České vysoké učení technické v Praze – Fakulta elektrotechnická
Katedra řídicí techniky

DIPLOMOVÁ PRÁCE

Profibus modul pro USB

2003 Miloš Hofman

Zadávací formulář

České vysoké učení technické v Praze - Fakulta elektrotechnická

Katedra řídicí techniky ***Školní rok:*** 2001/2002

ZADÁNÍ DIPLOMOVÉ PRÁCE

Student: Miloš Hofman

Obor: Technická kybernetika

Název tématu: Profibus modul pro USB

Zásady pro vypracování:

1. Seznamte se se specifikací USB a vrstvy FDL protokolu Profibus.
2. Navrhněte koncepci zařízení Profibus s rozhraním USB. V návrhu berte zřetel na možnost budoucího připojení redundantního systému Profibus.
3. Zařízení realizujte.

Seznam odborné literatury: Dodá vedoucí práce.

Vedoucí diplomové práce: Ing. Pavel Burget

Datum zadání diplomové práce: listopad 2001 (změna zadání: 07.01.2003)

Termín odevzdání diplomové práce: květen 2003

doc. Ing. Jiří Bayer, CSc.
vedoucí katedry

L.S.

prof. Ing. Vladimír Kučera, DrSc.
v zast. **děkan**

V Praze dne 08.01.2003

Anotace

Tato diplomová práce se zabývá návrhem komunikačního zařízení, které je vybaveno rozhraními Profibus, RS-232, USB a Ethernet. Jako hlavní komunikační člen se používá mikrořadič Motorola 68360. Důraz je kladen na univerzálnost použití zařízení s přihlédnutím k možnosti rozšíření dalšími moduly.

Ke styku s USB je použit samostatný komunikační řadič, který převádí sériová data z USB na paralelní k procesoru Motorola a naopak. Zbývající periferie pak obsluhuje přes optické oddělovače a budiče sběrnic přímo hlavní mikrořadič.

Softwarová část se zaměřuje na konfiguraci mikrořadiče a komunikačního procesoru, který je jeho součástí (integrovaný na jednom čipu). Základem je zprovoznění kanálu RS-232, který slouží převážně jako terminál k zobrazení důležitých ladicích informací. V závěrečné fázi byly vytvořeny obslužné rutiny kanálu Profibus a spustila se na něm komunikace se stanicí typu slave. Softwarové vybavení je vytvořené tak, aby bylo možné současně komunikovat na více kanálech. To dává možnost provozovat zařízení na redundantním vedení.

Annotation

This thesis describes a development of a communication device, which is equiped with interfaces Profibus, RS-232, USB and Ethernet. As a main communication item is used Motorola 68360 microcontroller. Universality with the possibility of expansion of this device by another modules is emphasised.

For the USB connection a separate controller is used, which converts serial data from USB to parallel data for Motorola controller. Remaining periphery are operated directly over optical couplers and bus drivers by the main controller.

The software part is focused at the configuration of the controller and the communication controller, which is its component (integrated on the chip). The basis is a take a run of the RS-232 channel, that mostly serves as a terminal to display the debugging data. In final phase Profibus channel service routines were created and tested for a communication with a slave station. Software equipment is designed so that it's possible to communicate on more channels simultaneously. This allows to connect the device to a redundant bus.

Prohlášení

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

Nemám závažný důvod proti užití tohoto školního díla ve smyslu § 60 Zákona č.121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon).

V Praze dne ……. …….
podpis

Poděkování

Tímto bych chtěl poděkovat všem, co se podíleli na vývoji diplomové práce, zejména vedoucímu projektu Ing. Pavlu Burgetovi za rady a směrování v průběhu řešení. Rovným dílem potom Ing. Pavlu Píšovi za pomoc při vývoji a oživování hardwaru a obrovskou trpělivost a podporu při řešení softwarové části. V neposlední řadě Ing. Petru Smolíkovi za rady při vývoji hardwarové části a Ing. Pavlu Růžičkovi.

Obsah

Zadávací formulář **2**

Anotace **3**

Annotation **3**

Prohlášení **4**

Poděkování **5**

Obsah **6**

Seznam obrázků **9**

Seznam tabulek **9**

1. Úvod **11**

2. Profibus DP **12**

2.1 ISO – OSI model 13
2.2 Fyzická vrstva 13
2.2.1 Konektor 14
2.2.2 Kabel 14
2.2.3 Zakončení 15
2.2.4 Formát dat 15
2.3 Spojová vrstva 16
2.3.1 Typy rámců 16
2.3.2 Pole SA, DA 17
2.3.2.1 Rozšířené adresování 17
2.3.3 Pole FC 18
2.3.4 Stavový diagram stanice typu master 20
2.3.4.1 Offline 21
2.3.4.2 Listen Token 21
2.3.4.3 Active Idle 21
2.3.4.4 Claim Token 21
2.3.4.5 Use token 21
2.3.4.6 Await Data Response 22
2.3.4.7 Check Access Time 22
2.3.4.8 Pass Token 22
2.3.4.9 Check Token Pass 23
2.3.4.10 Await Status Response 23
2.3.5 Časové operace 23
2.3.5.1 Bit Time 23
2.3.5.2 Syn Time 23
2.3.5.3 Syn Interval Time 23
2.3.5.4 Station Delay 24
2.3.5.5 Ready Time 24
2.3.5.6 Idle Time 24
2.3.5.7 Transmission Delay 25

2.3.5.8 Slot Time ... 25
2.3.5.9 Time Out Time ... 25
2.3.5.10 GAP Update Time ... 25
2.3.5.11 Token Cycle Time ... 26
2.3.5.12 Message Cycle Time ... 26
2.3.5.13 Systém Reaction Times ... 26
2.3.6 Časovače ... 27
2.3.6.1 Token Rotation Timer ... 27
2.3.6.2 Idle Timer ... 27
2.3.6.3 Slot Timer ... 27
2.3.6.4 Time – Out Timer ... 27
2.3.6.5 Syn Interval Timer ... 27
2.3.6.6 GAP Update Timer ... 27
2.3.7 Čítače ... 28

3. USB (Universal Series Bus) ... 29

3.1 Úvod ... 29
3.2 Architektura ... 29
3.2.1 Topologie sítě ... 30
3.3 Fyzická vrstva ... 31
3.4 Protokol sběrnice ... 31
3.5 Robustnost ... 32
3.6 Připojení zařízení ke sběrnici ... 32
3.7 Odpojení zařízení od sběrnice ... 32

4. Hardware ... 33

4.1 Popis celku ... 33
4.2 Procesorová část ... 34
4.2.1 Základní parametry mikrořadiče ... 34
4.2.2 Blokový popis ... 36
4.2.3 Časování ... 37
4.2.4 Resetování ... 38
4.2.5 Přiřazení pinů ... 39
4.3 USB ... 40
4.4 RS-232 ... 41
4.5 Profibus ... 42
4.6 Ethernet ... 43
4.7 BDM ... 44

5. Software ... 45

5.1 Vývojový řetězec ... 45
5.2 Mapa paměti procesoru ... 46
5.3 Konfigurace prostředí ... 49
5.3.1 GDB ... 49
5.3.2 Překladač jazyka C ... 51
5.4 Konfigurace procesoru ... 52
5.4.1 Registr MBAR (Module Base Address Register) ... 52
5.4.2 Registr MCR (Module Configuration Register) ... 52
5.4.3 Registr PEPAR (Port E Pin Assignment Register) ... 52
5.4.4 Registr SYPCR (SYstem Protection Control Register) ... 52
5.4.5 Registr GMR (Global Memory Register) ... 53

5.4.6 Registr BR (Base Register) ... 53
5.4.7 Registr OR (Option Register) ... 53
5.4.8 Registr VBR (Vector Base Register) ... 54
5.4.9 Ostatní registry ... 54
5.5 Příkazy pro komunikační procesor ... 54
5.6 Buffer deskriptory ... 55
5.7 UART ... 56
5.7.1 Parametrizační paměť ... 56
5.7.2 Buffer Deskriptory ... 59
5.7.2.1 Přijímací BD ... 60
5.7.2.2 Vysílací BD ... 61
5.7.3 Příkazy CP ... 62
5.7.4 Inicializace ... 63
5.8 Profibus ... 64
5.8.1 Parametrizační paměť ... 65
5.8.2 Příkazy CP ... 68
5.8.3 Konfigurační registry ... 68
5.8.3.1 Registr GSMR ... 68
5.8.3.2 Registr PSMR ... 69
5.8.4 Přijímací BD ... 69
5.8.5 Vysílací BD ... 71
5.8.6 Události ... 72
5.8.6.1 Registr událostí – SCCE ... 72
5.8.6.2 Registr masky – SCCM ... 73
5.8.7 Časovače ... 73
5.8.7.1 Token Rotation Timer ... 74
5.8.7.2 Syn Interval Timer ... 75
5.8.7.3 Slot Timer ... 75
5.8.7.4 Time-Out Timer ... 75
5.8.7.5 Inicializace časovačů ... 76
5.8.8 Seznam aktivních stanic ... 76
5.9 Zdrojový kód ... 77
5.9.1 Obsluha přerušení ... 78
5.9.2 Ukládání dat ... 79
5.9.3 Odesílání dat ... 80

6. Závěr ... 81

7. Přílohy ... 82
7.1 Technické údaje ... 82
7.2 Konfigurace hardwaru ... 83
7.2.1 Bity CONFIG 0 - 2 ... 83
7.2.2 Bity MODCK 0 - 1 ... 83
7.2.3 Přiřazení komunikačních rozhraní ... 84
7.3 Fotografie ... 85

8. Literatura ... 86

Seznam obrázků

Obrázek 2-1: Profibus DP, ISO – OSI model ... 13
Obrázek 2-2: Profibus DP – Konektor ... 14
Obrázek 2-3: Profibus DP – Zakončení kabelu ... 15
Obrázek 2-4: Profibus DP – Přenos znaku ... 15
Obrázek 2-5: Profibus DP – Stavový diagram stanice typu master ... 20
Obrázek 3-1: USB – Topologie sítě ... 30
Obrázek 3-2: USB – Propojovací kabel ... 31
Obrázek 4-1: Připojení periferií ... 33
Obrázek 4-2: Připojení pamětí ... 33
Obrázek 4-3: Blokové schéma mikrořadiče Motorola 68360 ... 36
Obrázek 4-4: Schéma časování MC68360 ... 37
Obrázek 4-5: Schéma resetování MC68360 ... 38
Obrázek 4-6: Schéma části USB ... 40
Obrázek 4-7: Schéma části RS-232 ... 41
Obrázek 4-8: Schéma části Profibus ... 43
Obrázek 4-9: Konektory pro připojení Ethernet kontroléru ... 44
Obrázek 4-10: Schéma připojení BDM ... 44
Obrázek 5-1: Vývojový řetězec ... 45
Obrázek 5-2: UART – Přenos znaku ... 56
Obrázek 5-3: UART – Parametrizační paměť ... 57
Obrázek 5-4: Umístění, funkce BD, buffery ... 59
Obrázek 5-5: Profibus – Parametrizační paměť ... 65
Obrázek 7-1: Fotografie Profi-USB ... 85

Seznam tabulek

Tabulka 2-1: Profibus DP – Fyzická vrstva, vlastnosti ... 13
Tabulka 2-2: Profibus DP – Zapojení konektoru ... 14
Tabulka 2-3: Profibus DP – Parametry kabelů ... 14
Tabulka 2-4: Profibus DP – Délky kabelů ... 15
Tabulka 2-5: Profibus DP – Přístupové body (SAP) ... 17
Tabulka 2-6: Profibus DP – Struktura pole FC ... 18
Tabulka 2-7: Profibus DP – Typy rámců, pole FC, b3 - b0 ... 19
Tabulka 3-1: USB – Přenosové rychlosti ... 31
Tabulka 4-1: Použité paměti ... 34
Tabulka 4-2: Jumpery Config 0, 1, 2 ... 34
Tabulka 4-3: Přiřazení pinů brány A ... 39
Tabulka 4-4: Přiřazení pinů brány B ... 39
Tabulka 4-5: Přiřazení pinů brány C ... 39
Tabulka 5-1: Kódy adresních prostorů ... 46
Tabulka 5-2: Mapa paměti ProfiUsb ... 47
Tabulka 5-3: Mapa Dual Port RAM ... 48
Tabulka 5-4: Bázové adresy modulů ... 48
Tabulka 5-5: Kódování šířky sběrnice pro makro set_CS_OR ... 50
Tabulka 5-6: Registr BR ... 53
Tabulka 5-7: Registr OR ... 53
Tabulka 5-8: Registr CR ... 54

Tabulka 5-9: Příkazy pro komunikační procesor ... 55
Tabulka 5-10: Kódování kanálů komunikačního procesoru ... 55
Tabulka 5-11: Struktura BD ... 55
Tabulka 5-12: UART – Společná parametrizační paměť ... 57
Tabulka 5-13: UART – Specifická parametrizační paměť ... 58
Tabulka 5-14: UART – Přijímací BD ... 60
Tabulka 5-15: UART – Vysílací BD ... 61
Tabulka 5-16: Profibus – Společná parametrizační paměť ... 65
Tabulka 5-17: Profibus – Specifická parametrizační paměť ... 66
Tabulka 5-18: Profibus – PSMR ... 69
Tabulka 5-19: Profibus – Přijímací BD ... 69
Tabulka 5-20: Profibus – Vysílací BD ... 71
Tabulka 5-21: Profibus – Registr událostí ... 72
Tabulka 5-22: Profibus – Časovače 1 ... 73
Tabulka 5-23: Profibus – Časovače 2 ... 74
Tabulka 5-24: Token Rotation Timer ... 74
Tabulka 5-25: Syn Interval Timer ... 75
Tabulka 5-26: Slot Timer ... 75
Tabulka 5-27: Time-Out Timer ... 75
Tabulka 5-28: Inicializace časovačů ... 76
Tabulka 5-29: Moduly projektu ... 77
Tabulka 5-30: Inicializace kanálu Profibus ... 78
Tabulka 5-31: Inicializace kanálu UART ... 78
Tabulka 7-1: Technické údaje ... 82
Tabulka 7-2: Jumpery CONFIG 0 - 2 ... 83
Tabulka 7-3: Jumpery MODCK 0 - 1 ... 83
Tabulka 7-4: Přiřazení RS-232 k SMC2, SCC4 ... 84
Tabulka 7-5: Přiřazení Profibus1 k SCC1, SCC3 ... 84
Tabulka 7-6: Přiřazení Profibus2 k SCC2, SCC4 ... 84

1. Úvod

V dnešní době se v automatizovaných průmyslových provozech stále častěji používají k řízení výrobní linky průmyslové počítače (PLC). Komunikace mezi nimi, senzory a akčními členy ve velké míře probíhá prostřednictvím sítě Profibus [13].

Celá řada současně produkovaných zařízení není vybavena rozhraním pro připojení na tento průmyslový standard. Jejich úprava by byla finančně velmi náročná, mnohdy nemožná. Většina stávajících přístrojů je vybavena komunikačním rozhraním RS-232 nebo jeho doplnění do systému je velmi snadné. Vzrůstá také množství komponent komunikujících po síti USB (Universal Serial Bus). Stále více se nabízí možnost komunikovat prostřednictvím sítě Ethernet. Díky snadnému propojení přes Internet lze přistupovat k zařízení prakticky odkudkoli. Tato fakta dala příležitost vzniku tématu mé diplomové práce. Jejím cílem je návrh zařízení (dostalo název ProfiUSB), které vytváří interface mezi následujícími sítěmi:

- RS-232
- Profibus (DP)
- USB
- Ethernet

Existuje mnoho variant jak takové zařízení realizovat. V každém případě se musel použít více či méně výkonný mikroprocesor a být doplněn alespoň o periferní obvody (většinou je nutné ještě přidat rozšiřující nebo samostatné paměťové moduly).

Jedna z variant je použít méně výkonný mikroprocesor (např. řady 8051) a přidat periferní obvody pro styk s výše uvedenými sítěmi. RS-232 je díky své jednoduchosti standardně podporovaná většinou výrobců mikroprocesorů. Pro styk s USB existuje samostatný čip např. od firmy Philips, který lze snadno připojit ke většině vyráběných mikrořadičů a neklade příliš velké nároky na jejich výkon. Situace je podobná u Profibus(u), kde je styk možné realizovat obvody Asic od firmy Siemens. Připojení k Ethernetu je možné realizovat periferním obvodem vybaveným vnějším transformátorem. Tato varianta se ukázala být velmi nekompaktní a pro dosažení nižších přenosových rychlostí také nevýkonná.

Další možnost je použít výkonnější mikroprocesor. Tím odpadá nutnost vybavit zařízení tak velkým množstvím periferních obvodů a datové přenosy se navíc zrychlí. Tuto volbu jsem upřednostnil a ProfiUSB jsem postavil na základě mikrořadiče Motorola 68360. Jedná se o výkonný 32bitový mikroprocesor, který obsahuje na jednom čipu integrovaný komunikační procesor. Ten dokáže obsluhovat současně až sedm sériových komunikačních kanálů, z nichž čtyři mohou být použity pro styk se sběrnicí Profibus/RS-232/…, dva pro RS-232/… a poslední pro SPI. Komunikační procesor podporuje také práci v síti Ethernet a na rozdíl od předchozí varianty nemusí být doplněn velkým množstvím periferních obvodů. K řadiči jsou připojeny vnější paměťové moduly a periferie. V této konfiguraci lze dosáhnout vyššího komunikačního výkonu (až 1,5 Mb/s na kanálech typu Profibus a 10Mb/s na kanálu typu Ethernet), o výpočetním výkonu mikroprocesoru ani nemluvě (4,5MIPS).

Práce je členěná do několika celků. Zpočátku se zabývám popisem jednotlivých sítí a jejich parametrů, které jsou potřebné pro realizaci zařízení. Další část přináší popis hardwarové realizace včetně konkrétních ukázek bloků schématu. Zde jsou zdůrazněny klíčové součásti, na kterých závisí přenosová rychlost zařízení. Poslední část se zabývá konfigurací mikrořadiče pro specifickou aplikaci a přiřazením periferií k jednotlivým kanálům komunikačního procesoru. Dále je zde popsáno rozhraní, jehož prostřednictvím komunikuje 32bitové jádro mikrořadiče s komunikačním procesorem.

2. Profibus DP

Profibus je průmyslová sběrnice, která je velice rozšířená v oblasti řízení technologických procesů. Používá se k propojení řídících a akčních členů. Její vznik pochází od firmy Siemens, která poskytla možnost použití i ostatním firmám.

Existují tři varianty:

- Profibus – FMS (Fieldbus Message Specification) se dnes již téměř nepoužívá.
- Profibus – DP (Decentralized Periphery) je hojně využíván v běžném průmyslu.
- Profibus – PA (Process Automation) se používá v náročných prostředích, jako je např. prostředí s nebezpečím výbuchu.

Na sběrnici Profibus jsou připojeny dva základní typy stanic:

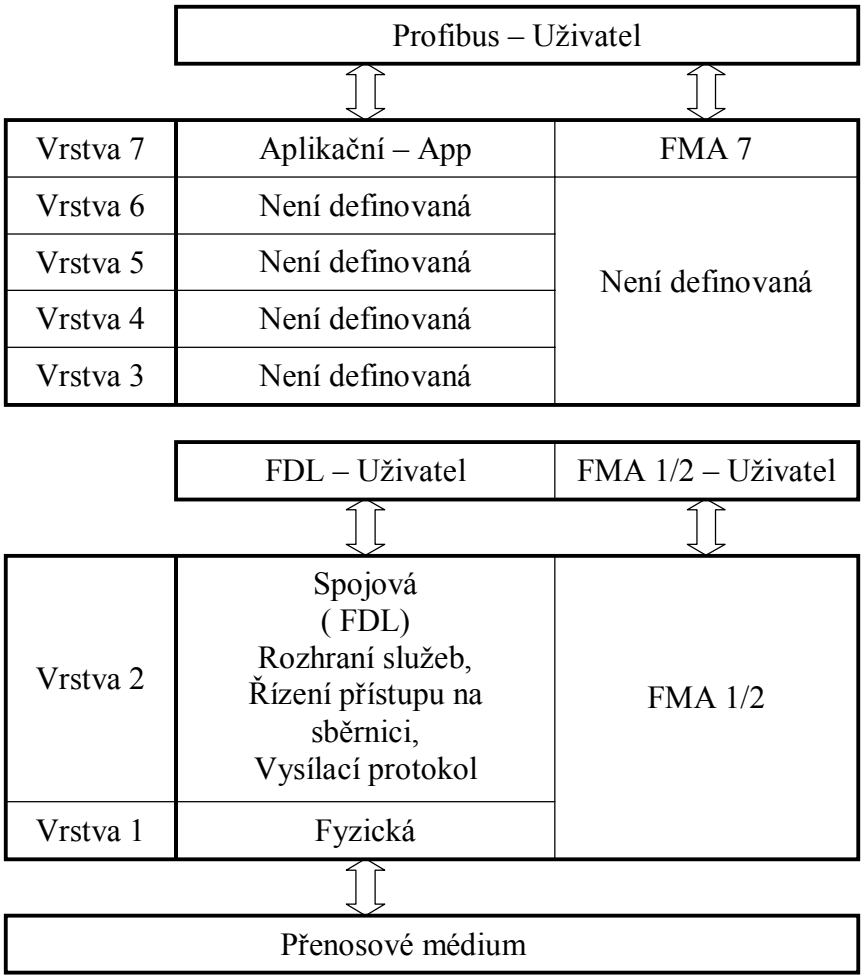
Master – Je iniciátorem všech přenosů po síti, může jich být připojeno zároveň více. Musí řešit řízení přístupu na sběrnici.

Slave – Jedná se o pasivní stanici, vykonává požadavky od mastera. Maximální doporučený počet těchto stanic na sběrnici je 32, v nenáročných aplikacích jich lze připojit až 127. Neřeší řízení přístupu na sběrnici.

Přístup ke sběrnici je možné zjednodušeně popsat takto: Zařízení typu Master si předávají mezi sebou pověření. Ten, který dostal pověření se ujímá sběrnice a nikdo jiný nemůže vysílat. Po vyřízení všech jeho úkolů se sběrnice vzdá tím, že předá pověření další stanici typu Master v pořadí v logickém kruhu.
Pokud je na sběrnici jen jeden Master, není potřeba předávat pověření dál a sběrnice patří pouze jemu. Minimální konfigurace sběrnice je jedno zařízení Master a jedno Slave nebo dvě zařízení typu Master.
Zařízení ProfiUSB je vybaveno rozhraním k variantě Profibus DP, proto se nyní zaměřím na popis této varianty.

2.1 ISO – OSI model

K popisu jednotlivých vrstev protokolu a jejich funkcí uvádím tento ISO – OSI model:



Obrázek 2-1: Profibus DP, ISO – OSI model

2.2 Fyzická vrstva

Zvolená fyzická vrstva Profibusu DP má následující vlastnosti:

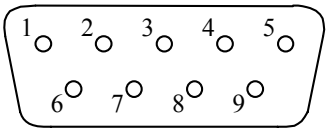
Topologie	Lineární sběrnice, terminátory na obou koncích, vývody ≤ 0,3m
Přenosové médium	Stíněná kroucená dvoulinka
Maximální délka	1200m
Maximální počet stanic	32
Přenosová rychlost	9,6 / 19,2 / 93,75 / 187,5 / 500 kbit/s 1,5 / 3 / 6 / 12 Mbit/s
Typ přenosu	Asynchronní
Konektor	9pinový D-Sub s kolíčky na kabelu

Tabulka 2-1: Profibus DP – Fyzická vrstva, vlastnosti

Tyto údaje platí pro sběrnici bez opakovačů.

2.2.1 Konektor

K připojení stanic k síti se používá 9pinový konektor D – sub:



Obrázek 2-2: Profibus DP – Konektor

Zapojení konektoru je následující:

Číslo vývodu	Označení RS – 485	Označení signálu	Význam
1		SHIELD	Stínění, ochranná zem
2		M24V	Mínus 24V, výstup
3	**A/A***	**RxD/TxD – P**	Datový – positivní
4		CTR – P	Řídicí – positivní
5	**C/C***	**DGND**	Datová zem
6		VP	Napájení
7		P24V	Plus 24V, výstup
8	**B/B***	**RxD/TxD – N**	Datový – negativní
9		CTR – N	Řídicí – negativní

Tabulka 2-2: Profibus DP – Zapojení konektoru

Pouze tučně označené položky jsou povinné. VP je nutné pouze při umístění zařízení na okraji sběrnice kvůli zakončovacímu pull-up rezistoru.

2.2.2 Kabel

Jako médium se zde používá kroucená dvoulinka. Lze použít dva typy kabelů (Typ A, Typ B). Charakteristická impedance by se měla pohybovat mezi 100Ω a 220Ω. Průřez má být větší než 0,22mm^2.

Kabel / Vlastnost	Typ A	Typ B
Impedance	135 ÷ 165 Ω (f = 3 ÷ 20 Mhz)	100 ÷ 130 Ω (f > 100 khz)
Kapacita	< 30 pF/m	< 60 pF/m
Odpor	< 110 Ω/km	–
Průřez	≥ 0,34 mm^2	≥ 0,22 mm^2

Tabulka 2-3: Profibus DP – Parametry kabelů

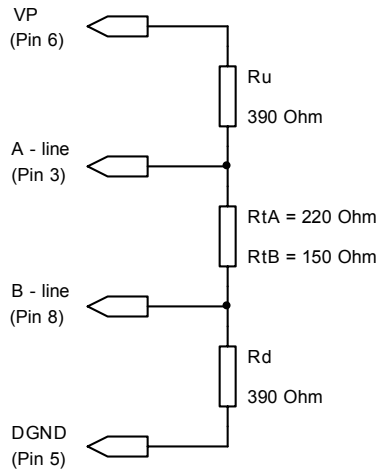
V závislosti na použitém kabelu je potom definovaná maximální délka:

Přenosová rychlost [kbit/s]	9.6	19.2	93.75	187.5	500	1500
Délka kabelu Typu A [m]	1200	1200	1200	1000	400	200
Délka kabelu Typu B [m]	1200	1200	1200	600	200	70

Tabulka 2-4: Profibus DP – Délky kabelů

2.2.3 Zakončení

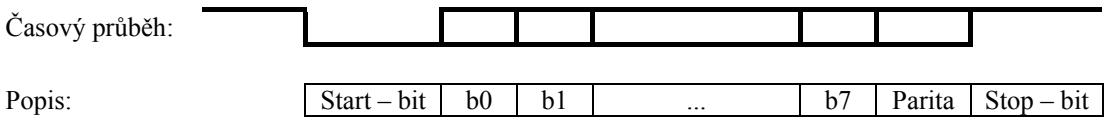
Aby při provozu na vyšších frekvencích nevznikaly odrazy na nepřizpůsobených koncích kabelu, jsou oba konce vybaveny zakončovacím zařízením – terminátory.
Zapojení je následující:



Obrázek 2-3: Profibus DP – Zakončení kabelu

2.2.4 Formát dat

Jak již bylo uvedeno, Profibus DP používá asynchronní typ komunikace. Když je sběrnice v klidu (nikdo nevysílá), A–vodič má vyšší potenciál než B–vodič. Tento stav se kóduje jako logická 1. Jakmile ze začne vysílat, objeví se na sběrnici start–bit (logická 0), osm datových bitů, sudý paritní bit a jeden stop–bit.



Obrázek 2-4: Profibus DP – Přenos znaku

2.3 Spojová vrstva

Spojová vrstva definuje metodu přístupu na sběrnici (MAC – Mefium Acces Control), služby datové výměny a řídicí služby. Data, požadavky a odpovědi jsou po sběrnici vysílány jako balíky (rámce, framy). Jsou sestaveny z jednotlivých znaků (viz Obrázek 2-4), které jsou řazeny bezprostředně za sebe.

2.3.1 Typy rámců

Rozlišujeme tyto typy rámců:

Rámec s pevnou délkou bez uživatelských dat
Tento rámec začíná vždy znakem SD1 a jeho struktura je následující:

SD1 (10h)	DA	SA	FC	FCS	ED (16h)

Rámec s proměnnou délkou dat
Používá se k datové výměně mezi stanicemi. Jeho struktura je:

SD2 (68h)	LE	LEr	SD2 (68h)	DA	SA	FC	DAE	SAE	DU	FCS	ED (16h)

Tato struktura platí obecně pro režim rozšířeného adresování. V případě, že rozšířené adresování nepoužíváme, pole SAE, DAE odpadají.

Rámec s konstantní délkou dat
Používá se k datové výměně mezi stanicemi. Má následující strukturu:

SD3 (A2h)	DA	SA	FC	DU	FCS	ED (16h)

Rámec pověření
Posílá ho stanice master k zaslání pověření (tokenu) další stanici master. Tento rámec vždy předchází synchronizační pole.

SD4 (DCh)	DA	SA

Rámec Short Acknowledgement
Jedná se o krátkou odpověď, slouží k indikaci správného přijetí předešlého rámce. Je to nejkratší rámec a tvoří ho pouze jeden znak – SC (Single Char).

SC (E5h)

Rámce musí předcházet tzv. synchronizační pole, pokud se jedná o rámce typu Request. Rámce používané k odpovědi (Response, Acknowledgement) nemusí mít předřazené synchronizační pole.

Význam jednotlivých symbolů:
SD1, SD2, SD3, SD4 – Start Delimiter znak, slouží k identifikaci typu rámce
ED – End Delimiter, ukončovací znak rámce

DA – Destination Address, cílová adresa (komu je rámec určený), přípustný rozsah 0 ÷ 127, 127 se používá pro Broadcast
SA – Source Address, zdrojová adresa (kdo rámec posílá), rozsah adres 0 ÷ 126
FC – Frame Control, specifikuje typ rámce (SRD, SDN, Ack)
DU – Data Unit, pole uživatelských dat
FCS – Frame Check Sequence, kontrolní součet
LE – LEngth, délka informačního pole (4 ÷ 249 znaků), do délky se počítají pole: DA, SA, FC, DU
LEr – Zopakování délky informačního pole, musí být stejná jako LE, jinak jde o chybový rámec
SAE – Destination Access Extension, oktet rozšířeného adresování zdroje
DAE – Destination Access Extension, oktet rozšířeného adresování cíle
SC – Single Char, tento znak se používá pro krátkou odpověď

Synchronizační pole – Má délku 33bitů (tři znaky) a je složené ze samých log.1

2.3.2 Pole SA, DA

Udávají adresu stanice na síti. Pro SA je přípustný rozsah adres 0 ÷ 126, pro DA se používají adresy 0 ÷ 127, 127 je určená pro zasílání zpráv typu Broadcast a Multicast.
Adresové pole mají sedm bitů (b0 – b6), osmý bit (b7) je označený jako EXT a používá se pro rozšířené adresování.
Zprávy typu Response, Acknowledgement by měly zaměňovat adresy SA a DA z předešlého akčního rámce. To znamená, že v akčním rámci je adresa SA shodná s adresou DA v rámci typu Acknowledgement nebo Response.

2.3.2.1 Rozšířené adresování

Pokud chceme používat rozšířené adresování (např. pro přístup ke službám definovaných DSAP, SSAP), nastavíme nejvyšší bity v polích SA, DA (bity EXT) na log.1. Při zvoleném rozšířeném adresování se bezprostředně za polem FC objeví rozšiřující oktety adres (DAE, SAE).
Rozšiřovací oktety mají specifický formát (viz str.127 v [13]). Pokud chceme rozšířené adresování použít k aktivaci služeb definovaných přístupovými body, uvedeme do polí DAE, SAE adresu DSAP, SSAP. Kódy důležitých přístupových bodů ukazuje následující tabulka:

SSAP	DSAP	Význam
62	55	Změna adresy
62	56	Čtení vstupů
62	57	Čtení výstupů
62	58	Globální řízení
62	59	Čtení konfigurace
62	60	Čtení diagnostických informací
62	61	Zaslání parametrů
62	62	Kontrola konfigurace

Tabulka 2-5: Profibus DP – Přístupové body (SAP)

Pokud nepoužijeme rozšířené adresování a pole DSAP, SSAP vynecháme, bude služba obsloužena z tzv. Default SAP. Default SAP je volitelný přístupový bod a má kódování adresy NIL.

2.3.3 Pole FC

Toto pole udává typ rámce (Request, Acknowledgment). Obsahuje příznaky pro identifikaci opakovaného posílání rámce. Zabezpečuje zprávy proti ztrátě. Pole má strukturu:

<table>
<tr><td>Bit</td><td>b7</td><td>b6</td><td>b5</td><td>b4</td><td>b3</td><td>b2</td><td>b1</td><td>b0</td></tr>
<tr><td rowspan="3">Význam</td><td rowspan="3">Res</td><td>1</td><td>FCB</td><td>FCV</td><td colspan="4" rowspan="3">Function</td></tr>
<tr><td>Frame</td><td colspan="2"></td></tr>
<tr><td>0</td><td colspan="2">Station Type</td></tr>
</table>

Tabulka 2-6: Profibus DP – Struktura pole FC

Význam jednotlivých příznaků:

Res – Rezervováno, zde by se měla zapisovat nula.
Frame – Typ rámce: 0 – Jde o rámec typu Request / Send Request
1 – Rámec je typu Acknowlesgment / Response
b6 = 1: FCB – Frame Count Bit, cyklicky mění úroveň 0 / 1
FCV – Frame Count Bit Valid, určuje, zda minulý rámec měl správnou úroveň na FCB bitu
b6 = 0: Station Type 00 – stanice typu Slave
01 – stanice Master, nepřipravena vstoupit do logického kruhu
10 – stanice Master, připravena vstoupit do logického kruhu
11 – stanice Master, připojena v logickém kruhu
Function – Udává typ rámce, viz Tabulka 2-7.

FCB bit zabraňuje zdvojení zprávy na straně příjemce a ztrátu zprávy na straně iniciátora. Výjimku tvoří zprávy: Send Data with No Acknowledge (SDN), Request FDL Status, Request Ident a Request LSAP Status. Master by měl udržovat hodnoty FCB bitu pro všechny stanice.
Když se vyšle akční rámec příjemci poprvé (nebo znovu, pokud je příjemce právě v seznamu „non operational“), FCB by měl být pevně nastavený na 1. To znamená, že iniciátor vyšle akční rámec s nastavením: FCV = 0, FCB = 1. Příjemce by to měl vyhodnotit jako vůbec první akční rámec od daného iniciátora a uložit si hodnotu FCB = 1 spolu s jeho adresou. Tento cyklus zpráv iniciátor neopakuje.
V dalších akčních rámcích iniciátor vysílá FCV = 1 společně s negovanou hodnotou FCB.
Když příjemce obdrží akční rámec s FCV = 1, porovná přijatou hodnotu FCB s hodnotou uloženou (hodnota FCB minulého akčního rámce od stejného iniciátora). Pokud detekuje změnu logické úrovně, tak je vše v pořádku.
Jestliže došlo ke ztrátě odpovědi nebo byla vyhodnocená jako chybná, iniciátor tím zjistí chybný předešlý cyklus a nezmění hodnotu FCB bitu pro opakované poslání zprávy.
Když příjemce dostane zprávu s FCV = 1 a se stejným FCB jako má uloženo (od stejného iniciátora), uskuteční opakování poslední odpovědi, kterou drží v pohotovosti. Pro rámce typu Send Data with No Acknowledge, Request FDL Status, Request Ident a Request LSAP Status, FCV i FCB jsou rovny nule a příjemce nevyhodnocuje FCB.

Bity b3 – b0 společně s bitem Frame udávají typ rámce podle následující tabulky:

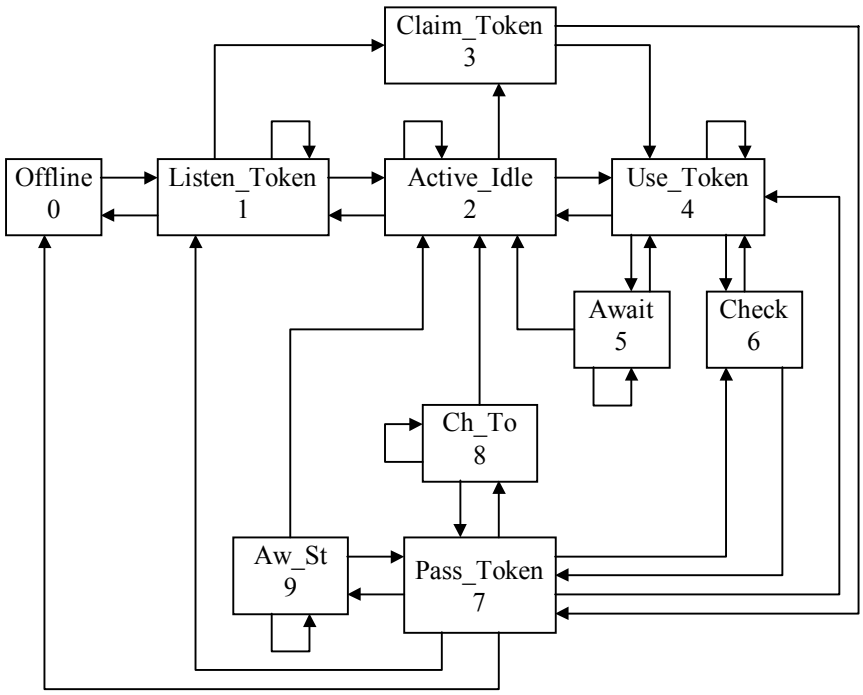
b3 – b0	Function
Frame Type – b6 = 1	
0,1,2	Reserved
3	Send Data with Acknowledge low
4	Send Data with No Acknow. low
5	Send Data with Acknowledge high
6	Send Data with No Acknow. high
7	Reserved (Req. Diagnosis Data)
8	Reserved
9	Request FDL Status with Reply
10,11	Reserved
12	Send and Request Data low
13	Send and Request Data high
14	Request Ident with Reply
15	Request LSAP Status with Reply
Frame Type – b6 = 0	
0	ACKnowledgement positive (OK) *)
1	ACK negative (FDL/FMA1/2 User Error)
2	ACK negative (no Resource for Send Data and no Response FDL Data) (RR)
3	ACK negative, no Service activated (RS)
4 to 7	Reserved
8	Response FDL/FMA1/2 Data low (& Send Data ok) (DL)
9	ACK negative, no Response FDL/FMA1/2 Data & Send Data ok (NR) *)
10	Response FDL Data high & Send Data ok (DH)
11	Reserved
12	Response FDL Data low, no Resource for Send Data (RDL)
13	Response FDL Data high, no Resource for Send Data (RDH)
14,15	Reserved

*) Rámce jsou ekvivalentní rámci Short Acknowledgement

Tabulka 2-7: Profibus DP – Typy rámců, pole FC, b3 - b0

2.3.4 Stavový diagram stanice typu master

Profibus používá smíšenou metodu přístupu na sběrnici. Jedná se o kombinaci principu Token Passing a Master – Slave. Sběrnici může začít používat pouze aktivní stanice typu master. Stanice slave se nemohou nezávisle ujmout sběrnice, pouze odpovídají na dotazy stanic master.
Iniciátor všech přenosů je stanice master, která vlastní pověření (token). Pověření si mezi sebou předávají stanice typu master, které jsou členy logického kruhu a to v rostoucím pořadí adres. Stanice master si vytvářejí vlastní seznam aktivních stanic (LAS), který určuje pořadí stanic v logickém kruhu. Každá aktivní stanice „zná“, jaká stanice je v logickém kruhu před ní (Previous Station – PS), za ní (Next Station – NS) a samozřejmě zná svou vlastní adresu (This Station – TS). Pokud je v kruhu pouze jedna stanice master, posílá pověření sama sobě.
Po studeném startu sběrnice je nutné vytvořit seznamy LAS v aktivních stanicích. Stanice s nejnižší adresou se ujme sběrnice a vysílá ostatním rámce Request FDL Status, aby zjistila jaké typy stanic jsou ke sběrnici připojeny a zda jsou aktivní stanice schopny vstoupit do logického kruhu.
Chování stanice master nejlépe popíše následující stavový diagram:



0 – Offline
1 – Listen Token
2 – Active Idle
3 – Claim Token
4 – Use Token
5 – Await Data Response
6 – Check Access Time
7 – Pass Token
8 – Check Token Pass
9 – Await Status Response

Obrázek 2-5: Profibus DP – Stavový diagram stanice typu master

2.3.4.1 Offline

Stanice se neúčastní komunikace v síti. V tomto stavu se nachází stanice po zapnutí napájení, po vyvolání služby vrstvy FMA1/2 – Reset FDL nebo po vyhodnocení nějaké chyby. Po zapnutí napájení stanice provádí tzv. Self Test, ten záleží na konkrétní implementaci a nesmí poškozovat ostatní stanice.
Zde dochází k inicializaci všech interních parametrů. Stanice může být pouze připojená ke sběrnici, ale vysílaní má zakázané.

2.3.4.2 Listen Token

Po inicializaci všech parametrů, jestliže je stanice master schopná zařadit se do logického kruhu, přejde do stavu Listen Token. Zde naslouchá na sběrnici a zjišťuje jaké aktivní stanice jsou účastníky logického kruhu. Analyzují se rámce typu Token a na základě zjištěných adres se buduje seznam aktivních stanic (LAS). Když stanice vyhodnotí dva kompletní oběhy tokenu, zůstává ve stavu Listen Token dokud neobdrží dotaz Request FDL Status od stanice, která se nachází v logickém kruhu před ní (PS). Na tento dotaz odpoví „ready to enter logical token ring“ a po obdržení rámce token, který obsahuje její adresu přejde do stavu Active Idle.
Během budování seznamu LAS na dotaz Request FDL Status buď neodpovídá nebo odpoví „not ready“. Žádné jiné rámce nezpracovává a neodpovídá na ně.
Když stanice vyhodnotí dva rámce s jeho adresou (SA), zjistí, že v logickém kruhu se nachází jiný master s identickou adresou a vrátí se do stavu Offline.
Jestliže FDL vrstva nezjistí během stanovené doby (Time–Out Time) aktivitu na sběrnici, je nutné inicializovat logický kruh. Stanice se snaží zaslat token sama sobě a poté logický kruh inicializovat.

2.3.4.3 Active Idle

Sem se dostáváme přechodem ze stavu Listen Token. Stanice naslouchá na sběrnici a pokud dostane akční rámec s vlastní adresou, reaguje na něj buď potvrzením nebo odpovědí. Když obdrží rámec token s vlastní adresou, tak pokud chce zachovat logický kruh, přejde do stavu Use Token, jinak se vrací do stavu Listen Token (to se stává při detekci chyby, jako je např. přijetí token rámce s jeho adresou dvakrát za sebou). Když stanice zjistí, že po dobu Time–Out je sběrnice „mrtvá“, přejde do stavu Claim Token a snaží se inicializovat kruh.

2.3.4.4 Claim Token

Do tohoto stavu stanice přejde buď ze stavu Active Idle nebo Listen Token, když vyprší její doba Time–Out. Zde se provádí inicializace logického kruhu. Po aktualizaci seznamů LAS a GAPL, přechází do stavu Use Token.
Inicializace začíná zasláním rámce token dvakrát za sebou na svou vlastní adresu. Děje se to proto, aby si ostatní stanice master založily v seznamu LAS záznam s jeho adresou. Po vyslání tokenů stanice přechází do stavu Await Status Response, aby vytvořila vlastní seznam GAPL. Stanice proto vysílá na všechny následující adresy rámce typu Request FDL Status.

2.3.4.5 Use token

Zde se nachází stanice po obdržení rámce token na svou adresu. FDL vrstva nyní může obsloužit své požadavky od těch s vysokou prioritou po ty s nízkou prioritou. Vykonává cykly zpráv (Message Cycles).

Přechodem do tohoto stavu by se měl přečíst čas z časovače Token Rotation Time čas T_{RR} (Real Rotation Time) a časovač restartovat. Stanice začíná vyřizovat požadavky s vysokou prioritou, přičemž jeden cyklus zpráv je povolené obsloužit vždy. Další cykly se provádí pouze tehdy, jestliže jsou splněné časové závislosti, tedy $T_{RR} < T_{TR}$ (Target Rotation Time).
Po každém vyslaném akčním rámci, který vyžaduje potvrzení/odpověď, přejde do stavu Await Data Response a spustí časovač Slot Timer. Když dostane potvrzení od adresáta, může opět přejít do stavu Use Token.
Po vykonání požadovaných služeb, nebo když na počátku stavu Use Token nejsou žádné požadavky s vysokou prioritou, přechází do stavu Check Access Time.

2.3.4.6 Await Data Response

Zde se nacházíme po odeslání akčních rámců. FDL vrstva čeká na odpověď/potvrzení po dobu Slot Time.
Nyní se čeká na jednu z následujících událostí:

a) Platné potvrzení nebo platná odpověď
b) Nějaký jiný platný rámec (token, akční rámec)
c) Neplatný rámec (start-, end-, length-, FCS-, parity-, stop- bit error) nebo uplynutí doby Slot Time.

Po přijetí a vyhodnocení potvrzení/odpovědi přejde FDL řadič do stavu Use Token, aby vyřídil další požadavky.
Jestliže dojde k přijetí nějakého jiného rámce, znamená to chybu, rámec se nebere v úvahu a přejde se do stavu Active Idle.
Po přijetí neplatného rámce nebo vypršení časovače Slot Timer by se měl předešlý akční rámec vyslat znovu. Pokud ani po zopakování nedostane stanice platnou odpověď/potvrzení, FDL by měla informovat uživatele a přejít do stavu Use Token. Další požadavky této stanici se neopakují, dokud od ní nedojde platná zpráva (po akčním rámci Send and Request Data).

2.3.4.7 Check Access Time

Nyní by se měl spočítat dostupný čas Token Holding Time jako rozdíl T_{TR} - T_{RR} a pokud je výsledek kladný (ještě je nějaký čas k dispozici), může se přejít do stavu Use Token. Jinak FDL aktivuje přechod do stavu Pass Token.

2.3.4.8 Pass Token

Zde se stanice snaží zaslat pověření stanici, která je po ní v logickém kruhu (NS). Při vysílání tokenu by měl přijímač sledovat sběrnici a pokud přijme něco jiného než vyslal, vyhodnotí to jako závažnou chybu na přijímacím nebo vysílacím kanálu. FDL vrstva by měla zastavit svou činnost v logickém kruhu a oznámit událost vrstvě FMA1/2.
V případě, že se přijme vlastní rámec poškozený, to může být způsobeno dočasnou závadou přijímače/vysílače nebo chybou na sběrnici, neznamená to hned opuštění logického kruhu, ale přejde se do stavu Check Token Pass, jako když se příjme rámec token v pořádku. Když by se rámec opětovně odesílal (v důsledku detekce nečinnosti stanice NS) a monitoroval se chybně, FDL by opustila logický kruh, přešla by do stavu Listen Token a uvědomila vrstvu FMA1/2.
Když vyprší čas aktualizace GAP seznamu a stanice má volný čas (Token Holding Time), pustí se do jeho aktualizace. Vyšle rámec Request FDL Status stanici, která je na řadě v seznamu GAP a přejde do stavu Await Status Response. Když se z odpovědi

stanice rozkóduje, že jde o stanici master, která má zájem se připojit do logického kruhu, následně se jí vyšle pověření. Jakmile se pověření podaří úspěšně odeslat (monitoruje se odesílání), GAP seznam se zkrátí o jeden záznam.
V případě, že odpověď identifikuje stanici slave nebo master, která nechce vstoupit do logického kruhu, stanice se přidá do seznamu GAP. Pokud existující stanice neodpoví ani po opětovném pokusu, vymaže se ze seznamu GAP (adresa se označí jako nepoužitá).
Když během údržby GAPL žádná nová stanice master neodpoví, předá se pověření aktuální NS stanici a přejde se do stavu Check Token Pass. Jestliže stanice nezná svého následníka v logickém kruhu, pošle pověření sama sobě a přejde do stavu Use Token.

2.3.4.9 Check Token Pass

FDL vrstva čeká po dobu Slot Time na reakci stanice, které se vyslalo pověření.
Když se během této doby přečte ze sběrnice platný rámec, vyhodnotí se předání pověření úspěšně, přejde se do stavu Active Idle a tam se vyhodnotí přijatý rámec.
Pokud se během doby Slot Time přijme neplatný rámec, vyhodnotí se stanice jako aktivní a také přejde do stavu Active Idle.
Jestliže FDL nepřijme žádný rámec, přejde do stavu Pass Token.

2.3.4.10 Await Status Response

Dojde se sem ze stavu Pass Token při inicializaci nebo údržbě GAP seznamu. FDL čeká po dobu jednoho Slot Time času na odpověď. Když je přijatý rámec poškozený nebo není přijatý žádný rámec, přejde se do stavu Pass Token buď pro opakování rámce Request nebo zaslání pověření sama sobě.

2.3.5 Časové operace

Činnost aktivní stanice je silně závislá na časování. Norma předepisuje časové posloupnosti ve stavovém modelu. Zde uvádím výpočet a definici důležitých časových intervalů:
Následující časy se měří na bitové okamžiky.

2.3.5.1 Bit Time

Bit Time (t_{BIT}) je čas, který uplyne během vyslání jednoho bitu.

$$t_{BIT} = \frac{1}{Bit\ rate\,[bit/s]}$$

2.3.5.2 Syn Time

Je to minimální interval, během kterého je nutné vysílat prázdné znaky nebo nechat sběrnici v klidu (naměří se log.1 na sběrnici). Tento časový úsek se vkládá před každý akční rámec.

$$T_{SYN} = 33 \cdot t_{BIT} = 33\ bit$$

2.3.5.3 Syn Interval Time

Udává maximální možný čas, který může trvat mezi dvěma vloženými poli SYN. Odpovídá době dvou kompletních cyklů zpráv, zprávy mají maximální povolenou

délku. Odchylka synchronizace vysílače je povolená v jednom ze synchronizačních polí.

$$T_{SYNI} = 2 \cdot (2 \cdot (33\ bit + 255 \cdot 11\ bit)) + 33\ bit = 11385\ bit$$

2.3.5.4 Station Delay

Značíme T_{SDx}, je to doba, která uplyne mezi vysláním/příjmem posledního bitu rámce a vysláním/příjmem prvního bitu následujícího rámce. Definují se následující tři zpoždění:

Station Delay Iniciátora (stanice, která vysílá akční nebo token rámec):

$$T_{SDI} = \frac{t_{SDI}}{t_{BIT}}$$

Minimum Station Delay of Responders (stanic, které odpovídají rámcem typu acknowledgment nebo response):

$$\min T_{SDR} = \frac{\min t_{SDR}}{t_{BIT}}$$

Maximum Station Delay of Responders:

$$\max T_{SDR} = \frac{\max t_{SDR}}{t_{BIT}}$$

2.3.5.5 Ready Time

Když zařízení master vyšle akční rámec, pasivní stanice za určitou dobu odpoví rámcem typu acknowledgement nebo response. Ready time je nejkratší doba, po jejímž uplynutí od odvysílání akčního rámce je master schopný přijímat odpovědi od zařízení slave. Pro tento čas musí být splněna následující podmínka:

$$T_{RDY} < \min T_{SDR}$$

2.3.5.6 Idle Time

Tento čas se vztahuje ke straně iniciátora. Je to čas, který uplyne mezi přijetím posledního bitu rámce a vysláním prvního bitu následujícího rámce iniciátorem.
Po rámci token, acknowledgement, response je definice následující:

$$T_{ID1} = \max\left(T_{SYN} + T_{SM}, \min T_{SDR}, T_{SDI}\right)$$

Po akčním rámci, který nevyžaduje odpověď, platí následující definice:

$$T_{ID2} = \max\left(T_{SYN} + T_{SM}, \max T_{SDR}\right)$$

2.3.5.7 Transmission Delay

Je to maximální čas, který uplyne od začátku vysílání jedné stanice na sběrnici do začátku přijímání druhé stanice na sběrnici. Odvíjí se od rychlosti šíření signálu po sběrnici a závisí též na její délce. Tato doba je vztažená k době trvání jednoho bitu.

$$T_{TD} = \frac{t_{TD}}{t_{BIT}}$$

2.3.5.8 Slot Time

Slot Time, značíme T_{SL}, je maximální doba, po kterou iniciátor čeká na přijetí prvního znaku odpovědi adresované stanice od vyslání posledního znaku akčního rámce. Jiná definice určuje Slot Time jako maximální dobu, kterou aktivní stanice čeká na objevení prvního znaku na sběrnici od vyslání posledního znaku rámce pověření zaslaného jiné aktivní stanici.

$$T_{SL1} = 2 \cdot T_{TD} + \max T_{SDR} + 11 \cdot T_{BIT} + T_{SM}$$

$$T_{SL2} = 2 \cdot T_{TD} + \max T_{ID1} + 11 \cdot T_{BIT} + T_{SM}$$

Pro zjednodušení realizace se používá pouze jeden čas Slot Time, který je roven maximu z obou výše definovaných, tedy:

$$T_{SL} = \max(T_{SL1}, T_{SL2})$$

2.3.5.9 Time Out Time

Monitoruje aktivitu aktivních a pasivních stanic na sběrnici. Měření začíná bezprostředně po zapnutí napájení přechodem do stavu Listen Token. Potom se měří vždy od přijetí posledního bitu rámce. Měření končí přijetím prvního bitu následujícího rámce. Když je sběrnice pasivní po dobu time-out, vyhodnotí se to jako chyba (ztráta pověření).

$$T_{TO} = 6 \cdot T_{SL} + 2 \cdot n \cdot T_{SL}$$

Pro stanice typu master je definováno *n* jako adresa stanice (0 ÷ 126), pro stanice typu slave je *n* pevně dané hodnotou 130.
První sčítanec respektuje maximální dobu mezi dvěma rámci, druhý je definován proto, aby nešlo k situaci, kdy všechny aktivní stanice na sběrnici se snaží chopit pověření (po ztrátě pověření nebo restartu logického kruhu) ve stejný časový okamžik.

2.3.5.10 GAP Update Time

Tento čas potřebuje stanice master k definici periody údržby seznamu GAP. Po zřízení seznamu GAP se provádí jeho údržba cyklicky s periodou T_{GUD}. Údržba se začne vykonávat po přijetí pověření po obsloužení požadavků s vyšší prioritou, jestliže je k

dispozici volný Token Holding Time nebo později během držení pověření. Definujeme jako násobky času Token Rotation Time – T_{TR} .

$$T_{GUD} = G \cdot T_{TR}, \qquad 1 \le G \le 100$$

2.3.5.11 Token Cycle Time

Je to doba, která uplyne od přijetí prvního bitu rámce pověření stanicí master na pozici *k* v logickém kruhu do přijetí prvního bitu rámce pověření stanici master na pozici *k+1* v logickém kruhu.

$$T_{TC} = T_{TF} + T_{TD} + T_{ID}$$

2.3.5.12 Message Cycle Time

Jedná se o dobu výměny dat mezi stanicemi. Skládá se z doby akčního rámce, zpoždění vlivem přenosu po sběrnici, zpoždění stanic a délky rámce odpovědi.

$$T_{MC} = T_{S/R} + T_{SDR} + T_{A/R} + T_{ID} + 2 \cdot T_{TD}$$

$T_{S/R}$ je délka rámce Send/Request, $T_{A/R}$ je délka rámce Acknowledgement/Reponse.

2.3.5.13 Systém Reaction Times

Nejdříve definujeme množství zpráv, které je možné vyslat po sběrnici za jednotku času, R_{SYS} .

$$R_{SYS} = \frac{1}{t_{MC}}, \qquad t_{MC} = T_{MC} \cdot t_{BIT}$$

Maximální Systém Reaction Time v konfiguraci sítě jeden master a *n* zařízení slave v režimu pure polling je definován:

$$T_{SR} = np \cdot T_{MC} + mp \cdot T_{RETMC},$$

kde: np počet zařízení typu slave,
mp počet opakovaných cyklů výměny zpráv,
T_{MC} délka výměny zpráv,
T_{RETMC} délka zpráv při opakované výměně.

Maximální Systém Reaction Time pro konfiguraci sítě několik zařízení typu master a slave je rovna času Target Rotation Time.

$$T_{SR} = T_{TR}$$

2.3.6 Časovače

K realizaci zařízení typu master jsou nezbytné následující časovače.

2.3.6.1 Token Rotation Timer

Když stanice master přijme rámec typu token, tento časovač se naplní na předvolbu Target Rotation Time – T_{TR} a dekrementuje se každý bitový okamžik. Jestliže stanice přijme pověření znovu, přečte se zbývající hodnota časovače – Token Holding Time – T_{TH} a časovač se opět naplní hodnotou T_{TR}. Real Rotation Time – T_{RR} se spočítá jako rozdíl T_{TR} - T_{TH}. Když zbývá čas T_{TH} (je větší než nula), zařízení může obsloužit zprávy s nízkou prioritou.

2.3.6.2 Idle Timer

Časovač se používá ke generování synchronizačních polí mezi rámce. Každý rámec typu Request musí předcházet synchronizační pole. Časovač se plní podle použití na předvolby T_{ID1}, T_{ID2}.

2.3.6.3 Slot Timer

Používá se ke zjištění, zda adresovaná stanice reaguje na rámec typu Request nebo Token. Časovač se naplní na hodnotu T_{SL} a spustí vždy po vyslání akčního rámce nebo pověření. Když během doby časování nedojde k přijetí žádného rámce (časovač vypršel), vyhodnotí se to jako chyba a stanice se pokusí vyslat opětovný požadavek.

2.3.6.4 Time – Out Timer

Sleduje aktivitu na sběrnici. Po přijetí posledního bitu rámce se naplní násobkem času Slot Time (viz odstavec 2.3.5.9 – Time Out Time) a spustí se. Dekrementace proběhne každý bitový okamžik, dokud se nepřijme nějaký rámec. Jestliže časovač vyprší a žádný rámec se nepřijal, došlo k závažné chybě (ztráta pověření) a musí se znovu inicializovat logický kruh. Vrstva FMA1/2 přijme oznámení o výskytu této události.

2.3.6.5 Syn Interval Timer

Tento časovač se používá ke zjištění doby mezi dvěmi po sobě jdoucími synchronizačními poli. Jakmile se dojde k synchronizaci přijímače (na sběrnici se objeví synchronizační pole), naplní se časovač na hodnotu T_{SYNI} a spustí dekrementaci s četností každý bitový okamžik. Když časovač dočítá k nule dříve, než se na sběrnici objeví další synchronizační pole, vyhodnotí se chyba na sběrnici jako stálá nula nebo stále se měnící posloupnost a informuje se vrstva FMA1/2.

2.3.6.6 GAP Update Timer

Vypršení tohoto časovače indikuje nutnost aktualizace GAP seznamu. Aktualizace může trvat i několik oběhů pověření. Když je dokončená, časovač se naplní na hodnotu několikanásobku T_{TR} (viz odstavec 2.3.5.10 – GAP Update Time).

Když aktivní stanice přejde do stavu Listen Token, Idle Timer se naplní na hodnotu T_{SYN}, Time – Out Timer hodnotou T_{TO}, Syn Interval Timer hodnotou T_{SYNI} a ostatní časovače se deaktivují.

2.3.7 Čítače

Definice čítačů je volitelná, záleží na konkrétní aplikaci, zda se čítače použijí k realizaci či ne. Norma Profibus [13] doporučuje pro aktivní stanici definovat následující pár čítačů:

a) Čítač odeslaných rámců (Frame Sent Count) pro rámce typu Request kromě služeb SDN a FDL Status.
b) Čítač opětovně odeslaných rámců (Retry Count)

Přechodem stanice do stavu Listen Token nebo Passive Idle se čítače vynulují a aktivují. Když dosáhnou maximální stanovené hodnoty, oba současně se vynulují a aktivují. Z vrstvy FMA1/2 jsou tyto čítače přístupné pomocí funkcí set/read value.

3. USB (Universal Series Bus)

3.1 Úvod

Původní motivací ke vzniku USB byly především tyto tři důvody:

- Spojení PC s telefonem
  Je zřejmé, že postupem času se stále více vyžaduje spolupráce počítačů a telefonních komunikačních zařízení. Bohužel výpočetní a komunikační technika se vyvíjela nezávisle, tak nebyl dán žádný standard. USB tvoří všude přítomné spojení PC a sdělovací techniky, které má širokou oblast použití.
- Zjednodušení používání periferií
  Špatná flexibilita v konfiguraci PC bránila dalšímu rozvoji. Příjemná grafická rozhraní a vývoj softwarového vybavení usnadňují provádění změn v konfiguraci počítače, avšak z pohledu konečného uživatele periferie jako např. sériový/paralelní port, klávesnice, myš nemají charakter plug and play.
- Zvýšení počtu rozhraní
  Nedostatek vstupně/výstupních konektorů na počítači omezuje počet současně připojených zařízení jako jsou modemy, myši, klávesnice, skenery, tiskárny, atd. Stávající rozhraní jsou určeny pouze pro několik druhů zařízení.

Kolem roku 1994 vznikla specifikace USB pro rychlosti 1,5 MB/s a 12 MB/s. Nárůstem výpočetní rychlosti a potřebou přenášet větší množství dat byla později škála rozšířena o další přenosovou rychlost 480 MB/s. Je to levná, obousměrná a rychlá sběrnice, kde lze dynamicky za běhu systému připojovat/odpojovat zařízení.

3.2 Architektura

Komponenty USB systému:

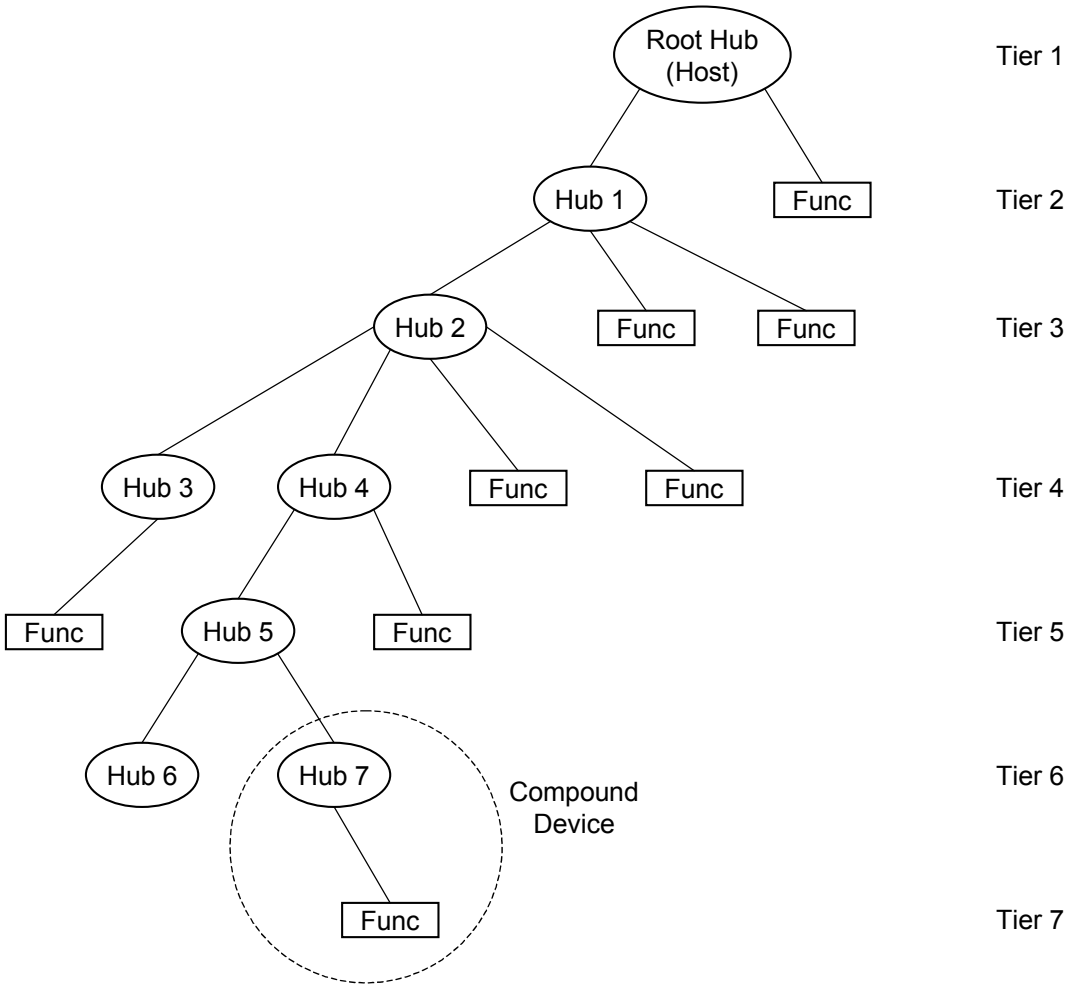
- USB propojení
- USB zařízení
- USB hostitel

USB propojení definuje způsob spojení a komunikace zařízení s hostitelem. Zde rozlišujeme další parametry:

- Topologie sítě
- Relace mezi vrstvami sítě
- Modely datových toků

3.2.1 Topologie sítě

USB má obecně topologii typu tiered star.

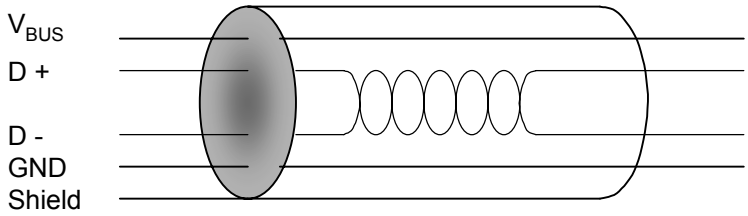


Obrázek 3-1: USB – Topologie sítě

Obrázek 3-1 znázorňuje způsob propojení jednotlivých zařízení (devices) s hostitelem (host). Hostitel je umístěný vždy na vrcholu struktury. Zařízení (device) je zde reprezentováno pomocí Hub nebo Func. Každá spojovací čára znamená propojovací kabel. Každý hub představuje střed každé hvězdy. Maximální hloubka 7 vychází z dovolených zpoždění signálu pro propojovací kabely a hub(y). Složené zařízení (Compound Device) obsazuje vždy dvě vrstvy, proto není povoleno v úrovni sedm. V sedmé vrstvě jsou povoleny pouze zařízení typu func. Maximální počet hub(ů) na komunikační cestě od hostitele směrem ke koncovému zařízení (Func) je pět.

3.3 Fyzická vrstva

Signál se přenáší pomocí čtyřvodičového stíněného kabelu, který obsahuje dva datové a dva napájecí vodiče, viz. Obrázek 3-2.



Obrázek 3-2: USB – Propojovací kabel

USB pracuje v následujících módech:

Mód	Přenosová rychlost [Mb/s]
high-speed	480
full-speed	12
low-speed	1,5

Tabulka 3-1: USB – Přenosové rychlosti

Komunikace v módu high-speed je umožněna pouze mezi hostitelem a hub(em). Přenos mezi hub(y) a zařízeními se uskutečňuje buď v režimu full-speed, nebo low-speed. To je výhoda pro přístroje, které byly zkonstruovány podle specifikace 1.1.
USB používá diferenční přenos po vodičích D+ a D- s kódováním signálu NRZI (Non Return to Zero Invert). Aby byla zaručena synchronizace vysílače a přijímače, kóduje se s použitím bit-stuffing (vkládání redundantních bitů). Každý konec vodiče musí mít předepsané zakončení a je různé pro nadřazené (attach) a podřazené (detach) zařízení. Typ zakončení se liší také pro zařízení pracující v high-speed/full-speed a low-speed módu. Napájení je stanoveno na 5V na straně hostitele. Napájecí vodiče mají dvě funkce:

- Rozvádějí elektrickou energii od hostitele k zařízením, které nemají vlastní zdroj.
- Hostitel řídí pomocí těchto vodičů stavy zařízení jako je suspend, resume.

3.4 Protokol sběrnice

Všechny přenosy po sběrnici zahajuje hostitel. Ten vyšle tzv. Token packet, který udává typ a směr přenosu, adresu zařízení a číslo koncového bodu (endpoint number). Poté probíhá vlastní komunikace a nakonec iniciátor přenosu vyšle paket, který přenos ukončí. Příjemce dat obvykle vyšle zpět paket indikující bezchybnost přenosu. Spojení mezi hostitelem a zařízením se nazývá roura (pipe). Rozlišujeme dva typy rour:

- Stream pipe
  - Nemá definovanou strukturu datového toku na úrovni USB specifikace.
  - Mají přiřazen typ přenosu, přenosovou rychlost a vlastnosti koncového bodu, jako je např. směr přenášených dat a velikosti vyrovnávacích pamětí.
  - Vznikají až po nastavení zařízení.

- Message pipe
  - Struktura definovaná USB specifikací.
  - Jeden z typů přenosové roury Message pipe – Default Control Pipe vznikne, jakmile se zařízení připojí k napájení.

3.5 Robustnost

Každý paket obsahuje samostatné kontrolní pole pro řídící a datové části. Protokol kontroluje CRC pro každé pole odděleně. Zabezpečení chybovosti odhalí se 100% úspěšností všechny jednoduché a dvojité chyby bitů. USB ošetřuje chyby hardwarově a softwarově. Hardwarové ošetření spočívá v tom, že když dojde k detekci chyby, hostitel opakuje přenos. Když se během následujících dvou pokusů přenos nepodaří bez chyb(y), je klientský software informován o chybě a záleží na konkrétní implementaci, jakým způsobem problém řeší.

3.6 Připojení zařízení ke sběrnici

Každé zařízení se připojuje k USB prostřednictvím jiného, které nazýváme hub. Každé zařízení typu hub má své příznakové bity, které indikují připojení dalších zařízení ke konkrétnímu hub(u). Hostitel se táže na tyto příznaky. Při zjištění připojení zařízení k síti hostitel povolí příslušný port a adresuje zařízení prostřednictvím řídící roury na implicitní adrese. Hostitel přidělí zařízení jedinečnou adresu a rozhodne, zda nově připojená stanice je hub nebo function. Hostitel založí cíl řídící roury s použitím přidělené adresy a koncového bodu (endpoint) číslo nula. Pokud připojené zařízení je typu hub, opakuje se výše uvedený postup pro každé z připojených zařízení k hub(u). Když se jedná o zařízení typu function, hlášení o připojení je obslouženo vhodným softwarovým vybavením hostitele.

3.7 Odpojení zařízení od sběrnice

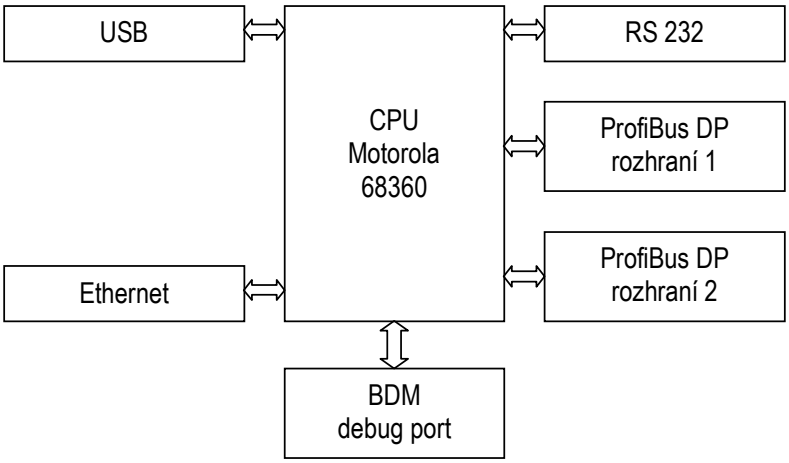
Jestliže je zařízení odpojeno z nějakého konektoru umístěného na hub(u), hub zakáže port a nastaví indikaci o odpojení pro hostitele. Událost o odpojení je obsloužena příslušným USB systémovým softwarem. Když odpojené zařízení je hub, systémový software USB musí odstranit i všechny dříve připojená zařízení k odstraněnému hub(u).

4. Hardware

4.1 Popis celku

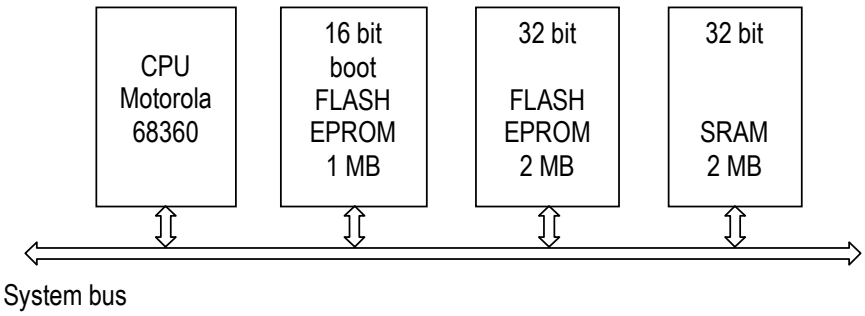
Jádrem celého systému je mikrořadič Motorola MC68EN360. Ten byl vybrán proto, že kromě výkonného 32bitového jádra obsahuje na jednom čipu ještě riskový komunikační procesor, který umožňuje pohodlnou obsluhu sériových kanálů jako je např. RS232, ProfiBus. Procesor disponuje 32bitovou datovou sběrnicí a programovatelnými signály pro výběr čipu, proto je připojení periferních zařízení velmi snadné. K němu jsou připojeny paměti a periferní obvody.
Při výběru jednotlivých součástek jsem vzhledem k dostupnosti volil pětivoltové provedení. S ohledem na velikost (snaha o minimalizaci) byla většina součástek použita v technologii SMD (Surface Mount Devices), pouze několik čipů (optočleny) bylo zapájeno klasicky, a to z důvodu nedostupnosti na našem trhu v provedení SMD.

Připojené periferie znázorňuje Obrázek 4-1.



Obrázek 4-1: Připojení periferií

Z hlediska bezpečnosti jsou rozhraní RS232, ProfiBus připojeny přes optické oddělovače. Připojení pamětí a procesoru k systémové sběrnici dokumentuje Obrázek 4-2.



Obrázek 4-2: Připojení pamětí

K uložení základního programu a zavaděče slouží dvě paměti typu Flash Eprom. Jedna 16bitová o kapacitě 1 MB a jedna 32bitová o kapacitě 2 MB. Paměti slouží též k uložení mikrokódu Profibus(u) pro komunikační procesor. Šířky datových sběrnic jednotlivých čipů jsou: AM29F800 – 16 bitů, M68AF511 – 8 bitů. Použité paměti jsem sestavil složením více paměťových obvodů, viz Tabulka 4-1.

Typ	Výrobce	Označení	Počet [ks]	Šířka sběrnice [bit]	Kapacita [MB]
Flash EPROM	AMD	AM29F800BB	1	16	1
Flash EPROM	AMD	AM29F800BB	2	32	2
SRAM	Thomson	M68AF511AL	4	32	2

Tabulka 4-1: Použité paměti

Po zapnutí napájení nebo po hardwarovém resetu mikroprocesoru se předpokládá, že se program nachází na adrese 0. Aby se procesor po zapnutí rozběhl se správnou konfigurací šířky sběrnice pro natažení programu (16 bitů), je třeba nastavit následující signály:

Signál	Logická úroveň	Jumper	Stav
Config 0	1	J39	otevřený
Config 1	0	J42	uzavřený
Config 2	1	J44	otevřený

Tabulka 4-2: Jumpery Config 0, 1, 2

Podrobný popis viz kapitola 2.1.13 Initial Configuration Pins v [1].

4.2 Procesorová část

4.2.1 Základní parametry mikrořadiče

Mikrořadič se skládá ze tří základních bloků:
– CPU32+ (32bitový mikroprocesor)
– SIM 60 (modul integrace systému)
– CPM (komunikační procesor)

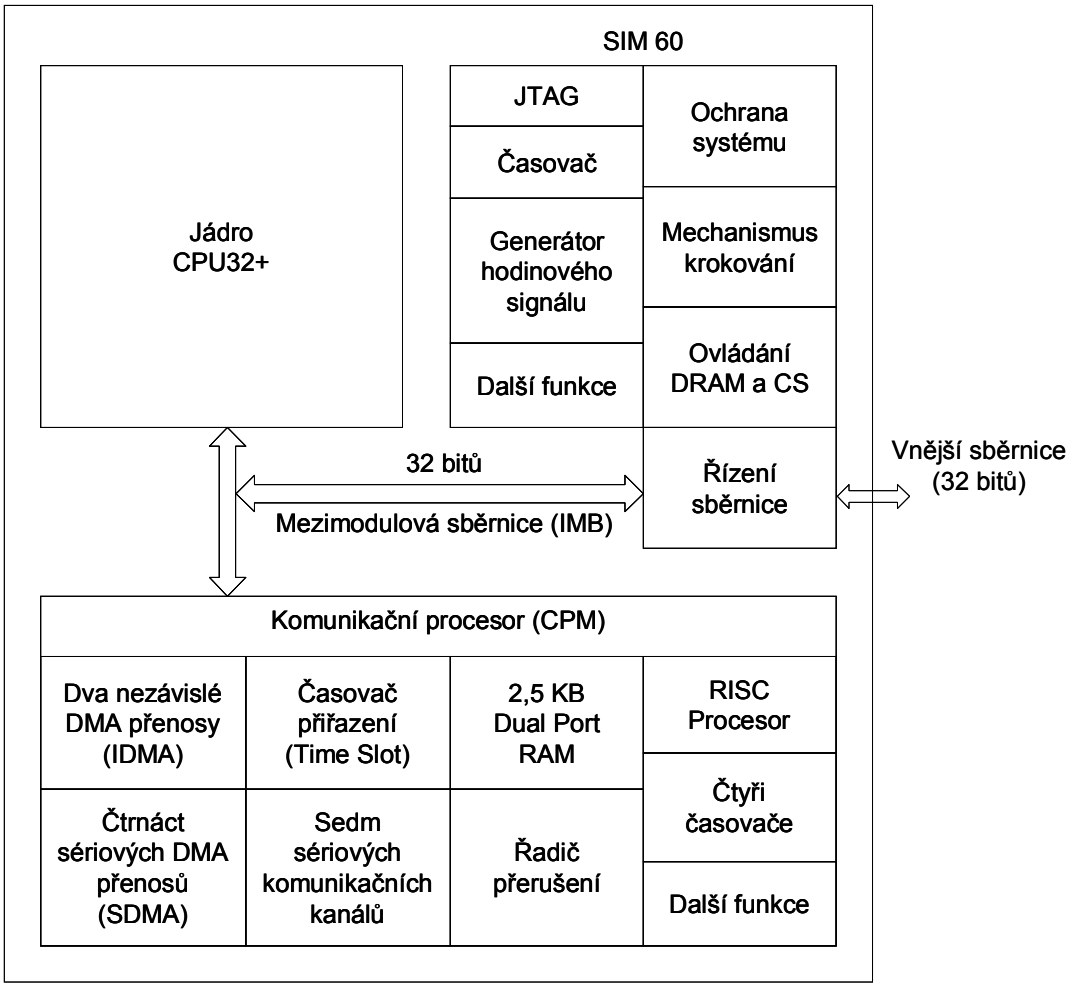
- CPU32+ – 4,5MIPS při frekvenci 25MHz
  – 32bitová verze procesoru (plně kompatibilní s CPU32)
  – BDM (Bacground Debud Mode) velmi užitečné při ladění a nahrávání programu
  – bytová adresace
- Dynamicky volená šířka sběrnice pro 8, 16 a 32 bitů.
- 32 adresových linek
- Plně statický návrh – umožňuje pohodlné ladění a hledání závad hardwaru.
- Možnost vypnutí CPU – vhodné pro využití pouze komunikačního procesoru.
- Řadič pamětí – obsluhuje až osm pamětí
  – obsahuje řadič pro dynamické paměti (DRAM)
  – každá pozice může fungovat jako CS (Chip Select) nebo dynamicky

– možnost vkládání čekacích impulsů při spolupráci s pomalou pamětí
– rozhraní pro paměti typu SIMM (Single In-Line Memory Modules), SRAM (Static Random Access Memory), EPROM (Electrically Programmable Read Only Memory), Flash EPROM, atd.
– čtyři linky CAS, RAS a jedna OE
– bootovací výběr čipu aktivovaný při resetu.
(Šířka sběrnice 8, 16 nebo 32 bitů)

- Čtyři časovače pro univerzální použití – čtyři 16bitové nebo dva 32bitové
  – mód Gate umožňuje povolení/zakázání čítání
- Dva nezávislé DMA – Single Address mód - nejrychlejší přenosy
  – Double Address mód - pro různé šířky sběrnice na straně zdroje a cíle přenášených dat
  – 32bitové vnitřní i vnější přenosy
- Modul integrace systému (SIM60) – sledování sběrnice, odhalení dvojité chyby na sběrnici
  – srogramový Watchdog
  – obvody pro časování
  – logika zastavování programu (Breakpoints)
  – JTAG - testovací port odhaluje závady hardwaru
- Přerušení – sedm vnějších - IRQ linky
  – 12 vodičů z vst/výst bran se schopností vyvolávat přerušení
  – 16 vnitřních zdrojů přerušení
  – programovatelná priorita mezi sériovými komunikačními linkami (SCC)
  – programovatelný požadavek nejvyšší priority
- Komunikační procesor (CPM) – typu RISC (Reduced Instruction Set Controller)
  – 2,5 kB Dual-Port RAM
  – podporuje současnou práci všech sériových kanálů
  – 14 sériových DMA kanálů
  – každý sériový kanál může mít své vlastní vodiče
- Čtyři generátory přenosové rychlosti
  – nezávislé - mohou být přiřazeny k libovolnému kanálu
  – umožňují dělat změny rychlosti během operací komunikačního procesoru
- Čtyři sériové komunikační kanály (SCC)
  – Ethernet/IEEE 802.3 na SCC1 - 10Mb/s
  – universální asynchronní vysílač/přijímač (UART)
  – synchronní UART
  – implementovaná kontrola zabezpečení cyklickým kódem (CRC)
  – ProfiBus (volbou mikrokódu v paměti)
  – mnoho dalších typů převážně volbou mikrokódu
- Dva kanály typu SMC – UART
  – může být připojen do časově přepínaného kanálu (TDM)
- Přiřazení do časového pásma (Time Slot Assigner)
- Podporuje dva nezávislé TDM kanály
- Definovaných 240 pinů, pouzdra PGA (241 pinů), PQFP (240 pinů) – použité provedení

4.2.2 Blokový popis

Blokové schéma mikrořadiče je znázorněno na obrázku 4-3.



Obrázek 4-3: Blokové schéma mikrořadiče Motorola 68360

Popis hlavních bloků:

CPU32+

Je to 32bitový procesor (z hlediska vnitřní struktury), který je připojen k mezimodulové sběrnici prostřednictvím 32bitové sběrnice. Umožňuje provádět operace s vnějšími 32bitovými operandy během jednoho cyklu sběrnice, zapisuje nebo čte 32bitové datové typy v jednom cyklu sběrnice. Má stejný instrukční soubor jako CPU32, obsahuje standardní BDM. Výkon zpracování je 4,5 MIPS při frekvenci hodin 25MHz. Pro zvýšení výkonu je možné připojit rychlejší mikroprocesor (MC68040) a pracovní výkon se zvýší na 22 MIPS při 25MHz.

SIM60 (System Integration Module)

Začleňuje do systému další důležité funkce. Obsahuje řadič dynamických pamětí (DRAM) a modulů SIMM. Dynamicky volí šířku sběrnice, takže při zvolené šířce datové sběrnice 32 bitů může obsluhovat periferie s šířkou 8, 16 a 32 bitů. Při zvolené šířce datové sběrnice 16 bitů pracuje s 8bitovým nebo 16bitovým periferním zařízením.

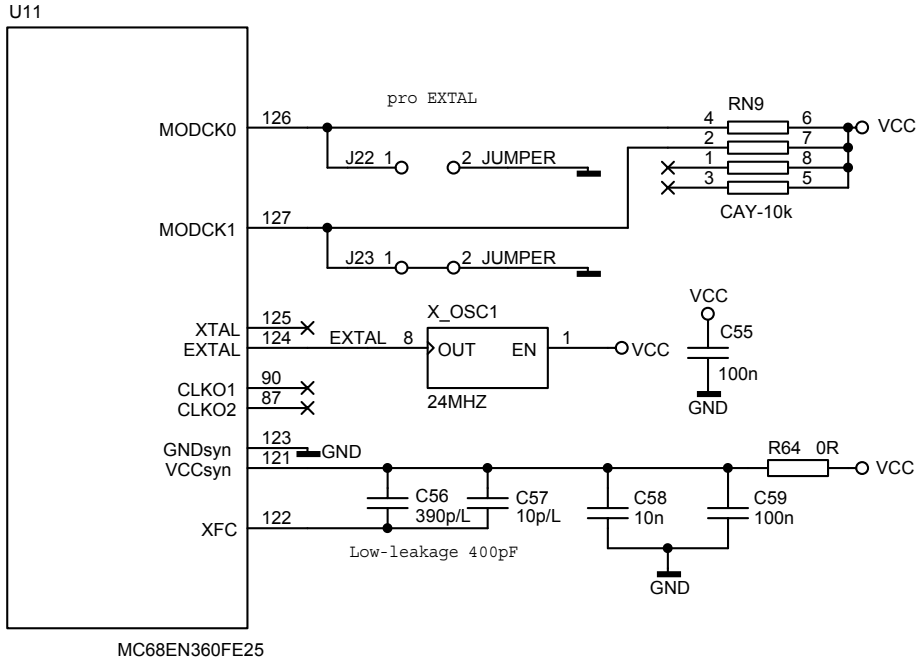
CPM – komunikační procesor

Jedná se o procesor typu RISC (Reduced Instruction Set Processor), který se stará o spojení s okolním světem. Je vybaven čtyřmi komunikačními kanály typu SCC, dvěma SMC a jedním rozhraním typu SPI. Přenosová rychlost se generuje pomocí čtyř generátorů (BRG). Nastavení požadované funkce se provádí zápisem do příslušných registrů, popřípadě zavedením mikrokódu do vnitřní paměti RAM. Disponuje 2,5Kbytovou dvojbránovou pamětí (Dual Port RAM), která se používá k předávání parametrů mezi CPU a CPM, uložení mikrokódu pro CPM. Obsahuje další mikrořadič, který obsluhuje dva nezávislé DMA přenosy. Jeho součástí jsou i čtyři časovače pro libovolné použití. Buď pracují všechny čtyři nezávisle v režimu 16bitů nebo lze spojit dvojici na časovač velikosti 32 bitů.

Podrobný popis mikrořadiče najdeme v [1].

4.2.3 Časování

SIM60 má obvody pro výrobu hodinového signálu při použití krystalu nebo krystalového oscilátoru. V mé aplikaci jsem použil krystalový oscilátor o frekvenci 24 MHz a připojení vypadá následovně:



Obrázek 4-4: Schéma časování MC68360

Popis důležitých vývodů:

MODCK0-1 – vývody pro nastavení požadované funkce obvodů pro výrobu hodinového signálu (násobení frekvence)

XTAL – první vývod pro připojení krystalu, při použití oscilátoru se nechává nezapojený

EXTAL – druhý vývod pro připojení krystalu, zde se připojuje krystalový oscilátor

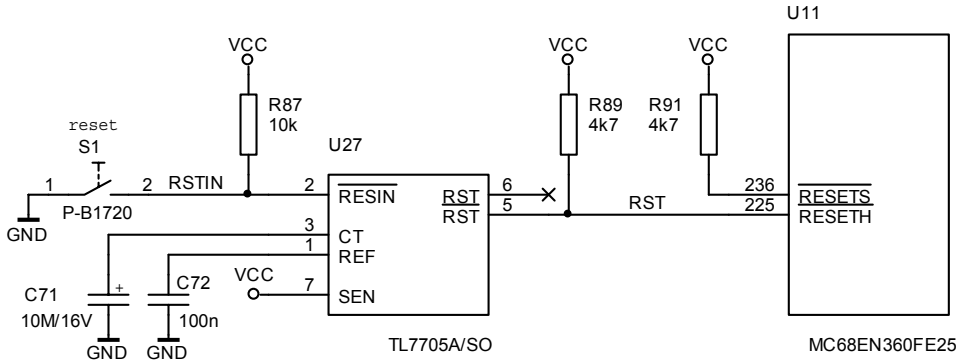
CLKO1 – výstup systémových hodin (24 MHz)

CLKO2 – dvojnásobná frekvence systémových hodin (48 MHz)
GNDsyn – napájení generátoru hodin, zem
VCCsyn – napájení generátoru hodin, +5V
XFC – zde je nutné připojit kondenzátor podle velikosti násobícího faktoru hodin, pro faktor 1 je doporučen kondenzátor s kapacitou 400pF, který je složen z paralelní kombinace 390nF a 10nF

Detailní popis časovacích obvodů je uveden v kapitole 6.5 – SIM60 System Clock generation v [1].

4.2.4 Resetování

Mikroprocesor uvnitř rozlišuje 7 resetovacích zdrojů. Jako vnitřní reprezentace se používají čtyři linky, které se aktivují v závislosti na zdroji resetu (podrobnosti v [1], kapitola 4.7). Mikroprocesor disponuje dvěma resetovacími vstupy – RESETH (hard reset), RESETS(soft reset). K úplnému resetování obvodu postačí aktivovat RESETH. Aby se procesor zaručeně resetoval, je třeba dodržet požadovanou dobu signálu. Na desce plošných spojů je osazen mikrospínač (S1) a resetovací obvod, který upravuje délku signálu. Délka signálu je nastavena kondenzátorem C71 na dostatečně dlouhou dobu – 130ms. Vstup RESETS je připojen na +5V a je trvale pasivní.



Obrázek 4-5: Schéma resetování MC68360

Popis resetovacího obvodu TL7705A nalezneme v [6].

4.2.5 Přiřazení pinů

Pro lepší orientaci při řešení hardwarových nebo softwarových problémů slouží následující tabulky. Ukazují připojení periferií k příslušným vývodům mikrořadiče a funkci pinu z hlediska mikrořadiče.

Pin	PA 15	PA 14	PA 13	PA 12	PA 11	PA 10	PA 9	PA 8
In/Out	O	O	O	O	I	I	I	I
Přiřazení	D7	D6	D5	D4	CLK4	CLK3	CLK2	CLK1
Skupina	Diody				Ethernet – Clock			

Pin	PA 7	PA 6	PA 5	PA 4	PA 3	PA 2	PA 1	PA 0
In/Out	O	I	O	I	O	I	O	I
Přiřazení	TxD4	RxD4	TxD3	RxD3	TxD2	RxD2	TxD1	RxD1
Skupina	Profibus2/UART		Profibus1		Profibus2/Ethernet		Profibus1/Ethernet	

Tabulka 4-3: Přiřazení pinů brány A

Pin	PB 17	PB 16	PB 15	PB 14	PB 13	PB 12	PB 11	PB 10	PB 9	PB 8
In/Out	I	I	I	I	I	I	I	O	I	I
Přiřazení	Eth	Eth	-	-	-	-	SM RxD2	SM TxD2	$\overline{SM}$ $\overline{SYN2}$	-
Skupina	Ethernet		-	-	-	-	UART			-

Pin	PB 7	PB 6	PB 5	PB 4	PB 3	PB 2	PB 1	PB 0
In/Out	I	O	O	I	I	I	I	O
Přiřazení	Eth	$\overline{Done1}$	$\overline{Dack1}$	$\overline{Dreq1}$	Eth	Eth	Eth	Suspend
Skupina	Ethernet	DMA řadič			Ethernet			USB

Tabulka 4-4: Přiřazení pinů brány B

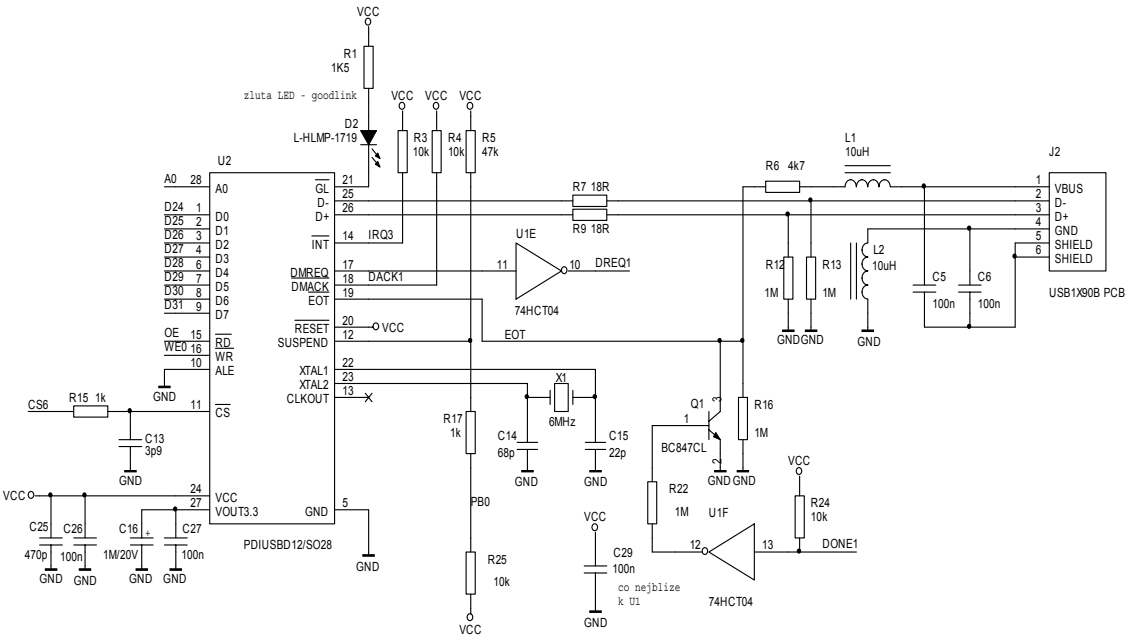
Pin	PC 11	PC 10	PC 9	PC 8
In/Out	I	I	I	I
Přiřazení	-	$\overline{CTS4}$	-	$\overline{CTS3}$
Skupina	-	Profibus2/UART	-	Profibus1

Pin	PC 7	PC 6	PC 5	PC 4	PC 3	PC 2	PC 1	PC 0
In/Out	I	I	I	I	O	O	O	O
Přiřazení	$\overline{CD2}$	$\overline{CTS2}$	$\overline{CD1}$	$\overline{CTS1}$	$\overline{RTS4}$	$\overline{RTS3}$	$\overline{RTS2}$	$\overline{RTS1}$
Skupina	Ethernet				Profibus2	Profibus1	Ethernet	

Tabulka 4-5: Přiřazení pinů brány C

4.3 USB

Při styku se sběrnicí USB (Universal Series Bus) mikroprocesor komunikuje prostřednictvím PDIUSBD12 od firmy Philips. Jedná se o obvod, který převádí sériový tok dat z USB na paralelní přenos mezi procesorem. Obvod je složený z mikroprocesoru, vnitřní RAM, jednotky řízení paměti, analogových vstupních obvodů pro připojení k USB, regulátoru napětí, generátoru hodin. Regulátor napětí při napájení z 5V zdroje vyrábí 3,3V pro analogové vstupní obvody. Generátor hodinového signálu obsahuje násobičku, která dokáže vyrábět signál o frekvenci 6 – 48 MHz. Obvod disponuje 8 bitovou datovou sběrnicí a jedním adresovým bitem. K nastavení požadované funkce a parametrů se používají vnitřní registry. Ty se dají modifikovat pomocí příkazů zadávaných po paralelním rozhraní (datová sběrnice). Adresní bit udává, zda obvod bude přijímat kód instrukce (přístupové kódy ke konfiguračním registrům) nebo datovou část přenosu (modifikace nebo čtení registrů). Když bit A0 = 1, jde o zápis příkazové instrukce, A0 = 0 signalizuje datovou fázi přenosu. Úspěšné spojení s USB adapterem je indikováno pomocí vestavěného GoodLink indikátoru a připojené led (D2). Obvod podporuje DMA přenos dat mezi vnitřní a vnější pamětí. Tento přenos obsluhuje DMA řadič, integrovaný na mikroprocesoru Motorola 68360. Je zde podpora práce v single address módu nebo dual address módu. Periferní zapojení pro styk s USB je vidět na obrázku 4-6.



Obrázek 4-6: Schéma části USB

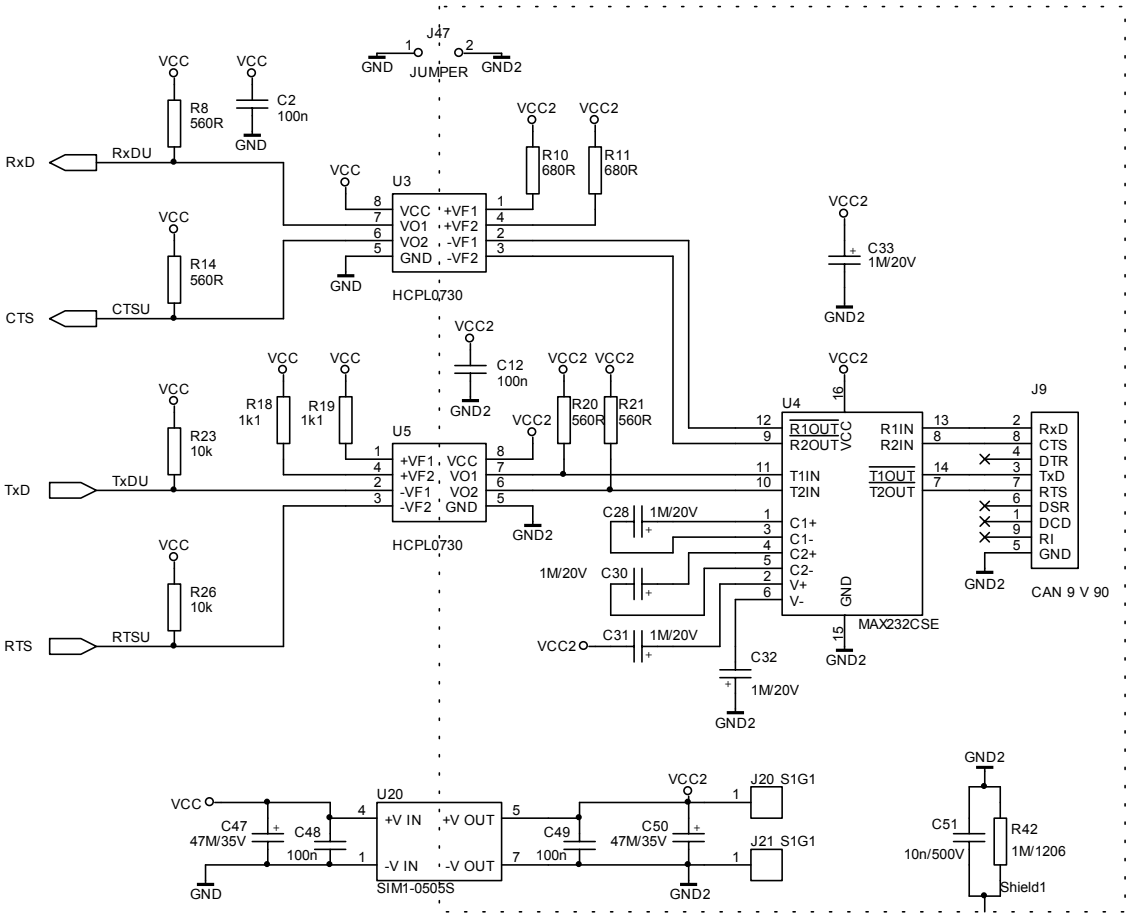
Konektor pro připojení USB kabelu je typu B, protože jde o downstream port. USB řadič na PC naopak používá konektor typu A.

Při pohledu na požadované průběhy signálů při čtení a zápisu (viz str. 26 v [7]) zjistíme, že obvod potřebuje přesah signálu $\overline{CS}$ přes signál $\overline{RD}$ při čtení nebo $\overline{WR}$ při zápisu a to minimálně o 5ns. Řídící procesor Motorola však tento překryv nezajišťuje. Problém lze odstranit např. přidáním integračního článku sestaveného z R15 a C13. Integrační článek způsobí i zpoždění signálu $\overline{CS}$ na začátku čtecího/zápisového cyklu. To při

dodržení zpožďovací doby 5ns není na závadu. Parametry součástek nelze přesně stanovit, protože neznáme charakteristiku vstupu $\overline{CS}$. Je vhodné tyto hodnoty doladit měřením na realizovaném zařízení.
Podrobnosti o obvodu PDIUSBD12 lze najít v [7].

4.4 RS-232

K jednoduché komunikaci mezi zařízením a PC je zde použito rozhraní RS 232.
Při zapojování konektorů mezi ProfiUsb a druhé zařízení (PC) se stane, že potenciály na obou stranách spojení nejsou na stejných úrovních, to by mohlo v nejhorším případě způsobit destrukci periferie na jedné nebo druhé straně. Z tohoto důvodu jsem přístroj vybavil optickými oddělovači (optočleny), ty potenciálově separují periferní obvod od ostatních součástí zařízení. Ke konverzi napěťových úrovní je použitý budič RS 232 od firmy Maxim – MAX232CSE, který je napájen z DC-DC měniče.
Tento interface se osvědčil jako velmi vhodný při ladění softwaru. Pomocí sériové linky je možné prostřednictvím znaků ovládat program, který běží na hlavním procesoru. Schéma zapojení ukazuje Obrázek 4-7.



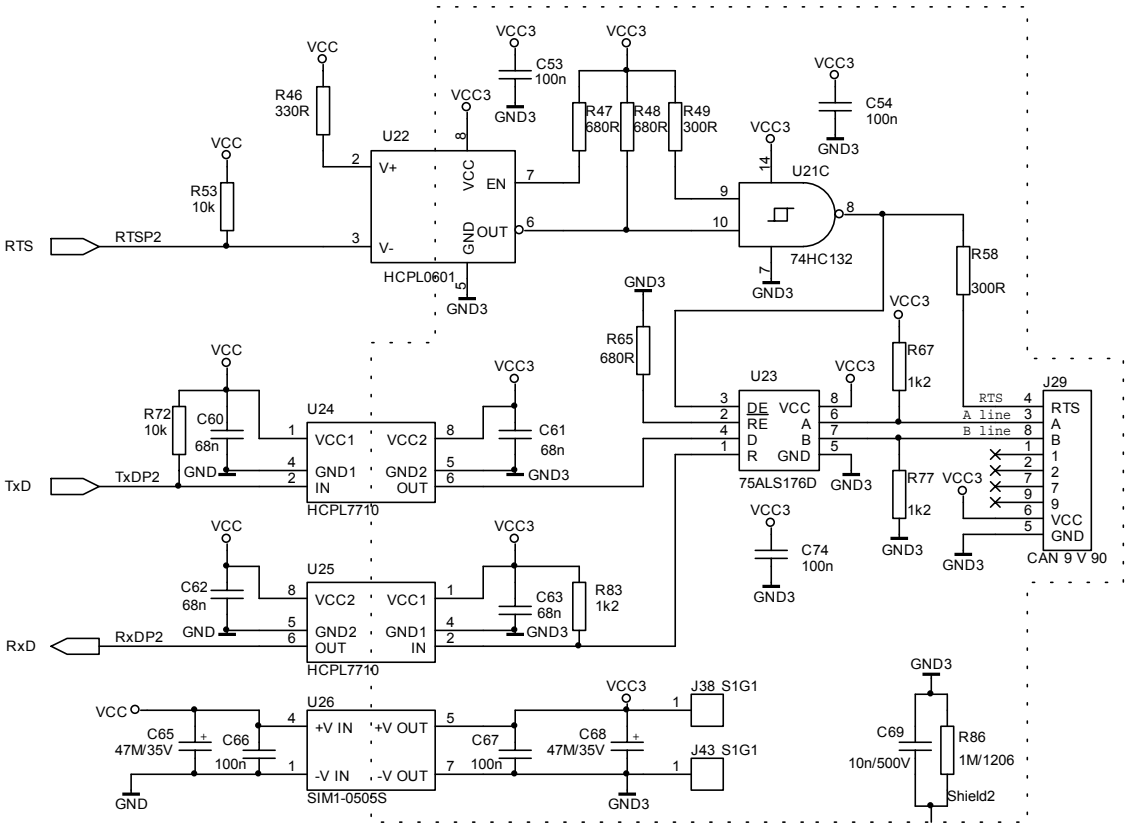
Obrázek 4-7: Schéma části RS-232

Budič MAX232CSE je jedna z nejjednodušších modifikací budičů RS232 z výrobní řady Maxim. Složitější sourozenci obsahují navíc logiku pro přechod obvodu do módu s nízkou spotřebou a/nebo více budících kanálů. V této aplikaci naprosto vyhovuje použitý model.

Jeho úkolem je transformovat signály z 0 – 5V, které používá mikroprocesor (RTS, TxD), na –12 – +12V, ty jsou standardní pro sériové rozhraní RS232 a naopak (CTS, RxD). Protože k dispozici je pouze napájecí napětí o velikosti 5V, MAX232 vyrábí pomocí nábojových pump napětí cca ±12V. Ke své činnosti potřebuje pouze několik vnějších kondenzátorů (obr. 4-7). Podrobnosti v [10].
K optickému oddělení potencialů jsem zvolil dva dvojité optočleny HCPL0730, jsou osazeny technologií SMD. Zde bych upozornil na správnou volbu předřazeného rezistoru vnitřní LED a v neposlední řadě na výstupní pull-up resistor, tato dvojice zásadně ovlivňuje modulační a tím i přenosovou rychlost kanálu, která je vždy na úkor spotřeby obvodu. K dosažení sepnutí výstupní brány je třeba vstupní proud min. 1,6mA. Čím vyšší je budící proud, tím je možné snižovat hodnotu výstupního pull-up rezistoru a tím pádem zvyšovat přenosovou rychlost (vyplývá to z [9]).
Výstupní kanály z procesoru (RTS, TxD) jsou limitovány maximálním proudem (v úrovni log 0), ten činí 3,2mA pro pětivoltovou modifikaci. Ten omezuje výběr obou rezistorů optočlenu U5 pro vysílání.
U opačného směru (optočlen U3) je rozhodující výstupní proud budiče (MAX232).

4.5 Profibus

Zařízení je vybaveno dvěma identickými rozhraní k síti Profibus. Obě rozhraní jsou schopné pracovat současně v režimu Profibus DP.
Interface k Profibusu je řešen pomocí budičů RS485 (75ALS176D, podrobnosti v [6]), o optické oddělení potencialů se starají optočleny od firmy Agilent Technologies – HCPL0601 (SMD) a HCPL7710. Zde je důležité věnovat velkou pozornost u RS232 již zmíněným rezistorům k dosažení požadované rychlosti. Mikroprocesor Motorola 68360 (za předpokladu systémových hodin 24MHz) je schopen pracovat v režimu Profibus s přenosovou rychlostí až 1,5Mb/s.
Pro přijímací/vysílací linky jsou použity HCPL7710. Ty mají integrovaná vstupní a výstupní hradla včetně všech potřebných rezistorů. Není třeba již jejich periferie opatřovat dalšími odpory. Podrobnosti v [9]
U kanálu RTS používám oddělovače HCPL0601. U nich je potřeba vhodně zvolit vstupní a výstupní rezistory, protože jejich vstup tvoří vývod od vnitřní LED a na výstupu nalezneme tranzistor s otevřeným kolektorem. Podrobnosti v [9].
Schéma této části zapojení dokumentuje Obrázek 4-8.



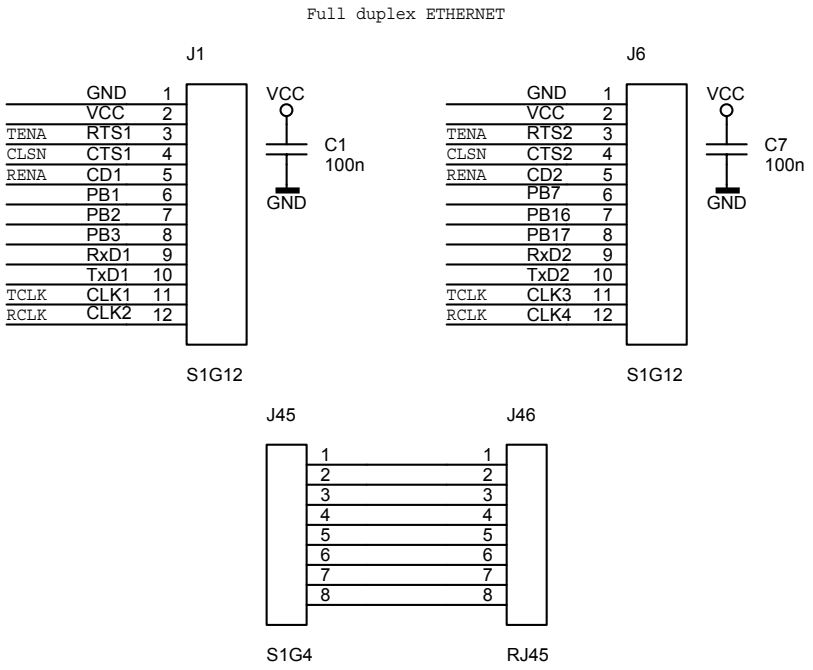
Obrázek 4-8: Schéma části Profibus

4.6 Ethernet

Komunikační procesor Motorola 68**EN**360 má podporu pro komunikaci po **E**ther**N**etu. Pro připojení procesoru k ethernetu je třeba ještě přidat periferní obvody, např. ethernetový čip AM7992B a impedanční tranformátor.

Deska ProfiUSB má pro připojení periferního obvodu pracujícím v režimu full-duplex připravené konektorové výstupy (viz Obrázek 4-9).

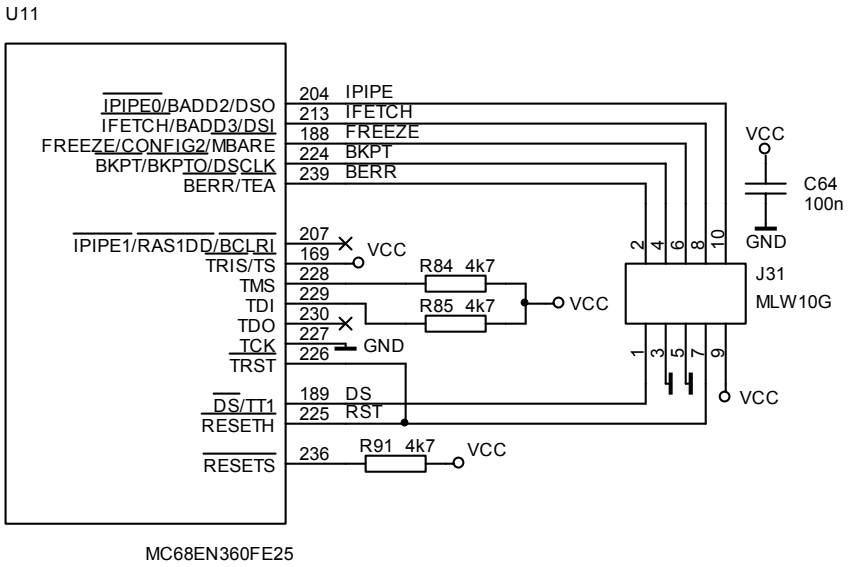
Konektory J1 a J6 jsou určeny k připojení periferie k procesoru. J45 slouží k propojení výstupu ethernetového budiče na fyzický konektor, který je umístěný také na základní desce (ProfiUSB).



Obrázek 4-9: Konektory pro připojení Ethernet kontroléru

4.7 BDM

Tento typ rozhraní používá Motorola u svých mikroprocesorů výhradně pro potřeby programátora při vývoji aplikací. V mikrořadiči jsou implementovány mikroprogramové instrukce, které dovolují pracovat přímo s jádrem mikroprocesoru. Odpadá tak nutnost použití vnějších emulátorů. Připojení lze uskutečnit k paralelnímu portu PC. Podrobnosti o BDM nalezneme v [1] a [20].
Na desce plošných spojů je umístěn desetipinový BDM konektor , jehož zapojení je:

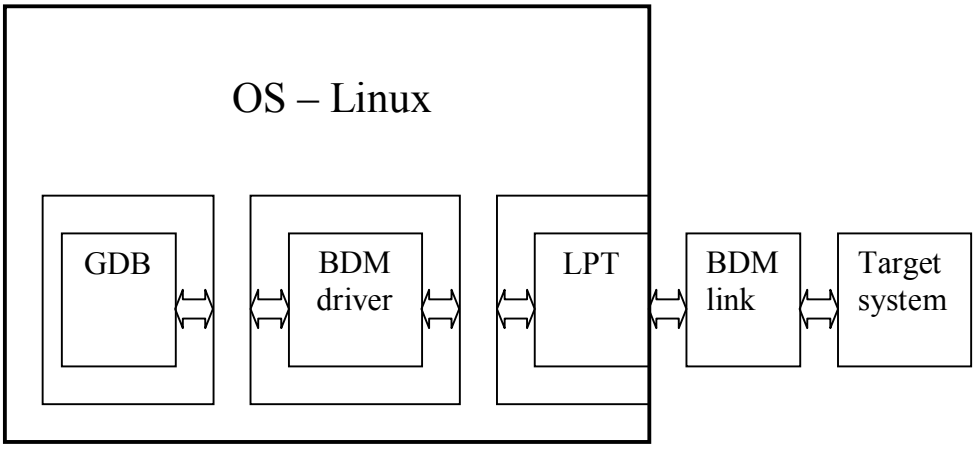


Obrázek 4-10: Schéma připojení BDM

5. Software

5.1 Vývojový řetězec

Při vývoji softwaru a testování zařízení jsem používal následující vybavení. PC s operačním systémem Linux (Mandrake 8.0), nainstalovaný BDM driver, prostředí pro debugování programu – Gnu DeBugger (GDB), jeho grafickou nadstavbu Data Desktop Debuger (DDD). Programování jsem prováděl v jazyce C s pomocí grafického editoru JCC. Uvedené softwarové produkty jsou volně přístupné na internetu.



Obrázek 5-1: Vývojový řetězec

5.2 Mapa paměti procesoru

Protože jde o plně 32bitový mikrořadič, máme k dispozici 4GB adresního prostoru. Rozlišuje se několik typů adresních prostorů, volbu provádíme zápisem kódu adresního prostoru do registrů procesoru SFC (Source Function Code), DFC (Destination Function Code). Registr SFC nastavuje zdrojový adresní prostor instrukce, DFC potom cílový adresní prostor instrukce. Při přístupech mikroprocesor signalizuje typ adresního prostoru svými výstupy FC3 – FC0 (Function Code). Typy a příslušné kódování adresních prostorů ukazuje Tabulka 5-1.

Kódování – Function Code Bits				Adresní prostor
3	2	1	0	
0	0	0	0	Reserved (Motorola)
0	0	0	1	User Data Space
0	0	1	0	User Program Space
0	0	1	1	Reserved (User)
0	1	0	0	Reserved (Motorola)
0	1	0	1	Supervisor Data Space
0	1	1	0	Supervisor Program Space
0	1	1	1	Supervisor CPU Space
1	x	x	x	DMA Space

Tabulka 5-1: Kódy adresních prostorů

Na počátku návrhu je nutné sestavit mapu paměti, podle které se nastaví registry definující práci chip select pinů. Mapa říká, kterými adresami jsou obsluhovány paměti a periferní obvody. Komunikační procesor obsahuje vnitřní paměť typu ROM, kde je uložen mikrokód pro podporované protokoly. Mezi podporované protokoly bez použití mikrokódu patří: HDLC, HDLC–Bus, Apple Talk, UART, Synchronní UART, BISYNC a model 68EN360 (použitý) také Ethernet. Po natažení mikrokódu je zde navíc podpora pro: Profibus, Asynchronní HDLC, vícekanálové GCI, atd.
Dále je na čipu umístěna paměť typu RAM, ta se používá k nastavení parametrů komunikačního procesoru (dále jen CP), nastavení parametrů modulu integrace systému (dále jen SIM), nastavení požadovaného módu a parametrů DMA řadiče, k uložení přídavného mikrokódu (aktuálně Profibus), k uložení popisovačů zásobníků pro CP (Buffer Descriptors, dále jen BD). Je zde také možné ukládat uživatelská data, ale to bych vzhledem k omezené kapacitě cca 3 kB nedoporučoval.
Paměť RAM tvoří souvislý blok o velikosti 8 kB. Její polohu v adresním prostoru určuje registr MBAR (Module Base Address Register), který leží na pevně definované adrese 3FF00h v CPU prostoru mikroprocesoru a vyžaduje přístup v režimu supervisor. Některé registry z interní RAM vyžadují přístupy v režimu supervisor, leží tedy v prostoru Supervisor Data Space. Ostatní jsou umístěné v adresním prostoru typu User Data Space.

Použitým periferním obvodům a pamětem jsem přidělil adresy viz tabulka 5-2. Mapa je navržena tak, aby ji nebylo nutné měnit při rozšiřování systému dalšími obvody (např. přídavná paměť, periferní zařízení).

Adresní prostor	Mapovaná jednotka	Kapacita	Šířka sběrnice [bit]
00000000 – 000FFFFF	Boot Flash	1 MB	16
00100000 – 001FFFFF	Volný prostor	1 MB	Není určena
00200000 – 003FFFFF	Flash Memory	2 MB	32
00400000 – 005FFFFF	SRAM	2 MB	32
00600000 – 007FFFFF	Volný prostor	2 MB	Není určena
00800000 – 00800FFF	USB chip	4 kB	8
00800100 – FFFFDFFF	Volný prostor	4088 MB	Není určena
FFFFE000 – FFFFFFFF	Vnitřní RAM	8 kB	32

Tabulka 5-2: Mapa paměti ProfiUsb

Vnitřní RAM procesoru se dělí na dvě části:
– Dual Port RAM
– Register RAM

Dual Port RAM tvoří první 4kB vnitřní RAM a leží na adrese, kterou obsahuje MBAR. V tomto případě je to adresa FFFFE000h. Její strukturu určuje tabulka 5-3.

Rozsah adres (hexa)	Velikost	Blok	Význam
DPRBASE + 000 DPRBASE + 3FF	1024 Bytů	Dual Port RAM	Uživatelská data, BD, Mikrokód – program
DPRBASE + 400 DPRBASE + 5FF	512 Bytů	Dual Port RAM	Uživatelská data, BD
DPRBASE + 600 DPRBASE + 6FF	256 Bytů	Dual Port RAM	Uživatelská data, BD, Mikrokód – pracovní paměť
DPRBASE + 700 DPRBASE + BFF	256 Bytů	Dual Port RAM	Uživatelská data, BD
DPRBASE + C00 DPRBASE + CBF	192 Bytů	Dual Port RAM	Parametrizační paměť Stránka 1
DPRBASE + CC0 DPRBASE + CFF		Rezervováno	Rezervováno
DPRBASE + D00 DPRBASE + DBF	192 Bytů	Dual Port RAM	Parametrizační paměť Stránka 2
DPRBASE + DC0 DPRBASE + DFF		Rezervováno	Rezervováno
DPRBASE + E00 DPRBASE + EBF	192 Bytů	Dual Port RAM	Parametrizační paměť Stránka 3
DPRBASE + EC0 DPRBASE + EFF		Rezervováno	Rezervováno
DPRBASE + F00 DPRBASE + FBF	192 Bytů	Dual Port RAM	Parametrizační paměť Stránka 4
DPRBASE + FC0 DPRBASE + FFF		Rezervováno	Rezervováno

Tabulka 5-3: Mapa Dual Port RAM

Stránka parametrizační paměti	Sub – modul	Bázová adresa (hexa)
1	SCC1 Base	DPRBASE + C00
1	Misc Base	DPRBASE + CB0
2	SCC2 Base	DPRBASE + D00
2	SPI Base	DPRBASE + D80
2	Timer Base	DPRBASE + DB0
3	SCC3 Base	DPRBASE + E00
3	IDMA1 Base	DPRBASE + E70
3	SMC1 Base	DPRBASE + E80
4	SCC4 Base	DPRBASE + F00
4	IDMA2 Base	DPRBASE + F70
4	SMC2 Base	DPRBASE + F80

Tabulka 5-4: Bázové adresy modulů

5.3 Konfigurace prostředí

5.3.1 GDB

Gnu DeBugger (GDB) používá ke své konfiguraci skript, v tomto konkrétním případě má název „profi360.gdb“ a nachází se v adresáři projektu. Ve stejném adresáři je umístěný symbolický odkaz, který ukazuje na soubor skriptu a je pojmenován „.gdbinit68“. Skript se vytváří pomocí modifikovaného jazyka C. Do skriptu je nutné zadat název souboru, který se zkopíruje do externí RAM řadiče a odtud bude také spuštěn. V našem případě se jedná o soubor „m.out“. Potom je třeba provést základní systémové nastavení řadiče včetně nastavení registrů potřebných pro práci chip-select pinů. Důležitý fragment s požadovaným nastavením vypadá následovně:

```
echo Setting bdm\n

file m.out

target bdm /dev/bdm

#==========================================================

define set_CS_BR
  #$arg0 = BA31-BA11
  #$arg1 = Offset From REGB
  set *(unsigned int*)($ptr_REGB+$arg1) = ((unsigned int)$arg0)+1
  #p /ox *(unsigned int*)($ptr_REGB+$arg1)
end

define set_CS_OR
  #$arg0 = AM27-AM11
  #$arg1 = Number Of Wait States
  #$arg2 = Sram Port Size
  #$arg3 = Offset From REGB
  set *(unsigned int*)($ptr_REGB+$arg3) = $arg0+(($arg1+1) << 28)+($arg2 << 1)
  #p /ox *(unsigned int*)($ptr_REGB+$arg3)
end

# sets chipselects and system configuration
define bdm_hw_init
echo bdm_hw_init ...\n

set remotecache off
bdm_timetocomeup 0
bdm_autoreset off
bdm_setdelay 70
bdm_reset
bdm_setdelay 0
set $sfc=5
set $dfc=5

# system configuration

# MBAR Module Base Address Register
# 31      30          13     12  11  10  9     8          1      0
# BA31  BA30  ....  BA13  0   0    0  AS8   AS7  ...  AS0   V
# BA = BaseAddress
# AS = Address Space = Maskovani adresniho prostoru
# V = data valid
# set *(unsigned int *)0x0003ff00=0
set $sfc=7
set $dfc=7
set $ptr_MBAR = (unsigned int *)0x0003ff00
```

```
set *$ptr_MBAR = 0x0ffffe001
set $ptr_DPRBASE = (unsigned char *)0x0ffffe000
set $ptr_REGB = $ptr_DPRBASE + 0x1000
set $sfc=5
set $dfc=5

#nastavi registry brany PA a rozsviti LED diodu
set $ptr_PAPAR = (unsigned short *)($ptr_REGB + 0x552)
set *$ptr_PAPAR = 0
set $ptr_PADIR = (unsigned short *)($ptr_REGB + 0x550)
set *$ptr_PADIR = 0xf000
set $ptr_PAODR = (unsigned short *)($ptr_REGB + 0x554)
set *$ptr_PAODR = 0xffff
set $ptr_PADAT = (unsigned short *)($ptr_REGB + 0x556)
set *$ptr_PADAT = 0xefff

# MCR Module Configuration Register
set *(unsigned int *)($ptr_REGB + 0x000) = 0x00006c7f

#PEPAR config
set *(unsigned short*)($ptr_REGB + 0x16) = 0x0080

#GMR
set *(unsigned int *)($ptr_REGB + 0x40) = 0x00001100

#SYPCR
set *($ptr_REGB + 0x22) = 0x03

#settings for chip selects

#set_CS_BR BaseAddress OffsetFromREGB
#set_CS_OR AddressMask NumberOfWaitStates(0-14) SramPortSize OffsetFromREGB

#CS0 - BootFlash 1 MB - 16Bit
set_CS_BR 0x0000000 0x50
set_CS_OR 0xff00000 0 1 0x54

#CS1 - Flash 2MB - 32Bit
set_CS_BR 0x0200000 0x60
set_CS_OR 0xfe00000 0 0 0x64

#CS7 - SRAM 2 MB - 32Bit
set_CS_BR 0x0400000 0xc0
set_CS_OR 0xfe00000 0 0 0xc4

#CS6 - USB Chip - 8Bit
set_CS_BR 0x0800000 0xb0
set_CS_OR 0xf800000 0 2 0xb4
```

Makro funkce set_CS_BR a set_CS_OR se používají k definici registrů chip-select pinů. Vstupní parametry funkce set_CS_BR jsou: bázová adresa chip-select pinu a offset registru BRx od začátku interní RAM řadiče.
Vstupní parametry funkce set_CS_OR jsou: adresní maska – velikost okna pro komparaci adres, počet čekacích cyklů – wait states, šířka sběrnice a offset registru ORx od začátku interní RAM řadiče. Parametr šířka sběrnice používá následující kódování:

Hodnota parametru	Šířka sběrnice [bit]
0	32
1	16
2	8

Tabulka 5-5: Kódování šířky sběrnice pro makro set_CS_OR

Pomocné proměnné a registry procesoru musí začínat vždy znakem $. Je možné používat i proměnné definované ve zdrojových souborech. Dokumentaci GDB obsahuje přiložené CD-ROM nebo [19].

Výpis nejčastěji používaných příkazů:

`set *(unsigned short *) 0xfffff556 = 0xf000`
– zapíše hodnotu f000h na adresu fffff556h

`set quicc->pio_padat = 0xffff`
– zapíše hodnotu ffffh do struktury quicc na položku pio_padat

`p *(unsigned short *) 0xfffff556`
– přečte a zobrazí v dekadickém formátu hodnotu z adresy fffff556h

`p /x quicc->pio_padat`
– přečte a zobrazí v hexadecimálním formátu hodnotu položky pio_padat ze struktury quicc

`x /10bx 0x00400000`
– přečte a zobrazí 10 bytů od adresy 400000h výš

`run` – spustí program

`^C` – zastaví vykonávání programu

`step` – vykoná se jedna instrukce

`until` – vykonává instrukce, dokud nedosáhne vyšší pozice ve zdrojovém kódu

`cont` – pokračuje ve vykonávání programu, dokud nenarazí na breakpoint

`b main` – umístí breakpoint do funkce main

5.3.2 Překladač jazyka C

V této aplikaci jsem používal překladač GCC, který je velmi univerzální. K jeho konfiguraci se používají ld-skripty, jedná se o soubory profi360.ld-boot, profi360.ld-flash, profi360.ld-ram a profi360.ld-cfg, které jsou umístěné v podadresáři ./boot. Tyto soubory definují adresy, od kterých je možné umísťovat program, uživatelská data, stack, atd.
Psaní zdrojových kódů jsem prováděl v grafickém prostředí JCC, které automaticky generuje soubor definující překlad a kompilační vazby – Makefile a dokáže provádět překlad projektu. Zde je nutné nastavit stejný název výstupního souboru, jako je uvedený v konfiguračním skriptu ke GDB (viz výše).

5.4 Konfigurace procesoru

Mikroprocesor Motorola 68360, nazývaný výrobcem též QUICC (QUad Integrated Communication Controller), nabízí mnohé funkce, kterým je třeba po resetu nastavit požadované parametry.
Existuje několik operačních systémů, které mohou usnadnit vývojářům práci při programování aplikací, např. ucLinux, RTEMS. Následující nastavení jsem používal pro samostatnou aplikaci, běžící na procesoru bez použití některého operačního systému.

5.4.1 Registr MBAR (Module Base Address Register)

Slouží k namapování vnitřní paměti do adresního prostoru mikrořadiče. Leží na pevně definovené adrese 3FF00h. Nastavení provedeme následujícím způsobem:

1. Přepnutí do režimu Supervisor CPU Space – nastavení registrů procesoru SFC a DFC na hodnotu 7.
2. Podle mapy paměti a definice bitů v registru (viz kapitola 6.9.1 Module Base Address Register v [1]) je nutné na adresu 3FF00h zapsat hodnotu FFFFE001h.
3. Uvedení zpět do režimu Supervisor Data Space – nastavení registrů procesoru SFC a DFC na hodnotu 5.

5.4.2 Registr MCR (Module Configuration Register)

Module Configuration Register určuje základní vlastnosti systému, jako je priorita SIM modulu při obsluze přerušení, maska přerušení pro signál $\overline{\text{BCLRO}}$, přístup do registrů SIM v režimu Supervisor/User, zapnutí/vypnutí monitoru sběrnice, zapnutí/vypnutí PIT (Periodic Interrupt Timer), SWT (Software Watchdog Timer), Synchronní/ asynchronní časování sběrnice. Přesnou definici bitů najdeme v kapitole 6.9.3.1 MODULE CONFIGURATION REGISTER v [1].
Požadovanou konfiguraci představuje hodnota 6C7Fh.

5.4.3 Registr PEPAR (Port E Pin Assignment Register)

Quicc má sice definovaných 240 vývodů, ale ty ještě nestačí k pokrytí všech jeho schopností. Mnohé vývody mají více než jednu funkci. Tyto funkce jsou mezi sebou multiplexovány a jejich aktuálně zvolený režim udávají příslušné registry. PEPAR určuje přiřazení pinů k bráně E. Ze zvoleného HW modelu vyplývá nastavení vývodů $\overline{\text{IPIPE1}}$, $\overline{\text{WE0}-\text{WE3}}$, $\overline{\text{OE}}$. Podle definice registru v kapitole 6.9.4 Port E Pin Assignment Register v [1] je konfigurační hodnota registru 0080h.

5.4.4 Registr SYPCR (SYstem Protection Control Register)

Řídí monitorování systému, časování SWT (Software Watchdog Timer) a monitor sběrnice. Nastavením tohoto registru jsem provedl:

- Vypnutí SWT
- Vypnutí monitoru dvojité chyby na sběrnici
- Zapnutí monitoru pro vnější cykly sběrnice
- Nastavení časování monitoru sběrnice na 128 tiků systémových hodin

V souladu s kapitolou 6.9.3.5 SYSTEM PROTECTION CONTROL REGISTER v [1] je konfigurační hodnota 03h.

5.4.5 Registr GMR (Global Memory Register)

Tento registr nastavuje parametry řadiče pamětí. Je společný pro všechny paměti. Určuje: vlastnosti občerstvování DRAM, vlastnosti skupiny DRAM, vlastnosti skupiny SRAM a některé společné vlastnosti. Po resetu ho přepíšeme hodnotou 00001100h (viz kapitola 6.13.1 Global Memory Register v [1]). Tato konfigurace potlačuje aktivaci výstupů $\overline{CS}$ při přístupu do vnitřní paměti procesoru.

Potom je nutné nastavit parametry chip-select pinů. To se provede modifikací registrů BR (Base Register) a OR (Option Register). Pro každý chip-select pin koresponduje dvojice BR a OR.

5.4.6 Registr BR (Base Register)

Je určený pro nastavení ovládání SRAM/DRAM pamětí.

Bit	31 - 11	10 - 7	6	5	4	3	2	1	0
Význam	BA31-BA11	FC3-FC0	TRLXQ	BACK40	CSNT40	CSNTQ	PAREN	WP	V
Reset	0	0	1	0	1	0	0	0	0

Tabulka 5-6: Registr BR

Tabulka 5-5 udává obsah BR a přidělení bitů. Některé bity popíši podrobně:

BA31-BA11 – Base Address, bity 31-11. Nastavují bázovou adresu chip-selectu. Zde je naprogramovaná adresa, od které se má příslušná skupina pamětí/periferií vybírat.
FC3-FC0 – Omezení přístupu na tyto adresy vzhledem k typu přístupu. Nastaveno na 0 – povoleny všechny přístupy.
V – Valid Bit, nastavením na 1 je řečeno, že dvojice BR a OR obsahuje platná data.

Ostatní bity jsou nastaveny na 0.

5.4.7 Registr OR (Option Register)

Je určený pro nastavení ovládání SRAM/DRAM pamětí.

Bit	31 - 28	27 - 11	10 - 7	6 - 5	4	3	2	1	0
Význam	TCYC3-0	AM27-11	FCM3-0	BCYC1-0	-	PGME	SPS1	SPS0	DSSEL
Reset	1	0	0	0	0	0	0/1	0/1	0

Tabulka 5-7: Registr OR

Tabulka 5-6 udává obsah OR a přidělení bitů. Některé bity popíši podrobně:

DSSEL – Udává, zda jde o DRAM nebo SRAM. Zapsaná 0 pro SRAM a periferní obvody.
SPS1-SPS0 – Šířka portu: 00 – 32 bitů, 01 – 16 bitů, 10 – 8 bitů.
PGME – Page Mode, zapsána 0 pro vypnutí stránkovacího režimu.
AM27-AM11 – Address Mask 27-11, udávají velikost okna pro výběr paměti. 1 znamená, že bit bude použit pro komparaci adresy, 0 naopak.

TCYC3-0 – Definují délku cyklu čtení/zápisu z/do paměti nebo periferie. Pro paměti SRAM má následující význam:
0 – Fast Termination (2 hodinové pulsy),
1 – 0 prodlužovacích cyklů (3 hodinové pulsy),
...
15 – 14 prodlužovacích cyklů (17 hodinových pulsů).
Použitá 1 – bez prodlužovacích cyklů.

Ostatní bity jsou nastaveny na 0.

5.4.8 Registr VBR (Vector Base Register)

Je to jeden z registrů, které se nacházejí přímo v registrech procesoru. Udává polohu tabulky přerušovacích vektorů. Podle mapy paměti (viz Tabulka 5-2) umístíme tabulku na začátek externí paměti RAM, tedy na adresu 400000h. Tabulka má velikost 256 32bitových slov.
Mapu tabulky přerušovacích vektorů udává tabulka 5-16 v [1].

5.4.9 Ostatní registry

Je zde mnoho dalších registrů, které je třeba modifikovat pro konkrétní případ použití ProfiUSB. Zde bych uvedl jen několik dalších, které je nutné nastavit pro toto hardwarové nastavení:
UART na SMC2, Profibus1 na SCC1, Profibus2 na SCC2, výstupy na D4 – D7.
Potřebujeme nastavit vývody mikroprocesoru na požadovanou funkci. To se provede postupně pro všechny I/O brány (PA – PC). Při zápisu je nutné dodržet pořadí registrů tak, jak je uvedeno níže (viz kapitola 7.14 PARALLEL I/O PORTS v [1])!

PAPAR (Port A Pin Assignment Register) = 0F0Fh,
PADIR (Port A Data Direction Register) = F000h,
PAODR (Port A Open Drain Register) = 0000h,

PBPAR = 00E00h, PBDIR = 00001h, PBODR = 00000h,

PCPAR = 000Fh, PCDIR = 0000h, PCSO = 0500h.

5.5 Příkazy pro komunikační procesor

CP vykonává program, který leží ve vnitřní paměti ROM nebo RAM (při použití jiného mikrokódu než standardně QUICC obsahuje). Aby bylo možné běh programu ovlivnit „zvenku“ (z CPU32+ nebo externího mastera), je CP vybaven registrem příkazů (CR – Command Register). Prostřednictvím tohoto registru lze vysílat příkazy komunikačnímu procesoru. Registr má následující strukturu:

Bit	15	14 - 12	11 - 8	7 - 4	3 - 1	0
Význam	RST	-	OPCODE	CHNUM	-	FLG

Tabulka 5-8: Registr CR

RST – Kompletní reset CP.
OPCODE – Operační kód instrukce (viz níže).
CHNUM – Číslo kanálu CP (viz níže).

FLG – Flag bit. Je to semafor, který nastavuje uživatel a nuluje CP po dokončení příkazu. Na nulování tohoto bitu se vždy čeká při vysílání další instrukce.

Instrukční sadu ukazuje tabulka 5-9. Některé instrukce nemusí daný protokol podporovat, proto uvedu podporované instrukce v popisu konkrétního protokolu. Instrukce se liší podle použitého protokolu.

Kód	Kanál / Název instrukce			
	SCC	SMC(UART)	IDMA	Timer
0000	Init Rx & Tx Params	Init Rx & Tx Params		
0001	Init Rx Params	Init Rx Params		
0010	Init Tx Params	Init Tx Params		
0011	Enter Hunt Mode	Enter Hunt Mode		
0100	Stop Tx	Stop Tx		
0101	GR Stop Tx		Init IDMA	
0110	Restart Tx	Restart Tx		
0111	Close Rx BD	Close Rx BD		
1000	Set Group Addr			Set Timer
1001				
1010	Reset BCS			

Tabulka 5-9: Příkazy pro komunikační procesor

Čísla kanálů se kódují podle tabulky 5-10.

Kanál	SCC1	SCC2	SPI, RISC Timers	SCC3	SMC1, IDMA1	SCC4	SMC2, IDMA2
Číslo	0000	0100	0101	1000	1001	1100	1101

Tabulka 5-10: Kódování kanálů komunikačního procesoru

5.6 Buffer deskriptory

Aby CP „věděl“, kam ukládat a odkud vysílat sériová data, používá pro popis přijímacích/vysílacích bufferů tzv. buffer deskriptory (BD). Jsou to datové struktury uložené v příslušné oblasti paměti RAM. Jedná se o čtveřici 16bitových slov uloženou v paměti lineárně za sebou.

Adresa v paměti	Význam
Offset + 0	Stavové a řídící slovo
Offset + 2	Objem dat [Byte]
Offset + 4	Vyšší slovo ukazatele na datový buffer
Offset + 6	Nižší slovo ukazatele na datový buffer

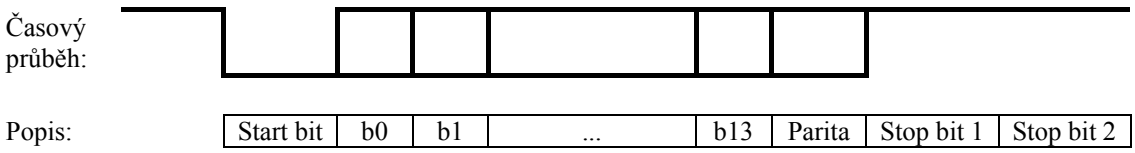
Tabulka 5-11: Struktura BD

Tabulka 5-11 ukazuje obecný význam položek BD. Detailnější popis bude proveden pro zvolené protokoly, protože položky se pro různé protokoly liší.

5.7 UART

V této části popíši nastavení a práci komunikačního procesoru v režimu UART. Tento protokol je podporován na obou typech kanálů (SMC, SCC). Popis se bude zabývat provozem na kanálu SMC2, protože ten může být (a také je) zvolen v hardwarové konfiguraci. Při použití SCC4, který je možné také vybrat je situace podobná.
UART (Universal Asynchronous Receiver/Transmitter) je, jak již název napovídá, protokol, který se vyznačuje asynchronním typem přenosu dat.
Fyzickou vrstvu zde představuje sběrnice RS-232. Data mezi ProfiUSB a externím zařízením (PC) proudí po třívodičovém vedení. Vodiče RxD – Receive Data (přijímaná data), TxD – Transmit Data (vysílaná data) a GND – Ground (společná zem).
Když neprobíhá přenos, na datových vodičích lze naměřit cca 12V (≈ logická 1). Přenos dat začíná v okamžiku poklesu úrovně na datovém vodiči do logické 0 (≈ cca –12V). Tento pokles trvá po dobu jednoho bitu a nazýváme ho start bit. Potom následuje přenos 5 – 14 datových bitů a volitelně paritního bitu (sudá, lichá nebo bez parity). Poslední část přenosu tvoří jeden nebo dva stop bity.

Struktura přenosu znaku:



Obrázek 5-2: UART – Přenos znaku

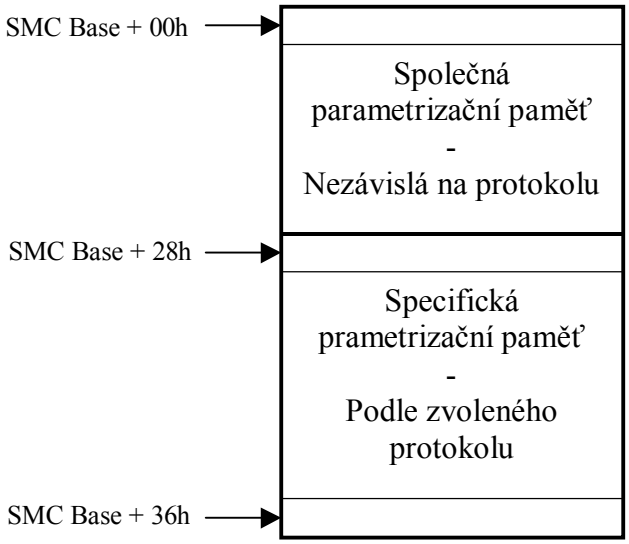
Zvolený model přenosu znaku je: 1 start bit, 8 datových bitů bez parity, 1 stop bit.

5.7.1 Parametrizační paměť

Informace o parametrech, stavu přenosu, chování při zápisu/čtení do/z přijímacích/vysílacích bufferů a umístění bufferů získává CP z parametrizační paměti. Ta je umístěna na pevně definovaném místě ve vnitřní RAM. Parametrizační paměť se dělí na dvě části:

1. Společná pro všechny protokoly, obsahuje obecné informace, CP ji využívá k uložení stavových informací o přenosu.
2. Část, která závisí na zvoleném protokolu, pro různé protokoly obsahuje různá data. CP zde ukládá dočasné informace, specifické pro daný protokol.

Na obr. 5-3 je vidět rozdělení parametrizační paměti a její umístění v adresním prostoru.



Obrázek 5-3: UART – Parametrizační paměť

Nyní rozeberu sestavení obou částí podrobněji. Část nezávislá na protokolu (její první slovo) začíná na bázové adrese příslušného SMC kanálu (SMC 2). Následující tabulka ukazuje skladbu první části.

Adresa	Označení	Šířka	Význam
SMC Base + 00h	**RBASE**	Word	Bázová adresa přijímacích buffer deskriptorů
SMC Base + 02h	**TBASE**	Word	Bázová adresa vysílacích buffer deskriptorů
SMC Base + 04h	**RFCR**	Byte	Přijímací funkční kód
SMC Base + 05h	**TFCR**	Byte	Vysílací funkční kód
SMC Base + 06h	**MRBLR**	Word	Maximální délka přijímaného datagramu
SMC Base + 08h	RSTATE	Long	Vnitřní stav přijímání
SMC Base + 0Ch		Long	Vnitřní ukazatel přijímaných dat
SMC Base + 10h	RBPTR	Word	Vnitřní ukazatel na přijímací buffer deskriptor
SMC Base + 12h		Word	Vnitřní počet přijatých bytů
SMC Base + 14h		Long	Dočasný registr pro příjem
SMC Base + 18h	TSTATE	Long	Vnitřní stav vysílání
SMC Base + 1Ch		Long	Vnitřní ukazatel vysílaných dat
SMC Base + 20h	TBPTR	Word	Vnitřní ukazatel na vysílací buffer deskriptor
SMC Base + 22h		Word	Vnitřní počet vyslaných bytů
SMC Base + 24h		Long	Dočasný registr pro vysílání

Tabulka 5-12: UART – Společná parametrizační paměť

Tučně zvýrazněné položky musíme nastavit před povolením kanálu pro příjem/vysílání.

RBASE, TBASE – Udávají počáteční adresu řetězce buffer deskriptorů ve vnitřní RAM. Hodnota je offset od počátku interní RAM.

RFCR, TFCR – Funkční kódy (viz Tabulka 5-1), kterými CP signalizuje přístupy do paměti.

MRBLR – Udává maximální možný objem dat, který lze zapsat do jednoho přijímacího bufferu. Je menší nebo rovna velikosti přijímacích bufferů.

Detailní popis přináší kapitola 7.11.4 SMC Parameter RAM v [1].

Specifickou část paměti pro parametrizaci UART protokolu udává tabulka 5-13.

Adresa	Označení	Šířka	Význam
SMC Base + 28h	MAX_IDL	Word	Maximální počet prázdných znaků
SMC Base + 2Ah	IDLC	Word	Dočasný čítač prázdných znaků
SMC Base + 2Ch	BRKLN	Word	Délka posledních přijatých dat (přerušeno)
SMC Base + 2Eh	BRKEC	Word	Čítač přerušení přijímání
SMC Base + 30h	BRKCR	Word	Počet přerušení vysílání
SMC Base + 32h	R_mask	Word	Dočasný maskovací bit

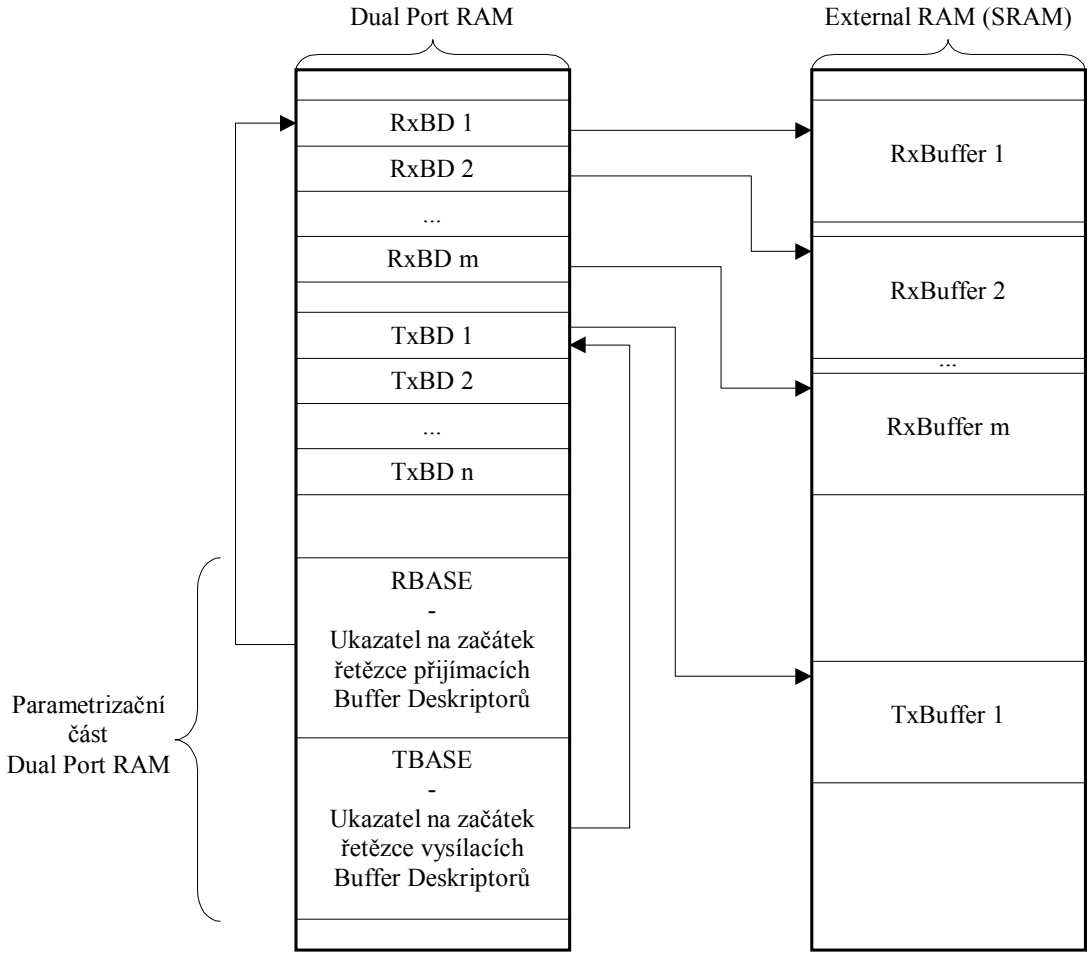
Tabulka 5-13: UART – Specifická parametrizační paměť

MAX_IDL – Přednastavený počet prázdných znaků. Prázdný znak = přijaté samé jedničky. Když CP během přijímání dostane tolik prázdných znaků jako udává MAX_IDL, uzavře aktuální přijímací buffer. Výhoda je v tom, že může být okamžitě generováno přerušení pro CPU32+ a nemusí se čekat na doplnění bufferu. Musí být nastaven před povolením kanálu pro příjem/vysílání.

Podrobný popis najdeme v kapitole 7.11.7.3 SMC UART MEMORY MAP v [1].

5.7.2 Buffer Deskriptory

V režimu UART se používají specifické buffer deskriptory pro příjem/vysílání dat. Pro přiblížení účelu a paměťovou vazbu uvádím obr. 5-4.



Obrázek 5-4: Umístění, funkce BD, buffery

Buffer Deskriptory (BD) mají vyhrazenou oblast paměti současně s mikrokódem pro CP. Jestliže CP nepoužívá žádný vnější mikrokód, tato oblast může být využita pro BD. Pokud však použijeme mikrokód, nesmíme do příslušných paměťových sektorů zasahovat a umísťovat BD.

5.7.2.1 Přijímací BD

Buffer deskriptory pro příjem dat mají následující tvar:

Adresa	Bit / Význam													
	15	14	13	12	11-10	9	8	7-6	5	4	3	2	1	0
Offset + 0	**E**	-	**W**	**I**	-	**CM**	ID	-	BR	FR	PR	-	OV	CD
Offset + 2	Objem dat													
Offset + 4	**Ukazatel na přijímací datový buffer**													
Offset + 6														

Tabulka 5-14: UART – Přijímací BD

Na adrese offset + 0 najdeme stavové slovo BD. Tímto slovem signalizuje CP, zda přijímací buffer je prázdný, jsou-li do něho zrovna dodávána data nebo je již naplněn. Po naplnění přijímacího bufferu CP tento buffer uzavře, může vyvolat přerušení (volitelně) a pokračuje s přijímáním do bufferu, na který ukazuje další BD v řetězci přijímacích BD. Přechod na další buffer může způsobit jedna z následujících událostí:

1. Detekce chyby během zpracovávání přijímaných dat.
2. Naplnění aktuálního přijímacího bufferu.
3. Přijetí zadaného počtu prázdných znaků (parametrizační paměť).

E – Prázdný buffer
- 0 – Datový buffer je naplněn přijatými daty nebo došlo k chybě přijímání. CP nebude tento buffer používat k dalšímu přijímání.
- 1 – Datový buffer je prázdný nebo se do něj právě přesunují data. BD a příslušný datový buffer teď obsluhuje CP a uživatel by neměl modifikovat žádné části.

W – Poslední BD v řetězci
- 0 – Tento BD není poslední v řetězci přijímacích BD.
- 1 – BD je poslední v řetězci přijímacích BD. Po naplnění příslušného bufferu CP pokračuje s přijímáním do bufferu, na který ukazuje první BD v řetězci (leží na adrese, kterou udává RBASE v parametrizační paměti).

I – Příznak generování přerušení
- 0 – Po naplnění a uzavření BD není generováno přerušení.
- 1 – Po naplnění a uzavření BD je v SCCE (scc event register) nastaven RX bit. Ten může vyvolat přerušení, pokud je povoleno.

CM – Volba módu
- 0 – Normální režim.
- 1 – Po naplnění BD komunikační procesor nevynuluje E příznak a při dalším oběhu přes tento BD se data začnou přesouvat do bufferu. Pokud mezitím nebyly vybrány, dojde k jejich přepsání. E bit se vynuluje pouze při detekci chyby.

ID – Prázdné znaky
- 1 – Buffer byl uzavřen po přijetí naprogramovaného počtu prázdných znaků. (MAX_IDL v parametrizační paměti)

BR – Ukončovací znak
1 – Buffer byl uzavřen po přijetí ukončovacího znaku (samé nuly).

FR – Chyba ohraničení
1 – Buffer byl uzavřen po detekci chyby ohraničení. Poslední přijatý znak obsahoval chybné ohraničení (znak bez stop bitu).

PR – Chyba parity
1 – Znak s chybnou paritou je poslední přijatý znak a poté došlo k uzavření BD.

OV – Přetečení
1 – Během přijímání zprávy došlo k přetečení přijímače.

Objem dat – Sem zapíše CP délku přijatých dat (v bytech) po uzavření BD.

Ukazatel na přijímací datový buffer – Zde je nutné zapsat adresu prvního bytu (nejnižší adresa) v alokovaném datovém bufferu pro tento BD.

5.7.2.2 Vysílací BD

Buffer deskriptory pro vysílání dat mají následující tvar:

Adresa	Bit / Význam							
	15	14	13	12	11-10	9	8	7-0
Offset + 0	**R**	-	**W**	**I**	-	**CM**	**P**	-
Offset + 2	**Objem dat**							
Offset + 4	**Ukazatel na vysílací datový buffer**							
Offset + 6								

Tabulka 5-15: UART – Vysílací BD

Tučně vyznačené položky by měl nastavit uživatel.

R – Připraven k odeslání
0 – Datový buffer náležící k BD není připraven k odeslání. CP čeká na nastavení R bitu a poté buffer odešle do fronty.
1 – Buffer obsahuje platná data a je připraven k odeslání nebo odesílání právě probíhá. Po přechodu CP k tomuto BD je buffer odeslán. Po odeslání bufferu CP tento bit vynuluje.

W – Poslední BD v řetězci
0 – Tento BD není poslední v řetězci vysílacích BD.
1 – BD je poslední v řetězci vysílacích BD. Po odeslání bufferu CP pokračuje ve vysílání z bufferu, na který ukazuje první BD v řetězci (leží na adrese, kterou udává TBASE v parametrizační paměti).

I – Příznak generování přerušení
0 – Po odesláni bufferu není generováno přerušení.
1 – Po odesláni bufferu je v SCCE (scc event register) nastaven TX bit. Ten může vyvolat přerušení, pokud je povoleno.

CM – Volba módu
- 0 – Normální režim.
- 1 – Po odesláni bufferu komunikační procesor nevynuluje R příznak a při dalším oběhu přes tento BD se data z bufferu začnou odesílat. Pokud se datový buffer mezitím nezměnil, dojde k opětovnému vyslání stejných dat.

P – Předvysílání
- 0 – Žádné prázdné znaky nejsou vysílány.
- 1 – Před odesláním dat je vyslán prázdný znak (samé jedničky). Když je objem dat roven nule, je vyslán pouze prázdný znak.

Objem dat – Udává množství odesílaných dat (v bytech) z daného bufferu.

Ukazatel na odesílaný datový buffer – Zde je nutné zapsat adresu prvního bytu (nejnižší adresa) v alokovaném datovém bufferu pro tento BD.

5.7.3 Příkazy CP

Běh komunikačního procesoru se ovládá pomocí příkazů vysílaných přes CR (Command Register). Přikazy, které podporuje CP v režimu UART na SMC jsou následující:

Stop Transmit – Přeruší vysílání znaků na příslušném kanálu CP. Když CP obdrží tuto instrukci během vysílání zprávy, dovysílá všechny znaky z vysílací FIFO a potom přestane vysílat data. Po dokončení instrukce se TBPTR nemodifikuje. Vysílač potom vyšle programovatelný počet ukončovacích znaků, poté jsou na lince prázdné znaky.

Restart Transmit – Způsobuje obnovení vysílání ve stavu, v jakém bylo přerušeno (buď bylo zakázáno v SMC Mode Registru nebo prostřednictvím Stop Transmit příkazu). Vysílání započne z aktuálního BD (ukazuje na něj TBPTR).

Init TX Parameters – Nastaví všechny parametry pro vysílání v parametrizační paměti na jejich počáteční hodnotu (zkopíruje TBASE do TBPTR, TSTATE na nulu). Příkaz by měl být vysílán pouze když je vysílač deaktivován (TEN v SMCMR musí být 0) .

Enter Hunt Mode – Tento příkaz by se pro SMC UART kanál neměl používat, doporučuje se použít Close RX BD.

Close RX BD – Přinutí CP uzavřít aktuální BD pro příjem dat a pokračovat v přijímání do dalšího bufferu dle pořadí tabulky. Když je CP ve stavu, kdy nepřijímá data, tato instrukce nemá žádný efekt.

Init RX Parameters – Vrátí všechny parametry pro příjem v parametrizační paměti na jejich počáteční hodnotu (zkopíruje RBASE do RBPTR a nastaví RSTATE na nulu). Příkaz by měl být vysílán, pouze když je vysílač deaktivován (REN v SMCMR musí být 0).

Init TX and RX Parameters – Provádí současně Init TX Parameters a Init RX Parameters. Instrukce by se měla vyslat pouze, když vysílač i přijímač jsou neaktivní (REN a současně TEN v SMCMR jsou nulové).

5.7.4 Inicializace

1. Zde uvádím postup, který je třeba vykonat, aby se zprovoznila komunikace na SMC2 UART kanálu.
2. Nastavit parametry SDMA (sériový DMA kanál). Do registru SDCR zapíšeme 0740h. SDMA kanálem přenáší CP data ze sériových kanálů (SCC, SMC, SPI) do paměti a naopak. Naprogramováním SDCR určíme úroveň priority přístupu na sběrnici.
3. Nakonfigurovat příslušné piny brány B na funkci SMRxD a SMTxD (PBPAR 10-11 nestavit na jedničku, PBDIR 10-11 nastavit na nulu, PBODR 10-11 nastavit na nulu).
4. Nastavit parametry generátoru přenosové rychlosti (BRG). Nejdříve je nutné zvolit přiřazení generátorů ke kanálům CP. Volím toto nastavení: BRG1 → SCC1; BRG2 → SCC2; BRG3 → SMC1, SCC3, SCC4; BRG4 → SMC2.
5. Nastavíme dělitel v registru BRGC4 tak, aby výsledná frekvence odpovídala šestnáctinásobku požadované přenosové rychlosti.
6. Provedeme spojení výstupu BRG4 → SMC2 a BRG3 → SMC1, přivedeme tak vygenerované hodiny na sériový kanál. Zapíšeme 30002000h do SIMODE registru.
7. Sestavíme řetězec přijímacích a vysílacích BD v paměti. Přijímacím BD přidělíme buffery v externí paměti SRAM a nastavíme ve stavovém slovu bity E, I, poslednímu BD v řetězci ještě bit W. Vysílacím BD vynulujeme všechny příznaky, poslednímu v řetězci nastavíme W.
8. Zapíšeme bázové adresy (první byty BD) obou řetězců do společné parametrizační paměti (položky RBASE, TBASE). Nastavíme RFCR, TFCR (18h pro normální operace). Stanovíme MRBLR z velikostí přijímacích bufferů (MRBLR ≤ velikost nejmenšího z přijímacích bufferů).
9. Nastavíme parametry UART parametrizační paměti: MAX_IDL = 3, BRKCR = 3, BRKEC = 0.
10. Povolíme generování přerušení od uzavření přijímacích BD (nastavení bitu 0 v SMCM registru na jedna).
11. Nakonfigurujeme SMCMR. Režim UART, nastavíme parametry znaku (počet datových bitů, parita, počet stop bitů). Přijímač i vysílač zatím ponecháme vypnutý.
12. Vyšleme příkaz CP na inicializaci parametrů – Init TX and RX Parameters.
13. Zapíšeme hodnotu FFh do SMCE registru, abychom vynulovali vyvolané předchozí události (mohli by způsobit přerušení).

14. Nastavíme parametry řadiče přerušení (registr CICR). Volba priorit mezi zdroji přerušení, priorita přerušení samotného řadiče, bázová adresa vektorů přerušení od CP (VBA3-0 volím 010b, aby první uživatelský vektor přerušení, tabulka 5-16 v [1], tedy vektor č. 64, odpovídal prvnímu vektoru CP). Výsledná hodnota je 00e4bf40h.

15. Na adresu vektoru přerušení odpovídající SMC2 umístíme adresu obslužné rutiny, která má za úkol vybírat přijatá data z bufferů. Situaci lze řešit elegantněji, a sice pouhou záměnou ukazatele na přijímací buffer v uzavřeném BD. Funkce musí zabezpečit opětovné povolení přijímání do tohoto BD nastavením E bitu a nezměnit ostatní příznaky.

16. Aktivujeme činnost přijímače/vysílače nastavením bitů 0-1 v SMCMR.

5.8 Profibus

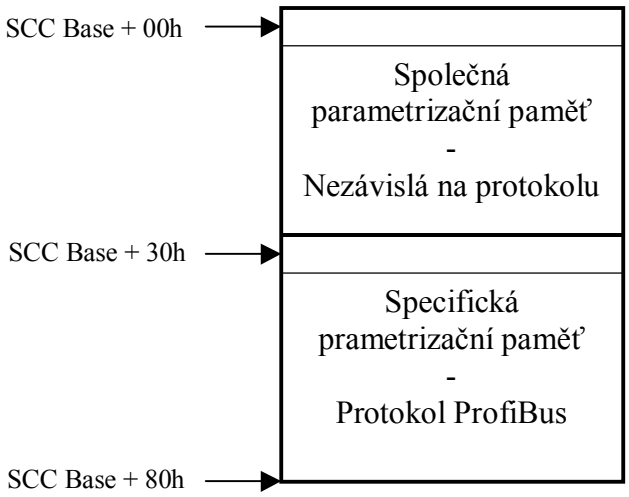
CP podporuje režim v Profibus módu pouze s mikrokódem, který je umístěn v interní RAM paměti. Výrobce procesoru (Motorola) nechal na svých webových stránkách (e-www.motorola.com) volně přístupný mikrokód ve formátu S-recordu. Tento S-record byl převeden do zdrojového textu. Byly vytvořeny dvě pole 32bitových záznamů. Obě pole je nutné překopírovat do vnitřní RAM procesoru. První (větší – cca 1024 bytů) je třeba nakopírovat od adresy DPRBASE, druhé (menší – cca 256 bytů) zkopírujeme na adresu DPRBASE + 600h.

Mikrokód běžící na CP má následující schopnosti:

- Automatická synchronizace rámců detekcí znaku SDx
- Generování/kontrola prázdných znaků, které musí předcházet některým rámcům
- Údržba pěti 16bitových chybových čítačů
- Dva registry a maska pro porovnávání adres
- Detekce chyby rámce, šumu nebo chyby parity
- Detekce prázdných znaků uprostřed rámce
- Generování/kontrola Check Sum položky rámce
- Generování/kontrola ED znaku
- Čtyři časovače: Token Rotation Timer, Syn Timer, Slot Timer, Time-out Timer
- Automatické vytváření seznamu aktivních stanic

5.8.1 Parametrizační paměť

Kanály SCC využívají podobně jako kanály SMC parametrizační paměť. První část je nezávislá na protokolu, druhá se pak liší podle zvoleného protokolu. Společná část je velmi podobná parametrizační paměti SMC.



Obrázek 5-5: Profibus – Parametrizační paměť

Část nezávislá na protokolu (její první slovo) začíná na bázové adrese příslušného SCC kanálu (SCC1). První část má následující strukturu:

Adresa	Označení	Šířka	Význam
SCC Base + 00h	**RBASE**	Word	Bázová adresa přijímacích buffer deskriptorů
SCC Base + 02h	**TBASE**	Word	Bázová adresa vysílacích buffer deskriptorů
SCC Base + 04h	**RFCR**	Byte	Přijímací funkční kód
SCC Base + 05h	**TFCR**	Byte	Vysílací funkční kód
SCC Base + 06h	**MRBLR**	Word	Maximální délka přijímaného datagramu
SCC Base + 08h	RSTATE	Long	Vnitřní stav přijímání
SCC Base + 0Ch		Long	Vnitřní ukazatel přijímaných dat
SCC Base + 10h	RBPTR	Word	Vnitřní ukazatel na přijímací buffer deskriptor
SCC Base + 12h		Word	Vnitřní počet přijatých bytů
SCC Base + 14h		Long	Dočasný registr pro příjem
SCC Base + 18h	TSTATE	Long	Vnitřní stav vysílání
SCC Base + 1Ch		Long	Vnitřní ukazatel vysílaných dat
SCC Base + 20h	TBPTR	Word	Vnitřní ukazatel na vysílací buffer deskriptor
SCC Base + 22h		Word	Vnitřní počet vyslaných bytů
SCC Base + 24h		Long	Dočasný registr pro vysílání
SCC Base + 28h	RCRC	Long	Dočasný registr pro přijaté CRC
SCC Base + 2Ch	TCRC	Long	Dočasný registr pro vysílané CRC

Tabulka 5-16: Profibus – Společná parametrizační paměť

Z obr. 5-5 je patrné, že Profibus parametrizační paměť sahá až po adresu SCC Base + 80h. Při provozu na kanálu SCC 1,2 se zdá být všechno v pořádku. Situace je složitější při použití SCC 3,4. Pohledem do tab. 5-4 zjistíme, že velikost vyhrazeného prostoru pro parametrizační paměti SCC 3,4 je 70h. Od adresy 70h začíná parametrizační paměť pro IDMA řadič. IDMA používá tuto část paměti pouze pro práci v režimu AUTO BUFFER nebo BUFFER CHAINING. Při běhu Profibus protokolu na těchto SCC musíme dbát na to, aby nepracoval příslušný DMA kanál ve zmíněných režimech.

Adresa	Označení	Šířka	Význam
SCC Base + 30h	**SD1**	Word	Start Delimiter 1
SCC Base + 32h	**SD2**	Word	Start Delimiter 2
SCC Base + 34h	**SD3**	Word	Start Delimiter 3
SCC Base + 36h	**SD4**	Word	Start Delimiter 4
SCC Base + 38h	**SC**	Word	Single Char Short Ack
SCC Base + 3Ah	**ED**	Word	End Delimiter
SCC Base + 3Ch	**IDLEC**	Word	Idle Counter
SCC Base + 3Eh	**PAREC**	Word	Receive Parity Error Count
SCC Base + 40h	**FRMEC**	Word	Receive Framing Error Count
SCC Base + 42h	**NOSEC**	Word	Receive Noise Counter
SCC Base + 44h	**PROFREF**	Word	Profibus Reference
SCC Base + 46h	**EFREC**	Word	End Frame Error Counter
SCC Base + 48h	**NMARC**	Word	Nonmatch Addr Count
SCC Base + 4Ah	**PADDR1**	Word	Address Char 1
SCC Base + 4Ch	**PADDR2**	Word	Address Char 2
SCC Base + 4Eh	**PMASK**	Word	Address Mask
SCC Base + 50h	**RFRAME**	Word	Receive Frame Register
SCC Base + 52h	MAX_cnt	Word	Max_length Counter
SCC Base + 54h	R_CK_SUM	Word	Temp RX Checksum
SCC Base + 56h	T_CK_SUM	Word	Temp TX Checksum
SCC Base + 58h	**TTRTIME**	Long	Target Rotation Time
SCC Base + 5Ch	TTHTIME	Long	Token Holding Time
SCC Base + 60h	**TRT**	Long	Token Rotation Time
SCC Base + 64h	**TOTIME**	Long	Time-out reference
SCC Base + 68h	**TOT**	Long	Time-out extension
SCC Base + 6Ch	**TSYNTIME**	Word	Syn Time
SCC Base + 6Eh	**SLOTTIME**	Word	Slot Time
SCC Base + 7Ch	**LAS_PTR**	Long	List of Active Stations
SCC Base + 26h	**TIM_PTR**	Word	Timer Pointer

Tabulka 5-17: Profibus – Specifická parametrizační paměť

Část parametrizační paměti charakteristická pro Profibus má strukturu dle tabulky **5-17**. Tučné položky by měl inicializovat uživatelský software. Za zmínku stojí určitě položka TIM_PTR. Na první pohled překvapí svým umístěním v paměti. Logicky by se dalo očekávat, že bude ležet v oblasti specifické parametrizační paměti. Proměnná má ovšem přiřazenou referenci do společné parametrizační paměti tak, že překrývá dočasný registr pro vysílání viz Tabulka 5-16. Autor mikrokódu potvrzuje, že dočasný registr pro vysílání se při činnosti CP v režimu Profibus nepoužívá a toto umístění není na závadu.

Parametry a význam některých položek:

SC – Zde se povoluje/zakazuje přijímání rámců typu Short Acknowledge. Je třeba zadat znak, který je daný normou Profibus pro rámce Short Ack (E5h). Registr má strukturu:

Bit	15	14-8	7-0
Označení	D	0	Hodnota SC – E5h

D – 0 – Přijímat rámce typu Short Ack
1 – Nepřijímat rámce typu Short Ack

IDLEC – Čítač vyslaných prázdných znaků. Hodnota koresponduje s počtem vyslaných prázdných znaků jako synchronizační část rámce. Podle Profibus normy vyhovuje hodnota 3.

PAREC, FRAMEC, NOSEC, EFREC, NMARC – Skupina 16bitových čítačů, které uživatel inicializuje a CP je inkrementuje vždy, když dojde k zaznamenání daných událostí.

PAREC – Počítá přijaté znaky s chybnou paritou.

FRAMEC – Počítá přijaté znaky s chybným stop bitem (bez stop bitu).

NOSEC – Počítá zašuměné přijaté znaky.

EFREC – Počítá chyby konců přijatých rámců – chybný kontrolní součet nebo ED.

NMARC – Počítá adresy, které nesouhlasí se zadanými (PADDR1,2) a maskou. Inkrementace se provede pouze pro rámce bez chyb.

PROFREF – Profibus Reference, používá ji mikrokód. Vyhovuje hodnota 000Ch.

PADDR1, PADDR2, PMASK – Registry k rozeznání přijímaných rámců (podle adresy v poli DA). Do vyššího bytu by se měly zapsat nuly. Nižší byte potom udává hodnotu adresy/masky použité pro komparaci s přečtenou adresou z přijímací FIFO. CP nejdříve porovná přečtenou adresu s oběma adresními registry (PADDR1,2), potom výsledek maskuje obsahem registru PMASK. Když adresa nesouhlasí a rámec je bez chyb, NMARC se inkrementuje.

RFRAME – Když je obsah registru FFFFh, CP přijímá rámce s libovolnou adresou. Hodnota 0000h podmiňuje příjem pouze těch rámců, u kterých souhlasí adresa (viz výše).

MAX_cnt, R_CK_SUM, T_CK_SUM – Vnitřní registry, které používá CP.

TTRTIME, TTHTIME, TRT – Zde je nutné zapsat hodnoty pro časování. Jedná se o Target Rotation Time, Token Holding Time, Rotation TIme pro Token Rotation Timer.

SLOTTIME – Předvolba Slot Time pro Slot Timer.

TSYNTIME – Synchronizační interval pro SYN Interval Timer.

TOTIME, TOT – Reference a rozšíření pro Time-Out Timer.

LAS_PTR – Ukazatel do externí paměti, kde CP vytváří seznam aktivních stanic.

TIM_PTR – Ukazatel do interní RAM na adresu, kde začíná tabulka Profibus časovačů.

5.8.2 Příkazy CP

Profibus mikrokód podporuje následující příkazy. Tyto příkazy se vysílají komunikačnímu procesoru prostřednictvím registru CR (viz kapitola 5.5–Příkazy pro komunikační procesor).

STOP TX – Přeruší vysílání poté, jakmile se vyprázdní vysílací FIFO.

RESTART TX – Použije se po příkazu STOP TX, aby CP začal vysílat z aktuálního BD.

INIT RX & TX PARAMETERS – Instrukce nastaví přijímací a vysílací parametry v parametrizační paměti na jejich počáteční hodnoty (zkopíruje TBASE do TBPTR, RBASE do RBPTR a nastaví TSTATE, RSTATE na nuly).

5.8.3 Konfigurační registry

K nastavení režimu práce určitého kanálu CP (SCC x) a parametrů přenosu na tomtéž kanálu slouží mimo jiné dvojice registrů (GSMR, PSMR). Jsou umístěné v interní RAM na adresách dle tabulky 3-4 v [1]. Registr GSMR je společný pro všechny protokoly, PSMR mění svůj obsah podle protokolu zvoleného v GSMR.

5.8.3.1 Registr GSMR

General SCC Mode Register určuje základní nastavení kanálu. Je široký 64 bitů a pro zápis/čtení se dělí na dvě části:

GSMR.low – Část s nižším řádem, bity 0-31,
GSMR.high – Část s vyšším řádem, bity 32-63.

Strukturu registru najdeme na straně 7-111 v [1]. Zde je důležité nastavit pole:

MODE – Protokol daného kanálu. Zapíšeme 0101b pro volbu Profibus.

Ostatní nastavení je shodné jako pro nastavení v režimu UART. Bity ENR a ENT (spuštění přijímače/vysílače) aktivujeme až po kompletní konfiguraci registrů a parametrizační paměti kanálu.

Použitelná konfigurační hodnota je: GSMR.high = 00000020h
GSMR.low = 00028005h

5.8.3.2 Registr PSMR

Protocol Specific Mode Register → Profibus Specific Mode Register. Zde najdeme specifickou konfiguraci v módu Profibus. Pro zvolený protokol má strukturu:

Bit	15	14	13-12	11-8	7	6-5	4	3-2	1-0
Označení	FLC	SL	CL	-	SYN	-	PEN	RPM	TPM

Tabulka 5-18: Profibus – PSMR

FLC – Řízení toku, **0 – Normální operace**
1 – Asynchronní řízení toku

SL – Počet stop bitů, **0 – Jeden stop bit**
1 – Dva stop bity

CL – Počet datových bitů znaku, 00 – 5 datových bitů
01 – 6 datových bitů
10 – 7 datových bitů
11 – 8 datových bitů

SYN – Synchronní mód, **0 – Normální asynchronní operace**
1 – Synchronní operace

PEN – Povolení parity, 0 – Bez parity
1 – Povolená parita

RPM – Parita přijímače, 00 – Lichá parita
01 – Trvale na nule
10 – Sudá parita
11 – Trvale na jedničce

TPM – Parita vysílače, 00 – Lichá parita
01 – Trvale na nule
10 – Sudá parita
11 – Trvale na jedničce

Registr jsem nakonfiguroval podle tučně označených variant.

5.8.4 Přijímací BD

K nastavení událostí a parametrů při ukládání přijatých dat se používá následující datová struktura:

Adresa	Bit / Význam													
	15	14	13	12	11	10	9-7	6	5	4	3	2	1	0
Offset + 0	**E**	-	**W**	**I**	L	3I	-	ID	HR	FR	ER	EF	OV	CD
Offset + 2	Objem dat [byte]													
Offset + 4	**Ukazatel na přijímací buffer**													
Offset + 6														

Tabulka 5-19: Profibus – Přijímací BD

Uživatel by měl nastavit tučně označené položky před povolením práce přijímače.

E – Prázdný buffer
- 0 – Buffer byl naplněný přijatými daty nebo se příjem do bufferu přerušil v důsledku nějaké chyby.
- 1 – Buffer je prázdný nebo přenos dat právě probíhá.

W – Poslední BD v řetězci
- 0 – Tento BD není poslední v řetězci přijímacích BD.
- 1 – BD je poslední v řetězci přijímacích BD. Po naplnění příslušného bufferu CP pokračuje s přijímáním do bufferu, na který ukazuje první BD v řetězci (leží na adrese, kterou udává RBASE v parametrizační paměti).

I – Příznak generování přerušení
- 0 – Po naplnění a uzavření BD není generováno přerušení.
- 1 – Po naplnění a uzavření BD je v SCCE (scc event register) nastaven RX bit. Ten může vyvolat přerušení, pokud je povoleno.

L – Poslední data z rámce
- 0 – Tento buffer neobsahuje poslední data z rámce. V dalším bufferu v pořadí najdeme následující data ze stejného rámce.
- 1 – Buffer obsahuje poslední data z rámce.

3I – Prázdné znaky
- 0 – Před přijímáním do tohoto bufferu proudily po sběrnici data nebo se na ní objevily méně než tři prázdné znaky.
- 1 – Před přijímáním do tohoto bufferu se na sběrnici nacházely více než tři prázdné znaky.

ID – Uzavření BD prázdnými znaky
- 1 – Buffer byl uzavřen v důsledku přijetí prázdných znaků uprostřed rámce.

HR – Chyba záhlaví
- 1 – Buffer byl uzavřen v důsledku chybného přijetí délky rámce u rámce s proměnnou délkou (LE není rovno LEr nebo po LEr nebyl přijat SD2).

FR – Chyba ohraničení
- 1 – Buffer byl uzavřen po detekci chyby ohraničení znaku. Poslední přijatý znak obsahoval chybné ohraničení (znak bez stop bitu).

PR – Chyba parity
- 1 – Znak s chybnou paritou je poslední přijatý znak a poté došlo k uzavření BD.

OV – Přetečení
- 1 – Během přijímání zprávy došlo k přetečení přijímače.

CD – Ztráta signálu CD
- 1 – Během přijímání zprávy došlo k deaktivaci signálu CD (Carrier Detect).

Objem dat – Sem zapíše CP délku přijatých dat (v bytech) po uzavření BD.

Ukazatel na přijímací datový buffer – Zde je nutné zapsat adresu prvního bytu (nejnižší adresa) v alokovaném datovém bufferu pro tento BD.

5.8.5 Vysílací BD

K nastavení událostí a parametrů při odesílání dat se používá následující datová struktura:

Adresa	Bit / Význam											
	15	14	13	12	11	10	9	8	7-5	4	3-1	0
Offset + 0	**R**	**-**	**W**	**I**	**CR**	**TC**	**-**	**P**	**-**	**ST**	**-**	CT
Offset + 2	**Objem dat [byte]**											
Offset + 4	**Ukazatel na vysílací buffer**											
Offset + 6												

Tabulka 5-20: Profibus – Vysílací BD

Uživatel by měl nastavit tučně označené položky před povolením práce vysílače.

R – Připraven k odeslání
- 0 – Datový buffer náležící k BD není připraven k odeslání. CP čeká na nastavení R bitu a poté buffer odešle do fronty.
- 1 – Buffer obsahuje platná data a je připraven k odeslání nebo odesílání právě probíhá. Po přechodu CP k tomuto BD je buffer odeslán. Po odeslání bufferu CP tento bit vynuluje.

W – Poslední BD v řetězci
- 0 – Tento BD není poslední v řetězci vysílacích BD.
- 1 – BD je poslední v řetězci vysílacích BD. Po odeslání bufferu CP pokračuje ve vysílání z bufferu, na který ukazuje první BD v řetězci (leží na adrese, kterou udává TBASE v parametrizační paměti).

I – Příznak generování přerušení
- 0 – Po odesláni bufferu není generováno přerušení.
- 1 – Po odesláni bufferu je v SCCE (scc event register) nastaven TX bit. Ten může vyvolat přerušení, pokud je povoleno.

CR – Sledování CTS
- 0 – Následující buffer bude vyslán bezprostředně po tomto.
- 1 – Sledování signálu CTS, mezi vysílanými buffery bude mezera tří prázdných znaků.

TC – Kontrolní součet
- 0 – Buffer bude uzavřen po odeslání všech dat, které obsahuje.
- 1 – Po odeslání posledního znaku z aktuálního bufferu bude odeslán kontrolní součet.

P – Předvysílání
- 0 – Žádné prázdné znaky nejsou vysílány.
- 1 – Před odesláním dat je vyslána sekvence prázdných znaků (samé jedničky). Počet vyslaných prázdných znaků udává IDLEC v parametrizační paměti.

ST – Spuštění časovače Slot Timer
0 – Nespouštět Slot Timer časovač po vyslání konce rámce.
1 – Po odeslání konce rámce se spustí časovač Slot Timer.

CT – Ztráta CTS
0 – Signál CTS byl aktivní během vysílání celého bufferu.
1 – Ztráta signálu CTS během odesílání dat z aktuálního bufferu.

Objem dat – Udává množství odesílaných dat (v bytech) z daného bufferu.

Ukazatel na odesílaný datový buffer – Zde je nutné zapsat adresu prvního bytu (nejnižší adresa) v alokovaném datovém bufferu pro tento BD.

5.8.6 Události

5.8.6.1 Registr událostí – SCCE

SCCE (SCC Event Register) v módu profibus nazýváme Profibus Event Register a v tomto režimu má specifickou strukturu. Komunikační procesor tento registr používá k hlášení událostí, které zaznamenal. Na základě těchto událostí potom může řadič přerušení generovat přerušeni pro CPU32+. Když CPU32+ pracuje v režimu bez přerušení, událost zjistí cyklickým čtením tohoto registru. Jedná se o šestnáctibitový registr, zapsáním jedničky dojde k vynulování příslušného bitu, zápis nuly nemá na obsah registru vliv. Po obsloužení událostí je nutné vynulovat příslušné bity. Přiřazení bitů vypadá následovně:

Bit	15-13	12	11	10-9	8	7-5	4	3	2	1	0
Označení	-	GLr	GLt	-	IDL	-	TxE	RxF	BSY	TX	RX

Tabulka 5-21: Profibus – Registr událostí

GLr – Na hodinovém signálu pro příjem se objevil „glitch“ impuls.

GLt – Na hodinovém signálu pro vysílání se objevil „glitch“ impuls.

IDL – Změna stavu sběrnice
Stav sběrnice (prázdné znaky nebo data) se změnil. Aktuální stav sériové linky je možné přečíst z registru SCCS. Stav se změní přijetím prázdného znaku (samé jedničky) nebo přijetím jedné nuly.

TxE – Chyba při vysílání
Během vysílání došlo k chybě (ztráta CTS signálu).

RxF – Přijatý rámec
Na SCC kanálu byl přijat kompletní rámec.

BSY – Příznak zaneprázdnění
Došlo k přijetí znaku ze sériové linky, ale CPM ho neměl kam uložit. Nastaví se, když další BD v pořadí nemá nastavený příznak E na jedna.

TX – Vyslaný buffer
Buffer byl vyslán po sériové lince.

RX – Přijatý buffer
Buffer byl přijat po sériové lince.

5.8.6.2 Registr masky – SCCM

Používá se k povolení/zakázání generování přerušení na základě objevení události. Má stejnou strukturu jako SCCE. Pokud se bit nastaví na jedničku, při objevení příslušné události se vygeneruje přerušení (jestli je správně nastaven řadič přerušení – registry CICR, CIMR). Když bit v SCCM nastavíme na nula, událost se tím zamaskuje.

5.8.7 Časovače

Mikrokód používá časovače typu RISC – Timer. Abychom dosáhli plné funkčnosti časovačů implementovaných v mikrokódu, musíme je správně definovat.
Časovače musí být umístěny na pevně dané adrese a není ji možné měnit za běhu kanálu. Profibus používá 4 časovače na každý SCC kanál, kde běží. Pokud Profibus běží zároveň na dvou kanálech CP (SCC 1, SCC 2), potřebujeme 2 * 4 = 8 časovačů.
Nejdříve je nutné nastavit tik hodin pro časování, to se provede v registru RCCR naplněním pole TIMEP na hodnotu nula. Tím se získá maximální rozlišovací schopnost časovačů, tedy $\frac{1024}{\text{System Clock}} = \frac{1024}{24\,000\,000} \doteq 42{,}67\mu s$ (= cyklus časovače).
Nastavení časovačů se provádí prostřednictvím parametrizační paměti časovačů a řídicího registru (CR – Command Register).
Parametrizační paměť časovačů má strukturu viz str. 7-13 v [1]. Zde je nutné nastavit TM_BASE na adresu do interní RAM, kde bude umístěna tabulka časovačů. Od této adresy musí být k dispozici alespoň 32 bytů pro osm časovačů (čtyři byty pro každý časovač). Časovače alokujeme v paměti ve vzestupném pořadí, abychom ušetřili paměťový prostor.
Pokud již nějaké časovače používáme nebo mikrokód Profibus běží na více než jednom kanále CP, musíme určit adresu každé čtveřice časovačů pro příslušný kanál CP a naprogramovat tuto adresu do TIM_PTR v parametrizační paměti pro stejný kanál (viz Tabulka 5-17). Když Profibus běží pouze na jednom kanálu a nejsou použity žádné další časovače, je hodnota TM_BASE shodná s TIM_PTR.
Při běhu Profibus protokolu na dvou kanálech CP (SCC 1, SCC 2) může situace vypadat následovně:

Kanál	Adresa v TIM_PTR
SCC 1	TM_BASE + 00h
SCC 2	TM_BASE + 10h

Tabulka 5-22: Profibus – Časovače 1

Umístění časovačů do tabulky časovačů v interní RAM procesoru ukazuje následující tabulka:

Adresa	Označení	Šířka	Název časovače
TM_BASE + 00	RTRT1	Long	Token Rotation Timer 1
TM_BASE + 04	RSYN1	Long	Sync Timer 1
TM_BASE + 08	RST1	Long	Slot Timer 1
TM_BASE + 0C	RTOT1	Long	Time-Out Timer 1
TM_BASE + 10	RTRT2	Long	Token Rotation Timer 2
TM_BASE + 14	RSYN2	Long	Sync Timer 2
TM_BASE + 18	RST2	Long	Slot Timer 2
TM_BASE + 1C	RTOT2	Long	Time-Out Timer 2

Tabulka 5-23: Profibus – Časovače 2

Všechny záznamy jsou 32bitové, nižší slova obsahují okamžité hodnoty časovačů, vyšší potom časovací předvolby. Uživatel by měl přistupovat na tyto registry pouze pomocí instrukce SET TIMER vyslané do registru příkazů (CR). Pro dosažení maximální periody je předvolba časovače rovna 0000h, odpovídá 65536 cyklům časovače. Maximální perioda $\doteq 65536 \cdot 42{,}67 \cdot 10^{-6}\,\mathrm{s} \doteq 2{,}75\,\mathrm{s}$.

5.8.7.1 Token Rotation Timer

Skládá se ze tří registrů a časovače, jejich umístění v paměti vypadá následovně:

Registr	Umístění	Část [bit] / Označení	
		31 - 16	15 - 0
TTRTIME	Parametrizační RAM	TR_EXT	TR_TIME
TTHTIME	Parametrizační RAM	TTH_EXT	TTH_TIME
TRT	Parametrizační RAM	TRT_EXT	TRT_TIME
RTRT	Tabulka časovačů	Předvolba	Aktuální hodnota - RTRT

Tabulka 5-24: Token Rotation Timer

TTRTIME – Udává čas Target Rotation Time, naprogramujeme ho na požadovanou hodnotu a potom ho používáme jako referenční hodnotu. Mikrokód tento registr bere v úvahu jako referenční hodnotu.

TTHTIME – Udává čas Token Holding Time, mikrokód ho aktualizuje po přijetí rámce typu token.

TRT – Čas Token Rotation Time, je zaveden jako rozšíření k časovači RTRT a údržba je ponechána na uživateli.

Tento časovač je implementován tak, že využívá RISC časovač (časovač, který udržuje CP). K časování delší doby než 2,75s (odpovídá 16 bitům) je zde zavedeno rozšíření TRT_EXT, jeho údržbu zařizuje uživatelský software.
Když dojde k přijetí rámce token, kde adresa DA souhlasí s adresou v PADDR1 v parametrizační paměti Profibusu, RTRT se naplní 16bitovou hodnotou z TR_TIME a TRT se naplní 32bitovou hodnotou z TTRTIME.
Když je přijatý druhý rámec typu token, zbývající čas (Token Holding Time) se uloží do TTHTIME. Nižší slovo se přečte z RTRT a vyšší byte se čte z TRT_EXT.

Záležitostí uživatele je aktualizovat a udržovat TRT_TIME. Spočívá to v překopírování hodnoty RTRT do TRT_TIME.
Když vyprší doba časovač RTRT, mikrokód automaticky aktualizuje TRT_TIME a vygeneruje přerušení.
Obsluha přerušení potom musí dekrementovat TRT_EXT.

5.8.7.2 Syn Interval Timer

Je to 16bitový časovač, situace v paměti je následující:

Registr	Umístění	Část [bit] / Označení	
		31 - 16	15 - 0
TSYNTIME	Parametrizační RAM	-	TSYNTIME
RSYN	Tabulka časovačů	Předvolba	Aktuální hodnota - RSYN

Tabulka 5-25: Syn Interval Timer

TSYNTIME – Syn Interval Timer, je to 16bitová hodnota, kterou nastavuje uživatelský software.

Když dojde k přijetí 33 jedničkových bitů, mikrokód naplní časovač RSYN hodnotou přednastavenou v TSYNTIME. Až vyprší časovací doba, CP vygeneruje maskovatelné přerušení.

5.8.7.3 Slot Timer

Registr	Umístění	Část [bit] / Označení	
		31 - 16	15 - 0
SLOTTIME	Parametrizační RAM	-	SLOTTIME
RST	Tabulka časovačů	Předvolba	Aktuální hodnota - RST

Tabulka 5-26: Slot Timer

SLOTTIME – Slot Timer, je to 16bitová předvolba, kterou nastavuje uživatelský software na požadovanou hodnotu.

Když mikrokód vyšle rámec prostřednictvím TxBD, kde je nastaven bit ST, RST se naplní předvolbou SLOTTIME. Po přijetí prvního znaku následujícího rámce se časování zastaví. Když časovač vyprší dříve, vygeneruje se maskovatelné přerušení. Musí ho obsloužit uživatelský software.

5.8.7.4 Time-Out Timer

Registr	Umístění	Část [bit] / Označení	
		31 - 16	15 - 0
TOTTIME	Parametrizační RAM	TTM_EXT	TTM_TIME
TOT	Parametrizační RAM	TOT_EXT	TOT_TIME
RTOT	Tabulka časovačů	Předvolba	Aktuální hodnota - RTOT

Tabulka 5-27: Time-Out Timer

TOTTIME – Referenční hodnota Time-Out Time, uživatelský software ji nastavuje a společně s mikrokódem používá jako referenci.

TOT – Time-Out Timer Extension, je to rozšíření RTOT. Hodnotu udržuje uživatelský software.

Časovač využívá 16bitový čítač (RTOT), jeho rozšíření TOT_EXT udržuje uživatelský software. Celkově tedy máme k dispozici 32bitový časovač. RTOT se spustí, jakmile se vyšle nebo přijme poslední bit z bufferu. RTOT se plní z 16bitového TTM_TIME registru, TOT se plní z 32bitového TOTTIME. Tyto operace provádí mikrokód.
RTOT se zastaví, pokud se přijme nový buffer.
Pokud chceme udržovat TOT_TIME, musíme to zabezpečit softwarově, překopírováním RTOT do TOT_TIME.
Po uběhnutí RTOT je to provedeno automaticky, mikrokód překopíruje RTOT do TOT_TIME a vygeneruje přerušení.
Dekrementaci TOT_EXT má za úkol uživatelská obslužná rutina přerušení.

5.8.7.5 Inicializace časovačů

V závislosti na použitém kanálu je nutné nastavit některé registry.

Kanál	TCHECKSUM	PROFREF	RCHECKSUM
SCC 1	0008h	000Ch	0DB6h
SCC 2	0080h	00C0h	0DB6h
SCC 3	0800h	0C00h	0DB6h
SCC 4	8000h	C000h	0DB6h

Tabulka 5-28: Inicializace časovačů

Ukazatel na časovače TIM_PTR by měl mít poslední tři bity rovny 000b.
RTRT by měl být nastaven jako platný (valid) a cyklicky spouštěný (restart). Měl by být nastaven na maximální čas. To odpovídá zápisu C000FFFF do TM_CMD registru a poté 0851 do CR pro vyvolání příkazu SET TIMER.
RSYN by měl být nastaven jako platný (valid) a cyklicky spouštěný (restart). Časovací předvolbu nastavíme na maximum. To odpovídá zápisu C001FFFF do TM_CMD registru a poté 0851 do CR.
RST musíme nejdříve nastavit jako povolený (enabled), potom ho zakázat. V obou nastaveních volíme maximální čas. Odpovídající posloupnost je: zápis 8002FFFF do TM_CMD, potom 0851 do CR, 0002FFFF do TM_CMD a 0851 do CR.

5.8.8 Seznam aktivních stanic

Mikrokód je schopen sestavovat seznam aktivních stanic – LAS (List of Active Stations). List aktivních stanic je interpretován jako pole bytů o délce 128. Pole se nachází v externí paměti SRAM. Index v poli odpovídá adrese připojené stanice, obsah potom odpovídá adrese další aktivní stanice v pořadí. Do položky LAS_PTR v parametrizační paměti musíme zapsat adresu pole v externí paměti. Po přijetí rámce token se stanice, které je token určený přidá do seznamu aktivních stanic.

5.9 Zdrojový kód

Celý projekt je sestaven z několika modulů, které jsou odděleně překládány a poté slinkovány do jednoho výstupního souboru.
Výpis modulů a jejich funkce:

Modul	Význam
`application.c`	hlavní modul aplikace, obsahuje funkci main, vysílá/přijímá po RS-232 a Profibusu
`board.h`	definice závislé na hw konfiguraci pro desku Profi-USB v1.0
`pram.h`	definice struktur parametrizačních pamětí různých protokolů (UART, Profibus, Ethernet, ...)
`profi.c`	modul obsahuje inicializaci a vnitřní procedury pro běh Profibusu na libovolném kanále SCC(1-4)
`profi.h`	definice konstant a export funkcí
`profi_example.c`	modul určený pro konkrétní instanci Profibusu (kanál SCC1)
`profi_example.h`	definice konstant a export funkcí
`profi_routines.c`	funkce pro sestavování a vysílání rámců na kanálech Profibus, nezávislé na kanálu
`profi_routines.h`	export funkcí
`quicc.h`	definice datové struktury registrů interní RAM, definice struktury BD (Buffer Deskriptoru)
`registers.h`	definice konstant pro některé registry
`scc_config.h`	definice společné pro všechny kanály, tabulka parametrů kanálů (`scc_table`)
`simple.c`	obsahuje funkce společné pro všechny kanály komunikačního procesoru, inicializaci systému
`simple.h`	definice a exporty
`smc_uart.c`	funkce pro inicializaci a vnitřní obsluhu SMC kanálu v režimu UART, společné pro oba kanály SMC
`smc_uart.h`	definice a exporty
`smc_uart_example.c`	specifický modul pro instanci UARTu na kanálu SMC2
`smc_uart_example.h`	definice a exporty
`ucode.c`	modul obsahující funkci pro natažení mikrokódu do interní paměti RAM procesoru (`copy_ucode`)
`ucode.h`	export funkce `copy_ucode`

Tabulka 5-29: Moduly projektu

Aplikace běží podle následující posloupnosti:

Nejdříve proběhne systémová inicializace (funkce `quicc_init`). Zde se založí globální proměnné – struktura `scc_table` a pointer na strukturu `quicc`. Následně se spustí funkce `copy_ucode`, která překopíruje mikrokód Profibusu do vnitřní paměti procesoru.

Potom se začnou konfigurovat kanály komunikačního procesoru v pořadí od nejnižšího indexu použitého SCC kanálu po nejvyšší index použitého SMC kanálu. Tato konfigurace se pro kanál typu SCC pracující s protokolem Profibus skládá ze spuštění následujících funkcí:

Funkce	Význam
`scc_init`	vloží počet BD pro vysílání/příjem do tabulky kanálů `scc_table`
`profi_init`	inicializace kanálu pro režim Profibus, inicializace řetězců BD, nastavení přenosových parametrů, inicializace časovačů, zapsání pointerů na funkce pro obsluhu přijímání, chyb během přenosu do tabulky kanálů `scc_table`
`profi_application_init`	inicializace konkrétní instance Profibus kanálu, zapsání vektoru pro obsluhu přerušení daného kanálu do tabulky vektorů

Tabulka 5-30: Inicializace kanálu Profibus

Pro konfiguraci kanálu typu SMC do režimu UART je posloupnost funkcí podobná:

Funkce	Význam
`smc_init`	vloží počet BD pro vysílání/příjem do tabulky kanálů `scc_table`
`smc_uart_init`	inicializace kanálu pro režim UART, inicializace řetězců BD, nastavení přenosových parametrů, zapsání pointerů na funkce pro obsluhu přijímání, chyb během přenosu do tabulky kanálů `scc_table`
`uart_application_init`	inicializace konkrétní instance UART kanálu, zapsání vektoru pro obsluhu přerušení daného kanálu do tabulky vektorů

Tabulka 5-31: Inicializace kanálu UART

K povolení práce komunikačních kanálů se potom spustí funkce `scc_start(`*index SCC kanálu*`)` a `smc_start(`*index SMC kanálu*`)`.

5.9.1 Obsluha přerušení

Pro správnou funkci obslužných rutin přerušení sestavíme pro každou z nich tzv. handler, který má následující strukturu:

```
typedef struct irq_handler {
  void            *handler)(int, void *, struct pt_regs *);
  unsigned long   flags;
  void            *dev_id;
  const char      *devname;
  struct irq_handler *next;
  short            vectno;
} irq_handler_t;
```

Do položky `handler` zapíšeme pointer na funkci obsluhy přerušení (např. scc1_isr), do pole `devname` uložíme pointer na řetězec, který definuje název obsluhovaného zařízení (např. scc1), ostatní položky není nutné vyplňovat, zapíšeme do nich nuly.
Handler je potom nutné zaregistrovat pomocí funkce `add_irq_handler` v modulu `system_sup.c`.
Při vyvolání přerušení od kanálu Profibus dojde k hierarchickému spouštění funkcí:

`scc1_isr` – obslužná rutina konkrétního kanálu

`scc_interrupt(0)` – funkce obsluhy přerušení pro index kanálu 0 (SCC1), zde se prohlédne tabulka kanálů `scc_table` a spustí se funkce, na kterou odkazuje pointer v této tabulce (`scc_table[0].intr_ptr`)

`profi_interrupt` – na tuto funkci odkazuje položka `intr_ptr` tabulky kanálů `scc_table`, zde se vyhodnotí, jaká událost způsobila požadavek přerušení a podle toho se předá řízení funkci určené pro obsluhu dané události

Posloupnost pro kanál UART je obdobná:

`smc2_isr` – obslužná rutina konkrétního kanálu

`smc_interrupt(1)` – zavolá funkci `scc_interrupt`, kde zvýší index o 4 (počet SCC kanálů)

`scc_interrupt(5)` – funkce obsluhy přerušení pro index kanálu 5 (SMC2), zde se prohlédne tabulka kanálů `scc_table` a spustí se funkce, na kterou odkazuje pointer v této tabulce (`scc_table[5].intr_ptr`)

`uart_interrupt` – na tuto funkci odkazuje položka `intr_ptr` tabulky kanálů `scc_table`, zde se vyhodnotí, jaká událost způsobila požadavek přerušení a podle toho se předá řízení funkci určené pro obsluhu dané události

5.9.2 Ukládání dat

Data, která se přijmou na sériových kanálech se automaticky kopírují do externí paměti RAM pomocí SDMA kanálů komunikačního procesoru. Adresu ukládání udávají přijímací buffer deskriptory v interní RAM řadiče, které jsou zinicializovány funkcí `profi_init`, resp. `smc_uart_init`.
Každý komunikační kanál má přidělenou svou strukturu (`workspace`), která uchovává pohromadě proměnné potřebné k provozu kanálu ve zvoleném protokolu. Pointer na tyto struktury obsahuje tabulka kanálů `scc_table`. Všechny kanály používají k ukládání dat staticky alokovaný buffer, který je členem struktury `workspace`.

Struktura na ukládání dat pro kanál UART:

```
typedef struct smc_uart_store_buf{
     volatile SMC_UART_IN_BUF in_buf[NO_OF_STORE_BUF];
     volatile unsigned short read_pos;
     volatile unsigned short store_pos;
     volatile unsigned char buf_overflow;
} SMC_UART_STORE_BUF;
```

Položka `in_buf[]` představuje ukládací buffer, který má následující strukturu:

```
typedef struct smc_uart_in_buf {
     volatile unsigned char buf[UART_BUF_LENGTH];
     volatile unsigned short length;
     volatile unsigned short closed;
} SMC_UART_IN_BUF;
```

Člen `buf` je ukládací pole pro přijatá data z rámců. Ploška `length` udává délku přijatých dat do bufferu (v bytech), `closed` potom indikuje, zda už skončilo přijímání dat do tohoto bufferu a je možné je odtud číst platná data.

Položka `read_pos` udává pozici rámce pro čtení, `store_pos` říká, na jakou pozici je možné umístit právě uvolněný buffer deskriptor pro příjem nových dat. Člen `buf_overflow` se používá k indikaci přepisování ještě nepřečtených dat nově přijatými daty.

Struktura pro ostatní kanály je podobná, liší se pouze velikostí a množstvím rámců k uložení.

5.9.3 Odesílání dat

Odesílání dat na sériové kanály se uskutečňuje přiřazením adresy a délky dat do následujícího vysílacího BD v pořadí. Pro kanál UART je třeba zavolat funkci `smc_uart_tx`, která se nachází v modulu `smc_uart.c`. Jiná možnost je použití funkce `printf`, která pro výstup používá funkci `write`, ta je překrytá v modulu `system_stub.c`. Pro kanály pracující s protokolem Profibus jsou vytvořené funkce, které sestavují požadované rámce a nastavují příznakové bity v buffer deskriptorech. Tyto funkce jsou umístěné v modulu `profi_routines.c`.

6. Závěr

Podařilo se navrhnout, realizovat a oživit zařízení ProfiUSB. Schéma i podklady pro výrobu (deska plošných spojů, servisní potisk, nepájivá maska, vrtací soubory) jsem vytvářel pomocí návrhového systému Orcad. Tomu bych vytkl mnohé nedostatky (nepříjemné chování funkce automatického umísťování prokovených otvorů, potíže s opětovným generováním vrtacích souborů, atd.), ale vhledem ke složitosti výsledného řešení si práci bez něj neumím představit. Volba součástkové základny převážně SMD a oboustranná montáž přispěla k minimalizaci rozměrů přístroje.

Výrobu čtyřvrstvého plošného spoje jsem přenechal profesionálům z firmy Pragoboard. Jednotlivé motivy prošly pracovištěm elektrického testování, aby se zamezilo výskytu vlasových spojů nebo naopak přerušených spojů. Jejich oprava po osazení by byla velmi obtížná, v některých případech téměř nemožná.

Po osazení desky specializovanou firmou bylo možné přistoupit k oživování. Po odstranění drobných chyb, které byly do schématu vneseny převážně chybnou dokumentací od firmy Siemens, bylo zařízení připraveno k vývoji softwaru.

Nejdříve bylo nutné nastavit prostředí GDB pro základní konfiguraci mikroprocesoru. Sestavil jsem konfigurační skript, který definuje obsah systémových registrů procesoru a obsahuje makro-funkce k nastavení registrů chip select pinů. Po nastavení překladače (GCC) již bylo možné spouštět na mikroprocesoru aplikace napsané v jazyce GNU C.

Testováním jednoduchých aplikací jsem zjistil, že USB čip vyvolává cyklicky přerušení, které v jeho konfiguraci nelze zakázat. Aby USB svým chováním příliš neobtěžovalo mikrořadič, je vhodné nastavit masku přerušení (ve stavovém slově mikroprocesoru) na hodnotu 3h nebo vyšší.

Vyzkoušením základních funkcí mikroprocesoru jsem přešel k programování rozhraní pro komunikační procesor. Jako základ jsem převzal ukázkové části kódu od společnosti Motorola. Tento software byl užitečný, obsahoval však množství chyb, které se obtížně a zdlouhavě odstraňovaly. Podařilo se spustit komunikaci přes sériové rozhraní RS-232, kterou jsem testoval s pomocí emulátoru terminálu Seyon.

Poté přišlo na řadu sestavení konfiguračních a obslužných rutin pro kanál komunikačního procesoru pracující v režimu Profibus. Vytvořil jsem funkce pro automatické ukládání přijatých rámců do vnější paměti procesoru. Komunikační procesor tak ukládá pomocí SDMA řadiče obsah rámců zcela samostatně, naplní celou kapacitu vyhrazené paměti s minimální spoluprácí 32bitového jádra hlavního procesoru. Práci jsem zakončil vytvořením funkcí pro skládání a vysílání rámců po síti Profibus. Funkčnost jsem otestoval nejdříve osciloskopem a potom konfigurací a přenosem dat se zařízením typu slave. Na komunikačním kanálu Profibus jsem dosáhl maximální přenosovou rychlost 1,5Mb/s.

Tato práce byla náročná, ale velmi zajímavá. Byl bych rád, kdyby na tento základ někdo navázal a vytvořil modul pro připojení k Ethernetu, rozšířil softwarové vybavení o obsluhu kanálu Ethernet a styk s osobním počítačem prostřednictvím sítě USB.

7. Přílohy

Kompletní dokumentaci včetně vytvořeného softwaru obsahuje přiložené CD-ROM.

7.1 Technické údaje

Maximální rozměry	154 x 151 x 19 mm
Hmotnost	cca 120 g
Napájecí napětí	5 V DC
Maximální odběr proudu	cca 400 mA
Frekvence krystalového oscilátoru	24 MHz
Frekvence krystalu pro USB čip	6 MHz
Velikost externích pamětí	
Boot Flash EPROM	1 MB
Flash EPROM	2 MB
SRAM	2 MB
Izolační napětí optických oddělovačů	
RS-232	2,5 kV
Profibus 1	2,5 kV
Profibus 2	2,5 kV
Maximální přenosová rychlost	
RS-232	120 kb/s
Profibus 1	1,5 Mb/s
Profibus 2	1,5 Mb/s
USB	12 Mb/s

Tabulka 7-1: Technické údaje

7.2 Konfigurace hardwaru

7.2.1 Bity CONFIG 0 - 2

K nastavení šířky sběrnice po resetu pro signál $\overline{CS0}$ na 16 bitů a povolení funkce 32bitového jádra procesoru (CPU32+ core) je třeba nastavit propojky (jumpery) podle následující tabulky:

Signál	Logická úroveň	Jumper	Stav
CONFIG 0	1	J39	otevřený
CONFIG 1	0	J42	uzavřený
CONFIG 2	1	J44	otevřený

Tabulka 7-2: Jumpery CONFIG 0 - 2

Podrobný popis viz kapitola 2.1.13 Initial Configuration Pins v [1].

7.2.2 Bity MODCK 0 - 1

Konfigurují časovací obvody řadiče. Následující nastavení odpovídá použití externího krystalového oscilátoru, systémová frekvence řadiče bude rovna frekvenci oscilátoru.

Signál	Logická úroveň	Jumper	Stav
MODCK 0	1	J22	otevřený
MODCK 1	0	J23	uzavřený

Tabulka 7-3: Jumpery MODCK 0 - 1

Detailní popis časovacích obvodů je uveden v kapitole 6.5 – SIM60 System Clock generation v [1].

7.2.3 Přiřazení komunikačních rozhraní

Přístroj je zkonstruován tak, aby bylo možné volit připojení periferních komunikačních obvodů (RS-232, Profibus1, Profibus2) k mikrořadiči Motorola. Rozhraní Ethernet je pevně připojené na komunikační kanály SCC1, SCC2, protože zde lze dosáhnout maximální přenosovou rychlost (10Mb/s).
Následující tabulky popisují, které zkratovací propojky je nutné uzavřít pro přiřazení periferie komunikačnímu kanálu řadiče.

Přiřazení	Jumper	Význam
RS-232→SMC2	J4	RxD
	J7	SYN
	J10	TxD
RS-232→SCC4	J5	RxD
	J8	CTS
	J11	TxD
	J12	RTS

Tabulka 7-4: Přiřazení RS-232 k SMC2, SCC4

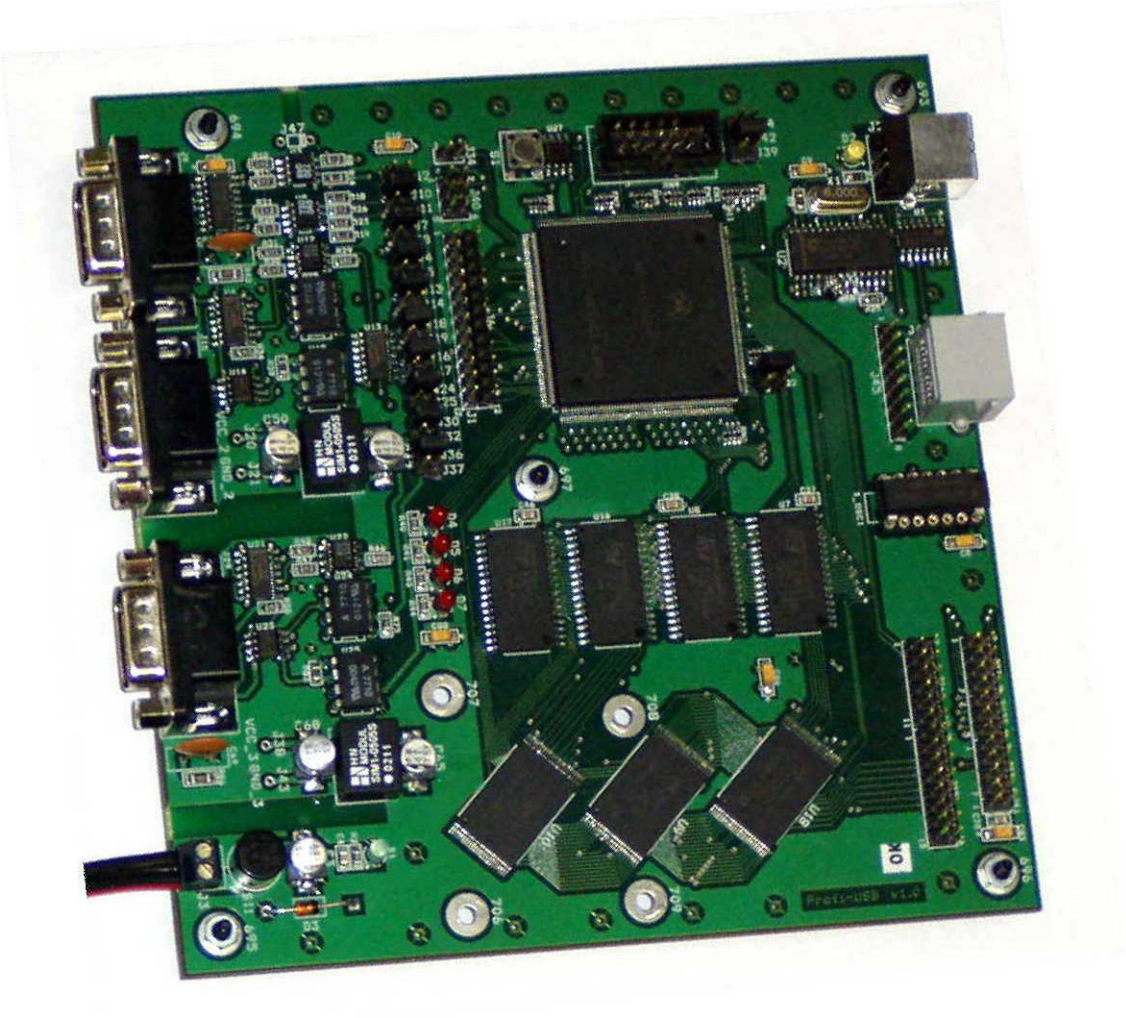
Přiřazení	Jumper	Význam
Profibus1→SCC1	J14	RTS
	J17	TxD
	J19	RxD
Profibus1→SCC3	J13	RTS
	J16	TxD
	J18	RxD

Tabulka 7-5: Přiřazení Profibus1 k SCC1, SCC3

Přiřazení	Jumper	Význam
Profibus2→SCC2	J25	RTS
	J32	TxD
	J37	RxD
Profibus2→SCC4	J24	RTS
	J30	TxD
	J36	RxD

Tabulka 7-6: Přiřazení Profibus2 k SCC2, SCC4

7.3 Fotografie



Obrázek 7-1: Fotografie Profi-USB

8. Literatura

[1] Motorola: *MC68360 User's manual*. Poslední změna 31.7.1996, velikost souboru 4,45MB, 959 stran. URL: http://e-www.motorola.com [cit. 11/2001]

[2] Motorola: *PROFIBUS µCODE - User's Manual.* European Data Communications Group, East Kilbride, Scotland, August 19th, 1996. Rev 2.2, 24 stran. URL: http://e-www.motorola.com [cit. 11/2001]

[3] Motorola: Instrukční soubor CPU32, rozhraní pro Ethernet a další dokumenty k mikroprocesoru Motorola 68360.
URL: http://e-www.motorola.com [cit. 11/2001]

[4] AMD: Manuál k paměti FALSH EPROM AM29F800.
URL: http://www.amd.com [cit. 11/2001]

[5] ST Microelectronics: Referenční manuál k paměti SRAM M68AF511AL a logickým obvodům. URL: http://www.st.com [cit. 11/2001]

[6] Texas Instruments: Manuál k resetovacímu obvodu TL7705A, manuál k budiči RS-485 (Profibus DP) 75ALS176D.
URL: http://www.ti.com [cit. 11/2001]

[7] Philips: *PDIUSBD12 Product data*. Revision 06, 23 April 2001. 36 stran
URL: http://www.semiconductors.philips.com [cit. 11/2001]

[8] Philips: Dokumenty k obvodu rozhraní USB PDIUSBD12.
URL: http://www.semiconductors.philips.com [cit. 11/2001]

[9] Agilent Technologies: Katalogové listy k optočlenům HCPL0730, HCPL7710, HCPL0601. URL: http://www.agilent.com [cit. 11/2001]

[10] Maxim Integrated Products: Katalogové listy k budiči sběrnice RS-232 MAX232CSE. URL: http://www.maxim-ic.com [cit. 11/2001]

[11] EPSON: Katalogové listy pro krystalový oscilátor SG – 51, 24MHz.
URL: http://www.epson-electronics.de [cit. 11/2001]

[12] Záhlava, V.: *OrCAD pro Windows.* Grada Publishing, Praha, 1999. 124 stran

[13] PROFIBUS: *PROFIBUS Specification*, Normative Parts of PROFIBUS -FMS, -DP, -PA according to the European Standard EN 50 170, Volume 2. Karlsruhe, March 1998. Edition 1.0

[14] Compaq, Intel, Microsoft, NEC: *Universal Serial Bus Specification.* Revision 1.1, September 23, 1998

[15] PROFIBUS – ProfiNet, Dokumentace. URL: http://www.profibus.com

[16] USB, Dokumentace. URL: http://www.usb.org

[17] Herout, P.: *Učebnice jazyka C*. III. upravené vydání. Nakladatelství Kopp, České Budějovice, 2001. 269 stran

[18] Herout, P.: *Učebnice jazyka C – 2. díl*. Nakladatelství Kopp, České Budějovice, 2002. 236 stran

[19] Stallman, R. M. – Pesch, R. H.: *Debugging with GDB*. Free Software Foundation, Boston, USA, 1998

[20] Píša, P.: *BDM Interface for Motorola 683xx MCU, Usage with GDB Debugger*, ČVUT FEL Katedra řídicí techniky, Praha, 1999

[21] Burget, P.: *Linková vrstva Profibus*, ČVUT FEL Katedra řídicí techniky, Praha

[22] Brandejs, M.: *Unix-Linux – praktický průvodce*. Grada Publishing, Praha, 1996. 344 stran

[23] Ifak: Profibus – USB interface, dokumentace.
URL: http://www.ifak.fhg.de [cit. 5/2002]

[24] Moeller: Gateways, dokumentace.
URL: http://www.moeller.net [cit. 5/2002]

Dodavatelé elektronických součástek a výrobce plošných spojů

[25] GM Electoronic, URL: http://www.gme.cz

[26] GES Electoronics, URL: http://www.ges.cz

[27] Spoerle Electoronic, URL: http://www.spoerle.cz

[28] Plošné spoje Pragoboard, URL: http://www.pragoboard.cz