VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY
DEPARTMENT OF CONTROL AND INSTRUMENTATION

DATALOGGER ŘÍDICÍ JEDNOTKY PLYNOVÉHO KOTLE

DATA LOGGER FOR GAS BOILER

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE — Peter Frnka
AUTHOR

VEDOUCÍ PRÁCE — Ing. Petr Petyovský
SUPERVISOR

BRNO 2016

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní obor **Automatizační a měřicí technika**
Ústav automatizace a měřicí techniky

Student: Peter Frnka ***ID:*** 164593
Ročník: 3 ***Akademický rok:*** 2015/16

NÁZEV TÉMATU:

Datalogger řídicí jednotky plynového kotle

POKYNY PRO VYPRACOVÁNÍ:

1) Seznamte se s dostupnými možnostmi a funkcemi zadavatelem využívaných řešení pro autonomní sběr a záznam digitálních dat při kontrole řídicí jednotky plynového kotle. Nastudujte a zhodnoťte zařízení pro autonomní sběr a záznam digitálních dat na aktuálním trhu.
2) Definujte veličiny určené ke sběru a jejich významy, povolené rozsahy a digitální reprezentaci. Definujte požadované periody vzorkování pro jednotlivé veličiny.
3) Navrhněte koncept celého zařízení pro sběr digitálních dat ze vstupů a výstupu řídicí jednotky plynového kotle. Diskutujte požadavky kladené na HW platformu zvolenou pro datalogger. Navrhněte vhodný formát pro sběr dat a jejich záznam do úložiště.
4) Realizujte firmware dataloggeru pro OS Linux umožňující autonomní sběr a záznam dat.
5) Navrhněte webovou prezentační vrstvu pro zobrazení dat uživateli. Definujte uživatelské role a jejich přístupové práva ve webové aplikaci.
6) Realizujte navrženou webovou aplikaci a prezentujte ji zadavateli. Vyhodnoťte výsledky testování na reálném zařízení zadavatele.
7) Zhodnoťte dosažené výsledky, uveďte výhody i nevýhody řešení. Navrhněte další úpravy a vylepšení.

DOPORUČENÁ LITERATURA:

[1] HEROUT, Pavel: Učebnice jazyka C. 4., přeprac. vyd., České Budějovice: Kopp, 2004, 271 s. ISBN 80-723--220-6.

Termín zadání: 8.2.2016 ***Termín odevzdání:*** 23.5.2016

Vedoucí práce: Ing. Petr Petyovský
Konzultant bakalářské práce: Ing. Jiří Macháček (ACS Honeywell)

doc. Ing. Václav Jirsík, CSc., *předseda oborové rady*

Abstrakt

Práca sa zaoberá návrhom a realizáciou dataloggeru určeného k zberu dát z riadiacich jednotiek plynových kotlov, ktoré sú vyvíjané firmou Honeywell Brno. Datalogger je realizovaný na hardwarovej platforme Raspberry Pi v spojení so zadávateľom dodanými konvertormi napätia. Meracia aplikácia, ktorá je súčasťou firmwaru dataloggeru a vytvára vzorky snímaných vstupov je naprogramovaná v jazyku C.

Všetky dáta, ktoré sú namerané počas behu dataloggeru sú ukladané na dátové úložisko, z ktorého sú prezentované užívateľovi pomocou webového rozhrania. Webová aplikácia a dátové úložisko sú v konečnom dôsledku realizované na hlavnom servery celého realizovaného systému, ktorý je súčasť lokálnej siete vo firme Honeywell Brno. Týmto riešením vznikol systém pre zber dát z riadiacich jednotiek plynových kotlov pri ich vývoji a testovaní.

Kľúčové slová

Datalogger, Plynový kotol, Embedded system, ARM Cortex, Raspberry Pi

Abstract

This bachelor thesis deals with design and realization of a datalogger designated for logging data acquired gas boiler controls. The gas boiler controls are under development by Honeywell Brno. The datalogger is implemented on Raspberry Pi hardware platform with voltage converters supplied by Honeywell Brno. Measurement application, which is part of datalogger firmware was developed in C programming language.

All data that are measured during datalogger runtime are stored on data storage. The measured data are presented to the user by a web application. The web application and data storage are put into operation on the main server, which is the part of local network in Honeywell Brno. In result of this thesis, data logging system capable of collecting data from furnace controls was developed. Created device helps with research, development and testing of furnace controls.

Keywords

Datalogger, Gas boiler, Embedded system, ARM Cortex, Raspberry Pi

Bibliografická citácia:

FRNKA, P. *Datalogger řídicí jednotky plynového kotle*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2016. 68s. Vedoucí bakalářské práce byl Ing. Petr Petyovský.

Prohlášení

Prohlašuji, že svou bakalářskou práci na téma Datalogger řídicí jednotky plynového kotle jsem vypracoval samostatně pod vedením vedoucího diplomové (bakalářské) práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové (bakalářské) práce dále prohlašuji, že v souvislosti s vytvořením této diplomové (bakalářské) práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: 23. května 2016 ……

podpis autora

Poďakovanie

Ďakujem vedúcemu bakalárskej práce Ing. Petrovi Petyovskému za účinnú metodickou, pedagogickou a odbornou pomoc a trpezlivosť pri zapracovaní mojej bakalárskej práce. Moje poďakovanie patrí aj konzultantovi Ing. Jířimu Macháčkovi za odbornú pomoc a konzultácie zadania.

V Brne dňa: 23.5 2016 ……

podpis autora

OBSAH

1 ÚVOD

Táto práca sa zaoberá návrhom a realizáciou hardwaru a firmwaru pre datalogger riadiacich jednotiek plynových kotlov, ktoré sú vyvíjané firmou Honeywell Brno. Návrh hardwaru v tejto práci spočíva v stanovení základných požiadaviek na hardwarovú platformu, ktorá bude použitá pre datalogger.

Firmware dataloggeru by mal byť navrhnutý a realizovaný tak, aby čo najlepšie spĺňal požiadavky zadávateľa na periódu vzorkovania snímaných vstupov a vytváral namerané dáta s čo najvyššou presnosťou periódy vzorkovania.

Ďalším cieľom tejto práce je realizácia webového rozhrania pre datalogger. Webové rozhranie by malo slúžiť k prezentácií nameraných dát a k celkovej správe dataloggeru. To znamená spúšťanie záznamu dát, výber vstupov dataloggeru, ktoré budú snímané a priradenie vzorkovacích periód k snímaným vstupom.

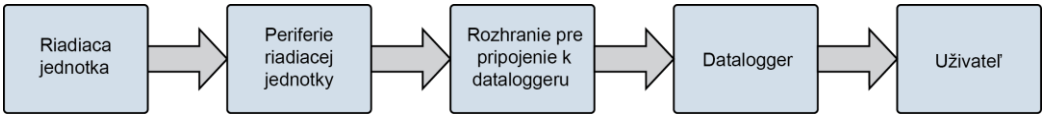
Hlavným dôvodom stanovenia tohto zadania je v prvom rade potreba záznamu dát zo vstupov a výstupov riadiacich jednotiek plynových kotlov. Ďalším dôvodom stanovenia tohto zadania je zvyšovanie nárokov pri zázname dát pri vývoji a testovaní, čo vedie k zlepšovaniu a rozširovaniu aktuálneho konceptu zberu dát z riadiacich jednotiek plynového kotla využívaného zadávateľom.

Záznam vstupov a výstupov riadiacich jednotiek plynového kotla je nevyhnutná súčasť pri ich vývoji a následnom testovaní. Znalosť týchto informácií pri testovaní častokrát môže viesť k odhaleniu niektorých chybových stavov, no zároveň k overeniu správnej činnosti a funkčnosti vyvíjaných riadiacich jednotiek plynových kotlov pred ich uvedením na trh.

V druhej časti práce bude potrebné navrhnúť a realizovať zadávateľom požadovanú webovú aplikáciu, ktorá bude slúžiť na prezentáciu nameraných dát. Webová aplikácia bude bežať na webovom serveri. Pomocou webovej aplikácie bude možné pridávať dataloggeru snímane kanály, priraďovať im mená a konfigurovať periódy vzorkovania, s ktorými sa budú vzorkovať dáta. Vo webovej aplikácií bude potrebné dáta prezentovať užívateľovi pomocou grafu.

2 ROZBOR ZADANIA

Zadanie tejto práce bolo definované firmou Honeywell Brno. Hlavným účelom stanoveného zadania je návrh a realizácia konceptu aplikácie pre autonómny zber digitálnych dát zo vstupov a výstupov riadiacej jednotky plynového kotla, ktoré sú vyvíjané uvedenou firmou. Toto zariadenie bude využívané pri testovaní a vývoji týchto riadiacich jednotiek.



Obrázok 1 Základná koncepcia meracieho reťazca

Obrázok 1 hovorí o základnej koncepcií meracieho reťazca vyplývajúcej z uvedenej blokovej schémy. Celý merací reťazec sa začína sledovanou riadiacou jednotkou, ku ktorej sú pripojené jej periférie. Bloky riadiacej jednotky a jej periférií sú v konečnom dôsledku súčasť celého plynového kotla, no z hľadiska testovania a vývoja sa nepoužíva celý kotol, ale len riadiaca jednotka, ku ktorej sú pripojené všetky potrebné periférie poprípade ostatné akčné členy.

Periférie riadiacej jednotky je možné na začiatku považovať za jednotlivé časti celého kotla, bez ktorých by celý proces spaľovania nemohol fungovať. To znamená, že ide napríklad o plynové ventily, no podrobnejšiemu popisu periférií riadiacich jednotiek bude venovaná samostatná kapitola, v ktorej bude uvedený praktický príklad pripojenia periférii riadiacej jednotky plynového kotla. Ďalej sú signály, pomocou ktorých sú periférie ovládané riadiacou jednotkou privedené do bloku, ktorý tvorí rozhranie medzi perifériami a dataloggerom. Hlavnou úlohou tohto bloku je vhodná príprava meracích signálov na spracovanie dataloggerom. Ďalším a najhlavnejším blokom v meracom procese je samotný datalogger. Z tejto blokovej schémy vyplýva, že datalogger bude realizovaný ako samostatné zariadenie, ktoré sa bude pripájať k signálom, pomocou ktorých sú ovládané periférie riadiacej jednotky plynového kotla. Posledným blokom je užívateľ, ktorý je z hľadiska meracieho procesu na najvyššej úrovni čo znamená sledovanie nameraných hodnôt alebo ich ďalšie spracovanie.

Ďalšou časťou rozboru zadania sú požiadavky zadávateľa na datalogger. Aby bolo možné navrhnúť koncept celého zariadenia na účel dlhodobého záznamu dát z riadiacej jednotky plynového kotla, bolo najprv potrebné stanoviť požiadavky zadávateľa na zariadenie. Tieto požiadavky sa skladajú z viacerých častí, no v základe ich je možné rozdeliť na dve veľké časti, čím sú požiadavky na hardware a požiadavky na firmware dataloggeru. Na základe analýzy riadiacich jednotiek vyvíjaných firmou Honeywell Brno a niekoľkých konzultáciách so zadávateľom boli stanovené jednotlivé nároky na datalogger, ktoré budú popísané v tejto kapitole. Konkrétne riešenia uvedených požiadaviek budú popísané a vyriešené v návrhovej časti dataloggeru tejto práce.

2.1 Požiadavky zadávateľa na hardware dataloggeru

V nasledujúcej kapitole budú bližšie popísané požiadavky zadávateľa na hardwarovú časť dataloggeru, čo bolo prvým krokom celej práce.

2.1.1 Počet snímaných vstupov

Počet vstupov je jedným zo základných hardwarových parametrov meracích zariadení, pod ktoré je konečnom dôsledku možné zaradiť aj popisovaný datalogger riadiacej jednotky plynového kotla.

Počet snímaných vstupov bol zadávateľom stanovený na 16, pričom je potrebné, aby celý návrh zariadenia počítal s rezervou pre budúce rozšírenie dataloggeru. V budúcnosti bude možné datalogger rozšíriť o minimálne 5 ďalších vstupov, z čoho vyplýva, že návrh zariadenia musí počítať s 21 snímanými vstupmi.

2.1.2 Napäťové úrovne a definícia signálov na vstupoch dataloggeru

Firmou vyvíjané riadiace jednotky plynových kotlov pracujú s rôznymi signálmi na ich vstupoch a výstupoch, ktoré je potrebné zariadením snímať. Z dôvodu návrhu ďalších hardwarových častí, ktoré budú v prípade ich potreby pripojené k dataloggeru na vhodnú úpravu vstupných signálov, bolo potrebné diskutovať a následne jasne definovať rozsahy napäťových úrovní a druh snímaných signálov na vstupoch dataloggeru.

Zadávateľom bolo definované, že pri návrhu zariadenia budú uvažované ideálne logické stavy na snímaných vstupoch dataloggeru, čo znamená 0 V pri stave vypnuté a 3.3 V pri stave zopnuté.

Zadávateľ musel už počas vývoja riadiacich jednotiek plynových kotlov vyvinúť zariadenie, ktoré je schopné konvertovať úrovne napätia, pomocou ktorého sú riadené periférie riadiacich jednotiek. Súčasťou riešenia tohto zariadenia je taktiež galvanické oddelenie vstupu od výstupu a možné prepätie na vstupe. Toto zariadenie bude zadávateľom dodané ako hotové riešenie, z tohto dôvodu je možné pri návrhu a následnej realizácii uvažovať na vstupoch dataloggeru ideálne logické stavy.

2.1.3 Rozsah periód vzorkovania

Pri dlhodobom zázname dát je jedným z hlavných parametrov perióda vzorkovania. Pri návrhu je nutné poznať limitné rozsahy pre minimálne a maximálne periódy vzorkovania na vstupoch dataloggeru. Minimálna perióda vzorkovania je v tomto smere dôležitá informácia z dôvodu celého prístupu k návrhu a realizácií. Ak by bolo potrebné vzorkovať snímané vstupy častejšie ako 10 krát za sekundu, celý prístup k návrhu by sa musel zmeniť, čo by sa v konečnom dôsledku odrazilo aj na návrhu hardwarovej časti.

Pri zázname zadávateľom požadovaných dát z riadiacej jednotky nebude potrebné používať periódy vzorkovania nižšie ako 100 milisekúnd a to z nasledujúcich dôvodov. Datalogger nebude zaznamenávať žiadne komunikačné zbernice riadiacej jednotky ani iné rýchle deje, pri ktorých by bolo potrebné vzorkovať s vyššou frekvenciou ako je 10 Hz. Pôjde len o jednoduchý záznam logických stavov pripojených periférií, ktoré sledovaná riadiaca jednotka spína alebo rozopína. Tieto deje prebiehajú rádovo v sekundách a nadobúdajú len logické stavy.

Zadávateľom boli stanovené rozsahy vzorkovacích periód od 100 milisekúnd až po 1 hodinu. Na splnenie správneho vzorkovania podľa zadávateľa nebude potrebné dosiahnuť striktne ekvidistantný krok vzorkovania, no bude potrebné, aby bol ekvidistantný krok dodržaný vo viac ako 75 percentách z celkového počtu vzoriek pri zázname dát. Pokiaľ nebude možné vzorkovať s danou periódou, vzorkovanie bude realizované s periódou, ktorá je pri aktuálnych podmienkach možná, pričom každý dátový bod bude mať k sebe informáciu o čase, kedy bol vykonaný.

2.2 Požiadavky zadávateľa na firmware dataloggeru

Jednou z ďalších požiadaviek zadávateľa na celé zariadenie sú taktiež požiadavky na firmware dataloggeru. Ich dostatočné splnenie bude tvoriť väčšiu časť tejto práce, nakoľko hardwarové požiadavky je možné splniť výberom vhodnej platformy na vývoj softwaru a použitím dodaných hardwarových častí zadávateľom, ktoré už existujú ako hotové riešenia a používajú sa pri vývoji a testovaní riadiacich jednotiek plynových kotlov.

Požiadavkou zadávateľa na firmware dataloggeru je, aby bol určený primárne pre operačný systém Linux. Túto požiadavku je možné splniť výberom vhodnej hardwarovej platformy, na ktorej bude prebiehať vývoj celého konceptu.

Pri požiadavkách na firmware dataloggeru je potrebné poznamenať, že bude podporovaný na ktorejkoľvek hardwarovej platforme, ktorá podporuje operačný systém Linux. Hardwarových platforiem je na aktuálnom trhu veľké množstvo, pričom sú postavené na rôznych procesoroch. Z tohto dôvodu sa predpokladajú maximálne zmeny, ktoré môžu nastať v naprogramovanom zdrojovom kóde pri použití na inej platforme. Tieto zmeny môžu byť napríklad prekonfigurovanie a mapovanie vstupných portov, ktoré bude závisieť na výbere platformy.

Keďže sa bude jednať o dlhodobý záznam dát, ďalšou z podmienok je, aby bol firmware dostatočne stabilný pri behu na zvolenej hardwarovej platforme. Ak firmware nebude spĺňať túto podmienku pri behu a zázname dát z riadiacej jednotky plynového kotla, môže to spôsobiť stratu dát, čím bude znehodnotené celé meranie a testovanie riadiacej jednotky.

2.2.1 Komunikácia s užívateľom

Aby bolo možne navrhnúť a realizovať komunikáciu s dataloggerom, bolo potrebné zistiť požiadavky na formu komunikácie, správu dataloggeru a jeho celkovú konfiguráciu, nakoľko bude možné zvoliť viacero parametrov pre jednotlivé snímané vstupy dataloggeru.

Komunikácia s užívateľom bude realizovaná pomocou webového rozhrania, pričom bude prebiehať vo firemnej lokálnej sieti. Súčasťou realizácie komunikácie s užívateľom bude aj webová aplikácia, ktorej prioritný účel bude konfigurácia celého dataloggeru a prezentácia nameraných dát. Pojem konfigurácia dataloggeru zahŕňa nastavenie celkového počtu snímaných kanálov, na ktorých bude datalogger spúšťať meranie a príslušné periódy vzorkovania v zadávateľom definovanom rozmedzí.

Ďalšou možnosťou tejto aplikácie bude prezentácia zaznamenaných dát vo zvolenom formáte, ktorý bude závisieť na konkrétnom návrhu. Táto webová aplikácia bude riešiť aj prístupové práva do celého systému. Zadávateľom bolo definované, že prístup do aplikácie bude realizovaný menom a heslom, na základe ktorého bude možné pristupovať k dátam, konfigurovať a komunikovať s dataloggerom.

2.2.2 Ukladanie nameraných dát

Firmware bude namerané dáta ukladať na dátové úložisko, ktoré bude ďalej preberané v návrhovej časti tejto práce. Požiadavkou zadávateľa na dátové úložisko je, aby bolo realizované na pamäťovej karte, ktorá má byť súčasťou hardwarovej platformy. Dátové úložisko môže byt realizované aj ako externý server, ku ktorému bude datalogger pristupovať. Konkrétne riešenie dátového úložiska bude preberané v návrhovej časti tejto práce.

3 MOŽNÉ RIEŠENIA ZBERU DÁT VYUŽÍVANÉ V PRAXI

Problém autonómneho záznamu dát z riadiacej jednotky plynového kotla bolo potrebné vyriešiť už počas jeho vývoja. Vzniklo viacero možností zberu dát, ktoré sa so zvyšovaním nárokov na záznam dát tohto charakteru stávali nedostačujúce pri vývoji a následnom testovaní, čo malo za následok požiadavky na vývoj nového a výhodnejšieho riešenia tohoto problému. Riešenia využívané v praxi je možné rozdeliť do troch nasledujúcich častí.

3.1 Použitie PC a prídavnej meracej karty

Pri vývoji a následnom testovaní riadiacich jednotiek plynových kotlov je jednou z možností autonómneho zberu dát použitie počítača v spojení s rôznymi vstupno-výstupnými meracími kartami. Tieto prídavné karty vytvárajú rozhranie medzi sledovanou riadiacou jednotkou a softwarom, ktorý je spustený v počítači. Riadiaca jednotka pracuje s viacerými úrovňami napätia, ktoré je potrebné pri vývoji a kontrole zaznamenávať. Používané meracie karty pripojené k počítaču majú, tak ako všetky meracie zariadenia, určité dovolené medze úrovní vstupného napätia, typicky maximálne 5 V alebo 3.3 V. Z tohto dôvodu sa používajú konvertory napätia, ktoré upravujú úrovne privedených napätí na vstupy meracej karty. Softwarové vybavenie takého počítača je zvyčajne naprogramovaná aplikácia v programovacích vývojových prostrediach ako je napríklad Matlab alebo LabVIEW, ktorej úlohou je komunikovať s meracou kartou, poskytovať jej všetky potrebné informácie o meraní a podobne.

Nakoľko sa pri tomto riešení využívajú programovacie vývojové prostredia a vstupno-výstupné meracie karty určené aj na tento účel, nároky na vývoj takýchto aplikácií je nízky a prebieha dostatočne rýchlo v porovnaní s vývojom iných riešení. Keďže sú namerané dáta priamo uložené v použitom počítači, ďalšou výhodou tohto riešenia je export nameraných dát a ich ďalšie použitie, poprípade vykresľovanie rôznych závislostí nameraných dát.

Hlavnou nevýhodou tohto riešenia je nutnosť použitia osobného počítača, ktorý samostatne vo svojom základe nemá žiadne vstupné a výstupné porty, ktoré by bolo možné použiť ako rozhranie medzi softwarom spusteným na počítači a sledovanou riadiacou jednotkou. Z tohto dôvodu, ako už bolo spomenuté, je potrebné použiť prídavné vstupno-výstupné karty na rozšírenie periférii počítača. Tieto karty a použitý počítač sú na tento účel zbytočne predimenzované, čo má za následok ich vysokú cenu, v porovnaní s požadovaným riešením tejto práce. Ďalšou nevýhodou spojenou s použitím počítača a prídavnej meracej karty v tomto koncepte riešenia je komplikovaná manipulovateľnosť s počítačom a použitou meracou kartou.

Častokrát je potrebné zaznamenávať niektoré dáta v konkrétnej aplikácií aj u zákazníka pri testovaniach, čo je s použitím klasického počítača viac menej nemožné.

Nakoľko sa na vývoj použitej aplikácie používajú programové vývojové prostredia ako je napríklad LabVIEW a Matlab, ich používanie a licencie na prevádzku navyšuje celkovú cenu konceptu, čo je samozrejme považované za nevýhodu tohto riešenia.

3.2 Použitie komerčného dataloggeru

Pokiaľ pôjde len o krátkodobý záznam dát z riadiacej jednotky plynového kotla, na zistenie potrebných informácií je možné použiť logický analyzátor alebo z časti osciloskop. Pokiaľ pôjde o dlhodobý záznam dát, je potrebné použiť zariadenie určené na tento účel. Na aktuálnom trhu je možné nájsť veľké množstvo zariadení na dlhodobý záznam a zber dát rôzneho druhu. V nasledujúcej podkapitole sú popísané niektoré zo zariadení určené na dlhodobý zber dát, ktoré sa svojou funkčnosťou a typickým použitím hodia na záznam dát zo vstupov a výstupov riadiacej jednotky plynového kotla.

3.2.1 Datalogger DATAQ Instruments DI - 160

Výrobcom Dataloggeru DI – 160 je firma DATAQ Instruments. Typickým použitým tohto dataloggeru je podľa výrobcu záznam udalostí, ktoré nastanú počas vykurovacieho cyklu plynového kotla. Datalogger je vybavený viacerými programovateľnými snímacími módami. V prvom prípade môžu byť snímané vstupy nastavené na detekciu udalostí. Detekcia udalosti znamená, že je potrebné vedieť informáciu len o začiatku udalostí, pričom nie je potrebná informácia, kedy udalosť skončila. Ak nastalo počas intervalu viacero udalostí, zaznamenaná bude iba jedna.

Ďalšou z možností je nastavenie vstupu na snímanie stavu. To znamená, že v okamžikoch periódy vzorkovania je zaznamenaný stav vstupu. Výstupom tohto módu môže byť napríklad: zariadenie bolo zapnuté o 9:00 a zostalo zapnuté do 12:00. Zariadenie bolo znova zapnuté o 13:00.

Ďalšou z možností nastavenia snímacieho módu je počítadlo udalostí. Týmto módom je možné počítať udalosti v jednom časovom intervale. Ako príklad môžeme použiť : zariadenie vyprodukovalo 80 výrobkov za minútu.

Vnútorné hodiny reálneho času poskytujú informáciu o dátume a čase pre každý zachytený dátový bod. Namerané dáta sú ukladané na úložisko, ktoré je realizované ako SD pamäťová karta. Dáta sú ukladané vo formáte, ktorý oddeľuje čiarkou všetky namerané hodnoty, takže zaznamenané dáta sú čitateľné pre užívateľa a ľahko importovateľné do iných aplikácií ako je napríklad Microsoft Excel.

Datalogger DI – 160 je vybavený 8 vstupnými kanálmi, ktoré sú rozdelené na štyri vysokonapäťové a ďalšie štyri na nízkonapäťové vstupy dataloggeru. Vstupom vysokonapäťových kanálov môže byť signál, ktorý má maximálnu hodnotu napätia ± 300 V alebo efektívnu hodnotu napätia 230 V.

Nízkonapäťové kanály sú vybavené vnútornými pull-up rezistormi a môžu byť použité na detekciu jednosmerného napätia s hodnotou menšou ako 30 V. Pomocou USB rozhrania a aplikácie je možné datalogger konfigurovať na nasledujúce meranie. Datalogger môže byť napájaný z AC adaptéra alebo dobíjacích batérií. Cena stanovená výrobcom je 276 EUR. Bližšie informácie o tomto zariadení je možné nájsť [1]. Obrázok 2 zobrazuje zariadenie DI – 160.

Obrázok 2 Datalogger DATAQ Instruments DI - 160 [1]

3.2.2 Datalogger MadgeTech State101A

Výrobcom dataloggeru State101A je firma MadgeTech a je určený do priemyselného prostredia. Datalogger monitoruje a nahráva udalosti, trvanie a stav preddefinovaných udalostí pomocou merania zmien v napätí, tak ako aj ostatné spomínané zariadenia. Typicky je tento datalogger pripájaný k relé alebo spínaču snímaného zariadenia.

Datalogger State101A je riešením pre radu priemyselných odvetví a aplikácii. Jeho primárnym účelom je monitorovanie a záznam dát vykurovacích a chladiacich systémov alebo monitorovanie rôznych čerpadiel ako sú napríklad vodné a plynové čerpadlá.

Zariadenie poskytuje iba jeden vstupný kanál, ktorého rýchlosť snímania je užívateľom konfigurovateľná a je možné ju nastaviť na maximálne 4 vzorky za sekundu a minimálne jednu vzorku za hodinu. Maximálne vstupné napätie je výrobcom definované na 30V, pričom úroveň napätia pre vyhodnotenie logického stavu „Low“ musí byť menšia ako 0.4 V a pre stav „High“ musí byť úroveň napätia väčšia ako 2.8V. Z uvedených rozsahov vyplýva, že tento datalogger je možné použiť aj na záznam TTL úrovní napätia. Úložisko dát je navrhnuté tak, aby bolo schopné uložiť maximálne 400 000 vzoriek nameraných dát. Tak ako spomínaný datalogger DI – 160, je možné aj toto zariadenie napájať nabíjateľnou batériou. Cena tohto zariadenia je výrobcom stanovená na približne 91 EUR. Bližšie informácie a špecifikáciu je možné nájsť v [2]. Obrázok 3 zobrazuje datalogger State 101A.

Obrázok 3 Datalogger MadgeTech State101A [2]

3.2.3 Datalogger MadgeTech RFPulse2000A

Ako už z jeho názvu RFPulse2000A vyplýva, jedná sa o datalogger, ktorý je používaný na záznam počtu pulzov. Ide o bezdrôtové zariadenie, primárne určené na záznam rôznych signálov, ktoré majú pulzný charakter. Typickým použitím tohto dataloggeru je napríklad záznam dát z prietokových senzorov, anemometrov a podobných senzorov, ktorých výstupom je práve určitý počet impulzov hovoriaci o zmeranej informácii.

Datalogger má na rozdiel od ostatných spomínaných zariadení zabudovaný display pre rýchli prístup k nameraným dátam, minimálnej a maximálnej hodnote a štatistikám o priemernej hodnote. Namerané hodnoty je možné užívateľom pomocou softwaru previesť na iné jednotky, ktoré priamo hovoria o nameraných dátach, ako napríklad liter za hodinu a podobne.

Toto zariadenie je vybavené len jedným snímacím vstupom. Rýchlosť čítania je možné nastaviť od 1 vzorky za sekundu až po jednu vzorku za 24 hodín. Maximálne napätie vstupného signálu je obmedzene na 30 V. Na dostupné úložisko dataloggeru je možné zaznamenať 16 128 vzoriek dát. Cena dataloggeru RFPulse2000A je až 210 EUR. Bližšie informácie je možné nájsť v [3]. Obrázok 4 zobrazuje zariadenie RFPulse 2000A.

Obrázok 4 Datalogger MadgeTech RFPulse 2000A [3]

3.2.4 Zhodnotenie riešení a uvedených dataloggerov

Výhodou použitia niektorého z dataloggeru dostupného na aktuálnom trhu je hlavne nepotrebný vývoj celého zariadenia z hľadiska spotrebiteľa. Na druhej strane sú zariadenia na zber dát v porovnaní s riešením, ktorým sa zaoberá táto práca, niekoľkokrát drahšie.

Ďalšou nevýhodou tohto riešenia je, že použitý datalogger nemusí spĺňať všetky stanovené požiadavky. Vývojom vlastného riešenia je možné navrhnúť a realizovať datalogger presne podľa stanovených požiadaviek a potrieb, ktoré konkrétne využitie, teda záznam dát zo vstupov a výstupov riadiacej jednotky plynového kotla vyžaduje.

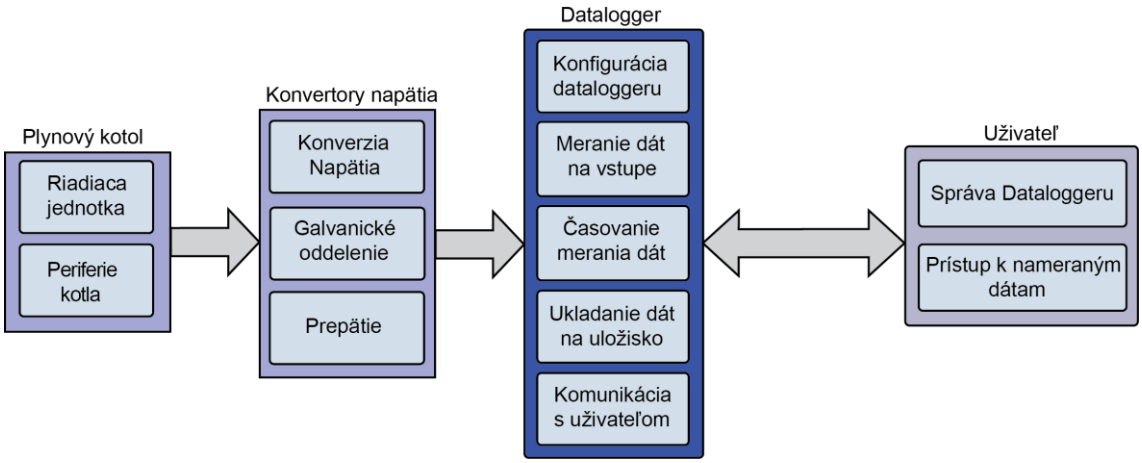
Tými najzaujímavejšími zariadeniami sú z hľadiska zberu dát z riadiacej jednotky DI – 160 a RFPulse 2000A. Datalogger DI – 160 ponúka zaujímavé možnosti a nastavenia vstupných portov. V budúcnosti by bolo vhodné implementovať do tejto práce podobnú konfiguráciu vstupov na snímanie stavu, udalostí a počítanie pulzov, čo by malo v konkrétnej aplikácií zberu dát z riadiacej jednotky plynového kotla veľké využitie.

Výhodou druhého dataloggeru RFPulse 2000A je hlavne vstavaný display a bezdrôtová komunikácia s užívateľským softwarom. Bezdrôtová komunikácia v tomto riešení dataloggeru by mohla byť ako jedna z možností komunikácie s užívateľom za podmienky výberu platformy, ktorá má USB port alebo priamo vstavaný Wi-Fi modul. Použitie Wi-Fi komunikácie však za predpokladu, že datalogger bude používaný aj v priemyslovom prostredí, nie je tou najbezpečnejšiu formou komunikácie.

4 ZÁKLADNÁ KONCEPCIA ZBERU DÁT Z PLYNOVÉHO KOTLA

Celkový výsledný koncept záznamu dát zo vstupov a výstupov riadiacej jednotky plynového kotla a popis funkčnosti jednotlivých blokov bude preberaný v tejto kapitole. Celý koncept môžeme rozdeliť na 4 základné bloky a to sú:

- Plynový kotol
- Napäťové konvertory
- Datalogger
- Užívateľ



Obrázok 5 Základná koncepcia zberu dát

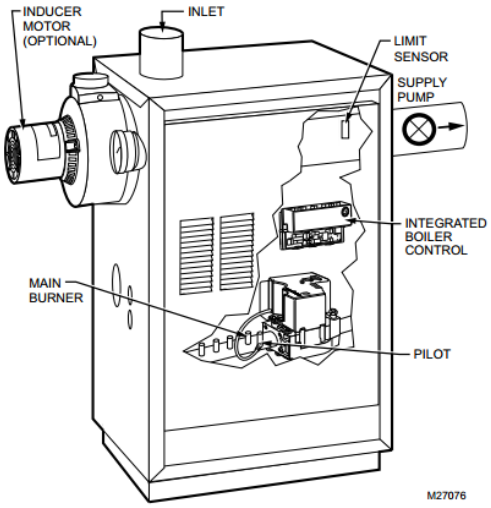
4.1 Plynový kotol

Plynový kotol je finálny celok, ktorého účelom je spaľovanie paliva a vyhrievanie vykurovacieho média cirkulujúceho vo vyhrievacom okruhu. Ako už zo samotného názvu vyplýva, palivom je v tomto prípade plyn privedený na vstup kotla.

Jednou z hlavných časti kotlov sú riadiace jednotky, ktorých vývojom sa zaoberá firma Honeywell Brno a ktorých periférie budú dataloggerom zaznamenávané. Riadiace jednotky plynového kotla používajú na riadenie spaľovania a celého kotla periférie, ku ktorým môžeme zaradiť všetky potrebné ventily na prívod plynu, rôzne kontrolné spínače, klapky v komíne a podobne. Podrobnejšie budú rozoberané tieto periférie a signály v ďalšej časti tejto kapitoly.

Z hľadiska riadenia plynového kotla považujeme za žiadanú hodnotu teplotu vo vykurovacom systéme, ktorá je nastavená v riadiacej jednotke. Spätná väzba je realizovaná niektorým zo senzorov teploty, ktorého výber závisí na konkrétnom kotle a aplikácií riadiacej jednotky.

Senzor teploty meria teplotu vykurovacieho média a zvyčajne sa jedná o klasický termočlánok. Regulácia v niektorých prípadoch môže byť realizovaná ako hysterézia. To znamená, že riadiaca jednotka pozná určenú hysteréznu hodnotu teploty, v ktorej rozmedzí ma spustiť vykurovací cyklus alebo naopak zastaviť vykurovací cyklus. Ak teplota vykurovacieho média dosiahne hodnotu súčtu hysteréznej hodnoty teploty a žiadanej teploty, dôjde k zastaveniu vykurovacieho cyklu. V druhom prípade ak teplota vykurovacieho média dosiahne hodnotu rozdielu žiadanej teploty a hysteréznej hodnoty teploty, vykurovací cyklus sa opäť spustí.



Obrázok 6 Plynový kotol [9]

Ako z vyplýva z obrázku 5, riadiaca jednotka je súčasťou plynového kotla. Ako už bolo spomenuté, riadiaca jednotka riadi plynový kotol a celé spaľovanie pomocou pripojených periférií. Spoločnosť Honeywell vyrába veľké množstvo riadiacich jednotiek plynových kotlov, ktoré na správnu funkčnosť potrebujú rôzne periférie, no ako príklad je použitá riadiaca jednotka plynového kotla s označením S936X ktorej oficiálny dokument je dostupný na odkaze [9]. Táto riadiaca jednotka má niekoľko signálov a periférií, ktoré by počas testovania mohli byť určené k zberu, no nie všetky sú zobrazené na obrázku 6. Ostatné periférie je možné vidieť v detailnejšom obrázku 7 použitej riadiacej jednotky.

Prvou periférioou riadiacej jednotky môže byť Inducer alebo Damper. Inducer je realizovaný ako elektrický motor, ktorý je použitý na vytvorenie správneho ťahu v komíne. V kotly sa teda jedná o ventilátor, ktorý rozháňa vzduch v komíne. Naopak Damper je realizovaný ako motor, ktorý vytáča klapku v komíne. Použitie obidvoch periférií je z hľadiska ich významu a podstaty zbytočné.

Ďalšou periférioou ktorá je pripojená k riadiacej jednotke je cirkulátor. Hlavnou úlohou cirkulátoru je rozháňanie vykurovacieho média v celom vyhrievacom okruhu. Cirkulátor je typicky realizovaný ako elektrické čerpadlo.

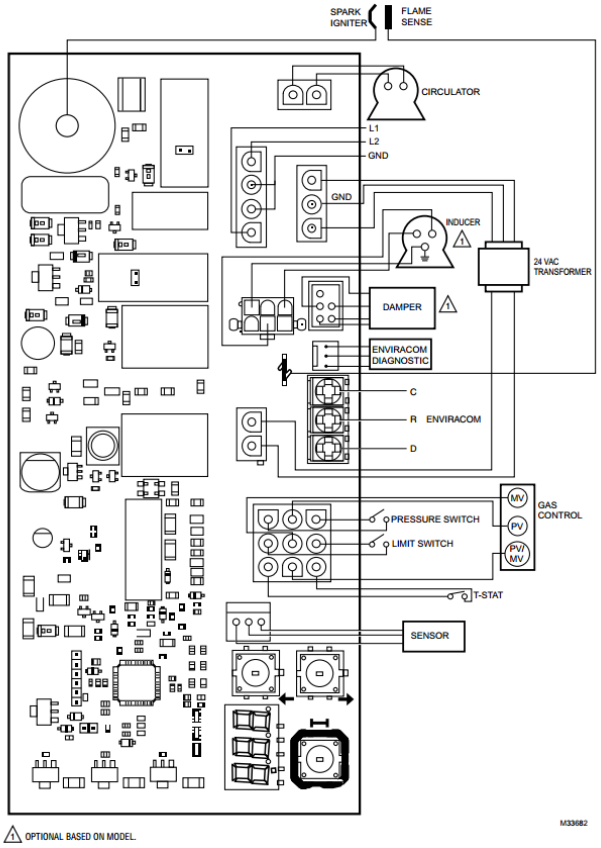
Najhlavnejšími perifériami riadiacej jednotky sú plynové ventily. Riadiaca jednotka S936X rozlišuje dva ventily a to sú pilotný ventil a hlavný ventil plynu. Tesne pred zapálením plynu sa otvorí pilotný ventil, ktorého úlohou je dodať dostatok plynu na jeho správne zapálenie plameňa v spaľovacej komore.

Poslednými perifériami sú riadiace vstupy riadiacej jednotky, ktoré sú realizované ako spínače. Prvým je spínač tlaku, ktorý zopnutím alebo rozopnutím kontroluje správny tlak v systéme. Ďalším vstupným spínačom je takzvaný limitný spínač, ktorého úlohou je kontrolovať prípadne prehriate celého kotla a zamedzenie spustenia vykurovacieho cyklu. Posledným vstupným spínačom je spínač označený na obrázku 7 ako T-STAT. Tento vstup riadiacej jednotky je vlastne požiadavka na začatie vykurovacieho cyklu, ktorá je odvodená od požadovanej a aktuálnej teploty v celom systéme.

4.2 Pripojenie periférií k dataloggeru

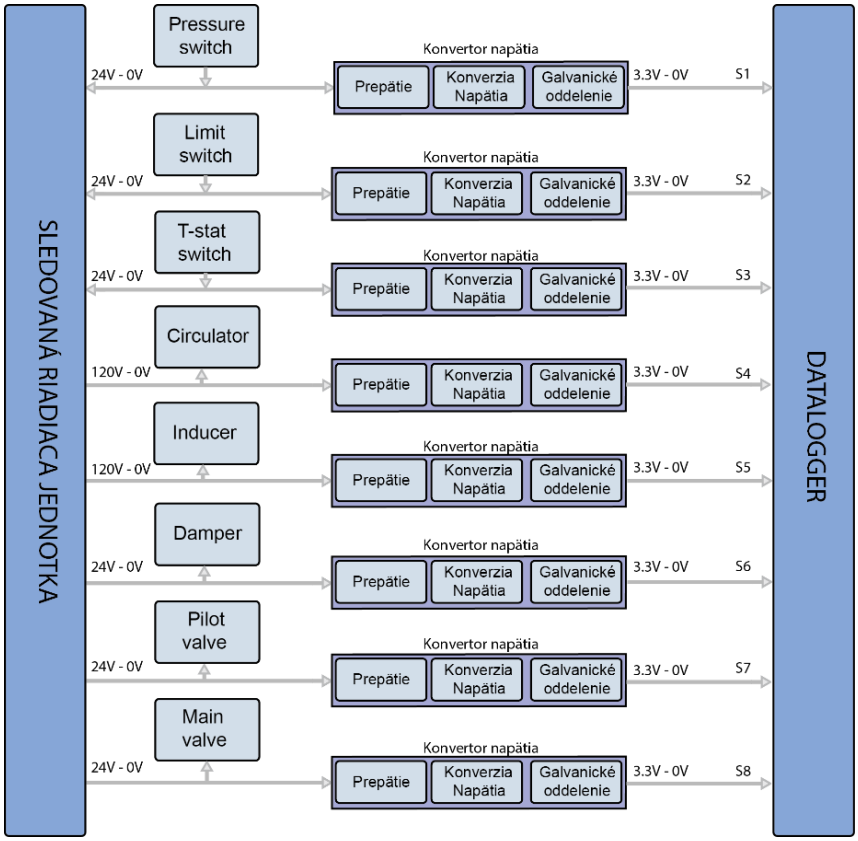
Na uvedenom obrázku 7 je zobrazený spôsob pripojenia jednotlivých periférií k riadiacim jednotkám typu S936X. Riadiaca jednotka uvedená ako príklad bola vyvíjaná pre Americký trh, z toho dôvodu sa budú úrovne riadiaceho napätia periférií od európskych noriem líšiť. V prvej časti je ukázané, ako sú jednotlivé periférie popísané v predchádzajúcej kapitole pripojené k riadiacej jednotke plynového kotla.

Klapka v komíne je riadená pomocou napätia 24 V z transformátora. Čerpadlo rozháňajúce vykurovacie médium riadiaca jednotka spína pomocou fázového napätia 120 V. Ak je použitý ventilátor v komíne, riadiaca jednotka ho taktiež spína pomocou fázového napätia. Obidva plynové ventily sú zvyčajne spínané pomocou úrovne napätia 24 V. Vstupy riadiacej jednotky ako je spínač tlaku, limitný spínač a podobne, majú zvyčajne úroveň napätia 24 V v závislosti od konkrétnej riadiacej jednotky plynového kotla.



Obrázok 7 Riadiaca jednotka S936X [9]

Na Obrázku 7 môžeme vidieť blokovú schému pripojenia periférii riadiacej jednotky popísanej v tejto kapitole s dataloggerom. Z tejto blokovej schémy je zrejme, že kanály dataloggeru budú pripojené paralelne k perifériám sledovanej riadiacej jednotky plynového kotla. Riadiaca jednotka plynového kotla, ktorá je použitá ako príklad ma 8 periférií, to znamená že datalogger bude snímať 8 vstupných kanálov. Signály, ktoré sú v tomto prípade privedené na datalogger sú označené ako S1 až S8. Toto označenie bude používané v nasledujúcom texte tejto práce.



Obrázok 8 Pripojenie snímaných periférií k dataloggeru

4.3 Konvertory napätia

Riadiaca jednotka plynového kotla zvyčajne priamo zopína pomocou napájacieho napätia prídavné periférie ako sú riadiaci ventil plynu, hlavný ventil plynu, klapka v komíne a podobne periférie, ktoré boli ako príklad uvedené a popísané v predošlej kapitole. Napájacie napätie spomínaných periférií je vo väčšine prípadoch niekoľkokrát vyššie ako maximálne napätie, ktoré je možné priviesť na vstupy použitej hardwarovej platformy.

Z tohto dôvodu je potrebné pred privedením signálu na snímané vstupy hardwarovej platformy, respektíve dataloggeru, ktorý nesie informáciu o stave snímanej periférie, úroveň tohto signálu správne upraviť. Konvertory napätia tak vytvárajú rozhranie medzi snímanými perifériami riadiacej jednotky plynového kotla a vstupmi dataloggeru.

Na túto úpravu úrovne sa používajú konvertory napätia. Správnym návrhom tohto konvertoru je možné z jeho výstupu priviesť napätie na vstup hardwarovej platformy bez toho, aby bol poškodený vysokým napätím. Úlohou konvertorov je taktiež galvanické oddelenie a ošetrenie stavu prepätia na jeho vstupe. Konvertory napätia, tak ako bolo spomenuté v rozbore zadania, budú dodané zadávateľom ako hotové riešenie, preto nie je potrebné tieto konvertory opäť vyvíjať.

Na vstup konvertorov je možné priviesť napätie s maximálnou úrovňou 250 V. Na konvertor napätia je potrebné priviesť signál, ktorý ma byť na výstupe zopínaný. Typicky ide o 5 V alebo 3.3 V. Keďže je vstup od výstupu galvanicky oddelený, nie je možné aby sa zopínaný vstup akokoľvek zničil.

4.4 Datalogger

Celý datalogger bude po hardwarovej stránke realizovaný ako hardwarová platforma podporujúca operačný systém Linux, ktorých ja na aktuálnom trhu mnoho. Datalogger bude schopný konfigurácie merania podľa požiadaviek zadávateľa popísaných v kapitole 2 tejto práce. Hlavnou funkciou firmwaru dataloggeru bude meranie dát v presne určených periódach vzorkovania. Tieto periódy vzorkovania musia byť tvorené vo firmware dataloggeru a dodržiavané na základe konfigurácie od užívateľa, s ktorým bude prebiehať komunikácia.

Aby bolo možné dlhodobo zaznamenávať dáta a po ukončení záznamu tieto dáta ďalej spracovať a vyvodiť závery vykonaných testov riadiacej jednotky, bude potrebné vytvoriť úložisko nameraných dát. Konkrétny návrh a riešenie úložiska a všetkých týchto potrebných častí dataloggeru bude popísaný v kapitole číslo 6 tejto práce.

4.5 Užívateľ

Užívateľom sa v tomto ponímaní myslí zamestnanec, ktorý bude mať prístup k všetkým dátam a bude s dataloggerom pracovať. Užívateľ bude na prácu s dataloggerom podľa požiadaviek zadávateľa používať webovú aplikáciu, ktorej primárnou úlohou bude konfigurácia dataloggeru a prezentácia nameraných dát. Užívatelia budú rozdelený do troch rolí, Administrátor, Tester a obyčajný užívateľ. Konkrétne prístupové práva a ich významy pre jednotlivé užívateľské role budú popísane v ďalších kapitolách.

5 VOLBA HARDWARU PRE DATALOGGER

Ako bolo už niekoľkokrát spomenuté, hlavným cieľom tejto práce je návrh a realizácia firmwaru pre datalogger riadiacej jednotky plynového kotla. Z časti je ale potrebné venovať pozornosť aj návrhu, respektíve voľbe hardwaru dataloggeru, aby bolo jasné, aké hardwarové platformy je možné použiť na správnu funkčnosť firmwaru pre datalogger a hlavne stanoviť platformu, na ktorej bude datalogger realizovaný.

5.1 Výber vhodnej platformy pre datalogger

Jednou z požiadaviek zadávateľa bolo, aby bol firmware dataloggeru v rámci možností čo najviac univerzálny a prenositeľný medzi podobnými hardwarovými platformami, ktoré budú spĺňať požiadavky popísané v tejto časti. Tieto požiadavky budú popisované ako všeobecne požiadavky na vhodnú platformu.

Výber hardwarovej platformy je z veľkej časti daný požiadavkou zadávateľa na operačný systém Linux. Na aktuálnom trhu je možné nájsť veľké množstvo platforiem, ktoré majú podporu tohto operačného systému.

Na druhej strane má použitie platformy s operačným systémom pre toto riešenie jednu nevýhodu. Operačný systém sa stará o priraďovanie času pre jednotlivé spustené procesy. Jedným z procesov jadra operačného systému bude aj meracia aplikácia, ktorej účelom bude vzorkovať snímané vstupy dataloggeru s danou periódou a ukladať ich na úložisko. Z tejto skutočnosti vyplýva, že pri návrhu firmwaru je nutné rátať s podmienkami, kedy jadro operačného systému priradí procesorový čas procesom s vyššou prioritou, čím môžu nastať komplikácie so vzorkovacími periódami.

5.1.1 Komunikačné periférie

Zvyčajne majú hardwarové platformy rôzne komunikačné periférie či už sa jedná o komunikácie na vyššej úrovni ako je USB a Ethernet alebo komunikačné zbernice ako sú SPI, I^2C a podobne. Ako bolo zadávateľom definované, v prípade tohto dataloggeru bude potrebné, aby komunikácia s užívateľom prebiehala pomocou webového rozhrania a webovej aplikácie. Z tejto požiadavky jasne vyplýva, že na realizáciu dataloggeru bude potrebná platforma, ktorá disponuje rozhraním pre Ethernet alebo priamo Wi-Fi modulom. Túto požiadavku je možné splniť napríklad aj v prípade použitia Ethernet alebo Wi-Fi adaptéru pripojeného do USB portu hardwarovej platformy.

5.1.2 Počet vstupov

Ďalším závažným parametrom pri výbere vhodnej platformy je počet pinov, ktoré je možné použiť ako vstup. Piny hardwarovej platformy, ktoré je možné použiť ako vstup, budú použité ako snímané vstupy dataloggeru.

Zadávateľom bol počet potrebných snímaných vstupov pre datalogger stanovený na 16, pričom je nutné rátať s budúcim rozšírením o ďalších 5 vstupov z čoho vyplýva, že je potrebné zvoliť platformu disponujúcu s minimálne 21 pinmi, ktoré je možné použiť ako vstup.

5.1.3 Výpočtový výkon

Minimálna perióda vzorkovania má taktiež vplyv na výkon zvolenej hardwarovej platformy. Ako bolo spomenuté v rozbore zadania, zadávateľom bola zvolená minimálna vzorkovacia perióda 100 milisekúnd. Pokiaľ budú zvolené najnižšie periódy vzorkovania na viacerých snímaných vstupoch, je možné očakávať nedostatok výkonu pri starších verziách platforiem, ktoré nedisponujú až tak veľkým výpočtovým výkonom. Tento problém môže mať za následok nesplnenie presností period vzorkovania a nedodržania ekvidistantného kroku pri vzorkovaní, čo je považované za nežiadúci stav pri zázname dát. Tento problém je možné riešiť dvomi spôsobmi.

Prvým spôsobom je pridanie ďalšieho jednoduchého mikrokontroléru do celého konceptu, ktorý by sa staral o záznam stavu vstupov, časového údaju, kedy bola vzorka vykonaná a odoslanie týchto informácií po určitej zbernici na vyššiu úroveň, ktorou by bola zvolená hardwarová platforma s operačným systémom Linux. Týmto riešením by bolo vyriešené dostatočne presné vzorkovanie dát, no na druhej strane by sa zvýšila zložitosť celého konceptu, s čím zadávateľ po návrhu tohto riešenia nesúhlasil.

Druhým riešením je zvoliť hardwarovú platformu, ktorá disponuje dostatočne vysokým výpočtovým výkonom na beh celého operačného systému a taktiež meracej aplikácie dataloggeru. Pri tomto riešení problému je taktiež potrebné rátať s občasným nedostatkom prideleného procesorového času pre meraciu aplikáciu, no výberom výkonnejšej platformy sa bude očakávať čo najvyššia eliminácia tohto problému. Toto riešenie bolo zvolené zadávateľom aj na úkor možných nepresností pri vzorkovaní vstupov dataloggeru.

5.1.4 Úložisko dát

Požiadavka zadávateľa na úložisko dát bola, aby dáta boli ukladané na pamäťovú kartu. Z tejto požiadavky je potrebné vychádzať a zvoliť hardwarovú platformu, ktorá disponuje slotom na pamäťovú kartu.

5.1.5 Platforma Raspberry Pi 2 model B

Raspberry Pi 2 je najnovšia generácia tejto hardwarovej platformy od firmy Raspberry Pi Fundation. Podporuje operačný systém Linux a jeho rôzne distribúcie. Najznámejšou a najpoužívanejšou distribúciou je Raspbian. Ďalej je možné použiť distribúcie ako sú Ubuntu, Fedora Arch Linux a podobne. Z hľadiska počtu vstupných pinov disponuje Raspberry Pi so 40 vstupno-výstupnými pinmi. Raspberry Pi 2 je vybavené 4 USB portami a jedným portom pre Ethernet. Z hľadiska výkonu sa táto platforma radí k výkonnejším na trhu a ponúka procesor ARM Cortex-A7 s taktom 900 MHz a pamäťou RAM s veľkosťou 1 GB, čo je z hľadiska použitia pre datalogger dostatočný výkon. Táto platforma je vybavená slotom na Micro SD pamäťové karty. Cena tejto hardwarovej platformy je podľa oficiálnych predajcov stanovená na 34,92 eur. Viac informácií a parametrov o tejto platforme je možné nájsť v [4].

Obrázok 9 Raspberry Pi 2 model B [4]

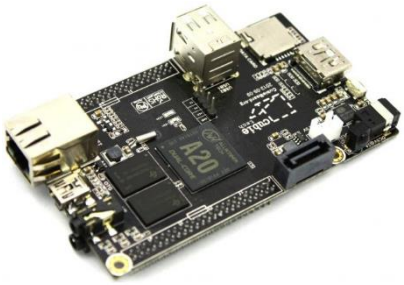
5.1.6 Platforma BeagleBone Black

Je to platforma vyrábaná firmou Texas Instruments v spolupráci s Digi-Key a Newark element 14. Tak ako spomínané Raspberry Pi a ostatné platformy tohto typu, podporuje rôzne distribúcie operačného systému Linux. Používa výkonnejší procesor ako predošlá platforma, ktorým je ARM Cortex-A8 s taktom 1 GHz a pamäťou RAM s veľkosťou 512 MB. Z hľadiska komunikačných portov disponuje BeagleBone Black jedným USB portom a jedným portom pre Ethernet pripojenie. Disponuje dvomi radami vstupno-výstupných portov, pričom jeden rad poskytuje až 40 portov. BeagleBone Black je vybavená, tak ako predchádzajúca platforma, portom pre Micro SD pamäťové karty. V porovnaní s Raspberry Pi 2 je táto platforma znateľne výkonnejši,a čo sa prejavuje na jej výslednej cene. Oficiálny predajcovia platformy BeagleBone Black uvádzajú cenu 46,79 eur. Bližšia špecifikácia a informácie o tejto platforme je dostupná v [5].

Obrázok 10 BeagleBone Black [5]

5.1.7 Platforma CubieBoard 2

Ani táto platforma nie je výnimka a tak ako ostatné spomenuté v tejto kapitole, podporuje rôzne distribúcie operačného systému Linux. V tejto platforme je použitý taktiež procesor ARM s jadrom Cortex-A7. Jedná sa o dvojjadrový procesor s dostatočným výkonom pre datalogger. Pamäť RAM má veľkosť, tak ako ostatné platformy 1 GB, čím sa táto platforma nijako nelíši. Z hľadiska komunikácie s okolitým prostredím je platforma vybavená dvomi USB portami a jedným portom pre Ethernet. Oproti ostatným platformám však vyniká v počte vstupno-výstupných pinov, ktorých je až 96. U oficiálneho predajcu je možné kúpiť túto platformu za 54,5 eur. Konkrétnejšiu špecifikáciu tejto platformy je možné nájsť v [6].



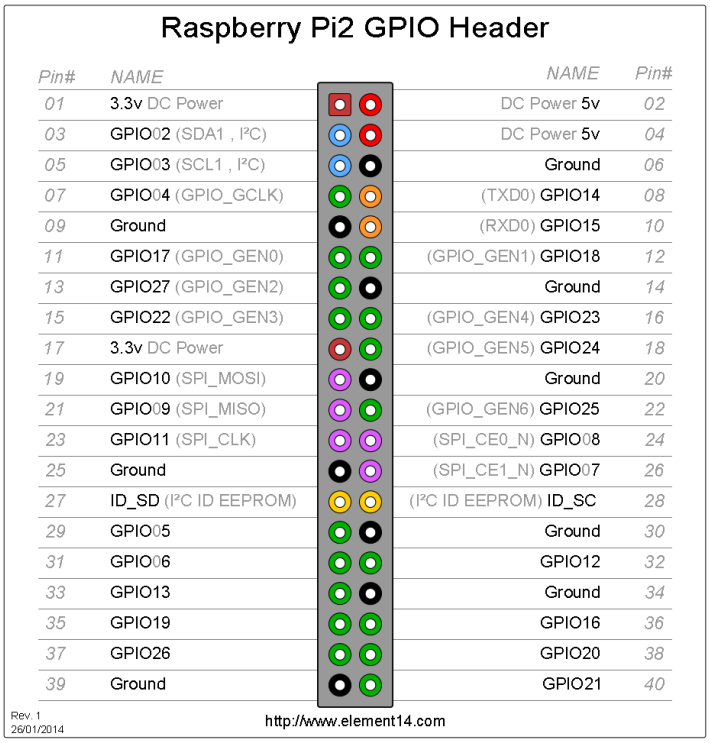
Obrázok 11 CubieBoard 2[6]

5.1.8 Zvolená platforma pre vývoj dataloggeru

Na vývoj a testovanie firmwaru pre datalogger bola zvolená platforma Raspberry Pi 2 model B. S touto platformou som sa stretol v minulosti a mám s ňou základné skúsenosti, čím bude možné aspoň z časti zrýchliť vývoj celého softwaru. Táto platforma má dostatočný počet vstupno-výstupných pinov, ktoré budú použité ako vstup dataloggeru, spĺňa podmienky z hľadiska komunikačných periférií. Hlavným dôvodom výberu tejto hardwarovej platformy je splnenie všetkých stanovených požiadaviek na hardware dataloggeru a jej najnižšia cena zo všetkých spomínaných platforiem.

5.2 Pomenovanie snímaných vstupov

Na realizáciu celého zadania bola zvolená platforma Raspberry Pi 2 model B. Táto platforma má celkovo 40 pinov, pričom všetky nie je možné použiť ako snímané vstupy pre datalogger. Je potrebné vybrať piny tejto platformy, ktoré je možné použiť ako vstup pre datalogger.



Obrázok 12 Usporiadanie pinov platformy Raspberry Pi 2 [7]

Na obrázku 12 je možné vidieť usporiadanie vstupno-výstupných pinov vybranej platformy. Všetky piny označené ako GPIO je možné použiť ako vstup. Niektoré z pinov majú špeciálne funkcie, no ak nie sú tieto funkcie použité, piny je možné použiť, tak ako všetky ostatné, ako vstup dataloggeru. Na vstupno-výstupné piny platformy Raspberry Pi je možné priviesť signál s maximálnou úrovňou napätia 3.3 V.

Všetky zvolené piny platformy Raspberry Pi 2, ktoré budú použité ako vstup pre datalogger budeme označovať v tomto riešení ako kanál s príslušným číslom. Zadávateľ požaduje 16 vstupov pre datalogger. Na základe tejto požiadavky boli vybrané tieto vstupné piny zvolenej platformy a budú považované za snímané vstupy dataloggeru, pričom ich označenie je definované ako kanál [číslo kanálu].

GPIO	Kanál číslo	Signál
2	1	S1
3	2	S2
4	3	S3
17	4	S4
27	5	S5
22	6	S6
10	7	S7
9	8	S8
11	9	Rezerva
5	10	Rezerva
6	11	Rezerva
13	12	Rezerva
19	13	Rezerva
26	14	Rezerva
14	15	Rezerva
15	16	Rezerva

Tabuľka 1 Výber vstupných pinov pre datalogger

Tabuľka 1 hovorí o konkrétnych pinoch platformy Raspberry Pi 2, ktoré budú požité ako vstup pre datalogger riadiacej jednotky plynového kotla. V tretom stĺpci tabuľky 1 je príklad pripojenia snímaných signálov z riadiacej jednotky plynového kotla použitej v kapitole 4.1 tejto práce ako príklad. To znamená, že signály S1 až S8 z obrázku 7 by boli po úprave riadiaceho napätia privedené na vstupy dataloggeru tak, ako je to v tabuľke 1. V konečnom riešení je užívateľ schopný zvoliť, ktorý vstupno-výstupný pin platformy Raspberry Pi bude predstavovať konkrétne kanály dataloggeru.

6 NÁVRH FIRMWARU PRE DATALOGGER

Návrh a realizácia sa v tejto časti delia na dva rôzne spôsoby prístupu k problému a tým je spôsob, kedy server, ktorý je potrebný pre beh webovej aplikácie a dátového úložiska je súčasťou samotného dataloggeru a spoločné s meracou aplikáciou tvoria samostatný blok a firmware dataloggeru.

Druhým riešením je vytvorenie hlavného servera, na ktorom je primárne realizované úložisko dát a webový server. V tomto prípade tvorí firmware dataloggeru len meracia aplikácia a databázový klient.

Z hľadiska návrhu je potrebné poznamenať, že na splnenie požiadaviek zadávateľa na operačný systém bude celý firmware dataloggeru fungovať pod operačným systémom Linux. Na obrázku 13 a 14 je možné vidieť blokové schémy, ktoré vyjadrujú obidva z týchto možných prístupov k realizácii firmwaru pre datalogger. Najprv budú popísané jednotlivé bloky použité v blokových schémach a následne diskutované obidva prístupy k návrhu a realizácií firmwaru pre datalogger.

6.1 Popis jednotlivých blokov

Funkčnosť jednotlivých blokov z obidvoch blokových schém sú totožné a preto budú v tejto kapitole popísané bez ohľadu na to, o ktorý spôsob návrhu a realizácie firmwaru ide. Z hľadiska rozdielov medzi obidvomi riešeniami ide len o umiestnenie podblokov v celkoch, ktoré je možné vidieť v blokových schémach ako Datalogger a Hlavný server, ktorých použitie tvorí hlavný rozdiel medzi týmito dvomi riešeniami. Obidve riešenia firmwaru pre datalogger sa skladajú z týchto hlavných blokov:

- Meracia Aplikácia
- Databázový server
- Databázový klient
- Webový server
- Webový klient

6.1.1 Meracia aplikácia

Meraciu aplikáciu dataloggeru je možné považovať ako za hlavnú časť celého firmwaru, ktorá vykonáva základné a zároveň tie najdôležitejšie funkcie, bez ktorých by nebolo možné splniť požiadavky pre správne fungovanie dataloggeru. Ako jediný blok má meracia aplikácia prístup k snímaným kanálom. Z tejto skutočnosti je zrejmé, že vytvára jediné rozhranie medzi snímanými kanálmi a celým firmwarom dataloggeru.

Jej primárnymi funkciami je v prvom rade snímanie vstupných kanálov dataloggeru v zvolených periódach vzorkovania, na ktoré sú privedené sledované signály a vytvorenie časovej značky v okamihu, kedy bola vzorka vykonaná. Meracia aplikácia musí byť schopná nakonfigurovať celé meranie.

Konkrétna realizácia konfigurácie bude popísaná v nasledujúcich kapitolách a z hľadiska pochopenia blokových schém na obrázku 13 a obrázku 14 nie je potrebné poznať konkrétne realizácie spomínaných funkcií.

Z hľadiska programovania je meracia aplikácia naprogramovaná v jazyku C. Tento jazyk bol vybraný hlavne z dôvodu rýchlosti programov, ktoré sú napísané v tomto jazyku, nakoľko je potrebné, aby bola meracia aplikácia čo najrýchlejšia a nebola zbytočne zložitá. Ak by v budúcnosti došlo k vyššej zložitosti meracej aplikácia kvôli vyšším nárokom na jej funkcie, bude potrebné zvažovať použitie niektorého z objektovo orientovaných programovacích jazykov ako je napríklad C++.

Jednou z hlavných úloh meracej aplikácie pre datalogger je správne časovať vzorkovacie periódy. Vzorkovacie periódy v tomto riešení je možné chápať ako časové intervaly, v ktorých bude meracia aplikácia merať potrebné dáta. Keďže na hardwarovej platforme bude použitý zvolený operačný systém, ktorý nemôžeme považovať za operačný systém reálneho času, je potrebné uvažovať o dolnej hranici vzorkovacích periód, ktorá bola stanovená v rozbore zadania na 100 ms. Ak by bola minimálna perióda vzorkovania ešte nižšia a bolo by potrebné striktne dodržať ekvidistantný krok, k zadaniu a riešeniu vzorkovacích periód by bolo potrebné pristúpiť iným spôsobom.

Zvolený operačný systém, tak ako všetky operačné systémy v tejto kategórií, nedokáže zaručiť vykonanie zvolenej úlohy v definovanom čase. Z tohto dôvodu nie je možné zaručiť, že meracia aplikácia bude vždy zaznamenávať dáta vo zvolenej perióde vzorkovania a dodržiavať ekvidistantný krok vzorkovania, čo podľa rozboru zadania nie je potrebné striktne splniť. Určite však bude vhodné, aby sa vzorkovacie periódy podľa možností čo najviac zhodovali so zvolenými periódami vzorkovania. Ak sa vzorkovacie periódy nebudú zhodovať s požadovanými periódami, užívateľ bude vedieť, že vykonaná vzorka nie je spoľahlivá a bola vykonaná o určitú dobu neskoršie, ako bolo požadované.

Tento problém bude potrebné čo najviac eliminovať a to niekoľkými spôsobmi. Zvoliť hardwarovú platformu s vyšším výpočtovým výkonom, meraciu aplikáciu naprogramovať s čo najnižšou zložitosťou a zaťažiteľnosť operačného systému obmedziť na minimum, poprípade použiť čo najjednoduchšiu distribúciu zvoleného operačného systému alebo konceptu firmwaru.

Na to, aby bolo možné vytvárať periódy vzorkovania dát, bude potrebné pre tento účel zvoliť a získavať referenčný čas v rozlíšení milisekúnd, od ktorého budú časované všetky vzorkovacie periódy. Tento čas bude porovnávaný s najbližším naplánovaným časovým údajom v ktorom má nastať meranie dát. V okamihu kedy tento čas nastane, dôjde k zmeraniu všetkých potrebných dát. Keď budú všetky dáta namerané, hneď ako to bude možné dôjde k naplánovaniu nového časového údaju, v ktorom má nastať ďalšie meranie a následne k uloženiu nameraných dát v správnom formáte na zvolené úložisko.

6.1.2 Databázový server

Z celkového konceptu zberu dát je zrejme, že namerané dáta je potrebné niekam dlhodobo uložiť. Už v požiadavkách zadávateľa na datalogger bolo spomínané, že datalogger bude mať dátové úložisko. Dôvod, prečo je tento blok nazvaný ako databázový server, bude popísaný v nasledujúcej časti.

Prvou myšlienkou na návrh dátového úložiska pre namerané dáta bol obyčajný textový súbor, ktorý by bol vytvorený niekde na pamäťovej karte hardwarovej platformy. Do tohto súboru by pristupovala meracia aplikácia a zapisovala namerané dáta. Toto riešenie by však viedlo k o veľa vyššej zložitosti, či už z hľadiska meracej aplikácie a jej úlohy uloženia dát, alebo z hľadiska webovej aplikácie a jej vyčítania uložených dát z úložiska. Zo strany webovej aplikácie je možné očakávať, že namerané dáta bude potrebné na základe požiadaviek užívateľa napríklad triediť, vyberať len niektoré časti dát, ktoré budú pre užívateľa najzaujímavejšie a podobné operácie. Toto riešenie by v prípade takýchto operácií bolo takmer nepoužiteľné a viedlo by k ďalšiemu zvyšovaniu zložitosti riešenia ako firmwaru tak aj webovej aplikácie.

O veľa rozumnejším riešením dátového úložiska je použitie databázy, ktorej funkcie sú pripravené práve na používanie, ktoré by zvolením prvej metódy úložiska bolo takmer nemožné a viedlo by k zbytočne komplikovaným riešeniam. Všetky funkcie ktoré bude potrebné použiť napríklad na uloženie, selektovanie dát a podobne, fungujú v databázach ako presne definované príkazy v jazyku SQL. Ďalším z mnoho dôvodov zvolenia tohto typu úložiska je, že databáza dokáže rozlišovať dátové typy, ktoré sú do nej vkladané. Databáza funguje ako databázový server, na ktorý je možné pripájať sa z ostatných klientov, ktorý práve potrebujú prístup k dátam.

Konkrétna databáza, ktorá je použitá ako úložisko nameraných je v tomto prípade MariaDB. Jedna sa o odnož veľmi známej databázy MySQL pričom funkcionalita ktorá je použitá na účel úložiska dát pre datalogger je navzájom úplne totožná. Viacej informácií a bližšej špecifikácií o databáze MariaDB je možné nájsť v [8].

6.1.3 Databázový klient

V tomto použití databázy, ako už bolo spomenuté predtým, je potrebné zo strany meracej aplikácie implementovať vytvorenie a zápis všetkých nameraných dát do databázy a zo strany webovej aplikácie bude potrebná implementácia vyčítania a selektovania nameraných dát.

Vo väčšine prípadov použitia databázy, tak ako je tomu aj v týchto riešeniach firmwaru, je potrebné riadiť databázu z inej aplikácie a požadované príkazy nie je možné zadávať priamo databáze. Z tohto dôvodu existujú knižnice, pomocou ktorých je možné naprogramovať zdrojové kódy v rôznych programovacích jazykoch, ktoré budú súčasťou potrebnej aplikácie a budú riadiť databázu a vykonávať potrebné príkazy.

6.1.4 Webový server

Zadávateľ na počiatku žiadal, aby bola komunikácia s užívateľom realizovaná pomocou webovej aplikácie. Na to, aby bolo možné pristupovať k webovej aplikácií zo strany užívateľa, je potrebné uvažovať v celom koncepte blok webového serveru, na ktorom sú dostupné zdrojové súbory webovej aplikácie. V tomto riešení je webový server realizovaný ako aplikácia bežiaca pod operačným systémom na hlavnom serveri.

Hlavnou úlohou aplikácie webového servera je vykonávanie požiadaviek od klientov, respektíve užívateľov webovej aplikácie pre datalogger. Vykonávanie požiadavky znamená odoslanie stránky a dát, o ktoré si užívateľ požiada.

Na to, aby boli dáta prezentované užívateľovi je potrebné pristupovať do dátového úložiska, v našom prípade do spomínanej databázy. Prístup do databázy musí prebiehať zásadne na strane serveru. Ak by prístup k dátam v databáze prebiehal až na strane klienta, respektíve užívateľa webovej aplikácie dataloggeru, užívateľ by bol schopný meniť všetky dotazy a príkazy smerované na databázu, čím by narušil bezpečnosť dát. Z tohto dôvodu musí byť prístup k dátam realizovaný na strane webového serveru. Webové stránky, ktoré server odosiela smerom k užívateľovi obsahujú len výsledok dotazov do databázy. Spomínaný prístup je realizovaný v skriptovacom programovacom jazyku PHP, ktorý je v blokových schémach na obrázku 13 a 14 reprezentovaný blokom PHP.

Existuje niekoľko riešení webových serverov, ktoré by bolo vhodné použiť, pričom server vybraný pre tento účel je aplikácia webového servera s názvom Apache. Ide o najrozšírenejšie riešenie v oblasti webových serverov. Výber práve tohto riešenia má niekoľko výhod, no tou najdôležitejšiu pre zadávateľa je open source software tohto webového serveru. Z toho vyplýva, že nie je potrebné použiť žiadnu platenú licenciu.

6.1.5 Webový klient

Jedinou požiadavkou na užívateľa z hľadiska návrhu a realizácie firmwaru pre datalogger je, aby u neho bežal webový klient. Táto požiadavka vyplýva z modelu klient-server, pri ktorom server čaká na požiadavky vyžiadané klientom. Pod pojmom webový klient sa rozumie webový prehliadač ako napríklad Google Chrome, Mozilla Firefox a podobne. Hlavnou výhodou použitia tohto riešenia je, že užívateľ nepotrebuje inštalovať žiadnu aplikáciu pomocou ktorej by spravoval datalogger. Je potrebné použiť len webový prehliadač, ktorý je zvyčajne súčasť operačného systému počítača.

6.2 Datalogger ako samostatné zariadenie

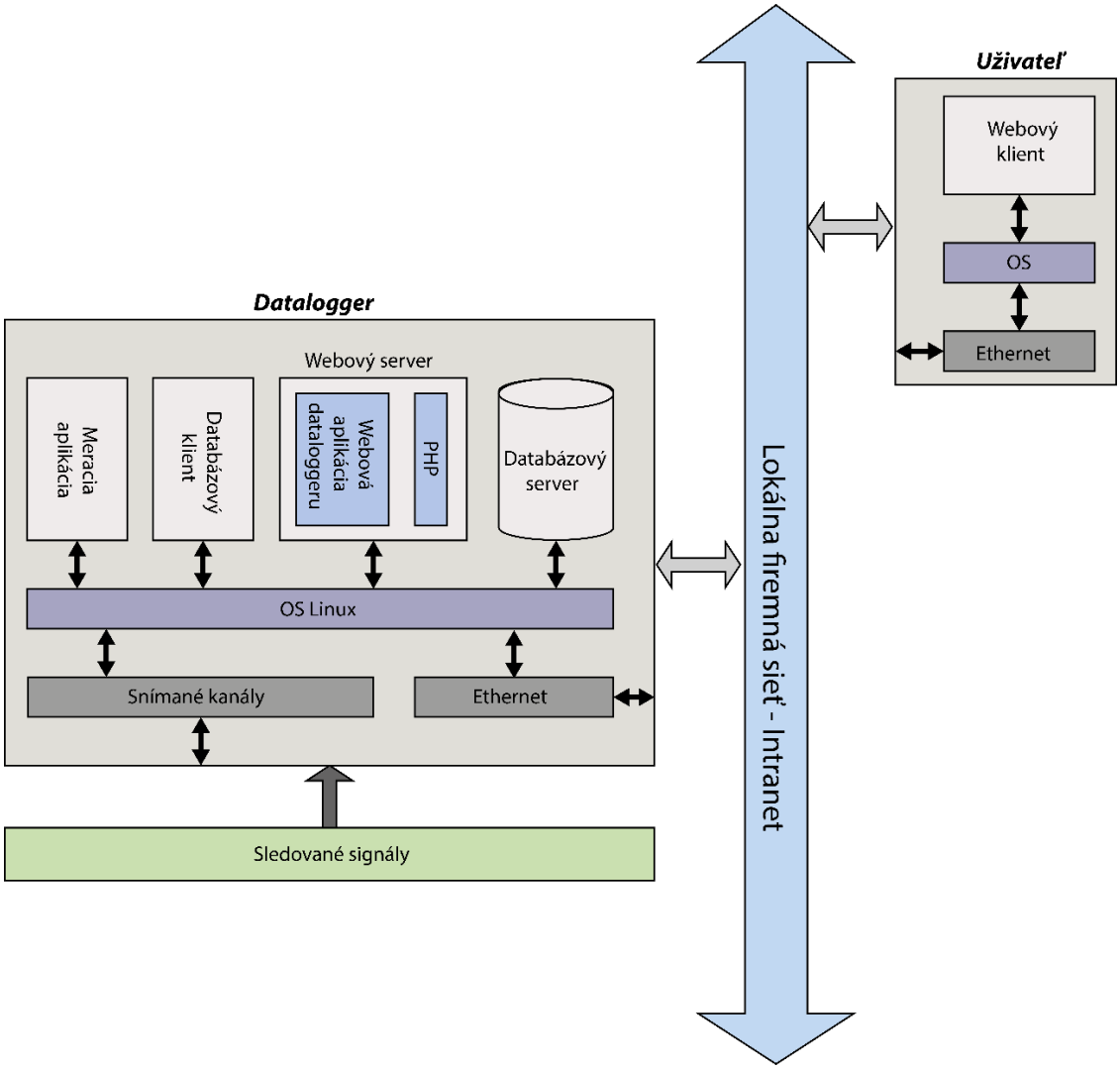
Tak, ako už bolo spomenuté na začiatku tejto kapitoly, prvý koncept firmwaru pre datalogger je tvorený dataloggerom ako jedným celkom v celom koncepte, na ktorom musí bežať všetko potrebné pre funkčnosť zberu dát z riadiacej jednotky plynového kotla podľa požiadaviek zadávateľa.

To znamená samotná meracia aplikácia, databázový klient na prístup k nameraným dátam v databáze, webový server a samotný databázový server. Z blokovej schémy tohto konceptu na obrázku 13 vyplýva, že webový a databázový server je spustený pod operačným systémom ako localhost na hardwarovej platforme dataloggeru. Toto riešenie má však množstvo nevýhod.

Hlavnou nevýhodou je niekoľkonásobne vyššie výpočtové zaťaženie dataloggeru, respektíve samotnej hardwarovej platformy určenej na realizáciu dataloggeru. Vybranú hardwarovú platformu Raspberry Pi 2 je síce možné zaradiť do výkonnejšej kategórie, no z hľadiska meracej aplikácie je potrebné, aby sa väčšina výpočtového výkonu sústredila na meraciu aplikáciu. Pri behu webového a databázového servera popri meracej aplikácií je však aj po spustení meracej aplikácie s najvyššou prioritou nemožné dosiahnuť zadávateľom požadovanú presnosť vzorkovania z dôvodu nedostačujúceho výkonu hardwarovej platformy na beh meracej aplikácie, aplikácie webového a databázového serveru.

Tento koncept firmwaru bol aj napriek predpokladaným problém otestovaný. Ako prvým testom sa meracia aplikácia spustila s tromi snímanými kanálmi a periódou vzorkovania 1 sekunda. Po skončení testu bolo vo všetkých nameraných vzorkách až 20 % vzoriek, v ktorých bol porušený ekvidistantný krok. V porovnaní s výslednými testami druhého konceptu, ktorý bol realizovaný je tento koncept firmwaru nevhodný a jeho presnosť je o veľa nižšia. Je možné predpokladať, že so zvyšovaním počtu kanálov by sa chyba ekvidistantného kroku ešte viac zvyšovala, čo je pri 10 násobku najmenšej periódy vzorkovania nevhodné.

V porovnaní s návrhom firmwaru pre datalogger, kde je webový a databázový server spustený na samostatnom zariadení a tvorí samostatný blok konceptu, je toto riešenie o niečo jednoduchšie. Nie je potrebné realizovať hlavný server konceptu a model klient-server je možné vytvoriť jednoduchším spôsobom ako je napríklad len prepojenie užívateľovho počítača s dataloggerom pomocou ethernetového rozhrania. Z hľadiska nameraných dát však užívateľ môže pristupovať a manipulovať len s dataloggerom, ku ktorému je pripojený. To znamená, že dáta nebudú centralizované na spoločnom úložisku dát. Každý z používaných dataloggerov má v tomto koncepte svoje úložisko dát, respektíve vlastnú databázu.



Obrázok 13 datalogger ako samostatné zariadenie

Keďže hlavnou prioritou zadávateľa je čo najpresnejšie vzorkovanie snímaných kanálov dataloggeru, tento koncept firmwaru pre datalogger bol po prezentácii výsledkov zamietnutý a v konečnej realizácií je použitý druhý koncept firmawaru popísaný v nasledujúcej kapitole, aj napriek zmenám hlavne na požiadavky zadávateľa na úložisko nameraných dát, ktoré malo byť pôvodne realizované na pamäťovej karte dataloggeru.

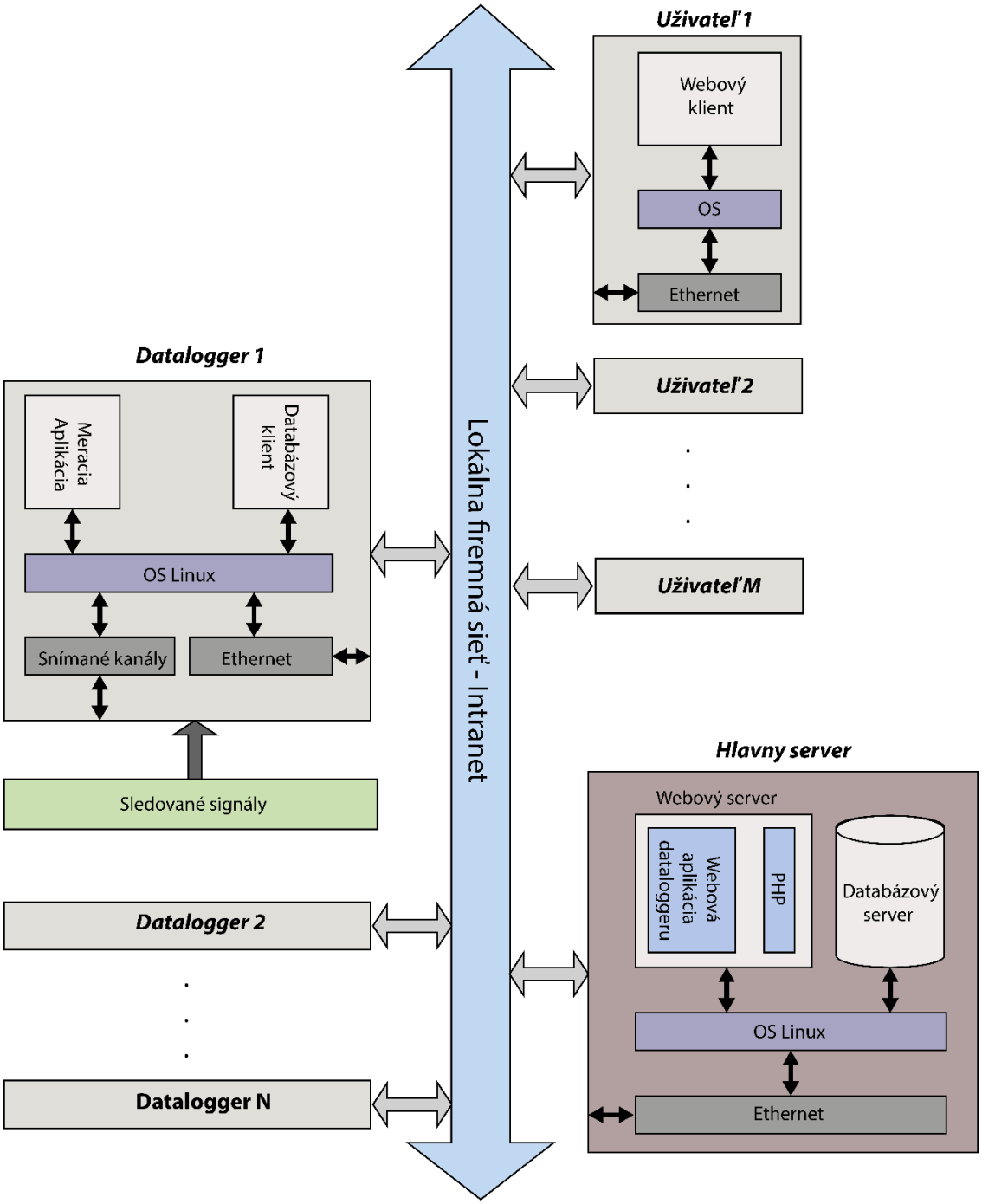
6.3 Datalogger s hlavným serverom

V porovnaní s prvým riešením firmwaru pre datalogger sa tento koncept odlišuje zásadne a to hlavne v pridaní ďalšieho bloku hlavného serveru do celého konceptu. Ako môžeme vidieť na obrázku 14, na samotnom dataloggeri je v tomto koncepte spustená len meracia aplikácia s databázovým klientom. To znamená že firmware bežiaci na dataloggeri je tvorený len meracou aplikáciou a databázovým klientom.

Na hlavný server, ktorý môže byť realizovaný napríklad ako jeden z počítačov v sieti, sa presunul webový a databázový server. Týmto spôsobom je zjednodušený firmware bežiaci na dataloggeri, čo má za následok využitie väčšej časti výpočtového výkonu hardwarovej platformy dataloggeru na beh meracej aplikácie a zároveň zvýšenie presnosti vzorkovania snímaných vstupov dataloggeru.

Pridaním hlavného serveru do pôvodného konceptu vznikol v podstate decentralizovaný systém, kde všetky namerané dáta a informácie o dátach sú ukladané na jedno dátové miesto. Týmto spôsobom je možné na rozdiel od prvého konceptu firmwaru pristupovať k nameraným dátam na jednom mieste, pomocou webovej aplikácie bežiacej na webovom serveri. Každý použitý datalogger sa pripája na hlavný server, získava potrebné informácie a zapisuje namerané vzorky do databázy a webová aplikácia má prístup ku všetkým dátam od všetkých dataloggerov. V tomto prípade začína webová aplikácia nadobúdať charakter informačného systému, ktorý zahŕňa všetky používané dataloggeri v lokálnej sieti vo firme. Konkrétne riešenia týchto funkcií budú preberané v ďalších kapitolách.

V porovnaní s prvým konceptom firmwaru má toto riešenie o veľa nižšie nároky na výpočtový výkon hardwarovej platformy použitej na realizáciu dataloggeru. Pridaním hlavného, respektíve centrálneho serveru do konceptu firmwaru vznikol globálny a decentralizovaný systém pre zber dát z riadiacich jednotiek plynových kotlov, ktorý má potenciál fungovať v celej firme a zaznamenávať dáta z ktorýchkoľvek riadiacich jednotiek plynových kotlov pri rôznych testovaniach. Užívateľ, ktorý bude pripojený na webový server a bude používať webovú aplikáciu má v tomto riešení prístup ku všetkým dataloggerom, ktoré sú alebo boli použité na zber dát z riadiacich jednotiek plynových kotlov. Ak by bol použitý pôvodný návrh firmwaru pre datalogger, užívateľ by mal v takom prípade prístup len k nameraným dátam jedného dataloggeru, na ktorého webovom servery by bol aktuálne pripojený.



Obrázok 14 Datalogger s hlavným serverom

Za jedinú nevýhodu tohto konceptu v porovnaní s pôvodným konceptom je možné považovať nutnosť realizácie hlavného serveru. Realizácia serveru môže viesť k celkovému zvyšovaniu nákladov na realizáciu, no nakoľko toto riešenie prináša množstvo výhod, po prezentácií zadávateľovi bol na konečnú realizáciu zvolený práve tento koncept firmwaru pre datalogger riadiacich jednotiek plynového kotla.

6.4 Konfigurácia merania dát

Jednou z dôležitých požiadaviek stanovených na začiatku práce bola možná konfigurácia dataloggeru zo strany užívateľa. Keďže meracia aplikácia sa bude starať o celé meranie dát, je potrebné brať do úvahy, že primárne parametre merania budú na začiatku celého procesu záznamu dát ovplyvňované zo strany užívateľa. Za primárne parametre merania je možné považovať počet všetkých snímaných vstupov, periódy vzorkovania priradené ku konkrétnym vstupom a konkrétne vstupy hardwarovej platformy, na ktoré je privedený zaznamenávaný signál. Všetky ostatné údaje ako je názov vstupu, jeho význam a ďalšie údaje budú riešené len vo webovej aplikácií nakoľko na správne fungovanie meracej aplikácie nie sú potrebné a ich zahrnutím by sa jeho zložitosť len zbytočne zvyšovala.

Prvým návrhom konfigurácie dataloggeru bola konfigurácia pomocou konfiguračného súboru vygenerovaného webovou aplikáciou na základe požiadaviek na konfiguráciu zo strany užívateľa. Po vygenerovaní správnej konfigurácie by bol tento súbor na začiatku celého procesu záznamu dát predložený a spracovaný meracou aplikáciou na základe presne stanovených pravidiel a informácií, ktoré by konfiguračný súbor v prípade jeho použitia obsahoval.

Po rozhodnutí použitia databázy v celom koncepte by však tento spôsob bol zbytočne komplikovaný. Z tohto dôvodu je databáza bežiaca na hlavnom serveri využitá ako na ukladanie nameraných dát, tak na vytváranie konfigurácií pre datalogger. Konkrétne riešenie a detailný popis bude v kapitole 7.1 zaoberajúcej sa databázovým modelom bežiacim na databázovom serveri.

7 REALIZÁCIA DATALOGGERU

V nasledujúcej kapitole je popísaná realizácia a implementácia jednotlivých časti firmwaru pre datalogger riadiacej jednotky plynového kotla. Aby bol vývoj firmwaru čo najjednoduchší, prebiehal z veľkej časti na osobnom počítači s operačným systémom Linux, pričom stavy na snímaných vstupoch boli softwarovo simulované pomocou generovania náhodného čísla 0 alebo 1. Pri programovaní boli používané knižnice, ktoré sú podporované na ktorejkoľvek distribúcií Linuxu. Použitím knižníc, ktoré spadajú pod normy Linuxových operačných systémov, bolo možné firmware bez väčších problémov preniesť a spustiť na vybranej hardwarovej platforme Raspberry Pi 2.

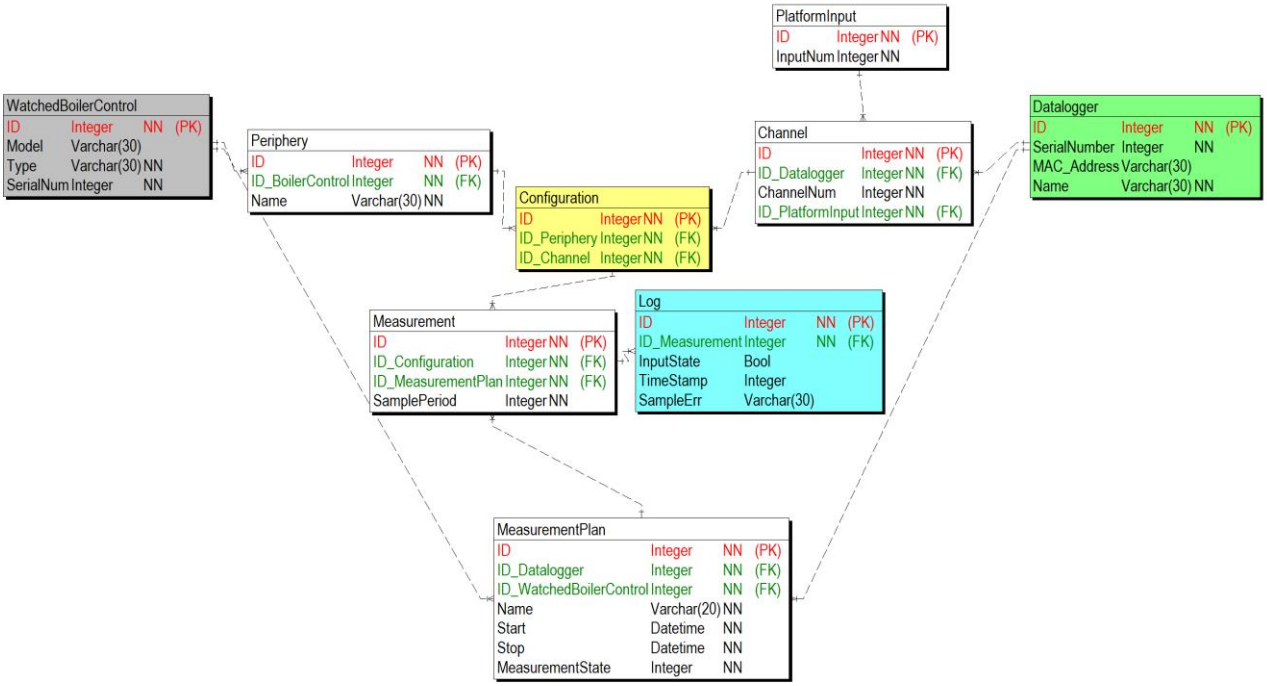
7.1 ER diagram databázy

ER Diagram z anglického výrazu Entity Relationship Diagram sa používa na modelovanie a znázornenie dátového modelu. Na základe namodelovaného dátového modelu sa v databáze vytvoria tabuľky pričom každá tabuľka obsahuje vlastné atribúty. Atribúty tabuľky po vytvorení databázy predstavujú názvy stĺpcov, ktoré realizovaná tabuľka obsahuje. Medzi tabuľkami v jednej databáze môžu vznikať väzby, ktoré sa v tomto type modelovania nazývajú relácie. Relácie popisujú vzťah tabuliek medzi sebou. V použitom ER Diagrame sú realizované dva druhy relácií a to 1:N a M:N.

Každá tabuľka vlastní atribút ID z dôvodu unikátnosti záznamu, ktoré obsahuje. Ide o dátový typ INTEGER, ktorý sa po vložení nového riadku do tabuľky vždy inkrementuje, čím je zaistená unikátnosť každého záznamu. Tento atribút je nazývaný primárny kľuč tabuľky. Ak vzniká relácia medzi dvomi tabuľkami, v druhej tabuľke sa vytvorí nový atribút, ktorý v podstate obsahuje hodnotu ID z prvej tabuľky. Takto vytvorený atribút sa nazýva cudzí kľuč.

Na začiatku realizácie firmwaru bolo potrebné vytvoriť čo najpresnejší dátový model celého konceptu firmwaru, ktorý je realizovaný v databáze bežiacej na hlavnom serveri. Výsledný dátový model ktorý je popísaný ER diagramom je možné vidieť na obrázku 15. Vytvorený dátový model je možné rozdeliť do piatich základných logických celkov, ktoré sú v tejto kapitole podrobne popísané:

- Datalogger
- Sledovaná riadiaca jednotka
- Konfigurácia
- Merací Plán
- Tabuľka nameraných dát – Log



Obrázok 15 ER diagram

7.1.1 Dátový model dataloggeru

Prvým a zároveň najhlavnejším logickým celkom ER diagramu je dátový popis dataloggeru. Nakoľko bol zvolený koncept firmwaru s hlavným serverom, kde databáza slúži ako centrálne dátové úložisko pre všetky dataloggeri, je potrebné ich navzájom odlišovať. Z tohto dôvodu musel v ER diagrame vzniknúť model popisujúci datalogger. Model dataloggeru sa skladá z tabuľky ***Datalogger***, ***Channel*** a ***PlatformInput***.

Prvým atribútom tabuľky Datalogger je INTEGER s názvom *SerialNumber* popisujúci sériové číslo každého dataloggeru nakoľko koncept firmwaru predpokladá použitie viacerých dataloggerov.

Atribút *MAC_Address* v tabuľke ***Datalogger*** predstavuje MAC adresu dataloggeru. Na základe tohto atribútu je schopný každý datalogger zistiť svoje unikátne ID a následne ID meracieho plánu, ku ktorému ho užívateľ pridelil. Detaily konfigurácie dataloggeru budú popísane v kapitole zaoberajúcej sa samotnou meracou aplikáciou dataloggeru.

Posledným atribútom tabuľky datalogger je jeho názov. Tento atribút je typu VARCHAR, čo znamená reťazec znakov. Významom atribútu *Name* je hlavne prehľadnosť vo webovej aplikácií pri prezentovaní dát, nakoľko napríklad sériové číslo môže byť pre užívateľa ťažšie zapamätateľné.

Ako už z celého konceptu a zadania vyplýva, každý datalogger má svoje snímané kanály. Kanál dataloggeru je reprezentovaný tabuľkou ***Channel***. Atribút tabuľky *ChannelNum* je typu INTEGER a špecifikuje, o ktorý kanál dataloggeru ide. Atribút *ChannelNum* môže nadobúdať hodnoty 0 až 15 práve kvôli stanoveniu počtu kanálov zadávateľom. Ďalšími atribútmi tabuľky popisujúcej kanál dataloggeru sú cudzie kľúče, ktoré vznikli reláciami medzi tabuľkou ***Datalogger*** a ***PlatformInput***.

Prvý cudzí kľuč *ID_Datalogger* bol vytvorení reláciou 1:N s tabuľkou ***Datalogger***. To znamená, že jeden datalogger môže mať N kanálov a naopak N kanálov môže byť priradených jednému dataloggeru. Vytvorením tejto relácie vznikne možnosť priraďovať existujúcemu dataloggeru nové kanály.

Druhým cudzím kľúčom v tabuľke ***Channel*** je *ID_PlatformInput* ktorý vznikol reláciou 1:N s tabuľkou *PlatformInput*. Tabuľka ***PlatformInput*** má zmysel len ako číselník. Obsahuje len jeden atribút *InputNum* typu INTEGER popisujúci vstup hardwarovej platformy Raspberry Pi 2, ktoré je možné použiť ako snímaný vstup dataloggeru. Tieto vstupy boli zvolené v kapitole 5.2 a konkrétne čísla vstupov hardwarovej platformy je možné vidieť v tabuľke 1.

7.1.2 Dátový model sledovanej riadiacej jednotky

Popis sledovanej riadiacej jednotky plynových kotlov je z hľadiska relácií medzi tabuľkami ***WatchedBoilerControl*** a ***Periphery*** takmer totožný s modelom dataloggeru. Prvým atribútom tabuľky ***WatchedBoilerControl*** je Model typu VARCHAR. Tento atribút predstavuje, tak ako atribút *Type*, popis riadiacej jednotky plynového kotla, ktorej signály je možné sledovať. Počas testovania sa častokrát môžu testovať riadiace jednotky rovnakého modelu. Z tohto dôvodu bol do tabuľky popisujúcej sledovanú riadiacu jednotku pridaný ďalší atribút typu INTEGER s názvom *SerialNum*, ktorého význam je vytvorenie jedinečnosti každej riadiacej jednotky, ktorá bude do databázy vložená.

Druhou tabuľkou popisujúcou riadiacu jednotku plynového kotla je ***Periphery***. S tabuľkou ***WatchedBoilerControl*** vytvárajú reláciu 1:N, to znamená, že jedna riadiaca jednotka má viacero periférií. Touto reláciou vznikol atribút *ID_WatchedBoilerControl*. Atribút *Name* tabuľky ***Periphery*** je typu VARCHAR a hovorí o názve periférie, ktorá je k riadiacej jednotke priradená. Tento model umožňuje pridávať novú riadiacu jednotku plynových kotlov a priraďovať k nej nové periférie.

7.1.3 Konfigurácia

Jednou z požiadaviek zadávateľa na datalogger je možnosť vytvárania konfigurácie dataloggeru. Konfigurácia dataloggeru spočíva v priradení snímaným kanálom dataloggeru jednotlivé periférie sledovanej riadiacej jednotky plynového kotla. Pod pojem konfigurácia možne z časti zaradiť aj periódy vzorkovania snímaných kanálov dataloggeru, no tento parameter bol zaradený do popisu meracieho plánu, čím je umožnené s zadávať meracie plány s rôznymi periódami vzorkovania.

Konfigurácia dataloggeru je realizovaná pomocou tabuľky ***Configuration*** v ER diagrame. Obidva atribúty tabuľky ***Configuration*** sú cudzie kľúče. Tak ako už z ich názvu vyplýva, *ID_Periphery* je cudzím kľúčom z tabuľky ***Periphery*** a *ID_Channel* predstavuje cudzí kľuč tabuľky ***Channel***.

Vložením jedného riadku do tejto tabuľky vytvoríme spojenie medzi konkrétnym kanálom dataloggeru a perifériou sledovanej riadiacej jednotky. Vytváranie konfigurácie predstavuje fyzické pripájanie dataloggeru k sledovanej riadiacej jednotky plynového kotla.

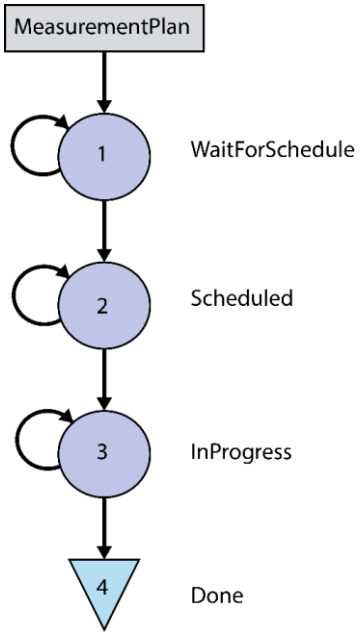
7.1.4 Merací plán

Merací plán predstavuje určitý časový úsek, počas ktorého datalogger zaznamenáva dáta na snímaných kanáloch. Záznam dát môže prebiehať vo viacerých cykloch bežiacich niekoľko hodín, poprípade dní. Z tohto dôvodu je v dátovom modeli tabuľka popisujúca merací plán s názvom ***MeasurementPlan***.

K meraciemu plánu môže byť priradený prvý atribút ktorý popisuje jeho meno. Atribút *Name* je typu VARCHAR, pretože obsahuje reťazec znakov. Z hľadiska zberu dát nie je tento atribút potrebný, ale jeho pridaním do tabuľky sa zvyšuje prehľadnosť užívateľa vo webovej aplikácií.

Správou tejto tabuľky v celom databázovom modeli je možné vytvárať meracie plány, ktoré sú definované ich začiatkom a koncom. Začiatok spustenia a koniec meracieho plánu sú definované ako dátový typ DATETIME. Zmyslom atribútov *Start* a *Stop* je vymedzenie časového úseku, v ktorom bude prebiehať merací plán.

Posledným atribútom tabuľky ***MeasurementPlan*** je stav meracieho plánu *MeasurementState* typu INTEGER. Merací plán sa môže nachádzať vo viacerých stavoch, ktoré informujú užívateľa o behu meracieho plánu. Na základe stavu meracieho plánu sa vyhodnocujú nasledujúce plány, podľa ktorých sa datalogger nakonfiguruje. Stavový popis tabuliek v dátovom modely sa nazýva životný cyklus. Na obrázku 16 je možné vidieť životný cyklus meracieho plánu. Prechody medzi jednotlivými stavmi sú definované pomocou tabuľky prechodov životného cyklu tabuľky ***MeasurementPlan***.



Obrázok 16 Životný cyklus tabuľky MeasurementPlan

Prechod	Akcia	Aktér
-1	Pridanie plánu	Web aplikácia
1 - 1	Nenaplánovaný plán	Web aplikácia
1 - 2	Naplánovanie plánu	Meracia aplikácia
2 - 2	Nenastal Start	Meracia aplikácia
2 - 3	Spustenie pánu	Meracia aplikácia
3 - 3	Beh plánu	Meracia aplikácia
3 - 4	Ukončenie merania	Meracia aplikácia

Tabuľka 2 Stavové prechody tabuľky MeasurementPlan

Po pridaní nového plánu do tabuľky ***MeasurementPlan*** má merací plán stav 1 prezentujúci, že meracia aplikácia dataloggeru nepozná merací plán s ktorým má začať pracovať. Meracia aplikácia bežiaca na dataloggeri porovnáva svoje ID z tabuľky dataloggerov s cudzím kľúčom *ID_Datalogger*. Ak nájde merací plán, ktorý jej patrí, vybraný plán prejde do stavu Scheduled. V prípade, že žiadny datalogger nenašiel merací plán, stav plánu ostáva nezmenený, teda WaitForSchedule. Stav Scheduled je zmenený na stav InProgress až vtedy, keď sa aktuálny čas rovná informácií o začiatku merania, v tomto prípade atribútu *Start*. V stave InProgress merací plán ostáva až po dobu skončenia merania. Po ukončení merania sa merací plán dostáva do konečného stavu, ktorým je stav Done.

Druhou tabuľkou popisujúcou merací plán je ***Measurement***. Až v tejto tabuľke sú obsiahnuté dáta, ktoré hovoria o perióde vzorkovania jednotlivých snímaných kanálov, respektíve periférií snímanej riadiacej jednotky plynového kotla. Pomocou cudzieho kľúča *ID_Configuration* je v tabuľke priradená perióda vzorkovania jednotlivým konfiguráciám. Atribút reprezentujúci periódu vzorkovania je nazvaný *SamplePeriod* a jeho dátovým typom je FLOAT, pretože perióda vzorkovania je do databázy vkladaná v sekundách a spodná hranica bola zadávateľom definovaná ako 0,1 sekundy. Druhý cudzí kľuč *ID_MeasurementPlan* hovorí o meracom pláne, v ktorom sú vzorkovacie periódy definované.

7.1.5 Namerané dáta

Na ukladanie nameraných dát je v dátovom modely vytvorená tabuľka ***Log***. Jej primárnymi atribútmi je *InputState* typu TINYINT a *TimeStamp*. TINYINT je najmenší možný dátový typ celého čísla použitého v databáze. Významom tohto atribútu je stav snímaného kanálu a môže nadobúdať hodnotu len 0 alebo 1.
Ako zo samotných požiadaviek zadávateľa vyplýva, každá vykonaná vzorka musí obsahovať udaj o čase, kedy bola vykonaná. Udaj o čase je prezentovaný atribútom *TimeStamp*.

Dátovým typom je BIGINT a to hlavne z dôvodu algoritmu vykonávania vzoriek meracej aplikácie. *TimeStamp* obsahuje počet milisekúnd od 1.1.1970 po čas, kedy bola vzorka vykonaná. Dôvod tohto formátu časového údaju bude vysvetlený v popise meracej aplikácie.

Posledným atribútom tabuľky nameraných dát je reťazec znakov, ktorý hovorí o dodržaní ekvidistantného kroku medzi periódami vzorkovania. Môže nadobúdať dve hodnoty LOG_OK a LATE_LOG_ERR. Prvá hodnota tohto atribútu hovorí, že vzorka bola vykonaná v správnom čase a nebol porušený ekvidistantný krok. Naopak LATE_LOG_ERR hovorí o posunutí vzorky v čase a jej vytvorení hneď ako to bolo možné, no krok vzorkovania nebol dodržaný.

7.1.6 Vytvorenie dátového modelu v databáze

Takto vytvorený dátový model je bolo potrebné spustiť na databázovom servery. Na vytvorenie ER diagramu bol použitý software s názvom Case Studio 2. Toto prostredie podporuje vygenerovanie skriptu v jazyku SQL, pomocou ktorého je možné vytvoriť a realizovať dátový model na servery. Ako príklad je možné vidieť vytvorenie tabuľky ***Datalogger***:

```
Create table Datalogger (
ID Int NOT NULL AUTO_INCREMENT,
SerialNumber Int NOT NULL,
MAC_Address Varchar(30),
Name Varchar(30) NOT NULL,
Primary Key (ID)) ENGINE = InnoDB;
```

7.2 Meracia aplikácia

Meracia aplikácia tvorí hlavnú časť firmwaru bežiaceho na dataloggeri riadiacej jednotky plynového kotla. Pri programovaní meracej aplikácie bolo potrebné dať dôraz na univerzálnosť a jednoduchosť zdrojového kódu. Modifikácia meracej aplikácie tohto dataloggeru môže nastať hlavne v type logovaných dát. To znamená, že v budúcnosti nemusí merať len logické stavy vstupov dataloggeru tak, ako je tomu teraz. Ak to bude potrebné, meracia aplikácia môže byť v budúcnosti rozšírená o meranie rôznych analógových hodnôt alebo digitálnych dát prijatých z niektorej zo zberníc. Z tohto dôvodu bol zdrojový kód programovaný čo najviac univerzálne a jeho riešenie a algoritmy bolo potrebné naprogramovať tak, aby sa minimalizovala závislosť na jednom aktuálnom riešení, čím je záznam logických stavov na snímaných vstupoch dataloggeru.

V meracej aplikácií sú použité viaceré knižnice, ktoré sú podporované na každom operačnom systéme Linux. Na to, aby sa meracia aplikácia mohla správať ako databázový klient, ktorý sa pripája na hlavný server, je potrebné nainštalovať balík knižníc príkazom:

sudo apt-get install libmysqlclient-dev

Meracia aplikácia pracuje s dvomi základnými dátovými typmi. Prvým je dátový typ struct, ktorého atribúty obsahujú všetky potrebné dáta z hľadiska meracej aplikácie a teda samotného dataloggeru. Riešenie sa môže zdať komplikované, no pri programovaní tento prístup umožnil zapuzdrenie všetkých potrebných dát do jedného celku, ktorým je štruktúra ***Datalogger*** a tak voliť parameter všetkých funkcií ako ukazovateľ na túto štruktúru.

```
typedef struct
{
     int iID;
     InputsData iInputs;
     MeasurementPlan iMeasurementPlan;
     DataloggerState iState;
} Datalogger;
```

Prvým atribútom štruktúry ***Datalogger*** je *iID* dataloggeru, ktoré bolo priradené dataloggeru pri jeho vložení do databázy. Pomocou ID dataloggeru získava meracia aplikácia všetky ďalšie informácie ako je napríklad meracie plány, počet snímaných kanálov a podobne.

Druhý premenná štruktúry ***Datalogger*** je štruktúra *InputsData*, ktorej prvým atribútom je premenná *iInputsNum*. Druhým atribútom je ukazovateľ na štruktúru popisujúcu jeden snímaný vstup dataloggeru, to znamená jeden pin platformy Raspberry.

```
typedef struct
{
   int iInputsNum;
   GPIOinput *iInputArray;
} InputsData;
```

Do premennej *iInputsNum* je uložený počet snímaných vstupov a ukazovateľ *iInputArray* obsahuje adresu pamäte, ktorá bola dynamicky pridelená pre snímané vstupy dataloggeru. Popis štruktúry *GPIOinput* bude popísaný v ďalšej samostatnej kapitole. Ďalším atribútom ktorý obsahuje štruktúra Datalogger je *iMeasurementPlan*. Táto štruktúra obsahuje premenne, ktoré patria meraciemu plánu.

```
typedef struct
{
   int iID;
   time_t iStart;
   time_t iStop;
} MeasurementPlan;
```

Ako už z dátového modelu popísaného v predchádzajúcej kapitole vyplýva, atribútmi meracieho plánu sú jeho *ID, Start* a *Stop*. Tieto informácie potrebuje meracia aplikácia na to, aby dokázala identifikovať merací plán, nakonfigurovať všetky snímané vstupy a merací plán začať a skončiť v zadanom čase.

V meracej aplikácií, respektíve v dataloggeri na ktorom beží meracia aplikácia, je implementovaný stavový automat popísaný v kapitole 7.2.2. Posledný atribút štruktúry Datalogger je stav, v ktorom sa práve datalogger, respektíve bežiaca meracia aplikácia nachádza.

Druhým dátovým typom ktorý je používaný ako parameter vo funkciách je štruktúra ***DataStorage***. Táto štruktúra obsahuje všetky premenné, ktoré je potrebné aby prebiehala komunikácia medzi databázovým serverom a klientom, teda meracou aplikáciou dataloggeru.

```
typedef struct
{
  MYSQL *iCon;
  const char *iHostName;
  const char *iUserName;
  const char *iPassword;
  const char *iNameDB;
  const char *iNameConfig;
  unsigned iPortNum;
  StorageErr iError;
} DataStorage;
```

Prvým a najdôležitejším atribútom tejto štruktúry je ukazovateľ *iCon* z knižnice mysql.h. Po jeho inicializácií sa vytvorí pripojenie na databázový server a všetky funkcie, ktoré pristupujú k databáze vyžadujú ako argument ukazovateľ *iCon*. Význam ostatných atribútov vyplýva z ich názvu.

7.2.1 Časovanie

Jednou z hlavných úloh meracej aplikácie je tvorenie vzorkovacích periód. Na to, aby bolo možné periódy správne vzorkovať, bolo potrebné nájsť správny algoritmus, ktorý bude fungovať dostatočne dobre a správne v celom zadávateľom definovanom rozmedzí vzorkovacích periód.

Prvou verziou časovania periód vzorkovania bolo použitie softwarových časovačov a signálov, ktoré je možné posielať medzi procesmi operačného systému. Pri použitý týchto časovačov je možné definovať presný časový interval pri ktorom má byť signál vyslaný. Tento spôsob sa z počiatku zdal ako správny. V konečnom dôsledku sa však ukázalo, že riešenie by bolo zbytočne komplikované, čo by zvyšovalo zložitosť celého firmwaru a meracej aplikácie.

Bolo potrebné vrátiť sa k pôvodnej myšlienke, ktorá bola popísaná v návrhovej časti meracej aplikácie. Celé časovanie meracej aplikácie a vzorkovacích periód bolo nakoniec postavené na časovej informácií, ktorá sa v Unixových systémoch nazýva ako Time since epoch. Ide o časový údaj ktorý hovorí o ubehnutom čase od referenčného dátumu 1.1.1970 a času 00:00. Tento údaj považujeme za referenčný čas meracej aplikácie, od ktorého sú odvádzane všetky periódy vzorkovania a na základe ktorého je vyhodnocovaný správny čas, pri ktorom má nastať nový záznam dát na snímanom vstupe. Funkcia pomocou ktorej je v meracej aplikácií získavaný časový údaj je z GNU knižnice time.h. Prototyp použitej funkcie:

```
int clock_gettime(clockid_t clk_id, struct timespec *tp)
```

Zdrojom časového údaju tejto funkcie je systémový čas operačného systému. Hardwarové platformy Raspberry Pi nedisponujú žiadnym RTC, teda zdrojom času ktorý nestráca informáciu o čase po strate napájania hardwarovej platformy, preto jedinou možnosťou zvolenia referenčného času bez použitia ďalšieho hardwaru bol systémový čas. Z tohto dôvodu je synchronizácia času realizovaná pomocou NTC. To znamená, že systémový čas sa synchronizuje len ak má Raspberry Pi prístup na internet. Druhou možnosťou synchronizácie času je užívateľské nestavenie pomocou spustenia konfigurácie raspi-config.

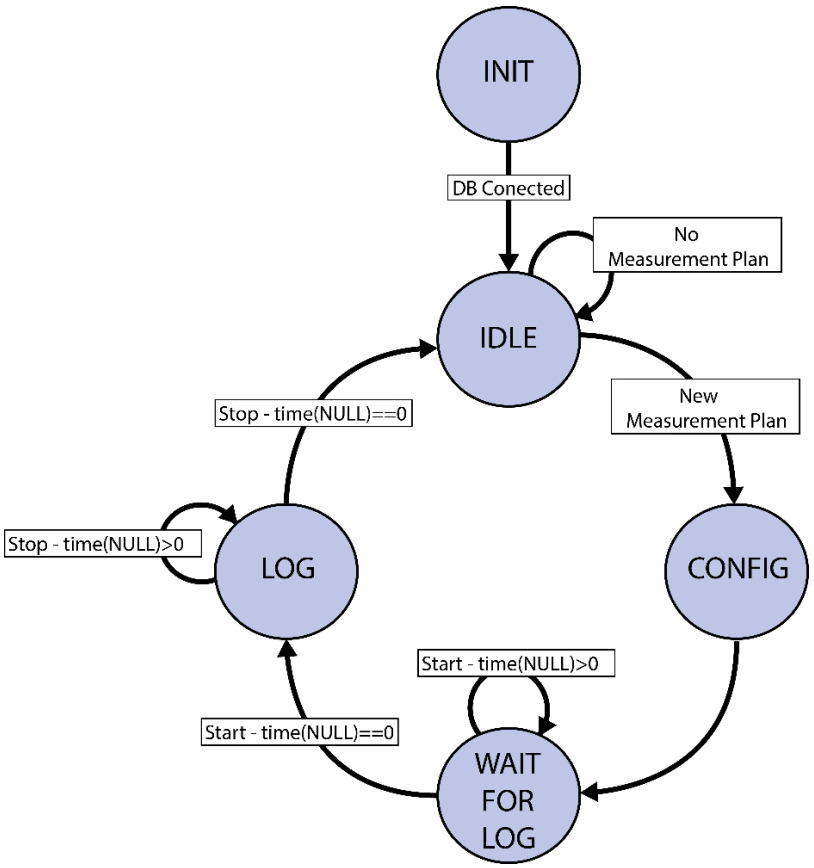
V meracej aplikácií je funkcia clock_gettime volaná s prvým argumentom ako CLOCK_REALTIME. Týmto argumentom je funkcií umožnené korigovať časový údaj podľa systémového času, čo však počas behu aplikácie môže vadiť. Je potrebné, aby bolo časovanie meracej aplikácie synchronizované so systémovým časom hlavne kvôli nastavovaniu času po odpojení dataloggeru z napájania. Užívateľovi počas používania nie je umožnená zmena systémového času a jedinou možnosťou jeho zmeny je počiatočné nastavenie dataloggeru, ktorú bežný užívateľ nebude vykonávať. Počas testovania meracej aplikácie bolo zistené, že synchronizácia času pomocou NTC zatiaľ nijako neovplyvnila periódy vzorkovania počas merania.

Druhým argumentom je ukazovateľ na štruktúru timespec, do ktorej je funkciou uložený časový údaj. Meracia aplikácia pracuje s časom v milisekundách, preto je potrebné časový údaj konvertovať do správnych jednotiek. Správny časový údaj je návratovou hodnotou funkcie s názvom EpochTime:

```
const epochTime_t EpochTime(void)
{
   struct timespec currentTime;
   clock_gettime(CLOCK_REALTIME, &currentTime);
   return(currentTime.tv_sec*1000LL+
   currentTime.tv_nsec/1000000LL);
}
```

7.2.2 Beh meracej aplikácie

Beh meracej aplikácie bol rozdelený do niekoľkých stavov. V celej meracej aplikácií sú implementované dva stavové automaty, pričom prvý stavový automat sa zaoberá behom celej meracej aplikácie. Druhý implementovaný stavový automat beží v rámci každého snímaného vstupu dataloggeru osobitne a bude popísaný v ďalších kapitolách realizácie meracej aplikácie. Hlavný stavový automat meracej aplikácie je možné vidieť na nasledujúcom obrázku 17.



Obrázok 17 Stavový automat meracej aplikácie

Prvým stavom meracej aplikácie je stav INIT, v ktorom sa meracia aplikácia pripája na hlavný server do databázy. Ak pripojenie prebehne správne, nasledujúci stav meracej aplikácie je stav IDLE. Ak sa na server nedokáže pripojiť, nastaví sa chybový stav a jej beh sa ukončí.

V stave IDLE je meracia aplikácia bez priradeného meracieho plánu. To znamená, že užívateľ datalogger, na ktorom beží meracia aplikácia nepoužíva a nepriradil mu nový merací plán. Na získanie ID meracieho plánu meracia aplikácia volá funkciu:

```
int GetMeasurementPlanID(DataStorage *aStorage,
                         Datalogger *aDatalogger)
```

Funkcia je pri behu meracej aplikácie volaná periodicky každých 10 sekúnd. Hlavnou úlohou tejto funkcie je získanie ID meracieho plánu, ktorý má v meracej aplikácií nasledovať. V tabuľke meracích plánov sa môže nachádzať niekoľko záznamov o meracích plánoch, ktoré sú v stave WaitForSchedule, teda pripravené pre meraciu aplikáciu. Funkcia GetMeasurementPlanID preto musí správne rozhodnúť, ktorý merací plán má začať skôr. Pomocou pripraveného SQL výrazu ktorý je predložený databázovému serveru je možné triediť meracie plány podľa atribútu Start použitím ORDER BY Start.

Návratovou hodnotou funkcie je v prípade nového meracieho plánu 0. ID nového meracieho plánu sa pomocou ukazovateľa na štruktúru Datalogger uloží do premennej iMeasurementPlan.iID a meracia aplikácia prejde do stavu CONFIG. Naopak, ak funkcia nenašla merací plán, ktorý má nasledovať, jej návratovou hodnotou je číslo menšie ako 0 a meracia aplikácia ostáva v stave IDLE.

V stave CONFIG meracej aplikácie sa naplnia všetky potrebné premenne štruktúry Datalogger. V tomto stave sú meranou aplikáciou volané dve funkcie:

```
int ConfigureDatalogger(DataStorage*aStorage,
                        Datalogger *aDatalogger)

int ScheduleMeasurementPlan(DataStorage*aStorage,
                            Datalogger*aDatalogger)
```

Funkcia ConfigureDatalogger v prvom kroku zistí počet snímaných kanálov, ktoré boli užívateľom nakonfigurované. Následne nastaví ukazovateľ iInputsData.iInputArray v štruktúre Datalogger na adresu, ktorej bola vyhradená pamäť na určený počet snímaných vstupov pre datalogger. V ďalšej časti zistí, ktoré konkrétne vstupy dataloggeru sa budú snímať a s akou periódou vzorkovania budú vzorky vykonávané. Týmto dátami nakoniec naplní pole vstupov, ktoré bolo predtým vytvorené dynamicky na základe počtu snímaných kanálov. Pomocou funkcie ScheduleMeasurementPlan sú získané informácie o začatí a skončení meracieho plánu, ktorého ID bolo zistené už v predchádzajúcom stave IDLE. Poslednou úlohou meracej aplikácie v stave CONFIG je zmeniť stav meracieho plánu z WaitForSchedule na Scheduled a nastaviť stav WAIT FOR LOG.

Ďalším stavom stavového automatu meracej aplikácie je stav čakania na začatie meracieho plánu. V predchádzajúcom stave bol merací plán naplánovaný. V stave WAIT FOR LOG datalogger čaká na čas, v ktorom má začať merací plán. Ak je rozdiel aktuálneho času a času v premennej iStart štruktúry i MeasurementPlan väčší ako 0, stav ostáva nezmenený. Ak sa tento rozdiel rovná nule, stavový automat meracej aplikácie prejde do stavu LOG.

Po prechode do stavu LOG sa zmení stav naplánovaného meracieho plánu na InProgress, čo znamená, že meracia aplikácia začala vykonávať záznam dát zo snímaných vstupov podľa meracieho plánu. V tomto stave je kontrolovaný atribút meracieho plánu Stop, aby sa meranie ukončilo v správny čas. Ak je rozdiel aktuálneho času a atribútu Stop väčší ako 0, stav ostáva nezmenený.

Ak sa tento rozdiel rovná 0, skončí sa meranie a odalokuje sa pamäť ktorá bola vyhradená na snímané vstupy. Stav meracieho plánu sa zmení z InProgress na Done a stavový automat prejde do stavu IDLE, kedy čaká na ďalší merací plán, ktorý je určený pre datalogger. Na záznam dát zo snímaných vstupov dataloggeru je volaná funkcia

```
void LogValue(GPIOinput *aInput, DataStorage *aStorage)
```

Prvým argumentom funkcie LogValue je ukazovateľ na jeden snímaný vstup dataloggeru. Druhým argumentom je štruktúra DataStorage popísaná na začiatku tejto kapitoly. Keďže prvým argumentom funkcie je len jeden vstup dataloggeru, v stave LOG sú prechádzané cyklom for všetky snímané vstupy dataloggeru, ktoré boli vytvorené v stave CONFIG. Následne je na každý snímaný vstup zavolaná funkcia LogValue v ktorej je implementovaný ďalší stavový automat ktorý beží pre každý snímaný vstup nezávisle. Týmto sa uzatvára celý beh meracej aplikácie a stavového automatu dataloggeru.

7.2.3 Snímané vstupy

Snímaný vstup je v zdrojových kódoch meracej aplikácie prezentovaný ako štruktúra GPIOinput.

```
typedef struct
{
  GPIOValue iValue;
  unsigned long iSampleTime;
  epochTime_t iEpochTimeNext;
  long long iSampleNum;
  void (*iLogFunction)(GPIOValue *);
  epochTime_t iLogTimeStamp;
  GPIOState iState;
} GPIOinput;
```

Prvým atribútom štruktúry ***GPIOinput*** je premenná *GPIOValue*. Tento atribút je ďalšia štruktúra ktorá obsahuje konkrétne údaje o snímanom vstupe.

```
typedef struct
{
   unsigned iGPIONum;
   LOG_TYPE iInput;
   GPIOLogErr iErr;
} GPIOValue;
```

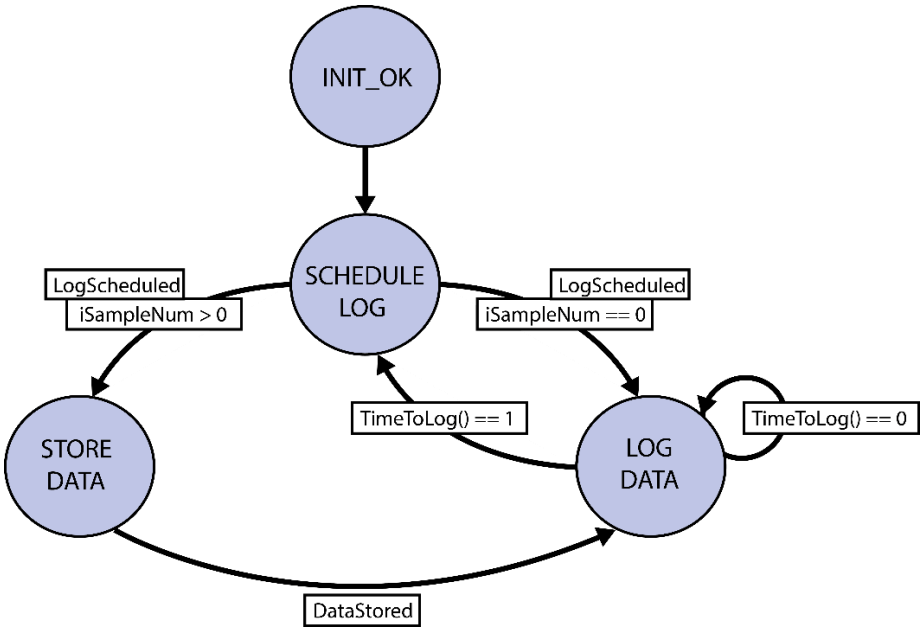
Prvá premenná v štruktúre ***GPIOValue*** je číslo vstupu, ktorý bol zvolený užívateľom ako kanál. Pomocou tohto vstupu sa pristupuje práve ku konkrétnemu GPIO. Toto číslo je totožné s číslovaním na obrázku 12. Ďalším atribútom je *iInput*. Typ tejto premennej v štruktúre ***GPIOValue*** je možné definovať pomocou #define LOG_TYPE. V tomto riešení je typ definovaný ako bool z knižnice stdbool.h. Pri zázname dát je do tejto premennej ukladaná logická hodnota v ktorom sa pri vykonaní vzorky snímaný vstup nachádza, to znamená že nadobúda hodnotu 0 alebo 1. Posledným atribútom tejto štruktúry je *GPIOLogErr*. Táto premenná obsahuje informáciu o dodržaní ekvidistantného kroku vzorkovania. Môže nadobúdať dve hodnoty. Pri dodržaní ekvidistantného kroku vzorkovania je táto hodnota LOG_OK, naopak pri jeho nedodržaní je hodnota premennej GPIOLogErr LATE_LOG_ERR.

Ďalšími atribútmi v štruktúre ***GPIOinput*** je *iSampleTime* a *iSampleNum*. Ako už z názvu premennej *iSampleTime* vyplýva, ide o vzorkovaciu periódu určenú pre snímaný vstup. Perióda vzorkovania je ukladaná do premennej v milisekundách. Premenná iSampleNum je po každej vykonanej vzorke inkrementovaná, čím obsahuje informáciu a počte vzoriek.

V úvode tejto kapitoly bolo zmienené, že typ dát, ktoré je potrebné zaznamenávať sa môže v budúcnosti zmeniť a je potrebné naprogramovať časť meracej aplikácie univerzálnejšie. Tento problém je zatiaľ vyriešený atribútom štruktúry ***GPIOinput***, ktorý má názov *iLogFunction*. Ide o ukazovateľ na funkciu, ktorá sa má pri vzorkovaní snímaného vstupu zavolať. To znamená, že ak bude potrebné v budúcnosti rozšíriť meraciu aplikáciu o ďalšie vstupy, ktoré budú zaznamenávať napríklad analógové hodnoty, bude potrebné prepísať #define LOG_TYPE na požadovaný dátový typ a napísať vlastnú funkciu, ktorá bude dáta zaznamenávať. Zároveň toto riešenie pri zbere dát umožňuje na jednotlivé snímané vstupy volať rôzne funkcie.

Ako zo zadania vyplýva, ku každej vzorke je potrebné vytvoriť údaj o čase, v ktorom bola vzorka vykonaná. Z tohto dôvodu obsahuje štruktúra ***GPIOinput*** atribút s názvom *iLogTimeStamp*. Do tejto premennej sa pri vykonaní vzorky ukladá počet milisekúnd od 1.1.1970 a času 00:00, ktorý je zistený pomocou funkcie EpochTime. Pri prezentácií dát je potom možné pužiť funkcie, ktoré dokážu tento formát konvertovať na klasický formát dátumu a času.

Záznam dát zo snímaného vstupu je realizovaný ako stavový automat, ktorý beží nezávisle voči ostatným vstupom, teda každý snímaný vstup môže byť v inom stave ako ostatné. Realizácia stavového automatu je nakreslená na nasledujúcom obrázku.



Obrázok 18 Stavový automat snímaného vstupu

Prvým stavom stavového automatu snímaného vstupu zobrazeného na obrázku 18 je inicializačný stav. Tento stav je nastavený hneď ako prebehne inicializácia všetkých premenných snímaného vstupu. Ak tento stav nie je nastavený, stavový automat sa nezačne vykonávať, čo znamená že pri inicializácií nastala chyba.

Ďalším stavom je plánovanie nasledujúceho zberu dát zo vstupu v stavovom automate nazvaný ako SCHEDULE LOG. V tomto stave je volaná inline funkcia:

```
inline void ScheduleLog(GPIOinput *aInput)
```

Naplánovanie nasledujúceho vykonania vzorky spočíva v sčítaní aktuálneho časového údaju získaním funkciou EpochTime() a periódy vzorkovania určenej pre snímaný vstup. Výsledok tohto sčítania sa ukladá do premennej *iEpochTimeNext*. Ak ide o prvú vzorku snímaného vstupu, stavový automat prejde do stavu záznamu dát LOG DATA a čaká na čas, kedy môže vykonať vzorku. Ak sa však nejedná o prvú vzorku, nasledujúcim stavom je uloženie zaznamenaných dát z premenných štruktúry ***GPIOinput***.

Ďalším stavom stavového automatu je LOG DATA. V tomto stave meracia aplikácia po naplánovaní čaká na čas, v ktorom má vykonať vzorku na snímanom vstupe dataloggeru. V tomto stave sú volané dve funkcie.

```
inline bool TimeToLog(const epochTime_t aEpochTimeNext)
inline bool CheckEpochTime(const epochTime_t aEpochTimeNext)
```

Prvá funkcia vracia výsledok porovnania premennej *iEpochTimeNext,* v ktorej je uložený časový údaj v ktorom má byť vykonaná vzorka s aktuálnym časom získaným funkciou EpochTime. Ak je návratová hodnota false, stav ostáva nezmenený.

Naopak ak je návratová hodnota true, zavolá sa funkcia na vykonanie vzorky pomocou ukazovateľa *iLogFunction*. V tomto prípade sa do premennej *GPIOLogErr* uloží LOG_OK, pretože bol dodržaný ekvidistantný krok vzorkovania. V stave Log však môže nastať aj stav, kedy sa záznam snímaného vstupu nestihol. V takom prípade je časový údaj vrátený funkciou väčší, ako ten ktorý bol naplánovaný v stave SCHEDULE LOG. Výsledok tohto porovnania vyhodnocuje funkcia CheckEpochTime. Ak je návratovou hodnotou tejto funkcie true, znamená to, že sa nepodarilo dodržať ekvidistantný krok vzorkovania. Vzorka sa vykoná hneď ako to je možné a do premennej *GPIOLogErr* sa uloží hodnota LATE_LOG_ERR. Dôvod nedodržania ekvidistantného kroku bol diskutovaný v predchádzajúcich kapitolách. Hneď po uložení všetkých potrebných dát zo vstupu, teda jeho hodnota a časové razítko prechádza stavový automat do stavu SCHEDULE LOG na okamžité naplánovanie časového údaju novej vzorky. Ak by stavový automat prechádzal zo stavu LOG DATA rovno do stavu STORE DATA, každá vykonaná vzorka by bola posunutá o čas, ktorý bol využitý na ukladanie nameraných dát v stave STORE DATA, čím by bol porušený ekvidistantný krok každej vzorky.

Posledným stavom v stavovom automate snímaného vstupu je stav ukladania nameraných dát. V tomto stave je volaná funkcia StoreData, ktorá pristupuje do databázy a zapisuje namerané hodnoty.

```
void StoreLogData(GPIOinput *aData, DataStorage *aStorage)
```

Funkcia StoreData po jej zavolaní pripraví SQL výraz, pomocou ktorého sa vykoná INSERT do tabuľky Log v databáze. Následne je zavolaná funkcia z knižnice mysql.h ktorá pripravený príkaz predloží databáze.

8 WEBOVÁ APLIKÁCIA

Webová aplikácia má vo zvolenom koncepte viac menej charakter informačného systému, pomocou ktorého je možné do systému vkladať nový datalogger, vkladať nové riadiace jednotky plynových kotlov, vytvárať konfigurácie dataloggeru s riadiacou jednotkou plynového kotla a zdávať meracie plány pre datalogger. Hlavnou časťou webovej aplikácie tvoria skripty napísané v jazyku PHP. Tieto skripty bežia na strane servera a ich hlavnou úlohou je poskytovanie dát z databázy. Na vytvorenie základu webovej stránky, to znamená tlačidiel, vstupných polí a podobane, bol použitý značkovací jazyk html s kombináciou CSS. Pomocou CSS bol definovaný základný štýl webovej aplikácie ktorý, však nebol vytváraný na základe žiadnej predlohy. To je dôvodom jednoduchého dizajnu webovej aplikácie, ktorý bude potrebné v budúcnosti prerobiť. Posledným nástrojom pri programovaní webovej aplikácie bol JavaScript pomocou ktorého sú vykresľované grafy nameraných hodnôt. Na vykresľovanie výsledných priebehov bola použitá knižnica vis.js [10]. Táto knižnica bola použitá na vykresľovanie priebehov a vizualizácií meracích plánov hlavne kvôli možnosti približovania a odďaľovania časovej osi grafu až na milisekundy, čo je pri najnižšej perióde vzorkovania 100 milisekúnd rozhodujúcim faktorom.

Rozsah len samotnej webovej aplikácie je približne 4600 riadkov a jej detailný popis by presahoval celkový rozsah tejto práce, preto budú v tejto kapitole stručne popísané hlavné časti a ich významy vo webovej aplikácií. Ich realizáciu je možné nájsť v prílohe ako priložené zdrojové kódy.

8.1 Prístup k dátam v databáze

Webová aplikácia pristupuje k dátam v databáze pomocou PDO objektov [11], ktoré sú podporované od verzie PHP 5. Dôvodom použitia PDO objektov je v prvom rade podpora objektovo orientovaného programovania pri ich používaní. Druhou dôležitou výhodou je implementácia takzvaných predpripravených výrazov, v angličtine prepared statements, pri vytváraní SQL príkazu. Pomocou tohto riešenia prístupu do databázy je zamedzená možnosť SQL injekcie, pomocou ktorej je možné meniť výsledný príkaz. Príklad prístupu do databázy vo webovej aplikácií je:

```
$statement = $this->iDB->prepare('SELECT ID
                                  FROM Datalogger
                                  WHERE
                                  SerialNumber=:serialnum');

$statement>bindparam(':serialnum',$this-
>iSerialNum,PDO::PARAM_INT);
$statement->execute();
```

Ak nie je použitá žiadna z bezpečnostných metód vytvárania SQL príkazu a príkaz je tvorený obyčajným skladaním reťazcov znakov, môže nastať jeho modifikácia, čím je narušená bezpečnosť webovej aplikácie. Všetky skripty v PHP ďalej obsahujú len obyčajné metódy, pomocou ktorých sú získavané dáta z databázy.

8.2 Užívateľské role

Vo webovej aplikácií boli vytvorené tri užívateľské role ktoré boli definované v kapitole 4.5. Administrátor má vo webovej aplikácií najvyššie práva. To znamená, že môže vykonávať všetky úkony, ktoré sú prístupné testerovi a obyčajnému užívateľovi s možnosťou pridávať nových užívateľov a nové dataloggeri, čo však ostatným užívateľským rolám umožnené nie je.

Testerovi je umožnené pridávať nové riadiace jednotky plynových kotlov, no nemôže pridávať nové dataloggeri do databázy všetkých dataloggerov. Ďalej je mu umožnené vytvárať konfigurácie všetkých dataloggerov, ktoré sú v databáze a k nim vytvárať ich nové konfigurácie. Ďalej je testerovi umožnené vytvárať meracie plány pre riadiace jednotky plynových kotlov v databáze.

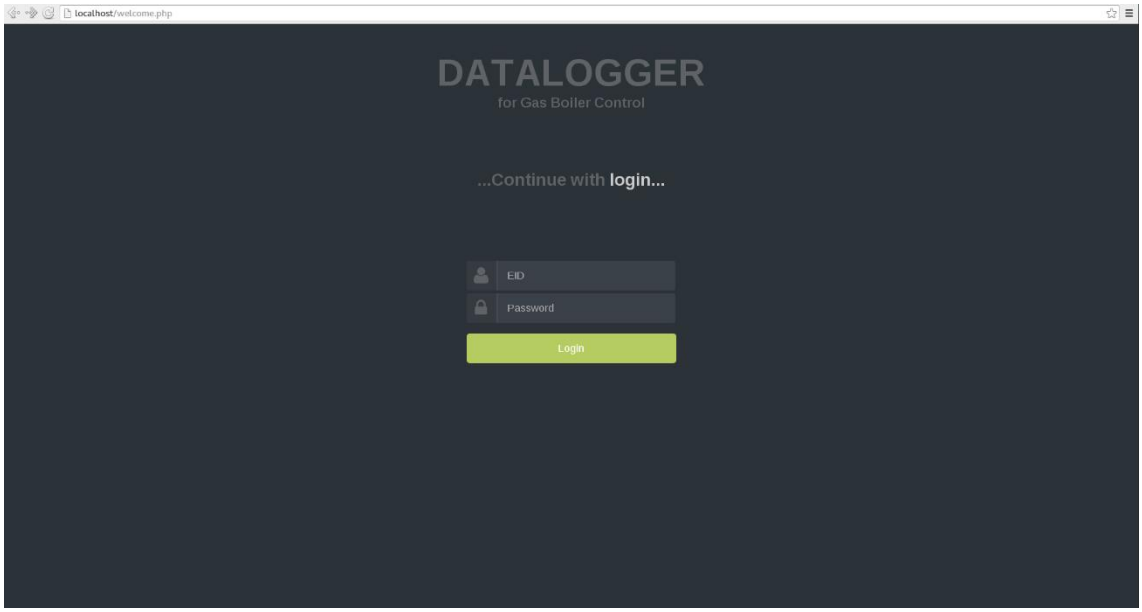
Poslednou užívateľskou rolou vo webovej aplikácií je obyčajný užívateľ. Tejto užívateľskej role nie je umožnené akokoľvek modifikovať alebo vytvárať záznamy v databáze. Obyčajný užívateľ vo webovej aplikácií môže všetky dáta len prezerať.

8.3 Popis modulov webovej aplikácie

V nasledujúcej kapitole sú popísané jednotlivé moduly webovej aplikácie. Význam modul je v tomto ponímaní časť webovej aplikácie, ku ktorej má užívateľ prístup a ktorá zahŕňa určitú funkciu. Celý popis je realizovaný pri prihlásení užívateľa, ktorý je označený ako Administrátor.

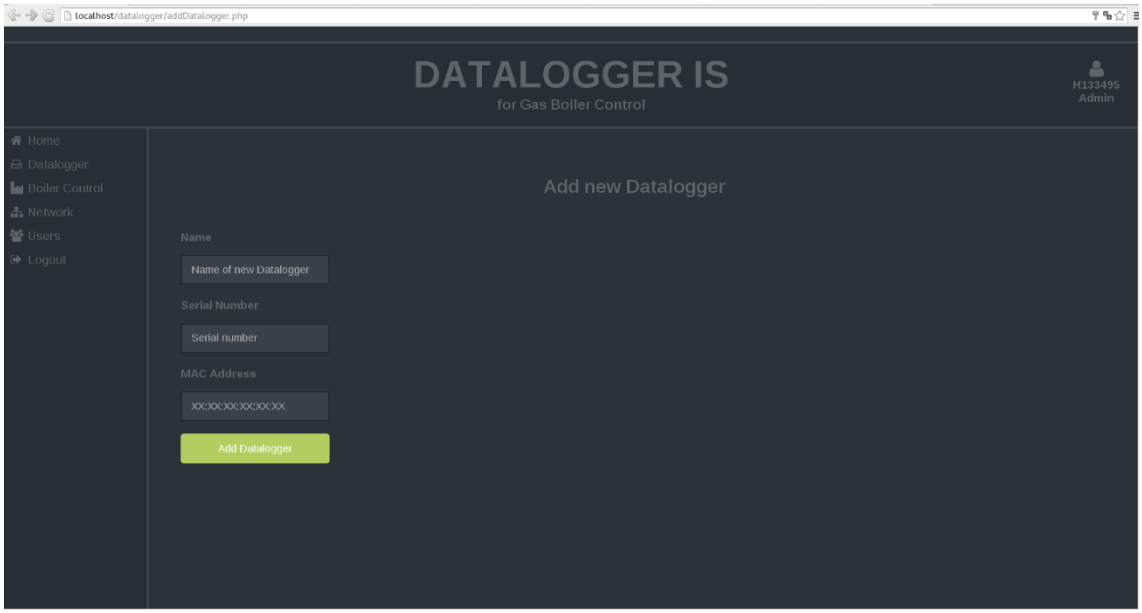
Hlavnou časťou webovej aplikácie je hlavné menu, pomocou ktorého je realizované celé ovládanie webovej aplikácie. Hlavné menu sa nachádza na ľavom okraji webovej aplikácie. Po kliknutí na jednotlivé názvy sú rozdelené na konkrétne úkony, ktoré sú pre aktuálne prihláseného užívateľa možné. V pravom hornom rohu je možné vidieť prihláseného užívateľa a jeho užívateľskú rolu vo webovej aplikácií.

Prvou stránkou, ktorá sa zobrazí pri pripojení užívateľa na server je prihlásenie do systému, tak ako bolo požadované zadávateľom. Prihlásenie je realizované prihlasovacím menom a heslom. Nových užívateľom je možné pridávať do databázy len ak je v systéme prihlásený užívateľ s administrátorskými právami. Z tohto dôvodu sa skladá prihlasovacia stránka len z dvoch vstupných polí, ktorých obsah sa po zatlačení tlačidla login odošle na server a skontroluje s databázou užívateľov. Heslá v databáze sú zakriptované algoritmom bcrypt pomocou funkcií, ktoré obsahuje PHP a nie je potrebné používať na kryptovanie hesiel nové knižnice. Výsledný modul prihlásenia užívateľa do systému je možné vidieť na obrázku 19.



Obrázok 19 Úvodná stránka webovej aplikácie

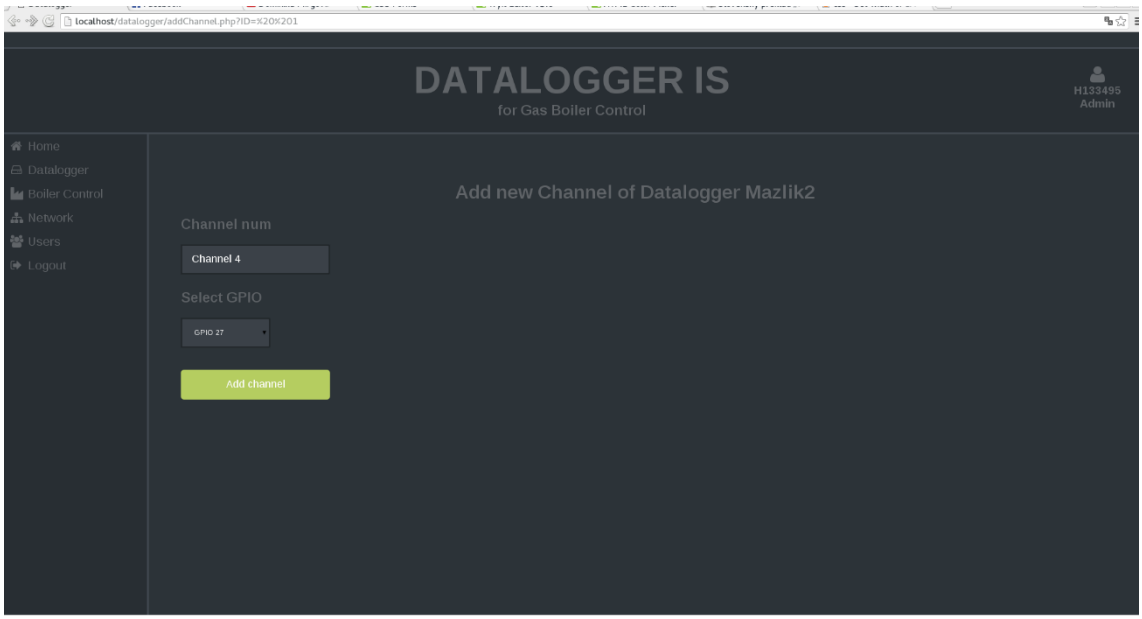
Ďalším dôležitým modulom webovej aplikácie je modul, pomocou ktorého je do databázy pridávaný nový datalogger. Tabuľka dataloggerov sa skladá z troch atribútov, ktorých vstupné polia je možné vidieť na obrázku 20. Po zatlačení tlačidla *Add Datalogger* sa na strane serveru skontroluje unikátnosť názvu a sériového čísla nového dataloggeru.



Obrázok 20 Stránka na pridanie nového dataloggeru

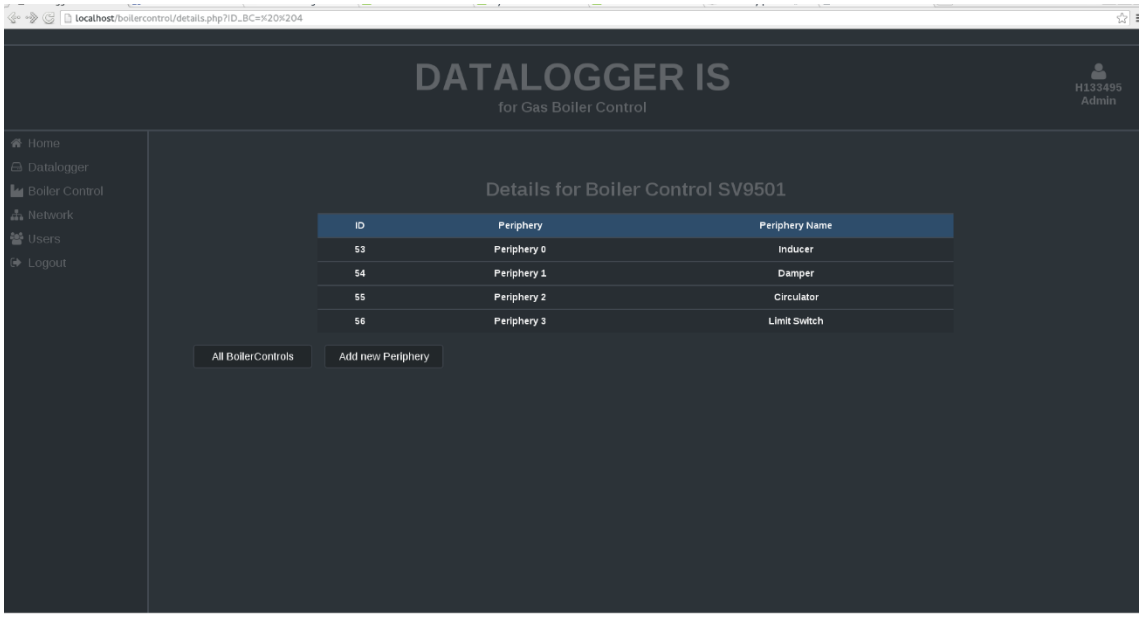
Do modulu pre datalogger je možné zaradiť aj stránku, pomocou ktorej je možné dataloggeru pridávať nové snímané kanály. Pridaním nových kanálov sa vytvárajú nové záznamy v tabuľke *Channel* popísanej v kapitole 7.1.1. Na priradenie konkrétneho vstupu hardwarovej platformy k snímanému kanálu sa používa textový vstup označený v html ako select.

Pri pridávaní nového kanálu sa užívateľovi ponúknu možnosti vstupov hardwarovej platformy, ktoré ešte neboli obsadené inými kanálmi. Stránku určenú na pridávanie kanálov dataloggeru je možné vidieť na obrázku 21.



Obrázok 21 Stránka na pridanie nového kanálu k dataloggeru

Po pridaní nového dataloggeru je potrebné do databázy vložiť nové riadiace jednotky plynových kotlov, ktorých periférie bude potrebné počas meracieho plánu snímať. Pridanie novej riadiacej jednotky plynového kotla je realizované pomocou hlavného menu v časti *Boiler Control - Add new*. Postup pridania novej riadiacej jednotky je totožný s pridávaním nového dataloggeru do systému, preto nie je potrebné ho ďalej popisovať a vysvetľovať. Pri popise tohto modulu je použitý príklad prezentácií dát v tabuľkách. Tento typ prezentácie dát je použitý v celej webovej aplikácií a nemá význam popisovať ho pri každom module webovej aplikácií zvlášť. Na obrázku 22 je možné vidieť prezentáciu periférií existujúcej riadiacej jednotky plynového kotla v systéme, ktorá bola použitá ako príklad pri vytváraní týchto obrázkov. Snahou Bolo vytvoriť čo najzrozumiteľnejšie tabuľky v celej aplikácií.



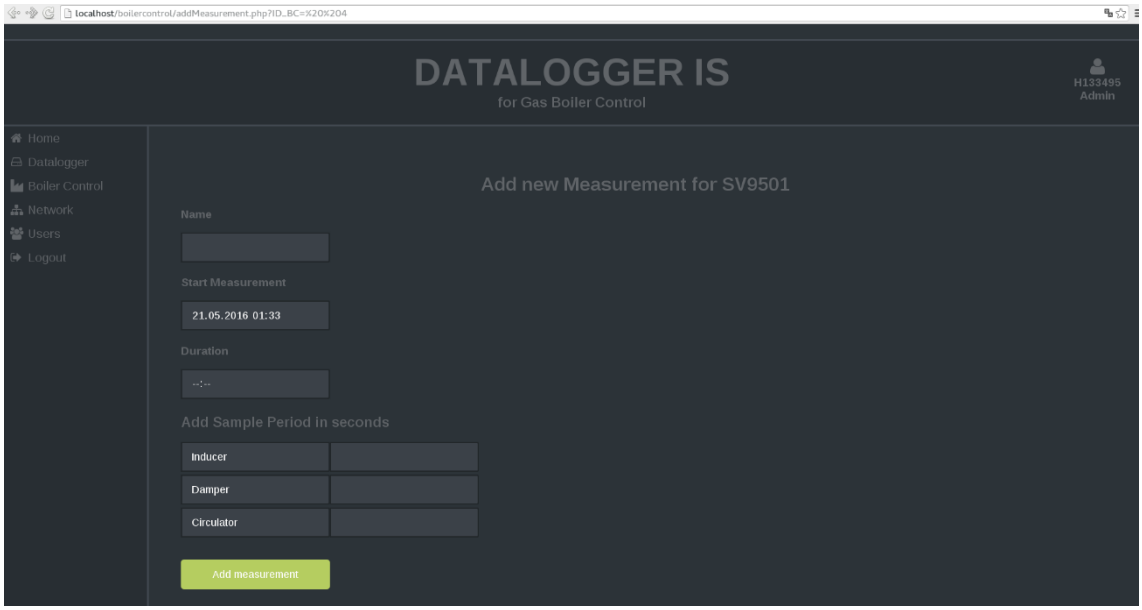
Obrázok 22 Stránka zobrazujúca všetky Periférie

Ak sa v databáze nachádza aspoň jeden záznam popisujúci datalogger a záznam popisujúci niektoré z požadovaných snímaných zariadení, je potrebné vytvoriť konfiguráciu dataloggeru. Konfigurácia dataloggeru spočíva v spojení kanálov dataloggeru a periférií sledovanej riadiacej jednotky plynového kotla, čo odpovedá ER diagramu v kapitole 7.1.3 a taktiež fyzickému pripojeniu sledovaných signálov k snímaným kanálom dataloggeru. Prístup k tomu modulu je realizovaný pomocou hlavného menu v časti *Datalogger – Config*.

Ďalším modulom webovej aplikácie je modul meracích plánov. V module meracích plánov je užívateľ so správnymi prístupovými právami schopný zadávať meracie plány konkrétnym sledovaným zariadeniam. Modul meracích plánov prezentovaný vo webovej aplikácií je možné vidieť na obrázku 23. Pomocou tohto modulu je možné naplánovať v ideálnom prípade nekonečný počet meracích plánov pre sledované zariadenie. Keďže algoritmus meracej aplikácie popísaný v kapitole 7.2.2 dokáže naplánovať merací plán, užívateľovi je umožnené vkladať meracie plány bez ohľadu na ich postupnosť. Po stlačení tlačidla *Add measurement* sa údaje o meracom pláne odošlú na server a skontroluje sa unikátnosť meracieho plánu a hlavne časový údaj o jeho začiatku a konci. Meracie plány určené jednému dataloggeru sa časovo nemôžu prekrývať. Ak by tento stav mal nastať, merací plán sa do databázy neuloží a je potrebné ho zmeniť.

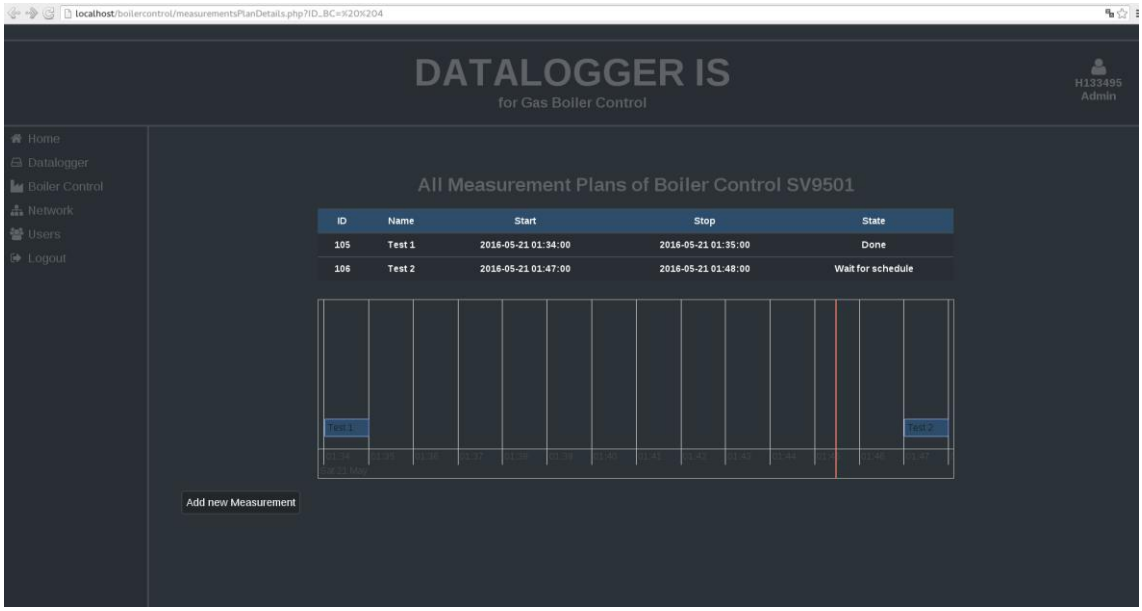
Užívateľovi nie je dovolené vkladať meracie plány, ktoré by mali začať v minulosti. Preto je vstupné pole *Start Measurement* ošetrené a automaticky pridá 2 minúty k údaju o začatí meracieho plánu. Pri pridávaní meracieho plánu je potrebné definovať periódy vzorkovania, ktorou sa budú vzorkovať jednotlivé periférie sledovaného zariadenia.

Po správnom vložení meracieho plánu sa už o konfiguráciu, beh a skončenie meracieho plánu postará meracia aplikácia bežiaca na dataloggeri.



Obrázok 23 Stránka na vytváranie meracích plánov

Všetky merania, ktoré sú naplánované, čakajú na naplánovanie, práve bežia alebo už boli vykonané, je možné vo webovej aplikácií vidieť vo forme tabuľky, alebo vo vizualizácií, ktorá je ukázaná na obrázku 24. Modré polia zobrazujú všetky meracie plány v databáze a červená čiara pohybujúca sa v reálnom čase zobrazuje, aký merací plán práve beží.



Obrázok 24 Stránka zobrazujúca všetky meracie plány

Prezentácia nameraných dát je realizovaná taktiež pomocou tabuliek a grafu priebehu jednotlivej periférie počas behu meracieho plánu. Príklad prezentácie nameraných dát je možné vidieť v kapitole 9 tejto práce. V budúcnosti bude potrebné prezentáciu dát užívateľovi o veľa zlepšiť.

9 VÝSLEDKY TESTOVANIA DATALOGGERU

Počas vývoja dataloggeru firma Raspberry Pi Fundation vydala novú verziu platformy s označením Raspberry Pi 3. Táto verzia je výkonnejšia v porovnaní so zvolenou platformou Raspberry Pi 2 a náklady na realizáciu dataloggeru sa jej použitím až tak razantne nezvýšia. Testovanie prebiehalo už na novej generácií Raspberry Pi 3.

9.1 Rozbor testovania dataloggeru

Prvou iteráciou testovania bolo percentuálne vyhodnotenie počtu vzoriek, v ktorých je porušený ekvidistantný krok vzorkovania. Počas testovania nebol na snímané vstupy privádzaný žiadny signál a datalogger nebol pripojený k žiadnemu snímanému zariadeniu. Zmyslom testu bolo aspoň približné vyhodnotenie závislosti periódy vzorkovania na počte kanálov a porušenia ekvidistantného kroku medzi periódami vzorkovania. Rozsah vzorkovacích periód bol zvolený od minimálnej hodnoty 0,1 sekundy až po 5 sekúnd.

Druhá časť testovania dataloggeru spočívala v privedení signálov z reálneho zariadenia poskytnutého zadávateľom na snímané vstupy dataloggeru. Na testovanie bola použitá riadiaca jednotka plynového kotla SLATE R8001 [12], ktorej testovaný modul je možné vidieť na obrázku 25. Riadiaca jednotka plynového kotla bola spustená v testovacom režime, počas ktorého opakovala vykurovací cyklus.

Obrázok 25 Použitý modul riadiacej jednotky SLATE R8001 [12]

Pri druhom testovaní dataloggeru boli použité 3 kanály, na ktoré boli privedené cez napäťové konvertory signály K1, IGN a PV. Nasledujúca tabuľka hovorí o konfigurácií a periódach vzorkovania, ktoré boli zvolené pomocou webovej aplikácie.

Periféria	Perióda vzorkovania [s]	Kanál dataloggeru [-]	GPIO [-]
K1	1	0	16
PV-Pilot Valve	10	1	20
IGN-Ignition	5	2	17

Tabuľka 3 Konfigurácia a periódy vzorkovania reálnych signálov

9.2 Predpokladané výsledky testovania

Z požiadaviek zadávateľa vyplýva, že najnižšia perióda vzorkovania by mala byť 100 milisekúnd a maximálny počet snímaných kanálov 16. Z návrhovej časti dataloggeru je možné očakávať, že pri nízkych periódach vzorkovania a vyššom počte snímaných kanálov bude porušený ekvidistantný krok medzi periódami vzorkovania.

Druhý test dataloggeru nebude zameraný na presnosť dodržania ekvidistantného kroku medzi periódami vzorkovania. Hlavnou úlohou druhého testu dataloggeru bude overenie zhody priebehu signálu na snímanom vstupe s nameranými dátami počas meracieho plánu. Jednou z požiadaviek zadávateľa na datalogger bola, aby bol beh dataloggeru stabilný a meracia aplikácia bezdôvodne neskončila počas meracieho plánu. Z tohto dôvodu bude bežať merací plán 30 minút a je možné predpokladať, že prebehne bez nejakých problémov počas merania.

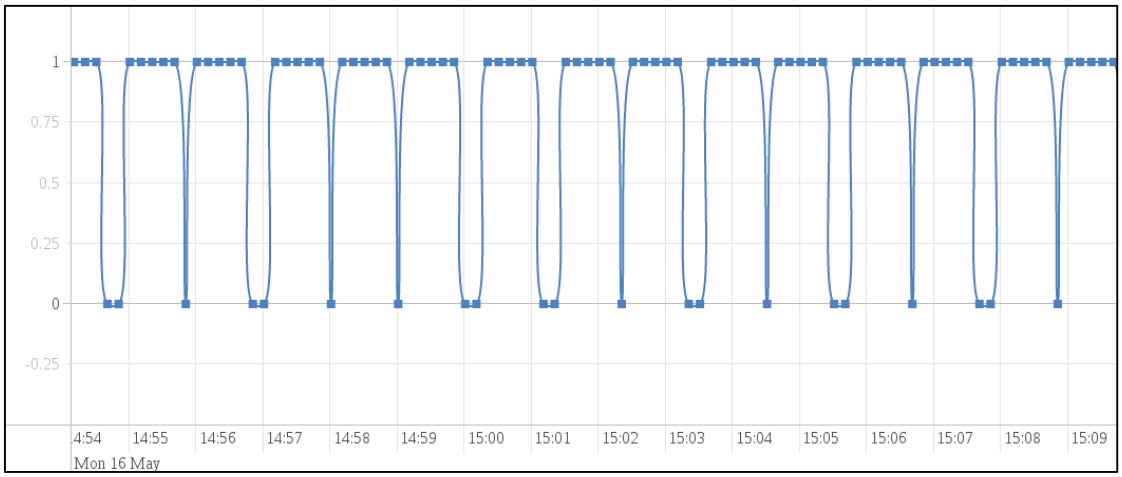
9.3 Skutočné výsledky testovania a zhodnotenie

Výsledky z priebehu prvého testu je možné vidieť v tabuľke 4 tejto kapitoly. Ako z tabuľky nameraných hodnôt vyplýva, periódy vzorkovania vyššie ako 1 sekunda nie sú pre meraciu aplikáciu kritické, to znamená že dokáže dodržať ekvidistantný krok vzorkovania. Je zrejmé, že najkritickejšou periódou vzorkovania je pre meraciu aplikáciu perióda nižšia ako 1 sekunda. Táto nepresnosť je vo veľkej miere spôsobená tým, že namerané dáta meracia aplikácia vkladá priamo do databázy. Ďalším dôvodom nepresností je operačný systém bežiaci na Raspberry Pi. Vplyv operačného systému, ktorý nie je kategorizovaný ako operačný systém reálneho času bol diskutovaný v návrhovej časti tejto práce.

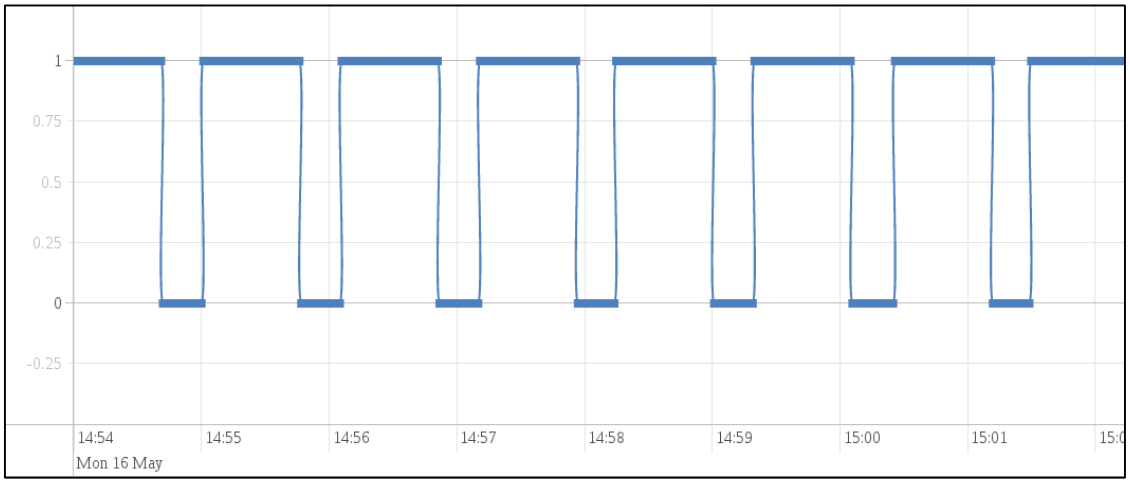
Z hľadiska druhého testovania prebehol merací plán bez chyby ktorá by ho mohla zastaviť a namerané hodnoty počas meracieho plánu odpovedali signálom na vstupoch dataloggeru. Výsledné priebehy z druhého testovania dataloggeru sú zobrazené na obrázku 26, obrázku 27 a obrázku 28.

Perióda Vzorkovania [s]	Počet snímaných kanálov [-]	Počet všetkých vzoriek [-]	Počet chybných vzoriek [-]	Chyba vzorkovania [%]
0,1	1	589	7	1,2
	3	1548	687	44,4
	5	1725	800	46,4
	10	2540	1590	62,6
	16	3424	3408	99,5
0,3	1	198	0	0,0
	3	591	9	1,5
	5	923	171	18,5
	10	1740	850	48,9
	16	1536	1520	99,0
0,5	1	118	0	0,0
	3	354	0	0,0
	5	585	35	6,0
	10	1110	363	32,7
	16	1504	1440	95,7
1	1	59	0	0,0
	3	177	0	0,0
	5	295	0	0,0
	10	580	0	0,0
	16	960	0	0,0
2	1	29	0	0,0
	3	87	0	0,0
	5	145	0	0,0
	10	290	0	0,0
	16	464	0	0,0
5	1	11	0	0,0
	3	33	0	0,0
	5	55	0	0,0
	10	110	0	0,0
	16	176	0	0,0

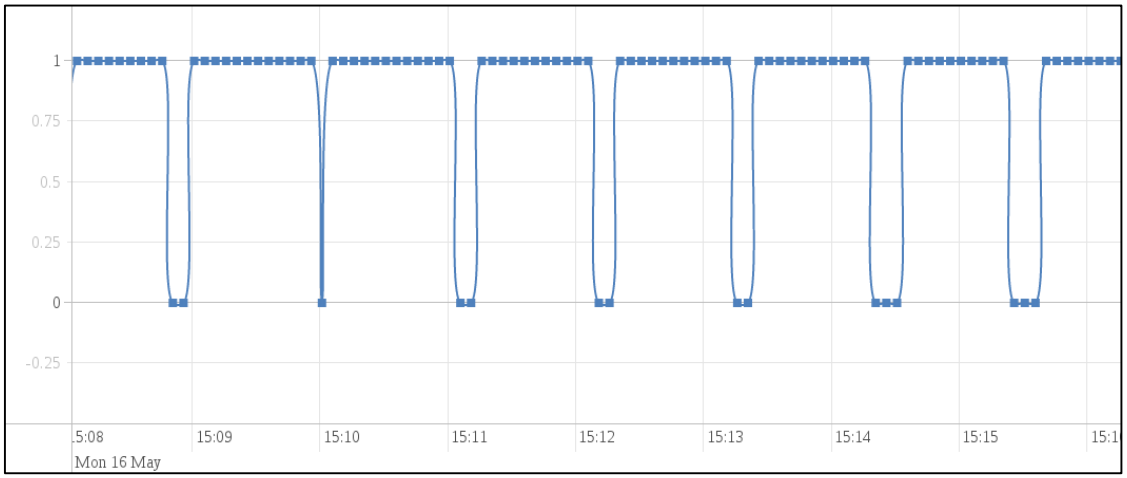
Tabuľka 4 Výsledky testovania dataloggeru



Obrázok 26 Priebeh signálu Ignition počas testovania



Obrázok 27 Priebeh signálu PV - Pilot valve



Obrázok 28 Priebeh signálu K1

10 ZÁVER

V tejto bakalárskej práci je uvedený návrh a realizácia dataloggeru určeného k zberu dát z riadiacich jednotiek plynových kotlov vyvíjaných firmou Honeywell Brno. Na začiatku práce je venovaná časť rozboru možných riešení zberu dát z riadiacich jednotiek plynových kotlov, pričom vznikom tohto zadania zadávateľ zvolil vývoj vlastného dataloggeru.

Po teoretickom rozbore všetkých požiadaviek na hardwarovú platformu dataloggeru, bola na realizáciu vybratá platforma Raspberry Pi 2. Počas vývoja dataloggeru bola vydaná novšia verzia tejto platformy, preto bol datalogger v konečnom dôsledku realizovaný na platforme Raspberry Pi 3, ktorá ponúka vyšší výpočtový výkon.

Datalogger disponuje 16 snímanými vstupmi, na ktoré je možné priviesť logické stavy s úrovňou napätia 3.3 V a 0 V. Signál, ktorý je privedený na vstup musí byť predtým prevedený cez zadávateľom dodaný konvertor napätia. Snímané vstupy dataloggeru môžu byť vzorkované s periódou od 100 milisekúnd až po 1 hodinu.

Počas vypracovania tohto zadania vznikli dva koncepty firmwaru. Prvým konceptom je firmware dataloggeru, ktorý tvorí meracia aplikácia, databázový a webový server, bežiaci na dataloggeri. Druhým konceptom je firmware, kde bol databázový a webový server presunutý na hlavný server realizovaný ako niektorý z počítačov v lokálnej sieti vo firme. Primárnym dôvodom realizácie konceptu s hlavným serverom je zjednodušenie firmwaru bežiaceho na dataloggeri, čím je zvýšená presnosť pri dodržaní ekvidistantného kroku vzorkovania. Problém s dodržaním ekvidistantného kroku však ani týmto riešením nebol úplne potlačený. Výsledky testovania hovoria, že datalogger dokáže spoľahlivo dodržať ekvidistantný krok vzorkovania na všetkých snímaných kanáloch až pri perióde vyššej ako 1 sekunda. Znižovaním počtu kanálov sa chyba pri vzorkovaní znižuje, no nie je možné zaručiť, že pri aktuálnom riešení meracej aplikácie bude úplne minimalizovaná. Dôvodom je využitie OS Linux, ktorý neobsahuje podporu dostupnú u operačných systémov reálneho času.

Podľa požiadaviek zadávateľa bola realizovaná webová aplikácia pre zobrazenie všetkých dát užívateľovi. Keďže vznikom hlavného serveru je umožnené všetkým dataloggerom ukladať dáta na jedno miesto, webová aplikácia bola naprogramovaná s charakterom informačného systému s tromi užívateľskými rolami, Administrátor, Tester a obyčajný užívateľ. Pomocou webovej aplikácie je možné spravovať ktorýkoľvek datalogger. Koncept bol navrhnutý tak, aby užívateľ mohol naplánovať akékoľvek meracie plány pre datalogger pripojený v lokálnej sieti.

V budúcnosti bude potrebné, aby meracia aplikácia prešla úpravami v algoritme a to hlavne pri ukladaní nameraných vzoriek do databázy, čím by sa z časti mohla znova zvýšiť presnosť ekvidistantného vzorkovania. V rámci prvej verzie dataloggeru bolo dosiahnuté maximum. Zariadenie ďalej pôjde do skúšobnej prevádzky u zadávateľa z ktorej vzniknú požiadavky na zlepšenie hardwaru a firmwaru dataloggeru.

ZOZNAM POUŽITEJ LITERATÚRY

[1] DATAQ Instruments [online]. 2011 [cit. 2015-12-2]. Špecifikácia DI – 160. Dostupné na URL: <http://www.dataq.com/resources/pdfs/datasheets/di-160-event-data-logger.pdf>

[2] MadgeTech [online]. 2014 [cit. 2015-12-7]. Špecifikácia State101A. Dostupné na URL: <http://www.madgetech.com/pdf_files/data_sheets/state101a_ds.pdf>

[3] MadgeTech [online]. 2014 [cit. 2015-12-7]. Špecifikácia RFPulse2000A. Dostupné na URL: <http://www.madgetech.com/pdf_files/data_sheets/RFPulse2000A_DS.pdf>

[4] Adafruit [online]. 2015 [cit. 2015-12-9]. Raspberry Pi 2, Model B. Dostupné na URL: <https://www.adafruit.com/pdfs/raspberrypi2modelb.pdf>

[5] BeagleBoard [online]. 2015 [cit. 2015-12-9]. BeagleBone Black. Dostupné na URL: <http://beagleboard.org/BLACK>

[6] CubieBoard Docs [online]. 2015 [cit. 2015-12-10]. CubieBoard Open-source Main-Boards. Dostupné na URL: <http://docs.cubieboard.org/products/start#a20-cubieboard>

[7] Element 14 [online]. 2015 [cit. 2016-1-16]. Raspberry Pi 2 GPIO Header Dostupné na URL: <http://www.element14.com/community/docs/DOC-73950/l/raspberry-pi-2-model-b-gpio-40-pin-block-pinout>

[8] MariaDB Corporation [online]. 2015 [cit. 2015-12-10]. CubieBoard Open-source Main-Boards. Dostupné na URL: <https://mariadb.com/kb/en/mariadb/documentation/>

[9] Honeywell [online]. 2015 [cit. 2016-1-27]. Integrated Boiler Controllers. Dostupné na URL: <https://customer.honeywell.com/resources/techlit/TechLitDocuments/66-0000s/66-1203.pdf>

[10] Vis.js [online]. 2016 [cit. 2016-5-10]. Vis.js Documentation. Dostupné na URL: <http://visjs.org/index.html#>

[11] The PHP Group [online]. 2016 [cit. 2016-5-15]. PHP Data Objects. Dostupné na URL: <http://php.net/manual/en/book.pdo.php>

[12] Honeywell [online]. 2016 [cit. 2016-5-15]. R8001 User Guide. Dostupné na URL: <https://www.lesman.com/unleashd/catalog/combustion/Honeywell-Slate/Honeywell-slate-Man-2015-10.pdf>

ZOZNAM OBRÁZKOV

ZOZNAM TABULIEK

ZOZNAM SKRATIEK A SYMBOLOV

TTL	Transistor Transistor Logic
V	Volt
ms	milisekunda
OS	Operačný systém
AC	Alternating current
Wi-Fi	Wireless Fidelity
IP	Internet Protocol
USB	Universal Serial Bus
ERD	Entity-relationship diagram
NTC	Network Time Protocol
RTC	Real Time Clock
SQL	Structured Query Language