Řízení a sběr dat z hardwarového simulačního modelu pomocí RaspberryPi a MQTT protokolu

Matouš Hucl, Miroslav Malaga, Zdeněk Ulrych

Západočeská univerzita v Plzni, Fakulta strojní, Katedra průmyslového inženýrství a managementu
Univerzitní 8, Plzeň, 306 14, Česká republika
mhucl@kpv.zcu.cz
malaga@kpv.zcu.cz
ulrychz@kpv.zcu.cz

Anotace: Příspěvek se zabývá možnostmi využití jednodeskového počítače RPi při řízení a sběru dat ze zkušebního modelu Fishertechnik. Fishertechnik je polytechnická stavebnice, skládající se z mechanických součástí (převodovky, ozubená kola apod.) a z výkonových součástí (elektromotory, ventily, světelné moduly atd.). Z této stavebnice byl vytvořen zkušební model, který simuluje výrobní linku. Výrobní linka reprezentuje pracoviště rentgenu, selekci zmetků a balení materiálu. Stavebnice Fishertechnik nabízí i řídící jednotku Fishertechnik Controller TXT4.0 a vývojové grafické prostředí ROBO Pro Coding. Tento článek se zabývá možnostmi využití tohoto hardwaru a softwaru. Zároveň demonstruje využití přímého napojení mechanických komponent stavebnice Fishertechnik na GPIO piny na RPi. Pro možnost síťového propojení a výměny dat mezi zařízeními byl použit protokol MQTT. V tomto případě slouží jako snadný způsob pro vzdálené získávání dat, ovládání Raspberry Pi a komunikace se softwarem ROBO Pro Coding od společnosti Fishertechnik. Díky tomu je možné vytvořit síť z více zařízení, řídit je a získávat z nich data s pomocí jednoho centrálního PC a demonstrovat tak základní principy průmyslu 4.0.

1 Úvod

V příspěvku je demonstrováno řízení a sběr dat z modelu Fischertechnik prostřednictvím jednodeskového počítače Raspberry Pi. Dalším požadavkem je ukládání dat do centrální databáze a umožnění vzdáleného centrálního řízení modelu s demonstrováním principů průmyslu 4.0.

Jednodeskové počítače našly uplatnění v technickém vzdělání, protože skvěle zapadají do konceptu STEM, kterým se označuje kombinace vzdělávání v oborech vědy, techniky, technologie a matematiky. Kromě vzdělávání se též začínají široce využívat v průmyslové praxi pro své projektové schopnosti, jak mezi inženýry, tak i výrobci. Tyto mikropočítače díky snadnému propojení a vzdálené komunikaci splňují požadavky konceptu Průmyslu 4.0 a jeho nástrojů, jako je automatizace a platforma IoT (Internet věcí). Raspberry Pi (dále již jen RPi) je řada malých jednodeskových počítačů vyvinutých společností Raspberry Pi Foundation pro výuku základů počítačové vědy. Díky jejich jednoduchosti a všestranné použitelnosti se staly populárním, ale i experimentálním nástrojem pro vývoj školních projektů, hardwarového

https://doi.org/10.24132/PI.2022.11146.043-060

programování, robotiky, základních automatizovaných strojů, obvodů atd. RPi je plně vybavený počítač velikosti kreditní karty, na kterém se nejčastěji používají operační systémy s linuxovým jádrem. Počítač má všechny potřebné propojovací porty, kam může uživatel připojit periferní zařízení. Další předností RPi je, že disponuje řadou portů GPIO (General Purpose Input/Output, vstupy/výstupy pro obecné účely), které lze použít k interakci s klávesnicemi, myší, monitorem, sensory, motory, světly a dalšími zařízeními. Díky těmto pinům, všestrannosti, velikosti a výkonu, RPi postupně nachází uplatnění i v celých řadách průmyslových aplikací. V tomto případě jsou piny GPIO využity k připojení mechanických komponent stavebnice Fischertechnik. V tomto článku byl zvolen produkt RPi 4 Model B. RPi 4 Model B je nejnovější produkt v řadě počítačů RPi. Tento produkt byl zvolen proto, že má možnosti připojení periferií a umožňuje tak tvorbu programů přímo v OS Rasbian.[1][2] Dále byl zvolen z důvodu možnosti připojení Raspberry Pi F5 Adapter, který slouží k vyrovnání napětí mezi RPi a mechanickými komponenty Fischertechnik.

Fischertechnik je polytechnická stavebnice, umožňující stavbu kostry modelů ze stavebních bloků a mechanických součástí (včetně převodovek, ozubených kol apod.), výkonových součástí (elektromotory, ventily, kompresory, USB kamery, optické barevné senzory, světelné moduly) a součástí pro sběr/vyhodnocení údajů (např. NTC rezistor pro měření teploty). Stavebnici lze v případě potřeby rozšířit o další mechanické i výkonové součástky. Fischertechnik byl dříve určen jako hračka pro děti zajímající se o konstrukci. Dnes se však využití stavebnice rozrůstá, například pro výuku a porozumění technologiím na středních odborných školách, výzkum a vývoj na univerzitách, v průmyslových firmách a IT odděleních. Uplatnění stavebnice lze nalézt například v prototypování a k tvorbě reálných simulací. V tomto příspěvku byla použita stavebnice Fischertechnik řady Robotics. Sada Robotics je soubor modulárních komponent, které jsou určeny pro stavbu robotů. I vzhledem k jednoduchosti stavebnice se dají vytvářet komplexní systémy. K oživení a řízení modelů Fischertechnik slouží programovatelná řídící jednotka Robotics TXT Controller. Set nabízí dvě verze řídící jednotky Robotics TXT controller a Robotics Controller TXT4.0, který byl použit pro tento článek. Tvorba řídících programů pro řídící jednotku TXT4.0 primárně probíhá ve vývojovém prostředí ROBO Pro Coding, které podporuje programování v grafickém programovacím jazyce pomocí bloků anebo Pythonu. Cílem je vyzkoušet možnosti ROBO Pro Coding při tvorbě řídícího programu a komunikace s jinými zařízeními, nejenom s těmi, které nabízí společnost Fischertechnik. [3][4]

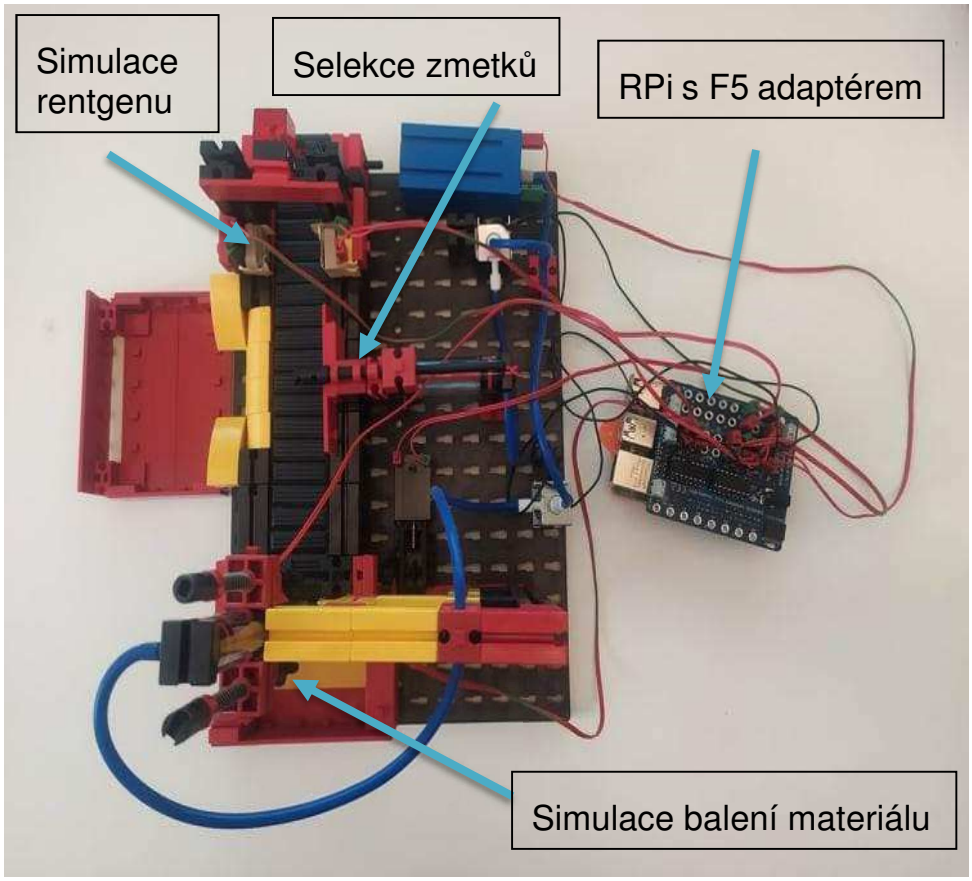
Pro výměnu informací mezi zařízeními byl zvolen protokol MQTT. MQTT (Message Queuing Telemetry Transport) je jednoduchý a nenáročný protokol pro předávání zpráv mezi klienty prostřednictvím centrálního bodu – brokeru. Díky této nenáročnosti a jednoduchosti je snadno implementovatelný i do zařízení s menším výpočetním výkonem. Tento síťový protokol umožňuje řídit a přenášet data z více modelů najednou, je tak možné snadno vytvořit IoT a je skvělým nástrojem pro průmysl 4.0.[5] Veškerá data se ukládají do lokální SQLite databáze umístěné na centrálním zařízení. Jako centrální zařízení byl

použit počítač s operačním systémem Windows. Na tomto zařízení byla vytvořena lokální SQLite databáze a program pro koncového uživatele pro ovládání modelu. Veškerý kód je psaný v programovacím jazyce Python. Tyto programy lze spustit po nainstalování požadovaných knihoven na kterémkoli zařízení s operačním systémem Windows nebo Linux. Byly vytvořeny dva koncepty možnosti sběru dat z Fischertechnik modelu:

- Koncept 1 využívá propojení mezi Fischertechnik a RPi GPIO piny a napájecí Raspberry Pi F5 adaptér. Jednotlivé piny jsou řízeny pomocí programu Python. Pomocí protokolu MQTT se data posílají z tohoto propojení do centrálního počítače, kde se ukládají do SQLite databáze. Z centrálního počítače se řídí model a jeho jednotlivé komponenty.
- Koncept 2 využívá Fischertechnik Controller TXT4.0 a vývojové prostředí ROBO Pro Coding. Sběr dat a řízení modelu je řešeno stejně jako v konceptu 1 pomocí protokolu MQTT.

2 Testovací model

Model Fischertechnik (obr.1) je část modelu výrobní linky postavené na KPV/ZČU s využitím setu ROBOTICS. Tato výrobní linka má selektovat materiál pomocí rentgenu a následně určit, jestli se materiál vyřadí (zmetek) nebo se dopraví na balící stanici a do přepravky na hotové výrobky.



Obrázek 1 - Testovací model Fischertechnik [Autor]

Materiál se pohybuje po dopravním pásu, který je poháněn elektromotorem. Rentgen je simulován světelnou bránou (fototranzistorem a LED žárovkou). Koncept funguje tak, že zastínění LED žárovky materiálem před fototranzistorem změní jeho hodnotu z 1 na 0. Simulace rentgenování probíhá pouze zastavením pásu na určitou dobu. Vyřazení zmetků je vyhodnocováno náhodně pomocí generování náhodného čísla (1 nebo 2) a je provedeno pomocí pneumatického válce, který je poháněn kompresorem a solenoidovým ventilem. Válec vytlačí zmetky do postranní přihrádky. Poslední část modelu je stanice, která simuluje balení materiálu. Tento úsek disponuje světelnou bránou, jež zaznamená přítomnost výrobku a zastaví pás. Simulace balení probíhá dalším pneumatickým válcem, který stlačí hlavu nástroje směrem dolů. Následně se materiál pomocí pásu dopraví do přihrádky na hotové výrobky. Přehled částí a komponent, z kterých se skládají:

- Dopravní pás (mini motor)
- Světelná brána - Rentgen (fototranzistor, LED žárovka)
- Odsun zmetků (kompresor, solenoidový ventil, pneumatický válec)
- Světelná brána (fototranzistor, LED žárovka)
- Balení materiálu (solenoidový ventil, pneumatický válec)

3 Implementace MQTT

Pro implementaci obou konceptů do IoT byl použit pro vzdálené propojení centrálního PC a ostatních zařízení protokol MQTT. Z tohoto PC je možné získávat data a komunikovat s RPi. Na PC byl použit operační systém Windows. Pro podporu protokolu MQTT, byl použit serverový software s názvem Mosquitto. Mosquitto je zprostředkovatel zpráv, který implementuje několik verzí protokolu MQTT, včetně nejnovější revize 5.0. Je to také relativně výpočetně nenáročný software, díky čemuž je Mosquitto perfektní volbou pro práci se zařízeními, které vyžadují malou spotřebu energie. [6]

Aby bylo možné používat MQTT, je potřeba server MQTT, takzvaný broker, a také odpovídající koncová zařízení, která odesílají (publishers) nebo přijímají data (subscribers). Obecně se tyto zařízení nazývají klienti. Protokol MQTT funguje tak, že klienti vystupují jako publishers (vydavatelé) nebo jako subscribers (předplatitelé). Publishers odešlou data na broker server, který spravuje veškerá data a slouží jako prostředník. Broker MQTT je zodpovědný za příjem všech zpráv, filtrování zpráv, rozhodování o tom, kdo o ně má zájem, a následné zveřejnění zprávy všem přihlášeným klientům k určitému tématu (subscriberům). Broker může být kterékoli zařízení v sítí, musí však být neustále zapnuté pro udržení MQTT spojení mezi zařízeními. V tomto příspěvku byl broker server nainstalován na RPi.

Broker Mosquitto MQTT je k dispozici pro RPi OS jako součást úložiště APT (Advanced Packaging Tool), takže instalace softwaru je jednoduchá. Mosquitto spolu s klientským softwarem byl nainstalován následujícím příkazem:

```
pi@raspberry:~ $ sudo apt install -y mosquitto mosquitto-
clients
```

Aby se Mosquitto server automaticky spustil při spuštění Raspberry Pi, je nutné spustit následující příkaz (to znamená, že zprostředkovatel Mosquitto se automaticky spustí při spuštění Raspberry Pi):

```
pi@raspberry:~ $ sudo systemctl enable mosquitto.service
```

Aby byl povolený vzdálený přístup, a aby bylo možné komunikovat s ostatními zařízeními v síti, byl upraven/vytvořen konfigurační soubor. Spuštěním následujícího příkazu byl otevřen soubor mosquitto.conf:

```
pi@raspberry:~$ sudo nano /etc/mosquitto/mosquitto.conf
```

Do konfiguračního souboru byly přidány dva řádky – listener 1883 a allow_anonymous true. Příkazem listener se stanovuje, kdo se může na daný server připojit. 1883 znamená že k připojení k brokeru není potřeba ověření a šifrování. Toto je výchozí port MQTT. Pro šifrované připojení je možné použít port 8883. Pokud jsou známy adresy IP klientů MQTT, lze povolit přístup pouze pro potřebné rozsahy IP. Toto opatření zablokuje všechny klienty, kteří nejsou v definovaných rozsazích IP adres. Jelikož je tento článek ukázková demonstrace použití MQTT, šifrováním a zabezpečením se nezabývá. Příkaz *allow_anonymous* true umožňuje připojení neznámých zařízení.

Pro připojení k broker serveru stačí IP adresa zařízení, na kterém je broker nainstalován. Konkrétně k zjištění IP adressy na RPi, lze použít příkaz:

```
pi@raspberry:~$ hostname -I
```

Knihovny MQTT pro OS Windows je možné nainstalovat pomocí PyPi. Pomocí PyPi byl nainstalován Eclipse Paho MQTT. Tento kód poskytuje třídu klienta, která umožňuje aplikacím připojit se k zprostředkovateli MQTT, publikovat zprávy a přihlásit se k odběru témat a přijímat publikované zprávy. Poskytuje také některé pomocné funkce, díky nimž je publikování jednorázových zpráv na server MQTT velmi jednoduché. Pro instalaci byl použit příkaz:

```
pip3 install paho-mqtt
```

MQTT rozděluje zařízení na publisher a client. Zařízení může být zároveň client a publisher. Všechny požadované MQTT funkce lze nalézt na [7]. Pro přístup ke všem funkcím potřebným pro připojení k brokerovi je potřeba naimportovat třídu klienta:

```
import paho.mqtt.client as mqtt
```

Pro publikování zpráv (publish) slouží modul paho.mqtt.publish:

```
import paho.mqtt.publish as publish
```

Pro připojení k serveru stačí využít následující funkci, zadat IP adresu a číslo portu:

```
client.connect("192.168.0.127", 1883, 60)
```

Funkce zpětného volání (callback function) *on_connect* se volá, když broker odpoví na žádost o připojení:

```
client.on_connect = on_connect
```

Obdobně funkce *on_message* se volá, když program dostane zprávu od brokera:

```
client.on_message = on_message
```

funkce loop*() slouží pro udržení toku síťového provozu s brokerem. Pokud nejsou volány, příchozí síťová data nebudou zpracována a odchozí síťová data nemusí být odeslána včas. Existují čtyři možnosti správy síťového cyklu. Forever je síťový cyklus a nevrátí se, dokud klient nezavolá disconnect(). Automaticky se stará o opětovné připojení:

```
client.loop_forever()
```

Aby bylo možné kontrolovat stav připojení k brokerovi, byla deklarována funkce on_connect, která určuje jaké téma program odebírá. Pro odběr určitého tématu byla použita funkce client.subscribe(“téma”):

```
def on_connect(client, userdata, flags, rc):
  print("Connected with result code "+str(rc))
  client.subscribe("Backmsg")
```

Funkce on_message určuje co se stane při obdržení zprávy. V tomto případě vypíše obdrženou zprávu. Msg je objekt a vlastnost payload obsahuje data zprávy, což jsou binární data. Pomocí této funkce je možné spustit příkazem jednotlivé komponenty Fishertechnik, popřípadě uložit obdrženou zprávu do databáze:

```
def on_message(client, userdata, msg):
  print(msg.topic+" "+str(msg.payload))
```

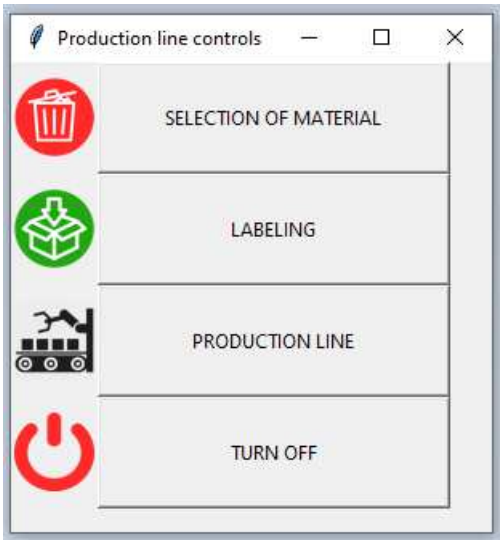
Pro odeslání zprávy brokerovi a následně od brokera všem klientům, kteří se přihlásí k odběru odpovídajících témat je nutné založit funkci publish. V této funkci můžeme určit o jakou kvalitu služeb se jedná (QoS). QoS je defaultně nastavené na 0. Tento modul poskytuje několik pomocných funkcí, které umožňují přímočaré jednorázové publikování zpráv. Jsou užitečné v situaci, kdy je žádoucí publikovat zprostředkovateli zprávu, a pak se od něj odpojit, aniž by bylo potřeba cokoliv dalšího. K dispozici jsou dvě funkce: single() - publikuje jedinou zprávu zprostředkovateli, multiple() – publikuje více zpráv najednou. Jako příklad je uvedena funkce publish.single, která je připojena na téma "Control" a odešle zprávu "4":

```
publish.single("Control", payload="4",hostname=
"192.168.0.127")
```

4 Vzdálené řízení modelu

Vzdálené řízení je řešeno pomocí jednoduchého uživatelského rozhraní (UI – user interface). UI se skládá ze 4 tlačítek (obr. 2):

1. SELECTION OF MATERIAL – spustí pracoviště pro selekci zmetků (pouze jednou)
2. LABELING – spustí pracoviště na balení materiálu (pouze jednou)
3. PRODUCTION LINE – spustí se cyklus řízení celé výrobní linky
4. TURN OFF – vypne cyklus a jednotlivé komponenty



Obrázek 2 - UI programu pro vzdálené řízení modelu [Autor]

Program funguje tak, že po stisku zvoleného tlačítka se odešle zpráva na RPi prostřednictvím MQTT pomocí funkce *publish.single*. Na RPi se po obdržení zprávy provede požadovaná funkce. Funkce *publish* je připojena na téma *Control* a k příslušnému brokeru pomocí IP adresy. Příklad publikování zprávy pomocí tlačítka *Selection*:

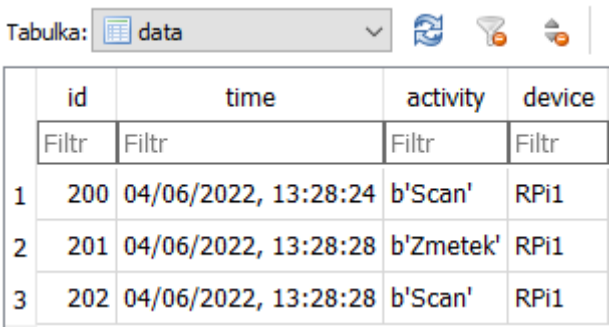
```
Selection.config(command=publishSelection)
def publishSelection():
    publish.single("Control",payload="1",hostname=
    "192.168.0.127")
    print("Selection material")
```

UI bylo vytvořeno pomocí knihovny Tkinter. Tkinter je knihovna grafického uživatelského rozhraní pro Python. Python ve spojení s Tkinterem poskytuje rychlý a snadný způsob vytváření aplikací s grafickým uživatelským rozhraním. Tkinter poskytuje výkonné objektově orientované rozhraní pro sadu nástrojů Tk GUI. Je multiplatformní, takže stejný kód funguje v systémech Windows, MacOS i Linux. Vizuální prvky jsou vykreslovány pomocí nativních prvků operačního systému, takže aplikace vytvořené pomocí Tkinteru vypadají, jako by patřily na platformu, na které jsou spuštěny.

5 Propojení databáze s použitím Python, SQLite a MQTT

Pro ukládání dat z modelu Fischertechnik do lokální databáze byl využit SQLite. Tento přístup byl zvolen z důvodu, že SQLite má malou kódovou stopu, efektivně využívá paměť, místo na disku a šířku pásma disku, je vysoce spolehlivý a nevyžaduje žádnou údržbu od správce databáze. Na rozdíl od databází založených na principu klient-server, kde je databázový server spuštěn jako samostatný proces, je SQLite pouze nevelká knihovna. Po přilinkování k aplikaci je k dispozici pomocí jednoduchého rozhraní. Každá databáze je uložena v samostatném souboru *.dbm* (Database Manager), kde se data ukládají za použití primárního klíče. Nevýhodou SQLite je, že funguje pouze jako lokální vestavěná databáze. Pro přístup k databázím v síti online se musí použít např. některé rozhraní API webové služby používané přes HTTP(S) nebo zmíněný protokol MQTT.

Byla vytvořena databáze, která slouží jako přehled zpráv z jednotlivých zařízení připojených k RPi. Jedná se o jednoduché zprávy, které nesou informaci kdy a jaké zařízení bylo spuštěno. K tomu poslouží jedna tabulka (s názvem *data*) s položkami – id, time, activity, device (viz. obr. 3). Id slouží jako primární klíč tabulky.



	id	time	activity	device
	Filtr	Filtr	Filtr	Filtr
1	200	04/06/2022, 13:28:24	b'Scan'	RPi1
2	201	04/06/2022, 13:28:28	b'Zmetek'	RPi1
3	202	04/06/2022, 13:28:28	b'Scan'	RPi1

Obrázek 3 - SQLite databáze, tabulka data [Autor]

Vkládání dat do databáze prostřednictvím MQTT se provádí funkcí *on_message*, která se volá po obdržení zprávy od brokera. Funkce vytvoří proměnou *date_time*, která využívá python knihovnu *datetime* pro určení času *a* následně zavolá funkci *main* s potřebnými proměnnými, která vloží data do databáze:

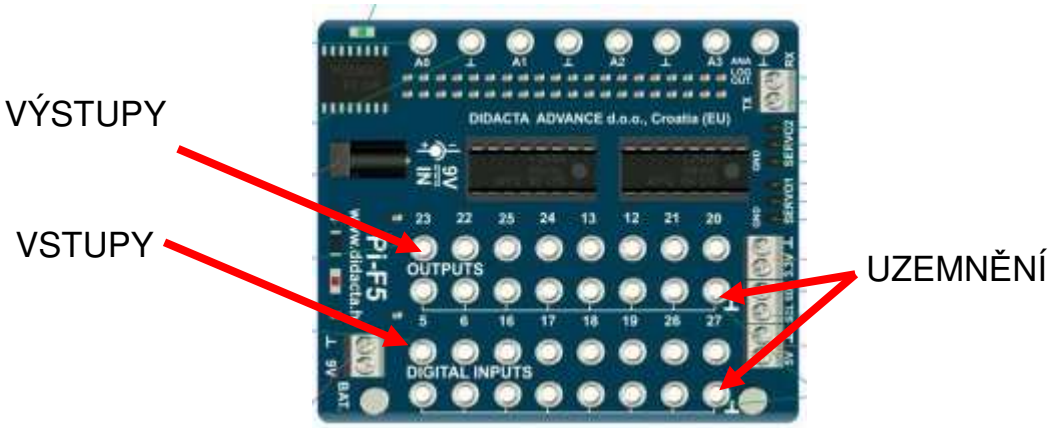
```
def on_message(client, userdata, msg):
    today = datetime.datetime.now()
    date_time = today.strftime("%m/%d/%Y, %H:%M:%S")
     print("message received ",
    str(date_time),str(msg.payload.decode("utf-8")))
    main(msg, date_time)


def main(msg, date_time):
     database = r"C:\Users\mhucl\OneDrive\
     Plocha\Diplomka\DeviceDatabase.db"
     conn = create_connection(database)
     with conn:
      data_1 = (str(date_time), str(msg.payload),"RPi1")
        create_data(conn, data_1)
```

Tento přístup pro ukládání dat do databáze a vzdálené řízení modelu pomocí uživatelského rozhraní, byl pro oba koncepty použit stejně.

6 Koncept 1

První koncept je navržen tak, že RPi přímo komunikuje s modelem Fischertechnik prostřednictvím GPIO pinů. Fischertechnik a GPIO RPi však pracují s jinými hodnotami napětí. GPIO RPi pracuje s napětím 3,3V, k

neopravitelnému poškození může dojít i při napětí 5 voltů. Komponenty Fischertechnik naproti tomu využívají provozní napětí 9 voltů, což rozhodně převyšuje možnosti RPi. Motory Fischertechnik také vyžadují proud kolem 250 miliampérů. Porty GPIO RPi nemohou poskytnout takové množství energie. V praxi však scénář nepředstavuje zásadní problém, protože úpravy úrovní byly v elektronice vždy nutné a existují běžné komponenty, které tento problém řeší. K připojení Fischertechnik k počítači Raspberry Pi nám poslouží speciální Raspberry Pi F5 Adapter viz obr. 4. Fischertechnik 179448 Raspberry Pi F5 Adapter je speciálně navržený elektronický adaptér pro ovládání DC motorů a servomotorů, stejně tak digitálních a analogových senzorů Fischertechnik. [8]



Obrázek 4 - Fischertechnik 179448 Raspberry Pi F5 Adapter [8]

Pro ovládání a adresování GPIO pinů byl použit modul RPi.GPIO. RPi.GPIO je modul Pythonu pro ovládání rozhraní GPIO na Raspberry Pi. Byl vyvinut Benem Crostonem a uvolněn pod licencí softwaru MIT.

Existují dva způsoby číslování IO pinů na Raspberry Pi v rámci RPi.GPIO. Prvním je použití systému číslování BOARD. Ten odkazuje na čísla pinů na hlavičce P1 desky Raspberry Pi. Druhým systémem číslování jsou čísla BCM. Jedná se o způsob práce na nižší úrovni – odkazuje na čísla kanálů v systému Broadcom SOC. Je nutné vždy nutné určit které číslo kanálu patří ke kterému pinu na desce RPi. Čísla pinů pro Raspberry Pi F5 Adapter jsou označena na vrchní straně desky počítače. Modul RPi.GPIO se importoval následujícím kódem:

```
import RPi.GPIO as GPIO
```

Systém číslování byl zvolen BCM:

```
GPIO.setmode(GPIO.BCM)
```

Pro nastavení kanálu je nutné určit, zda se jedná o vstup nebo výstup. Nastavení kanálu jako input(vstup):

```
GPIO.setup(pinnumber, GPIO.IN)
```

Pro čtení hodnot input pinu GPIO se používá kód:

```
GPIO.input(pinnumber)
```

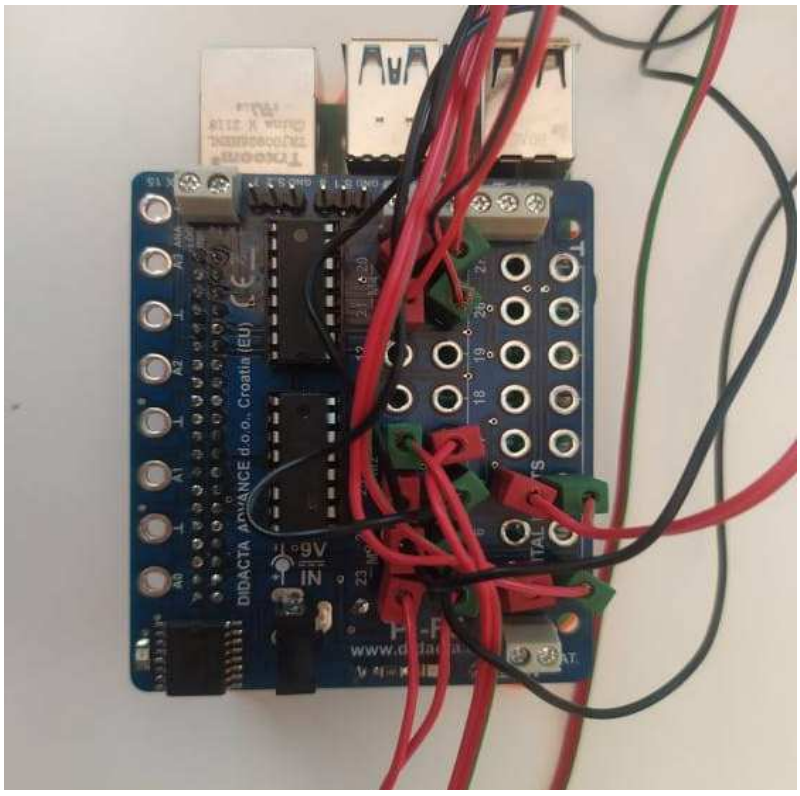
Nastavení kanálu jako output(výstup):

```
GPIO.setup(pinnumber, GPIO.OUT)
```

Lze zadat počáteční hodnotu výstupního kanálu (output):

```
GPIO.setup(pinnumber, GPIO.OUT, initial=GPIO.HIGH)
```

Pomocí změny výstupů tak tyto komponenty lze vypínat a zapínat, popřípadě získávat požadovaný input. Na obr.5 je zobrazeno zapojení elektronických komponent.



Obrázek 5 - Zapojení elektronických komponent do RPi [Autor]

6.1 Publikování zpráv a řízení RPi

RPi figuruje zároveň jako publisher i jako client. Cílem příspěvku je zaznamenávat aktivitu jednotlivých úseků modelu fishertechnik a tyto úseky i řídit. Při aktivitě jednotlivých mechanických komponent se odešle zpráva do

databáze prostřednictvím MQTT příkazu publish. Každá komponenta odesílá jiná data, která se pak ukládají do databáze.

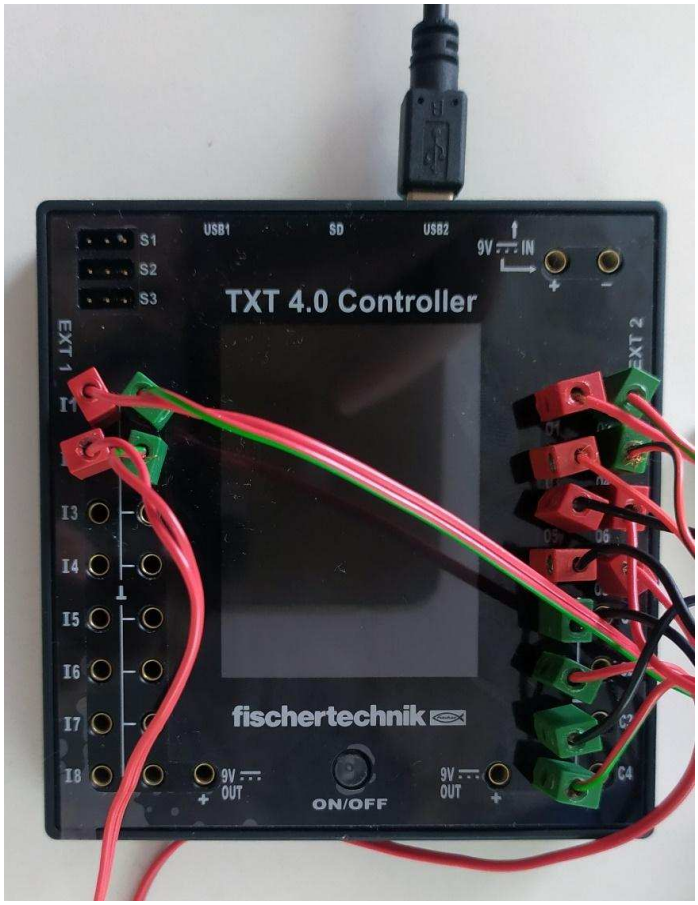
Pro řízení RPi byla vytvořena funkce *on_message*, která se volá po obdržení zprávy od brokera. Na základě obdržené zprávy se spustí požadované funkce. Pomocí funkce *client.subscribe* je klient napojený na určité téma. RPi je napojené na téma pro ovládání jednotlivých komponent. Klient může obdržet od uživatele celkem 4 zprávy (*1, 2, 3, 4*) a na ty následně reagovat. Problém nastává, když je nutné zapnout některý z cyklů, protože v programu už jeden cyklus běží - *client.loop_forever*. Aby se udrželo MQTT spojení a uživatel mohl program i nadále ovládat a cyklus pomocí příkazu libovolně zastavit, byla použita metoda multithreadingu. Multithreading neboli vícevláknové zpracování v jazyce Python umožňuje procesorům spouštět různé části (vlákna) procesu současně, aby se maximalizovalo využití procesoru. Vícevláknové zpracování tak umožňuje procesoru zpracovávat více cyklů/programů najednou, aby se neovlivňovaly. Knihovna pro multithreading se importuje pomocí příkazu *import threading*. Tento problém by se dal řešit i pomocí zvolení *loop_start* funkce. Tato funkce spustí na pozadí vlákno, které automaticky zavolá funkci *loop()*. Voláním *loop_stop()* se vlákno na pozadí zastaví a uvolní hlavní vlákno pro jinou práci. Opětovně se funkcí *loop_start* může zapnout požadovaný cyklus. Objevil se však problém s duplicitou odeslaných zpráv po opětovném připojením k brokerovi, a proto byla zvolena metoda spuštění funkce *mainprogram* na jiném vláknu. Funkce pro obdržení zpráv a následné reakci na ně, může vypadat následovně:

```
def on_message(client, userdata, msg):


  if msg.payload == b'3':
      exit_event.clear()
      global main_thread
     main_thread = threading.Thread(target=mainprogram,
      name="mainprogram")
      main_thread.start()


  elif msg.payload == b'4':
      exit_event.set()
      GPIO.output(ledone,GPIO.LOW)
      GPIO.output(ledtwo,GPIO.LOW)
      GPIO.output(pistone,GPIO.LOW)
```

Pro uvolnění vlákna a zastavení cyklu *mainprogram* se použila metoda zpracování události (Event handling). Zpracování událostí vytváří responzivní aplikaci, která dokáže detekovat a generovat akce na základě odezvy. Byla vytvořena událost *exit_event = threading.Event().* V programu je pak možné sledovat stav této události a podle stavu této události provádět vypínat a zapínat jednotlivé cykly.

7 Koncept 2

V další variantě je model Fischertechnik ovládán pomocí řídící jednotky TXT Controller 4.0. Ten je naprogramován pomocí programu ROBO Pro Coding. TXT4.0 komunikuje s PC pomocí WiFi sítě. Vzdálené řízení a sběr dat ze zkušebního modelu je řešeno stejně jako v konceptu 1 pomocí MQTT protokolu. Na obr. 6 je zobrazena řídící jednotka a zapojení jednotlivých komponent do pinů.



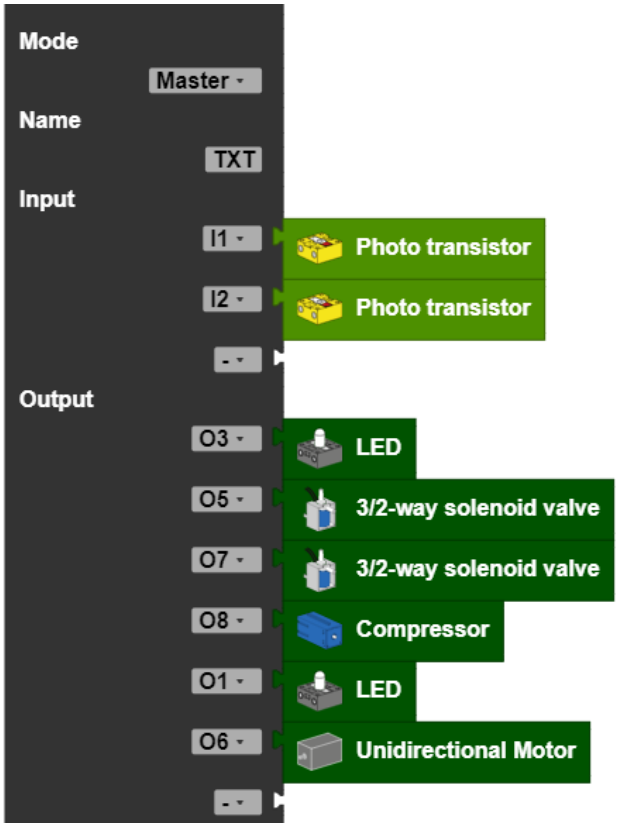
Obrázek 6 - Zapojení komponentů Fischertechnik do TXT4.0 [Autor]

7.1 ROBO Pro Coding

Pomocí softwaru ROBO Pro Coding byl pro model vytvořen řídící program. ROBO Pro Coding je grafický programovací software, kde se programuje formou skládání programovacích bloků, nicméně umožňuje i textové

programování přes Python. To umožňuje zobrazit funkce programu srozumitelným způsobem. V případě potřeby lze programovat řídící kód čistě i v jazyku C/C++ nebo v Pythonu. Software je velice přehledný, jednoduchý a přístupný pro začátečníky. Programovací prostředí Fischertechnik ROBO Pro Coding je dostupné na platformách Windows10 / Mac OS / Linux / iOS / Android a je k dispozici zdarma v příslušných obchodech s aplikacemi. [9] Nicméně se nepovedlo software nainstalovat na RPi OS, proto byl pro programování v softwaru a řízení řídící jednotky použit počítač s operačním systémem Windows. K zařízení se počítač připojil prostřednictvím WiFi sítě. Zařízení musí být připojená ke stejné síti, pro vzdálené propojení např. z domova by se propojení mohlo řešit přes routování. Fischertechnik nabízí také možnost komunikovat s řídící jednotkou prostřednictvím Fischertechnik Cloud. Tato varianta neni geograficky omezená a model se může ovládat pomocí smartphonu, tabletu nebo jakéhokoli počítače s přístupem k internetu. Stačí se zaregistrovat na oficiálních stránkách Fischertechnik Cloud a pomocí programovacího bloku *fischertechnik cloud connect* daný program ke cloudu připojit.

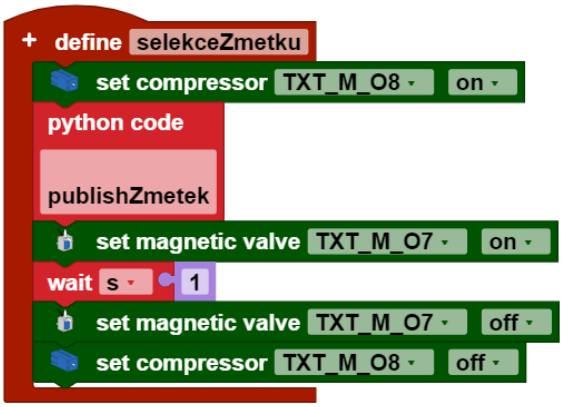
Pro použití jednotlivých akčních komponent stavebnice Fischertechnik je nutné je připojit k řídící jednotce nejen fyzicky, ale i softwarově. Pro softwarové připojení v programu ROBO Pro Coding slouží záložka *Controller Configuration*, kde se nachází všechny bloky komponentů Fischertechnik pro připojení k řídící jednotce. Příklad použitých komponent a jejich softwarové zapojení lze vidět na obr. 7.



Obrázek 7 - Konfigurace řídící jednotky v programu ROBO Pro Coding [Autor]

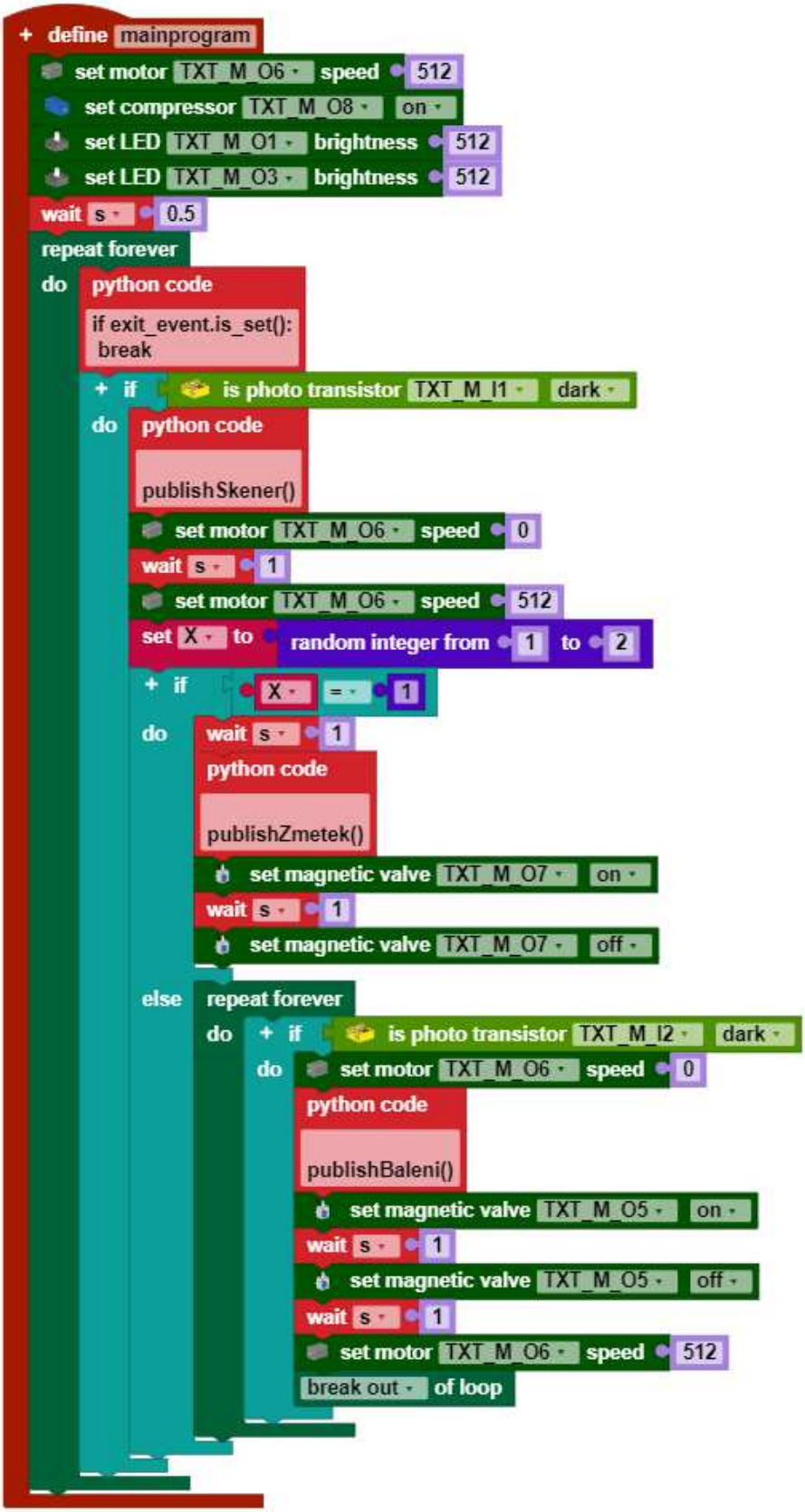
7.2 Robo Pro Coding program pro řízení zkušebního modelu

Program vytvořený ve vývojovým prostředí Robo Pro Coding má stejnou funkcionalitu jako python program uvedený v konceptu 1. Program funguje jako klient a publisher zároveň. Respektive po obdržení určité zprávy MQTT se spustí požadovaná funkce, s tím rozdílem, že program byl vytvořen pomocí grafických programových bloků. Jak už bylo zmíněno Robo Pro Coding umožňuje kombinovat programovací bloky s textovým kódem python. Je tak možné vkládat python knihovny a zakládat jednotlivé funkce, které pak můžeme pomocí python code volat. Jako příklad je uvedená funkce selekceZmetku (viz. obr. 8), která zapne pracoviště pro selekci zmetků. Funkce zapne kompresor zavolá další funkci *publishZmetek*, otevře zvolený píst na 1 vteřinu a vypne kompresor.



Obrázek 8 - Příklad vkládání funkce v programu ROBO Pro Coding [Autor]

Jednotlivé funkce se volají, stejně jako v konceptu 1, pomocí obdržené zprávy přes protokol MQTT. Zpracování těchto zpráv a volání jednotlivých funkcí zajišťuje funkce *on_message*. Logika této funkce je stejná jako v konceptu 1. Respektive po obdržení zprávy od uživatele spustí určité funkce. Jako příklad je uvedená funkce *mainprogram*, což je cyklus, který zajišťuje chod pro celou linku, a je zobrazen na obrázku 9.



Obrázek 9 - Funkce mainprogram – cyklus pro ovládání modelu [Autor]

8 Závěr a hodnocení

V tomto článku bylo představeno řešení problematiky řízení modelu Fischertechnik a sběru dat s využitím RaspberryPi. K simulaci výrobní linky byl použit model Fischertechnik. Výrobní linka reprezentuje pracoviště rentgenu, selekci zmetků a balení. Byly vytvořeny dva koncepty propojení:

Koncept 1 využívá GPIO piny Raspberry Pi k přímé komunikaci se stavebnicí Fischertechnik. K řízení modelu byl zvolen programovací jazyk Python. Python byl zvolen, protože existuje velké množství knihoven od uživatelů pro ovládání Raspberry Pi, a proto nebylo nutné se zabývat nízkoúrovňovým programováním.

Koncept 2 pracuje s řídící jednotkou Controller TXT4.0 od výrobce Fischertechnik, který je propojený s počítačem prostřednictvím WiFi sítě pomocí programu ROBO Pro Coding.

Oba dva koncepty jsou přístupem takřka identické. Požadovanou funkcionalitu řízení a sběru dat splňují oba. U konceptů pro výměnu dat mezi jednotlivými zařízeními byl zvolen serverový přístup pomocí protokolu MQTT. Pro obě varianty byl zpracován koncept pro ukládání dat do SQLite databáze a řízení modelu pomocí jednoduchého UI. Python programy pro obě funkcionality byly spuštěny na centrálním PC s operačním systémem Windows. Koncept je navržen tak, aby programy mohly být spuštěny na kterémkoliv zařízení, které podporuje python a má nainstalované požadované knihovny. Jako centrální PC může figurovat například další RPi. RPi slouží kromě toho jako řídící jednotka v konceptu 1 i jako MQTT server (broker).

Hlavní rozdíl je v přístupu tvorby řídícího programu. ROBO Pro Coding nabízí zajímavý přístup tvoření pomocí programovatelných grafických bloků na rozdíl od RPi, kde probíhala tvorba programu v klasickém python prostředí pomocí GPIO knihovny. Tento grafický přístup se hodí pro výuku programovací logiky, základů robotiky a průmyslového inženýrství. Dalším rozdílem je způsob připojení k řídící jednotce. ROBO Pro Coding nabízí snadný přístup ke Controlleru TXT4.0 prostřednictvím WiFi sítě a umožňuje číst a ukládat data přímo ze softwaru do databáze pomocí python nadstavby. Není tak nutný vzdálený přístup pomocí protokolu MQTT a vše se může ovládat v rámci programu ROBO Pro Coding, kde lze vytvořit i optimalizované uživatelské prostředí pro ovládání robota pomocí záložky Display Configuration. Komunikace s programem ROBO Pro Coding prostřednictvím MQTT byl zvolen z důvodů vyzkoušení možnosti využití softwaru od společnosti Fishertechnik v „klasickém“ programování a možnosti komunikace s jinými zařízeními a softwarem, nejenom s těmi, které nabízí společnost Fischertechnik.

Pomocí tohoto přístupu je možné vytvořit síť modelů, které na sebe mohou reagovat a spolu komunikovat (IoT). Další zařízení (klienty) stačí přihlásit k odběru k požadovaném tématu zprávy odeslané brokerem, popřípadě data publikovat. Je tak možné vytvořit funkční, automatizovanou a inteligentní továrnu (smart factory), jež by mohla flexibilně reagovat na tok materiálu a

zaznamenávat veškerou aktivitu. Model je možné ovládat a získávat data i z domova prostřednictvím internetové sítě a IP adresy. Zařízení nejsou omezená operačním systémem díky multiplatformní podpoře programovacího jazyka Python a MQTT. Výměna dat pomocí protokolu MQTT byla zvolena pro vyzkoušení principů IoT a jeho rostoucí popularity v průmyslové praxi. Principy demonstrované v tomto článku se dají aplikovat na řízení a sběr dat takřka jakýchkoliv robotických komponent.

Poděkování

Příspěvek byl vytvořen za podpory projektu SGS-2021-028 s názvem "Vývojové a tréninkové prostředky pro interakci člověka a kyber-fyzického výrobního systému“ řešeného v rámci Interní grantové agentury Západočeské univerzity v Plzni.

Použitá literatura

[1] UPTON, Eben a HALFACREE, Gareth, Raspberry Pi Uživatelská příručka. 2013. Brno: COMPUTER PRESS. 232 s. ISBN 978-80-251-4116-8.

[2] RASPBERRY PI FOUNDATION, Raspberry Pi. [online] [cit. 27.8.2022]. Dostupné z: https://www.raspberrypi.org/

[3] FISHERTECHNIK, Fischertechnik. [online] [cit. 3.12.2021]. Dostupné z: https://www.fischertechnik.de/en/

[4] MALAGA, Miroslav a ULRYCH, Zdeněk. (2019). Koncept STEM se zaměřením na problematiku Industry 4.0. DOi: 10.24132/PI.2019.08948.094-100. [online] dostupné z: https://www.researchgate.net/publication/336948497_Koncept_STEM_se_zamerenim_na_problematiku_Industry_40

[5] WEISE, Jorn. Controlling a robot with MQTT - [Part 1]. 2021. [online] [cit. 27.8.2022]. Dostupné z: https://www.az-delivery.de/en/blogs/azdelivery-blog-fur-arduino-und-raspberry-pi/mit-mqtt-einen-roboter-steuern-teil-1

[6] MOSQUITTO. Eclipse Mosquitto. [online] [cit. 26.8.2022]. Dostupné z: https://mosquitto.org/ [6]

[7] RALIGHT, eclipse/paho.mqtt.python , 2021, [online] [cit. 26.8.2022]. Dostupné z: https://github.com/eclipse/paho.mqtt.python

[8] FISHERTECHNIK, Raspberry PI-F5, [online] [cit. 3.12.2021]. Dostupné z: https://www.fischertechnik.de/en/teaching/arduino-microbit-u-co/raspberry-pi/raspberry-pi-f5

[9] FISHERTECHNIK, Robotics-downloads, [online] [cit. 3.7.2022]. Dostupné z: https://www.fischertechnik.de/en/service/downloads/robotics