České vysoké učení technické v Praze

Fakulta elektrotechnická

Katedra řídicí techniky

Diplomová práce

Řízení pohonů po sběrnici Profibus DP

22.května 2009 Martin Váňa

České vysoké učení technické v Praze
Fakulta elektrotechnická

Katedra řídicí techniky

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: **Bc. Martin Váňa**

Studijní program: Elektrotechnika a informatika (magisterský), strukturovaný
Obor: Kybernetika a měření, blok KM1 - Řídicí technika

Název tématu: **Řízení pohonů po sběrnici Profibus DP**

Pokyny pro vypracování:

Cílem práce je realizace zařízení Profibus DP slave vhodného pro řízení stejnosměrných a bezkartáčových motorů.

1. Realizujte zařízení Profibus DP slave s procesorem LPC2129 a komunikačním čipem VPC3+ s využitím starter kitu pro VPC3+.
2. Implementujte verze protokolu DPV0 - DPV2 na základě dodaného zdrojového kódu ke komunikačnímu čipu, který je pro procesor na bázi 8051.
3. Upravte stávající výkonovou část pro řízení krokových motorů z procesoru LPC2129 tak, aby byla vhodná pro připojení DC a BLDC motorů. Navrhněte finální desku s VPC3+. Při návrhu desky počítejte jak s připojením procesoru LPC pro méně náročné aplikace, tak s připojením procesoru Freescale ColdFire nebo PowerPC.
4. V hlavních rysech implementujte profil ProfiDrive.

Seznam odborné literatury:

Dodá vedoucí práce

Vedoucí: Ing. Pavel Burget, Ph.D.

Platnost zadání: do konce letního semestru 2009/10

L.S.

prof. Ing. Michaela Šebek, DrSc.
vedoucí katedry

doc. Ing. Boris Šimák, CSc.
děkan

V Praze dne 27. 2. 2009

Prohlášení

Prohlašuji, že jsem svou diplomovou práci zpracoval samostatně a použil jsem pouze podklady (literaturu, projekty, SW atd.) uvedené v přiloženém seznamu.

V Praze dne 21. května 2009

Anotace

Tato práce je zaměřena na krátké seznámení se sběrnicí Profibus DP a vývojovým kitem od firmy Profichip. Dále pak na návrh hardware zařízení Profibus DP slave s využitím obvodu ASIC VPC3+C od firmy Profichip. Tento obvod je využíván jako periferie k mikroprocesorovému modulu LPC2119, který splňuje předpoklady pro vývoj nízkonákladového zařízení. V druhém případě pak spojení s procesorem H8S2638, který je naopak určen pro náročné aplikace. Poslední část se zabývá vývojem softwaru a portací automatů Profibus DP V0 a V1 do zmíněných systémů a jejich využití v realizované aplikaci pro ovládání výukového robota.

Annotation

This thesis is aimed at the brief introduction to the industrial field bus Profibus DP and to start with the Profichip VPC3+C evolution board. Further goal is to design a slave machine based on ASIC VPC3+C manufactured by Profichip company. For this purpose are used two different microprocessors. The first one is the LPC2119 designated for a low-cost applications and the second one is the H8S2638 suggested for high performance applications. The last part of this thesis is about software development and a program customization for different architectures. There is also briefly described a realized application for an educational purposes.

Poděkování

Chtěl bych poděkovat všem lidem, bez nichž by nemohla tato práce vzniknout, především svému vedoucímu diplomové práce *ing. Pavlu Burgetovi, Ph.D.* za vedení, připomínky a poznámky k práci. Dále bych rád poděkoval osobám blízkým, zejména za jejich morální podporu.

Obsah

1 Úvod **6**

2 Profibus DP **7**

2.1 Profibus 7

2.2 Fyzická vrstva 7

2.3 FDL vrstva 8

2.3.1 Typy rámců 8

2.3.2 SAP pro cyklickou datovou výměnu 9

2.3.3 Typy služeb 10

2.3.4 Stavový automat DP-V0 slave 11

2.4 Profibus DP-V1 - Acyklická komunikace 12

2.5 Profibus DP-V2 13

2.5.1 PROFIdrive 13

2.6 GSD soubory 13

3 Komunikační obvod VPC3+C **15**

3.1 Charakteristika obvodu 15

3.2 Starterkit Profichip 16

4 Řídicí a podpůrné obvody **20**

4.1 LPC2119 20

4.1.1 Fast GPIO 20

4.1.2 UART 21

4.1.3 Časovače 21

4.1.4 Systém přerušení 21

4.2 Renesas H8S/2638 21

4.2.1 Externí sběrnice 22

4.2.2 UART 22

4.2.3 Motor Control PWM Timer 22
4.2.4 Řadič přerušení 22
4.3 Siemens SIM1-2 23

5 Realizace hardware 24
5.1 Deska PB-LPC-VPC 24
5.2 Renesas H8S2638 board 26
5.3 Robot Oscar 27

6 Realizace software 31
6.1 Vývojové nástroje 31
6.2 Časování na sběrnici 31
6.3 Popis komunikačního stacku Profichip 32
6.4 Portace programu pro LPC2119 34
6.4.1 Náhrada externí sběrnice pomocí GPIO 35
6.4.2 Funkce pro práci s pamětí 37
6.4.3 Ostatní funkce a konfigurace 38
6.5 Portace programu pro H8S2638 38
6.6 Aplikace robot Oscar 41

7 Závěr 43

A Popis příloh na CD 45

B Schéma, DPS a propojení použitého HW 47
B.1 Deska s procesorem LPC2119 47
B.2 Deska PB-LPC-VPC 47
B.3 Deska Renesas H8S2638 47

Seznam obrázků

2.1 Struktura SD1 8
2.2 Struktura SD2 9
2.3 Struktura SD3 9
2.4 Struktura SD4 9
2.5 Struktura SC 10
2.6 Stavový automat Profibus DP slave [8] 11

3.1 Blokové schéma obvodu VPC3+C 17
3.2 Grafické rozhraní aplikace Flip 18

4.1 Blokové schéma části Motor Control PWM Timer 23

5.1 Blokové schéma propojení LPC2119 s částí VPC Profichip kitu 24
5.2 Blokové schéma navržené desky s VPC3+C 25
5.3 Rozmístění součástek na navržené desce PB-LPC-VPC 26
5.4 Rozmístění součástek na navržené desce H8S2638 28

6.1 Průběh časování při čtení z VPC3+C. 32
6.2 Průběh časování při zápisu do VPC3+C. 33
6.3 Hardwarová konfigurace systému 42
6.4 Jednoduché ovládání robota z prostředí STEP7 42

B.1 Schéma desky s procesorem LPC2119 48
B.2 Schéma navržené desky s PB-LPC-VPC 1.část 50
B.3 Schéma navržené desky s PB-LPC-VPC 2.část 51
B.4 Pohled na spodní vrstvu spojů desky PB-LPC-VPC 52
B.5 Pohled na vrchní vrstvu spojů desky PB-LPC-VPC 53
B.6 Rozvržení součástek na desce PB-LPC-VPC 54
B.7 Schéma navržené desky Renesas H8S2638 56
B.8 Pohled na spodní vrstvu spojů desky Renesas H8S2638 57

B.9 Pohled na vrchní vrstvu spojů desky Renesas H8S2638 57
B.10 Rozvržení součástek na desce Renesas H8S2638 58

Seznam tabulek

2.1 Zkratky v rámcích SDx 10
2.2 SAP používané pro cyklickou datovou výměnu 11
2.3 Aplikační třídy profilu PROFIdrive 14

3.1 Nastavení propojek na komunikační desce Profichip 19

5.1 Popis konektoru z robota Oscar do konektoru SV1 29

6.1 Režim komunikace VPC3+C 32
6.2 Mezní hodnoty časování při čtení a zápisu z/do VPC3+C. 33

B.1 Propojení pinů obvodu VPC3+C s deskou LPC2119 49
B.2 Popis universálního konektoru SV1 a propojení s deskou LPC2119 . . 55
B.3 Propojení desky Renesas H8S2638 s Profichip VPC3+C board 59

Kapitola 1

Úvod

V úvodu práce se stručně seznámíme s protokolem Profibus a jeho modifikacemi DP a PA. Profibus DP je jednou z nejvíce rozšířených verzí standardu Profibus a průmyslových sběrnic vůbec, zatímco verze PA je předurčen pro aplikace do nebezpečného prostředí. Úvodní seznámení je zaměřeno zejména z pohledu vrstvy FDL (Field Datalink Layer). Zádání práce je navrhnout zařízení s obvodem od firmy Profichip VPC3. VPC3 je tzv. ASIC obvod (Aplication Specific Integrated Circuit), který má implementován komunikační standard Profibus DP. Tento obvod však není schopen pracovat samostatně. Potřebuje ke své činnosti řídící mikroprocesor, takže vlastně tvoří řadič sběrnice. Takto lze tedy vytvořit vlastní inteligentní zařízení dle vlastních požadavků a nároků, které bude schopno pracovat v průmyslové síti i s jinými komerčně vyráběnými zařízeními.

Cíle práce jsou rozděleny do několika částí. Nejprve seznámení se starterkitem od firmy Profichip s obvodem VPC3 a dále pak s použitými procesory. Následně návrh vlastního low-costového zařízení s procesorem LPC2129 a implementace stavových automatů do tohoto systému.

Dalším krokem je portace programu do systému, který je založený na výkonějším procesoru Hitachi H8S/2638. Tento procesor disponuje narozdíl od levného LPC2129 takovými periferiemi, které umožňují snázeji vytvořit aplikaci pro řízení motorů.

V poslední kapitole se věnujeme vytvořeným aplikacím s navrženým hardwarovým řešením.

Kapitola 2

Profibus DP

2.1 Profibus

Kompletní popis protokolu Profibus je nad rámec této kapitoly a není cílem této práce. Proto bude v této kapitole uveden pouze stručný úvod do verze DP a PA, které jsou v práci využívány. Další popis bude uveden z pohledu fyzické a FDL (spojové) vrstvy. Konec kapitoly bude věnován popisu GSD souboru a profilům verze DP.

Profibus DP (Distributed Periphery) vznikl jako rozšíření předchozího protokolu FMS viz. [1]. Jeho jednoduchost způsobila rozšíření do aplikací automatizované výroby. Dále byl rozšířen o dvě verze DP-V1 a DP-V2.[8]

2.2 Fyzická vrstva

Fyzická vrstva ISO/OSI modelu popisuje hardwarové řešení sběrnice či sítě. Profibus DP využívá jako fyzickou vrstvu metalické vedení RS-485 nebo optický kabel. Optické řešení se používa v prostředí s velkým rušením nebo pro velké vzdálenosti. V této práci je využíváno rozhraní RS-485, ale starterkit od firmy Profichip umožňuje i optické připojení. V normě najdeme doporučení typu kabelu a zapojení konektoru D-sub.[1]

Profil Profibus PA se od profilu DP odlišuje právě fyzickou vrstvou. Navíc má verze PA oproti DP některá omezení jako například maximální rychlost komunikace 31.25 kbps oproti maximální rychlosti 12MB profilu DP. Tato verze využíva podobně jako standartní proudová smyčka 4-20mA nebo protokol HART napájení a komunikaci po jednom kabelu. Je vhodná dokonce i do výbušného prostředí. Ochrana proti

výbuchu je zajištěna především omezeným napájením. To je podobně jako použité kódování popsáno v normě IEC 61158-2.[2]

V praxi je možné sběrnice typu DP a PA propojit dvěma způsoby:

- Vazebním členem (coupler), který se jeví v síti transparentně, takže adresy zařízení PA jsou viditelné. Toto řešení má nevýhodu, že celá komunikace může probíhat maximální rychlosti, kterou umožňuje profil PA, tedy 31.25kbps.

- Druhou možností je použít zařízení DP/PA link, který se v segmentu DP jeví jako samostatné zařízení typu slave a v segmentu PA se chová jako typ master pro všechny moduly PA. Toto řešení je netransparentní a adresový prostor je oddělen. Cyklická komunikace mezi DP/PA linkem shrnuje data ze všech PA zařízení do jediného telegramu v segmentu DP. Komunikační rychlosti v obou segmentech mohou být rozdílné.[2]

2.3 FDL vrstva

Datová výměna v Profibus DP je prováděna zasíláním telegramů (rámců) ze zařízení typu master do zařízení typu slave a naopak. Vrstva 2 ISO/OSI modelu zahrnuje obecný popis rámců, zabezpečovací mechanismus a typy služeb.

2.3.1 Typy rámců

FDL vrstva rozlišuje čtyři typy rámců. Druh rámce se rozpozná podle prvního znaku SD = start delimiter.

- **SD1** = Request_FDL_Status: Tento rámec je zaslán vždy aktivní stanicí po vypršení GAP time. Podrobnější vysvětlení a popis je uveden v knize [7]. Struktura rámce je uvedena na obrázku 2.1.

SD	DA	SA	FC	FCS	ED
0x10	xx	Xx	X	x	0x16

Obrázek 2.1: Struktura SD1

- **SD2** = Telegram: Rámec pro přenos s proměnnou délkou dat. Obsahuje políčka LE a LEr, které obsahují délku dat včetně DA, SA, FC, DU. Používá se ve službě SRD viz. 2.3.3. Obrázek 2.2.

SD	LE	LEr	SD	DA	SA	FC	DU	FCS	ED
0x68	X	X	0x68	xx	xx	x	x ...	X	0x16

Obrázek 2.2: Struktura SD2

- **SD3** = Telegram: Rámec s pevnou délkou dat. DU je vždy dlouhé 8 bytů. Obrázek 2.3.

SD	DA	SA	FC	DU	FCS	ED
0xA2	xx	Xx	x	x ...	x	0x16

Obrázek 2.3: Struktura SD3

- **SD4** = Token telegram: Rámec zasílaný mezi dvěma aktivními stanicemi, nebo-li mastery, určený k předání přístupu na sběrnici. Obrázek 2.4.

SD	DA	SA
0xDC	xx	Xx

Obrázek 2.4: Struktura SD4

- **SC** = Short acknoledgement: Používá se pouze pro potvrzení příjmu. Hodnota je vždy 0xE5. Obrázek 2.5.

  Použité zkratky v popisu struktur jednotlivých rámců jsou vysvětleny v tabulce 2.1.

Pro adresu je použit jeden znak, ale využívá se pouze 7 bitů, takže lze adresovat 127 stanic - 0 až 126. Adresa 127 má speciální význam, je-li tento bit nastaven, v adrese DA nebo SA, začíná datové pole adresním rozšířením - tzv. přístupovým bodem SAP (Service Acces Point). Tyto body nabízejí další adresní možnosti, mohou identifikovat určitou funkci. Podrobnější popis lze nalézt v [7, 1].

2.3.2 SAP pro cyklickou datovou výměnu

Pro cyklickou datovou výměnu se využívá tzv. SAP - Service Access Point. Zjednodušeně si lze představit, že každý SAP je místem v paměti, které je vstupním nebo výstupním bodem nějakého cyklického procesu. Například vstupy do zařízení Profibus DP slave nastavují hodnoty v příslušném SAP a na druhé straně jsou tato data čtena nadřazeným zařízením. [8]

Pro cyklickou datovou výměnu pro verzi Profibus DP-V0 jsou definovány SAP uvedené v tabulce 2.2.

SC
0xE5

Obrázek 2.5: Struktura SC

Tabulka 2.1: Zkratky v rámcích SDx

SD:	Start Delimiter
LE:	Length: délka dat včetně DA, SA, FC, DU
LEr:	Repetition of the lenght: opakování délky
DA:	Destination Addres: cílová adresa
SA:	Source Addres: adresa vysílající stanice
FC:	Function Code: obsahuje řídící znak
DU:	Data Unit: uživatelská data, rozsah 1..244 bytů
FCS:	Frame Checking Sequence: obsahuje kontrolní součet CRC
ED:	End Delimiter: obsažen vždy na konci mimo SD4. Hodnota vždy 0x16

2.3.3 Typy služeb

Uživatel má k dispozici služby poskytované vrstvou FDL, kterou o provedení žádá prostřednictvím komunikačních primitiv REQ - request = žádost. FDL vrstva odpovídá primitivou CON - confirm = potvrzení pozitivní/negativní. Naopak vrstva FDL může volat vrstvu nadřazenou primitivou IND - indication = událost. Pro odpověď je používána primitiva RES - response = odpověď.[8]

Pro vrstvu FDL jsou definovány dva typy stanic:

- aktivní - může sama vysílat zprávy, jakmile získá přístup ke sběrnici,
- pasivní - může pouze odpovídat na žádosti aktivní stanice.

Nyní popíšeme definované služby vrstvy FDL.

- **SDA** Send Data with Acknowledge = pošli data a čekej na potvrzení. Data jsou poslána stanici master nebo slave a jako odpověď je posláno potvrzení. Je používáno pouze aktivními stanicemi.
- **SRD** Send and Request Data with acknowledge = pošli data a čekej zprávu s odpovědí. Tato služba je určena pro přímou výměnu dat mezi dvěma stanicemi. Buď mezi dvěma aktivními stanicemi nebo mezi aktivní a pasivní stanicí.

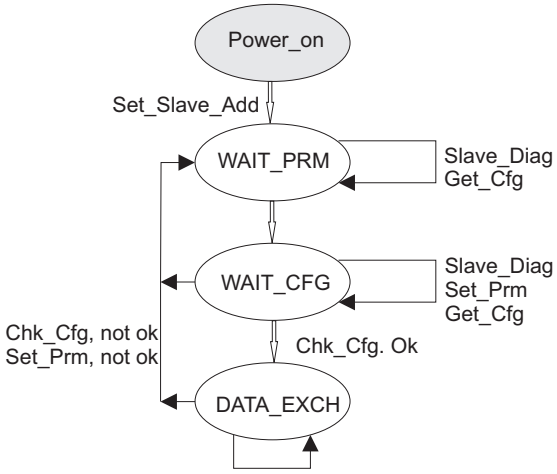
Tabulka 2.2: SAP používané pro cyklickou datovou výměnu

Default SAP:	Data exchange (Input_Output_Data)
SAP 0x36:	Master-master SAP (M-M communication)
SAP 0x37:	Change station address (Set_Slave_Add)
SAP 0x38:	Read inputs (Rd_Inp)
SAP 0x39:	Read outputs (Rd_Outp)
SAP 0x3A:	Control commands to the DP slave (Global_Control)
SAP 0x3B:	Read configuration data (Get_Cfg)
SAP 0x3C:	Read diagnostic information (Slave_Diagnosis)
SAP 0x3D:	Send parametrization data (Set_Prm)
SAP 0x3E:	Check configuration data (Set_Cfg)

- **SDN** Send Data with No Acknowledge = pošli data a neočekávej potvrzení. Používá se pro odeslání zprávy od jedné stanice k více stanicím či skupinám. Uživatel dostane potvrzení o odeslání zprávy, nikoliv však o jejím doručení.

2.3.4 Stavový automat DP-V0 slave

Pro snadné pochopení, jak Profibus DP funguje, je znázorněn na obrázku 2.6 jednoduchý stavový automat DP-V0 slave. Ten popisuje, jak se stanice bude chovat v možných situacích.



Obrázek 2.6: Stavový automat Profibus DP slave [8]

Ve stavu DATA_EXCH přijímá stanice tyto telegramy: Data Exchange-ok, Rd_Inp, Rd_Outp, příkazy (Sync, Freeze...), Slave_Diag, Chk_Cfg ok, Prm-ok,

Get_Cfg. Podrobný popis těchto příkazů lze nalézt v knize [7].

2.4 Profibus DP-V1 - Acyklická komunikace

Požadavky dnešních aplikací na komplexnost není možné pokrýt cyklickou komunikací popsanou v normě EN 50 170. Z tohoto důvodu bylo nezbytné definovat acyklickou komunikaci jako rozšíření stávající verze Profibus.

Zařízení mohou volitelně používat nasledující služby:

- Reparametrizaci aplikačního procesu bez prodlevy.
- Volný přístup k jakémukoliv parametru v zařízení.
- Přenos dat proměnné délky.

Z důvodu jednoduchosti je možné tyto služby provádět s danými zařízeními acyklicky souběžně se standartní cyklickou datovou výměnou.

Následující služby jsou specifikovány jako volitelné mezi zařízením typu master 1 a zařízením typu slave. Tyto služby jsou nazývany jako MSAC_C1, nebo-li Master-Slave acyclic communication Class 1.

- Read data set of a slave (DS_Read)
- Write data set of a slave (DS_Write)
- Alarm acknowledgement (Alarm_Ack)

Dále jsou podobně specifikovány volitelné služby mezi zařízením typu master třídy 2 a zařízením typu slave. Tyto služby jsou nazývany jako MSAC_C2. Význam je obdobný jako v předchozím případě Master-Slave acyclic communication Class 2.

- Initiate
- Read the data set of a slave (DS_Read)
- Write the data set of a slave (DS_Write)
- Transport (Data_Transport)
- Abort

Komunikace MSAC_C1 může probíhat pouze tehdy, pokud mezi master a slave zařízeními probíhá i cyklická datová výměna. Vzhledem k tomu, že slave může cyklicky vyměňovat data pouze s jedním zařízením master, může mít vytvořené pouze jedno spojení typu MSAC_C1.

Na druhé straně spojení typu MSAC_C2 může mít slave s několika mastery najednou. Zařízení splňující profil PA 3.0 a vyšší musí podporovat spojení MSAC_C2, ačkoliv spojení MSAC_C1 je pouze volitelné. [2]

2.5 Profibus DP-V2

Profibus verze DP-V2 přidává mnoho nových možností k předchozím verzím protokolu. Především pak za účelem komunikace mezi zařízeními typu slave, synchronizaci času a isochronní komunikaci. Díky těmto možnostem je možné pomocí Profibus vytvořit nové typy aplikací především v oblasti řízení pohonů, profil PROFIdrive, a oblasti bezpečnosti, profil PROFIsafe.

Zajištění isochronního cyklu komunikace na sběrnici (nastává po stejném časovém intervalu) umožňuje uzavřenou řídicí smyčku mezi master a slave zařízením. S odchylkou časování méně než mikrosekunda lze vytvořit aplikaci pro velmi přesné polohování. Komunikace mezi zařízeními typu slave snižuje požadovaný cyklus komunikace mezi slave a master a celkově tak sníží čas odezvy o 60 - 90%. [7]

2.5.1 PROFIdrive

Popis profilu zabírá obsáhlý dokument, na tomto místě je však vhodné krátké nahlédnutí co je v profilu specifikováno. Profidrive definuje šest základních tříd zařízení a k nim jejich popis funkce a vyžadované služby. Jejich přehled je uveden v tabulce 2.3 převzaté z [3].

Dále je vyžadována pro všechny aplikační třídy acyklická komunikace pro parametrizaci, diagnostiku a identifikaci. Snahou v této praci je implementovat, alespoň částečně, třídu 1 a třídu 3.

2.6 GSD soubory

Při konfiguraci řídicího systému, typicky master třídy 1, se musí provést tzv. hardwarová konfigurace. V ní je popsáno jaká zařízení jsou v systému použita, jaké mají adresy a podobné parametry. Protože Profibus je otevřený systém a je žádoucí

Tabulka 2.3: Aplikační třídy profilu PROFIdrive

Nr.	Application class	Interface	Functions
1	Standart drive	n-setpoint torque-setpoint current-setpoint	Cyclic interface
2	Standart drive with distributed technology controller	Technological setpoint-actual values	Cyclic interface with Cyclic data exchange between DU
3	Single axis positioning drive, with local motion control	pos-setpoint run request	Cyclic interface
4	Motion control with central interpolation and speed setpoint interface	n-setpoint x-actual	Cyclic interface Clock synchronous operation
5	Motion control with central interpolation and position setpoint interface	x-setpoint	Cyclic interface Clock synchronous operation
6	Motion control for clocked processes, or distributed angular synchronism	command variables motion instructions	Cyclic interface Clock synchronous operation Cyclic data exchange between DU

interoperabilita zařízení od různých výrobců, existuje ke každému zařízení GSD soubor. Tento ASCII soubor je vytvářen navrhovately fyzických zařízení. V souboru je uveden popis zařízení pomocí definovaných tagů, které jsou vhodné pro čtení programátory zařízení a neobtěžují tak uživatele těžko čitelnými bity a byty.

Pro tvorbu GSD souboru je k dispozici volně stažitelný nástroj z adresy www.profibus.com, který nás provede vytvořením GSD souboru krok za krokem. Na výše uvedené adrese je dostupná také volně stažitelná knihovna GSD souborů většiny vytvořených zařízení. Nevýhodou však je, že zařazení do této knihovny nezaručuje, že je zařízení certifikované.

Kapitola 3

Komunikační obvod VPC3+C

Při návrhu zařízení komunikujícího prostřednictvím rozhraní Profibus DP máme tři možnosti řešení. První možností je implementovat celé rozhraní v mikroprocesoru, který bude disponovat linkou RS-485. Toto řešení má nevýhodu ve velké režii procesoru, která bude použita pro samotnou komunikaci a zajištění vhodného časování. Téměř s jistotou však můžeme prohlásit, že maximální rychlost 12MB bude jen těžko dosažitelná.

Druhou možností je použít zcela autonomní obvod, který má však již předem definovanou funkčnost a možnost vlastní konfigurace a změny funkcionality zařízení je nemožná. Tyto obvody jsou vhodné zejména pro jednoduché aplikace typu digitální vstupy nebo výstupy.

Poslední uvažovanou možností je využití tzv. ASIC obvodu, který má implementovánu část komunikace a zbývající část je ponechána uživateli, aby měl možnost vytvořit aplikaci dle vlastních požadavků. Tento obvod funguje jako řadič sběrnice, ke kterému se přistupuje z nadřazeného mikroprocesoru, takže není schopen pracovat samostatně.

Na trhu existuje více obvodů v této kategorii, především od firmy Siemens. V této práci je však využíváno alternativního výrobce těchto obvodů firmy Profichip a obvodu VPC3+C. V následující kapitole bude popsána funkčnost a možnosti tohoto chipu. V závěru bude představen vývojový starterkit.

3.1 Charakteristika obvodu

Obvod má implementován kompletní řadič Profibus DP, který je řízen tzv. microsequencerem. Automatické rozpoznání komunikační rychlosti až do maximální hodnoty 12MB je samořejmostí. Dále obvod disponuje 4kB paměti Dual-port RAM

oproti 1.5KB obvodu SPC3 od firmy Siemens, se kterým je plně kompatibilní jak funkčností, tak pouzdrem. Obvod může být napájen 3.3V nebo 5V. Pokud je použito napájení 3.3V jsou vstupy tolerantní pro hodnotu 5V. Podrobnější blokové schéma obvodu je znázorněno na obrázku 3.1.

Obvod podporuje komunikační rozhraní s následujícími procesory:

- Intel: 80C31, 80X86
- Siemens: 80C166/165/167
- Motorola: HC11-, HC16-, HC916.

V této práci je využito dalších procesorů, které nejsou uvedeny v seznamu.

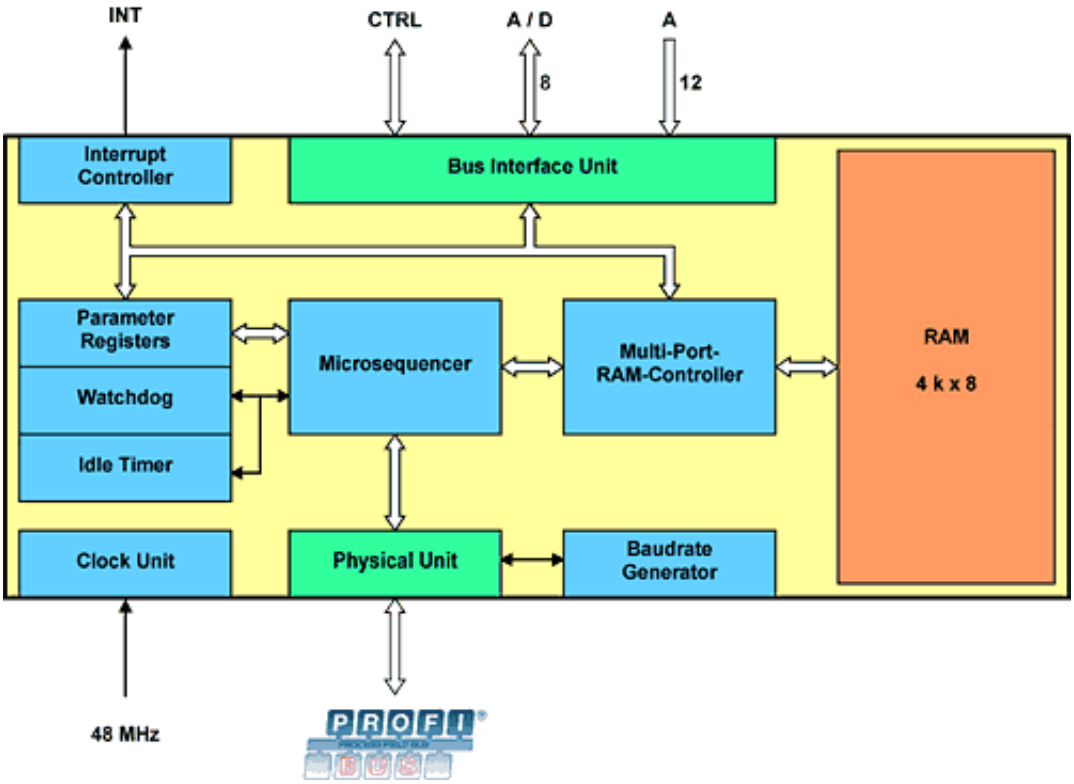
Integrované 4kB Dual-port RAM je využíváno jako prostředníka mezi mikrořadičem a nadřazeným počítačem. Pamět je rozdělena na tři organizační části. První částí jsou řídicí registry. Zde najdeme nastaveni generovaných přerušení. Mode registry, kde jsou uloženy informace o stavu obvodu, například komunikační rychlost, verze obvodu, stav DP automatu, status registry, parametrizační příznaky a konfigurační příznaky. Druhá část je nazývána organizační parametry a jsou v ní uložené především ukazatele a délky na příchozí či odchozí data a také adresa zařízení. Poslední, třetí, částí je samotná pamět, kde jsou uložena příchozí a odchozí data, na která ukazují pointery z předchozí části paměti.

Další popis obvodu jako například detailnější popis ASIC rozhraní nebo rozhraní DP by byl zcela jistě možný, nicméně je nad rámec této kapitoly. Některé podrobnější informace se mohou vyskytnout v kapitole o implementaci software. Detailnější informace lze nalézt v uživatelském manuálu součástky [4].

3.2 Starterkit Profichip

V této kapitole se stručně seznámíme se starterkitem od firmy Profichip, ze kterého vycházela celá práce. Ta postupně došla ke stavu, kdy není v Profibus zařízení použita žádná část z tohoto kitu.

Starter kit je složený ze dvou desek plošných spojů, které je možné oddělit. První deska obsahuje již zmíněný obvod VPC3+C a řadu dalších podpůrných obvodů jako galvanické oddělení pomocí optočlenů nebo elektrickým měničem. Budiče sběrnice RS-485 a taktéž optické rozhraní pro případ, že by byla vyvíjena aplikace využívající jako fyzickou vrstvu optický kabel. Dále je na desce osazeno programovatelné hradlové pole, které však není v našich aplikacích využíváno. Vzhledem k

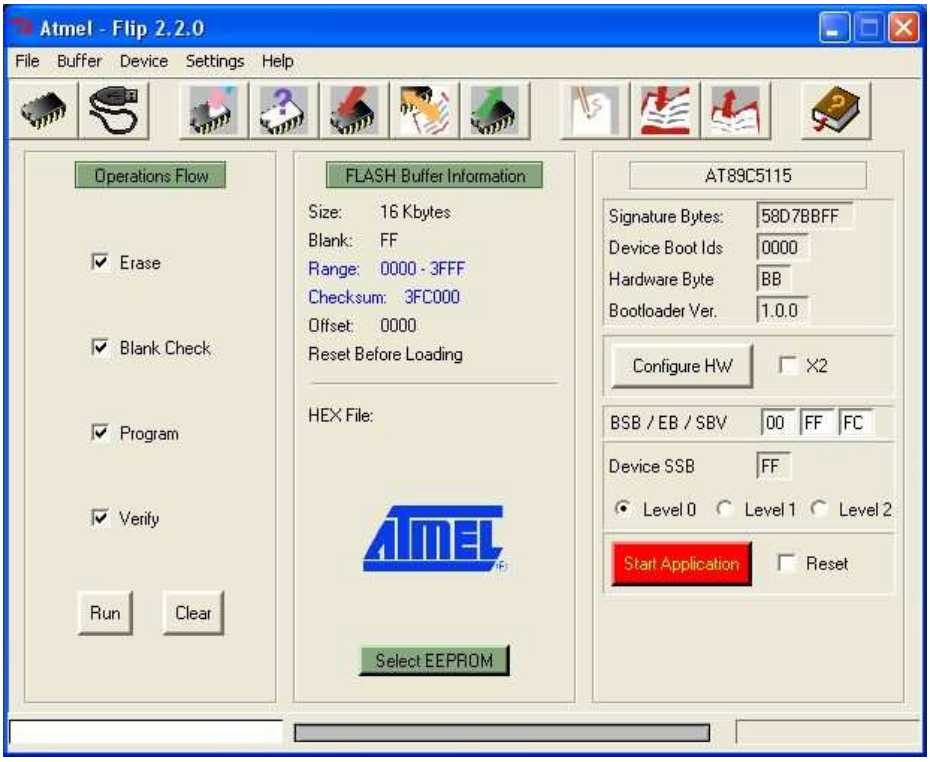


Obrázek 3.1: Blokové schéma obvodu VPC3+C

návrhu umožňujícího široké možnosti nastavení je deska osazena řadou jumperů a propojek, kterými je volen pracovní mód. Potisk označující tyto propojky však obsahuje chyby, které se podařilo po komunikaci s firmou Profichip odstranit. Tato část kitu byla nejvíce využívána při vývoji vlastní aplikace, protože po otestování zaručovala funkční spojení až na fyzickou vrstvu.

Druhá deska kitu byla využívána pouze z počátku práce, pro ověření funkčnosti zařízení a nastavení správné komunikace a HW konfigurace s PLC. Tato deska je osazena mikrokontrolerem AT89C5132 od firmy Atmel. Dále pak dvouřádkovým LCD displayem, třemi DIP přepínači, řadou indikačních diod a čidly teploty či fotosenzorem. Komunikační rozhraní je zajištěno sériovou linkou a převodníkem na USB rozhraní, přes které je obvod programován. S komunikační deskou s obvodem VPC3+C je deska propojena pomocí adresové, datové a řídicí sběrnice. Dále pak samozřejmě všemi napájecími úrovněmi 3,3V, 5V a napájecím napětím, které se může pohybovat mezi 7-24V.

Společně s kitem byl dodán také softwarový stack DP pro uvedený procesor, který byl do procesoru nahrán pomocí aplikace FLIP, která je taktéž součástí dodávky. Aplikace je ale volně stažitelná ze stránek www.atmel.com. Následně je možné apli-



Obrázek 3.2: Grafické rozhraní aplikace Flip

kaci z programu FLIP spustit tlačítkem start application. Grafické rozhraní aplikace je znázorněno na obrázku 3.2. Display zobrazí název aplikace a verzi součástky. Pokud máme před spuštěním pomocí DIP0 přepínače správně nastavenou adresu zařízení, která odpovídá hardwarové konfiguraci, můžeme v PLC sledovat proměnné a nastavovat jejich hodnoty. Některé z nich jsou zobrazovány na indikačních LED a pomocí DIP1 přepínače můžeme simulovat digitální vstupy. Stav DATA EXCHANGE je indikován na komunikační desce svítící zelenou LED.

Pro případ dalšího použítí starterkitu je v následujícím odstavci popsán stručný návod jak aplikaci uvést do chodu a především všechna nastavení jumperů na komunikační desce včetně nalezených chyb.

Nejprve budou uvedeny změny v programu pro PLC Siemens Simatic 315-DP2 0AB0 v.1.2, které je umístěno ve strojovně. Po otevření ukázkového programu, například DPV0AFFE, je potřeba změnit HW konfiguraci systému. Vyměnit stávající PLC za automat uvedený výše a ve vlastnostech změnit hodnotu diagnostické adresy na 1022. Pokud není nainstalován GSD soubor zařízení, tak jej dle standartního postupu nainstalujeme.

Nyní bude popsáno nastavení samotné desky. Nejprve nezapomeneme nastavit DIP přepínačem 0, který je nejblíže k LCD, adresu zařízení, která je nastavena v

Tabulka 3.1: Nastavení propojek na komunikační desce Profichip

Přepínač	Hodnota	Popis
X6,7,8	OPTO	budič RS-485 použije jako zdroj signálu výstup optočlenu
X9,10,11	OPTO	galvanické oddělení optickou cestou
X12 PB Interface	LWL	použít RS-485 jako fyzickou vrstvu, zde je chyba v potisku
X16 MODE	sync	synchronní komunikace s procesorem
X16 Interface	Intel	Intel architektura obvodu

HW konfiguraci v Simatic manageru. Poloha přepínače 1, 2, 3 on a 4-8 off znamená adresu slave zařízení 7, která je standartně v projektu nastavena. Pokud takto adresu nenastavíme před spuštěním programu ve slave zařízení, bude při inicializaci načtena špatná adresa a hlášena chyba bus failed na PLC. Na zbývající desce se nastaví přepínače dle uvedené tabulky 3.1.

Posledním krokem k úspěšnému oživení ukázkové aplikace je nahrání programu do procesoru Atmel. Nejprve zapneme napájení desky a připojíme USB kabel do počítače, v případě užití OS Windows XP, který je nainstalován na notebooku používaném pro analyzátor Profibus, budeme informováni o nalezení nového hardware. Pokud jsme již ovladač instalovali, neměli bychom být informování, avšak systém několikrát upozorňoval na chybějící ovladač. Po spuštění programu FLIP otevřeme komunikaci pomocí rozhraní USB. Je-li spojení navázáno vybereme příslušný obvod AT89C5132 z nabídky součástek. Otevřeme soubor s aplikací dpv0affe.hex a zaškrtneme všechny možnosti v levém panelu (erase, blank check, program, verify) a stiskneme tlačítko run. Po úspěšném nahrání stačí aplikaci spustit červeným tlačítkem start application. Běžící aplikace nás informuje o své verzi a typu pomocí LCD. Pokud budeme testovat tu samou aplikaci později, můžeme vynechat programovací část a stačí aplikaci pouze spustit, jelikož je uložena v paměti flash. Nicméně nutnost počítače ke spuštění programu je malou nevýhodou tohoto starterkitu.

V následující kapitole budou popsány procesory, které byly vybrány pro realizaci vlastního zařízení Profibus DP a některé podpůrné obvody.

Kapitola 4

Řídicí a podpůrné obvody

V této kapitole budou popsány vybrané procesory a jejich periferie využívané v práci. Dále pak některé podpůrné obvody umožňující rozšíření fyzické vrstvy.

4.1 LPC2119

Pro zvolenou aplikaci byly vybrány dva procesory. Prvním procesorem je LPC2119 od firmy Philips, který splňuje předpoklady pro vývoj nízkonákladového zařízení. Na katedře řídicí techniky je již vyvinuta deska s tímto procesorem a realizována řada aplikací.

Procesor je založen na jádře ARM7 a disponuje 64kB flash paměti. Použitý typ je v pouzdře LQFP64. Procesor disponuje řadou standartních periferii jako CAN, UART, časovače, PWM, analogové vstupy a mnoho dalších. V této kapitole však uvedeme pouze popis těch periferii a služeb, které jsou použity ve vlastní aplikaci. Více o dalších rozhraních lze nalézt v manuálu [6].

4.1.1 Fast GPIO

GPIO (General Purpose Input Output), nebo-li obecné vstupy a výstupy, jsou standartní součástí každého mikrokontroleru. Jsou využívány k nejrůznějším účelům od jednotlivých pinů po celé porty. Ve vytvořené aplikaci jsou GPIO použity především pro simulaci adresové a datové sběrnice, kterou bohužel tento mikrokontroler postrádá. Zbytek nepoužitých IO je v ukázkové aplikaci použit pro ovládání krokových motorů robota Oscar.

Přístup ke GPIO portům 0 a 1 je umožněn přes dvě různá rozhraní. Buď standartní pomalé rozhraní nebo rychlé, které bylo zvoleno pro naši aplikaci. Rozdíl je

ve vnitřním zpracování dat a jejich přenosu na fyzický pin. Pokud použijeme rychlé rozhraní získáváme tak možnost až čtyřikrát rychlejšího přístupu. Porty lze konfigurovat pro standartní nebo rychlý přístup po jednotlivých bytech. Konkrétní registry, které slouží pro nastavení, budou popsány v kapitole 6 realizace SW.

4.1.2 UART

Standardní sériové rozhraní UART (Universal Asynchronous Receiver Transmitter) je dalším standartním rozhraním běžného mikrokontroleru. Tento disponuje dvěma UART rozhraními. Použitý je však pouze UART0 k funkci ISP (In System Programming) pro nahrávání aplikace přes sériové rozhraní, které je pomocí obvodu FTDI232 převedeno na rozhraní USB. Po stejném rozhraní je možné po spuštění aplikace komunikovat se standartním terminálem a přijímat tak ladicí hlášky a řídit program pomocí zasílání znaků.

4.1.3 Časovače

Dalším běžným vybavením moderních kontrolerů jsou časovače, které mohou pracovat v mnoha režimech. Především jsou využívány ke generování vnitřního času, PWM a dalších. LPC2119 má k dispozici celkem dva nezávislé 32bitové časovače. V aplikaci je využíván časovač Timer1 za účelem generování vnitřních hodin pro řízení krokových motorků robota.

4.1.4 Systém přerušení

Použitý obvod disponuje poměrně komplexním systémem přerušení, který umožňuje použití ve vektorovém i nevektorovém režimu s konfigurovatelnými prioritami a adresami přerušení. Přerušení jsou samozřejmě maskovatelná. V aplikaci jsou využita přerušení od externího zdroje (pin VPC3+C obvodu) a dále pak přerušení od časovače. Nabízelo by se i užití přerušení seriové linky, to však v aplikaci využito není a UART je obsluhován programově.

4.2 Renesas H8S/2638

Druhým vybraným procesorem, který je určen pro náročnější aplikace, je H8S/2638 od společnosti Renesas. Tento procesor byl opět vybrán z důvodu používání v mnoha aplikacích vyvíjených na katedře řídicí techniky. S tímto obvodem jsou realizovány

některé projekty, avšak navržená deska nebyla pro požadovanou aplikaci vyhovující, proto byla v rámci práce navržena nová deska opravující nedostatky starší verze a doplňující nové požadavky. Pro obě použité desky existují volné vývojové nástroje, které jsou dostupné v repozitáři na rtime.felk.cvut.cz. V této podkapitole však opět budou uvedeny pouze ty části, které jsou nebo je jejich předpoklad využití v uvažované aplikaci.

4.2.1 Externí sběrnice

U levnějších typů jako byl LPC2119 nebývá zvykem, aby byla z pouzdra vyvedena adresová a datová sběrnice pro komunikaci s externí pamětí či jinými periferiemi. Tento obvod však umožňuje vybrané porty použít právě v režimu adresové, datové a řídicí sběrnice. Tato vlastnost byla jedním ze základních předpokladů při výběru procesoru, protože tak umožňuje snadnou portaci Profibus stacku. Sběrnice je konfigurovatelná a může adresovat až osm 2MB bloků, kterým mohou být přiřazeny různé typy přístupů ke sběrnici.

4.2.2 UART

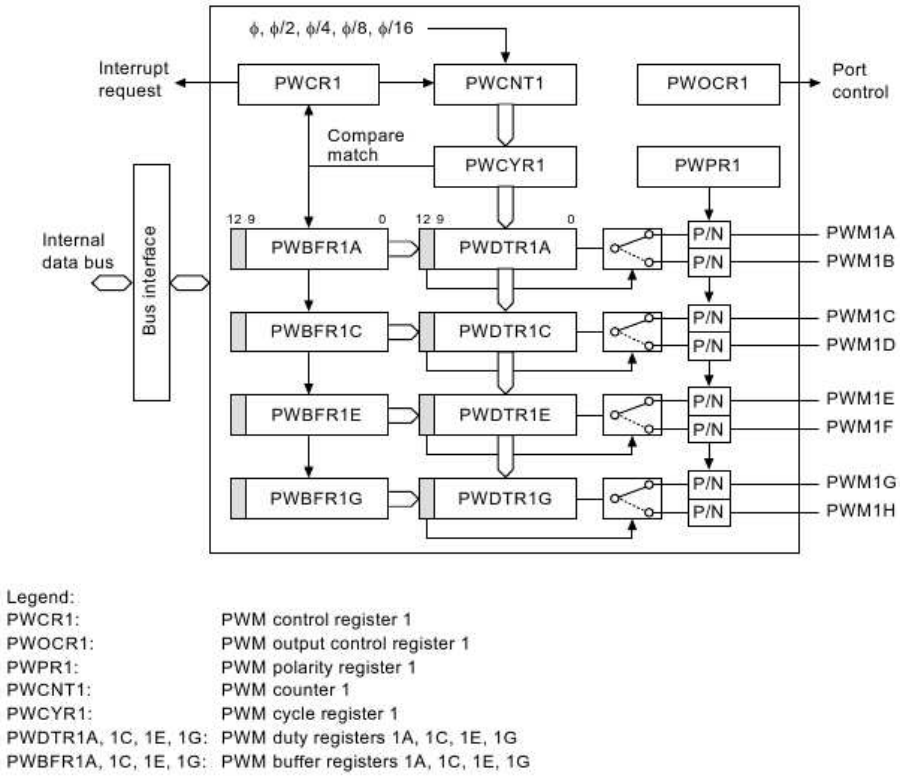
Opět nezbytné rozhraní pro odladění a komunikaci s procesorem prostřednictvím terminálu. Narozdíl od LPC2119 tento procesor umožňuje využití funkce printf, takže je formátovaní a komunikace z pohledu programátora snadnější.

4.2.3 Motor Control PWM Timer

Dalším důležitým faktorem pro výběr tohoto obvodu byl fakt, že disponuje jednotkou určenou k řízení motorů. Tato jednotka disponuje dvěma 10-ti bitovými PWM jednotkami, kde každá má 8 výstupů. Střída a polarita je nastavitelná pro každý výstup samostatně. Více je znázorněno na blokovém schématu z manuálu obvodu [5] na obrázku 4.1

4.2.4 Řadič přerušení

Tento obvod je opět vybaven sofistikovaným systémem přerušení, který je v aplikaci použit pro generování přerušení od časovače a externího vstupu. Ten je spojen s VPC3+C. Přerušením lze přiřadit priority a adresy přerušení. Samozřejmostí je možnost volby spádové či náběžné hrany nebo úrovně vstupu.



Obrázek 4.1: Blokové schéma části Motor Control PWM Timer

4.3 Siemens SIM1-2

V průběhu práce došlo k rozhodnutí, že by bylo vhodné navrhovanou desku osadit rozhraním pro Profibus PA, které může být realizováno pomocí obvodu SIM1-2 od firmy Siemens. Tento obvod zprostředkovává interface mezi běžným zákaznickým obvodem, jako například VPC3+C, a rozhraním Profibus PA na komunikační rychlosti 31.25kb/s dle normy IEC 61158-2.

Obvod je parazitně napájen ze sítě a umožňuje pracovat v konfiguraci, kdy jsou signály od komunikačního obvodu galvanicky odděleny a mohou být napájeny přes měnič, který je řízen vnitřním kontrolérem v obvodu SIM1-2. Obvod tak má vlastní oscilátor a několik páru RC členů, které nastavují citlivost a funkčnost obvodu. Jejich hodnoty je nutné odladit po vytvoření zařízení. Jejich hodnota je výrobcem doporučená pouze rozsahem hodnot. Tento obvod se nyní vyrábí v pouzdře MLPQ40.

Kapitola 5

Realizace hardware

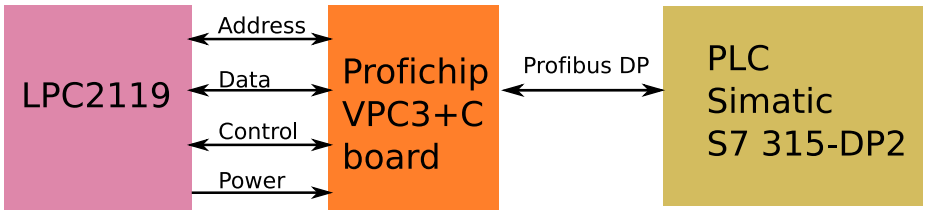
V této kapitole bude popsán návrh všech zařízení a významnějších úprav, které při práci vznikly. Popis desky LPC2119 zde není uveden z důvodu, že její návrh nebyl součástí této práce. Pro úplnost je uvedeno pouze schéma zapojení v příloze B.1.

Návrh hardware byl v průběhu práce vyvíjen tak, aby zaručoval jistou funkčnost po vytvoření podkladů a výrobě DPS. Proto bylo zpočátku navrženo pouze propojení stávajích funkčních bloků znázorněných na obrázku 5.1. Tento postup zaručoval odladění funkce a urychlení vývoje, protože nemusel být nejprve prováděn návrh a oživení desky s nejistým výsledkem.

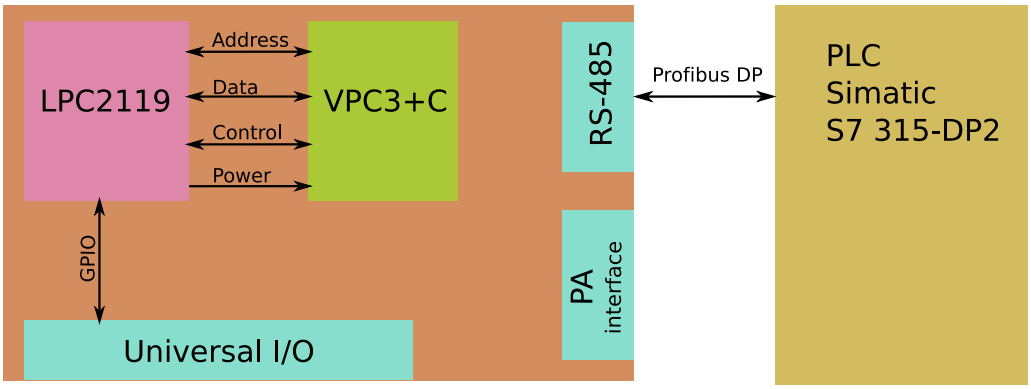
5.1 Deska PB-LPC-VPC

Dalším krokem byl návrh samostatného zařízení s obvodem VPC3+C, které vychází ze zapojení VPC3+C části starterkitu tak, aby bylo možné vytvořit kompaktní zařízení vhodné pro další vývoj a zároveň umožňovalo implementaci libovolné funkčnosti za použití neobsazených pinů procesoru. Blokový návrh řešení je znázorněn na obrázku 5.2.

Obrázek 5.3 znázorňuje rozmístění součástek na navržené desce. Kromě obvodu



Obrázek 5.1: Blokové schéma propojení LPC2119 s částí VPC Profichip kitu



Obrázek 5.2: Blokové schéma navržené desky s VPC3+C

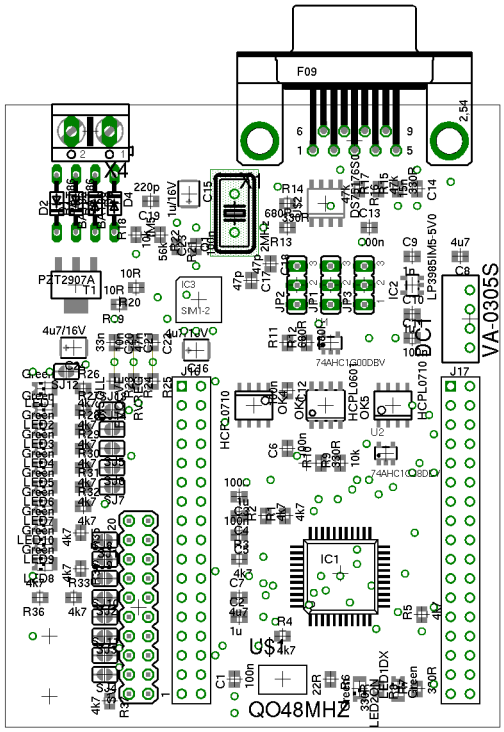
VPC3+C jsou na desce osazeny rychlé optočleny HCPL0710 a HCPL0601, které jsou schopny přenášet data rychlostí až 12MB/s. V signálové cestě za těmito optočleny a příslušnými budiči jsou vloženy propojky JP1, JP2 a JP3, které slouží k volbě rozhraní mezi standartním RS-485 nebo směrují signál do obvodu SIM1-2. Jak již bylo zmíněno v předchozí kapitole, tento obvod slouží jako budič pro rozhraní typu Profibus PA.

Konektory J16 a J17 odpovídají svým protějškům na desce s obvodem LPC2119. Vedle konektoru J16 nalezneme část, která může plnit funkci testovacích vstupů prostřednictvím dvou tlačítek a výstupů indikovaných osmi zelenými LED. Tyto vstupy a výstupy je nutné připojit pájecími propojkami, protože ve výchozím stavu jsou odpojeny a jim odpovídající piny jsou přivedeny na konektor SV1 sloužící k připojení dalších funkčních zařízení.

Tabulka B.1 popisuje propojení příslušných pinů obvodu VPC3+C a desky s procesorem LPC2119. Tabulka B.2 znázorňuje volné piny použitelné k libovolné funkci na desce PB-LPC-VPC.

Vyrobená deska plošných spojů byla osazena a úspěšně oživena. Nedostatkem je chybějící propojení vývodů R23 s vývodem C20, R24 s C21 a R25 s C22. Tato chyba je však velmi snadno opravitelná při osazení součástek.

Kromě otestování desky při simulaci vstupů a výstupů pomocí terminálu, byla deska použita k vytvoření Profibus DP rozhraní pro výukového robota Oscar. Více je popsáno v samostatné sekci 5.3. Kompletní schéma B.2, plošné spoje B.4 B.5 a osazovací plán B.6 jsou uvedeny v příloze B.2.



Obrázek 5.3: Rozmístění součástek na navržené desce PB-LPC-VPC

5.2 Renesas H8S2638 board

Při počátku projektu se počítalo s využitím již navržené a vyrobené desky používané v soutěži Eurobot. V průběhu řešení však došlo ke zjištění, že hotové desky jednak nejsou k dispozici a je vhodné upravit na nich stávající nedostatky a za druhé, že původní deska nemá vyvedenou nutně potřebnou adresovou a datovou sběrnici na jakékoliv konektory desky. Proto došlo k návrhu nové desky splňující požadavky pro použití v zamýšlené aplikaci a zároveň její zdokonalení v oblastech, které jsou uvedeny v následujícím odstavci.

Deska byla navržena stejně jako předchozí deska PB-LPC-VPC v programu Eagle ve verzi free a vyrobena ve dvou kusech ve firmě Pragoboard. Oproti původní desce tedy byly provedeny následující úpravy.

- Přidány konektory s adresovou, datovou a řídicí sběrnicí. Ostatní konektory určené pro řízení motorů byly ponechány.
- Upraven resetovací obvod, který byl nahrazen obvodem MAX811.
- Opraveno pouzdro diody BAT99.
- Přidány perličkové cívky do napájecí cesty procesoru.

- Připojení resetovací časti na výstup RTS obvodu FTDI.
- Vyměněn stabilizátor napětí za moderní obvod LF50.
- Upraveno pouzdro krystalu z typu SMD na běžné pouzdro s roztečí 5mm.

Přesto však při návrhu vznikla nová chyba v podobě špatného rozmístění vývodu na resetovacím obvodu MAX811. Chyba vznikla záměnou dvou signálu při vytváření součástky v knihovně návrhového systému. Tato chyba byla odhalena při osazování desky ještě dříve, než došlo k pokusu o oživení, takže nebyly způsobeny žádné škody.

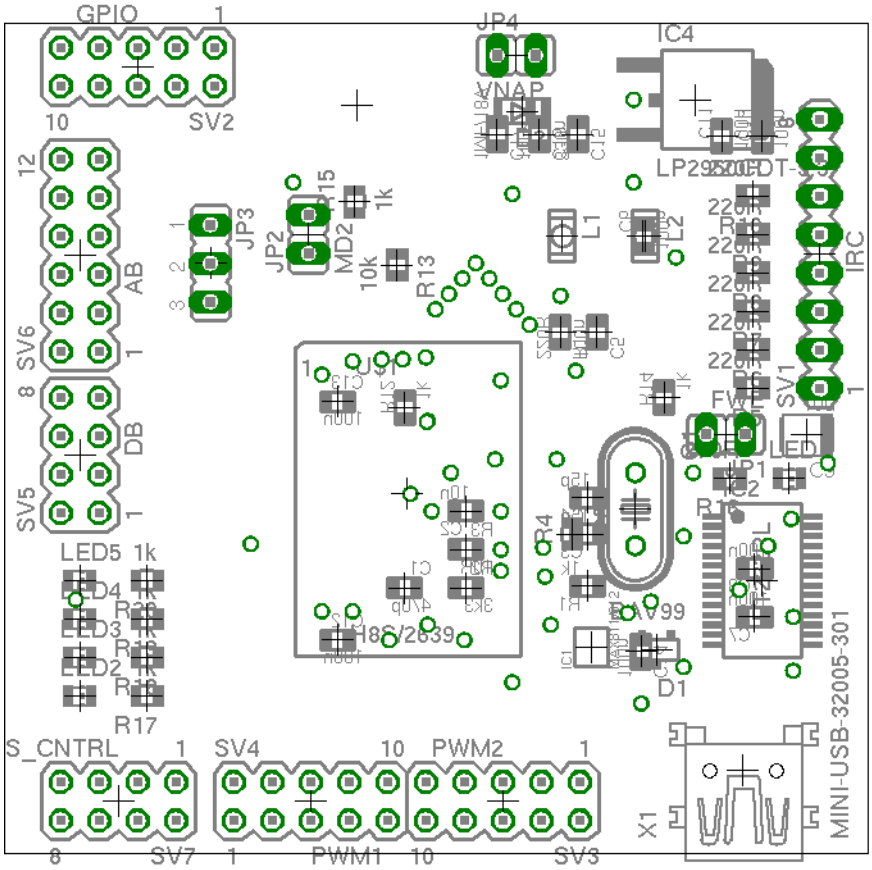
Obrázek 5.4 znázorňuje výsledný navrh desky H8S2638, kde je stále výše uvedená chyba. Kompletní schéma B.7, plošné spoje B.8 B.9 a osazovací plán B.10 jsou uvedeny v příloze B.3.

Tabulka B.3 popisuje propojení příslušných pinů obvodu VPC3+C a desky s procesorem H8S2638. Procesor komunikuje s obvodem VPC3+C v asynchronním režimu Intel. Datová sběrnice využívá při osmibitovém přístupu pouze horní část šestnáctibitové sběrnice, proto je propojení provedeno tak, jak je uvedeno v již zmíněné tabulce. Signál ready nemá na procesoru ekvivalent, proto v zapojení není použit. V komunikaci je použit maximální počet wait state, kdy by měla být data na sběrnici validní. Konektory J17-IRC, J23-PWM2 a J24-PWM1 odpovídají svým protějškům na desce použité na robotu Eurobot. Jejich zapojení bylo ponecháno v původní podobě, avšak z důvodu přidání konektorů adresové a datové sběrnice nemohly být dodrženy původní rozměry desky, takže konektory nyní nejsou umístěny přesně proti sobě a na jejich propojení je třeba použít kabelů.

Z důvodu nepřítomnosti napěťové úrovně +3.3V na desce Renesas musela být kvůli propojení s deskou Profichip VPC3+C board provedena úprava tak, aby byla získána i tato napájecí úroveň. Jak již bylo popsáno ve vlastnostech obvodu VPC3+C, tento obvod je vybaven tolerantními vstupy pro 5V, avšak deska Profichip je osazena napěťovým měničem z úrovně 3.3V na 5V. Tento měnič je použit pro galvanické oddělení budiče RS-485 samotného ASIC obvodu. Vzhledem k uváděné toleranci vstupu 10 % byl použit obvod LE33, který vytvořil požadovanou úroveň z napětí +5V.

5.3 Robot Oscar

V rámci předmětu Řídicí systémy se používá při jedné z úloh malý robot Oscar, který byl doposud řízen z PLC přímo kartou digitálních vstupů a výstupů podle



Obrázek 5.4: Rozmístění součástek na navržené desce H8S2638

zvoleného režimu. V rámci této práce byl nedestruktivní vratnou cestou upraven hardware robota tak, že nyní je možné připojit výstup robota přímo do realizované desky PB-LPC-VPC, která ovládá robota v tzv. kopírovacím režimu, více lze nalézt v dokumentaci robota Oscar. Robota je možné ovládat ve třech různých režimech, avšak pouze kopírovací režim byl z pohledu řízení a požadavku minimálního počtu použitých vstupů a výstupů pro realizaci aplikace přípustný.

Robot Oscar je vybaven čtyřmi krokovými motory. Tři motory ovládají jednotlivá ramena a tím pohyb robota v prostoru a čtvrtý motor je použit pro ovládání čelisti nástroje. Bohužel v době testování robota byl jeden výkonový člen nefunkční, takže bylo možné ovládat pouze tři motory zároveň.

Uvedený kopírovací režim funguje velice jednoduše a je vysvětlen zároveň pomocí tabulky 5.1, která popisuje konektor vyvedený z robota po jeho úpravě. Chceme-li provést krok s příslušným motorem, je třeba zvolený motor aktivovat pomocí signálu motor x enable, vybrat směr vpřed či vzad pomocí motor x direction a krok bude proveden při spádové hraně na signálu trigger. Takto lze pro každý krok nastavit

Tabulka 5.1: Popis konektoru z robota Oscar do konektoru SV1

Pin	Signál	Popis
1	VNAP	Napájecí napětí
2	+5V	+5V
3	D7	Motor 4 direction
4	D6	Motor 4 enable
5	D5	Motor 3 direction
6	D4	Motor 3 enable
7	D3	Motor 2 direction
8	D2	Motor 2 enable
9	H4	Home 4 - optická závora od motoru 4
10	H3	Home 3 - optická závora od motoru 3
11	H2	Home 2 - optická závora od motoru 2
12	H1	Home 1 - optická závora od motoru 1
13	D1	Motor 1 direction
14	D2	Motor 1 enable
15	INT	Trigger
16	LED D2	nepoužito
17	CANL	nepoužito
18	CANH	nepoužito
19	GND	GND
20	+3.3V	+3.3V

směry, vybrat příslušné motory a poté všechny kroky vybraných motorů aktivovat naráz. V aplikaci je taktéž využito signálu enable pro regulaci rychlosti, kdy může být libovolný počet kroků (po odladění maximálně 7) vypuštěno, zatímco ostatní motory své kroky provádějí. Více o implementaci je uvedeno v kapitole 6.6 realizace software.

Připojení ke stávájícímu hardwaru bylo provedeno pomocí dutinkových konektorů, které byly vloženy mezi desku procesoru LPC2119-oscar a desku robota, která galvanicky odděluje signály motorů, PLC a mikrokontroléru. Z těchto dutinkových konektorů se odvádí pomocí plochého kabelu požadované signály na desku PB-LPC-VPC. Aby nedošlo k možným propojením nepřípustných napětí z PLC a signálů pro procesor LPC, jsou vstupní piny vzhledem k robotovi na přidaném konektoru zkráceny tak, že nedochází k propojení s deskou in-board. Takto zůstávají vstupy a

výstupy z PLC nepřipojeny a nejsou využívány. Pokud by byl požadavek opět robota řídit přímo z PLC, stačí odebrat přidané dutinkové konektory. Výstupní signál z čidel home má úroveň +5V, což je pro tolerantní vstupy LPC2119 přípustné i při použitém napájení +3.3V.

Kapitola 6

Realizace software

Tato kapitola popisuje nejrozsáhlejší činnost práce, tedy portaci programu od firmy Profichip na použité procesory. V kapitole realizace HW 5 bylo popsáno jak jsou procesory a obvod VPC3+C fyzicky propojeny. Na začátku této kapitoly budou popsány vývojové nástroje, vybraný typ komunikace a její průběh a portace programu na použité platformy.

6.1 Vývojové nástroje

Celá práce byla realizována v OS Linux, distribuci Kubuntu. Pro programování a překlad aplikací do obou použitých procesorů byly použity otevřené vývojové řetězce z projektu OMK. Program pro obvod LPC2119 využívá framework sysless-lpc21xx, který je dostupný a dokumentovaný na stránkách http://rtime.felk.cvut.cz. Jeho ovládání bylo prováděno z příkazové řádky volání příkazu *make* a *make load-flash*. Pro komunikaci byl použit terminál gtkterm, který umožňuje ovládat piny *drs* a *rts* použité pro reset obvodu.

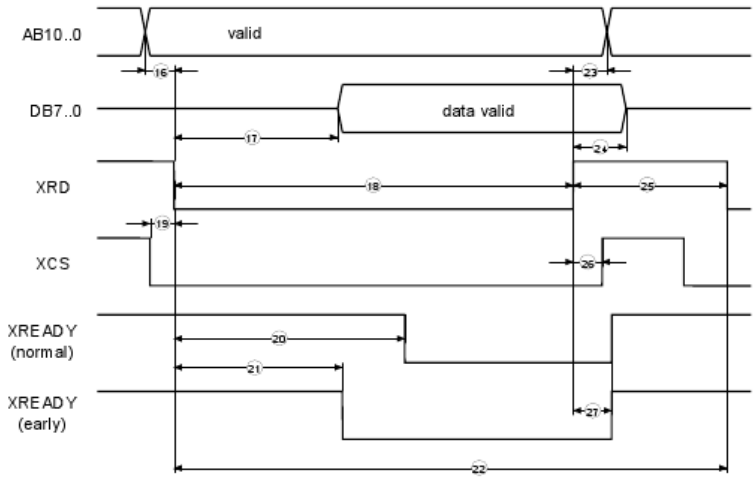
Procesor Renesas využívá obdobných nástrojů, které jsou na zmíněném webu vedeny pod názvem sysless-h8300. U tohoto obvodu bylo třeba navíc nahrát bootloader, který byl ve staženém balíčku, avšak nastaly komplikace, které se podařilo vyřešit až použitím starší verze nástrojů. Tato verze je dostupná ve zmíněném repozitáři pod tagem bootstrap-ok. Pro editaci kódu byla používána aplikace Kdevelop.

6.2 Časování na sběrnici

Obvod VPC3+C umožňuje pomocí dvou pinů typ a mode zvolit jednu ze čtyř možných komunikací, které jsou uvedeny v tabulce 6.1.

Tabulka 6.1: Režim komunikace VPC3+C

Úroveň pin type	Úroveň pin mode	Režim komunikace
0	0	Intel asynchronní
0	1	Intel synchronní
1	0	Motorola asynchronní
1	1	Motorola synchronní



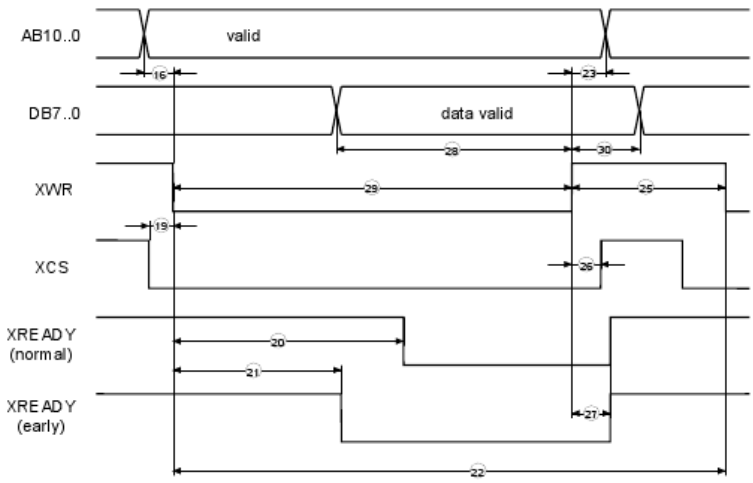
Obrázek 6.1: Průběh časování při čtení z VPC3+C.

Pro zvolenou aplikaci byl vybrán režim Intel s asynchronním časováním. V tomto módu je obvodem VPC3+C generován signál READY, který potvrzuje validní data na datové sběrnici při čtení nebo jejich úspěšné uložení v režimu zápisu. Obrázky 6.1 a 6.2 znázorňují průběh vybraného režimu komunikace. V tabulce 6.2 jsou uvedeny mezní hodnoty vztahující se k těmto průběhům. Obrázky a tabulka jsou převzaty z [4].

6.3 Popis komunikačního stacku Profichip

Společně se starterkitem od firmy Profichip je dodáváno CD s dokumentací a zdrovovým kódem automatu DPV-0 a DPV-1 pro procesor Atmel, založený na architektuře x51. Proto bude v této kapitole uvedena struktura programu. V následujících sekcích budou detailněji popsány provedené úpravy při portaci na vybrané architektury.

Detailní popis každého souboru nemá v této práci význam a pokud zájemce



Obrázek 6.2: Průběh časování při zápisu do VPC3+C.

Tabulka 6.2: Mezní hodnoty časování při čtení a zápisu z/do VPC3+C.

No.	Parameter	VDD = 3.3 V		VDD = 5 V		Unit
		MIN	MAX	MIN	MAX	
16	address-setuptime to XRD / XWR ↓	0		0		ns
17	XRD ↓ to data valid		103		97	ns
18	XRD pulsewidth	115		115		ns
19	XCS ↓ setuptime to XRD / XWR ↓	0		0		ns
20	XRD ↓ to XREADY ↓ (Normal-Ready)		132		126	ns
21	XRD ↓ to XREADY ↓ (Early-Ready)		111		105	ns
22	XRD / XWR cycletime	125		125		ns
23	address holdtime after XRD / XWR ↑	0		0		ns
24	data holdtime after XRD ↑	4	16	4	13	ns
25	read/write inactive time	10		10		ns
26	XCS holdtime after XRD / XWR ↑	0		0		ns
27	XREADY holdtime after XRD / XWR	6	21	5	16	ns
28	data setuptime to XWR ↑	10		10		ns
29	XWR pulsewidth	83		83		ns
30	data holdtime after XWR ↑	10		10		ns

bude chtít pochopit celou funkčnost programu, musí do jisté míry kód nastudovat ze zdrojových souborů. Proto bude v této kapitole popsána struktura a jednotlivé soubory s jejich stručným popisem. Soubory můžeme rozdělit do tří skupin:

- DP-INC, kde jsou zejména konfigurační soubory pro různé aplikace,
- DPV0-DRV, do které patří soubory nutné pro funkci automatu DPV0,
- DPV1-DRV, kde nalezneme zdrojový kód pro funkce Profibus DPV1.

Do první skupiny DP-INC patří tyto soubory.

- `dp_cfg.h` - tento soubor je individuální pro různé aplikace a zahrnuje konfiguraci VPC3+C a samotnou aplikaci.

- `dp_inc.h` - zde jsou hlavičky pro celý systém.
- `platform.h` - obsahuje konfiguraci týkající se především použité platformy, datových typů, kompilátoru a mapování adres.

Druhá skupina DPV0-DRV zahrnuje následující soubory nutné pro funkci každého slave zařízení.

- `dpl_list.h` - makra pro double pointered list.
- `dp_if.h` - obsahuje definice a struktury popisující VPC3+C.
- `dp_if.c` - základní funkce pro práci s obvodem VPC3+C.
- `dp_isr.c` - funkce volané při přerušení od VPC3+C.

Třetí skupina DPV1-DRV zahrnuje soubory zajišťující funkci DPV1. Tyto soubory nejsou potřeba, pokud je implementována pouze verze automatu DPV0.

- `dp_fdl.c` - základní ovladač pro verzi 1.
- `dp_msac1.c` - funkce pro acyklickou komunikaci třídy 1.
- `dp_msac2.c` - funkce pro acyklickou komunikaci třídy 2.

Dále je pak třeba soubor `dp_user.c`, který je specifický pro každou aplikaci. V něm je kromě jiného implementována její funčnost, respektivě volány funkce z modulů, které provádějí dané úkoly. Takto byla například implementována ukázková aplikace robota Oscar.

6.4 Portace programu pro LPC2119

Tato podkapitola uvede vzniklé problémy a jejich řešení při portaci kódu pro procesor LPC2119. Jak již bylo zmíňeno v předchozích kapitolách, procesor LPC2119 byl vybrán za účelem návrhu nízkonákladové aplikace. Procesor však postrádá vyvedení adresové a datové sběrnice z pouzdra, takže její funkce musela být nahrazena využitím standartních GPIO portů.

Snahou v počátku úpravy kódu bylo napsat aplikaci tak, aby využívala původního zdrojového kódu a pouze nastavením v souboru `platform.h` došlo k výběru procesoru a správnému přeložení. Avšak zcela záhy bylo zjištěno, že tento postup nebude pro úpravu aplikace vhodný, protože zásah do kódu je až příliš veliký. Proto došlo

k rozhodnutí upravit kód samostatně tak, aby byl vytvořen Profibus stack pro procesory s chybějící externí sběrnicí. V průběhu práce došlo ke komunikaci s firmou Profichip a jejich informování o nově vyráběné variantě obvodu VPC3+, která podporuje komunikaci po UART rozhraní. Tato zpráva byla pozitivní z pohledu, že upravovaný program je téměř ihned použitelný pro komunikaci s tímto obvodem po rozhraní UART.

6.4.1 Náhrada externí sběrnice pomocí GPIO

Nepřítomnost sběrnice byla nahrazena funkcemi `vpc_read()` a `vpc_write`, které budou popsány a doplněny ukázkou kódu. Po restartu procesoru dojde k volání inicializačních funkcí, které nastavují příslušné periferie. Mezi nimi je i funkce `init_vpc()`, která nastaví přislušné piny k přístupu z režimu fast GPIO a na závěr provede reset obvodu VPC3+C. Tyto funkce byly společně s dalšími funkcemi, které budou buď pouze zmíněny nebo popsány dale, implementovány do souboru `vpc_if.c` a jeho hlavičkového souboru `vpc_if.h`.

Funkce `vpc_read()` zajišťuje čtení jednoho bytu z obvodu VPC3+C. Tuto funkci využívá funkce `vpc_read_word()`, která vrací namísto typu uint8_t hodnotu word, když volá funkci dvakrát za sebou a automaticky inkrementuje adresu. Funkce pro čtení je uvedena ve zkrácené verzi bez ladicích řádků. Její funkce by měla být zřejmá z uvedených komentářů. Pro úplnost uvedeme, že registry FIO0SET nastaví do logické úrovně 1 ty piny, na které je zapsána hodnota 1. Ostatní ponechá bez změny. Stejně tak funguje registr FIO0CLR, který však vybrané piny nuluje. `VPC_DB_DIR` je makro nastavující piny nakonfigurované jako datová sběrnice do režimu vstupu či výstupu. Funkce `vpc_rdy_get()` vrací hodnotu pinu READY.

```
uint8_t vpc_read(uint16_t address)
{
  int i = 0;
  uint32_t vpc_address_neg \
  =  vpc_ab_prepare(~address & VPC_AB_WIDTH_m);
  uint8_t data;
  FIO0SET = VPC_AB_MASK << VPC_AB_SHIFT; // set all ab bits
    to high level
  FIO0CLR = vpc_address_neg << VPC_AB_SHIFT; //set zero bits
  VPC_DB_DIR = 0x00; // all DB pins are inputs
  vpc_cs_off();
```

```
  vpc_rd_off();
  while(vpc_rdy_get()) // wait for ready signal
  {
    if(i > 100)
    {
      uart0SendStr("\nvpc_read->timeout-wait for ready signal");
      break;
    }
    i++;
  }
  data = VPC_DB_PIN;
  vpc_rd_on();
  vpc_cs_on();
  return data;
}
```

Funkce `vpc_ab_prepare()` upravuje zvolenou adresu tak, aby bylo možné provést zápis celé adresy najednou, přestože nejsou dané piny řazeny na použitém portu za sebou.

```
uint32_t vpc_ab_prepare(uint16_t address)
{
    return ((address & ~0x003F)<< 2) | (address & 0x003F);
}
```

Funkce `vpc_write()` funguje velice obdobně jako funkce pro čtení. Má však navíc parametr data, která mají být do obvodu zapsána.

```
void vpc_write(uint16_t address, uint8_t data)
{
  int i = 0;
  uint32_t vpc_address_neg \
  = vpc_ab_prepare(~address & VPC_AB_WIDTH_m);
  FIO0SET = VPC_AB_MASK << VPC_AB_SHIFT; // set all ab bits
    to high level
  FIO0CLR = vpc_address_neg << VPC_AB_SHIFT; //set zero bits
  VPC_DB_DIR = 0xFF; // all DB pins are outputs
  VPC_DB_PIN = data;
```

```
  vpc_cs_off();
  vpc_wr_off();
  while(vpc_rdy_get())
  {
    if(i > 100)
    {
      uart0SendStr("\nvpc_write->timeout-wait for ready signal");
      break;
    }
    i++;
  }
  vpc_wr_on();
  vpc_cs_on();
}
```

Pro ověření, že komunikace mezi procesorem a obvodem VPC3+ v kterékoliv variantě funguje správně, je implementována funkce `vpc_memory_test()`. Ta provede zápis od adresy 0x16 až po konec uživatelské paměti a následně čtení ze stejného rozsahu. Pokud jsou hodnoty totožné, odešle výsledek o bezchybnosti na uart0, jinak jsou vypsána chybová hlášení.

Zasadním problémem při přepisování aplikace bylo, že v původním kódu je přistupováno do paměti VPC3+C, zcela logicky, pomocí ukazatelů. Avšak při našem použití je třeba přesně vědět, zda proměnná leží v paměti procesoru nebo je uložena ve VPC3+C a přístup k ní musí být ošetřen pomocí funkcí `vpc_read` a `vpc_write`. Navíc pokud jsou ukazetele ukládány do struktur uložených v procesoru, je nutné upravit proměnnou z typu pointer na proměnnou typu word, která představuje absolutní adresu v paměti VPC3+C.

6.4.2 Funkce pro práci s pamětí

V programu je mnohokrát využíváno funkcí `memcpy`, `memset` a `memcmp`, které mají jako parametry ukazatele do paměti a délku v bytech. V našem případě, kdy je přístup omezen pomocí funkce čtení a zápisu, však tyto funkce nemají význam a museli být implementovány jejich náhrady. Pro stručnost jsou uvedeny pouze hlavičky funkcí.

```
int vpc_memcmp_lpc_vpc(uint8_t *lpc_mem, uint16_t vpc_mem, uint8_t n)
```

Tato funkce porovnává obsah paměti v procesoru LPC2119 a v paměti VPC3+C. Pokud jsou hodnoty všech bytu stejné, tak funkce vrátí 0, pokud jsou různé, vrátí nenulovou hodnotu.

```
void vpc_memcpy_sm(uint16_t s1, uint8_t *s2, uint8_t n)
void vpc_memcpy_ms(uint8_t *s1, uint16_t s2, uint8_t n)
```

Funkce `vpc_memcpy_sm` kopíruje data z paměti slave do paměti master z pohledu Profibus, takže z paměti LPC2119 do VPC3+C. Zatímco funkce `vpc_memcpy_ms` kopíruje data z paměti VPC3+C do paměti procesoru.

```
void vpc_memcpy_in(uint16_t dest, uint16_t src, uint8_t n)
```

Poslední funkce nahrazující standartní funkci `memcpy` je `vpc_memcpy_in`, která provádí kopírování dat v paměti VPC3+C. Ve všech uvedených funkcích jsou využívány funkce `vpc_read` nebo `vpc_write`.

6.4.3 Ostatní funkce a konfigurace

Kromě funkcí pro čtení, zápis a práci s pamětí jsou v knihovně implementovány funkce pro ovládání resetu (`vpc_reset_on`, `vpc_reset_off`) a chip selectu (`vpc_cs_on` a `vpc_cs_off`). Podobně funkce pro ovládání pinu read, write a čtení hodnoty pinu ready. Nakonec funkce pro povolení a zakázání externího přerušení.

Konfigurace v případě jiného hardwarového připojení, než je použito ve vytvořené aplikaci, je provedeno v souboru `vpc_if.h`, kde se přiřadí adresové a datové sběrnici odpovídající porty. V případě složitějšího rozdělení adresové sběrnice je nutno upravit funkci `vpc_ab_prepare`. Řídicím signálům pak ve stejném souboru přiřadíme čísla pinů. Tak byla zajištěna snaha o maximální oddělení závislosti na použité platformě a konfiguraci pouze jednoho modulu.

6.5 Portace programu pro H8S2638

Portace programu do procesoru Renesas je o poznání snadnější díky externí adresové a datové sběrnici. Program by měl být teoreticky použitelný po úpravě souboru `platform.h` a konfiguraci vlastního procesoru pro správnou funkčnost externí sběrnice. Všechny struktury musely být doplněny o atribut *packed*, který zajistí správné zarovnání struktury tak, aby její prvky následovaly hned za sebou a mapování tak bylo správné. Dále je nutné zmínit, že daný procesor nemá ve svém

rozhraní signály chip select a ready. Vzhledem k tomu, že na externí sběrnici je připojeno pouze jediné zařízení, můžeme signál chip select nastavit na počátku aplikace a tím mít vybránu součástku po celou dobu jejího běhu. Vstupní signál *ready* však takto jednoduše nahradit nelze, proto je ponechán bez připojení a v nastavení sběrnice je vložen maximální počet wait state při čtení i zápisu, aby bylo zajištěno, že data na sběrnici při jejich vzorkování budou validní. Zde se několikrát objevil problém, že i maximální počet wait state někdy nestačil k zajištění správné funkce. Tato chyba nastala při testu paměti při spouštění aplikace, nejčastěji na adresách typu x00h. Tento problém je nadále monitorován a je snaha o jeho vyřešení.

Nyní bude následovat ukázka z konfuračního souboru `platform.h`, který byl doplněn o překladač GCC a desku HitachiBoard.

```
#ifdef GCC_H8300_ELF
  #define UBYTE                 volatile uint8_t
  #define UWORD                 volatile uint16_t
  #define BOOL                  UBYTE
  #define ULONG                 volatile uint32_t
  #define PTR_ATTR
  #define VPC3_PTR              PTR_ATTR *
  #define VPC3_ADR              ULONG
  #define VPC3_UNSIGNED8_PTR    UBYTE PTR_ATTR *
  #define NULL_PTR  (void VPC3_PTR)0
  #define ROMCONST__            const
  #define NOP__()
#endif
```

Dále je pak nutné správně namapovat obvod VPC3+C, který je v našem případě namapován do segmentu 2 na adresu 40000h.

```
#define VPC3_ASIC_ADDRESS   ((volatile uint8_t *)0x40000L)
```

Aby byla zajištěna co největší kompatibilita a minimální zásah do kódu, byla dále předefinována makra pro ovládání dostupných indikačních LED, ovládání pinu reset pro VPC3+C a podobně pinu chip select.

```
#ifdef GCC_H8300_ELF
  #define VPC3_RESET__          (*DIO_PEDR |= (PEDR_PEODRm << 4))
  #define VPC3_CLEAR_RESET__    (*DIO_PEDR &=~(PEDR_PEODRm << 4))
  #define VPC_CS_ON             (*DIO_PEDR |= (PEDR_PEODRm << 5))
```

```
  #define VPC_CS_OFF            (*DIO_PEDR &=~(PEDR_PE0DRm << 5))
#endif

#ifdef GCC_H8300_ELF
  #define CLR_LED_RED__        DEB_LED_OFF(2)
  #define SET_LED_RED__        DEB_LED_ON(2)
  #define CLR_LED_YLW__        DEB_LED_ON(0)
  #define SET_LED_YLW__        DEB_LED_ON(0)
#endif
```

Za zmínění zcela jistě stojí poznámka k VPC3_ADR, která je pro původní kód definována pouze jako WORD a bez této změny program nefunguje. Vysvětlení je logické, avšak jeho odhalení nebylo zprvu zřejmé. V procesoru s architekturou x51 je přistupováno ke vnější paměti, v našem případě VPC3+C, pomocí klíčového slova xdata, v definici jako PTR_ATTR. V moderních architekturách však toto rozlišení pamětí nebývá časté a obvod je namapován do paměti na definovanou adresu, jako by byl součástí obvodu. Vnitřní logika pak sama řeší, zda je přístup do vnitřní nebo vnější paměti. Protože obvod VPC3+C má velikost paměti maximálně 4kB, postačuje k jeho přímé adresaci pouze datový typ UWORD = uint16_t. V procesoru Renesas je však obvod VPC3+C namapován na adresu 40000h. Proto při počítání adresy, kde jsou uložena příchozí data v obvodu VPC3+C, docházelo po přetypování na typ VPC3_ADR ke ztrátě informace a chybnému čtení. Navíc při výpisu adresy ukazatele jsou zobrazeny pouze čtyři číslice, proto odhalení chyby nebylo při portaci zřejmé ihned.

V souboru main je procesor uveden do požadovaného stavu a externí sběrnice nastavena ve funkci `vpc_bus_init`. V této funkci jsou změněny pouze ty registry, jejichž požadovaná hodnota je různá od hodnoty po reset v módu 6, ve kterém je procesor při běhu aplikace provozován. Ostatní registry jsou ponechány zakomentované a jsou pouze ve zdrojové kódu.

```
void vpc_bus_init(){
  *SYS_PFCR = PFCR_AE2m;          // port B and C as address bus mode,
                                     A8-A11 enabled as AB
  *DIO_PEDDR |= (3 << 4)|(0x0F); // direction=output for 4th, 5th pin
                                     and pins 0-3 output for DEB leds
  *DIO_PCDDR = 0xFF;              // port C as address bus
  VPC_CS_OFF;
```

```
  VPC3_RESET__;
  FlWait(10000);
  VPC3_CLEAR_RESET__;
  FlWait(10000);
}
```

Překladač při kompilaci hlasí větší počet warning hlášek, které se však týkají stejného problému. Vzhledem ke snaze minimálně zasahovat do kódu nebyly řešeny, protože nemají vliv na funkčnost programu. Tyto hlášky se týkají volání funkcí `memset`, `memcpy` a `memcmp`.

```
...dp_user.c:835: warning: passing arg 1 of ‘memcpy’
       discards qualifiers from pointer target type
```

Tato hláška je způsobena tím, že ukazetele jsou téměř vždy na nějaký typ struktury nebo na něj jsou přetypovávány při volání funkce.

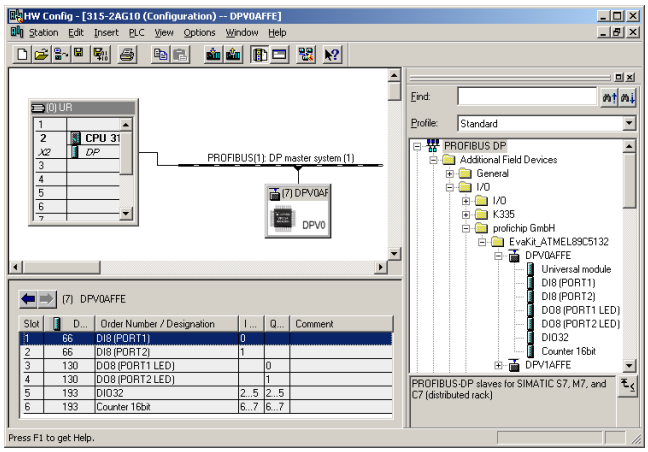
Aplikace byly úspěšně přeloženy a otestovány v realizovaném hardware. Otestování proběhlo v laboratoři K09, kde bylo zařízení připojeno k PLC Siemens Simatic S7 315-DP2.

6.6 Aplikace robot Oscar

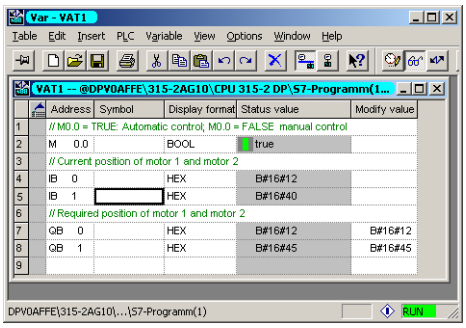
V kapitole realizace hardware bylo popsáno připojení k robotu Oscar a princip ovládání. Stávající aplikace byla upravena na aplikaci DPV0AFFE_robot přidáním modulu robot oscar (`robot_oscar.h` a `robot_oscar.c`).

V tomto modulu byly implementovány jak funkce pro ovládání jednotlivých pinů, tak jednoduchý regulátor pro robota, který se snaží vždy dosáhnout zvolené pozice motoru. Pokud jsou hodnoty mimo dovolený rozsah je dosaženo maximální dovolené hodnoty a poté je pohyb zastaven. Je-li dosaženo optické závory je aktuální hodnota přepsána na hodnotu nula. Tento jev může nastat, pokud robotu brání v pohybu nějaká mechanická překážka či závada na robotu samotném. Následně dojde ke ztrátě kroku. Dále byla implementována funkce home, která nastaví všechna ramena do pozic, kdy jsou aktivovány optické závory. Rychlost každého motorku může být ovládána nezávisle pomocí hodnoty `motorX_speed`, kde X je číslo motoru. Hodnoty mohou být v rozmezí 1 až 8. Zvolená rychlost udává, kolik kroků bude vynecháno mezi po sobě následujícími skutečnými kroky motoru při intervalu generování 2ms.

Jako rozhraní pro užití modulu slouží každému motoru tři proměnné. Ukázka je pro motor číslo jedna.



Obrázek 6.3: Hardwarová konfigurace systému



Obrázek 6.4: Jednoduché ovládání robota z prostředí STEP7

```
uint16_t motor1_current
uint16_t motor1_required
uint8_t motor1_speed
```

Do proměnné current zapisujeme požadovanou hodnotu a následně je možné vyčítat aktuální hodnotu z proměnné current. Motor speed již byla popsána. Dosažení požadované hodnoty zajišťuje funkce `motor1_to_position()`. Tato funkce je periodicky volána z funkce přerušení od časovače každé 2ms, pokud je povolená proměnná `robot_go` povolující celkový běh robota. Obrázek 6.3 znázorňuje obrazovku z prostředí STEP7 při hardwarové konfiguraci systému. Pro ukázkovou aplikaci je použit nezměněný GSD soubor z aplikace DPV0AFFE. Obrázek 6.4 zobrazuje namapování vstupních proměnných `motor1_current`, `motor2_current` a výstupních proměnných `motor1_required`, `motor2_required`. Ty jsou dle obrázku namapovány na adresy IB0, IB1, QB0 a QB1.

Kapitola 7

Závěr

V této práci postupně docházelo k prohloubení dosavadních znalostí autora o protokolu Profibus DP. Počátek práce byl věnován seznámení se s novým zákaznickým ASIC obvodem od firmy Profichip a dodaným vývojovým starterkitem. V této fázi šlo především o úspěšné oživení aplikace a správnou konfiguraci PLC, při které se vyskytlo mnoho menších problémů, které se nakonec podařilo po komunikaci s firmou Profichip vyřešit.

Dalším krokem bylo seznámení se s procesorem LPC2119 a vývojovými nástroji. Poté došlo k použití pouze komunikační části starterkitu a jejímu propojení s obvodem LPC2119. Nejvíce náročnou prací pak byla portace kódu určeného pro procesory založené na jádře x51 do zmíněného procesoru LPC2119, který navíc nemá vyvedenu adresovou a datovou sběrnici z pouzdra, takže musela být nahrazena použitím standartních GPIO portů. Úprava kódu pro procesor nedisponující externí sběrnicí se v průběhu práce ukázala jako vhodná zejména z důvodu nově navrženého obvodu VPC3+S, který bude disponovat rozhraním SPI a IIC. Realizovaný kód tak bude velice snadno přenositelný pro aplikace se zmíněným obvodem.

Po úspěšné portaci a otestování aplikace DP V0 byl zahájen návrh vlastního hardware s obvodem VPC3+C tak, aby bylo možné snadné propojení s již existující deskou LPC2119. Navíc na této desce bylo navrženo rozhraní pro Profibus PA, které je vytvořeno pomocí zakázkového obvodu Siemens SIM1-2. Tato část je sice navržena, nicméně nebyla otestována z důvodu nedostupnosti obvodu v navrženém pouzdře, které je uváděno v nejnovějším datasheetu. Navržená universální deska dále disponuje konektorem, kde jsou vyvedeny zbývající neobsazené vstupy a výstupy procesoru LPC2119, které mohou být použity pro implementaci libovolné funkčnosti.

Jako ukázková aplikace bylo realizováno po menší vratné hardwarové úpravě Profibus rozhraní pro výukového robota Oscar v laboratoři K09. Tento robot dis-

ponuje čtveřicí krokových motorů, které mohou být nyní ovládány z PLC pouhým zápisem požadované polohy daného motoru a rychlosti, kterou se má motor otáčet. Zpětně je do PLC načítána aktuální poloha každého motorku.

Dále byla v rámci práce navržena nová deska s procesorem Renesas H8S2638, která vychází z desky se stejným procesorem jako v robotu Eurobot. Ta však nemá vyvedenu adresovou a datovou sběrnici, která je pro zamýšlenou aplikaci žádoucí. Nová deska navíc opravuje řadu chyb předchozí desky či modernizuje některé její části. Tato deska byla oživena a propojena s komunikační deskou starterkitu pomocí univerzálního konektoru. Do procesoru byl upraven a naprogramován kód pro Profibus automat verze 0 a verze 1. Deska je rovněž připravena k otestování ovládání pohonů s využitím sofistikované knihovny PXMC vyvinuté ing. Pavlem Píšou.

Po dohodě s vedoucím práce a časovému omezení nebyla implementována verze protokolu DP V2 a profil PROFIdrive. Důvodem bylo navržení rozhraní Profibus PA a především nepředpokládaný návrh nové desky s procesorem Renesas.

Celá práce byla realizována v OS Linux, distribuci Kubuntu. K programování PLC byl využit počítač v laboratoři K09 s nainstalovaným prostředím STEP7. Pro odladění a oživení komunikace byl použit Profibus analyzátor s kartou ProfiCore a dodaným softwarem ProfiTrace. Aplikace byla rovněž testována programem ProfiScript s různými výsledky. Všechny tyto aplikace byly instalovány a licencovány na katedrálním notebooku.

Celkově byly v práci navrženy a oživeny dvě desky plošných spojů a realizována funkční aplikace Profibus typu slave jak po hardwarové, tak po softwarové stránce.

Dodatek A

Popis příloh na CD

Přílohou této práce jsou dokumenty, zdrojové kódy, manuály a výkresy desek plošných spojů. Struktura obashu je uvedena na této stránce.

```
MartinVana-DP
  |- doc
  |   |- robotOscar  - původní dokumentace k robotu Oscar
  |   |                použitá k řešení projektu
  |   |- datasheets  - dokumentace k použitým obvodům,
  |                    především LPC2119, H8S2638 a VPC3+C
  |
  |- pcb
  |   |- eaglelibs   - vytvořené knihovny a nástroje pro
  |   |                program Eagle
  |   |- hitachi_cpu_board  - soubor schématu a desky
  |   |                pro procesor H8S2638
  |   |- PB_slave_LPC       - soubor schématu a desky
  |                    pro procesor LPC2119
  |
  |- profichip_cd     - adresář obsahuje cele CD dodávané
  |                    ke starterkitu od firmy Profichip
  |
  |- src
  |   |- syssless-h8300  - zdrojové kódy a nástroje pro
  |   |                    procesor H8S2638. Aplikace
  |   |                    dpv0affe a dpv1affe
```

```
|   |- syssless-lpc21xx - zdrojové kódy a nástroje pro
|                         pro procesor LPC2119.
|                         Aplikace pb_dpv0affe
|                         a pb_dpv0affe_robot
|
|- tools  - balíky .deb a tar.gz s nástroji pro použité
|           procesory
|
|- MartinVana-DP.pdf - tento dokument
```

Dodatek B

Schéma, DPS a propojení použitého HW

B.1 Deska s procesorem LPC2119

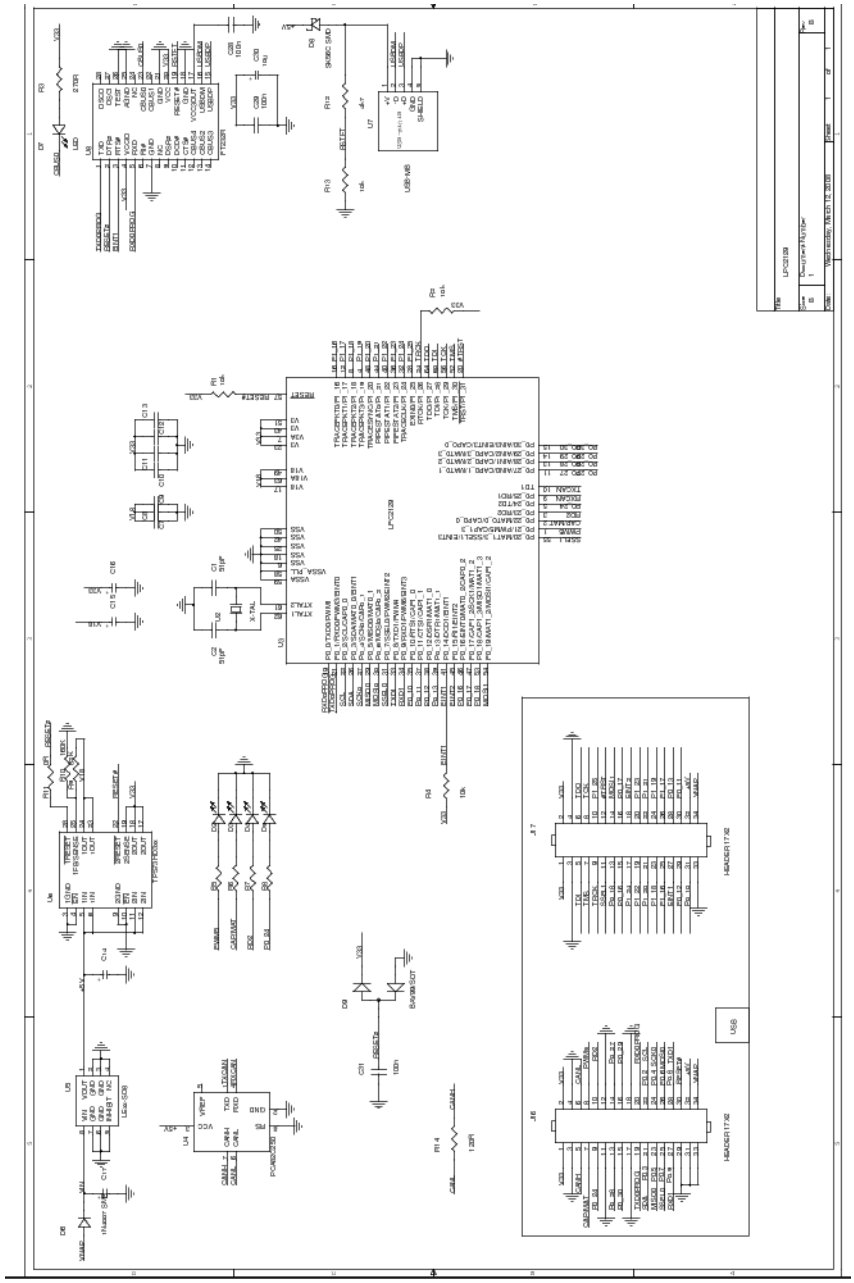
- Schéma zapojení desky LPC2119.
- Propojení desky s obvodem VPC3+C.

B.2 Deska PB-LPC-VPC

- Schéma zapojení desky PB-LPC-VPC.
- Desky plošných spojů.
- Propojení desky s obvodem VPC3+C.

B.3 Deska Renesas H8S2638

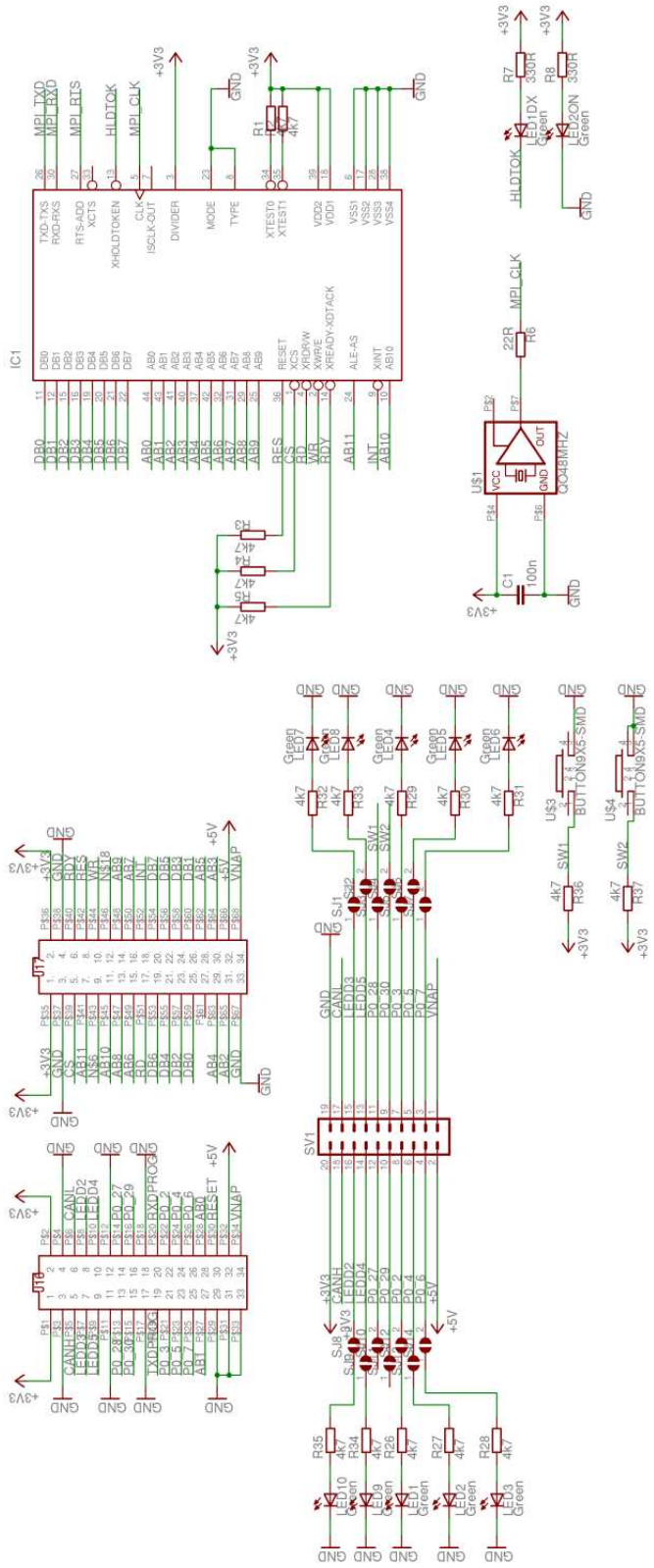
- Schéma zapojení desky H8S2638.
- Desky plošných spojů.
- Propojení desky s obvodem VPC3+C.



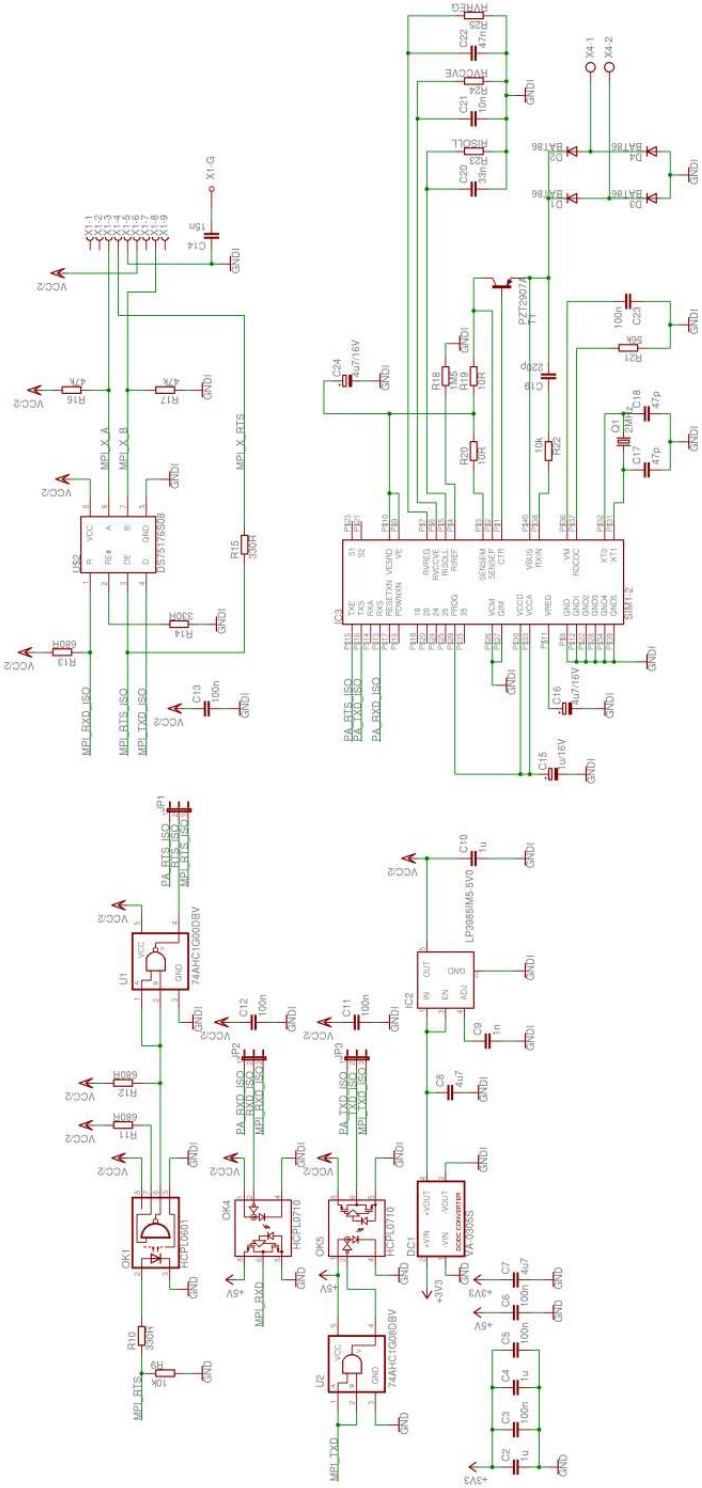
Obrázek B.1: Schéma desky s procesorem LPC2119

Tabulka B.1: Propojení pinů obvodu VPC3+C s deskou LPC2119

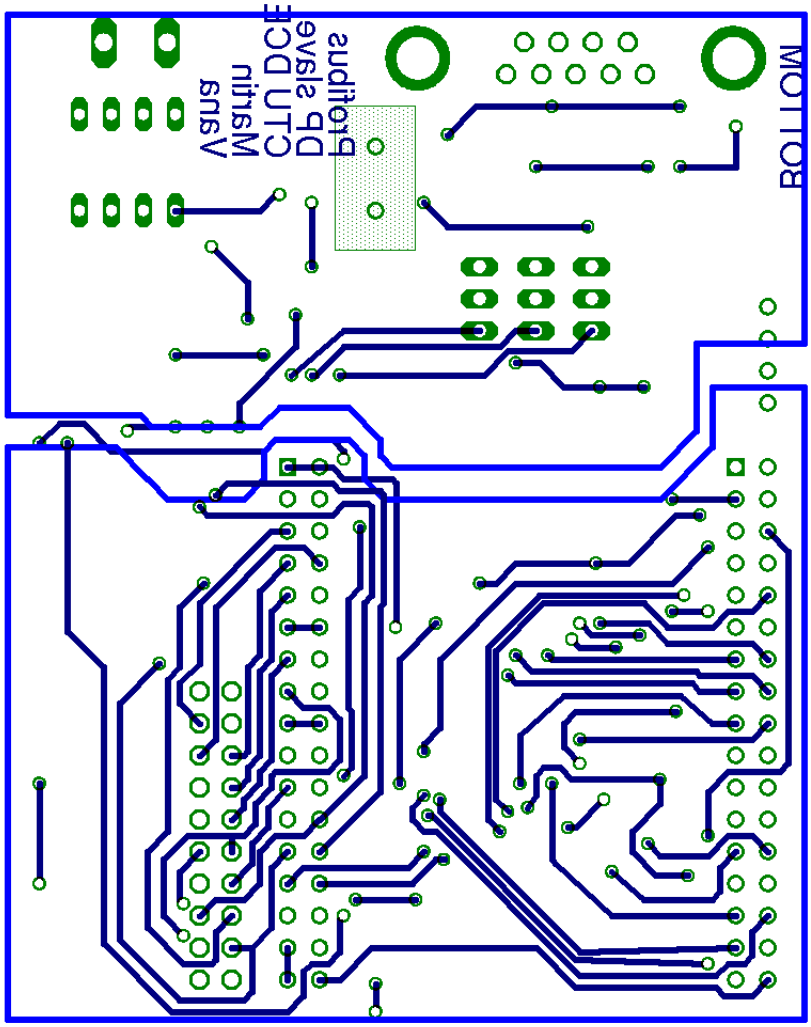
Pin VPC3+C	Název VPC3+C	Pin LPC2119	Konektor LPCBoard	Název LPC2119	Popis signálu
1	XCS	60	J17-5	P1_28	Chip select
2	XWR	28	J17-10	P1_25	Write
4	XRD	32	J17-17	P1_24	Read
9	XINT	45	J17-18	P0_15	Interrupt
10	AB10	55	J17-11	P0_20	AB10
11	DB0	16	J17-25	P1_16	DB0
12	DB1	12	J17-26	P1_17	DB1
14	XREADY	64	J17-6	P1_27	Ready
15	DB2	8	J17-23	P1_18	DB2
16	DB3	4	J17-24	P1_19	DB3
19	DB4	48	J17-21	P1_20	DB4
20	DB5	44	J17-22	P1_21	DB5
21	DB6	40	J17-19	P1_22	DB6
22	DB7	36	J17-20	P1_23	DB7
24	ALE-AS	52	J17-7	P1_30	AB11
25	AB9	54	J17-14	P0_19	AB9
29	AB8	53	J17-13	P0_18	AB8
31	AB7	47	J17-16	P0_17	AB7
32	AB6	46	J17-15	P0_16	AB6
36	RESET	56	J17-8	P1_29	Reset
37	AB4	38	J17-29	P0_12	AB4
40	AB3	37	J17-30	P0_11	AB3
41	AB2	35	J17-31	P0_10	AB2
42	AB5	39	J17-28	P0_13	AB5
43	AB1	34	J16-27	P0_9	AB1
44	AB0	33	J16-28	P0_8	AB0



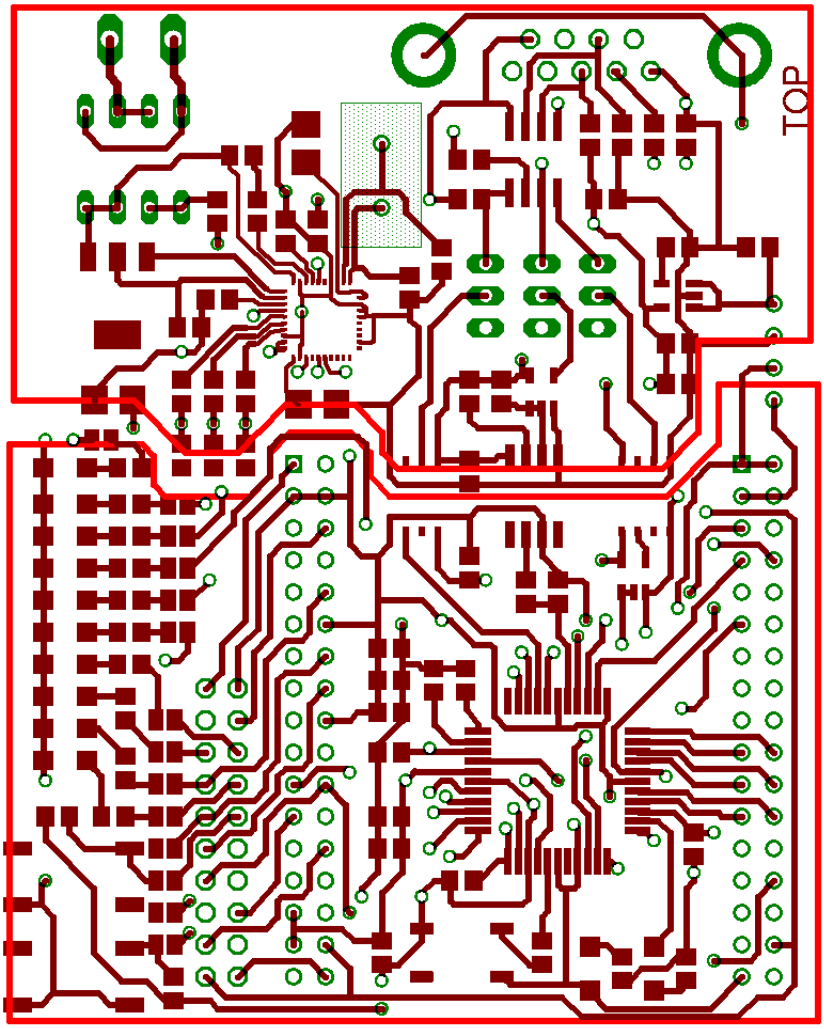
Obrázek B.2: Schéma navržené desky s PB-LPC-VPC 1.část



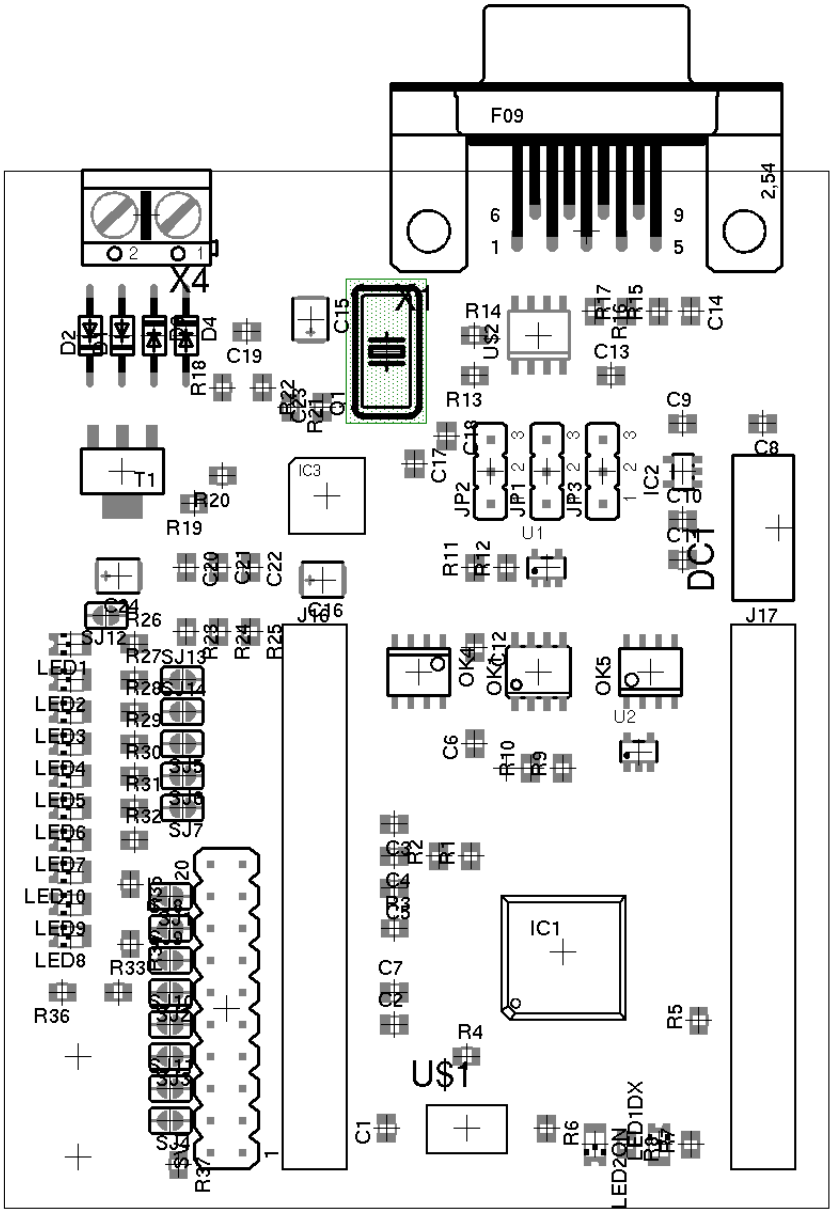
Obrázek B.3: Schéma navržené desky s PB-LPC-VPC 2.část



Obrázek B.4: Pohled na spodní vrstvu spojů desky PB-LPC-VPC



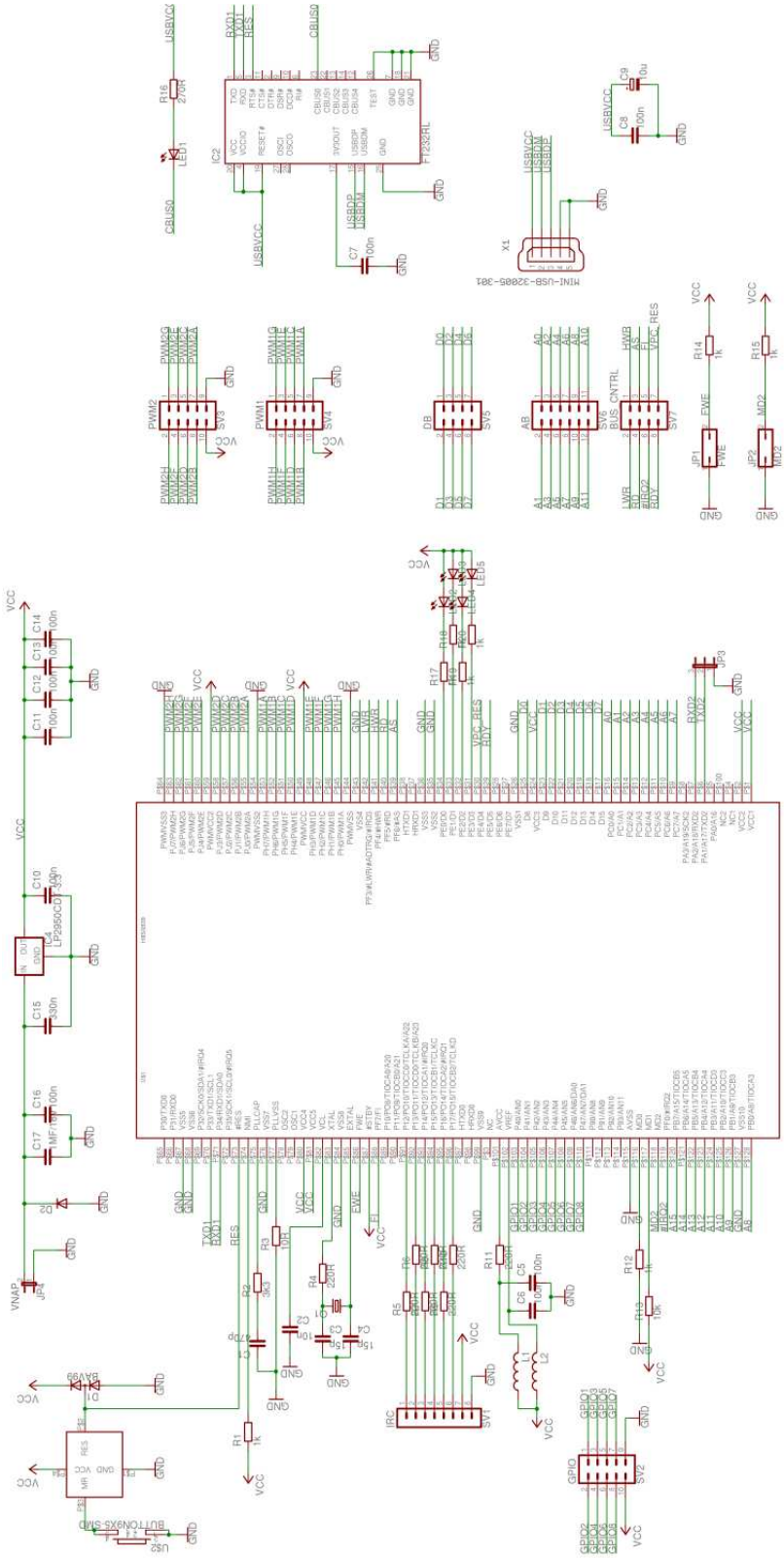
Obrázek B.5: Pohled na vrchní vrstvu spojů desky PB-LPC-VPC



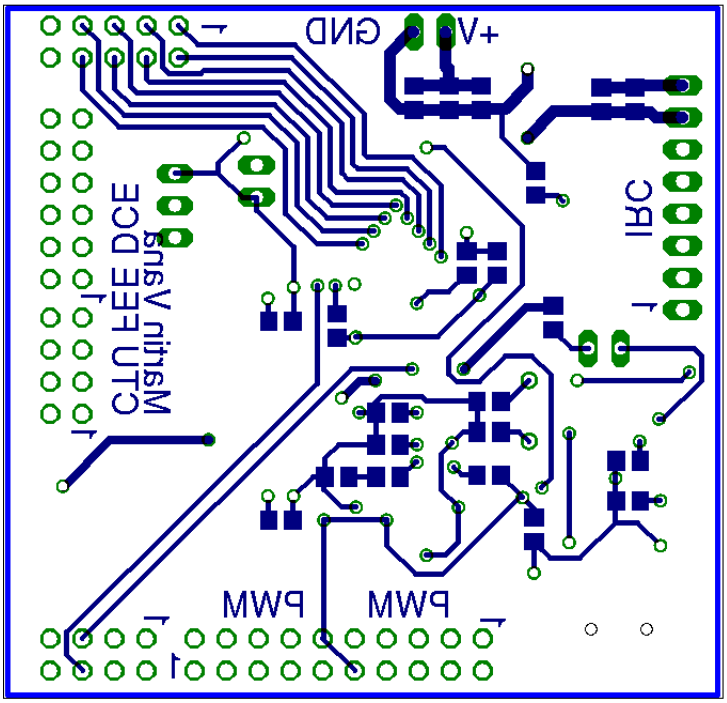
Obrázek B.6: Rozvržení součástek na desce PB-LPC-VPC

Tabulka B.2: Popis universálního konektoru SV1 a propojení s deskou LPC2119

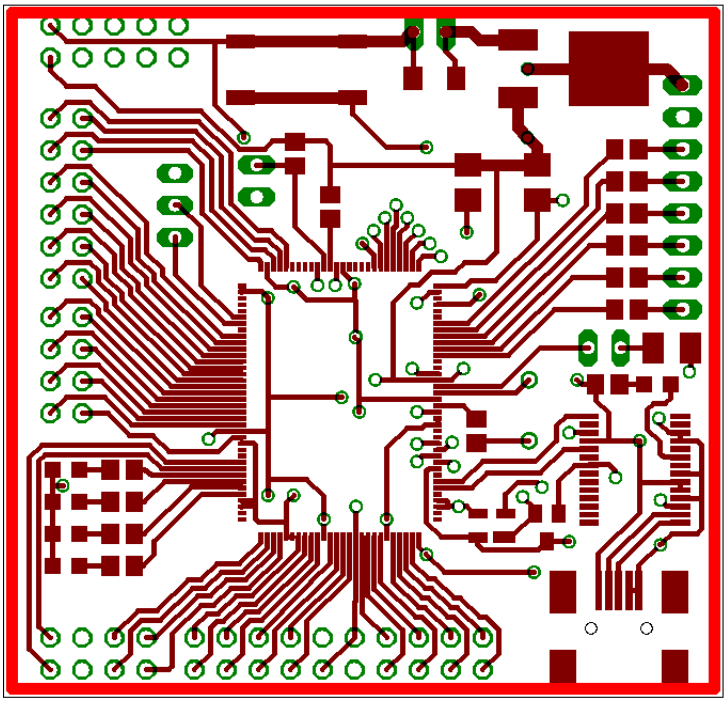
Konektor SV1	Pin LPC2119	Konektor LPCBoard	Název LPC2119	Popis signálu
1	-	J17-34	-	Napájecí napětí
2	-	J17-32	-	+5V
3	31	J16-25	P0_7	I/O
4	30	J16-26	P0_6	I/O
5	29	J16-23	P0_5	I/O
6	27	J16-24	P0_4	I/O
7	26	J16-21	P0_3	I/O
8	22	J16-22	P0_2	I/O
9	15	J16-15	P0_30	I/O
10	14	J16-16	P0_29	I/O
11	13	J16-13	P0_28	I/O
12	11	J16-14	P0_27	I/O
13	5	J16-9	P0_24	LED D5
14	3	J16-10	P0_23	LED D4
15	2	J16-7	P0_22	LED D3
16	1	J16-8	P0_21	LED D2
17	-	J16-6	-	CANL
18	-	J16-5	-	CANH
19	-	J17-1	-	+3.3V
20	-	J17-33	-	GND



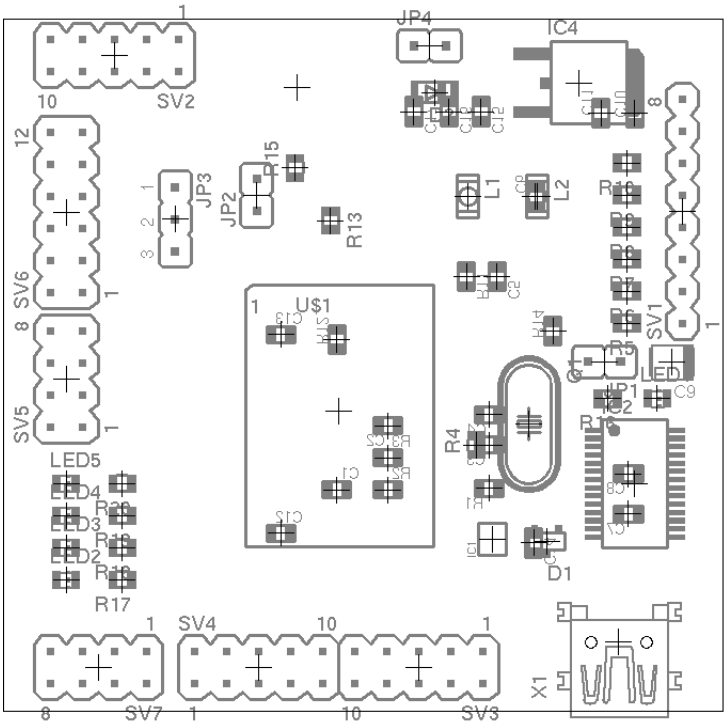
Obrázek B.7: Schéma navržené desky Renesas H8S2638



Obrázek B.8: Pohled na spodní vrstvu spojů desky Renesas H8S2638



Obrázek B.9: Pohled na vrchní vrstvu spojů desky Renesas H8S2638



Obrázek B.10: Rozvržení součástek na desce Renesas H8S2638

Tabulka B.3: Propojení desky Renesas H8S2638 s Profichip VPC3+C board

Pin VPC3+C	Název VPC3+C	Pin Renesas	Konektor H8S2638 board	Název H8S2638	Popis signálu
1	XCS	29	SV7-8	CS	Chip select
2	XWR	41	SV7-1	HWR	Write
4	XRD	40	SV7-4	RD	Read
9	XINT	119	SV7-6	IRQ2	Interrupt
10	AB10	125	SV6-11	A10	AB10
11	DB0	25	SV5-1	D8	DB0
12	DB1	23	SV5-2	D9	DB1
14	XREADY	-	-	-	Ready
15	DB2	22	SV5-3	D10	DB2
16	DB3	21	SV5-4	D11	DB3
19	DB4	20	SV5-5	D12	DB4
20	DB5	19	SV5-6	D13	DB5
21	DB6	18	SV5-7	D14	DB6
22	DB7	17	SV5-8	D15	DB7
24	ALE-AS	124	SV6-12	A11	AB11
25	AB9	126	SV6-10	A9	AB9
29	AB8	128	SV6-9	A8	AB8
31	AB7	9	SV6-8	A7	AB7
32	AB6	10	SV6-7	A6	AB6
36	RESET	30	SV7-7	PE4	Reset
37	AB4	12	SV6-5	A4	AB4
40	AB3	13	SV6-4	A3	AB3
41	AB2	14	SV6-3	A2	AB2
42	AB5	11	SV6-6	A5	AB5
43	AB1	15	SV6-2	A1	AB1
44	AB0	16	SV6-1	A0	AB0

Literatura

[1] *Normative parts of Profibus FMS,DP and PA according to EN50170 volume 2.* PROFIBUS Nutzerorganisation e.V., 1998.

[2] http://pa.profibus.com/transmission/index.html. Profibus pa - transmission technology and installation. 2009.

[3] http://www.profibus.com. Profil-profidrive_3172_v40_aug05.pdf. 2005.

[4] http://www.profichip.com/products/overviewasics/dp-slave-vpc3 c/. Vpc3+c user manual revision 1.03. 2006.

[5] http://www.renesas.com. H8s/2638 hardware manual revision 6. 2005.

[6] http://www.standardics.nxp.com/support/documents/microcontrollers/pdf/user.manual.lpc2109.lpc2114.lpc2119.lpc2124.lpc2129.lpc2194.lpc2210.lpc2212.lpc2214.lpc2220.lpc2290.lpc2292.lpc2294.pdf. Lpc2119.pdf user manual revision 03. 2008.

[7] Manfred Popp. *The New Rapid Way to PROFIBUS DP.* PROFIBUS Nutzerorganisation e.V., Karlsruhe, 2003.

[8] M. Váňa. *Bakalářská práce, Profibus DP slave.* 2006.