SŠPU
STŘEDNÍ ŠKOLA PRŮMYSLOVÁ
A UMĚLECKÁ, OPAVA

ZÁVĚREČNÁ STUDIJNÍ PRÁCE
dokumentace

ESP-32 CAM Motion Detection

Vojtěch Dohnal

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

Poděkování

Především děkuji panu učiteli Ing. Petru Grussmannovi za cenné rady, za pomoc se součástkami a také za neustálou motivaci do projektu a panu učiteli Ing. Jiřímu Miekischovi za pomoc s pájením.

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 projektu bylo vytvořit funkční senzor pohybu s kamerou, přičemž vždy, když senzor zaznamená pohyb, dostanu upozornění a také fotografii v reálném čase. Součástí projektu je také senzor, pomocí kterého si lze zjistit teplotu a vlhkost. Systém tvoří dvě části: hardware, který je umístěn na nepájivém poli a bot v mobilní aplikaci Telegram. Komunikace mezi nimi probíhá samozřejmě bezdrátově, pomocí Wi-Fi. Pro zajištění bezpečnosti má bot svůj vlastní token, díky kterému se spojí s ESP-32 CAM, a až pak může začít probíhání komunikace. Administrátor může také aktivovat či deaktivovat senzor pohybu podle potřeby. Také lze poslat příkaz, který si vyžádá aktuální fotografii z ESP-32 CAM a pošle jej do mobilní aplikace nezávazně na senzoru pohybu.

Klíčová slova

Bezpečnost; Arduino; bezdrátová komunikace; Wi-Fi; Telegram; senzor pohybu; ESP-32 CAM

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 SYSTÉMU

Prvním krokem bylo vytvoření schématu za pomocí programu KiCAD, kde jsem si odzkoušel rozvržení součástek na nepájivém poli a následovalo zajištění všech potřebných součástek. Komunikace mezi zařízením a mobilní aplikací je zařízena pomocí součástky ESP-32 CAM. Rozhodl jsem se pro model Ai-Thinker, jelikož už má vestavěný Wi-Fi modul a disponuje nízkoenergetickým 32bitovým procesorem.

Vzhledem k tomu, že jsem s většinou nakoupených součástek nikdy dříve nepracoval, rozhodl jsem se zapojovat jednu součástku po druhé do nepájivého pole a zkoušel jsem její funkčnost pomocí jednoduchých příkladů z Arduino IDE. Následovalo programování zařízení a nastavování bota v mobilní aplikaci Telegram, aby byla zajištěna bezproblémová komunikace mezi ním a zařízením.

Kód jsem postupně upravoval a ladil s tím, jak přicházely nápady na přidání nových funkcí či součástek. Následovalo sestavení zařízení na nepájivém poli. Rozhodl jsem se pro nepájivé pole proto, aby bylo možné ESP jednoduše přeprogramovat, popřípadě zařízení upravit. Do budoucna bych chtěl určitě zařízení přesunout na PCB desku, kde se již všechny součástky spájí a následně se zařízení vloží do krabičky, aby bylo možné ho nechat i třeba venku.

2 PRINCIP FUNGOVÁNÍ SYSTÉMU

Během vývoje jsem princip fungování několikrát změnil v závislosti na postupně přicházejících nápadech, které přišly až po testování v praxi.

Systém funguje následovně: pokud je funkce senzoru pohybu zapnutá a senzor zachytí pohyb, vyvolá se funkce, která vyfotí fotografii pomocí ESP-32 CAM a poté odešle požadavek pro odeslání fotografie do Telegram bota. Pokud souhlasí token, je nejprve odeslána zpráva, která informuje o detekci pohybu a následně samotná fotografie. Toto vše probíhá automaticky, bez potřeby zásahu uživatele.

Uživatel si však může fotografii vyžádat nezávisle na senzoru pohybu pomocí příkazu s mobilní aplikace. Požadavek se přes Wi-Fi pošle do zařízení, zde ho zpracuje funkce pro přijetí nového požadavku a následně je vyvolána funkce, která vyfotí fotografii a odešle požadavek pro odeslání fotografie do Telegram bota.

Dále je možné si vyžádat údaje o teplotě a vlhkosti ze senzoru BME280. Funkce pro přijetí nového požadavku opět požadavek nejprve zpracuje, poté si další funkce vyžádá hodnoty ze senzoru a vrátí je jako proměnnou řetězce. Text se pak pošle do aplikace.

V neposlední řadě si uživatel může vypnout/zapnout funkci senzoru pohybu podle libosti. Když je funkce vypnutá, senzor bude stále zaznamenávat detekci pohybu, uživatel však v aplikaci dostane pouze upozornění o vypnuté detekci pohybu a nebude dostávat žádné fotografie či další upozornění.

3 VYUŽITÉ TECHNOLOGIE

3.1 Hardware

3.1.1 Seznam součástek

- ESP-32 CAM Ai-Thinker
- Mini AM312 PIR Motion Sensor
- BME280 Modul
- Nepájivé pole se 400 kontakty
- Arduino Nano

3.1.2 ESP-32 CAM Ai-Thinker

Základ projektu tvoří vývojová deska ESP-32 CAM (obrázek č. 1) s kamerovým modulem OV2640. Deska disponuje integrovanou pamětí 520kB SRAM a externí 4M PSRAM. Pracuje při frekvenci až 240MHz a obsahuje Wi-Fi i Bluetooth modul, který jsem ovšem nevyužil. Příležitostně se na desce nachází i LEDka, kterou si uživatel může zapnout/vypnout podle libosti.

Obrázek č. 1 ESP-32 CAM

Obrázek č. 2 Piny ESP-32 CAM

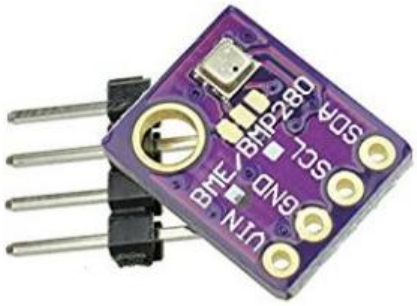
3.1.3 Mini AM312 PIR Motion Sensor

Pohybové čidlo AM312 je kompatibilní s deskami Arduino/Genuino, ESP-32 CAM a jinými vývojovými deskami. Výhodou čidla je malá vzdálenost, detekční vzdálenost je 3-5 metrů a detekční úhel < 100°. Pracuje při teplotě -20 °C ~ 60 °C.

Obrázek č. 3 AM312 PIR Motion Sensor

3.1.4 BME280 Modul

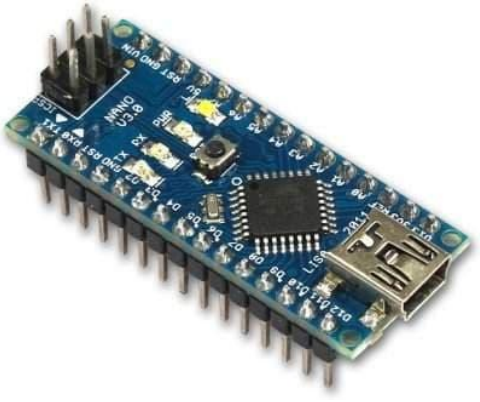
Precizní senzor, měřící teplotu od -40 °C až do 85 °C a vlhkost 0% až 100%. Používá I2C sběrnici (až 3.4 MHz) a jeho přesnost je + 1 °C u teploty a + 3% u vlhkosti. BME280 dokáže měřit i tlak, ten jsem však vynechal.



Obrázek č. 4 BME280

3.1.5 Arduino Nano

Arduino Nano je jednodeskový počítač založený na mikrokontroléru ATmega328. Obsahuje 30 pinů, ze kterých jsem použil jen piny TX1 a RX0 z důvodu nahrání kódu do ESP-32 CAM. Rozhodl jsem se pro Arduino Nano, jelikož jsem měl problémy se sehnáním FTDI programmeru. V době psaní dokumentace používám i piny 5V a GND pro napájení všech sou-



Obrázek č. 5 Arduino Nano

částek z počítače, plánuji však dokoupit baterii a napájet přes ni, aby bylo zařízení přenosné.

3.2 Software

3.2.1 Arduino IDE ver. 1.8.13

Arduino IDE je Open-Source platforma pro snadný vývoj elektronických programovatelných zařízení. Naprogramovat v něm lze všechny desky Arduino. Software je intuitivní a jednoduchý na ovládání a má velké množství příznivců s velkou komunitou, kde se případné problémy řeší většinou snadno. Arduino IDE jsem použil na otestování všech součástek, abych věděl jestli jsou funkční.

3.2.2 Visual Studio Code 1.52.1

Editor zdrojového kódu Visual Studio Code jsem použil právě proto, jelikož s ním pracuji již od prvního ročníku, a tak je mi nejbližší. Zvolil jsem jej i pro jeho podporu množství programovacích jazyků.

3.2.3 KiCAD

KiCad je multiplatformní Open-Source profesionální software pro návrh a projektování plošných spojů. Kromě samotného návrhu schémat zapojení elektrických obvodů a jejich převodu do plošných spojů zahrnuje například také nástroje pro sestavení kusovníku a export do formátu Gerber.

4 ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY

4.1 Hardwarové zařízení

4.1.1 Základní funkce jazyka Arduino

Jazyk Arduino obsahuje dvě základní funkce. Funkce setup() a loop(). Funkce setup(), slouží k inicializaci proměnných a nastavení potřebných hodnot. Je zavolána při spuštění nebo po restartu. Funkce loop() je hlavní funkce programu, která se stále opakuje.

4.1.2 Komunikace s aplikací

Pro vzdálené ovládání zařízení je nutná jeho komunikace s aplikací. Tu zajišťuji pomocí HTTP POST požadavků z ESP-32 CAM do aplikace.

Odesílání hodnot

Odesílání dat do aplikace mi zajišťuje funkce sendPhotoTelegram(), ta se připojí do aplikace a pomocí http POST požadavku tato data odešle (obrázek č. 6).

```
clientTCP.println("POST /bot" + BOTtoken + "/sendPhoto HTTP/1.1");
clientTCP.println("Host: " + String(myDomain));
clientTCP.println("Content-Length: " + String(totalLen));
clientTCP.println("Content-Type: multipart/form-data");
clientTCP.println();
clientTCP.print(head);
```

Obrázek č. 6 HTTP POST

4.1.3 Připojení k WiFi

Pro snadnější a také bezpečnější připojení k WiFi jsem využil knihovny WiFi.h a WiFiClientSecure.h. Díky nim se zařízení samo pokusí o připojení k dané WiFi, jejíž jméno a heslo je nutno předem zapsat do zdrojového kódu.

```
WiFi.mode(WIFI_STA);
Serial.println();
Serial.print("Connecting to ");
Serial.println(ssid);
WiFi.begin(ssid, password);
while (WiFi.status() != WL_CONNECTED) {
  Serial.print(".");
  delay(500);
}
Serial.println();
Serial.print("ESP32-CAM IP Address: ");
Serial.println(WiFi.localIP());
```

Obrázek č. 7 Připojení ESP-32 CAM k WiFi

4.1.4 Ověření uživatele

Každá zpráva z aplikace přichází s unikátním chat_id, které je následně porovnáváno s mým chat_Id, a pokud se id neshodují, program nevykoná příkaz a pošle zprávu že jste neoprávněný uživatel.

```
String chat_id = String(bot.messages[i].chat_id);
if (chat_id != chatId){
  bot.sendMessage(chat_id, "Unauthorized user", "");
  continue;
}
```

Obrázek č. 8 Ověření uživatele

4.1.5 Vyfocení fotografie a odeslání do aplikace

Funkce sendPhotoTelegram(), která vyfotí fotografii pomocí ESP-32 CAM (obrázek č. 9) a pošle HTTP POST požadavek pro odeslání fotografie do aplikace se vyvolá jen tehdy, když je to vyžádáno od uživatele pomocí příkazu z aplikace, nebo když je zachycen pohyb funkcí motionDetected.

```
camera_fb_t * fb = NULL;
fb = esp_camera_fb_get();
if(!fb) {
  Serial.println("Camera capture failed");
  delay(1000);
  ESP.restart();
  return "Camera capture failed";
}
```

Obrázek č. 9 Vyfocení fotografie

4.1.6 Detekce pohybu

Pokud funkce motionDetected() pomocí senzoru zaznamená pohyb a je manuálně zapnutá detekce pohybu, nejprve se odešle zpráva uživateli do aplikace o zaznamenání pohybu a následně se vyvolá funkce sendPhotoTelegram() (obrázek č. 10).

```
if (motionDetected && motionDetectEnable)
{
  bot.sendMessage(chatId, "Zaznamenán pohyb!", "");
  Serial.println("Motion Detected");
  sendPhotoTelegram();
  motionDetected = false;
}
```

Obrázek č. 10 Detekce pohybu

4.1.7 Knihovna SparkFunBME280.h

Pomocí knihovny SparkFunBME280.h si vyžádám teplotu pomocí funkce readTempC() a vlhkost pomocí funkce readFloatHumidity() (obrázek č. 11). Hodnoty se následně uloží do proměnných typu float následně se odešlou do aplikace.

```
String getReadings(){
  float temperature, humidity;
  temperature = bme.readTempC();
  //temperature = bme.readTempF();
  humidity = bme.readFloatHumidity();
```

Obrázek č. 11 Vyžádání si hodnot BME280

4.2 Mobilní aplikace

4.2.1 Kontrola nových zpráv

Zařízení si samo kontroluje, jestli uživatel z aplikace odešle novou zprávu (obrázek č. 12), a to každou vteřinu. Pokud se zaznamená nová zpráva, vyvolá se funkce handleNewMessages() (obrázek č. 13), která rozhodne, o jakou zprávu se jedná a co se má dále udělat.

```
if (millis() > lastTimeBotRan + botRequestDelay){
  int numNewMessages = bot.getUpdates(bot.last_message_received + 1);
  while (numNewMessages){
    Serial.println("got response");
    handleNewMessages(numNewMessages);
    numNewMessages = bot.getUpdates(bot.last_message_received + 1);
  }
  lastTimeBotRan = millis();
}
```

Obrázek č. 12 Kontrola nových zpráv

```
while (numNewMessages){
  Serial.println("got response");
  handleNewMessages(numNewMessages);
  numNewMessages = bot.getUpdates(bot.last_message_received + 1);
}
```

Obrázek č. 13 Funkce handleNewMessages()

4.2.2 Ovládání aplikace

Ovládání aplikace je velice jednoduché a intuitivní, pomocí krátkých příkazů. Přidal jsem příkaz /start, který vypíše všechny ostatní příkazy, které zařízení umí společně s krátkým popisem jejich funkce.

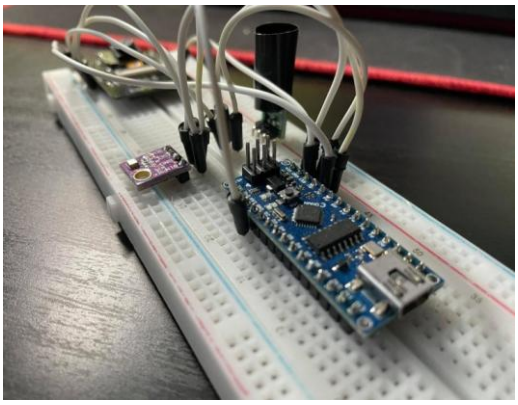
```
if (text == "/start")
{
  String welcome = "Vítej, jsem ESP32-CAM Telegram bot.\n";
  welcome += "Zde jsou příkazy, které ovládám.\n\n";
  welcome += "/photo : vyfotím fotku\n";
  welcome += "/flash : zapnu LED světlo\n";
  welcome += "/mon : zapnu detekci pohybu\n";
  welcome += "/moff : vypnu detekci pohybu\n";
  welcome += "/teplota : vypíšu teplotu a vlhkost\n\n";
  welcome += "Když zaznamenám pohyb, pošlu ti fotku.\n";
  bot.sendMessage(chatId, welcome, "Markdown");
}
```

Obrázek č. 14 Ovládání aplikace

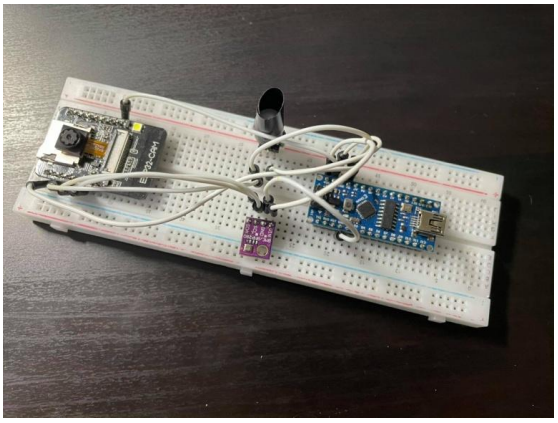
5 VÝSLEDKY ŘEŠENÍ

5.1 Podoba hardwarového zařízeni

Zařízení je momentálně plně funkční a s výsledkem jsem velice spokojen, po několikadenním testování splňuje mé počáteční očekávání a všechny funkce fungují bez problémů. V průběhu vývoje zařízení mě napadlo nespočet změn, které jsem buď využil, nebo ne. Proto je zařízení usazené na nepájivém poli. Do budoucna mám v plánu si zajistit nějakou krabičku, například vytištěnou na 3D tiskárně, do které zařízení umístím. Dalším možným vylepšením je napájet zařízení baterií schovanou v krabičce a to je prospěšné kvůli hned dvěma věcem. Zaprvé je zařízení přenosné, a to je u takového zařízení takměř nutností a zadruhé, když je zařízení napájeno z počítače kabelem, stačí kabel z počítače vytrhnout a zařízení je mimo provoz, proto je vhodnější mít uvnitř krabičky baterii a nebude tak jednoduché zařízení odpojit.



Obrázek č. 15 Detailní záběr na zařízení



Obrázek č. 16 Finální zařízení

ZÁVĚR

Cílem mého projektu bylo vytvořit funkční menší bezpečnostní systém s ovládáním na dálku prostřednictvím mobilní aplikace. Všechny mé cíle byly splněny, zařízení i aplikace fungují bez problému a spolehlivě. Při vývoji mě nicméně napadl nespočet dalších nápadů, které by zařízení rozhodně vylepšily, jako například napájení z baterie, nebo umístění zařízení do krabičky. Ovšem to podle mě nejdůležitější vylepšení, na které mi už bohužel nezbyl čas, by bylo spojení zařízení s chytrou žárovkou, která by se po určité večerní hodině sama rozsvítila těsně před vyfocením fotografie. Na ESP-32 CAM je umístěna malá LED dioda, ta ovšem na dostatečné osvětlení osoby, aby ji šlo identifikovat, nestačí. Samotné zařízení bych dále rád umístil na desku plošných spojů místo nepájivého pole, jelikož je zařízení poměrně křehké a sem tam se některý drátek vypojí. Těmto vylepšením se budu po odevzdání projektu věnovat.

SEZNAM POUŽITÝCH INFORMAČNÍCH ZDROJŮ

[1] *ArduinoJson* [online].[cit.2020-12-28]. Dostupné z: https://github.com/bblanchon/ArduinoJson

[2] *SparkFunBME280* [online].[cit.2020-12-28]. Dostupné z: https://github.com/sparkfun/SparkFun_BME280_Arduino_Library/blob/master/src/SparkFunBME280.h

[3] *UniversalTelegramBot* [online].[cit.2020-12-28]. Dostupné z: https://github.com/witnessmenow/Universal-Arduino-Telegram-Bot/blob/master/src/UniversalTelegramBot.h

[4] *Create new Telegram bot* [online].[cit.2020-12-28]. Dostupné z: https://docs.microsoft.com/en-us/azure/bot-service/bot-service-channel-connect-telegram?view=azure-bot-service-4.0

[5] *KiCAD first schema* [online].[cit.2020-12-28]. Dostupné z: https://vyvoj.hw.cz/teorie-a-praxe/navrh-elektroniky-v-programu-kicad-1-dil.html

[6] *ESP-32 CAM pins* [online].[cit.2020-12-28]. Dostupné z: https://randomnerdtutorials.com/esp32-cam-ai-thinker-pinout/

[7] *HTTP POST request* [online].[cit.2020-12-28]. Dostupné z: https://developer.mozilla.org/en-US/docs/Web/HTTP/Methods/POST

SEZNAM OBRÁZKŮ