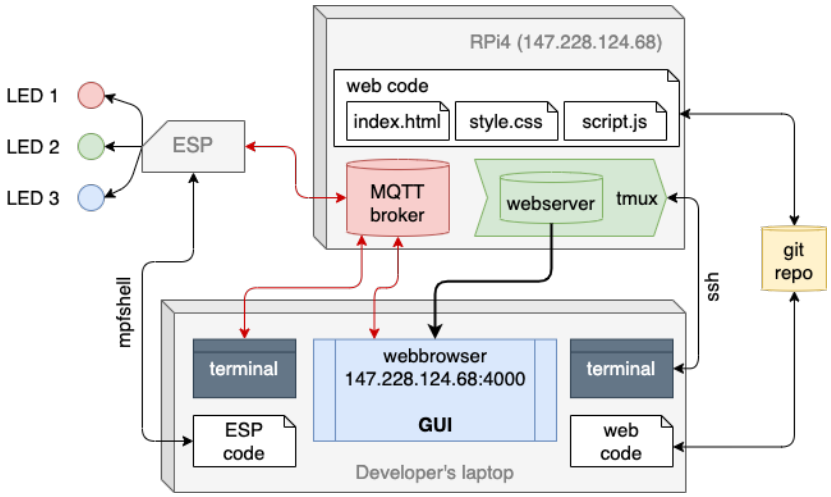
Projekt UIA4 - Smartlights

Západočeská univerzita v Plzni
Katedra Kybernetiky
Projekt Smartlights

Vojtěch Breník
letní semestr
24. června 2021

Obsah



Obrázek 1: Náhled na celý systém

1 Úvod

Cílem tohoto projektu bylo vytvořit vzdálené ovládání světel pomocí webové stránky s hlasovým vstupem a výstupem. Světla byla reprezentována svítívými diodami připojenými k mikrokontroléru ESP8266.

2 Metody

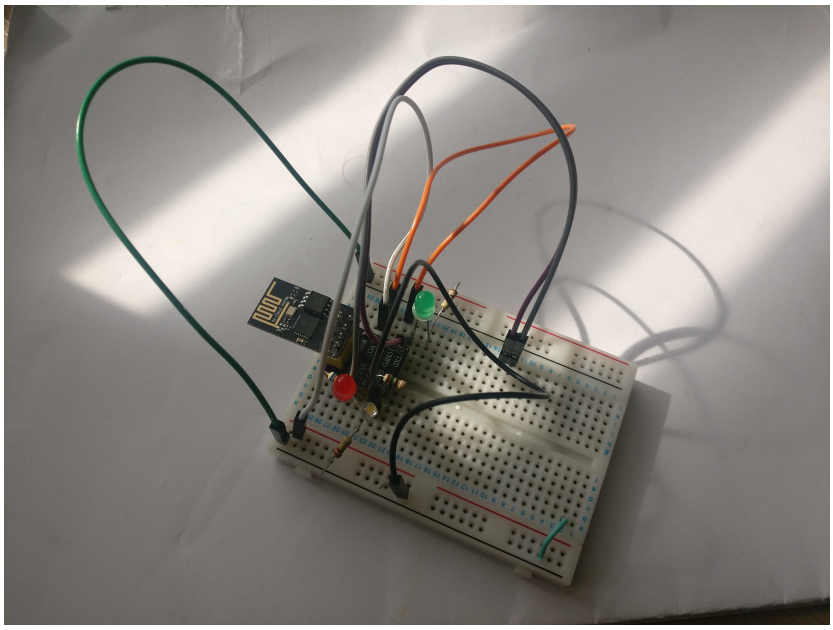
Řešení se skládá z několika součástí (obrázek 1). První z nich je jednodeskový počítač Raspberry Pi, na kterém běží webový server poskytující stránku klientskému počítači a MQTT broker starající se o předávání zpráv mezi jednotlivými komponentami. Další součástí je mikrokontrolér s čipem ESP8266, k tomu jsou připojené tři LEDky - bílá, červená, zelená, které reprezentují světla. Pomocí MQTT přijímá zprávy od uživatelů a předává aktuální stav světel. Běží na něm prostředí MicroPythonu a kód, který zajišťuje komunikaci a obsluhu příkazů. Webová stránka zobrazuje uživateli stav světel a dává mu možnost měnit jejich stav pomocí tlačítek nebo příkazy v přirozeném jazyce (např. „rozsviť bílé světlo“). Předávání zpráv probíhá prostřednictvím protokolu MQTT přes broker na Raspberry Pi, ke kterému se prohlížeč připojí pomocí websockets. Veškerou funkcionalitu webu zajišťuje JavaScript. O rozpoznávání hlasu a převod textu na řeč se stará SpeechCloud prostřednictvím JavaScriptové knihovny.

2.1 ESP8266

Modul ESP-01 sdružuje čip ESP8266, který implementuje standard IEEE 802.11 b/g/n a umožňuje tak připojení a komunikaci pomocí IP přes Wi-Fi, s několika málo externími komponentami jako flash paměť, pasivní součástky a anténa. K jeho výstupním pinům byly připojeny svítivé diody s ochrannými rezistory, které reprezentují světla. Při přivedení napájení čip zkontroluje stav pinů řídících režim startu, lze zavést program z paměti nebo pomocí UART rozhraní do paměti nový nahrát. Nejprve bylo do paměti nahráno běhové prostředí MicroPythonu, to rozdělí paměť na bootovací část a na část primitivního souborového systému, ze kterého je možné přímo za běhu spouštět kód v MicroPythonu. Po zavedení se spustí nejprve soubor `boot.py`, který zajistí správné nastavení síťových rozhraní a připojení k síti definované v konfiguračním souboru `config.py`. Poté následuje spuštění hlavního programu `main.py`. Ten se připojí k MQTT brokeru na Raspberry Pi, zaregistruje se k odběru zpráv v příslušném vlákně a vyčkává přijetí zprávy. Když se tak stane, je spuštěna metoda callback, která dekóduje přijatou zprávu ve formátu JSON a podle významu vykoná očekávanou činnost, o které pomocí MQTT a zprávy v JSONu informuje zpět. Důležité konstanty připojení jsou definované v konfiguračním souboru. Rozlišují se dva druhy zpráv: 1. je to zpráva s požadavkem na změnu stavu konkrétní LEDky, 2. pak požadavek na vrácení stavu všech diod. V obou případech je zpět odeslána zpráva o stavu, v 1. případě jen právě přepnuté, ve 2. všech. Nastane-li neočekávaná situace (chybně specifikovaná LEDka, neznámý příkaz, špatný formát zprávy), vrací se chybová zpráva s kódem a popisem chyby. Obvod byl sestaven na nepájivém kontaktním poli (obrázek 2), napájení poskytoval laboratorní zdroj s výstupem nastaveným na napětí 3,3 V. LEDky jsou připojené na piny 0, 1 a 2.

2.2 Webová stránka

Webová stránka (obrázek 3) se skládá ze 3 základních komponent - HTML kód (`index.html`), JavaScript (`script.js`) a kaskádové styly (`style.css`). V HTML je popsaná struktura stránky a její prvky, CSS pak upravuje jejich vzhled a kód v JavaScriptu zajišťuje funkčnost. Po jejím načtení klientským prohlížečem ze serveru běžícího na Raspberry Pi se odešle pomocí MQTT dotaz na stav světel. Pokud dorazí zpět zpráva v rámci časového limitu, vygenerují se podle ní tlačítka k ovládání vrácených světel, v opačném případě se předpokládá, že ESP pravděpodobně není připojeno a je o tomto zobrazena hláška. Stav světel se odráží ve způsobu zobrazení tlačítek - světla, která jsou rozsvícená, mají na webu okolo sebe záři a jsou probarvená, zhasnutá jsou černá. Pro přechod mezi stavy se využívá animace. V případě připojení více klientů najednou je předávání stavu vyřešeno z principu architektury řešení - vnitřní stav světla v prohlížeči se mění vždy, když je doručena zpráva o změně stavu světla, tedy i v případě, že požadavek odeslal jiný klient. Na webu existuje jednoduchý ladicí režim, který se přepíná pomocí tlačítka v patičce stránky a zobrazuje surové přijaté MQTT zprávy a pole pro testování textových příkazů,

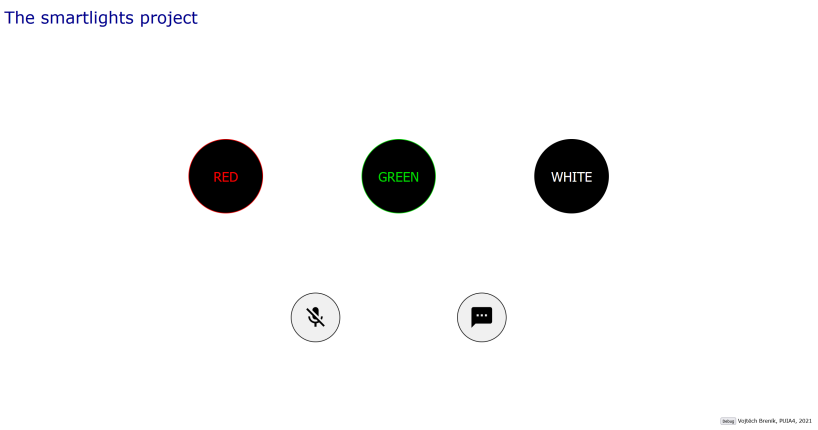


Obrázek 2: Obvod s ESP8266

které by měl vracet Speech Cloud. CSS obsahuje styly jednotlivých komponent a tlačítek pro přednastavené barvy světel (bílá, červená, zelená). Propojení se SpeechCloudem by měla zajišťovat externí knihovna. Webová stránka má také implementovaný jazykový model pro tuto úlohu, který by umožnil komunikaci pomocí přirozeného jazyka - například by tak mělo být možné vyslovit „rozsviť bílou“, „zhasni červené světlo“ atp., pomocí regulárních výrazů. Každý druh výrazu (zapnutí, vypnutí, přepnutí a dotaz na stav) má vlastní výraz, který se porovnává s přijatým řetězcem. V případě shody se ještě rozpozná barva, která je vždy zachycena v určité skupině výrazu.

2.3 Raspberry Pi

Na tomto jednodeskovém počítači běží pod Debianem MQTT broker, který se stará o výměnu zpráv mezi klientem (webová stránka) a ESP, které dle požadavků klientů na základě těchto zpráv vykonává příkazy. Také zajišťuje provoz webového serveru, předávajícího potřebné soubory klientskému prohlížeči. Ten se připojí k brokeru a přihlásí se k odběru zpráv. Pro účely tohoto projektu běželo RPi v budově NTIS a mělo veřejnou IP adresu, dalo se k němu tak přistupovat z Internetu. Přestože ESP bylo v jiné síti (která ale byla připojená do Internetu), bylo možné ho i tak ovládat. Pomocí programu tmux bylo udržováno sezení terminálu i po odpojení SSH terminálu od RPi a server tak běžel nepřetržitě. MQTT broker se instaluje jako služba, takže byl udržován v provozu pomocí systémových nástrojů.



Obrázek 3: Webová stránka

2.4 Vývoj

Veškerý kód byl během vývoje hostován v repozitáři služby GitHub. Postupovalo se od jednosušších funkčních součástí ke složitějším - nejprve se pomocí kódu ovládala jedna svítivá dioda a postupně byla přidávána další funkcionalita jako připojení k Wi-Fi a k brokeru, reakce na jednoduchou zprávu, strukturované zprávy a odpovědi atd. K nahrávání kódu do paměti ESP8266 byl použit nástroj mpfshell, který ale bohužel nefungoval přes IP a tak bylo nutné zařízení k vývojářskému PC připojit přes sériové rozhraní. Nástroj pak umožnil přenos souborů a přístup k interaktivní Python konzoli, kde se daly testovat jednotlivé příkazy. Repozitář byl pak využit nejen k verzování, ale i k distribuci kódu do RPi, kam se nasazovaly lokálně otestované funkční verze.

3 Výsledky

V současnosti je výstupem projektu funkční webová stránka s tlačítky pro ovládání světel a ovládání hlasového vstupu, ten se ale bohužel nepodařilo implementovat kvůli problémům se serverem na Raspberry Pi a následnému napojení na SpeechCloud. Stránka ale přijímá psané příkazy v přirozeném jazyce v rámci debug režimu, který se přepíná tlačítkem v patičce. Prototyp ovládaného zařízení je sestaven na nepájivém poli, v případě reálného použití by bylo vhodné ho přesunout na desku plošných spojů, doplnit o vlastní napájecí zdroj a LEDky nahradit výkonovým spínačem pro skutečná světla. Místo barev diod by se pak v příkazech daly využívat například názvy místností, ve kterých by byla světla fyzicky umístěna, to by vyžadovalo úpravu jazykového modelu pro SpeechCloud i jeho implementaci v JavaScriptu.
Přepínání stavu diod probíhá subjektivně s dostatečně malou odezvou, včetně reflexe stavu v prohlížeči. Totéž platí i o ovládání pomocí přirozeného jazyka, pokud by bylo funkční propojení na SpeechCloud, zanesla by se při ovládání

hlasem dodatečné zpoždění odezvy zejména vlivem přenosu zvuku na server a jeho zpracováním.

4 Závěr

Tento projekt ukazuje možnosti propojení několika součástí do funkčního celku pomocí protokolů MQTT a websockets, relativní jednoduchost tvorby zařízení pro IoT s čipem ESP8266 a nastiňuje nepříliš složitý přechod k reálnému využití. Ovládání pomocí jednoduchého uživatelského rozhraní nebo hlasových příkazů by zvládla naprostá většina uživatelů.

A Oživení

Kód pro ESP8266 je napsán v MicroPythonu, na ESP je tedy nutné nejprve nainstalovat jeho běhové prostředí. Poté použijeme program mpfshell, který umožňuje nahrání souborů `boot.py`, `main.py` a `config.py` pomocí příkazu `put <soubor>`. Konfigurace se musí upravit podle prostředí, ve kterém zařízení poběží. Je nutné upravit SSID a heslo WI-Fi sítě, IP adresu a port brokeru a přístupové údaje, dále také vlákno pro příjem příkazů a pro odesílání odpovědí zpět. Dle zapojení je možné upravit také konfiguraci diod. Web je připraven pouze pro provoz s těmito třemi barvami, je možné diodu pojmenovat jakkoli, ale v uživatelském rozhraní se bude ukazovat jako obecná (tj. bez barvy). Pro využití hlasového ovládání by také bylo nutné upravit jazykový model a jeho implementaci pomocí regexů - definice najdeme v souboru `script.js` ve funkci `parseStatement()` na ř. 158 - 162.
Na RPi s OS nainstalujeme ze systémového repozitáře MQTT broker (např `$ sudo apt install mosquitto`, web server z GitHubu společně se soubory webové stránky a například program tmux, který mimo jiné umožňuje udržení terminálové relace. Spustíme MQTT broker a pomocí tmux vytvoříme nový terminál, ve kterém spustíme webserver. V HTML opět upravíme konfiguraci MQTT. Poté by již měla být k dispozici webová stránka k ovládání světel.
Přivedeme napájení na ESP8266 - to se připojí k síti a k MQTT (pozor, ESP musí mít přístup na adresu RPi - Raspberry musí mít veřejnou nebo musí být ve stejné síti), poté načteme web a měla by se načíst tlačítka k ovládání světel dle konfigurace.