České vysoké učení technické v Praze
Fakulta strojní

Ústav přístrojové a řídící techniky
Odbor Automatického řízení

Využití IoT prvků pro měření fyziologických indikátorů duševního stavu operátora

The use of IoT elements to measure physiological indicators of mental state of the operator

BAKALÁŘSKÁ PRÁCE

Vypracoval: Pavel Hron
Vedoucí práce: Ing. Mgr. Jakub Jura, Ph.D.
Rok: 2024

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Hron** Jméno: **Pavel** Osobní číslo: **500269**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav přístrojové a řídící techniky**

Studijní program: **Teoretický základ strojního inženýrství**

Studijní obor: **bez oboru**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Využití IoT prvků pro měření fyziologických indikátorů duševního stavu operátora

Název bakalářské práce anglicky:

The use of IoT elements to measure physiological indicators of mental state of the operator

Pokyny pro vypracování:

Monitoring aktuálního duševního stavu operátora (například řidiče, pilota, operátora zařízení s požadavkem na vysokou spolehlivost) se často realizije na základě dat ze snímačů EDA (elektrodermální aktivity), dýchání eventuelně oběhových charakteristik. Cílem této práce je:
1) navrhnoutt systému měření, zazanamenávání a zobrazování dat ze zmíněných snímačů
2) využít distribuované komunikační architektury, IoT prvků komunikace a opensource nástrojů
3) vytvořit prototyp daného zařízení

Seznam doporučené literatury:

[1] A. Zare and M. T. Iqbal, "Low-Cost ESP32, Raspberry Pi, Node-Red, and MQTT Protocol Based SCADA System," 2020 IEEE International IOT, Electronics and Mechatronics Conference (IEMTRONICS), Vancouver, BC, Canada, 2020, pp. 1-5, doi: 10.1109/IEMTRONICS51293.2020.9216412.
[2] Lawrence O. Aghenta, M. Tariq Iqbal. Design and implementation of a low-cost, open source IoT-based SCADA system using ESP32 with OLED, ThingsBoard and MQTT protocol[J]. AIMS Electronics and Electrical Engineering, 2020, 4(1): 57-86. doi: 10.3934/ElectrEng.2020.1.57
[3] M. Babiuch, P. Foltýnek and P. Smutný, "Using the ESP32 Microcontroller for Data Processing," 2019 20th International Carpathian Control Conference (ICCC), Krakow-Wieliczka, Poland, 2019, pp. 1-6, doi: 10.1109/CarpathianCC.2019.8765944.

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Mgr. Jakub Jura, Ph.D. U12110.3

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **26.04.2024** Termín odevzdání bakalářské práce: **31.05.2024**

Platnost zadání bakalářské práce: _____

Ing. Mgr. Jakub Jura, Ph.D. podpis vedoucí(ho) práce	prof. Ing. Tomáš Vyhlídal, Ph.D. podpis vedoucí(ho) ústavu/katedry	doc. Ing. Miroslav Španiel, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

24. 4. 2024
Datum převzetí zadání

Podpis studenta

Prohlášení

Prohlašuji, že jsem svou bakalářskou práci vypracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze dne 30. 5. 2024 .....

.....

Pavel Hron

Poděkování

Děkuji Ing. Mgr. Jakubu Jurovi, Ph.D. za vedení mé bakalářské práce a za podnětné návrhy, které ji obohatily. Dále děkuji rodině a blízkým za podporu a trpělivost.

Pavel Hron

Název práce:
Využití IoT prvků pro měření fyziologických indikátorů duševního stavu operátora

Autor: Pavel Hron

Studijní program: Teoretický základ strojního inženýrství

Obor: Bez oboru

Druh práce: Bakalářská práce

Vedoucí práce: Ing. Mgr. Jakub Jura, Ph.D.

Abstrakt: Cílem této práce je sestrojení zařízení pro měření elektrodermální aktivity kůže, dýchání a srdeční frekvence. S tím souvisí i vytvoření grafického rozhraní pro zobrazení průběhů měření. Práce obsahuje rešerši psychofyziologických jevů a jejich měření v technických systémech. Dále jsou zde rozebrána možná zařízení pro snímání a přepočet hodnot ze senzorů, protokoly pro bezdrátovou komunikaci (IoT). Práce také dokumentuje postupy při sestavení senzorů a jejich zapojení, popis prostředí pro tvorbu grafického rozhraní a jeho vývoj.

Klíčová slova: ESP32, MQTT protokol, IoT, Raspberry Pi, elektrodermální aktivita, psychofyziologie, dýchání, srdeční frekvence, aktivita mozku, HMI

Title:
The use of IoT elements to measure physiological indicators of mental state of the operator

Author: Pavel Hron

Abstract: The aim of this work is to construct a device for measuring electrodermal activity of the skin, respiration and heart rate. Related to this is the creation of a graphical interface to display the measurement waveforms. The thesis includes a survey of psychophysiological phenomena and their measurement in technical systems. Possible devices for receiving and recalculating sensor readings, protocols for wireless communication (IoT) are also discussed. The work also includes the procedures for sensor assembling and wiring, a description of the GUI environment and the GUI creation procedure.

Key words: ESP32, MQTT protocol, IoT, Raspberry Pi, electrodermal activity, psychophysiology, breathing, heart rate, brain activity, HMI

Obsah

Seznam použitých zkratek **vii**

Seznam obrázků **ix**

Úvod **1**

1 Psychofyziologické jevy **2**
1.1 EDA . . . 2
1.1.1 Historie . . . 3
1.1.2 Měření . . . 3
1.2 Dýchání . . . 4
1.3 Aktivita mozku . . . 5
1.4 Srdeční aktivita . . . 7

2 Zjišťování duševního stavu v technických systémech **9**
2.1 BIOCYS . . . 9
2.2 CPAI . . . 10
2.3 Další kontrolní principy . . . 11

3 Dostupné řídící zařízení **13**
3.1 Další vývojové desky . . . 14
3.1.1 ESP8266 . . . 14
3.1.2 Arduino . . . 14
3.1.3 STM32 . . . 17
3.2 Raspberry Pi . . . 17

4 Bezdrátová komunikace **19**
4.1 MQTT protokol . . . 20
4.2 Další protokoly IoT . . . 21

5 Praktická část **23**
5.1 Výroba senzoru měření EDA . . . 23
5.2 Výroba senzoru dýchání . . . 24
5.3 Zapojení senzorů . . . 25
5.4 Napájení vývojových desek ESP32 . . . 28
5.5 Komunikace přes MQTT . . . 30
5.6 Konfigurace Raspberry Pi . . . 35
5.6.1 Vytvoření dockeru . . . 36
5.6.2 Sběr a ukládání dat . . . 38
5.6.3 Vizualizace dat . . . 41
5.7 Návod k použití . . . 43
5.8 Postup při nabíjení . . . 44
5.9 Postup při nahrání kódu . . . 44

Závěr **45**

Bibliografie **47**

Seznam použitých zkratek

EDA Elektrodermální aktivita

GSR Galvanic Skin Response

AC Střídavý proud

DC Stejnosměrný proud

EEG Elektroencefalografie

ARAS Ascendentní retikulární aktivační systém

mmHg Milimetry rtuťového sloupce

EKG Elektrokardiograf

PPG Fotopletysmograf

BIOCYS Biocybernetic Analysis System

MTTF Mean Time To Failure

MTBF Mean Time Between Failures

MTTR Mean Time To Repair

SPO Single Pilot Operations

CPAI Cognitive Pilot-Aircraft Interface

fNIRS Functional Near-Infrared Spectroscopy

DMS Dead Man's Switch

SRAM Static Random Access Memory

GPIO General Purpose Input/Output

3V3 3.3 V - pin s výstupním napětím 3,3 V

GND Ground - zemnicí pin

VIN Voltage In - napájecí pin

UART Univerzální asynchronní přijímač/vysílač

SPI Sériové periferní rozhraní

I2C Inter-Integrated Circuit - sběrnice

ADC Analogově-digitální převodník

DAC Digitálně-analogový převodník

PWM Pulzně šířková modulace

MQTT Message Queuing Telemetry Transport

IoT Internet of Things

QoS Quality of Service

TCP Transmission Control Protocol

TLS Transport Layer Security

HTTP Hypertext Transfer Protocol

API Application Programming Interface

CoAP Constrained Application Protocol

UDP User Datagram Protocol

AMQP Advanced Message Queuing Protocol

PPP Point-to-Point Protocol

XMPP Extensible Messaging and Presence Protocol

XML Extensible Markup Language

DDS Data Distribution Service

M2M Machine-to-Machine

DTLS Datagram Transport Layer Security

SSL Secure Sockets Layer

NFC Near Field Communication

BAT+ Pin pro kladný pól baterie

BAT- Pin pro záporný pól baterie

VOUT+ Kladný výstupní pin

VOUT- Záporný výstupní pin

JSON JavaScript Object Notation

SSH Secure Shell

Seznam obrázků

1.1 Schéma exosomatické metody [6] 4
1.2 Označení elektrod podle rozmístění na hlavě [11] 6
1.3 Psychické stavy subjektů měřených s pomocí EEG [12] 6
1.4 Popis duševních stavů pomocí EEG [13] 7

2.1 Koncepční pracovní postup CPAI [25] 11

3.1 ESP32 DEVKIT DOIT pinout [29] 13
3.2 Arduino Micro [30] 14
3.3 LilyPad Arduino [30] 15
3.4 Arduino Uno pinout [31] 16
3.5 Arduino Tre [30] 17
3.6 Raspberry Pi 4 Model B [33] 18

4.1 Schéma IoT [34] 19

5.1 Senzor měření EDA 23
5.2 Aplikace senzoru na bříška prstů 24
5.3 Detail stěžejní části senzoru dýchání 25
5.4 Schéma zapojení 26
5.5 Schéma nahrazení děliče napětí [40] 26
5.6 Zapojení měřicího můstku [40] 27
5.7 Zapojení senzorů do první desky ESP32 28
5.8 Zapojení senzoru dýchání do druhé desky ESP32 28
5.9 Obvod pro napájení ESP32 a nabíjení baterie 29
5.10 Finální podoba senzorů EDA a srdeční aktivity 30
5.11 Finální podoba senzoru dýchání 30
5.12 Úvodní obrazovka programu IOT Stack 37
5.13 Výběr aplikací 37
5.14 Možnosti kontejneru aplikace Node-RED 38
5.15 Kontrola stavu aplikací 38
5.16 Databáze InfluxDB v terminálu 39
5.17 Prostředí Node-RED 39
5.18 Nastavení uzlu *change* 40
5.19 Nastavení uzlu *influxdb out* 40
5.20 Potvrzovací zpráva ze sériového monitoru 41
5.21 Tvorba vizualizace 42
5.22 Úprava souboru *docker-compose.yml* 43

Seznam zdrojových kódů

5.1 Kód pro výpočet odporu kůže a srdeční frekvence 31
5.2 Kód pro výpočet dýchání 33
5.3 Příkaz pro vyzkoušení spojení 36
5.4 Příkaz navázání bezdrátového spojení 36
5.5 Příkazy pro instalaci dockeru 36
5.6 Příkazy pro spuštění dockeru 36
5.7 Kontrola aktivity kontejnerů 38
5.8 Příkaz pro spuštění databázového programu 39
5.9 Příkaz pro vytvoření databáze 39
5.10 Příkazy pro kontrolu sběru dat 41
5.11 Příkazy ke vstupu do editace souboru dockeru 42

Úvod

V dnešní době je stále větší zájem o porozumění složitosti lidského organismu a jeho reakcím na různé vnější i vnitřní podněty. S narůstajícími požadavky systému člověk - stroj roste i požadavek na monitorování duševních stavů operátora. Duševní stav můžeme odhadovat na základě vybraných fyziologických veličin. Hledáním spojitostí mezi fyziologickými proměnnými a duševním stavem se zabývá psychofyziologie a tato oblast zažívá rozvoj díky relativní dostupnosti senzorové techniky i výpočetního hardwaru.

V práci se budu zabývat vývojem prototypu zařízení pro měření vybraných fyziologických veličin a na základě toho estimovat psychické charakteristiky člověka. To bude zahrnovat rozvahu nad možnými koncepty řešení. Nezbytností bude vybrat správný hardware pro zpracování dat a jejich následnou vizualizaci, pro naplnění požadavků internetu věcí a jednoduchost musí hardwarové prvky komunikovat bezdrátově. V internetu věcí se podobné problémy často řeší použitím vývojových desek Arduino, ESP32, pro složitější aplikace se používají třeba počítače Raspberry Pi. Bude nutné vyřešit otázku sestavení a zapojení senzorů pro měření elektrodermální aktivity, dýchání a tepové frekvence. Senzory budou napojené na vývojové desky ESP32, ty budou data zpracovávat, přepočítávat na potřebné hodnoty a bezdrátově je posílat dále do počítače Raspberry Pi, jenž bude fungovat jako médium zprostředkující komunikaci. Dále musí mít celý projekt vyřešené nezávislé napájení. Nejlepší volba bude využití baterií, s nimi bude zařízení přenosné. Baterie bude potřebovat nabíjení. Na trhu se dají sehnat poměrně jednoduché efektivní moduly, které baterie chrání před nešetrným nabíjením a zároveň mohou převádět výstupní napětí z baterie na požadovanou hodnotu. Aby bylo zařízení méně náchylné na poničení a vhodnější k manipulaci, bude 3D tiskárnou vytvořen kryt. Přes výše zmíněné Raspberry budu dále plnit vytvořenou databázi daty ze senzorů, která si bude brát webová aplikace Grafana a bude vykreslovat průběhy měřených veličin.

Kapitola 1

Psychofyziologické jevy

Psychofyziologie představuje obor, který zkoumá psychické procesy a chování jedince ve vztahu k fyziologickým pochodům. Zaměřuje se tedy na výzkum kognitivních, emocionálních, motivačních a behaviorálních procesů z pohledu biologických dějů a následně i z pohledu na zdraví a nemoci [1]. Psychofyziologické jevy jsou procesy, které propojují psychické procesy a fyziologické reakce. Důležité je, že se dají zaznamenávat a na jejich základě je možné vyhodnocovat duševní stav člověka.

1.1 EDA

Elektrodermální aktivita (EDA) zahrnuje širší škálu pojmů jako jsou odezva kožního odporu (SRR - Skin Resitance Response), hladina kožního odporu (SRL - Skin Resistence Level), odezva kožní vodivosti (SCR - Skin Conduction Response), hladina kožní vodivosti (SCL - Skin Conduction Level), odezva kožního potenciálu (SPR - Skin Potential Response), hladina kožního potenciálu (SPL - Skin Potential Level). Každá proměnná specifikuje jiného a největší rozdíl mezi nimi představuje použitá měřicí metoda [2]. V tomto projektu .

EDA je fenomén zahrnující proměnlivý elektrický odpor lidské kůže, těsně související s aktivací sympatického větve nervového systému. Vodivost kůže se zvyšuje s produkcí potu. Pocení je v lidském těle nejintenzivnější na dlaních a chodidlech, kde jsou potní žlázy často drážděny v důsledku hmatových vjemů. Z toho důvodu jsou tyto oblasti lidského těla jedny z nejcitlivějších. Pot obsahuje vodu s ionty, takže dokáže vést elektrický proud. Pokud se dostane na kůži, snižuje se její odpor, který je narozdíl od vnitřku těla nezanedbatelně velký a pokud je kůže dokonale suchá, brání vedení proudu lidským tělem. Za léta výzkumu bylo zjištěno, že pokud je okolní teplota vyšší než 25 °C začíná psychologický význam EDA klesat [3].

Časovou řadu kožního odporu lze charakterizovat pomalu se měnící tonickou aktivitou (tj. úrovní kožní vodivosti; SCL) a rychle se měnící fázickou aktivitou (tj. SCR) [4]. Fázická aktivita vyjadřuje okamžitou změnu kožní vodivosti jako odpověď na zřetelný podnět. Tonická aktivita se dá vysvětlit jako reakce na podněty v delším časovém úseku, alespoň 30 sekund. Používá se pro popis celkového emocionálního stavu zkoumaného člověka a jeho bdělosti.

1.1.1 Historie

Emil du Bois-Reymond v roce 1849 poprvé popsal proudění elektrického proudu kůží. Tehdy si vůbec neuvědomoval, že by to mohlo souviset s potními žlázami a jejich sekrecí. K tomuto poznatku se dostali až o téměř třicet let později dva Švýcaři Hermann a Luchsinger, kteří si všimli, že proud je nejlépe veden na dlaních rukou a tehdy je napadla spojitost, že EDA by mohla souviset s aktivitou potních žláz. Spojitost mezi změnami krevního toku a změnami kožního odporu popsal ve své studii francouzský vědec Joseph Vigourox [5].

O pár let později začalo testování na lidech, šlo zejména o nervově labilní nebo drogami posilněné pacienty, zkrátka chtěli vědci zjistit, jak moc vodivost kůže ovlivňují nestandardní psychické stavy člověka. Velký příspěvek k tomuto rozvoji učinil ruský vědec Ivane Tarkhnishvili, který vyvinul přístroj pro sledování změn kožního odporu v čase.

Prvního pojmenování se EDA dočkala v roce 1907. Tehdy ji Jordan Peterson s Carlem Jungem pojmenovali „psychogalvanický reflex“. Už o dva roky později přišel švýcarský neurolog Otto Veraguth s názvem, který se dlouho používal, a sice „galvanický kožní reflex“ (GSR). Výraz „elektrodermální aktivita“ se pro popis tohoto psychofyziologického fenoménu používá celosvětově od roku 1981.

Význam EDA začal ve 20. století exponenciálně růst a tento trend trvá dodnes. Za poměrně krátkou dobu se EDA objevila ve stovkách vědeckých publikací a její monitorování se čím dál častěji prosazuje do moderních lékařských metod.

1.1.2 Měření

Od počátků snímání EDA byl popsány dvě metody měření: endosomatická a exosomatická. Měřicí zařízení v dnešní době jsou založené na těchto dvou metodách.

Před samotným měřením může proběhnout statická a dynamická kalibrace měřicího zařízení. Statická kalibrace se provede prostým připojením elektrod exosomatického měřicího zařízení k rezistoru se známým odporem. Dynamická kalibrace se provede po připojení subjektu k zařízení. Konkrétně hluboké nadechnutí, hlasité tlesknutí, plácnutí do vnitřní dolní části paže nebo tváře, zakašlání, poškrábání, lehké štípnutí by měly vést ke zvýšení úrovně kožní vodivosti přibližně po jedné sekundě [6]. Pokud je jisté, že měřicí zařízení funguje, ale dynamická kalibrace se na výsledku neprojevila, je možné, že zkoumaný subjekt je odolný vůči změnám EDA. Přesné číslo se neví, ale odhaduje se, že mezi obyvateli planety je 5 až 20 % tzv. non-responderů.

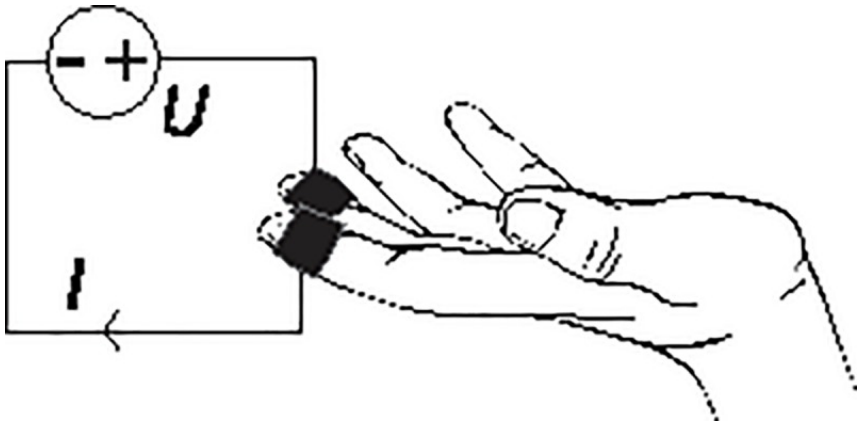
Exosomatická metoda

Exosomatická metoda spočívá v přivedení velmi malého napětí na kůži zkoumaného subjektu. Velikost tohoto napětí bývá zhruba 0,5 až 1 V [6]. Tato metoda se může ještě dále větvit podle použitého typu proudu na AC exosomatickou metodu (na kůži subjektu je přivedeno střídavé napětí) a DC exosomatickou metodu (na kůži subjektu je přivedeno stejnosměrné napětí). Zkoumá se změna intenzity vnějšího proudu, která závisí na změnách kožního odporu.

DC exosomatická metoda je v dnešní době nejpoužívanější ze všech, protože je jednodušší než AC exosomatická (elektrody nemusí měnit svou polaritu). V oblasti DC ještě nedošlo ke shodě, zda je lepší měřit se zdrojem konstantního napětí nebo se zdrojem konstantního proudu. V případě obvodu s konstantním zdrojem napětí lze vodivost kůže měřit přímo jako výstup obvodu, nemusí docházet k další transformaci dat. Obvody s konstantním zdrojem proudu jsou však stabilnější a přesnější, avšak je třeba věnovat velkou pozornost možnému poškození potních kanálků v důsledku injektovaného proudu přes malou plochu kůže [7].

Endosomatická metoda

Její výskyt není tak častý jako u metody exosomatické. Při endosomatické metodě není na kůži subjektu přivedeno žádné přídavné napětí. Závisí zde pouze na rozdílu elektrických potenciálu mezi měřeným a zvoleným referenčním bodem. Ačkoli se jedná o poměrně neznámý bioproces, připouští se, že změny kožního potenciálu během sympatické aktivity mohou být vyvolány reabsorpcí sodíku přes stěny potních kanálků a následnou změnou iontového potenciálu v potních kanálcích [7]. Endosomatická metoda je sice ještě jednodušší než DC exosomatická, protože nejsou potřeba žádné zesilovače nebo spojovací elektrické obvody ale může mít větší odchylky.



Obrázek 1.1: Schéma exosomatické metody [6]

1.2 Dýchání

Dýchání, jakožto jeden z vrozených reflexů člověka, ovlivňuje činnost všech životně důležitých orgánů, a tedy celého organismu. Frekvence dýchání může být velmi snadno ovlivněna emocionálními výkyvy, podobně jako elektrodermální aktivita. Je však v silách člověka naučit se dýchání ovládat a tím zlepšit schopnost regulace emocí. Rychlé nádechy a výdechy zvyšují hladinu stresových hormonů, což dostává psychiku do stavů úzkosti a napětí. Při zaměření na pomalé a hluboké dýchání dochází ke stimulování parasympatické nervové větve, tudíž se snižuje hladina stresu a uvolňuje se mysl.

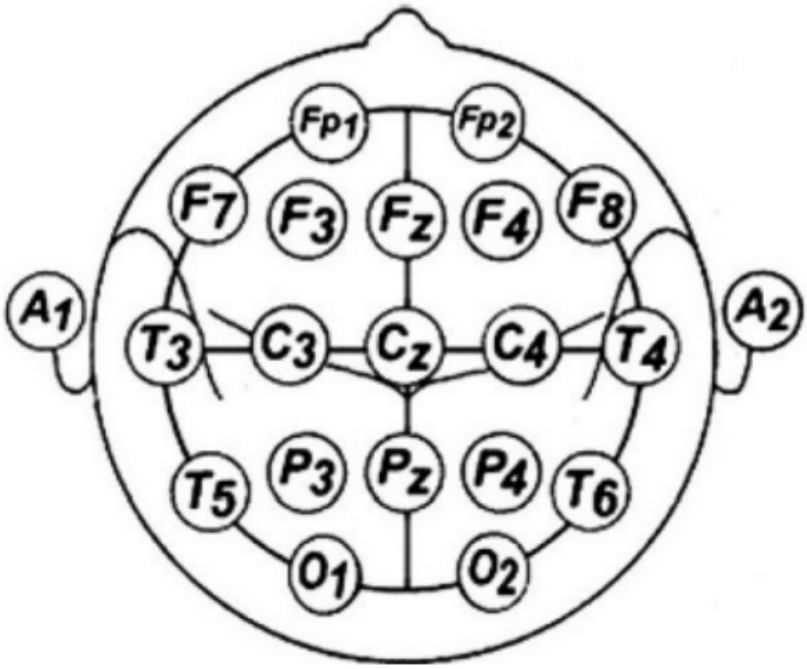
Při zjišťování psychického stavu subjektu hraje velkou roli poměr délky nádechu ku délce výdechu. Tomuto poměru se říká střída. Pokud je poměr blízký nebo roven 1,

tzn. nádech a výdech trvá přibližně stejnou dobu, je velmi pravděpodobné, že subjekt se nachází v nepříjemné situaci a je vystresovaný. Pokud se naopak poměr blíží hodnotě 1:9, zřejmě máme co do činění se subjektem v režimu relaxace. Typická hodnota střídy pro běžný stav dýchání je někde okolo 0,5.

1.3 Aktivita mozku

Jedna z nejčastějších technik pro vyhodnocování psychického stavu člověka v technických systémech spočívá v měření aktivity mozkových vln. Tato metoda se nazývá elektroencefalografie (zkráceně EEG) a spočívá v časovém záznamu bioelektrického potenciálu na povrchu hlavy pomocí speciální čepice na hlavě zkoumaných pacientů. Čepice obsahuje elektrody, které detekují sumární iontové proudy neuronů a hlásí je jako rozdíly napětí v mimobuněčném prostoru s nízkým odporem [8]. Frekvence záznamu EEG závisí na aktivitě ascendentního retikulárního aktivačního systému (ARAS). ARAS se skládá z několika neuronálních okruhů spojujících mozkový kmen s mozkovou kůrou [9]. Fylogeneticky se jedná o velmi starou strukturu nervstva, která je tvořená mnoha malými neurony nahromaděnými v nestejně velikých jádrech propojených velkým množstvím spojů. S postupující únavou se aktivita tohoto systému zpomaluje, naopak při vyrušení ze spánku nebo probouzení začíná aktivita systému opět nabíhat a člověk se dostává do stavu bdělosti. Průběhy této aktivity jsou vidět na záznamech z EEG. Jak aktivita systému v různých stavech vypadá je možné si prohlédnout na obrázku 1.3.

Při snímání pomocí EEG se měří signál miliardových populací neuronů. Signál jednoho neuronu by se totiž přes lebku, jež působí jako elektrický izolant, neprojevil. To se zajistí způsobem, jemuž se říká „systém 10-20“. Zvolí se referenční body, v praxi to jsou body v sagitální rovině nasion (na kořeni nosu) a inion (na hrbolku týlní kosti) a body v koronární rovině, tzv. preaurikulární (před ušními boltci). Další body jsou od referenčních bodů rozmístěny mezi jejich spojnicemi vždy ve stejných vzdálenostech od 10 do 20 % v závislosti na velikosti hlavy [10]. Na tyto body se umisťují elektrody, jejichž počet je standardně 19, z čehož je 8 párových a 3 nepárové. Elektrody jsou označené písmeny podle části hlavy, kam patří (např. písmeno T označuje elektrodu na spánkovou kost), a čísly podle toho, zda patří na levou (liché číslo) nebo pravou (sudé číslo) hemisféru. Nepárové elektrody mají místo číselného symbolu označení malým písmenem *z*. I přes fakt, že se měří miliardové populace neuronů, je potřeba signál zesílit pomocí zesilovače a odfiltrovat od šumu. Takto upravený signál se vykresluje na obrazovku. I když se měření metodou EEG v praxi používá docela často, může být poměrně komplikované, protože elektrické signály jsou slabé, a navíc ty vyslané z protilehlých záhybů šedé kůry mozkové ve stejnou dobu a s podobnou velikostí se mohou mezi sebou vyrušit.



Obrázek 1.2: Označení elektrod podle rozmístění na hlavě [11]

Na obrázku níže jsou vidět příklady stavů, které při měření aktivity mohou nastat. Na obrázku 1.4 jsou poté všechny stavy popsány.

Frequency band	Speed (Hz)	Mental state	Electroencephalography (EEG) recording
Delta	1-4	Deep sleep	
Theta	4-8	Drowsy	
Alpha	8-12	Relaxed	
Beta	12-30	Focused	

Obrázek 1.3: Psychické stavy subjektů měřených s pomocí EEG [12]

typ rytmu	normální nebo nenormální	rozsah frekvence (Hz)	amplitu da	doba přítomnost i rytmu	lokální nebo difuzní	oblast převahy nebo maxima	podmínky přítomnosti
alfa	normální	8-12	5-100	5-100%	difusní	okcipitální a parietální	bdění, relaxace, zavřené oči
beta	normální	18-30	2-20	5-100%	difusní	precentrální a frontální	bdění, motorický klid
gama	normální a spánková deprimace	30-50	2-10	5-100%	difusní	precentrální a frontální	bdění
delta	normální, nenormální	0,5-4,0	20-200	variabilní	difusní	variabilní	ospalost bdění
theta	normální (?) nenormální	5-7	5-100	variabilní	lokální	frontální a temporální	bdění, vzrušení nebo stres
kappa	normální	8-12	5-40	lokální	variabilní	přední a temporální	bdění při řešení problému
lambda	normální (?)	pozitiv. negativní hrot nebo ostré vlny	5-100	variabilní	lokální	parieto-okcipitální	vizuální stimul. nebo otevření očí
K-komplex	normální (?)	pozitivně ostrá vlna + jiné pomalé pozit-negativní	20-50	variabilní	difusní	vertex	bdění – sluchová stimulace
	normální		50-100	variabilní	difusní	vertex	ospalost – různá stimulace
spánková vřetena	normální	12-14	5-100	variabilní	lokální	precentrální	nástup spánku

Obrázek 1.4: Popis duševních stavů pomocí EEG [13]

1.4 Srdeční aktivita

Mezi další významné ukazatele psychického stavu může být zahrnuta také srdeční aktivita. Pod tímto srdeční frekvenci neboli tep. Krevní tlak je definován jako tlak krve na stěny tepen. Tlak měřený při kontrakci srdce se nazývá systolický, zatímco tlak měřený při relaxaci srdce se nazývá diastolický [14]. Diastolický tlak by tedy měl být nepřekvapivě nižší, obecně se uvažují hodnoty okolo 80 mmHg (milimetry rtuťového sloupce), u systolického potom přibližně 120 mmHg, záleží hlavně na stáří a pohlaví. Jak se tepny stahují a zase roztahují v důsledku proudění krve, vzniká oscilující průběh označovaný jako srdeční tep. Podobně jako u předchozích psychofyziologický jevů se ve vypjatých situacích mění krevní tlak i srdeční frekvence. Normální hodnoty srdeční frekvence jsou 60 až 100 pulzů za minutu u dospělých a 100 až 120 pulzů za minutu u dětí [14]. Při velké fyzické nebo psychické zátěži se může srdeční frekvence až zdvojnásobit. Tlak může také velmi stoupnout, což není dobré pro stěny tepen. Kromě toho se ukázalo, že lidské orgány, jsou více náchylné k poškození spíše změnami tlaku než jeho ustálenými hodnotami, a v důsledku toho lze variabilitu tlaku považovat za nezávislý rizikový faktor kardiovaskulární morbidity [15]. Pro měření krevního tlaku se v praxi používají dvě metody - přímé invazivní a nepřímé neinvazivní.

Invazivní (intraarteriální) monitorování krevního tlaku je běžně používanou technikou na jednotkách intenzivní péče a často se používá také na operačním sále. Tato technika zahrnuje zavedení katétru do vhodné tepny a následné zobrazení naměřené tlakové vlny na monitoru [16]. Tento záznam je nejpřesnější pro zjištění průběhu, ale je pochopitelně člověku nepříjemný, a tak se používá jen v nejnutnějších případech, kdy se očekávají náhlé výkyvy krevního tlaku, kdy pacienti dostávají léky na udržení hodnoty tlaku, anebo při vážných poranění hlavy. Výhoda této metody je

právě přesnost, dále odpadá riziko stresu z nafukování manžety při neinvazivní metodě. Také to může být jediná možná metoda pro pacienty s morbidní obezitou nebo edémem. Zavedení celého systému však může být v některých případech (pacient v šoku) obtížné, v oblasti zavedené kanyly mohou vznikat infekce, lokální trombózy, to se však děje v ojedinělých případech. Celý systém je ale nákladnější než u neinvazivní metody a monitorovací zařízení je plně závislé na dodávce elektrické energie [16].

Neinvazivní metody jsou tu s námi přibližně od počátku 19. století. Jejich základem bývají nafukovací manžety, které se standardně obepínají okolo horní části levé paže. Pokud je však tato část u pacienta poraněna, lze použít i předloktí nebo nohu. Manžeta by měla být zhruba o 20 % širší než průměr použité části končetiny. Příliš malé manžety vedou k nadhodnocení krevního tlaku a naopak [17]. Manžeta se pak nafoukne na tlak vyšší, než je systolický tlak v tepně. Při dosažení tohoto tlakového bodu stěny tepny brání průtoku krve. Poté se manžeta vypustí pod systolický tlak, čímž se obnoví průtok krve [15].

Pro měření srdeční frekvence se používají dva druhy zařízení. První typ je asi obecně známější elektrokardiograf (EKG) a funguje tak, že zaznamenává elektrickou aktivitu generovanou depolarizacemi srdečního svalu (negativní změna elektrického náboje), která se šíří jako pulzující elektrické vlny směrem ke kůži. Ačkoli je množství elektrické energie ve skutečnosti velmi malé, lze ji spolehlivě zachytit pomocí EKG elektrod přiložených ke kůži (v µV) [18]. Průběh vln snímá monitor s funkcí detekce elektrického proudu. Druhým typem je tzv. fotopletysmograf (PPG) fungující na základě odrazu vysílaného infračerveného světla. Podle množství odraženého světla se pozná rozšíření tepen. Možné je tímto typem sledovat i hladinu kyslíku v krvi [19].

Kapitola 2

Zjišťování duševního stavu v technických systémech

Spousta pracovních pozic v dnešním světě je založena na interakci mezi člověkem (operátorem) a strojem. V případech, kdy má člověk velkou zodpovědnost, může dojít ke katastrofě v důsledku jeho selhání. To už svět z historie zná, ať už je řeč o černobylské jaderné elektrárně nebo třeba bhópálské chemické továrně. Z toho důvodu jsou dnes silné tendence redukovat nebezpečí tím, že bude interakce pro člověka přiblížena jeho momentálním schopnostem a dovednostem. Jelikož je poměrně snadné popsat chování stroje, je důležité správně popsat i chování člověka. To však není zdaleka tak jednoduché, poněvadž u člověka činí velký vliv na výkon i jeho psychický stav. Spolehlivost systémů člověk – stroj s uvažováním vlivů prostředí lze popisovat mj. pomocí fuzzy modelů. Pravděpodobnostní analýza spolehlivosti používá základní intenzitu poruch pro stroj a základní intenzitu chyb pro člověka, jejich závislosti na vlivech prostředí se zjišťují empiricky [20]. Pomocí fuzzy modelů umíme zmíněné závislosti vyhodnotit kvalitativně. Spolehlivost stroje a spolehlivost člověka se vyjadřuje na základě odhadů *základní intenzity poruch*, respektive *základní intenzity chyb*. V obou případech jsou to ještě indikátory spolehlivosti systému. Mezi tyto indikátory se řadí například MTTF (Mean Time To Failure) nebo MTBF (Mean Time Between Failures). Střední doba do vzniku poruchy (MTTF) vyjadřuje statickou spolehlivost majetku nebo zařízení, které se neopravuje. Střední doba mezi opravami (MTBF) je obdoba předchozího, ale počítá se s opravami. Tyto indikátory závisí na teplotě okolí, environmentálních podmínkách, četnosti pracovních cyklů a zatěžovacím profilu zařízení [21]. Dále existuje ještě například střední doba do obnovení MTTR (Mean Time To Repair), jež vyjadřuje průměrný čas nečinnosti, než dojde k opravě nebo výměně (např. baterie v zařízení). Spolehlivost člověka zase ovlivňuje únava, stres, kvalifikace nebo prostředí, které může na psychiku působit znepokojivě nebo ho může nějakým způsobem rozptylovat. Zavedením lingvistických proměnných pro uvedené veličiny a použitím fuzzy pravidel lze vytvořit model respektující uvažované činitele [20].

2.1 BIOCYS

V osmdesátých letech se jako jedni z prvních pokusili o inteligentní rozhraní člověk - stroj vědci z japonského Kjóta. Nejdříve byly vytvořeny experimentální studie popisu myšlenkových projevů člověka na základě fyziologických měření jeho základ-

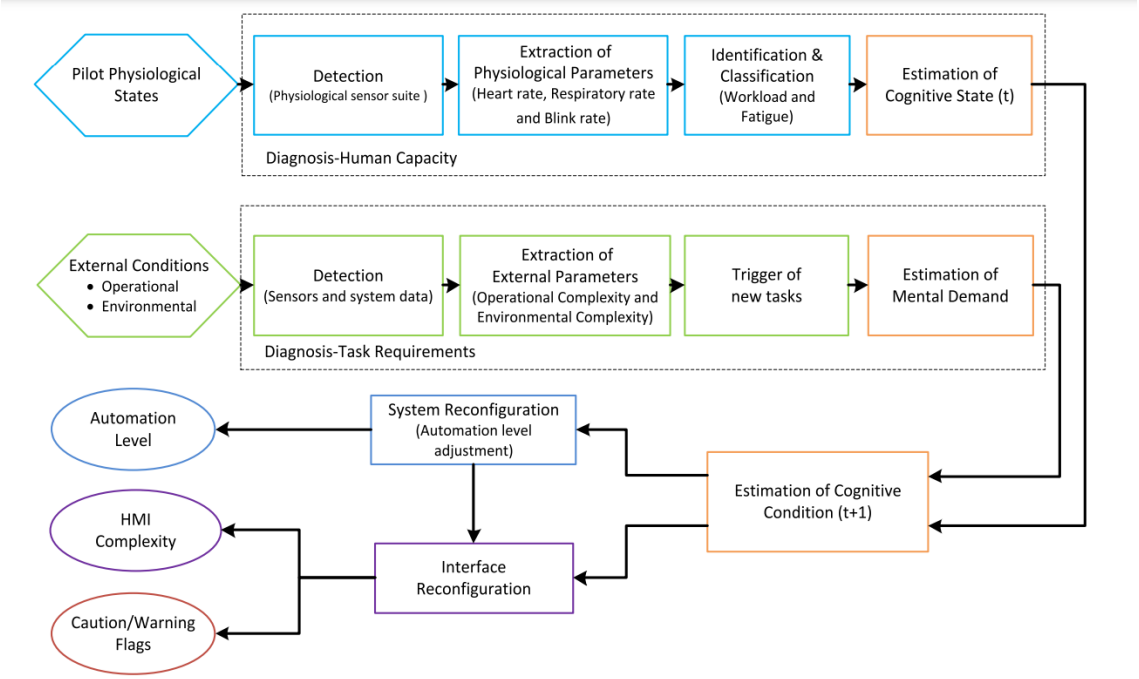
ních psychosomatických parametrů [22]. Vytvořili k tomu biokybernetický analyzační systém (BIOCYbernetic analysis System), který slouží k získávání i zpracování fyziologických dat. Nezbytnou součást celého systému tvoří expertní systém NUROBIO, podílející se velkou měrou na analýze poznávacích stavů operátora. Kromě něj se celý BIOCYS skládá ze systému určeného pro tvorbu modelových nepříjemných situací, do nichž se může operátor během obsluhy stroje dostat. Dále systémy pro zpracování řeči a naměřených dat z fyziologických senzorů. Systém vyhodnocoval celkem 6 fyziologických veličin, na jejichž základě byly vyhodnocovány poznávací stavy operátora. Měřila se velikost oční zřítelnice vyjadřující míru zájmu o stimulační podnět, frekvence mrkání vypovídající o zásobě nových informací v rámci podnětu, rychlost pohybu očí, jež vypovídá o schopnosti udržení pozornosti. Další veličinou byl kožní potenciál, určující, jaké emoce operátor prožívá, srdeční frekvence vypovídající o stupni fyzické nebo psychické zátěže a dýchací frekvence, jež odráží emoční aspekt poznávacího stavu [22]. Na základě těchto fyziologických dat expertní systém NUROBIO určuje v časovém intervalu tří vteřin jeden ze sedmi předpokládaných poznávacích (kognitivních) stavů operátora, a sice získávání informace, zapamatování si zjištěných informací a identifikace stavu, rozvaha o organizaci řešení, zpracování v paměti, hledání klíče k řešení, změny v postupu řešení a zmatek [23].

Samozřejmě by takový systém neměl velký smysl, kdyby kromě dynamického zjišťování stavu operátora neposkytoval i dynamickou zpětnou vazbu. Jelikož se s takovými systémy počítá zejména do prostředí, kde se chyba operátora neodpouští a může být fatální (velín jaderné elektrárny) obsahuje systém i tři možnosti dynamické zpětné vazby. První možností je poradní, jež pomáhá operátorovi při záchvatu paniky. Systém má v této chvíli za úkol posílat smysluplné rady a pomoct operátorovi překonat těžkou chvíli. Druhá možnost zpětné vazby je dohlížecí, která se uplatňuje, když u operátora dochází ke ztrátě pozornosti. Třetí volba je vlastní adaptivní zpětná vazba zajišťující přísun příslušného množství informace v závislosti na aktuálním stavu operátora [22].

2.2 CPAI

S exponenciálním rozvojem letecké dopravy strmě stoupá poptávka po pilotech. Těch se však v odvětví nepohybuje tolik, aby byla poptávka naplněna, a tak se vymýšlejí způsoby, jak nastalý problém vyřešit. Objevují se tzv. Single-Pilot Operatinos (SPO) tendence, kdy se na palubě letadla nachází pouze jeden pilot. Udává se, že zhruba do 5 let by se tato myšlenka mohla začít celkem často objevovat v praxi [24]. Jelikož je ale pilotování letadla náročná činnost, kde se chyby rovněž neodpouští, protože vedou ke ztrátám mnoha životů, je potřeba nějak zajistit bezpečnost tohoto trendu. Až 70 % leteckých havárií je způsobených právě lidskou chybou [25]. Proto probíhá v leteckém odvětví velký vývoj technologií, podobných jako je výše popsaný BIOCYS, které by výrazně pomáhaly pilotům případným chybám předejít. Jednou takovou technologií je CPAI (Cognitive Pilot-Aircraft Interface). Pracovní proces systému CPAI je rozdělen do tří fází. Nejdříve je snímán fyziologický stav pilota několika možnými senzory. Ty jsou součástí speciálního obleku se senzory PPG měřícími rozšíření tepen, EKG pro měření srdeční frekvence, senzory pro měření krevního tlaku, tělesné teploty a kožního odporu. Pilot je v kokpitu snímán také kamerou zabírající jeho tvář a upírající pozornost na jeho počínání. Také se může

starat i o eye tracking a s ním spojenou frekvenci mrkání, velikost zřítelnice apod., pokud nejsou k tomuto účelu určeny například speciální brýle. Pomocí fNIRS nebo EEG čelenek a čepic měří systém CPAI aktivitu mozku [26]. Po této fázi následuje fáze vyhodnocování psychického stavu pilota. Poslední fáze je rekonfigurace systému dle současného stavu pilota, podobně jako je tomu u systému BIOCYS. Varovné a výstražné symboly jsou rozsvíceny, pokud jsou zjištěny fyziologické stavy pilota v nepřijatelném rozsahu [25]. Na obrázku níže je vidět koncepční pracovní postup.



Obrázek 2.1: Koncepční pracovní postup CPAI [25]

2.3 Další kontrolní principy

V dnešní době, kdy za volant aut usedají starší a starší řidiči, zažívá velký boom monitorování životních funkcí řidiče, což by mohlo vést k redukci automobilových havárií. Ve společném projektu odborníků z BMW a pracovníků Technické univerzity v Mnichově se v předchozím desetiletí podařilo vyvinout senzorovu jednotku integrovanou do volantu měřící parametry životních funkcí řidiče, jako např. srdeční frekvenci, elektrický odpor pokožky a sycení krve kyslíkem [27]. Tato jednotka byla jako první vhodná pro sériovou výrobu a integraci do vozu a řidiče při jízdě vůbec neomezuje. Hlavní součástí zařízení k měření základních životních parametrů řidiče automobilu během jízdy jsou dva dotykové senzory běžně dostupné na trhu [27]. První z nich slouží k měření srdeční frekvence a okysličení krve a funguje, podobně jako ten použitý v tomto projektu, na bázi odraženého světla. Světlo, které vysílá je infračervené. Druhý senzor měří elektrický odpor kůže. Senzory posílají bezdrátově hodnoty do mikrořadiče, který je vyhodnocuje a je schopen určit stav řidiče. Umí zjistit nadměrný stres, nadkritickou hodnotu tlaku, řidič si při přestávce v řízení může udělat i menší test sám. Systém by dále mohl umět ztišit sám hudbu, zablokovat hovory nebo při případném zjištění nestandardního zdravotního stavu řidiče snížit rychlost vozidla či zapnout výstražná znamení.

Dále stav řidiče mohou snímat i kamery, které kontrolují pohyby očí a míru jejich otevření. Zkoumají tak, zda řidič není příliš unavený.

V rámci kontroly operátora jakéhokoli řízení se dnes hojně využívá i poměrně jednoduché „tlačítko mrtvého muže“ (DMS - Dead Man Switch). Tlačítko mrtvého muže, nebo také tlačítko bdělosti, funguje tak, že pokud operátor nevykoná nějakou činnost, kterou dá najevo, že je plně při vědomí, stroj se zastaví, chod se přeruší. Tlačítko může sepnout okamžitě, například v newyorském metru je páčka, kterou pokud strojvedoucí pustí, vlak se ihned zastaví. Tlačítko však také nemusí sepnout okamžitě, příkladem může být systém kontroly bdělosti v autopilotu letadla. Pilot si může odskočit na toaletu a v tu chvíli by nebylo dobré, aby systémy v letadle začaly jednat samy. To samé se děje v tramvaji, kde řidič zmáčkne tlačítko a má chvíli na to, než ho musí zmáčknout znovu. Mnohé DMS také fungují tak, že vyžadují, aby se operátor dotýkal více bodů na stroji, aby jej mohl ovládat. Princip spočívá v tom, že pokud máte obě ruce na spínačích, ani jedna z nich nemůže být uvnitř nebezpečné části stroje [28].

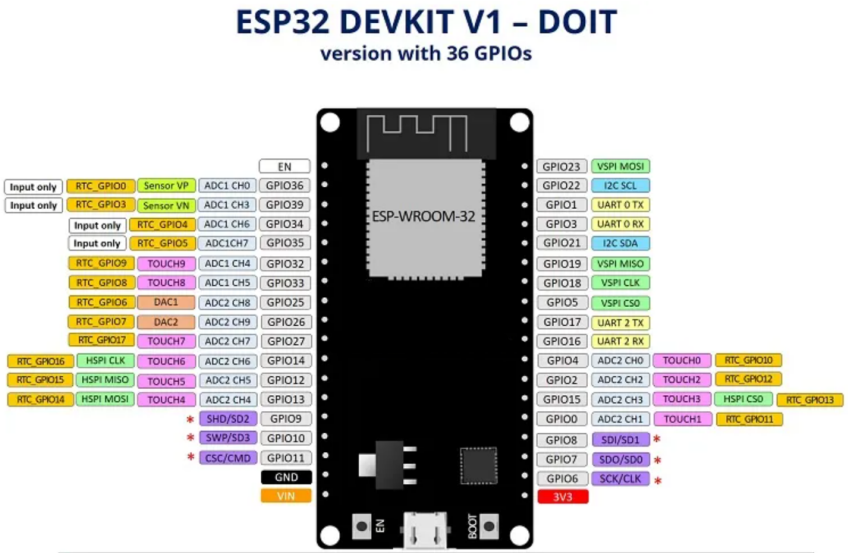
Nabízí se samozřejmě otázka, proč nenechat stroje v době takového technického pokroku pracovat sami, ale je spousta argumentů, proč to nedělat. V takovém případě by schopnosti operátora naprosto degradovaly. Operátor by ztratil veškerou chuť a vůli o systém a stroj pečovat a při případné poruše by neměl šanci uvést stroj opět do chodu.

Kapitola 3

Dostupné řídící zařízení

V tomto projektu byla pro výpočty použita vývojová deska ESP32, ale existují i další hojně používané. ESP32 je označení holého čipu, použitá vývojová deska se nazývá ESP32 DEVKIT DOIT, pro zkrácení ale bude používáno označení ESP32. Vývojové desky se dodávají se všemi potřebnými obvody pro napájení a programování čipu, jeho připojení k počítači a piny pro připojení periferií [29].

ESP32 je výkonný, levný a nízkoenergetický modul s možností připojení k WiFi a zařízením s Bluetooth, a proto se hodí pro celou škálu bezdrátových aplikací. Disponuje dvaatřicetibitovým LX6 mikroprocesorem Tensilica Xtensa Dual-Core se SRAM pamětí o velikosti 512 kB. Existuje několik typů vývojových desek ESP32 s různými parametry. Samotný čip ESP32 obsahuje 48 pinů, ne všechny však bývají dostupné pro připojení periferií. Deska ESP32 DEVKIT DOIT obsahuje 36 GPIO pinů, ke kterým můžeme periferie připojit. Piny 3V3, GND a VIN slouží k napájení periferií, pokud je deska napájena ze zdroje přes mikro USB port. Desku je možné napájet přes piny samotné, pokud není napájena přes port. ESP32 má velkou výhodu ve své multiplexnosti, protože o tom, jestli bude pin fungovat jako UART (univerzální asynchronní přijímač-vysílač), SPI (sériové periferní rozhraní) nebo I2C (sběrnice), rozhoduje programátor kódu. Samozřejmě má ESP32 defaultní nastavení pinů.



Obrázek 3.1: ESP32 DEVKIT DOIT pinout [29]

Dále je možné použít některé z pinů jako digitálně-analogové (DAC) nebo

analogově-digitální (ADC) převodníky. Deska dále obsahuje tlačítka BOOT a RESET pro restart desky, integrovanou anténu pro WiFi připojení, LED diody (červená svítí při napájení desky, modrá může být použita například pro signalizaci, že je deska připojena k WiFi, apod.) a již zmíněný mikro USB port pro napájení nebo import kódu z počítače.

3.1 Další vývojové desky

3.1.1 ESP8266

Vývojové desky s modulem ESP8266 jsou vlastně předchůdci desek s modulem ESP32. Oproti ESP32 mají pouze jednojádrový mikroprocesor, paměť SRAM má kapacitu pouze 160 kB. Dále mají pomalejší WiFi připojení, Bluetooth u nich není podporováno vůbec. V počtu GPIO pinů se se svým počtem 17 pinů nemohou rovnat generaci ESP32. Byly představeny o dva roky dříve v roce 2014 a jejich cena se pohybuje průměrně zhruba na polovině svých mladších příbuzných, tedy zhruba na 70 Kč.

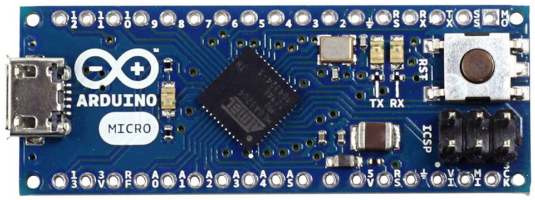
3.1.2 Arduino

Existuje velké množství modelů desek Arduino, jejich vývoj započal v roce 2005. U každého typu desek Arduino lze nalézt procesor od firmy Atmel a další elektronické komponenty. Další společná vlastnost je modrá barva.

Arduino Mini je nejmenší oficiální verze Arduina. Za takto okleštěnou velikost bylo nutné zaplatit daň v podobě nepřítomnosti USB portu, a proto je při jeho programování nutné použít externí USB 2 Serial převodník [30]. Běží na procesoru ATmega328, což pro něj znamená, že v rychlosti, narozdíl od velikosti, nezaostává.

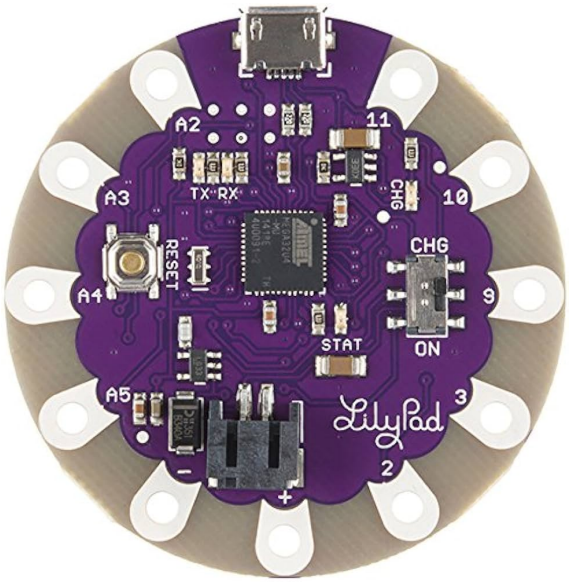
Podobně je na tom s velikostí Arduino Nano, ale jelikož už mikro USB port obsahuje, je přece jen o kousek větší a zároveň není už žádný externí převodník potřeba. Jinak jsou jejich vlastnosti téměř identické.

Pokud by si někdo chtěl vytvořit vlastní herní ovladač, zvolil by desku Arduino Micro. Obsahuje totiž nestandardní typ čipu ATmega32u4, který se pro počítač může tvářit jako myš nebo klávesnice a je schopen vyslat příkazy podobné stisknutí klávesy nebo posunutí myši [30]. Ostatní desky by to zvládly také, ale bylo by nutné přepsat kód převodníku v čipu.



Obrázek 3.2: Arduino Micro [30]

S velmi zajímavou inovací se Arduino ukázalo v roce 2007, kdy přišlo na scénu LilyPad Arduino. Vzhledem se ostatním Arduino deskám vymyká, protože je určená k našití na oblek a vytvoření například blinkrů integrovaných v oděvu. Pro vášnivé cyklisty tedy něco, co by v jejich výbavě nemělo chybět.



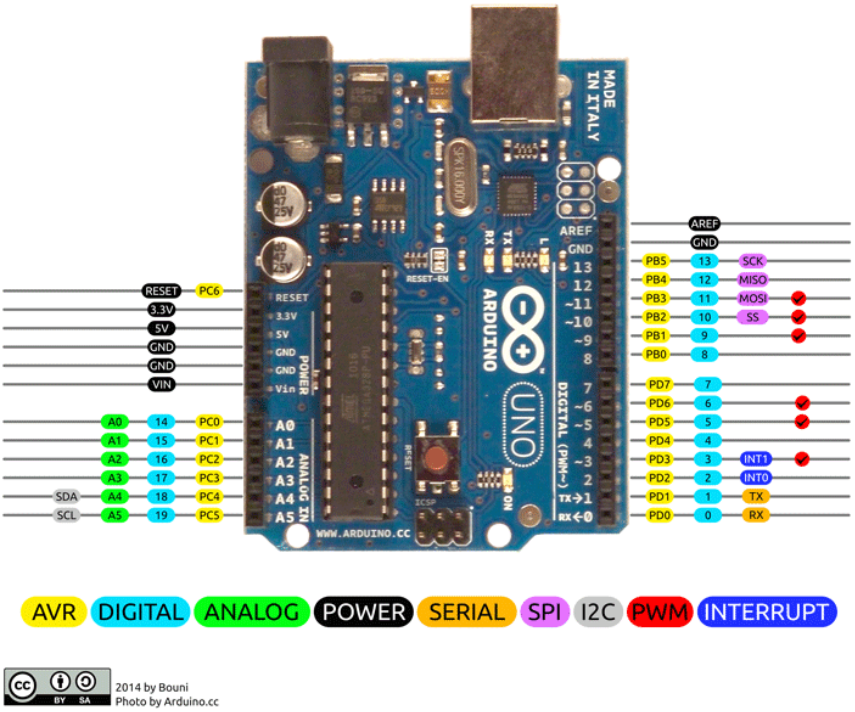
Obrázek 3.3: LilyPad Arduino [30]

Následující věty budou věnovány nejpopulárnější vývojové desce Arduino, kterou je Arduino Uno. Zajímavostí je, že společnost Espressif, která stojí za vznikem ESP32 se zabývá návrhem softwaru, který zajišťuje kompatibilitu Arduina Una a ESP32, i když podobnostní rozdíly jsou mezi těmito deskami obrovské. Malou odbočkou bych ještě rád zmínil, že největší podobnost desce ESP32 představuje Arduino Zero nebo Arduino 101, které například dokáží komunikovat přes WiFi a Bluetooth.

Ale zpět k Unu. Arduino Uno je deska založená na mikrokontroléru typu ATmega328P, jehož paměť SRAM má pouze 2 kB, což je pouze zlomek toho, co má ESP32. Uno bylo společností Arduino.cc vyvinuto v roce 2010. Od té doby prošlo několika inovacemi a upgrady. Deska obsahuje 20 vstupních/výstupních pinů pro připojení periferií. Z toho je 14 pinů digitálních. Šest z těchto pinů může fungovat jako PWM (Pulse Width Modulation) výstupy. PWM se používá například pro regulaci jasu LED diod nebo řízení rychlosti motoru. Dalších 6 pinů je pro vstupy nebo výstupy analogové. Uno dále disponuje USB portem typu B. Port má velké uplatnění, neboť se přes něj buď nahrává kód, anebo se přes něj Arduino dá napájet. Dále se dá napájet přes běžný barelový konektor, který přijímá zdroje v rozmezí 6 - 20 V. Optimální napájecí napětí je však mezi 7 - 12 V. Samotné pracovní napětí, které Uno poskytuje je 5 V. Na deskách Arduino se nejčastěji objevuje 16 MHz krystalický oscilátor, který umožňuje vykonávat časově závislé funkce. Nejinak je tomu v tomto případě.

V porovnání s deskou ESP32 je Arduino Uno prostší zařízení, jednodušší na programování. Není zde tak vysoké riziko pokažení během zapojování. Je robustnější, takže se s ním lépe manipuluje. Na druhou stranu díky větší paměti a možnosti připojení Bluetooth a WiFi dovoluje ESP32 pracovat na komplexnějších projektech. Je navíc levnější, takže se dá říct, že za méně peněz je možné dostat více možností.

Arduino Uno má své velmi blízké příbuzné. Například Arduino Ethernet má místo USB portu Ethernet port pro připojení k síti a slot pro microSD kartu. Arduino Bluetooth zase nahrazuje USB port Bluetooth modulem pro zajištění dalšího typu bezdrátové komunikace. Arduino Pro postrádá USB port úplně a pro jeho programování je nutný u Arduina Mini zmíněný externí převodník. Proto se hodí spíše k pevnému zabudování do projektu [30]. Všechny v tomto odstavci zmíněné desky jsou až na své popsané rozdíly stejné jako Arduino Uno.



Obrázek 3.4: Arduino Uno pinout [31]

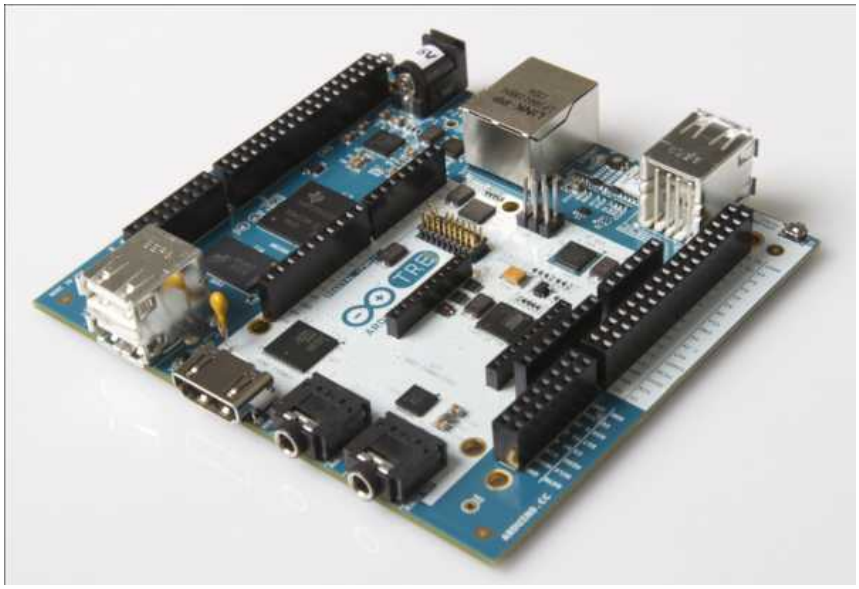
Dále existují desky jako Arduino Leonardo, Arduino Yún, které jsou designově také podobné Unu, ale používají jiné čipy. U Leonarda je to stejný čip, jako používá Arduino Micro, tedy ATmega32u4. Arduino Yún je pak ještě specifičtější, poněvadž používá čipy 2. Jednak také ATmega32u4, jednak Atheros AR9331 určený pro běh odlehčené verze Linuxu. Ve výbavě je softwarový bridge (prostředník, most), který zajišťuje komunikaci mezi oběma čipy [30].

Následují dva zástupci, kteří vznikli rozšířením desky Arduino Uno, a sice Arduino Mega2560, a ještě výkonnější Arduino Due s čipem Atmel SAM3X8E. Má taktovací frekvenci 84 MHz. Celkově zvětšení desek znamená prostor pro výkonnější čipy a další volné piny.

Přecházím ke kategorii hybridních desek Arduino. Arduino Esplora je speciální typ desky, který je ideální pro vytvoření vlastní herní konzole. Komunikuje rovněž pomocí čipu ATmega32u4 a obsahuje joystick, piny pro připojení LCD displeje, potenciometr atd. Prostě komponenty potřebné k vytvoření správného herního zážitku. Další hybridní deska s čipem ATmega32u4 a integrovaným kompasem se nazývá Arduino Robot. Celkem nepřekvapivě se hodí k vytvoření robota.

Ze série těch velmi výkonných desek je záhodno zmínit ještě Arduino Intel Galileo,

která vznikla ve spolupráci s firmou na výrobu procesorů Intel. Celkem nepřekvapivě tak běží na čipu Intel® Quark SoC X1000, což je 32bitový procesor s frekvencí 400 MHz [30].
Arduino Tre je nejvýkonnější ze všech Arduino desek, podobně jako Yún obsahuje 2 čipy a svou rychlostí a hardwarovými možnostmi dokáže konkurovat i minipočítačům Raspberry Pi.



Obrázek 3.5: Arduino Tre [30]

3.1.3 STM32

Stejně jako v případě desek Arduino, i v případě vývojových desek STM32 od společnosti STMicroelectronics existuje široká škála modelů, které nabízejí různé možnosti pro vývoj embedded systémů. Od roku 2005, kdy začal vývoj desek Arduino, se také objevila vývojová deska STM32, která se rovněž stala populární díky svým vlastnostem a výkonu.

Desek STM32 je podobně jako desek Arduino nepřeberné množství a existují modely s podobnými vlastnostmi. Vývojové desky STM32 však bývají výkonnější a sofistikovanější, jejich paměť SRAM bývá řádově vyšší než desek Arduino. Běží na procesorech ARM Cortex. I když jsou k začátečníkům, co se programování a zapojování týče, poměrně přívětivé, v tomto ohledu se Arduinu nevyrovnají.

S vývojovými deskami se v prvním desetiletí 21. století lidově řečeno roztrhl pytel, a tak není divu, že se objevily i další vývojové desky jako 8bit nebo ARM. V podobných, ale ještě komplexnějších projektech (ostatně i v dalším rozšíření tohoto projektu) se běžně používají moduly s názvem Raspberry Pi.

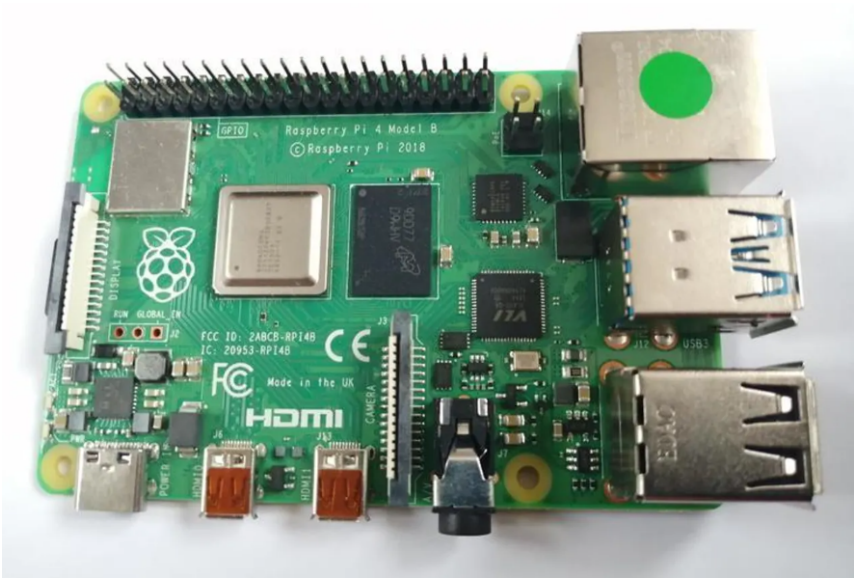
3.2 Raspberry Pi

V tomto projektu slouží mikropočítač Raspberry Pi 4B jako broker MQTT komunikace (vysvětleno v následující kapitole) a zároveň jako médium pro tvorbu grafického rozhraní. S tímto počítačem, jehož cena startuje zhruba od 1000 Kč v

závislosti na velikosti operační paměti, se dají vytvořit už velmi komplexní projekty v rámci automatizace. Aby byl plně nezávislý na jiném zařízení, je k němu třeba nabíječky, myši, klávesnice, micro HDMI kabelu, monitoru a micro SD karty, jež bude obsahovat operační systém Raspbian. Čtvrtá řada mikropočítačů Raspberry Pi má operační paměť RAM od 1 do 8 GB, taktovací frekvence procesoru je 1,5 GHz. To je zhruba o 1 GHz méně než mají dnes běžně dostupné procesory ve stolních počítačích. Je ale třeba vzít v potaz, že se čtvrtá řada těchto zařízení vyrábí už pět let. Obsahuje USB porty generace 3.0, pro nabíjení má USB-C port. Je možné připojení jak k Bluetooth, tak k WiFi (2,4/5 GHz), součástí je i Ethernet. Raspberry také disponuje 40 GPIO piny. Zvládne dekódovat videa se 4K rozlišením. Na desce je zabudováno 5 LED diod. Červená svítí při napájení, zelená signalizuje detekci vložené SD karty, jedna oranžová značí připojení přes Ethernet, další oranžová se rozsvítí při přenosu dat přes Ethernet, které přesáhne rychlost 100 Mbps. Poslední oranžová dioda v rozsvíceném stavu značí přenos dat oběma směry [32].

Modelů mikropočítačů Raspberr Pi už vyšlo celkem sedm. Kromě pěti modelů z hlavní řady vyšly ještě modely Zero a Pico. Po vydání modelu 4 následovaly už jen Pico a model 5. Oproti předchozím modelům má Raspberry Pi 4 poměrně zásadně větší operační paměť. Oproti první řadě vydané v roce 2012 má více GPIO pinů (Raspberry Pi verze A a Raspberry Pi verze B pouze 26 pinů). Poprvé bylo bezdrátově připojitelné k síti až Raspberry Pi 3 z roku 2016. Modely Zero a Pico zase neobsahují Ethernet a mají menší operační paměť než hlavní řada.

Raspberry Pi může mít v domácnosti řadu využití. Může sloužit jako běžný stolní počítač, někteří fajnšmekři si s jeho pomocí vytvářejí i retro herní konzole. V kombinaci s kamerami může vytvořit sledovací systém. Velkou roli hraje v oblasti IoT jako řídící jednotka různých systémů. V rámci automatizace ho můžeme najít například v programovatelných logických automatech (PLC). S jeho poměrně přívětivou pořizovací cenou se stal ve svém odvětví velmi populární komponentou.



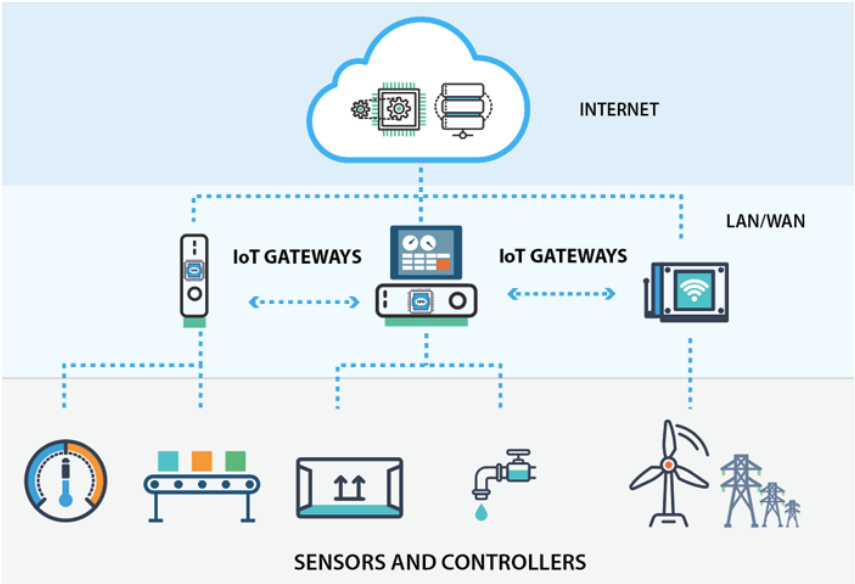
Obrázek 3.6: Raspberry Pi 4 Model B [33]

Kapitola 4

Bezdrátová komunikace

Ve fungování projektu hraje stěžejní roli protokol MQTT, pomocí kterého jsou vypočítané hodnoty z vývojové desky ESP32 posílány do počítače Raspberry Pi 4, který funguje jako tzv. MQTT broker.

MQTT protokol je dnes jedním z nejpoužívanějších protokolů internetu věcí (IoT). Internet věcí se v poslednch letech stal ve světě důležitým pojmem. Moderní firmy, které chtějí poskytovat zákazníkům co nejlepší služby a chtějí zlepšit efektivitu chodu, by se bez IoT v dnešní době neobešly. Stejně tak chytré domácnosti. IoT zprostředkovává tok dat sítí bez jakékoliv interakce člověka (kromě počátečního nastavení zařízení nebo úpravy jeho instrukcí), umožňuje neustálý monitoring aktuálních stavů, popřípadě příkazy k automatickému vykonání požadovaných úkonů. Jelikož IoT tvoří zařízení obsahující vestavěné systémy s přístupem k síti, není potřeba komunikující zařízení v IoT propojovat datovými kabely. Komunikace je proto dostupnější, jednodušší, méně cenově náročná a instalace takové sítě zařízení je snadnější, protože není nutné tahat kabely napříč dílnami nebo výrobními halami v rámci společností.



Obrázek 4.1: Schéma IoT [34]

Stejně jako v běžné internetové komunikaci, kdy se používají WiFi routery pro propojení lokální sítě s globální sítí poskytovanou některou z internetových

společností, se i v případě IoT používají centrální zařízení, které komunikaci mezi periferními zařízeními zprostředkovávají. Říká se jim IoT gateway neboli IoT brána. IoT brány přijímají signál ze zařízení (například senzorů) a posílají je v případě potřeby dál. Ne však všem zařízením, ale většinou těm, jimž je informace z přijatého signálu určena. Do této sítě se dá implementovat ještě další logika, jež může zpozdit signál k vypnutí světla při opouštění místnosti nebo je možné ji použít při postupném rozšiřování aplikace na přemapování aj.

4.1 MQTT protokol

Message Queuing Telemetry Transport je jedním z nejdůležitějších protokolů používaných v rámci IoT, původně byl navržen firmou IBM. Tento protokol využívá mezi zúčastněnými zařízeními vztahu Publish - Subscribe. Zařízení fungující jako publisher (často jimi bývají různé senzory) odesílají zprávy, přičemž se nestarají o to, co se s nimi dále děje. Tyto zprávy přijímá MQTT broker, který je uchovává, dokud se o ně nepřihlásí další zařízení. V tuto chvíli přichází na řadu zařízení typu subscriber, jenž do MQTT brokeru vyšle signál, které zprávy si vyžaduje. To je zprostředkováno způsobem, že všechny zprávy odeslané publisherem dostanou svoje téma (topic) a na základě tohoto tématu se o ně hlásí právě subscriber. Příklad zápisu tématu ze senzoru teploty na zahradě rodinného domu může být „rodinny_dum/zahrada/senzor_teploty“. Datová hodnota zprávy se pak nazývá payload. Payload může být různého datového typu. Omezen je pouze velikostí (dnes 256 MB), jinak záleží pouze na aplikaci, zda je v payloadu uložen textový řetězec, číselná hodnota, soubor s koncovkou JSON nebo jiné možnosti uložení dat.

Zajímavou funkci tvoří v protokolu tzv. Quality of Service. QoS se definuje jako dohoda mezi odesílatelem zprávy (publisherem) a příjemcem (subscriberem), která definuje úroveň záruky doručení pro konkrétní zprávu [35]. Úroveň záruky se pohybuje v množině 3 čísel - 0, 1, 2. Úroveň 0 znamená, že publisher pošle zprávu a více už se o ní nestará. Při úrovni 1 čeká publisher, zda byla zpráva doručena alespoň jednomu subscriberovi, přičemž nejvyšší úroveň zajišťuje, že publisher čeká, dokud mu není oznámeno, že o vše potřebné se přihlásil právě jeden subscriber. Ze všech nejčastější je základní nultá úroveň.

Komunikace startuje tím, že se zařízení připojí k brokeru pomocí TCP (Transmission Control Protocol) nejčastěji přes port 1883. Pokud chce uživatel zvýšit bezpečnost vzhledem k odposlouchávání toku dat, může použít protokol TLS (Transport Layer Security) na portu 8883. Když chce publisher odeslat nějakou zprávu, sám vytvoří téma. Pokud broker o daném tématu ještě neslyšel, tak ho nezahodí, ale automaticky téma založí. Subscriber musí téma znát, jinak by se o zprávu neměl jak přihlásit. Subscribeři mohou i mezi některá lomítka použít znaky jako je „+“ nebo „#“. Pokud subscriber vyžaduje téma „rodinny_dum/+/senzor_teploty“, dostane hodnoty ze všech senzorů teploty v rodinném domě. Pokud by se přihlásil o téma „rodinny_dum/zahrada/#“, dostane informace ze všech senzorů umístěných na zahradě rodinného domu. Existuje ještě jeden znak používaný v zápisu tématu, jímž je znak $. Tento znak je vyhrazen tzv. speciálnímu tématu. Často ho používá broker a slouží k ovlivnění chování klientů. Příkladem můžou být zprávy „$CONNECT“ pro navázání spojení mezi klientem a brokerem nebo „$PUBLISH“ sloužící k publikování zpráv do brokeru.

4.2 Další protokoly IoT

Původně se mezi zasvěcenými tak nějak počítalo s tím, že protokolem používaným pro komunikaci v rámci IoT bude obecně známý protokol HTTP (Hypertext Transfer Protocol). Jelikož byl již velmi rozšířen a celosvětově užíván, jeho implementace do IoT se zdála jako ideální řešení. Pro uchovávání dat mělo v praxi sloužit REST API. Při testování se však ukázalo, že pro komunikaci v rámci IoT je protokol HTTP nevhodný, protože si vyžaduje velkou režii provozu. Během komunikace se totiž objevuje velké množství informací, jež je velmi složité v rámci malých vestavěných zařízení vůbec zpracovat. Celý proces komunikace si potom vyžaduje velké množství času a energie, což se pro naplnění původního plánu s jednoduchostí a rychlostí IoT zdálo nepřípustné. Proto se od protokolu HTTP upustilo.

Ne však úplně. Na podobném principu byl vyvinut protokol CoAP (Constrained Application Protocol). Jedná se o „odlehčenou“ verzi HTTP, stejně jako zmíněný protokol využívá modelu request-response (požadavek-odpověď), ale tento model je v něm přizpůsoben potřebám omezené sítě. CoAP například podporuje koncept pozorování, který umožňuje zařízením přihlásit se k odběru změn ve zdrojích namísto neustálého dotazování na aktualizace. Tím se omezuje zbytečná výměna dat a šetří se energie [36]. Požadavky tradičních protokolů (jako HTTP) na neustálé připojení a správu relací mohou být pro zařízení s přerušovaným připojením a nízkými energetickými rezervami příliš zatěžující, což v konečném důsledku snižuje jejich provozní efektivitu [36]. HTTP hlavičky, které přenášejí doplňující informace ke komunikaci, jsou v CoAP nahrazeny binárními příznaky, TCP protokol je nahrazen protokolem UDP, čímž se zjednodušuje režie s navazováním a správou spojení, avšak zároveň UDP funguje bez záruky doručení zprávy. Protokol CoAP podporuje unicastovou i multicastovou komunikaci. Unicastová komunikace znamená komunikaci čistě mezi dvěma zařízeními. Oproti tomu znakem multicastové komunikace je možnost odeslat jednu a tu samou informaci ve stejný čas mnoha jiným zařízením. Zrovna to se může v IoT často hodit, pokud je například více akčních členů závislých na jednom senzoru [36]. Z porovnání CoAP s MQTT vychází, že oba protokoly jsou nenáročné, navržené pro omezené sítě, fungují na návrhovém modelu publish-subscribe a nabízejí více úrovní kvality služeb (QoS). Jedna z odlišností je ta, že protokol MQTT funguje podobně jako HTTP na protokolu TCP, ne UDP jako je tomu u protokolu CoAP. Dále je potřeba zmínit, že CoAP se hodí pro scénáře, kde je důležitá přímá interakce se zdroji a principy RESTful, například pro monitorování a ovládání zařízení IoT. Oproti tomu MQTT je ideální pro scénáře, kde zařízení potřebují efektivně publikovat data více odběratelům, jako jsou telemetrie v reálném čase, monitorování a aplikace pro vzdálené snímání [36]. Postupem času s růstem požadavků začíná být protokol CoAP rozšiřován a jeho výhoda oproti protokolu HTTP začíná zanikat.

Protokol AMQP (Advanced Message Queuing Protocol) funguje na velmi podobných principech jako výše vysvětlený protokol MQTT. Také zde existuje jeden centrální broker přijímající zprávy od publisherů a rozesílající je čekajícím subscriberům. Narozdíl od MQTT může AMQP fungovat i jako PPP (Point-to-Point) protokol. Ale zpět k architektuře typu publish-subscribe. Zprávy odeslané publisherem procházejí poštou (Exchange modul), jež zprávy roztřídí do front, odkud už si každý subscriber vybírá ty, které ho zajímají. AMQP také používá protokol TCP. Co může být jeho obrovskou nevýhodou v pomyslném souboji s MQTT, je fakt, že není možné s

jeho pomocí zjistit informace o stavu klienta. V rámci komunikace pomocí MQTT je běžné, že když dojde k odpojení klienta, uživateli přijde ihned zpráva. Maximální teoretická velikost zpráv se zde uvádí až 2 GB, maximální doporučená velikost je však 128 MB. Služba QoS má v případě AMQP o úroveň méně. Úroveň 0 je bez potvrzení doručení, úroveň 1 je s potvrzením. Vzhledem k tomu, že MQTT protokol byl vytvořen přímo pro IoT, což AMQP nebyl, je první jmenovaný v rámci IoT rozšířenější.

S protokolem XMPP (eXtensible Messaging and Presence Protocol) se dnes potká téměř každý. Byl vytvořen pro zprostředkování komunikace mezi uživateli přes textové a hlasové zprávy nebo videohovory na základě jazyka XML (eXtensible Markup Language). Funguje na architektuře klient-server. To znamená, že při odesílání zprávy prostřednictvím protokolu je zpráva nejprve odeslána na server, který ji následně přesměruje na správného klienta [37]. Server správného klienta bez problému pozná, protože každému klientovi k serveru připojeném je přirazen tzv. Jabber ID. Jedná se o unikátní identifikátor, který má tvar podobný emailovým adresám, a sice „user@server“, kde „user“ může být jméno uživatele klienta a „server“ značí klienta, ze kterého je odeslána zpráva. Pokud je klient ověřen a spojení je navázáno (používá se pojem „otevřené“) umožňuje protokol odesílat data mezi uzly libovolně. To je velký rozdíl oproti protokolu HTTP, kde je možné, aby server odeslal data ke klientovi pouze na základě klientova požadavku.

Jedním z mnoha dalších protokolů je DDS (Data Distribution Service). DDS byl navržen pro implementaci M2M komunikace (Machine-to-Machine) do IoT. Nepotřebuje tedy narozdíl od MQTT nebo AMQP centralizovaného brokera, který by řídil proces doručování zpráv [38]. Protokol DDS je jednoduchý a vhodný k rozšiřování, protože pracuje na multicastové úrovni komunikace se všemi zainteresovanými zařízeními a využívá principu plug-and-play. Tedy je schopen nově přidanému zařízení přidat buď roli publishera, nebo roli subscribera, takže je znám datový typ informací od zařízení a směr jejich toku. Pro bezpečnost komunikace užívá protokolů DTLS a SSL. Je však zde pár nevýhod, kvůli kterým se četností výskytu neřadí před MQTT nebo CoAP. Je poměrně náročný pro malá zařízení (klienty v rámci IoT). Ačkoliv narozdíl od XMPP je u něj možná služba QoS, k jejímu využití je třeba čisté komunikace pomocí DDS.

V rámci IoT je možné se setkat také s dalšími protokoly, jakou jsou Bluetooth, Zigbee, Z-Wave, 6LowPAN, Thread, WiFi nebo NFC.

Kapitola 5

Praktická část

5.1 Výroba senzoru měření EDA

Obrázek 5.1: Senzor měření EDA

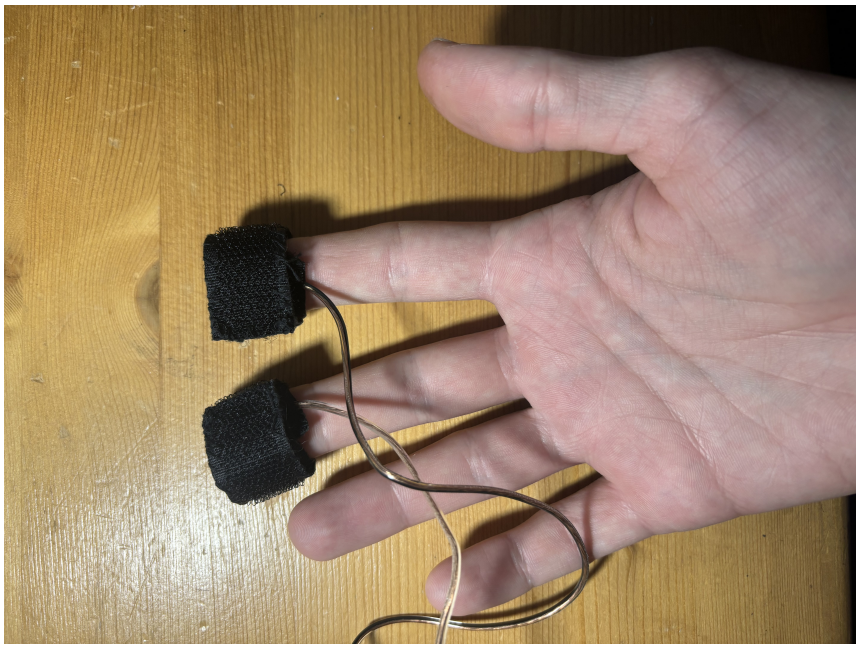
Jelikož jsem měl možnost vidět pár zařízení pro měření EDA, snažil jsem se držet fungujícího konceptu. Řešení je založeno v podstatě na schématu exosomatické metody na obrázku 1.1. Opatřil jsem si našívací stiskací knoflíky (patentky), neoboustranný suchý zip a měkké měděné lanko s izolací od staré elektroniky.

Patentky byly použity z důvodu potřeby větší hladké plochy vodivého materiálu. Tyto plochy se potom za pomoci suchého zipu připevní na bříška dvou prstů. Mechanismus je zapájen do vývojové desky ESP32, která do obvodu pustí stejnosměrné napětí. Lidská kůže potom funguje jako rezistor v obvodu.

Opatřil jsem si dvě zhruba čtyřiceticentimetrová vlákna měděného lanka. Po zahřátí hrotu pájecí stanice jsem vlákna napájel jedním koncem k patentkám.

Dále jsem rozdělil suchý zip, abych měl dva stejně dlouhé pruhy na dva prsty. Jelikož

byl suchý zip neoboustranný, byla sešita jedna podélná strana a před sešitím druhé strany jsem na kraji pásku vytvořil malou dírku pro secvaknutí dvou částí patentky k sobě. Když byla patentka usazená a držela na místě u kraje pásku ze suchého zipu, byla sešita i druhá podélná strana pásku. Tak došlo k zakrytí jedné strany patentky, a napájený spoj patentky a drátu byl zakryt, což redukuje možnost jeho vytrhnutí.



Obrázek 5.2: Aplikace senzoru na bříška prstů

Následovalo napájení druhých konců vláken na desku ESP32, ale o tom více v sekci *Zapojení senzorů*.

5.2 Výroba senzoru dýchání

Na výrobu druhého senzoru, tedy senzoru dýchání bylo v základu potřeba stejných věcí jako na senzor měření EDA. Stejně jako v prvním případě jsem si opatřil patentky, měkké lanko s izolací a neoboustranný suchý zip. Dále však bylo nutné sehnat gumovou textilii (šíře alespoň cca 7 milimetrů, délka cca 2 metry) a zažádat vedoucího této práce o kousek speciálního pružného vlákna (conductive rubber cord).

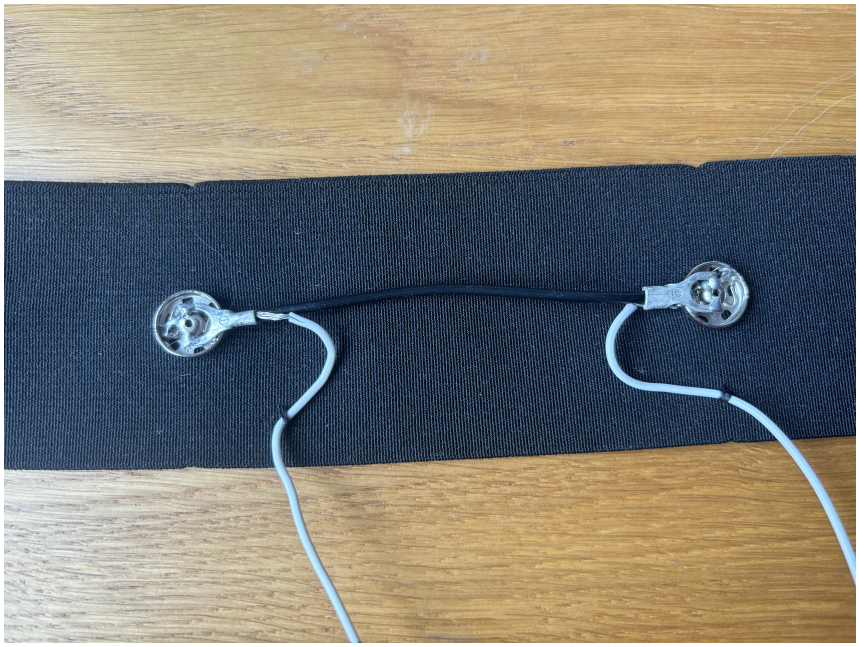
Conductive rubber cord je vlákno vyrobené z pryže, která je impregnována uhlíkovou černí. Vlákno je vodivé a měří se pomocí něj změna polohy. Vlákno totiž svým natahováním a stahováním mění odpor. Na trhu není příliš snadné k dostání a nejedná se o úplně nejlevnější záležitost, a proto byl při výrobě použit jen zhruba deseticentimetrový kousek.

Jako v případě s prvním senzorem jsem odstranil izolaci z konců vlákna a napájel konce na patentky.

Nachystal jsem si dvě kabelové izolované koncovky ve tvaru vidlice, vysunul jsem je za pomoci kleští z izolace a koncem s otvorem naklepal na hrot ostrého předmětu, abych si otvory rozevřel. Poté jsem do nich vložil oba konce vodivého pryžového

vlákna, nasadil je na lanko těsně u patentek a kleštěmi secvakl k sobě. Tím došlo k tomu, že jsem si vodivě propojil oba konce lanka. Aby se kabelové koncovky nemohly samovolně hýbat, poškodit lanko nebo pryžové vlákno napájel jsem jejich vidlicový konec k patentkám. Vidlice byly napájeny na zadní stranu patentek, aby se patentka dala secvaknout.

Gumový pás jsem si na dvou místech proděravěl. Vzdálenost děr je zhruba taková, aby pryžové vlákno nebylo příliš napnuté (při aplikaci by se mohlo přetrhnout) a zároveň, aby nebylo příliš volné (je potřeba, aby se při aplikaci natahovalo a stahovalo - nádech, výdech). Lanka vedoucí z obou patentek jsem kousek od nich stáhl pár stehy, abych ještě snížil riziko vytrhnutí lanka. Pás jsem dále pošil suchým zipem. Aby nedošlo k velké redukci napínání gumy, není pošitý po celé délce, ale pouze na čtyřech oblastech. Nejspíš to není zcela ideální řešení, lepší by bylo použít nějaký druh elastického suchého zipu, ale po vlastních zkušenostech vím, že není téměř nikde k sehnání v podobě, v jaké by bylo potřeba [39].



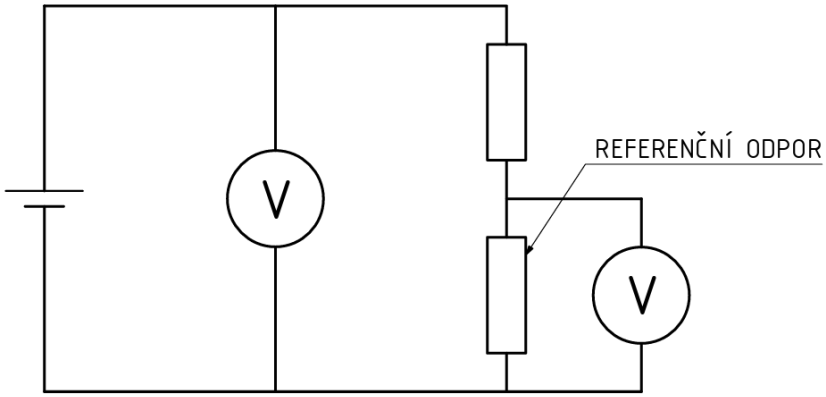
Obrázek 5.3: Detail stěžejní části senzoru dýchání

5.3 Zapojení senzorů

Zapojení senzoru pro měření EDA vypadá následovně. Z pinu D26, který může fungovat jako D/A převodníky, je vysílán spojitý průběh napětí. Pin D25 je vodivě propojen s pinem D35. Sem je zároveň přivedeno i první vlákno senzoru. Druhé vlákno je spojené s rezistorem a jsou společně napájené na pin D34. Druhý konec rezistoru je napájen na pin GND. Hodnota rezistoru je 0.5 MΩ. Tato hodnota není zcela náhodná, řádově se rovná skutečnému odporu kůže, tudíž dochází k menším výchylkám ve výpočtech.

Zapojení senzoru dýchání je vlastně úplně stejné, pouze došlo k zapojení na další vývojovou desku ESP32. Druhý rozdíl je, že rezistor má tentokráte hodnotu pouze 2 kΩ. Důvod, proč má tento rezistor 250krát menší hodnotu než v prvním případě,

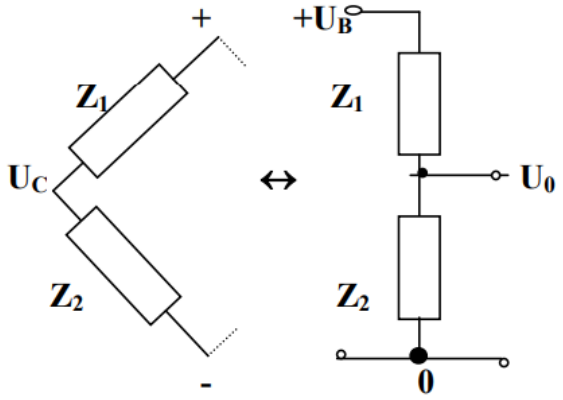
je prostý. Odpor 10 cm vodivého pryžového lanka je řádově v nižších jednotkách kΩ. Kdyby byl použit při zapojení rezistor o odporu tolikrát vyšším, docházelo by k výrazným výchylkám vypočtených hodnot. Skutečné zapojení obou desek je vidět na obrázcích 5.7 a 5.8.



Obrázek 5.4: Schéma zapojení

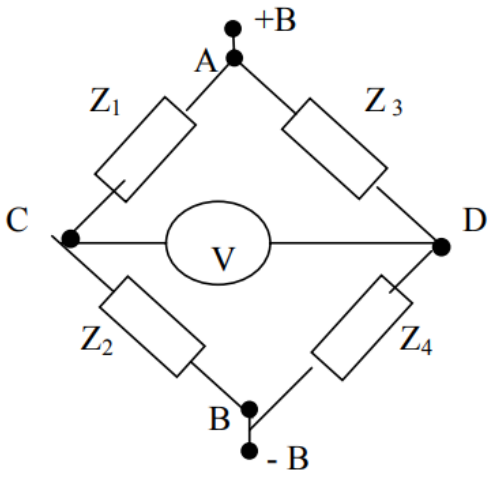
Zapojení má tvořit tzv. měřicí můstek. Při řešení můstku se vychází z řešení klasického děliče napětí [40]. Pomocí Kirchhoffových zákonů se určí:

$$[h]\frac{U_B}{U_0} = \frac{Z_1 + Z_2}{Z_2} \Rightarrow U_0 = \frac{Z_2}{Z_1 + Z_2}U_B \tag{5.1}$$



Obrázek 5.5: Schéma nahrazení děliče napětí [40]

Můstková metoda se začala používat, protože měřicí zařízení, která měla často velkou základní jednotku, měly ve svých aplikacích měřit změny velmi malé.

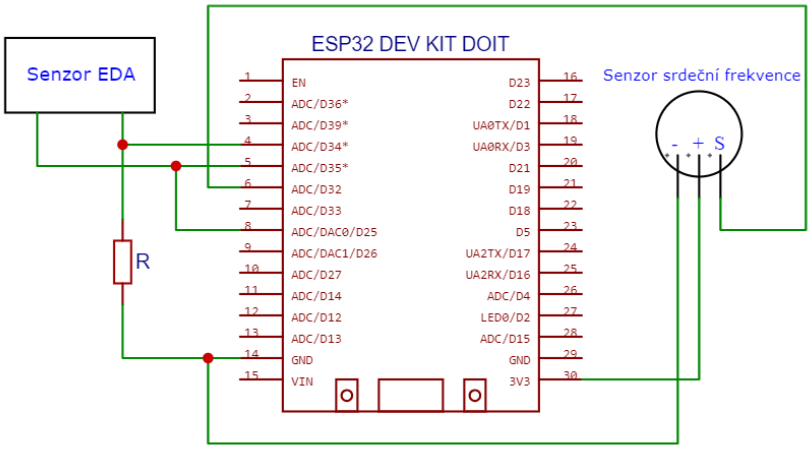


Obrázek 5.6: Zapojení měřicího můstku [40]

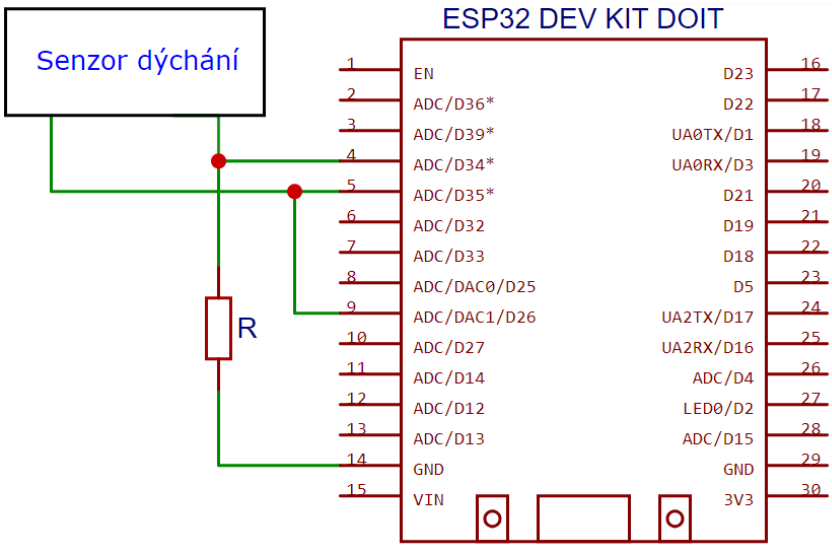
Můstkové zapojení, jak je znázorněno na obrázku se skládá ze čtyř odporů. Čtyři odpory můžou být tvořeny libovolnou kombinací R, L, C součástek. Měří se výstupní napětí mezi body C, D, jak je znázorněno na obrázku. Napětí mezi body B+ a B- je napájecí. Konstantní hodnoty odporů se odečtou, drobná změna napětí je způsobena změnou odporu na některé části můstku. Častou komponentou, která se do obvod přidává je zesilovač. Změny napětí jsou totiž pořád poměrně malé. Navíc zesilovač zmenšuje i výstupní impedanci. Standardizovaný výstup je buď napěťový s hodnotami od 0 do 10 V, který se používá při přenosu signálu na větší vzdálenost. Je ale náchylnější k rušení než proudový výstup, jehož standardizované hodnoty výstupu jsou 0 - 20 nebo 4 - 20 A.

Měřicí můstky jsou děleny podle počtu pevných známých odporů na čtvrtmůstek (1 pevný odpor ze 4), půlmůstek (2 pevné odpory ze 4) a můstek (žádný pevný odpor), dále podle použitého vstupního napětí na stejnosměrné a střídavé. Velký význam má v praxi dělení na můstky vyvážené a nevyvážené. Vyvážený můstek funguje tak, že se kompenzačním rezistorem reguluje výstupní napětí mezi body C a D na nulu. Tento typ se však v praxi nepoužívá zdaleka tolik jako můstek nevyvážený. V něm se měří rozvážení můstku a měřená veličina (odpor) je přímo úměrná výstupnímu napětí. Můstky se dále dělí také podle měřené veličiny, kterou může být kapacita (De Sautyho, Wienův nebo Scheringův můstek), malý odpor (Wheatstoneův můstek), velmi malý odpor (Thomsonův můstek) nebo například indukčnost (Owenův můstek) [41].

Senzor srdečního tepu je zapojen následovně. Pin, jenž je na senzoru označen „+“, je spojen s pinem *3V3* na desce ESP32. Pin senzoru s označením „-“ je přiveden do pinu *GND* a pin se symbolem „S“ musí být přiveden do jednoho z analogových pinů desky ESP32. V tomto případě je to pin s číslem 32.



Obrázek 5.7: Zapojení senzorů do první desky ESP32



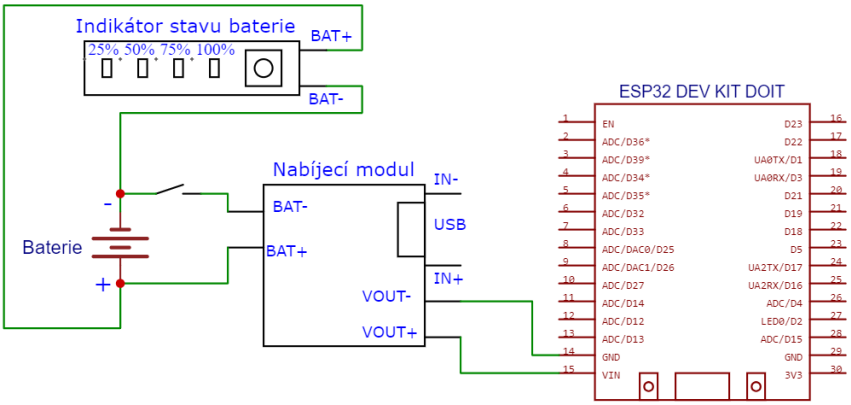
Obrázek 5.8: Zapojení senzoru dýchání do druhé desky ESP32

5.4 Napájení vývojových desek ESP32

Aby nemuselo být ESP32 neustále napájeno přes USB port a ten mohl sloužit pouze pro nahrávání aktualizovaných verzí kódu, bylo potřeba vymyslet nezávislé napájení. To jsem nakonec vyřešil použitím baterie LiPol (lithium-polymer) s kapacitou 1500 mAh a napětím 3,7 V a nabíjecím modulem lithiových baterií TP4056 s boostem MT3608, který převádí napětí do rozsahu 2-24 V, a tak se dají z baterie spolehlivě napájet další komponenty, které mají potřebné napětí v tomto intervalu. Nabíjecí modul zase sám o sobě kontroluje a ochraňuje baterii před přebitím a nešetrným nabíjením při podbití. Abych mohl napájet z baterie vývojové desky a zároveň mohl kontrolovat stav baterie, použil jsem LED indikátor stavu kapacity, který obsahuje 4 LED diody symbolizující 25 %, 50 %, 75 % a 100 % nabití baterie.

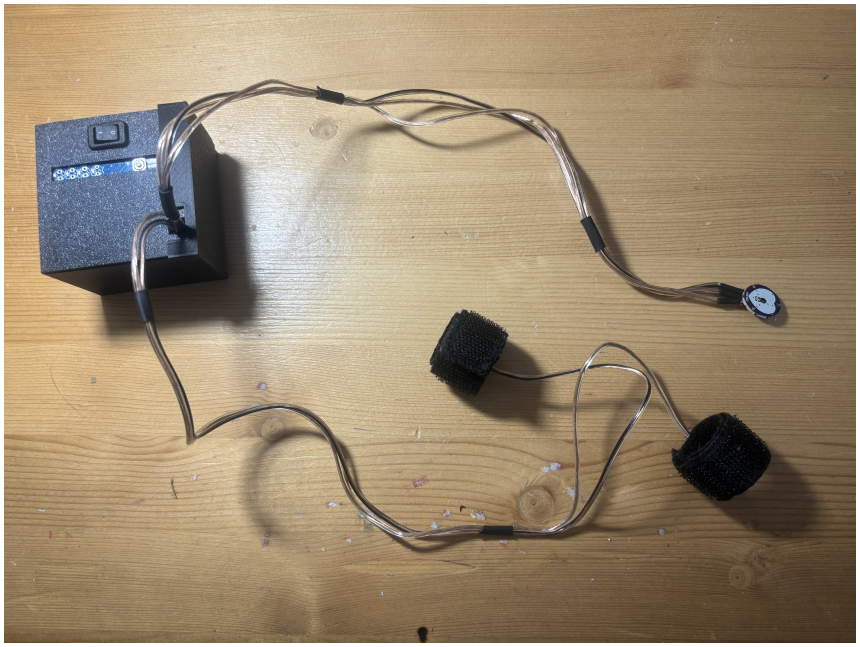
Zapojení pak vypadá takto. Bateriím jsem uřízl konektory, jež se nehodily mým záměrů a vzniklé konce jsem napájel na LED indikátor. Zároveň jsem ke zmíněným

koncům napájel další vodivá vlákna, která jsem přivedl na piny BAT+ a BAT- na nabíjecím modulu. Samozřejmě bylo nutné dodržet, aby kladný pól baterie pasoval ke kladnému pinu a záporný k zápornému. Mezi záporný pól baterie jsem ještě vložil kolébkový vypínač, tím zajistím možnost vypnutí napájení, která v poloze 0 na kolébkovém vypínači přestane obvod zásobovat napětím. Dalším kouskem vodivého vlákna jsem propojil pin VOUT- nabíjecího modulu s pinem GND na desce ESP32. Stejným způsobem také pin VOUT+ na modulu a VIN pin na ESP32.



Obrázek 5.9: Obvod pro napájení ESP32 a nabíjení baterie

Tímto způsobem je docíleno poměrně efektivního napájení vývojových desek a zároveň efektivní kontroly stavu baterií a jejich dobíjení. Bylo by však poměrně neefektní, kdyby se zařízení nějak nezakrytalo a mohlo by to způsobit i problémy s manipulací. Proto bylo nutné vytvořit si v programu Autodesk Inventor model krytu pro celý mechanismus. Tento model byl poté převeden na formát vhodný pro slicer (program převádějící vytvořené modely součástí na modely vhodné pro 3D tisk) a vytisknut na 3D tiskárně. Jak je vidět na obrázku níže, kryt má zespodu háčky, kterými se dá uchytit například za kus látky. V mém případě jsem si pořídil řemínek na ruku a mechanismus pro měření a vyhodnocování elektrodermální aktivity jsem nasadil na něj. Druhý mechanismus pro dýchání má své místo přímo na hrudním gumovém pásu, který je jeho součástí.



Obrázek 5.10: Finální podoba senzorů EDA a srdeční aktivity

Obrázek 5.11: Finální podoba senzoru dýchání

5.5 Komunikace přes MQTT

Po zapojení senzorů bylo načase přejít k programování a nastavování bezdrátové komunikace mezi vývojovými deskami ESP32 a počítačem Raspberry Pi 4. Níže je okomentovaný kód vytvořený v platformě Arduino IDE, který zajišťuje připojení desky ESP32 k WiFi a nastartování komunikace pomocí MQTT protokolu mezi deskami a právě počítačem Raspberry Pi 4. Dále kód obsahuje výpočtové vzorce pro potřebné hodnoty odporů, zabalení výsledných hodnot do souboru formátu JSON a odeslání (publishing) přes MQTT protokol. Zároveň nechávám vypočtené hodnoty

posílat na obrazovku přes sériový monitor, což může v nějakých situacích posloužit jako jednoduchý debugger, pokud by se někde ve výpočtu nacházela chyba. Sériový monitor ale funguje pouze, pokud je deska ESP32 připojena USB kabelem do počítače, ze kterého byl kód importován, a je zapnut program Arduino IDE. Ještě je dobré zmínit, že kód obsahuje funkci, která přiřadí desce ID a dokud se komunikace neuskuteční, každých 5 vteřin kontroluje její stav.

Řádek s heslem k WiFi ve skutečnosti vypadá jinak. Heslo musí být ve správném tvaru. Zde jsem ho kvůli bezpečnosti přepsal.

```
#include <ArduinoJson.h>
#include <ArduinoJson.hpp>
#include <WiFi.h>
#include <Wire.h>
#include <PubSubClient.h>

#define VOLTAGE 3.3 // maximum supply voltage
#define PINSCALE 4096.0 // ESP32 pins show values between 0 and 4096
#define DACGSR 25
#define GSRPIN1 35
#define GSRPIN2 34
#define RESISTORGSR 510000.0 // resistor value in Ω
#define PULSEPIN 36

const char* ssid      = "Pajik15";
const char* password  = "**********";
const char* mqtt_serv = "172.20.10.4";

WiFiClient espClient;
PubSubClient client(espClient);

long lastMsg = 0;
float gsr = 0.0; // galvanic skin response initiation
float pulse= 0.0; // pulse value initiation
int sumOfLoops = 0; // when this variable reach value of 10, sending average of 10 GSR values over MQTT
float arrayOfGsrValues[10]; // array for 10 GSR values to make an average
float arrayOfPulseValues[10]; // array for 10 pulse values to make an average

void setup() {
  delay(100);

  Serial.begin(115200);

  // Connect to WiFi
  Serial.println("");
  Serial.print("Connecting to: ");
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  while(WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
```

```
45
46    Serial.println("");
47    Serial.print("WiFi Connected, IP: ");
48    Serial.println(WiFi.localIP());
49
50    // Set MQTT Server details
51    client.setServer(mqtt_serv, 1883);
52  }
53
54  void loop() {
55
56    // Check if connected to MQTT Server
57    if(!client.connected()) {
58      reconnect();
59    }
60
61    StaticJsonDocument<40> doc;
62    char output[40];
63
64    // Delay between sending messages
65    long now = millis();
66    if(now - lastMsg > 10) {
67      lastMsg = now;
68
69      pulse = analogRead(PULSEPIN);
70
71      // Generate a sine wave
72      int value = 255;
73      dacWrite(DACGSR, value);
74
75      // GSR sensor; read the inputs on analog pins 34, 35:
76      int gsrSensorValue1 = analogRead(GSRPIN1);
77      int gsrSensorValue2 = analogRead(GSRPIN2);
78      // Convert the analog reading (which goes from 0 - 1023) to a
     voltage (0 - 3.3 V):
79      float gsrVoltage1 = gsrSensorValue1 * (VOLTAGE / PINSCALE);
80      float gsrVoltage2 = gsrSensorValue2 * (VOLTAGE / PINSCALE);
81      // Skin resistance calculation
82      float gsr = (gsrVoltage1 - gsrVoltage2)*RESISTORGSR/gsrVoltage2
     ;
83
84      // Adding values to arrays and increasing variable
85      arrayOfGsrValues[sumOfLoops] = gsr;
86      arrayOfPulseValues[sumOfLoops] = pulse;
87      sumOfLoops++;
88
89      // Average and entry of values
90      if(sumOfLoops == 10) {
91        float sumOfGsrValues = 0.0;
92        float sumOfPulseValues = 0.0;
93        for(int i = 0; i < sumOfLoops; i++) {
94          sumOfGsrValues = sumOfGsrValues + arrayOfGsrValues[i];
95          sumOfPulseValues = sumOfPulseValues + arrayOfPulseValues[i
     ];
96        }
97        float avgOfGsr = sumOfGsrValues / sumOfLoops;
98        float avgOfPulse = sumOfPulseValues / sumOfLoops;
99        if(isfinite(avgOfGsr)) { // Checking the finiteness
100         doc["g"] = avgOfGsr; // Adding average to an array of chars
```

```
      }
      if(isfinite(avgOfPulse)) {
        doc["p"] = avgOfPulse;
      }
      // Serialise JSON and send
      serializeJson(doc, output);
      Serial.println(output);
      client.publish("/esp/sensors/gsr", output);
      sumOfLoops = 0;
    }
  }
}

void reconnect() {
  while(!client.connected()) {
    Serial.print("MQTT not connected... Trying to connect... ");

    // Create client ID
    String clientId = "ESP32Client-";
    clientId += String(random(0xffff), HEX);

    // Attempt connection
    if(client.connect(clientId.c_str())) {
      Serial.println("Connected!");
    } else {
      Serial.print("Failed, code: ");
      Serial.println(client.state());
      delay(5000);
    }
  }
}
```

Zdrojový kód 5.1: Kód pro výpočet odporu kůže a srdeční frekvence

Kód nahraný do druhé desky následuje níže. Až na pár pozměněných jmen proměnných a pár konstant je stejný. Také je v něm absence řádků účastnících se výpočtu srdeční frekvence.

```
#include <ArduinoJson.h>
#include <ArduinoJson.hpp>
#include <WiFi.h>
#include <Wire.h>
#include <PubSubClient.h>

#define VOLTAGE 3.3 // maximum supply voltage
#define PINSCALE 4096.0 // ESP32 pins show values between 0 and
    4096
#define DACBR 26
#define BRPIN1 35
#define BRPIN2 34
#define RESISTORBR 2000.0 // resistor value in Ω

const char* ssid      = "Pajik15";
const char* password  = "kubajekokotscislem34";
const char* mqtt_serv = "172.20.10.4";

WiFiClient espClient;
PubSubClient client(espClient);

long lastMsg = 0;
```

```
22 float br = 0.0; // breathing resistance initiation
23 int sumOfLoops = 0; // when this variable reach value of 10,
       sending average of 10 breathing resistance values over MQTT
24 float arrayOfBrValues[10]; // array for 10 breathing resistance
       values to make average
25
26 void setup() {
27   delay(100);
28
29   Serial.begin(115200);
30
31   // Connect to WiFi
32   Serial.println("");
33   Serial.print("Connecting to: ");
34   Serial.println(ssid);
35
36   WiFi.begin(ssid, password);
37
38   while(WiFi.status() != WL_CONNECTED) {
39     delay(500);
40     Serial.print(".");
41   }
42
43   Serial.println("");
44   Serial.print("WiFi Connected, IP: ");
45   Serial.println(WiFi.localIP());
46
47   // Set MQTT Server details
48   client.setServer(mqtt_serv, 1883);
49 }
50
51 void loop() {
52
53   // Check if connected to MQTT Server
54   if(!client.connected()) {
55     reconnect();
56   }
57
58   StaticJsonDocument<40> doc;
59   char output[40];
60
61   // Delay between sending messages
62   long now = millis();
63   if(now - lastMsg > 10) {
64     lastMsg = now;
65
66     // Generate a sine wave
67     int value = 255;
68     dacWrite(DACBR, value);
69
70     // Breathing resistance sensor; read the inputs on analog pins
       34, 35:
71     int brSensorValue1 = analogRead(BRPIN1);
72     int brSensorValue2 = analogRead(BRPIN2);
73     // Convert the analog reading (which goes from 0 - 1023) to a
       voltage (0 - 3.3 V):
74     float brVoltage1 = brSensorValue1 * (VOLTAGE / PINSCALE);
75     float brVoltage2 = brSensorValue2 * (VOLTAGE / PINSCALE);
76     // Breathing resistance calculation
```

```
77      float br = (brVoltage1 - brVoltage2)*RESISTORBR/brVoltage2;
78
79      // Adding value to the array and increasing variable
80      arrayOfBrValues[sumOfLoops] = br;
81      sumOfLoops++;
82
83      // Average and entry of values
84      if(sumOfLoops == 10) {
85        float sumOfBrValues = 0.0;
86        for(int i = 0; i < sumOfLoops; i++) {
87          sumOfBrValues = sumOfBrValues + arrayOfBrValues[i];
88        }
89        float avgOfBr = sumOfBrValues / sumOfLoops;
90        if(isfinite(avgOfBr)) {
91          doc["b"] = avgOfBr; // Adding average to an array of chars
92        }
93        // Serialise JSON and send
94        serializeJson(doc, output);
95        Serial.println(output);
96        client.publish("/esp/sensors/br", output);
97        sumOfLoops = 0;
98      }
99    }
100 }
101
102 void reconnect() {
103   while(!client.connected()) {
104     Serial.print("MQTT not connected... Trying to connect... ");
105
106     // Create client ID
107     String clientId = "ESP32Client-";
108     clientId += String(random(0xffff), HEX);
109
110     // Attempt connection
111     if(client.connect(clientId.c_str())) {
112       Serial.println("Connected!");
113     } else {
114       Serial.print("Failed, code: ");
115       Serial.println(client.state());
116       delay(5000);
117     }
118   }
119 }
```

Zdrojový kód 5.2: Kód pro výpočet dýchání

Funkce *void setup*, *void reconnect* a *void loop* po řádek 67 převzato z [42].

5.6 Konfigurace Raspberry Pi

Po připojení vývojových desek ESP32 k napájení a nahrání kódu následovalo nastavování počítače Raspberry Pi 4. Je dobré si k počítači pořídit i speciální napájení. Napájení obsahuje USB-C konektor, jeho hodnota napětí je U = 5,1 V, hodnota proudu je I = 3 A. Poněvadž jsem k projektu využil zcela nové Raspberry Pi 4, musel jsem z oficiálích stránek Raspberry stáhnout operační systém. Soubor jsem nahrál na mikro SD kartu, spustil ho, nastavil potřebné věci a takto nachystanou SD kartu jsem vložil do slotu na Raspberry. Dále bylo nutné připojit Raspberry

ke stejné WiFi, k níž má přístup ESP32. Název připojení je uveden v kódu. Abych toho docílil, musel jsem připojit Raspberry k síti přes kabel s konektorem RJ45. Pro vyzkoušení připojení k síti jsem využil příkazu níže.

```
ping <ip-address>
```

Zdrojový kód 5.3: Příkaz pro vyzkoušení spojení

Poté jsem využil dalšího komunikačního protokolu SSH (Secure Shell), abych se připojil po síti k Raspberry a mohl ho ovládat přes terminál ve svém notebooku. Díky tomu nebylo potřeba dalších věcí, jako jsou micro HDMI kabel, počítačová myš nebo klávesnice.

Připojení přes SSH úspěšně proběhlo, následovalo připojení ke zmiňované WiFi síti.

```
sudo nmcli dev wifi connect Pajik15 password <network-password>
```

Zdrojový kód 5.4: Příkaz navázání bezdrátového spojení

Raspberry bylo tímto připojeno bezdrátově k síti a mohlo dojít k odpojení síťového kabelu.

5.6.1 Vytvoření dockeru

Pro účely mé práce jsem na Raspberry potřeboval zprovoznit aplikace pro MQTT komunikaci, příjem dat, uchovávání dat, a grafické zpracování dat. Pro MQTT komunikaci, aby se z Raspberry mohl stát broker, jsem použil open source aplikaci od společnosti Eclipse s názvem Mosquitto. O příjem dat a jejich následné odesílání do databáze InfluxDB se stará Node-RED, vývojový nástroj pro vizuální programování. Samotná databáze InfluxDB byla vyvinuta společností InfluxData na programovacím jazyce Rust. O grafický výstup celého projektu se pak stará aplikace Grafana, což je open source webová aplikace pro analýzu a interaktivní vizualizaci. Pro jednoduchou správu těchto aplikací z různých zařízení jsem zvolil cestu jejich zabalení do kontejnerů, k tomu jsem využil softwaru s názvem IOT Stack, který se řadí mezi tzv. dockery. Docker zařídí jednotné rozhraní a jednoduché spuštění aplikací na mnoha operačních systémech.

Bude následovat seznam příkazů, které bylo nutné vepsat do terminálu, aby se všechny nainstaloval docker, který bude základem pro všechno další.

```
sudo apt update
sudo apt upgrade
curl -fsSL https://raw.githubusercontent.com/SensorsIot/IOTstack/
   master/install.sh | bash
```

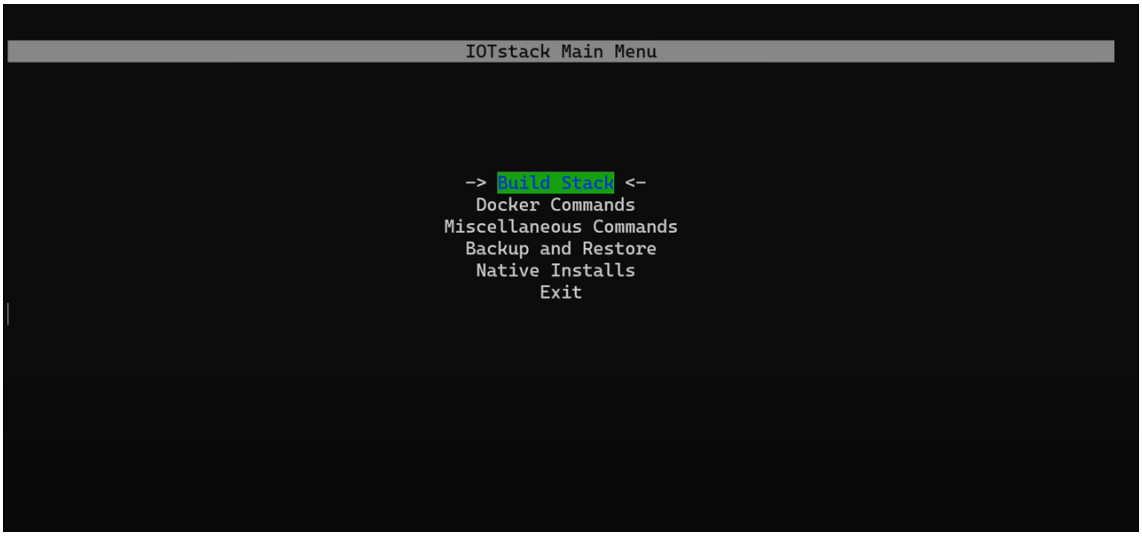
Zdrojový kód 5.5: Příkazy pro instalaci dockeru

Po úspěšné instalaci bylo možno vrhnout se k nastavování všech stažených aplikací. Přesunul jsem se do složky a pustil IOT Stack.

```
cd IOTstack/
./menu.sh
```

Zdrojový kód 5.6: Příkazy pro spuštění dockeru

Po spuštění se objeví obrazovka, která obsahuje menu.

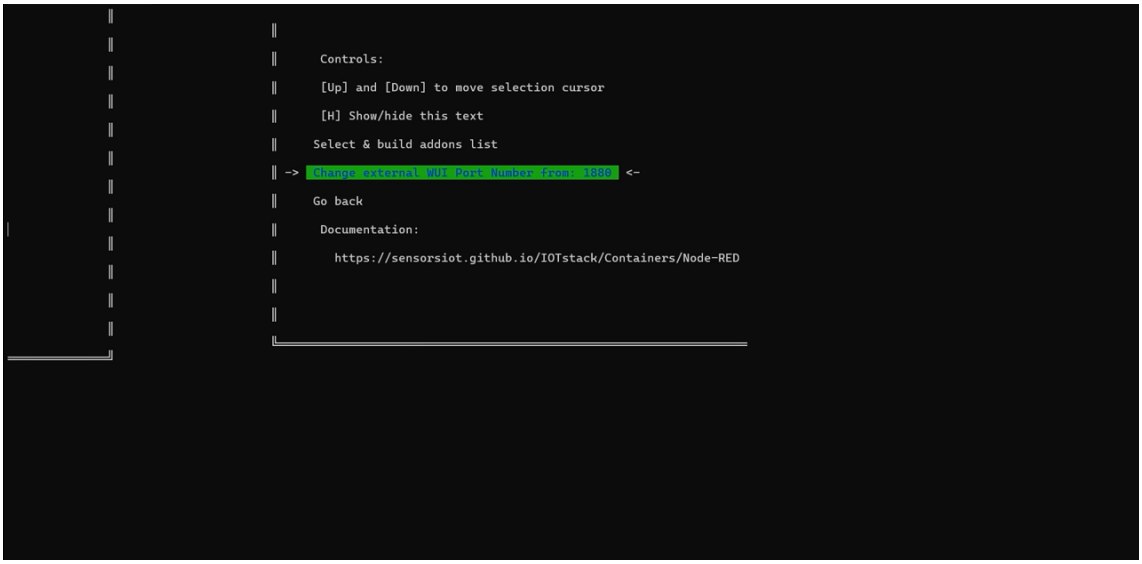


Obrázek 5.12: Úvodní obrazovka programu IOT Stack

Vybral jsem první možnost „Build Stack“. Objevila se obrazovka se seznamem aplikací k volbě vytvoření kontejneru.

Obrázek 5.13: Výběr aplikací

Projel jsem seznam a mezerníkem vybral možnosti grafana, influxdb, mosquitto a nodered. U prvních třech se vše obešlo bez problému a výběr byl potvrzen rozsvícením modré vlaječky napravo s nápisem „Pass“. Při výběru možnosti nodered se však objevily varovné vykřičníky a nápis „Issue“. Po stisknutí pravé šipky se objeví následující stránka.



Obrázek 5.14: Možnosti kontejneru aplikace Node-RED

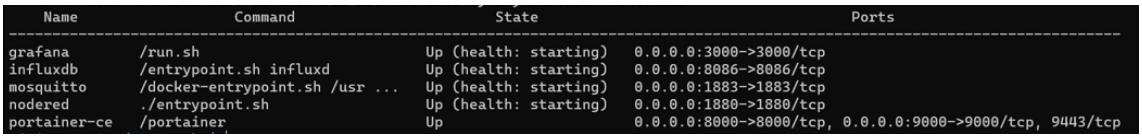
Bylo potřeba vybrat možnost „Select & build addons list“ a poté se vrátit zpět do seznamu aplikací. V seznamu jsem zaškrtl ještě možnost portainer-ce. Ten umí poskytnout přes webové rozhraní kvalitní přehled a správu aplikací v dockeru. Následovalo potvrzení stiskem klávesy Enter. To byl vytvořen speciální soubor, který obsahuje všechny potřebné komponenty.

Zpátky v úvodním menu jsem poté zvolil možnost „Docker Commands“ a na další straně „Start stack“. Po doběhnutí celého procesu jsem IOT Stack opustil. Do terminálu se dá napsat příkaz, který zkontroluje, zda jsou všechny aplikace v běhu.

```
1 docker-compose ps
```

Zdrojový kód 5.7: Kontrola aktivity kontejnerů

Objevila se zpráva na obrázku níže, jež značí, že všechny aplikace běží a je možno pokračovat dále.



Obrázek 5.15: Kontrola stavu aplikací

Toto je možné zkontrolovat i jinou cestou, a sice přes Portainer. Portainer je k nalezení na adrese ve tvaru „http://ipadresazařízení:9000“, kam se musí napsat skutečná IP adresa zařízení, na kterém docker běží, v tomto případě tedy počítač Raspberry Pi 4. Tam je při prvním přihlášení nutné nastavit přihlašovací jméno a heslo.

5.6.2 Sběr a ukládání dat

Po kontrole, že vše běží, jak má, se přešlo k vytvoření databáze, abych si připravil místo pro ukládání dat ze senzorů.

```
docker exec -it influxdb influx
```

Zdrojový kód 5.8: Příkaz pro spuštění databázového programu

Tento první příkaz, který jsem napsal do terminálu mě přenesl do kontejneru, kde jsem mohl začít tvořit databázi, na obrázku níže je vidět prostředí pro tvorbu databáze.

```
pajik@raspberrypi:~/IOTstack $ docker exec -it influxdb influx
Connected to http://localhost:8086 version 1.8.10
InfluxDB shell version: 1.8.10
>
```

Obrázek 5.16: Databáze InfluxDB v terminálu

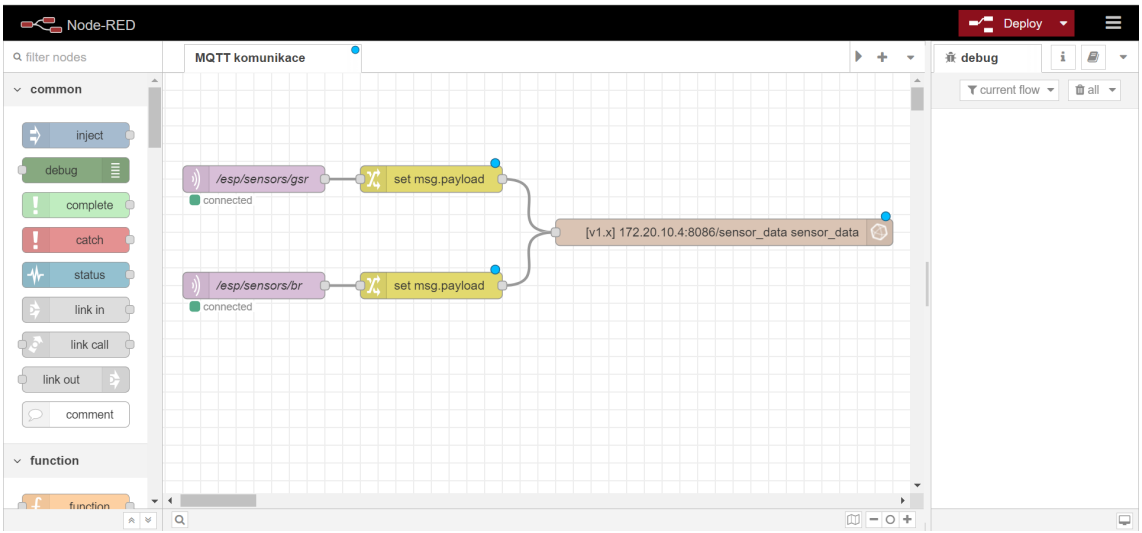
Následoval příkaz, jímž jsem vytvořil novou databázi s názvem *sensor_data*.

```
CREATE DATABASE sensor_data
```

Zdrojový kód 5.9: Příkaz pro vytvoření databáze

Databázi jsem pomocí příkazu *quit* opustil a jal jsem se vytvořit kód do prostředí Node-RED, abych databázi mohl začít plnit daty.

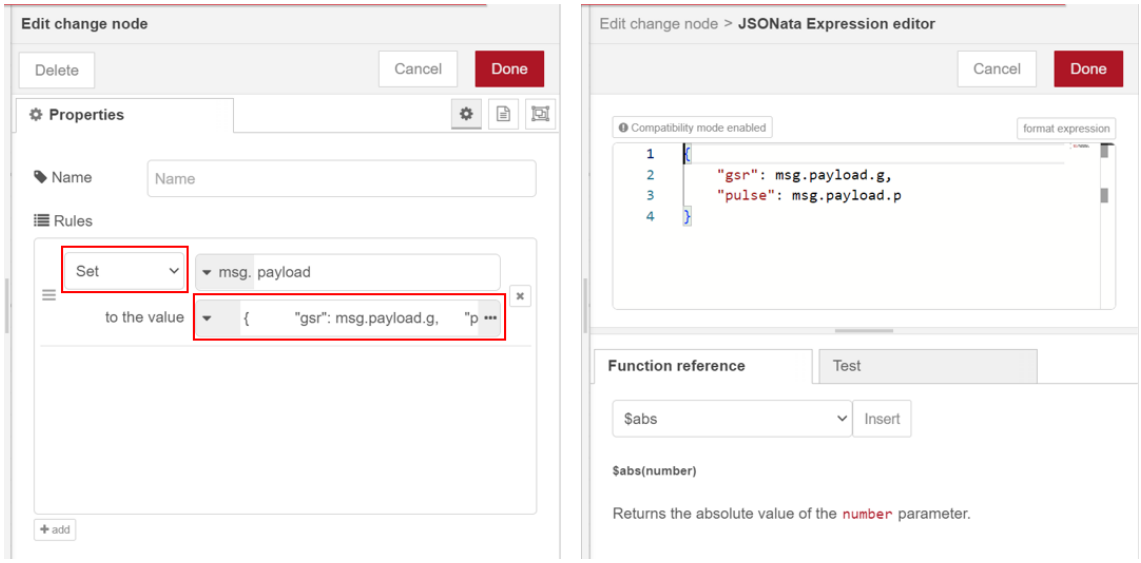
Node-RED je k nalezení na adrese podobné té u aplikace Portainer, ale za dvojtečku se místo čísla 9000 musí napsat číslo 1880. Tato čísla závisí na portu, na němž probíhá komunikace příslušné aplikace se zařízením.



Obrázek 5.17: Prostředí Node-RED

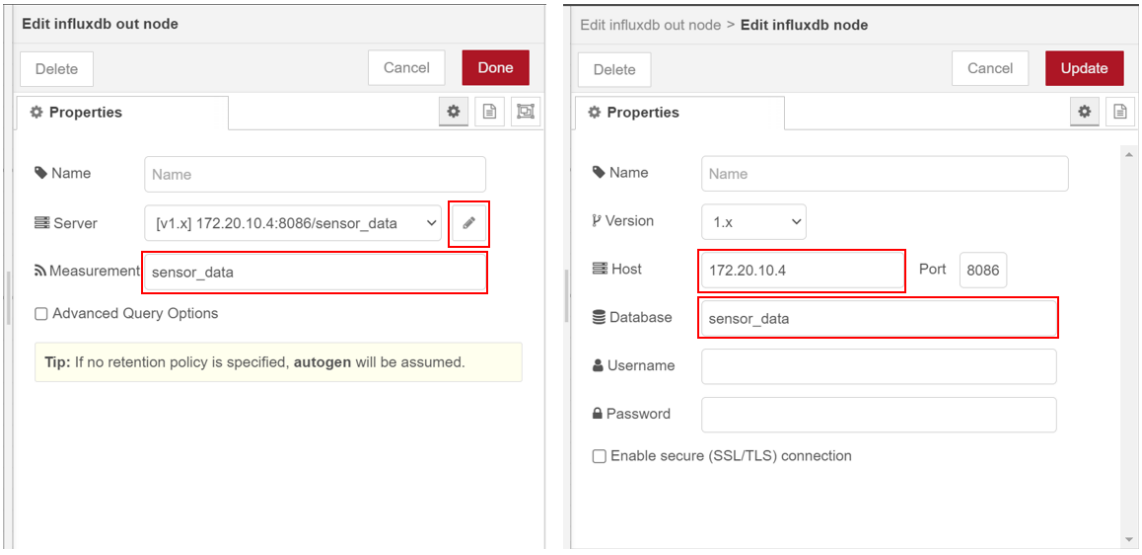
Do Node-REDu nebylo třeba pro účely mé práce doinstalovávat další knihovny. Použil jsem uzly *MQTT in* ze sekce *network*, dále *change* ze sekce *function* a nakonec *influxdb out* ze *storage*. Aby došlo ke správnému nastavení a celá komunikace mohla probíhat, bylo nutné všechny uzly upravit. Dvojklikem jsem je tedy postupně rozklikl. Do prvního uzlu jsem vyplnil sekci *Topic* a *Name*. *Topic* musí být vyplněn stejnou sekvencí, jež se nachází v kódu nahraném do desky ESP32 na řádku 108 v uvozovkách, tedy /esp/sensors/gsr. Jméno může být libovolné, já zvolil pro přehlednost to stejné. Protokol také samozřejmě musí vědět, jaké zařízení má být zvoleno

jako broker. Po rozkliknutí ikony tužky na prvním řádku napravo se objevila, záložka *Connection*, kde jsem do sekce *Server* vyplnil IP adresu použitého počítače Raspberry. Uzel *MQTT in* bylo třeba použít dvakrát, ale ve druhém jsem nahradil /esp/sensors/gsr za /esp/sensors/br jak v sekci *Topic*, tak v sekci *Name*.



Obrázek 5.18: Nastavení uzlu *change*

Nastavení dalšího uzlu je znázorněno na obrázku výše. Dvojklikem jsem otevřel nastavení uzlu change a změnil funkci uzlu na *Set* a do kolonky níže jsem vybral možnost *Expression*. Tu jsem potom ještě upravil rozkliknutím tří teček a vepsáním textu, jak je vidět na obrázku vpravo. To mi zařizuje správné parsování přijímaného JSON souboru. Ve druhém uzlu *change* je tato část nahrazena pouze výrazem „"breathing": msg.payload.b“.



Obrázek 5.19: Nastavení uzlu *influxdb out*

Zbýval poslední uzel, a sice *influxdb out*. Opět je potřebné nastavení vidět na obrázku níže. Bylo nutné vyplnit IP adresu MQTT brokeru, vyplnit jméno databáze, kterou jsem předtím přes terminál vytvořil a zvolit jméno tabulky, do níž se data

ukládají. Já zvolil stejný název jako pro celou databázi, to jest „sensor_data“.

S Node-REDem jsem byl tímto hotov, poslední přišla na řadu aplikace Grafana. Ještě předtím jsem ale chtěl zkontrolovat, zda se databáze správně plní daty. Uvedl jsem do chodu desky ESP32 a pomocí USB kabelu jsem do nich nahrál aktuální podobu kódu. Že je vše v pořádku a komunikace je navázána, nám potvrdí zpráva ze sériového monitoru na obrázku níže, ale pouze v případě, že jsou desky ESP32 připojeny přes USB.

```
.
WiFi Connected, IP: 172.20.10.6
MQTT not connected... Trying to connect... Connected!
{"g":107848.75,"p":2776.800049}
```

Obrázek 5.20: Potvrzovací zpráva ze sériového monitoru

V terminálu jsem opět použil příkaz níže.

```
1 docker exec -it influxdb influx
```

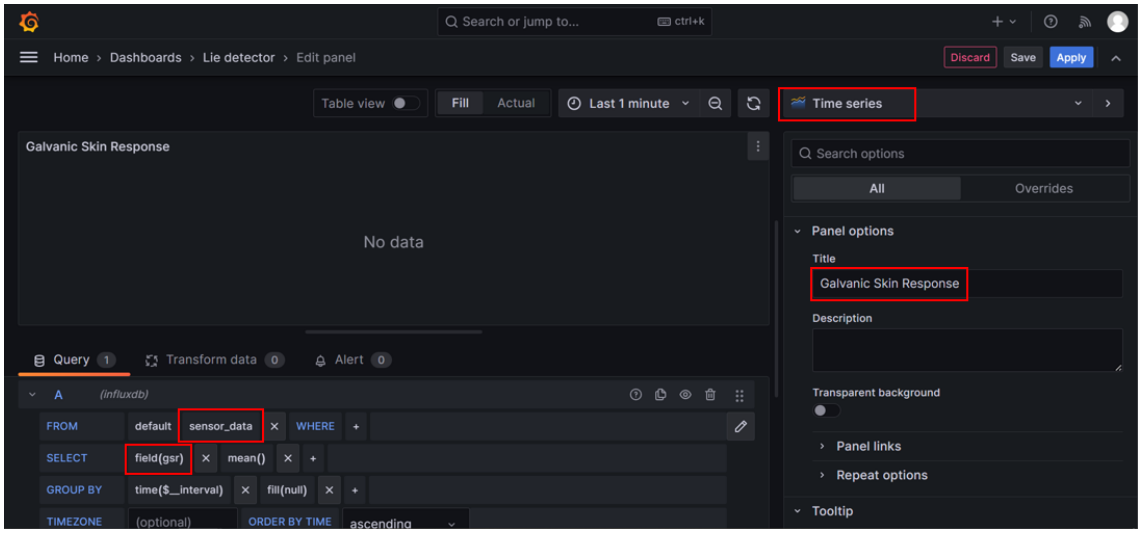
Dále následují příkazy pro kontrolu sběru dat v databázi. Senzory jsem si na sebe nasadil, protože u senzoru elektrodermální aktivity by nebyly hodnoty, které by se mohly zobrazit, protože ruka vytvoří sepnutí celého obvodu.

```
1 USE sensor_data
2 show measurements
3 select * from sensor_data
```

Zdrojový kód 5.10: Příkazy pro kontrolu sběru dat

5.6.3 Vizualizace dat

A teď tedy ke Grafaně. Opět její adresa vypadá podobně jako předchozí, za dvojtečkou následuje číslo 3000. Počáteční uživatelské jméno i heslo bylo na začátku „admin“, hned na začátku je vyžadováno změnění hesla. Po změně jsem rozklikl možnost *Data Sources - Add your first data source*, poté *InfluxDB*. V sekci *HTTP* jsem vyplnil URL adresu opět v podobě jako v předchozích případech, za dvojtečkou číslo 8086, v sekci *InfluxDB Details* název databáze „sensor_data“. U *HTTP Method* jsem zvolil možnost *GET* a minimální časový interval nastavil na 1 vteřinu. V záložce nalevo jsem vybral ikonu *Dashboards* se čtyřmi čtverečky a zvolil možnost *New dashboard*, následně *Add a new panel*.



Obrázek 5.21: Tvorba vizualizace

Zde na obrázku je vidět, jak jsem postupoval přímo při tvorbě grafů. Je důležité zvolit v sekci *FROM* správnou databázi, pokud jich je více. Já v tomto případě zvolil jedinou možnost, a sice „sensor_data“. V sekci pod tím se dalo zvolit, jaký psychofyziologický jev chci v daném grafu zobrazit. V pravém panelu jsem změnil názvy a nakonec jsem rozvinul možnost *Time series* a změnil vizáž grafu.

Jelikož všechny aplikace běží v dockeru, bylo trochu složitější výsledné grafy ještě doupravit. Abych zmenšil refreshovací frekvenci vykreslování hodnot a také byl schopen zobrazit grafy na libovolně vytvořené webové stránce, musel jsem přes terminál upravit soubor *docker-compose.yml*.

```
cd IOTstack/
nano docker-compose.yml
```

Zdrojový kód 5.11: Příkazy ke vstupu do editace souboru dockeru

```yaml
services:
  grafana:
    container_name: grafana
    image: grafana/grafana
    restart: unless-stopped
    user: "0"
    ports:
    - "3000:3000"
    environment:
    - TZ=Etc/UTC
    - GF_PATHS_DATA=/var/lib/grafana
    - GF_PATHS_LOGS=/var/log/grafana
    - GF_SECURITY_ALLOW_EMBEDDING=true
    - GF_AUTH_ANONYMOUS_ENABLED=true
    - GF_DASHBOARDS_MIN_REFRESH_INTERVAL=1s
    volumes:
    - ./volumes/grafana/data:/var/lib/grafana
    - ./volumes/grafana/log:/var/log/grafana
    healthcheck:
      test: ["CMD", "wget", "-O", "/dev/null", "http://localhost:3000"]
      interval: 30s
      timeout: 10s
      retries: 3
```

Obrázek 5.22: Úprava souboru *docker-compose.yml*

Pro výše zmíněné účely jsem do zmíněného souboru přidal poslední 3 odrážky pod úsek *environment*. Takto připravený projekt už potřeboval pouze figuranta.

5.7 Návod k použití

1. Připojte mikropočítač Raspberry Pi 4 model B do napájení.
2. V internetovém prohlížeči (na jakémkoliv zařízení, klidně i přes Raspberry) vyhledejte adresu *http://ipadresaraspnerrypi:3000*.
3. Nasaďte na pásek krabičku se senzory měření odporu kůže a srdeční frekvence a pásek obepněte kolem libovolné ruky zkoumaného člověka.
4. Gumový pás senzoru dýchání omotejte kolem hrudníku zkoumaného člověka a poté zahákněte za pás krabičku s elektronikou.
5. Nasaďte suché zipy senzoru měření odporu kůže na dva prsty na ruce zkoumaného a třetí prst nechť lehce položí na senzor srdeční frekvence.
6. Na obou krabičkách uveďte vypínač do polohy I a sledujte na obrazovce průběhy. Grafy jsou popsané.
7. Pro odstranění senzorů aplikujte reverzní postup.

5.8 Postup při nabíjení

1. Nabíječku s USB mini konektorem strčte do otvoru pod vypínačem.
2. Vypínač musí být pro nabíjení uveden do polohy I.
3. Kontrolujte stav baterie pomocí tlačítka na LED indikátoru.
4. Až budou na indikátoru svítit všechny 4 LED diody, odpojte nabíječku a vypínač uveďte zpět do polohy 0.

5.9 Postup při nahrání kódu

Tento postup je zde pouze v případě, že byste zařízení chtěli použít k jiným účelům, než je určeno. Do desky ESP32 se dá nahrát kód dle libosti. V základě bude zařízení funkční v podobě, jaká byla v tomto projektu popsána.

1. Strčte kabel s USB mini konektorem do druhého otvoru než při nabíjení. Deska ESP32 by se měla rozsvítit červeně, protože detekuje napájení.
2. Nahrajte z počítače libovolný kód.
3. Kabel můžete odpojit a zapnou vypínačem napájení z baterie, anebo můžete nabíjet i samotným kabelem.
4. Pro vrácení projektu do počáteční podoby se zde nacházejí kódy původního projektu. Nahrajte je přes kabel zpět do desek ESP32.

Závěr

V rámci této práce byl navržen a sestrojen přístroj pro měření elektrodermální aktivity, dýchání a srdeční frekvence. Na začátku proběhla rešerše základních psychofyziologických jevů a také rešerše jejich snímání a vyhodnocování duševního stavu v technických systémech. Následoval rozbor všech možných použitelných hardwarových prvků a používaných komunikačních protokolů v rámci IoT.

V praktické části jsem se nejprve obeznámil s možností sestrojení celého měřicího zařízení. Podle fungujícího konceptu jsem vytvořil senzory měření elektrodermální aktivity a dýchání, senzor tepové frekvence byl koupen. Desky ESP32 jsem si připravil, aby se na ně daly pájením připevnit senzory. Dále jsem desky připojil k modulům TP4056, na něž jsem vyvedl i póly baterie. Mezi baterii a modul jsem vložil vypínač. Měl jsem možnost umístit tlačítko buď takto nebo ho dát mezi modul a vývojovou desku. Rozhodl jsem se pro první řešení, protože modul by baterii pomalu vybíjel a pokud by dlouho neměla být připojena k nabíječce, mohlo by jí to poškodit. V aplikaci Autodesk Inventor jsem vytvořil kryt, který jsem nechal na 3D tiskárně vytisknout. Přemýšlel jsem o dvou konceptech krytu, jeden by byl trošku otevřenější a bylo by snáze vidět, co se skrývá uvnitř, ale po úvaze jsem zvolil krytí celého mechanismu. Je to pohodlnější pro manipulaci, redukuje možnost vytrhnutí kabelu nebo jiného poškození zařízení. Připravil jsem si kód pro zpracování hodnot ze senzorů a pro zprostředkování bezdrátové komunikace. Po zvážení kladů a záporů jednotlivých protokolů v rámci IoT jsem usoudil, že nejlepší bude použít protokol MQTT. Bylo třeba nastavit bezdrátovou komunikaci i na Raspberry Pi, kde jsem dále přes aplikaci Node-RED vytvořil kód pro získávání dat z brokeru, databázi přes InfluxDB pro uchovávání dat a grafický výstup přes aplikaci Grafana.

Senzor tepové frekvence bohužel ukazuje po uvedení do chodu zvláštní hodnoty. Došel jsem k názoru, že chyba je způsobena buď velkou poruchovostí daného senzoru nebo nedostatečným napájecím napětím. Podle dokumentace by sice napětí 3,3 V z pinu 3V3 desky ESP32 mělo stačit, ale na internetu jsem se často dočetl o stejném problému, který uživatelé často řešili buď změnou napájení nebo koupí jiného senzoru.

Projekt ukázal, že komponenty používané v rámci internetu věcí jsou vhodné pro širokou škálu aplikací, zejména díky své cenové dostupnosti a nepříliš obtížné manipulaci. Je s nimi možný sběr a analýza dat v reálnem čase, což je vlastnost, která se při zjišťování duševního stavu operátora velmi hodí.

Byly identifikovány také možnosti dalšího vylepšení zařízení, včetně zvýšení přesnosti měření, optimalizace uživatelského rozhraní a další miniaturizace, což by umožnilo integraci zařízení do běžného oblečení a poskytlo uživatelům pohodlí a volnost

pohybu.

Bibliografie

1. PROCHÁZKA, Roman; SEDLACKOVA, Zuzana. *Vybrané kapitoly z psychofyziologie*. 2015. ISBN 978-80-244-4490-1.
2. CHOCHOLOUŠKOVÁ, Veronika. *VZTAH POZICE TĚLA A AUTONOMNÍ NERVOVÉ SOUSTAVY*. 2019. Dis. pr. ZÁPADOČESKÁ UNIVERZITA V PLZNI.
3. SPIELBERGER, Charles D. *Encyclopedia of Applied Psychology*. Elsevier Inc, 2004. ISBN 978-0-12-657410-4.
4. BENEDEK, Mathias; KAERNBACH, Christian. A continuous measure of phasic electrodermal activity. *Journal of Neuroscience Methods*. 2010, roč. 190, s. 80–91. ISSN 0165-0270. Dostupné z DOI: 10.1016/j.jneumeth.2010.04.028.
5. BOUCSEIN, Wolfram. *Electrodermal Activity*. Springer, 2012. ISBN 978-1-4614-1125-3.
6. GERŠAK, Gregor; DRNOVŠEK, Janko. Electrodermal activity patient simulator. *PLOS ONE*. 2020, roč. 15, e0228949. ISSN 1932-6203. Dostupné z DOI: 10.1371/journal.pone.0228949.
7. GRECO, Alberto; LANATA, Antonio; CITI, Luca; VANELLO, Nicola; VALENZA, Gaetano; SCILINGO, Enzo. Skin Admittance Measurement for Emotion Recognition: A Study over Frequency Sweep. *Electronics*. 2016, roč. 5, s. 46. ISSN 2079-9292. Dostupné z DOI: 10.3390/electronics5030046.
8. *Electroencephalogram Lab*. 2007. California State University.
9. YEO, Sang Seok; CHANG, Pyung Hun; JANG, Sung Ho. The Ascending Reticular Activating System from Pontine Reticular Formation to the Thalamus in the Human Brain. *Frontiers in Human Neuroscience*. 2013, roč. 7, Not available. ISSN 1662-5161. Dostupné z DOI: 10.3389/fnhum.2013.00416.
10. LOIZAGA, Erlantz; EYAM, Aitor Toichoa; BASTIDA, Leire; LASTRA, José l. Martínez. A Comprehensive Study of Human Factors, Sensory Principles, and Commercial Solutions for Future Human-Centered Working Operations in Industry 5.0. *IEEE Access*. 2023, roč. 11, s. 53806–53829. Dostupné z DOI: 10.1109/ACCESS.2023.3280071.
11. KRIJTOVÁ, Hana. *Standardní EEG – základy, indikace a základní nálezy* [online]. 2. lékařská fakulta Univerzity Karlovy, 2016. [cit. 2024-03-24]. Dostupné z: https://www.lf2.cuni.cz/files/page/files/2016/zaklady_eeg.pdf.
12. NIENKE VAN ATTEVELDT Tieme W. P. Janssen, Ido Davidesco. Measuring Brain Waves in the Classroom. *Frontiers for Young Minds*. 2020, roč. 8, s. 96. Dostupné z DOI: 10.3389/frym.2020.00096.

13. WIKISKRIPTA. *Elektroencefalografie* [online]. 2021. [cit. 2024-05-26]. Dostupné z: https://www.wikiskripta.eu/w/Elektroencefalografie.

14. İLHAN, İlhan. Smart blood pressure holter. *Computer Methods and Programs in Biomedicine*. 2018, roč. 156, s. 1–12. ISSN 0169-2607. Dostupné z DOI: 10.1016/j.cmpb.2017.12.025.

15. SILVA, Pedro Manuel Pinto da. *A pervasive system for real-time blood pressure Monitoring*. 2013. Dis. pr. FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO.

16. GUPTA, Ben. *Invasive blood pressure monitoring* [online]. 2007. [cit. 2024-04-29]. Dostupné z: https://resources.wfsahq.org/wp-content/uploads/uia23-Invasive-blood-pressure-monitoring.pdf.

17. MATTHEW WARD, Jeremy A. Langton. Blood pressure measurement. *Continuing Education in Anaesthesia, Critical Care & Pain*. 2007, roč. 7, s. 122–126. ISSN 4. Dostupné z DOI: 10.1093/bjaceaccp/mkm022.

18. FARNSWORTH, Bryn. *What Is ECG and How Does It Work?* [online]. 2021. [cit. 2024-05-25]. Dostupné z: https://imotions.com/blog/learning/research-fundamentals/what-is-ecg/.

19. CLINIC, Cleveland. *Heart Rate Monitor* [online]. 2022. [cit. 2024-05-06]. Dostupné z: https://my.clevelandclinic.org/health/diagnostics/23429-heart-rate-monitor.

20. VLADIMÍR ECK, Miroslav Razím. *Biokybernetika*. Vydavatelství ČVUT, 1996.

21. ABB, Klub. *Hodnoty MTTF, MTBF a FIT u jističů ABB* [online]. 2022. [cit. 2024-05-25]. Dostupné z: https://nizke-napeti.cz.abb.com/otazky-a-odpovedi-hodnoty-mttf-mtbf-a-fit-u-jisticu-abb-19915.

22. ECK, Vladimír. *Bionika*. Vydavatelství ČVUT, 1998.

23. JURA, J. Psychological Instrumentation: From the Wundt's Laboratory to Artificial Intelligence. In: *Nové metody a postupy v oblasti přístrojové techniky, automatického řízení a informatiky*. Praha: ČVUT FS, Ústav přístrojové a řídící techniky, 2012, s. 90–94. NTjura4.pdf.

24. JAMIE FREED, Allison Lampert. *European regulator rules out single-pilot flying by 2030* [online]. 2023. [cit. 2024-05-26]. Dostupné z: https://www.reuters.com/business/aerospace-defense/european-regulator-rules-out-single-pilot-flying-by-2030-2023-02-06/.

25. LIU, Jing; GARDI, Alessandro; RAMASAMY, Subramanian; LIM, Yixiang; SABATINI, Roberto. Cognitive pilot-aircraft interface for single-pilot operations. *Knowledge-Based Systems*. 2016, roč. 112, s. 37–53. ISSN 0950-7051. Dostupné z DOI: 10.1016/j.knosys.2016.08.031.

26. AYAZ, Hasan; SHEWOKIS, Patricia A.; BUNCE, Scott; IZZETOGLU, Kurtulus; WILLEMS, Ben; ONARAL, Banu. Optical brain monitoring for operator training and mental workload assessment. *NeuroImage*. 2012, roč. 59, s. 36–47. ISSN 1053-8119. Dostupné z DOI: 10.1016/j.neuroimage.2011.06.023.

27. KABEŠ, Ing. Karel. Senzory ve volantu sledují životní funkce řidiče. *Automatizace v dopravě*. 2012, roč. 6, s. 50. Dostupné také z: https://www.automa.cz/cz/casopis-clanky/senzory-ve-volantu-sleduji-zivotni-funkce-ridice-2012_06_0_9670/.

28. HAMILTON, Zach. *What Is A Dead Man's Switch?* [online]. 2021. [cit. 2024-05-06]. Dostupné z: https://medium.com/sarcophagus/what-is-a-dead-mans-switch-86a1f4853eed.

29. TOMEŠEK, Jiří. *Lekce 1 - Seznámení s ESP-32 - Online kurz* [online]. 2023. [cit. 2024-02-01]. Dostupné z: https://www.itnetwork.cz/hardware-pc/arduino/esp32/seznameni-s-esp-32.

30. VODA, Zbyšek. *SEZNÁMENÍ S ARDUINEM* [online]. 2014. [cit. 2024-02-01]. Dostupné z: https://bastlirna.hwkitchen.cz/seznameni-s-arduinem/.

31. 101, Components. *Arduino Uno* [online]. 2021. [cit. 2024-02-02]. Dostupné z: https://components101.com/microcontrollers/arduino-uno.

32. BASUMALLICK, Chiradeep. *What Is Raspberry Pi? Models, Features, and Uses* [online]. 2022. [cit. 2024-05-07]. Dostupné z: https://www.spiceworks.com/tech/networking/articles/what-is-raspberry-pi/.

33. HEATH, Nick. *What is the Raspberry Pi 4? Everything you need to know about the tiny, low-cost computer* [online]. 2019. [cit. 2024-05-08]. Dostupné z: https://www.zdnet.com/article/what-is-the-raspberry-pi-4-everything-you-need-to-know-about-the-tiny-low-cost-computer/.

34. SOFTWARE, Open Automation. *What is an IoT Gateway?* [online]. [cit. 2024-02-29]. Dostupné z: https://openautomationsoftware.com/open-automation-systems-blog/what-is-an-iot-gateway/.

35. TEAM, HiveMQ. *What is MQTT Quality of Service (QoS) 0,1, & 2? – MQTT Essentials: Part 6* [online]. 2024. [cit. 2024-03-03]. Dostupné z: https://www.hivemq.com/blog/mqtt-essentials-part-6-mqtt-quality-of-service-levels/.

36. RADWARE. *What is CoAP? Understanding the Constrained Application Protocol* [online]. 2024. [cit. 2024-03-06]. Dostupné z: https://www.radware.com/security/ddos-knowledge-center/ddospedia/coap/.

37. CRESSLER, Cosette. *Everything About XMPP - Extensible Messaging & Presence Protocol* [online]. 2021. [cit. 2024-03-11]. Dostupné z: https://www.cometchat.com/blog/xmpp-extensible-messaging-presence-protocol.

38. BABUN, Leonardo; DENNEY, Kyle; CELIK, Z. Berkay; MCDANIEL, Patrick; ULUAGAC, A. Selcuk. A survey on IoT platforms: Communication, security, and privacy perspectives. *Computer Networks*. 2021, roč. 192, s. 108040. ISSN 1389-1286. Dostupné z DOI: 10.1016/j.comnet.2021.108040.

39. HRON, Pavel. *Výroba a zapojení senzoru dýchání*. 2024. Dis. pr. České vysoké učení technické v Praze.

40. FORMÁNEK, Josef. *Můstkové metody*. Plzeň, Česká republika: Západočeská univerzita v Plzni.

41. MERRET, Orbit. *Můstky - II* [online]. 2021. [cit. 2024-05-26]. Dostupné z: https://www.orbitmerret.eu/cs/mustky-ii.

42. SYSTEMS, Learn Embedded. *ESP32 Sensor Node Source Code* [Available online]. 2022. Dostupné také z: https://learnembeddedsystems.co.uk/esp32-sensor-node-source-code.