VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV AUTOMATIZACE A MĚŘICÍ TECHNIKY

DEPARTMENT OF CONTROL AND INSTRUMENTATION

UNIVERZÁLNÍ MODUL ROZHRANÍ PRO TESTOVACÍ SYSTÉM

UNIVERSAL DEVICE FOR HARDWARE MODULE TESTING

DIPLOMOVÁ PRÁCE

MASTER'S THESIS

AUTOR PRÁCE — Bc. Jan Štěpka

AUTHOR

VEDOUCÍ PRÁCE — Ing. Jakub Arm, Ph.D.

SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Diplomová práce

magisterský navazující studijní program **Kybernetika, automatizace a měření**

Ústav automatizace a měřicí techniky

Student: Bc. Jan Štěpka ***ID:*** 206775

Ročník: 2 ***Akademický rok:*** 2022/23

NÁZEV TÉMATU:

Univerzální modul rozhraní pro testovací systém

POKYNY PRO VYPRACOVÁNÍ:

Úkolem této práce je navrhnout a vytvořit komunikační modul (zařízení) podporující spektrum průmyslových a embedded komunikací a I/O rozhraní, který bude přenášet příkazy z ovládacího modulu do testovaného modulu a monitorovat odezvu. Komunikace s ovládanou jednotkou bude pomocí vybrané technologie. Zařízení bude postaveno na vhodném MCU. Nakonec bude potřeba vytvořit obslužný software.

1. Definujte požadavky na rozhraní, konfiguraci a rozsahy.
2. Navrhněte a odzkoušejte hardware.
3. Navrhněte a implementujte firmware.
4. Vytvořte obslužný software.
5. Zhodnoťte vytvořený systém a Vytvořte dokumentaci.

DOPORUČENÁ LITERATURA:

VIRIUS, M. Programování v C++. Grada, 2017. 416 s. ISBN: 978-80-271-0502-1

Termín zadání: 6.2.2023 ***Termín odevzdání:*** 17.5.2023

Vedoucí práce: Ing. Jakub Arm, Ph.D.

doc. Ing. Petr Fiedler, Ph.D.
předseda rady studijního programu

ABSTRAKT

Práce je věnována návrhu elektroniky pro univerzální modul určený pro vývoj a testování v oblasti embedded systémů. V první části jsou popsány požadavky na zařízení v rámci použití komunikačních sběrnic, nastavení a rozsahy komunikací. Dále jsou popsány komunikační sběrnice používané v embedded systémech. Ve druhé části je popsaný návrh samotného zařízení.

KLÍČOVÁ SLOVA

STM32, STM32H723, RS232, RS485, CAN FD, Embedded systém, sběrnice

ABSTRACT

The thesis deals with the design of electronics for a universal module for development and testing in the field of embedded systems. The first part describes the requirements of the device, in terms of the use of communication buses, setup and communication ranges. Next, the communication buses used in embedded systems are described. The second part describes the design of the device itself.

KEYWORDS

STM32, STM32H723, RS232, RS485, CAN FD, Embedded system, bus

ŠTĚPKA, Jan. *Univerzální modul rozhraní pro testovací systém*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav automatizace a měřicí techniky, 2023, 91 s. Diplomová práce. Vedoucí práce: Ing. Jakub Arm, Ph.D.

Prohlášení autora o původnosti díla

Jméno a příjmení autora:	Bc. Jan Štěpka
VUT ID autora:	206775
Typ práce:	Diplomová práce
Akademický rok:	2022/23
Téma závěrečné práce:	Univerzální modul rozhraní pro testovací systém

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 16.5.2023 .... podpis autora*

*Autor podepisuje pouze v tištěné verzi.

PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu diplomové práce panu Ing. Jakubovi Armovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci. Dále bych také rád poděkoval společnosti DIVELIT system s.r.o za poskytnutí zázemí a zdrojů k vytvoření této práce.

Obsah

Seznam obrázků

Seznam tabulek

Seznam výpisů

Úvod

Na trhu lze najít nespočet samostatných převodníků, například převodník sběrnice RS485 na USB, který slouží k monitorování dané sběrnice připojením k počítači. Při zkoumání dostupných řešení nebylo nalezené žádné řešení, kde by bylo možné připojit a monitorovat více sběrnic současně s možností libovolně konfigurovat či je mezi sebou přesměrovat s možným nastavením napěťové úrovně sběrnic založených na TTL (Transistor Transistor Logic) logice. Zařízení navrhnuté v této práci bude sloužit jako univerzální alternativa jednotlivých převodníků.

Cílem práce je navrhnout a realizovat univerzální zařízení pro vývoj a testování v oblasti embedded systémů. Jedná se o kompaktní zařízení, na které bude možné připojení více digitálních komunikačních sběrnic jako je například CAN, LIN, RS485, ale i SPI či sběrnice I^2C. Tyto sběrnice bude možno konfigurovat v rámci možností definovaných v práci. Důležitou součástí navrhovaného zařízení bude možnost komunikace s počítačem v reálném čase pomocí rozhraní USB. To umožní přenos dat z digitálních sběrnic přímo do počítače, kde budou data zpracována a analyzována. Uživatelé budou mít také možnost odesílat data z počítače do zařízení a simulovat tak různé scénáře komunikace. Zařízení bude krom digitálních komunikačních sběrnic také obsahovat digitální vstupy-výstupy, výstup pro generování PWM (Pulzně šířková modulace), analogově-digitální a digitálně-analogové převodníky. Pro monitorování a správu zařízení bude vytvořeno uživatelské rozhraní na straně počítače, které bude sloužit jako nástroj pro sledování a analyzování přenosu dat mezi zařízením a počítačem.

Práce je rozdělena do dvou částí. V teoretické části jsou uvedeny požadavky na zařízení, se kterým je spjatý teoretický popis použitých sběrnic. Praktická část diplomové práce se zabývá návrhem zařízení samotného. Návrh obsahuje vytvoření hardwaru zařízení, firmware mikrokontroléru a obslužný software pro počítač.

1 Požadavky na rozhraní, konfiguraci a rozsahy

Idea testeru spočívá v možnosti využití jednoho zařízení k otestování a odladění co nejvíce možných prvků při vývoji embedded zařízení. Krom testování a ladění bude zařízení také sloužit k vývoji ovladačů pro různé snímače a bezdrátové moduly. Výsledné zařízení by mělo obsahovat běžně používající komunikační periferie jako jsou I^2C, SPI, UART, RS232, RS485, CAN či CAN FD, Lin, USB a ethernet. USB bude možno využít k získávání dat. Zařízení bude také obsahovat AD a DA převodníky, možnost generování PWM a alespoň 8 digitálních vstupů, výstupů. Na desce také bude k dispozici SD kartu pro případné logování dat.

Konfigurace periferií:

- Konfigurace SPI:
  - Rychlost komunikace alespoň do 10 MHz,
  - Nastavení polarity a fáze hodin.
- Konfigurace UART:
  - Znaková rychlost 1200 bd - 230400 bd.
- Konfigurace I^2C:
  - Rychlost komunikace alespoň do 400 kb/s.
- Konfigurace RS232:
  - Znaková rychlost 1200 bd - 230400 bd,
  - Řízení datového toku.
- Konfigurace RS485:
  - Znaková rychlost 1200 bd - 230400 bd,
  - Terminační odpor.
- Konfigurace CAN:
  - Komunikační rychlost alespoň do 1 Mb/s,
  - Filtr ID,
  - ID k poslání,
  - Terminační odpor.

1.1 Výběr mikrokontroléru

Výběr mikrokontroléru byl zaměřený na procesory STM32 a to z důvodu předešlých zkušeností. Hlavními parametry při výběru bylo, aby daný procesor byl dostatečně rychlý, a měl možnost připojení všech potřebných periferií. Vzhledem k nutnosti využití systému reálného času a velkého množství komunikačních sběrnic byl hlavní požadavek především na dostatek paměti. Byl zde ohled na možnost připojení externí RAM paměti. Z důvodu poměru ceny za množství paměti byl brán ohled na možnost využití SDRAM (Synchronous Dynamic Random Access Memory). Z těchto důvodů byly vybrány mikrokontroléry STM32H723ZG, STM32H725IE a STM32F746ZE. Mikrokontrolér STM32F746ZE nebyl zvolen, jelikož má oproti mikrokontrolérům STM32H723ZG a STM32H725IE pomalejší takt (216 MHz oproti 550 MHz). Kromě vyšší frekvence mají mikrokontroléry STM32H7 možnost využití CAN FD oproti CAN 2.0. CAN FD je zpětně kompatibilní s CAN 2.0, a vychází jako nejvhodnější. Mikrokontroléry STM32H723ZG a STM32H725IE mají obdobné parametry. V této práci byl zvolen mikrokontrolér STM32H723ZG pro jeho lepší dostupnost v době návrhu.

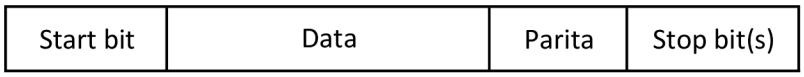
2 Komunikační sběrnice

Pro možnost přenosu dat existuje mnoho komunikačních sběrnic. Ty se mohou dělit na sériové a paralelní. Sběrnice se používají nejen pro komunikaci s jinými zařízeními, ale také pro komunikaci uvnitř procesoru a i s jednotlivými periferiemi v rámci DPS. V této části práce jsou popsány jednotlivé komunikační sběrnice, které budou použity ve finálním zařízení.

2.1 RS232

RS232 slouží jako fyzická vrstva pro UART, který nemá definované elektrické vlastnosti. Sběrnice má možnost komunikace oběma směry zároveň a je tzv. "Full-duplex". Sběrnice také podporuje komunikaci pouze mezi dvěma zařízeními. Napěťové úrovně pro RS232 jsou definovány jako +5 V až +15 V pro logickou 0 a -5 V až -15 V pro logickou 1. [18]

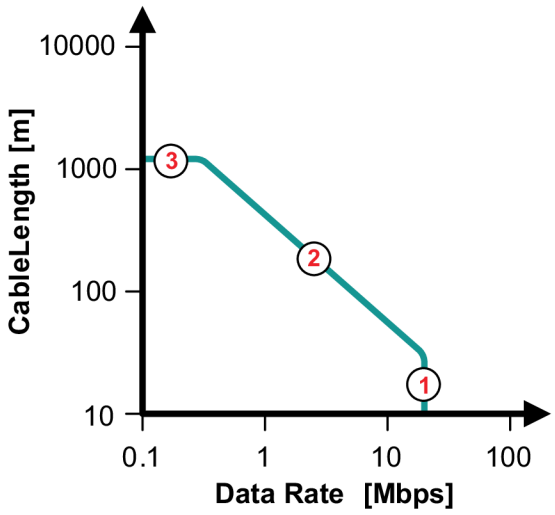
Datový rámec, který lze vidět na obrázku 2.1, začíná start bitem. Ten slouží k synchronizaci mezi zařízeními. Neboť nečinná sběrnice má stav logické 1, start bit je reprezentovaný změnou na sběrnici, to znamená jako logická 0. Ihned po start bitu začínají data. Ty mohou mít délku 7, 8 , nebo 9 bitů. Nejběžnější využití sběrnice RS232 je pro přenos ASCII znaků. K detekci chyby přenosu slouží paritní bit. Ten může být sudý, nebo lichý. Paritní byt je nastavený v závislosti na počtu logických 1 dat. V případě lichého počtu logických 1 je lichá parita nastavená na logickou 1, sudá na logickou 0. V případě sudého počtu je to naopak. Nevýhodou tohoto jednoduchého principu je možnost detekce pouze lichého počtu změněných bitů. V případě změny dvou bitů chyba nebude odhalena. Z tohoto důvodu se pro přenos většího množství dat využívají navíc jiné principy detekce chyb, jako je například cyklický redundantní součet, nebo-li CRC. Pakety jsou zakončené Stop bity, které je možné nastavit na 1 bit, 1,5 bitu, nebo 2 bity. [18]



Obr. 2.1: Datový rámec pro UART/RS232

2.2 RS485

RS485 definuje pouze fyzickou vrstvu zařízení. Oproti sběrnicím jako je CAN nemá definované datové rámce. RS485 využívá pro komunikaci diferenciální pár zakončený terminačním odporem. Na sběrnici je možné připojit až 32 zařízení. Maximální datový tok je závislý na délce vedení podle grafu zobrazeným na obrázku číslo 2.2. V grafu jsou definovány 3 sekce. První sekce pro velmi krátké vedení, kde lze zanedbat ztráty na vedení. Standard doporučuje maximální rychlost 10 Mbps, nicméně dnešní rychlé integrované obvody zvládnou rychlost až 40 Mbps. Druhá část definuje vedení, které má již nezanedbatelný odpor. Ztráty vedení již v této sekci nelze zanedbat a při zvětšující se vzdálenosti je nutné snížit přenosovou rychlost. Zde by délka vedení v metrech vynásobená rychlostí v bps neměla přesáhnout 10^7. Pro třetí sekci se již odpor vedení blíží hodnotě terminačního odporu a napěťový dělič vzniklý díky tomuto odporu vedení způsobí útlum -6 dB.[23]



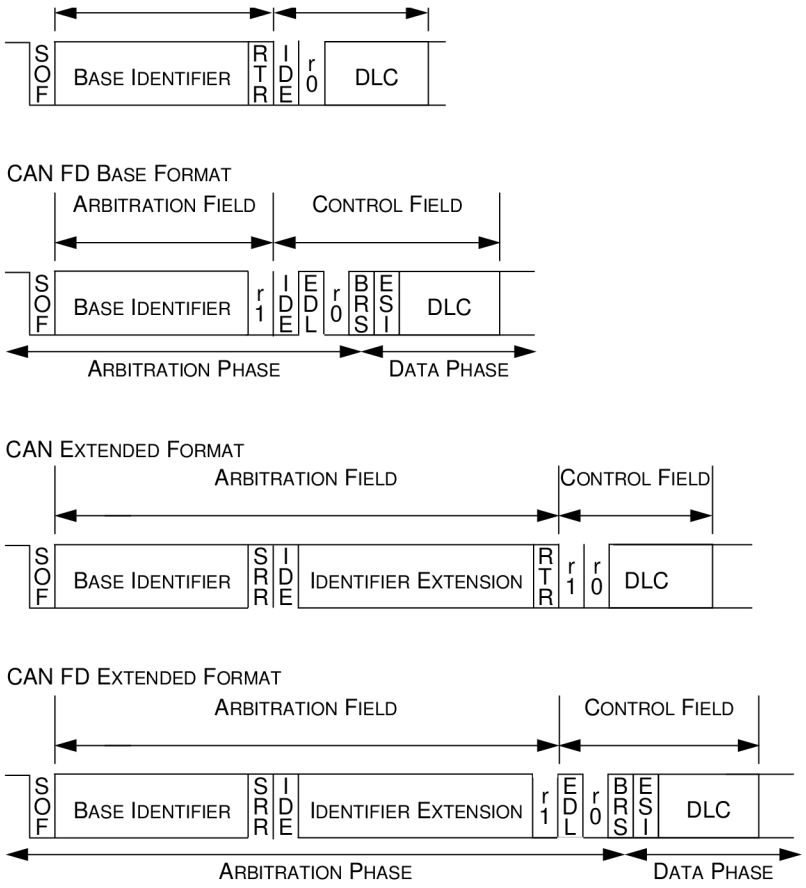
Obr. 2.2: Závislost maximální délky kabelu na datovém toku[23]

Pro správný přenos dat bez odrazů je nutné sběrnici zakončit terminačním odporem. Jeho hodnota by se měla rovnat impedanci vedení. RS485 standard doporučuje kabely s impedancí 120 Ω a z toho důvodu je doporučená hodnota terminačního odporu právě 120 Ω. V případě využití aplikace v zarušeném prostředí, se 120 Ω odpor nahrazuje RC článkem zapojeným jako T-filter se dvěma 60 Ω odpory.

2.3 CAN

CAN, nebo-li Controller Area Network, je sběrnice navržená společností Robert Bosch. CAN je navržený jako řešení pro redukci kabeláže v automotivu. Sběrnice je navržená pro vysokou imunitu proti rušení a schopnost odhalit a opravit chybu v odeslaných datech. Komunikace probíhá po jednom diferenciálním páru, který se zakončuje terminačním odporem. Oproti USB či ethernetu, tato Multimaster sběrnice posílá krátké zprávy. Zprávy jsou vysílány všem zařízení na sběrnici, což zajistí konsistentní data pro každý prvek sběrnice. Pro využití v embedded systémech je nutné využít mikrokontrolér obsahující CAN periferii. [20]

CAN FD, neboli CAN s flexibilní datovou rychlostí, sdílí fyzickou vrstvu s CAN 2.0. Nicméně datové rámce jsou oproti CAN 2.0 odlišné. Jejich rozdíly jsou zobrazeny na obrázku 2.3. CAN FD umožňuje větší přenosovou rychlost než 1 Mbps a odesílání více než 8 bytů. Maximální počet bytů poslaný přes CAN FD je 64. Pro CAN FD je možná zpětná kompatibilita s CAN 2.0, naopak CAN 2.0 není kompatibilní s CAN FD.



Obr. 2.3: Rozdíly datových rámců pro CAN 2.0 a CAN FD[21]

Vysvětlivky k obrázku 2.3:

- SOF - Start of Frame - začátek datového rámce
- Base Identifier - 11 bitová identifikace
- RTR - REMOTE TRANSMISSION REQUEST - Využit pouze pro CAN 2.0. Pro CAN FD je nahrazen rezervním bitem r1
- IDE - IDENTIFIER EXTENSION FLAG
- EDL - EXTENDED DATA LENGTH - Pouze pro CAN FD.
- r0 - Rezervní bit na budoucí rozšíření.
- BRS - BIT RATE SWITCH - Slouží k změně bitové rychlosti na alternativní předkonfigurovanou rychlost
- ESI - ERROR STATE INDICATOR
- SRR - SUBSTITUTE REMOTE REQUEST
- Identifier Extension - Rozšíření identifikace o 18 bitů
- DLC - Data Length Code - Počet bytů podle tabulky 2.1[21]

Tab. 2.1: Kód datové délky pro CAN [21]

	Počet bytů	Kód datové délky			
		DLC3	DLC2	DLC1	DLC0
CAN / CAN FD	0	0	0	0	0
	1	0	0	0	1
	2	0	0	1	0
	3	0	0	1	1
	4	0	1	0	0
	5	0	1	0	1
	6	0	1	1	0
	7	0	1	1	1
CAN	8	1	x	x	x
CAN FD	8	1	0	0	0
	12	1	0	0	1
	16	1	0	1	0
	20	1	0	1	1
	24	1	1	0	0
	32	1	1	0	1
	48	1	1	1	0
	64	1	1	1	1

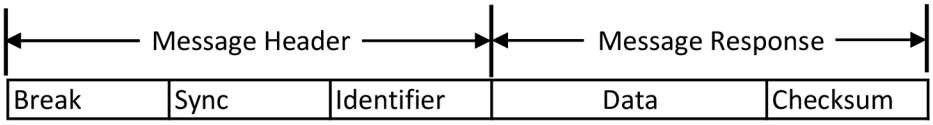
2.4 LIN

Local Interconnect Network, nebo-li LIN je nízkonákladové řešení pro embedded systémy sloužící k přenosu dat. Tato jednovodičová sběrnice slouží jako levnější náhrada CAN sběrnice pro aplikace, kde není potřeba její rychlost a všestrannost. LIN nabízí přenosovou rychlost až 20 kbps. Ke implementaci v embedded systémech se využívá UART. To umožňuje využití méně komplexních mikrokontrolérů. Oproti CAN sběrnici, která má Multi-master topologii, LIN využívá topologii master-slave.[19]

Datový rámec LIN sběrnice, který lze vidět na obrázku 2.4, začíná s tzv. "Break", který slouží jako upozornění začátku komunikace pro všechny zařízení na sběrnici. Break obsahuje typicky 13, nebo více bitů s hodnotou logická 0, který následují jedním bitem s hodnotou logická 1. Po breaku následuje synchronizace, která je definovaná jako 0x55. Synchronizace umožňuje detekci znakové rychlosti. Zařízení na sběrnici se přizpůsobí k detekované znakové rychlosti. Po synchronizaci následuje identifikace. Identifikace obsahuje spodních šest bitů pro adresaci a horní dva bity pro paritu. Samotné ID může nabývat hodnot 0 až 63. Paritní bity se vypočítají pomocí rovnice 2.1 a 2.2. Datová délka je standardně součástí ID. Samotná datová část obsahuje 1 až 8 bytů. Poslední část datového rámce je kontrolní součet. Klasický kontrolní součet je počítaný pouze z samotných dat. Vylepšený kontrolní součet krom samotných dat využívá také ID. Klasický kontrolní součet je definovaný v standardu LIN 1.3 a pro adresy 60-63. Vylepšený kontrolní součet se využívá pro standard 2.0. Kontrolní součet se počítá jako suma všech hodnot s odečtením 255 pokaždé, kdy hodnota přesáhne právě 255.[22]

$$P(0) = ID(0) \oplus ID(1) \oplus ID(2) \oplus ID(4) \tag{2.1}$$

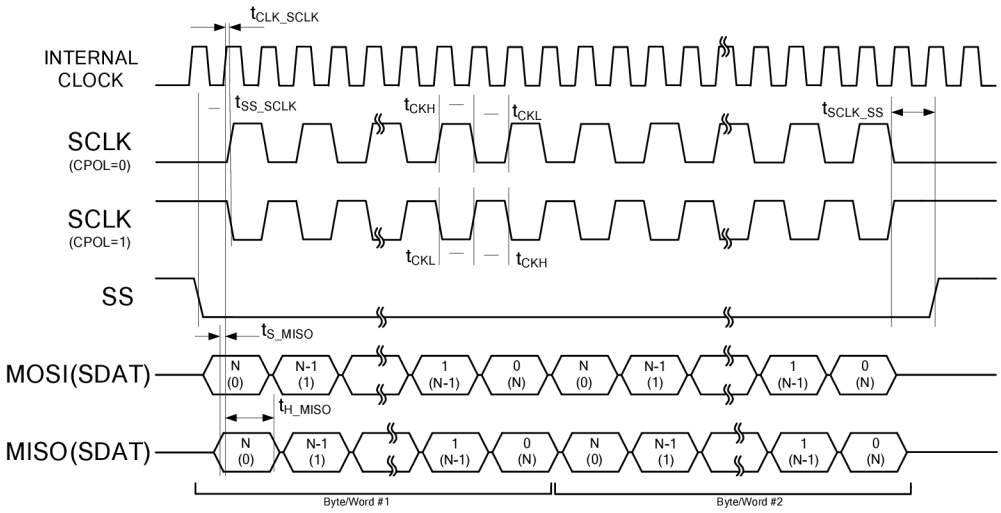
$$P(1) = ID(1) \oplus ID(3) \oplus ID(4) \oplus ID(5) \tag{2.2}$$



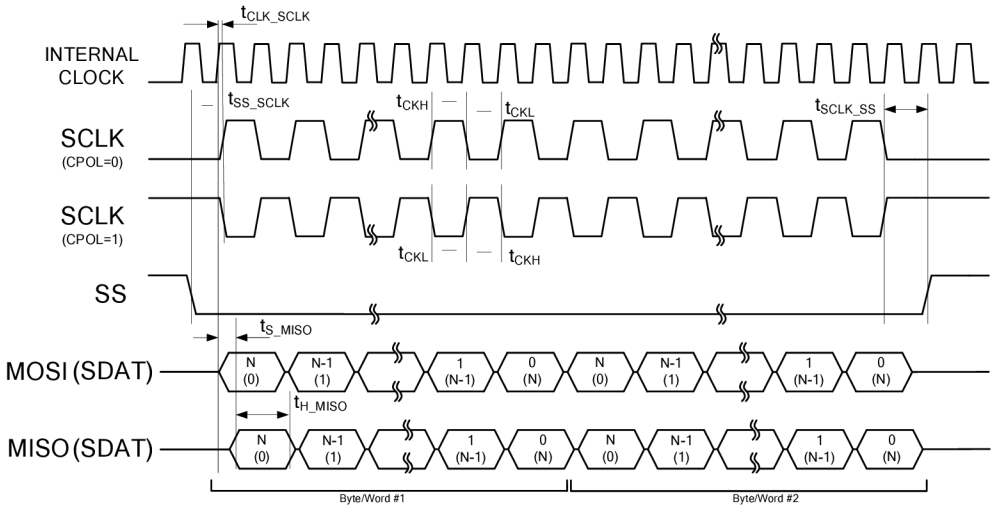
Obr. 2.4: Datový rámec LIN sběrnice[22]

2.5 SPI

SPI, nebo-li Serial Peripheral Interface, je sériová synchronní sběrnice, která umožňuje připojit dva a více zařízení na topologii master-slave. Sběrnice využívá čtyři signály. Dva signály jsou datové, tzv. MOSI (Master Output Slave Input) a MISO (Master Input Slave Output). Jeden signál je hodinový a poslední signál pro výběr zařízení se kterým chce komunikovat (SS - Slave Select). Ten bývá často negovaný. [18]



Obr. 2.5: Časový diagram pro CPHA = 0[24]



Obr. 2.6: Časový diagram pro CPHA = 1[24]

Na obrázku 2.5 a 2.5 jsou vyobrazeny časové diagramy SPI sběrnice. Jelikož synchronizace pouze pomocí hodinového taktu by mohla vést ke ztrátě dat, jsou definovány i synchronizační protokoly. Zde se využívá konceptu polarity hodin (CPOL) a fáze hodin (CPHA).

Pro případ CPOL = 0 jsou hodiny v logické 0 pro stav kdy neprobíhá komunikace. Při CPOL = 1 jsou hodiny v logické 1. Fáze hodin definuje hranu na kterou zařízení reaguje. Kombinací těchto parametrů vznikají módy, které jsou zobrazeny v tabulce 2.2.

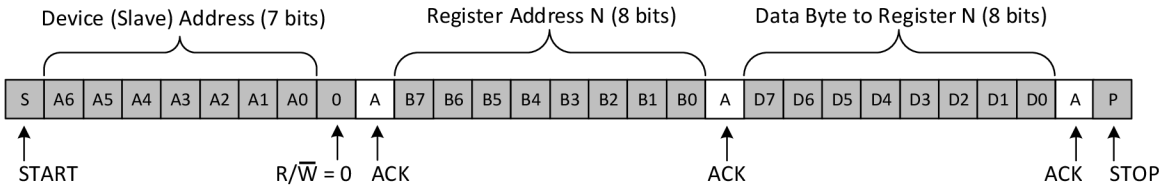
Tab. 2.2: Módy definované pro sběrnici SPI

SPI mód	**Polarita Hodin** CPOL	**Fáze hodin** CPHA	**Hrana hodin**
0	0	0	Náběžná
1	0	1	Sestupná
2	1	0	Náběžná
3	1	1	Sestupná

2.6 I²C

I²C je dvouvodičová multi-master multi-slave sběrnice, která využívá open-drain vstup-výstup pro obousměrnou komunikaci. Z toho důvodu se ke sběrnici připojuje pull-up. Ke komunikaci se využívá jeden datový (SDA) signál a jeden hodinový signál (SCL). [25]

Komunikace je zahájená startovacím stavem, což je vytvořeno "stažením" datové linky na logickou 0 s následujícím zahájením hodinových pulsů. Nejprve se po sběrnici vysílá adresa zařízení se kterým bude komunikovat. Adresa je 7 bitová a následuje požadavek na zápis nebo čtení. Každý byte poslaný po sběrnici je potvrzený takzvaným "Acknowledge" bitem ze strany příjemce. Ten potvrzuje příjem dat a dává najevo vysílači, že v komunikaci může pokračovat. V případě uplynutí stanovené doby, kdy po vyslání bytu zůstane sběrnice v logické 1, se stav bere jako "Not Acknowledge", nebo-li NACK. Komunikace se ukončuje pomocí stop kondice. NACK může nastat vlivem odeslání špatné adresy, nebo když je zařízení zaneprázdněno vykonáváním jiných například real-timových funkcí, či není schopný přijmout další data. Příklad komunikace je zobrazený na obrázku 2.7.[25]

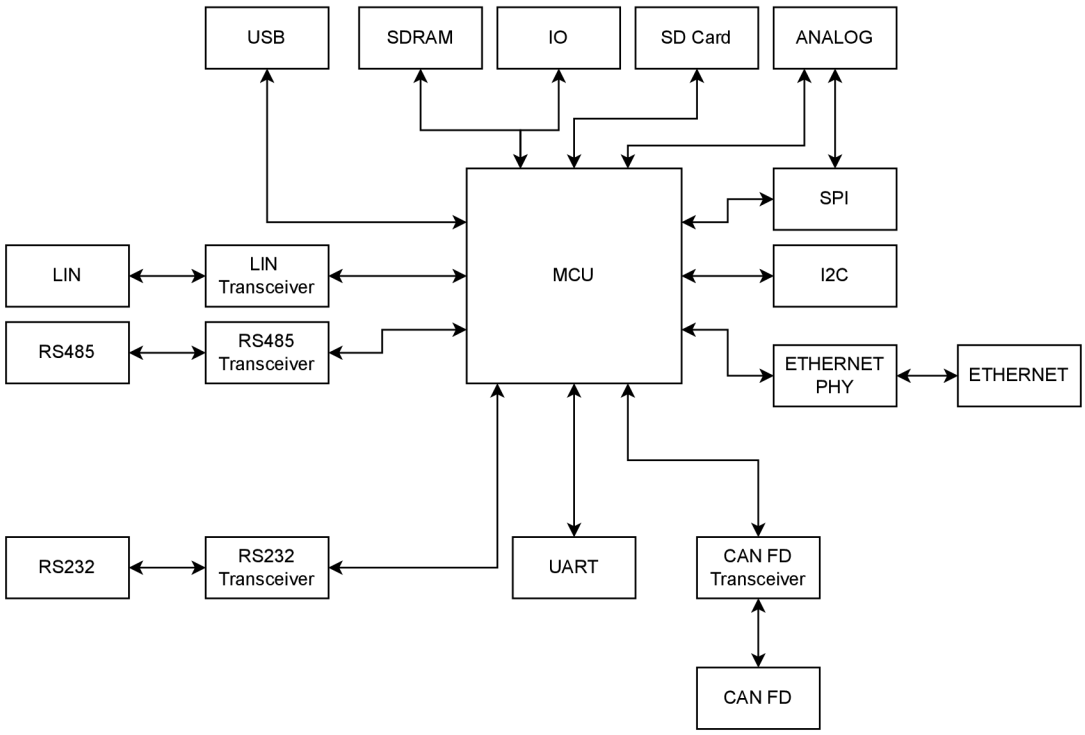


Obr. 2.7: Příklad komunikace po I²C sběrnici.[25]

3 Návrh elektrického schématu

Tato kapitola je věnována návrhu desky plošného spoje. Celý hardware byl navržený pomocí programu Altium Designer. Deska obsahuje značný počet sběrnic využívaných ve vestavěných systémech, analogové i digitální vstupy-výstupy. Blokové schéma zapojení lze vidět na obrázku 3.1.

Jádrem celé desky je mikrokontrolér STM32H723ZGT6. Ten disponuje 114 vstupy-výstupy. Z tohoto počtu nejsou využity pouze 3 vstupy-výstupy. Takt jádra lze nastavit až na 550 MHz, což umožní rychlé zpracování dat pro potřeby chodu v reálném čase. Mikrokontrolér obsahuje 1 MB Flash paměti a 564 kB RAM paměti. Paměť je rozdělena na 128 kB úzce spjaté paměti (TCM) určené pro kritické data v reálném čase, 432 kB systémové paměti a 4 kB záložní paměti. Ze systémové RAM lze až 256 kB přemapovat na TCM instrukční paměť. K mikrokontroléru je navíc připojena externí SDRAM o velikosti 16 MB přes paralelní sběrnici. Také je možné využít až čtyři jednotky pro přímý přístup do paměti (DMA). Díky těmto parametrům mikrokontroléru je možné zajistit vykonávání aplikace v reálném čase. Také rozsáhlé možnosti periferií tohoto procesoru umožňují snížení nutných kompromisů pro vývoj této aplikace.[1]



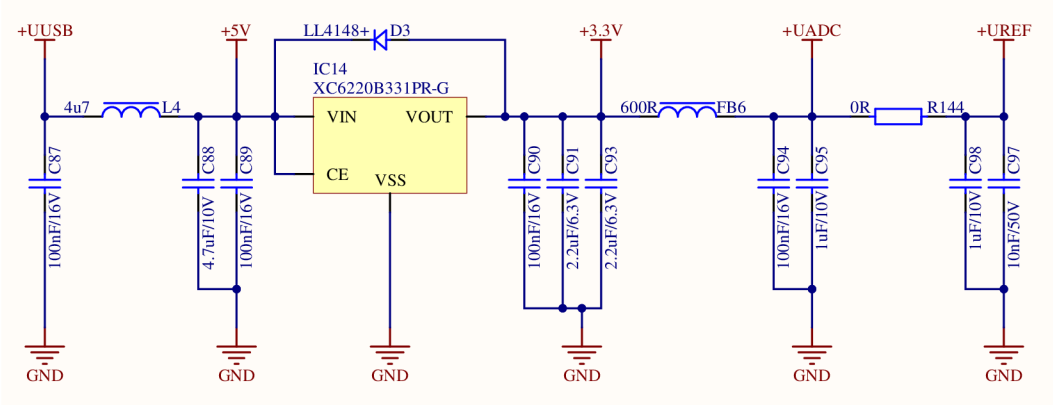
Obr. 3.1: Blokové schéma zapojení.

3.1 Napájení

Deska je napájena pomocí USB, jako vstupní napětí je 5 V. Hlavní 3.3 voltovou větev zajišťuje lineární napěťový regulátor s nízkým úbytkem napětí XC6220B331PR-G. Tento regulátor napětí nabízí maximální výstupní proud 1 A. Uvedený úbytek napětí při odběru 300 mA je typicky 60 mV, maximálně 95 mV v závislosti na teplotě.[4]

Vstupní napětí je filtrováno pomocí LC filtru. Výstup je spojení se vstupem přes diodu D3, jak je vidět na obrázku 3.2. Ta slouží jako ochrana před zpětným proudem, kdy proud teče z výstupu na vstup. Zpětný proud může způsobit zhoršení spolehlivosti regulátoru napětí, nebo ho i trvale poškodit.[5]

Pro analogovou část je 3,3 voltová větev filtrovaná pomocí feritové perly a také pro případ většího zarušení je možné vyměnit odpor R144 za nenulový a vytvořit tak RC filtr pro referenční napětí.



Obr. 3.2: Zapojení 3,3 V zdroje napětí.

K provozu LIN sběrnice je potřeba napájení v nominálním rozsahu 7 V až 18 V, s maximálním napětím 27 V. Potřebné napětí zajišťuje pulzně šířkově modulovatelný spínaný zdroj MIC2619YD6-TR. Ten má přivedené vstupní napětí 5 V a má nastavenou zpětnou vazbu pomocí vzorečku 3.1. Výstupní napětí je nastaveno na 13,9 V. Dále má nastavenou přepěťovou ochranu pomocí vzorečku 3.2, což vychází na 26,5 V.[6]

$$V_{OUT} = 1,265 \cdot \left(\frac{R139}{R143} + 1\right) = 1,265 \cdot \left(\frac{100}{10} + 1\right) = 13,9V \tag{3.1}$$

$$V_{OVP} = 1,265 \cdot \left(\frac{R140 + 141}{R142} + 1\right) = 1,265 \cdot \left(\frac{100 + 100}{10} + 1\right) = 26.5V \tag{3.2}$$

Hlavní napájecí větev celé desky má hlídané napětí pomocí napěťového detektoru XC6120N302NR-G. V případě napětí nižšího než 3 V vypne napájecí zdroj pro LIN sběrnici. Výstup tohoto detektoru napětí je také připojený na pin PG3 mikrokontroléru.

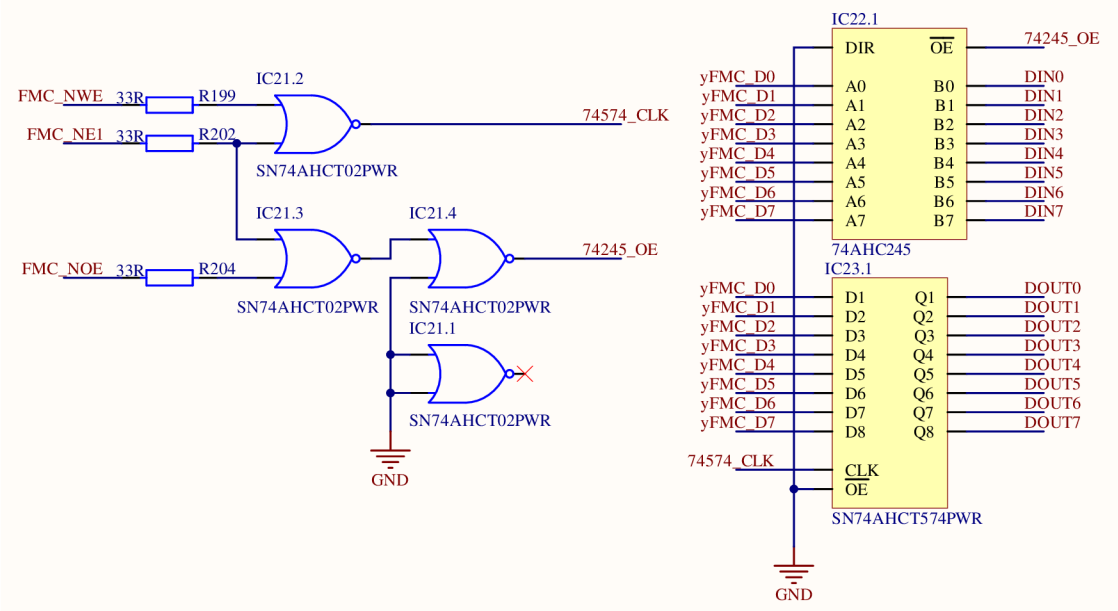
3.2 Digitální vstupy výstupy

Digitální vstupy a výstupy jsou navrženy z pohledu procesoru jako osmibitová SRAM o kapacitě 1 bajte, která sdílí na FMC (Flexible Memory Controller) datovou sběrnici s SDRAM pamětí. Důvodem tohoto kroku je nedostatek volných pinů procesoru.

Digitální vstupy desky jsou připojeny na osmibitový transceiver 74AHC245, který je zapojený s uzemněným DIR signálem, tak aby data byla vždy směrovaná z konektoru na paměťovou sběrnici procesoru. Signál pro povolení výstupu je řízený pomocí negovaného logického součtu signálů NE1 (výběr čipu) a NOE (signál pro čtení). [1]

Digitální výstupy jsou navrženy pomocí osmibitového bistabilního klopného obvodu typu D SN74AHCT574PWR. Data se na výstup dostávají pomocí náběžné hrany. Toho je dosaženo logickým součtem signálů NE1 (výběr čipu) a NWE (povolení zápisu).

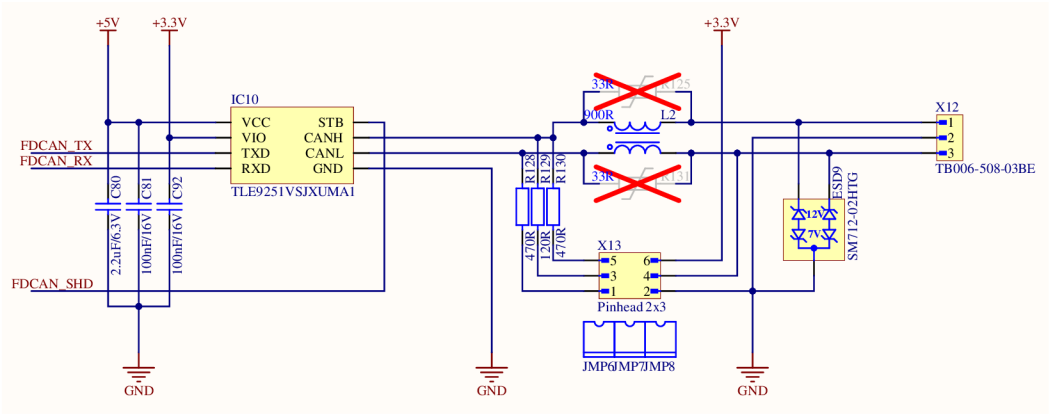
Pro logické operace potřebné k ovládání vstupů a výstupů jsou realizované pomocí čipu se čtyřmi integrovanými NOR hradly. Datová sběrnice je připojena přes 33 Ω odpory z důvodu ochrany. Zapojení ovládání digitálních vstupů-výstupů lze vidět na obrázku 3.3.



Obr. 3.3: Zapojení digitálních vstupů-výstupů.

3.3 Sběrnice CAN

Pro sběrnici CAN je využit transceiver TLE9251VSJXUMA1. Ten je připojený k interní periferii mikrokontroléru. Oba tyto prvky jsou kompatibilní se standardem CAN FD (Controller Area Network with Flexible Data-Rate). Z mikrokontroléru jsou vyvedeny dva datové signály, a to pro vysílání a pro příjem. Krom datových signálů je také vyveden signál pro uvedení transceiveru do úsporného režimu. Výstup transceiver je jeden diferenciální pár. Na výstup sběrnice lze osadit buďto filtr souhlasného napětí, nebo 33 Ω termistory. V základním osazení je využit filtr souhlasného napětí, viz obrázek 3.4. Výstup je dále opatřený ESD (Electro-static discharge) ochranou. Pomocí konektoru X13 lze k výstupu připojit pull-up na CANH, pull-down na CANL a také terminační odpor podle potřeb uživatele.

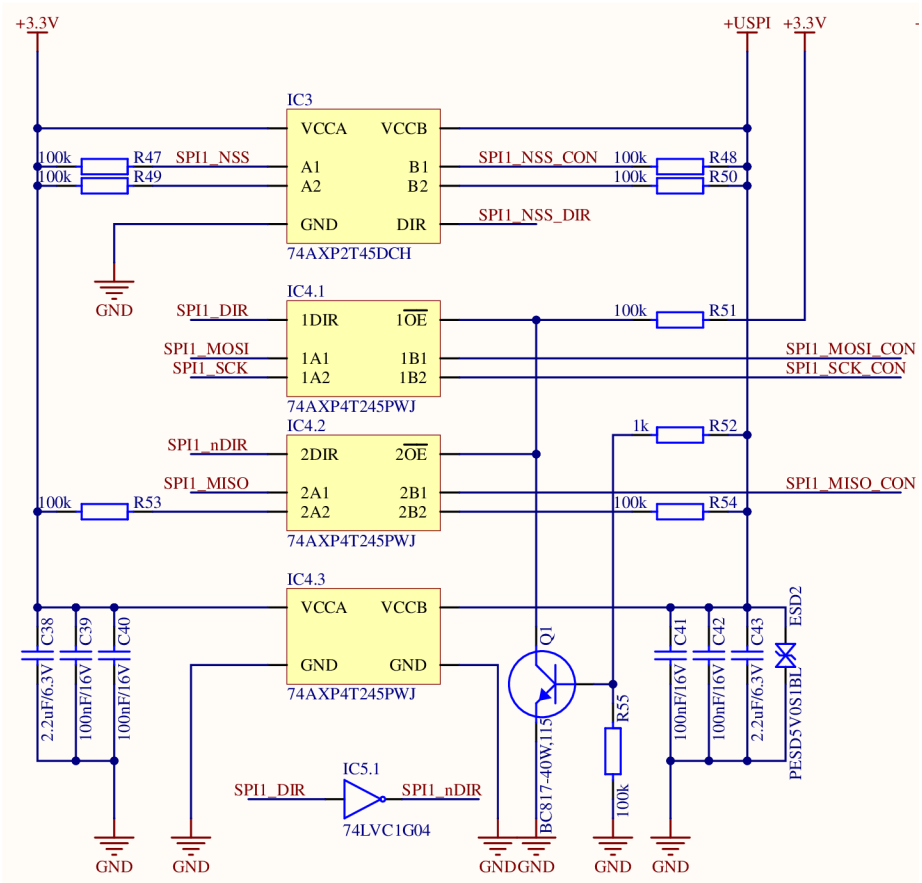


Obr. 3.4: Zapojení CAN transceiveru.

3.4 Sběrnice SPI

Vstupy a výstup SPI na desce je připojený přes integrovaný obousměrný buffer s funkcí dvojího napájení. Využitím tohoto integrovaného obvodu se omezí maximální provozní frekvence SPI na 25 MHz při napájení 3,3 V. Uživatel má možnost připojit externí napájení v rozmezí 1 V až 5 V pro případ využití jiné napěťové úrovně SPI, než je využita na mikrokontroléru STM32. Pro případ využití právě tohoto napětí, je možnost propojit konektor X2. Ten slouží k propojení externího napájení SPI s interní 3.3 V větví. Výstup integrovaných obousměrných bufferů je opatřen detekcí napětí pomocí tranzistoru Q1, jehož zapojení lze vidět na obrázku 3.5.

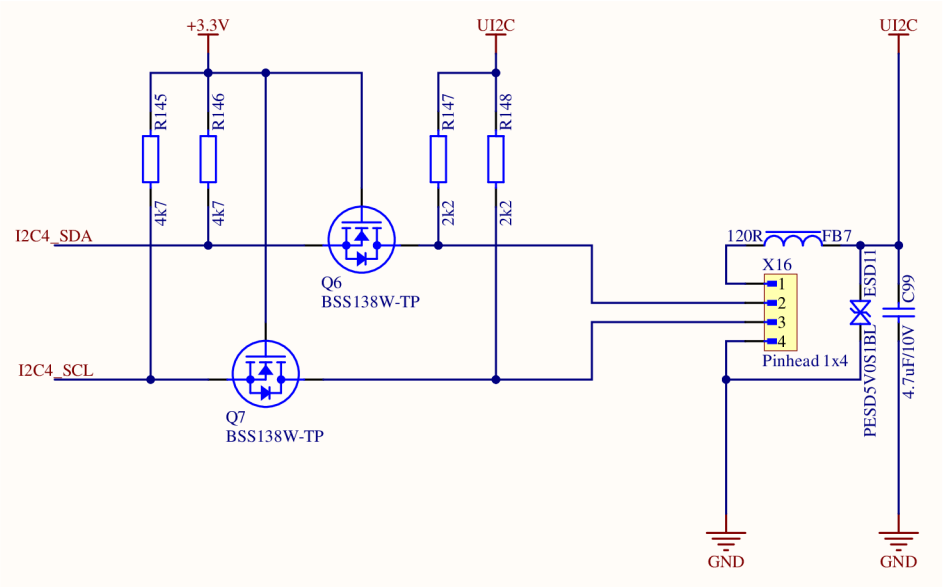
Směr SPI komunikace lze měnit pomocí SPI1_DIR signálu připojeného na pin mikrokontroléru PB6. Signály MOSI, SCK jsou ovládány přímou logikou. Signál MISO je ovládaný přes negaci a NSS lze ovládat nezávisle pomocí SPI_NSS_DIR signálu připojený na pin PG13. Přepínáním směru lze volit zda zařízení pracuje jako master a to v případě logické úrovně 1 signálu SPI1_DIR. Pro případ logické úrovně 0 se chová zařízení jako slave. Jelikož se většinou využívá negovaný výběr čipu, signál je opatřený pull-upem, aby nebyl aktivní v době inicializace procesoru, nebo jiných změnách konfigurace výstupu mikrokontroléru. Sběrnice je také opatřena ESD ochranou, která zároveň slouží jako filtr typu dolní propust pro snížení EMI (Elektro-magnetic Interference). Externí napájení je filtrované pomocí kombinace feritového korálku a kondenzátoru. Napájení je chráněno obousměrnou TVS diodou proti ESD.



Obr. 3.5: Zapojení SPI sběrnice přes obousměrný buffery.

3.5 Sběrnice I²C

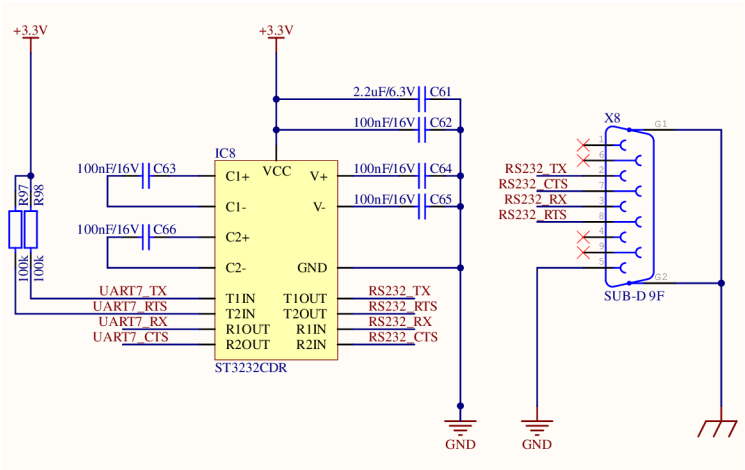
I²C sběrnice je navržená tak, aby bylo možné měnit napěťovou úroveň na výstupu. K tomu jsou využity MOSFET tranzistory v zapojení se společným gate, viz. obrázek 3.6. Datová i hodinová linka je připojena na tranzistory. Na straně sourcu je připojená přes pull-upy interní 3,3 V větev a na stranu drainu je připojené externí napětí, které je vyvedeno na konektor. Pro případ využití 3,3 V výstupu je navržený konektor X15, který připojí externí napájení na interní 3,3 V větev. Externí napájení je filtrované pomocí filtrované kombinací feritového korálku a kondenzátoru. Také je chráněný proti ESD pomocí obousměrné TVS diody.[7]



Obr. 3.6: Zapojení I²C sběrnice.

3.6 Sběrnice RS232, RS485 a UART

Transceiver pro RS232 byl zvolen ST3232CDR. Ten převádí TTL logiku na RS232 logiku. Pro komunikaci jsou využity čtyři signály a to RX (příjem), TX (vysílání), RTS (požadavek na vysílání) a CTS (povolení k vysílání). Mikrokontrolér komunikuje s Transceiverem pomocí UART sběrnice, který má pull-upy na TX a RTS pro zajištění aktivní logické úrovně v době inicializace procesoru. Hodnoty kondenzátorů jsou zvoleny podle dokumentace. Zapojení, včetně připojení signálů na konektor lze vidět na obrázku 3.7 [8]



Obr. 3.7: Zapojení RS232 sběrnice.

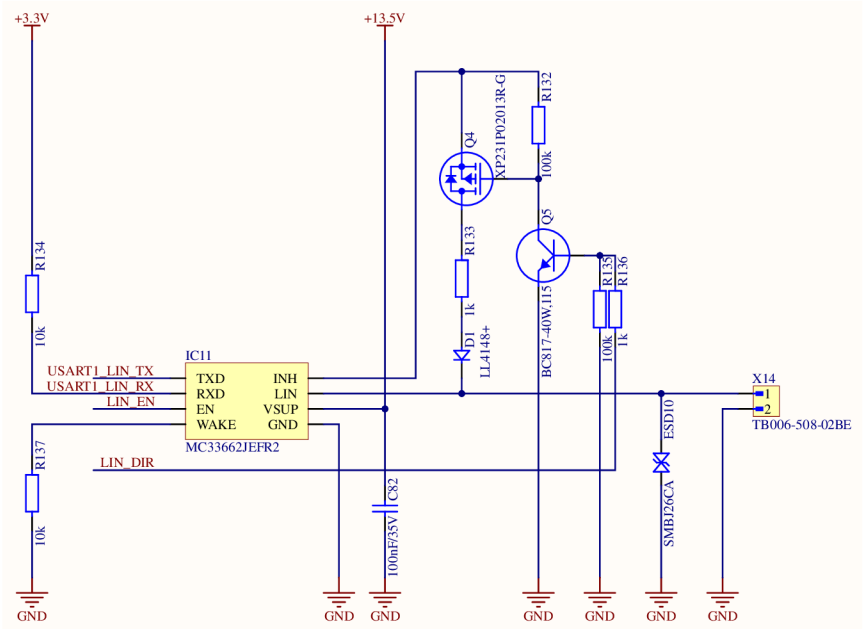
Pro sběrnici RS485 je navržen transceiver LTC2862MPS8-2#PBF. Nicméně z důvodu vyprodání komponenty, byl jako náhrada vybrán kompatibilní transceiver ADM3061EARZ. Mikrokontrolér komunikuje s transceiverem pomocí UART sběrnice. Na transceiver jsou přivedeny tři signály a to dva datové, jeden ovládací. Datovými signály jsou vysílání a příjem, ovládací signál je DE (Driver Enable). Ten slouží k přepínání mezi vysílacím a přijímacím módem tím, že je připojený na oba piny transceiveru DE (Driver Enable) a $\overline{\text{RE}}$ (Receiver Enable). Díky tomu že tyto signály mají opačnou logiku, lze je ovládat jedním signálem. Výstup transceiveru je jeden diferenciální pár. Zařízení komunikuje tzv. “half-duplex“, což znamená pro příjem i vysílání využívá pouze jednu datovou linku. Obdobně jako zapojení pro CAN FD, na výstup RS485 je také možné osadit 33 Ω termistory, nebo filtrem souhlasného napětí. Možné je také pomocí konektoru X7 připojovat pull-up, pull-down a terminační odpor. Výstup je zároveň chráněný ESD ochranou pomocí TVS diod.[9]

Vývod UART sběrnice je obdobně jako u SPI připojený přes integrovaný obousměrný buffer s funkcí dvojího napájení, viz sekce 3.4. Rozdíl je pouze v tom, že tento integrovaný obvod slouží u tohoto zapojení pouze k převodu napěťových úrovní mezi

vstupem a výstupem bez nutnosti měnit směr převodu. Externí napájení lze volit mezi 1 V až 5 V. Pro případ potřeby využití interní 3,3 V větve, napětí lze připojit pomocí konektoru X10. Výstup UART sběrnice je připojený přes ESD ochranu typu USBLC6-2. Externí napájení je filtrované pomocí kombinace feritového korálku a kondenzátoru a chráněno obousměrnou TVS diodou proti ESD.[1]

3.7 Sběrnice LIN

Pro sběrnici LIN je využit integrovaný obvod MC33662JEFR2, který slouží jako fyzická vrstva. Tento integrovaný obvod je k mikrokontroléru připojený pomocí UART sběrnice. Maximální definovaná rychlost tohoto čipu je 10 kbps. Zapojení je provedeno podle dokumentace tak, aby bylo možné přepínat mezi zapojením zařízení jako hlavní a vedlejší uzel. Režim hlavního uzlu je v případě aktivního signálu LIN_DIR. V tom případě je pin INH propojený s LIN výstupem přes 1 kΩ odpor a diodu. Připojení této kombinace odporu a diody je provedeno pomocí MOSFET tranzistoru. Jelikož je na pinu INH vstupní napětí napětí integrovaného obvodu 13,9 V, tak byl pro sepnutí tohoto tranzistoru NPN tranzistor, který je spínaný mikrokontrolérem. Výstup je opatřený ESD ochranou pomocí obousměrné TVS diody.[10]



Obr. 3.8: Zapojení LIN sběrnice.

3.8 Externí paměti

Pro případ potřeby větších bufferů byla zvolena paměť IS42S16800F-7TL. Jedná se o 16 MB SDRAM komunikující na frekvenci až 143 MHz s 16 bitovou datovou sběrnicí, 12 bitovou adresovou sběrnicí a se čtyřmi bankami. Paměť je připojena na FMC sběrnici přes 33 Ω odpory. Ty slouží k ochraně paměti a procesoru v případě nekorektní konfiguraci procesoru. [11]

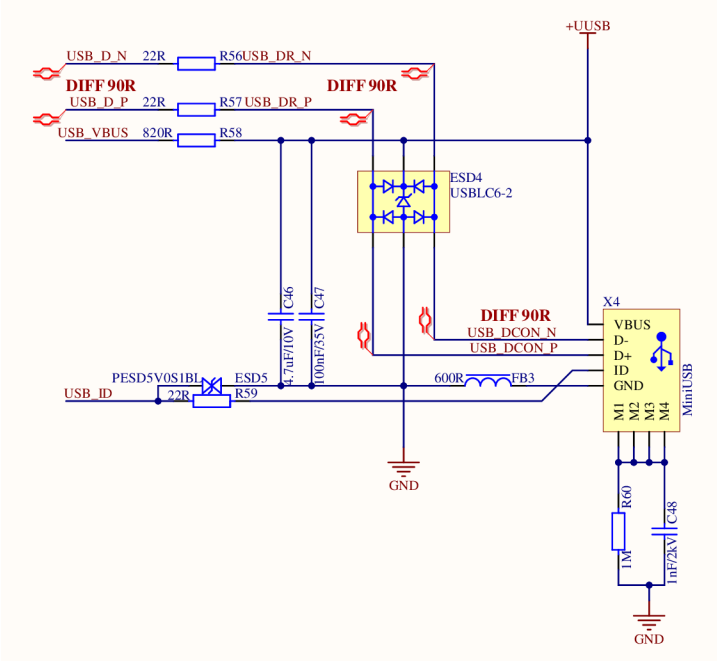
SD karta je navržena pro možnost logování dat, které lze jednoduše přemístit. SD karta je k mikrokontroléru připojena na periferii SDMMC (Secure digital input/output MultiMediaCard interface). Periferie využívá čtyři datové signály, hodinový signál a příkazový signál. Dále je k mikrokontroléru připojena detekce SD karty. Veškeré signálové linky jsou připojena přes integrovaný obvod IP4252CZ16-8-TTL,1. Obvod slouží jako RC dolní propust, čímž se sníží EMI, a také ESD ochrana. Zároveň také poskytuje sériový odpor. Každý signál SD karty je připojený na pull-up. Kostra konektoru je připojena na zem pomocí jednoduchého RC filtru, který slouží jako ESD ochrana. Jelikož bude tato část konektoru zasažena ESD výbojem jako první, měla by daný výboj kompletně zatížit. K tomuto účelu byl zvolen kondenzátor s hodnotou 1 nF zapojení paralelně s 1 MΩ rezistorem.

3.9 USB

USB interface využívá integrovanou periferii mikrokontroléru STM32 s integrovanou fyzickou vrstvou. Tato fyzická vrstva je stavěna pouze pro rychlost splňující standard USB 1.1, tzv. "FullSpeed"s maximální přenosovou rychlostí 12 Mbps. Pro zajištění možnosti použití rychlejší komunikační vrstvy, USB "HighSpeed", by bylo nutné připojit k použitému mikrokontroléru externí fyzickou vrstvu za pomoci osmi-bitové ULPI sběrnice. Z důvodu kolize ULPI sběrnice s jinými nezbytnými periferiemi, jako je například paměťová sběrnice pro externí operační paměť nebo obvod ethernetu, který není plně integrován v použitém mikrokontroléru a je nutno použít externí fyzickou vrstvu.

Z hlediska propojení USB periferie a uživatelského konektoru pro připojení a napájení samotného zařízení je potřeba zajistit definované vedení splňující parametry USB sběrnice. Mezi tyto parametry patří například definovaná impedance daného USB diferenciálního páru, konkrétně 90 Ω. Mimo dodržení této impedance vedení je potřeba zajistit stejné délky jednotlivých vodičů, ale také zajistit jejich ochrany zejména z hlediska ESD a z hlediska možného zkratu na vedení. Za účelem těchto ochran byl na vedení umístěn ESD diodový můstek typu USBLC6-2 (ESD4). Ten zajišťuje ochranu celého diferenciálního páru a napájecího vodiče USB linky. Jelikož je zařízení uvažované i pro možnost využití pro pokročilé připojení, konkrétně typu

OTG, je navíc zapojen i pátý vývod USB konektoru s označením "USB-ID". Tento pin nese samostatnou ESD (ESD5) ochranu zajištěnou obousměrnou TVS diodou s průrazným napětím 5 V. Schéma zapojení lze vidět na obrázku 3.9.[12]



Obr. 3.9: Zapojení USB.

Pro zajištění ochrany zařízení proti zkratu, nebo nekorektní USB konfigurace na straně připojeného zařízení jsou nadále na lince zařazeny sériové odpory s hodnotou 22 Ω. Standard USB dále stanovuje nutnost kontroly hladiny napětí USB linky. Tu je potřeba monitorovat zejména z důvodu možnosti dlouhého vedení s vysokou zátěží, kdy může být komunikace chybová a nestabilní. Za tímto účelem byl na USB interface použitého mikrokontroléru zapojen i VBUS vývod. Napájecí linka USB interface přímo připojená ke speciálnímu analogovému pinu vedenému napřímo k integrované fyzické vrstvě. V rámci ochrany mikrokontroléru je na tomto vedení zařazen sériový odpor.[13]

V rámci filtrování napájení a ochrany před ESD výbojem v době připojení konektoru do USB je zajištěna doplňková ochrana pomocí jednoduchého RC filtru na kostru konektoru samotného. Tato část konektoru bude ESD výbojem zasažena jako první a měla by daný výboj kompletně zatížit tak, aby nezbyl dostatek energie k přenosu na datové piny. K tomuto účelu byl zvolen kapacitor s hodnotou 1 nF zapojení paralelně s 1 M Ω rezistorem. Pro odrušení napájení samotného, je na vývodu 0 V daného USB konektoru připojen feritový korálek tvořící filtr spolu s kondenzátorem C46.

3.10 Ethernet

Ethernetové rozhraní mikrokontroléru STM32 vyžaduje připojení externí fyzické vrstvy. Pro výběr fyzické vrstvy byly uváženy zejména parametry jako potřebná přenosová rychlost, potřebné hardwarové možnosti a také jednoduchost zapojení. Dle těchto parametrů byl vybrán integrovaný obvod typu LAN8742A. Z hlediska vlastností tohoto obvodu byla vybrána sběrnice RMII pro propojení mikrokontroléru a této fyzické vrstvy. Důvodem k tomuto omezení je zejména malé využití plochy na desce a minimalistického zapojení, které neobsahuje třetí datový vývod pro komunikaci nebo další řídicí signály potřebné k tomu, aby bylo možné zapojit jej na MII sběrnici.

Obvod vyžaduje připojení externího zdroje taktovacího kmitočtu 25 MHz, ze kterého je následně vytvořen referenční hodinový signál 50 MHz pomocí integrovaného PLL (Phase-Locked loop). Jako zdroj tohoto kmitočtu byl vybrán kvalitní hodinový oscilátor, který je možné použít zároveň i jako zdroj taktovacího kmitočtu pro hlavní mikrokontrolér. K obvodu je dále potřeba vybrat vhodnou konfiguraci pull-upů a pull-downů, která je díky minimalistickému footprintu volena sdílenými vývody pro indikační LED diody a pro vývod signálu oznamující chybu zařízení (RXERR). Dalšími konfiguračními vývody jsou poté signály pro příjem datového paketu z mikrokontroléru. Jednotlivá konfigurace je zvolena zapojením pull-up nebo pull-down rezistoru na dané sdílené vývody, čímž je stanovena logická úroveň signálu v době při napájecí sekvenci obvodu, kdy je obvod interně konfigurován dle tohoto nastavení. Jednotlivé mody jsou popsány v tabulce 3.1. Zapojením byl nastaven poslední mód, který se nastaví v rámci prvotní komunikace.[15]

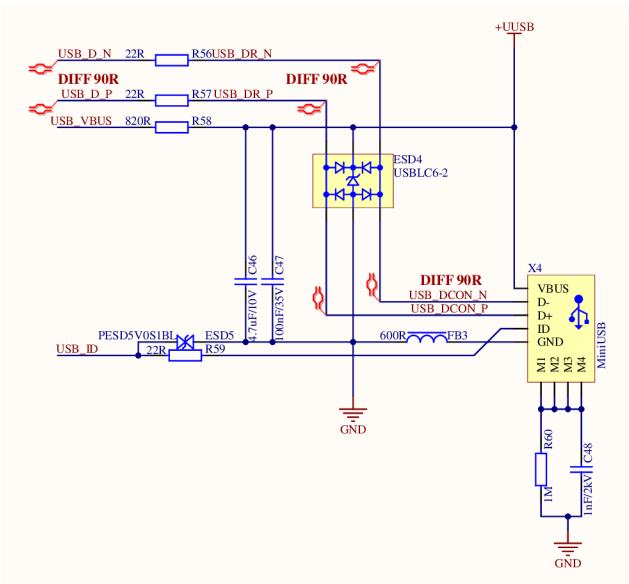
Z hlediska doplnění ochrany a omezení EMI rušení daného zařízení byly do série RMII sběrnice zařazeny sériové rezistory 22 Ω. Další ochrana byla zařazena na datovou sběrnici s diferenciálními páry samotného ethernetového LAN vedení, skládajícího se z celkem dvou datových párů, to znamená s maximální datovou rychlostí 100 Mbps. Použitá ESD ochrana na tomto vedení je typu USBLC6-4 a je zařazena až za vazebním transformátorem. Samostatný vazební transformátor je použit z důvodu jeho absence v použitém ethernetovém konektoru.

V rámci zajištění kompatibility ethernetového interface a zachování korektní impedance daného diferenciálního vedení je zapotřebí korektně přizpůsobit vedení na desce plošného spoje a zajistit jeho stejné délky. Zároveň jsou pro zachování integrity vedení použity i terminační rezistory u samotného fyzického rozhraní, které zajistí zamezení odrazů na vedení při neočekávané komunikaci. Na straně oddělujícího transformátoru jsou poté využity rezistory zapojeny na středech primární strany transformátorů (viz obrázek 3.10 zajišťující přidržení vhodné napěťové hladiny komunikačního signálu.[16] Na straně sekundární jsou středy transformátorového vi-

nutí zapojeny napřímo na napájecí napětí fyzické vrstvy ethernetu, čímž je zajištěna korektní polarita a napěťová hladina příchozího signálu.

Tab. 3.1: Mody konfigurace po resetu

MODE[2:0]	definice modu
0b000	10BASE-T Half duplex bez automatického vyjednávání
0b001	10BASE-T Full duplex bez automatického vyjednávání
0b010	100BASE-TX Half duplex bez automatického vyjednávání, CRS (Carrier Sense) aktivní při vysílání i přijmu
0b011	100BASE-TX Full duplex bez automatického vyjednávání, CRS (Carrier Sense) aktivní při přijmu
0b100	100BASE-TX Half duplex je doporučován s automatickým vyjednávání, CRS (Carrier Sense) aktivní při vysílání i přijmu
0b101	Mod opakovače. 100BASE-TX Half duplex je doporučován s automatickým vyjednávání, CRS (Carrier Sense) aktivní při přijmu
0b110	Režim vypnutí
0b111	Všehoschopný s automatickým vyjednáváním



Obr. 3.10: Zapojení oddělovací transformátoru pro ethernet.

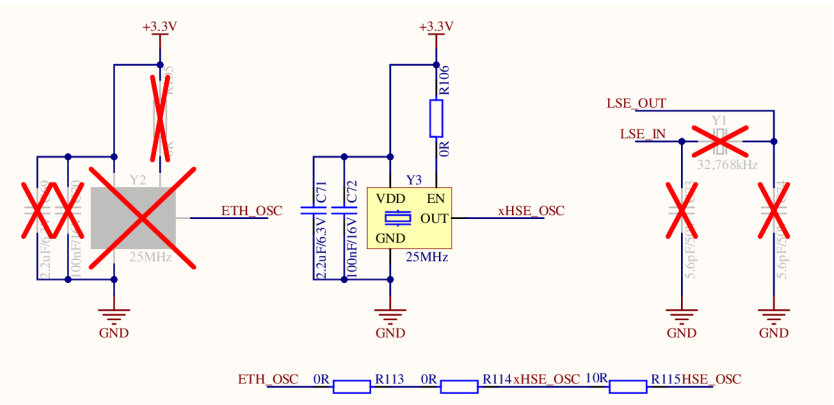
Pro indikaci připojení k internetu jsou zapojeny obě indikační LED diody v použitém konektoru typu RJ45, to je LED dioda indikující připojení aktivního prvku k této přípojce i LED dioda indikující probíhající datovou komunikaci.

3.11 Uživatelské rozhraní, hodiny a restart

Na desce jsou navržena dvě tlačítka. Obě tlačítka jsou zapojena s pull-upy. Jedno tlačítko je uživatelsky programovatelné, druhé tlačítko slouží jako pro restartování mikrokontroléru. Pro kontrolu funkčnosti je na desce navrženo šest LED diod. Tři z toho slouží jako ukazatele funkčního napájení pro jednotlivé napěťové úrovně, které jsou využity na desce. Další tři jsou uživatelsky programovatelná. Každá dioda má vypočítaný odpor tak, aby při jejich úbytku napětí tekl diodou výrobcem doporučovaný proud.

K programování a ladění aplikace je na desce navrženo rozhraní SWD (Serial Wire Debug). SWD rozhraní v základu využívá šesti pinové připojení. V této aplikaci jsou využity pouze čtyři piny a to data, hodiny, napájení a zem. SWD nabízí připojení reset signálu, který lze ovládat pomocí tlačítka. Zároveň není využit signál SWO (Serial Wire trace Output).[17]

Pro zařízení jsou navrženy celkem 3 zdroje hodinového signálu. Jeden nízkofrekvenční 32,768 kHz pro případ potřeby RTC (Real Time Clock), který v je základu neosazený. Dále jsou navrženy dva 25 MHz. Jeden je určený jako hlavní hodiny pro mikrokontrolér s možností propojení pomocí 0 Ω odporů R113 a R114 s hodinami pro ethernet, viz obrázek 3.11. Pro případ velkého zarušení signálu je možnost osadit druhý oscilátor, který je navržený v blízkosti integrovaného obvodu pro ethernet. V případě, že jsou osazeny oba oscilátory, je nutné obvod rozpojit odpojením odporů R113 a R114. Pro možnost osazení oscilátoru s ovládáním je na prvním pinu navržený 0 Ω odpor, který je připojený na napájení. Vynecháním tohoto odporu je možné připojit oscilátory, které mají pin číslo 1 určený jako třístavový a v případě plovoucího připojení jsou aktivní.



Obr. 3.11: Zapojení zdroje hodin pro MCU a ethernet.

3.12 Analogově digitální převodník

Jako analogově digitální (AD) převodníky jsou využity interní převodníky s postupnou aproximací, které pracují v rozsahu 0 až 3,3 V. Připojeny jsou dva různé AD převodníky. První má možnost nastavení rozlišení 8, 10, 12, 14 nebo 16 bitů a maximální nastavitelnou vzorkovací rychlost 3,6 Msps. Druhý má rozlišení nastavitelný na 6, 8, 10 nebo 12 bitů a jeho maximální nastavitelná vzorkovací rychlost je 5 Msps. Každý převodník má využité dva kanály připojený přes multiplexor. Celkově je možno provádět čtyři převody najednou.[1]

Vstupy jsou připojené přes rail-to-rail operační zesilovač, napájený pomocí stejného napětí, které je využito i pro napájení převodníků AD. Operační zesilovač je zapojen jako diferenciální zesilovač s možností uzemnění záporného vstupu za pomoci MOSFET tranzistoru s nízkým odporem kanálu. Lze měřit napětí diferenciálně i vůči zemi.

Aby bylo možné implementovat změnu zesílení řízenou softwarovým příkazem, byly do obvodu zařazeny analogové spínače ADG612YRUZ zapojené ve zpětné vazbě operačního zesilovače. V základu jsou na desce osazeny odpory tak, aby bylo možné volit mezi zesílením 1 nebo 10. Ty lze vyměnit za libovolnou kombinaci odporů pro dosažení potřebného zesílení. Vzhledem k nedostupnosti analogových spínačů v době tvorby práce, implementace přepínatelného zesílení není možná.

Na výstupech operačních zesilovačů jsou zapojeny RC filtry typu dolní propust prvního řádu. Jeho frekvence je nastavená pro základní osazení pomocí odporů R157, R159, R177 a R180 s hodnotou 470 Ω a kondenzátorů C105, C106, C107 a C108 s hodnotou 1 nF. Mezní frekvence tohoto filtru je 338 kHz. Změnou kombinace osazeného odporu a kondenzátoru na desce lze hodnotu filtru měnit podle vlastních potřeb. Tyto filtry mají dvojí využití, kdy slouží k odstranění možného šumu z okolní digitální komunikace a také jako antialiasing filtr.

Pro ovládání přepínání zesílení a uzemnění vstupu operačních zesilovačů je využit posuvný registr M74HC595YTTR. Ten byl zvolen z důvodu nedostatku výstupů mikrokontroléru. K ovládání tohoto posuvného registru je využita SPI sběrnice.

3.13 Digitálně analogový převodník

Obdobně jako AD převodníky jsou využity interní Digitálně analogové (DA) převodníky mikrokontroléru. Ten má možnost využívat dva kanály a to buď nezávisle na sobě, nebo spřaženě. DA převodník integrovaný v mikrokontroléru lze nastavit na osmi-bitové, nebo dvanácti-bitové rozlišení. Výstup je připojen přes napěťový sledovač. Ten slouží jako ochrana mikrokontroléru. [1]

3.14 Vstup-Výstup časovače

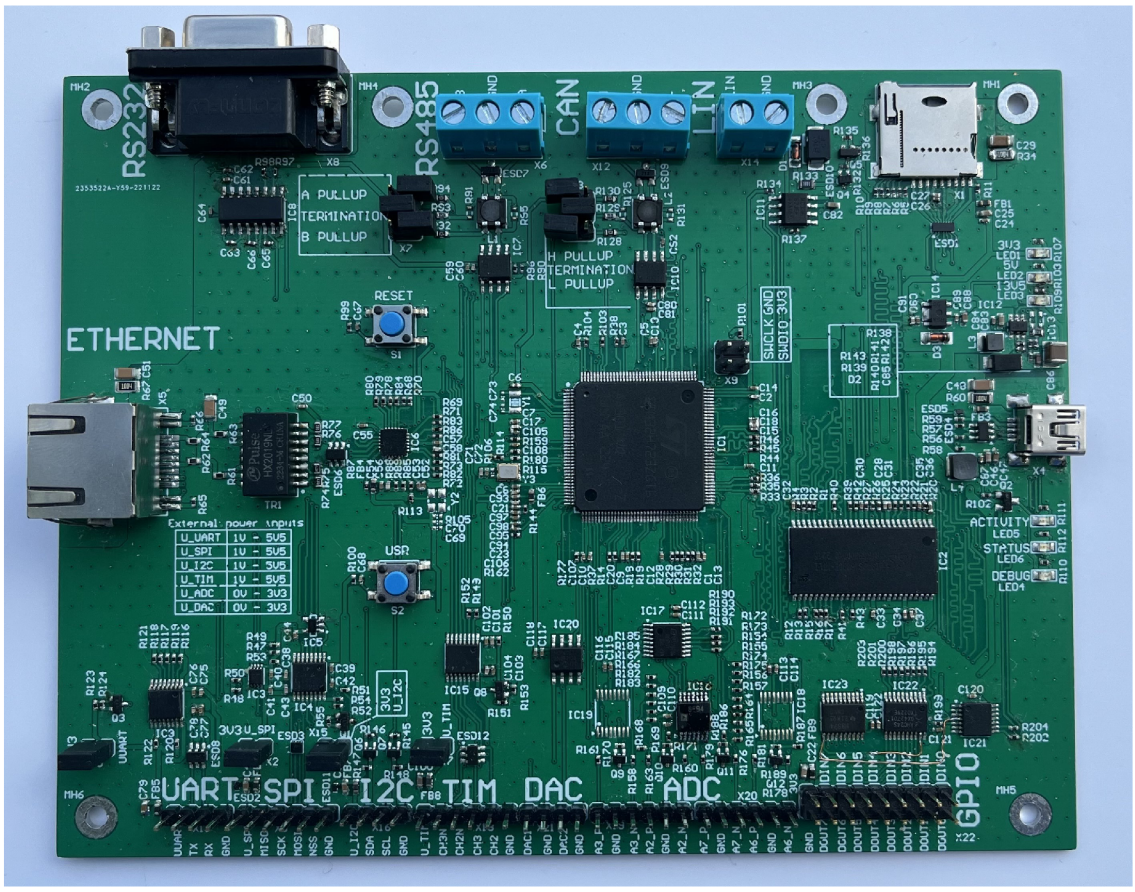
Výstupy časovače jsou určené především k generování PWM (Pulsně Šířková Modulace). Výstupy jsou navrženy tak, aby bylo možné generovat dvě PWM s možností využití i negovaného výstupu.[1]

Obdobně jako u SPI, i zde lze měnit výstupní napětí pomocí integrovaného obvodu 74AXP4T245PWJ, který je popsaný v sekci 3.4. U kladného Vstupu-Výstupu časovače je možné přepínat směr signálu a to právě z důvodu využití časovače například jako enkodér. Externí napájení je filtrované pomocí kombinace feritového korálku a kondenzátoru. Napájení je také chráněno obousměrnou TVS diodou proti ESD.

3.15 Návrh Desky plošného spoje

Součástí práce je návrh desky plošného spoje, která zajistí propojení hlavního mikrokontroléru s jednotlivými fyzickými vrstvami použitých komunikačních periferií a dalších vstupů či výstupů. Deska je zvolena dvouvrstvá s osazenými součástkami na jedné straně. Dvouvrstvá deska je zvolena z důvodu snížení nákladů na výrobu bez větších omezení. Rozměry desky jsou 155 mm na 120 mm. Rozměr delší strany desky je stanoven konektory pro jednotlivé vstupy, výstupy a sběrnice. Kratší strana byla stanovena rozmístěním součástek pro jednotlivé periferie. Vyrobenou a osazenou desku lze vidět na obrázku 3.12.

Návrh plošného spoje s sebou nese jistá omezení a pravidla, vycházející z vlastností a požadavků kladených na dané zapojení. Určitá pravidla, jako je umístění vazebního kondenzátoru nebo použití filtrů, také zajišťují bezpečný provoz zařízení, bezproblémový provoz z pohledu ochrany před okolím (zejména ESD) a také zajištění snížení EMI rušení do okolí. Protože zařízení obsahuje několik vysokofrekvenčních datových komunikací, je nutné dodržet pravidla pro eliminaci EMI rušení z důvodu možného ovlivnění mezi jednotlivými komponenty.[29]



Obr. 3.12: Vyrobená deska plošného spoje.

Mezi kritickými komponenty z hlediska dodržení návrhových pravidel je operační paměť SDRAM. Ta je připojená na sběrnici mikrokontroléru STM32 označenou FMC. Tato periferie je obvykle taktovaná na frekvenci rovné polovině frekvence jádra. U rodiny mikrokontrolérů STM32H7 je kvůli vysoké frekvenci jádra tato periferie připojena na samostatnou jednotku PLL, která generuje samostatné taktovací hodiny. Zvolené taktovací hodiny pro tuto paměť jsou drženy kolem 100 MHz. Takto vysoké frekvence spolu s nutností zařízení přesné posloupnosti aktivních hran signálů s sebou nesou vysoké nároky na layout DPS (desky plošného spoje). Mezi tyto nároky spadá vhodný výběr a rozložení komponent na DPS z důvodu zajištění co nejkratšího spoje mezi mikrokontrolérem a SDRAM pamětí. Správně zarovnané vedení ve smyslu co nejkratších úseků vedení několika signálů v blízkosti sebe. Spolu se zajištěním vhodné mezery mezi vodiči. Dodržením těchto pravidel je eliminováno riziko přeslechů mezi vodiči a jsou eliminovány rizika poškození datového přenosu. Aby bylo zajištěno korektní časování na ovládacích vodičích paměti a zároveň byla zapsaná korektní data do paměti, je potřeba zajistit i správný sled aktivních hran na jednotlivých signálech. K této části slouží jednak správné nastavení FMC perife-

rie v mikrokontroléru pro zajištění správného časování mezi příkazy, daty a adresami pro paměť, ale také shodná délka vedení k paměti s dodržením výrobcem předepsané tolerance maximálně 100 mil.[14]

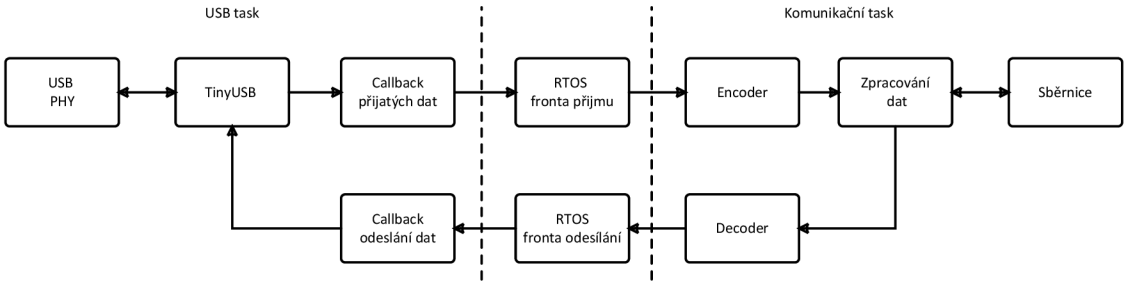
Velice důležitým parametrem k řešení v rámci návrhu DPS je rozvod napájení. K napájení celého zařízení je použito USB připojení, které slouží zároveň pro datový přenos mezi obslužnou aplikací a zařízením. Limit pro napájení z USB je při standardním zapojení 500 mA při 5 V, což je víc jak dostatečné napájení pro obsluhu celé DPS, spolu s napěťovými překladači na některých sběrnicích. [14]

Napětí z USB je zpracováno pomocí řady několika měničů, které pracují v kooperaci. Pro napájení mikrokontroléru a dalších zařízení na 3,3V sběrnici je použit stabilizátor s nízkým úbytkem napětí, zajišťující stabilní pokrytí celé 3,3 V linky a spotřeby na ní. Pomocné napájení pro analogové obvody je zajištěno dodatečnými filtry. Zbývající napěťová linka pro napájení LIN PHY s napěťovou hladinou 13,9 V je zajištěno pomocí malého konvertoru ve zvyšující topologii. Díky kompletně integrovanému řešení tohoto měniče je celé zapojení skládající se z pouze několika komponent, kde je potřeba vzít v potaz návrhové pravidla stanovených výrobcem měniče. Díky základní podstatě spínaného měniče, práce s obdélníkovým signálem s velkou energií a vysokou frekvencí, je potřeba zajistit vhodné okolí měniče, aby nedošlo k zarušení digitálních a analogových vedení na DPS. Potřebné je také zajistit dostatečnou filtraci napájecího vedení, aby se rušivý signál nepřenášel v rámci napájecích linek po DPS. Dané řešení je díky použití pouze dvou vrstev DPS celkem komplexní, zejména z hlediska zajištění minimálního EMI rušení a také z hlediska dodržení požadovaných maximálních impedancí na vedení mezi indukčností a PMIC (Power management integrated circuit) obvodem měniče. Díky nutnosti zajištění nízké impedance na zemním vedení mezi vstupní a výstupní kapacitní bankou a společnou pracovní zemí měniče, je zajištěno i dobré vedení tepla od měniče do volné měděné plochy DPS, která slouží jako chlazení této aktivní komponenty. Chlazení u stabilizátoru s nízkým úbytkem napětí pro 3.3 V sběrnici musí být v kontrastu s tímto zapojením větší a pro její chlazení musí být dedikovaná separátní chladicí plocha.

4 Návrh firmware desky

Firmware desky byl naprogramován využitím jazyka C ve standartu ISO/IEC 9899:1999 (C99) a to za pomoci softwaru Keil μVision verze 5.37. Projekt je strukturován do čtyř hlavních složek. Složka "IDE" obsahuje samotný projektový soubor a knihovny vygenerovaný pomocí softwaru Keil μVision. Další složka "Drivers" obsahuje ovladače periferií napsané pro účel této aplikace. Složka "Project" obsahuje systémové knihovny, například jednotlivé tasky pro RTOS. V poslední složce "Libraries" jsou obsaženy použité knihovny třetích stran, včetně souborů nutných k jejich portování.

Firmware je postavený na základním CMSIS (Cortex Microcontroller Software Interface Standard) standardu a jako operační systém je využitý FreeRTOS[31] (Real Time Operating System). V operačním systému jsou použity tři úlohy (tasky). První task má za úkol jednoduché blikání oranžové (status) LED diody s frekvencí 2 Hz. Účelem tasku je tvořit tzv. hearthbeat. Díky tomu lze identifikovat, zda deska s operačním systémem v provozním stavu. V případě zastavení systému LED dioda přestane blikat. Při používání aplikace může nastat chybový stav jako např. chyba přístupu do paměti vyvolaná MPU (Memory Protection Unit), nebo Hard Fault. Tento stav rozsvítí červenou LED diodu. Druhý task se stará o komunikaci po jednotlivých sběrnicích a třetí task obsluhuje USB. Komunikační a USB task si navzájem předávají data, které jsou zpracována a odeslána podle diagramu na obrázku 4.1.



Obr. 4.1: Vývojový diagram firmwaru desky.

4.1 Systémové a periferní hodiny

Mikrokontrolér STM32H723ZG má k dispozici čtyři interní zdroje hodin a dva externí. Interní oscilátory jsou 64 MHz a 48 Mhz HSI (High Speed Internal oscillator) oscilátor, 4 Mhz CSI (Low-power internal oscillator) oscilátor pro nízkoodběrový aplikace a 32 kHz oscilátor pro hodiny reálného času. Co se týče externích oscilátorů, je zde připojený 25 MHz oscilátor jako HSE (High speed External oscillator). Jako zdroj systémových hodin lze nastavit HSI, CSI, HSE a nebo PLL (Phase Locked Loop). Mikrokontorlér obsahuje celkově 3 PLL z nichž jeden lze využít pro systémové hodiny a zbylé dva jako kernelové hodiny, které slouží jako zdroj hodin pro periferie. Každý PLL má tři výstupy označený jako PLLP, PLLQ a PLLR. Pro každý z těchto tří lze nastavit vlastní frekvenční děličku. Výstupní frekvence lze nastavit pomocí vzorečku 4.1. Kde x značí číslo PLL a y jeho výstup. Hodnota x může být 1 až 3 a y může být P, Q, nebo R. Při nastavování je nutné dbát, aby DIVM vytvořila frekvenci mezi 1 MHz a 16 MHz. V závislosti na požadované frekvenci je nutné nastavit PLL konfigurační registr "RCC_PLLCFGR" podle tabulky 4.1 a to pro každý použitý PLL. Tato frekvence je následně násobená pomocí DIVN, jehož výstup se musí pohybovat mezi 150 MHz až 420 MHz, nebo 192 MHz a 836 MHz. Tento rozsah stanovuje bit "PLLxVCOSEL" v PLL konfiguračním registru. Pro první případ je nastaven na logickou 0, pro druhý na logickou 1. [2]

$$f_{PLLx} = \frac{f_{REF}}{DIVM} \cdot \frac{DIVN}{DIVy} \tag{4.1}$$

Kde

- f_{PLLx} je obecně výstupní frekvence hodinového signálu z PLL,
- f_{REF} je zdroj hodinového signálu pro PLL,
- $DIVM$ je první dělička vstupního hodinového signálu,
- $DIVN$ je násobička hodinového signálu z děličky DIVM,
- $DIVy$ je výstupní dělička hodinového signálu. Každý PLL má tři výstupy (P, Q, R)

Tab. 4.1: Nastavení bitů v PLL konfiguračním registru

PLLxRGE[1:0]	DIVM výstup [MHz]
0b00	1 - 2
0b01	2 - 4
0b10	4 - 8
0b11	8 - 16

Pro korektní práci mikrokontroléru při různých frekvenci je nezbytné nastavení napěťového rozsahu regulátoru v registru "PWR_D3CR" a počet čekacích cyklů

FLASH paměti v registru “FLASH_ACR“. Jejich nastavení je závislé na frekvenci AXI sběrnice mikrokontroléru. Ta je stanovená systémovými hodinami a vstupní děličkou pro první oblast (HPRE). Nastavení čekacích cyklů a napěťového rozsahu lze provést podle tabulky. 4.2. [2]

Tab. 4.2: Nastavení čekacích cyklů a napěťového rozsahu regulátoru

	Frekvence AXI sběrnice při napěťových rozsazích			
Počet čekacích cyklů	VOS3 0,95 - 1,05 V	VOS2 1,05 - 1,15 V	VOS1 1,15 - 1,26 V	VOS0 1,26 - 1,40 V
0	0 - 35 MHz	0 - 50 MHz	0 - 67 MHz	0 - 70 MHz
1	35 - 70 MHz	50 - 100 MHz	67 - 133 MHz	70 - 140 MHz
2	70 - 85 MHz	100 - 150 MHz	133 - 200 MHz	140 - 210 MHz
3	-	-	-	210 - 275 MHz

V této práci jsou využity všechny tři PLL a to s HSE jako vstupní zdroj hodin. První (PLL1) je nastavené jako zdroj systémových hodin. Nastavením DIV1M na 5, DIV1N na 110 a DIV1P na 1 se po dosazení do vzorečku 4.1 dosáhne systémových hodin 550 MHz, což je nejvyšší povolená provozní frekvence uvedená výrobcem. Pro tuto frekvenci je nastavená dělička HPRE na 2. Frekvence AXI sběrnice je nastavená na frekvenci 275 MHz, škálování napětí na režim 0 a počet čekacích cyklů FLASH paměti na 3. Hodnoty jsou určeny podle tabulky 4.2. Výstupní frekvence z děličky DIV1M je 5 MHz a výstup z děličky DIV1N je 550MHz, v konfiguračním registru pro PLL (RCC_PLLCFGR) je nastaven bit “PLLxVCOSEL“ na logickou 1 a bity “PLL1RGE[1:0]“ na 0b10. Nastavení jednotlivých děliček lze vidět v tabulce 4.3.

Tab. 4.3: Nastavení jednotlivých děliček pro PLL

PLL	DIVM	DIVN	DIVP	DIVQ	DIVR
PLL1	5	110	1	5	2
PLL2	25	180	2	2	1
PLL3	25	200	2	2	2

Z důvodu toho, že každá periferie má nějaký zdroj hodin, je třeba znát přesnou frekvenci tohoto zdroje. Mikrokontrolér STM32H723 má možnost konfigurace kernelových hodin, což znamená že každá periferie má možnost výběru z více zdrojů hodin pro každou periferii. Jednotlivé periferie jsou nastaveny podle tabulky 4.4. Pro nastavení hodin slouží funkce “sys_init“ z knihovny “SYS_Lib.h“.

Tab. 4.4: Nastavení zdroje hodin pro jednotlivé periferie

Zdroj hodin	f [MHz]	Periferie
PLL2P	90	SPI1, SPI2, SPI3
PLL2Q	90	FDCAN SPI4, SPI5. SPI6
PLL2R	180	FMC, SDMMC
PLL3Q	100	I^2C4
PLL3R	100	ADC, I^2C1, I^2C2, I^2C3, I^2C5

4.2 Jednotka pro ochranu paměti

Jednotka pro ochranu paměti, nebo-li MPU (Memory Protection Unit), spravuje přístupové práva procesoru do paměti a její atributy. Lze tak definovat oblasti paměti ke kterým je možné přistupovat a které jsou chráněny před neoprávněnými zápisy nebo čtením. V případě porušení některého pravidla nastaveného pomocí MPU pro danou část paměti, vyvolá přerušení.

Jednotku pro ochranu paměti lze rozdělit buďto do osmi, nebo šestnácti regionů. U každého z nich lze nastavit vlastní pravidla a lze je překrývat. V případě překrytí regionů má vyšší prioritu region s vyšším číslem.
Existují tři běžné typy pamětí:

- Normal memory - umožňuje procesoru efektivně ukládat 1 bajtové, 2 bajtové a 4 bajtové data bez zásahu kompilátoru.
- Device memory - zápisy a čtení z této paměti jsou ve striktně daném pořadí. Tímto je zajištěno správné pořadí registrů.
- Strongly ordered memory - vše se vždy provádí v programově uvedeném pořadí, kdy procesor čeká na konec provádění instrukcí pro čtení či zápis, než provede další instrukci.

Na tyhle typy lze v rámci MPU nastavit jednotlivé části paměti. Podstatnou funkcí MPU je také možnost přidělení oprávnění čtení a zápisu ve dvou módech a to privilegovaný a neprivilegovaný. Také lze nastavit možnost vykonávání kódu z dané části paměti, umožnit cachování, bufferování, či nastavit paměť jako sdílenou, nebo danou část paměti zakázat. Regiony se definují počáteční adresou a velikostí. [26]

V rámci aplikace byl nastaven jeden region jako pozadí obsahující celý adresový prostor a čtyři regiony pro které je nevyhovující nastavení jednotného regionu. Jednotlivé regiony jsou popsány v tabulce 4.5. Pro nastavení hodin slouží funkce MPU_init z knihovny SYS_Lib.h.

Tab. 4.5: Nastavení MPU

Oblast	Vykonávání kódu	Cachování	Bufferování	Sdílení	Přístup	Typ paměti
Background	Ano	Ne	Ne	Ano	Žádný	Strongly ordered
DTCM RAM	Ne	Ne	Ne	Ano	Čtení/Zápis	Normal
Interní SRAM	Ne	Ano	Ne	Ano	Čtení/Zápis	Normal
FMC	Ano	Ne	Ano	Ano	Čtení/Zápis	Device
Externí SDRAM	Ne	Ne	Ne	Ne	Čtení/Zápis	Normal

4.3 FMC sběrnice

FMC sběrnice slouží k ovládání externích sériových pamětí. Aplikace má definované dvě externí paměti, externí SDRAM a digitální vstupy/výstupy který se v rámci procesoru chovají jako SRAM.

Digitální vstupy/výstupy jako SRAM paměť

Pro obsluhu digitálních vstupů/výstupů slouží knihovna "SRAM_Lib.h". Nastavení SRAM pro potřeby digitálních vstupů/výstupů je třeba povolení sběrnice. K tomu slouží funkce "SRAM_Init". Pro zápis digitálních výstupů je definována funkce "SRAM_Write", která má jako vstupní parametr osmi bitové číslo, kde každý bit reprezentuje jeden výstup. Obdobným způsobem funkce "SRAM_Read" navrací osmi bitové číslo reprezentující digitální vstupy.

SDRAM paměť

Externí SDRAM má striktní nastavení parametrů a časování, které je nutné dodržet. Mezi důležité nastavitelné parametry patří CAS (Column Address Strobe) latence, počet adresových bitů sloupců a řádků, počet bank, šířka datové sběrnice a hodinový signál. FMC sběrnice má hodinový signál nastavený na 180 MHz. Sběrnice FMC má nastavitelný hodinový výstup pro SDRAM ovladač jako polovinu, nebo třetinu. Při nastavení poloviční hodinového signálu je hodinový signál pro SDRAM 90 MHz. Tomu dle datasheetu paměti odpovídá CAS latence rovna 2. Další parametry potřebné pro konfiguraci paměti vyčtené z datasheetu je, že paměť má 4 banky, 12 bitů pro adresaci řádků a 9 bitů pro adresaci sloupců.

Registr pro nastavení časování definuje celkem 7 opožďujících cyklů pro různé operace, které je nutné vypočítat. Pro výpočet těchto cyklů slouží vzorec 4.2. [2]

$$D_x = T_x \cdot F_{CLK} \tag{4.2}$$

Kde

- D_x je počet cyklů potřebný ke korektnímu časování zaokrouhlený vždy nahoru,
- T_x je čas pro danou akci definovaný výrobcem paměti,
- F_{CLK} jsou hodiny FMC sběrnice pro SDRAM, 90 MHz.

Tab. 4.6: Výpočet časování pro SDRAM

Symbol	T_x [ns]	D_x [-]
T_{MRD}	14	2
T_{XSR}	67	7
T_{RAS}	37	4
T_{RC}	60	6
T_{WR}/T_{DPL}	14	2
T_{RP}	15	2
T_{RCD}	15	2

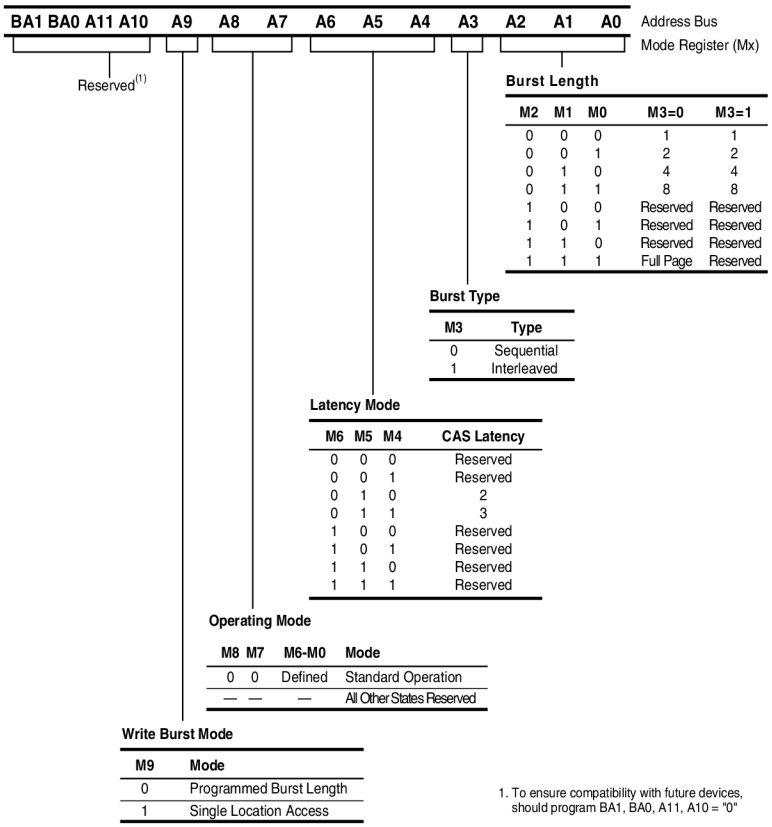
V tabulce 4.6 lze vidět vypočtené časování pro SDRAM. Podle referenčního manuálu musí platit rovnice 4.3 a 4.4. T_{WR} není nutné upravovat, jelikož jsou podmínky splněny.[2]

$$T_{WR} >= T_{RAS} - T_{RCD} \quad 2 >= 4 - 2 \quad 2 >= 2 \tag{4.3}$$

$$T_{WR} >= T_{RC} - T_{RCD} - T_{RP} \quad 2 >= 6 - 2 - 2 \quad 2 >= 2 \tag{4.4}$$

Po nastavení parametrů a časování je potřeba provést start-up sekvenci, která se skládá z pěti kroků. K tomu slouží Command registr “FMC_SDCMR“ s následujícím postupem:

- 1. Nastavení MODE bitů na 0b001 a výběr banky, pro kterou je příkaz určený.
- 2. Čekání podle požadované prodlevy po zapnutí určenou výrobcem. V případě paměti použité v této práci je čas prodlevy alespoň 100 μs
- 3. Nastavení MODE bitů na 0b010
- 4. Nastavení MODE bitů na 0b011 společně s počtem automatických obnov, který je doporučený alespoň 4 obnovy.
- 5. Nastavení MODE bitů na 0b100 a “MRD“ bity, který reprezentují mode registr paměti podle obrázku 4.2.



Obr. 4.2: Mode registr SDRAM paměti.[11]

Mode registr SDRAM paměti je nastavený na číslo 0b0010 0010 0000. To znamená nastavení CAS latence na 2 a zápis bez burstu. Poslední krok pro nastavení SDRAM ovladače je nastavení obnovovacího registru FMC. Ta se vypočítá pomocí vzorce 4.5.[1]

$$COUNT = \frac{64 \cdot 10^{-3}}{2^{raws}} \cdot F_{CLK} - 20 = \frac{64 \cdot 10^{-3}}{4096} \cdot 90^{6} - 20 = 1406,25 \qquad (4.5)$$

Počáteční adresa SDRAM paměti v rámci adresového prostoru je 0xC0000000. Aby nebylo nutné rozkládat v rámci projektu adresový prostor paměti manuálně, je možné nadefinovat její adresový prostor do scatter souboru. Linker kompilátoru je schopen pracovat i s oblastí nadefinovanou v scatter souboru.

Ve výpisu 4.1 lze vidět nastavení SDRAM paměti do scatter souboru. Přidáním řádků 15 až 18 byl nadefinován adresový prostor ".ram_ext", který lze využít jako atribut pro zakládání proměnných podle příkladu z výpisu 4.2. Definice tohoto atributu lze nalézt v souboru "SDRAM_Lib.h", kde je také funkce pro inicializaci FMC sběrnice. Pro možnost vytváření globálních proměnných s nastavením jejich hodnot, funkci pro inicializaci paměti je nutné vykonat před inicializací BSS (Block Started

by Symbol), kde se inicializují proměnné na jejich definovaný stav a neobsahuje tak náhodné data z paměti. [27]

```
LR_IROM1 0x08000000 0x00100000  {    ; load region size_region
  ER_IROM1 0x08000000 0x00100000  {  ; load address = execution
   address
    *.o (RESET, +First)
    *(InRoot$$Sections)
    .ANY (+RO)
    .ANY (+XO)
  }
  RW_IRAM1 0x20000000 0x00020000  {  ; RW data
    .ANY (+RW +ZI)
  }
  RW_IRAM2 0x24000000 0x00050000  {
    .ANY (+RW +ZI)
  }
  ; EXTERNAL SDRAM
  RAM_EXT 0xC0000000 0x1000000
  {
    .ANY (.ram_ext)
  }
}
```

Výpis 4.1: Nastavení SDRAM paměti do scatter file

```
#define __RAM_EXT __attribute__((aligned(4),__section__(".ram_ext")
   ))
__RAM_EXT uint8_t Buffer[10000];
```

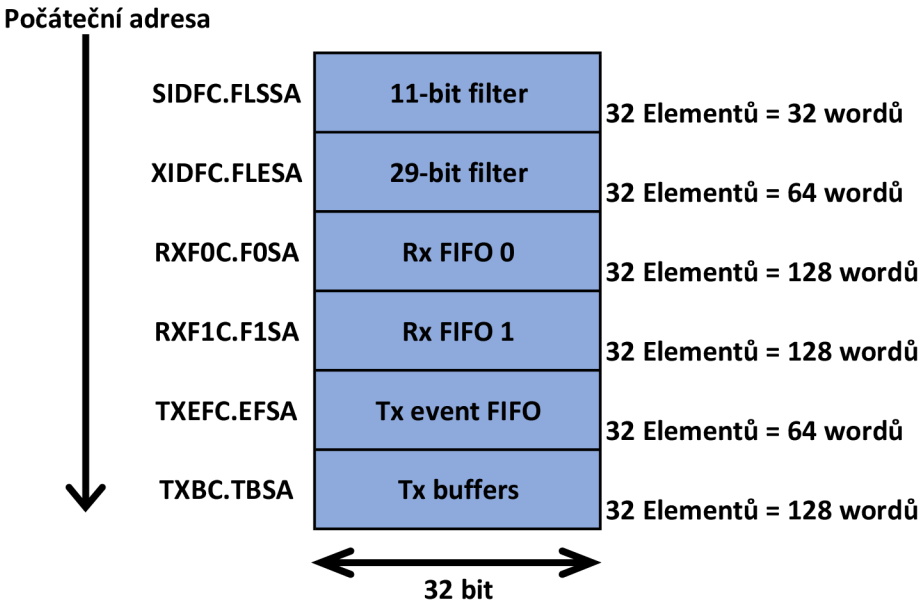
Výpis 4.2: Zakládání proměných v externí paměti

4.4 CAN sběrnice

V rámci knihovny pro CAN “FDCAN_Lib.h“ jsou implementovány funkce pro potřeby CAN 2.0. Knihovna je psaná způsobem, aby byla možná jednoduchá úprava pro podporu CAN FD. Inicializace CAN periferie pomocí funkce “FDCAN_init“ má definovaný jeden vstupní parametr a návratovou hodnotu. Funkce navrátí 0 v případě úspěšného nastavení periferie. Jako vstupní parametr slouží struktura s časováním. Knihovna má definice s předem spočítanými konstantami pro nastavení časování. K vypočítání časových konstant byla využita stránka pro výpočet bitového časování pro sběrnice CAN[28].

Can periferie mikrokontroléru STM32H723 využívá dedikovanou RAM paměť pro obsluhu sběrnice. Do této RAM paměti je nutné pomocí registrů periferie alokovat bloky paměti podle potřeb aplikace.

Rozložení paměti lze vidět na obrázku 4.3. Každý prvek má nastavených 32 elementů. To znamená, že lze například nastavit až 32 různých filtrů. Šířka datové sběrnice této paměti je 32 bitů (v rámci mikrokontroléru označený jako word). Popis jednotlivých elementů včetně vkládání dat k operaci CAN sběrnice lze zjistit v referenčním manuálu [2] na stráně 2556 až 2565. K zapnutí komunikace slouží funkce “FDCAN_Start“ a k jejímu vypnutí slouží funkce “FDCAN_Stop“.



Obr. 4.3: Rozloženi RAM paměti sběrnice CAN.

Odeslání zprávy

Princip odesílání dat po CAN sběrnici s mikrokontrolérem STM32H723 je přidání zprávy do odesílací fronty a předáním indexu místa v paměti pro jeho odeslání. K plnění této paměti slouží funkce “FDCAN_AddTXData“ s návratovou hodnotou 0 v případě úspěšného vložení zprávy k odeslání. Pokud je odesílací fronta plná, návratová hodnota je -1. Funkce má tři parametry. Prvním parametrem je struktura definující hlavičku zprávy. Ta obsahuje ID zprávy, typ ID (standard, nebo extended), a počet bajtů k odeslání. Druhý vstupní parametr je ukazatel na pole s daty k odeslání. Poslední parametr je ukazatel na 32 bitové číslo, který slouží jako návratová hodnota. V této hodnotě je nastavený bit jehož pořadí určuje index na kterém je zpráva uložená. Tato informace je podstatná ke zjištění, zdali se zpráva již odeslala, nebo k jejímu zrušení. K tomu slouží funkce “CAN_IsTherePendingTXMsg“. Ke zrušení odeslání slouží funkce “FDCAN_AbortTXMsg“.

Příjem zprávy

Příjem zprávy je úzce spjatý s filtry. Ty jsou popsány v další části této práce. Mikrokontrolér má definované dvě fronty (FIFO0 a FIFO1). Podle nastavení filtrů se zprávy ukládají do příslušné fronty. Každý datový rámec obsahuje v případě CAN 2.0 protokolu čtyři 32 bitové prvky. Přijatá zpráva je rozdělena na dvě části. První část je hlavička zprávy obsahující informace o přijatých datech (například ID a počet dat). Druhá část se skládá z přijatých dat.

Ve vytvořené knihovně jsou implementovaný dva způsoby detekce přijetí zprávy. První způsob je přes přerušení. Zde je definovaný tzv. "callback", nebo-li funkce která je v přerušení vyvolaná. Uživatel si může tento callback implementovat. Pro povolení přerušení je využitý define "USE_INTERUPT" v "FDCAN_Lib.c" souboru. Druhá metoda spočívá v manuálním kontrolování obou front za pomocí funkce "FDCAN_GetRXFIFOFillLevel". Ta má jako vstupní parametr výčtový datový typ (enum) s definovanými frontami, kterým si uživatel volí kterou frontu kontroluje. Funkce navrací počet zpráv v dané frontě. Vzhledem k využití FreeRTOS, je v aplikaci zvolena manuální kontrola front.

Pro získání zpracovaných dat z paměti slouží funkce "FDCAN_GetRXData". Ta má celkem tři vstupní parametry. První parametr je ukazatel na hlavičkovou strukturu. Do té se uloží ID zprávy, typ ID, kolik dat se přijalo, časová značka a index filtru, pod který zpráva náleží. V případě že zpráva nespadá pod žádný filtr, hodnota indexu je -1. Jako druhý vstupní parametr slouží ukazatel na pole, do kterého budou data uložena. Pole musí být alespoň o velikosti 8 bajtů, aby bylo možné přijmout nejdelší možnou zprávu pro CAN 2.0. Poslední vstupní parametr definuje ze které fronty se má zpráva číst. Funkce má návratovou hodnotu 0 v případě úspěchu a -1 v případě že je fronta prázdná.

Nastavení filtrů

Pro nastavení filtrů jsou v knihovně vytvořeny dvě funkce. První z nich je "FDCAN_GlobalFillterConfiguration". Ta slouží k nastavení chování filtrů a její obsah lze vykonat pouze v případě vypnuté komunikace. Funkce má jeden vstupní parametr a je bez návratové hodnoty. Vstupní proměnou je struktura, která obsahuje nastavení pro 11 bitové filtry (standard filtr) a pro 29 bitové filtry (extended filtr). Nastavení je výčtový datový typ (enum) a definuje chování v případě že ID zprávy neodpovídá žádnému nastavenému filtru. U těchto zpráv lze nastavit, zda budou ignorovány, nebo přijaty s možností výběru do které fronty se data uloží. Dalšími parametry ve struktuře jsou bool hodnoty, které nastavují zda budou přijaty tzv. "remote frames".

```
typedef enum
{
  FDCAN_FilterModeRange         = 0, // Rozsah od ID1 do ID2
  FDCAN_FilterModeDualID        = 1, // Dvě různá ID
  FDCAN_FilterModeIDMASK        = 2, // Druhé ID maskuje první
  FDCAN_FilterModeRange_NO_EIDM = 3  // Pouze extended Rozsah bez maskování
} e_FDCAN_FilterMode;
typedef enum
{
  FDCAN_FilterDISABLE  = 0,        // Disable filter
  FDCAN_Filter_FIFO0   = 1,        // Příjem do fronty 0
  FDCAN_Filter_FIFO1   = 2,        // Příjem do fronty 1
  FDCAN_FilterReject   = 3,        // Odmítnutí zprávy
  FDCAN_SetPriority    = 4,        // Nastaví prioritu
  FDCAN_FilterFIFO0_Priority = 5, // Nastaví prioritu a uloží do FIFO0
  FDCAN_FilterFIFO1_Priority = 6  // Nastaví prioritu a uloží do FIFO1
} e_FDCAN_FilterFIFO;
typedef enum
{
  FDCAN_ID_Standard = 0,
  FDCAN_ID_Extended = 1
} e_FDCAN_ID;
typedef struct
{
  // Typ ID (standard or extended)
  e_FDCAN_ID ID_Type;
  // Akce při přijetí zprávy s ID spadajídí do filtru (e_FDCAN_FilterFIFO)
  e_FDCAN_FilterFIFO FDCAN_FilterFIFO;
  // Výběr modu z e_FDCAN_FilterMode
  e_FDCAN_FilterMode FDCAN_FilterMode;
  // První ID
  uint32_t CAN_FilterID1;
  // Filter Druhé ID
  uint32_t CAN_FilterID2;
  // Filter Index 0-64
  uint32_t Filter_Index;
} t_FDCAN_FilterCFG;
```

Výpis 4.3: Struktura pro nastavení filtrů

Pro nastavení samotných filtrů slouží funkce “FDCAN_ConfigureFilter“. Vstupním parametrem je struktura definovaná ve výpisu 4.3 a návratová hodnota -1 v případě neplatné konfigurace, 0 v případě úspěšného nastavení filtru. První parametr

“ID_Type“ definuje, zda-li má být nastavený 11 bitový, nebo 29 bitový filtr. Pomocí parametru “FDCAN_FilterFIFO“ je nastavena akce pro případ přijmu zprávy se souhlasným ID s konfigurací filtru. Akce k nastavení jsou definované pomocí výčtového datového typu “e_FDCAN_FilterFIFO“. Parametr “FDCAN_FilterMode“ slouží k určení vlastností ID. Každý filtr má definované dvě ID a FilterMode určuje zda se jedná o rozsah mezi nimi, dvě separátní ID a nebo zda druhý ID slouží jako maska k prvnímu. Samotný ID se vkládá do parametrů “CAN_FilterID“. Poslední “Filter_Index“ je index filtru v rámci paměti a při konfiguraci dvou ID na stejný index se první zapsaný filtr přepíše nově nastaveným filtrem.

4.5 Sběrnice UART, RS232, RS485 a LIN

Sběrnice UART, RS232, RS485 i LIN mají v rámci knihovny jednotné ovladače. Pro mikrokontrolér je jediným rozdílem mezi těmito sběrnicemi tzv. “flow control“, neboli řízení datového toku. Pro tyto tři sběrnice komunikuje mikrokontrolér za pomoci UART periferie, která má možnost řídit datový tok podle potřeb pro RS232, RS485 i LIN.

Pro inicializaci slouží funkce “UART_Init“. Ta má jako vstupní parametr ukazatel na periferii, kterou má nastavit. Na desce jsou využité čtyři UART periferie, které jsou definovány v hlavičkovém souboru, viz výpis 4.4. V případě, že ukazatel nebude jeden z těchto definovaných, funkce se ukončí s návratovou hodnotou -1. Druhý vstupní parametr je konfigurační struktura. Struktura obsahuje znakovou rychlost na kterou se periferie nastaví a definuje se v ní řízení datového toku pomocí výčtového datového typu “e_UART_FlowControl“, která je definována ve výpisu 4.4. Periferie UART lze nastavit tak, aby mohl ovládat RTS a CTS pro RS232, či DE (data enable) signál pro RS485. V případě LIN sběrnice se jedná o odeslání ukončovacích znaků. Ovládání těchto signálů je na hardware úrovni mikrokontroléru, není potřeba je ovládat softwarově. Návratová hodnota funkce v případě správné konfigurace je 0.

```
#define UART_CONNECTOR USART3
#define UART_RS232     UART7
#define UART_RS485     USART6
#define UART_LIN       USART1
enum e_UART_FlowControl
{
  UART_FlowControl_None = 0,
  UART_FlowControl_RTS,
  UART_FlowControl_CTS,
  UART_FlowControl_RTSCTS,
  UART_FlowControl_RS485,
```

```
12   UART_FlowControl_LIN
13 };
14 typedef struct
15 {
16   uint32_t BaudRate;
17   enum e_UART_FlowControl FlowControl;
18 } t_UART_Config;
```

Výpis 4.4: Struktura pro nastavení sběrnice UART

Pro odesílání dat je definovaná funkce "UART_Transmit", která má tři vstupní parametry. Stejně jako pro inicializaci, i zde je vstupní parametr ukazatel na periferii. Tímto způsobem lze použít funkci k odesílání dat pro UART, RS232, RS485 i LIN. Jako druhý parametr se předává ukazatel na pole s daty k odeslání. Pro odesílání je potřeba znát počet dat k odeslání, k čemuž slouží třetí parametr funkce. Jako poslední je zadán maximální čas, za který musí být data odeslány. V případě vypršení tohoto času je funkce ukončena s návratovou hodnotou počtu dat, které nebyly odeslány. V případě odeslání všech dat je návratová hodnota 0.

Podobným způsobem jako pro odesílání dat, se volá i funkce pro čtení dat pomocí funkce "UART_Recieve", s rozdílem že do pole se data ukládají. Získávání dat je také možné pomocí neblokující funkce "UART_GetReceivedByte". Ta je především určená pro čtení dat při přerušení a má jako vstupní parametr pouze ukazatel na periferii a ukazatel na bajt, do kterého se přijatá data uloží.

4.6 SPI sběrnice

SPI sběrnice je na desce využita pro dva účely. Hlavní činností je komunikace s připojeným externím zařízení, jako jsou například paměti a snímače, na SPI konektor desky. SPI sběrnice slouží také pro ovládání zesílení AD převodníků. Funkce ke komunikaci po sběrnici jsou obsažené v knihovně "SPI_Lib.h". Knihovna slouží pro ovládání SPI sběrnice jako master. Pro nastavení a ovládání SPI sběrnice je definovaná struktura "t_SPIConfig" podle výpisu 4.5. Struktura obsahuje ukazatel na periferii, kterou bude nastavovat, nebo přes ní bude komunikovat. Zde jsou definované dvě periferie, které jsou na desce použity. Periferie SPI1 je vyvedena na konektor a SPI4 připojena k posuvnému registru pro AD převodníky. Další dva prvky, které struktura obsahuje slouží k ovládání NSS (Slave select). Zde je definovaná struktura obsahující informace na kterém pinu se tento signál nachází. Tato struktura je definovaná v knihovně "GPIO_Lib.h", která je popsaná v sekci 4.11. Prvkem struktury "NSS_Mode", definovaný jako výčtový datový typ "e_SPI_NSS_Setting", kde lze volit způsob ovládání NSS signálu. Možnosti ovládání jsou dvě: automaticky a manuálně. V automatickém režimu je NSS signál ovládaný knihovou a musí

být definovaný pomocí struktury pro GPIO. Knihovna ovládá signál tím způsobem, že na začátku komunikace sepne signál do 0 a na konci zpět do 1. Manuální režim přenechá kontrolu NSS signálu na uživateli. Toho lze využít při potřebě vykonání více různých přenosů bez přepínání NSS signálu. Manuální režim lze také využít v případě využití SPI sběrnice bez použití NSS signálu. Pro tento případ není nutné definovat GPIO strukturu. Poslední dva prvky struktury slouží ke konfiguraci módu podle tabulky 2.2 a k nastavení frekvence hodin podle výčtového datového typu "e_SPI_Speed_Setting".

```
#define SPI_CONNECTOR  SPI1
#define SPI_ADC_SWITCH SPI4
enum e_SPI_ModeConf
{
  SPI_ModeConfig_Mode0 = 0,
  SPI_ModeConfig_Mode1 = SPI_CFG2_CPHA,
  SPI_ModeConfig_Mode2 = SPI_CFG2_CPOL,
  SPI_ModeConfig_Mode3 = SPI_CFG2_CPHA | SPI_CFG2_CPOL
};
enum e_SPI_Speed_Setting
{
  SPI_SpeedSetting_350kHz = 0x07U,
  SPI_SpeedSetting_700kHz = 0x06U,
  SPI_SpeedSetting_1400kHz = 0x05U,
  SPI_SpeedSetting_2800kHz = 0x04U,
  SPI_SpeedSetting_5600kHz = 0x03U,
  SPI_SpeedSetting_11250kHz = 0x02U,
};
enum e_SPI_NSS_Setting
{
  SPI_NSSSetting_Manual    = 0,
  SPI_NSSSetting_Automatic = 1
};
typedef struct
{
  SPI_TypeDef *SPIx;
  t_GPIO_Pin NSS_GPIO;
  enum e_SPI_NSS_Setting NSS_Mode;
  enum e_SPI_ModeConf SPI_ModeConfig;
  enum e_SPI_Speed_Setting speed;
} t_SPIConfig;
```

Výpis 4.5: Struktura pro nastavení a ovládání sběrnice SPI

Struktura z výpisu 4.5 je jediným vstupním parametrem pro SPI inicializaci pomocí funkce "SPI_Init". Návratová hodnota funkce je 0 a to v případě úspěšné konfigurace sběrnice. Funkce "SPI_MasterTransmitReceive" slouží k přijmu a ode-

sílání dat. Jako vstupní parametry má kromě ukazatele na konfigurační strukturu také dva ukazatele na pole s daty k odeslání a příjmu. Dále se funkci předává počet bajtů k odeslání a přijmu. Poslední vstupní parametr je časový limit, po který může komunikace blokovat běh programu. Návratová hodnota funkce je 0 v případě úspěšného dokončení komunikace, nebo -1 v případě že nebylo možné odeslat či přijmout všechny požadované data v určeném časovém limitu. Pro případ potřeby pouze vysílání, lze použít funkci "SPI_Transmit".

4.7 I²C sběrnice

Ovladače pro I²C sběrnici jsou obsaženy v knihovně "I2C_Lib.h". Knihovna je psaná pro komunikaci v master módu. Nastavení sběrnice se vykonává pomocí funkce "I2C_Init". Vstupním parametrem funkce lze volit mezi standardní rychlostí 100 kHz a nebo tzv. "fast" rychlostí 400 kHz. Hodnoty registru pro časování jsou vypočteny pomocí programu STM32CubeMX.

Funkce pro odesílání, "I2C_transmit", i funkce pro příjem, "I2C_receive", mají stejné vstupní parametry. První vstupní parametr je osmi-bitová adresa zařízení na sběrnici. Nejméně významný bit adresy se nastavuje v rámci funkcí podle pravidel pro adresaci na I²C sběrnici. V případě odesílání dat je nastavena logická 0, nebo logická 1 pro čtení dat. Dalšími parametry jsou ukazatel na pole dat a počet bajtů k odeslání či přijmu. Poslední parametr je maximální časový limit, po který funkce můžou blokovat běh programu. Funkce mají návratovou hodnotu 0 v případě úspěšného odeslání, nebo přijetí dat. V případě že nepřijde potvrzení od zařízení na sběrnici nebo uplyne časový limit, funkce navrací počet bajtů, které zbývá odeslat.

4.8 Analogově Digitální převodník

V rámci aplikace využitý integrovaný analogově digitální převodníky, nebo-li AD převodníky. Z důvodu využití dvou rozdílných integrovaných převodníků, který mají jiné vlastnosti z hlediska rozlišení a rychlosti vzorkování, jsou pro každý z nich definovaný vlastní funkce v knihovně "ADC_Lib.h". AD převodníky jsou inicializovány pomocí funkce "ADCx_init", kde x značí číslo periferie integrovaného převodníku. Vstupní parametr je konfigurační struktura daného převodníku. Výpis 4.6 ukazuje možnosti nastavení pro integrovaný AD převodník označený číslem tři. V rámci inicializace se provede kalibrace převodníku a změření referenčního napětí. Funkce má návratovou hodnotu 0 v případě úspěšné inicializace převodníku, -1 v případě že nastane chyba periferie.

```
enum e_ADC3_Resolution
{
  ADC3_Resolution_6bit  = 0x03U,
  ADC3_Resolution_8bit  = 0x02U,
  ADC3_Resolution_10bit = 0x01U,
  ADC3_Resolution_12bit = 0x00U
};
enum e_ADC_ClockSpeed
{
  ADC_ClockSpeed_24000kHz = 0x04U,
  ADC_ClockSpeed_12000kHz = 0x05U,
  ADC_ClockSpeed_6000kHz  = 0x07U,
  ADC_ClockSpeed_3000kHz  = 0x08U,
  ADC_ClockSpeed_1500kHz  = 0x09U,
  ADC_ClockSpeed_780kHz   = 0x0AU,
  ADC_ClockSpeed_390kHz   = 0x0BU,
};
enum e_ADC3_SampleRate
{
  ADC3_SampleRate_2cycle   = 0x00U,
  ADC3_SampleRate_6cycle   = 0x01U,
  ADC3_SampleRate_12cycle  = 0x02U,
  ADC3_SampleRate_24cycle  = 0x03U,
  ADC3_SampleRate_47cycle  = 0x04U,
  ADC3_SampleRate_92cycle  = 0x05U,
  ADC3_SampleRate_247cycle = 0x06U,
  ADC3_SampleRate_640cycle = 0x07U,
};
typedef struct
{
  enum e_ADC_ClockSpeed clock_speed;
  enum e_ADC3_Resolution resolution;
  enum e_ADC3_SampleRate chanel3;
  enum e_ADC3_SampleRate chanel6;
} t_ADC3_Config;
```

Výpis 4.6: Struktura pro nastavení AD převodníku číslo 3

Funkce “ADCx_read“ slouží pro vyčtení naměřených dat. Jako vstupní parametr má také konfigurační strukturu, která slouží ke zjištění rozlišení převodu pro standardizaci naměřených dat. Dalšími vstupními parametry jsou dva ukazatele na 16 bitové číslo bez znaménka. Ty slouží k uložení naměřených dat z obou kanálů připojených na převodníky. Jelikož AD převodníky v mikrokontroléru STM32H723 měří data v sekvencích nastavených v registrech, které se poté ve stejné sekvenci navrací z datového registru, funkce je vytvořena tak, že navrací hodnoty obou kanálů. Naměřená data jsou vrácena v jednotkách mV. Posledním parametrem je časový limit v mi-

lisekundách, po který funkce může blokovat běh programu. Funkce má návratovou hodnotu 0 v případě úspěšného vyčtení naměřených dat, nebo -1 při uplynutí časového limitu.

4.9 Digitálně analogový převodník

Obdobně jako AD převodníky, i digitálně analogový (DA) převodníky je využita integrovaná periferie mikrokontroléru s dvěma výstupy. Její obsluha je implementovaná v knihovně "DAC_Lib.h". Pro inicializaci je definovaná funkce "DAC_init", která nemá žádné vstupní parametry, ani návratovou hodnotu. Pro nastavení výstupu na jednotlivé kanály slouží funkce "DAC_SetOut". Ta má jako vstupní parametry číslo kanálu, který může nabývat hodnot 1 nebo 2. Další vstupní parametr je hodnota výstupu převodníku v rozsahu 0 až 3300 mV, která je ve funkci přepočtena na hodnotu zapsanou do registru. Funkce navrací hodnotu 0 v případě úspěšného nastavení výstupu, -1 pro případ výstupního napětí mimo rozsah, nebo -2 v případě zvolení špatného kanálu.

4.10 Generování PWM pomocí časovače

Výstup časovače desky lze nastavit jako generátor PWM. K tomu slouží knihovna "TIMER_Lib.h". Pro inicializaci časovače je definovaná funkce "TIMER_PWMInit", která má jako vstupní parametr frekvenci na kterou bude PWM generovaná. Frekvence je omezená na 15 Hz až 100 kHz a je určená číslem, kdy se resetuje časovač do 0. Střída PWM je určena číslem, při kterém se překlopí hrany. Při generování frekvencí nad 10 kHz se vlivem zmenšeného rozsahu čísel, které musí být menší než je hodnota při které se časovač resetuje. Hodiny čítače jsou nastavené na frekvenci 1 MHz. To znamená že při frekvenci 100 kHz je hodnota restartování čítače 10 a pro střídu lze nastavit čísla 1-10. Z toho vyplývá, že rozlišení nastavení střídy pro frekvenci 100 kHz je 10 %. Jelikož se jedná o 16-bitový časovač, minimální frekvence je omezena maximální hodnotou časovače $2^{16} - 1$ při kterým je výstupní frekvence PWM 15 Hz. Funkce má návratovou hodnotu 0 v případě úspěšného nastavení časovače, nebo -1 pro případ zadání frekvence mimo výše definovaný rozsah.

Pro nastavení střídy slouží funkce "TIMER_PWMDuty". Ta má jako vstupní parametr kanál pro, který je střída určena, a samotná střída v procentech. Jelikož jsou výstupy připojeny na kanály 2 a 3 integrovaného čítače, parametr kanál musí nabývat těchto hodnot. Střída má omezení na 0 až 100 %. Funkce má návratovou hodnotu -1 pro případ zadání střídy mimo rozsah, -2 pro případ zvolení nedefinovaného kanálu, nebo 0 v případě úspěšného nastavení střídy.

4.11 Pomocné ovladače

Pro jednoduchou obsluhu vstupů a výstupů mikrokontroléru, dále jen GPIO (General Purpose Input-Outup), je napsaná knihovna "GPIO_Lib.h". Knihovna pracuje se strukturou, ve který je definovaný port i číslo pinu zvoleného pinu. Výpis 4.7 ukazuje definici pinů, které jsou využity jako ovládací, například LED diody, tlačítko, nebo slave select pro SPI.

```
#define GPIO_PIN_LED_STATUS (t_GPIO_Pin){.GPIOx = GPIOD, .Pin =  3}
#define GPIO_PIN_LED_ACTIV  (t_GPIO_Pin){.GPIOx = GPIOD, .Pin =  6}
#define GPIO_PIN_LED_DEBUG  (t_GPIO_Pin){.GPIOx = GPIOC, .Pin = 13}
#define GPIO_PIN_SPI1_NSS   (t_GPIO_Pin){.GPIOx = GPIOG, .Pin = 10}
#define GPIO_PIN_SPINSS_DIR (t_GPIO_Pin){.GPIOx = GPIOG, .Pin = 13}
#define GPIO_PIN_SPI4_NSS   (t_GPIO_Pin){.GPIOx = GPIOE, .Pin =  4}
#define GPIO_PIN_TIM8_DIR   (t_GPIO_Pin){.GPIOx = GPIOA, .Pin =  8}
#define GPIO_PIN_USR_BUTTON (t_GPIO_Pin){.GPIOx = GPIOB, .Pin =  2}
#define GPIO_PIN_SPI1_DIR   (t_GPIO_Pin){.GPIOx = GPIOB, .Pin =  6}
#define GPIO_PIN_FDCAN_SHD  (t_GPIO_Pin){.GPIOx = GPIOB, .Pin =  7}
#define GPIO_PIN_LIN_EN     (t_GPIO_Pin){.GPIOx = GPIOD, .Pin = 11}
#define GPIO_PIN_LIN_DIR    (t_GPIO_Pin){.GPIOx = GPIOG, .Pin =  2}

typedef struct
{
  GPIO_TypeDef *GPIOx;
  uint8_t Pin;
} t_GPIO_Pin;
```

Výpis 4.7: Struktura pro ovládání vstupů a výstupů mikrokontroléru

V rámci GPIO knihovny jsou k dispozici funkce pro nastavení a ovládání pinů. Pro kompletní nastavení pinů slouží funkce "GPIO_Configure". Její vstupní parametry jsou krom struktury definující pin i počáteční stav na který pin bude nastaven (log. 0, nebo log. 1). Dále mód operace určující zda je pin využit jako vstup, výstup, analogový a nebo je připojený na interní periferii. Pro případ že se jedná o připojení k interní periferii, je potřeba určit o kterou periferii se jedná. K tomu slouží další vstupní parametr. Jedná se o číslo jehož hodnota se určuje podle tabulek v datasheetu[1] na stránkách 72 až 85. Poslední tři parametry určují zda se jedná o push-pull nebo open drain výstup, rychlost spínání a interní připojení pull-upu, nebo pull-downu.

Pro ovládání hodnoty výstupu slouží čtyři funkce. První funkce "GPIO_SetPin" nastaví výstup do logické 1. Pro nastavení výstupu do logické 0 lze využít funkci "GPIO_ResetPin". Pro změnu stavu výstupu lze slouží funkce "GPIO_WritePin", která překlopí aktuální hodnotu na opačnou. Poslední funkce "GPIO_WritePin" má logickou hodnotu, na kterou se pin nastaví jako vstupní parametr. Vyčítání aktu-

ální hodnoty vstupu je provedeno funkcí “GPIO_ReadPin“, která navrací logickou úroveň daného pinu.

Dalším pomocným ovladačem je knihovna pro SysTick. SysTick je integrovaný čítač mikrokontroléru, který lze univerzálně využít na jakémkoliv mikrokontroléru rodiny STM. V rámci knihovny “SysTick_Lib.h“ je implementované jako přerušení s periodou 1 ms, kde se obsluhuje přepínání Free-RTOS systému. Kromě obsluhy systému se zde také inkrementuje proměnná, sloužící k blokujícímu zpoždění běhu programu přes funkci “Delay_ms“. Funkce má jako vstupní proměnou čas, po který má systém zdržet. Krom zpoždění se knihovna také využívá ke zjištění uplynulých časových limitů, a to pomocí funkce “CheckTimeout“. Funkce má vstupní parametry počáteční čas a velikost časového limitu. Návratovou hodnotou je logická 0 v případě, že čas neuplynul a logická 1 v případě, že stanovený čas uplynul. Pro zjištění počátečního času lze využít funkci “GetTick“, která navrací aktuální čas od zapnutí mikrokontroléru.

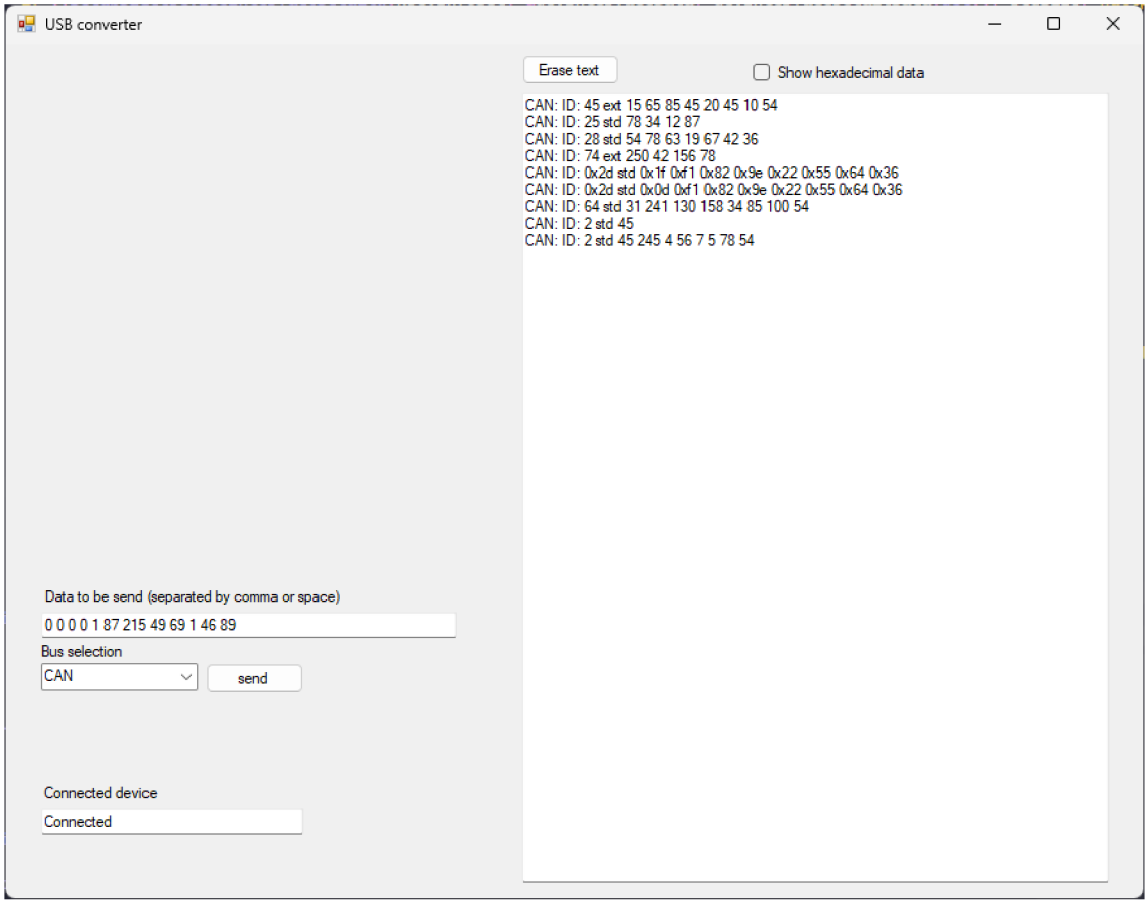
4.12 USB ovladač

Pro obsluhu USB je využitá knihovna v otevřeným zdrojovým kódem Tinyusb[30]. Knihovna byla zvolena na základě její podpory mikrokontroléru STM32h7, jednoduchou a dobře zdokumentovanou implementací do projektu a také pro její podporu operačního systému Free-RTOS. Knihovna se nastavuje konfiguračním souboru “tusb_config.h“. Konfigurace je provedena podle příkladů, které jsou v rámci knihovny k dispozici. Zařízení využívá HID (Human Interface Device) třídu USB knihovny Tinyusb.

Obsluha knihovny Tinyusb je provedena v souboru “usb_task.c“. Zde je definovaný FreeRTOS task, který slouží k periodickému volání obslužní funkce “tud_task“. Na začátku tohoto tasku je provedena inicializace periferie mikrokontroléru a inicializace tinyusb knihovny. Pro inicializaci periferie je nutné pouze povolení hodin pro periferii a povolení přerušení pro USB. V přerušení se poté volá funkce knihovny “tud_int_handler“. K přijímání i odesílání reportů jsou knihovnou vytvořeny callbacky. Callback “tud_hid_set_report_cb“ je vyvolaný při odeslání dat hostem do zařízení. Přijatá data jsou zde překopírována do bufferu komunikačního tasku, kde jsou zpracována. Pro odesílání dat ze zařízení hostovi slouží callback “tud_hid_get_report_cb“. Ten je vyvolaný ve chvíli, kdy host požádá o zaslání dat. Tento callback má jako parametry ukazatel na pole dat a požadovaný počet dat k odeslání. Data se odesílají vyplněním pole dat pro odeslání, a jako návratová hodnota callbacku slouží počet dat k odeslání.

5 Návrh obslužného software

Obslužný software je navržen pomocí integrovaného vývojového prostředí Microsoft Visual Studio 2022 verze 17.5.4. Program je napsaný v jazyce C++/CLI (Common Language Infrastructure), který umožňuje spojování spravovaného kódu (managed code) a nespravovaného kódu (unmanaged code). Spravovaný kód je spouštěn a spravován pomocí .NET runtime. To znamená, že je poskytována automatická správa paměti a další funkce. Nespravovaný kód běží přímo na hardwaru a nemá přístup k funkcím .NET frameworku. Díky spojení je možné vytvářet aplikace v jazyce C++ a zároveň využívat výhod poskytovaných .NET frameworkem. To zahrnuje například snadnou práci s grafickým rozhraním. [32]



Obr. 5.1: Grafické rozhraní pro zobrazování a odesílání dat.

Pro ovládání USB jsou využity knihovny od společnosti Microsoft pro komunikaci s USB HID zařízením. Hlavní část komunikace probíhá v časovači s periodou 1 ms. Zde program kontroluje, zda je deska připojená. V případě že deska není připojená, nebo není inicializovaná komunikace, program se pokouší získat ovladač zařízení (device handler) s konkrétním VID (Vendor ID) a PID (Product ID), které

jsou na desce nastaveny. VID desky je nastavené na 0x4005 a PID na 0xCAFE. Po získání ovladače zařízení a detekci desky jsou v tomto časovači dotazovány data (data polling), zpracovány a zobrazeny v grafickém rozhraní. Příklad přijatých zpráv lze vidět na obrázku 5.1. Uživatel má možnost přepínat zobrazení nových přijatých dat mezi decimálním a hexadecimálním zobrazením.

Tab. 5.1: Tabulka datových paketů po USB pro jednotlivé periferie.

Sběrnice / Periferie	Odesílaná zpráva														
	0	1	2	3	4	5	6	7	8	9	10	11	12	...	62
CAN	0- standard ID 1- extend ID	ID	ID	ID	ID	Data	Data	Data	Data	Data	Data	Data	Data		
I2C	Adresa	Počet dat k přijetí	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	...	Data
Lin	ID	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	...	Data
RS232	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	...	Data
RS485	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	...	Data
UART	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	...	Data
SPI	Počet dat k přijetí	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	Data	...	Data
GPIO	0 - Zápis 1 - Čtení	Data k zápisu													
ADC	1 - Čtení ADC 2	1 - Čtení ADC 3	1 - Čtení ADC 6	1 - Čtení ADC 7											
DAC	1 -Zápis kanál 1	1 - Zápis kanál 2	Data1	Data2											
PWM	1 -Zápis kanál 2	1 - Zápis kanál 3	Střída ch2	Střída ch3											

Data k odesílání se vyplňují v poli “Data to be send“. Data jsou odesílaná jako bajty a jejich rozsah může nabývat 0 až 255. Jednotlivé bajty jsou v poli oddělovány čárkou, nebo mezerou. Data je také možné vložit jako hexadecimální. Vložená data jsou po stisknutí tlačítka “send“ odeslána po USB do desky. Výběr sběrnice nebo periferie je možný pomocí rozbalovací nabídky “Bus selection“. Sekvence dat k odeslání pro jednotlivé periferie je možné vidět v předchozí tabulce 5.1.

6 Obsluha zařízení

Zařízení je napájeno pomocí USB, které zároveň slouží i jako komunikační rozhraní s počítačem. Pro jednoduchou orientaci na desce jsou veškeré komponenty a konektory popsány. Obrázek vyrobené desky lze vidět na obrázku 3.12 v sekci 3.15. Popis obsluhy vytvořeného firmware je obsažený v kapitole 4 a obsluha počítačového software v kapitole 5.

Deska obsahuje tři napěťové úrovně, které jsou indikovány pomocí LED diod. Ty slouží především k jednoduché identifikaci problému se zdrojem napájení jednotlivých komponent. Popisky konektorů značí vstup či výstup mikrokontroléru. To znamená, že pro připojení UART a SPI sběrnice je potřeba připojit signály korektně. To znamená výstupní signál desky na vstupní signál zařízení. K nahrávání programu do mikrokontroléru slouží konektor X9.

Konektory pro sběrnice UART, SPI, I^2C a periferie časovače mají možnost připojení externího napájení určující napěťovou úroveň komunikace. V případě připojení externího napětí je nutné zajistit rozpojení propojovacích svorek příslušného konektoru, který slouží k propojení 3,3 V větve s výstupním napětím. Konektor X10 je určený k propojení napěťových větví sběrnice UART, X2 pro SPI, X15 pro I^2C a X17 pro časovač. V případě připojení externího napětí s propojením na těchto konektorech může dojít ke zničení desky. Vstupní napětí AD převodníků a digitálních vstupů je omezené na 3,3 V.

Konektory pro sběrnice RS485, CAN a LIN mají možnost uzemnění stínění kabelu. V případě připojení uzemnění na straně desky se nesmí připojit na straně zařízení se kterým komunikuje z důvodu možnosti vytvoření zemní smyčky. Konektory X7 a X13 slouží k možnosti připojení terminačních odporů, pull-upu a pull-downu za pomoci propojovací svorky, nebo přidáním externího odporu, na sběrnicích RS485 a CAN.

Závěr

Diplomová práce se zabývala návrhem univerzálního zařízení pro vývoj a testování v oblasti embedded systémů. Cílem této práce bylo vytvořit jediné univerzální zařízení, které zvládne efektivně pokrýt širokou škálu potřeb, které se mohou objevit při vývoji nových systémů. Zařízení obsahuje nejen sběrnice, které bývají standardně využívány při vývoji embedded systémů, jako např. CAN, LIN, RS485, SPI, I^2C, UART. Dále také analogové a digitální vstupy a výstupy. Komunikace s počítačem byla zajištěna pomocí USB, které zároveň slouží jako napájení všech periferií.

V teoretické části práce byly definovány požadavky na komunikační sběrnice. Součástí definic požadavků byl proveden výběr vhodného mikrokontroléru, který byl použit pro realizaci tohoto zařízení. Dále byly popsány použité sběrnice.

Praktické část práce byla věnována návrhu hardwaru desky, která je součástí této práce zdokumentována. Navržená deska byla vyrobena společností JLCPCB. Během testování první revize desky, která byla vyrobena, byla zjištěna chyba spočívající v prohození dvou řídicích signálů pro digitální vstupy a výstupy. Tato chyba byla identifikována a následně na desce opravena. V práci nejsou obsaženy navržené analogové přepínače ADG612YRUZ z důvodu jejich nedostatku na trhu. K přepínačům nebyla nalezena žádná kompatibilní alternativa. Jejich implementace bude možná v rámci návaznosti na tuto práci.

Důležitou částí praktického návrhu bylo vytvořit firmware pro navržený hardware. V rámci práce byly vytvořeny ovladače pro základní periferie mikrokontroléru s možností jejich nastavení. Ovladače slouží k interakci s periferiemi, jako jsou sběrnice (např. CAN, LIN, SPI, I^2C, UART), analogové a digitální vstupy, nebo výstupy. Zmíněné ovladače zajišťují správnou komunikaci mezi mikrokontrolérem a připojenými zařízeními. Nastavení ovladačů bylo vytvořeno tak, aby bylo možné přizpůsobit jejich chování konkrétním potřebám a požadavkům vývoje a testování embedded systémů. V rámci práce jsou ovladače zdokumentovány. Dokumentace zahrnuje popis ovladačů, jejich funkce, možnosti použití a nastavení.

V praktické části diplomové práce byl dále navržen a implementován obslužný software. Tento software představuje grafické uživatelské rozhraní, které umožňuje komunikaci s mikrokontrolérem pomocí USB. Jeho hlavním úkolem je poskytovat uživateli prostředí pro odesílání zpráv na zvolenou sběrnici a zobrazování přijatých či naměřených dat.

Výstupem diplomové práce je funkční zařízení, které komunikuje s vytvořeným software pomocí USB. Firmware zařízení má možnost přizpůsobení požadované aplikace v rámci komunikace po zvolené sběrnici. Komunikaci lze obsluhovat pomocí vytvořeného software. Zařízení je funkčně připraveno na některé aplikace při vývoji a testování v oblasti embedded systémů. Za pomoci navrženého zařízení lze

například uskutečnit vývoj a ladění ovladačů pro bezdrátové moduly typu LTE, WiFi. Zařízení lze také využít pro monitorování komunikace mezi vícero zařízeními po sběrnicích CAN nebo RS485. Vhodné nasazení také může být testování nových modulů pro modulární zařízení, které se ve společnosti DIVELIT system vyvíjí.

Jako navazující krok na práci lze rozšířit funkcionalitu obslužného software tak, aby bylo možné parametry sběrnic nastavit přímo z něj. Také lze rozšířit ovladače sběrnic desky o možnost práce v slave módech a jejich optimalizace na práci s DMA místo současného blokujícího režimu. Aplikaci je možné rozšířit o logování dat na připravenou SD kartu, či možnost využití ethernetu pro komunikaci.

Literatura

[1] *DS13313 Rev. 3, STM32H723 - Data Sheet* [online]. STMicroelectronics, 07.12.2021 [cit. 20. 12. 2022]. Dostupné z URL: <https://www.st.com/resource/en/datasheet/stm32h723zg.pdf>.

[2] *RM0468 Rev. 3, STM32H723/733, STM32H725/735 and STM32H730 Reference manual* [online]. STMicroelectronics, 13.12.2021 [cit. 29. 4. 2023]. Dostupné z URL: <https://www.st.com/resource/en/reference_manual/rm0468-stm32h723733-stm32h725735-and-stm32h730-value-line-advanced-armbased-32bit-mcus-stmicroelectronics.pdf>.

[3] *PM0253 Rev. 5, STM32F7 Series and STM32H7 Series Cortex®-M7 processor programming manual* [online]. STMicroelectronics, 15.6.2019 [cit. 29. 4. 2023]. Dostupné z URL: <https://www.st.com/resource/en/programming_manual/pm0253-stm32f7-series-and-stm32h7-series-cortexm7-processor-programming-manual-stmicroelectronics.pdf>.

[4] *XC6220 Series - Data Sheet* [online]. Torex Semiconductor, 30.06.2022 [cit. 20. 12. 2022]. Dostupné z URL: <https://cz.mouser.com/datasheet/2/760/TOSL_S_A0010556302_1-2575005.pdf>.

[5] *LDO Basics: Preventing reverse current* [online]. Mark Sellers, 25.07.2018 [cit. 20. 12. 2022]. Dostupné z URL: <https://e2e.ti.com/blogs_/b/powerhouse/posts/ldo-basics-preventing-reverse-current-in-ldos>.

[6] *MIC2619 - Data Sheet* [online]. MICREL, 01.03.2010 [cit. 20. 12. 2022]. Dostupné z URL: <https://datasheet.octopart.com/MIC2619YD6-TR-Microchip-datasheet-8819734.pdf>.

[7] *Level Shifting Techniques in I2C-Bus Design* [online]. NXP Semiconductors, 14.11.2012 [cit. 21. 12. 2022]. Dostupné z URL: <https://www.eeweb.com/level-shifting-techniques-in-i2c-bus-design/>.

[8] *ST3232 Rev. 11 - Data Sheet* [online]. STMicroelectronics, 08.02.2008 [cit. 21. 12. 2022]. Dostupné z URL: <https://datasheet.ciiva.com/23807/1733543-23807584.pdf>.

[9] *ADM3073EARZ - Data Sheet* [online]. ANALOG DEVICES, 19.09.2019 [cit. 21. 12. 2022]. Dostupné z URL: <https://cz.mouser.com/datasheet/2/609/ADM3070E_3071E_3072E_3073E_3074E_3075E_3076E_3077E-1503155.pdf>.

[10] *MC33662 Rev.7.0, LIN 2.1 / SAEJ2602-2, LIN Physical Layer - Data Sheet* [online]. Freescale Semiconductor, 20.01.2014 [cit. 21. 12. 2022]. Dostupné z URL: <https://datasheet.ciiva.com/11212/mc33662-11212932.pdf>.

[11] *IS42/45S81600F, IS42/45S16800F - Data Sheet* [online]. Integrated Silicon Solution, Inc., 05.08.2012 [cit. 21. 12. 2022]. Dostupné z URL: <https://datasheet.ciiva.com/15125/getdatasheetpartid-636404-15125564.pdf>.

[12] *Routing Requirements for a USB 2.0 Impedance Interface on a 2-Layer PCB* [online]. Zachariah Peterson, 30.11.2021 [cit. 22. 12. 2022]. Dostupné z URL: <https://resources.altium.com/p/routing-requirements-usb-20-2-layer-pcb>.

[13] *AN4879 Rev.5, USB hardware and PCB guidelines using STM32 MCUs* [online]. STMicroelectronics, 30.5.2022 [cit. 27. 12. 2022]. Dostupné z URL: <https://www.st.com/resource/en/application_note/an4879-usb-hardware-and-pcb-guidelines-using-stm32-mcus-stmicroelectronics.pdf>.

[14] *AN5419 Rev.2, Getting started with STM32H723/733, STM32H725/735 and STM32H730 Value Line hardware development* [online]. STMicroelectronics, 20.3.2020 [cit. 27. 4. 2023]. Dostupné z URL: <https://www.st.com/resource/en/application_note/an5419-getting-started-with-stm32h723733-stm32h725735-and-stm32h730-value-line-hardware-development-stmicroelectronics.pdf>.

[15] *LAN8742A/LAN8742Ai - Data Sheet* [online]. MICROCHIP, 21.5.2013 [cit. 28. 12. 2022]. Dostupné z URL: <https://datasheet.ciiva.com/29882/8742a-29882621.pdf>.

[16] *Bob Smith Termination: Is it Correct for Ethernet?* [online]. Zachariah Peterson, 28.10.2020 [cit. 28. 12. 2022]. Dostupné z URL: <https://resources.altium.com/p/bob-smith-termination-it-correct-ethernet#what-is-bob-smith-termination>.

[17] *JTAG/SWD Interface* [online]. ARM, 03.02.2022 [cit. 28. 12. 2022]. Dostupné z URL: <https://developer.arm.com/documentation/101636/0100/Debug-and-Trace/JTAG-SWD-Interface>.

[18] DAWOUD, Dawoud Shenouda a Peter DAWOUD. *Serial Communication Protocols and Standards*. 1. Aalborg: Routledge, 2020. ISBN 9788770221542. Dostupné z: doi:10.1201/9781003339496

[19] PARET, Dominique. *Multiplexed networks for embedded systems: CAN, LIN, flexray, safe-by-wire..* Hoboken: Chichester: John Wiley, 2007, xiii, 418 s. : il. ISBN 978-0-470-03416-3.

[20] *Introduction to the Controller Area Network (CAN)* [online]. Texas Instruments, 01.05.2016 [cit. 01. 01. 2023]. Dostupné z URL: <https://www.ti.com/lit/an/sloa101b/sloa101b.pdf>.

[21] *CAN with flexible data-rate Specification v1.0* [online]. Bosch, 01.05.2016 [cit. 01. 01. 2023]. Dostupné z URL: <https://can-newsletter.org/assets/files/ttmedia/raw/e5740b7b5781b8960f55efcc2b93edf8.pdf>.

[22] *Introduction to the Local Interconnect Network (LIN) Bus* [online]. NATIONAL INSTRUMENTS, 16.08.2022 [cit. 01. 01. 2023]. Dostupné z URL: <https://www.ni.com/cs-cz/innovations/white-papers/09/introduction-to-the-local-interconnect-network--lin--bus.html>.

[23] *The RS-485 Design Guide* [online]. Texas Instruments, 01.05.2021 [cit. 02. 01. 2023]. Dostupné z URL: <https://www.ti.com/lit/an/slla272d/slla272d.pdf>.

[24] *Serial Peripheral Interface (SPI) Master* [online]. Cypress Semiconductor, 17.05.2012 [cit. 02. 01. 2023]. Dostupné z URL: <https://www.infineon.com/dgdl/Infineon-Component_SPI_V2.30-Software%20Module%20Datasheets-v02_05-EN.pdf?fileId=8ac78c8c7d0d8da4017d0e7bb27e0d3b>.

[25] *Aplication Report SLVA704, Understanding the I^2C Bus* [online]. Texas Instruments, 01.06.2015 [cit. 02. 01. 2023]. Dostupné z URL: <https://www.ti.com/lit/an/slva704/slva704.pdf>.

[26] *AN4838 Rev. 6, Introduction to memory protection unit management on STM32 MCUs* [online]. STMicroelectronics, 7.2.2023 [cit. 29. 4. 2023]. Dostupné z URL:
<https://www.st.com/resource/en/application_note/an4838-introduction-to-memory-protection-unit-management-on-stm32-mcus-stmicroelectronics.pdf>.

[27] *ARM Compiler armlink User Guide Version 5.06* [online]. arm, [cit. 29. 4. 2023]. Dostupné z URL:
<https://developer.arm.com/documentation/dui0474/m/scatter-loading-features/the-scatter-loading-mechanism/overview-of-scatter-loading?lang=en>.

[28] *CAN Bit Time Calculation* [online]. Heinz-Jürgen Oertel, [cit. 30. 4. 2023]. Dostupné z URL:
<http://www.bittiming.can-wiki.info>.

[29] *PCB Design Techniques to Reduce EMI* [online]. Altium, [cit. 30. 4. 2023]. Dostupné z URL:
<https://resources.altium.com/sites/default/files/uberflip_docs/file_731.pdf>.

[30] *Ha Thach. 2023. tinyusb* [online]. GitHub, [cit. 20. 4. 2023]. Dostupné z URL:
<https://github.com/hathach/tinyusb>.

[31] *FreeRTOS Real-time operating system for microcontrollers* [online]. Amazon, [cit. 20. 4. 2023]. Dostupné z URL:
<https://www.freertos.org>.

[32] *C++/CLI pro .NET* [online]. Microsoft, 03. 04. 2023 [cit. 20. 4. 2023]. Dostupné z URL:
<https://learn.microsoft.com/cs-cz/cpp/dotnet/walkthrough-compiling-a-cpp-program-that-targets-the-clr-in-visual-studio>.

Seznam symbolů a zkratek

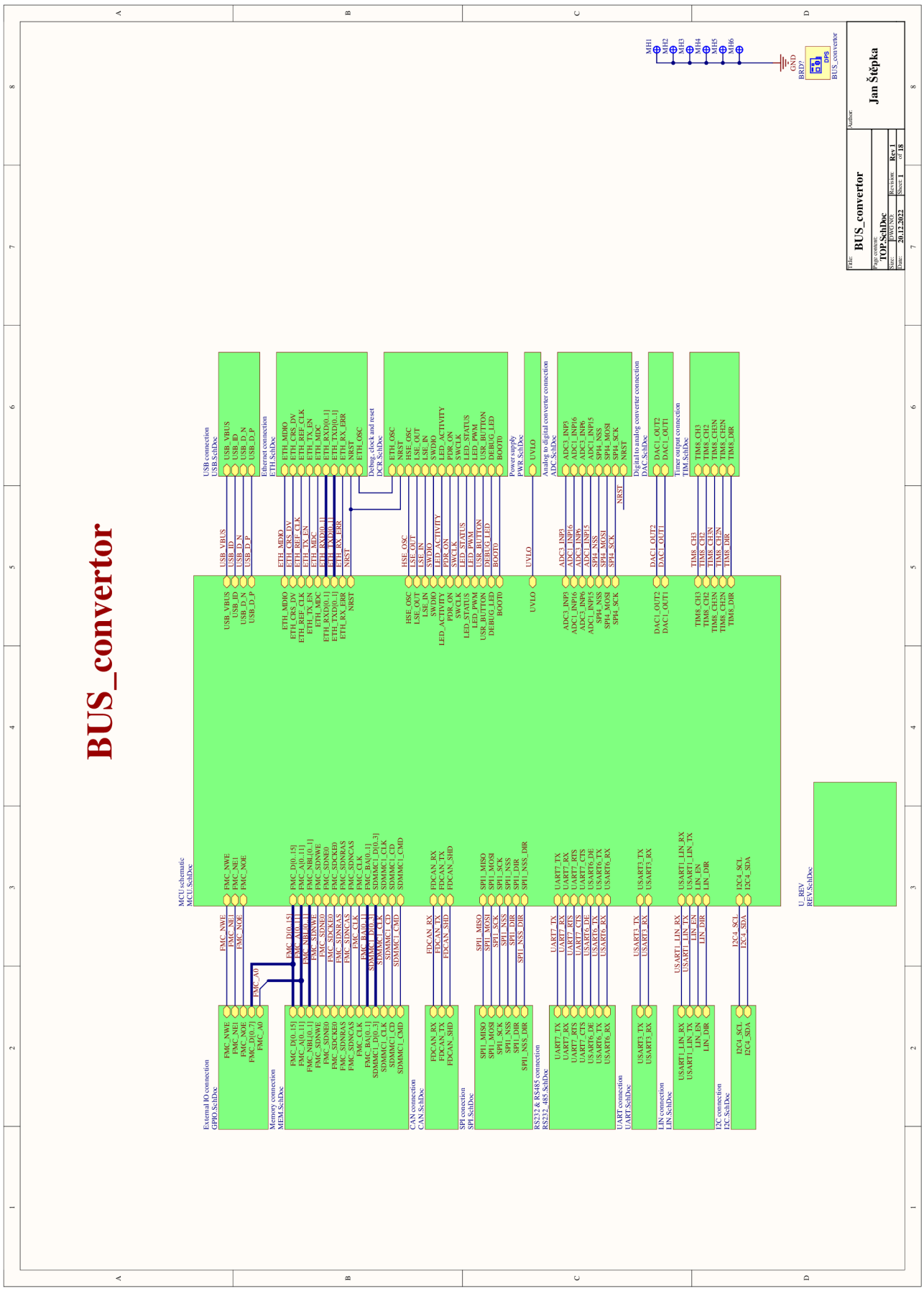
BSS	Block Started by Symbol
CAN	Controller Area Network
CAN FD	Controller Area Network with Flexible Data-Rate
CTS	Clear to Send
DMA	Direct Memory Access
DPS	Deska plošného spoje
EMI	Electromagnetic Interference
ESD	Electrostatic Discharge
FMC	Flexible Memory Controller
HID	Human Interface Device
LIN	Local Interconnect Network
MII	Media-independent interface
MISO	Master Input Slave Output
MOSI	Master Output Slave Input
NSS	Slave Select
PLL	Phase-Locked loop
PMIC	Power management integrated circuit
PWM	Pulse Width Modulation
RMII	Reduced media-independent interfacea
RTS	Request to Send
Rx	Receive Data
SDRAM	Synchronous Dynamic Random Access Memory
SPI	Serial Peripheral Interface
SWD	Serial Wire Debug

SWO	Serial Wire trace Output
TCM	Tightly-Coupled Memory
TTL	Transistor Transistor Logic
TVS	Transient Voltage Suppression
Tx	Transmit Data
UART	Universal asynchronous receiver-transmitter

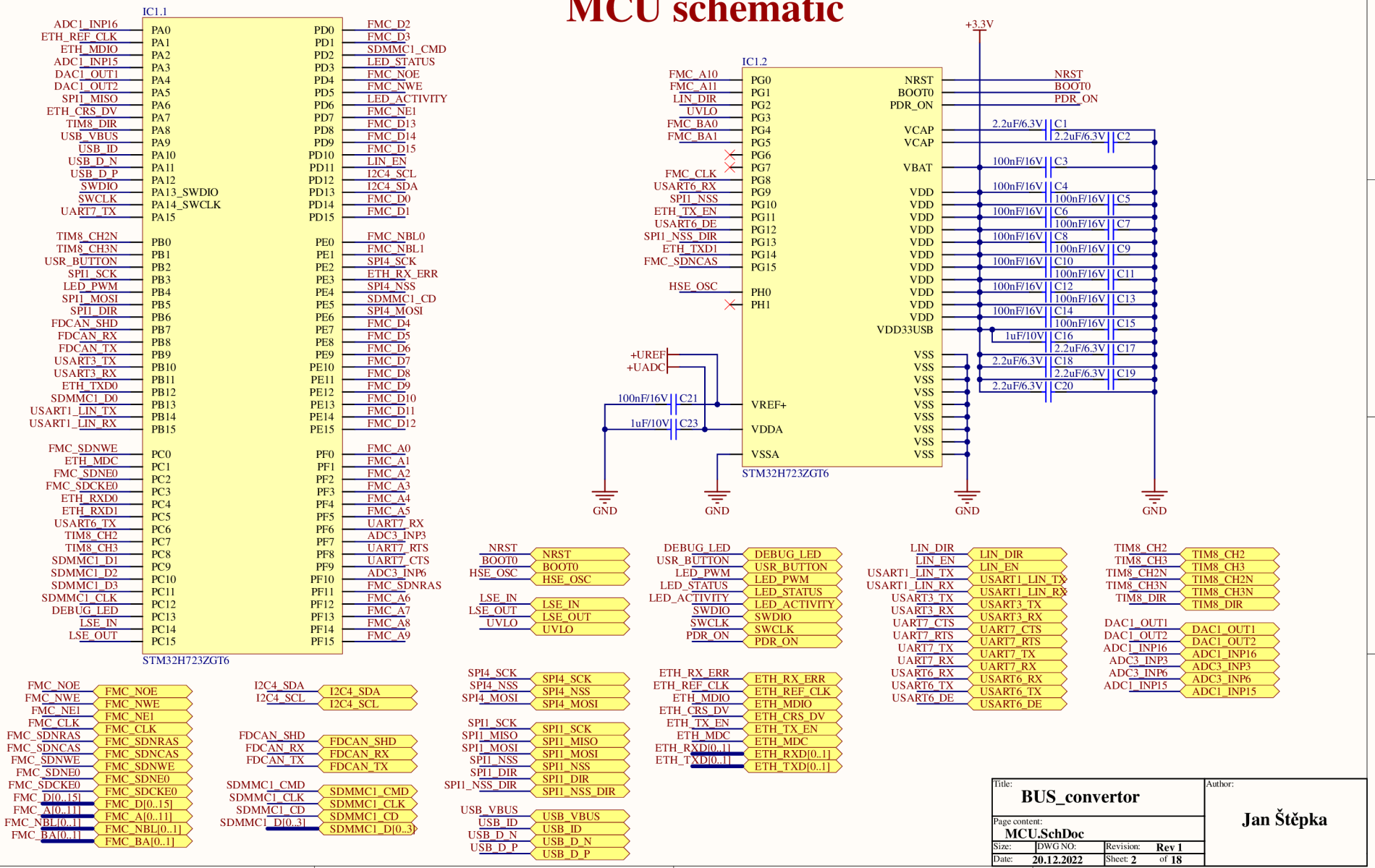
Seznam příloh

A Schéma

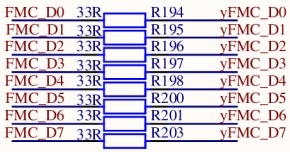
A.1 Schéma - blokové zapojení

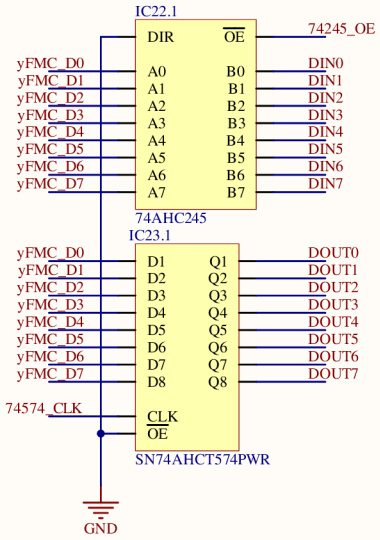


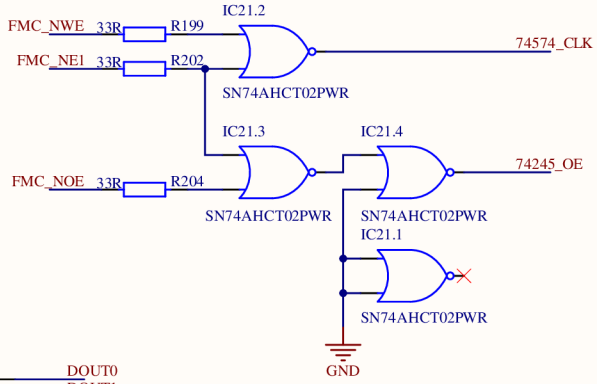
MCU schematic

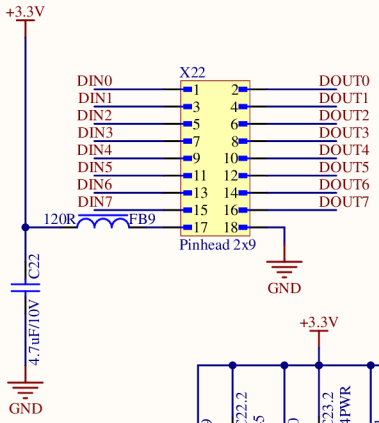


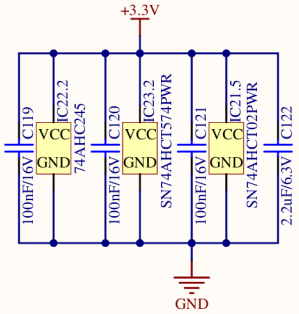
External IO connection

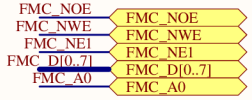








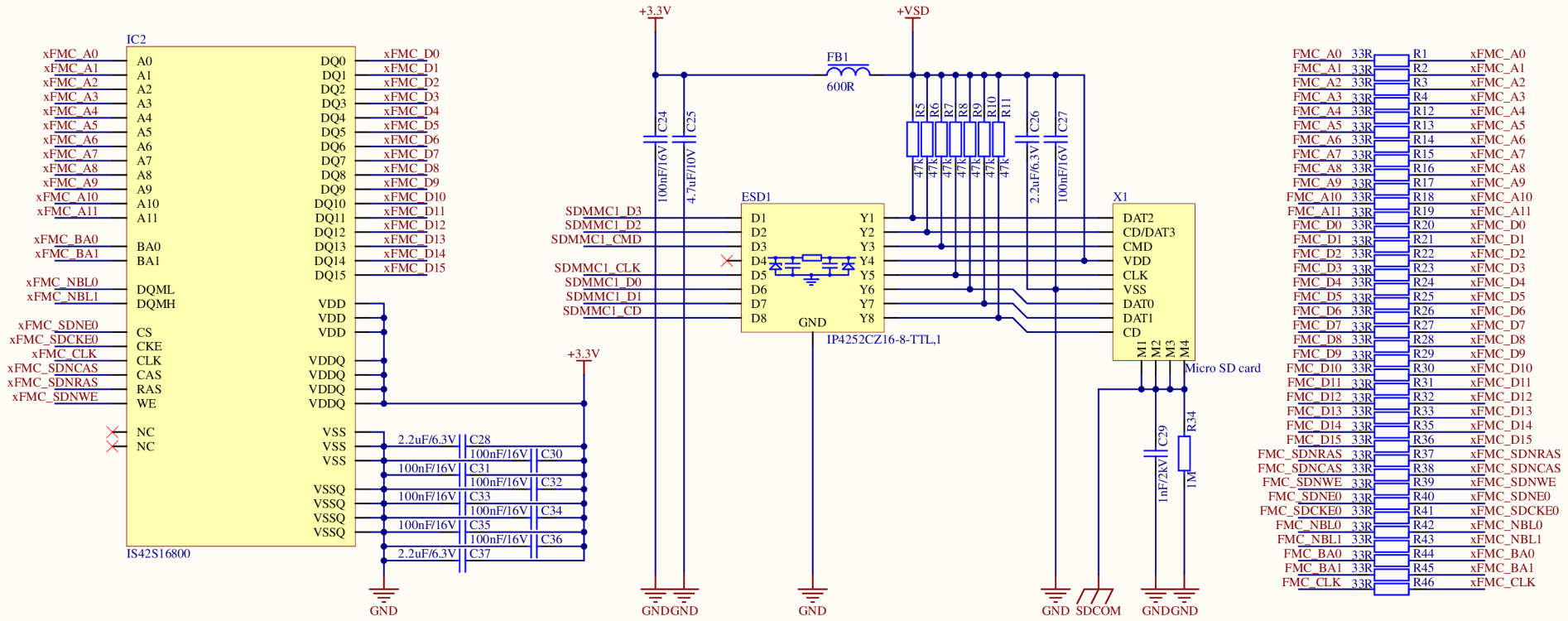




Title: **BUS_convertor**			Author: **Jan Štěpka**
Page content: **GPIO.SchDoc**			
Size:	DWG NO:	Revision: **Rev 1**	
Date: **20.12.2022**		Sheet: **3** of **18**	

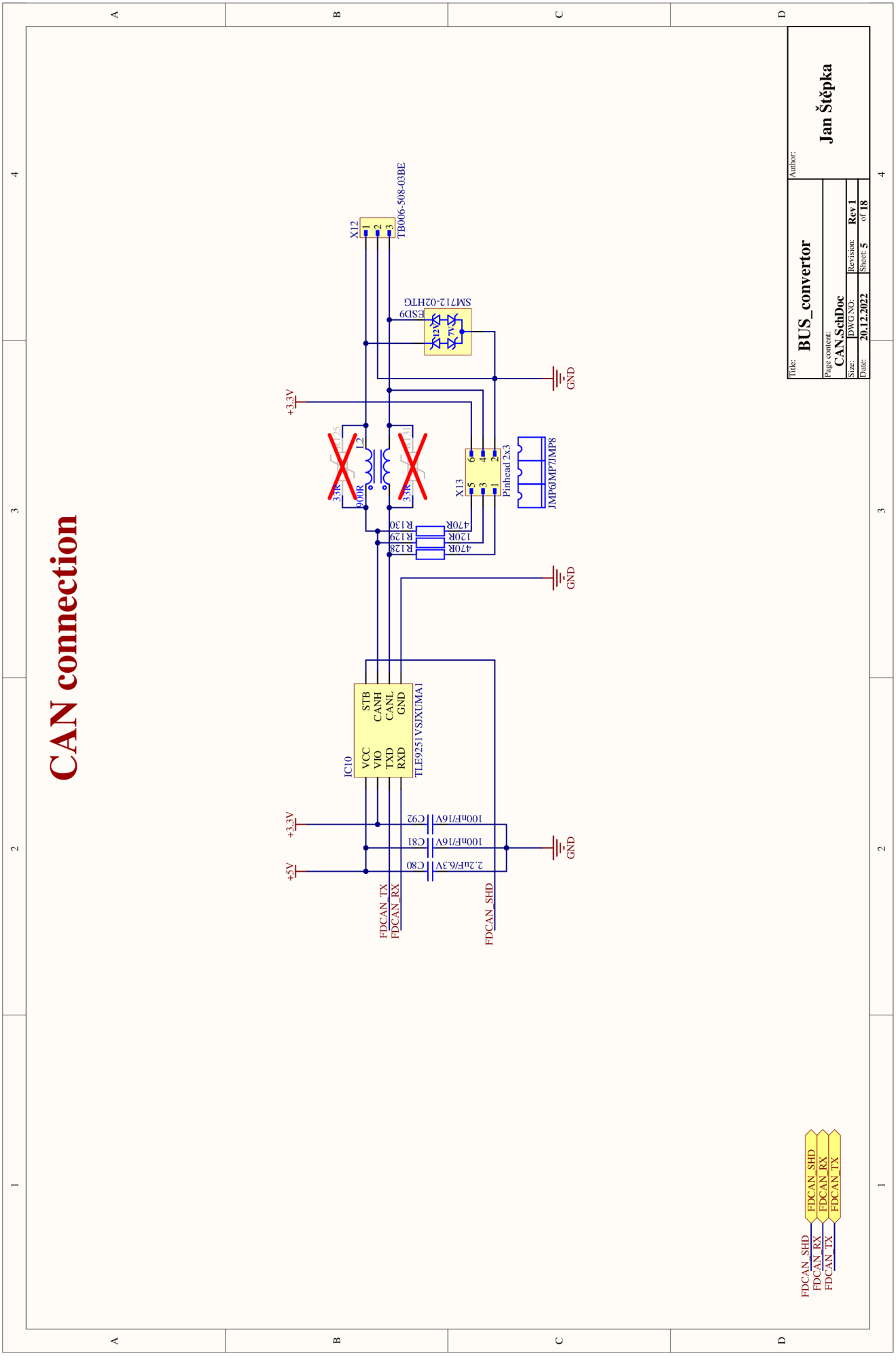
A.3 Schéma - Digitální vstupy-výstupy

Memory connection

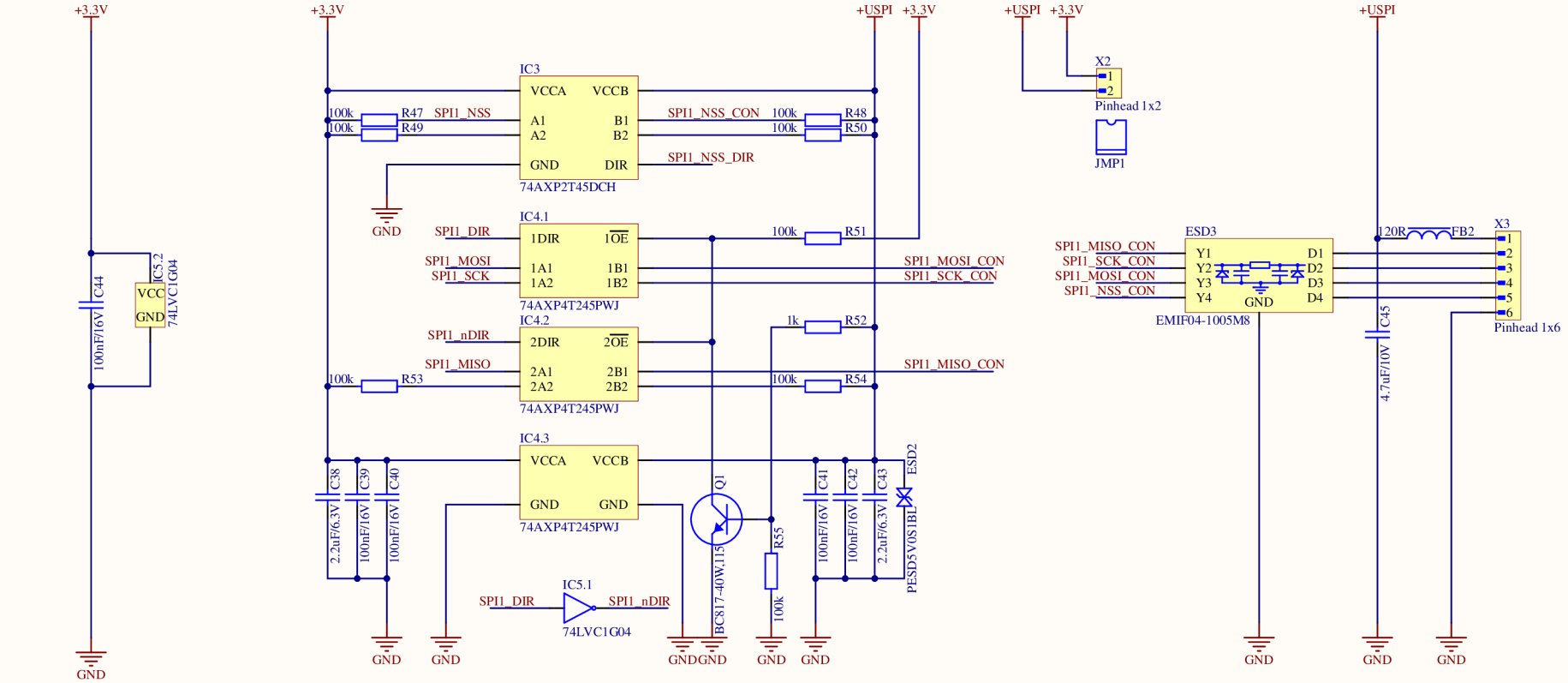


Title: **BUS_convertor**			Author: **Jan Štěpka**
Page content: **MEM.SchDoc**			
Size:	DWG NO:	Revision: **Rev 1**	
Date: **20.12.2022**		Sheet: **4** of **18**	

A.5 Schéma - CAN sběrnice



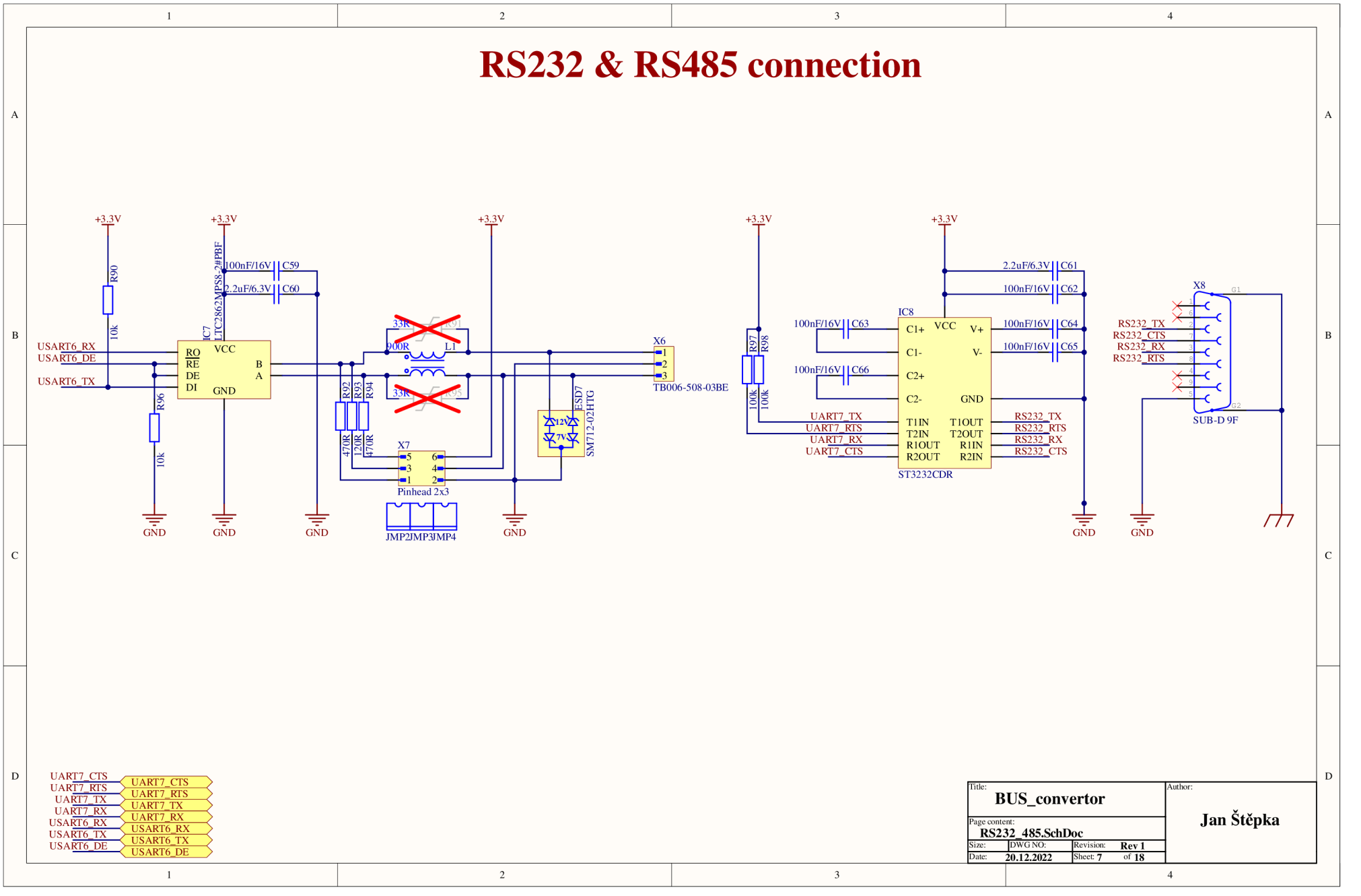
SPI connection



SPI1_SCK
SPI1_MISO
SPI1_MOSI
SPI1_NSS
SPI1_DIR
SPI1_NSS_DIR

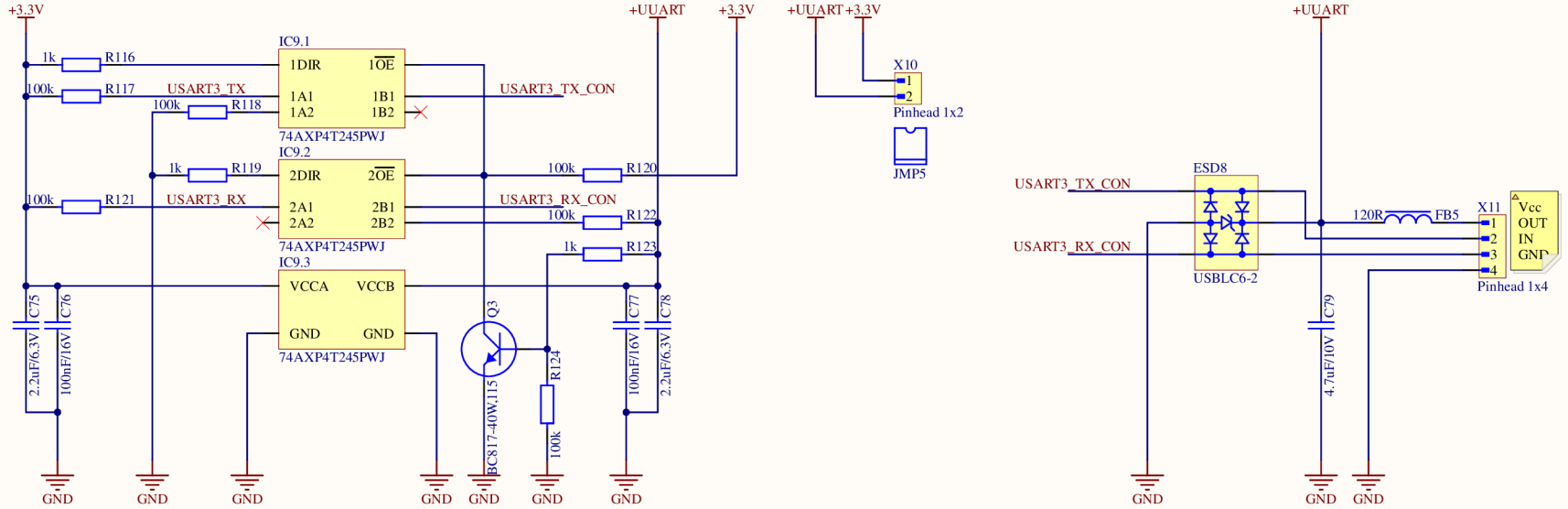
Title:	**BUS_convertor**
Author:	**Jan Štěpka**
Page content:	**SPI.SchDoc**
Size:	DWG NO:
Revision:	**Rev 1**
Date:	**20.12.2022**
Sheet:	**6** of **18**

A.7 Schéma - RS232 a RS485 sběrnice



A.8 Schéma - UART sběrnice

UART connection

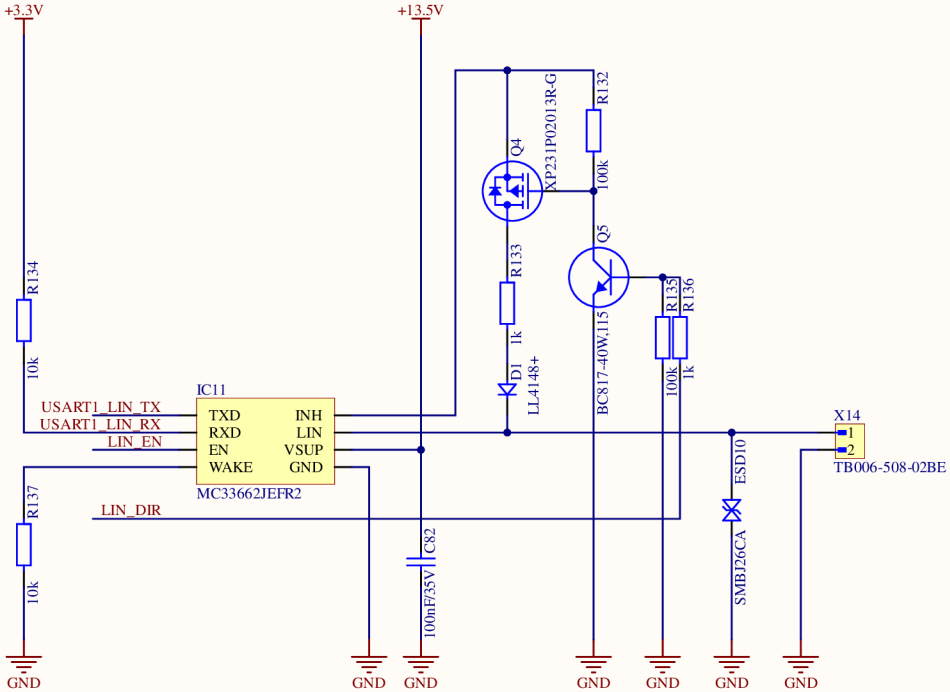


USART3_TX
USART3_RX
USART3_TX
USART3_RX

Title: **BUS_convertor**			Author: **Jan Štěpka**
Page content: **UART.SchDoc**			
Size:	DWG NO:	Revision: **Rev 1**	
Date: **20.12.2022**		Sheet: **8** of **18**	

A.9 Schéma - LIN sběrnice

LIN connection

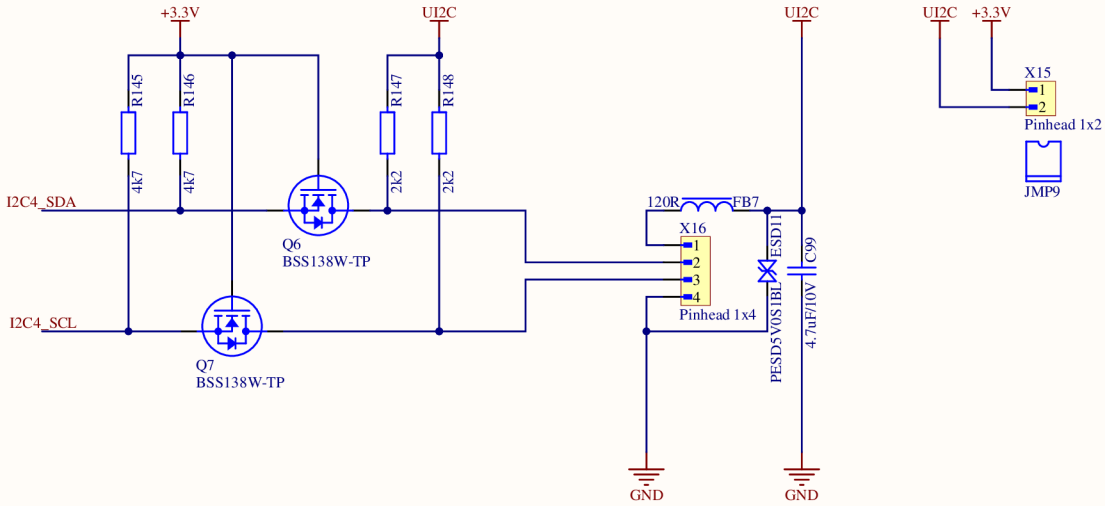


LIN_DIR
LIN_EN
USART1_LIN_TX
USART1_LIN_RX

Title: BUS_convertor			Author: Jan Štěpka
Page content: LIN.SchDoc			
Size:	DWG NO:	Revision: Rev 1	
Date: 20.12.2022		Sheet: 9 of 18	

A.10 Schéma - I²C sběrnice

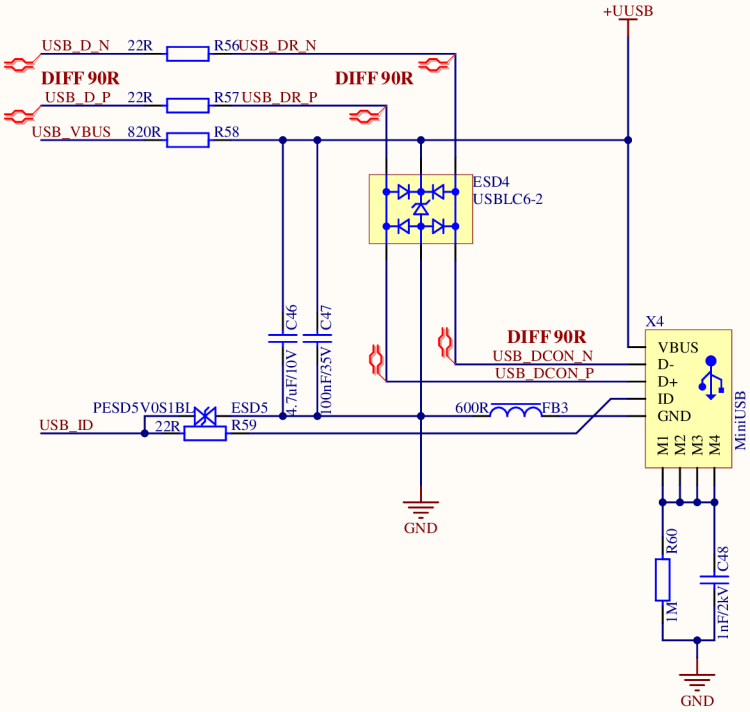
I2C connection

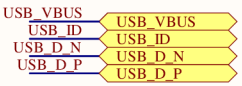


I2C4_SDA
I2C4_SCL
I2C4_SDA
I2C4_SCL

Title: BUS_convertor			Author: Jan Štěpka
Page content: I2C.SchDoc			
Size:	DWG NO:	Revision: Rev 1	
Date: 20.12.2022		Sheet: 10 of 18	

USB connection

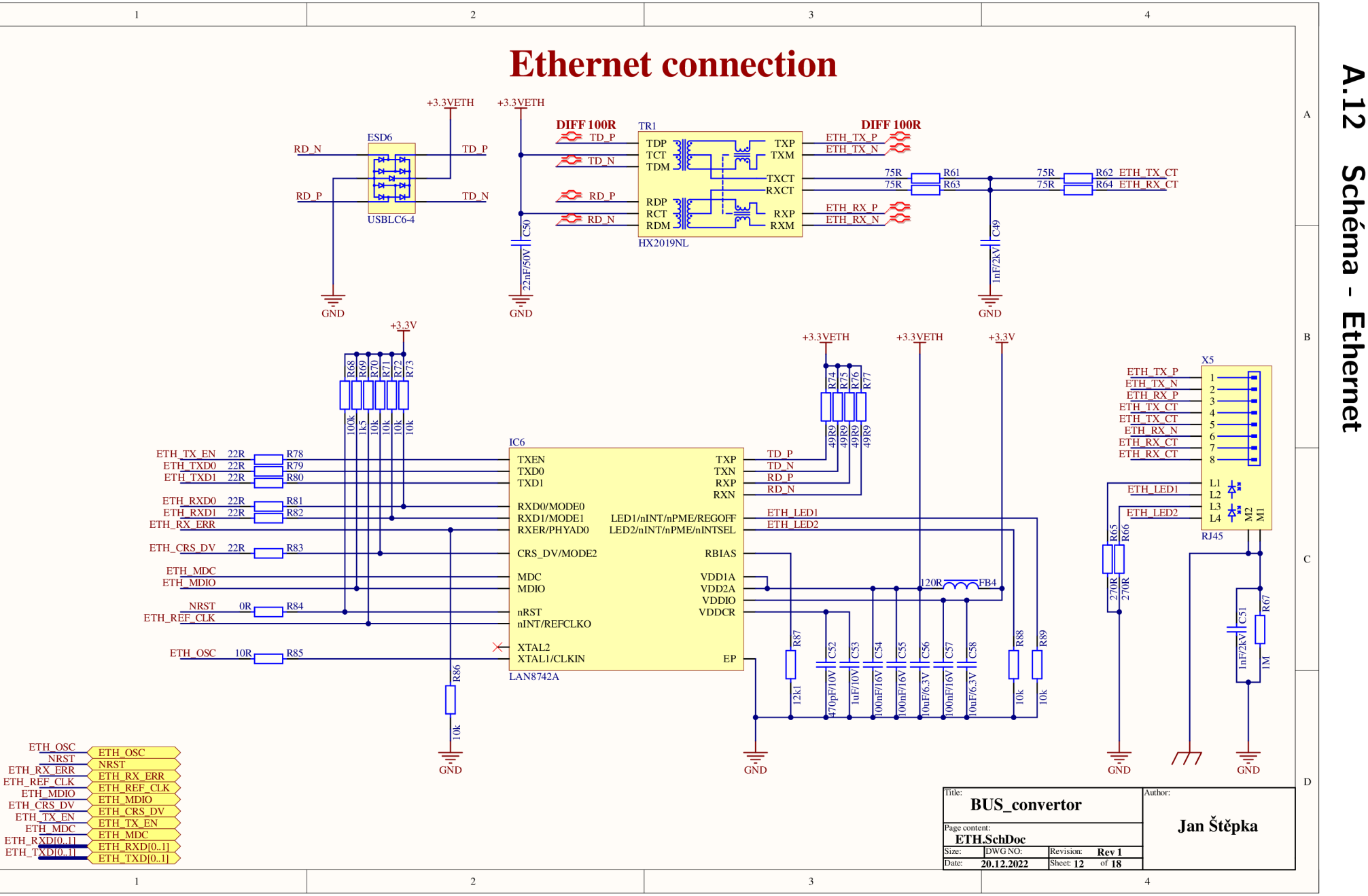




Title: BUS_convertor			Author: Jan Štěpka
Page content: USB.SchDoc			
Size:	DWG NO:	Revision: Rev 1	
Date: 20.12.2022		Sheet: 11 of 18	

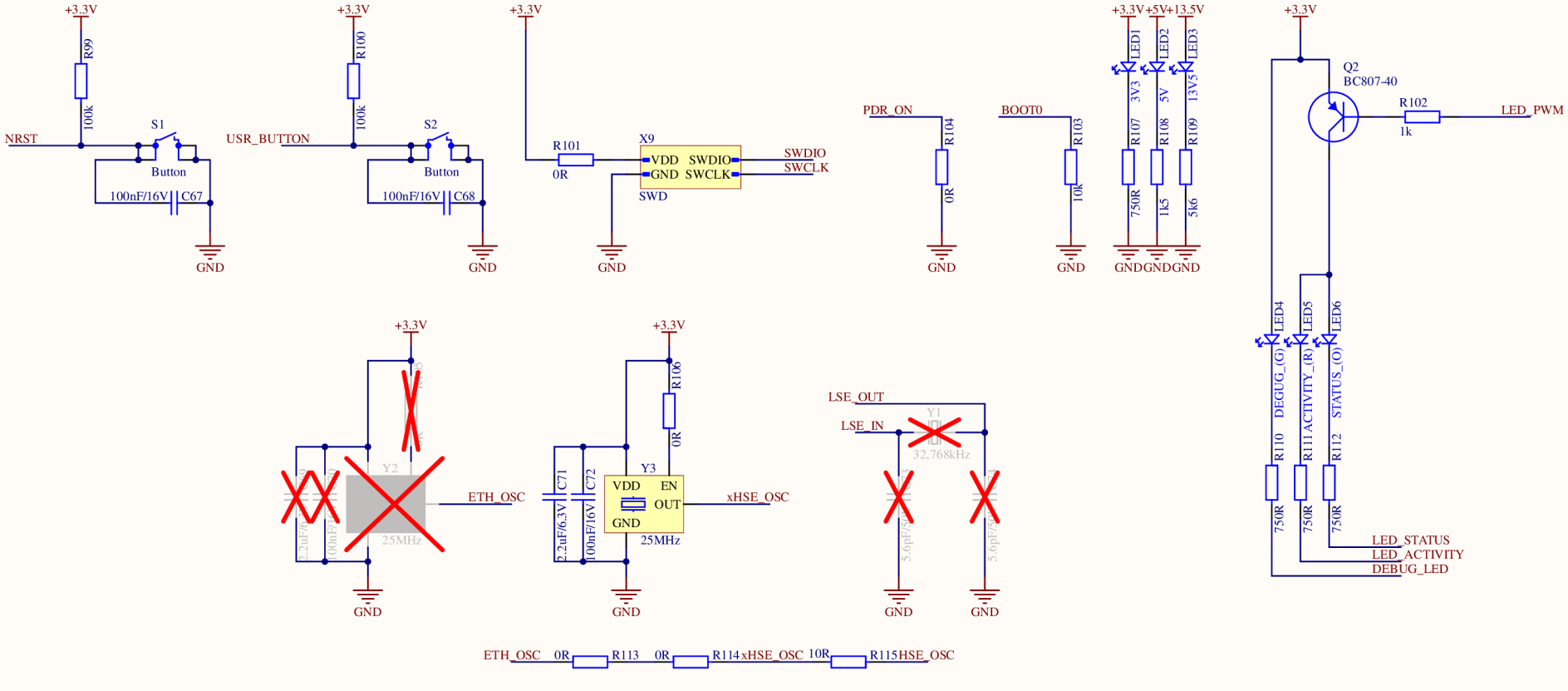
A.11 Schéma - USB

Ethernet connection



Title:	**BUS_convertor**
Page content:	**ETH.SchDoc**
Size:	DWG NO:
Revision:	**Rev 1**
Date:	**20.12.2022**
Sheet:	**12** of **18**
Author:	**Jan Štěpka**

Debug, clock and reset

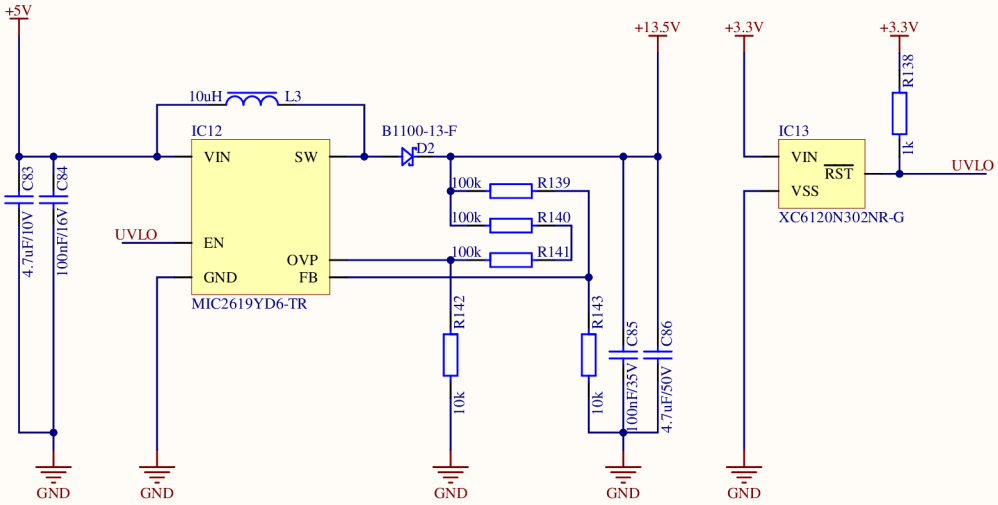


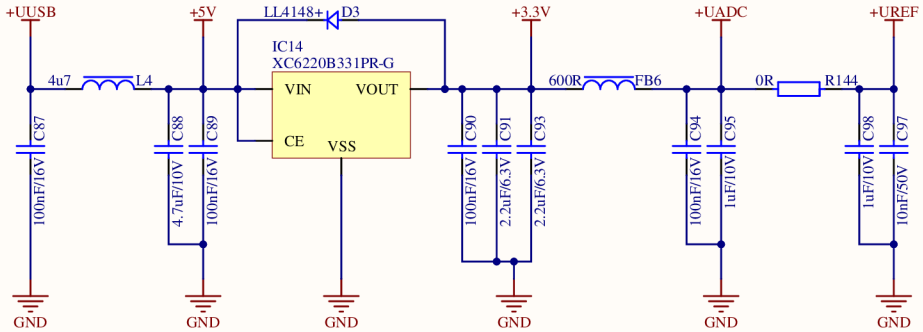
NRST	NRST
BOOT0	BOOT0
HSE_OSC	HSE_OSC
LSE_IN	LSE_IN
LSE_OUT	LSE_OUT
ETH_OSC	ETH_OSC

DEBUG_LED	DEBUG_LED
USR_BUTTON	USR_BUTTON
LED_PWM	LED_PWM
LED_STATUS	LED_STATUS
LED_ACTIVITY	LED_ACTIVITY
SWDIO	SWDIO
SWCLK	SWCLK
PDR_ON	PDR_ON

Title: BUS_convertor			Author: Jan Štěpka
Page content: DCR.SchDoc			
Size:	DWG NO:	Revision: Rev 1	
Date: 20.12.2022		Sheet: 13 of 18	

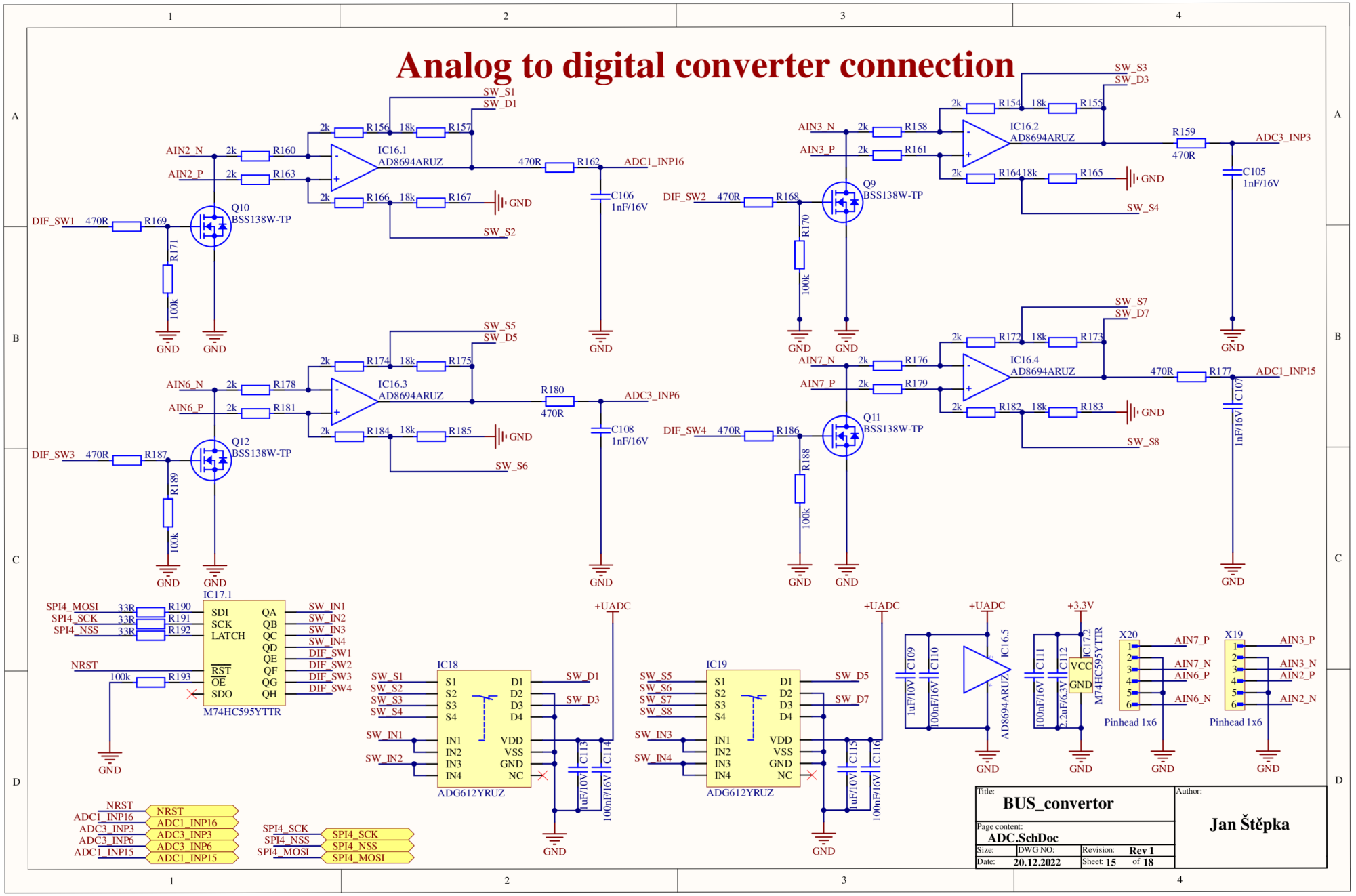
Power supply



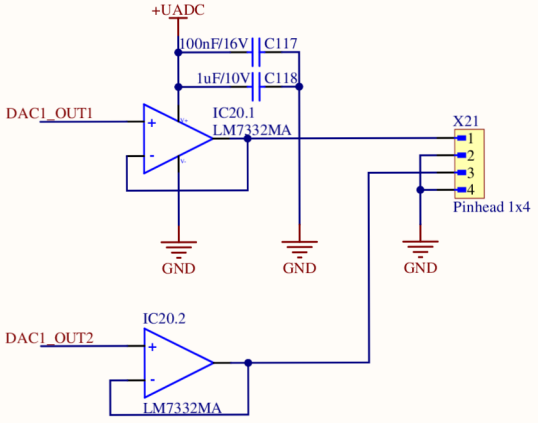


UVLO UVLO

Title: BUS_convertor			Author: Jan Štěpka
Page content: PWR.SchDoc			
Size:	DWG NO:	Revision: Rev 1	
Date: 20.12.2022		Sheet: 14 of 18	


Analog to digital converter connection
IC16.1 AD8694ARUZ
IC16.2 AD8694ARUZ
IC16.3 AD8694ARUZ
IC16.4 AD8694ARUZ
IC16.5 AD8694ARUZ
Q9 BSS138W-TP
Q10 BSS138W-TP
Q11 BSS138W-TP
Q12 BSS138W-TP
IC17.1 M74HC595YTTR
IC17.2 M74HC595YTTR
IC18 ADG612YRUZ
IC19 ADG612YRUZ
ADC1_INP16
ADC1_INP15
ADC3_INP3
ADC3_INP6
AIN2_N
AIN2_P
AIN3_N
AIN3_P
AIN6_N
AIN6_P
AIN7_N
AIN7_P
DIF_SW1
DIF_SW2
DIF_SW3
DIF_SW4
SW_S1
SW_S2
SW_S3
SW_S4
SW_S5
SW_S6
SW_S7
SW_S8
SW_D1
SW_D3
SW_D5
SW_D7
SW_IN1
SW_IN2
SW_IN3
SW_IN4
SPI4_MOSI
SPI4_SCK
SPI4_NSS
NRST
+UADC
+3.3V
GND
Pinhead 1x6
Title: BUS_convertor
Author: Jan Štěpka
Page content: ADC.SchDoc
Size:
DWG NO:
Revision: Rev 1
Date: 20.12.2022
Sheet: 15 of 18

Digital to analog converter connection

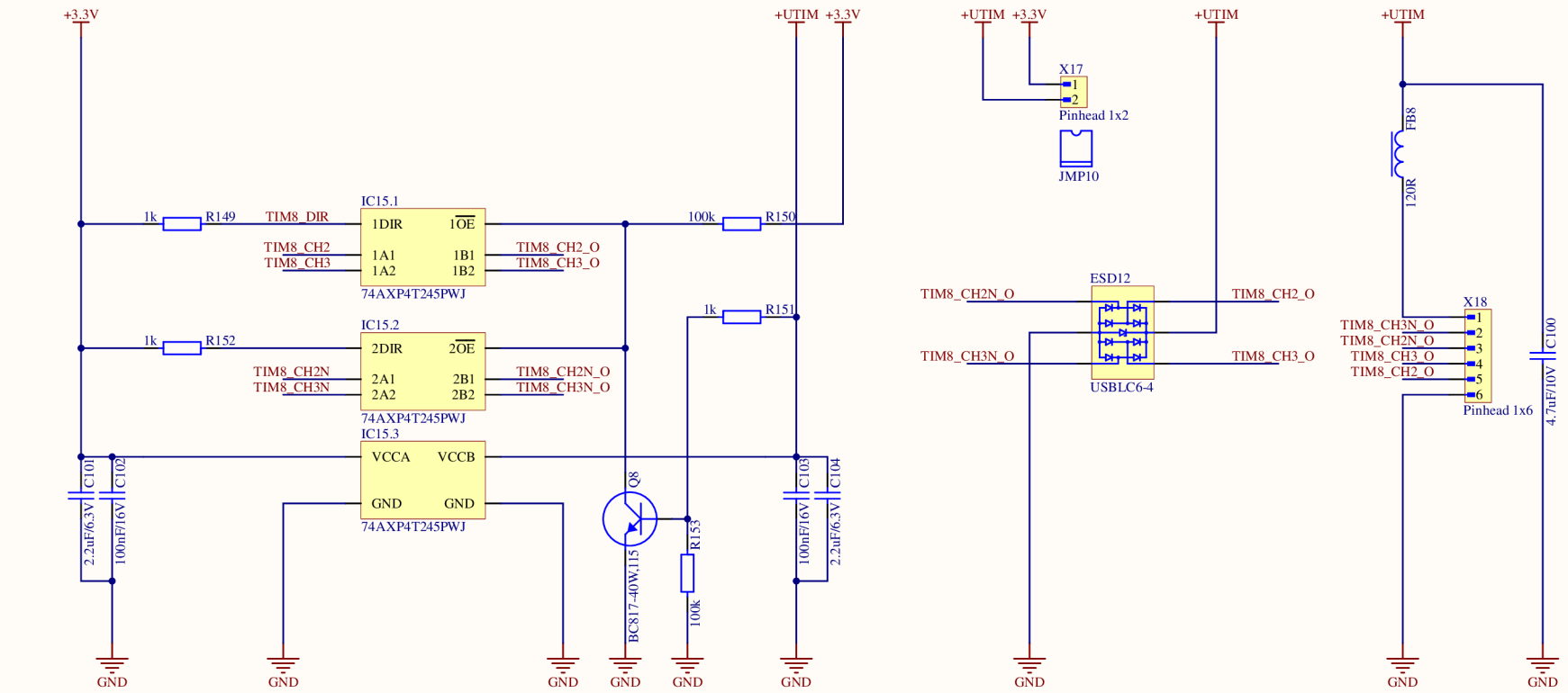


DAC1_OUT1
DAC1_OUT2
DAC1_OUT1
DAC1_OUT2

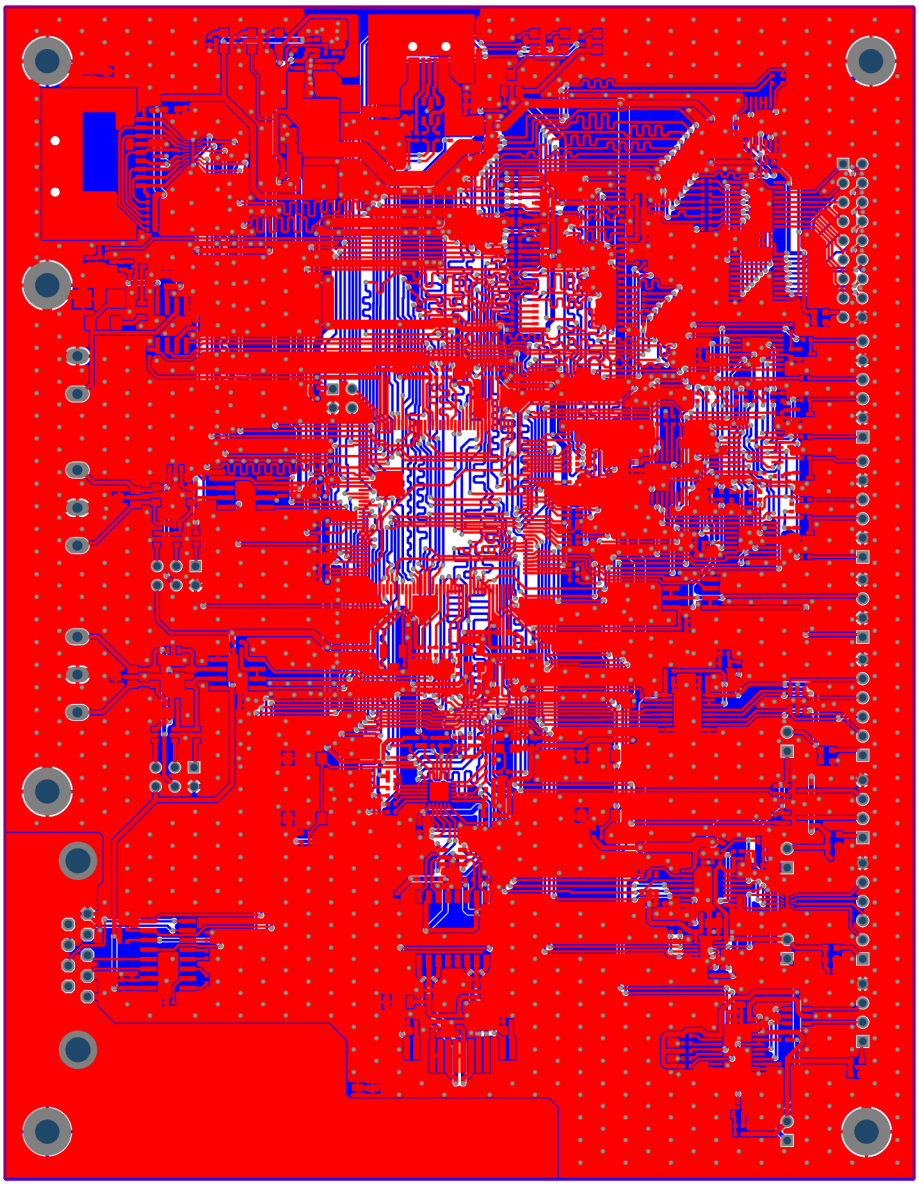
Title: BUS_convertor			Author: Jan Štěpka
Page content: DAC.SchDoc			
Size:	DWG NO:	Revision: Rev 1	
Date: 20.12.2022		Sheet: 16 of 18	

A.16 Schéma - Digitálně analogové převodníky

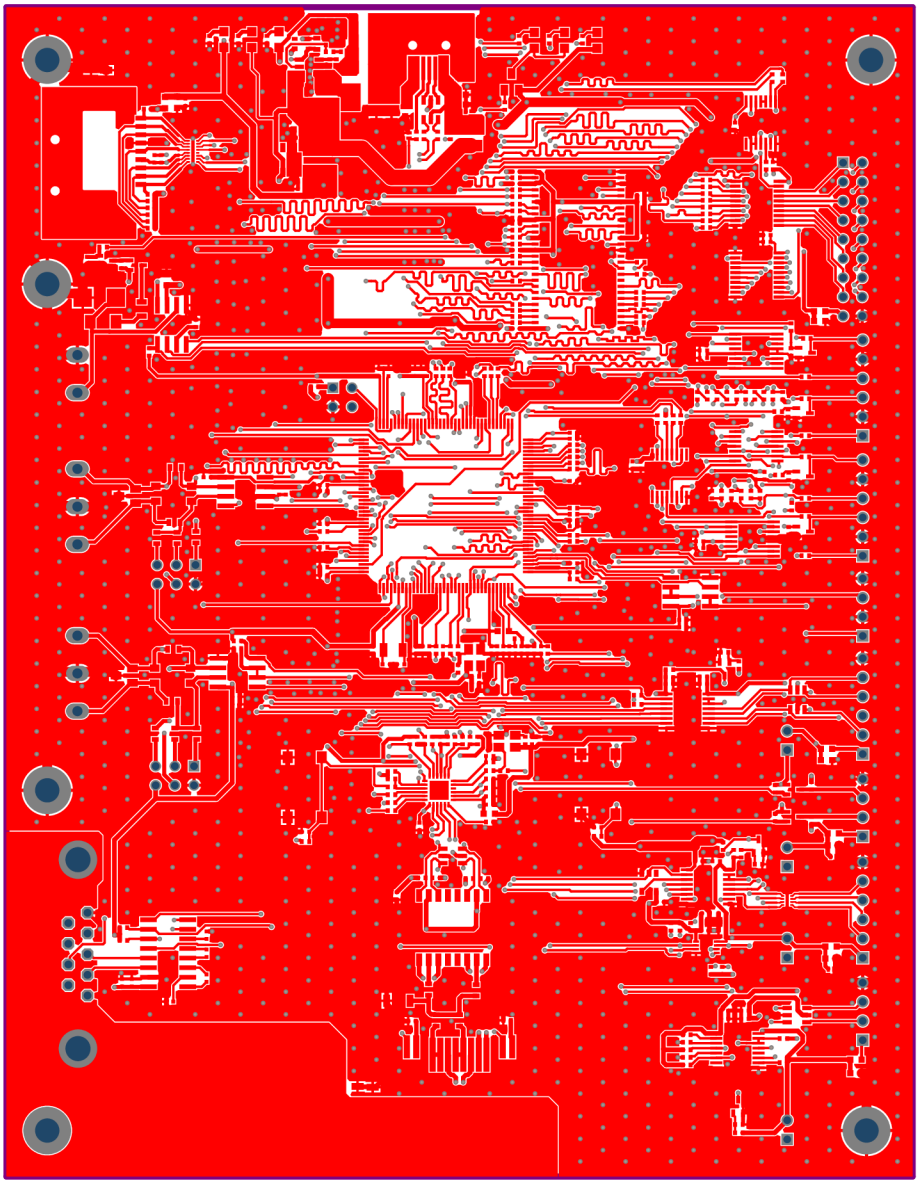
A.17 Schéma - Vstup a výstup časovače



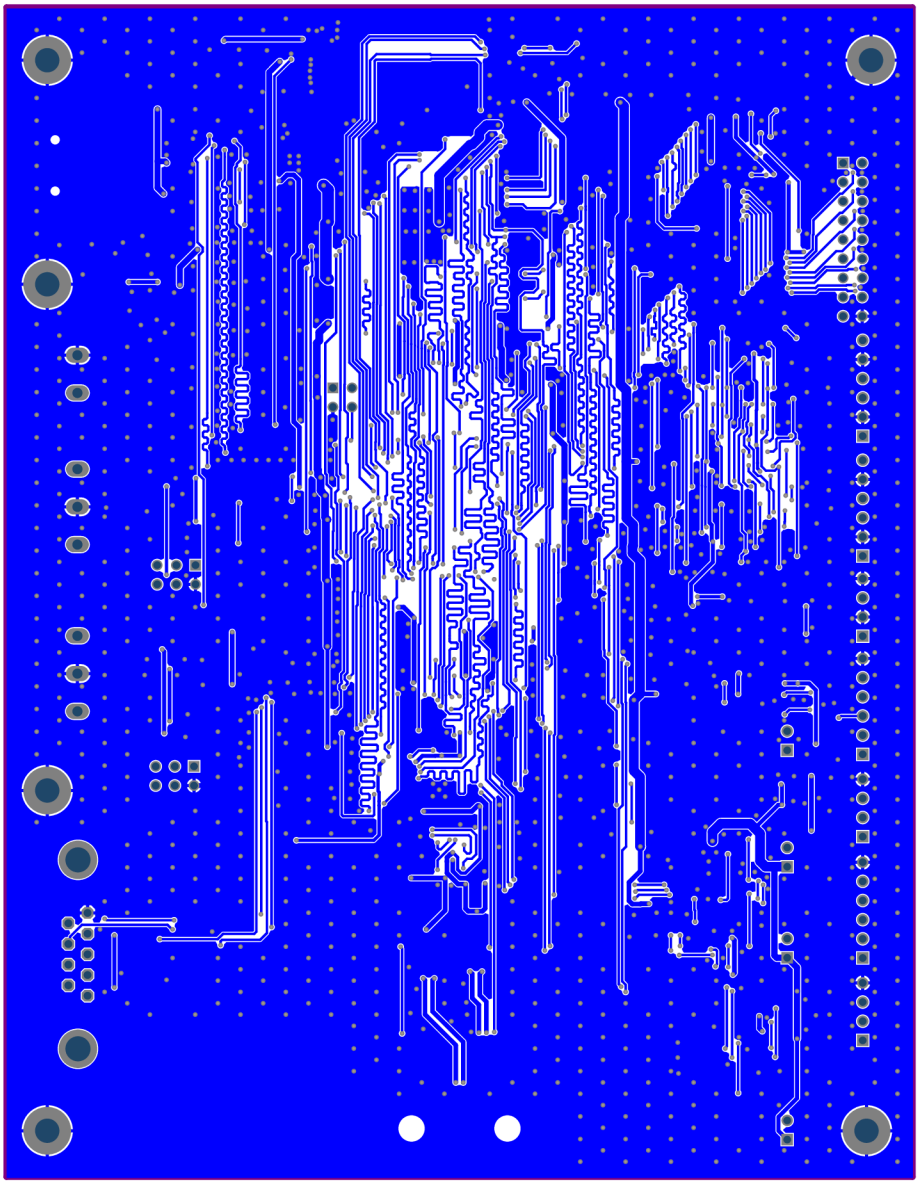
B Deska plošného spoje rozměry 1:1



B.1 Deska plošného spoje - Horní vrstva rozměry 1:1



B.2 Deska plošného spoje - Spodní vrstva rozměry 1:1



B.3 Deska plošného spoje - Osazovací výkres rozměry 1:1