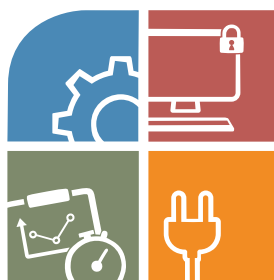


VYŠŠÍ ODBORNÁ ŠKOLA, STŘEDNÍ ŠKOLA,  
CENTRUM ODBORNÉ PŘÍPRAVY



## ABSOLVENTSKÁ PRÁCE

Rozhraní pro komunikaci programu  
Matlab/Simulink s externími zařízeními

Sezimovo Ústí, 2023

Autor: Miroslav Kubů





## ZADÁNÍ ABSOLVENTSKÉ PRÁCE

Student: **Miroslav Kubů**  
Obor studia: 26-41-N/01 Elektrotechnika – mechatronické systémy  
Název práce: **Rozhraní pro komunikaci programu Matlab/Simulink s externími zařízeními**  
Anglický název práce: **Interface for Communication of the Matlab/Simulink Program with External Devices**

### Zásady pro vypracování:

1. Proveďte analýzu současného řešení připojení externích zařízení ve školní Laboratoři aplikované informatiky k programu Matlab/Simulink.
2. Navrhněte a realizujte hardware pro komunikaci s programem Matlab/Simulink pomocí USB rozhraní, které umožní připojení binárních i analogových signálů externích zařízení.
3. Vytvořte software pro řídicí elektroniku a Matlab/Simulink, který implementuje USB komunikaci pro přenos dat mezi hardwarem a PC.
4. V prostředí Simulink navrhněte bloky, které zajistí integraci připojeného externího zařízení do simulinkového modelu.
5. Absolventskou práci vypracujte problémově ve struktuře odpovídající vědecké práci.

### Doporučená literatura:

- [1] MATHWORKS: Matlab & Simulink [online]. [cit. 2022-04-07]. Dostupné z: <https://nl.mathworks.com/>.
- [2] SELECKÝ, M. *Arduino: Uživatelská příručka*. Brno: Computer Press, 2016. ISBN 9788025148402.

Vedoucí práce: Ing. Jiří Roubal, Ph.D., VOŠ, SŠ, COP Sezimovo Ústí  
Odborný konzultant práce: Mgr. Jakub Macillis, VOŠ, SŠ, COP Sezimovo Ústí  
Oponent práce: Mgr. Bc. Miroslav V. Hospodářský, VOŠ, SŠ, COP Sezimovo Ústí

Datum zadání absolventské práce: **2. 9. 2022**

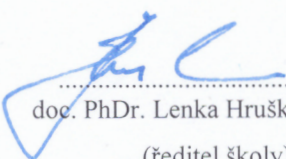
Datum odevzdání absolventské práce: **12. 5. 2023**

  
Ing. Jiří Roubal, Ph.D.

(vedoucí práce)



V Sezimově Ústí dne 2. 9. 2022

  
doc. PhDr. Lenka Hrušková, Ph.D.

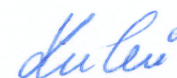
(ředitel školy)



## Prohlášení

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

V Sezimově Ústí dne 28. 4. 2023



\_\_\_\_\_ podpis

## Poděkování

Na tomto místě bych rád poděkoval Ing. Jiřímu Roubalovi, Ph.D., za vedení absolventské práce a cenné připomínky a rady při vytváření této práce. V neposlední řadě bych rád poděkoval své rodině za podporu nejen při zpracování absolventské práce, ale i během celého studia.

## Anotace

Tato absolventská práce se zabývá návrhem a realizací rozhraní pro komunikaci programu Matlab/Simulink s externími zařízeními, které komunikuje s programem Simulink pomocí sběrnice USB. Práce popisuje některé současné možnosti připojení externích zařízení k programu Simulink a ukázky jejich praktických využití v simulinkových modelech. Hlavním záměrem práce je návrh vlastního řešení a praktická realizace rozhraní s využitím platformy Arduino, umožňující programu Simulink komunikovat s externími zařízeními pomocí sběrnice USB.

**Klíčová slova:** Matlab, Simulink, Arduino, pulzně šířková modulace, kvadraturní enkodér, digitální signál, analogový signál, sériová komunikace.

## Annotation

This graduate thesis deals with the design and implementation of an interface for Matlab/Simulink communication with external devices that communicate with Simulink via USB. The thesis describes some of the current options for connecting external devices to Simulink and examples of their practical use in Simulink models. The main purpose of the work is to design a custom solution and practical implementation of an interface using the Arduino platform, allowing Simulink to communicate with external devices using the USB.

**Key words:** Matlab, Simulink, Arduino, pulse width modulation, quadrature encoder, digital signal, analog signal, serial communication.





# Obsah

Seznam obrázků	ix
Seznam tabulek	xi
<b>1 Úvod</b>	<b>1</b>
<b>2 Připojení zařízení k Matlabu/Simulinku</b>	<b>3</b>
2.1 Měřicí karta MF 624	3
2.2 Knihovna Support Package for Arduino	5
<b>3 Realizace komunikačního rozhraní</b>	<b>7</b>
3.1 Hardwarová část navrhovaného rozhraní	10
3.1.1 Digitální vstupy a výstupy navrhovaného rozhraní	11
3.1.2 Analogové vstupy a výstupy	11
3.1.3 Kvadraturní enkodér	13
3.1.4 Pulzně šířková modulace	15
3.1.5 Napájecí zdroj pro navrhované rozhraní	17
3.1.6 Zapojení konektorů X1 a X2	18
3.2 Softwarová část navrhovaného rozhraní	20
3.2.1 Software pro Matlab	21
3.2.2 Bloky pro Simulink	23
3.2.3 Software pro Arduino	26
<b>4 Oživení rozhraní a uvedení do provozu</b>	<b>29</b>
4.1 Popis finálního výrobku	29
4.2 Výsledky získané při testování	31
<b>5 Závěr</b>	<b>37</b>

Literatura	40
A Obsah přiloženého DVD	I
B Použitý software	III
C Časový plán absolventské práce	V
D Rozpočet projektu	VII
E Schémata zapojení elektroniky	IX
F Seznam součástek	XV

# Seznam obrázků

2.1	Karta MF 624 . . . . .	4
2.2	Schéma pro komunikaci s laboratorním modelem . . . . .	4
2.3	Ukázka bločků Support Package for Arduino . . . . .	5
2.4	Použití knihovny Support Package for Arduino . . . . .	6
3.1	Arduino Mega . . . . .	7
3.2	Princip toku dat mezi simulinkovým modelem a Arduinem . . . . .	8
3.3	Blokové schéma hardwarové části navrhovaného rozhraní . . . . .	10
3.4	Napěťové úrovně pro TTL logiku . . . . .	11
3.5	Zapojení analogového výstupu navrhovaného rozhraní . . . . .	12
3.6	Zapojení analogového vstupu navrhovaného rozhraní . . . . .	12
3.7	Ukázka kvadraturního enkodéru . . . . .	14
3.8	Princip vytváření signálů kvadraturního enkodéru . . . . .	14
3.9	Příklad PWM signálu a vyfiltrovaného výstupního napětí . . . . .	15
3.10	Časový diagram Fast PWM módu . . . . .	16
3.11	Schéma zapojení napájecího zdroje . . . . .	17
3.12	Konektor D-Sub 37 female . . . . .	18
3.13	Konektor D-Sub 37 female – číslování vývodů . . . . .	19
3.14	Bločky pro integraci externích signálů do simulinkového modelu . . . . .	24
3.15	Simulinkový blok analogového vstupu . . . . .	24
3.16	Simulinkový blok analogového výstupu . . . . .	25
3.17	Simulinkový blok Init . . . . .	25
4.1	Finální zařízení – přední pohled . . . . .	30
4.2	Finální zařízení – zadní pohled . . . . .	30
4.3	Dialogové okno instalace ovladače pro CH340 . . . . .	31
4.4	Knihovna Simulinku s bločkem Simulation Pace . . . . .	32
4.5	Správce zařízení systému Windows . . . . .	33

4.6	Testovací simulinkový model . . . . .	33
4.7	Pravoúhlý signál 1 Hz na vstupu DIN . . . . .	34
4.8	Pravoúhlý signál 1 Hz na výstupu DOUT . . . . .	34
4.9	Generování PWM signálu 1 kHz, střída 25 % . . . . .	35
4.10	Data generovaná simulinkovým modelem pro DA výstup . . . . .	35
4.11	Naměřená analogová data na DA výstupu . . . . .	36

# Seznam tabulek

3.1	Teoretická převodní tabulka AD/DA . . . . .	13
3.2	Zapojení vývodů konektoru X1 . . . . .	19
3.3	Zapojení vývodů konektoru X2 . . . . .	20
D.1	Finanční rozpočet projektu . . . . .	VII
F.1	Seznam součástek propojovací desky . . . . .	XV
F.2	Seznam součástek desky analogových vstupů . . . . .	XVI
F.3	Seznam součástek desky analogových výstupů . . . . .	XVII
F.4	Seznam součástek desky napájecího zdroje . . . . .	XVII



# Kapitola 1

## Úvod

V dnešní době je simulace běžnou součástí vývoje různých zařízení. Před zahájením výroby, nebo stavby prototypu, je tak možno odhalit chyby již ve fázi vývoje. Pro simulace je dnes používáno mnoho softwarových produktů, nicméně program Matlab se stal jakýmsi standardem v mnoha inženýrských oborech. Matlab byl vytvořen profesorem Cleverem Molerem na konci sedmdesátých let dvacátého století a byl původně určen pro matematické výpočty (KREJČÍ, A. et al., 2018). V současnosti lze program Matlab a jeho nadstavbu Simulink, která slouží pro modelování a vizualizaci dynamických jevů (ROUBAL, J. et al., 2011), jednoduše rozšířit pro různé vědeckotechnické účely. Na Vyšší odborné škole v Sezimově Ústí byl například Matlab mnohokrát využit při návrhu učebních pomůcek pro výuku automatizace (ŠIKÝŘ, T., 2011; RABIŇÁK, P., 2014; BOŠTIČKA, J., 2014; PAVLÁT, P., 2015).

Matlab společně se Simulinkem dokáže softwarově generovat velké množství signálů, které lze použít v modelech simulace. Pokud ale vznikne potřeba zapojit do modelu signály, které jsou generovány externím zařízením, nebo naopak signály ze simulace odeslat do reálného zařízení, je potřeba použít další zařízení. Pro tyto účely existuje několik možností jako například rozšiřující desky pro PC (HUMUSOFT, 2021), nebo softwarové balíčky určené přímo pro Simulink, například Simulink Support Package for Arduino Hardware (SIMULINK TEAM, 2023) nebo další možnosti. Všechny tyto možnosti jsou určitě dobrou volbou pro rozšíření možností Simulinku, ale jejich funkcionalita je pevně daná a modifikace je ve většině případů nemožná.

V současné době je připojení externích zařízení ve školní Laboratoři aplikované informatiky k programu Matlab/Simulink řešeno pomocí měřicí karty MF 624 (HUMUSOFT, 2021), která je popsána v následující kapitole. Používaná verze programu Matlab R2010b je dnes sice zastaralá, ale pro školní potřeby dostačující v kontrastu s finanční náročností

na aktualizaci tohoto softwaru. Pro použití karty MF 624 je však nutné rozhraní PCI (PCIe), které nemusí být vždy dostupné u novějších PC. I když je toto uspořádání pro potřeby školy dostačující, vznikla myšlenka vytvořit univerzálnější rozhraní, které by bylo možné využívat na novějších verzích operačního systému Windows a především umožňovalo připojení k PC pomocí dnes běžně používaného rozhraní USB.

**Cílem** této práce je vytvořit rozhraní pro připojení externích zařízení k programu Simulink založené na platformě Arduino, konkrétně Arduino Mega 2560 Rev3. Toto rozhraní umožní začlenit externí analogové a digitální signály do simulinkového modelu a využít je při simulaci. Dále bude rozhraní schopno odesílat analogové, digitální a pulzně šířkově modulované signály vygenerované během simulace do reálných externích zařízení. Komunikace mezi tímto rozhraním a počítačem bude realizována pomocí sběrnice USB. Program pro řídicí mikroprocesor bude napsán v jazyce C s využití knihoven Arduina. Knihovna pro PC bude vytvořena pomocí standardních bloků Simulinku a funkcí Matlabu.

Struktura práce, která je napsána v  $\text{\LaTeX} 2_{\epsilon}$ <sup>1</sup> (SCHENK, C., 2009), je rozvržena do čtyř logických celků. V kapitole 2 jsou popsány některé stávající možnosti komunikace programu Matlab/Simulink s externími zařízeními. V kapitole 3 je návrh vlastního rozhraní pro komunikaci Simulinku s externími zařízeními. Kapitola je zaměřena na praktickou část návrhu pro hardwarové a softwarové vybavení vlastního rozhraní. V kapitole 4 jsou popsány poznatky získané při oživování a testování rozhraní. V příloze A je uveden obsah příloženého DVD, v příloze B je seznam použitého softwaru, v příloze C je časový plán absolventské práce, v příloze D je finanční rozpočet projektu, v příloze E jsou schemata zapojení elektroniky hardwarové části a v příloze F je uveden seznam součástí použitých při výrobě jednotlivých desek plošných spojů.

---

<sup>1</sup> $\text{\LaTeX} 2_{\epsilon}$  je rozšíření systému  $\text{\LaTeX}$ , což je kolekce maker pro  $\text{\TeX}$ .  $\text{\TeX}$  je ochranná známka American Mathematical Society.



# Kapitola 2

## Možnosti připojení externích zařízení k Matlabu/Simulinku

V této kapitole bude popsáno několik možností připojení externích zařízení k programu Matlab/Simulink. Nejedná se zdaleka o úplný výpis všech možností ale, dle názoru autora práce, relevantních možností připojení. Jedná se kartu MF 624 od společnosti Humusoft (HUMUSOFT, 2021) a knihovnu pro Simulink, která umožňuje komunikaci s Arduinem.

### 2.1 Měřicí karta MF 624

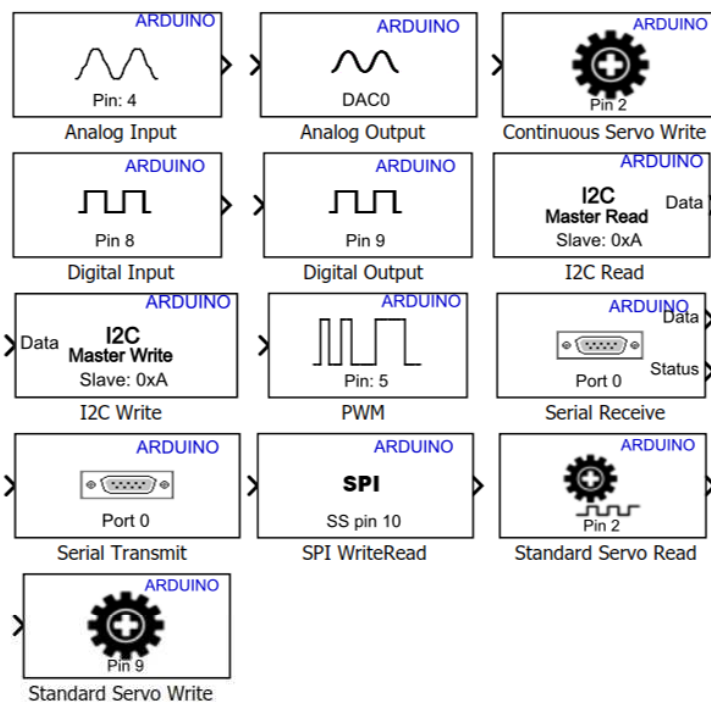
Vstupně-výstupní karta MF 624, které je na obr. 2.1, se připojuje do PC pomocí interní sběrnice PCI a výstupem této karty jsou dva 37-pinové konektory. Jedná se o 32-bitovou architekturu, což ji předurčuje pro velmi dobré přenosové rychlosti. Karta MF 624 má tyto základní vstupy a výstupy:

- 8 single-ended 14-bitových analogových vstupů
- 8 14-bitových analogových výstupů
- 8 digitálních vstupů, 8 digitálních výstupů
- 4 vstupy inkrementálních snímačů
- 4 čítače/časovače



## 2.2 Knihovna Support Package for Arduino

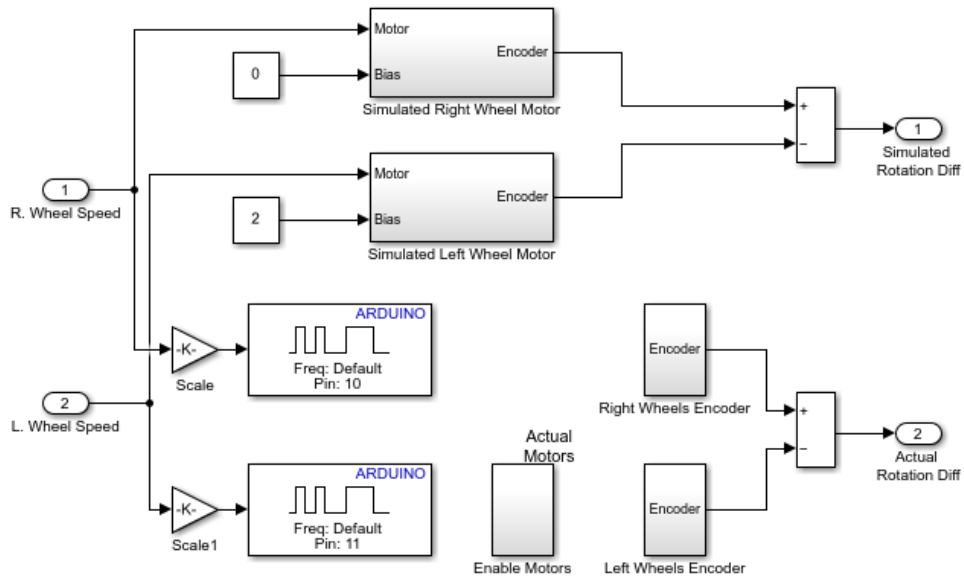
Další možností, jak komunikovat mezi Simulinkem a externím zařízením, je nainstalování toolboxů nebo Add-ons souborů. Toolbox je knihovna komponent, které umožňují některé specifické funkce simulace. Pro podporu hardwaru a komunikaci s ním existuje několik toolboxů, jak pro implementaci konkrétního hardwaru, tak toolboxy, které umožní pouze navázat komunikaci se sériovou linkou např. `Instrument Control Toolbox`, na kterou lze připojit jakékoliv zařízení, které podporuje komunikaci po této lince. Jedna z nejzajímavějších možností je použití balíčku `Package for Arduino Hardware`. Pomocí tohoto balíčku je možno vkládat do simulinkového modelu bločky znázorněné na obr. 2.3, které komunikují v reálné čase s připojenou deskou Arduino přes rozhraní USB.



Obrázek 2.3: Ukázka simulinkových bločků Support Package for Arduino – převzato z (SIMULINK TEAM, 2023)

Práce s touto knihovnou je velmi jednoduchá a intuitivní. Uživatel pouze vloží do obvodu bloček s požadovanou funkcí, nastaví, na kterém pinu Arduina je připojeno externí zařízení, dále připojí do USB portu Arduino desku a spustí simulaci. V některých případech je nutno nastavit typ Arduino desky a číslo COM portu, pokud ho systém nerozpozná automaticky. Jedná se tedy o velmi jednoduchou a levnou komunikaci Simulinku s okolím. Bohužel pro případ použití ve školních učebnách tato možnost naráží

na zásadní problém, a to je kompatibilita různých verzí Matlabu. Školní verze Matlabu R2010b nepodporuje tento toolbox, který je podporován až od verze R2014a. Na obr. 2.4 je ukázka použití této knihovny.

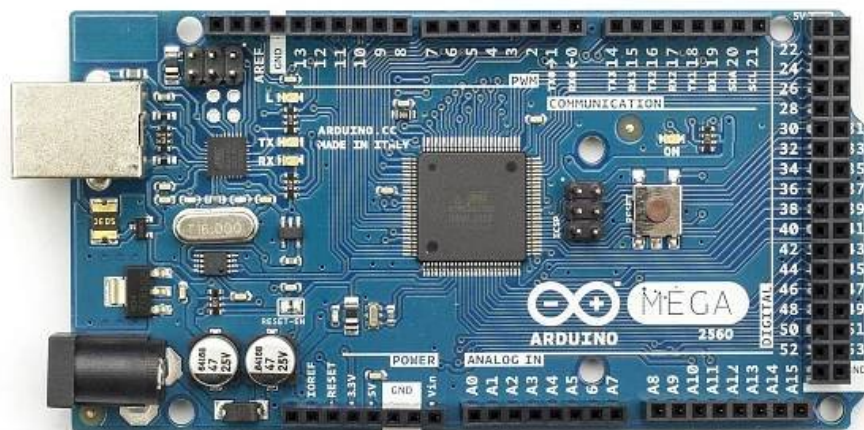


Obrázek 2.4: Ukázka použití simulinkové knihovny Support Package for Arduino – převzato z (SIMULINK TEAM, 2023)

# Kapitola 3

## Realizace komunikačního rozhraní

Pro návrh vlastního komunikačního rozhraní pro spojení Simulinku s externími zařízeními sloužila jako inspirace předcházející kapitola. Pro připojení externího zařízení byl zvolen USB port, což je v dnešní době standardní a mnohdy jediná možnost, jak připojit externí zařízení k PC. Pro navrhované rozhraní byl zvolen mikropočítač Arduino Mega, viz obr. 3.1. Jedná se o levné zařízení, které obsahuje hlavně USB rozhraní, což velice zjednodušuje stavbu celého navrhovaného rozhraní. Mikroprocesor ATmega2560, který je hlavní součástí desky Arduino Mega, integruje různé periferie, které lze velmi jednoduše použít.

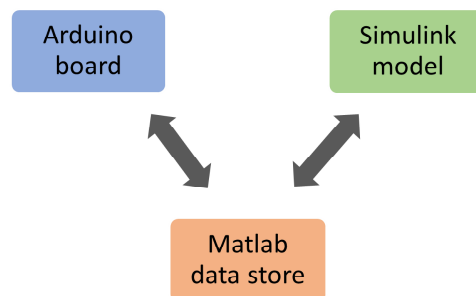


Obrázek 3.1: Arduino Mega – převzato z (BOL, 2023)

Deska Arduino však musí být rozšířena o další analogové obvody, zejména obvody pro úpravu analogového signálu v rozsahu  $\pm 10\text{ V}$ , protože Arduino deska pracuje v rozsahu

0 V až 5 V. Snahou bylo vytvořit rozhraní, které se bude ohledně vstupů a výstupů co nejvíce podobat měřicí kartě MF 624, z důvodu kompatibility již hotových a oživených modelů ve školní laboratoři. Některé funkce vytvářeného rozhraní byly oproti desce MF 624 změněny, nicméně základní koncept hardwaru vychází právě z této desky.

Pro komunikaci Simulinku s externím zařízením pomocí USB portu bude nutné v PC spustit program, který tuto komunikaci umožní. Simulink komunikaci s hardwarem počítače neumožňuje, ale je součástí programu Matlab, který disponuje funkcemi pro komunikaci se sériovým portem. Zjednodušený princip toku dat mezi simulinkovým modelem a Arduinem je znázorněn na obr. 3.2.



Obrázek 3.2: Princip toku dat mezi simulinkovým modelem a Arduinem

Jak obr. 3.2 naznačuje, způsob načítání dat je jednoduchý. Před spuštěním simulace jsou v Matlabu vytvořeny proměnné, respektive struktura dat, které jsou v reálném čase aktualizovány z připojené desky Arduinu a následně načítány programem Simulink při každé aktualizaci simulinkového modelu. Podrobnější popis komunikace je popsán v další části této práce.

Při návrhu tohoto rozhraní byl kladen důraz především na jednoduchost, možnost rychlého přeprogramování a univerzálnost. Z těchto důvodů bylo použito „pouze“ Arduino. Při komerčním vývoji, nebo při vývoji rozhraní založeném na rychlosti, nebo objemu přenesených dat, by bylo vhodné zvolit výkonnější alternativu jako například použití hradlových polí FPGA nebo výkonnějších procesorů ARM, SAM, či jiných. Programování těchto součástek však není tolik intuitivní jako programování osmibitových mikroprocesorů. Vyžaduje pokročilejší vývojové nástroje a v neposlední řadě je cena těchto součástek násobně vyšší.

Celé zařízení, které tvoří rozhraní pro komunikaci, lze rozdělit na dvě základní části a to na část hardwarovou a softwarovou. Základem hardwarové části je deska Arduino

Mega, což je levná vývojová deska s mikroprocesorem AVR a možností komunikace pomocí USB rozhraní. Softwarová část zahrnuje program pro mikroprocesor Arduina a program pro Matlab/Simulink, které budou popsány v další části této kapitoly. Základními vlastnostmi navrhovaného rozhraní jsou:

**Analogové vstupy:**

- vstupy AD0 až AD7
- rozlišení: 12 bitů
- počet kanálů: 8
- rychlost vzorkování: 100 ksps
- rychlost komunikace: 2 MHz
- vstupní napětí:  $\pm 10$  V
- vstupní ochrana:  $\pm 12$  V
- přesnost:  $\pm 1$  LSB

**Analogové výstupy:**

- výstupy DA0 až DA7
- rozlišení: 12 bitů
- počet kanálů: 8
- výstupní proud: max. 20 mA
- napěťový rozsah:  $\pm 10$  V
- přesnost:  $\pm 1$  LSB

**Digitální vstupy:**

- vstupy DIN0 až DIN15
- počet vstupů: 16
- napěťová úroveň: TTL

**Digitální výstupy:**

- výstupy DOUT0 až DOUT15
- počet výstupů: 16

- napěťová úroveň: TTL
- výstupní proud: max. 40 mA/výstup, 200 mA celkem

#### Kvadraturní enkodér:

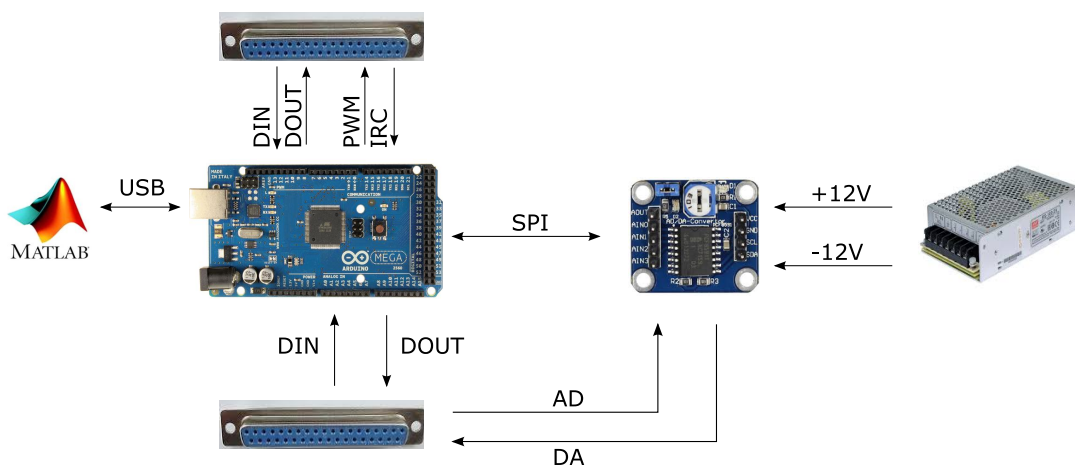
- vstupy ICR0A+, ICR0B+, ICR0I+
- počet kanálů: 1
- napěťová úroveň: TTL

#### Pulzně šířková modulace – PWM:

- výstupy PWM0 až PWM7
- výstupní frekvence: 250 Hz – 32 kHz pro výstupy PWM0 až PWM4, 62,5kHz PWM5, 30 Hz pro PWM6 a PWM7
- rozlišení: 16-bitů PWM0 až PWM4, 8-bitů PWM5 až PWM7

### 3.1 Hardwarová část navrhovaného rozhraní

Jak již bylo řečeno, základ hardwarové konfigurace je deska Arduino Mega, viz obr. 3.3, která obsahuje jednočipový mikroprocesor AVR ATmega2560 (MICROCHIP, 2014), na kterou jsou připojeny další podpůrné obvody. Blokové schéma celého rozhraní je na obr. 3.3.

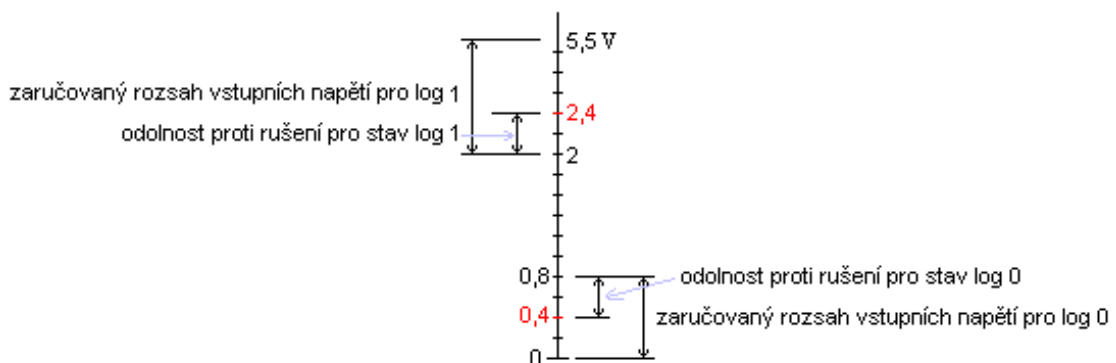


Obrázek 3.3: Blokové schéma hardwarové části navrhovaného rozhraní



### 3.1.1 Digitální vstupy a výstupy navrhovaného rozhraní

Rozšiřující deska pro Matlab/Simulink obsahuje 16 digitálních vstupů DIN0 – DIN15 a 16 digitálních výstupů DOU0 – DOU15. Tyto vstupy a výstupy jsou napojeny přímo na desku Arduina a jsou kompatibilní s TTL napěťovými úrovněmi. Maximální napětí na těchto vstupech je proto +5 V. Při překročení tohoto napětí dojde s velkou pravděpodobností ke zničení mikroprocesoru a je proto nutné zvýšit pozornost při připojování signálů. Vhodné napěťové úrovně jsou zobrazeny na obr. 3.4.

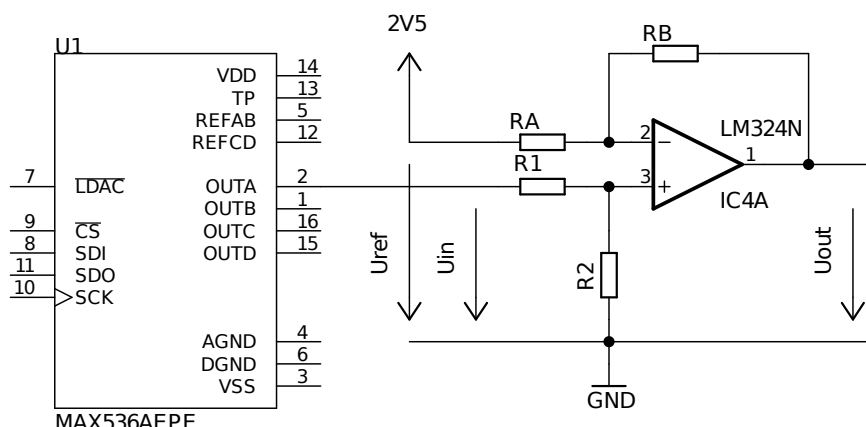


Obrázek 3.4: Napěťové úrovně pro TTL logiku

### 3.1.2 Analogové vstupy a výstupy

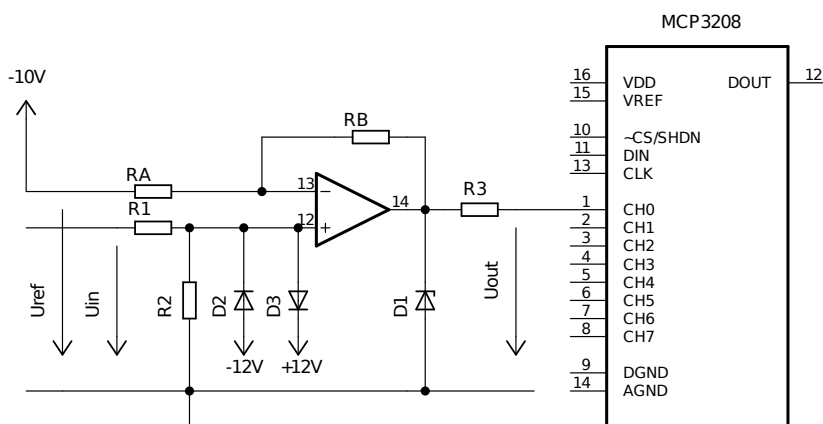
Na analogové vstupy AIN0 – AIN7 je možné připojit vstupní napětí v rozsahu  $\pm 10$  V, taktéž analogové výstupy poskytují napěťové úrovně v rozsahu  $\pm 10$  V. Pro převod digitální hodnoty na hodnotu analogovou je použit 12-bitový digitálně analogový převodník MAX536 (MAXIM, 2011) od firmy Maxim a pro převod analogové veličiny 12-bitový analogově digitální převodník MCP3208 (MICROCHIP, 2008). Tyto integrované obvody byly zvoleny především s ohledem na jejich dostupnost a cenu. Zejména u DA obvodů je v současné době (léto 2022) dostupnost značně omezena.

Oba integrované obvody komunikují s Arduinem pomocí sběrnice SPI a poskytují analogové napětí v rozmezí 0 V až +5 V. Je tedy nutné použít převodník, který převede napětí 0 V až +5 V na požadované  $-10$  V až +10 V. K tomu slouží operační zesilovač LM324 (TEXAS INSTRUMENTS, 2015), zapojený jako rozdílový zesilovač. Zapojení všech osmi vstupů, respektive výstupů, je shodné a je proto zobrazeno pouze zapojení pro jeden výstup na obr. 3.5, respektive jeden vstup na obr. 3.6.



Obrázek 3.5: Zapojení analogového výstupu navrhovaného rozhraní

Jedná se o základní zapojení, kde zesílení zesilovače je dáno rovnicí (3.1) pro převodník 0 V až +5 V na  $-10$  V až +10 V a rovnicí (3.2) pro převod rozsahu  $-10$  V až +10 V na rozsah 0 V až +5 V. Rozdíl v obou rovnicích je pouze ve znaménku a je dán použitím záporného referenčního napětí na invertujícím vstupu operačního zesilovače. Tyto rovnice platí pouze pokud  $R_A = R_B$  a  $R_1 = R_2$ .



Obrázek 3.6: Zapojení analogového vstupu navrhovaného rozhraní

$$U_{\text{out}} = (U_{\text{ref}} - U_{\text{in}}) \cdot \frac{R_B}{R_A} \quad (3.1)$$

$$U_{\text{out}} = (U_{\text{ref}} + U_{\text{in}}) \cdot \frac{R_B}{R_A} \quad (3.2)$$

Velikost referenčního napětí, potažmo zesílení, bylo určeno s ohledem na elektronické řady součástek. Bylo zvoleno zesílení 4 (1/4) a referenční napětí 2,5 V ( $-10$  V), čímž je

Tabulka 3.1: Teoretická převodní tabulka AD/DA

Digitální hodnota	Analogová hodnota
0	-10 V
1	-9,995 V
2048	0 V
2049	0,005 V
4095	+10 V

možné zvolit hodnoty rezistorů z běžné řady E24 a to 30 k $\Omega$  a 120 k $\Omega$ . Referenční napětí +2,5 V a -10 V je získáno pomocí napěťové reference TL431 (TEXAS INSTRUMENTS, 2022). Obvod analogového vstupu též obsahuje ochrany proti vyššímu napětí, Zenerova dioda D<sub>1</sub> chrání obvod MCP3208 proti napětí vyššímu než +5,1 V a Schottkyho diody D<sub>2</sub> a D<sub>3</sub> chrání operační zesilovač proti napětí mimo rozsah  $\pm 10$  V. Proudové zatížení analogových výstupů je dáno obvodem LM324 a je 20 mA pro každý výstup. Součet všech proudů by neměl překročit 85 mA, jinak hrozí zničení výstupních obvodů rozhraní.

Vstupem respektive výstupem analogově digitálního převodníku je dekadická hodnota daná rozlišením použitých obvodů, tedy 12 bitů (0 až 4095), viz tabulka 3.1. Je nutné zdůraznit, že se jedná o hodnoty teoretické (vypočítané) a skutečnou hodnotu je třeba ověřit na referenčních hodnotách. Velikost poskytnutých analogových hodnot je silně závislá na přesnosti použitých součástek, teplotě součástek a frekvenci analogového signálu.

### 3.1.3 Kvadraturní enkodér

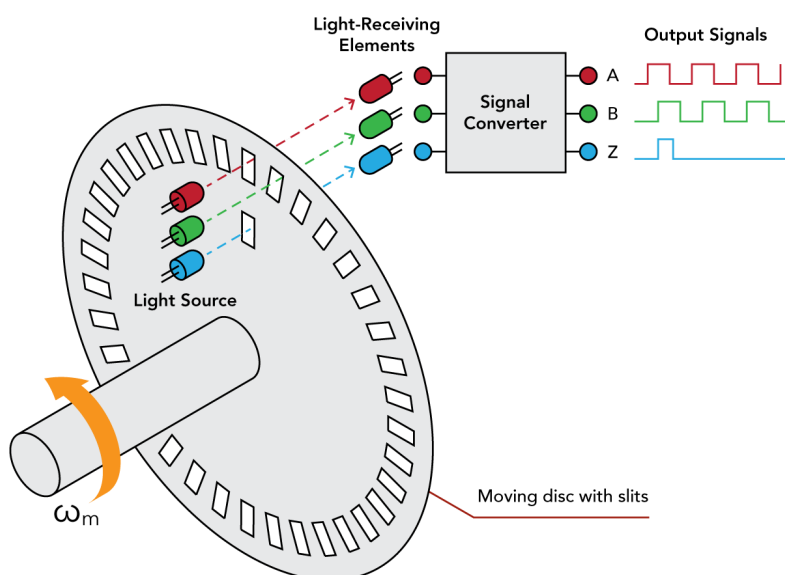
Na konektor X2 je možné připojit kvadraturní enkodér, někdy též nazývaný inkrementální enkodér, viz obr. 3.7. Jedná se zpravidla o rotační zařízení, které poskytuje informace o natočení a směru otočení hřídelky enkodéru.

Výstupem kvadraturního enkodéru jsou zpravidla tři signály. Dva signály ICR0A+ (pin 1) a ICR0B+ (pin 3), které určují pootočení hřídelky enkodéru a jejich posunutí, které určuje směr otáčení. Principiální schéma je na obr. 3.8. Třetí signál, nazývaný též nulový ICR0I+ (pin 5), umožňuje stanovení polohy hřídelky enkodéru a je většinou generován jednou za otáčku hřídelky. Tento nulový signál je v našem rozhraní vyveden na výstupní konektor X2, ale není programově ošetřen a pro jeho využití je potřeba provést softwarové úpravy.



Obrázek 3.7: Ukázka kvadrurního enkodéru – převzato z (DEX, 2023)

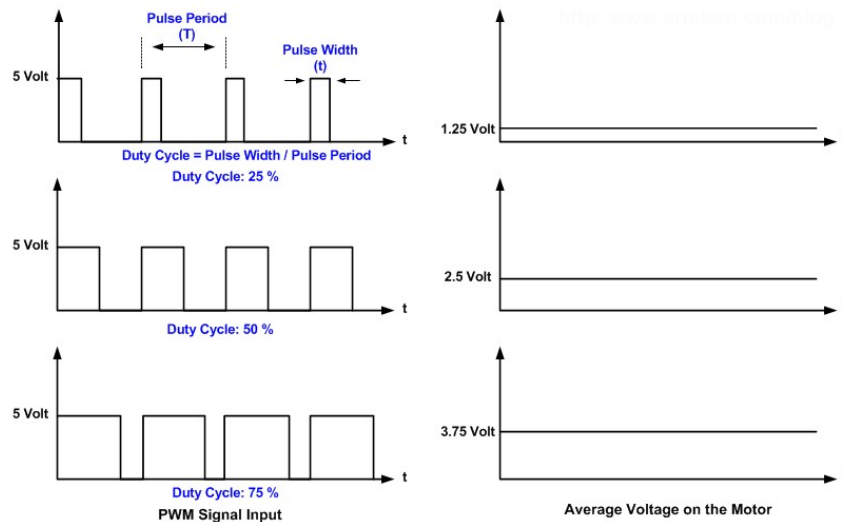
Na konektor X2 (piny 2, 4, 6) jsou též předpřipraveny signály pro připojení diferenciálních vstupů ICR0A–, ICR0B–, ICR0I–. Diferenciální vstupy se používají pro eliminaci výskytu chyb způsobené rušením. Toto rušení v podmínkách laboratoře školy není nijak kritické, a proto bylo od možnosti využití diferenciálních vstupů upuštěno a nejsou programově ošetřeny. Softwarový blok pro rotační enkodér tedy zpracovává pouze signály A+ a B+ a podle jejich stavů inkrementuje nebo dekrementuje proměnnou typu integer v rozsahu  $-32768$  až  $+32767$ .



Obrázek 3.8: Princip vytváření signálů kvadrurního enkodéru – převzato z (FUTEK, 2023)

### 3.1.4 Pulzně šířková modulace

Pulzně šířková modulace (PWM) slouží převážně pro regulaci výkonu. Jedná se v podstatě o rychlé zapínání a vypínání výstupu a tím změnu doby mezi zapnutím a vypnutím výstupu. Tím docílíme toho, že střední hodnota proudu či napětí výstupu, tedy i výkon, odpovídá střídě signálu PWM, viz obr. 3.9.



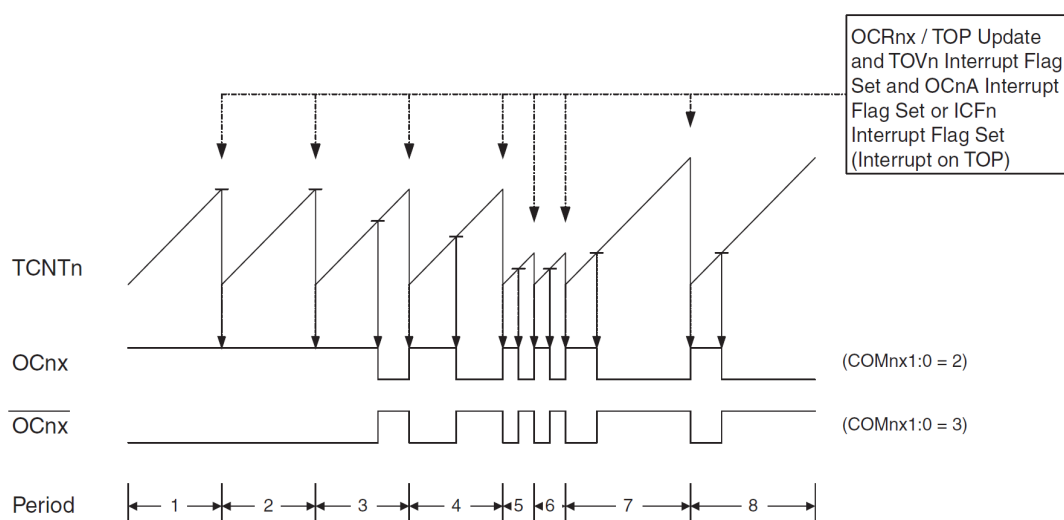
Obrázek 3.9: Příklad PWM signálu a vyfiltrovaného výstupního napětí – převzato z (ERMICRO, 2009)

Obvod ATmega2560 disponuje několika výstupními kanály pro generování PWM signálu. Tyto výstupy jsou řízeny časovači Timer0 až Timer4. Časovače mikroprocesoru Arduina řídí dva až tři výstupy PWM a proto lze u těchto společných výstupů nastavit jednotlivě pouze střidu výstupního PWM signálu, ale frekvence signálu musí být společná. Toto uspořádání trochu komplikuje nastavení jednotlivých výstupů. Pro maximální využití možností mikroprocesoru bylo použito níže popsání řešení.

- Výstup PWM0 je řízen časovačem Timer1 a je možno nastavit frekvenci výstupního signálu v rozsahu 250 Hz až 32 000 Hz nezávisle na ostatních výstupech. Taktěž střidu je možno nastavit nezávisle v rozsahu 0 % až 100 %.
- Výstupy PWM1, PWM2 a PWM3 používají Timer4. Z toho plyne, že frekvence všech těchto výstupů (pokud jsou v simulinkovém modelu použity) musí být nastavena na stejnou hodnotu a to v rozsahu 250 Hz až 32 000 Hz. Pokud ne, bude docházet k neustálé změně výstupní frekvence a tím k neočekávanému chování

těchto výstupů. Střídu lze nastavit nezávisle na ostatních výstupech v rozsahu 0 % až 100 %.

- Výstup PWM4 využívá pro svoji činnost Timer3 a pracuje nezávisle na ostatních výstupech. Platí pro něj stejná omezení jako v případě výstupu PWM0.
- Výstup PWM5 používá osmibitový časovač Timer0. Možnosti pro řízení frekvence u osmibitových časovačů jsou menší než u šestnáctibitových časovačů a proto byla zvolena fixní výstupní frekvence 62 500 Hz s možností změny střídy v rozsahu 0 % až 100 %. Nastavení frekvence v simulinkovém bločku nebude mít žádný efekt na výslednou frekvenci generovaného PWM signálu.
- Poslední výstupy PWM6 a PWM7 jsou řízeny časovačem Timer2, což je osmibitový časovač. Platí pro ně stejné podmínky jako pro výstup PWM5 s tím, že výstupní frekvence je pevně nastavena na 30 Hz.



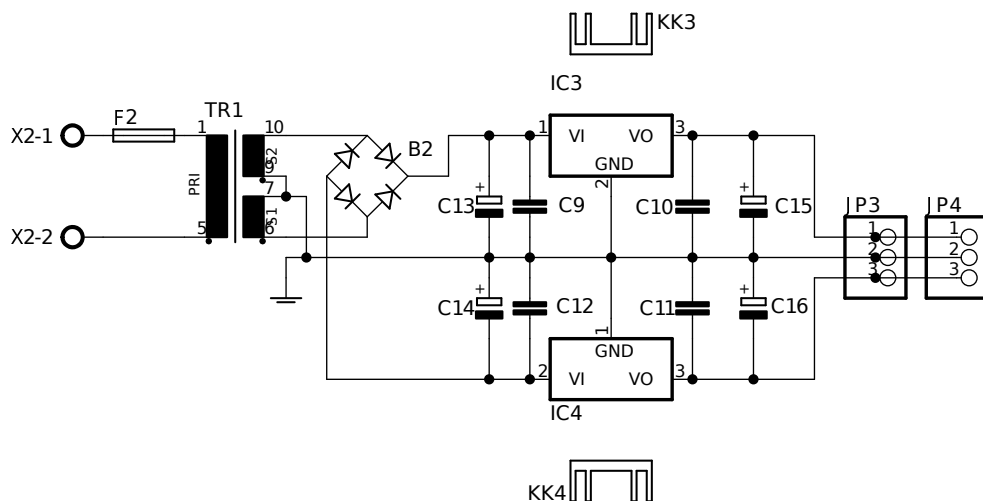
Obrázek 3.10: Časový diagram Fast PWM módu – převzato z (MICROCHIP, 2014)

Všechny PWM výstupy pracují v tzv. Fast PWM módu, viz obr. 3.10. V tomto módu čítač čítá do svého maxima a během čítání neustále porovnává hodnotu registru TCNTn s hodnotou komparačního registru OCRn. Pokud dojde ke shodě registrů TCNTn a OCRn, výstup OCnx mikroprocesoru je vynulován. Hodnota registru OCRn tedy určuje dobu trvání logické jedničky na výstupu OCnx a tím střídu generovaného

PWM signálu. Maximum (vrchol) čítače určuje do jaké hodnoty bude čítač čítat, respektive jak dlouho. Tím je stanovena doba trvání jedné periody PWM signálu, tedy frekvence. Vrchol čítače lze nastavit buď na pevnou hodnotu, nebo na hodnotu danou registrem ICRn. Toto je právě oním omezením, kdy výstupy, které používají stejný časovač, musí mít stejnou frekvenci, protože každý časovač má pouze jednu hodnotu maxima. V případě výstupů PWM1 až PWM4 je použita možnost nastavení vrcholu pomocí registru ICRn a tím je možno nastavovat frekvenci v daném rozsahu. V případě PWM5 až PWM7 je hodnota vrcholu nastavena pevně na maximum (0xFF).

### 3.1.5 Napájecí zdroj pro navrhované rozhraní

Celé zařízení je napájeno síťovým napětím 230 V. Toto napětí je dále transformováno na napětí  $2 \times 15$  V, usměrněno, vyfiltrováno a stabilizováno integrovanými stabilizátory 7812 a 7912. Symetrické napětí je potřebné pro funkci analogově-digitálních a digitálně-analogových převodníků. Proto je zde použit transformátor s dvojitým vinutím. Výstupní napětí  $\pm 12$  V je též použito jako vstupní napětí pro napěťové reference TL431, pomocí kterých je získáváno referenční napětí  $-10$  V a  $+2,5$  V. Napětí  $\pm 12$  V je též vyvedeno na výstupní konektor X1.



Obrázek 3.11: Schéma zapojení napájecího zdroje

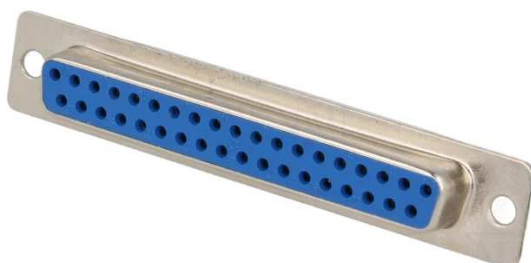
Deska Arduino a tedy i digitální výstupy a výstupy PWM jsou napájeny napětím  $+5$  V přímo ze sběrnice USB. Pro správnou funkci celého rozhraní je proto nutné propojit nulové potenciály obou zdrojů, což je řešeno interně přímo na desce Arduino.

### 3.1.6 Zapojení konektorů X1 a X2

Výstupy rozhraní jsou fyzicky vyvedeny na dva konektory typu D-Sub 37, viz obr. 3.12. Číslování jednotlivých pinů konektoru je na obr. 3.13. Konektor X1 je zapojen stejně jako konektor desky MF 624, viz tabulka 3.2. Zapojení druhého konektoru X2 bylo změněno. Původní konektor umožňoval připojit čtyři kvadrurní enkodéry, což bylo vyhodnoceno jako zbytečně mnoho. Dále obsahoval vstupy a výstupy čítače/časovače. Tyto vývody byly zrušeny, protože se autor práce domnívá, že funkce čítání a časování lze mnohem jednodušeji implementovat přímo v Simulinku a není potřeba další zařízení. Zrušené vývody byly nahrazeny digitálními vstupy, digitálními výstupy a výstupy pulzně šířkové modulační. Vše je shrnuto v níže uvedené tabulce 3.2 a v tabulce 3.3.

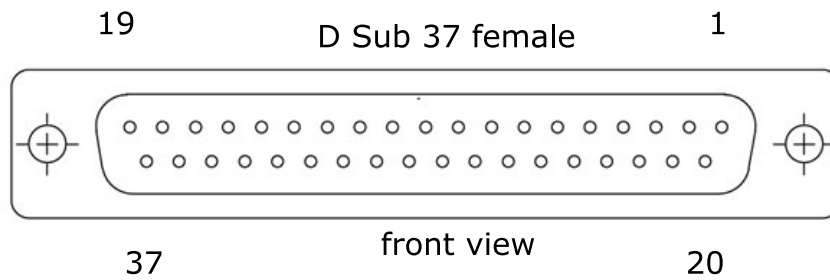
V tabulkách je uvedeno kromě funkcionality jednotlivých pinů také číslo vývodu Arduina a název fyzického vývodu mikroprocesoru ATmega2560. Je to zde uvedeno pro lepší orientaci při programování Arduina, ale hlavně z důvodu, že většina pinů AVR mikroprocesorů má alternativní funkce. Pokud tedy funkcionality určitého vývodu nebude vyhovující, lze ji změnit malou úpravou v kódu programu Arduina. Tuto změnu lze ovšem provést jen u vývodů, na které nejsou napojeny další elektronické součástky, tedy u digitálních vstupů a výstupů a výstupů PWM signálů. Princip naprogramování jednotlivých vývodů bude popsán v následujících kapitolách.

Jak je patrné z tabulky 3.3, některé vývody nemají přiřazenu žádnou funkcionality. Jsou však fyzicky napojeny na piny mikroprocesoru ATmega2560. Tyto piny byly ponechány pro případné další využití, například vývody PH0 a PH1 lze nakonfigurovat jako vstup a výstup (Rx, Tx) sériové komunikace, a tím rozšířit možnosti desky o možnost komunikovat s dalšími zařízeními.



Obrázek 3.12: Konektor D-Sub 37 female





Obrázek 3.13: Konektor D-Sub 37 female – číslování vývodů

Tabulka 3.2: Zapojení vývodů konektoru X1

Pin X1	Funkce	Arduino pin	AVR pin	Pin	Funkce	Arduino pin	AVR pin
1	AD0	50,51,52,53	SPIa	20	DA0	12,13,51,52	SPIb
2	AD1	50,51,52,53	SPIa	21	DA1	12,13,51,52	SPIb
3	AD2	50,51,52,53	SPIa	22	DA2	12,13,51,52	SPIb
4	AD3	50,51,52,53	SPIa	23	DA3	12,13,51,52	SPIb
5	AD4	50,51,52,53	SPIa	24	DA4	12,13,51,52	SPIb
6	AD5	50,51,52,53	SPIa	25	DA5	12,13,51,52	SPIb
7	AD6	50,51,52,53	SPIa	26	-12 V	-	-
8	AD7	50,51,52,53	SPIa	27	+12 V	-	-
9	AGND	-	-	28	+5 V	-	-
10	DA6	12,13,51,52	SPIb	29	GND	-	-
11	DA7	12,13,51,52	SPIb	30	DOUT0	62	PK0
12	DIN0	54	PF0	31	DOUT1	63	PK1
13	DIN1	55	PF1	32	DOUT2	64	PK2
14	DIN2	56	PF2	33	DOUT3	65	PK3
15	DIN3	57	PF3	34	DOUT4	66	PK4
16	DIN4	58	PF4	35	DOUT5	67	PK5
17	DIN5	59	PF5	36	DOUT6	68	PK6
18	DIN6	60	PF6	37	DOUT7	69	PK7
19	DIN7	61	PF7				

Tabulka 3.3: Zapojení vývodů konektoru X2

Pin X2	Funkce	Arduino pin	AVR pin	Pin	Funkce	Arduino pin	AVR pin
1	ICR0A+	2	PE4	20	PWM0	11	PB5
2	–	–	–	21	PWM6	10	PB4
3	ICR0B+	3	PE5	22	PWM7	9	PH6
4	–	–	–	23	PWM1	8	PH5
5	ICR0I+	18	PD3	24	PWM2	7	PH4
6	–	–	–	25	PWM3	6	PH3
7	DIN8	35	PC2	26	PWM4	5	PE3
8	DIN9	34	PC3	27	PWM5	4	PG5
9	DIN12	31	PC6	28	+5V	–	–
10	DIN13	30	PC7	29	GND	–	–
11	DOUT8	22	PA0	30	DIN14	14	PJ1
12	DOUT9	23	PA1	31	DIN15	15	PJ0
13	DOUT10	24	PA2	32	–	16	PH1
14	DOUT11	25	PA3	33	–	17	PH0
15	DOUT12	26	PA4	34	+5V	–	–
16	DOUT13	27	PA5	35	–	19	PD2
17	DOUT14	28	PA6	36	DIN11	32	PC5
18	DOUT15	29	PA7	37	DIN10	33	PC4
19	GND	–	–				

### 3.2 Softwarová část navrhovaného rozhraní

V této části práce bude popsán software, který je nutný pro úspěšnou komunikaci programu Simulink s externími zařízeními. Jak je z hardwarového řešení zřejmé musí být vytvořeny programy dva, program pro Arduino a program, který poběží v prostředí Matlabu. V neposlední řadě je nutné navrhnout v Simulinku bloky, které budou schopny data z Matlabu číst/zapisovat a dále je implementovat do simulinkového modelu.

### 3.2.1 Software pro Matlab

Při prvotním návrhu rozhraní se jako nejvíce problematické ukázalo, jakým způsobem inicializovat proměnné pro ukládání příchozích dat a jakým způsobem tato data do proměnných ukládat. Nakonec byla pro ukládání dat zvolena datová struktura mapa. Jedná se v podstatě o asociativní pole, kde jednotlivé uložené hodnoty lze indexovat pomocí unikátního klíče. Jako hodnota klíče bylo zvoleno třímístné celé číslo, kde řád stovek určuje typ signálu a řád desítek a jednotek nesou informaci o požadovaném vstupu. Například hodnota klíče 412 bude představovat digitální vstup DIN12. Rozsahy klíčů pro různé signály jsou v následujícím výpisu.

- Inicializace: 100
- Analogové vstupy: 200 – 299
- Vstup inkrementálního enkodéru: 300
- Digitální vstupy: 400 – 499
- Analogové výstupy: 500 – 599
- PWM výstupy: 600 – 699
- Digitální výstupy: 700 – 799

Celý program pro Matlab se skládá ze dvou skriptů a dvou funkcí. Skript `OpenPort` slouží k otevření sériového portu a vytvoření proměnných pro ukládání dat. Dále tento skript spustí časovač, který periodicky volá funkci `ReadWritePort`, která komunikuje s deskou Arduino. Skript `OpenPort` je volán při startu simulace, tedy pouze jednou, pomocí Callback `InitFcn`. Skript `ClosePort`, jak již název napovídá, uzavírá port, zastavuje časovač a ruší proměnné. Výpis těchto skriptů je uveden níže.

```
%OpenPort
port_num = get_param(gcs, 'com');
port_name = strcat('COM', port_num);
port = serial(port_name, 'BaudRate', 115200);
fopen(port);
fprintf(port, '100,1');
ardu_cop_data = containers.Map('KeyType', 'double', 'ValueType', 'double');
t = timer;
t.Period = 0.3;
```

```
t.StartDelay = 1;
set(t,'ExecutionMode', 'fixedRate');
set(t,'TimerFcn', 'ReadWritePort');
```

```
%ClosePort
stop(t);
delete(t);
fprintf(port, '100,0');
fclose(port);
clear port;
clear t;
clear ardu_cop_data;
clear port_num;
clear port_name;
```

Funkce `ReadWritePort`, jejíž výpis je uveden níže, zajišťuje veškerou komunikaci pomocí již dříve otevřeného portu. Tato funkce prochází v cyklu datovou strukturu a podle uložených klíčů čte/zapisuje hodnoty z/do Arduina. Komunikační protokol musí být v tomto formátu: „KLÍČ,HODNOTA\n“. Znak nového řádku je zde důležitý, protože určuje konec přenášeného slova.

```
function ReadWritePort()
    acd = evalin('base','ardu_cop_data');
    com = evalin('base','port');

    for k = keys(acd)
        key = k{1};
        if key >= 500
            str = sprintf('%u,%u',key ,acd(key));
            fprintf(com,str);
        elseif key >= 200
            str = sprintf('%u,1',key);
            fprintf(com,str);
            pause(0.01);
            [tline,count] = fscanf(com,'%c');
            if count > 0
                [index,rem] = strtok(tline,',');
                if index ~= 'E'
                    value = strtok(rem,'\n');
                    acd(str2double(index)) = str2double(value);
                end
            end
        end
    end
```

```

        end
    end
end
assignin('base','ardu_cop_data',acd);
end

```

Poslední funkcí, která je použita v programu Matlab, je funkce `SetOrGetValue`. Její výpis je uveden v následujícím textu a její činností je pouze ukládat data do proměnné `ardu_cop_data`, nebo naopak vracet hodnotu podle obdržného klíče. Tato funkce je volána ze Simulinku pomocí komponenty MATLAB Fcn.

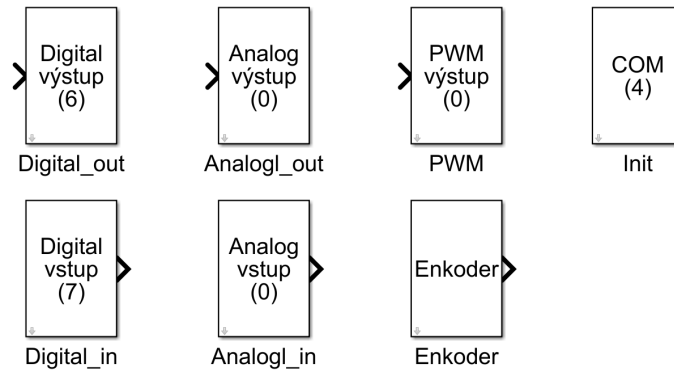
```

function [y] = SetOrGetValue(val)
    %val(1) = type
    %val(2) = pin
    %val(3) = value
    acd = evalin('base','ardu_cop_data');
    key = val(1) + val(2);
    if ~(isKey(acd,key))
        acd(key) = 0;
        assignin('base','ardu_cop_data',acd);
    end
    if key >= 500
        acd(key) = floor(val(3));
        assignin('base','ardu_cop_data',acd);
    end
    y = acd(key);
end

```

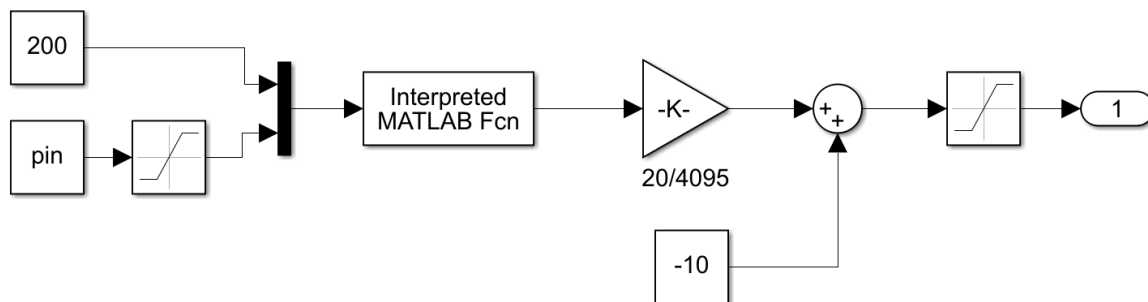
### 3.2.2 Bloky pro Simulink

Pro integraci signálů do simulinkového modelu byly navrženy bločky, viz obr. 3.14. Jejich funkce je intuitivní a vychází z požadavků, které byly stanoveny pro vytvářené komunikační rozhraní. Uživatel pouze volí číslo požadovaného výstupu. Tento výstup musí být samozřejmě dostupný fyzicky na konektoru X1 nebo X2. Jediný blok, který nezpracovává přímo signály je blok `Init`. Tento blok slouží pro zadání čísla sériového COM portu, na který je deska Arduino připojena. Tento blok musí být použit vždy, při použití některých ostatních bloček.



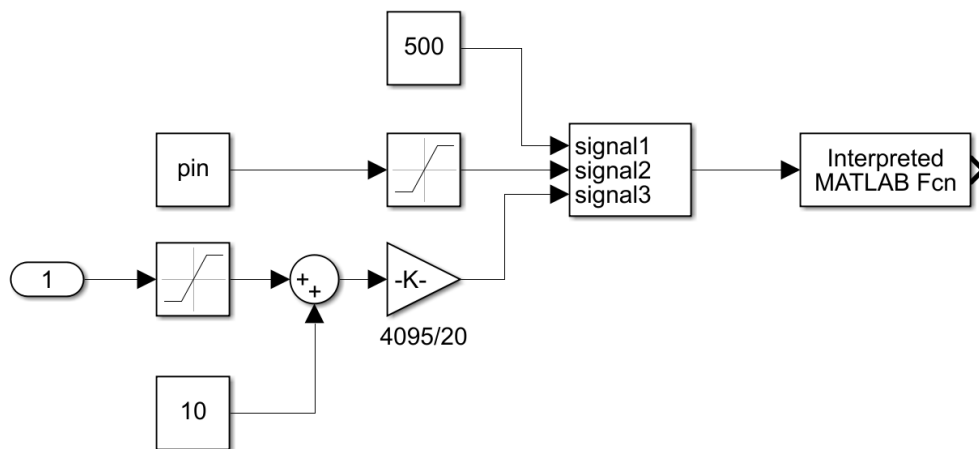
Obrázek 3.14: Bločky pro integraci externích signálů do simulinkového modelu

Všechny bloky vstupů si jsou velmi podobné, a proto zde bude představen pouze blok analogového vstupu, který je zobrazen na obr. 3.15. Jak je patrné z obrázku, blok obsahuje konstantu 200, která určuje, o jaký typ bloku se jedná, a uživatelem zadávaný parametr `pin`. Tyto dvě hodnoty tvoří společně klíč, se kterým je volána funkce `SetOrGetValue` pomocí bločku `MATLAB Fcn`. Software, který zpracovává analogové signály, používá pro hodnoty analogových signálů celé číslo, které odpovídá dané hodnotě analogového signálu. Proto je zde proveden přepočet zpět na hodnotu signálu v rozsahu  $-10\text{ V}$  až  $+10\text{ V}$ .



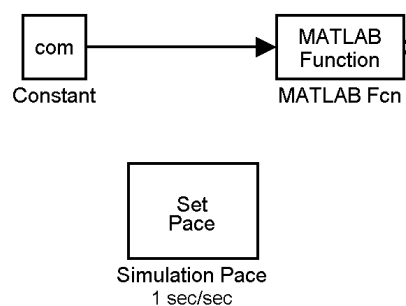
Obrázek 3.15: Simulinkový blok analogového vstupu

Obdobně jako bloky vstupů jsou zapojeny i bloky výstupů, viz obr. 3.16. Funkce spočívá v převedení vstupní analogové hodnoty na celé číslo a společně s klíčem, který je tvořen konstantou 500 a hodnotou konstanty `pin`, jsou tyto hodnoty odeslány pomocí bločku `MATLAB Fcn` jako parametr funkce `SetOrGetValue`.



Obrázek 3.16: Simulinkový blok analogového výstupu

Posledním blokem, který zde bude popsán, je blok `Init` na obr. 3.17. Tento bloček přijímá pouze jeden parametr, který zadává uživatel. Jedná se o číslo sériového portu, na který je deska Arduino připojena. Tato informace je čtena při spuštění simulace a použita ve skriptu pro otevření sériového portu `OpenPort`. Dále je zde použit bloček `Simulation Pace`, který zajistí běh simulace v reálném čase. Jak již bylo řečeno, je nutné, aby byl tento bloček součástí simulinkového modelu, pokud budou využívány některé jiné bločky pro připojení externích signálů. Jinak nemůže být otevřen sériový port a simulace skončí chybou. Detekování COM portu by bylo možné provést softwarově, ale z důvodu jednoduchosti byla zvolena tato možnost, kdy je hodnota zadávána ručně.

Obrázek 3.17: Simulinkový blok `Init`

Pro všechny bloky, které je možné využít pro připojení externích signálů byla vytvořena knihovna `ArduCOP`, pomocí které je možné jednoduše přidávat bločky do simulinkového modelu. Knihovnu je potřeba nakopírovat do požadované složky a poté nastavit

v Matlabu cestu k této složce, což lze nejjednodušeji provést kliknutím pravého tlačítka myši na složku s knihovnou a poté zvolit položku `Add to Path`. Dále je potřeba vytvořit, nebo upravit funkci `slblocks.m`, pomocí kterého bude knihovna zobrazena v `Simulink Library Browseru`. Funkce `slblocks` se pro novější a starší verze Simulinku mírně liší, proto je vhodné prostudovat nápovědu k programu Simulink. U novějších verzí Simulinku je též nutné nastavit parametr `EnableLBRepository` pomocí příkazu. Více informací lze nalézt v nápovědě (MATHWORKS, 2022).

### 3.2.3 Software pro Arduino

Hlavní program pro Arduino je na výpisu níže. Jeho hlavním úkolem je inicializace všech potřebných komponent pro běh programu, což zajišťuje funkce `setup`. Dále program ve smyčce kontroluje stav sériové sběrnice USART. Pokud je přijat validní řetězec znaků, je tento řetězec zpracován funkcí `parseLine`. Tato funkce rozdělí vstupní řetězec na klíč a požadovanou hodnotu. Podle klíče je následně volána odpovídající funkce.

```
#include "MCP3208.h"
#include"MAX536.h"
#include "Pwm.h"

#define MAX_CHARS 16
#define QUAD_A 2
#define QUAD_B 3

enum state{INIT=1,ANALOG_IN=2,QUAD=3,DIGITAL_IN=4,ANALOG_OUT=5,PWM=6,DIGITAL_OUT=7};
char buf[MAX_CHARS];
char digitalOutput[] = {62,63,64,65,66,67,68,69,22,23,24,25,26,27,28,29};
char digitalInput[] = {54,55,56,57,58,59,60,61,35,34,33,32,31,30,14,15};
int count = 0;
int increment = 0;
char c;
MCP3208 adc;
MAX536 dac;
Pwm pwm;

//inicializace
void setup() {
    for(unsigned char i = 0; i < 16; i++){
        pinMode(digitalOutput[i], OUTPUT);
```



```

    digitalWrite(digitalOutput[i], LOW);
}
attachInterrupt(digitalPinToInterrupt(QUAD_A), interruptQuadA, RISING);
attachInterrupt(digitalPinToInterrupt(QUAD_B), interruptQuadB, RISING);
Serial.begin(115200);
adc.begin();
dac.begin();
pwm.begin();
count = 0;
increment = 0;
}

//rozdeli prijaty retezec na klic a hodnotu
void parseLine(){
    int index = atoi(strtok(buf, ","));
    unsigned int value = atol(strtok(NULL, "\n"));
    //zavola pozadovanou funkci podle klice
    switch (index/100){
        case INIT:          setup();
                           break;
        case DIGITAL_OUT:   digitalWrite(digitalOutput[index%100], value);
                           break;
        case ANALOG_OUT:    dac.setValueDac(index%100, value);
                           break;
        case DIGITAL_IN:
            sprintf(buf, "%d,%d\n", index, digitalRead(digitalInput[index%100]));
            Serial.write(buf);
            break;
        case ANALOG_IN:     sprintf(buf, "%d,%d\n", index, adc.readADC(index%100));
            Serial.write(buf);
            break;
        case QUAD:          sprintf(buf, "%d,%d\n", index, increment);
            Serial.write(buf);
            break;
        case PWM:           pwm.setPWM(index%100, value);
            break;
        default:            sprintf(buf, "E\n");//send error message
            Serial.write(buf);
            break;
    }
}
}

```

```

//hlavni smycka programu
void loop() {
  if(Serial.available()){
    c = Serial.read();
    buf[count++] = c;
    if((c == '\n') || (count == MAX_CHARS -1)){
      buf[count] = '\0';
      count = 0;
      parseLine();
    }
  }
}

//preruseni pri nabezne hrane na vstupu ICROA
void interruptQuadA(){
  if(QUAD_B) increment--;
  else increment++;
}

//preruseni pri nabezne hrane na vstupu ICROB
void interruptQuadB(){
  if(QUAD_A) increment++;
  else increment--;
}

```

Jak je vidět z výpisu kódu, jsou v hlavním programu zpracovávány pouze požadavky pro digitální vstupy a pro digitální výstupy. Pro analogové vstupy, výstupy a PWM výstupy jsou použité funkce implementovány v samostatných .cpp souborech a jejich deklarace v odpovídajících hlavičkových souborech .h. Pro analogové vstupy je to soubor MCP3208.h, pro analogové výstupy MAX538.h a pro PWM výstupy se jedná o soubor Pwm.h. Toto rozdělení bylo zvoleno především kvůli přehlednosti a rychlé změně kódu.

Ve spodní části uvedeného výpisu se nacházejí dvě funkce pro obsluhu přerušení. Tyto funkce jsou volány při detekci změny stavu signálu na vstupech pro inkrementální enkodér. Toto přerušení je generováno pouze při detekci náběžných hran vstupního signálu. Pokud by byla požadována větší přesnost, je možné povolit detekci i sestupných hran, a tím i zvýšit rozlišení odměřování. Samozřejmě by bylo potřeba upravit i vyhodnocovací logiku uvnitř těchto funkcí.

# Kapitola 4

## Oživení rozhraní a uvedení do provozu

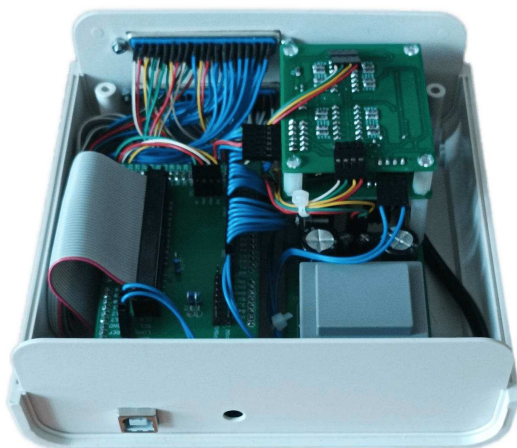
V první části této kapitoly bude popsána finální mechanická konstrukce a její mechanické uspořádání v montážní krabici. V druhé části pak budou uvedeny některé naměřené signály získané během ožívování a testování finálního rozhraní.

### 4.1 Popis finálního výrobku

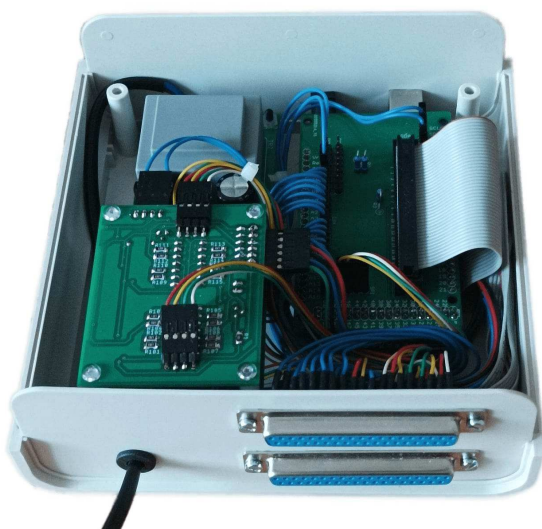
Finální komunikační rozhraní je na obr. 4.1 a obr. 4.2. Celé rozhraní bylo navrhováno tak, aby jej bylo možno umístit do krabičky PP079 o rozměrech 155 mm×157 mm×69 mm. Z tohoto důvodu byly jednotlivé logické celky umístěny na samostatné desky plošných spojů. Kompletní rozhraní obsahuje pět desek. Základní deskou je Arduino Mega, na jehož konektory je nasazena deska, která slouží jako propojovací a je na ní umístěna většina vstupních a výstupních vývodů. Na dalších dvou deskách jsou umístěny AD a DA převodníky pro zpracování analogových signálů. Poslední deska slouží jako napájecí zdroj.

Při mechanické konstrukci rozhraní nenastaly žádné vážnější problémy. Většina součástí byla nakoupena u firmy TME, pouze AD a DA převodníky byly zakoupeny u firmy Mouser. Desky byly navrženy v programu Eagle a pomocí vygenerovaných Gerber dat zhotoveny firmou JLCPCB. Cena pěti kusů jednoho motivu je dva dolary, takže i přes dodatečné náklady na dopravu a clo je cena nesrovnatelně nižší oproti tuzemským i evropským dodavatelům. Drobné komplikace nastaly při propojování jednotlivých desek. Číslování vývodů mikroprocesoru a vývodů Arduina se neshodují a při práci s konkrétními

vývody mikroprocesoru bylo potřeba velké obezřetnosti při propojování všech 74 pinů konektorů X1 a X2.



Obrázek 4.1: Finální zařízení – přední pohled



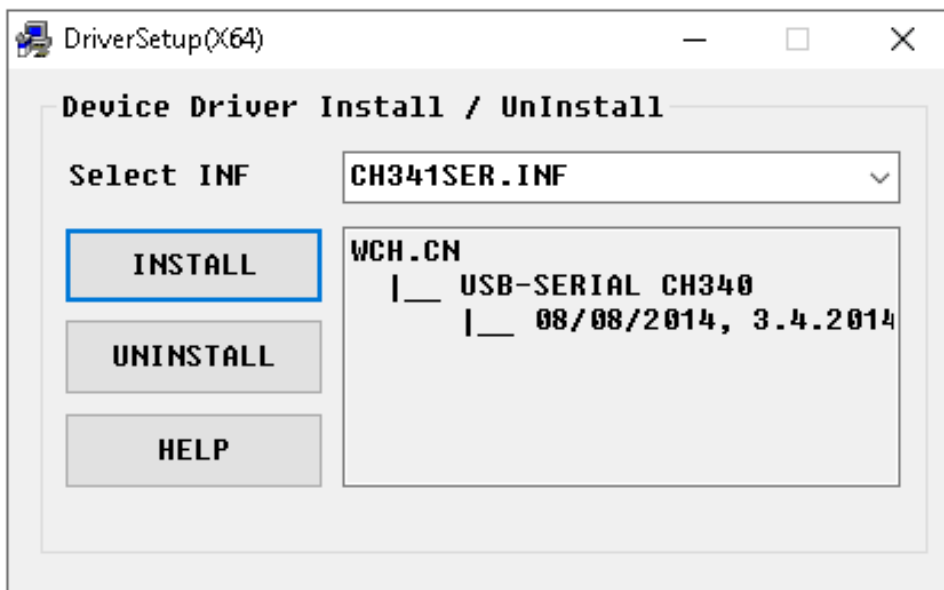
Obrázek 4.2: Finální zařízení – zadní pohled

Jak je patrné z obr. 4.1, přední panel obsahuje pouze USB konektor typu B pro komunikaci s PC. Dále se zde nachází konektor pro napájení Arduina. Tento zdroj se může pohybovat v rozsahu 7 V až 12 V. Tento externí zdroj je vhodné použít při větším proudovém odběru z digitálních výstupů, ale není to podmínkou pro funkci zařízení. Pokud není tento zdroj použit, je napájení Arduina řešeno přímo z USB sběrnice. Na obr. 4.2 je

zachycena zadní strana zařízení, na které jsou umístěny konektory X1 a X2. Dále je zde vyveden síťový napájecí kabel.

## 4.2 Výsledky získané při testování

Před prvním spuštěním aplikace je nutné provést některé kroky. Zejména zkontrolovat, zda jsou dostupné ovladače pro obvod CH340, který je součástí desky Arduino a zajišťuje komunikaci pomocí sériového portu. Pokud je v počítači nainstalované vývojové prostředí Arduino IDE, jsou ovladače již nainstalované. V opačném případě je nutné ovladače doinstalovat. Ovladače lze stáhnout například ze stránek Arduined (ADRUINED, 2023) nebo jsou uloženy na příloženém DVD. Instalace je velice jednoduchá. Stačí stažený soubor rozbalit a spustit instalační .exe soubor. Poté bude zobrazeno dialogové okno, jak je vidět na obr. 4.3. Nyní stačí stisknout tlačítko Install.

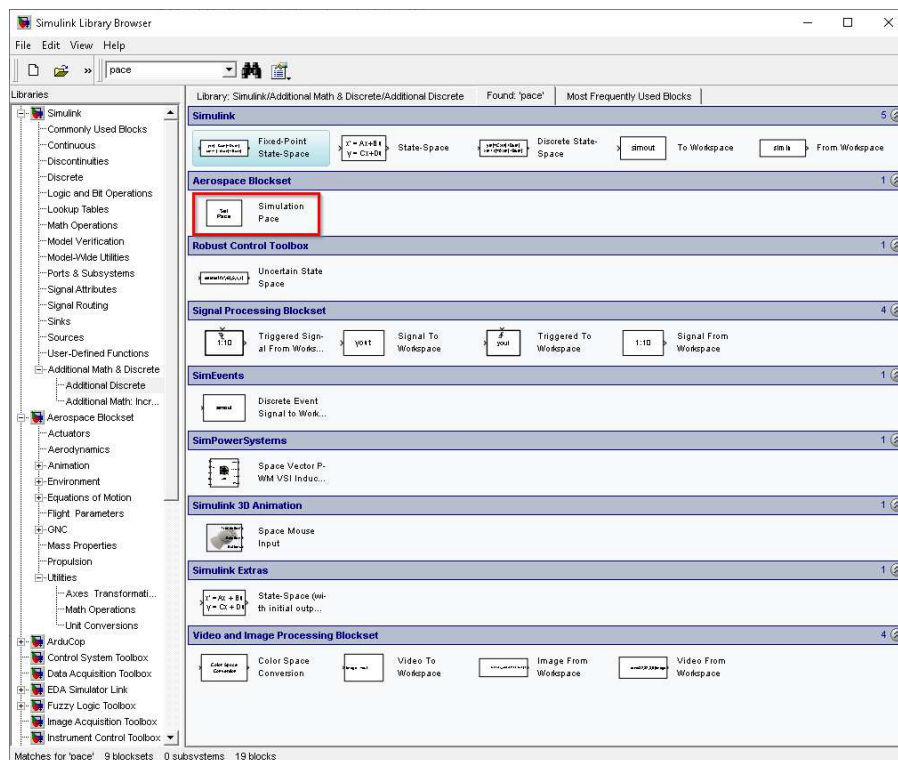


Obrázek 4.3: Dialogové okno instalace ovladače pro CH340

Dále je potřeba ověřit, zda je v Simulinku dostupný bloček `Simulation Pace`, jak je vidět na obr. 4.4. Tento bloček umožňuje běh simulace v reálném čase, respektive téměř v reálném čase a je umístěn v bloku `Init`. Tento bloček se pouze snaží udržet nastavený počet vzorků simulace za sekundu. Výchozí počet vzorků je třicet za sekundu. Takto vysoká frekvence snímání je v případě tohoto rozhraní zbytečně vysoká, proto čtení

dat z Arduina je prováděno pouze třikrát za sekundu. Z tohoto důvodu byla obnovovací frekvence simulinkového modelu nastavena na čtyřikrát za sekundu. Z těchto dvou údajů je patrné, že celková rychlost vzorkování externích signálů je závislá právě na těchto dvou hodnotách. Změnou těchto parametrů lze zvýšit rychlost přenosu dat, ale velmi záleží na hardwarovém vybavení počítače.

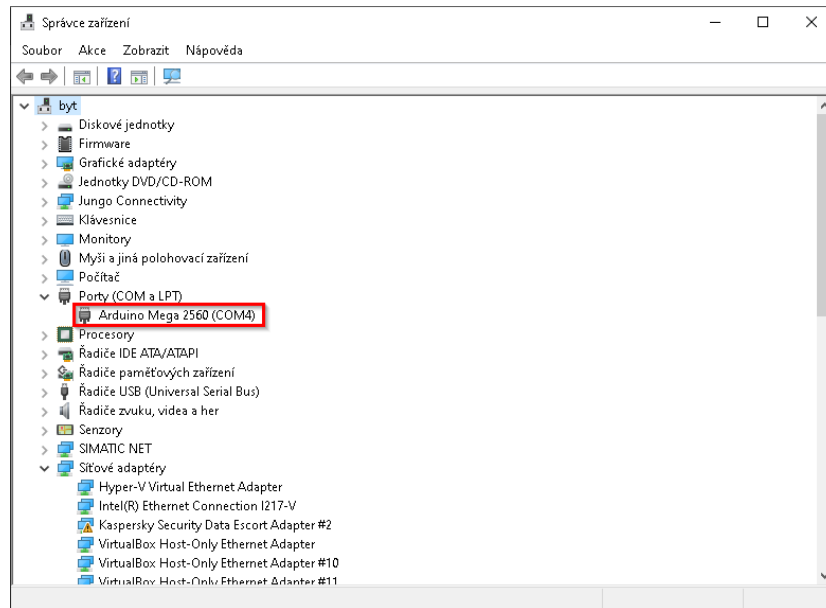
V neposlední řadě je potřeba zmínit, že Simulink a systém Windows běží v multitaskovém režimu a správa těchto vláken (tasků) je plně v režii operačního systému. Proto může dojít k rozdílu v reálném a simulovaném čase. Proto se nedoporučuje při běhu simulace spouštět další, časově nebo výpočtově náročné úlohy. Nicméně při použití ve školní laboratoři zpravidla běží pouze simulace a výsledky lze považovat za reálné v čase.



Obrázek 4.4: Knihovna Simulinku s bločkem Simulation Pace

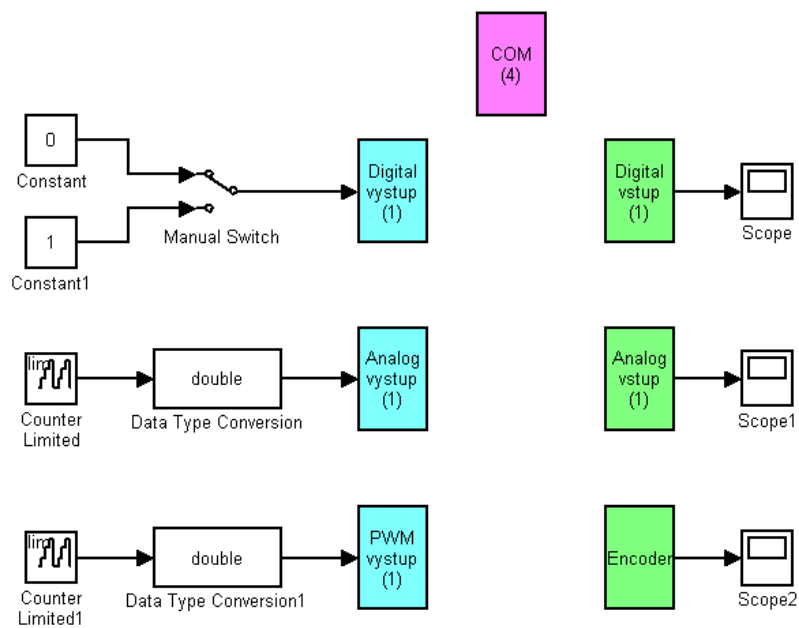
Pokud tento bloček **Simulation Pace** není v Simulinku dostupný, je ho potřeba nainstalovat. Podrobný postup je popsán na stránkách NPSWiki (NPSWIKI, 2023).

Poslední věc, kterou bude potřeba určit, je číslo portu, na který je rozhraní připojeno. To je možno zjistit ve Správci zařízení operačního systému, jak je vidět na obr. 4.5. Toto číslo je nutné zadat do bloku **Init**.

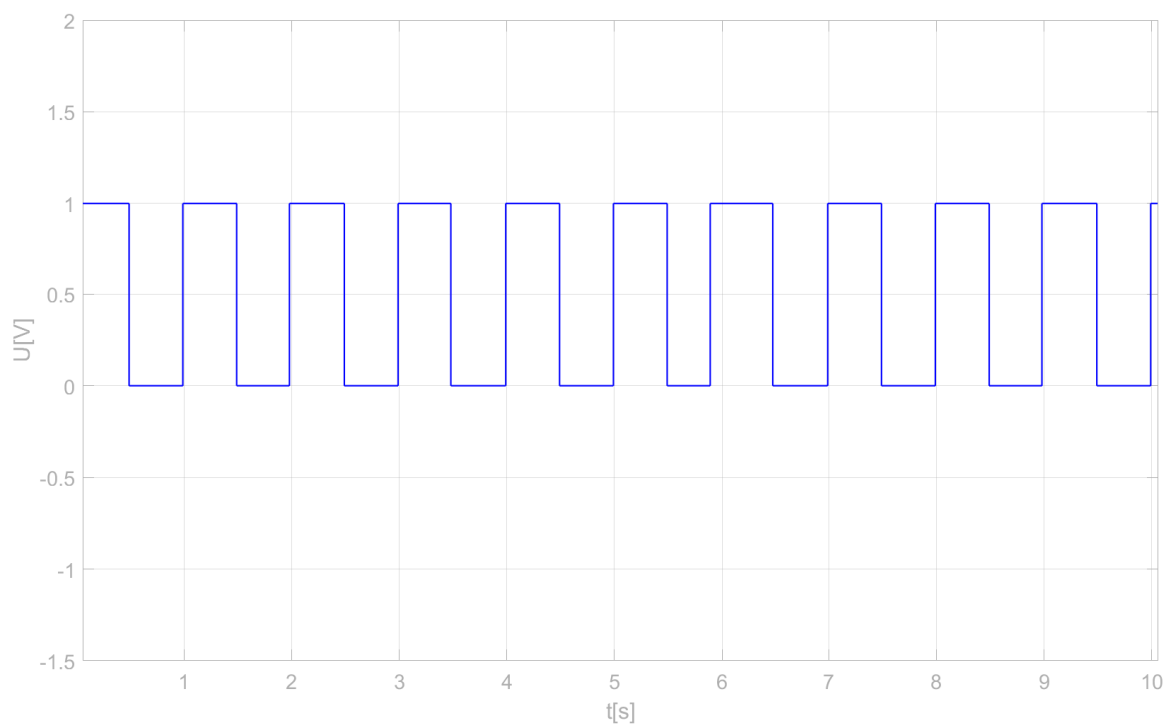


Obrázek 4.5: Správce zařízení systému Windows

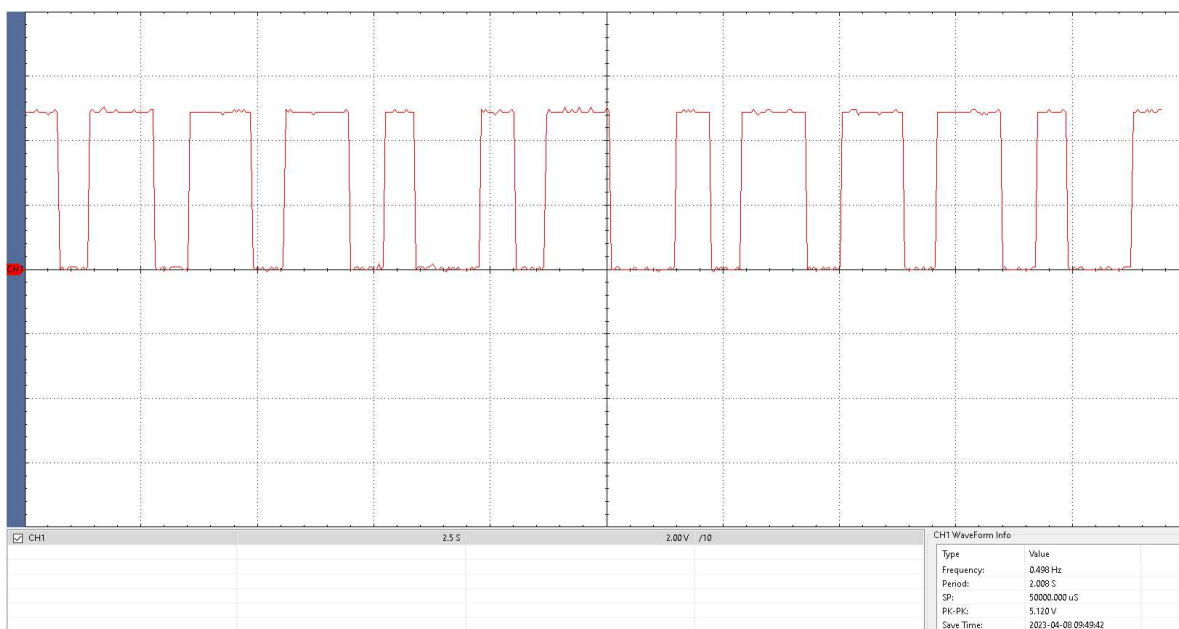
Pro otestování funkcionality byl vytvořen jednoduchý simulinkový model, který je na obr. 4.6. Byly zde použity všechny vytvořené bločky a naměřené hodnoty pro některé signály jsou zobrazeny na následujících obrázcích.



Obrázek 4.6: Testovací simulinkový model

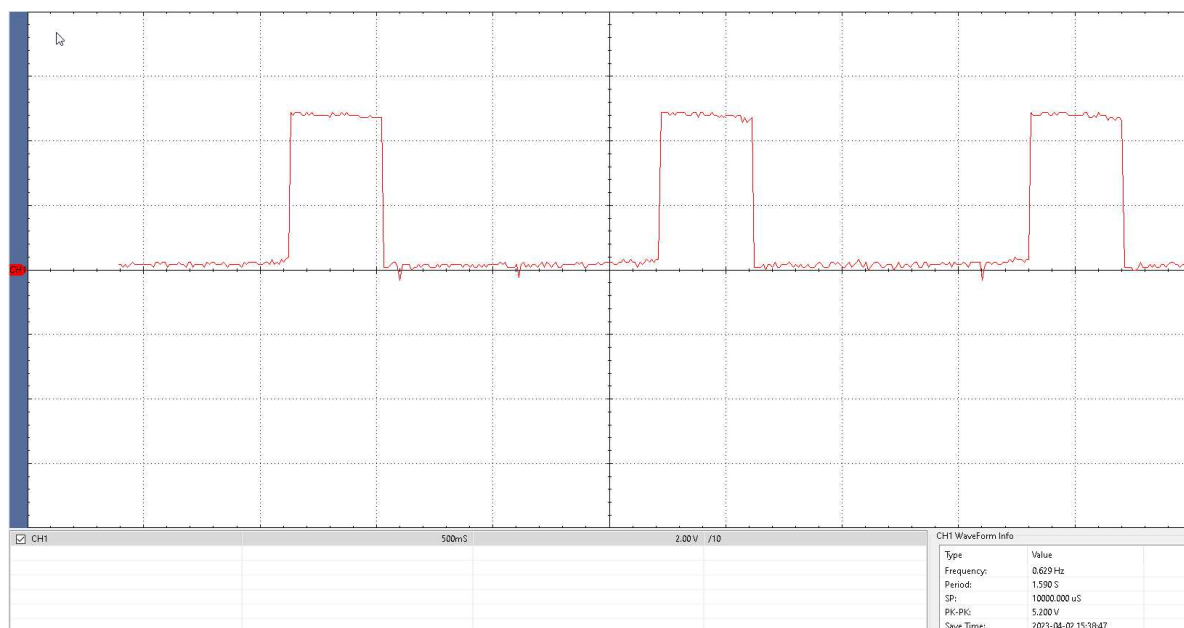


Obrázek 4.7: Pravoúhlý signál 1 Hz na vstupu DIN

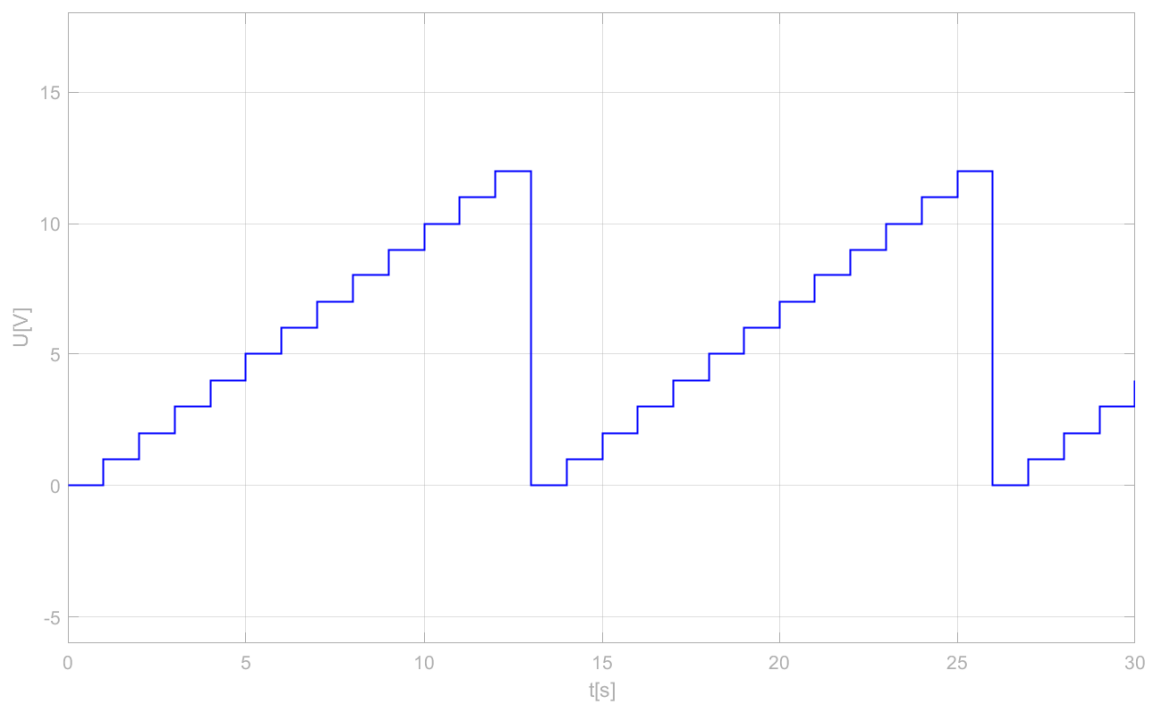


Obrázek 4.8: Pravoúhlý signál 1 Hz na výstupu DOUT (snímek z osciloskopu)

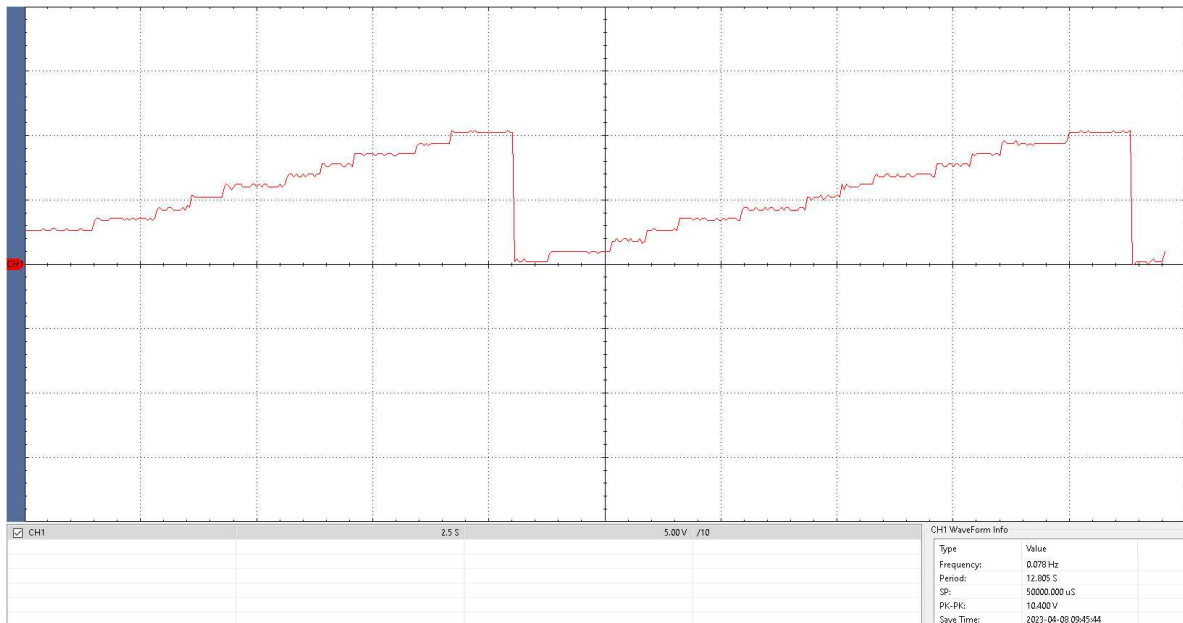




Obrázek 4.9: Generování PWM signálu 1 kHz, střída 25 % (snímek z osciloskopu)



Obrázek 4.10: Data generovaná simulinkovým modelem pro DA výstup



Obrázek 4.11: Naměřená analogová data na DA výstupu (snímek z osciloskopu)

Jak je patrné z naměřených dat, vytvořené simulinkové bločky pro komunikaci s externím zařízením fungují. Problém nastává u periodických signálů vyšší frekvence než je frekvence vzorkovací. Matlab čte/zapisuje data do Arduina každých 300 ms, z toho vyplývá, že signály s vyšší frekvencí nemohou být snímány korektně. Nejhorší situace může nastat pokud se frekvence signálu rovná nebo je celočíselným násobkem vzorkovací frekvence. V takovémto případě může nastat situace, kdy je výstup (vstup) stále na konstantní hodnotě. Toto je však daň za jednoduchost celého zapojení a může být inspirací pro další vylepšení vytvořeného rozhraní.

# Kapitola 5

## Závěr

Cílem této práce bylo vytvořit komunikační rozhraní, které umožní pomocí sběrnice USB připojit externí signály do simulinkového modelu. Pro tento účel byly v programu Simulink vytvořeny bločky, které umožňují připojit do simulinkového modelu digitální signály napěťové úrovně TTL a analogové signály v rozsahu  $\pm 10$  V. Dále je možno generovat pomocí bločku PWM pulzně šířkově modulovaný signál, jehož periodu a střidu lze nastavovat. Dále lze monitorovat vstupy inkrementálního enkodéru a následně je zpracovávat v simulinkovém modelu.

Hlavní hardwarovou komponentou celého rozhraní je Arduino Mega, jehož základem je mikroprocesor AVR ATmega2560. S využitím tohoto mikroprocesoru lze implementovat všechny požadované funkce rozhraní. Pro zpracování analogových signálů bylo potřeba použít analogově digitální a digitálně analogové převodníky. Pro měření analogových signálů byl použit obvod MCP3208 a pro generování analogového signálu z digitální podoby byl použit obvod MAX536. Oba tyto obvody zpracovávají signály pouze v rozsahu 0 V až +5 V, proto bylo potřeba vytvořit převodník napěťových úrovní pomocí operačních zesilovačů typu LM324. Celá analogová část je napájena z napájecího zdroje  $\pm 12$  V, který je součástí zařízení. Napájení digitální části je řešeno přímo ze sběrnice USB, případně lze použít externí napájecí zdroj 7 V až 12 V.

Jako největší problém při ožívování rozhraní se ukázala správná synchronizace simulinkového modelu a čtení/zapisování dat do fyzického zařízení. Pro získání co nejrelevantnějších dat v reálném čase je potřeba co největší obnovovací frekvence, ale zároveň je potřeba zajistit dostatečně rychlý, běh simulinkové simulace samotné. Tyto protichůdné požadavky se podařilo vyřešit pomocí simulinkového bločku `Simulation Spacer`. Tento bloček upravuje rychlost běhu simulace tak, aby se simulační data obnovovala v požadované počtu vzorků za sekundu.

Toto rozhraní si neklade za cíl konkurovat komerčním, profesionálním zařízením, ale snahou bylo vytvořit funkční zařízení v rozumné finanční rovině. Zařízení je jednoduché, cenově dostupné a velmi jednoduše modifikovatelné. Pro další vylepšení je potřeba provést další měření zejména ohledně přenosu dat pomocí sériového portu a synchronizace dat se simulinkovým modelem. Tato činnost je z velké části ovlivněna jak hardwarovými možnostmi počítače, tak operačním systémem samotným. Hlubší analýzou systémových funkcí by bylo možné dosáhnout lepších výsledků při komunikaci, a tím i dosažení věrohodné simulace.

# Literatura

- ADRUINED (2023), CH340 Windows 10 driver download [online]. [cit. 2023-04-07], [⟨https://www.arduined.eu/ch340-windows-10-driver-download/⟩](https://www.arduined.eu/ch340-windows-10-driver-download/).
- BOL (2023), ARDUINO MEGA [online]. [cit. 2023-04-07], [⟨https://www.bol.com/be/nl/p/arduino-compatible-mega-2560-r3/9200000049772089/⟩](https://www.bol.com/be/nl/p/arduino-compatible-mega-2560-r3/9200000049772089/).
- BOŠTIČKA, J. (2014), Model vodní elektrárny – elektronika, (Absolventská práce), VOŠ, SŠ, COP Sezimovo Ústí, Sezimovo Ústí.
- DEX (2023), Rozdíl mezi inkrementálním a absolutním rotačním enkodérem [online]. [cit. 2023-04-07], [⟨https://www.dex.sk/⟩](https://www.dex.sk/).
- ERMICRO (2009), H-Bridge Microchip PIC Microcontroller PWM Motor Controller [online]. [cit. 2023-04-07], [⟨http://www.ermicro.com/blog/?p=706⟩](http://www.ermicro.com/blog/?p=706).
- FUTEK (2023), What is an Incremental Encoder [online]. [cit. 2023-04-07], [⟨https://www.futek.com/Incremental-Encoder-Signal-Converter⟩](https://www.futek.com/Incremental-Encoder-Signal-Converter).
- HUMUSOFT (2021), Humusoft–Měřicí karty [online]. [cit. 2023-04-23], [⟨https://www.humusoft.cz/datacq/⟩](https://www.humusoft.cz/datacq/).
- KREJČÍ, A., REITINGER, J., TIHELKA, D. A VANĚK, J. (2018), Úvod do programového prostředí, (Studijní text), Západočeská univerzita v Plzni, FAV, Plzeň.
- MATHWORKS (2022), Create Custom Library [online]. [cit. 2022-10-01], [⟨https://nl.mathworks.com/help/simulink/ug/creating-block-libraries.html⟩](https://nl.mathworks.com/help/simulink/ug/creating-block-libraries.html).
- MAXIM (2011), Datasheet MAX536 [online]. [cit. 2022-07-30], [⟨https://datasheets.maximintegrated.com/en/ds/MAX536-MAX537.pdf⟩](https://datasheets.maximintegrated.com/en/ds/MAX536-MAX537.pdf).

- MICROCHIP (2008), Datasheet MCP3208 [online]. [cit. 2022-07-30], [⟨http://ww1.microchip.com/downloads/en/devicedoc/21298e.pdf⟩](http://ww1.microchip.com/downloads/en/devicedoc/21298e.pdf).
- MICROCHIP (2014), Datasheet ATmega2650 [online]. [cit. 2022-07-30], [⟨https://ww1.microchip.com/downloads/en/DeviceDoc/ATmega640-1280-1281-2560-2561-Datasheet-DS40002211A.pdf⟩](https://ww1.microchip.com/downloads/en/DeviceDoc/ATmega640-1280-1281-2560-2561-Datasheet-DS40002211A.pdf).
- NPSWIKI (2023), Simulink Pacer Block [online]. [cit. 2023-04-07], [⟨https://wiki.nps.edu/display/RC/Simulink+Pacer+Block#SimulinkPacerBlock-2.AddingSimulationPaceFromAerospaceBlockset⟩](https://wiki.nps.edu/display/RC/Simulink+Pacer+Block#SimulinkPacerBlock-2.AddingSimulationPaceFromAerospaceBlockset).
- PAVLÁT, P. (2015), Řízení otáček turbíny pomocí Wattova odstředivého regulátoru, (Absolventská práce), VOŠ, SŠ, COP Sezimovo Ústí, Sezimovo Ústí.
- RABIŇÁK, P. (2014), Model vytápěného domku – elektronika, (Absolventská práce), VOŠ, SŠ, COP Sezimovo Ústí, Sezimovo Ústí.
- ROUBAL, J. (2012), Výukové materiály pro Laboratoř aplikované informatiky na VOŠ, (Bakalářská práce), ČVUT v Praze, MUVS, Praha.
- ROUBAL, J., HUŠEK, P. A KOL. (2011), *Regulační technika v příkladech*, Praha: BEN – technická literatura. ISBN 978-80-7300-260-2.
- SCHENK, C. (2009), MiKTeX [online]. [cit. 2009-06-16], [⟨http://www.miktex.org/⟩](http://www.miktex.org/).
- ŠIKÝŘ, T. (2011), Systém vodního hospodářství – laboratorní model, (Absolventská práce), VOŠ, SŠ, COP Sezimovo Ústí, Sezimovo Ústí.
- SIMULINK TEAM (2023), Simulink Support Package for Arduino Hardware [online]. [cit. 2023-04-07], [⟨https://nl.mathworks.com/matlabcentral/fileexchange/⟩](https://nl.mathworks.com/matlabcentral/fileexchange/).
- TEXAS INSTRUMENTS (2015), Datasheet LM324 [online]. [cit. 2022-07-30], [⟨https://www.ti.com/lit/ds/snosc16d/snosc16d.pdf⟩](https://www.ti.com/lit/ds/snosc16d/snosc16d.pdf).
- TEXAS INSTRUMENTS (2022), Datasheet TL431 [online]. [cit. 2022-07-30], [⟨https://www.ti.com/lit/ds/symlink/tl431.pdf?ts=1659148839624⟩](https://www.ti.com/lit/ds/symlink/tl431.pdf?ts=1659148839624).

# Příloha A

## Obsah přiloženého DVD

K této práci je přiloženo DVD s následující adresářovou strukturou.

- Absolventská práce v  $\text{\LaTeX}$ 2 $\epsilon$
- Datové listy
  - ATmega2650.pdf
  - LM324.pdf
  - MAX536.pdf
  - MCP3208.pdf
  - TL431.pdf
  - 78xx.pdf
  - 79xx.pdf
- Elektronika
  - Analogové vstupy
  - Analogové výstupy
  - Arduino Mega
  - Napájecí zdroj
  - Základní deska
- Software
  - Driver CH340 pro Windows

- RealTime Pacer
- Software Arduino Mega
- Software Matlab
  
- Kubu\_AP\_2022\_2023.pdf – absolventská práce ve formátu PDF



# Příloha B

## Použitý software

**Arduino IDE 1.8.19** [⟨https://www.arduino.cc⟩](https://www.arduino.cc)

**Eagle 9.1.0** [⟨https://www.autodesk.com⟩](https://www.autodesk.com)

**Inkscape 1.1** [⟨https://www.inkscape.org⟩](https://www.inkscape.org)

**MATLAB/Simulink R2010b** [⟨https://www.mathworks.com⟩](https://www.mathworks.com)

**Texmarker 5.0.4** [⟨https://www.xmlmath.net⟩](https://www.xmlmath.net)

Software z výše uvedeného seznamu je buď volně dostupný, nebo jeho licenci toho času vlastní Vyšší odborná škola, Střední škola, Centrum odborné přípravy, Sezimovo Ústí, Budějovická 421, kde autor téhož času studoval a vytvořil tuto práci.



# Příloha C

## Časový plán absolventské práce

Činnost	Časová náročnost	Termín ukončení	Splněno
návrh hardwaru	4 týdny	31.03.2022	03.04.2022
nákup elektronických součástek	2 týdny	30.04.2022	25.04.2022
návrh a výroba plošných spojů	5 týdnů	30.07.2022	17.07.2022
osazení desky, testování HW	1 měsíc	31.08.2022	25.08.2022
vytvoření softwaru	4 týdny	31.09.2022	17.09.2022
testování softwaru a hardwaru	2 měsíce	30.11.2022	18.03.2023
kompletní text AP	4 měsíce	31.03.2023	31.03.2023
tisk a vazba AP	2 týdny	14.04.2023	28.04.2023



# Příloha D

## Rozpočet projektu

Následující tabulka uvádí finanční rozpočet modelu zahrnující nákupy jednotlivých součástí a zakázky realizované mimo školu. Ceny jsou uvedeny včetně DPH a obvykle včetně poštovného a balného.

Tabulka D.1: Finanční rozpočet projektu

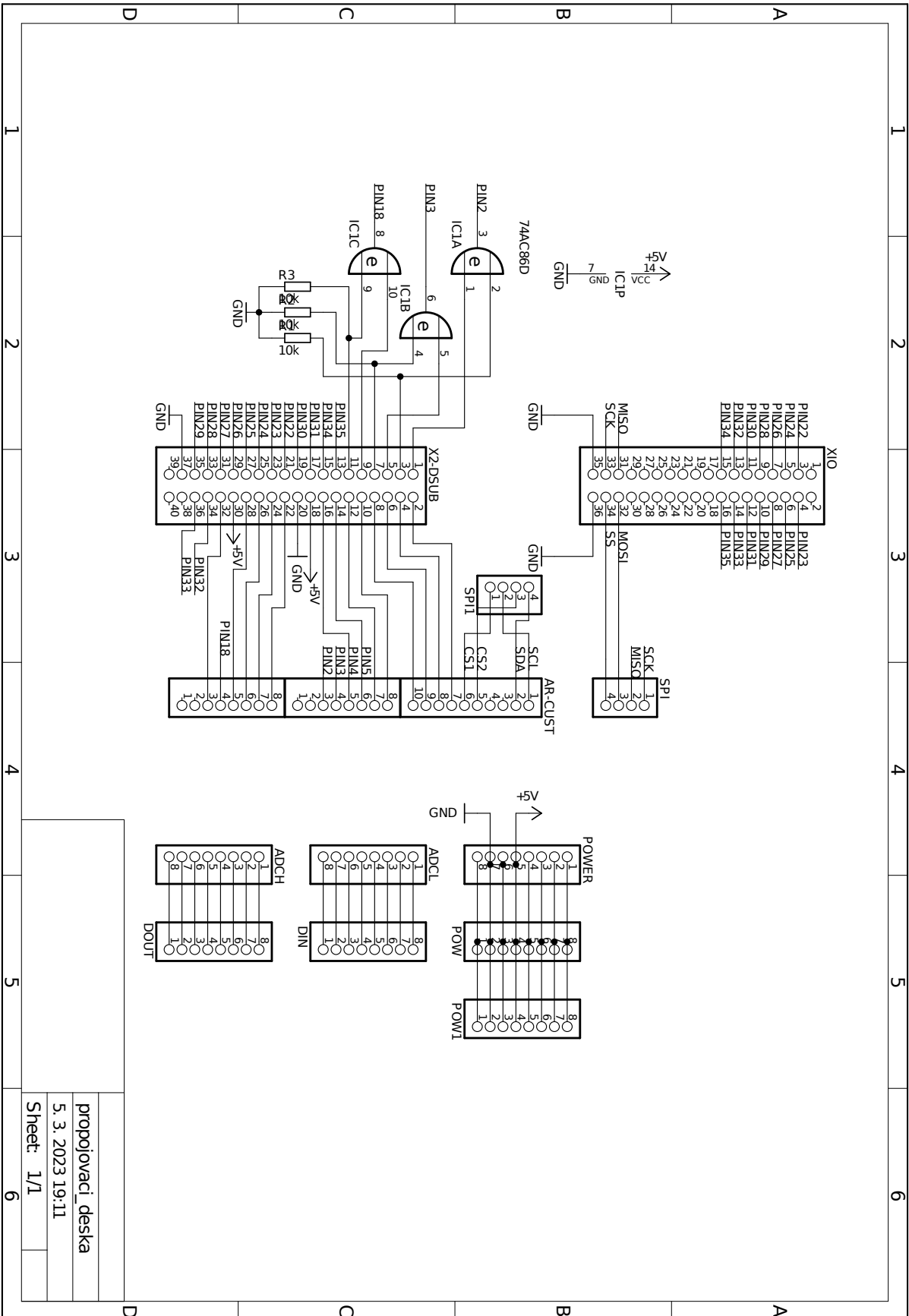
Součástka	Kusů	Cena za kus	Cena celkem
DA převodník MAX536	2	1 200,-	2 400,-
AD převodník MCP3208	1	125,-	125,-
Arduino Mega	1	470,-	470,-
Operační zesilovač LM324	4	11,-	44,-
Napěťová reference TL431	2	12,-	44,-
Transformátor 2x12V	1	110,-	110,-
Desky plošných spojů	4	36,-	144,-
Ostatní součástky	-	-	400,-
Celkem	-	-	3 737,-



# Příloha E

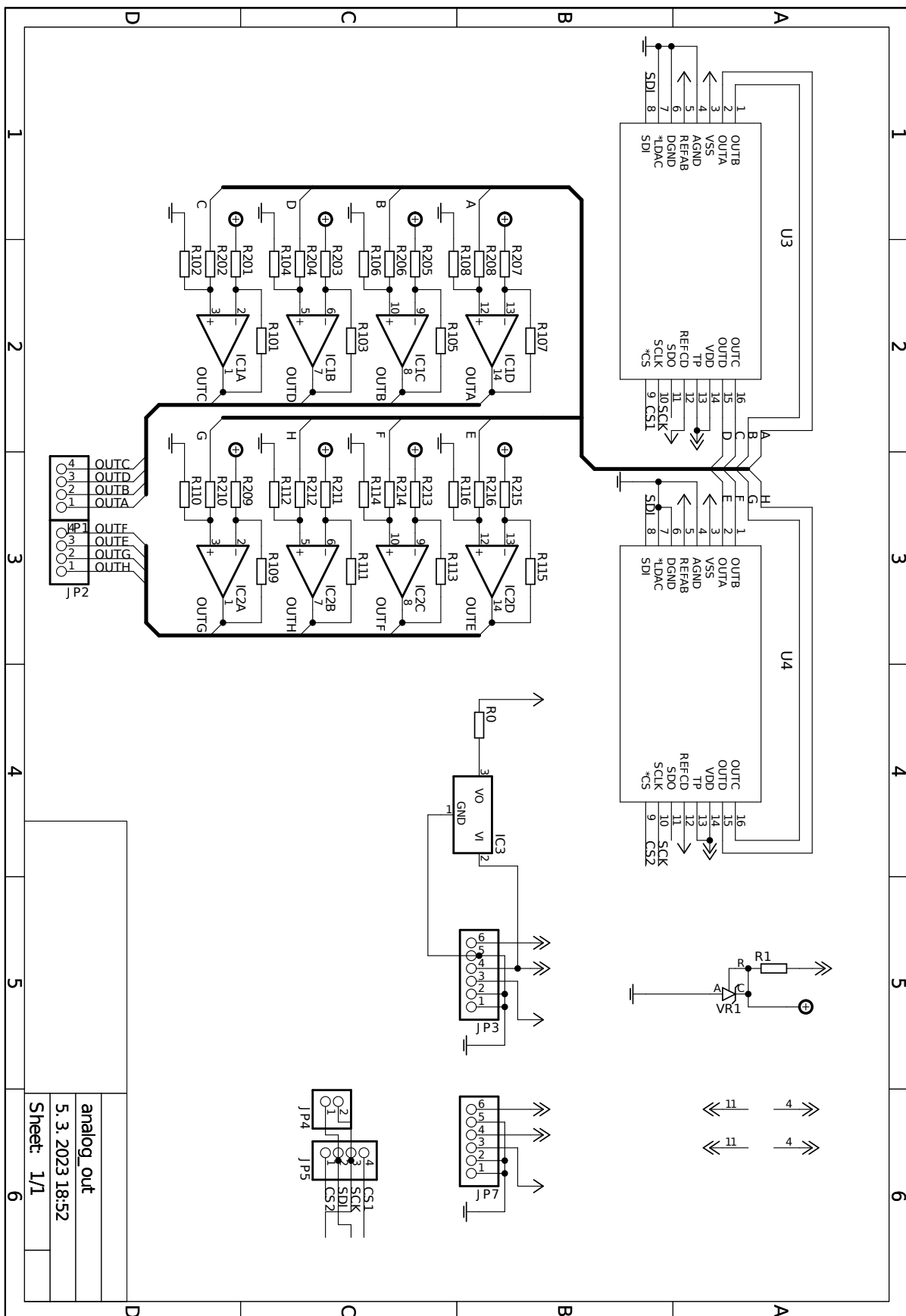
## Schémata zapojení elektroniky

Tato příloha obsahuje schémata zapojení elektroniky, která byla vyráběna. Neobsahuje schéma zapojení desky Arduino Mega, která byla zakoupena jako hotový výrobek a její schéma zapojení lze nalézt na přiloženém DVD.









analog\_out  
 5.3.2023 18:52  
 Sheet: 1/1





# Příloha F

## Seznam součástek

Tabulka F.1: Seznam součástek propojovací desky

Part	Value	Device	Description
ADCH	8x1F-H8.5	PINHD-1X8	PIN HEADER
ADCL	8x1F-H8.5	PINHD-1X8	PIN HEADER
AR-CUST	10x1F-H8.5	PINHD-1X10	PIN HEADER
COM	8x1F-H8.5	PINHD-1X8	PIN HEADER
DIN	8x1F-H8.5	PINHD-1X8	PIN HEADER
DOUT	8x1F-H8.5	PINHD-1X8	PIN HEADER
IC1	74AC86D	74AC86D	OR gate
POW	8x1F-H8.5	PINHD-1X8	PIN HEADER
POW1	8x1F-H8.5	PINHD-1X8	PIN HEADER
POWER	8x1F-H8.5	PINHD-1X8	PIN HEADER
PWML	8x1F-H8.5	PINHD-1X8	PIN HEADER
R1	10k	R-EU_0207/5V	RESISTOR
R2	10k	R-EU_0207/5V	RESISTOR
R3	10k	R-EU_0207/5V	RESISTOR
SPI		PINHD-1X4	PIN HEADER
SPI1		PINHD-1X4	PIN HEADER
X2-DSUB		PINHD-2X20	PIN HEADER
XIO	18x2F-H8.5	PINHD-2X18	PIN HEADER

Tabulka F.2: Seznam součástek desky analogových vstupů

Part	Value	Device	Description
D1 - D4	BAS40-04	BAS40-04	Silicon Schottky Diodes
D5 - D12	5V0	ZENER-DIODE	Z-Diode
D17 - D20	BAS40-04	BAS40-04	Silicon Schottky Diodes
IC1	MCP3208	MCP3208-CI/P	A/D Converters
IC2	LM324N	LM324N	OP AMP
IC3	LM324N	LM324N	OP AMP
IC4	7805TV	7805TV	VOLTAGE REGULATOR
JP1		PINHD-1X6	PIN HEADER
JP2		PINHD-1X2	PIN HEADER
JP3		PINHD-1X4	PIN HEADER
JP4		PINHD-1X4	PIN HEADER
JP6		PINHD-1X4	PIN HEADER
JP7		PINHD-1X6	PIN HEADER
R1	1k	R-EU_R1206	RESISTOR
R2	680R	R-EU_R1206	RESISTOR
R3	3k3	R-EU_R1206	RESISTOR
R4	10k	R-EU_R1206	RESISTOR
R5 - R11	1k	R-EU_R1206	RESISTOR
R117 - R132	120k	R-EU_R1206	RESISTOR
R217 - R232	30k	R-EU_R1206	RESISTOR
VR2	TL431	TL431CLP	VOLTAGE REGULATOR

Tabulka F.3: Seznam součástek desky analogových výstupů

Part	Value	Device	Description
IC1, IC2	LM324N	LM324N	OP AMP
IC3	79L05Z	79L05Z	VOLTAGE REGULATOR
JP1		PINHD-1X4	PIN HEADER
JP2		PINHD-1X4	PIN HEADER
JP3		PINHD-1X6	PIN HEADER
JP4		PINHD-1X2	PIN HEADER
JP5		PINHD-1X4	PIN HEADER
JP7		PINHD-1X6	PIN HEADER
R0	0R	R-EU_0207/7	JUMPER
R1	680R	R-EU_R1206	RESISTOR
R101 - R116	120k	R-EU_R1206	RESISTOR
R201 - R216	30k	R-EU_R1206	RESISTOR
U3	MAX536	MAX536ACWE	DAC
U4	MAX536	MAX536ACWE	DAC
VR1	TL431	TL431CLP	VOLTAGE REGULATOR

Tabulka F.4: Seznam součástek desky napájecího zdroje

Part	Value	Device	Description
B1	2A/1000V	RB1A	RECTIFIER
C1 - C4	100nF/60V	C-EU025-024X044	CAPACITOR
C5, C6	2200uF/25V	CPOL-EUE5-13	POL CAPACITOR
C7, C8	100uF/25V	CPOL-EUE2.5-6	POL CAPACITOR
F1	100mA	19560	FUSE HOLDER
IC1	7812TV	7812TV	VOLTAGE REGULATOR
IC2	7912TV	7912TV	VOLTAGE REGULATOR
JP1, JP2		PINHD-1X3	PIN HEADER
KK1, KK2	D01S	D01S	HEATSINK
TR2	2x12V	EI38-2	TRANSFORMER
X1		AK500/2	CONNECTOR