

UNIVERZA V LJUBLJANI  
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Andraž Novak

**Porazdeljeni sistem za pridobivanje,  
hranjenje in analizo podatkov o hrupu  
v urbanem okolju**

DIPLOMSKO DELO

UNIVERZITETNI ŠTUDIJSKI PROGRAM  
PRVE STOPNJE  
RAČUNALNIŠTVO IN INFORMATIKA

MENTOR: prof. dr. Blaž Zupan

Ljubljana, 2020

COPYRIGHT. Rezultati diplomske naloge so intelektualna lastnina avtorja in Fakultete za računalništvo in informatiko Univerze v Ljubljani. Za objavo in koriščenje rezultatov diplomske naloge je potrebno pisno privoljenje avtorja, Fakultete za računalništvo in informatiko ter mentorja.

*Besedilo je oblikovano z urejevalnikom besedil L<sup>A</sup>T<sub>E</sub>X.*

Fakulteta za računalništvo in informatiko izdaja naslednjo nalogu:

Tematika naloge:

Besedilo teme diplomskega dela študent prepiše iz študijskega informacijskega sistema, kamor ga je vnesel mentor. V nekaj stavkih bo opisal, kaj pričakuje od kandidatovega diplomskega dela. Kaj so cilji, kakšne metode uporabiti, morda bo zapisal tudi ključno literaturo.



*Na tem mestu zapišite, komu se zahvaljujete za izdelavo diplomske naloge. Pazite, da ne boste koga pozabili. Utegnil vam bo zameriti. Temu se da izogniti tako, da celotno zahvalo izpustite.*



Svoji dragi Nanici.



# Kazalo

## Povzetek

## Abstract

|  |           |
|--|-----------|
| <b>1 Uvod</b>                                      | <b>1</b>  |
| <b>2 Tehnologije</b>                               | <b>5</b>  |
| 2.1 Mikrokontrolerji . . . . .                     | 5         |
| 2.2 Merjenje hrupa . . . . .                       | 5         |
| 2.3 Povezljivost . . . . .                         | 9         |
| 2.4 Programska oprema za internet stvari . . . . . | 10        |
| 2.5 Strežniški del . . . . .                       | 11        |
| <b>3 Rešitev</b>                                   | <b>13</b> |
| 3.1 Uporabniške zahteve . . . . .                  | 13        |
| 3.2 Arhitektura . . . . .                          | 14        |
| 3.3 Merilna enota . . . . .                        | 14        |
| 3.4 Zbirna enota . . . . .                         | 23        |
| 3.5 Strežnik . . . . .                             | 29        |
| 3.6 Poraba energije . . . . .                      | 29        |
| 3.7 Podatkovna analitika . . . . .                 | 36        |
| <b>4 Primerja uporabe</b>                          | <b>41</b> |
| 4.1 Kosilnica . . . . .                            | 41        |
| 4.2 Glasnost na fakulteti . . . . .                | 41        |

|  |           |
|--|-----------|
| <b>5 Zaključek</b>                                     | <b>47</b> |
| <b>Appendices</b>                                      | <b>49</b> |
| <b>A Dodatek: Uporabniška navodila</b>                 | <b>51</b> |
| A.1 Polnjenje merilnih enot . . . . .                  | 51        |
| A.2 Prižiganje enot v različnih načinih . . . . .      | 51        |
| A.3 Registracija merilne enote . . . . .               | 52        |
| A.4 Ustvarjanje raziskave . . . . .                    | 54        |
| A.5 Urejanje raziskave . . . . .                       | 55        |
| A.6 Prenašanje podatkov na enote . . . . .             | 56        |
| A.7 Postavljanje enot na lokacijo zaznavanja . . . . . | 56        |
| A.8 Uravnavanje intervala zaznavanja . . . . .         | 57        |
| A.9 Konec zaznavanja . . . . .                         | 57        |
| <b>Literatura</b>                                      | <b>61</b> |

# Povzetek

**Naslov:** Porazdeljeni sistem za pridobivanje, hranjenje in analizo podatkov o hrupu v urbanem okolju

**Avtor:** Andraž Novak

**Ključne besede:** hrup, internet stvari, zaznavanje, mesto, senzorji.



# Abstract

**Title:** Urban noise data sensing and collection with internet of things

**Author:** Andraž Novak

**Keywords:** noise, internet of things, sensing, city, sensors.



# Poglavlje 1

## Uvod

(vsaka črtica svoj odstavek)

- iot na splošno, kakšne probleme rešuje - opis našega problema, zahteve - obstoječe rešitve (tri) \* fotografije obstoječih merilnih enot \* tabela primerjav med sistemi - delo v nalogi

Internet stvari je tehnologija s katero lahko upravljamo in nadzorujemo svet okoli nas. Omogoča vse od povezanih pametnih luči, do domačega ogrevanja, do spremeljanja stanja pametnih mest. V domači uporabi se povezane naprave v veliki meri zanašajo na tehnologijo WiFi, ki napravam omogoča prenos podatkov do strežnika. V mestih in širši okolici pa se razširja uporaba tehnologij LoRa in 5G.

Moderno načrtovanje javnih površin vedno bolj upošteva vpliv hrupa na vedenje in zdravje ljudi. Na oboje lahko vplivamo s premišljeno zasnovno prostorov in izbiro primernih materialov. Pravilne odločitve lahko sprejemamo le s pomočjo primernih podatkov o trenutnem stanju in o uspešnosti takšnih odločitev v ostalih projektih. Vpliv teh odločitev mora biti merljiv. Zato potrebujemo robusten in fleksibilen sistem za merjenje in analizo glasnosti ter tipa hrupa. Tak sistem mora biti cenovno ugoden, prenosen in lahek za uporabo. Upravljanje s potekom raziskav o hrupu in analizi pridobljenih podatkov mora biti intuitivno in prijazno uporabnikom.

S problemom spremeljanja in analize hrupa se ukvarja več podjetij in raz-

iskovalcev, ki ponujajo različne rešitve. Podjetja Libelium, Cesva in Brüel Kjær ponujajo zaprtokodne profesionalne rešitve za trajno namestitev. Problem teh rešitev je, da so cenovno nedostopne in večinoma zahtevajo trajno namestitev na lokaciji.

Mogoče tabela?? idk def linki

<https://www.the-iot-marketplace.com/libelium-exm-lorawan-noise-analytics-solution> <https://www.iotsoundsensor.com/> <https://www.cesva.com/en/products/sensors-terminals/TA120/> <https://www.bksv.com/en/products/measuring-instruments/sound-level-meter>

obstoječe rešitve: -SONYC

- eni so isto k js nardil sam pred esp32 in MEMS mikrofon dobo hmmm

<https://www.ncbi.nlm.nih.gov/pmc/articles/PMC6165576/>

-?? -?

gremoooooo

V sklopu te diplomske naloge smo implementirali nizko cenovni porazdeljen sistem za zaznavanje in analizo hrupa na poljubno velikem območju. Zasnovali in sestavili smo cenovno ugodne meritve in zbirne enote. Implementirali smo sistem za upravljanje s temi enotami in razvili widget za podatkovno analizo pridobljenih podatkov v programu Orange. Izvedli smo tudi nekaj testov za prikaz primerov uporabe.



Slika 1.1: Material za štiri zbirne in 20 meritnih enot za potrebe projekta. Zadaj so ohišja za enote, levo spredaj so držala za baterije, na sredi so elektronski deli v antistatičnih vrečkah. Spredaj desno so USB kabli za napajanje in polnjenje.



# Poglavlje 2

## Tehnologije

(vsaka črtica tri odstavki, dodaj kakšno sliko, pazi na licence slik, vse mora biti v slovenščini, lahko tudi zrišeš kakšno shemo)

### 2.1 Mikrokontrolerji

### 2.2 Merjenje hrupa

Glavni del tega projekta je zaznavanje hrupa. Ta naloga je razdeljena na pridobivanje podatkov in obdelavo teh podatkov. Glavni zahtevi sta konsistentnost podatkov med napravami in točnost meritev. Kar se tiče zaznavanja podatkov smo tema dvema zahtevama lahko brez problemov ustregli s pravilno programsko kodo, pri zaznavanju pa rešitev ni bila trivialna.

Izbira pravilnega mikrofona je bila tako tu ključnega pomena. Večina mikrofonov deluje po principu, da zračni valovi premikajo del mikrofona, kar običajno spremeni kapacitivnost med kontaktoma mikrofona ta signal potem vezje ojača in logična enota to pretvori v digitalni signal. Ker so razlike v kapacitivnosti majhne, morajo biti vezja zelo občutljiva, kar pomeni da lahko majhna razlika v vezjih ali mikrofonu povzroči veliko razliko v rezultatih. Na srečo obstaja vrsta mikrofonov, ki odpravi in . Mikrofoni, ki temeljijo na tehnologiji mikroelektromehanskih sistemov poskrbijo za



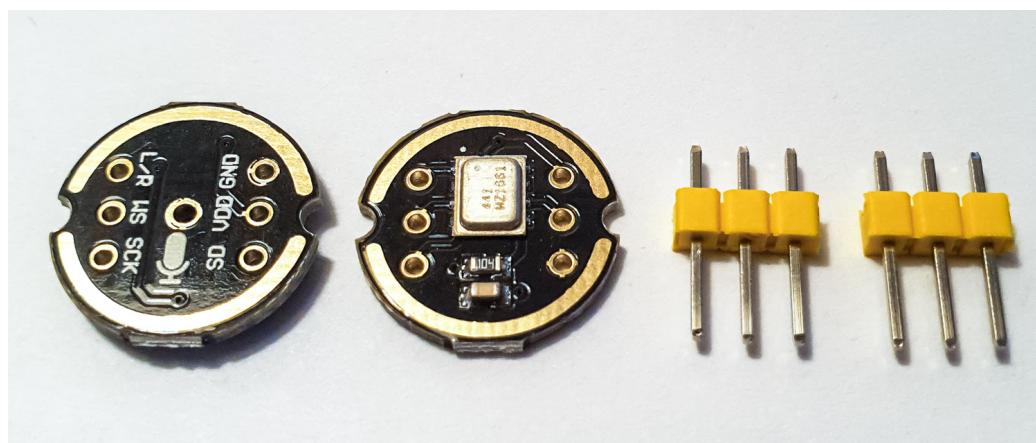
Slika 2.1: Razvojna ploščica TTGO T7 z mikrokontrolerjem ESP32. Uporabili smo jo v merilnih in zbirnih enotah. Izbrali smo jo zaradi majhne porabe energije v načinu spanja in priključkom za zunanjo anteno.

---

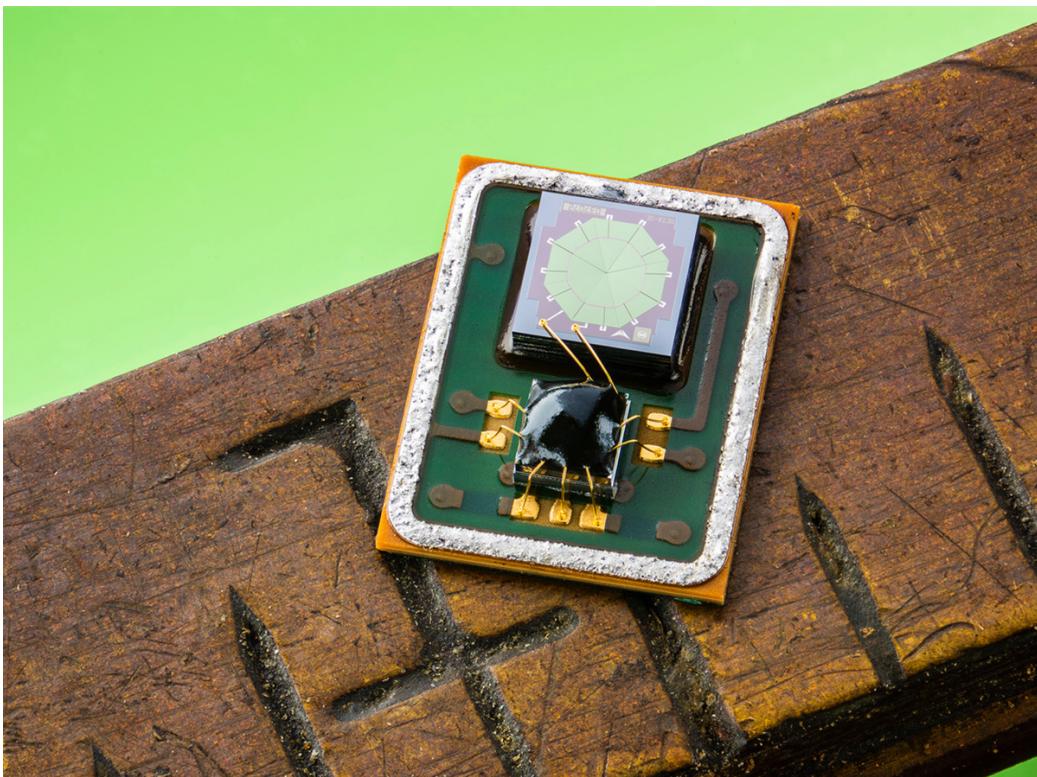
|                                       |  |
|---------------------------------------|--|
| Ime                                   | ESP32  |
| Proizvajalec                          | Espressif Systems                            |
| Procesor                              | Tensilica Xtensa LX6 dvojederni + koprocesor |
| SRAM                                  | 512 KiB                                      |
| Možnosti povezovanja                  | Wi-Fi 802.11 b/g/n, v4.2 BT + BLE            |
| Največja moč oddajanja                | 20 dB  |
| Periferne naprave                     | UART, I2C, I2S, SPI, CAN, ADC, PWM           |
| Napajalna napetost                    | 3.3V   |
| Poraba električne energije pri 240MHz | 55mA   |
| Poraba električne energije pri 20MHz  | 20mA   |
| Poraba električne energije v spanju   | 0.8mA  |
| Največja poraba                       | 350mA  |

---

Tabela 2.1: Značilnosti mikrokontrolerja ESP32.



Slika 2.2: Mikrofon INMP441. Montiran je na svoji ploščici za lažji dostop do kontaktov. Primeren je zaradi majhne porabe, majhne velikosti, natančnosti meritev in enostavnosti uporabe.



Slika 2.3: Notranjost mikrofona. Večja struktura je membrana iz silicija, ki zvočne valove pretvori v razlike v kapacitivnosti. Z dvema žicama je povezana s posebno enoto za pretvarjanje analognega signala v digitalnega in pošiljanje preko protokola I2S. Oboje je običajno prekrito s kovinskim ohišjem, ki je bilo za fotografijo odstranjeno.

---

|   |                |
|---|----------------|
| Ime                                       | INMP441        |
| Proizvajalec                              | InvenSense     |
| Protokol                                  | 24bit I2S      |
| Razpon frekvence vzorčenja                | 7.8kHz - 50kHz |
| Frekvenčni razpon                         | 60Hz - 15kHz   |
| Napajalna napetost                        | 3.3V           |
| Poraba elektrike v aktivnem stanju        | 2.2mA          |
| Poraba elektrike v stanju pripravljenosti | 0.8mA          |
| Poraba elektrike pri spanju               | 0.0045mA       |

---

Tabela 2.2: Značilnosti mikrofona INMP441.

## 2.3 Povezljivost

Pošiljanje podatkov je večdimenzionalni problem s katerim smo se ukvarjali precej časa. Tehnologija za povezovanje mora ustrezati strogim pogojem glede porabe energije, hitrosti prenosa, načina naslavljanja, dosega signala in cene enote. Preizkusili in raziskali smo precej možnosti tabela 2.3. In se na koncu odločili za protokol ESP-NOW, ki ga izbrani mikrokontroler podpira brez dodatnih naprav.

Prva in najbolj naivna raziskana možnost je bila protokol WiFi. Ta bi ustrezal našim zahtevam po hitrosti prenosa podatkov, ne bi pa bil primeren s stališča porabe energije, saj povezovanje na omrežje in vzpostavitev povezave s strežnikom trajata predolgo. Pod vprašajem bi bil tudi domet, sam je WiFi primeren za povezavo na nekaj deset metrih. Naslednji protokol, ki smo ga preučili je bil LoRa. Ta ustreza tako dometu signala, kot tudi porabi energije, ampak je zaradi prepočasnega prenosa podatkov neprimeren za pošiljanje meritev vsako sekundo. In z uporabo te tehnologije bi se cena meritve enote podvojila. Predzadnji na seznamu je NRF24. Neprimeren je zaradi omejitve šestih povezav naenkrat, nezanesljivega delovanja in premajhnega dosega povezave.

Zadnjo možnost smo odkrili, ko smo pregledovali dokumentacijo mikro-

kontrolerja ESP32. Proizvajalec teh čipov je namreč omogočil dokaj prosto pošiljanje paketov podatkov med ESP napravami. Protokol ESP-NOW ustreza vsem našim zahtevam. Hitrost podatkov do 250kBps, pošiljanje podatkov brez vzpostavitve povezave, kar drastično zmnjša porabo energije in več kot 1.5km dosega signala s poceni zunanjim anteno v perfektnih pogojih in do 200m v urbanem okolju.

|         | Doseg    | Hitrost prenosa | Poraba energije | Cena                    |
|---------|----------|-----------------|-----------------|-------------------------|
| ESP-NOW | 100m-1km | 250kbps         | 100mA           | 4€                      |
| NRF24   | 100m-1km | 250kbps         | 100mA           | MCU + 4€ za NRF24 modul |
| LoRa    | 1km-10km | 300bps-19kbps   | 30mA            | MCU + 10€ za LoRa modul |
| WiFi    | 10m-100m | 1mbps-150mbps   | 150mA           | 4€                      |

Tabela 2.3: Primerjava različnih tehnologij za povezovanje.



Slika 2.4: Prikaz testiranega dometa v urbanem okolju. Oddaljenost mejnih točk od zbirne točke je 250m, 157m in 270m. Testiranje na prostem je ob optimalnih pogojih pokazalo domet do 1500m.

## 2.4 Programska oprema za internet stvari

\* tabela prednosti in slabosti raziskani opcij

## 2.5 Strežniški del

NodeJS? tudi tu ne vem točno kaj in kako



# Poglavlje 3

## Rešitev

### 3.1 Uporabniške zahteve

Cilj tega projekta je bil implementacija sistema za podrobno razumevanje hrupa v poljubno velikih delih okolja. Zato smo se odločili za fleksibilen porazdeljen sistem za zaznavanje in analizo hrupa. Določili smo, da bo sistem omogočal prosto razporejanje meritnih enot, ki bodo sinhronizirano v specifičiranih časovnih intervalih zajemale podatke o hrupu in jih posredovale v podatkovno bazo, kjer bodo potem dostopne za analizo v programu Orange. Za analizo potrebujemo podatke o glasnosti in podatke o tipu hrupa, zato smo določili, da bomo zbirali podatke o jakosti hrupa in rezultate spektralne analize tega hrupa.

Velik poudarek smo pri tem projektu namenili cenovni učinkovitosti. Na tržišču obstaja veliko podobnih rešitev, ki po našem mnenju stanejo preveč. Zato je bila ena naših zahtev to, da zaledni sistem temelji na odprtokodnih okoljih in je nameščen na strežniku, ki je del laboratorija za bioinformatiko. Prav tako pa morajo biti meritne enote veliko bolj cenovno dostopne kot trenutno obstoječe podobne enote.

Večina obstoječih meritnih enot poleg velikega finančnega vložka zahteva tudi zunanje napajanje in trajno namestitev. Tako smo zanje določili, da morajo biti dovolj majhne in lahke, da z montiranjem ne bomo imeli pre-

velikih težav. V isti luči morajo imeti dovolj majhno porabo, da lahko vsaj en teden delujejo z napajanjem iz vgrajenih baterij in tako ne potrebujejo zunanjega napajanja.

Ker želimo zbrane podatke obdelati in prikazati, smo se odločili, da bomo omogočili dostop do podatkov v programu Orange. Tako smo se odločili implementirati widget, ki bo podpiral poizvedbe na zalednjem delu.

## 3.2 Arhitektura

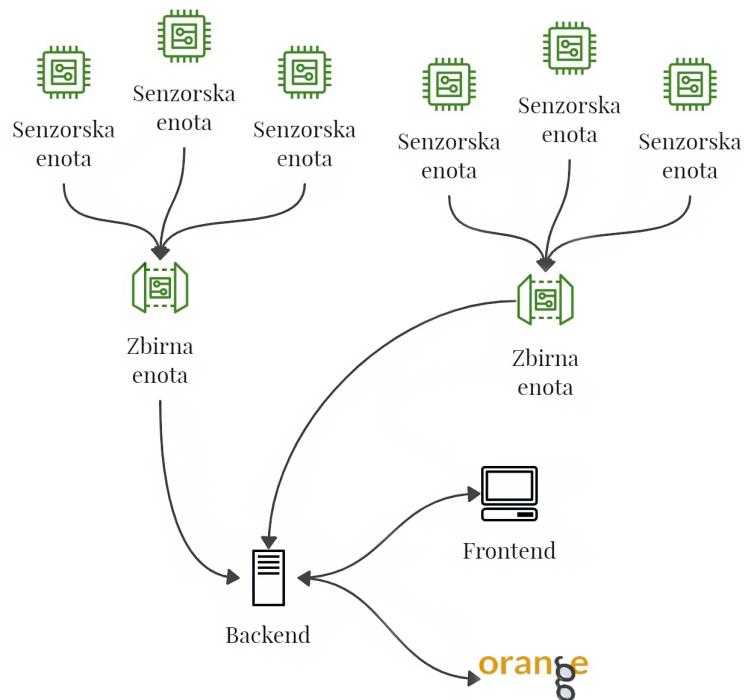
Arhitekturo sistema so določile omejitve izbranih tehnologij in uporabniške zahteve.

Osrčje sistema je zaledni del, ki koordinira delovanje zbirnih in merilnih enot, shranjuje podatke in streže podatke uporabniškemu vmesniku in programu Orange. @@@

## 3.3 Merilna enota

Merilna enota je zaključen del sistema, ki je odgovoren za zbiranje in pošiljanje podatkov na zbirno enoto. Zasnova enote temelji na uporabniških zahtevah in omejitvah tehnologije. Odločili smo se, da bomo uporabili prosto dostopne komponente, ki jih lahko z malo dela z žicami povežemo in montiramo v ohišje. To v primerjavi z oblikovanjem svojih tiskanih vezij zmanjša možnosti za napake in omogoča lažja popravila in menjavo delov enote, ko je ta že sestavljena.

Glavni del je mikrokontroler ESP32 na razvojni ploščici T7 podjetja TTGO. Za ta projekt je primerna zaradi majhne porabe v načinu spanja in priključku za zunanjo anteno. Nanjo se povezujejo vsi ostali deli merilne enote. Zaznavanje zvoka smo omogočili z MEMS mikrofonom INMP441, ki je s protokolom I2S povezan z mikrokontrolerjem. Ker je ta povezava povsem digitalna, smo eliminirali veliko problemov s kvaliteto žic in spojev. Za pomoč pri nastavljanju enote smo se odločili, da bo vsaka enota opremljena



Slika 3.1: Struktura razvitega sistema. Senzorske enote so na robu in komunicirajo z zbirnimi enotami. Nanjo pošiljajo podatke o hrupu, ki bodo preneseni na zaledni del. Zbirne enote prejemajo podatke z merilnih enot in jih posredujejo na zaledni del. Na zalednjem delu so shranjeni vsi podatki, do katerih lahko dostopa Orange widget in uporabniški vmesnik.

z OLED ekranom SSD1306 z resolucijo 32 krat 128 slikovnih točk. Z mikrokontrolerjem komunicira s protokolom I2C.

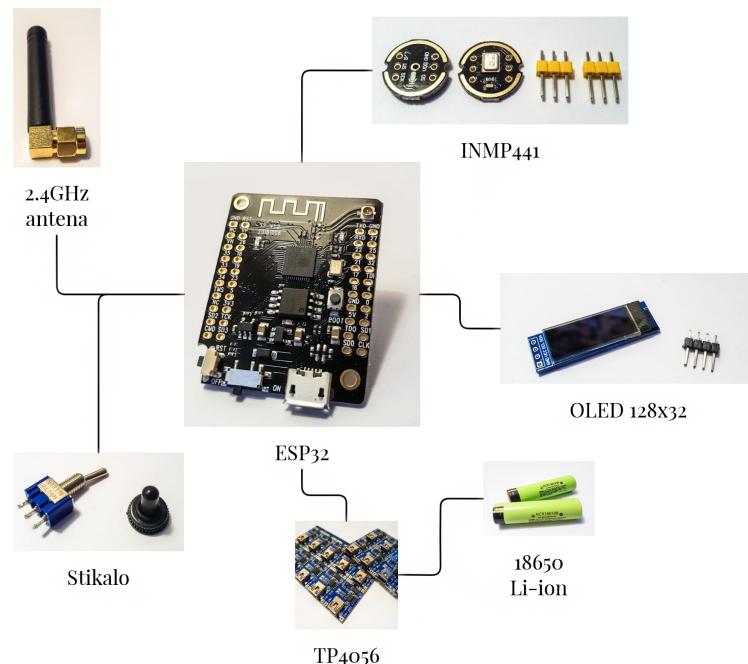
Za napajanje skrbita dve 18650 Li-ion polnilni bateriji, ki lahko ob polni napolnjenosti in zaznavanju hrupa vsako sekundo napajata enoto približno tri tedne. Za polnjenje in varno uporabo skrbi vezje TP4056, ki skrbi, da baterije niso prenapolnjene ali preveč prazne in da skozi baterijo ne steče previsok tok.

Na razvojno ploščico sta povezana še zunanjia antena za daljši doseg signala in stikalo, s katerim uporabnik prižge enoto in izbere način delovanja.

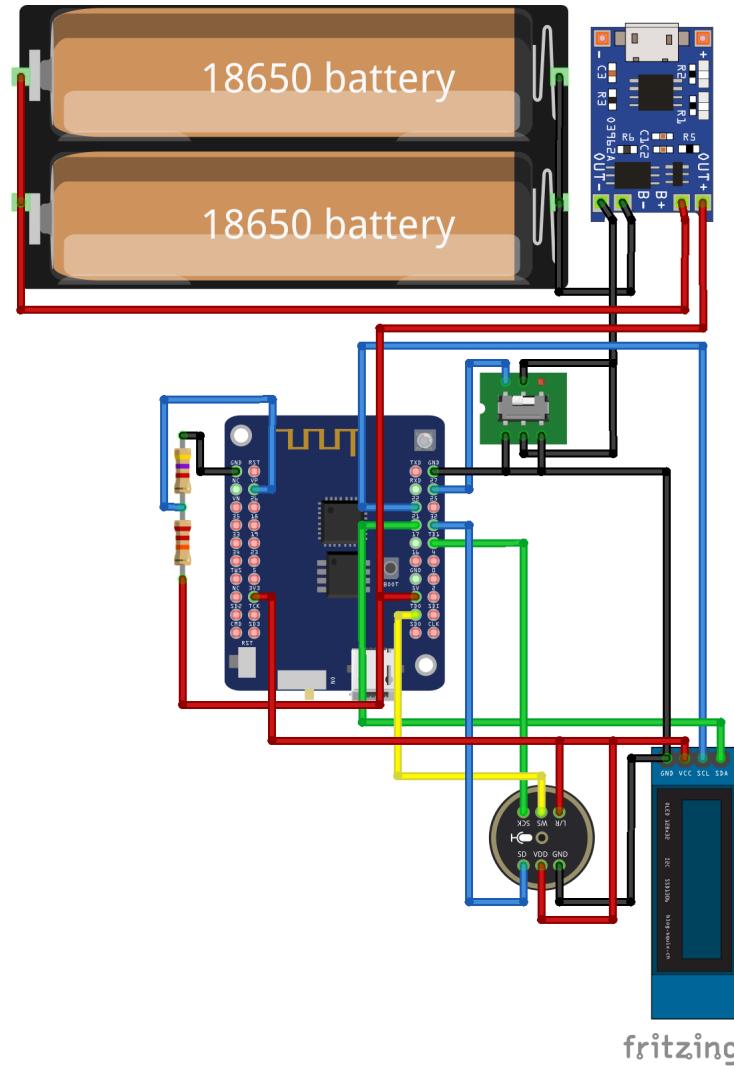
- sestava kaj je v njem in to - nastavljanje - načina delovanja - kako se spopadam s tem da mogoče ne bo mogu poslat I guess - nastavljanje Vse skupaj je montirano v plastično ohišje s tesnilom proti vdoru vode. Na zunani strani ohišja so montirani stikalo za upravljanje, zunanjia antena, ekran in pena za preprečevanja vdora vode, ki prekriva odprtino z mikrofonom.

Merilna enota podpira dva načina delovanja - način za nastavljanje in način za zaznavanje. Med njima lahko uporabnik preklaplja s stikalom, ki je montirano zunaj enote.

Način za nastavljanje (slika 3.4, leva stran) potrebujemo, ker je na vse enote naložena enaka programska koda, kar zmanjša zmedo pri nalaganju kode in olajša nastavljanje. Ko enota začne z delovanjem v načinu za nastavljanje, se poveže na določeno WiFi omrežje preko katerega komunicira z zalednim delom. Osnovna identifikacija poteka z uporabo MAC naslova, ki je v vsakem mikrokontrolerju unikaten. Ko zaledni del prejme naslov, v bazi podatkov ustvari nov zapis za merilno enoto, kamor ga shrani za prihodnje identifikacije, dodeli identifikacijsko številko za notranjo identifikacijo in enoti naključno dodeli ljudem razumljivo ime. V odgovoru na MAC naslov enoti odgovori z MAC naslovom zbirne enote, če je ta zbirna enota trenutno dodeljena študiji. Enota si naslov zbirne enote shrani v spomin za kasnejšo uporabo, na ekranu pa se izpišejo podatki o imenu, trenutni jakosti zvoka in napetosti na bateriji. Enota potem periodično na zaledni del pošilja podatke o napolnjenosti baterije in preverja, če je ta enota dodeljena drugi zbirni



Slika 3.2: Merilna enota z vrisanimi povezavami med sestavnimi deli. Osrčje vsake enote je mikrokontroler ESP32, ki sprejema podatke, jih obdeluje in pošilja na zbirne enote. Podatke zbira s pomočjo mikrofona INMP441. Za lažje nastavljanje se ob zagonu glavni podatki o enoti izpišejo na OLED ekranu. Enoto napajata dve 18650 Li-ion polnilni bateriji, za kateri skrbi vezje TP4056. Za daljši doseg signalov, smo uporabili zunanjou anteno.



Slika 3.3: Električne povezave v merilnih enotah. Barva povezav na sliki se čim bolj ujema z uporabljenimi kabli v sestavljenih enotah. Črna in rdeča barva sta vedno uporabljeni za napajanje, zelena je uporabljena za urni signal, modra pa za prenos podatkov. Rumena in bela pa se uporablja za druge namene.

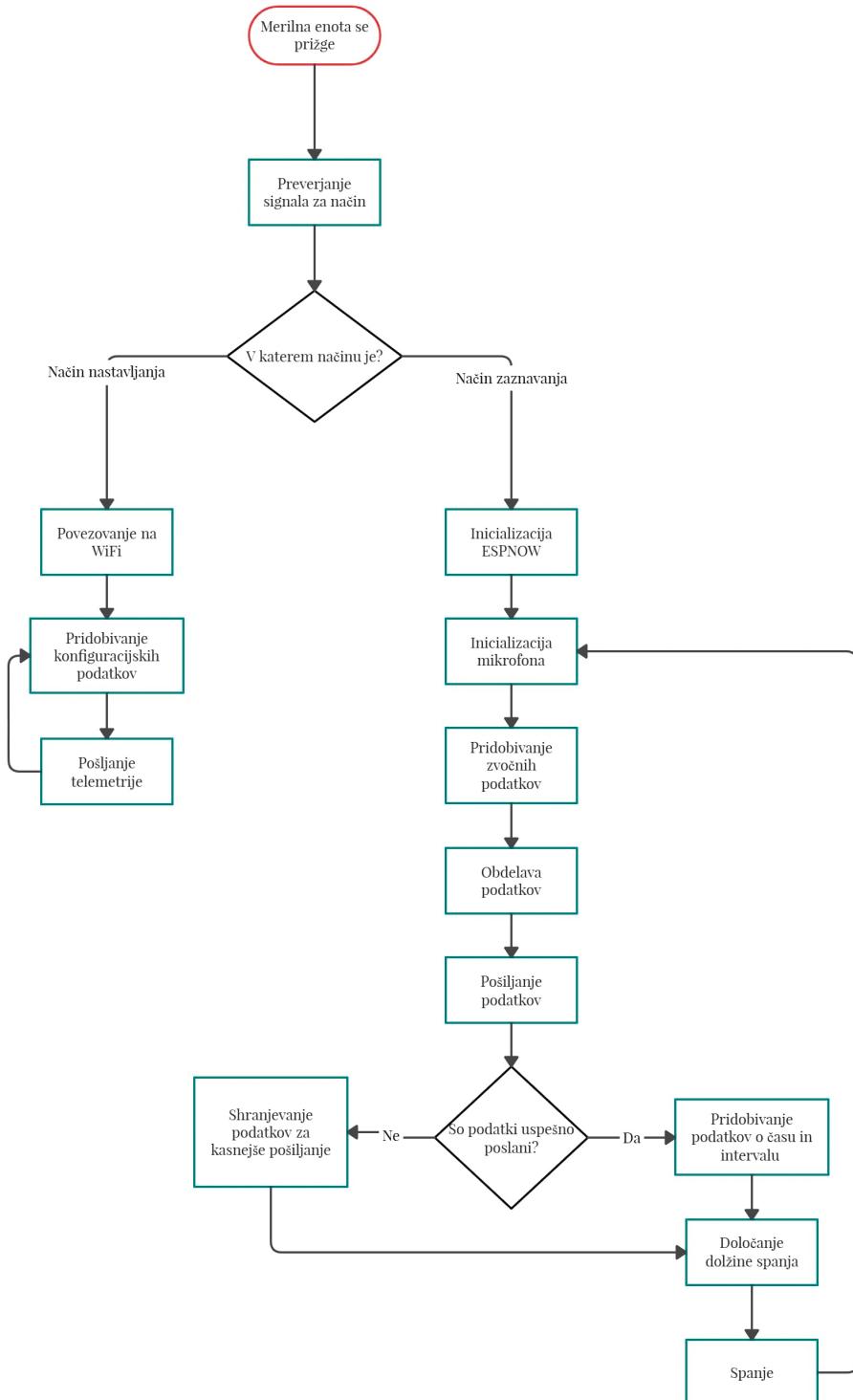
enoti.

Način za zaznavanje (slika 3.4, desna stran) je namenjen zbiranju in pošiljanju podatkov na zbirno enoto. Ko enota začne z delovanjem v načinu zaznavanja, se najprej inicializira brezžični vmesnik za komunikacijo s protokolom ESP-NOW. Temu sledi inicializacija mikrofona in zbiranje ter obdelava podatkov. Obdelani podatki se skupaj s časovno oznako zapisejo v sporočilo, ki se doda v vrsto. Sledi pošiljanje, ki smo ga implementirali paketno z naključnim intervalom med paketi sporočil. To pomaga pri zmanjševanju možnosti za trčenje paketov in za zmanjševanje porabe energije. Ko enota pošilja podatke, vzame sporočilo iz vrste sporočil in ga skuša poslati. Če je pošiljanje uspešno, je sporočilo zbrisano, če pošiljanje ni uspešno, se sporočilo ohrani v vrsti, nakar enota določi nov čas naslednjega pošiljanja. Na koncu določi čas spanja do naslednjega zaznavanja in začne spanje. Poleg tega enota vsake tri sekunde pošilja podatke o napolnjenosti baterije in zahteva podatke o točnem času in o trenutnem intervalu zaznavanja.

Programska koda, ki se izvede ob vsakem zajemanju in obdelavi podatkov. Mikrokontroler se najprej nastavi na najmanjšo frekvenco delovanja, saj med zajemanjem zvočnega signala večino časa ne dela nič računsko zahtevnega in samo čaka, da se medpomnilnik napolni. Sledi procesiranje podatkov, zato se nastavi na najhitrejši način delovanja, kjer na podatkih izvede spektralno analizo in izračuna jakost zvoka. Na koncu nastavi frekvenco na najnižjo in podatke prepiše v vrsto sporočil.

```
void sensing_and_data_preparation(){
    setCpuFrequencyMhz(20);
    // set cpu frequency to 20mhz to lower the consumption

    // get noise data
    sensing_start = get_secs();
    get_samples((int*)&samples_pub);
```



Slika 3.4: Merilna enota lahko deluje v dveh načinih. V načinu za nastavljanje enota pridobi podatke o delovanju z zaledja in pošilja svojo telemetrijo. V načinu zaznavanja enota zajame podatke o zvoku, jih obdelava in jih periodično pošilja na zbirno enoto.

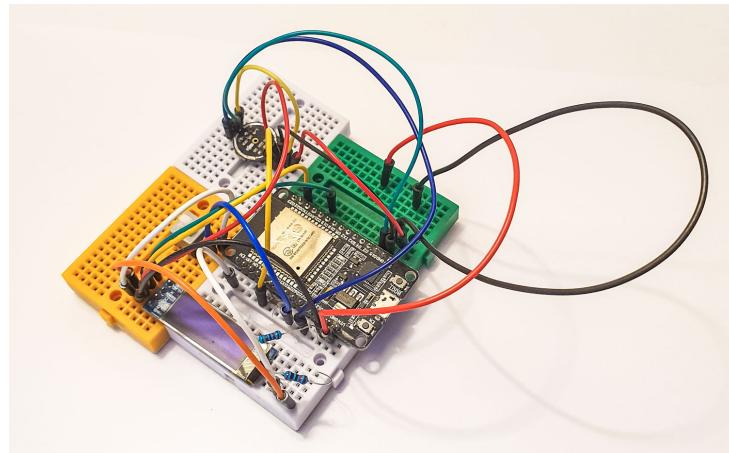
```
setCpuFrequencyMhz(240);
// set cpu frequency to 240mhz for processing

// process the data
calculate_fft((int*)&samples_pub, (double*)&fft_downsampled, DOWNSAMPLED_F
decibels = 0.0;
decibels = calculate_decibels((int*)&samples_pub, SAMPLES_SIZE);

setCpuFrequencyMhz(20);
// set cpu frequency to 20mhz to lower the consumption

// put data into a sending queue
add_to_sending_queue((double*) &fft_downsampled, decibels, sensing_start);
}
```

@@@



Slika 3.5: Prototip meritne enote na plošči. S prototipi smo si pomagali pri razvoju programske opreme, testiranju mikrofonov in merjenju porabe energije.



Slika 3.6: Material za meritve enote. Na sliki sta dve bateriji 18650 z držaloma, upravljalno vezje za baterije TP4056, mikrofon INMP441, stikalo za vklop in nastavljanje načina delovanja, prikazovalnik SSD1306, razvojna ploščica TTGO T7 z mikrokontrolerjem ESP32, antena in kabel za anteno, pena za zaščito mikrofona in ohišje v katerem je vse montirano. Manjkajo žice, ki povezujejo različne dele enote in vroče lepilo, ki smo ga uporabili za pritrjevanje v ohišje.



(a) Vidita se stikali za vklop in nastavljanje načina delovanja, druge strani. Vidi se pokrivalo ekran za prikaz osnovnih podatkov in antena.  
(b) Sestavljena merilna enota iz mikrofona namenjeno dušenju vetra in zaščito pred dežjem.

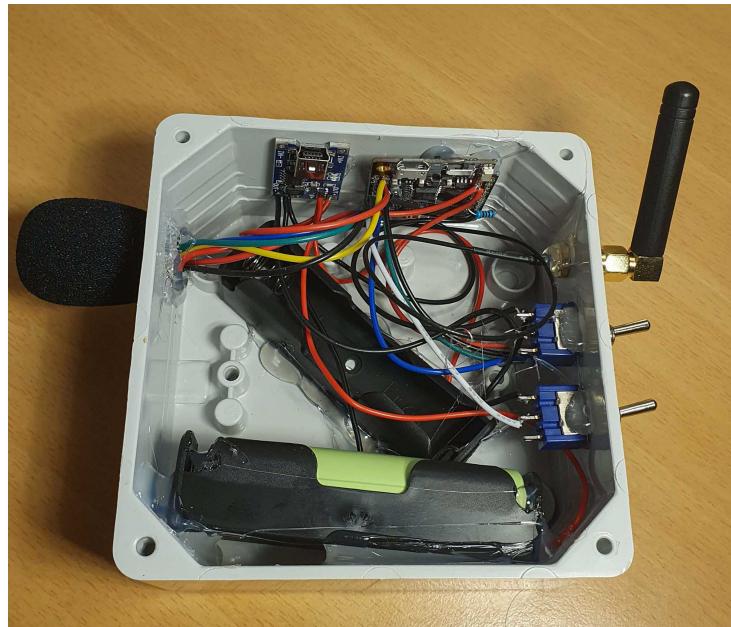
Slika 3.7: Sestavljena merilna enota iz dveh zornih kotov.

### 3.4 Zbirna enota

Zbirna enota (slika 3.9) je zaključen del sistema, ki je odgovoren za sprejemanje sporočil z merilnih enot, posredovanje podatkov na zaledni del in časovno usklajevanje delovanja merilnih enot. Zbirne enote potrebujemo, ker merilne enote v načinu zaznavanja ne morejo komunicirati z zalednim delom, ker nimajo dostopa do interneta. Zato smo razvili enoto, ki sprejme pakete, ki so poslani po protokolu ESP-NOW in jih preko WiFi povezave pošlje na zaledni del.

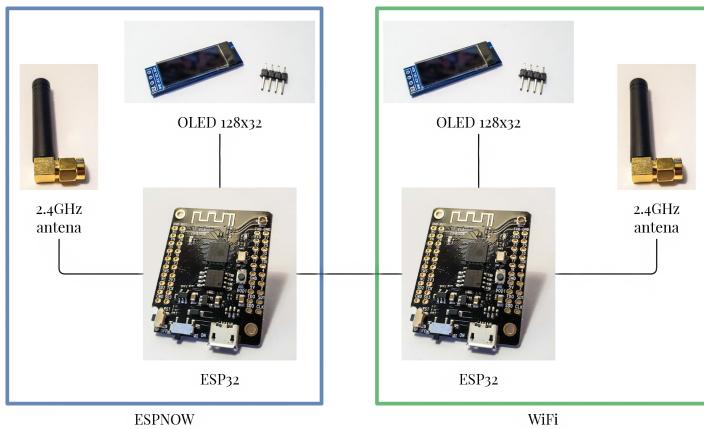
Osrčje te enote sta dva mikrokontrolerja ESP32 na razvojnih ploščah T7. Dva potrebujemo ker lahko en komunicira naenkrat samo z enim protokolom - torej eden lahko komunicira samo s protokolom ESP-NOW, drugi pa samo s protokolom WiFi. Podatke si izmenjujeta po serijski UART povezavi s protokolom PJON. Ta skrbi za zanesljivo prenašanje sporočil in naslavljjanje.

Vsaka zbirna enota je opremljena z dvema OLED prikazovalnikoma, ki uporabniku sporočata uporabne podatke - količina preostalega ram pomnilnika, povprečen povratni čas do zalednega dela, koliko sporočil iz merilnih enot je bilo sprejetih v zadnji sekundi, moč WiFi signala, MAC naslov te



Slika 3.8: Notranjost sestavljenih merilnih enot. Vidijo se povezave med mikrofonom, ekranom, polnilnim vezjem, stikali in TTGO T7. Vse povezave so barvno koordinirane. Povezave za napajanje so rdeče in črne barve, povezave po katerih se prenašajo podatki so modre barve, povezave za prenašanje urinega signala pa so zelene barve. Luknje so zatesnjene z vročim lepilom, za preprečevanje vdora vode.

enote, dodeljeno ime enote in število sekund delovanja brez napak.



Slika 3.9: Koncept zbirne enote. Razdeljena je na del, ki komunicira s protokolom WiFi, in na del, ki komunicira s protokolom ESP-NOW. Povezana sta z UART serijsko povezavo preko katere komunicirata s protokolom PJON. Takšna zasnova je potrebna, ker ESP32 ne more istočasno komunicirati preko obeh protokolov.

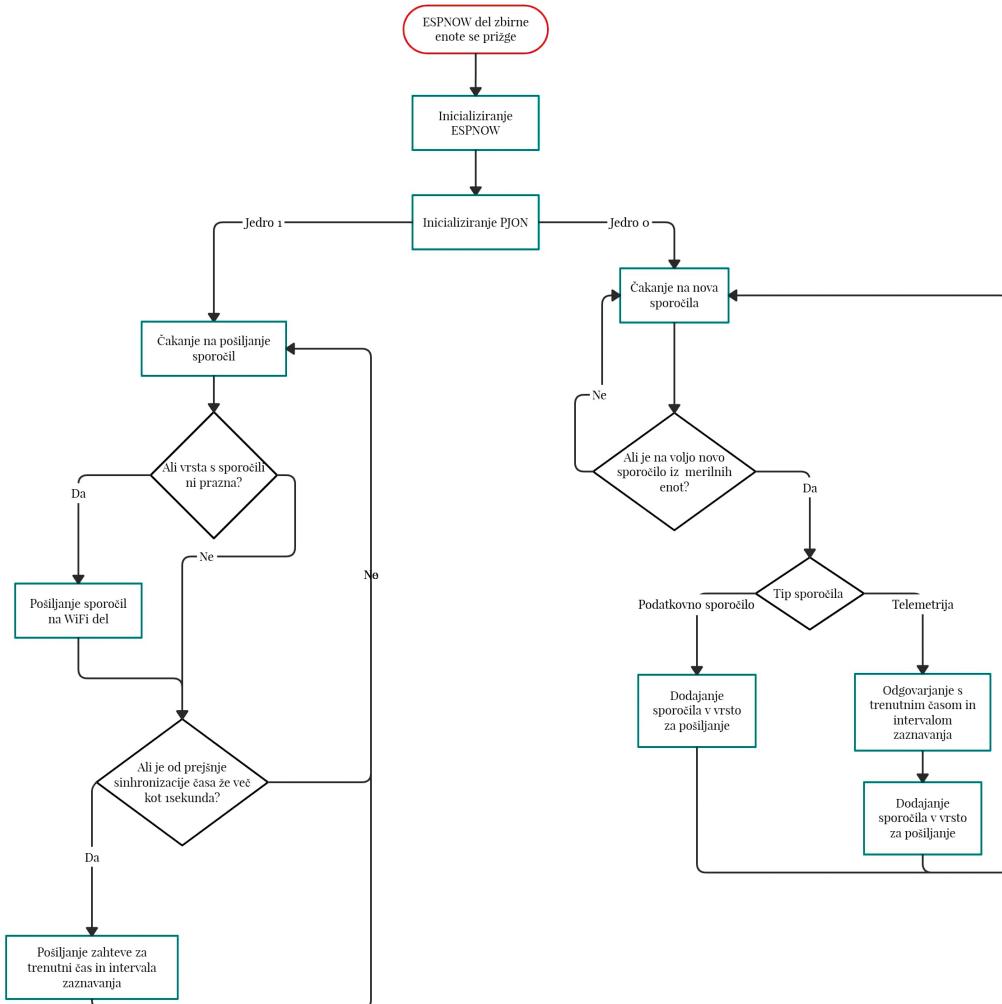
Posebnost ESP32 je to, da ima dvojederni procesor. V primeru merilne enote je to zaradi večje porabe hiba, pri zbirni enoti pa smo to uporabili v našo korist, saj smo lahko pri vsakem mikrokontrolerju eno celo jedro namenili komunikaciji z drugim delom enote, drugo jedro pa komunikaciji s protokolom ESP-NOW ali s protokolom WiFi. Za vsako aktivnost smo napisali svoj proces in ga dodelili svojemu jedru. Pri tem smo si pomagali z operacijskim sistemom FreeRTOS. Za komunikacijo med deloma enote smo definirali posebna sporočila, ki so namenjena prenosu podatkov o meritvah, zahtevah za sinhronizacijo časa, sinhronizaciji časa in pošiljanju telemetričnih podatkov. S tem smo ukrotili asinhrono delovanje mikrokontrolerjev in ustvarili iluzijo, da je zbirna enota ena homogena celota.

Del zbirne enote, ki komunicira s protokolom ESP-NOW (slika 3.10) z merilnimi enotami je zadolžen za sprejem podatkov, sinhronizacijo časa in sprejem telemetričnih podatkov. Ob zagonu najprej inicializira komunikacijo s protokoloma ESP-NOW in PJON, potem pa vsakemu jedru dodeli svoj

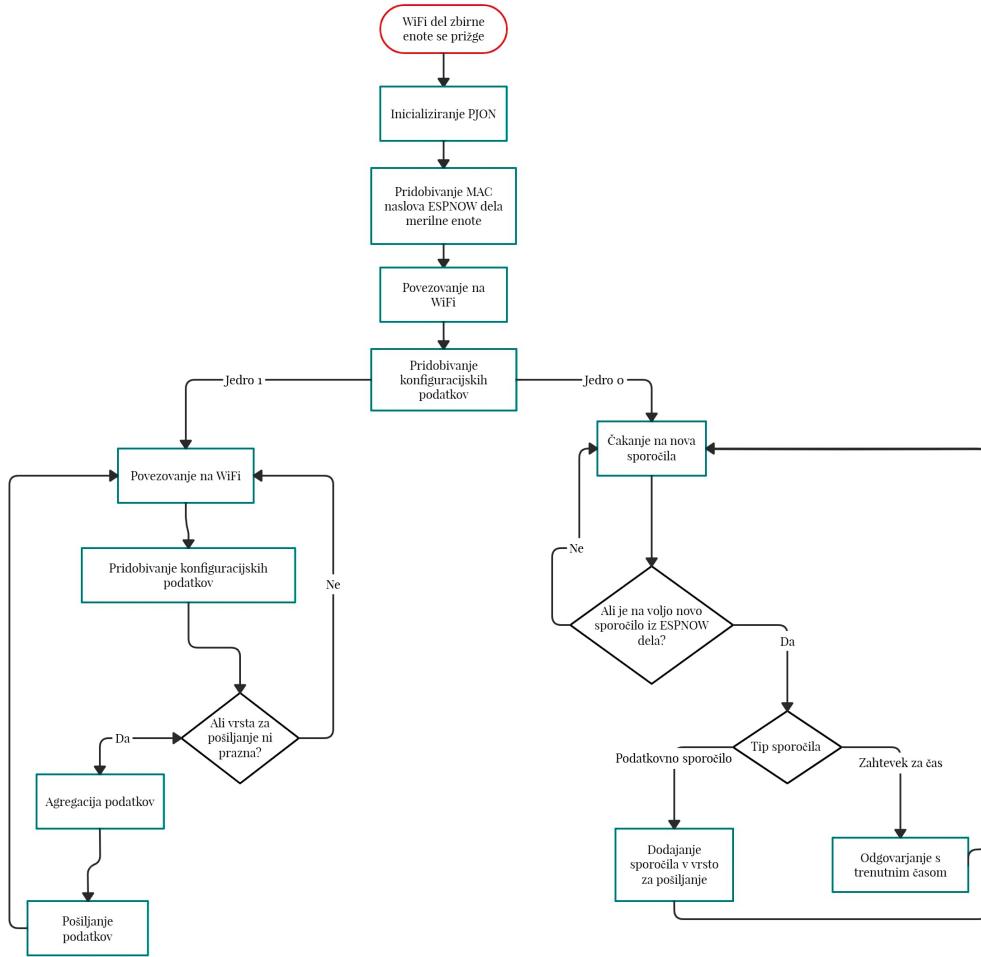
proces. Jedro 0 potem čaka na sporočila z meritnih enot. Ko sporočilo prejme, najprej določi tip sporočila. Če sporočilo vsebuje podatke o meritvi, jih prepiše v vrsto za pošiljanje na WiFi del. V nasprotnem primeru pa gre za sporočilo s telemetričnimi podatki na katere odgovori s trenutnim časom v milisekundah, telemetrične podatke pa prav tako zapiše v vrsto za pošiljanje na WiFi del. Jedro 1 je zadolženo za prenašanje podatkov na WiFi del enote in na sinhronizacijo časa. Proses vsako sekundo s posebnim sporočilom zahteva podatke o trenutnem času, hkrati pa preverja, če je v vrsti za pošiljanje novo sporočilo. V primeru da je, to sporočilo prenese na drug del enote.

Del zbirne enote, ki komunicira s protokolom WiFi (slika 3.11) z zalednim delom, je zadolžen za registracijo enote, sinhronizacijo časa in prenos podatkov o meritvah in o telemetriji na zaledni del. Ob zagonu najprej inicializira komunikacijo s protokoloma WiFi in PJON in pošlje MAC naslov enote na zaledni del za identifikacijo, nato vsakemu jedru dodeli svoj proces. Jedro 0 čaka na sporočila. Če je sporočilo podatkovno, ga doda v vrsto za pošiljanje na zaledni del, v nasprotnem primeru odgovori s podatki o trenutnem času. Jedro 1 najprej inicializira časovno sinhronizacijo s protokolom NTP, nato z zalednega dela pridobi konfiguracijske podatke, končno dodana sporočila iz vrste za pošiljanje agregira glede na to ali sporočilo vsebuje podatke o meritvi ali telemetrijo in vsake posebej paketno pošlje na zaledni del preko WiFi povezave.

Enoto smo sestavili iz več manjših komponent, ki smo jih z v naprej narezanimi žicami električno povezali. Celemu projektu smo določili enotno barvno shemo žic in njihovih namenov (slika 3.12). Pri napajanju smo se držali standardne črne in rdeče barve, pri signalih pa smo se načeloma držali principa, da je zelena barva namenjena urinemu signalu, modra pa podatkovnemu signalu. Vse električne komponente (slika 3.13) smo namestili v plastično ABS ohišje z zaščito proti vodi in drugimi tujki. Vse zvrtane luknje smo zatesnili z vročim silikonskim lepilom (slika 3.15), da smo preprečili vdor vode. Na zunanjost enote (slika 3.14), smo montirali antene, prikazo-

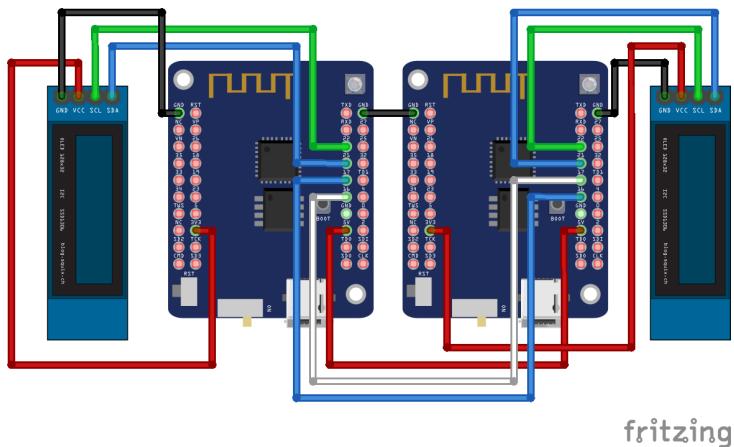


Slika 3.10: Pотek delovanja dela zbirne enote, ki komunicira z merilnimi enotami. Ker ima ESP32 dvojedrni procesor, se potek programa razdeli v dva dela, ki se izvajata istočasno. Jedro 0 je zadolženo za brezžično komunikacijo z merilnimi enotami preko protokola ESP-NOW, jedro 1 pa je zadolženo za komunikacijo z drugim delom zbirne enote s protokolom PJON.



Slika 3.11: Potek delovanja dela zbirne enote, ki komunicira z zalednim delom. Ker ima ESP32 dvojedrni procesor, se potek programa razdzieli v dva dela, ki se izvajata istočasno. Jedro 0 je zadolženo za komunikacijo z delom zbirne enote, ki iz merilnih enot prejema podatke o hrupu. Jedro 1 je zadolženo za komunikacijo z zalednim delom, kamor posreduje podatke iz merilnih enot.

valnike in trimetrski USB kabel za napajanje. Ker mora enota preprečevati vdor dežja, je kabel trajno z vročim lepilom pritrjen v ohišje.



Slika 3.12: Električne povezave v zbirni enoti. Barve povezav se ujemajo s celim projektom. Rdeča in črna sta namenjeni napajanju, modra je namenjena prenosu podatkov, zelena pa urnemu signalu. Posebnost je UART povezava med enotama, ki je tu prikazana z modro in belo povezavo.

## 3.5 Strežnik

(konfigurator, API)

\* shema podatkov v podatkovni bazi

## 3.6 Poraba energije

Ena od glavnih zahtev za ta projekt je bila možnost napajanja meritnih enot z baterijami. To je predpogoj za fleksibilnost in prosto postavljanje enot na lokacijo merjenja. Cilj je bil vsaj en teden delovanja z baterijskim napajanjem. Ta cilj smo presegli s teoretičnimi vsaj tremi tedni napajanja iz dveh baterij če je interval med zaznavanji dolg eno sekkundo.

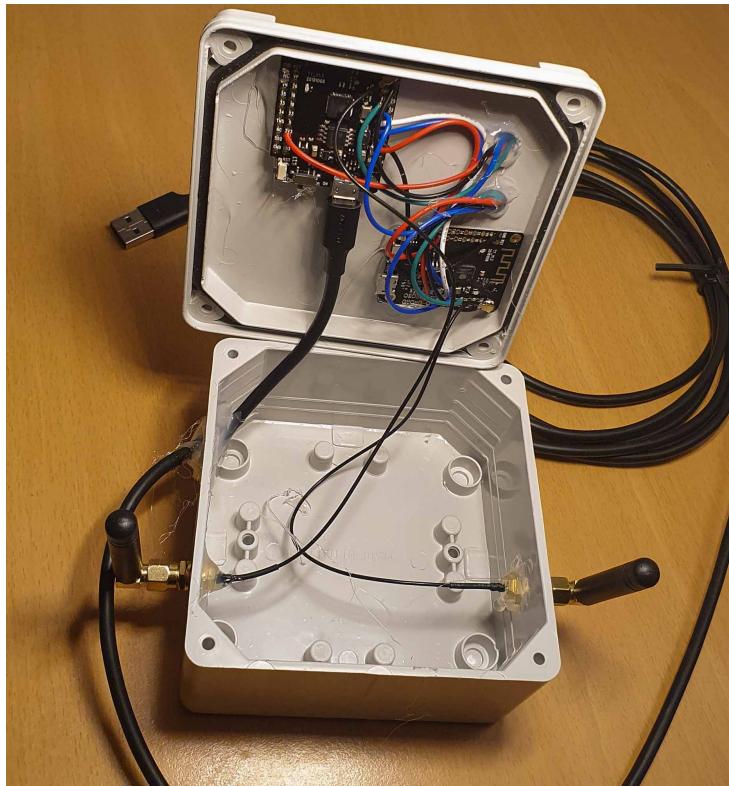
Ker ima povprečna 18650 Li-ion polnilna baterija kapaciteto 3000mAh, bi morala poraba biti nižja od 17mA. Električna poraba mikrokontrolerja



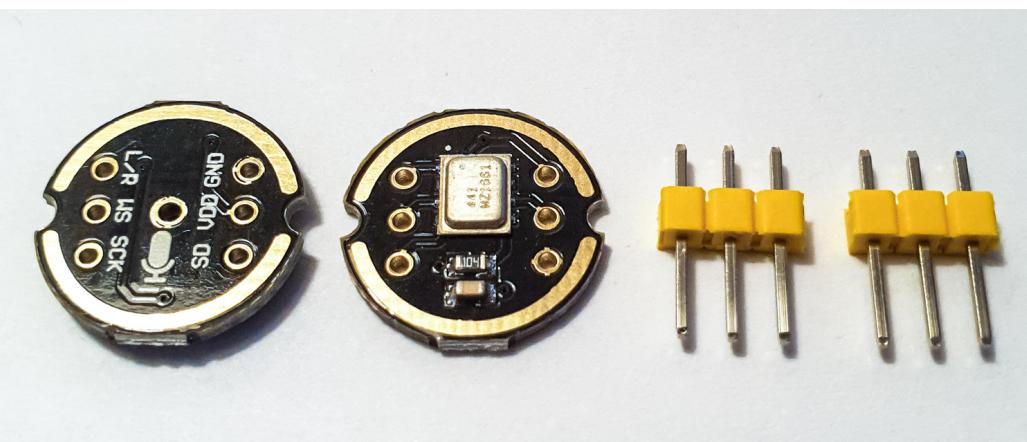
Slika 3.13: Material za zbirno enoto. Uporabili smo dve razvojni ploščici TTGO T7, dva prikazovalnika SSD1306, dve anteni s kabli in ohišje. Manjkajo žice, ki povezujejo sestavne dele, vroče lepilo, s katerim smo zatesnili luknje v ohišju in pritrdili električne dele v ohišje, in napajalni kabel, ki mora biti med delovanjem priklopiljen v USB polnilnik.



Slika 3.14: Zbirna enota. Na pokrovu sta montirana dva prikazovalnika, pri straneh sta dve anteni in na eni strani je napajalni USB kabel.



Slika 3.15: Notranjost zbirne enote. Razvidna je uporaba enotne barvne sheme za električne povezave in uporaba vročega lepila za zatesnitev luknj in pritrjevanje električnih komponent. Napajalni kabel je zaradi potrebe po tesnjenju trajno pritrjen v ohišje.



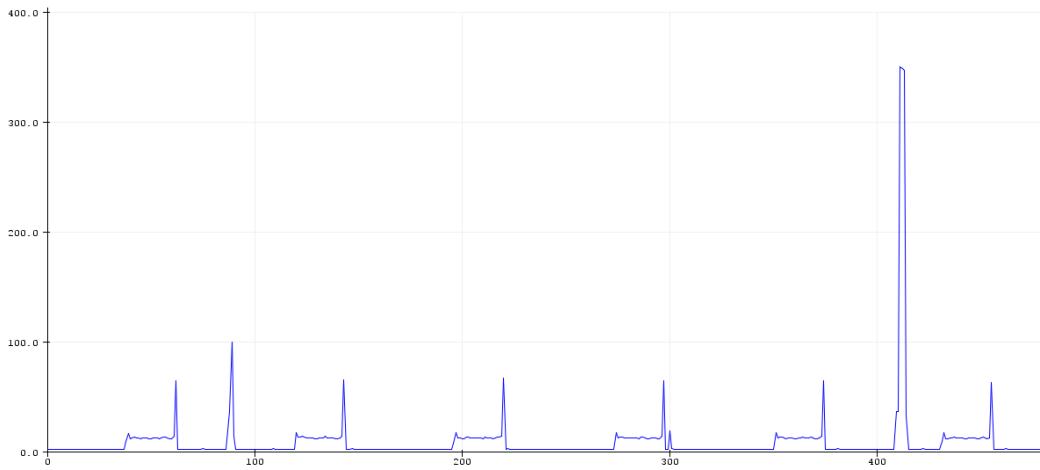
Slika 3.16: Caption

ESP32 je 55mA pri polni hitrosti obratovanja in povprečno 100mA pri pošiljanju podatkov. Brez optimizacij je poraba vsaj trikrat prevelika za naše potrebe.  
@@@

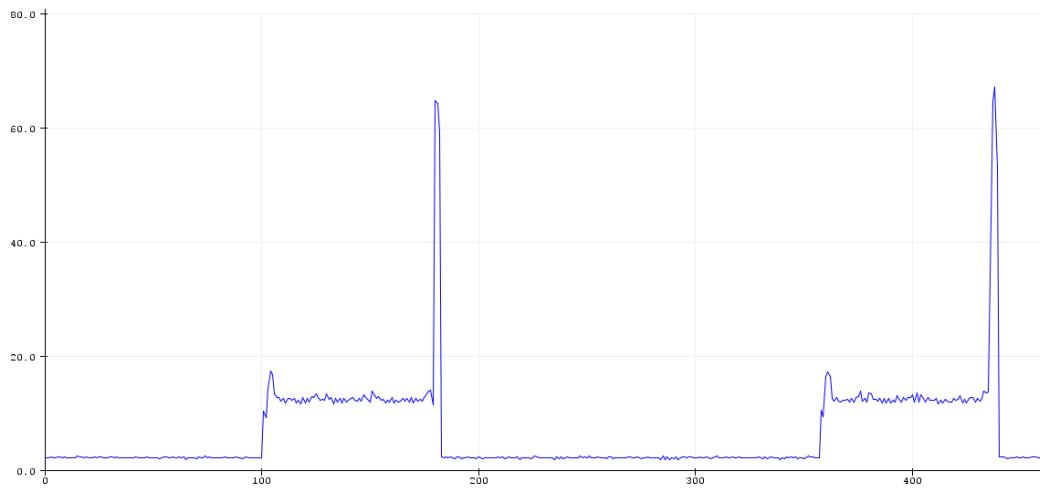
Prvi koncept, ki smo ga implementirali je spanje. Celotno zajemanje zvoka, procesiranje in pošiljanje traja manj kot eno tretjino sekunde, tako da lahko ostali dve tretjini mikrokontroler spi. Ta mikrokontroler podpira dve vrsti spanja - globoko spanje in dremež. V načinu globokega spanja enota porabi le nekaj mikroamperov, ampak se ob spanju delovni pomnilnik izbriše in se enota efektivno gledano ponovno zažene. Čeprav bi bila majhna poraba med spanjem zelo priročna, se mikrokontroler iz globokega spanja zbuja 250ms pri polni hitrosti delovanja. V načinu dremeža porabi približno en miliamper, ampak se vsebina delovnega pomnilnika ohrani in se izvajanje programa po spanju takoj nadaljuje. Zaradi tega je dremež v tem primeru veliko bolj primeren način spanja. S tem smo porabo zmanjšali na približno 20mA. Drugi koncept, ki smo ga implementirali je dinamično nastavljanje frekvence delovanja mikrokontrolerja. Izbiramo lahko med 240MHz, 160MHz, 80MHz, 40MHz, 20MHz in 10MHz. Počasnejše delovanje procesorja pomeni tudi daljše izvajanje iste kode. Tega načeloma nočemo, ker je najbolje da je čim več časa v načinu spanja in ker se poraba s frekvenco delovanja ne veča linearно, ampak v korist višjim frekvencam. Torej lahko frekvenco procesorja zmanjšamo na delih, ki niso računsko zahtevni. Na srečo je zbiranje podatkov iz mikrofona, ki predstavlja najdaljši del aktivnosti tudi najmanj računsko zahteven, zato je frekvanca med zbiranjem podatkov nastavljena na 20MHz.

```
// do indefinitely
while (true) {

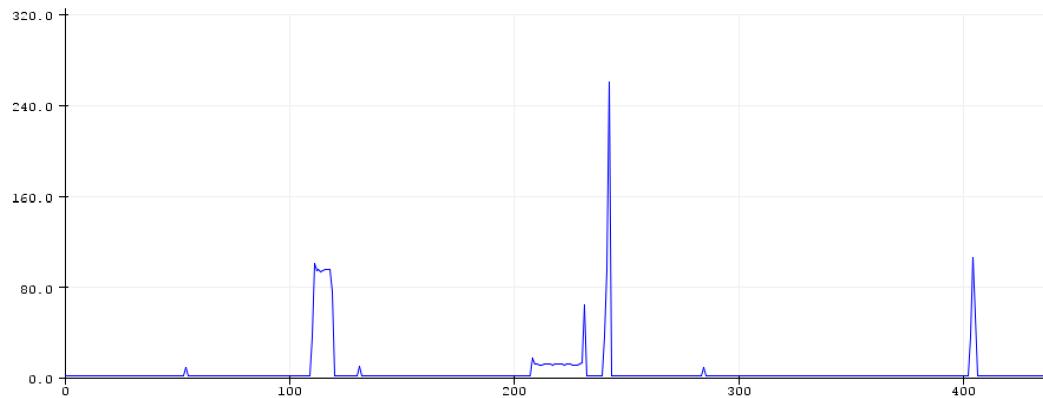
    if(is_interval_now()){
        sensing_and_data_preparation();
    }
}
```



Slika 3.17: Na grafu porabe se jasno vidijo zaznavanja zvoka vsako sekundo in pošiljanja podatkov. Manjši vrh se ujema s pridobivanjem podatkov o času in intervalu, večji vrh pa s pošiljanjem podatkov. Manjši vrhovi se ujemajo s porabo energije med zbiranjem in obdelavo podatkov.



Slika 3.18: Na grafu porabe so jasno razvidni različni deli običajnega zaznavnega cikla. Enota najprej približno 650ms spi, kar se ujema z zelo nizko porabo. Sledi pridobivanje podatkov o zvoku, ki traja približno 250ms in ima zaradi nizke frekvence procesorja majhno porabo. Na koncu sledi obdelava podatkov, ki se zgodi pri največji hitrosti procesorja, in se ujema z vrhom, ki traja le nekaj milisekund.



Slika 3.19: Na grafu porabe se vidi princip delovanja, ko je interval zaznavanja daljši od ene sekunde. Enota se eno sekundo po začetku prejšnjega merjenja zbudi in preveri, ali je trenutno število sekund deljivo z intervalom merjenja. Ker ni, gre nazaj v način spanja. Nekaj milisekund kasneje, se enota spet zbudi in spet preveri če je čas za pošiljanje ali usklajevanje časa.

| Interval zaznavanja (s) | Električni tok (mA) |
|-------------------------|---------------------|
| 1                       | 8.35                |
| 2                       | 5.65                |
| 3                       | 4.65                |
| 4                       | 4.14                |
| 5                       | 3.80                |
| 6                       | 3.60                |
| 7                       | 3.50                |
| 8                       | 3.35                |
| 9                       | 3.25                |
| 10                      | 3.15                |
| 1000                    | 2.61                |

Tabela 3.1: Povprečne porabe ob različnih intervalih merjenja. Tok smo merili na sestavljeni enoti. Uporabili smo čip INA219 za merjenje napetosti. V porabo je vključena poraba mikrokontrolerja, ekrana in mikrofona.

```
// sleep for a random amount of time to prevent signal congestion
setCpuFrequencyMhz(20);

int random_sleep = (int)get_random_sleep_time();
if (random_sleep > 0) {
    esp_sleep_enable_timer_wakeup(random_sleep);
    esp_light_sleep_start();
}

// set cpu frequency to 80mhz and send
setCpuFrequencyMhz(80);
sending_and_telemetry();

// calculate time till next second and enter light sleep
setCpuFrequencyMhz(10);
long left = 0;
left = get_remaining_sleep_time();
if (left > 0) {
    esp_sleep_enable_timer_wakeup(left);
    esp_light_sleep_start();
}
}
```

### 3.7 Podatkovna analitika

\* posnetek zaslona widjeta

```

// do indefinitely
while (true) {

    if(is_interval_now()){
        sensing_and_data_preparation();
    }

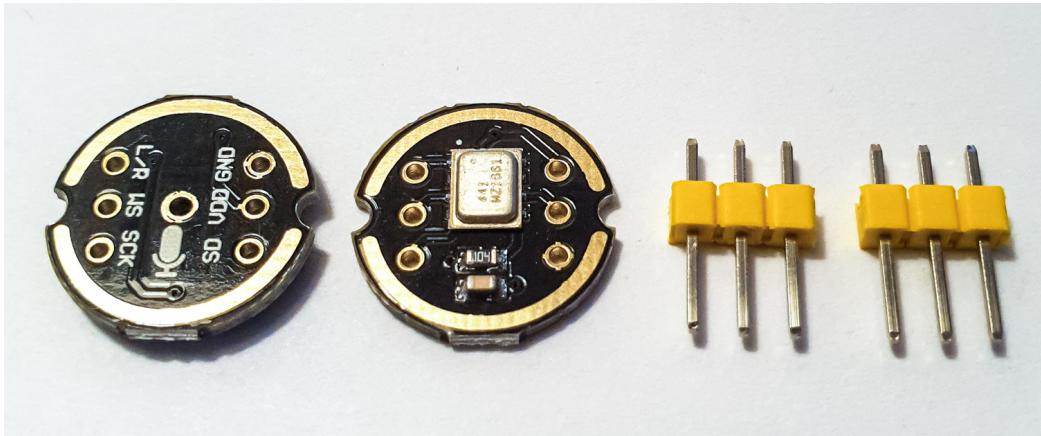
    // sleep for a random amount of time to prevent signal congestion
    setCpuFrequencyMhz(20);
    int random_sleep = (int)get_random_sleep_time();
    if (random_sleep > 0) {
        esp_sleep_enable_timer_wakeup(random_sleep);
        esp_light_sleep_start();
    }

    // set cpu frequency to 80mhz and send
    setCpuFrequencyMhz(80);
    sending_and_telemetry();

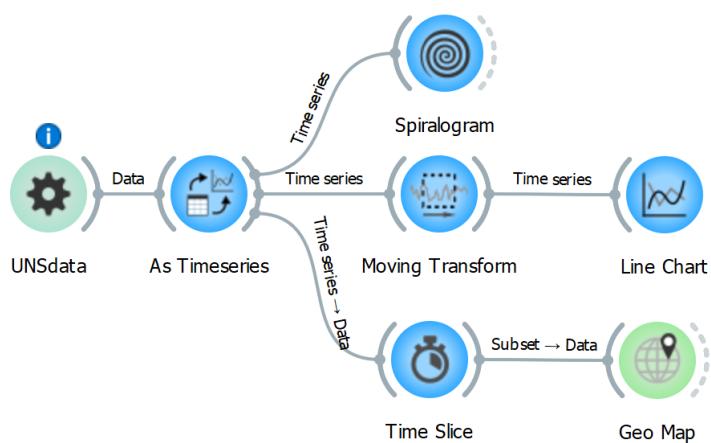
    // calculate time till next second and enter light sleep
    setCpuFrequencyMhz(10);
    long left = 0;
    left = get_remaining_sleep_time();
    if (left > 0) {
        esp_sleep_enable_timer_wakeup(left);
        esp_light_sleep_start();
    }
}

```

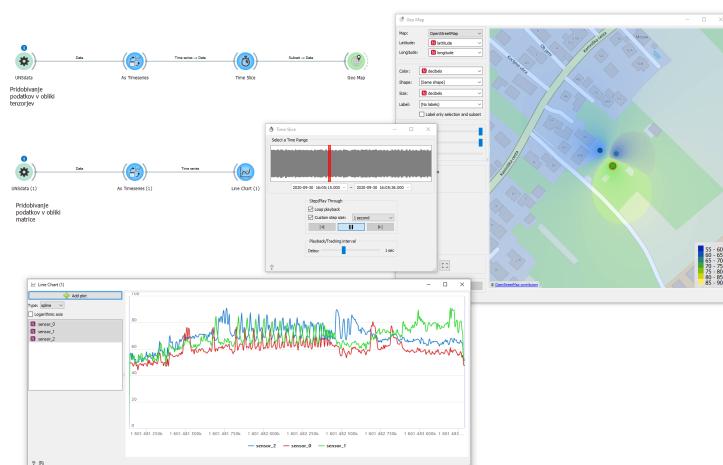
Listing 1: Programska koda, ki koordinira celotno delovanje meritne enote v načinu merjenja. Posebno pozornost smo namenili nastavljanju hitrosti delovanja procesorja glede na to kaj se v določenem trenutku dogaja. Ob zbiranju podatkov smo uporabili frekvenco 20 MHz, ker procesor večino časa čaka na nove podatke. Procesiranje podatkov dela s frekvenco 240 MHz, za pošiljanje ESP32 potrebuje frekvenco vsaj 80 MHz.



Slika 3.20: Caption



Slika 3.21: Primer workflowa v programu Orange. Podatke pridobljene s pomočjo UNSdata widgeta, se najprej pretvori v časovno zaporedne podatke, potem se lahko periodičnost podatkov prikaže na spiralnem prikazu. Podatke se lahko zgladi in osnovno obdela s funkcijo "Moving Transform" in se jih prikaže na grafu, ali pa se izbere rezino podatkov, ki so potem prikazani na zemljevidu.



Slika 3.22: Primer workflowa v programu Orange na primeru kosilnice.



# Poglavlje 4

## Primera uporabe

(motivacija, postavitev, rezultati, interpretacija - štirje odstavki)

### 4.1 Kosilnica

Iz opisa slike in kot osnutek za dejansko besedilo:

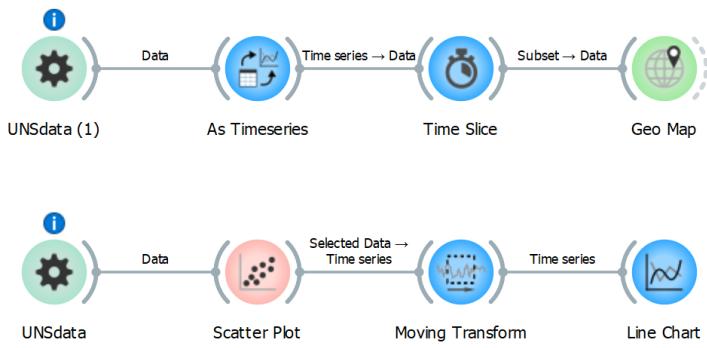
Pri primeru kosilnice smo raziskovali kako natančno lahko določimo gibanje izvira hrupa glede na izmerjeno glasnost na vsakem senzorju. Podatke smo za prikaz na mapi izvozili kot tenzorje in jih potem s pomočjo geografskih orodij prikazali na zemljevidu dejanske lokacije. Za prikaz grafa po času smo podatke izvozili kot matrico in jih tako prikazali na grafu.

(motivacija, postavitev, rezultati, interpretacija - štirje odstavki)

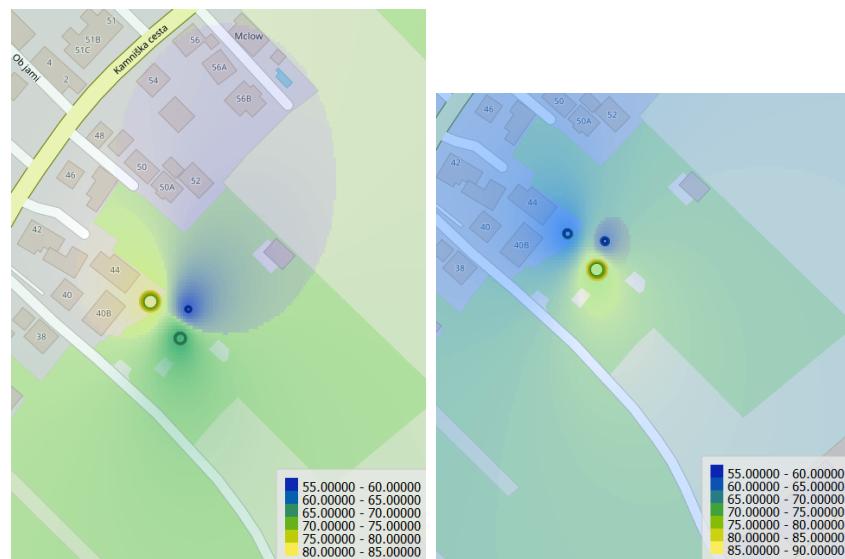
S tem poskusom smo želeli opazovati možnost spremeljanja lokacije vira hrupa. Merilne enote smo pred košenjem postavili ob rob travnika, da se jasno vidijo oscilacije hrupa, ki ga je proizvedla kosilnica. Jasno se vidi, kdaj je bila kosilnica najbližje posamezni merilni enoti.

### 4.2 Glasnost na fakulteti

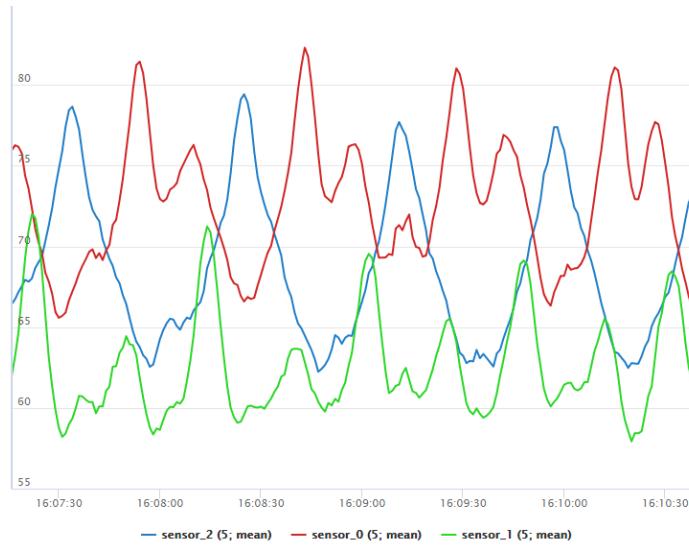
\* posnetek zaslona zemljevida z lokacijami merilnih enot \* posnetek zaslona workflowa v Orange \* zglajen graf po meritvah \* spiralogram za prikaz peri-



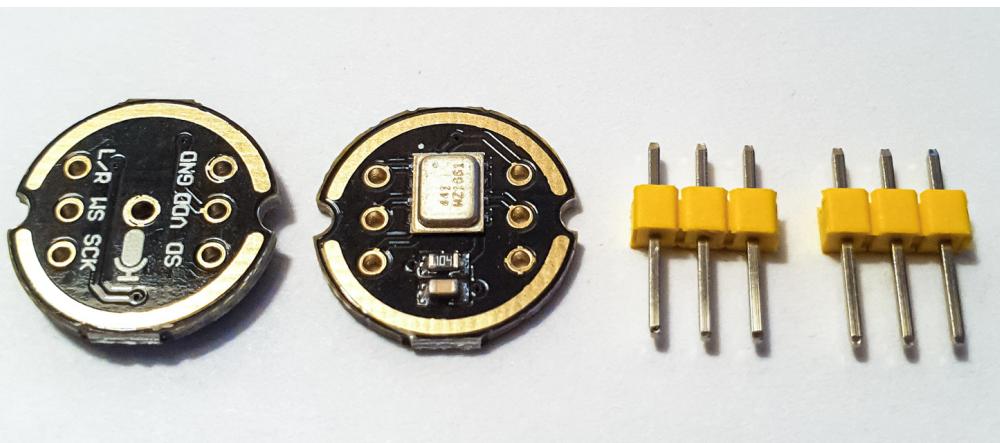
Slika 4.1: Wokrflow v programu Orange. Za prikaz na zemljevidu smo najprej z widgetom UNSData pridobili podatke v obliki tenzorjev, jih spremenili v časovno vrsto, izbrali časovno rezino za pregled in to prikazali na zemljevidu. Za prikaz podatkov na grafu, smo pridobili podatke z widgetom UNSData v obliki matrice, v razsevnem diagramu izbrali zanimivo časovno rezino, podatke zgladili in jih prikazali na grafu.



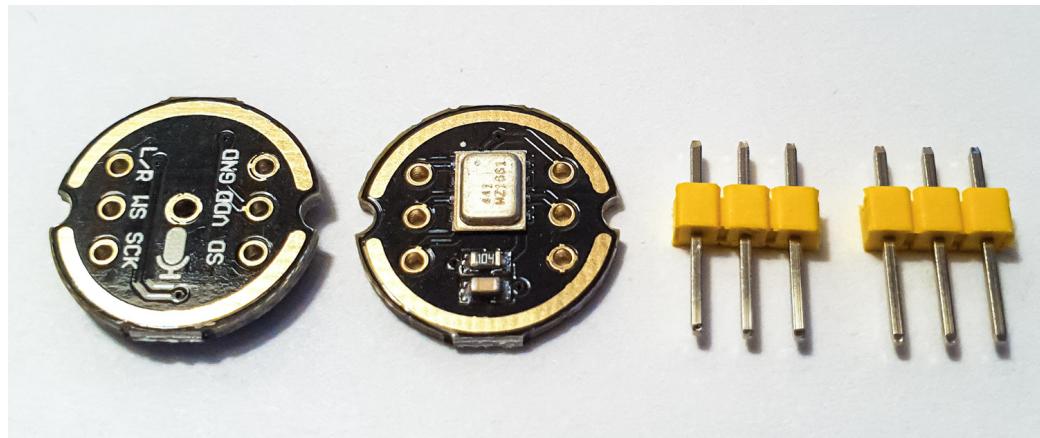
Slika 4.2: Prikaz izmerjene glasnosti v različnih časovnih obdobjih. S prikazom na zemljevidu in izbiro dovolj kratke časovne rezine, lahko prikažemo približno pozicijo izvora zvoka in kako se je ta izvor premikal skozi čas. To je primerno tako za akutne spremembe v poziciji izvora zvoka kot je prikazano tu, kot tudi za prikaz počasnejših sprememb.



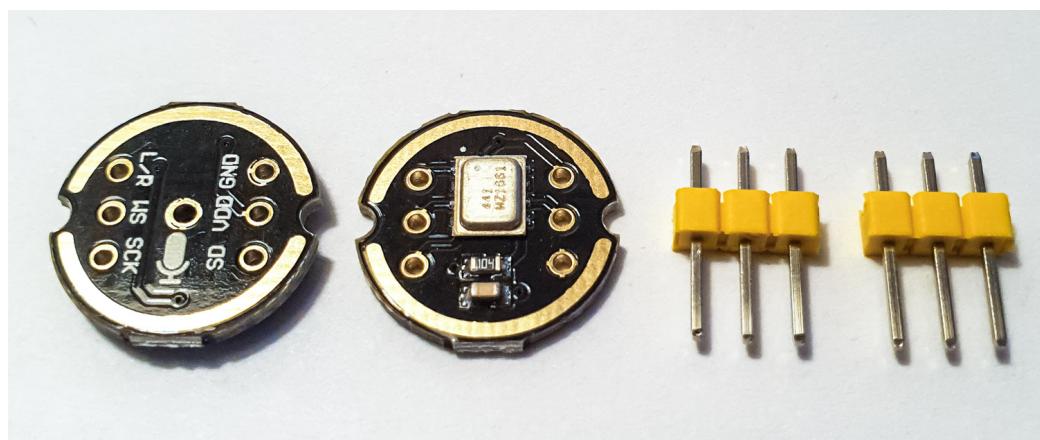
Slika 4.3: Na grafu vseh meritev po času in senzorju, se jasno vidi, kako se je kosilnica približevala in oddaljevala vsakemu od senzorjev. Ker smo senzorje postavili na rob travnika, kosilnica pa se je v spirali gibala proti sredini, je mogoče opaziti, da so vrhovi po času vedno manjši, ker je razdalja med kosilnico in senzorjem z zmanjševanjem premera kroga vedno večja.



Slika 4.4: Caption

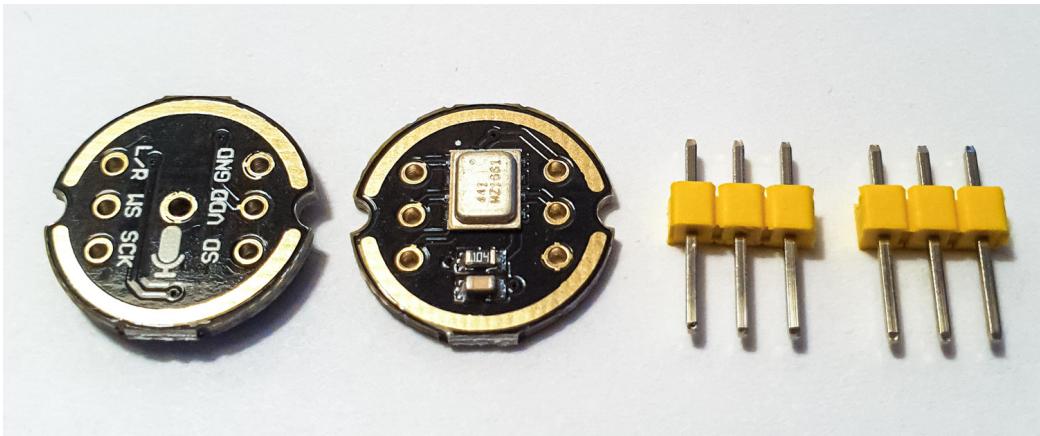


Slika 4.5: Caption



Slika 4.6: Caption

odnega ponavljanja vzorcev v glasnosti



Slika 4.7: Caption

(motivacija, postavitev, rezultati, interpretacija - štirje odstavki)

# **Poglavlje 5**

## **Zaključek**

(ni podpoglavlji, samo trije odstavki) - sklepne misli (kaj je bila naša naloga)  
- rezultat (kako smo jo uspešno rešili) - kaj še ostane (če bi imel čas in denar,  
kaj bi še lahko naredil iz tega, urbanisti, ...)



# Appendices



# Dodatek A

## Dodatek: Uporabniška navodila

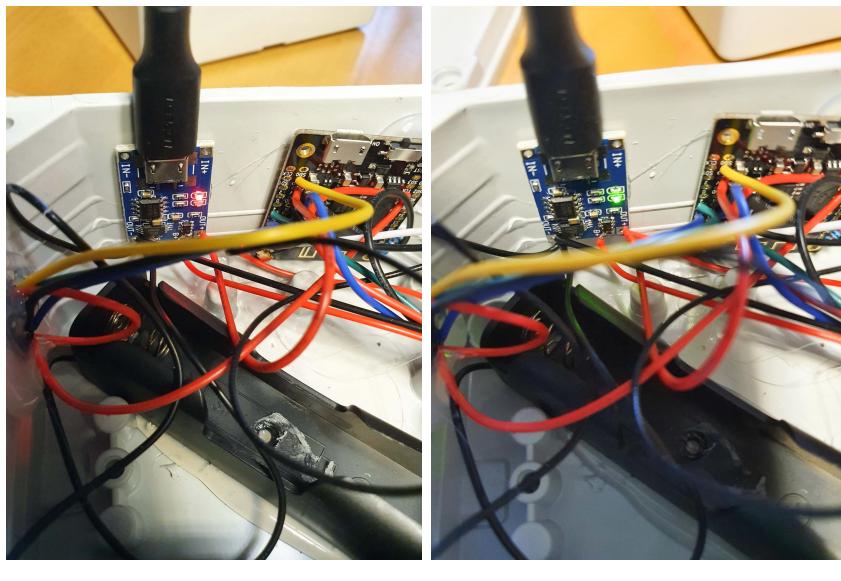
### A.1 Polnjenje meritnih enot

Meritne enote napajajo li-ion polnilne baterije. Napoljenost posamezne meritne enote se lahko preveri na strani "Sensors". Baterije so prazne pri napetosti približno 3.0V.

1. Odstranite pokrov enote, ki je pritrjen s 4 vijaki.
2. Z USB kablom povežite polnilno vezje z USB napajalnikom. Polnilno vezje je manjša ploščica z USB priključkom, montirana zraven mikrokontrolerja.
3. Polnite dokler ne sveti zelena luč. Če je naprava med polnjenjem prižgana v načinu nastavljanja, lahko preverite napoljenost na strani "Sensors". Enota je napolnjena, ko napetost preseže 4.05 V.

### A.2 Prižiganje enot v različnih načinih

Prva iteracija meritnih enot ima dve stikali, ki sta uporabljeni za prižiganje in izbiro načina delovanja. Naslednje bodo imele le eno stikalo in bo prižiganje in izbira delovanja trivialna operacija.



(a) Med polnjenjem.

(b) Polnjenje je končano.

Slika A.1: Prikaz načina polnjenja meritne enote.

1. Stikalo za napajanje premaknete v pozicijo OFF in s tem ugasnete napravo.
2. Stikalo za način delovanja prestavite v način ki se ujema z želenim načinom delovanja. "N" predstavlja način za nastavljanje, Ž pa način za zaznavanje.
3. Stikalo za napajanje premaknete v pozicijo ON in s tem prižgete napravo.

### A.3 Registracija meritne enote

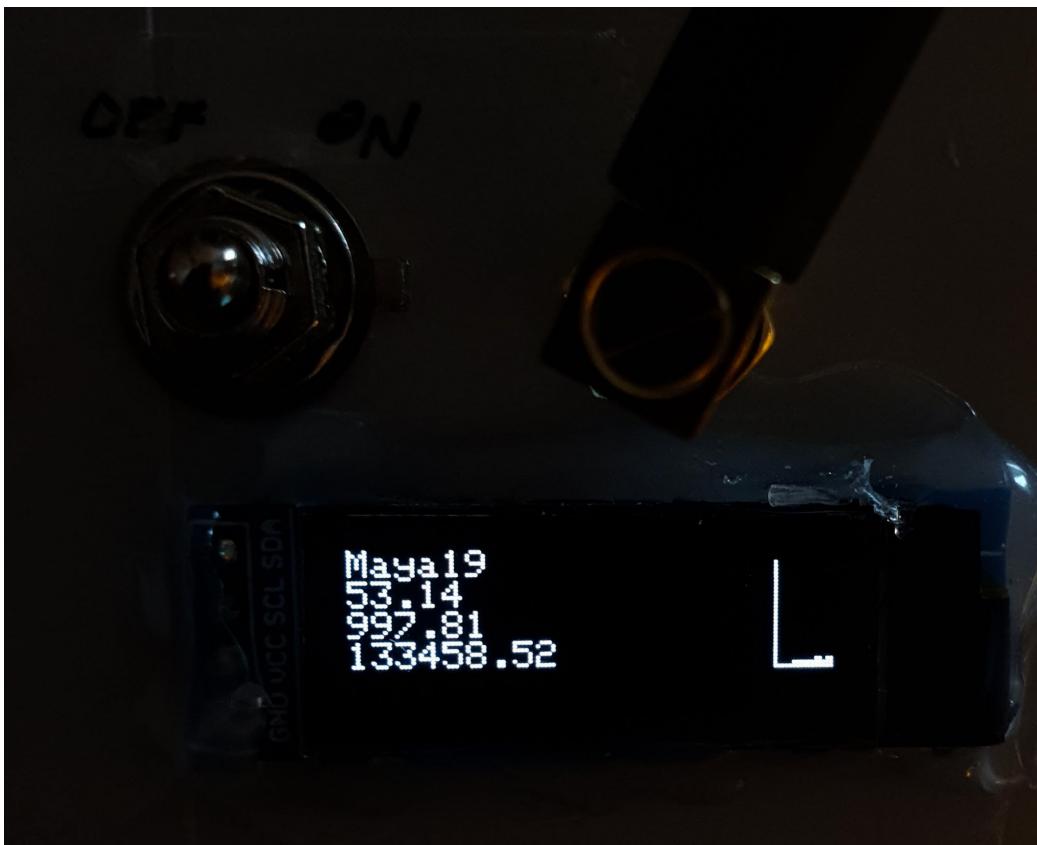
Preden enote uporabite za raziskave, jih je potrebno registrirati v sistem.

1. Poskrbite, da je napravam na voljo WiFi omrežje s povezavo na internet z SSID "UNSwifi" in geslom "uns12wifi34".
2. Prižgite enote v način za nastavljanje.



Slika A.2: Označba stikal za prižiganje enote in izbiro načina delovanja, kjer ON in OFF predstavlja prižgano in ugasnjeno enoto, N in Z pa način za nastavljanje in način za zaznavanje.

3. Počakajte, da se na ekranu izpiše ime enote.



Slika A.3: Ekran meritne enote, ko je enota v načinu nastavljanja in se je naprava uspešno registrirala v sistem. V prvi vrstici je izpisano ime meritne enote, v drugi so trenutni decibeli, v tretji pa najmočnejša frekvenca zvoka.

## A.4 Ustvarjanje raziskave

Projekt je zasnovan na principu raziskav, ki so urejene v ”deploymente”. Ustvarja in upravlja se jih preko spletnega vmesnika, ki je dostopen na naslovu: ”<http://urbannoisesensing.biolab.si/>”.

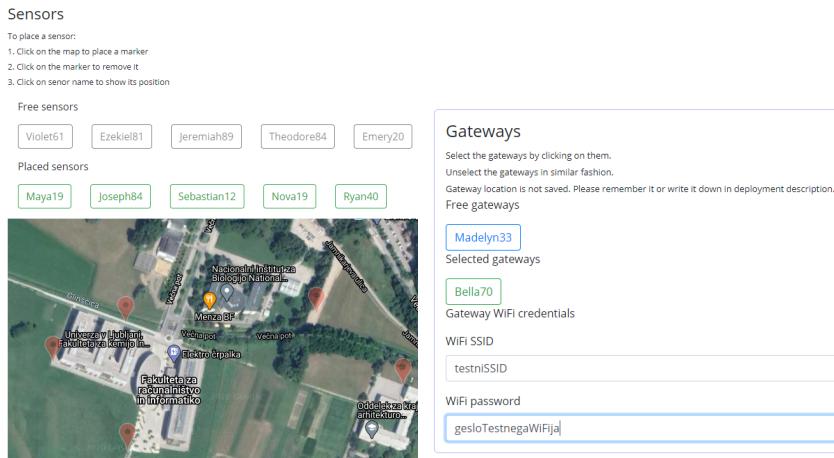
1. Registrirajte vse meritne in zbirne enote, ki bodo uporabljeni v tej raziskavi.

2. V spletnem vmesniku v zavihku "Deployments" pritisnite gumb "+", vpišite ime raziskave in pritisnite gumb "Create Deployment".
3. Nadaljujte z urejanjem raziskave.

## A.5 Urejanje raziskave

Pred začetkom zaznavanja je treba nastaviti in zagnati raziskavo. Spletni vmesnik je dostopen na naslovu: "<http://urbannoisesensing.biolab.si/>".

1. Ustvarite raziskavo, ali pa na strani "Deployments" v razdelku "Not yet deployed" izberite raziskavo, ki jo želite urediti.
2. Raziskavi uredite ime in opis tako, da spremenite besedilo v vnosnih poljih in pritisnete gumb "Save changes".
3. Merilne enote lahko raziskavi dodajate tako, da v razdelku "Sensors" s klikom na zemljevid na ustrezna mesta postavljate zaznamke.
  - S klikom na zaznamek na zemljevidu odstranite enoto iz raziskave.
  - S klikom na ime postavljene merilne enote, se zaznamek na zemljevidu obarva.
4. Zbirno enoto lahko dodate v razdelku "Gateways" s klikom na ime izbrane zbirne enote.
5. Zbirni enoti lahko dodate možnost povezovanja na dodatno WiFi dostopno točko tako, da vpišete podatke o tem omrežju v ustrezna polja v razdelku "Gateways".
6. Raziskavo zaženete s klikom na gumb "DEPLOY".
7. Raziskavo zbrišete s klikom na gumb "DELETE DEPLOYMENT".



(a) Primer postavljenih meritnih enot na zemljevidu      (b) Primer izbire zbirne enote in nastavljanja WiFi omrežja.

Slika A.4: Zaslonski posnetki uporabniškega vmesnika med nastavljanjem raziskave.

## A.6 Prenašanje podatkov na enote

Ko je raziskava zagnana, je treba podatke o nastavivtah prenesti na enote.

1. Poskrbite, da je napravam na voljo WiFi omrežje s povezavo na internet z SSID "UNSwifi" in gesлом "uns12wifi34".
2. Prižgite meritne enote v načinu za nastavljanje in počakajte, da se enota poveže na WiFi in pridobi podatke o nastavivtah.
3. Povežite zbirno enoto na napajanje in počakajte, da se poveže na WiFi omrežje.
4. Ugasnite enote.

## A.7 Postavljanje enot na lokacijo zaznavanja

Ko je raziskava zagnana in so podatki preneseni na enote, lahko postavite enote na lokacijo zaznavanja.

1. Poskrbite, da so pokrovi vseh enot ustreznno pritrjeni in da so baterije merilnih enot ustreznno polne.
2. Enote vsako posebej postavite na specificirane lokacije in jih prižgite v način za zaznavanje. Točne lokacije merilnih enot lahko preverite tako, da v spletnem vmesniku na strani "Deployments" v razdelku "Deployed" izberete trenutno raziskavo in v zavihku "Sensors" s klikom na posamezno ime enote na zemljevidu preverite, kje točno naj bi se posamezna enota nahajala.
3. Postavite in priklopite zbirno enoto v dosegu prej specificiranega WiFi omrežja.

## A.8 Uravnavanje intervala zaznavanja

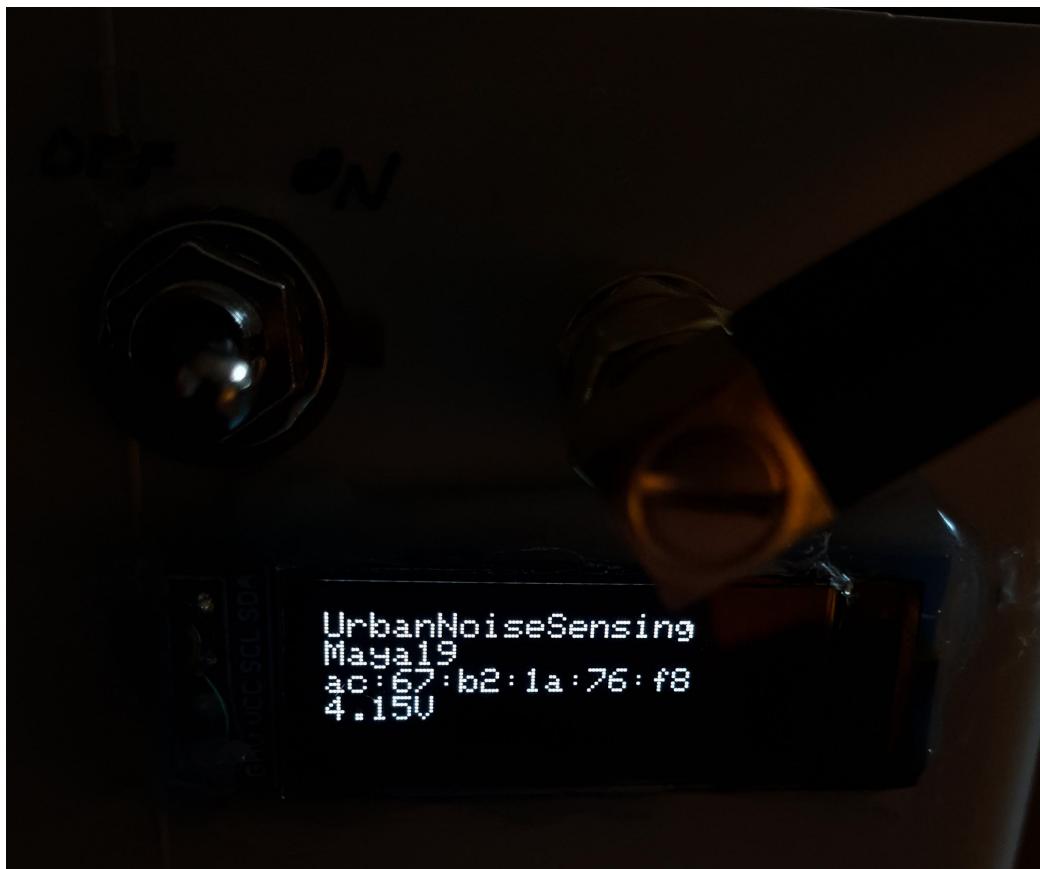
Privzeti interval zaznavanja je 1 sekunda. S povečevanjem intervala se podaljša doba delovanja baterij in zmanjša količina podatkov. Spletni vmesnik je dostopen na naslovu: "<http://urbannoisesensing.biolab.si/>".

1. V spletnem vmesniku na strani "Deployments" v razdelku "Deployed" izberite trenutno raziskavo.
2. V razdelku "Sensing interval" v vpisno polje vpišite ustreznno dolžino intervala v sekundah.
3. Pritisnite gumb "Save changes".

## A.9 Konec zaznavanja

Po končani raziskavi, je treba zaključiti "deployment". Spletni vmesnik je dostopen na naslovu: <http://urbannoisesensing.biolab.si/>.

1. V spletnem vmesniku na strani "Deployments" v razdelku "Deployed" izberite trenutno raziskavo.



Slika A.5: Ekran meritne enote, ko se pravilno nastavljena enota prižiga v načinu zaznavanja. V prvi vrstici je napisano ime projekta, v drugi je napisano ime enote, v tretji je napisan MAC naslov zbirne enote, v zadnji pa napetost na bateriji.

2. Pritisnite gumb "FINISH SENSING". Če ni bilo opravljenih nič uspešnih meritov, je treba raziskavo izbrisati z gumbom "DELETE DEPLOYMENT".



# **Literatura**