Napojení na STEM model pro potřeby průmyslových inženýrů

Petr Švrčula[1], Miroslav Malaga[1], Zdeněk Ulrych[1]

[1] Západočeská univerzita v Plzni, Fakulta strojní, Katedra průmyslového inženýrství a managementu

Univerzitní 8, 306 14, Plzeň, Česká republika
svrculap@kpv.zcu.cz
malaga@kpv.zcu.cz
ulrychz@kpv.zcu.cz

Anotace: Práce se zabývá navržením funkčního propojení mezi jednotkou Fischertechnik TXT Controller a osobním počítačem pomocí mikrokontroleru Arduino. Navržené řešení umožňuje použít blokové programování jednotky TXT Controller pro komunikaci s počítačem. Řešení využívá Arduino jako překladač mezi sériovým portem PC a rozhraním I2C. Odpadá tak složité programování v jazycích C/C++ a kompilace pro linuxové jádro běžící na TXT Controlleru a ARM procesor. To poskytuje možnost využít potenciál jednotky TXT Controller i studenty v oboru průmyslového inženýrství. Během návrhu bylo identifikováno několik komunikačních omezení, jejichž odstranění je předmětem dalšího vývoje.

1 Úvod

Cílem této práce je demonstrovat možnosti propojení průmyslového modelu Fischertechnik s C# aplikací pro jeho řízení a sběr dat při zachování co největšího rozsahu řídícího kódu HW modelu v grafickém programovacím jazyce. Z důvodu nemožnosti komunikace s TXT Controllerem napřímo např. přes COM Port, je potřeba navrhnout řešení pomocí I^2C sběrnice, která je v grafickém programovacím jazyce pro programování Fischertechnik TXT Controlleru jediná možnost komunikace řídící jednotky s okolím.

Průmyslový model využitý v této práci je Fischertechnik Color Sorter. Jedná se o model robotického ramene, které třídí barevné puky podle přečtené barvy do tří úložných boxů. Tento model je vybaven řídící jednotkou FischertechnikTXT Controller.

Součástí projektu je vytvoření programových vrstev použitých zařízení a provedení jejich fyzického propojení. Pro programovou výbavu PC byly zvoleny jazyky C# a WPF. Program výsledného řešení musí v reálném čase vyčítat definované hodnoty z průmyslového modelu Fischertechnik, tyto hodnoty opatřit časovým razítkem a zaznamenat je do textového souboru. Program dále musí umožňovat spuštění manuálního režimu a uživatelský výběr úložného boxu.

https://doi.org/10.24132/PI.2021.07927.203-219

2 Průmyslový model Fischertechnik

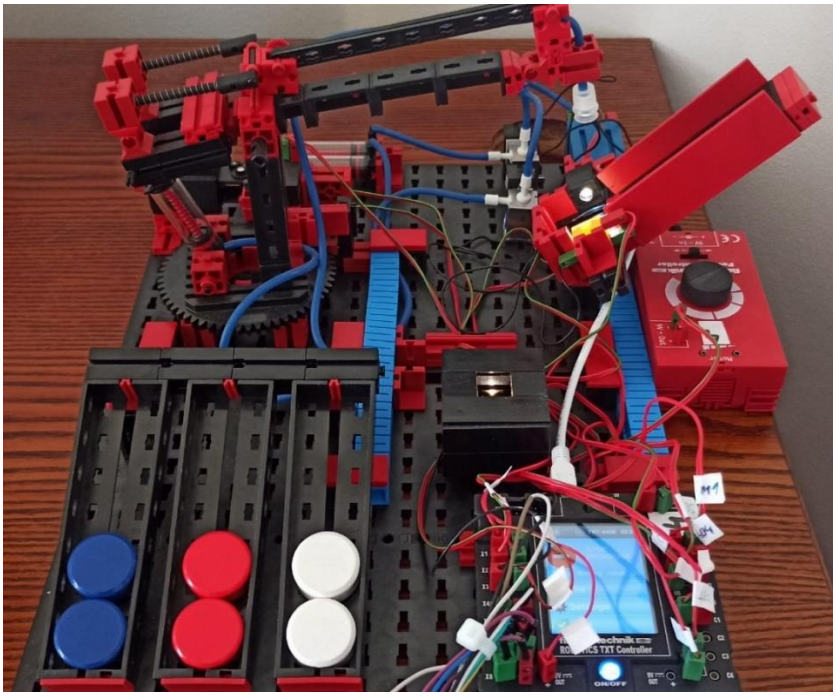
Fischertechnik STEM Engineering je studijní demonstrační stavebnice, která obsahuje komplexní přehled robotických, kódovacích a automatizačních systémů. Stavebnice se standardně programuje pomocí objektového programovacího prostředí ROBO Pro grafickým programovacím jazykem. [1]

Využitý model, viz Obrázek 1, disponuje robotickým ramenem s vakuovým efektorem, vstupním zásobníkem s opto závorou, analogovým barevným sensorem, třemi úložnými boxy, pneumatickým systémem a řídící jednotkou Fischertechnik TXT Controller.

Jednotka Fischertechnik TXT Controller je tvořená dvoujádrovým 32bitový procesorem ARM Cortex A8 s rychlostí 500 MHz. Je vybavena pamětí RAM o velikosti 128 MB a 64 MB pamětí typu flash pro ukládání dat. Tato paměť je rozšiřitelná pomocí SD karty. Na přední straně jednotky je umístěný barevný dotykový displej s rozlišením 320x240, 16 vstupů a výstupů (8 univerzálních vstupů, 4 čítací vstupy a 4 výstupy pro motor nebo LED). Rozšířená konektivita jednotky umožňuje komunikaci s vnějším světem přes rozhraní USB, infraport, wifi nebo Bluetooth. [2]

V základním režimu pracuje robot tak, že při vložení barevného puku do zásobníku řídící jednotka detekuje přerušení optické závory a spustí robotické rameno. To uchopí puk, přiloží ho na čidlo barvy a tam dojde k odečtení analogové hodnoty reprezentující jednu ze tří barev. Na základě těchto dat vydá řídící jednotka povel k uložení do boxu přiřazeného k barvě.

Pohyb robota je realizován pomocí motoru s mechanickým odečtem polohy. Pneumatický systém je pak složen z kompresoru, elektro ventilů a soustavy pístů pro vytvoření zdroje vakua.

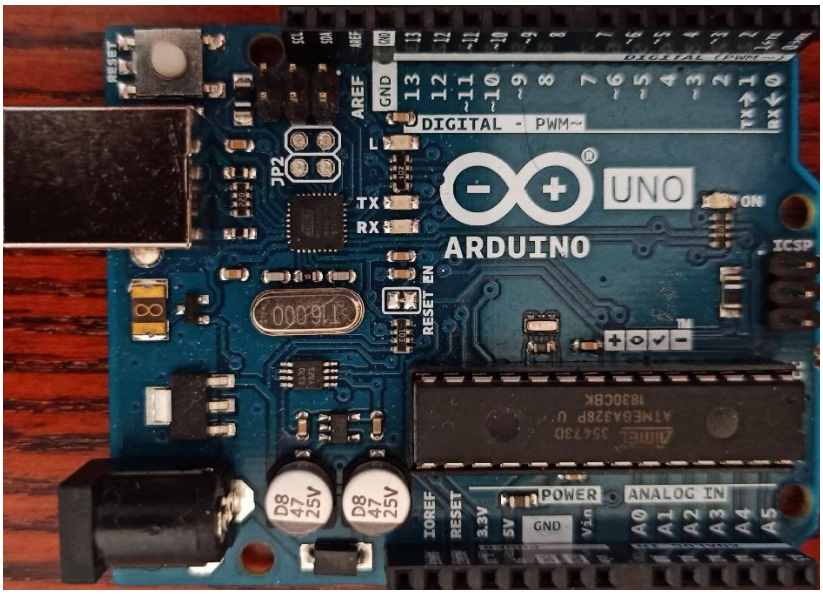


Obrázek 1 - Průmyslový model Fischertechnik

2.1 Arduino Uno

Pro účel řešeného projektu byla použita originální deska Arduino Uno viz Obrázek 2. Tato verze byla zvolena s ohledem na to, že se jedná o jednu z nejvíce používaných desek. Řešení v této práci bude fungovat i na neoriginálních klonech Arduina, nicméně může docházet k neočekávanému chování.

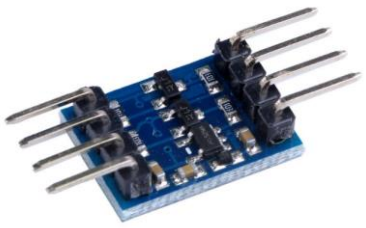
Deska je založená na čipu ATmega328. Má 14 digitálních vstupních/výstupních pinů (6 lze použít jako PWM), 6 analogových vstupů, 16 MHz oscilátor, připojení USB, DC napájecí konektor, konektor ICSP a resetovací tlačítko. Uno bylo první verzí desky Arduino pro USB připojení, která byla vydaná společně s vývojovým prostředím Arduino IDE 1.0. [3]



Obrázek 2 - Arduino UNO

2.2 Převodník napěťových úrovní

Kvůli rozdílným napěťovým hladinám I^2C sběrnic na Arduinu a Fischertechnik TXT Controller, je nutné použít převodník napěťových úrovní 3,3 V a 5 V viz Obrázek 3. Jedná se o klíčový prvek komunikace I^2C, a bez jeho použití by mohlo dojít k poškození jednotky Fischertechnik TXT Controller. Pro sběrnici I^2C musí být použitý obousměrný převodník, jinak dané řešení nebude fungovat.



Obrázek 3 - Převodník napěťových úrovní

2.3 Návrh komunikačního rozhraní

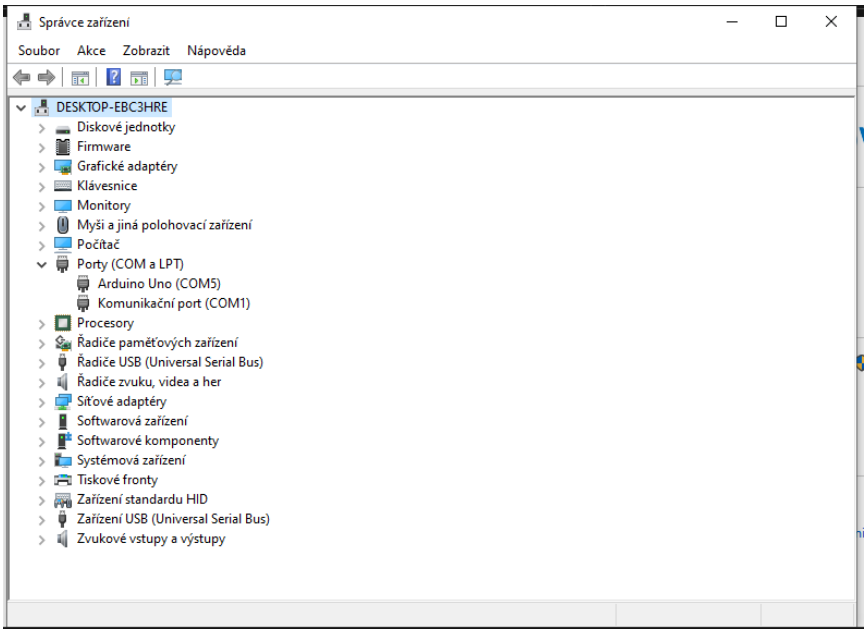
V projektu jsou použity dva typy komunikačního rozhraní a mikrokontroler Arduino mezi nimi slouží jako překladač viz Obrázek 4 - Komunikační schéma případové studie. Toto řešení rozšiřuje možnosti použití stavebnice Fischertechnik s počítačem a zároveň nabízí uživatelsky přívětivější možnost konfigurace tohoto propojení.

Obrázek 4 - Komunikační schéma případové studie

2.3.1 USB sériová linka

Pomocí sériové komunikace jsou posílány data po 1 bitu v řadě za sebou. Sériová linka je dvouvodičová, jeden vodič je přijímací a druhý vysílací. Díky tomu dokáže sběrnice současně číst i zapisovat pomocí dvou vláken programu, kdy jedno vlákno zajišťuje čtení a druhé odesílání dat. Pro správnou funkci sériové komunikace musí mít obě zařízení nastavenou shodnou rychlost baud (jednotka modulační rychlosti přenosu dat v asynchronním komunikačním kanále) [4]. Ta zajišťuje synchronizaci přenosu a integritu přenášených dat. Jelikož je standard RS-232 výhradou již spíše v průmyslu a u osobních počítačů byl nahrazen univerzálním sériovým portem (USB), je v tomto projektu použitá komunikace pomocí virtuálního COM portu. Rychlosti sériové linky dosahují až 115 200 baud, se vzrůstající rychlostí ale může docházet k vyšší chybovosti. Pro většinu projektů je dostačující rychlost 9 600 baudů, ta je použita i v řešeném projektu.

USB virtuální COM port slouží pro připojení desky Arduino UNO k osobnímu počítači. Čip FTDI, který zprostředkovává tento port je implementován přímo na desce Arduino, ta se jen připojí k PC pomocí portu USB. Po nastavení správného COM portu v komunikačním rozhraní, pak přes tento port probíhá obousměrná sériová komunikace počítače a mikrokontroleru Arduino. Virtuální COM port s připojeným mikrokontrolerem Arduino je možné zjistit ve správci zařízení daného PC viz Obrázek 5.



Obrázek 5 - Zjištění správného COM portu ve správci zařízení

2.3.2 Sběrnice I²C

Sběrnice I²C byla vyvinuta společností Philips v 80. letech minulého století. Jedná se o dvou vodičovou sběrnici, která umožňuje jednoduchou komunikaci mezi jednotlivými komponenty. Jedná se o *multi-leader* sběrnici, to znamená, že sběrnice dokáže pracovat s více leader jednotkami a má mechanismy, které zajišťují, aby nedocházelo ke kolizím a ztrátám dat. [5]

Základní rychlost komunikace je 100 kbit, tato rychlost je použita i v řešeném projektu. Sběrnice nicméně umožňuje i takzvaný „FASTMODE“ s rychlostí 400 kbit a od roku 1998 je dokonce dostupná rychlost 3,4 Mbit. [5]

Sběrnice se často používá i pro komunikaci s jednotlivými periferními zařízeními, jako jsou různé snímače, displeje a jiné typy komponent. Zařízení zapojená do komunikační linky jsou adresovatelná a mohou fungovat jako *leader*, nebo *follower*. Zařízení *leader* se stará o časování sběrnice pomocí vodiče CLK, odpadá tak nastavování rychlosti baud. *Follower* kontinuálně přijímá data od *leader* jednotky, odesílá je však pouze pokud je dotázán. [5]

Pro komunikace využívá sběrnice dvou vodičů. Jeden vodič používá pro časovou synchronizaci (SCL) a druhý pro data (SDA). Sběrnice funguje na napěťových úrovních 5 V nebo 3,3 V. Jelikož zařízení použitá v tomto projektu používají odlišné napěťové úrovně, je v práci použit převodník napěťových úrovní pro I²C, viz kapitola 2.2.

3 Realizace vlastní případové studie

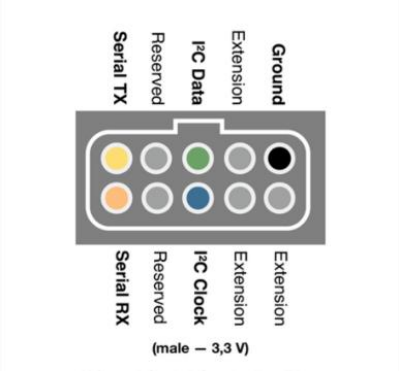
Během realizace napojení modelu byla zjištěna omezující vlastnost jednotky Fischertechnik TXT Controller. Jednotka je defaultně nastavena jako leader a používá interně uloženou I²C adresu, kterou není možné změnit, a ačkoliv je sběrnice I²C *multi-leader*, nedokáže řídící jednotka spolupracovat s další *leader*

jednotkou připojenou na sběrnici. V rámci zkušebního řešení bylo testováno několik variant, ale řešení bylo funkční pouze v režimu Fischertechnik TXT Controller jako *leader* a Arduino jako *follower*.

Toto omezení se vztahuje na jednotku TXT Controller naprogramovanou pomocí vývojového prostředí RoboPro. Je pravděpodobné, že s použitím pokročilejšího programování, pomocí jazyka C/C++, nebo Python, by bylo možné toto omezení odstranit.

S tímto omezením je možné realizovat odesílání informací z TXT Controlleru do Arduina, v opačném směru to ale lze pouze na vyžádání leader jednotky, to znamená, že Fischertechnik TXT Controller nejprve musí Arduino o tyto data požádat. Arduino není schopné samovolně poslat příkaz do jednotky Fischertechnik TXT Controller tak, aby ho zaznamenala. Druhá možnost je přeš Arduino simulovat a odeslat stisknutí HW tlačítka – v takovém případě se výstup z Arduina na straně TXT Controlleru připojí jako HW tlačítko. V tomto případě vzhledem k jiné napěťové hladině I/O pinů bylo nutné ověřit, zda TXT Controller bude detekovat 5 V jako stav HIGH, jelikož standardně používá 9 V. V rámci zkoušky bylo ověřeno, že tato varianta je funkční a TXT Controller spolehlivě detekuje 5 V jako stav HIGH.

Jednotka průmyslového modelu Fischertechnik byla připojena pomocí I^2C převodníku k Arduinu pomocí rozšiřujícího rozhraní EXT viz Obrázek 6. Pro dodatečnou komunikaci s modelem byly použity 4 I/O digitální piny.



Obrázek 6 - Rozšiřující rozhraní jednotky Fischertechnik TXT Controller [6]

3.1 Program řídící jednotky

Program je složen ze základních funkčních bloků a vnořených sekvencí. Vnořené sekvence jsou použity pro komplexnější funkční celky, jako je pohyb robota, ovládání efektoru, nebo identifikace barvy a jsou v programu označeny zeleným podbarvením.

V programu bylo nutné provést úpravy pro potřeby I^2C komunikace. Do zájmových částí kódu byly vloženy bloky zápisu na sběrnici I^2C. TXT Controller tak vždy v požadovaném místě odešle stavovou informaci. Ta je vyjádřena unikátním číslem, které odpovídá různým stavům uvedeným v Tabulka 1. Tyto zápisové bloky obsahují veškerou nezbytnou konfiguraci pro odeslání dat po I^2C a konfigurují se nezávisle.

Tabulka 1 - Přiřazení znaků pro jednotlivé stavy (kódovací tabulka)

Znak	Stav
1	Puk detekován
2	Zahájení pohybu ramene
3	Detekována bílá
4	Detekována červená
5	Detekována modrá
6	Uložení do boxu
7	Výchozí pozice – konec
8	Vstupní pozice je prázdná

Příchozí požadavky (příkazy z PC) je v tomto řešení nutné zpracovávat pomocí čtení stavu digitálních vstupů. Program vytvořený v RoboPro ve výchozím stavu kontroluje stav vstupu I8 pro určení automatického nebo manuálního režimu. Pokud je v automatickém režimu, tak při detekci puku provede jeho zatřídění na základě změřené barvy. Pokud je ale tento vstup ve stavu HIGH, tak program místo kontroly optické závory začne kontrolovat stav digitálního vstupu I7, který spouští manuální zatřídění do zvoleného boxu. Pokud je na vstupu I7 detekován stav HIGH, vyčte Fischertechnik TXT Controller stav vstupů I5 a I6 a na základě jejich kombinace provede zatřídění do zvoleného boxu.

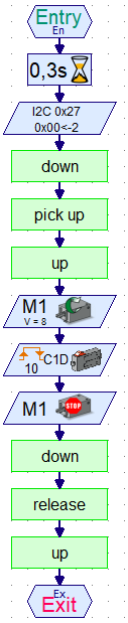
Pokud se v té době nenachází puk na vstupní pozici, program zatřídění neprovede a zašle chybový kód pomocí sběrnice I^2C, ten následně zpracuje Arduino pod stavem „Vstupní pozice je prázdná“.

Určení požadovaného boxu probíhá pomocí komparátoru na vstupech I5 a I6 a nabývá tří hodnot viz Tabulka 2:

Tabulka 2 - Rozhodovací podmínky pro určení zvoleného boxu

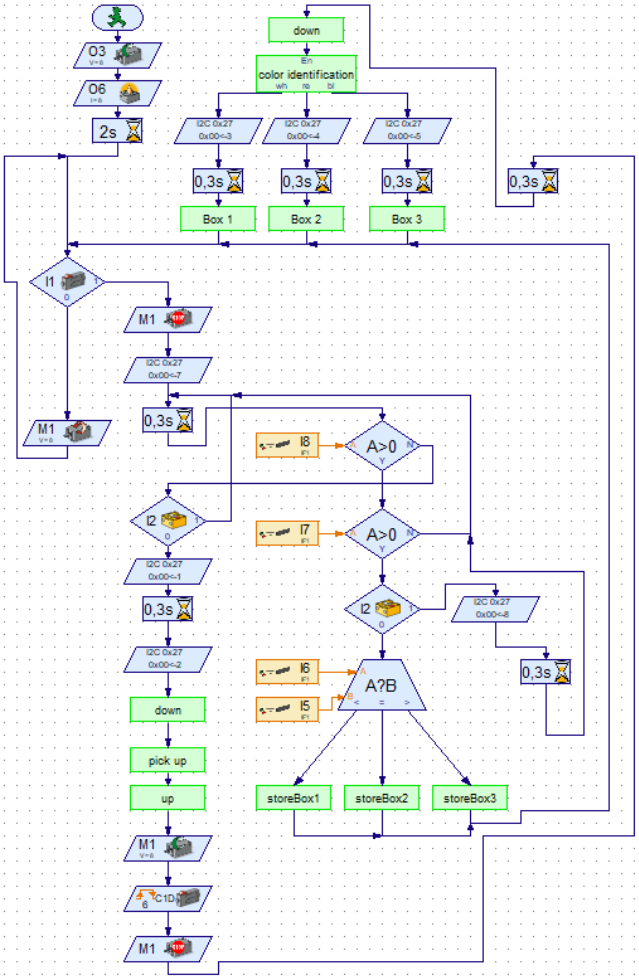
I5> I6	Box 1 (modrá)
I5 = I6	Box 2 (červená)
I5 <I6	Box 3 (bílá)

Pro založení do zvoleného boxu byla vytvořena nová pohybová sekvence robota pro každý box. Její ukázka je uvedena na Obrázek 7, jednotlivé sekvence se navzájem liší počtem impulsů na čítacím pinu *C1D*. V úvodu funkce je uveden příklad funkce I^2C, která odesílá informaci o zahájení pohybu robota a je použita v různých částech kódu.



Obrázek 7 - Příklad nově vytvořené sekvence

V každé části však odesílá rozdílnou hodnotu. Blokové zobrazení celého programu vytvořeném v RoboPro je uvedeno na Obrázek 8.



Obrázek 8 - Vytvořený program RoboPro

3.2 Software Arduino

Úkolem Arduina je překládat komunikaci mezi sériovou linkou a sběrnicí I^2C. Pro I^2C sběrnici je Arduino nakonfigurováno jako follower a zpracuje každou přijatou informaci ze strany leader jednotky, tu přeloží do *string* a odešle na sériovou linku. Pokud Arduino detekuje příjem bitu na vstupním zásobníku sériové linky, vyčte vstupní *string* a provede vybrané části kódu.

Pro práci se sérovou linkou není nutné využívat externí knihovny, metody pro ovládání sériové linky jsou součástí základního kódu. Otevření sériového portu probíhá pomocí metody *Serial.begin(),* hodnota v závorce uvádí zvolenou rychlost v baudech. Tato metoda je volána funkci *setup()*, která je vykonána vždy pouze jednou při startu programu. V další části kódu je pak možné s připojením pracovat pomocí sady metod, které jsou k tomu určené. Základní metody pro práci se sériovou linkou jsou: [8]

Serial.avalible()	Vyčte počet bajtů dostupných na sériové lince, tyto data jsou v příchozím zásobníku
Serial.print()	Odeslání dat bez zalomení řádku
Serial.println()	Odeslání dat se zalomením řádku
Serial.readString()	Načtení příchozích dat ve formě string

Komunikace se sběrnicí I^2C funguje trochu odlišně. Aby bylo možné pracovat se sběrnicí I^2C, je nutné použít v programu knihovnu *wire.h,* ta je definována na začátku kódu společně s proměnnými použitými v programu viz Obrázek 9. Proměnné využívané kódem viz Tabulka 3. V části *setup()* se provede základní nastavení rozhraní. Metoda *Wire.begin()* iniciuje knihovnu a připojení k I^2C sběrnici, v závorce je uvedena adresa, které zařízení využívá pro komunikaci na sběrnici. Metody *Wire.onReceive()* a *Wire.onRequest* zaregistrují funkce, které se vyvolají při požadavku na příjem, nebo odeslání dat na sběrnici. Základní metody knihovny *Wire.h* jsou:

Wire.send()	odesílá po sběrnici data v několika typech (value, string, data). Použití v režimu *follower* umožňuje odeslat data na vyžádání jednotkou *leader.* Pro odeslání dat z jednotky *leader* je nutné volat tuto metodu mezi metodami *beginTransmission()* a *endTransmission().*
Wire.read()	vyčítá přijatá data z I^2C

```
#include <Wire.h>
const byte myAddress = 0x27;
int fromF = 0;
String pcRead;
const int enable = 9;
const int a = 10;
const int b = 11;
const int apply = 12;
```

Obrázek 9 - Definice použité knihovny a proměnných

Tabulka 3 - Seznam proměnných použitých v programu Arduino

Proměnná	Význam
myAddress	Adresa zařízení (konstanta)
fromF	Proměnná pro ukládání přijatých dat z Fischertechnik TXT Controller
pcRead	Proměnná pro ukládání dat přijmutích z PC
enable	Pin volby režimu (konstanta)
a	Pin volby boxu (konstanta)
b	Pin volby boxu (konstanta)
apply	Pin pro spuštění sekvence (konstanta)
numBytes	Přijímaná data ve funkci receiveEvent

V části kódu „setup“, viz Obrázek 10, je definováno nastavení digitálních pinů, rychlost baud pro sériové spojení a část kódu pro obsluhu sběrnice I^2C. V té je definováno zahájení I^2C komunikace s přidělenou adresou, a také funkce, která se spouští v případě příchozích dat na sběrnici I^2C. [7] Na konci této části je pak výchozí nastavení výstupních pinů *enable* a *apply* do pozice *LOW*. To zaručuje, že po restartu zařízení bude Fischertechnik TXT Controller vykonávat činnost v automatickém režimu.

```
void setup() {
pinMode(enable, OUTPUT);
pinMode(a, OUTPUT);
pinMode(b, OUTPUT);
pinMode(apply, OUTPUT);
Serial.begin(9600);
Wire.begin(myAddress);
Wire.onReceive(receiveEvent);
Wire.onRequest(requestEvent);
digitalWrite(enable, LOW);
digitalWrite(apply, LOW);
}
```

Obrázek 10 - Funkce setup v kódu Arduino

V hlavní smyčce programu, viz Obrázek 11, probíhá kontrola příchozích dat na sériové lince. Pokud program detekuje příchozí bit, provede načtení celého příchozího *string* pomocí funkce *Serial.readString().* Podle vyčteného *string* se

následně provede nastavení konfigurace výstupních pinů a odeslání potvrzující zprávy do PC. V následující tabulce, viz Tabulka 4, je uveden přehled používaných stringů. Před deaktivací potvrzujícího pinu *apply* je nastavena prodleva 0,5 s tak, aby byl zaručen prostor pro zaregistrování změny jednotkou Fischertechnik TXT Controller.

Tabulka 4 - Seznam použitých proměnných string a jejích význam

String	Význam
0	Manuální třídění vypnuto
1	Manuální třídění zapnuto
Box1	Zvolen Box 1
Box2	Zvolen Box 2
Box3	Zvolen Box 3

```
void loop() {
    while (Serial.available() > 0) {
    pcRead = Serial.readString();
    if (pcRead == "1")
    {
     Serial.println("Manualni trideni zapnuto");
     digitalWrite(enable, HIGH);
    }
    if (pcRead == "0") {
      Serial.println("Manualni trideni vypnuto");
     digitalWrite(enable, LOW);
    }
    if (pcRead == "Box1") {
      Serial.println("Zvolen Box1");
    digitalWrite(enable, HIGH);
    digitalWrite(a, LOW);
    digitalWrite(b, HIGH);
    digitalWrite(apply, HIGH);
    delay(500);
    digitalWrite(apply, LOW);
    }
    if (pcRead == "Box2") {
      Serial.println("Zvolen Box2");
     digitalWrite(enable, HIGH);
    digitalWrite(a, HIGH);
    digitalWrite(b, HIGH);
    digitalWrite(apply, HIGH);
     delay(500);
    digitalWrite(apply, LOW);
    }
    if (pcRead == "Box3") {
      Serial.println("Zvolen Box3");
      digitalWrite(enable, HIGH);
    digitalWrite(a, HIGH);
    digitalWrite(b, LOW);
    digitalWrite(apply, HIGH);
     delay(500);
    digitalWrite(apply, LOW);
    }
  }
}
```

Obrázek 11 - Funkce loop v kódu Arduino

Funkce *receiveEvent* viz Obrázek 12, která byla definovaná v části setup, se stará o zpracovaní příchozích dat ze sběrnice I^2C. Při zjištění příchozích dat provede jejich uložení do proměnné *fromF* a pomocí prvku switch odešle odpovídající zprávu na sériovou linku. Přiřazení je uvedeno v Tabulka 1

```
void receiveEvent(int numBytes) {
  while (Wire.available() > 0) {
    fromF = Wire.read();
    switch (fromF) {
      case 1:
        Serial.println("Puk detekovan");
        break;
      case 2:
        Serial.println("Zahajeni pohybu ramene");
        break;
      case 3:
        Serial.println("Detekovana bila");
        break;
      case 4:
        Serial.println("Detekovana cervena");
        break;
      case 5:
        Serial.println("Detekovana modra");
        break;
      case 6:
        Serial.println("Ulozeni do boxu");
        break;
      case 7:
        Serial.println("Vychozi pozice - konec");
        break;
      case 8:
        Serial.println("Vstupni pozice je prazdna!");
        break;
    }
  }
}
```

Obrázek 12 - Funkce receiveEvent v kódu Arduino

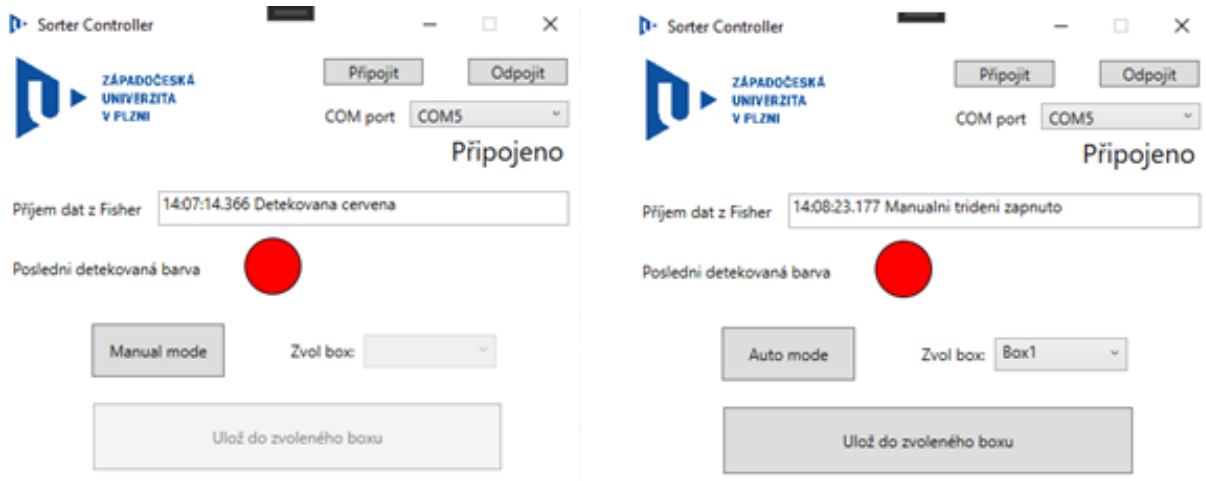
3.3 Aplikace pro počítač

Aplikace byla naprogramována pomocí programovacích jazyků C# (obslužný kód) a WPF (grafické rozhraní aplikace).

Grafický návrh aplikace byl navržen s ohledem na funkčnost a uživatelskou orientaci. V horní pravé části se nachází komunikační rozhraní. Pro výběr COM portů je připraven *ComboBox* s automatický načítáním dostupných portů COM. Nad výběrem COM portu se nachází tlačítka pro připojení a odpojení vybraného COM portu. Status připojení je signalizován pod výběrem COM portu, program signalizuje stavy odpojeno a připojeno.

Pod tímto oddílem jsou umístěny prvky pro čtení dat a ovládání modelu. V poli příjem dat z Fischertechnik TXT Controller se vyčítají informace zasílané jednotkou Fischertechnik TXT Controller. Tyto informace jsou opatřeny časovým razítkem a v souladu se zadáním jsou ukládány do textového souboru *log.txt* v kořenovém adresáři projektu. Pod tímto polem se nachází grafické znázornění poslední detekované barvy. Posledním oddílem je oddíl s ovládacími prvky. Ve výchozím stavu je robot v automatickém režimu a z ovládacího oddílu je aktivní pouze tlačítko *Manual Mode*, ostatní položky jsou uzamčeny. Po stisku tlačítka *Manual mode* se robot a aplikace přepne do manuálního řízení. V tomto okamžiku jsou odemčeny i položky pro volbu boxu a tlačítko pro provedení založení. K volbě boxu je připraven další *ComboBox* s přednastavenými hodnotami. Tlačítko *Ulož do zvoleného boxu* slouží k provedení příkazu, před odesláním příkazu do jednotky Fischertechnik TXT Controller program ověří, zda je vybrán nějaký box, pokud by tomu tak nebylo, odeslání neprovede a zobrazí chybovou hlášku.

V programu byly ošetřeny výjimky v případě požadavku na akci, která není možná. Ukázka programu je na Obrázek 13, vlevo je zobrazen program s připojeným Arduinem v automatickém režimu, vpravo je pak zobrazení programu v manuálním režimu.



Obrázek 13 - Ukázka navrhnuté aplikace Sorter Controller

3.4 Propojení řídící jednotky a mikrokontroleru

K propojení I^2C byl použitý obousměrný převodník logických úrovní 5 V a 3,3 V. Do převodníku jsou zapojeny výstupy SDA a SCL z mikrokontroleru Arduino z 5 V strany a z jednotky Fischertechnik TXT Controller ze strany 3,3 V. Dále je nutné provést propojení společných záporných pólů, pro tento účel má převodník 2 piny GND: Záporné připojení GND je použito zároveň u spínaných logických stavů na úrovni propojení digitálních I/O pinů mezi jednotkami. Pro rozdělení záporného pólu byla použita napěťová část nepájivého pole. Zapojení převodníku již nevyžaduje zapojené piny VCC, jelikož má každá jednotka vlastní napájení. Napěťové strany jsou rozlišeny písmeny A (5 V) a B (3,3 V). Propojení

jednotlivých pinů převodníku je uvedeno v Tabulka 6. Digitální piny Arduina a TXT Controlleru jsou propojeny napřímo dle Tabulka 5.

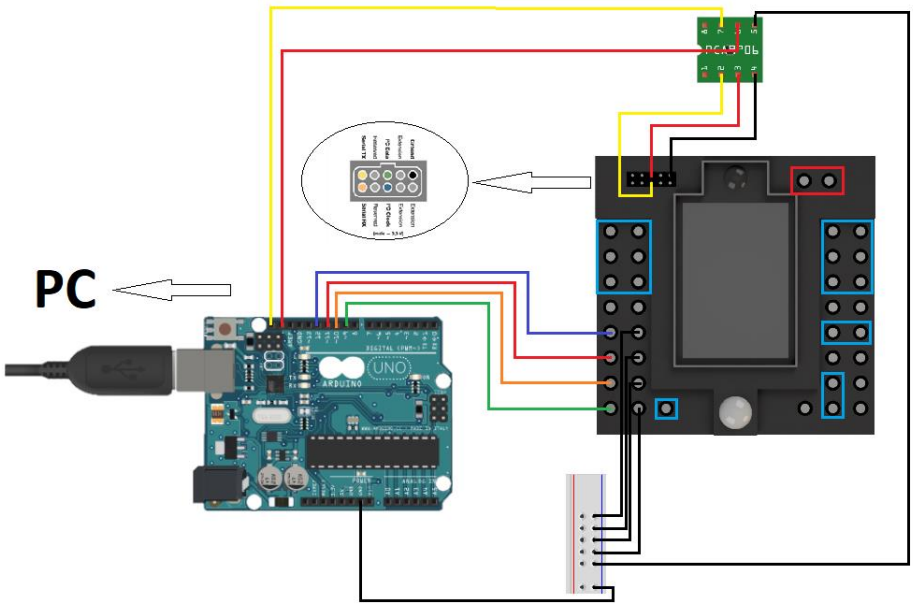
Tabulka 5 - Přiřazení digitálních pinů

FUNKCE	ARDUINO PIN	FISCHERTECHNIK TXT CONTROLLER PIN
ZMĚNA REŽIMU	9	I8
SPUŠTĚNÍ SEKVENCE	10	I7
ADRESA ZVOLENÉHO BOXU	11	I6
	12	I5

Diagram zapojení finálního řešení je znázorněn na Obrázek 14. K vytvoření diagramu byla použita z části vlastní grafika a z části grafika z open source softwaru Fritzing [9]. Piny jednotky Fischertechnik TXT Controller použité pro připojení robota jsou označeny modrým rámečkem. Červeným rámečkem jsou označeny napájecí piny jednotky TXT Controller. K napájení TXT Controlleru je použitých 9 V zdroj Fischertechnik, Arduino je napájeno pomocí USB portu (5 V), kterým je zároveň připojeno k PC.

Tabulka 6 - Zapojení I2C převodníku

Arduino	I^2C převodník		Fischertechnik TXT Controller
GND	AGND	BGND	GND
SDA	ASDA	BSDA	SDA
SCL	ASCL	BSCL	SCL
-	AVCC	BVCC	-



Obrázek 14 - Diagram finálního zapojení

3.5 Popis funkce výsledného řešení

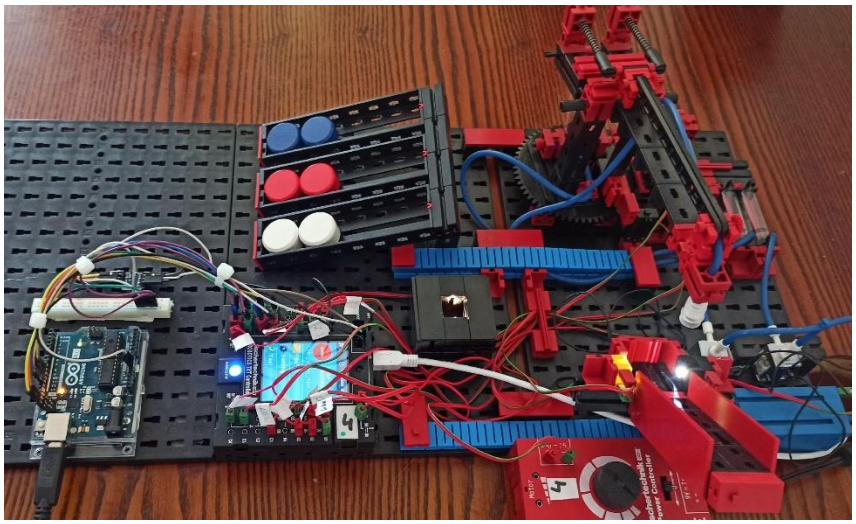
Po zapnutí průmyslového modelu Fischertechnik a Arduina je možné spustit program Sorter Controller. V programu se nejprve provede volba COM portu,

na kterém je připojené Arduino, následně je možné připojit sériovou linku pomocí tlačítka *Připojit.* Úspěšné připojení je signalizováno změnou stavu pod *comboBox* pro výběr COM portů.

V tomto režimu čeká model robotického zakladače na vložení barevného puku do zásobníku. Při detekci puku provede model automatické zatřídění, tak jak je popsané v kap. 2. Tento režim probíhá zcela automaticky a odesílá požadované informace o jednotlivých stavech do PC, kde jsou tyto záznamy zapisovány do textového souboru *log.txt.*

Při stisknutí tlačítka *Manual mode*, dojde k zapnutí manuálního režimu. V tomto režimu přestane robot automaticky třídit a čeká na pokyny uživatele. Při obdržení pokynu na zatřídění, provede robot kontrolu, zdali je na vstupní pozici umístěn puk, pokud ano, provede jeho zatřídění, bez určování barvy, přímo do zvoleného boxu. Pokud robot při požadavku na manuální založení zjistí, že na vstupní pozici se nenachází puk, odešle chybovou hlášku, která se zobrazí uživateli programu na PC.

Pro ukončení práce slouží tlačítko *Odpojit*. To se postará o nastavení robota zpět do automatického režimu, odpojení sériové linky a výchozího nastavení prvků grafického rozhraní. Model celého zařízení je zobrazen na Obrázek 15.



Obrázek 15 - Finální řešení projektu

4 Závěr a hodnocení

Práce se zabývá realizací průmyslového modelu FischerTechnik s počítačem při zachování řídícího kódu řídící jednotky Fischertechnik v grafickém programovacím jazyce. Proto bylo navrženo řešení, které využívá jako mezičlánek mikrokontrolér Arduino – ten slouží jako překládací prvek mezi PC a modelem Fischertechnik. Během řešení byly otestovány možnosti a hranice použití I^2C komunikace s modelem Fischertechnik. Při návrhu funkčního řešení bylo nutné vyřešit několik problémů. Tyto problémy se podařilo vyřešit drobnými úpravami výsledného řešení.

Nejvýznamnější změnou je použití digitálních I/O pinů pro komunikaci ze strany Arduino do jednotky TXT Controller. Ačkoliv se podařilo najít funkční řešení,

jedná se stále o značně omezující problém. Řešení dodatečné komunikace pomocí digitálních pinů je omezeno celkovým počtem pinů na jednotce TXT Controller. Odstranění tohoto omezení bude předmětem dalšího vývoje.

Výsledné řešení projektu vykazuje vysokou spolehlivost provozu. Během testování se projevilo minimum problémů. Převážně se jednalo o problémy s komunikací na sběrnici I^2C, které se občas projevily během zapínání zařízení Arduino a modelu Fischertechnik. K odstranění problému stačilo provést reset mikrokontroleru Arduino. Jiné problémy se za dobu testování neprojevily.

Výsledky této práce budou uplatněny nejen pro vzdělávací účely, ale mají využití při návrhu modelů digitálních podniků, či podobném nasazení přímo v průmyslu. Použití STEM modelu Fischertechnik a Arduina umožňuje využít jednoduchost sériové komunikace a data získávaná z modelu libovolně zpracovávat. To například usnadňuje propojení fyzického modelu Fischertechnik a virtuálního modelu např. v Plant Simulation.

Navrhnuté řešení spadá do konceptu STEM a představuje užitečné znalosti v oblasti průmyslového inženýrství. Výsledky této práce mohou být použity při optimalizaci výrobních, vzdělávacích nebo logistických procesů uvnitř podniku. Vzhledem k širokým možnostem používaných technologiích, může být práce dále rozvíjena a zlepšována. Další rozvoj může být například propojení fyzického průmyslového modelu a jeho virtuálního dvojčete, to by pak sloužilo jako vzdělávací pomůcka při školení a rozvoji zaměstnanců. Virtuální a fyzické modely mohou sloužit pro porovnání a rozhodování o volených technologických operacích. V průmyslových provozech je možné tyto výsledky využít pro sledování a optimalizaci provozních parametrů výrobních procesů.

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.

Použité zdroje

[1] MALAGA, M. a ULRYCH, Z. *Koncept STEM se zaměřením na problematiku Industry 4.0.* Plzeň : Západočeská univerzita v Plzni, 2019.

[2] TXT Controller - fischertechnik. *Fishertechnik.* [Online] Fishertechnik. [Citace: 2021. září 20.] https://www.fischertechnik.de/en/products/playing/robotics/522429-txt-controller.

[3]. Arduino Store. *Arduino Uno Rev3.* [Online] Arduino, 2021. [Citace: 12. Duben 2021.] https://store.arduino.cc/products/arduino-uno-rev3/.

[4] ŘÍHA, P. *Slovník počítačové informatiky: výkladový slovník pro práci s informacemi: hardware a software včetně počítačových sítí, internetu a mobilních technologiích.* Ostrava : Montanex, 2002. ISBN 80-7225-083-3.

[5] I2C - What`s That? . *I2C Bus.* [Online] [Citace: 8. Duben 2021.] http://www.i2c-bus.org.

[6] How to Connect an I²C Device to the Robotics TXT? *Rei Vilo's fischertechnik Corner.* [Online] [Citace: 3. Březen 2021.] https://reivilofischertechnik.weebly.com/how-to-connect-an-isup2c-device-to-the-robotics-txt.html.

[7] Arduino - Wire. *Arduino - Home.* [Online] Arduino. [Citace: 12. Duben 2021.] https://www.arduino.cc/en/reference/wire.

[8] Serial - Arduino Reference. *Arduino - Home.* [Online] Arduino. [Citace: 4. Duben 2021.] https://www.arduino.cc/reference/en/language/functions/communication/serial/.

[9] Fritzing. *Fritzing.* [Online] [Citace: 12. Duben 2021.] http://fritzing.org.