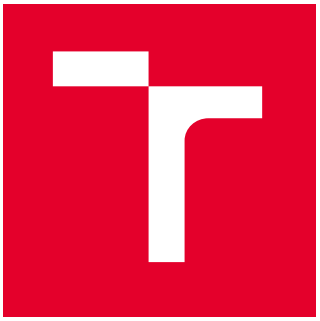
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

Fakulta elektrotechniky
a komunikačních technologií

BAKALÁŘSKÁ PRÁCE

Brno, 2024 Martin Hámor



VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

FACULTY OF ELECTRICAL ENGINEERING AND COMMUNICATION

ÚSTAV VÝKONOVÉ ELEKTROTECHNIKY A ELEKTRONIKY

DEPARTMENT OF POWER ELECTRICAL AND ELECTRONIC ENGINEERING

ŘÍZENI CANOPEN POHONU Z PLC

CANOPEN DRIVE PLC CONTROL

BAKALÁŘSKÁ PRÁCE

BACHELOR'S THESIS

AUTOR PRÁCE — **Martin Hámor**

AUTHOR

VEDOUCÍ PRÁCE — **Ing. Jiří Ctibor**

SUPERVISOR

BRNO 2024

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ

Bakalářská práce

bakalářský studijní program **Silnoproudá elektrotechnika a elektroenergetika**

Ústav výkonové elektrotechniky a elektroniky

Student: Martin Hámor ***ID:*** 240741

Ročník: 3 ***Akademický rok:*** 2023/24

NÁZEV TÉMATU:

Řízeni CANopen pohonu z PLC

POKYNY PRO VYPRACOVÁNÍ:

1. Seznamte se s dodaným BLDC pohonem.
2. Nastudujte CANopen protokol.
3. Navrhněte program do PLC Modicon M241.

DOPORUČENÁ LITERATURA:

[1] ŠMEJKAL, Ladislav a MARTINÁSKOVÁ, Marie. PLC a automatizace. 1. díl, Základní pojmy, úvod do programování. Praha: BEN - technická literatura, 1999. ISBN 80-86056-58-9.

[2] SMEJKAL, Ladislav. PLC a automatizace. 2. díl, Sekvenční logické systémy a základy fuzzy logiky. Praha: BEN - technická literatura, 2005. ISBN 80-7300-087-3.

[3] PFEIFFER, Olaf; AYRE, Andrew a KEYDEL, Christian. Embedded networking with CAN and CANopen. San Clemente: RTC Books, 2003. ISBN 0-929392-78-7.

[4] VERČIMÁK, Mário. Protokol CAN v automatizaci.

[5] PAVLIŠIN, Tomáš. Protokol CAN pro řízení.

[6] Datasheet: PCAN-MicroMod FD DR CANopen Digital 1

Termín zadání: 5.2.2024 ***Termín odevzdání:*** 29.5.2024

Vedoucí práce: Ing. Jiří Ctibor

prof. Ing. Petr Toman, Ph.D.
předseda rady studijního programu

Abstrakt

Hlavním cílem této bakalářské práce bylo vytvořit elektrický pohon ovládaný PLC pomocí protokolu CANopen. Byly vyzkoušeny tři metody přenosu zpráv z PLC pomocí softwaru Schneider Electric Machine Expert Logic Builder, určeného pro programování PLC, a PCAN-View, který příchozí a odchozí zprávy sledoval v reálném čase. Metody ovládání jsou v podrobnosti uvedeny v 6. kapitole této práce, jedna z nich byla úspěšná.

Klíčová slova

PLC, CAN, CANopen, sběrnice, BLDC motor, Modicon M241.

Abstract

The main goal of this bachelor's thesis was to create an electric drive controlled by a PLC using the CANopen protocol. Three methods of message transmission were tested using Schneider Electric Machine Expert Logic Builder software, designed for PLC programming, and PCAN-View, which monitored incoming and outcoming messages in real time. Control methods are detailed in chapter 6, one of these methods was successful.

Keywords

PLC, CAN, CANopen, bus, BLDC motor, Modicon M241.

Bibliografická citace

HÁMOR, Martin. *Řízení CANopen pohonu z PLC*. Brno, 2024. Dostupné také z: https://www.vut.cz/studenti/zav-prace/detail/160597. Bakalářská práce. Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, Ústav výkonové elektrotechniky a elektroniky. Vedoucí práce Ing. Jiří Ctibor.

Prohlášení autora o původnosti díla

Jméno a příjmení studenta:	*Martin Hámor*
VUT ID studenta:	*240741*
Typ práce:	*Bakalářská práce*
Akademický rok:	*2023/24*
Téma závěrečné práce:	*Řízení CANopen pohonu z PLC*

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 jsem si plně vědom následků porušení ustanovení § 11 a následujících autorského zákona č. 121/2000 Sb., včetně možných trestněprávních důsledků vyplývajících z ustanovení části druhé, hlavy VI. díl 4 Trestního zákoníku č. 40/2009 Sb.

V Brně dne: 29. května 2024

Poděkování

Děkuji Ing. Jiřímu Ctiborovi za odborné vedení, konzultace, vstřícnost a cenné rady, které mi pomohly k vytvoření této práce.

V Brně dne: 29. května 2024

Obsah

SEZNAM OBRÁZKŮ

SEZNAM TABULEK

Úvod

V dnešní době je v průmyslu kladen velký důraz na automatizaci výroby. S postupným technologickým rozvojem a vývojem mikroelektroniky v posledních 50 letech se možnosti automatizace velmi rozrostly, jedním z technologických zařízení umožňující ovládání a řízení výroby je PLC. Na trhu je dnes spousta výrobců těchto automatů, mezi které se řadí Siemens, Allen-Bradley, Schneider Electric, EATON, Mitsubishi, které je ale více známé výrobou automobilů, ale i spousta dalších. Ceny se pohybují v rozmezí nižších tisíců korun pro menší a jednoduchá PLC až do stovek tisíc pro ty nejkomplexnější aplikace.

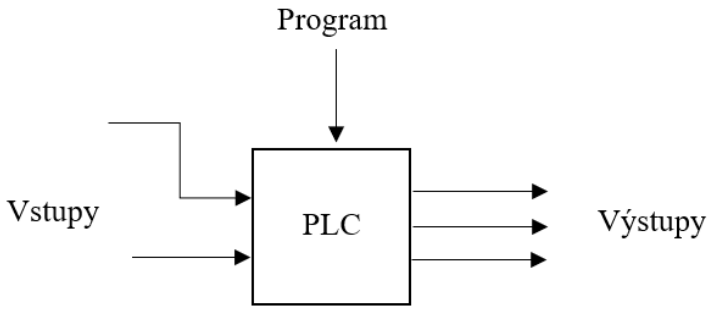
Přenos dat z a do PLC lze jednoduše pomocí různých vstupů a výstupů, mezi které se řadí i různé sběrnice. Jednou z těchto sběrnic je i CAN a s ním spojený protokol CANopen, což je jedno z hlavních témat této bakalářské práce, tedy spolupráce PLC a CANopen k ovládání akčních členů, v tomto případě motoru a k němu připojeného měniče.

Práce obsahuje kapitoly vysvětlující stručné fungování PLC a jejich programování, CAN sběrnici a CANopen, je uveden přehled použitých zařízení – motor, měnič, PLC. Jedna kapitola je věnována BLDC motorům, jejich funkce, charakteristiky a řízení. V poslední části je uveden řídící program a jeho vývoj.

1. PROGRAMOVATELNÉ AUTOMATY

1.1 Základní popis PLC

Programovatelný logický automat (zkratka PLC z angl. Programmable Logic Controller) je zařízení umožňující řízení průmyslových a technologických procesů založené na procesoru používajícím programovatelnou paměť k implementaci logických, časových, matematických a sekvenčních funkcí. Hodnoty vstupu jsou vyhodnoceny programem nastaveným v PLC a podle něj převádí hodnoty na výstup [1]. Celý tento systém lze ilustrovat jednoduchým obrázkem:



Obrázek 1.1 Programovatelný logický automat, převzato z [1].

Logické automaty se řadí do několika kategorií.

1.1.1 Mikro PLC

Nabízejí pevně danou sestavu převážně binárních vstupů a výstupů. Nejmenší mívají 4 vstupy/4 výstupy, pro větší sestavy mívají 6/6, 8/6, 12/12 atd. Takovýto systém se nedá více rozšiřovat. Jejich největší výhodou jsou malé rozměry a nízká cena (nižší tisíce Kč). Nevýhodou již zmíněná nemožnost rozšíření a nízký počet vstupů a výstupů pro komplikovanější aplikace.

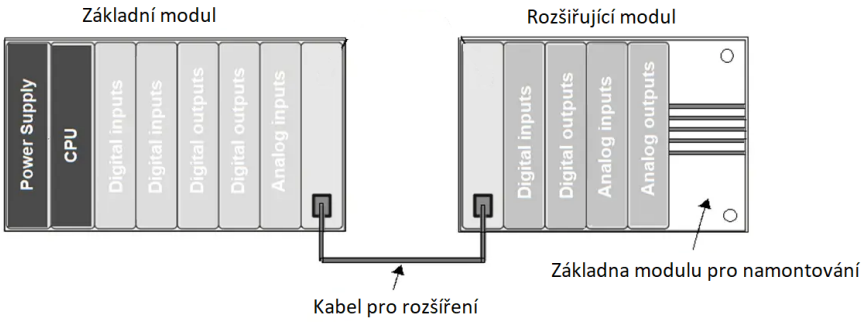
1.1.2 Kompaktní PLC

Tato PLC nabízejí větší variabilitu v konfiguraci, ale pouze omezeně. Některé umožňují přidat k základnímu modulu jen jeden nebo více přídavných modulů z omezeného sortimentu. Jedná se např. o moduly rychlých čítačů, rozšíření počtu vstupů a výstupů, vstupní analogový nebo výstupní analogový modul. Některé systémy umožňují připojení modulu přímo na základní desku vhodného typu. Používají se pro jednoduché aplikace.

1.1.3 Modulární PLC

Jedná se o systém, který poskytuje mnohem větší volnost v konfiguraci. Jsou navrženy tak, aby bylo možné zasouvat libovolné moduly dle potřeby ve vyšších počtech (typicky

4, 6, 8, a 11). Tyto rozšiřovací moduly mohou být připojeny i na vzdálenosti stovek metrů [2]. Obvykle se ale připojují přímo na desku se základním modulem do tzv. „racku“ Jde např. o moduly rozšíření I/O, moduly pro měření a regulaci teploty, řízení servoventilů a proporciálních ventilů, moduly diagnostiky a další [3]



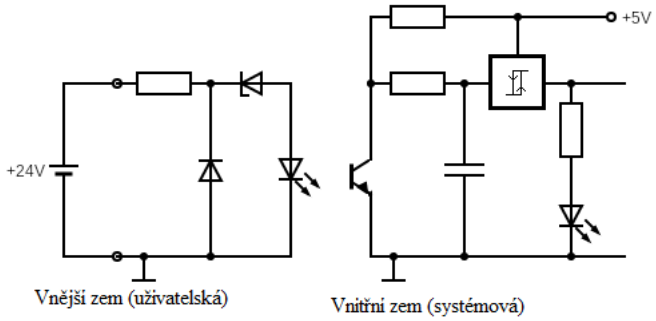
Obrázek 1.2 Příklad spojení modulárního PLC s rozšiřujícím modulem, převzato z [3]

1.2 Vstupy a výstupy PLC

Vstupy PLC jsou dvojího druhu – digitální (binární) a analogové. Na digitální vstupy se připojují tlačítka, přepínače, koncové spínače, senzory vydávající binární signál (ano-ne), detektory přiblížení atd. Digitální vstupy mohou být čistě stejnosměrné, nebo univerzální – střídavé a stejnosměrné. Jsou vybaveny galvanickým oddělením, RC filtrem pro odfiltrování poruchových signálů a diodami pro ochranu proti napěťovým špičkám a přepólování.

Analogové vstupy obecně slouží ke zprostředkování kontaktu PLC se spojitým prostředím. Jedná se např. o snímače teploty, vlhkosti, tlaku, rychlosti, ale třeba i měření elektrického proudu nebo napětí.

Výstupy mohou také být digitální a analogové. Mezi digitální se řadí např. buzení cívek relé nebo stykačů, ovládání signalizace a zobrazovačů, stupňovité řízení pohonů. Mohou být v nevýkonovém provedení do stovek mA s tranzistorem, nebo výkonové s tyristorem, nebo reléové pro spínání větších střídavých a stejnosměrných výkonů. Analogovými výstupy lze spojitě ovládat otáčky motorů, ručkové měřící přístroje nebo jiné spojitě ovládané akční členy [1][2][3]. Dalším vstupem mohou být různé přenosové sběrnice, např. CAN, Ethernet, Profibus atd.



Obrázek 1.3 Stejnosměrný digitální vstup, převzato z [3]

1.3 Provádění programu PLC

Program PLC je prováděn procesorem v cyklech. Jako první dochází ke čtení a vzorkování vstupních signálů, ty jsou uloženy do paměti – vstupy nejsou měřeny kontinuálně, pokud by došlo ke změně vstupu jindy v cyklu, projevila by se až v cyklu dalším. Po načtení všech vstupů dochází k provedení uživatelského programu. Na konci tohoto programu se zapisují a vzorkují výstupy. PLC pak vynuluje příslušné časovače a čítače a cyklus se opakuje [2].

2. PROGRAMOVÁNÍ PLC

Každé PLC operuje podle uživatelsky nastaveného programu. Takový program lze realizovat použitím několika dostupných jazyků – grafických a textových (algebraických). ČSN EN 61131-3 definuje dva grafické a dva textové jazyky, a jedna doplňující sada grafických a ekvivalentních textových prvků (ozn. SFC) [4]. V rámci jednotlivých funkcí je možné tyto jazyky kombinovat.

2.1 Jazyk kontaktních schémat

Jazyk kontaktních schémat (někdy označován jazykem reléových schémat, „Ladder diagram“, „LD“) je grafický. Jedná se o jeden z nejjednodušších [2].

Obrázek 2.1 Příklad programu v jazyku kontaktních schémat vytvořený v programu Mosaic.

Tabulka 2.1 Pravdivostní tabulka bloku RS

S	R	Q1	Funkce
0	0	0/1	Hold
1	0	1	Set
0	1	0	Reset
1	1	0	Reset

Výše uvedený program má funkci zapínání a vypínání světla. Na počátku programu jsou vstupy (kontakty) „Start“ a „Stop“ rozepnuté, na vstupech „S“ a „R1“ bloku „RS_1“ je logická 0. Funkce bloku RS, někdy nazývaným klopným obvodem, je převádět hodnotu vstupu na výstup a tuto hodnotu držet i po odpojení vstupu. Na začátku tedy drží hodnotu 0 a převádí jí na výstup „Q1“. Výstup „Q1“ je přímo spojen s výstupem PLC na „Svetlo“. Pokud sepne relé „Start“, dojde k přenosu vstupní binární hodnoty 1 na vstup „S“ bloku „RS_1“. Kontakty „Start“ a „Stop“ uvažujeme jako tlačítka, tedy na blok RS přichází pouze impuls. Dle pravdivostní tabulky 2.1 je hodnota „S“ převedena na výstup „Q1“. Po uvolnění tlačítka „Start“ blok „RS_1“ na výstupu udržuje 1, to je převáděno na výstup PLC a světlo svítí. Pokud nyní sepne relé (tlačítko) „Stop“, výstup se přepne do 0 a světlo zhasne. Pokud bychom zmáčkli tlačítka „Start“ a „Stop“ současně,

výstup „Q1“ se resetuje, protože R má v tomto případě prioritu. Existuje blok SR, který priorizuje hodnotu S.

Jazyk kontaktních schémat obsahuje celou řadu funkcí, od zmíněných RS a SR bloků také různé časovače, čítače nahoru a dolů, různé logické funkce – AND, OR, XOR, a jejich negace, matematické funkce, případně konverzní funkce mezi datovými typy (např. Real_to_Int apod.).

2.2 Jazyk funkčních bloků

Jazyk funkčních bloků („Function Block Diagram“, „FBD“) je druhým grafickým jazykem [2].

Obrázek 2.2 Příklad programu v jazyku funkčních bloků vytvořený v programu Mosaic.

Tabulka 2.2 Pravdivostní tabulka bloku AND

x	y	x*y
0	0	0
1	0	0
0	1	0
1	1	1

Na obrázku 2.2 je uveden stejný program jako na obrázku 2.1, nyní ale uvažujme vstupy jako spínače, a ne jako tlačítka. Pokud spustíme „Start“, jeho hodnota je udržována na 1. „Stop“ není spuštěn, jeho hodnota je 0. Po negaci nám vzniká 1, ta je přivedena na druhý vstup bloku AND. Podle pravdivostní tabulky nám tato kombinace přenese na výstup 1, světlo svítí. Pokud bychom buď vypnuli spínač „Start“ nebo sepnuli spínač „Stop“, světlo přestává svítit. Kdybychom uvažovali vstupy jako tlačítka, výsledný program by vypadal velmi podobně jako v ladder diagramu.

Obrázek 2.3 Stejný příklad programu v jazyku funkčních bloků při uvažování vstupů jako tlačítek vytvořený v programu Mosaic.

V jazyku funkčních bloků se program nesestavuje z kontaktů, ale především z logických funkcí AND, OR, XOR a jejich negace, NOT, může být ale opět doplněn čítači, časovači, konverzními a porovnávacími funkcemi.

2.3 Strukturovaný text

Strukturovaný text („Structured Text“, „ST“) je obdobou programovacích jazyků PC, je úsporný a názorný [2]. Následující program funguje stejně jako programy v obr. 2.3 a 2.1:

```
  VAR_GLOBAL
  Start : Bool;
  Stop : Bool;
  Svetlo : Bool;
  END_VAR

PROGRAM STUkazka

If start = true
then svetlo := true;
END_IF
If stop = true
then svetlo := false;
END_IF


END_PROGRAM
```

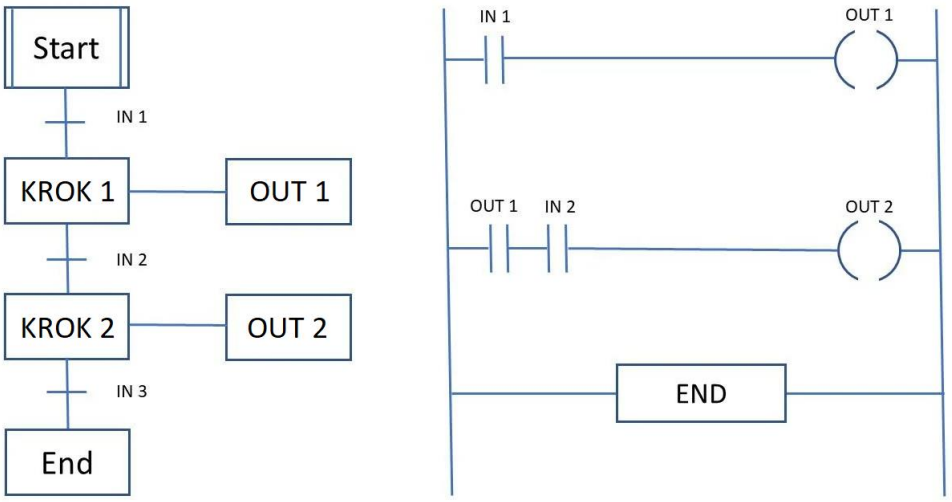
Jako první se definují globální proměnné, poté probíhá samotný program. Po stisknutí tlačítka „Start“ se světlo rozsvítí. Po stisknutí „Stop“ se světlo zhasne. Pokud stiskneme obě tlačítka najednou, světlo se buď zhasne nebo zůstane zhasnuté, to je dáno posloupností kódu, protože podmínková funkce pro zhasnutí je níže. Pokud by byly funkce prohozeny, světlo by se buď rozsvítilo nebo zůstalo rozsvícené.

2.4 Seznam instrukcí

Seznam instrukcí (někdy označován jazykem mnemokódů, „Instruction List“, „IL“) je druhým textovým jazykem. Svojí strukturou připomíná assembler. Program je složen ze sekvence instrukcí, každá na samostatném řádku. Pomocí modifikátorů se vyjadřují podmínky, priority [2][4]. Normou ČSN EN 61131-3 je považován za zastaralý a v dalších vydáních již nebude uváděn [4].

2.5 Sekvenční funkční diagram

Popisuje sekvenční chod řídícího programu, svojí strukturou připomínají „flow chart“. Umožňují rozložení komplikovaných úloh na zvládnutelné části, a přitom zachovat přehled o chování. Používají se k vizualizaci a designu komplexních sekvenčních systémů.



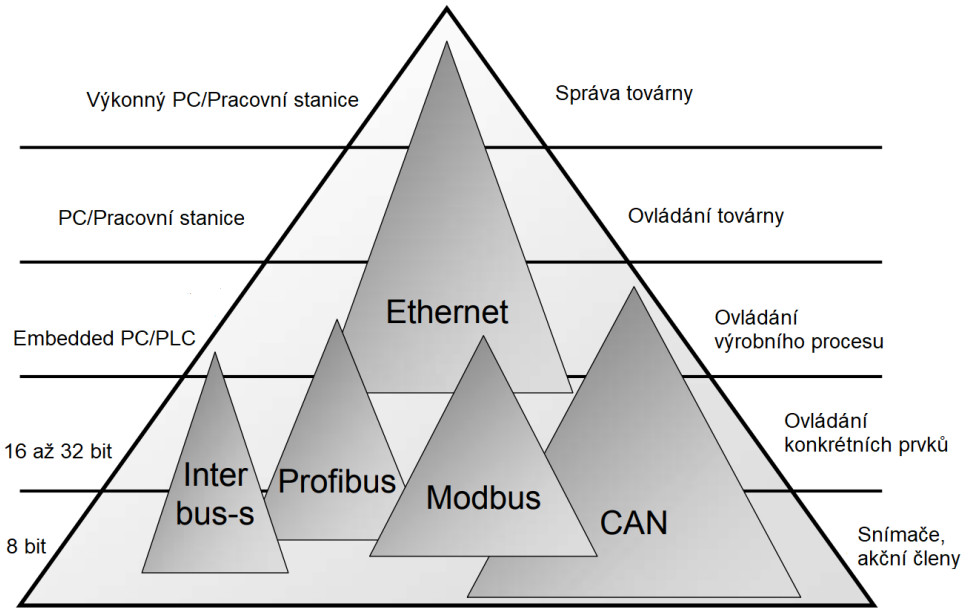
Obrázek 2.4 Příklad programu jako sekvenční funkční diagram a vpravo stejný program v LD, převzato z [5].

První stav reprezentuje start programu. Mezi dvěma stavy se nachází přenosová podmínka. Po dokončení kroku se program přenese do dalšího splněním této podmínky. Přenosová podmínka musí být přítomna mezi všemi kroky. Každý krok má korespondující výstup, který je spojen horizontálně s daným krokem [6][5].

3. CAN A CANOPEN

Datová sběrnice CAN byla vytvořena firmou Bosch v roce 1986, roku 1987 firma Intel přichází se samostatným CAN čipem – 82526, načež firma Phillips Semiconductors vytvořila čip 28C200. V roce 1993 soubor evropských společností a institucí (Newcastle University a Reutlingen University of Applied Sciences) v čele s firmou Bosch začali vyvíjet prototyp CANopen – komunikační protokol pro sběrnici CAN. Hlavním cílem bylo vytvořit ovládací architekturu a zařízení umožňující flexibilních a modulárních kombinací již existujících výrobních buněk. Brzy nato společnosti začali využívat tento protokol ve skutečných aplikacích, ale stále byly nutné další aktualizace a revize, než se CANopen stal stabilním. Rok 1996 dal vzniku verzi 3.0, která je na některých místech používána dodnes [7][8][9]. Sběrnice CAN je velmi využívána především v automobilech, letadlech, výtazích, lodích, lékařských nástrojích, u domácích spotřebičů (pračky, sušičky), ale i v automatizaci [10].

Standartní sběrnice CAN dosahuje 1Mbps, což v dnešní době s rostoucí komplexitou v automobilovém průmyslu není někdy dostačující, u automobilů s vysokými nároky na rychlost sběrnic je nahrazován sběrnicí CAN FD, kde délka jedná zprávy byla zvětšena z 8 na 64 bajtů a rychlost přenosu se zvýšila na 8Mbps. Spolu s CAN FD se používají sběrnice LIN nebo SENT, např. pro ovládání oken (pro nenáročné aplikace), s rychlostmi 40kbps, respektive 30kbps [10].



Obrázek 3.1 Pyramida komunikací v automatizaci, převzato z [8]

3.1 Fyzická vrstva CAN

Sběrnice je vytvořená tak, že všechny uzly (zařízení na sběrnici) dokáží sledovat všechny probíhající datové přenosy. CAN má dobře navrženou detekci chyb a přenášená data se dostanou tam, kam mají. O nadřazenost jednotlivých uzlů se stará protokol, což umožňuje přidávání nebo odebírání uzlů za chodu [11].

3.1.1 Délka sběrnice a rychlost přenosu

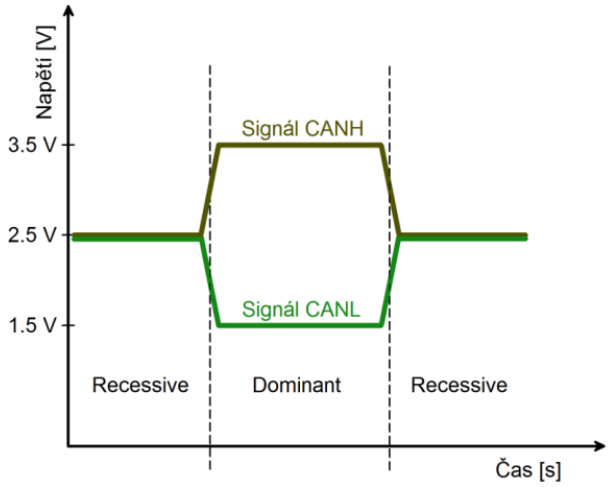
Tabulka 3.1 Vztah délky sběrnice a rychlosti přenosu [12]

Délka sběrnice (m)	Rychlost přenosu (Mbps)
40	1
100	0,5
200	0,25
500	0,1
1000	0,05

Rychlost přenosu klesá s délkou sběrnice. Omezenosti šířky pásma (bandwidth) kabelu ovlivňují přenos a přináší ISI (Inter-Symbol Interference, kdy jeden symbol ruší následující symboly, má podobný efekt jako šum). Rychlost je také ovlivněna zpožděním systému, sumou všech zpoždění do a z uzlu, přibližně 5ns/m kabelu [12].

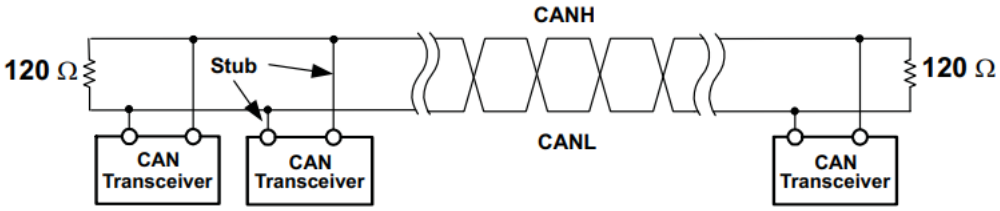
3.1.2 Provedení sběrnice

Fyzikální provedení sběrnice není přesně definováno, liší se podle dané realizace. Normou ISO-11898-2 jsou definovány dva stavy sběrnice – Dominant a Recessive. Pokud všechny uzly vysílají stav Recessive, na sběrnici bude také Recessive; pokud alespoň jeden uzel vysílá stav Dominant, na sběrnici bude stav Dominant. Realizace přenosu je tvořena dvěma vodiči **CANH** a **CANL.** Stav na sběrnici odpovídá rozdílu napětí mezi vodiči. pro stav Recessive je Δu = 0 V, pro Dominant je Δu = 2 V [13]. Vodiče CANH a CANL jsou svázány do kroucené dvojlinky pro omezení rušení (oba vodiče jsou rušeny stejně, relativní chyba diferenciálního napětí Δu je tak nulová) [12].



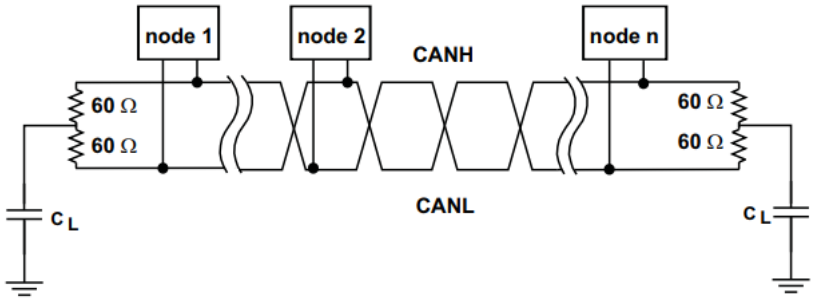
Obrázek 3.2 Průběh napětí na vodičích CANH a CANL pro Recessive a Dominant, převzato z [13]

Pro eliminaci odrazů na sběrnici se používají dva zakončovací rezistory 120 Ω, pro rozvětvenou síť by tyto rezistory měly být připojeny na nejvzdálenějších místech sítě [12][13].



Obrázek 3.3 Zapojení sběrnice se zakončovacími rezistory, převzato z [12]

Další způsob eliminace odrazů je využití tzv. split terminace, kdy je mezi dvěma 60 Ω sériovými rezistory zapojený uzemněný kondenzátor[12][13].



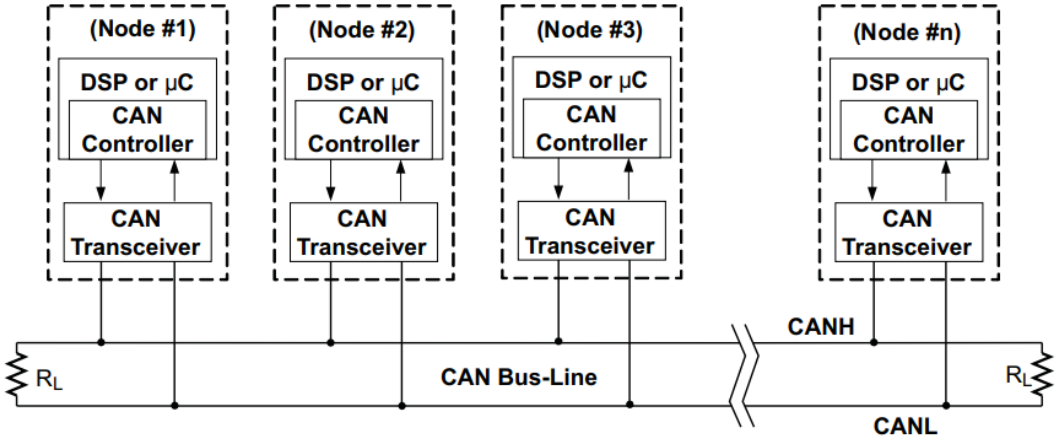
Obrázek 3.4 Zapojení sběrnice se split terminací, převzato z [11]

Takto zapojená sběrnice dokáže filtrovat vysokofrekvenční rušení. Oba rezistory musí mít odpor ~60Ω ± 1 %, aby filtr fungoval správně, typická hodnota kondenzátoru je 4,7 nF [12].

Pro další ochranu sběrnice je možně použít TVS diodu (transil) k ochraně vůči přepěťovým špičkám a elektrostatickým výbojům. Může být také použita tlumivka souhlasného napětí pro potlačení šumu [13].

3.1.3 CAN uzel

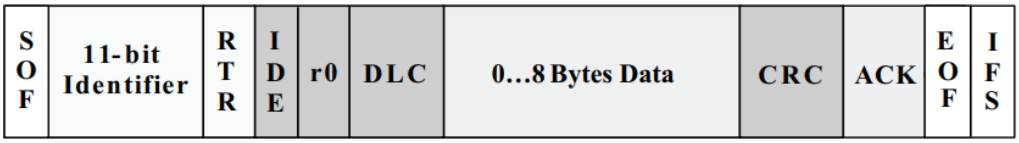
Podle [12] je maximální počet uzlů 64 pro DeviceNet, **127** pro **CANopen**, až až 255 pro CANKingdom. Pokud je více než 30 uzlů na sběrnici, doporučuje se použití transceiverů s vysokou vstupní impedancí. Problém může vznikat, když velké množství uzlů vyšle nebo přijímá zprávy; pokud zařízení začne vysílat zprávu, musí dodat proud do sběrnice, při vysokém počtu uzlů může být sběrnice přetížena a může dojít k vybavení tepelné ochrany nebo k poškození.



Obrázek 3.5 Zapojení sběrnice s detailním popisem uzlů, převzato z [11]

3.2 Linková vrstva CAN (Data-Link Layer)

3.2.1 Struktura datového paketu



Obrázek 3.6 Standartní zpráva, 11bitový identifikátor, převzato z [9]

Na obrázku 3.6 je znázorněna jedna zpráva, rozdělena do bitových polí. První pole **SOF** (Start of Frame) označuje začátek přenášené zprávy, slouží k synchronizaci uzlů na sběrnici po nečinnosti. Má velikost jednoho bitu.

11bitový identifikátor určuje prioritu zprávy, je více vysvětleno v kapitole 3.2.2.

Bit **RTR** (Remote Transition Request) je dominantní, když zprávu požaduje jiný uzel určený identifikátorem.

Dominantní bit **IDE** (IDentifier Extention) označuje, zda je vyslána standartní zpráva s 11bitovým identifikátorem nebo rozšířená zpráva s 29bitovým identifikátorem.

Bit **r0** je rezervován pro budoucí změny standartu.

4bitové pole označeno **DLC** (Data Length Code) obsahuje informaci o počtu bajtů přenášených dat.

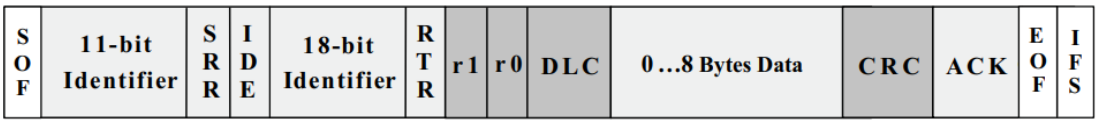
Následují samotná **Data**, do velikosti 8 bajtů.

16bitové pole (15bitů + 1bit delimiter – specifikuje hranici mezi přenášenými daty) **CRC** (Cyclic Redundancy Check) obsahuje kontrolní součet pro detekci chyb.

Pokud všechny uzly na sběrnici přijmou zprávu v pořádku, **ACK** (ACKnowledgement) je přepsán v originální zprávě z recesivní do dominantní, indikující vyslání bezchybné zprávy. Pokud přijímající uzel zjistí chybu a nechá tento bit recesivní, zpráva bude ignorována a vysílací uzel zprávu zopakuje. Tímto způsobem každý uzel potvrdí integritu dat. ACK je dvoubitový, druhý bit funguje jako delimiter.

EOF (End of Frame) o velikosti 7 bitů označuje konec zprávy.

IFS (InterFrame Space) je 7bitová mezera za zprávou; poskytuje dostatek času, aby se zpráva dostala v CAN controlleru dostala do správné pozice v bufferu [9][10].

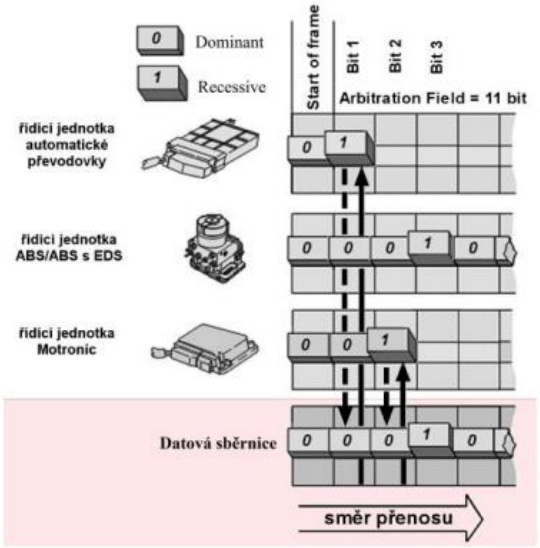


Obrázek 3.7 Rozšířená zpráva, 29bitový identifikátor, převzato z [9]

Bit **SRR** nahrazuje bit RTR a má stejnou funkci jako r0, tedy je rezervován pro budoucí změny, k bitu r0 byl přidán **r1** ze stejného důvodu [9].

3.2.2 Priority zpráv

Uzel začíná svoje vysílání při volné sběrnici. Pokud uzel zahájí vysílání dříve než ostatní uzly, uzly musí čekat do konce vysílání prvního. Při současném vysílání více uzlů má prioritu k přístupu na sběrnici ten uzel, jehož zpráva má nižší identifikátor.



Obrázek 3.8 Příklad priorit zpráv určená identifikátorem, převzato z [13]

Podle výše uvedeného obrázku má nejnižší hodnotu identifikátoru řídící jednotka ABS, jeho zpráva má tedy prioritu a je poslána po sběrnici [13].

3.2.3 Kontrola chyb

Robustnost CAN sběrnice je z velké části zásluhou mechanismy kontrol chyb. Je zabudováno 5 metod kontroly – 3 na úrovni zpráv a 2 na úrovni jednotlivých bitů. Pokud jakákoliv zpráva neprojde kontrolou, není přijata a přijímací uzel vyšle chybovou zprávu (Error frame), což donutí vysílací uzel poslat zprávu znovu, dokud nedojde k jejímu

úspěšnému přijetí. Jestliže vysílací uzel bude stále vysílat chybu a zabírat tak sběrnici, jeho schopnost vysílat je odstraněna [9].

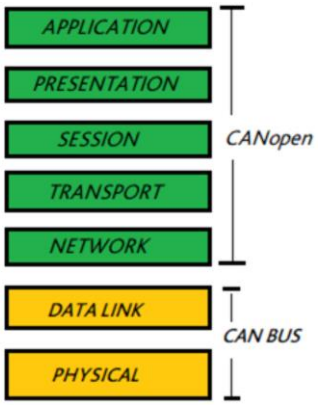
Kontrola chyb na úrovni zpráv je prováděna prostřednictvím CRC a ACK, třetí kontrolou je zjištění, zda v místech, kde mají být recesivní bity, opravdu jsou. Jedná se o bity v SOF, EOF, ACK delimiteru, a CRC delimiteru. Detekcí dominantního bitu dojde k vyslání chyby.

Na bitové úrovni dochází ke čtení každého vyslaného bitu zprávy. Při zjištění nesouladu (monitorováno vysílačem) mezi datovým bitem zapsaným na sběrnici a čteným bitem, dochází k chybě.

Poslední metoda se provádí kontrolou vložených bitů. Po každých 5 bitech stejné úrovně je vložen 1 bit opačné úrovně pro synchronizování časování. Detekcí 6 a více stejných bitů za sebou je generována chyba [9][13].

3.3 CANopen

CANopen je komunikační systém založený na CANu, zjednodušeně řečeno je CANopen komunikační jazyk, kde CAN sběrnice je použita jako přenosové médium. CAN je nízkoúrovňový protokol, který pracuje s fyzickým hardwarem, skládá se z fyzické a linkové vrstvy. CANopen je vysokoúrovňový, definuje 5 vyšších vrstev protokolového modelu následovně:



Obrázek 3.9 Vrstvy CANopen a sběrnice CAN, referenční model OSI, převzato z [14].

Síťová (network) vrstva se stará o síťové adresování a směrování; transportní vrstva poskytuje spolehlivý přenos, stará se o doručení dat. Relační (session) vrstva organizuje a synchronizuje výměnu dat; prezentační přetváří data do tvaru, kterému rozumí další aplikace. Aplikační vrstva popisuje, jak nakonfigurovat a synchronizovat zařízení CANopen [14].

3.3.1 Slovník objektů

Všechny uzly CANopen musejí mít **slovník objektu OD** (Object Dictionary). Jedná se o tabulku, která ukládá data týkající se uzlu a jeho operace – obsahuje záznam pro typ zařízení používaný pro účely identifikace, odečty senzorů nebo procesní stavy (zda je aktuálně vyslán signál nebo zda je aktivní operace zařízení) [15][16].

3.3.2 Komunikace uzlů

Vztah a komunikace mezi uzly určují tři různé architektury – master/slave, klient/server a výrobce/zákazník. U modelu master/slave uzel master požaduje data od uzlu slave a ten také postupuje podle jeho pokynů, slave nemůže požadovat data. Modelem klient/server klienti čtou nebo zapisují data z nebo do serveru. Výrobci v modelu výrobce/zákazník vysílají data do všech ostatních uzlů – spotřebitelů dat výrobce [15].

3.3.3 Objekty technologických dat (PDO)

CANopen protokol je složen z datových objektů. Jedním z nich jsou objekty pro technologická data (Process Data Objects), která jsou nepřetržitě kontrolována při chodu aplikace. Slouží pro přenos veličin jako teplota, napětí, proud, stavy binárních nebo analogových vstupů a výstupů. Každý má svůj identifikátor s vysokou prioritou a může být vyslán pouze jedním uzlem v síti [17].

3.3.4 Objekty servisních dat (SDO)

Informace o konfiguraci, např. počet analogových vstupů a výstupů; konstanty PID regulátorů, rozsahy snímačů jsou přenášeny objekty servisních dat (Service Data Objects). Jsou používány pro konfiguraci zařízení nebo pro přenášení větších zpráv, jejich délka může být libovolně velká, ale mají malou prioritu. Jsou nejčastěji přenášeny na začátku aplikace po spuštění [17].

3.3.5 Objekty pro správu sítě

Objekty pro správu sítě zajišťují správnou konfiguraci a řízení chodu sítě. Skládají se např. z Heartbeat object, Node/Life-guarding object, NMT control object a Emergency object.

Heartbeat object je zpráva odesílaná zařízením o svém stavu; sděluje ostatním zařízením, že pracuje správně. Pokud zpráva nedorazí za určitý čas, reaguje nadřazené zařízení, které vyvolá akci, kterou se podřízené zařízení začne opět hlásit.

Přítomnost podřízených zařízení kontroluje **Node/Life-guarding object**, posílá dotazy a kontroluje odpověď podřízených. Podřízené zařízení může také kontrolovat správnou činnost nadřízených – pokud po určité době nepřijde dotaz na činnost, podřízené zařízení může zareagovat oznámením nadřazenému aplikačnímu programu.

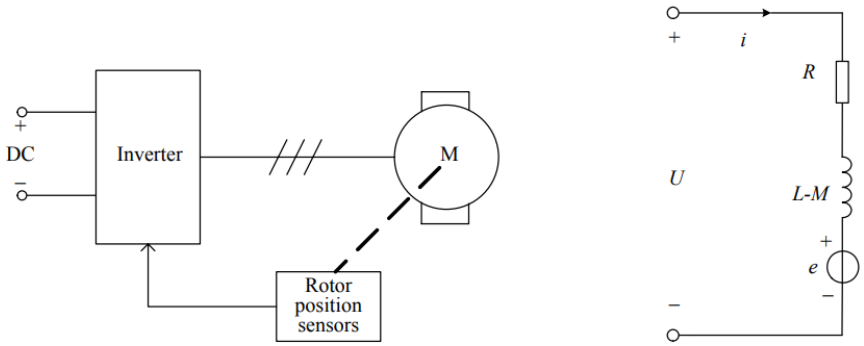
NMT control object poskytuje prostředky pro řízení stavu podřízených v CAN síti.

Emergency object je zpráva vyslaná v případě kritické chyby, nese informaci o typu chyby a chybného zařízení, má vysokou prioritu [17].

Obrázek 3.10 Vnitřní architektura CANopen, převzato z [16].

4. BLDC MOTORY

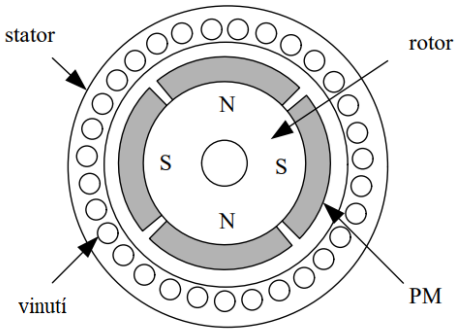
Bezkartáčové stejnosměrné motory – zkratka BLDC (Brushless DC), nebo EC motor (Electronically Commutated), jsou točivé stroje napájené zdrojem stejnosměrného napětí. Jsou charakteristické vysokými otáčkami, vysokým momentem, vysokou účinností a jednoduchou řídící charakteristikou, která je podobná stejnosměrnému motoru s cizím buzením. Od normálních stejnosměrných motorů se liší, jak název napovídá, absencí kartáčů a komutátoru, díky tomu je prodloužena životnost motoru a usnadněna údržba, motor je také tišší. Má permanentní magnety na rotoru, cívky ve statoru. Rotor se otáčí pomocí řídícího měniče, který spíná cívky ve statoru pro vytvoření točivého magnetického pole. Rotor může být buď vnitřní (inrunner), nebo vnější (outrunner). [19][20].



Obrázek 4.1 Topologie a ekvivalentní obvod BLDC motoru, převzato z [21].

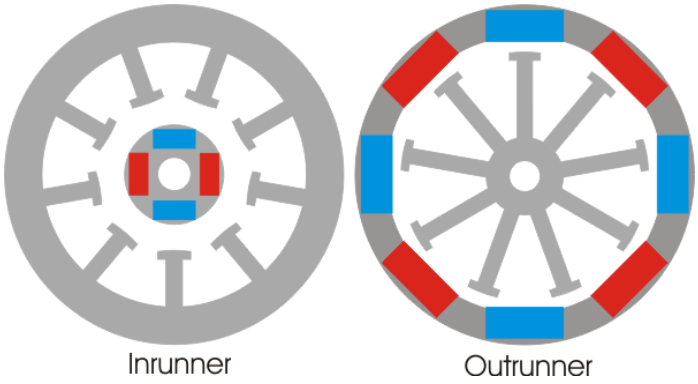
BLDC motory mají široké využití v odvětvích automobilového průmyslu, konkrétně např. pro klimatizaci, stěrače a okna; v leteckém průmyslu, ale i v domácích spotřebičích.

Pro fungování motoru je zapotřebí znát pozici rotoru, aby měl měnič informaci, jakou cívku sepnout, k čemuž se nejčastěji používají Hallovy sondy, ale existují i bezsenzorové metody [19][21].



Obrázek 4.2 Průřez BLDC motorem, převzato z [21].

Konstrukce statoru je podobná střídavým synchronním a asynchronním motorům.

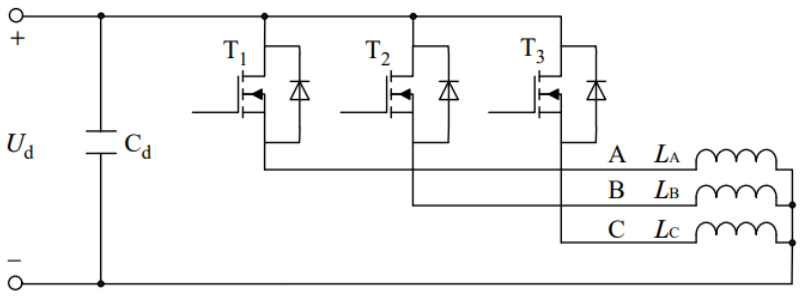


Obrázek 4.3 Průřez BLDC motorem, inrunner vs outrunner, převzato z [24].

4.1 Řízení BLDC motoru

Jako spínače k řízení otáčení se používají tranzistory MOSFET a IGBT, případně modul IPM (Intelligent Power Module), který má v sobě integrované IGBT a obvody s funkcemi zpracování signálů, ochrany a diagnózy. Výhodou IPM jsou malé rozměry, nízká hmotnost a vysoká spolehlivost a jsou tak ideálními zařízeními pro ovládání BLDC motorů [21]. Ke spínání cívek pomocí tranzistorů se používá několik zapojení:

4.1.1 Poloviční můstkové zapojení



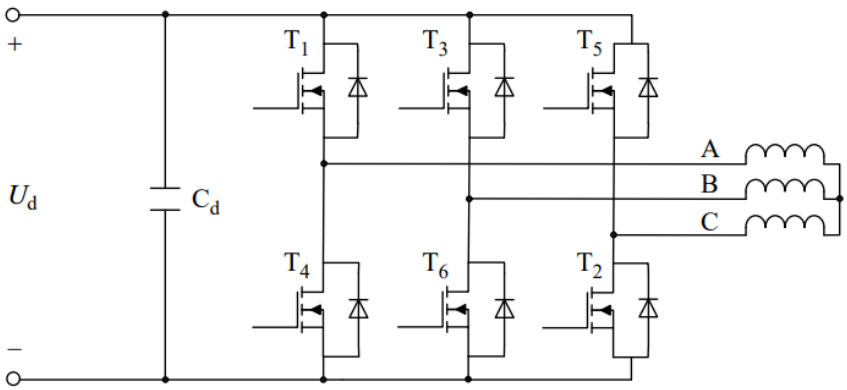
Obrázek 4.4 Schéma půlmůstkového řídícího obvodu, vinutí spojeno do hvězdy, převzato z [21]

Spínací tranzistory T1, T2 a T3 jsou připojeny k příslušným vinutím A, B a C. Tranzistory jsou spínány podle signálů z hallových sond.

Během komutace má točivé magnetické generované statorovým vinutím tři magnetické stavy ve 360° elektrických, každý stav má 120° elektrických.

Ačkoliv má poloviční můstek výhody jako malé množství komponentů, nízkou cenu a jednoduchost řízení, je využit výjimečně kvůli velkému zvlnění točivého momentu (tzv. torque ripple) a malému využití vinutí, v tomto případě prochází proud jednou cívkou pouze 1/3 periody [21].

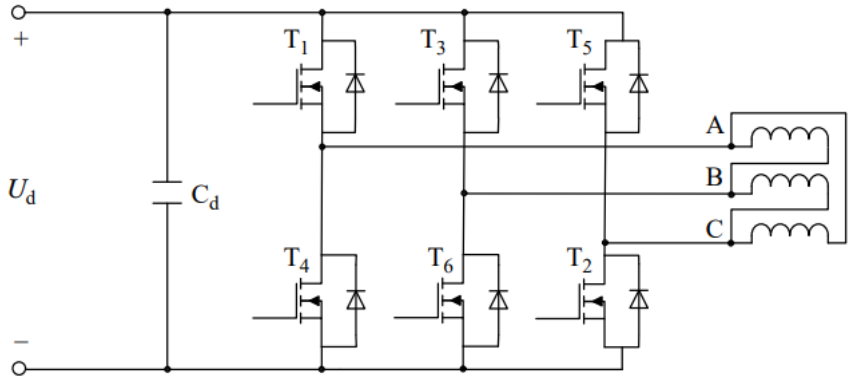
4.1.2 Můstkové zapojení



Obrázek 4.5 Schéma můstkového řídícího obvodu, vinutí spojeno do hvězdy, převzato z [21]

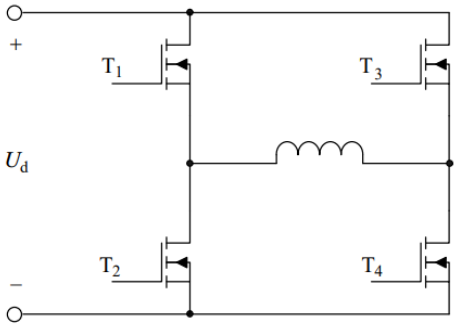
Takto zapojený měnič lze řídit dvěma způsoby:

- **Dvoufázové řízení**
  - V tomto případě můstkový měnič komutuje jednou za 60° el., tedy celkově šest magnetických stavu, proud vždy vedou dvě vinutí najednou. V jednu chvíli vždy vede jeden spínač z horní řady a jeden z dolní. Horní posílají proud do korespondujícího vinutí vpřed, což vytváří moment. Další moment je vytvářen záporným proudem dolním spínačem do jiného vinutí. Suma těchto momentů otočí rotor o 60°.
- **Třífázové řízení**
  - V jednu chvíli vedou proud vždy tři spínače. Každý spínač ve spínacím cyklu vede proud 180°. Takovéto řízení zvyšuje využití každé fáze a snižuje kmitání momentu [21].



Obrázek 4.6 Schéma můstkového řídícího obvodu, vinutí spojeno do trojúhelníka, převzato z [21]

4.1.3 H-můstek



Obrázek 4.7 Schéma H-Můstku, převzato z [21]

Každé vinutí je ovládáno samostatně, každému náleží čtyři spínače; většinou se používá pro jedno a dvoufázové motory. Ovládání proudu je v tomto zapojení jednoduché a zajišťuje čtyřkvadrantové řízení. Je ale nutné zajistit zpoždění mezi horními a dolními spínači stejného ramene měniče, aby proud netekl ve stejnou chvíli oběma [21].

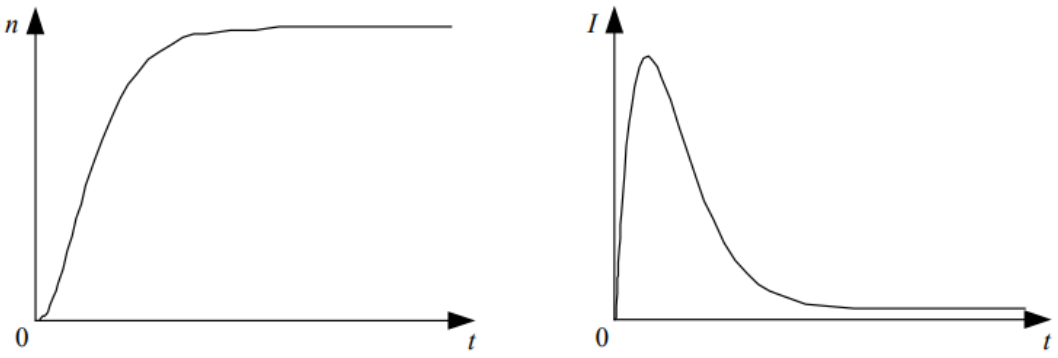
4.2 Charakteristiky BLDC motoru

4.2.1 Rozběhové charakteristiky

V okamžiku rozběhu motoru má motor nulové otáčky, proud vinutí je dán vztahem

$$I = \frac{U_d - \Delta U}{R_a} \qquad (4.1)\ [21]$$

kde I představuje proud vinutím, U_d napětí zdroje, ΔU úbytek napětí na spínačích měniče a R_a odpor vinutí. Charakteristiky otáček a proudu vinutí jsou na obrázku 4.7.



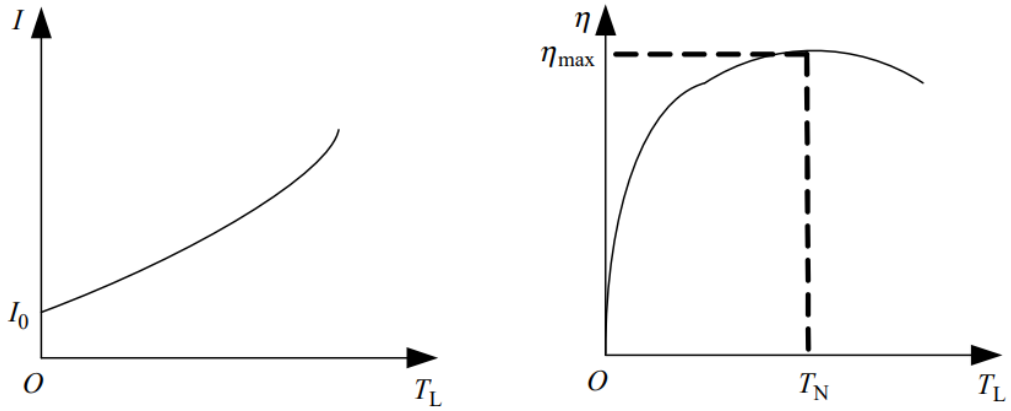
Obrázek 4.8 Závislost otáček a proudu v čase, převzato z [21]

Rozběhový proud v počátku prudce narůstá. Protože úbytky napětí na spínačích a na vinutí jsou velmi malé, proud může dosahovat více než desetinásobku jmenovitého. Částečně je ale tento proudový náraz do určité míry užitečný, jelikož umožňuje rychlé

roztočení na jmenovité otáčky i při plném zatížení. Proud začíná klesat vlivem působící zpětné elektromagnetické síly (dle Lenzova zákona), což snižuje moment a zrychlení motoru. Po dosažení ustáleného stavu otáčky motoru zůstávají konstantní.

Proudová špička představuje problém pro spínače v měniči, tranzistory jsou velmi citlivé na nadproudy (např. IGBT tranzistory dokážou vydržet tento nadproud méně než 10 μs), je nutné dimenzovat spínače na hodnotu rozběhového proudu, což ale zvyšuje cenu [21]

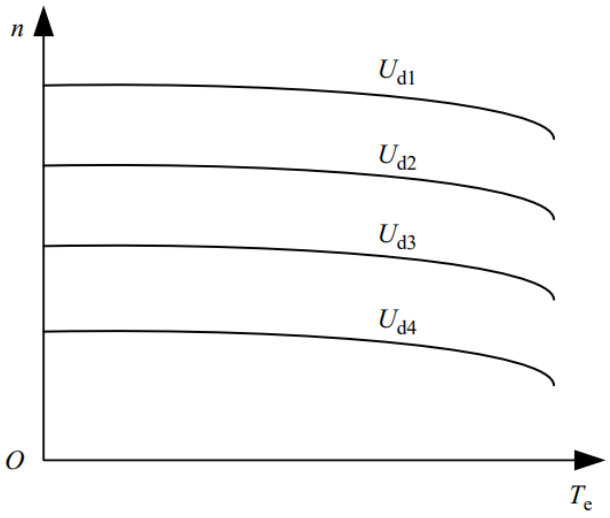
4.2.2 Provozní charakteristiky



Obrázek 4.9 Závislost proudu vinutí a účinnosti na zátěžném momentu T_L při konstantním napětí U_d, převzato z [21]

Ztráty motoru lze rozdělit do dvou částí – proměnné a stálé. Do stálých se řadí ztráty na spínačích a ztráty naprázdno (ztráty v železe a třením), do proměnných patří ztráty v mědi vinutí, protože odpor se mění se zatížením (stoupá teplota, a tedy i odpor mědi). Motor dosahuje nejvyšší účinnosti, když se stálá a proměnná složka ztrát rovnají [21].

4.2.3 Mechanická charakteristika



Obrázek 4.10 Závislost otáček na elektromagnetickém momentu T_e, $U_{d1} > U_{d2} > U_{d3} > U_{d4}$, převzato z [21]

Tyto charakteristiky podobné charakteristikám stejnosměrného motoru s cizím buzením [21].

5. POUŽITÁ ZAŘÍZENÍ

5.1 Použitý motor

Motor použitý v této bakalářské práci je třífázový, osmipólový BLDC motor firmy Berger Positec model 42BLF03 s vinutím zapojeným do hvězdy [26].

Obrázek 5.1 Použitý BLDC motor, převzato z [26]

Tabulka 5.1 Technické specifikace BLDC motoru [26]

Popis	**Jednotka**	**Hodnota**
Jmenovité napětí	V	24
Jmenovité otáčky	1/min	4000
Jmenovitý moment	Nm	0,188
Jmenovitý proud	A	5,7
Výstupní výkon	W	78
Špičkový moment*	Nm	0,75
Špičkový proud	A	18
Momentová konstanta	Nm/A	0,036

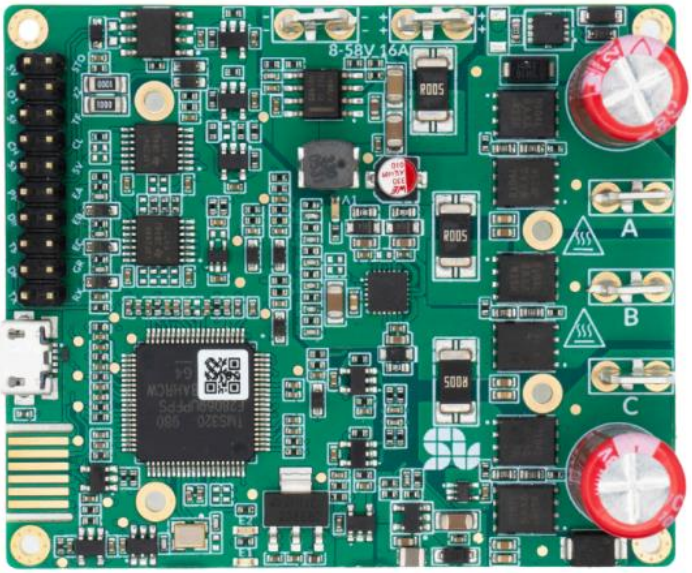
**špičkový moment je uváděn pouze pro účely výpočtu, provoz motoru na této hodnotě může způsobit poškození motoru*

5.2 Použitý měnič

Jedná se o měnič SOLO PICO schopný ovládat kartáčové a bezkartáčové stejnosměrné motory a synchronní motory s permanentními magnety. Jeho schopnostmi jsou:

- Ovládání momentu, rychlosti a pozice
- Čtyřkvadrantová operace
- Podpora rozhraní USB, UART, CANopen
- Podpora teplotních senzorů PT1000
- …

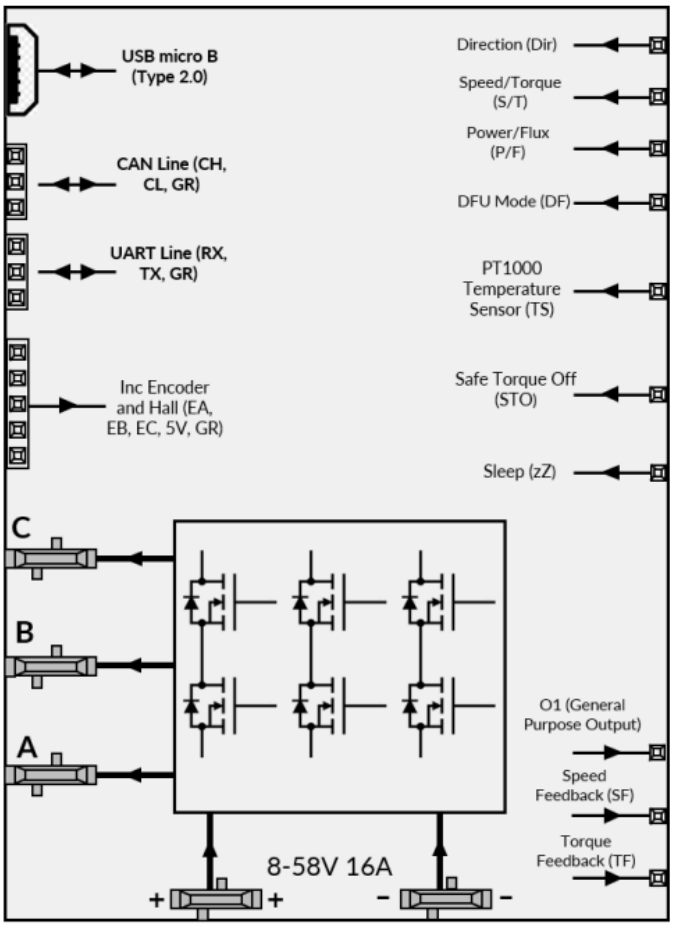
Měnič nachází využití v průmyslové automatizaci, robotice, dronech, v automobilovém průmyslu, zdravotnických přístrojích, letectví, textilním průmyslu, AGV (Automated Guided Vehicle – robot schopný jet podle čar na podlaze, nebo využívá kamer, rádiových vln, magnetů nebo laserů k navigaci. Je využíván k transportu těžkých objektů uvnitř průmyslových budov [23]), pro vytápění a ventilaci [22].



Obrázek 5.2 Použitý SOLO PICO měnič, převzato z [22]

Tabulka 5.2 Technické specifikace řídícího měniče [22]

Popis	**Jednotka**	**Hodnota**
Rozmezí napětí DC zdroje	V	8 až 58
Softwarové přepěťové aktivační napětí	V	60
Podpěťová hranice DC sběrnice	V	7
Maximální stálý výstupní stejnosměrný proud	A	16
Maximální stálý výstupní střídavý proud	$A_{(RMS)}$	11.5
Maximální stálý výstupní výkon	W	300
Špičkový výstupní výkon	W	450
Kapacita vnitřní sběrnice	µF	440
Spínací frekvence (výstupní PWM frekvence)	kHz	8 až 80
Rozmezí teplot desky	°C	-20 až +120



Obrázek 5.3 Blokový diagram měniče, převzato z [22]

5.2.1 Měnič a CANopen

Všechny funkce podporovány protokolem CANopen v měniči SOLO jsou rozděleny do objektů, každý objekt může odkazovat na specifický úkon nebo funkci ovladače (aktuální otáčky, proudový limit, pozice, …). Ovladač má specifický objekt pro každý parametr k uložení a použití, přístup k objektu se liší podle toho, k čemu je objekt používán. Objekty mohou být ke čtení, zapisovatelné, nebo obojí. Každý objekt je přístupný 16bitovou adresou nazývanou object index, některé objekty mají 8bitový subindex. Čtení a psaní do specifických objektů se koná CANopen zprávami. SOLO podporuje dva druhy zpráv:

- NMT Control Objects 1000h – 1FFFh: objekty správy sítě a funkce jako Node Guarding, Lifeguarding, Heartbeat atd.
- Objekty specifikované výrobcem 3000h – 5FFFh: objekty k nastavení a ovládání měniče (viz. kapitola 6) [25]

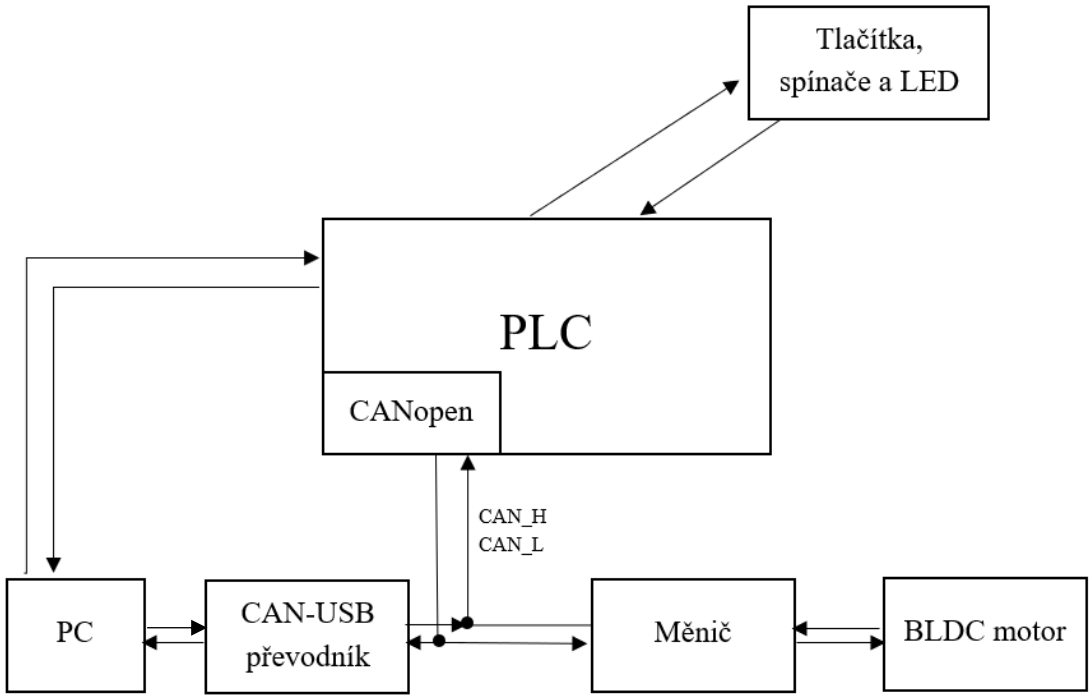
5.3 Použité PLC

Modicon M241 od firmy Schneider Electric napájený z 24 V DC zdroje. Je vybavený 8 rychlými vstupy (zpracovává signály do frekvence 200 kHz) a 6 normálními vstupy (do 1 kHz), 4 rychlými výstupy a 6 normálními výstupy, ethernetovým portem, portem pro CANopen, sériovou linku a USB portem pro programování. Disponuje také vestavěnými funkcemi pulsně šířkové modulace PWM, frekvenčním generátorem a generátorem impulsů. Umožňuje vložení paměťové karty 256 MB pro zálohu a přenos aplikací, zápis dat a aktualizaci firmware.

Obrázek 5.4 PLC Modicon M241, převzato z [18].

Logické kontroléry Modicon M241 jsou vyvinuty pro stroje pracující s vysokými rychlostmi a s kontrolou pozic. Ethernetový port nabízí FTP (File Tranfer Protocol) a SQL client, díky čemuž mohou být snadno integrovány do ovládacích systémů s dálkovým ovládáním a monitoringem [18].

6. OVLÁDÁNÍ POHONU

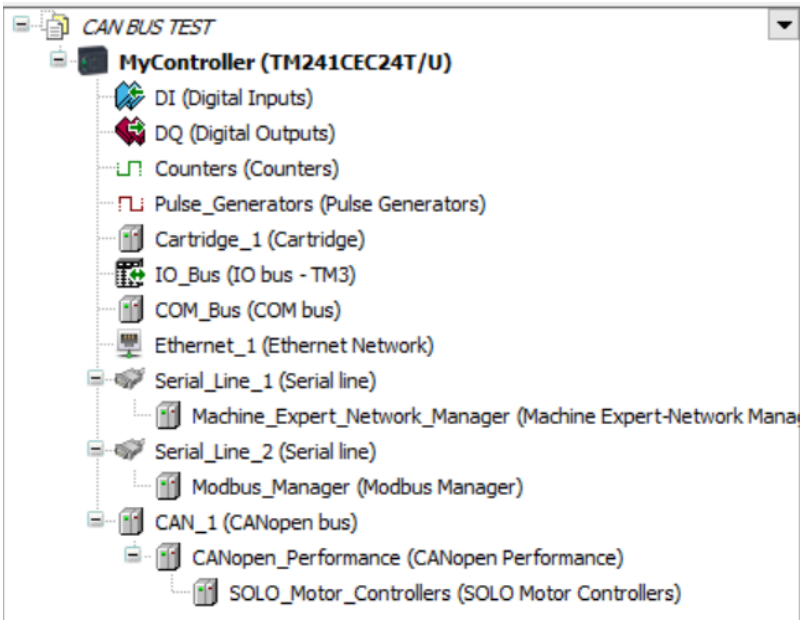


Obrázek 6.1 Blokové schéma zapojení

Výše uvedené schéma je reprezentací reálného zapojení. PLC je srdcem celého pohonu, na něj jsou připojené binární vstupy a výstupy ve formě tlačítek, spínačů a LED, kterými je program ovládán, LED byly využity k signalizaci. Z CANopen portu vystupují vodiče CAN_H a CAN_L do měniče přes CAN-USB převodník a k němu je připojený počítač, který monitoruje příchozí a odchozí zprávy do a z měniče (respektive do a z PLC). CANopen zprávy přichází do měniče, ten je zpracovává a podle nich ovládá BLDC motor. Ten posílá zpětnou vazbu, např. aktuální hodnotu otáček, aktuální pozici, do měniče a ten ji dokáže poslat do PLC (v programu toto ale z časových důvodů nebylo implementováno). PC je přes USB připojen k samotnému PLC pro nahrání programu a kontrolu jeho aktuálního stavu. Měnič i PLC jsou napájeny z 24VDC zdroje.

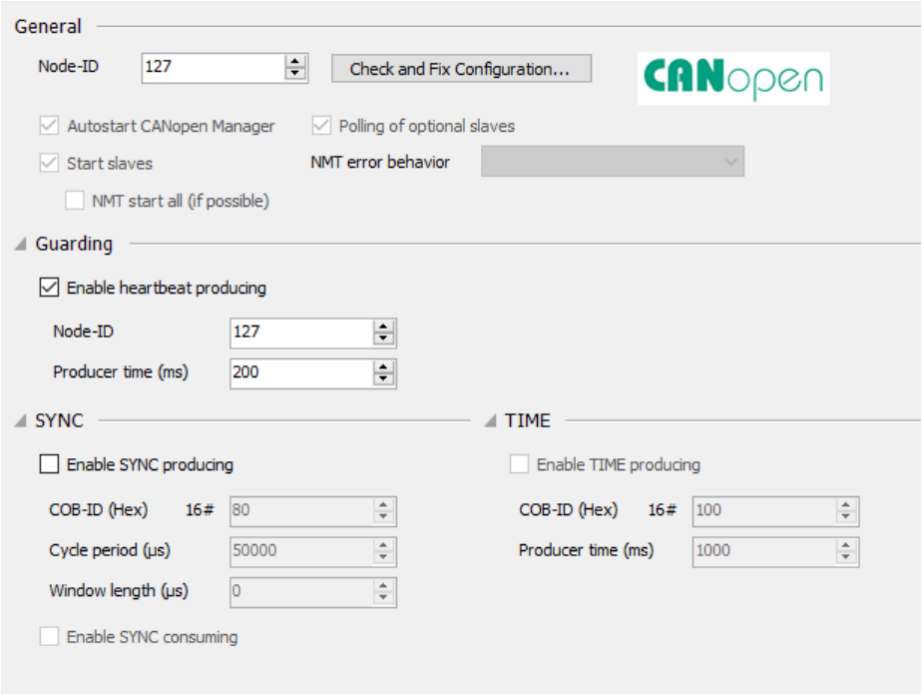
6.1 1. metoda – pomocí .eds souboru pro SOLO měnič

První vyzkoušený postup pro ovládání motoru bylo nahrání .eds souboru (.eds – Electronic Data Sheet) do programu Schneider Electric Machine Expert Logic Builder (dále uváděno pouze jako Machine Expert). Po založení projektu byl do stromu zařízení přidán CANopen_Performance modul, který spravuje CANopen zařízení, a pod něj v hierarchii nově přidaný modul SOLO_Motor_Controllers následovně:



Obrázek 6.2 Strom zařízení v Machine Expert po nahrání .eds souboru a přidání modulů pod CAN_1.

CANopen_Performance umožňuje nastavení Heartbeat a SYNC, po jeho rozkliknutí vypadá rozhraní podle následujícího obrázku.



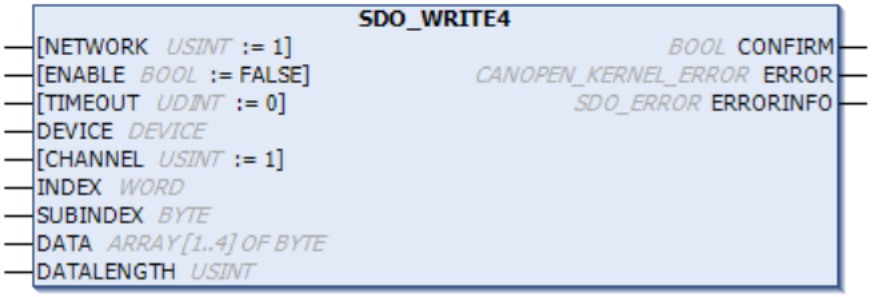
Obrázek 6.3 Možnosti nastavení CANopen_Performance modulu.

Velmi podobně vypadá i modul SOLO_Motor_Controllers, ten ale ještě umožňuje Nodeguarding, Emergency a kontrolu ID výrobce, produktové číslo a revision number.

Zde ale dochází k problému, jelikož původní premisa byla, že nahráním .eds souboru dojde k vytvoření funkčního bloku k ovládání motoru, což se nestalo. Další možnost byla, že program už obsahuje bloky k ovládání, to ale také nebylo pravdou. Tento postup byl tak opuštěn a přešlo se k nové metodě.

6.2 2. metoda – přímo z dostupných bloků Machine Expert

V knihovně bloku byla nalezena knihovna CIA405, která obsahovala řadu bloků pro CANopen, mezi nimi i blok SDO_WRITE4. Stejně jako u první metody je potřeba přidat do stromu zařízení CANopen_Performance modul, jinak blok nebude fungovat.



Obrázek 6.4 SDO_WRITE4 blok z knihovny CIA405

Tabulka 6.1 Popis vstupů a výstupů bloku SDO_WRITE4

Název	**Typ**	**Popis**
Network	USINT	CAN síť na které má funkční blok fungovat
Enable	BOOL	Spustí blok na vzestupné hraně, ukončí operaci na sestupné
Timeout	UDINT	Timeout v ms, 0 = bez timeoutu
Confirm	BOOL	True = blok dokončil operaci bez chyb
Error	CANOPEN_KERNEL_ERROR	Poskytuje detalní informace ohledně chyby
Device	DEVICE	NodeID cílového zařízení
Channel	USINT	SDO kanál, 0 = jakýkoliv volný kanál
Index	WORD	Index objektu
Subindex	BYTE	Subindex objektu
Data	ARRAY [1..4] OF BYTE	Data k zapsání
Datalenght	USINT	Počet bytů k zapsání
Errorinfo	SDO_ERROR	Obsahuje kód chyby

Tento blok umožňuje odeslání zprávy dle zadaných parametrů. Pro roztočení motoru na 1000 min^{-1} je nutné odeslat zprávu:

601	8	22 05 30 00 E8 03 00 00	Set the Speed reference at 1000 RPM

Obrázek 6.5 Zpráva, která roztočí motor na 1000 min^{-1}, převzato z [25].

Tato zpráva se skládá z několika částí:

COB-ID	DLC	Byte 1	Byte 2	Byte 3	Byte 4	Byte 5	Byte 6	Byte 7	Byte 8
SDO identifier	Data length	Command	Index L	Index H	Sub-index	Data 0	Data 1	Data 2	Data 3

Obrázek 6.6 Složení zprávy, převzato z [27].

COB-ID je definován:

- klient -> server: 600h + node ID
- server -> klient: 580h + node ID

DLC označuje délku dat v bytech od 0 do 8.

Veškeré zprávy jsou ve formátu little endian, data jsou rozděleny do jednotlivých bytů a ten nejméně důležitý je poslán první, např. číslo 0x87963510 je posláno jako 0x10 0x35 0x96 0x87. To je podstatné pro indexy v bytech 2 a 3. Dle manuálu má příkaz na otáčky index 0x3005, ve zprávě je tak napsáno 0x05 0x30 [25].

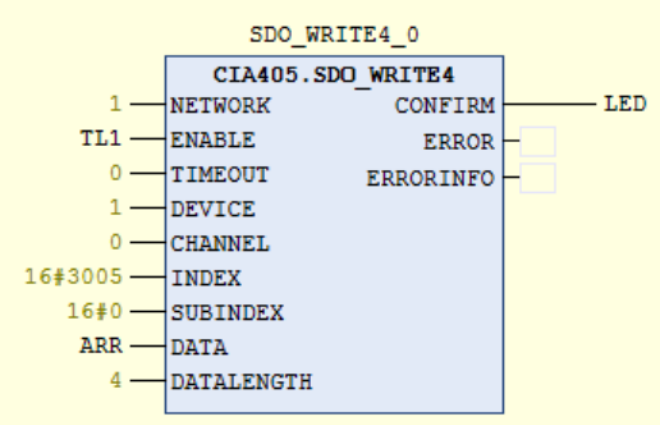
První byte odkazuje na délku dat dle následujícího obrázku:

Command	Description	User data	Function
22h	SDO(rx), Download Request	undefined	Send parameters to sensor
23h		4 bytes	
2Bh		2 bytes	
2Fh		1 bytes	
60h	SDO(tx), Download Response	-	Confirm parameter transfer to client
40h	SDO(rx), Upload Request	-	Request parameters from sensor
42h	SDO(rx), Upload Response	undefined	Send parameters to client
43h		4 bytes	
4Bh		2 bytes	
4Fh		1 bytes	
80h	SDO(tx), Abort Domain Transfer	4 bytes	Sensor sends error code to client

Obrázek 6.7 Složení prvního bytu zprávy, převzato z [27]

Naše zpráva k roztočení na 1000 otáček má tedy nedefinovanou délku zprávy. Tato zpráva nemá subindex, na jeho pozici je 0x00. Poté následují data, což jsou otáčky motoru v hexadecimálním tvaru (0x00 0x00 0x03 0xE8 = 1000 dec.).

S touto znalostí nyní můžeme vytvořit program s blokem SDO_WRITE4.



Obrázek 6.8 Použití bloku SDO_WRITE4

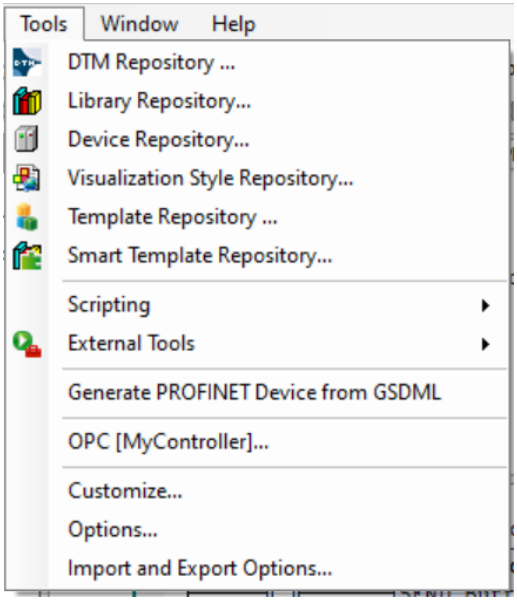
Na ENABLE bylo přivedeno tlačítko TL1, aby po jeho zmáčknutí byla poslána tížená zpráva, na CONFIRM LED, která měla signalizovat úspěšné vyslání zprávy bez timeoutu na jakýkoliv volný kanál. Index 16#3005 je 0x3005. DATA je pole bytů proměnné ARR.

```
ARR: ARRAY [1..4] OF BYTE := [16#E8, 16#03, 16#00, 16#00];
```

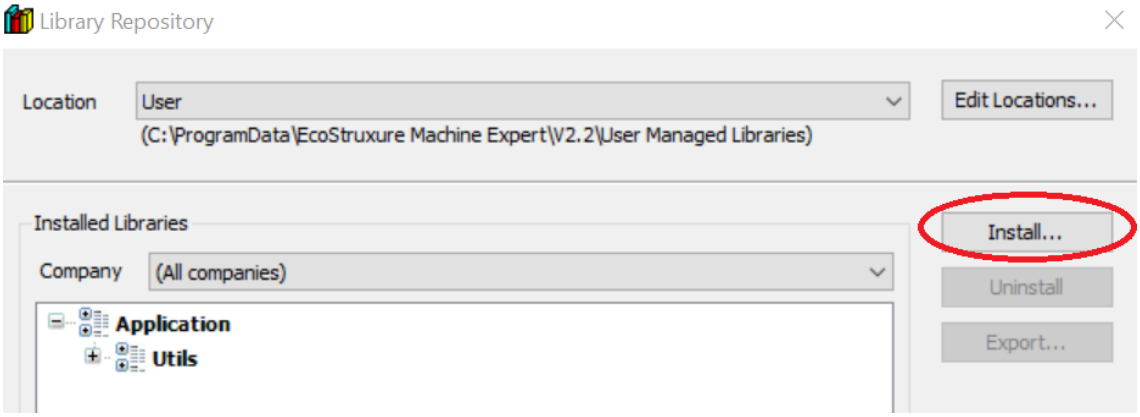
U DATALENGTH dochází ale k problému, jelikož naše zpráva má nedefinovanou délku dat, vstup by tak neměl být definován. Bohužel blok ale definici délky potřebuje, pokud bychom vstup nechali prázdný, použije poslední zapamatovanou hodnotu vstupu, v tomto případě 4. Zprávu s nedefinovanou délkou nedokáže odeslat. Takto nastavený blok odeslal zprávu ***601h 8 23 05 30 00 E8 03 00 00***, na což měnič odpověděl ***581h 8 80 05 30 00 00 00 00 00***, což je podle obrázku 6.7 kód chyby. Pokud bychom nastavili DATALENGTH na 0, zpráva má prázdné byty DATA, motor se neroztočí. Jakákoliv jiná délka zprávy má podle obrázku 6.7 také jiný první byte Command, motor na ní odpovídá chybou. Tato metoda byla tak opuštěna a bylo přistoupeno na další.

6.3 3. metoda – vytvořením vlastního řídícího bloku

Na internetu na stránkách webu CODESYS byla nalezena knihovna CANbus Example, která poskytuje funkce potřebné k vytvoření vlastního funkčního bloku. Stažená knihovna je ve formátu souboru .package, kterou je nutné rozbalit programem 7zip. Po rozbalení je cesta k potřebné knihovně – Component\Library. Knihovnu lze nainstalovat do Library Repository v sekci Tools v horní liště programu.

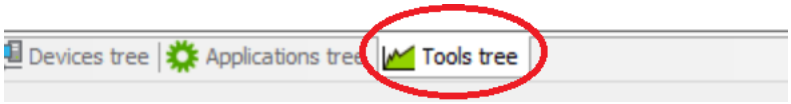


Obrázek 6.9 Rozhraní Tools v programu Machine Expert

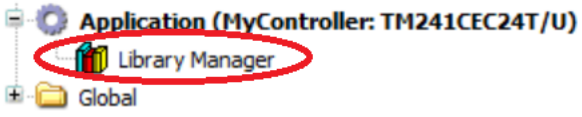


Obrázek 6.10 Instalace knihovny

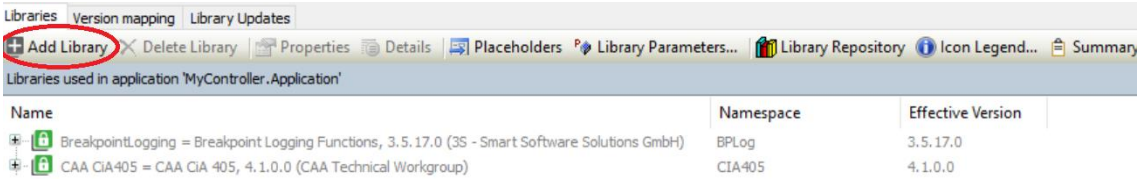
Po instalaci knihovny je nutné ji přidat do Library Manager ve stromu nástrojů (Tools Tree) v levé dolní části obrazovky.



Obrázek 6.11 Umístění stromu nástrojů



Obrázek 6.12 Umístění Library Manager



Obrázek 6.13 Přidání knihovny do Library Manager

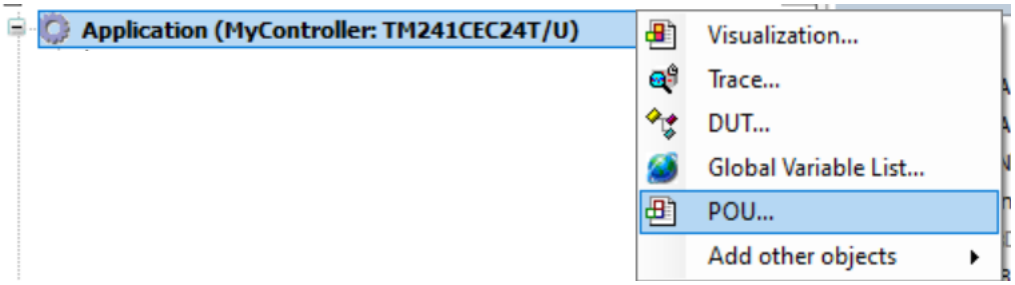
Zde je knihovna umístěna pod Application\Utils, případně jí lze najít vyhledáním výrazu „CAN API“

Obrázek 6.14 Hledání knihovny

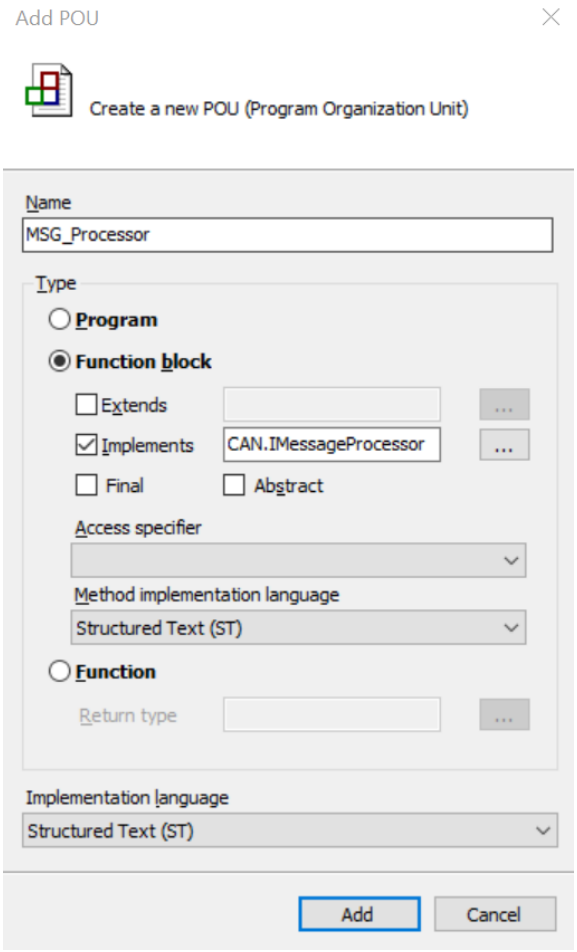
Po zakliknutí knihovny a tlačítka „OK“ bude knihovna přidána. Nyní můžeme vytvořit funkční blok použitím funkcí této knihovny.

Stejně jako u předchozích metod je zapotřebí přidání modulu CANopen_Performance do stromu zařízení.

Vytvoříme POU typu Program do stromu aplikací v jazyku dle preference, který zatím necháme prázdný (v našem programu pojmenováno „Control_PRG“). Vytvoříme další POU typu Function Block, který bude využívat naši přidanou knihovnu následovně:



Obrázek 6.15 Vkládání objektů pod Application.



Obrázek 6.16 POU MSG_Processor implementující funkce knihovny CANbus Example.

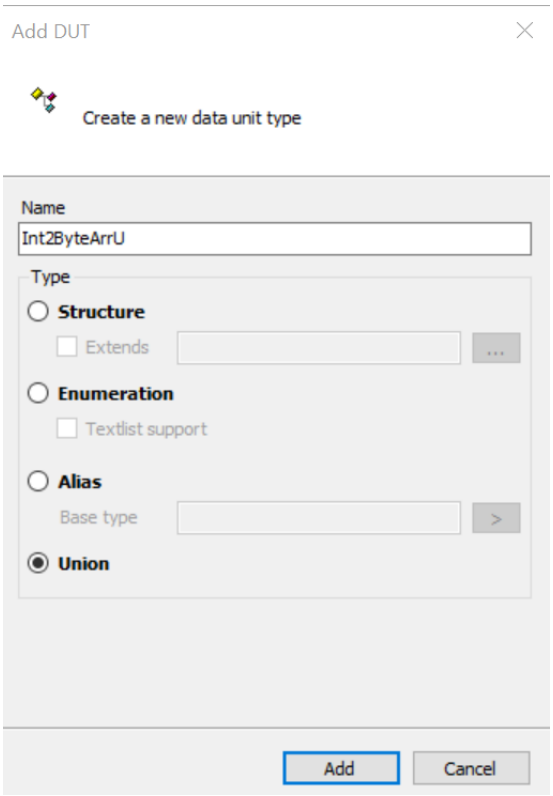
Rozklikneme vytvořenou metodu pod novým POU MSG_Processor a vymažeme řádek:

```
{warning 'Add method implementation '}
```

Jako další vytvoříme POU typu Function Block v jazyku ST, v našem případě s názvem „RPM_Block“, ale nezaškrtáváme kolonku Implements. Nyní definujeme proměnné do bloku:

```
FUNCTION_BLOCK RPM_Block
VAR_INPUT
      SEND_Button:       BOOL;
      RPM:               INT;
END_VAR
VAR_OUTPUT
END_VAR
VAR
      driverCAN11bit:   CAN.CANBus_11bit;
      stCANbusConfig:   CAN.DRIVER_CONFIG   :=   (uiBaudrate   :=   250,
ctMessages := 10);
      stMessageSend:    can.RxMESSAGE;
      R_trig1:          R_trig;
      SEND :            BOOL;
      Int2Byte:         Int2ByteArrU;
END_VAR
```

SEND_Button je vstupní tlačítko, RPM je vstupní číselná hodnota požadovaných otáček. Proměnná driverCAN11bit je typu CAN.CANBus_11bit, který přidala knihovna, stejně jako CAN.DRIVER_CONFIG, kde byl definován Baudrate a ctMessages, což je dle popisu knihovny délka fronty zpráv pro odchozí zprávy, a can.RxMESSAGE, R_Trig1 je typu R_trig, tedy reakce na vzestupnou hranu signálu, SEND je výstupem R_Trig1, jak je uvedeno v dalším kódu. Int2Byte je nově vytvořeného typu Int2ByteArrU. Tento typ byl vytvořen jako nový DUT typu UNION.



Obrázek 6.17 Vytváření DUT typu UNION

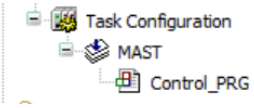
Jeho kód je:

```
TYPE Int2ByteArrU :
UNION
   i: INT;
   bytes: ARRAY [0..2] OF BYTE;
END_UNION
END_TYPE
```

Toto nám umožňuje přeměnit vstupní INT hodnoty otáček do pole 2 bytů, které jsou vloženy do vyslané zprávy dle uvedeného kódu funkčního bloku „RPM Block“.

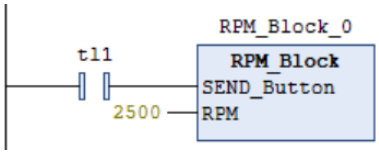
```
Int2byte.i := RPM;

driverCAN11bit(DriverConfig:=stCANbusConfig, eError =>);


stMessageSend.udiCanID := 16#601;
stMessageSend.xIs29BitMessage := FALSE;
stMessageSend.usiDataLength := 16#8;
stMessageSend.abyData [0] := 16#22;
stMessageSend.abyData [1] := 16#05;
stMessageSend.abyData [2] := 16#30;
stMessageSend.abyData [3] := 16#00;
stMessageSend.abyData [4] := int2byte.bytes [0];
stMessageSend.abyData [5] := int2byte.bytes [1];
stMessageSend.abyData [6] := 16#00;
stMessageSend.abyData [7] := 16#00;

R_trig1 (CLK:= SEND_Button, Q=> SEND);

IF SEND THEN
driverCAN11bit.SendMessage (Message:=stMessageSend);
END_IF
```

Zde je byte po bytu složena zpráva k odeslání otáček, otáčky je možné přímo zadat proměnou RPM, nejsme limitováni pouze na 1000; byty na pozici 4 a 5 jsou přivedeny pomocí nového DUT. Vzestupná hrana vstupu SEND_Button odešle zprávu na sběrnici. Nyní je nutné tento blok implementovat do POU typu Program, který jsme vytvořili na začátku a nechali prázdný. Ten je ale nejprve nutné zavolat jeho přesunutím do Tasku, což lze pouhým přetáhnutím programu ze stromu myší.



Obrázek 6.18 Program Control_PRG volán Taskem MAST

Do programu vložíme náš vytvořený funkční blok a přivedeme na něj vstupy:



Obrázek 6.19 Implementace nově vytvořeného funkčního bloku RPM_Block

Tlačítkem tl1 odesíláme zprávu na roztočení motoru dle zadaných otáček, při RPM = 0 se motor zastaví. RPM nemusí být takto zadaná hodnota, může být např. připojeno rozhraní, kde se otáčky zadají na grafickém panelu do proměnné nebo pomocí analogového vstupu.

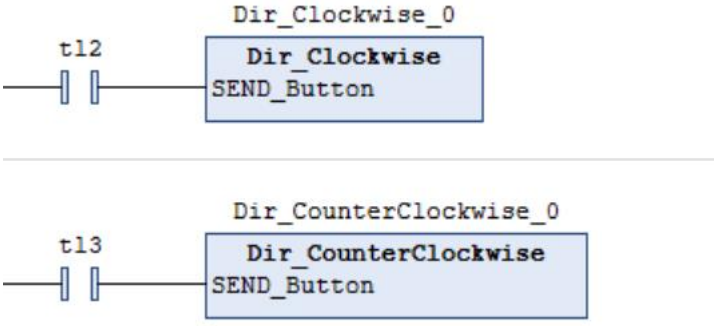
Měnič ale dokáže nejenom ovládat otáčky, ale má spoustu dalších funkcí, mezi které se řadí i směr otáčení. Podle manuálu [25] je index zprávy pro směr otáčení 0x300C, číslo pro směr otáčení proti směru hodinových ručiček je 0x00000000, po směru je 0x00000001. čemuž odpovídají zprávy ***601h 8 22 0C 30 00 00 00 00***, respektive ***601h 8 22 0C 30 00 01 00 00 00***. Stejným postupem lze tak vytvořit další funkční bloky se stejným kódem, pouze na místech bytů 4 a 5 budou tentokrát přímo konkrétní čísla. Proti směru hodinových ručiček:

```
stMessageSend.abyData [4] := 16#00;
stMessageSend.abyData [5] := 16#00;
```

A po směru hodinových ručiček:

```
stMessageSend.abyData [4] := 16#01;
stMessageSend.abyData [5] := 16#00;
```

Tyto nové bloky se použijí v programu následovně:



Obrázek 6.20 Implementace funkčních bloků pro směr otáčení.

Zmáčknutím tlačítka tl2 se motor otáčí podle směru hodinových ručiček, tlačítkem tl3 proti.

V manuálu k měniči je uvedena řada dalších funkcí, které lze měničem řídit, případně i číst v rámci zpětné vazby do PLC.

ZÁVĚR

Proniknutím sběrnice CAN a protokolu CANopen do automatizace, i díky sdružení CIA, se otevírají další možnosti komunikace mezi výrobními jednotkami, akčními členy a řídícími zařízeními. Tato práce o těchto možnostech pojednává na konkrétním příkladu – řízení CANopen pohonu z PLC. Touto prací bylo dosaženo bodů zadání:

1. **Seznamte se s dodaným BLDC pohonem.**
   Dodaný BLDC pohon 42BLF03 byl popsán v 5. kapitole, stejně jako řídící měnič. Vysvětlení vlastností motoru, jeho charakteristik a řízení je obsaženo v kapitole 4.
2. **Nastudujte protokol CANopen**
   Tomu byla věnována především kapitola 3, kde je v krátkosti pojednáno o historii a vývoji sběrnice CAN a protokolu CANopen, jsou vysvětleny jednotlivé vrstvy fyzické sběrnice CAN – fyzická a linková a vrstvy CANopen – síťová, transportní, relační, prezentační a aplikační, jsou popsány druhy dat, které se mohou na sběrnici objevit. V kapitole 6. je popsána struktura zpráv.
3. **Navrhněte program do PLC Modicon M241**
   Popis a parametry dodaného PLC Modicon M241 jsou nastíněny v kapitole 5, samotný vývoj programu v kapitole 6. Bylo otestováno několik možných postupů, z nichž jeden byl úspěšný. Postup vytváření funkčního programu je krok po kroku uveden a celý program je v příloze.

Byla popsána základní charakteristika PLC a příklady jejich programování, jsou uvedeny jednotlivé programovací jazyky.

Všechny body zadání byly splněny.

LITERATURA

[1] *Programmable Logic Controllers*. 4th ed. Oxford (UK): Elsevier Newnes, 2006. ISBN 978-0-7506-8112-4.

[2] *PLC a automatizace. 1. díl, Základní pojmy, úvod do programování*. Praha: BEN - technická literatura, 2007. ISBN 978-80-86056-58-6.

[3] *Prostředky průmyslové automatizace*. Brno: VUTIUM, 2004. ISBN 80-214-2610-1.

[4] TECHNICKÁ NORMALIZAČNÍ KOMISE. IEC 61131-3:2013, *Programovatelné řídící jednotky - Část 3: Programovací Jazyky*. 3. vydání.

[5] *An Overview of Sequential Function Chart (SFC) PLC Programming*. Online. Control Automation. 2022. Dostupné z: https://control.com/technical-articles/an-overview-of-sequential-function-chart-sfc-programming/. [cit. 2023-11-20].

[6] *Programovací jazyky pro PLC*. Online. Výukový portál COPTel. 2019. Dostupné z: https://coptel.cz/mod/page/view.php?id=6737. [cit. 2023-11-20].

[7] *CANopen history*. Online. CAN in Automation. C1992-2023. Dostupné z: https://www.can-cia.org/can-knowledge/canopen/canopen-history/. [cit. 2023-12-28].

[8] *Embedded Networking with CAN and CANopen*. Copperhill Technologies Corporation, 2003. ISBN 978-0-9765116-2-5.

[9] *Introduction to the Controller Area Network (CAN)*. Revision B. Texas Instruments Incorporated, 2016. Dostupné z: https://www.ti.com/lit/an/sloa101b/sloa101b.pdf. [cit. 2023-12-28].

[10] *What is Can Bus (Controller Area Network)*. Online. DEWEsoft. 2021. Dostupné z: https://dewesoft.com/blog/what-is-can-bus. [cit. 2023-12-29].

[11] *Protokol CAN v automatizaci*. Online, Bakalářská práce, vedoucí doc. Ing. Ondřej Hynčica. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2015. Dostupné z: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=100717. [cit. 2023-12-29].

[12] *Controller Area Network Physical Layer Requirements*. Texas Instruments Incorporated, 2008. Dostupné z: https://www.ti.com/lit/an/slla270/slla270.pdf?ts=1703929283104. [cit. 2023-12-30].

[13] *Evaluační platforma pro CAN transceivery*. Online, Diplomová práce, vedoucí doc. Ing. Petr Fiedler, Ph.D. Brno: Vysoké učení technické v Brně, Fakulta elektrotechniky a komunikačních technologií, 2013. Dostupné z: https://www.vut.cz/www_base/zav_prace_soubor_verejne.php?file_id=82627. [cit. 2023-12-30].

[14] *What is CANopen?* Online. Influx Technology. 2021. Dostupné z: https://www.influxtechnology.com/post/what-is-canopen. [cit. 2024-01-03].

[15] *Napájení inteligentní automatizace pomocí systému CANopen*. Innodisk Corporation, 2020. Dostupné z: https://www.innodisk.com/epaper/multi-lang/Czech/assets/files/Innodisk_Powering_Smart_Automation_with_CANopen_White_Paper_202002_CZ.pdf. [cit. 2024-01-23].

[16] *CANopen internal device architecture*. Online. CAN in Automation. C1992-2024. Dostupné z: https://www.can-cia.org/can-knowledge/canopen/device-architecture/. [cit. 2024-01-03].

[17] *Řízení frekvenčního měniče ATV320 po komunikační sběrnici CANopen z PLC*. Online, Bakalářská práce, vedoucí Ing. Radek Havlíček Ph.D. Praha: ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE, Fakulta elektrotechnická, Katedra elektrických pohonů a trakce, 2017. Dostupné z: https://dspace.cvut.cz/bitstream/handle/10467/69531/F3-BP-2017-Markvart-Jan-Rizeni%20frekvencniho%20menice%20ATV320%20po%20komunikacni%20sbernici%20CANopen%20z%20PLC.pdf?sequence=1&isAllowed=y. [cit. 2024-01-03].

[18] *Catalog Modicon M241 Programmable Logic Controller for performance demanding applications*. Paříž: Schneider Electric, 2023. Dostupné z: https://www.se.com/cz/cs/product/TM241CEC24T/plc-modicon-m241-24-io-poz-logika-ethernet-can-master/. [cit. 2023-12-25].

[19] *How do brushless DC motors work? The need for a drive circuit explained*. Online. Aspina. 2021. Dostupné z: https://www.aspina-group.com/en/learning-zone/columns/what-is/014/. [cit. 2024-05-21].

[20] *Brushless Motors and Controllers*. Norderstedt: Books on Demand, 2012. ISBN 978-3-8423-9146-8.

[21] *Permanent Magnet Brushless DC Motor Drives and Controls*. John Wiley & Sons Singapore Pte., 2012. ISBN 978-1-118-18833-0.

[22] *SOLO PICO User Manual*. Revision V1.0.1. 2024. Dostupné také z: https://www.solomotorcontrollers.com/wpcontent/uploads/materials/SOLO_PICO_UserManual.pdf.

[23] *Automated guided vehicle*. Online. In: Wikipedia: the free encyclopedia. San Francisco (CA): Wikimedia Foundation, 2001-. Dostupné z: https://en.wikipedia.org/wiki/Automated_guided_vehicle. [cit. 2024-05-26].

[24] *Brushless Motor*. Online. Rozum Robotics. C2015-2024. Dostupné z: https://rozum.com/brushless-motors/. [cit. 2024-05-28].

[25] *CANopen User Manual*. Revision V 1.0.5. 2024. Dostupné také z: https://www.solomotorcontrollers.com/wpcontent/uploads/materials/SOLO_PICO_Communication_Manual_CANopen.pdf.

[26] *Brushless DC Motor 42BLF Series*. Online. Mikroe. C2024. Dostupné z: https://download.mikroe.com/documents/datasheets/BLDC%20Motor_BLDC%20Motor%20with%20Hall%20sensor%20Datasheet.pdf. [cit. 2024-05-28].

[27] *CANopen Manual*. PDF. V 3.0. 2019. Dostupné z: https://www.waycon.biz/fileadmin/draw-wire-sensors/CANopen-Manual.pdf. [cit. 2024-05-29].

Seznam symbolů a zkratek

Zkratky:

PLC	Programovatelný logický automat
CAN	Controller Area Network
I/O	Vstupy/výstupy, Input/Output
ČSN	Česká státní norma
LD	Jazyk kontaktních schémat, Ladder Diagram
RS	Reset – set
FBD	Jazyk funkčních bloků, Function Block Diagram
ST	Strukturovaný text
PC	Osobní počítač
IL	Seznam instrukcí, Instruction List
CAN FD	Controller Area Network Flexible Data-Rate
LIN	Local Interconnect Network
SENT	Single Edge Nibble Transmission
ISI	Inter-Symbol Interference
CANH	Controller Area Network High
CANL	Controller Area Network Low
TVS	Transil, Transient Voltage Suppression Diode
ABS	Anti-blokovací systém brzd
RTR	Remote Transition Request
IDE	IDentifier Extention
DLC	Data Length Code
CRC	Cyclic Redundancy Check
ACK	ACKnowledgement
EOF	End of Frame
IFS	InterFrame Space
SRR	Substitute Remote Request
OSI	Open Systems Interconnection
OD	Slovník objektů, Object Dictionary
PDO	Objekty technologických dat, Process Data Objects
SDO	Objekty servisních dat, Service Data Objects
PID	Proporcionální, integrační, derivační
NMT	Network Management
DC	Stejnosměrný
USB	Universal Serial Bus
FTP	File Transfer Protocol
SQL	Structured Query Language
LED	Light-Emitting Diode

BLDC	BrushLess Direct Current, bezkartáčový stejnosměrný
EC	Electronically Commutated
IPM	Intelligent Power Module
MOSFET	Metal Oxide Semiconductor Field Effect Transistor
IGBT	Insulated-Gate Bipolar Transistor
UART	Universal Asynchronous Reciever-Transmitter
AGV	Automated Guided Vehicle
RMS	Root Mean Square
PWM	Pulse Width Modulation, pulzně šířková modulace
CIA	CAN In Automation, CAN v automatizaci
API	Application Programming Interface
POU	Program Organization Unit
DUT	Data Unit Type
RPM	Revolutions per Minute, otáčky za minutu

Symboly:

Δu	diferenciální napětí sběrnice	(V)
I	proud vinutím motoru	(A)
U_d	napětí zdroje motoru	(V)
ΔU	úbytek napětí na spínačích měniče motoru	(V)
R_a	odpor vinutí motoru	(Ω)