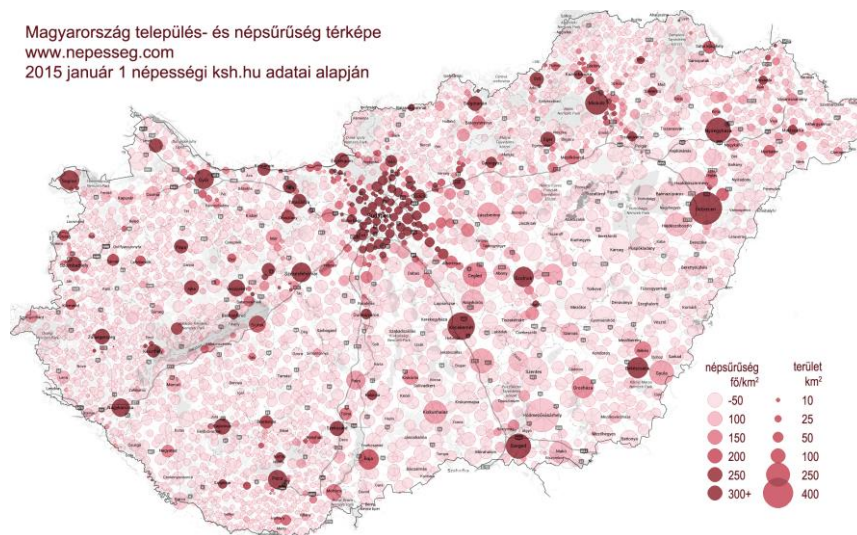
.....
hallgató aláírása

TARTALOM

1. BEVEZETÉS.....	1
2. AZ OKOS VÁROSOK ÉS AZ IOT KAPCSOLATA	3
2.1. Mi tesz okossá egy várost?.....	3
2.2. Az IoT definíciója	3
2.3. Az IoT rendszerek felépítése, működése	4
3. A SZENZOREGYSÉG FIZIKAI FELÉPÍTÉSE	6
3.1. Az eszközzel szemben támasztott követelmények.....	6
3.2. Rendszerterv.....	6
3.3. Az eszköz helye egy Smart Cityben.....	8
3.4. A megvalósított eszköz	9
4. AZ ADAT ÚTJA A MIKROKONTROLLERTŐL A DÖNTÉSIG	11
4.1. A mikrokontrolleren futó szoftver	11
4.2. Adattovábbítási lehetőségek.....	13
4.3. Üzenetkezelés MQTT-vel	13
4.4. A Node-Red mint adatfeldolgozó szoftver	15
5. AZ ADATFELDOLGOZÁS	17
5.1. Az adatfeldolgozás jelentősége	18
5.2. Adatfeldolgozás módszerek.....	19
5.3. A begyűjtött adatok elemzése.....	22
5.3.1. Hőmérséklet	24
5.3.2. Páratartalom	27
5.3.3. Légnyomás.....	30
5.4. Az adatfeldolgozás előkészítése.....	33
5.5. Hőmérséklet-kontroll beltérben	33
5.6. Döntési alternatívák a páratartalom ismeretében	36
5.7. Időjárás-előrejelzés légnyomás adatok felhasználásával	38
5.8. Konklúzió	39
6. ÖSSZEGZÉS.....	40
7. SUMMARY	41
Köszönetnyilvánítás.....	42
Irodalomjegyzék	43
Ábrajegyzék	45

1. BEVEZETÉS

A közeljövő egyik, ha nem a legizgalmasabb demográfiai kihívása az urbanizáció. Napjainkban Magyarországon a városi lakosság aránya 69,5%, ami azt jelenti, hogy hazánkban közel 7 millió ember él városi környezetben, és ez a szám folyamatosan növekszik. [1] A városok lakosságszámának növekedése komoly kihívások elé állítja a városüzemeltetésért és fejlesztésért felelős önkormányzatokat, cégeket, vállalatokat. A nagyvárosokban komoly gondot okoz például az infrastruktúra üzemeltetése, a megfelelő működés monitorozása, fejlesztésének tervezése, illetve annak ütemezése. Ha ezek a folyamatok nem megfelelően működnek, a város kritikus infrastruktúrái túlterhelődnek vagy megrongálódnak, ami nagy mértékben kihat a városlakók életére, mindennapjaira.



1. ábra
Magyarország népességének eloszlása [1]

De a problémára van megoldás. Napjainkban éljük a negyedik ipari forradalom korát, amely elsősorban a modern digitális eszközök robbanásszerű fejlődését hozta. Ezen eszközök segítségével képesek vagyunk a városüzemeltetést érintő problémák kezelésére is. Hiszen a rohamosan fejlődő infokommunikációs technológiáknak köszönhetően a város minden „rezdülését” megfigyelhetjük és monitorozhatjuk, legyen szó a kritikus infrastruktúra működéséről, a levegőtisztaságról vagy akár egy kereszteződés forgalmáról. A megfelelő adatok begyűjtésével, tárolásával, feldolgozásával még időben

meghozhatunk olyan döntéseket, amelyek támogatják a városüzemeltető szakemberek munkáját, elősegítik a település konzisztens fejlődését és megfelelő üzemelését. Egy ilyen, modern mérőeszközökkel és döntéstámogató rendszerekkel felszerelt várost nevezünk okos városnak, vagyis Smart Citynek.

Projektem során betekintést szeretnék nyújtani ebbe az igen gyorsan fejlődő technológiába, illetve bemutatom egy kiskereskedelmi forgalomban beszerezhető eszközökből összeállított szenzoregység működését: a vezérléshez szükséges szoftveres hátteret, a mért adatok feldolgozását, a begyűjtött adatok segítségével történő döntéshozatal menetét. Munkám során igyekeztem olcsón beszerezhető hardverrel, illetve nyílt forráskódú ingyenes szoftverekkel dolgozni, hogy az így elkészített rendszer kis költségből üzemeltethető és könnyedén reprodukálható legyen, amely véleményem szerint elengedhetetlen egy későbbiekben megvalósítandó nagyobb szenzorhálózat létrehozásához.

2. AZ OKOS VÁROSOK ÉS AZ IOT KAPCSOLATA

2.1. Mi tesz okossá egy várost?

Egy várost akkor nevezünk „okosnak”, ha a tradicionális és a modern információs és kommunikációs infrastruktúrába történő együttes befektetés ösztönöz egy fenntartható gazdasági fejlődést, és ez növeli a lakosság életszínvonalát. [2] Az ilyen típusú fejlesztések megkerülhetetlenek, mivel egyre fontosabb a takarékosabb működés, a ráfordítások és költségek optimalizálása, és ezzel együtt a városok ökológiai lábnyomának csökkentése. Ezek a települések azonban külső megjelenésükben nem különböznek a hagyományos városoktól. Ami más: a bennük lévő rendszerek és szolgáltatások működése. Hiszen a Smart Cityben minden történéshez adatok tartoznak, amelyeket az itt elhelyezett berendezések gyűjtenek és továbbítanak helyben lévő, vagy nagyobb, országos, esetleg globális adatbázisokba. Az így kinyert információk, s az azokra alapozott döntések pedig egy költséghatékonyabb, észszerűbb, rugalmasabb működést biztosítanak. [3] Egy Smart City működéséhez tehát két nagyon fontos dologra van szükség: az adatok mérésére specializált hardverre, illetve a hardverek által begyűjtött adatok feldolgozását megvalósító szoftverre. A működés alapja, hogy napjainkban szinte már minden méréshez létezik egy speciális szenzor. Ahhoz viszont, hogy egy rendszerbe mindig a legjobbkor, a megfelelő ponton tudjunk beavatkozni, nem elegendő néhány mérés. Mivel ismernünk kell annak minden egyes tulajdonságát, ezért rengeteg mérésre van szükségünk. A sok mérésből származó sok adatra alapozva leszünk képesek valós idejű döntéseket hozni, és a szükséges pontokon a rendszerbe beavatkozni. Ezen elvárások megvalósításában nagy szerepet játszik a negyedik ipari forradalom vívmánya, a dolgok internete, azaz az IoT.

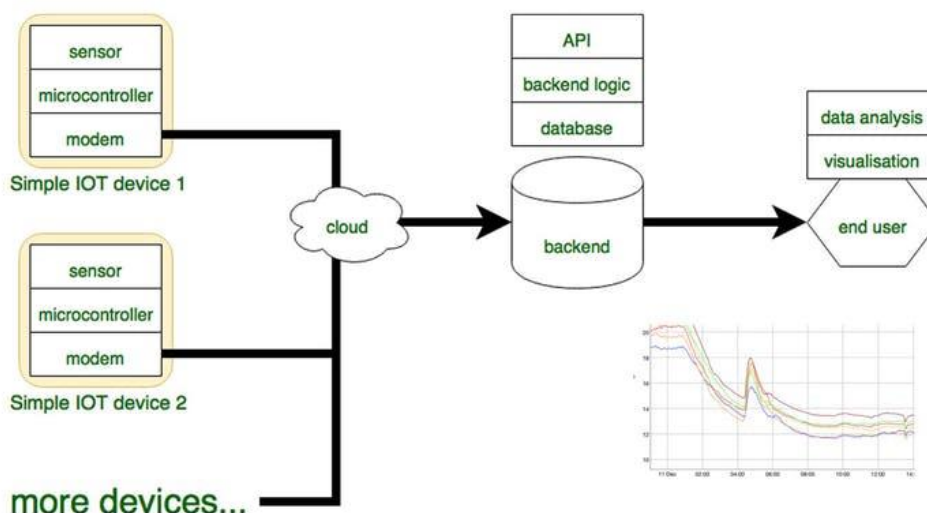
2.2. Az IoT definíciója

Napjainkban jelent meg új fogalomként az IoT, azaz a dolgok internete (Internet of Things). A fogalomra többféle meghatározást találhatunk, ezek azonban egy dologban megegyeznek: míg a „hagyományos” internet esetében az adatokat élő személyek generálják (pl.: honlapokon), addig a dolgok internetén az adatokat eszközök generálják. Vagyis ezidáig az internethez csatlakozó hagyományos eszközeink pl.: számítógépeink, okostelefonjaink mind valamilyen formában kapcsolatban álltak a felhasználóval. Ezzel

szemben az IoT eszközök legfontosabb tulajdonsága, hogy nem állnak folyamatos kapcsolatban semmilyen külső kezelő személlyel, hanem önállóan képesek üzemelni. A gépek már nemcsak az emberek közötti kommunikáció folyamatát valósítják meg, hanem maguk az eszközök kommunikálnak egymással. Lehetővé teszik, hogy az okos eszközök között áramló információkból, adatokból hatalmas, statisztikai célú adatbázisokat lehessen létrehozni. [4]

Az IoT tehát „egy okos eszközökből álló nyílt és átfogó hálózat, mely képes automatikusan rendszerezni és megosztani adatokat, illetve információkat.” [5]

2.3. Az IoT rendszerek felépítése, működése



2. ábra
Az IoT működése [18]

Ahogy a 2. ábra (Az IoT működése) is mutatja, egy ilyen rendszer első és legfontosabb elemei a bemenetet biztosító szenzorok. De egy szenzor önállóan működésképtelen, működtetéséhez valamilyen mikrokontrollerre, és egy azon futó (általában szenzorspecifikus) szoftverre is szükség van. A mérések vezérlése és megfelelő ütemezése mellett a mikrovezérlő felel az adattovábbításért is, ami valamilyen (jellemzően IoT specifikus) kapcsolat segítségével juttatja el a mért adatokat egy internetes felhőszolgáltatásba. A vezérlő és a szenzor által alkotott eszközt nevezzük szenzoregységnek. Ezen szenzoregységek működésében fontos szempont az alacsony

fogyasztás, mivel rendszerint olyan tényezők mérésére használják őket, ahol nem áll rendelkezésre helyben tápellátás. Egy IoT rendszerbe jellemzően több száz vagy több ezer ilyen szenzoregység is tartozhat.

Ezek a modulok gyűjtik az adatokat, melyeket azután tárolni és feldolgozni kell. A mért értékek tárolását általában internetes felhőszolgáltatásban valósítják meg addig, amíg azok feldolgozásra nem kerülnek. Ez a megoldás erőforráshatékony, mivel egy több száz vagy több ezer szenzorból álló szenzorhálózat annyi adatot generál, hogy azok tárolása hosszú távon lehetetlen, így csak az átmeneti tárolás nyújthat megoldást a problémára.

Az átmeneti adattárolást követő feldolgozásra többféle adatkezelő és -vizualizáló eszköz is létezik. Egy Smart City esetében egy ilyen eszközzel szemben a legfontosabb követelmény, hogy képes legyen a valós idejű adatfeldolgozásra és döntéshozatalra, valamint a vizuális megjelenítésre is.

3. A SZENZOREGYSÉG FIZIKAI FELÉPÍTÉSE

3.1. Az eszközzel szemben támasztott követelmények

Munkám során igyekeztem az egyszerűsége törekedni, könnyen beszerezhető eszközökkel dolgozni. Az eszközömet a próbaüzem ideje alatt csak beltérben kívántam üzemeltetni, ezért esetemben nem volt elvárás néhány, a való életben nagyon fontos követelmény, mint például a vízállóság, ám annál fontosabb volt a folyamatos és megbízható működőképesség, mivel egy Smart Cityben nagyon fontos a szenzorok hosszú távú és megbízhatóan működő adatszolgáltatása.

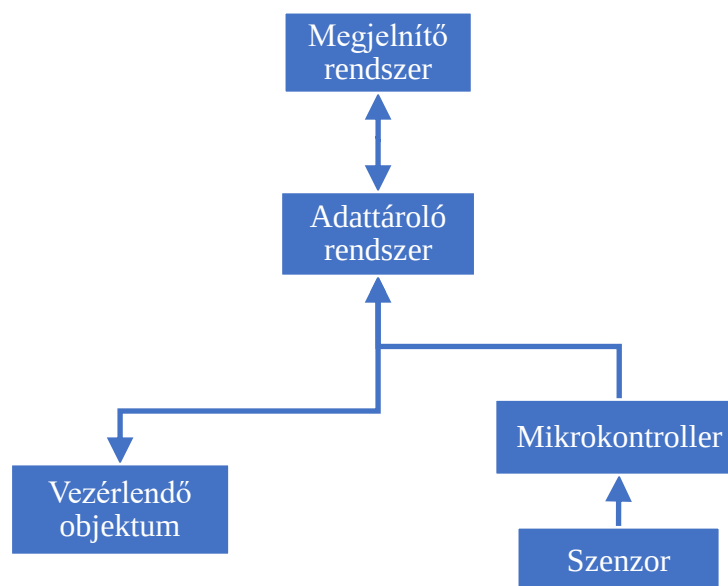
Fontos elvárásom volt a készülő prototípussal szemben, hogy mindenképpen rendelkezzen egy szenzoregységgel, amely képes a hőmérséklet, páratartalom és légnyomás mérésére, valamint a vezérléséhez elengedhetetlen mikrokontrollerrel. A megvalósítás során igyekeztem olyan mikrokontrollert választani, amelyben integráltan megtalálható valamilyen kommunikációs interfész, így nem kellett foglalkoznom egyéb külső kommunikációs modul illesztésével a haderes megvalósítás során.

Ezen felül, miután egy IoT szenzoregységet szerettem volna elkészíteni, nagy hangsúlyt kellett fektetnem az alacsony fogyasztásra is. A rendszer legnagyobb fogyasztója a megvalósítás egyszerűségéből fakadóan értelemszerűen a mikrokontroller, ezért a mikrokontroller kiválasztásánál figyelembe kellett vennem a gyártó által a specifikációban meghatározott fogyasztásértékeket is.

3.2. Rendszerterv

A fent említett követelmények megfogalmazása után nekiláthattam a rendszerterv elkészítésének, ami a 3. ábrán látható. A rendszerterv ábráját úgy állítottam össze, hogy azon nyilakkal jelöltem a mért adat, illetve a vezérlő egységtől a vezérlendő objektum felé irányuló adatcsomag útját, így könnyedén leolvasható, hogy mely modulok között milyen irányú kommunikációt kell megvalósítani.

A rendszer (a mért adat szempontjából vett) kiindulópontja a szenzor, amely a mikrokontrollerrel igen szoros kapcsolatban áll, mivel működtetéséhez elengedhetetlen valamilyen szoftveres vezérlés. A mikrokontroller amellet, hogy megfelelő vezérlőjellel látja el a szenzormodult, időzítési és előfeldolgozási funkciót is el fog látni.



3. ábra
Rendszerterv

Erre azért van szükség, mert az általam mérendő jellemzők lassú változásúak, így elegendő nagyobb időtávokban (pl. percenként) mérni azokat, és összegyűjtés után adatcsomagokban továbbítani, ami energiahatékonyság szempontjából is lényeges. A szenzorból a vezérlő által kiolvasott mérési eredmények tehát már előfeldolgozott formában egy kommunikációs interfészen keresztül jutnak át az adattároló rendszerbe, ahol átmenetileg tárolásra kerülnek. A későbbiekben az adatfeldolgozó egység innen olvassa ki az adatcsomagokat. Az adatfeldolgozó és -megjelenítő rendszer áll a rendszerterv „csúcán”, mivel ezen modul végzi el a beérkező adatok szétválogatását, illetve itt kerül megvalósításra az a döntéshozó algoritmus is, ami a megfelelő parancsok segítségével kommunikál a vezérlendő objektummal. Ezen funkciókon kívül tartalmaz egy megjelenítőt is, amelynek segítségével információkat tud átadni a felhasználónak az aktuális állapotról és trendekről, valamint szükség esetén a felhasználó külsőleg is beavatkozhat a működésbe.

3.3. Az eszköz helye egy Smart Cityben

Az általam megvalósított eszköz nemcsak önállóan működőképes, hanem egy okos város szenzorhálózatának részeként is megállja a helyét. Egy modern Smart Cityben rendszerint alkalmazzuk a „több szem többet lát” elvet, ezért városszerte több ponton mérünk egy adott jellemzőt. Egy ilyen módon kialakított rendszer gyorsan képes érzékelni a legapróbb változást is.

A páratartalom, a hőmérséklet és a légnyomás segítségével könnyedén megállapítható több igen fontos légköri jellemző is. Egyrészt a légnyomás és páratartalom hirtelen változásának segítségével előre jelezhető egy helyi csapadékháza érkezése, amely egy nyári zivataros napon, amikor a meteorológusok a gyorsan változó légköri viszonyok miatt nem tudják nagy biztonsággal és földrajzi pontossággal megjósolni ezt, nagyon hasznos lehet. Egy ilyen rendszer segítségével a városlakókat nem érheti meglepetésként egy nagyobb zápor vagy zivatar sem, így egyszerűen mérsékelhetőek például a gépjárművek viharkárai.

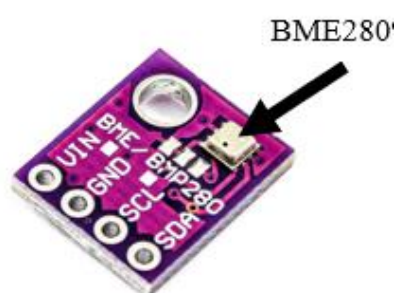
Másrészt a páratartalomból és a hőmérsékletből meghatározható a harmatpont, amely nagyon fontos jellemző az olyan ipari területeken, ahol sűrített levegővel dolgoznak. A harmatpont az a hőmérséklet, amely alatt a levegőben található nedvesség kondenzálódik, víz formájában kicsapódik, majd pedig harmat vagy (fagypont alatti harmatpont esetében) dér és zúzmara formájában kiválik. Ezen ismeret birtokában megelőzhető az ipari pneumatikus eszközök tönkremenetele.

S végül nagyon fontos a szenzormodul által márt hőmérséklet is, mivel amellet, hogy egy város több pontján mért hőmérsékletből úgynevezett hőmérsékleti térképet készíthetünk, több, további levegőminőség-szenzor számára is bemeneti paraméterként szolgál ez a jellemző. Például így működnek a szén-dioxid vagy szervesanyag tartalmat mérő légszennyezettség szenzorok.

Nyilvánvaló tehát, hogy egy Smart Cityben fontos szerepe van azoknak az eszközöknek, amelyek páratartalmat, légnyomást és hőmérsékleti adatokat mérnek több ponton és rendszerszerűen, hiszen ezek nemcsak önmagukban hasznos információk, hanem együttesen is értelmezhetőek, további szenzorok számára bemeneti paramétereket szolgáltatnak, s mindezzel döntéshozó algoritmusok alapelemeivé válnak.

3.4. A megvalósított eszköz

Munkám során a Bosch BME280 típusú egységével dolgoztam, mivel ezen egység képes az általam kiválasztott három jellemző mérésére. A szenzor amellet, hogy igen alacsony fogyasztású ($3,6 \mu\text{A}$ [6]), a környezeti viszonyoknak is széleskörűen ellenáll [6], illetve a mérete is igen kicsi. Kereskedelmi forgalomban legtöbbször (az egyszerű kezelhetőség miatt) egy kisméretű NYÁK lapra előre beültetve vásárolható meg, ahogyan a 4.ábrán is látható.



4. ábra
BME280 szenzor

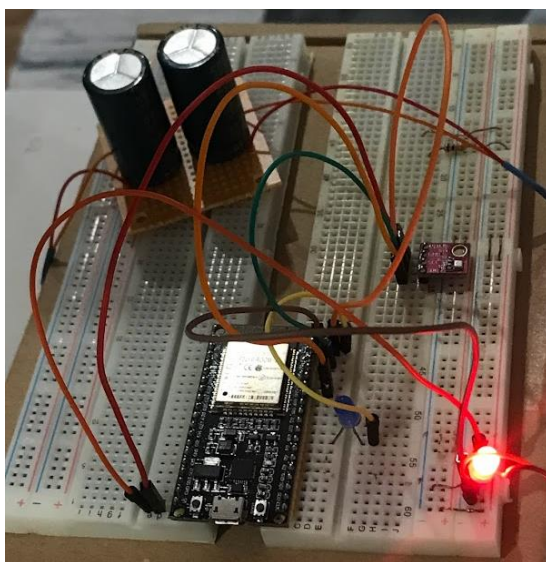
A szenzor vezérléséhez az Espressif által gyártott ESP32-WROOM típusú mikrokontrollert választottam (5. ábra), mivel rendelkezik előre beépített WiFi modullal, így a követelményekben is előírt kommunikációs interfészről nem kellett külön gondoskodnom. A mért adatokat a későbbiekben ennek segítségével juttatom el az átmeneti adattároló egységhez.



5. ábra
ESP32 mikrokontroller

Az eszköz emellett támogat számtalan kommunikációs protokollt is, amelyek közül esetemben a legfontosabb az I²C támogatás, mivel az általam használt szenzor ezen a kommunikációs csatornán kommunikál a külvilággal. Az ESP32 mikrokontroller egyébként kifejezetten IoT alkalmazásokhoz lett kifejlesztve, így az általam támasztott alacsony fogyasztási kritériumnak is képes megfelelni. Áramfelvétele az integrált WiFi modul működése közben sem haladja meg a 200 mA-t, ha pedig a kommunikáció inaktív, a névleges áramfelvétele sem lépi túl a 60 mA-t. [7]

A hardveres összeszerelést egy próbapanelen valósítottam meg (6. ábra), mivel ez a segédeszköz adja a legnagyobb rugalmasságot és szabadságot mind az alkatrészek elhelyezése, mind pedig az összeköttetések kipróbálása szempontjából. A panelre a mikrovezérlő és a szenzormodul mellett elhelyezésre került egy kék színű LED is, ami segít a hardver állapotának a külvilág felé történő visszajelzését megvalósítani. Az elkészült eszköz jelenleg külső tápellátásról üzemel, mert a beltéri tesztüzem során nem láttam szükségét akkumulátor használatának.



6. ábra
Az elkészült szenzormodul

4. AZ ADAT ÚTJA A MIKROKONTROLLERTŐL A DÖNTÉSIG

4.1. A mikrokontrolleren futó szoftver

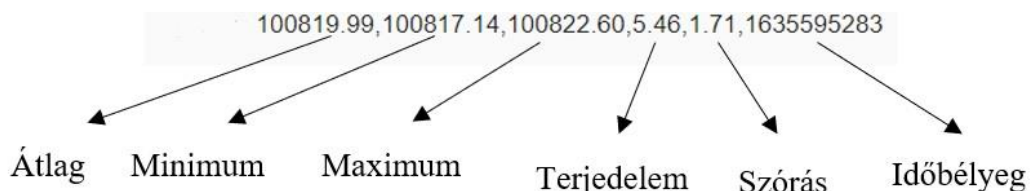
Az általam választott ESP32-WROOM típusú mikrokontroller korábban felsorolt előnyei mellett a programozhatósága is rugalmas, szabad kezet ad felhasználójának. Az eszköz kiválóan működik együtt az Arduino platformmal, ennek segítségével nagyon gyorsan és egyszerűen megírhatjuk rá a saját szoftverünket. A programozási nyelv igen egyszerű, nagyon sok funkció a fejlesztők által már előre implementálva header file-ok formájában megtalálható benne. Munkám során 5 db előre elkészített mappát használtam fel, amikben a számomra legfontosabb funkciók kezelése került megvalósításra. A programom 3 fő részből áll: a használandó kommunikációs interfészek inicializálása, a mérés szoftveres megvalósítása és a mért adatok előfeldolgozása.

A szoftver elindulását követően legelőször a szükséges paraméterek beállítása történik meg. A program ezen része csak egyszer, feszültség alá helyezéskor fut le. Itt kerül beállításra az internetes kommunikációhoz elengedhetetlen WiFi, azaz megadjuk a használni kívánt vezeték nélküli hálózat nevét, titkosítását és jelszavát. Az internetes kommunikációért felelős függvények az Arduino által biztosított *WiFi.h* könyvtárban már előre rendelkezésre állnak, így én is ezzel dolgoztam. A kommunikáció inicializálása során a *PubSubClient.h* header file segítségével megtörténik a későbbi adatkezeléshez elengedhetetlen MQTT protokoll beállítása. Az MQTT segítségével fogom majd eljuttatni a mért és előfeldolgozott adatsomagokat a szerver felé. A beállításnál meg kell adni a használni kívánt MQTT broker címét, felhasználónevét és jelszavát, illetve a kommunikációs portot (jellemzően 1883-as port). Ezt követi a BME280 típusú szenzor konfigurálása. Ez az eszköz I²C protokollon kommunikál, így a konfigurálásnál felhasználtam az Arduinoban megtalálható *Wire.h* függvény könyvtárat, amelyben már előre megvalósításra kerültek a működéshez elengedhetetlen funkciók. A szenzor vezérléséhez szükséges függvényeket a gyártó által készített és közzétett *BME280.h* file tartalmazza, így ezzel könnyedén működésre bírtam a szenzoromat.

Miután megtörtént a megfelelő paraméterek és kommunikációs protokollok konfigurálása, a szoftver belép egy végtelen ciklusba. Itt valósítottam meg a szenzor által mért adatok kiolvasását, illetve ezek előfeldolgozását és továbbítását. A szenzorból az adatok kiolvasását a *BME280.h* fileban megtalálható függvény segítségével hajtom végre.

A program a hőmérséklet, páratartalom és nyomás adatokat °C, % és Pa mértékegységekben olvassa ki. A szoftver 10 másodpercenként hajt végre 1 db mérést. Ez a mérési módszer legfőképpen az energiatakarékosság és a kiküldött adat mennyisége miatt nem egyesével továbbítja a mért értékeket a szerver felé, hanem a mérést követően az adatok eltárolásra kerülnek a mikrokontrolleren, majd 5 perc mérési eredményeit (30 darab mérési pont) összegyűjtve történik meg az előfeldolgozás és küldés. Az előfeldolgozás nagyon fontos, mivel a mért jellemzők normál körülmények között igen lassan változnak, így a 10 másodperc különbséggel vett minták között nagyon kicsi, sokszor nehezen megfigyelhető különbségek vannak.

Ezen különbségek szemléletesebbé tételéhez a szoftver a begyűjtött 30 adatpontból kiszámolja azok átlagát, terjedelmét, illetve szórását. Így sokkal könnyebben eldönthető, hogy az adott (jelen esetben 5 perces) cikluson belül történt-e bármilyen figyelemre méltó változás a mért paraméterekben. Az előfeldolgozó algoritmus amellet, hogy kiszámítja az említett jellemzőket, elkészíti az előre definiált keretformátumot is. A keretformátum a 7. ábrán látható.



7. ábra
A kiküldött adatok keretformátuma

A kiküldött adatcsomag vesszővel elválasztva tartalmazza a mért adatok átlagát, minimumát, maximumát, terjedelmét, szórását és az adatfeldolgozás során egy külső időszerverről (úgynevezett NTP szerverről) lekért időbélyeget epoch formátumban. Az így előállított adatcsomagok kerülnek kiküldésre WiFi kommunikáció segítségével az adattovábbítást végző MQTT szervernek.

4.2. Adattovábbítási lehetőségek

Az IoT eszközök esetében nagyon fontos kérdés az energiahatékonyság. A legkézenfekvőbb lehetőség erre a kommunikáció területe, a mért és előfeldolgozott adatok energiahatékony továbbítása a szenzormodul és az adattároló eszköz között. Projektem megvalósítása során a WiFi kommunikációt választottam, mivel, ahogyan korábban is jeleztem, ezen kommunikációs modul rendelkezésemre állt a mikrokontrollerbe beépítve. Ám az adattovábbítás ezen módja egy Smart Cityben nem a legalkalmasabb, hiszen egy városi környezetben nem lehet mindenhol lefedettséget biztosítani, így egy esetleges nagyobb szenzorhálózat megvalósításánál a már meglévő eszköz továbbfejlesztésére lenne szükség. Ebben az esetben sokkal kifizetődőbb valamilyen IoT specifikus kommunikációs technológia alkalmazása. Például az úgynevezett LPWAN hálózatok, amelyeket nagyobb területek energiahatékony kommunikációval történő lefedésére használnak. Ezen hálózatok kétféle módon képesek eljuttatni az általunk kiküldött adatot a szerverek felé: vagy valamilyen átjárón keresztül, vagy közvetlenül. Átjárón keresztül üzemelő hálózatok például a LoRaWAN, a SigFox és a ZigBee. Átjáró nélküli hálózatok a különböző LTE szabványra épülő kommunikációs protokollok, amelyek közül jelenleg az egyik legnépszerűbb a NarrowBand IoT (NB-IoT). [8]

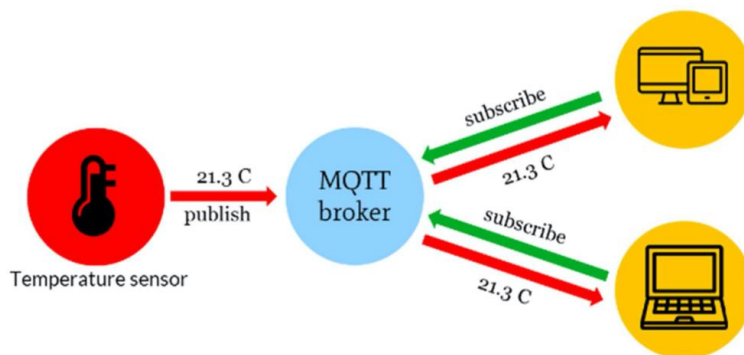
A NarrowBand IoT kis energiaigényű, nagy adatbiztonságot tesz lehetővé, emellett pedig közvetlenül képes az adatainkat eljuttatni a megfelelő szerverre. Jelenleg hazánk NB-IoT lefedettsége igen nagymértékű, két nagy mobilszolgáltató is üzemeltet ilyen hálózatot, ezért felhasználóként nem szükséges új infrastruktúra kiépítése, a speciálisan erre a kommunikációs formára kifejlesztett mikroprocesszor, illetve egy azonosításra szolgáló SIM kártya segítségével az adattovábbítás azonnal működőképes. Előnyei miatt a további fejlesztések esetén mindenképpen ezt a kommunikációs protokollt választanám.

4.3. Üzenetkezelés MQTT-vel

IoT megoldások esetében az energiahatékonyság mellett az adatok biztonságos célba juttatásáért felelős adatkezelő háttérinfrastruktúrának is nagy szerepe van. A beérkező adatcsomagokat a feldolgozásukig kezelni, illetve akár átmenetileg tárolni is szükséges. Erre a feladatra többféle, IoT felhasználásra kifejlesztett üzenetkezelő megoldás létezik. Ezek rendszerint TCP vagy UDP protokollon megvalósított adattovábbító protokollok,

melyek nagy segítséget nyújtanak az üzenetkezeléshez. Ilyen IoT specifikus protokollok például: az AMQP, a CoAP, a DDS és az MQTT. [8]

Az MQTT (Message Queue Telemetry Transport) az IBM által kifejlesztett és 1999-ben publikált szabványos publish-subscribe működésű üzenetkezelő protokoll. [9] A rendszer lényege, hogy kétféle egységet különböztethetünk meg benne: a szervert (broker) és a klienst (subscriber). Az üzeneteket úgynevezett topicokba rendezve kezeli. Az üzenetkezelés igen egyszerű: a feladó, a publisher, az általa kiküldendő adatcsomagot, illetve a topic nevét, amelybe az adatcsomag tartozik, elküldi a brokernek. A kliensek, a subscriberek, feliratkozhatnak bármely, a brokeren megtalálható topicra, a broker pedig az adott topicba érkező adatot továbbítja a kliensek felé (8. ábra). Ez a megoldás azért erőforrástakaró, mert a brokeren adattárolás nem valósul meg, a szerver erőforrásait csak az üzenet subscriberek felé történő továbbítására kell felhasználnunk. Így a rendszer könnyedén skálázható, ami kifejezetten alkalmassá teszi IoT feladatokra, ahol egyszerre akár több ezer szenzor által küldött adatot kell kezelni.



8. ábra

Az MQTT működése

Az általam kifejlesztett szenzormodul működtetéséhez azért az MQTT üzenetkezelő protokollt választottam, mert rendelkezésemre állt egy, az egyetem által üzemeltetett MQTT broker, így ez kézenfekvő megoldás volt a feladatra. Emellett a kommunikációnak igen kicsi a sávszélesség igénye, kis adatmennyiség felhasználásával képes az üzeneteket célba juttatni, így a későbbi továbbfejlesztés vagy skálázás esetén már nem okozhat problémát a megfelelő adatkezelés. A mikrokontroller által WiFi kapcsolaton keresztül továbbított adatok 1 db topicba érkeznek (BME280), amelyen belül a megfelelő

elkülöníthetőség és a későbbi feldolgozás megkönnyítése érdekében 3 db altopicba, úgynevezett subtopicba (temp, hum, pres) rendeződnek. Ezt látjuk a 9. ábrán.

```
▼ BME280
temp = 19.41,19.40,19.41,0.01,0.00,1635595283
hum = 50.39,50.36,50.43,0.07,0.02,1635595283
pres = 100819.99,100817.14,100822.60,5.46,1.71,1635595283
```

9. ábra
Az MQTT topicok

4.4. A Node-Red mint adatfeldolgozó szoftver

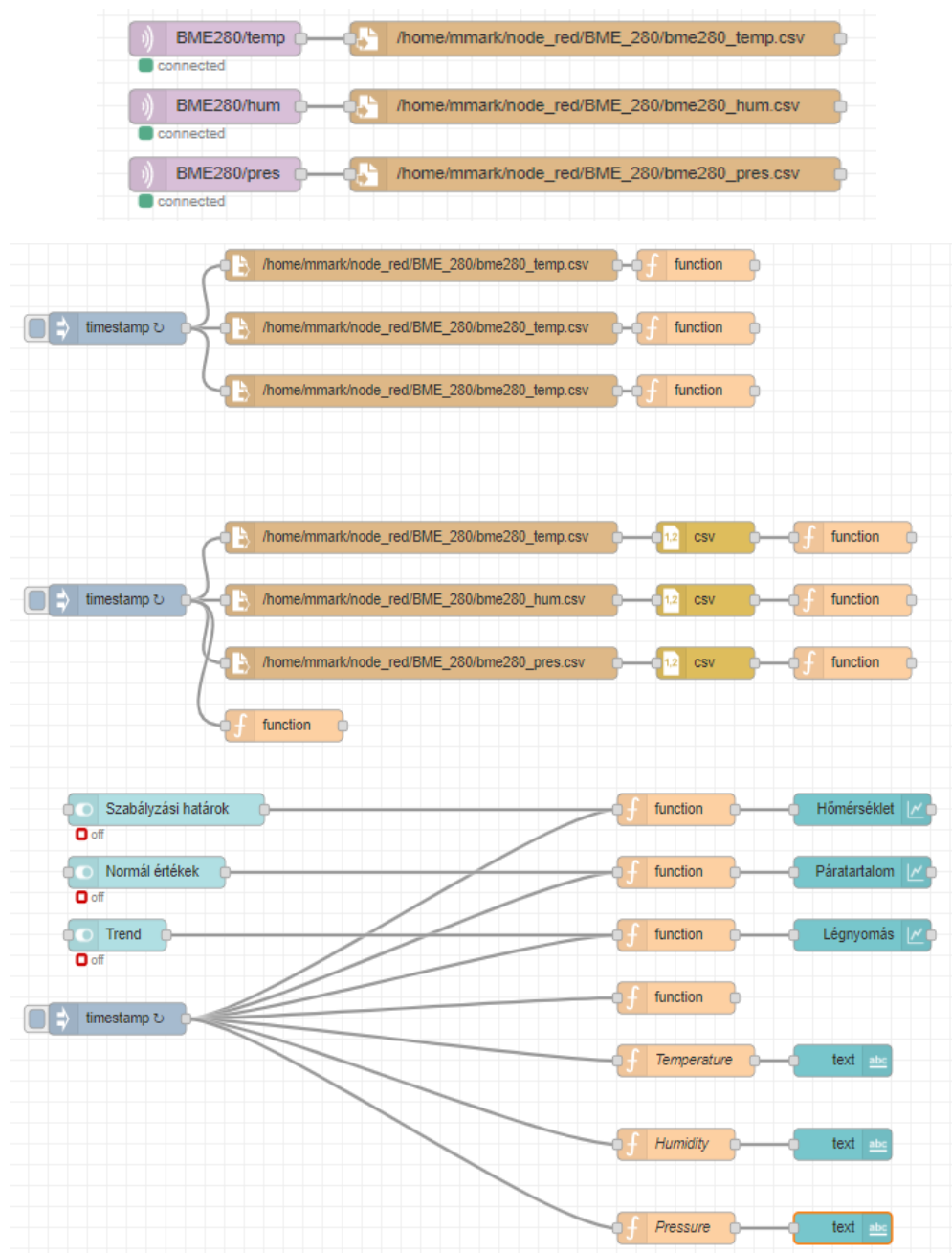
A szenzormodul által begyűjtött adat az MQTT brokeren keresztül az adatfeldolgozó egységbe mint subscriberbe érkezik. Ez egy felhőben futó szoftver. Feladata a begyűjtött adatok elemzése és feldolgozása, majd a feldolgozás után az előre megírt döntési algoritmus segítségével a döntéshozatal. Erre a célra napjainkra több nagy szoftveróriás is előállt a saját megoldásával, így a piacon megtalálható számtalan (rendszerint igen költséges) megoldás IoT adatfeldolgozásra, például a Microsoft Azure, az SAP Cloud Platform Internet of Things vagy a DELL IoT Solutions. [10]. Mint már korábban is említettem, prototípus projektem esetében a költséghatékonyságra és az arányaiban olcsó skálázhatóságra törekedtem. Hosszan tartó kutatási munka eredményeként találtam rá a blokk alapú fejlesztői eszközökre, amelyek kiválóan alkalmasak (rengeteg egyéb feladat mellett) IoT adatfeldolgozásra is. Jelenleg a piacon több száz ilyen, legtöbbször nyílt forráskódú fejlesztői eszköz található meg, például a Noflo, a Phpflo, a Flowjs vagy a Node-Red. [11] Ezek a blokk alapú fejlesztőeszközök valamilyen programnyelvre épülnek (pl.: JavaScript, PHP, Python). Jelenleg az egyik legnépszerűbb és legnagyobb felhasználó bázissal rendelkező eszköz a Node-Red, így erre esett a választásom a feladat megvalósítása során.

A Node-Red egy JavaScript nyelven íródott fejlesztő eszköz, amelynek egyik legnagyobb előnye, hogy nem szükséges hozzá semmilyen különleges program telepítése, felülete egy egyszerű webböngészőből elérhető, illetve itt tudjuk „megírni” a saját algoritmusunkat is. A blokkalapú programozás legnagyobb előnye, hogy a legtöbb funkció már a kezdetektől rendelkezésünkre áll. Például egy file beolvasását nem kell a felhasználónak leprogramoznia, hiszen arra már megtalálható egy dedikált, konfigurálható file olvasó blokk, amelyet „behúзва” és konfigurálva gyorsan és

egyszerűen megvalósítható a művelet. Emellett másik nagy előnye, hogy mivel nagy felhasználótáborral rendelkezik és nyílt forráskódú, így a felhasználók több ezer saját fejlesztésű blokkal látták már el a rendszert, amelyet bárki egy egyszerű importálással betölthet a saját környezetébe és hasznosíthat. [12] Az eszköz használatának egyetlen hátránya, hogy ahhoz, hogy az általunk megírt algoritmus folyamatosan a háttérben is fusson, egy folyamatos működésű külön szerverre van szükségünk.

Az általam megvalósított adatfeldolgozó algoritmus célja egyrészt, hogy bemutasson egy példát szenzor által mért jellemzők feldolgozására, másrészt, hogy az adatok grafikus megjeleníthetőségét is demonstrálja. Utóbbit azért tartottam fontosnak, mert, ahogyan fogalmazni szokták, ma már a „képregények korszakát éljük”, azaz az emberek általában vizuális típusúak, és emellett egy 100-150 elemből álló adatsort vizsgálva vizuálisan sokkal könnyebb észrevenni a kisebb változásokat is. Az általam megvalósított blokk alapú hálózat, amely a teljes adatfeldolgozásért felel a 10. ábrán látható. Ezen megfigyelhető, hogy az algoritmus nem tartalmaz sokféle blokkot, csak a filekezelésért, időzítésért és megjelenítésért felelősek szerepelnek benne. A valós döntéshozatalért az úgynevezett „function” modulokon belüli, JavaScript nyelven íródott algoritmusok felelnek.

Az elkészült „programomnak” két fő része van. Egyrészt a 10. ábra felső részén látható 3 blokkosor, ami felelős az MQTT szerverre való feliratkozásért, és ezzel a szenzortól beérkező adatok tárolásáért. Az adatok file-okban történő tárolására egy több száz szenzorból álló szenzorhálózat esetében a hatalmas adatmennyiség miatt nyilván nincs lehetőség. Projektemben azonban a megfelelő döntéshozatali módszer meghatározásához, és a működés helyességének ellenőrzéséhez mindenképpen szükségem volt hosszabb távú adatgyűjtésre is, amelyet így valósítottam meg. Emellett az ábrán is szemléltetett többi blokkosor hajtja végre a beérkező adatokra alapozott valós idejű döntéshozatalt, illetve ezek készítik elő az adatokat a grafikus megjelenítésre.



10. ábra

A megvalósított Node-Red háló

5. AZ ADATFELDOLGOZÁS

5.1. Az adatfeldolgozás jelentősége

Egy IoT szenzormodul feladata az adatgyűjtés. A lemért, megfelelő keretformátumú adatcsomagokat a szerverre történő beérkezésük után fel kell dolgozni. A feldolgozásra az általunk monitorozott jellemző és a döntési feladat követelményeinek megfelelően kell kidolgozni a döntési algoritmust. Ez egy olyan a program, ami a bemenetén fogadja a beérkező adatcsomagokat, majd kimenetén a megfelelő beavatkozó felé továbbítandó „kész döntés” jelenik meg. Ezen programok igen változatosak lehetnek. A működésük az egyszerű kisebb, nagyobb vagy egyenlő relációkon alapuló összehasonlítástól egészen az igen bonyolult matematikai számításokig terjedhet. Egy jól működő, akár Smart City vezérlésére is alkalmas döntéshozó algoritmus kidolgozása igen komoly feladat. A megvalósítás nem tisztán csak programozói munka, hiszen a bemeneten érkező paraméterek alapján szükséges megalkotni azokat képleteket, amelyek minden esetben a megfelelő döntést adják a kimeneten. Mindehhez komoly matematikai és statisztikai tudásra van szükség.

Ennek a folyamatnak első és legfontosabb lépéseként meg kell vizsgálnunk, hogy milyen jellemzőt vagy paramétert mér és továbbít számunkra az adott szenzormodul. Ezt követheti a kívánt feladat meghatározása, azaz ennek során választjuk ki azt, hogy mit is szeretnénk csinálni, mi lenne számunkra az ideális beavatkozási pont a rendszerbe. Például egy forgalmas kereszteződés lámpaprogramjának vezérlése esetében a bemenő adatunk a forgalomszámláló által mért forgalomsűrűség. Feladatunk pedig, hogy igyekezzünk a kereszteződés lámpaprogramját úgy hangolni, hogy az mindig a forgalom intenzitásához igazodjon; illetve egy (vagy akár több) szomszédos kereszteződés hasonló adatai alapján megjósolni az érkező forgalmat, ezzel is megpróbálni elkerülni a nagyobb dugók kialakulását. Az itt vázolt feladat megoldásához komoly háttértudásra is szükségünk van, hiszen fontos ismernünk a környék közlekedési kultúráját, az adott közlekedési csomópont adottságait, és az arra vonatkozó és kötelezően betartandó szabványokat is. Mint látható ennek az egyszerűnek tűnő feladatnak a megoldásához is több szakember szerteágazó szaktudására van szükség.

Az algoritmus által megvalósítani kívánt feladat meghatározása után a legtöbb esetben egy hosszabb távú adatgyűjtés következik, mivel minden terület más és más, é a mért paraméterek változása sem periodikus, hanem sztochasztikus jellegű a legtöbb esetben. A fenti példából kiindulva egy kereszteződés forgalomirányításának vezérléséhez célszerű először adatfelvételt tartani, amely során a használni kívánt szenzoraink segítségével monitorozzuk az adott csomópont forgalmát. Ennek segítségével megfigyelhetjük a jellemző trendeket, a csúcsforgalmi időszakokat és a szerényebb forgalmú időszakokat. Az így felvett adatsor alapján már könnyedén kidolgozhatjuk saját specifikus algoritmusunkat.

Munkám során én is a fenti folyamathoz hasonlóan jártam el. Az elkészült eszközöm, ahogyan korábban is említettem hőmérséklet, páratartalom és légnyomás mérésére szolgál. A megvalósítást követően körülbelül 1 hónapig egy lakás legnagyobb helyiségében, zárt térben adatgyűjtést végeztem. Az így begyűjtött adatsort elemezve, illetve az egyes fizikai jellemzők megfelelő értékeinek utánajárva készítettem el a saját döntési mechanizmusom megvalósítására szolgáló Node-Red algoritmusomat.

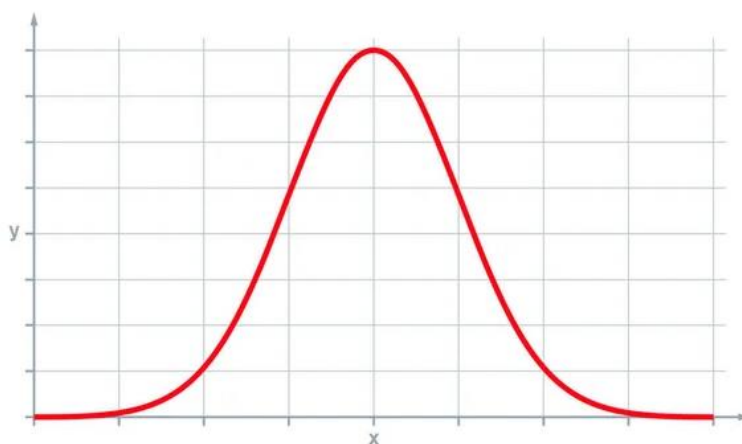
5.2. Adatfeldolgozási módszerek

Egy szenzor által mért jellemző rendszerint sztochasztikus. Elsősorban azért, mert egy fizikai jel időben változik, a múltból pedig nem lehet megállapítani jövőbeni lefutását. Emellett a mért érték zajjal is terhelt. Minden szenzornak van zaja. A választott szenzor minőségétől függően kisebb vagy nagyobb mértékű zajjal terhelt a mért adat. Ezen paraméterek mellett minden mérőeszköznek van úgynevezett mérési hibája is, ami azt jelenti, hogy az eszköz nem a pontos valós értéket méri, hanem valamilyen mértékben torzít. A mérési hiba legtöbb esetben állandó, néhány százalék körüli. A szenzorból eredő hibák mellett a kommunikáció sem százszázalékos, így előfordulhatnak kommunikációs hibák is. Ezek rendszerint igen látványosak, mivel rossz minőségű kommunikáció esetén az átlagostól nagy mértékben eltérő, rendszerint a hihetőségi határokon kívül eső érték érkezik. A hiba ezen formája könnyedén kiküszöbölhető egy egyszerű ellenőrzéssel, amely során azt vizsgáljuk, hogy a beérkező adatcsomag ép-e, illetve az adatok megfelelőek-e.

Amint korábban is említettem, a mérési eredmények véletlenszerűek, a mért értékekben szabályszerűség csak nagyon nehezen fedezhető fel. Ezért a döntéshozatal során célszerű

a múltból táplálkozni. Ez azt jelenti, hogy az előzetesen begyűjtött adatokat elemezzük, és ez alapján úgynevezett beavatkozási határokat állapítunk meg. Ezen határokon belül tekinthetjük a jellemzőnket normálisnak. Abban az esetben, ha a mért jellemző a korábban megállapított határok alá vagy felé kerül, beavatkozás válik szükségessé, amit az eltérés „iránya” határoz meg.

Ezt a döntési mechanizmust nevezzük statisztikus folyamatszabályzásnak, angolul Statistical Process Controlling (SPC)-nak. A módszer alapjait az 1920-as években a Bell laboratórium egyik munkatársa, Dr. Walter Shewart fektette le. Később, a második világháború idején Dr. W. Edwards Deming fejlesztette tovább az eljárást, amely a japán iparban került első körben alkalmazásra. Itt nagy sikereket és kitűnő gyártásminőséget tudtak garantálni a technika alkalmazásával, így később világszerte elterjedt, napjainkban is széles körben alkalmazzák. A módszer elsődleges célja a gyártási selejt minimalizálása volt. Egy gyártási folyamat során természetes módon képződik selejt. Például egy termék csomagolásával foglalkozó vállalkozás esetén a kikerülő kész, csomagolt termékek súlya eltérhet egymástól. Gyártósorról lekerülő termékek különböző paraméterei (pl. átmérő, hossz, súly) között bizonyos fokú szórás figyelhető meg. Az így megfigyelhető eloszlás az úgynevezett normális eloszlás, melyet grafikusan ábrázolva egy Gauss (vagy más néven harang) görbét figyelhetünk meg. Ez a görbe a 11. ábrán látható.



11. ábra

Gauss görbe

Ez azt jelenti, hogy a mérési eredmények többsége egy jól működő gyártósor esetén az elvárt paraméter közelében jelentkezik, míg néhány elem pozitív, illetve negatív irányban

kiugró módon elkülönül tőlük. Ez a selejt. Így a statisztikus folyamatszabályzás lényege nem más, minthogy a paraméternek az előzetes adatfelvétel alapján meghatározott tartományokon belüli ingadozását megengedi, míg kiugró értékek esetén „jelzést küld”. Ezzel a módszerrel könnyedén kiszűrhető a selejt, illetve nagyon gyorsan megállapítható, ha az adott gyártósor a természetestől eltérő mennyiségű selejtet gyárt, azaz valamilyen hiba lépett fel a rendszerben, vagy valami elromlott.

Az SPC alkalmazása során szabályozó kártyákat vezetünk be. Ehhez egy nagy mintájú adatfelvételre van szükség, melynek során az adatokat csoportosítani kell. Egy adatcsoport jellemzően több (nagyjából 10-30) adatpontból áll, például egy mérési ciklusban leért adatokból. A szabályzási határok meghatározásához szükséges ezen csoportok átlagát, terjedelmét vagy szórását kiszámítani. Az így kiszámított átlag-terjedelem vagy átlag-szórás párok és a megfelelő matematikai képletek segítségével már kiszámíthatók a beavatkozási határok. A határoknak rendszerint a 3σ (háromszoros szórás érték) szélességű sávot szokták választani, ebben a sávban tekinthető egy mérési eredmény normálisnak. A legnépszerűbb szabályozókártya az úgynevezett átlag-terjedelem kártya, mivel egy több mérési pontból álló adatcsomag két, könnyen meghatározható jellemzője a mért értékek átlaga, illetve terjedelme. Emellett egy másik gyakran használatos szabályozókártya az átlag-szórás kártya. Ennek alkalmazásához az átlag-terjedelem kártyához képest nagyobb számú (több mint 10 darab) mérési pont szükséges. A beavatkozási határok megállapításához szükséges a mért értékek átlaga, illetve szórása. Ez a módszer azért kevésbé népszerű, mint az átlag-terjedelem kártya, mivel a szórás meghatározása jóval komplexebb matematikai számításokat igényel.

A statisztikus folyamatszabályzás alatt nem kizárólag a szabályozó kártyák alkalmazását értjük. Az SPC lényege nem más, mint hogy a „régmúltat” (azaz a korábban mért eredményeket) elemezve megpróbál valamilyen hasonlóságot felfedezni a már látott és kielemezett változások, trendek és az aktuálisan megfigyelt változások és trendek között. Ezért a statisztikus folyamatszabályzás témakörébe tartozik a trendek megfigyelése is. Ez a módszer nem más, mint hogy a leért adatok változását figyeljük. Ezt több módon tehetjük meg, ám a legtöbb esetben a mért értékek fel- vagy lefutását, illetve a változás meredekségét figyeljük, értékeljük ki, majd ez alapján hozunk döntést.

5.3. A begyűjtött adatok elemzése

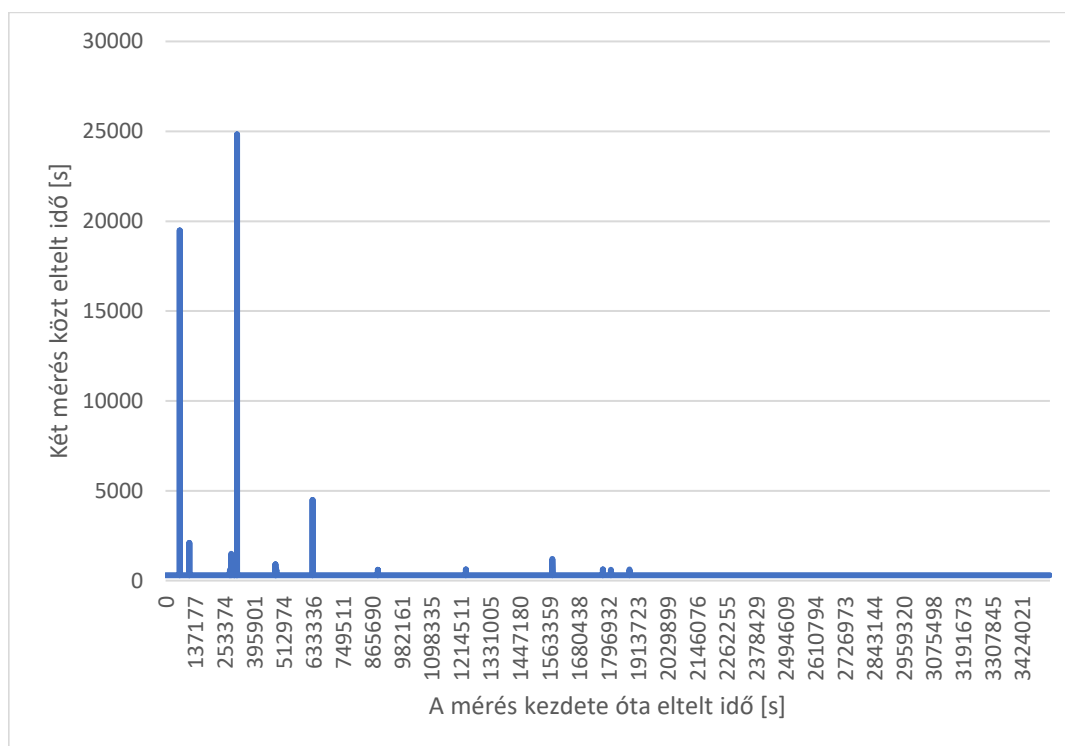
Ahogy korábban is említettem, az elkészült eszközt körülbelül 1 hónapig üzemeltettem adatgyűjtés céljából. A lehető legtöbb adatcsomag begyűjtése miatt a mérési időközt 10 másodpercre választottam, így a 30 elemből álló adatcsomag 5 perc mérési eredményeit tartalmazza. Ez az időköz csak prezentációs célokat szolgál, egy kihelyezett „valós” Smart Cityben működő szenzor esetében ezt az időtávot jóval nagyobbra célszerű választani, mivel az általam mért három jellemző változása igen lassú. Normál üzemállapotban kiugró változás nem következik be 10 másodperc alatt a paraméterekben. Egy „éles” rendszer esetén a mérési időközt eddigi tapasztalataim alapján perc nagyságrendűre érdemes megválasztani (például 1 percre), így a 30 mérés legalább egy fél óra adatait tartalmazza, amely időközben már megfigyelhetőek nagyobb változások is. Ez a módszer mindamellett, hogy nem mér feleslegesen túl sokat, energiahatékonysági szempontból is hasznos. Abban az esetben, ha a szenzormodul akkumulátorról üzemel, az akku kapacitás „beosztása” igen lényeges kérdés.

Egy IoT szenzor esetében az egyik, ha nem a legfontosabb tényező a rendelkezésre állás, hiszen már említettem, hogy egy okos városban a cél: mérni és nyomon követni a rendszer minden egyes „rezdülését”. Abban az esetben, ha egy szenzormodul valamilyen okból kifolyólag nem üzemképes a nap 24 órájában, vagy nem képes stabilan, nagyjából azonos időközönként adatokat szolgáltatni a szervernek, nem tudunk az általa mért adatokra támaszkodni, azokból megfelelő és jó döntést hozni.

Ennek tükrében az adatgyűjtés után első dolgom az eszközöm rendelkezésre állásának elemzése volt. A tesztüzem ideje alatt a szenzormodul éjjel-nappal üzemelt, így a végeredmény egy közel 11600 pontból álló adatsor lett. Az eszközöm üzembiztonságának elemzését úgy hajtottam végre, hogy a beérkező adatcsomagok időbélyegei közötti különbséget számítottam ki. Korábban említettem, hogy az adatcsomagok 5 perc adatait tartalmazzák, így két adatcsomag időbélyege közt nagyjából 300 másodperces különbség figyelhető meg. A mérési eredményeket három file-ban tároltam, így a három jellemzőt külön-külön tudtam elemezni, illetve azokból következtetéseket levonni.

Mindhárom esetben ugyanaz a grafikon adódott, így ezek közül csak egyet elemeztem részletesen.

Ahogy az a 12. ábrán is látható a szenzormodul (néhány nagyobb kiugrástól eltekintve) stabilan szolgáltatott adatokat a tesztidőszak alatt. Az adatelemzés során megfelelő időköznek a 303 másodperces értéket tekintetem, mivel az internetkapcsolat s egyéb késleltetések miatt könnyedén előállhat ez a néhány másodperces eltérés, ami így is igen alacsonynak tekinthető (jelen esetben 1%).



12. ábra

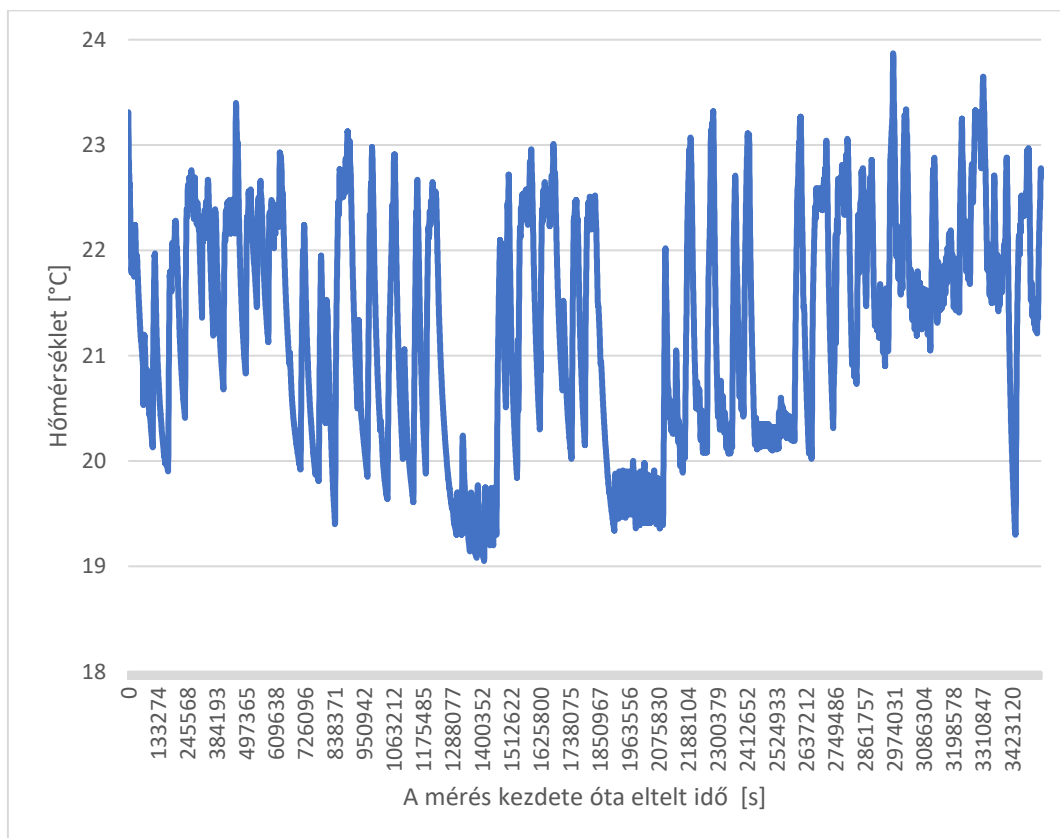
Mérési időközök

Az adatgyűjtés során 15 adatpont volt, amely az előző mérési eredményhez képest több mint 303 másodperces időkülönbséggel került feldolgozásra. A grafikonon látható néhány kiugró eredmény is, ezek számszerűsítve: 1000 másodpercnél nagyobb időkülönbség 6 helyen figyelhető meg, míg 10000 másodpercnél nagyobb időkülönbséget csupán 2 helyen figyelhetünk meg. Ez azt jelenti, hogy a teljes 11592 adatpontból mindössze 15 hibát fedezhetünk fel, azaz a mérések 99,871%-megfelelő időközzel történt. A hibáknak több oka is lehet. Egyrészt a mérés ideje alatt egy kisebb áramszünet is volt, így ez alatt nem volt tápellátása az eszköznek, amely így nem tudott üzemelni. Ez okozta a grafikon legnagyobb kiugrását. A többi eltérésre többféle magyarázat is lehetséges. Egyrészt az adat mérésétől a szerverig történő eljutása igen

bonyolult folyamat, így bármelyik alkotóelemben egy kisebb hiba vagy üzemzavar okozhatja a nagyobb időkülönbséget. Másrészt a kommunikáció sem hibátlan, így az esetleges internet elérési vagy egyéb kommunikációs hibák is jelentkezhetnek ilyen formában.

5.3.1. Hőmérséklet

A mért értékek közül először a hőmérsékleti adatok elemzésével foglalkoztam. Első lépésként a begyűjtött adatokat ábrázoltam. Ez a grafikon a 13. ábrán látható.

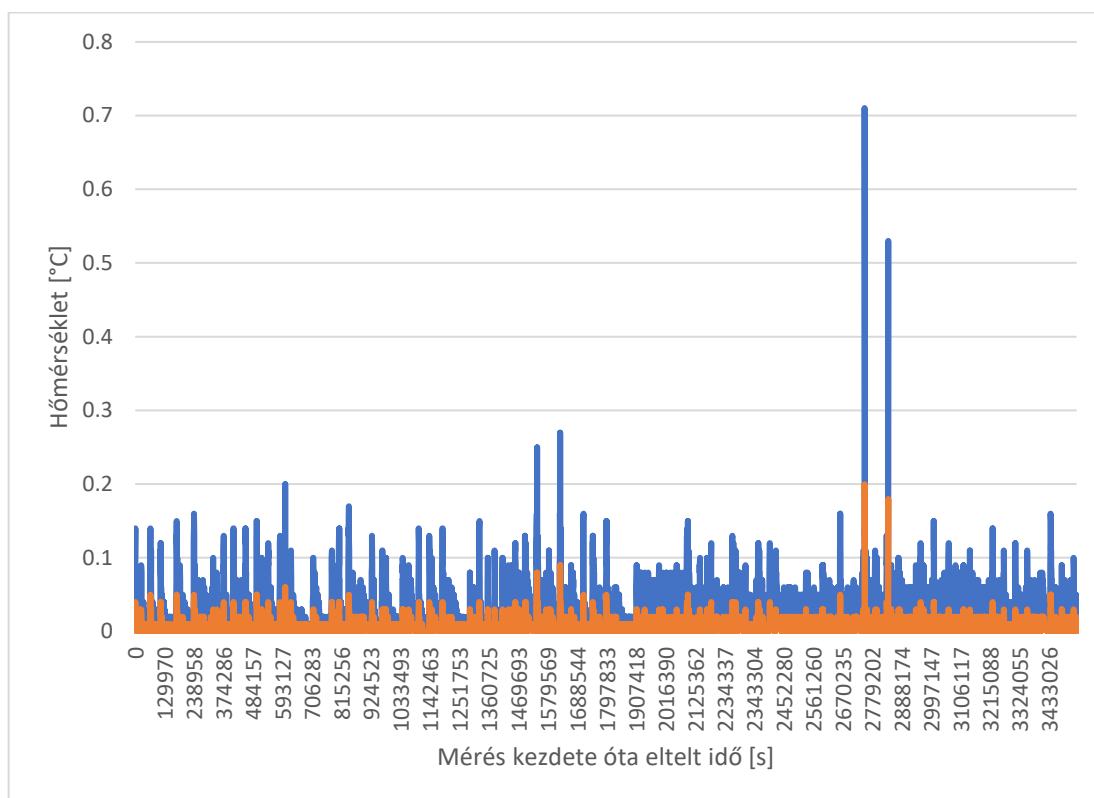


13. ábra
Hőmérsékleti adatok

Megfigyelhető, hogy a mért értékek helyenként ingadoznak, helyenként pedig viszonylag stabilan egy adott érték környezetében tartózkodnak. Emellett látható, hogy a grafikon két szélső értéke viszonylag közel helyezkedik el egymáshoz. Az értékek maximuma

23,87 °C, minimuma pedig 19,05 °C. Ez is mutatja, hogy a szenzor beltérben üzemelt, hiszen a hőmérsékleti értékek terjedelme mindössze 4,82 °C. A kicsi kilengésből látható az is, hogy a lakásban termosztát üzemel. Ezt támasztja alá a mért adatsor módusza (22,47 °C) és mediánja (21,4 °C) is.

A korábban is említett lassú változásra, illetve az indokolatlanul kicsi mérési időközre bizonyíték az adatsor szórása és terjedelme. Ahogyan az a 14. ábrán is látható 2-3 nagyobb kiugrástól eltekintve 0,1-0,2 °C körül értékek adódnak a terjedelem, míg a szórás ennek körülbelül a tizede. Ebből megállapítható, hogy amellet, hogy a jellemző lassan változik a mérési intervallumon belül, az értéke stabilan emelkedik, csökken vagy éppen stagnál. Nem „ugrál”, erre bizonyíték az alacsony szórás érték.

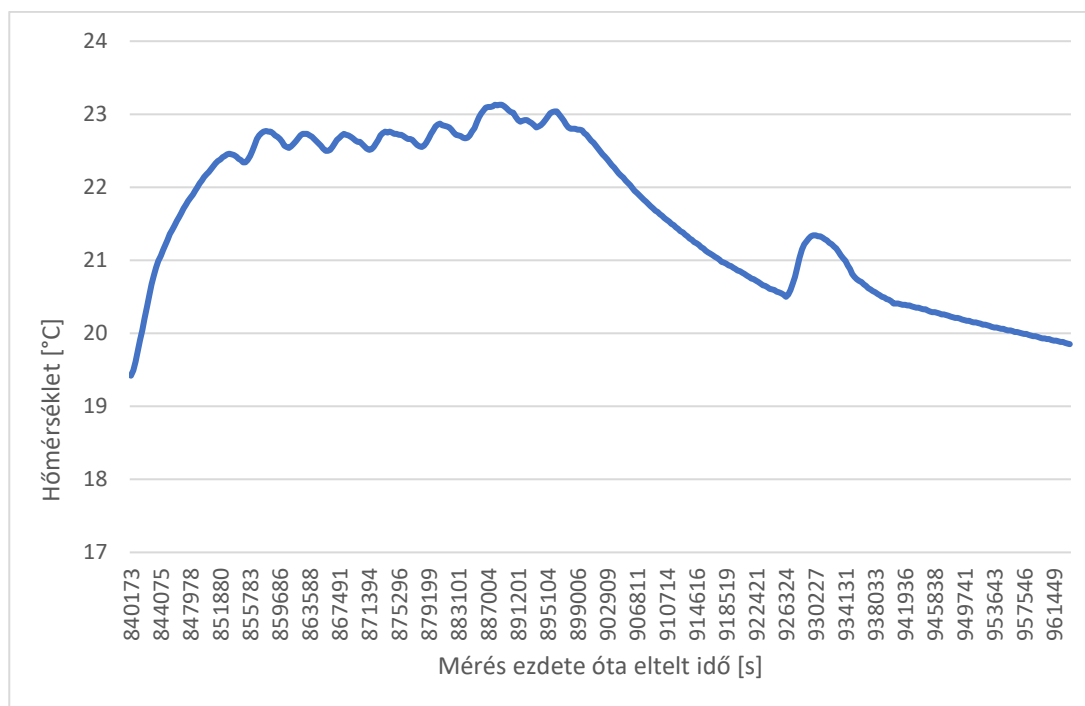


14. ábra
Szórás és terjedelem viszonyok a mérés során

Nagyobb mérési időköz és kültéri alkalmazás esetén ezek az értékek minden bizonnyal nagyobbak lennének, mivel egy hosszabb, 30-40 perces időtartamon belül szabadtéren nagyobb, akár ugrásszerű változások is megfigyelhetők.

Amint már említettem, a helyiségben termosztát és gázkazán biztosítja a megfelelő hőmérsékletet. A termosztátnak két állása van: nappali hőmérséklet (22 °C) és éjszakai hőmérséklet (20 °C). A grafikonon ez a két állapot kiválóan megfigyelhető. A 13. ábra grafikonján négyszög jelhez hasonló változások figyelhetők meg. Ennek oka a termosztát módváltása. A négyszög jel maximuma kb. 22 °C, míg minimuma kb. 19 °C. A „kitöltési tényező” pedig a két állapot közötti periodikus váltogatásnak köszönhető. Az állapotok közti váltogatás viszont nem hozható összefüggésbe a napszakokkal, mert előfordul olyan, hogy a fűtés nappal is éjszakai állapotban üzemel, mivel nem tartózkodik éppen a lakásban senki, emiatt nem szeretnénk feleslegesen fűteni sem. Emellett megfigyelhető egy hosszabb minimum is a görbén. Ez annak köszönhető, hogy egy hétvégén elutaztunk vidékre, így a lakás üresen állt, a fűtés pedig éjszakai üzemmódban üzemelt ebben az időszakban.

A hőmérséklet grafikon egy kinagyított részlete a 15. ábrán figyelhető meg. Látható, hogy a grafikon 0 időpillanatában (ami a mérés kezdetétől számított 840173-ik másodperc) megtörténik a termosztát nappali üzemmódra történő átkapcsolása. A kazán fűteni kezd, a hőmérséklet pedig eléri a kívánt hőmérsékletet. A termosztát hőmérsékleti érzékelője a szenzormodulomtól távolabb helyezkedik el, illetve a szenzorok hibájából is adódhat, hogy a hőfok valamivel 22 °C fölé áll be. Ezt követően a termosztát körülbelül 12 órán keresztül ezen az értéken működött. Látható, hogy az eszköz nem stabilan a beállított értéken, hanem annak környezetében tartja a hőfokot. Ennek oka nem más, minthogy a termosztát az egy hiszterézises komparátornak feleltethető meg. Ez azt jelenti, hogy kazánt úgy üzemelteti, hogy az valamivel a beállított érték fölé emelje a hőmérsékletet, majd kikapcsolja. Abban az esetben, ha a hőmérséklet a beállított szint alá esik, ismét bekapcsol a fűtés. Ha stabilan a kívánt hőmérsékletet próbálnánk meg tartani, az azt eredményezné, hogy a kazánunk nagyon sűrűn (akár néhány másodpercenként) kapcsolna ki és be. Ez egyrészt egyáltalán nem lenne energiahatékony, másrészt a berendezést is könnyen tönkretethetné a rengeteg ki- és bekapcsolás.



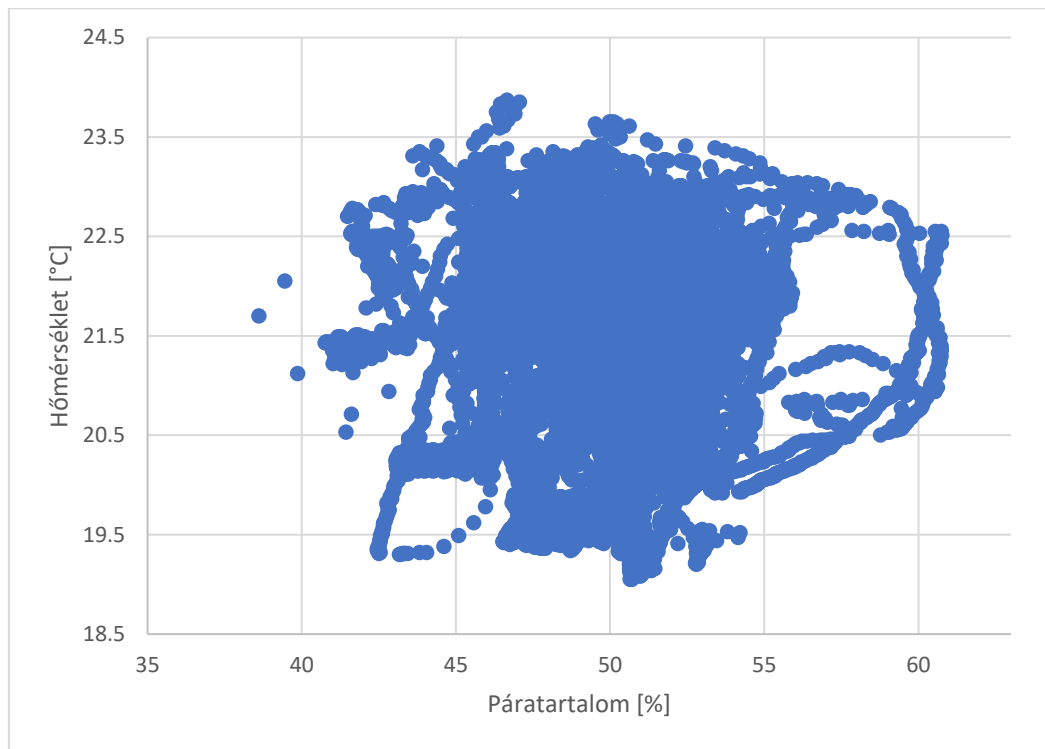
15. ábra
Termosztát működése

A grafikon közepén (a mérés kezdetétől számított 898106 másodpercben) a termosztátot ismét éjszakai üzemmódba kapcsolták, így az nem fűt tovább, hagyja a hőmérséklet szabadon „lefutni” az alsó, 19 °C-os határértékig. Látható, hogy a hőmérséklet lefutása a melegedésnél lankásabb, azaz a környezet lassabban hűlt le, mint ahogyan felmelegedett. Ez a lakás, a benne lévő berendezési tárgyak és a fűtési rendszer hőkapacitásának köszönhető. Ezek a külső energiabefektetés megszűnése után még képesek átadni a magukban tárolt hőenergiát a levegőnek, ezzel ellensúlyozva a levegő hőmérsékletének a falakon keresztül történő természetes visszahűlését.

5.3.2. Páratartalom

A páratartalom a hőmérséklet mellett egy másik fontos jellemző. A levegő páratartalmát általában százalékos formában, úgynevezett relatív páratartalomként mérjük. Ez azt jelenti, hogy az adott hőmérsékletű levegő a maximális pára kapacitásának hány százalékát érte el aktuálisan. Például a 20 °C-os szobahőmérsékletű levegő 17,3 g nedvességet képes felvenni köbméterenként, míg ugyanez az érték 15 °C-on már csak 12,8 g/m³. Ebből látható, hogy a relatív páratartalom erősen függ a levegő

hőmérsékletétől. Ha a levegő eléri a maximális nedvességtartalmát, páralecsapódás jön létre. Ez kültéren köd (fagypont alatt dér vagy zúzmara) formájában jelentkezik. Azt a hőmérsékletet, ahol a levegő nedvességgel telítetté válik, harmatpontnak nevezzük. A hőmérséklet és a páratartalom összefüggését a 16. ábra bizonyítja.



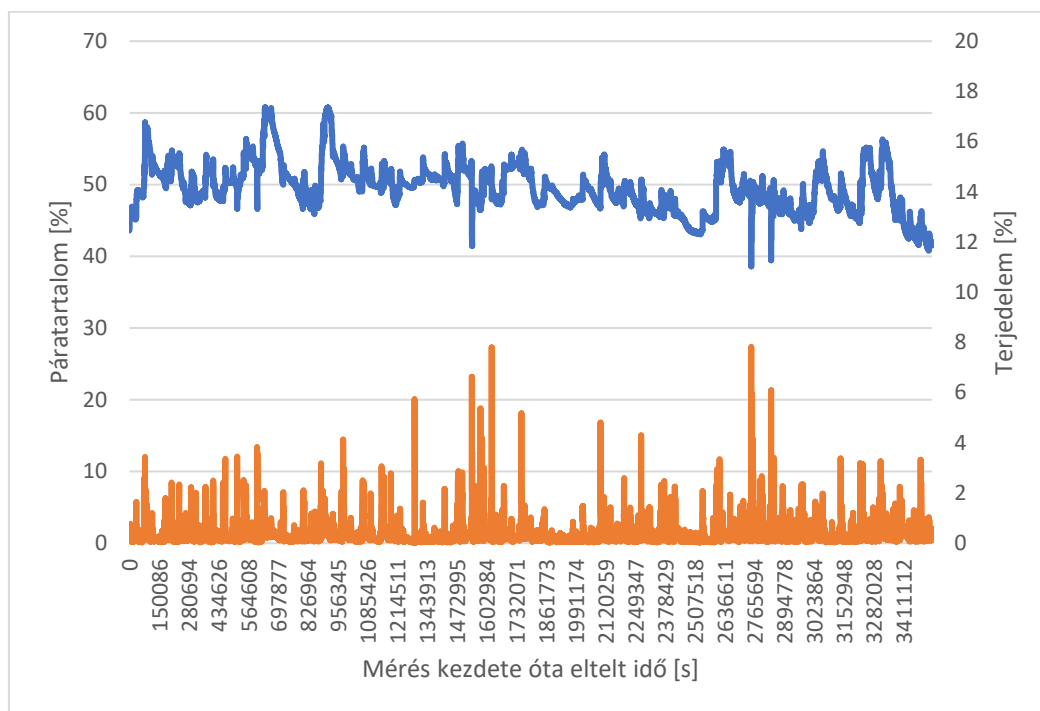
16. ábra

Páratartalom-hőmérséklet összefüggés

Ezen látható, hogy ha a páratartalmat az x tengelyen, a hőmérsékletet pedig az y tengelyen ábrázoljuk, a diagram egy „nagy folt” lesz. Látható néhány érdekesség is az ábrán. A „kevésbé zavaros” részeken például növekvő és csökkenő tendenciák figyelhetők meg. Ennek oka a levegő nedvességtartalmának változása, miközben hőmérséklete állandó volt. Ezek általában lineáris és exponenciális változások. Emellett a grafikon szélein van néhány „eltévedt” pont is. Ennek egyik lehetséges oka a levegő nedvességtartalmának azonos hőmérséklet melletti nagymértékű változása, de mérési hiba is lehet. Ezt igen nehéz eldönteni. A grafikon egyes részein a pontthalmaz „besűrűsödik”. A jelenség oka nem más, minthogy ezeken a helyeken a levegő tényleges nedvességtartalma nem

változott, a hőmérséklet viszont igen, és ezzel együtt (az ismert összefüggés miatt) a relatív páratartalom is.

A hőmérséklettel ellentétben a páratartalom nem stabil, néhány kiugró érték, illetve változás is megfigyelhető a grafikonján (17. ábra).



17. ábra

Páratartalom és terjedelme az idő függvényében

A hőmérséklet elemzésével ellentétben itt célszerűbb a mért értékeket és annak terjedelmét egy közös grafikonon két y tengellyel ábrázolni, hiszen ennél a paraméternél komolyabb változások is megfigyelhetők. A nagyobb kiugrások és beszakadások nem minden esetben vannak összhangban a terjedelemmel. Ennek oka, ahogyan a hőmérséklet-páratartalom grafikonon is látszik, a jellemző folyamatos (közel lineáris) változása. Ebben az esetben az adatcsomag terjedelme nem mutat kiugróan magas értéket, hiszen a változás folyamatos. A terjedelem görbén a változásra mindössze annyi utal, hogy a terjedelem értékek „pálcikái” egy kisebb dombot alkotnak a grafikonon, illetve a felette lévő kék vonalban is változás figyelhető meg. Ezzel ellentétben azonban néhány ponton a páratartalom változásával összhangban kiugró terjedelem értékek jelennek meg. Attól a hirtelen változástól, ami (hirtelen csökkenés esetén) például egy szellőztetéssel

jár. Emellett előfordul még olyan kiugró terjedelem érték is, amelyhez nem tartozik lényeges változás a páratartalom adatban. Ezt az okozza, hogy a mérési cikluson belül a jellemző két szélső érték között mozog, nem folyamatos és egyenletes változást mutat. Általában valamilyen zavar, mérési hiba, vagy a hőmérséklet változása vezet ehhez.

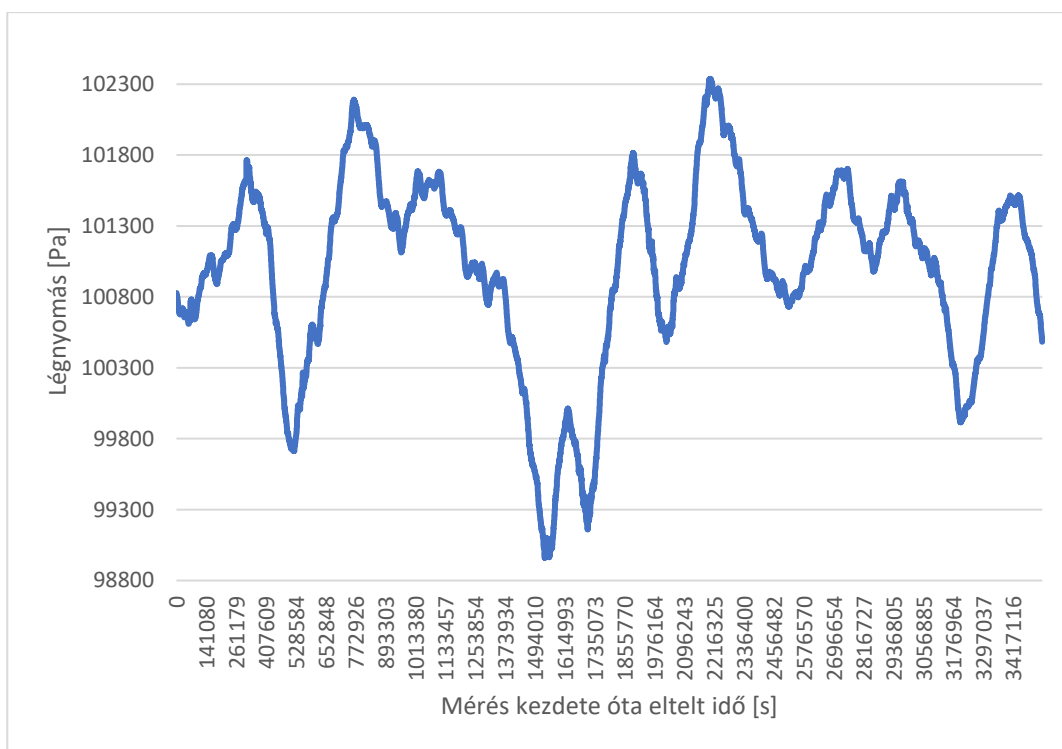
5.3.3. Légnyomás

Az eszköz által mért harmadik jellemző a meteorológia számára igen hasznos légnyomás. Ugyan a szenzor jelenleg beltérben üzemel, ám a lakás nincsen hermetikusan elzárva a külvilágtól. Ennek következtében a helyiségben mérhető légnyomás érték gyakorlatilag teljes mértékben megegyezik a kültéren mért értékekkel. Ennek ellenőrzésére az adatfelvétel ideje alatt többször is megnéztem a meteorológiai szolgálat által közzétett és a szenzorom által mért értékeket, amely rendszerint ugyanaz volt.

A légnyomás értékét a felületegységre eső légoszlop súlyából származó nyomóerő adja. Ennek mértékegysége a pascal [Pa], ám a standard légnyomás 10^5 Pa, így előszeretettel használják a kPa mértékegységet. A légnyomás a földfelszínen nem egyenlő nagyságú. A felszíntől felfelé haladva a levegőoszlop mérete, így annak súlya is csökken, ezért a nyomásérték is csökken. Ez a változás exponenciális jellegű, ám függ a levegő hőmérsékletétől is, emiatt nem tisztán exponenciális csökkenést figyelhetünk meg. Egy adott pontra a tengerszint feletti magasságának ismeretében kiszámítható a standard nyomás értéke az alábbi képlettel: $p_{standard} = 10^5 - \frac{h}{15500}$, ahol h az adott pont tengerszint feletti magassága. A képletbe behelyettesítve a szenzormodulom által standardnak tekinthető érték (a szenzor a tengerszint felett kb. 110m el helyezkedik el) $p_{standard} = 101200 \text{ Pa}$.

A levegő nyomásának változása összefüggésben van az időjárással. Időjárásunkat frontok és frontrendszer alkotják. A frontok mozgását rendszerint légnyomás változás előzi meg. A meteorológia nemcsak egy ponton méri a jellemzőt, hanem több ponton elhelyezkedő mérőállomások segítségével végzi az előrejelzéseket. Egy területen a légnyomás értékek eltérhetnek, az azonos nyomású pontok úgynevezett izobárokat alkotnak. Az izobárok elhelyezkedéséből és mozgásából könnyedén megállapítható egy front vagy frontrendszer mozgása is. A 18. ábrán az 1 hónap alatt begyűjtött adatok láthatóak. Megfigyelhető, hogy a grafikon „hullámzik”, periodikus jelleget hordoz. Ennek oka, hogy 1 hónap alatt időjárásunk igen változatos. Hideg- és melegfrontok

követik egymást, napsütéses napok után akár hosszabb esős időszak is következhet. Az adatfelvétel időtartama alatt jellemzően száraz, többször napos időjárásban volt részünk. Ezt a görbe is alátámasztja, hiszen néhány komolyabb mélyrepülést leszámítva az értékek a viszonylag magasnak tekinthető 100 kPa-os érték felett helyezkednek el.

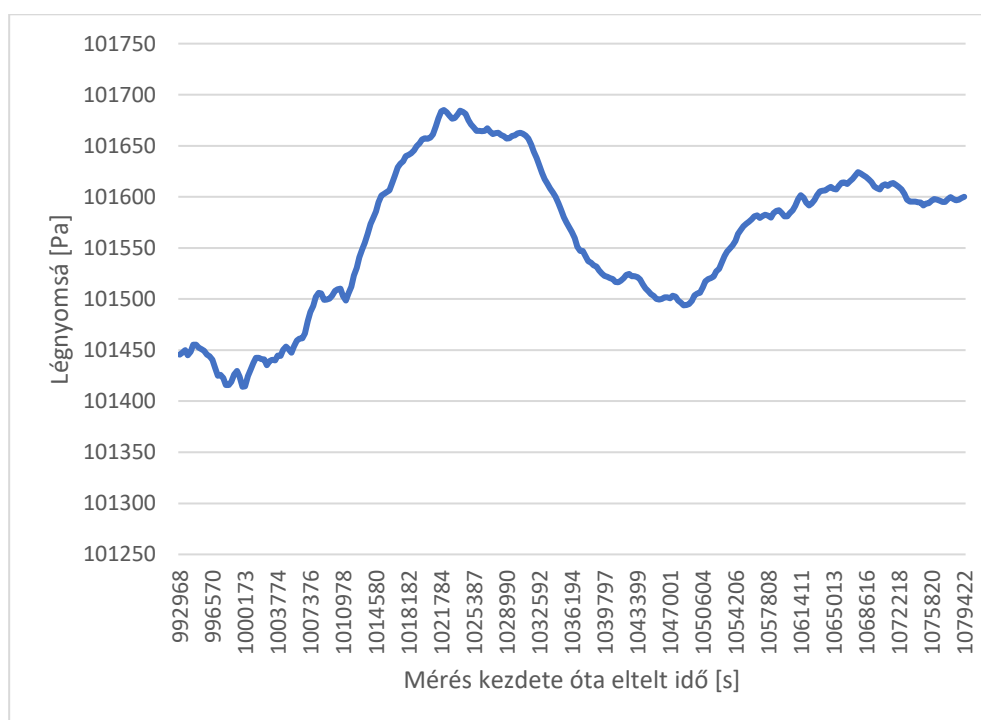


18. ábra
Légnyomás értékek

Ahogy említettem, megfigyelhető 2-3 komolyabb beszakadás is. Azokban a napokban, amikor ezek keletkeztek, időjárásunk változást hozott. Rendszerint a napos, száraz időszakot csapadékosabb, felhős idő váltotta. De látható néhány kiugrás is a görbén, ezekhez nem társult komolyabb időjárás változás, mindössze annyit észlelhettünk, hogy egyik napról a másikra felhők érkeztek, vagy újra naposra fordult az idő. Ezek valószínűsíthetően gyengébb, száraz légtömeget hozó frontok voltak, amelyek nem befolyásolták számottevően időjárásunkat, ám légnyomás értékekben visszaköszöttek.

A légnyomás viszont nemcsak frontok és időjárási jelenségek hatására változik. Létezik egy 24 órán belüli periodikus ingadozása is. Ez két maximum és két minimum hellyel jár. A maximum pontok délelőtt 10 és este 22 órakor, míg a minimum értékek hajnal 4 és

délután 16 órakor állnak be. Jellemzően a nappali, azaz a 10 és 16 órai szélső értékek látványosabbak, könnyebb őket azonosítani. Ezt a változást nekem is sikerült megfigyelnem. A 19. ábrán 24 óra légnyomás értékeit ábrázoltam. ezen könnyedén megfigyelhető, hogy a grafikon „lejtmenetben” indul, majd eléri a minimumát (kb. 4 órakor). Ezt követően látványos emelkedésbe kezd, majd valamivel a nap közepe előtt ér fel a csúcsra. Ez egy erősen kirajzolódó szélsőérték, az ábrázolt tartomány abszolút maximuma is egyben. A maximumot követően csökkenő tendenciát vesz fel a folyamat egészen 16 óráig, amikor ismét minimumot ér el. Ez az egyik helyi minimum érték. Ezt követően ismét emelkedni kezd a légnyomás, majd a 24 órás ciklus végéhez közeledve helyi maximumot ér el. Ezt a napi, periodikus változást a Nap és Hold ciklusok idézik elő. Meredek változások esetén kevésbé látványos ez a tendencia.



19. ábra

24 órás légnyomás érték

5.4. Az adatfeldolgozás előkészítése

Egy IoT rendszerben a feldolgozott adatok segítségével tudjuk támogatni a döntéshozatalt, és bemeneti paramétereket szolgáltatathatunk a különböző döntéshozó algoritmusoknak. A beérkező adatok esetében a tényleges feldolgozás előtt fontos az adatcsomagok ellenőrzése. Ennek során végignézzük a beérkező adatokat, hiszen a sok mérés között bármikor előfordulhat hibás (a technológiai értékhatároknak nem megfelelő) érték vagy kommunikációs zavar is, melynek következménye hiányos adatcsomagok érkezése. Mindezeket még a tényleges adatfeldolgozás előtt eltávolítjuk. A mérési eredmények feldolgozására többféle módszert alkalmazhatunk: statisztikai módszereket, valószínűségszámítási módszereket, de segítségül hívhatjuk a legmodernebb informatikai megoldásokat is, azaz a tanuló algoritmusokat (pl.: neurális hálózat). Ezt a feldolgozási folyamatot végzi esetemben a Node-Red.

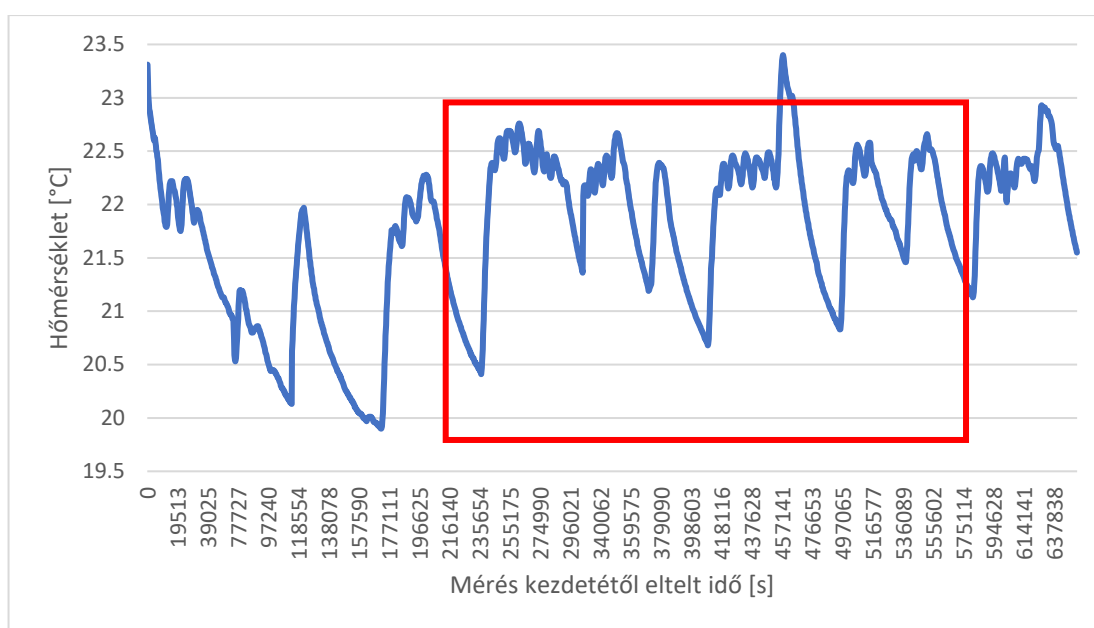
A szenzormodul által az MQTT brokeren keresztül a Node-Redbe eljutatott adatcsomagok 6 elemű, vesszővel elválasztott karaktorsorozatok. Ezt a karaktorsorozatot alakítom át a Node-Redben megtalálható blokk segítségével JSON objectté, amit a JavaScript programozási nyelv kiválóan kezel. Az adatok ellenőrzése az átalakításkor történik meg, így hibás adatcsomag feldolgozása nem kerülhet sor. A döntéshozó algoritmusok megvalósítása során a legfontosabb szempont az volt, hogy megpróbáljak képet adni szenzormodul által gyűjtött adatok feldolgozási lehetőségeiről, ezért az általam mért három különböző jellemzőt három, egymástól különböző módszerrel értékeltem ki.

5.5. Hőmérséklet-kontroll beltérben

A hőmérséklet értékek a szenzormodul elhelyezkedéséből adódóan közel állandóak, ahogyan korábban is látható volt, mindössze kisebb eltérések tapasztalhatóak. Ezért az általam választott feldolgozási módszer a szabályozó kártyák bevezetése volt. Legnagyobb előnye, hogy segítségével azonnal képesek vagyunk észlelni, ha egy paraméter az általunk meghatározott „optimális” tartományból pozitív vagy negatív irányba kilép. Ez abban az esetben hasznos igazán, ha az „optimális” tartományt a termosztát által tartott hőmérsékletre állítjuk, mert így azonnal észlelhetjük, ha a fűtési rendszerünkben valamilyen rendellenesség következik be, vagy azt is, ha valaki (akár véletlenül) elállította a termosztát hőmérsékletét.

Mint korábban jeleztem, a szenzormodulom az 5 perc alatt mért hőmérsékletek átlaga mellett a mért értékek terjedelmét és szórását is továbbítja, így könnyedén felvehettem egy úgynevezett átlag-szórás kártyát, amit célszerűbb alkalmazni, mint az átlag-terjedelem kártyát, mivel rendelkezésre áll a szórás értéke és a minta nagysága is elég nagy (30 db). [13]

A szabályzási középérték meghatározásához egy úgynevezett „alapvonal” alapján vettem fel az értékeket. Az előzetes adatfelvételből kiválasztottam egy kisebb szakaszt, amelynek a grafikonja a 20. ábrán látható.



20. ábra

Hosszútávú hőmérsékleti adatok

A későbbiekben ezzel dolgoztam tovább. Ez egy 1200 elemből álló sorozat volt, amelyet az ábrán piros négyzettel jelöltem. Ezt követhette volna az átlag-szórás kártya értékeinek meghatározása, ám szembesülnöm kellett vele, hogy a hőmérséklet a lakásban igen-igen lassan változik, ezért az 1200 adatpontból először kiszámított beavatkozási határok nagyon kevés mozgást engedtek volna. Erre azt a megoldást alkalmaztam, hogy az 1200 pontot szintén 30-as csoportokba gyűjtöttem, kiszámítottam az így keletkezett adatsorok átlagát és szórását, majd a kapott eredményeket használtam fel a megfelelő képletekbe történő behelyettesítésnél. A beavatkozási határok megállapításához először

ki kellett számítanom az átlagok átlagát ($\bar{\bar{X}}$) és a szórások átlagát ($\bar{S}$). Ezt követte az átlag alapján az alsó és felső beavatkozási határ kiszámítása. A számítás során a szokásos 3σ szórástávolság helyett a hőmérsékleti jellemző számára valamivel nagyobb „mozgásteret” biztosító 5σ értéket választottam, így ki tudtam számítani az A_5 konstanst az $A_5 = 5/(c4_{30} \sqrt{30})$ képlettel (21. ábra).

Innen már csak az alsó és felső beavatkozási határértékek meghatározása maradt hátra, amit az $UCL_x = \bar{\bar{X}} + \bar{S}A5_{30}$, $LCL_x = \bar{\bar{X}} - \bar{S}A5_{30}$ képletekkel számítottam ki [14]. Ezen műveletek alapján az alsó és felső beavatkozási határértékek a következők: $UCL_x = 22.59^\circ C$ $LCL_x = 20.96^\circ C$.

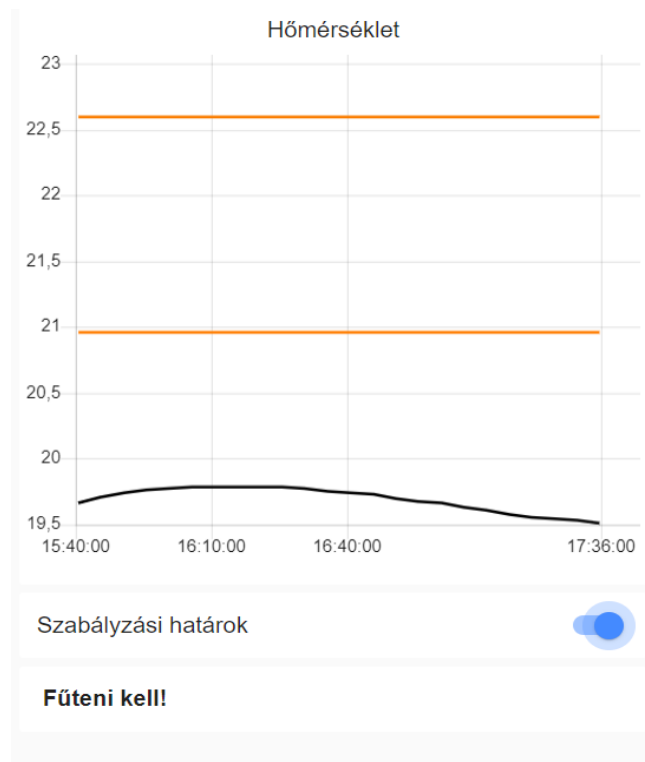
n ▲	c4 ♠	A3 ♠	B3 ♠	B4 ♠
27	0.9904	0.5829	0.5820	1.4180
28	0.9908	0.5722	0.5899	1.4101
29	0.9911	0.5621	0.5974	1.4026
30	0.9914	0.5525	0.6044	1.3956

21. ábra

Számításhoz szükséges konstansok [14]

A mért hőmérsékletet akkor tekinthetjük „normálisnak”, ha az a két szélső beavatkozási érték közé esik. Ezek segítségével már könnyedén észlelhető a rendellenesség. A döntéshozatali algoritmus eredményeinek szemléltetéséhez hívtam segítségül a Node-Red grafikus megjelenítési lehetőségét. A felület a 22. ábrán látható.

Megfigyelhető, hogy a mért értékeket kb. 2 órára visszamenőleg egy grafikonon is ábrázoltam, illetve feltüntettem a két beavatkozási határértéket is (utóbbi egy gombbal ki is kapcsolható). A grafikon alatt pedig a döntéshozatal aktuális eredménye jelenik meg. Vezérlendő objektum (valamilyen fizikai eszköz) hiányában erről egy felirattal tájékoztatom a felhasználót. A két beavatkozási határhoz kétféle szöveg tartozik: a „Fűteni kell!” (a 22. ábrán is ez látható), illetve a „Ki kell kapcsolni a fűtést!”. Ez a felület, ahogyan a Node-Red is, könnyedén elérhető bárki számára, hiszen böngészőből megnyitva akár a telefonunk segítségével is rápillanthatunk a lakás aktuális hőmérsékletére, illetve tájékozódhatunk az esetleges rendellenességekről is.



22. ábra

Hőmérséklet-kontroll panel

5.6. Döntési alternatívák a páratartalom ismeretében

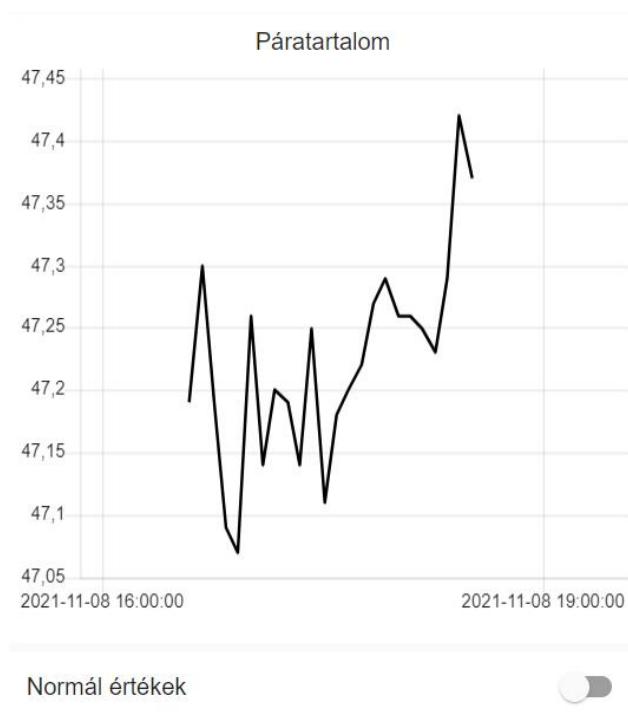
A páratartalom a hőmérséklet mellett egy másik fontos légköri jellemző. A páratartalomhoz kapcsolódó döntéshozó algoritmus az előzőnél jóval egyszerűbb megoldás, mivel ebben az esetben az alsó és felső beavatkozási határérték megválasztásánál a természettudományt hívtam segítségül.

Kevésbé közzismert tény, de hőérzetünk nagymértékben függ a levegő páratartalmától. Képesek vagyunk fázni 25°C -ban is, de 19°C -ban is lehet melegünk. Emellett a páratartalom befolyásolja a közérzetünket is: túlságosan nedves levegőben az emberi szervezet nem képes a megfelelő mennyiségű folyadékot elpárologtatni, ami gyakran lehet oka a rossz közérzetnek vagy „nyomott hangulatnak”. De nemcsak életünkre van nagy hatással ez a légköri jellemző, hanem épületeink, lakóházaink épségének megőrzésében is fontos szerepet játszik. Olykor otthonainkban csak annyit érzékelünk, hogy a fal egy-egy helyen melegebb, máshol pedig hidegebb. Ez azt jelenti, hogy a hideg falfelületeken úgynevezett hőhíd keletkezett, ami sarokpontokban vagy különböző

hővezetési tulajdonságú anyagok találkozásánál alakulhat ki. Ilyen helyeken a túl magas nedvességtartalmú levegő könnyen harmatpontja alá hűlhet, ami párakicsapódással jár, s ez melegágya a penészgombáknak. A jelenség amellett, hogy kevésbé esztétikus, az egészségre is ártalmas [15].

Egy lakóház páratartalma körülbelül 40 - 60 százalék között tekinthető optimálisnak [16], így a rendszerem alsó és felső beavatkozási határértékét is ennek tükrében választottam meg. Ezen értékek alatt és felett az algoritmus jelzést küld felhasználójának a grafikus felületen keresztül. Túlságosan alacsony páratartalom esetén párasítással lehet növelni ezt az értéket, míg magas értékek esetén (ami gyakoribb) páramentesítést kell alkalmazni a probléma megoldására. Hosszan tartó magas légnedvesség tartalom viszont gyanakvásra is okot adhat, mivel előjele lehet egy komolyabb falnedvesedésnek, ami rossz szigetelésből adódhat vagy csőtörés miatt keletkezhet.

Az általam megvalósított visszajelző felület a 23. ábrán látható. Itt is, csakúgy, mint a hőmérséklet esetében gombbal megjeleníthetőek a beavatkozási határértékek. A felső küszöbérték túllépése esetén „Páramentesítés szükséges!”, míg az alsó beavatkozási határ esetén „Párasítás szükséges!” visszajelzéseket ad a döntési algoritmus.



23. ábra

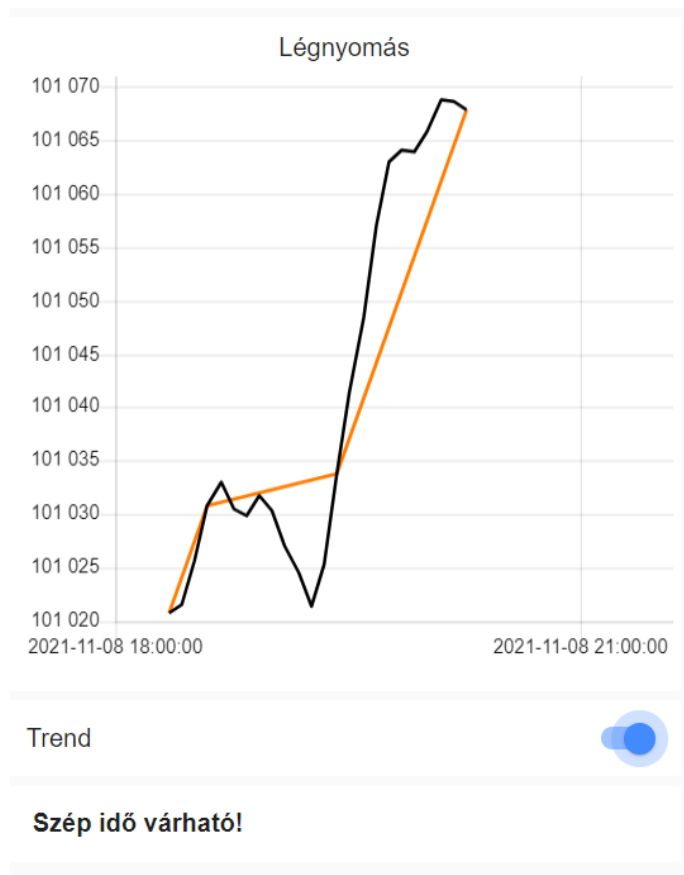
Páratartalom megjelenítése

5.7. Időjárás-előrejelzés légnyomás adatok felhasználásával

A szenzormodulom által mért harmadik jellemző a légnyomás. A légnyomás aperiodikus változásait rendszerint valamilyen légköri jelenség idéz elő: frontok érkezése, azaz meleg és hideg légáramlatok találkozása. Melegfront esetében általában nyomásnövekedést, míg hidegfrontok esetében nyomáscsökkenést tapasztalunk. Ebből persze még nem vonhatóak le kiváló pontosságú előrejelzések, ám jó közelítést adhat az előttünk álló időről.

Ugyan a szenzormodulom jelenleg beltérben helyezkedik el, de ahogyan már korábban is említettem a benti légköri nyomás megegyezik a kültéri értékekkel. Ezt kihasználva tudtam elkészíteni a beltéri „digitális meteorológusomat”. A megfelelő algoritmus megválasztásához és kidolgozásához ebben az esetben is (csakúgy, mint a hőmérséklet mérésénél) előzetes adatfelvétel eredményeire támaszkodtam. Megállapítottam, hogy ebben a mérési elrendezésben (földrajzi hely, szenzor mérési hibája) a normál nyomásérték körülbelül 101200 Pa. A későbbiekben ezt az értéket vettem figyelembe.

A döntéshozatal során fontos a légnyomás értéke és a változás (ha van) tendenciája és sebessége [17]. Előrejelző algoritmusom működési elve a következő: a rendszer az aktuális legfrissebb adathoz képest visszamenőleg 1 órát dolgoz fel, ez 12 mérési pontot jelent. A mérési pontokat egymáshoz hasonlítva először a növekedést vagy csökkenést, majd pedig a különbség abszolútértékéből a változás sebességét. Az aktuális trendet egy mérőszámmal adja meg (csökkenő trendhez negatív növekvő trendhez pozitív szám tartozik). Az így keletkező mérőszám és az aktuális nyomásérték segítségével hozza meg a megfelelő döntést a várható időjárásról. Közel állandó értékek esetén „Nem várható időjárás változás!”, csökkenő tendencia esetén „Eső várható!”, míg növekvő tendencia esetén „Szép idő várható!” üzeneteket küld a felhasználó számára (24. ábra). A rendszer jelenleg nem tökéletes, a nagyobb előrejelzési pontosság eléréséhez még finomhangolásra van szükség, de rövid távú tapasztalataim alapján elmondhatom, hogy így is egészen jó közelítést ad a várható időjárásról.



24. ábra

Előrejelzés légnyomás segítségével

Egy IoT szenzor itt említett felhasználási lehetőségein kívül (csak e három jellemzőt figyelembe véve is) a lehetőségek tárháza szinte végtelen. A páratartalom monitorozásával könnyedén készíthető olyan termosztát például, ami nem pusztán a hőmérsékletet veszi figyelembe a fűtés vezérlésénél, hanem a hőérzetet is. Emellett a légnyomás értékek felhasználásával támogatjuk családunk azon tagjait, akik szeretnek horgászni. Hiszen (kevésbé ismert) tény, hogy a halak táplálkozása és ezzel együtt a kapási hajlandósága összefügg a külső légnyomással, tehát egyszerűen megvalósítható lenne egy „IoT halfogó” is.

6. ÖSSZEGZÉS

A Smart City világa ma már nem science fiction. Néhány nagyvárosban okospadokon üldögélve tölthetjük mobil telefonjainkat. Már a háztartásban is megtalálhatóak olyan egyszerű okos kukák, amelyek automatikusan nyílnak megközelítéskor. Még rácsodálkozunk egy-egy ilyen eszközre, de be kell látnunk, hogy életünk egyre több területén jelennek majd meg olyan technikai megoldások, amelyek megkönnyítik a mindennapjainkat, szervezhetőbbé teszik az életünket.

Munkám során én is egy ilyen, a későbbiekben továbbfejlesztve a mindennapokban jól használható eszközt igyekeztem elkészíteni. Célom volt az is, hogy bemutassam, hogyan tud egy szenzor egy nagyobb rendszer részeként működni. Csak akkor nem öncélúak az ilyen típusú fejlesztések, ha a mindennapi döntéseinkben segítségünkre vannak. Azaz a folyamat minden elemének működnie kell: a mérőeszköz megalkotása, nagyszámú mérés lebonyolítása, az adatok feldolgozása és döntéshozatal. Ezt igazoltam egy légköri jellemzőket mérő szenzoregység megépítésével, több ezer adatpont lemérésével, ezek kiértékelésével és három különböző döntéshozó algoritmus megalkotásával.

Távlati célom eszközöm továbbfejlesztése, elsősorban Narrow Band IoT kommunikációs modullal történő kiegészítése. Ez a fejlesztés hozzájárulna későbbi terveimhez: egy nagyobb, akár több száz egységből álló szenzorhálózat kiépítéséhez. Ezzel létrehozhatjuk egy okos város egyik összetevőjét, ugyanakkor kutatásban is jól használható lenne, mivel a begyűjtött adatokból nemcsak valós idejű döntések hozhatóak, hanem hosszútávú, mélyebb elemzési célokat is szolgálhatnak. S nem utolsósorban segítségével a döntéshozó vagy tanuló algoritmusok is fejleszthetőek, tesztelhetőek lehetnek.

7. SUMMARY

In the present the urbanization when people into cities is a great tendency. Much more people live and work in big settlements as 50 years ago. Big cities have big problems too. The proper work of a town with 10.000 people is essential. In addition the citizens also have demands. For example everyone would like to live in a green city with clean and fresh air in it or there isn't any people who don't want to save money on the operation costs of a town.

For these and much more reasons have been created a system which called Smart City. In my thesis I would like to present the technical side of this beautiful and magical world. The heart of this system are the sensors which measure everything and every „vibration” of the settlement. The black magic what operates this scheme is the IoT (Internet of Things) this solution is the child of the fourth industrial revolution which lasts in our days too. To give a good presentation to the reader about this new technical innovation I created my own Smart City sensor. My device is a simple temperature, humidity and air pressure sensor which operated in the biggest room of my apartment. In my work I try to give an insight into the creation process of this sensor module and also would like to outline some usage of the collected data.

An IoT sensor operates 7/24 and collects many data on its own. Our task is to process these data and make a good decision which helps to maintain the proper operation of our Smart City. In my thesis I give some alternatives to the postprocessing of the collected data. I also created a display panel to the project so it can also visualise the result of the decision algorithm. As I mentioned before I measure three parameters of the air. From these data I made an algorithm with which we can supervise the proper work of our thermostat. From the humidity data I made a software which decides if we should humidify or dry the air of our house. From my final data which is the air pressure I made a small weather forecast software which watches the tendency of the air pressure and give some informations about the expected weather to the user.

KÖSZÖNETNYILVÁNÍTÁS

Ezúton szeretnék köszönetet mondani témavezetőmnek, Dr. Balázs Lászlónak a sok segítségért, amit kaptam tőle ahhoz, hogy létrejöhessen ez a dolgozat. Köszönöm tanulmányaim alatt adott tanácsait, kitüntető bizalmát, és hogy lehetővé tette, hogy egyetemi éveim alatt több kutatási projektben is részt vehessek.

Köszönöm dr. Kopják Józsefnek, hogy biztosította az MQTT szerveret és az adatfeldolgozáshoz szükséges háttérinfrastruktúrát.

S végül, de nem utolsósorban köszönöm Jacek Kondratowicznek, az ORING képviselőjének, hogy több hasznos előadással és konzultációs lehetőséggel hozzájárult ahhoz, hogy megismerhessem a modern IoT megoldásokat.

IRODALOMJEGYZÉK

- [1] d. fogalomtár, „A városi lakosság aránya,” [Online]. Available:
<http://demografia.hu/hu/letoltes/fogalomtar/pdf/varosi-lakossag-aranya.pdf>.
- [2] S. Zsolt, „SMART CITY, A JÖVŐ VÁROSA,” [Online]. Available:
https://www.urban.bme.hu/segedlet/varos1/eloadasok_2014/07B_SMART%20CITY_SZENDREI%20ZSOLT_kivonat.pdf.
- [3] dr. Dobos Klára, Kulcsár Sándor, Nagy Péter, dr. Sik András, Szemerey Samu, Vasváriné dr. Menyhárt Éva, „Smart city tudásplatform (Metodikai javaslat),” [Online]. Available: <https://lechnerkozpont.hu/doc/okos-varos/smart-city-tudasplatform-metodikai-javaslat.pdf>.
- [4] „Az IoT jelentése - Dolgok internete,” [Online]. Available: <https://eletokosan.hu/az-iot-jelentese-dolgok-internete>.
- [5] R. R. S. T. Somayya Madakam, „Internet of Things (IoT): A Literature Review,” [Online]. Available:
https://www.scirp.org/html/56616_56616.htm?pagespeed=noscript.
- [6] „BME280 adatlap,” [Online]. Available:
https://www.hestore.hu/prod_getfile.php?id=10549.
- [7] Espressif, „ESP32 Series Datasheet,” [Online]. Available:
https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf.
- [8] „IoT-technológiák és -protokollok,” Microsoft, [Online]. Available:
<https://azure.microsoft.com/hu-hu/overview/internet-of-things-iot/iot-technology-protocols/>.
- [9] „MQTT,” [Online]. Available: <https://en.wikipedia.org/wiki/MQTT>.

- [10] „Best IoT Analytics Software 2021,” [Online]. Available: <https://www.360quadrants.com/software/iot-analytics>.
- [11] „The Top 83 Flow Based Programming Open Source Projects on Github,” [Online]. Available: <https://awesomeopensource.com/projects/flow-based-programming>.
- [12] „Node-RED,” [Online]. Available: <https://nodered.org/about/>.
- [13] „Átlag-szórás kártyák,” [Online]. Available: http://www.statisztika-online.hu/pluginfile.php/3796/mod_resource/content/15/index.html.
- [14] „Control Chart Constants | Tables and Brief Explanation,” [Online]. Available: https://r-bar.net/control-chart-constants-tables-explanations/#XbarS_Sec.
- [15] „Mekkora lehet a megengedhető belső páratartalom ?,” [Online]. Available: http://szelloztetes.hu/index.php?m_id=2&id=102.
- [16] „Páratartalom kisokos,” [Online]. Available: <https://sensortech.hu/paratartalom-kisokos/>.
- [17] W. m. e. B. liest. [Online]. Available: <https://www.greelane.com/wissenschaft-technologie-mathematik/wissenschaft/how-to-read-a-barometer-3444043/>.
- [18] „Simple IOT,” [Online]. Available: <https://www.hackster.io/radhoo/simple-iot-14efa1>.

ÁBRAJEGYZÉK

1. ábra Magyarország népességének eloszlása [1].....	1
2. ábra Az IoT működése [18]	4
3. ábra Rendszerterv	7
4. ábra BME280 szenzor.....	9
5. ábra ESP32 mikrokontroller	9
6. ábra Az elkészült szenzormodul	10
7. ábra A kiküldött adatok keretformátuma	12
8. ábra Az MQTT működése	14
9. ábra Az MQTT topicok.....	15
10. ábra A megvalósított Node-Red háló.....	17
11. ábra Gauss görbe.....	20
12. ábra Mérési időközök.....	23
13. ábra Hőmérsékleti adatok	24
14. ábra Szórás és terjedelem viszonyok a mérés során	25
15. ábra Termosztát működése	27
16. ábra Páratartalom-hőmérséklet összefüggés	28
17. ábra Páratartalom és terjedelme az idő függvényében.....	29
18. ábra Légnyomás értékek	31
19. ábra 24 órás légnyomás érték.....	32
20. ábra Hosszútávú hőmérsékleti adatok.....	34
21. ábra Számításhoz szükséges konstansok [14].....	35
22. ábra Hőmérséklet-kontroll panel	36
23. ábra Páratartalom megjelenítése	37
24. ábra Előrejelzés légnyomás segítségével.....	39