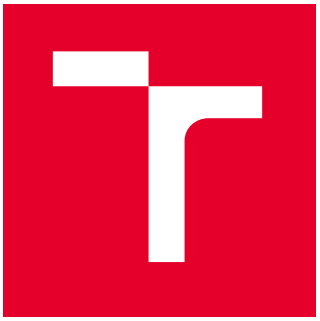



# VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

## FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

## ÚSTAV TELEKOMUNIKACÍ

DEPARTMENT OF TELECOMMUNICATIONS


# ANALYZÁTOR PROTOKOLU DMX 512 PRO OSOBNÍ POČÍTAČ

DMX 512 PROTOCOL ANALYZER FOR PERSONAL COMPUTER

**BAKALÁŘSKÁ PRÁCE**
BACHELOR'S THESIS


**AUTOR PRÁCE** — **Jiří Vahalík**
AUTHOR

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



**BRNO 2023**

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

# Bakalářská práce

bakalářský studijní program **Audio inženýrství**
specializace Zvuková produkce a nahrávání
Ústav telekomunikací

***Student:*** Jiří Vahalík ***ID:*** 220490
***Ročník:*** 3 ***Akademický rok:*** 2022/23

**NÁZEV TÉMATU:**

## Analyzátor protokolu DMX 512 pro osobní počítač

**POKYNY PRO VYPRACOVÁNÍ:**

Cílem práce je návrh a realizace USB - DMX512 převodníku a vhodného software, který by zobrazoval provoz na DMX rozhraní a umožňoval jeho analýzu s možností generování DMX zpráv.

**DOPORUČENÁ LITERATURA:**

[1] HOWELL, Wayne. Control Freak: a real world guide to DMX512 and Remote Device Management. Cambridge: Entertainment Technology Press, 2008, 226 s. : il. ISBN 978-1-904031-55-0
[2] MAZIDI, Muhammad Ali, Sarmad NAIMI a Sepehr NAIMI. The AVR microcontroller and embedded systems: using Assembly and C. Second edition. 2017. ISBN 9780997925968.

***Termín zadání:*** 6.2.2023 ***Termín odevzdání:*** 26.5.2023

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

**doc. Ing. Jiří Schimmel, Ph.D.**
předseda rady studijního programu

## ABSTRAKT

Bakalářská práce se věnuje návrhu DMX převodníku pro osobní počítač. V teoretické části shrnuje protokoly DMX, RDM a Art-Net a další běžně používané komunikační protokoly užívaných mikrokontroléry. V praktické části je zhotoven obvod zařízení, který je následně zpracován jako deska plošných spojů. V poslední části práce jsou navrženy programy zajišťující funkčnost celého převodníku.



## KLÍČOVÁ SLOVA

DMX, DMX převodník, mikrokontrolér, DPS, ESP32, UART



## ABSTRACT

The bachelor thesis deals with the design of a DMX converter for a personal computer. The theoretical part summarizes the DMX, RDM and Art-Net protocols and other commonly used communication protocols used by microcontrollers. In the practical part, a device circuit is fabricated, which is subsequently processed as a printed circuit board. In the last part of the thesis, programs are designed to ensure the functionality of the whole converter.



## KEYWORDS

Překlad klíčových slov DMX, DMX converter, microcontroller, PCB, ESP32, UART

VAHALÍK, Jiří. *Analyzátor protokolu DMX 512 pro osobní počítač*. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav telekomunikací, 2021, 57 s. Bakalářská práce. Vedoucí práce: Ing. Ondřej Krajsa, Ph.D.

# Prohlášení autora o původnosti díla

| | |
|---|---|
| **Jméno a příjmení autora:** | Jiří Vahalík |
| **VUT ID autora:** | 220490 |
| **Typ práce:** | Bakalářská práce |
| **Akademický rok:** | 2022/23 |
| **Téma závěrečné práce:** | Analyzátor protokolu DMX 512 pro osobní počítač |

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

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

Brno ..... .....

podpis autora*

*Autor podepisuje pouze v tištěné verzi.

# PODĚKOVÁNÍ

Rád bych poděkoval vedoucímu bakalářské práce panu Ing. Onřejovi Krajsovi, Ph.D. za odborné vedení, konzultace, trpělivost a podnětné návrhy k práci.

# Obsah

# Seznam obrázků

# Seznam tabulek

# Úvod

Bakalářská práce se věnuje návrhu analyzátoru protokolu DMX512, který lze přes rozhraní USB připojit k řídícímu osobnímu počítači. Práce řeší návrh tohoto převodníku s mikrokontrolérem ESP32. Aby tento cíl mohl být splněn práce popisuje komunikační protokoly jejichž znalost je pro konstrukci takového zařízení potřebná.

Jmenovitě jde o protokoly DMX, Art-Net, RDM, SPI a komunikační rozhraní UART. Dále se práce věnuje výběru součástek pro konstrukci převodníku. Konkrétněji pak výběru mikrokontroléru, potřebných budičů sběrnice, ehternetového modulu a dalších periferií. V práci jsou také uvedeny příklady zapojení jednotlivých částí obvodu.

Dále práce sumarizuje návrh desky plošných spojů a umístění součástek na tuto desku. Následně je v práci popsaný návrh kódu nahraného na samotný mikrokontrolér a stručně je popsána jeho funkčnost.

V poslední části se práce zabývá návrhem aplikace v jazyce Python pro osobní počítač, která slouží k řízení funkcí zařízení. Této aplikaci je následně jednoduché uživatelské rozhraní v knihovně *tkinter*.

# 1 DMX512

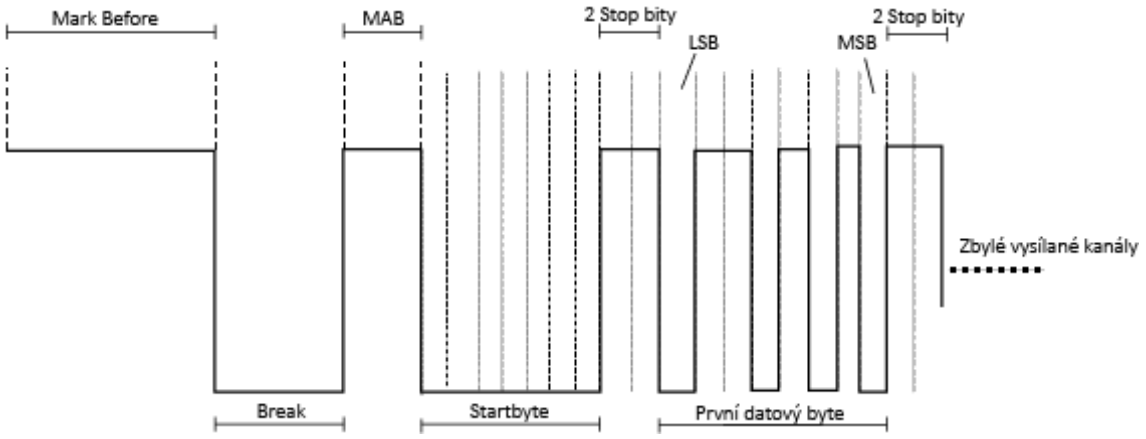
## 1.1 Vznik protokolu DMX512

První verze protokolu DMX512 (Digital Multiplex) byla navržena v roce 1986 Americkým institutem pro divadelní technologii (USITT). Jeho pozdější revize v roce 1990 dala za vznik první publikované verzi tohoto protokolu USITT DMX512/1990. DMX512 měl nahradit klasická analogová řešení řízení divadelního osvětlení, která využívala okamžitou hodnotu napětí na řídícím kabelu jako řídící veličinu. Proto byl nejprve protokol DMX koncipován hlavně pro ovládání stmívačů (dimmerů), ale nyní se používá na ovládání všech atributů jevištního osvětlení. [1]

Protokol DMX se kromě nahrazení analogových stmívačů využívá k polohování jednotlivých hlav, ovládání barvy a jiných parametrů svitu, řízení tvorby mlhy a mnoho dalších speciálních efektů. Z důvodu nedokonalé ochrany proti chybám by ale neměl být využit k ovládání pyrotechniky. Základ protokolu DMX512 vychází z průmyslového standardu EIA-485 (RS485), který se pro sériovou datovou komunikaci běžně používá. [2]

## 1.2 Základní specifikace

- Symetrický přenos dat, schopnost pracovat od napájecího napětí +5 V
- Přípustný rozsah napětí na sběrnici od -7 V do +12 V
- Schopnost připojení až 32 zařízení na jednu sběrnici
- Poslední zařízení musí být připojeno na koncový rezistor (terminator) o odporu 120 Ω
- Minimální vstupní odpor 120 Ω
- Minimální zatěžovací impedance vysílače je 60 Ω
- Maximální zkratový proud vysílače je 150 mA ku zemi a 250 mA proti 12 V
- Maximální přenosová rychlost je 400 kbps pro maximální délku 1200 m
- Kabely a konektory XLR (Canon) 5-ti pinové




Obr. 1.1: Rámec DMX signálu

## 1.3 Komunikace protokolu

Standard DMX definuje sériovou komunikaci založenou na elektrickém standardu RS485. Tato komunikace funguje na principu Master-Slave. Přenosová rychlost je pevně stanovena na 250 kbit/s. Komunikace je pouze jednosměrná, na sběrnici je pouze jeden vysílač (Master), typicky světelný pult nebo osobní počítač, který posílá povely všem ostatním zařízením na sběrnici. Tato zařízení jsou pouze přijímače signálu (Slave). Na obrázku 1.1 lze vidět jeden rámec komunikace protokolu DMX512. [3]

### 1.3.1 Časování komunikace

Komunikace spočívá ve vysílání rámců o maximální délce 512 bajtů. Po směrnici se posílají pouze data bez adresy cílového zařízení. Pro každý kanál mohou data nabývat hodnoty 0-255, kdy 0 je vypnuto a 255 je maximální hodnota.

V počátečním stavu je sběrnice na úrovni HI. Začátek přenosu informací je synchronizován nulovou úrovní „Break“ (Reset), která trvá nejméně 88 µs a následující synchronizační mezerou MAB (Mark After Break) o úrovni HI trvající minimálně 8 µs. Poté je poslán první rámec (Startbyte) a za ním 512 datových rámců. Každý přenesený rámec se skládá ze Start bytu, osmi datových bitů a dvou Stop bitů s úrovní HI. Mezi jednotlivými rámci mohou být mezery MTBF (Mark Time Between Frames) a MTBP (Mark Time Between Packet) s maximální délkou 1 s. V tabulce 1.1 je toto časování znázorněno pro větší přehlednost. [4]

| **Popis** | **Min.** | **Typ.** | **Max.** |
|---|---|---|---|
| Break (Reset) | 88 µs | 176 µs | 1 s |
| MAB | 8 µs | - | 1 s |
| Rámec | 43,12 µs | 44 µs | 44,48 µs |
| MSB | 3,92 µs | 4 µs | 4,08 µs |
| LSB | 3,92 µs | 4 µs | 4,08 µs |
| MTBF | 0 | 0 | 1 s |

Tab. 1.1: Časování DMX signálu

## 1.4 Fyzická vrstva

Fyzická vrstva protokolu DMX je odvozena od standardu EIA-485 (RS485). Tento standard je používán pro poloduplexní vícebodovou asynchronní komunikaci. Výhodou je, že logické stavy jsou vyjádřeny diferenciálním napětím mezi dvěma vodiči (označovanými A a B). Díky tomu je kabel odolný proti indukovanému šumu, protože indukované napětí má na obou vodičích stejnou polaritu, tudíž bude odečteno. Vysílač by sběrnici měl budit minimálním napětím ±1,5 V, maximálně však ±6 V mezi vodiči A a B.

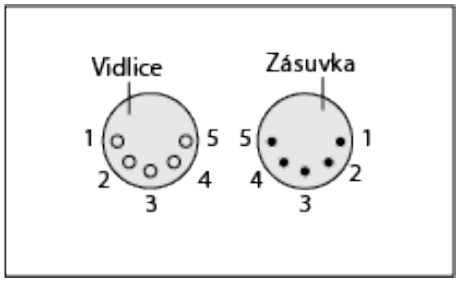
Dle normy DMX512/1988 je standardním konektorem pětipinový XLR konektor. Na straně vysílače je použita zásuvka, na straně přijímače vidlice. Dále je dle normy ANSI E1.11-2008 (R2018) povoleno použití konektoru IEC 60603-7 (označován RJ45) pouze v případě trvalého zapojení nepřístupného běžnému uživateli.

V praxi se nejčastěji setkáváme s použitím třípinového XLR konektoru, pětipinovým XLR konektorem disponuje o mnoho méně zařízení. Třípinové zapojení se dle normy nedoporučuje. Mohlo by dojít k nechtěné záměně DMX kabelu a mikrofoního XLR, následně by napětí myšleno pro audio-zařízení mohlo DMX zařízení poškodit. [1]

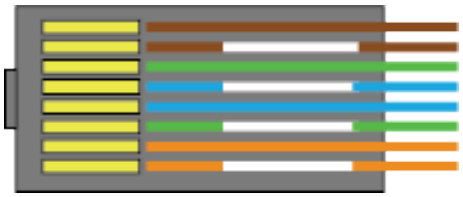
| Pin (vodič) | Barva | DMX 512 funkce | XLR pin |
|---|---|---|---|
| 1 | bílá / oranžová | DATA+ | 3 |
| 2 | oranžová | DATA- | 2 |
| 3 | bílá / zelená | DATA2+ | 5 |
| 4 | modrá | nepoužitý | n/c |
| 5 | bílá / modrá | nepoužitý | n/c |
| 6 | zelená | DATA2- | 4 |
| 7 | bílá / hnědá | DATA1 COM | 1 |
| 8 | hnědá | DATA2 COM | 1 |

Tab. 1.2: Zapojení konektoru RJ45 pro DMX

### 1.4.1 Konektory




Obr. 1.2: DMX 5 pinů



Obr. 1.3: Konektor RJ45

**Zapojení pětipinového konektoru**

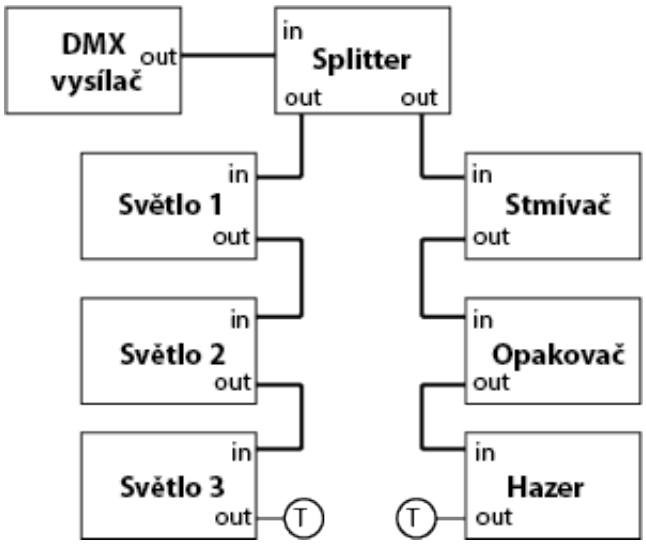
- 1 - stínění (nespojeno s tělem konektoru)
- 2 - DMX data - pól
- 3 - DMX data + pól
- 4 - nepoužitý (DMX data 2 -)
- 5 - nepoužitý (DMX data 2 +)

### 1.4.2 Propojení zařízení na sběrnici

Signál vychází z jednoho vysílače (osvětlovacího pultu nebo PC) a dále je veden do prvků řízených, které jsou na sběrnici řazeny v sérii. Signál tedy putuje z vysílače do konektoru IN prvního zařízení a z konektoru OUT stejného zařízení je veden dále. Jednotlivým zařízením na sběrnici je nutno nastavit kanál, podle kterého se mají řídit. Dvě zařízení mohou číst stejný kanál, nelze je ale poté řídit samostatně a vykonávají totožné úkony. Uživatel nastavuje sice jen jeden kanál, ale různá zařízení ke

správnému fungování čtou kanálů více, některá i 14 až 20. V posledních letech nejmodernější zařízení vyžadují velké množství kanálů a horní limit 512 kanálů začíná být omezující. Každý kanál poté řídí jinou vlastnost světla (poloha, barva, gobo, atd.). Konec této linky musí být opatřen zakončovacím odporem (terminátorem).

Hodnota zakončovacího odporu je 120 Ω, což je nejbližší hodnota z řady E12 k hodnotě impedance kabelu. Pokud by sběrnice nebyla ošetřena terminátorem, mohlo by docházet ke zkreslení dat odrazem signálu. Tento odpor se v praxi řeší rezistorem zapájeným mezi 2 a 3 pin XLR konektoru. Pokud chceme signál DMX paralelně větvit, musíme použít rozbočovač (splitter). K posílení signálu vedeného na delší vzdálenosti se používá opakovače. Obrázek 1.4 ukazuje jak může typické zapojení vypadat. [2]




Obr. 1.4: Sběrnice DMX

### 1.4.3 Možnosti DMX sběrnice

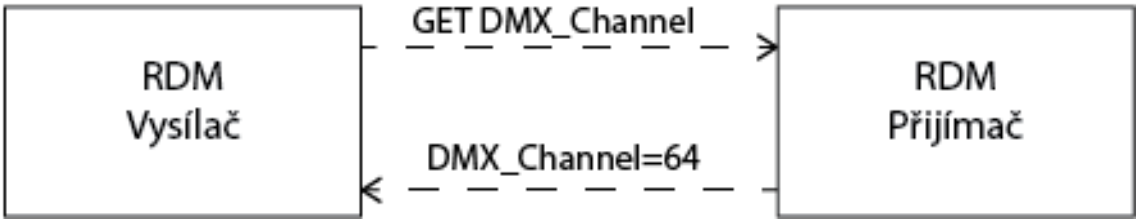
Plně využitá sběrnice DMX512 má opakovací frekvenci okolo 44 Hz, z toho vyplývá, že minimální dosažitelná doba přenosu plně zatížené sběrnice je 23 ms. Nedostatečná rychlost takto zatížené sběrnice by mohla být pro některé aplikace nevyhovující. Obecně se doporučuje využít více sběrnic DMX512 nebo signál pomocí spliterů větvit. Světelná technika je k tomu uzpůsobená. Nejvýkonnější DMX pulty dokáží vysílat až 16 DMX sběrnic (universů) po 512 kanálech, takovéto pulty jsou pro většinu uživatelů velmi drahé a bývají trvalou instalací například v divadlech nebo jiných kulturních prostorách. Z tohoto důvodu se setkáváme spíše s pulty, které jsou schopny vysílat pouze jednu DMX sběrnici.

## 1.5 Wireless DMX

Stejně jako v jiných odvětvích, i v oblasti osvětlovací techniky se v současné době objevuje stále více bezdrátových prvků, od bezdrátových výrobníku mlhy až po různé bezdrátové osvětlovací prvky. Taková zařízení se v hojně využívají zejména jako doplňkové nebo efektové osvětlení. Nevýhodou pro takováto zařízení je, že musí využívat pouze veřejně dostupná pásma. Běžně se používají pásma 2,4 GHz a 5 GHz, ta jsou ale obecně zahlcená i jinou jevištní bezdrátovou technikou, například IEM (In ear monitors) nebo bezdrátovým přenosem nástrojů, typicky kytary. Proto se téměř nesetkáváme s případem, že by celá scénická technika byla řešena bezdrátově. Realizace bezdrátového přenosu DMX signálu typicky vypadá tak, že se nejprve signál převede na Art-Net pakety a ty jsou poté vysílány prostřednictvím Wi-Fi. Tato varianta je poměrně nenáročná, zejména díky nízkým pořizovacím nákladům. Další variantou je pořízení speciálního vysílače a přijímače, který signál DMX přímo převádí na rádiové vlny. Z důvodu velkého rizika výpadků signálu se bezdrátové DMX prvky využívají jen okrajově, zejména jako doplnění scény. [5]

## 1.6 RDM

RDM (Remote Device Management) je protokol, který se pojí k DMX protokolu. Data protokolu DMX putují pouze ve směru od vysílače k přijímači. S přidáním protokolu RDM je již komunikace oboustranná, takzvaně polo duplexní, protože data nemohou zároveň putovat oběma směry, ale směry komunikace se střídají. Ilustraci této komunikace lze vidět na obrázku 1.5. Vysílač posílá s daty DMX zároveň také data na která jednotlivé přijímače odpovídají. Obsah těchto odpovědí může obsahovat například DMX adresu jednotlivých zařízení, chybové hlášky nebo údaje o aktuálně vykonávaných operacích. Tento typ RDM zpráv se označuje jako GET zpráva, která je pro kontrolér pouze informační. RDM kontroléry kromě toho mohou posílat také SET zprávy, které na dálku mohou například změnit DMX adresu jednotlivých zařízení na sběrnici. Takto komunikovat umí pouze zařízení které podporuje RDM protokol. Také je třeba speciálního RDM vysílače nebo světelného pultu, který je schopen tento protokol vysílat společně s DMX zprávami. [6]




Obr. 1.5: Poloduplexní komunikace RDM

# 2 Art-Net

## 2.1 Vznik protokolu Art-Net

Art-Net je datový protokol pro šíření DMX512 a RDM protokolů po ethernetové síti. Tvůrcem protokolu Art-Net je Wayne Howell a jeho firma Artistic Licence Ltd, která první verzi tohoto protokolu Art-Net I vydala již v roce 1998. Aktuální verze tohoto protokolu Art-Net 4 byla vydána v roce 2016, v tomto roce byla také firma oceněna cenou PLASA Award for Innovation. Vznik tohoto protokolu zapříčinilo neustálé zvyšování nároků na řízení světel a jiné pódiové techniky. Hlavním cílem bylo zvětšit množství kanálů signálu DMX přenášené jedním kabelem. Art-Net je volně dostupný k bezplatnému použití pokud výrobce uvede zásluhu firmy Artistic Licence Ltd. [7]

## 2.2 Protokol Art-Net

Tento komunikační protokol umožňuje zapouzdření signálů z jednoho nebo více DMX vysílačů do internetových paketů. Takto je možné jedním ethernetovým kabelem vést signál, který obsahuje více sběrnic DMX512 (universů). Obvykle je takovýto signál přiveden do rozbočovače a odtud je dále k jednotlivým osvětlovacím prvkům již veden jako klasický DMX signál. Některé přijímače ale dnes už mohou pracovat s paketovaným signálem, s takovými zařízeními se setkáváme spíše u větších produkcí. Přestože existují 4 verze Art-Net protokolu, všechny jsou zpětně kompatibilní. Maximální přenosová rychlost a počet zařízení tohoto protokolu jsou omezovány zejména použitou síťovou technologií a způsobem jejího adresování. Maximální počet universů na jednom ethernetovém kabelu je teoreticky až 32768. Protokol Art-Net obsahuje velké množství paketů pro kontrolu a správu zařízení v sítí. ArtPoll paket je vysílán všemi řídícími zařízeními v sítí, na tento paket odpovídají všechna připojená zařízení paketem ArtPollReply. V odpovědi ArtPollReply zařízení posílá řídícímu zařízení základní informace o jeho funkci, vykonávaných pokynech nebo chybové hlášky. Jakékoli zařízení, které do 3 sekund neodpoví na paket ArtPoll je řídícím zařízením považováno za odpojené. Nejdůležitější paket je paket ArtDmx, který obsahuje řídící data pro přijímače, tedy data protokolu DMX512.

### 2.2.1 Paket ArtDmx

Na začátku ArtDmx paketu je 8bytové ID, prvních 7 bytů obsahuje v ASCII kódu nápis „A r t - N e t“, poslední byte ID má vždy nulovou hodnotu. ID slouží pouze k identifikaci Art-Net paketu. Následuje 16 bitů OpCode, který definuje typ paketu. Dalších 16 bitů ProtVer poté identifikuje použitou verzi Art-Net protokolu. Následujících 8 bitů Sequence určuje původní pořadí paketů, které mohou po síti dorazit v jiném pořadí než v jakém byly vyslány. Tato informace umožňuje přijímači opravit případné chyby. Physical je 8bitový informativní údaj o fyzickém portu, který tento paket vygeneroval. Další dvě části Net a SubUni spolu tvoří 16bitovou adresu, která určuje kam má být paket zaslán. Pole Length je 16bitový údaj o množství přenášených DMX kanálů, proto se hodnotota tohoto pole pohybuje od 2 do 512. Jakékoli vyšší číslo v poli Length bude vyhodnoceno jako chyba. Následuje pole Data, ve kterém jsou data DMX protokolu. Prvních 8 bitů je vždy kanál 1. Pro větší přehlednost je celá struktura ArtDmx paketu rozepsána v tabulce 2.1. [8]

| **Bytový offset** | **Název** | **Velikost[Byte]** | **Popis** |
|---|---|---|---|
| 0 | ID | 8 | Identifikace Art-Net |
| 9 | OpCode | 2 | Definuje typ paketu |
| 11 | ProtVer | 2 | Verze protokolu |
| 12 | Sequence | 1 | Určuje pořadí paketů |
| 13 | Physical | 1 | Port který paket generoval |
| 14 | SubUni | 1 | Low 8 bitů cílové adresy |
| 15 | Net | 1 | High 8 bitů cílové adresy |
| 17 | Length | 2 | Délka pole Data |
| 18 | Data | 2-512 | Hodnota kanálů DMX |

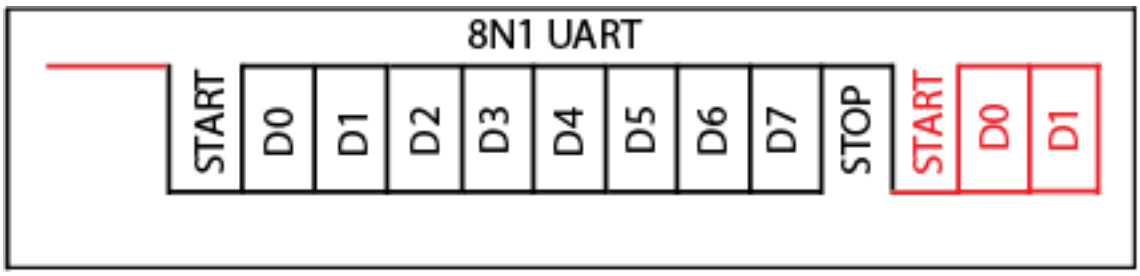
Tab. 2.1: Definice paketu ArtDmx

# 3 UART

UART (Universal asynchronous receiver-transmitter) je rozhraní pro sériovou datovou komunikaci. Řadí se mezi tzv. plně duplexní (full duplex) komunikace, což znamená, že lze současně jak přijímat informace, tak je vysílat. UART rozkládá bajty na sekvenci jednotlivých bitů, ty je pak například druhý UART schopen opět interpretovat jako celé bajty. Vzhledem k tomu, že se v této komunikaci nepřenáší žádný signál časování CLOCK, je třeba nastavit obě strany na stejnou přenosovou rychlost. Tato rychlost se uvádí v baudech (bitech přenesených za jednu sekundu). Nejčastěji se setkáváme s rychlostmi 9600 baud, 19200 baud a 115200 baud. Tyto rychlosti jsou kompromisem mezi přenosovou rychlostí dat a spolehlivostí komunikace rozhraní UART. [9]

## 3.1 Komunikace rozhraní UART

Aby UART komunikace fungovala správně musí mít obě strany nejen správně nastavenou stejnou rychlost, ale i správně nastavené počty bitů, které mají očekávat. To zahrnuje počet paritních bitů, počet datových bitů i počet stop bitů. V počátečním stavu je UART na hladině HIGH. Začátkem komunikace je jeden start bit s úrovní LOW. Poté následuje 5–9 data bitů, dále může být obsažen jeden paritní bit. Paritní bit v této komunikaci určuje zda jde o sudý nebo lichý bit. Celý rámec této komunikace ukončují 1–2 stop bity. Typické nastavení tohoto rozhraní se označuje jako 8N1 a skládá se z osmi data bitů a jednoho stop bitu bez obsažení paritního bitu, takovéto nastavení komunikace je vyobrazeno na obrázku 3.1. [10]




Obr. 3.1: 8N1 UART komunikace

# 4 SPI komunikační protokol

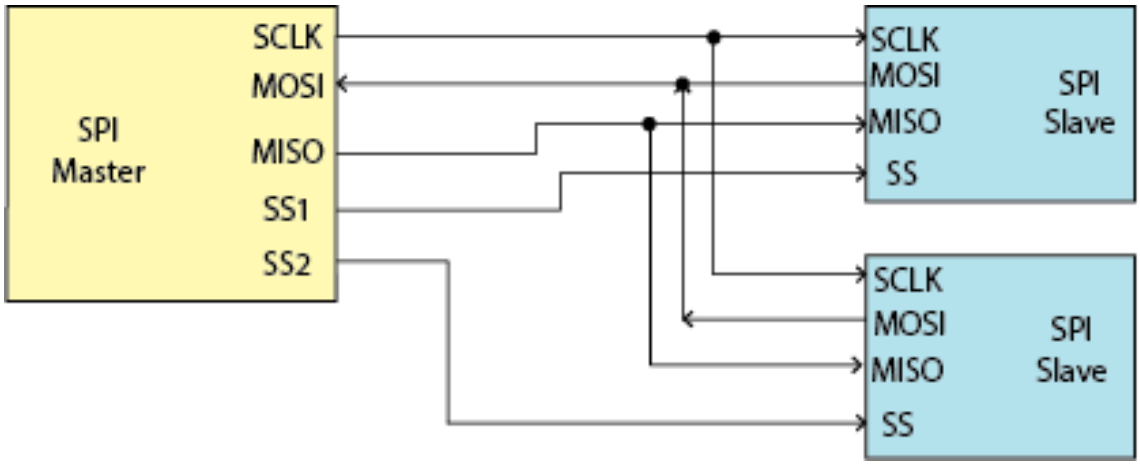
SPI (Serial Peripheral Interface) protokol je plně duplexní synchronní komunikační protokol využíván zejména u mikroprocesorů a mikrokontrolérů. Tato komunikace je založená na vztahu Maste-Slave, kdy jedno zařízení Master ovládá funkci všech ostatních zařízení Slave. Typicky je Master zařízením mikrokontroler a zařízení Slave jsou jeho periférie. Tento protokol využívá čtyři signály. Časovací signál SCLK je vysílaný z řídícího zařízení, ostatní signály protokolu SPI jsou pak právě tímto signálem časovány. Dále signál SS (Slave Select), který určuje se kterým zařízením řídící zařízení komunikuje. Pokud je na pinu SS řízeného zařízení hodnota log. 0 znamená to, že byla periferie vybrána a může přijímat a odesílat data. Pokud je na tomto pinu naopak hodnota log. 1, zařízení není schopné komunikace. Další dva signály jsou MOSI (Master Out-Slave In) a MISO (Master In-Slave Out), kdy signál MOSI reprezentuje datový signál z řídícího zařízení do řízených a signál MISO reprezentuje datový proud v opačném směru. Schéma této komunikace je vyobrazeno na obrázku 4.1 [11].

## 4.1 Komunikace SPI protokolu

Během jednoho cyklu signálu SCLK vždy řídící zařízení pošle bit signálu MISO. Řízená periferie jej přečte a zároveň pošle bit signálu MOSI, který přečte řídící zařízení. Tato sekvence je zachována i když jde pouze o jednosměrnou koumunikaci. Komunikace probíhá od MSB (Most Significant Bit), kdy je s každým cyklem signálu SCLK přijatý bit zařazen jako nový LSB (Least Significant Bit). Toto probíhá až do chvíle, kdy je přenos kompletní. Poté řídící zařízení typicky přestává vysílat signál SCLK a mění hodnotu signálu SS, tudíž zařízení se kterým řídící zařízení komunikovalo již není zvoleno jako cílové zařízení této komunikace. [12]

## 4.2 Polarita a fáze signálu SCLK

Mimo nastavení správné frekvence musí řídící zařízení nastavit i správnou polaritu a fázi signálu SCLK. Motorola SPI Block Guide [13] pojmenovává tyto dvě vlastnosti časovacího signálu jako CPOL (Clock Polarity) a CPHA (Clock Phase). Pokud je CPOL=0 znamená to, že časovací signál je na úrovni 0 a každý puls takového signálu nabývá hodnoty 1. Pro CPOL=1 pak platí, že je časovací signál na úrovni 1 a jednotlivé časovací pulsy jsou o hodnotě 0.




Obr. 4.1: Master-Slave komunikace protokolu SPI

CPHA určuje časový posun časovacího signálu vůči datům, tedy jeho fázi. Pokud se CPHA=0 budou se bity signálů MISO a MOSI řadit podle sestupné hrany časovacího signálu. Pokud se naopak CPHA=1 budou se tyto bity řadit podle hrany náběžné. [14]

## 4.3 Konfigurace řízených periferií

Řízená zařízení protokolu SPI na sobě mohou být zcela nezávislá. V tomto případě spolu řízená zařízení vůbec nekomunikují a data přijímají pouze od řídící jednotky. Řídící zařízení je pak spojeno se všemi piny MISO řízených zařízení. Takové zapojení je pro protokol SPI zdaleka nejběžnější. Odpovídá mu i zapojení na obrázku 4.1.

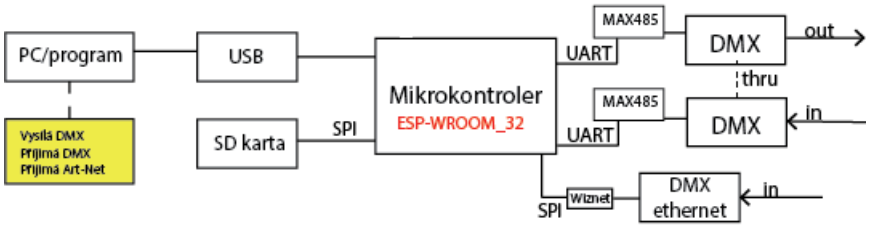
Jako opak k první možnosti zapojení je možná konfigurace, kdy je řídící zařízení připojeno pouze na MISO pin prvního řízeného zařízení. Další jsou pak zapojená v sérii, kdy pin MOSI zařízení které je spojeno s řídící jednotkou vede na pin MISO dalšího řízeného zařízení. Až poslední zařízení v sérii posílá data z pinu MISO zpět na řídící zařízení. [12]

# 5 Koncepce vlastního DMX převodníku

Na základě předešlých kapitol vyplývá, že DMX analyzátor lze pojmout mnoha způsoby. V tomto případě půjde o jednoduchý DMX512 analyzátor, který lze spojit pomocí USB (Universal Serial Bus) s osobním počítačem a analyzovat signály na DMX sběrnici, kterou do zařízení přivedeme 3pinovým XLR konektorem. Ačkoli norma doporučuje použití 5pinových konektorů je pro využití analyzátoru praktičtější 3pinový konektor, protože většina dostupné techniky je opatřena právě tímto konektorem. Dále by zařízení mělo být schopné analyzovat i signály protokolu Art-Net, proto na něj bude osazen i ethernet port. DMX analyzátor by také měl mít možnost data která přijmá rovnou i ukládat na vnitřní pamět. [1]

## 5.1 Požadované vlastnosti a funkce

Navrhované zařízení by mělo mít jak DMX vstup tak DMX výstup. Vstup slouží pro analýzu signálu na zapojeného DMX kabelu, na výstup bude zařízení vysílat vlastní DMX zprávy vygenerované pomocí programu na počítači. Z důvodu spojení s osobním počítačem by mělo zařízení disponovat USB portem. Aby bylo toto zařízení schopné analyzovat i protokol Art-Net, měl by na něj být osazen ethernetový port RJ45. Pro správnou funkci ethernetového portu je třeba na zařízení osadit i čip, který bude ethernetový signál zpracovávat. Pro praktické účely by zařízení mělo být schopné ukládat data, které na sběrnici DMX přijímá a také z uložiště spouštět předdefinované DMX sekvence. Z tohoto důvodu by bylo vhodné osadit také port na microSD paměťovou kartu. Na obrázku 5.1 je zobrazeno blokové schéma zařízení.




Obr. 5.1: Základní blokové schéma DMX převodníku

## 5.2 Návrh DMX převodníku

### 5.2.1 Mikrokontrolér

Celé zařízení bude řízeno mikrokontrolérem, který bude přijímat pokyny od počítače a řídit ostatní periferie. Při výběru mikrokontroléru bylo třeba splnit několik požadavků k plné funkčnosti celého návrhu. Mikrokontolér bude přijímat a zpracovávat data ze vstupů a také generovat DMX data na výstup. Dále bude spojený s SD kartou, která za nízkou cenu umožňuje ukládat velké množství dat. Komunikace s výstupním blokem DMX bude probíhat pomocí sběrnice UART, jejíž přenosová rychlost bude pevně nastavená na 250 kbit/s. Dále bude potřeba převést TTL signál na standart EIA-485 (RS485), který je fyzickou vrstvou protokolu DMX. Tento převod bude obstarán specializovaným obvodem.

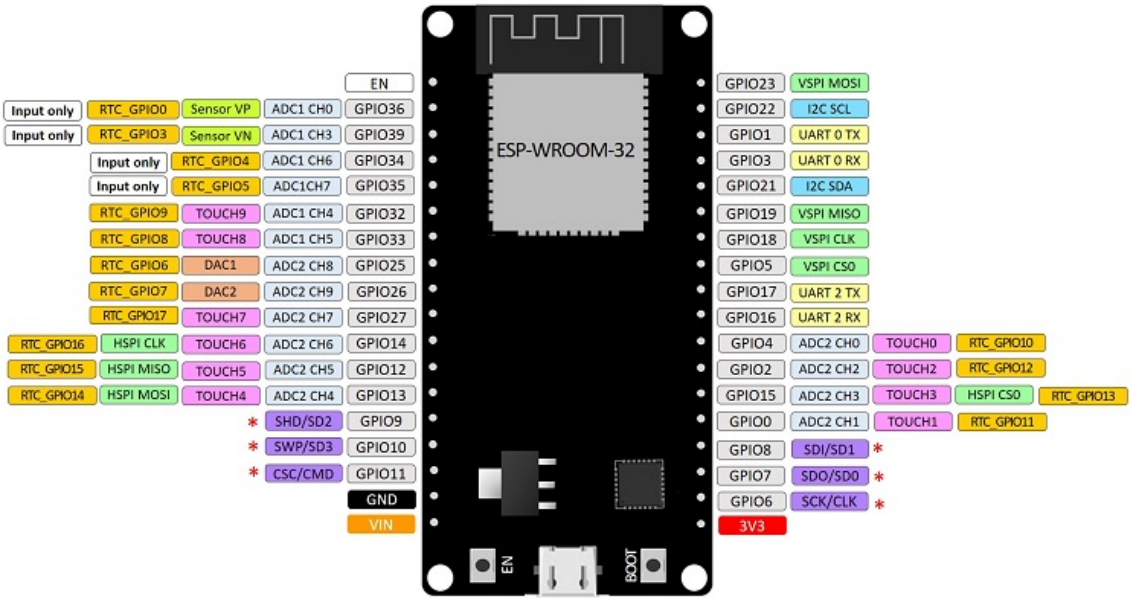
V první verzi návrhu byl zvolen čip ATmega32U4 od firmy Atmel. Ten podporuje připojení k USB, ovšem kromě samotného USB disponuje pouze jednou sběrnicí UART, což je pro naši aplikaci bohužel nedostatečné. Dále disponuje pouze jednou sběrnicí SPI (Serial Peripheral Interface) [15].

Další možnost, která by našemu použití plně vyhovovala bylo využití čipu ATxmega32C3 od firmy Atmel, který také disponuje plnohodnotným USB a dále má 2 sběrnice SPI a kromě podpory USB také až dvě periferie UART. Jediným problémem byla limitní rychlost přenosu na sběrnici UART, 250 kb/s. Bohužel je v době navrhování zařízení tento čip celosvětově nedostupný, tudíž nemohl být použit. [16]

### 5.2.2 ESP32

Z důvodů uvedených v předchozí podkapitole byl použit mikrokontrolér ESP32. ESP32 je mikrokontrolér založený na jádře procesoru Xtensa LX6 od společnosti Espressif. Je vybaven dvěma jádry procesoru a podporuje Wi-Fi a Bluetooth. Pro vysílání Wi-Fi a Bluetooth je již mikrokontrolér osazen i anténou. ESP32 může být použit jako samostatný mikrokontrolér, nebo v rámci větších zařízení jako příslušenství. Je vybaven řadou pinů GPIO (General Purpose Input Output), kterým lze funkce nastavit v kódu samotného EPS32. Tyto piny lze nastavit tak, že EPS32 disponuje až 3 sběrnicemi UART, 3 SPI sběrnicemi a 18 ADC (Analog-to-Digital) kanály. Periferií má tedy pro naše použití dostatek. ESP32 je také podporován širokou škálou programovacích jazyků a má aktivní komunitu uživatelů, která pomáhá s vývojem a poskytuje podporu pro nové uživatele. Rozložení pinů ESP32 Development board je možno vidět na obrázku. 5.2. [17]

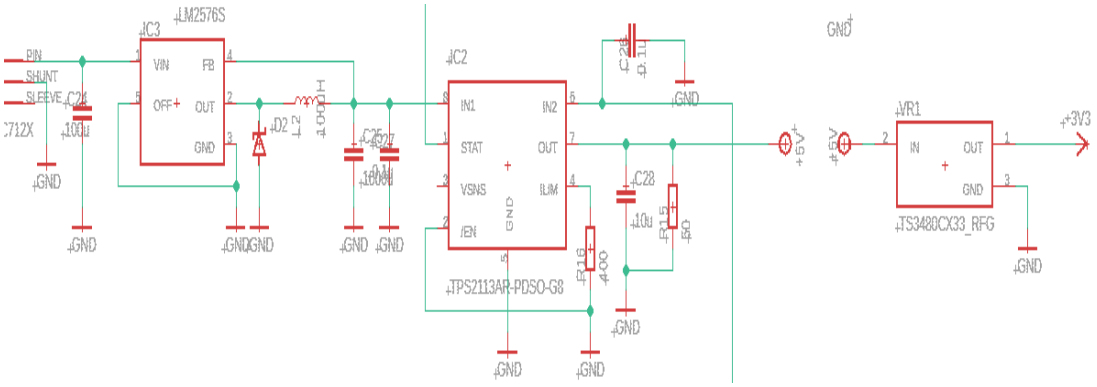
ESP32 je také známý svou nízkou spotřebou energie a schopností fungovat na baterie. To je důležité pro aplikace, které vyžadují dlouhou výdrž baterie nebo pro aplikace, které se používají v prostředích s omezeným přístupem k elektrické energii. Díky skvělé konektivitě a podpoře Wi-Fi a Bluetooth se ESP32 také často využívá na projekty takzvané chytré domácnosti. Dokáže se na něm také navrhnout například domácí server.




Obr. 5.2: Piny desky ESP32devboard

### 5.2.3 Napájení obvodu

Napájení obvodu je řešeno dvojím způsobem. Konkrétně lze obvod napájet přes USB nebo přes dedikovaný konektor pro 5V. Oba zdroje jsou připojeny přes součástku TPS2113A od firmy Texas instruments. Tato součástka je zapojena tak, aby při připojení obou zdrojů byl zvolen zdroj z dedikovaného konektoru napájení, pokud na vstupu „IN1“ není zjištěno napětí, je automaticky připojeno jako zdroj USB. Dále je potřeba napěťový převod, protože ESP32 pro správnou funkčnosti potřebuje 3V. O napěťový převod se stará součástka LM2576.




Obr. 5.3: Zapojení napájení obvodu

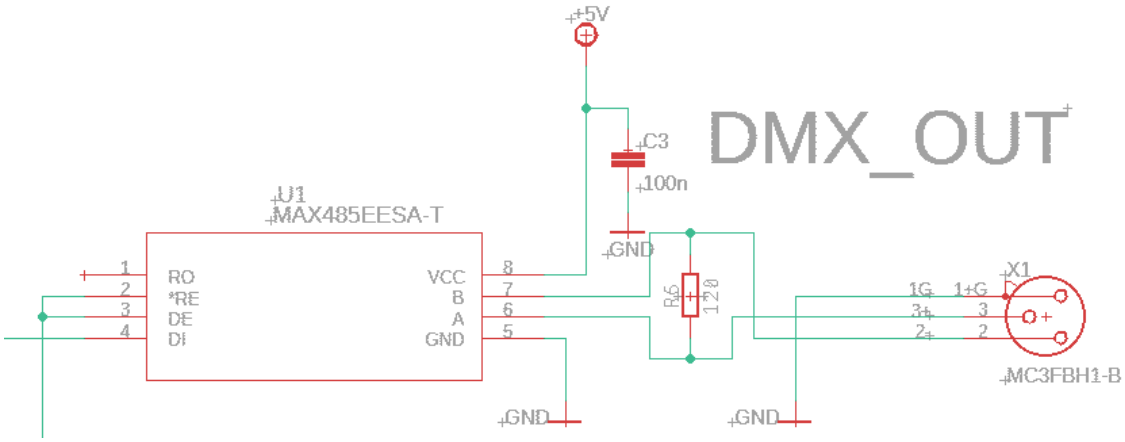
### 5.2.4 Budič sběrnice RS485

Signál DMX generovaný miktrokontrolérem je vyveden pomocí sběrnice UART s napětím 0 V pro logickou 0 a 3,3 V pro logickou 1. Jak bylo popsáno v první kapitole této práce, fyzická vrstva protkolu DMX512 je tvořena průmyslovým standardem pro sériovou komunikaci EIA-485 (RS485). Proto je pro převod na tento standard použit obvod MAX485 od firmy Maxim Integrated.

Jako ekvivalent tohoto obvodu zde mohl být použit například obvod ADM485 od firmy Exar Corporation nebo obvod SN75176B od firmy Texas Instruments. Tyto obvody mají obdobnou funkci jako obvod MAX485, ten byl zvolen zejména z důvodu jeho dostupnosti.

Zapojení obvodu MAX485 je na obrázku 5.4. Do obvodu je přivedeno napájecí napětí 5 V, které je u obvodu blokované kondenzátorem C1. Výrobce v datovém listu uvádí, že minimální vstupní napětí, které obvod vnímá jako log. 1, jsou 2 V. To znamená, že do obvodu může být přiveden signál přímo z UART sběrnice mikrokontroléru, bez nutné napěťové konverze. Řídící piny DE (Data Enable) a RE (Reciever Enable) tohoto obvodu jsou připojeny na signál EN (Enable signal). Měněním hodnoty tohoto signálu přepíná mikrokontroler funkci tohoto budiče z přijímače

na vysílač. Pro impedační přizpůsobení sběrnice je mezi výstupní svorky A a B obvodu připojen rezistor R1 o odporu 120 Ω. Po tomto impedančním přizpůsobení je signál vyveden na DMX konektor.

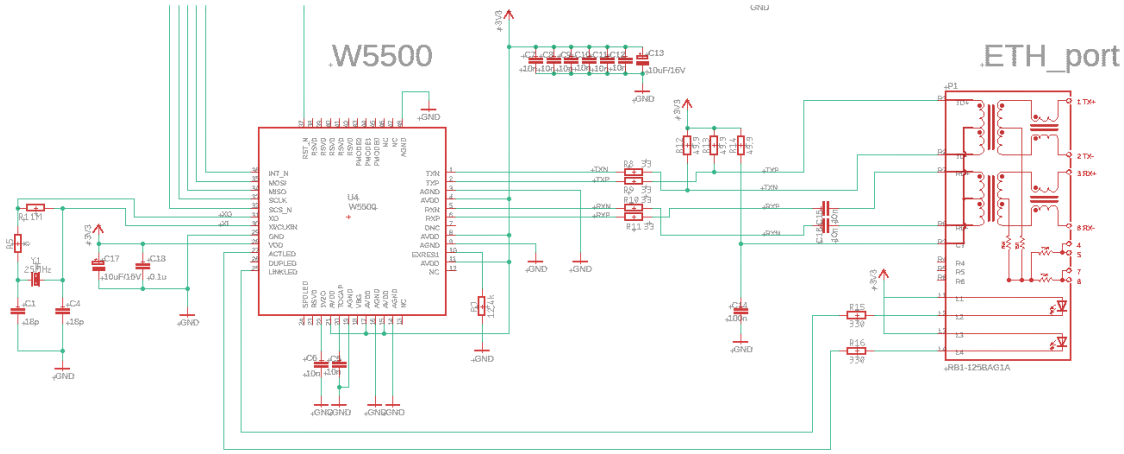



Obr. 5.4: Zapojení obvodu MAX485

### 5.2.5 Ethernet kontrolér

S úmyslem vytvořit zařízení, které bude schopné číst data protokolu Art-Net jej bylo potřeba osadit obvodem pro ethernetovou komunikaci. K tomu byl zvolen čip W5500 od firmy WIZnet. W5500 je jednočipový ethernetový kontrolér, který umožňuje snadné připojení k internetu pro systémy založené na mikrokontrolérech. Díky specifikacím W5500 je tento čip schopen obstarat zpracování protokolu Art-Net a nezatěžovat mikrokontrolér těmito úkony. Přestože je Wi-Fi modul obsažen přímo na ESP32, byla potřeba možnost zapojení ethernetového kabelu s konektorem RJ45. Proto byl tento čip v návrhu propojen s RJ45 konektorem a přes komunikační sběrnici SPI spojen s mikrokontrolérem ESP32. W5500 má také implementované metody k úspoře energie. Je to například WOL (Wake on LAN), kdy čip pracuje pouze pokud přijímá signál a PDM (Power Down Mode) – po stanovené době, kdy nepřijímá žádná data sám přestane pracovat bez nutnosti odpojení od napájení. [18]

Zmíněné zapojení je možno vidět na obrázku 5.5. Čip WIZnet W5500 je s mikrokontrolérem ESP32 spojen přes sběrnici SPI, čemuž odpovídají signály MOSI pro vysílání komunikace do mikrokontroléru a MISO pro přijímání pokynů od ESP32. Dále pak časovací signál SCLK, který je vysílán mikrokontrolérem a celou tuto komunikaci časuje. Piny čipu W5500, které dále vedou do ethernetového portu, jsou TXP a TXN pro vysílání dat a RXP a RXN pro přijímání dat z ethernetového kabelu. Na čip wiznet je také připojen signál z krystalového oscilátoru.

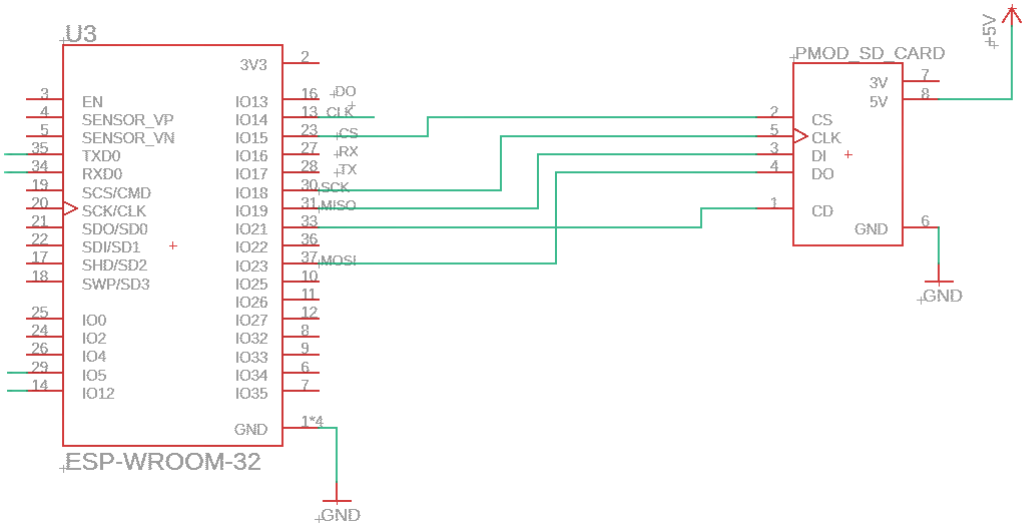



Obr. 5.5: Zapojení čipu WIZnet W5500

### 5.2.6 MicroSD karta

Jak již bylo zmíněno, pro ukládání dat bude využita microSD karta. Ta je ideálním řešením vzhledem k nízké pořizovací ceně a velké paměti. MicroSD je formát výměnné flashové paměťové karty odvozen od formátu SD. V současnosti se jedná o rozměrově nejmenší dostupný formát tohoto typu paměťových karet. Rozměry takovéto karty jsou 15 × 11 × 1 mm a její váha je zhruba 0,25 g. Port pro microSD karty se dá spojit s mikrokontrolérem přes sběrnici SPI, kterých má ESP32 dostatek. I proto byla tato forma uložiště vyhovující pro tuto aplikaci.

Na obrázku 5.6 je zobrazeno připojení slotu pro microSD paměťové karty na komunikační sběrnici SPI. Slot mircoSD karty je napájen 5 V. S mikrokontrolérem komunikuje přes signály DI (Data In) a DO (Data Out), které časuje signál CLK. Signály DI a DO odpovídají signálům MISO a MOSI protokolu SPI. Dále je připojen do ESP32 signál CD (Card Detect), který detekuje připojení mircoSD karty. Pokud je microSD karta vložena, signál klesne na hladinu LOW, které odpovídá napětí 0 V.

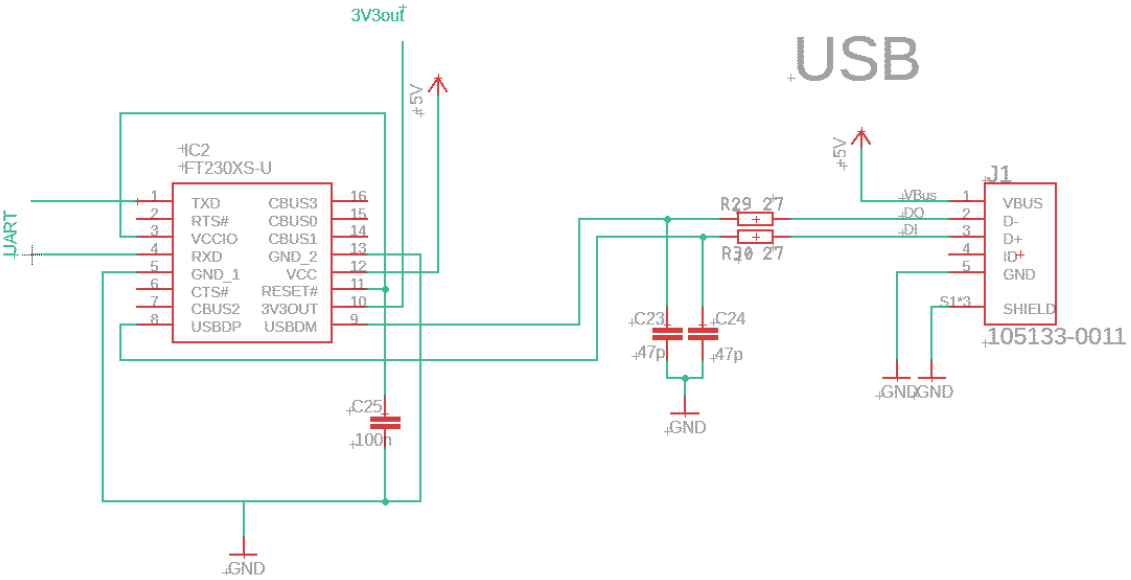



Obr. 5.6: Zapojení slotu pro microSD kartu

### 5.2.7 Připojení konektoru USB

Pro správné fungování celého zařízení je třeba zajistit propojení s osobním počítačem, který bude softwarově řídit chod mikrokontroléru. Pro takové spojení je třeba osadit konektor USB. Konektorů USB existuje celá řada, typ USB (například A, B nebo Micro-A) určuje druh použitého konektoru. Dále se uvádí standard USB (například USB 1.0, USB 2.1, USB 3.0), ten uvádí rychlost přenosu dat.

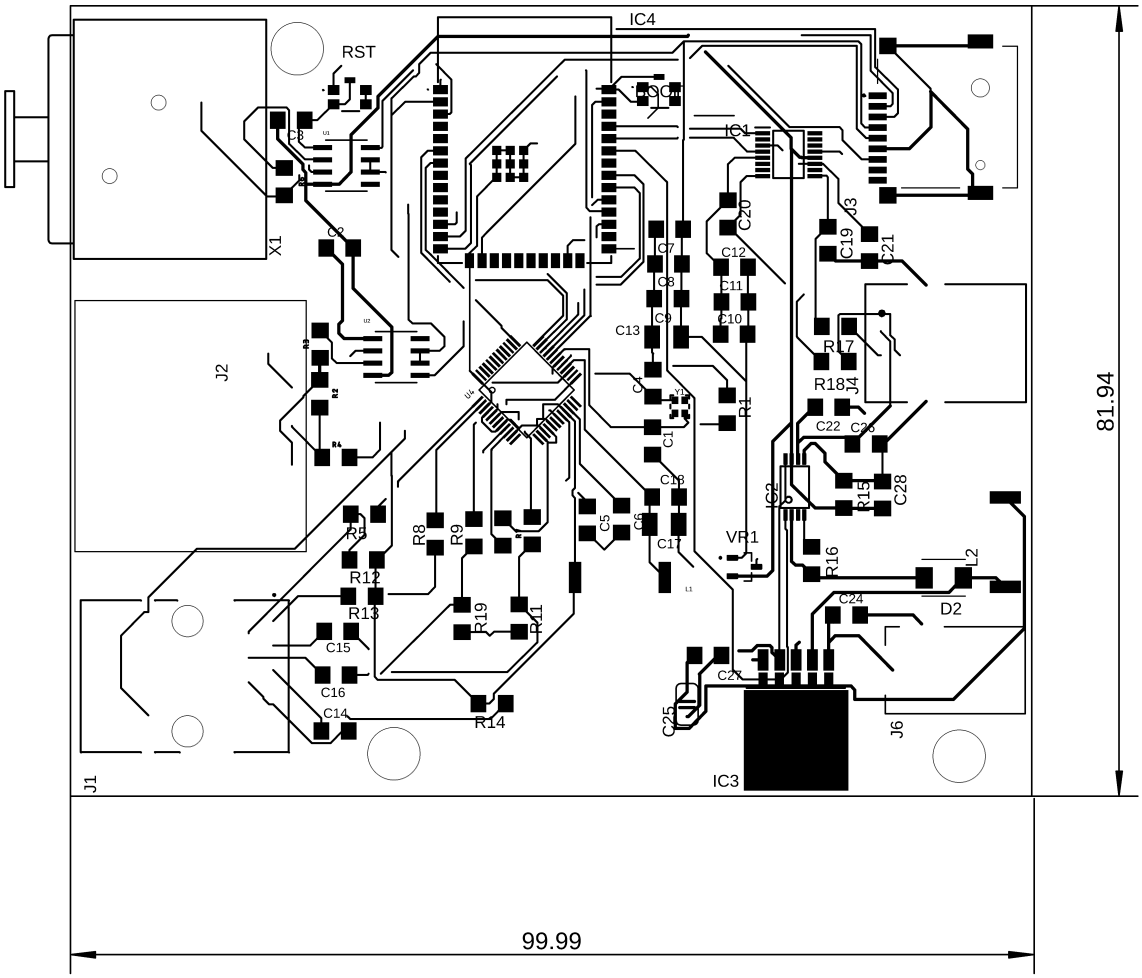
Pro zařízení bylo zvoleno USB typu B. Tento typ je sice oproti micro USB rozměrově větší alternativou, ale jeho robustní konstrukce byla přednější. USB je na ESP32 připojeno přes komunikační rozhraní UART. K fungování tohoto připojení je třeba osadit také specializovaný obvod pro převod standardu UART na standard USB. V tomto konkrétním případě byl použit obvod FT230XS-U od firmy Future Technology Devices International. Byl zvolen, protože podporuje standard USB 2.0. Konkrétní zapojení USB rozhraní lze vidět na obrázku 5.7. Piny konektoru USB D+ a D- slouží k přenosu samotných dat. Tyto piny jsou připojeny USBDP (USB Data Plus) a USBDM (USB Data Minus) obvodu FT230XS-U. Tento obvod je připojen na rozhraní UART přes piny TXD (Transmit Data) a RXD (Recieve Data). [19]




Obr. 5.7: Zapojení USB rozhraní

## 5.3 Deska plošných spojů

Samotná deska plošných spojů byla navržena ve studentské verzi programu Autodesk EAGLE. Plošné spoje jsou na oboustrané desce z klasického materiálu. Napájení je vedeno tloušťkou 0,4 mm pro 5V a 0,3 mm pro 3V. Ostatní signálové spoje jsou vedeny tloušťkou 0,2 mm. Součástky jsou osazeny kombinovanou metodou, větší část desky tvoří SMD součástky, zejména v pouzdrech 1206. Některé součástky, především v napájecím obvodu, jsou ve formě THT součástek. Většina spojů je vedena ve vrchní vrstvě DPS. Na spodní vrstvě DPS je zemní vrstva pro efektivní uzemnění součástek. Vrchní vrstva desky je zobrazena na obrázku 5.8. Levou stranu desky tvoří vstup a výstup DMX a ethernetový port. Ve vrchní části pravé strany desky se nachází slot pro microSD kartu. Pod ním je umístěn USB konektor. Ve spodní části desky je konektor pro přivedení napájení.




Obr. 5.8: Hladina top DPS

# 6 Software pro čip ESP32

Kód pro mikrokontrolér ESP32 byl vyvíjen v rozšíření programu Visual Studio Code PlatformIO. PlatformIO nabízí řadu funkcí, které se snaží zpříjemnit jednotlivé kroky vývoje. Toto rozšíření umožňuje přímé načtení knihoven do jednotlivých projektů a také velké množství předdefinovaných mikrokontrolérů. Proto si stačí vybrat s jakým hardwarem pracujeme.

Jak již bylo zmíněno, ESP32 podporuje více programovacích jazyků. Po zvolení desky v PlatformIO máme možnost zvolit si framework tohoto kódu. Na výběr je k dispozici espresifIDF nebo Arduino [20]. Pro tento konkrétní případ byl zvolen framework Arduino. I přes to, že obecně je framework arduino pomalejší, bylo nutné jej zvolit, protože autor knihovny *esp-dmx* varoval před možnými chybami knihovny právě při použití frameworku espresifIDF [21].

## 6.1 Struktura kódu pro ESP32

Pro správné fungování kódu je nutno na začátku importovat knihovny. V tomto konkrétním případě jde o knihovny *arduino* a *esp-dmx*. Knihovna *arduino* je nutná pouze proto, že jsme si zvolili framework Arduino. Dále je nutné uvést použité funkce. Toto je typické pro prostředí založené na programovacím jazyku C. Pro přehlednost zde funkce nejsou definované, ale musí být zmíněny. Následuje deklarace a inicializace proměnných *datarx* a *datatx*, do kterých se následně zapisují data pro přenos DMX. Krom těchto proměnných je také definována proměnná *lastupdate*, která určuje čas poslední aktualizace.

Další částí kódu je *setup*. Tato část je typická pro framework Arduino a probíhá vždy jen jednou po aktivování zařízení. V této části je zahájena sériová komunikace a dále je GPIO pinům ESP32 přiřazen režim ve kterém operují. Ve výpisu 6.1 lze vidět nastavení pinu 16 jako vstup a 17 jako výstup. Dále je zde vidět přiřazení DMX portu a pinů které jsou pro tuto komunikaci využity.

Výpis 6.1: Setup kódu pro ESP32

```
void setup() {
  Serial.begin(115200);
  pinMode(dmxrx, INPUT);
  pinMode(dmxtx, OUTPUT);
  pinMode(enablePin, OUTPUT);

  dmx_set_pin(dmxPort, dmxtx, dmxrx, enablePin);
  dmx_driver_install(dmxPort, DMX_DEFAULT_INTR_FLAGS);
```

Po části setup následuje nekonečný cyklus, ve kterém zařízení čeká na příchozí data ze sériového portu a provádí příslušné akce na základě přijatých příkazů. Zařízení má dva operační módy. Jeden je určen pro přijímání DMX dat a druhý je pro jejich vysílání. Ve výpisku 6.1 jsou zobrazeny funkce přepínající režimy zařízení.

Výpis 6.2: Funkce pro přepínání operačních režimů

```
void switchToRX() {
  digitalWrite(enablePin, LOW);
  Serial.println("Režim přijímání aktivován");
}

void switchToTX() {
  digitalWrite(enablePin, HIGH);
  Serial.println("Režim vysílání aktivován");
}
```

### 6.1.1 Režim přijmu dat

V režimu přijímání dat se volá funkce *switchToRX()*, která změní hodnotu pinu řídícího integrovaný obvod MAX485 a uvede zařízení do režimu přijímání dat. Pokud jsou v tomto režimu přijata data, probíhá jejich kontrola. Pokud nejde o relevantní DMX data, zařízení zahlásí chybu, která se vypíše na sériový port. Pokud příjem dat proběhne v pořádku jsou následně zapsána do proměnné *datarx*. Tato data jsou pak každou sekundu aktualizována a posílána na sériový port. Pokud dojde k odpojení DMX, vypíše se chybová zpráva a zařízení opustí režim přijímání dat.

### 6.1.2 Režim vysílání dat

V režimu vysílání se volá funkce *switchToTX()*, která změní hodnotu řídícího pinu a přepne zařízení do režimu vysílání dat. Poté následuje smyčka, ve které zařízení čeká na příkazy přicházející ze sériového portu. Podle příkazu se nastaví hodnoty DMX signálu, které se uloží do proměnné *datatx*. Tato data se následně vysílají. Příkaz *N* nastaví všechna data na hodnotu 0. Příkaz *H* nastaví všechna data na poloviční hodnotu (128). Příkaz *F* nastaví všechna data na maximální hodnotu (255). Tyto příkazy jsou generovány tlačítky v uživatelském rozhraní ovládacího programu v počítači.

Pokud je v průběhu vysílání přijat jiný řídící příkaz, je aktualizována proměnná *datatx* a jsou odeslána nová data. Zařízení také každou sekundu posílá hodnoty prvních pěti vysílaných kanálů DMX na sériový port. Tyto hodnoty jsou poté zobrazeny v uživatelském rozhraní. Příkaz *S* ukončuje režim vysílání.

# 7 Program pro osobní počítač

## 7.1 Funkce programu

Pro ovládání zařízení je třeba vytvořit aplikaci pro osobní počítač, která bude zpracovávat přijaté informace z převodníku a bude tyto informace zobrazovat uživateli. Pro zajištění maximální přehlednosti je vhodné, aby tato aplikace měla poměrně jednoduché UI (User Interface). Dále je třeba z této aplikace vysílat pokyny samotnému převodníku. Přijaté zprávy i pokyny putují sériovou komunikací přes USB.

### 7.1.1 Prostředí pro tvorbu UI

Pro vývoj této aplikace jsem zvolil programovací jazyk Python. Konkrétně byla tato aplikace vyvíjena v PyCharm Community IDE. PyCharm Community je volně dostupná verze vývojového prostředí od společnosti JetBrains. Je primárně určená pro psaní v jazyce Python a nabízí řadu funkcí usnadňující vývoj, jako například zvýrazňování chyb syntaxu, automatické doplňování rozepsaných funkcí nebo integraci se systémem Github.[0]

## 7.2 Struktura programu pro osobní počítač

Nejprve je třeba importovat knihovny v Pythonu nazvané modules, které nám poskytují potřebné funkce pro návrh tohoto UI. V tomto případě se zde importují modul *serial*, který poskytuje potřebné funkce pro zavedení sériové komunikace a modul *tkinter*, který je použit pro tvorbu UI pro ovládáni tohoto programu. Dále je nastaven port, přes který bude probíhat sériová komunikace a její baudrate. Dále jsou definovány funkce samotného programu. Nakonec je pomocí knihovny *tkinter* vytvořeno UI prostředí a jednotlivá tlačítka, která po stisknutí volají dříve definované funkce.

### 7.2.1 Kód programu pro osobní počítač

Po konfiguraci sériové komunikace je třeba definovat funkce zodpovědné za připojení a odpojení této komunikace. Definování těchto funkcích lze vidět ve výpisu 7.2.1. Funkce *connect* se po stlačení tlačítka „Připojit“ pokusí navázat sériovou komunikaci o dříve definované rychlosti této komunikace. Pokud je komunikace zavedena, zpřístupní tlačítka pro posílání příkazů zařízení. Funkce *disconnect* tuto komunikaci po stlačení tlačítka „Odpojit“ přeruší a zablokuje tlačítka pro posílání informací zařízení.

Výpis 7.1: Funkce sériové komunikace

```
def connect_serial():
    global ser
    try:
        ser = serial.Serial(SERIAL_PORT, BAUD_RATE,
           timeout=TIMEOUT)
        status_label.config(text="Připojeno␣k␣sériovému␣
           portu")
        disconnect_button.config(state=tk.NORMAL
        button_receive.config(state=tk.NORMAL)
        button_transmit.config(state=tk.NORMAL)
        button_full.config(state=tk.NORMAL)
        button_half.config(state=tk.NORMAL)
        button_null.config(state=tk.NORMAL)
    except serial.SerialException:
        status_label.config(text="Chyba␣při␣připojování␣k
           ␣sériovému␣portu")

def disconnect_serial():
    global ser
    if ser:
        ser.close()
        ser = None
        status_label.config(text="Odpojeno␣od␣sériového␣
           portu")
        disconnect_button.config(state=tk.DISABLED)
        button_receive.config(state=tk.DISABLED)
        button_transmit.config(state=tk.DISABLED)
        button_full.config(state=tk.DISABLED)
        button_half.config(state=tk.DISABLED)
        button_null.config(state=tk.DISABLED)
```

Po úspěšném připojení sériové komunikace můžeme pomocí tlačítek „Vysílač“ a „Přijímač“ přepínat režim nastavení zařízení. Tato tlačítka využívají funkci *send-command*, která při stlačení tlačítka „Vysílač“ pošle po sériové komunikaci znak *D*. Zařízení se po obdržení tohoto znaku přepne do režimu vysílání a vyzve uživatele k zadání hodnot pro vysílání DMX signálu. K zadání těchto hodnot slouží tři tlačítka. Po stisku jednoho z nich je opět vyslán znak, který zařízení určuje, jaký signál bude vysílán. Například tlačítko „Plný“ pošle znak *F*. Když zařízení tento

znak přijme začne vysílat hodnotu 255 na všechny kanály DMX. Kód pro vytvoření zmiňovaných třech tlačítek můžeme vidět na kódovém výpisu 7.2.1. Pro zastavení vysílání a opuštění režimu vysílání slouží tlačítko „STOP“. Rozložení těchto tlačítek je možné vidět na obrázku 7.1.

Výpis 7.2: Definování ovládacích tlačítek

```
def connect_serial():
    global ser
    # Vytvoření tlačítka pro odeslání 'F'
button_full = tk.Button(char_frame, text="Plný", command=
   lambda: send_char('F'), state=tk.DISABLED)
button_full.pack(side=tk.LEFT, padx=5)

# Vytvoření tlačítka pro odeslání 'H'
button_half = tk.Button(char_frame, text="Půl", command=
   lambda: send_char('H'), state=tk.DISABLED)
button_half.pack(side=tk.LEFT, padx=5)

# Vytvoření tlačítka pro odeslání 'N'
button_null = tk.Button(char_frame, text="Nulový",
   command=lambda: send_char('N'), state=tk.DISABLED)
button_null.pack(side=tk.LEFT, padx=5)
```


Obr. 7.1: Rozložení tlačítek v UI

Celé UI je možné vidět na obrázku 7.2. Tento obrázek zobrazuje zařízení v režimu vysílání, kdy pro kontrolu funkce zařízení posílá hodnoty prvních pěti DMX kanálů do sériového monitoru, který se zobrazuje v samotném UI. Na posledním řádku v uživatelském rozhraní vidíme, že zařízení vypsalo zprávu „Režim vysílání ukončen“, protože bylo stlačeno tlačítko „STOP“.


Obr. 7.2: Uživatelské rozhraní aplikace

# Závěr

Tato bakalářská práce v prvních kapitolách popisuje komunikační protokol DMX, rámce jeho signálu a typické zapojení jednotlivých zařízení na DMX sběrnici. Poté je popsán duplexní nástavba na protokol DMX a to RDM. Následně je popsán protkol Art-Net. Dále pak práce popisuje komunikační protokoly běžně využívané v systémech, které jsou postaveny na mikrokontrolérech. Konkrétněji rozebírá rozhraní UART a sériový komunikační protokol SPI.

Následně práce popisuje návrh DMX převodníku s USB portem pro komunikaci s osobním počítačem. Tento návrh je postaven na mikrokontroléru ESP32. Krom USB portu je DMX převodník osazen také vstupem a výstupem DMX, ethernetovým portem a čipem pro zpracování ethernetového signálu WIZnet W5500 a slotem pro microSD paměťovou kartu. Tato práce dále popisuje schématický návrh zapojení takového zařízení a výběr součástek pro jeho sestrojení. Výstupem této práce je kompletní schématický návrh převodníku a návrh desky plošných spojů v programu Autodesk Eagle.

V další části se pak práce věnuje návrhu kódu pro mikrokotrolér ESP32 v jazyce C++. Funkčnost tohoto programu je v práci popsána. Tento kód je napsaný ve frameworku arduino a umožňuje vysílání a přijimání zpráv DMX. Dále také zajišťuje komunikaci s USB portem osobního počítače. Z tohoto počítače mikrokontrolér přijímá příkazy, které řídí jeho funkci.

Následně práce popisuje tvorbu řídícího programu pro osobní počítač s grafickým rozhraním. Ten je psán v jazyce python ve vývojovém prostředí PyCharm Community. Tento program je schopen zobrazovat data, která jsou přivedena na DMX vstup zařízení a řídit generování zpráv na DMX výstup. Oba tyto programy a jejich interkce byli testovány na desce ESP32 Development board.

# Literatura


[1] Entertainment Services and Technology Association. Asynchronous serial digital data transmission standard for controlling lighting equipment and accessories. [cit. 22-10-7].

[2] Wayne Howell. *Control freak: A real world guide to DMX512 and Remote Device Management*. Entertainment Technology Press, 2012. [cit. 22-10-5].

[3] J. Pihrt. Teorie protokolu dmx512, 2016. [cit. 22-11-12]. URL: `http://www.soh.cz/podpora/teorie`.

[4] Gerard J. Informations and facts about dmx 512 protocol. 2022. [cit. 22-10-5].

[5] Petr Průcha. Dmx řízení techniky: Za světlem s drátem i bez drátu. [cit. 22-11-3]. URL: `https://www.frontman.cz/dmx-rizeni-techniky-za-svetlem-s-dratem-i-bez-dratu`.

[6] What is rdm? [cit. 22-11-13]. URL: `https://www.rdmprotocol.org/rdm/what-is-rdm/`.

[7] Johnatan Trust. Art-net, Jul 2020. [cit. 22-11-11]. URL: `https://art-net.org.uk/`.

[8] Wayne Howell. *Light Bytes: Inside Art-Net and sACN*. Artistic Licence, 2017. [cit. 22-11-5].

[9] Umakanta Nanda and Sushant Kumar Pattnaik. Universal asynchronous receiver and transmitter (uart). In *2016 3rd International Conference on Advanced Computing and Communication Systems (ICACCS)*, volume 01, pages 1–5, 2016. `doi:10.1109/ICACCS.2016.7586376`.

[10] Frank Durda. Serial and uart tutorial. *FreeBSD Documentation*, 1996. [cit. 22-11-10].

[11] MANISH Kundu and ABHIJEET Kumar. A review on low power spi protocol. *International Journal of VLSI Design and Communication Systems (IJVDCS)*, 2:193–195, 2014. [cit. 22-11-10].

[12] S Choudhury, GK Singh, and RM Mehra. Design and verification serial peripheral interface (spi) protocol for low power applications. *International Journal of Innovative Research in Science, Engineering and Tecgnology*, pages 16750–16758, 2014. [cit. 22-11-11].

[13] Inc. Motorola. Spi block guide, 2001. [cit. 23-5-5].

[14] Li Li Li, Jing Yu He, Yong Peng Zhao, and Jian Hong Yang. Design of microcontroller standard spi interface. 618:563–568, 2014. [cit. 23-4-5].

[15] Atmel atmega16u4, atmega32u4 datasheet summary, 2016. [cit. 22-11-16]. URL: https://ww1.microchip.com/downloads/en/DeviceDoc/Atmel-7766-8-bit-AVR-ATmega16U4-32U4_Summary.pdf.

[16] Microchip technology, 2014. [cit. 22-11-13]. URL: https://ww1.microchip.com/downloads/aemDocuments/documents/OTH/ProductDocuments/DataSheets/Atmel-8492-8-and-16-bit-AVR-microcontroller-ATxmega32C3_64C3_128C3_192C3_256C3_datasheet.pdf.

[17] Rui Santos. Esp 32 pinout refference. [cit. 22-11-13]. URL: https://i0.wp.com/randomnerdtutorials.com/wp-content/uploads/2018/08/ESP32-DOIT-DEVKIT-V1-Board-Pinout-36-GPIOs-updated.jpg?quality=100&strip=all&ssl=1.

[18] WIZnet. Overview of wiznet 5500, 2018. [cit. 22-11-13]. URL: https://docs.wiznet.io/Product/iEthernet/W5500/overview.

[19] Jan Axelson. Serial port complete second edition: Com ports, usb virtual com ports, and ports for embedded systems. *Lakeview Research LLC*, 2007. [cit. 23-4-15].

[20] Systems Espressif. Esp-idf, 2019. [cit. 23-4-5]. URL: https://www.espressif.com/en/products/sdks/esp-idf.

[21] Mitch Weisbrod. Someweisguy/esp$_d mx$ : $Espressif esp32 implementation of ansi - estae1.11 dmx - 512 aande1.20 rdm, 2023. [cit. 23 - 5 - 16]. URL$ : .

Jetbrains. Pycharm: The python ide for professional developers by jetbrains. [cit. 23-4-5]. URL: https://www.jetbrains.com/pycharm/.

# A Seznam součástek

| Název | Hodnota | Pouzdro | Popis |
|---|---|---|---|
| C1 | 18p | C1206 | Kondenzátor |
| C2 | 100n | C1206 | Kondenzátor |
| C3 | 100n | C1206 | Kondenzátor |
| C4 | 18p | C1206 | Kondenzátor |
| C5 | 10n | C1206 | Kondenzátor |
| C6 | 10n | C1206 | Kondenzátor |
| C7 | 10n | C1206 | Kondenzátor |
| C8 | 10n | C1206 | Kondenzátor |
| C9 | 10n | C1206 | Kondenzátor |
| C10 | 10n | C1206 | Kondenzátor |
| C11 | 10n | C1206 | Kondenzátor |
| C12 | 10n | C1206 | Kondenzátor |
| C13 | 10uF/16V | CPOL-EUSMCB | Kondenzátor |
| C14 | 100n | C1206 | Kondenzátor |
| C15 | 10n | C1206 | Kondenzátor |
| C16 | 10n | C1206 | Kondenzátor |
| C17 | 10uF/16V | CPOL-EUSMCB | Kondenzátor |
| C18 | 0,1u | C1206 | Kondenzátor |
| C19 | 47p | C1206 | Kondenzátor |
| C20 | 47p | C1206 | Kondenzátor |
| C21 | 100n | C1206 | Kondenzátor |
| C22 | 100u | C1206 | Kondenzátor |
| C23 | 100u | C1206 | Kondenzátor |
| C24 | 100u | C1206 | Kondenzátor |
| C25 | 1000u | C025-024X044 | Kondenzátor |
| C26 | 0,1u | C1206 | Kondenzátor |
| C27 | 0,1u | C1206 | Kondenzátor |
| C28 | 10u | C1206 | Kondenzátor |
| D2 | | | dioda |
| IC1 | | | FT230XS |
| IC2 | | | TPS2113 |

| Název | Hodnota | Pouzdro | Popis |
|---|---|---|---|
| IC3 | LM2576S | TO263-5 | Kondenzátor |
| IC4 | ESP32 | | Kondenzátor |
| J1 | 100n | | Konektor RJ45 |
| J2 | NC3MD-H | | XLR konektor |
| J3 | 693071020811 | | Micro SD |
| J4 | 61400416121 | | USB 2.0 |
| J5 | RAPC712X | | 2.50mm konektor |
| R1 | 1M | R1206 | Rezistor |
| R2 | 120 | R1206 | Rezistor |
| R3 | 22 | R1206 | Rezistor |
| R4 | 22 | R1206 | Rezistor |
| R5 | 33 | R1206 | Rezistor |
| R6 | 120 | R1206 | Rezistor |
| R7 | 12.4K | R1206 | Rezistor |
| R8 | 33 | R1206 | Rezistor |
| R9 | 33 | R1206 | Rezistor |
| R10 | 33 | R1206 | Rezistor |
| R11 | 82 | R1206 | Rezistor |
| R12 | 49.9 | R1206 | Rezistor |
| R13 | 49.9 | R1206 | Rezistor |
| R14 | 10 | R1206 | Rezistor |
| R15 | 50 | R1206 | Rezistor |
| R16 | 400 | R1206 | Rezistor |
| R17 | 27 | R1206 | Rezistor |
| R18 | 27 | R1206 | Rezistor |
| R19 | 27 | R1206 | Rezistor |
| B1 | | | EVP-3AA02Q |
| B2 | | | EVP-3AA02Q |
| U1 | | | 5500 |
| VR1 | | | TS3480CX33 |
| X1 | | | XLR konektor |
| Y1 | | SMD-2.0X1.6MM | 25MHz Crystal |

# B Obsah elektronické přílohy

Soubory ESP_DMX_Vahalik.brd a ESP_DMX_Vahalik.sch jsou soubory programu EAGLE, první z nich obsahuje návrh desky plošných spojů, druhý schéma tohoto návrhu. Soubor ESP32_code.cpp je program pro mikrokontrolér ESP32 a ESP32_UI.py je uživatelské rozhraní v jazyku Python.

```
/ ..... kořenový adresář přiloženého archivu
├── ESP_DMX_Vahalik.brd
├── ESP_DMX_Vahalik.sch
├── ESP32_code.cpp
└── ESP32_UI.py
```