VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

DIPLOMOVÁ PRÁCE

Brno, 2021 Bc. Dominik Galád

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV TELEKOMUNIKACÍ
DEPARTMENT OF TELECOMMUNICATIONS

USB ADAPTÉR PRO PŘIPOJENÍ DISKETOVÝCH MECHANIK
FLOPPY DISK DRIVE TO USB ADAPTER

DIPLOMOVÁ PRÁCE
MASTER'S THESIS

AUTOR PRÁCE — **Bc. Dominik Galád**
AUTHOR

VEDOUCÍ PRÁCE — **Ing. Ondřej Krajsa, Ph.D.**
SUPERVISOR

BRNO 2021

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Telekomunikační a informační technika**

Ústav telekomunikací

Student: Bc. Dominik Galád ***ID:*** 195300
Ročník: 2 ***Akademický rok:*** 2020/21

NÁZEV TÉMATU:

USB adaptér pro připojení disketových mechanik

POKYNY PRO VYPRACOVÁNÍ:

Navrhněte a realizujte adaptér pro připojení 3,5" a 5,25" disketových mechanik, umožňující čtení a zápis dat. Adaptér se bude k PC připojovat pomocí rozhraní USB.

DOPORUČENÁ LITERATURA:

[1] TAMMY, Noergaard. Embedded Processors. Embedded Systems Architecture - A Comprehensive Guide for Engineers and Programmers, 2nd edition. Elsevier, 2013, s. 1-4. ISBN 9780123821966.

[2] American National Standard for Information Technology - AT Attachment Interface with Extensions (ATA-2) [online]. American National Standards Institute [cit. 2020-09-14]. Dostupné z: http://www.t13.org/Documents/UploadedDocuments/project/d0948r4c-ATA-2.pdf

Termín zadání: 1.2.2021 ***Termín odevzdání:*** 24.5.2021

Vedoucí práce: Ing. Ondřej Krajsa, Ph.D.

prof. Ing. Jiří Mišurec, CSc.
předseda rady studijního programu

UPOZORNĚNÍ:
Autor diplomové práce nesmí při vytváření diplomové práce porušit autorská práva třetích osob, zejména nesmí zasahovat nedovoleným způsobem do cizích autorských práv osobnostních a musí si být plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č.40/2009 Sb.

ABSTRAKT

Práce se zabývá návrhem USB ovladače disketové mechaniky. Je v ní popsána teorie potřebná k návrhu vlastního ovladače disketové mechaniky a vývěr jednotlivých komponent. Dále je uvedeno schéma návrhu ovladače disketové mechaniky. Jsou popsány funkce pro ovládání disketové mechaniky, a také úskalí použitého procesoru ATSAMD21J18A-AU.

KLÍČOVÁ SLOVA

Disketová mechanika, ARM® Cortex-M0+, Disketa, USB, ATSAMD21J18A-AU, CRC

ABSTRACT

The work deals with the design of a USB floppy drive driver. It describes the theory needed to design your own floppy drive driver and the output of individual components. The following is a schematic of the floppy drive driver design. The functions for controlling the floppy drive are described, as well as the pitfalls of the ATSAMD21J18A-AU processor used.

KEYWORDS

Floppy disk drive, ARM® Cortex-M0+, floppy disk, USB, ATSAMD21J18A-AU, CRC

Vysázeno pomocí balíčku `thesis` verze 4.03; `http://latex.feec.vutbr.cz`

GALÁD, Dominik. *USB adaptér pro připojení disketových mechanik*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2021, 61 s. Diplomová práce. Vedoucí práce: Ing. Ondřej Krajsa, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Bc. Dominik Galád
VUT ID autora:	007
Typ práce:	Diplomová práce
Akademický rok:	2020/21
Téma závěrečné práce:	USB adaptér pro připojení disketových mechanik

Prohlašuji, že svou závěrečnou práci jsem vypracoval samostatně pod vedením vedoucí/ho závěrečné práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou všechny citovány v práci a uvedeny v seznamu literatury na konci práce.

Jako autor uvedené závěrečné práce dále prohlašuji, že v souvislosti s vytvořením této závěrečné práce jsem neporušil autorská práva třetích osob, zejména jsem nezasáhl nedovoleným způsobem do cizích autorských práv osobnostních a/nebo majetkových a jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., o právu autorském, o právech souvisejících s právem autorským a o změně některých zákonů (autorský zákon), ve znění pozdějších předpisů, včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

Brno ....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Ondřeji Kajsovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci. Také za obrovské množství času, které strávil při konzultacích nad problematikou diplomové práce a v neposlední řadě za jeho neprodlené reakce na mé dotazy.

Obsah

Úvod **12**

1 Disketová mechanika **13**
1.1 Přístup na médium 14

2 Disketa **15**
2.1 Mechanické vlastnosti 15
2.2 Uspořádaní dat 17
2.3 Skutečné uspořádání dat 18

3 Cyklický redundantní součet **20**
3.1 Výpočet 20

4 Modifikovaná frekvenční modulace **22**

5 Souborový systém FAT **24**
5.1 FAT12 25

6 USB **26**
6.1 USB 1.0 a 1.1 26
6.2 USB 2.0 26
6.3 USB konektory 26

7 USB 2.0 **28**
7.1 Topologie sběrnice 28
7.2 Komunikace po sběrnici 29
7.3 Typy datových přenosů 29
7.4 Komunikace mezi Hostem a zařízením 30
7.5 Tok dat 30
7.6 Koncový bod 31
7.7 Roury 31
7.8 Rámec a mikrorámec 32
7.9 Řídicí přenosy 32
7.10 Nepřetržité přenosy 32

8 Třídy zařízení USB **33**
8.1 Třída zařízení Mass Storage 33
8.2 UFI Příkazy 33

9 Mikrokontroler **35**
9.1 Požadavky . . . 35
9.2 ARM® Cortex™-M0+ . . . 35
9.3 Atmel SAM D . . . 36

10 Výsledky studentské práce **37**
10.1 Volba komponent . . . 37
10.1.1 Verze 1 . . . 37
10.1.2 Verze 2 . . . 38
10.2 Návrh desky plošného spoje . . . 38
10.2.1 Verze 1 . . . 39
10.2.2 Verze 2 . . . 39
10.3 Ochranný kryt ovladače disketové mechaniky . . . 40
10.4 Software ovladače disketové mechaniky . . . 41
10.4.1 Inicializace systému . . . 41
10.4.2 Detekce připojení disketové mechaniky . . . 41
10.4.3 Nastavení čtecí/zapisovací hlavy na stopu 0 . . . 43
10.4.4 Detekce vložení diskety . . . 43
10.4.5 Zapnutí motoru pro rotaci diskety . . . 44
10.4.6 Nastavení čtecí hlavy na konkrétní stopu . . . 45
10.4.7 Volba strany diskety . . . 46
10.4.8 Vyčítaní dat z diskety . . . 46
10.4.9 Zápis dat na disketu . . . 47
10.4.10 Výpočet fyzické adresy . . . 48
10.4.11 Kódování a dekódování dat za pomocí MFM . . . 48
10.4.12 Příjem a odesílaní dat po USB . . . 48
10.4.13 Komplexnost mikrokontroleru s jádrem ARM® . . . 49

Závěr **52**

Literatura **53**

Seznam symbolů a zkratek **57**

Seznam příloh **58**

A Schéma ovladače disketové mechaniky V1 **59**

B Schéma ovladače disketové mechaniky V2 **60**

C Obsah elektronické přílohy **61**

Seznam obrázků

2.1 Ukázka jednotlivých velikosti používaných disket 15
2.2 Popis vybraných částí diskety 16
2.3 Řazení dat na disketě 17
3.1 Počáteční stav registru CRC 20
3.2 Koncový stav registru CRC 21
4.1 Porovnání zakódovaní posloupnosti bitů za pomoci FM a MFM . . . 23
5.1 Uspořádání souborového sytému FAT12 24
6.1 Přehled používaných konektorů pro USB 2.0 a 3.x 27
6.2 Konektor používaný pro USB-C 27
7.1 Topologie USB 28
7.2 Tok dat na sběrnici USB 30
10.1 Vygenerovaný náhled výsledné desky Veze 1. 39
10.2 Vygenerovaný náhled výsledné desky Veze 2. 40
10.3 Vygenerovaný náhled sestavy 40
10.4 Blokové schéma programu 42

Seznam tabulek

1.1 Označení vývodů pro IBM/PC 13
1.2 Označení vývodů pro Shugart 14
2.1 Srovnání vybraných velikostí a typů disket 18
2.2 Rozložení polí v ID sektoru 19
2.3 Rozložení polí v Datovém sektoru 19
4.1 Kódování jednotlivých bytů za pomocí FM 22
4.2 Kódování jednotlivých bytů za pomocí MFM 22
5.1 Význam jednotlivých hodnot ve FAT tabulce 25
8.1 Ukázka nejpoužívanějších UFI příkazů 34
10.1 Doba příchodů jednotlivých pulzů převedená na bitovou posloupnost 46

Seznam výpisů

10.1 Funkce pro detekci připojení disketové mechaniky 42
10.2 Funkce pro nastavení čtecí/zapisovací hlavy na stou 0 43
10.3 Funkce pro detekci vložení diskety 44
10.4 Funkce pro start a zastavení motoru disketové jednotky 45
10.5 Nastavení čítače pro kontinuální vyčítaní 50
10.6 Funkce obsluhující přerušení 51
10.7 Funkce sloužící pro uložení časů 51

Úvod

Disketové mechaniky slouží ke čtení a zápisu dat na pružné magnetické disky. Často se také označují jako disketa. Byla to jedna z možností, jak uchovávat data i mimo pevný disk počítače a umožnila také tímto způsobem data snadno přenášet mezi počítači v době, kdy ještě nebyla tolik rozvinuta počítačová síť. Na začátku 70. let 20. století se začaly objevovat diskety o velikosti 8 in (inch) a maximální kapacitě 1,2 MB. Poté ve druhé polovině 70. let přichází pružné disky s velikostí 5,25 in a úložném prostoru až 1,2 MB. Jako poslední se objevují diskety o rozměru 3,5 in, ty přinášejí největší kapacitu až 2,88 MB. Diskety zažívají největší rozmach v 80. a 90. letech minulého století. Později obliba upadá kvůli zvětšující se kapacitě používaných softwarů a malé kapacitě disket. V dnešní době je pružný magnetický disk spíše kuriozitou a není obvyklé na něj narazit. Z tohoto důvodu se také již základní desky dnešních počítačů neosazují jejich řadičem a vzniká potřeba využít převodník, aby bylo možno číst a zapisovat data na toto medium.

Dnes je hojně rozšířené rozhraní USB, které se využívá k připojení vstupních periferií počítače. Z tohoto důvodu je nejvhodnější koncipovat převodník mezi rozhraním USB a rozhraním disketové mechaniky. Na trhu jsou dostupné přímo čipy, které jsou určené právě k tomuto účelu. U těchto čipů je ale problém, že často podporují jen nejběžněji používané diskety. Proto bude nejvhodnějším řešením navrhnout vlastní převodník, který bude podporovat i atypické typy disket.

Jak zvýše uvedeného vyplývá, je potřeba zajistit konverzi mezi USB sběrnicí a rozhraním disketové mechaniky. Proto je v první části práce popsáno rozhraní disketové mechaniky a podrobně se zaobírá rozložením dat na disketě. Poté je popsáno rozhraní a sběrnice USB. Z vyplývajících požadavků je vybrán vhodný mikrokontroler, který bude zajišťovat konverzi dat mezi jednotlivými rozhraními.

Na základě zvolených komponent je proveden návrh samotného kontroléru disketové mechaniky a popsán potřebný software pro řízení disketové mechaniky.

1 Disketová mechanika

Disketová mechanika je zařízení určené k přístupu na datové medium, kterým je v tomto případě pružný magnetický disk. V dnešní době se, ale toto medium téměř nepoužívá, a proto je disketová mechanika s klasickým 34 pinovým konektorem spíše vzácností a ani samotné počítače již nejsou standardně vybaveny řadičem disketové mechaniky.

Disketové mechaniky lze dělit dle samotných fyzických rozměrů media, které se do nich vkládá. Původní 8 in diskety se používali u velkých sálových počítačů, proto jsou zde zanedbány. Potom tedy lze dělit disketové mechaniky dle velikosti na dvě základní skupiny. Pro 5,25 in velké diskety, anebo pro pružné magnetické disky o velikosti 3,5 in. Podstatně zásadnějším rozdílem mezi jednotlivými disketovými mechanikami je přiřazená funkce jednotlivým pinům na konektoru, v obou případech je použit 34pinový konektor pro připojení, ale jednotlivé funkce pinů jsou rozdílné a jsou navzájem nekompatibilní. Tento fakt dost často zapříčiňoval nefunkčnost samotných mechanik a při jejich pořizovaní bylo důležité zvolit správné rozložení jednotlivých pinů. Dělení rozložení těchto vývodů bylo na IBM/PC a Shugart. Přiřazení jednotlivých funkcí pinům v konektoru je popsáni v následujících tabulkách pro IBM/PC Tab. 1.1 a pro Shugart Tab. 1.2. [1]

Tab. 1.1: Označení vývodů pro IBM/PC [1]

PIN	Název	Popis
2	/REDWC	Určení hustoty dat na mediu
4	nepřipojeno	
6	nepřipojeno	
8	/INDEX	Začátek otáčky
10	/MOTEA	Spuštění motoru pro jednotku A
12	/DRVSB	Výběr jednotky B
14	/DRVSA	Výběr jednotky A
16	/MOTEB	Spuštění motoru pro jednotku B
18	/DIR	Směr pohybu čtecí/zapisovací hlavy
20	/STEP	Posunutí čtecí / zapisovací hlavy
22	/WDATE	Zápisovaná data
24	/WGATE	Povolení zapisu na medium
26	/TRK00	Indikace stopy 0
28	/WPT	Indikace ochrany media proti zápisu
30	/RDATA	Čtená data
32	/SIDE1	Přepínání strany čtení na čtěcí hlavě
34	/DSKCHG	Vložený disk / Disk připravený

Neuvedené čísla pinů jsou připojena na zem

V další části se bude práce zabývat disketovými mechanikami s rozložením pinů na konektoru odpovídající IBM/PC, protože toto rozmístění funkcí pinů bylo častější u později používaných disketových mechanik.

Tab. 1.2: Označení vývodů pro Shugart [1]

PIN	Název	Popis
2	/DCD	Indikace vloženého disku
3	Zámek	
4	/INUSE	Řízení led diody
6	/DS3	Volba jednotky 3
8	/INDEX	Začátek stopy
10	/DS0	Volba jednotky 0
12	/DS1	Volba jednotky B
14	/DS2	Volba jednotky 2
16	/MTRON	Zapnutí motorů
18	/DIR	Směr pohybu čtecí/zapisovací hlavy
20	/STEP	Posunutí čtecí/zapisovací hlavy
22	/WDATE	Zapisovaná data
24	/WGATE	Povolení zápisu na medium
26	/TRK00	Indikace stopy 0
28	/WPT	Indikace ochrany media proti zápisu
30	/RDATA	Čtená data
32	/SIDE1	Přepínání strany čtení na čtěcí hlavě
34	/RDY	Disk připravený

Neuvedené čísla pinů jsou připojena na zem

1.1 Přístup na médium

Princip fungování disketové mechaniky je velmi jednoduchý. Přístup na médium je realizován za pomocí magnetické hlavy, která se pohybuje nad magnetickým diskem.

Všechny piny jsou aktivní v log. (logická hodnota úrovně signálu) 0, informace je důležitá, aby bylo možné mechaniku správně řídit a vyčítat z ní data. Dále je důležité vědět napěťové úrovně signálu v log. 0 dosahuje 0 V a v log. 1 je to 5 V.

Po vložení diskety do mechaniky přijde na pin s označeném /DSKCHG log. 0, ta indikuje, že byl do disketové mechaniky vložen pružný magnetický disk a je připraven k použití. V tento moment je možné aktivovat /DRVSA nebo /DRVSB, tím že na něm bude log. 0. Pin /DRVSA a /DRVSB má za úkol výběr disketové mechaniky, která bude ovládna. Aby mohl probíhat zápis nebo čtení na medium, hlava musí být ve výchozí pozici tedy nad stopou 00. Správnou polohu indikuje přítomností log. 0 na /TRK00. Pokud je log. 1 na pinu musí být hlava přesunuta nad stopu 00. Posun se provede nastavením log. 0 nebo log. 1 na pinu /DIR, logická hodnota na tomto pinu určuje, zda se bude hlava pohybovat směrem do středu disku, anebo od středu disku. Samotný pohyb hlavy je poté aktivován pinem /STEP, na který jsou přiváděny impulzy, počet impulzu určí o kolik stop se hlava posune. Ve stopě 00 je uložena FAT tabulka, která určuje adresy, na kterých se nacházejí soubory. Začátek každé stopy je indikován log. 0 na /INDEX. Čtená data jsou přítomna na pinu s označením /RDATA, naopak zapisovaná data se vysílají po /WDATE. [2]

2 Disketa

Disketa vznikla kvůli požadavku uchovávat data i mimo pevný disk počítače a do té doby používané magnetické pásky se vyznačovali velkou nespolehlivostí, kdy bylo potřeba záznam i několikrát opakovat, protože jediná chyba při zápisu znamenala poškození celých dat na pásce. Také přístup k datům byl zdlouhavý kvůli nutnosti přetáčet celé medium, pokud byla data uložená na jeho konci.

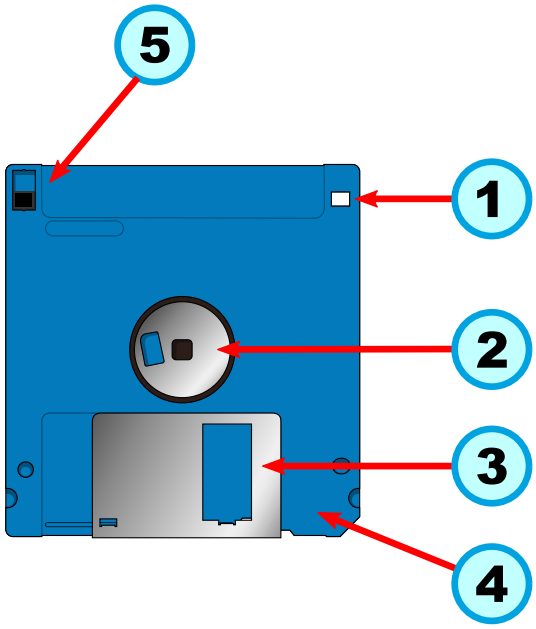
První diskety se začali používat pro komerční účely na konci 60. let minulého století. Rozměr diskety byl 8 in a pro svoje rozměry byla určena především pro sálové počítače. Postupem času se počítače rozšířily i do firem a domácností, které již nebyly tak velkých rozměrů a vešly se i na kancelářský stůl. Proto byla potřeba zavést menší rozměr pružného magnetického disku, se který by se lépe manipulovalo a nezabíral by tolik prostoru. V roce 1976 přichází společnost Shugart Associates s disketou o velikosti 5,25 palce. Rozměр se stává velmi populárním a postupně nahradil pružné magnetické disky o velikosti 8 placů, kdy bylo na menší ploše umožněno uchovávat stejný nebo i větší objem dat. IBM následně v roce 1986 přichází s disketami o velikosti 3,5 in, kdy je používá ve svých laptopech. Začátkem 80. let se již pružné magnetické disky o velikosti 5,25 palce téměř nepoužívají a jsou nahrazeny praktičtějšími 3,5 in. [4]

Obr. 2.1: Ukázka jednotlivých velikosti používaných disket Převzato z:[3]

2.1 Mechanické vlastnosti

Samotná disketa jako celek je tvořena vnějším ochranným obalem a pružným magnetickým diskem, který se nachází uvnitř ochranného obalu. Kryt má oba vnější roz-

měry stejné a jejich délka odpovídá rozměrům dle označení velikosti diskety. Udává se v inch, což je jednotka nepatřící do SI. Ochranný obal má za úkol zabránit poškození pružného magnetického disku, který je velmi citlivý na poškození magnetické vrstvy především poškrábáním. Proto se na vnitřní straně obalu nachází vložená folie, která předchází poškození pružného magnetického disku, když se otáčí uvnitř ochranného obalu. Diskety o rozměrech 8 a 5,25 inch mají v ochranném krytu otvor, přes který je možno přistupovat za pomocí čtecí/zapisovací hlavy přímo k mediu. Proto je nutno diskety uchovávat ještě například v papírovém pouzdře, které brání poškození odhalené části pružného magnetického disku. Nevýhodu se podařilo odstranit vnějšímu ochrannému krytu pro 3,5 in diskety. Má v sobě zabudovaný krycí mechanismus, který se automaticky při vkládaní do disketové mechaniky otevře a umožní, tak přístup čtecí/zapisovací hlavě přímo k samotnému médiu. Po vyjmutí dojde opět k jeho uzavření. To přispívá k oblíbenosti disket o rozměru 3,5in, protože dochází ke zvýšení uživatelského pohodlí. Ale za největším rozmachem stojí jejich poměrně malá velikost v porovnání s rozměry ostatních disket, kdy je zachována stále shodná kapacita. Dále vnější kryty disket mohou obsahovat zámek, který udává, pokud je přítomen, že se jedná o disketu s vysokou hustotou dat a je potřeba přizpůsobit způsob vyčítaní dat z media.[5]



Obr. 2.2: Popis vybraných částí diskety: 1-Zámek určující hustotu dat 2-Otvor pro unašeč motoru 3-Kryt čtecího otvoru 4-Ochranný obal diskety 5-Zámek proti zápisu dalších dat Převzato z:[10]

Samotný pružný magnetický disk je tvořen plastovým kotoučem, na který je nanesená tenká vrstva magnetického materiálu, u kterého je požadováno, aby měl

schopnost dlouho udržet získaný magnetický náboj. U disket 3,5 in je to:

- DD: 2 µm magnetický oxid železitý,
- HD: 1,2 µm kobaltem dotovaný oxid železa,
- ED: 3 µm ferit barnatý.

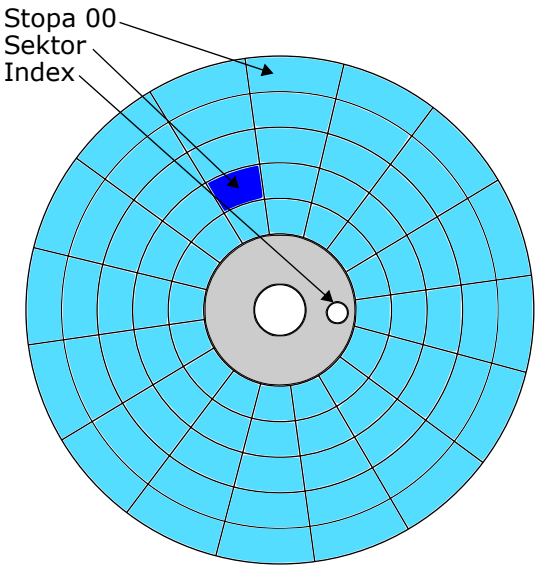
DD - dvojitá hustota – Double Density

HD - vyská hustota – High Density

ED - extra vyská hustota – Extra-high Density

Kotouč je opatřen ve svém středu otvorem pro unašeč motoru, který mediem otáčí. Dále se na pružném disku nachází otvor indikující začátek všech stop, otvor je umístěn kousek od středu a opisuje kruhovou trajektorii. Pomocí něj je také identifikována celá otáčka pružného magnetického disku. [5]

2.2 Uspořádaní dat



Obr. 2.3: Řazení dat na disketě [7]

Data jsou na pružném magnetickém disku uspořádaná v kruhových stopách. Jednotlivé stopy jsou následně členěny do sektorů. Jak je patrné z Obr. 2.3, všechny stopy mají začátek 1. sektoru shodný. Tuto pozici určuje otvor Index. Jednotlivé stopy jsou řazeny následovně, na vnějším okraji se nachází vždy stopa 00, v ní je uložen popis rozložení jednotlivých dat na disketě, které je udáváno FAT(5)(tabulka umístění souboru – File Allocation Table). V případě všech disket se jedná o FAT12. Celkový počet stop a sektorů na jedné straně diskety se liší. Dále se může na pružný magnetický disk zapisovat z jedné, anebo z obou stran. Všechny tyto parametry určuje užitá velikost disku a typ. Samotná data jsou na médium

poté kódována za pomocí MFM (4)(Modifikovaná frekvenční modulace – Modified frequency modulation), dále se také používala frekvenční modulace, ale protože se nevyužívá u disket IBM PC kompatibilních nebude dále rozebíraná. [7] V další část je zaměřena na diskety IBM PC kompatibilní. V Tab. 2.1 je stručný přehled běžně používaných druhů a typů disket. Přístupová rychlost k médiu se liší podle hustoty

Tab. 2.1: Srovnání vybraných velikostí a typů disket [9]

Rozměr diskety (in)	Hustota uložených dat	Byty/ Sektor	Sektory/ Stopa	Stopy/ Strana	Počet stran	Velikost	Otáčky za minutu	Kódování
5,25	DD	512	8	40	1	160 KB	300	MFM
5,25	DD	512	8	40	2	320 KB	300	MFM
5,25	DD	512	9	40	1	180 KB	300	MFM
5,25	DD	512	9	40	2	360 KB	300	MFM
5,25	HD	512	15	80	2	1200 KB	360	MFM
3,5	DD	512	8	80	1	320 KB	300	MFM
3,5	DD	512	9	80	1	360 KB	300	MFM
3,5	DD	512	8	80	2	640 KB	300	MFM
3,5	DD	512	9	80	2	720 KB	300	MFM
3,5	HD	512	18	80	2	1440 KB	300	MFM
3,5	HD	512	21	80	2	1680 KB	300	MFM
3,5	HD	512	21	82	2	1720 KB	300	MFM
3,5	ED	512	36	80	2	2880 KB	300	MFM

dat s jakou jsou na něm uložena. Výčet rychlostí pro přístup na medium:

- DD: 250 kbit/s,
- HD: 500 kbit/s,
- ED: 1 Mbit/s.

Podle přítomnosti identifikátoru (viz 2.1)je možno přesně určit s jakou přístupovou rychlostí se musí data zapisovat, anebo číst z pružného magnetického disku.

2.3 Skutečné uspořádání dat

Protože disketová jednotka a ani samotná disketa nedisponují možností samostatného výstupu pro hodinový signál. Je potřeba aby byla na disku umístěna i data, která nejsou uživatelská a slouží pro potřeby ovladače disketové mechaniky. Mezi taková data patří například synchronizační mezery nebo kontrolní součty uložených uživatelských dat. [11]

Kdy jsou mez jednotlivé segmenty dat vkládány mezery, mezera má hexadecimální hodnotu 4E. Tato hodnota je se o pakuje v různých počtech. Počet opakovaní je dán typem mezery tedy tím, v jaké části datové struktury se vyskytuje. Dále se na disku vyskytují synchronizační značky ty jsou hodnotou 0x00 a opakují se dvanáctkrát. Po této synchronizační značce následuje označení pro to data, ty jsou

značena 0xA1, značka se před užitečnými daty objeví třikrát po té následují již užitečná data. Buď to se jedná o ID sektor, ve kterém je uložené číslo stopy, strana, sektor a velikost sektoru s daty. Celý ID sektor je ukončen dvěma kontrolními součty a mezerou, kdy se opakuje hodnota 0x4E. Přesné rozložení jednotlivých polí v ID sektoru znázorňuje tabulka Tab.2.2. Kde pole IDAM je identifikátor ID segmentu a znak # zastupuje číselnou hodnotu. [11]

Tab. 2.2: Rozložení polí v ID sektoru [11]

ID Segment									
Úvod		ID Pole							Zakončení
12X 0x00	3X 0xA1	IDAM 0xFE	Stopa #	Strana #	Sektor #	Velikost	CRC 1	CRC 2	22X 0x4E

Za ID sektorem následuje Datový sektor. Ten je synchronizovaný úplně stejným způsobem jako ID sektor, tedy 12krát hodnota 0x00. Poté následuje značka pro začátek dat, 0xA1 opakující se třikrát. Po tomto začátku již následují samotná uživatelská data. Konec sektoru je určen pro dva kontrolní součty a mezeru kde se opakuje 0x4E. Tab. 2.3 ukazuje rozmístění polí v datovém segmentu. [11]

Tab. 2.3: Rozložení polí v Datovém sektoru [11]

DATA Segment						
Úvod		Datové pole				Zakončení
12X 0x00	3X 0xA1	DAM 0xFB / DDAM 0xF8	Uživatelská data od 128 po 1024 Bytů	CRC 1	CRC 2	Počet opakovaní 0x4E zaleží na velikosti uživatelských dat

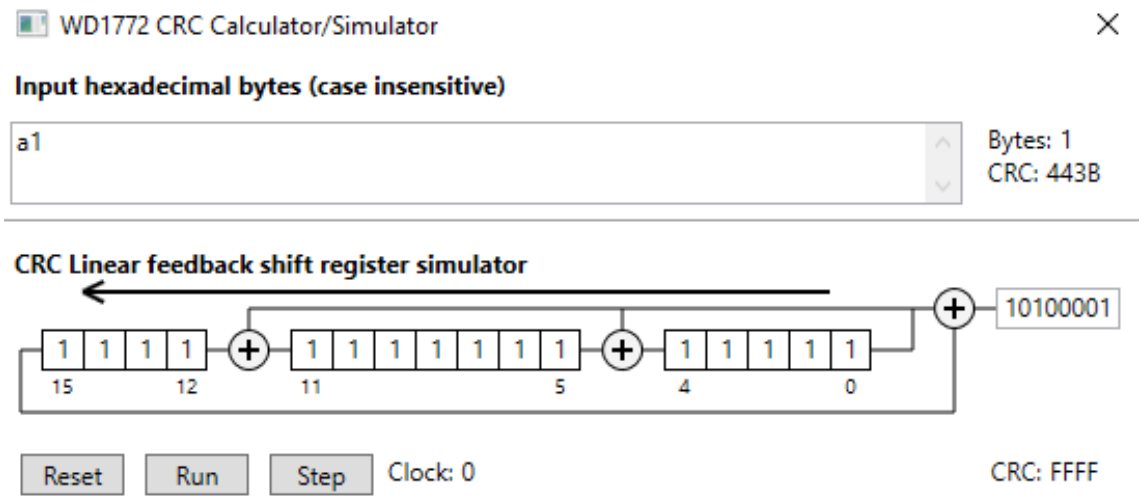
Velikost mezery mezi jednotlivými sektory je dána typem media a použitou velikostí sektoru uživatelských dat. [11]

3 Cyklický redundantní součet

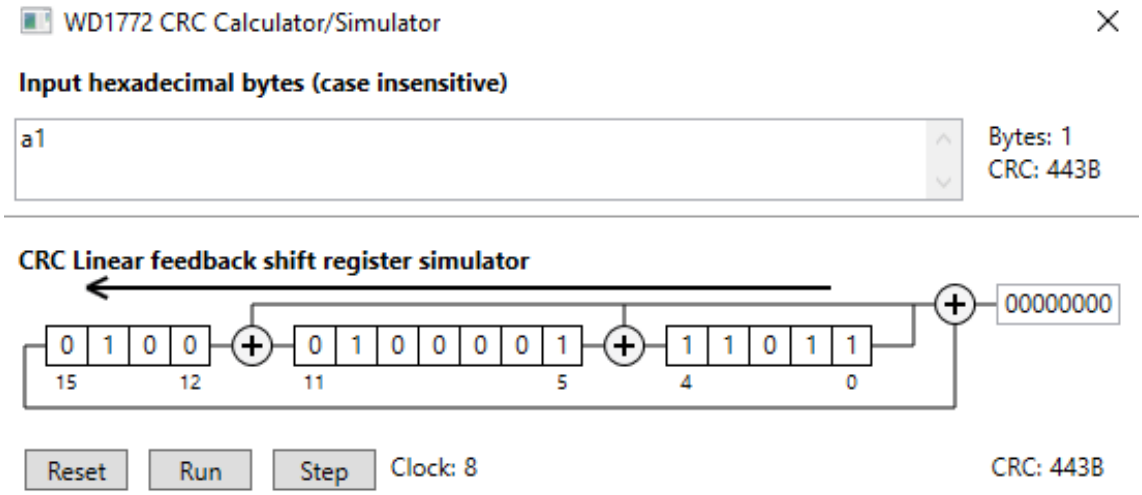
CRC (Cyklický redundantní součet – Cyclic redundancy check) je hashovací funkce sloužící k detekci chyb během přenosu dat. Je rozšířený díky své jednoduchost a nenáročnosti na výpočetní výkon. Lze je snadno i realizovat za pomocí hardwaru a procesor nemusí být vůbec jeho výpočtem zatěžovaný. CRC je ukládán a nebo posílán s daty. Jedná se o nadbytečná data, která jsou přidávaná k uživatelským datům. Samotné zabezpečení proti chybám spočívá v tom, že z dat která mají být zabezpečena proti chybám je spočítáno CRC a to je společně s daty předáno druhé stranně. Následně druhá strana spočítá podle shodného polynomu své CRC z přijatých dat. Pokud se vypočtené a přijaté CRC shodují přijatá data jsou prohlášená za platná. Vhodně zvolené CRC by mělo být schopno detekovat 2-bitovou chybu v zabezpečených datech. [12]

3.1 Výpočet

CRC je zbytek po dělení dvou polynomu. Kdy jeden z polynomů jsou zabezpečovaná data a druhý je takzvaný generující polynom. Generující polynom používaný je disketové mechaniky je 0x1021 a inicializační hodnota posuvného registru je 0xFFFF. Data jsou poté postupně nasouvána do generujícího polynomu, ve kterém po nasunutí posledního bitu dat zůstane výsledek CRC. Obr. 3.1 zachycuje počáteční stav CRC registru a Obr. 3.2 ukazuje výslednou hodnotu CRC registru, kdy byl nasunut poslední bit hodnoty. [13]



Obr. 3.1: Počáteční stav registru CRC Převzato z:[13]



Obr. 3.2: Koncový stav registru CRC Převzato z:[13]

4 MFM[1]

MFM je upravená FM(Frekvenční modulace – Frequency modulation). Pokud je FM používána na magnetickém médiu, tak se v takovém případě uvažuje změna magnetické polarity. V obou případech modulace je jeden bit vysílané zprávy kódován dvěma znaky. Oba kódy jsou navrženy, tak aby po překročení maximálního počtu shodných log. úrovní signálu došlo ke změně úrovně signálu, pomocí které může přijímač synchronizovat svoje vnitřní hodiny podle vysílače. U MFM k tomu dochází maximálně po třech shodných úrovních log. signálu. Pokud je zpráva kódována za pomocí FM, kombinace aktuálně vysílaných znaků pro daný bit nezávisí na předchozím bitu vysílané zprávy. V Tab. 4.1 jsou uvedeny kombinace znaků pro jednotlivé bity ve zprávě. I označuje změna mezi logickými rovněmi signálu – Inversion a N označuje zachování předcházející logické úrovně signálu – Non Inversion. Z Tab. 4.2 je patrné, že oproti kódování FM pří použít MFM výsledná kombinace

Bit	Znak
0	IN
1	II

Tab. 4.1: Kódování jednotlivých bytů za pomocí FM [14]

Bit	Znak	Předchozí Bit
0	IN	0
0	NN	1
1	NI	Nezáleží

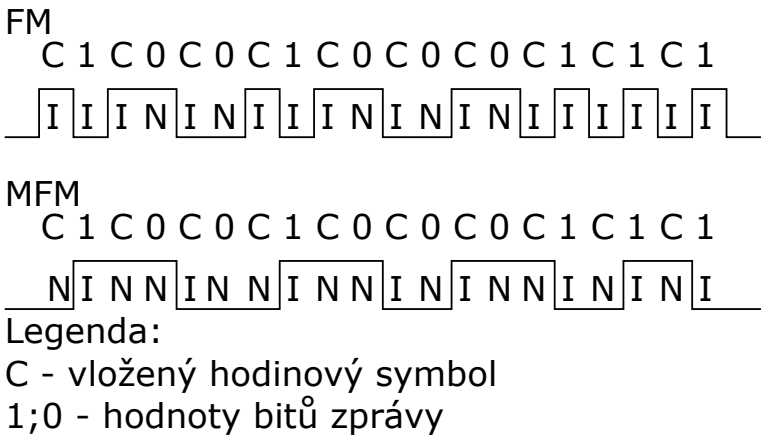
Tab. 4.2: Kódování jednotlivých bytů za pomocí MFM [14]

vystupováních znaků závisí na předcházejícím bitu zprávy. Tato závislost dovoluje zredukovat průměrnou změnu logické úrovně na polovinu oproti FM, znamená to že i magnetické změny budou poloviční. Jak vyplývá z Obr. 4.1 časový úsek změny u MFM je i v nejhorším případě, kdy po sobě následuje několik bitů s hodnotou 1, dvojnásobný v porovnání s FM. Potom je tedy možné zvýšit hustotu ukládaných dat na dvojnásobek vůči FM při zachování shodného časového intervalu magnetické změny. Proto je využívána pro kódování dat na pružném magnetickém disku. Pokud je MFM převedena z abstraktní roviny I a N do praktické a I je rovno log. 1 a N odpovídá log. 0. Je možno pro bit 0 zprávy generovat dvojici znaků jako x0, kde x je negovaná hodnota předchozího bytu ve zprávě, která je kódována. Pro bit 1 obsažený ve právě je vždy vygenerována dvojice znaků 01. [14]

V předchozí části je popsáno, jak jsou bity zprávy kódovány, z tohoto popisu je vyplývá, že vysílaná kódovaná zpráva na medium má dvojnásobnou délku ve srovnaní s původní délkou nekódované zprávy. Proto je důležité, aby původní zpráva bitů byla vyčítána dvojnásobnou rychlostí, než je kódovaná zpráva vysílána na medium, to neplatí v případě překódované zprávy. Například pokud je kódovaná zpráva vysílaná na medium rychlostí 500 kbit/s, tak původní nekódovaná zpráva musí být

[1]Modifikovaná frekvenční modulace

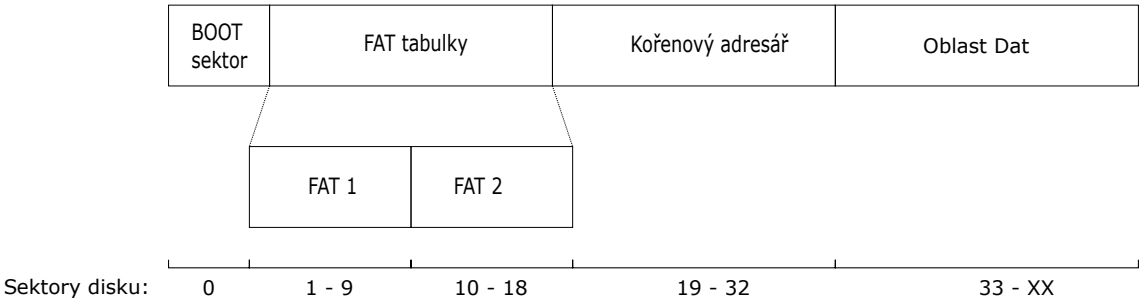
vyčítána rychlostí 1 Mbit/s. To klade vysoké nároky na mikrokontroler, který vysílá a kóduje data[14]



Obr. 4.1: Porovnání zakódovaní posloupnosti bitů za pomoci FM a MFM [14]

5 Souborový systém FAT[1]

Tabulka v sobě uchovává informace o souborech obsažených na disku a jejich pozici. Také se shodně označuje i souborový systém FAT, který je pro svoji jednoduchost používán na médiích o malé úložné kapacitě. Pokud dojde k poškození FAT tabulky stává se medium nedostupné a sytém není schopen přistupovat k souborům obsažených na něm. Proto se na mediu uchovává ještě záložní FAT, která je přesnou kopií původní a je použita v případě poškození původní FAT tabulky. V případě poškození obou FAT tabulek je jedinou možností provést „naformátovaní“ media, kdy dojde ke kompletnímu přemazávání začátku úložného media a je na něm vytvořen nový souborový systém i s prázdnými FAT tabulkami. Dnes existuje v několika verzích a je možné se s ním setkat například u USB flash disků, které používají FAT32, jako výchozí souborový systém. U disket se používá souborový sytém FAT12 viz Obr. 5.1. [15]



Obr. 5.1: Uspořádání souborového sytému FAT12 [15]

Boot sektor obsahuje informace o samotném mediu, například o použitém souborovém systému a FAT tabulky, potom velikost jednoho sektoru, počet sektoru na stopu, dále také identifikátor o jaké medium se jedná. Za pomocí těchto parametrů je následně možno spočítat výslednou kapacitu media. V kořenovém adresáři se nacházejí odkazy na soubory nebo další adresáře, které vedou ke konkrétnímu prvnímu indexu ve FAT tabulce. [15]

Princip fungování souborového systému FAT je následující. Postup přístupu je stejný pro soubor i adresář. V momentě kdy je požadován přístup k určitému souboru, dojde k jeho vyhledání v adresáři (například v kořenovém), jedním z parametrů je hodnota indexu ve FAT tabulce, která přísluší tomuto souboru, zároveň se jedná o logickou adresu, na které jsou uložena data. Ta se vyčtou z požadované adresy, poté dojde ke kontrole uložené hodnoty na příslušném indexu ve FAT tabulce. V případě, že se zde nenachází specifická hodnota (viz například Tab. 5.1) jedná se o další index v tabulace a zároveň o logickou adresu, na které pokračují data souboru. Celý

[1] tabulka umístění souboru – File Allocation Table

tento postu se opakuje, mimo vyhledávaní v adresáři, dokud není na indexu ve FAT tabulce uložena hodnota znamenající konec souboru. [15]

Přepočet mezi logickou adresou a fyzickou adresou sektoru je popsán ve vzorci 5.1, kde:

- A_f = Výsledná fyzická adresa sektoru,
- A_d = Fyzická adresa prvního datového sektoru,
- A_F = Logická adresa ve FAT tabulce.

$$A_f = A_d + A_F - 2 \tag{5.1}$$

Ve 5.1 je zahrnuta i korekce kde první dva indexy ve FAT tabulce nepoužívají. [15]

5.1 FAT12

Tato verze souborového systému se používá pro magnetické pružné disky. Kde délka jednoho sektoru na disku je 512 Byte. Jeho největší nevýhodou je, že používá pro indexování pozic souborů 12 bitů. Protože z tohoto důvodu se špatně přizpůsobuje na počítačové sběrnice, které jsou ve převažující většině násobkem 8. [15]

Boot sektor pro FAT12 má délku jednoho sektoru a fyzicky se nachází na sektoru 0. Za ním následuje sektor s FAT tabulkami, jedna zabírá 8 sektorů, kdy první je umístěna v prvním až devátém sektoru, druhá je volitelná a nemusí být přítomna, pokud se zde nachází je umístěna na pozici 10 až 18. Následujících 14 sektorů je rezervováno pro kořenový adresář, poté následuje již oblast pro data, která je adresována FAT tabulkou. [15]

V Tab. 5.1 jsou uvedeny specifické hodnoty pro indexy ve FAT tabulce pro souborový systém FAT12.

Tab. 5.1: Význam jednotlivých hodnot ve FAT tabulce [15]

Hodnota	Význam
0x00	Volný sektor
0xFF0 - 0xFF6	Rezervovaný sektor
0xFF7	Vadný sektr
0xFF8 - 0xFFF	Poslední sektr souboru
Ostatní hodnoty	Číslo následujícího sektoru souboru

6 USB

USB (univerzání seriová sběrnice – Universal Serial Bus)je sběrnicí pro připojování externích periferií k počítači. Ke vzniku vedla především snaha vytvořit univerzální rozhraní, kterým by bylo možno připojit externí zařízení a podporoval „plug and play“. Protože dříve měl každý druh externí periferie vlastní specifické rozhraní a bylo potřeba instalovat pro každé zařízení specifický ovladač. Pomocí USB je možno připojit běžně používané periferie počítače, jako je například klávesnice, myš, web kamera a další. USB tvoří standart, který specifikuje používané konektory a také komunikaci na sběrnici. Ucelenost standartu vedla k tomu, že se jedná o nejrozšířenější způsob připojení externích periferií k počítači a vytlačil všechny ostatní standarty do ústraní. Pro tento standart existuje několik verzí a také několik druhů konektorů. [16]

6.1 USB 1.0 a 1.1

Specifikace pro verzi 1.0 byla představena v roce 1994 a podporovala dvě rychlosti. LOW-Speed podporoval přenos o rychlosti 1,5 Mbit/s a druhá FULL-Speed s rychlostí 12 Mbit/s. O čtyři roky později přichází specifikace verze 1.1. [16]

6.2 USB 2.0

Specifikace verze byla vydána v roce 2000 a o rok později byla standardizovaná. Několik společností se podílelo na vývoji, který vedl k vyšší přenosové rychlosti značené jako Hi-Speed, umožňovala přenášet data rychlostí až 480 Mbit/s je to asi 40krát víc než standart 1.0. Je také kompatibilní s touto verzí.

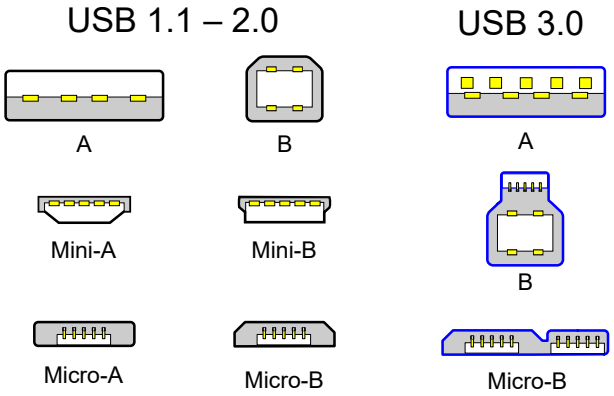
Je zde také podpora OTG (USB na přenosném zařízení – USB On-The-Go), ta umožní přenosnému zařízení jako je například mobilní telefon aby řídil komunikaci na USB sběrnici, poté se k němu dá připojit například přenosná paměť, anebo klávesnice. Ale pokud je přenosné zařízení připojeno k počítači, tak naopak počítač řídí komunikaci a přenosné zařízení je mu podřízeno. [16]

6.3 USB konektory

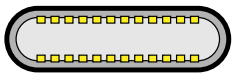
Pro standart USB 2.0 a 3.x se používá několik druhů konektorů, ty musí být přesně orientovány, aby je bylo možno do sebe zapojit (Obr. 6.1). Ale je zaručená zpětná kompatibilita a konektory pro standart 1.1 a 2.0 lze použít pro připojení zařízení

se standardem 3.x, bude ovšem zaručena jen základní funkčnost a rychlost přenosu není plně využita.

Naproti tomu je konektor USB-C (Obr. 6.2) symetrický a je možno jej zapojit oběma směry, tento nový design není kompatibilní s předchozími typy konektorů. I přesto se velmi rychle si získal oblibu díky své symetričnosti a začíná se dostávat do čím dál více zařízení. Podle počtu připojených pinů podporuje různé standardy USB. [17]



Obr. 6.1: Přehled používaných konektorů pro USB 2.0 a 3.x Převzato z:[18]



Obr. 6.2: Konektor používaný pro USB-C Převzato z:[19]

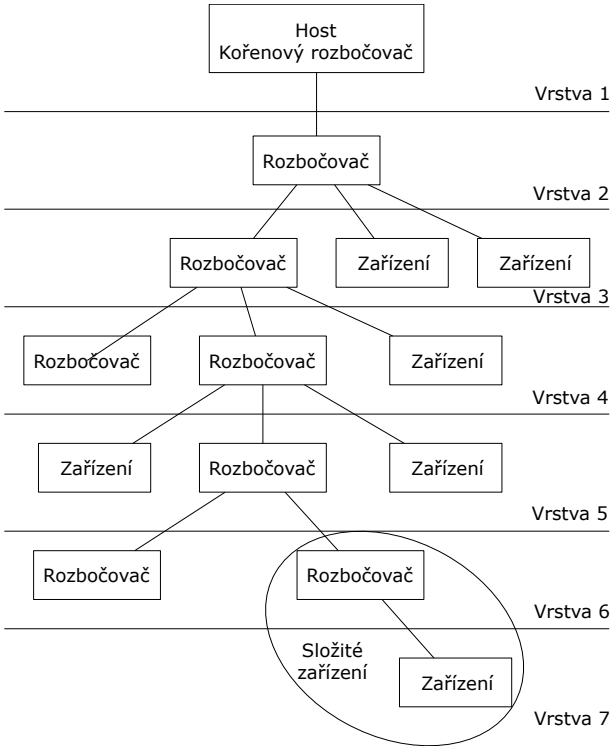
7 USB 2.0

V kapitole bude vysvětlený základní princip fungování standardu USB 2.0. Popis a verze USB jsou vysvětleny v kapitole 6. Během komunikace po USB v ní vystupují následující entity:

- Host - řídí celou komunikaci na sběrnici,
- Rozbočovač - dovoluje navýšit počet portů na sběrnici,
- Zařízení - připoje se na sběrnici a je ovládáno Hostem a poskytuje u svoje funkcionality.

7.1 Topologie sběrnice

Jedná se hvězdicovou topologii, kdy uprostřed se nachází vždy kořenový rozbočovač a na něj jsou připojeny další koncová zařízení anebo rozbočovače. Aby bylo možné lépe pochopit celou topologii je lepší ji převést na pyramidu. Ta může mít maximálně 7 vrstev, s ohledem na zpoždění na kabelech a aktivních prvcích, dále je povoleno použít maximálně 5 rozbočovačů. Na vrcholu pyramidy se vždy nachází kořenový rozbočovač, obdobně jako je ve středu hvězdicové topologie. [20]



Obr. 7.1: Topologie USB [20]

7.2 Komunikace po sběrnici

Komunikace probíhá v podobě paketů a je zahajována vždy Hostem. Vysílá paket zvaný Token. Jeho obsahem jsou informace o adrese, které přísluší zařízení, jenž bude vysílat nebo přijímat data, tedy obsahuje směr komunikace. Dále jsou přenášena data z nebo do Hosta. Komunikace končí vysláním paketu Handshake, je vyslán příjemcem dat. Pokud dochází ke komunikaci mezi Hostem a rozbočovačem jsou použity čtyři pakety, kde je navíc řízena rychlost přenosu. [20]

Přenášena data cestují takzvanou „rourou“, jsou dva druhy rour:

- Datové proudy - přenáší užitečná data, nemají pevně stanovenou strukturu.
- Zprávy - používají se pro přenos servisních informací, máji pevně stanovenou strukturu.

Roury vznikají při konfiguraci USB zařízení a jsou jí přiřazeny některé parametry: šířka přenosového pásma, typ přenosu, velikost vyrovnávací paměti a další. Jediná Defaultní roura vzniká v okamžiku připojí USB zařízení a je určena pro přenos zpráv, díky kterým je možno nakonfigurovat komunikaci mezi Hostem a zařízením. [20]

7.3 Typy datových přenosů

Pro komunikaci mezi Hostem a zařízením vznikají jednotlivé roury, s zařízením může vzniknout několik rour zároveň podle počtu koncovkových bodů podporovaných zařízením. Roury mohou být jednosměrné anebo obousměrné. [20]

Základní typy přenosů:

- Řídící procesy - slouží ke konfiguraci zařízení po připojení, vyjednání dalších parametrům přenosu a vytváření dalších rour,
- Hromadné přenosy - přenáší velký objem dat a vztahují se na ně minimální omezení,
- Přerušovací přenosy - slouží pro nepředvídatelné události a včasnému doručení dat,
- Nepřetržité přenosy - používají se na přenos dat v reálném čase, mají předem stanovenou šířku pásma a zpoždění.

Typickými daty, která vyžadují nepřetržitý proud dat, jsou například web kamery, kdy se přenáší obraz a zvuk. V takovém případě ani nepřetržitý proud dat není užitečný, pokud dochází k vysokému zpoždění. [20]

7.4 Komunikace mezi Hostem a zařízením

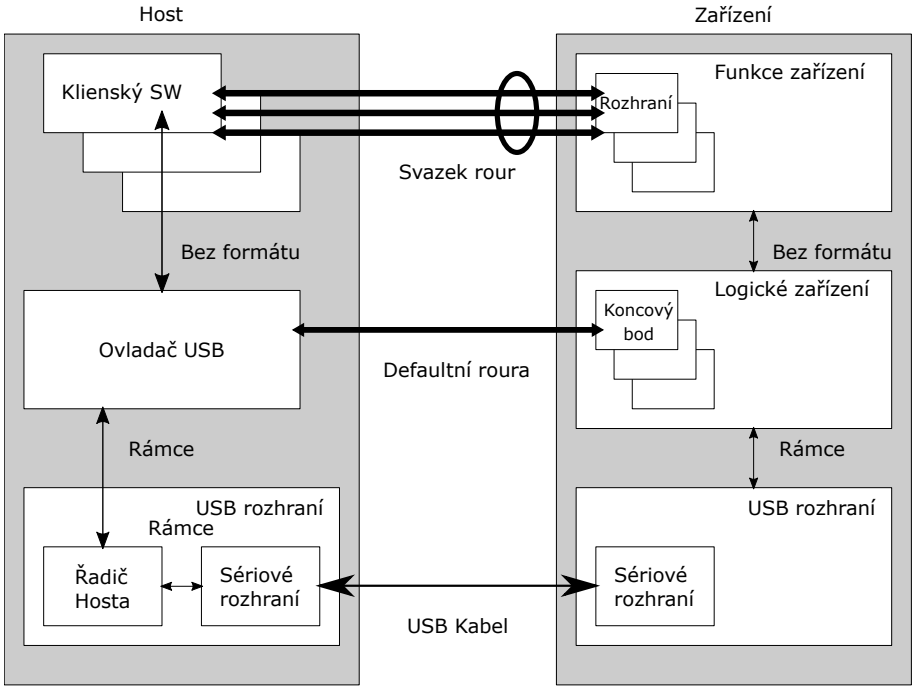
Samotná komunikace probíhá obdobně jako u protokolu internetu, tedy nižší vrstvy poskytují své služby vyšším a mezi shodnými vrstvami na obou zařízeních vznikají logické spoje. Jediná vrstva USB rozhraní má mezi sebou fyzický spoj. [20]

Pro samotnou implementaci USB jsou důležité následující části:

- USB zařízení - je připojeno ke konci USB kabelu a poskytuje své funkce Hostovi,
- Klientský software (Ovladač) - Běží na Hostovi a zajišťuje komunikaci se zařízením,
- USB software Hosta - obsluhuje USB sběrnici a je součástí jádra systému.

7.5 Tok dat

Obr. 7.2 podrobně zobrazuje přenos dat za pomocí USB sběrnice.



Obr. 7.2: Tok dat na sběrnici USB [21]

Každé logické zařízení představuje v systému několik koncových bodů a ty představují vlastní rozhraní, které reprezentuje jednu z funkcí zařízení. Ovladač řídí zařízení pomocí Defaultní roury, klientský software poté je připojen za pomoci svazku rour. Přenos dat je řízen řadičem Hosta. [21]

7.6 Koncový bod

V každém logickém zařízení se nachází několik koncových bodů. Ty mají výrobcem předem dané adresy a směr komunikace. Tedy je možné, aby se stejnou adresou existovaly dva koncové body, ale mají rozdílný směr. Všechny koncové body se připojují v neurčitém stavu a pro není možné s nimi komunikovat, dokud není zařízení registrováno. Jedinou výjimkou je koncový bod 0, přes ten probíhá provoní komunikace a registrace zařízení v systému Hosta. [21]

Koncový bod 0 musí být použit v každém USB zařízení. Bod předává informace o ostatních koncových bodech a jejich konfiguraci, také o stavu zařízení. [21]

High speed mód, pokud zařízení pracuje v tomto režimu musí splnit pro to minimální požadavky. Poté může zařízení využít všech zbylých 15 koncových bodů. V případě, že zařízení podporuje jen mód **Low speed** může kromě koncového bodu 0 využít ještě další dva. Pro oba režimy platí, že je možno využít počet koncových bodů ve směru vstupu i výstupu. [21]

7.7 Roury

Proudí skrz ně komunikace mezi softwarem Hosta a koncovými body zařízení. Roury pracují ve dvou režimech, které se navzájem vylučují:

- Proud - přenášená data nemají pevně danou strukturu
- Zpráva - její podoba je přesně určená specifikací USB

Jediná Defaultní roura vzniká v momentě připojení, probíhá přes ní konfigurace koncového zařízení a registrace v softwaru Hosta. Slouží také k otvírání dalších rour, přes které probíhá komunikace s dalšími koncovými body zařízení, Defaultní roura poté slouží ke kontrole komunikačního kanálu a používá se k vyjednání podmínek komunikace.[21]

Požadavek na přenos dat je realizován za pomocí IRP (paket s požadavkem na přenos dat vstup/výstup – I/O Request Packet), který vyšle software Hosta na konkrétní rouru. Host je informován o výsledku svého požadavku, ať již byl úspěšně či neúspěšně vyřízen. Požadavky může Host také stornovat.[21]

Pokud není roura v aktivním stavu, řadič Hosta ji neobsluhuje a nedochází k přidělování přenosového pásma pro ni.[21]

Pomocí IRP je možno požadovat přenos, který je větší než samotný USB paket, v takovém případě jsou data rozdělena do více paketů. Při tom může dojít k tomu, že poslední paket není plně naplněn daty a má kratší délku. V takovém případě se řadič Hosta může zachovat dvojím způsobem, kratší paket je vyhodnocen jako konec vysílání, anebo pokud není doplněn na předepsanou délku je považován za chybný přenos. Jak se má řadič hosta zachovat je specifikováno v IRP. [21]

Proudová roura pracuje s koncovým bodem zařízení a je možné říct, že slouží k přenosu užitečných dat mezi Hostem a zařízením. Jsou jednosměrné a předpokládají zpracování jedním klientem. [21]

Roura zpráv, tohoto typu je vždy Defaultní roura. Mají pevně danou strukturu, díky tomu je možno přesně určit řídící příkazy v zařízení. Je možné přes ně komunikovat v obou směrech, ale komunikaci musí zahájit Host. Také běžně dochází k vyslání dalšího IRP, dokud není vyřízen předchozí. [21]

7.8 Rámec a mikrorámec

Je zde zaveden rámec o délce 1 ms pro rychlost přenosu Low a Full Speed, pro High Speed je definován rámec o délce 125 µs. V jednom rámci může být přenášeno několik požadavků. Které požadavky mohou být přenášeny současně definuje typ přenosu. [21]

7.9 Řídicí přenosy

Umožňují přístup k jednotlivým částem zařízení. Přenášejí se skrze ně konfigurační, řídící a stavové zprávy mezi softwarem Hosta a zařízením. Přenos řídících zpráv zahajuje Host zprávou SETUP a zařízení na tento požadavek odpovídá status zprávou. V té se vrací vyřízení požadavku ať již úspěšné anebo neúspěšné. Řídící přenosy se posílají výhradně skrz rouru zpráv. Tvar řídících zprav je přesně daný, aby byl správně vyhodnocený všemi zařízeními. [21]

7.10 Nepřetržité přenosy

Jsou to přenosy, které generují pravidelný proud dat. Jde například o přenos videa z web kamery, převážně se jedná o data ze zařízení, která pracují v reálném čase. Je požadováno pro tyto přenosy aby:

- Měl přenos garantované maximální zpoždění přenášených dat,
- Rezervovanou určitou šířku pásma roury,
- Nedocházelo k opakovanému přenosu dat v případě chyby.

Tvar nepřetržitého přenosu není nijak specifikován. Přenáší je proudové roury. [22]

8 Třídy zařízení USB

Standart USB definuje několik tříd zařízení,třídy slouží k identifikaci připojených zařízení a popisují jeho základní funkce. USB zařízení je popsáno třídou zařízení, podtřídou zařízení a protokolem.

Kód třídy se může nacházet na dvou místech. Jedno je popis zařízení, druhé je popis rozhraní. Pro některé druhy zařízení se může vyskytovat kód třídy v popisu zařízení a zbylé informace jsou umístěny v popisu rozhraní, nebo jsou všechny informace umístěny v popisu rozhraní. U některých tříd zařízení je povoleno použít pro umístění informací o třídě, podtřídě a protokolu pouze popis zařízení. Kódu pro určení třídy zařízení je více, zde jsou uvede dva příklady. Kód 0x00 označuje, že všechny informace jsou všechny informace, 0x08 označuje třídu Mass Storage (Velké úložiště). [23]

8.1 Třída zařízení Mass Storage

Se především používá pro zařízení, které v jsou v sobě schopné uchovávat data a je možné potom k nim přistupovat. Takovými zařízeními můžou být například: externí pevné disky, přenosné optické mechaniky, mobilní telefony, čtečky paměťových karet a podobně.

Dále je v této třídě specifikováno několik podtřídy. Ty určují použitou sadu příkazů a tedy i samotný typ zařízení. Kód 0x06 je určený pro Flash disky a pevné disky a například je kód 0x04 je určená pro externí disketové mechaniky se sadou instrukcí UFI.

V poslední řadě je zde definovaný i protokol přenosu, který je zařízením podporován. [24]

8.2 UFI Příkazy

Protože specifikace UFI příkazů je poměrně rozsáhlá, a proto zde budou popsány podrobněji jen některé vybrané příkazy.

Tyto příkazy slouží pro ovládání samotné disketové mechaniky tedy funkce, které vykonává samotný kontrolér mechaniky. Kterými jsou například formátování samotného neformátovaného media, čtení dat, zápis dat a případně získání informací o disketě. Každý příkaz má poté přesně specifikovaný tvar zprávy, a určenou pozici polí ve zprávě. V Tab. 8.1 jsou vybrané nejpoužívanější příkazy. Jsou zde vynechány příkazy pro nastavení čtecí hlavy na stopu 0 anebo pro volbu typu media. Moderní disketové mechaniky tyto části provádí nezávisle na počítači. [25]

Tab. 8.1: Ukázka nejpoužívanějších UFI příkazů [25]

Příkaz	Popis	OP Kód
Formátovaní jednotky	Naformátování neformátovaného media	0x04
Dotaz	Získání inormací o disketě	0x12
Start / Stop	Požadavek na na načtení nebo vysunutí odjímtelného media	0x1B
Čti (10)	Přenos dat z media do hosta	0x28
Čti (12)	Přenos dat z media do hosta	0xA9
Kapacita	Vráti aktuální velikost media	0x25
Kapacita a naformátovaná kapacita	Vrátí aktuální kapacitu a kapacitu naformátovaného media	0x23
Test jednotky	Požadavek na potvrzení, že je jednotka připravená	0x00
Zápis (10)	Požadavek na zápis dat na médium	0x2a
Zapis (12)	Požadavek na zápis dat na médium	0xAA

9 Mikrokontroler

9.1 Požadavky

Mikrokontroler musí splnit několik požadavků, které vyplývají z předchozí teorie.

Z kapitoly 1 plyne, že je potřeba, aby obsahoval dostatečné množství vstupů a výstupů. Protože samotná disketová mechanika jich potřebuje 17 pro svoji plnou funkčnost. Dále je vhodné, pokud mikrokontroler podporuje napětí 5 V na vstupech, ale není to podmínkou.

V teorii o MFM (4) je popsán způsob kódování. Je v ní také uvedeno, že pokud jsem data na medium vysílána určitou rychlostí data musí být předkovávána dvojnásobnou rychlostí. To klade zvýšené nároky na výpočetní rychlost mikrokontroleru zejména během zápisu dat na pružný magnetický disk.

Dalším požadavkem na mikrokontroler je aby umožňoval komunikaci po USB, tento požadavek vzešel ze zadaní diplomové práce.

Po vyhodnocení všech požadavků na mikrokontroler, se dospělo k závěru, že bude nejvýhodnější použít mikrokontroler s moderní architekturou ARM® Cortex™-M0+. Ta přináší značné výhody oproti běžným 8 bitovým mikrokontrolerům, především je to její vyšší výpočetní výkon. [26]

9.2 ARM® Cortex™-M0+

Jedná se o 32 bitový mikrokontroler, který je založen na architektuře ARM® a instrukční sadě RISC (redukovaní instrukční sada – Reduced Instruction Set Computer). Jde o jednoho ze zástupců z ucelené řady procesorů Cortex™-M, ta je určená pro aplikace vyžadující mikrokontroler a má konkurovat běžným 8 bitovým mikrokontrolerům, kdy jejich cena je téměř srovnatelná, ale výpočetní výkon je o několik řádů vyšší. Mikrokontrolery Cortex™-M0 a Cortex™-M0+ nepodporují původní sadu instrukcí označovanou jako A32, ale podporují sady instrukcí Thumb (s výjimkou 3 instrukcí) a Thumb2 (částečně). Tyto sady instrukcí obsahují jen pár instrukcí, a proto je velmi jednoduché je programovat i za pomocí Assembleru. Cortex™-M0+ je vylepšením Cortex™-M0, kdy největší vylepšení spočívá ještě v nižší spotřebě na jeden MHz taktovací frekvence. Běžně se jedná o 48 MHz, 80 Mhz a 120 Mhz. Z toho vyplývá také to, že se jedná o mikrokontrolery s nejmenším výpočetním výkonem z celé řady Cortex™-M a jsou především určeny pro aplikace, které nemají neomezený přísun elektrické energie, jako například chytré hodinky. [27]

Politika firmy ARM Holding je taková, že jednotlivým výrobcům mikrokontrolerů prodává licence a ti poté mohou vyrábět mikrokontrolery s označením ARM®.

Protože se jedná o jádro mikrokontroleru, každý z výrobců jej vybavuje dalšími periferiemi, které se nachází v jednom pouzdře, může jednat o acsUSB, Ethernet a další. Proto je na trhu velký výběr různých kombinací specifikací mikrokontrolerů. [27]

9.3 Atmel SAM D

Dle požadavku na mikrokontroler (9.1) bude plně vyhovovat mikrokontroler od výrobce Atmel, konkrétně řada SAM D. Tato řada disponuje integrovaným řadičem USB s podporou přenosové rychlosti Full Speed. Další předností je možnost velkého počtu vstupů a výstupů podle zvoleného pouzdra. Tato řada s procesorem Cortex™-M0+ pracuje na frekvenci 48 MHz. To poskytne dostatečný výkon pro překódován MFM (4). Také je cenově dostupná. [28]

10 Výsledky studentské práce

Výstupem semestrální práce je návrh DPS (deska plošného spoje), který vychází z předchozí teoretické části.

10.1 Volba komponent

10.1.1 Verze 1

Jako hlavní ovládací prvek byl zvolen 64 pinový mikrokontroler ATSAMD21J18A-AU v pouzdře TQFP64. Disponuje 32 kB SRAM a 256 kB Flash paměti, to vytváří dostatečný paměťový prostor pro uložení samotného programu a proměnných během chodu programu. Maximální pracovní frekvence mikrokontroleru dosahuje až 48 MHz a je vybaven vnitřním oscilátorem, který za pomocí vnitřní násobičky může generovat až maximální pracovní frekvenci. Rozsah napájecího napětí se pohybuje do 1,62 po 3,63 V stejnosměrných.

V kapitole 1 je uvedeno, že pracovní napětí disketové mechaniky na vstupních a výstupních pinech může být v rozmezí 0 V až 5 V. Ale zvolený mikrokontroler umí pracovat pouze s napětím do 3,63 V, proto je nutné použít převodník napěťových úrovní. Konkrétně byl vybrán TXS0108EPWR, jedná se o obousměrný převodník vyráběný firmou Texas Instruments. Podporuje rozdílný směr převodu na jednotlivých kanálech, proto je možné využít každý kanál podle potřeby a nemusí se přepínat směr převodu pro všechny kanály.

Protože některé disketové mechaniky vyžadují i 12 V napájení. Bylo to rozhodnuto, že hodnota bude výchozí pro celý systém a pro komponenty, které vyžadují nižší napájecí napětí bude vytvořen převodník napětí. Systém obsahuje komponenty vyžadující napájení 5 V, pro ty je převodník řešen jako již hotový spínaný zdroj s vývody shodnými s klasickým lineárním zdrojem 7805. V tomto případě se jedná o spínaný zdroj AMSR1.5-7805-NZ, který je schopný poskytnout na výstupu až 1,5 A. To zajistí dostatečný výkon pro napájení disketové mechaniky i pro zbylé komponenty. Dále se zde vyskytují části systému, které vyžadují napájení 3,3 V, toto napětí bude dodáváno z lineárního zdroje AZ1117H-3.3TRE1, který je připojen na zdroj 5 V.

Dále deska bude obsahovat konektory pro připojení všech potřebných zařízení. Tedy konektor USB pro komunikaci s počítačem, šroubovací svorkovnici na připojení 12 V napájecího napětí, poté konektory pro samotnou disketovou mechaniku komunikační 34 pinový a standardní Molex konektor používaný v počítačích jako napájecí.

Deska je také osazena řadou indikačních diod aby bylo možno určit činnost, kterou právě vykonává mikrokontroler.

V poslední řadě se na desce nachází několik výstupních pinů, kterými je možno nastavit parametry používané diskety.

10.1.2 Verze 2

Při testovaní první verze ovladače disketové mechaniky se vyskytl problém s obousměrným převodníkem úrovní napětí s označením TXS0108EPWR. Kdy tento převodník nebyl schopný mikrokontroler v řídícím směru dostat na hodnotu 0 V, když byla připojena disketová mechanika a nebyl ji schopen ovládat. Důvodem je, že řídící vstupy na disketové mechanice, jsou zvedaný na 5 V za pomocí 1,2k Ω a bylo potřeba aby mikrokontroler byl schopen přes sebe přenést vetší proudy do země, ale jeho velký vnitřní odpor to nedovolil. Proto vznikla druhá verze desky pro ovladač disketové mechaniky. Všechny komponenty zůstávají shodné s první verzí, pouze je vyměněn obousměrný převodní za SN7406DR. Jedná se bipolární invertor s otevřeným kolektorem. Díky jeho vnitřní struktuře, kdy jeden kanál je tvořen několika tranzistory, poté je možné malými proudy ovládat několik násobně vetší proudy na výstupu. Dále má toto řešení výhodu v tom, že vstup kanálu je zakončen otevřeným kolektorem. To dovoluje připojit na vstup napětí 3,3V, tak i 5 V a na výstupu muže být téže připojeno napětí 3,3V nebo 5 V. Proto je možné tento invertor zároveň využít i jako převodní úrovní. Jedinou nevýhodou je, že převodníky jsou pouze šesti kanálové, proto muselo být upuštěno od myšlenky, že by byly současně řízeny obě disketové mechaniky. Volba, zda bude ovládaná mechanika připojená ke konektoru A nebo B se provádí mechanicky za pomocí přepínačů umístěných na desce. Kdy se je jedná o stejný kanál invertoru SN7406DR a je změněn pouze řízený pin na 34 pinovém konektoru pro disketovou mechaniku.

10.2 Návrh DPS

Pro návrh DPS (deska plošného spoje) byl použit opensource program KiCad. Program umožňuje tvorbu schémat a samotný návrh DPS, dále také umožňuje generovat 3D model desky. Model je možno následně využít při tvorbě mechanických částí zařízení a přesně nasimulovat mechanické vazby.

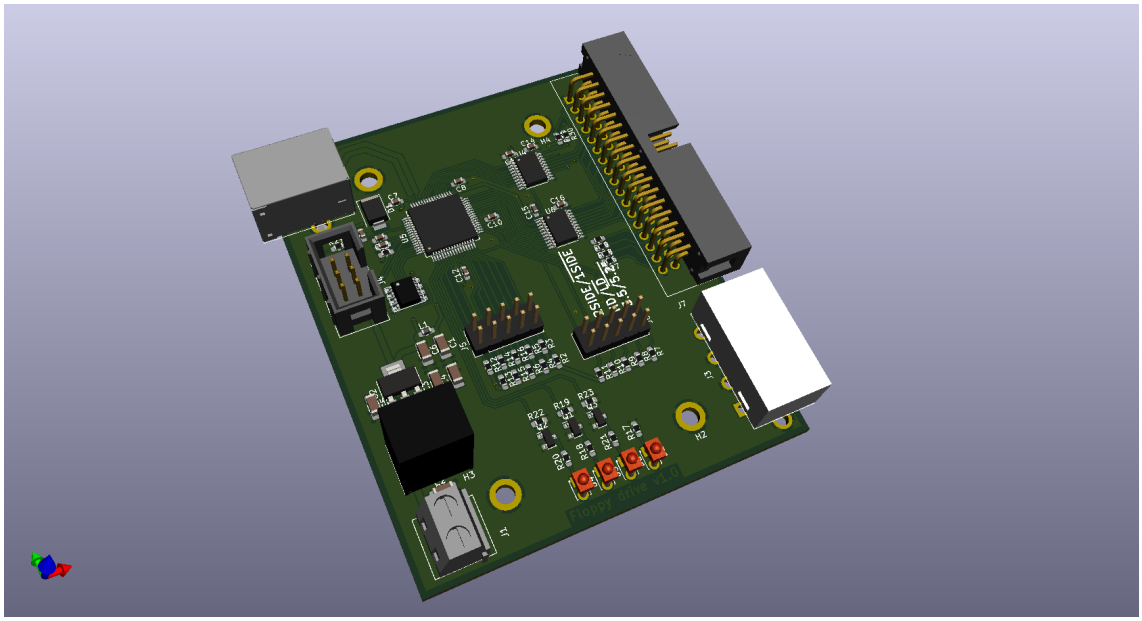
Při samotném návrhu DPS je nutné především dodržet předepsanou šířku jednotlivých spojů pro dané proudové zatížení, zejména pro cesty napájení. Dalším parametrem, který je nutné sledovat je odpor datových cest, kdy jednotlivé komunikační standardy mají předepsaný odpor a ten je nutné, pokud je to možné dodržet,

aby byla zajištěná plná funkčnost daného standartu. Například pro USB je to hodnota 90 Ω. S tím úzce souvisí i diferenční páry vodičů, u niž je důležité, aby obě cesty měli pokud možno shodnou délku.

10.2.1 Verze 1

Schéma zapojení ovladače disketové mechaniky je přiloženo v příloze této práce. Příloha A

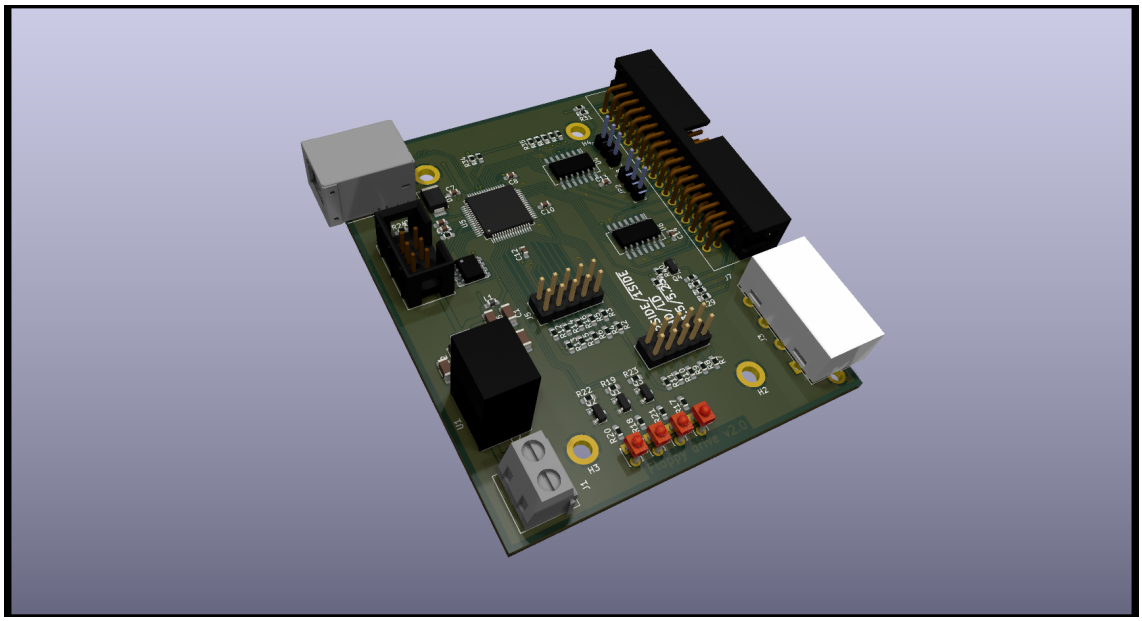
Výsledná deska má vnější rozměry 84x70 mm a obsahuje montážní otvory, které jsou vzdáleny v ose X od hrany desky 18 mm a v Y 10 mm. Náhled DPS je možné vidět na Obr. 10.1.



Obr. 10.1: Vygenerovaný náhled výsledné desky Veze 1.

10.2.2 Verze 2

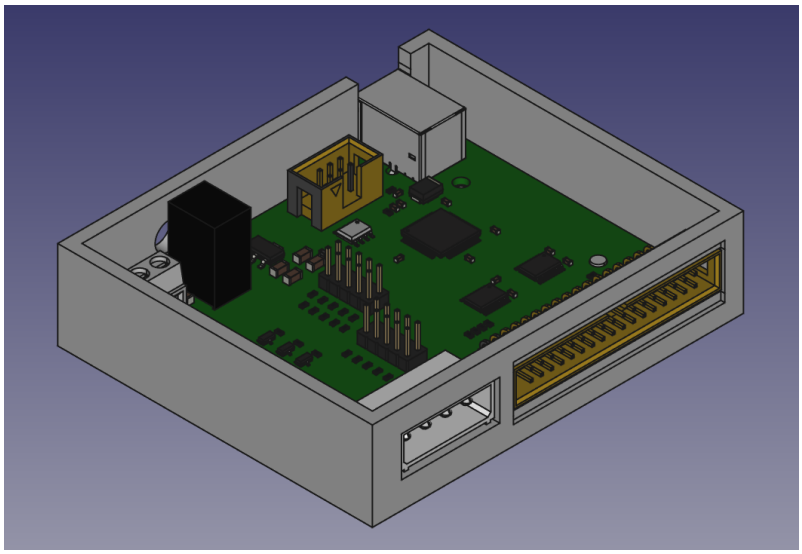
Druhá verze desky plošného spoje na navržena, tak aby byli zachovány podstatné konstrukční prvky a bylo ji možné přímo nahradit první verzi. Tedy jsou zachovány všechny rozměry, jako jsou rozměry desky a montážní otvory. Dále zůstalo shodné rozmístění konektorů. Rozložení součástek a konektorů je patrné z Obr. 10.2. Opravené schéma se nachází v příloze B.



Obr. 10.2: Vygenerovaný náhled výsledné desky Veze 2.

10.3 Ochranný kryt ovladače disketové mechaniky

V rámci práce také vznikl ochranný kryt pro samotnou DPS. Jeho účelem je zabránit náhodnému poškození komponent osazených na DPS, které by mohlo vzniknout v důsledku neopatrného zacházení nebo zkratování vývodů na desce. Kryt je vržen tak, aby jej bylo možné vytisknout na 3D tiskárně a DPS do něj usadit pomocí montážních otvorů. Na Obr. 10.3 je zobrazená sestava ochranného krytu a DPS.



Obr. 10.3: Vygenerovaný náhled sestavy

10.4 Software ovladače disketové mechaniky

Při návrhu softwaru je potřeba brát v potaz všechny možnosti, které mohou nastat a zohlednit je již při samotném návrhu hardwaru zařízení. Po zvážení všech variant se dospělo k závěru, že jsou možné dvě varianty řešení softwaru ovladače disketové mechaniky.

Software ovladače disketové mechaniky bude emulovat USB. To ho se docílí patřičným na nastavením USB rozhraní na mikrokontroleru, který se ohlásí počítači jako Mass Storage a to s podtřídou pro USB disketové mechaniky označené kódem 0X04. S instrukční sadou CBI (Comtorl, Bulk, Interrupt).

Vykonávání samotných příkazů bude probíhat na základě přerušení od USB, tedy jak čtení, tak i zápis dat, případně formátování media.

Na Obr. 10.4 je vývojový diagram programu.

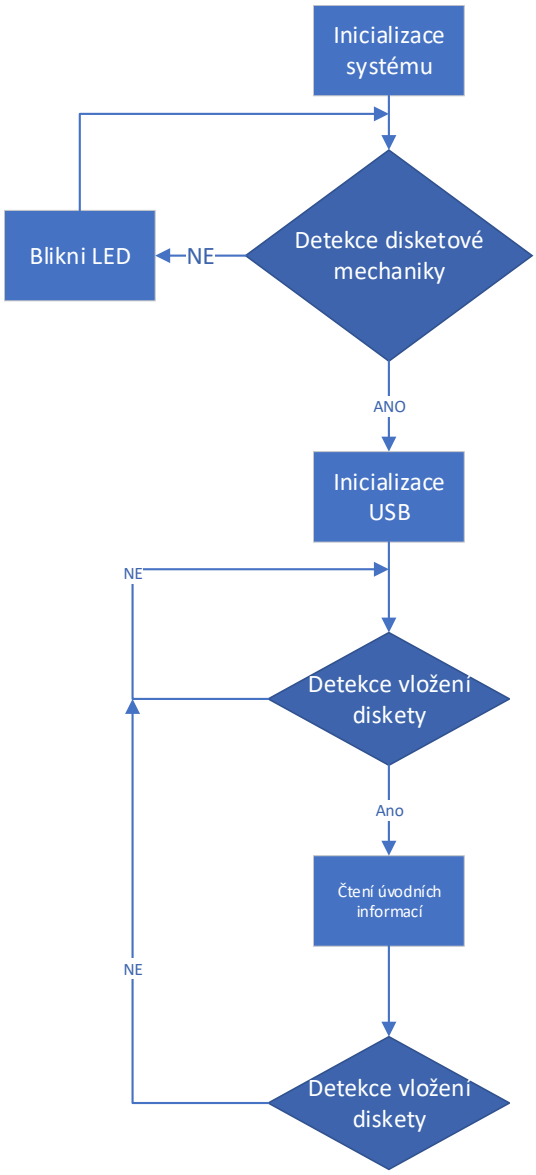
10.4.1 Inicializace systému

Jako první je potřeba inicializovat celý ovladač disketové mechaniky. Kdy v tomto kroku je zapotřebí nastavit frekvenci procesoru na 48 MHz. Inicializovat funkci pro zpoždění procesoru. Jako další musí být zjištěno, zda je připojená samotná disketová mechanika. Do doby než není detekována disketová mechanika nedojde ani k inicializaci USB rozhraní mikrokontroleru. Dochází pouze k blikání Ready LED každých 500ms. V momentě kdy je detekovaná dojde k jejímu nastavení do výchozí pozice, tedy čtecí/zapisovací hlava se posune nad stopu 0 (kapitola 10.4.3). Po provedení tohoto nastavení se zahájí detekce vložení diskety (10.4.4).

10.4.2 Detekce připojení disketové mechaniky

Detekce připojení disketové mechaniky probíhá následovně. Je aktivován pin **DRVSx** pro výběr disketové mechaniky, pokud pin aktivní je možné ovládat disketovou mechaniku, pokud tento pin není aktivní disketová mechanika ignoruje všechny ostatní řídící piny. Důležité je, aby tento pin byl aktivován před každým požadavkem, který má disketová mechanika vykonat a po provedení požadované sekvence příkazů by měl být zase deaktivován. Pokud je tedy disketová mechanika aktivována a připojena k ovladači disketové mechaniky, tak mechanika uvede pin **DSKCHG** do log. 1 maximálně jednu 1us po aktivaci disketové mechaniky. Pokud není připojena mechanika k ovladači tento výstup z ní zůstane v log. 0.

Výpis 10.1 ukazuje jak by mohla vypadat funkce pro detekovaní připojené disketové mechaniky.



Obr. 10.4: Blokové schéma programu

Výpis 10.1: Funkce pro detekci připojení disketové mechaniky

```
1  bool floppyConnect(){
2    floppySelect();
3    delay_us(1);
4    if (PORT->Group[DSKCHG_PORT].IN.reg & DSKCHG_PIN_MASK)
5    { floppyUnselect();
6      return true;
7    }
8    else
9    { floppyUnselect();
10     return false;
11   }
12 }
```

10.4.3 Nastavení čtecí/zapisovací hlavy na stopu 0

Po detekování disketové mechaniky je důležité, aby byla čtecí/zapisovací hlava zarovnána nad pozici stopy 0. V této pozici dochází k ignorování požadavku na posun čtecí/zapisovací hlavy směrem ven. To je velmi důležité protože, jak se uvádí v kapitole 10.4.4 pro detekci vložení diskety se využívá i pin, který ovládá posun čtecí/zapisovací hlavy.

Nalezení stopy 0 je provedeno následovně. Dojde k aktivaci disketové mechaniky. Poté se nastaví pin **DIR** pro výběr směru posunu čtecí/zapisovací hlavy do log. 0, tedy pro směr pohybu od středu diskety. Pokud by byl nastaven do log. 1 znamenalo by to směrem pohybu do středu diskety. Je nesmírně důležité, aby byl tento pin nastaven správně, protože směr do středu diskety není zakončen detektorem, který by indikoval, že již čtecí/zapisovací hlava dorazila na konec dráhy a mohlo by dojít k poškození krokového motoru, který pohybuje čtecí/zapisovací hlavou. Následně jsou přiváděny impulzy na pin **STEP** log. 1 trvá 1us. Poté smí přijít další impulz nejdříve za 3ms. Proces se opakuje ve smyčce, dokud mechanika neaktivuje výstup **TRK00**. Nyní je čtecí/zapisovací hlava nastavena na stopu 0. Po nastavení dojde vynulování hodnoty předchozí pozice čtecí/zapisovací hlavy. Ukázku kódu pro nastavení čtecí hlavy na stopu 0 je možné vidět na Výpise 10.2.

Výpis 10.2: Funkce pro nastavení čtecí/zapisovací hlavy na stou 0

```
1  bool floppyInitHead(){
2    PORT->Group[DIR_PORT].OUTCLR.reg = DIR_PIN_MASK;
3    while (!(PORT->Group[TRK00_PORT].IN.reg & TRK00_PIN_MASK))
4    {
5      PORT->Group[STEP_PORT].OUTSET.reg = STEP_PIN_MASK;
6      delay_us(1);
7      PORT->Group[STEP_PORT].OUTCLR.reg = STEP_PIN_MASK;
8      delay_ms(3);
9    }
10   actPoloha = 0;
11   return true;
12 }
```

10.4.4 Detekce vložení diskety

Na zjištění přítomnosti diskety je využíván čítač, aby nebyl procesor zbytečně blokován čekáním. Čítač generuje přerušení každých 500ms, kdy dochází ke kontrole, zda byla vložena disketa. Aby bylo možné zjistit přítomnost diskety v disketové mechanice, musí být první disketová mechanika aktivována. Následně musí být nastaven směr pohybu čtecí/zapisovací hlavy směrem od středu diskety, kdy by se měla hlava

nacházet na pozici stopy 0 a nemělo by tedy dojít k žádnému pohybu. Protože vložení diskety je zjišťováno následovně. Pin **DRVSx** aktivuje disketovou mechaniku. Ta uvede výstup **DSKCHG** do log. 1 a následně jsou na pin **STEP** převáděny impulzy z mikrokontroleru. Pokud je disketa vložena do mechaniky, dojde k zapsání na výstup **DSKCHG** hodnoty log. 0. V případě vytažení diskety z disketové mechaniky, zůstane po provedení testovaní na přítomnost diskety výstup **DSKCHG** v log. 1. Kód zobrazený ve Výpise 10.3 je pravidelně spouštěn každých 500ms při přerušení vytvořené časovačem.

Výpis 10.3: Funkce pro detekci vložení diskety

```
1  bool floppyDiskCharge(){
2    floppySelect();
3    PORT->Group[STEP_PORT].OUTSET.reg = STEP_PIN_MASK;
4    if (!(PORT->Group[DSKCHG_PORT].IN.reg & DSKCHG_PIN_MASK))
5    { floppyUnselect();
6      return true;
7    }
8    else
9    {
10     floppyUnselect();
11     return false;
12   }
13 }
```

10.4.5 Zapnutí motoru pro rotaci diskety

Proto aby bylo možné vyčítat data z diskety, tak je nutné, aby magnetický pružný disk uvnitř rotoval. Toho je docíleno za pomocí motoru. Ten ovládá pin **MOTEx**, pokud je v log. 1, dochází k rotaci media. To, aby bylo možné získat korektní data je nutné dosáhnout ustálené rychlosti media, té dosáhne disketa maximálně za 800ms a zastaví maximálně za 300ms. Dále je nutné, aby tato doba byla počítána až od momentu kdy se medium určíte otáčí. K tomu nedojde hned po aktivaci motoru, ale až v momentě zapadnutí unašeče motoru do diskety. K indikaci toho, že se disketa otáčí je využit výstup **INDEX**, výstup slouží pro indikaci začátku stopy, ale je možné jej využít i pro indikaci, že se medium skutečen otáčí. Při každém průchodu stopy 0 je na výstupu **INDEX** pulz log. 1. Ukázku funkce pro roztočení a zastavení motoru pro rotaci diskety je možné vidět na výpise 10.4

Výpis 10.4: Funkce pro start a zastavení motoru disketové jednotky

```
1  bool floppyMotorON(bool ON){
2    if (ON)
3    {
4      PORT->Group[MOTEB_PORT].OUTSET.reg= MOTEB_PIN_MASK;
5      while(!(PORT->Group[INDEX_PORT].IN.reg & INDEX_PIN_MASK));
6      delay_ms(800);
7      return true;
8    }
9    else
10   { delay_ms(300);
11     PORT->Group[MOTEB_PORT].OUTCLR.reg = MOTEB_PIN_MASK;
12     delay_ms(3);
13     return false;
14   }
15 }
```

10.4.6 Nastavení čtecí hlavy na konkrétní stopu

Během čtení dat muže dojít k tomu, že jednotlivé části souboru nebudou uloženy v jedné stopě, případně může být zapisováno více souborů a bude zapotřebí aby se čtecí/zapisovací hlava přesouvala mezi jednotlivými stopami, kde se nachází části souborů. V souborovém systému FAT jsou uloženy pouze logické adresy, proto je potřeba vypočíst fyzickou adresu, která je složena z čísla požadované stopy a sektoru ve stopě. Konkrétní postup výpočtu je popsán kapitole 10.4.10. Vstupem do této funkce bude tedy číslo požadované stopy, na kterou má být nastavena čtecí/-zapisovací hlava. Dále je vhodné, aby se ukládala i aktuální pozice čtecí/zapisovací hlavy. Posun na další stopu bude počítaný jako rozdíl požadované a aktuální stopy, aby se nemusela hlava pří každém požadavku na posun, vracet do výchozí pozice a z ní teprve počítat požadovaný posun.

Samotný posun čtecí/zapisovací hlavy, poté co je aktivována disketová mechanika, probíhá, tak že nejdříve se nastaví směr pohybu za pomocí pinu **DIR**. Dále přichází impulzy na vstup **STEP**. Hodnota pro nastavení požadovaného směru, délka trvání pulzu a frekvence příchodů jednotlivých pulzů se nachází v kapitole 10.4.3. Pokud dojde ke změně směru další impulzy pro posun čtecí/zapisovací hlavy mohou přijít nejdříve po 18ms. Proto je ukládán předchozí směr posunu. Po skončení posuvu se spouští čítač, který odpočítává 18ms, aby nebyl zbytečně vytěžován procesor a mohl pokračovat případně dál v programu. Po příchodu přerušení se nastaví příznak, že již uplynul časový interval. V případě, že se se měr posunu sho-

duje s předchozím je toto zpoždění ignorováno. Pokud se směry pohybu liší program kontroluje, zda je nastaven příznakový bit a uplynulo 18ms, jinak zůstane čekat na uplynutí této doby, aby mohl začít pohybovat čtecí/zapisovací hlavou.

10.4.7 Volba strany diskety

Před samotným zahájením vyčítáním dat z diskety nebo zápisem je nutné ještě nastavit i stranu ze které budou data vyčítaná. To se provází za pomocí vstupu **SIDE1**, kdy pokud je tento vstup v log. 0 je vyčítána nebo zapisována strana 0 diskety, tedy strana, na které se nachází stopa a sektor 0. Pokud je nastavena log. 1 jsou data čteny, anebo zapisovány na stranu 1.

10.4.8 Vyčítaní dat z diskety

Vyčítání dat z disketové mechaniky se provádí za pomocí výstupů **INDEX** a **RDATA**. Jak již bylo uvedeno v 10.4.5 pin **INDEX** indikuje začátek stopy pulzem v log. 1. Pulz přichází každých 200ms pro 3,5 in disketu. Na pin **RDATA** přichází také impulzy v různých časových intervalech. Převod doby mezi jednotlivými impulzy a bitovou posloupností je v Tab. 10.1. Původně dle všech dostupných informací z popisu vyplývalo, že impulz bude označovat pouze změnu magnetické úrovně a bude potřeba toto ještě převést na data MFM. Nakonec se prokázalo, že se jedná přímo o data kódu MFM. Čtení z diskety je proces velmi citlivý na nepřesnost v měření času mezi jednotlivými impulzy. Snadno může dojít k chybě čtení. Protože jak je popsáno v kapitole 2.3, disketová mechanika nedisponuje výstupem pro hodinový signál. Výrobce v návodu k disketové mechanice uvádí, že chyba doby mezi jednotlivými pulzy by měla být maximálně +-0,5%. [29]

Tab. 10.1: Doba příchodů jednotlivých pulzů převedená na bitovou posloupnost [29]

Čas mezi pulzy [us]	Bity
2	01
3	001
4	0001

Jak již bylo napsáno na začátku této kapitoly jedná se operaci kritickou na přesnost času. Jsou dvě možnosti jak realizovat tuto část. Jedna z možností je bez použití přerušení čekat ve smyčce na příchod spouštěcího impulzu ze vstupu **INDEX** poté vynulovat čítač a začít čítat impulzy z oscilátoru za pomocí čítače a čekat ve smyčce na příchod impulzu na pinu **RDATA**. Po jeho detekování uložit aktuální hodnotu čítače, vynulovat jej a zpracovat uloženou hodnotu. Volitelně může být přidána čekací smyčka na to, dokud impulz neklesne opět k log. 0, poté bude jasné, že není

detekován ten samí impulz dvakrát. Proces vyčítaní dat z diskety poté pokračuje ve smyčce a čeká se na příchody dalších impulzu na vstup **RDATA**. Počet opakování tohoto procesu je dán délkou dat, které mají být vyčteny.

Druhou možností jak realizovat vyčítaní dat z diskety je využití externího přerušení na vstupu **RDATA**. Průběh vyčítaní poté bude následující. Opět se bude čekat na začátek stopo dat. Tedy na impulz na pinu **INDEX**, po jeho detekci dojde se spuštění čítače a zároveň je povoleno i přerušení na pinu **RDATA**. Po příchodu přerušení je uložena hodnota čítače čítač je vynulován a program se vrací do hlavní smyčky, kde čekal ve smyčce až bude indikován stavovým bitem příchod přerušení. Poté zpracuje data.

Protože výstupem z mechaniky jsou různě dlouhé bitové posloupnosti, jak je patrné z Tab. 10.1. A zpracování takto různě dlouhých dat by bylo časově náročné, protože může dojít k tomu, že detekovaná posloupnost může přesáhnout délku 16 bit, do které je ukládána, protože aktuální kód MFM záleží i na předchozí hodnotě datového bitu. Je výhodnější uložit posloupnost bitů ve vlastním kódu a ten posléze převést na kód MFM.

10.4.9 Zápis dat na disketu

Před samotným zápisem na disketu je důležité ověřit, zda je zápis povolen. Ověření probíhá kontrolou vstupu **WTP**, pokud je vyčtena hodnota log. 1, tak je disketa chráněna proto zápisu.

Pokud není disketa chráněna proti zápisu, tak pro samotný zápis dat na disketu slouží vstupy **WGATE** a **WDATE**. Případně ještě může být volitelně použit i výstup **RDATA**, na kterém je možné ověřit zda vyslaná data dorazila správně do disketové mechaniky. Aby bylo možné vůbec data na disketu zapsat je potřeba povolit zápis dat, povolení zápisu se nastaví pinem **WGATE**, který musí být v log. 1. Dokud tento pin je v log. 0, jsou všechny data přicházející po **WDATE** ignorována a nezapisují se na disketu. Samotná data tedy přichází na vstupu **WDATE** a jak již bylo několikrát zmíněno čtení a zápis dat jsou operace velmi náchylné na přesnost časových intervalů. Proto musí zápis dat na disketu probíhat pod přerušením časovače, kdy hodnoty pro časovač jsou připraveny z kódu MFM, v době kdy probíhá odpočet předchozí nastaveného intervalu. Po uplynutí předem nastavené doby dojde k nastavení log. 1 na pinu **WDATE** a spuštění odpočtu doby trvání tohoto impulzu. Doba trvání je mezi 0,15 až 1,1 us. Na reálné disketě byla naměřena šířka impulzu blížící se 1us. Tento proces se opakuje podle požadované délky zapisovaných dat.

10.4.10 Výpočet fyzické adresy

Požadovaná adresa, ze které má být čteno anebo na ní zapisováno, přichází v takzvaném Logický Blok Adres (LBA). Z LBA je poté možné dopočítat požadovaný sektor a stopu, se kterou se mají následně provádět operace. Podle rovnice 10.1 [25] je možné dopočítat požadovaný sektor. Dále je potřeba dopočítat stopu, na kterou se má čtecí/zapisovací hlava nastavit. Ta se vypočítá podle rovnice 10.2 [25].

$$Sektor = (LBA \; mod \; SekStop) + 1 \tag{10.1}$$

$$Stopa = (LBA \div SekStop) \div HlaStop \tag{10.2}$$

kde:

LBA - Logický Blok Adres

SekStop - Počet sektorů na stopu (Počátek v 0)

HlaStop - Počet hlaviček na stopu

10.4.11 Kódování a dekódování dat za pomocí MFM

Data jsou na disketu ukládána za pomocí MFM. Princip kódování popisuje kapitola 4. A jak již bylo popsáno v 10.4.8. výstup dat odpovídá přímo kódu MFM, tedy je možné prohlásit, že každá změna magnetického pole, která je označena v Tab. 4.2 odpovídá log. 1. a pokud neodchází k inverzi jedná se o log. 0. Nyní když jsou převedeny hodnoty inverze a ne inverze do bitové podoby je možné kódovat a dekódovat data. Kódované slovo oproti původnímu má dvojnásobnou velikost.

10.4.12 Příjem a odesílaní dat po USB

Velikost zprávy po USB je nastavena 512B. Poté jsou sou data přijata řadič USB nastaví stavový bit o tom, že přijal data. Mikrokontroler poté vyhodnotí příkaz pro ovládání disketové mechaniky. Pokud se jedná například o požadavek o čtení, tak je dále rozpoznána adresa požadovaného bloku, který má být přečten. Po vyčtení a dekódování požadovaného bloku jsou uživatelská data předána zpět USB řadiči v mikrokontroleru, Od tohoto momentu není zatěžován procesor mikrokontroleru, řadič tento proces provede nezávisle, ten oznámí že má data k přenosu, jak mile jej Host vyzve tak mu předá připravená data. Při požadavku na zápis dat, je proces obdobný. Řadič USB ohlásí příjetí dat. Mikrokontroler vyhodnotí, že se jedná o požadavek na čtení. A nastaví patřiční signalizační bit, že řadič může opět přijímat data. Přijatá data jsou zakódována dle požadavku diskety a jsou na ní zapsána.

10.4.13 Komplexnost mikrokontroleru s jádrem ARM®

Při zprovozňování kontroleru disketové mechaniky se projevila složitost vnitřní struktury mikrokontroleru ATSAMD21j18A s jádrem ARM® a neexistence velkého množství návodů a příkladů pro tento mikokrontroler. Především na přesnost času náročných aplikacích, tedy hlavně u čtení a zápisu dat, která je popsána v kapitole 10.4.8. Kdy čítač je externí periferií, a tedy procesor má přístup pouze ke stínovým registrům a ty musí být aktualizovány periferií. Tato aktualizace nejčastěji probíhá při události vyvolané čítačem, tedy například při přetečení anebo při shodě s porovnávanou hodnotou.

Původní myšlenka byla, že program po příchodu impulzu ze vstupu **INDEX** spustí vynulovaný čítač a bude čekat ve smyčce na příchod pulzu na vstupu **RDATA**, po jeho zachycení uloží aktuální hodnotu čítače a ten poté vynuluje a hodnota bude zpracována. Počet opakování detekce pinu **RDATA** bude záviset na délce čtených dat. Čítač, který odměřuje dobu mezi jednotlivými pulzy, byl nastaven na vstupní frekvenci 24 MHz. Tedy jeho přesnost by měla být dostatečná. Jenže vyčítané časy se od uváděných časů v Tab. 10.1 lišily v rozmezí od 0,2 až 0,6 us. Doba mezi jednotlivými pulzy byla ověřena i měřením na osciloskopu a z tohoto měření vyplynulo, že doba mezi jednotlivými pulzy je 2,1 us v nejhorším případě. V tomto momentě se plně projevila komplikovanost mikrokontroleru s jádrem ARM®, kdy čítač je externí periferie, a pokud se provádí operace čtení a zápisu, musí být následně synchronizována hodnota stínového registru a registru čítače. Tato synchronizace přichází po různě dlouhé době, protože závisí na tom, kdy bude vyvolána událost čítačem. Po podrobném prozkoumání datasheetu k mikrokontroleru se podařilo odstranit požadavek na synchronizaci registrů až po vyvolání události čítačem pro operaci čtení, ale k synchronizaci dochází s každým hodinovým impulzem vstupujícím do čítače, a tak se sníží synchronizační zpoždění u operace čtení, ale není možné nulovat čítač, protože tato operace vyžaduje synchronizaci registrů. Tento problém se odstranil, takže doba mezi pulzy je počítaná z aktuální hodnoty čítače a z předešlé, rozdíl těchto hodnot vrací dobu mezi příchody pulzu. Toto řešení snížilo chybu ve vyčítané době mezi jednotlivými pulzy o 0,2 až 0,3 us. Zbylé zpoždění je připisováno překladu mezi kódem jazyku C a strojovým kódem. Po této úpravě se podařilo v pár případech vyčíst validní hodnoty bytů podle jejich pozice, která je popsána v kapitole 2.3.

Druhou myšlenkou bylo použít přerušení na vstupu **RDATA**, kdy po zachycení začátku datové stopy na pinu **INDEX** se uloží aktuální hodnota čítače, který byl vynulován a spuštěn při požadavku na čtení z disketové mechaniky, tedy měl dostatečný čas na synchronizaci a povolí se přerušení na **RDATA**. Tato cesta se zdála v první chvíli jako správná, protože vyčítané hodnoty měly pouze velmi malý rozptyl

od hodnoty uváděné v Tab. 10.1, tedy konkrétně v nejhorších případech se jednalo o 0,2 us. Tato hodnota byla přijatelná, protože bylo možné přesně oddělit jednotlivé doby mezi impulzy. Problém nastal až v momentě, kdy se měla tato získaná hodnota zpracovat v hlavním programu. Hlavní program čekal ve smyčce až bude nastaven stavový bit, že došlo k uložení hodnoty čítače v přerušení. Poté jednotlivé hodnoty mezi pulzy začaly odpovídat součtu dvou až tří dob mezi impulzy. Tento problém způsobuje kompilátor a vývojové prostředí, které pokud nahraje program do mikrokontroleru v režimu debugg a ten je i spuštěn v režimu ladění, tak dochází k zastavování mikrokontroleru a ten poté nestíhal obsluhovat hlavní program. Z tohoto důvodu není možné přesně odladit program v režimu debugg. Navíc pokud není čítač korektně nastaven, zastavuje chod hlavního programu. Zastavení odpovídá 6 taktům hodinového signálu pro čítač.

Před samotným ověřením problému bylo odstraněno nekorektní nastavení čítače pro kontinuální vyčítaní. Potřebná konfigurace pro kontinuální vyčítaní je uvedena ve výpise 10.5. Poté byl problém ověřen za pomoci následujícího kódu uvedeného ve výpisech 10.6, který obsluhuje přerušení, a 10.7, který slouží pro uložení časů do pole, ze kterého jsou následně po pozastavení vyčteny. Správnost zachycených časů je následně ověřena vykopírováním uložených hodnot do tabulkového kalkulátoru, kde je od sebe odečten aktuální a předchozí čas. Rozdíl je potom výsledný čas mezi jednotlivými impulzy. Dále je v přerušení počítáno, kolikrát bylo aktivováno. Když byl kód spuštěn ve standardním režimu, počet přerušení odpovídal počtu průchodů cyklu For. V nejhorším případě byl počet přerušení +1 v porovnání s počtem průchodů cyklu For. Pokud ovšem byl program spuštěn v režimu debugg, počet přerušení se v nejlepším případě rovnal 1511 a v nejhorším 2004. Z toho je patrné, že se hlavní program zastavuje.

Výpis 10.5: Nastavení čítače pro kontinuální vyčítaní

```
1 TC6->COUNT16.READREQ.bit.RCONT = 1;
2 TC6->COUNT16.READREQ.bit.RREQ = 1;
3 TC6->COUNT16.READREQ.bit.ADDR = 0x10;
4 while (TC6->COUNT16.STATUS.bit.SYNCBUSY);
```

Výpis 10.6: Funkce obsluhující přerušení

```
void EIC_Handler(void) {
  if ( EIC->INTFLAG.bit.EXTINT6 == 1 )
  {
    temp = REG_TC6_COUNT16_COUNT;
    pocetPreruseni++;
    cekej = false;
    // clear interrupt flag
    EIC->INTFLAG.reg = EIC_INTFLAG_EXTINT6;
  }
}
```

Výpis 10.7: Funkce sloužící pro uložení časů

```
bool floppyRead(u_int8_t* code){
  timerT6Zerro();
  timerT6Enable();
  pocetPreruseni = 0;
  cekej  = true;
  while(!(PORT->Group[INDEX_PORT].IN.reg & INDEX_PIN_MASK));
  EIC->INTENSET.bit.EXTINT6 = 1;
  for (int i = 0; i<1024; i++)
  {
    while(cekej){};
    casy[i] = temp;
    cekej = true;
  }
  EIC->INTENCLR.bit.EXTINT6 = 1;
  timerT6Disable();
  return true;
}
```

Závěr

V diplomové práci je popsána teorie k pružným magnetickým diskům. Je v ní uvedeno, jak na toto medium přistupovat. Dále se zde vysvětluje rozmístění dat na pružném magnetickém disku.

Podrobněji se rozebírá standard USB 2.0, který slouží pro připojení ovladače disketové mechaniky k počítači.

Výstupem práce je deska plošného spoje. Během testování se objevila chyba v první verzi, která se nedala předpokládat. Kombinace mikrokontroleru ATSAMD21J18A a převodníku úrovní TXS0108EPWR není schopna řídit disketovou mechaniku kvůli velkým proudům na vstupech do disketové mechaniky. Proto vznikla druhá verze ovladače disketové mechaniky, kde se využívá jako převodník úrovní invertor SN7406DR, protože jeho vstupy jsou otevřené kolektory. Tato verze desky již funguje bez problémů, navíc se signály invertují a log. 1 je v hodnotě napájecího napětí.

Software ovladače disketové mechaniky se nepodařilo dokončit. Volba mikrokontroleru ATSAMD21J18A založeném na jádře ARM® se v průběhu zprovozňování ukázala jako ne úplně nejvhodnější kvůli složitosti vnitřní konstrukce a malé dostupnosti příkladů a návodů pro tento mikrokontroler. Podařilo se nakonfigurovat USB rozhraní a je schopno se ohlásit počítači jako disketová mechanika. Dále je možné uložit data do mikrokontroleru, data jsou uchovávaná jen po dobu napájení mikrokontroleru, po odpojení jsou ztracena. Komplexnost tohoto mikrokontroleru se plně projevila v části pro vyčítaní dat z diskety, podrobněji je problém rozebrán v kapitole 10.4.13. Bohužel se tedy nepodařilo dosáhnout úspěšného řešení této problematiky. Jedno z možných řešení by bylo více používané mikrokontrolery od firmy STMicroelectronics, které poskytují také velký výpočetní výkon a existuje k nim spousta příkladů a návodů, ale bohužel situace, která nyní panuje kolem dostupností čipů způsobená celosvětovou situací, značně komplikuje jejich dostupnost.

Literatura

[1] KAMPHUIS, Sven Olaf, archyx, Malvineous, Kikinou a Peter BYE. *Floppy Diskdrive pinout. Old.Pinouts.ru* [online]. Rusko: Old.Pinouts.ru, 2019 [cit. 2020-10-31]. Dostupné z: <https://old.pinouts.ru/HD/InternalDisk_pinout.shtml>

[2] *Interfacing A Floppy Drive. Quantumnet* [online]. 2020 [cit. 2020-11-02]. Dostupné z: <http://quantumnet.wikidot.com/interfacing-a-floppy-drive>

[3] CHERNILEVSKY, George. *Floppy disk 2009 G1.jpg.* In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2009 [cit. 2020-11-15]. Dostupné z: <https://en.wikipedia.org/wiki/File:Floppy_disk_2009_G1.jpg>

[4] KASÍK, Pavel. *První disketa měla průměr 20 cm. Vzpomínáme.* In: *IDnes.cz: Technet.cz* [online]. Praha: MAFRA, 2007 [cit. 2020-11-15]. Dostupné z: <https://www.idnes.cz/technet/pc-mac/prvni-disketa-mela-prumer-20-cm-vzpominame.A070202_132416_hardware_pka>

[5] RIAZ, Rashid. *Floppy Disk. How Products Are Made* [online]. United States: Advameg, © 2020 [cit. 2020-11-15]. Dostupné z: <http://www.madehow.com/Volume-1/Floppy-Disk.html>

[6] BROWN, Gary. *How Floppy Disk Drives Work.* In: *HowStuffWorks* [online]. Venice, CA, © 2020 [cit. 2020-11-15]. Dostupné z: <https://computer.howstuffworks.com/floppy-disk-drive2.html>

[7] STORR, Phil. *Phil Storrs PC Hardware book: DOS Computers and floppy disk drives. Phil Storrs PC Hardware book* [online]. United States: Phil Storrs, 1998 [cit. 2020-11-16]. Dostupné z: <http://www.manmrk.net/tutorials/DOS/PSBOOK/book4/floppyd.html>

[8] DAVIS, Larry. *Floppy Disk Drive Pinout: FDD. Interfacebus.com: Engineering Tools and Manufacturing Links* [online]. United States: interfacebus.com, © 1998 - 2020 [cit. 2020-11-16]. Dostupné z: <http://www.interfacebus.com/PC_Floppy_Drive_PinOut.html>

[9] *List of floppy disk formats.* In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2020 [cit. 2020-11-16]. Dostupné z: <https://en.wikipedia.org/wiki/List_of_floppy_disk_formats>

[10] Fastfission. *File:Floppy disk internal diagram.svg.* In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2020 [cit. 2020-11-17]. Dostupné z: <https://commons.wikimedia.org/wiki/File:Floppy_disk_internal_diagram.svg>

[11] *Atari Low Level Formatting. Info-Coach* [online]. Francie: Info-Coach - DrCoolZic (Jean Louis-Guérin), 2015, January 15 [cit. 2021-5-19]. Dostupné z: <http://info-coach.fr/atari/software/FD-Soft.php#fd_soft_llf>

[12] *Cyklické redundantní součty a generátory. DOCPLAYER* [online]. Česká Republika: DocPlayer.cz, 2016 [cit. 2021-5-19]. Dostupné z: <https://docplayer.cz/17120355-Cyklicke-redundantni-soucty-a-generatory.html>

[13] General Floppy Drives information. *Info-Coach* [online]. Francie: Info-Coach - DrCoolZic (Jean Louis-Guérin), 2017, April 7 [cit. 2021-5-19]. Dostupné z: http://info-coach.fr/atari/software/FD-Soft.php#fd_soft_llf

[14] *The bits and the magnetic fields. The Logic of The Computer* [online]. Portugalsko: Sabercomlogica.com, 2020 [cit. 2020-11-18]. Dostupné z: <https://sabercomlogica.com/en/the-bits-and-the-magnetic-fields/>

[15] CHIDANANDAN, Prof. Archana. *An overview of FAT12* [online]. 1. Indiana, 2005 [cit. 2020-11-19]. Dostupné z: <https://www.eit.lth.se/fileadmin/eit/courses/eitn50/Literature/fat12_description.pdf>. Skripta. Rose—Hulman Institute of Technology.

[16] ANTON, Ben. *A History of the USB Standard. EzineArticles* [online]. 2009 [cit. 2020-11-21]. Dostupné z: <https://ezinearticles.com/?A-History-of-the-USB-Standard&id=2532259>

[17] *USB-IF* [online]. 2020 [cit. 2020-11-21]. Dostupné z: <https://www.usb.org/>

[18] ROBLA. *File:USB 2.0 and 3.0 connectors.svg.* In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2020 [cit. 2020-11-21]. Dostupné z: <https://commons.wikimedia.org/wiki/File:USB_2.0_and_3.0_connectors.svg>

[19] NIRIDYA. *File:USB Type-C icon.svg.* In: *Wikipedia: the free encyclopedia* [online]. San Francisco (CA): Wikimedia Foundation, 2020 [cit. 2020-11-21]. Dostupné z:
<https://commons.wikimedia.org/wiki/File:USB_Type-C_icon.svg>

[20] *USB 2.0 - díl 1. Vyvoj.hw.cz: profesionální elektronika* [online]. Praha: vyvoj.hw.cz, 2005 [cit. 2020-11-21]. Dostupné z:
<https://vyvoj.hw.cz/navrh-obvodu/rozhrani/rs-485-rs-422/usb-20-dil-1.html>

[21] *USB 2.0 - díl 2. Vyvoj.hw.cz: profesionální elektronika* [online]. Praha: vyvoj.hw.cz, 2005-03-100 [cit. 2020-11-21]. D ostupné z:
<https://vyvoj.hw.cz/navrh-obvodu/rozhrani/rs-485-rs-422/usb-20-dil-2.html>

[22] *USB 2.0 - Typy a formáty přenosů. Vyvoj.hw.cz: profesionální elektronika* [online]. Praha: vyvoj.hw.cz, 2005-03-100 [cit. 2020-11-21]. Dostupné z:
<https://vyvoj.hw.cz/navrh-obvodu/rozhrani/rs-485-rs-422/usb-20-typy-a-formaty-prenosu.html>

[23] *Defined Class Codes.* In: *USB* [online]. Oregon: VTM Group, 2016 [cit. 2021-04-08]. Dostupné z:
<https://www.usb.org/defined-class-codes>

[24] *Mass Storage Class Specification Overview 1.4.* In: *USB* [online]. Oregon: VTM Group, 2010 [cit. 2021-04-08]. Dostupné z:
<https://www.usb.org/document-library/mass-storage-class-specification-overview-14>

[25] *Mass Storage UFI Command Specification 1.0. USB* [online]. Oregon: VTM Group, 1998 [cit. 2021-4-8]. Dostupné z:
<https://www.usb.org/document-library/mass-storage-ufi-command-specification-10>

[26] YIU, Joseph. *The definitive guide to the ARM Cortex-M0.* Amsterdam: Elsevier Newnes, 2011. ISBN 978-0-12-385477-3.

[27] TIŠNOVSKÝ, Pavel. Architektura mikrořadičů s jádry ARM Cortex-M0 a ARM Cortex-M0+. *ROOT.CZ* [online]. Praha: ROOT.CZ, © 1998 — 2020 [cit. 2020-11-28]. Dostupné z:
<https://www.root.cz/clanky/architektura-mikroradicu-s-jadry-arm-cortex-m0-a-arm-cortex-m0/>

[28] *SAM D MCUs. Microchip* [online]. Chandler (AZ): Microchip Technology, c1998-2020 [cit. 2020-11-28]. Dostupné z: <https://www.microchip.com/design-centers/32-bit/sam-32-bit-mcus/sam-d-mcus>

[29] *SAMSUNG-SFD321B-070103. Techtravels.org* [online]. techtravels.org, C2018 [cit. 2021-5-21]. Dostupné z: <https://www.google.cz/url?sa=t&rct=j&q=&esrc=s&source=web&cd=&ved=2ahUKEwio38zTgtrwAhUMkhQKHcplDucQFjAAegQIBBAD&url=http%3A%2F%2Fwww.techtravels.org%2Fwp-content%2Fuploads%2Fpefiles%2FSAMSUNG-SFD321B-070103.pdf&usg=AOvVaw2VC-ZWMz-PYlZlm2TJoWJT>

Seznam symbolů a zkratek

log.	logická hodnota úrovně signálu
LBA	Logický Blok Adres
MFM	Modifikovaná frekvenční modulace – Modified frequency modulation
FM	Frekvenční modulace – Frequency modulation
FAT	tabulka umístění souboru – File Allocation Table
in	inch
HD	vyská hustota – High Density
DD	dvojitá hustota – Double Density
ED	extra vyská hustota – Extra-high Density
I	změna mezi logickými rovněmi signálu – Inversion
N	zachování předcházející logické úrovně signálu – Non Inversion
USB	univerzání seriová sběrnice – Universal Serial Bus
OTG	USB na přenosném zařízení – USB On-The-Go
IRP	paket s požadavkem na přenos dat vstup/výstup – I/O Request Packet
RISC	redukovaní instrukční sada – Reduced Instruction Set Computer
DPS	deska plošného spoje
CRC	Cyklický redundantní součet – Cyclic redundancy check

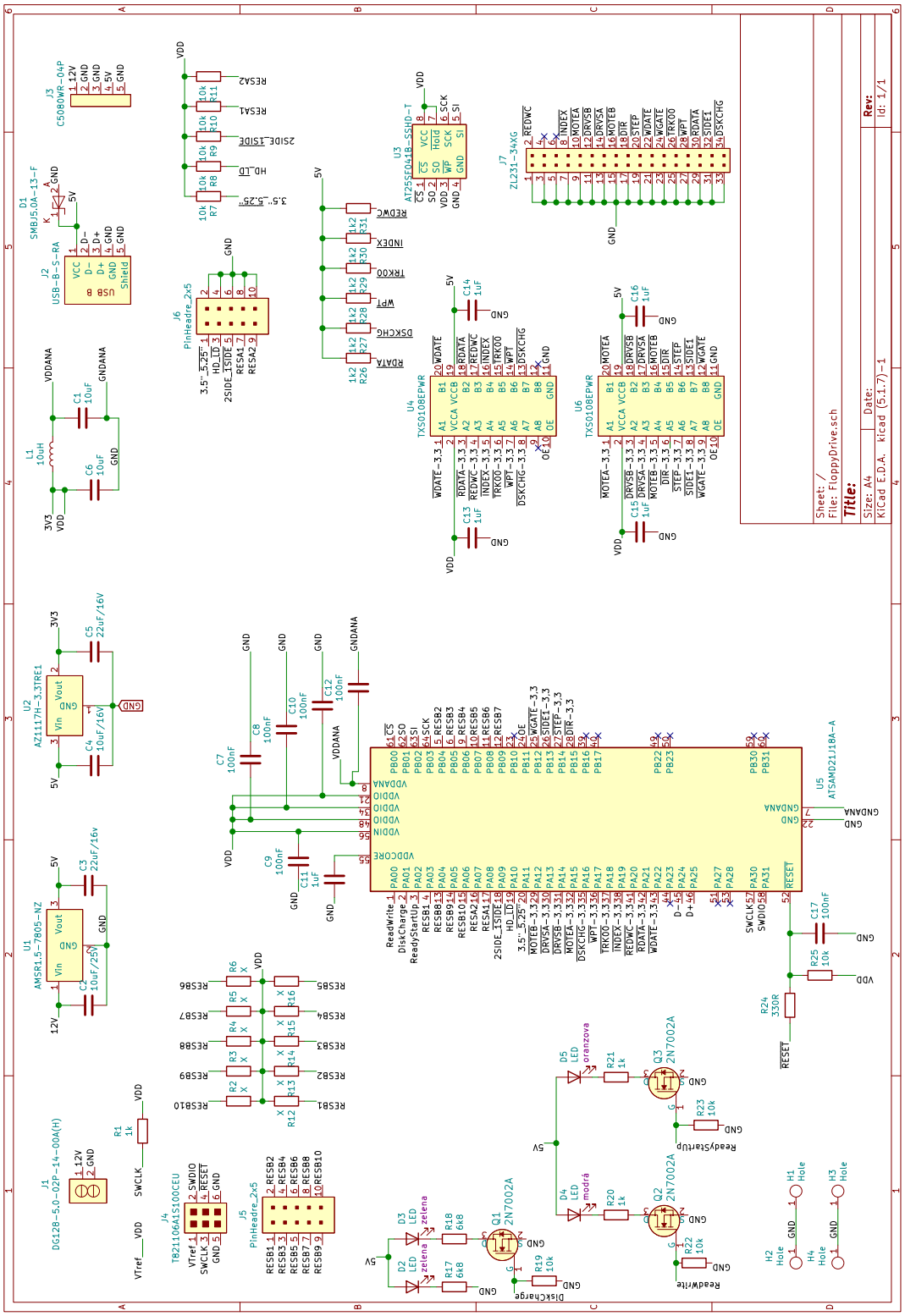
Seznam příloh

A Schéma ovladače disketové mechaniky V1 **59**

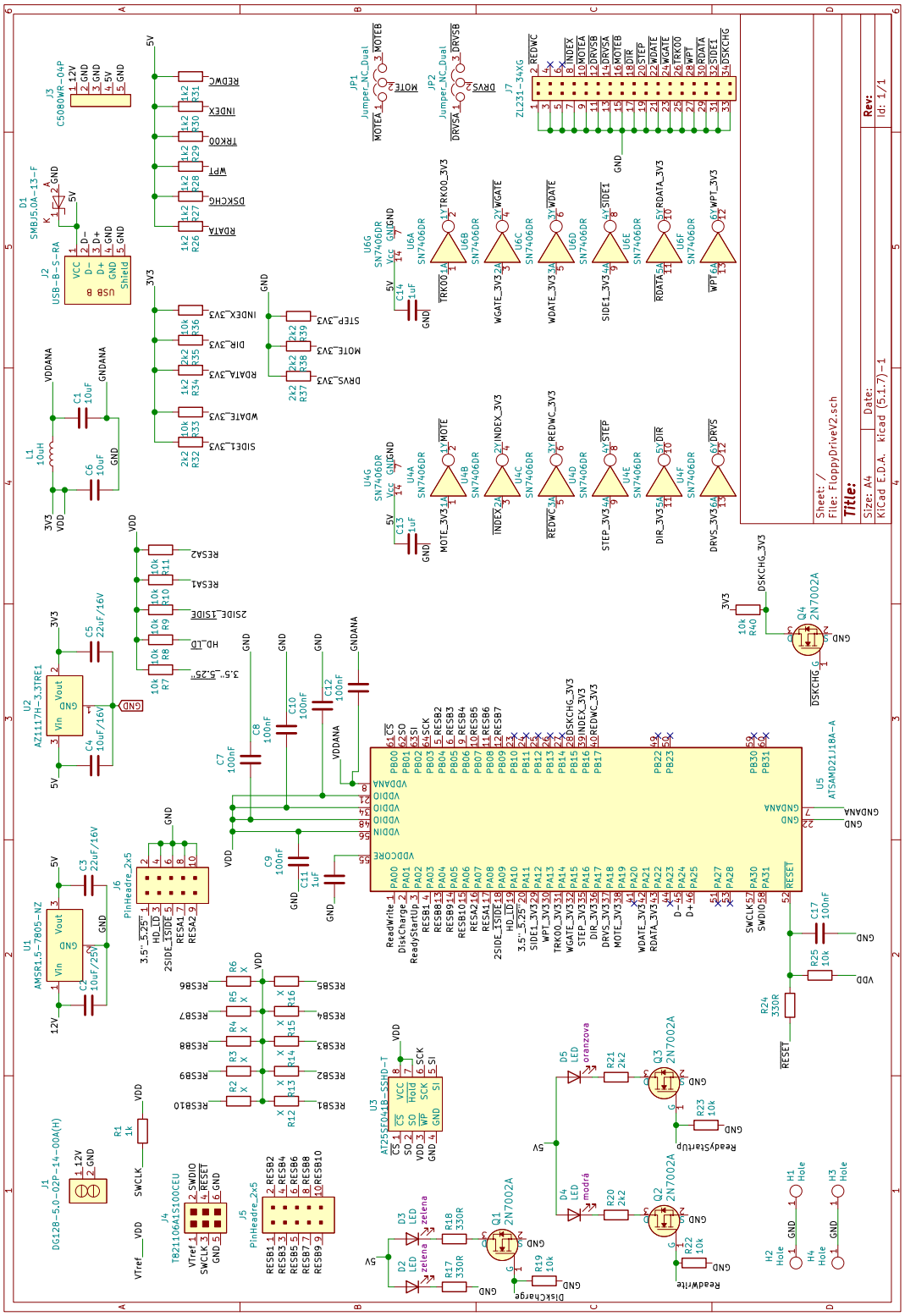
B Schéma ovladače disketové mechaniky V2 **60**

C Obsah elektronické přílohy **61**

A Schéma ovladače disketové mechaniky V1



B Schéma ovladače disketové mechaniky V2



C Obsah elektronické přílohy

V elektronická příloha umístěná na webu VUT obsahuje následující soubory:

- `/` ..... kořenový adresář přílohy zip
  - `3D` ..... 3D model krytu pro ovladač disketové mechaniky
    - `deska.FCStd`
    - `Krabička.FCStd`
    - `Krabička.FCStd1`
    - `Krabička.pdf`
    - `Krabička.step`
    - `Krabička.stl`
    - `sestava.FCStd`
    - `sestava.FCStd1`
    - `Sestava.png`
    - `Sestava.step`
  - `Floppy` ..... FW pro ovladač disketové mechaniky
    - `EXTINT_UNIT_TEST1` ..... Výstupy vývojového prostředí
    - `Floppy` ..... Zdrojové soubory projektu
    - `USB` ..... Zdrojové soubory projektu
    - `Floppy.atsln`
  - `FloppyDrive` ..... První verze DPS ovladače disketové mechaniky
    - `Gerbr` ..... Výrobní soubory pro DPS
    - `FloppyDrive.kicad_pcb`
    - `FloppyDrive.pro`
    - `FloppyDrive.sch`
    - `Gerbr.zip` ..... Výrobní soubory pro DPS
    - `FloppyDrive.xlsx` ..... Seznam použitých součástek
  - `FloppyDriveV2_KiCad5_99` ..... Druhá verze DPS ovladače disketové mechaniky
    - `Gerbr` ..... Výrobní soubory pro DPS
    - `FloppyDriveV2.kicad_pcb`
    - `FloppyDriveV2.kicad_pro`
    - `FloppyDriveV2.sch`
    - `Gerbr.zip` ..... Výrobní soubory pro DPS
    - `FloppyDriveV2.xlsx` ..... Seznam použitých součástek