VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

VZDÁLENÉ OVLÁDÁNÍ EMBEDDED SYSTÉMU MOBILNÍ APLIKACÍ S VYUŽITÍM PŘENOSU DAT PŘES BLUETOOTH

REMOTE CONTROL OF AN EMBEDDED SYSTEM VIA A MOBILE APPLICATION USING DATA TRANSFER OVER BLUETOOTH

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — Richard Dvořák

AUTHOR

VEDOUCÍ PRÁCE — Ing. Jakub Dokoupil, Ph.D.

SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Automatizační a měřicí technika**

Ústav automatizace a měřicí techniky

Student: Richard Dvořák ***ID:*** 240335

Ročník: 3 ***Akademický rok:*** 2023/24

NÁZEV TÉMATU:

Vzdálené ovládání embedded systému mobilní aplikací s využitím přenosu dat přes Bluetooth

POKYNY PRO VYPRACOVÁNÍ:

1. Seznamte se s principy komunikace mobilních zařízení postavených na bezdrátové technologii Bluetooth.
2. Na zadaném hardwaru realizujte aplikaci pro obousměrný přenos dat.
3. Naprogramujte uživatelskou aplikaci kompatibilní s běžnými operačními systémy mobilních zařízení.
4. Systém komunikace otestujte a demonstrujte jeho funkčnost.

DOPORUČENÁ LITERATURA:

L. Potter, "Hands-On Mobile and Embedded Development with Qt 5". Birmingham, U.K.: Packt Publishing Ltd, 2019.

Termín zadání: 5.2.2024 ***Termín odevzdání:*** 22.5.2024

Vedoucí práce: Ing. Jakub Dokoupil, Ph.D.

Ing. Miroslav Jirgl, Ph.D.
předseda rady studijního programu

ABSTRAKT

Tato práce je zaměřena na naprogramovaní mobilní aplikace, která bude kompatibilní s běžnými mobilními operačními systémy, a demonstraci obousměrné bezdrátové komunikace mezi aplikací a zadaným modulem ESP32. Pro komunikaci mezi zařízeními byla zvolena bezdrátová technologie Bluetooth a mobilní aplikace byla navrhnuta a naprogramována v Qt Frameworku. Pro demonstraci funkčnosti bezdrátové komunikace je modul ESP32 nakonfigurován do role SPP serveru, ke kterému se mobilní zařízení připojuje v roli klienta.

KLÍČOVÁ SLOVA

Bluetooth, Bluetooth low energy, bezdrátová komunikace, mobilní aplikace, ESP32, Qt Framework

ABSTRACT

This work is focused on programming a mobile application that will be compatible with common mobile operating systems, and demonstrating two-way wireless communication between the application and the specified ESP32 module. Bluetooth wireless technology was chosen for communication between devices, and the mobile application was designed and programmed in the Qt Framework. To demonstrate the functionality of wireless communication, the ESP32 module is configured in the role of an SPP server, to which the mobile device connects in the role of a client.

KEYWORDS

Bluetooth, Bluetooth low energy, wireless communication, mobile application, ESP32, Qt Framework

DVOŘÁK, Richard. *Vzdálené ovládání embedded systému pomocí mobilního zařízení*. Bakalářská práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2024. Vedoucí práce: Ing. Jakub Dokoupil, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Richard Dvořák
VUT ID autora:	240335
Typ práce:	Bakalářská práce
Akademický rok:	2023/24
Téma závěrečné práce:	Vzdálené ovládání embedded systému pomocí mobilního zařízení

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Tímto chci poděkovat mému vedoucímu bakalářské práce, panu Ing. Jakubu Dokoupilovi, Ph.D. za odborné vedení. Dále patří mé poděkování panu doc. Ing. Bohumilu Klímovi, Ph.D. za trpělivost, vstřícný přístup a hodnotné poznámky k technické stránce mé bakalářské práce.

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Před vznikem technologie Bluetooth se lidé museli spolehnout na přenos dat pomocí kabelových spojů. Možnosti komunikace tak byly velmi omezené, s nástupem bezdrátových technologií se však otevřely nové možnosti.

Bluetooth je bezdrátový komunikační protokol, který původně vznikl za účelem nahradit sériovou linku RS-232. Od roku 1999, kdy se na trhu poprvé objevilo zařízení s touto technologií, se stal součástí lidských životů a dnes je Bluetooth podporován širokým spektrem zařízení. Umožňuje snadno propojovat sluchátka, reproduktory, mobilní a embedded zařízení, počítače a další.

Díky dostupným softwarovým nástrojům, které podporují bezdrátové a mobilní technologie, je dnes poměrně jednoduché vyvinout mobilní aplikaci, která za pomocí Bluetooth přenáší data do embedded zařízení. Jedním z takovýchto softwarů je QtFramework, který poskytuje sadu specializovaných nástrojů pro tvorbu a design počítačových a mobilních aplikací.

1 Bluethooth

V této kapitole je popis technologie Bluetooth, její architektury, technických specifikací standardu a procesu párování a spojení dvou či více zařízení. V sekci historie je letmo popsaný vznik tohoto standardu.

Bluetooth je bezdrátovou technologií, kterou v dnešní době nalezneme téměř v každém mobilním zařízení. Slouží pro přenos dat, hudby, videa a dalších. Původně vznikla jako náhrada sériové linky RS-232 s vidinou vytvoření univerzálního komunikačního rozhraní.

1.1 Historie Bluetooth

Název technologie Bluetooth vznikl odvozením ze jména dánského krále Haralda Modrozuba, nebo-li Bluetooth. Tento král, vládnoucí v 10. století, je známý pro své významné diplomatické schopnosti, díky kterým byl schopen ukončit probíhající konflikty mezi kmeny za pomocí vzájemné diskuse. Při vzniku technologie Bluetooth bylo využito této analogie při jejím pojmenování. [1, 2]

Vývoj bezdrátové rádiové technologie byl iniciován v roce 1989 firmou Ericsson mobile ve Švédsku, jejímž prvotním zájmem bylo vyvinout bezdrátová sluchátka. S hlavním návrhem a prvním vývojem přišla firma v roce 1994 a o tři roky později, v roce 1997, představila funkční řešení. První zařízení s technologií Bluetooth bylo uvedeno na trh v roce 1999, jednalo se o verzi Bluetooth 1.0, která obsahovalo značné množství technických problémů, zejména nefunkční párování a téměř nulové zabezpečení připojení. [2, 3]

Do dnešního dne je technologie Bluetooth vyvíjena Bluetooth Special Interest Group (BSIG), která má více než 38 000 členských společností [4].

1.2 Technická specifikace Bluetooth

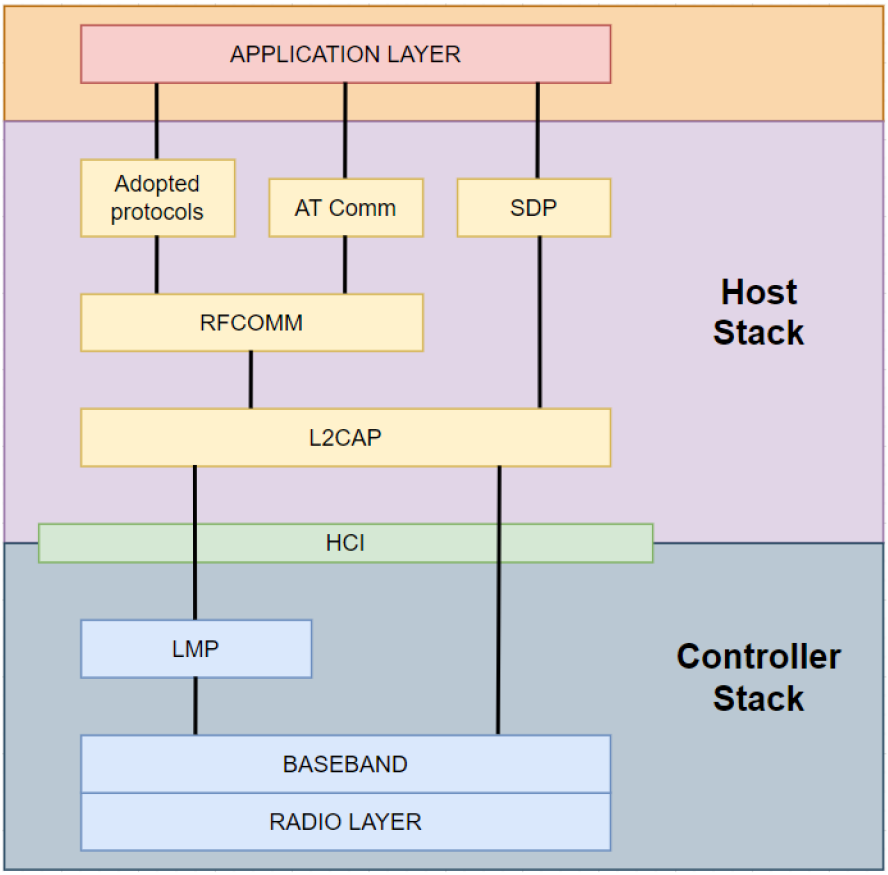
Technologie Bluetooth je normalizovaným standardem IEEE 802.15.1. Spadá do kategorie osobních počítačových sítí, tzv. PAN [1].

Bluetooth využívá pro svou komunikaci rádiové vlny. Pracuje v bezlicenčním ISM pásmu 2,4 GHz, což jsou části rádiového spektra, které byly mezinárodně určeny pro telekomunikaci, zdravotnictví či průmysl [6]. Při přenosu dat využívá metodu Frequency Hopping Spread Spectrum (FHSS), která přeskakuje mezi frekvencemi při přenosu jednotlivých informací [5]. Pracuje se 79 frekvenčními kanály, které mají vzájemný rozestup 1 MHz. Mezi těmito kanály je provedeno 1600 přeskoků za sekundu. Tato metoda má za úkol zejména zabránit rušení na stejné frekvenci. [1]

Tab. 1.1: Výkonové třídy Bluetooth [1]

Třída	Maximální povolený výkon		Dosah (přibližný)
	mW	dBm	
1	100	20	100 m
2	2.5	4	10 m
3	1	0	1 m

Bezdrátový standard Bluetooth byl primárně vyvíjen tak, aby dokázal přenášet co největší objem dat při co nejmenší spotřebě energie. Dělí se do tří základních výkonových tříd, které jsou v tabulce 1.1. Dosahy uvedeny v tabulce jsou pouze přibližné, jelikož počítají s přenosem ve volném prostranství. Překážky jako jsou lidé, zdi nebo budovy, značně ovlivní dosah signálu. Dosah signálu taktéž závisí na hardwarové konfiguraci zařízení či stavu baterie. [1]



Obr. 1.1: Architektura Bluetooth protocol stacku [9]

1.3 Architektura Bluetooth

Technologie Bluetooth se skládá z různých vrstev, od fyzické až po aplikační vrstvu. Tyto jednotlivé vrstvy tvoří Bluetooth protocol stack (viz. obrázek 1.1). Bluetooth protocol stack se dělí na části controller stack a host stack. Každý z těchto stacků pracuje se svou sadou protokolů, jejich funkce a konkrétní protokoly jsou popsány v této kapitole níže. Základní protokoly pro všechny Bluetooth protocol stacky jsou definovány BSIG a dodatečné protokoly byly převzaty od jiných normalizačních orgánů. Povinnými protokoly pro Bluetooth jsou SDP, LMP a L2CAP. Kromě toho protokoly HCI a RFCOMM jsou všeobecně podporovány. [7, 8]

1.3.1 Controller stack

Hardwarová část skládající se z mikroprocesoru a Bluetooth rádia.

Jednotlivé vrstvy controller stacku jsou:

- **Radio layer** - Stanovuje specifikace a fyzickou strukturu pro přenos rádiových vln. Definuje frekvenční pásma, specifikace přeskakování frekvencí a modulační techniky. [9]
- **Baseband** - Přímo využívá služeb rádiového protokolu a s jeho pomocí definuje schéma adresování, časování, algoritmy řízení výkonu a formát paketového rámce [9].
- **Link Manager Protocol (LMP)** - Tento protokol vytváří logická spojení mezi jednotlivými Bluetooth zařízeními, udržuje spojení pro umožnění komunikace mezi nimi, ověřuje zařízení, šifruje zprávy a vyjednává velikost paketů [9].
- **Host Controller Interface (HCI)** - Standard společný jak pro controller stack, tak pro host stack. Jedná se o standardizovanou komunikaci právě mezi controller stackem a host stackem. Hlavní výhodou HCI je, že umožňuje výměnu jednoho ze stacků bez problémů s komunikací mezi nimi. Je specifikován pro komunikační rozhraní RS-232, USB a UART. [11]

1.3.2 Host stack

Softwarová část implementována jako součást operačního systému daného zařízení zabývající se high level data (daty na vysoké úrovni) [8].

U integrovaných zařízení jako jsou například Bluetooth headseaty, či handsfree může běžně controller i host stack řídit jeden mikroprocesor. Hlavním důvodem je snížení výrobních nákladů. Takovýto systém se nazývá hostless system. [8]

Jednotlivé vrstvy host stacku jsou:

- **Logical Link Control and Adaptation Protocol (L2CAP)** - Jeho hlavní funkcí je předávání paketů do rozhraní HCI, nebo v případě hostless systému předává pakety přímo do LMP. Mezi další funkce patří například rozdělování a poté opětovné sestavení příliš velkých paketů. [8, 10]
- **Radio Frequency Communication (RFCOMM)** - Jedná se o sadu přenosových protokolů, které jsou vytvořeny nad protokolem L2CAP. Tato sada protokolů poskytuje emulované sériové porty RS-232, kterých může být až šedesát na jedno zařízení. RFCOMM umožňuje komunikovat s Bluetooth zařízením přes klasickou sériovou linku. Primárním důvodem používání tohoto standardu je kompatibilita většiny operačních systémů. [8, 10]
- **Service Discovery Protocol (SDP)** - Zajišťuje zařízení možnost být vyhledáno a také vyhledávat jiná zařízení. Zjišťuje, jaké služby podporují párovaná zařízení a jaké parametry je nutno k párování použít. Určuje, které Bluetooth profily jsou podporovány párovaným zařízením, a tedy které profily má při spojení využít. Dále také zajišťuje přiřazení Universally Unique Identifier (UUID) zařízení, které hraje důležitou roli při procesu spojení zařízení. [8, 10]
- **AT commands** - Sada příkazů, pomocí kterých je možné konfigurovat parametry Bluetooth zařízení. Příkazy jsou posílány po sériové lince. [10, 12]
- **Adaptované protokoly** - Mezi běžné používané přijaté protokoly patří Point-to-Point Protocol (PPP), Wireless Application Protocol (WAP) a další [8, 10].

1.3.3 Profily Bluetooth

Následující podkapitola, která se věnuje popisu Bluetooth profilů, čerpá z [13, 14]. Bluetooth profil je standard, který určuje parametry komunikace mezi dvěma nebo více zařízeními, včetně komunikačních protokolů, které budou použity. Každé zařízení Bluetooth podporuje určité profily, aby bylo možno propojit dvě různá Bluetooth zařízení, musí mít tato zařízení stejný profil. Existuje mnoho různých profilů, přičemž každý profil je určen pro konkrétní aplikaci.

Příklad základních Bluetooth profilů:

- **Generic Access Profile (GAP)** - Je základem pro veškeré profily. Definuje vyhledání a vytvoření spojení mezi dvěma zařízeními.
- **Advanced Audio Distribution Profile (A2DP)** - Definuje, jak se přenáší multimediální zvuk z jednoho zařízení do druhého, jako je například hudba z telefonu do bezdrátových sluchátek, nebo přenášení telefonního hovoru do audiosystému automobilu.
- **Video Distribution Profile (VDP)** - Definuje, jak se přenáší videozáznam z jednoho zařízení do druhého. Může se jednat také o živý přenos videa.
- **Audio/Video Remote Control Profile (AVRCP)** - Často se využívá v

kombinaci s profily A2DP a VDP. Je určen především k definici rozhraní pro ovládání televizorů, rádia, či navigačních systémů v automobilech tak, aby se celé zařízení snadno ovládalo dálkovým ovladačem nebo jiným zařízením.

- **Basic Imaging Profile (BIP)** - Definuje přenos obrázků mezi zařízeními a má schopnost měnit velikost obrázků nebo je konvertovat, aby vyhovovaly požadavkům přijímacího zařízení.
- **Basic Printing Profile (BPP)**- Definuje přenos textových zpráv, emailů a dalších souborů ze zařízení (mobilní telefon, digitální kamera) do tiskáren.
- **Hands-Free Profile (HFP)** - Zajišťuje komunikaci mezi mobilními telefony a hands-free sadami v automobilech.
- **Headset Profile (HSP)** - Nejvíce využívaný profil, který definuje přenos zvuku mezi bezdrátovými sluchátky a mobilním telefonem nebo herní konzolí.
- **Human Interface Device profile (HID)** - Definuje podporu pro zařízení lidského rozhraní, jako jsou myši, klávesnice, herní ovladače a další.
- **Serial Port Profile (SPP)** - Emuluje sériovou linku a slouží jako náhrada za RS-232.

1.4 Spojení a párování zařízení pomocí Bluetooth

1.4.1 Adresy Bluetooth zařízení

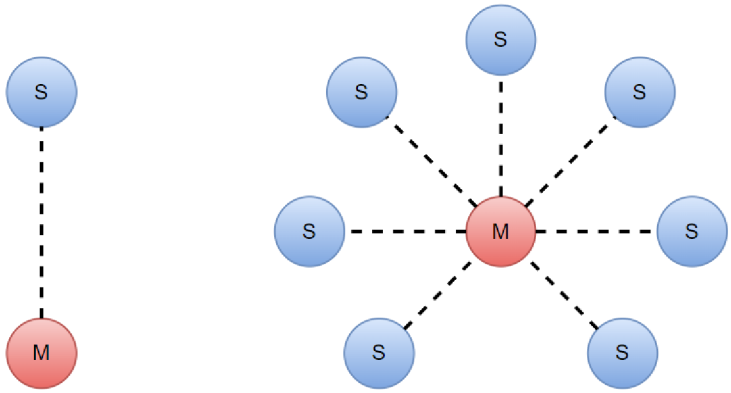
Každé zařízení má svou jedinečnou 48-bitovou adresu, která je zařízení přidělena výrobcem. V některých případech se také označuje jako Bluetooth MAC adresa [15]. Většinou je adresa reprezentována jako 12 hexadecimálních číslic [7], avšak na místo této reprezentace se používají uživatelsky přívětivé názvy zařízení, které si může zvolit sám uživatel [3].

1.4.2 Topologie Bluetooth sítí

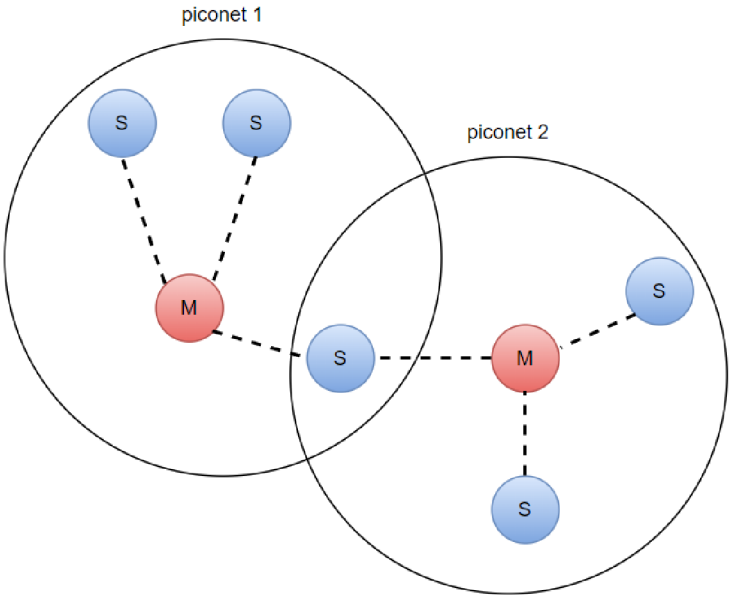
Následující podkapitola, která se věnuje popisu topologie Bluetooth sítí, čerpá z [2, 16]. Existují dvě topologie sítě, které technologie Bluetooth využívá.

Síť piconet může tvořit až osm zařízení, jedno zařízení typu master a až sedm zařízení typu slave (obrázek 1.2). Přičemž zařízení typu master obsluhuje až sedm zařízení typu slave a zařízení typu slave komunikuje pouze s jedním zařízením typu master.

Síť scatternet vznikne propojením více piconetů a je tak schopna pokrýt mnohem větší oblast (obrázek 1.3). Zařízení, které je součástí více piconetů, může přenášet data mezi těmito sítěmi. Avšak základní Bluetooth protokol toto přenášení nepodporuje, musí jej tedy spravovat software daného zařízení, které se v této pozici nachází.



Obr. 1.2: Topologie piconet (M - Master, S - Slave) [2]



Obr. 1.3: Topologie scatternet (M - Master, S - Slave) [2]

1.4.3 Párování a vázání

Následující podkapitola, která se věnuje popisu procesu párování a vázání Bluetooth zařízení, čerpá z [1, 3, 17]. Aby vznikla vazba mezi dvěma zařízeními, je nejprve nutno projít procesem párování. Pro spárování dvou zařízení je potřebná interakce uživatele (přidání Bluetooth zařízení). Po započetí procesu párování si obě zařízení vytvoří společný tajný klíč. Pokud je klíč uložen v obou zařízeních, jsou zařízení

spárována a vzniká mezi nimi vazba.

Dvě již svázaná zařízení budou v dostatečné blízkosti navazovat spojení automaticky, tudíž už nevyžadují interakci uživatele. Uživatel však může vytvořenou vazbu mezi zařízeními odstranit a při jejich dalším spojení bude nutné je znovu spárovat.

1.4.4 Nastavení připojení

Následující podkapitola, která se věnuje popisu procesu připojení Bluetooth zařízení, čerpá z [1, 3, 17]. Navázání spojení dvou zařízení se dělí na části dotazování, připojování a spojení.

V momentě dotazování jedno ze zařízení pošle inquiry požadavek (hledá další zařízení). Jakmile druhé zařízení detekuje tento požadavek, odešle prvnímu zařízení svou adresu.

Jakmile první zařízení příjme adresu zařízení druhého a naopak, nastává proces připojování. Při tomto procesu je důležité, aby obě zařízení znala navzájem své adresy, což je zajištěno procesem dotazování.

Po dokončení procesu připojování jsou zařízení spojena a v tomto stavu jsou připravena na vzájemnou komunikaci, jejíž typ závisí na použitých zařízeních.

2 Bluetooth Low Energy

Bluetooth low energy, někdy označován jako Bluetooth 4.0 nebo Bluetooth Smart, byl uveden na trh v roce 2011. Z názvu této technologie je zřejmé, za jakým účelem byla tato verze Bluetooth vytvořena, a to ze jména pro snížení spotřeby energie. [27]

2.1 Spotřeba energie

S nízkou spotřebou energie můžou aplikace běžet na menších bateriích a po delší dobu, není tedy potřeba baterie tak často měnit nebo dobíjet. Nízké spotřeby energie je dosaženo tím, že připojené zařízení vyčkává ve spánku do té doby, než je iniciována jeho aktivní komunikace. Komunikace mezi zařízeními však trvá jen několik málo milisekund a během této krátké doby si zařízení vymění malý objem dat. [27]

Právě malé přenášené objemy dat stojí za nízkou spotřebou energie. Teoreticky se přenosové rychlosti pohybují od 125 Kb/s do 2 Mb/s [28].

2.2 Hlavní rozdíly BLE a BT

Následující podkapitola popisující hlavní rozdíly mezi technologiemi BLE a BT čerpá z [27, 28, 29, 30]. BLE využívá pro svou komunikaci rádiové vlny a pracuje v bezlicenčním ISM pásmu 2,4 GHz, stejně jako BT classic. Taktéž využívá metodu FHSS, avšak oproti BT classic využívá pouze 40 frekvenčních kanálů, které mají vzájemný rozestup 2 MHz.

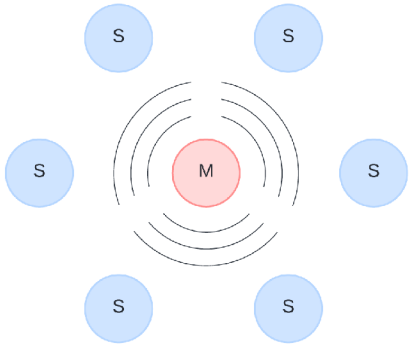
BT classic umožňuje připojit k jednomu zařízení typu master až 7 zařízení typu slave. Kdežto BLE umožňuje připojit k jednomu zařízení typu master libovolný počet zařízení typu slave.

BLE podporuje oproti klasickému BT navíc topologie sítí broadcast (obrázek 2.1) a mesh (obrázek 2.2). Síť broadcast umožňuje vytvořit komunikaci z jednoho na mnoho zařízení. Síť mesh umožňuje vytvořit komunikaci z mnoha na mnoho zařízení.

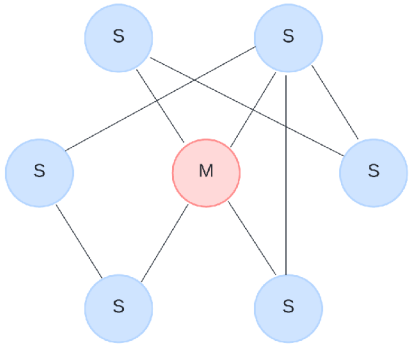
BLE obsahuje funkce, které povolují zařízení určit přítomnost, vzdálenost a směr jiného zařízení. Využívá se zejména v oblasti spotřební elektroniky, v průmyslu a v lékařství. Další rozdíly jsou shrnuty v tabulce 2.1.

Tab. 2.1: Ostatní rozdíly mezi BLE a BT Classic

	BLE	**BT Classic**
Spotřeba energie	1 W	0.01 až 0.5 W
Dosah signálu	do 50 m	do 100 m
Přenos hlasu	Ne	Ano
Maximální proudový odběr	15 mA	30 mA



Obr. 2.1: Topologie broadcast (M - Master, S - Slave) [28]



Obr. 2.2: Topologie mesh (M - Master, S - Slave) [28]

2.3 Generic ATTribute Profile

Tato podkapitola popisující Generic ATTribute Profile čerpá z [31, 32]. Zkráceně GATT, je profil definující způsob výměny dat mezi dvěma BLE zařízeními, která jsou k sobě připojena, za pomocí servisů a charakteristik.

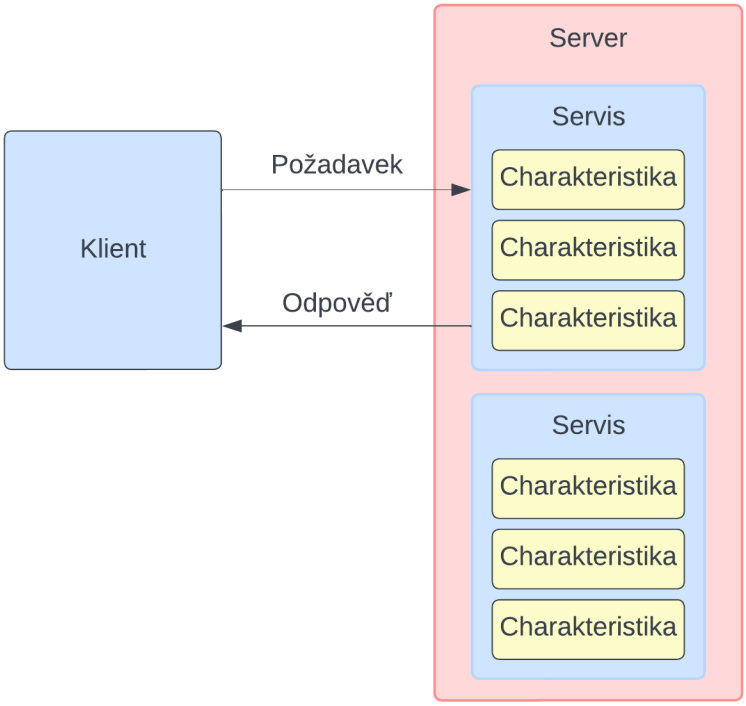
Servisy jsou ucelené logické celky dat, které jsou rozděleny do jednotlivých charakteristik. Servis může obsahovat jednu a nebo více charakteristik a navzájem jsou

od sebe servisy odlišeny unikátním klíčem UUID.

Charakteristiky jsou nejnižším konceptem v GATT procesu sdílení dat. Můžou obsahovat samotnou datovou hodnotu (číslo, znak, ...) a nebo pole hodnot. Záleží na typu BLE zařízení a jeho servisů. Stejně jako servisy jsou od sebe navzájem charakteristiky odlišeny unikátním klíčem UUID.

Zařízení, na kterém je realizován BLE server, obsahuje veškeré servisy a jejich charakteristiky. Pokud chce klient číst nebo posílat data, musí být připojen k BLE serveru a za pomocí UUID se připojit k žádanému servisu, aby bylo možné číst či zapisovat charakteristiky. Tato situace je znázorněna na obrázku 2.3.

K jednomu GATT servisu může být připojeno více klientů najednou. Pokud si však chtějí klienti posílat data mezi sebou, musí být na serveru implementovaná vlastní datová schránka, přes kterou všechna data klientů prochází.



Obr. 2.3: Interakce serveru s klientem [32]

3 QT Framework

Qt je multiplatformní C++ framework pro vytváření aplikací a grafických uživatelských rozhraní. Software vyvinutý v Qt běží na všech hlavních desktopových, mobilních a embedded platformách, jelikož podporuje princip Write Once, Compile Anywhere (WOCA). Vývoj Qt zajišťuje firma The Qt Company, poskytují řadu specializovaných nástrojů pro vývoj aplikací a grafických uživatelských rozhraní jako Qt Creator, Qt Quick, Qt Design Studio a další. [33]

V rámci této práce byl používán software Qt Creator, který je popsán v následující podkapitole.

3.1 QT Creator

Qt Creator je integrované vývojové prostředí využívané pro vývoj a design aplikací pro počítačové, embedded a mobilní platformy. Při instalaci je možno si vybrat ze všech dostupných verzí Qt frameworku, dostupných knihoven a dodatků k softwaru. [34]

3.1.1 Možnosti projektu

Qt Creator využívá k sestavení projektu systémy CMake, qmake nebo Qbs, ze kterých čerpá informace pro nápovědu, navigaci a úpravy při psaní zdrojového kódu. Tak stejně získává informace pro nasazení a spouštění aplikací, další možná nastavení se ukládají do nastavení projektu. [34]

Při vytvoření nového projektu je nutné si vybrat druh aplikace, uživatelské rozhraní založené na Qt Quick nebo Qt Widgetech. Následně si zvolit programovací jazyk, C++ nebo Python, který bude použit pro implementaci aplikační logiky. [34]

V rámci této bakalářské práce je pro uživatelské rozhraní mobilní aplikaci použit Qt Quick a pro aplikační logiku programovací jazyk C++.

3.1.2 Qt Quick

Qt Quick modul je standardní knihovnou pro psaní QML aplikací. Popis jazyka QML je v kapitole 3.2. Tento modul poskytuje všechny potřebné typy, které slouží pro vytváření uživatelského rozhraní pomocí jazyka QML, to je docíleno díky QML API. Modul dále poskytuje C++ API, díky které je možné QML aplikace propojit s C++ zdrojovým kódem. [35]

3.2 QML

QML je multiparadigmatický jazyk pro vytváření dynamických aplikací. Za pomoci QML se deklarují bloky uživatelského rozhraní, nastavují se jejich vlastnosti, které definují chování aplikace. Aplikaci lze dále rozšířit pomocí skriptů napsaných v JavaScriptu, který je podmnožinou QML jazyka. QML intenzivně využívá Qt, což umožňuje využití jeho funkcí a typů přímo v QML aplikacích. [36]

3.2.1 Deklarace objektů

Tato podkapitola popisující deklarace objektů v jazyce QML čerpá z [37, 38]. Syntaktický blok QML kódu definuje strom QML objektů, které mají být vytvořeny. Deklarace objektu popisuje typ objektu a jeho atributy, které mu mají být přiděleny. Každému objektu můžou být deklarovány vnořené podřízené objekty.

Příklad deklarace jednoduchého QML objektu:

```
import QtQuick

Rectangle{
    width: 100
    height: 100
    color: "red"
}
```

Tímto způsobem lze vytvořit objekt typu Rectangle, tedy obdélník, který bude mít rozměry 100x100 pixelů a bude červené barvy. Vytvoření objektu předchází připojení knihovny QtQuick.

Příklad deklarace QML tlačítka:

```
import QtQuick

Button{
    text: "Potvrdit"
    onClicked: model.submit()
}
```

Tímto způsobem lze vytvořit objekt typu Button, tedy tlačítko, který přes signál onClicked může volat například C++ metody.

Příklad deklarace podřízeného QML objektu:

```
import QtQuick

Rectangle{
    width: 200
    height: 200
    color: "red"

    Text {
        anchors.centerIn: parent
        text: "Test␣text"
    }
}
```

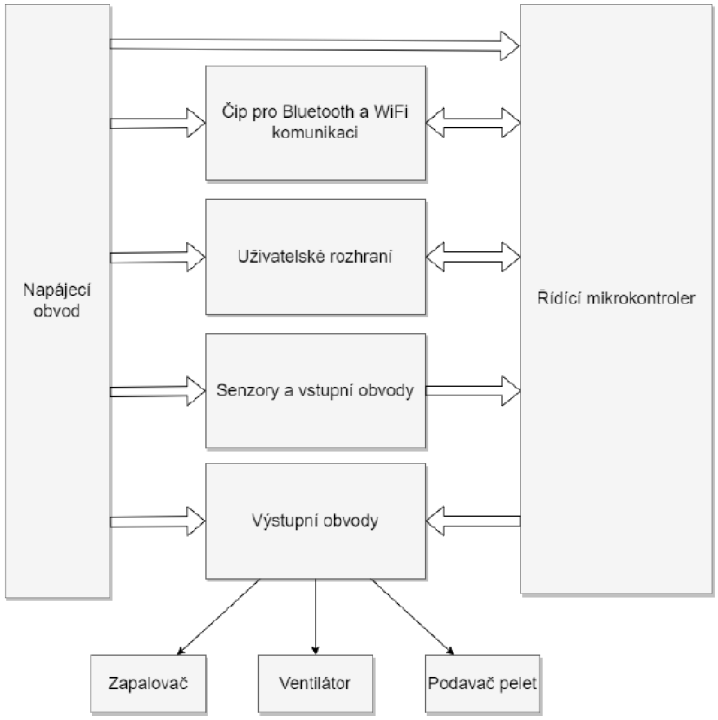
Tímto způsobem lze vytvořit objekt typu Rectangle, tedy obdélník, který bude mít rozměry 200x200 pixelů a bude červené barvy. V objektu Rectangle je vytvořen podřízený objekt typu Text, který má nastavenou vlastnost ukotvení, díky které se bude nacházet v centru svého rodiče, tedy objektu Rectangle.

4 Zadaný přípravek s modulem ESP32

Praktická část navazuje a rozšiřuje bakalářskou práci [18]. Hlavním cílem bakalářské práce je navrhnout uživatelskou mobilní aplikaci, která bude kompatibilní s běžnými mobilními operačními systémy. Následně pak otestovat a demonstrovat obousměrnou bezdrátovou komunikaci mezi aplikací a zadaným hardwarem.

4.1 Popis přípravku

Na obrázku 4.1 je blokové schéma přípravku, pro realizaci uživatelské mobilní aplikace, která bude schopna bezdrátového přenosu, je zapotřebí pouze Řídicí mikrokontrolér a čip pro Bluetooth a WiFi komunikaci. Tudíž jiným částem přípravku se v této bakalářské práci nevěnuji, i když jsou některé z nich nezbytné pro jeho fungování. Veškerá kritéria výběru těchto částí jsou zohledněna v [18].

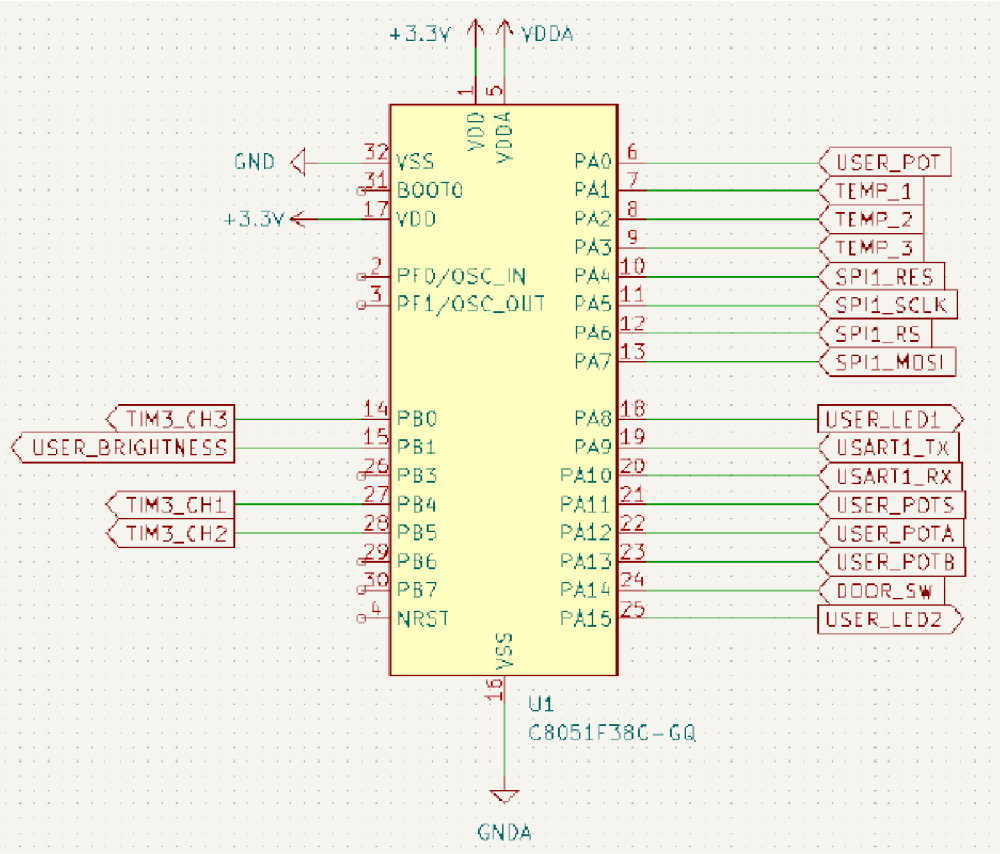


Obr. 4.1: Blokové schéma zapojení přípravku [18]

Na přípravku se nachází mikrokontrolér STM32F030K6T6 [22]. Ten byl zvolen s ohledem na požadavky a počet připojovaných periferních zařízení. Jako jsou například počet PWM výstupů, počet digitálních vstupů a výstupů, počet analogových

vstupů a podporované komunikační standardy. Dalšími požadavky byla dostupnost a přijatelná cena mikrokontroléru. [18]

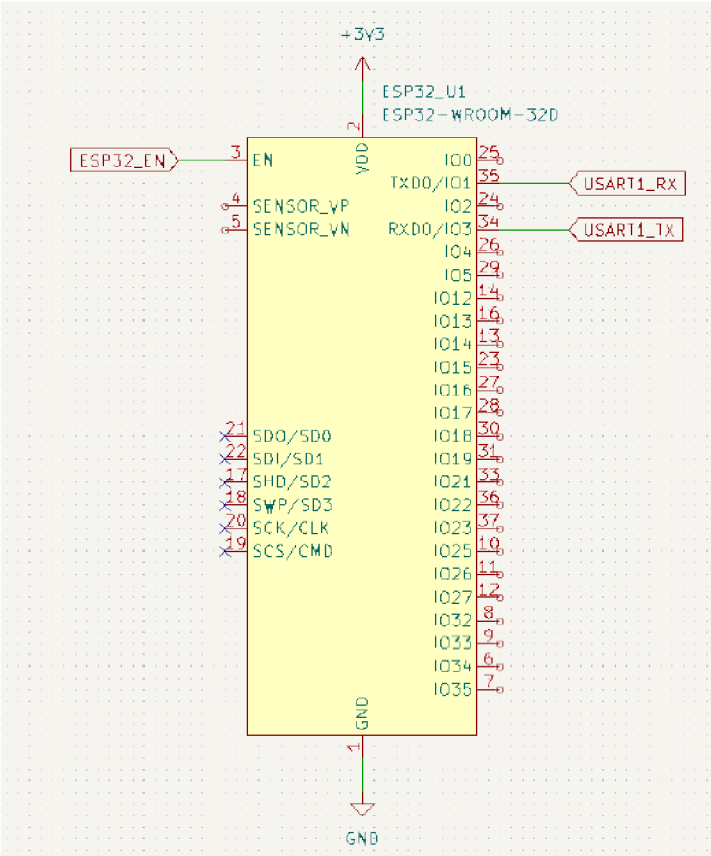
Mikrokontrolér komunikuje s modulem pro bezdrátovou komunikaci pomocí sériového rozhraní UART. Mikrokontrolér však disponuje sériovým rozhraním USART, které lze nakonfigurovat pro komunikaci pomocí rozhraní UART či SPI. Na obrázku 4.2 je zobrazeno schéma zapojení mikrokontroléru, ze kterého lze vyčíst, že modul pro bezdrátovou komunikaci je připojen na piny PA9 a PA10 mikrokontroléru.



Obr. 4.2: Schéma zapojení mikrokontroléru STM32F030K6T6 [18]

4.1.1 Použitý modul pro bezdrátovou komunikaci

Na přípravku je použitý modul ESP32-WROOM32-UE [19]. Schéma zapojení modulu je na obrázku 4.3, avšak jak si lze povšimnout, na schématu se nachází modul ESP32-WROOM32-D. Jedná se o chybu v původní práci [18], kterou lze zanedbat, jelikož moduly ESP32-WROOM32-D a ESP32-WROOM32-UE mají stejné rozložení pinů.[19, 20]



Obr. 4.3: Schéma zapojení ESP32-WROOM-32UE [18]

Tab. 4.1: Popis pinů ESP32-WROOM-32UE důležitých pro aplikaci [19]

Číslo	Označení	Funkce
1	GND	Uzemnění
2	3V3	Napájení 3.3 V
3	EN	Povoluje modul
25	IO0	Přepíná modul do boot módu
27	IO16	U2RXD
28	IO17	U2TXD
34	RXD0	U0RXD
35	TXD0	U0TXD
38	GND	Uzemnění

V tabulce 4.1 je přehled pinů modulu ESP32-WROOM-32UE, které budou důležité pro realizaci aplikace. Zde byl objeven první problém původního návrhu zapojení modulu. Jak je zmíněno v kapitole 4.1, mikrokontrolér je propojen s modulem za pomocí sériového rozhraní UART. Signály z mikrokontroléru USART1-RX

a USART1-TX (obrázek 4.3 a 4.2) vedou na piny modulu TXD0 a RXD0, jenže ty využívá ESP32 AT Firmware pro tisk zpráv a také se využívá pro nahrání samotného firmwaru. Piny modulu IO16 a IO17 (UART2) slouží pro posílání AT příkazů a přijímání AT odpovědí. [21]

Druhým problémem v původním návrhu je, že pin IO0, který slouží k přepnutí modulu do boot módu [19], není na přípravku vůbec zapojen. Dostat modul do boot módu je nutné pro nahrání nového AT firmwaru.

Aby bylo možné uživatelskou mobilní aplikaci pro obousměrný bezdrátový přenos realizovat, musely být výše uvedené problémy vyřešeny. Postup řešení je v podkapitole 4.1.2.

4.1.2 Úpravy přípravku

Aby bylo možné posílat AT příkazy a přijímat AT odpovědi přímo z mikrokontroléru do modulu, bylo nutné přerušit původní cestu signálů USART1-RX a USART1-TX. Nově byly tyto dva signály k modulu připájeny následovně: signál USART1-RX na pin IO17 a signál USART1-TX na pin IO16.

Tím, že se původní cesta signálů USART1 přerušila, tak piny modulu RXD0 a TXD0 zůstaly odpojené. K pinům RXD0, TXD0 a GND (číslo 38) byly připájeny drátky, zakončeny dutinkovou lištou. Důvodem tohoto kroku byla možnost připojení externího zařízení (počítače) k modulu. Externí zařízení je schopno přes převodník z USB na sériovou linku konfigurovat modul ESP32-WROOM-32UE.

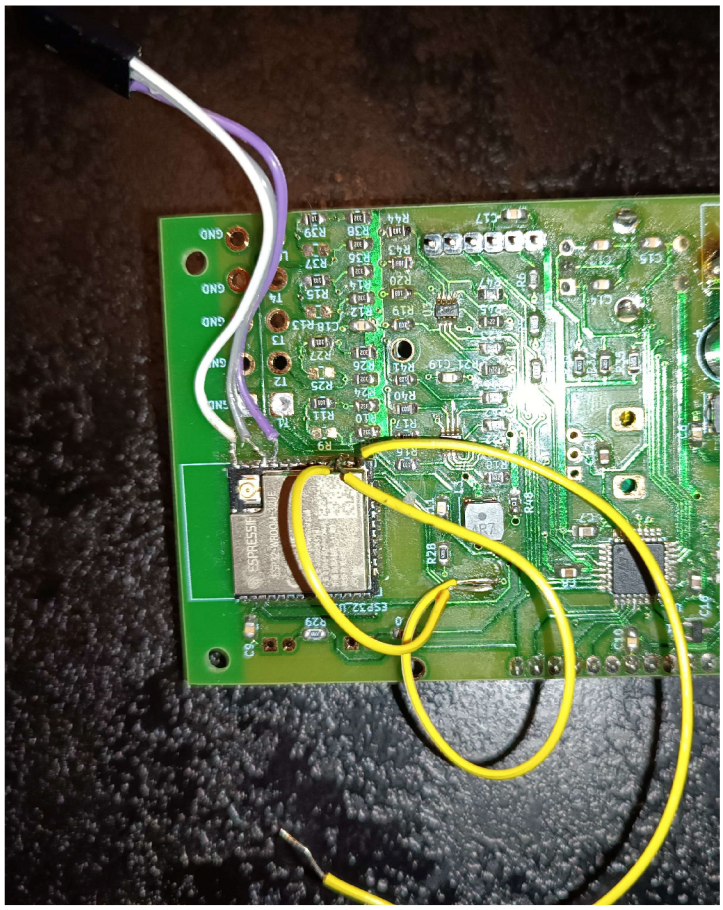
Poslední fyzickou úpravou bylo připájení drátku k pinu IO0. Druhý konec drátku zůstal nepřipájený. ESP32-WROOM-32UE se přepne do boot módu následující sekvencí: pin IO0 se spojí s GND, signálem ESP32-EN se pin modulu EN shodí do 0 a následně zpět do 1 (reset), pin IO se rozpojí.

Výše uvedené úpravy nejsou ideální a v budoucnu by bylo vhodné zvolit optimálnější metodu úpravy desky plošného spoje. Nedokonalost úprav by v tomto případě mohla způsobit nefunkčnost aplikace. Toto řešení bylo zvoleno po konzultaci s vedoucím práce jako prozatím dostačující. Na obrázku 4.4 je zachycena fotografie části přípravku s modulem ESP32-WROOM-32UE po provedených úpravách.

4.2 Nahrání AT Firmwaru

V této podkapitole je popsán proces nahrání AT Firmwaru do modulu pro bezdrátovou komunikaci ESP32-WROOM-32UE.

V první řadě je nutno si stáhnout AT Firmware ESP32-WROOM-32-AT-V3.2.0.0.zip dostupný z [25]. Zip soubor s AT firmwarem se nachází v příloze A, obsahuje binární



Obr. 4.4: Fotografie modulu ESP32-WROOM-32UE po úpravách

soubory s daty firmwaru a také soubor download.config, ve kterém se nachází adresy začátků těchto binárních souborů ve flash paměti ESP32.

Pro nahrání firmwaru je určen UART0 [21], na který připojíme převodník USB na sériovou linku, ten je zapojený na jednom z portů COM. K samotnému nahrání firmwaru do ESP32 byl použit nezávislý nástroj založený na Pythonu esptool.py. Jedná se o kompletní sadu nástrojů pro práci s ESP moduly. [26] Nástroj se spouští pomocí příkazové řádky. V příloze A se nachází soubor flashSequence.txt, který obsahuje řetězec, jež spustí nahrání firmwaru, ovšem jen za předpokladu, že se ESP32 nachází v boot módu. Jak vypadá výpis na konzolu během procesu nahrávání firmwaru je v příloze A v souboru flashLogOutput.txt.

Jakmile je firmware do modulu nahrán, je nutno modul restartovat. Pokud byl firmware nahrán správně, tak se po restartu na sériovém monitoru zobrazí informace o nahraném firmwaru [21]. Tyto informace jsou uvedeny v příloze A v souboru checkAtFirmware.txt.

4.3 Konfigurace modulu ESP32

Komunikace s modulem ESP32 na zadaném přípravku je realizována ve vývojovém prostředí STM32 Cube IDE, které je založeno na platformě Eclipse a je primárně optimalizováno pro mikrokontroléry STM32. Tato platforma umožňuje konfigurovat periferie, generovat kód, kompilovat kód a nabízí funkci ladění, která je využitelná zejména při testování komunikace. [23]

4.3.1 Otestování AT Firmwaru

Po všech úpravách přípravku, které jsou uvedeny v podkapitole 4.1.2, a po nahrání správného firmwaru, bylo nutno otestovat, zda je možné modul za pomocí AT příkazů nastavit. AT příkazy a odpovědi jsou vysílány a přijímány mikrokontrolérem STM32 po sériovém rozhraní UART.

Pokud modulu ESP32 pošleme AT příkaz: "AT", očekáváme AT odpověď: "OK", pokud AT příkazy fungují a "ERROR", pokud nefungují. Jedná se o základní AT příkaz, pomocí kterého můžeme otestovat, zda modul reaguje na AT příkazy. AT odpovědi jsou vypsány na LCD displej, který se na zadaném přípravku nachází.

4.3.2 Konfigurace ESP32 v roli serveru navazující SPP spojení

Po úspěšném otestování AT Firmwaru bylo nutno modul ESP32 nakonfigurovat do role BLE serveru, který naváže s klientem SPP spojení, ve kterém si budou schopny vyměňovat data. Klientem je v tomto případě mobilní aplikace, která je popsána v kapitole 5. Postup pro konfiguraci modulu jsem čerpal z [21].

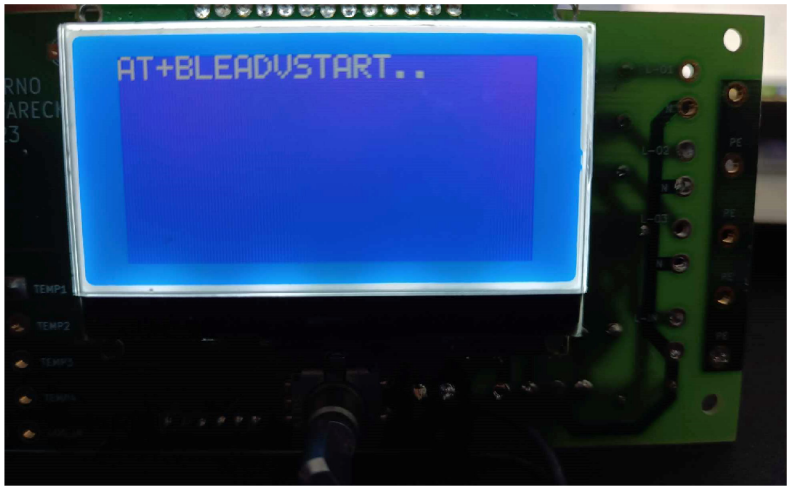
V tabulce 4.2 je popsána sekvence AT příkazů pro konfiguraci modulu ESP32. Nejdříve je nutné BLE na modulu inicializovat do role serveru, následuje vytvoření GATTS servisů a jejich spuštění. Dále se modulu nastaví data pro vysílání a parametry Security Manager Protocolu. Následuje nastavení párovacího klíče, který bude vyžadován při pokusu o připojení od neznámého zařízení. Modul ESP32 spustí vysílání, pokud se bude o připojení pokoušet neznámé zařízení, bude vyzváno k zadání párovacího klíče. V případě již známého zařízení tento proces nenastává. Po úspěšném připojení klienta k serveru modul nastaví parametry Serial Port Profile (SPP) komunikace a přechází do SPP módu, ve kterém je modul připraven posílat a přijímat data.

Jestliže klient pošle přes SPP spojení data (text, číslo, znak, ...), server je po přijetí pošle pomocí sériového rozhraní UART mikrokontroléru STM32 a ten přijatou zprávu vytiskne na LCD displej přípravku.

Projekt v prostředí STM32 Cube IDE, který slouží pro konfiguraci modulu ESP32, jež je popsána v této podkapitole výše, se nachází v příloze B.

Tab. 4.2: Sekvence AT příkazů pro konfiguraci modulu ESP32 BLE SPP serveru [21]

AT+BLEINIT=2	Inicializuje BLE v roli serveru
AT+BLEGATTSSRVCRE	Generic Attribute Server (GATTS) vytvoří BLE servisy
AT+BLEGATTSSRVSTART	Generic Attribute Server (GATTS) spustí všechny vytvořené servisy
AT+BLEADVDATAEX=<>	Nastaví potřebná data pro vysílání(jméno, uuid, ...) a začne vysílat
AT+BLESECPARAM=<>	Nastaví parametry BLE Security Manager Protocolu (SMP)
AT+BLESETKEY=<>	Nastaví BLE statický párovací klíč, který je platný pro všechna BLE připojení
AT+BLEADVSTART	Spustí vysílání
AT+BLEENC=<>	Pokud nejsou zařízení spárována, vyvolá požadavek na párování
AT+BLESPPCFG=<>	Nastaví nebo resetuje parametry BLE Serial Port Profile (SPP)
AT+BLESPP	Vstoupí do BLE BLE Serial Port Profile (SPP) módu



Obr. 4.5: Fotografie přípravku vyčkávajícího na párování/připojení

Po úspěšné inicializaci BLE, GATT a nastavení potřebných parametrů pro vysílání začne modul vysílat a čeká na připojení klienta. Tato situace je zachycena na obrázku 4.5.

Po úspěšném spárování neznámého zařízení, a nebo po připojení již spárovaného

Obr. 4.6: Fotografie přípravku v SPP módu

zařízení modul nastaví parametry SPP profilu a vstoupí do SPP módu, ve kterém může posílat a přijímat data. Tato situace je zachycena na obrázku 4.6.

5 Vývoj mobilní aplikace

V této kapitole je popsán vývoj mobilní aplikace, která je kompatibilní s běžným mobilním operačním systémem, v tomto případě se jedná o systém Android. Aplikace využívá Bluetooth Low Energy pro bezdrátovou komunikaci, je schopna skenovat dostupná vysílající BLE zařízení v okolí a připojit se k nalezenému zařízení dle volby uživatele. V rámci této práce komunikuje pouze s modulem ESP32, který je nakonfigurován do role SPP serveru, jak je popsáno v kapitole 4.3.2. Aplikace je tedy v tomto případě klientem připojujícím se k serveru a má za úkol přijímat data z modulu, zobrazovat je do grafu, a v případě uživatelského zásahu zaslat data po sériovém spojení modulu.

Pro vývoj aplikační logiky a uživatelského rozhraní aplikace byl zvolen software Qt Framework, konkrétně nástroj Qt Creator, oba tyto softwary jsou popsány v kapitole 3 a 3.1. Jako druh aplikace byl zvolen Qt Quick, který podporuje jazyk QML a C++ API, díky které lze tyto dva jazyky propojit. Výhody a způsob použití QML a Qt Quick jsou popsány v kapitole 3.1.2 a 3.2. Projekt mobilní aplikace naprogramované v Qt Creatoru se nachází v příloze C.

Qt má velmi přehledně zpracovanou dokumentaci [39], ve které jsou popsány jednotlivé knihovny a jejich metody, mají diskuzní fórum [40], které je neustále aktivní, a poměrně rozsáhlou knihovnu ukázkových aplikací [41]. Těchto zmíněných zdrojů bylo využito při vývoji aplikace.

5.1 Architektura aplikace

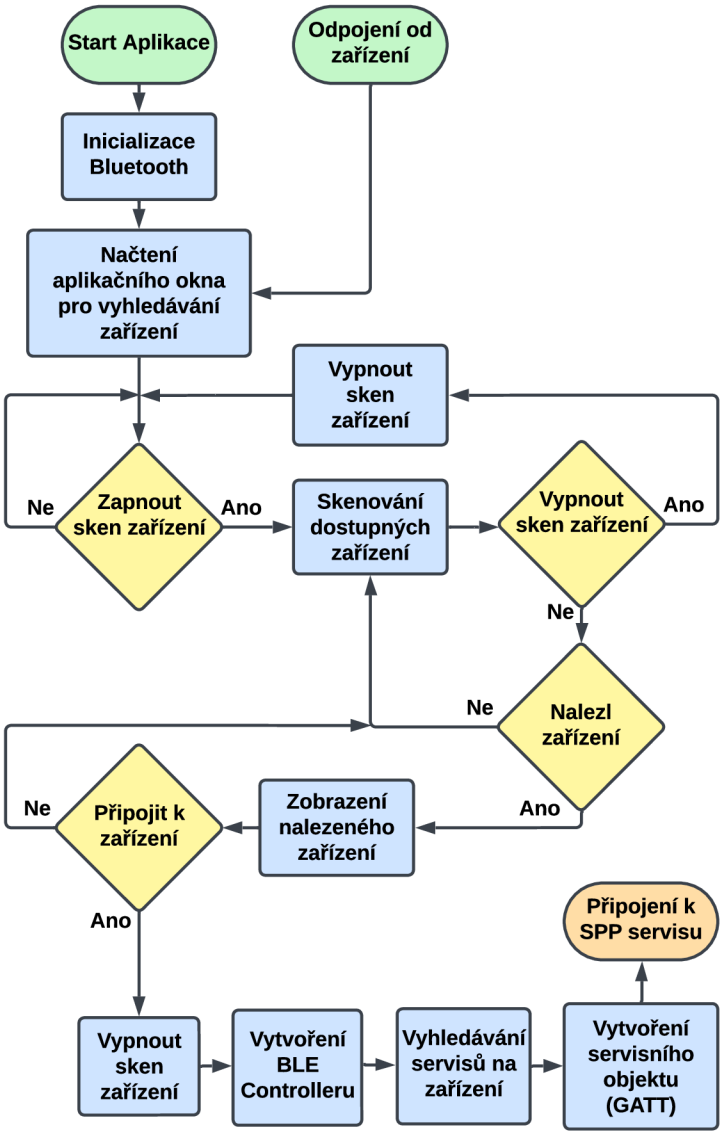
V této podkapitole je popsána architektura aplikace, ve smyslu propojení uživatelského rozhraní, jednotlivých tříd, jejich metod a tříd volaných z dostupných knihoven Qt frameworku. Jednotlivé pohledy na architekturu aplikace jsou rozděleny do podkapitol, ve kterých je z daného úhlu pohledu architektura rozebrána.

5.1.1 Aplikační logika

Aplikační logika je z důvodu lepší přehlednosti rozdělena do dvou vývojových diagramů.

Na obrázku 5.1 je vývojový diagram znázorňující algoritmus pro vyhledávání a připojení BLE zařízení. Po startu aplikace, nebo po odpojení od zařízení, může uživatel za pomocí tlačítek (popsány v kapitole 5.3) zapnout nebo vypnout skenování zařízení. Nalezená zařízení budou zobrazena pod sebou na displeji. Jakmile uživatel klikne na políčko se zařízením, ke kterému se chce připojit, skenování zařízení se

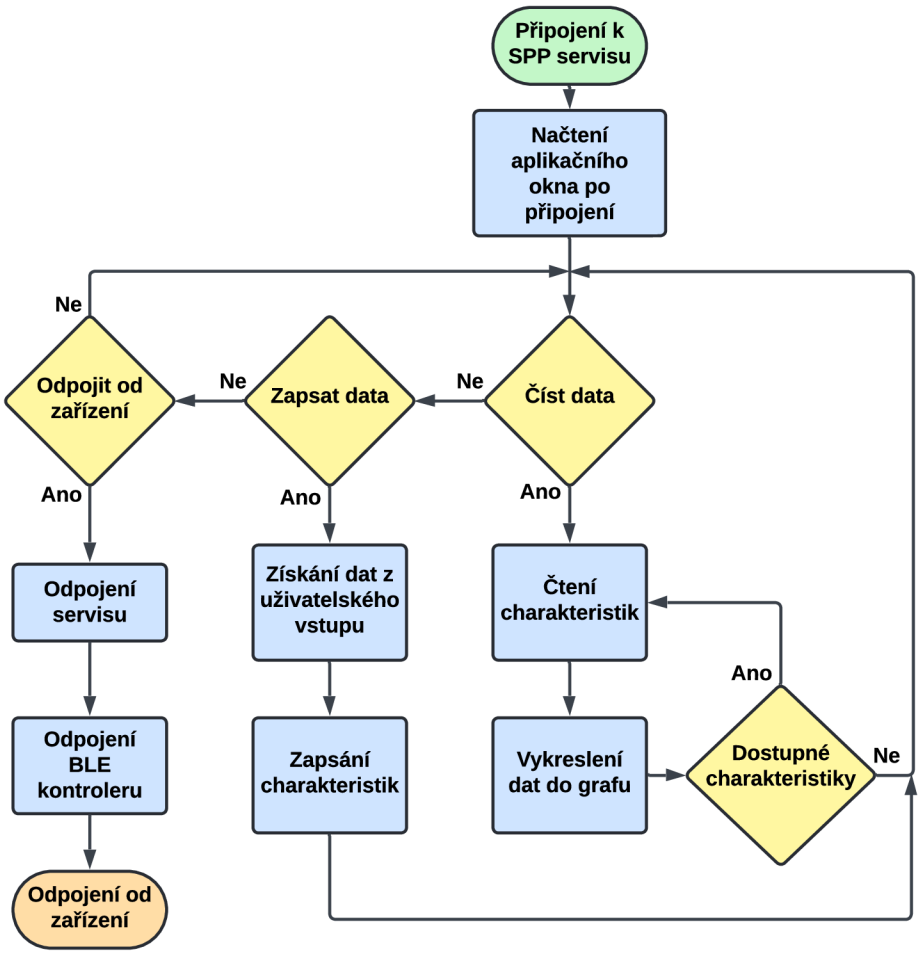
přeruší, vytvoří se BLE kontrolér, vyhledá se na zařízení příslušný servis a pokud je nalezen, aplikace naváže se zařízením spojení.



Obr. 5.1: Vývojový diagram pro vyhledávání a připojení BLE zařízení

Na obrázku 5.2 je vývojový diagram, který popisuje algoritmus výměny dat mezi zařízeními po připojení k SPP servisu. Uživatel může opět za pomocí tlačítek ovládat akce aplikace. Možnosti uživatele jsou zapisování a čtení dat, nebo se od zařízení odpojit. V případě čtení dat se aplikace začne dotazovat na příchozí charakteristiky, pokud jsou charakteristiky dostupné, aplikace je přečte a zobrazí hodnotu do grafu. V případě zapisování dat může uživatel zadat libovolná data (text, číslo, znak, ...)

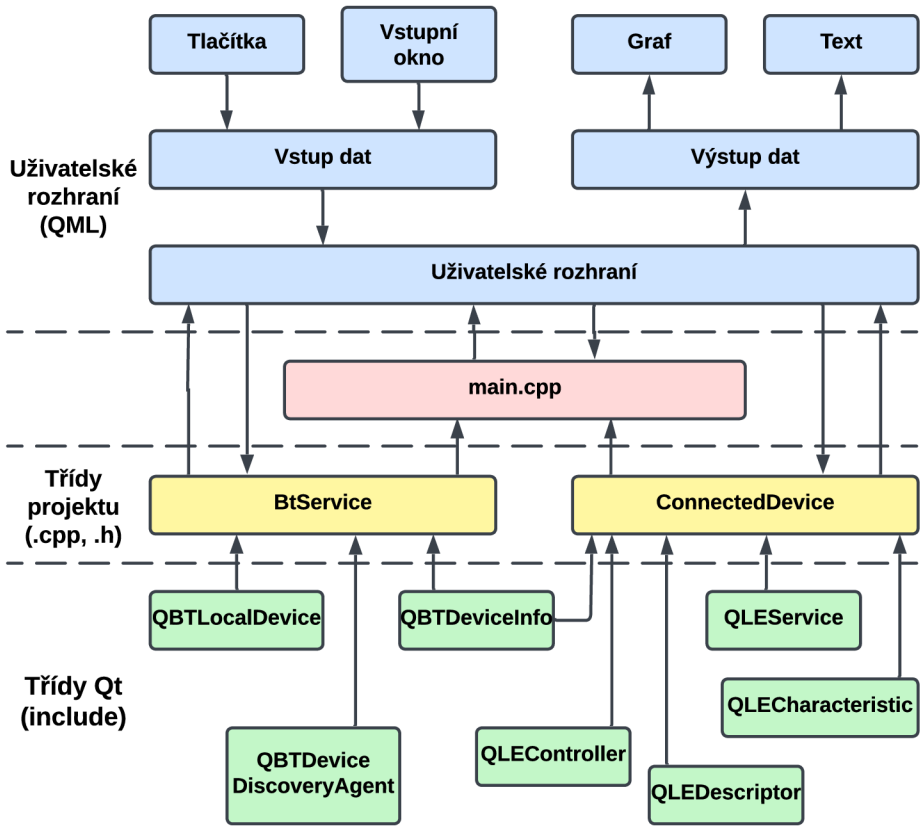
do vstupního pole a po stisku tlačitka dojde k zapsání charakteristiky s těmito daty, data se tedy odešlou na připojené zařízení.



Obr. 5.2: Vývojový diagram pro obousměrnou komunikaci mezi aplikací a připojeným zařízením

5.1.2 Architektura tříd a souborů

Na obrázku 5.3 je vývojový diagram, který zachycuje propojení uživatelského rozhraní (QML) se souborem main.cpp a se třídami BtService a ConnectedDevice. Instance těchto tříd se nacházejí jak v QML souboru uživatelského rozhraní, tak v souboru main.cpp, podrobněji je propojení QML a C++ popsáno v kapitole 5.2. Dále jsou na digramu zobrazeny Qt třídy, které byly v rámci tříd aplikace použity pro realizaci aplikační logiky.



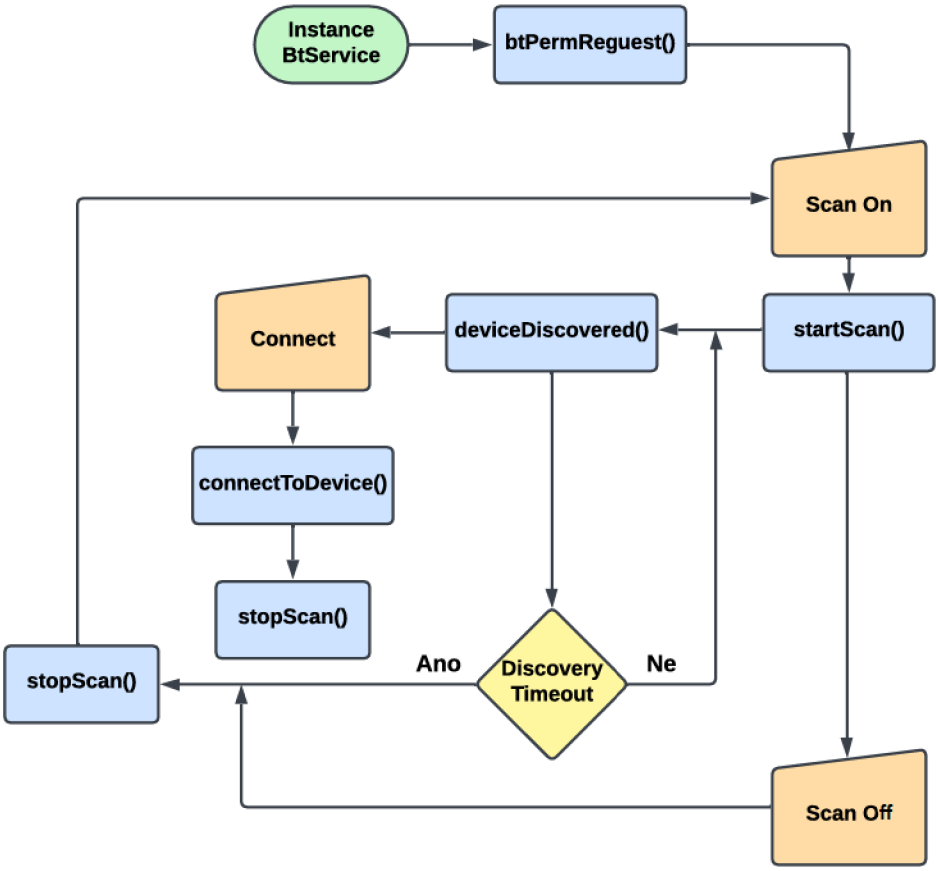
Obr. 5.3: Diagram propojení uživatelského rozhraní s Qt třídami a třídami aplikace

5.2 Popis funkčnosti aplikace

V této podkapitole je popsána funkčnost metod, které jsou nezbytné pro vyhledávání zařízení, zajištění BLE připojení k serveru, vzájemnou výměnu dat a propojení s uživatelským rozhraním. Dále v kapitole 5.3 je popsána funkčnost samotného uživatelského rozhraní.

5.2.1 Třída BtService

Třída BtService slouží pro inicializaci BT lokálního zařízení, na kterém je aplikace spuštěna. Zajišťuje skenování ostatních BLE zařízení, jejich uložení a vypsání na obrazovku. Při žádosti uživatele o připojení k jinému zařízení zjišťuje, zda zařízení vyžaduje párování, pokud ano, je uživatel vyzván k zadání párovacího klíče. Následně se volá metoda setDevice z instance třídy ConnectedDevice, kde dochází k navázání spojení aplikace s jiným zařízením.



Obr. 5.4: Zobrazení propojení jednotlivých metod a akcí (tlačítek) třídy btService

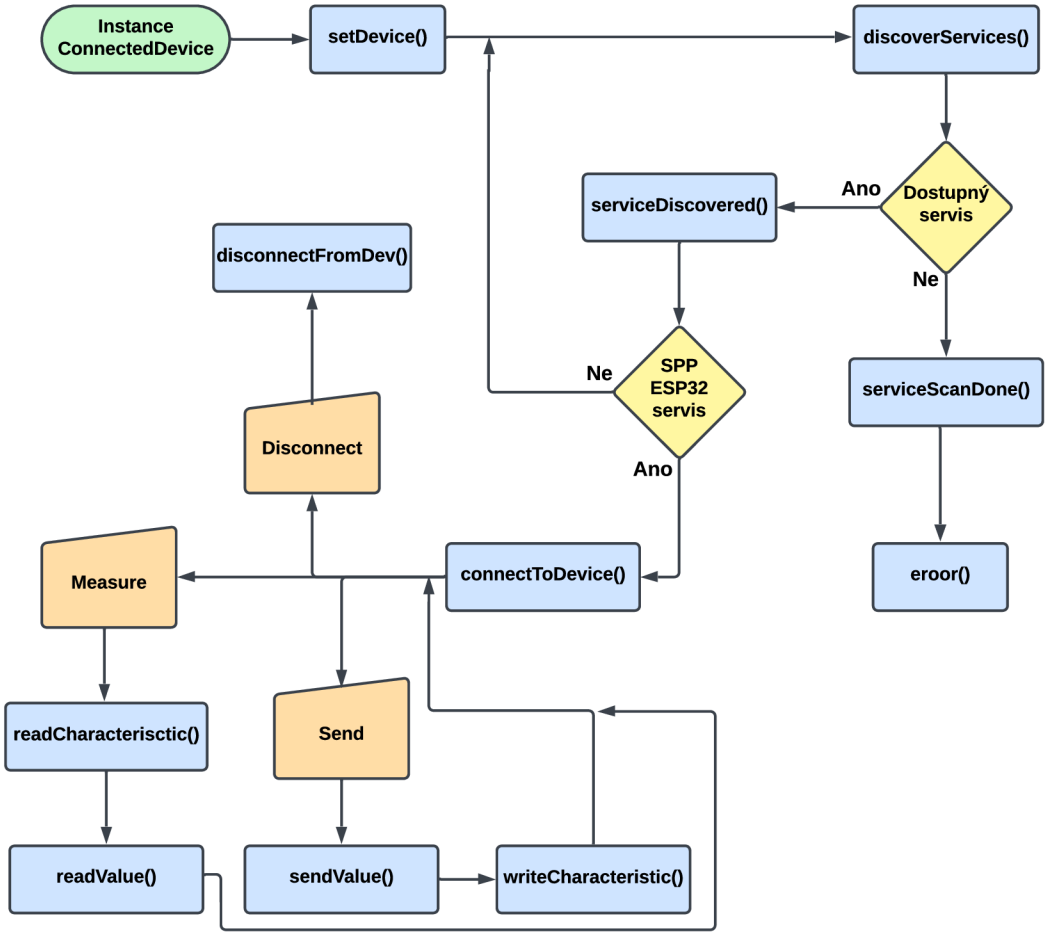
Na obrázku 5.4 je zobrazen diagram logického propojení jednotlivých metod a akcí (tlačítek) v rámci třídy btService.

Funkce jednotlivých metod jsou:

- **btPermRequest()** - metoda, která inicializuje BT a zjišťuje, zda má od zařízení povolení BT využívat
- **startScan()** - zahájí skenování dostupných vzdálených zařízení
- **stopScan()** - zastaví skenování dostupných vzdálených zařízení
- **deviceDiscovered()** - volá se v případě, že je při skenování objeveno nové zařízení, údaje objeveného zařízení se uloží do listu, který je poté zobrazován na displej
- **connectToDevice()** - volá se v případě pokusu o připojení k zařízení a slouží ke spárování se vzdáleným zařízením, volá se jen v případě, že zařízení nejsou spárována

5.2.2 Třída ConnectedDevice

Při zavolání metody setDevice se nejprve vytváří BLE kontrolér, který na vzdáleném zařízení oskenuje všechny jeho servisy. V případě že narazí na SPP servis ESP32, který má své UUID, tak se aplikace k tomuto servisu, a tedy zařízení, připojí. Dále tato třída zajišťuje obousměrnou výměnu dat mezi aplikací a vzdáleným zařízením. A v poslední řadě se stará o ukončení spojení aplikace se vzdáleným zařízením po žádosti uživatele.



Obr. 5.5: Zobrazení propojení jednotlivých metod a akcí (tlačítek) třídy connectedDevice

Na obrázku 5.5 je zobrazen diagram logického propojení jednotlivých metod a akcí (tlačítek) v rámci třídy connectedDevice.

Funkce jednotlivých metod jsou:

- **setDevice()** - metoda se volá po dotazu na připojení k zařízení, inicializuje low energy kontrolér

- **discoverServices()** - metoda low energy kontroléru, jež spustí skenování dostupných servisů na zařízení
- **serviceDiscovered()** - metoda se volá při nalezení servisu na zařízení a slouží ke kontrole, zda se jedná o SPP ESP32 servis
- **serviceScanDone()** - volá se v případě, že mezi dostupnými servisy se nenachází SPP servis ESP32
- **error()** - volá se v případě dokončení skenování servisů na zařízení z důvodu nenalezení SPP servisu ESP32
- **connectToDevice()** - metoda low energy kontroléru, jež se volá po úspěšném nalezení SPP servisu na ESP32 a slouží k připojení k servisu a zařízení
- **disconnectFromDev()** - metoda se volá po stisku tlačítka Disconnect a slouží k odpojení od připojeného zařízení
- **readCharacteristic()** - metoda low energy servisu, která čte příchozí charakteristiky od připojeného zařízení
- **writeCharacteristic()** - metoda low energy servisu, která zapisuje charakteristiky na připojené zařízení
- **readValue()** - metoda, která periodicky čte charakteristiky pomocí readCharacteristic() od připojeného zařízení a je volána pomocí tlačítka Measure
- **sendValue()** - metoda, která zapisuje charakteristiku pomocí writeCharacteristic() na připojené zařízení a je volána pomocí tlačítka Send, zapisovaná hodnota je zadávána do vstupního okna

5.2.3 Soubor main.cpp

Soubor main.cpp je hlavním vstupním bodem celé aplikace. V tomto souboru se inicializuje samotná aplikace a vytváří se instance třídy BtService, čímž se inicializuje BT. V dalším kroku načítá QML soubor main.qml, kde se nachází uživatelské rozhraní aplikace.

Pomocí následujícího kódu propojuje uživatelské rozhraní (QML) s C++ třídami BtService a ConnectedDevice.

```
BtService btService;
ConnectedDevice connectedDevice;

QQmlApplicationEngine engine;
engine.setInitialProperties({
    {u"btService"_s, QVariant::fromValue(&btService)},
    {u"connectedDevice"_s, QVariant::fromValue(&connected
    Device)}
});
```

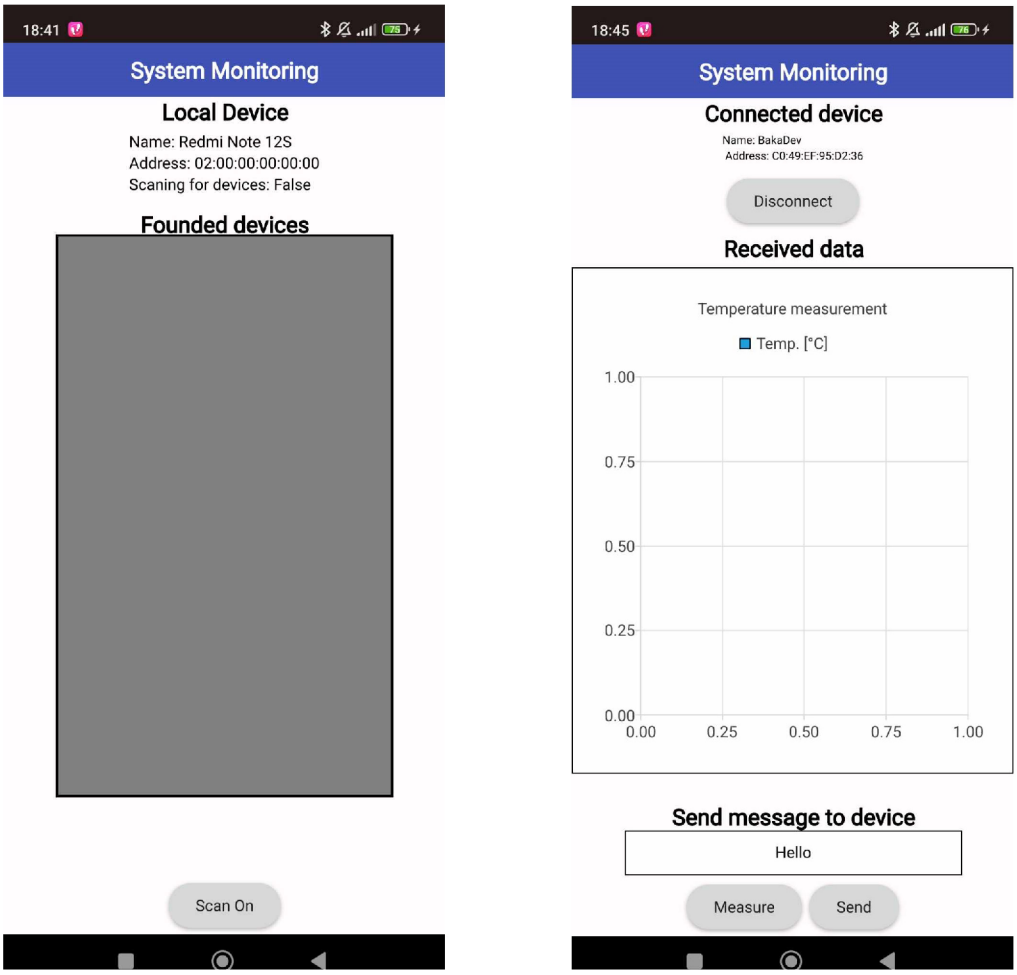
V posledním kroku tento soubor spouští hlavní smyčku aplikace.

5.3 Uživatelské rozhraní

V této podkapitole je popsán vzhled uživatelského rozhraní. Uživatelské rozhraní bylo vzhledem k složitosti aplikace zvoleno co nejjednodušší. Tedy rozhraní obsahuje pouze nezbytné údaje o lokálním a připojeném zařízení, vyhledaných zařízeních, zobrazení příchozích a odchozích dat a tlačítka zajišťující přepínaní stavů aplikace.

Vzhled uživatelského rozhraní je zobrazen na obrázku 5.6. Lze si všimnout, že se skládá celkem ze dvou obrazovek, kdy levá obrazovka se zobrazí hned po načtení aplikace. V horní liště obrazovky se nachází název aplikace, System Monitoring. Na této obrazovce se v horní části nachází informace o zařízení, na kterém běží aplikace (jméno, adresa, stav skenování zařízení), zbytek obrazovky zabírá kontejner na nalezená zařízení. V tomto kontejneru se zobrazí všechna nalezená zařízení, na každé zařízení je možné kliknout a tím se k němu připojit. Ve spodní částí se nachází tlačítko, které aktivuje a deaktivuje skenování dostupných zařízení.

Pravá obrazovka se načte po kliknutí na zařízení. V horní části jsou vypsány informace o připojeném zařízení (jméno a adresa), hned pod těmito informacemi se nachází tlačítko pro odpojení od zařízení, které uživatele zároveň přesměruje zpět na první obrazovku. Většinu displeje zabírá graf, který slouží pro zobrazení příchozích dat (teploty) od vzdáleného zařízení připojeného pomocí BLE (ESP32). Měření charakteristik vzdáleného zařízení je aktivováno tlačítkem Measure. Ve spodní části okna se nachází editovací okno, do kterého je možno zadat data a pomocí tlačítka Send tato data poslat na vzdálené zařízení.



Obr. 5.6: Uživatelské rozhraní aplikace

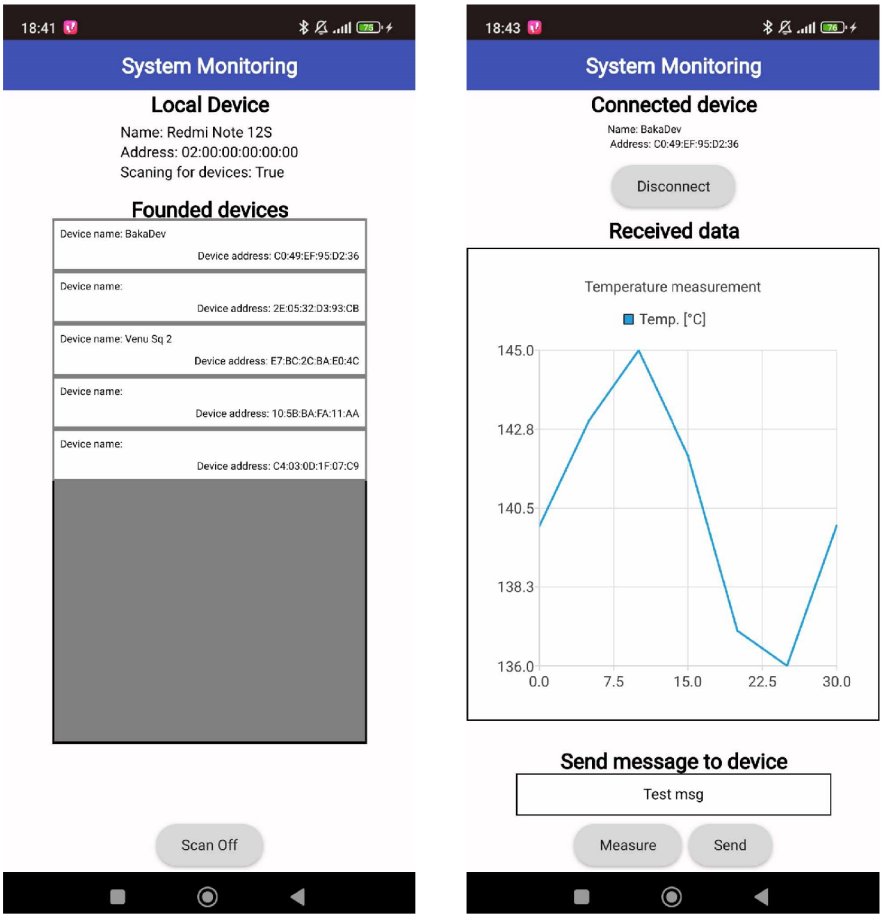
6 Demonstrace obousměrného přenosu dat

V této kapitole je demonstrována obousměrná komunikace mezi vyvinutou mobilní aplikací a zadaným přípravkem, na kterém se nachází modul pro bezdrátovou komunikaci ESP32.

Pro demonstraci funkčnosti přenosu dat byla zvolena situace, kdy modul ESP32 byl nakonfigurován do role SPP serveru, jak je popsáno v podkapitole 4.3.2, a mobilní aplikace, jejíž vývoj je popsán v kapitole 5, byla v roli klienta, který se k serveru připojuje pomocí BLE.

Pro účely této demonstrace byla konfigurace modulu ESP32 mírně upravena. Poté, co modul přejde do SPP módu, začne na 20 vteřin vysílat testovací data (teplotu), po odvysílání dat modul čeká na příchozí charakteristiky.

6.1 V rámci mobilní aplikace



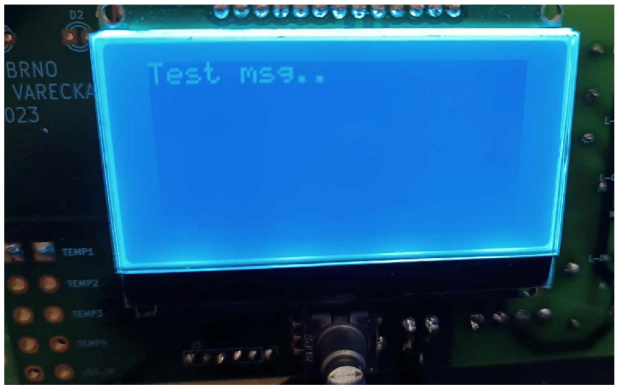
Obr. 6.1: Demonstrace obousměrného přenosu dat na straně aplikace

Na obrázku 6.1 je obrazovka aplikace s výsledky přenosu dat. Na obrazovce vlevo je zobrazen stav po stisknutí tlačítka Scan on, v kontejneru pro nalezená zařízení se nachází zařízení se jménem BakaDev a adresou C0:49:EF:95:D2:36, což je právě modul ESP32, který se nachází na zadaném přípravku. Po kliknutí na zařízení BakaDev je uživatel přesměrován na druhou obrazovku, tedy pravou obrazovku na obrázku 6.1, a mobilní zařízení se připojí k modulu.

Po stisknutí tlačítka Measure začne aplikace přijímat charakteristiky od zařízení BakaDev, pokud nějaké vysílá. V tomto případě zařízení vysílá 20 vteřin po připojení. Přijatá data jsou zaznamenávána do grafu. Pomocí tlačítka Send je možno odeslat data, která jsou zadána v editačním okně, zařízení BakaDev. V tomto případě byla posílaná data: "Test msg". Zpracování dat modulem ESP32 je popsáno níže.

6.2 V rámci zadaného přípravku

Modul ESP32 se nachází v SPP módu, jak je zmíněno výše, je tedy připraven přijímat a odesílat data po sériovém spojení. Po odvysílání testovacích dat čeká modul na příchozí zprávy. Jakmile uživatel zadá data do editovacího okna a stiskne tlačítko Send, data se z mobilního zařízení pošlou pomocí BLE modulu ESP32. Modul data po přijetí posílá pomocí sériového rozhraní UART mikrokontroléru STM32, který se na přípravku nachází (kapitola 4.1) a ten následně data vytiskne na LCD displej. Tato situace je zachycena na obrázku 6.2.



Obr. 6.2: Demonstrace přijmutí zprávy modulem ESP32

Závěr

V rámci této bakalářské práce bylo mým úkolem seznámit se s principy komunikace mobilních zařízení postavených na bezdrátové technologii Bluetooth. Následně pak nakonfigurovat bezdrátový modul ESP32 na zadaném hardwaru a naprogramovat mobilní aplikaci, která se k modulu připojí a obousměrně si s modulem bude vyměňovat data. Konečným výsledkem práce je demonstrace funkčnosti obousměrné výměny dat pomocí technologie Bluetooth low energy.

První kapitola je věnována popisu bezdrátové technologie Bluetooth, druhá kapitola, která je věnována technologii Bluetooth low energy, navazuje a rozšiřuje první kapitolu. Třetí kapitola se věnuje popisu Qt Frameworku a dalším softwarům, které byly využity pro design a programování mobilní aplikace.

Čtvrtá kapitola je věnována zadanému přípravku s bezdrátovým modulem ESP32. První podkapitola je věnována popisu zadaného přípravku. Zadaný přípravek musel projít jistými úpravami, které jsou popsány v kapitole 3.1.2. A to z důvodu špatného zapojení modulu na zadaném přípravku. V druhé podkapitole je popsán proces nahrávání AT firmwaru. Ten je nezbytný pro konfiguraci modulu za pomocí AT příkazů. AT firmware se po nutných úpravách přípravku podařilo do modulu ESP32 nahrát. Použitý firmware, sekvence pro nahrání firmwaru pomocí příkazové řádky a výpisy z konzoly jsou v příloze A. Třetí podkapitola je věnována testování AT firmwaru a konfiguraci modulu v prostředí STM32 Cube IDE za pomocí AT příkazů. Celý projekt pro konfiguraci modulu se nachází v příloze B.

Pátá kapitola je věnována návrhu,vývoji a popisu funkčnosti mobilní aplikace, která využívá Bluetooth low energy pro bezdrátovou, obousměrnou komunikaci s modulem ESP32. Aplikace byla navržena a naprogramována v software Qt Framework za použití jazyků QML a C++. Projekt aplikace naprogramované v Qt Creatoru se nachází v příloze C.

V šesté a poslední kapitole je demonstrována obousměrná komunikace mezi naprogramovanou mobilní aplikací a modulem ESP32 na zadaném přípravku. Modul po vytvoření připojení začíná posílat testovací data, která aplikace zaznamenává do grafu, avšak jen v případě akce uživatele, která proces přijímání dat spustí. Jakmile přenos dat ze strany modulu skončí, má uživatel možnost pomocí editačního okna poslat data modulu, ten data po přijetí zašle mikrokontroléru, který data vytiskne na LCD displej.

Naprogramovaná mobilní aplikace nabízí spoustu prostoru pro její vylepšení. Například při zapnutí aplikace neupozorní uživatele v případě, že na zařízení nemá aktivovaný Bluetooth. V rámci samotného kódu nejsou ošetřeny některé méně důležité výjimky, které můžou průběh bezdrátové komunikace narušit. Aplikace taky dokáže správně komunikovat jen se zadaným hardwarem.

Literatura

[1] *Bluetooth.* Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2005. Dostupné z: https://cs.wikipedia.org/wiki/Bluetooth. [cit. 2023-12-29].

[2] KOVAŘÍK, David. *Bluetooth – modrozub pod drobnohledem (vědecké okénko).* Online. In: mobilizujeme.cz. 2011. Dostupné z: https://mobilizujeme.cz/clanky/bluetooth-modrozub-pod-drobnohledem-vedecke-okenko. [cit. 2023-12-29].

[3] *Bluetooth.* Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2004. Dostupné z: https://en.wikipedia.org/wiki/Bluetooth. [cit. 2023-12-29].

[4] *Bluetooth | About us.* Online. Bluetooth.com. 1998. Dostupné z: https://www.bluetooth.com/about-us/. [cit. 2023-12-29].

[5] *FHSS.* Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2006. Dostupné z: https://cs.wikipedia.org/wiki/FHSS. [cit. 2023-12-29].

[6] *ISM pásmo.* Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2022. Dostupné z: https://cs.wikipedia.org/wiki/ISM_pÃąsmo. [cit. 2023-12-29].

[7] POOLE, Ian. *How Does Bluetooth Work.* Online. In: Electronics notes. 2022. Dostupné z: https://www.electronics-notes.com/articles/connectivity/bluetooth/how-bluetooth-works.php. [cit. 2023-12-29].

[8] *List of Bluetooth protocols.* Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2014. Dostupné z: https://en.wikipedia.org/wiki/List_of_Bluetooth_protocols. [cit. 2023-12-29].

[9] MOUMITA. *The Bluetooth Protocol Stack.* Online. In: Tutorialspoint. 2021. Dostupné z: https://www.tutorialspoint.com/the-bluetooth-protocol-stack. [cit. 2023-12-30].

[10] MOUMITA. *The Bluetooth Protocol Architecture.* Online. In: Tutorialspoint. 2021. Dostupné z:

https://www.tutorialspoint.com/the-bluetooth-protocol-architecture. [cit. 2023-12-30].

[11] *What is Bluetooth Host Controller Interface (HCI).* Online. In: Feasycom.com. 2023. Dostupné z: https://www.feasycom.com/bluetooth-host-controller-interface-hci.html. [cit. 2023-12-30].

[12] *AT Command Reference.* Online. In: LM Technologies. 2016. Dostupné z: https://wiki.lm-technologies.com/at-command-reference/. [cit. 2023-12-30].

[13] *List of Bluetooth profiles.* Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2014. Dostupné z: https://en.wikipedia.org/wiki/List_of_Bluetooth_profiles. [cit. 2023-12-30].

[14] *Bluetooth Wireless Technology Profiles and Descriptions.* Online. In: Sony. 2020. Dostupné z: https://www.sony.com/electronics/support/articles/00007501. [cit. 2023-12-30].

[15] *NamingSystem: Bluetooth Address as a device identifier.* Online. In: HL7 Terminology. 2020. Dostupné z: https://terminology.hl7.org/3.1.0/NamingSystem-bluetooth-address-identifier.html. [cit. 2023-12-30].

[16] ZAIDI, Syed Ali Jafar. *Basic Study of Bluetooth Networking Topologies.* Online. In: ResearchGate. 2016. Dostupné z: https://www.researchgate.net/publication/336020195_Basic_Study_of_Bluetooth_Networking_Topologies. [cit. 2023-12-30].

[17] BLUETOOTH BEACON. *Understanding Bluetooth Pairing — A Guide to Its Significance and Process.* Online. In: Medium. 2023. Dostupné z: https://medium.com/@iamtecksay/understanding-bluetooth-pairing-a-guide-to-its-sign [cit. 2023-12-30].

[18] VAŘEČKA, Ondřej. *Řídicí systém peletového grillu.* Bakalářská práce. Brno: Vysoké učení technické, Fakulta elektrotechniky a komunikačních technologií, 2023. [cit. 2024-01-03].

[19] *Datasheet ESP32-WROOM32-E, ESP32-WROOM32-UE.* Online. Espressif. 2020. Dostupné z:

https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32e_esp32-wroom-32ue_datasheet_en.pdf. [cit. 2024-01-03].

[20] *Datasheet ESP32-WROOM32-D, ESP32WROOM32-U* Online. Espressif. 2019. Dostupné z: https://www.espressif.com/sites/default/files/documentation/esp32-wroom-32d_esp32-wroom-32u_datasheet_en.pdf. [cit. 2024-01-03].

[21] *ESP32 ESP-AT User Guide.* Online. Espressif. 2022. Dostupné z: https://docs.espressif.com/projects/esp-at/en/latest/esp32/esp-at-en-v2.3.0.0_esp32c3-698-gc3ef017bb2-esp32.pdf. [cit. 2024-01-03].

[22] *Datasheet STM32F030x4, STM32F030x6, STM32F030x8, STM32F030xC.* Online. Mouser. 2019. Dostupné z: https://cz.mouser.com/datasheet/2/389/stm32f030f4-1851168.pdf. [cit. 2024-01-03].

[23] *Integrated Development Environment for STM32, Product overview.* Online. St.com. 2019. Dostupné z: https://www.st.com/en/development-tools/stm32cubeide.html. [cit. 2024-01-03].

[24] *ESP-32 Dev Kit C V2 EN.pdf.* Online. Edu.xunta.gal. 2020. Dostupné z: http://www.edu.xunta.gal/centros/ieslaxeiro/system/files/ESP-32%20Dev%20Kit%20C%20V2_EN.pdf. [cit. 2024-01-03].

[25] *Released Firmware - ESP32.* Online. In: Espressif. 2022. Dostupné z: https://docs.espressif.com/projects/esp-at/en/latest/esp32/AT_Binary_Lists/ESP32_AT_binaries.html. [cit. 2024-01-03].

[26] *Esptool.py Documentation.* Online. In: Espressif. 2022. Dostupné z: https://docs.espressif.com/projects/esptool/en/latest/esp32/. [cit. 2024-01-03].

[27] PROCTOR, Bob. *Bluetooth Vs. Bluetooth Low Energy: What's The Difference? [2023 Update].* Online. In: Link Labs. 2023. Dostupné z: https://www.link-labs.com/blog/bluetooth-vs-bluetooth-low-energy. [cit. 2024-05-18].

[28] *Bluetooth Technology Overview.* Online. In: Bluetooth. 2021. Dostupné z: https://www.bluetooth.com/learn-about-bluetooth/tech-overview/. [cit. 2024-05-18].

[29] *LEARN ABOUT BLUETOOTH Topology Options.* Online. In: Bluetooth. 2021. Dostupné z: https://www.bluetooth.com/learn-about-bluetooth/topology-options/. [cit. 2024-05-18].

[30] *Bluetooth Low Energy není jen nová verze standardu Bluetooth.* Online. *Automa.* 2013, roč. 2013, č. 12, s. 4. Dostupné z: https://www.automa.cz/cz/casopis-clanky/bluetooth-low-energy-neni-jen-nova-verze-s 12_0_10907/. [cit. 2024-05-18].

[31] *Introduction to Bluetooth Low Energy: GATT.* Online. In: Adafruit. 2014, 8.3.2024. Dostupné z: https://learn.adafruit.com/introduction-to-bluetooth-low-energy/gatt. [cit. 2024-05-18].

[32] *Developing a Bluetooth Low Energy Application: Generic Attribute Profile (GATT).* Online. In: Software-dl.ti. 2016. Dostupné z: https://software-dl.ti.com/lprf/sdg-latest/html/ble-stack-3.x/gatt.html. [cit. 2024-05-18].

[33] KUBARA, Ivan. *What Is Qt framework, Why to Use It, and How?* Online. In: Lemberg Solutions. 2023. Dostupné z: https://lembergsolutions.com/blog/why-use-qt-framework. [cit. 2024-05-18].

[34] *Qt Creator Manual: Developing with Qt Creator.* Online. In: Qt Documentation. 2014, 2024. Dostupné z: https://doc.qt.io/qtcreator/creator-overview.html. [cit. 2024-05-18].

[35] *Qt 6.7: Qt Quick.* Online. In: Qt Documentation. 2021, 2024. Dostupné z: https://doc.qt.io/qt-6/qtquick-index.html. [cit. 2024-05-18].

[36] *Qt 6.7: Qt QML: The QML Reference.* Online. In: Qt Documentation. 2021, 2024. Dostupné z: https://doc.qt.io/qt-6/qmlreference.html. [cit. 2024-05-18].

[37] *Qt 6.7: Qt QML: QML Syntax Basics.* Online. In: Qt Documentation. 2021, 2024. Dostupné z: https://doc.qt.io/qt-6/qtqml-syntax-basics.html. [cit. 2024-05-18].

[38] *Qt Quick Controls: QML Types: Button QML Type.* Online. In: Qt Documentation. 2016, 2024. Dostupné z:

https://doc.qt.io/qt-5/qml-qtquick-controls2-button.html. [cit. 2024-05-18].

[39] QT GROUP. *Qt Documentation.* Online. 2014, 2024. Dostupné z: https://doc.qt.io. [cit. 2024-05-19].

[40] QT GROUP. *Qt Forum.* Online. 2015, 2024. Dostupné z: https://forum.qt.io. [cit. 2024-05-19].

[41] QT GROUP. *Qt 6.7: Qt Examples And Tutorials.* Online. Qt Documentation. 2021, 2024. Dostupné z: https://doc.qt.io/qt-6/qtexamplesandtutorials.html. [cit. 2024-05-19].

Seznam symbolů a zkratek

API	Application Programming Interface
BLE	Bluetooth low energy
BT	Bluetooth
BSIG	Bluetooth Special Interest Group
FHSS	Frequency Hopping Spread Spectrum
GATT	Generic ATTribute Profile
IEEE	Institute of Electrical and Electronics Engineers
ISM	Industrial, Scientific and Medical
LCD	Liquid Crystal Display
MAC	Media Access Control
PWM	Pulse Width Modulation
QML	Qt Modeling Language
SPP	Serial Port Profile
SPI	Serial Peripheral Interface
PAN	Personal Area Network
UUID	Universaly Unique Identifier
USB	Universal Serial Bus
UART	Universal Asynchronous Receiver-Transmitter
USART	Universal Synchronous / Asynchronous Receiver-Transmitter
WiFi	Wireless Fidelity

Seznam příloh

A Firmware, flashovací sekvence a výpisy na konzoli

Všechny tyto soubory jsou přiloženy ve složce PrilohaA ve společném .zip souboru, ve kterém byla tato bakalářská práce odevzdána.

B STM32 Cube IDE projekt pro konfiguraci modulu ESP32

Všechny tyto soubory jsou přiloženy ve složce PrilohaB ve společném .zip souboru, ve kterém byla tato bakalářská práce odevzdána.

C Qt Creator projekt mobilní aplikace pro bezdrátový přenos dat

Všechny tyto soubory jsou přiloženy ve složce PrilohaC ve společném .zip souboru, ve kterém byla tato bakalářská práce odevzdána.