**EDZÉSTÁMOGATÓ RENDSZER KÜZDŐSPORTOKHOZ**

**OE-NIK** Hallgató neve: **Kolozsi Norbert**

**2021** Hallgató törzskönyvi száma: **T/008850/FI12904/N**

Óbudai Egyetem

Neumann János Informatikai Kar

Biomatika és Alkalmazott Mesterséges Intelligencia Intézet

DIPLOMAMUNKA

FELADATLAP

Hallgató neve: **Kolozsi Norbert**

Törzskönyvi száma: T/008850/FI12904/N

Neptun kódja: U95726

A diplomamunka címe:

# Edzéstámogató Rendszer Küzdősportokhoz

### Martial Arts Training Supporting System

Intézményi konzulens: Prof. Dr. Kozlovszky Miklós

Külső konzulens:

Beadási határidő: 2021. december 15.

**A feladat**

A versenysportok világában egyre elterjedtebb a különböző technológiák beépítése az edzéstervekbe, a sportolókról óriási mennyiségű információ érhető el, amikből jó képet kaphatnak jelenlegi fizikai, illetve mentális állapotukról. A diplomamunka célja egy olyan mikroprocesszorral és szenzorokkal ellátott eszköz létrehozása, ami segítséget nyújt a harcművészeknek a kombinációk begyakorlásában, valamint a reflexek javításában. A feladat, egy olyan rendszer létrehozása, ami egy, az emberi testen fellelhető kiütési felületeken szenzorokkal, valamint jelzőfénnyel rendelkezik. A hallgató hozzon létre egy eszközt, ami fény effekttel képes jelezni a találatokat majd az ezekből származó adatokat egy okostelefonra továbbítani. Az okostelefonon létrehozott alkalmazás lehetőséget biztosít a kinyert adatok elemzésére, valamint az edző számára kombinációk létrehozására. A feladatnak része a hasonló rendszerek megismerése, a rendszer megtervezése, majd megalkotása. Végezetül pedig a rendszer tesztelése éles környezetben, majd az eredmények elemzése.

**A diplomamunkának tartalmaznia kell**:

* a megvalósítandó feladat részletes bemutatását,
* a megvalósítandó feladatban használt elemek ismertetését
* a megvalósítandó feladattal szemben támasztott követelményeket,
* az alkalmazás megvalósításához elkészített rendszertervet,
* az alkalmazás elkészítésének folyamatát, a munka során felmerült problémákat,
* a kialakított rendszer tesztelési tervét,
* az elkészített rendszer tesztkörnyezetének leírását, tesztelését, valamint a tesztek eredményeit,
* a teszteredmények értékelését, továbbfejlesztési javaslatokat.

Ph.

……....……………….

Dr. Eigner György

intézetigazgató

A diplomamunka elévülésének határideje: **2024. december 15.**

(OE TVSz 55.§ szerint)

A diplomamunkát beadásra alkalmasnak tartom:

|  |  |
| --- | --- |
| ……………….. | ..…………………. |
| külső konzulens | intézményi konzulens |

HALLGATÓI NYILATKOZAT

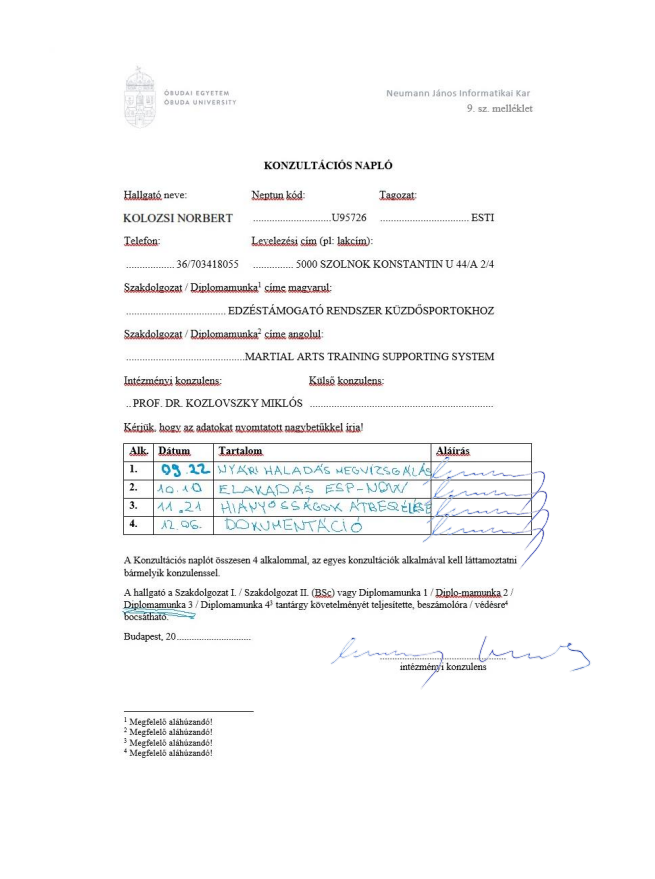
Alulírott hallgató kijelentem, hogy a szakdolgozat / diplomamunka saját munkám eredménye, a felhasznált szakirodalmat és eszközöket azonosíthatóan közöltem. Az elkészült szakdolgozatomban / diplomamunkámban található eredményeket az egyetem és a feladatot kiíró intézmény saját céljára térítés nélkül felhasználhatja.

Budapest, 20

hallgató aláírása

**A képen asztal látható

Automatikusan generált leírás**



Tartalom

[**Absztrakt** 8](#_Toc122120021)

[**1.** **Bevezetés** 9](#_Toc122120022)

[**2.** **Követelményspecifikáció** 10](#_Toc122120023)

[2.1 Szenzor modul 10](#_Toc122120024)

[2.2 Vezérlő modul 11](#_Toc122120025)

[2.3 Felhasználói felület 11](#_Toc122120026)

[**3.** **Alternatív megoldások** 13](#_Toc122120027)

[3.1 „Simon” 13](#_Toc122120028)

[3.2 Redflash 13](#_Toc122120029)

[3.3 Miért másabb a MATTS? 15](#_Toc122120030)

[**4.** **Kábel nélküli kapcsolat lehetőségei** 16](#_Toc122120031)

[4.1 ZigBee 16](#_Toc122120032)

[4.2 Bluetooth LE 16](#_Toc122120033)

[4.3 WiFi 16](#_Toc122120034)

[4.4 Összegzés 17](#_Toc122120035)

[**5.** **ESP-32** 18](#_Toc122120036)

[**6.** **ESP-NOW** 20](#_Toc122120037)

[**7.** **ESP-32 Kapcsolata a külvilággal** 22](#_Toc122120038)

[7.1 ESP WiFi módok 22](#_Toc122120039)

[7.2 REST API 23](#_Toc122120040)

[**8.** **Szenzormodul implementációja** 25](#_Toc122120041)

[**9.** **Főmodul implementációja** 28](#_Toc122120042)

[9.1.1 Szenzormodul hozzáadása 28](#_Toc122120043)

[**11.** **Két edzéstípus implementálása** 32](#_Toc122120044)

[**12.** **Tesztelés módja** 34](#_Toc122120045)

[**13.** **Ami még hiányzik, időbeosztás** 35](#_Toc122120046)

[**14.** **Irodalomjegyzék** 36](#_Toc122120047)

# **Absztrakt**

A dolgozat egy olyan rendszer létrehozását mutatja be, ami mikrokontrollerekből épül fel. A rendszer nyomásérzékelésen alapszik, az érzékelőből kinyert adatot dolgozzák fel a mikrokontrollerek. Az adatmozgást egy központi mikrokontroller irányítja. A dolgozat ezen felül tartalmaz egy ezen rendszerhez tartozó mobilos felhasználó felület lefejlesztését is. Ezzel kilépve a hardver közeli programozásból nyitva a felhasználói élményt megcélzó mobilos felületek felé is. A dolgozat folyamán ismertetem a különböző rádiófrekvenciás adatközléseket is, amiket a rendszer használ. Ezen felül érinti a harcművészetek világát is.

The thesis presents a system and its development. The system is built up by microcontrollers and the data it acquires is based on pressure measuring sensors. This data is recorded and then sent to a central computing microcontroller. Interaction with the system is possible through an application developed as part of the thesis. The connection between the different parts of the system is managed by wireless connection, so the thesis touches up on the different radiofrequency-based communication protocols used by the system as well. Other than that, we can get a glance at the world of martial arts.

# **Bevezetés**

A XXI. századra az információ vált az egyik legértékesebb dologgá a világon. Gyakran halljuk, hogy „information is the new oil”, ennek a hasonlatnak az az alapja, hogy egy nyers adathalmaz pont olyan értéktelen, mint a frissen szivattyúzott olaj. Mindkettőnek azután lesz értéke, hogy alaposan szortírozták, majd feldolgozták őket.

Az emberek pedig folyamatosan információt gyártanak, ugyanis számos készülékünk adatokat rögzít rólunk. Gondoljunk csak az utóbbi években megjelent okosórák, illetve fitneszkarkötők térnyerésére az elmúlt pár évben. Ezeken a felhasználó végig követheti napi lépésszámát, pulzusát, hogy mennyit és milyen minőségben aludt, valamint a készülék akár egy esést is képes érzékelni. Mostanra sportokat is teljesen átformálta a technológia, az edzéseket virtuális dómokban tartják a focisták, ahol az ügyességüket vagy a sebességüket tudják javítani. A meccseken a bírókat gólvonaltechnológia vagy éppen a video assisted referee (VAR) segíti ki.

A fentebb említett példákhoz képest a harcművészetek bizonyos el vannak maradva, bizonyos mértékig talán a hagyománytisztelő harcművészi mentalitás miatt is, ami nehezen összeegyeztethető az új technológiai elemek felhasználásával. Akár több száz éve létező, régóta kiforrott edzési technikákkal rendelkező küzdősportok esetén a mestereket nehéz meggyőzni, hogy próbáljanak ki új technológiákat mivel ezek jelentősen segíthetnek fejleszteni tanítványaik vagy akár saját képességeiket is. A mestereknek egyes esetekben életkoruk miatt is jobban nehezére esik egy-egy új eszköz használatának elsajátítása. Idővel azonban be kell majd látniuk, hogy előnyhöz juttatja a tanítványaikat a versenyeken és bajnokságokon, ha az felkészülési idő során a modern technika által kínált eszköztár is a rendelkezésükre áll. Ahogy Sun Tzu (Kr. e. 544-496), a Háború Művészete című mű alkotója írta:

“Ha ismerjük az ellenséget és ismerjük magunkat is, akkor száz csatában sem jutunk veszedelembe; ha azonban nem ismerjük az ellenséget, hanem csak magunkat, akkor egyszer győzünk, másszor vereséget szenvedünk; és ha sem az ellenséget, sem magunkat nem ismerjük, akkor minden egyes csatában feltétlenül végveszély fenyeget bennünket.”

Én magam is utánpótlás válogatott Kyokushin Karatés voltam, ezért közel áll a szívemhez a téma. Ezért is, illetve a fentebb már említett eszközhiány miatt választottam diplomamunkámnak egy edzéstámogató rendszer megtervezését, megépítését, valamint dokumentálását. A diplomamunkám elkészítése során az általam megtervezett rendszer segítséget nyújthat a reflexek javításában, a versenyzők által használt technikák finomításában, valamint a mozgáskombinációk megtanulásának könnyebbé és gyorsabbá tételében. Az eszköz segítségével a sportolók objektíven láthatják képességeik pillanatnyi állapotát és javulását. A rendszer megtervezésekor a legfontosabb szempontok közé tartozott a felhasználóbarát működés, hogy az eszköz minél modulárisabb és bővítésre nyitott legyen, a könnyű és gyors mozgathatóság, valamint, hogy minél kisebb erőforrásigénye legyen. Ezeken a szempontokon kívül az is fontos volt, hogy akár gyerekek is képesek legyenek használni a rendszert, ugyanis a legjobb sportolók már kiskoruk óta keményen edzenek és készülnek a versenyekre.

# **Követelményspecifikáció**

A diplomamunka keretein belül létrehozott rendszernek különböző feltételeknek kell megfelelnie, amiket a dolgozat ezen részében mutatok be, valamint felvázolom a rendszer absztrakcióját. Ennek a tervnek az elemeit a későbbiekben megvizsgálom és feltérképezem azokat a megfelelő elemeket, amiket végül az implementációs fázisban végül használni fogok.

A rendszerrel szemben támasztott főbb követelmények:

* 1. A rendszer vizuális úton képes jelezni a felhasználó felé, hogy melyik érzékelőre várja a találatot.
  2. A rendszer képes a bevitt találatot érzékelni és annak sikerességét vizuális úton a felhasználó felé jelezni.
  3. A rendszer képes az előző két pontban definiált lépések közti időt mérni és memória egységében eltárolni.
  4. A rendszernek biztosítania kell egy olyan kommunikációs felületet, amelyen keresztül a felhasználóval képes vezeték nélküli információcserére.
  5. A felhasználónak tudnia kell konfigurálnia a rendszert, újabb érzékelőket hozzáadni, valamint az edzésmódokat testreszabni.
  6. A rendszeren belüli egységeknek képesnek kell lenniük egymás között adott feladatuknak megfelelően kábelmentesen kommunikálni.
  7. A felhasználó képes legyen bármely pillanatban megszakítani a jelenleg futó folyamatokat.
  8. A rendszer képes összegezni az edzés alatt gyűjtött adatokat és ezekről információt szolgáltatni a felhasználónak.
  9. A rendszer képes naplózni („logolni”).

A rendszer tervezésekor a moduláris felépítés több okból is rendkívül fontos szempont volt. Az egyik, hogy az elemeket könnyen lehessen meghibásodás vagy esetleges hardware upgrade esetén cserélni, valamint, hogy minél nagyobb szabadságot biztosítson a felhasználónak az ütést detektáló felületek elhelyezésében. Végül, de nem utolsó sorban pedig mivel a kábelek folyamatos sérülés veszélyt jelentenek mind az felhasználókra, mind a rendszerre, a tervezés folyamán minden esetben preferáltam a vezeték nélküli megoldásokat.

A fentebb említett modularitást elérése céljából a rendszer a következő, egymástól elkülönülő és egymás között csak vezeték nélkül kommunikáló alrendszereket különítettem el:

1.     Szenzor modul

2.     Vezérlő/Főmodul modul

3.     Felhasználói felület

A következő részben az előző modulok részletezése, valamint szerepkörük leírása látható:

## 2.1 Szenzor modul

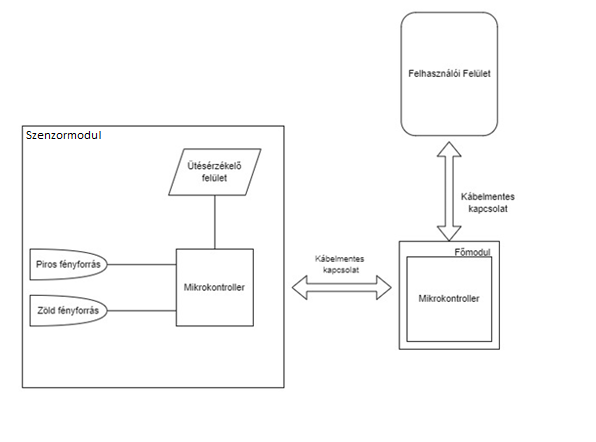
Ez a modul az, ami megvalósítja az 1-2. valamint a 3. pontban lévő időméréssel kapcsolatos követelményeket. Ebből a modulból egy rendszeren belül többet is lehet találni. Az 1. és a 2. pontban felírt vizuális úton lévő visszajelzést 2 különböző fényforrással éri el. Az egyik jelzi, hogy ez a felület várja a támadás azzal, hogy bekapcsol, majd, amikor a találat megtörténik a fényforrás lekapcsol és a másik színt kapcsol fel egy pillanatra, ezzel értesítve a sportolót a sikeres támadásról. Ahhoz, hogy a modul az ütést detektálhassa, szükséges egy olyan szenzor (a modul a nevét is erről kapta), ami képes azt érzékelni. A modul központi része egy mikrokontroller, ami képes a fényforrásokat, valamint a szenzort irányítani, illetve a szenzor által érzékelt nyomásértéket leolvasni. Ez a mikrokontroller nem áll kapcsolatban a külvilággal, a rendszeren belül is csak egyetlen központi egységgel, a vezérlő modullal kommunikál.

## 2.2 Vezérlő modul

A vezérlő modul a hardware különböző elemeinek összehangolásáért felelős. Fő feladata, hogy biztosítsa a kommunikációt az összes nagyobb elem között. A követelmények közül a 3. részeként összegyűjtött adatok tárolásáért felelős, de a legfontosabb feladata 4. követelmény, azaz a kommunikációs felület létrehozása a szenzor modul(ok) és a felhasználó között. A vezérlő modul a mikrokontrollerek számára összeköttetést nyújt a külvilág felé, így válik lehetségessé felhasználói oldalról a konfigurálás (5. követelmény), valamint az egyes edzésmódok elindítása. Ezáltal képes olyan interfészt biztosítani, amelyhez való csatlakozás a felhasználói felületről könnyedén megoldható. Itt történik a naplózás (9. követelmény). Ez a szenzor egy, a szakirodalom által „master-slave”-nek nevezett konstrukciót követve a master szerepkörét betöltve, a felhasználótól kapott parancsokat a korábban kiválasztott edzésmódnak megfelelően osztja ki a szenzor moduloknak, az edzés befejeztével pedig a begyűjtött adatokat a felhasználói felületre juttatja el (8. követelmény első fele).

## 2.3 Felhasználói felület

A rendszerrel, konkrétan a Főmodullal a felhasználó egy okostelefonnal képes kommunikálni (4.követelmény). A felhasználó egy alkalmazáson képes követni a fejlődését, felvenni kombinációkat, elindítani a különböző edzésterveket. Az alkalmazás az, ami az adatokat feldolgozza, a rendszer többi része csak az adatkinyerését végzi. A felhasználó ezen a felületen képes felépíteni a szenzor modulokból a saját virtuális bábúját, amin a kombinációkat képes létrehozni. Miután a kombinációk befejeződtek az alkalmazás képes kiértékelni, majd megjeleníteni a végrehajtás részleteit, mennyi idő alatt sikerült a kombináció, melyik támadás tartott esetleg több ideig végrehajtani. Ezekkel az adatokkal a sportoló képes megismerni a gyengeségeit. Az alkalmazás ezen felül lementi az összes sorozat minden információját, hogy amennyiben később vissza akarja nézni a teljesítményét a sportoló legyen rá lehetősége, valamint ezzel utat engedve későbbi vizsgálatok futtatására is.



2.1 ábra A rendszer absztraktja

2.4 Edzésmódok

A rendszernek kettő edzéstípust kell tudnia megvalósítani a diplomamunka keretein belül, a véletlen, valamint az adott kombináció módot.

2.4.1 Véletlen mód

Ennek a módnak a célja, hogy képet adjon a sportoló reakció idejéről, valamint, hogy segítsen a sportoló reflexeinek javításában, koordinációjának fejlesztésében. Ilyenkor nem az a fontos, hogy a támadások sorozata valós küzdelembe használható legyen, hanem hogy a sportoló a nyílt (védtelen) támadási felületet küzdelem közbe meglátva, a lehető leggyorsabban tudjon élni a lehetőséggel. Ehhez a rendszer véletlenszerűen kiválaszt egy ütési felületet és várja a kontaktot.

2.4.2 Adott kombináció

Az adott kombináció mód az, amivel az edzők képesek kombinációkat létrehozni, amiket a sportolóknak ezután lehetősége van a teljes fokú magabiztosság és maximális gyorsaság eléréséig begyakorolni. Ha elégszer gyakorlunk egy ilyen sorozatot, akkor az megmarad az izommemóriában. Ez az a szint, amikor már nem kell gondolkodnunk azon, hogy a mozdulatok hogyan követik egymást, egyszerűen csak automatikusan cselekednünk kell. Így érhetjük el azt, hogy egy különösen hosszú küzdelem során, amikor már mindkét versenyző fáradt, az egyik elindít egy kombinációt és ugyan az ellenfél már a második ütéstől elbukik, a mozgásban lévő fél ezt nem veszi figyelembe, hanem akár a levegőbe is befejezi a korábban elsajátított mozdulatsort.

# **Alternatív megoldások**

A 2. fejezetben megadott követelményeknek megfelelő rendszer keresése eredménytelen volt, nem találtam olyan hasonló rendszert, ami képes lenne az általam felállított követelményeket maradéktalanul teljesíteni. Találtam azonban két, funkcióját illetően valamennyire hasonló eszközt, amelyet a következőkben szeretnék bemutatni. Az egyik egy elvontabb, a másik azonban egy jóval egyszerűbb alternatíva.

## 3.1 „Simon”

A “Simon” egy elektronikus memóriajáték, amelyet 1978-ban hozott létre Ralph H. Baer Howard J. Morrisonnal együttműködve. A 70-es, valamint 80-as évek egyik legnépszerűbb pop kulturális játéka volt. A játékhoz szükséges eszköz 4 színes, világítani képes gombbal rendelkezik. Ezekhez a gombokhoz 4 különböző hangot rendeltek, melyeket az eszköz le tud játszani.

A játékmenetet a következőképpen zajlik minden körben:

1.     Felvillan eggyel több gomb, mint az előző körben, valamint lejátszódik mindig az adott gombhoz tartozó hang is.

2.     A játékosnak meg kell nyomnia a gombokat ugyanabban a sorrendben, amiben felvillantak

3.     A leírt procedúra addig ismétlődik, amíg a játékos el nem rontja a kombinációt.

A játék bár nem függ össze a harcművészetekkel, de jól bemutatja a fényforrások használatát instrukciók továbbítására a játékos felé. Valamint a játék alapját képező, folyamatosan növekvő, megjegyzendő kombináció kísértetiesen hasonlít a diplomamunka tárgyát képező rendszerre. A játék másik kommunikációs módszere, a hanghatások ugyanúgy hasznosak lehetnek és használatuk mindenképp megfontolandó a rendszerben.

## 3.2 Redflash

A második, és egyben ténylegesen harcművészetekhez készített eszközcsalád a Redflash, amelyben a RED az a „Reflex Energy Dynamism” rövidítése. Ez egy magyar cég által létrehozott termékcsalád, ami a diplomamunka írásakor 4 különböző „Pajzsból áll. Ezek részletes bemutatása a következő

A „[REDflash elektronikus kézi ütőpajzs](https://redflash.hu/redflash-kezi-utopajzs)” a „[REDflash elektronikus rúgópajzs](https://redflash.hu/redflash-rugopajzs)” és a „REDflash elektronikus fali ütőpajzs” funkcionalitásukat tekintve nem térnek el egymástól csak méretben, illetve a stabilizálási módban, ezért ezeket egyként kezelem a bemutatás során. A pajzsok első ránézésre nem különböznek egy hétköznapi pajzstól, megtalálható rajta az ütési felület, illetve valamilyen megoldás a tartásra, felszerelésre. Az egyedüli dolog, ami különbözik az egy kis kivágás, a pajzsok tetején, itt található a vezérlő.  Ez két gombból, valamint egy kis kijelzőből áll. Az edzés elindítása után megfigyelhető a másik eltérés a hétköznapi pajzsokhoz képest, felvillan egy LED csík a pajzs elején. A termék bemutatójában elhangzottak alapján az eszköz 0.5-5 másodperces intervallumon belül véletlenszerűen kapcsolja be a fényforrást, ezzel jelezve a sportolónak, hogy várja a támadást. A párna a reakcióidőt 0.001 másodperces pontossággal méri, amennyiben megtörtént a találat, újból 0.5-5 másodperces intervallumon belül felkapcsolja a fényforrást. A mérési adatokat 10-20-30-40-50 támadás után szolgáltatja vissza, átlagolva a reakcióhoz szükséges időket.

A képen szöveg látható

Automatikusan generált leírás  
3.1 ábra Redflash elektronikus ütőpajzs

A REDflash pajzscsalád másik típusa a „REDflash elektronikus ütőfal”, ami egy falra szerelhető, 7 kiütési ponttal rendelkező eszköz. A bemutató videó alapján ez áll a legközelebb a diplomamunkában létrehozott termékhez. A pajzs a már megismert fényjelzéssel ad jelet a sportolónak az ütés megkezdésére, azonban itt nem egy pajzsot, hanem egy nagyméretű nyomógombhoz hasonlatos támadási felületet kell eltalálni. Az eszköz a termékcsalád korábban ismertetett tagjaihoz hasonlóan kijelzi az ütéssorozat végén a találat beérkezéséig átlagosan eltelt időt. Ennél a pajzsnál van lehetőség 8db előreprogramozott kombináció betanulására is, amire a diplomamunka eredményeképpen megalkotott eszköz használata is alkalmat ad.

  
3.2 ábra Redflash elektronikus ütőfal

## 3.3 Miért másabb a MATTS?

A fentebb bemutatott, REDflash pajzscsalád volt az egyetlen általam talált olyan termék, amely ötvözi a technológiát és a harcművészeteket, de ezek is csupán korlátozottan tűnnek használhatónak. A pajzsok bár kiírják a legutolsó ütés beérkezéséig eltelt időt, a sorozat végén teljesítményünk elemzéshez már csak egy átlagos időintervallum áll rendelkezésünkre. Az átlagos idő kijelzésének hátulütője, hogy nem mérhető sem fáradás, sem pedig az, hogy esetleg vannak-e olyan kiütési pontok, amikre kimondottan lassan reagál a sportoló. Mivel a rendszer 0.5-5 másodpercenként véletlenszerűen kér támadást a sportolótól, nem is nagyon van lehetőség a pajzs áthelyezésére másik testrészre, azaz nem tudja szimulálni a valós küzdőhelyzetet. A versenyzőnek elég egyetlen helyre figyelni, hiszen nincs lehetőség több pajzs szinkronizálására, holott egy életszerű küzdelemben előforduló szituációban a sportolónak az ellenfél teljes testét figyelnie kell, az esetleges védtelen támadható felület megtalálására. A REDflash elektronikus ütőfal, bár hasonló helyzetet akar szimulálni, azonban a falra szerelt síkfelületen elszórt támadási pontok szintén nem hasonlíthatóak a valós küzdelemhez.

# **Kábel nélküli kapcsolat lehetőségei**

## 4.1 ZigBee

Egy IEEE 802.15.14 (standard az alacsony rátájú, kis távolságú, vezeték nélküli hálózatokra) alapú implementáció, amit okos otthonok, orvosi eszközök és egyéb kis energiafogyasztású eszközök használnak adatátvitelre. A tervezésekor a cél az volt, hogy egyszerűbb és olcsóbb megoldást biztosítson a Bluetooth-nál. A kis energiaigény általában a kisebb hatótávolságot is jelent. Ugyan a ZigBee képes 10-100 méternyi távolságra jelet küldeni, a 2.4Ghz-es sávon 10-20 méter a használható táv, ami 250kbit/s adatmennyiséget jelent. A távolság növelhető amennyiben több eszközt kötünk össze egymással, és ezeken keresztül továbbítjuk az adatokat. A technológia energiaigénye a 3 ismertetett elem közül a 2. helyre elegendő, kb 100mW. Ezt a protokollt leggyakrabban a Local Area Network, azaz Helyi hálózatok (LAN) rendszerek építésekor használják, azaz általában a diplomamunkában létrehozott rendszernél nagyobb hatókörű elemeknél.

## 4.2 Bluetooth LE

A Bluetooth LE a széles körben ismert Bluetooth technológia „Low Energy”, azaz alacsony energiaigényű változata. A Bluetooth egy rövid távú, ultra magas frekvencián (2.402- 2.48 GHz) alapuló vezeték nélküli standard. Működési elve a szórt spektrumú frekvenciaugrás, azaz az adatcsatorna másodpercenként változik a küldő és a fogadó által előre meghatározott módon. A Bluetooth 79 csatornával rendelkezik, míg a BLE 40-nel, a teljes sáv ugyanakkora, azonban a BLE az nagyobb csatornákra bontja fel azt (2MHz-esekre az 1MHz-vel szemben). A Bluetooth ezeket a csatornákat használja fel arra, hogy a feldarabolt üzenetből létrejött csomagokat elküldje a fogadó oldalnak. A BLE abban az esetben éri meg, amikor egymáshoz közeli, kis mennyiségű adatot megosztó eszközöket akarunk összekapcsolni. Óriási előnye, hogy csak akkor működik, amikor kapcsolatot vesznek fel vele, egyébként alvó állapotban van. A BLE-t Service-ek definiálásával lehet implementálni, amiket egy Universally Unique Identifier, azaz univerzálisan egyedi azonosító (UUID) segítségével lehet identifikálni, ami egy 128 bites azonosító. A Servicek általában Characteristics-ekből állnak, amik szintén rendelkeznek saját UUID-vel. A UUID-k alapján egy kliens rá tud kapcsolódni a BLE szerverre, adott UUID-vel elérni a Service-t majd azon belül a Characteristics-et.

## 4.3 WiFi

A WiFi egy vezeték nélküli hálózati protokoll család, amelynek már számos generációja jelent meg, jelen diplomamunka írásakor a 2018 óta létező, 6-os generációs jelzéssel ellátott a legújabb. Ezek 2.4 vagy 5 Ghz-en működnek és a maximális sebességük 9600 Mbit/s. Manapság már nehéz olyan eszközt találni, ami nem rendelkezik WiFi kapcsolat létrehozására használható modullal. A kommunikáció csomagok küldésével valósul meg, rádióhullámokon keresztül. Ahogy a rádióhullámoknál általában, a kommunikáció itt is a modulációval és demodulációval jön létre. Ez azt jelenti, hogy a digitális jelet analóg hullámokká alakítja, azt átküldi rádióhullámként, a vevő egység pedig visszaalakítja digitális jellé. A küldőnek ismernie kell a fogadó MAC címét, ami egy 48 bit-es egyedi azonosító. A WiFi által használt csatornák fél duplex megvalósítást követnek, azaz a kommunikáció megvalósulhat mindkét irányban, azonban egy adott időpillanatban csak egyik irányba van lehetőség adatküldésre. Amikor egy eszköz adatot küld a csatornán, mindegyik hálózaton lévő eszköz megkapja őket, azonban a hálózati kártya kiszűri azokat, amiket nem az adott eszköznek küldtek. A csatornák száma országonként változó, az EU-ban 13, míg a USA-ban 11 használatos. Minél több eszköz van adott csatornán annyiad részére csökken az átvihető adatmennyiség. A WiFi előnye a nagyfokú biztonság, és hogy az elterjedése miatt a legtöbb okos eszköz rendelkezik hozzá való modullal. Hátránya azonban, hogy a már felsorolt protokolloknál jóval nagyobb az energiaigénye.

## 4.4 Összegzés

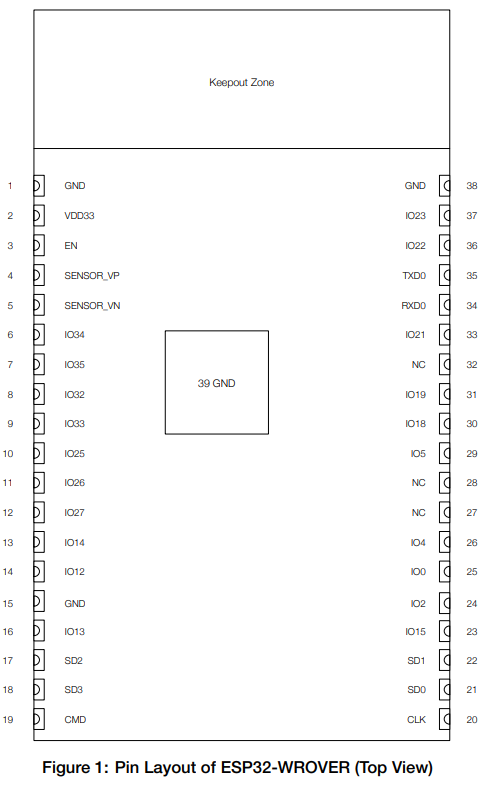
A fenti technológiákat összehasonlítva a projekthez leginkább a BLE, illetve a WiFi tűnik preferált megoldásnak, a ZigBee profilja hiába lenne megfelelő, sajnos a protokoll nem elterjedt a ma használatos fejlett operációs rendszerekbe, azaz mindenképp kellene használni egy második technológiát is, ami képes a mai eszközökkel is kommunikálni. A projekt PAN (Personnal Area Network) stílusa miatt a BLE tűnik használhatóbbnak, de a WiFi sokrétű felhasználhatósága miatt nem elvetendő. A mikrokontroller választása folyamán preferált, ha rendelkezik mind BLE mind WiFi támogatással.

# **ESP-32**

A használt eszközök bemutatásakor a Vezérlőmodulnál, valamint a Szenzormodulnál is mikrokontrollereket neveztem meg az adatok összegyűjtésére, továbbítására, valamint a szenzorok figyelésére. A mikrokontroller egy kis méretű számítógép, ami rendelkezik központi feldolgozóegységgel, memóriával, valamint programozható IO perifériákkal. A programot a saját ROM-ján tárolja. Ezek az egységek általában a beépített rendszerekben használatosak, kicsi méretük, valamint alacsony fogyasztásuk miatt. Általában kisebb számítási kapacitással rendelkeznek, valamint az áruk is alacsonyabb egy nagy teljesítményű rendszeréhez képest. Az egyik legelterjedtebb, jelenleg a hobbi barkácsolók körében nagy népszerűségnek örvendő mikrokontroller az Arduino. Ezek az eszközök könnyen használhatóak, kimondottan felhasználóbarát fejlesztői környezettel rendelkeznek. Ebben a fejezetben szeretném bemutatni, hogy miért is nem az Arduino-t, hanem egy másik mikrokontrollert, az Espressif által készített ESP32-t használom.

Az ESP32 egy, az Espressif (ami egy shanghaj-i székhelyű kínai cég) által létrehozott mikrokontroller. Kimondottan alacsony árkategóriás egységként tekinthetünk rá, azonban ehhez képest sok funkcióval rendelkezik. Egyik a projekt számára legfontosabb funkció, hogy a vezeték nélküli technológiák integrálva vannak a processzorba. Megtalálható mind a WiFi, mind a Bluetooth (dual mode) modul is, az egyébként kis méretű egységen. A dual mode jelen esetben azt jelenti, hogy használható mind a Bluetooth mind a Bluetooth LE a chipen akár egyszerre is. Ez az eszköz óriási előnye az Arduinoval szemben, ami csak jóval drágábban, illetve nagyobb méretű elemeken biztosítja ugyanezeket a lehetőségeket, vagy pedig elvárja egy külön elem megvételét, amit rá kell kötni a mikrokontrollerre. Az ESP32 rendelkezik több modellel is, én a wroover-b nevű egységet szereztem be.  Ez egy ESP32-D0WD központi feldolgozóegységgel van ellátva, ami két alacsony energiaigényű Xtensa 32 mikroprocesszorral rendelkezik.

Az ESP32 egy elterjedt mikrokontroller család, aminek nagy a felhasználóbázisa. Ennek köszönhetően sok videóanyag, valamint dokumentum, ezen túl több program könyvtár érhető el hozzá. A popularitását jól mutatja az is, hogy több fejlesztői környezetbe is integrálva lett az ESP32, ilyen például a Visual Studio Code, MicroPython, de a legtöbbet ajánlott és általam is preferált Arduino IDE. Az ESP32 modul telepítése egyszerű az IDE-n belül, ezután pedig használható az egyébként Arduino fejlesztésnél megszokott összes funkció. A dolgozat folyamán innentől kezdve a mikrokontroller tulajdonságairól, illetve funkcióiról beszélve mindig a Wroover-B egységre fogok hivatkozni. Az ára töredéke egy Arduino egységhez képest, ami fontos, hiszen több ilyen mikrokontrollert is használ a rendszer. Ezen felül kisebb az eszköz és többet is tud, mint az olcsó Arduinok.

A képen szöveg, elektronika látható

Automatikusan generált leírás

5.1 ábra ESP-32 talpazat, illetve használt eszköz képe

# **ESP-NOW**

A „Kábel nélküli kapcsolat lehetőségei” fejezetben a BLE protokoll volt, ami a leginkább kiszolgálta a rendszerrel szemben támasztott igényeket ezért is volt fontos, hogy a választott mikrokontroller rendelkezzen Bluetooth (és BLE) támogatással. Azonban az ESP-32 fórumokon valamit a dokumentációban is megjelölik, hogy a preferált módszer ESP-32 mikrokontrollerek közötti adatátvitelre az ESP-NOW technológia, amennyiben kicsi az átvitt adat.

Az ESP-NOW egy gyors, vezetéknélküli, kapcsolatmentes kommunikációra használható csomagokon alapuló technológia. A technológia a következő tulajdonságokkal rendelkezik:

* Titkosított, valamint titkosítatlan unicast kommunikáció
* Akár 250-byte-nyi adatot lehet küldeni
* Van lehetőség callback function megadására, ami jelzi, hogy sikerült-e elküldeni az üzenetet
* Az üzenetszórás nem támogatott
* Maximum 10 titkosított, de 20 akár titkosítatlan eszköz támogatott

A működéshez az ESP-32 lementi az információt az eszközről, aminek küldeni szeretne adatot. A legfőbb ilyen információ, ami szükséges az a MAC címe, amit mindenképp tudnia kell, ezt könnyen le lehet kérni minden egyes egységhez.

Ezt az alábbi kóddal lehet elérni:

A képen szöveg látható

Automatikusan generált leírás

6.1 ábra MAC cím lekérdezés

Ami kiírja a Monitorra az adott elem MAC címét, sajnos a vásárolt ESP-32-nek se a dobozán se a papírjain nem található meg ez a cím, ezért minden egyes használt egységre külön le kell futtatni ezt a kódot első használat előtt és letárolni valamilyen formában azt az eszköz mellé rakva.

A képen szöveg látható

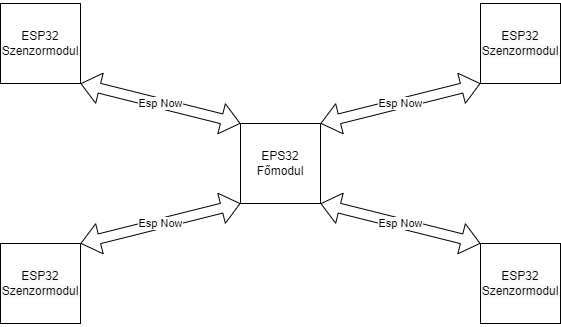
Automatikusan generált leírás

6.2 ábra a MAC kiírása

Az eszköz a peer-ről a következő információkat tárolja el:

* Key, Ez a titkosításnál használandó
* MAC Cím
* Role, Funkcionálisan nem különbözteti meg őket, csak az applikációs rétegben, ezek alapján lehet Idle, Controller, Slave, Combo.
* Channel, nem változtatja a funkcionalitást a hivatalos dokumentáció alapján, azonban bizonyos esetekben okozhat anomáliát a rossz csatorna használata.
* IFIDIX: ez adja meg, hogy melyik WiFi interfészt használja az ESP-NOW üzenet küldésére/fogadására a peer.

Az ESP-NOW használatának elve, hogy feliratkozik a mikrokontroller a különböző, az Espressif által előre deklarált eseményekre, amikre definiálni kell eseménykezelő metódusokat. Ilyen esemény például az esp\_now\_register\_recv\_cb amire feliratkozva a metódus akkor fut le, ha üzenet érkezik az ESP-NOW-ra. Ez a feliratkozott metódus a WiFi Task-on fog futni, ezért érdemes a kapott üzenetet lementeni és utána egy másik Task-al futtatni, hogy ne tartsa fel a WiFi task-ot. Ez az ESP-NOW dokumentációjában az ajánlott működési elv, ezért ezt is használja a rendszer, bár a dolgozat szempontjából nem lényeges, ugyanis egyszerre nem kommunikál több ESP egymással, egy időben egyszerre mindig csak egy feladó van és egy fogadó fél.



6.3 ábra ESP-NOW rendszer példa

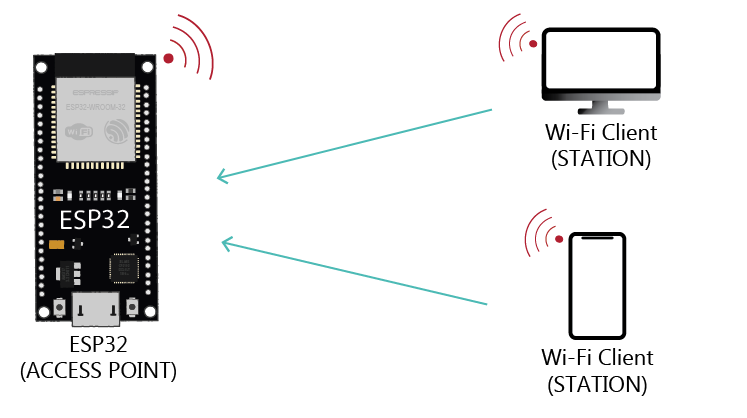
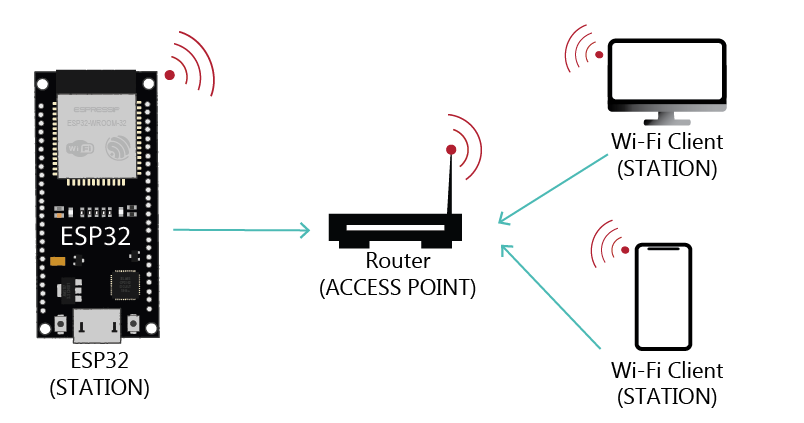
# **ESP-32 Kapcsolata a külvilággal**

A 6. fejezetben bemutatott ESP-NOW használata a mikrokontroller WiFi moduljának bekapcsolásával érhető csak el. A 4. fejezetben bemutatott kábel nélküli kapcsolatokra szánt protokollok közül a Bluetooth volt a legoptimálisabb, azonban ahhoz inicializálni kell azt a modult is, konfigurálni, ezzel a Főmodul két különböző kábelnélküli kapcsolatot kell fenntartania. A 4. fejezetben másik elfogadható megoldás volt a WiFi használata, ez már az ESP-NOW miatt úgyis üzemel, valamint a manapság használatos összes fejlett operációs rendszer használja, ezzel kimondottan megengedő a felhasználóit felület választása terén.

## 7.1 ESP WiFi módok

Az ESP-32 a Wi-Fi könyvtárán keresztül implementálja a konfigurációs, valamint monitorozó funkcionalitását. Az esp működhet 3 módban:

1. Station mód (Állomás) - Ebben a módban az ESP képes csatlakozni routerehez, amitől kap egy egyedi IP címet. Az ESP-vel ekkor a IP címe használatával a hálózaton belül megcímezhetővé válik, ha a Router csatlakozik az Internethez, akkor az ESP akár az internetre is képes üzeneteket küldeni
2. Access Point (Hozzáférési pont) – Ebben a módban az ESP egy Router szerepkört vesz fel. Létrehozza a saját hálózatát, amire más WiFi eszközök tudnak kapcsolódni (ezek a már említett Station szerepkört használják). Mivel az ESP nem csatlakozik ilyenkor az Internetre, ezért az ESP-t SoftAP-nak (gyenge hozzáférési pont) hívják. Azaz ilyekor értelemszerűen nem címezhetünk meg interneten elérhető címeket.
3. AP\_STA mód, ez hibrid mód, ilyenkor képes felváltva üzemelni az eszköz AP és STA módban.



Ábra 7.1 Különböző Módok szemléltetése

A módok közül bár a Station mód nagy szabadságot ad a tervezéskor, hiszen bármikor képes a kinyert adatokat továbbítani az internetre, sajnos azt jelenti, hogy valamilyen módon konfigurálni kellene az eszközt arra, hogy tudjon az adott használati helyen lévő hálózathoz csatlakozni, azaz megadni a hálózat SSID-ját, valamint jelszavát. Ezért a külvilággal a rendszer SoftAp módban kommunikál. Ahhoz, hogy ez a kapcsolat létrejöjjön, a setup metódusban meg kell hívni a WifFi.mode(WIFI\_AP) valamint a WiFI.softAP(ssid, password) metódusokat. Ez beállítja a módot, majd utána megadja a hálózathoz az SSID-t, illetve a jelszót. Az SSID „MATTS” a jelszó pedig a tesztelés folyamán „1234”. Ezeknek az információknak a birtokában, bárki képes csatlakozni a hálózathoz.

## 7.2 REST API

A SoftAp implementálására kétféle ajánlott megoldási módot találtam az interneten, az első módszer az, amikor egy html dokumentumot érhetünk el a hálózatra kapcsolódva és ezzel egy „internetes” felülethez hasonló konfigurálási opciókat tudunk megtenni. Ezzel a megoldással a probléma, hogy bár minden eszközről elsőre elérhetővé válna egy böngészőből, sajnos egy applikációba integrálása (ami fontos a felhasználó kezeléshez, fejlettebb, könnyebben frissíthető UI, illetve UX-et biztosít).

A fenti ok miatt a második megoldást használom, ami egy Rest API létrehozása a Főmodulon. A REST a „REpresentational State Transfer” rövidítése, az API pedig az „Application Programming Interface”. Ahhoz, hogy megértsük a REST API működését szeretném bemutatni a 6 szempontot, aminek meg kell felelnie egy webszolgáltatásnak, hogy annak nevezzük:

* 1. Egyértelmű, rendszeresített interfész: Minden egyes a szolgáltatás által használt adatot egyféleképpen kell megosztani a külvilággal, valamint érdemes egyetlen URL-en keresztül engedélyezni ennek a módosítását. Az adott elemek interfészei követniük kell egy jól definiált változó nevekre vonatkozó szabályozást, valamint változó típusok egységességét (például számra mindenhol double használata).
  2. Kliens-Szerver elkülönítése: Ennek a meghatározásnak a gyakorlatban az implementálása azt jelenti, hogy a kliens és a szerver képes egymástól teljesen elkülönülve fejlődni, a kliens csak az URL-t tudja, nem épít és nem is ismeri a szerver kódbázisát.
  3. Állapotnélküliség: A szerver minden egyes lekérést úgy kezel mintha sose történt volna lekérdezés adott IP címről. Ez azt jelenti, hogy szerver oldalon nincsen lementve adat a lekérdezések feladójáról. Amennyiben kell azonosítás vagy autorizáció akkor a lekérdezésnek biztosítania kell mindig ezeket az adatokat.
  4. Gyorsítótár használata: Szerver oldalon azokat az adatokat, amiket lehetséges érdemes gyorsítótárba venni, ezzel csökkentve az elérési időt (ez a dolgozat folyamán számunkra nem releváns)
  5. Rétegezhető: A kliens nem tudja, hogy egy szerverrel, esetleg többel kommunikál. Lehetővé kell tenni, hogy ez teljesen elfedhető legyen a felhasználó számára.
  6. Kód, ha szükség van rá (opcionális): Ez egy opcionális megkötés, arra irányul, hogy a REST API-nak képesnek kell legyen igény esetén futtatható kódot visszaadnia.

A fentebb említett megkötéseket, amennyiben lehetséges és van relevanciája követi a diplomamunkában létrehozott webszervíz. Az ESP-32 WebServer nevű osztály használatával képes a webszervízt létrehozni. Meg kell határozni a használni kívánt port-ot, ez a 80-as port a dolgozat esetében, ugyanis ez a http protokoll által használt, amit a REST kialakításakor használhatunk. Ezután meg kell adni az URL-eket, amiken keresztül a felhasználói felületről elérhetjük a szervízt. Ezt az alábbi kódrészlet teszi meg:

A képen szöveg látható

Automatikusan generált leírás

7.2 ábra Webszerver API definiálása

A server az egy WebServer osztály példánya és ezen az on() függvény pedig az eseménykezelő hozzáadását végzi el. Ehhez meg kell adni az URL-t, amin a hívás történik, mint például „/listlinked”. Meg lehet adni, hogy milyen http lekérésről van szó adott esetben (ez alapállapotában a HTTP\_GET). Majd az utolsó paraméter a metódus, ami kezeli, ha adott URL-re adott http lekérés érkezik meg. Ezután a server.hasArg(„plain") metódussal megkaphatjuk lekérdezésben használt esetleges JSON adat String-esített formáját. Ezt utána deserializál-hatjuk egy StaticJsonDocument-be, aminek meg kell adni, hogy mennyi byte-nyi információt tárolunk. Ezen már a JSON objektumoknál standard jsonDocument[„kulcs”] hívással elérhetjük az objektum adott értékeit.

# **Szenzormodul implementációja**

A követelményspecifikációban részletezett felelősségek közül a Szenzor modulnak feladata a:

1. Képes üzenetet fogadni a Főmodultól
2. Fényforrással jelzés a sportolónak, hogy támadhat
3. Ütés detektálása, fényforrással jelzés a találatról a sportolónak
4. Eltelt idő érzékelése és elküldése a Főmodulnak

Ahhoz, hogy az 1. követelményt teljesítse, a már fentebb említett ESP-NOW protokollt kell inicializálni. Ehhez meg kell határozni a struktúráját az üzenetnek a könnyebb adatmozgatás érdekében. Ezt a struktúrát létre kell hozni és fontos, hogy ugyanaz legyen mind a Főmodulban mind a Szenzormodulban.

A képen szöveg látható

Automatikusan generált leírás

8.1 ábra Üzenet struktúra az ESP-NOW kapcsolatnál

Az üzenetben a következő elemek találhatóak:

* ID – A rendszeren belül kapott ID-ja Szenzormodulnak.
* Milis – Ez a mező tárolja el az ezredmásodpercet, ami alatt megtörtént a támadás.
* Mac – A mac címe a küldő félnek, a Szenzormodulnak így nem kell beadni a Főmodul címét azt hozzáadja az első üzenet megérkezésekor.

Az ESP-32 Setup metódusában (Ez az a metódus, ami lefut minden egyes induláskor) inicializálni kell az esp-now-t ez amennyiben nem sikerül akkor a rendszer újraindítja magát remélve, hogy az megoldja az esetleges hibát, ugyanis hibás init után nem lehetne használni az eszközt hiszen nem tudna parancsot kapni.

A képen szöveg látható

Automatikusan generált leírás

8.2 ábra ESP-NOW inicializálása

Amennyiben az init sikeres volt regisztrálni kell az ESP-NOW által használt eseménykezelő metódusokat. Az „esp\_now\_register\_send\_cb”-ben regisztrált metódus akkor hívódik meg amikor a Szenzormodul üzenetet küld a Főmodulnak, míg a „esp\_now\_register\_recv\_cb” az, ami üzenet érkezésekor.

Az üzenet érkezésekor az „incomingData” -t bemásolja a saját myData struktúrába. Amennyiben az első üzenet érkezik a Főmodultól, a Szenzormodul lementi a kapott ID-ját, majd átáll figyelő módba. Az ID-k használatával elkerülhetőek és számolhatóak a mellé ütések, valamint adhatóak meg speciális utasítások is.

Amennyiben az id mezőben -99-es érték van, a szenzor mindenképpen átáll pihenő módba.

A 2. követelmény a fényforrással való jelzés, ehhez 1-1 LED izzót használ a rendszer. Az ESP-32 több lábbal is rendelkezik, amin keresztül képes egy LED izzó fel/le kapcsolására. Amit viszont fontos említeni, hogy a tesztelések folyamán kiderült, a WiFi modul bekapcsolásakor (ami elengedhetetlen az ESP-NOW használatához) több PIN is használhatatlanná válik, ezt később a fórumon [3] több poszt alapján megerősítették, hogy nem eszközhibáról beszélhetünk, hanem valamilyen tervezési hibájáról az ESP32-nek. Az első lámpa felgyullad, ahogy átáll az eszköz figyelő módba. Ez a lámpa a GPIO4-re van kötve.

A 3. követelményhez az ütésérzékelő szenzor szerepét egy FSR szenzor látja el. A szenzorok olyan eszközök, amik a világ fizikai vagy kémiai tulajdonságait képesek digitalizálni. Az FSR a Force Sensitive Resistor rövidítése, ez magyarul „erőbehatásra érzékeny ellenállás” -nak feleltethető meg, ami jól leírja a működési elvét is az egységnek. Amennyiben nyomást fejtünk ki az érzékelő felületre, akkor változik az ellenállása, ezt pedig a feszültségváltozást figyelve az egység két lába között tudja érzékelni egy szenzor. Bizonyos mértékig lehetséges számolni az ellenállás változás mértékéből az őt ért nyomást is, azonban ez az ütések rúgások intenzitásának mérésére nem elegendőek, a nagyobb egységek sem képesek pontos értéket adni ezeknek az erejéről, csak a becsapódás tényét képesek jelezni. Az FSR szenzor ellenállását a GPIO32-es lábon olvassuk. A GPIO32-es lábon a feszültség változásának mértékétől függően tudjuk észlelni az ellenállásváltozást, ami valamilyen erőbehatást indikál a szenzoron.

A képen szöveg látható

Automatikusan generált leírás

8.3 ábra FSR szenzor

Amikor az eszköz átáll figyelő módba lekéri és lementi a millis funkció visszatérési értékét egy startMillis változóba. A millis() egy beépített függvény, ami lekéri a mikroprocesszor indítása óta eltelt ezredmásodpercek számát. Amikor az ütést érzékeli az eszköz újból lekéri a millis funkció értékét és kivonja a startMillis értékéből, ezzel megkapva ezredmásodpercre pontosan azt az időt amire a sportolónak szüksége van az izzó felkapcsolásától az ütés pillanatáig. Ezután egyből elküldi az eltelt időt a Főmodulnak, valamint felkapcsolja 0.3 másodpercre a másik LED izzót is, ami a GPIO2-re van kötve. Amennyiben újra az adott szenzor következik az edzés folyamán, akkor a második LED izzó lekapcsol a 0.3 másodperc lejárta előtt is.

A szenzormodul kötési rajza az alábbi ábrán látható:

A képen szöveg, elektronika látható

Automatikusan generált leírás

8.4 ábra A szenzormodul kötési rajza

A wokwi.com-on sajnos nincs FSR szenzor, ezért az ugyanúgy ellenállás változtatáson alapuló potenciométerrel helyettesítettem.

# **Főmodul implementációja**

A főmodul felelős a felhasználói felülettel való kommunikálásra, ez az eszköz a master-slave kapcsolatból a master szerepkört tölti be az ESP-k között. A felhasználóknak egy webszerver üzemeltetésével képes információt biztosítani, valamint a rendszer konfigurálására is ezt a REST API-t tudják elérni a felhasználói felületről. Ez az eszköz tárolja el az edzés folyamán mért támadási időket is, amiket a felhasználói felület az edzés befejezéséve képes lekérni. A rendszeren belül használt elemeket itt lehet hozzáadni/törölni.

9.1 Rendszer konfigurálása

A rendszer konfigurálása a már bemutatott REST API-n keresztül történik. Ezt képes elérni bármelyik a Főmodul által létrehozott WiFi hálózathoz kapcsolódott eszköz, akár UI nélkül is.

### 9.1.1 Szenzormodul hozzáadása

Ahhoz, hogy hozzáadhassunk egy új Szenzormodult a rendszerhez, miután csatlakoztunk a WiFi hálózathoz, nem kell mást tenni, mint egy http requestet indítani a felhasználói felületről. A http request-nek tartalmaznia kell a hozzáadni kívánt eszköz MAC címét (Ezt a 6. fejezetben taglaltak szerint előre lekérdezzük és tároljuk a mikrokontroller mellett, hogy könnyen leolvasható legyen), valamint a helyének azonosítóját. Ezeket egy JSON objektumban küldjük, aminek a struktúrája a következő:

{

"mac\_addr": "10:97:BD:E2:D6:B8",

"place\_id": "0"

}

Ezt a request-et a /addSlave URL-en várja a Szenzormodul egy POST request-ként. Amennyiben ezek teljesülnek meghívódik az addSlavePost metódus, ennek keretein belül a következők történnek:

* 1. A program megnézi, hogy a letárolt place\_id változó kevesebb-e mint 10. Amennyiben nem akkor hibát jelez vissza a felhasználónak a http response-ban. Ha kevesebb akkor átalakítja MAC címre ({0x10, 0x97, 0xBD, 0xE2, 0xD6, 0xB8} formátumra).
  2. Miután megtörténik az átalakítás a MAC címet bemásolja a peerInfo (esp\_now\_peer\_info\_t példánya) peer\_addr mezőjébe, valamint beállítja a titkosítást hamisra, a csatornát a webszerver által is használt csatornára (ez nagyon fontos, az esp-now és a WiFi egyazon csatornán kell, hogy működjenek), valamint beállítja a peer WiFi interfészét WIFI\_IF\_STA-ra. Ezután megpróbálja felvenni az ESP-NOW peer-ek közé a beállított peerInfo alapján. Amennyiben nem sikerül, a hibaüzenetet elküldi a felhasználói felületre (Külön van „már létezik ilyen peer” hibaüzenet, ezzel lekezelve azt a lehetőséget is). Amennyiben sikerül, folytatódik a folyamat.
  3. A linkedDevices tömbben a place\_id helyre beírja a MAC címet.
  4. Sikeres hozzáadásról tájékoztatja a felhasználóifelületet a http response-ban.

9.1.2 Szenzormodul törlése

A Szenzormodul törlésére a /removeSlave URL-t lehet használni, ez egy ugyanolyan http POST request, amiben elküldi a felhasználó a MAC címet, valamint az ID-t. Ugyanaz a JSON struktúra használatos, mint a szenzormodul hozzáadásakor. A program a következő módon fut:

1. A rendszer megnézi, hogy a linkedDevices place\_id helyén ténylegesen van-e szenzor MAC cím beírva. Ha nem akkor hibajelzést küld a felhasználónak.
2. A kapott MAC címet megfelelő formátumra alakítja a program.
3. Az esp\_now\_del\_peer metódust meghívja az átalakított MAC címmel. Amennyiben nem sikerül (nem volt ilyen MAC cím társítva) hibajelzést küld a felhasználónak a http response-ban.
4. A linkedDevices place\_id helyén lévő elemet kinullázza.

9.2 Edzés megszakítás és kiértékelése

Az edzés folyamán bármikor meg kell tudnia szakítani a felhasználónak a gyakorlatot, ehhez a /stopTraining URL-re küldött http request-re figyel a rendszer. A folyamat a következőképpen zajlik:

* + 1. A stopTraining metódus meghívódik, ebben megnézi, hogy ténylegesen fut-e edzésprogram a rendszeren. Ha nem akkor hibajelzést küld az eszköz.
    2. Amennyiben igen, akkor egy ciklusban minden egyes a linkedDevices-ben felvett szenzormodulra elküldi egy üzenetet, mégpedig úgy, hogy az id mezőjébe az üzenetnek -99 -et ír be (Ezt a 8. fejezetben leírt módon kezeli a Szenzormodul)

Az edzés kiértékelése a felhasználói felületen történik, a Főmodultól csak az adatokat kéri le. Erre a Főmodul a /getTrainingDetails URL-t használhatjuk. Ez egy JSON formátumban válaszol a felhasználói felületnek. A JSON formátuma:

{

"trainingDone": "true",

"details": "784;error;232;822"

}

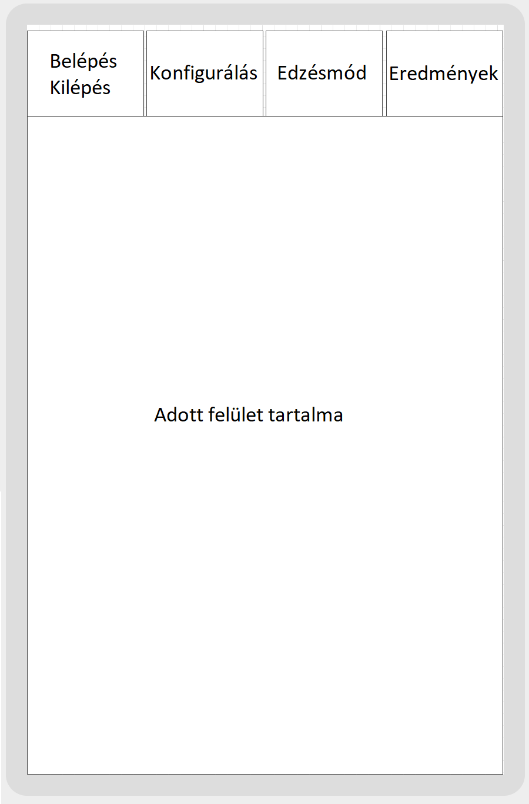
A trainingDone mező jelzi, hogy befejezte-e a legutolsó gyakorlatot a sportoló, vagy pedig még fut a gyakorlatsor. A details mezőben találjuk meg az adott támadásokhoz tartozó szükséges ezredmásodpercek számát, valamint az esetleges félreütéseket. A felhasználói felület ez alapján képes összegezni az edzéseket. A details-hez folyamatosan fűzi a rendszer az újabb értékeket, egészen a gyakorlat végéig, amikor is a következő lekérdezésnél már a trainingDone true, azaz ténylegesen kiértékelhető az egész gyakorlat.

1. **Felhasználói felület**

A felhasználó felület tervezésekor végig az volt a cél, hogy minél egyszerűbben lehessen akármilyen jelenleg elterjedt fejlett operációs rendszeren megvalósítani, legyen szó okostelefonról, webes felületről, vagy éppen alkalmazásról. A Főmodul által létrehozott WiFi hálózathoz bármelyik ilyen eszköz hozzá tud kapcsolódni, a REST API-t pedig akár felhasználói felület nélkül egy POSTMAN-el (számítógépen) vagy bármilyen REST API klienssel lehet használni. Azért, hogy ténylegesen élvezhető felhasználói élményt biztosítson és ne csak technikailag legyen használható a rendszer, ezért egy android-os applikáció fejlesztése is része a dolgozatnak.

Az applikációnak 4 fő komponense van, ezeket lehet elérni a főmenüből:

1. Belépő felület
2. Konfiguráló felület
3. Edzésmód felület
4. Kiértékelő felület



10.1 ábra A alkalmazás tervrajza

10.1 Belépő Felület

A belépő felület azért felel, hogy el tudja különíteni a sportolókat egymástól az adott készüléken. A felület rendelkezik 2 mezővel, amibe felhasználó és jelszó írható, valamint két gombbal. Miután a felhasználó beírta az azonosítókat, vagy regisztrál az egyik gombbal, vagy belép a másikkal. Amennyiben van már adott felhasználó, akkor hibajelzést ad, egyébként pedig jelzi, hogy sikeres regisztráció történt.

A jelszavakat SHA-512-vel hash-eli el a program, majd ezután menti le a belső memóriájába a felhasználónévvel együtt egyszerű szöveges formátumba JSON architektúrában.

A sikeres belépést követően a felhasználót továbbítja a Konfiguráló felületre.

10.2 Konfiguráló felület

A felhasználó itt képes felépíteni a Szenzormodulokból a rendszert. Egy előre létrehozott bábu sziluettjén képes hozzáadni és elhelyezni a szenzorokat. Itt a már a Főmodulnál bemutatott place\_id a bábu adott helyzetét jelöli, azt küldi el a MAC cím mellett a /addSlave URL-re. A törlésre is itt van lehetősége a felhasználónak. Az alkalmazás kezeli elvégez MAC cím-re vonatkozó ellenőrzéseket, ezzel nem engedve, hogy olyan struktúra kerüljön a Főmodulhoz, ami problémát okozhat a MAC címre fordításkor.

10.3 Edzésmód felület

Ezen a felületen lehet kiválasztani a kívánt edzésmódokat, valamint megszakítani az adott gyakorlatot. A felhasználó két mód közül választhat, a véletlenszerű, valamint az adott kombinációs módok. A véletlenszerű kiválasztásakor meg kell adnia, hogy hány darab sikeres támadást vár el a gyakorlat folyamán. Ekkor elküldi a /random URL-re ezt a számot. A kombinációs mód kiválasztása esetén megjelenik a már megismert bábú sziluettje és azokon a helyeken, ahova a felhasználó felvett Szenzormodulokat lehetővé válik a Szenzormodul megnyomása, ezzel hozzáadva egy veremhez, ami a kombináció place\_id-kat tartalmazza. Természetesen a felhasználó bármikor kiveheti az utoljára hozzáadott elemet. Amennyiben a felhasználó befejezte a kombinációhoz az elemek hozzáadását, a rendszer megkérdezi, hányszor ismételje meg a sorozatot. Ezután ezt a két adatot elküldi a /specificTranining URL-re JSON formátumban. Ezután adott időnként lekéri a gyakorlat státuszát a Főmodultól egészen addig amíg az be nem fejeződik.

A fentebb említett két mód pontos implementációjára a 11. fejezet tér ki bővebben.

10.4 Kiértékelő felület

Az edzésmód befejeztével a felhasználó felület kiolvassa a Főmodultól kapott értékeket, összeszámolja a hibák számát, valamint összefűzi a kapott időket adott helyekkel. Ezen a felületen megjeleníti az egyes helyekre az átlag támadáshoz szükséges időt, valamint a teljes átlagot a kombinációra nézve. Kiírja a hibák arányát. A felületen listába szedve található meg adott napok statisztikái, azon belül is gyakorlatokra szűkítve. Azért, hogy ez megoldható legyen, a felhasználónévhez vannak csatolva a lementett összefűzött eredmények. A statisztika kiszámolását a rendszer elvégzi minden egyes lekérdezésnél.

# **Két edzéstípus implementálása**

A véletlen kombináció implementálása:

A rendszer ezt így valósítja meg:

1. A felhasználói felületen kiválasztja a felhasználó a véletlen kombináció módot, ez értesíti a Főmodult REST API-n kapcsolaton keresztül, egy számértékkel, ami a szükséges sikeres támadások száma, ez lementésre kerül a siker számlálóba.
2. A Főmodul létrehoz egy véletlenszerű számot egy a nullától a szenzorok számánál egyel kisebb zárt tartományba. Ezután az asszociációs tábla alapján megadott MAC címre megküldi a Szenzormodulnak, hogy kezdjen el figyelni a támadásra
3. A Szenzormodul felkapcsolja a lámpát, valamint lementi az éppen aktuális időpontot egy változóba. Ezután várja a kontaktot.
   1. Megtörténhet, hogy nem a megfelelő felületet ütötte meg a sportoló, ekkor a nem megfelelő szenzorhoz tartozó mikrokontroller elküld egy hibás találat-ot jelző üzenetet a Főmodulnak. Ennek hatására a hibaszámláló változó értéke egyel nő.
   2. Amennyiben jó érzékelőt talált el a sportoló, akkor elküldi a Főmodulnak a jelenlegi időpontból kivont mentett időpont értékét, ezzel megkapva, hogy ténylegesen a lámpa felkapcsolásától kezdődően mennyi időbe tellett a sportolónak eltalálni a célpontot. A Főmodulon ezután csökken a siker számláló értéke egyel. Valamint hozzáadódik a teljes idő változóhoz az eltelt idő.
4. Ezután a kettes ponttól kezdődően ismétlődik egészen addig amíg el nem éri a siker számláló a nullát, azaz megvolt pontosan annyi sikeres támadás amennyit a felhasználó kért.
5. A felhasználói felület bizonyos időnként lekérdezi az edzés státuszát a Főmodultól, amennyiben a sikerszámláló 0, a Főmodul elküldi a kinyert adatokat a Felhasználói Felületre.

Adott kombináció implementálása

Ezt a következőképpen éri el a rendszer:

1. A felhasználó a telefonon kiválasztja az „Adott Kombináció” módot.
2. A telefon kijelzi a konfiguráció folyamán létrehozott szenzorokat, a felhasználó pedig megérinti az egyiket. Ekkor a képernyő alján megjelenik a szenzor azonosítója, valamint bekerül egy verembe az azonosító.
   1. Amennyiben meggondolja magát a felhasználó kitudja törölni egyesével a veremből.
3. Miután végzett a kombináció felvételével a felhasználó, megadja, hogy hány ismétlést kell csinálnia a sportolónak, ezután a telefon elküldi a sorozatot a Főmodulnak http requesten.
4. A Főmodul lementi egy tömbbe ezt.
5. A siker számlálót 0-ra állítja. Valamint a teljes idő változót is.
6. Kiolvassa a tömb „siker számláló” -adik elemét. Ez lesz a kombinációban következő elem azonosítója.
7. Kiküldi a megfelelő Szenzormodulnak a parancsot, hogy várja a találatot.
   1. Megfelelő találat esetén a Főmodul megkapja a találathoz szükséges időt, amivel növeli a teljes időt, majd inkrementálja a siker számlálót.  8-as pont
   2. Amennyiben rossz elemet ütött meg a sportoló felvillan a piros LED lámpa.  5-ös pont
8. Megvizsgálja, hogy a tömb mérete kisebb-e, mint a siker számláló (Van-e még hátra elem a kombinációból.)
   1. Amennyiben igen  6-os pont
   2. Ha nem  9-es pont
9. A főmodul elküldi a kombinációhoz tartozó időt a telefonnak, ami megnézi, hogy meg volt-e a megfelelő számú ismétlés.
   1. Ha igen akkor a sorozat befejeződik és az értékek megjelennek a telefonon.
   2. Ha nem  10-es pont
10. A telefon csökkenti a szükséges sorozatszámot egyel, majd elküldi a kombinációt újra a Főmodulnak http request-ként.  4-es ponttól folytatódik.

# **Tesztelés módja**

A tesztelés folyamán két ESP32-t használtam, az egyiket Főmodulként a másikat Szenzormodulként. A Szenzormodult a 8.4-es ábrán bemutatott módon állítottam össze.

A képen szöveg, elektronika látható

Automatikusan generált leírás

12.1 ábra a Szenzormodul tesztmodulja

A tesztelés során az FSR szenzor helyett egy potenciométert használtam, ami ugyanúgy az ellenállás változás mértékét képes jelezni fizikai behatásra. A Főmodul, valamint a Szenzormodul is a tesztelés folyamán USB-ről kapják a tápellátást. Ez a hibakövetés miatt is fontos, hiszen a soros monitorra USB-n keresztül képes az ESP32 folyamatosan üzeneteket küldeni, ezzel segítve a fejlesztést.

A Főmodul által létrehozott hálózatra egy Samsung Galaxy A52S-el kapcsolódtam majd a Google Play áruházban elérhető „REST Api Client”-et használtam a webszolgáltatás tesztelésére, amíg el nem készül a felhasználói felület. Ez az alkalmazás lehetővé teszi, hogy adott IP adott URL-jére lehessen JSON objektumokat küldeni http requestként.

# **Ami még hiányzik, időbeosztás**

A hiányzó elemek a Diplomamunka befejezéséig a következőek:

* 1. Felhasználói Felület létrehozása
  2. Átfogó integrációs tesztek megírása
  3. Tesztesetek elvégzése
  4. Dokumentáció 1-3. folyamatokkal egyidejűleg

+1 Ütési felület elkészítése

A fentebb említett 4 pont a dolgozat befejezéséhez elengedhetetlen, az 5. pont pedig az ütési felület újra tervezésére vonatkozik. A jelenlegi konstrukció az FSR szenzorral két faelem elem között nem elegáns és hosszútávon nem használható. Amennyiben lesz idő az újra tervezésre szeretnék egy kifinomultabb, szivacsos, ruhaanyaggal körbevont tényleges pajzsot létrehozni, ezzel komolyabb megjelenést biztosítva az eszköznek.

A Diplomamunka 3 befejeztével eljutottam oda, hogy az eszközök képesek egymással kommunikálni, a Főmodul ellátja a master szerepkört és képes több Szenzormodult összehangoltan irányítani, a kombinációkat elvégeztetni.

A tanulmányaimat tekintve pedig elvégeztem az előírt összes tárgyat az MSc képzésen, ezzel az utolsó félévben csak a Diplomamunka 4 befejezésére kell koncentráljak. Ennek megfelelően az időbeosztás a 2023 tavaszi félévre vonatkozóan hónapokra lebontva:

|  |  |
| --- | --- |
| Hónap | Cél |
| Január | Felhasználói architektúrájának megtervezése,  Optimalizálás a kódon |
| Február | Felhasználói felület megírása |
| Március | Tesztesetek definiálása |
| Április | Tesztelés |
| Május | Dokumentáció véglegesítése |

Természetesen a dokumentálás a folyamatokat végig követi és így májusra már csak a dokumentáció véglegesítése marad, ezzel időt hagyva a védésre készülésre esetleges csúszásokra.

# **Irodalomjegyzék**

Idézet:

[1] <https://mek.oszk.hu/01300/01345/01345.htm> : 2022.12.14

[2] <https://www.link-labs.com/blog/bluetooth-zigbee-comparison> : 2022.12.14

[3] <https://www.esp32.com/viewtopic.php?t=8728> : 2022.12.14

Felhasznált irodalom:

<https://docs.espressif.com/projects/esp-idf/en/latest/esp32/api-reference/> : 2022.12.14

<https://en.wikipedia.org/wiki/Bluetooth> : 2022.12.14

<https://en.wikipedia.org/wiki/Wi-Fi> : 2022.12.14

<https://en.wikipedia.org/wiki/Simon_(game)> : 2022.12.14

<https://redflash.hu/> : 2022.12.14