Návrh a vývoj softwarových modulov v prostredí LabVIEW pre meranie príkonu naftových kotlov

Juraj Hanečka

Bakalárska práca
2020

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky

Univerzita Tomáše Bati ve Zlíně
Fakulta aplikované informatiky
Ústav automatizace a řídicí techniky

Akademický rok: 2019/2020

ZADÁNÍ BAKALÁŘSKÉ PRÁCE
(projektu, uměleckého díla, uměleckého výkonu)

Jméno a příjmení:	**Juraj Hanečka**
Osobní číslo:	**A17398**
Studijní program:	**B3902 Inženýrská informatika**
Studijní obor:	**Informační a řídicí technologie**
Forma studia:	**Kombinovaná**
Téma práce:	**Návrh a vývoj softwarových modulů v prostředí LabVIEW pro měření příkonu naftových kotlů**
Téma práce anglicky:	**A LabVIEW Environment Software Module: The Design and Development of Diesel Oil Boiler Power Input Measurements**

Zásady pro vypracování

1. Vypracujte literární rešerši na dané téma.
2. Navrhněte a realizujte samostatný funkční modul v programovacím prostředí LabVIEW pro měření příkonu naftových kotlů.
3. Modul bude obsahovat části realizující simulaci měřených vstupů testovací stanice, komunikaci s diferenčním průtokoměrem KRAL BEM 500 pomocí sběrnice Modbus.
4. Vytvořte kalkulační modul pro výpočet příkonu v souladu s normou EN303-1, uvažujte i možnost nastavení parametrů použitého paliva.
5. Navrhněte vhodné grafické uživatelské prostředí s možností reportu naměřených dat.

Rozsah bakalářské práce:
Rozsah příloh:
Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam doporučené literatury:

1. TRAVIS, Jeffrey a Jim KRING. *LabVIEW for everyone: graphical programming made easy and fun*. 3rd ed. Upper Saddle River, NJ: Prentice Hall, c2007. ISBN 0131856723.
2. ESSICK, John. *Hands-on introduction to LabVIEW for scientists and engineers*. 2nd ed. New York: Oxford University Press, c2013. ISBN 0199925151.
3. FRENZEL, Louis E. *Handbook of serial communications interfaces: a comprehensive compendium of serial digital input/output (I/o) standards*. Waltham, MA: Newnes, an imprint of Elsevier, [2016]. ISBN 978-012-8006-290.
4. BEATY, H. Wayne. *Handbook of electric power calculations*. 3rd ed. New York: McGraw-Hill, c2001. ISBN 9780071362986.
5. ČSN EN 303-1. *Kotle pro ústřední vytápění ? Část 1: Kotle pro ústřední vytápění s hořáky s ventilátorem ? Terminologie, všeobecné požadavky, zkoušení a značení.* Praha: Úřad pro technickou normalizaci, metrologii a státní zkušebnictví, 1999. Třídící znak 07 5303.

Vedoucí bakalářské práce: **Ing. Milan Navrátil, Ph.D.**
Ústav elektroniky a měření

Datum zadání bakalářské práce: **20. prosince 2019**
Termín odevzdání bakalářské práce: **15. května 2020**

doc. Mgr. Milan Adámek, Ph.D.
děkan

prof. Ing. Vladimír Vašek, CSc.
ředitel ústavu

Ve Zlíně dne 20. prosince 2019

Prohlašuji, že

- beru na vědomí, že odevzdáním bakalářské práce souhlasím se zveřejněním své práce podle zákona č. 111/1998 Sb. o vysokých školách a o změně a doplnění dalších zákonů (zákon o vysokých školách), ve znění pozdějších právních předpisů, bez ohledu na výsledek obhajoby;
- beru na vědomí, že bakalářská práce bude uložena v elektronické podobě v univerzitním informačním systému dostupná k prezenčnímu nahlédnutí, že jeden výtisk diplomové/bakalářské práce bude uložen v příruční knihovně Fakulty aplikované informatiky Univerzity Tomáše Bati ve Zlíně a jeden výtisk bude uložen u vedoucího práce;
- byl/a jsem seznámen/a s tím, že na moji bakalářskou práci se plně vztahuje zákon č. 121/2000 Sb. o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon) ve znění pozdějších právních předpisů, zejm. § 35 odst. 3;
- beru na vědomí, že podle § 60 odst. 1 autorského zákona má UTB ve Zlíně právo na uzavření licenční smlouvy o užití školního díla v rozsahu § 12 odst. 4 autorského zákona;
- beru na vědomí, že podle § 60 odst. 2 a 3 autorského zákona mohu užít své dílo – diplomovou/bakalářskou práci nebo poskytnout licenci k jejímu využití jen připouští-li tak licenční smlouva uzavřená mezi mnou a Univerzitou Tomáše Bati ve Zlíně s tím, že vyrovnání případného přiměřeného příspěvku na úhradu nákladů, které byly Univerzitou Tomáše Bati ve Zlíně na vytvoření díla vynaloženy (až do jejich skutečné výše) bude rovněž předmětem této licenční smlouvy;
- beru na vědomí, že pokud bylo k vypracování bakalářské práce využito softwaru poskytnutého Univerzitou Tomáše Bati ve Zlíně nebo jinými subjekty pouze ke studijním a výzkumným účelům (tedy pouze k nekomerčnímu využití), nelze výsledky diplomové/bakalářské práce využít ke komerčním účelům;
- beru na vědomí, že pokud je výstupem bakalářské práce jakýkoliv softwarový produkt, považují se za součást práce rovněž i zdrojové kódy, popř. soubory, ze kterých se projekt skládá. Neodevzdání této součásti může být důvodem k neobhájení práce.

Prohlašuji,

- že jsem na bakalářské práci pracoval samostatně a použitou literaturu jsem citoval. V případě publikace výsledků budu uveden jako spoluautor;
- **že odevzdaná verze bakalářské práce a verze elektronická nahraná do IS/STAG jsou totožné.**

Ve Zlíně, dne
22/07/2020

Juraj Hanečka, v.r.
podpis diplomanta

ABSTRAKT

Testovanie príkonu olejových kotlov pozostáva z hmotnostného prietoku a energetickej hodnoty paliva. Výhrevnosť, teda pridaná energia paliva je známa, dohľadateľná konštanta. Meranie prietoku paliva pomocou prietokomera vyžaduje korekciu hustoty paliva z dôvodu vplyvu prostredia teda teploty. Prietokomery poskytujú veľkú škálu rozsahov a možnosti sériovej komunikácie. Na základe týchto informácií bol navrhnutý softvér v prostredí LabVIEW pre meranie naftových kotlov Toto riešenie poskytuje presné a rýchle meranie z ohľadom na parametre paliva s možnosťou vizualizácie priebehu spotreby paliva a tvorby reportov.

Kľúčová slová: naftový kotol, prietokomer, LabVIEW, MODBUS, komunikácia, simulácia

ABSTRACT

Testing the power input of oil boilers consists of the mass flow and the fuels energy value. The calorific value, i.e. the added energy of the fuel is known, traceable constant. The measurement of the fuel flow by a flow meter requires a correction of the fuel density due to the influence of the environment, i.e. the temperature. Flowmeters provide a wide range of scales and serial communication options. Based on this information, software was designed in the LabVIEW environment for the measurement of oil boilers. This solution provides accurate and fast testing with respect to fuel parameters with the possibility of visualizing the course of fuel consumption and reports creation.

Keywords: oil boiler, flow meter, LabVIEW, MODBUS, communication, simulation

Všetko, čo sa stane sa stane. Všetko, čo pritom, ako sa stane, spôsobí, aby sa stalo niečo iné, spôsobí, aby sa stalo niečo iné. Všetko, čo pritom, ako sa stane, spôsobí, aby sa stalo znova sa stane znova. Nemusí sa to však stať presne v tomto poradí.
Douglas Noël Adams (11. marec 1952,– † 11. máj 2001,)*

OBSAH

ÚVOD

Teplo, teda tepelná energia je vlastná energia, ktorú teleso príjme, alebo odovzdá. Teplo je pre človeka vnímané ako komfortná zóna pre prežitie. Toto teplo sa vyrába, respektíve transformuje z rôznych zdrojov z obnoviteľných či neobnoviteľných. Jedným zo zdrojov sú i vykurovacie olejové palivá a s nimi spojené spaľovacie olejové kotle. Pre navrhnutie a zrealizovanie takéhoto zariadenia, ako je napríklad kondenzačný olejový kotol je potreba nie len konštruktérov a elektrotechnikov, ale i inžinierov, ktorí budú mať možnosť overiť vypočítanú spotrebu paliva či pridaného príkonu, teda energie, ktorá sa v konečnom dôsledku transformuje na teplo.

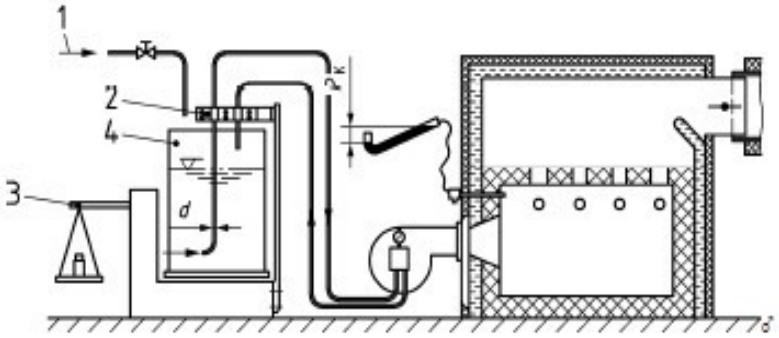
Pre vypočítanie energie, ktorú zariadenie spotrebuje sú potrebné dva parametre. Spotreba paliva a jeho energetická hodnota k hmotnostnému prietoku. Energetická hodnota oleja je daná fyzikálnymi vzťahmi pre výpočet kalorickej hodnoty podľa kompozície zloženia daného paliva. A celková spotreba je meraná v čase. Meranie spotreby paliva je možné realizovať niekoľkými spôsobmi, kde ich základné rozdelenie môžeme definovať ako priamu a nepriamu metódu. Priamou metódou dostávame informáciu o spotrebovanej hmotnosti priamo odčítaním z váhy v čase, teda koľko kilogramov oleja sa spotrebovalo za daný časový interval. Nepriama metóda spočíva v meraní objemu prietoku paliva taktiež v danom časovom intervale, ale jej hmotnostný prietok je treba prepočítať pomocou hustoty paliva a jeho teploty.

Vďaka prietokomerom je možnosť pokrytia rôznych typov komunikácií, rozsahov merania či teploty a viskozity paliva široká. Preto pre aplikáciu do testovacích laboratórií alebo overovacích skúšobných inštitútov je vhodnejšia nepriama metóda. Táto metóda bola aplikovaná i v bakalárskej práci s cieľom dosiahnuť porovnateľné výsledky s priamou metódou a zníženie času potrebného pre samostatné meranie. Neposlednou časťou je digitalizácia procesu merania s možnosťou vizualizovať merané veličiny a taktiež prezentovať výsledky merania formou finálneho test reportu.

I. TEORETICKÁ ČASŤ

1 LITERÁRNA REŠERŠ

Pre danú tému bolo preskúmaných niekoľko článkov zaoberajúcich sa problematikou merania prietokov, spotreby paliva a vyjadrenia spotreby v čase. Zadanie práce je špecifikované k meraniu priamo naftových kotlov podľa normy EN 303-1 pre vykurovacie kotly s tlakovými horákmi. Nakoľko v súčasnosti sa zariadenia viac orientujú na nižšiu záťaž emisií a taktiež na nižšiu spotrebu, bola k rešerši zahrnutá norma EN 304, vykurovacie kotly, pravidlá skúšania vykurovacích kotlov s rozprašovacími horákmi na kvapalné palivá [9] Podľa normy je popísaný spôsob merania na Obrázok 1 Spôsob merania spotreby paliva. Zariadenie má inštalované dýzy v nádobe (zásobník oleja) s palivom. Spotreba sa meria priamou metódou odčítania poklesu váhy spáleného vykurovacieho oleja za určitý čas. Konečná spotreba sa prevedie na spotrebu v $kg.sec^{-1}$. Pri minimálnom príkonu naftových kotlov pre domácnosť, napríklad 6 kW a podľa vzťahov na výpočet daného príkonu z kapitoly 4.1 Kalkulácia príkonu olejových kotlov je odpovedajúca spotreba rovná 0,00014 $kg.sec^{-1}$.



Legenda

1	prívod paliva	4	zásobník oleja
2	držiak rozvodov	5	priemer rúrky na prívod oleja
3	váha	P_K	tlak v spaľovacej komore

Obrázok 1 Spôsob merania spotreby paliva. Zdroj EN 304

Pre uplatnenie požiadavky z danej normy minimálne troch platných číslic na meriacom zariadení je treba využiť váhy so zobrazením šiestich desatinných miest. Druhým variantom je váha v inom rozsahu napríklad $g.sec^{-1}$a manuálny prepočet. Treťou možnosťou môže byť predĺženie času odčítavania s kombináciou váhy s rozsahom $kg.sec^{-1}$ o troch desatinných zobrazovacích miestach (najčastejší variant). V takomto prípade je minimálny nut-

ný čas pre meranie 12 minút. Dané informácie boli získané formou ankety u nemenovaných testovacích inštitútov z Rakúska, Holandska a Českej republiky (viď Príloha P 2: Anketa formy merania).

Veľmi pozoruhodný prístup je popísaný v článku od autora NAUMCHIK, I.V. [10]. Autori sa zaoberajú problémami stanovenia hmotnostného prietoku korozívnych a nestabilných kvapalín, berúc do úvahy ich skutočnú hustotu. Objavený prietokomer na meranie prietoku kvapalín na báze Venturiho trubice, vynikajúci prístroj na meranie hustoty kvapaliny, ktorá umožňuje určiť jej hmotnostný tok. Článok pojednáva o výhodách merania touto metódou, avšak v súčasnosti existuje široká škála prietokomerov založených na rôznych fyzikálnych princípoch, preto táto časť výsledkov nebola aplikovaná v rešerši a zaoberalo sa len dopadom merania prietoku k hustote tekutín. Pri meraní hmotnostného prietoku tekutín, najmä agresívnych a nestabilných, je nevyhnutné zmerať jeho hustotu, preto sa navrhuje použiť metódu ultrazvukového merania, to znamená určenie rýchlosti prechodu ultrazvukovej vlny meraným médiom. Súčinom hustoty kvapaliny merané pomocou zvukovej vlny a hmotnostného prietoku kvapaliny odhalíme hmotnostný prietok paliva. Pri meraní bol odhalený vplyv okolitých vlastností na aktuálnu hustotu meraného média, a teda i chybu merania v prípade zanedbania týchto faktorov. Výsledky článku ukázali, že pre sľubné oblasti ďalšieho výskumu by do úvahy mali byť zapracované:

a) výpočet a experimentálne spresnenie charakteristík prietokomeru, ak je to potrebné tak i zvýšená presnosť merania s ohľadom na skutočnú aktuálnu hustotu kvapaliny.

b) experimentálne štúdie o širokej škále tekutín rôznych zložení s účelom získania kalibračných závislostí, napríklad na teplote.

1.1 DISKUSIA O POZNATKOCH

Pre danú tému boli preskúmané dva rôzne prístupy k možnostiam merania príkonu, respektíve spotreby paliva.

Priama metóda, kde spotreba je priamo meraná v závislosti na váženom poklese vstupného paliva v čase. Teda je nevyhnutné odčítavanie počiatočnej a konečnej hodnoty váh v definovanom časovom úseku. Z hodnôt poklesu a časového rastru je možné odvodiť hmotnostný prietok nutný ku konečnému určeniu príkonu.

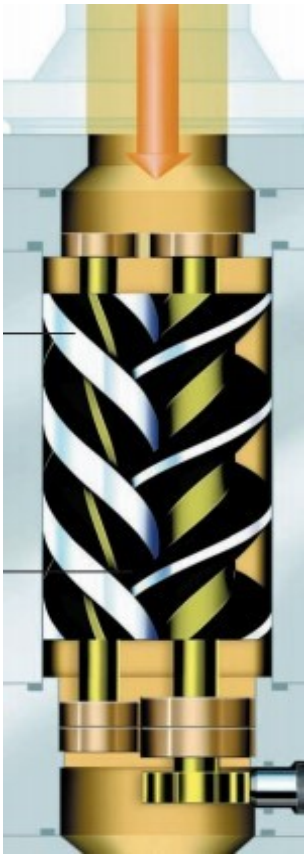
Nepriama metóda využiteľná pri priemyselných aplikáciách, kde investície do presného prietokomeru sú adekvátne k frekvencii meraní. Prístup týmto spôsobom je náchylný k odchýlkam prepočtu objemového prietoku na hmotnostný z dôvodu hustoty média v závislosti od prostredia (napr. teplote).

2 POPIS MERANIA PRIETOKU

U viac ako 40 % všetkých meraní kvapalín, plynov a pary uskutočňovaných v priemysle sa stále využívajú bežné typy prietokomeru s diferenčným tlakom, to znamená, clonová doska, Venturiho trubica a dýza. Avšak prietokomer použitý v aplikácii tejto práce je moderný axiálny turbínový prietokomer. Ak je správne nainštalovaný a kalibrovaný, potom sa jedná o spoľahlivé zariadenie, schopné poskytovať najvyššiu presnosť dosiahnuteľnú akýmkoľvek v súčasnosti dostupným snímačom prietoku, nie len pre kvapalinu, ale aj pre meranie objemového prietoku plynu. Dnes, vzhľadom na obrovský úspech tohto princípu, sú prietokomery s axiálnymi turbínami rôznych prevedení. Prednostne aplikácie v ktorých sa v mnohých prípadoch vyžaduje presnosť, spoľahlivosť a možnosť prepojenia, nie sú len priemyselné odvetvia zaoberajúce sa meraním prietoku vody a zemného plynu, ale taktiež ropy, petrochemického, chemického procesu, či kryogeniky.

V súčasnej literatúre sú opísané iba dva prístupy na analýzu výkonu axiálnej turbíny. Prvý prístup opisuje hnací moment kvapaliny z hľadiska výmeny hybnosti, zatiaľ čo druhý ho opisuje z hľadiska aerodynamického zdvihu pomocou teórie krídla. Jedným z významných priekopníkov v prístupe k dynamike je W. F. Z. Lee, ktorý pomocou tohto prístupu neskôr pokračoval vynálezom jedného z mála úspešných prietokomerov s dvoma rotorovými turbínami [6].

Axiálne turbíny s dvoma rotormi majú výkonové vlastnosti, ktoré sa nenachádzajú v konštrukciách s jedným rotorom. V roku 1981 [7] bol vydaný americký patent na samo opravný, samo kontrolný obehový turbínový prietokomer, ktorý v súčasnosti vyrába výlučne spoločnosť Equimeter, Inc. a predáva sa ako Auto-Adjust. Zahŕňa dva tesne spojené turbínové rotory, ktoré sa otáčajú rovnakým smerom. Predný rotor je hlavný rotor a druhý rotor má omnoho plytší povrch. Nepretržitá a automatická korekcia chýb merania v dôsledku premenlivého trenia ložiska sa dosahuje výpočtom prietoku na základe rozdielu medzi rýchlosťami rotora. [8].



Obrázok 2 Turbína s dvomi rotormi Zdroj: Kral Volumeter® – OMG Series. Universal Flowmeters

3 POPIS SÉRIOVÝCH KOMUNIKÁCIÍ

Na začiatku šesťdesiatych rokov minulého storočia bol výbor pre normalizáciu, dnes známy ako združenie Electronic Industries Association (EIA). Tento vyvinul spoločný štandard rozhrania pre zariadenia na dátovú komunikáciu. V tom čase sa dátová komunikácia považovala za digitálnu výmenu údajov medzi centrálne umiestnenými sálovými počítačmi a vzdialenými počítačovými terminálmi, alebo prípadne medzi dvoma terminálmi bez počítača nad nimi. Tieto zariadenia boli spojené telefónnymi hlasovými linkami a následne sa vyžadoval modem na každom konci na preklad signálu. Aj keď je koncepcia jednoduchá, existovalo veľa príležitosti na chyby v prenášaných údajoch, ku ktorým došlo pri prenose údajov cez analógový kanál. Predpokladalo sa, že je potrebná norma pre uskutočnenie spoľahlivej komunikácie a taktiež, umožnenie prepojenia zariadení vyrobených rôznymi výrobcami. Zrodili sa tieto myšlienky vytvorenia štandardu (RS232C). V tomto je špecifikované napätie signálu, časovanie signálu, funkcia signálu, protokoly na výmenu

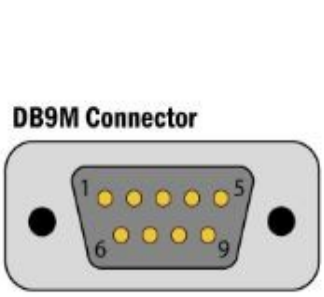
informácií i mechanické konektory. Sériovú komunikáciu je možné rozdeliť do rôznych skupín na základe ich princípov činnosti [5].

Pre sériovú komunikáciu existujú mnohé aplikácie a možnosti prevedenia ako z pohľadu hardvérového riešenia, tak i z pohľadu softvérových protokolov. Najvyužívanejšie skupiny sú RS-232, RS-422, RS-485, USB, CAN a tiež škála protokolov ako napríklad IAC, MNS, COMLI, MODBUS RTU. Pre realizáciu bol použitý protokol MODBUS RTU v skupine RS-232.

3.1 SÉRIOVÝ PORT RS-232

Ako už bolo spomenuté, RS-232 je port vyvinutý a definovaný EIA v 60. rokoch 20. storočia a bol použitý v skorej sériovej dátovej komunikácii. Vďaka svojej jednoduchosti a popularite sa RS-232 široko uplatňuje vo všetkých oblastiach dátovej komunikácie vrátane priemyselnej, obchodnej, vzdelávacej a dokonca aj spotrebnej elektroniky. RS-232 patrí k plne duplexnému komunikačnému portu, čo znamená, že odosielateľ aj príjemca si môžu vymieňať informácie súčasne. Komunikačný protokol polovičného duplexu umožňuje používateľom (odosielateľom a prijímačom) odosielať alebo prijímať informácie medzi sebou iba v rôznych časových obdobiach; nemôžu odosielať a prijímať súčasne. To znamená, že príjemca musí čakať, až odosielateľ dokončí zasielanie informácií, potom príjemca môže vyzdvihnúť informácie odosielateľa a reagovať na nich. V rovnakom čase môže byť kontrolovaný iba jeden prenos údajov, teda môže pracovať iba jeden používateľ, odosielateľ alebo prijímač. Najjednoduchší port RS-232 využíva tri vodiče: jeden vodič sa používa na odosielanie informácií, druhý sa používa na prijímanie informácií a tretí vodič funguje ako zem, alebo referencia medzi nimi. Informácie prenášané na drôtoch RS-232 sú znázornené ako sled binárnych bitov a hodnoty týchto binárnych bitov sú spojené s dvoma úrovňami napätia: 12 voltov (logická 1) a 12 voltov (logická 0). Rýchlosť prenosu údajov je riadená pomocou prenosových rýchlostí (počet binárnych bitov, ktoré je možné prenášať za sekundu), ktoré môžu byť uvedené a nastavené používateľom pred prenosom údajov. V prvých dňoch bola rýchlosť prenosu údajov relatívne pomalá kvôli pomalým rýchlostiam procesora. Typické prenosové rýchlosti boli 1 200, 4 800 a 9600. Prenosová rýchlosť používaná v dnešnej sériovej dátovej komunikácii sa výrazne zvýšila na 19 200, 38 400 a dokonca vyššie. Stručne povedané, port RS-232 je navrhnutý na komunikáciu s miestnymi zariadeniami a bude podporovať jeden ovládač a jeden prijímač. Typická prenosová vzdia-

lenosť protokolu RS-232 je menšia ako 15 metrov. Pre zvýšenie tejto vzdialenosti a zníženie hluku a rušenia bol vyvinutý RS-422[5].



RS232 Pin Out

Pin #	Signal
1	DCD
2	RX
3	TX
4	DTR
5	GND
6	DSR
7	RTS
8	CTS
9	RI

Obrázok 3 Seriový port RS232 s popisom pinov Zdroj: www.sealevel.com/product/usb-to-1-port-rs-232

Fyzické zapojenie sa dá odvodiť zo štandardného usporiadania pinov konektora RS-232. Teda pin 2 pre čítanie (RX), pin 3 pre zápis (TX) a pin 5 pre vyrovnanie potenciálov (GND).

3.2 SÉRIOVÝ PORT RS-485

RS-485 je podobný portu RS-422 v protokole prenosu diferenciálneho signálu, ale má schopnosť vybudovať viacbodový komunikačný systém. To znamená, že viac terminálov alebo počítačov, ktoré sa považujú za uzly, môžu byť pripojené k spoločnej zbernici a každý uzol môže pracovať ako odosielateľ alebo príjemca informácií z tejto zbernice. Každý terminál, alebo počítač pripojený k zbernici má funkčnosť ovládania tristátu a jedinečnú adresu (alebo ID) a komunikácia medzi odosielateľom a príjemcom sa vykonáva na základe tohto ID. Sieť RS-485 môže byť pripojená v dvoj alebo štvorvodičovom režime. Maximálna dĺžka kábla môže byť až 1200 metrov a používa sa systém prenosu diferenciálneho napätia. Typické použitie RS-485 je pre pripojenie jedného PC na niekoľko adresovateľných zariadení, ktoré zdieľajú ten istý kábel [5].

3.3 PROTOCOL MODBUS RTU

MODBUS RTU je štandardný protokol široko rozšírený kvôli jeho ľahkému použitiu a skutočnosti, že podporuje komunikáciu cez širokú škálu médií, ako sú drôty, optická vlákna, rádio a telefón. MODBUS sa vykonáva sériovo a asynchrónne podľa master/slave princípu a v jednom smere súčasne. MODBUS sa používa hlavne na čítanie a písanie premenných medzi zariadeniami riadiacej siete pomocou point-to-point, alebo viac prístupovej komunikácii. Rámcovanie správ je implementované v režime RTU, čo je binárny formát. Protokol MODBUS je určený na bezpečný prenos údajov, kontrolu každého bajtu, ako aj kontrolu chyby prenosu celej správy [5].

Protokolom je podporované množstvo príkazov pozri Tabuľka 1 . Aplikácia alebo programátor majú prístup k funkciám protokolu prostredníctvom funkčných blokov podľa normy IEC 61131-5. Softvér protokolu prekladá požiadavku z pripojenia, výnimky, čítania a zápisu blokov do protokolových príkazov MODBUS.

Tabuľka 1 Príkazy protokolu

Protokol	***Popis***	***Protokol***	***Popis***
FC1	Čítanie stavu cievky	FC6	Prednastaviť jednu cievku
FC2	Čítanie stavu vstupu	FC7	Čítanie výnimky
FC3	Čítanie holdingového registra	FC8	Diagnostická požiadavka
FC4	Čítanie vstupného registra	FC15	Vynútenie viacerých cievok
FC5	Vynútiť jednu cievku	FC16	Prednastavenie viacerých registrov

Výkon je ovplyvnený rýchlosťou prenosu, dĺžkou správy a samotnou aplikáciou. Pre kanály RS-232 je možné zvoliť prenosovú rýchlosť medzi 2400 a 19200 bit/s. Na odoslanie jedného bajtu je potrebných 11 bitov (počiatočný bit, 8 dátových bitov, paritný bit a stop bit). Následne je možné zaslať 9600/11 = 872 bajtov za sekundu, ak je prenosová rýchlosť 9600. Komunikácia prebieha sériovo a asynchrónne podľa protokolu princíp master/slave (alebo klient/server). Spustí sa hlavný kanál systému prenosu správ, zatiaľ čo sys-

tém, ktorý jednoducho funguje ako slave odpovedá na volania z hlavného zariadenia cez podradený kanál [5].

3.3.1 ZOSTAVENIE A ČÍTANIE SPRÁVY

Pre zostavenie požiadavky na dopyt pomocou protokolu MODBUS sa odošle správa, jej tvar je popísaný v Tabuľka 2. Správa v hexadecimálnom tvare bude obsahovať na prvej pozícii číslo zariadenia slave, s ktorým sa komunikácia nadviaže. Druhá pozícia je vyhradená pre kód funkcie z Tabuľka 1. Tretia a štvrtá pozícia slúži k adrese premennej, ktorú chceme čítať (napríklad hodnota prietoku). Piata a šiesta pozícia slúži pre určenie z koľkých registrov je treba správu čítať (16 alebo 32 bitov). Posledné pozície sú určené pre kontrolný súčet.

Tabuľka 2 Správa na odoslanie

Časť správy (HEX)	***Popis***
01	Adresa zariadenia
03	Kód funkcie
4100	Adresa prvého registru
0002	Počet požadovaných registrov
D037	Kontrolní súčet CRC

Odoslaný rámec má tvar : 0103 4100 0002 D037

a ako odpoveď prijatý rámec má tvar : 0103 0400 0954 7C11 70, zo zariadenie slave protokolu MODBUS RTU v Tabuľka 3.

Tabuľka 3 Odpoveď na dopyt

Časť správy (HEX)	***Popis***
01	Adresa zariadenia
03	Kód funkcie
04	Počet bajtov
00	Horná časť hodnoty registrov AO 0
09	Dolná časť hodnoty registrov AO 0
5A	Horná časť hodnoty registrov AO 1
7C	Dolná časť hodnoty registrov AO 1
11	Horná časť hodnoty CRC
70	Dolná časť hodnoty CRC

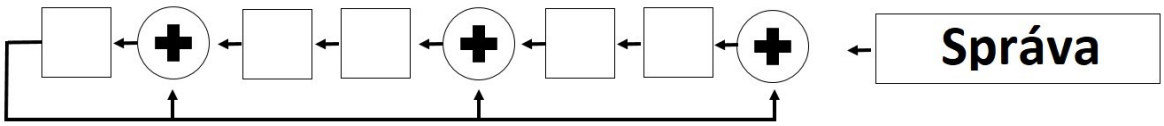
Hodnota registrov v hexadecimálnom tvare je teda 0009 547C, čo v desiatkovej sústave predstavuje číslo 611452.

3.3.2 KONTROLNÝ SÚČET CRC

CRC je algoritmus na kontrolu chýb široko používaný v dátovej komunikácii. Jeho celé meno je Cyklický redundantný súčet („*Cyclic redundancy check*“). Algoritmus CRC je založený na delení modulo-2 (zvyšok po delení dvomi). Prijaté údaje možno považovať za správu, ktorá sa tiež nazýva polynom správy. Správa by mala byť rozdelená deliteľom, ktorý sa tiež nazýva generátorový polynóm. Tento proces delenia sa vykonáva v delení modulo-2, čo môže byť ekvivalentné posunu v smerovom registri plus niektoré exkluzívne

operandy OR (XOR). Po spracovaní všetkých údajov v tomto delení modulo-2 je hodnota zvyšku výsledkom CRC alebo zvyškom CRC. Na rozdiel od tradičného delenia, CRC venuje pozornosť iba zvyšku, nie kvocientu. Algoritmus CRC je definovaný ako zvyšok z delenia modulo-2, v ktorom sa počet 0 bitov rovná počtu bitov v zvyšnom registri pripojených k správe. Pri normálnych operáciách binárneho delenia sa deliteľ vždy odpočíta od zvyšku a kvocient je nastavený na 1, ak je výsledok odčítania pozitívny, inak bude kvocient nastavený na 0. Ale v algoritme CRC sa používa logická operácia XOR (exkluzívna disjunkcia, teda pravdivostná hodnota je 1 práve vtedy, keď nemajú obidva argumenty rovnakú hodnotu) [5].

V praxi sa používajú posuvné registre. Jeden z nich je popísaný v Obrázok 4 Posuvný register CRC. Teda ak existuje správa v binárnom kóde 010011b. Kontrolný CRC kód bude o jeden bit menší než počiatočná správa, takže sa priradí 5 nulových bitov k našej správe, teda 01001100000. Register taktiež začne s nulovými bitmi na pozíciách. Posuvný register začne absorbovať postupne každý bit správy. Každým posunom sa vytlačí jeden bit z posuvného registra, ktorý tvorí jeden argument a druhým argumentom sa stáva nasledujúci bit správy. Výsledná operácia XOR sa zapíše do registra. Taktiež sa prevedú operácie XOR-u argumentov vo vnútri posuvného registra. Pri spracovaní posledného bitu správy sa v registroch nachádza kontrolný CRC kód 11101. Naša kompletná správa pozostáva z originálnej správy a kontrolného kódu, kde dostávame : 101001111101b.



Obrázok 4 Posuvný register CRC Zdroj: vlastný

Táto správa je poslaná a na strane príjemcu prebehne rovnaký postup. Prijatá správa je transformovaná cez posuvný register. Táto správa pri prenose nebola poškodená, ak posledné bity správy nachádzajúce sa na pozíciách registra sú bitmi nulovými. Účinnosť metódy je 99 %.

4 KALKULÁCIA PRÍKONU OLEJOVÝCH KOTLOV

Pre kalkuláciu olejových kotlov sa odvolávame na platnú Európsku normu STN EN304: 2018. Táto norma platí na stanovenie vlastností vykurovacích kotlov a kombinovaných kotlov na kvapalné palivá. Požiadavky na vykurovacie parametre sú stanovené v EN 303-1: 2017 a EN 303-2: 2017.

Pri kotloch na kvapalné palivá sa tepelný príkon QB vypočíta takto:

$Qb = B \times NCV$ [W] [1]

kde B je hmotnostný tok paliva [$kg.sec^{-1}$];

NCV výhrevnosť („net calorific value“) [$J.kg^{-1}$] [9]

4.1 STANOVENIE VÝHREVNOSTI PALIVA

V nasledujúcich kapitolách je popísaný spôsob určenia výhrevnosti paliva niektorých spaľovacích olejov.

4.1.1 ĽAHKÉ VYKUROVACIE PALIVÁ

a) Ak nie je výhrevnosť stanovená kalorimetrickou metódou alebo ak nie je vykonaný celkový rozbor paliva, môže sa výhrevnosť vykurovacieho oleja stanoviť s dostatočnou presnosťou takto:
   $NCV = 42{,}689\ MJ.kg^{-1}$
b) Ak je známa hustota paliva ρ_{15} a hmotnostný podiel síry (napr. rozborom paliva), môže sa výhrevnosť vykurovacieho oleja vypočítať zo vzorca:
   $NCV = 52{,}92 - (11{,}93 \times \rho_{15}) - (0{,}3 - S)$ [$MJ.kg^{-1}$] [2]

   Kde:
   ρ_{15} je hustota vykurovacieho oleja pri teplote 15 °C v [$kg.m^{-3}$];
   S je hmotnostný podiel síry [-][9].

4.1.2 PETROLEJ

a) Ak nie je výhrevnosť stanovená kalorimetrickou metódou alebo ak nie je vykonaný celkový rozbor paliva, môže sa výhrevnosť petroleja stanoviť s dostatočnou presnosťou takto:
   $NCV = 43{,}300\ MJ.kg^{-1}$
b) Ak je zistená hustota paliva a hmotnostný podiel síry (napríklad rozborom paliva), môže sa výhrevnosť petroleja vypočítať podľa vzorca:
   $NCV = 52{,}92 - (11{,}93 \times \rho_{15}) - (0{,}3 - S)\ [MJ.kg^{-1}]$ [3]

   Kde:
   ρ_{15} je hustota vykurovacieho oleja pri teplote 15 °C v $[kg.m^{-3}]$;
   S je hmotnostný podiel síry [-][9].

5 VÝVOJOVÉ PROSTREDIE LABVIEW

LabVIEW, skratka pre Laboratory Virtual Instrument Engineering Workbench, je programovacie prostredie, v ktorom sa vytvárajú programy pomocou grafického zápisu (spájanie funkčných uzlov prostredníctvom drôtov, cez ktoré sú sprostredkované toky údajov); v tomto ohľade sa líši od tradičných programovacích jazykov ako C, C ++ alebo Java, v ktorých sa programuje textom. LabVIEW je však oveľa viac ako programovací jazyk. Je to interaktívny systém na vývoj a vykonávanie programov určený pre ľudí ako sú vedci a inžinieri, ktorí potrebujú programovanie ako súčasť svojej práce. Program LabVIEW pozostáva z jedného alebo viacerých virtuálnych nástrojov (VI). Virtuálne nástroje sú nazývané tak, pretože ich vzhľad a fungovanie často napodobňuje skutočné fyzické nástroje. Za scénami sú však analogické hlavným programom, funkciám a podprogramom populárne programovacie jazyky ako C alebo Basic. Neoddeliteľnou súčasťou je prepojenie s meracími prístrojmi, na ktoré má National Instruments patentované zberače dát s prevodníkmi prispôsobenými knižniciam LabVIEW. Avšak pre našu aplikáciu bola využitá sériová komunikácia, ktorá je ďalším populárnym prostriedkom prenosu údajov medzi dvomi počítačmi. LabVIEW môže vykonávať sériovú komunikáciu (štandardy RS-232, RS-422 alebo RS-485) pomocou vstavaného alebo externe pripojeného sériového portu vášho počítača (napríklad sériový adaptér USB) [4].

Jedno virtuálne zariadenie, teda virtuálny nástroj (VI) pozostáva z troch základných častí: predný panel, blokový diagram a vstupy a výstupy.

5.1 PREDNÝ PANEL

Predný panel je okno, cez ktoré používateľ komunikuje s programom. Aby sa mohli zadávať údaje na vykonanie programu, musí byť v mieste spustenia VI otvorený i predný panel. Predný panel je tiež nevyhnutný, pretože sa tu zobrazujú vytvárané výstupy programu [4].

5.2 VSTUPY A VÝSTUPY

Predný panel je primárne kombináciou ovládacích prvkov, vstupov a indikátorov, výstupov. Ovládacie prvky simulujú typický vstup pomocou predmetov, ktoré sa môžu nachádzať na bežnom nástroji, ako sú gombíky a prepínače. Ovládacie prvky umožňujú užívateľovi zadávať hodnoty, dodávajú údaje do blokovej schémy VI. Indikátory zobrazujú výstupné hodnoty vyrobené programom.

Zjednodušene:

Ovládacie prvky = Vstupy od používateľa = zdroj údajov

Indikátory = Výstupy pre používateľa = Destinácie alebo "Sinks" pre dáta [4].

5.3 BLOKOVÝ DIAGRAM

Okno blokovej schémy obsahuje grafický zdrojový kód LabVIEW VI. Blokový diagram LabVIEW zodpovedá riadkom textu, ktoré sa nachádzajú v bežnejšom jazyku, napríklad C alebo BASIC, považovaný za skutočný spustiteľný kód. Bloková schéma sa vytvára spojením objektov, ktoré vykonávajú špecifické funkcie. Využívajú sa rôzne komponenty blokovej schémy, terminály, uzly a drôty [4].

V blokovom diagrame sa pracovalo s rôznymi paletami knižníc a funkciami poskytovaných programom LabVIEW. Okrem štandardných slučiek „while“ alebo „for“ používaných vo všetkých programovacích jazykoch, boli využité i viac špecifické štruktúry pre toto vývojové prostredie.

5.3.1 POSUVNÉ REGISTRE

Posuvné registre, ktoré sú k dispozícii pre slučky „while“ a „for“, sú špeciálnym typom premennej, používanej na prenos hodnoty z jednej iterácie slučky do ďalšej. Sú jedinečné a potrebné. Posuvný register môže byť nakonfigurovaný tak, aby si pamätal hodnoty z niekoľkých predchádzajúcich iterácií. To je užitočná funkcia, keď užívateľ potrebuje spriemerovať hodnoty údajov získané v rôznych iteráciách. Kým sa VI nezatvorí, LabVIEW nezruší hodnoty uložené v zozname zásobníka a nevyberie ich z pamäte. Inými slovami, ak sa spustí VI obsahujúce neinicializovaný posun registrov, počiatočné hodnoty pre nasledujúci beh budú tie, ktoré zostali v zásobníku z predchádzajúceho behu [4].

5.3.2 SPÄTNE-VÄZOBNÝ UZOL

LabVIEW vo všeobecnosti neumožňuje vytvoriť „cyklus“, v ktorom sa používa výstup bloku kódu, pretože vstup pre ten istý blok kódu a pokus o to spôsobí zlomenie káblov, teda ak sa pokúsime spojiť výstup bloku do jeho vlastného vstupu. Cyklus je prerušený kvôli pravidlám toku údajov, v ktorom sa uvádza, po prvé uzol sa nemôže vykonať, kým nedôjde k toku údajov do všetkých jeho vstupných terminálov a po druhé, dáta nevytečú z výstupných terminálov uzla, kým uzol nedokončí vykonávanie. Pretože "cyklus" využíva výstupné terminály ako zdroj vstupných terminálov, údaje nemôžu nikdy prúdiť do systému vstupných terminálov. Dáta nikdy nevytečú z výstupných terminálov, z toho vyplýva, že uzol nebude bežať kým dáta nepreniknú do vstupných terminálov, a to paradoxne. Ak vykonávaná inštrukcia prebieha vo vnútri slučky „while“ alebo „for“, môže sa medzi výstupným terminálom a vstupným terminálom umiestniť uzol spätnej väzby a kód bude funkčný. Pre pochopenie, ako presne funguje spätne-väzobný uzol („Feedback Node“), je treba si uvedomiť, že spätne-väzobný uzol je naozaj iba maskovaný posúvací register [4].

5.3.3 ŠTRUKTÚRA PRÍPADU

Štruktúra prípadu v LabVIEW je spôsob vykonávania podmieneného textu, niečo ako príkaz „if-then-else“. Ak je k terminálu selektora pripojená logická hodnota, štruktúra má dva prípady: FALSE a TRUE. Ak je k selektoru zapojený číselný alebo reťazový dátový typ, štruktúra môže mať od jedného do takmer neobmedzeného počtu prípadov. Spočiatku sú k dispozícii iba dva prípady, ale ďalšie sa môžu ľahko pridať. Môže byť zadaná viac ako jedna hodnota prípadu oddelená čiarkami. Štruktúra musí mať prípady, ktoré spracúvajú všetky možné hodnoty, ktoré by mohli prísť do systému ako selekčný terminál, inak VI bude prerušené (nie je možné spustiť). Toto rieši nastavenie základného prípadu "Default" Je to jednoduchý spôsob, ako toho dosiahnuť, nie je to však jediný spôsob. Rámce pre číselné údaje možno nakonfigurovať aj na spracovanie a rozsah hodnôt; napríklad „3 ..“ spracováva všetko od 3 po maximálne celé číslo. Hodnota typu vstupných údajov „2..2“ spracováva všetko v rozsahu 2 až 2 [4].

5.3.4 ŠTRUKTÚRA SEKVENCIE ALEBO SKLADANÁ

Poradie vykonávania programu určením usporiadania jeho prvkov v určitom slede je nazývaný riadiaci tok. Visual Basic, C a väčšina ďalších procedurálnych programovacích jazykov sú vlastným kontrolným tokom, pretože príkazy sa vykonávajú v poradí, v akom sa nachádzajú v programe. LabVIEW používa štruktúru sekvencie na získanie riadiaceho toku v oblasti rámca toku údajov. Sekvenčná štruktúra je usporiadaná skupina rámcov, ktoré sa vykonávajú postupne. Vykonáva sa sekvenčná štruktúra snímka 0, nasledovaná snímkou 1, potom snímkou 2, až kým sa nespustí posledný snímok. Iba vtedy, keď sa posledný rám dokončí, opustia údaje štruktúru. Sekvenčná štruktúra má dve prístupy: plochá sekvenčná štruktúra a skladaná sekvencia. Plochá štruktúra je takmer identická a má rámy, ktoré vyzerajú ako rámčeky filmu. Rozdiel je v tom, že rámce plochej sekvenčnej štruktúry sú usporiadané postupne vedľa seba. Skladaná sekvenčná štruktúra je usporiadaná postupne v hromade a javí sa podobne ako štruktúra prípadov [4].

5.3.5 ŠTRUKTÚRA UDALOSTÍ

Daná štruktúra vám umožní písať vysoko efektívny kód, ktorý čaká na udalosti skôr ako neefektívny kód, ktorý pravidelne kontroluje, či sa udalosti stali. Čo je vlastne udalosť? Udalosťou môže byť takmer všetko, čo sa v LabVIEW „stane“, príklad:

- Stlačí sa Boolean na prednom paneli.
- Hodnota číselného ovládača sa zmení.
- Kurzor myši vstúpi do okna VI.
- Stlačí sa tlačidlo.

Udalosti sa zvyčajne týkajú udalostí predného panela (GUI), ale môžu sa vzťahovať aj na ovládacie prvky na prednom panely ich zmeny hodnoty. Môžu dokonca byť definované vlastné, jedinečné udalosti. Aby sa zistilo, či bolo užívateľom stlačené tlačidlo (napr.: „STOP“) bez štruktúry udalostí, program VI by sa musel pravidelne dopytovať na jeho hodnotu v slučke „while“. So štruktúrou udalostí sa nemusí VI dopytovať, pretože bude „vedieť“, keď nastane táto udalosť. Štruktúra udalostí vyzerá podobne ako štruktúra prípadov, a to v tom, že má viacero rámcov pre prípad. Každý prípad štruktúry udalosti možno zaregistrovať na zvládnutie jednej alebo viacerých udalostí. Pri tejto štruktúre, táto bude čakať, až nastane udalosť, a potom vykoná presne jeden prípad od prípadu, ktorý je nakonfigurovaný na zvládnutie udalosti, ktorá nastala [4].

II. PRAKTICKÁ ČASŤ

6 MOŽNOSTI ACHITEKTÚRY PROGRAMU LABVIEW

Architektúra vo vývojových prostrediach teda i v prostrediu LabVIEW , tiež známy ako model dizajnu softvéru, je opakovane použiteľným riešením problému softvérového inžinierstva. Dizajnové vzory dávajú vývojárovi východiskový bod a môžu pomôcť zlepšiť efektívnosť, čitateľnosť, škálovateľnosť a udržiavateľnosť. Použitie vzoru dizajnu nám pomôže ľahko rozšíriť aplikáciu a znova použiť svoje vlastné vývojové úsilie, keď je potreba pridať nové funkcie. Po vytvorení kvalitnej architektúry môžu byť navrhnuté šablóny, ktoré sa dajú následne použiť aj pre budúce projekty.

Ďalej v tejto práci sú popísané niektoré užitočné návrhové vzory, ktoré môže vývojár použiť vo svojej architektúre aplikácií a ktoré komunita vývojárov v spoločnosti National inštrument neustále zdokonaľuje. V žiadnom prípade nejde o komplexný zoznam a žiadna architektúra by sa nemala považovať za „najlepšiu“, môže však byť užitočné začať s jedným z týchto vzorov alebo ich variáciou. Dobrá architektúra je taká, ktorá sa najlepšie hodí pre danú aplikáciu a môže byť kombináciou nižšie uvedených vzorov [1].

6.1 STAVOVÝ STROJ

Stavový stroj je jedným zo základných vzorových šablón, ktoré vývojári spoločnosti LabVIEW často používajú na rýchle vytváranie aplikácií. Architektúra stavového stroja môže byť použitá na implementáciu komplexných rozhodovacích algoritmov reprezentovaných stavovými diagramami alebo vývojovými diagramami. Stavové stroje sa používajú v aplikáciách, kde existujú rozlíšiteľné stavy. Každý stav môže viesť k jednému alebo viacerým stavom a môže tiež ukončiť procesný tok. Stavový stroj sa spolieha na vstup používateľa alebo výpočet v stave, aby určil, ktorý stav prejde na ďalší.

6.2 OBSLUHA UDALOSTÍ

Návrhový vzor obsluhy udalostí poskytuje výkonnú a efektívnu architektúru pre spracovanie interakcie používateľa s LabVIEW. Obslužný program udalostí slúži na zistenie, kedy sa vyskytnú udalosti, ako napríklad zmena hodnoty ovládacieho prvku, presunutie alebo kliknutie myšou, stlačenie klávesu atď. Štandardná šablóna obsluhy udalostí pozostáva zo štruktúry udalostí obsiahnutej v slučke „while“. Štruktúra udalosti by mala byť

nakonfigurovaná tak, aby mala jeden prípad pre každú kategóriu udalostí, ktorú chceme zistiť. Každý prípad udalosti obsahuje manipulačný kód, ktorý sa vykoná okamžite po výskyte udalosti.

6.3 MASTER/SLAVE

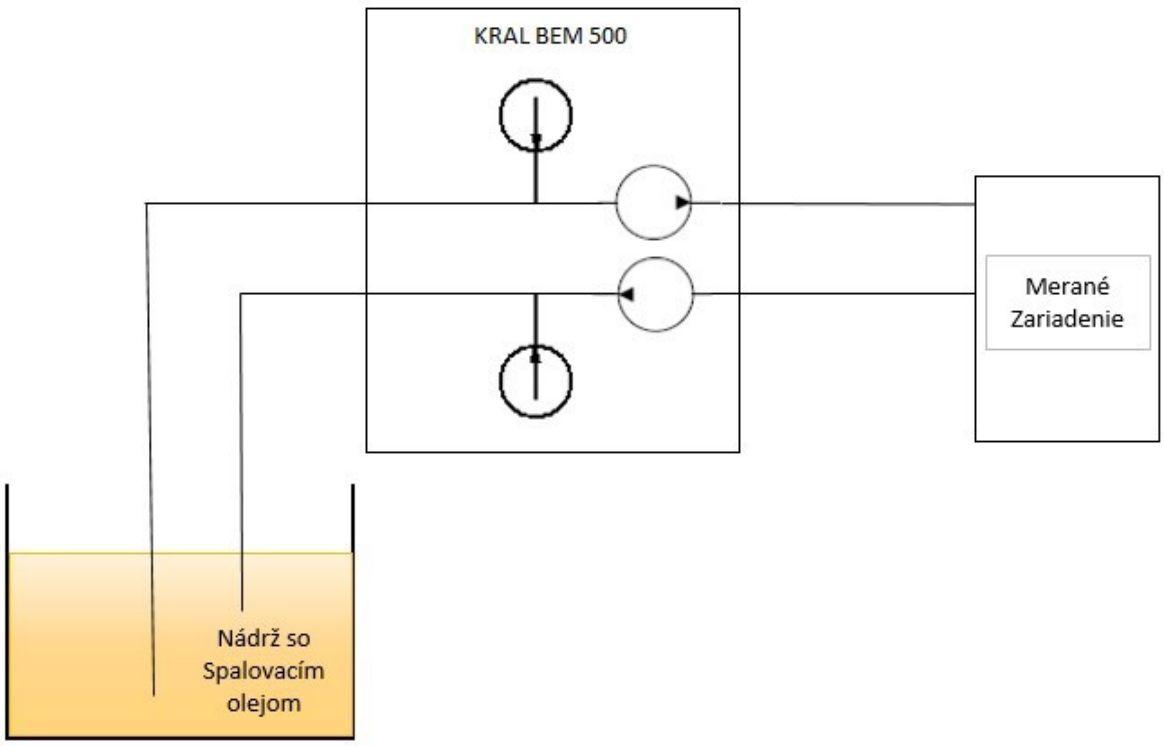
Návrhový vzor Master/Slave je ďalšou základnou architektúrou, ktorú používajú vývojári LabVIEW. Skladá sa z viacerých paralelných slučiek, kde každá zo slučiek môže vykonávať úlohy rôznym tempom. Z týchto paralelných slučiek, jedna slučka pôsobí ako master a ostatné fungujú ako slave. Master slučka riadi všetky slučky slave a komunikuje s nimi pomocou architektúr zasielania správ [2].

6.4 VÝROBCA / SPOTREBITEĽ

Na zdieľanie údajov medzi viacerými slučkami bežiacimi pri rôznych rýchlostiach sa používa návrhový vzor výrobcu/spotrebiteľa. Paralelné slučky vzoru výrobca/spotrebiteľa sa členia na dve kategórie; tie, ktoré produkujú údaje, a tie, ktoré produkujú získané údaje. Dátové fronty komunikujú údaje medzi slučkami v dizajnovom modeli výrobca/spotrebiteľ. Tieto fronty ponúkajú výhodu vyrovnávacej pamäte údajov medzi slučkami výrobcu a spotrebiteľa.

7 SCHÉMATICKÉ ZAPOJENIE MERANEJ SÚSTAVY

Pred navrhnutím softvérového riešenia pre opisovanú tému bolo nutné zrevidovať reálne zapojenie meranej sústavy. Daná aplikácia pozostáva z meraného zariadenia (v našom prípade spaľovacieho kondenzačného kotla so zmiešavacím horákom olej/vzduch), prietokomeru KRAL BEM 500 a nádrže pre spaľovací olej. Prietokomer KRAL BEM 500 obsahuje dva pomerové ultrazvukové prietokomery s integrovanými teplomermi pre každú vetvu. Merané zariadenie cirkuluje palivo a spotrebováva len nutné množstvo v zmiešavacom horáku pre dosiahnutie požadovaného výkonu. Prietokomer ja nakoniec spojený sériovou linkou (komunikácia pomocou protokolu MODBUS) s počítačom, na ktorom je aplikovaný nami navrhnutý softvér.

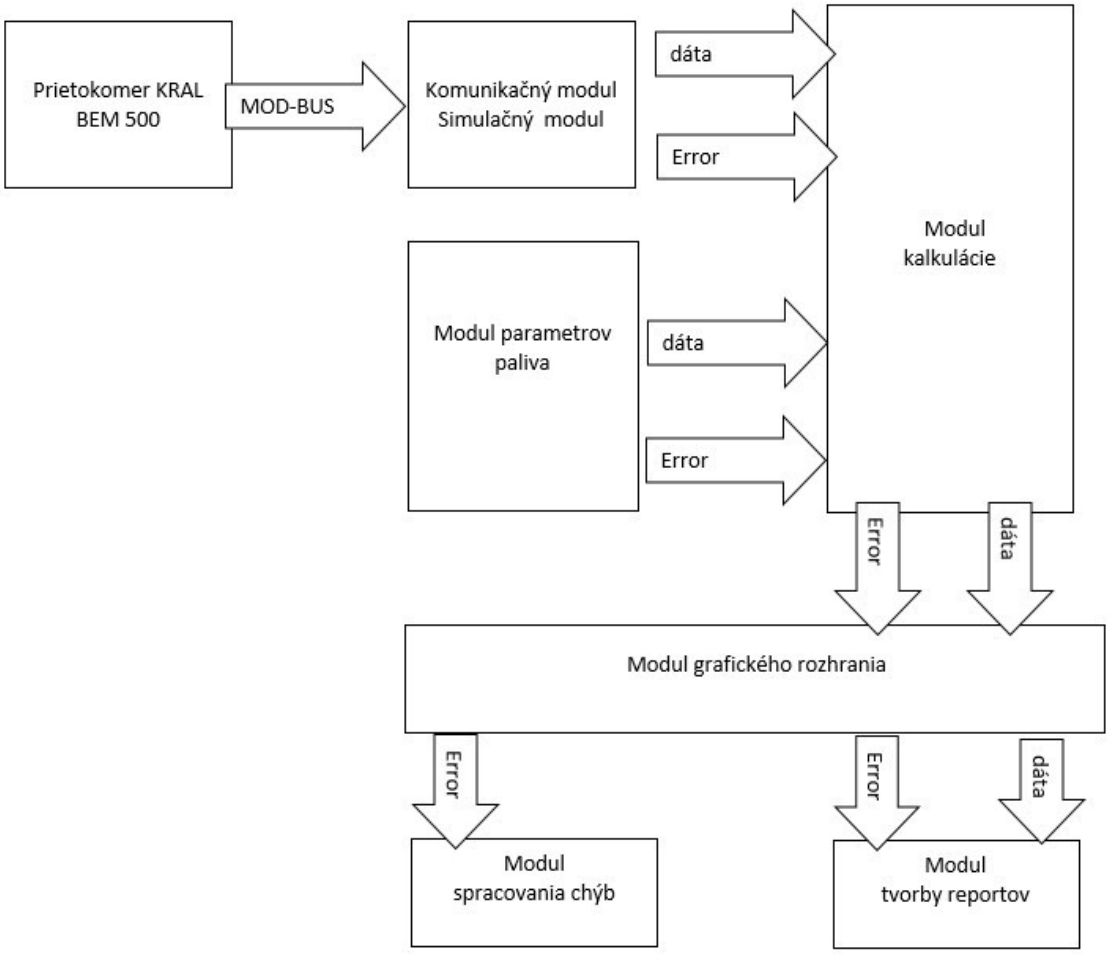


Obrázok 5 Schéma zapojenia meranej sústavy Zdroj: vlastný

8 VÝVOJOVÝ DIAGRAM PROGRAMU

Celkové riešenie danej témy vyžaduje riešenie čiastkových úloh, ktoré boli definované v moduloch daného softvéru. Daný tok informácií v konečnom riešení nie je možné nakresliť v podrobnom vývojovom diagrame preto bol zvolený blokový diagram, ktorý zob-

razuje dátový tok, funkcionalitu a nadväznosť navrhnutých modulov. Program je asynchrónne spúšťaný dvomi modulmi - komunikačného/simulačného a parametrov paliva, tieto sú popísané v separátnych kapitolách práce. Dáta putujú do časti softvéru pre kalkulácie, kde sú odovzdané grafickému rozhraniu pre vizualizáciu. Posledné časti programu sa zaoberajú spracovaním chýb a tvorbou reportu z nameraných údajov.

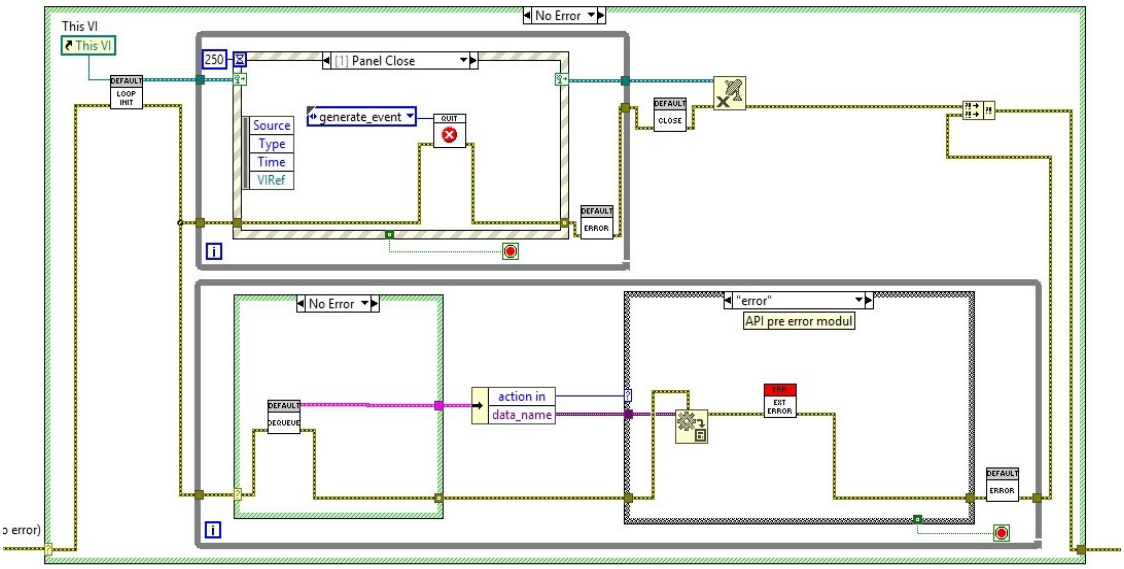


Obrázok 6 Blokový diagram programu Zdroj: vlastný

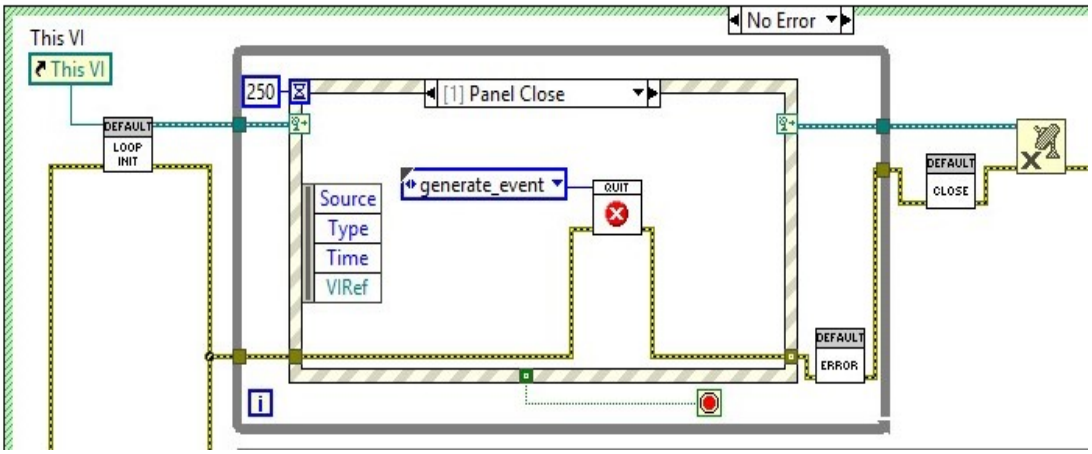
9 ZVOLENÁ ARCHITEKTÚRA PRE DANÚ TÉMU

Pre riešenie zadanej úlohy bol zvolený návrhový vzor obsluhy správ vo fronte (QMH). Vzor je kombináciou architektúry výrobcu/spotrebiteľa a obsluhy udalostí. Producentská slučka nazývaná slučka obsluhy udalostí (EHL) obsahuje štruktúru udalostí, ktorá odosiela správy do spotrebiteľskej slučky nazývanej slučka obsluhy správ (MHL). MHL prijíma a spracováva správy. Správa je vyžadovaná, keď je spustená udalosť. QMH môže

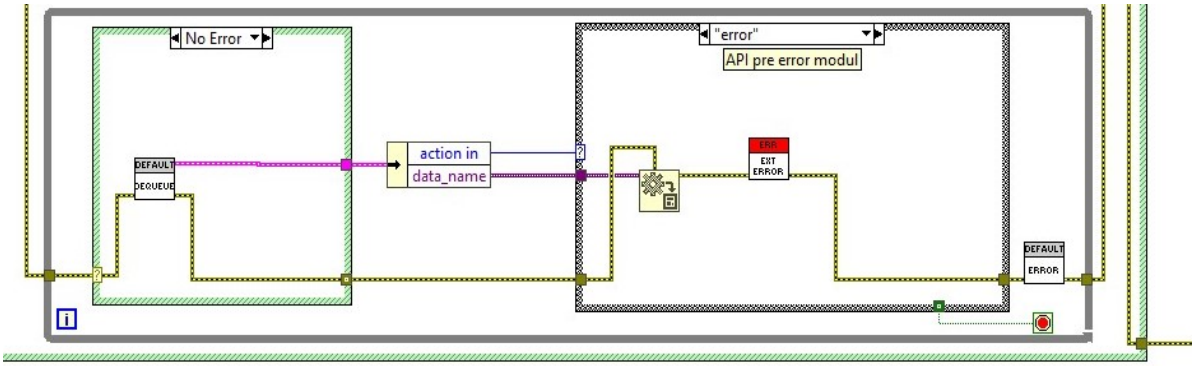
byť tiež navrhnutý tak, aby poskytoval spätnú väzbu od spotrebiteľa výrobcovi pomocou užívateľských udalostí [3].



Obrázok 7.1 Základná Architektúra programu kompletná. Zdroj vlastný



Obrázok 7.2 Základná Architektúra programu – detail EHL. Zdroj vlastný



Obrázok 7.3 Základná Architektúra programu –detail MHL. Zdroj vlastný

9.1 SLUČKA NA SPRACOVANIE UDALOSTÍ (EHL)

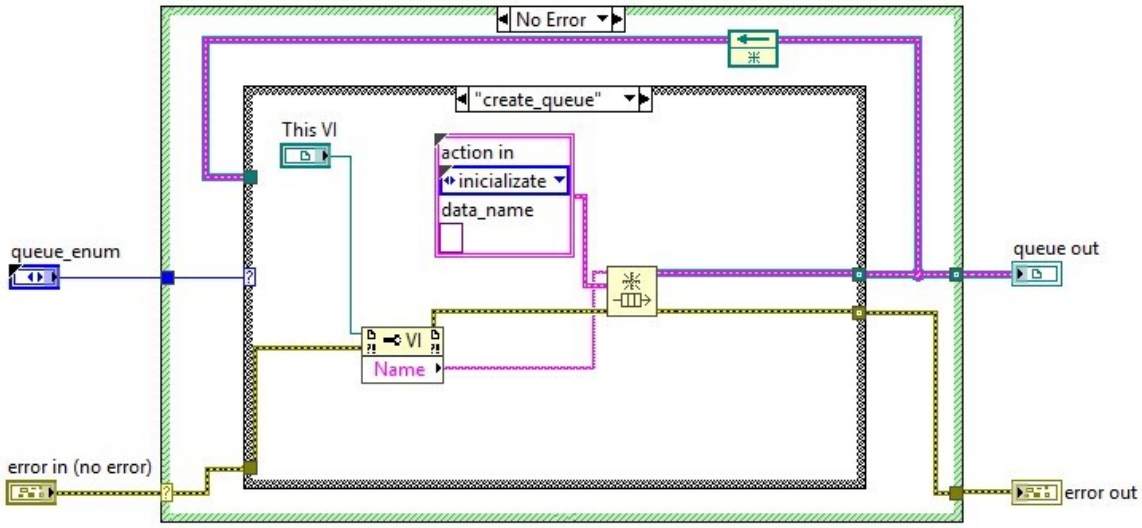
Hlavným účelom EHL je reagovať na udalosti vyvolané používateľom prostredníctvom interakcií s ovládacími prvkami na prednom paneli. Udalosti však môžu pochádzať z iných zdrojov pomocou udalostí používateľov. Keď je udalosť spustená alebo vygenerovaná, vykoná sa diagram štruktúry udalosti zodpovedajúci tejto udalosti. Zodpovedajúca správa sa potom vytvorí a požiada o zaslanie do MHL

9.2 SLUČKA OBSLUHY SPRÁV (MHL)

Hlavným účelom MHL je vykonávať podnety v prichádzajúcich správach. MHL čaká, kým sa v zásobníku neobjaví správa a následne ju spracuje. Pri vyššie uvedenej štruktúre správy je správa neviazaná, aby sa zistilo, či sa jedná o názov správy alebo príkaz. Štruktúra prípadu potom vykoná diagram zodpovedajúci správe. V diagrame sú údaje prevedené zo skupiny („Cluster“) a použité pri vykonávaní prípadu.

9.3 SPRÁVY (EHL PRE MHL)

Správy môžu byť štruktúrované mnohými spôsobmi, ale zvyčajne pozostávajú z dvoch častí: Názov správy alebo príkaz a dáta. V mojom prípade som zvolil správcu fronty, ktorý radí požiadavky do zásobníka na základe udalostí prevedených v slučke EHL.

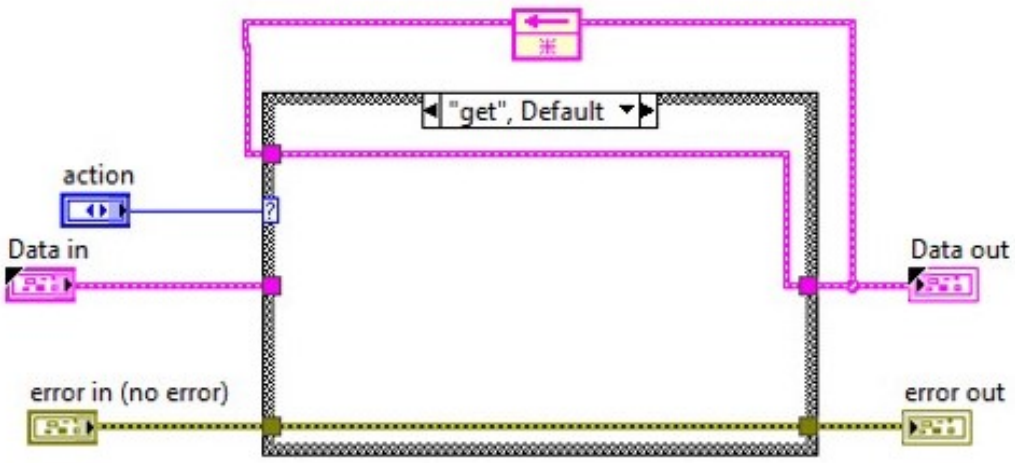


Obrázok 8 Správca zásobníka. Zdroj: vlastný

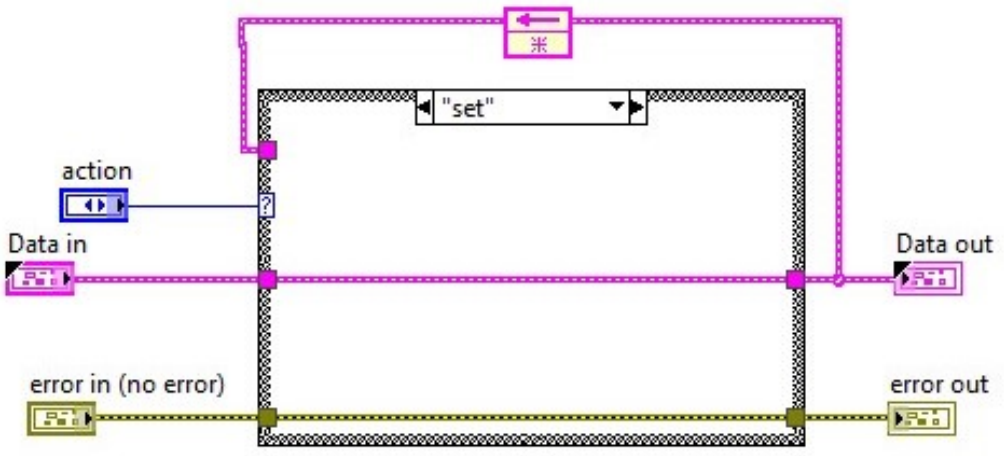
Pre každý modul je vytvorený separátny zásobník. Základné štruktúry sú, vytvorenie zásobníka, čítanie zo zásobníka a zničenie zásobníka. Vďaka správcovi zásobníka prebieha komunikácia medzi slučkami.

10 SOFTVÉROVÉ RIEŠENIE MODULOV

Zvolená architektúra sa implementovala pre každý samostatný modul s modifikáciami tak, aby najlepšie plnili účel. Pre zdieľanie zmeny stavov medzi jednotlivými premennými boli využité globálne premenné. Avšak pre komunikáciu transfer získaných dát medzi jednotlivými modulmi bol vytvorený dátový správca. Dátový správca je riešený podobne ako komunikácia medzi EHL a MHL, len dáta nie sú uložené v zásobníku, ale v preddefinovanom zhluku („Cluster“). Štruktúra príkazov obsahuje metódy získaj (get) a nastav (set). Dáta sú uložené vďaka integrovanej funkcii prostredia LabVIEW, konkrétne pomocou spätnoväzobného uzla („feedback node“). Táto funkcia ukladá dáta medzi iteráciami slučky. Pamäťová náročnosť je zrovnateľná s globálnymi premennými, ale poskytuje výhodu v budúcom odfiltrovaní nepotrebných šumov a odladení snímaných informácií. I keď to nebola prvotná požiadavka definovanej témy, považujem za výhodné riešiť transfer dát týmto spôsobom.



Obrázok 9.1 Dátový správca príkaz get. Zdroj: vlastný



Obrázok 9.2 Dátový správca príkaz set. Zdroj: vlastný

10.1 KOMUNIKÁCIA / SIMULÁCIA

Časť programu komunikácia/ simulácia je jeden z hlavných modulov zabezpečujúci spojenie s prietokomerom, zber dát a tiež simuláciu signálov pre potrebu odladenia softvéru v režime off-line. Daná časť programu je z hľadiska operačného času najviac zaťažovaná. Architektúra bola modifikovaná o tretiu slučku, v ktorej sa kvázi paralelne vykonáva neustále odčítanie hodnôt zo sériovej komunikácie alebo zo simulovaných hodnôt.

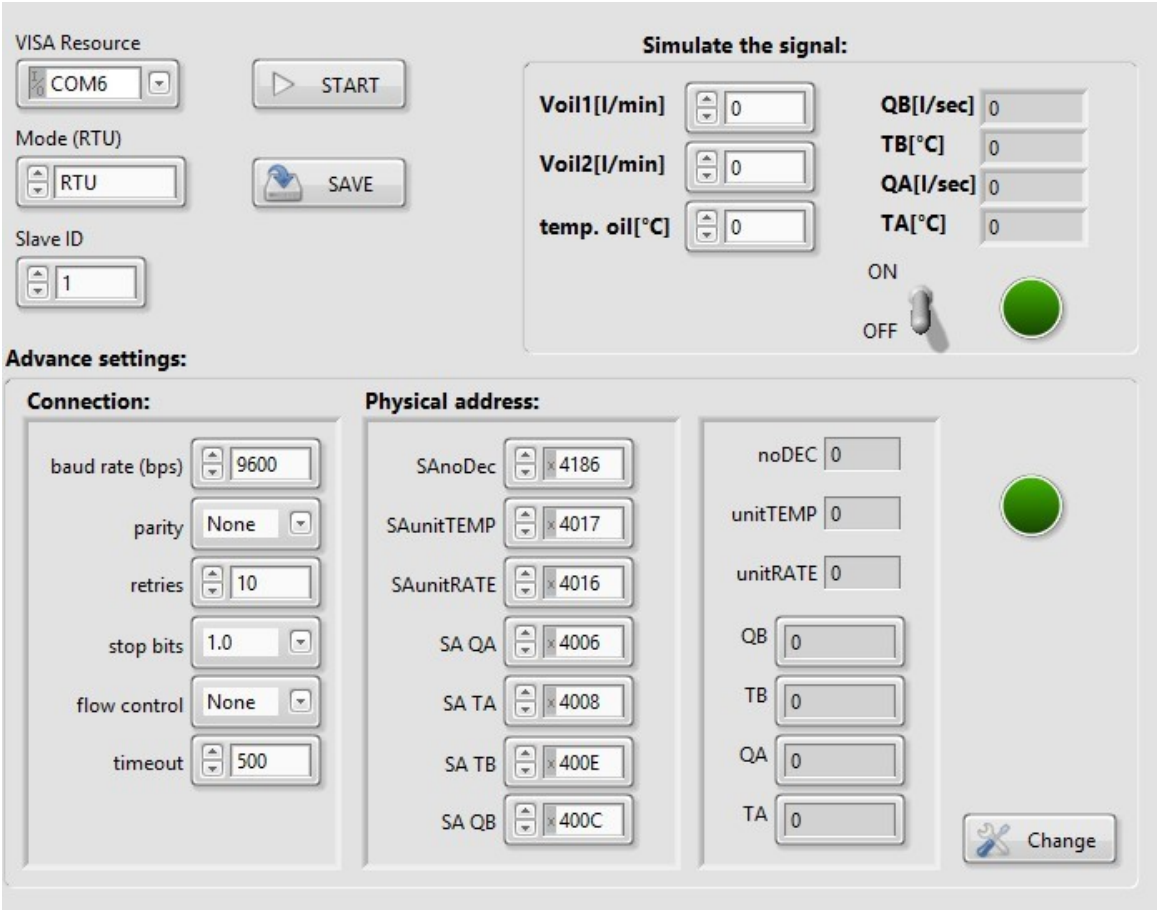
10.1.1 GRAFICKÉ ROZHRANIE MODULU

Sekcia komunikácie a simulácie sa asynchrónne aktivuje po spustení programu, preto je dôležité vytvoriť i grafické rozhranie (GUI) pre užívateľa, ktorý bude musieť nastaviť základné parametre komunikácie, či požadované hodnoty signálu simulácie. Grafické rozhranie pozostáva z 3 hlavných sekcií.

Ľavá horná časť obsahuje nastavenie periférie pre sériový port (VISA Resource), mód komunikácie (Mode (RTU)) a identifikačné číslo zariadenia s ktorým sa na sériovej linke naviaže spojenie. Ďalej sú k dispozícii dve tlačidlá. Tlačidlo „START“ spustí sériovú komunikáciu s prietokomerom. Tlačidlo „SAVE“ uloží definované parametre sériovej komunikácie do štandardného „config_Comunication.ini“ súboru (.ini súbor popísaný v teoretickej časti práce), toto uľahčuje prácu po znovu spustení programu.

Pravá horná časť zabezpečuje rozhranie pre užívateľa v prípade potreby simulácie signálu. Požadované vstupy pre simuláciu výstupného prietoku QA,QB a teplôt TB,TA. Ďalej prepínač ON/OFF ktorý aktivuje simuláciu.

Spodná časť rozhrania rieši formu spojenia pomocou protokolu MODBUS. Definícia spojenia podľa manuálu výrobcu prietokomeru (baud rate = 9600, parity = 0, flow controle = 0, retries bits = 10, stop bit = 10, timeout = 500). Fyzické adresy na ktorých sa nachádzajú potrebné informácie k odčítaniu (address UnitRate = 16406, address UnitTemp = 16407, address NoDecimal = 16774, address TA = 16392, address QA = 16390, address TB = 16398, address QB = 16396). V neposlednom rade tlačidlo „Change“, ktoré slúži na prerušenie komunikácie a následnú modifikáciu parametrov.



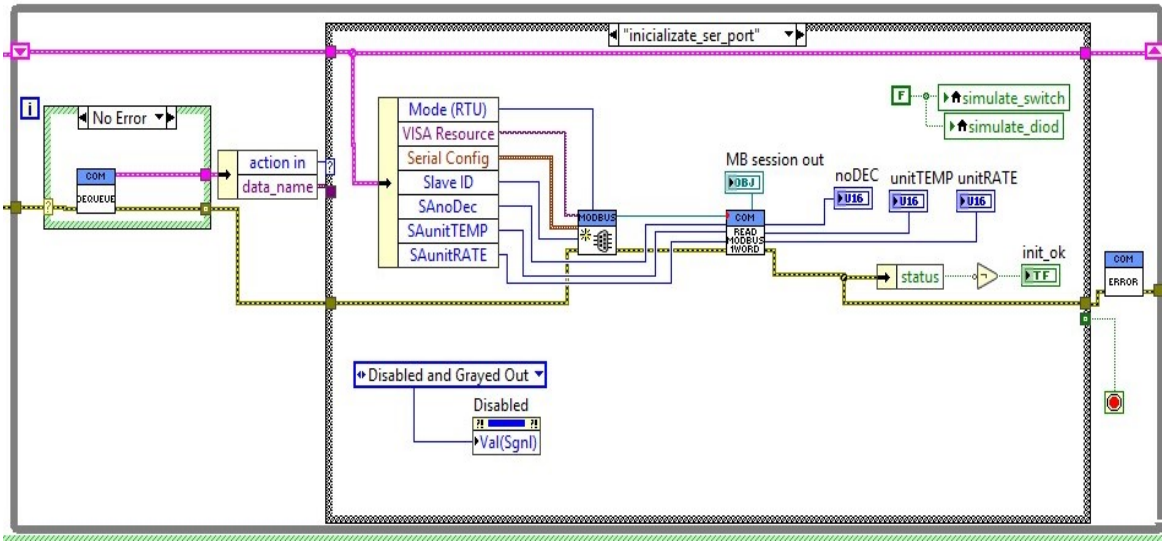
Obrázok 10 GUI Komunikácie. Zdroj: vlastný

10.1.2OBJEKTOVÉ ROZHRANIE MODULU

Modul zahajuje činnosť prvou správou v MHL, načítavanie dát pre správu spojenia MODBUS zo súboru „config_Comunication.ini“. Túto akciu spravuje pod program „Get Data. INIT“, ktorý preverí či dokument existuje alebo ihneď vytvorí prázdny. Nasleduje inicializácia vstupných a výstupných premenných pre Grafické rozhranie modulu. Po úspešnej inicializácii program počúva a čaká na interakciu užívateľa skrz EHL.

10.1.3 INTERAKCIA „START“

Po doručení správy zo zásobníka MHL prevedie softvér prvú komunikáciu so sériovým portom pomocou obsiahnutej knižnice v LabVIEW pre správu a komunikáciu protokolom MODBUS. V tomto kroku sa pri prvej komunikáciu načítajú i prvé údaje z fyzických adries pripojeného prietokomera (jednotky teploty, jednotky prietoku, počet desatinných miest). Pred ukončením inštrukcií sa kontroluje chybovosť spojenia a v prípade naviazania komunikácie sa lokálnou premenou „init_ok“ aktivuje kontinuálne čítanie dát v tretej paralelnej slučke tohto modulu. Taktiež sa deaktivuje simulácia, ak bola už spustená a zneprístupnia sa vstupné polia pre nastavenie komunikácie u grafického rozhrania.



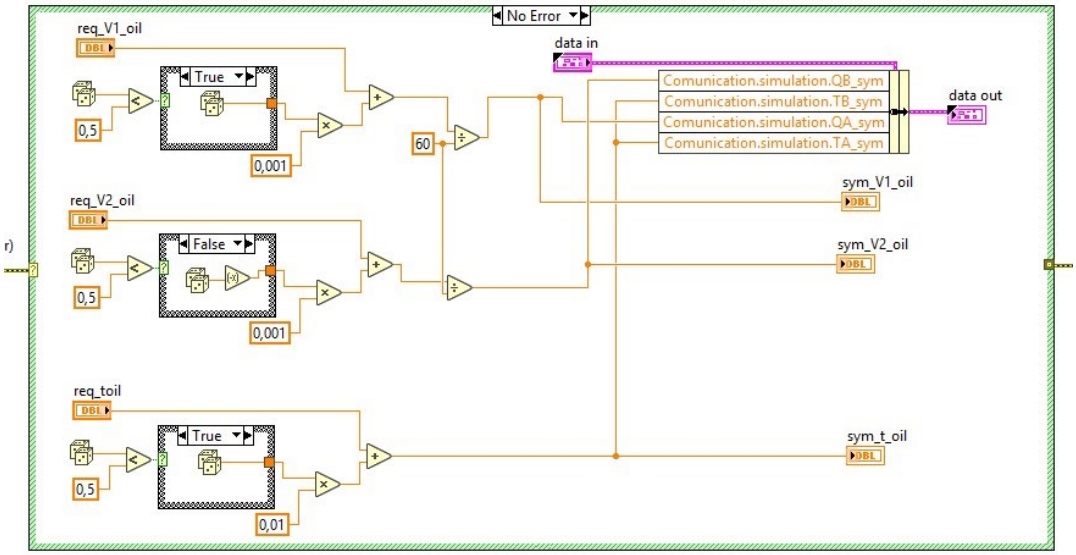
Obrázok 11 Interakcia „Start“. Zdroj: vlastný

10.1.4INTERAKCIA „CHANGE“

Daná správa pre program znamená len odpojiť sériovú komunikáciu pomocou knižnice MODBUS, aktivovať polia grafického rozhrania a inicializovať všetky stavové lokálne premenné pre tretiu paralelnú slučku do stavu NEPRAVDA (FALSE).

10.1.5 INTERAKCIA „ON/OFF“

Prepínačom ON/OFF grafického rozhrania sa z EHL vygeneruje správa pre reaktiváciu sériovej komunikácie. Lokálna premenná „simulate_diod“ spustí podprogram v simulácií signálu v tretej paralelnej slučke. Podprogram na základe zadaných požadovaných hodnôt generuje signál s náhodným kolísaním stotín a tisícin. Program generuje 32-bitové desatinné číslo v intervalu <0,1>. Pre čísla menšie ako 0,5 generuje ďalšie kladné a pre väčšie záporné v rádoch stotín a desaťtisícin, ktoré sú pripočítavané ku požadovanej hodnote. Týmto vzniká dojem náhodnej simulácie signálu.



Obrázok 12 Simulácia signálov. Zdroj: vlastný

10.1.6 INTERAKCIA „SAVE“

MHL dostane správu pre uloženie dát z grafického rozhrania do už zinicializovaného súboru „config_Comunication.ini“. Táto časť sa vykonáva podprogramom „SAVE DA-

TA.INIT“. Ten pri každej inicializácii požiada užívateľa o potvrdenie či chce naozaj zmeniť súbor a kde sa dáta nachádzajú. Následne pomocou kľúčov (názvy premenných) a ich hodnôt súbor aktualizuje. Pre správu sa využíva dostupná knižnica pre „.ini“ súbore od LabVIEW.

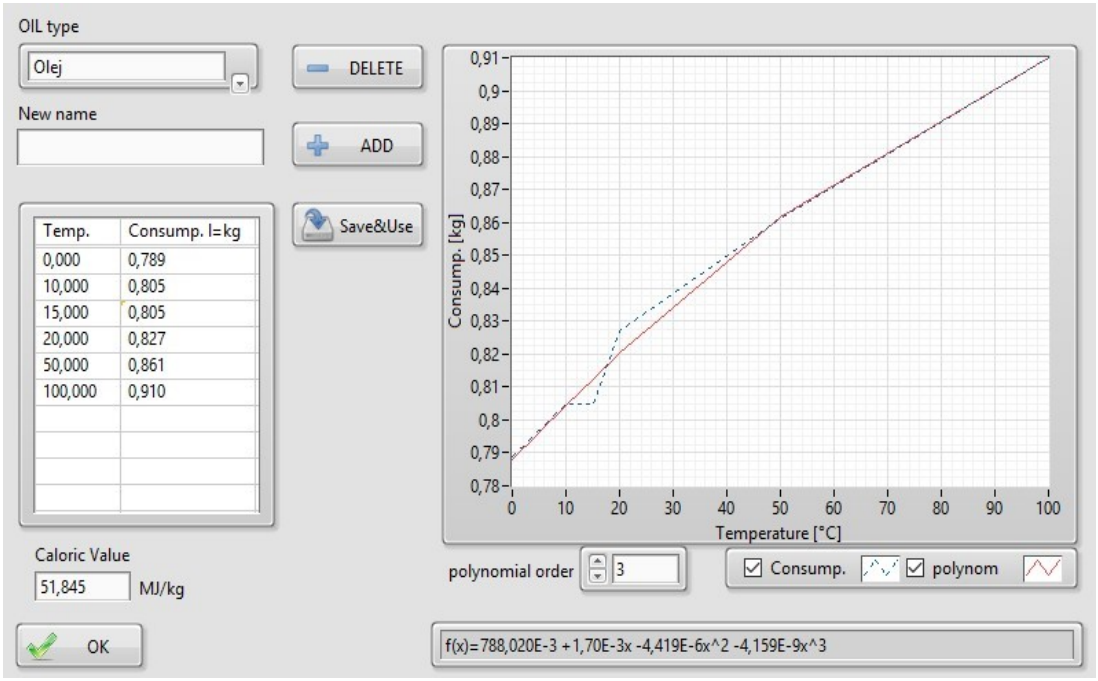
10.2 PARAMETRE PALIVA

Časť programu „Parametre paliva“ je modul zabezpečujúci prepočet a konfiguráciu použitého paliva. Pretože kalkulácia vyžaduje hmotnostný prietok a z fyzického prevedenia prietokomeru odčítame objemový, je treba na základe čistoty prepočítať výslednú hmotnosť. Požiadavka z teoretického rozboru je i uplatniť teplotu pretekajúceho paliva pre presnejší prepočet jeho hmotnosti. Modul bol preto navrhnutý tak, aby bolo možné ukladať viac typov paliva a ním prislúchajúcimi tabuľkami hmotnostných vzťahov. Zo vzťahov je možné vybrať stupeň polynómu tak aby najlepšie odpovedal hodnotám. Tento krok je dôležitý, nakoľko práve funkcia zvoleného polynómu je funkčná hodnota pre nasledujúci prepočet. Neoddeliteľnou informáciou je i výhrevnosť daného paliva, ktorá je definovaná buď z rozboru daného paliva, alebo v prislúchajúcich normách. Pre realizáciu nebola nutné nami zvolenú architektúru nijako modifikovať a len sa nakonfigurovali jednotlivé časti softvéru.

10.2.1 GRAFICKÉ ROZHRANIE MODULU

Rovnako ako sekcia komunikácie a simulácie sa asynchrónne po spustení programu aktivuje i daný modul, preto je taktiež dôležité nakonfigurovať relevantné rozhranie pre užívateľa. Po spustení, panel obsahuje rolovacie menu „Oil type“ v ľavom hornom rohu pre možnosť voľby paliva a taktiež vstupné pole pre reťazec znakov “New name“. Tento slúži na pomenovanie nového paliva ak je to nutné. V rovnakej časti nájdeme i dve tlačidlá pre správu palív, teda mazanie „DELETE“ a pridanie „ADD“ a tretie tlačidlo slúžiace k uloženiu nastavených dát „Save&Use“. V ľavom strede sa nachádza tabuľka 2x10, táto vyžaduje implementáciu vzťahov medzi teplotou a hmotnosťou v kg pre jeden liter paliva. V pravej časti sa nachádza graf pre zobrazenie vzťahov z opísanej tabuľky. Vzťah sa zobrazuje modrou prerušovanou čiarou a jeho opísaný polynóm červenou plnou. Pod grafom je vstupné pole pre modifikáciu stupňa polynómu „polynominal order“ a taktiež výstupné

pole reťazca daného polynómu. Posledná ľavá dolná časť obsahuje vstupné údaje výhrevnosti paliva „Caloric Value“ a taktiež tlačidlo pre potvrdenie a postup k ďalším častiam kompletného programu.



Obrázok 13 GUI Parametrov paliva. Zdroj: vlastný

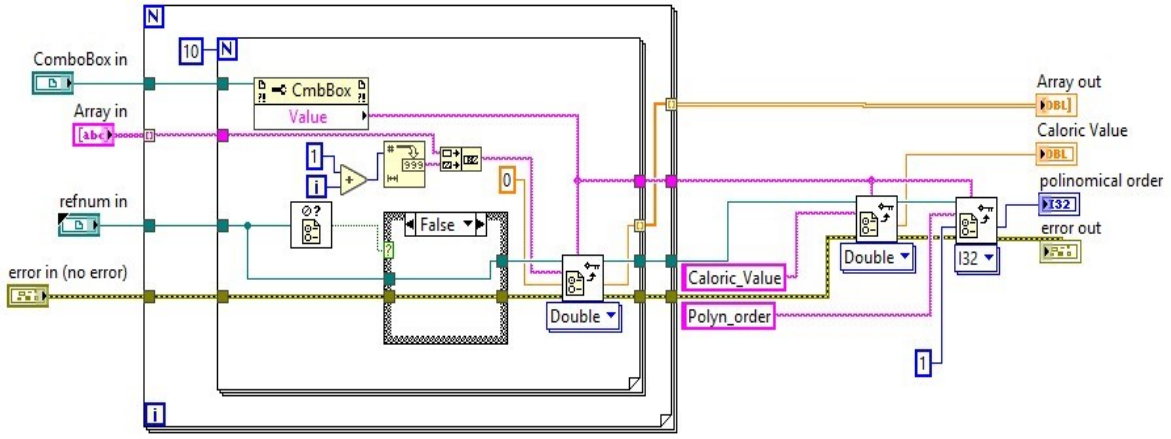
10.2.2 OBJEKTOVÉ ROZHRANIE MODULU

Modul vždy začína inicializáciou seba sama a modul paliva pri tejto činnosti vytvorí pre užívateľa prvé palivo s názvom „OIL-1“. Inicializácia vždy preverí existenciu súboru „oil_data.ini“ a načíta z neho dáta do dynamického poľa. Ak je veľkosť poľa nulová, vloží predom definované hodnoty z programu s uvedeným menom a do súboru ich uloží. Ak nie je nulová, načíta dáta pre rolovacie menu. Po úspešnej inicializácii program počúva a čaká na interakciu užívateľa skrz EHL.

10.2.3 INTERAKCIA „LOAD“

Načítanie dát zo súboru „oil_data.ini“ sa uskutoční pri volaní funkcie z EHL pri každej zmene typu paliva v roletovom menu. V MHL po načítaní správy, dáta spravuje podprogram „Load Data3“. Podprogram spracuje dvojrozmerné pole tak, aby načítal 10 hodnôt pre teplotu a 10 hodnôt pre vzťah vyplnený v tabuľke grafického rozhrania. Hodnoty

sú vždy uložené pod sekciou typu paliva z roletového menu. Ďalej načíta z danej sekcie i hodnotu tepelnej výhrevnosti a stupeň polynómu.



Obrázok 14 Podprogram load data. Zdroj: vlastný

10.2.4 INTERAKCIA „SAVE“

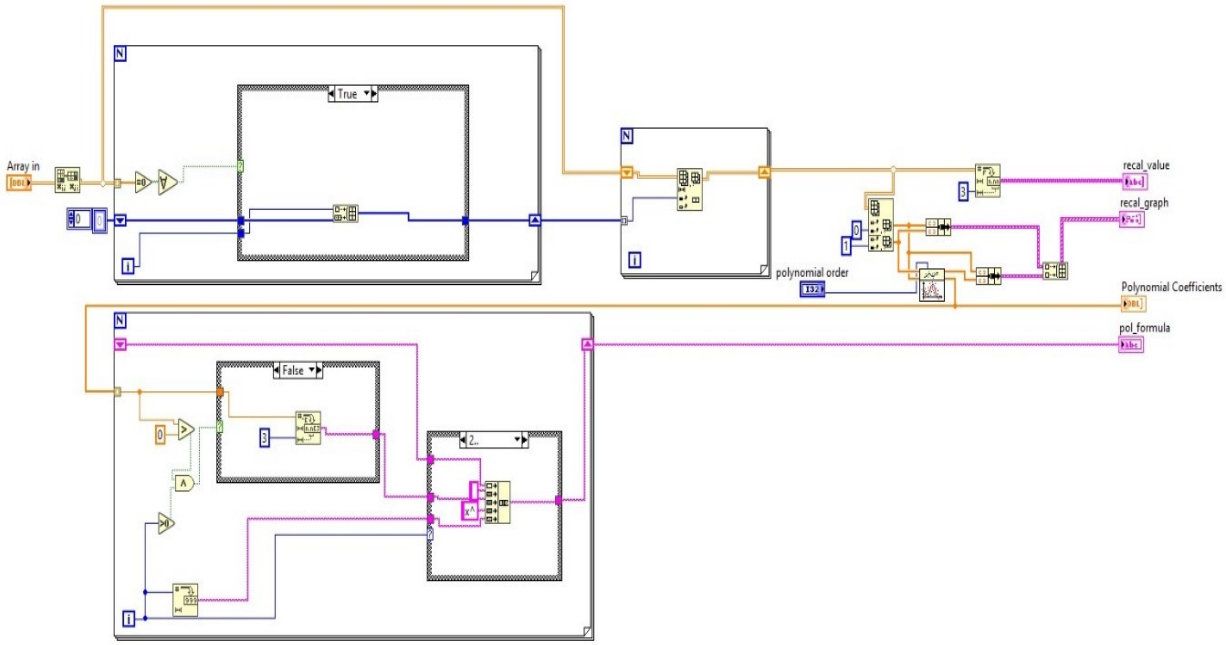
Pri udalosti aktivácie tlačila „Save&Use“ sa typovo prekonfiguruje tabuľka vzťahov paliva na pole 32 bitového desatinných čísiel a transponovaná sa podprogramom „SAVE TABLE“ uloží do súboru „oil_data.ini“. Reverzným spôsobom k načítaní dát ich pomocou dvoch for-cyklov uloží. Správu zabezpečuje knižnica „.ini“ súboru.

10.2.5 INTERAKCIA „ADD“ A „DELETE“

Udalosti slúžiace na správu palív. LabVIEW vývojové prostredie pristupuje ku každej premennej ako k objektu, ktorý má vlastné triedy a vlastnosti, ktoré sa môžu meniť a nastavovať. Táto výhoda bola využitá pri konfigurácii listu palív. Do roletového menu bolo cez triedu „value“ teda (hodnota) vložené pole palív, ktoré sa načítali z „oil_data.ini“. Ak teda je treba pridať ďalšie palivo po reakcii na tlačidlo „ADD“, modul len skontroluje, či je meno nového paliva dostatočne dlhé (minimálne 3 znaky) a či rovnaký názov náhodou už neexistuje a nakoniec cez danú triedu vloží novú sekciu (meno paliva). Odstránenie paliva sa však prevádza po aktivácii tlačidla „DELETE“ a to tak, že pomocou knižnice na správu „.ini“ súboru sa zvolená sekcia zo súboru „oil_data.ini“ vymaže a znovu načíta upravený súbor pomocou podprogramu „Load Data3“.

10.2.6INTERAKCIA „KALKULÁCIE POLYNOMU“

Pri každej zmene stupňa polynomickej rovnice sa aktivuje podprogram v MHL s názvom „POL.CREATION“. Vstupom tejto časti programu je dvojrozmerné pole dát o teplote k vzťahom vybraného paliva. Pole sa ďalej spracuje v niekoľkých krokoch. V prvom kroku sa pole skontroluje a vygeneruje nové (pomocné) s informáciami na ktorých pozíciách sa nenachádzajú údaje. V druhom kroku sa z poľa dát odstránia pozície ktoré neobsahujú dáta, pomocou nami vygenerovaného poľa z prvého kroku. Oba kroky sú dôležité pre správne vyčíslenie finálnej polynomickej rovnice. Bez tejto korekcie by rovnica počítala s prázdnymi miestami v poli, ako s nulovými hodnotami.



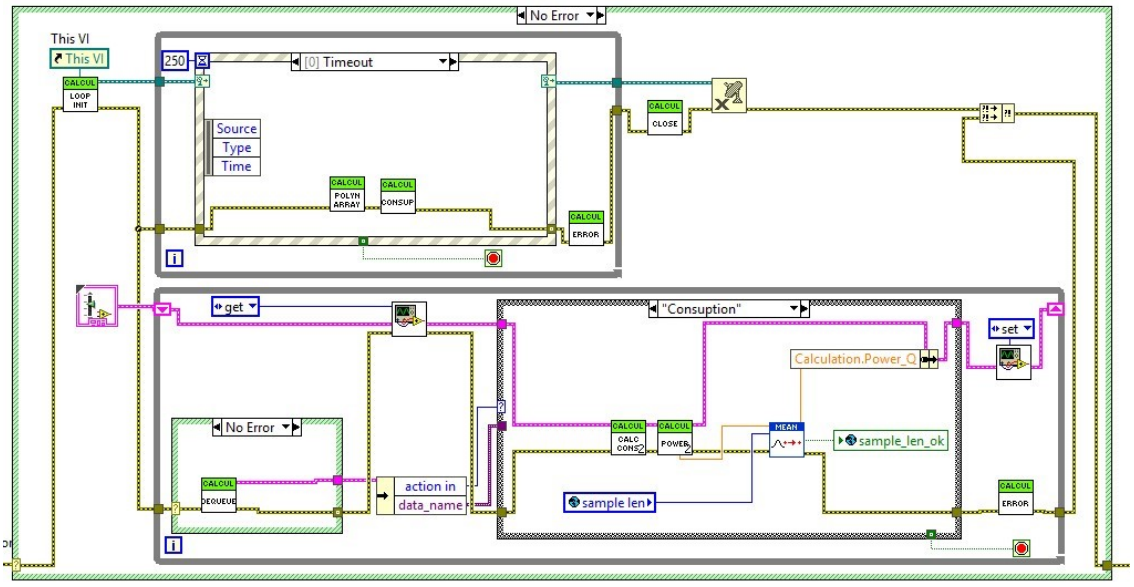
Obrázok 15 Kalkulácia polynomickej rovnice. Zdroj: vlastný

V treťom kroku pomocou matematickej knižnice dodávanej v základnom balíčku vývojového prostredia LabVIEW sa vygenerujú koeficienty pre polynomickú rovnicu. Spojením údajov z orezaného poľa a výsledného priebehu po uplatnení funkcie sa vytvoria dáta pre naše zobrazovacie prostredie.

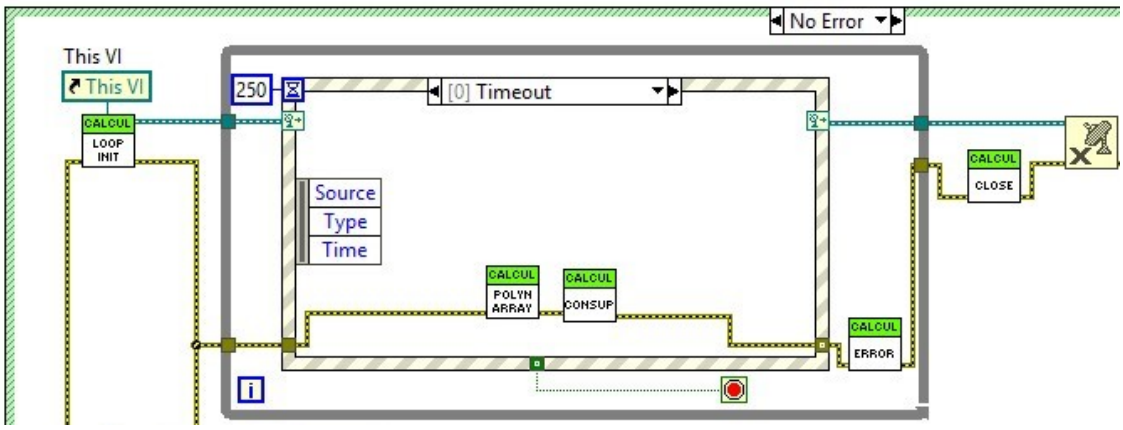
V štvrtom a poslednom kroku sa pomocou for-cyklu a spájania reťazcov zostaví matematická interpretácia polynomickej rovnice.

10.3 KALKULÁCIA

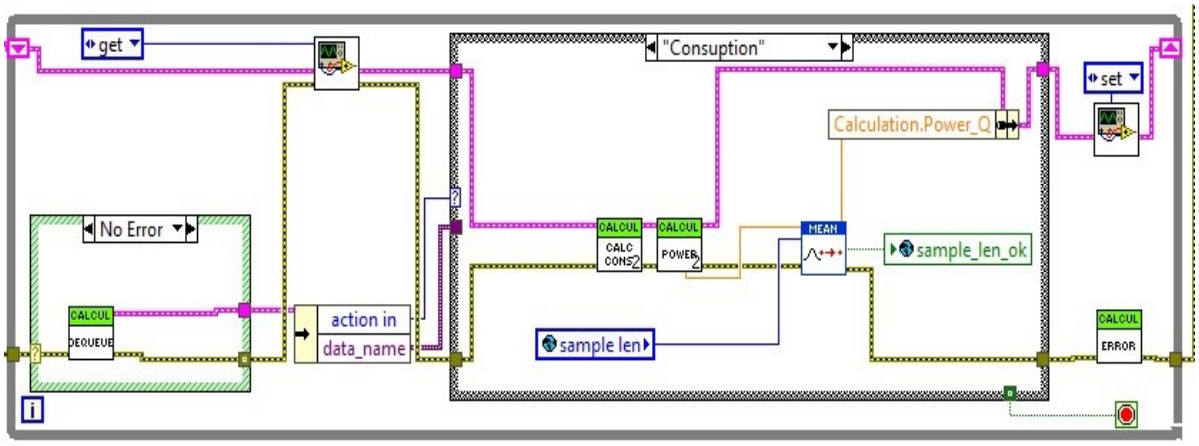
V moduly kalkulácie sa spájajú informačné toky z predchádzajúcich modulov a to z komunikačného a z modulu pre parametre paliva. Daná časť softvéru nepotrebuje interakciu s užívateľom, preto ani Grafické rozhranie nebolo nutné. No i napriek tomu bola zvolená rovnaká architektúra manažéra udalostí so spracovaním správ. Architektúra okrem štandardných udalostí pre ukončenie a zatvorenie okna obsahuje jednu špecifickú udalosť „medzičas“ (Timeout). V každom medzičase sa vygeneruje udalosť zaradená do zásobníka pre spracovanie polyfonickej hodnoty a spotreby. Modul zahajuje činnosť inicializáciou sama seba, kde sa načítajú prvé informácie z komunikačného modulu a to informácie o jednotkách meraných veličín. Ďalej v rovnakej fáze inicializácie modul prvý raz komunikuje s modulom parametrov paliva a vypočíta koeficient prepočtu objemového prietoku na hmotnostný. Softvér pracuje s podprogramami jednotky („UNIT“), aplikácia polynómu („POLYN APLIC2“), prepočet spotreby („CALC CONS2“), priemer („MEAN“) a príkon („POWER“).



Obrázok 16.1 Modul kalkulácie kompletný. Zdroj: vlastný



Obrázok 16.2 Modul kalkulácie detail EHL. Zdroj: vlastný



Obrázok 16.3 Modul kalkulácie detail MHL. Zdroj: vlastný

10.3.1 PODPROGRAM JEDNOTKY („UNIT“)

Podprogram je jednoduchý rozhodovací algoritmus, ktorý uvažuje za prvé stav simulácie. V tom prípade pridelí jednotku prietoku $l.sec^{-1}$ a teplote °C. Pri stave komunikácie pridelí jednotky na základe prislúchajúcich informácií z načítanej fyzickej adresy prietokomeru. Odpovedajúce jednotky sú pevne nastavené v zdrojovom kóde modulu, nakoľko softvér je navrhnutý pre jeden konkrétny prietokomer.

10.3.2 PODPROGRAM APLIKÁCIA POLYNÓMU („POLYN APLIC2“)

Časť programu ktorá vypočíta koeficient prepočtu prietoku objemového na hmotnostný. Výpočet sa spracováva na základe nameranej respektíve simulovanej teploty oleja

a zvolenej polyfonickej rovnice z modulu parametrov paliva. Podprogram uvažuje i prepočet °F na °C v prípade, že je takto prietokomer nastavený.

10.3.3 PODPROGRAM PREPOČET SPOTREBY („CALC CONS2“)

V danej časti sa spracovávajú informácie z predchádzajúceho podprogramu aplikácie polynómu a reálneho, či simulovaného prietoku. Jednoduchým matematickým úkonom násobenia koeficientu spotreby a prietoku. Výsledkom je hmotnostný prietok v kilogramoch za sekundu, Dané jednotky sú základné jednotky popísané pre výpočet príkonu olejových kotlov v norme EN 303-1. Ako v predošlom prípade, uvažujú sa všetky možné jednotky, ktoré sú nastaviteľné v danom prietokomeru. Podľa informácie z fyzickej adresy o jednotkách prietoku, sa zvolí odpovedajúci prepočet na litre za sekundu. Konečný meraný prietok je len absolútna hodnota rozdielu vstupného a výstupného pretoku „QA“ a „QB“.

10.3.4 PODPROGRAM PRÍKON („POWER“)

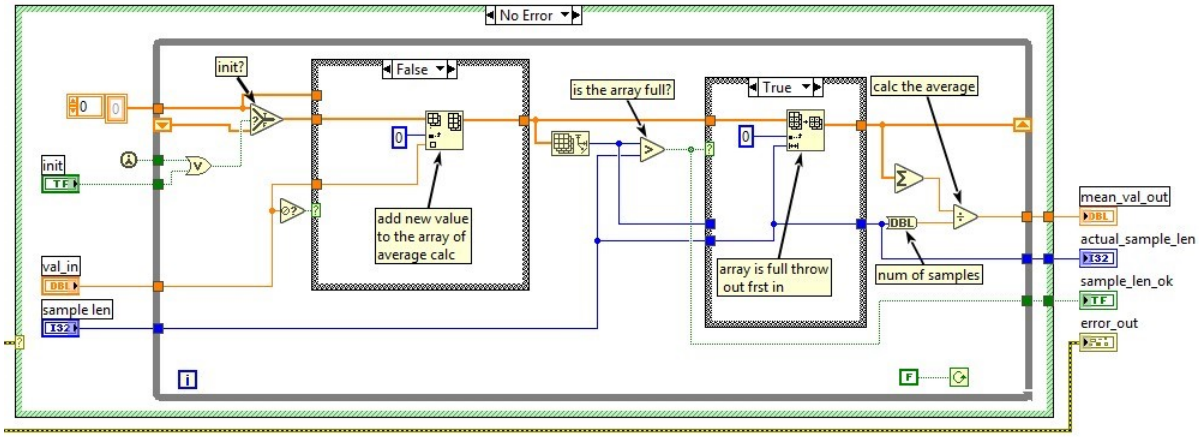
Daná časť nie je ani podprogram ako len funkcia vracajúca informáciu konečného príkonu meraného zariadenia vo Wattoch. Argumentmi funkcie je výhrevnosť paliva a hmotnostný prietok z predchádzajúceho podprogramu PREPOČET SPOTREBY („CALC CONS2“).

10.3.5 PODPROGRAM PRIEMER („MEAN“)

Pre výpočet priemeru existuje niekoľko štandardných funkcií v matematickej knižnici LabVIEW. Pre aplikáciu plávajúceho priemeru, teda priemeru kde na základe zvolených iterácií sa priemeruje meraná hodnota vždy z posledných odčítaných hodnôt, funkcia neexistuje. Preto bola daná časť naprogramovaná nami. Dôvodom implementácie tejto časti programu je možnosť priemerovať výslednú hodnotu príkonu.

Podprogram pracuje s posuvným registrom. Posuvný register je dynamicky alokovaná pamäť, na ktorej sa prepisujú informácie pri každej iterácii slučky. V posuvnom registri je uložené jednorozmerné pole hodnôt, teda vektor o veľkosti počtu priemerovaných hod-

nôt („sample len“). Slučka sa zopakuje vždy pri novej vstupnej hodnote („val_in“). Hodnota sa ukladá do poľa a porovnáva s požadovanou dĺžkou. Ak pole nie je plné, hodnoty v poli sa spriemerujú. Ak pole plné je, vymaže sa prvá, teda najstaršia hodnota a zvyšné hodnoty sa spriemerujú. Výstup je teda priemerná hodnota („mean_val“) a informácia dosiahnutia plného požadovaného počtu priemerovaných hodnôt („sample_len_ok“).



Obrázok 17 Podprogram priemeru. Zdroj: vlastný

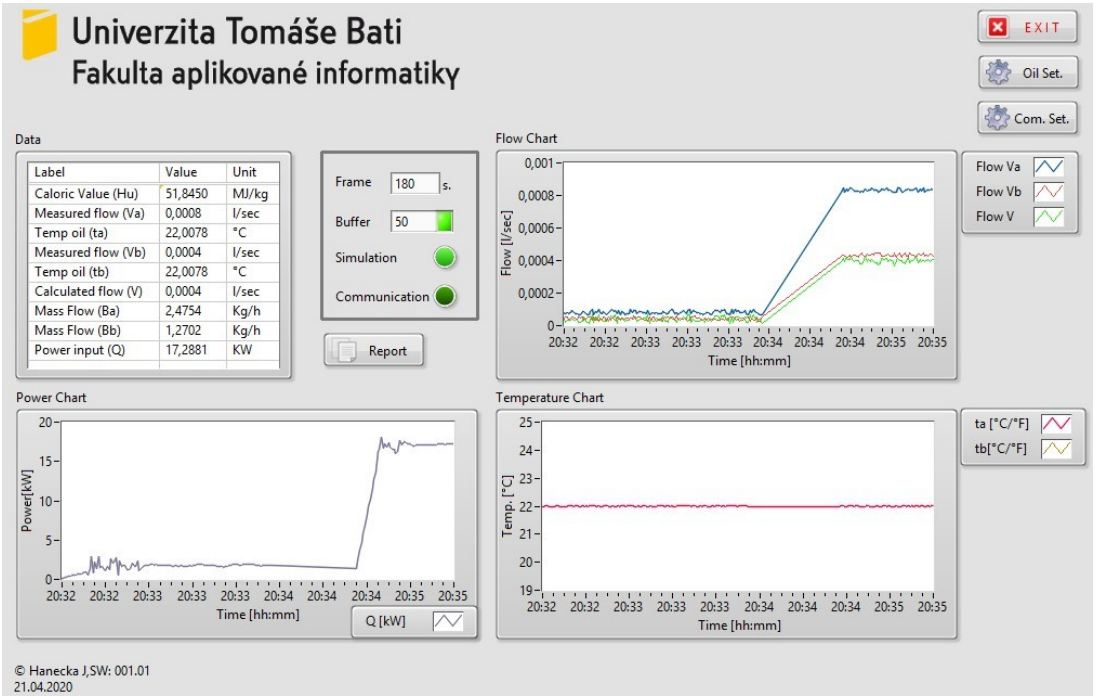
10.4 SPRACOVANIE CHYBNÝCH STAVOV

Daný modul sme implementovali z dôvodu budúceho odlaďovania stavových chýb, ktoré môžu byť riešené samotným programom. Myšlienkou je neriešiť chybný stav v danom moduly zvlášť a riskovať zaseknutie programu v nebezpečnom, respektíve nežiaducom stave. Ale naopak, riešiť problém po ukončení všetkých interakcií. Po každom vykonaní MHL jednotlivých modulov je v prípade výskytu chybového stavu odoslaná správa do EHL modulu na spracovanie chýb. Ten na základe chybných stavových kódov rozpozná chybu a v prípade, že sme si naprogramovali opatrenia, tie sa vykonajú, alebo sa chyba ohlási užívateľovi. Modul nebol súčasťou zadania ale javil sa mi, ako žiaduci v budúcom vylepšení softvéru.

10.5 GRAFICKÉ ROZHRANIE

Rozhranie bolo navrhnuté tak, aby užívateľovi poskytol jednoduché možnosti ovládania a zobrazenia potrebných nameraných dát. Rozhranie poskytuje tri základné priebehy meraných veličín v čase. Teda prietok vetvy „A“ , vetvy „B“ a ich rozdielu. Priebeh teplôt

daných vetiev. Konečnú hodnotu príkonu v Kilo Wattoch. Na displeji je možné pozorovať i aktuálne merané hodnoty v prehľadnej tabuľke. Pri meraných hodnotách sa vygeneruje aj informácia, o aké dáta sa jedná, teda či merané alebo simulované. Taktiež vstupnú veličinu pre dĺžku priemerovaných hodnôt z podprogramu priemer („MEAN“), modulu kalkulácie a jej spätnú väzbu. A v neposlednom rade, môžeme ovládať dĺžku zobrazovania priebehov na meraných grafoch v sekundách. Nezanedbateľnú časť tvorí i množina tlačidiel, ktoré volajú aktivované moduly pre ich konfiguráciu.



Obrázok 18 Grafické rozhranie softvéru. Zdroj: vlastný

Grafické rozhranie je jedno z najdôležitejších častí programu, nie len z dôvodu interakcie užívateľa so samotným softvérom, ale hlavne daná časť aktivuje jednotlivé moduly. Pre zostavenie nebola použitá naša základná architektúra ako pre jednotlivé module, ale použila sa štruktúra skladaných sekvencií. Pre životný cyklus programu boli nastavené štyri základné sekvencie. Nultá sekvencia je inicializácia kompletného softvéru, kde sa asynchrónne pomocou zinicializovaného poľa referencií našich modulov i aktivujú. Ako bolo už spomenuté, program potrebuje naviazať komunikáciu a nastaviť parametre paliva, preto sa v ďalšej prvej sekvencii volajú tieto dva moduly taktiež sekvenčne. Teda čakajú, kým užívateľ nedokončí jednu operáciu, aby sa volal modul nasledujúci. V tretej sekvencii sa odohráva hlavná časť programu. Daná sekvencie obsahuje slučku s manažérom udalostí. Manažér pracuje s udalosťou medzičasu, kde sa vykresľujú výsledky pomocou podpro-

gramu „GUI_data“ a spracovávajú interakcie s užívateľom pre znovuzvolenie potrebných modulov komunikácie, parametrov paliva alebo tvorby reportov. V poslednej štvrtej sekvencii sa všetky spustené moduly pozatvárajú. Uzatvorenie všetkých modulov je sekvenčne uzatvorené z poľa referencií aktívnych modulov vytvoreného v nultej sekvencii a taktiež odoslanie udalosti do všetkých modulov pre ukončenie činnosti.

10.5.1 PODPROGRAM „GUI_DATA“

Podprogram je volaný v každej slučke udalosti nazývanej medzičas („timeout“). Ide o prevzatie rámca údajov z opísaného správcu dát v predošlej kapitole. Dáta sú, koncipované v zobrazovacích grafoch poskytnutých priamo vývojovým prostredím LabVIEW. Tieto grafy sú pre x-ovú os konfigurované reálnym systémovým časom a jeho nastaviteľným počiatkom podľa požiadaviek užívateľa. Os x-ová je fixne definovaná v hodinách a minútach. Rozsah y-ovej os je dynamiky nastaviteľná podľa maximálnej hodnoty v zobrazovacom rámci, jej popis je prevzatý z modulu kalkulácie, kde sa pridelil rámec znakov odpovedajúci meranej jednotke. Podprogram tiež zostavuje pole reťazcov pre konečnú tabuľku meraných hodnôt.

10.6 TVORBA REPORTU

Posledný modul pridaný pre zlepšenie profesionálneho merania, bola možnosť tvorby reportov a tým zníženie času potrebného na ukončenie administratívnych prác po samotnom teste. Modul je možné volať z grafického rozhrania softvéru. Modul pracuje so základnou architektúrou obsluhy správ vo fronte. Teda pri prvej správe inicializácie nastaví cestu kde report uložiť a názov zostaví podľa nasledujúceho kľúča: „Report_“ + “aktuálny mesiac“ + “aktuálny deň“ + “_“ + “aktuálna hodina“ + “aktuálna minúta“, čím sa zabezpečí jedinečnosť názvoslovia, pretože sa nepredpokladá meranie s tvorbou reportov v každej minúte. Ďalej slučka spracovanie udalostí hibernuje do aktivácie tlačidla z grafickej časti daného modulu.

Modul obsahuje sadu nástrojov ktoré sa mi podaril získať z voľne dostupných fór pre vývoj v prostrediu LabVIEW. Túto sadu nástrojov na generovanie správ som prevzal z vývoju pre PDF, pretože som chcel byť schopný generovať správy PDF v LabVIEW bez toho, aby sa musela kupovať licencia. Využitie knižnice s otvoreným zdrojom iTextSharp

4.1.6 („Mozilla Public License Version 1.1“) pomohlo vyvinúť sériu LabVIEW nástrojov, ktoré umožňujú priamo generovať správy PDF bez nutnosti kúpiť komerčný produkt. Nevýhodou je však neaktuálnosť s najnovšími spúšťacími balíčkami pre Microsoft .NET Framework. Aplikácia .NET Framework 4.0 umožňuje načítať čisto spravované zostavy zabudované v akejkoľvek verzii .NET Framework a zostavy zmiešaného režimu zabudované v .NET 4.0. Aj keď LabVIEW štandardne načíta .NET „Common Language Runtime (CLR) 4.0“, môže LabVIEW použiť CLR 4.0 na načítanie zostáv .NET so zmiešaným režimom, ktoré sú zamerané na CLR 2.0. Pre nakonfigurovanie LabVIEW na načítanie zostáv zmiešaného režimu .NET, ktoré sú zacielené na CLR 2.0, sa museli vykonať nasledujúce kroky:

1. Skopírovať nasledujúci skript do textového editora:

```
<? xml version = "1.0" encoding = "utf-8"?>
< configuration >
        <startup useLegacyV2RuntimeActivationPolicy = "true">
                <supportRuntime version = "v4.0.30319" />
        </ startup>
</ configuration>
```

2. Tento nový konfiguračný súbor je uložený do rovnakého adresára ako LabVIEW.exe.

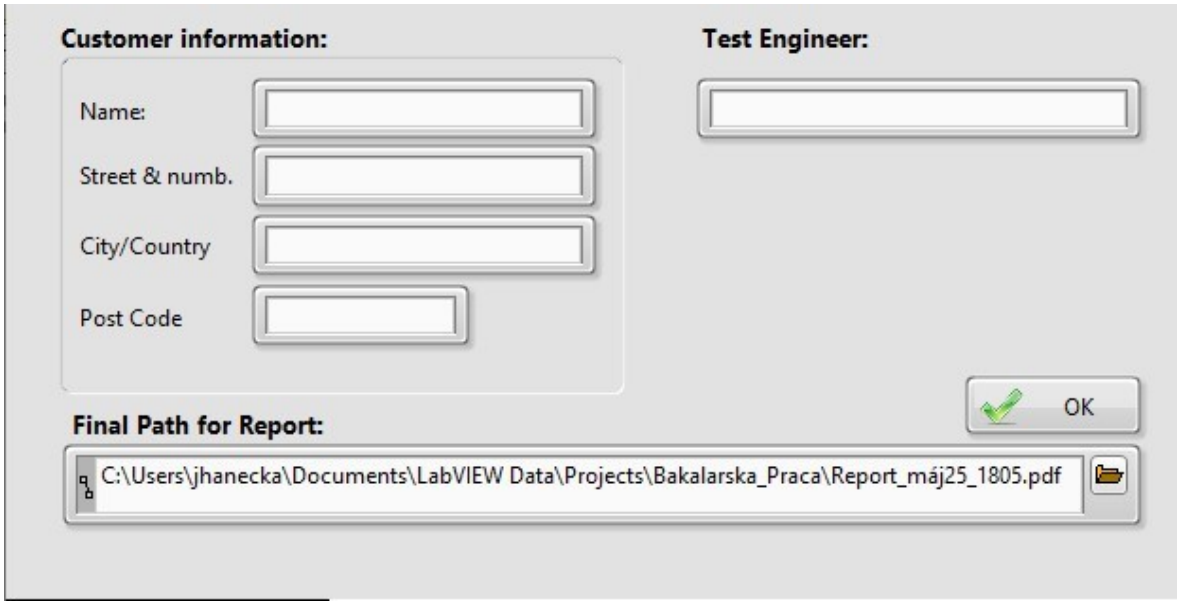
3. Pomenovanie súboru LabVIEW.exe.config. Je potreba znovu spustiť LabVIEW.

Ak by som uvažoval vybudovať samostatne stojací program spustiteľný len z LabVIEW-runtimovou podporou, teda bez LabVIEW prostredia bolo by nutné tieto kroky nakonfigurovať i v inštalačnom balíčku.

Využitý balíček nástrojov bol aplikovaný tak, aby sa mohol nakonfigurovať konečný PDF report zobrazujúci merané dáta. Príklad vytvoreného reportu nájdete v Príloha P 1: Príklad Reportu .

10.6.1 GRAFICKÉ ROZHRANIE MODULU

Z dôvodu možnosti v reporte spomenúť meno zákazníka pre ktorého je meranie podniknuté, bolo taktiež navrhnuté grafické rozhranie poskytujúce istý komfort z hľadiska obsluhy. Okno sa skladá z 3 častí. Spodná časť zobrazuje miesto ukladania reportov a tlačidlo pre jeho vytvorenie. Ľavá časť poskytuje vstupné polia pre iniciály zákazníka. Pravá časť vyžaduje meno testovacieho inžiniera, jeho ďalšie iniciály sú priamo zakorenené v kóde daného modulu a to hlavne z dôvodu uchovania značnej exkluzivity pre vlastníka softvéru.



Obrázok 19 GUI tvorby reportu. Zdroj: vlastný

11 OVERENIE

Pre potvrdenie teoretického rozboru kapitoly, bolo navrhnuté porovnať metódu nepriameho merania pomocou aplikovaného prietokomera KRAL BEM 500 so softvérom vychádzajúceho z tejto bakalárskej práce, k metóde priameho merania pomocou váhy a stopiek.

Meranie bolo prevedené súčasne, kedy sa u meraného zariadenia nastavil kontinuálny výkonnostný stupeň 6 kW. V rovnaký čas boli odčítavané hodnoty z váhy a softvéru. Výsledky sú uvedené v Tabuľka 4 Výsledok priamej a nepriamej . Za smerodajný porovnávací faktor bol zvolený rozptyl meraných hodnôt v celom časovom úseku.

Rozptyl bol počítaný podľa daného vzorca:

$$Rozptyl = \frac{\sum(x-\bar{x})^2}{(n-1)} \text{ [4]}$$

kde x je stredná (priemerná) hodnota vzorky a n je veľkosť vzorky.

Tabuľka 4 Výsledok priamej a nepriamej metódy

	Priama metóda			Nepriama metóda		
časový odber [min]	spotreba [kg]	hm. prietok [kg.sec^{-1}]	príkon Q [kW]	obj. prietok [l.min^{-1}]	hm. prietok [kg.sec^{-1}]	príkon Q [kW]
1	0,498	0,0083	5,9053	0,60457	0,008332	5,9282
5	2,496	0,00832	5,9195	0,60451	0,008333	5,9288
10	4,996	0,008327	5,9243	0,60457	0,008333	5,9288
15	7,499	0,0083322	5,9282	0,60457	0,008333	5,9288
20	9,999	0,0083325	5,9284	0,60457	0,008333	5,9288
60	29,999	0,0083331	5,9288	0,60457	0,008333	5,9288
Rozptyl [W]			**831 x10^{-4}**			**0,577 x10^{-4}**

Ďalším vyhodnotením bolo určenie potrebného času u priamej a nepriamej metódy k dosiahnutie zhody vypočítaného príkonu Q opísaného v kapitole 4 Kalkulácia príkonu olejových kotlovs podmienkou zhody na druhom desatinnom mieste v kilo wattoch.

ZÁVER

Práca v teoretickej časti bola zameraná na prieskum možnosti merania príkonu vykurovacích naftových kotlov a možnosť digitalizácie tohto merania. Výsledok teoretického rozboru poukázal na fakt, že v praxi najčastejším spôsobom merania je metóda váženia spotrebovaného paliva pomocou váhy, ako potvrdzuje príloha P 2: Anketa formy merania. Pre priemyselné aplikácie, kde sú požiadavky digitalizovať meranie viac a viac populárne, je metóda merania hmotnostného prietoku pomocou prietokomerov vyhľadávaná. Široký trh s prietokomermi a štandardmi pre sériovú komunikáciu medzi zariadeniami poskytujú dostatočný priestor pre realizáciu. V tomto prípade je však nutné implementovať korekciu použitého paliva na jeho hustotu ovplyvnenú prostredím, v ktorom sa nachádza, teda teplotou.

V praktickej časti sa dôraz kládol na návrh softvérového systému komunikujúceho s určeným meracím zariadením KRAL BEM 500. Platforma LabVIEW poskytla požadované nástroje k realizácii tohto zadania. Po naprogramovaní jednotlivých modulov, komunikácie, simulácie, výpočtov, korekcie paliva a tvorby reportov spolupracovali ako celok.

Výsledkom praktickej časti nie je len splnenie zadania, ale najmä demonštrácia porovnateľnej, respektíve lepšej presnosti merania. Taktiež doba nutná pre vyhodnotenie merania je kratšia rádovo o niekoľko minút, teda je možné v rovnakom čase vykonať viac meraní (testov). Taktiež čas nutný pre manuálnu kalkuláciu a tvorbu reportu je vďaka softvéru eliminovaný V neposlednom rade digitalizácia a grafické rozhranie pre užívateľa poskytuje možnosť sledovať priebeh spotreby paliva a teda správanie zariadenia, ktoré sa testuje (v reálnom čase je možnosť sledovať stabilitu, priebeh spotrebovaného paliva).

ZOZNAM POUŽITEJ LITERATÚRY

[1] Scalable Design Patterns in LabVIEW. In: Ni.com [online]. 2012 [cit. 2020-05-05]. Dostupné z: https://labviewwiki.org/wiki/Design_pattern#cite_note-1

[2] Application Design Patterns: Master/Slave. In: Ni.com [online]. 2018 [cit. 2020-05-05]. Dostupné z: http://www.ni.com/white-paper/3022/en.

[3] Queued Message Handler Template documentation. In: Ni.com [online]. 2017 [cit. 2020-05-05]. Dostupné z: http://www.ni.com/tutorial/53391/en/

[4] TRAVIS, Jeffrey a Jim KRING. LabVIEW for everyone: graphical programming made easy and fun. 3rd ed. Upper Saddle River: Prentice Hall, 2006. ISBN 978-0-13-185672-1.

[5] BAI, Ying. The Windows Serial Port Programming Handbook: 1. Computer interfaces. 2. Parallel programming (Computer science) 3. Ports (Electronic computer system). I. Title. Florida 33431: CRC Press, 2005. ISBN 0-8493-2213-8.

[6] W. F. Z. Lee and H. J. Evans, Density effect and Reynolds number effect on gas turbine flowmeters, Trans. ASME, J. Basic Eng., 87 (4): 1043-1057, 1965.

[7] W. F. Z. Lee, R. V. White, F. M. Sciulli, and A. Charwat, Self-correcting self-checking turbine meter, U.S. Patent No. 4,305,281, 1981.

[8] W. F. Z. Lee, D. C. Blakeslee, and R. V. White, A self-correcting and self-checking gas turbine meter, Trans. ASME, J. Fluids Eng., 104, 143-149, 1982

[9] STN EN 304: Vykurovacie kotly. Pravidlá skúšania vykurovacích kotlov s rozprašovacími horákmi na kvapalné palivá. Bratislava: Úrad pre normalizáciu, metrológiu a skúšobníctvo Slovenskej republiky, 2018. Triediaci znak: 07 5304

[10] NAUMCHIK, I.V., I.Yu. KINZHAGULOV a A.P. KREN. Mass flow meter for liquids. Scientific and Technical Journal of Information Technologies, Mechanics and Optics. 2015, , 900-906. DOI: 10.17586/2226-1494-2015-15-5-900-906. ISSN 22261494.

[11] ESSICK, John. Hands-on introduction to LabVIEW for scientists and engineers. 2nd ed. New York: Oxford University Press, c2013. ISBN 0199925151

[12] FRENZEL, Louis E. Handbook of seriál communications interfaces: a compehensive compedium of seriál digital input/output (I/O) standrds. Waltham, MA: Newnes, an imprint of Elsevier, [2016]. ISBN 978-012-8006-290

[13] ČSN EN 303-1. Kotle pro ústřední vytápění. Část 1: Kotle pro ústřední vytápění s hořáky s ventilátorem. Terminológie, všeobecné požadavky, zkoušení a značení. Praha: Úřad pro technickou normalizaci, metrologii a státní zkušebnictví, 1999. Třídící znak 07 5303

[14] BEATY, H. Wayne. Handbook of electric power calculations. 3rd ed. New York: McGraw-Hill, c2001. ISBN 9780071362986.

ZOZNAM POUŽITÝCH SYMBOLOV A SKRATIEK

B	Hmotnostný tok paliva [$kg.sec^{-1}$]
CAN	Controller Area Network.(zbernica riadiacej oblasti)
CLR	Common Language Runtime
COMLI	Communication Link (protokol pre sériovú komunikáciu)
CRC	Cyclic redundancy check (kontrola cyklickým kódom)
EHL	Event handler loop (slučka obsluhy udalostí)
EIA	Electronic industries association (Asociácia elektronického priemyslu).
GND	Ground (v elektronike označenie zeme)
GUI	Graphic User Interface (grafické užívateľské rozhranie)
IAC	Inter-application communication (komunikácia medzi aplikáciami)
ID	Identifikačné číslo
MHL	Message handler loop (slučka obsluhy správ)
MNS	Microsoft Notification Protocol (oznamovací protokol spoločnosti Microsoft)
MODBUS RTU	Dátový komunikačný protokol
NCV	Net calorific value (výhrevnosť) [$J.kg^{-1}$]
PDF	Portable Document Format (Prenosný formát dokumentov)

QA	Prietok vetvy „A“ v prietokomeru KRAL 500
Qb	Tepelný príkon kotlov podľa EN 304
QB	Prietok vetvy „B“ v prietokomeru KRAL 500
QMH	Queued Message Handler (Spracovateľ správ vo fronte)
RX	Receive (obdržaný signál)
S	hmotnostný podiel síry [-]
TA	Teplota vetvy „A“ v prietokomeru KRAL 500
TB	Teplota vetvy „B“ v prietokomeru KRAL 500
TX	Transmit (vysielaný signál)
USB	Universal Serial Bus (Univerzálna sériová zbernica).
VI	Virtual Instrument (program napísaný v grafickom jazyku LabVIEW)
ρ_{15}	hustota vykurovacieho oleja pri teplote 15 °C v [$kg.m^{-3}$]

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ZOZNAM PRÍLOH

Príloha P 1: Príklad Reportu

Príloha P 2: Anketa formy merania

Príloha P3: prilohy.zip

.

PRÍLOHA P 1: PRÍKLAD REPORTU

Laboratory:

Hanecka-lab.s.r.o.
Vystavni 27
Hodonin
69501

Customer:

Nikola
Výstavní 27
Hodonín
69501

Test Results:

Measurement has been created according to test description in line with EN 303-1. Instruments were calibrated according to ISO 17025. The test results refer exclusively to the samples under test. Excerpts from this document may only be reproduced with the written approval of Customer of this report.

Table1 : Measured Value from selected OIL type

Label	*Value*	*Unit*
Caloric Value (Hu)	51,8450	MJ/kg
Measured flow (Va)	0,0008	l/sec
Temp oil (ta)	20,9903	°C
Measured flow (Vb)	0,0004	l/sec
Temp oil (tb)	20,9903	°C
Calculated flow (V)	0,0004	l/sec
Mass Flow (Ba)	2,8041	Kg/h
Mass Flow (Bb)	1,3879	Kg/h
Power input (Q)	20,2577	KW

Test Engineer:

Haneka Juraj: .....(Sign)

Annex

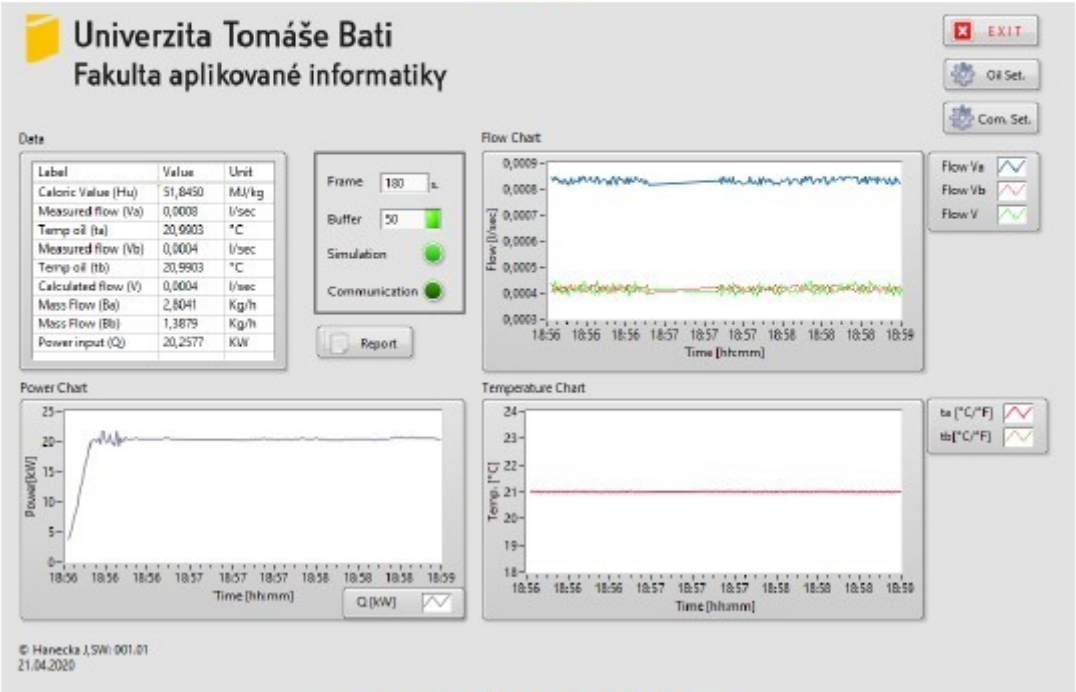


Figure1 : detail view of test panel

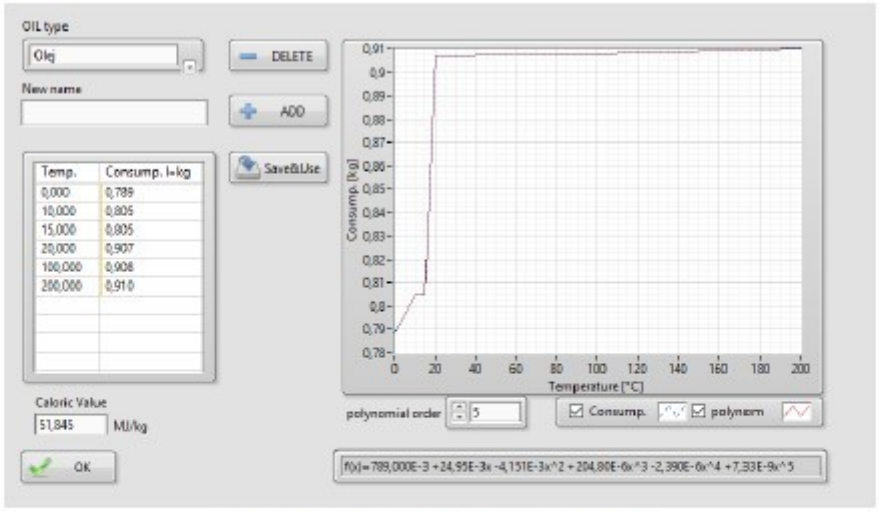


Figure2 : detail view of selected oil

PRÍLOHA P 2: ANKETA FORMY MERANIA

Anketa bola spracovaná za účelom porozumenia testovacích metód, postupov a technického vybavenia pri určení spotreby naftových kotlov s tlakovými alebo zmiešavajúcimi rozprašujúcimi horákmi palivo-vzduch. Dopytované inštitúty nechceli byť spomenuté v bakalárskej práci „Návrh a vývoj softwarových modulov v prostredí LabVIEW pre meranie príkonu naftových kotlov“, pre ktorú bola táto anketa spracovaná.

Otázky ankety:

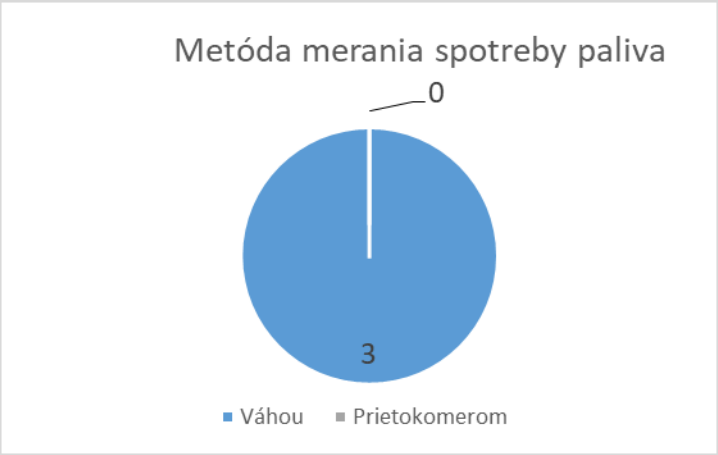
1. Je inštitút akreditovaný na test naftových zariadení?
2. Akou metódou je meraná spotreba paliva: váhou / prietokomerom / iné ?
3. V prípade využitia váhy:
    a. Aký rozsah je požitý?
    b. Aká presnosť je využitá?
    c. Aká je minimálna doba pre meranie jedného výkonového stupňa?
4. V prípade prietokomera:
    a. Aká je presnosť?
    b. Aká je minimálna doba pre meranie jedného výkonového stupňa?

Spracovanie ankety:

Z piatich dopytovaných testovacích inštitútov boli tri akreditované na meranie naftových kotlov podľa normy ISO 17025:2015. Výsledok Graf 1 vyhodnotenie prvej otázky ankety.

Graf 1 vyhodnotenie prvej otázky ankety

Z akreditovaných testovacích inštitútov, 100 % používa meranie pomocou váhy. Výsledok Graf 2 vyhodnotenie druhej otázky ankety.



Graf 2 vyhodnotenie druhej otázky ankety

Rozbor presností, rozsahov a doby merania môžeme nájsť v Tabuľka 5 Vyhodnotenie tretej otázky ankety.

Tabuľka 5 Vyhodnotenie tretej otázky ankety

Rozsah	Zastúpenie v inštitútoch	Presnosť	Doba merania jedného výkonového stupňa
0-10 kg	1	$\pm 2x10^{-5}$	10 minút
0-100 kg	2	$\pm 2x10^{-3}$	20 minút

Vyhodnotenie poslednej anketovej otázky („Aká je minimálna doba pre meranie jedného výkonového stupňa?“) nebolo relevantné, nakoľko ani jeden inštitút túto metódu nevyužíval. Na dodatočný dotaz z akého dôvodu sa táto metóda nevyužíva?: „*ďalšia investícia pre presnejšie meranie a rýchlejšie meranie neadekvátna ku frekvencii dopytovaných testov od zákazníkov, investícia by bola opodstatnená pri 30-50 meraní za rok*“. Investičná rozvaha však nebola súčasťou ankety.

Záver ankety:

Z ankety bolo objasnené akým smerom sa akreditované inštitúty odoberajú pri meraní spotreby paliva pre naftové zariadenia. Metóda popísaná v EN 303-1 pre vykurovacie kotly s tlakovými horákmi je jediná implementovaná. Doba testu pre jeden výkonový stupeň nie je kritická k dopytu požadovaných testov a implementácia objemových alebo hmotnostných prietokomerov má opodstatnenie pri vyššej frekvencii testov napríklad na strane výrobcu.