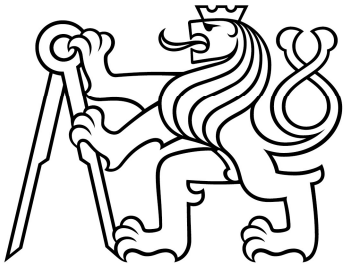
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA STROJNÍ

ÚSTAV PŘÍSTROJOVÉ A ŘÍDICÍ TECHNIKY



# DIPLOMOVÁ PRÁCE

AUTOR: Bc. Jan Janovský

STUDIJNÍ PROGRAM: Strojní inženýrství

STUDIJNÍ OBOR: Přístrojová a řídicí technika

VEDOUCÍ PRÁCE: Ing. Marie Martinásková, Ph.D.

PRAHA 2020

# ZADÁNÍ DIPLOMOVÉ PRÁCE

## I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Janovský** Jméno: **Jan** Osobní číslo: **459547**

Fakulta/ústav: **Fakulta strojní**

Zadávající katedra/ústav: **Ústav přístrojové a řídící techniky**

Studijní program: **Strojní inženýrství**

Studijní obor: **Přístrojová a řídicí technika**

## II. ÚDAJE K DIPLOMOVÉ PRÁCI

Název diplomové práce:

**Návrh systému domácího kina s pohyblivou sedačkou s využitím FMT (Festo Motion Terminal)**

Název diplomové práce anglicky:

**Design of home theater system with moving seat using FMT (Festo Motion Terminal)**

Pokyny pro vypracování:

1. Prostudujte historii a aplikace pneumatických ventilových terminálů od počátku vývoje až po ventilový terminál pro digitální pneumatiku FMT (Festo Motion Terminal)
2. Prostudujte podrobně funkci FMT terminálu pomocí aplikačních příkladů zrealizovaných v laboratoři
3. Navrhněte systém kina s pohyblivou sedačkou pro možné domácí použití

Seznam doporučené literatury:

[1] Beneš, P. a A. Mykiska. Pneumatické řídící systémy. Nadstavbový seminář z pneumatiky. Učebnice Festo Didactic pro postgraduální studium. Praha: ČVUT v Praze, 1989. ISBN 80-01-00130-X.
[2] Martinásková, M. a M. Šmejkal. Řízení programovatelnými automaty III: Softwarové vybavení. Praha: ČVUT v Praze, 2003. ISBN 80-0102804-6.
[3] Haumer, Zdeněk. Pneumatické ventilové a instalační terminály. Automa: časopis pro automatizační techniku [online]. Děčín, 2012(8-9), 3. ISSN 1210-9592.
[4] Haumer, Zdeněk. Ventilové a elektrické terminály „na míru". Automa: časopis pro automatizační techniku [online]. Děčín, 2004(11). ISSN 1210-9592.
[5] Valve and valve sensor/terminals with field bus connection: New ideas for faster installation. Festo Pneumatic, 1991.
[6] FMT – system manual

Jméno a pracoviště vedoucí(ho) diplomové práce:

**Ing. Marie Martinásková, Ph.D., U12110.3**

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) diplomové práce:

Datum zadání diplomové práce: **30.10.2020** Termín odevzdání diplomové práce: **20.01.2021**

Platnost zadání diplomové práce: _____

| Ing. Marie Martinásková, Ph.D.<br>podpis vedoucí(ho) práce | podpis vedoucí(ho) ústavu/katedry | prof. Ing. Michael Valášek, DrSc.<br>podpis děkana(ky) |
|---|---|---|

## III. PŘEVZETÍ ZADÁNÍ

Diplomant bere na vědomí, že je povinen vypracovat diplomovou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v diplomové práci.

| Datum převzetí zadání | Podpis studenta |
|---|---|

# Prohlášení

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně s tím, že její výsledky mohou být dále použity podle uvážení vedoucího diplomové práce jako jejího spoluautora. Souhlasím také s případnou publikací výsledků diplomové práce nebo její podstatné části, pokud budu uveden jako její spoluautor.

V Praze dne: …….

……
podpis

# Abstrakt

Diplomová práce „Návrh systému domácího kina s pohyblivou sedačkou s využitím FMT (Festo Motion Terminal)“ je rozdělena na dvě hlavní části.

Teoretická část práce se nejprve zabývá historií vývoje pneumatických ventilových terminálů od samostatně stojících rozváděčů, přes ventilové bloky a několik typů ventilových terminálů až po ventilový terminál pro digitální pneumatiku Festo Motion Terminal. V každé fázi vývoje je také uveden příklad aplikace v dnešním průmyslu. Dále podrobně prozkoumává funkce tohoto terminálu pomocí existujícího aplikačního příkladu zrealizovaného v laboratoři.

Praktická část se již věnuje návrhu systému domácího kina s pohyblivou sedačkou. Nejdříve se stručně zabývá vybíráním vhodných komponent pro tento projekt a návrhu konstrukce. V druhé části je představen použitý software. Dále je již podrobněji popsán princip jednotlivých funkcí navrženého systému domácího kina s pohyblivou sedačkou a možností užití jak z hlediska uživatelského, tak vývojářského. V následující části je sepsán návod k bezproblémové instalaci celého projektu a jeho spuštění. Na závěr práce nechybí zhodnocení dosaženého výsledku, aktuálních nedostatků návrhu, jejich možných řešení a možností navazujícího vývoje.

# Klíčová slova

ventilový terminál, vývoj, sedadlo, kino, pohyb, Festo Motion Terminal, Node-RED, MovTheater

# Abstract

The diploma thesis "Design of home theater system with moving seat using FMT (Festo Motion Terminal)" is divided into two main parts.

The theoretical part first dedicates to history of development pneumatic valve terminals starting from standalone valves, through valve blocks and several types of valve terminals ending at terminal for digital pneumatics Festo Motion Terminal. In each developmental phase, there is also listed one example of application in today's industry. Next, it explores in detail the functions of this terminal thanks to existing application example, which is realized in the laboratory.

The practical part dedicates to the design of home theater system with moving seat. First, it briefly deals with the choice of suitable components for this project and the construction design. In the second part, it introduces used software. Next, it describes in more detail principles of individual functions of the designed home theater system with moving seat and possibilities of usage from users' and developers' points of view. There is included a manual for smooth installation of the whole project and its launch. At the end of the thesis there is not missing the conclusion of achieved result, actual shortcomings of the design and their possible solutions and future development options.

# Keywords

valve terminal, development, seat, theater, motion, Festo Motion Terminal, Node-RED, MovTheater

# Poděkování

Rád bych poděkoval vedoucí mé práce, Ing. Marii Martináskové Ph.D., která mi ochotně radila při plnění stanovených úkolů a také společnosti Festo SE & Co. KG, která mi poskytla potřebnou dokumentaci k tématu teoretické části mé práce. Nejvíce bych chtěl poděkovat svým nejbližším, kteří mě nejen v této závěrečné fázi, ale po celou dobu mých studií bezmezně podporovali.

# Obsah

# Úvod

Pneumatické ventilové terminály patří mezi nedílnou součást průmyslového oboru pneumatiky. Jedná se o komplexní zařízení, které kombinují pneumatické ventily, elektrická připojení, komunikaci, řízení a další prvky. Pro pochopení jejich funkce a důvodu popularity je nejjednodušší dozvědět se některé informace z jejich vzniku a historického vývoje.

V první kapitole teoretické části práce jsem nejdříve vypracoval rešerši na téma historického vývoje ventilových terminálů od počátků ovládání pneumatických pohonů samostatnými rozváděči přes ventilové bloky a terminály až k nejnovějšímu trendu tzv. digitální pneumatiky, zastoupené produktem Festo Motion Terminal. V této kapitole si lze všimnout jasných vývojových trendů, kterých se výrobci neustále snažily a stále snaží držet nejen v pneumatice, ale i v celém průmyslu: miniaturizace, kompatibilita, jednoduchost a univerzálnost. Zpracovaná rešerše přehledně ukazuje vývojové kroky v historii ventilových terminálů a také možnosti dnešních průmyslových aplikací každé z těchto vývojových fází.

V druhé kapitole teoretické části nejvíce kladu důraz na pochopení funkcí a možností Festo Motion Terminal a uvádím principy jeho programování. Zaměřil jsem se také na aplikaci FMT ve školní laboratoři, kde jsem pečlivě prostudoval jednotlivé programovací části. Kapitola jistě nepopisuje všechny dostupné možnosti tohoto terminálu, ale spíše slouží k pochopení základních principů programování nejen pro mne, ale hlavně pro budoucí studenty Ústavu přístrojové a řídicí techniky, které toto téma bude zajímat.

V praktické části se již zabývám návrhem systému domácího kina s použitím Festo Motion Terminal. Nejdříve představuji můj konstrukční návrh, ze kterého vychází i potřebné komponenty k možné realizaci tohoto projektu. Poté již popisuji výběr software, který také stručně představím a který použiji k celkové realizaci projektu. V hlavní části praktické části detailně prezentuji hlavní funkce systému, a to jak z uživatelského, tak také programátorského pohledu. V těchto úsecích také využívám nabyté znalosti z předchozí teoretické kapitoly. Po představení hlavních funkcí systému jsem do práce umístil sepsaný a vyzkoušený návod pro bezproblémovou instalaci celého systému. Na závěr uvádím výsledek mé práce s několika nedostatky a jejich možnými řešeními a také možnosti navazujícího vývoje celého projektu.

# I. Teoretická část

## 1. Historický vývoj ventilových terminálů

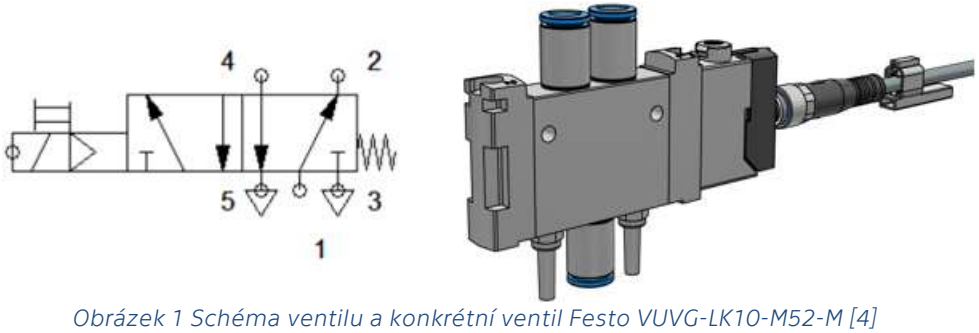
Tato část pojednává o historii pneumatických ventilových terminálů od počátků vývoje až po novou éru digitální pneumatiky, zastoupenou Festo Motion Terminal. Dále se zaměřuje na aplikační oblasti ve všech fázích vývoje ventilových terminálů v dnešním průmyslu.

### Samostatné rozváděče

Pneumatické rozváděče, jinak také nazývány ventily, jsou zařízení užívané k řízení běhu pneumatických pracovních prvků, například motorů. [1] Stlačený vzduch je přiváděn skrz vstupní port do těla rozváděče, kde jsou jeho směr toku a popřípadě i vlastnosti upraveny dle zadání a stavu řídicího signálu. Konstrukce těl pneumatických rozváděčů jsou přizpůsobeny typu připojených motorů, aplikaci a způsobu řízení. [2] Pokud je rozvaděč ovládán jednostranně s návratem do základní polohy (díky mechanické pružině například), nazývá se monostabilní. Bistabilní rozváděče jsou ovládány oboustranně a pro změnu polohy těla je nutné vždy přivézt řídicí signál na opačnou stranu, jinak setrvávají v aktivní poloze. [2] [3] Pneumatické rozváděče mohou být ovládány manuálně (tlačítkem), mechanicky (kladkou), pneumaticky (přívodem stlačeného vzduchu na řídicí porty) či elektricky (přívodem napětí na řídicí cívku). [1] Upravený stlačený vzduch je poté přiveden na výstupní porty rozváděče. Názvosloví rozváděčů je tvořeno popořadě, dle následujícího postupu: [2] [3]

1. počet portů / počet poloh (například 5/2, čti „5 na 2“)
2. monostabilní či bistabilní
3. typ ovládání (manuálně, mechanicky, …)
4. doplňková informace (pilotní řízení, stav v základní poloze, …)

Konkrétním příkladem může být samostatně stojící ventil 5/2, monostabilní, elektricky ovládaný s pomocným ručním ovládáním s aretací, vhodný pro ovládání dvojčinného pohonu.




*Obrázek 1 Schéma ventilu a konkrétní ventil Festo VUVG-LK10-M52-M [4]*

Takto samostatně stojící ventily a rozvaděče s sebou při aplikaci nesly nespočet nevýhod. První z nich se vyskytovala již při konstruování stroje. Ve většině aplikací je nutné pro každý pneumatický pohon, který je umístěn ve stroji, použít také zařízení pro řízení jeho pohybu, zastoupený právě ventily a rozvaděči. Čím více je tedy použito pohonů, tím více ventilů je nutné do stroje umístit, což klade velké nároky na velikost prostoru. Dalším problémem je složitost nákupu položek při použití většího množství ventilů a jejich uskladnění. S rostoucím počtem takovýchto samostatných členů dále roste velikost rozvodné sítě stlačeného vzduchu ve stroji a tím i šance chybné instalace hadic a případně i vodičů při použití elektricky ovládaných ventilů. S tímto bodem také dále souvisí složitá údržba celého zdroje. [5] [6]

I přes nevýhody popsané výše, samostatně stojící ventily a rozvaděče nachází uplatnění i dnes. Jedná se tedy hlavně o jednoduché aplikace, kde se tyto členy používají pro ovládání systému obsluhou jako ruční tlačítka, pedály či páky. [3] Dále je výhodné je použít v systémech s malým počtem pneumatických pohonů, kde mohou být rozvaděče ovládané pneumaticky či elektricky či ve speciálních případech, kdy je nutné umístit ventil přímo na stanici, mimo ventilový blok či terminál.

Konkrétními příklady uplatnění mohou být pneumatické a hydraulicko-pneumatické lisy. Ty mohou být ovládány buď čistě pneumaticky, nebo elektropneumaticky. Například výrobek firmy mäder pressen, pneumatický kolenopákový lis řady XL – NP, který využívá elektricky ovládaný ventilový rozváděč k řízení pohybu lisovací hlavy. [7]

*Obrázek 2 Pneumatický kolenopákový lis XL – NP [7]*

Dalším konkrétním příkladem může sloužit umístění samostatného rozváděče na svářecí stanici. Na tento rozváděč je připojen pneumatický pohon, který zajišťuje posun svářecí hlavy do pracovní polohy. Důvodem použití samostatného ventilu je zvýšení průtoku stlačeného vzduchu přímo do pohonu, který nese těžké břemeno v podobě svářecí hlavy, a tím i zajištění rychlejšího chodu jak svářečky, tak celého stroje. Samostatný bistabilní rozváděč je ovládán pneumatickými signály z ventilového terminálu, ale jeho vstupní port je napojen přímo na zdroj stlačeného vzduchu stroje.

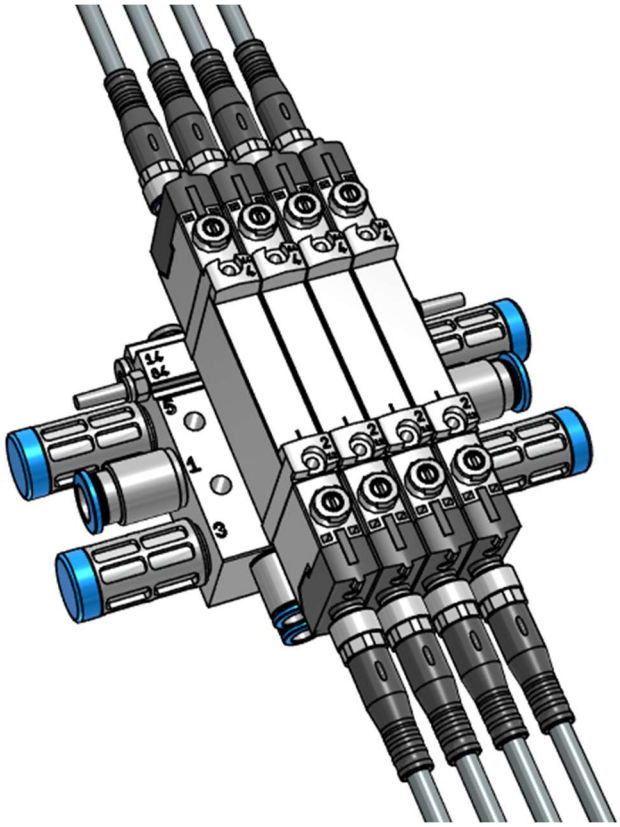
*Obrázek 3 Svářecí stanice [8]*

## Ventilové bloky

Ventilové bloky znamenají jakýsi první milník v historii vývoje používání pneumatických ventilových rozváděčů směrem k jejich integraci do ventilových terminálů. [9] U každého ze samostatně stojících rozváděčů je nutné řešit mechanické upevnění, přívod stlačeného vzduchu a také přívod řídicích signálů. Nejprve se tyto rozváděče začínají objevovat ve strojích na různých lištách a blocích, které shlukují jednotlivé ventily do skupin. [9] Reakcí na tento trend přicházejí společnosti, které vyvíjí pneumatické systémy, na trh s ventilovými bloky a bateriemi, které řeší jak mechanické problémy upevnění, tak jednotný přívod stlačeného vzduchu do všech rozváděčů umístěných na ventilovém bloku jedinou hadicí. Po úspěchu s jednotným přívodem

vzduchu se objevují také bloky, které implementují i společný odfuk vzduchu ze všech umístěných ventilů. [6] [10]

Konkrétním příkladem ventilového bloku z aktuální nabídky může být ventilový blok společnosti Festo VTUG se samostatným elektrickým přívodem pro 4 bistabilní ventily VUVG. [4]



*Obrázek 4 Ventilový blok Festo VTUG se samostatným elektrickým přívodem [4]*

Tato vylepšení s sebou nesla řadu výhod ve všech fázích použití. V první řadě ulehčila práci konstruktérům při řešení problémů s umístěním nejen samotných ventilů, ale také rozvodů stlačeného vzduchu. Snad největší usnadnění práce a ušetření času přinesla do fáze montáže, která se těmito kroky značně zpřehlednila a tím se i zmenšila šance pro chybnou instalaci. Přehlednější zapojení také zrychlovalo diagnostiku případných závad. Elektrická část však stále zůstala řešená jednotlivě pro každý rozváděč. Cívky jednotlivých elektropneumaticky ovládaných ventilů byly tím pádem stále ještě ovládány skrze jednotlivé vodiče k nim přivedené. [10] [11]

V současné době se kvůli neintegrované elektrické části ventilové bloky často nevyužívají. Uplatnění nachází v jednoduchých strojích s malým počtem ventilů a při snaze co nejvíce snížit výrobní náklady stroje. Někteří výrobci však pod pojmem „ventilový blok" uvádějí i výrobky s integrovanou elektrickou částí s použitím například vícepólového připojení či dokonce ovládání jednotlivých ventilů skrze průmyslové

komunikační protokoly. [12] Společnosti nabízející ventilové bloky s nimi zároveň v dnešní době nabízejí i varianty s jednotnou elektrickou částí, které vytváří mnohem ekonomičtější možnosti, zejména ve složitějších aplikacích. [4]

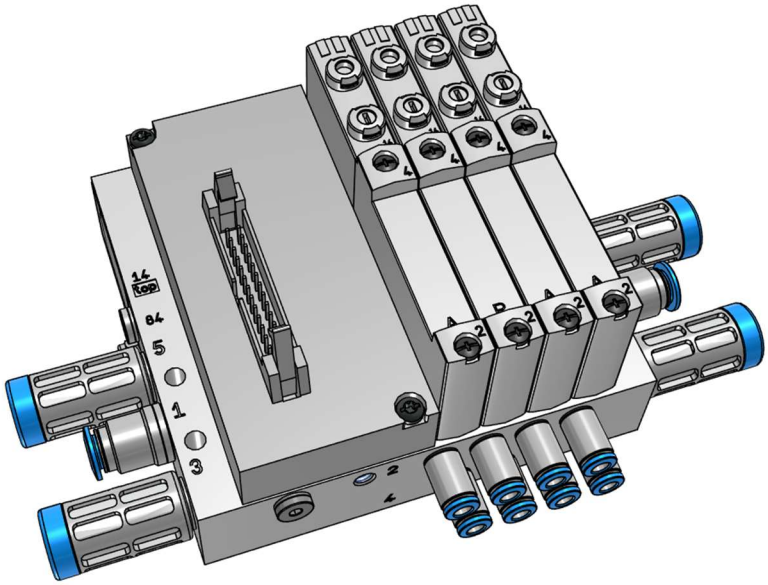
Konkrétním příkladem ventilového bloku může sloužit přibližně 25 let starý stroj na montáž těsnících U-manžet do čerpadel palivových nádrží v automobilovém průmyslu. Na obrázku lze vidět 2 ventilové bloky s instalovanými rozváděči, které jsou ovládány elektricky.

*Obrázek 5 Ventilové bloky montážního stroje U-manžet [8]*

## Ventilové terminály

První zmínky pojmu „ventilový terminál" se začaly objevovat na přelomu 80. a 90. let minulého století. V této době došlo ve vývoji k integraci elektrické části. Elektrická lišta spojovala kontakty ventilových cívek uvnitř bloku. Přístup k těmto kontaktům zajišťovalo vícepólové připojení. [5] [9] [10]

Konkrétním příkladem prvního typu ventilového terminálu z aktuální nabídky může být ventilový terminál společnosti Festo VTUG s připojením vícepólovým konektorem pro 4 ventily VUVG. [4]



*Obrázek 6 Ventilový terminál Festo VTUG s připojením vícepólovým konektorem [4]*

Tento krok znamenal razantní zmenšení počtu kabelů a tím i zjednodušil a ještě více zpřehlednil montáž ventilového terminálu. Problémová však stále zůstávala elektrická montáž jednotlivých vodičů k řídicímu zařízení, ve většině případů k PLC. [11] Výrobci ventilových terminálů nabízely pouze omezený počet typů vícepólových konektorů. Tato skutečnost v některých případech způsobovala přebytek nevyužitých kontaktů kabelu.

V dnešním průmyslu se tyto ventilové terminály s vícepólovým připojením vyskytují v jednodušších strojích či jednotlivých stanicích při šetření nákladů. Umísťují se především na zařízení, které disponující programovatelnými automaty bez komunikačních modulů. Jednotlivé kontakty vícepólového kabelu jsou tedy na straně PLC připojeny v rozvaděči přímo na jeho výstupy.

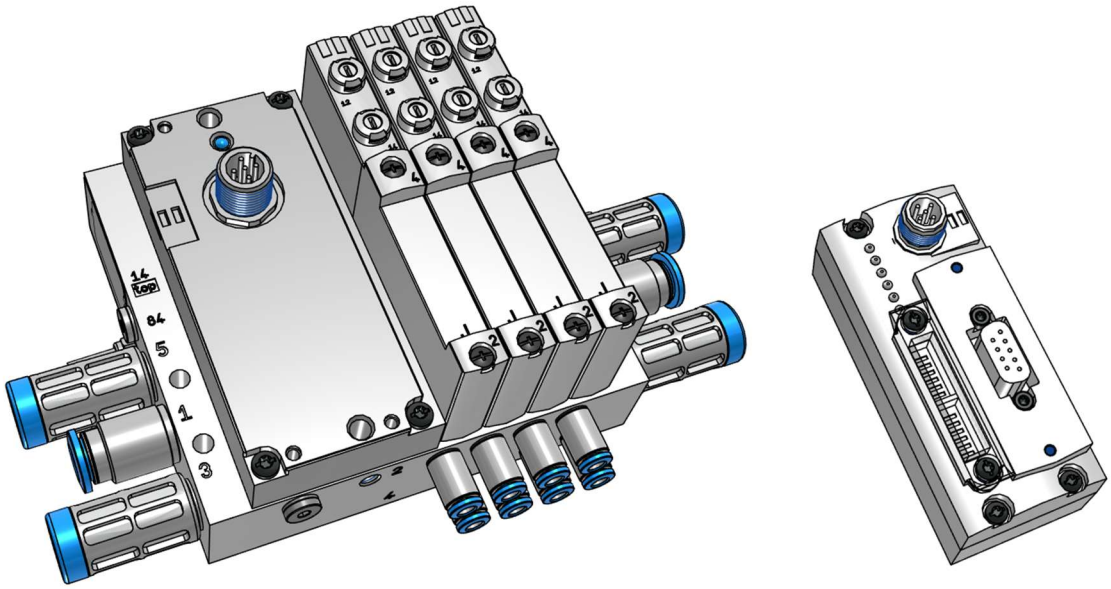
Konkrétním případem použití ventilového terminálu s vícepólovým připojením může být jeho umístění na stanici balící linky. Každá stanice se skládá ze dvou vibračních podavačů, ze kterých díly putují do počítacího mechanismu. Tato část stroje se skládá ze tří klapek, ovládaných pneumatickými pohony. Tyto pohony jsou napojeny na ventilový terminál. Každá z těchto stanic disponuje vlastním PLC umístěným v rozvaděči, které jsou dále napojeny na hlavní řídicí člen. Pro ušetření nákladů jsou tyto PLC vybaveny pouze modulem s několika vstupy/výstupy, na které jsou napojeny kontakty vícepólového kabelu, vedoucího z ventilového terminálu.

*Obrázek 7 Stanice balící linky [8]*

## Instalační ventilové terminály

Připojení pomocí vícepólového konektoru znamenalo sice ušetření kabeláže při instalaci ventilového terminálu, ale stále bylo potřeba pro spínání cívek ventilů mnoho vodičů uvnitř spojovacího kabelu. V reakci na tuto skutečnost se již několik let po první integraci elektrické části začaly používat k připojení různé průmyslové sběrnice. [9] Přidaný komunikační modul se stará o dekódování signálů vysílané řídicím systémem a dle přijaté instrukce spíná jednotlivé adresy cívek. [5] Mezi podporované průmyslové komunikační protokoly té doby patřily například: Profibus DP, Profibus FMS, CANopen a DeviceNet. Ty zajišťovaly velkou kompatibilitu s tehdejšími největšími výrobci řídicích systémů. [13] Později se inteligentní komunikace přesunula i do vnitřku terminálu v podobě interních komunikačních sběrnic. Díky nim bylo možné umístit do terminálu i další komponenty, například čidla tlaku a proporcionální ventily. [14]

Konkrétním příkladem instalačního ventilového terminálu z aktuální nabídky je opět instalační ventilový terminál Festo VTUG s připojením na síť. Ten v základní konfiguraci obsahuje modul s rozhraním I-Port pro připojení síťového modulu CTEU s vybraným komunikačním protokolem. [4] [5]



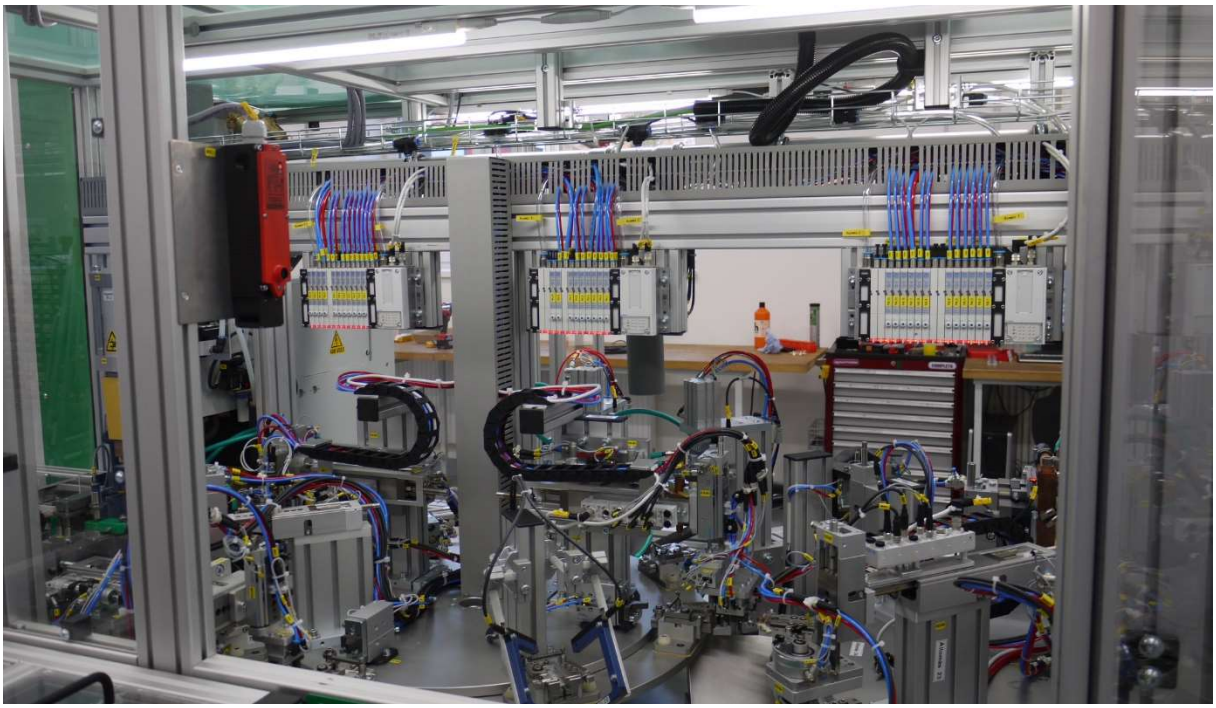
*Obrázek 8 Instalační ventilový terminál Festo VTUG a modul CTEU Profibus [4]*

Implementace komunikačních protokolů přináší mnoho výhod jak pro zákazníky, tak pro výrobce ventilových terminálů. Na první pohled zřetelnou změnou je snížení počtu vodičů vedoucích z terminálu. Tato výhoda se však promítá i do samotného ovládání cívek uvnitř zařízení, jelikož každý připojený ventil „vlastní" svoji adresu a jeho cívky již nejsou fyzicky napojeny na jednotlivé kontakty konektoru a díky tomu není nutné měnit připojovací komunikační techniku pro rozdílné počty ventilů. Tím je možné zobecnit a tím i zlevnit výrobu celého instalačního ventilového terminálu díky sériovosti výrobků. Další výhodou je možná implementace jiných komponent, například čidel tlaku, modulů vstupů a výstupů, proporcionálních redukčních ventilů a dalších. [5] [9] [15] Komunikační protokoly také poskytují pokrokovou diagnostiku zařízení. Díky použití sběrnic se také velmi zlehčuje tvorba elektrické dokumentace. [5]

Ventilové terminály s tímto způsobem komunikace byly nejdříve používány ve velkých strojích. S rostoucí složitostí všech zařízení si však později našly cestu i do menších strojů. [9] V dnešní době bývá nejvýhodnější použití instalačních ventilových terminálů v decentralizovaných systémech, kdy je zařízení připojeno na průmyslovou sběrnici a řízeno na dálku z PLC, umístěného například v rozvaděči. Terminály jsou upevněny blízko pracovních stanic pro ušetření hadic a kabelů čidel. V menších strojích jsou hlavně implementovány ke zjednodušení elektrické instalace použitím sběrnice. [5] [16]

Konkrétním příkladem může být použití instalačních ventilových terminálů ve stroji, který montuje magnetické členy do proudových chráničů. Na obrázku níže lze vidět 3 instalační ventilové terminály, které disponují komunikačním rozhraním IO-Link. Jedná

se o sériovou sběrnici, založenou na komunikaci z bodu do bodu s modelem master/slave. [17] Každý z těchto terminálů je připojen na ventilový terminál typu master, který dále předává informace do hlavního PLC umístěného v rozvaděči stroje. Kromě ovládání pohonů v přilehlých stanicích tyto instalační ventilové terminály také slouží ke sběru dat z blízkých slučovačů, připojenými pomocí komunikační sběrnice IO-Link.

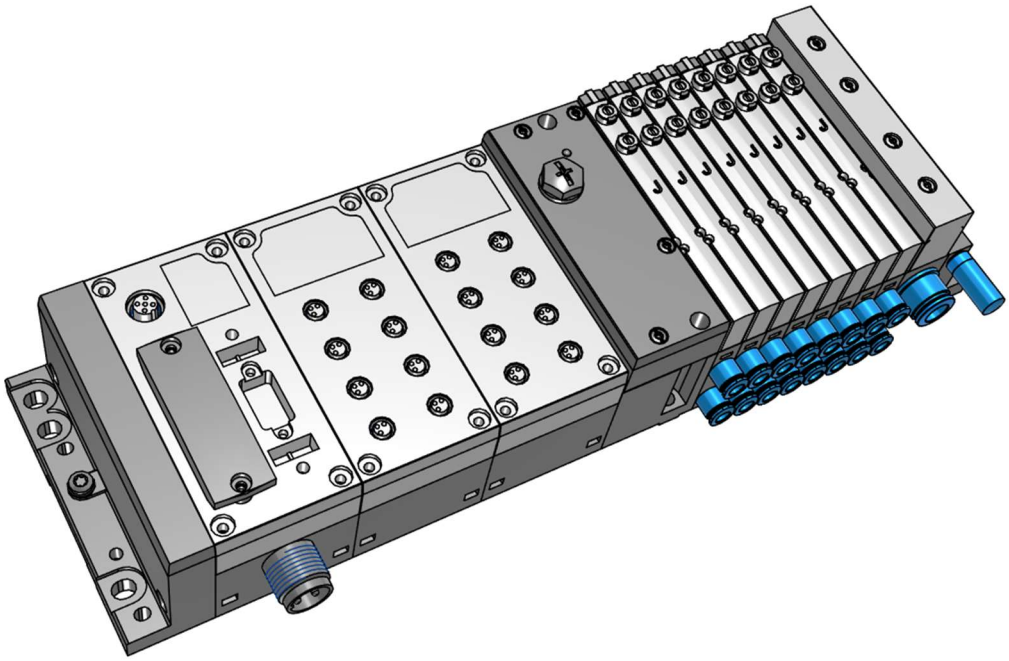


*Obrázek 9 Stroj pro montáž magnetických členů do proudových chráničů [8]*

## Programovatelné instalační ventilové terminály

Pomyslným nejvyšším stupněm klasických pneumatických ventilových terminálů jsou programovatelné instalační ventilové terminály. [10] Ty spojují všechny výhody předchozího typu terminálu s plnohodnotným integrovaným řídicím členem v podobě PLC přímo v zařízení. Tím se může stát daný terminál součástí nadřazeného systému či dokonce celé zařízení přímo řídit. Integrovaný automat navíc dokáže obsluhovat velkou škálu různých komponent, ať už umístěných přímo na terminálu či mimo něj, připojených skrze komunikační moduly. Takto lze kompletně centralizovat celý systém. [16] Vývoj těchto terminálů stále pokračuje směrem k implementaci dalších elektrických funkcí, například řízení elektrických pohonů jako jsou stejnosměrné motory, krokové motory či střídavé servomotory. Trendem dnešní doby je propojení běžné komunikační techniky s tou průmyslovou pomocí tzv. průmyslového Ethernetu. S tímto trendem lze zaznamenat i rozvoj komunikačních protokolů přenášející technologické informace v této síti nové generace. V některých případech terminál dokonce neobsahuje žádné pneumatické ventily a tvoří tzv. instalační elektrický terminál. [9] [15] [14]

Rozmanitost programovatelných instalačních ventilových terminálů se nachází na velmi vysoké úrovni. Příkladem takovéhoto terminálu může být tato konfigurace Festo CPX/MPA-L terminálu s integrovaným PLC v podobě CPX-CEC, 2 moduly vstupů pro koncová čidla motorů a 8 bistabilními ventily se společným přívodem a odfukem vzduchu vpravo. [4]

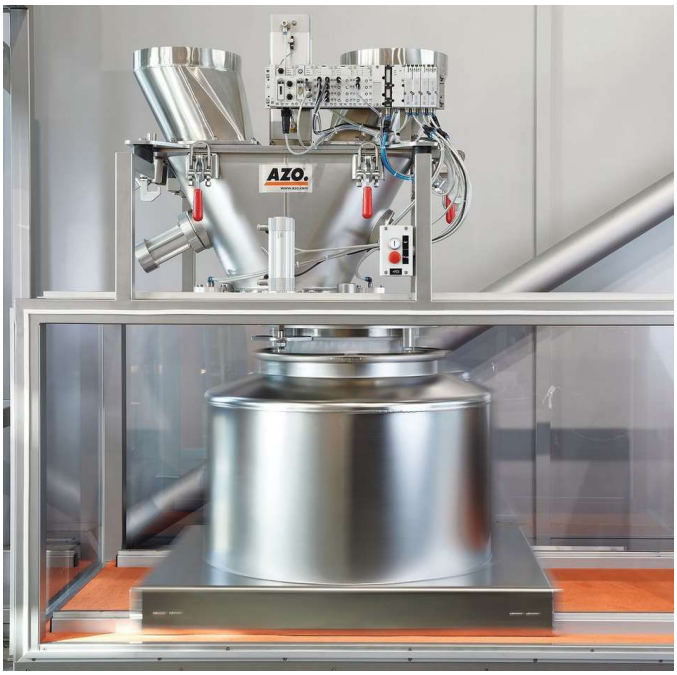


*Obrázek 10 Programovatelný instalační ventilový terminál MPA-L [4]*

Výhodou tohoto typu terminálů je jistě jejich soběstačnost v oblasti řízení a možnostech centralizovat celý systém. Díky vysoké modularitě je možné řídit celý proces z jediného terminálu i při použití kombinaci pneumatických a elektrických pohonů ve stroji. [18] Ta také zajišťuje libovolné rozšíření terminálu o další moduly při změnách v koncepci již smontovaného stroje či chybě při objednávce, bez nutných zásadních změn do konstrukce a mechaniky. [19] [20]

V dnešní době již výrobci nabízejí stejné konfigurace terminálů jak s integrovaným programovatelným automatem, tak i s komunikačním modulem pro připojení na průmyslovou síť stroje. Z tohoto důvodu není příliš časté použití programovatelných instalačních ventilových terminálů ve velkých strojích, kde je celý stroj řízen ze samostatného PLC. [9] V menších či jednodušších zařízeních však mohou zastávat roli hlavního řídicího členu a tak celý systém centralizovat i bez nutnosti použít elektrický rozvaděč díky dostupnému krytí terminálu. [19] V decentralizovaných systémech mohou řídit proces jednotlivých menších stanic. [21]

Příkladem decentralizovaného systému s použitím programovatelného instalačního ventilového terminálu může být plnící stanice pro sypké materiály Clean Dock společnosti Azo. Pneumatický uchopovací systém nejprve uchopí uzávěr plnící nádoby a pomocí stlačeného vzduchu zjistí hladké spojení mezi násypkou a nádobou. Po tárování váhy, umístěné pod kontejnerem, začíná proces plnění. Jakmile tento proces skončí, sekce se hladce oddělí a nádoba pokračuje k další stanici. Zatímco proces celé linky řídí vyšší ovladač, chod každé ze stanic je řízen jedním CPX/MPA terminálem. Díky tomu je možné zvýšit bezpečnost celé linky a zároveň urychlit diagnostiku případných chyb. [21]



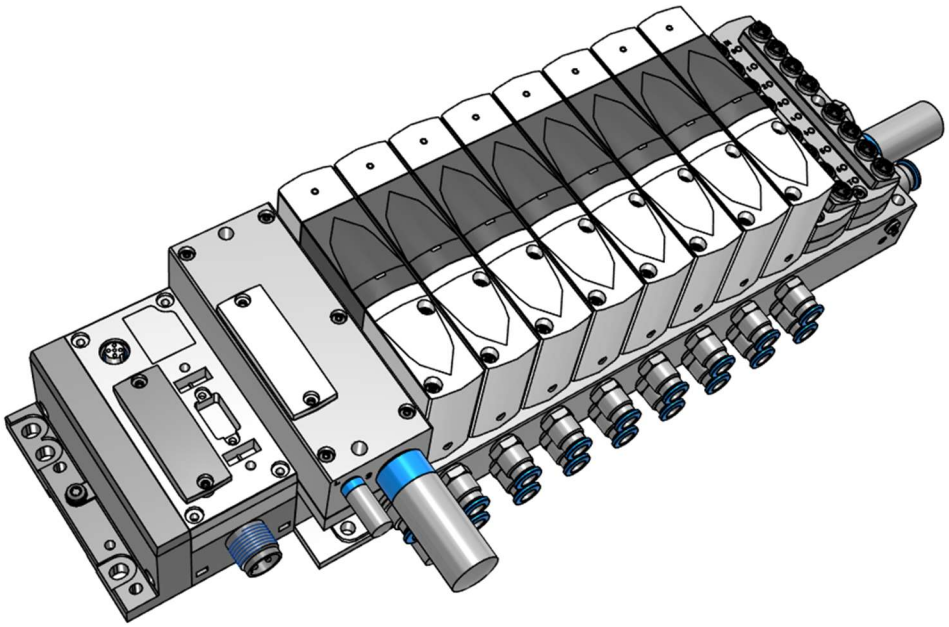

*Obrázek 11 Plnící stanice Azo Clean Dock [21]*

## Festo Motion Terminal

Nový ventilový terminál od společnosti Festo, Festo Motion Terminal (někdy také označován zkratkami FMT nebo VTEM) lze zařadit mezi novou éru ve vývoji pneumatiky, tzv. digitální pneumatiku. Jedná se o novodobé pojetí tohoto odvětví průmyslu, kdy je inteligentně propojena mechanika, elektronika a software k dosažení nejvyšší možné přizpůsobivosti a principů Průmyslu 4.0. [22] Díky univerzálním ventilům VEVM, které tvoří základ celého FMT, je možné přizpůsobit pohony konkrétním situacím v různých cyklech či dokonce i uvnitř jediného cyklu. Ventily se skládají z dvojice piezoventilů, která reguluje řídicí tlak a nastavuje chování čtyř sedlových ventilů zapojených do můstku. Takto fungující ventil VEVM dokáže zastoupit funkce až 50 dnešních konvenčních výrobků pro úpravu a usměrnění toku stlačeného vzduchu do pohonu beze změny v hardware. Jeden terminál FMT může obsahovat až 8 ventilů VEVM. Chování a funkce každého z ventilů jsou programovatelné pomocí různých aplikací MotionApp. Ty jsou přístupné v balíčcích a

vztahují se vždy na výrobní číslo daného VTEM, mezi jednotlivými terminály jsou nepřenosné. Systém je dodáván s programovatelným automatem CPX, který zaručuje nejen roli řídicího členu či zajišťuje připojení terminálu na průmyslovou sběrnici, ale také plnou kompatibilitu s dalšími přídavnými moduly. [22] [23] [24]

Konkrétním příkladem Festo Motion Terminal může být konfigurace obsahující 8 ventilů VEVM, 2 moduly pro 16 digitálních vstupů a programovatelný automat CPX s možností připojení na sběrnici Profibus. [4]



*Obrázek 12 Festo Motion Terminal [4]*

U tohoto produktu jsou některé výhody na první seznámení zřejmé: přizpůsobivost a kompaktnost, bez ohledu na mechaniku problému. S tím se i pojí jednoduchost objednávek a snadná montáž celého terminálu, který tak dokáže zastoupit několik pneumatických komponent najednou. Přesné funkce ventilů lze navíc stanovit až po samotné montáži stroje. Zároveň terminál pomáhá při sběru a vyhodnocení dat bez nutnosti použití přídavných snímačů. [22] Při použití některých MotionApp lze také ušetřit až 70% energie. Zároveň lze také aktivně sledovat stav připojených pohonů, včas rozpoznat jejich netěsnosti a tak ušetřit další energetické ztráty. O tuto prediktivní údržbu se opět stará daná MotionApp. [25]

Z vlastností Festo Motion Terminal a jeho výhod, které přináší, lze dobře odhadnout jeho aplikační oblasti. Nejvíce je vhodný pro systémy, kde je nutné měnit vlastnosti a popřípadě i funkce pohonů mezi jednotlivými výrobními sériemi dle daného výrobku. Další možností použití je v aplikacích, kde je třeba přesně polohovat pneumatický pohon v rámci jeho pracovního zdvihu či synchronizovat pohyb dvou motorů. [22]

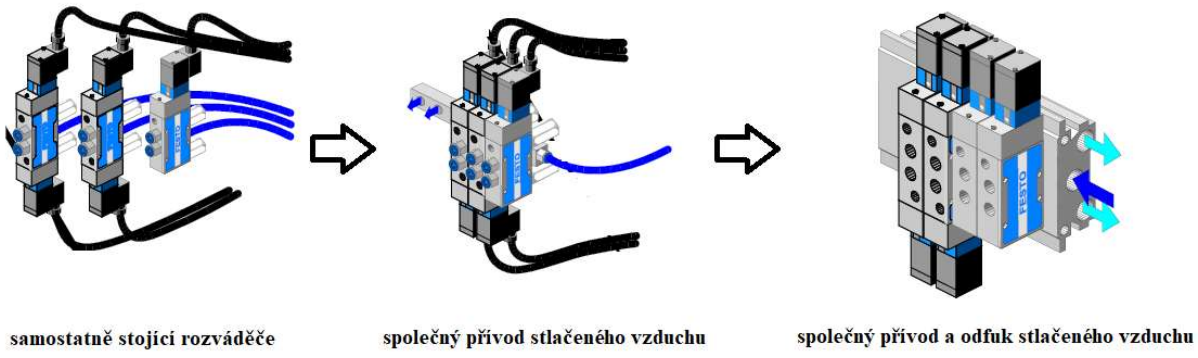
Jedním z konkrétních příkladů aplikace Festo Motion Terminal je vytváření různých vzorků v jednotlivých vrstvách gumy, předcházející výrobu automobilových pneumatik. Původně byly malé pneumatické válce ovládány několik proporcionálními ventily pro každou tlakovou zónu ležící mezi 0,05 až 8 bar. To zapříčiňovalo vysoké nároky na velikost instalačních prostor a komplexnost řídicího členu. Díky Festo Motion Terminal se celý proces zjednodušil a zlevnil. 8 ventilů VEVM nastavuje 16 tlakových zón díky použití MotionApp č. 4: proporcionální redukce tlaku dle modelu. Ta dokáže velmi přesně kontrolovat tlak a zároveň ho zpětně měřit pro kontrolu kvality. S použitím FMT se také zmenšil seznam použitých dílů, které univerzální ventily zastupují a plní jejich funkci. [26]

*Obrázek 13 Vzorkovač gumy před výrobou pneumatik [26]*

## Shrnutí

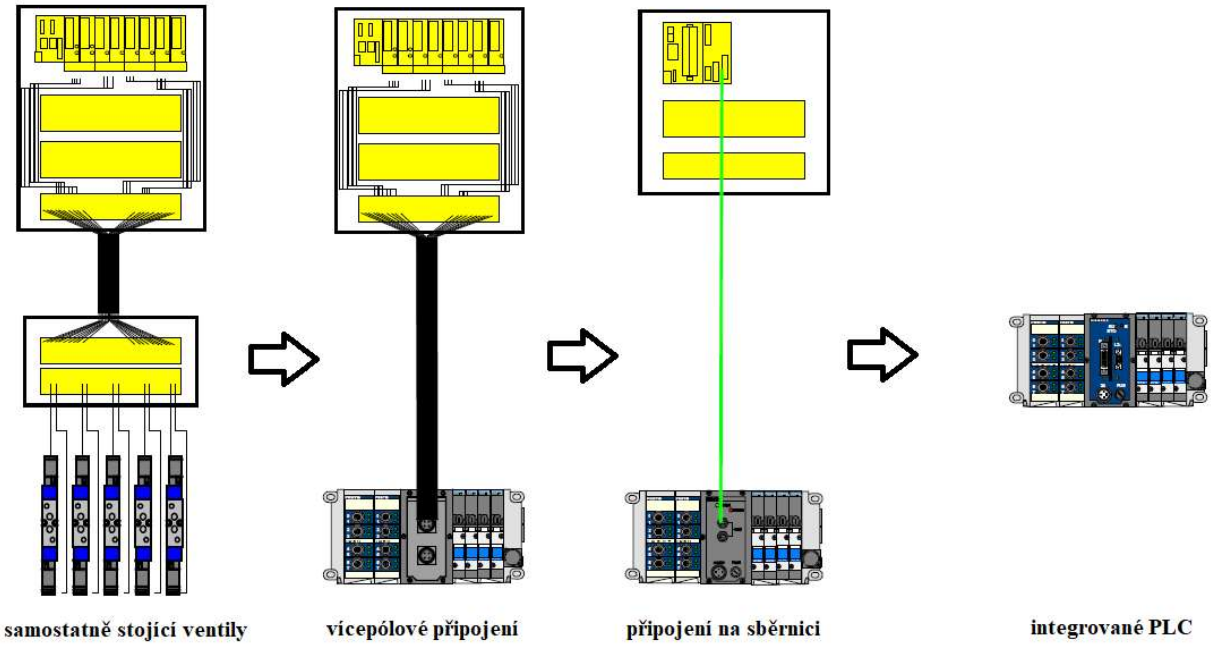
Vývoj pneumatických ventilových terminálů od samostatně stojících rozváděčů po programovatelné instalační ventilové terminály lze rozdělit do dvou fází.

V první byla postupně integrována pneumatická část ventilů společným přívodem stlačeného vzduchu a později i společným odfukem (viz Obrázek 10). [6]




*Obrázek 14 Vývoj pneumatické části [6]*

V druhé fázi bylo řešeno elektrické připojení ovládacích cívek ventilů spojených do ventilového bloku. Integrováním elektrické lišty do těla bloků a zpřístupnění kontaktů pomocí vícepólového připojení. Vznikly tak pneumatické ventilové terminály. Rozvoj průmyslových komunikačních protokolů zapříčinil další postup ve vývoji implementace připojením terminálů na průmyslové sběrnice a zásadním zvýšením jejich modulárnosti pro rozšíření o další funkce. Zatím posledním velkým krokem v rozvoji tradičních ventilových terminálů byla integrace programovatelných automatů přímo do terminálu, která vytvořila z ventilového terminálu samostatný člen s možností vlastního řízení i řízení dalších komponent (viz Obrázek 11). [11]




*Obrázek 15 Vývoj elektrické části a konektivity [11]*

Festo Motion Terminal posunuje hranice kompatibility a schopnosti přizpůsobit se ještě dál. Jeho univerzální ventily VEVM dokáží zastoupit funkce až 50 konvenčních pneumatických komponent a přizpůsobit tak v jakémkoliv okamžiku pohyb pneumatických motorů konkrétní situaci. Nová éra tzv. digitální pneumatiky tak plně přesunuje mechanické problémy do světa software. [23]

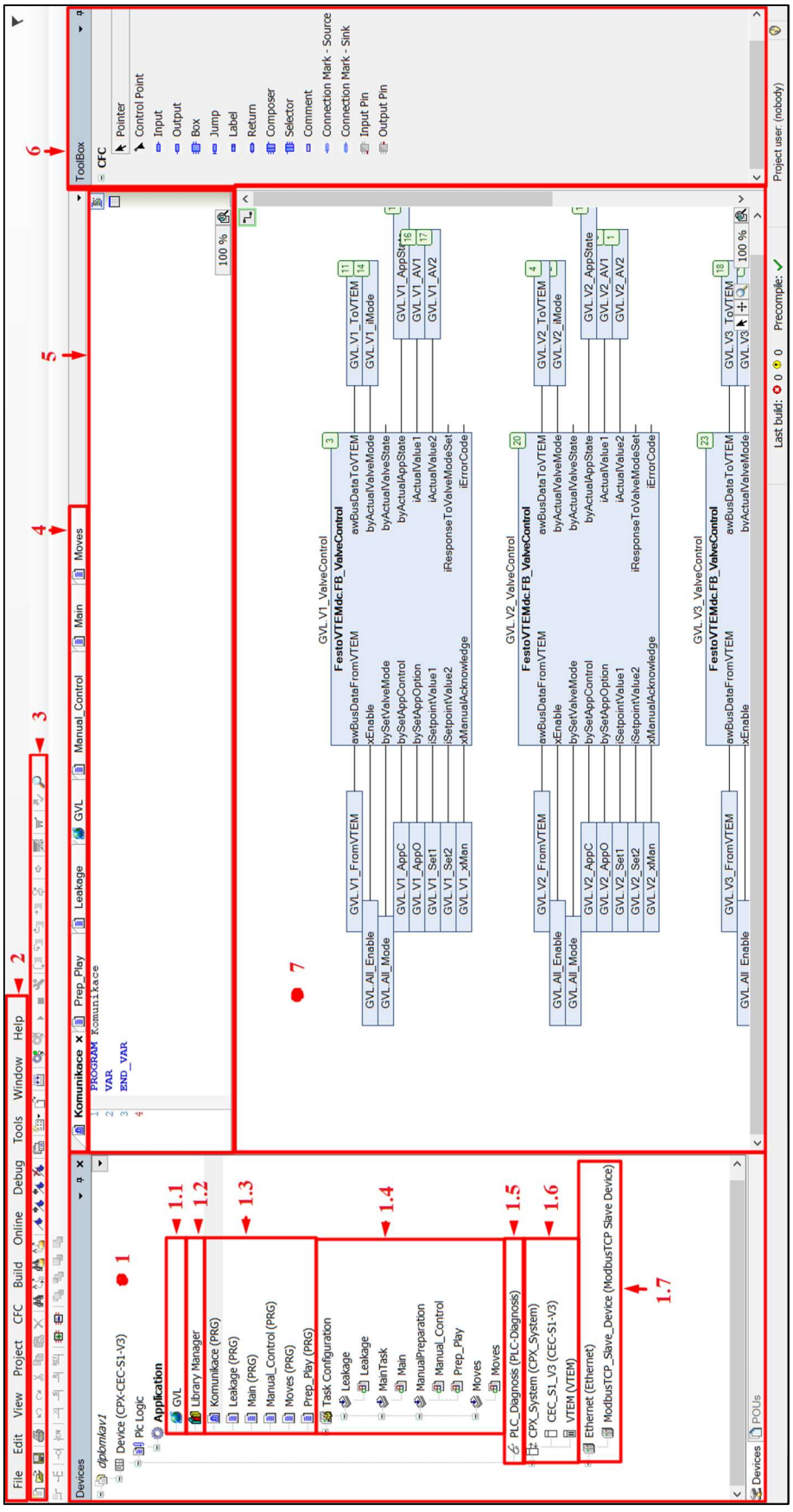
# 2. Funkce Festo Motion Terminal

Tato část může sloužit jako studijní materiál pro pochopení základních funkcí a principů programování Festo Motion Terminal. Čtenáři nejdříve stručně představuje funkční části vývojového prostředí CODESYS. Dále se zabývá knihovnou Festo_VTEM_DevCon (v. 3.5.7.221), která je určená k celkové komunikaci mezi programovatelným automatem CPX a ventily VEVM. [27] Na závěr jsou prostudovány některé funkce FMT a jejich naprogramování pomocí již zrealizované a funkční laboratorní úlohy. Tato kapitola diplomové práce byla vypracována s pomocí materiálů z předchozích akademických prací, dostupných u Ing. Marie Martináskové, Ph.D. [60] [61]

Předpoklady k práci s touto kapitolou:

- základní povědomí o zařízení Festo Motion Terminal
- nainstalované vývojové prostředí CODESYS
- základní znalosti programování PLC (jazyků ST a FBD)

# Vývojové prostředí CODESYS




*Obrázek 16 Hlavní okno vývojového prostředí CODESYS*

Po otevření existujícího či založení projektu vývojového prostředí CODESYS se zobrazí jeho hlavní okno, které se skládá z několika částí:

1. okno hierarchie projektu a zařízení
    - obsahuje všechny celky celého projektu

1.1. seznam globálních proměnných

1.2. organizátor knihoven

1.3. programové organizační jednotky
    - přidání POU: pravým tlačítkem Application > Add Object > POU… > Vybrat typ POU

1.4. nastavení úkolů
    - spravuje úkoly, jejich typy, interval provádění, obsažené POU v jednotlivých úkolech

1.5. diagnóza PLC

1.6. aktuální konfigurace zařízení
    - přidání aktuální konfigurace připojeného zařízení: dvojité rozkliknutí > Actual configuration > Scan > Apply

1.7. komunikační rozhraní
    - více o komunikačních rozhraních a jejich nastavení: viz kapitola 8 Komunikace: Vytvoření serveru

2. hlavní nabídka
    - všechny možnosti projektu
3. lišta zrychlené nabídky
4. lišta záložek
5. okno seznamu lokálních proměnných právě otevřeného POU
6. okno palety nástrojů právě otevřeného POU
7. programové okno právě otevřeného POU

Při otevření nového projektu je možné, že okno hierarchie projektu obsahuje jiné celky. Přidávání a odebírání celků se provádí vždy rozkliknutím celku pravým tlačítkem myši a následného výběru typu objektu, respektive možností odebrat.

### Knihovna Festo_VTEM_DevCon

Pro možnost založení či správy projektu obsahující řízení Festo Motion Terminal je nutné učinit tyto kroky:

1. stáhnout a nainstalovat balíček Target Support Package CODESYS verze 3.5.7.356
    - balíček podpory CPX a Festo Motion Terminal

- dostupné: https://www.festo.com/net/cs_cz/SupportPortal/InternetSearch.aspx?q=codesys&tab=16&type=73#result
- instalace probíhá přes CODESYS > Tools > Package Manager… > Install…

2. stáhnout a nainstalovat knihovnu Function blocks CODESYS verze 3.5.7
    - funkční bloky pro Festo Motion Terminal
    - dostupné: https://www.festo.com/net/cs_cz/SupportPortal/InternetSearch.aspx?q=codesys&tab=16&type=74#result
    - instalace probíhá přes CODESYS > Tools > Library Repository… > Install…

Tato kapitola je věnována funkčním blokům, které nově nainstalovaná knihovna obsahuje. Zvláště se zaměřuje na funkční blok pro ovládání jednotlivých ventilů skrze MotionApp s názvem *FB_ValveControl*.

### *FB_ValveControl*

Prvním, a také nejdůležitějším funkčním blokem, je již zmíněný *FB_ValveControl*. Ten slouží k ovládání jednotlivých MotionApp a komunikaci mezi CPX terminálem a ventilem VEVM. [27] Jeden tento blok vždy komunikuje s jedním VEVM ventilem. Informace se přenáší pomocí 6 vstupních a 6 výstupních bytů, v každém funkčním bloku umístěných nejvýše.


*Obrázek 17 Funkční blok FB_ValveControl pro ovládání ventilu VEVM [27]*

Vstupy, umístěné vlevo, jsou kódovány do 6 bytů a poté poslány do příslušného ventilu VTEM. Těchto 6 bytů lze získat také na výstupu *awBusDataToVTEM*.

- *xEnable* – řídící hodnota bloku, pokud je FALSE, komunikace s ventilem je neaktivní
- *bySetValveMode* – nastavení módu ventilu (číslo MotionApp)
- *bySetAppControl* – nastavení ovládání ventilu (MotionApp)
- *bySetAppOption* – nastavení možnosti ventilu (MotionApp)
- *iSetPointValue1/2* – nastavení hodnoty, specifické pro každou MotionApp

- *xManualAcknowledge* – potvrzení chyb při stavu ventilu „not ready“

Výstupy, umístěné vpravo, jsou dekódovány ze vstupu *awBusDataFromVTEM*.

- *byActualValveMode* – aktuální mód ventilu (číslo MotionApp)
- *byActualValveState* – aktuální stavu ventilu
- *byActualAppState* – aktuální stav MotionApp
- *iActualValue1/2* – aktuální hodnoty, specifické pro každou MotionApp
- *iResponseToValveModeSet* – diagnostika chyb způsobené při přijmutí zprávy ve VTEM
- *iErrorCode* – uložená hodnota poslední chyby

#### *FB_StateInterpreter_MA_##*

Tento typ bloku slouží k podrobnějšímu popisu aktuálního stavu dané MotionApp. Vstup bloku se napojí přímo na výstup *byActualAppState* a na jeho jednotlivých výstupech lze přehledněji sledovat aktuální stav, ve které se ventil a právě pracující MotionApp nachází, například stav koncových snímačů poloh pohonu.

#### *FB_Download_Single_Parameter/FB_Download_Multiple_Parameters*

Tyto dva bloky slouží k nahrání jednoho/několika parametrů do Motion Terminal. Nejedná se však o parametry řídicí MotionApp, ale o hodnoty charakterizující připojenou komponentu k danému ventilu, jako jsou například: typ pohonu, jeho zdvih, délka a průměr hadic, zatížení při vyjetí/zajetí, a další. Předávání informací pomocí těchto bloků je povoleno pouze v „převodním módu“ Motion Terminal (Transfer Mode).

#### *FB_Write_config_MA_##*

Tento blok funguje na stejném principu jako předchozí dva bloky. Skrze něj lze parametrizovat jednotlivé MotionApp. Rozdíl mezi výše zmiňovanými bloky a tímto je ten, že tento poskytuje pro každý parametr vlastní vstup. Vnitřně tedy pouze zakóduje vstupní parametry do pole a poté zavolá blok *FB_Download_Multiple_Parametres*. Použití tohoto bloku je tedy také podmíněno „převodním módem“ Motion Terminal (Transfer Mode).

#### *FB_Save_Configuration_Persistent*

Jednoduchý blok, jehož funkce je trvalé uložení právě parametrizovaného ventilu.

#### *FB_Upload_Single_Parameter/FB_Upload_Multiple_Parameters*

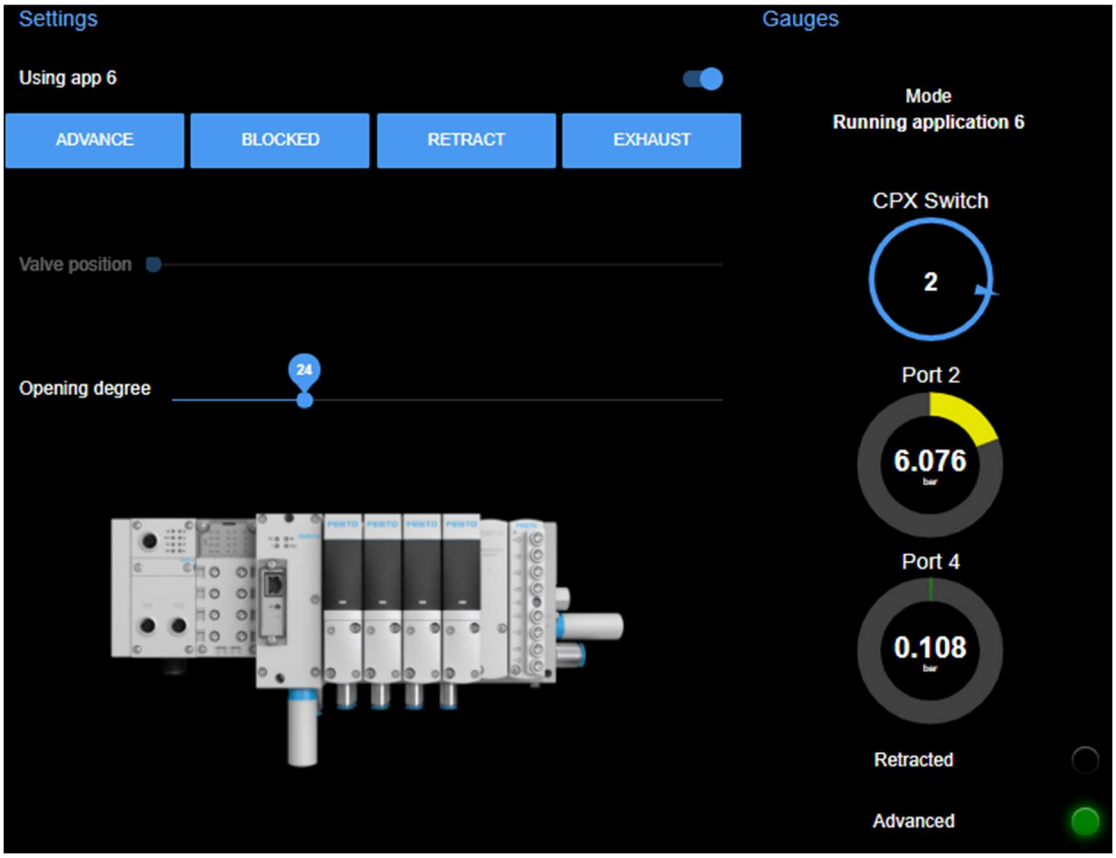
Tyto bloky slouží k získání uložených parametrů z Motion Terminal. Parametry charakterizují připojenou komponentu k danému ventilu. Předávání informací pomocí těchto bloků je povoleno pouze v „převodním módu“ Motion Terminal (Transfer Mode).

Navíc ještě knihovna obsahuje pomocné bloky pro čtení a zápis jednoho či několika bitů do bytu či wordu. Více o celé knihovně bloků a jejím použití se lze dočíst v příručce, která je součástí stáhnutého souboru.

## Aplikace Festo Motion Terminal v laboratoři

### *Představení aplikace*

Pro pečlivé prostudování některých funkcí a principu programování Festo Motion Terminal jsem si vybral jednu, již zrealizovanou, aplikaci v laboratoři. Ta se zabývá řízením pohybu jednoho pohonu s nainstalovanými senzory koncových poloh skrze dispečerské okno grafického uživatelského rozhraní. Ve stejném okně se také nachází několik vizualizačních prvků. Pro přenos informací využívá komunikační protokol Modbus TCP.




*Obrázek 18 Dispečerské okno grafického rozhraní aplikace*

Výše lze vidět grafické uživatelské rozhraní vybrané aplikace v laboratoři. Levá část dispečerského okna, při používání MotionApp 6, posílá příkazy do Festo Motion Terminal, jako jsou: vyjet, blokovat, zajet a vypustit. V tento moment lze také nastavovat úroveň otevření škrcení spodním posuvníkem v %. V pravé části okna se nachází vizualizační prvky zobrazující právě používanou MotionApp, stav tlačítka CPX, velikosti tlaků na jednotlivých portech a stav koncových snímačů pohonu. Po přepnutí MotionApp z čísla 6 na 2 lze v prvním posuvníku nastavovat pozici ventilu od -100% do 100% a ovládat pohon

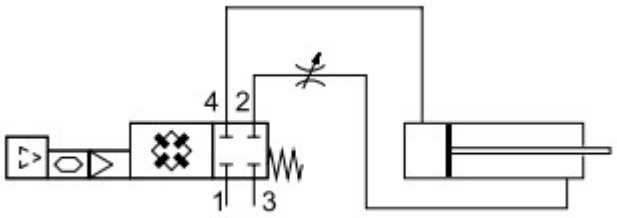
pomocí dvou tlačítek – aktivace portů a blokování portů. Ve vizualizační části proběhne změna ukazatelů, které poté ukazují pozici ventilu v % na jednotlivých portech.

### *Funkce Festo Motion Terminal*

Aplikace používá k ovládání směru a rychlosti pohybu motoru dvě různé MotionApp:

- Pohyb ECO (MotionApp 6)
  - škrcení přívodu vzduchu s uzavřením v koncové poloze
  - energeticky úsporné vysunutí a zasunutí
  - nutné použití čidel koncových poloh a modulu s digitálními vstupy
  - *iSetpointValue2* – ovládá úroveň škrcení

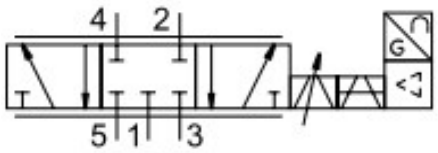
*Obrázek 19 ECO drive [24]*




*Obrázek 20 Funkce MotionApp 6: Pohyb ECO [24]*

- Proporcionální regulace průtoku (MotionApp 2)
  - regulace průtoku pomocí proporcionálních průtokových ventilů
  - výběr mezi 2 typy ventilů
  - *iSetpointValue1* – ovládá pozici ventilu 1
  - *iSetpointValue2* – ovládá pozici ventilu 2 (pokud je vybrán)

*Obrázek 21 Proportional directional control valve [24]*

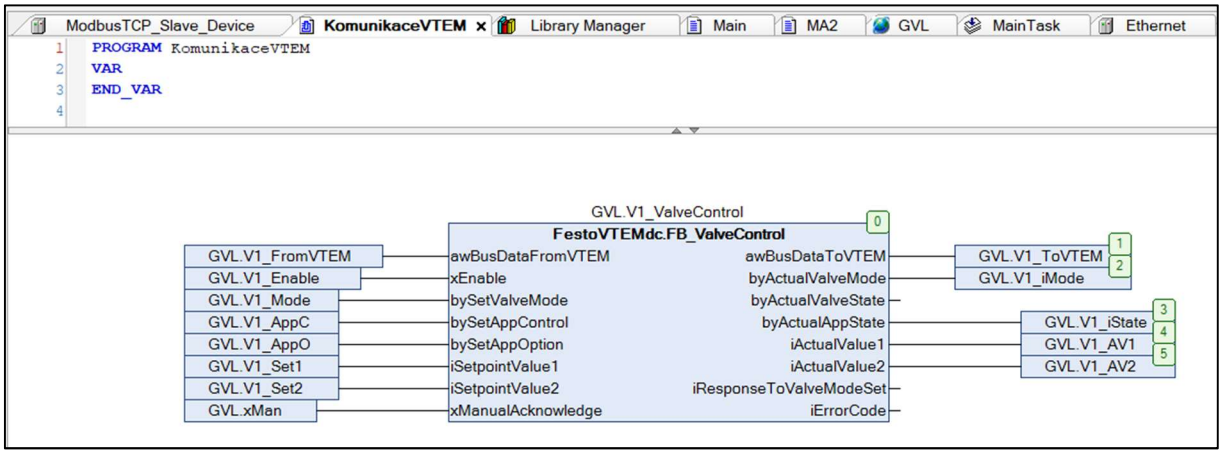



*Obrázek 22 Funkce MotionApp 2: Proporcionální průtokový ventil [24]*

### *Programování funkcí Festo Motion Terminal*

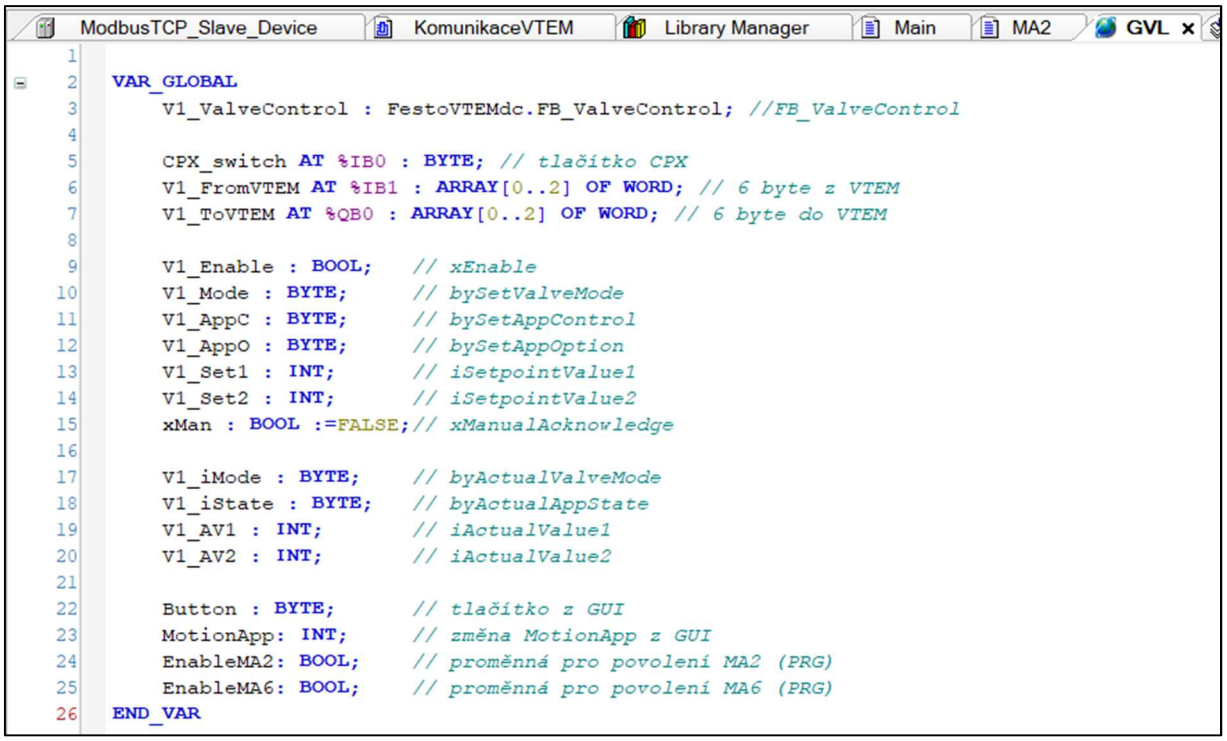
Po objevení všech dostupných možností tohoto aplikačního příkladu jsem se zaměřil na principy programování této úlohy. Projekt se skládá ze 4 programových organizačních jednotek, z nichž jedna obsahuje funkční blok pro komunikaci CPX

s ventilem VEVM a další 3 slouží přímo k řízení procesu. Pro lepší pochopení významu jednotek je vhodné se nejdříve zaměřit na funkční blok.




*Obrázek 23 Funkční blok FB_ValveControl v aplikačním příkladu*

Funkční blok pro ovládání prvního ventilu FMT *FB_ValveControl* se nachází v POU označeném jako *KomunikaceVTEM*, který je napsán jazykem FBD (Function Block Diagram). Všechny jeho jednotlivé vstupy jsou napojené na globální proměnné projektu, s přidáním i několika jeho výstupů. Jeho název je také definován globálně, což nás vede k seznamu globálních proměnných *GVL*.




*Obrázek 24 Seznam globálních proměnných GVL v aplikačním příkladu*

V tomto seznamu jsem si již okomentoval příslušné definované proměnné jejich funkcí. Většina z nich se přímo týká funkčního bloku *GVL.V1_ValveControl*. Poslední

odstavec proměnných slouží k získání povelů z grafického uživatelského rozhraní a povolení dalších POU.

GVL | ModbusTCP_Slave_Device | MainTask | Ethernet | MA6

Find | Filter Show all

| Variable | Mapping | Channel | Address | Type | Default Value | Unit | Description |
|---|---|---|---|---|---|---|---|
| regin | | Inputs | %IW25 | ARRAY [0..3] OF WORD | | | Modbus Holding Registers |
| regin0 | | Inputs[0] | %IW25 | WORD | 0 | | MA2 Set1 |
| regin1 | | Inputs[1] | %IW26 | WORD | 0 | | MA6 Set2 |
| regin2 | | Inputs[2] | %IW27 | WORD | 0 | | Tlačítko v GUI |
| regin3 | | Inputs[3] | %IW28 | WORD | 0 | | #MA |
| regout | | Outputs | %QW24 | ARRAY [0..4] OF WORD | | | Modbus Input Registers |
| regout0 | | Outputs[0] | %QW24 | WORD | 0 | | iActualValue1 |
| regout1 | | Outputs[1] | %QW25 | WORD | 0 | | iActualValue2 |
| regout2 | | Outputs[2] | %QW26 | WORD | 0 | | CPX tlačítko |
| regout3 | | Outputs[3] | %QW27 | WORD | 0 | | aktuální MA |
| regout4 | | Outputs[4] | %QW28 | WORD | 0 | | čidla válce |

*Obrázek 25 Modbus TCP server v aplikačním příkladu*

Jelikož se grafické uživatelské rozhraní běží na počítači mimo vývojový systém CODESYS, komunikace mezi ním a CPX je řešena pomocí Modbus TCP. Konkrétně je v automatu CPX vytvořen server, přes který si počítač a CPX vyměňují informace. Ve sloupci „Description" jsem v původním projektu okomentoval, pro lepší přehlednost, význam všech proměnných v registrech tohoto serveru.

Dále jsem se již zaměřil na samotné programování funkcí Festo Motion Terminal pro rozpohybování pohonu. V tomto projektu se tato úloha týká dvou programových organizačních jednotek: *MA6 (PRG)* a *MA2 (PRG)*, napsaných v jazyku ST (Structured Text). Ty jsou nazvány dle právě používaných MotionApp uživatelem. Pro lepší přehled nad principem kontroly obou MotionApp jsem přiložil výstřižek z tabulky Cbus-Protocol VTEM, ve které je vyznačen význam a hranice hodnot jednotlivých vstupů funkčního bloku *FB_ValveControl*.

| MotionApp | Sensors: without sensors | Sensors: End switches | Sensors: partial stroke | Sensors: full stroke sensors | cyclic process data: 6 Bytes from PLC to VTEM: Mode | App-Control | App-Option | SetPoint Value 1 | SetPoint Value 2 |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | 1 | | 2 | 3 4 | 5 6 |
| | | | | | 6 | 2 | 8 | 16 | 16 |
| **Proportional directional control valve** | x | | | | **2** | App-Control<br>0 ≙ (2) blocked / (4) blocked<br>1 ≙ (2) blocked / (4) active<br>2 ≙ (2) active / (4) blocked<br>3 ≙ (2) active / (4) active | ValveType<br>14 ≙ 4/3<br>15 ≙ 2 x 3/3 | Valve position (2)<br>-1000 ≙ -100 %<br>0 ≙ 0 %<br>10000 ≙ +100 %<br>Increment 0,01% | Valve position (4)<br>(only for 2 x 3/3)<br>-10000 ≙ -100 %<br>0 ≙ 0 %<br>10000 ≙ +100 %<br>Increment 0,01% |
| **ECO drive** | | x | | | **6** | App-Control<br>0 ≙ blocked<br>1 ≙ advance ((4) pressurized)<br>2 ≙ Retract ((2) pressurized)<br>3 ≙ exhaust | | | opening degree supply air flow control<br>1 ≙ 0,01%<br>10000 ≙ 100 % |

*Obrázek 26 Výstřižek z tabulky Cbus-Protocol VTEM [28]*

Main | MA2 | GVL | MA6 | ModbusTCP_Slave_Device | MainTask | Ethernet

```
PROGRAM MA6
VAR
    StateMA6 : INT;
END_VAR
```

```
GVL.V1_Set2 := WORD_TO_INT(regin1); // iSetpointValue2
// tato hodnota se mění dle posuvníku 2 ve vizu okně
// u MotionApp 6 se jedná o procentuální otevření škrcení
GVL.Button := WORD_TO_BYTE(regin2); // tlačítka z GUI
IF GVL.CPX_switch = 2 THEN
// MotionApp 6 - ECO drive //
    IF GVL.EnableMA6 THEN
        // cyklus CASE
        CASE StateMA6 OF
        0:
            StateMA6 := 5;
        5:
            // MotionApp 6, blocked
            GVL.V1_Mode := 6;
            GVL.V1_AppC := 0;
            StateMA6 := 10;
        10:
            // povolit FB_ValveControl
            GVL.V1_Enable := TRUE;
            StateMA6 := 15;
        15:
            //vizualizační tlačítko = ovládání pohonu
            IF GVL.Button = 0 THEN
                // pokud je 0 => stav: blokován
                GVL.V1_AppC := 0;
            ELSIF GVL.Button = 1 THEN
                // pokud je 1 => stav: vyjet
                GVL.V1_AppC := 1;
            ELSIF GVL.Button = 2 THEN
                // pokud je 2 => stav: zajet
                GVL.V1_AppC := 2;
            ELSIF GVL.Button = 3 THEN
                // pokud je 3 => stav: vypustit vzduch
                GVL.V1_AppC := 3;
            END_IF
        END_CASE

    END_IF
ELSE
    // při změně MotionApp vynulovat cyklus CASE
    StateMA6 := 0;
END_IF
```

*Obrázek 27 MA6 (PRG) v aplikačním příkladu*

Jak již název napovídá, tato POU se stará o ovládání MotionApp 6 – pohyb ECO. Při splnění podmínek CPX tlačítka v poloze 2 a povolení této MotionApp z dispečerského okna se nejdříve nastaví vstupy pro specifikaci čísla MotionApp (*GVL.V1_Mode := 6*) a počátečního stavu (*GVL.V1_AppC := 0*) - blokován. V dalším kroku se povolí komunikace (*GVL.V1_Enable*). V posledním kroku je již možné ovládat stav ventilu z dispečerského okna tlačítky (*GVL.Button*). Tato proměnná, společně s proměnnou pro ovládání úrovně škrcení (*GVL.V1_Set2*), je napojena na server Modbus TCP, odkud přijímá řídicí informace. Význam proměnných a jejich hodnot *GVL.V1_Set2* a *GVL.V1_AppC* lze dohledat v tabulce výše.

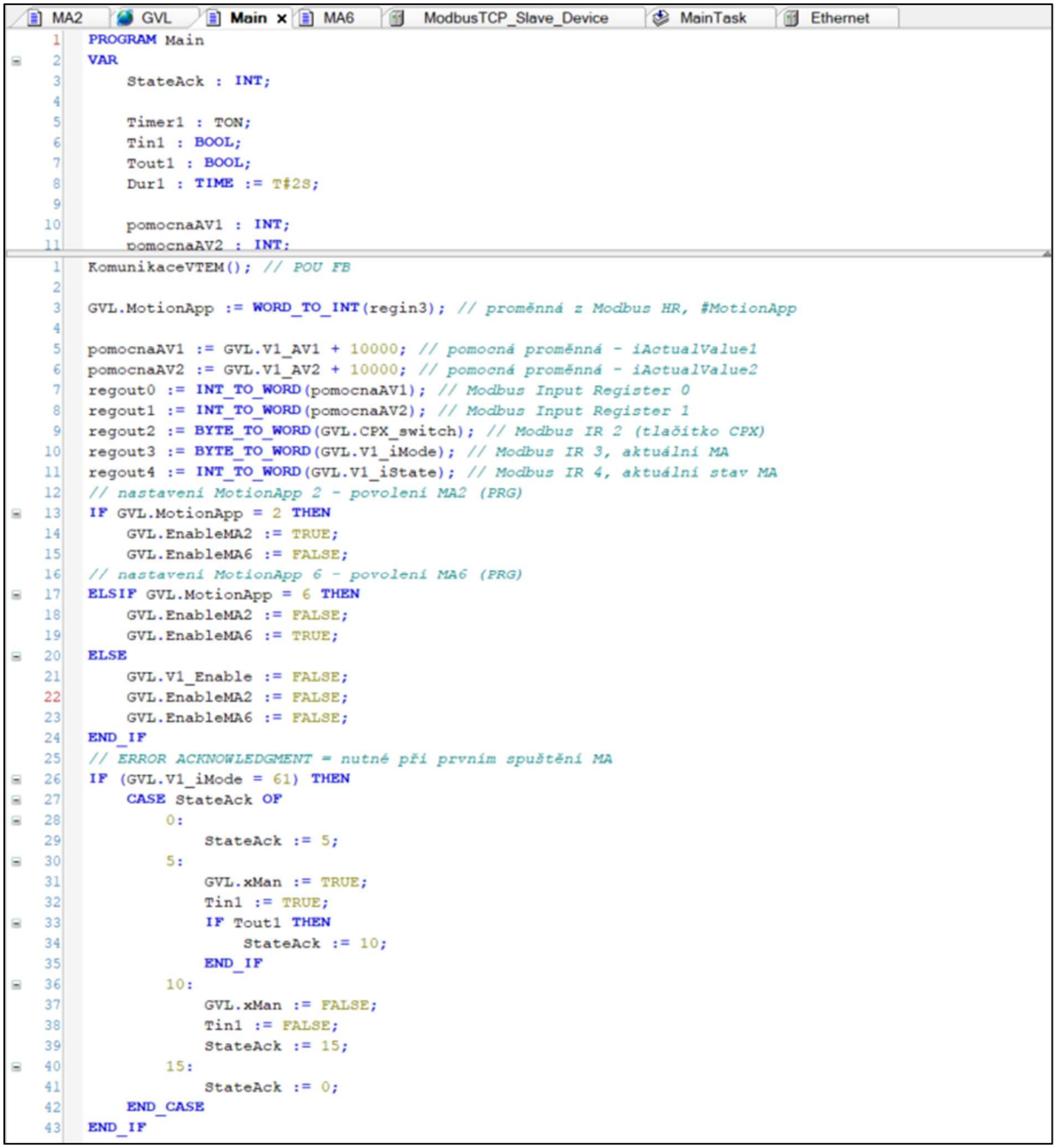
Main | MA2 | GVL | MA6 | ModbusTCP_Slave_Device | MainTask | Ethernet

```
PROGRAM MA2
VAR
    StateMA2 : INT;
    pomocnaSet1: INT;
END_VAR
```

```
pomocnaSet1 := WORD_TO_INT(regin0); // pomocná proměnná pro získání hodnoty Set1
GVL.V1_Set1 := pomocnaSet1 - 10000; // hodnota vstupu iSetpointValue1
// tato hodnota se mění dle nastavení ve vizu okně, z Modbus HoldR
// u MotionApp 2 se jedná o pozici průtokového proporc. ventilu
GVL.Button := WORD_TO_BYTE(regin2); // tlačítka z GUI

IF GVL.CPX_switch = 2 THEN
    // MotionApp 2 - valve position //
    IF GVL.EnableMA2 THEN
        // cyklus CASE
        CASE StateMA2 OF
        0:
            StateMA2 := 5;
        5:
            // MotionApp 2, blocked, typ ventilu 4/3
            GVL.V1_Mode := 2;
            GVL.V1_AppC := 0;
            GVL.V1_AppO := 14;
            StateMA2 := 10;
        10:
            // povolit FB_ValveControl
            GVL.V1_Enable := TRUE;
            StateMA2 := 15;
        15:
            IF GVL.Button = 0 THEN
                // pokud je 0 => stav: oba porty blokovány, bez pohybu
                GVL.V1_AppC := 0;
            ELSIF GVL.Button = 3 THEN
                // pokud je 3 => stav: oba porty aktivovány, pohyb umožněn
                GVL.V1_AppC := 3;
            END_IF
        END_CASE
    END_IF
ELSE
    // při změně MotionApp vynulovat cyklus CASE
    StateMA2 := 0;
END_IF
```

*Obrázek 28 MA2 (PRG) v aplikačním příkladu*

Druhý POU ovládá MotionApp 2 – proporcionální regulace průtoku. Program je velice podobný tomu předchozímu. Rozdílem je pouze jiná hodnota čísla MotionApp (*GVL.V1_Mode := 2*). Počáteční stav ventilu zůstává stejný (*GVL.V1_AppC := 0*) - blokován. Dále zde přibyla ještě specifikace ventilu, a to dle tabulky výše 4/3 (*GVL.V1_AppO := 14*). Poslední změnou je nutný přepočet hodnoty uložené v registru Modbus serveru. Jelikož Modbus nepodporuje přenos záporných čísel, a pozice ventilu, nastavovaná proměnnou *GVL.V1_Set1*, může být i záporná, je zde zaimplementován invertovaný přepočet hodnoty.




*Obrázek 29 Main (PRG) v aplikačním příkladu*

Posledním neprozkoumaným celkem aplikačního příkladu zbývá POU s názvem *Main (PRG)*. Jedná se o hlavní program, který na začátku posílá informace z VTEM do registru Modbus serveru. Dále přiděluje povolení funkce dalším POU dle přání uživatele. Pro programování budoucích úloh je však nejdůležitější poslední část. Jedná se o potvrzení chyby. Tato situace se objevuje vždy při prvním použití po zapojení Festo Motion Terminal, před kalibrací jednotlivých ventilů. Ventil se v tomto případě nachází ve stavu „not ready", čili není připraven a vykazuje aktuální stav číslo 61, což se jedná o pozastavení všech MotionApp. Řešením je přivedením náběžné hrany na vstup (*xManualAcknowledge*) funkčního bloku *FB_ValveControl*, jako tomu je i zde. Poté se ventil kalibruje a je možné ho začít používat.

## Shrnutí

V této kapitole teoretické části práce popisuji funkce a princip programování Festo Motion Terminal pomocí již zrealizované a funkční aplikace v laboratoři. Cílem je seznámení se s programováním tohoto terminálu v prostředí CODESYS.

V první části vysvětluji hlavní funkce jednotlivých oken vývojového prostředí CODESYS. Dále uvádím nutné balíčky a knihovny, které je třeba před začátkem programování mít nainstalované. V této části také popisuji jednotlivé funkční bloky, které obsahuje knihovna pro komunikaci mezi CPX a VTEM.

V hlavní části poté rozebírám aplikační příklad, který k řízení jednoho pohonu s nainstalovanými koncovými senzory využívá dispečerské okno běžící na počítači. Tato aplikace ve své činnosti používá dvě Motion App: proporcionální regulaci průtoku (MotionApp 2) a pohyb ECO (MotionApp 6). Z dostupných zdrojů jsem nejdříve teoreticky prostudoval funkce a možnosti těchto dvou aplikací a připravil podklady pro porozumění programu nahraného v programovatelném automatu CPX, který řídí Festo Motion Terminal. Po této teoretické průpravě vysvětluji jednotlivé programové organizační jednotky a další celky celého projektu.

Celá tato kapitola může sloužit jako zdroj informací pro pochopení principů programování Festo Motion Terminal.

# II. Praktická část

Hlavním cílem této práce je vytvořit koncepční návrh pohyblivého sedadla, které by mohlo být použito při promítání filmů v prostředí domácího kina. Hlavním účelem tohoto sedadla bude jistě prohloubit divácký zážitek z filmu.

## 3. Současný trh

V současné době jsou již na trhu nabízeny produkty tohoto odvětví zábavního průmyslu. Poptávány jsou však pouze velkými komerčními řetězci kin a jednoduché řešení pro domácnosti stále chybí. Příkladem může sloužit produkt MX4D, vynalezený americkou společností MediaMation, který je zobrazen na obrázku níže. První otevřený kinosál se systémem MX4D byl otevřen v Kolumbii v roce 2014. [29]


*Obrázek 30 Sedadlo MX4D Motion EFX společnosti MediaMation [30]*

Jak lze z obrázku vidět, sedadlo disponuje třemi druhy pohybů:

- zvedání nahoru a dolu
- natáčení vpřed a vzad
- natáčení vpravo a vlevo

Dále také celý systém dokáže simulovat vítr, déšť, různé druhy vůní, vibrace, bouchnutí v části zad a další. [30]

K dosažení pohybů jsou u systému MX4D použity pneumatické motory. Konkurenční zařízení 4XD nahrazuje tyto komponenty elektrickými servomotory. [31] Další efekty jsou zprostředkovány pomocí fukarů, rozprašovačů, trysek a elektrických či

pneumatických pohonů menších rozměrů. K pohonu a řízení pohybů sedadel MX4D Motion EFX je použit Festo Motion Terminal VTEM. Tři ventily VEVM kontrolují tři pneumatické motory pomocí několika MotionApp. Zbytek efektů je ovládán standartními ventily VUVG. [32] Díky použití stlačeného vzduchu jako poháněcího média je zajištěn velmi tichý provoz celého systému.


*Obrázek 31 Pneumatické komponenty sedadla MX4D Motion EFX [32]*

Synchronizaci promítaného filmu s pohyby zajišťuje speciální soubor, který je nahrán do řídícího počítače, kde je dále zpracován. Na tvorbě pohybové stopy se podílejí samotní vývojáři společně s filmovými studii, včetně režisérů, k dosažení toho nejlepšího výsledku. K programování celé stopy je vyvinuta speciální platforma velmi podobná klasickým vývojovým prostředím. [31] [33]

V České Republice jsou v současné době 3 multikina, všechny stejného řetězce, nabízející technologii 4XD. [34] Konkurenční systém MX4D je nejvíce rozšířen na americkém kontinentu. [35]

# 4. Specifikace vlastností a základních parametrů sedadla

Jelikož se na trhu zábavního průmyslu objevují sedadla s elektrickým i pneumatickým pohonem a ani jedno z těchto řešení výrazně neporáží to druhé, nabízí se otázka, jaký typ tedy pro tento projekt vybrat. Díky mým předchozím zkušenostem s Festo Motion Terminal VTEM jsem se rozhodl pro stlačený vzduch jako poháněcí medium. Ve velké míře tím bude možné využít schopnost digitálních ventilů VEVM měnit svoje ovládací vlastnosti v čase a zastoupit tak více než 50 konvenčních dílů pro zpracování stlačeného vzduchu k této aplikaci.

Kvůli použití pohyblivé sedačky v domácím prostředí by bylo hlavně z hygienických důvodů a neustálé náročné údržby nevhodné zaimplementovat do projektu takové efekty, jako jsou simulace větru, deště a vůní. Dalším důvodem by také byla technická náročnost provedení, jelikož u sedadel do velkých kin většinu těchto účinků zajišťuje jiné sedadlo umístěné v řadě před divákem. Mým hlavním cílem tedy bude navrhnout systém přehrávače s pohyblivou sedačkou, kterou bude možné rozpohybovat do minimálně tří pohybů popsaných v předchozí kapitole.

Dalším parametrem, který je nutno navrhnout, je velikost sedadla. To musí rozhodně splňovat kritéria lehké manipulovatelnosti, nenáročnosti na velikost zabíraného prostoru a také adekvátnost k potencionálnímu počtu diváků. Z těchto důvodů jsem volil sedadlo pro dva lidi s nosností do 200 kg.

Použití křesla hlavně v domácím prostředí průměrné rodiny také dává požadavky na výběr řídícího členu. V současné době je již v každé domácnosti přítomen stolní počítač či notebook, převážně s nainstalovaným systémem Microsoft Windows. Z tohoto důvodu jsem i tento systém vybral jako nejvhodnější k tomuto projektu.

Další skutečností je programování a distribuce pohybové stopy. Tento bod budu detailně popisovat v dalších částech práce, ale obecně jsem vytvořil snahu o co nejjednodušší způsob řešení jak z pohledu programátora, tak z pohledu uživatele, diváka.

Posledním důležitým bodem je jistě pořizovací cena nejen komponent společně s křeslem, ale také potřebného software. Cenu hardware vybavení bohužel není možné nijak zásadně ovlivnit, ale v oblasti programového vybavení projektu již lze ušetřit, pomocí například užitím volně dostupných programů.

V tabulce lze vidět všechny důležité vlastnosti navrhovaného sedadla, vyplývající z této kapitoly:

| Navržené vlastnosti a parametry sedadla | |
|---|---|
| počet diváků | 2 |
| nosnost | 200 kg |
| efekty | 3 základní pohyby |
| poháněcí medium | stlačený vzduch |
| řídící a přehrávací člen | PC/notebook s Windows 8, 10 |
| ovládání pohybu | Festo Motion Terminal VTEM |

*Tabulka 1 Navržené parametry sedadla*

# 5. Komponenty

## Konstrukční řešení

Prvním úkolem je jistě najít vhodné konstrukční řešení v rámci umístění pneumatických motorů a stanovit jejich počet. Při plnění důrazu na jednoduchost konstrukčního řešení a zároveň jeho efektivnost jsem se rozhodl pro čtyři pneumatické pohony, které budou umístěny každý pod jedním rohem sedačky. Tímto způsobem bude možné realizovat více než tři základní pohyby stanovené v základních parametrech.

## Pneumatické pohony

Pro výběr pneumatických motorů jsem využil katalog Festo. [4] Nejdříve však bylo nutné spočítat silové účinky zatížené sedačky, které budou muset motory pro pohyb překonat. Mějme tedy maximální zatížení sedačky osobami $m_{zat}$, váhu sedačky $m_{sed}$, váhu všech komponent $m_{komp}$, a koeficient bezpečnosti $k$. Předpokládaný tlak v pneumatickém systému je 6 barů.

Silové zatížení maximálně zatížené sedačky je tedy

$$F = \left(k \,.\, m_{zat} + m_{sed} + m_{komp}\right).\, g \tag{1}$$

kde

$$m_{zat} = 200\ \mathrm{kg}$$

$$m_{sed} = 50\ \mathrm{kg}$$

$$m_{komp} = 40\ \mathrm{kg}$$

$$g = 9{,}81\ \mathrm{kg}\,.\,\mathrm{m}^{-3}$$

$$k = 1{,}3$$

Po výpočtu vychází celkové silové zatížení $F_c$:

$$F_c = 3434\ \mathrm{N}$$

Na každý z pneumatických motorů tedy připadá maximální protichůdná síla $F$:

$$F = \frac{F_c}{4} \tag{2}$$

$$F = 859\ \mathrm{N}$$

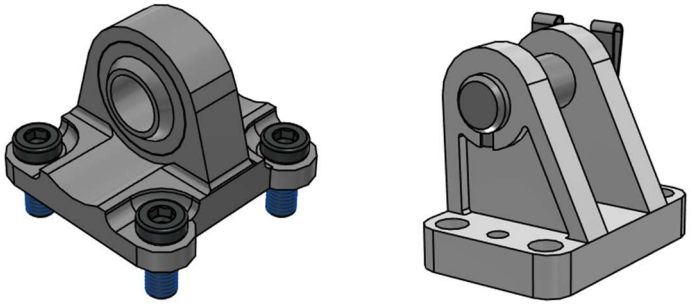
Pro tento účel jsem vybral kompaktní válce dle norem ADN ISO 21287 společnosti Festo se zdvihem 100 mm. Dle katalogového listu byl zvolen průměr pístu 50 mm, který

dokáže při tlaku vzduchu 6 bar vyvinout sílu 1057 N, což je dle výpočtů v porovnání se sílou *F* výše dostačující. [36] Válec disponuje samočinně nastavitelným pneumatickým tlumením PPS, které je vhodné použít při vysokých rychlostech či velkých hmotnostech a má velké tlumící schopnosti pro pohodlí diváka při dojezdu do krajních poloh.

*Obrázek 32 Pneumatický válec ADN [4]*

### Upevňovací sada

K upevnění válců ke křeslu budou použity taktéž produkty Festo, zajišťující kompatibilitu s výše vybranými válci. Kyvné příruby SNCS a ložisková tělesa LBG zajistí volné natočení při různých výškách vyjetí pneumatických motorů v různých rozích sedačky. [4]

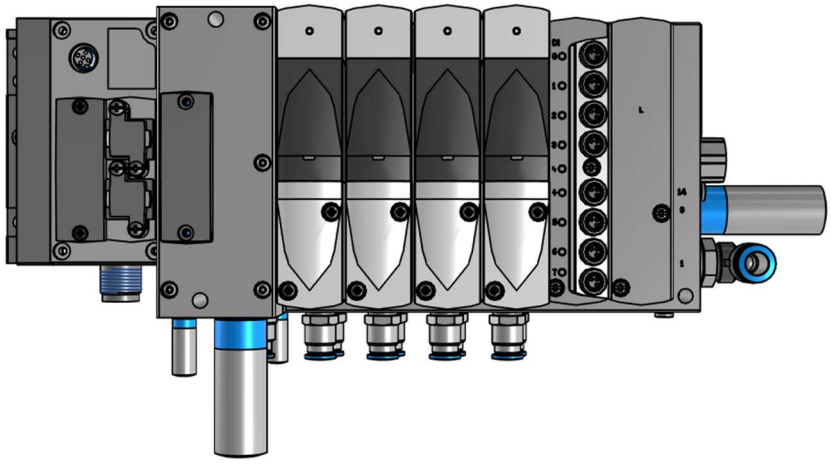


*Obrázek 33 Příruba SNCS a ložiskové těleso LBG [4]*

### Festo Motion Terminal VTEM

K ovládání pohybů pneumatických válců jsem již v předchozí kapitole zvolil Festo Motion Terminal. Vzhledem k rozhodnutí použít 4 motory, budou potřeba 4 ventily VEVM. Programovatelný automat CPX využiji jako vnitřní ovladač, přes který bude navázána komunikace na ovladač vyššího řádu, v tomto případě počítač, pomocí síťového kabelu Ethernet.. K tomuto účelu jsem se rozhodl pro model CPX-CEC-S1-V3 (T32). Jako vstup

VTEM terminálu bude možné připojit koncové snímače poloh motorů přes modul digitálních vstupů CTMM-S1-D, avšak tato možnost zůstane volitelná a bude ošetřena v dalších částech projektu. [4]

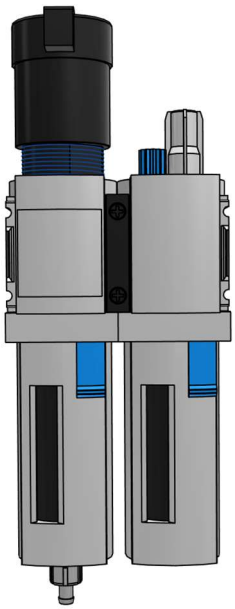


*Obrázek 34 Použitá konfigurace Festo Motion Terminal VTEM [4]*

## Zdroj vzduchu

S výběrem stlačeného vzduchu jako poháněcího média bohužel vzniká jeden problém, a to jeho výroba a skladování. Kompresory jsou dnes běžně dostupným zařízením, avšak mají jisté nevýhody. Asi největší z nich, která se přímo dotýká projektu, je jejich hlučnost za chodu. Na trhu lze najít i tzv. tiché kompresory, jejichž míra hluku se může pohybovat až kolem 50 dB, což odpovídá přibližně normálnímu bavení se mezi sebou. [37] [38] Umístěním takovéhoto kompresoru pod sedačku, která by tento hluk i dále tlumila, je jedno možné řešení. Druhé řešení je umístit kompresor mimo promítací místnost, což by však znamenalo řešit další problém rozvodu vzduchu. Jeho řešení by poté bylo čistě individuální záležitostí. Objem kompresoru by se mohl pohybovat mezi 10 až 20 litry.

Současně se zdrojem stlačeného vzduchu nelze také opomenout jeho kvalitativní úpravu. K tomuto účelu poslouží baterie MSB4-FRC, skládající se z redukčního ventilu s manometrem a filtrem a maznicí. [4]



*Obrázek 35 Úpravna vzduchu MSB4 [4]*

### Seznam základních komponent

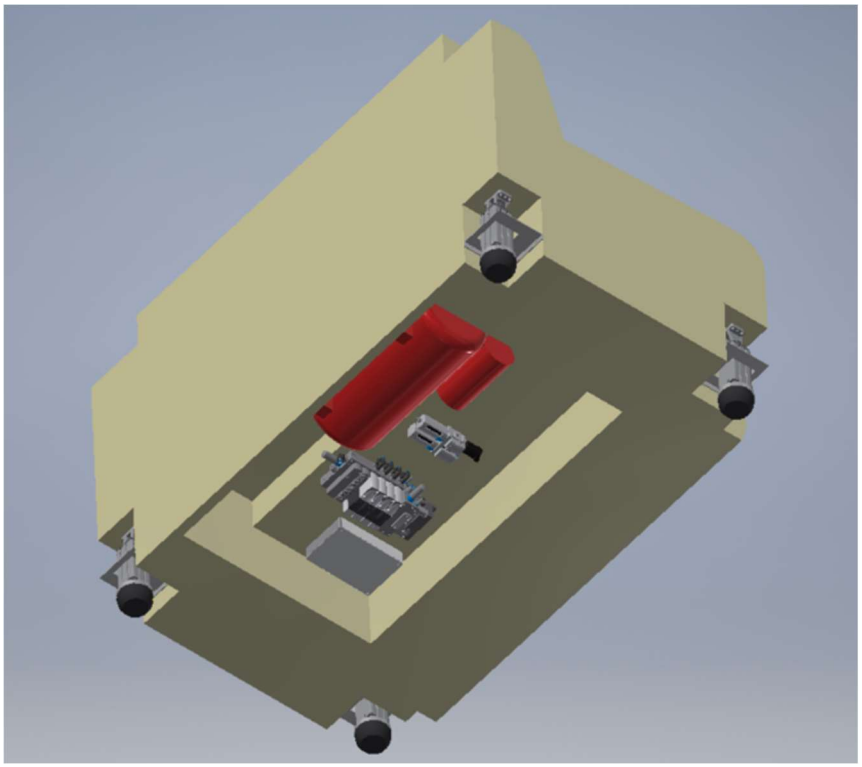
Ceny uvedených výrobků jsou přibližné a vycházejí z katalogu firmy Festo. Zdroj: [4]

| Základní komponenty | |
|---|---|
| Výrobek | Cena za kus |
| Festo Motion Terminal VTEM | 1600 EUR |
| ADN-50-100-A-PPS-A | 90 EUR |
| SNCS-50 | 60 EUR |
| LBG-50 | 60 EUR |
| MSB4-FRC | 300 EUR |

*Tabulka 2 Základní komponenty a jejich cena za kus*

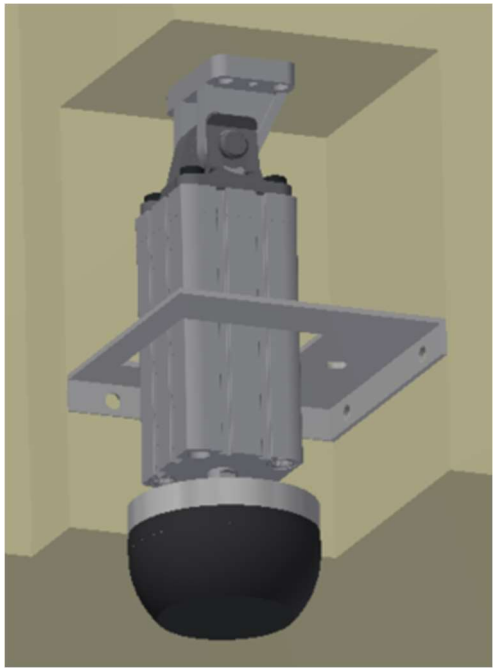
# 6. Koncepční návrh sestavy

Celý koncept modelu sedadla jsem vytvořil v CAD softwaru Autodesk Inventor Professional 2018 s využitím komponent volně stažitelných z [39] (sedadlo) a z [4] (výrobky Festo). Komponenty jsou umístěny ve spodní části sedadla, včetně tichého kompresoru. Pokud by se při konstrukční realizaci kompresor z rozměrových důvodů nevešel do prostoru pod sedačkou, či způsoboval nadměrný hluk a snižoval by tak pohodlí diváků, bylo by nutné přistoupit k druhému řešení umístěním kompresoru mimo sedačku.



*Obrázek 36 Koncepční návrh modelu sedadla, pohled zespodu*

Na pístnici pneumatických motorů je připevněna „pata“ z tvrdé gumy polokulového tvaru pro možnost natočení a šetrnosti k podložce. Z důvodu vymezení pohybu „noh“ pouze na nutné natáčení při pohyblivých funkcích sedadla je okolo motorů připevněna plechová deska, jak lze vidět detailněji na obrázku číslo 7.



*Obrázek 37 Pohled na konstrukční řešení připevnění pneumatických motorů*

# 7. Výběr software

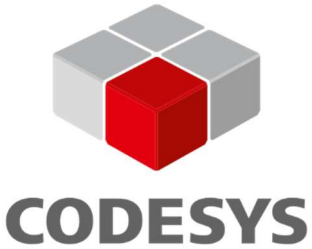
Po předchozích dvou kapitolách věnované výhradně hardware komponentům celého projektu jsem se přesunul k dalšímu důležitému úkolu, a to výběru programových prostředků. V minulé části nemohl být příliš dbán důraz na nízké náklady, což však v software oblasti již neplatí.

Prvním kritériem je cena programů a vývojových prostředí. Ideálním cílem je založit celý projekt na volně dostupném software. Druhým kritériem je kompatibilita s operačním systémem Windows NT, nejlépe verzí 8 a vyšší. Nadstavbou tohoto bodu je případná snadná přenositelnost na Raspberry Pi OS.

Po programové stránce lze celý projekt rozdělit do tří hlavních kategorií, ve kterých bude probíhat výběr software: vývojové prostředí pro PLC, programovací jazyk a prostředí pro vytvoření aplikace a multimediální přehrávač.

## PLC vývojové prostředí

K programování Festo Motion Terminal, respektive integrovaného programovatelného automatu CPX, bude využito vývojového prostředí CODESYS V3.5. Toto prostředí je přímo podporováno výrobcem Festo a nabízí programování dle mezinárodního průmyslového standartu IEC 61131-3 pro programovatelné logické automaty. [40] Pro naše užití je software kompletně zdarma.



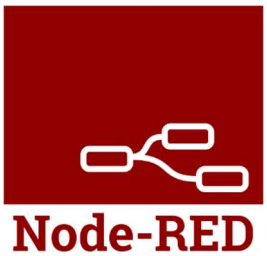

*Obrázek 38 PLC vývojové prostředí CODESYS [41]*

## Programovací jazyk a prostředí pro aplikaci

Pro tvorbu počítačového programu, který bude ovládat celý projekt a také obsahovat grafické uživatelské rozhraní jsem zvolil programovací prostředí Node-RED. Jedná se o programovací nástroj vhodný k propojování hardware zařízení, internetových služeb a aplikací. [42] Využívá principu toku zpráv mezi propojenými uzly (angl. nodes). Každý uzel po přijmutí zprávy vykoná svoji určitou funkci v závislosti na příchozím obsahu, případně zprávu upraví a pošle ji zase dále do dalších uzlů. K vytvoření vlastních aplikací je dostupná široká škála uzlů, volně dostupných v online knihovně. Programování JavaScript funkcí probíhá graficky v internetovém prohlížeči. Dalším mocným nástrojem,

který je s Node-RED spojen jako volně stažitelný doplněk, je tzv. *dashboard*, neboli přístrojová deska. Jedná se o programovatelné vizualizační prostředí sloužící k živé vizualizaci dat a dispečerskému řízení. Je plně programovatelné ve stejném editoru a lze jej buď sestavit z různých předpřipravených ovládacích prvků a ukazatelů, nebo si vytvořit své vlastní použitím HTML a JavaScript.

Node-RED byl původně vyvinut společností IBM jako multiplatformní programovací prostředí s hlavním cílem propojování zařízení jako části internetu věcí. Nyní se jedná o open-source projekt dále vyvíjený společností OpenJS Foundation. [43] Běh programu je založen na Node.js, díky čemuž může aplikace běžet i na méně výkonných zařízeních, jako je například Raspberry Pi. [42] Díky širokému rozšíření nástroje na mnoho platforem a zároveň velké oblíbenosti, zejména u uživatelů využívající Raspberry Pi k automatizaci svých domácností, není nutné často programovat jednoduché funkce od začátku, ale lze využít komunitní online knihovny ke stažení a přidání si do své „palety" další funkční sady uzlů.




*Obrázek 39 Programovací prostředí Node-RED [42]*

## Multimediální přehrávač

Pro přehrávání filmů je nutností vybrat multimediální přehrávač. Operační systém Windows sice obsahuje nativní přehrávač, avšak jeho možnosti konfigurace a ovládání jsou velice omezeny. Navíc není nabízen pro Raspberry Pi. Nabízí se tedy využít jeden z nejrozšířenějších volně dostupných přehrávačů na trhu, VLC Media Player. Jedná se o zcela otevřený projekt francouzských studentů, původně vyvíjený pro operační systémy Linux. [44] Je dostupný jak pro Windows, tak pro Raspberry Pi OS. Z těchto důvodů se mi jeví jako nejlepší volba pro přehrávání filmů na jakémkoliv ze zařízení disponující těmito operačními systémy.

*Obrázek 40 VLC Media Player [45]*

# 8. Komunikace

## Výběr komunikačního protokolu

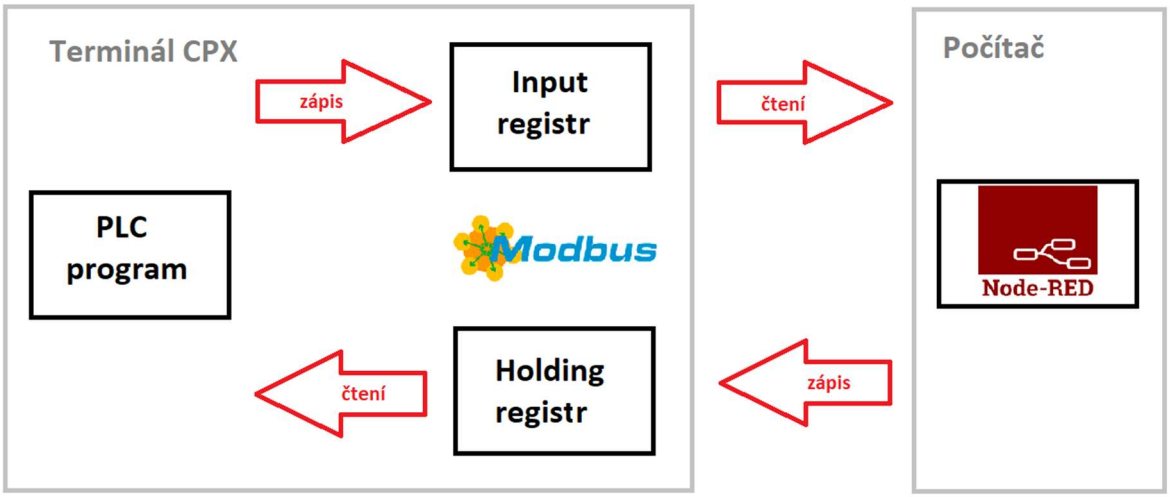
Dalším krokem je zvolit způsob komunikace mezi počítačem a terminálem CPX umístěným na Festo Motion Terminal. Varianta modulu T32 nabízí dvě komunikační rozhraní: RS-232 a RJ-45. [46] Vzhledem k faktu, že většina stolních počítačů i notebooků disponují v dnešní době konektory RJ-45 pro připojení k místní síti LAN přes Ethernet kabel, ke komunikaci mezi CPX a počítačem bude využito právě tohoto rozhraní.

Dle dokumentace Festo k výrobku CPX-CEC [46] rozhraní Ethernet obsahuje protokoly TCP/IP, EasyIP a Modbus TCP. K průmyslovému účelu, i k tomuto projektu, je nejvíce vhodný právě poslední zmíněný komunikační protokol Modbus TCP. Oproti klasické sériové verzi, která je založena na hierarchické struktuře master/slave, verze TCP pracuje na principu klient/server. Přenos informací mezi serverem a klientem je inicializována žádostmi klienta provézt některý z dostupných příkazů. Server poté tento příkaz vykoná a odešle klientovi data či potvrzení o vykonání příkazu. [47] Server podporuje 4 typy přenášených a ukládaných dat: [48]

- Discrete input (1 bit pouze ke čtení)
- Coil (1 bit ke čtení a zápisu)
- Input registr (16 bitový registr pouze ke čtení)
- Holding registr (16 bitový registr ke čtení a zápisu)

V tomto projektu přidělím roli serveru programovatelnému automatu CPX, respektive jeho rozhraní Ethernet, a roli klienta počítači a PLC programu, který však bude také zpracováván CPX. Tuto volbu lze zdůvodnit několika tvrzeními. Při opačném výběru by byla zbytečně zatěžována síť neustálými požadavky CPX terminálu o získání dat ze serveru umístěného na počítači. Navíc tímto přidělením rolí jsou data ukládány přímo v terminálu a jejich zpracování PLC programem je tak rychlejší. Počítač pouze v určité okamžiky posílá data na server nebo žádá o jejich čtení, což se však neděje zdaleka tak často.

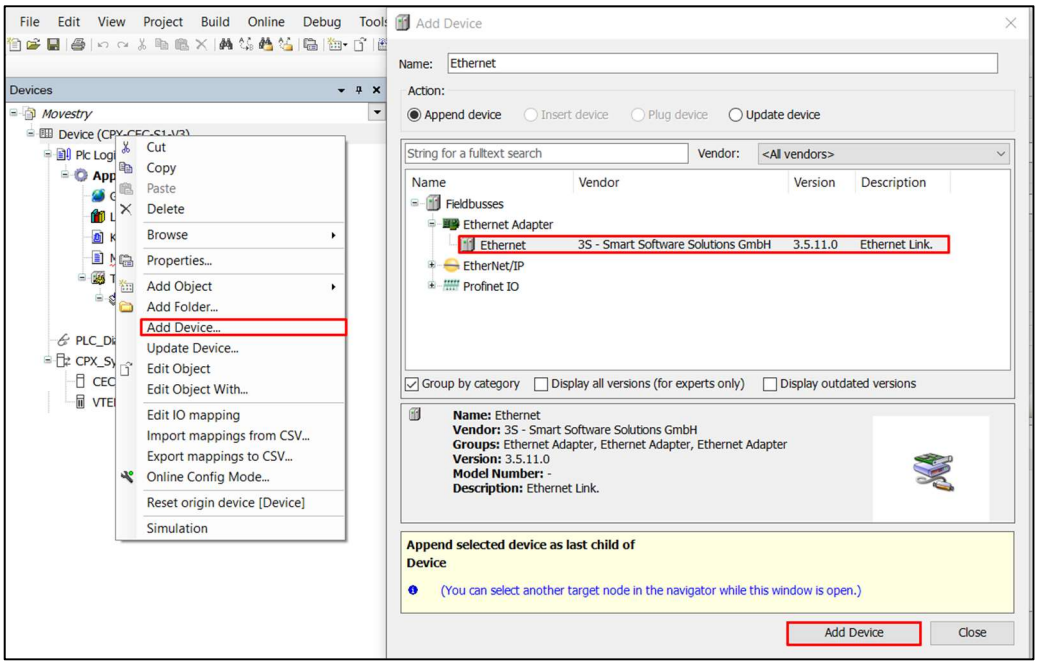
Data budou přenášena pomocí 10 proměnných v Holding registru, do kterých bude mít práva zapisovat pouze počítač a PLC program bude pouze získávat informace, a 14 proměnných v Input registru, do kterých bude aktuální hodnoty zapisovat automat CPX, respektive PLC program, a počítač bude z tohoto registru číst. Přesný význam dat bude uveden v dalších částech práce.



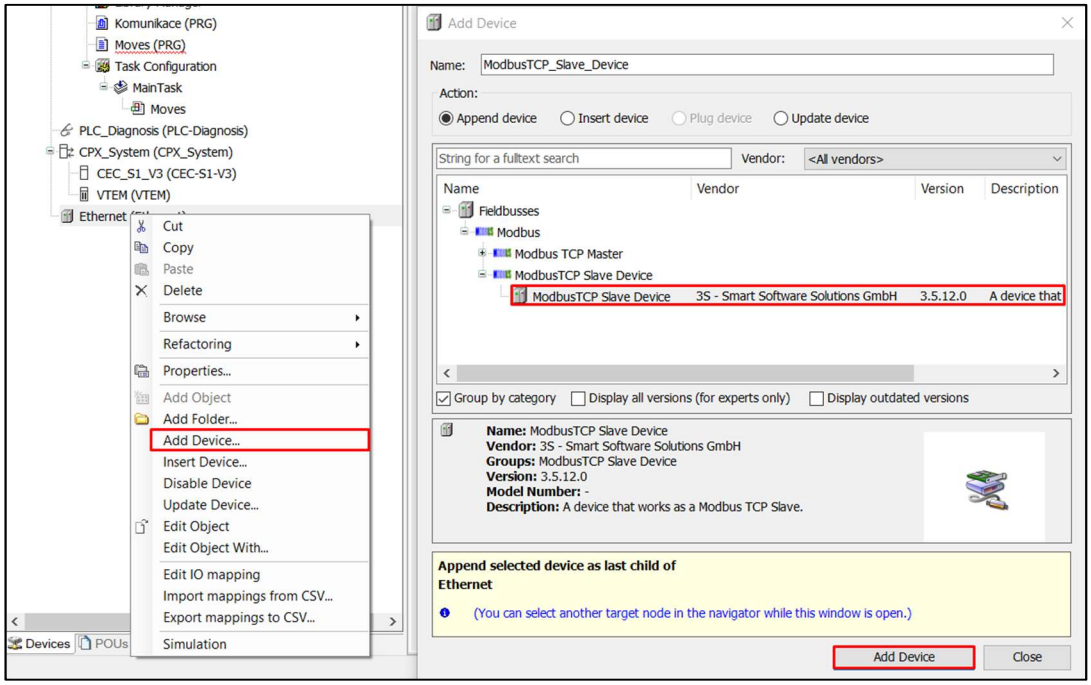

*Obrázek 41 Princip komunikace*

## Vytvoření serveru

Nejdříve je třeba vytvořit server na straně CPX terminálu. V otevřeném projektu v programu CODESYS se klikne pravým tlačítkem na Device > Add Device … > Fieldbusses > Ethernet Adapter > Ethernet > Add Device. Po přidání Ethernet zařízení ještě stejným způsobem charakterizuji funkci adaptéru, že se jedná o ModbusTCP Slave Device (čili server). V záložce Ethernet lze nastavit základní parametry adaptéru, avšak při připojeném počítači k CPX terminálu se tyto hodnoty vyplní automaticky po zvolení rozhraní. Samozřejmě je nutné si tyto hodnoty zapamatovat. V druhé záložce ModbusTCP lze již přímo nastavit vlastnosti serveru, jako jsou například port, identifikátor a počet registrů. Důležitým krokem je definice proměnných v podzáložce I/O Mapping.




*Obrázek 42 Přidání zařízení Ethernet (CODESYS)*




*Obrázek 43 Přidání zařízení ModbusTCP Slave (CODESYS)*

## Node-RED na straně klienta

Dalším úkolem je zajistit správnou funkci Modbus klienta v Node-RED. K tomu využiji otevřenou online knihovnu, ze které stáhnu sadu uzlů node-red-contrib-modbus. [49] Z této sady mne budou nejvíce zajímat uzly *modbus-read*, *modbus-write* a popřípadě uzly *flex*. Jedná se o uzly pro čtení a zápis hodnot na server, v případě uzlů *flex* při zápisu více hodnot najednou.

K jejich použití je nejdříve nutné správně charakterizovat server v nastavení uzlu, prioritně jeho IP adresu (192.168.2.20), port (502) a identifikátor (1). Tyto informace se uloží pod uživatelem přiděleným jménem serveru. Dále je nutné ještě seřídit samotný chování samotných uzlů. Zde se nastavení liší dle použité funkce. Při zápisu nastavuji adresu proměnné, při čtení ze serveru jde hlavně o adresu, množství čtených hodnot a velikost dotazovací periody.

Práce s uzly node-red-contrib-modbus je velice jednoduchá. Při čtení uzel vždy po uplynutí časové periody vypustí zprávu. Hodnota se skrývá na prvním místě v poli msg.payload[0]. Při zápisu uzel zpracuje hodnotu uloženou ve vlastnosti objektu msg.payload a tu pošle na zvolenou adresu serveru.

# 9. Grafické rozhraní

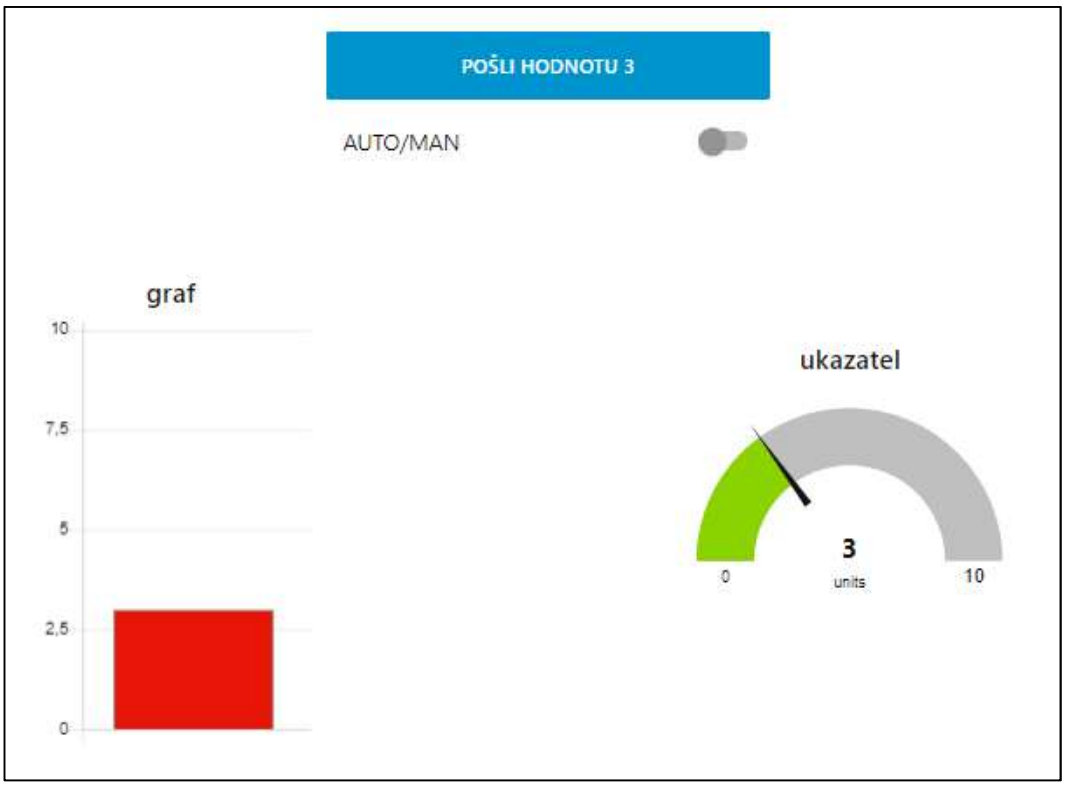
Po zpříjemnění filmového zážitku je další důležitou interakcí projektu s uživatelem samozřejmě grafické rozhraní, které bude především sloužit k ovládání a údržbě. Jeho sekundární úlohou poté bude informovat hloubavější jedince o možnostech zapojení se do rozvoje projektu a také částečného nastínění funkcí.

V tomto bodě také tkví značná výhoda Node-RED, v podobě volně dostupného modulu přístrojové desky *node-red-dashboard*. [50] Celá sada poskytuje mnoho základních ovládacích a vizualizačních prvků v podobě uzlů pro tvorbu interaktivních záložek k prezentaci dat a komunikaci s programem. Při nedostatku uzlů ve standartní knihovně je možné rozšířit paletu o další, uživatelsky vytvořené, či vytvořit svoje vlastní použitím HTML a Javascript. Po vložení prvního uzlu tohoto typu se automaticky vytvoří vizualizační okno s příponou /ui na portu 1880 (výchozí adresa na stejném počítači tedy http://localhost:1880/ui). Ta může obsahovat několik záložek. Tvorba vizualizačních a dispečerských aplikací probíhá stále stejným způsobem v editoru Node-RED, což velmi usnadňuje práci při vytváření projektů.

Pro moji potřebu bude stačit základní balíček uzlů s tlačítky, přepínači, ukazateli, poli pro vkládání textu a další. Pro vložení některého z objektů stačí přetáhnout daný uzel z palety do editoru, vybrat příslušnou záložku umístění a charakterizovat jeho vzhled a programové chování. Pro uspořádání všech grafických elementů v okně je nutné využít záložky v pravém sloupci editoru *dashboard*. Zde lze také upřesnit vzhled stránek a chování.

K rychlému přiblížení tvorby jsou přidány dva obrázky níže. Vrchní obrázek ukazuje pohled v editoru, kde jsou právě umístěny 4 uzly: *button* (tlačítko), *switch* (přepínač), *gauge* (ukazatel) a *chart* (graf). Vpravo je také vidět nastavení uzlu ukazatele. Ve spodním obrázku je již konečné vizualizační okno s vybranými elementy. Při kliknutí na tlačítko se v programu vyšle zpráva do případných dalších napojených uzlů na výstup tlačítka, v tomto případě do uzlu grafu a ukazatele. Po kliknutí na tlačítko oba ukáží hodnotu 3.


*Obrázek 44 Node-RED dashboard - pohled editoru*




*Obrázek 45 Node-RED dashboard - vizualizační okno*

# 10. Separace projektu na funkční celky

Ke snadnější orientaci v celém projektu, pochopení všech hlavních funkcí aplikace nejen z pohledu uživatele, ale také z pohledu programátora a také k nastínění mého časového postupu při tvorbě projektu rozdělím v tomto bodě mé práce celý projekt na menší funkční celky. V následujících částech práce tyto celky přiblížím z pohledu programátora jak na straně Node-RED editoru, tak na straně Festo Motion Terminal VTEM, respektive programovatelného automatu CPX.

Projekt tedy rozdělím na 4 hlavní části, které budou ještě dále rozděleny na jednotlivé funkční celky:

- Přehrávání filmů
    - Příprava systému
    - Režim promítání filmu
- Údržba
    - Manuální režim
    - Diagnostika úniků vzduchu ze systému
    - Řešení problémů VTEM
- Nastavení
- Informace o bezpečnosti a vývoji
    - Bezpečnost
    - Vývoj pohybového souboru

Následující obsah mé práce popisuje můj chronologický postup při tvorbě hlavních funkcí jednotlivých celků i celého projektu. Popis však není primárně určen k edukativním účelům a neslouží jako návod, jak programovat v systému CODESYS či pomocí nástroje Node-RED. Pro snadné pochopení dalších bodů je předpokladem alespoň částečná znalost prostředí CODESYS a jazyku ST (Structured Text) dle mezinárodního průmyslového standartu IEC 61131-3 a také programování v editoru Node-RED. Dalším předpokladem je jisté povědomí o produktu Festo Motion Terminal VTEM. V této práci budou použity některé bloky z knihovny Festo_VTEM_DevCon (v. 3.5.7.221) pro ovládání a komunikaci mezi programovatelným automatem CPX a ventily VEVM. Tyto funkční bloky a práce s nimi byly popsány v teoretické části diplomové práce.

Současně s tvorbou programu v Node-RED jsem zároveň tvořil i grafické rozhraní. Ukázky grafického uživatelského prostředí jsou vždy součástí daných bodů. V kapitole Režim promítání filmu – klávesnice se nachází ukázka vytvoření vlastního uzlu pro použití v prostředí Node-RED.

# 11. Režimy

Po předchozím rozdělení je jasné, že systém bude možné ovládat v několika režimech. Z funkčního a bezpečnostního hlediska mne nejvíce zajímají ty stavy systému, při kterém může dojít k pohybu sedačky. Pro zvýšenou bezpečnost jsem každý z těchto režimů očísloval dle tabulky dole. Této hodnotě také přísluší samostatná proměnná na Modbus serveru označená jako *Output0*, která je každou sekundu aktualizována z počítače. V hlavním programu PLC je poté tato proměnná zpracována a dle její hodnoty je vybrán příslušný stav ventilů Festo Motion Terminal. V nulovém, tedy základním stavu jsou všechny motory nejdříve zasunuty a poté dochází k deaktivování všech ventilů pro zamezení nečekaných pohybů sedačky. Zároveň se tak děje, pokud je narušena komunikace mezi počítačem a CPX automatem, respektive serverem Modbus.

| Režimy | | |
|---|---|---|
| Hodnota proměnné | Popis | MotionApp |
| 0 | základní režim | - STOP |
| 1 | zkouška komunikace | - STOP |
| 2 | příprava na přehrávání | 6 – ECO drive |
| 5 | přehrávání | více |
| 10 | manuální režim | 6 – ECO drive |
| 12 | diagnostika úniků | 12 – Leakage diagnostics |

*Tabulka 3 Očíslování režimů a jejich význam*

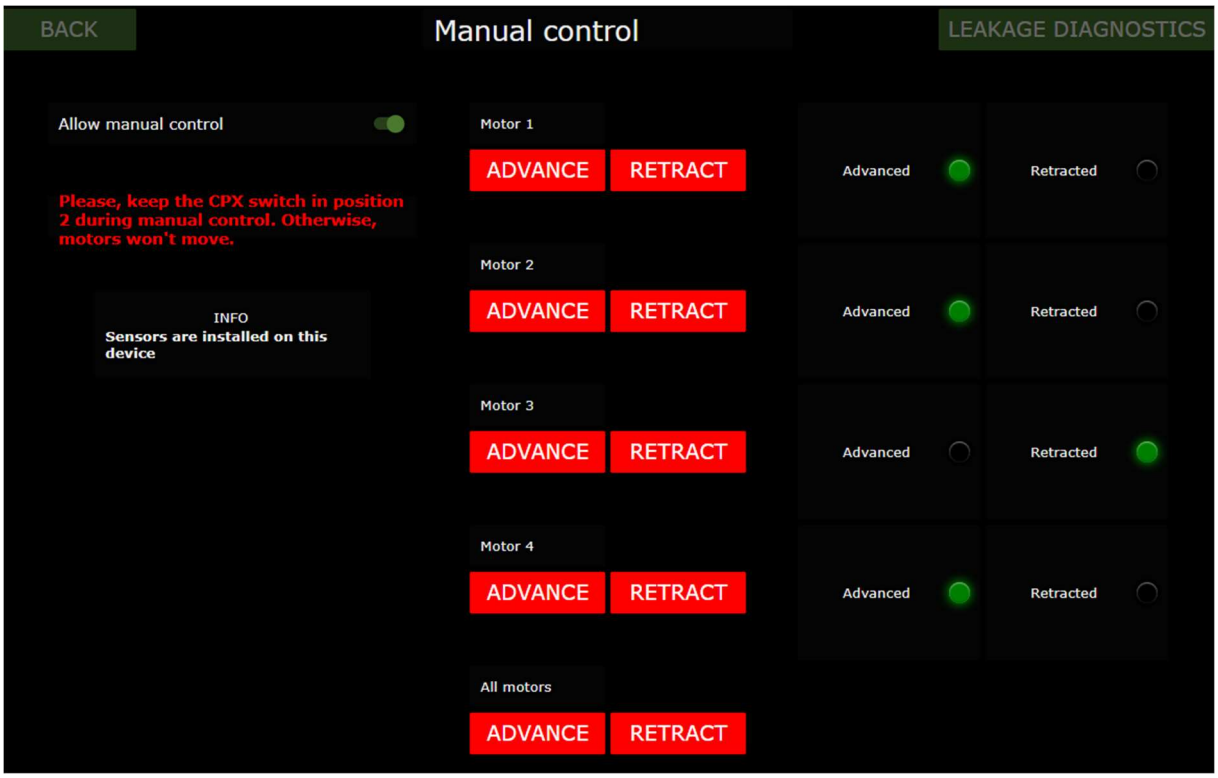
# 12. Údržba

Jako první úkol je žádoucí zprovoznit dispečerský režim a prostředí údržby. Tento krok má hned několik logických vysvětlení. Zaprvé ozkoušet funkčnost komunikace mezi počítačem a CPX terminálem, tzn. mezi klientem a serverem. Druhým důvodem se stává kontrola funkčnosti jednotlivých ventilů a ladění rychlostí motorů s použitím několika různých MotionApp. Největší přínos však tento krok přináší při pozdějším programování daleko složitějších částí, jelikož skrze funkce této záložky je možné diagnostikovat chyby při tvorbě programu. V této části jsou obsaženy dva režimy, a to manuální režim a režim diagnostiky úniků.

## Manuální režim

Pro manuální ovládání každého ze 4 motorů jsem vytvořil dispečerské okno s několika tlačítky pro vysunutí (advance) a zasunutí (retract) motorů jednotlivě či všech dohromady. Při nainstalovaných koncových senzorech lze vidět vpravo i svítící kontrolky ukazující aktuální stav motorů. Při chybějících senzorech jsou tyto kontrolky skryty. Pro

povolení manuálního ovládání motorů je nutné použít přepínač v levém horním rohu. Tento režim slouží primárně pro diagnostiku pohybu motorů. Při nainstalovaných koncových senzorech může být zkušebně použit k ověření správné instalace snímačů.

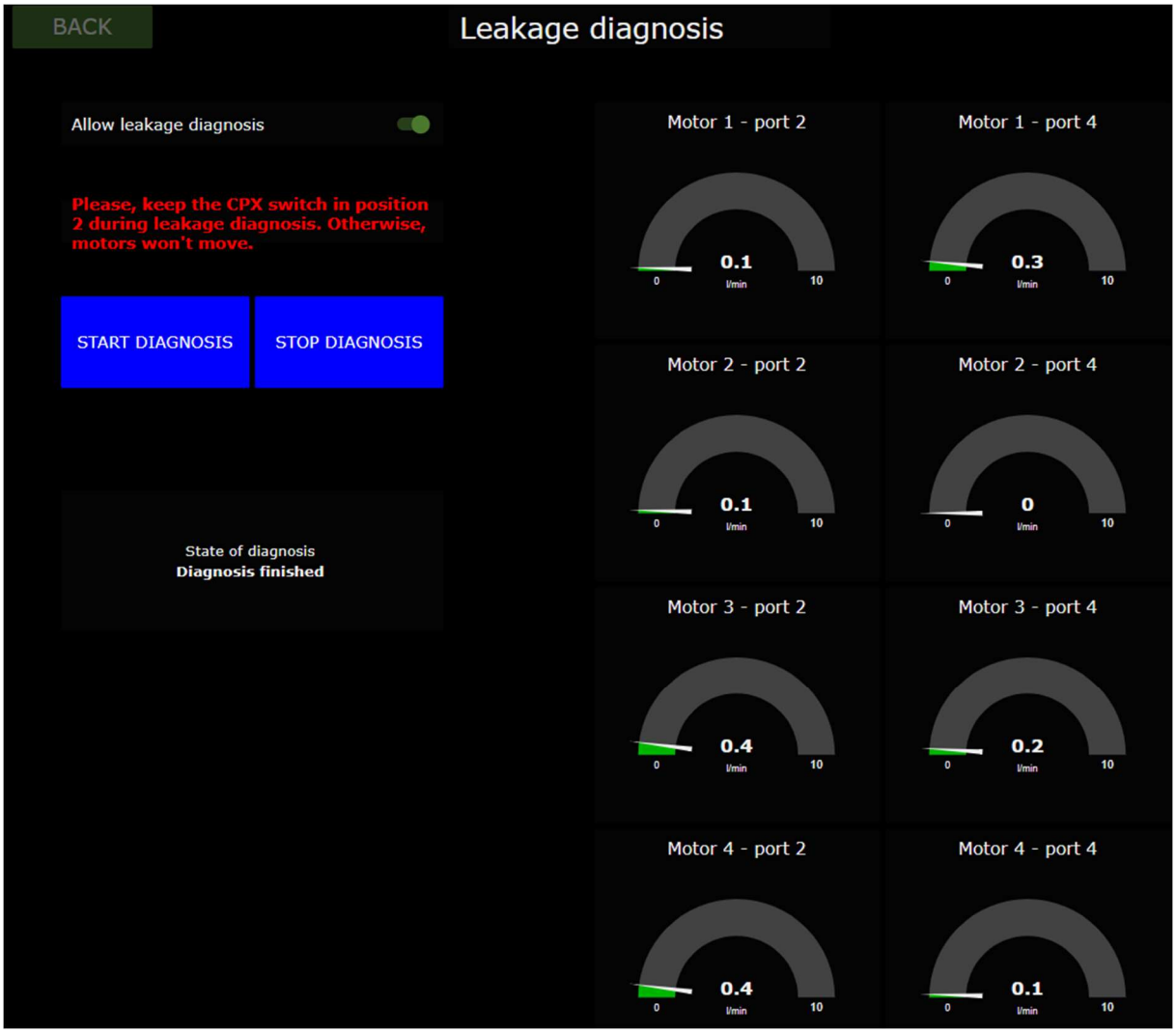



*Obrázek 46 Dispečerské okno manuálního režimu*

Po přepnutí do manuálního režimu povolí hlavní program programovou organizační jednotku *Manual_Control*, která změní stav ventilů na požadovanou MotionApp číslo 6 s názvem pohyb ECO (ECO drive). V tomto stavu lze měnit úroveň otevření škrtícího ventilu pomocí vstupu *iSetpointValue2* (její hodnota však zůstává pro tento režim neměnná) funkčního bloku *FB_ValveControl* každého motoru. Pomocí vstupu *bySetAppControl* je ovládán stav motorů mezi vyjetím a zajetím. Tato hodnota je již získávána z Modbus serveru, kde je aktualizována z prostředí Node-RED ovládacími tlačítky. Pokud jsou nainstalovány koncové senzory na válcích, do komunikačních registrů je zapisován stav každého z motorů z výstupu *byActualAppState*, který CPX automat získá z dat přicházejících z digitalizovaných ventilů.

## Diagnostika úniků

Vizualizační okno pro ovládání a prezentaci výsledků diagnostiky úniků se skládá z 8 ukazatelů. Dvojice ukazatelů zobrazuje výsledky každého ze 4 ventilů na jeho obou výstupních portech v litrech za minutu.




*Obrázek 47 Vizualizační okno diagnostiky úniků*

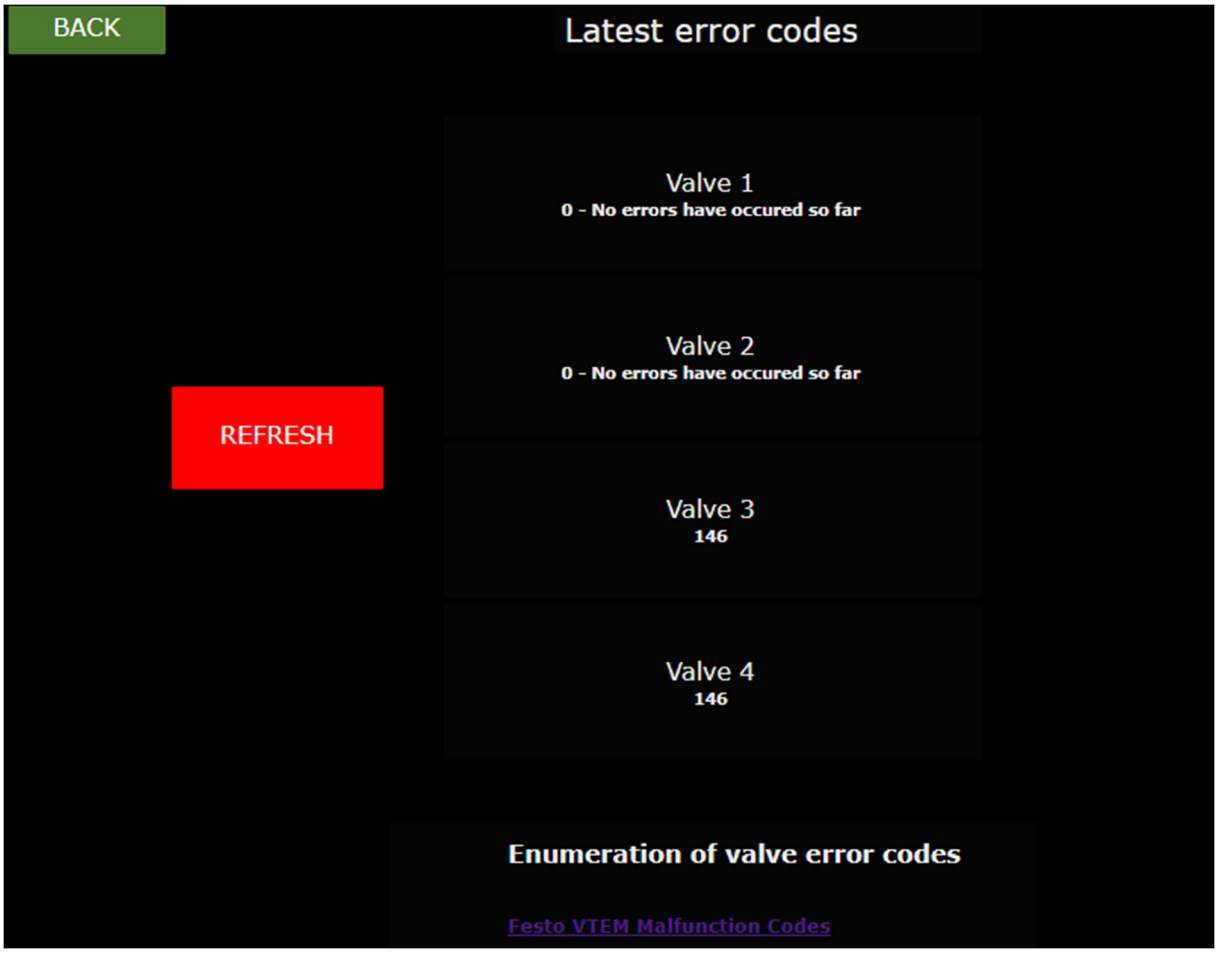
K získání dat a zpracování výsledků slouží MotionApp číslo 12 – Diagnostika úniků (Leakage diagnostics). Chování ventilů je řízeno proměnnou *bySetAppControl* mezi třemi stavy: aktivace diagnostiky, zastavení a referenční pohyb. Před použitím celé aplikace je nutné přímo nahrát do Festo Motion Terminal data o připojeném typu motoru a jeho zdvihu. Tento krok je možné provést buď funkčními bloky pro parametrizaci ventilů z knihovny Festo_VTEM_DevCon, nebo ho lze jednodušeji splnit připojením počítače přímo k terminálu CTMM a v jeho webovém rozhraní vybrat příslušný výrobek Festo a jeho další parametry. Před odpojením se od terminálu je ještě nezbytné uvést ventil do stavu referenčního pohybu, kdy se uloží referenční hodnota, se kterou budou všechny nastávající diagnostiky porovnávány a výsledky jsou prezentovány jako rozdílové hodnoty oproti právě té referenční, uložené v paměti terminálu. Tento postup však není v mém projektu zahrnut, jelikož je pro uživatele zbytečný a mohl by se provádět ještě před instalací celého systému.

Při podezření výskytu úniku vzduchu ze systému je poté možné tento problém diagnostikovat skrz toto vizualizační okno aplikace. Celý proces řídí sama MotionApp 12, kdy dochází k několika různým pohybům motoru a data do CPX terminálu jsou poté dostupné skrze výstupy *iActualValve1* a *iActualValve2* funkčního bloku. Status diagnostiky je možné sledovat v okně na levé straně. Po skončení celého procesu se výsledky promítají ve vizualizačním okně. Data jsou přenášeny pomocí 8 proměnných registru, plus 1 proměnná slouží ke sledování aktuálního stavu průběhu diagnostiky z výstupu *byActualAppState*. Přepočet mezi hodnotou proměnné a reálnou veličinou l/min lze najít v tabulce Cbus-Protocol VTEM. [28]

### Diagnostika chyb VTEM

Do této chvíle je možné diagnostikovat pohyby motorů skrze dispečerské okno manuálního režimu. Bohužel dosud není možné identifikovat jakékoliv vnitřní příčiny chyb způsobující nepohyblivost motorů. Bylo by užitečné nejen pro uživatele potýkající se s případnými problémy, ale také při mém dalším programování projektu, získávat zpětnou vazbu z ventilů VTEM při případných chybách způsobených například špatným nastavením ventilů. K tomuto úkolu jsem využil výstup z funkčního bloku pro komunikaci s ventily VTEM *iErrorCode*.

Na tento výstup, který vždy charakterizuje hodnotu poslední zaznamenané chyby ventilu VTEM, je navázána proměnná. Ta je ukládána do Input registru na Modbus TCP server, ze kterého ji Node-RED přečte při každém stisknutí tlačítka „Refresh“ v okně diagnostiky chyb. Pro každý ventil je nutné vytvořit jednu proměnnou. Ty poté okupují poslední čtyři místa v Input registru, na serveru pojmenované jako *Output10* až *Output13*. Tato funkce je nezávislá na aktuálním režimu a je tak dostupná v jakékoliv situaci. Pokud je na výstupu z funkčního bloku hodnota 0, v systému VTEM nebyla dosud zaznamenána žádná chyba. Celá enumerace hodnot je dostupná při kliknutí na hypertextový odkaz přímo v okně. [51]




*Obrázek 48 Okno diagnostiky chyb*

# 13. Přehrávání filmů

Hlavní cíl celého projektu je právě tato část – návrh a naprogramování systému pro pohyb domácí sedačky při promítání filmů po vzoru již velkých systémů umístěných v kinech. V první části se věnuji přípravě k promítání, která je do aplikace zakomponována z bezpečnostních a funkčních důvodů. V druhé fázi tohoto bodu je již pohled na hlavní programu ovládající pohyby sedačky a upřesnění základních použitých funkcí.

## Příprava systému

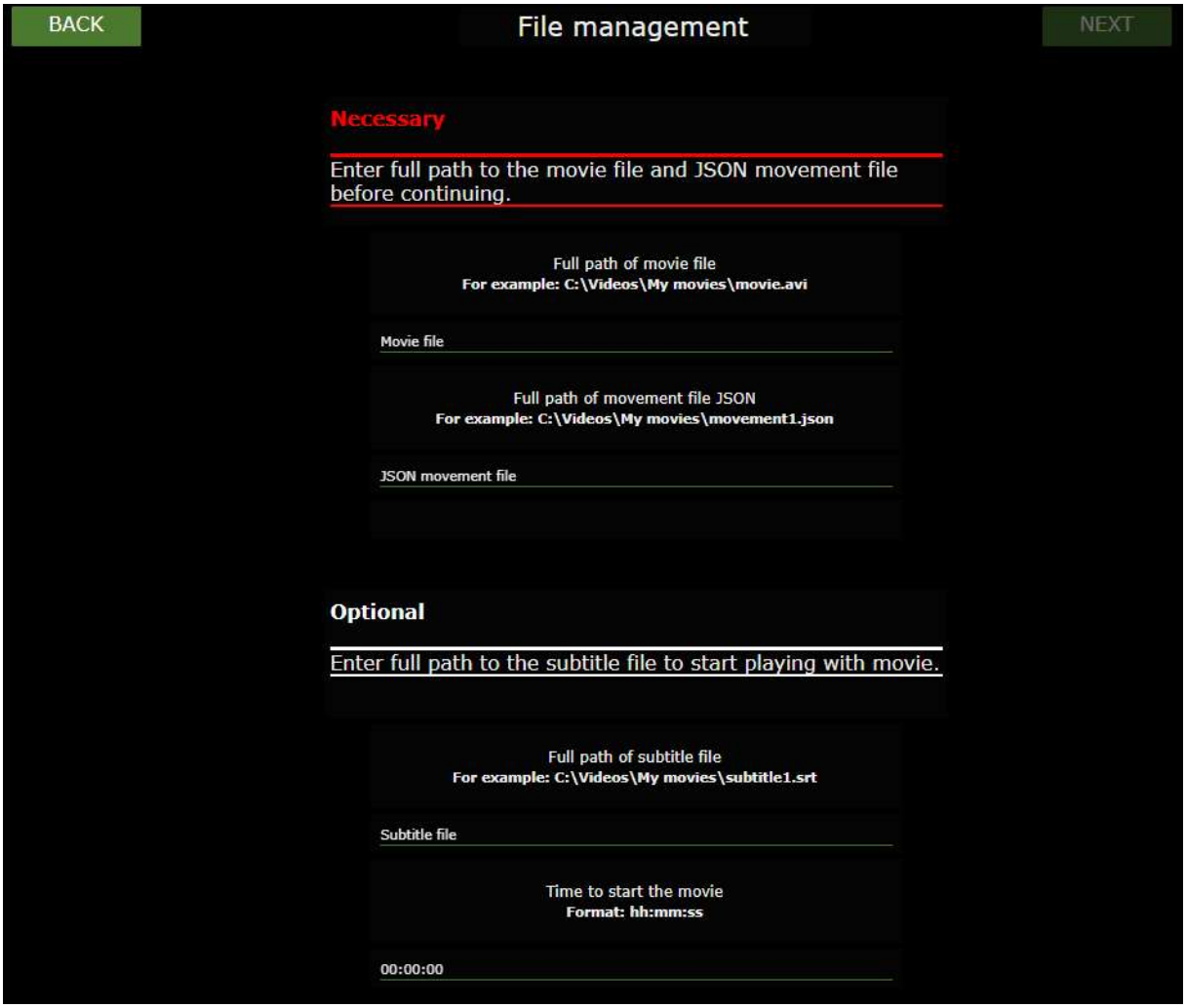
První část přípravy spočívá ve zkoušce komunikace mezi CPX automatem a aplikací Node-RED běžící na počítači. Ta funguje v jednoduchém odeslání hodnoty na Modbus server do Holding registru a její zpětnou kontrolou čtením z Input registru. Po celý režim přehrávání je nutné z bezpečnostních důvodů mít kruhový přepínač umístěný na CPX modulu v poloze 2. Pokud se tak neděje, systém tuto skutečnost na záložce kontroly komunikace hlásí a také se objevuje červeně zbarvené upozornění v pravém horním rohu po celou dobu zapnuté aplikace. Pokud vše proběhne v pořádku, uživatel může pokračovat dále na záložku zkoušky pohybů. Aplikace také v tomto okně

upozorňuje uživatele, že musí mít nastavenu IP adresu adaptéru Ethernet na dosažitelnou (viz Návod ke zprovoznění systému).

V dalším okně je nutné prověřit správné propojení pneumatických motorů s ventily VEVM a jejich funkčnost pomocí jednoduchého pohybu nahoru. Pokud jsou nainstalované koncové snímače motorů, systém na straně Node-RED čte skrze Modbus server stav snímačů získaný z výstupu funkčního bloku *byActualAppState* stejně jako v manuálním režimu. Při splnění podmínky u všech ventilů se uživateli odemkne tlačítko pro pohyb dolů a kontrola se opakuje. Poté je povoleno pokračovat dále. Při chybějících snímačích kontrolu pohybů provádí vizuálně sám uživatel.

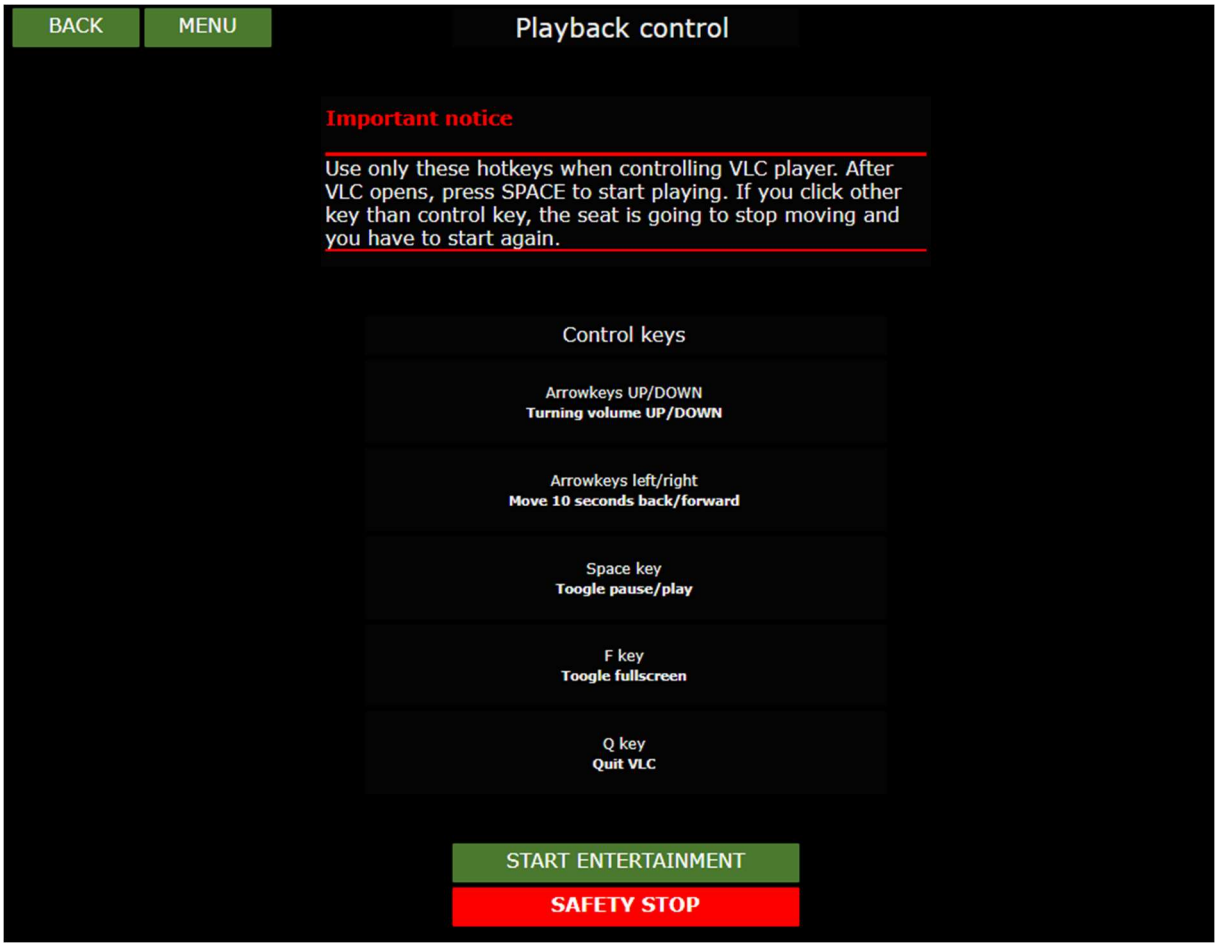

*Obrázek 49 Okno kontroly pohybů motorů*

Třetí část se již netýká kontroly, ale zadání cesty k filmovému souboru pro bezproblémové spuštění přehrávače na počítači. Zároveň je také nutné zadat cestu k pohybovému souboru (viz kapitola Informace k vývoji souboru k filmu). Tyto vrchní dva parametry je nutné zadat pro další pokračování. Níže je možné vyplnit ještě další dva volitelné parametry: cestu k uloženému souboru s titulky a počáteční čas spuštění filmu. Tyto dvě kolonky je však možné vynechat. Správnost zadané cesty k souboru s pohyby kontrolují uzly v Node-RED a při neexistenci souboru uživatele nepustí dále.




*Obrázek 50 Okno spravující soubory*

Na poslední záložce před samotným spuštěním přehrávání se skrývá návod k ovládání aplikace v režimu přehrávání s otevřeným VLC přehrávačem. V přehrávacím režimu lze z bezpečnostních důvodů používat pouze zobrazené klávesy. Jakékoliv použití jiné klávesy má za následek okamžité zastavení veškerých pohybů sedačky a proces spouštění je nutné absolvovat znovu. Detekci zmáčknutých kláves hlídá předem vlastně vytvořený uzel v Node-RED (viz kapitola Režim promítání filmu - klávesnice).




*Obrázek 51 Okno spouštění přehrávání*

Jak jsem již v předchozí kapitole výběru software určil, k přehrávání filmů jsem zvolil program VLC Media Player. Ten má značnou výhodu proti ostatním volně dostupným programům v obsáhlých možnostech režimů spuštění z příkazového řádku. Toho jsem využil i zde. Z předchozí záložky jsou uloženy všechny informace, nutné i volitelné, ke spuštění VLC přehrávače. Dle dokumentace VLC [52] je vytvořen kód v uzlu *function*, která umožňuje naprogramovat jakoukoliv funkci proti příchozí zprávě v jazyce JavaScript:

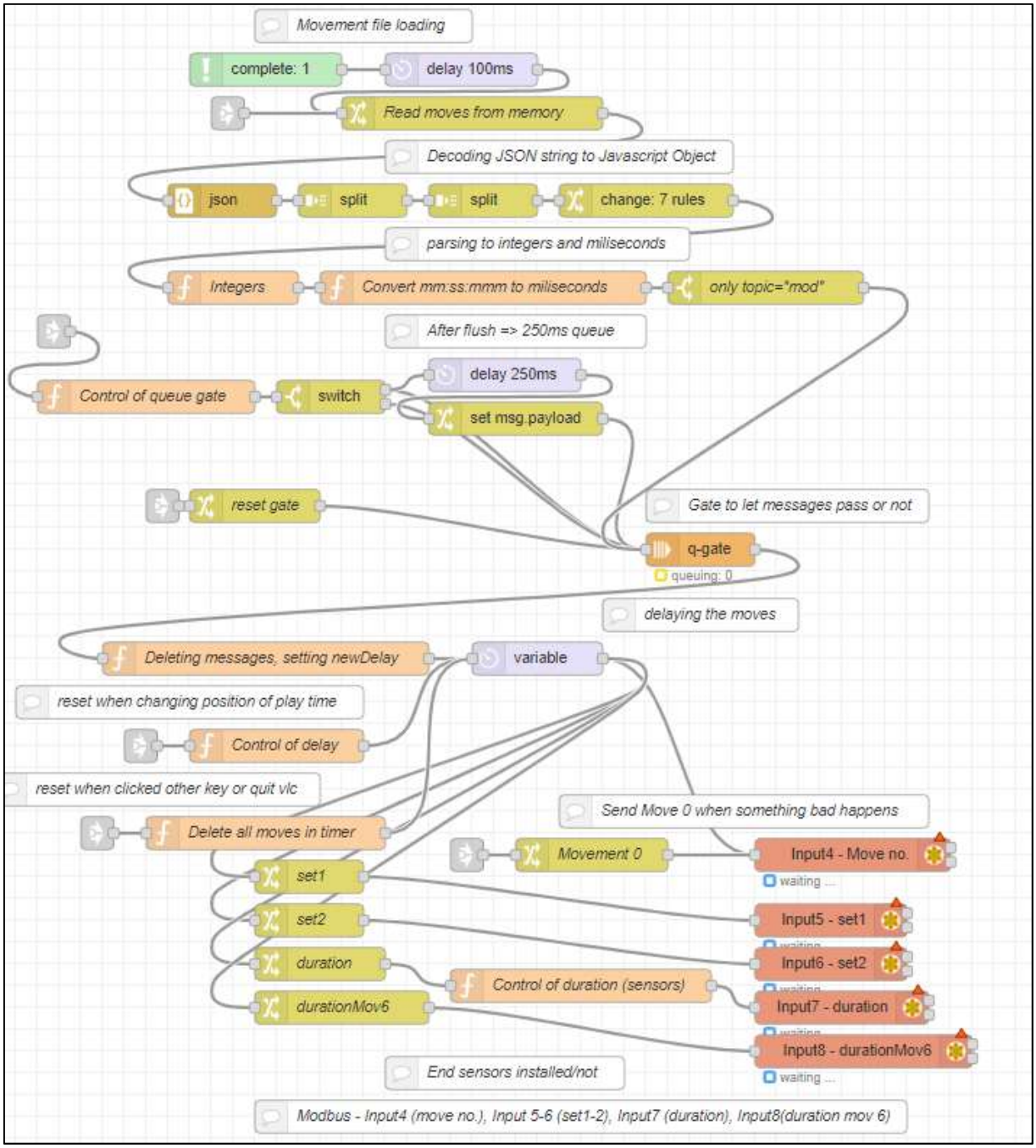
```
var vlc = global.get('vlcpath');
var movie = global.get('moviepath');
var subtitle = global.get('subtitlepath');
var start = global.get('starttime');
var d;

if(subtitle){
    d = `--fullscreen --start-paused --qt-minimal-view --play-and-exit --key-quit=\"q\" --sub-file=\"${subtitle}\"`;
}else{
    d = `--fullscreen --start-paused --qt-minimal-view --play-and-exit --key-quit=\"q\"`;
}
if(typeof(start)=="number" && start > 0){
    time = start/1000;
    msg.payload =`"${vlc}" "${movie}" ${d} --start-time=${time}`;
}else{
msg.payload =`"${vlc}" "${movie}" ${d}`;
}
return msg;

```

*Obrázek 52 Kód pro spuštění VLC z příkazového řádku v Node-RED, uzel function*

Tato funkce po doražení zprávy o zmáčknutí tlačítka pro start přehrávání nejdříve načte z úložného prostoru pro sdílení mezi všemi uzly umístěnými v programu všechny potřebné i volitelné parametry z okna správce souborů a poté dle existence zadaných parametrů vytvoří zprávu, příkaz, který je skrze Node-RED nativní uzel *exec* spuštěn v příkazovém řádku systému Windows. VLC přehrávač se poté spustí v minimálním režimu přes celou obrazovku promítacího monitoru s načteným filmem a případně i titulky ve zvolený čas stopy. Chybné zadání cesty k souboru filmu je ošetřeno dalšími uzly, které uživatele na tuto skutečnost upozorní a vrátí ho zpět na předchozí stránku. Při chybně zadané cestě k přehrávači VLC je uživatel nasměrován do záložky nastavení k opravě zadání. Stopa filmu je z technických důvodů na začátku pozastavena. Pro vypnutí programu VLC byla zvolena klávesa „Q", zkratka pro „quit"(česky ukončit).

### Režim promítání filmu – dekódování souboru

Po bezproblémovém spuštění VLC přehrávače vyšle Node-RED informaci CPX terminálu o změně režimu na režim přehrávání. V tuto chvíli je také načten soubor pohybů v nezformátované textové podobě. Tento soubor (viz také kapitola Informace o vývoji pohybového souboru) je původně zformátován jako JSON (JavaScript Object Notation), kde každý objekt v poli hlavního JSON objektu tvoří samostatnou zprávu o změně pohybu sedačky. Pomocí několika uzlů typu *json* a *split* je text dekódován a rozdělen na jednotlivé zprávy s několika atributy, charakterizující změny pohybu, putující dále programem. V dalším kroku jsou převedena všechna čísla z formátu dat *string* na *int* a je pro každou zprávu vypočteno zpoždění v milisekundách z časového formátu minuty:sekundy:milisekundy, přítomného v původním souboru. V tomto formátu již zprávy putují do datové struktury typu fronta, uzel *q-gate*, která je ovládána uzly umístěnými nalevo od ní. Tyto uzly vysílají ovládací zprávy pro pozastavení, vypuštění a mazání fronty v reakci na přijaté podněty od uživatele skrze klávesnici. Po vypuštění se zprávy dostávají do uzlu *delay*, kde je přečten jejich atribut msg.delay. Tento uzel již kontroluje tok zpráv dle zadaného zpoždění každé z nich. Zároveň je ovládán dalšími uzly, které upravují velikost zpoždění v závislosti na pohybu uživatele po stopě filmu klávesami. Po překonání zpožďovacího uzlu už putují zprávy samostatně do připravených dekodérů, kde jsou jejich atributy pro řízení pohybů rozebrány na jednotlivé hodnoty a zapsány na adresy do Holding registru serveru Modbus TCP. Celý tento popsaný proces v pohledu programovacího prostředí Node-RED lze vidět níže.




*Obrázek 53 Kód Node-RED pro zpracování pohybového souboru a zaslání dat do CPX*

## Režim promítání filmu – klávesnice

Druhým nejdůležitějším úkolem je zpracování uživatelských stisků klávesnice při ovládání stopy filmu. Klávesnici, jako hlavní zdroj komunikace mezi uživatelem a VLC přehrávačem, jsem vybral z důvodu již existující možnosti takto ovládat přehrávač. Je však nutné podobně implementovat zachycení stisků klávesnice také v Node-RED. Bohužel ani po několika pokusech zprovoznit některou z již existujících sad uzlů vhodnou pro tento účel z otevřené knihovny se mi tento úkol nepodařil. Jediné možné řešení se tedy naskytuje vytvořit si vlastní uzel pro zachycení a rozpoznání stisků klávesnice.

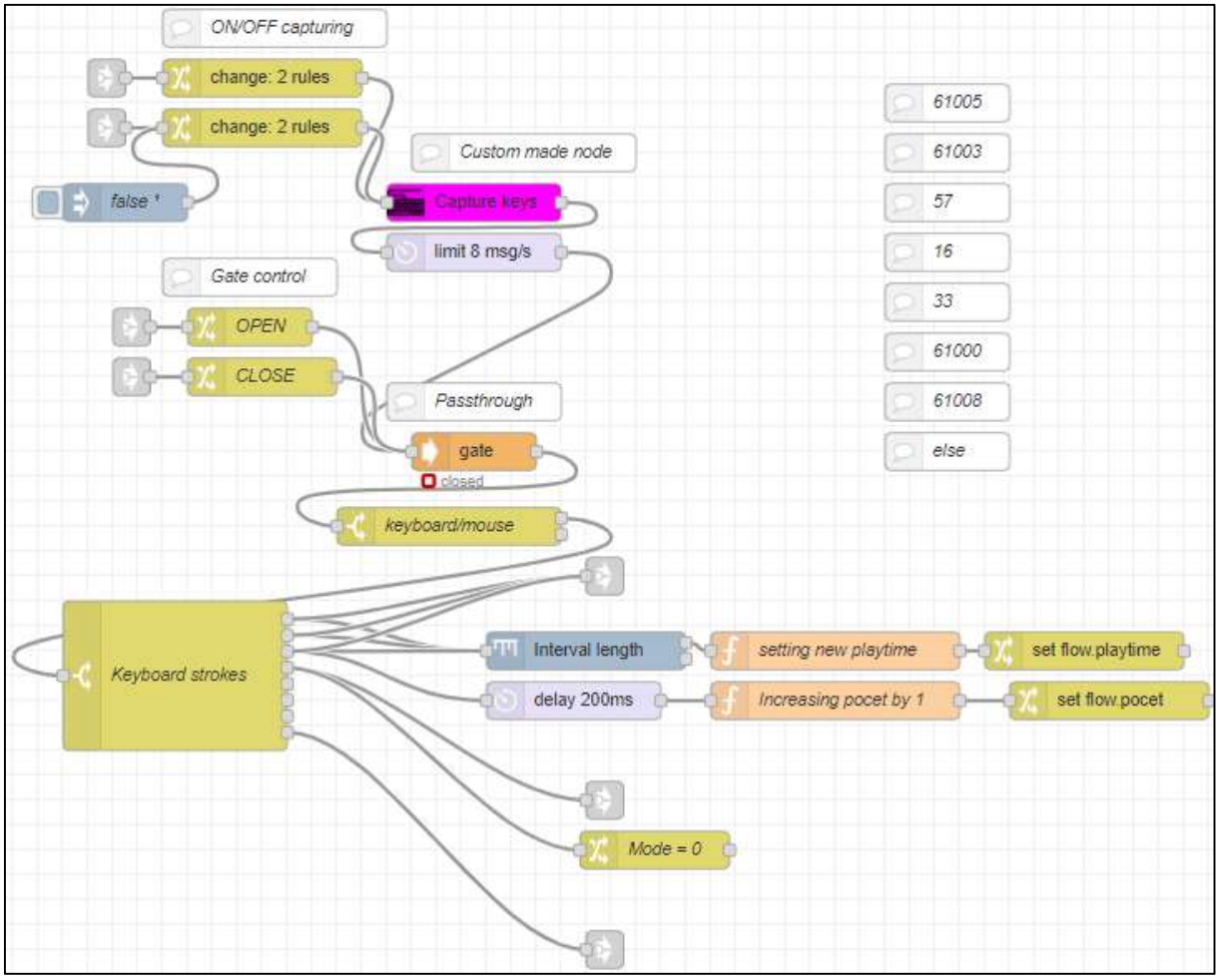
Jako první krok jsem hledal vhodný modul pro Node.js, který by dokázal plnit funkci „odposlouchávače“ stisků klávesnice a popřípadě i myši. Při hledání jsem se zároveň zaměřil na co nejmenší počet, nejlépe nulový, dalších modulů, na kterých by byl tento vybraný závislý. Po chvílích hledání jsem se rozhodl pro modul *iohook* dostupný z [53], který splňuje veškeré mé předpoklady.

Mým dalším krokem bylo vytvoření funkčního uzlu s využitím návodu na oficiálních stránkách Node-RED. [54] Nejdříve bylo nutné vytvořit 3 soubory:

- package.json – standardizovaný soubor popisující obsah pro moduly Node.js
- XXX.js – vlastní JavaScript funkce
- XXX.html – soubor pro definici uzlu v editoru a přiložení pomocných informací

Nejzajímavější částí pro mé účely je samozřejmě tvorba vlastní funkce. Ta je dle již zmiňovaného návodu společně s dokumentací modulu *iohook* vytvořená tak, aby při jakékoliv události vyvolané myší či klávesnicí uzel vyslal zprávu s informací o události. Ta se skládá z číselného kódu právě stisknuté klávesy či čísla tlačítka myši a dodatkové informace, zda byla událost vyvolána tlačítkem myši či klávesnice. Dále lze tento uzel deaktivovat vysláním zprávy na vstup v definované struktuře standardně používané pro většinu uzlů v Node-RED. Po dokončení veškerých prací s dokumentací a odstranění chyb jsem mohl publikovat tuto sadu uzlů k dispozici ostatním uživatelům prostřednictvím npm. Tento krok také velmi ulehčuje instalaci vlastně vytvořeného uzlu pro ostatní uživatele, kteří si nainstalují tento projekt, jelikož je volně stažitelný z knihovny npm a knihovny Node-RED. [55]

Níže na obrázku je pohled ze strany editoru Node-RED při promítání filmu přímo na zpracování těchto událostí právě s pomocí tohoto vlastně vytvořeného uzlu s názvem *Capture keys*. Na začátku jsou vytvořeny ovládací zprávy pro aktivaci/deaktivaci v závislosti na aktuálním režimu systému. Dále je umístěn uzel *gate* pro kontrolu toku zpráv dle stopy filmu a poté již rozdělovací uzly *switch* pro dělení zpráv dle kódu jednotlivých kláves. Ovládací funkce kláves s jejich číselným kódem se nachází v tabulce níže.




*Obrázek 54 Kód Node-RED pro zpracování stisků kláves*

| Seznam ovládacích kláves | | |
|---|---|---|
| Klávesa | Význam | Číselný kód |
| mezerník | pozastavit/spustit | 57 |
| šipka nahoru/dolů | hlasitost | 61000/61008 |
| šipka doprava/doleva | 10 sekund dál/zpět | 61005/61003 |
| Q | vypnutí přehrávače | 16 |
| F | vynucení celé obrazovky | 33 |

*Tabulka 4 Seznam ovládacích kláves*

## Režim promítání filmu – pohyby

V tomto bodu se již přesunu do vývojového prostředí CODESYS a programování programovatelného automatu CPX.

Po změně režimu hlavní program povolí programovou organizační jednotku *Moves*, ve které je naprogramováno několik MotionApp pro kontrolu a zprostředkování několika typů pohybů. Při každé změně pohybu je nejdříve vybrána MotionApp pomocí jedné proměnné pro všechny ventily skrze vstup funkčního bloku *bySetValveMode*. S ní

je také nutné zadána kontrolní proměnná MotionApp na vstup funkčního bloku *bySetAppControl*, která v tomto kroku zapříčiňuje zablokování proudění vzduchu skrz výstupní porty ventilu hodnotou 0. Dále jsou v programu vázány globální proměnné ovládající chování MotionApp na proměnné umístěné v registrech Modbus serveru a také hodnota trvání, která je poté využita v časovačích typu TON ke kontrole změny směrů pohybů pohonů. Nakonec jsou vynulovány všechny stavy nekonečných smyček ostatních pohybů.

Jak jsem již zmínil výše, všechny pohyby, poté co je inicializována změna pohybu, jsou naprogramovány jako nekonečné smyčky. Prvním příkladem, uvedeným částečně níže, je kód pohybu číslo 2, při kterém se sedačka natáčí střídavě vpravo a vlevo. Tento pohyb využívá MotionApp číslo 5 – škrcení přívodu a odvětrání (supply and exhaust air flow control).

Počátek instrukce CASE OF v každém pohybu slouží v prvním cyklu PLC pro inicializaci ventilů VEVM výše uvedeným způsobem, a tak je nevyužit. V dalším kroku jsou již změněny hodnoty proměnných ovládající porty každého z ventilů, dle příslušného pohybu. V tomto případě jsou motory 1 a 2 vysunuty, motory 3 a 4 zůstávají zablokovány a program čeká na signál z koncových snímačů (skrze výstupní hodnotu funkčního bloku *iActualAppState*) nebo na změnu výstupu na časovači TON, který je řízen hodnotou z pohybového souboru. Po splnění jedné z těchto podmínek se hodnota proměnné ovládací CASE OF opět zvýší do dalšího kroku, kde se motory 3 a 4 vysunou a motory 1 a 2 zůstanou zablokovány a celý proces se opět opakuje se zasouváním motorů. Na konci smyčky se opět program vrátí na její začátek a vzniká tak nekonečná zpětná smyčka.

Tato zobrazená konkrétní smyčka (na obrázku níže) je také ve velice podobné struktuře použita u pohybů číslo 1 a 3. Rozdílem je pouze různé pořadí vysouvání a zasouvání motorů. Všechny tyto pohyby mají společnou vlastnost, a to konstantní volitelnou rychlost motorů při vysunutí i zasunutí. Ta je ovládána primárním a sekundárním škrcením vzduchu pomocí proměnných *GVL.VX__Set1* a *GVL.VX__Set2*, ovládající hodnoty vstupů *iSetPointValue1* a *iSetPointValue2* všech ventilů současně, které jsou tím napojeny na příchozí hodnoty do Modbus registru z dekódovaného pohybového souboru.

```
    ELSIF (Movement = 2) THEN
// Set MotionApp and AppControl
        GVL.All_Mode := 5;
        GVL.V1_AppC := 0;
        GVL.V2_AppC := 0;
        GVL.V3_AppC := 0;
        GVL.V4_AppC := 0;
// Set values for MotionApp
        GVL.V1_Set1 := Set1;
        GVL.V1_Set2 := Set2;

        GVL.V2_Set1 := Set1;
        GVL.V2_Set2 := Set2;

        GVL.V3_Set1 := Set1;
        GVL.V3_Set2 := Set2;

        GVL.V4_Set1 := Set1;
        GVL.V4_Set2 := Set2;
// Set timer duration
        Dur1 := Duration;
        Dur2 := Duration;
// Set other CASE variables to 0
        StateMov0 := 0;
        StateMov1 := 0;
        StateMov3 := 0;
        StateMov4 := 0;
        StateMov5 := 0;
        StateMov6 := 0;

        CASE StateMov2 OF
            0:
                    Tin1 := FALSE;
                    Tin2 := FALSE;
                    StateMov2 := 5;
            5:
                    GVL.V1_AppC := 0;
                    GVL.V2_AppC := 1;
                    GVL.V3_AppC := 0;
                    GVL.V4_AppC := 1;
                    Tin1 := TRUE;
                IF ((GVL.V2_AppState = 4) AND (GVL.V4_AppState = 4)) OR Tout1 THEN
                    StateMov2 := 10;
                END_IF
            10:
                    GVL.V1_AppC := 1;
                    GVL.V2_AppC := 0;
                    GVL.V3_AppC := 1;
                    GVL.V4_AppC := 0;
                    Tin2 := TRUE;
                    Tin1 := FALSE;
                IF ((GVL.V1_AppState = 4) AND (GVL.V3_AppState = 4)) OR Tout2 THEN
                    StateMov2 := 15;
                END_IF
```

*Obrázek 55 Část ST kódu pohybu č. 2 v programovém prostředí CODESYS*

Další skupinou pohybů tvoří pohyby číslo 4 a 6. Ty využívají MotionApp číslo 2, proporcionální regulaci průtoku (proportional directional control valve). Pomocí vstupu *bySetAppOption* je vybrán typ ventilu 4/3, jehož poloha je ovládána pomocí vstupu *iSetPointValue1*. Volitelnou hodnotou v pohybovém souboru je pro tento pohyb krajní poloha ventilu. Zpětná smyčka je založena na změně polohy proporcionálního ventilu z krajní polohy ke střední. Po dosažení nulové polohy je nastaven na opačnou krajní pozici a proces je opakován. Tímto způsobem se zapříčiní změna rychlosti pohybu v čase a sedačka „pluje“. Při pohybu 4 všechny motory pracují ve stejném rytmu, a tak se sedačka hýbe celá nahoru a dolů. U pohybu 6 pracují dvojice motorů v opačných fázích a sedačka se naklání dopředu a dozadu.

```
    ELSIF (Movement = 4) THEN
// Set MotionApp and AppControl
        GVL.All_Mode := 2;
        GVL.V1_AppO := 14;
        GVL.V2_AppO := 14;
        GVL.V3_AppO := 14;
        GVL.V4_AppO := 14;
        GVL.V1_AppC := 3;
        GVL.V2_AppC := 3;
        GVL.V3_AppC := 3;
        GVL.V4_AppC := 3;
// Set other CASE variables to 0
        StateMov0 := 0;
        StateMov1 := 0;
        StateMov2 := 0;
        StateMov3 := 0;
        StateMov5 := 0;
        StateMov6 := 0;
// Set boundary for floating from Modbus
        Mov4Boundary := Set1;

        CASE StateMov4 OF
            0:
                StateMov4 := 5;
                Mov4Set := Mov4Boundary;
            5:
                IF Mov4Set > 0 THEN
                    Mov4Set := Mov4Set - 20;
                ELSE
                    StateMov4 := 10;
                END_IF
            10:
                Mov4Set := (-1)*Mov4Boundary;
                StateMov4 := 15;
            15:
                IF Mov4Set < 0 THEN
                    Mov4Set := Mov4Set + 20;
                ELSE
                    StateMov4 := 0;
                END_IF
        END_CASE
// Set values for MotionApp
        GVL.V1_Set1 := Mov4Set;
        GVL.V2_Set1 := Mov4Set;
        GVL.V3_Set1 := Mov4Set;
        GVL.V4_Set1 := Mov4Set;
```

*Obrázek 56 ST kód pohybu č. 4 v programovém prostředí CODESYS*

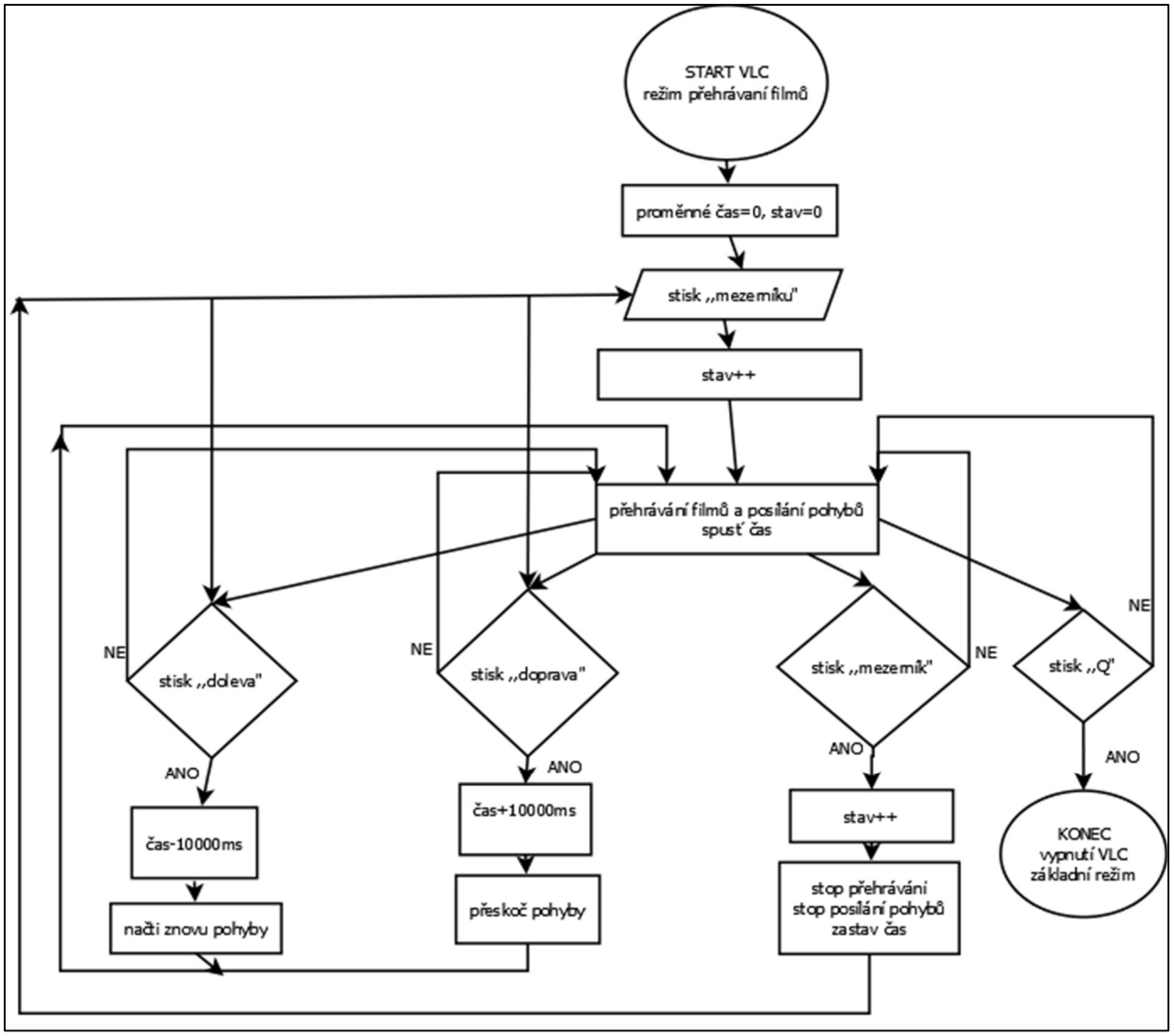
V poslední typové skupině se nachází samotný pohyb číslo 5. Ten se nijak neliší od pohybu 3 sekvencí pneumatických motorů ani kódem. Rozdíl je v použití jiné MotionApp, a to konkrétně číslem 8 – volitelný tlak ECO (selectable pressure level ECO). Tento mód umožňuje nastavit úroveň tlaku na volitelnou hodnotu a současně škrtit přívod vzduchu a regulovat tak rychlost. Toho by bylo možné využít při krátkých pohybech sedačky nahoru a dolů. Tlak je nastavitelný hodnotou *iSetPointValue2*, škrcení *iSetPointValue1*.

## Seznam pohybů

| Seznam dostupných pohybů | | |
|---|---|---|
| Pohyb | MotionApp | Vysvětlivka |
| 0 | č. 5 | základní pozice, pohyby pozastaveny, motory zasunuty |
| 1 | č. 5 | natáčení vpřed a vzad, vždy 2 motory naráz |
| 2 | č. 5 | natáčení vpravo a vlevo, vždy 2 motory naráz |
| 3 | č. 5 | natáčení se dokola, motory vykonávají pohyb postupně po jednom |
| 4 | č. 2 | plavání nahoru a dolu, všechny motory vykonávají stejný pohyb, který v čase mění rychlost |
| 5 | č. 8 | zvedání nahoru a dolu, všechny motory vykonávají stejný pohyb |
| 6 | č. 2 | plavání vpřed a vzad, dvojice motorů vykonávají stejný pohyb, mezi sebou opačný, který v čase mění rychlost |

*Tabulka 5 Seznam dostupných pohybů*

## Režim promítání filmu – princip




*Obrázek 57 Vývojový diagram režimu přehrávání filmů*

Princip běhu programu v režimu promítání filmu a výměny informací mezi Modbus TCP serverem a Node-RED jsem přiblížil ve vývojovém diagramu výše.

Při vstupu do režimu promítání jsou inicializovány 2 globální proměnné v systému Node-RED. První z nich odpovídá svojí hodnotou stavu VLC přehrávače: pokud je její hodnota lichá, systém právě přehrává stopu filmu, pokud sudá, film je pozastaven. Její hodnota se vždy mění o hodnotu 1 při zaznamenání stisku klávesnice „mezerník", sloužící k pozastavení/přehrávání filmu.

Ve druhé proměnné je ukládána aktuální časová hodnota filmové stopy v milisekundách. Ta se aktualizuje při každém pozastavení nebo je k ní přičteno/odečteno 10000ms v závislosti na pohybu uživatele filmovou stopou pomocí šipkových kláves doprava/doleva. Po každé aktualizaci této hodnoty se hodnoty zpoždění jednotlivých změn pohybů musí přepočítat odečtením těchto dvou hodnot. Zprávy se

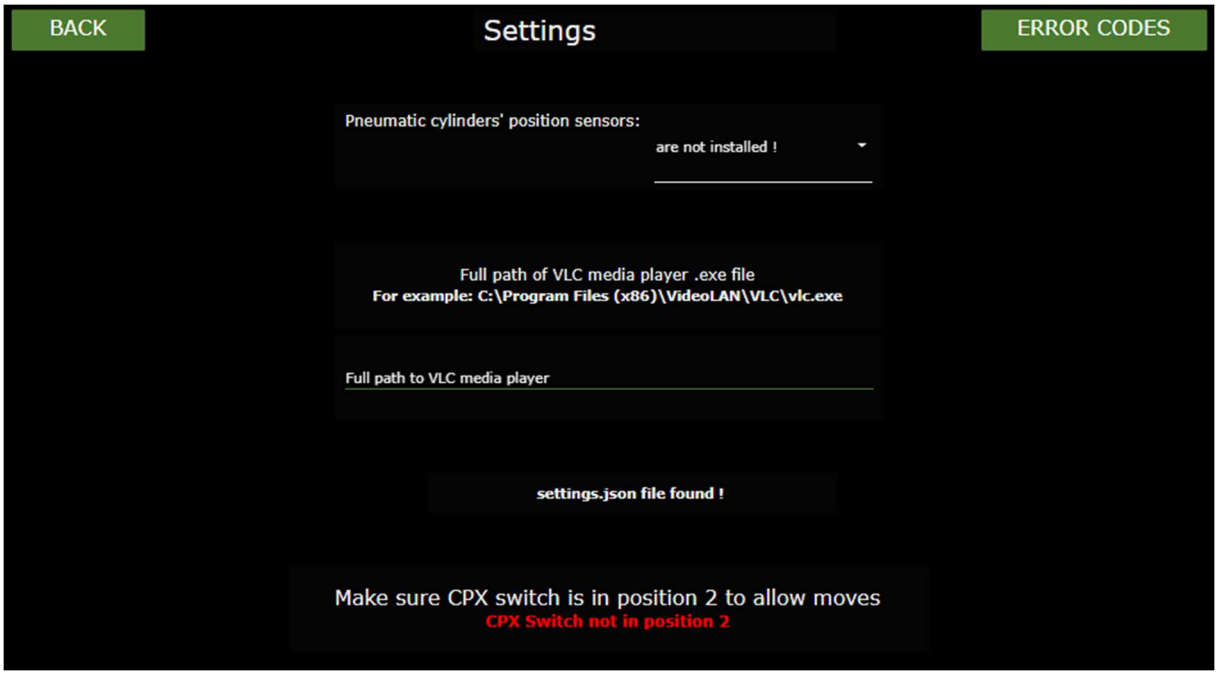
zápornou hodnotou zpožďující proměnné jsou poté zahozeny, jelikož film se již nachází dále, než má nastat daný pohyb. Při posouvání se zpět je nutné znovu načíst zprávy z pohybového souboru, jelikož některé zprávy již byly v minulosti zahozeny. Pokud uživatel zvolí přehrávání filmu až od nějakého času, tento čas je rovnou započten do této proměnné a zpožďovací hodnoty zpráv jsou odečteny ještě před samotným spuštěním filmu.

Bohužel se mi nepodařilo najít jiný způsob, jak kontrolovat čas stopy přímo z VLC přehrávače bez instalovaných dodatkových funkcí. S VLC přehrávačem je možné komunikovat skrze síťový socket protokolu TCP, ale tato možnost neumožňuje získat jakoukoliv zpětnou vazbu o stavu přehrávače, slouží pouze pro ovládání.

Při startu VLC je film pozastaven a uživatel je nucen zmáčknout klávesu pro přehrávání, tím je zaručena synchronizace počátečního momentu spuštění filmu a načítání času v programu. Dále je VLC přehrávač spuštěn v minimálním režimu s neviditelným kurzorem myši pro znemožnění změny času stopy jinak než ovládacími klávesami.

# 14. Nastavení

Pro doplnění několika informací a zároveň k urychlení chodu aplikace slouží okno nastavení. Zde je uživatel nucen zadat informaci, zda jeho systém obsahu koncové snímače poloh pneumatických motorů a také úplnou systémovou cestu k přehrávači VLC. Tyto dvě informace jsou nutné pro běh systému, a pokud nejsou zadány, aplikace sama uživatele v některém z kroků upozorní na jejich doplnění. Parametry jsou uloženy v souboru *settings.json*, není tedy třeba je při každém spuštění aplikace zadávat znovu.




*Obrázek 58 Okno nastavení*

# 15. Informace o bezpečnosti a vývoji

## Bezpečnost

Jedním z hlavních cílů celého projektu je jeho užívání v domácím prostředí. Tato skutečnost však s sebou nese určitá rizika, spojená s předpokládanou neodborností uživatelů či jejich nízkým věkem. Sedadlo samozřejmě lze do jisté míry zabezpečit proti vniknutí cizích předmětů nebo neoprávněné manipulace osob různými přepážkami, převážně v oblasti umístění pneumatických motorů. Avšak je nutné brát v potaz, že bude umístěno například v obývacím pokoji a tím pádem musí také plnit jakousi estetickou funkci. Ačkoliv nikdy nelze vyloučit riziko zranění spojené s důvody uvedenými výše, při vytváření celého systému jsem se snažil co nejvíce snížit tyto šance na minimum. Učiněné kroky jsem popsal zde:

- jakýkoliv pohyb sedačky (kromě zablokování motorů v základním režimu) je povolen pouze s tlačítkem umístěným na CPX modulu v poloze 2
- při údržbě lze vždy ihned zrušit pohyby vyskočením z daného režimu do základního a tak zablokovat sedačku v základní poloze
- při chybě načtení VLC přehrávače či filmu systém nepřejde do režimu přehrávání filmů, nenačte pohyby
- pokud je při přehrávání filmů zmáčknuta jiná klávesa než jedna z ovládacích, veškeré pohyby z pohybového souboru jsou zastaveny a sedačka se vrátí do základní polohy

- pokud je při přehrávání filmů časová stopa přehrávače pozastavena či změněna, sedačka se sama umístí do základní polohy a při opětovném spuštění vždy setrvá v této poloze nejméně po dobu dalších 5 sekund
- při přehrávání filmů lze všechny pohyby také zastavit červeným tlačítkem „SAFETY STOP“ umístěným v okně spouštění přehrávání

### Vývoj pohybového souboru

Jak jsem již v předchozích kapitolách o režimu promítání filmů zmínil, synchronizace pohybů sedačky a filmové stopy je zajištěna pomocí pohybového souboru. Textový soubor je zformátován jako JSON (JavaScript Object Notation), který zajišťuje v největší míře výměnu dat, je velmi lehce uživatelsky čitelný a zároveň i programovatelný. [56] Z těchto důvodů je také nejvíce vhodný k tomuto účelu, kdy si bude moci každý uživatel sám tento soubor vytvořit a sdílet jej s ostatními, či jej jednoduše poupravit při výskytu chyb. K jeho vytvoření však platí několik striktních pravidel pro bezproblémové dekódování systémem.

```
{
    "Priklad": [
        {
            "topic": "mod",
            "payload": "1",
            "time": "00:10:500",
            "set1": "500",
            "set2": "500",
            "duration": "700"
        },
        {
            "topic": "mod",
            "payload": "2",
            "time": "00:30:010",
            "set1": "500",
            "set2": "500",
            "duration": "700"
        },
        {
            "topic": "mod",
            "payload": "0",
            "time": "00:45:500",
            "set1": "3000",
            "set2": "3000",
            "duration": "1000"
        },
        {
            "topic": "mod",
            "payload": "3",
            "time": "01:10:000",
            "set1": "500",
            "set2": "500",
            "duration": "600"
        },
        {
            "topic": "mod",
            "payload": "6",
            "time": "01:30:100",
            "set1": "600",
            "set2": "600",
            "duration": "1000"
        }
    ]
}
```


*Obrázek 59 Příklad funkčního pohybového souboru*

Z příkladu pohybového souboru výše je zřejmé, že každý pohyb, respektive změna pohybu, tvoří jeden objekt z celého pole objektů s několika, vždy stejnými, atributy. Název hlavního objektu JSON se všemi pohyby je libovolný. Při programování je důležité brát v potaz, že každý pohyb je brán jako nekonečná zpětná smyčka, a tak při každé změně pohybu je nutné zadat příslušný objekt do pole s volitelnými hodnotami. V hlavní složce aplikace jsou dostupné dva krátké pohybové soubory jako příklady. Atributy je možné psát v jakémkoliv pořadí uvnitř objektu jedné změny pohybu. Zde uvádím tabulku atributů, jejich význam a dovolené hodnoty. Tato tabulka se také nachází v aplikaci, v sekci „For developers“:

| Tabulka atributů pohybů | | | |
|---|---|---|---|
| Název | Formát | Meze hodnot | Význam |
| topic | string | mod | pokud má jinou hodnotu, systém tento pohyb vynechá |
| payload | int | 0-6 | číslo vykonávaného pohybu |
| time | min:sek:mili | 00:00:000 - … | čas pro vykonání pohybu ve formátu minuty:sekundy:milisekundy |
| set1 | int | 0 – 10000 | hodnota MotionApp – *iSetPointValue1* |
| set2 | int | 0 – 10000 | hodnota MotionApp – *iSetPointValue2* |
| duration | int | 0 - … | doba jednoho pohybu motoru, milisekundy |

*Tabulka 6 Atributy pohybů v pohybovém souboru*

- set1 – pohyby 0, 1, 2, 3 a 5 – úroveň škrcení /100[%]
- set1 – pohyby 4 a 6 – pozice proporcionálního ventilu /100[%]
- set2 – pohyby 0, 1, 2, 3 – úroveň škrcení /100[%]
- set2 – pohyb 4 – nemá význam
- set2 – pohyb 5 – tlak vzduchu na ventilových portech /1000[bar]
- set2 - pohyb 6 – pozice proporcionálního ventilu, pohyb zpět /100[%]
- duration – tato hodnota je používána pouze pokud nejsou nainstalovány koncové snímače poloh motorů, výjimkou je pohyb č. 6

# 16. Návod ke zprovoznění systému

## Instalace aplikace a její spuštění

V tomto bodu práce se zabývám postupem instalace nezbytného programového vybavení k zabezpečení správného chodu celého systému od úplného začátku.

Před spuštěním aplikace Node-RED na počítači (nejlépe s operačním systémem Windows 8 či 10) je nejdříve nutné nainstalovat následující software:

- Node.js
  - volně dostupný z: https://nodejs.org/en/
- Node-RED
  - po instalaci Node.js lze nainstalovat přímo z příkazového řádku příkazem:
  - *npm install -g --unsafe-perm node-red*
- VLC Media Player
  - volně dostupný z: https://www.videolan.org/vlc/

Po splnění prvního kroku je dále třeba nainstalovat Node.js modul *iohook* ke správné funkci vytvořeného uzlu odposlouchávajícího stisky klávesnice a myši. Tento modul lze nainstalovat přímo z příkazového řádku pomocí správce balíčků pro JavaScript npm příkazem (nutné je spustit příkazový řádek Windows jako správce):

- *npm install iohook –save*

Pro jednodušší sdílení a instalaci je aplikace vytvořena jako Node-RED projekt. Tuto funkci je však nutné nejdříve povolit po první instalaci editoru. Před povolením této funkce je nejdříve nutné mít založený Git účet a ve svém počítači mít dostupné příkazy *git* a *ssh-keygen.* Toho je nejjednodušší docílit pomocí instalace některého z klientů Git na Windows, například:

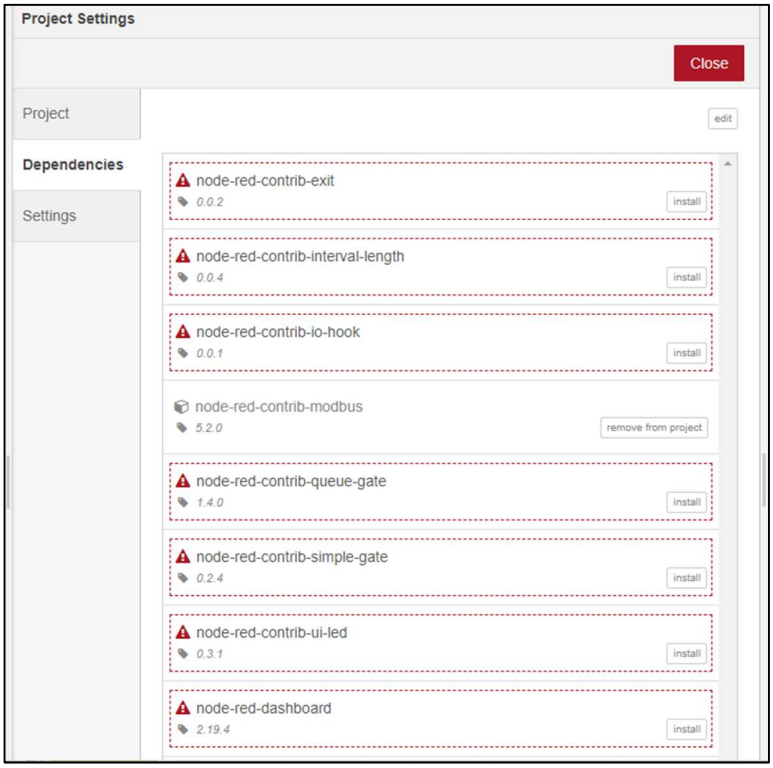
- Git for Windows
  - volně dostupný z: https://gitforwindows.org/

Po jeho instalaci důležité se držet návodu [57] k povolení Node-RED projektů. Hlavním cílem je změnit v souboru *setting.js* umístěného v hlavní složce /.node-red kolonku projects.enabled na true. Poté již stačí spustit Node-RED z příkazové řádky, otevřít editor v prohlížeči na portu 1880 a postupovat dle instrukcí. Po vytvoření vlastního prvního projektu je počítač připraven k „nainstalování" a spuštění projektu MovTheater. Složka Node-RED je při dodržení postupu výše umístěna v:

- C:\Users\<Uživatel>\.node-red (při zaměnění <Uživatel> za aktuálního uživatele)

Zkomprimovaný soubor *MovTheaterNR.rar* stačí rozbalit a složku překopírovat do složky projektů /.node-red/projects. V něm se nachází celý program Node-RED. Dále je nutné rozbalit i druhou složku *MovTheaterSYS.rar*, a to přímo do jednotky C:/. Ta obsahuje spouštěcí soubor *MovTheater.bat* se zástupcem, textový soubor nastavení *settings.json*, složku s krátkými příklady pohybových souborů a CODESYS projekt pro nahrání do Festo Motion Terminal, respektive CPX terminálu (*MovTheater.project*).

Při prvním spuštěním projektu přímo pomocí souboru *MovTheater.bat* nebo skrze jeho zástupce ze složky C:\MovTheater je nutné otevřít editor Node-RED v tomto počítači na adrese http://localhost:1880/ (v jiném zařízení nutno použít místo *localhost* aktuální IP adresu počítače, na kterém běží Node-RED) a v červeném rámečku kliknout na tlačítko „Manage project dependencies". Po kliknutí se zobrazí okno (viz níže), kde je nutné nainstalovat všechny chybějící moduly uzlů zvýrazněných červeně. Poté je nutné restartovat celý Node-RED vypnutím jeho terminálového okna.

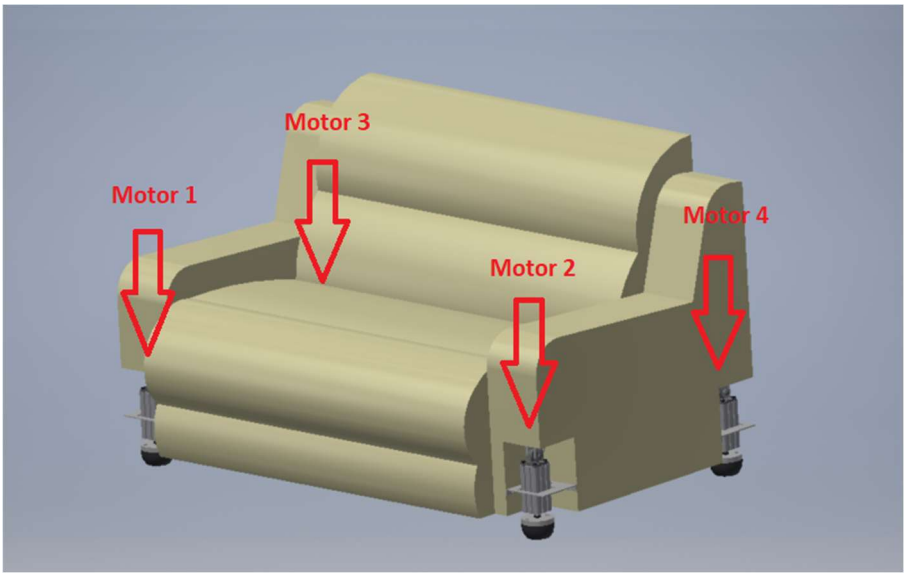



*Obrázek 60 Okno instalace chybějících modulů uzlů*

Po splnění všech předchozích kroků již lze spouštět aplikaci skrze soubor *MovTheater.bat* nebo jeho zástupce bez nutnosti ručního spouštění prohlížeče. Po spuštění terminálového okna Node-RED by se měl sám spustit nativní internetový prohlížeč Windows – Microsoft Edge, na hlavní nabídce aplikace. Pokud tomu tak není, hlavní stránku je možné najít v jakémkoliv prohlížeči pod adresou http://localhost:1880/ui (při přístupu z jiného zařízení na http://<IPadresa>:1880/ui). Ihned poté je doporučeno zamířit do okna nastavení a zadat cestu k nainstalovanému přehrávači VLC, popřípadě i typ FMT systému s pneumatickými motory s/bez senzorů pro bezproblémový další provoz.

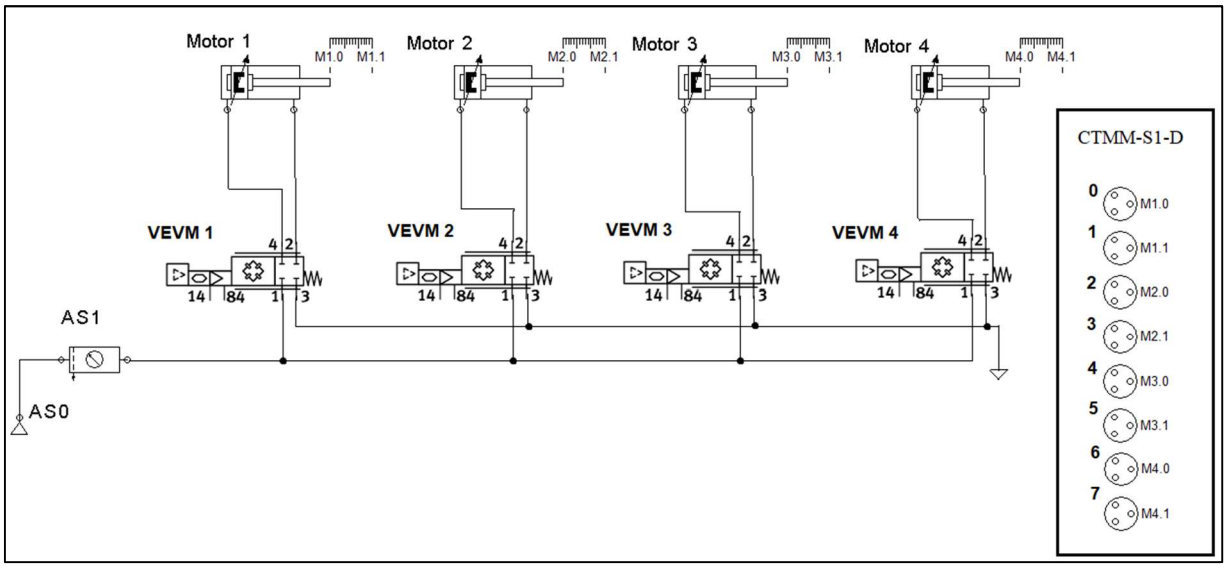
## Zprovoznění FMT

V této části se zabývám zprovozněním Festo Motion Terminal. Nejdříve popisuji zapojení pneumatických motorů a dále i způsob nahrání celého projektu do CPX terminálu.




*Obrázek 61 Očíslování motorů*

Očíslování motorů odpovídá obrázku výše. Níže lze vidět zjednodušené schéma zapojení jak pneumatické části, tak té elektrické. Schéma jsem vytvořil v programu FluidSim 4.2. Ventily VEVM jsou číslované od 1 do 4 směrem od modulu CPX a motory k nim jsou připojeny ve stejném pořadí. Motor je vždy připojen tak, že v základní poloze je zajetý s předpokladem tlaku přivedeného stlačeného vzduchu na portu 2. Pokud jsou instalovány koncové snímače poloh, dle VTEM dokumentace ventilu 1 odpovídají pozice digitálních vstupů 0 a 1 s tím, že senzor základní polohy je zapojen do pozice 0. Ventilu a motoru číslo 2 poté odpovídají pozice 2 a 3 a tak dále.




*Obrázek 62 Zjednodušené schéma zapojení*

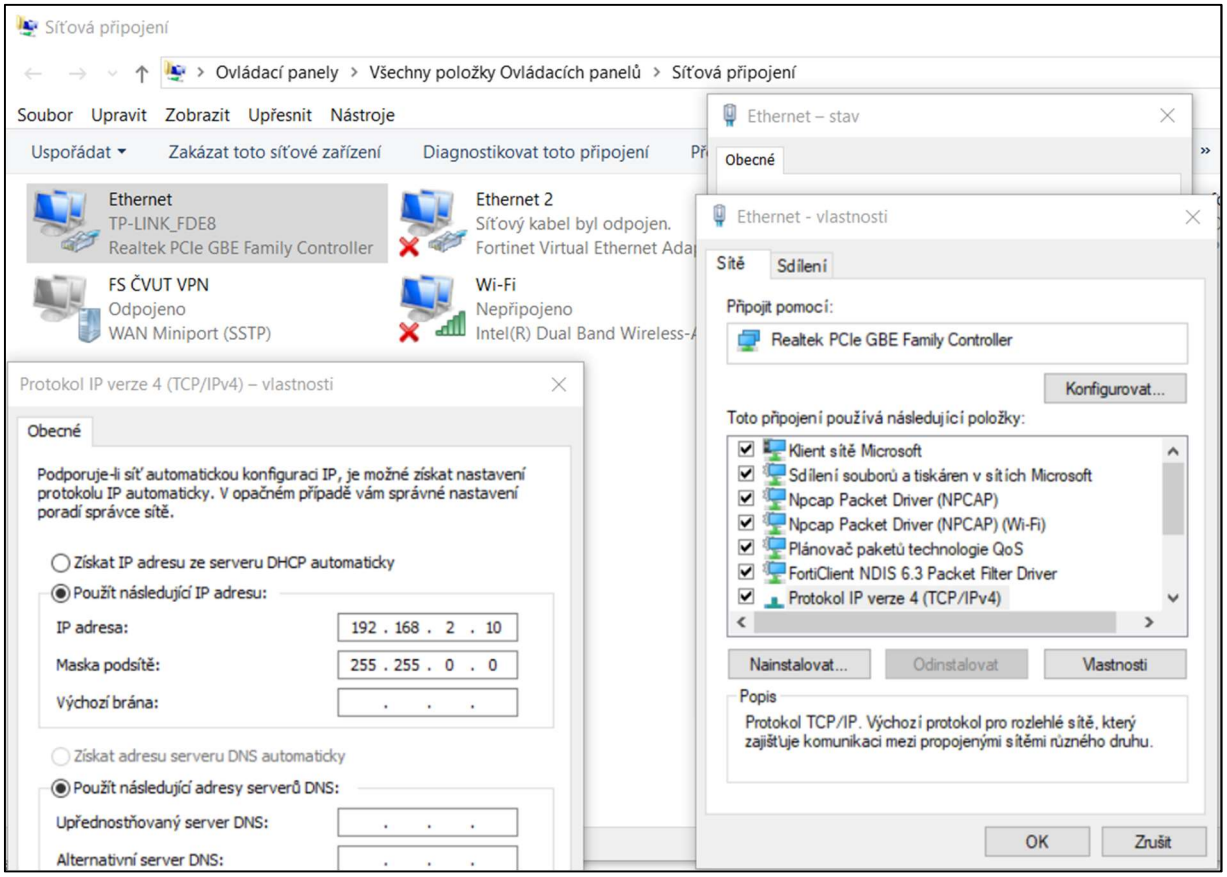
Pro nahrání programu PLC do automatu CPX je nutné mít nainstalované PLC vývojové prostředí a k němu doplňky:

- CODESYS

    - poskytované firmou Festo
    - volně dostupné z: https://www.festo.com/net/cs_cz/SupportPortal/
- Balíček Target Support Package CODESYS verze 3.5.7.356
    - dostupné: https://www.festo.com/net/cs_cz/SupportPortal/InternetSearch.aspx?q=codesys&tab=16&type=73#result
    - instalace probíhá přes CODESYS>Tools>Package Manager…>Install…
- Knihovnu Function blocks CODESYS verze 3.5.7.221
    - dostupné: https://www.festo.com/net/cs_cz/SupportPortal/InternetSearch.aspx?q=codesys&tab=16&type=74#result
    - instalace probíhá přes CODESYS>Tools>Library Repository…>Install…

Po instalaci programu a všech potřebných balíčků a knihoven je potřeba připravit počítač a FMT ke zprovoznění komunikace. Po propojení počítače a CPX terminálu kabelem Ethernet je nutné změnit IP adresu adaptéru Ethernet v systému Windows na dosažitelnou. IP adresa CPX je z výroby nastavena na 192.168.2.20. Kroky ve Windows jsou následující: Otevřít nastavení Síť a internet > Změnit možnosti adaptéru > Ethernet (adaptér Ethernet) > Vlastnosti > Protokol verze IPv4 (TCP/IPv4) > Použít následující IP adresu: > IP adresa: 192.168.2.10 (například, poslední číslo se může lišit).

Toto nastavení je nutné ponechat pro každé použití projektu, jelikož počítač a CPX terminál spolu komunikují skrze protokol Modbus TCP. Aplikace na tuto skutečnost sama upozorní při přípravě k přehrávání v okně zkoušky komunikace.



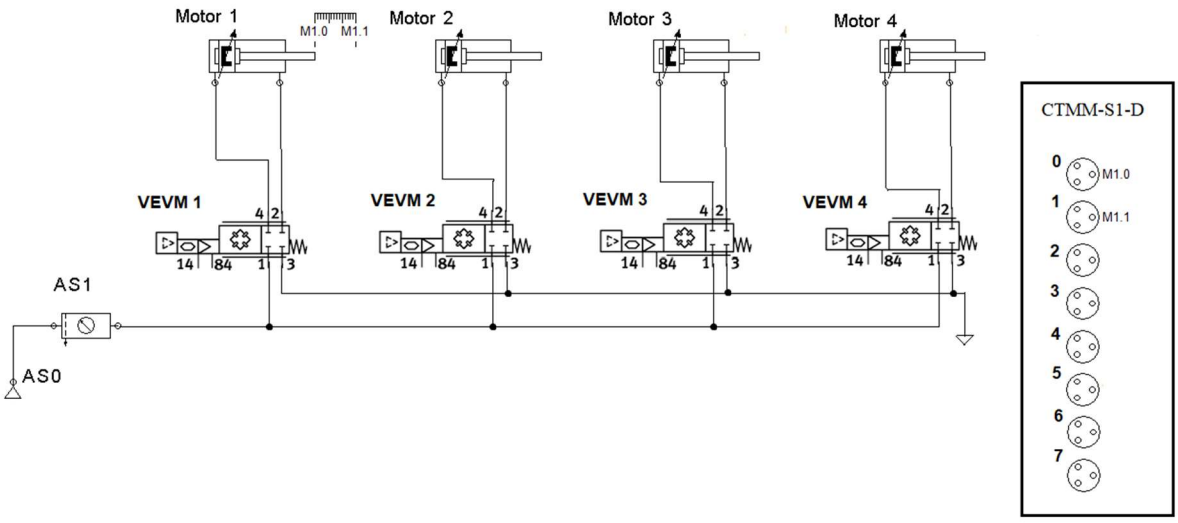

*Obrázek 63 Postup nastavení IP adresy adaptéru Ethernet*

Soubor projektu CODESYS je umístěn v komprimovaném souboru *MovTheaterSYS.rar*. Po jeho extrahování do jednotky C:/ (uvedené i v předchozí kapitole) se nachází ve složce C:/MovTheater počítače. Tento soubor je nutné otevřít pomocí prostředí CODESYS a nahrát ho do CPX terminálu.

Po jeho otevření je ještě nutné nastavit bránu pro komunikaci Gateway dvojitým kliknutím na Device > Add gateway … > OK > Scan network. Po těchto krocích by měl CODESYS najít připojený CPX terminál, který dvakrát rozklikneme a tím nastavíme aktivní trasu pro nahrávání programu. Na konec už jen stačí nahrát program do modulu stisknutím šedého kolečka se zelenou tečkou v horní liště a vybrat možnost „Login with download“. Tím se projekt nahraje do CPX terminálu a celý Festo Motion Terminal již čeká na povely z počítačové aplikace.

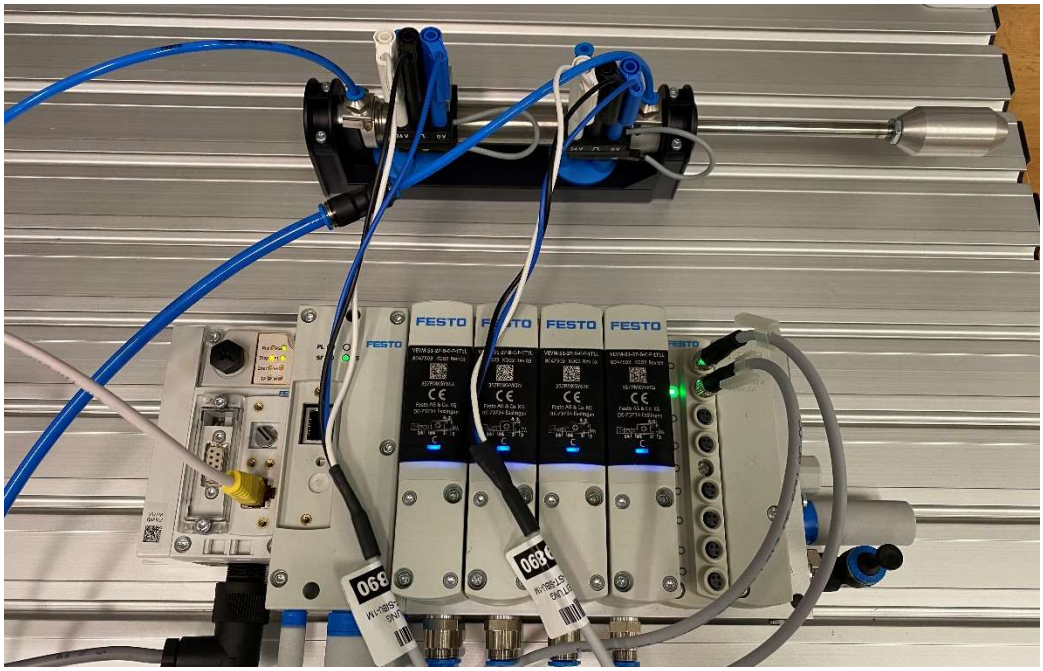
# 17. Testování

Testování funkcí celého systému jsem provedl ve školní laboratoři. Bohužel jsem neměl k dispozici reálný model sedadla, a tak jsem sestrojil zkušební systém pro co nejvěrnější simulaci provozu původního modelu pomocí dostupných sad Festo a Festo Didactic. K Festo Motion Terminal jsem připojil 4 pneumatické motory se stejným zdvihem, se kterým se počítá i v koncepčním modelu, pouze jiného typu a s menším průměrem pístu, konkrétně model DSNU-20-100-PPV-A. Tyto motory jsem zapojil dle schématu níže. Některé funkce systému jsem testoval pouze s jedním motorem (číslo 1) s instalovanými koncovými senzory.




*Obrázek 64 Schéma zapojení zkušebního systému*

Nejdříve jsem provedl instalaci celého projektu na záložní notebook s pomocí návodu uvedeného v předchozí kapitole. Po tomto kroku jsem postupně ozkoušel všechny funkce celého systému od manuálního řízení až po režim přehrávání filmů s pohyby. Výsledky a připomínky jsem zaznamenal v této kapitole.




*Obrázek 65 Ukázka zkoušky funkcí na jednom motoru s instalovanými koncovými senzory*

## Základní režim

V základním režimu, čili v situaci pohybu uživatele v nabídce aplikace, nejsou motory z bezpečnostních důvodů pod tlakem vzduchu (oba výstupní porty ventilů VEVM jsou ve stavu „vypuštěný“). Pokud jsou motory před vstupem do tohoto režimu v jakékoliv pozici, při přechodu do tohoto stavu jsou automaticky odpojeny od tlaku vzduchu a lze s nimi volně pohybovat (například při přechodu z manuálního režimu do základního). Při zatížení sedačky se předpokládá jejich samovolné zajetí.

## Manuální režim

Po stisknutí přepínače pro povolení manuálního řízení jsou pneumatické motory stále bez tlaku vzduchu až do jejich prvního, uživatelem vynuceného pohybu. Zde byl systém testován pro všechny 4 zapojené motory. Manuální řízení pomocí dispečerského okna fungovala dle předpokladů, společně s ukazatelem pozice prvního motoru s koncovými senzory. Pohyby motorů jsou umožněny pouze pod podmínkou, je-li přepínač CPX v pozici 2.

## Diagnostika úniků

Po přepnutí do režimu diagnostiky úniků a spuštění celého procesu tlačítkem jsou provedeny měření, které jsou interně řízeny danou MotionApp. Festo Motion Terminal provede několik pohybů motorů a po chvíli se objeví výsledky diagnostiky ve vizualizačním okně. Zde je možné také sledovat aktuální stav procesu. Pohyby motorů jsou opět podmíněny pozicí 2 CPX přepínače. Pro správný chod této funkce je třeba mít

uložená data o připojených komponentách přímo v paměti FMT, tak jak bylo zmíněno v předešlých částech práce. Díky mému předchozímu zacházení se systémem však tyto informace již byly uloženy dříve a celý proces proběhl bez problému.

## Příprava na přehrávání filmů

Jak již bylo zmíněno v popisu programování systému, před vstupem k přehrávání filmů je pokaždé nutné projít celou procedurou přípravy. V prvním okně je nejdříve ozkoušena komunikace mezi PC a FMT. Další částí je zkouška pohybů, kdy na uživatelův povel jsou nejprve všechny motory vysunuty a po kontrole zase zasunuty. Pokud je v nastavení uvedeno, že jsou na motorech umístěny koncové senzory, systém tuto kontrolu pohybů provádí sám. V opačném případě tuto kontrolu musí provézt sám uživatel. Pohyby jsou opět podmíněny pozicí 2 CPX přepínače. V dalších částech je nutné zadat veškeré informace k nadcházejícímu přehrávání. Systém kontroluje zadané cesty k potřebným souborům a při jejich špatném zadání uživatele vrátí zpět s upozorněním pravděpodobné chyby, či ho vůbec nepustí dále. Po projití celého procesu příprav, které mají za úkol primárně ujistit se o bezpečnosti sedačky a také o její funkčnosti, se aplikace nachází na posledním okně před samotným spuštěním filmu tlačítkem. Při zkoušce jsem celý tento proces provedl několikrát s různými scénáři (špatně instalovaný koncový senzor, špatně zadaný pohybový soubor, špatně zadaný film, chybně zadaná cesta k VLC v nastavení, atd.). Systém se celou dobu choval dle předpokladů, dokonce dokázal odhadnout mnou schválně udělanou chybu a nasměrovat mě k její nápravě.

## Přehrávání filmů

V této kapitole byl systém podroben testům se všemi motory bez koncových senzorů. Tuto část zkoušek si z důvodu velké obsáhlosti dovolím rozdělit na dvě části:

- zkouška samotné funkce přehrávání filmů
- zkouška bezpečnostních prvků

### *Zkouška samotné funkce přehrávání filmů*

Pokud jsme úspěšně prošli přípravou systému a správně zadali všechny nezbytné parametry pro spuštění přehrávání, můžeme stisknout tlačítko pro přehrávání. Po něm se zobrazí černá obrazovka VLC přehrávače s filmem, který je pozastaven v zadaném čase. Pro spuštění přehrávání je nutné zmáčknout klávesu „mezerník".

Testování jsem provedl opakovaně. Při prvním pokusu jsem nijak neměnil časovou stopu filmu pomocí kláves a sledoval jsem hlavně časovou správnost výskytů pohybů. Pro účely testování jsem použil prodloužený příklad pohybového souboru se všemi pohyby

od sebe vzdálenými zhruba 20 sekund s délkou trvání 10 a 20 sekundy v prvních 3 minutách. Dále jsem již snížil frekvenci pohybů a dalších zhruba 15 minut sledoval časy. Po dokončení této první části jsem došel k závěru, že synchronizace časové stopy VLC přehrávače a Node-RED je velice dobrá. I po 15 minutách přehrávání nelze lidskými smysly rozeznat jakékoliv časové vychýlení naprogramovaných pohybů oproti stopě filmu. Z tohoto usuzuji, že při nepohybování časovou stopou lze systém použít na jakkoliv dlouhé filmy.

V druhém pokusu jsem se soustředil na časový posun mezi VLC a Node-RED při neustálém přetáčení filmu po 10-ti sekundových intervalech pomocí šipek. Zde už je však situace více komplikovaná. Během posouvání stopy při pozastaveném filmu je systém ve skvělé časové synchronizaci a rozdíly nelze rozeznat. Avšak při posouvání stopy za chodu filmu se synchronizace po několika desítkách přetočení vytrácí. Tento jev bych přisuzoval špatné odezvě a rychlosti načítání filmové stopy u VLC přehrávače. Při používání systému bych tedy doporučil přetáčet film pouze při pozastaveném přehrávání.

V poslední zkoušce jsem se zaměřil přímo na pohyby pneumatických motorů při jednotlivých sekvencích. Na první pohled fungují všechny naprogramované pohyby tak, jak mají. Sekvence motorů i délky trvání jednoho manévru motorů jsou v pořádku. Při důkladnějším pozorování si však lze občas všimnout (přibližně u 5 % změn pohybů) jiného nastavení parametrů MotionApp vždy na začátku daného pohybu. Tuto chybu bych přisuzoval pomalému přenosu dat mezi Node-RED a CPX, jelikož aplikace vysílá povely pro změnu pohybu vždy v přesný čas, kdy se má pohyb vyskytnout. Co se týče samotných naprogramovaných pohybů, je otázkou, zda by všechny z nich našly použití v praxi.

Závěrem však usuzuji, že funkce přehrávání filmu a pohyb sedačky fungují velice dobře. Při nepřetáčení filmu za chodu přehrávání je synchronizace časové stopy VLC přehrávače a Node-RED aplikace na velmi dobré úrovni. Všechny pohyby fungují dle zadaných parametrů v pohybovém souboru s občasnou chybou při okamžité změně z jednoho pohybu do druhého.

### *Zkouška bezpečnostních prvků*

V této části testování jsem prověřil bezpečnostní funkce systému v režimu přehrávání filmů. Jak již bylo uvedeno dříve, z důvodu použití v domácím prostředí jsem se důkladně zaměřil na tyto prvky a považuji je za stejně důležité jako hlavní funkci celého projektu, kterou je zábava.

Prvním bezpečnostním prvkem je 5 sekundová prodleva bez pohybu od znovuspuštění/přetočení filmové stopy. Tuto funkci jsem několikrát prověřil při přetáčení filmu v předchozích testech a vždy fungovala.

Další funkcí je úplné zastavení pohybů při stisknutí jiné klávesy než některé z ovládacích. V tomto případě se motory zablokují v aktuální pozici a celý proces přehrávání filmu je nutné zopakovat od začátku, i když filmová stopa běží dál.

Dalším bezpečnostním prvkem je tlačítko „Safety stop" umístěné v okně aplikace při spouštění filmu. Po stisknutí tohoto tlačítka přejde systém do základního režimu, kde jsou motory bez tlaku vzduchu. Toto tlačítko by v ideálním (a pravděpodobně také nutném) případě muselo být umístěno jako hardware tlačítko přímo na boku sedačky.

Posledním, ale již známým bezpečnostním prvkem, je podmínka pohybu skrz CPX přepínač, který musí být nastaven do polohy 2. V opačném případě sedačka setrvává v základním režimu.

## Shrnutí

Na úplný závěr kapitoly o testování celého systému bych rád shrnul dosažené výsledky. Bohužel z důvodu probíhající pandemie při tvorbě práce a následném testování jsem neměl takové množství času, ve kterém bych byl zcela schopen skutečně otestovat všechny funkce projektu a případně najít ojedinělejší chyby a opravit je.

Jediný viditelný problém, který se vyskytl při testech, je již zmíněná problematika přenosu dat v daný časový okamžik změny pohybu. Tento bod by se však dal napravit například předesíláním parametrů následujícího pohybu před samotnou změnou pohybu.

I přes tuto skutečnost si však troufám tvrdit, že se systém chová tak jak je očekáváno. Všechny režimy vykonávají svojí funkci na velmi dobré úrovni a systém by jistě mohl plnit svojí hlavní funkci, prohloubení filmového zážitku, již v tomto stavu.

# 18. Zhodnocení projektu

## Aktuální nedostatky

- Složitá instalace
    - náročná instalace jak nutného software pro spuštění počítačové aplikace, tak také i nahrání programu PLC do CPX modulu, zvláště pro uživatele s menšími dovednostmi s PC
- Složitá diagnostika chyb a jejich řešení
    - v aktuálním provedení aplikace je diagnostika chyb možná pouze skrz okno chybových kódů ventilů VEVM, dále si však již musí uživatel poradit sám
- Slabé zabezpečení
    - ač byla v průběhu celé tvorby projektu brána velká zřetel na implementaci zabezpečovacích prvků, v této fázi návrhu jistě nejsou splněny všechny podmínky pro bezpečné provozování zařízení v domácím prostředí

## Řešení nedostatků

Instalace počítačové aplikace by se dala vyřešit nebo alespoň zjednodušit vytvořením instalačního souboru. Ten by nainstaloval do počítače všechen potřebný software, včetně modulů uzlů potřebných pro spuštění aplikace v Node-RED. Zprovoznění Festo Motion Terminal by však stále zůstalo na uživateli.

Pro zjednodušení celého problému, týkajícího se vyskytujících se chyb, by bylo vhodné vytvořit seznam či vývojové diagramy příznaků a chyb, jejichž výstupem by bylo jejich další řešení. Tohoto kroku však lze dosáhnout až po určité zkušební době používání prototypu výrobku, vyhledání a opravení častých chyb.

Zabezpečení celého projektu se jistě dá posílit ve dvou rovinách. Tou první je programová část, ve které již systém některé zabezpečovací prvky obsahuje, zejména při možném pohybu sedačky. Dále by mohlo přibýt zabezpečení přihlašovacími údaji při spuštění aplikace. Druhou rovinnou je zabezpečení samotné sedačky, která se nachází v návrhové fázi. Zde by bylo nutné zabezpečit hlavně spodní stranu před vniknutím jakýchkoliv předmětů či neoprávněné manipulace. Zároveň by však musel být brán zřetel na možnost jednoduchých oprav či údržby ze strany uživatele.

## Možnosti budoucího vývoje

Další práce, týkající se tohoto projektu, by se měly týkat především zlepšení dostupnosti a zjednodušení používání produktu. Konkrétně by se tyto činnosti mohly zaměřovat na tvorbu instalačního souboru či vytvoření editoru pohybových souborů, což

by velice usnadnilo práci s tímto projektem nejen uživatelům, ale i vývojářům. Čím více by se však dařilo zlepšovat tuto stránku projektu, tím více by musel být kladen důraz na zlepšení již zmíněné bezpečnosti.

K zamyšlení by určitě stála i otázka rozšíření projektu v oblasti řídicí, a to v nahrazení PC či notebooku jednodeskovým počítačem Raspberry Pi 4. generace. Ten disponuje 4 jádrovým 1,5 GHz procesorem ARM, operační pamětí až 4 GB a veškerou konektivitou potřebnou k chodu i řízení celého projektu. Oproti předchozím generacím navíc nabízí rozlišení obrazu 4K, což je velmi vhodné k použití v systému domácího kina. Přenos celého projektu z operačního systému Windows na Raspberry Pi OS by neměl způsobovat velké problémy, jelikož všechen použitý software je dostupný i na tento operační systém, a to s minimálními nutnými zásahy do již naprogramovaných funkcí. Dokonce by tato změna řídicího členu mohla přinést mnoho výhod, jako například větší otevřenost operačního systému či distribuci sedačky s implementovaným řídícím členem uvnitř. Na celkové ceně projektu by přidání Raspberry Pi nemělo velký vliv, jeho cena za kus se pohybuje okolo 60 EUR. [58]

Další otázkou je rozšíření systému v oblasti pohonu. Při prvotním rozhodování, zda použít stlačený vzduch či elektřinu jako zdroj energie pohonů byla vybrána první možnost. Stlačený vzduch, i přes všechny jeho výhody, s sebou nese jeden značný problém, a to jeho výrobu v domácím prostředí. Použití elektrických motorů, například krokových, by znamenalo zažehnání problému s výrobou a uchováváním stlačeného vzduchu. Práce by se zabývaly výběrem vhodného typu motorů, jejich ovladači a řízením. Dále by bylo samozřejmě nutné učinit změny v počítačové aplikaci, hlavně v posílaných parametrech pohybů. Je možné, že by se tímto krokem stal projekt i cenově dostupnějším.

*Obrázek 66 Ikona projektu MovTheater [59]*

# Závěr

V první kapitole teoretické části jsem zpracoval rešerši, zabývající se historickým vývojem pneumatických ventilových terminálů od samostatně stojících ventilů přes ventilové bloky a terminály až po Festo Motion Terminal, který zastupuje novou éru tzv. digitální pneumatiky. Ke všem vývojovým fázím jsem také přidal možnosti aplikačních příkladů vyskytujících se v dnešním průmyslu. Tím jsem splnil pokyn pro vypracování diplomové práce:

1. Prostudujte historii a aplikace pneumatických ventilových terminálů od počátku vývoje až po ventilový terminál pro digitální pneumatiku FMT (Festo Motion Terminal)

V následující části jsem vysvětlil základní funkce Festo Motion Terminal a principy jeho programování. Tato kapitola může sloužit jako seznámení se s možnostmi FMT pro další generace studentů, kteří již budou mít teoretické znalosti o tomto produktu a budou toužit o rozšíření svých znalostí při práci s tímto zařízením. K vytvoření tohoto návodu mi pomohla již zrealizovaná a funkční aplikace ve školní laboratoři. Tím jsem také splnil další pokyn:

2. Prostudujte podrobně funkci FMT terminálu pomocí aplikačních příkladů zrealizovaných v laboratoři

V praktické části jsem již postupně popsal postup mých prací ke splnění hlavního cíle mé diplomové práce: vytvoření návrhu systému domácího kina s pohyblivou sedačkou pomocí Festo Motion Terminal. V první kapitole jsem navrhl konstrukci sedačky a vybral vhodné komponenty a poté jsem vybral vhodný software pro splnění celého úkolu. V dalších kapitolách jsou již popsány základní funkce celého systému z uživatelského i programátorského pohledu. Po dokončení tvorby projektu jsem vytvořil návod pro bezproblémovou instalaci celého systému a jeho spuštění. Následně jsem všechny funkce dosavadního systému otestoval ve školní laboratoři. Na závěr zhodnocuji dosažené výsledky, aktuální nedostatky a jejich možná řešení. Dále uvádím možnosti navazujících prací k rozšíření celého projektu. Praktickou částí jsem také splnil i hlavní cíl mé diplomové práce:

3. Navrhněte systém kina s pohyblivou sedačkou pro možné domácí použití

Tím byly naplněny všechny body zadání mé diplomové práce.

# Seznam obrázků

# Seznam tabulek

# Seznam citovaných zdrojů

[1] ČERNOHOUZOVÁ, Lucie. *Tvorba výukových materiálů pro předmět AOV* [online]. Liberec, 2019 [cit. 2020-11-07]. Dostupné z: https://dspace.tul.cz/bitstream/handle/15240/4202/bc_16533.pdf?sequence=1&isAllowed=y. Bakalářská práce. Technická univerzita, Fakulta textilní. Vedoucí práce Ing. Lea Farská.

[2] MARTINÁSKOVÁ, Marie. *Rozváděče: konstrukční řešení* [online]. Praha [cit. 2020-11-07]. Prezentace. ČVUT v Praze, Fakulta strojní, Ústav přístrojové a řídicí techniky.

[3] Ventily. *Střední odborné učiliště elektrotechnické* [online]. Plzeň [cit. 2020-11-07]. Dostupné z: https://www.souepl.cz/wp-content/ucitele/moc/ventily/ventily_1.htm

[4] *Festo Czech Republic: výrobky* [online]. [cit. 2020-10-17]. Dostupné z: https://www.festo.com/cat/cs_cz/products

[5] LACINA, Václav. Pneumatika a komunikace. *MM Spektrum: Odborně-vzdělávací a zpravodajský portál z oblasti strojírenství a navazujících oborů* [online]. **2012**(7) [cit. 2020-11-07]. ISSN 1212-2572. Dostupné z: https://www.mmspektrum.com/clanek/pneumatika-a-komunikace.html

[6] *Festo Valve terminals: Philosophy*. 1997.

[7] Hauptkatalog: Hydropneumatikpressen, Druckluftpressen, Handhebelpressen. *Maeder pressen* [online]. 2020 [cit. 2020-11-07]. Dostupné z: https://www.maederpressen.de/Deutsch:Download:Katalog.asp

[8] *Atelier Technik s.r.o.: foto archiv*.

[9] HAUMER, Zdeněk. Pneumatické ventilové a instalační terminály. *Automa: časopis pro automatizační techniku* [online]. Děčín, **2012**(8-9), 3 [cit. 2020-10-14]. ISSN 1210-9592. Dostupné z: http://automa.cz/Aton/FileRepository/pdf_articles/9777.pdf

[10] MARTINÁSKOVÁ, Marie. *Ventilové a instalační terminály* [online]. Praha, 2018 [cit. 2020-11-07]. Prezentace. ČVUT v Praze, Fakulta strojní, Ústav přístrojové a řídicí techniky.

[11] *Festo: ECONOMICS: Installation concepts and their relative economy*. 1997.

[12] Ventilové bloky ComboBox série PS. *Intratech: pneumaticke-prvky.cz* [online]. [cit. 2020-11-07]. Dostupné z: http://www.pneumaticke-prvky.cz/ventilove-bloky-combobox-serie-ps

[13] *Festo: Valve terminal, type 02*. 1997.

[14] FESTO. Samozřejmě, opět komunikace!. *Automa: časopis pro automatizační techniku* [online]. Děčín, **2015**(8-9) [cit. 2020-10-14]. ISSN 1210-9592. Dostupné z: http://automa.cz/Aton/FileRepository/pdf_articles/54025.pdf

[15] Brief overview of valve terminals. *Festo* [online]. [cit. 2020-11-07]. Dostupné z: https://www.festo.com/net/SupportPortal/Files/429716/Flyer_Overview_valve_terminals_135237_en.pdf

[16] Ventilové terminály: Koncepce instalace a řízení. *Festo: výrobky* [online]. [cit. 2020-11-07]. Dostupné z: https://www.festo.com/cms/cs_cz/9749.htm

[17] VOJÁČEK, Antonín. IO-Link – popis digitální komunikace pro senzory. *Automatizace.hw.cz* [online]. [cit. 2020-11-13]. Dostupné z: https://automatizace.hw.cz/iolink-popis-digitalni-komunikace-pro-senzory

[18] FESTO. The Benefits Of Valve Terminal Technology. *Pharmaceutical Online* [online]. [cit. 2020-11-07]. Dostupné z: https://www.pharmaceuticalonline.com/doc/the-benefits-of-valve-terminal-technology-0002

[19] HAUMER, Zdeněk. Ventilové a elektrické terminály „na míru". *Automa: časopis pro automatizační techniku* [online]. Děčín, **2004**(11) [cit. 2020-10-14]. ISSN 1210-9592. Dostupné z: https://automa.cz/cz/casopis-clanky/ventilove-a-elektricke-terminaly-na-miru-2004_11_32623_2340/

[20] FESTO. Souhra elektroniky s pneumatikou – ventilové terminály CPX aneb stavebnice na druhou. *Automa: časopis pro automatizační techniku* [online]. Děčín, **2003**(8) [cit. 2020-10-14]. ISSN 1210-9592. Dostupné z: https://www.automa.cz/cz/casopis-clanky/souhra-elektroniky-s-pneumatikou-ventilove-terminaly-cpx-aneb-stavebnice-na-druhou-2003_08_28924_3610/

[21] KEMPF, Jörg. Modular Automation: Discover the Advantages of Decentralised Control in Bulk Material Handling. *Proccess worldwide* [online]. [cit. 2020-11-06]. Dostupné z: https://www.process-worldwide.com/discover-the-advantages-of-decentralised-control-in-bulk-material-handling-a-815322/

[22] FESTO. Digitalizace pneumatiky: Festo Motion Terminal VTEM. *Automa: časopis pro automatizační techniku* [online]. Děčín, **2018**(8-9) [cit. 2020-10-14]. ISSN 1210-9592. Dostupné z: https://www.automa.cz/Aton/FileRepository/pdf_articles/11710.pdf

[23] Digitalizovaný pneumatický systém: První ventily na světě, které jsou řízeny aplikacemi. *Festo* [online]. [cit. 2020-11-06]. Dostupné z: https://www.festo.com/vtem/cs/cms/10169.htm

[24] Motion Terminal VTEM: Technické údaje. *Festo* [online]. [cit. 2020-11-07]. Dostupné z: https://www.festo.com/cat/cs_cz/data/doc_cs/PDF/CZ/VTEM_CZ.PDF

[25] Mimořádně skvělá energetická efektivita: Technika pro úspornou a udržitelnou budoucnost. *Festo* [online]. [cit. 2020-11-07]. Dostupné z: https://www.festo.com/vtem/cs/cms/10170.htm

[26] Festo's Motion Terminal digitises pneumatics. *MECHATRONICA MACHINEBOUW* [online]. [cit. 2020-11-06]. Dostupné z: https://mechatronicamachinebouw.nl/product/festos-motion-terminal-digitises-pneumatics/

[27] *Festo_VTEM_DevCon Version 3.5.7.221: Library for the control (Device Control) of Motion Terminal VTEM from Festo including the elements it contains.* 2018. Dostupné také z: https://www.festo.com/net/cs_cz/SupportPortal/

[28] *Cbus-Protocol VTEM*. Dostupné také z: https://www.festo.com/net/cs_cz/SupportPortal/

[29] MX4D. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-10-17]. Dostupné z: https://en.wikipedia.org/wiki/MX4D

[30] *MX-4D: Motion EFX Experience* [online]. MediaMation, 2019 [cit. 2020-10-17]. Dostupné z: http://mx-4d.com/

[31] CORPORATE INFORMATION: The World-Leading 4D Cinema Technology Company. *CJ 4DPLEX* [online]. [cit. 2020-10-17]. Dostupné z: https://www.cj4dx.com/corporate/corporate.php#hardware

[32] KORANE, Ken. Servopneumatics embraces Cinema 4.0. *Pneumatic tips: A Fluid Power World Resource* [online]. [cit. 2020-10-17]. Dostupné z: https://www.pneumatictips.com/servopneumatics-embraces-cinema-4-0/

[33] Vše o technologii 4DX: Film s vůněmi z vás vyklepe duši! Technologie 4DX na vlastní kůži v českém kině. *CDR* [online]. [cit. 2020-10-17]. ISSN 1213-2225. Dostupné z: https://cdr.cz/clanek/vse-o-technologii-4dx-jak-vznikaji-filmy-s-pohyby-vunemi-3d

[34] 4DX. *Cinema City* [online]. [cit. 2020-10-17]. Dostupné z: https://www.cinemacity.cz/4dx#/

[35] MX4D Theatres. *MediaMation* [online]. [cit. 2020-10-17]. Dostupné z: https://www.mediamation.com/mx4d-theatres/

[36] *Kompaktní válec ADN-50: Katalogový list*. 2020. Dostupné také z: https://www.festo.com/cat/cs_cz/products_ADN_AEN

[37] Odhlučněné, tiché kompresory. *Kompresory - nářadí: Vše pro stlačený vzduch* [online]. [cit. 2020-10-17]. Dostupné z: https://www.naradi-vzduch.cz/levnekompresory-cz/eshop/38-1-Kompresory/1078-2-ODHLUCNENE-TICHE-KOMPRESORY

[38] Hlučnost, decibely, tabulka hluku. *EPrehledy* [online]. [cit. 2020-10-17]. Dostupné z: https://www.eprehledy.cz/hlucnost-decibely-priklady-hluku.php

[39] *GrabCAD.com: Couch Sofa 2 Seater* [online]. [cit. 2020-10-17]. Dostupné z: https://grabcad.com/library/couch-sofa-2-seater-1

[40] VOJÁČEK, Antonín. Programovací režimy pro PLC dle IEC 61131-3 (CoDeSys). *Automatizace.hw.cz: rady a poslední novinky z oboru* [online]. [cit. 2020-10-17]. Dostupné z: https://automatizace.hw.cz/programovaci-rezimy-pro-plc-dle-iec-611313-codesys

[41] *CODESYS* [online]. [cit. 2020-10-17]. Dostupné z: https://www.codesys.com/

[42] *NodeRED: Low-code programming for event-driven applications* [online]. [cit. 2020-10-17]. Dostupné z: https://nodered.org/

[43] Node-RED. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-10-17]. Dostupné z: https://en.wikipedia.org/wiki/Node-RED

[44] VLC media player. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-10-17]. Dostupné z: https://cs.wikipedia.org/wiki/VLC_media_player

[45] *VideoLAN: VLC Media Player* [online]. [cit. 2020-10-17]. Dostupné z: https://www.videolan.org/vlc/

[46] *Řídicí bloky CPX-CEC: Katalogový list*. 2016. Dostupné také z: https://www.festo.com/cat/cs_cz/products_CPX

[47] Modbus TCP Server/Client. *Schneider Electric* [online]. [cit. 2020-10-17]. Dostupné z: https://product-help.schneider-electric.com/Machine%20Expert/V1.1/en/m218prg/m218prg/M218_Ethernet_Configuration/M218_Ethernet_Configuration-4.htm

[48] Modbus. In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2001 [cit. 2020-10-17]. Dostupné z: https://cs.wikipedia.org/wiki/Modbus

[49] LANDSDORF, Klaud. Node-red-contrib-modbus. *Node-RED: flows* [online]. [cit. 2020-10-17]. Dostupné z: https://flows.nodered.org/node/node-red-contrib-modbus

[50] Node-red-dashboard. *Node-RED: flows* [online]. [cit. 2020-10-17]. Dostupné z: https://flows.nodered.org/node/node-red-dashboard

[51] *Festo Motion Terminal (VTEM): Malfunction codes*. 2020. Dostupné také z: https://www.festo.com/net/SupportPortal/Files/493569/VTEM__MalfunctionCodes.pdf

[52] VLC command-line help. *VideoLAN wiki* [online]. [cit. 2020-10-17]. Dostupné z: https://wiki.videolan.org/VLC__command-line__help

[53] Iohook. *Npm* [online]. [cit. 2020-10-17]. Dostupné z: https://www.npmjs.com/package/iohook

[54] Creating nodes. *Node-RED: docs* [online]. [cit. 2020-10-17]. Dostupné z: https://nodered.org/docs/creating-nodes/

[55] JANOVSKÝ, Jan. Node-red-contrib-io-hook. *Node-RED: flows* [online]. [cit. 2020-10-21]. Dostupné z: https://flows.nodered.org/node/node-red-contrib-io-hook

[56] *JSON: Introducing JSON* [online]. [cit. 2020-10-17]. Dostupné z: https://www.json.org/json-en.html

[57] Projects: user guide. *Node-RED* [online]. [cit. 2020-10-20]. Dostupné z: https://nodered.org/docs/user-guide/projects/

[58] Raspberry Pi 4 Model B. *RPishop.cz* [online]. [cit. 2020-10-22]. Dostupné z: https://rpishop.cz/raspberry-pi-4b/1599-RPI402-765756931175.html

[59] Cinema Film - Share Icon - Symbol Transparent PNG. In: *Pnghut.com* [online]. [cit. 2020-10-22]. Dostupné z: https://pnghut.com/png/4kcN32FqxS/cinema-film-share-icon-symbol-transparent-png

[60] JANOVSKÝ, Jan. *CONTROLLING MOTION APPS OF MOTION TERMINAL FESTO VTEM BY NODE-RED*. Praha, 2020. Semestrální práce. ČVUT v Praze, Fakulta strojní, Ústav přístrojové a řídicí techniky. Vedoucí práce Ing. Marie Martinásková, Ph.D.

[61] KOUBOVÁ, Jana, Patrik DOLEŽAL a Jan JANOVSKÝ. *Programování VTEM pomocí modulu CPX*. Praha, 2019. Semestrální práce. ČVUT v Praze, Fakulta strojní, Ústav přístrojové a řídicí techniky. Vedoucí práce Ing. Marie Martinásková, Ph.D.