Fakulta elektrotechnická

Katedra aplikované elektroniky a telekomunikací

BAKALÁŘSKÁ PRÁCE

Monitoring vytápění rodinného domku

Autor práce: Jan Dolák

Vedoucí práce: Ing. Petr Weissar, Ph.D.

Plzeň 2012

ZÁPADOČESKÁ UNIVERZITA V PLZNI
Fakulta elektrotechnická
Akademický rok: **2011/2012**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Jan DOLÁK**
Osobní číslo:	**E09B0257P**
Studijní program:	**B2612 Elektrotechnika a informatika**
Studijní obor:	**Elektronika a telekomunikace**
Název tématu:	**Monitoring vytápění rodinného domku**
Zadávající katedra:	**Katedra aplikované elektroniky a telekomunikací**

Zásady pro vypracování:

Navrhněte a realizujte systém pro monitorování provozu vytápění rodinného domku. V práci se zaměřte především na tyto části:

1. Sběr dat z teplotních čidel, digitálních vstupů (běh čerpadla, ventilátoru kotle, HDO, ohřívání vody v bojleru) pomocí vhodné serverové aplikace a ukládání do databáze.
2. Klientská aplikace pro zobrazení aktuálního stavu z databáze. Zobrazení historie včetně grafického vyjádření.
3. Aplikace pro online sledování stavu vytápění přes internet prostřednictvím webového prohlížeče.

Rozsah grafických prací: **podle doporučení vedoucího**

Rozsah pracovní zprávy: **20 - 30 stran**

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

Seznam odborné literatury:

Student si vhodnou literaturu vyhledá v dostupných pramenech podle doporučení vedoucího práce.

Vedoucí bakalářské práce: **Ing. Petr Weissar, Ph.D.**
Katedra aplikované elektroniky a telekomunikací

Konzultant bakalářské práce: **Ing. Petr Weissar, Ph.D.**
Katedra aplikované elektroniky a telekomunikací

Datum zadání bakalářské práce: **17. října 2011**

Termín odevzdání bakalářské práce: **3. června 2012**

Doc. Ing. Jiří Hammerbauer, Ph.D.
děkan

L.S.

Doc. Dr. Ing. Vjačeslav Georgiev
vedoucí katedry

V Plzni dne 17. října 2011

Abstrakt

Cílem této bakalářské práce je navrhnout monitorování vytápění rodinného domu pomocí osobního počítače. Sledování jednotlivých teplot topné soustavy, snímání digitálních a analogových vstupů od jednotlivých prvků topení. Jejich vhodné zpracování, zaznamenávání do databáze, následné zobrazení v PC pomocí klientské aplikace včetně grafů a možnost on-line sledování vytápění rodinného domku přes internet. V první části bakalářské práce je seznámení s použitými technologiemi. Další část podrobně popisuje návrh a realizace snímacích převodníků pro sériovou linku a třetí část obsahuje popis jednotlivých aplikací.

Klíčová slova

Monitorování vytápění, převodník sériové linky, čidlo teploty, ukládání do databáze, tvorba grafů, webová aplikace.

Abstract

Dolák, Jan. Monitoring of family house heating [Monitoring vytápění rodinného domku]. Pilsen 2012. Bachelor's thesis (in Czech). University of West Bohemia. Faculty of Electrical Engineering.
Department of Applied Electronics and Telecommunications. Supervisors: Ing. Petr Weissar, Ph.D.

The aim of this thesis is to suggest monitoring of heating the house by using a personal computer. Monitor temperature of each heating system, scanning digital and analog inputs from various elements of the heating. The appropriate treatment recording to the database, subsequent displaing on the PC by using a client application, including charts and on-line monitoring of heating family house over the internet. In the first part of the thesis is to introducing the mentioned technologies. The next part describes in detail the design and implementation of scanning for the serial converter and a third section contains a description of each application.

Key words

Monitoring of heating, serial line converter, temperature sensor, storing in a database, graphs, web application.

Prohlášení

Předkládám tímto k posouzení a obhajobě bakalářskou práci, zpracovanou na závěr studia na Fakultě elektrotechnické Západočeské univerzity v Plzni.

Prohlašuji, že jsem svou závěrečnou práci vypracoval samostatně pod vedením vedoucího 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 bakalářské práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné 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 § 270 trestního zákona č. 40/2009 Sb.

Také prohlašuji, že veškerý software, použitý při řešení této bakalářské práce, je legální.

V Plzni dne 1. června 2012

Jan Dolák

……
Podpis

Obsah

Seznam obrázků

Seznam tabulek

Seznam zkratek

TUV ..... Teplá Užitková Voda.

HDO ..... Hromadné Dálkové Ovládání.

TTL..... Transistor-Transistor-Logic. Tranzistorově-Tranzistorová Logika.

RS-232..... Rozhraní pro přenos informací mezi dvěma zařízeními.

RS-485..... Průmyslové rozhraní pro sériový přenos informací.

COM..... Označení sériového portu na počítačích PC

CMOS..... Complementary Metal–Oxide–Semiconductor. Technologie výroby integrovaných obvodů.

A/D ..... Analogově-Digitální převodník. Převádí spojitý (analogový) signál na signál diskrétní (digitální).

UART ..... Universal Asynchronous Receiver and Transmitter. Asynchronní sériové rozhraní

BASCOM ..... Programovací jazyk založený na programovacím jazyce basic.

PHP..... Hypertext Preprocessor. Skriptovací jazyk určený k programování dynamických internetových stránek.

IIS ..... Internet Information Services. Internetová Informační Služba – softwarový webový server.

DLL ..... Dynamic-Link Library. Dynamicky linkovaná knihovna.

CSS..... Cascading Style Sheets. Kaskádové styly. Jazyk popisující způsob zobrazení webových stránek

HTML..... HyperText Markup Language. Jazyk pro vatváření www stránek.

1. Úvod

K vytvoření této práce mě vedl nápad, jak usnadnit detailní sledování vytápění našeho rodinného domu. K vytápění v zimních měsících se používá ústřední teplovodní vytápění se zplynovacím kotlem na tuhá paliva (uhlí, dříví) od firmy Atmos s automatickou řídící jednotkou, která zabezpečuje regulaci a optimální řízení tak, aby měl kotel co možná největší účinnost a zároveň optimální podmínky k provozu. Monitoring vytápění tedy vytváří nástavbu této automatiky a umožní uživateli jednoduchou a přehlednou kontrolu, zda je regulace funkční tak, jak by měla být s možností nahlédnout do historie měření.

Částečnou inspirací byl produkt firmy Regulus [11]. Ta svým zákazníkům s dodávkou svých produktům nabízí online monitoring zařízení přes webový prohlížeč.

Součástí této práce je návrh a praktická realizace komunikačních převodníků mezi sériovou linkou počítače a jednotlivými čidly teplot a vstupů. Celkové blokové schéma ze kterého je vše zřejmé, se nachází v příloze A.5. O komunikaci s jednotlivými převodníky se stará serverová aplikace. Serverová aplikace zpracovává jednotlivá data a podle jejich druhu je ukládá do tabulky v databázi. Zároveň aktuální stav jednotlivých čidel zobrazuje v přehledném okně. Pro komunikaci s uživatelem byly vytvořeny dvě aplikace. První z nich je napsána pro platformu Windows, ta se používá v rámci domácí sítě a umožňuje sledovat aktuální stav, historii a zobrazovat grafy. Druhá aplikace je určena pro web a slouží ke sledování aktuálního stavu přes internet, stačí pouze zadat internetovou adresu www.jandolak.cz.

2. Snímání dat hardware

2.1. Karta s digitálními vstupy

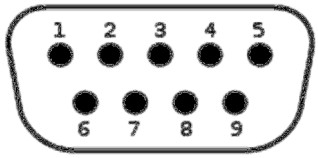
K monitorování vytápění kotle na tuhá paliva je zapotřebí kromě snímání teplot také sledovat běh jednotlivých zařízení nutných pro bezchybný provoz. Mezi takové zařízení patří sledování běhu oběžného čerpadla pro cirkulaci otopné vody. Dále pak běh ventilátoru kotle pro roztopení přiloženého paliva. Primární okruh topné soustavy se také skládá z ohřívače teplé užitkové vody (TUV). V letních měsících, kdy je kotel odstaven z provozu je potřebné užitkovou vodu ohřívat jinak než kotlem. O ohřívání teplé vody se proto stará elektřina a topné těleso o příkonu 2,2kW. Proto dalším potřebným vstupem je ohřev TUV pomocí elektřiny. Ohřívání TUV v letních měsících se děje pouze v přítomnosti nízké sazby elektřiny (tzv. nočního proudu). Čtvrtý vstup proto monitoruje přítomnost signálu HDO (hromadné dálkové ovládání nízké sazby elektřiny).

Pro snímání signálů z těchto zařízení je potřeba dvou stavů. Zapnuto (logická „1“) a vypnuto (log. „0“). Celkem je tedy třeba pokrýt tato čtyři zařízení. Při tomto požadavku jsem hledal vhodný převodník, který by pokryl tyto požadavky. Jako výhodné mi přišlo využití klasického konektoru sériové linky COM, kterou má každý počítač. Signály od ventilátoru kotle, čerpadla atd. jsou vytaženy přímo z řídící automatiky kotle, která má v sobě již připraveny galvanicky oddělené výstupy.

2.1.1. Sériová linka RS-232

Původní účel sériové linky počítače bylo propojení počítače a modemu. Modemu se využívalo pro přenášení dat po běžné telefonní lince. Dále se sériová linka používala k připojení periferií PC jako je myš, tiskárna apod.

Výhody využití RS-232 jsou všeobecně známé, je to především odolnost proti zničení portu, připojování a odpojování připojeného zařízení je možné provádět při zapnutém počítači a v neposlední řadě také možnost napájení cílového zařízení pokud má relativně nízkou spotřebu. Mezi nevýhodu patří ústup tohoto zařízení z osobních počítačů a nahrazování ekvivalenty USB (Universal serial bus), v průmyslu RS-485.



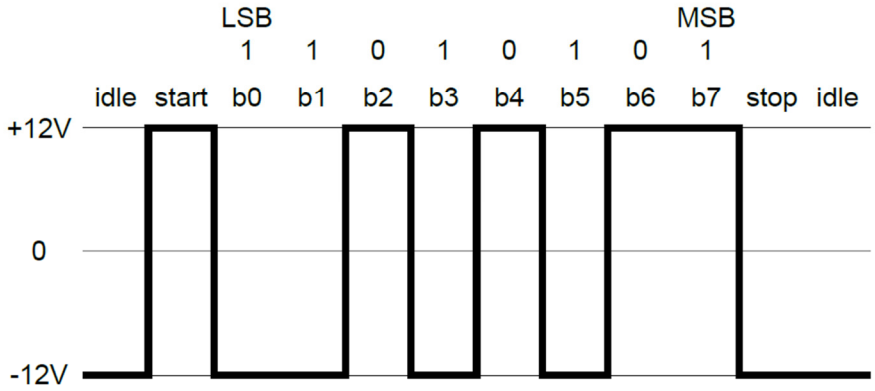
Obr. 2.1: Zapojení pinů konektoru CANON 9 na počítači.

vývod (CANON 9)	Označení	Funkce
1	DCD (data carrier detect)	vstup
2	RxD (receive data)	vstup
3	TxD (transmit data)	výstup
4	DTR (data terminal ready)	výstup
5	GND (ground)	zem
6	DSR (data set ready)	vstup
7	RTS (request to send)	výstup
8	CTS (clear to send)	vstup
9	RING (ring indicator)	vstup

Tab. 2.1: Zapojení vývodů konektoru COM

2.1.2. Popis standardu RS-232

Jedná se o asynchronní sériovou komunikaci pro přenos dat. Přenos probíhá standardně po třech vodičích. Pro příjem RxD, pro vysílání TxD a společná zem GND. Přenos datových bitů je od nejméně významného bitu (LSB) po nejvýznamnější bit (MSB). Běžně se posílá 8bitů pro přenos dat (lze ale také posílat 7 nebo 9 bitů). Přenos jednoho bytu po sériové lince je zřejmý z Obr. 2.2. Z elektrické stránky jsou logické stavy „0“ a „1“ vyjádřeny -12V a +12V. všechny výstupy jsou odolné proti zkratu a mohou dávat proud až 20mA. [19]



Obr. 2.2: Průběh signálů po sériové lince.

Při ovládání sériového portu je možné přímo využívat linky modemu. Takto je možné ovládat až 4 vstupy a 3 výstupy. To by pro mé využití plně stačilo, ale nezbyla by již možnost dalšího rozšíření. To se mi jevilo jako nedostatek, proto jsem začal studovat dostupnou literaturu a hledal nějaké levné a spolehlivé řešení, jak počet vstupů jednoho COM portu rozšířit. V knize Udělejte si z PC v Delphi [2] jsem narazil na přípravek, který je napájen přímo ze sériového portu a rozšíří počet vstupů na celkem 8 v úrovních CMOS logiky (5V). Toto zapojení jsem převzal, navrhl vlastní plošný spoj s vhodnějším připojením jednotlivých vstupů.

2.1.3. Popis 8 vstupové karty

Celé zařízení se skládá z obvodu CMOS 4021, napájecího zdroje LM317L a komparátoru TL061 zajišťujícího převodník úrovní CMOS – RS-232. Schéma 8 vstupové kraty je v příloze A.1.

CMOS 4021 je 8bitový statický posuvný registr s paralelním vstupem a sériovým výstupem. Při aktivaci P/S do log. „1“ se uloží do vnitřního registru stav ze vstupů P1 až P8. Při změně signálu P/S do log. „0“ je na výstupu Q8 hodnota odpovídající pinu P8. Při příchodu impulsů na vstup CLK se na Q8 dají vyčíst stav dalších vstupů v pořadí P8 až P1. [20]

Výstupy sériového portu PC musí být omezeny z důvodu použití 5V logiky pomocí Zenerových diod 4,7V. Pro ovládání hodinových impulsů CLK je využita linka TxD. Signál P/S ovládá linka RTS. Pro napájení operačního zesilovače je zapotřebí symetrického napětí, záporné je získáno z RTS a kladné z linky DTR. O napájení integrovaného obvodu 4021 se stará linka DTR.

Pro napájení se využívají linky DTR a RTS, napětí z těchto pinů je usměrněno diodami D1, D2 a vyfiltrováno kondenzátory C1, C2. Pro zajištění napájecího napětí CMOS logiky je použit nízkopříkonový stabilizátor napětí IO1. Pro správnou funkci je zapotřebí nastavit hodnotu odporu R1 tak, aby jeho výstup poskytoval napájecí napětí 5V.

Signál z Q8 jdoucí na linku RING je potřeba kvůli nestejným úrovním RS-232 a CMOS přizpůsobit. Toto přizpůsobení zajišťuje jednoduchý převodník úrovní s operačním zesilovačem IO3 v zapojení komparátoru.

Piny P1 až P8 jsou přes zdvihací rezistory připojeny na napájecí napětí +5V. Je to z toho důvodu, pokud by byly některé vstupy nepřipojené tak, aby na nich byl definovaný stav a to log. „1“. [2]

2.2. Karta pro snímání teplot

Při hledání vhodného řešení na internetu jsem narazil na systém firmy Dallas - Maxim [21]. Jedná se o využití 1-Wire sběrnice a teplotních čidel DS18B20 [18]. Pro sběr dat z těchto čidel je použit ovládací program LogTemp [14].

2.2.1. Popis sběrnice 1-Wire

Rozhraní 1-Wire bylo vyvinuto v 90. letech firmou Dallas Semiconductor. Jedná se o sériovou asynchronní dvojvodičovou sběrnici, pomocí které komunikují jednoúčelové obvody, například teploměry DS18B20, paměti DS2433, A/D nebo D/A převodníky, čidla vlhkosti a další periferie. Každé zařízení určené pro tuto technologii má při výrobě určenu svojí pevnou adresu a je tak možné jednoduše rozeznávat jednotlivé 1-Wire periferie.

V síti 1-Wire se používají standardní TTL úrovně signálů a napájení komponent může být z vnějšího zdroje s pracovním napětím od 2,8 do 6 V. Alternativou vnějšího zdroje je tzv. „parazitní napájení“. Je založené na využití elektrické energie impulzů předávaných po lince dat a akumulované vestavěnou kapacitou. Mimo to mohou jednotlivé komponenty 1-Wire sítí využívat režim napájení po sběrnici dat (blíže popsáno v převodníku). [21]

Čidla lze připojovat podle topologie buďto lineárně podél jednoho páru drátů nebo hvězdicově, kdy má každé slave zařízení svůj pár drátů, který se rozvětvuje na výstupu řadiče (masteru). Maximální délka je limitovaná zpožděním signálu. V mém případě vzdálenost jednotlivých čidel k serveru nepřesahuje 6 metrů a funguje zcela bezproblémově.

Hlavní předností technologie 1-Wire je výjimečně jednoduché nastavení, odladění a obsluha sítě s libovolnou konfigurací. Pro nejjednodušší aplikaci postačí libovolný osobní počítač se sériovým portem COM, jednoduchý převodník 1-Wire/RS232 a volně šiřitelný program od firmy Dallas LogTemp pro snímání teplot.

2.2.2. Technické parametry čidla DS18B20

Rozsah měřené teploty: -55 až 125°C
Maximální odchylka teploty: ±0,5°C v rozsahu -10 až 85°C
Napájení: 3 až 5,5V

Formát dat teploty: 9-12 bitů

2.2.3. Program LogTemp

LogTemp je ovládací program pro čidla Dallas pracující na 1-Wire s běrnici. Teplota je snímána pomocí čidla DS18B20.

Program je určen pro operační systém Windows 98/2000/XP/Vista/7 a je volně šiřitelný s licencí freeware. Je možné jej stáhnout na stránkách [14]. Používám verzi 2.21.0.78. Velikost instalačního souboru je 1MB. Pro funkci tohoto softwaru je zapotřebí nainstalovat ovladač sběrnice 1-Wire (verze 401r2) 5,3MB [15]. Tento ovladač zajišťuje komunikaci mezi sběrnicí 1-Wire a sériovým rozhraním osobního počítače COM - RS232. Základní instalace je v anglickém jazyce. Na stránkách programu LogTemp se nachází i kompletní čeština. Základní práce s LogTemp je velmi intuitivní a zvládne jí i začátečník.

2.2.4. Popis programu LogTemp

Tento software umožňuje sledovat teplotu několika čidel najednou. Ukázka hlavního okna programu je v příloze C.2. Teploty z čidel zobrazuje v dolní liště okna programu. Každému čidlu lze přiřadit svůj vlastní název, barvu a tloušťku čáry, která se bude následně zobrazovat v grafu. K čidlům je také možno nastavit limity minimální a maximální hodnoty. Při jejich překročení je možné nastavit spuštění nějaké aplikace nebo nastavit varovný signál. Snímání teplot se nechá nastavit v intervalu 15sec až 24hod. Volitelně se může zobrazovat jedna vybraná teplota v ikonce „SysTray“ (v pravém dolním rohu obrazovky vedle hodin). Možnost využití různých čidel je bohatý, jedná se výhradně o produkty Dallas.

Jedna podstatná funkce, kterou využívám je ukládání dat. Zde jsou možnosti opravdu široké. Mezi ty základní, nejjednodušší možnosti je ukládání dat do zvolené složky na disk v počítači. Program po sejmutí naměřených hodnot vytvoří/aktualizuje *.csv soubor kde jsou po řádcích naměřená data z čidel. Jedná se o ID čidla, hodnota, datum a čas měření oddělené tabulátorem. Další možnost je ukládání dat z každého čidla zvlášť. Pro každé čidlo se vytvoří samostatný *.csv soubor. V *.csv souboru je na každé řádce vypsán jeden záznam naměřené hodnoty (interval je dán vlastním nastavením). Řádek obsahuje datum, čas a teplotu která má dvě desetinná místa. Tento soubor se dá dále využít pro import do tabulkového kalkulátoru (Excel, Calc) a následně jej zpracovat do grafu – tento způsob ukládání dat nevyužívám.

Další variantou je generování HTML souboru, ukládání na FTP server, odesílání e-mailu atd.

2.2.5. Převodník 1-Wire/RS232

Ke správnému fungování je potřeba mezi sériový port počítače a teplotní čidla vložit převodník. Převodník zajišťuje napájení sběrnice 1-Wire a přenos dat po ní. Pro jednoduchost jsou čidla DS18B20 napájena přímo ze sběrnice pomocí tzv. parazitního napájení. To znamená, že vývody čidla 1 a 3 (GND, VDD) jsou připojeny na GND. Vývod 2 z čidla je připojen na DATA. Propojovací kabel mezi převodníkem a počítačem pak stačí pouze dvouvodičový.

Schéma zapojení (příloha A.2) je podle doporučení výrobce a částečně čerpá ze stránek [16]. Vlastní převodník je jednoduchý. Skládá se ze dvou křemíkových diod D1,D2, dvou zenerových diod ZD1, ZD2 a odporu R1. Výstup na sběrnici 1-Wire je realizován přes konektor RJ-45 na němž jsou využity pouze dva vodiče. Celé zařízení je připájeno na univerzálním plošném spoji a umístěné v plastové krytce.

2.2.6. Rozbočovač 1-Wire

Rozbočovač je použit proto, aby se mohla snímat teplota z více čidel na jedné sběrnici 1-Wire. Tyto rozbočovače jsem použil dva shodné zapojené v sérii. Schéma jejich zapojení je v příloze A.3.

Jsou vyleptány na jednostranné desce s plošnými spoji pro jednoduchou výrobu. Deska má rozměr 100x50mm. Na desce se nachází několik drátových propojek a 5 paralelně zapojených zásuvek RJ-45. Na jeden rozbočovač je možné připojit až 4 čidla. Pokud za tímto rozbočovačem chceme použít další rozbočovač, pak zapojíme pouze 3 čidla. Místo čtvrtého čidla se zapojí přívodní kabel k dalšímu rozbočovači. Toto propojení je vidět na blokovém schématu v příloze.

2.3. Karta s analogovým vstupem - Duomix

Hlavním řídícím prvkem topné soustavy je čtyřcestný ventil Duomix. Duomix je ovládán servem a řídící jednotkou, která je součástí kotle. Podle potřeby tento ventil reguluje teplotu vody do radiátorů a vratnou vodu zpět do kotle. Regulace probíhá tak, aby vratná voda do kotle neklesla pod 60°C a nedocházelo tak k nízkoteplotní korozi a dehtování kotle.

2.3.1. Systém snímání polohy ventilu Duomix

Ventil má úhel otevření v rozsahu 0 až 90° a tomu odpovídá jeho stupnice 0 až 10. Při běžném provozu při venkovních teplotách cca okolo 0°C je ventil otevřen přibližně na polohu č. 4, v mrazivých dnech kdy je venkovní teplota hluboko pod bodem mrazu je otevřen až na 6. Polohu otevření toho ventilu je proto vhodné sledovat v rámci monitorování v počítači při provozu kotle.

Při řešení tohoto úkolu se vyskytl problém, jak co nejjednodušeji a nejsnáze dostat data ze stupnice 0 až 10 do PC ve formě dat. Servo bohužel nemá žádnou možnost zpětné vazby v analogové podobě o poloze otevření. Má pouze možnost sejmutí dvou krajních poloh pomocí koncových spínačů: poloha 0 (zavřeno), poloha 10 (otevřeno). To ale neřeší problém, jak dostat aktuální polohu otevření do PC. Bylo proto nutné doplnit servo o nějaké externí zařízení, které by tuto polohu snímalo. Jako velice vhodné bylo použít průmyslový lineární odporový spínač od výrobce Gefran, model: LT-M-0100-S o hodnotě odporu 5kΩ [13]. Tento snímač se mi podařilo sehnat z vyřazeného stroje určeného k likvidaci. Na rameno ventilu, který má rozsah 0 - 90° je přes kloub přišroubován odporový snímač. Schématické znázornění je v příloze A.5. Při otevírání a zavírání se tato poloha přivádí na lineární odporový snímač, který interpretuje tuto polohu jako odpor 0 – 5kΩ.

Odpor je nutné převést na digitální signál. Z předchozích zkušeností s mikrokontrolérem ATmega8 [22] jsem využil jeho funkcí. Zejména vnitřní 10-bitový A/D převodník a hardwarový UART.

2.3.2. Popis firmwaru karty Duomix

Jako programovací jazyk je v tomto případě použit BASCOM. Bascom je prostředí pro programování aplikací s procesory od firmy AVR, prostředí využívá programovací jazyk BASIC. Bascom je velmi vhodný pro začátečníky v oblasti programování mikrokontrolérů, protože obsahuje ustálené rutiny, které se nemusí pokaždé znovu psát. To má ovšem své výhody i nevýhody. Výhoda tohoto programovacího jazyka je v rutinách, obsažených v knihovnách definovaných výrobcem. Nevýhody jsou ty, že po zkompilování má výsledný soubor .hex asi o 20% větší velikost než u programovacího jazyka assembler. Tato nevýhoda však nemá vliv na moji aplikaci, protože velikost programu v mikrokontroléru je asi 10% kapacity paměti. Schéma zapojení elektrické části je v příloze A.4. Napájení je zajištěno externím síťovým napáječem 5V/500mA.

Program na obsluhu A/D převodníku je velice jednoduchý. Na začátku programu jsou instrukce pro nastavení typu procesoru a jeho knihovny. V ní se nacházejí informace o portech, PWM, AD převodnících, velikost pamětí Flash a EEPROM. Další instrukcí je nastavení frekvence použitého krystalu, zapisuje se v Hz v mém případě 4 MHz. Z této informace program vypočítá dobu trvání jedné instrukce, dobu čekání, rychlost běhu časovačů a podobně. Dále je nastavení rychlosti sériového kanálu na 9600 Bd, deklarace proměnné typu Word (0 - 65535) a konfigurace A/D převodníku - referenční napětí je 5V. Celý program pracuje v nekonečné smyčce Do - Loop. V ní se vždy na začátku sejme aktuální hodnota A/D převodníku. Jelikož se jedná o desetibitový převodník, tak hodnota napětí na jezdci potenciometru 0V odpovídá hodnotě digitálního signálu 0 dekadicky. Hodnota napětí 5V odpovídá hodnotě 1023. Pokud se jedná o desetibitový převodník tak počet stavů se vypočte podle vzorce (1):

$$počet\ stavů = 2^n = 2^{10} = 1023 \quad \textbf{(1)}$$

Takže například pokud bude měřené napětí jezdce 2V, bude po převodu v dekadické interpretaci 409. Výpočet je zřejmý ze vzorce (2).

$$A = \frac{2^n}{U_{ref}} . U_{jezdce} = \frac{2^{10}}{5} . 2 = 409 \quad \textbf{(2)}$$

Hodnota A/D převodu se uloží do proměnné A. Pomocí příkazu print se obsah proměnné odešle na sériovou linku. Poté se čeká 3s a celý cyklus se opakuje stále dokola.

3. Serverová aplikace

Serverová aplikace spolu s aplikací klient pro Windows využívá vývojové prostředí Delphi 7 a databázový systém PostgreSQL.

3.1. Databázový systém PostgreSQL

Obecně lze o databázovém systému říci, že se jedná o standardizovaný dotazovací jazyk používaný pro práci s daty v relačních databázích. Relační databáze je založena na tabulkách, jejichž řádky jsou záznamy a jednotlivé sloupce v nich uchovávají informace o relacích mezi jednotlivými záznamy. SQL je zkratka anglických slov Structured Query Language (strukturovaný dotazovací jazyk).

Práce s databázemi je z principu odlišná od programů, se kterými se běžně setkáváme. Například v tabulkovém kalkulátoru pracujeme s určitým souborem, se kterým postupně pracujeme, přidáváme hodnoty do tabulek, zvýrazňujeme důležité části a podobně. Nakonec tyto informace vytiskneme na tiskárně. Práce s databázemi je poněkud jiná. Databáze většinou obsahuje nějaká pro nás zajímavá data, která hodláme dále zpracovávat. Potřebujeme je tedy z databáze nějakým způsobem přehledně získat. Postupujeme tak, že databázi zadáme dotaz, kterým popíšeme informaci, kterou chceme získat a po chvíli se nám na obrazovce objeví výsledek tohoto dotazu. Výsledek může být buďto správný a získali jsme požadovaná data, nebo chybný a musíme dotaz upravit a lépe specifikovat pro nalezení námi požadovaného cíle. [7]

PostgreSQL je plnohodnotným relačním databázovým systémem s otevřeným zdrojovým kódem. Má za sebou více než patnáct let aktivního vývoje a má vynikající pověst pro svou spolehlivost a bezpečnost. Běží na všech rozšířených operačních systémech včetně Linuxu, UNIXů, Mac OS X, Solaris a Windows. Plně podporuje cizí klíče, operace JOIN, pohledy, spouště a uložené procedury. Obsahuje většinu datových typů, např. INTEGER, NUMERIC, BOOLEAN, CHAR, VARCHAR, DATE, INTERVAL a TIMESTAMP. K systému existuje kvalitní volně dostupná dokumentace včetně českých překladů FAQ. Výkonnostně PostgreSQL nezaostává za srovnatelnými komerčními systémy a v častokrát je i předčí. [10]

PostgreSQL je šířen pod BSD open source licencí. Tato licence umožňuje neomezené bezplatné používání, modifikaci a distribuci PostgreSQL a to ať pro komerční nebo nekomerční využití. PostgreSQL se může šířit i se zdrojovými kódy, zdarma nebo komerčně.

PostgreSQL umožňuje běh procedur napsaných v několika programovacích jazycích v Perlu, Pythonu, v jazyku C, JDBC, ODBC, dbExpress, Open Office, PHP, .NET a Delphi. [10]

3.1.1. Struktura tabulek

Pro ukládání dat jsem v databázi založil dvě tabulky. První tabulka má název „cidla“ a druhá „historie“.

3.1.1.1. Tabulka Čidla

Tabulka čidla se skládá z celkem devíti sloupců, její struktura je znázorněna v Tab. 3.1. Tato tabulka slouží k zaznamenání posledních naměřených hodnot (krátkodobá historie). V tabulce jsou dále uloženy informace o ID čidel, popis čidel, jejich hodnota, kritické hodnoty čidel (jejich význam bude vysvětlen v kapitole Klientská aplikace - Windows) a datum posledního měření.

1	2	3	4	5	6	7	8	9
ciddruh	cidcislo	cidid	cidpopis	cidhodnota	ciddatum	cidminimum	cidstred	cidmaximum
TEPLOTA	1	1B000001392ED528;	teplota kotle	79,03	2012-04-20 19:23:39	65	80	90
VSTUP	3		HDO (noční proud)	1.00	2012-04-20 09:10:00	0	0	0
ANALOG	1		duomix	5.00	2012-04-20 19:18:20	3	6	7
...								
...								

Tab. 3.1: PostgreSQL tabulka „cidla“

První sloupec se jmenuje ciddruh, je datového typu character varying a délka znaků je 16. Nese informaci o druhu čidla. Prakticky se v tomto sloupci objevují pouze tři textové hodnoty (teplota, vstup a analog). Další sloupec cidcislo je typu integer, v něm je pořadové číslo čidla. Je zde 9 teplotních čidel, 8 digitálních vstupů (číslovány od 1) a 1 analogový vstup. Sloupec cidid je vyplněn pouze pokud ciddruh=teplota, jeho typ je character varying (16) a jsou v něm uloženy unikátní šestnáctimístná identifikační čísla teplotních čidel DS18B20. Sloupec cidpopis slouží ke stručnému popisu jednotlivých čidel, tento popis se využívá v klientských aplikacích pro snadné rozeznání toho k čemu, které čidlo slouží. Cidhodnota je desetinné číslo typu numeric, celková délka je 5 čísel s přesností na setiny. V tomto sloupci jsou uloženy teploty, stav analogového vstupu 0 až 10 a digitální vstupy,

které mohou nabývat pouze dvou hodnot 1 nebo 0. Sloupec ciddatum je typu timestamp without time zone a je v něm datum a čas ve formátu „rrrr-mm-dd hh:mm:ss“. Sloupce cidminimum, cidstred a cidmaximum jsou typu integer a jejich číslo určuje při jaké hodnotě se změní v klientské aplikaci pozadí pod jednotlivými čidly (4 možné barvy: modrá, zelená, žlutá a červená) dále slouží k zobrazení hlášek ve stavovém okně.

Při zápisu do této tabulky se využívá příkazu UPDATE a dotaz vypadá takto: „UPDATE cidla SET cidhodnota=79.03, ciddatum='2012-04-20 19:23:39' WHERE cidcislo=1 and ciddruh='TEPLOTA'“. Tento příkaz aktualizuje data (cidhodnota a ciddatum) v tabulce cidla řádek který obsahuje sloupec cidcislo=1 a zároveň ciddruh=TEPLOTA. Takto nedojde k zápisu další řádky, ale pouze se modifikuje již existující řádka. Takto malá tabulka, která obsahuje pouze asi 18 řádků je velmi rychle zpracována a jednoduše se zní vyčtou potřebné údaje. Pokud potřebujeme sáhnout do databáze pro větší počet záznamů například pro vykreslení grafu nebo seznamu hodnot využijeme následující tabulku historie.

3.1.1.2. Tabulka Historie

Tabulka historie má oproti tabulce čidla mnohem méně sloupců. Slouží především jako uložiště naměřených hodnot. Komunikuje se s ní v případě zápisu nově naměřeného údaje, nebo když je potřeba číst historii měření. Struktura tabulky historie je znázorněna v Tab. 3.2.

1	2	3	4
histdruh	histcislo	histhodnota	ciddatum
TEPLOTA	1	79,03	2012-04-20 19:23:39
VSTUP	3	1.00	2012-04-20 09:10:00
ANALOG	1	5.00	2012-04-20 19:18:20
...			
...			

Tab. 3.2: PostgreSQL tabulka „historie“

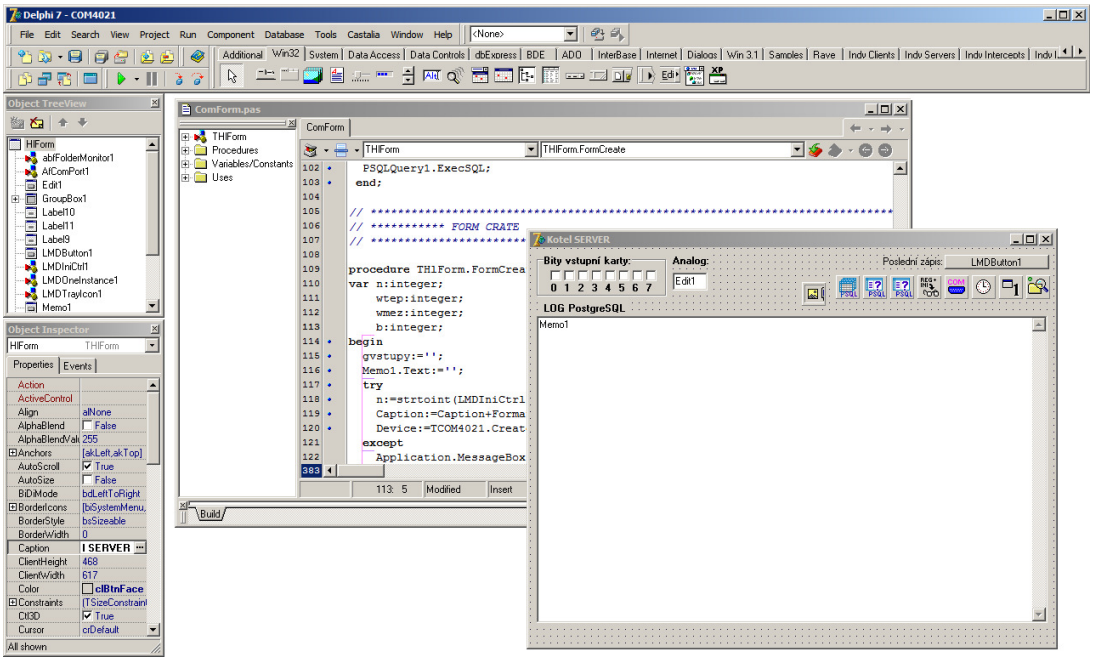
Obsahuje čtyři sloupce, první sloupec je obdobný sloupci ciddruh a může nabývat pouze tří hodnot. Druhý sloupec histcislo nese informaci o čísle čidla. Třetí sloupec hist hodnota je typu numeric (5,2) a obsahuje informaci o hodnotě jednotlivých čidel. Poslední sloupec ciddatum slouží k zápisu data a času měření.

Zápis do tabulky je prostřednictvím příkazu INSERT: „INSERT INTO historie (histdruh, histcislo, histhodnota, histdatum) VALUES ('TEPLOTA',7,20.56, '2012-04-30 19:29:20')“. Do tabulky s názvem historie zapíše do slopců: histdruh, histcislo, histhodnota a histdatum hodnoty umístěné v následné závorce, data jsou oddělená čárkou po sloupcích.

3.2. Vývojové prostředí Delphi

Při vytváření serverové aplikace a aplikace klienta pod operačním systémem Windows jsem využil vývojové prostředí Delphi 7 od firmy Borland. Delphi je vyšší programovací jazyk tzn., že jako příkazy využívá struktury podobné přirozenému jazyku. Delphi je vlastně kompilátor jazyka Object Pascal s integrovaným vizuálním prostředím a jedná se tedy o upravenou verzi původního jazyka Pascal.

Programování v Delphi je velmi intuitivní a jednoduché. V Pascalu se příkazy provádějí v takovém pořadí jak byli napsány ve zdrojovém kódu. Programovací jazyk Object Pascal stejně tak jako jiné programovací jazyky obsahuje mimo jiné také svůj vlastní kompilátor. Kompilátor slouží k přeložení námi psaného kódu do strojového kódu, který využívá počítač ke spuštění samotného programu. Kompilátor v Delphi je velmi rychlý a spolu s intuitivním vytvářením programů se jedná o kvalitní vývojový nástroj pro začínající i pokročilé programátory. Delphi také generuje značnou část kódu automaticky, takže programátor prakticky sestavuje pouze samotné tělo programu. O zbytek se postará prostředí Delphi. [1]



Obr. 3.1: Vývojové prostředí Delphi a serverová aplikace.

Pro vývoj v Delphi jsou k dispozici rozsáhlé knihovny komponent. Jedná se například o klasické obdélníkové šedivé Windows tlačítko s nějakým textem a funkcí. Delphi tak přebírá jeho funkčnost – schopnost reagovat na klepnutí myši, zobrazovat text, měnit barvu, rozměry atd. Takto vytvořený objekt se nazývá komponenta. Jednotlivé komponenty jsou uložené v knihovně komponent. Delphi má v základu velké množství knihoven předinstalované, další

se nechají jednoduchou instalací přidat. V mém případě se jedná o knihovnu pro práci se sériovou linkou RS-232 s názvem AfComPort a PSQLDatabase pro komunikaci s databází PostgreSQL. Pomocí komponent lze tedy vytvářet plnohodnotné aplikace pro operační systém Windows.

Delphi umožňuje vytvářet vlastní DLL (Dynamicky linkované knihovny), které obsahují libovolné komponenty a je možné je využít i v dalších Delphi aplikacích, nebo i v jiných aplikacích vyvíjených v jiném programovacím prostředí. Delphi také mimo jiné obsahuje interní kontrolu chyb a jejich ošetření. Ta je prováděna pomocí výjimek. V praxi to vypadá tak, že se píše kód předpokládající kladný průběh všech příkazů. Pokud příkaz neuspěje, tak je vygenerována výjimka a tu je možné ošetřit v obsluze výjimky s minimálním úsilí na testování výsledku. [3]

3.2.1. Princip snímání dat

Serverová aplikace slouží k získávání dat od jednotlivých snímačů celého systému a ukládání jednotlivých hodnot do databáze. Aplikace je kompletně napsána v jazyce Delphi 7 s propojením na databázi PostgreSQL [10], kterou je možné používat zdarma bez jakéhokoliv omezení i pro komerční využití. Jako server je použit starší notebook Acer s operačním systémem Windows XP Professional, který je neustále zapnutý, běží na něm databáze PostgreSQL, IIS server s PHP, Program LogTemp a serverová aplikace monitoringu teplot.

Pro běh serverové aplikace je zapotřebí dvou sériových COM portů (jeden pro kartu digitálních vstupů a druhý pro analogovou kartu). Po spuštění aplikace je vyžadován *.ini soubor s názvem kotel.ini. Ten nese informace o číslech jednotlivých COM portů, dále přiřazení názvu čidla k jeho identifikačnímu číslu ID. Například se přiřadí název „cidlo_1“ k ID: „1B000001392ED528“ . Takže do databáze se dále uloží cidlo_1, jeho hodnota a datum měření

3.2.1.1. Digitální vstupy 8 vstupová karta

Aplikace pro snímání digitálních vstupů je poměrně jednoduchá. Stav jednotlivých vstupů se zobrazuje v hlavním okně ve formě checkboxů. Aplikace dále obsahuje časovač, který zajišťuje kontrolu vstupů každých 100ms. Údaje o čísle COM portu jsou uloženy v souboru kotel.ini, který je umístěn ve stejném adresáři jako spustitelný *.exe soubor.

Část programu, která zpracovává jednotlivé vstupy se skládá z několika procedur. Jsou to procedury PS, CLK, VCC a funkce DIN a GetData, které se starají o ovládání vývodů P/S a

CLK. Funkce DIN čte stav vývodu Q7 a slouží k přečtení celého datového bajtu. Funkce GetData slouží k přečtení osmi bitů na vstupu karty, přičemž nejdříve nastaví CLK = 0 a P/S = 1 tímto se stav vývodů P0 až P7 uloží do posuvného registru, po krátké pauze se obvod přepne do režimu čtení P/S = 0. Nyní se funkcí DIN dá sejmout stav vývodu P7, ten se uloží do pomocné proměnné a následně se přivede hodinový impuls na vývod CLK a tím se bity posunou a dalším krokem tohoto cyklu se přečte hodnota P6 a tak dále až do bitu P0. Přijaté bity se v pomocné proměnné PomData posouvají vlevo je to tím, že nejdříve přijímáme nejvýznamnější bity (MSB) a poté až méně významné bity (LSB).

Dále se využívá funkce DecodeBits, ta má dva parametry, jeden parametr je jméno datového Byte a druhá číslo bitu kterou potřebujeme zjistit. Funkce vrací buď hodnotu true nebo false. Takto lze jednoduše zjistit stav jednoho z osmi bitů 8 vstupové karty. Bližší popis 8 vstupové karty je v [2].

Zpracování a ukládání dat probíhá následovně: Při spuštění programu se uloží aktuální stav vstupů do databáze a zároveň do pomocné proměnné. Po odčasování 100ms dojde k druhé kontrole stavů vstupů. Tento stav se porovnává bit po bitu s předchozím stavem z pomocné proměnné. Pokud nalezne změnu, tak zapíše nový stav do databáze s aktuálním datem a časem. Na závěr nový stav uloží do pomocné proměnné a je připraven na další kontrolu vstupů. Tímto krokem se v každém taktu uspoří místo v databázi (pokud se změnil pouze jeden bit tak, ostatní se do databáze zapisovat nebudou a zapíše pouze bit, který byl změněn).

3.2.1.2. Teploty

Pro snímání naměřených teplot se využívá program LogTemp. Tento software ukládá naměřené teploty do *.csv souboru v intervalu, který se nechá nastavit. V mém případě každou 1 minutu. Jednou za minutu se tedy tento soubor přepíše.

Uvnitř *.csv souboru první řádek je hlavička, která vypadá následovně: "ID","Value","dd.mm.yyyy","hh:nn:ss","Type",. Každému čidlu tedy připadá jedna řádka, která obsahuje identifikační číslo, naměřenou hodnotu v desetinném tvaru s přesností na setiny. Dále datum měření, čas a typ čidla zda se jedná o teploměr, vlhkoměr, atd. Jednotlivé položky na řádku jsou oddělené čárkou, řetězce jako je ID a type jsou navíc umístěny v uvozovkách "".

Serverová aplikace kontroluje atributy tohoto *.csv souboru. Kontrola atributů je prováděna každou vteřinu. Kontroluje se pouze datum a čas. Když dojde ke změněně času souboru tzn. že došlo k měření a uložení teplot. Aplikace počká 100ms aby se bezpečně uložila všechna data a poté provede otevření tohoto souboru a vyčte podle ID jednotlivých čidel jejich naměřené hodnoty. Tyto hodnoty poté porovná s předchozím stavem, a pokud došlo ke změně tak novou teplotu zapíše do databáze s datem změny *.csv souboru. Pokud nově naměřená teplota byla shodná s předchozím měřením tak se nezapíše do databáze a zůstává uložen její předchozí stav.

Zpracování samotného *.csv souboru probíhá tak, že se postupně načítají jednotlivé řádky. Pokud

3.2.1.3. Analogový vstup

Přenos do počítače probíhá prostřednictvím sériové linky. Přenos probíhá rychlostí 9600Bd, 8 bitů, řízené toku žádné, parita žádná a jeden stop bit. V Delphi je pro tuto komunikaci vyhrazena komponenta AfComPort. V okně Object inspector se nadefinují parametry přenosu. Číslo com portu se načte při spuštění aplikace z *.ini souboru. Po náběhu aplikace se zároveň tento port otevře pro komunikaci.

Pokud dojde k přijetí znaku po sériové lince, vyvolá se událost AfComPort1DataRecived. Analogová karta jednou za 3 vteřiny provede A/D převod na vstupní bráně převodníku. Změřená hodnota může nabývat hodnot 0 až 1023, protože se jedná o 10-bitový převodník. Tato data odesílá po sériové lince a na konci vloží dva znaky ukončující přenos. Jde o line-feed (LF #10) a carriage retun (CR #13). Data jsou tedy

přijímána například ve formátu: „565 CR LF“. Přijatý znak se přiřadí do pomocné proměnné ws. Následně se vyhodnotí podmínka, pokud není přijímaný znak CR, tak ještě zkontroluje, zda-li to náhodou není LF. Pokud není, tak do předem vynulované globální proměnné analog přidá přijatý znak. Takto se postupně přijmou čísla 5 6 5 a v proměnné analog nyní bude číslo „565“. V dalším kroku je přijat znak CR, ten je podmínkou ignorován, následně ale přijde znak LF (konec řádky), proběhne vyhodnocení podmínky a podmínka přejde do bloku else. V této chvíli víme, že prom. analog obsahuje námi požadovanou hodnotu A/D převodníku 565. Nyní je potřeba zkonvertovat hodnotu A/D převodníku (0 až 1023) a stupnici duomixu (0 až 10). Lineární konverze přepočtem zde není možná, protože snímací rezistor zcela nevyužívá celý svůj rozsah a navíc dráha, kterou jezdec urazí, odpovídá kruhové výseči páky ventilu duomix. Pokud by se provedl přepočet změřené hodnoty a porovnal se s reálnou stupnicí tak, by neodpovídal reálné hodnotě. Proto jsem se rozhodl provést ruční konverzi, tu jsem provedl tak, že jsem ručně pohyboval pákou duomixu a přitom si zapisoval jednotlivé polohy stupnice. Zároveň jsem měl připojen A/D převodník program hyperterminál jsem zapisoval odpovídající číslo převodníku. Tímto způsobem vznikla převodní tabulka s jedenácti úrovněmi (Tab. 3.3).

Stupnice duomix	**Hodnota A/D**
0	0
1	625
2	730
3	765
4	837
5	864
6	880
7	910
8	917
9	928
10	932

Tab. 3.3: Naměřené hodnoty odpovídající stupnici duomix a A/D převodníku

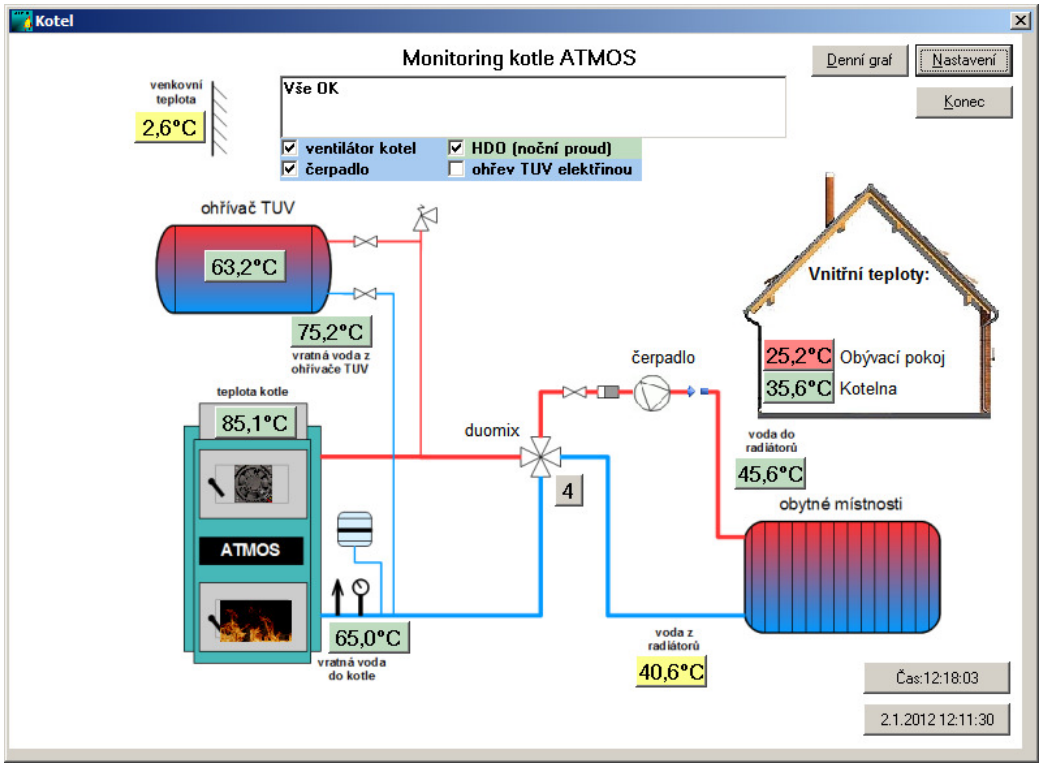
Následně jsem tyto hodnoty vložil do konfiguračního *.ini souboru a následně využil v programu v delphi. Pokud se tedy vrátíme zpět k programové konverzi tak to v reálném programu vypadá následovně. Cyklem for se postupně kontrolují jednotlivé meze podle následující podmínky: pokud hodnota proměnné analog je větší nebo rovna mezi z *.ini souboru a zároveň menší než mez následující, tak do proměnné analog_zapis vlož číslo pomocné proměnné z cyklu for. Po dokončení cyklu for se ještě vyhodnotí hodnota

analog_zapis se stavem předchozího měření. Pokud byla stejná, tak se do databáze zapisovat nebude. Pokud je různá, tak tuto hodnotu program vypíše do okna serverové aplikace k položce „Duomix“, zkonvertuje aktuální datum do formátu potřebného pro databázi, vloží ho do kolonky „poslední zápis“ a zapíše novou hodnotu do databáze.

4. Klientská aplikace – Windows

4.1. Popis aplikace pro Windows

Celá aplikace pro Windows je podobně jako serverová aplikace vytvořená v programovacím jazyce Delphi 7. Aplikace se skládá z hlavního okna (Obr. 4.1) a třech dalších pod oken. Jedná se o krátkodobou historii jednotlivých čidel – toto okno se zobrazí po kliknutí na jedno tlačítko z devíti teplot. Druhé okno slouží k nastavení barev pozadí pro jednotlivá čidla. Třetí okno s názvem „Denní graf“ načte historii naměřených hodnot pro dnešní den a dále umožňuje změnit den a zpětně tak zobrazit grafickou historii.

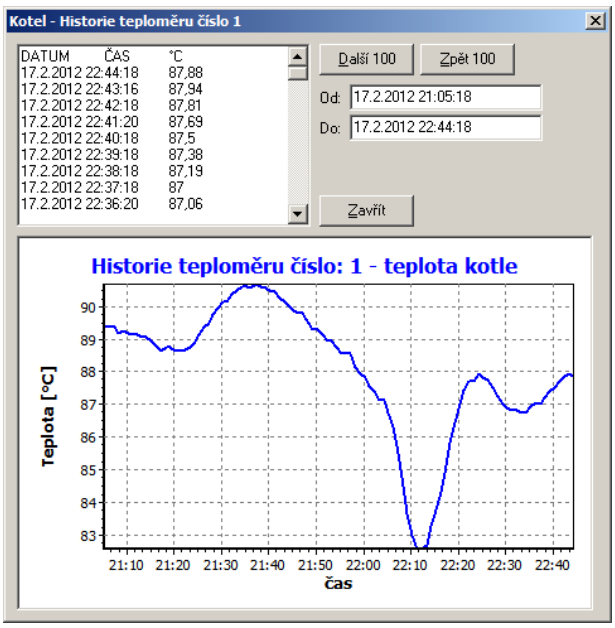


Obr. 4.1: Hlavní okno aplikace Kotel.

Hlavní okno se skládá z několika hlavních částí. Na pozadí aplikace je vložen obrázek – schéma topné soustavy (kresleno v OpenOffice Draw). Zobrazuje topení tak jak je skutečně zapojené. Nahoře uprostřed je stavový rámeček, kde se na základě změřených teplot zobrazuje, zda je vše v pořádku. Pokud není tak se v něm mohou zobrazovat varovná hlášení, například „V ohřívači TUV je nízká teplota vody“, „Zvýšená teplota kotle!“ a podobně. Pokud je zobrazeno chybové hlášení tak pozadí stavového řádku střídavě bliká červená/bílá barva, pokud je vše v pořádku pozadí je trvale bílé. Pod tímto stavovým rámečkem jsou celkem čtyři tzv. checkboxy (zaškrtávací políčka) ty slouží pro signalizaci běhu ventilátoru

kotle, čerpadla, dále pak zda-li je přítomen signál HDO (hromadné dálkové ovládání) signalizující nízkou sazbu elektřiny při které se spíná termostat tělesa v ohřívači TUV a ohřívá se jím užitková voda, pokud kotel není v provozu – například v letních měsících. Ohřívání TUV elektřinou je signalizováno čtvrtým checkboxem.

Pro zobrazení teplot jsou použitá tlačítka, jejichž popis obsahuje hodnotu zjištěnou při posledním měření. Barva pozadí tlačítek se mění v závislosti na velikosti teploty a nastavení jednotlivých mezních hodnot v tabulce v databázi. To uživateli umožňuje vizuálně sledovat, jestli je daná teplota v toleranci (například pokud dojde k přetopení kotle a jeho teplota bude více než 90°C tak pozadí tlačítka bude červené a ve stavovém rámečku bude hláška o přetopení kotle) Tyto hodnoty lze pohodlně měnit v nastavení aplikace Kotel, který je ukládá do databáze do tabulky Čidla. Tlačítka s teplotami mají ještě jeden účel. Pokud na ně uživatel klikne, tak se zobrazí okno (Obr. 4.2),

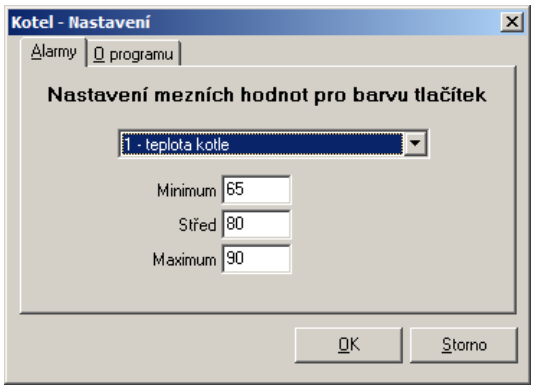


Obr. 4.2: Okno krátkodobé historie jednoho teploměru, aplikace klient.

ve kterém se zobrazí graf a seznam posledních 100 naměřených hodnot, což odpovídá při intervalu měření 1 minuta historii cca 1h 40min zpět. Časový interval lze dále měnit pomocí tlačítek „Další 100“ a „Zpět 100“. Toto okno lze ukončit buď tlačítkem zavřít, standardně kliknutím na křížek nebo klávesou Esc. Graf lze také přibližovat a oddalovat. To se děje prostřednictvím myši, levým tlačítkem se klikne na začátek výřezu v grafu a táhne se libovolným směrem, po uvolnění tlačítka se vybraná oblast zvětší (podobně jako je tomu při označování více ikon na ploše Windows). Pokud chceme přiblížení oddálit, použijeme stejný

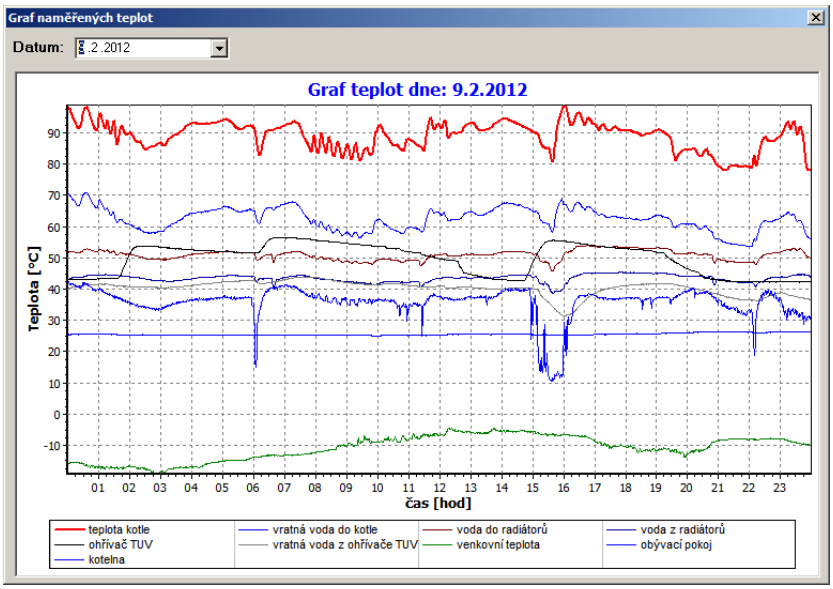
postup, ale tentokrát s pravým tlačítkem myši. Seznam hodnot je také možné kopírovat do tabulkového kalkulátoru pro další zpracování.

Pokud se vrátíme k hlavnímu oknu (Obr. 4.1), tak zde je ještě jedno tlačítko vedle symbolu čtyřcestného ventilu duomix. V něm se zobrazuje jeho stupnice otevření jako číslo od 0 do 10. V pravém dolním rohu se nalézají další dvě tlačítka. Prvním z nich je aktuální čas a druhým čas posledního měření. V pravém horním rohu je tlačítko pro ukončení programu a tlačítko nastavení. Po kliknutí na tlačítko nastavení se zobrazí okno Kotel – nastavení (Obr. 4.3). Obsahuje dvě záložky, jedna nese informace o autorovi programu a druhá s názvem alarmy mění tři rozhodovací úrovně teplot. Podle těchto hodnot se mění barva pozadí tlačítek s velikostí teplot, jak bylo popsáno výše. V rozbalovacím seznamu se vybere jedno z devíti čidel a v popisových polích pod tímto seznamem se jednoduše přepíše požadovaná rozhodovací hodnota. Zadávání je omezeno tak, aby bylo možné zadávat pouze celé číslice a případně znaménko „-“ . Po zadání je potřeba ještě kliknout na tlačítko OK pro uložení. Pokud se tak neučiní a klikne se na křížek pro zavření okna, tak se program dotáže, jestli má hodnoty uložit. Tyto hodnoty jsou ukládány do databáze.



Obr. 4.3: Okno nastavení aplikace klient.

Posledním tlačítkem v hlavním okně je tlačítko s názvem „Denní graf“. Po kliknutí se otevře okno graf naměřených hodnot dnešního dne (Obr. 4.4). Vlevo nahoře je možné zvolit libovolný den. Po vybrání data se graf automaticky překreslí na požadovaný den. Je zde přehledně zobrazeno všech devět teploměrů za 24 hodin. V hlavičce grafu je, datum ze kterého měření pochází, na vodorovné ose je zobrazen čas v hodinách. Na svislé ose pak teplota ve °C. Na spodní straně grafu je rámeček s legendou a popisem jednotlivých čar. V grafu opět funguje možnost přiblížení, jako je tomu u grafu jednotlivých čidel. Okno je možné ukončit buď klávesou Esc nebo kliknutím na křížek.



Obr. 4.4: Okno denní graf aplikace klient.

Pro lepší grafické vyjádření znázornění běhu programu jsem využil možnosti vkládat do aplikace animované obrázky. Pokud například teplota kotle bude větší než 65°C, tak se ve schématu topné soustavy v místě, kde jsou zakresleny spodní dvířka kotle Atmos objeví animovaný plápolající plamínek. Pokud bude teplota menší než 65°C tak se obrázek plamínku skryje. Obdobně je tomu u čerpadla – pokud je čerpadlo v provozu, tak se schematická značka čerpadla začne otáčet dokola a za čerpadlem se pohybuje šipka symbolizující pohyb vody v topném okruhu. Pokud je ventilátor pro roztápění kotle v provozu, tak se opět otáčí obrázek ventilátoru na horních dvířkách kotle. Jedná se o vizuální prvky, které spolu se změnou barev pozadí tlačítek zatraktivňují jinak šedivé okno aplikace.

4.2. Hlavní okno programu

Program kotel se skládá celkem z pěti formulářů. V prvním Form1 je umístěn kód hlavního okna (Obr. 4.1), druhý formulář Form2 obsahuje okno krátkodobé historie jednotlivých teploměrů (Obr. 4.2). Ve třetím Form3 je kód okna pro nastavení mezních hodnot pro barvu pozadí tlačítek (Obr. 4.3), čtvrtý Form4 obsahuje okno denního grafu (Obr. 4.4) a v posledním Form99 jsou umístěny funkce používané ve všech formulářích.

4.2.1. Příkazy po spuštění aplikace Kotel

Zdrojový kód hlavního okna je umístěn v souboru Form1.pas. Po spuštění programu nejprve proběhne načtení *.ini souboru, ve kterém jsou přihlašovací údaje do databáze. Po načtení se program připojí do databáze prostřednictvím funkce HlavickaSQL1, pošle dotaz

„SELECT * FROM cidla ORDER BY cidcislo“. Ten zajistí načtení všech dat z tabulky Čidla, přičemž data budou seřazena podle sloupce cidcislo vzestupně. Tento krok vede k načtení ID čísel, které se zobrazí při najetí šipky myši na jednotlivé teploty a dále se provede načtení mezních hodnot pro rozdělení barevného pozadí tlačítek. Jakmile program obdrží data, přejde do cyklu while, který bude probíhat pořád dokola, dokud nebude splněna podmínka, která testuje zda-li již byla vyčerpána všechna načtená data z tabulky Čidla. Cyklus tedy načítá řádku po řádce a přiřazuje obsah sloupců do lokálních proměnných (jedná se o proměnné: druh, číslo, id, minimum, stred, maximum a jim přiřazuje sloupce: ciddruh, cidcislo, …). Po vyčtení jedné řádky program skočí do rozhodovací podmínky IF, kde se testuje jestli proměnná druh obsahuje text ‚TEPLOTA‘. Pokud ano, tak do struktury nastaveni.alarmy.minimum typu pole na pozici [cislo cidla] přiřadí hodnotu proměnné minimum. Obdobným způsobem přiradí nastaveni.alarmy.stred a nastaveni.alarmy.maximum. V dalším kroku do struktury nastaveni.teplomer[cislo cidla] přiřadí ID číslo. Následuje příkaz case který podle proměnné „cislo“ přiřadí do hintu tlačítka (hint je popis po najetí myši nad určitý objekt - tlačítko) podle čísla čidla jeho ID číslo. Tímto způsobem se zpracují další řádky přijaté z databáze.

Dalším krokem je načtení animovaných obrázků *.gif do komponenty AdvPicture1 až 4 ze složky img, která je ve stejném umístění jako program kotel.exe. Ručně jsou ještě nastaveny vlastnosti okna jako je velikost, vypnutý posuvník, načtení pozadí, na kterém je obrázek topné soustavy, ikona v levém horním rohu atd. Tímto je provedena základní inicializace po spuštění programu.

4.2.2. Princip načítání a zpracování dat z databáze

V programu běží jeden časovač, který každých 1000ms vyvolá proceduru s názvem „TForm1.Timer1Timer“. V této proceduře se nachází hlavní dění programu Kotel, protože každým vyvoláním zkontroluje stav v databázi a pokud nalezne změnu, tak jí zpracuje a zobrazí v hlavním okně aplikace.

Nyní blíže popíši dění v proceduře vyvolané timerem1. Po vyvolání této události se do tlačítka v pravém dolním rohu vloží aktuální čas. Poté se testuje podmínka, která souvisí se stavovým rámečkem v horní části aplikace. Pokud se v tomto rámečku na libovolné pozici nenachází „!“ tak pozadí toho rámečku nastaví na bílé to značí že stav systému je ok. Pokud v rámečku z předchozího taktu zůstala varovná hláška, tak vždy obsahuje vykřičník a potom se testuje podmínka jestli sekunda, která právě proběhla, byla lichá, tak nastaví pozadí na bílé,

pokud byla sudá, tak nastaví pozadí na červené. Tímto se zajistí periodické blikání pozadí při varovném hlášení ve stavovém rámečku. Dále proběhne pomocí funkce HlavickaSQL1 nový dotaz do databáze stejný jako při spuštění programu a přiřazení hodnot ze sloupečků jednotlivým proměnným. Do tlačítka poslední aktualizace se vloží datum posledního načteného záznamu z databáze.

Poté se vyhodnotí tři podmínky podle proměnné druh jestli obsahuje název teplota, analog nebo vstup. Nejprve se vyhodnotí, jestli je proměnná druh ‚TEPLOTA', jestliže ano proveď příkaz (teplota[cislo]:=strtofloat(hodnota);) pro vysvětlení: do proměnné „teplota", která je typu pole na číslo pozice, která je dána číslem z proměnné „cislo" přiřaď proměnnou „hodnota", která je přetypovaná z formátu string do formátu float. Dalším příkazem je cyklus FOR, který zkontroluje všechny komponenty v programu a hledá pouze tlačítka s názvy btnTeplota1 až btnTeplota9, ty nalezne tak, že prohledává názvy komponent, konkrétně jejich prvních 10 znaků jestli obsahuje název „btnTeplota". Pokud ano, tak do pomocné proměnné „cislo" vloží 11. znak z názvu této komponenty a to je klíč mezi číslem tlačítka a číslem čidla z databáze, čili takto se například do tlačítka s názvem btnTeplota5 přiřadí správná hodnota čidla číslo 5. Po přiřazení hodnoty teploty do popisu tlačítka je nutné ještě přiřadit barvu pozadí tlačítka, k tomu slouží funkce s názvem tep2tep. Jako parametry jsou do ní posílány čtyři čísla, první je typu real a zbylé tři typu integer. První je desetinné číslo – teplota a zbylá tři čísla byla načtena z databáze při startu programu (čísla minimum, střed a maximum). Ve funkci tep2tep jsou tyto tři meze porovnány s teplotou a výsledkem funkce tep2tep je číslo barvy, která se vloží na pozadí.

Další v pořadí je vyhodnocení, jestli druh čidla je „ANALOG". Protože analogové čidlo je v systému pouze jedno a to čidlo snímače duomix, tak následné přiřazení velice jednoduché. Pouze se vyhodnotí podmínka, jestli se skutečně jedná o analogové čidlo s číslem 1, pokud ano tak do popisu tlačítka btnAnalog1 vlož jeho hodnotu.

Poslední se testuje, jestli druh="VSTUP". Pokud tomu tak opravdu je, tak proběhne větvení příkazem case. Víme, že čidla digitálních vstupů máme využitá celkem čtyři. Příkaz case se bude větvit podle hodnoty proměnné „cislo". Pro jednotlivá digitální čidla bude různé vyhodnocení, proto tedy postupně. Digitální vstup č. 1 (čerpadlo): Pokud jeho hodnota bude nulová, tak se zaškrtne checkbox (zaškrtávací tlačítko), zapne animace symbolizující běh čerpadla a šipku ukazující proudění vody v systému. Když bude mít hodnotu „1", tak odstraní zaškrtnutí checkboxu, vypne animaci čerpadla a skryje obrázek šipky proudění vody. Digitální vstup č. 2 (ventilátor kotle): Pokud jeho hodnota bude nulová, tak se zaškrtne

checkbox, spustí animaci běžícího ventilátoru. Když bude mít hodnotu „1“, tak odstraní zaškrtnutí checkboxu, vypne animaci. Digitální vstup č. 3 (signalizace HDO): Pokud jeho hodnota bude nulová, tak se zaškrtne checkbox a nastaví pozadí popisu checkboxu na zelenou. Když bude mít hodnotu „1“, tak odstraní zaškrtnutí checkboxu a změní pozadí popisu na defaultní modrou. Digitální vstup č. 4 (ohřívání TUV): Pokud jeho hodnota bude nulová, tak se zaškrtne checkbox a nastaví pozadí popisu checkboxu na červenou. Když bude mít hodnotu „1“, tak odstraní zaškrtnutí checkboxu a změní pozadí popisu na defaultní modrou.

4.2.3. Vyhodnocení varovných hlášení

Po vyhodnocení všech řádek databáze ještě musí dojít k vyhodnocení případných chyb a jejich zobrazení ve stavovém rámečku. Nejprve se provede vymazání všeho, co chybový rámeček obsahoval. Jako první je otestována teplota v ohřívači teplé užitkové vody, když bude voda studenější než 50°C tak se ve stavovém rámečku objeví „V ohřívači TUV je nízká teplota vody.“. Další je testování, jestli není zavzdušněný ohřívač teplé užitkové vody. Pokud bude rozdíl teplot mezi vodou v ohřívači a vratnou vodou z ohřívače větší nebo roven 20°C za podmínky, že bude teplota kotle větší než 65°C, tak je velmi pravděpodobné, že bude ohřívač zavzdušněný. Voda skrz výměníkovou spirálu ohřívače necirkuluje tak, jak by měla (Je to způsobeno nedokonalým technickým řešením topné soustavy, ohřívač je umístěn v kotelně až u stropu a není tedy možné namontovat automatický odvzdušňovací ventil, proto je potřeba jednou za čas tento okruh odvzdušnit manuálně přímo u ohřívače. Program Kotel tuto činnost značně zjednodušil, protože uživatel snadno na první pohled rozpozná, kdy je potřeba zakročit v čas. Před zavedením tohoto sledování se stávalo zejména v zimních měsících, že při zavzdušnění ohřívače nebyla teplá voda a poznalo se to například až při mytí rukou.). To, že je potřeba odvzdušnit ohřívač se uživateli zobrazí ve stavovém rámečku „Odvzdušnit ohřívač TUV!“. Jak je vidět ve varovném hlášení je vykřičník na konci a jak bylo popsáno na začátku této kapitoly, pokud bude ve stavovém rámečku vykřičník, tak bude pozadí blikat střídavě bílá/červená. Když bude venkovní teplota nižší než 10°C, tak se ve stavovém rámečku vypíše „Venku začíná být zima, bude třeba zatopit.“. Jako poslední se kontroluje teplota kotle. Když bude teplota mezi 90 a 100°C vypíše se „Zvýšená teplota kotle.“, pokud bude mezi 100 (včetně) - 105°C vypíše „Kotel je přetopen!“, okno začne blikat červeně. Pokud bude větší než 105°C (včetně) tak vypíše „Kotel je nebezpečně přetopen - hrozí výbuch!“.

Toto jsou všechna hlášení. Když se po vyhodnocení všech těchto podmínek ve stavovém rámečku nic neobjeví tak do něj program vypíše „Vše OK“. V opačném případě pokud se v rámečku bude nacházet více jak 3 řádky, tak zapne vertikální rolovací posuvník, aby si obsluha mohla přečíst všechna hlášení.

Tento blok příkazů z posledních dvou kapitol se vykonává každou vteřinu a je tak zajištěna neustálá kontrola databáze.

4.3. Graf krátkodobá historie

Zdrojový kód okna krátkodobé historie je umístěn v souboru Form2.pas. A je vyvolán hlavním programem Form1.pas s parametrem tag, ve kterém je jedno číslo. Toto číslo předává nově otevřenému oknu informaci o čísle čidla. Když se tedy otevře okno grafu krátkodobé historie, tak je pomocí tagu zjištěno o jaké čidlo se jedná. Číslo tohoto čidla se objeví v horní liště okna a dále se využívá pro komunikaci s databází. Následuje zavolání procedury s názvem „NactiHistorii()“.

Procedura NactiHistorii vymaže všechny řádky v textové rámečku, kde se následně vypíšou naměřené hodnoty a přidá řádku s hlavičkou, ve které je napsáno datum, čas a °C. Graf se vytváří pomocí komponenty Chart1, před jejím použití je obsah předchozího grafu vymazán. Následně proběhne dotaz do databáze pro načtení hlavičky grafu (název grafu a popis čidla). Dotaz vypadá následovně: „SELECT * FROM cidla WHERE ciddruh='TEPLOTA' AND cidcislo='+inttostr(Form2.Tag)“ vybere z tabulky čidla pouze jeden řádek, ve kterém druh čidla je teplota a číslo čidla je hodnota z tagu, který byl předán z hlavního okna. Do nadpisu grafu se tímto přidá jeho jméno – název čidla.

Nyní je potřeba zjistit posledních 100 naměřených hodnot. Provedeme tedy druhý dotaz tentokrát do tabulky historie: „SELECT * FROM historie WHERE histdruh='TEPLOTA' AND histcislo='+inttostr(Form2.Tag)+' ORDER BY histdatum DESC LIMIT 100 OFFSET '+inttostr(ofset)“. Dotaz tedy vybere z tabulky historie, kde je druh čidla TEPLOTA a zároveň číslo čidla odpovídá proměnné „tag“, data seřadí sestupně podle data měření a limit je nastaven na 100 záznamů z tabulky dále a jejich posun je dán hodnotou v proměnné ofset, který je nastaven na nulu (pokud budeme chtít vidět dalších 100 hodnot, tak se pomocí tlačítka další změní proměnná ofset tak, že k jejímu obsahu je přičítána hodnota 100 a tak se posouvají vypsané záznamy hlouběji do historiie).

V cyklu while jsou nyní zpracovávány všechny přijaté řádky (celkem 100) a roztříděny do proměnných podle názvu sloupce. Každá hodnota je přidána do rámečku se všemi naměřenými hodnotami. Dále je teplota přidána do grafu – na osu x čas měření a na osu y hodnota ve °C. Datum prvního a posledního prvku načteného z databáze je vložen do dvou textových polí, aby měl uživatel přehled od, kdy do kdy jsou vykreslené hodnoty grafu. V okně grafu se nachází dvě tlačítka, která umožňují načítat další a další hodnoty v historii.

4.4. Graf denní historie

Grafické vyjádření naměřených hodnot prostřednictvím denního grafu pracuje obdobně jako předchozí graf, ale s tím rozdílem, že nyní bude potřeba devět čar pro pokrytí všech čidel. K načítání slouží procedura NactiHistorii(). Zdrojový kód Form4.pas

Po zobrazení okna programu je vyvolána procedura NactiHistorii(). Nejprve se smažou všechny staré křivky z předchozího vykreslení grafu. Zároveň se smaže název grafu. Do proměnné casod je vloží dnešní datum a čas 00:00:00, do proměnné casdo je vloží stejný den, ale čas bude 23:59:59. Tím máme zajištěno, že se vykreslí celý den. Rozsahu data lze poté měnit. Dotazování do databáze proběhne devětkrát pomocí cyklu for s proměnnou počet, která postupně nabývá hodnot od 1 do 9. Databázový dotaz má tvar: „SELECT * FROM historie WHERE histdruh="TEPLOTA' AND histcislo='+inttostr(pocet)+' AND histdatum>‘+casod+‘ AND histdatum<'+casdo+‘“‘. Následuje větvení příkazem case podle proměnné počet. Pokud bude např. pocet = 3, tak se do třetí křivky vloží všchna data z databáze (osa x = čas, osa y = teplota).

Po načtení všech devíti křivek je nutno ještě připojit legendu. Ta se získá SQL dotazem do tabulky cidla, ze které se získá popis pro jednotlivá čidla. Nakonec se přidá titulek grafu nesoucí datum, který je v grafu zobrazován.

Pro načtení jiného dne z databáze se pouze změní datum v levém horním rohu okna a graf se automaticky překreslí.

4.5. Nastavení programu

Toto okno (Form.3.pas) slouží k nastavení mezních parametrů pro změnu barvy pozadí tlačítek zobrazovaných teplot. Nejprve se provede databázový dotaz „SELECT * FROM cidla ORDER BY cidcislo“ – vybere všechny řádky a seřadí je vzestupně. Pokud druh čidla bude teplota, tak se do struktur typu pole uloží minimální, střední a maximální hodnota pro další

zpracování. Do rozbalovacího seznamu se zároveň vloží čísla jednotlivých čidel a jejich popis.

Při zvolení jednoho z 9 čidel rozbalovacího seznamu se do třech textových políček vloží tři mezní hodnoty z příslušné struktury. Zde je možné tyto hodnoty libovolně měnit. Vkládání je ošetřeno tak, že lze zadávat pouze čísla od 0 do 9 a znaménko „-“. Po upravení mezních hodnot je potřeba kliknout na tlačítko uložit, nebo zavřít okno a program se dotáže, jestli chce uživatel změny uložit.

Při kliknutí na tlačítko uložit se vyšle do databáze dotaz „UPDATE cidla SET cidminimum= '+AdvEdit1.Text+', cidstred= '+AdvEdit2.Text+', cidmaximum= '+ AdvEdit3.Text+' WHERE cidcislo= '+inttostr(cidlo)+' and ciddruh= 'TEPLOTA'“. Dotaz postupně rozebereme. Příkaz update provede změnu určitého prvku v databáze. Do tabulky čidla, kde cidcislo = číslu čidla a zároveň ciddruh = teplota nastaví sloupce cidminimum, cidstred a cidmaximum na hodnotu z příslušných textových polí AdvEdit1 až 3.

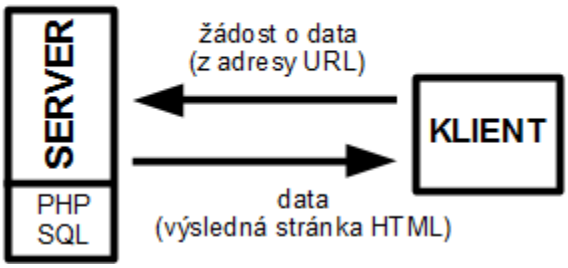
Pokud budeme chtít zavřít okno a data nebudou uložena v databázi, tak se program dotáže, jestli má data uložit, pokud uživatel odpoví OK, pak se vyvolá stejná událost, jako při kliknutí na tlačítko ulož a okno se zavře. Pokud stiskne NO tak se pouze zavře okno a uložení neproběhne.

5. Klientská aplikace – Web

5.1. Programovací jazyk PHP

PHP je serverový skriptovací jazyk. Je určený pro programování dynamických internetových stránek a webových aplikací například ve formátu HTML. PHP pro dynamické stránky využívá principu klient/server. To znamená, že skripty jsou prováděny na straně serveru a k uživateli je přenášen pouze výsledek jeho činnosti. Syntaxe jazyka vychází z několika programovacích jazyků (Perl, C, Pascal a Java). [6]

PHP je nezávislý na platformě operačního systému, rozdíly v různých operačních systémech se omezují na několik systémově závislých funkcí. Skripty lze většinou mezi operačními systémy přenášet bez jakýchkoli úprav.



Obr. 5.1: Princip klient/server, získávání webové stránky ze serveru.

PHP podporuje mnoho knihoven - jedná se například o zpracování grafiky, grafů, práci se soubory, přístup k většině databázových systémů (MySQL, ODBC, Oracle, PostgreSQL) a podporuje celou řadu internetových protokolů (HTTP, FTP, POP3 ,..).

PHP je nejrozšířenějším skriptovacím jazykem pro web. Je velice oblíbený především díky jednoduchosti, velkému množství funkcí, databázovým systémem (MySQL, PostgreSQL). Spolu s webovým serverem Apache je často využíván k tvorbě webových aplikací. [6]

5.2. Princip běhu webové klientské aplikace

Pro on-line sledování teplot topné soustavy jsem s výhodou využil možnosti spojení programovacího jazyka PHP a databáze PostgreSQL. Domácí server pro měření teplot je neustále v provozu a sleduje v zadaném časovém intervalu teploty, digitální vstupy a analogový vstup. Na serveru je nainstalován operační systém Windows XP Professional. Dále

na něm běží databáze PostgreSQL, program LogTemp, serverová aplikace měření teplot a IIS (Internetová Informační Služba) operačního systému s nainstalovaným PHP verze 5.2.3. PHP server má v sobě implementovány knihovny pro komunikaci s databází PostgreSQL. Webová aplikace pro sledování vytápění (v PHP) je uložena na serveru ve složce C:\Inetpub\wwwroot\kotel2\index.php.

Celý server je připojen prostřednictvím domácí sítě LAN k routeru, který je zároveň připojen k internetu. Připojení k internetu probíhá prostřednictvím širokopásmového připojení ADSL přes pevnou linku. ADSL připojení je s pevnou, veřejnou IP adresou. Router má tedy nastaveno přesměrování veřejné IP adresy (109.80.149.9) na lokální IP adresu (10.0.0.7) pouze na portu 80. Port 80 slouží k přenosu www stránek (HTTP). Pokud tedy vzdálený uživatel zadá kdekoli na světě v internetovém prohlížeči adresu: http://109.80.149.9/kotel2/index.php tak může sledovat poslední naměřené hodnoty. Zadávání IP adresy má značnou nevýhodu v tom, že je potřeba si jí pamatovat. Pro zjednodušení mám zaregistrovanou u poskytovatele webhostingů pipni.cz svoji internetovou doménu www.jandolak.cz. Tato stránka obsahuje jednoduchý odkaz na výše uvedenou IP adresu a uživatel může pohodlně přejít na systém sledování teplot.

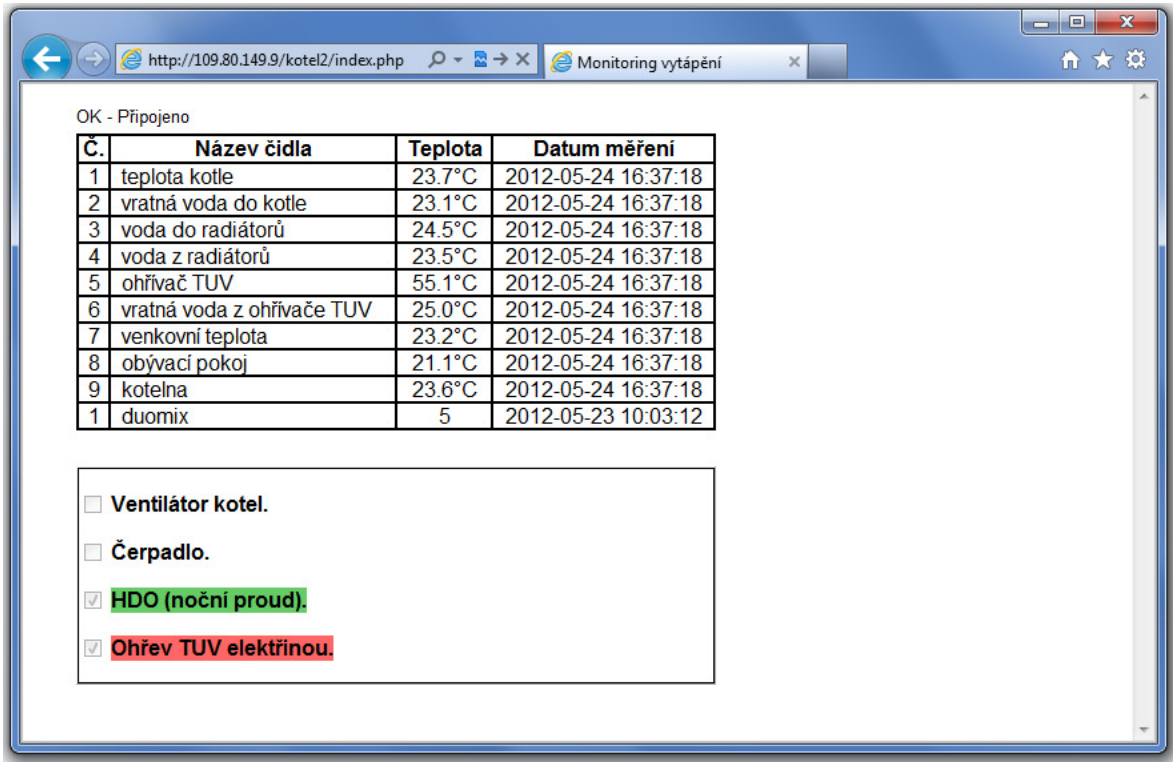
5.3. Popis webové aplikace

Webová aplikace se nachází na adrese www.jandolak.cz. Přičemž princip běhu těchto internetových stránek je popsán v předchozím odstavci.

Webová aplikace se skládá z celkem tří souborů. Jsou to index.php a pak ve složce options – nastaveni.php a teploty.css. První soubor index.php obsahuje hlavní tělo programu. Nastaveni.php nese informaci o přihlašovacích údajích k databázi PostgreSQL (jméno uživatele: postgres, heslo, IP adresu, na které běží databáze – lokální 127.0.0.1 včetně čísla portu: 5432 a název databáze: kotel). A konečně teploty.css je soubor kaskádových stylů používaných při tvorbě HTML stránek. Pomocí kaskádových stylů lze upravovat konečnou grafickou podobu stránek. Grafickou podobou se rozumí definice barev, zarovnání textu, umístění a orámování tabulek, obrázků, volba pozadí atd. V mém případě je v tomto souboru definováno tělo html stránky, styl tabulky a styl checkboxů symbolizující stav digitálních vstupů (pokud například probíhá ohřívání TUV elektřinou, tak je pozadí textu v okolí checkboxu červené, v opačném případě má barvu pozadí).

Nyní bude následovat popis hlavního souboru index.php. Zdrojový kód příloha D.3 obsahuje klasickou html hlavičku která sebou nese informaci o nastavení kódování jazyka

UTF8 a dále připojení souboru s kaskádovými styly options/index.css. Následuje tělo samotné stránky s přirazením css identifikátoru „obsah“. Následuje první PHP script, který připojí soubor options/nastaveni.php a provede pomocí příkazu if pokus o připojení k databázi. Pokud se spojení nepodaří, tak na obrazovku vypíše hlášku „CHYBA – nepřipojeno!“ a činnost programu bude ukončena. V případě připojení do databáze vypíše hlášku „OK - připojeno“ a program pokračuje dál. Následuje vytvoření hlavičky tabulky o 4 sloupcích. První sloupec obsahuje číslo čidla, druhý název čidla, třetí hodnotu a čtvrtý datum a čas měření. Dále proběhne konverze jazyka s databází (UTF8). Do proměnné $db se uloží textový řetězec obsahující dotaz do databáze: „SELECT * FROM cidla WHERE ciddruh='TEPLOTA' ORDER BY cidcislo“. Provede se dotaz s textem v proměnné $db. Databáze vrátí výsledek ve formě řádek, který je uložen v proměnné $vysledek. Jednotlivé řádky jsou uloženy do proměnné typu pole hodnot, kde každému prvku odpovídá jeden sloupec tabulky. Následuje cyklus WHILE, který bude zpracovávat jednotlivé řádky výsledku a probíhá tak dlouho, dokud nejsou všechny řádky zpracovány. Pokud tedy narazí na první řádek, tak nejprve upraví formát měřené hodnoty. Ta má na výstupu z databáze dvě desetinná místa, což příliš nepřidává na přehlednosti, proto bude oříznuta pouze na jedno desetinné místo. Poté se formátovaným výstupem printf přidá jeden řádek tabulky a příslušným obsahem dat. Tento cyklus while proběhne celkem 9 pro každé číslo jedna řádka. Obdobným způsobem se načte analogová hodnota polohy otevření ventilu duomix a vloží se do následující řádky, na poslední místo tabulky. Nyní následuje ukončení tabulky.



Obr. 5.2: Klientská aplikace v internetovém prohlížeči.

Pro digitální vstupy je vytvořena další tabulka, která má ovšem pouze jeden řádek a jeden sloupec pro přehlednost. Je vytvořena proměnná s názvem $pole_vstupy a následuje SQL dotaz „"SELECT * FROM cidla WHERE ciddruh='VSTUP' ORDER BY cidcislo“. Ve while cyklu se stejným způsobem zpracují všechny řádky. Nyní je potřeba podle čísla vstupu rozlišit jednotlivé stavy. Stav jednotlivých vstupů se indikuje pomocí zaškrtávacích políček tzv. „checkboxů“, ty mají v html stav buď zaškrtnuto = „CHECKED“ anebo nezaškrtnuto = „“. Pro stav „log. 0“ se do $pole_vstupy vloží hodnota CHECKED a pro „log. 1“ prázdná hodnota „“. Po proběhnutí cyklu while a naplnění $pole_vstupy se vypíší jednotlivé checkboxy. Jejich vlastnosti se liší názvem, stavem (zaškrtnuto / nezašrtnuto), dále barvou textu, která se odvíjí od jejich stavu a nadefinování v css stylech. Všechny checkboxy jsou deaktivované, takže při pokusu uživatele kliknout myší se nic nestane. Toto je stručný popis klient aplikace přes webový prohlížeč.

6. Závěr

Hlavním cílem práce bylo vytvořit systém sledování vytápění v rodinném domku. Podařilo se vytvořit serverovou aplikaci a aplikaci klienta pro platformu Windows, která je funkční a otestována v provozu. Jako zajímavé se jevilo naprogramování webové aplikace v jazyce PHP. Myslím si, že další vývoj by měl jít tímto směrem, protože vytvořená aplikace klienta je dostupná pouze na počítači v místní lokální síti a pouze s operačním systémem Windows. Webová aplikace se dá spustit jak na počítačích s operačním systémem Windows tak i na Linuxu. V dnešní době jsou stále více rozšířené tzv. chytré telefony s operačními systémy Android, iOS a Windows Phone, na kterých webovou aplikaci není problém spustit kdekoli na světě.

Protože můj systém využívá sériovou linku RS-232, která je v dnešní době spíše na ústupu a v nových počítačích bývá méně často zakomponována, tak by bylo zajímavé do budoucna navržené převodníky upravit a použít dnes velmi rozšířenou linku USB. Na trhu jsou běžně k dostání obvody pro USB 2.0 od výrobce FTDI. Například integrovaný obvod typu FT4232H v sobě ukrývá celkem 4 kanály RS-232, přičemž jsou připojeny na jediný USB port. To by plně pokrylo potřebu tří převodníků popsaných v této bakalářské práci.

Použitá literatura

[1] Procházka, M., Strakoš, M. Borland Delphi průvodce vývojáře kniha I. Praha: Mobil Media a.s., 2002. ISBN 80-86593-31-2.

[2] Matoušek, D. Udělejte si z PC v Delphi – 1.díl. Praha: BEN – Technická literatura, 2003. ISBN 80-7300-111-X.

[3] Holan, T. Delphi v příkladech. Praha: BEN – Technická literatura, 2001. ISBN 807300-033-4.

[4] Svoboda, L., Voneš, P., Konšal, T., Mareš, M. 1001 tipů a triků pro Delphi. Praha: Computer Press, a. s., 2003. ISBN 80-7226-488-5.

[5] Váňa, V. Mikrokontroléry ATMEL AVR – programování v jazyce Bascom. Praha: BEN – Technická literatura, 2004. ISBN 80-7300-115-2.

[6] Ponkrác, M. PHP a MySQL bez předchozích znalostí. Praha: Computer Press, a. s., 2007. ISBN 978-80-251-1758-3.

[7] Šimůnek, M. SQL kompletní kapesní průvodce. Praha: Grada Publishing, s. r. o., 1999. ISBN 80-7169-692-7.

[8] Juránek, A., Hrabovský, M. EAGLE pro začátečníky – uživatelská a referenční příručka. Praha: BEN – Technická literatura, 2005. ISBN 80-7300-177-2.

[9] Navrátil, J. Domácí kutil a … dřevoplyn. Olomouc: Tisk a vazba Moravské tiskárny a. s., 1998. ISBN 80-902244-2-3

[10] PostgreSQL: [online]. [cit. 24.1.2012]
URL: http://postgres.cz

[11] Regulus.cz: [online]. [cit. 20.2.2012]
URL: http://regulus.dnh.cz/hsksolar

[12] Atmos.cz, Zplynovací kotle na uhlí a dřevo: [online]. [cit. 20.2.2012]
URL: http://atmos.cz/czech/kotle-002-zplynovaci-kotle-na-uhli-drevo

[13] Gefran LT-M-0100-S Datasheet: [cit. 24.2.2012]
URL: http://www.gefran.com/en/product_categories/38/products/68#downloads

[14] MR Soft – LogTemp: [cit. 2.3.2012]
URL: http://www.mrsoft.fi/index.htm

[15] Driver 1-Wire: [cit. 2.3.2012]
URL: http://www.maxim-ic.com/products/ibutton/software/tmex/

[16] Teploměr pro PC: [cit. 2.3.2012]
URL: http://trifid2.sweb.cz/teplomer/

[18] DS18B20 Datasheet: [cit. 15.3.2012]
URL: http://datasheets.maxim-ic.com/en/ds/DS18B20.pdf/

[19] Kainka, B., Berndt, H. *Využití rozhraní PC pod Windows*. Brno: Nakladatelství HEL, 2000. ISBN 80-86167-13-5.

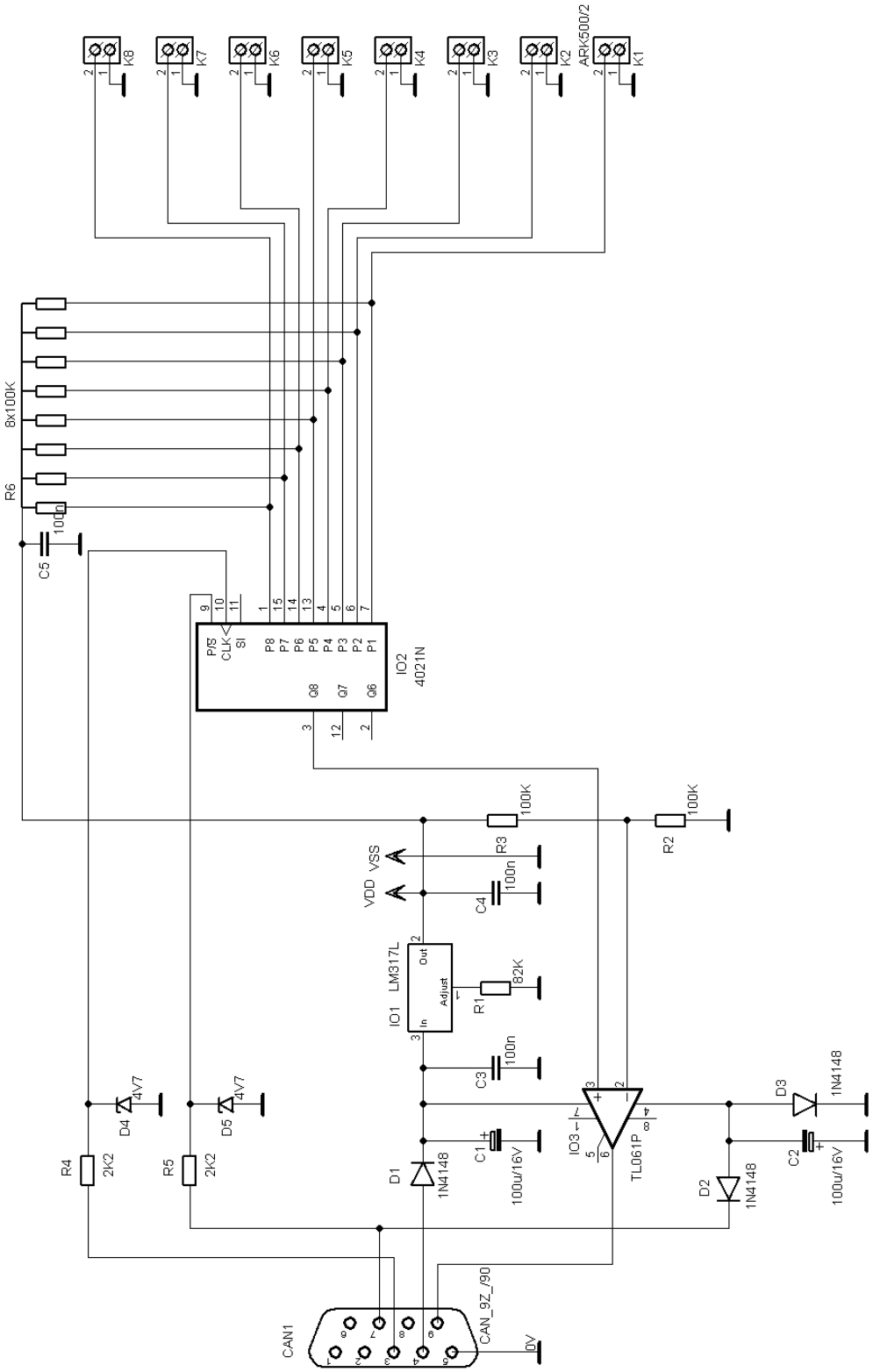
[20] Jedlička, P. *Přehled obvodů řady CMOS 4000 díl I. 4000…4099.* Praha: BEN – Technická literatura, 1994. ISBN 80-901984-.

[21] Popis sběrnice 1-Wire: [cit. 16.5.2012]
URL: http://www.maxim-ic.com/products/1-wire/

[22] ATmega8 Datasheet: [cit. 31.5.2012]
URL: http://www.atmel.com/images/doc2486.pdf

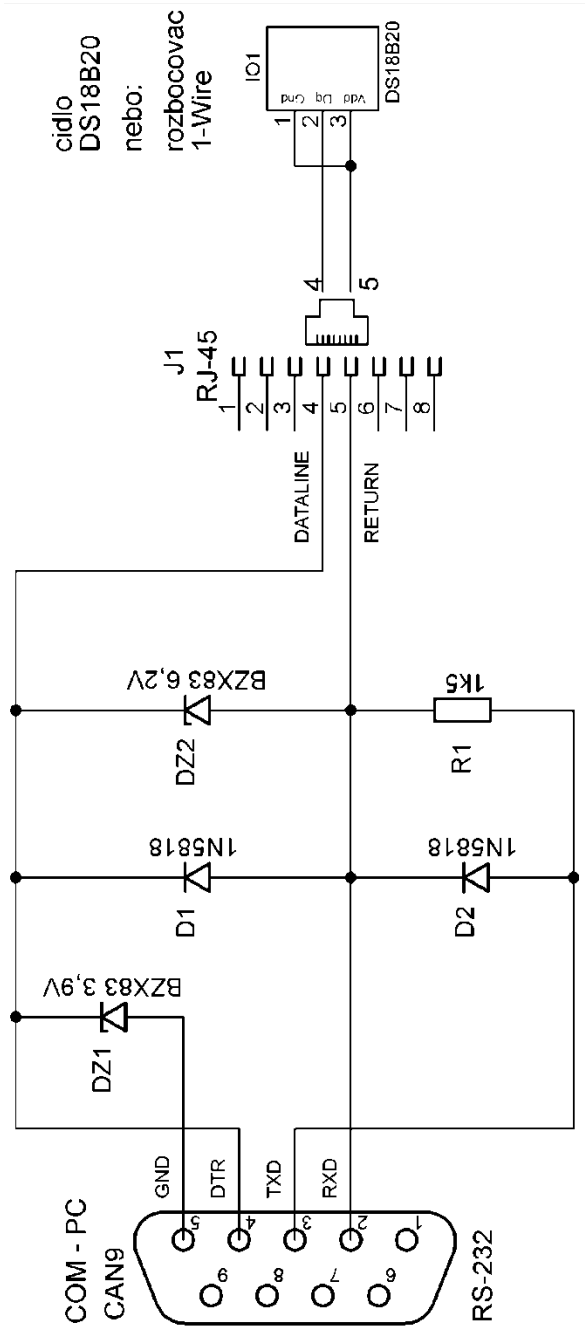
Příloha A: Schémata zapojení

A.1 Karta s 8 digitálními vstupy



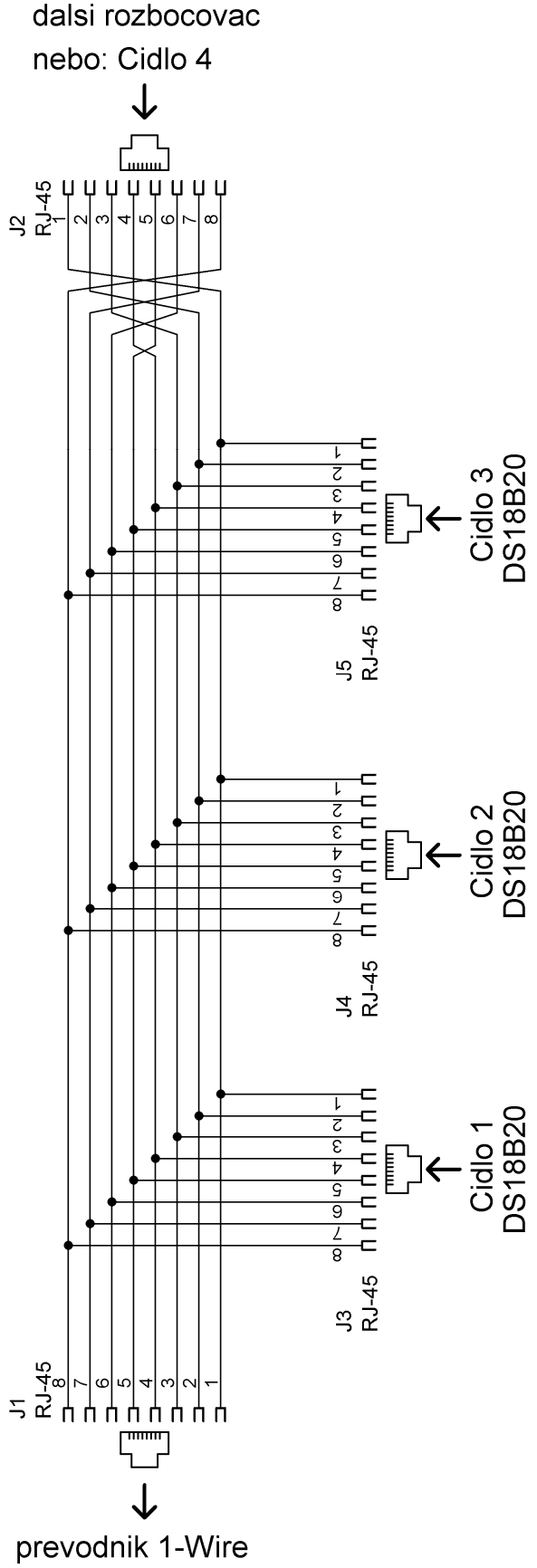
Obr. A.1: Schéma zapojení digitální katry s osmi vstupy.

A.2 Převodník sítě 1-Wire/RS-232



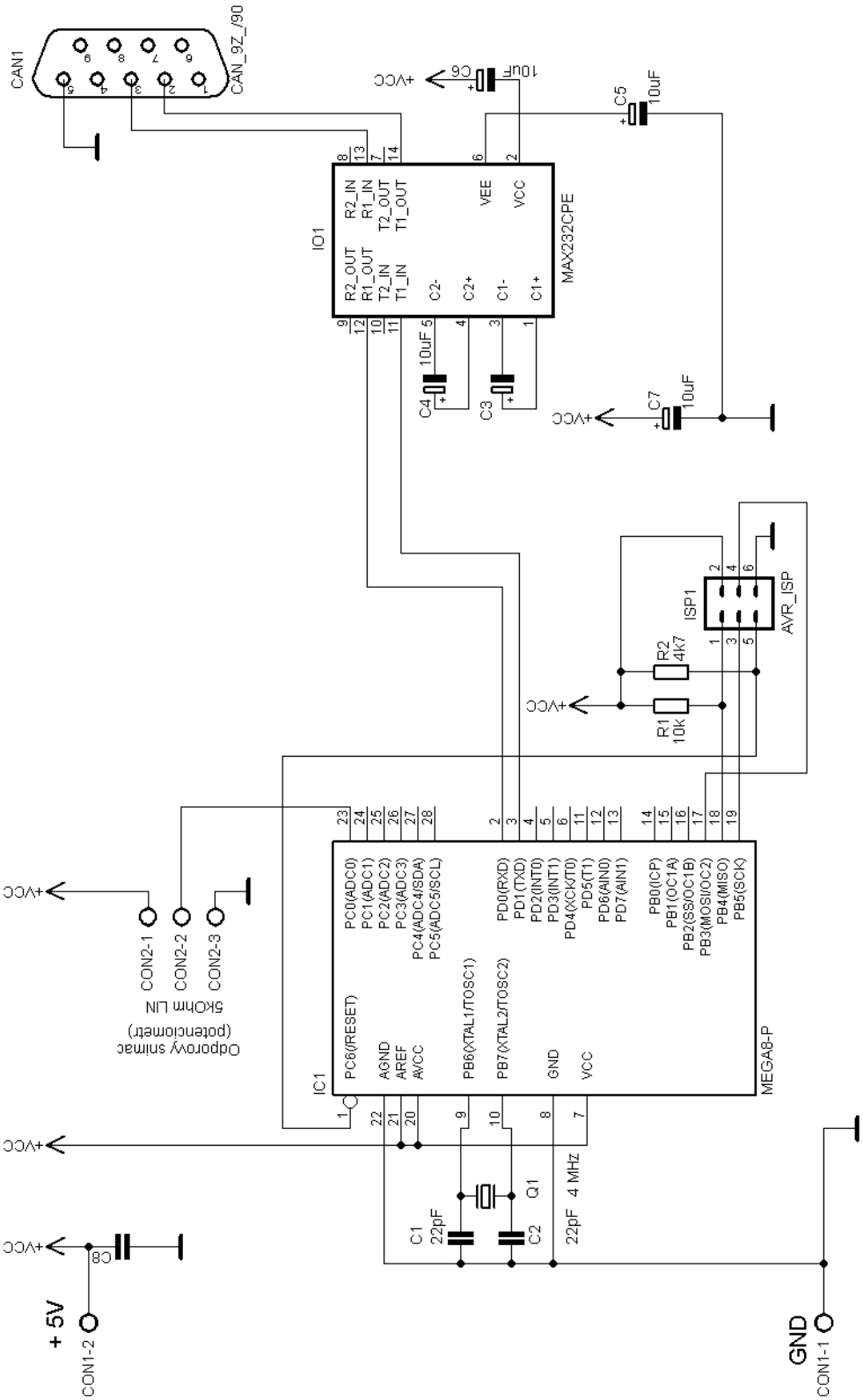
Obr. A.2: Schéma zapojení převodníku pro čidla 1-Wire.

A.3 Rozbočovač sítě 1-Wire



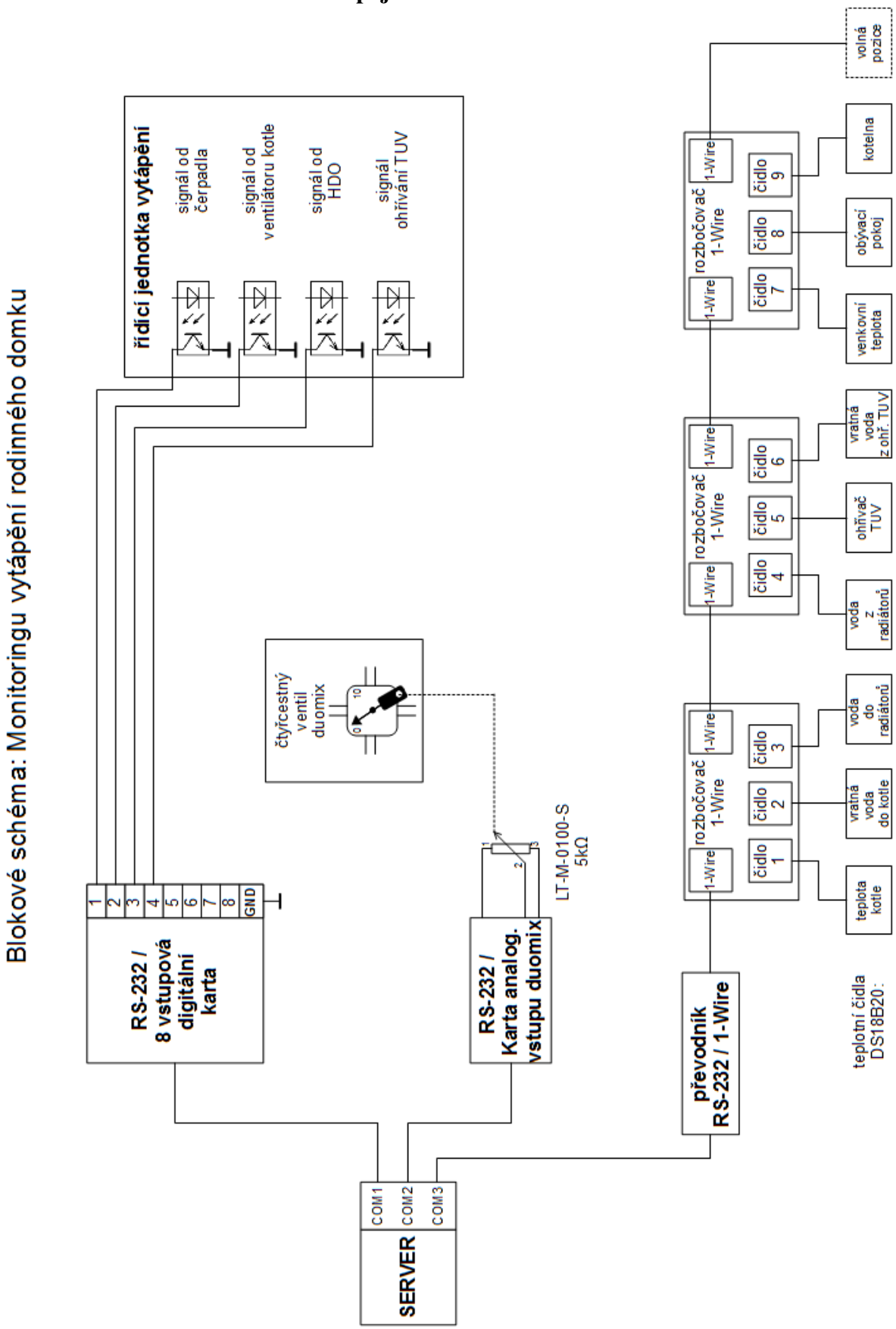
Obr. A.3: Schéma zapojení rozbočovače pro více čidel 1-Wire.

A.4 Karta s analogovým vstupem - Duomix



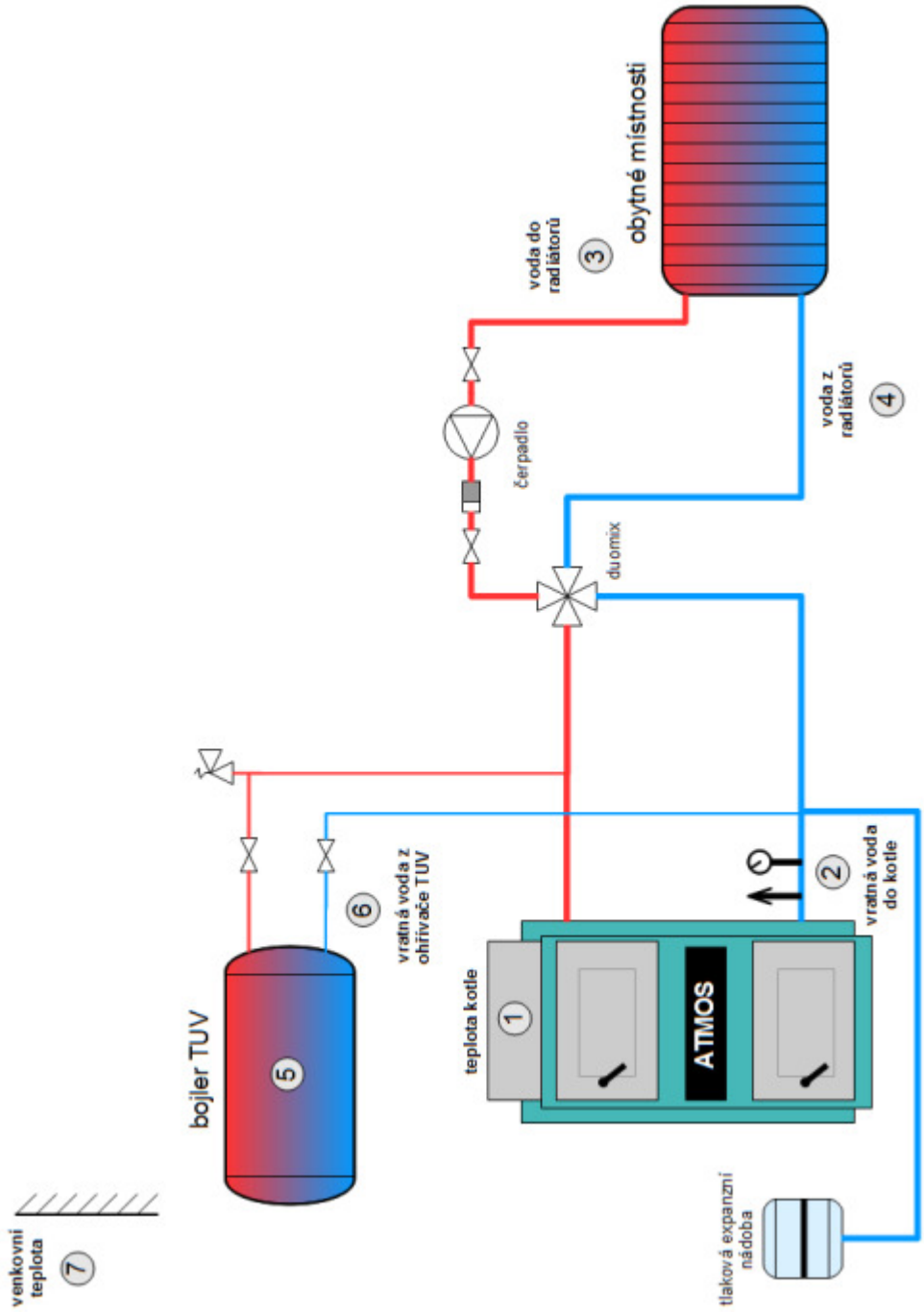
Obr. A.4: Schéma zapojení analogové karty duomixu.

A.5 Blokové schéma celkového zapojení



Obr. A.5: Blokové schéma zapojení jednotlivých komponent monitoringu vytápění

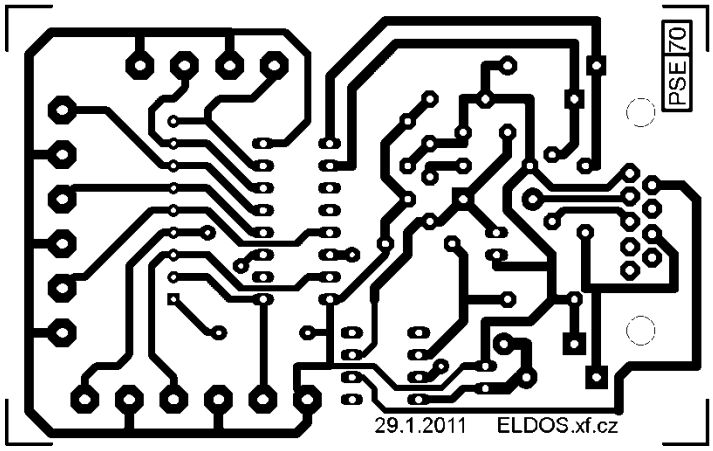
A.6 Topná soustava



Obr. A.6: Schéma zapojení topné soustavy s rozmístěním jednotlivých čidel.

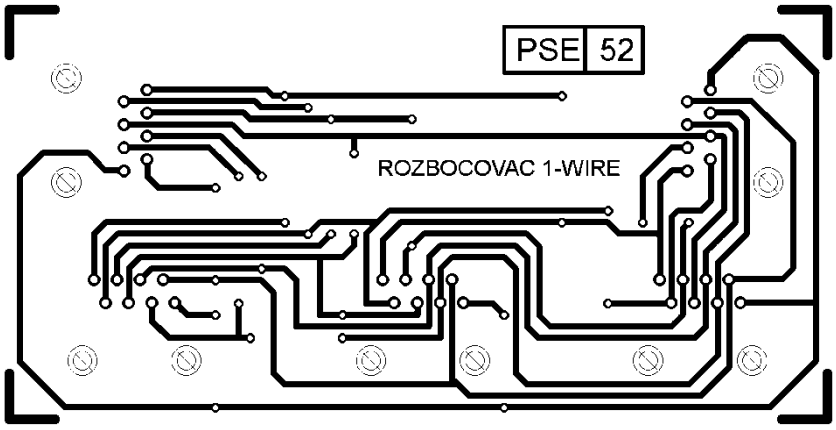
Příloha B: Desky plošných spojů

B.1 Karta s 8 digitálními vstupy



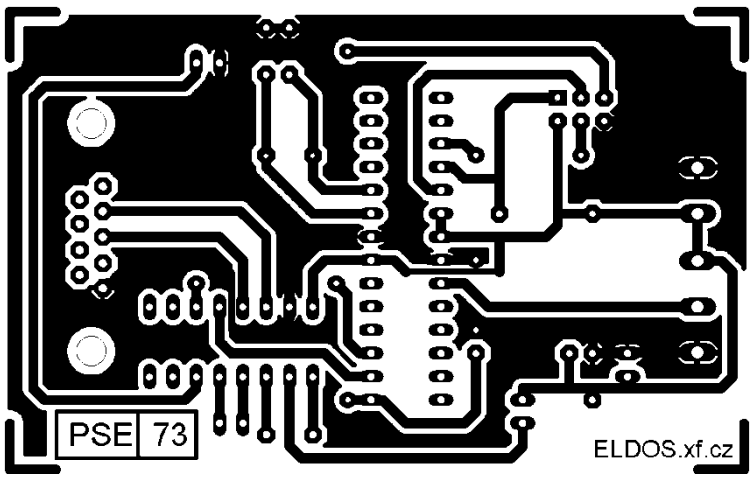
Obr. B.1: Jednovrstvá deska plošného spoje karty s osmi vstupy, pohled ze strany spojů (rozměry 80 x 50mm)

B.2 Rozbočovač sítě 1-Wire



Obr. B.2: Jednovrstvá deska plošného spoje raozbočovače 1-Wire, pohled ze strany spojů (rozměry 90 x 45mm)

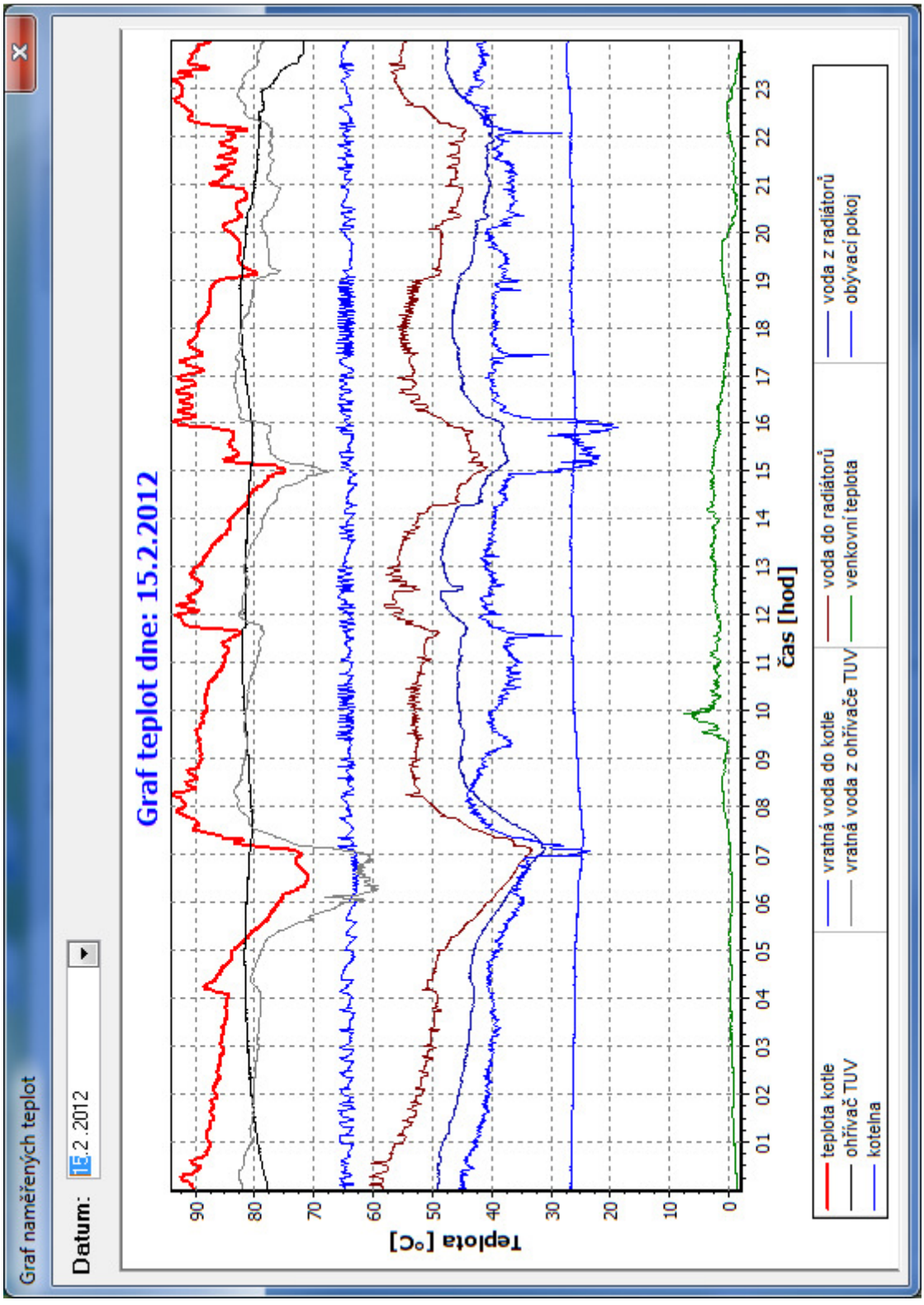
B.3 Karta s analogovým vstupem - Duomix



Obr. B.3: Jednovrstvá deska plošného spoje karty s analogovým vstupem Duomix, pohled ze strany spojů (rozměry 80 x 50mm)

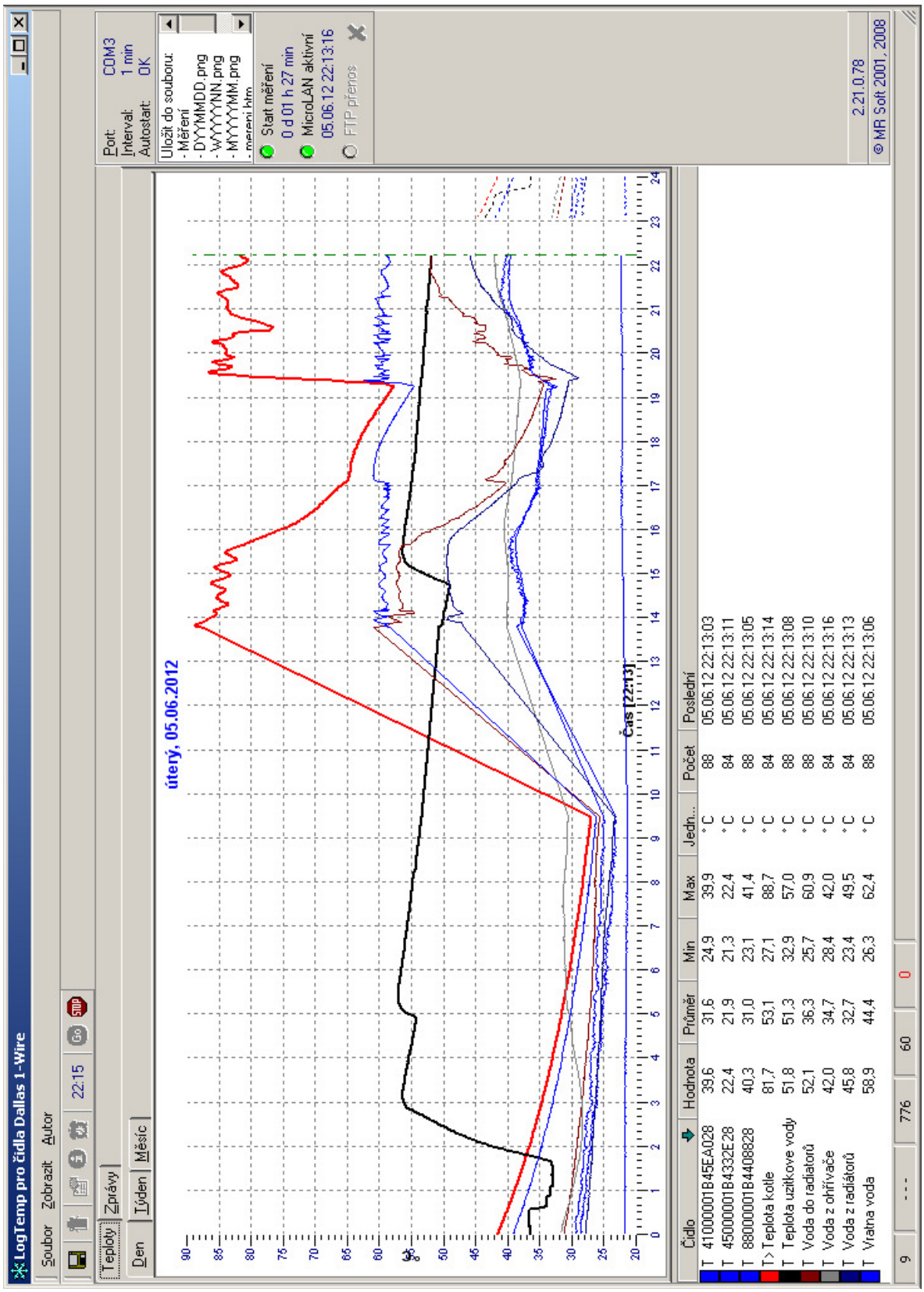
Příloha C: Naměřené hodnoty – graf

C.1 Denní graf



Obr. C.1: Graf měřených teplot dne 15.2.2012

C.2 Ukázka programu LogTemp



Obr. C.2: Program LogTemp

Příloha D: Použité skripty, zdrojové kódy

D.1 Serverová aplikace (ComForm.pas)

```
var
  HlForm: THlForm;
  gvstupy:string;
  ganalogy_zapis:string;
  analogy:string;

    nastaveni:record teplomer:array[1..9] of string;
              end;
    analog_duomix:record mez:array[0..11] of string;
              end;
implementation
{$R *.dfm}
// ***************************************************************************
// *********** Převod data a času ve formátu pro PostgreSQL (2011-11-11 20:00:00)
// ***************************************************************************
 function DateTimeToPostgreSQL(const dateTime : TDateTime) : string;
 var
   rok,mesic,den:word;
   hour,min,sec,msec:word;
 begin
   DecodeDate(dateTime, rok, mesic, den) ;
   DecodeTime(dateTime, hour, min, sec, msec) ;

   result := Format('%.*d-%.*d-%.*d %.*d:%.*d:%.*d',[4, rok, 2, mesic, 2, den, 2, hour, 2,
min, 2, sec]) ;
 end;
// ***************************************************************************
// *********** Procedura pro zápis do tabulky cidla(posledních hodnot) a historie do
PostgreSQL
// ***************************************************************************
 Procedure THlForm.ZapisPostgreSQL(druh:string; cislo:integer; hodnota:string; datum:String) ;
 begin
  PSQLQuery1.SQL.Text:='INSERT INTO historie (histdruh, histcislo, histhodnota, histdatum)
VALUES ('+#39+druh+#39+','+inttostr(cislo)+','+hodnota+', '+#39+datum+#39+')';
  Memo1.Lines.Add('INSERT INTO historie (histdruh, histcislo, histhodnota, histdatum) VALUES
('+#39+druh+#39+','+inttostr(cislo)+','+hodnota+', '+#39+datum+#39+')');
  PSQLQuery1.ExecSQL;
  PSQLQuery1.SQL.Text:='UPDATE cidla SET cidhodnota='+hodnota+', ciddatum='+#39+datum+#39+'
WHERE cidcislo='+inttostr(cislo)+' and ciddruh='+#39+druh+#39+'';
//  Memo1.Lines.Add('UPDATE cidla SET cidhodnota='+hodnota+', ciddatum='+#39+datum+#39+' WHERE
cidcislo='+inttostr(cislo)+' and ciddruh='+#39+druh+#39+'');
  PSQLQuery1.ExecSQL;
 end;
// ***************************************************************************
// *********** FORM CRATE
// ***************************************************************************
procedure THlForm.FormCreate(Sender: TObject);
var n:integer;
    wtep:integer;
    wmez:integer;
begin
  gvstupy:='';
  Memo1.Text:='';
  try
    n:=strtoint(LMDIniCtrl1.ReadString('PORT','Port_INPUT','1'));
    Caption:=Caption+Format(' vstupy: (COM%d)',[n]);
    Device:=TCOM4021.Create(n);
  except
    Application.MessageBox(
      'Port není k dispozici',
      PChar(Caption),
      MB_ICONHAND);
    Application.Terminate;
  end;
  Timer.Enabled:=Assigned(Device);
  //otevreni Com pro analog kartu
  ganalogy_zapis:='';
  analogy:='';
  if StrToBool(LMDIniCtrl1.ReadString('nastaveni','analog','false'))=true
    then
```

```
      begin
        AfComPort1.ComNumber:=StrToInt(LMDIniCtrl1.ReadString('PORT','Port_ANALOG','2'));
        AfComPort1.Open;
        Caption:=Caption+Format(' duomix: (COM%d)',[AfComPort1.ComNumber]);
      end
    else AfComPort1.Close;
  //Ma se program minimalizovat k hodinam?
  if StrToBool(LMDIniCtrl1.ReadString('nastaveni','minimalizace','false'))=true then
Application.Minimize else HlForm.Show;
  // Načtení ID teploměrů z .ini file
  for wtep:=1 to 9 do
    begin
      nastaveni.teplomer[wtep]:=LMDIniCtrl1.ReadString('Cidlo_id','cidlo_'+inttostr(wtep),'');
    end;
  abfFolderMonitor1Change;
  // Načtení Meze duomixu z .ini file
  for wmez:=0 to 11 do
    begin

analog_duomix.mez[wmez]:=LMDIniCtrl1.ReadString('Analog_duomix','mez_'+inttostr(wmez),'');
    end;
  //nacteni last.csv a last_vstupy.csv s poslednim stavem
  abfFolderMonitor1Change;
  end;
// ******************************************************************************
// *********** KONTROLA VSTUPů
// ******************************************************************************
procedure THlForm.FormDestroy(Sender: TObject);
begin
  Device.Free;
end;

procedure THlForm.Scan(Sender: TObject);
var Data:Byte;
    vstupy:string;
    vstup_id:integer;
    vstup_stav,vstup_datum:string;
begin

  Data:=Device.Data;

  cb0.Checked:=DecodeBits(Data,0);
  cb1.Checked:=DecodeBits(Data,1);
  cb2.Checked:=DecodeBits(Data,2);
  cb3.Checked:=DecodeBits(Data,3);
  cb4.Checked:=DecodeBits(Data,4);
  cb5.Checked:=DecodeBits(Data,5);
  cb6.Checked:=DecodeBits(Data,6);
  cb7.Checked:=DecodeBits(Data,7);

  vstupy:=IntToStr(ord(cb0.Checked))
  +IntToStr(ord(cb1.Checked))
  +IntToStr(ord(cb2.Checked))
  +IntToStr(ord(cb3.Checked))
  +IntToStr(ord(cb4.Checked))
  +IntToStr(ord(cb5.Checked))
  +IntToStr(ord(cb6.Checked))
  +IntToStr(ord(cb7.Checked));
// ******************************************************************************
// *********** ZAZNAMENáNí ZMěNY VSTUPů DO PostgreSQL
// ******************************************************************************
  if gvstupy<>vstupy then
  begin
   vstup_datum:=DateTimeToPostgreSQL(now);
   LMDButton1.Caption:=vstup_datum;
   for vstup_id:=1 to 8 do
    begin
     if Copy(gvstupy,vstup_id,1) <> Copy(vstupy,vstup_id,1) then
      begin
       vstup_stav:=vstupy[vstup_id];
       ZapisPostgreSQL('VSTUP',vstup_id,vstup_stav,vstup_datum);
      end;
    end;
  gvstupy:=vstupy;
  end;
end;
// ******************************************************************************
```

```
// *********** ZAZNAMENáNÍ ZMěNY TEPLOTY DO PostgreSQL
// ******************************************************************************
procedure THlForm.abfFolderMonitor1Change;
var f1:textfile;
    ws:string;
    col,row:integer;
    hodnota,vstup_datum:string;
    wtep,wtep2:integer;

begin
 wtep:=0;wtep2:=0;
 vstup_datum:=DateTimeToPostgreSQL(FileDateToDateTime(fileage('.\data\last.csv')));
 LMDButton1.Caption:=vstup_datum;
 sleep(100);//bezpecnostni pockani, ze je soubor last.csv donakopirovan, aby nedoslo k potkani
se
 if FileExists('.\data\last.csv') then
  begin
   AssignFile(f1,'.\data\last.csv');
   reset(f1);
   row:=0;
   while not eof(f1) do
    begin         //col je sloupec
     readln(f1,ws);
     row:=row+1;
     if row>1 then
      begin
       col:=0;
       while pos(',',ws)>0 do
        begin
         hodnota:=copy(ws,1,pos(',',ws)-1);
         delete(ws,1,pos(',',ws));
         col:=col+1;
         if col=1 then
          begin
           for wtep:=1 to 9 do if ('"'+nastaveni.teplomer[wtep]+'"'=hodnota) then begin
wtep2:=wtep;break; end;
          end;
         if col=2 then
          begin
           case wtep2 of
            1:begin
              ZapisPostgreSQL('TEPLOTA',wtep,hodnota,vstup_datum);
              end;
            2:begin
              ZapisPostgreSQL('TEPLOTA',wtep,hodnota,vstup_datum);
              end;
            3:begin
              ZapisPostgreSQL('TEPLOTA',wtep,hodnota,vstup_datum);
              end;
            4:begin
              ZapisPostgreSQL('TEPLOTA',wtep,hodnota,vstup_datum);
              end;
            5:begin
              ZapisPostgreSQL('TEPLOTA',wtep,hodnota,vstup_datum);
              end;
            6:begin
              ZapisPostgreSQL('TEPLOTA',wtep,hodnota,vstup_datum);
              end;
            7:begin
              ZapisPostgreSQL('TEPLOTA',wtep,hodnota,vstup_datum);
              end;
            8:begin
              ZapisPostgreSQL('TEPLOTA',wtep,hodnota,vstup_datum);
              end;
            9:begin
              ZapisPostgreSQL('TEPLOTA',wtep,hodnota,vstup_datum);
              end;
           end;
          end;
        end;
      end;
    end;
   closefile(f1);
  end;
end;
// *********** Načtení Analogové hodnoty Duomixu
// ******************************************************************************
```

```
procedure THlForm.AfComPort1DataRecived(Sender: TObject; Count: Integer);
var ws:string;
    a:integer;
    analogy_datum:string;
    analogy_zapis:string;
begin
 ws := AfComPort1.ReadChar;              //čte postupně znaky z COMu jak jsou posílány do PC
 if pos(#13,ws)=0                        //pokud neni prijimany znak CR carriage return(13)
tak ho ignoruje
  then
    begin
           if pos(#10,ws)=0              //pokud neobsahuje znak LF line feed(10) potom
            then
              begin
                analogy:=analogy+ws;     // neni line feed(10) tj. konec řádky tak postupně
poslepuj přijaté znaky do slova=stringu
              end
            else
              begin
                for a:=0  to 10 do
                  begin
                    if (analogy>=analog_duomix.mez[a]) and (analogy<analog_duomix.mez[a+1])
                      then
                        begin
                          analogy_zapis:=inttostr(a);
                        end;
                  end;
                    analogy:='';              //nuluj přijaté slovo
                if ganalogy_zapis<>analogy_zapis      // je line feed(10) = tj. konec řádky
tak zjišťuj zdali byla změna oproti předchozímu slovu, případně zapiš do Edit1 a vymaž analogy
                  then
                    begin
                     Edit1.Text:=analogy_zapis;
                     analogy_datum:=DateTimeToPostgreSQL(now);
                     LMDButton1.Caption:=analogy_datum;
                     if analogy_zapis<>'' then
ZapisPostgreSQL('ANALOG',1,analogy_zapis,analogy_datum); //aby to tam nehrnulo prázdné znaky
                     if edit1.Color=clWindow then edit1.Color:=clGray else
edit1.Color:=clWindow; //je změna=přehoď pozadí
                    end;
                ganalogy_zapis:=analogy_zapis;        // ulož do globalni proměnné pro
nádcházející slova
              end;
    end;
end;

procedure THlForm.LMDTrayIcon1DblClick(Sender: TObject);
begin
HlForm.Show;
end;

end.
```

D.2 Klientská aplikace pro Windows – hlavní okno (Unit1.pas)

```
var
  Form1: TForm1;
  nastaveni:record
              aktualizace:integer;
              teplomer:array[1..9] of string;
              alarmy:record
                     minimum:array[1..9] of integer;
                     stred:array[1..9] of integer;
                     maximum:array[1..9] of integer;
                     end;
            end;
  teplota:array[1..9] of real;

implementation
uses Unit2,Unit3,Unit4,Unit99_funkce;
{$R *.dfm}
//******************** Převod teploty 19.31 převede na  19,3
function tep2tep(teplota:string):Double;
begin
 teplota:=StringReplace(teplota,'.',',',[rfReplaceAll,rfIgnoreCase]);
 result:=StrToFloatDef(teplota,0);
end;

//******************** Rozdělení "alarmů" k rozmezí 3 hodnot přiřadí
čtyři barvy a vrátí číslo barvy
function teplota2barva(tep:real;m1,m2,m3:integer):TColor;
var wtc:TColor;
begin
 wtc:=clskyblue;
 //ShowMessage(floattostr(teplota));
 if (tep<m1) then  wtc:=clskyblue;
 if (tep>=m1) and (tep<m2) then  wtc:=$008CFFFC;
 if (tep>=m2) and (tep<m3) then  wtc:=clMoneyGreen;
 if (tep>=m3) then  wtc:=$008484FF;
 result:=wtc;
end;

procedure  TForm1.FormCloseQuery(Sender:  TObject;  var  CanClose:
Boolean);
begin
end;

procedure TForm1.Button1Click(Sender: TObject);
begin
 Form1.Close;
end;

procedure TForm1.FormCreate(Sender: TObject);
var wtep,meze:integer;
     druh,id:string;
     cislo,minimum,stred,maximum:integer;

begin
//******************** Heslo a IP pro přístup do BD

PSQLDatabase1.Host:=LMDIniCtrl1.ReadString('PostgreSQL','ip_adresa_db','');

PSQLDatabase1.UserPassword:=LMDIniCtrl1.ReadString('PostgreSQL','heslo','')
;
```

```
  HlavickaSQL1('SELECT * FROM cidla ORDER BY cidcislo');
  while not PSQLQuery1.Eof do
  begin
    druh:=PSQLQuery1.FieldByName('ciddruh').AsString;
    cislo:=PSQLQuery1.FieldByName('cidcislo').AsInteger;
    id:=PSQLQuery1.FieldByName('cidid').AsString;
    minimum:=PSQLQuery1.FieldByName('cidminimum').AsInteger;
    stred:=PSQLQuery1.FieldByName('cidstred').AsInteger;
    maximum:=PSQLQuery1.FieldByName('cidmaximum').AsInteger;

    // Načtení mezí teplot minimum, stred, maximum a ID teploměrů z DB
    if druh='TEPLOTA' then
      begin
        nastaveni.alarmy.minimum[cislo]:=minimum;
        nastaveni.alarmy.stred[cislo]:=stred;
        nastaveni.alarmy.maximum[cislo]:=maximum;
        nastaveni.teplomer[cislo]:=id;
        case cislo of
          1:btnTeplota1.Hint:='ID:'+id;
          2:btnTeplota2.Hint:='ID:'+id;
          3:btnTeplota3.Hint:='ID:'+id;
          4:btnTeplota4.Hint:='ID:'+id;
          5:btnTeplota5.Hint:='ID:'+id;
          6:btnTeplota6.Hint:='ID:'+id;
          7:btnTeplota7.Hint:='ID:'+id;
          8:btnTeplota8.Hint:='ID:'+id;
          9:btnTeplota9.Hint:='ID:'+id;
        end;

      end;
    PSQLQuery1.Next;
  end;
  PSQLQuery1.Close;

 AdvPicture1.Picture.LoadFromFile('.\img\flame_block_e0.gif');
 AdvPicture2.Picture.LoadFromFile('.\img\cerp_anim.gif');
 AdvPicture3.Picture.LoadFromFile('.\img\fan-anim.gif');
 AdvPicture4.Picture.LoadFromFile('.\img\sipka_anim1.gif');
end;

procedure TForm1.btnTeplota9Click(Sender: TObject);
begin
 Form2.Tag:=9;
 Form2.ShowModal;
end;
procedure TForm1.btnTeplota8Click(Sender: TObject);
begin
 Form2.Tag:=8;
 Form2.ShowModal;
end;
procedure TForm1.btnTeplota7Click(Sender: TObject);
begin
 Form2.Tag:=7;
 Form2.ShowModal;
end;
procedure TForm1.btnTeplota5Click(Sender: TObject);
begin
 Form2.Tag:=5;
 Form2.ShowModal;
end;
procedure TForm1.btnTeplota3Click(Sender: TObject);
```

```
     begin
      Form2.Tag:=3;
      Form2.ShowModal;
     end;
     procedure TForm1.btnTeplota6Click(Sender: TObject);
     begin
      Form2.Tag:=6;
      Form2.ShowModal;
     end;
     procedure TForm1.btnTeplota1Click(Sender: TObject);
     begin
      Form2.Tag:=1;
      Form2.ShowModal;
     end;
     procedure TForm1.btnTeplota4Click(Sender: TObject);
     begin
      Form2.Tag:=4;
      Form2.ShowModal;
     end;
     procedure TForm1.btnTeplota2Click(Sender: TObject);
     begin
      Form2.Tag:=2;
      Form2.ShowModal;
     end;

     //**********************************************************************
     //********* Obsluha od časovače, tato událost nastane každých 1000ms
     //**********************************************************************
     procedure TForm1.Timer1Timer(Sender: TObject);
     var popis,druh,hodnota,datum:string;
         i,cislo:integer;

     begin
      btnCas.Caption:='Čas:'+TimeToStr(now);
      if pos('!',memStav.Text)=0
      then memStav.Color:= clWhite
      else
       begin
        if   odd(secondof(now))   then   memstav.Color:=   clWhite   else
memstav.Color:= clred;
       end;
     //***************** nacteni teplot a vstupů z databáze a vyhodnocení
stavů v MEMO1
       HlavickaSQL1('SELECT * FROM cidla ORDER BY cidcislo');
       while not PSQLQuery1.Eof do
       begin
         druh:=PSQLQuery1.FieldByName('ciddruh').AsString;
         cislo:=PSQLQuery1.FieldByName('cidcislo').AsInteger;
         hodnota:=PSQLQuery1.FieldByName('cidhodnota').AsString;
         datum:=PSQLQuery1.FieldByName('ciddatum').AsString;
         btnAktualizace.Caption:=datum;

     //************************** Z databáze do btnTeplota
         if druh='TEPLOTA' then
           begin
             teplota[cislo]:=strtofloat(hodnota);
             for i:=0 to form1.ControlCount-1 do
             begin
              if      ((Form1.Components[i]      is      TLMDButton)      and
(copy((Form1.Components[i] as TLMDButton).Name,1,10)='btnTeplota')) then
               begin
```

```
                cislo:=strtointdef(copy((Form1.Components[i]                as
TLMDButton).Name,11,1),0);
                (Form1.Components[i]                                       as
TLMDButton).Caption:=FloatToStrF(teplota[cislo], ffFixed, 3, 1)+'°C';
                (Form1.Components[i]                                       as
TLMDButton).Color:=teplota2barva(teplota[cislo],nastaveni.alarmy.minimum[ci
slo],nastaveni.alarmy.stred[cislo],nastaveni.alarmy.maximum[cislo]);
              end;
            end;
          end;
    //************************** Z databáze do btnAnalog
        if druh='ANALOG' then
          begin
          if cislo=1 then
            btnAnalog1.Caption:=hodnota;
          end;
    //************************** Z databáze do LMD Check box
        if druh='VSTUP' then
          begin
          case cislo of
              1:begin
                  if hodnota='0' then
                   begin
                    AdvPicture2.Animate:=true;
                    LMDCheckBox2.Checked:=true;
                    AdvPicture2.Hint:='čerpadlo běží';
                    AdvPicture4.Animate:=true;              //zobrazí šipku
(AdvPicture4) že teče v okruhu voda
                    AdvPicture4.Visible:=true;
                   end
                  else
                   begin
                    AdvPicture2.Animate:=false;
                    LMDCheckBox2.Checked:=false;
                    AdvPicture2.Hint:='čerpadlo zastaveno';
                    AdvPicture4.Animate:=false;
                    AdvPicture4.Visible:=false;
                  end;
                end;
              2:begin
                   // Zapnutí animace když běží ventilátor -> 1. resp. 2.
bit vstupní karty
                   if hodnota='0' then
                       begin
                        AdvPicture3.Animate:=true;
                        LMDCheckBox1.Checked:=true;
                        AdvPicture3.Hint:='Ventilátor běží';
                       end
                    else
                       begin
                        AdvPicture3.Animate:=false;
                        LMDCheckBox1.Checked:=false;
                        AdvPicture3.Hint:='Ventilátor zastaveno';
                      end;
                end;
              3:begin
                   // Signalizace HDO -> 2. resp. 3. bit vstupní karty
                   if hodnota='0' then
                       begin
                        LMDCheckBox3.Checked:=true;
                        LMDCheckBox3.Color:=clMoneyGreen; //HDO = zelená
```

```
                    end
                  else
                    begin
                     LMDCheckBox3.Checked:=false;
                     LMDCheckBox3.Color:=clSkyBlue;
                    end;
                end;
              4:begin
                  // Signalizace Ohřívání TUV -> 3. resp. 4. bit vstupní
karty
                  if hodnota='0' then
                    begin
                     LMDCheckBox4.Checked:=true;
                     LMDCheckBox4.Color:=$008484FF; //ohřívá = červená
                    end
                  else
                    begin
                     LMDCheckBox4.Checked:=false;
                     LMDCheckBox4.Color:=clSkyBlue;
                    end;
                end;
              //5://pripraveno pro dalsi vstupy
              //6:
              //7:
              //8:
            else
            end;
          end;
        PSQLQuery1.Next;
      end;
      PSQLQuery1.Close;

    //*************************************zpracovani hodnot teplot,
vydnocovani stavu
      memStav.Lines.Clear;
      // Vyhodnocení nízké teploty vody v ohřívači TUV:
      if teplota[5]<50 then
      begin
       memStav.Lines.Add('V ohřívači TUV je nízká teplota vody.');
      end;

      // Vyhodnocení zavzdušněného ohřívače TUV, zatopení a ovládání
animace plamene na kotli
      if teplota[1] >= 65 then
       Begin
        if (teplota[5]-teplota[6]>=20)
         then memStav.Lines.Add('Odvzdušnit ohřívač TUV!');
        AdvPicture1.Animate:=true; AdvPicture1.Visible:=true;
       end
      else
       Begin
       if teplota[7] <10
        then  memStav.Lines.Add('Venku  začíná  být  zima,  bude  třeba
zatopit.');
       AdvPicture1.Animate:=false; AdvPicture1.Visible:=false;
       end;

      // Vyhodnocení nebezpečných teplot kotle:
      if (teplota[1]>=90) and (teplota[1]<100)
       then memStav.Lines.Add('Zvýšená teplota kotle.');
      if (teplota[1]>=100) and (teplota[1]<105)
```

```
         then memStav.Lines.Add('Kotel je přetopen !');
        if (teplota[1]>=105)
         then  memStav.Lines.Add('Kotel  je  nebezpečně  přetopen  -  hrozí
výbuch!');

        // Jestliže neni v memStav žádný text tak napiš OK, vyhodnocení
jestli je potřeba scrollovat:
        if memStav.text='' then memStav.Lines.Add('Vše OK');
        if  memStav.Lines.Count>=3  then  memStav.ScrollBars:=ssVertical  else
memStav.ScrollBars:=ssNone ;
     end;

     procedure TForm1.btnNastaveniClick(Sender: TObject);
     begin
      Form3.ShowModal; //aktivní bude pouze form 3
     end;

     procedure TForm1.Button2Click(Sender: TObject);
     begin
      Form4.ShowModal; //aktivní bude pouze form 4
     end;

     procedure TForm1.FormKeyPress(Sender: TObject; var Key: Char);
     begin
      if key=#27 then Form1.Close;
     end;

     end.
```

D.3 Klientská aplikace v PHP – hlavní program (index.php)

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8"><!-
- kódování jazyka -->
<link rel="stylesheet" type="text/css" href="options/teploty.css"><!--
odkaz na kaskádové styly -->
<title>Monitoring vytápění</title>
</head>
<body>
<div id="obsah">
<?
require("options/nastaveni.php");
$spojeni=Pg_Connect("host=$conn_server port=$conn_port dbname=$conn_db
user=$conn_user password=$conn_pass");
if (!$spojeni) {echo "CHYBA - nepřipojeno!";exit; }echo "OK -
Připojeno";
echo "<br>";
?>
<table div id="tabulka_teploty" border="1" cellspacing="0"
cellpadding="0">
  <tr>
    <th width=5%>
      Č.
    </th>
    <th width=45%>
      Název čidla
    </th>
    <th width=15%>
      Hodnota
    </th>
    <th width=35%>
      Datum měření
    </th>
  </tr>
<?
pg_query("SET NAMES 'utf8'"); // konverze cestiny s databazi
$db = "SELECT * FROM cidla WHERE ciddruh='TEPLOTA' ORDER BY cidcislo";
$vysledek = pg_query($db); /* Konec přímé práce s databází. */
while($zaznam = pg_fetch_assoc($vysledek)) // konec přímé práce s
databází.
  {
    $zaznam['cidhodnota'] = preg_replace('~[0123456789]$~', '',
$zaznam['cidhodnota']); //Ořez setin naměřené teploty tak aby ukozovalo jen
desetiny
    printf ("<tr> <td>%s</td><td align=left> 
%s</td><td>%s°C</td><td>%s</td> </tr>", $zaznam['cidcislo'],
$zaznam['cidpopis'], $zaznam['cidhodnota'], $zaznam['ciddatum']);
  }

pg_query("SET NAMES 'utf8'"); // konverze cestiny s databazi
$db = "SELECT * FROM cidla WHERE ciddruh='ANALOG' ORDER BY cidcislo";
$vysledek = pg_query($db); /* Konec přímé práce s databází. */
while($zaznam = pg_fetch_assoc($vysledek)) // konec přímé práce s
databází.
  {
    $zaznam['cidhodnota'] = round($zaznam['cidhodnota']); //z desetiny
ukaze jen cely cislo na kolik je otevren duomix
    printf ("<tr> <td>%s</td><td align=left> 
%s</td><td>%s</td><td>%s</td> </tr>", $zaznam['cidcislo'],
```

```
$zaznam['cidpopis'], $zaznam['cidhodnota'], $zaznam['ciddatum']);
      }
    ?>
    </table>
    <br>
    <table div id="tabulka_teploty" width=100% border="1" cellspacing="0"
cellpadding="0">
      <tr>
        <th>
          <?
          $pole_vstupy=array("","","","","","","","","");  //Indexy v poli od
0 do 7! protoze mam 8 vstupu
          pg_query("SET NAMES 'utf8'"); // konverze cestiny s databazi
          $db = "SELECT * FROM cidla WHERE ciddruh='VSTUP' ORDER BY
cidcislo";
          $vysledek = pg_query($db); // konec přímé práce s databází.

          while($zaznam = pg_fetch_assoc($vysledek))
            {
              for ($i=1; $i<=8; $i++)
              {
                if ((($zaznam['ciddruh']=='VSTUP') &&
($zaznam['cidcislo']==$i))
                {
                  if ($zaznam['cidhodnota']==1)
                  $pole_vstupy[$i-1] = "";
                  else
                  $pole_vstupy[$i-1] = "CHECKED";
                }
              }
            }
          ?>
          <p><input type="checkbox" name="KOTEL" value="KOTEL" disabled
<?echo $pole_vstupy[1]?> >
          <span class='<? if ($pole_vstupy[1]=='CHECKED')
echo('styl_checked_blue'); else echo('styl_unchecked');?>'>Ventilátor
kotel.</span> </p>
<p><input type="checkbox" name="CERPADLO" value="CERPADLO" disabled <?echo
$pole_vstupy[0]?> >
<span class='<? if ($pole_vstupy[0]=='CHECKED') echo('styl_checked_blue');
else echo('styl_unchecked');?>'>Čerpadlo.</span> </p>
<p><input type="checkbox" name="HDO" value="HDO" disabled <?echo
$pole_vstupy[2]?> >
<span class='<? if ($pole_vstupy[2]=='CHECKED') echo('styl_checked_green');
else echo('styl_unchecked');?>'>HDO (noční proud).</span> </p>
<p><input type="checkbox" name="TUV" value="TUV" disabled <?echo
$pole_vstupy[3]?> >
<span class='<? if ($pole_vstupy[3]=='CHECKED') echo('styl_checked_red');
else echo('styl_unchecked');?>'>Ohřev TUV elektřinou.</span> </p>
        </th>
      </tr>
    </table>

    </body>
    </html>
    </div>

    </body>
    </html>
```

D.4 Analogová karta duomix – program (AD_prevodnik.bas)

```
'============================
'AD převodník 10bitů s ATMEGA8
'13.01.2012
'============================
'T51Prog configuration Bits:
'zaskrtnuto Flash pamet a Device configurations (Lock & Fuses)
'Externi crystal 3-8MHz 1K CK
'============================
$regfile = "M8def.dat"
$crystal = 4000000
$baud = 9600

Dim A As Word
Config Adc = Single , Prescaler = Auto
Start Adc

Do
 A = Getadc(0)
 Print A
 Waitms 3000
Loop
End
```