SŠPU
STŘEDNÍ ŠKOLA PRŮMYSLOVÁ
A UMĚLECKÁ, OPAVA

ZÁVĚREČNÁ STUDIJNÍ PRÁCE

dokumentace

Thermistor Thermometer

Jakub Komenda

Obor:	18-20-M/01 INFORMAČNÍ TECHNOLOGIE se zaměřením na počítačové sítě a programování
Třída:	IT4
Školní rok:	2020/2021

Poděkování

Rád bych poděkoval panu učiteli Ing. Petru Grussmannovi za cenné rady, za pomoc se součástkami a řešení některých problémů. Také bych rád poděkoval panu učiteli Mgr. Marceli Godovskému, který mi také hodně pomohl a dost věcí vysvětlil.

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové a umělecké škole v Opavě, Praskova 399/8.

V Opavě 31. 12. 2020

podpis autora práce

Anotace

Cílem mého závěrečného projektu bylo vytvořit funkční teploměr, který pomocí termistoru změří napětí a následně ho převede na stupně Celsia. Tuto hodnotu posílám skrz MQTT broker na vlastní Home Assistant v podobě grafu. Projekt tvoří zapojený hardware na nepájivém poli a Home Assistant nainstalovaný skrz Docker na ubuntu serveru. Komunikace probíhá pomocí Wi-Fi. Teplota se začne posílat do HA po zapsání příkazu do Visual Studio Code a vytvářet graf.

Klíčová slova

Teploměr; Termistor; Wi-Fi; Home Assistant; MQTT; ESP8266; Docker

OBSAH

ÚVOD

Cílem tohoto projektu bylo vytvořit takový menší, dalo by se říct i bezpečnostní, systém. Jeden z mých prvních nápadů na projekt byl sestavit a naprogramovat svůj vlastní dron. Přišlo mi to jako skvělý nápad, jelikož jsem to u minulých ročníků neviděl a chtěl jsem být originální. Hned jsem ovšem zjistil, že tenhle projekt by byl nad moje schopnosti a sám bych to nezvládl. Později přišel nápad na projekt s použitím ESP-32 CAM, součástky, která se u projektů také zas tak často nevidí, no a bylo rozhodnuto.

Ze začátku bylo hlavní myšlenkou propojit ESP-32 CAM a senzor pohybu AM312 a následně přes aplikaci Telegram pomocí bota ovládat celý projekt pomocí Wi-Fi. Poté však přišel nápad s přidáním senzoru BME280, který bylo velice jednoduché naprogramovat, díky knihovny, o které se ještě zmíním v dalších částech dokumentace. „Bezpečnostní systém“, chcete-li, je naprogramován v jazyce Arduino, jedná se o kombinaci jazyků C a C++.

Ve své dokumentaci zmiňuji použité technologie a blíže popisuji princip fungování bezpečnostního systému i ovládání přes mobilní aplikaci. Zmiňuji se o problémech, se kterými jsem se setkal při vývoji bezpečnostního systému, pokračuji popisem technologií nezbytných k jeho výrobě i k jeho současné funkčnosti a rozebírám, jak funguje nejen samotné bezpečnostní zařízení, ale i mobilní aplikace potřebná k jeho vzdálenému ovládání. V další části popisuji jednotlivé úkony obou částí systému.

1 VÝROBA A VÝVOJ TEPLOMĚRU

Prvním krokem bylo objednání součástek na internetu a následné sestavení, které není vůbec těžké, jelikož zapojujete pouze 3 součástky. Zapojení hardwaru nebylo moc složité, jelikož stačilo zapojit 3 drátky. Po zapojení by to bylo k hardwaru asi všechno, zbytek projektu se skládá ze softwarových částí jako programováví teploměru a Home Assistantu.

Vzhledem k tomu, že jsem s potřebnými softwarovými aplikacemi ještě nesetkal, musel jsem si zjistit co přesně dělají a jak fungují. Poté jsem nainstaloval ubuntu server, kde jsem musel nainstalovat Docker Engine bez kterého bych vlastně nezprovoznil celý Home Assistant a MQTT systém. Následovalo naistalování a nastavení už řečeného Home Assistantu a MQTT. MQTT jsem si prvně zkoušel používat přes terminály a až poté do Home Assistantu.

Kod jsem upravoval a kontroloval postupně. Prvně jsem musel napsat kod na změření teploty. Měření teploty spočívá v tom, že termistor neměří ze začátku teplotu ale napětí a vy to musíte převést na teplotu. Vzoreček na tenhle výpočet jsem si našel na internetu. Poté následuje připojení ESP8266 na WiFi, MQTT broker a zaslání změřené teploty na Home Assistant, kde se to vykresluje do grafu. Do budoucna bych chtěl zařízení přesunout na PCB desku a vložil do krabičky, abych ho mohl mít venku nebo v jinačích částech domu. Také mě napadlo v budoucnu spojit teploměr s thermostatem a fungovalo by to tak, že by se v domě topilo do doby než by přesáhl zadanou teplotu a následně by thermostat vypl. Dále se dají připojit další zařízení na měření hodnot (např. tlak, vlhkost).

2 PRINCIP FUNGOVÁNÍ TEPLOMĚRU

Během vývoje se fungování přístroje několikrát změnilo v závislosti na nápadech.

Systém funguje jednoduše, když se teploměr zapojí do zdroje tak začne měřit napětí, které následně pomocí vzorce vypočítá na teplotu. Dále se připojí na WiFi a broker. MQTT pošle hodnoty na Home Assistant do grafu nebo jiného zobrazení, které si můžete přednastavit. Uživatel, který je připojen na síti, kde Home Assistant běží, si může prohlídnout graf teplot, které, teploměr změřil. Uživatel také nepotřebuje do procesu nějak zasahovat.

3 VYUŽITÉ TECHNOLOGIE

3.1 Hardware

3.1.1 Seznam součástek

- ESP-8266 D1 Mini
- Termistor NTC Vishay 10k
- Rezistor 10k
- Nepájivé pole se 400 kontakty

3.1.2 ESP-8266 D1 Mini

Základ projektu tvoří vývojová deska ESP-8266. ESP-8266 je nízkonákladový mikročip Wi-Fi s plnou schopností protokolu TCP/IP a schopností mikrokontroléru vyráběný společností Espressif Systems. Na čipu se náchazí 2x8 pinů. 11 pinů je digitálních, napětí výstup 5V a 3,3V a analogový vstup.

Parametry

- Pracovní napětí: 3.3V

-Frekvence procesoru: 80MHz/160MHz

-Micro USB konektor

-Flash: 4MB

Obrázek č. 1 ESP-8266

Obrázek č. 2 Piny ESP-8266

3.1.3 Termistor NTC Vishay 10k

Termistor NTC pro měření teploty

Parametry

- Odpor: 10k Ohm
- Výkon P: 500mW
- Rozteč: 2.54mm
- Teplotní rozsah: - 40°C až 125°C
- Výrobce: Vishay

3.1.4 Rezistor 10k

je pasivní elektrotechnická
součástka projevující se v elektrickém
obvodu v ideálním případě jedinou vlastností
– elektrickým odporem (jednotka Ohm,
značka Ω).

Parametry

- Odpor: 10k Ohm
- Výkon: 250mW
- Tolerance rezistence: ±0.1%

3.2 Software

3.2.1 Oracle VM VirtualBox 6.1.16

Oracle VM VirtualBox je multiplatformní virtualizační nástroj distribuovaný jak pro Linux/Unix tak pro Windows a Mac OS. Ten jsem potřeboval pro instalování ubuntu serveru ve kterém běží Docker Engine, Home Assistant a MQTT.

3.2.2 Docker Engine 20.10.1

Docker je v informatice název pro otevřený software (open source projekt), jehož cílem je poskytnout jednotné rozhraní pro izolaci aplikací do kontejnerů v prostředí macOS, Linuxu i Windows („odlehčená virtualizace“). Docker Engine jsem instaloval z důvodu Home Assistant a poté MQTT. Předtím než jsem měl Docker tak jsem spouštěl Home Assistant přes VDI a následně jsem zjistil problém, že mi nepůjde nainstalovat MQTT, proto jsem použil Docker.

3.2.3 Eclipse Mosquitto 2.0.4

Eclipse Mosquitto je zprostředkovatel zpráv s otevřeným zdrojovým kodem, který implementuje protokol MQTT verze 5.0, 3.1.1 a 3.1. MQTT je potřeba abych mohl posílat změřenou teplotu na Home Assistant.

3.2.4 Home Assistant

Home Assistant je bezplatný a open-source software pro automatizaci domácnosti navržený jako centrální systém řízení domácí automatizace pro ovládání technologie inteligentních domů. Pomoci Home Assistantu můžu vidět hodnoty z teploměru. Určitě bych do budoucna chtěl s Home Assistantem více pracovat a vytvořit si chytrou domácnost.

3.2.5 Visual Studio Code

Visual Studio Code je bezplatný editor zdrojového kodu vytvořenený společností Microsoft pro Windows, Linux a macOS. Vybíral jsem mezi Arduino IDE a Visual Studio Code, ale díky tomu, že jsme s VS Codem v minulosti více pracovali a měl jsem s ním více zkušeností tak jsem si ho vybral jako editor kodu.

4 ZPŮSOBY ŘEŠENÍ A VYUŽITÉ TECHNOLOGIE

4.1 Hardwarové zařízení

4.1.1 Základní funkce jazyka Arduino

Jazyk Arduino obsahuje dvě základní funkce. Funkce setup() a loop(). Až na drobné úpravy je velmi podobný jazyku C a C++. Funkce setup(), slouží k inicializaci proměnných a nastavení potřebných hodnot. Je zavolá-na při spuštění nebo po restartu. Funkce loop() je hlavní funkce programu, která se stále opakuje.

4.1.2 Připojení k WiFi

Pro snadnější připojení k WiFi jsem využil knihovnu *ESP8266WiFi.h*. Díky této knihovně se zařízení samo pokusí o připojení k přednastavené WiFi, jméno a heslo od WiFi je nutno předem zapsat do zdrojového kódu.

```
void setup_wifi() {

  delay(10);
  // Připojení k WiFi
  Serial.println();
  Serial.print("Připojování k ");
  Serial.println(ssid);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

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

  randomSeed(micros());

  Serial.println("");
  Serial.println("WiFi připojena");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}
```

Obrázek č.3 Připojení k WiFi

4.1.3 Připojení k MQTT brokeru

Abych dostal hodnoty do Home Assistantu, tak se musím připojit na vytvořený MQTT broker. K tomu mi slouží funkce *reconnect,* která se po vyvolání snaží připojit k přednastavené IP adrese od vytvořeného MQTT brokeru. Pokud se připojení nepovede, funkce se o to pokusí znova po 5 sekundách a tak se to opakuje dokud se nepřipojí.

```
void reconnect() {
  // Opakování dokud se nepřipojí
  while (!client.connected()) {
    Serial.print("Zkouška připojení k MQTT...");
    // Vytvoření náhodného ID klienta
    String clientId = "ESP8266Client-";
    clientId += String(random(0xffff), HEX);
    // Pokus o připojení
    if (client.connect(clientId.c_str())) {
      Serial.println("připojeno");
      client.subscribe("inTopic");
    } else {
      Serial.print("nepřipojeno, rc=");
      Serial.print(client.state());
      Serial.println(" zkus znova za 5 sekund");
      // Čekej 5 sekund na opakování
      delay(5000);
    }
  }
}
```

Obrázek č.4 Připojení k MQTT

4.1.4 Měření teploty

Samotný termistor nedokáže změřit teplotu ale pouze napětí. K tomu slouží Steinhart-Hartova rovnice termistoru, která vypočítá pomoci parametrů rovnice termitoru samotnou teplotu. Problem je v tom, že se ale teplota vypočítá v Kelvinech a proto aby mohla být ve stupních, tak se musí od vypočítané hodnoty odečíst 273.15.

```
const double A = 0.001129148;
const double B = 0.000234125;
const double C = 0.0000000876741;
```

Obrázek č.5 Parametry rovnice termistoru

$$\frac{1}{T_1} = A + B\ln(R_1) + C[\ln(R_1)]^3$$
$$\frac{1}{T_2} = A + B\ln(R_2) + C[\ln(R_2)]^3$$
$$\frac{1}{T_3} = A + B\ln(R_3) + C[\ln(R_3)]^3$$

Obrázek č.6 Steinhart-Hartova rovnice

```
double Vout, Rth, teplota, adc_value;

adc_value = analogRead(A0);
Vout = (adc_value * VCC) / adc_resolution;
Rth = (VCC * R2 / Vout) - R2;

teplota = (1 / (A + (B * log(Rth)) + (C * pow((log(Rth)),3))));

teplota = teplota - 273.15;
```

Obrázek č.7 Samotný výpočet teplo

4.1.5 Posílání hodnot na Home Assistant

Pomocí knihovny PubSubClient.h mohu jednoduše publikovat zprávy se serverem, který podporuje MQTT. V kodu si nastavíte topic, do kterého budete zasílat hodnoty a na MQTT serveru budete tento topic poslouchat a hodnoty z něho brát.

```
unsigned long now = millis();
if (now - lastMsg > 2000) {
  lastMsg = now;
  ++value;
  snprintf (msg, MSG_BUFFER_SIZE, " %lf", teplota);
  Serial.println(teplota);
  client.publish("/testtopic/teplota", msg);
}
```

Obrázek č.8 Posílání teploty na mqtt

V mojem případě posílám hodnoty z funkce *teplota* na topic */testtopic/teplota.*

4.2 Webová aplikace

Jak jsem už psal, instalace probíhá v dockeru na ubuntu serveru. Po instalaci se vytvoří konfigurační soubor *configuration.yaml* ve kterém je potřeba aby se nám teplota vypisovala do grafu, nakonfigurovat sensor, který si poté vyberem při tvorbě karty grafu. V konfiguraci se nastavuje platforma, jméno, topic a jednotky ve kterých měříme teplotu.

```
sensor:
  - platform: mqtt
    name: "Měření teploty"
    state_topic: "/testtopic/teplota"
    unit_of_measurement: 'C'
#    value_template: "{{ value_json.RSSI }}"
```

Obrázek č.9 Konfigurace grafu

5 VÝSLEDEK ŘEŠENÍ

S výsledkem jsem velice spokojen a s momentální funkčností taky, po několikaddením testování funguje, jak má a splňuje mé očekávání. Během vytváření mě napadalo mnoho vylepšení mého zařízení. Do budoucna mám rozhodně hodně plánů s mojím teploměrem jako třeba ho dát do vytvořené krabičky, aby byl přenosný a mohl bych měřit teplotu ve všech místnostech. Také by bylo dobré napájet teploměr baterií a nemuset ho pořád zapojovat do počítače, což má svoje výhody. Možné by bylo taky propojení teploměru s thermostatem aby se zapl nebo vypl podle naměřené teploty a takhle udržoval stálou teplotu. Rozhodně mám v plánu s tímhle pokračovat a budovat si chytrou domácnost. Myslím si, že tenhle projekt je dobrý pro opdpich k takové záležitosti jako je tvorba chytré domácnosti.

ZÁVĚR

Cílem projektu bylo vytvořit funkční teploměr, který bude posílat teplotu na Home Assistant a já se tak mohl sledovat teplotu mého pokoje. Všechny cíle, které jsem měl jsem splnil, zařízení i aplikace fungují jak mají a nejsou s teploměrem žádné problémy. Můj pokoj je nejchladnější místnost v domě a takhle se můžu já a rodiče dívat na aktuální teplotu pokoje a rozhodnout se, jestli zatopit nebo ne. V budoucnu svůj výtvor vylepším o spojení s thermostatem a ovládání teploty na dálku, ale to je jenom jeden z mála vylepšení, které se můžou provést. S chytrou domácností jsem neměl v minulosti žádné zkušenosti a tohle je si myslím dobrý začátek.

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

[1] *ESP8266 MQTT* [online].[cit.2021-1-9]. Dostupné z: https://iotdesignpro.com/projects/how-to-connect-esp8266-with-mqtt

[2] *Thermistor* [online].[cit.2021-1-9]. Dostupné z https://navody.dratek.cz/arduino-projekty/mereni-teploty-s-termistorem.html

[3] *PubSubClient* [online].[cit.2021-1-9]. Dostupné z https://github.com/knolleary/pubsubclient

[4] *Docker HA* [online].[cit.2021-1-9]. Dostupné z https://www.home-assistant.io/docs/installation/docker/

[5] *Docker MQTT* [online].[cit.2021-1-9]. Dostupné z https://hub.docker.com/_/eclipse-mosquitto

[6] Sensor [online].[cit.2021-1-9]. Dostupné z https://www.home-assistant.io/integrations/template/