****

**Kauno technologijos universitetas**

Elektros ir elektronikos fakultetas

Individualus darbas

**Garso signalų įrašymas ir filtravimas**

Skaitmeninis signalų apdorojimas realaus laiko sistemose (T121M001)

|  |
| --- |
|  |
| **Žygimantas Marma**  Studentas |
|  |
| **Doc. Šarūnas Kilius**  Dėstytojas |
|  |

**Kaunas, 2023**

Turinys

[Paveikslų sąrašas 3](#_Toc150858447)

[Santrumpų ir terminų sąrašas 4](#_Toc150858448)

[Įvadas 5](#_Toc150858449)

[1. Literatūros analizė 6](#_Toc150858450)

[2. Kuriamos sistemos koncepcija 11](#_Toc150858451)

[3. Sistemos realizavimas 14](#_Toc150858452)

[Išvados 15](#_Toc150858453)

[Literatūros sąrašas 16](#_Toc150858454)

Paveikslų sąrašas

[1 pav. Miego apnėjos vizualizacija 6](#_Toc150809946)

[2 pav. Įvairių elektrodų ir jutiklių, naudojamų miegui stebėti polisomnografijos metu 7](#_Toc150809947)

[3 pav. Mokslininkų sukurtos sistemos struktūrinė schema [ref] 8](#_Toc150809948)

[4 pav. Mokslininkų sukurtos sistemos apnėjos detektavimui maketas 9](#_Toc150809949)

[5 pav. Mokslininkų sukurtos sistemos akselometro pagrindu [ref-other2] 10](#_Toc150809950)

[6 pav. „BlueCoin“ maketo mikrofono ir kitais sensoriais 11](#_Toc150809951)

[7 pav. „BlueCoin“ maketo dalis su SD kortele 12](#_Toc150809952)

Santrumpų ir terminų sąrašas

**Santrumpos:**

PSG – polisomnografija (angl. P*olysomnography*);

OMA – obstrukcinė miego apnėja;

MEMS (angl. micro-electro-mechanical systems) mikrofonas:

Įvadas

Miegas yra esminis geros savijautos elementas, tačiau gerai išsimiegoti pavyksta nevisiems. Miego sutrikimai, tokie kaip miego trūkumas ir nemiga, veikia mūsų imunitetą, širdies ir kraujagyslių sistemą, gali sukeli vėžį ir yra galima depresijos atsiradimo priežastis [**Error! Reference source not found.**]. Viena iš dažniausiai pasitaikančių miego sutrikimo formų yra obstrukcinė miego apnėja, kurią patiria maždaug 5-15% populiacijos [**Error! Reference source not found.**].

Miego apnėjos diagnostikos problema susijusi ne tik su brangia ir nepatogi polisomnografijos procedūra, bet ir su dideliu nediagnozuotų atvejų skaičiumi. Remiantis moksliniais tyrimais, apie 80% obstrukcinės miego apnėjos atvejų lieka nediagnozuota, o tai reiškia, kad pacientai negauna būtinos gydymo ir rizikuoja susirgti kitomis sveikatos problemomis [**Error! Reference source not found.**]. Polisomnografija (angl. *polysomnography – PSG*) yra labiausiai paplitęs metodas apnėjos diagnostikai. Atliekant šį tyrimą, pacientas guldomas per naktį miego laboratorijoje kur yra stebimi jo fiziologiniai kintamieji [**Error! Reference source not found.**]. PSG tyrimo metu registruojama kūno padėtis, galūnių judesiai, prisotinimas deguonimi, širdies ritmas ir dažnis, kvėpavimo pastangos, smegenų veikla, akių judesiai ir miego fazės [**Error! Reference source not found.**]. Nors PSG tyrimas gali pateikti tikslią diagnozę, jis yra brangus medicinos tyrimas ir sukelia nepatogumų pacientams.

Būtent todėl modernios sistemos, galinčios surinkti miego kokybės duomenis, o vėliau naudojant dirbtinio intelekto technologijas, analizuoti juos, gali būti inovatyvus žingsnis šioje srityje. Šios sistemos gali ne tik efektyviai nustatyti miego apnėjos požymius, bet ir sumažinti diagnozavimo ir stebėjimo sąnaudas, kurių kitu atveju reikalauja tradicinės polisomnografijos procedūros. Naujos, neinvazinės, sistemos naudojimas taip pat gali sumažinti laiką, per kurį galima nustatyti miego sutrikimus, prisidedant prie ankstyvo gydymo ir mažinant galimas sveikatos rizikas. Be to, šios technologijos gali suteikti galimybę pacientams stebėti savo miego kokybę namuose, neatsitraukiant nuo įprasto gyvenimo ritmo. Būtent dėl šios priežasties šiame kursiniame darbe yra pasiūloma apnėjos detektavimo sistemos dalis kuri yra atsakinga už garsų miego metu įrašymą ir jų išsaugojimą tolesniam apdorojimui.

**Darbo tikslas**: sukurti garso signalų įrašymas ir filtravimas sistemos maketą

**Darbo uždaviniai**:

1. Išanalizuoti rinkoje esančius sprendimus.
2. Suprojektuoti bei sukurti garsus gebančią įrašyti sistemą.
3. Atlikti audio signalų filtravimą.

# Literatūros analizė

## Miego apnėja

Obstrukcinė miego apnėja yra kvėpavimo sistemos sutrikimas, kuris turi įtakos kvėpavimui miegant. Šį sutrikimą sukelia pasikartojantys viršutinių kvėpavimo takų obstrukcijos (užsivėrimo) kurios trunka mažiausiai 10 sekundžių, bet gali trukti iki minutės [**Error! Reference source not found.**]. Tai gali atsirasti dėl kelių priežasčių, įskaitant raumenų, palaikančių gerklės ir liežuvio minkštuosius audinius, atsipalaidavimą, dėl kurio šie audiniai gali žlugti ir užkimšti kvėpavimo takus. Taip pat, riebalinių audinių perteklius gerklėje ir liežuvyje, padidėjusios tonzilės, didelis liežuvis ar mažas žandikaulis gali prisidėti prie kvėpavimo takų obstrukcijos miego metu. Kiti veiksniai, galintys padidinti kvėpavimo takų kolapso riziką, yra nutukimas, alkoholio vartojimas, rūkymas ir miegas ant nugaros. Šio sutrikimo liustracija vizualiai pateikiama paveikslėlyje (1 pav.).

A picture containing screenshot

Description automatically generated

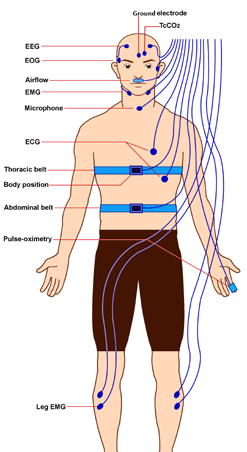
1 pav. Miego apnėjos vizualizacija[[1]](#footnote-2)

## Esantys sprendimai

### Polisomnografija

Dabartinis auksinis obstrukcinės miego apnėjos diagnozavimo standartas yra naktinė daugiakanalė polisomnografija (PSG), brangi, daug darbo reikalaujanti ir nepatogi procedūra. Detaliau polisomnograma yra procedūra, kurios metu naudojama elektroencefalograma, elektrookulograma, elektromiograma, elektrokardiograma, pulso oksimetrija, matuojamas oro srautas ir kvėpavimo pastangos, siekiant įvertinti pagrindines miego sutrikimų priežastis [**Error! Reference source not found.**]. Įprastam PSG reikalinga visapusiška stebėjimo sistema, leidžianti registruoti miego stadijas, galūnių judesius, oro srautą, kvėpavimo pastangas, širdies susitraukimų dažnį ir ritmą, deguonies prisotinimą ir kūno padėtį. Šio tipo tyrimai atliekami miego laboratorijoje, kurioje visą tyrimo laiką dalyvauja apmokytas miego technikas. PSG pirmiausia naudojami diagnozuoti su miegu susijusius kvėpavimo sutrikimus, įskaitant OMA, centrinę miego apnėją ir su miegu susijusią hipoventiliaciją / hipoksiją. Vis dėlto miego apnėja yra labiausiai paplitęs miego sutrikimas, dėl kurio diagnostiniai tyrimai atliekami miego centruose.

Kita šio tyrimo problema yra tai, kad jis yra labai brangus ir sudėtingas diagnostikos metodas, reikalaujantis specializuotos įrangos ir medicinos personalo. Lietuvoje yra mažai PSG tyrimų centų, o klinikų, turinčių PSG įrangą, yra tik kelios. Dėl to, žmonės dažnai turi laukti ilgą laiką, kol jie gali gauti polisomnografijos tyrimą. Dažniausiai tyrimo laukimo laikas yra kelios savaitės ar net iki keleto mėnesių. Toks ilgas laukimo laikas gali būti ypač problemiškas pacientams, kuriems yra įtariama miego apnėja, kadangi jie jų būklė gali dar pablogėti. Tačiau net ir atlikus PSG tyrimą dažnu atveju liga nėra diagnupzojama, dėl neįprastų asmeniui miego sąlygų. Šis stresas dėl pasikeitusios aplinkos lemią kitokį miego režimą. To būtų galima išvengti atliekant tyrimus namuose arba kitais pažangesniais neinvaziniais metodais. Būtent tai ir yra pagrindinė šiame darbe siūlomos sistemos panaudojimo motyvacija.



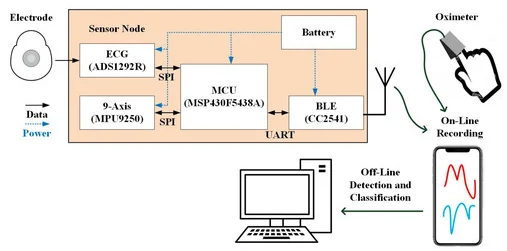
2 pav. Įvairių elektrodų ir jutiklių, naudojamų miegui stebėti polisomnografijos metu

### Naujoviškų sistemų sprendimai

Analizuojant kitų mokslininkų darbus galima pastebėti jog tobulėjant mašininio mokymosi metodams bei augant įterptinių sistemų skaičiavimų resursams vis atsiranda vis daugiau darbų tiriančių šio ligos aptikimą intymiais metodais.

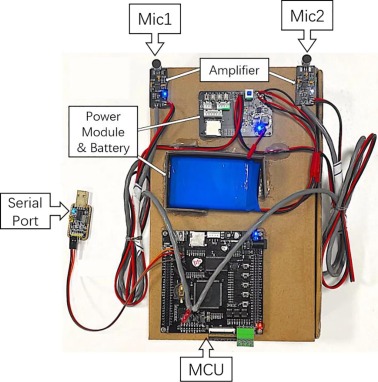
Kaip viena iš pavyzdžių galima būtų pateigti 2020 metų Taivano ir Amekiriečių mokslininkų darbą „Nešiojamojo miego apnėjos sindromo stebėjimas ir įvykių aptikimas naudojant ilgalaikę trumpalaikę atmintį pasikartojantį neuronų tinklą“ [ref others-1]. Šiame tyrime mokslininkai sprendė apnėjos aptikimo problemą sukurdami išsamų matavimo modulį. Šiame modulyje integruoti pilvo ir krūtinės ląstos triašiai akselerometrai, pulso oksimetras (SpO2) ir elektrokardiogramos (EKG) jutiklis. Norint užfiksuoti krūtinės ir pilvo judesius, buvo naudojamos pjezoelektrinės juostos, o signalai buvo registruojami 100 Hz diskretizavimo dažniu. Mokslininkai kombinavo aparatinės įrangos sprendimą kartu su naujoviška neuroninio tinklo klasifikavimo technika, kad nustatytų obstrukcinę miego apnėją.

Kad būtų galima nuolat stebėti, buvo sukurtas 27 gramų jutiklis, turintis devynių ašių akselerometrą, EKG jutiklį, Bluetooth modulį ir mikrovaldiklį. Itin mažos galios mikrovaldiklis (MSP430) valdė MPU9250 akselerometrą, kad užfiksuotų signalus, kurie per Bluetooth modulį (CC2541) buvo perduodami į mobilųjį įrenginį (išmanųjį telefoną ar planšetinį kompiuterį). EKG ir akselerometro signalo žodžio ilgis ir diskretizavimo dažnis buvo atitinkamai 12 bitų/500 Hz ir 16 bitų/50 Hz. Perdavimo sparta iš jutiklio įrenginio į iOS įrenginį buvo nustatyta 115 200 bps. Šis išsamus požiūris ne tik nagrinėjo techninius signalų gavimo aspektus, bet ir suteikė praktišką bei efektyvų sprendimą ilgalaikiam su miegu susijusių fiziologinių parametrų stebėjimui.



3 pav. Mokslininkų sukurtos sistemos struktūrinė schema [ref others-1]

Naudojantis konvoliucinių ir ilgos-trumpos atminties neuroninių tinklų kombinacija Xu Lin 2022 metų darbe pasiūlė neinvazinę apnėjos detektavimo sistemą [**Error! Reference source not found.**]. Šiame darbe mokslininkai naudojo garsus surinktus iš 32 pacientų ir sugebėjo pasiekti 73,92 procentų tikslumą naudojant keturių klasių klasifikavimą. Knarkimo garsai buvo įrašomi su *Sony PCM A100* įrenginiu naudojant du mikrofonus. Įrašymo prietaisai buvo padėti vieno metro atstumu nuo pacientų. Garso duomenų fiksavimo dažnis buvo 44,1 kHz, o įrašytas garsas buvo išsaugotas kaip dviejų kanalų *wav* failas. Tyrime buvo naudojamos Melo filtrų banku išskirti požymiai iš knarkimo garsų. Būtent Melo filtrai buvo pasirinkti remiantis žmogaus klausos sistemos netiesinio garso dažnio suvokimo ypatybėmis. Būtent naudojant šiuos Melų filtrus tyrėjai gavo 16x16 pikselių dydžio nuotraukas, kurias naudojo kaip neuroninių tinklų įvestis. Mokslininkai sukurtą modelį testavo su *STM32 ARM* procesoriaus tipo realaus laiko įterptine sistema. Dėl įterptinių sistemų resursų ribotumo tinklo parametrų skaičius buvo sumažintas 40,27 % tuo tarpu tikslumas sumažėjo tik 0,35 %. Tai parodo potencialą sukurti išmaniuosius įrenginius, naudojamus namuose, šios ligos prevencijos tikslams

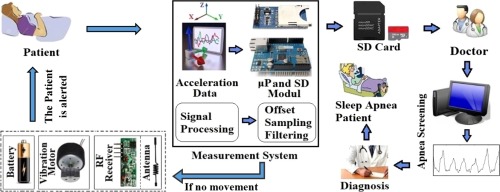


4 pav. Mokslininkų sukurtos sistemos apnėjos detektavimui maketas

Kaip kitą mokslininkų naudojančių įterptine sistema paremtą apnėjos aptikimo sistemą, galima pateikti A.H. Yüzer „Nauja nešiojama realaus laiko miego apnėjos aptikimo sistema, pagrįsta pagreičio jutikliu“ darbą sutelktą aplink pagreičio jutiklį. Pagrindinis tikslas buvo greitai nustatyti apnėjos reiškinį ir atitinkamai įspėti pacientą. Pagreičio jutiklis, esantis ant diafragmos, nuolat registruoja diafragmos judesius, naudodamas ADXL345, MEMS pagrįstą 3 ašių akselerometrą, kurio matavimo diapazonas yra ±2 g. Šio jutiklio skiriamoji geba yra 13 bitų.

Per akselerometru pagrįstą sistemą aptikus apnėjos įvykį, signalas perduodamas į apyrankę. Apyrankė su vibraciniu varikliu inicijuoja vibraciją tol, kol paciento atnaujins normalų kvėpavimą. Pažymėtina, kad vibracijos variklio intensyvumą galima pritaikyti atsižvelgiant į individualius paciento veiksnius, tokius kaip miego trūkumas, ypač aprūpinant senyvus, neįgalius ar vaikus. Pagrindinės šios sistemos yra jos gebėjimas pasiekti ir saugoti atitinkamus kvėpavimo parametrus, nereikalaujant tam skirto miego kambario. Nuolatiniai diafragmos pagreičio, imami 20 Hz dažniu, yra saugomi SD kortelėje arba per USB perkeliami į kompiuterį. Kiekvienas duomenų rinkinys sudaro iš trijų ašių duomenų, todėl duomenų perdavimo sparta yra 1920 bitų per sekundę.

Siekiant patvirtinti sistemos veiksmingumą, miego ekspertai ligoninės aplinkoje tiksliai pažymėjo tikrus miego apnėjos atveju. Siūloma realaus laiko miego apnėjos aptikimo sistema įrodė savo veiksmingumą atlikdama našumo analizę, suderinama su ekspertų pažymėtinais miego apnėjos atvejais. Šis naujoviškas sprendimas ne tik pristato naują požiūrį į apnėjos apkrovą, bet ir patenkina poreikius, todėl tai yra perspektyvi miego stebėjimo technologijos plėtra.



5 pav. Mokslininkų sukurtos sistemos akselometro pagrindu [ref-other2]

Taigi atlikus literatūros analizę matyti, kad yra paklausa sukurti išmaniėją apnėjos detektavimo sistema, bei mikrofono ir garsaai paremta sistema yra vienas iš variantų.

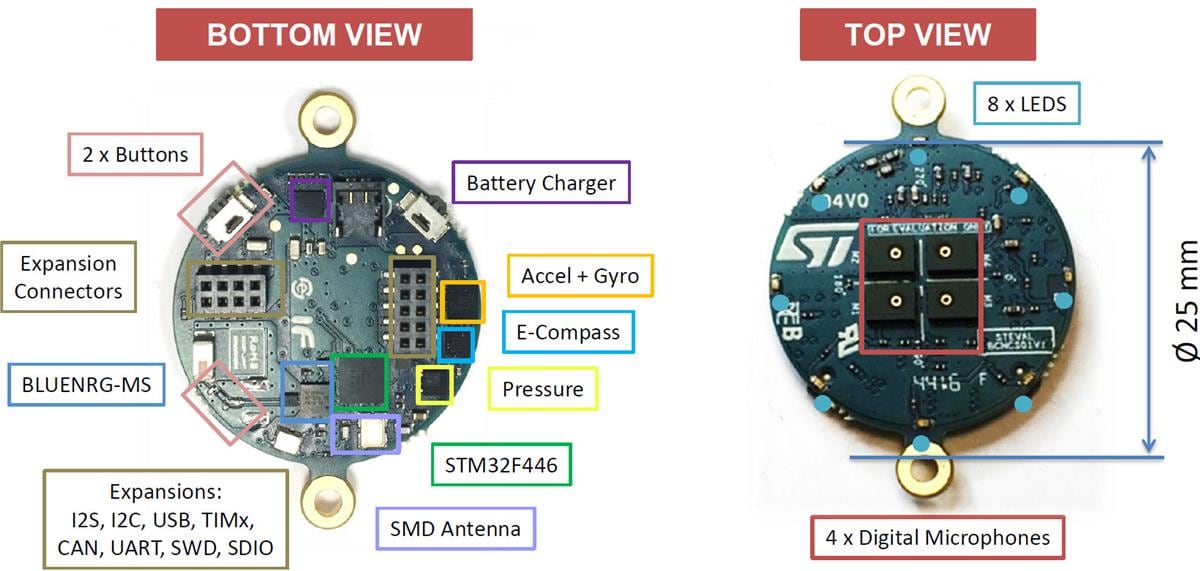
Taip pat duomenų perdavimas per blueteuht ar saugojimas į SD kortelę yra labai svarbi sistems dalsi kadangi tolimesnis apdorojimas tikriausiai bus atlikatas mokslinikų arba dirbtinio intelekto, kadangi įterpinės sistemos negali palaikyti mašininiam mokymuisi reikalingų skaičiavimo resursų.

# Kuriamos sistemos koncepcija

Šiame darbe yra siūloma apnėjos aptikimo sistemos dalis kuri gebėtų kokybiškai įrašyti garsus miego metu, kurie toliau jau bus apdorojami ir klasifikuojami medikų ar inotavių mašininių mokymosi pagrįstų sistemų ligos aptikimui ir jos stadijos klasifikavimui. Todėl norint sukurti tokį prietaisą svarbu pasirinkti tinkamą apartinę įrangą garso įrašymui ir jo saugojimui.

## STM32 „Bluecoin“ prietaisas

Taigi, kadangi norit sukurti garso analizės sistema reikalingas mikrofonas tai yra viena svarbiausių projekto dalių. Dėl greitesnės integracijos buvo pasirinktas STM32 „BlueCoin Starter kit“ prietaisas.

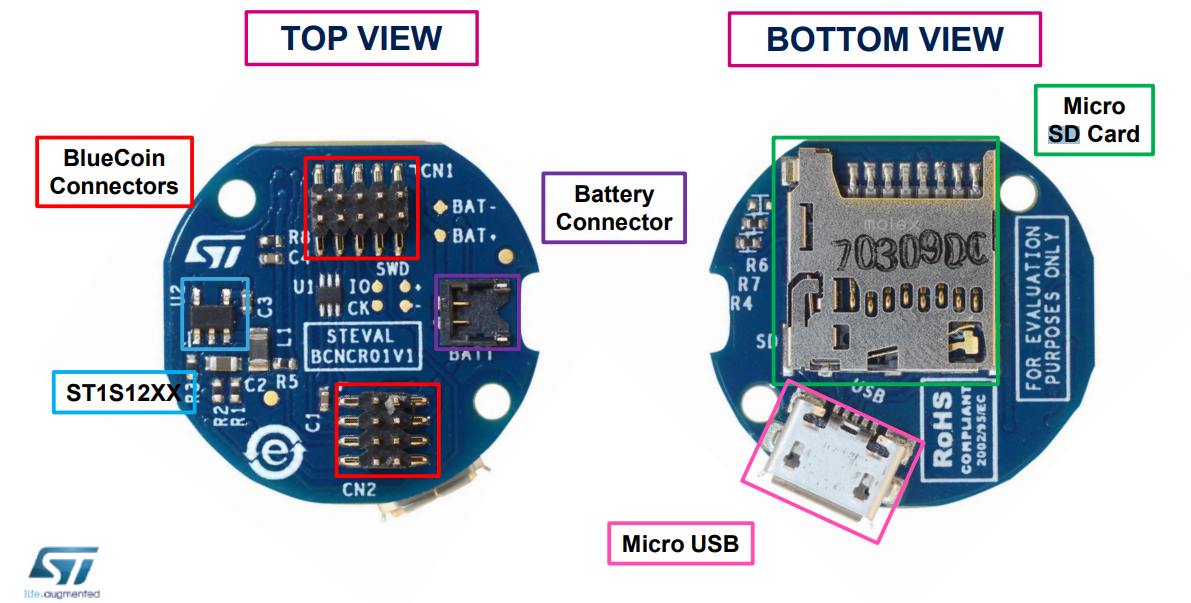


6 pav. „BlueCoin“ maketo mikrofono ir kitais sensoriais

Ši integruota testavimo ir prototipų kūrimo platforma, skirta išplėstiniam akustiniam ir judesio aptikimui daiktų interneto programoms, pagrįsta žmogaus ausies klausymosi gebėjimais. „BlueCoin“ leidžia tyrinėti pažangius jutiklius ir jų signalų apdorojimo funkcijas. Įrenginys skirtas robotikos ir automatikos programoms, naudojant 4 skaitmeninius MEMS mikrofonų rinkinius, didelio našumo 9 ašių inercinį ir aplinkos jutiklių bloką bei veikimo laiko nuotolio jutiklius.

Didelio našumo STM32F446 180 MHz CPU leidžia realiuoju laiku įgyvendinti labai pažangius jutiklių algoritmus, tokius kaip adaptyvus pluošto formavimas (angl. *adaptive beamforming*) kuris pagerina garso šaltinio lokalizavimą. Šias funkcijas naudoti yra paruoštos nemokamos bibliotekos. „BlueCoin“ taip pat turi integruotą BLE sąsają su kuria gali prisijungti per bet kurio daiktų interneto ir išmaniojo pramonės belaidžio jutiklio BlueThooth tinklo. Taip pat, naudojant šį maketą galimas neapdorotų jutiklių duomenų srautinio perdavimas per USB sąsają, duomenų registravimas į SD kortelę ar garso perdavimas.

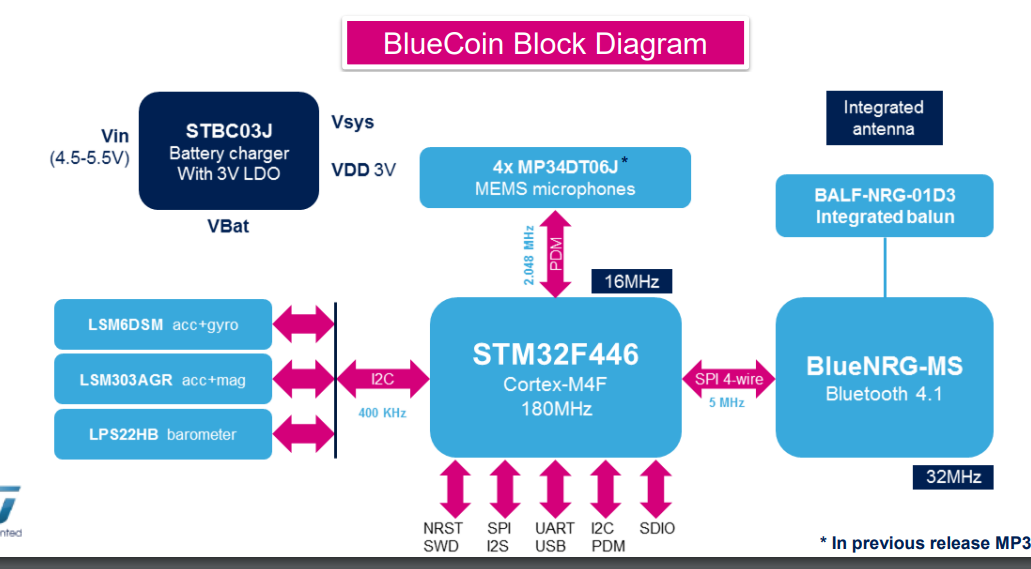
Būtent šios funkcijos ir yra reikalingos norint sukurti apnėjos detekcijos sistemą. Kadangi dažniausiai miegame ne vieni, todėl yra reikalinga atstumo kalibravimą nuo pasirinkto šaltinio lokacijos kad galima būtų identifikuoti specifinį pacientą. Taip pat SD kortelė jau yra integruota šiame įrenginyje todėl nereikia papildomos sąsajos ir prietaiso norint duomenys saugoti SD laikmenoje.



7 pav. „BlueCoin“ maketo dalis su SD kortele

## Strukturinė schema MEMS mikrofonas

Papaišyti basic mano sistemos architektūrą



pav. 8

Struktūrinė schema

# Sistemos realizavimas

## Programos algoritmas

HAL ir low level init

Jeigu nėra SD korta aptinka yra miego rėžimas

  while(!BSP\_SD\_IsDetected())

  {

    // togle some two pins

    // BSP\_LED\_Toggle(LED1);

    // BSP\_LED\_Toggle(LED2);

    /\* Go to Sleep \*/

    \_\_WFI();

  }

Nustatomi mikrofono parametrai

Tada vykdomas while ciklas

Kurio metu laukiama mygtuko pastaudimo pertraukimo.

Įvykus pertraukimui

Pasikeičia įrašymo į SD kortelę būsena rašymas/pausė

## Papildomos funkcijos

Supaprastinti failų įrašymą į SD kortelę buvo panaudota failų sistema – „FatFs“.

FatFs is a generic FAT/exFAT filesystem module for small embedded systems. The FatFs module is written in compliance with ANSI C (C89) and completely separated from the disk I/O layer. Therefore it is independent of the platform. It can be incorporated into small microcontrollers with limited resource, such as 8051, PIC, AVR, ARM, Z80, RX and etc. Also Petit FatFs module for tiny microcontrollers is available [here](http://elm-chan.org/fsw/ff/00index_p.html).

#### Features

* DOS/Windows Compatible FAT/exFAT Filesystem.
* Platform Independent. [Easy to port](http://elm-chan.org/fsw/ff/doc/appnote.html" \l "port).
* Very Small [Footprint](http://elm-chan.org/fsw/ff/doc/appnote.html" \l "memory) for Program Code and Work Area.
* Various [Configuration Options](http://elm-chan.org/fsw/ff/doc/config.html) to Support for:
  + Long File Name in ANSI/OEM or Unicode.
  + exFAT Filesystem, 64-bit LBA and GPT for Huge Storages.
  + Thread Safe for RTOS.
  + Multiple Volumes. (Physical Drives and Partitions)
  + Variable Sector Size.
  + Multiple Code Pages Including DBCS.
  + Read-only, Optional APIs, I/O Buffer and etc...

http://elm-chan.org/fsw/ff/00index\_e.html

FatFs library architecture logically separates the abstractions of the user app and the platform-dependent code. The user application and the low level disk I/O layer ([driver](https://en.wikipedia.org/wiki/Device_driver)) must be added by the implementer.[[5]](https://en.wikipedia.org/wiki/FatFs#cite_note-stm32doc-5): 11 Also, the architecture of the library implies that the system can have several storage devices with different drivers and the library can work in a multi-threaded operating system. At the application level it is hidden which physical media is used.

In the minimum implementation, the [driver](https://en.wikipedia.org/wiki/Device_driver) layer must support at least these 3 interfaces:[[5]](https://en.wikipedia.org/wiki/FatFs#cite_note-stm32doc-5): 14

* disk\_status — return [block device](https://en.wikipedia.org/wiki/Block_(data_storage)) status (not initialized, missing, protected, ready)
* disk\_initialize — initialize the physical disk
* disk\_read — read block from physical disk

This level of abstraction allows implementers to write an application once, and then [port](https://en.wikipedia.org/wiki/Porting) it on different platforms, changing only the implementation of the driver.[[6]](https://en.wikipedia.org/wiki/FatFs#cite_note-book3-6)

**FatFs** is a lightweight software library for microcontrollers and embedded systems that implements [FAT](https://en.wikipedia.org/wiki/File_Allocation_Table)/[exFAT](https://en.wikipedia.org/wiki/ExFAT) file system support.[[1]](https://en.wikipedia.org/wiki/FatFs#cite_note-book1-1) Written on pure [ANSI C](https://en.wikipedia.org/wiki/ANSI_C), FatFs is platform-independent and easy to [port](https://en.wikipedia.org/wiki/Porting) on many hardware platforms such as 8051, PIC, AVR, ARM, Z80. FatFs is designed as [thread-safe](https://en.wikipedia.org/wiki/Thread_safety) and is built into [ChibiOS](https://en.wikipedia.org/wiki/ChibiOS/RT), [RT-Thread](https://en.wikipedia.org/wiki/RT-Thread), ErlendOS,[[2]](https://en.wikipedia.org/wiki/FatFs#cite_note-2) and [Zephyr](https://en.wikipedia.org/wiki/Zephyr_(operating_system)) [real-time operating systems](https://en.wikipedia.org/wiki/Real-time_operating_system).[[3]](https://en.wikipedia.org/wiki/FatFs#cite_note-book2-3)

Most often, FatFs is used in low-power [Embedded systems](https://en.wikipedia.org/wiki/Embedded_system) where memory is limited, since the library takes up little space in RAM and program code. In the minimum version, the working code takes from 2 to 10 kB of RAM.[[4]](https://en.wikipedia.org/wiki/FatFs#cite_note-4)

## Overview[[edit](https://en.wikipedia.org/w/index.php?title=FatFs&action=edit&section=1)]

FatFs is designed to be a [Filesystem Layer](https://en.wikipedia.org/wiki/File_system) that is agnostic to the platform and [storage media](https://en.wikipedia.org/wiki/Data_storage) it is used with. This is achieved by providing a media access interface that is used to communicate with the storage device control module which is provided by the implementer.[[5]](https://en.wikipedia.org/wiki/FatFs#cite_note-stm32doc-5) This means that FatFs can work with any physical device such as an [SD card](https://en.wikipedia.org/wiki/SD_card) or a [hard disk](https://en.wikipedia.org/wiki/Hard_disk_drive) on any platform that can run plain [C code](https://en.wikipedia.org/wiki/C_(programming_language)) if the implementer provides a control module interface.

## License[[edit](https://en.wikipedia.org/w/index.php?title=FatFs&action=edit&section=3)]

FatFs has its own minimalistic license [[7]](https://en.wikipedia.org/wiki/FatFs#cite_note-7) similar to the [BSD license](https://en.wikipedia.org/wiki/BSD_License). It allows usage in commercial products without disclosing the [source code](https://en.wikipedia.org/wiki/Source_code).[[5]](https://en.wikipedia.org/wiki/FatFs#cite_note-stm32doc-5): 12 The only condition is to keep the [copyright](https://en.wikipedia.org/wiki/Copyright) notice in case of [redistribution](https://en.wikipedia.org/wiki/Freely_redistributable_software) of the source code. The conditions of FatFs license are not cover any redistributions in binary form, such as [embedded code](https://en.wikipedia.org/wiki/Embedded_software) and hex files.

lentelė 1

|  |
| --- |
| void DATALOG\_SD\_Init(void)  {    char SDPath[4];      if(FATFS\_LinkDriver(&SD\_Driver, SDPath) == 0)    {      /\* Register the file system object to the FatFs module \*/      if(f\_mount(&SDFatFs, (TCHAR const\*)SDPath, 0) != FR\_OK)      {        /\* FatFs Initialization Error \*/        while(1)        {          BSP\_LED\_On(LED1);          HAL\_Delay(500);          BSP\_LED\_Off(LED1);          HAL\_Delay(100);        }      }    }  } |

lentelė 2

|  |
| --- |
| /\*\*    \* @brief  Start SD-Card demo    \* @param  None    \* @retval None    \*/  uint8\_t DATALOG\_SD\_Log\_Enable(void)  {    static uint16\_t sdcard\_file\_counter = 0;      uint32\_t byteswritten; /\* written byte count \*/    char file\_name[30] = {0};      WavProcess\_HeaderInit();      sprintf(file\_name, "%s%.3d%s", "Audio\_Record\_N", sdcard\_file\_counter, ".wav");    sdcard\_file\_counter++;    if(f\_open(&MyFile, (char const\*)file\_name, FA\_CREATE\_ALWAYS | FA\_WRITE) != FR\_OK)    {      return 0;    }      if(f\_write(&MyFile, pHeader, sizeof(pHeader), (void \*)&byteswritten) != FR\_OK)    {      return 0;    }    return 1;  } |

lentelė 3

|  |
| --- |
| /\*\*  \* @brief Highest priority interrupt handler routine  \* @param None  \* @retval None  \*/  void SW\_Task1\_Callback(void)  {    FRESULT s;    uint32\_t byteswritten;                     /\* File write/read counts \*/        /\* Check Push Button Event  \*/    if (SD\_Log\_Enabled)    {      s=f\_write(&MyFile, &(((uint8\_t \*)Audio\_OUT\_Buff)[index\_buff]), SIZE\_BUFF, (void \*)&byteswritten);      if(s != FR\_OK)        BSP\_LED\_On(LED4);      }  } |

## Signalo filtravimas

Audio signalų filtravimas – naudojant STM api

Vartotojo sasaja ar CLI api kazkoks

Išvados

1. Darbe buv apžvelgti miego apnėjos problematika ir egiztuojantis sprendimo metodai.
2. Sukurta, sistema naudojantis MEMS mikrofonais galinti įrašyti duomenis į SD kortelę
3. Signalų filtravimas

Literatūros sąrašas

1. Šaltinis 1. Prieiga per: https://www.
2. Šaltinis 2, pp.584-595. Prieiga per: <https://www.sciencedirect.com/science/article/abs/pii/S0094576510001347>

[ref others-1] https://www.mdpi.com/1424-8220/20/21/6067

[ref others2] - <https://www.sciencedirect.com/science/article/pii/S1959031819302520>

1. Apnėjos vizualizacija, prieiga per: https://www.uvmhealth.org/healthwise/topic/tp12620 [↑](#footnote-ref-2)