ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

FAKULTA ELEKTROTECHNICKÁ

KATEDRA ELEKTRICKÝCH POHONŮ A TRAKCE

NÁVRH A REALIZACE ŘÍDICÍHO SYSTÉMU CIP SANITAČNÍ STANICE NA PLATFORMĚ PLC SIEMENS

BAKALÁŘSKÁ PRÁCE

Studijní program: Elektrotechnika, energetika a management

Studijní obor: Aplikovaná elektrotechnika

Vedoucí práce: Ing. Vít Hlinovský, CSc.

Jan Kollmann

PRAHA 2018

ČVUT
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

I. OSOBNÍ A STUDIJNÍ ÚDAJE

Příjmení: **Kollmann** Jméno: **Jan** Osobní číslo: **434988**

Fakulta/ústav: **Fakulta elektrotechnická**

Zadávající katedra/ústav: **Katedra elektrických pohonů a trakce**

Studijní program: **Elektrotechnika, energetika a management**

Studijní obor: **Aplikovaná elektrotechnika**

II. ÚDAJE K BAKALÁŘSKÉ PRÁCI

Název bakalářské práce:

Návrh a realizace řídicího systému CIP sanitační stanice na platformě PLC SIEMENS

Název bakalářské práce anglicky:

Design of CIP Station Controll System on SIEMENS PLC

Pokyny pro vypracování:

1) Návrhněte FDS a konfigurace řídicího systému založeného na jednotce SIEMENS S7 1500.
2) Vytvořte aplikační SW v prostředí TIA Portal pro řízení výrobního procesu, s využitím knihoven SIDAT v jazyku STL v prostředí S7.
3) Vytvořte SW simulující reálnou technologii.
4) Otestujte SW v režimu simulace s využitím aplikace simulující proces a uvedení do provozu.

Seznam doporučené literatury:

[1] HEROUT, Pavel. Učebnice jazyka C. Praha, 2014
[2] Firemní literatura Siemens
[3] Häberle, H. a kol. Průmyslové elektronika a informační technologie, EUROPA - SOBOTÁLES, Praha, 2003

Jméno a pracoviště vedoucí(ho) bakalářské práce:

Ing. Vít Hlinovský, CSc., katedra elektrických pohonů a trakce FEL

Jméno a pracoviště druhé(ho) vedoucí(ho) nebo konzultanta(ky) bakalářské práce:

Datum zadání bakalářské práce: **26.04.2018** Termín odevzdání bakalářské práce: 25. 5. 2018

Platnost zadání bakalářské práce: **30.09.2019**

Ing. Vít Hlinovský, CSc. podpis vedoucí(ho) práce	podpis vedoucí(ho) ústavu/katedry	prof. Ing. Pavel Ripka, CSc. podpis děkana(ky)

III. PŘEVZETÍ ZADÁNÍ

Student bere na vědomí, že je povinen vypracovat bakalářskou práci samostatně, bez cizí pomoci, s výjimkou poskytnutých konzultací. Seznam použité literatury, jiných pramenů a jmen konzultantů je třeba uvést v bakalářské práci.

15. 5. 2018
Datum převzetí zadání

Podpis studenta

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o dodržování etických principů při přípravě vysokoškolských závěrečných prací.

V Praze dne ……..…… Podpis ……

PODĚKOVÁNÍ

Touto cestou bych rád poděkoval svému vedoucímu bakalářské práce Ing. Vítu Hlinovskému, CSc. Také děkuji mé rodině za psychickou podporu při vypracovávání této bakalářské práce.

ABSTRAKT

Tato bakalářská práce se zabývá vytvořením programu pro CIP sanitační stanici. Řízení stanice zajišťuje PLC od firmy SIEMENS SIMATIC S7-1500. Program je napsán v jazyce STL s využitím firemních knihoven firmy SIDAT. Dále se tato práce zabývá otestováním programu v režimu simulace a uvedením zařízení do provozu.

KLÍČOVÁ SLOVA

PLC, SIMATIC S7-1500, CIP, sanitační stanice, programovací jazyk STL, TIA Portal, simulace technologických procesů

ABSTRACT

This thesis is concerned about creating a program for CIP sanitation station. This station is controlled by PLC from SIEMENS Corporation SIMATIC S7-1500. Program is written in STL language with the use of standard libraries SIDAT Corporation. Thesis is also concerned about testing the program in simulation mode and commissioning of the installation.

KEYWORDS

PLC, SIMATIC S7-1500, CIP, sanitation station, programming language STL, TIA Portal, simulation of technological processes

OBSAH

1 ÚVOD 1
2 NÁVRH FDS A KONFIGURACE ŘÍDICÍHO SYSTÉMU JEDNOTKY SIEMENS S7 1500 2
2.1 Popis Systému CIP 2
2.1.1 Popis obecné CIP sanitační stanice 3
2.1.2 Popis blokového schéma zadané CIP sanitační stanice 3
2.1.3 Cyklus CIP stanice 4
2.2 Návrh FDS 5
2.3 Hardwarová konfigurace PLC SIMATIC 6
2.3.1 Vlastnosti PLC SIMATIC řady S7-1500 6
2.3.2 Hardwarová konfigurace PLC 6
3 TVORBA APLIKAČNÍHO SW V PROSTŘEDÍ TIA PORTAL 8
3.1 Funkce programu Step 7 a prostředí TIA Portal 8
3.1.1 Datové typy PLC S7-1500 8
3.1.2 Stavové bity procesoru Step 7 9
3.1.3 Programové bloky Step 7 9
3.1.4 Vizualizace operátorského panelu 10
3.2 Instrukční sada jazyka STL 11
3.3 Programové bloky standartní knihovny SIDAT 14
3.3.1 Programové bloky zajišťující dílčí funkce a řízení technologických prvků 15
3.3.2 Programové bloky zajišťující automatické řízení 19
4 VYTVOŘENÍ SW SIMULUJÍCÍHO REÁLNOU TECHNOLOGII 23
4.1 Výhody simulačního softwaru 23
4.2 Tvorba simulačního softwaru 23
5 TESTOVÁNÍ V REŽIMU SIMULACE A UVÁDĚNÍ DO PROVOZU 25
5.1 Postup simulace 25
5.2 Uvádění do provozu 25
5.3 Zákonné normy hygieny 26
6 ZÁVĚR 27
7 PŘÍLOHY 28
8 SEZNAM POUŽITÉ LITERATURY 37

SEZNAM POUŽITÝCH ZKRATEK A SYMBOLŮ

PLC	Programovatelný logický automat
FDS	Functional and Design Specification, návrh funkce a vzhledu řídicího systému
CIP	Cleaning in Place, sanitace během provozu
CPU	Centrální procesorová jednotka
TIA	prostředí pro tvorbu softwaru pro PLC SIMATIC S7 a pro operátorské panely SIEMENS je zkratkou Totally Integrated Automation
SW	software, tedy algoritmus, podle kterého zařízení pracuje
VFD	Variable-frequency drive je pohon řízený frekvenčním měničem
PID	Proporcionálně integračně derivační typ regulace
HACCP	analýza rizik a hlídání kritických kontrolních bodů
STL	Statement list, programovací jazyk SIMATIC
RLO	současný výsledek logických operací
OB	organizační blok programu Step 7
HLS	high limit sensor, senzor maxima
LLS	low limit sensor, senzor minima
csv	comma-separated values, typ dokumentu s hodnotami oddělenými čárkami

1 ÚVOD

Hlavním cílem práce je uvézt do provozu CIP sanitační stanici zákazníka firmy SIDAT. Systém CIP je používán již více jak padesát let a je využívaný nejen v potravinářské výrobě, sanitace je totiž přímo integrována ve výrobním procesu a není proto nutné zařízením při čištění rozebírat.

Sanitace zahrnuje čištění a dezinfekci technologického zařízení přímo zevnitř. Dříve bylo čištění prováděno ručně a nebylo proto možné dosáhnout dostatečné důkladnosti a zároveň se muselo se zařízením složitě manipulovat. Dnes čištění probíhá automatizovaně pomocí přesně napsaných algoritmů, a tudíž je vyloučeno narušení sanitace operátorem. Mezi další nesporné výhody CIP stanice patří úspora zaměstnanců, úspory čisticích prostředků i vody, efektivita čištění, přesné postupy sanitace závislé na zadaných parametrech a tedy i možnost hledání nejefektivnějšího řešení sanitace pro každý výrobní postup. CIP stanice sestává z nádob s čisticími prostředky a roztoky, ze systému potrubních cest, kudy prostředky cirkulují, čerpadla, tepelného výměníku nebo jiného zdroje tepla a důležité měřící a regulační techniky.

Stanice bude řízena pomocí PLC od firmy SIEMENS typu SIMATIC S7-1500. Vytváření programu je realizováno v prostředí TIA Portal. Program využívá standartních bloků knihovny S7 firmy SIDAT a je psán v jazyce STL. CIP stanice je plně automatickým zařízením. Zákazník od systému požadoval plně automatické řízení procesu na základě vyplněné receptury a FDS jednotlivých kroků čištění, import a export receptur pomocí vyplněného souboru typu „csv“ jako funkci operátorského panelu. Import nastavení technologických prvků a symbolů globálních proměnných do prostředí TIA Portal vyplněním tabulky v aplikaci Microsoft Excel, možnost manuálního ovládání jednotlivých technologických prvků a simulační software pro snadnější uvádění do provozu.

Dalšími dílčími cíli jsou návrh FDS a konfigurace řídicího systému jednotky SIEMENS SIMATIC S7-1500, tvorba aplikačního SW v prostředí TIA Portal pro řízení výrobního procesu s využitím knihoven SIDAT v jazyku STL v prostředí S7, tvorba SW simulujícího reálnou technologii a otestování SW v režimu simulace s využitím aplikace simulující proces.

2 NÁVRH FDS A KONFIGURACE ŘÍDICÍHO SYSTÉMU JEDNOTKY SIEMENS S7 1500

Pro návrh FDS a konfiguraci řídicího systému je vhodné nejdříve znát fungování a parametry zařízení. Několik úvodních podkapitol se zabývá popisem funkce sanitační stanice. FDS upřesňuje algoritmus zařízení a je zadáním pro programátora. Po vytvoření FDS „Functional and Design Specification“ se vytvoří návrh konfigurace řídicího systému, v tomto případě se zákazník rozhodl pro PLC výrobce SIEMENS řady S7-1500.

2.1 Popis Systému CIP

Systém CIP (z anglického Cleaning in Place) je používán v praxi už více než padesát let. CIP bývá velmi efektivním čištěním a používá se v různých odvětvích průmyslu, především v potravinářských, nápojových a farmaceutických provozech. Sanitační stanice v těchto provozech dokonale čistí stroje a potrubí. CIP využívá kombinaci chemických prostředků, teplotu a vodu pro čištění strojů, nádob a trubek bez nutnosti demontování zařízení. [3]

Principy CIP se využívají v jakémkoli odvětví, ve kterém je důležitá čistota, a proces čištění je obvykle důležitou integrovanou součástí kteréhokoli zautomatizovaného podniku. Zpřísňování bezpečnostních a zdravotních opatření má za následek častější využívání principů CIP v praxi.

Při čištění se výrobní nádoba musí zbavit všech nežádoucích nečistot. Nečistoty by mohli způsobit rychlejší zkažení výrobku nebo jeho znehodnocení. Nečistoty nemusí být vždy viditelné pouhým okem, mohou mít formu bakterie nebo například výtrusu kvasinek. Sanitace objektu trvá nejméně patnáct minut za použití vhodných chemikálií s teplotami v rozsahu mezi padesáti a sedmdesáti pěti stupňů Celsia, další zvyšování teploty nepřináší žádné výhody. [3]

Obvykle využívané chemikálie pro odstraňování nečistot zahrnují zásadu (louh) ve formě hydroxidu sodného, kyseliny na bázi fosforu a dusíku, chlornan sodný a kyselinu peroctovou.

Hydroxid sodný se používá v roztocích v rozsahu mezi polovinou až dvěma procenty obsahu. Reaguje s tuky v nečistotách a zjednodušuje tak odstranění. Nevýhodou je, že není efektivní v odstraňování silnějších vrstev a anorganického odpadu.

Chlornan sodný přináší výhodu velmi nízké ceny. Primárně je používán jako dezinfekce, jeho schopnost odstraňování nečistot je nižší. Aktivní složkou je chlór, který může ve vysokých koncentracích narušit i nerezovou ocel a rovněž může poškodit čerpadla, musí se důkladně vyplachovat, neboť se může zkazit nebo se také může později smísit s kyselinou, čímž by vznikl jedovatý plyn. [3]

Kyselina peroctová je rovnovážná směs kyseliny octové a peroxidu vodíku. Je to silné oxidační činidlo, které je z hlediska oxidační kapacity porovnatelné s ozonem.

Vysoká efektivita čištění systému CIP se dosahuje pomocí kombinace chemikálií. Při použití pouze jednoho druhu chemikálie by účinnost klesla na pouhých zhruba patnáct procent. Při použití kombinace chemikálií se dosáhne tzv. synergického efektu, který zvýší účinnost až pětinásobně. [1]

Aby se zajistilo správné čištění, je nutné udržovat správnou rychlost proudění $\underline{v}$ uvnitř CIP. Nejefektivnější rychlost proudění se pohybuje mezi 1,5 až 2,1 [ms^{-1}]. [3]

Pro čištění nádob se obecně používají dvě hlavní metody. První využívá vysokého tlaku kartáčových hlavic pro mechanické odstranění nečistot, další pracuje pouze s nižším tlakem a využívá zejména efektivních chemikálií, které nečistoty chemicky rozloží.

Většina problémů systému CIP se vztahuje ke slabému zpětnému proudění. To může mít za následek zpomalení procesu, nesprávné aplikování čisticího prostředku a vyšší tepelné ztráty.

Pro překonání těchto problémů je nutné mít efektivní zpětné proudění do vratné linie CIP. V tomto ohledu je zcela zásadní výběr čerpadel. [3]

2.1.1 Popis obecné CIP sanitační stanice

CIP stanice sestává z nádob s čisticími látkami, potrubí, čerpadel a ohřívacího nástroje, kterým bývá tepelný výměník. Nejčastějším ohřevným médiem je pára. Nádoby jsou válcové a vyrobené z nerezové oceli. Objem nádob pro správnou funkci stanice musí být o dvacet procent vyšší, než zádrž nejdelšího čištěného potrubního okruhu. Nádoby bývají vybaveny průlezem, přepadem, sondou maximální a minimální výšky hladiny, vzorkovacím kohoutem, mycí hlavicí a přívodem vody a čisticího roztoku. Stanice může být přenosná a může vážit pouze několik desítek kilogramů, na druhou stranu jsou ale i CIP stanice, které zabírají několik místností. Na obrázku 2.1-1 je ilustrační fotografie. [1]

Obr. 2.1-1 [6]

Stanice sestává z několika nádob s čisticími prostředky. Nejčastěji se používají čtyři nádoby. Ve čtvrté nebývá čisticí prostředek, nýbrž použitá voda z posledního oplachu předchozího čistícího cyklu pro takzvaný předvýplach zařízení u cyklu následujícího.

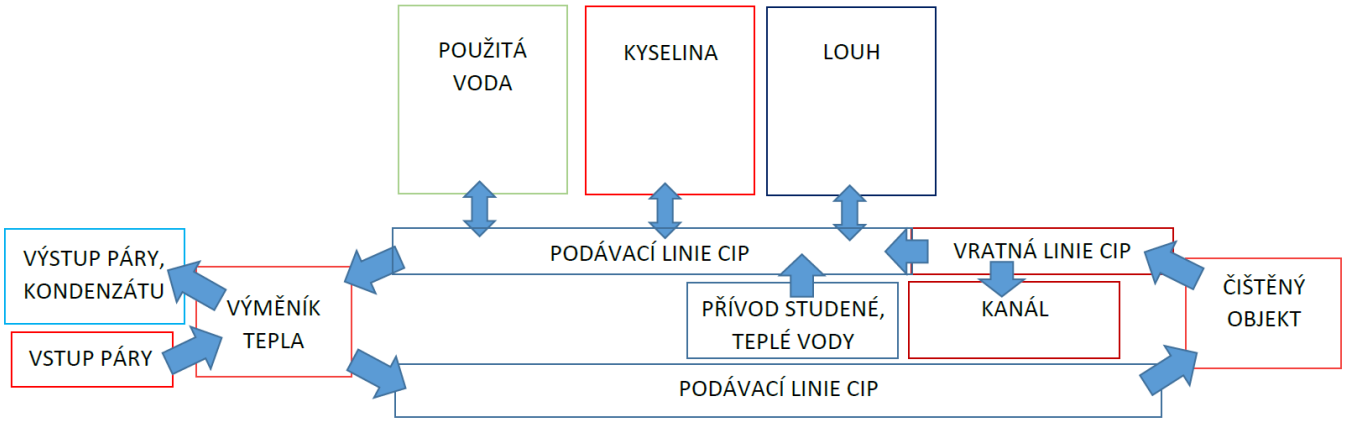
Čerpadla jsou odstředivá, také jsou vyrobena z nerezavějící oceli a jsou poháněna frekvenčním měničem. Pro správnou funkci musí mít výkonovou rezervu.

CIP stanice obsahuje také měřící prvky a prvky, které je nutné řídit nebo regulovat, kterými jsou frekvenční měnič pohonu čerpadel a ventily, popřípadě regulační ventily. Mezi měřící prvky patří teplotní senzor na podávací i na vratné linii, tlakový senzor, průtokový senzor, senzor vodivosti pro měření konduktivity a koncentrace čistícího roztoku a senzory nádob.

Teplotní čidla jsou zásadní pro regulaci teploty. **Tlakové čidlo** kontroluje činnost čerpadla a je nutné pro vyhlášení poruchových stavů vysokého, popřípadě nízkého tlaku. **Měření průtoku** je důležité pro kontrolu aktuálního průtoku, který je nutný pro zajištění správného proudění a efektivní proces čištění. Dále měření průtoku slouží ke kalkulaci proteklého množství a zajištění správných objemů proteklého média potrubních tras. **Senzor vodivosti** je nedílnou součástí zařízení, neboť s jeho pomocí můžeme poznat, zda skutečně cirkuluje odpovídající koncentrace čisticího prostředek. **Senzory nádob** přinášejí informace o aktuálním zaplnění nádob s čisticími látkami, tyto informace jsou rovněž velmi důležité pro řízení procesů sanitace.

2.1.2 Popis blokového schéma zadané CIP sanitační stanice

Téma této bakalářské práce je návrh a realizace řídicího systému CIP sanitační stanice na platformě PLC SIEMENS. Níže uvádím blokové schéma reálné CIP stanice, ke které jsem tvořil řídicí systém.



Obr. 2.1-2

Tato stanice sestává ze tří nádob, kterými jsou chemické prostředky na bázi kyseliny (nízké pH), chemické prostředky na bázi zásady (vysoké pH, na schématu označené jako louh) a použitá voda, která bývá použita na takzvaný předvýplach v dalším cyklu mytí.

Nádoby jsou vybaveny čidly, která poznají minimální popřípadě maximální objem prostředku v tanku.

Tyto tanky jsou přes řízené ventily spojeny s podávací linií CIP, což umožňuje uložení nebo vyzvednutí chemikálií za použití čerpadla.

Součástí podávací linie je podávací čerpadlo řízené frekvenčním měničem, průtokoměr, výměník tepla, měření teploty a měření tlaku. Podávací linie je spojena s přívodem studené nebo i horké vody. Výměník tepla používá jako ohřívací médium páru a výkon výměníku je regulován regulačním ventilem přívodu páry.

Podávací linii je nutné manuálně připojit k čištěnému objektu. Případné vratné čerpadlo není součástí naší sanitační stanice.

Vratná linie se musí také připojit manuálně. Součástí vratné linie CIP je čidlo průtoku, měření konduktivity nebo koncentrace a měření teploty. Vratná linie CIP je propojena s kanálem, s jednotlivými tanky a lze ji propojit do okruhu s podávací linií.

2.1.3 Cyklus CIP stanice

Typický průběh čištění CIP stanice se skládá z několika kroků. Jde o sled působení čistících látek, proto nezáleží na čištěném objektu.

Nejdříve dochází k předvýplachu vodou. Voda bývá brána z nádoby použité vody. Předvýplachem se mají odstranit nejhrubější nečistoty, může být prováděn studenou nebo teplou vodou, avšak do teploty padesát stupňů Celsia. Kapalina z tanku použité vody navíc bývá již předehřátá z minulého procesu čištění, takže není potřeba ji dohřívat. Délku předvýplachu je nutné experimentálně stanovit pro každou aplikaci při uvádění do provozu. Po předvýplachu je tato voda poslána do filtru a následně do kanalizace.

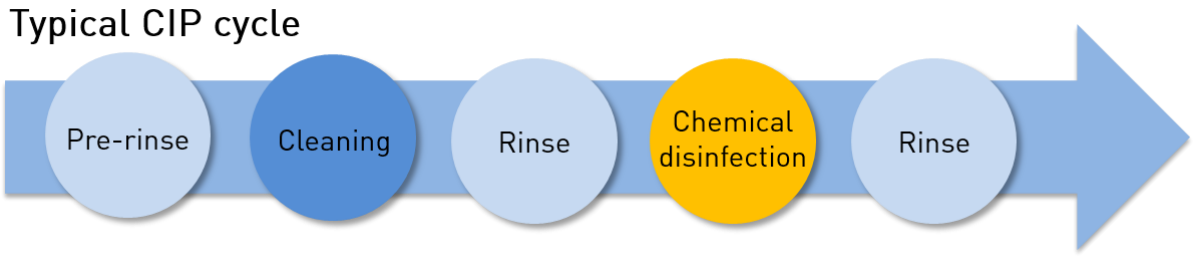
Dalším krokem je alkalické čištění. Dochází k němu při teplotě zhruba osmdesát stupňů Celsia a při koncentraci čisticího prostředku do dvou a půl procenta objemu roztoku.

Následuje mezivýplach čistou vodou. Voda z mezivýplachu se může načerpat do použité vody, slouží k odstranění zbytku čisticích prostředků.

Dále může následovat jiná forma čištění, například čištění prostředky na bázi kyseliny, po kterém by opět následoval mezivýplach. Kyselé čištění zabrání usazení vodního kamene, může mít i částečně dezinfekční účinek. Teplota pro čištění bývá do třiceti pěti stupňů Celsia.

Po posledním mezivýplachu přichází dezinfekce. Dezinfekce se nechává cirkulovat poměrně krátce, poté se ale nechává delší čas na účinkování dezinfekce.

Oplach pitnou vodou odstraní poslední zbytky po čištěních a dezinfekci. Oplach trvá, dokud není pH vody, která cirkuluje neutrální (pH=7), kvůli zajištění bezpečnosti výrobků se často nechává oplach až polovinu svého trvání (čas do neutrálního pH popřípadě odpovídající vodivosti) navíc. Pokud je použita nezávadná dezinfekce (například chlordioxidová voda), není poslední oplach nutný. [1]



Obr. 2.1-3 [7]

2.2 Návrh FDS

V příloze je maticový zápis zadané FDS. FDS je anglickou zkratkou „Functional and Design Specification“, což znamená specifikace funkce a návrhu řídicího systému. FDS jsem spoluvytvořil na základě zadání od zákazníka a diskuze s ním.

Maticový zápis je velmi přesná a přehledná forma zápisu FDS. Z těchto důvodů je v praxi nejčastěji používána. V řádcích vidíme jednotlivé procedury, které musí obsahovat řídicí systém. Tyto procedury jsou popsány parametry ve sloupcích.

Prvním parametrem je číslo procedury, které je jedinečné v projektu, v dalším sloupci vidíme typ procedury, které jsou rozdělené na O, P, T a CIP. P je určeno pro proceduru pro čištění potrubí, T je určeno pro proceduru pro čištění tanku a O je obecná procedura, která se využívá v receptech pro čištění potrubí i tanku. Procedura typu CIP je interní procedurou pro přípravu tanků a médií. Čištění samotné CIP stanice probíhá při využití vnitřního okruhu (bit F), operátor pak musí otevřít a zavřít příslušné ruční ventily.

Dalšími parametry jsou povely pro jednotlivé akční členy technologie CIP stanice v závislosti na podmínkách procesu, v zásadě na proměnných a na receptuře. Jednotlivé číselné poznámky maticového zápisu jsou vysvětleny v přiložené tabulce k FDS, který je rovněž v příloze. Například podávací čerpadlo je spuštěno ve všech procedurách, kromě procedur typu „odčerpání“. Pokud je pole matice prázdné, znamená to, že se daný prvek v této proceduře nevyužívá nebo že se daná veličina v této proceduře nehlídá. Zkratka HLS znamená „high limit sensor“, tedy senzor maximální hladiny tanku, LLS pak znamená „low limit sensor“, senzor minimální hladiny tanku.

Sloupce veličin se dělí na veličiny, které mají vliv na přechodové podmínky do dalších procedur, hlídané veličiny a regulované veličiny. Při splnění podmínek vyplněné veličiny s přechodovou podmínkou (pokud je veličina v receptuře nevyplněna, přechod nenastane), proces sanitace přejde do další procedury v receptuře. Při splnění podmínek hlídaných veličin musí řídicí systém zareagovat podle zadání. Například pokud je v jakékoli proceduře déle než minutu teplota na podávací linii vyšší než devadesát pět stupňů Celsia, musí se proces pozastavit a vypíše se chybové hlášení: „Vysoká teplota na výtlaku“. Až v případě, kdy je teplota nižší, než devadesát pět stupňů Celsia je možné proces obnovit.

Posledními veličinami jsou veličiny regulované, ty předznamenávají spouštění PID regulací. Regulace průtoku čerpadla je spuštěna pokaždé se zapnutím motoru čerpadla, regulace regulačního ventilu páry se spouští s požadavkem na zvýšení teploty v okruhu podle receptury nebo požadavku operátora. Simultánně se zapnutím regulace teploty se zapíná i odpovídající regulační ventil páry.

2.3 Hardwarová konfigurace PLC SIMATIC

Pro fungování strojů, vybavení a procesů v téměř všech odvětvích průmyslu jsou nutná řídicí zařízení spolu s dodávkou elektřiny. Musí být možné začít, kontrolovat, monitorovat a ukončit operaci každého stroje nebo procesu.

Dříve bylo nutné procesy řídit procesy specifickým zapojením hradel a relé pro danou aplikaci. Dnes se pro řízení využívají programovatelné logické automaty (dále PLC). Díky nim je možné výsledný efekt řízení kdykoli přizpůsobit podmínkám provozu pouhým přeprogramováním zařízení.

Procesy již není nutné dělit na jednotlivé malé aplikace, ale spíše jsou součástí celého produkčního procesu.

2.3.1 Vlastnosti PLC SIMATIC řady S7-1500

Je to modulární řídicí systém se středním až vyšším výkonem, centrální procesorová jednotka (dále CPU) se dodává v různých výkonech, má svůj systémový čas, diagnostický buffer, ve kterém je uložen archiv událostí a aktuální stav. K jednotce CPU je možné přiřadit širokou škálu modulů. Rychlá komunikační sběrnice (bus) pro komunikaci se vstupy a výstupy řídicího systému takzvané I/O. K CPU je možné připojit až třicet dva rozšiřujících modulů. Jednotka CPU může být přímo připojena k síti PROFIBUS nebo PROFINET, záleží na typu procesorové jednotky. Vlevo od jednotky CPU se instaluje zdroj napájení systému, obvyklou velikostí napájení je 24 [V] stejnosměrného napětí. Napravo od CPU rozšiřující moduly pro vstupy a výstupy (digitální i analogové), technologické moduly (rychlé čítací a pro polohování), komunikační moduly, případně další zdroje napájení. Není-li dostatečný systém pro napájení více rozšiřujících modulů, je nutné do sestavy zařadit další zdroj napětí. Každý rozšiřující modul vstupů a výstupů má speciální výklopný konektor pro připojení až čtyřicet vodičů. CPU řady 1500 je vybaveno malou barevnou obrazovkou a několika ovládacími tlačítky, se kterými je možné provést několik základních úkonů. Zde je možné změnit stav CPU, kterými jsou STOP, RUN, zobrazit si diagnostický buffer, verzi CPU, statusy od rozšiřujících modulů, zobrazení alarmů a zpráv, změnit nastavení jazyka, Ethernetovou nebo Profibusí adresu, čas, restartovat základní procesorovou jednotku. Také je možné zaheslovat tuto obrazovku nebo změnit její nastavení jasu a délky intervalu, po kterém obrazovka zhasne. Rovněž má tři signalizační LED diody, které ukazují stavy CPU, kterými jsou RUN, STOP a reset paměti.

Systém S7 1500 podporuje používání technologických funkcí, což je propojení speciálních modulů s již připravenými podprogramy pro časté technologické části, jakými jsou: rychlý čítač, který čítá až rychlostí 100 [kHz], řízení pohonů (rychlost a polohování) a regulační smyčky typu PID. Pro funkci CPU je nutná paměťová karta Memory Card pro SIMATIC SIEMENS, což je speciální druh paměťové karty typu Secure Digital- SD, jejíž médiem je flash paměť. U řady S7 – 1500 je možné na tuto kartu archivovat procesní data. Je možné využít distribuovaný IO systém, v tom případě nejsou tyto vstupy a výstupy součástí centrálního rozvaděče s CPU, ale jsou instalovány v podružném rozvaděči za pomoci systému ET 200MP a ET 200SP, pro komunikaci se používá PROFIBUS nebo PROFINET.

2.3.2 Hardwarová konfigurace PLC

Zákazník objednal a firma SIDAT s.r.o. pro tuto aplikaci PLC schválila následující sestavu zařízení.

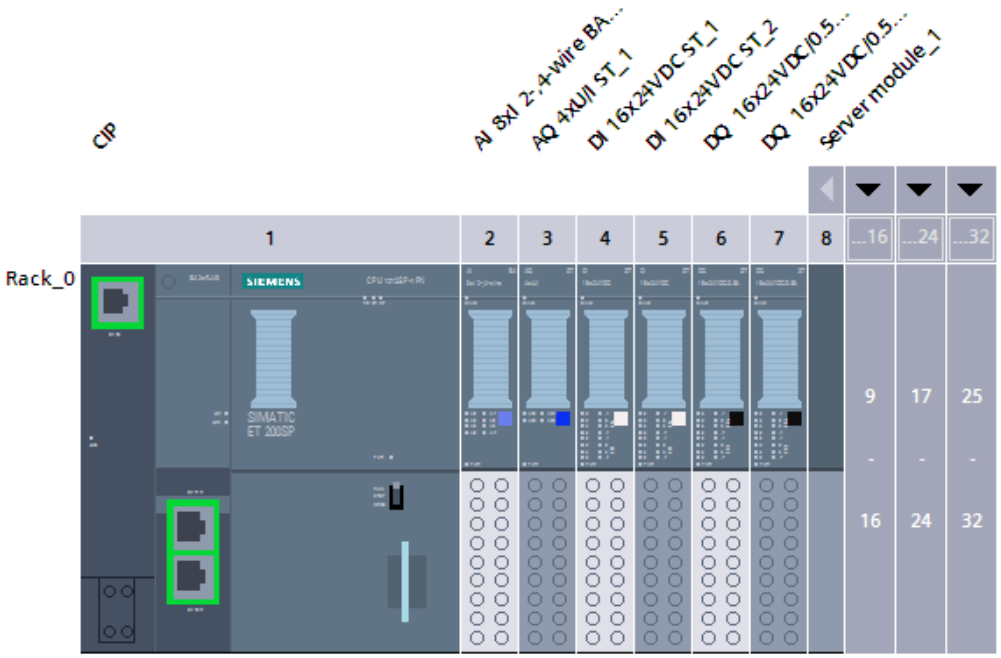
CPU1512SP-1 PN je CPU s pracovní pamětí 200 [kB] pro program a 1 [MB] pro data, bitová instrukce tomuto typu centrální procesorové jednotky trvá $4{,}8 \cdot 10^{-8}$ [s]. Tato jednotka podporuje technologické funkce, řízení pohonů (řízení rychlosti a polohování) a řízení uzavřených smyček. Jednotka CPU podporuje PROFINET IO, s protokolem TCP/IP propojení s místní PROFINET sítí. Na CPU je integrovaný web server. Ten slouží k vytvoření internetového rozhraní, na které je možné se připojit z kteréhokoli dostupného počítače v místní síti a zde je možné sledovat nadefinované údaje, případně je i měnit. Podporuje komunikaci s DNS (Domain name systém) severy a je možné vytvořit OPC UA server, který komunikuje s cizími zařízeními a aplikacemi.

Vybrané rozšířené moduly k této jednotce CPU:

- AI 8xI, 2-,4- wire BA
- AQ 4xU/I ST
- 2x DI 16x24VDC ST
- 2x DQ 16x24VDC/0.5A ST

Dále je k PLC připojen operační panel **TP 700 Comfort.**

Na obrázku 2.3-1 je hardwarová konfigurace PLC z prostředí TIA.



Obr. 2.3-1

Rozšiřující modul **AI 8xI** je modul analogových vstupů se šestnácti bitovým analogově digitálním převodníkem. Je možné ho zapojit systémem dvou drát nebo čtyř drát (při zapojení dvou drát poskytuje napájení do měřící smyčky AI modul, při zapojení čtyř drátovém, je AI modul pouze pasivní a měří napětí smyčky). Je možné nastavit rozsah vstupního proudu na 0 [mA] až 20 [mA], 4 [mA] až 20 [mA] nebo Bipolární od -20 [mA] do 20 [mA]. Má osm analogových kanálů velikosti šestnáct bitů, je tedy možné do něj zapojit až osm měření.

Rozšiřující modul **AQ 4xU/I** je modul analogových výstupů se šestnácti bitovým analogově digitálním převodníkem. Je možné nastavit pro výstupní analogovou smyčku následující rozsahy 0 [mA] až 20 [mA], 4 [mA] až 20 [mA] nebo Bipolární od -20 [mA] do 20 [mA] proudu nebo 1 [V] až 5 [V], 0 [V] až 10 [V], - 5 [V] až 5 [V] či -10 [V] až 10 [V] stejnosměrného napětí. Má čtyři analogové kanály.

Rozšiřující modul **DI 16x24VDC** je modul digitálních vstupů pro připojení až šestnácti vstupů o stejnosměrném napětí 24 [V].

Rozšiřující modul **DQ 16x24VDC/0.5A** je modul digitálních výstupů pro připojení až šestnácti kanálů s výstupními parametry: stejnosměrné napětí 24 [V], proud 0,5 [A].

Operační panel **TP 700 Comfort** je sedmi palcový operační panel s TFT obrazovkou. Jeho rozlišení je 800 x 480 pixelů, podporuje 16M barevný systém, který zobrazí až 2^{32} barevných odstínů. Má dotykové ovládání, je možné se s ním propojit přes MPI, PROFIBUS nebo PROFINET, Industrial Ethernet, navíc má dva vstupy multimediálních karet a tři USB rozhraní.

3 TVORBA APLIKAČNÍHO SW V PROSTŘEDÍ TIA PORTAL

3.1 Funkce programu Step 7 a prostředí TIA Portal

3.1.1 Datové typy PLC S7-1500

Základní datové typy SIMATICU S7 se dělí na binární čísla, integery, čísla ve formátu floating point, časovače, datový typ, který ukazuje datum a čas nebo řetězec znaků (písmen). Možnost vytváření a používání šedesáti čtyř bitových základních datových typů je novou funkcí spjatou pouze s PLC typu S7-1500. Níže je tabulka základních datových typů.

Globální proměnné v prostředí TIA se nazývají „tag“. Proměnné se na rozdíl od klasického prostředí Step 7 hledají v první řadě podle symbolické, nikoli pomocí absolutní adresy.

Typ proměnné	Využitá paměť	Formát proměnné
Boolean	1 bit	Nula nebo jedna
Byte	8 bitů	Řetězec Booleanů
Word	16 bitů	Řetězec Booleanů
Double Word	32 bitů	Řetězec Booleanů
Long Word	64 bitů	Řetězec Booleanů
Short Integer	8 bitů	Celé číslo
Unsigned Short Integer	8 bitů	Přirozené číslo nebo nula
Integer	16 bitů	Celé číslo
Unsigned Integer	16 bitů	Přirozené číslo nebo nula
Double Integer	32 bitů	Celé číslo
Unsigned Double Integer	32 bitů	Přirozené číslo nebo nula
Long integer	64 bitů	Celé číslo
Unsigned Long Integer	64 bitů	Přirozené číslo nebo nula
Real	32 bitů	Reálné číslo (racionální)
Long Real	64 bitů	Reálné číslo (racionální)
S5 Time	16 bitů	Délka trvání časovače
Time	32 bitů	Délka trvání časovače
Long Time	64 bitů	Délka trvání časovače
Date	16 bitů	Datum
Time of day	32 bitů	Datum a čas
Long time of day	64 bitů	Datum a čas
Characters	8 bitů	Znak ASCII
Wide characteres	16 bitů	Znak Unicode

Tabulka 3.1-1

3.1.2 Stavové bity procesoru Step 7

Procesor S7 SIMATIC využívá logických příznaků, které nastavuje při operacích, což jsou **stavové bity.** Stavové bity procesoru vytváří stavové slovo. Procesor využívá stavové bity pro řízení logických operací a zjišťování výsledků aritmetických a matematických funkcí. Níže je tabulka stavových bitů. [2]

Bit	Jméno registru	Funkce registru
0	/ER	První dotaz
1	VKE/RLO	Logický operátor
2	STA	Stav
3	OR	Stavový bit OR
4	OS	Přetečení s pamětí
5	OV	Přetečení
6	A0	Stavový bit A0
7	A1	Stavový bit A1
8	BIE	Logický příznak

Tabulka 3.1-2 [2]

První dotaz je stavový bit, který nabývá jedné, v případě rozpracování, logické funkce. Po zpracování instrukce je zase nastaven do nuly.

Logický operátor je paměť, do které se ukládají průběžné výsledky logických operací. Často se označuje RLO, výsledkem logických operací.

Stav odpovídá hodnotě testované podmínky, popřípadě logické hodnotě, kterou procesor ukládá do proměnné.

Stavový bit OR nabývá hodnoty jedna v případě, že procesor má hodnotu RLO jedna a zpracovává instrukci O (And before Or), dává tím najevo, že výsledek následné logické operace je jistě jedna a není potřeba jej zpracovávat.

Bit **přetečení** indikuje překročení rozsahu při počítání nebo použití neplatných čísel formátu REAL.

Přetečení s pamětí je trvale nastavený bit při překročení rozsahu při kterékoli minulé operaci. Zatímco bit přetečení se při další instrukci, která nezpůsobí přetečení, vynuluje, přetečení s pamětí se dá kdykoli zpětně vyhodnotit.

Stavové bity A0 a A1 se používají při porovnáních, aritmetických a matematických funkcích, operacích se slovy, případně při operacích posuvu.

Logický příznak se používá v grafických jazycích, v jazyce STL se dá využít jako volný paměťový bit, který je možné vyhodnotit logickými instrukcemi nebo skokovými operacemi.

3.1.3 Programové bloky Step 7

S7 SIMATIC kód programu nejen v prostředí TIA se píše do třech základních programových bloků.

Prvním je OB, **organizační blok**. Organizační blok vytváří spojení mezi programem uživatele a operacemi procesoru. OB 1, první organizační blok, je cyklicky volané operačním systémem po každém dokončení cyklu OB 1, popřípadě po uběhnutí naprogramovaného zpoždění (minimální čas exekuce OB1). Program proto může být celý napsaný přímo v OB1 (lineární program), nebo můžeme odtud volat další bloky (strukturovaný program nebo program rozdělený na části). Strukturovaný program na rozdíl od programu rozděleného na části obsahuje

programové bloky, které jsou natolik obecné, že je možné je používat ve více aplikacích, například stejná funkce, která byla vytvořena pro obsluhu ventilu, může být díky parametrům použita pro každý ventil.

Na začátku každého cyklu OB 1 se zkopírují data periferií do paměti. Paměť vstupních karet (PI) se zkopíruje do vstupních proměnných jednotky CPU (I). Na konci cyklu OB1 se paměť výstupních proměnných jednotky CPU (Q) zkopíruje do výstupních periferií (PQ), pokud se tedy během cyklu mění několikrát výstupní proměnná (Q), ve skutečnosti se změna projeví pouze po skončení cyklu OB, stejně tak pokud se během cyklu změní vstupní periferie, vstupní proměnná bude mít zaktualizovanou hodnotu až pří dalším cyklu OB 1.

Organizační bloky se dělí na bloky aktivní při spouštění PLC, bloky, které se vykonávají v cyklu, bloky, které se vykonávají při přerušení, můžou mít cyklické přerušení nebo přerušení v určitý čas každý den a organizační bloky, které jsou volány při určité události, může se jednat o časové zpoždění, hardwarové volání, asynchronní přerušení, přerušení řízení pohybu nebo chybovou událost. Podle čísla OB, které program při chybě zavolá, poznáme, jaká chyba nastala. Pokud dojde k chybě v kódu a je v programu příslušné chybové OB, program pouze zavolá příslušné OB, zůstane ve stavu RUN a pokračuje ve vykonávání programu. Je vhodné tato OB vkládat při hledání chyb, která způsobují zastavení vykonávání programu, mohou přinést cenné informace o chybě.

Dalším typem programového bloku S7 programování je **funkce**, nebo také FC. Funkce bývají volány s různými parametry a mohou být zavolány vícekrát během jednoho cyklu OB. Funkce jsou vhodné k plnění programu, který se často opakuje a to i během jednoho OB, typickým příkladem bývají výpočty. Proměnné vytvořené ve funkci jsou pouze dočasné, to znamená, že se nedají používat jako paměť mezi cykly OB, protože se jejich hodnota může při dalším cyklu nezávisle měnit. Bývají tedy pouze mezivýsledkem. Pokud chceme vytvořit proměnou s pamětí, je vhodnější vytvořit místo funkce funkční blok. Můžeme ale jako paměť využít i paměť programu (markery) nebo některý z datových bloků. Při využití programové paměti nebo datového bloku, hrozí, že pokud funkci používáme pro více stejných prvků, uložená data se budou přepisovat a je nutné vyřešit, aby každé volání funkce využívalo jinou část paměti.

Posledním typem jsou **funkční bloky** nazývané zkratkou FB, které v zásadě mají stejné možnosti jako funkce, ale navíc mají možnost statických proměnných, což jsou proměnné uložené v instančním datovém bloku, který je volán spolu s funkčním blokem. Proto jsou funkční bloky vhodné k řešení komplexnějších úloh, například k řízení uzavřených smyček.

Programové bloky v prostředí TIA, což je jediné programovací prostředí, ve kterém je možné vytvářet program pro PLC S7-1200 a S7-1500, mají při vytvoření optimalizovaný přístup. Proměnné v optimalizovaných blocích jsou automaticky uspořádány podle jejich datového typu. Díky tomu je zajištěno, že je úložný prostor efektivně využit a že procesor k datům přistupuje optimálně. Optimalizovaný přístup bloků je výhodný díky rychlejšímu zpracovávání dat, je sníženo riziko nesouladu dat, každá proměnná (tag) zvlášť může být zvolena jako „retentive“, což je nastavení proměnné, které umožňuje v případě chyby přesun aktuální hodnoty této proměnné z pracovní paměti do stálé paměti jednotky CPU. Při opětovném zapnutí (Startup) se získá hodnota ze stálé paměti jednotky a uloží se do „retentive“ proměnné. Nevýhodou optimalizovaného přístupu je, že k datům uvnitř optimalizovaných bloků se dá přistupovat jen symbolicky, ne jejich absolutní adresou. Každá jednotlivá proměnná typu „Boolean“ je uložená ve vlastním datovém Bytu, proto k nim má procesor jednodušší přístup, nemusí totiž maskovat jiné proměnné. Optimalizovaný přístup je výhodný pro efektivní práci procesoru, ale způsobuje omezení při tvorbě programu. Programátor nemůže používat absolutní adresy. Většina způsobů nepřímého volání proměnných v S7 SIMATIC je založených právě na absolutních adresách.

Parametry funkce i funkčního bloku se dělí na parametry vstupní, výstupní a na parametry, které jsou zároveň vstupními i výstupními, v případě funkčního bloku jsou aktuální parametry uložené v instančním bloku tj. datovém bloku přiřazenému k funkčnímu bloku.

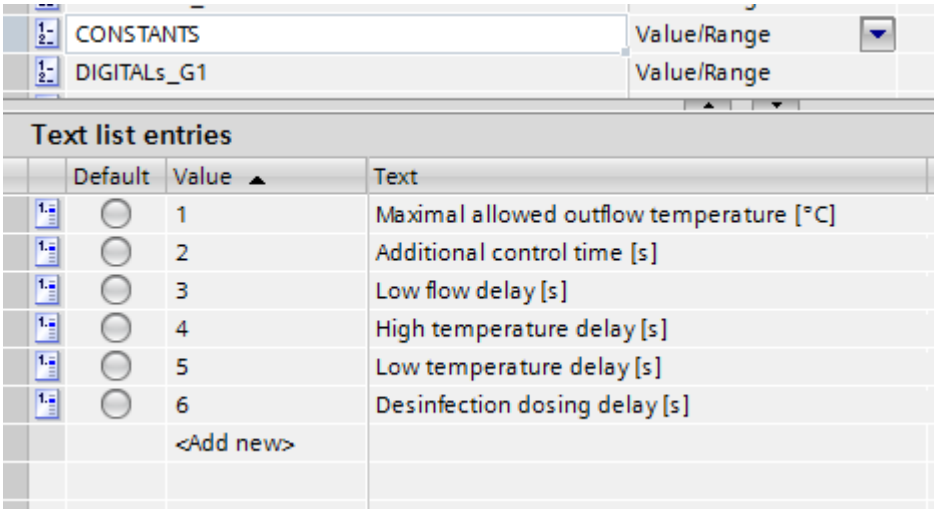
3.1.4 Vizualizace operátorského panelu

Prostředí TIA Portal podporuje vytváření vizualizace a funkcí operátorského panelu pomocí zavedeného SIEMENS „Win CC“. Častým firemním standardem bývá psaní programu v prostředí Step 7 pro PLC S7 300 a S7 400 a vytváření programu pro operační panel právě v prostředí TIA. Vizualizaci panelu jsem nevytvářel já, ale na některých dílčích úkolech jsem spolupracoval. Například psaním textlistů a pomocí s designovou stranou.

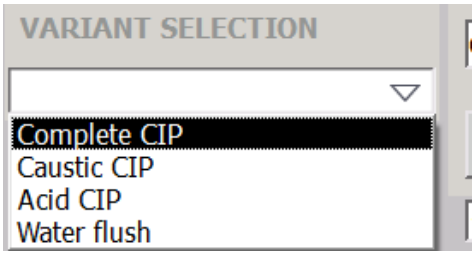
Na operátorském panelu je možné zapnout nebo vypnout simulační bit, ten rozhoduje, zda PLC pouze simuluje chování technologie nebo se používají reálná data zařízení. Některá data datových bloků technologie je možné měnit přímo z operátorského panelu. Je možné měnit předpokládaná data, která program přijímá z daného prvku nazpět. Jimi jsou zpětná hlášení stavu technologie.

Každý technologický prvek je možné přepnout do **manuálního režimu**, ve kterém jsou ovládány přímo z panelu nebo do automatického režimu. Pokud jsou veškeré technologické prvky stanice v automatickém režimu a nejsou v chybě, je možné spustit automatické čištění stanice v závislosti na receptu.

Pro správnou funkci operátorského panelu je nutné vytvořit takzvané **textlisty**. Díky nim se na základě hodnoty některé proměnné objeví text přiřazený této hodnotě na operátorském panelu. Takto na základě textlistů může operátor vybrat správný typ zařízení a nemusí si pamatovat, co které číslo typu znamená nebo například se díky hodnotě proměnné chyby objeví správný název chyby přímo na panelu.



Nejdůležitějšími ovládacími prvky na operátorském panelu jsou tlačítka a vstupní a výstupní pole. Tlačítka po stisku nebo během stisku konají nějakou akci, nejčastěji nastavují určitý bit souvislé proměnné. Výstupní pole informuje operátora ohledně důležitých proměnných uvnitř PLC. Výstupní pole se nedají měnit a ukazují hodnotu těchto proměnných nebo informaci s touto hodnotou související na základě textlistu. Pomocí vstupního pole nebo vstupního a současně i výstupního pole může operátor měnit hodnotu proměnné, na rozdíl od tlačítka nabízí pole více možných variant změny.



Všechny ovládací prvky lze nastavit, aby se dali použít pouze oprávněnými osobami, které se autorizují napsáním příslušného jména a hesla, nebo je v některých případech znepřístupnit a skrýt. Je také možné nastavit jejich vzhled (barvu, text) na základě proměnných a přímo tak předestřít funkci tlačítka nebo vstupního pole.

3.2 Instrukční sada jazyka STL

Kód pro S7 SIMATIC typu 1500 lze psát v několika programovacích jazycích. Jsou jimi grafické jazyky LAD, FBD a GRAPH, jejich výhodou je názornost. Grafický jazyk LAD vychází z Ladder, který je často používán na školách a mnoho lidí je se základy jeho logiky seznámeno. Dále lze kód napsat v jazycích STL a SCL. SCL je jazyk vycházející z jazyku PASCAL. Je zařazován mezi vyšší textové jazyky, zjednodušuje programování matematických algoritmů a komplexní práci s daty. V poslední době se SIEMENS snaží tento jazyk prosadit, například tím, že je to jediný textový jazyk, kterým lze programovat PLC S7 SIMATIC typu 1200. V naší aplikaci používáme především jazyk STL (Statement list), nižší programovací jazyk, který vychází

z jazyku Instruction List. Řízení programu je zařízeno instrukcemi podmíněných skoků, které nahrazují instrukci „if“ z jiných programovacích jazyků.

Instrukce v jazyce STL se dělí na bitové logické instrukce, instrukce porovnávací, instrukce určené ke konverzi proměnných, instrukce pro čítače, instrukce, které pracují s datovými bloky, logické instrukce určené pro řízení programu, kterými jsou podmíněné skoky a smyčky, instrukce pro počítání s proměnnými typu integer a dalších typů s ním podobných (Double integer, short integer, unsigned integer, …), instrukce pro počítání s proměnnými typu float a real, instrukce pro čtení a zápis proměnných z akumulátoru, programové instrukce TIA (pro volání bloku a ukončení bloku), instrukce posunů a rotace bitů v akumulátoru, instrukce pro časovače, logické instrukce, které zpracovávají více bitů zároveň (šestnáct popřípadě třicet dva bitů) a instrukce pro práci s akumulátory.

Bitové logické instrukce jsou instrukce, které jsou základem v Booleovy algebry. Pracují s nulou a jedničkou. Tyto dvě číslice jsou základem binární číselné soustavy. Číslice jedna a nula se nazývají binárními číslicemi nebo také bity. Ve světě kontaktů a cívek je číslicí jedna myšleno, že je daná věc aktivní nebo že má energii, číslice nula znamená, že daná věc je neaktivní popřípadě, že je bez energie. Bitové logické instrukce ukazují signálové stavy jedna a nula a jejich kombinaci na základě Booleovy logiky. Booleova algebra bývá definována jako distributivní komplementární svaz.

Základní instrukce bitové logiky se v jazyce STL dají pomocí závorek kombinovat, aby velmi přesně vyjádřily výslednou funkci. Výsledkem těchto kombinací na základě proměnných je vždycky znovu nula nebo jednička, tento výsledek se označuje „result of logic operation“ (**RLO**)

První instrukcí je **A** (And), ve výrokové logice je spjata s logickou spojkou konjunkcí, která má znak $\wedge$. Tato instrukce porovnává současné RLO s bitovou proměnnou, která je napsána za instrukcí. Pouze pokud jsou oba bity ekvivalentní jedné (současné RLO i současně zpracovávaná proměnná), je výsledek, a i následné RLO po této instrukci ekvivalentní jedné. Naopak, pokud je současné RLO nebo současně zpracovávaná proměnná nulou, následné RLO po této instrukci bude ekvivalentní nule.

Další velmi používanou logickou instrukcí v jazyce STL je **AN** (And not), což je komplementární instrukce k And, která jako následné RLO ukazuje jedna právě a pouze tehdy, pokud je současné RLO ekvivalentní jedné a zpracovávaná proměnná je ekvivalentní nule.

Instrukce **O** (Or) je ve výrokové logice spjata s logickou spojkou disjunkcí, jejíž symbol je $\vee$. Právě tehdy, když je současné RLO nula a současně je právě zpracovávaná proměnná ekvivalentní nule, je následné RLO po této instrukci ekvivalentní nule. Pokud je RLO nebo proměnná ekvivalentní jedné, je výsledné RLO také ekvivalentní jedné.

Komplementární instrukcí k Or je **ON** (Or not). RLO po ON je ekvivalentní nule právě a pouze tehdy, když je současné RLO ekvivalentní nule a současně zpracovávaná instrukce je ekvivalentní jedné.

V rámci Booleovy logiky lze jakákoli logická funkce napsat pomocí jedné z výše uvedených instrukcí a negace. Programátor zpravidla využívá instrukcí více, aby byli pochopitelnější pro jeho nástupce, který by s programem pracoval a dotvářel jej. Dalšími možnými instrukcemi je **XOR** (exclusice Or) a **XORN** (exclusice Or not), které se v praxi tolik nevyužívají, a které jsou spjaty s logickou spojkou ekvivalence.

Na základě RLO fungují instrukce **„=“**(Assign), **R** (Reset) a **S** (Set). Tyto instrukce změní hodnotu bitové proměnné za instrukcí. Příkaz Assign předá hodnotu současného RLO do proměnné. Příkaz Reset změní hodnotu proměnné na nulu, pokud je současné RLO jedna. Příkaz Set je opakem Reset, pokud je současné RLO jedna, změní hodnotu bitové proměnné na jedna. Příkazy set a reset se používají k ovládání tzv. klopného obvodu, který v remanentí části paměti drží poslední hodnotu.

Dále jsou v STL příkazy, které změní současné RLO, příkaz **NOT** (Negate RLO) zneguje současné RLO (změní jeho hodnotu na opačnou, tedy nulu na jedna a naopak), příkaz **SET** (Set RLO (=1)) změní RLO na jedna, příkaz **CLR** (Clear RLO (=0)) změní RLO na nulu.

Příkazy **FN** (Edge Negative) a **FP** (Edge Positive) hlídají náběžnou případně sestupnou hranu současného RLO. Pro fungování tohoto příkazu je nutné mu přidělit bitovou paměť, která si bude pamatovat poslední vstup, musí to tedy být paměť, která není použita jinde v programu a současně nesmí patřit do dočasné paměti.

Příkazy porovnávací porovnávají dvě poslední načtená čísla v Akumulátorech procesoru. Jsou různé příkazy pro různé typy proměnných. Můžeme porovnávat Integer, Double Integer nebo Float. Pro správné porovnání je zapotřebí vybrat správný příkaz a mít obě proměnné stejného typu. Příkaz obsahuje porovnávací specifikaci (>, <, =) a formát načtených proměnných (I znamená integer, D znamená double integer nebo R znamená float (Real)).

Operace sloužící ke konverzi proměnných změní číslo načtené v akumulátoru na odpovídající číslo jiného datového typu. Nejčastěji se používají příkazy **ITD** (integer to double), který změní datový typ šestnácti bitového integeru na třiceti dvou bitový, **DTR** (double to floating-point nebo real), který změní třiceti dvou bitový integer na real a **RND**, které zaokrouhlí číslo typu real na double integer.

Pro práci s **čítačem** je nutné použít jeden z dvou set padesáti šesti čítačů, které má PLC k dispozici nebo si naprogramovat vlastní. Pro čítače PLC existují následující instrukce.

- Instrukce **L** (load) načte integerovou hodnotu, se kterou bude PLC počítat.
- Instrukce **S** (set) dopíše do čítače aktuálně načtenou hodnotu, při náběžné hraně RLO.
- Instrukce **R** (reset) vymaže aktuální hodnotu čítače, při náběžné hraně RLO.
- Instrukce **CU** (Counter up) inkrementuje dopsaný čítač, při náběžné hraně RLO.
- Instrukce **CD** (Counter down) dekrementuje dopsaný čítač, při náběžné hraně RLO.

Příklad ovládání čítače PLC: [9]

```
L     14     //Načtení přednastavené hodnoty čítače.
A     I 0.1  //Přednastavení čítače při náběžné hraně vstupu I 0.1.
S     C1     //Do čítače 1 ulož přednastavenou hodnotu při náběžné hraně.
A     I 0.0  //Náběžné hrana vstupu I 0.0.
CD    C1     //Odečti z čítače 1 jedničku při náběžné hraně RLO
AN    C1     //Pokud je v čítači 1 aktuální hodnota nula.
=     Q 0.0  //Nastav hodnotu výstupu Q 0.0 na jedna, pokud je v čítači 1 hodnota nula.
```

Instrukce skoku, jsou v programovacím jazyku STL velmi důležité. Zastupují instrukci „if“ vyšších programovacích jazyků. Nejčastějšími používanými skoky jsou skoky na základě současného RLO. Příkaz **JC** provede skok na návěstí dopsané za instrukci při RLO jedna. Příkaz **JCN** provede skok na návěstí dopsané za instrukci při RLO nula. Dále existují skoky bezpodmínečné nebo skoky na základě stavových bitů popřípadě skoky pro různé číselné hodnoty v akumulátoru.

Příkaz **Loop** vytváří programovou smyčku s jednodušším ovládáním. Instrukce skočí na návěstí v případě, že v akumulátoru jedna je jiná hodnota než nula, spolu se skokem dekrementuje akumulátor o jedna.

Matematické funkce pracují s posledními dvěma načtenými čísly v Akumulátorech procesoru. Jsou různé příkazy pro různé typy proměnných. Můžeme počítat s Integer, Double Integer nebo Float. Pro správné porovnání je zapotřebí vybrat správný příkaz a mít obě proměnné stejného typu. Příkaz obsahuje specifikaci operace (+, -, ·, /) a formát načtených proměnných (I znamená integer, D znamená double integer nebo R znamená float (Real)). Pro čísla ve formátu Real je navíc možné využít pokročilé příkazy, kterými jsou druhá mocnina nebo odmocnina, exponenciála, logaritmus nebo goniometrické funkce.

Instrukce načtení a uložení patří mezi nejpoužívanější příkazy v programu. Používají se k matematickým funkcím, k ovládání čítačů i časovačů nebo k pouhému kopírování proměnné. Instrukce **L** (Load) načte proměnnou dopsanou za instrukcí do akumulátoru jedna a bývalou hodnotu akumulátoru jedna přepíše do akumulátoru dva. Příkaz **T** (Transfer) zkopíruje hodnotu akumulátoru jedna do proměnné dopsané za instrukcí. Kromě těchto dvou základních příkazů ještě existují příkazy, které pracují s konkrétním akumulátorem nebo které mění hodnoty akumulátorů.

Instrukce posunu a rotace pracují s hodnotou v akumulátoru nebo s konkrétní proměnnou.

Pro práci s **časovači** je nutné použít jeden z dvou set padesáti šesti časovačů, které má PLC k dispozici nebo si naprogramovat vlastní. Pro časovače PLC existují následující instrukce.

- Instrukce **L** (load) načte dobu trvání, po kterou bude časovač časovat (S5Time nebo Time).
- Instrukce **R** (reset) vymaže aktuální hodnotu časovače, při náběžné hraně RLO.
- V závislosti na aplikaci časovače je nutné zvolit správný příkaz běhu časovače.
- Příkaz **SD** (On-Delay Timer) vytvoří časovač, který přejde do jedničky, pokud je RLO jedna nepřetržitě po dobu uloženého času.
- Příkaz **SE** (Extended Pulse Timer) vytvoří časovač, který při náběžné hraně RLO vytvoří puls časovače po dobu chodu časovače, nezávisle na RLO vstupu.
- Příkaz **SF** (Off-Delay Timer) vytvoří časovač, který po sestupné hraně RLO zanechá signál časovače v jedné po dobu chodu časovače.
- Příkaz **SP** (Pulse Timer) vytvoří časovač, který při náběžné hraně RLO vytvoří puls časovače po dobu chodu časovače a při RLO vstupu.
- Příkaz **SS** (Retentive On-Delay Timer) vytvoří časovač, který přejde do jedničky, po náběžné hraně RLO a po dopočtení uloženého času.

```
      O     "V502".B_opened
      O     "V602".B_opened
      O     "V702".B_opened
      L     s5t#5s
      SD    "Tag_7"

      A     "Tag_7"
      JCN   nemale_Q
      L     5000
      T     "Tag_4"

nemale_Q : NOP 0
```

Obr. 3.2-1

Na obrázku 3.2-1 je pomocí příkazu SD vytvořen On-Delay Timer. Podmínka Tag_7 bude splněna, pokud bude pět sekund nepřetržitě otevřen alespoň jeden ze tří technologických prvků V502, V602, V702.

3.3 Programové bloky standartní knihovny SIDAT

Funkční bloky S0_MOTOR[FB200], S1_VALVE[FB201], S2_ANALOG[FB202], S3_PID_C[FB203] a S4_DIGITAL[FB204] vykonávají programové funkce, které se vztahují k řízení nebo přijímání hodnot od reálné technologie. Tyto bloky jsou vytvořeny a používány firmou SIDAT s.r.o..

Dají se nastavit na přijímání hodnot ze vstupních hodnot těchto funkcí nebo se vstupní hodnota může dát nepřímým způsobem, pomocí tzv. pointeru, což je programový odkaz na skupinu dat o určité velikosti s libovolným formátem, může se jednat o datový blok, paměťové markery, vstupní i výstupní data nebo i dočasnou paměť právě vykonávaného programu. Při obsluze vytvořeného programu se z velké většiny v těchto blocích využívají pointery. Zákazník vyžadoval možnost měnit vstupní i výstupní data programů na základě vytvoření dokumentu v programu Excel firmy Microsoft Office, který přepíše datové bloky veškeré technologie a změní ukazatele (pointery) na nové používané vstupy a výstupy. Pro splnění tohoto požadavku jsem uzpůsobil již vytvořený dokument programu Excel tak, aby fungoval pro prostředí TIA Portal. Téměř všechny technologické prvky jsou řízeny pomocí pointerů, pouze regulace typu PID jsou řízeny přímým přijímáním ze vstupních hodnot a zapisováním na hodnoty výstupní.

Name	Description	Type	READY	FB OPEN	FB CLOSE
V102	V102 Valve	2	M 0.0	Q 0.3	I 2.1

LOCAL	OPEN	CLOSE		MovingDelay	EndposDelay	ClosingDelay
M 0.0	Q 0.3	M 0.1		10000	0	3000

Obr. 3.3-1

Data vyplněná do tabulky v programu Excel se pomocí funkcí programu přemění na data, která se v rámci struktury shodují se strukturou externích zdrojových souborů. Obrázek 3.3-1 zobrazuje vyplněnou tabulku Excelu pro technologický prvek „V102“, obrázek 3.3-2 ukazuje výsledný sloupec dat určený pro import do CPU.

```
DATA_BLOCK "V102"
{ S7_Optimized_Access := 'FALSE' }
VERSION : 0.0

"S1_VALVE"
BEGIN
  OBJects := 0;
  EXECution := 0;
  BegOfMerkers := P#P  0.0;
NUMBER := 0;
A_INPUT_1 := DW#16#83000000;
A_INPUT_2 :=  DW#16#82000003;
A_INPUT_3 :=  DW#16#81000011;
A_INPUT_4 :=  DW#16#83000000;
A_OUTPUT_1 :=  DW#16#82000003;
A_OUTPUT_2 :=  DW#16#83000001;
A_OUTPUT_3 := DW#16#83000001;
TypE_OBJECT := B#16#2;
```

Obr. 3.3-2

Pro import je nutné vytvořit textový soubor s příponou „*.db“. Obsah souboru je tvořen záznamy z programu Excel. Soubor „*.db“ pak použijeme jako zdrojový soubor v prostředí TIA pro vygenerování datového bloku. Po dodržení tohoto postupu se změní datový blok s danou symbolickou adresou a uloží se do něj nově vyplněné hodnoty.

V případě programového řízení regulátorů typu PID je využit funkční blok CONT_C, což je standartní technologický blok SIEMENS S7 knihovny, který se zabývá řízením regulátorů typu PID a navíc má několik speciálních funkcí. Přímo z programovacího prostředí TIA se tento blok dá konfigurovat přímým vkládáním parametrů a limit, vybráním správného typu regulace a jejích parametrů a povolením či zakázáním možnosti manuálního módu. Funkce během svého provádění ukládá data a je schopna vytvářet i grafy změn důležitých hlídaných parametrů.

3.3.1 Programové bloky zajišťující dílčí funkce a řízení technologických prvků

S0_MOTOR[FB200]

Funkční blok, který obsluhuje technologické prvky motory, v tomto případě čerpadlo.

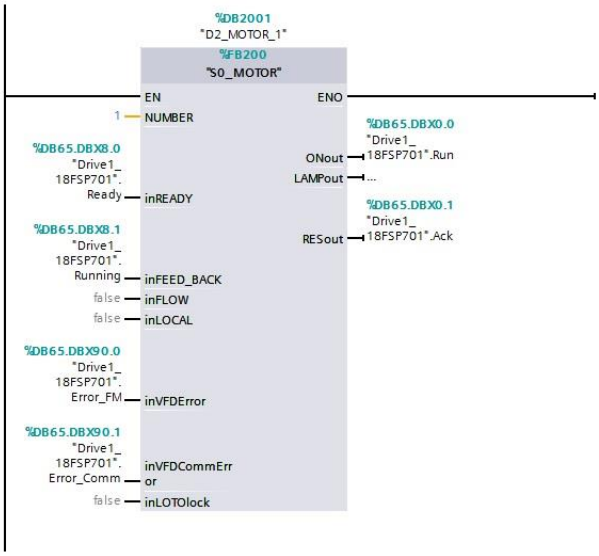
Vstupy:

- NUMBER: číslo technologického prvku, v tomto případě motoru čerpadla ve formátu Integer
- inREADY: digitální vstup, elektrický signál, motor čerpadla je připraven ve formátu Boolean
- inFEED_BACK: digitální vstup, elektrický signál, motor čerpadla je v provozu ve formátu Boolean

- inFLOW: digitální vstup, elektrický signál ve formátu kapalina proudí motorem čerpadla – Boolean
- inLOCAL: přepínač místního provozu ve formátu Boolean
- inVFDError: parametr využívaný pro motory typu VFD, čerpadlo je řízeno regulační smyčkou
- inVFDCommError: parametr využívaný pro motory typu VFD, čerpadlo je řízeno regulační smyčkou
- inLOTOlock : softwarové vyřazení motoru ve formátu Boolean

Výstupy:

- ONout: digitální výstup, zapnutí motoru ve formátu Boolean, pokud není řízen otáčkami, motor se roztočí, pokud ano, musí být zároveň požadavek na otáčky motoru
- LAMPout: digitální výstup, chybové hlášení motoru pro signalizaci se signalizací chyby (různé frekvence mají různý význam) ve formátu Boolean
- RESout: reservní digitální výstup ve formátu Boolean



Obr. 3.3-3

S1_VALVE[FB201]

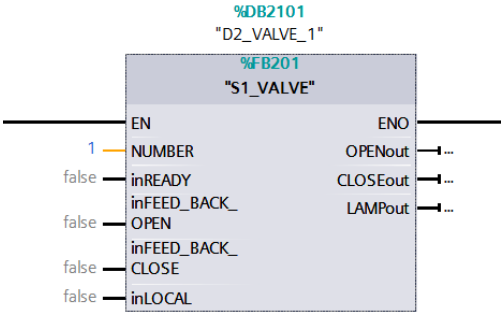
Funkční blok, který obsluhuje technologické prvky ventily

Vstupy:

- NUMBER: číslo technologického prvku, v tomto případě ventilu, ve formátu Integer
- inREADY: digitální vstup, elektrický signál, ventil je připraven, ve formátu Boolean
- inFEED_BACK_OPEN: digitální vstup, elektrický signál ventil je otevřen, ve formátu Boolean
- inFEED_BACK_CLOSED: digitální vstup, elektrický signál- ventil je uzavřen ve formátu Boolean
- inLOCAL: přepínač místního provozu ve formátu Boolean

Výstupy:

- OPENout: digitální výstup, ventil otevřen ve formátu Boolean
- CLOSEout: digitální výstup, ventil uzavřen ve formátu Boolean
- LAMPout: digitální výstup, chybové hlášení ventilu pro signalizaci se signalizací chyby (různé frekvence mají různý význam) ve formátu Boolean



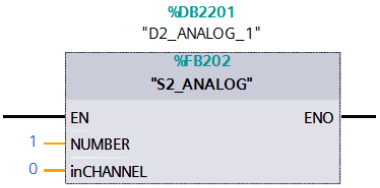
Obr. 3.3-4

S2_ANALOG[FB202]

Funkční blok, který obsluhuje technologické prvky analogická měření

Vstupy:

- NUMBER: číslo prvku, v tomto případě analogového vstupního signálu ve formátu Integer
- inCHANNEL: číslo periferijního Wordu, na kterém přichází daný analogový signál ve formátu Integer



Obr. 3.3-5

S3_PID_C[FB203]

Funkční blok, který reguluje činnost regulovatelných technologických komponentů. V tomto případě mění rychlost otáčení čerpadla a otevření regulačního ventilu páry.

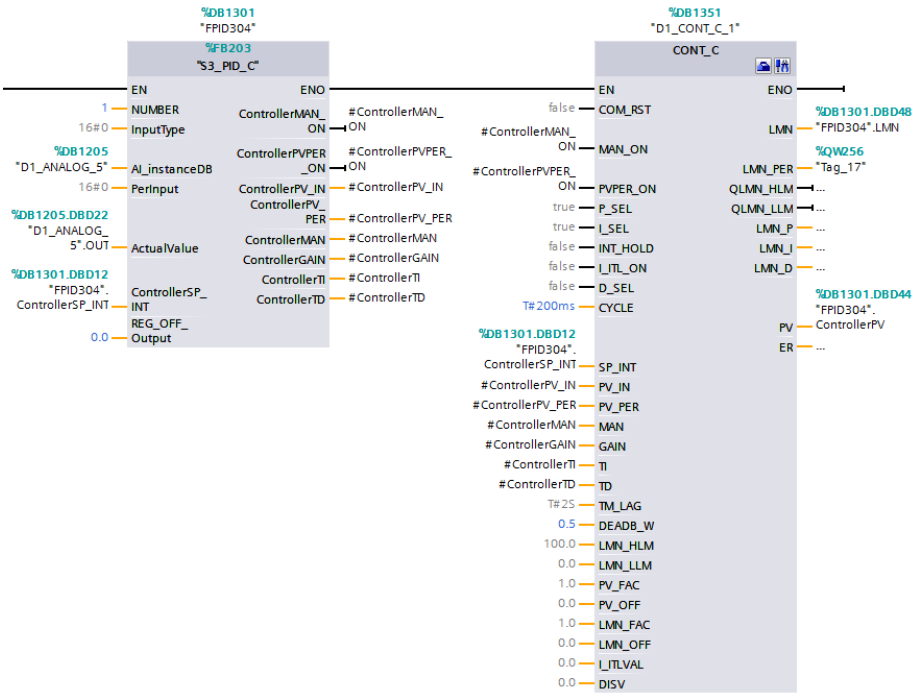
Vstupy:

- NUMBER: číslo prvku, v tomto případě řízení PID softwaru ve formátu Integer
- InputType: typ vstupu, různé hodnoty říkají, odkud si program má vzít vstupní hodnotu ve formátu Byte
- AI_instanceDB: Instanční DB, ze kterého se vezme vstup při správné hodnotě InputType ve formátu Boolean
- PerInput: Vstup z periferie, číslo vstupního Wordu ve formátu Word
- ActualValue: Aktuální hodnota vstupu v případě přímé hodnoty ve formátu Real
- ControllerSP_INT: hodnota setpointu ve formátu Real
- REG_OFF_Output: Výstupní hodnota v případě vypnuté regulace ve formátu Real

Výstupy:

- ControllerMAN_ON: Regulace je v manuálním modu, nebo je neaktivní ve formátu Boolean
- ControllerPVPER_ON: Aktuální hodnota je brána z periferie ve formátu Boolean
- ControllerPV_IN: Aktuální hodnota vstupu v případě přímé hodnoty ve formátu Real

- ControllerPV_PER: Aktuální hodnota vstupu v případě, že je používána hodnota z periferie ve formátu Real
- ControllerMAN: Substituční hodnota, která má být PID výstupu v případě manuálu
- ControllerGAIN: P parametr PID řízení
- ControllerTI: I parametr PID řízení
- ControllerTD: D parametr PID řízení



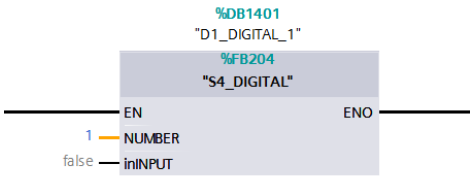
Obr. 3.3-6

S4_DIGITAL[FB204]

Funkční blok, který obsluhuje technologické prvky digitální senzory popřípadě digitální výstupy.

Vstupy:

- NUMBER: číslo prvku, v tomto případě digitálního vstupního signálu ve formátu Integer
- inINPUT: připravenost prvku, použití tohoto technologického prvku ve formátu Boolean



Obr. 3.3-7

S_GROUP[FB1000]

Jedná se o funkční blok, který je volán v každém organizačním bloku, jak v hlavních cyklech, tak v přerušeních. Na základě přerušení, ve kterém je volán se vykonávají programy odpovídajících technologických prvků. V přerušení 200ms se vykonávají programy analogových vstupů a PID regulátorů, v přerušení 100ms se vykonávají programy digitálních vstupů, motorů a ventilů.

D_GROUP[DB1000]

Je datovým blokem, který je volán s funkčním blokem S_GROUP. Obsahuje příkazy k programovému zapnutí technologických prvků patřících do skupiny prvků v S_GROUP, jejich hodnoty, chybová a stavová hlášení jednotlivých prvků.

S_ACK&Mess[FC199]

Je podprogramem S_GROUP, jejím úkolem je generovat chybová hlášení technologických prvků ve skupině S_GROUP. Nemá vstupní ani výstupní parametry. Jeho potenciál jsem v programu plně nevyužil.

TONR[FC191] (timer on delay)

Program obsluhy časovačů v procedurách, jeho autorem jsem já na základě obdobného programu standartní knihovny S7 pro PLC řady 300 a 400, který již ale není podporován pro PLC řady 1500. Na základě fungování původní funkce knihovny S7 jsem tedy vytvořil její kopii pro PLC řady 1500.

Vstupy:

- TMR_EN: aktivace počítání časovače ve formátu Boolean
- RESET: vynulování aktuální hodnoty časovače ve formátu Boolean
- PV: hlídaná hodnota časovače ve formátu Double Integer
- DELTA_T: změna aktuální hodnoty časovače během jednoho cyklu ve formátu Integer

Výstupy:

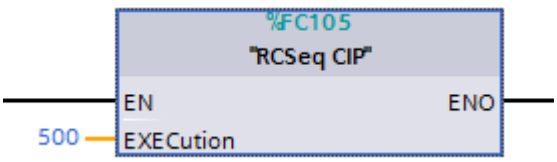
- Q: aktuální hodnota časovače dosáhla hlídané hodnoty ve formátu Boolean
- Parametry, které se používají jako vstup i jako výstup:
- ET: aktuální hodnota časovače ve formátu Double Integer

```
CALL  "TONR"
   TMR_EN  :=#ITF.TIM_1_release
   RESET   :=#ITF.TIM_1_reset
   PV      :=#RCP_element_mem.T_PAR[1]
   DELTA_T :=#EXECution
   Q       :=#ITF.TIM_1_result
   ET      :=#Act_T_PAR[1]
```

Obr. 3.3-8

3.3.2 Programové bloky zajišťující automatické řízení

Sekvenční program, který obsluhuje procedury a který zajišťuje změny technologických prvků na základě konaného čištění je volán s přerušením 500ms. Sestává z bloku RCSeq CIP [FC 105] a bloku FB_RCP_SEQUENCER [FB 212]. Tyto bloky jsou vytvořeny a používány firmou SIDAT s.r.o..



Obr. 3.3-9

```
CALL  "FB_RCP_SEQUENCER", "DB_Seq CIP_5"
   RELEASE   :=            //"Release"
   START     :=
   HOLD      :=
   RESET     :=
   ACK       :=
   RELEASED  :=
   SEQRUN    :=
   HELD      :=
   FINISHED  :=
   RECIPE_NR :=
   STEP_NR   :=
   PROC_NR   :=
```

Obr. 3.3-10

Vstupní a výstupní parametry funkčního bloku FB_RCP_SEQUENCER nejsou využity, neboť do odpovídajících parametrů zapisuje operátorský panel. Při počátečních testech softwaru bez operátorského panelu jsem využíval těchto vstupů. Později bylo možné testovat software CPU a operátorského panelu zároveň a pak nebylo nutné ani žádoucí vstupní parametry bloku FB_RCP_SEQUENCER používat.

Na základě programu čištění, který vybere operátor, sekvenční program zavolá příslušnou proceduru.

Procedura je částí programu pro automatické řízení technologického procesu. Na základě vstupních parametrů, dává povel jednotlivým technologickým objektům a tím řídí proces. Při splnění příslušné přechodové podmínky je procedura hotova a sekvenční program bude volat další proceduru v pořadí. Procedury jsem vytvořil podle návrhu FDS.

Procedury jsou volány nepřímo na základě vyplnění receptu, což je datový blok, se kterým sekvenční program pracuje. Sekvenční program při změně kroku, tedy při volání nové procedury zkopíruje data z receptu do datového bloku sekvence. Na základě dat z receptu je volána další procedura v pořadí. Data z receptu upřesňují podmínky dané procedury, kterými jsou teplota, rychlost proudění a přechodové podmínky pro přechod do další procedury nebo jiné informace v závislosti na volané proceduře. Rozhraní procedur je možné najít v příloze.

Procedury obsahují následující části:

- Přečtení a zápis důležitých proměnných, uložených v datovém bloku sekvence, do dočasných parametrů procedury
- Počáteční krok, což je část programu, která se vykonává pouze při prvním volání nové procedury
- Používání programových časovačů realizovaný blokem TONR, viz Funkční blok TONR

```
//Timer TIM01 - Condition 34 -
      AN    #itf.LastScan
      AN    #itf.Held
      AN(
      L     %DID130
      L     DINT#0
      ==D
      )
      =     #itf.TIM_1_release

      A     #itf.FirstScan
      =     #itf.TIM_1_reset
```

Obr. 3.3-11

Příkaz „#itf.TIM_1_release“ spouští časování časovače realizovaného blokem TONR. K jeho spuštění dojde, když není poslední průchod procedury, když není sekvence pozastavena, a když je v receptu pro časovač nenulová hodnota časování.

Příkaz „#itf.TIM_1_reset“ ukládá do časovače realizovaného blokem TONR nulovou výchozí hodnotu a je připraven k dalšímu chodu. K jeho spuštění dochází v prvním volání nové procedury.

- Aktivaci technologických prvků na základě podmínek k jejich spuštění

```
AN    #itf.LastScan
AN    #itf.Held
A     #itf.index[5]
=     "AON_V102"
```

Obr. 3.3-12

Na obrázku 3.3-12 je příklad spuštění technologického prvku v proceduře. Pro větší přehlednost se zákazník rozhodl nevyužít možnosti zapnutí technologického prvku z datového bloku D_GROUP, ale nepřímo pomocí markeru, na který odkazuje pointer v datovém bloku, který spouští technologický prvek. V případě na obrázku se spouští daný technologický prvek „ventil 102“, pokud není poslední průchod procedurou, není pozastavena sekvence a pokud je povoleno použití horké vody.

- Sledování zásadních proměnných a v případě změn, informování operátora (warning tj. upozornění), nebo i zastavení procesu (alarm tj. chyba).

```
AN    #itf.LastScan
=     "Recipe_Control".Control_Enable[1]//TIM1
=     "Recipe_Control".Control_Enable[2]//F2
=     "Recipe_Control".Control_Enable[3]//T1
=     "Recipe_Control".Control_Enable[7]//LLS RW
=     "Recipe_Control".Control_Enable[13]//HLS CAUSTIC
```

Obr. 3.3-13

V proceduře se pod podmínkou, že není poslední přečtení kódu, aktivuje hlídání daných upozornění a chyb podle požadavků dané procedury.

```
A     "Recipe_Control".Control_Enable[11]
A     "OUT_LS704"
=     "Recipe_Control".Control_Result[11]
```

Obr. 3.3-14

Ze sekvenčního funkčního bloku se volá funkce Message, která na základě aktuálních stavů proměnných a aktivovaných hlídání upozornění a poruch, generuje bity upozornění a poruch.

- Sledování proměnných, které zajišťují přechod do další procedury.

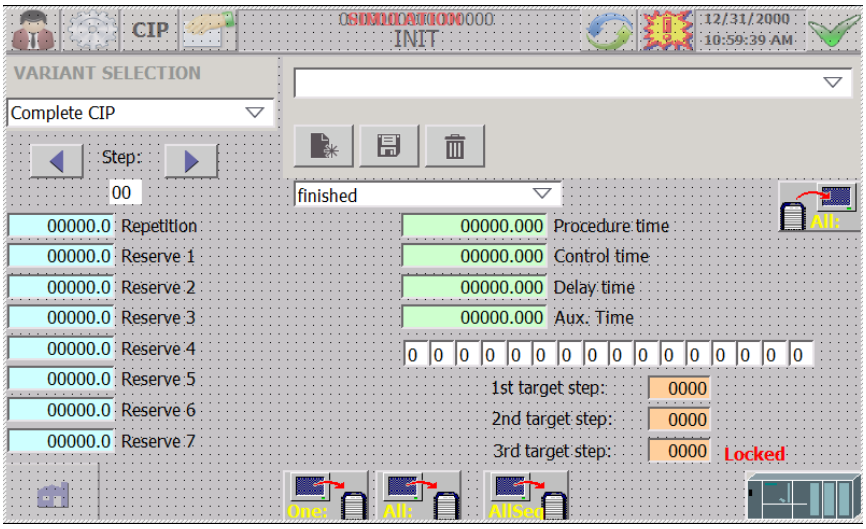
```
AN    #itf.FirstScan
AN    #itf.Held
A     #itf.TIM_1_result
=     #itf.TransNext
```

Obr. 3.3-15

Na obrázku 3.3-15 je příklad přechodu do dalšího kroku za podmínek, že není první přečtení procedury, není zastavena sekvence a doběhl zadaný čas časovače realizovaného blokem TONR.

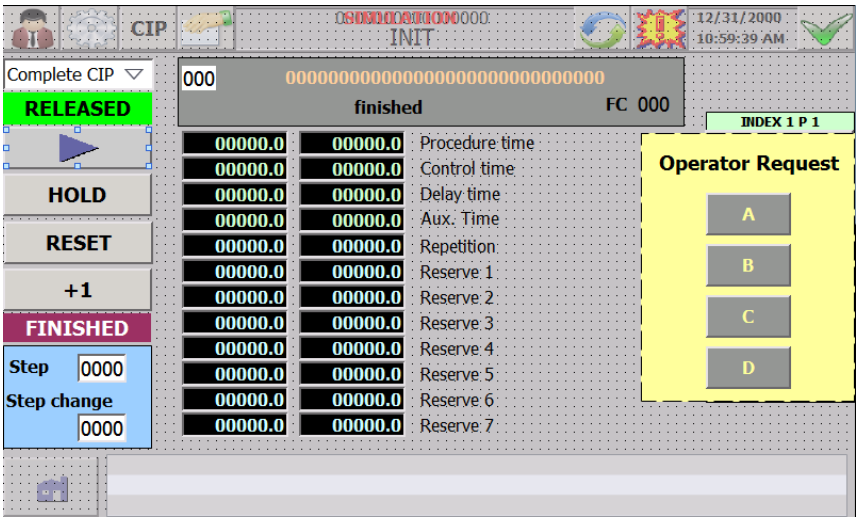
- Poslední krok, což je část programu, která se vykoná pouze při posledním volání procedury, tedy po splnění přechodové podmínky.
- Přepis hodnot zpět z dočasných parametrů procedury do datového bloku sekvence.

Receptura procesu je tvořena sledem procedur a jejich parametry. Receptura se importuje do operátorského panelu jako dokument typu „.csv“ (comma-separated values). Data je možné sledovat a měnit i přímo v operátorském panelu. Před spuštěním procesu čištění se vybere správná receptura v operátorském panelu a varianta tohoto procesu A, B, C nebo D. Varianta pak udává kombinaci procedur při použití správných čistících látek na bázi zásady, na bázi kyseliny popřípadě vody a při správné intenzitě. Každá varianta tedy obsahuje jiný sled procedur. V praxi to znamená, že každá procedura má navolené bity A, B, C a D a pokud je bit zvolené varianty u dané procedury 0, procedura bude během procesu sanitace přeskočena. Následně se spuštěním speciálního skriptu z panelu importují data do datového bloku centrální procesorové jednotky.



Obr. 3.3-16

Přepnutím stanice do automatu a zvolením tlačítka start se spustí zvolený proces čištění.



Obr. 3.3-17 tlačítko start má podobu symbolu šipky vpravo

4 VYTVOŘENÍ SW SIMULUJÍCÍHO REÁLNOU TECHNOLOGII

4.1 Výhody simulačního softwaru

Testování v režimu simulace je vhodný způsob vyzkoušení funkcí řídicího systému bez reálné vazby na řízenou technologii. Pomocí simulace se výrazně zkrátí čas uvedení zařízení do provozu, sníží se riziko možných škod způsobených chybami v programu, v neposlední řadě pak šetří suroviny případně další materiály nutné pro zkoušky a zkušební provoz zařízení. Při komplexnějších simulacích je možné odhalit slabá místa a optimalizovat systém dříve, než je systém vůbec realizován. Zkoumání chování již reálného systému je ekonomicky nevýhodné, často nebezpečné a existuje i riziko poškození zařízení nebo produktu. [4]

Simulace nejen části, ale i celé technologie se využívá v přístupu Industry 4.0, kde se za použití metod simulace zkoumají možnosti, průchodnost a efektivita celých linek.

4.2 Tvorba simulačního softwaru

Vyzkoušení programového vybavení v režimu simulace bylo požadavkem zákazníka.

Během simulace procesu se na základě povelů pro technologické prvky vytváří programové odezvy, což je předpokládané chování celé technologie. Například při povelu pro otevření ventilu simulační program zareaguje způsobem, že nastaví zpětné hlášení ventilu jako otevřen. Zapnutí čerpadla má za následek zpětné hlášení čerpadla a zvýšení hodnoty senzoru průtoku nebo při otevření regulačního ventilu páry se zvýší hodnota senzoru teploty. Simulace je aktivní zapnutím simulační globální proměnné.

V každé proceduře se kontroluje přechodová podmínka podle zadání FDS

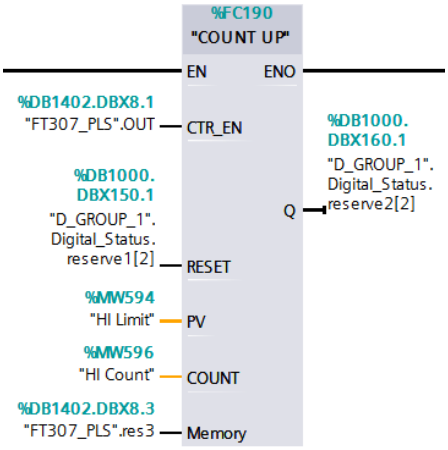
V každé proceduře se kontrolují stavy technologie podle zadání FDS a v případě, že je splněna podmínka pro generování hlášení varování, vypíše se na operačním panelu varování pro operátora, v případě, že je splněna podmínka pro generování hlášení alarmu, vypíše se na operačním panelu alarm, který pozastaví nebo přeruší proces sanitace.

Na základě kombinace otevřených ventilů tj. otevření vodní, kyselinové popřípadě louhové cesty pak simulační program mění programovou hodnotu konduktivity a tím může splnit přechodovou podmínku pro přechod do další procedury. Při zapnutí motoru čerpadla simulační program změní hodnotu aktuálního průtoku a kontinuálně zvyšuje počítadlo proteklého množství. Proteklé množství může být také podmínkou pro přechod do další procedury. Další podmínkou přechodu může být doběhnutí časovače, jehož podmínku na vstupu splníme při simulaci technologickou odezvou, proto není třeba pro časovače dělat simulační program.

```
A     "V101".outOPEN
A     "V102".outOPEN
O     "V701".outOPEN
O     "V601".outOPEN
O     "V501".outOPEN
O     "V303".outOPEN
A     "P203".B_Run
A     "Clock_0.5Hz"
=     "DIGI_2_SIMU"
```

Obr. 4.2-1

Na obrázku 4.2-1 je kód simulace počítadla proteklého množství. Při otevření příslušné kombinace ventilů a současně provozu motoru čerpadla se kontinuálně nastavuje bit průtoku. Náběžná hrana tohoto signálu zvyšuje hodnotu čítače objemu proteklého množství, který je realizován mnou navrhlou funkcí COUNT UP viz obrázek 4.2-2



Obr. 4.2-2

```
      AN    "V502".B_opened
      AN    "V602".B_opened
      AN    "V702".B_opened
      L     s5t#5s
      SD    "Tag_1"

      O     "V502".B_opened
      O     "V602".B_opened
      O     "V702".B_opened
      L     s5t#5s
      SD    "Tag_7"

      A     "Tag_7"
      JCN   nemale_Q
      L     5000
      T     "Tag_4"

nemale_Q : NOP 0

      A     "Tag_1"
      JCN   nevelke_Q
      L     1000
      T     "Tag_4"
nevelke_Q : NOP 0
```

Obr. 4.2-3

Na obrázku 4.2-3 je kód simulace konduktivity. Při otevření příslušné kombinace ventilů se po časovém zpoždění nastaví nová hodnota konduktivity, čímž se splní podmínka pro přechod do dalšího kroku.

5 TESTOVÁNÍ V REŽIMU SIMULACE A UVÁDĚNÍ DO PROVOZU

5.1 Postup simulace

Zapnutí režimu simulace a vyzkoušení všech procedur CIP stanice i jednotlivých receptur. Vyzkoušení zahrnuje i vizualizaci procesu na operátorském panelu. Při použití vizualizace jsou zkoušky simulací dostatečně názorné i pro budoucí operátorskou obsluhu a pro technologa, kteří nemusí být odborníky v oblasti funkce a programování PLC SIEMENS.

Vyzkoušení reakce řídicího systému na vybranou událost procesu. Seznam událostí, které budou předmětem zkoušek, musí zahrnovat takzvané kritické body, což jsou zejména pokles konduktivity, ztráta průtoku, nárůst tlaku, minima v tancích a další. Postupně se vyzkouší v simulačním provozu příslušné reakce v odpovídajících procedurách. Nejčastějšími reakcemi řídicího systému bývá upozornění obsluhy výpisem odpovídajícího hlášení na operátorském panelu (warning), bezpečné zastavení procesu a výpis odpovídající poruchy na operátorském panelu (alarm), popřípadě otevření nebo zavření ventilu.

Například nedosažení teploty:

V režimu simulace v procedurách s cirkulací změníme hodnotu teploty tak, aby překročila kontrolované limity. Sledujeme reakci řídicího systému. Očekávanou reakcí dle FDS je po uběhnutí kontrolního času zastavení procesu a výpis hlášení v případě příliš vysoké teploty. Pro nízkou teplotu se po uplynutí kontrolního času pouze vypíše hlášení nízké teploty.

5.2 Uvádění do provozu

Při uvádění do provozu nejdříve zkontrolujeme zapojení řídicího systému PLC podle dokumentace.

Obr. 5.2-1

Řídicí systém dáme pod napájení a zkontrolujeme, že je aktivní. V případě S7 1500 zkontrolujeme vložení paměťové karty, bez paměťové karty není možné nahrát program.

Navážeme komunikaci PLC s počítačem, který obsahuje prostředí TIA. Zprovozníme komunikaci v prostředí TIA portálu a přehrajeme hardwarovou konfiguraci. Po nahrání hardwarové konfigurace se pokusíme dát centrální procesorovou jednotku do stavu RUN. V případě, že hlásí v diagnostickém bufferu poruchu, odstraníme ji. Pokud hlásí jednotka CPU poruchu, svítí červená LED dioda. Po odstranění poruch a po přechodu do stavu RUN zhasnou všechny červené LED diody na PLC.

Vyzkoušíme pomocí Microsoft programu „Command Table“ správné nastavení IP adresy. Do příkazového řádku napíšeme příkaz ping a IP adresu centrální jednotky. Při správném nastavení by měla jednotka CPU odpovídat. Pro funkci příkazu ping musí mít náš počítač IP adresu ve shodném rozsahu, se stejnou maskou a centrála i počítač musí mít jedinečnou IP adresu v rámci sítě.

Nyní by mělo být možné nahrát program. Zkompilujeme celý hotový program, který jsme simulovali, a nahrajeme ho do Centrální procesorové jednotky. Zkusíme dát jednotku do stavu RUN. Pokud se CPU nedaří dát do stavu RUN, pomocí diagnostického bufferu vyhodnotíme problém a pro případ testování můžeme vložit poruchové organizační bloky, detailně se této problematice věnuji v kapitole programování v prostředí TIA. Pokud se podaří, nahrajeme program i do operátorského panelu. Zkontrolujeme komunikaci mezi panelem a PLC.

Následují signálové testy. Postupně vyzkoušíme senzory a jejich odezvy na vstupních kartách PLC a v programu. Zkontrolujeme zapojení a rozsahy analogových snímačů tedy vstupů. Zkontrolujeme hardwarovou konfiguraci oproti správné konfiguraci zapojených snímačů. Dokumentaci v tomto případě vytvořil zákazník a sám zapojení zkontroloval. Signálové testy odhalili, že zkontrolované zapojení nebylo bez chyb a museli jsme tak upozorňovat na nedostatky v zapojení. Na operátorském panelu by měly být zobrazeny reálné hodnoty teplot, vodivosti, průtoku a tlaku.

Jednotlivě vyzkoušíme akční prvky. V tabulce proměnných nebo v servisním režimu z operátorského panelu nastavíme hodnoty výstupů, které ovládají akční prvky a sledujeme otevírání ventilů a spouštění motorů, popřípadě spouštění dalších akčních prvků.

Pokud všechny signály odpovídají, otestujeme spuštění procesu mytí. Zvolíme vhodnou recepturu a variantu a vyzkoušíme umytí s médiem vodou. Při procedurách porovnáváme funkci zařízení vzhledem k FDS. Vyzkoušíme reakci na poruchové stavy a funkčnost nouzového zastavení.

Většina sanitačních stanic nebývá měněna od doby uvedení do provozu. Stanice bývá nastavena na výchozí hodnoty pro daný provoz. Pokud se podmínky provozu změní, například zavedením nového výrobku, topologií zařízení nebo čisticích prostředků je potřeba upravit parametry receptur pro nové podmínky. Operátoři mohou zoptimalizovat zavedený systém pomocí změn některých důležitých parametrů, jakými jsou

- Teplota a konduktivita nebo koncentrace louhových tanků (ta bývá někdy příliš vysoká)
- Hlídání správných parametrů podávací linie před samotným začátkem čištění
- Nastavení vratné konduktivity nebo koncentrace a teplotního čidla
- Odstranění louhového roztoku a zchlazení před sterilizací nebo po sterilizaci
- Síla sterilizačního činidla a délka působení

Nakonec se shromáždí změny po monitorování, zdokumentují a ověří se splnění zákonných norem a speciálních požadavků zákazníka. [3]

5.3 Zákonné normy hygieny

Po vstupu České republiky do Evropské unie došlo ke zpřísnění požadavků na sanitaci a jejího správného provádění. [1]

Od roku 2006 vstoupilo na našem území v platnost Nařízení o hygieně potravin 852/2004/EC. Toto nařízení patří do takzvaného „hygienického balíčku“ a je právně závazným a vymahatelným ve všech členských zemích Evropské unie.

Ve výrobním procesu se na základě HACCP (Hazard analysis and critical control point) nachází takzvané kritické body, které by mohly vést ke znehodnocení potravin. Proti kritickým bodům se zavádí postupy kontroly, které mají za cíl snížit rizika práce s potravinami ve výrobním procesu z hlediska hygieny na minimum. V praxi to znamená dodržování systémových norem EN ISO 9000 a EN ISO 22000:2006. Stát je povinen hlídat, že výrobci řádně hledají kritické body a adekvátně je řeší.

HACCP sestává z následujících kroků: [1]

- Zpracování postupového diagramu
- Identifikace kritických kontrolních bodů
- Stanovení kritických mezí
- Stanovení systému monitorování stavu v kritických kontrolních bodech
- Stanovení způsobu řešení mimořádných situací
- Zpracování dokumentace systému
- Ověřování účinnosti HACCP

6 ZÁVĚR

Hlavním cílem práce bylo uvedení do provozu plně automatické CIP sanitační stanice, která není plně součástí technologie.

V teoretické části jsem uvedl problematiku CIP sanitačních stanic a vysvětlil jejich fungování, následně jsem popsal maticový zápis zadání FDS a definoval vlastnosti jednotlivých objednaných komponent SIMATIC S7-1500.

V další části jsem popsal vytváření programu v prostředí TIA Portal, popsal jsem nové funkce prostředí a datové typy PLC S7-1500, popsal jsem programové bloky Step 7 a vizualizaci operátorského panelu Win CC v prostředí TIA Portal. Následně jsem charakterizoval použité programové bloky standartní knihovny SIDAT a charakterizoval vlastní vytvořené programové bloky a jejich účel. Také jsem popsal možnosti programování v jazyce STL a definoval nejvíce používané instrukce z instrukční sady tohoto jazyka.

Následně jsem popsal vytváření simulačního softwaru a výhody z něho plynoucích a na příkladech ukázal vlastní vytvořený simulační software určený k testování softwaru za pomoci simulace.

V závěru bylo nutné celý program otestovat v režimu simulace a prověřit tak správné chování řídicího systému a uvést zařízení do provozu ve spojení s řízenou technologií.

7 PŘÍLOHY

itf	"procITF"	0.0
TIM_1_release	Bool	0.0
TIM_1_reset	Bool	0.1
TIM_1_result	Bool	0.2
TIM_2_release	Bool	0.3
TIM_2_reset	Bool	0.4
TIM_2_result	Bool	0.5
TIM_3_release	Bool	0.6
TIM_3_reset	Bool	0.7
TIM_3_result	Bool	1.0
TIM_4_release	Bool	1.1
TIM_4_reset	Bool	1.2
TIM_4_result	Bool	1.3
FirstScan	Bool	1.4
LastScan	Bool	1.5
Held	Bool	1.6
UserHold	Bool	1.7
Msg	Array[16..31] of Bool	2.0
TransNext	Bool	4.0
TransT1	Bool	4.1
TransT2	Bool	4.2
TransT3	Bool	4.3
Request	Bool	4.4
ReqBtnA	Bool	4.5
ReqBtnB	Bool	4.6
ReqBtnC	Bool	4.7
ReqBtnD	Bool	5.0
ACK	Bool	5.1
Aux07	Bool	5.2
Aux08	Bool	5.3
Aux09	Bool	5.4
Aux10	Bool	5.5
Aux11	Bool	5.6
Aux12	Bool	5.7
R_PAR	Array[1..8] of Real	6.0
index	Array[1..16] of Bool	38.0

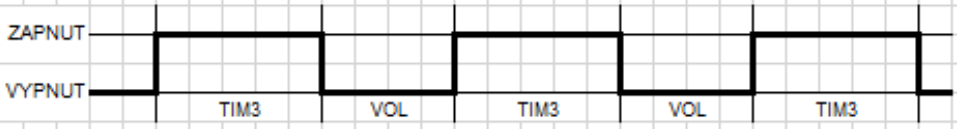
Příloha 7-1- interface procedury

	číslo procedury	0	1	3	5	6	7	8	11	12	13	20	22	25	26	28	31	32	34	36	37	38	39	41	42	43	45	46	47	49	50	67
			O	O	O	O	T	O	O	O	T	O	O	T	O	O	T	T	T	T	T	T	P	P	P	P	P	P	CIP	CIP	CIP	O
		KONEC PROGRAMU	FW do KANÁL	FW do RW	FW do LOUH	FW do KYSE	FW plnění okruhu	RW do KANÁL	RW do LOUH	RW do KYSE	RW plnění okruhu	LOUH do KANÁL	LOUH do RW	LOUH plnění okruhu	KYSE do KANÁL	KYSE do RW	KYSE plnění okruhu	Odčerpání do KANÁL	Odčerpání do RW	Odčerpání do LOUH	Odčerpání do KYSE	Cirkulace obchvat	Cirkulace obchvat, RW	Cirkulace obchvat, LOUH	Cirkulace obchvat, KYSE	Cirkulace přes tank RW	Cirkulace přes tank LOUH	Cirkulace přes tank KYSE	FW plnění do RW	FW plnění do LOUH	FW plnění do KYSE	Přerušení programu
BINÁRNÍ VÝSTUPY	Ventil studené vody		6	6	6	6	6	19	19	19	19																		6	6	6	
	Ventil z tanku použité vody							18	18	18	18												■			■						
	Ventil horké vody		7	7	7	7	7	19	19	19	19																		7	7	7	
	Ventil z tanku LOUH											■	■	■										■			■					
	Ventil z tanku KYSELINA														■	■	■								■			■				
	Podávací čerpadlo		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■					■	■	■	■	■	■	■	■	■	■	
	ruční ventil			1	1	1	1		1	1	1		1	1		1	1		1	1	1	1	1	1	1	1	1	1	■	■	■	
	Ventil propojení okruhu																					■	■	■	■							
	ruční ventil		■	2	2	2	2	■	2	2	2	■	2	2	■	2	2		2	2	2	2	2	2	2	2	2	2				
	Ventil kanál		■					■				■			■			■														
	Ventil do tanku použité vody			4	14	65							4			4			4	14	65					■			■			
	Ventil do tanku LOUH				8				8											8							■			■		
	Ventil do tanku KYSELINA					10				10											10							■			■	
	KOM/ DÁVKOVAT KYSE		30	30	30	30	30	30	30	30	30	30	30	30	30	30	30					48	48	48	48	48	48	48	30	30	30	
	KOM/ DÁVKOVAT LOUH		31	31	31	31	31	31	31	31	31	31	31	31	31	31	31					49	49	49	49	49	49	49	31	31	31	
	Ventil pára		3	3	3	3	3	3	3	3	3	3	3	3	3	3	3					3	3	3	3	3	3	3	3	3	3	
Přechodové podmínky do další procedury	TIM1			62				62					62					20	20	20	20	22	22	22	22	22	22	22				28
	VOL		20	24	20	20	■	24	20	20	■	20	24	■	20	20	■												24	24	24	
	CND růst		21					21				21	21		21	21		21	21													
	HLS RW																												23			
	HLS LOUH																													23		
	HLS KYSE																														23	
	CND pokles			21	21	21			21	21										21	21											
VELIČINA – Hlídané veličiny	TIM1		34		34	34	34		34	34	34	34		34	34	34	34												34	34	34	
	F2		35	35	35	35	35	35	35	35	35	35	35	35	35	35	35					35	35	35	35	35	35	35	35	35	35	
	T1		38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38	38
	T2																					37	37	37	37	37	37	37				
	TIM2																					38	38	38	38	38	38	38				
	CND																					38	38	38	38	38	38	38				
	LLS RW							43	43	43	43												43			43						
	LLS LOUH											45	45	45										45			45					
	LLS KYSE														46	46	46								46			46				
	HLS RW			39									39			39			39							39			39			
	HLS LOUH				41				41											41							41			41		
	HLS KYSE					42				42											42							42			42	
Regulované	PID regulace čerpadla		■	■	■	■	■	■	■	■	■	■	■	■	■	■	■					■	■	■	■	■	■	■	■	■	■	
	PID regulace Regulačního ventilu pára		3	3	3	3	3	3	3	3	3	3	3	3	3	3	3					3	3	3	3	3	3	3	3	3	3	

Příloha 7-2 Maticový zápis FDS

Konstanty:	KONSTANTA 1	Maximání povolená teplota na výtlaku	95 °C
	KONSTANTA 2	Přídavný kontrolní čas	15 s
	KONSTANTA 3	Zpoždění pro nízký průtok	30 s
	KONSTANTA 4	Zpoždění pro vysokou teplotu	10 s
	KONSTANTA 5	Zpoždění pro nízkou teplotu	20 s
	KONSTANTA 6	Zpoždění dávkování desinfekce	60 s
	bit F	Vniřní okruh	

1	je-li bit F = 1
2	je-li bit F = 0
3	je-li T1 > 0 AND teplota na podávací linii < KONSTANTA 1
	start možný: bez další podmínky
4	není-li HLS použité vody
6	požadavek studené vody
7	požadavek horké vody
8	není-li HLS LOUH
10	není-li HLS KYSELINA
14	je-li HLS LOUH a není-li HLS použité vody
20	je-li CND = 0
21	je-li CND > 0
	Pro procedury 1, 2, 3, 4, 8, 9, 10, 14, 15, 16, 20, 21, 22, 23, 26, 27, 28, 29, 32, 33, 34, 35 přechod do dalšího kroku je-li konduktivita > CND.
	Pro procedury 5, 6, 11, 12, 17, 18, 24, 30, 36, 37 přechod do dalšího kroku je-li konduktivita < CND.
22	časovač TIM1 v chodu je-li: (teplota na vratné linii > T2) AND (konduktivita > CND).
	Není-li tato podmínka splněna přerušit počítání času, ale časovač nenulovat!
23	je-li VOL = 0
24	je-li VOL > 0
28	je-li TIM1 = 0, potom je pauza trvalá, přechod do další procedury povelem KROK
34	je-li dosaženo času TIM1: PAUZA + výpis: - PŘEKROČEN POVOLENÝ ČAS KROKU.
	Start možný ihned. Časovač vynulovat a po startu programu kontrolovat dokončení kroku časovačem o hodnotě KONSTANTA 2.
35	je-li průtok na podávací linii < F2 nepřetržitě déle než KONSTANTA 3: PAUZA + výpis: **NÍZKÝ PRŮTOK**
36	je-li teplota na podávací linii > KONSTANTA 1 nepřetržitě déle než KONSTANTA 4: PAUZA + výpis: **- VYSOKÁ TEPLOTA NA VÝTLAKU**
	start programu možný je-li teplota na podávací linii < KONSTANTA 1.
37	při poklesu teploty na vratné linii pod hodnotu T2 po dobu nepřetržitě delší než KONSTANTA 5: výpis: - NÍZKÁ TEPLOTA ČIŠTĚNÍ.
38	je-li dosaženo času TIM2: výpis: - NÍZKÁ KONCENTRACE V OKRUHU.
	Obsluze umožnit pokračování programu (počítání časovače TIM1):
	1. TIM2 vynulovat a tím dávkování opakovat,
	2. ignorovat aktuální koncentraci, t.j. do konce kroku nastavit konduktivita > CND.
39	je-li HLS použité vody: výpis: PLNÝ TANK RW
41	je-li HLS LOUH: výpis: PLNÝ TANK LOUH
42	je-li HLS KYSELINA: výpis: **PLNÝ TANK KYSELINA**
43	je-li LLS použitá voda: výpis: PRÁZDNÝ TANK RW
45	je-li LLS LOUH: PAUZA + výpis: PRÁZDNÝ TANK LOUH
	Start programu možný: není-li LLS LOUH
46	je-li LS KYSELINA: PAUZA + výpis: PRÁZDNÝ TANK KYSELINA
	Start programu možný: není-li LLS KYSELINA
48	dávkovat podle algoritmu:

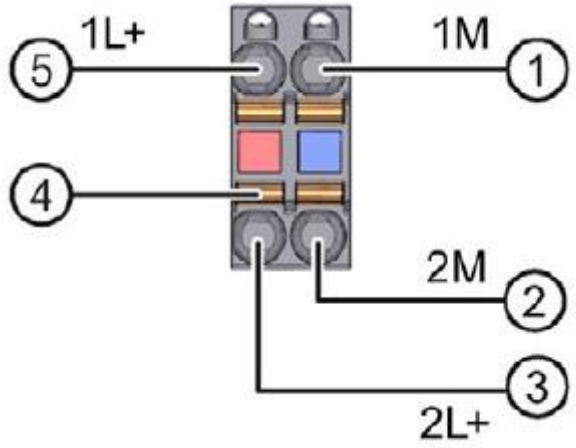


	je-li CND = 0, dávkuje se po celou dobu kroku
	je-li CND > 0, dávkuje se podle grafu je-li konduktivita < CND po dobu nepřetržitě delší než T4
	do časovače TIM2 se načítá součet časovačů TIM3
62	je-li VOL=0 a CND=0
65	je-li HLS KYSELINA
■	logická jednička

Příloha 7-3 Vysvětlující informace k maticovému zápisu FDS

Zapojení CPU

Obrázek níže- příloha 6-4 zobrazuje připojení zdroje stejnosměrného napětí 24 [V]



Příloha 7-4 [5]

Na pin 1 se připojuje uzemnění ke zdroji.

Pin 2- Zem- je vnitřně spojen s pinem 1, je využíván pro externí napájení maximálně 10 [A]

Pin 3- +24 [V]- je vnitřně spojen s pinem 5, je využíván s pinem 2 pro externí napájení maximálně 10 [A]

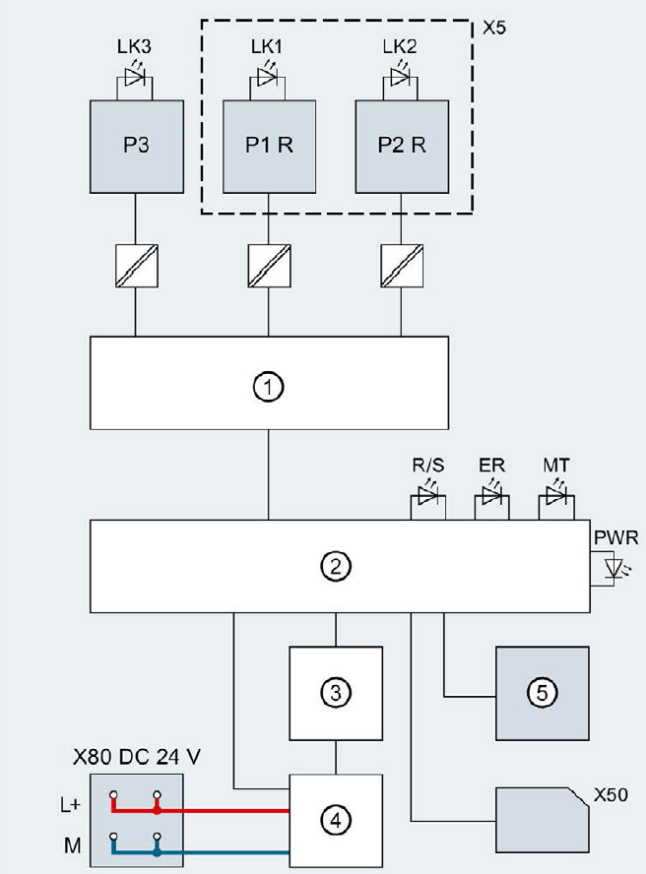
4- pružinová svorka

Pin 5 – připojení zdroje 24 [V] stejnosměrného napětí

Připojení k PROFINET je zajištěno konektorem RJ45. Porty pro připojení jsou dva a jsou vnitřně propojeny.

Každý port pro připojení k síti má vlastní MAC adresu.

Níže je blokové schéma jednotky CPU.



Příloha 7-5 [5]

1- Profinet switch
2- Elektronika
3- Propojení sběrnice komunikačního rozhraní
4- Interní zdroj napájení jednotky CPU
5- Přepínač stavu jednotky CPU RUN/STOP/MRES

X5- Adaptér sběrnice

X50- Memory karta typu SD SIEMENS

X80 24 V DC- vstup 24 [V] napájení

P1 R- PROFINET rozhraní X1 Port 1

P2 R- PROFINET rozhraní X1 Port 2

P3- PROFINET rozhraní X1 Port 3

L+ - Zdroj stejnosměrného napětí 24 [V]

M- uzemnění

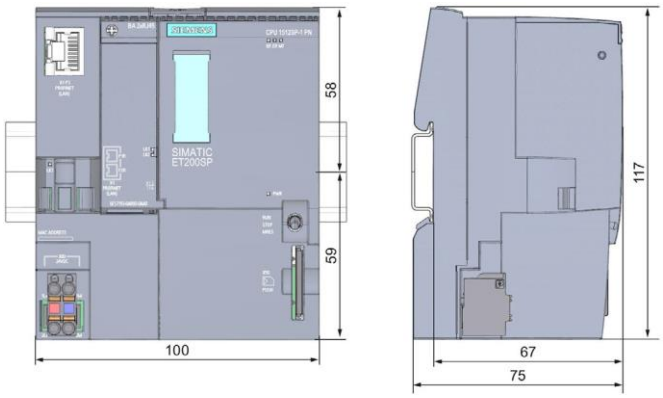
LK1, 2, 3- signalizace zobrazení průběhu komunikace TX/RX (LED diody)

R/S- signalizace stavu CPU RUN/STOP (LED diody žluté nebo zelené barvy)

ER- chybová signalizace (LED dioda červené barvy)

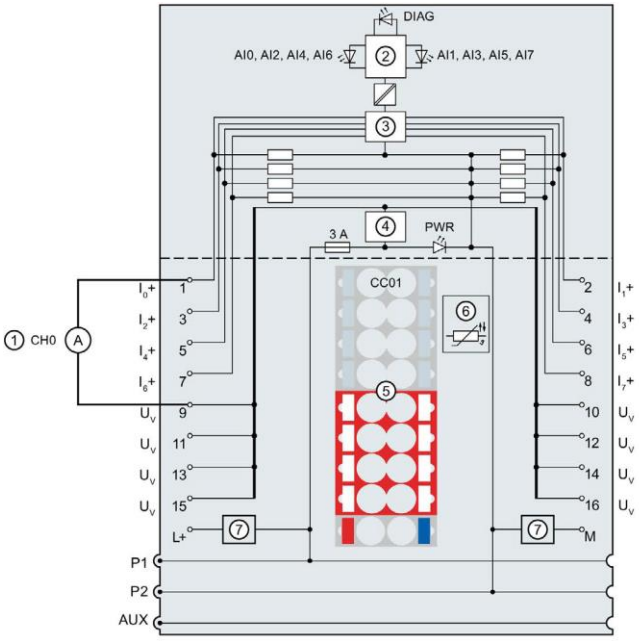
MT- signalizace údržby (LED dioda žluté barvy)

PWR- signalizace, že napájení jednotky je bez chyb (LED dioda zelené barvy)



Příloha 7-6 [5]

Zapojení modulu analogových vstupů AI 8xI 2-/4- wire BA



Příloha 7-7 [5]

1- Dvou drátové zapojení pro měření proudu
2- Propojení sběrnice komunikačního rozhraní
3- Šestnácti bitový ADC převodník
4- Limitace proudu
5- Identifikační barvený proužek
6- U tohoto modulu nevyužita
7- Připojení externího napětí

I_n+ - Vstupní signál kanálu n

U_V – Napájení měřící smyčky

L+ - Externí zdroj stejnosměrného napětí 24 [V]

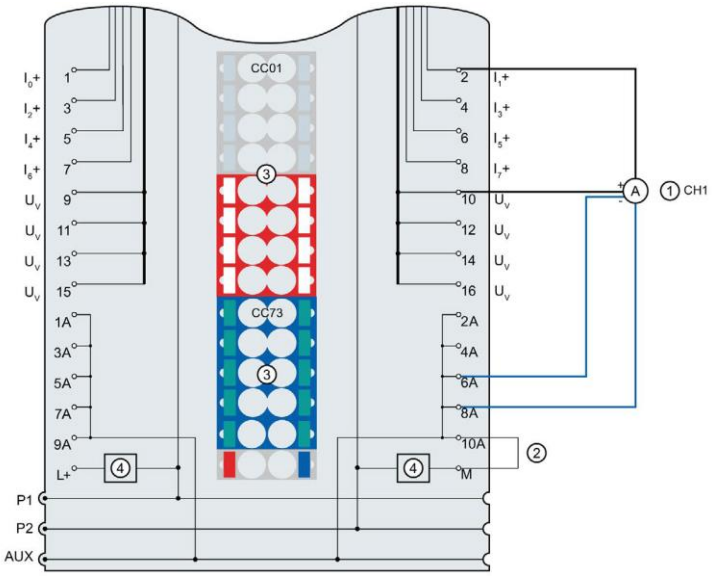
M- uzemnění

P1, P2, AUX- vnitřní samo získávací napěťové sběrnice

DIAG- diagnostické LED diody (zelené a červené barvy)

AI_n-status kanálu n

PWR- signalizace, že napájení jednotky je bez chyb (LED dioda zelené barvy)



Příloha 7-8 [5]

1- Čtyř drátové zapojení pro měření proudu
2- Uzemnění externích terminálů
3- Identifikační barvený proužek
4- Připojení externího napětí

I_n+ - Vstupní signál kanálu n

U_V – Napájení měřící smyčky

L+ - Externí zdroj stejnosměrného napětí 24 [V]

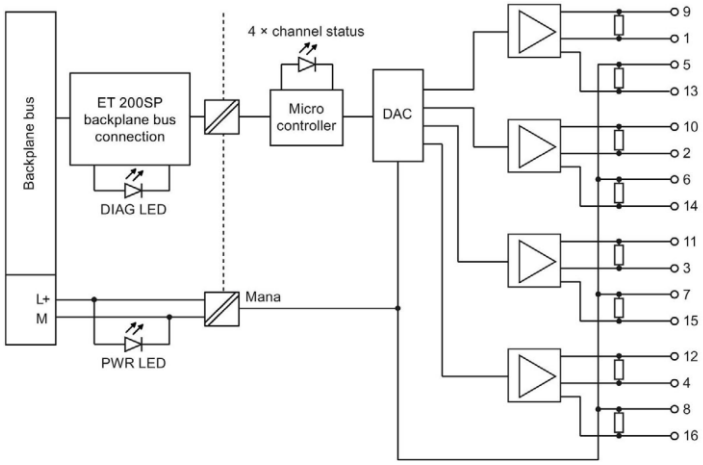
M- uzemnění

1 A až 10 A- externí terminály

P1, P2, AUX- vnitřní samo získávací napěťové sběrnice

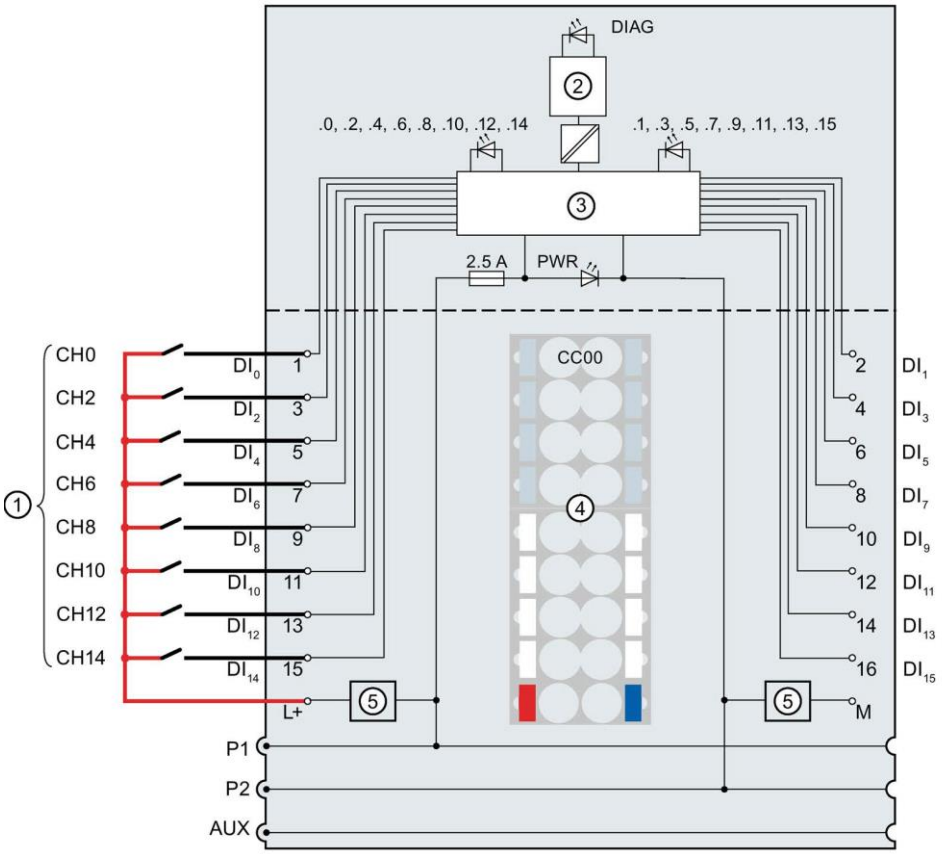
Zapojení modulu analogových výstupů AQ 4xU/I ST

Na Následujícím schéma se jednotlivé kanály zapojují mezi dvojice svorek 1, 5; 2, 6; 3, 7; nebo 4, 8. Stejné zapojení lze uplatnit jak pro proud, tak pro napětí. Pro napěťový výstup je možné zapojit i jeden nebo oba přídavné senzorové svorky a použít tak tří nebo čtyř drátové zapojení.



Příloha 7-9 [5]

Zapojení modulu digitálních vstupů DI 16x24VDC ST



Příloha 7-10 [5]

1- jedno drátové zapojení
2- Propojení sběrnice komunikačního rozhraní
3- vstupní elektronika
4- Identifikační barvený proužek
5- Připojení externího napětí

DI_n - Vstupní signál kanálu n

L+ - Externí zdroj stejnosměrného napětí 24 [V]

P1, P2, AUX- vnitřní samo získávací napěťové sběrnice

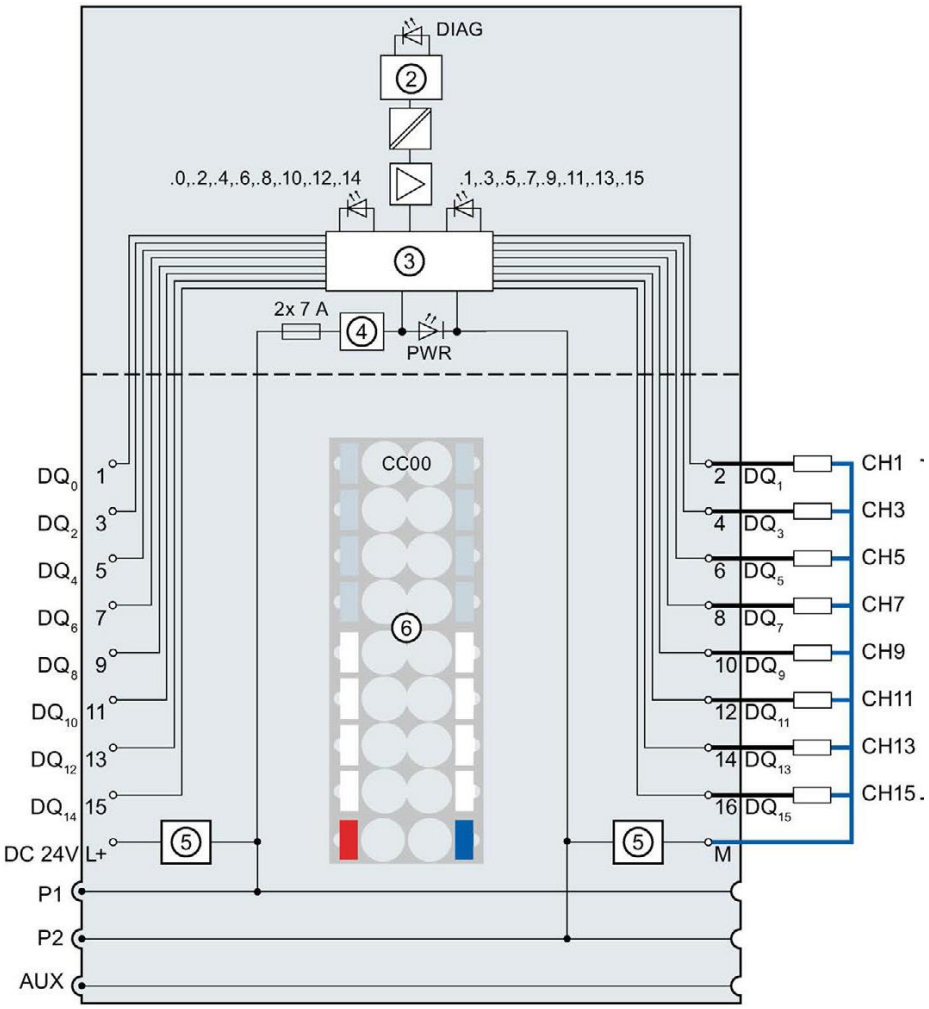
M- uzemnění

DIAG- diagnostické LED diody (zelené a červené barvy)

.0 až .15- status kanálu (LED dioda zelené barvy)

PWR- signalizace, že napájení jednotky je bez chyb (LED dioda zelené barvy)

Zapojení modulu digitálních výstupů DQ 16x24VDC/0.5A ST



Příloha 7-11 [5]

1- jedno drátové zapojení
2- Propojení sběrnice komunikačního rozhraní
3- výstupní elektronika
4- ochrana proti opačné polaritě napětí
5- Připojení externího napětí
6- Identifikační barvený proužek

DQ_n - Výstupní signál kanálu n

24 V DC - Externí zdroj stejnosměrného napětí 24 [V]

P1, P2, AUX- vnitřní samo získávací napěťové sběrnice

M- uzemnění

DIAG- diagnostické LED diody (zelené a červené barvy)

.0 až .15- status kanálu (LED dioda zelené barvy)

PWR- signalizace, že napájení jednotky je bez chyb (LED dioda zelené barvy)

8 SEZNAM POUŽITÉ LITERATURY

[1] KOSAŘ, Karel a Stanislav PROCHÁZKA. *Techhnologie výroby sladu a piva*. 3. Praha: Výzkumný ústav pivovarský a sladařský, 2012. ISBN 978-80-86576-52-7.

[2] BERGER, Hans. *Automating with STEP 7 in STL and SCL: programmable controllers SIMATIC S7-300/400*. 4th rev. ed. Erlangen: Publicis Corporate Pub., 2007. ISBN 978-3-89578-295-4.

[3] STEPHAN, Dominik, ed. Cleaning in Place: What is Cleaning in Place and How Does it Work. *Process Worldwide* [online]. Würzburg: Vogel Business Media, 2011, 06/22/2011 [cit. 2018-04-28]. Dostupné z: https://www.process-worldwide.com/what-is-cleaning-in-place-and-how-does-it-work-a-320588/

[4]SIMEONOVOVÁ, Ivana a Robert HOFMAN. Využití simulačního softwaru pro pokročilé plánování a rozvrhování. *Automa: časopis pro automatizační techniku*. 2013, **2013**(5), 12-13.

[5] SIEMENS. *SIMATIC ET 200SP: Manual Collection*[online]. NÜRNBERG, 2018[cit.2018-05-22]. Dostupné z: https://support.industry.siemens.com/cs/document/84133942/simatic-et-200sp-manual-collection?dti=0&lc=en-WW

[6] SANITAČNÍ STANICE. In: *Denwel* [online]. Havlíčkův Brod: Denwel, 2018 [cit. 2018-05-17]. Dostupné z: http://www.denwel.cz/procesni-jednotky

[7] Typical CIP cycle. In: *Ozonetech* [online]. Hägersten: Ozonetech, 2018 [cit. 2018-05-24]. Dostupné z: http://www.ozonetech.com/industries/breweries-ozone

[8] SIEMENS. *PROGRAMMING GUIDELINE FOR S7-1200/S7-1500*[online]. 2017[cit.2018-05-22]. Dostupné z: https://support.industry.siemens.com/cs/document/90885040/programming-guideline-for-s7-1200-s7-1500?dti=0&lc=en-US

[9] SIEMENS. *STATEMENT LIST (STL) FOR S7-300 AND S7-400 PROGRAMMING*[online]. NÜRNBERG, 2006[cit.2018-05-17]. Dostupné z: http://iat.fs.cvut.cz/109/files/S7/S7-SW2.pdf

[10] KOCOUREK, Vladimír. *ÚVOD DO POTRAVINÁŘSKÉ LEGISLATIVY* [online]. Praha: VŠCHT, 2014 [cit. 2018-05-15]. Dostupné z: https://web.vscht.cz/~kocourev/files/uvod-pl-skript.pdf