

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

SYSTEM PRO SBĚR DAT ZE SÍTĚ PLC FIRMY MICROPEL

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE
AUTHOR

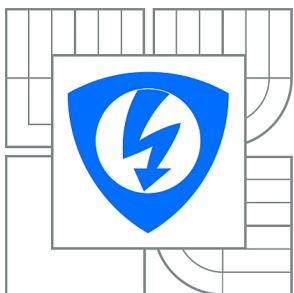
Bc. JAN KLUSÁČEK

BRNO 2012



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY



**FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH
TECHNOLOGIÍ**

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION
DEPARTMENT OF CONTROL AND INSTRUMENTATION

SYSTÉM PRO SBĚR DAT ZE SÍTĚ PLC FIRMY MICROPEL

DATE ACQUIRE SYSTEM FOR MICROPEL'S PLC NETWORK

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE

AUTHOR

Bc. JAN KLUSÁČEK

VEDOUcí PRÁCE

SUPERVISOR

Ing. TOMÁŠ MACHO, Ph.D.

BRNO 2012



VYSOKÉ UČENÍ
TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

Ústav automatizace a měřicí techniky

Diplomová práce

magisterský navazující studijní obor
Kybernetika, automatizace a měření

Student: Bc. Jan Klusáček

ID: 72972

Ročník: 2

Akademický rok: 2011/2012

NÁZEV TÉMATU:

Systém pro sběr dat ze sítě PLC firmy Micropel

POKYNY PRO VYPRACOVÁNÍ:

1. Seznamte se s PLC firmy Micropel a jejich propojením do sítě a s jednodeskovým počítačem FoxBoard G20 včetně programového vybavení.
2. Navrhnete koncepci sběru dat (údaje o teplotě a tlaku) z PLC prostřednictvím počítače FoxBoard G20. Data by měly být ukládána do databáze a přístupná uživatelům prostřednictvím webového rozhraní.
3. Doplníte počítač FoxBoard G20 o rozhraní pro sběrnici RS485 a napájecí zdroj.
4. Řešíte možnost vzdáleného přístupu k počítači FoxBoard 20 pomocí GSM modemu.
5. Naprogramujete a odladíte ovladač pro komunikaci s PLC firmy Micropel do OS Linux.
6. Implementujete databázový systém pro sběr naměřených dat a webové rozhraní pro vizualizaci dat. Systém odladíte.

DOPORUČENÁ LITERATURA:

- [1] Maslakowski, M. Naučte se MySQL za 21 dní. Praha: Computer Press, 2001. 478 s. ISBN 80-72226-448-6.
- [2] Brázda, J. PHP 5 začínáme programovat. Praha: Grada Publishing a.s., 2006. 244 s. ISBN 80-247-1146-X.

Termín zadání: 6.2.2012

Termín odevzdání: 21.5.2012

Vedoucí práce: Ing. Tomáš Macho, Ph.D.

Konzultanti diplomové práce:

doc. Ing. Václav Jirsík, CSc.

Předseda oborové rady

ABSTRAKT

Tato práce se zabývá návrhem a implementací obvodové a programové stránky zařízení sloužícího pro sběr dat. Zařízení je založeno na jednodeskovém mikropočítači FOX Board G20. Ten je umístěn na desku s krátkodobým zálohovaným zdrojem a rozhraním pro komunikaci pomocí linky RS485. Na zařízení běží operační systém GNU/Linux s ovladačem umožňujícím komunikovat s PLC firmy Micropel. Získaná data jsou ukládána do databáze a prezentována uživateli přes webové rozhraní.

KLÍČOVÁ SLOVA

ARM, Linux, Micropel, rs485

ABSTRACT

This paper deals with design and implementation of hardware and software for data acquisition device. Device is based on single board computer FoxBoard G20 which is placed on custom board which provide RS485 interface and short-term UPS. Device is running Linux with device driver which enabling communication with Micropel PLCs across RS485 bus. Data are read from PLCs are saved to database and afterwards presented to user via web interface.)

KEYWORDS

ARM, Linux, Micropel, rs485

KLUSÁČEK, Jan *Systém pro sběr dat ze sítě PLC firmy Micropel*: diplomová práce. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2012. 77 s. Vedoucí práce byl prof. Ing. Macho Tomáš, CSc.

PROHLÁŠENÍ

Prohlašuji, že svou diplomovou práci na téma „Systém pro sběr dat ze sítě PLC firmy Micropel“ jsem vypracoval samostatně pod vedením vedoucího diplomové práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené diplomové práce dále prohlašuji, že v souvislosti s vytvořením této diplomové 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 § 152 trestního zákona č. 140/1961 Sb.

Brno

.....

(podpis autora)

Děkuji vedoucímu diplomové práce Ing. Tomáši Machovi, Ph.D. za účinnou metodickou, pedagogickou a odbornou pomoc a další cenné rady při pracování diplomové práce.

V Brně dne: 20. května 2012

.....
podpis autora

Bibliografická citace:

KLUSÁČEK, J. *Systém pro sběr dat ze sítě PLC firmy Micropel*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2012. 78s. Vedoucí diplomové práce Ing. Tomáš Macho, Ph.D.

OBSAH

1	Úvod	11
2	Koncepce sběru dat	12
2.1	Popis monitorovaných technologických procesů	12
2.1.1	Parní sterilizace	12
2.1.2	Lyofilizační přístroj	14
2.2	Řešení dostupná na trhu	15
2.3	Návrh systému	16
3	Použité technologie	18
3.1	Programovatelné automaty Micropel	18
3.1.1	PES M66	18
3.1.2	PES K10 a K1	18
3.1.3	PES MPC300	19
3.1.4	Jazyk Simple	19
3.1.5	Komunikační protokol PesNet	19
3.2	FOX Board G20	20
3.2.1	AT91SAM9G20	21
3.3	GSM modem	21
3.3.1	MC75i	21
3.4	Senzory	22
3.4.1	Měření teploty	22
3.4.2	Měření tlaku	22
3.5	Operační systém	23
3.5.1	GNU/Linux	23
3.6	Lighttpd	24
3.7	PHP	24
3.8	XHTML	24
3.9	CSS	24
3.10	SQLite	25
4	Návrh a implementace	26
4.1	Podpůrné obvody	26
4.1.1	Digitální vstupy a výstupy	26
4.1.2	Budič sběrnice RS485	27
4.1.3	Zdroj	28
4.2	Příprava operačního systému	31

4.2.1	Kompilace jádra	31
4.2.2	Instalace operačního systému	34
4.2.3	Spouštění a vypínání systému	35
4.2.4	Init skripty	37
4.3	Komunikace přes RS485	40
4.3.1	Ovladač	40
4.4	Paměťové médium	42
4.4.1	Kapacita	44
4.5	Databazový model	44
4.6	Daemon logger	47
4.7	Webové rozhraní	50
4.7.1	Knihovna pro práci s daty	51
4.7.2	Knihovna pro práci s cache	52
4.7.3	Vlastní stránky	53
4.8	Vzdálený přístup	55
4.8.1	Připojení GSM modemu	55
4.8.2	Komunikace pomocí SMS	55
4.8.3	Spojení pomocí GPRS	56
4.8.4	Síťový tunel	57
5	Závěr	61
	Literatura	63
	Seznam příloh	65
A	Deska plošných spojů	66
A.1	Celá deska	66
A.2	Vrchní strana PCB	67
A.3	Spodní strana PCB	68
A.4	Osazovací výkres	69
B	Instalace debianu	70
B.1	Příprava před spuštěním	70
B.2	Skript stage2	71
C	Fotky zařízení v provozu	72
D	Ukázka vytvořeného protokolu	74

SEZNAM OBRÁZKŮ

2.1	Sběr dat v aplikaci parního sterilizátoru	17
4.1	Schéma digitálních vstupů	26
4.2	Schéma digitálních výstupů	27
4.3	Schéma budiče RS485	28
4.4	Schéma zdroje	30
4.5	Databázový systém modelu.	45
4.6	Vývojový diagram programu logger.	49
4.7	Uspořádání sítě.	58
C.1	Sterilizátor s vysunutým rozvaděčem	72
C.2	Vysunutý rozvaděč	72
C.3	Detail rozvaděče s FOX Boardem a podpůrnými obvody na levé straně	73
C.4	Detail desky s podpůrnými obvody	73

1 ÚVOD

Firma ing. Ivan Klusáček se zabývá především automatizací technologických procesů v potravinářství a farmacii. U všech těchto procesů je vyžadován záznam průběhu technologických procesů, pomocí kterého je možné zpětně dohledat celý průběh a zjistit zda odpovídá příslušným požadavkům. Pro pořízení tohoto záznamu je většinou používán mechanický zapisovač, který provádí zápis pomocí pohyblivého pera přímo na odvíjející se papír. U několika přístrojů byl použit program běžící na PC s OS Windows, zajišťující čtení a ukládání potřebných dat. Ten je po funkční stránce vyhovující, problémem je samotné použití PC. Zařízení z nichž je pořizován záznam jsou často umístěna v čistých prostorách, do kterých je umístění PC značně nevhodné. Také z hlediska spolehlivosti by bylo vhodnější použití jednoduššího zařízení. Cílem této práce je navrhnout zařízení schopné zajistit záznam dat ze sítě PLC, které by bylo dostatečně malé, aby ho bylo možné umístit přímo do rozvaděče monitorovaného zařízení a zároveň umožňovalo vzdálený přístup k naměřeným datům, ideálně přes www rozhraní a podnikovou síť ethernet.

2 KONCEPCE SBĚRU DAT

Úkolem navrhovaného systému je zaznamenávat časový průběh veličin z technologického procesu. Veličiny jsou měřeny čidly připojenými na analogové vstupy průmyslových automatů firmy Micropel. Odkud mohou být přečteny přes průmyslovou sběrnici RS485. Naměřená data by měla být dlouhodobě uchována (dle aplikace i několik let). Zaznamenaná data by měla být pohodlně přístupná uživateli tak, aby z nich šlo jednoduše vytvářet protokoly. Kde je to možné měly by být vytvářeny protokoly automaticky. Z časových úseků odpovídajících jednotlivým šaržím produktu (bude popsáno podrobně u konkrétních aplikací). Každý protokol musí obsahovat jednak graf, pomocí kterého je možné získat rychlý přehled o průběhu procesu a také tabulku hodnot, ze které lze získat přesné hodnoty. V některých případech je vhodné, aby protokol obsahoval závěr shrnující průběh procesu a zhodnocení zda proběhl dle požadavků (ty jsou popsány u jednotlivých aplikací). Vytvořené protokoly by mělo být možné uložit pro pozdější kontrolu a vytisknout pro archivaci či přiložení jako průvodní dokumentace výrobku.

U navrhovaného systému se předpokládá nejčastější aplikace pro záznam veličin technologických procesů ve farmacii. Zařízení, ze kterých je pořizován záznam bývají často umístěna v čistých prostorách z čehož vyplývají značná omezení. Zařízení umístěná v těchto prostorách musí mít omyvatelné povrchy a nesmějí obsahovat nepřístupné štěrby. Umístění klasického osobního počítače s tiskárnou do těchto prostor tedy není možné. Vlastní počítač by bylo možné nahradit vestavným průmyslovým počítačem s dotykovým panelem. Nahrazení tiskárny by bylo velmi problematické. Z těchto důvodů se zdá jako nejvhodnější přesunout část systému pro interakci s uživatelem dál od monitorovaného zařízení, mimo čisté prostory.

2.1 Popis monitorovaných technologických procesů

Celý systém má být koncipován tak, aby byl co nejuniversálnější a bylo možné ho nasadit při záznamu dat z různých technologických zařízení. Již v době návrhu zařízení jsou známy některé budoucí aplikace, které určují požadavky na celý systém. Typickými aplikacemi, kde se předpokládá nasazení navrhovaného systému sběru dat je např. parní sterilizátor a lyofilizační přístroj.

2.1.1 Parní sterilizace

Ve zdravotnictví, farmacii i průmyslu je často potřeba zajistit, aby použitý materiál a nástroje neobsahovaly žádné životaschopné mikroorganismy (bakterie, houby, viry atp.). V závislosti na odolnosti příslušného materiálu je možné k jeho sterilizaci

použít několik principů. Nejjednodušší možností je jejich zničení použitím zvýšené teploty. Jako medium pro ohřev se používá přefiltrovaný vzduch nebo sytá vodní pára. Pokud je potřeba sterilizovat materiály, které není možné vystavovat vysoké teplotě je možné použít sterilizaci využívající chemických látek nebo ionizujícího záření.

Navrhovaný systém bude sloužit k registraci údajů ze sterilizátoru využívajícího jako medium sytou vodní páru. Sytá vodní pára se využívá pro svoji velkou ohřívací schopnosti (teplo předává svou kondenzací). Další výhodnou vlastností je u syté vodní páry přesně daný vztah mezi teplotou a tlakem, takže se dá regulací tlaku v prostoru zajistit rovnoměrné rozložení teploty. Celý sterilizační proces se skládá z několika fází. Na začátku se v komoře sterilizátoru opakovaně vytváří podtlak a napustí se párou. Tím se odstraní většina vzduchu (směs vzduchu a páry nemá dobré sterilizační vlastnosti). Po té se přejde k vlastní sterilizaci, která spočívá v udržování předepsané teploty po odpovídající čas. Na závěr se provádí sušení materiálu, které spočívá v udržování podtlaku v komoře při současném ohřívání jejich stěn.

Požadavky na parní sterilizátory, včetně vlastností systému záznamu dat jsou určeny normou ČSN EN 285. Záznam teploty musí být prováděn s přesností ± 1 °C v rozsahu (50 °C až 150 °C) s vzorkovací periodou maximálně 2.5 s a záznam tlaku s přesností ± 5 kPa a vzorkovací periodou maximálně 1 s.

Teplota je měřena v odpadním hrdle komory, kde se předpokládá nejnižší teplota pracovního prostoru v důsledku stékání kondenzátu a shromažďování zbytkového vzduchu (je těžší než vodní pára). V případě sterilizace tekutin je nutné měřit teplotu také v referenční lahvičce za pomoci pohyblivého materiálového čidla. Tlak je měřen tlakovým snímačem připojeným ke komoře. Norma vyžaduje, aby byl celý systém použitý pro kontrolu a registraci nezávislý na systému regulačním. Z toho důvodu jsou použity dvousystémové snímače teploty, které v jednom pouzdru obsahují dvě nezávislá čidla. Jedno z dvojice čidel je vždy vedeno do automatu sloužícího k regulaci procesu a druhé do automatu použitého pro záznam. Pro měření tlaku jsou použity dva stejné nezávislé snímače.

V maximální konfiguraci je potřeba zaznamenávat tři údaje o teplotě (jedna teplota v odpadu a maximálně 2 materiálová čidla) a jeden údaj o tlaku. Aby bylo možné jednodušeji odhalit případné vadné čidlo, bylo by vhodné zaznamenávat také druhou sadu čidel používanou k regulaci. Kvůli vyhodnocení průběhu sterilizace je potřeba zaznamenávat údaj o aktuální fázi sterilizace. Celkem je potřeba zaznamenávat údaje s vzorkovací periodou 1s (teplotu by stačilo vzorkovat méně často, vzhledem k zobrazení v tabulce je však vhodné, aby byly všechny hodnoty vzorkovány současně). Za předpokladu, že bude zařízení v provozu v průměru 2 hodin denně je tedy nutné každý den uložit 57 600 vzorků.

Vzhledem k charakteru procesu, kde jeden průběh procesu odpovídá vždy jedné šarži výrobku by bylo vhodné, aby byly protokoly vytvářeny automaticky. Ideálně přímo na základě údaje z automatu udávajícího aktuální fázi sterilizace. Vytvořený protokol by měl kromě grafu a tabulky hodnot obsahovat také závěr, ve kterém bude vyhodnoceno zda celý proces proběhl úspěšně. Pro hodnocení účinnosti sterilizačního procesu se používá hodnota ekvivalentní doby sterilizace. Tato hodnota udává účinnost celého sterilizačního procesu ve srovnání se sterilizací za konstantní referenční teploty. Pro sterilizaci sytou vodní párou se bere jako referenční teplota 121°C. Hodnotu ekvivalentní doby sterilizace je možné spočítat podle následujícího vzorce:

$$F0 = \sum \Delta t \cdot 10^{\frac{T-121}{10}} \quad (2.1)$$

Kde:

- Δt [s] je délka časového úseku
- T [°C] je teplota

2.1.2 Lyofilizační přístroj

Lyofilizace je proces sušení sublimací. Je využíván k sušení a chemické stabilizaci materiálů, které není možné sušit při zvýšené teplotě. Při tomto procesu se materiál zmrazí na velmi nízkou teplotu (<-60 °C), následně je vložen na police do podtlakové nádoby, ve které je vývěvou snížen tlak na řádově jednotky Pa. Police s produktem jsou ohřívány. Vodní pára vznikající sublimací vody musí být odstranována namražováním na kondenzátor ochlazovaný na velmi nízkou teplotu (<=70 °C).

Lyofilizační proces se používá pro prodloužení životnosti léčiv, potravin a také při vysoušení vodou poškozených knih a dokumentů. Jeho hlavní nevýhodou je velká časová a energetická náročnost.

Lyofilizační přístroj, pro který má být navrhovaný systém použit, se skládá z tlakové nádoby s 9 vytápěnými policemi. Na vnitřní straně nádoby je navinuta trubka, kterou prochází chladicí kapalina a slouží pro kondenzaci vzniklé vodní páry. V každé polici jsou dvě čidla teploty, umístěná ve šroubech z boku zašroubovaných do každé police. Pro měření teploty vlastního materiálu umístěného na policích jsou k dispozici až čtyři přemístitelná čidla teploty. Další čidlo teploty je umístěno na trubce kondenzátoru. Tlak je měřen čidlem připojeným na trubce propojující komoru s vývěvou.

Zaznamenávat je tedy potřeba údaje z 14 čidel teploty (9 na policích, 4 materiálové a 1 čidlo na kondenzátoru) a 1 čidlo tlaku. Pro případ poruchy je vhodné zaznamenávat i čidla použitá pro regulaci. Vzhledem k velké setrvačnosti desek a pomalým změnám tlaku je dostačující vzorkovací perioda 5min. Vlastní lyofilizační

proces trvá několik dnů a celá výroba je poměrně dost vytížená. Je potřeba počítat s tím, že záznam běží prakticky nepřetržitě. Při celkovém počtu 30 zaznamenávaných veličin a vzorkovací periodě 5min je nutné každý den uložit 8640 vzorků.

2.2 Řešení dostupná na trhu

U samostatných zařízení¹ se v současné době se používají 4 základní možnosti záznamu průběhu technologických veličin: mechanické zapisovače, elektronické registrační přístroje, záznam za pomoci PC a záznam za použití systému sloužícího zároveň pro regulaci.

Mechanické zapisovače byly v minulosti jedinou možností automatizovaného záznamu veličin. Většina z těchto zařízení obsahuje cívku s dlouhým pásem papíru, který je pomocí krokového motoru (dříve hodinového strojku) převíjen na druhou cívku. Na procházející papír je pomocí pisátek s barevnými inkousty zaznamenáván průběh měřených veličin. Výstupem je záznam v podobě grafu, ze kterého je možné poměrně přesně odečítat hodnoty. Z konstrukce těchto zařízení však vychází také mnoho problémů. Pokud se pro zápis používají klasická pisátka při zaznamenávání více veličin na jeden papír musejí být vůči sobě posunutá, aby do sebe nenarážela. V důsledku tohoto posunutí jsou vůči sobě jednotlivé křivky posunuty v čase a při přesném odečtu hodnoty v čase je tedy potřeba brát ohled také na posuv každého pisátka. Posuv mezi jednotlivými záznamy není jediným problémem. Při větším počtu pisátek může docházet, zvláště pokud všechny zapisují stejnou hodnotu, k rozpíjení inkoustu. Některé zapisovače fungují na principu běžné tiskárny a není tedy nutné, aby byly vůči sobě jednotlivé zápisy posunuty, přesto mají některé společné nevýhody. Výstup v podobě grafu je sice výhodný z hlediska rychlého přehledu. Pokud je však nutné pracovat s číselnými údaji je potřeba hodnoty pracně odečítat.

Elektronické registrační přístroje dnes prakticky nahradily zapisovače mechanické. Někdy bývají rozměrově kompatibilní se svými mechanickými předchůdci, takže mohou být použity jako přímá náhrada. Jsou uloženy na mikro počítači, na kterém běží software zajišťující zápis dat do vnitřní paměti. Pro ukládání dat bývá přítomna interní paměť typu flash. Pokud je potřeba pořizovat větší množství záznamů bývá k dispozici slot umožňující použití paměťové karty. Některé elektronické registrační přístroje umožňují kontrolovat zda jsou měřené veličiny v definovaném intervalu a v případě vybočení signalizovat chybu. Jakékoliv další zpracování naměřených dat, případně možnost automatické tvorby protokolů, většinou není možné.

¹V rozsáhlejších systémech se většinou používá centralizovaný sběr dat na SCADA systému. V tomto případě je centrálně zajišťován sběr dat z většího množství zařízení. Touto možností se tato práce nezabývá.

Pokud je další zpracování vyžadováno je nutné data přenést na paměťové kartě (nebo po síti, pokud to přístroj umožňuje) do PC a zpracování provést tam. Z hlediska záznamu dat z popisovaných technologií má toto řešení několik nevýhod. Vzhledem k tomu, že automaty použité v této práci mezi sebou komunikují nestandardním protokolem, není možné použít k nim připojená čidla a pro záznam dat by tak musela být využita další sada. Dalším problémem by byla automatická tvorba protokolů.

Další možností je záznam dat pomocí běžného PC. V tomto případě je PC doplněno o hardware umožňující přístup k měřeným veličinám jako je karta s A/D převodníkem nebo s rozhraním pro komunikaci se sítí PLC. Pro vlastní záznam dat je k dispozici poměrně velké množství programů, často vytvořených výrobcí příslušného HW. Jedním příkladem je program DataStore[2] firmy Micropel. Hlavními výhodami tohoto řešení je poměrně nízká cena, velký výkon a velká kapacita paměti (pevného disku). Použití PC má však několik nevýhod. Patří mezi ně nižší spolehlivost, vyšší spotřeba, velké rozměry.

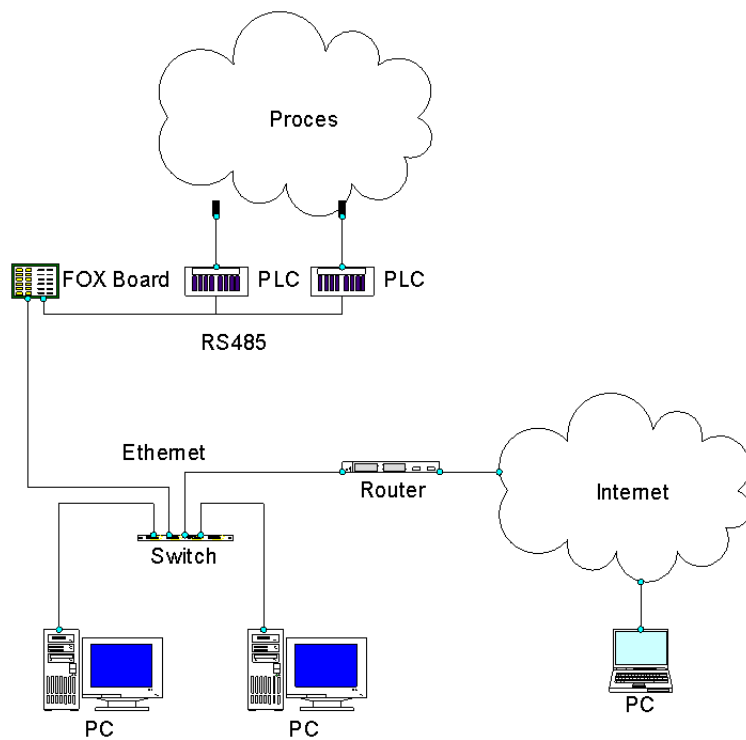
Poslední často využívanou formou sběru dat je záznam za pomoci systému použitého pro řízení vlastního technologického procesu. Záznam dat je prováděn stejným automatem, který řídí proces, často s využitím stejných čidel. Naměřená data jsou v tomto případě většinou přímo tisknuta na vestavěné tiskárně připojené přímo k řídicímu systému. Hlavní výhodou tohoto řešení je nízká cena. Problémem je kvalita použitých tiskáren. Jedná se většinou o jednoduché jehličkové nebo termotiskárny, tisknoucí na úzký papír odvíjený z cívky. Grafy tisknuté z těchto tiskáren mají tedy spíše informativní charakter. Pokud jsou vyžadovány přesné hodnoty je nutno tisknout hodnoty číselně. Tímto způsobem je možné tisknout hodnoty jen s poměrně malou vzorkovací frekvencí, například záznam z procesu z parní sterilizace by měřil několik metrů. Toto řešení je také problematické z hlediska častého požadavku na oddělenost systému pro řízení a systému pro záznam ².

2.3 Návrh systému

Z výše popsaných možností vyplývá, že žádné z běžně dostupných hotových řešení zcela nevyhovuje zadání. Problémy působí především potřeba číst hodnoty veličin z automatů pomocí nestandardního protokolu. Dalším problémem je požadavek na automatické vytváření protokolů z jednotlivých procesů. Z těchto důvodů bylo rozhodnuto vytvořit vlastní zařízení pro záznam dat, které bude možné plně přizpůsobit požadavkům jednotlivých aplikací.

²Pokud je použit jeden systém měření jak pro řízení tak pro záznam dat nemusí být některé problémy vůbec odhaleny. Pokud je například teplota měřena s chybou +1°C, bude teplota regulovaná o 1°C níže než je požadována. Pokud by v tomto případě byl pořizován záznam pouze z čidla použitého pro regulaci, nemusela by být tato chyba vůbec odhalena

Z popisu problému na začátku kapitoly a u jednotlivých monitorovaných technologických procesů vyplývá několik požadavků na navrhované zařízení. Hlavními z nich jsou: Možnost čtení dat z automatů firmy Micropel pomocí sběrnice RS485, schopnost uchování velkého množství naměřených dat, automatické vytváření protokolů specifických pro jednotlivá zařízení a jednoduchý přístup k naměřeným datům ze vzdáleného počítače. Koncepce sběru dat s navrhovaným registračním zařízením je znázorněna na obr. 2.1. Data by měla být čtena přes rozhraní RS485 z automatů firmy Micropel, zaznamenávána pomocí navrhovaného registračního zařízení a zpřístupněna pomocí protokolu http přes rozhraní Ethernet. Celé navrhované zařízení by mělo být možné umístit na DIN lištu do rozvodné skříně. Všechny tyto požadavky je nutné zohlednit při výběru použitých technologií, kterým se zabývá následující kapitola.



Obr. 2.1: Sběr dat v aplikaci parního sterilizátoru

3 POUŽITÉ TECHNOLOGIE

3.1 Programovatelné automaty Micropel

V projektu jsou použity PLC od firmy Micropel. Řada těchto automatů se skládá z automatů s pevnou konfigurací, modulárních automatů a jednoúčelových periférií. Jednotlivé komponenty se spojují do sítě pomocí komunikační linky RS485. Pro snadnější komunikaci je možné použít některý z komunikačních převodníků, které umožňují komunikaci se sítí automatů pomocí rozhraní RS232, USB nebo Ethernet. K programování používají vlastní jazyk Simple.

3.1.1 PES M66

Automat je určen k jednoduchým aplikacím, nevyžadujícím velké množství vstupů a výstupů. Jako poslední z dostupných automatů nepodporuje programování v novějším jazyce SIMPLE V4 a používá starší variantu SIMPLE V2. Tím je ochuzen o některé možnosti novějších automatů. Mezi největší omezení patří možnost použití pouhých dvou datových typů (bit a word). Automat má k dispozici 6 analogových vstupů 0-10 V (použitelných i jako digitální) a 6 digitálních výstupů. Pro indikaci stavu jsou k dispozici 4 programem ovládané kontrolky na čelním panelu, v jejich blízkosti je otočný potenciometr a jedno tlačítko. Vzhledem k výbavě je vhodný hlavně pro jednodušší úlohy.

3.1.2 PES K10 a K1

Tato řada automatů má pevnou konfiguraci vstupů a výstupů. Jsou určeny hlavně jako operátorské panely. Za tímto účelem jsou vybaveny fóliovou klávesnicí, podsvětleným znakovým LCD displayem s 4x20 znaky a zvukovou signalizací. Čelní panel je proveden s krytím IP65 a je připraven k montáži do panelu. V této typové řadě jsou dostupné dva automaty: K10 a K1. K10 je vybaven 6 napěťovými analogovými vstupy 0-10 V, 6 analogovými odporovými vstupy 0-500 Ω a 8 galvanicky oddělenými digitálními vstupy. Z výstupu obsahuje 16 digitálních galvanicky oddělených výstupů a 4 analogové výstupy 0-10 V. Automat K1 má shodné fyzické rozměry jako automat K10, obsahuje však pouze 6 analogových vstupů (použitelných i jako digitální) a 2 digitální výstupy. Tím je určen převážně pro použití jako ovládací terminál.

Jednotlivé automaty v této řadě se liší jen množstvím a typem dostupných vstupů a výstupů.

3.1.3 PES MPC300

Tato řada automatů je řešena jako modulární. Základem je tělo, které obsahuje vlastní procesorovou desku, několik vstupů a výstupů, komunikační rozhraní a v některých variantách také klávesnici a display. K tomuto základu je možné přidat maximálně dva rozšiřující moduly. Mezi dostupné moduly patří klasické digitální vstupy a výstupy, analogové napěťové vstupy a výstupy 0-10V, proudové vstupy 0-25mA a odporové vstupy pro čidla Pt100, Pt1000 a Ni1000.

3.1.4 Jazyk Simple

Všechny dnes dostupné perspektivní automaty firmy je možné programovat v jazyce SIMPLE V4. Jazyk je navržen pro řízení průmyslových procesů a má tedy některé méně obvyklé vlastnosti, podporující bezpečný běh programu.

Celý program je vždy vykonáván od začátku do konce a potom znovu od začátku. Není zde žádná možnost skoků nebo cyklů. Tím je zabráněno vzniku nekonečných smyček. Vlastní program pak většinou realizuje stavový automat. Další vlastností podporující spolehlivost programu je tzv. bezpečná aritmetika (její použití je volitelné). Ta se snaží zabránit nečekaným výsledkům matematických operací. Například zabraňuje přetečení/podtečení čísla tak, že výsledek operace, u které by k němu mělo dojít, nahradí maximální/minimální hodnotou uchovatelnou v cílové proměnné. Podobně řeší dělení nulou, kde opět nahrazuje výsledek maximální kladnou/zápornou hodnotou proměnné. Při dělení nuly nulou je výsledek definován jako nula.

3.1.5 Komunikační protokol PesNet

PLC firmy Micropel mezi sebou komunikují svým vlastním protokolem. Jeho fyzická vrstva je realizována rozhraním RS485. Z hlediska řízení přenosu se jedná o síť typu multi-master s předáváním tokenu. Všechny stanice v síti jsou rovnocenné a postupně si předávají právo vysílat. Tím je zajištěno, že při komunikaci nebude docházet ke kolizím.

Při zapnutí prvních stanic v síti nebo při přerušení komunikace v důsledku chyby, je síť ve stavu, kdy žádná stanice nemá token a celá komunikace je tedy zablokována. Pro obnovení provozu v této situaci každá stanice počítá čas od poslední úspěšné komunikace. Je-li tento čas větší než určitá hranice, automat sám začne vysílat a tím se komunikace obnoví. Délka čekání je dána mimo jiné adresou automatu (ta je unikátní - nemůže dojít ke kolizi). Automat, který se rozhodne zahájit komunikaci, začne posílat tokeny na všechny adresy pod sebou, dokud nedostane odpověď. Po té přebírá řízení další nalezený automat a prohledává dále prostor pod sebou. Tímto

způsobem si automaty vytvoří přehled o obsazení jednotlivých adres. Další komunikace již probíhá jen mezi obsazenými adresami. Po delší době se pokusí automaty poslat token na některou z neobsazených adres, aby mohly být detekovány nově připojené automaty. Nastartování přerušené komunikace může trvat maximálně 5s, nalezení nově připojené stanice může trvat až 20s.

Veškerá komunikace mezi jednotlivými automaty probíhá po rámcích, které obsahují adresu cílového automatu, kód příkazu a kontrolní součet. Některé rámce mají ještě prostor pro data. Komunikace po lince RS485 je poloduplexní. Přenos vždy zahajuje MASTER (automat držící token). U některých příkazů master čeká na odpověď od SLAVE, jiné jsou bez odezvy. Automat předá token dalšímu, pokud nemá další příkazy k vysílání nebo po odeslání maximálně tří příkazů (tím je zabráněno tomu, aby jeden automat blokoval linku příliš dlouho).

Vlastní komunikace probíhá asynchronně s jedním startbitem, jedním stopbitem a devíti bity dat. Data obsahuje prvních osm bitů, poslední označuje začátek rámce. Podporované rychlosti jsou 2400, 9600, 19200, 57600 baudu. Všechny příkazy se dají rozdělit do tří skupin. První jsou příkazy pro čtení a zápis z paměti automatu. Druhá skupina obsahuje příkazy ovlivňující běh automatu a jeho konfiguraci (zastavení, restart, nahrání programu atd.). Poslední skupina obsahuje příkazy řídící komunikaci (předání tokenu). Všechny příkazy, které poskytují odpověď, existují ve dvou variantách. U první automat ihned po jejich přijmutí a zpracování odešle celou odpověď. U druhé varianty odesílá stejnou odpověď, ale odešle jen první byte. Další odesílá po jednom na požádání. Tato varianta je určena pro komunikaci se zařízeními, která by mohla mít problém zpracovat celou odpověď najednou.

Podrobný popis všech příkazů a možných odpovědí na ně je dostupný na stránkách výrobce[3]

3.2 FOX Board G20

FOX Board G20 je jednodeskový počítač vyráběný firmou ACME Systems. Je založen na mikroprocesoru AT91SAM9G20. Kromě procesoru obsahuje 64MB SDRAM, 8MB Flash, obvod reálného času se zálohovací baterií, podpůrné obvody pro rozhraní ethernet a slot na microSD kartu. Díky procesoru s architekturou ARM9 a dostatku paměti umožňuje běh operačního systému Linux. Jediným problémem může být velmi malá paměť Flash určená k ukládání operačního systému, programů a naměřených dat. K jejímu rozšíření je možné použít microSD slot.

3.2.1 AT91SAM9G20

Jedná se o 32bitový procesor od firmy Atmel, postavený na architektuře ARM9. Pro jeho použití v embeded systémech ho předurčuje jeho malá spotřeba (80mW) a velké množství integrovaných periférii. Mezi ně patří:

- Jeden USB device port
- Dva USB host porty
- ethernet 10/100Mb port
- rozhraní pro MMC karty
- čtyři synchronní/asynchronní porty (USART)
- dva asynchronní porty (UART)
- SPI rozhraní
- čtyřkanálový desetibitový A/D převodník

3.3 GSM modem

K navrhovanému zařízení je v některých případech potřeba přistupovat vzdáleně. Pokud není k dispozici připojení k internetu přes podnikovou síť je nevhodnější volbou připojení pomocí sítě GSM. Jeho hlavními výhodami je jednoduchá instalace nevyžadující pokládání kabelů a široká dostupnost. Pro přenos dat v GSM sítích je využíváno několik technologií. Historicky první technologie pro přenos dat je CSD která umožňuje přenos rychlostí 9,6kb/s a je zpoplatněna jako běžný hovor, pro zpřístupnění webového rozhraní je tedy značně pomalá a drahá. Další technologií je GPRS, ta umožňuje, dle varianty protokolu, download i upload rychlostí až 80kb/s (ne zároveň, poměr mezi uploadem a downloadem se nastavuje automaticky v závislosti na charakteru přenosu). S touto rychlostí je již možné webové rozhraní používat, i když odezvy jsou poměrně pomalé. Výhodou GPRS je jeho dobré pokrytí (je dostupné prakticky všude kde jsou dostupné hlasové služby) a cena. Platba většinou probíhá tarifně, nebo je účtován oběm přenesených dat. Z GPRS vychází technologie EDGE (někdy EGPRS) která přináší vyšší rychlost (v závislosti na variantě download až 236.8kbit/s a upload až 118kbit/s). Pokrytí je dnes prakticky totožné s GPRS a cena je shodná. V posledních letech se objevují technologie souhrnně označované jako 3G, které nabízejí podstatně větší rychlosti. Jejich pokrytí je však velmi omezené a tím je také omezena možnost jejich použití.

3.3.1 MC75i

V této práci je pro připojení přes GSM síť použit modem MC75i firmy Sectron. Tento modem podporuje komunikaci pomocí GPRS i EDGE. Připojení k počítači

je možné pomocí USB portu nebo RS232. V obou případech se chová jako Hayes kompatibilní modem, takže v případě potřeby by nebyl problém ho nahradit jiným podobným modemem. Pro připojení antény je přítomen SMA konektor. Napájení je připojeno přes RJ11 konektor. Velikost napájecího napětí se může pohybovat v rozmezí 8-30V. Spotřeba je velice závislá na aktuální činnosti modemu. V nečinnosti, kdy je modem pouze přihlášen do sítě, se pohybuje okolo 30mA. Při datovém přenosu může stoupnout až na 200mA (při napájecím napětí 12V).

3.4 Senzory

Z hlediska vlastního SW pro záznam dat není důležité o jaké snímače se jedná, je jen nutné, aby na straně PLC byl odpovídající vstup. V navrhovaném systému jsou měřeny dvě veličiny: teplota a tlak. Převod odporu na teplotu případně proudu na tlak je prováděn už v PLC. Systém je však při použití vhodných snímačů, použitelný k zaznamenávání libovolné veličiny.

3.4.1 Měření teploty

Pro měření teploty jsou použita čidla Pt100 a Pt1000 připojená do odporových vstupů automatu. Při trojvodičovém nebo čtyřvodičovém připojení je automaticky provedena korekce odporu přírodních vodičů. Převod odporu na odpovídající teplotu je také proveden přímo automatem. Zaznamenání teploty je tedy možné provést přímo přečtením příslušné hodnoty.

3.4.2 Měření tlaku

V navrhovaném systému jsou použity dva různé typy senzorů tlaku. Jeden pro měření vysokých tlaků (10 - 400kPa). Konkrétně se jedná o snímač DMP333, od firmy BD SENSORS, fungující na piezorezistivním principu. Tento snímač již obsahuje elektroniku, která zajišťuje převod odporu a zajišťuje výstup 4-20mA. Druhý typ senzoru slouží k měření velmi malých tlaků (0,1Pa - 100kPa, přesnost se však s rostoucím tlakem velmi rychle zhoršuje). Tento senzor funguje na principu měření tepelné vodivosti prostředí. Výstupní hodnoty napětí jsou v rozsahu 2,2 - 8,5V. Závislost výstupního napětí na tlaku je však silně nelineární (logaritmická), takže je potřeba provést převod měřeného napětí na odpovídající tlak. Tento převod je pro účel řízení procesu prováděn za pomoci tabulky přímo v automatu a pro záznam je tedy opět možné použít přímo hodnotu v paměti automatu.

3.5 Operační systém

Zamýšlenou aplikaci by bylo možné programovat přímo pro mikroprocesor AT91SAM9G20, bez použití operačního systému. Kód obstarávající komunikaci přes RS485 a obsluhu příslušných přerušení by byl dokonce jednodušší než při použití operačního systému. Na druhou stranu použití operačního systému umožňuje použít hotové knihovny a programy, které vývoj některých součástí značně urychlí.

Při použití OS je možné pro ukládání dat použít existující databázový engine. Dosažení podobných vlastností a kvalit vlastním kódem by bylo velmi obtížné a časově náročné. Podobná situace je u webového rozhraní. Nezanedbatelnou výhodou je také možnost vzdálené správy kterou umožňuje většina dostupných OS. Z hlediska flexibility je užitečná existence standartních rozhraní a ovladačů pro některá zařízení.

Z výše uvedených důvodů bylo rozhodnuto vytvořit navrhovaný systém s použitím operačního systému. Operačního systému pro embeded zařízení využívající mikroprocesor ARM je dnes celá řada od MicroC přes WinCE a Linux po Android. Pro tvorbu navrhovaného systému byl vybrán operační systém Linux. Jeho hlavní výhodou je dostatek informací, příkladů a otevřenost, která značně usnadňuje.

3.5.1 GNU/Linux

Pod názvem Linux bývá většinou chápán operační systém postavený na jádru Linuxu, GNU utilit a dalších programů. Přesněji je Linux pouze jádro operačního systému, ke kterému přidáním programů a knihoven z projektu GNU a dalších, vzniká celý operační systém označovaný jako GNU/Linux¹. Naopak GNU může být použito s jiným jádrem, například poslední verze Debianu (6.0 Squeeze) umožňuje výběr mezi Linuxovým a FreeBSD jádrem.

Při použití operačního systému Linux je důležitá také volba distribuce. Množství distribucí podporujících architekturu ARM rychle roste, pro použití v embeded zařízení se však hodí jen některé z nich. Předem je možné vyloučit čistě desktopové distribuce (Ubuntu, Fedora), které se zaměřují na aplikace pro toto použití naprosto zbytečné. Jako vhodné se jeví buď distribuce přímo určené pro embeded zařízení (např. Emdedian, OpenWRT, Kaeilos) nebo nenáročnou serverovou distribuci (např. Debian). Vzhledem k předchozím zkušenostem a dobré podpoře sestavení vlastního operačního systému byla nakonec zvolena distribuce Debian.

¹Linuxové jádro je používáno velmi často právě ve spojení s utilitami z projektu GNU, tím také pravděpodobně vznikly tyto nejasnosti v označení. Nejedná se však o jedinou možnou kombinaci. Linuxové jádro je možné používat bez GNU programů, jako například v operačním systému Android nebo v různých embeded zařízeních, kde je GNU nahrazeno knihovnou uclibc a programy z projektu busybox

3.6 Lighttpd

Lighttpd je open-source webserver, vydaný pod BSD licencí. Mezi jeho hlavní vlastnosti patří malá paměťová náročnost a malá zátěž procesoru. I přes tyto vlastnosti podporuje všechna běžná rozhraní a dosahuje vysokých výkonů. Z hlediska navrhované aplikace je důležitá podpora FastCGI, která umožňuje využití například PHP a podpora šifrovaných přenosů pomocí https. Lighttpd je využíván například službami YouTube, Wikimedia a v mnoha SOHO routerech.

3.7 PHP

PHP (rekurzivní zkratka PHP: Hypertext Preprocessor) je skriptovací jazyk primárně určený pro tvorbu dynamických internetových stránek ve formátu HTML a XHTML. Lze ho však použít i pro tvorbu konzolových a desktopových aplikací. PHP skripty většinou běží na straně serveru a ke klientovy odesílají pouze svůj výstup v podobě HTML kódu. Patří mezi nejrozšířenější jazyky používané pro tvorbu dynamických stránek. Díky tomu je pro něj k dispozici velké množství knihoven pro práci s databázemi apod. Aktuálně je používáno PHP verze 5, jehož hlavní změnou proti předchozí verzi je podpora objektově orientovaného programování.

3.8 XHTML

XHTML (anglická zkratka extensible hypertext markup language) je značkovací jazyk vytvořený jako nástupce HTML. HTML na základě SGML (anglická zkratka Standard Generalized Markup Language), naproti tomu XHTML je vytvořeno na základě XML, které je podmnožinou SGML. XHTML přebírá vlastnosti HTML, ale v některých ohledech je restriktivnější. Na rozdíl od HTML musí být všechny tagy párové, hodnoty atributů musí být v úvozovkách a názvy atributů a hodnot jsou case sensitive. Výhodou XHTML je možnost použít běžný XML parser a rozšířitelnost. Díky rozšířitelnosti je například možné přímo do zdrojového kódu XHTML stránky vložit SVG obrázek, který je také založen na XML technologii.

3.9 CSS

CSS (anglická zkratka Cascading Style Sheets) je jazyk určený k popisu vzhledu dokumentů HTML, XML a z něho vycházejících. Hlavním účelem tohoto jazyka je oddělit obsah dokumentu od jeho vzhledu. Díky tomu je možné jeden dokument zobrazovat několika odlišnými způsoby, například v závislosti na zařízení použitém

k jeho zobrazení. V této práci je vlastnosti využito pro vytvoření rozdílných stylů pro zobrazení stránky na monitoru a pro tisk. V současné době je používána verze CSS2.1 a pracuje se na verzi CSS3.

3.10 SQLite

SQLite je databázový engine, původně vytvořený pro systém raketového torpédo-borce, nyní vydaný pod public domain licenci. Na rozdíl od většiny databázových systémů, které jsou uspořádány jako klient-server, SQLite neběží jako samostatný proces. Místo toho se jedná o knihovnu, která může být staticky nebo dynamicky přilinkována a stane se součástí programu. Tímto přístupem je zaručena nižší latence a nároky na výkon než při meziprocesové komunikaci. Celá databáze je uložena v jediném souboru. Konkurenční přístup je pak řešen zamykáním tohoto souboru. K databázi může paralelně přistupovat několik procesů, pouze pokud se jedná o přístup pouze pro čtení. Pokud se do databáze zapisuje, nesmí probíhat žádná jiná operace. Výhodou tohoto přístupu je jednoduchost a malá náročnost. Problém může nastat při vyšším zatížení databáze, protože v důsledku velkého množství konkurenčních přístupů dochází k velkému snížení výkonu.

Zvláštností SQLite je přístup k datovým typům. Při tvorbě tabulky je možné definovat datové typy jednotlivých sloupců. Při další práci s tabulkou však nejsou nijak kontrolovány² a je tedy možné do sloupce typu INT uložit řetězec apod. Definované datové typy jsou použity jen v případě použití klauzule ORDER BY.

SQLite je používáno například v aplikacích Mozilla Firefox, Mozilla Thunderbird a v operačních systémech některých mobilních telefonů (Symbian, Maemo, BlackBerry, Android, iPhone). Jeho hlavní výhodou v těchto aplikacích je malá velikost a nízké nároky na paměť. Ze stejných důvodů se hodí i pro embedded aplikace.

²Vyjímkou je typ INTEGER PRIMARY KEY, který musí obsahovat celočíselné hodnoty

4 NÁVRH A IMPLEMENTACE

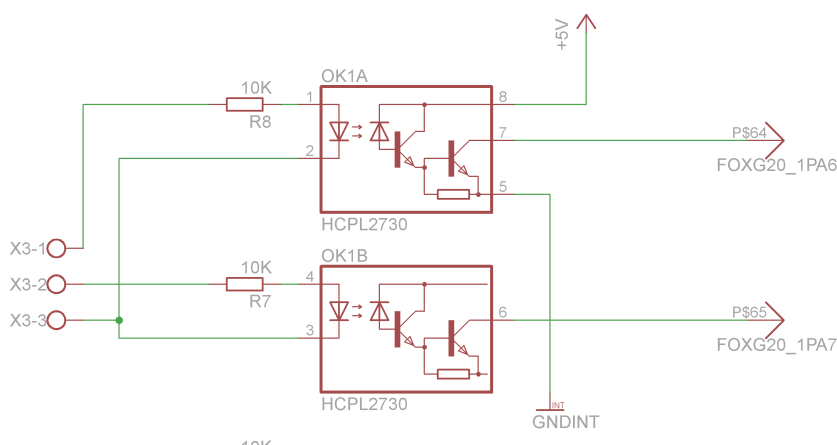
4.1 Podpůrné obvody

Samotný FOX Board obsahuje jen součásti nezbytné pro svoji funkci: Zdroj napětí pro mikroprocesor, baterii pro hodiny reálného času, integrovaný obvod pro ethernetové rozhraní, paměť RAM a FLASH. Pro zamýšlené použití je potřeba doplnit další obvody zajišťující potřebné funkce.

4.1.1 Digitální vstupy a výstupy

Hlavním zdrojem dat pro záznam jsou připojená PLC. Během řešení však nastala potřeba vstupů a výstupů připojených, přímo k FOX Boardu. Vstup je nutný zejména pro signalizaci výpadku napájení od externího zdroje. Při výpadku napájení jsou PLC nefunkční a FOX Board by bez vlastních vstupů byl úplně odříznut od okolí. Výstupy mohou být využity například pro signalizaci stavu atd.

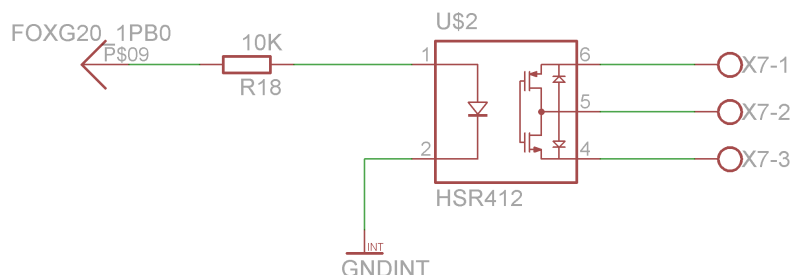
Mikroprocesor AT91SAM9G20 má celkem 3 PIO porty po 32 vstupně/výstupních pinech. Některé piny, ale vůbec nejsou vyvedeny na kolíkovou lištu. Část z vyvedených pinů neslouží jen jako universální vstupy/výstupy, ale sdílí svou funkci s dalšími perifériemi (UART, I2C, MMC atd.), u těch je nutné zvolit, zda budou použity příslušné periférie, nebo budou použity jako universální vstupy/výstupy. U všech pinů universálních vstupů/výstupů lze navíc nastavit, zda se mají chovat jako vstupy nebo výstupy.



Obr. 4.1: Schéma digitálních vstupů

Maximální vstupní napětí většiny vstupů je 3.3V (některých jen 1.8V). Všechny vstupy jsou vybaveny Schmittovým klopným obvodem a možností připojení vnitřního pull-up odporu. Pro galvanické oddělení a úpravu napěťové úrovně vstupů je

použit dvojitý optron HCPL273 [13]. Odpor připojený sériově k vnitřní LED optronu je navržen pro vstupní napětí 24V.



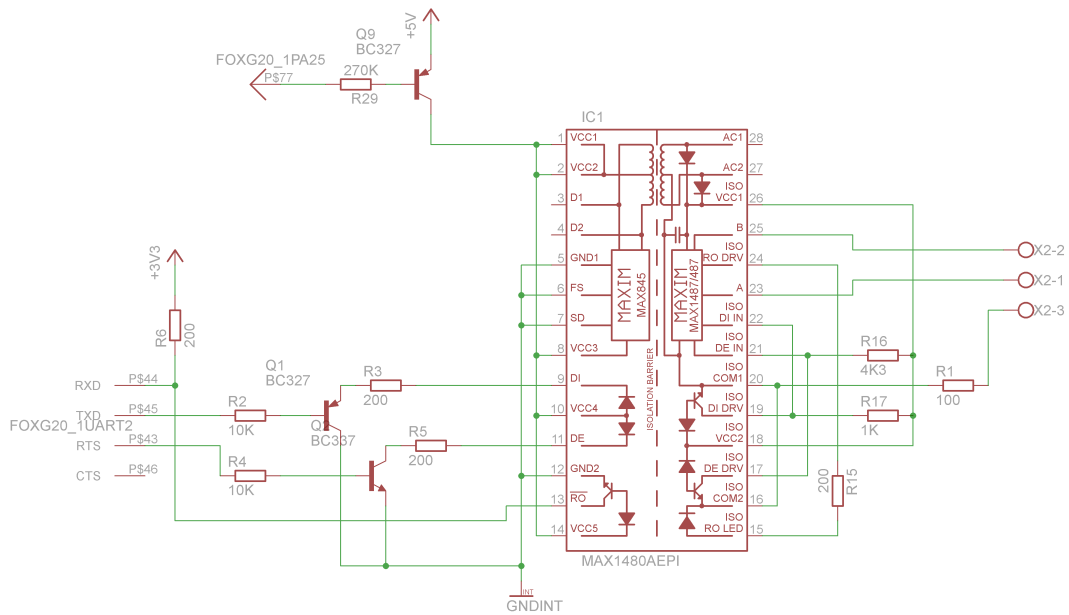
Obr. 4.2: Schéma digitálních výstupů

Napěťové úrovně většiny výstupních pinů jsou také 3.3V, u některých 1.8V. Proudová zatížitelnost je také různá. Vstupy 3.3V je možné zatížit maximálně 8mA. Pro posílení a oddělení výstupů je použit optron HSR412L [14]. Ten obsahuje na výstupní straně MOSFET tranzistory. Ty umožňují spínat poměrně velké zátěže (až 200mA). Zároveň mohou pracovat s napětím 400V, takže jsou použitelné k přímému spínání síťového napětí. Pokud je v době výroby jisté, že nebudou potřeba galvanicky oddělené vstupy a výstupy, je možné příslušné obvody neosadit.

4.1.2 Budič sběrnice RS485

Pro komunikaci se sítí PLC je potřeba rozhraní RS485. FOX Board přímo toto rozhraní neobsahuje, má však UART s podporu RS485. Podpora spočívá v možnosti přepnout signál RTS do režimu, ve kterém přechází do stavu high při odesílání znaků. Tento signál je tedy možné použít k přepínání budiče sběrnice mezi režimem vysílání a režimem vysoké impedance.

Na trhu je velké množství budičů sběrnice RS485. Mezi rozšířené typy patří MAX485, který byl použit při vývoji a testování ovladače. Při nasazení v reálném provozu bývá sběrnice rozlehlá, a proto je vhodné, aby byla galvanicky oddělená od zbytku systému. Některé budiče s tímto požadavkem počítají a obsahují optrony, které zajistí oddělení. Jejich nevýhodou je potřeba dalšího nezávislého zdroje napětí, pro napájení výstupní strany. Pro jeho získání často obsahuje budič pomocné obvody pro vytvoření izolovaného zdroje napětí, vyžadující jen doplnění vnějšího transformátoru.



Obr. 4.3: Schéma budiče RS485

V této práci byl zvolen obvod MAX1480ACPI [15]. Tento obvod obsahuje optrony zajišťující galvanické oddělení a obvody pro vytvoření DC/DC měniče. Na rozdíl od většiny ostatních obvodů obsahuje i potřebný transformátor. Diody v optronu mají poměrně velkou spotřebu (20mA). Výstup z FOX Boardu je tedy potřeba posílit tranzistorem. Vlastní spotřeba obvodu MAX1480ACPI může být v závislosti na zatížení sběrnice až 145mA. Obvod budiče tedy může celkově odebírat téměř 200mA. To je přibližně dvojnásobek spotřeby vlastního FOX Boardu. Vzhledem k nutnosti zálohovat napájení je vhodné mít možnost celý obvod budiče odpojit od napájení. To je zajištěno za pomoci tranzistoru Q9. Jeho báze je přes odpor připojena k vstupu FOX Boardu použitého pro detekci napájecího napětí na vstupu a zajistí tedy odpojení budiče v okamžiku výpadku proudu.

4.1.3 Zdroj

Pro napájení počítače FOX Board a podpurných obvodů je použit zdroj 24V stejnosměrného napětí který je k dispozici přímo v rozvaděči. FOX Board však vyžaduje napětí 5V. Dalším požadavkem na napájení je zabránění náhlé ztráty napájecího napětí v případě přerušení napájení z rozvaděče. V případě výpadku vnějšího zdroje by měl zůstat FOX Board v provozu po dobu nutnou k bezpečnému ukončení práce

(ukončení zápisu do databáze, odmountování disku). Tato doba byla experimentálně určena jako 30s.

V UPS bývají většinou pro zálohování používány olověné baterie. Hlavními pozitivy olověných baterií je poměrně jednoduché nabíjení a možnost odebrat velký proud. Spolu s dalšími chemickými akumulátory však sdílejí některé nedostatky. Z hlediska navrhovaného zařízení je důležitá dlouhá životnost při poměrně vysokých teplotách (až 65°C), rychlé nabíjení a možnost jednoduchého testu kapacity. Z těchto důvodů byly namísto běžných akumulátorů použity superkapacitory, které mají oproti akumulátorům delší životnost, je možné je nabíjet vysokou rychlostí a test kapacity je možné provést měřením doby vybíjení přes známý odpor. Celý tento test může proběhnout poměrně rychle.

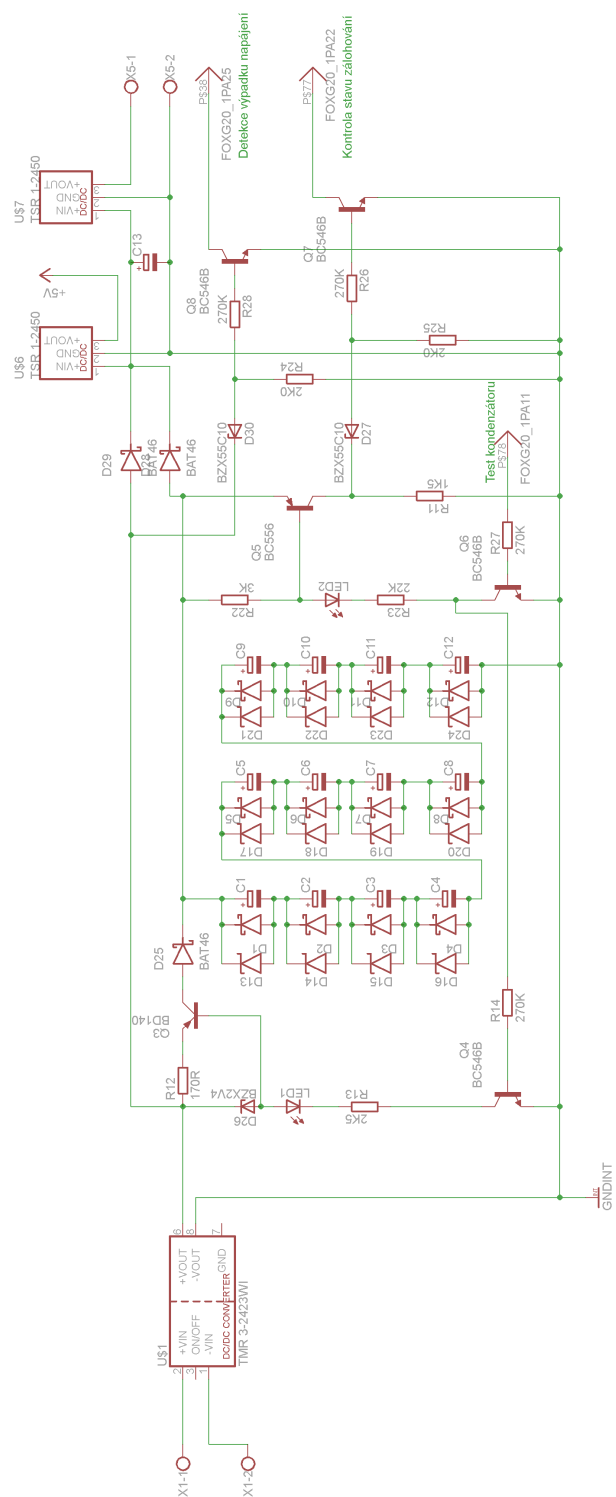
Hlavním parametrem při návrhu zálohovaného zdroje je potřebná doba zálohování. Aby byla k dispozici dostatečná časová rezerva pro vypnutí systému i v případě poklesu kapacity kondenzátorů, měl by být tento zdroj navržen alespoň na 60s provozu. Po vypnutí budiče sběrnice RS485 zbývá jako jediný velký spotřebič FOX Board. Výrobce udává spotřebu bez zátěže jako 80mA při 5V. Při praktickém měření při plném zatížení nebyla nikdy naměřena spotřeba větší než 100mA. Při účinnosti spínaného zdroje 80% (výrobce udává min. účinnost 84%) je tedy možné včetně podpůrných obvodů počítat s odběrem 650mW na vstupní straně DC/DC měniče s výstupem 5V. Pro provoz po dobu 60s je tedy potřeba energie 39J. Energie uložená v kondenzátoru je určena vztahem 4.1

$$E = \frac{1}{2}CU^2 \quad (4.1)$$

Z hlediska využití maxima energie uložené v kondenzátorech je lepší použít sériové zapojení kondenzátorů¹. Na základě těchto požadavků byla vybrána konfigurace zobrazená na obr. 4.4. Dvanáct kondenzátorů se jmenovitým napětím 2.7V a kapacitou 2.2F je zapojeno do série a nabíjeno na napětí 30V, získané pomocí DC/DC měniče. Dosazením do vztahu 4.1 tedy určíme energii v plně nabitě baterii kondenzátorů jako 82J. Výstupní měnič však přestane fungovat při poklesu napětí pod 6.5V a 4J tedy zůstanou nevyužitý. Celkově je tedy k 78J a navržený zdroj má tedy dostatek kapacity pro napájení zařízení po požadovanou dobu včetně rezervy pro případ poklesu kapacity kondenzátorů.

Každý kondenzátor má paralelně připojenou jednu Zenerovu a jednu Schotkyho diodu. Tyto diody zajišťují ochranu kondenzátorů v případě že nemají stejnou kapacitu. Zenerova dioda zabraňuje, aby v důsledku rozptylu jmenovité kapacity, došlo

¹Při vybíjení kondenzátorů postupně klesá napětí. Pokud je tedy na výstupu zálohovaného zdroje vyžadováno konstantní napájení je nutné použít spínaný regulátor. Ten je schopen pracovat jen s určitým rozsahem vstupních napětí. Zvolený regulátor umí pracovat se vstupním napětím 6.5 - 36V. Z hlediska využití maxima energie je tak možné využít co největší část tohoto rozsahu.



Obr. 4.4: Schéma zdroje

k překročení jmenovitého napětí kondenzátoru. Schotkyho dioda chrání slabší kondenzátory předpřepólováním při vybíjení.

Test kapacity kondenzátorů je možné spustit nastavením výstupu PA11 do stavu high. Tím dojde k přerušení nabíjení kondenzátorů a připojení vybíjecího odporu R11. Pomocí vstupu PA22 je možné sledovat kdy poklesne napětí na kondenzátorech pod 10V. Z naměřené doby je pak možné odvodit kapacitu kondenzátorů. FOX Board je po dobu tohoto testu napájen rovnou z výstupu DC/DC měniče sloužícího pro nabíjení kondenzátorů a nehrozí tedy že by ovlivňoval svou spotřebou výsledek měření.

Výpadek napájecího napětí je možné sedovat pomocí vstupu PA25.

4.2 Příprava operačního systému

Instalace operačního systému na embedded systémech má naprosto odlišný postup než v případě běžných PC (AMD64 a IA-32). Při instalaci na PC je díky poměrně velké standardizaci základních rozhraní možné připravit univerzální instalační medium, které umožňuje úspěšně nainstalovat operační systém i nezkušenému uživateli. U embedded systému (ale i spotřební elektroniky, mobilních telefonů, televizí apod.) je situace značně odlišná. I pokud zařízení používají mikroprocesor se stejnou architekturou, jsou mezi nimi velké rozdíly ve způsobu připojení a typu periférií, způsobu připojení paměti a postupu zavádění programu při spuštění. Často není vůbec přítomné rozhraní pro připojení vhodného instalačního media. Operační systém pro tato zařízení je tedy nutné připravit na míru konkrétnímu hardware a vlastní instalační proces je také náročnější.

V případě GNU/Linuxu je potřeba upravit pro konkrétní zařízení hlavně jádro operačního systému. To zprostředkovává funkce hardwaru uživatelským programům a je tedy závislé na konkrétní konfiguraci zařízení. Uživatelské programy jsou od hardware odděleny abstrakcí poskytovanou právě jádrem a tak jsou na konkrétní konfiguraci hardware poměrně nezávislé. Pro jejich funkčnost tedy většinou stačí aby byly zkompileovány pro příslušnou procesorovou architekturu.

4.2.1 Kompilace jádra

Z výše uvedených důvodů je nutné připravit vlastní Linuxové jádro. Při přípravě operačního systému byla použita v té době aktuální verze Linuxového jádra 2.6.35.4. a popsáný postup tedy platí pro tuto verzi. Postup pro novější jádra by byl velice podobný, od verze 2.6.38 však odpadá nutnost aplikace patche od firmy ACME který přidával do kódu jádra informace nutné pro správnou inicializaci hardware

specifického pro FOX Board, protože tento patch byl přijat do hlavní vývojové větve jádra a je tedy přímo součástí všech následujících verzí jádra.

Prvním krokem potřebným pro kompilaci vlastního jádra je připravení prostředí pro kompilaci. Toto prostředí je souhrnem nástrojů, knihoven a kompilátoru potřebným k vytvoření binárního obrazu jádra ze zdrojového kódu. Ve světě osobních počítačů je příprava tohoto prostředí poměrně jednoduchá a v případě běžných distribucí spočívá v instalaci několika málo předpřipravených balíků programů které obsahují všechny výše popsané součásti. Tyto předpřipravené balíky však předpokládají že cílová platforma je shodná s platformou na které probíhá kompilace. Jinak řečeno kompilátor běžící na platformě AMD64 vytváří spustitelný kód pro AMD64 a kompilátor běžící na platformě IA-32 vytváří spustitelný kód pro IA-32. Při kompilaci jádra pro delší platformy lze postupovat stejným způsobem. Linuxová distribuce Debian ve verzi pro mikroprocesor ARM, použitá v tomto projektu, obsahuje všechny k tomu potřebné nástroje. Pokud je tedy na cílovém zařízení funkční systém je možné zkompilovat jádro přímo na něm. V případě desky FOX Board je možné získat od výrobce SDKartu s již připraveným systémem, takže je možné přistoupit ke kompilaci jádra přímo na FOX Boardu. Problémem při tomto postupu je poměrně nízký výkon použitého procesoru, malá paměť RAM a nedostatek místa na SDKartě sloužící jako jediné úložiště dat. Proces kompilace by byl tedy poměrně komplikovaný a hlavně velice zdlouhavý. U velkého množství zařízení není přímá kompilace vůbec možná, už jen kvůli tomu že na něm není k dispozici funkční systém na kterém by mohla kompilace proběhnout. Tyto problémy se dají obejít křížovou kompilací. Ta spočívá v tom že kompilace probíhá na jiné architektuře než pro kterou je určen výsledný binární soubor. Nástroje používané pro běžnou kompilaci však nejsou schopné provádět křížovou kompilaci a je tedy nutné připravit novou sadu nástrojů tzv. toolchain schopný křížové kompilace. Minimum pro křížovou kompilaci je upravená verze binutils (nástroje pro překlad z assembleru a linker), překladač gcc a knihovna libc. Příprava tohoto toolchainu je poměrně složitý proces. Pokud je však pro kompilaci použita linuxová distribuce založená na debianu je možné použít předpřipravený toolchain z projektu Emdebian [6]. V tomto projektu byla pro kompilaci jádra použita distribuce Debian a příprava toolchainu byla tedy značně zjednodušená.

Aby bylo možné využít repozitář² projektu Emededian je ho potřeba přidat do seznamu používaného správcem balíků.

Toho je docíleno přidáním následujícího záznamu do souboru `/etc/apt/sources.list`:
deb http://www.emdebian.org/debian/ squeeze main Aby byl tento repozitář považován za důvěryhodný a bylo možné ho používat je ještě nutné nainstalovat příslušný klíč příkazem `aptitude install emdebian-archive-keyring`. Tím je dokončeno přidání nového repozitáře a po aktualizaci seznamu dostupných balíků pomocí příkazu `aptitude update` je možné přistoupit k instalaci nástrojů potřebných pro křížovou kompilaci. Pro vlastní kompilaci jádra jsou potřeba balíky `binutils-arm-linux-gnueabi`, `gcc-4.3-arm-linux-gnueabi` a `uboot-mkimage`. První dva uvedené jsou potřeba k vlastní kompilaci jádra. Třetí obsahuje nástroj, který umí zabalit jádro do souboru doplněného o další informace (CRC32, nastavení jádra atd.), který je možné zavést pomocí bootloADERu v cílovém zařízení. Po dokončení těchto kroků je k dispozici plnohodnotný toolchain připravený pro kompilaci nového jádra.

Prvním krokem při kompilaci je získání zdrojového kódu jádra. Ten je možné získat například z [7]. Na stažený a rozbalený zdrojový kód jádra, před vlastní kompilací, je možné aplikovat jeden nebo více patchů upravujících funkci jádra. Vzhledem k tomu že použitý mikroprocesor je již jádrem podporován stačí poměrně malé množství úprav, které zajišťují specifické nastavení odpovídající připojeným obvodům na FOX Boardu. Především se jedná o konfiguraci microSD karty, UART portů a ethernetového řadiče. Tento patch je dostupný na stránkách výrobce FOX Boardu[8]. Vzhledem k tomu že jeden z UART portů bude použit s vlastním ovladačem je navíc potřeba přidat patch `RS485.patch` (na příloženém CD), který odstraní kód který inicializuje `USART1` jako běžný sériový port a uvolní ho pro použití vlastním ovladačem. Aplikací těchto dvou patchů je připraven vlastní zdrojový kód a je možné přistoupit ke konfiguraci jádra.

Linuxové jádro je velice rozsáhlé a může běžet na velice rozdílných platformách s podporou různých funkcí a hardware. Aby bylo možné z jednoho zdrojového kódu možné vytvářet tolik rozdílných variant binárních obrazů jádra je používán mechanismus podmíněného překladu. Až během kompilace se na základě nastavení

²V OS GNU/Linux a mnoha dalších se provádí instalace software odlišným způsobem než jaký je běžný u OS Windows. Pro instalaci software slouží balíčkovací systém který zajistí instalaci z připraveného balíčku. Tento balíček obsahuje soubory určené ke zkopírování do systému, ale také instalační skript který zajistí další potřebná nastavení. Instalační balíček může být dodán přímo uživatelem, většinou je však obstarán přímo balíčkovacím systémem, z repozitářů spravovaných vydavatelem příslušné distribuce. Uživateli stačí zadat název požadovaného software a balíčkovací systém se sám postará o stažení instalačních balíků z repozitáře příslušné distribuce a jejich následující instalaci. Instalace z repozitářů umožňuje jednodušší a bezpečnější (při použití důvěryhodného repozitáře) instalaci software a jednotnou kontrolu aktualizací pro všechny nainstalovaný software.

uložených v souboru `.config` vybírají konkrétní možnosti. Pro editaci tohoto konfiguračního souboru jsou k dispozici dvě možnosti. První je dostupná přes příkaz `make config` a konfigurace probíhá přes sérii otázek. Druhou možností je použití příkazu `make menuconfig`, v tomto případě probíhá konfigurace pomocí vnořených menu. Jako základní konfiguraci je možné použít soubor `.config` poskytnutý výrobcem FOX Boardu[8]. V tomto konfiguračním souboru je většina ovladačů kompilována jako moduly, takže zbytečně nezvyšují velikost jádra i tak je však většina z nich zbytečná a je tedy možné je vypnout (ovladače pro SATA, USB zvukové a grafické karty apod.). Naopak je potřeba přidat podporu pro PPP protokol a TUN rozhraní. Upravený soubor `.config`, obsahující výše popsaná nastavení, je na přiloženém CD.

Po dokončení konfigurace je možné příkazem `make` spustit kompilaci jádra a následně příkazem `make modules` kompilaci modulů. Takto vytvořené jádro je ještě nutné připravit do souboru použitelného zavaděčem U-Boot. K tomu je možné použít soubor `makefile`³ ze stránek výrobce výrobce FOX Boardu[8]. Nově zkompilevané jádro je díky přidanému `makefile` zabaleno do souboru `uImage` s přídavnými informacemi potřebnými pro zavaděč U-Boot. Zkompilevané ovladače jsou v podadresáři `foxx20-modules`.

4.2.2 Instalace operačního systému

Linuxová distribuce Debian je poměrně dobře připravena na instalaci v případech kdy není možné použít klasický instalátor. V těchto případech je možné použít nástroj `debootstrap` který umožňuje nainstalovat Debian z prostředí jiného běžícího systému do libovolného adresáře. V případě tohoto projektu je jako původní systém použit také Debian což celý proces značně usnadňuje. Instalace vlastního `debootstrap` spočívá pouze v instalaci jediného balíku příkazem `aptitude install debootstrap`.

Před vlastní instalací je potřeba připravit paměťovou kartu. Na nově dodávaných paměťových kartách bývá zpravidla jeden oddíl se souborovým systémem `fat32` přes celou kartu. Toto rozdělení je pro použití s FOX Boardem nevhodné a je ho tedy potřeba změnit. Na začátku je potřeba odstranit původní oddíl. Poté je možné vytvořit první malou partition o velikosti 32MB, která bude sloužit pro uložení jádra. Další partition s velikostí 1GB bude použita pro vlastní OS. Poslední partition je určená pro zaznamenávaná data a využívá zbývající místo (v případě 2GB karty tedy necelý 1GB).

³Program `make` hledá v aktuálním adresáři soubory `makefile` a `Makefile` v tomto pořadí. Přidaný `makefile` je tedy zavolán dříve než původní `Makefile` dodávaný s jádrem. Nový `makefile` využívá funkce původního, jen přidává schopnost vytvořit z jádra obraz pro U-Boot

Pro vytváření a úpravu oddílů je možné pod OS GNU/Linux použít nástroj fdisk[9]. Na takto vytvořených oddílech je následně potřeba vytvořit souborový systém. Zavaděč UBOOT umí pracovat jen se souborovým systémem FAT32 a první oddíl určený pro uložení jádra tedy musí být v tomto formátu. Naformátovat oddíl v tomto formátu je možné příkazem `mkfs.vfat /dev/sdX1`. Další dva oddíly je možné zformátovat souborovým systémem ext2 příkazem `mkfs.ext2 /dev/sdXY`. Aby bylo možné s těmito oddíly pracovat je potřeba je připojit. To je v unixových operačních systémech možné příkazem `mount /dev/sdXY /mnt/sdXY` Na první oddíl připravené paměťové karty je následně možné zkopírovat dříve připravený soubor s jádrem uImage.

Následně je možné přistoupit k vlastní instalaci. V nejjednodušším případě stačí použít příkaz `debootstrap -arch armel -download-only -foreign squeeze /mnt/sdX2 http://ftp.us.debian.org/debian`. Ten zajistí stažení balíčků obsahujících základní systém a jejich rozbalení do určeného adresáře. Parametry `foreign` a `download-only` zajistí že nedojde k spuštění instalačních skriptů. Toto opatření je nutné protože cílová architektura je rozdílná od aktuální a instalační skripty které jsou závislé na architektuře by tedy nemusely proběhnout. Druhou část instalace je tedy nutné provést na cílovém zařízení. Za tímto účelem je potřeba nastavit aby byl při spuštění systému spuštěn soubor `/stage2`, který zajistí dokončení instalace na cílovém zařízení. Tohoto automatického spuštění je možné dosáhnout vložением řádku `"si:1:respawn:/stage2"` do souboru `/etc/inittab` (namísto běžných `init` skriptů tedy proběhne druhá fáze instalace).

Tuto paměťovou kartu je možné vložit do FOX Boardu a po připojení napájení a zavedení jádra se spustí druhá fáze instalace. Průběh této instalace je možné sledovat přes rozhraní RS232 na příslušném konektoru FOX Boardu. Po skončení instalace je na paměťové kartě nainstalována minimální, ale plně funkční verze Debianu. Další funkce a programy je možné doinstalovat běžným způsobem za pomoci balíčkovacího systému.

Tento postup je odvozen z návodu na stránkách firmy ACME[8] a podrobně je rozepsán v příloze B.1.

4.2.3 Spouštění a vypínání systému

Vzhledem k specifickým požadavkům na chování během spouštění systému a v případě výpadku proudu, je nutné provést několik úprav do procesu spouštění OS. Navíc by měly být přidány skripty pro spouštění nově vytvořených služeb.

U běžných osobních počítačů je jako první spuštěn kód biosu, jeho úkolem je provést nutnou inicializaci hardware a poté najít paměťové médium obsahující operační systém, či spíše zavaděč operačního systému. Ten se většinou nachází v MBR

pevného disku (může být však využit například zavaděč uložený v EEPROM na síťové kartě), nalezený kód je zkopírován do paměti RAM a spuštěn. Zavaděč na rozdíl od BIOSu dokáže pracovat se souborovým systémem⁴ na pevném disku a může z něj tedy načíst obraz jádra do RAM a následně jej spustit.

V případě mikroprocesoru AT91SAM9G20 použitým ve FOX Boardu je postup velice podobný. Na rozdíl od osobních počítačů však chybí standardizovaná vrstva v podobě rozhraní BIOS. Namísto něj je bezprostředně po připojení napájení spuštěn program uložený v paměti ROM procesoru. Ten se pokusí najít další kód ke spuštění na externě připojené FLASH paměti. Pokud neuspěje čeká na příkazy posílané přes sériový port nebo USB, s jejichž pomocí je možné nahrávat data do připojené paměti FLASH nebo nabootovat a spustit program odeslaný přes sériový port. Na FOX Boardu je umístěna paměť NAND FLASH obsahující dva zavaděče⁵ AT91BootStrap a U-Boot. Jako první je spuštěn AT91BootStrap, který zajistí základní konfiguraci mikroprocesoru a poté spustí U-Boot. Ten z prvního oddílu paměťové karty načte do RAM obraz jádra a spustí jej.

Jádro po svém spuštění dokončí inicializaci procesoru (nastavení virtuální paměti, obsluha přerušení atd.) a přejde k inicializaci funkcí jádra (alokace paměti, plánovač atd.). Na konci inicializace jádra je vytvořen proces `init`, první program běžící v uživatelském prostoru⁶. Úkolem procesu `init` je spouštět a zastavovat ostatní programy v závislosti na `runlevel`/footnote. Většina programů však není spouštěna přímo ale přes tzv. `init` skripty umístěné v adresáři `/etc/init.d/`. `Init` je nakonfigurován tak aby při změně `runlevelu` spouštěl skript `/etc/init.d/rc`, který dále prohledává adresáře `/etc/rcX.d/` (kde `X` je číslo nového `runlevelu`). V těchto adresářích jsou umístěny simlinky odkazující na skripty které mají být v tomto `runlevelu` spuštěny. Název simlinky se skládá z písmena určujícího zda má být skript spuštěn s parametrem `start` (písmeno `S`), nebo `stop` (písmeno `K`), za ním následuje dvojčíferné číslo určující pořadí skriptu a název vlastního skriptu. Pořadí spouštění `init` skriptů bylo

⁴Práce se souborovým systémem je poměrně složitá a velikost kódu který je možné uložit do MBR je pouze 440B. Například zavaděč GRUB je z tohoto důvodu rozdělen do několika částí. První část, označovaná jako `stage 1`, je uložena v MBR a jejím jediným úkolem načíst je načíst následující část. Ta je označována jako `stage 1.5` a většinou je uložena bezprostředně za MBR nebo na vlastní partition. Její velikost může být větší a je v ní tedy možné realizovat funkce potřebné pro práci se souborovým systémem. Díky tomu může ze systémového oddílu načíst `stage 2`, která zpracuje konfigurační soubor a následně načte a spustí jádro operačního systému.

⁵Novější revize FOX Boardu jsou vybaveny pouze jediným zavaděčem `AcmeBoot`. Tento zavaděč je vytvořen na míru pro FOX Board a umožňuje rychlejší start.

⁶V Linuxu a většině ostatních víceúlohových víceuživatelských operačních systémech, je paměť rozdělena na jaderný prostor (`kernel space`) a uživatelský prostor (`user space`). V jaderném prostoru běží vlastní jádro operačního systému a ovladače. Běžné aplikace běží v uživatelském prostoru a mají přístup pouze k paměti kterou jim na požádání alokovalo jádro.

dříve určováno ručně. Zvláště při použití Linuxu na desktopech se však stala rychlost spouštění systému důležitým parametrem a jedním ze způsobů jak dosáhnout zrychlení je paralelní spouštění init skriptů. V případě vícejádrových procesorů je přínos paralelního spouštění skriptů evidentní, podstatné zrychlení ale nastává i u jednojádrových procesorů, protože mnoho procesů není bržděno výkonem procesoru ale prací s pevným diskem nebo sítí. Všechny skripty však není možné spustit paralelně, protože služby spouštěné init skripty na sobě vzájemně závisí. Například pro spuštění webserveru musí být funkční síť a daemon obstarávající logování a pro logování musí být připojen oddíl obsahující logy. Aby nebylo nutné tyto závislosti řešit ručně byly vytvořeny nástroje které umožňují tento proces automatizovat. V případě Debianu se jedná o program insserv. Ten na základě informací v hlavičce initskriptu určí kdy může být příslušný skript spuštěn a na základě toho vytvoří potřebné simlinky.

4.2.4 Init skripty

Init skripty zajišťují spouštění služeb (poskytovaných programy nazývanými daemony) ale také jednorázové akce jako je konfigurace sítě nebo připojení diskových oddílů. Aby bylo možné automaticky určit v jakém pořadí má docházet k jejich spouštění musí ve své hlavičce obsahovat potřebné informace o jejich vzájemných závislostech. Formát této hlavičky používaný v Debianu byl určen v rámci projektu Linux Standard Base[10]. Ukázka hlavičky skriptu vytvořeného pro navržené zařízení je zde:

```
# Provides:          logger
# Required-Start:    $syslog $time $pesnet
# Required-Stop:     $syslog $time $pesnet
# Should-Start:
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start logger daemon.
# Description:
```

Hodnota Provides určuje název služby poskytované tímto skriptem. Přes tento název se na tento skript odkazují další skripty které ke svému běhu tuto službu potřebují. Hodnoty Required-Start a Required-Stop definují, které další služby musí běžet aby mohl být příslušný skript spuštěn s parametrem start, případně stop. Hodnota Should-Start určuje které služby by měly být spuštěny, pokud jsou k dispozici. Pokud však k dispozici nejsou je možné je přeskočit. Hodnoty Default-Start a Default-Stop definuje runlevely na kterých má být služba spuštěna či zastavena. Pro správnou funkci navrhovaného zařízení muselo být několik nových init skriptů

vytvořeno a několik původních upraveno. Nově vytvořené a upravené skripty jsou zde stručně popsány a uloženy na příloženém CD.

gpioinit

Tento skript inicializuje stav universálních vstupů a výstupů. Pin PA11, sloužící pro testování kapacity zálohovacích kondenzátorů, je nastaven jako výstup a jeho hodnota je nastavena na 0. Tím je zajištěno jejich nabíjení. Piny PA25 a PA22, sloužící pro čtení stavu napájecího napětí a napětí na kondenzátorech jsou nastaveny jako vstupy.

powerwait

Úkolem tohoto skriptu je pozastavit proces startování systému až do doby kdy nejsou dostatečně nabity zálohovací kondenzátory. Vlastní skript nespouští žádnou službu ani neprovádí žádné nastavení. Pouze v nekonečné smyčce čeká na změnu stavu na vstupním pinu PAxx. Jakmile přejde do stavu X skript se ukončí. Zároveň s přidáním tohoto skriptu byl upraven skript mountall tak aby ke svému spuštění vyžadoval dokončení skriptu powerwait. Tím je zajištěno že disk nebude připojen pro zápis dříve než je v kondenzátorech dostatek energie na bezpečné dokončení tohoto zápisu a zároveň nejsou blokovány další skripty, které ke svému běhu nevyžadují zápis na disk.

powercheckstart

Jediným úkolem tohoto skriptu je spustit na pozadí další skript, powercheck. Ten běží trvale na pozadí a v jednosekundových intervalech kontroluje stav vstupu PA25. Případný výpadek proudu je signalizován přechodem vstupu PA25 ze stavu L do H. Pokud je stav vstupu PA25 dvakrát za sebou H (tím je zajištěno ignorování krátkých výpadků) je vyhodnocen výpadek proudu. V reakci na výpadek je do souboru `/etc/powerstatus` uložen znak F a poslán signál SIGPWR procesu init.

Proces init reaguje na příchod signálu SIGPWR zkontrolováním obsahu souboru `/etc/powerstatus`. Na základě hodnoty v tomto souboru je provedena jedna z následujících činností:

- F - Spustí skripty powerwait a powerfail
- O - Spustí skript powerokwait
- L - Spustí skript powerfailnow

V případě navrhovaného zařízení je použit pouze skript powerfail, ve kterém je zavolán příkaz halt. Ten předá procesu init signál pro přechod do runlevlu 0.

halt

Tento skript zajišťuje vypnutí systému. Je spouštěn jako poslední a zajišťuje vlastní vypnutí systému. U počítačů s ATX zdrojem zajistí přímo odpojení napájení. FOX Board není vybaven žádným elektronickým spínačem který by umožňoval odpojit napájení. Systém se tedy nevypne ale čeká na odpojení napájení FOX Boardu, externí reset, případně příkaz předaný přes sériovou konzoli. Externí reset ani sériová konzole nejsou v navrženém zařízení použity a k resetu by tedy mohlo dojít jen odpojením napájení. Toto chování je v případě navrhovaného zařízení krajně nevhodné. V případě výpadku vstupního napájení dostatečně dlouhého na to, aby zahájil vypínání systému, ale ne dost dlouhého pro úplné vybití zálohovacích kondenzátorů, by systém zůstal čekat na dostatečně dlouhý výpadek napájení.

Z tohoto důvodu byl skript upraven tak, že namísto pokusu o vypnutí systému v nekonečné smyčce, čeká na obnovení napájení a pokud k němu dojde, vyvolá softwarový reset. Pokud k obnovení vstupního napájení nedojde před vybitím zálohovacích kondenzátorů dojde k vypnutí napájení FOX Boardu. Vypnutí systému v tomto okamžiku je již bezpečné, protože v době vykonávání skriptu halt jsou již ukončeny všechny daemony a odmountovány všechny disky.

pesnet

Tento skript při spouštění systému zavede příkazem insmod ovladač sběrnice pesnet, používaný pro čtení dat ze sítě automatů Micropel a následně v adresáři /dev/-pesnet0/ vytvoří speciální soubory pro přístup k jednotlivým automatům v síti.

logger

Tento skript zajišťuje spuštění daemonu logger při spouštění systému a jeho ukončení při vypínání. Tuto činnost provádí přes volání nástroje start-stop-daemon. Použití tohoto programu značně zjednodušuje spouštění a vypínání daemonů. Při spouštění zkontroluje, zda již požadovaný daemon neběží a pokud ano tak ho znovu nespouští. Při vypínání najde příslušný daemon a zašle mu signál SIGTERM, kterým ho žádá o ukončení. Určitým problémem může být identifikace již běžícího daemonu při spouštění a ukončování. Pro základní rozpoznávání daemonu je využit jen název příslušného spustitelného souboru. V tomto případě však nastanou problémy pokud je spuštěno více instancí jednoho daemonu a start-stop-daemon se chová podobně jako příkaz killall. Pro jednoznačnou identifikaci je v navrhovaném řešení použit přepínač pidfile. Při použití tohoto přepínače se k identifikaci daemonu nepoužívá

jeho název, ale přímo PID uložené v souboru zadaném bezprostředně za tímto přepínačem. Aby bylo možné využít tuto funkci musel být program logger doplněn o schopnost vytvořit tento soubor.

4.3 Komunikace přes RS485

Standardní linuxové jádro neobsahuje ovladač pro RS485. Potřebná funkcionalita je však velmi podobná funkci běžného ovladače sériového portu, který je přítomen. V zásadě jsou potřeba vyřešit dva problémy. Zajistit přepnutí do RS485 módu, zajišťujícího správné přepínání budiče sběrnice a zajistit 9-ti bitovou komunikaci. Přepnutí do RS485 módu je možné zajistit jednoduchou modifikací hodnoty jediného konfiguračního registru (v ovladači implementovaného pomocí volání `ioctl`). Větší problém je zajištění 9-ti bitové komunikace.

Nejjednodušší přístup, nevyžadující další modifikace, spočívá ve využití parity. Při vysílání se postupuje tak, že se před vysláním každého byte nastaví sudá nebo lichá parita tak, aby měl 9-tý bit správnou hodnotu. Při příjmu se postupuje podobně. V prvním kroku je nutné nastavit, aby byly programu předávány i byty přijaté se špatnou paritou a pak je možné zjistit, podle správnosti parity, hodnotu 9-tého bitu. Tento přístup má však jednu zásadní nevýhodu. Při změně nastavení parity dochází k resetu hw vysílače, který vede k přerušení vysílání aktuálního znaku. Program by tedy musel zajistit, aby nebyl odeslán nový znak před dokončením předchozího, což by bylo z hlediska načasování velmi náročné.

Další možností, jak se vypořádat s 9-ti bitovým přenosem, je úprava původního ovladače. Zde je však již nutné provést poměrně rozsáhle úpravy původního ovladače. Navíc se tato cesta ukázala jako neschůdná, protože i v tomto případě nemůže program běžící v uživatelském prostoru dodržet rychlost odezvy vyžadovanou při komunikaci (odezva musí přijít v čase odpovídajícím třem znakům, to při rychlosti 57600 baudu odpovídá přibližně 0,6 ms).

Dosáhnout potřebné rychlosti odezvy je možné v zásadě dvěma způsoby. Buď použitím realtime jádra, nebo napsáním vlastního ovladače. V této práci byla zvolena cesta vlastního ovladače. Ten využívá obsluhu přerušení od příslušného portu, v kterém zajišťuje kromě přijmutí a odeslání jednotlivých znaků i základní vyhodnocení přijatého rámce a pokud to vyžaduje, tak i odeslání příslušné odpovědi. Tímto postupem je možné dosáhnout velmi krátké odezvy.

4.3.1 Ovladač

Jak již bylo rozebráno výše, SW pro komunikaci přes sběrnici RS485 byl realizován jako ovladač v podobě jaderného modulu. Původní pokus vytvořit ovladač pouhou

úpravou ovladače standardního UART se ukázal jako neschůdný. Úprava komunikace na 9-ti bitovou vyžadovala velké zásahy do struktury ovladače. Dalším problémem byla rychlost odezvy v případě že byla komunikace vyhodnocována v uživatelské aplikaci. Při nízkém zatížení systému komunikace probíhala i když ne příliš spolehlivě, při vyšším nefungovala vůbec. Oba tyto problémy byly odstraněny vytvořením vlastního ovladače.

Vlastní ovladač se dá rozdělit do dvou částí. První část se stará o vlastní komunikaci a celá je prováděna v obsluze přerušení. Druhá se stará o komunikaci s procesy běžícími v uživatelském prostoru.

Obsluha přerušení

Použité přerušení může být vyvoláno z několika důvodů. Procedura zavolaná po jeho vzniku tedy musí zjistit, z jakého důvodu přerušení vzniklo a rozhodnout se, zda ho zpracuje nebo předá zpracování dál. U vytvořeného ovladače přicházejí v úvahu dva důvody pro vznik přerušení. Prvním je připravenost UARTu k odeslání dalšího znaku. V tom případě ovladač zkontroluje buffer odchozích znaků a pokud není prázdný, tak z něj vyjme jeden znak a předá ho k odeslání. Druhou možností je přijetí znaku. Pokud má přijatý znak nastaven 9tý bit (jedná se o první znak packetu), je výstupní buffer vyprázdněn a je do něho přidán právě přijatý znak. V opačném případě je znak pouze přidán na konec bufferu. Po přijetí každého znaku je provedena kontrola, zda již není v bufferu uložen celý packet (vyhodnocení je provedeno na základě typu packetu uvedeném v jeho hlavičce a počtu přijatých znaků). Pokud není je obsluha přerušení ukončena, v opačném případě je provedena kontrola CRC a packet je předán k dalšímu zpracování. Naprostá většina packetu nemá pro náš systém význam a je tedy zahozena bez jakékoliv odpovědi. Výjimkou je packet, který předává token pro řízení sběrnice našemu zařízení. Pokud má naše zařízení požadavek na čtení či zápis, odešle tento požadavek na sběrnici. Po přijetí odpovědi tuto odpověď předá do procesu v uživatelském prostoru a probudí proces, který na tuto operaci čekal. Nakonec musí vždy předat řízení sběrnice další stanici. Kromě výše popsané funkce musí ovladač řešit množství problémů vznikajících při chybách v komunikaci, které zde kvůli své složitosti nejsou rozebírány.

Komunikace uživatelského procesu s ovladačem

Komunikace s procesem běžícím v uživatelském prostoru probíhá přes standardní volání jádra OS funkcemi: open, release, seek, read a write. Funkce open a release jsou volány při otevření/zavření příslušného souboru a v případě tohoto ovladače provádí pouze přípravu některých dále používaných struktur. Funkce seek slouží k nastavení místa v souboru. To odpovídá adrese v paměti příslušného automatu se

kterým chceme dále pracovat. Funkce read a write jsou jediné, které přímo vyvolávají komunikaci. Jejich funkce je velmi podobná. Na začátku zamknou mutex, který zajišťuje, že bude v jeden okamžik zpracováván pouze jeden požadavek. Poté na připravené místo v paměti (které je kontrolováno z obsluhy přerušení) uloží adresu automatu, s kterým má proběhnout komunikace, typ operace (čtení/zápis), případně data která mají být zapsána. Nakonec pomocí atomické operace označí požadavek za připravený a uspí se. Po probuzení (vyvolaném v obsluze přerušení) zkontroluje výsledek operace, zkopíruje příslušná data do uživatelského procesu a uvolní zámek.

4.4 Paměťové médium

Jak je popsáno dříve, na desce FOX Board jsou k dispozici dvě uložistě dat. Přímo na desce je umístěna 8MB flash paměť. Pokud je její kapacita nedostatečná, je možné použít microSD kartu. Integrovaná paměť je pro ukládání dat příliš malá a je tedy nutné použít microSD kartu. Dnes dostupné microSD karty jsou k dispozici v kapacitách od 2GB a poskytují tedy dostatek prostoru pro uložení systému i znamenaných dat. Jejich použití však přináší určité problémy. Technologie FLASH umožňuje jen značně omezený počet přepisů. Ten bývá většinou udáván v rozmezí 10 000 - 100 000. V běžných SD kartách je využívána technologie MLC, která umožňuje v jedné paměťové buňce uchovat více bitů. Tímto postupem je dosaženo zvýšení kapacity bez zvětšení plochy vlastní paměti a tím i snížení ceny. Toto zvýšení kapacity je však vykoupeno sníženou trvanlivostí paměti. Dá se tedy předpokládat, že FLASH paměti použité v běžných SD kartách budou mít trvanlivost blížící se spodní hranici.

Operační systém GNU/Linux obsahuje ovladače pro mnoho souborových systémů určených speciálně pro použití s FLASH paměti, které organizují zápis do paměti tak, aby i v případě opakovaného zápisu do stejného místa v jednom souboru docházelo k zápisu na různá místa v paměti. V případě častého zápisu malého množství dat do jednoho souboru je tímto postupem zvýšena životnost paměti o několik řádů. Všechny tyto souborové systémy vyžadují přímý přístup k paměti.

SD karty jsou nejčastěji využívány se souborovým systémem FAT, který vůbec nepočítá s použitím na paměťovém mediu s omezeným množstvím přepisů. Aby byla prodloužena životnost karet při tomto použití, provádí řadič mapování logických bloků na bloky fyzické (Wear Leveling). Pokud bylo do některého bloku provedeno velké množství zápisů nebo pokud došlo k chybě při čtení, může tento blok přesunout na místo jiného bloku (za tímto účelem obsahuje karta rezervní bloky nad svou nominální kapacitu). Pro detekci a opravu chyb obsahuje každý fyzický blok

samoopravný kód. Pokud tedy dojde k selhání omezeného množství bitů je možné blok přesunout a přemapovat bez ztráty dat.

Celý výše popsaný mechanismus sice zvyšuje životnost karty při typickém využití karty, bohužel však znemožňuje použití souborových systémů určených pro FLASH paměti. Mapováním logických bloků na fyzické totiž probíhá zcela transparentně bez možnosti ho jakkoliv vypnout nebo obejít a neumožňuje tedy přímý přístup k paměti. Pokud by byl v tomto případě použit souborový systém, který se snaží o rozprostření zápisu stejně jako řadič uvnitř karty, mohlo by docházet k těžko předvídatelným interakcím.

Jako nejlepší varianta se tedy jeví použít na kartě běžný souborový systém a omezit množství zápisu. Za tímto účelem byl zvolen souborový systém ext2. Jehož hlavní výhodou v tomto ohledu je absence journalování, které by vedlo k zbytečným přepisům. Dalšího omezení je dosaženo použitím parametru `noatime` při mountování oddílu. Ten zabrání zapisování času posledního přístupu k souboru. K omezení zápisů také přispěje změna parametrů ovlivňujících práci s cache. Použito je následující nastavení:

```
echo 60000 > /proc/sys/vm/dirty_writeback_centisecs
echo 60000 > /proc/sys/vm/dirty_expire_centisecs
```

Tyto hodnoty prodlužují dobu po kterou může být odložen zápis dat z cache na paměťové medium. Poslední postup je implementován na aplikační vrstvě. Program pro záznam dat do databáze neukládá naměřená data do databáze okamžitě, místo toho je shromažďuje a zapíše vždy všechny naráz v určitém intervalu.

Použití souborového systému bez journalování přináší problém s jeho možným poškozením při nenadálém vypnutí⁷. K tomu může dojít buď úmyslným vypnutím napájení celého rozvaděče nebo výpadkem sítě. V případě vypnutí by bylo možné situaci řešit trojpolohovým vypínačem nebo časovým relé, které by daly operačnímu systému čas na ukončení práce. V případě nepředvídaného výpadku je jedinou možností použití UPS.

Pro zvýšení spolehlivosti systému je také vhodné namísto běžné SD karty použít kartu průmyslovou. Tyto karty bývají vyráběny SLC technologií a mají tedy podstatně delší životnost. Například karty společnosti STEC umožňují 100 000 přepisů jedné buňky, 2 000 000 v kombinaci s wear-levelingem[12]. Kromě toho průmyslové karty dosahují větší spolehlivosti a odolnosti.

⁷Při testování se i když v menší míře, projeví problémy i při použití journalovacího souborového systému. Pravděpodobně jsou způsobeny tím, že `erase block` je podstatně větší než jedna stránka, takže přerušovaný zápis do jednoho místa může způsobit poškození dat na místě, z pohledu souborového systému, naprosto nesouvisejícím. Možná je také interakce s interním wear-levelingem.

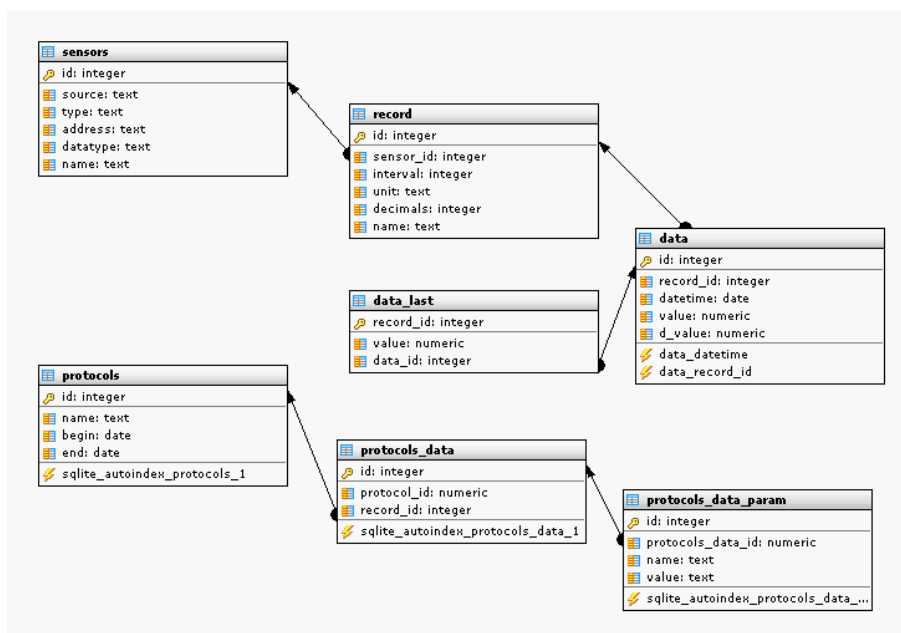
Z výše uvedených důvodů byl v navrhovaném systému použit souborový systém ext2. Problémy s výpadkem napájení jsou vyřešeny krátkodobě zálohovaným zdrojem, který dá systému dostatek času pro ukončení činnosti v případě nenadálého výpadku proudu.

4.4.1 Kapacita

Důležitým parametrem je kapacita paměťového média. Vzhledem k tomu že naměřená data jsou ukládány do databázového souboru, je poměrně složité předem vypočítat kolik vzorků bude možné uložit. Tato hodnota byla tedy určena experimentálně. Databáze obsahující 30 851 315 vzorků má velikost 1.7GB a na uložení jednoho vzorku je tedy potřeba přibližně 55B. Jeden den používání sterilizačního přístroje, který vygeneruje 57600 vzorků, spotřebuje 3.16MB prostoru na paměťové kartě. Na 2GB kartě zbývá po instalaci systému přibližně 800MB prostoru pro uživatelská data. Tento prostor stačí pro cca 250 dní záznamu. Pokud je potřeba archivovat více dat stačí použít 4GB kartu. Ta bude stačit na přibližně 875 dní. Záznam lyofilizačního procesu generuje pouze 8640 vzorků za den a při použití 2GB karty je možné archivovat záznam za 1600 dní.

4.5 Databazový model

Pro uložení nastavení parametrů záznamu i vlastní zaznamenaná data je použita jedna databáze v souboru umístěném v adresáři `/var/db/`. Pro účely komunikace přes GSM je vytvořena druhá databáze v dalším souboru umístěném ve stejném adresáři, ta však bude popsána až v kapitole zabývající se GSM komunikací. E-R diagram navržené databáze je na obrázku 4.5. V začátcích vývoje zařízení byla používána v té době aktuální distribuce Debian Lenny a s ní distribuovaná verze SQLite 2.8.17, která nepodporovala cizí klíče. Funkcionalita cizích klíčů tedy musela být ručně doplněna pro jednotlivé tabulky za pomoci triggeru. Později byl vývoj převeden na nově vydanou verzi Debianu Squeeze, která přinesla novější verzi SQLite, která již podporovala cizí klíče a bylo tedy možné nahradit velké množství triggeru cizími klíči a tím celý návrh výrazně zpřehlednit. Vzhledem k poměrně nízkému výkonu použitého mikroprocesoru, malé paměti RAM a pomalému přístupu k paměti FLASH bylo nutné vytvořit co nejjednodušší návrh a vyhnout se nákladným operacím jako je JOIN. Navržené tabulky budou popsány dále.



Obr. 4.5: Databázový systém modelu.

Tabulka sensors

Tato tabulka obsahuje informace o dostupných senzorech. Jako primární klíč je použit INT s funkcí AUTOINCREMENT. V sloupci source je uložena cesta k souboru, ze kterého jsou získávány naměřené hodnoty. Typ souboru z kterého jsou hodnoty načítána je dán sloupcem type. Ta může nabývat hodnot: binary, text, pesnet. Z binárních souborů je přečten počet bytů odpovídající datovému typu přímo do proměnné. U textových souborů je načten řetězec, který je před uložením převeden na číslo. Typ souboru pesnet je v podstatě aliasem pro typ binary, liší se jen ve způsobu zápisu adresy. Ve sloupci adresa je uložena pozice v souboru, ze které bude hodnota načtena. U binárních souboru je proveden seek na tuto adresu a následně přečten počet bytů odpovídající datovému typu. U textových souborů je přečten počet bytů odpovídající adrese a následně je načten řetězec zakončený nečíselným znakem (vše kromě znaku plus, minus, číslic a tečky). Tento řetězec je následně převeden na číslo. Pokud je zadán typ pesnet je přístup k souboru stejný jako u binárního, jen syntaxe sloupce adresa je rozdílná, neobsahuje přímo číslo, ale definici proměnné v syntaxi používané v jazyce SIMPLE. Sloupec datatype určuje datový typ příslušné veličiny, může nabývat hodnot:

- INT8 Pro 8-bitovou hodnotu se znaménkem
- INT16 Pro 16-bitovou hodnotu se znaménkem
- INT32 Pro 32-bitovou hodnotu se znaménkem
- UINT8 Pro 8-bitovou hodnotu bez znaménka
- UINT16 Pro 16-bitovou hodnotu bez znaménka
- UINT32 Pro 32-bitovou hodnotu bez znaménka
- FLOAT Pro 32-bitovou číslo s plovoucí desetinou čárkou dle IEEE754

Sloupec name obsahuje pouze textový popis senzoru použitý pro jednoduchou identifikaci příslušného senzoru.

Tabulka records

Tato tabulka obsahuje nastavení, týkající se parametrů jednotlivých záznamů. Cizí klíč sensor_id odkazuje na sensor ze kterého má být pořizován záznam. Sloupec interval udává vzorkovací frekvenci příslušného záznamu v milisekundách. Sloupec unit obsahuje jméno jednotky (°C, kPa apod.) příslušné veličiny. To je Využito při tvorbě tabulek a grafů. Sloupec decimals má význam u veličin, které jsou čteny jako celočíselné, ale znázorňují zlomky nebo násobky základní jednotky. Například teplota je v použitých automatech reprezentována celočíselným datovým typem v desetínách °C. Pro získání skutečné hodnoty ve °C je tedy potřeba přečtenou hodnotu vydělit deseti a atribut decimals tedy bude mít hodnotu 1. Sloupec name obsahuje textový popis záznamu použitý pro jednoduchou identifikaci záznamu.

Tabulka data

Do tabulky data jsou ukládána vlastní naměřená data. Cizí klíč record_id určuje ke kterému záznamu naměřená položka přísluší. Sloupec datetime obsahuje čas ve který byl záznam pořízen ve formátu unixového času vynásobeného tisícem (místo sekund od 1.1. 1970 jsou tedy použity milisekundy). Sloupec value obsahuje vlastní přečtenou hodnotu. Sloupec d_value obsahuje změnu hodnoty oproti minulému měření a je použit primárně pro automatické vytváření protokolů. Díky němu je jednoduše možné například určit okamžik ve kterém došlo k rychlé změně měřené veličiny. Hodnoty jsou do tohoto sloupce nastavovány triggerem, který je spouštěn po přidání nového záznamu.

Tabulka data_last

Tato tabulka neobsahuje žádná unikátní data. Pomocí triggeru spouštěného přidáním nového záznamu do tabulky data, jsou v této tabulce aktualizovány hodnoty naměřené na jednotlivých senzorech. Tabulka tedy obsahuje poslední naměřenou hodnotu na každém čidle. Stejného výsledku by bylo možné dosáhnout i dotazem

nad tabulkou data, tento přístup je však z hlediska rychlosti výhodnější. Vzhledem k tomu, že poslední naměřená hodnota je používána pro výpočet d_value při každém novém vložení hodnoty, je rychlost tohoto dotazu zásadní.

Tabulka protocols

V této tabulce jsou uloženy informace o začátku a konci všech protokolů, vytvořených z naměřených dat. Začátek i konec je uložen ve formátu unixového času ⁸ Dále obsahuje sloupec name který slouží k uchování názvu protokolu.

Tabulka protocols_data

Tato tabulka určuje, které z měřených veličin budou použity jako datové řady pro tvorbu protokolu. Sloupec protocol_id slouží jako cizí klíč a vytváří vazbu s tabulkou protocols. Dalším cizím klíčem je atribut record_id, který vytváří vazbu s tabulkou record.

Tabulka protocols_param

V této tabulce jsou uloženy další parametry jednotlivých datových řad vytvořených z naměřených dat, jako je nastavení rozsahu hodnot (použité pro určení měřítka os grafu), barva veličiny v grafu, maximální a minimální přípustná hodnota (hodnoty mimo toto rozmezí jsou v tabulce barevně zvýrazněny) apod. Tyto parametry jsou k jednotlivým datovým řadám přiřazeny pomocí cizího klíče protocols_data_id. Sloupec name určuje název parametru a může nabývat hodnot: MINVALUE, MAXVALUE, COLOR, MINVALID, MAXVALID. Sloupec value pak obsahuje hodnotu příslušného parametru. Všechny tyto parametry by mohli být přímo v tabulce protocols_data, jejich uložení do zvláštní tabulky však přináší větší flexibilitu při přidávání nových možností. Pokud by byly parametry přímo součástí tabulky protocols_data, bylo by nutné pro přidání nových parametrů upravit návrh této tabulky. V případě, že jsou však uloženy výše popsaným způsobem ve zvláštní tabulce, je možné přidávat nové parametry bez úpravy návrhu databáze.

4.6 Daemon logger

Úkolem tohoto daemonu je načítání dat ze speciálních souborů (poskytovaných například ovladačem sběrnice Pesnet) a jejich ukládání do databáze. Program byl vytvořen v C++ a jeho zdrojové soubory i makefile je na příloženém CD.

⁸V operačním systému UNIX a standartu POSIX je čas měřen jako počet sekund od 1.1.1970 GMT.

Celý program je rozdělen do dvou částí. Soubory `reclist.c` a `reclist.h` obsahují třídu `RecList`, která zajišťuje vlastní čtení dat a jejich zápis do databáze. Soubor `logger.c` obsahuje kód starající se o inicializaci programu, zpracování signálů a běh programu jako daemonu.

První věc, kterou program po svém spuštění dělá, je zpracování parametrů předaných přes příkazový řádek. K tomu je využívána funkce `getopt`, která je součástí `glibc`. Vzhledem k tomu, že většina konfigurace je načítána z databáze, je těchto parametrů poměrně málo:

-p cesta

Určuje cestu k použitému databasovému souboru. Pokud není žádný určen je použit soubor `/var/db/database.sqlite`

-d

Pokud je použita tato volba tak se program spustí jako daemon.

-c t

Tato volba určuje časový interval, po kterém jsou data z bufferu zapsána na disk.

Dalším krokem po zpracování parametrů je, pokud byl použit příslušný přepínač, převedení běhu programu na pozadí. Aby mohl program běžet na pozadí je potřeba provést několik kroků. Prvním z nich je odpojení se od svého rodiče. To je provedeno tak, že program zavolá funkci `fork`, přičemž původní proces skončí a osiřelý potomek je adoptován procesem `init`. Poté je pomocí příkazu `setsid()` vytvořena nová *session*⁹ a nová skupina procesů¹⁰. Tím je zajištěno, že se daemon stane součástí nové skupiny procesů a nebude tak dostávat nežádoucí signály. Následně nastaví svůj pracovní adresář na `/`, aby nebránil případnému odmontování souborového systému. Aby byl nezávislý na terminálu (případně na jiném otevřeném souboru) musí uzavřít všechny otevřené soubory. Na závěr jsou standartní vstup, výstup a chybový výstup přesměrovány do souboru `/dev/null` a veškeré chybové výstupy jsou namísto toho pomocí příkazu `syslog` zaznamenávány do systémového logu.

Ať již běží program na pozadí jako daemon, nebo zůstal na popředí, musí si zaregistrovat obsluhu signálů. V případě programu `logger` je zvláštním způsobem ošetřen pouze signál `TERM` a `INT`. Oba tyto signály zavolají funkci `sigTermHandler`. Ta změní obsah globální proměnné, která je testována v dalších částech programu.

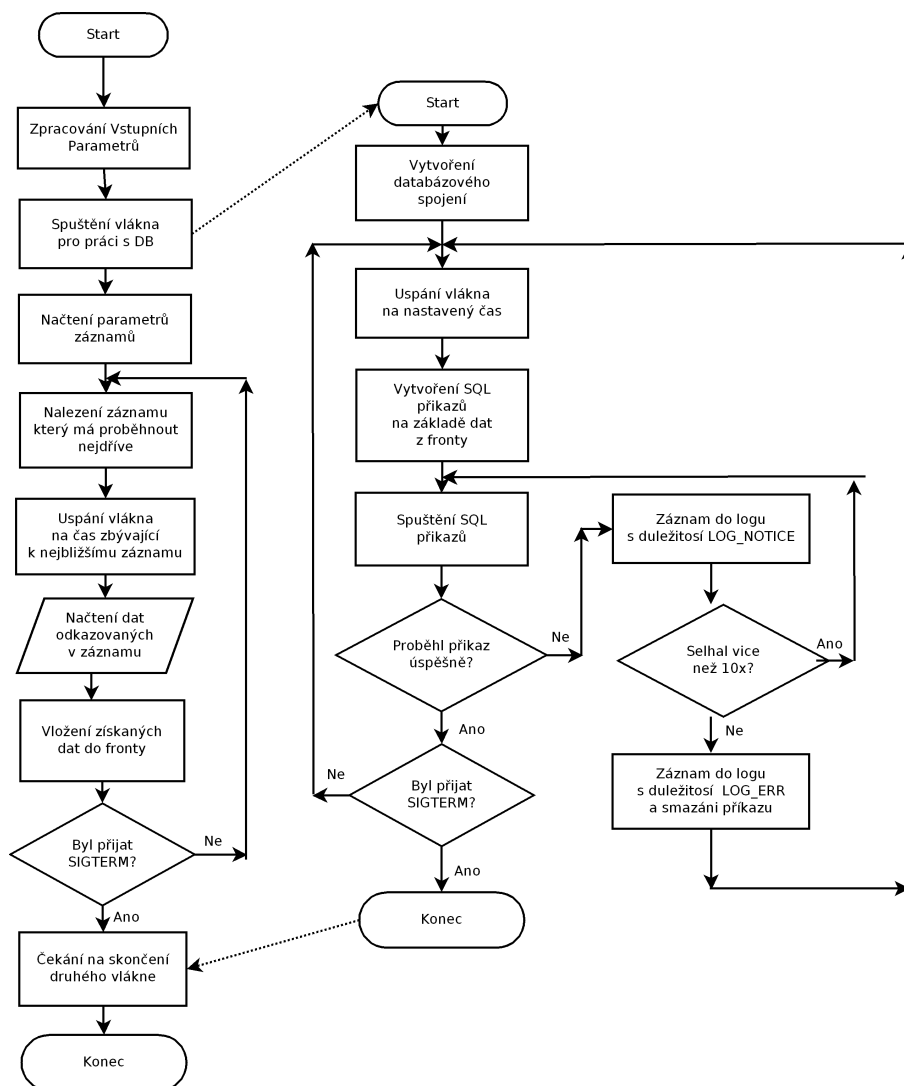
Poté již následuje vytvoření objektu `RecList`. Konstruktor tohoto objektu si načte konfiguraci záznamu dat z databáze a na základě těchto informací naplní vnitřní seznam (jedná se o objekt `std::map`, který jako klíč používá `id` z tabulky `records`) položkami typu `RecItem`. Každá tato položka představuje jednu veličinu, kterou je potřeba zaznamenávat a obsahuje parametry záznamu (zdrojový soubor, adresu,

⁹Session je skupina skupin procesů. Každá session může mít přidělen jeden řídicí terminál

¹⁰Skupina procesů (process group) je množina procesů, kterým lze posílat signály jako celku

vzorkovací frekvenci a čas posledního záznamu). Tím je dokončena inicializace programu a je možné přejít k vlastnímu záznamu dat.

Celý záznam dat běží ve dvou vláknech vytvořených za pomoci knihovny pthread. První vlákno obstarává čtení dat v nastavených intervalech, ale nezapisuje je přímo do databáze (jedná se o poměrně pomalý proces a mohlo by kvůli němu dojít k pozdnímu přečtení následujícího vzorku), ale místo toho jsou ukládány do fronty. Druhé vlákno v nastavených intervalech přečte data z fronty a uloží je do databáze. Synchronizace přístupu k frontě je řešena mutexem. Zjednodušená funkce programu je znázorněna na 4.6.



Obr. 4.6: Vývojový diagram programu logger.

Čtení dat ze speciálních souborů probíhá ve smyčce, která je ukončena jen v případě přijetí signálu TERM. Na začátku této smyčky je pro každou položku v

seznamu zaznamenávaných veličin určen čas, za který má být přečtena. Nejnižší čas je poté použit jako parametr funkce `usleep`. Tím je zajištěno uspání vlákna do doby, kdy má dojít k záznamu. Po probuzení je zjištěna skutečná délka spánku¹¹ a na základě tohoto údaje je provedeno přečtení dat z položek seznamu u kterých vypršel čas. Čtení jednotlivých údajů probíhá pomocí synchroních operací. Jednotlivé vzorky, i pokud mají stejnou vzorkovací frekvenci, tedy nejsou pořízeny současně¹², ale je mezi nimi určitá co?. Při použití s ovladačem `PesNet` je tato prodleva menší jak 10ms. Pokud je to vyžadováno je u přečtených dat provedena konverze endianness. Následně je hodnota použita na vytvoření sql příkazu, sloužícího k uložení této hodnoty do databáze. Tento příkaz však není ihned vykonán, ale je přidán do fronty (přístup do této fronty je chráněn mutexem). Poslední akcí po přečtení každé položky je naplánování příštího čtení.

Zápis dat do databáze je zajišťován druhým, paralelně běžícím vláknem. Před vlastním zápisem se vždy vlákno uspí, na dobu určenou parametrem `-c`. Poté jsou z fronty sql příkazů postupně odebírány (přístup je opět synchronizován mutexem) jednotlivé položky a je z nich připraven jeden dlouhý příkaz, který je následně proveden. Vzhledem k vlastnostem sqlite databáze (omezená možnost konkurenčního přístupu) je možné, že příkaz selže. V tom případě se program pokusí příkaz zopakovat. Pokud opakovaně neuspěje, jsou data zahozena a je odeslán příslušný záznam do logu. Celý tento postup je opakován v cyklu, dokud není přijmut signál `TERM`.

4.7 Webové rozhraní

Vlastní webové rozhraní je napsáno v jazyce PHP5, který generuje stránky ve značkovacím jazyce `xhtml`. Vzhled stránek je na straně klienta upraven pomocí kaskádových stylů (`css`). Zdrojový kód webového rozhraní se dá rozdělit na několik základních částí.

Funkce `SaveToCache` má dva parametry. Prvním z nich je `md5` hash. Druhým jsou data určená k uložení. Funkce po svém zavolání vloží do tabulky řádek s příslušným hashem a daty. Zároveň zkontroluje počet záznamů v cache a pokud je větší než zadaná hodnota smaže nejstarší záznam.

¹¹Specifikace funkcí `sleep` a `usleep` zajišťují, že se proces neprobudí před uplynutím zadaného intervalu (pokud nepřijde asynchroní signál), nemohou však zaručit, že se proces probudí přesně v zadaném okamžiku. Velikost tohoto zpoždění je závislá především na aktuální zátěži systému a parametrech jádra.

¹²Při použití asynchronních operací by se situace příliš nezlepšila, protože ovladač sběrnice `PesNet` dokáže současně obsluhovat pouze jeden požadavek. Zlepšení by bylo patrné pouze v případě, že by byla data čtena ze speciálního souboru, jehož ovladač je schopný zpracovávat paralelní dotazy, nebo při čtení z nezávislých zařízení

4.7.1 Knihovna pro práci s daty

Pro práci s naměřenými daty byly vytvořeny dvě třídy `dataserie` a `datatable`. Zdrojové kódy obou tříd jsou umístěny ve stejnojmenných souborech. Obě třídy využívají cachování výstupů, za účelem zrychlení odezvy.

Třída `dataserie`

Tato třída zajišťuje práci s jednou měřenou veličinou v zadaném časovém intervalu. Její konstruktor má jako parametr číslo záznamu v tabulce `record`, začátek a konec časového intervalu. Všechny tyto parametry jsou použity k vytvoření mdř hashe přístupného přes metodu `getHash()`, která je dále používána pro cachování výsledků. Třída obsahuje parametry získané z tabulky `record` jako je vzorkovací frekvence a jednotka měřené veličiny. Dále poskytuje informaci o maximální a minimální hodnotě měřené veličiny v zadaném intervalu. Hlavní funkcí třídy je poskytnutí přístupu k jednotlivým naměřeným hodnotám, zajišťovaném metodou `getValue`. Ta má jako jediný parametr čas měření a vrací hodnotu naměřenou v tento okamžik. Pokud přesně v tento okamžik nebylo provedeno měření, vrátí lineární aproximaci hodnoty získanou ze dvou nejbližších vzorků. Pokud je potřeba přistupovat pouze k skutečně naměřeným hodnotám, je možné procházet přímo pole `data`. Jedná se o asociativní pole, kde klíč je čas pořízení vzorku a hodnota obsahuje velikost naměřené veličiny. Toto pole není naplněno hodnotami hned v konstruktoru třídy, ale až ve chvíli prvního přístupu k němu. Tohoto chování je docíleno použitím magických metod `__get()` a `__set()`, které jsou volány pokaždé, kdy dojde k přístupu k nedefinované nebo nepřístupné vlastnosti. Účelem tohoto chování je zrychlení programu v případě, kdy byla vytvořena instance objektu `dataserie` jen za účelem získání informací jako je maximální hodnota veličiny v určitém časovém intervalu nebo získání názvu jednotky.

Třída `datatable`

Tato třída slouží k vytvoření tabulky nebo grafu složeného z několika měřených veličin v určeném časovém intervalu. Prvním parametrem konstruktoru je pole `id`, které odkazuje na záznamy v tabulce `records`. Na základě těchto `id` jsou vytvořeny instance třídy `dataserie`, které jsou dále používány pro přístup k naměřeným hodnotám. Dalšími parametry je začátek a konec časového intervalu, se kterým se má pracovat. Hlavní funkcí této třídy je tvorba tabulky hodnot a grafů z naměřených hodnot.

K zobrazení tabulky hodnot slouží metoda `showTable`. Tato funkce má jako parametr formátovací řetězec určující formát zobrazení času u jednotlivých vzorků.

Další nepovinný parametr může nastavit název grafu použitý v jeho hlavičce. Na začátku funkce je vypočítán md5 hash, skládající se z hashu všech použitých objektů dataserie, parametrů metody showTable a názvu funkce. Výsledný hash je použit jako parametr funkce GetFromCache, která se pokusí získat z cache dříve uložený výsledek funkce. Pokud je úspěšná jsou vrácena data získaná z cache. V opačném případě musí být celá tabulka sestavena od začátku. Hlavička tabulky je vytvořena na základě informací získaných z tabulky record pomocí třídy dataserie (jméno záznamu a název měřené veličiny). Celá hlavička je uzavřena do tagu thead, ten zajišťuje, že při tisku dlouhé tabulky je hlavička vytištěna na začátku každé stránky ¹³. Po hlavičce již řádky obsahující naměřená data. Každý řádek obsahuje hodnoty všech veličin v jednom čase. Tyto časy jsou určeny dle časů, ve kterých byly naměřeny hodnoty první datové řady. Hodnoty dalších veličin jsou získány metodou getValue. Z toho mimo jiného vyplývá, že pokud nejsou jednotlivé veličiny vzorkovány ve stejný okamžik je použita lineární aproximace. Celá tabulka je po sestavení uložena funkcí SaveToCache do cache pro další použití.

Další možností výstupu je metoda showGraph. Její funkce je podobná jako u metody showTable, včetně cachování výsledků. Tato metoda vytváří ze zadaných dat graf ve formátu SVG. Tento formát byl zvolen zejména kvůli tomu, že se jedná o formát vektorový, takže je možné jeden obrázek použít jak pro zobrazení na obrazovce, tak pro tisk. V případě použití rastrového obrázku by bylo pro kvalitní tisk potřeba použít obrázek o vysokém rozlišení, jehož generování by bylo vzhledem k omezeným paměťovým možnostem problematické. Díky tomu, že je pro tvorbu vlastního webového rozhraní použita technologie xhtml, vycházející z xml, je možné vložit vytvořený obrázek přímo do kódu stránky. V případě, že je použit starší prohlížeč, který neobsahuje podporu pro zobrazení obrázků ve formátu SVG, je prohlížeči automaticky předložen obrázek ve formátu PNG. Ten je z původního SVG vygenerován pomocí utility rsvg-convert. Vzhledem k omezeným možnostem FOX Boardu je obrázek v nízkém rozlišení (1024x586), které dostačuje pro zobrazení na monitoru, ale je zcela nevhovující pro tisk.

4.7.2 Knihovna pro práci s cache

Při tvorbě tabulek a grafů z velkého množství dat se objevily problémy s rychlostí. Pro zrychlení odezvy jsou výsledky ukládány do cache, ve které mohou být v případě opakování stejného dotazu rychle nalezeny, bez nutnosti znovu sestavovat celou tabulku či graf.

¹³Chování tohoto tagu se mezi jednotlivými prohlížeči značně liší. Ve Firefoxu jsou hlavičky tisknuty bez problému na všech stránkách, stejně jako v Internet Exploreru, naopak prohlížeč Google Chrome tiskne hlavičku jen na první stránce.

Vlastní cache je tvořena jednoduchou tabulkou se třemi sloupci. První ze sloupců slouží jako index a obsahuje md5 hash. Tento hash musí být vytvořen ze všech parametrů ovlivňujících výsledek funkce, aby bylo zajištěno že funkce se stejným hashem mají vždy stejný výsledek (pravděpodobnost kolize md5 hashe je zanedbána). Druhý sloupec obsahuje data vrácená funkcí při předchozím volání. Poslední sloupec obsahuje datum vložení. Pro práci s touto cache jsou k dispozici dvě funkce: `GetFromCache` a `SaveToCache`.

Funkce `GetFromCache` přijímá jediný parametr a to md5 hash. Pokud se v cache nachází záznam s příslušnou hodnotou je vrácen řetězec obsahující data uložená s tímto hashem, v opačném případě vrátí hodnotu `false`.

Funkce `SaveToCache` má dva parametry. Prvním z nich je md5 hash. Druhým jsou data určená k uložení. Funkce po svém zavolání vloží do tabulky řádek s příslušným hashem a daty. Zároveň zkontroluje počet záznamů v cache a pokud je větší než zadaná hodnota smaže nejstarší záznam.

4.7.3 Vlastní stránky

Jednotlivé stránky mají jednotný vzhled definovaný společnými css soubory. Ty zároveň zajišťují rozdílný vzhled stránek při zobrazení na monitoru a při tisku. Jednotlivé stránky jsou přístupné pomocí vlevo umístěného menu. Na horním okraji je umístěn stavový řádek obsahující název zařízení, reálný čas na zařízení a jméno uživatele s možností odhlásit se. Všechny stránky hned na začátku provádějí kontrolu session a v případě, že není platná přesměrují uživatele na přihlašovací stránku.

Přihlášení uživatele

Zdrojový kód této stránky je v souboru `login.php`. Stránka zajišťuje autentifikaci uživatele. Zadané jméno a heslo je kontrolováno proti údajům uloženým v tabulce `users`. Pokud údaje odpovídají, je vytvořena nová session a uživatel je přesměrován na hlavní stránku.

Hlavní stránka

Zdrojový kód této stránky je v souboru `main.php`. Na tuto stránku je uživatel přesměrován po přihlášení. Je zde umístěna tabulka všech čidel a jejich posledních hodnot a graf zobrazující průběh všech veličin za posledních 30 min. Časový interval grafu je možné upravit zadáním konkrétních hodnot do připravených editboxů.

Nastavení uživatelů

Zdrojový kód této stránky je v souboru users.php. Na tuto stránku může uživatel vstoupit pouze pokud má administrátorská práva. Je zde možné vytvářet a mazat jednotlivé uživatele, měnit jejich oprávnění a měnit jejich hesla.

Nastavení snímačů

Zdrojový kód této stránky je v souboru sensors.php. Na tuto stránku může uživatel vstoupit pouze pokud má administrátorská práva. Je zde možné upravovat a vytvářet nová čidla. Stránka zajišťuje editaci tabulky sensors. Pro každý sensor je možné nastavit cestu odkud mají být čtena data a datový typ.

Nastavení záznamu

Zdrojový kód této stránky je v souboru records.php. Na tuto stránku může uživatel vstoupit pouze pokud má administrátorská práva. Je zde možné upravovat parametry záznamu z jednotlivých čidel. Hlavním parametrem je vzorkovací frekvence. Další parametry neovlivňují přímo záznam, ale spíše způsob jeho zobrazení. Popisují barvu příslušné veličiny v grafu a podobně.

Vytvoření protokolu

Zdrojový kód této stránky je v souboru records.php. Na tuto stránku může vstoupit každý přihlášený uživatel. Je na ní možné vybrat časový interval a požadované veličiny a vytvořit z nich protokol. Ten obsahuje jednak graf a také výpis všech hodnot v podobě tabulky. Takto vytvořený protokol může být uložen do archívu nebo vytisknut.

Prohlížení protokolu

Zdrojový kód této stránky je v souboru records.php. Na tuto stránku může vstoupit každý přihlášený uživatel. Na této stránce je seznam všech ručně i automaticky vytvořených protokolů. Tyto protokoly je možné přejmenovat, smazat a prohlížet. Vzhledem k tomu že jsou použity dva různé CSS styly pro zobrazení na monitoru a pro tisk, je možné tuto stránku přímo vytisknout. Součástí jako je menu a seznam protokolů jsou při tisku automaticky odstraněny.

4.8 Vzdálený přístup

Naměřená i aktuální data jsou primárně přístupná přes podnikovou síť. V případě potřeby je možné vhodným nastavením hraničního routeru, zpřístupnit webové rozhraní i z internetu. V některých případech není intranet k dispozici nebo není intranet připojen k internetu a pro práci s webovým rozhraním je připojen počítač přímo k FOX Boardu kříženým kabelem. Pokud je i přesto potřeba zajistit vzdálený přístup k zařízení je potřeba zvolit alternativní cestu. Vzhledem k poměrně dobrému pokrytí a relativně nízkým cenám se nabízí využití připojení přes síť GSM. Aby bylo možné připojit zařízení, je potřeba připojit k zařízení GSM modem.

4.8.1 Připojení GSM modemu

V navrhovaném zařízení byl využit modem MC75i, připojený přes USB port. Připojení modemu přes USB port je rozpoznáno daemonem `udev`¹⁴, který vytvoří soubor `/dev/ttyACM0`. Pokud by byl modem připojen přes sériové rozhraní, byl by přístupný přes příslušný speciální soubor (`/dev/ttyS4`). Obě možnosti připojení jsou rovnocenné a jediný rozdíl je v nutnosti odkazovat v konfiguračních souborech na jiný speciální soubor. Aby nebylo nutné, v případě změny připojení, modifikovat několik konfiguračních souborů, je ve všech použit odkaz na `/dev/modem`. Tento soubor je symbolickým odkazem na soubor odpovídající aktuálnímu způsobu připojení.

4.8.2 Komunikace pomocí SMS

SMS je možné použít pro jednoduchou konfiguraci a zjištění stavu zařízení. Hlavní výhodou je jednoduché ovládání (stačí kterýkoli mobilní telefon), relativně nízká cena a nenáročnost na kvalitu sítě. Přenos většího množství informací by však byl velmi nákladný a také rychlost odezvy není příliš velká.

Pro práci se SMS byl vytvořen databázový `/var/db/sms.sqlite`. Ten obsahuje 3 tabulky. Tabulka `received` obsahuje přijaté SMS. Tabulka `tosend` obsahuje SMS určené k odeslání. Tabulka `smsusers` obsahuje telefonní čísla a práva, která k těmto číslům náleží. Veškerá práce se SMS je obsloužena dvěma PHP skripty, které jsou spouštěny pomocí daemonu `cron` každé tři minuty.

¹⁴`Udev` je linuxový správce zařízení. Běží jako daemon na pozadí a přijímá zprávy o inicializaci zařízení vysílané systémem `uevents`. V reakci na tyto zprávy pak provádí přednastavené úkoly, jako je vytváření speciálních souborů v adresáři `/dev/`.

skript readsms

Tento PHP skript každé tři minuty spustí program gammu, s parametrem geteachsms. Tím získá výpis všech SMS uložených na simkartě ve formátu popsaném v manuálu programu gammu[16]. Z takto získaného výpisu jsou za pomoci regulárního výrazu získána těla jednotlivých zpráv a čísla odesílatelů. Takto získané zprávy jsou uloženy do tabulky tosend.

Po přečtení všech SMS projde tabulku tosend a pokud obsahuje nějaké zprávy k odeslání, tak je připraví ve tvaru vhodném pro gammu a následně je pomocí tohoto programu odešle. Odeslané zprávy jsou z tabulky tosend smazány. Pokud z nějakého důvodu odeslání zprávy selže, zůstane v tabulce a pokus o její odeslání se opakuje při dalším spuštění skriptu.

skript execsms

Tento skript zajistí vyřízení jednotlivých přijatých zpráv. Každá zpráva určená pro zpracování tímto skriptem se skládá z prefixu určujícího typ zprávy a vlastního těla zprávy. Prvním krokem při zpracování SMS je ověření práv odesílatele. Pokud není v tabulce smsusers číslo odesílatele spolu s prefixem obsaženým ve zprávě, je tato zpráva ignorována.

Momentálně jsou používány dva prefixy: CONNECT a BASH. Prefix CONNECT znamená požadavek na sestavení připojení k internetu. Po jeho přijetí je spuštěn skript, který toto připojení vytvoří. Prefix BASH znamená požadavek na spuštění bashového skriptu, uloženého v těle zprávy. Po skončení tohoto skriptu je jeho výstup odeslán na telefonní číslo odesílatele zprávy.

4.8.3 Spojení pomocí GPRS

Přenos dat pomocí GPRS (nebo podobné technologie viz. kapitola Použité technologie) zajišťuje přenos pomocí protokolu TCP/IP a je s ním možné přistupovat ke všem funkcím jako přes lokální síť (i když s nižší rychlostí). Pro připojení pomocí GPRS se používá protokol PPP¹⁵ (Point-to-Point Protocol). V operačním systému GNU/Linux se o komunikaci pomocí protokolu PPP stará daemon pppd ve spojení s ovladačem PPP v jádře. Veškerá konfigurace potřebná pro navázání spojení je uložena ve dvou souborech.

První z nich je umístěn v adresáři /etc/ppp/peers (v případě realizovaného zařízení se soubor jmenuje oskar). Tento soubor obsahuje nastavení pppd daemonu, jako je použitý komunikační port (příslušný speciální soubor), rychlost, přihlašovací

¹⁵Jedná se o linkový protokol umožňující autentifikaci, šifrování a kompresi přenášených dat. Používá se k propojení dvou síťových uzlů přes sériový kabel, telefonní modem, radiomodem apod.

údaje, nastavení dns a podobně. Tento soubor také odkazuje na další konfigurační soubor (tzv. Chatscript), uložený v adresáři `/etc/chatscripts` (v případě této práce se jedná o soubor `gprs`). Tento soubor obsahuje instrukce pro komunikaci s vlastním modemem. Obsah tohoto souboru udává, jaké příkazy mají být posílány modemu a jaké odpovědi mají být očekávány. Pokud nedojde očekávaná odpověď je celé spojení ukončeno.

Po úspěšném provedení chatscriptu dojde k zahájení komunikace mezi oběma stranami PPP spojení a ty na základě nastavení v souboru `/etc/ppp/peers/oskar` vzájemně vyjednájí nastavení síťových rozhraní (IP adresa, maska, DNS server). Na závěr přidá záznamy do routovací tabulky, které umožní komunikovat s uzlem na opačné straně PPP spojení.

4.8.4 Síťový tunel

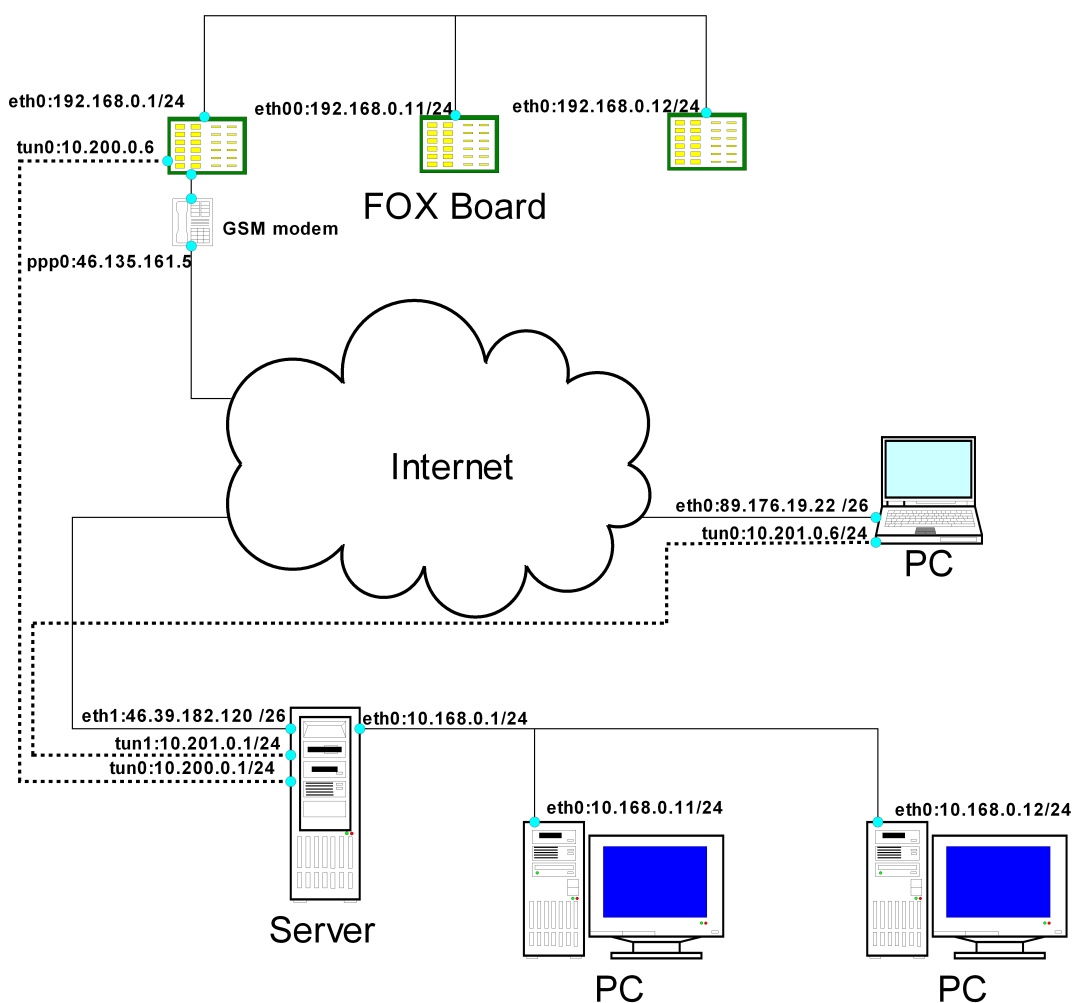
Zařízení připojená přes GPRS většinou nedostanou svoji vlastní veřejnou IP adresu, ale pouze adresu privátní. Přístup do internetu je tak řešen za pomoci překladu adres, označovaného anglickou zkratkou NAT (Network address translation)¹⁶. K takto připojenému zařízení se tedy není možné připojit zvenku. Aby bylo možné připojit se k zařízení zvenčí, je potřeba, aby mělo přidělenou veřejnou IP adresu, která navíc musí být statická (v opačném případě by bylo nutné před každým připojením k zařízení zjistit, jakou má adresu). Všichni operátoři nabízejí za příplatek možnost přidělení statické veřejné IP adresy. Někteří operátoři mohou a tak umožňují blokovat příchozí spojení a je tedy nutné vybrat službu, kde k tomuto blokování nedochází¹⁷.

Všechny výše uvedené problémy je možné vyřešit vytvořením síťového tunelu ze zařízení (klienta) s privátní adresou na server s adresou veřejnou. Vytvoření tunelu může být iniciováno ze zařízení s privátní adresou. Veškerá další komunikace mezi serverem a klientem prochází přes již sestavený tunel a je tedy možné navazovat spojení i směrem ze serveru ke klientovi.

¹⁶NAT je funkce routeru, která u paketů odchazejících z privátní sítě nahrazuje adresu odesilatele adresou svého WAN portu. Příchozí pakety pak na základě informací o navázaných spojeních přesměrovává jednotlivým stanicím na místní síti. Vzhledem k tomu, že pro určení adresáta je nutná informace o navázaném spojení, musí komunikaci vždy iniciovat klient v privátní síti

¹⁷Na první pohled se může zdát blokování příchozích spojení jako nežádoucí funkce. Většina mobilních připojení má však poměrně nízké limity přenesených dat nebo je účtován přímo objem přenesených dat. Pokud tedy nejsou příchozí spojení blokována na straně poskytovatele připojení, může kdokoliv se znalostí adresy zařízení generovat velký provoz, který vyčerpá datový limit nebo zvýší účet za přenesená data. Častým zdrojem těchto přenosů jsou programy procházející velké rozsahy adres a snažící se hrubou silou prolomit heslo pro přihlášení k vzdálenému systému (většinou používají protokol SSH, někdy se však pokoušejí i o přihlášení pomocí webového rozhraní a dalších služeb). Tyto útoky mohou běžně trvat několik hodin a vygenerovat desítky MB přenosu.

V této práci je pro vytvoření síťového tunelu použit program openVPN pro vytvoření šifrovaného tunelu na třetí vrstvě ISO/OSI modelu. Pro správnou funkci je potřeba nakonfigurovat jednak vlastní openvpn daemon na straně serveru i klienta a také routovací tabulky na obou stranách spojení, tak aby odpovídaly konkrétnímu uspořádání sítě. Na obr. 4.7 je znázorněna situace kdy jsou pomocí tunelu propojeny dvě privátní sítě. V první síti jsou registrační zařízení a počítače používané k přístupu k naměřeným datům, z druhé sítě se jen přistupuje k naměřeným datům. Tato konfigurace bude dále popsána.



Obr. 4.7: Uspořádání sítě.

Instalace a nastavení openVPN

Na straně serveru je potřeba jádro s podporou TUN zařízení (ta je v případě Debianu ve výchozím stavu zapnuta) a vlastní openVPN daemon. Ten je možné nainstalovat

příkazem:

```
aptitude install openvpn
```

Dále je nutné připravit certifikáty, které budou použity pro zabezpečení komunikace. K jejich tvorbě je možné použít předpřipravenou sadu skriptů distribuovanou v balíčku spolu s openVPN. Před další prací s těmito skripty je vhodné zkopírovat je z původního umístění příkazem:

```
cp -R /usr/share/doc/openvpn/examples/easy-rsa/ /etc/openvpn
```

Poté je nutné do souboru `/etc/openvpn/easy-rsa/2.0/vars` vyplnit údaje použité pro tvorbu certifikační autority. Tu je následně možné vytvořit příkazy:

```
. /etc/openvpn/easy-rsa/2.0/vars  
. /etc/openvpn/easy-rsa/2.0/clean-all  
. /etc/openvpn/easy-rsa/2.0/build-ca
```

S připravenou certifikační autoritou je možné vygenerovat vlastní certifikát pro server:

```
. /etc/openvpn/easy-rsa/2.0/build-key-server server
```

A následně také certifikáty pro klienta (pokud bude k jednomu openVPN serveru připojeno více klientů, musí být pro každého z nich vytvořen vlastní certifikát).

```
. /etc/openvpn/easy-rsa/2.0/build-key client1
```

Na závěr stačí vygenerovat parametry použité pro Diffie-Hellman výměnu klíčů příkazem:

```
. /etc/openvpn/easy-rsa/2.0/build-dh
```

Všechny takto vytvořené soubory s privátními klíči a certifikáty obsahujícími veřejné klíče jsou uloženy v adresáři `/etc/openvpn/easy-rsa/2.0/keys`. Některé z těchto souborů je potřeba nakopírovat do adresáře s konfigurací openVPN (`/etc/openvpn/`) na serveru a některé na klientovy. Na serveru jsou potřeba soubory: `ca.crt`, `ca.key`, `dh1024.pem`, `server.crt` a `server.key`. Na straně klienta: `ca.crt`, `client1.crt`, `client1.key`.

Jako základ konfiguračních souborů je možné použít vzory připravené v `/usr/share/doc/openvpn/examples/sample-config-files/`. Dále zde budou popsány jen provedené změny. Kompletní konfigurační soubory jsou na přiloženém CD.

Konfigurace serveru

K vytvoření tunelu byl použit protokol UDP, který sice nezajišťuje doručení paketu, ale má lepší latency než TCP/IP. Většina protokolů (ssh, http, scp) navíc sama používá TCP/IP a dvojnásobná kontrola doručení by tedy jen vytvářela zbytečnou zátěž. Pro vlastní tvorbu tunelu je použito virtuální zařízení TUN. To zajišťuje propojení sítí na 3. síťové vrstvě ISO/OSI modelu (další možností by bylo použití TAP zařízení pracujícího na 2. vrstvě ISO/OSI modelu.). Pro zajištění správného routování je také nutné přidat řádek:

```
route 192.168.0.0 255.255.255.0
```

Ten přidá do routovací tabulky záznam který udává přes který směrovač se dá připojit do vzdálené sítě (Adresu směrovače a rozhraní není nutné zadávat,. Ta je určena openvpn serverem v době zpracování konfiguračního souboru). Poslední důležitou volbou je:

```
client-config-dir ccd
```

Ta zajistí že každý připojený klient dostane od serveru konfiguraci uloženou v adresáři na serveru. Na straně klienta tedy stačí v konfiguračním souboru použít jen volby nutné pro navázání spojení a zbytek může být získán od serveru. Tento přístup je pohodlnější z hlediska budoucích změn konfigurace, protože většinu změn v nastaveních je možné provést pouze na serveru a není nutné aktualizovat konfigurační soubory na všech klientech.

Konfigurace klienta

Konfigurace klienta je ve většině voleb shodná se serverem. Zde budou popsány pouze volby které má klient navíc oproti serveru. Do konfiguračního souboru klienta jsou přidány následující řádky:

```
ping 2  
ping-restart 10  
up-delay
```

První řádek zajistí odeslání kontrolního paketu každé 2s. Druhý zajišťuje restart spojení pokud nedojde odezva na kontrolní paket po dobu 10s. Poslední odkládá zapnutí TUN rozhraní až do doby kdy je skutečně navázáno spojení. V opačném případě je rozhraní zapnuto při prvním pokusu o spojení a zároveň s tím je upravena routovací tabulka, přesto že vůbec není jisté že se spojení podaří. Pro zajištění správného routování je důležitá volba:

```
redirect-gateway def1
```

Ta zajistí nahrazení výchozí brany adresou opačné strany tunelu a tím zajistí že veškerý provoz bude směřovat skrz tunel. Nevýhodou této volby je, že skrz tunel půjde i veškerý provoz z klienta do internetu. Na druhou stranu zjednodušuje konfiguraci. Alternativou by bylo přidat do routovací tabulky záznamy pro všechny sítě které mají být přístupné přes tunel.

5 ZÁVĚR

Cílem této práce bylo navrhnout zařízení zajišťující záznam a prezentaci dat naměřených v technologickém procesu pomocí automatů firmy Micropel zapojených v síti.

V rámci práce byla vytvořena deska s podpůrnými obvody doplňující desku mikropočítače FOX Board o potřebné funkce. Mezi nejdůležitější doplněné funkce patří galvanicky oddělený budič rozhraní RS485 pro komunikaci se sítí automatů a zálohovaný zdroj. Zálohovaný zdroj napájení umožňuje krátkodobé napájení při výpadku, potřebné hlavně pro bezpečné ukončení práce se souborovým systémem. Kromě toho jsou přidány další obvody, které v současnosti nemusí být využity, ale dá se předpokládat jejich využití v budoucích aplikacích. Jedná se o obvody zajišťující přizpůsobení napěťových úrovní a galvanické oddělení vstupů a výstupů, převodník úrovní pro rozhraní RS232 a zdroj napětí pro GSM modem umožňující využít společné zálohování jako hlavní zdroj pro FOX Board.

Součástí práce je také řešení softwarového vybavení zajišťující požadované funkce. Jsou zde řešeny specifické problémy přípravy a instalace operačního systému GNU/Linux na embeded zařízení a tvorba nových programů. Dále byl vyvinut ovladač zajišťující komunikaci se sítí automatů firmy Micropel, propojených sběrnici RS485 a komunikujících vlastním protokolem Pesnet. Dále byla navržena pro databáze ukládání konfigurace záznamu a zaznamenaná data. Pro záznam dat slouží program logger, který používá konfiguraci z databáze a na jejím základě zajišťuje čtení a záznam dat. Pro prezentaci dat pak bylo vytvořeno webové rozhraní, které umožňuje prohlížet zaznamenaná data, vytvářet z nich protokoly a tisknout je.

Navržené zařízení bylo vyrobeno a oživeno s výše popsáním software a v současnosti je v celkovém počtu 7 kusů nasazeno k monitorování technologických procesů ve farmacii. Tři kusy byly červnu 2011 nasazeny ve zkušebním provozu k záznamu průběhu sterilizace na třech nezávislých parních sterilizátorech. Během tohoto testovacího provozu byly odstraněny některé chyby ve webovém rozhraní, které bylo zároveň upraveno dle přání zákazníka. Dnes je používáno v běžném provozu pro tvorbu protokolů používaných jako průvodní dokumentace k produktům.

Další 3 kusy jsou použity pro záznam lyofilizačního procesu na třech nezávislých lyofilizačních zařízeních. V tomto případě navíc zaznamenávají proces mražení materiálu při přípravě na lyofilizaci. V tomto případě byl zkušební provoz zahájen na konci prosince 2011. Od té doby je v provozu prakticky nepřetržitě bez větších problémů.

Poslední zařízení nezajišťuje žádný záznam veličin ale je vyhrazeno jen pro zprostředkování vzdáleného přístupu pomocí GSM modemu. S ostatními zařízeními je propojeno pomocí Ethernetu. Toto řešení umožňuje umístit GSM modem kamkoliv

v dosahu sítě Ethernet což je důležité vzhledem k tomu že v oblasti je velmi špatné pokrytí.

Při zkušebním provozu se projeví některé problémy v software i v podpůrných obvodech, které byly následně vyřešeny. Návrh podpůrných obvodů v této práci již tyto úpravy obsahuje a jsou podrobněji popsány v příslušných kapitolách.

Velký prostor pro budoucí rozšíření je hlavně na straně software. Pokud by byl vyžadován záznam se vzorkovací frekvencí kratší než cca 1s, bylo by vhodné upravit program logger tak aby využíval neblokujícího čtení. Zároveň by bylo vhodné použít linuxové jádro s real-time patchem, které by umožnilo přesnější odměření času jednotlivých vzorků. U webového rozhraní se v některých případech ukazuje jako problém práce s velkým množstvím dat. Tyto problémy jsou způsobeny převážně použitím jazyka PHP a jeho prací s poli. V PHP jsou všechna pole asociativní a práce s nimi je v některých případech mnohonásobně pomalejší než například s běžnými dynamicky alokovanými poli v jazyce C++. Řešením by tedy mohlo být nahradit funkce tříd pro práci s daty programem v C++ a příslušné třídy použít pouze jako wrapper. Vzhledem k tomu že již nyní je k zařízení připojen GSM modem, přímo se nabízí možnost sledování hodnot jednotlivých veličin s možností zasílat upozornění v případě překročení havarijních mezí. Tuto funkci by bylo možné realizovat jednoúčelovým skriptem spouštěným z cronu. Pokud by se však mělo jednat o universálnější řešení bylo by vhodné zakomponovat konfiguraci této funkce do webového rozhraní.

LITERATURA

- [1] MICROPEL. *Dokumentace automatů MICROPEL* [online],[2011-05-18]
URL: http://www.micropel.cz/index.php?option=com_nahledy&Itemid=7&lang=cs
- [2] MICROPEL. *Archivační program DataStore* [online],[2011-05-18] URL:
http://www.micropel.cz/index.php?option=com_content&view=article&id=87&Itemid=92&lang=cs
- [3] MICROPEL. *Dokumentace protokolu pesnet* [online],[2011-05-18] URL: <http://www.micropel.cz/images/stories/Dokumenty/pesnet.pdf>
- [4] STALLMAN, Richard. *Linux and the GNU Project* [online],[2011-05-18] URL:
<http://www.gnu.org/gnu/linux-and-gnu.html>
- [5] ACME. *FOX Board G20 developers* [online],[2011-05-18] URL: <http://foxg20.acmesystems.it/doku.php>
- [6] EMBEDIAN. *Embedded Debian Project* [online],[2011-05-18] URL: <http://www.emdebian.org/grip/>
- [7] Linux Kernel Organization. *The Linux Kernel Archives* [online],[2011-05-18]
URL: <http://www.kernel.org/>
- [8] ACME, *Compiling the Linux Kernel 2.6.35.4* [online],[2011-05-18]
URL: http://foxg20old.acmesystems.it/doku.php?id=tutorial:compiling_the_linux_kernel
- [9] *fdisk(8) - Linux man page*[online],[2011-05-18] URL: <http://linux.die.net/man/8/fdisk>
- [10] Linux Foundation. *Comment Conventions for Init Scripts*[online],[2011-05-18]
URL: http://refspecs.linuxfoundation.org/LSB_3.1.0/LSB-Core-generic/LSB-Core-generic/initscrcomconv.html
- [11] ATMEL. *AT91SAM9G20 Summary* [online],[2011-05-18] URL: http://www.atmel.com/dyn/resources/prod_documents/6384s.pdf
- [12] STEC. *Application note*[online],[2011-05-18] URL: http://www.stec-inc.com/downloads/AN-0702_STEC_SMALL_CARDS_WEAR_LEVELING_LIFETIME_CALCULATOR.pdf

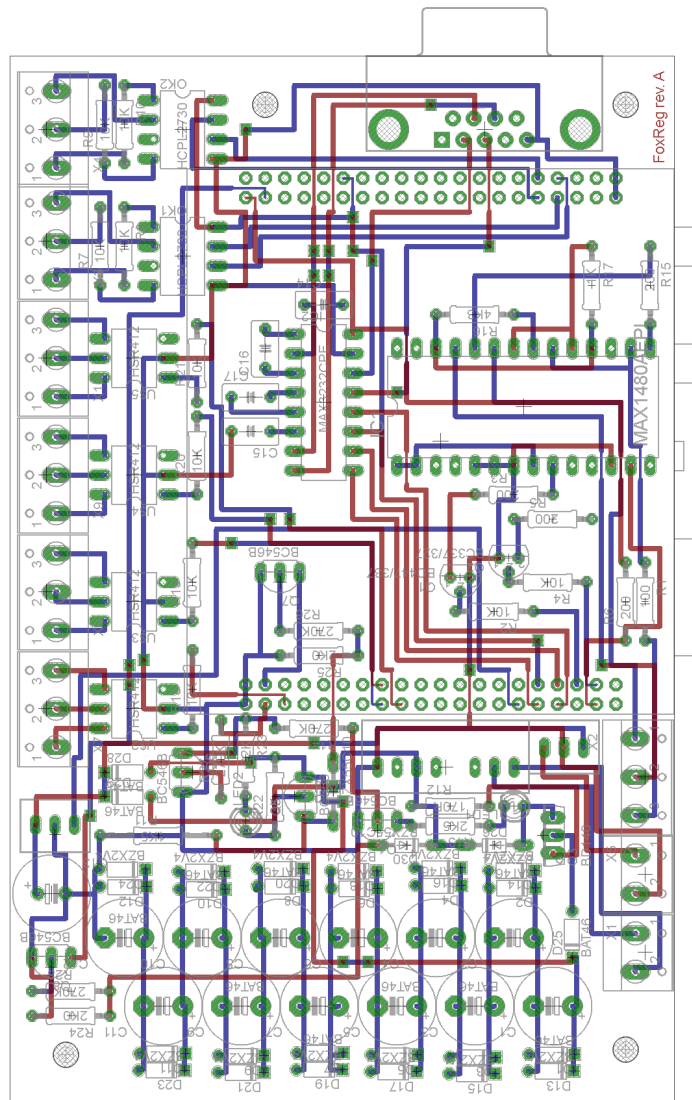
- [13] Agilent. *Agilent HCPL-270L/ 070L/273L/073L Low Input Current High Gain LVTTTL/ VCMOS Compatible 3.3 V Optocouplers Data Sheet*[online],[2011-05-18] URL: <http://www.farnell.com/datasheets/2794.pdf>
- [14] FAIRCHILD SEMIDUCTOR. *HSR312, HSR312L, HSR412, HSR412L Photo-voltaic Solid-State Relay Optocouplers*[online],[2011-05-18] URL: <http://www.farnell.com/datasheets/87286.pdf>
- [15] MAXIM. *±15kV ESD-Protected, Isolated RS-485/RS-422 Data Interfaces*[online],[2011-05-18] URL: <http://datasheets.maxim-ic.com/en/ds/MAX1480E-MAX1490E.pdf>
- [16] Michal Čihař. *Gammu Utility* [online],[2011-05-18] URL: <http://wammu.eu/docs/manual/gammu/>
- [17] JONATHAN C., ALESSANDRO R., GREG K. *Linux Device Drivers, Third Edition* O'Reilly Media 2005. ISBN 0-596-00590-3
- [18] EMBEDIAN. *Embedded Debian Project* [online],[2011-05-18] URL: <http://www.emdebian.org/grip/>
- [19] PFEIFFER VACUUM. *Compact Pirani Gauge TPR 280 281* [online],[2011-05-18] URL: http://www.mpi-hd.mpg.de/gerda/TG04_TECHNICAL/manuals/bg805178be_d_web.pdf

SEZNAM PŘÍLOH

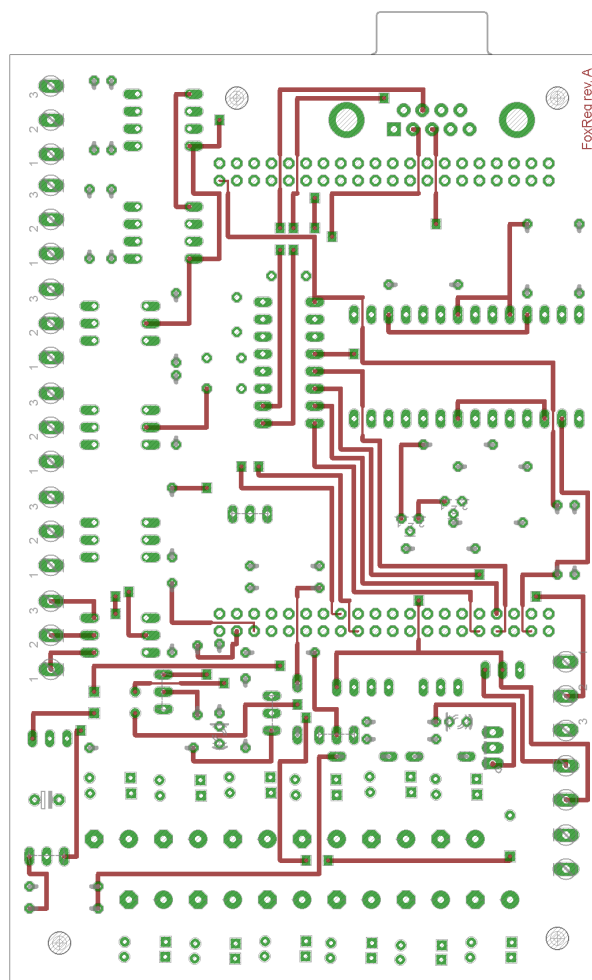
A	Deska plošných spojů	66
A.1	Celá deska	66
A.2	Vrchní strana PCB	67
A.3	Spodní strana PCB	68
A.4	Osazovací výkres	69
B	Instalace debianu	70
B.1	Příprava před spuštěním	70
B.2	Skript stage2	71
C	Fotky zařízení v provozu	72
D	Ukázka vytvořeného protokolu	74

A DESKA PLOŠNÝCH SPOJŮ

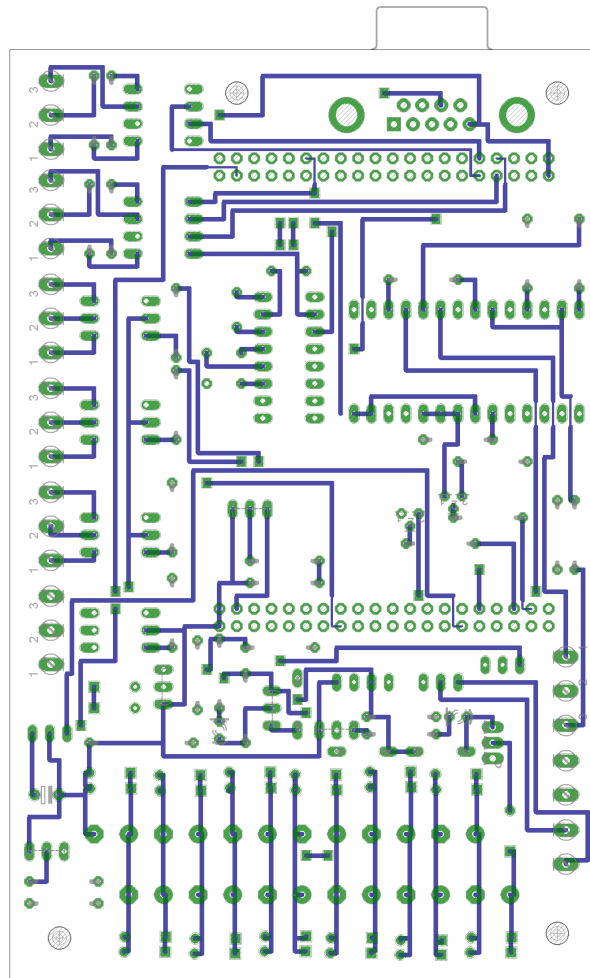
A.1 Celá deska



A.2 Vrchní strana PCB



A.3 Spodní strana PCB



B INSTALACE DEBIANU

B.1 Příprava před spuštěním

```
#instalace základního systému
debootstrap --arch armel --download-only --foreign squeeze \\\
/mnt/sdX2 http://ftp.us.debian.org/debian
cd /mnt/sdX2

#Příprava fstab
echo '/dev/mmcbk0p2 / ext4 noatime 0 1' > etc/fstab
echo 'proc /proc proc none 0 0' >> etc/fstab

#Příprava inittab
echo 'id:1:initdefault:' > etc/inittab
echo 'si:1:respawn:/stage2' >> etc/inittab

#vytvoření souboru hosts
echo '127.0.0.1 localhost foxreg' > etc/hosts

#vytvoření interfaces
echo 'auto lo' > etc/network/interfaces
echo 'iface lo inet loopback' >> etc/network/interfaces
echo 'allow-hotplug eth0' >> etc/network/interfaces
echo 'iface eth0 inet static' >> etc/network/interfaces
echo 'address 192.168.0.1' >> etc/network/interfaces
echo 'netmask 255.255.255.0' >> etc/network/interfaces
echo 'auto ppp0' >> etc/network/interfaces

#Vytvoření speciálního souboru konzole
mknod -m 600 dev/console c 5 1
```

B.2 Skript stage2

```
#Tento soubor je spuštěn až na cílovém systému
./debootstrap/debootstrap --second-stage

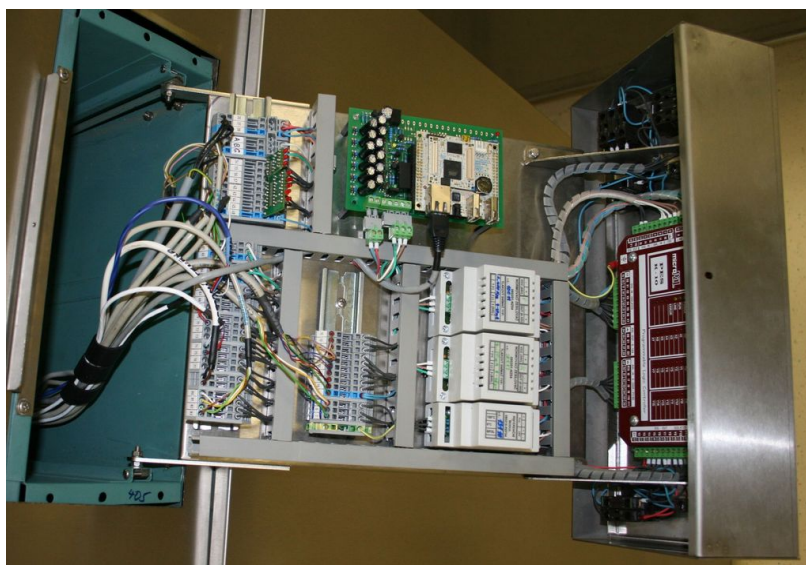
#Příprava zdroje pro aptitude
echo 'deb http://ftp.cz.debian.org/debian/ squeeze main' \\  
>> /etc/apt/sources.list
echo 'deb-src http://ftp.cz.debian.org/debian/ squeeze main' \\  
>> /etc/apt/sources.list
echo 'deb http://security.debian.org/ squeeze/updates main' \\  
>> /etc/apt/sources.list
echo 'deb-src http://security.debian.org/ squeeze/updates main' \\  
>> /etc/apt/sources.list

#Instalace ssh
/usr/bin/aptitude install ssh
```

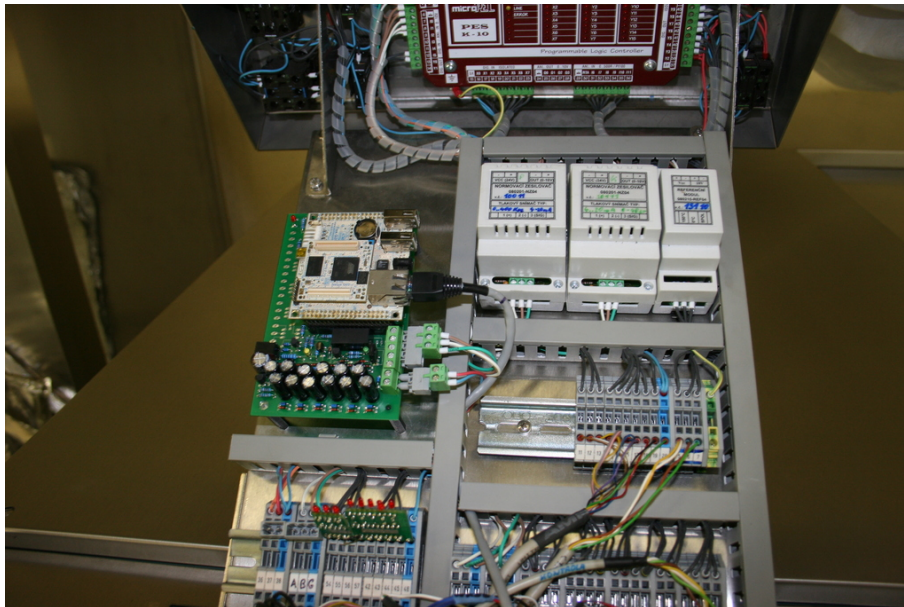
C FOTKY ZAŘÍZENÍ V PROVOZU



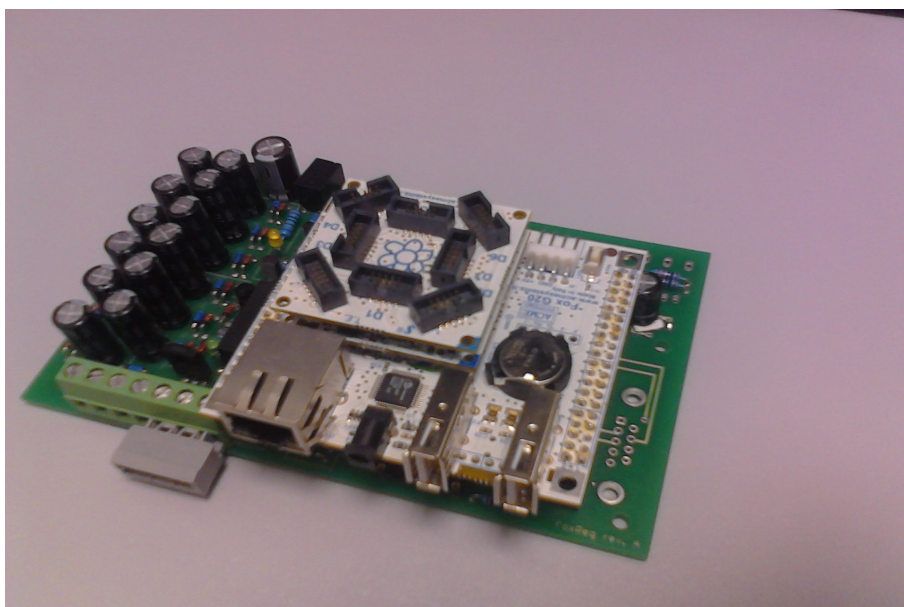
Obr. C.1: Sterilizátor s vysunutým rozvaděčem



Obr. C.2: Vysunutý rozvaděč

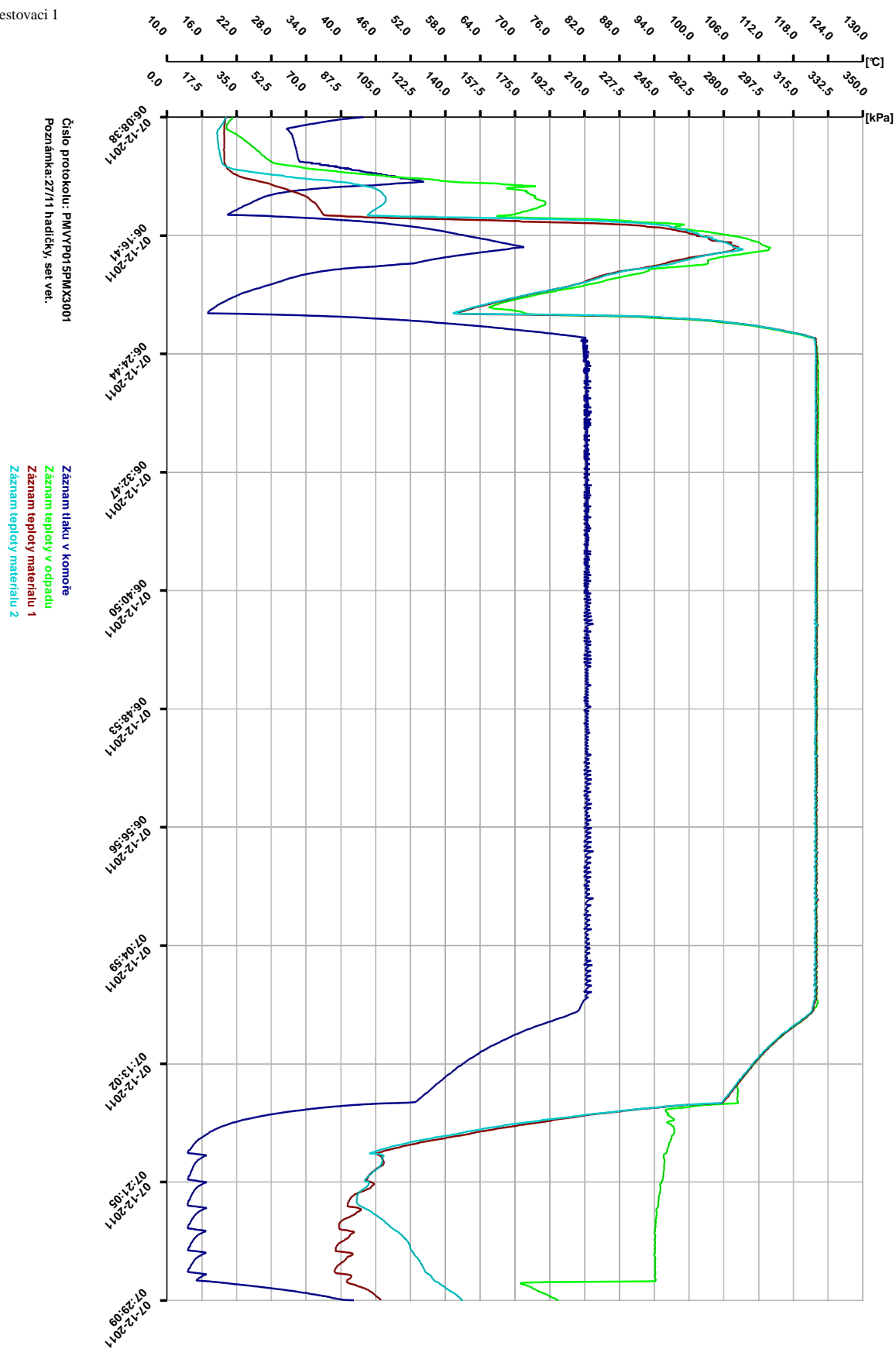


Obr. C.3: Detail rozvaděče s FOX Boardem a podpůrnými obvody na levé straně



Obr. C.4: Detail desky s podpůrnými obvody

D UKÁZKA VYTVOŘENÉHO PROTOKOLU



Testovací 1				
Čas	Záznam tlaku v komoře [kPa]	Záznam teploty v odpadu [°C]	Záznam teploty materiálu 1 [°C]	Záznam teploty materiálu 2 [°C]
07-12-2011 06:08:38	98.8	21.2	20.0	20.1
07-12-2011 06:09:38	61.6	21.1	19.9	18.7
07-12-2011 06:10:38	64.4	24.7	19.9	18.9
07-12-2011 06:11:38	66.4	27.9	19.9	19.4
07-12-2011 06:12:38	112.0	47.8	22.4	30.2
07-12-2011 06:13:38	64.8	71.8	31.2	46.6
07-12-2011 06:14:38	39.2	75.0	35.8	46.9
07-12-2011 06:15:38	88.8	87.3	65.2	82.2
07-12-2011 06:16:38	148.4	106.6	101.3	101.8
07-12-2011 06:17:38	169.2	113.8	107.7	109.3
07-12-2011 06:18:38	118.0	103.1	95.4	96.8
07-12-2011 06:19:38	60.8	86.9	83.4	84.0
07-12-2011 06:20:38	37.6	74.7	73.3	73.1
07-12-2011 06:21:38	23.2	66.3	62.7	62.2
07-12-2011 06:22:38	137.6	106.7	107.5	107.8
07-12-2011 06:23:38	210.4	121.0	121.4	121.3
07-12-2011 06:24:38	212.4	122.1	121.9	121.8
07-12-2011 06:25:40	210.4	122.2	121.9	121.8
07-12-2011 06:26:40	210.4	122.3	122.1	122.0
07-12-2011 06:27:39	209.6	122.3	122.0	121.8
07-12-2011 06:28:40	213.6	122.3	122.0	121.8
07-12-2011 06:29:40	210.4	122.2	122.1	121.8
07-12-2011 06:30:39	210.8	122.2	121.9	121.9
07-12-2011 06:31:40	209.6	122.2	122.0	121.8
07-12-2011 06:32:40	210.4	122.2	122.0	121.9
07-12-2011 06:33:40	209.6	122.2	122.0	121.8
07-12-2011 06:34:39	210.4	122.2	122.0	121.8
07-12-2011 06:35:39	210.8	122.1	122.0	121.8
07-12-2011 06:36:39	211.6	122.2	122.1	122.0
07-12-2011 06:37:39	210.8	122.1	122.1	121.9
07-12-2011 06:38:39	210.8	122.1	122.1	121.8
07-12-2011 06:39:39	211.2	122.1	122.1	121.9
07-12-2011 06:40:39	210.4	122.1	122.0	121.8
07-12-2011 06:41:39	210.4	122.1	121.8	121.6
07-12-2011 06:42:39	212.0	122.1	122.1	122.0
07-12-2011 06:43:39	212.4	122.0	122.0	121.9
07-12-2011 06:44:39	210.8	121.9	122.0	121.9
07-12-2011 06:45:39	210.8	122.0	122.1	121.8
07-12-2011 06:46:39	210.4	121.9	121.8	121.7
07-12-2011 06:47:39	210.8	122.0	122.0	121.8
07-12-2011 06:48:39	210.8	122.1	121.9	121.8
07-12-2011 06:49:39	210.4	122.1	121.9	121.8
07-12-2011 06:50:39	210.8	122.1	122.0	121.8
07-12-2011 06:51:39	211.6	122.1	122.0	121.9
07-12-2011 06:52:39	211.2	122.0	122.1	121.9
07-12-2011 06:53:39	210.4	122.0	121.9	121.7
07-12-2011 06:54:39	211.2	121.9	122.0	121.9
07-12-2011 06:55:39	213.2	121.9	121.8	121.6
07-12-2011 06:56:40	210.4	122.0	121.9	121.7
07-12-2011 06:57:40	212.4	121.9	122.1	122.0
07-12-2011 06:58:40	212.4	121.9	122.2	122.1
07-12-2011 06:59:39	210.4	122.0	121.8	121.6
07-12-2011 07:00:39	212.4	122.0	122.1	121.9
07-12-2011 07:01:39	210.8	122.0	122.0	121.8
07-12-2011 07:02:39	212.0	121.9	121.8	121.7
07-12-2011 07:03:39	211.6	122.0	122.1	121.9
07-12-2011 07:04:40	211.2	122.0	122.0	121.9
07-12-2011 07:05:40	210.8	121.9	122.0	121.8
07-12-2011 07:06:39	210.4	121.9	121.8	121.6
07-12-2011 07:07:39	210.4	121.9	121.8	121.6
07-12-2011 07:08:39	210.4	122.0	122.0	121.8
07-12-2011 07:09:39	203.2	120.9	120.9	120.7
07-12-2011 07:10:39	182.8	117.7	117.6	117.4

Testovací 1				
Čas	Záznam tlaku v komoře [kPa]	Záznam teploty v odpadu [°C]	Záznam teploty materialu 1 [°C]	Záznam teploty materialu 2 [°C]
07-12-2011 07:11:39	166.0	114.6	114.6	114.3
07-12-2011 07:12:39	152.8	112.1	112.1	111.9
07-12-2011 07:13:39	142.8	110.0	110.0	109.8
07-12-2011 07:14:39	133.6	108.3	107.9	107.8
07-12-2011 07:15:39	124.4	108.5	105.8	105.7
07-12-2011 07:16:39	45.6	96.9	80.5	79.6
07-12-2011 07:17:39	22.0	97.5	64.1	62.0
07-12-2011 07:18:39	12.8	96.5	50.7	49.0
07-12-2011 07:19:39	15.2	95.8	47.3	47.2
07-12-2011 07:20:39	11.6	95.6	44.8	44.8
07-12-2011 07:21:39	13.6	95.1	44.1	43.5
07-12-2011 07:22:39	10.4	94.7	41.2	43.1
07-12-2011 07:23:39	12.4	94.2	40.4	47.2
07-12-2011 07:24:39	16.0	94.2	41.6	50.3
07-12-2011 07:25:39	10.8	94.1	39.1	52.0
07-12-2011 07:26:39	12.4	94.2	39.9	53.8
07-12-2011 07:27:39	15.6	94.2	41.5	55.8
07-12-2011 07:28:39	70.4	74.9	45.5	59.5

Ekvivalentní doba sterilizace je 57 minut
Sterilizace proběhla úspěšně