

IoT-architektúra Intelligens Meteorológiai Állomásokhoz Firebase Segítségével Nagyobb Mennyiségű Adatkezeléshez

Tóth Áron, Tornai Tamás, Simon János

Tóth Áron, Kandó Kálmán Villamosmérnöki Kar, Óbudai Egyetem, Bécsi út 94-96., 1034 Budapest, Magyarország, tarontoth@vts.su.ac.rs

Tornai Tamás, Biztonságtudományi Doktori Iskola, Óbudai Egyetem, Népszínház utca 8, 1081 Budapest, Magyarország, tam.tornai@vts.su.ac.rs

Simon János, Mechatronikai és Automatizálási Intézet, Mérnöki Kar, Szegedi Tudományegyetem, Moszkvai krt. 9., 6720 Szeged, simon@mk.u-szeged.hu

Absztrakt: Napjainkban az Internet of Things (IoT) egyre nagyobb népszerűségnek örvend, és új lehetőségeket kínál okosotthonok számára a környezeti adatok gyűjtése, feldolgozása és megosztása terén. Ebben a munkában egy ESP32 mikrokontroller valós időben gyűjti az adatokat a különböző érzékelőktől, mérve az UV sugárzást, a hőmérsékletet, a páratartalmat, a légnyomást és a szélsébséget. Az ESP32 mikrokontroller képes vezeték nélküli kommunikációra, így képes Firebase segítségével történő adatkezelésre. A bemutatott architektúra a Firebase Realtime Database-el kombinálva nagyon hatékonyan és folyamatosan képes a szenzorok adatainak gyűjtésére, tárolására, frissítésére, szinkronizálására. Egy ilyen rendszer hasznos lehet az okosotthonok számára, a ház közelében történő időjárási viszonyok nyomon követésére, és nincs szüksége a helyi meteorológiai állomásokra, amelyek távolabb lehetnek az otthontól. A környezetből adatokat gyűjtő és tároló okosház csatlakoztatása az IoT-hez előnyösnek bizonyulhat a rendszert képező eszközök automatikus és kézi vezérlésében, mint például ebben az esetben egy árnyékoló. A szélsőséges időjárási viszonyok, valamint a szél és a napfény hirtelen változásai veszélyt jelenthetnek bizonyos eszközökre, amelyek képesek vagyunk megvédeni, ha olyan felhőszolgáltatáshoz csatlakoznak, ahonnan adatokat fogadnak az időjárási viszonyokról. A felhőhöz csatlakoztatott okoseszközök másik előnye a mezőgazdaságban rejlik, a helyi környezetből származó adatokra alapozva, ahol az ilyen rendszerek felhasználhatók a háztartás körüli növények öntözésének automatizálására, biztosítva az optimális vízfelhasználást és az egészségesebb növényzetet. Az érzékelők adatainak valós idejű megjelenítése és a kézi vezérlési lehetőségek javítják a felhasználói interakciót, míg az érzékelők adatainak tárolása és elemzése biztosítja a környezeti információk hosszú távú kezelését és elemzését. Eredményekként nyilvánvaló, hogy egy meteorológiai állomás integrálása egy okos házban egy adatbázissal nagyon hasznos lehet, és mind a jobb felhasználói élményhez, mind az ilyen eszközök automatikus vezérlésének továbbfejlesztéséhez vezethet.

Kulcsszavak: Adatbázis; Érzékelők; Firebase; Internet of Things (IoT); Környezeti adatok; Okoseszközök; Okosotthon; Valós idejű adatkezelés

1 Bevezető

Az elmúlt években egyre nagyobb mértékben terjednek el olyan okoseszközök, amelyek felhőalapú szolgáltatásokkal képesek kommunikálni. Számos előnnyel rendelkeznek az ilyen módon megvalósított eszközök, legfőképpen kommunikációs és adatgyűjtési szempontból, hiszen az IoT rendszerek lehetővé teszik a folyamatos információcserét [1].

Felhőn keresztüli kommunikáció számos új utat és lehetőséget megnyit előttünk, az adatok tárolásában és feldolgozásában, mivel többé nem feltétel az eszközök közötti fizikai kapcsolat létesítése. Továbbá megszűnnek a hardveres például ami a vezérlőnket illeti, nem fogynak el a kommunikációra szánt lábak, nem kell aggódnunk az adott kommunikációs protokoll maximális kapacitása elérésétől. Ebből adódóan a rendszerünkben jelen lévő eszközök száma könnyedén növelhető vagy csökkenthető tetszés szerint, valamint minden eszköz lekérheti a számára szükséges adatokat, nincs szükség Master-Slave struktúrára. Az adatok egyszerűen gyűjthetők egy adatbázisba és bármely eszköz kiértékelheti a számára jelentőséggel bíró paramétereket, és irányíthatja a rá bízott folyamatot. Az egyes eszközön mentett adatok számát így jelentősen le tudjuk csökkenteni, ezzel memóriakapacitást és energiát megtakarítva [2].

A felhő alapú kommunikáció lehetővé teszi számunkra, hogy a rendszerre olyan eszközöket is csatlakoztassunk, amelyek működésükhöz a szükséges energiát egy akkumulátorból nyerik. Ilyen eszközök általában két formában fordulnak elő egy okosotthonban: vagy hordozható eszközök vagy olyan eszközök, amelyek telepítésekor már nem volt lehetőség a megtáplálás odajuttatására. Hordozható eszközök nagyban tudják javítani a felhasználók interakcióját a rendszerrel, valamint kényelmesebbé és effektívebbé teszik a felhasználói élményt. Az akkumulátoros eszközök egyetlen hátránya, hogy töltésre vagy az elem cseréjére szorulnak bizonyos időközönként, szerencsére napjainkban az eszközökben használatos mikrovezérlők energiaigénye nem túl magas [3] [4].

1.1 Felhőalapú IoT-architektúrák és kommunikációs modellek áttekintése

Az IoT-re csatlakozó mikrovezérlők és szenzorok között lehetséges a valós idejű adatcsere egy felhő alapú adatbázison keresztül, a mai technológiai fejlettségnek köszönhetően. Li és társai [5] kidolgoztak egy JustIoT nevű rendszert, amelyben szemléltették, hogyan lehetséges a szenzoroktól gyűjtött adatokat eltárolni egy felhő alapú valós idejű adatbázisban, kifejezetten egy Firebase Real Time adatbázisban. A tanulmányból bebizonyosodik, a felhő alapú valós idejű architektúra és szenzoradatcsere rugalmassága és skálázhatósága.

Domíniguez-Bolaño [6] és munkatársai tanulmányából sokat megtudhatunk az IoT architektúrák mai állásáról, a technológia alapjairól, elképzelhető jövőjéről. Részletekbe merülő munka, amelyből megtudhatjuk, milyen módon kapcsolódnak az érzékelők és hogyan zajlik le az adatcsere valós időben az eszközeink között. Hangsúlyozzák, hogy a skálázhatóság és a platformfüggetlenség elengedhetetlen a megbízható adatáramláshoz.

A Cameroni Egyetem kutatócsoportja [7] felhívta a figyelmünket, egy kommunikációs problémára, amely abból ered, hogy az IoT rendszerekhez eszközöket gyártó cégek, mindegyike valamilyen egyedi protokollt használ eszközei közötti kommunikációra. A csapat ajánlott egy megoldást a problémára, mégpedig olyat, amellyel nem kellene módosítani a már beépített eszközök fizikai jellemzőit. A felajánlott megoldás egy felhőben megtalálható adatbázison keresztül megvalósított kommunikáció az eszközök között, amely növeli az IoT rendszerek skálázhatóságát.

2 Felhőalapú meteorológiai rendszer tervezése

Az előzőekben ismertetett tulajdonságokat figyelembe véve érdemes felállítani egy tervet és elvárásokat egy felhőalapú kommunikációval rendelkező meteorológiai állomással szemben. Egy ilyen rendszer képes kell legyen kommunikációra, adatgyűjtésre és kiértékelésre, skálázható kell legyen és biztosítani kell módot az adatok megjelenítésére is. Tervezéskor figyelembe kell venni mely adatokat érdemes kinyerni a környezetből, valamint meghatározni milyen érzékelőkkel, vezérlővel és más hardverrel érjük el a kitűzött célokat.

2.1 A rendszerrel szemben támasztott elvárások

A rendszerrel szemben támasztott elvárásokat össze kell állítani funkcionalitás, kezelhetőség és megbízhatóság szempontjából. Legfontosabb szempontok, a valós idejű adatgyűjtés, valós idejű felhőalapú kommunikáció és az adott eszközben történő adatkiértékelés, és lehetőleg alacsony energiaigény. A szempontokat figyelembe véve kiderül, hogy a legfontosabb kiválasztani a megfelelő mikrovezérlőt, valamint felhő szolgáltatást.

A mikrovezérlő képes kell legyen vezeték nélküli internetkapcsolatra, amelyre automatikusan újra csatlakozik minden bekapcsolás után, emberi beavatkozás nélkül. Továbbá képesnek kell lennie ezen kapcsolaton keresztül kommunikálni egy adatbázissal, különböző kommunikációs típusok megvalósítására a perifériális lábakon, a különböző szenzoradatok begyűjtéséhez, elegendő lábbal a különböző jelenlegi perifériák számára és jövőbeli skálázhatósághoz.

A felhőalapú szolgáltatáshoz is fontos kitűzni néhány elvárást, éspedig képes kell legyen valós idejű kétirányú adatszinkronizációra, hogy a csatlakoztatott kliensek között az adatáramlás zökkenőmentesen történjen és a kommunikációban ilyen módon az eszközök egymásnak parancsokat is tudjanak küldeni, ne csak adatokat lekérni. Az adatok strukturált tárolása is elengedhetetlen, hogy elkülönüljenek egymástól a mérési adatok és a vezérlési parancsok. Az adathozzáférés is lekérdezés is gördülékenyen kell történjen, úgy ahogy a kliensek becsatlakozása is miután újból feszültséghez jutottak. Magas újraírhatósági sebessége, és alacsony késleltetéssel kell rendelkezzen az adatbázis, valamint skálázhatónak kell lennie a későbbi bővítések céljából. Elvárás, hogy tartalmazzon hitelesítési és engedélyezési mechanizmusokat is, hogy minden eszköz csak a számára engedélyezett helyre tudjon adatokat bevinni és olvasni.

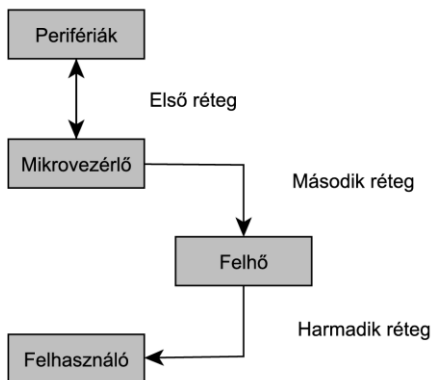
A szenzoroktól elvárható, hogy magas mérési pontossággal és nagy felbontással rendelkezzenek. A mért értékek nem térhetnek el a valós fizikai értékektől, fontos a hiteles adatokkal való munka [8]. Elvárás továbbá az érzékelőktől, hogy integrálhatóak a rendszerben található vezérlőkkel, tehát olyan vagy valamilyen kommunikációs protokollt, vagy jelszint alapján lehetőséget kell biztosítsanak az adatok kinyerésére. Az érzékelők alapvetően a kinti környezetről szolgáltatnak adatokat, ezért fontos, hogy alapvetően vízhatlanok és napállóak legyenek vagy ha ez nem adott, akkor az kint használatos eszközöket ilyen módon kell tervezni. Ha a rendszerünk tartalmaz hordozható mérő egységet, akkor ezen elvárások bővülnek egy szemponttal, amely az alacsony energiaigény.

A rendszernek nemcsak a hozzá tartozó eszközökkel, és a felhővel kell kommunikációt létesítsen, hanem a felhasználóval is. Különböző módok léteznek a rendszer számára, amelyekkel tud kommunikálni a felhasználóval, legfontosabb kikötés, hogy a felhasználónak rálátása legyen a mért adatokra és beleszólása a vezérlés módjába vagy a vezérlés felülírásába, kedve szerint. Lehetséges módok közé tartoznak a különböző kijelző egységek használata, amely lehet vagy hordozható, további lehetőség egy mobilapplikáció, amellyel bárholonnan hozzáférhetünk rendszerhez, vagy egy weboldal, amelynek elérése a belső hálózaton keresztül történik [9].

2.2 Lehetséges kommunikációs- és rendszerarchitektúra

Egy IoT rendszer kommunikációs architektúrája alapvetően 3 rétegből áll. Az első kommunikációs rétegbe tartozik a mikrokontroller és a hozzá tartozó perifériák közötti kommunikáció, fontos a gyors és folyamatos adat frissítés, amely általában vezetékkel valósul meg. A második réteg a mikrokontroller felhővel való kommunikációját jelenti, amely kétirányú, tehát a mikrokontroller nem csak küldd adatokat a valós idejű adatbázis felé, hanem fogad is. A harmadik kommunikációs réteg a felhasználói kommunikáció, amely engedélyezi a felhasználót az adatok

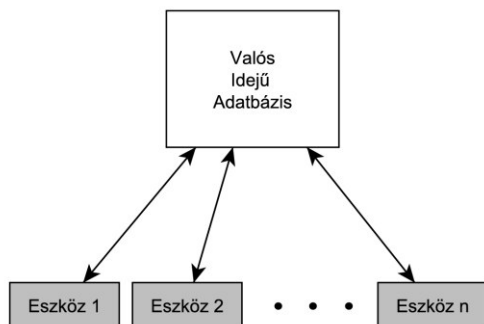
megfigyelésére és vezérlésre (1. ábra). Megvalósítható folyamatok távoli működtetése is az adott kommunikációs struktúrából kifolyólag [11].



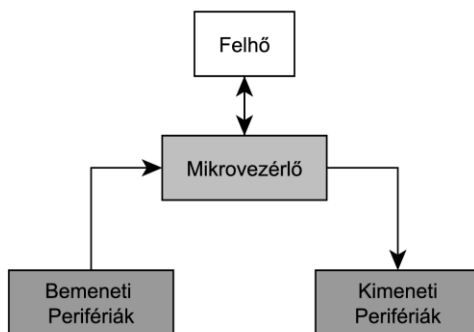
1. ábra
Kommunikációs rétegek

2.2.2 Lehetséges rendszerarchitektúra

Rendszerarchitektúra szempontjából fontos figyelembe venni, hogy a kommunikáció a felhőn keresztül zajlik az egyes eszközök között (2. ábra). Egy lokális környezeti adatokat gyűjtő rendszer architektúráját képezhetik mérőeszközök, amelyek adatokat gyűjtenek a környezetből, vagy felhasználói eszközök/felületek, amelyek az adatok felhasználó felé való eljuttatásáért felelnek, valamint képesek néhány utasítás fogadására és továbbítására a felhő alapú kommunikáción keresztül. Vezérlő egység is külön megjelenhet a rendszerbe, az adott folyamat vezérléséhez vagy irányításához, de elképzelhető, hogy a felsorolt eszközfajták kombinálva közös tokba kerülnek. A rendszerarchitektúra minden esetben olyan módon fog megvalósulni, hogy maga a valós idejű adatbázis kommunikál a mikrovezérlőkkel a felhőn keresztül, a mikrovezérlők pedig a rájuk kapcsolt perifériákkal, az eszközarchitektúrát a 3. ábra szemlélteti.



2. ábra
Rendszerarchitektúra



3. ábra
Eszközarchitektúra

3 Meteorológiai állomás megvalósítása

Figyelembe véve a 2. fejezetben ismertetett elvárásokat és lehetőségeket, elkészíthető a felhőalapú kommunikációval rendelkező lokális meteorológiai állomás. Elsősorban a vezérlőt érdemes jól megválasztani, majd ügyelni az érzékelők megválasztására, és adatbázisra. A rendszer és a felhasználó interakciójára sok lehetőség megvalósítható, ezen munkában egy egyszerűbb megoldásra esett a választás, csupán abból kifolyólag, hogy szemléltesse a felhőalapú kommunikációt, mikrovezérlők között, nem pedig egy mobileszközzel vagy egy számítógéppel.

3.1 ESP32 Mikrovezérlő

Az ESP32 mikrovezérlő egy nagy teljesítményű, vezeték nélküli kommunikációra alkalmas mikrovezérlőlap, amelyet az Espressif Systems fejlesztett ki (). Optimális megoldás IoT projektek megvalósítására, nemcsak a gyors és megbízható vezeték nélküli kapcsolat miatt, hanem rendelkezik számos kommunikációs móddal a lábain, amely lehetővé teszi az adatkinyerést a legtöbb érzékelőből. A mikrovezérlő 3.3V feszültség szinten üzemel, programozására az Arduino IDE környezetben került sor.

A munkában az ESP32-Wroom típus volt használva, amely architektúrája két Xtensa LX6 32 bites mikroprocesszort tartalmaz, amely lehetővé teszi, hogy egyik mag a kommunikációs feladatokra lássa el, a másikon pedig a lábakra csatlakoztatott perifériák vezérlése történjen. Vezeték nélküli kommunikációk terén rendelkezésre áll Wi-Fi (IEEE 802.11 b/g/n) és Bluetooth (BLE⁴). Perifériák széles skálája csatlakoztatható a be- és kimeneti lábakra (GPIO⁵ portokra), amelyek többféle funkciót láthatnak el, egyes lábak képesek digitális jelet analóggá alakítani (DAC⁶), vagy fordítva (ADC⁷), mások rendelkeznek digitális kommunikációs protollokkal akár. A lábkiosztást az 4. ábra szemlélteti, az egyes lábak funkciója a **Hiba! A hivatkozási forrás nem található.** kerül ismertetésre, azon funkciókkal feltüntetve, amelyek ezen munka szempontjából fontosak lehetnek. A vezérlő rendelkezik 25 GPIO porttal, ezek közül 2 lett előre látva I2C kommunikációra, 9 port 3 különböző SPI⁸ kommunikációs módra, 4 port 2 különböző UART⁹ kommunikáció számára. 21 port rendelkezik PWM modulációval, akár ADC, DAC vagy mindkettővel, 15 GPIO pin úgynevezett RTC¹⁰ portok, amelyek képesek felébreszteni a vezérlőt alvó állapotából, ha olyan környezetben használjuk.

⁴ BLE – Bluetooth Low Energy

⁵ GPIO – General Purpose Input Output

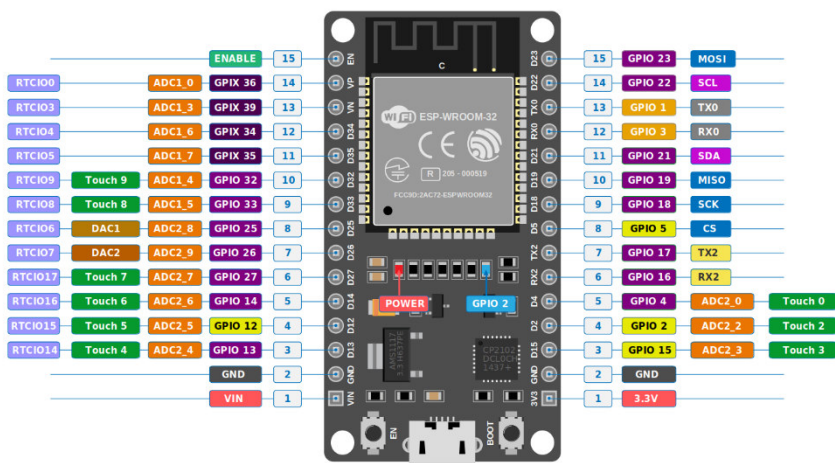
⁶ DAC – Digital Analog Converter

⁷ ADC – Analog Digital Converter

⁸ SPI – Serial Peripheral Interface

⁹ UART – Universal Asynchronous Receiver/Transmitter

¹⁰ RTC – Real Time Clock



4. ábra

ESP32 Wroom lábkiosztása

3.2 Érzékelők

3.2.1 BME280

A BME280 egy a Bosch Sensortec által fejlesztett kombinált környezeti szenzor, amely képes légnyomás, relatív páratartalom és hőmérséklet mérésére. Modulra szerelt változata a ábrán látható, amely változat ebben a munkában használatra került.

Légnyomás mérésekor fény derül a tengerszint feletti magasságra, hektopascalban (hPa) kifejezve. A meteorológiai állomások ebből tudnak következtetni, a rövid távú időjárás előrejelzésre, alacsony légnyomás a rossz időjárás, viharok hírnöke, a magas légnyomás pedig a derűs, stabil időjárásé, a nyomásváltozás is figyelhető és elemezhető. Mérési tartomány 300 hPa- tól 1100 hPa-ig terjed, ebben a tartományban a pontosság ± 2.25 hPa, a szenzor felbontása 0.0018 hPa. A hőmérsékletet fok-Celsiusban ($^{\circ}\text{C}$) fejezi ki, mérési tartománya -40°C -tól $+85^{\circ}\text{C}$ -ig terjed, pontossága 0°C - tól 65°C -ig $\pm 0.5^{\circ}\text{C}$, felbontása 0.01°C . Relatív páratartalom mérési eredményét százalékosan (%) fejezi ki, nagy linearitással és Histerézis-menteséggel ($\pm 1\%$), $\pm 3\%$ pontossággal és 0.008%-os felbontással [10]. A relatív páratartalomból és a hőmérsékletből kiszámolható a harmatpont, amely a levegő valódi nedvességtartalmát eredményezi, valamint a hőérzet. A hőérzet számítására leggyakrabban a Magnus-Teten egyenlet használatos:

$$T_{DP} = \frac{b \cdot \gamma(T, RH)}{a - \gamma(T, RH)} \quad (1)$$

Ahol:

$$\gamma(T, RH) = \frac{a \cdot T}{b + T} + \ln(RH/100)$$

T: Hőmérséklet Celsius-fokban

RH: Relatív páratartalom százalékban

a és b: Empirikus állandók, általában $a \approx 17.27$, $b \approx 237.7$

Kommunikációs protokollokkal lehetséges a mikrovezérlő és az érzékelő közötti adatsere, név szerint SPI és I2C, jelen munkában az I2C került alkalmazásra egyszerűsége és kisebb lábigénye miatt, a 0x76 címen érhető el. A szenzor alapvetően kalibrálva érkezik, a belső memóriájában el vannak tárolva a mikrovezérlő számára a mérések pontos lebonyolításához szükséges kompenzációs adatok. A szenzor továbbá támogat alacsony energiafogyasztású üzemmódokat, alvó módba helyezhető.

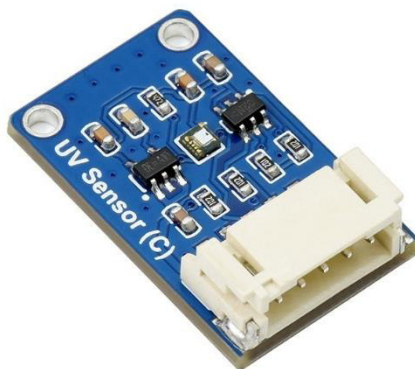


5. ábra
BME280 érzékelő

3.2.2 LTR-390

Az LTR-390 egy digitális érzékelő, amely a látható fény és az ultraibolya sugárzás mérését képes megvalósítani. Két speciális fotodióda segítségével méri az UV-A (315-400 nm) és UV-B (280-315 nm) hullámhosszokat, amelyekből számítandó az UV Index. Látható fény érzékelése a 0.01-64000 Lux tartományban történik, az adatokat maximum 20 bites felbontással kódol. Akárcsak a BME280 az LTR-390 is egy modulra felszerelt szenzor formájában került alkalmazásra ezen munkában,

amelyet a 6. ábra szemléltet. A mikrovezérlő az érzékelővel, jelen modul esetében I2C protokollal létesíthet kapcsolatot, a 0x53 címen.



6. ábra
LTR390 érzékelő

3.2.3 Anemométer

Jelen munkában a szélesebbség mérésére egy háromkanalas mechanikus rotációs anemométer (7. ábra) került alkalmazásra, ezen struktúra a szél kinetikus energiáját a tengely forgásába fordítja át. A forgást egy a tengely végére csatlakoztatott kis generátor alakítja analóg jellé, 0-5V tartományban, ezen analóg jelet a mikrovezérlő egyik ADC képes lábára kapcsolva az így kinyert adatból számítható a szél sebessége. Az ESP32 mivel nem 5V hanem 3.3V feszültségen üzemel, ezért feszültségosztó ellenállásokat szükséges beiktatni és a programkódban megfelelően korrigálni a beolvasott értéket. Mérési tartománya 0-30m/s, 0.01m/s pontosság mellett. Fontos megemlíteni, hogy ezen anemométer számára szükséges külső feszültségforrás, mivel 10-30V egyenfeszültséggel üzemel, így ez az érzékelő, ilyen kivitelezésben nem felhasználható akkumulátoros eszközökben.



7. ábra

Háromkanalas anemométer

3.3 Kijelző egység

A kijelző egység magába foglal egy kijelzőt, két nyomógombot, és egy ESP32 Wroom mikrovezérlőt. A kijelzőn megjelennek a mért adatok, a mellette található gomb pedig a vezérlésre szolgál. Az adatokat, amelyek megjelennek a kijelzőn a mikrovezérlő a felhőből olvassa le, és tölti fel a gombokra beprogramozott vezérlő utasításokat a Firebase adatbázisba. Kijelzőként egy SSD1306 0.96" kék háttérvilágítású OLED 128x64 felbontású kijelző került kiválasztásra, amely I2C kommunikációval létesít kapcsolatot a mikrovezérlővel, és egy 3D nyomtatással PLA-CF¹¹ anyagból készített dobozba lett becsomagolva, 8. ábra szemlélteti.



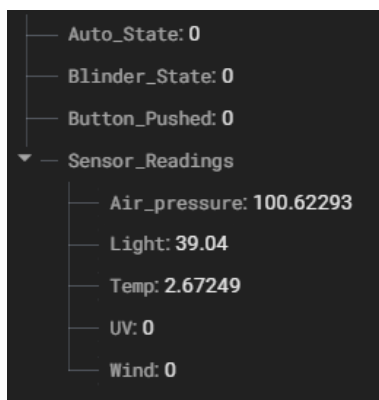
8. ábra

Kijelző egység

¹¹ PLA-CF – PolyLactic Acid – Carbon Fiber (Politejsav szénzál-erősítéssel)

3.4 Firebase Realtime Database

A Firebase Realtime Database egyike a Google Firebase platform szolgáltatásainak, felhő alapú, NoSQL¹² adatbázis, amely valós idejű adatszinkronizációra képes. A Firebase Realtime Database ellentétben a hagyományos modellekkel nem a lekérdezésekre válaszol az eszközöknek, hanem folyamatos, perzisztens kapcsolatot tart fenn a feliratkozott kliensekkel, és ha bármely adat módosul, az leküldésre kerül minden csatlakoztatott eszköz felé. A NoSQL struktúrájának köszönhetően sokkal rugalmasabban oszlanak el az adatok, nem sorokból és oszlopokból áll össze, mint egy relációs adatbázis, hanem az összes adat egy hatalmas JSON¹³ fa struktúrában tárolódik. A fa struktúrában úgynevezett kulcs-érték párokként kerülnek tárolásra az adatok, a kulcsokat az elérési útvonalon keresztül lehetséges megközelíteni. A Firebase Realtime Database fa struktúrája egy fajta séma-nélküliséget eredményez, amelynek köszönhetően, rugalmasan, módosítható az adatbázis, minden ága tetszés szerint, a kulcs-érték pároknak köszönhetően, 9. ábraán megfigyelhető a struktúra. Az adatbázis bővíthető biztonsági szabályokkal (Security Rules), amelyek segítségével csak azon kliensek férhetnek hozzá az adatbázishoz, akik követik a megadott JSON szabályrendszert. Ilyen módon akár meg is határozható, hogy mely eszközök írhatnak felül bizonyos adatokat, és melyek csak olvashatják őket. Az Arduino IDE környezetbe betölthető a Firebase dedikált könyvtára, amely segítségével az ESP32 mikrokontroller eléri az adatbázist, és le tudja futtatni a szükséges biztonsági bejelentkezéseket, és létrehozzák a folyamatos adatszinkronizálást.



9. ábra

Firebase Realtime Database-ben tárolt adatok

¹² NoSQL – Rugalmas séma, nem táblázatokhoz kötött adattárolás

¹³ JSON – JavaScript Object Notation

4 A rendszer skálázhatósága és előnyei

4.1 Skálázhatóság

A megvalósított rendszerről elmondható, hogy hardveres és szoftveres szempontból is rugalmasan fejleszthető és bővíthető. A felhőalapú kommunikáció rugalmasan skálázható, magas interoperabilitást engedélyez az eszközök között. Hardveres szempontból a skálázhatóság két módon képzelhető el, az egyik, hogy újabb eszközök kapcsolódnak a rendszerbe, amelyik az egyszerűbb eset a rugalmas szoftveres skálázás miatt. A másik eset, ha valamelyik eszköz bővítése a cél, például egy új érzékelő vagy vezérlési folyamat hozzáadásával, ebben az esetben felléphetnek hardveres korlátok. Megoldás lehet ilyen típusú problémára, hogy az eszközökben található nyomtatott lapokon több becsatlakozási pont kerül felforrasztásra, mint ahány kihasználásra kerül. Az ilyen hozzáadott pontok csatlakozhatnak, a különböző kommunikációs protokollokkal ellátott buszokhoz (I2C, SPI), vagy akár bármely GPIO porthoz, egy referenciaponttal párban egy csatlakozóval [12].

4.2 Lokális környezeti adatok gyűjtésének előnyei

A lokálisan gyűjtött adatok rendelkeznek egy nagy előnnyel a távolabb található meteorológiai állomásoktól, mégpedig a mért adatok közvetlenül a helyszínről származnak, így pontosabb adatokat szolgáltatva az adott terület időjárási viszonyairól. Az ilyen módon gyűjtött adatok különösen fontosak lehetnek amikor a mikroklíma eltér az állomás adataitól, valamint alkalmazható különböző körülmények melletti mérésre, amelyek esetében nem is releváns az állomások által szolgáltatott adatok. Lehet mérni akár árnyékban, akár valamilyen belső környezetben a viszonyokat és az így kapott adatok alapján vezérelni eszközöket. Egy IoT rendszerbe akár több mérő egység is bekapcsolható, és a Firebase Realtime Database fa struktúráját és biztonsági funkcióit kihasználva az eszközök között optimálisan eloszthatóak az adatok. Minden eszköz a számára megszabott helyre tölt fel az adatokat, így akár felosztható az egész okosotthon még szobánként is ha szükséges.

4.2.1 Lokálisan gyűjtött adatok és meteorológiai állomástól kapott adatok összehasonlítása

Az 1. táblázat szemlélteti a hivatalos meteorológiai állomás által mért időjárási adatokat, 10 napon keresztül, az időpontok megközelítőleg kerültek kiválasztásra a lokális mérés megvalósulásainak időpontjaihoz. A mért adatok a hivatalos szerbiai meteorológiai szövetség honlapjáról kerültek a táblázatba [13], a Szabadkához legközelebbi állomás Palicson található.

1. táblázat
Hivatalos meteorológiai állomás által mért adatok

Dátum és idő	Hőmérséklet	Páratartalom	Légnyomás	UV Index	Szélsébség
2025.11.06 13:00	14°C	58%	1021 hPa	2	4 km/h
2025.11.08 10:00	9°C	97%	1016 hPa	1	4 km/h
2025.11.11 13:00	7°C	97%	1020 hPa	1	4 km/h
2025.11.12 16:00	7°C	99%	1024 hPa	1	2 km/h
2025.11.13 16:00	8°C	95%	1024 hPa	1	4 km/h
2025.11.14 10:00	4°C	97%	1023 hPa	1	2 km/h
2025.11.15 10:00	11°C	88%	1017 hPa	1	4 km/h
2025.11.16 16:00	16°C	64%	1012 hPa	1	6 km/h
2025.11.17 13:00	19 °C	51%	1002 hPa	1	11 km/h
2025.11.18 13:00	8°C	60%	1021hPa	1	4 km/h

A 2. táblázat szemlélteti a munkában ismertetett rendszer által mért lokális adatokat, a mérések kint a szabadban lettek elvégezve. Megfigyelhetők kis eltérések az adatokban, az adott időpontokra, amelyeket betudhatunk a hivatalos állomástól való távolságnak és folyamatosan kissé változó időjárásnak. Viszont megfigyelhető két kiugró adat, egyik 2025. 11. 06.-án az UV Index, másik 2025. 11. 08.-án a szélsébség. Fontos figyelembe venni az említett mérésekhez hasonló mérési adatokat, mivel nem szerepelnek a hivatalos állomás által végzett mérésekben, pontosan az ilyen adatok miatt érdemes végezni a lokális adatgyűjtést. Ezen munkában megvalósított rendszer mindkét adatra reagált volna, első esetben nyitotta volna, másodikban zárta volna az árnyékolót. Környezeti adatok mérésekor nemcsak a kiugró adatok bizonyulhatnak fontosnak, hanem olyan adatok is amelyek eltérnek az állomás által közvetített adatoktól, mivel a

mérőegység nem feltétlenül kell a szabadnak helyezkedjen el (például a páratartalom üvegházak esetén).

2. táblázat
Lokálisan mért időjárási adatok

Dátum és idő	Hőmérséklet	Páratartalom	Légnyomás	UV Index	Szélesebesség
2025.11.06 13:11	13.6°C	61%	1019 hPa	3	5.2 km/h
2025.11.08 9:58	8.5°C	96%	1016 hPa	1	29.1 km/h
2025.11.11 13:23	6.7°C	97%	1022 hPa	1	6.7 km/h
2025.11.12 15:49	6.9°C	98%	1020 hPa	1	4.5 km/h
2025.11.13 16:01	8.2°C	92%	1021 hPa	1	5.0 km/h
2025.11.14 10:13	5.1°C	96%	1023 hPa	1	3.3 km/h
2025.11.15 10:12	10.7°C	83%	1015 hPa	1	5.9 km/h
2025.11.16 15:50	15.1°C	62%	1010 hPa	1	6.4 km/h
2025.11.17 13:17	19.5°C	47%	998 hPa	1	12.1 km/h
2025.11.18 13:05	7.5°C	55%	1017 hPa	1	5.2 km/h

4.2.2 Lokálisan gyűjtött adatok felhasználhatósága

A lokálisan gyűjtött adatok nagy előnye, hogy a mérési adatok konkrétan a mérés helyszínének adottságait tükrözik, sokkal pontosabban jellemzik a környezeti viszonyokat. Az ilyen rendszerek lehetővé teszik a helyspecifikus döntéshozatalt, különböző folyamatokhoz való időjárási adatok gyűjtésére, figyelhető a mikroklíma, vagy figyelhetőek csak egyes időjárási adatok, amelyek fontosak az adott folyamat irányítása szempontjából. Egy rendszerbe akár csatlakoztatható

több mérő egység, amelyek pontosan azon adatokat figyelik és töltik fel az adatbázisba, amelyek az adott folyamat szempontjából fontosnak bizonyulnak. Ilyen módszert alkalmazva még kisebb területekre felbontható a lokális meteorológiai adatgyűjtés, tehát még lokálisabbá tehető, minél lokálisabban végezhető az adatgyűjtés annál pontosabban képes a rendszer a folyamatok irányítására. Különösen fontos a lokális adatok gyűjtése olyan környezetekben, ahol az adatok gyors változása és a pontatlan adatokból eredő döntéshozatal kárt okozhat az eszközökben.

A lokális meteorológia adatok gyűjtése felhasználható például árnyékoló vezérlésére, öntözőrendszerek működtetésére, szellőztetés irányítására, fűtés ütemezésére, precíziós növénytermesztésre stb.

Összefoglalás

Összefoglalásként elmondható, hogy a megvalósított IoT-architektúra hatékony és megbízható valós idejű lokális adatok gyűjtésére, továbbítására és kiértékelésére. A rendszer fő eleme az ESP32 mikrovezérlő megfelelő vezeték nélküli kapcsolat létesítésére képes a Firebase Realtime Database-el valós idejű adatkommunikáció szempontjából. A különböző szenzoradatok gyorsan, és strukturáltan tárolódnak, amelyek később könnyen és késedelem nélkül elérésre kerülnek, és felhasználhatóak folyamatok irányítására.

A lokálisan mért adatok összevetése a hivatalos meteorológiai állomás által mért értékekkel rávilágított, a helyszínspecifikus mérések fontosságára és előnyeire felhasználhatóság szempontjából.

Az ismertetett felhő alapú időjárási viszonyok mérésére szolgáló architektúra rugalmasnak, bővíthetőnek és jól felhasználhatónak bizonyult, alkalmas valós idejű adatgyűjtésre, adatkommunikációra és elemzésre. A munka eredményeképpen levonható, hogy az IoT-ra csatlakoztatott környezeti adatgyűjtő állomások számos előnnyel rendelkeznek, egyes folyamatok elengedhetetlen részét kell képezzék, képesek megkönnyíteni és biztonságosabbá tenni a mindennapi életet a felhasználó számára.

Források

- [1] Simon, J.: Possible Implementations of The Cloud-Based Data-Centric IoT Pilot Platforms For Smart Cities And Agriculture. *Analecta Technica Szegedinensia*, Szegedi Tudományegyetem, Szeged, 2018, pp. 39-44.
- [2] Padmaja, B., Kumar, P., Sai Teja, S.: Google Firebase Based IoT System Architecture. *IJERT – International Journal of Engineerig Research and Technology*, Hyderabad, 2021, pp. 1-4.
- [3] Bhavya, A.R., Sudharshan, K.M.: Optimizing Energy Efficiency in Battery-Powered IoT Devices through Hardware Optimization and Voltage Scaling.

- Engineering, Technology and Applied Science Research (ETASR), Bengaluru, 2025, pp. 21769-21773.
- [4] Yüksel ME. Power Consumption Analysis of a Wi-Fi-based IoT Device. *Electrica*, Burdur, Turkey, 2020, pp. 62-70
- [5] Li, Wu-J., Yen, C., Lin, You-S., Tung, Shu-C., Huang, Shih-M.: JustIoT Internet of Things based on the Firebase Real-time Database. *Proceedings of the 2018 International Symposium on Semiconductor Manufacturing Intelligence (ISMI2018)*, Hsinchu, Taiwan, 2018, pp. 1-5.
- [6] Domínguez-Bolaño, T., Campos, O., Barral, V., Escudero, C. J., García-Naya, J. A.: *An overview of IoT architectures, technologies, and existing open-source projects*. Elsevier, Coruña, Spain, 2024, pp 1-15.
- [7] Petracci, R., Culmone, R.: *A Scalable Approach to IoT Interoperability: The Share Pattern*. MDPI Sensors, Camrino, Italy, 2025, pp 1-22.
- [8] Feng, Y., Hu, J., Duan, R., Chen, Z.: *Credibility Assessment Method of Sensor Data Based on Multi-Source Heterogeneous Information Fusion*. MDPI Sensors, Changdu, China, 2021, pp 1-21.
- [9] Reiss, P. S.: *Building an Interface for Controlling IoT Devices*. Brown CS Research Technical report, Providence, USA, 2019, pp 1-4.
- [10] Kusuma, H.A., Yuliani, Suhendra, T., Devendra, D., Setyono, D. E. D.: *Evaluating the Accuracy of BMP280 and BME280 Sensors for Sea Level in a Coastal Environment: A Field Study at Tanjung Siambang Pier*. *Indonesian Journal of Marine Science*, Tanjungpinang, Indonesia, 2023, pp 189-202.
- [11] Sung, W.-T., Devi, I.V., Hsiao, S.-J.: *Smart Lamp Using Google Firebase as Realtime Database*. *Intelligent Automation & Soft Computing*, Taichung, Taiwan, 2022, pp 967-982.
- [12] Chen, T., Barbarossa, S., Wang, X., Giannaks, G. B., Zhang, Z.-L.: *Learning and Management for Internet-of-Things: Accounting for Adaptivity and Scalability*. *arXiv preprint*, Cornell University, Ithaca, USA, 2018, pp. 1-15.
- [13] Republički hidrometeorološki zavod Srbije. (n.d.). *Aktuelni Podaci*. Letöltve: 2025. 11. 18.
<https://www.hidmet.gov.rs/latin/osmotreni/index.php>