Univerzita Hradec Králové
Fakulta informatiky a managementu
Katedra informačních technologií

Implementace bezdrátových modulů do systému domácí automatizace

Bakalářská práce

Autor: Adam Zmeškal
Studijní obor: Aplikovaná informatika

Vedoucí práce: Ing. Karel Mls, Ph.D.

Odborný konzultant: Ing. Jan Štěpán, Ph.D.

Hradec Králové Srpen 2022

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a s použitím uvedené literatury.

vlastnoruční podpis

V Hradci Králové dne 15.8.2022

Poděkování:

Děkuji vedoucímu bakalářské práce Ing. Karlu Mlsovi, Ph.D. a také odbornému konzultantovi Ing. Janu Štěpánovi, Ph.D. za metodické vedení práce, trpělivost a ochotu, kterou mi v průběhu zpracování věnovali.

Anotace

Tato bakalářská práce se zabývá implementací bezdrátových modulů ESP32 do systémů domácí automatizace. Součástí práce je rozebrání technologií a technických specifikací hardwarových prvků, které jsou v práci použity a také osvětlení, jak funguje komunikace mezi těmito prvky. Bližší pohled na open-source vývojový nástroj Node-red a službu MQTT, pomocí kterých je vytvořen systém internetů věcí. Hlavní část práce se věnuje zapojení senzorů a jejich praktického využití, implementaci a popisu jednotlivých funkcí kódu, který běží na ESP32 modulech. Součástí je instalace nástroje Node-red a síťového protokolu MQTT na Raspberry Pi doplněná o popis flow(toků) v nástroji Node-Red a ukázku a popis uživatelského rozhraní.

Klíčová slova: ESP32, Node-RED, Wi-Fi, MQTT, Raspberry Pi, Monitoring, Automatizace

Annotation

Implementation of wireless modules into a home automation system

This bachelor's thesis deals with the implementation of ESP32 wireless modules in home automation systems. Part of the work is a breakdown of the technologies and technical specifications of the hardware elements that are used in the work, as well as an explanation of how communication between these elements works. A closer look at the open-source development tool Node-red and the MQTT service, with which the Internet of Things system is created.

The main part of the work is devoted to the example of the wiring diagram of the sensors and their practical use, the implementation and description of the individual functions of the code that runs on ESP32 modules. The installation of the Node-red tool and the MQTT network protocol on the Raspberry Pi is included, along with a description of the flows in the Node-Red tool and a demonstration and description of the user interface.

Keywords: ESP32, Node-RED, Wi-Fi, MQTT, Raspberry Pi, Monitoring, Automation

Obsah

Seznam obrázků

Seznam tabulek

Seznam Zkratek

VoIP - Voice over Internet Protocol

DNS - Domain Name System

IANA - Internet Assigned Numbers Authority

ICANN - Internet Corporation for Assigned Names and Numbers

MQTT - Message Queuning Telemetry Transport

UI - User Interface

GPIO - General Purpose Input/Output

BLE - Bluetooth Low Energy

ARM - Advenced RISC Machine

GND - Ground

I/O - Input/Output

IoT - Internet of Things

IDE - Integrated Development Enviroment

SSH - Secure Shell

ADC - Analog to Digital Converter

DAC - Digital to Analog Converter

PWM - Pulse-width modulation

UART - Universal Asynchronous Receiver-Transmitter

I2C - Inter-Integrated Circuit

SCL - Serial Clock

SDA - Serial DAta line

WPA - Wi-Fi Protected Access

HTTPS - Hypertext Transfer Protocol Secure

HTTP - Hypertext Transfer Protocol

SMTP - Simple Mail Transfer Protocol

POP - Post Office Protocol

FTP - File Transport Protocol

TCP/IP - Transmission Control Protocol/Internet Protocol

SoC - Systém on a Chip

UDP - User Datagram Protocol

1 Úvod

Domácí automatizace je téma, které existuje už mnoho let. Základem tohoto tématu je ovládání a automatizování prvků našeho domova, tak abychom do nich nemuseli zasahovat a v případě nutného zásahu je mohli ovládat vzdáleně přes internet. Domácí automatizace se rozmohla v posledních letech takzvaně „mílovými kroky“, jelikož zařízení jako jsou malé počítače či mikrokontrolery s komunikací za pomocí Wi-Fi nebo Bluetooth, jsou snadno dostupné.

Tyto systémy mají tři základní prvky, a to senzory, které zaznamenávají požadovanou veličinu – například teplotu, dále akční prvky, které řídí mechanismy – mohou to být například spínače světel. Třetím prvkem jsou kontroléry – tím může být například osobní počítač nebo mobilní telefon, na kterém lze monitorovat teplotu uvnitř budovy, nebo zkontrolovat vypnutí světel v místnostech nebo tuto akci provést. Mezi systémy pro domácí automatizace patří například termostat, ovládání světel, bezpečností systémy, jako kamery, pohybové senzory a alarmy, které nám odešlou zprávu do mobilního telefonu, e-mailu, webové aplikace, nebo pošle příkaz do jiného zařízení, které například při přijetí zprávy ze senzoru pohybu otočí kameru na příslušné místo.

2 Cíl práce

Cílem této práce je navrhnout a vytvořit jednoduchý systém domácí automatizace a také představit nástroje k jeho použití. Systém použije bezdrátové vývojové desky ESP32 pro připojení k routeru v místní síti pomocí Wi-Fi. Tyto vývojové desky mohou zpracovávat různá data v závislosti na typu připojeného senzoru, nebo ovládat komponenty k nim připojené. Vývojové desky ESP32 budou používat síťový protokol MQTT ke zveřejnění a odebíraní těchto dat. Jako server pro služby MQTT a Node-RED bude využit malý počítač Raspberry Pi. Node-Red bude využit na flow programování a zpracování dat, které následně vytvoří monitorovací a ovládací systém, který bude přístupný z lokální sítě Wi-Fi.

3 Teoretická část

3.1 ESP32

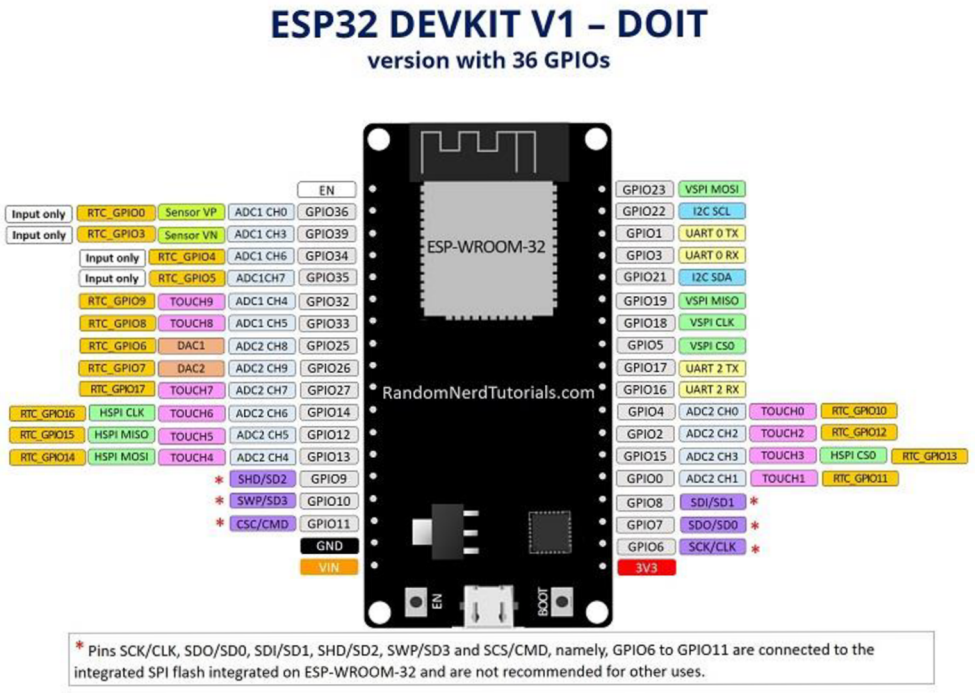
ESP32 je název mikrokontroleru, který byl navržen společností Espressif Systems. Espressif je společnost se sídlem v Šanghaji. ESP32 je samostatné Wi-Fi síťové řešení, které je použito jako most stávajících mikrokontrolerů k Wi-Fi, také je schopný spouštět samostatné aplikace. Sériová výroba ESP32 započala na konci roku 2016. [1]

Tyto ESP32 moduly nejsou přímo připraveny pro běžné použití, proto je využívají výrobci třetích stran, kteří navrhují základní obvody, navrhují desky plošných spojů a konstruují předem připravené desky. Tyto desky se již dají použít, klasickým připojením na GPIO piny. Cena těchto desek se pohybuje od stovek do tisíců korun, záleží na velikosti desky, počtů GPIO pinů nebo další možnou výbavou desky.

Obrázek 1. ESP-WROOM-32[1]

3.1.1 Pinout vývojové desky ESP-WROOM-32

ESP-WROOM-32 je modul používaný na vývojových deskách různých výrobců, pinout na různých vývojových deskách se může lišit, ale ve většině případů bude schéma zapojení odpovídat obrázku č. 2. Pro nahrávání zdrojového kódu modul využívá sériovou komunikaci UART.



Obrázek 2. Pinout schéma ESP-WROOM-32 modulu [2]

Na vývojové desce jsou GPIO (General Purpose Input/Output) neboli univerzální vstup/výstup [2]:

- **Input only** (Pouze pro čtení)

GPIO 34 až 39 jsou GPI – pouze vstupní piny. Tyto piny nemají vnitřní pull-up nebo pull-down rezistory.[2]

- **SPI flash integrated**

GPIO 6 až GPIO 11 jsou vystaveny na některých vývojových deskách ESP32. Tyto piny jsou však připojeny k integrovanému flash SPI na čipu ESP-WROOM-32 a pro jiné použití se nedoporučují.

- **Touch** (dotykové)

ESP32 má 10 interních kapacitních dotykových senzorů. Mohou vnímat změny v jakémkoli nabitém předmětu, jako je lidská kůže. Proto mohou detekovat podmínky způsobené dotykem prstu na GPIO.

- **Analog to Digital Converter** (Analogově digitální převodník)

ESP32 má 18 x 12bitové vstupní kanály ADC, které se používají pro vstup analogových signálů a jejich převod na digitální signály.

- **Digital to Analog Converter** (Digitálně analogový převodník)

ESP32 má 2 x 8bitové DAC kanály pro převod digitálních signálů na analogový napěťový výstup.

- **PWM** (Pulzně šířková modulace)

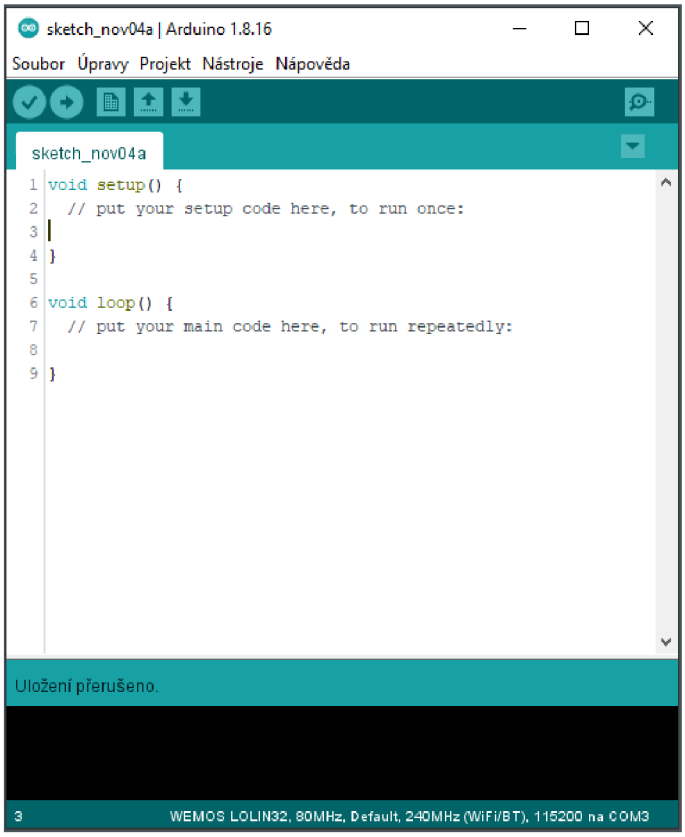
Ovladač ESP32 LED PWM má 16 nezávislých kanálů, které lze nakonfigurovat tak, aby generovaly signály PWM s různými charakteristikami. Všechny piny, které lze použít jako výstupy, lze použít jako piny PWM. PWM je používán pro ovládání napájení elektronického zařízení, má dva stavy 1 nebo 0, proud dodaný do zařízení je řízen dobou trvání jeho stavu.

- **I2C sběrnice**

ESP32 má dva I2C kanály a jakýkoli pin lze nastavit jako SDA (serial data) nebo SCL (seriál clock). I2C sběrnice celým názvem (Inter-Integrated Circuit), obousměrný přenos probíhá pouze na dvou vodičích SDA a SCL. Funguje na principu MASTER/SLAVE neboli většinou mikrokontroler je nastaven jako MASTER a ostatní obvody na SLAVE. Ve chvíli, kdy čip vysílá, ostatní přijímají a podle adresy rozlišují, jestli data jsou pro ně.

3.1.2 Vývojové prostředí Arduino IDE

Vývojové prostředí Arduino IDE je multiplatformní aplikace fungující na operačních systémech Windows, macOS a Linux. Aplikace podporuje programovací jazyky C a C++. Aplikace je nativně kompatibilní s deskou Arduino. Pro přidání desky třetí strany je potřeba v nastavení přidat URL s příslušnými daty desky, nebo ji stáhnout a ručně vložit do adresáře, kam byla aplikace nainstalována. Aplikaci lze snadno rozšířit pomocí uživatelsky vytvořených knihoven. Kód, který se nahraje na desku, je založen na funkcích setup() a loop(), viz obrázek 3. Všechny projekty - takzvané skici – jsou v aplikaci uloženy jako soubory s příponou .ino, např. „sketch.ino“.



Obrázek 3. Vývojové prostředí Arduino IDE [autor]

- **Funkce setup()**

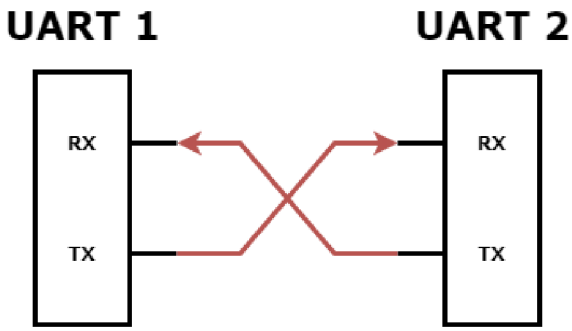
Volá se, při startu zařízení. Použita je k inicializaci proměnných, připnutí vzorů, zahájení používání knihoven atd. Funkce setup() se spustí pouze jednou, při každém zapnutí nebo resetu desky. [3]

- **Funkce loop()**

Po vytvoření funkce setup() pro inicializaci a nastavení počáteční hodnoty, funkce loop() opakuje kód , který je v ní vepsán, jak nám také napovídá anglický název, což umožňuje programu reagovat na změny v zařízení. Používá se pro aktivní ovládání desky.[4]

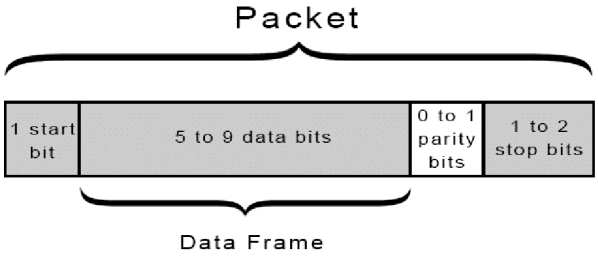
3.1.3 Sériová komunikace UART

UART (Universal asynchronous receiver-transmitter) je asynchronní sériovou komunikaci mezi dvěma UART porty přímo mezi sebou, kterou používáme například v komunikaci mezi PC a ESP32. Vysílající UART převádí paralelní data z řídícího zařízení, jako je CPU, do sériové podoby a tato data odesílá do přijímacího UART, který tato data převádí zpět do paralelní podoby a posílá do řídícího zařízení. Data se vysílají z pinu TX do příjímacího pinu RX.[5]



Obrázek 4. Schéma Odesílaní TX-RX [autor]

V UART komunikaci chybí hodinový signál, a proto přidává vysílající UART do datového paketu start bit a stop bit. Tyto bity nám dávají informaci, kde je začátek a konec datového paketu a přijímající UART ví, kdy má začít číst bity. Když příjímací UART detekuje start bit, začne číst na specifické přenosové rychlosti která se udává v bitech za sekundu nebo také Baud rate, tato přenosová rychlost musí být na vysílajícím i příjímacím zařízení stejná. [5]



Obrázek 5. UART packet [5]

- **START BIT**

Datová linka UART obvykle zůstává na vysoké úrovni, když nepřenáší data. Za účelem zahájení přenosu dat přepne odesílající přenosovou linku UART z vysoké úrovně na nízké úrovni po dobu jednoho hodinového cyklu. Když přijímací UART detekuje přechod z vysokého úrovně na nízkou úroveň, začne číst bity v datovém rámci přenosovou rychlostí. [5]

- **DATA FRAME**

Datový rámec obsahuje aktuální přenášená data. Pokud je použit paritní bit, může být dlouhý 5 až 8 bitů. Pokud není použit žádný paritní bit, může být datový rámec dlouhý 9 bitů.[5]

- **PARITY**

Parita popisuje sudost nebo lichost čísla. Paritní bit je způsob, jak přijímající UART zjistí, zda se nějaká data během přenosu změnila. Poté, co přijímající UART načte datový rámec, spočítá počet bitů s hodnotou 1 a zkontroluje, zda je součet sudé nebo liché číslo. Pokud je paritní bit 0 (sudá parita), jedničkové bity v datovém rámci by měly dát součet sudé číslo. Pokud je paritní bit 1 (lichá parita), jedničkové bity v datovém rámci by měly dát součet liché číslo. Když paritní bit odpovídá datům, UART ví, že přenos byl bez chyb. Ale pokud je paritní bit 0 a součet je lichý; nebo paritní bit je 1 a součet je sudý, UART ví, že se bity v datovém rámci změnily. [5]

- **STOP BIT**

Pro signalizaci konce datového paketu převede odesílající UART linku přenosu dat z nízké úrovně na vysokou úroveň po dobu alespoň dvou bitů.[5]

3.2 Raspberry Pi

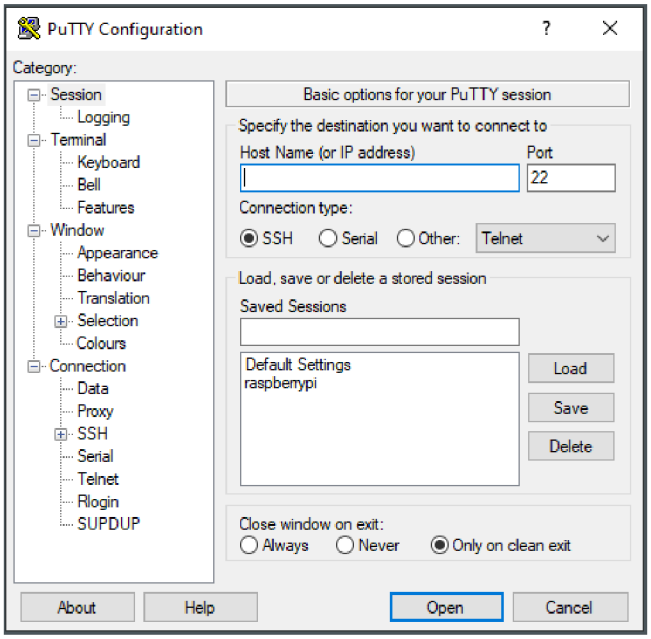
Raspberry Pi je levný jednodeskový počítač s velmi malými rozměry. Obsahuje stejná výstupní zařízení jako výstup stolního počítače používaný pro displej (HDMI) a nechybí ani USB pro připojení myši nebo klávesnice. Společnost vyvinula několik generací takových počítačů, které se liší velikostí, výkonem a účelem. Mikroprocesor na základní desce tohoto počítače má architekturu ARM a používá se kvůli nízké spotřebě energie. Oficiálním systémem Raspberry Pi je Raspberry Pi OS, dříve známý jako Raspbian, ale na tomto zařízení lze provozovat i různé linuxové distribuce.

Jak tvrdí výrobce, Raspberry Pi je počítač, který lidem všech věkových kategorií dovolí prozkoumávat svět výpočetní techniky a naučit se v programovacích jazycích. Také je to zařízení, které je schopno dělat vše co bychom očekávali od stolního počítače.[6]

Na rozdíl od platforem ESP32 a Arduino, lze Raspberry Pi použít k vytvoření firmwaru, nejen k ovládání zařízení, také lze použít piny GPIO na jednodeskovém počítači.

3.2.1 PuTTY

PuTTY je open-source aplikace pod licencí MIT, která využívá síťové protokoly jako SSH, Telnet, rlogin. Tato aplikace umožnuje vzdálenou textovou komunikaci mezi počítači. Aplikace je dostupná pro Windows a různé Unixové platformy. [7]



Obrázek 6. Prostředí PuTTY [autor]

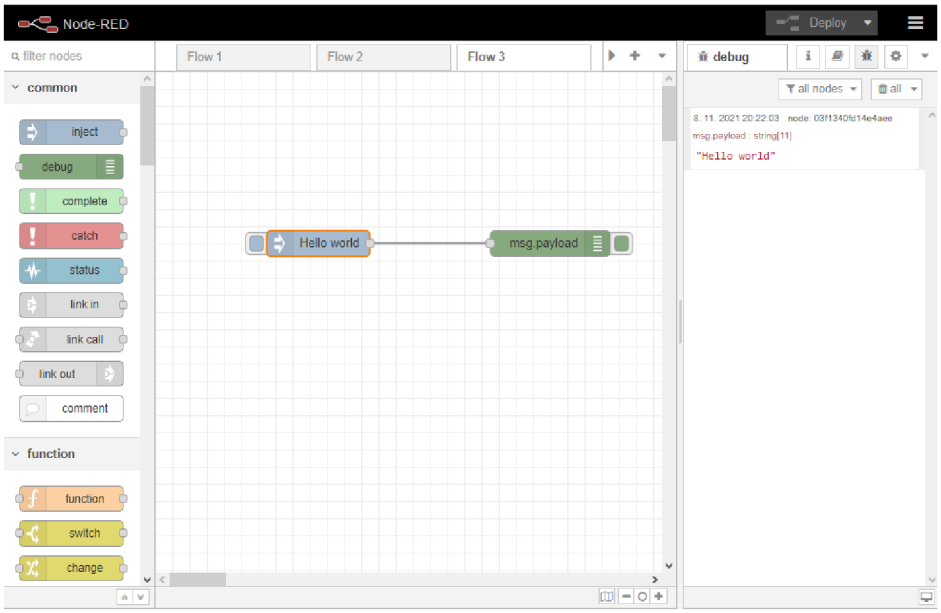
- SSH

SSH (také známý jako Secure Shell) je metoda pro vzdálené a bezpečné přihlášení ke vzdálenému počítači z místního počítače. Poskytuje řadu alternativ k silnému ověření identity a chrání bezpečnost a integritu komunikace prostřednictvím silného šifrování. Jedná se o bezpečnou alternativu k nechráněným přihlašovacím protokolům (jako je telnet, rlogin) a nezabezpečeným metodám přenosu souborů (jako je FTP). [8;9]

3.2.2 Node-RED

Node-RED poskytuje flow editor neboli editor toků založený na prohlížeči, který může snadno propojit toky pomocí různých uzlů z nabídky. Tyto toky lze poté nasadit v běhovém prostředí. K vytvoření funkcí JavaScriptu v editoru lze použít editor formátovaného textu. Vestavěná knihovna umožňuje ukládat užitečné funkce,

šablony nebo streamy pro opětovné použití. Lehký runtime je postaven na Node.js a plně využívá jeho událostmi řízený neblokovací model. Díky tomu je ideální pro provoz na okraji sítě na levném hardwaru (jako je Raspberry Pi) a cloudu. S více než 225 000 modulů v balíčku Node-RED je snadné nabídku uzlů rozšířit a přidat nové funkce. [10]



Obrázek 7. Vývojové prostředí Node-Red [autor]

3.3 Technologie pro komunikaci s vývojovými deskami

3.3.1 Wi-Fi

Wi-Fi, jak se pro spotřebitele označuje, je sada standardů IEEE 802.11, které definují komunikaci pro bezdrátové sítě WLAN. Tyto standardy jsou vytvářeny organizací IEEE[11], která specifikuje vlastnosti daného standardu - jedná se o frekvenční pásma, ve kterých funguje a maximální přenosovou rychlost, jak je uvedeno v tabulce č.1.

Tabulka 1. Wi-Fi generace a standarty [11]

Standard	Přijato	Frekvence	Rychlost
IEEE 802.11	1997	2.4 GHz	1 to 2 Mbit/s
Wi-Fi 1/IEEE 802.11a	1999	5 GHz	6 to 54 Mbit/s

Wi-Fi 2/IEEE 802.11b	1999	2.4 GHz	1 to 11 Mbit/s
Wi-Fi 3/IEEE 802.11g	2003	2.4 GHz	6 to 54 Mbit/s
Wi-Fi 4/IEEE 802.11n	2008	2.4/5 GHz	72 to 600 Mbit/s
Wi-Fi 5/IEEE 802.11ac	2014	5 GHz	433 to 6933 Mbit/s
Wi-Fi 6/IEEE 802.11ax	2021	2.4/5 GHz	600 to 9608 Mbit/s

V klasické lokální Wi-Fi síti se síťový prvek nazývá přístupový bod a tento prvek je nejčastěji zastoupen wi-fi routerem. Existuje mnoho typů routerů, ale nejdůležitější jsou komunikační a signálové standardy. Přístupový bod má svůj SSID, což je název sítě. Síť lze chránit jako WPA, což je termín pro zabezpečení sítě, a pak jsou klienti, kteří se chtějí k síti připojit, jako jsou notebooky, telefony, televize atd. Klientská zařízení však mají také své standardy IEEE a přístupové body pro komunikaci a tyto standardy jsou zpětně kompatibilní. Pokud je k dispozici směrovač (802.11.ac) a klientské zařízení (802.11n), použije směrovač ke komunikaci s klientem standard (802.11n).

3.3.2 MQTT

MQTT je lehký síťový protokol, který slouží pro odesílání a přijímání zpráv pro zařízení s úzkou šířkou pásma. Protokol obvykle funguje skrze TCP/IP ale může fungovat i na jiných, pokud protokol je uspořádaný, bezztrátový a má obousměrné připojení. Využívá se ve spoustě odvětví, jako například průmyslové výrobě, chytrých domácnostech atp. [12;13]

Základní elementy MQTT [12]:

- **Broker** (zprostředkovatel)
  Zprostředkovatel je odpovědný za přijímání zpráv a jejich filtrování a také za zveřejnění zprávy, když o ní někdo požádá. [12]
- **Publish** (publikovat)
  Jak už nám název naznačuje, když publikujeme, tak odesíláme příslušná data našemu MQTT zprostředkovateli. Publikovat můžeme například data ze senzorů atd.

- **Subscribe** (odebírat)

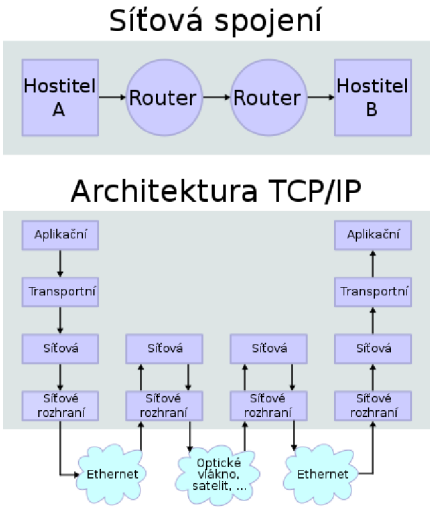
  Naopak odběratel čeká na zprávu, kterou nám MQTT zprostředkovatel zveřejní, pokud o ní požádáme, na což může tento odběratel vykonat nějakou akci po přijmutí této zprávy.
- **Topic** (téma)

  Téma má za úkol systematicky klasifikovat, o jaká data se jedná, tato témata se oddělují lomítkem, například (domov/loznice/svetlo1). Témata jsou case sensitive neboli je potřeba si dát pozor na velká a malá písmena.
- **Message** (zpráva)

  Zpráva jsou už pouze data, která chceme mezi zařízeními vyměnit.

3.3.3 TCP/IP

Kvůli složitosti síťové komunikace je TCP/IP protokol rozdělen na tzv. vrstvy, které mají přesně definovanou komunikaci informací mezi nimi. Komunikace mezi vrstvami funguje v hierarchii, neboli každá vrstva, která je výš a využívá služeb nižší vrstvy a každá nižší vrstva poskytuje služby vyšší vrstvě. Komunikace stejných vrstev ve dvou rozdílných systémech je navázaná pomocí spojení vytvořené sousední nižší vrstvou. TCP/IP je vytvořeno pomocí 4 vrstev.[14]

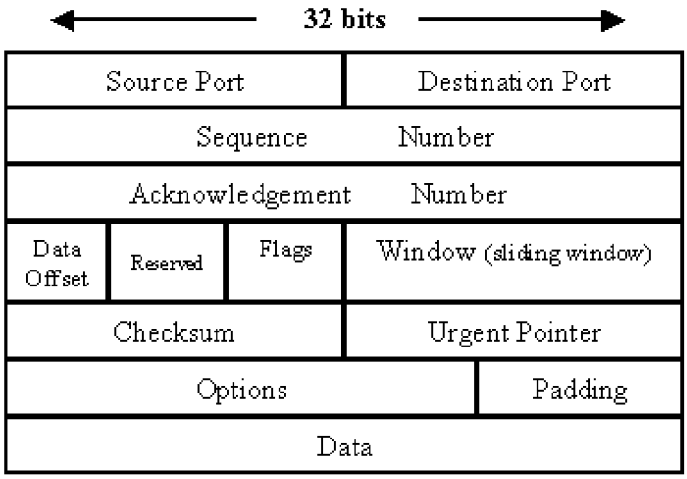


Obrázek 8. TCP/IP vrstvy [15]

- **Aplikační vrstva** (application layer)
  Poskytne protokoly pro přenos specifických dat, jako je Telnet, FTP, HTTP, DHCP, DNS. Protokol vždy využívá jednu ze dvou základních služeb transportní vrstvy: TCP nebo UDP. Z důvodu rozlišení protokolu je použit port, což je dohodnuté číslo protokolu. [14]
- **Transportní vrstva** (transport layer)
  Vrstva definuje úroveň služby a stav připojení používaného při přenosu dat. Hlavní protokoly zahrnuté v transportní vrstvě jsou TCP (Transmission Control Protocol) a UDP (User Datagram Protocol) se vyznačuje nespolehlivým spojením pro služby, které nepotřebují kontrolu. [14]
- **Síťová vrstva** (internet layer)
  Je vrstvou, která zajišťuje veškeré adresování a směrování v síti.[14]
- **Vrstva síťového rozhraní** (network interface layer)
  Je nejnižší vrstvou TCP/IP a stará se o přístup ke všem fyzickým součástem síťové konektivity a definuje, jak jsou data fyzicky odesílána. [14]

3.3.3.1 TCP

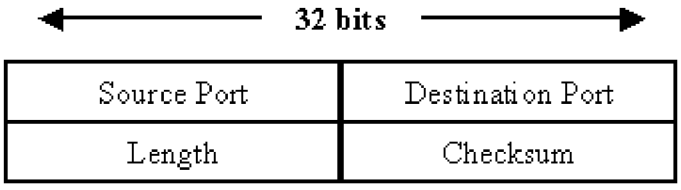
TCP (Transmission Control Protocol) je zkratka pro protokol řízení přenosu, což je typ metody datové komunikace orientované na spojení v transportní vrstvě TCP/IP. Byl navržen tak, aby pomohl vytvořit spolehlivé obousměrné připojení mezi zařízeními. TCP kontroluje chyby a zaručuje odeslání dat v pořadí, v jakém byla odeslána, ale také je náročnější na naši síť. Užívá se například pro HTTPS, HTTP, SMTP, POP, FTP.[16;17]



Obrázek 9. Struktura TCP packet [16]

3.3.3.2 UDP

UDP (User Datagram Protocol) odkazuje na User Datagram Protocol. Přestože se jedná o komunikační protokol transportní vrstvy jako TCP, rozdíl je v tom, že se jedná o komunikační protokol bez připojení, což znamená, že před odesláním datagramu není nutné navazovat spojení. Používá se speciálně pro sítě, kde může dojít ke ztrátě dat. Používá se například pro videokonference, VoIP, DNS.[16;18]



Obrázek 10. Struktura UDP packet [16]

3.3.3.3 Síťový port

Síťový port je číslo, které je standardizováno na všech zařízeních připojených k síti. Porty jsou rozděleny do 3 skupin organizací IANA, která toto rozdělení vypracovala a jsou dostupná na jejich webových stránkách, v dnešní době již IANA je řízena organizací ICANN. IP adresy umožňují směrování do a z konkrétního zařízení. Čísla portů, která jsou uvedena za IP adresou oddělené dvojtečkou směřují na konkrétní

službu nebo aplikaci v rámci tohoto zařízení [19]. Síťový port můžeme nalézt v hlavičce paketu, jak můžeme vidět na obrázku č.9. a 10.

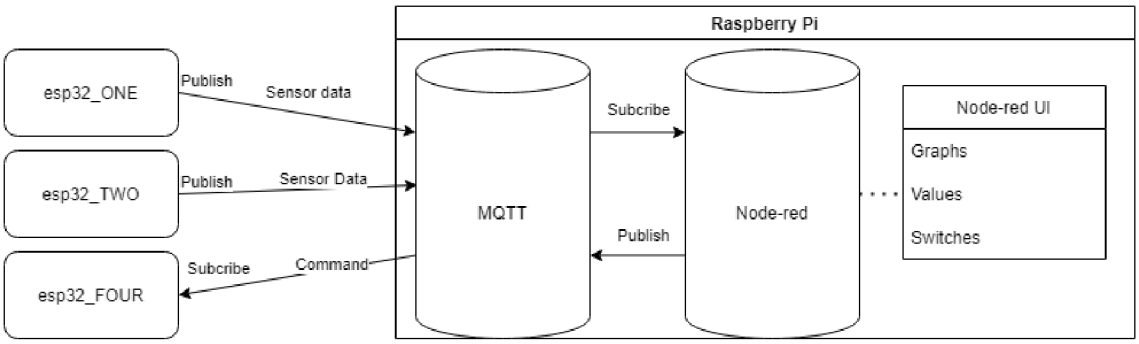
Tyto porty jsou rozděleny organizací ICANN na:

- **Známé porty** – v rozsahu 0 až 1023.
- **Registrované porty** –v rozsahu 1024 až 49151.
- **Dynamické a soukromé porty** – v 49152 až 65535

4 Praktická část

4.1 Návrh

V praktické části budeme vytvářet systém domácí automatizace, kde budeme používat třikrát vývojové desky ESP32, které budou připojeny k lokální Wi-Fi síti. Tyto vývojové desky ESP32 budou zpracovávat data ze senzorů a také ovládání elektro komponentů, které k nim budou připojeny, následně tato data budou přijímat nebo odesílat přes Wi-Fi připojení pomocí MQTT protokolu na IP adresu Raspberry Pi 4, na kterém bude naistalována služba MQTT zprostředkovatele. Na Raspberry Pi bude naistalována služba Node-red, která nám umožní přístup k datům, jejich manipulaci a ovládání připojených prvků na vývojových deskách ESP32 a jejich grafickou vizualizaci. Zároveň v Node-red UI (User interface – uživatelské rozhraní) budeme monitorovat data o Raspberry Pi, například teplotu procesoru, využití procesoru, využití disku.



Obrázek 11. Návrh systému pro domácí automatizaci[autor]

4.1.1 Použité komponenty

- **Raspberry Pi 4 model B**

  Raspberry Pi 4 je nástupcem starého Raspberry Pi 3. Základem je 64bitový čtyřjádrový procesor s taktem 1,5 GHz Broadcom BCM2711. Ve srovnání s Raspberry Pi 3 se od května 2020 prodává ve třech variantách na základě velikosti RAM-1 GiB, 2 GiB, 4 GiB a 8 GiB. Napájení micro-USB bylo nahrazeno zdrojem USB-C. Jeden port HDMI byl nahrazen dvěma porty micro HDMI, Raspberry Pi 4 nyní podporuje 4K displeje a dva porty USB 2.0 byly nahrazeny porty USB 3.0.[20]

Obrázek 12. Raspberry Pi 4 model B [21]

Tabulka 2 – Specifikace Raspberry Pi 4 model B [20]

Procesor	• Broadcom BCM2711 • 1.5 GHz quad-core ARM Cortex-A72 • ARM v8 • 64-bit SoC
Paměť	• LPDDR4 • 1 GB / 2 GB / 4 GB / 8 GB
Konektivita	• 2,4GHz a 5GHz IEEE 802.11.b/g/n/ac Wi-Fi • Bluetooth 5.0 BLE • Gigabit Ethernet • 2 × USB 2.0 konektor

	• 2 × USB 3.0 konektor
GPIO	• GPIO header se 40 piny
Video a zvuk	• 2 × microHDMI 2.0 konektor (až 4Kp60) • MIPI DSI konektor pro připojení displeje • MIPI CSI konektor pro připojení kamery • čtyřpólový 3,5mm jack – výstup zvuku a kompozitního videa (PAL a NTSC)
Multimedia	• H.265 (4Kp60 dekódování) • H.264 (1080p60 dekódování, 1080p30 kódování) • OpenGL ES, 3.0 grafika
Úložiště	• Micro SD slot pro načítání operačního systému a ukládání dat
Napájení	• 5 V DC přes USB-C konektor (minimálně 3 A) • 5 V DC přes GPIO header (minimálně 3 A) • PoE – napájení přes ethernet (vyžaduje přídavný modul Raspberry Pi PoE HAT)

V projektu jsou využity dva typy vývojových desek ESP32:

- **Klon WeMos LOLIN 32 WiFi + Bluetooth**

  Vývojová deska s Wi-Fi modulem ESP-WROOM-32 dokáže realizovat bezdrátovou komunikaci, analýzu Wi-Fi sítě nebo podporovat síťové servery s nižšími nároky. ESP-WROOM-32 je programově kompatibilní s moduly ESP8266. Vývojová platforma je vhodná pro účely průmyslové, domácí automatizace nebo vzdělávání v rámci vytváření jednodušších síťových uzlů. Tento modul zajišťuje funkce jako čtení a odesílání naměřených dat, analýzu sítě, poskytování síťových služeb nebo ovládání jednodušších automatizačních prvků. Součástí vývojové platformy je i rozhraní Bluetooth.[22]

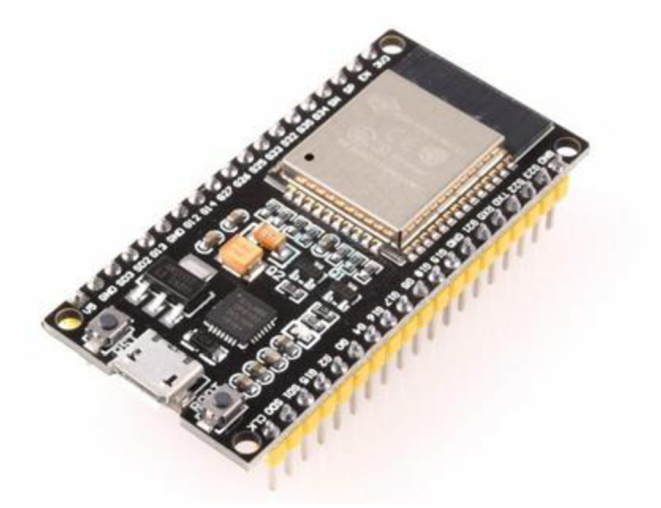
Obrázek 13. Klon WeMos LOLIN 32 WiFi + Bluetooth [22]

- **IoT ESP-WROOM-32 2.4GHz Dual-Mode WiFi+Bluetooth**

  Základem vývojové desky je dostatečně výkonný a individuálně ovladatelný dvoujádrový procesor Tensilica LX6, její taktovací frekvence je nastavitelná od 80 MHz do 240 MHz, obsahuje 520 kB RAM a 4 MB interní paměť SPI Flash je dostatečně velká pro komplexní aplikace. Díky integrované anténě může modul komunikovat bezdrátově přes 2,4GHz Wi-Fi a Bluetooth 4.0. USB rozhraní ovladače CP2102, přes které lze modul napájet a programovat.[23]

  ESP32 integruje velké množství periferií, kapacitní dotykové senzory, Hallovy senzory, nízkošumové zesilovače senzorů, vysokorychlostní SDIO rozhraní až 50 MHz, SD/MMC kartu a rozšíření na 16 MB, 3x UART, 3x SPI, 2 I2S přes SPI, 2 I2C. [23]

  K dispozici je 26 digitálních vstupních a výstupních pinů, 16 PMW pinů, 18 analogových vstupů a 2 digitálně-analogové DAC. 8kanálové infračervené ovládání.[23]

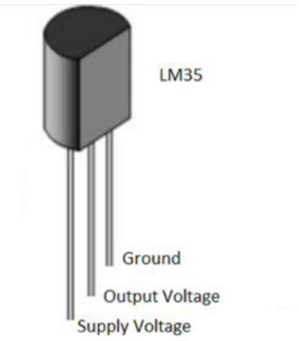


Obrázek 14. IoT ESP-WROOM-32 2.4GHz Dual-Mode WiFi+Bluetooth [23]

Komponenty, které jsou připojeny na vývojové desky ESP32 jsou:

- **Senzor teploty LM35C**

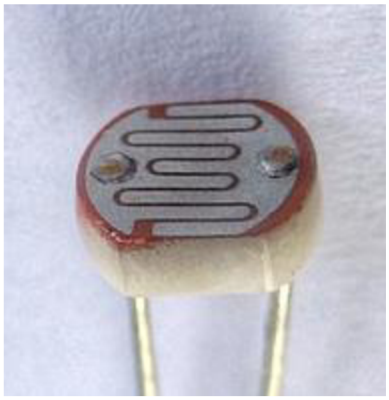
  Řada LM35 jsou přesná teplotní zařízení s integrovanými obvody, jejichž výstupní napětí je lineárně úměrné teplotě. LM35 spotřebovává ze zdroje pouze 60 µA a má velmi nízké samo zahřívání pod 0,1 °C v klidném vzduchu. LM35C je navržen pro provoz v teplotním rozsahu -40 °C až 110 °C. [24]



Obrázek 15. Senzor LM35[25]

- **Fotorezistor**

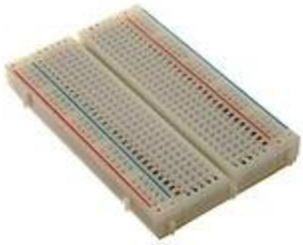
  Fotorezistory jsou rezistory vyrobené z polovodičových materiálů a jejich vodivost se mění se změnami jasu. Podle této charakteristiky mohou být fotorezistory vyrobeny v různých hodnotách a ozařovaných oblastech. Fotorezistory jsou široce používány v mnoha průmyslových odvětvích. [26]



Obrázek 16. Fotorezistor [27]

- **Nepájivé pole**

  Nepájená pole použitá pro návrh obvodů a prototypování jsou opakovaně použitelná. Při používání není potřeba svařovat komponenty, ale používá se propojovací vodič, který lze snadno zastrčit a kdykoliv přesunout.[28]



Obrázek 17. Nepájivé pole [29]

- **LED dioda 5 mm**

  LED je v elektrotechnice označení pro diodu, která emituje světlo.

Obrázek 18- LED dioda, červená [30]

- **Aktivní Bzučák**

  Aktivní bzučák je elektrosoučástka, která má v sobě oscilátor a při připojení k napětí vydává zvuk.

Obrázek 19. Aktivní Bzučák [31]

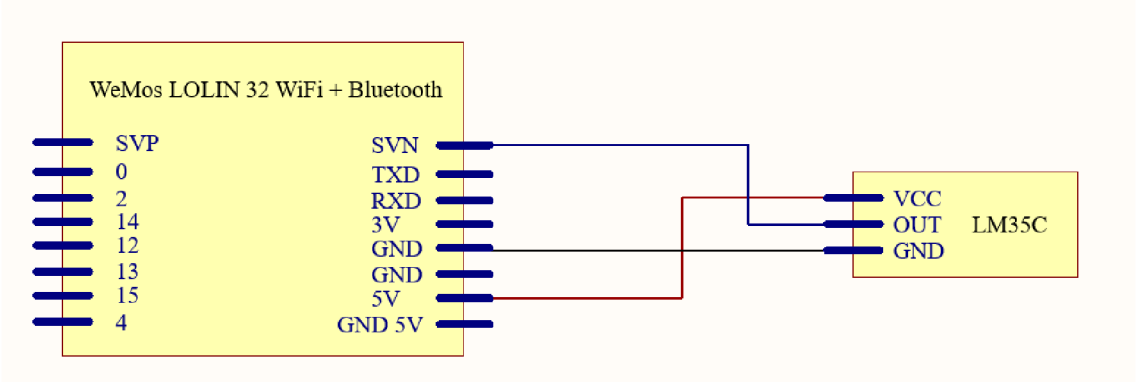
4.1.2 Schéma zapojení jednotlivých ESP32

Schéma propojení ESP vývojových desek a elektro komponentů byla vytvořena v programu CircuitMaker, který je bezplatný, a je možné tento software používat i pro komerční účely.

Tato schémata máme o číslované v angličtině.

- **ESP32-ONE**

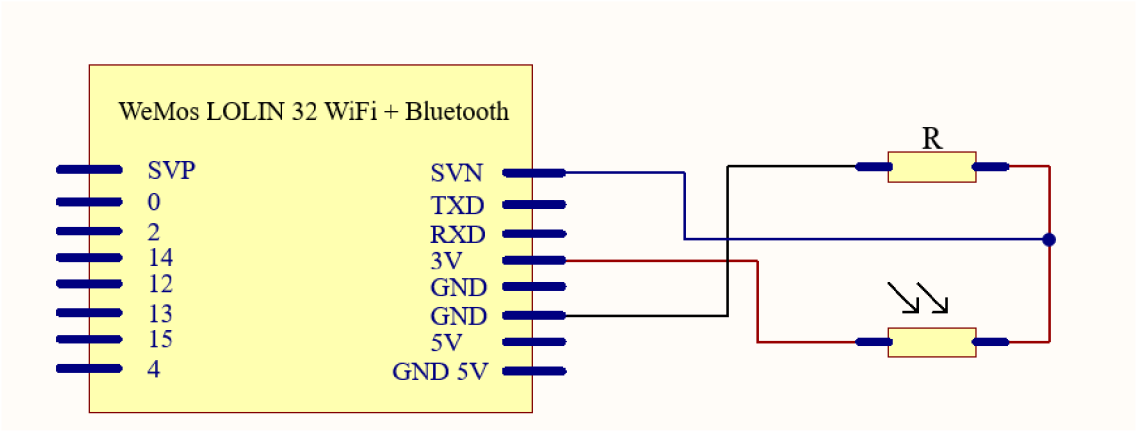
  Na první vývojové desce WeMos LOLIN 32 WiFi + Bluetooth tuto desku použijeme, protože je kompaktní a obsahuje méně I/O pinů. Potřebujeme pouze pin SVN (sensor VN), který je pouze vstupem, dále pin GND (Ground) a pin 5V pro napájení. Teplotní snímač LM35 má tři vývody: vstupní napětí, výstupní napětí a zem nebo GND.



Obrázek 20. Schéma zapojení ESP32 ONE [autor]

- **ESP32-TWO**

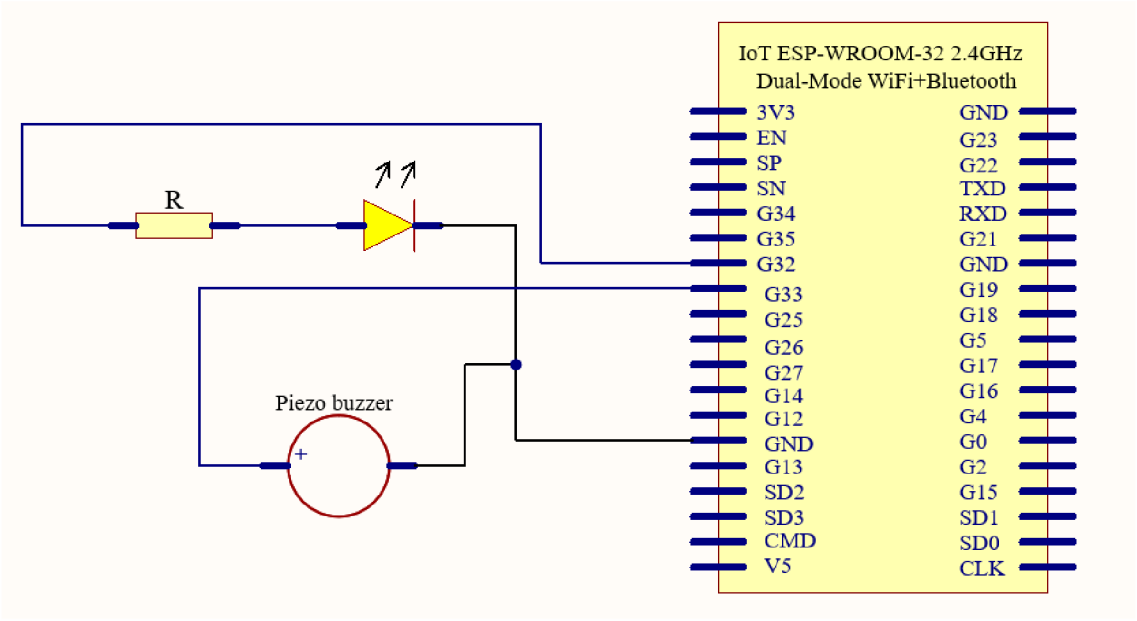
  Na druhé vývojové desce použijeme znovu WeMos LOLIN 32 WIFI + Bluetooth, na vývojovou desku připojíme fotorezistor ten má pouze dva vývody. Na první vývod zapojíme pin 3V, na druhý vývod zapojíme SVN a ještě mezi druhý vývod a pin SVN zapojíme rezistor, který směřuje na pin GND.



Obrázek 21. Schéma zapojení ESP32 TWO [autor]

- **ESP32-THREE**

  Za třetí, použijeme vývojovou desku IoT ESP-WROOM-32 2,4 GHz Dual-Mode WiFi + Bluetooth z několika důvodů, za prvé proto, že tato vývojová deska obsahuje více I/O pinů, a za druhé proto, že tato vývojová deska disponuje více piny na ADC1, protože když je vývojová deska připojena k Wi-Fi, ovladač Wi-Fi využívá všechny piny na ADC2. K vývojové desce je připojena LED a aktivní bzučák. Pin 32 GPIO je zapojen do série před odporem a poté připojen ke kladnému pólu LED a záporný pól LED je připojen ke GND na vývojové desce. Aktivním bzučákem bude kladná svorka připojená k 33 pinu GPIO a záporná svorka připojená ke GND.



Obrázek 22. Schéma zapojení ESP32 THREE [autor]

4.2 Implementace

V části implementace je popsán kód pouze „ESP32 ONE“ jak byl nazván, také zde nalezneme část kódu z „ESP32 THREE“. Rozdíl mezi těmito zařízeními je v tom, že ESP32 ONE pouze data odesílá z MQTT serveru a ESP32 THREE pouze přijímá. Kód byl vytvořen v aplikaci Arduino IDE.

4.2.1 Popis vytvořených metod a kódu ESP32

4.2.1.1 Zdrojový kód globální konstanty a proměnné

```
#include <Arduino.h>
//knihovna na Wi-Fi
#include <WiFi.h>
//knihovna na MQTT
#include <PubSubClient.h>

//WIFI konstanta pro jméno síťě a heslo
const char* ssid = "Cukrovar1";
const char* password = "1158jarda";

//MQTT
```

```
//konstanta pro ip hostujeme MQTT(v našem případě na raspberry pi)
#define mqtt_server "192.168.1.113"
WiFiClient wifiClientEsp;
PubSubClient client(wifiClientEsp);

//MQTT Topics
//definování našeho topicu
#define mqttTempC "esp32One/tempSensor/tempC"

//definování pinu na kterém přímáme data
#define PIN_LM35 39

//globální proměná pro teplotu
float tempC;

//proměná času místo funkce delay
unsigned long millisNow = 0;
unsigned int sendDelay = 1000;
```

Jak je vidět, ve zdrojovém kódu byla vložena knihovna pro Wi-Fi a MQTT, abychom mohli používat funkce z těchto knihoven. Tyto knihovny jsou volně dostupné skrze manažér knihoven, v aplikaci Arduino IDE. Dále vytváříme globální konstanty, které v sobě uchovávají informace název sítě a heslo. Definujeme konstantu, v níž se nachází ip adresa MQTT serveru, v našem případě ip adresa raspberry pi, inicializujeme instanci „WiFiClient“ a „PubSubClient“, také definujeme v tomto případě piny, na kterých budeme číst nebo odesílat data.

4.2.1.2 Metoda wifiSetup()

```
// metoda Wi-Fi nastavení
void wifiSetup(){
  Serial.println();
  Serial.print("Connecting to ");
  //Vypíše název sítě ke které se napojuje
  Serial.println(ssid);
```

```
  //Přepne Wi-Fi mode do station nastavení
  WiFi.mode(WIFI_STA);
  WiFi.disconnect();
  delay(2000);

  //pokusí se připojit pomocí nazvu sítě a hesla
  WiFi.begin(ssid, password);

  //Opakuj dokud esp není připojeno k Wi-Fi
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("Wi-Fi connected");
  Serial.println("IP address: ");
  //Vypíše IP adresu na které se připojil do lokální sítě
  Serial.println(WiFi.localIP());
}
```

Metodu wifiSetup() pro vytvoření a nastavení Wi-Fi spojení s routerem na našem ESP modulu, je na všech modulech stejná.

Použité funkce jsou z knihovny „WiFiClient“:

- **WiFi.mode()** – Je nastavení tří modů, v jakém bude ESP fungovat, první mode je stanice parametr „WIFI_STA“, přístupový bod parametr „WIFI_AP“ a kombinovaný parametr „WIFI_STA_AP“.
- **WiFi.disconnect()** – Funkce, která nás od Wi-Fi odpojí.
- **WiFi.begin()** – Započne připojení k Wi-Fi, vkládají se dva parametry ssid (název sítě) a password (heslo k síti).
- **WiFi.status()** – Funkce, která nám vrací stav, ve kterém se Wi-Fi nachází například „WL_CONNECTED“ při připojeni.

- **WiFi.localIP()** – Funkce vrátí IP adresu desky v lokální síti.

4.2.1.3 Metoda setup()

```
void setup() {
  //rychlost na 115200 baudů
  Serial.begin(115200);
  // definování pin nastavení na vstup
  pinMode(PIN_LM35, INPUT);
  //volání metody na nastavení Wi-Fi
  wifiSetup();
  //Nastavení serveru MQTT pro klienta, ip adressa a port na kterém běží
  client.setServer(mqtt_server, 1883);
}
```

Metoda setup() je volána při každém startu ESP a proběhne pouze jednou. Také zde voláme naši metodu wifiSetup(), kdybychom chtěli i přijímat z MQTT serveru museli bychom definovat ještě funkci „client.setCallback()“.

Použité funkce:

- **Serial.begin()** – Funkce, která nám nastaví rychlost sériové komunikace na 115200 baudů, pro zobrazení hodnot v sériovém monitoru je potřeba stejnou hodnotu nastavit i tam.
- **pinMode()** – Přijímá dva parametry číslo pinu podle pinout vývojové desky a na jaký mód vstupní (INPUT) nebo výstupní (OUTPUT).
- **client.setServer()** – Funkce, která přijímá dva parametry, první je IP adresa zařízení kde je MQTT broker a druhý parametr odesíláme port služby.
- **client.setCallback()** – Nastavuje metodu, která se stará o tzv. (zpětné volání).

4.2.1.4 Metoda loop()

```
void loop() {
  // po každé kontrolujeme jestli je klient připojen na MQTT server
  if (!client.connected()){
```

```
    //pokud není tak se znovu chceme připojit
    reconnectAndConnectMQTT();
  }
  //tato podmínka slouží pro publikování se zpožděním
  if (millis() > millisNow + sendDelay) {
    millisNow = millis();
    getValueInCelsius();
    client.publish(mqttTempC, String(tempC).c_str(),true);
  }

  client.loop();
}
```

Metoda loop() se opakuje stále dokola a umožňuje nám program měnit a reagovat na změny. Ve funkci je podmínka, kde se ptáme, jestli je MQTT klient připojen a pokud není, tak se zavolá funkce pro znovu připojení klienta k MQTT serveru. Druhá podmínka nám vytváří zpoždění, abychom odesílali zprávy na MQTT server každou jednu sekundu.

Použité funkce:

- **client.connected()** – Nám vrací hodnotu „true“, pokud je klient připojen a hodnotu „false“, když není.
- **client.publish()** – Je funkce, která odešle náš „topic“ a „message“ na MQTT server.
- **client.loop()** – tato funkce je by měla být volána pravidelně pokud chceme příchozí data z MQTT serveru a udržovat spojení se serverem.

4.2.1.5 Metoda reconnectAndConnectMQTT()

```
//metoda pro znovu připojení k MQTT a připojení
void reconnectAndConnectMQTT() {
  int counter = 0;
  // opakuj a pokud není klient připojen k MQTT
  while (!client.connected()) {
```

```
    //pokud se 5 krát nepřipojí
    if (counter==5){
      //restartujeme ESP
      ESP.restart();
    }
    counter+=1;
    Serial.print("Attempting MQTT connection... ");

    if (client.connect("esp32Three")) {
      Serial.println("connected");
      //topics o kterých chceme ze serveru dostávat data
      client.subscribe(mqttLedOnOff);
      client.subscribe(mqttBuzzer);
    } else {
      Serial.print("failed, rc=");
      //výpis chybi proč se nemůže připojit
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");
      delay(5000);
    }
  }
  client.subscribe(mqttLedOnOff);
  client.subscribe(mqttBuzzer);
}
```

Metoda reconnectAndConnectMQTT(), která se stará o znovu připojení k MQTT serveru, v případě, kdyby spojení selhalo. Metoda se bude snažit připojit k MQTT serveru, pokud se jí to nepodaří vypíše stav, proč se nepovedlo připojit. Pokud se to nepodaří do pěti pokusů, restartuje ESP modul.

Použité funkce:

- **client.connect()** – Funkce slouží pro napojení na MQTT server a na text vkládaný do funkce značí název připojeného klienta.

- **client.state()** – Funkce, která nám vrátí int hodnotu, podle které se dá dohledat stav v jakém je naše spojení.
- **client.subscribe()** – Funkce do které vkládáme jeden parametr „topic“ typu string, abychom o tomto tématu dostávali data z MQTT serveru.

4.2.1.6 Metoda mqttCallback()

```
//tato metoda přijme tři parametry a to je topic, zprávu a délku zprávy
void mqttCallback(char* topic, byte* message, unsigned int length) {
  Serial.print("MQTT message received on topic: ");
  //vypsaní topicu o jaký se jedná
  Serial.print(topic);
  Serial.print(". Message: ");
  String messageTemp;
  //vypsaní zprávy po charakteru a uložení do proměnné
  for (int i = 0; i < length; i++) {
    messageTemp += (char)message[i];
  }
  //vypsání přijatého příkazu
  Serial.println(messageTemp);
  //rozhodnutí i jaký příkaz se jedná a jeho vykonání
  if (messageTemp == "ledOn"){
     digitalWrite(PIN_LED, HIGH);
  }
  if (messageTemp == "ledOff"){
     digitalWrite(PIN_LED, LOW);
  }
  if (messageTemp == "buzz"){
     for (char i = 0; i<100; i++){
          digitalWrite(PIN_BUZZER,HIGH);
          delay(1);
          digitalWrite(PIN_BUZZER,LOW);
          delay(1);
     }
  }
}
```

Metoda mqttCallback zajišťuje přijetí téma a zprávy z MQTT serveru. Tuto metodu je potřeba nastavit v setup() metodě, pomocí funkce klientu „client.setCallback(mqttCallback)“. Metoda má tři parametry „topic“, „message“ a „length“, tyto parametry následně zpracuje a rozhodne co se má vykonat, například rozsvícení led diody.

Použité funkce:

- **digitalWrite() –** Do funkce se zadávají dva parametry „pin“ na jakém máme komponent připojen a „value“ má dvě možnosti „HIGH“ nebo „LOW“.

4.2.1.7 Metoda getValueInCelsius()

```
// metoda pro získání stupňů celsia ze sensor
void getValueInCelsius() {
 // naměřená hodnota
  int value = analogRead(PIN_LM35);
 // výpočet milivoltů = hodnota z pinu děleno rozlišením to celé krát
napětí
  float milliVolt = (value /2047.5)*3300;
  //uložení teploty do globální proměnné tempC vzorcem milivolty děleno
deseti, jelikož rozdíl teplot je 10 mV/Celsius
  tempC = milliVolt / 10;
  Serial.print("Teplopta v pokoji je: ");
  Serial.print(tempC);
  Serial.print("°C");
  Serial.println("");
}
```

Zde vidíme metodu, která nám čte hodnoty z pinu, na kterém máme připojen senzor a převádí analogové hodnoty na stupně Celsia, výsledek vypisuje do sériového monitoru. Výpočet této hodnoty se odvíjí od ADC na esp32, které je 12bitové a jeho rozlišení je od 0 do 4095, ale také je nelineární to znamená, že není schopen rozlišovat napětí 3,3V a 3,2, proto dostaneme rozlišení 4095. Po testu rozlišení 4095

teplota nebyla totožná s rtuťovým teploměrem, proto bylo použito poloviční rozlišení, které již souhlasilo s teploměrem.

Použité funkce:

- **analogRead()** – Funkce má jeden parametr „pin“ zařízení ze kterého čte hodnotu.

4.2.2 Instalace OS na Raspberry Pi

Pro instalaci operačního systému na Raspberry Pi potřebujeme microSD kartu jako úložný prostor Raspberry Pi. Ke čtečce karet PC se připojíme přes USB. Aplikaci Raspberry Pi Imager si stáhneme do počítače z oficiálních stránek, kde si vybereme operační systém, jaký chceme. Pro tento projekt byl vybrán operační systém Raspberry Pi OS Lite (32bitový). Poté v aplikaci vybereme umístění operačního systému na mikro SD kartě, vložíme ji do čtečky karet a stiskneme tlačítko zápisu. Před vložením mikro SD karty do zařízení si v systémové složce vytvoříme SSH soubor, abychom mohli pomocí aplikace PuTTY komunikovat s Raspberry Pi. Operační systém Raspeberry Pi OS Lite je negrafické prostředí s prostým textem.

4.2.3 Služby na Raspberry Pi

Pro instalaci služeb na MQTT a Node-red na Raspberry Pi, byl použit program PuTTY, který slouží jako klient vzdálené komunikace s Raspberry Pi, pomocí zabezpečeného komunikačního protokolu SSH.

Po připojení se zobrazí příkazová řádka v textové formě a za jakého uživatele se chceme připojit, výchozím přihlašovacím jménem je „pi“, následně se nás dotáže na heslo k tomuto přihlášení, výchozím heslem je „raspberry“. Následně můžeme začít ovládat naše Raspberry Pi, v příkazové řádce bychom měli vidět řádku v niž bude napsáno „pi@raspberrypi ~ $“, což označuje, za jakého uživatele jsme přihlášeni a po zavináči název hostitele. V příkazové řádce Raspbery Pi OS, používáme příkazy Debianu což je distribuce Linuxu. Když se přihlásíme za uživatele pi tak budeme přihlášení za normálního uživatele, pokud bychom chtěli pravomoc na úrovni

administrátora, musíme použít na začátku příkazu „sudo“ (super user do). Tento super uživatel v systému existuje, kvůli zabezpečení počítače.

Před instalací jakéhokoliv softwaru bychom měli nejdříve aktualizovat seznam softwarových balíčků na našem Raspberry Pi pomocí příkazu „sudo apt update“, po dokončení dále ještě příkaz „sudo apt full-upgrade“, který nám aktualizuje všechny staré baličky na novou verzi.

Pro instalaci balíčku na naše zařízení se používá příkaz „sudo apt install <package-name>“, název balíčku můžeme nalézt na webových stránkách služby nebo balíčku který chceme naistalovat. Pro odinstalování balíčku se používá příkaz „sudo apt remove <package-name>“.

Když chceme zjistit, jestli služba je aktivní nebo deaktivována, použijeme příkaz „sudo systemctl status <service-name>“, v případě že služba není aktivní a chceme, aby služba byla spuštěna při spuštění Raspberry Pi, použijeme příkaz „sudo systemctl enable <service-name>“

- **Instalace služby MQTT**
  Pro instalaci služby MQTT brokera musíme stáhnout balíček od organizace Eclipse Mosquitto, která poskytuje open source MQTT brokera. Pro stáhnutí balíčku použijeme příkaz „sudo apt install mosquitto“ a následně příkaz „sudo apt install mosquitto-clients“. Když chceme Raspberry Pi používat jako server pro MQTT brokera, tak chceme, aby se nám služba spouštěla při spuštění našeho zařízení, proto použijeme příkaz „sudo systemctl enable mosquitto“. Pak ještě chceme zjistit, jestli byl balíček správně naistalován příkazem „mosquitto“, ten nám vypíše verzi, confiq který je použit a port na kterém je služba dostupná.

- **Instalace služby Node-red**
  Pro instalaci balíčku služby Node-red je na oficiálních stránkách poskytnut návod na instalaci pro Raspberry Pi. Zde nám poskytují script, který nám

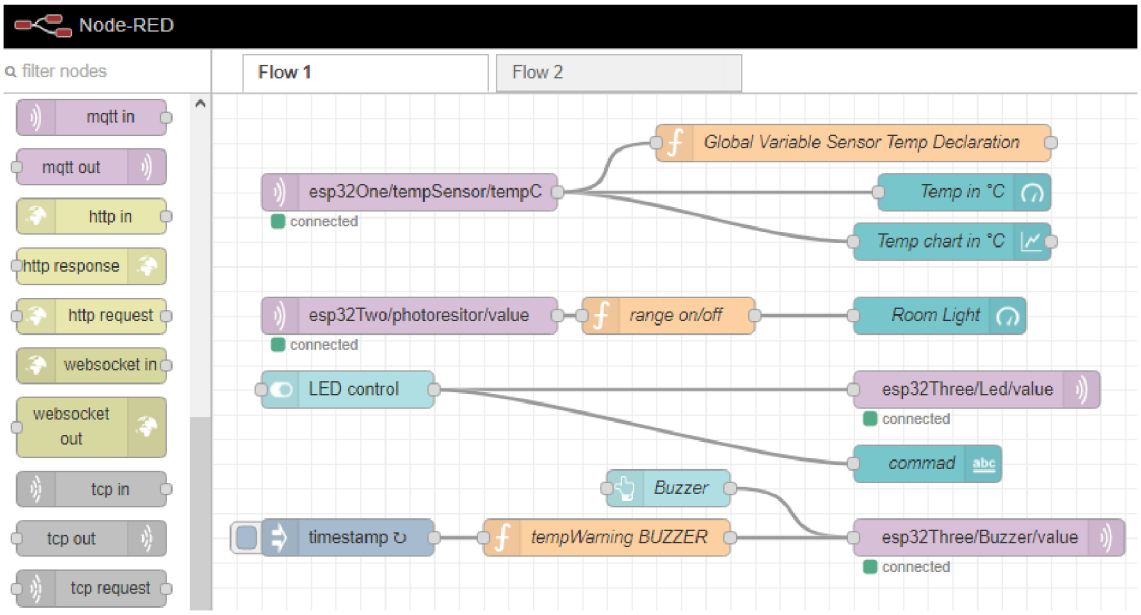
naistaluje potřebné balíčky pro Node-red, tyto balíčky jsou Node.js, npm a samotný Node-red. Tento script „bash <(curl – sL https://raw.githubusercontent.com/node-red/linux-installers/master/deb/update-nodejs-and-nodered)“ vložíme do příkazové řádky Raspberry Pi. Když instalace proběhne, použijeme příkaz „sudo systemctl enable node-red“, aby nám služba fungovala při každém startu zařízení a příkazem „node-red“ zkontrolujeme, jestli se v pořádku naistalovala a na jakém portu je provozována.

4.2.4 Popis vytvořených toků v Node-Red

Pro vytvoření vizuální části projektu využíváme Node-red, který je založený na programování flow(toku), tyto toky v sobě obsahují nodes(uzly), každý uzel má předem daný účel na co je využíván. Uzly mohou být mezi sebou spojeny a předávají si data nebo vykonají činnost na základě přijatých dat.

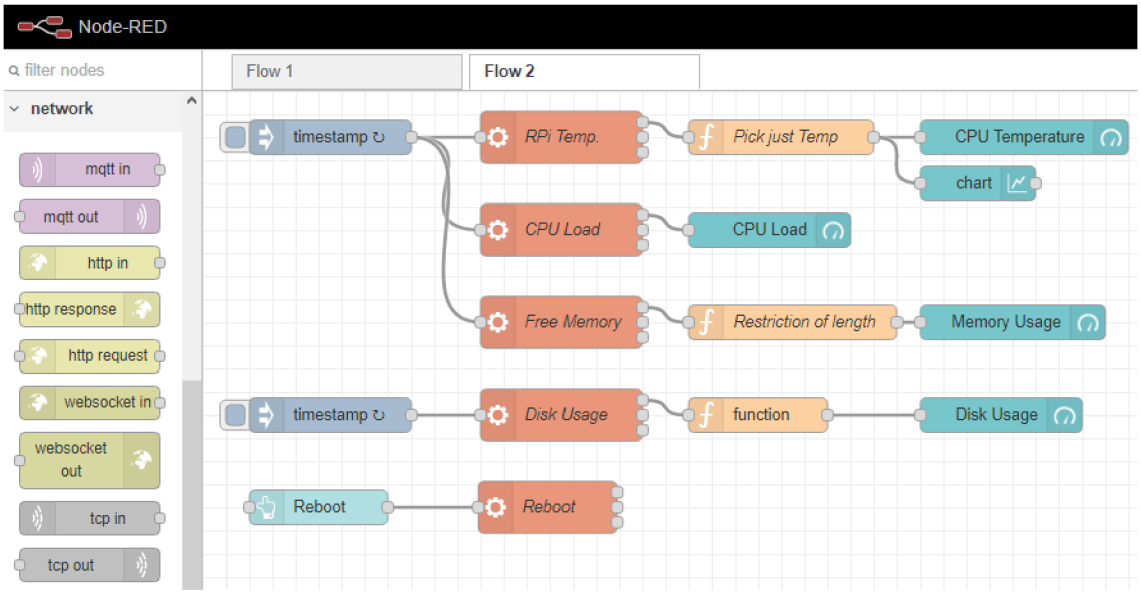
Při prvním spuštění zde jsou jen základní uzly, pro přidaní těchto uzlů musíme v nastavení otevřít manažer palet, který umožňuje přidání nových uzlů. V projektu je využita paleta „node-red-dashboard“ k vytvoření grafického uživatelského rozhraní, tato paleta přidává uzly například na grafy, tlačítka, přepínače atd., také nám přidá možnost dashboard, kde můžeme měnit rozložení a velikosti grafických komponent na stránce.

Po vložení nebo úpravě uzlů v toku je potřeba zmačknout tlačítko deploy (nasadit), které náš tok zprovozní a uloží.



Obrázek 23. Tok č.1, Schéma uzlů ESP [autor]

Jak je vidět na obrázku č. 23, máme zde tok neboli flow, který se stará o přijetí dat z ESP32 modulů za pomocí MQTT brokera. V toku používáme uzly „mqtt in“, do kterých je vložen topik, ze kterého chceme odebírat data. Tyto uzly většinou směřují do uzlu funkce napsané v Javascriptu, kde můžeme například vytvořit podmínku, jestli se budou data odesílat dále do grafických uzlů viditelných v uživatelském rozhraní. Uzel funkce může mít tři různé průběhy „On Start“, „On Message“ a „On End“. První vyvolá průběh funkce při spuštění služby, druhá vyvolá funkci, když příjme data z jiného uzlu a třetí vyvolá funkci při dokončení metody v uzlu. Také je použit uzel „inject“, který odesílá námi vytvořenou zprávu datového typu podle výběru. Zprávu tohoto uzlu můžeme posílat ručně kliknutím nebo nastavit časovač, který ji bude odesílat opakovaně za určitou dobu. V projektu tento uzel používáme pro vyvolání funkce, která pošle příkaz na uzel „mqtt out“, který příkaz publikuje na topik, který je zde uveden, jen pokud teplota ze sensoru teploty z jiného topik přesáhne stanovenou teplotu, která je uložena v globální proměnné. Také jsou zde uzly pro grafickou reprezentaci v uživatelském rozhraní, kterým posíláme data nebo jimi odesíláme data, to jsou uzly tlačítka, přepínače, grafu, měřidla a textu. Přepínač odesílá příkaz, který rozsvítí nebo zhasne led diodu a tlačítko zvonku pošle příkaz na zazvonění toto tlačítko je pouze na otestování, jestli zvonek funguje.



Obrázek 24. Tok č. 2, Schéma uzlů Raspberry Pi[autor]

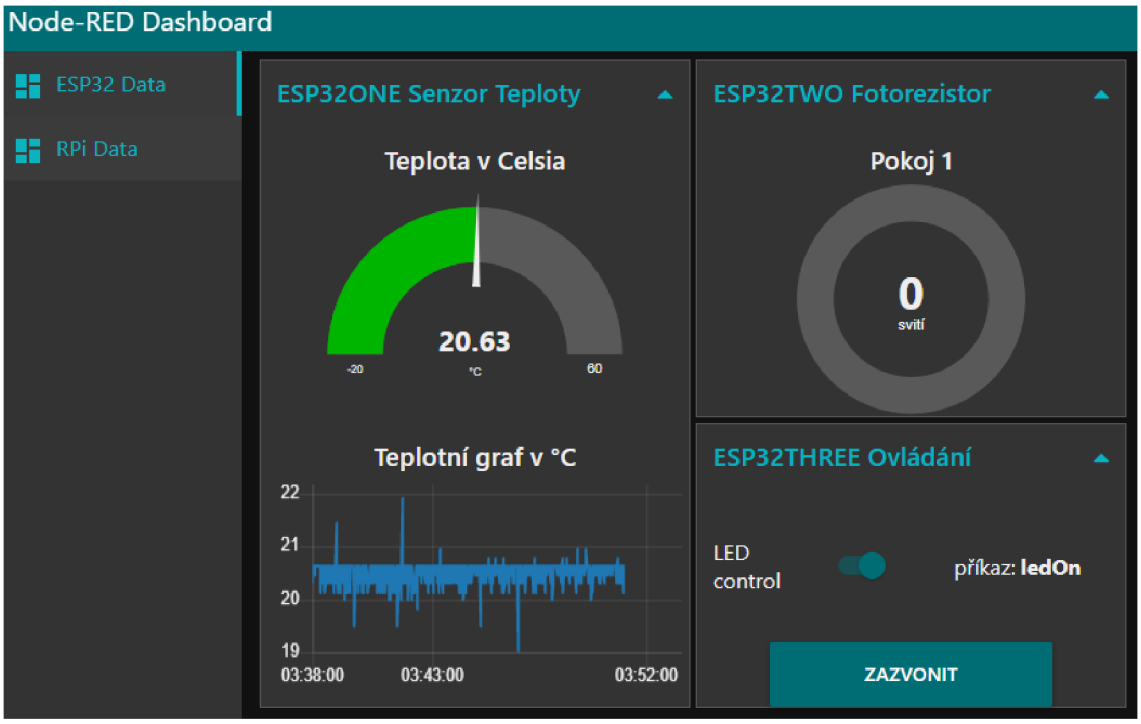
Na obrázku č. 24 je vidět tok, který získává data přímo z Raspberry Pi, pomocí stejných uzlů jako u obrázku č. 23, ale ještě je zde navíc uzel „exec“ který posílá příkaz do příkazové řádky našeho zařízení a odpověď na tento příkaz posílá pomocí výstupu tohoto uzlu. Tento tok zobrazuje data jako jsou teplota zařízení, zatížení procesoru, dostupnou volnou paměť RAM, využití disku a taky je zde tlačítko, které zařízení restartuje.

4.3 Uživatelské rozhraní Node-red

Cílem je vytvořit jednoduché uživatelské rozhraní. To bude intuitivní a snadno ovladatelné, takže je vytvořeno pomocí palety v Node-red, nazývané "node-red-dashboard", a umožňuje nám vytvořit graficky přívětivé rozhraní.

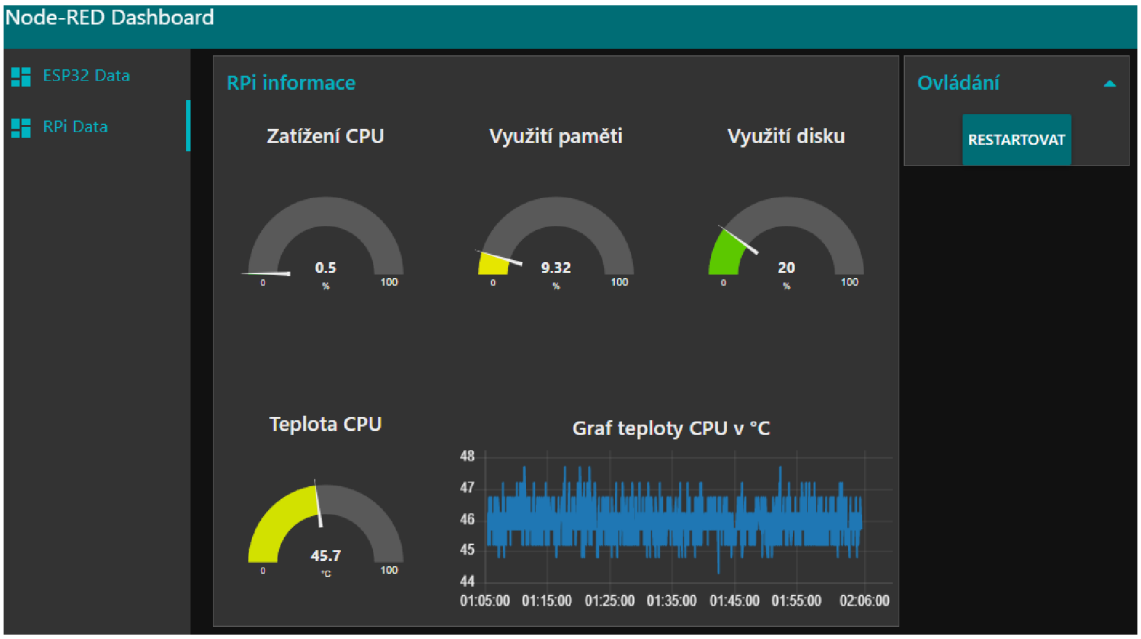
Pro vytvoření uživatelského rozhraní je použita služba Node-red nainstalovaná na Raspberry Pi. Pomocí webového prohlížeče na místním síťovém zařízení lze zadat IP adresu a port služby 1880 Raspberry Pi. Ve správci palet byla nainstalována paleta node-red-dashboard, která do seznamu node (uzlů) přidává nové prvky, jako jsou textová pole, tlačítka, přepínače, HTML řetězce atd. Umožňuje nám také měnit velikost a polohu prvků. Pro vizualizaci vytvořeného UI (uživatelského rozhraní) je třeba znovu zadat IP adresu Raspberry Pi a servisní port 1880 za dvojtečkou ve

webovém prohlížeči a přidat lomítko, vzorově „[ip-adresa-raspberry-pi]:[port-služby]/ui ". V nastavení routeru je třeba nastavit rezervaci IP adres, jinak se může změnit IP adresa Raspberry Pi v naší lokální síti.



Obrázek 25. Node-Red, Uživatelské rozhraní, ESP32 data [autor]

Obrázek č. 25 nám zobrazuje, jak vypadá uživatelské rozhraní pro ovládání a monitorování dat z naších ESP32 modulů. Na levé straně obrázku můžeme vidět menu, kde si vybereme záložku tok, který chceme zobrazit. Také je zde ukázáno měřidlo, které nám zobrazuje aktuální teplotu a graf aktuální teploty v posledních třiceti minutách ze senzoru, který je připojen na ESP32ONE. Dále je zde indikátor, jestli světlo v pokoji svítí nebo je zhasnuto, tato data dostává ze senzoru na ESP32TWO. Poslední prvek v uživatelském rozhraní je přepínač, který ovládá led diodu a tlačítko které ovládá zvonek na ESP32THREE.



Obrázek 26. Node-Red, Uživatelské rozhraní, RPi data [autor]

Na obrázku č. 26 je uživatelské rozhraní druhého toku nazvané „RPi data“, toto rozhraní obsahuje měřidla, která zobrazují zatížení procesoru, využití paměti, využití disku, teplotu procesoru. Pak graf teploty procesoru, který zobrazuje aktuální teplotu v průběhu jedné hodiny a tlačítko „restartovat“, které po zmačknutí restartuje Raspberry Pi.

5 Závěry a doporučení

V této práci byl vytvořen přehledný systém domácí automatizace, který využívá ESP32 vývojové desky a senzorů a elektro komponentů na nich připojené. Komunikuje pomocí protokolu MQTT, který slouží pro přenášení zpráv mezi zařízeními, skrze lokální Wi-Fi síť. Pro vytvoření uživatelského rozhraní byl použit vývojový nástroj Node-red, který také komunikuje pomocí MQTT a publikuje a odesílá zprávy do jednotlivých zařízení. Zprávy přijímané ze senzorů používá pro automatizaci úkonů.

V teoretické části byla osvětlena platforma vývojových desek ESP32, pinout modulu, na kterém byla práce vytvořena, vývojové prostředí, které bylo použito pro implementaci, dále byl vysvětlen způsob komunikace pro nahrání programu na

vývojovou desku. Také byla vysvětlena platforma Raspberry Pi, která byla v práci použita jako server a byla přiblížena aplikace vzdálené komunikace a zabezpečeného protokolu SSH. Vysvětlena byla zkratka Wi-Fi a popis protokolů pro komunikaci v síti a jejich využití.

V praktické části byl vytvořen návrh propojení tohoto systému domácí automatizace a popsány použité komponenty. Také bylo vytvořeno schéma zapojení a popis jednotlivých vývojových desek ESP32 s použitými komponenty, byl uveden program pro vytvoření schémat. Dále následuje implementační část, kde byly vysvětleny použité knihovny, funkce a vytvořené metody, které byly použity pro implementaci a nahrání do vývojové desky, následně instalace operačního systému a služeb na Raspberry Pi. Popis a vysvětlení tokového programování a jeho komponent. Na závěr byla ukázána vizualizace uživatelského rozhraní a jeho popis.

Hlavní přínosy této práce spočívají ve vysvětlení použití jednotlivých programů a osvětlení bezdrátové komunikace mezi zařízeními a komunikace pomocí sériové sběrnice, která se dá využít i na jiných platformách než jen ESP32. Systém byl vytvořen na nepájivých polích, takže je možná další experimentální rozšiřitelnost o různé druhy senzorů nebo elektronických komponent. Celková cena komponentů a zařízeních použité v práci je přibližně do tří tisíc korun. Protože je práce velice dostupná, mohla být sloužit jako výukový materiál pro práci s mikrokontrolery.

6 Seznam použité literatury

[1] Kolban's book on ESP32 by Neil Kolban [Leanpub PDF/iPad/Kindle]. Leanpub: Publish Early, Publish Often [online]. Copyright © 2017 [cit. 28.04.2022]. Dostupné z: https://leanpub.com/kolban-ESP32

[2] ESP32 Pinout Reference: Which GPIO pins should you use? | Random Nerd Tutorials. Random Nerd Tutorials | Learn ESP32, ESP8266, Arduino, and Raspberry Pi [online]. Copyright © 2018 [cit. 28.04.2022]. Dostupné z: https://randomnerdtutorials.com/esp32-pinout-reference-gpios/

[3] setup() - Arduino Reference. Arduino - Home [online]. Dostupné z: https://www.arduino.cc/reference/en/language/structure/sketch/setup/

[4] loop() - Arduino Reference. Arduino - Home [online]. Dostupné z: https://www.arduino.cc/reference/en/language/structure/sketch/loop/

[5] Campbell, S., 2021. Basics of UART Communication. [online] Circuit Basics. [cit. 4.11.2021]. Dostupné z: https://www.circuitbasics.com/basics-uart-communication

[6] Teach, Learn, and Make with Raspberry Pi [online]. Dostupné z: https://www.raspberrypi.org/help/what-%20is-a-raspberry-pi

[7] PuTTY – a free SSH and telnet client for Windows. PuTTY - a free SSH and telnet client for Windows [online]. Dostupné z: https://www.putty.org/

[8] The SSH protocol. SSH [online]. [cit. 2022-04-28]. Dostupné z: https://www.ssh.com/academy/ssh

[9] Ylönen, T., 1996. SSH — Secure Login Connections over the Internet. [online] Usenix.org. [cit. 4.11.2021] Dostupné z: https://www.usenix.org/legacy/publications/library/proceedings/sec96/full_papers/ylonen/index.html

[10] Node-RED. Node-RED [online]. Dostupné z: https://nodered.org

[11] IEEE 802.11, The Working Group Setting the Standards for Wireless LANs. IEEE802 [online]. Dostupné z: https://www.ieee802.org/11/

[12] MQTT Version 5.0. [online]. Dostupné z: https://docs.oasis-open.org/mqtt/mqtt/v5.0/mqtt-v5.0.html

[13] MQTT - The Standard for IoT Messaging. MQTT - The Standard for IoT Messaging [online]. Copyright © 2022 MQTT.org [cit. 28.04.2022]. Dostupné z: https://mqtt.org/

[14] The TCP/IP Guide - TCP/IP Overview and History. Welcome to The TCP/IP Guide! [online]. Dostupné z: http://www.tcpipguide.com/free/t_TCPIPOverviewandHistory-3.htm

[15] Tcpip vrstvy.svg – Wikipedie. [online]. Dostupné z: <https://cs.m.wikipedia.org/wiki/Soubor:Tcpip_vrstvy>.

[16] Exploring the anatomy of a data packet | TechRepublic. TechRepublic: News, Tips & Advice for Technology Professionals [online]. Copyright © 2022 TechnologyAdvice. All rights reserved. [cit. 28.04.2022]. Dostupné z: https://www.techrepublic.com/article/exploring-the-anatomy-of-a-data-packet/

[17] TCP provides an error-free, point-to-point connection. The Linux Information Project (LINFO) Home Page [online]. Dostupné z: http://www.linfo.org/tcp.html

[18] UDP - User Datagram Protocol | IPv6.com. IPv6.com - Learn About Internet Protocol Version 6 and Computer Networking | Official IPv6 Website [online]. Copyright © Copyright 2019 [cit. 15.08.2022]. Dostupné z: https://www.ipv6.com/general/udp-user-datagram-protocol/

[19] List of TCP and UDP port numbers – Wikipedia. [online]. Dostupné z: https://en.wikipedia.org/wiki/List_of_TCP_and_UDP_port_numbers

[20] Raspberry Pi Datasheets [online]. Copyright © [cit. 28.04.2022]. Dostupné z: https://datasheets.raspberrypi.com/rpi4/raspberry-pi-4-datasheet.pdf

[21] File: Raspberry Pi 4 Model B - Side.jpg - Wikimedia Commons. [online]. Dostupné z: https://commons.wikimedia.org/wiki/File:Raspberry_Pi_4_Model_B_-_Side.jpg

[22] WeMos LOLIN 32 WiFi + Bluetooth, *dratek.cz:* [online].[cit. 4.11.2021]. Dostupné z: https://dratek.cz/docs/produkty/0/637/1501223637.pdf

[23] IoT ESP-WROOM-32 2.4GHz Dual-Mode WiFi+Bluetooth rev.1, CP2102 | laskakit.cz. LASKARDUINO.cz | by Makers for Makers [online]. Dostupné z: https://www.laskakit.cz/iot-esp-32s-2-4ghz-dual-mode-wifi-bluetooth-rev-1--cp2102/#productDiscussion

[24] LM35 data sheet, product information and support | TI.com. Analog | Embedded processing | Semiconductor company | TI.com [online]. Dostupné z: https://www.ti.com/product/LM35#tech-docs

[25] LM35 Analog Temperature Sensor | OpzLab. | OpzLab [online]. Copyright © OpzLab 2022 [cit. 28.04.2022]. Dostupné z: https://www.opzlab.com/product/lm35-analog-temperature-sensor/

[26] Photoresistor – EEPOWER [online]. Dostupné z: https://eepower.com/resistor-guide/resistor-types/photo-resistor/#

[27] Photoresistor – Wikipedia. [online]. Dostupné z: https://en.wikipedia.org/wiki/Photoresistor#/media/File:LDR_1480405_6_7_HDR_Enhancer_1.jpg

[28] What is a "Breadboard"?. Tangentsoft [online]. Copyright © 2011 [cit. 15.08.2022]. Dostupné z: https://tangentsoft.net/elec/breadboard.html

[29] Breadboard – Wikipedia. [online]. Dostupné z: https://en.wikipedia.org/wiki/Breadboard#/media/File:400_points_breadboard.jpg

[30] 5 mm Red LED.jpg - Wikimedia. [online]. [cit. 10.11.2021]. Dostupné z: https://upload.wikimedia.org/wikipedia/commons/c/c8/5mm_Red_LED.jpg

[31] Aktivní bzučák 5 V | laskakit.cz. LASKARDUINO.cz | by Makers for Makers [online]. Dostupné z: https://www.laskakit.cz/aktivni-bzucak-5v/

7 Přílohy

Součástí bakalářské práce je složka, která obsahuje:

1) Výkres schémat zapojení vývojových desek ESP32. (v PDF formátu)
2) Programy nahrané na jednotlivých vývojových deskách. (tyto programy se otevírají pomocí vývojového prostředí Arduino IDE)
3) Toky (Flows) vytvořené ve vývojovém prostředí Node-Red a exportovány do souboru JSON. (pro importování toků je potřeba mít naistalovánu paletu „node-red-dashboard“ v prostředí Node-red)

Oskenované zadání práce

UNIVERZITA HRADEC KRÁLOVÉ
Fakulta informatiky a managementu
Akademický rok: 2021/2022

Studijní program: Aplikovaná informatika
Forma studia: Prezenční
Obor/kombinace: Aplikovaná informatika (ai3-p)

Podklad pro zadání BAKALÁŘSKÉ práce studenta

Jméno a příjmení:	**Adam Zmeškal**
Osobní číslo:	**I1700161**
Adresa:	Cukrovarská 719, Poděbrady – Poděbrady II, 29001 Poděbrady 1, Česká republika
Téma práce:	Implementace bezdrátových modulů do systému domácí automatizace
Téma práce anglicky:	Implementation of wireless modules into a home automation system
Vedoucí práce:	Ing. Karel Mls, Ph.D. Katedra informačních technologií

Zásady pro vypracování:

Cílem práce je vytvoření jednoduchého a cenově dostupného automatizačního systému postaveného na modulech ESP32, které používají bezdrátovou technologii Wi-Fi.
1. Úvod
2. Cíle práce
3. ESP32
4. Node-red
5. MQTT
6. Wi-Fi
7. Návrh
8. Implementace
9. Závěr

Seznam doporučené literatury:

MODDABLE, S. D. K.; HODDIE, Peter; PRADER, Lizzie. IoT Development for ESP32 and ESP8266 with JavaScript.
SPANULESCU, Sever. *Esp32 Programming for the Internet of Things*. Lulu Press, Inc, 2018.

Podpis studenta: Datum:

Podpis vedoucího práce: Datum: