UNIVERZITA PARDUBICE
Fakulta elektrotechniky a informatiky

Laboratorní úloha "Řízení pomocí logického modulu LOGO!"

Dominik Papp

Bakalářská práce
2012

Univerzita Pardubice
Fakulta elektrotechniky a informatiky
Akademický rok: **2011/2012**

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

(PROJEKTU, UMĚLECKÉHO DÍLA, UMĚLECKÉHO VÝKONU)

Jméno a příjmení:	**Dominik Papp**
Osobní číslo:	**I09341**
Studijní program:	**B2646 Informační technologie**
Studijní obor:	**Řízení procesů**
Název tématu:	**Laboratorní úloha "Řízení pomocí logického modulu LOGO!"**
Zadávající katedra:	**Katedra řízení procesů**

Zásady pro vypracování:

Cíl: Cílem je vytvořit laboratorní úlohu "Řízení pomocí logického modulu LOGO!".
Obsah teoretické části: Teorie logického řízení. Programovatelné logické automaty a inteligentní relé - hardware a programování.
Obsah implementační části: Instalace softwaru pro programování logického modulu, simulaci a vizualizaci úlohy logického řízení. Vytvoření jednoduché logické úlohy - jak po hardwarové tak i softwarové stránce.

Rozsah grafických prací:

Rozsah pracovní zprávy:

Forma zpracování bakalářské práce: **tištěná/elektronická**

Seznam odborné literatury:

L. Šmejkal, M. Martinásková, PLC a automatizace 1, Základní pojmy, úvod do programování. BEN - technická literatura, Praha 1999
L. Šmejkal, PLC a automatizace 2, Sekvenční logické systémy a základy fuzzy logiky. BEN - technická literatura, Praha 2005
M. Martinásková, Programovací jazyky pro PLC. Automatizace, ročník 47, číslo 6, strana 380, 2004

Vedoucí bakalářské práce: **Ing. Daniel Honc, Ph.D.**
Katedra řízení procesů

Datum zadání bakalářské práce: **19. prosince 2011**
Termín odevzdání bakalářské práce: **11. května 2012**

Univerzita Pardubice
Česká republika

L.S.

prof. Ing. Simeon Karamazov, Dr.
děkan

doc. Ing. František Dušek, CSc.
vedoucí katedry

V Pardubicích dne 30. března 2012

Prohlášení autora

Prohlašuji, že jsem tuto práci vypracoval samostatně. Veškeré literární prameny a informace, které jsem v práci využil, jsou uvedeny v seznamu použité literatury.

Byl jsem seznámen s tím, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorský zákon, zejména se skutečností, že Univerzita Pardubice má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona, a s tím, že pokud dojde k užití této práce mnou nebo bude poskytnuta licence o užití jinému subjektu, je Univerzita Pardubice oprávněna ode mne požadovat přiměřený příspěvek na úhradu nákladů, které na vytvoření díla vynaložila, a to podle okolností až do jejich skutečné výše.

Souhlasím s prezenčním zpřístupněním své práce v Univerzitní knihovně.

V Pardubicích dne 11. 7. 2012 Dominik Papp

Poděkování
Tímto bych rád poděkoval vedoucímu bakalářské práce Ing. Danielovi Honcovi, Ph.D. za vedení práce, cenné rady a připomínky ke zpracování práce a pomoc s realizací hardwarové části modelu. V neposlední řadě bych chtěl také poděkovat mé rodině a přátelům za podporu během studia.

Anotace
V práci je popsána teorie logického řízení a programovatelné logické automaty po hardwarové stránce a jejich programování. Cílem práce je vytvořit laboratorní úlohu realizovanou logickým modulem LOGO!. K programování je použito vývojové prostředí LOGO!Soft Comfort. Je navržena a vytvořena úloha s modelem hydraulické posuvné jednotky

Klíčová slova
logické řízení, PLC, programování PLC, LOGO!

Title
Laboratory task „Control using logic module LOGO!“

Annotation
The work describes the theory of logic control and programmable logic controllers the hardware and programming. The aim is to create a laboratory task realized by logic module LOGO!. The LOGO! Soft Comfort is use for programming. Laboratory task with model of hydraulic shift unit is designed and realized.

Keywords
logic control, PLC, programming PLC, LOGO!

Obsah

Seznam zkratek

DNF	Disjunktivní normální forma
KNF	Konjunktivní normální forma
PLC	Programovatelný logický automat
RAM	Random access memory
EPROM	Erasable programmable read-only memory
EEPROM	Electrically erasable programmable read-only memory
CPU	Central processing unit
PTM	Pravé tlačítko myši
USB	Universal serial bus
PC	Personal computer
LD	Ladder Diagram
FBD	Function Blok Diagram
IL	Instructions List
ST	Structured Text
R	Reset
S	Set
PAR	Parameter
CNT	Count
DIR	Direction
Q	Output Q
TRG	Trigger
T	Parameter T
LED	Light emitting diode
DI	Digital input
DO	Digital output

Seznam obrázků

Seznam tabulek

1 Úvod

Cílem této bakalářské práce je vytvoření laboratorní úlohy po hardwarové a softwarové stránce za použití logického modulu LOGO!.
V teoretické části bakalářské práce je uvedena teorie logického řízení a programovatelné logické automaty, jejich hardware a programování.
V praktické části bakalářská práce je popsán logický modul LOGO!, vývojové prostředí LOGO!Soft Comfort pro návrh programů, model posuvné jednotky a je vytvořena sada úloh pro tento model.

Řízení je vzájemné působení dvou objektů (řídicí a řízený), jeden řídí druhý. Rozeznáváme dva druhy řízeni:

- ovládáni, řídicí člen pouze ovládá řízený objekt. Nemá informaci o skutečném stavu řízeného objektu.
- regulace, mezi řídicím a řízeným členem je zpětná vazba, která poskytuje řídicímu členu informaci o skutečném stavu řízeného objektu.

Logické řízení je cílená činnost, při níž se logickým obvodem zpracovávají informace o řízeném procesu a podle nich ovládají příslušná zařízení tak, aby se dosáhlo předepsaného cíle. [3]
U tohoto způsob řízení se používají pouze dva stavy, mezi nimiž neexistuje žádný mezistav. Označují se pomocí číslic dvojkové soustavy nejčastěji jako:

- logická nula - 0,
- logická jedna - 1,

nebo také L/H, F/T, VYPNUTO/ZAPNUTO.

Logický obvod je obvod, jehož veličina může nabývat pouze dvou hodnot. Logické obvody jsou tvořeny logickými členy, které realizují základní logické operace .

Programovatelné logické automaty(PLC) jsou logické automaty schopné vykonávat uživatelský program, který by jinak musel být realizován pomocí reléové, nebo číslicové logiky. PLC zjednodušuje a mnohdy i zlevňuje technické provedení. Ulehčuje také hledání a odstraňování chyb, neboť se nemusí hledat ve „spleti drátů“ ten špatně připojený. Ve vývojovém prostředí PLC lze program simulovat a odstranit chyby ještě před uvedením do provozu.

2 Teoretická část

2.1 Teorie logického řízení

2.1.1 Základní logické funkce

Logická funkce určuje kombinacím vstupních proměnných logickou hodnotu nula, nebo jedna výstupní proměnné.

Logickou funkci můžeme zapsat

- logickou rovnicí,
- pravdivostní tabulkou, nebo
- Karnaughovou mapou.

Logické funkce jedné proměnné

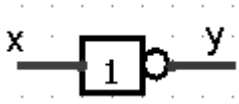
Nejjednodušší případ logické funkce je funkce jedné proměnné, kterou lze vyjádřit jedním ze čtyř způsobů:

- **falsum** (lež), kde pro libovolné x je y rovno 0,
- **verum** (pravda), kde pro libovolné x je y rovno 1,
- **aserce** (opakování), hodnota y má vždy stejnou hodnotu jako x,
- **negace** (opak), hodnota y má vždy opačnou hodnotu než x.

Tabulka 1: Logické funkce jedné proměnné

falsum		verum		aserce		negace	
x	y	x	y	x	y	x	y
0	0	0	1	0	0	0	1
1	0	1	1	1	1	1	0
y = 0		y = 1		y = x		y = $\overline{x}$	

Z těchto logických funkcí má praktický význam pouze **negace**, která patří k nejdůležitějším logickým funkcím.

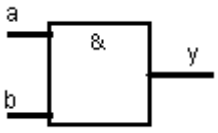


Obrázek 1: NOT

Logické funkce dvou proměnných

Pro logické funkce dvou proměnných mohou nastat čtyři různé kombinace vstupů, celkem existuje tedy 16 funkcí dvou proměnných, v praxi má však největší význam šest následujících funkcí:

- **Disjunkce** (logický součet, **OR**)
  Logický součet je logická funkce dvou, nebo více proměnných a má hodnotu logická jedna tehdy, má-li alespoň jedena ze vstupních nezávisle proměnných hodnotu logická jedna.
  Operátor této funkce je +, používá se též znak ∨.

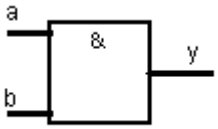


Obrázek 2: OR

Tabulka 2: OR

a	b	y	
0	0	0	y = a + b
0	1	1	y = a ∨ b
1	0	1	
1	1	1	

- **Konjunkce** (logický součin, **AND**)
  Logický součin je logická funkce dvou, nebo více proměnných a má hodnotu logická jedna jen tehdy, když všechny vstupní nezávisle proměnné mají hodnotu logická jedna.
  Operátor této funkce je ●, používá se též znak ∧.

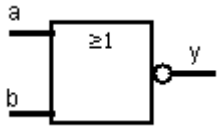


Obrázek 3: AND

Tabulka 3: AND

a	b	y	
0	0	0	y = a . b
0	1	0	y = a ∧ b
1	0	0	
1	1	1	

- **Negace disjunkce** (negovaný logický součet - Piercova funkce, **NOR**)
  Negovaný logický součet je logická funkce dvou, nebo více proměnných a má hodnotu logická jedna jen tehdy, jsou-li všechny ze vstupních nezávislých proměnných v hodnotě logická nula.

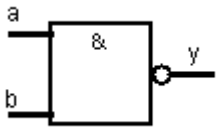


Obrázek 4: NOR

Tabulka 4: NOR

a	b	y	
0	0	1	$y = \overline{a + b}$
0	1	0	$y = \overline{a \vee b}$
1	0	0	
1	1	0	

- **Negace konjunkce** (negace logického součinu - Shefferova funkce, **NAND**)
  Negovaný logický součin je logická funkce dvou, nebo více proměnných a má hodnotu logická nula jen tehdy, jsou-li všechny vstupní nezávisle proměnné ve stavu logická jedna.

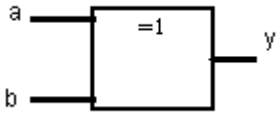


Obrázek 5: NAND

Tabulka 5: NAND

a	b	y	
0	0	1	$y = \overline{a \cdot b}$
0	1	1	$y = \overline{a \wedge b}$
1	0	1	
1	1	0	

- **Nonekvivalence** (exkluzivní součet, **XOR**)
  Exkluzivní součet je logická funkce dvou, nebo více proměnných a má hodnotu logická nula jen tehdy, jsou-li všechny vstupní nezávisle proměnné ve stavu logická jedna, nebo ve stavu logická nula.

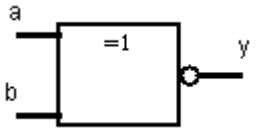


Obrázek 6: XOR

Tabulka 6: XOR

a	b	y	
0	0	0	$y = \overline{a} \wedge b \vee a \wedge \overline{b}$
0	1	1	
1	0	1	$y = a \oplus b$
1	1	0	

- **Ekvivalence** (inkluzivní součet, **XNOR**)
  Funkce ekvivalence vznikne negací funkce nonekvivalence a má hodnotu logická jedna jen tehdy, jsou-li všechny vstupní nezávisle proměnné ve stavu logická jedna, nebo ve stavu logická nula.



Obrázek 7: XNOR

Tabulka 7: XNOR

a	b	y	
0	0	0	$y = a \wedge b \vee \overline{a} \wedge \overline{b}$
0	1	1	
1	0	1	$y = \overline{a \oplus b}$
1	1	0	

2.1.2 Pravdivostní tabulka

Pravdivostní tabulka vyjadřuje závislost výstupních (závislých) proměnných na vstupních (nezávislých) proměnných. V pravdivostní tabulce se nacházejí veškeré možné kombinace vstupních proměnných, kterým je přiřazena hodnota logické funkce – 0 nebo 1. Jedna kombinace představuje jeden řádek pravdivostní tabulky, počet sloupců je dán počtem všech proměnných.

Počet kombinací (řádků) pravdivostní tabulky je dán vztahem

$$N = 2^n$$

kde,

N - je výsledný počet kombinací
n - je počet vstupních proměnných

Při větším počtu vstupních proměnných tabulka obsahuje značný počet řádků, čímž se stává velmi obsáhlá – pro 8 vstupních proměnných má tabulka $2^8 = 256$ řádků.

Příklad tabulky o třech vstupních proměnných (a, b, c), tedy o

$N = 2^3 = 8$ kombinacích (řádků).

Tabulka 8: Pravdivostní tabulka

stav	a	b	c	y
1	0	0	0	0
2	0	0	1	0
3	0	1	0	1
4	0	1	1	1
5	1	0	0	0
6	1	0	1	x
7	1	1	0	0
8	1	1	1	1

První sloupec tabulky (stav) udává pořadí kombinace. Sloupce a, b, c jsou vstupní proměnné a vyjadřují všechny možné kombinace, které mohou pro tyto proměnné nastat. Poslední sloupec y vyjadřuje hodnoty logické funkce pro jednotlivé stavy

- 1 pro kombinace, které jsou pravdivé
- 0 pro kombinace, které nejsou pravdivé
- X je neurčitý stav, stav u kterého nezáleží na hodnotě výstupní proměnné

Z tabulky mohou být vyjmuty stavy, které nemohou nastat (motor se nemůže točit doleva a zároveň doprava).

2.1.3 Booleova algebra

Logická algebra pro práci s logickými funkcemi, je pojmenována podle irského matematika G. Boolea (1515-1864). Používá tři základní logické funkce – negaci, disjunkci a konjunkci. Pomocí Booleovy algebry lze výraz minimalizovat, to znamená upravit jej na co nejmenší počet základních logických funkcí, díky čemuž se pro realizaci použije méně logických prvků. Výsledný logický obvod tak bude ekonomičtější, jednodušší i spolehlivější. Pro minimalizaci se za pomoci zákonů Booleovy algebry upravují výrazy tak dlouho, dokud nenalezneme nejjednodušší tvar.

Základní zákony Booleovy algebry

- Komutativní zákon:

$$a \vee b = b \vee a$$
$$a \wedge b = b \wedge a$$

- Distributivní zákon:

$$(a \vee b) \wedge c = (a \wedge c) \vee (b \wedge c)$$
$$(a \wedge b) \vee c = (a \vee c) \wedge (b \vee c)$$

- Asociativní zákon:

$$a \vee (b \vee c) = (a \vee b) \vee c$$
$$a \wedge (b \wedge c) = (a \wedge b) \wedge c$$

- Zákon vyloučení třetího:

$$a \vee \bar{a} = 1$$
$$a \wedge \bar{a} = 0$$

- Zákon neutrálnosti logické nuly a logické jedničky:

$$a \vee 0 = a$$
$$a \wedge 1 = a$$

- Zákon agresivity logické nuly a logické jedničky:

$$a \wedge 0 = 0$$
$$a \vee 1 = 1$$

- Zákon idempotence:

$$a \vee a = a$$
$$a \wedge a = a$$

- Zákon dvojí negace:

$$\bar{\bar{a}} = a$$

- Zákon absorpce:

$$a \vee a \wedge b = a$$
$$a \wedge (a \vee b) = a$$

- Zákon absorpce negace:

$$a \vee (\bar{a} \wedge b) = a \vee b$$
$$a \wedge (\bar{a} \vee b) = a \wedge b$$

- De-Morganovy zákony:

$$\overline{a \vee b \vee c} = \bar{a} \wedge \bar{b} \wedge \bar{c}$$
$$\overline{a \wedge b \wedge c} = \bar{a} \vee \bar{b} \vee \bar{c}$$

2.1.4 Karnaughova mapa

Mapa je čtverec, nebo obdélník, který má n vstupních proměnných rozdělených do 2^n čtverečků. Mapa má tolik čtverečků, kolik má pravdivostní tabulka řádků. Každý čtvereček v mapě odpovídá jednomu řádku z pravdivostní tabulky. Rozeznáváme různé druhy map, z nichž nejznámější je Karnaughova mapa.

Pomocí Karnaughovy mapy se vyjadřují Booleovské funkce, které je pak jednodušší minimalizovat. V Karnaughově mapě jsou k jednotlivým políčkům přiřazeny stavové indexy a jsou v nich zapsány hodnoty výstupních proměnné. Indexy sousedních políček se od sebe liší hodnotou jedné proměnné. Pro každou výstupní proměnnou se tvoří samostatná mapa.

Vyjádření logické funkce z Karnaughovy mapy může mít dva tvary:

- **Součtový tvar** – disjunktivní normální forma (DNF)
  Jedná se o součet součinů. Získáme jej tak, že zahrneme všechny stavy, které nabývají hodnoty logická jedna, vstupní proměnné mezi sebou násobíme a jednotlivé stavy mezi sebou sčítáme.

  Je-li vstupní proměnná pokryta pruhem, znamená hodnotu logická jedna a označuje se symbolem vstupní proměnné – např. a. Není-li pokryta pruhem, znamená hodnotu logická nula a označuje se negovaným symbolem – např. $\bar{a}$.

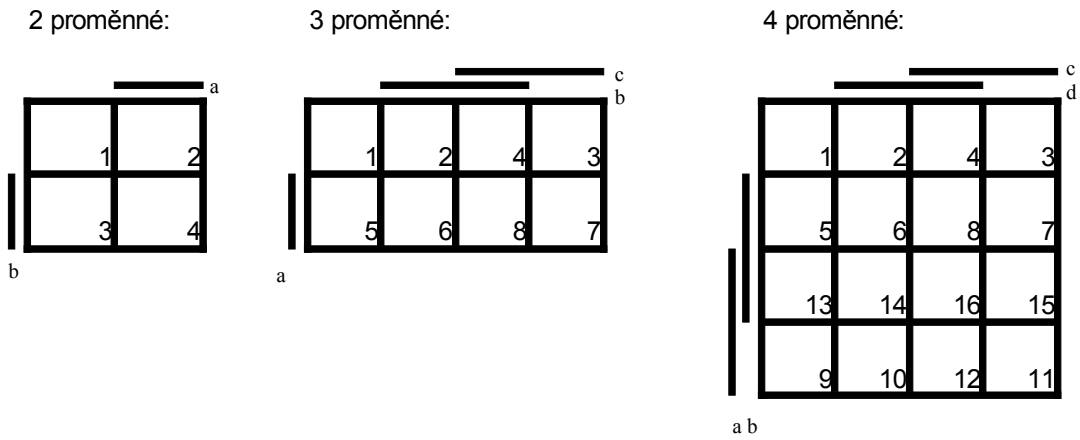
Příklad $y = \bar{a} \wedge b \wedge c \vee \bar{a} \wedge \bar{b} \wedge c \vee a \wedge b \wedge \bar{c} \quad \equiv \quad y = \bar{a} \cdot b \cdot c + \bar{a} \cdot \bar{b} \cdot c + a \cdot b \cdot \bar{c}$

- **Součinový tvar** – konjunktivní normální forma (KNF)
  Jedná se o součin součtů. Získáme jej tak, že zahrneme všechny stavy, které nabývají hodnoty logická nula, vstupní proměnné mezi sebou sčítáme a jednotlivé stavy mezi sebou násobíme.

  Je-li vstupní proměnná pokryta pruhem, znamená hodnotu logická nula a označuje se negovaným symbolem vstupní proměnné – např. $\bar{a}$. Není-li pokryta pruhem, znamená hodnotu logická jedna a označuje se symbolem – např. a.

Příklad

$$y = (\bar{a} \vee b \vee c) \wedge (\bar{a} \vee \bar{b} \vee c) \wedge (a \vee b \vee \bar{c}) \quad \equiv \quad y = (\bar{a} + b + c) \cdot (\bar{a} + \bar{b} + c) \cdot (a + b + \bar{c})$$



Obrázek 8: Karnaughovy mapy pro 2, 3 a 4 proměnné

2.1.5 Zjednodušování logických funkcí

Logická funkce může být zapsána několika různými tvary, které jsou z matematického hlediska rovnocenné. Pro technickou realizaci je ale nezbytné upravit takovou funkci na nejjednodušší tvar – minimalizovat. Zjednoduší se tím její realizace, protože bude zapotřebí méně logických členů. Obvod bude jak ekonomičtější tak i spolehlivější.

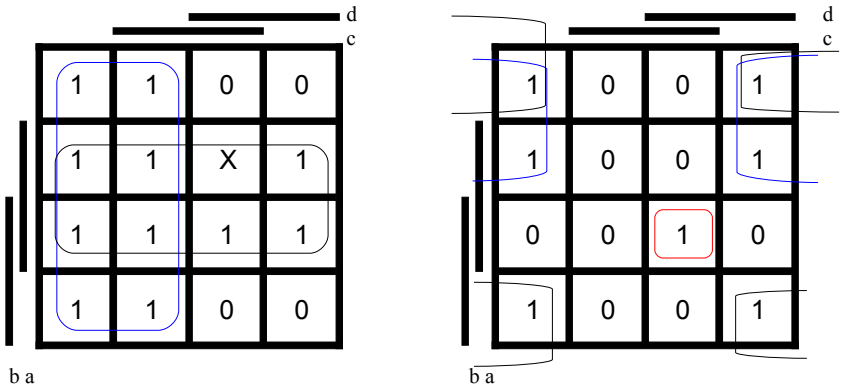
Algebraická minimalizace

Za pomoci Booleovy algebry postupně zjednodušujeme výraz do té doby, než nenalezneme minimální výraz. Tato metoda je však velmi pracná a zdlouhavá, zejména pro složitější výrazy. U této metody nemáme stoprocentní jistotu, že jsme nalezli minimální tvar, také se snadno můžeme dopustit chyby, která se obtížně hledá.

Minimalizace pomocí Karnaughovy mapy

V Karnaughově mapě ohraničujeme sousední políčka tak, že vzniknou čtverce, nebo obdélníky = smyčky. Smyčky mají vždy co největší velikost, mohou též z části překrývat jedna druhou. Počet jedniček, nebo nul ve smyčce musí být mocninou čísla dvě (2^0, 2^1, 2^2, …), přičemž do smyčky lze zahrnout i políčko označující neurčitý stav X (ovšem je-li to výhodné).

Sousedními políčky jsou taková políčka, která mají společnou alespoň jednu hranu. Karnaughovu mapu lze přehnout, tak že pravý okraj mapy přiložíme k levému (levý okraj bude sousedit s pravým), totéž můžeme provést se spodním a vrchním okrajem (spodní okraj bude sousedit s vrchním), takže jedničky na okrajích mapy jsou taktéž sousedními. Mapu lze přehnout také oběma způsoby současně (levý okraj bude sousedit s pravým a zároveň spodní okraj bude sousedit s vrchním), tedy jedničky v rozích se stávají taktéž sousedními.



Obrázek 9: Karnaughova mapa - minimalizace

Základní pravidla pro minimalizaci v Karnaughově mapě

- Všechny jedničky (nuly) v mapě musí být pokryty smyčkou
- Každá jednička (nula) může být součástí několika smyček
- Smyčky by měli pokrýt co nejvíc jedniček (nul) naráz,
- Snažíme se vytvářet co nejméně smyček

Příklad minimalizace

Máme minimalizovat následující logickou funkci do DNF tvaru

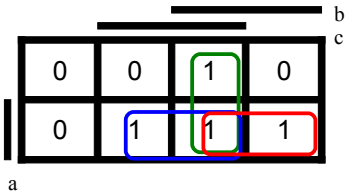
$$y = a \cdot \bar{b} \cdot c + a \cdot b \cdot c + a \cdot b \cdot \bar{c} + \bar{a} \cdot b \cdot c$$

Vytvoříme odpovídající pravdivostní tabulku:

Tabulka 9: Příklad minimalizace - tabulka

stav	a	b	c	y
1	0	0	0	0
2	0	0	1	0
3	0	1	0	0
4	0	1	1	1
5	1	0	0	0
6	1	0	1	1
7	1	1	0	1
8	1	1	1	1

Sestavíme Karnaughovu mapu:



Obrázek 10: Karnaughova mapa - příklad

V mapě zakreslíme smyčky, ze kterých určíme členy logické funkce.

$$a\cdot\bar{b}\cdot c+a\cdot b\cdot c=a\cdot c\cdot(\bar{b}+b)=\underline{a\cdot c}$$
$$a\cdot b\cdot c+a\cdot b\cdot\bar{c}=a\cdot b\cdot(\bar{c}+c)=\underline{a\cdot b}$$
$$a\cdot b\cdot c+\bar{a}\cdot b\cdot c=b\cdot c\cdot(\bar{a}+a)=\underline{b\cdot c}$$

Při minimalizaci ty nezávisle proměnné, které pokrývají smyčku jen z poloviny nepíšeme. Pro první smyčku tedy dostáváme výraz: $\underline{a\cdot c}$, neboť pruh platný pro proměnnou **b** pokrývá smyčku jen z poloviny.

Výsledná minimalizovaná logická funkce je $y=a\cdot c+a\cdot b+b\cdot c$

2.1.6 Realizace kombinačních logických obvodů

Kombinační logické funkce lze realizovat technickými prostředky, které jsou založeny na následujících technických prvcích

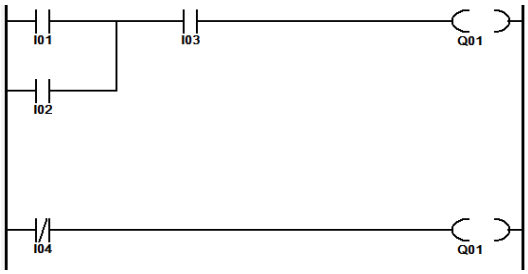
- mechanická relé,
- integrované obvody, nebo
- programovatelné logické automaty.

Každému z těchto prvků odpovídá specifický způsob znázorňování schématu zapojení těchto prvků.
Reléové obvody se znázorňují v liniových kontaktních schématech.
Elektronické logické prvky využívají blokových schémat a funkce programových logických automatů je popsána některou formou dokumentace software. [2]

Liniové kontaktní schéma

Schéma zapojení je tvořeno pomocí elektrických kontaktů, které umožní průchod proudu mezi póly zdroje. Tyto póly se kreslí jako dvě svislé čáry, fáze (L) a zem (N). Proud prochází přes jednotlivé spínače do zátěže, což může být libovolný spotřebič jako žárovka, cívka elektromagnetického relé, atd.

- Logický součin je tvořen dvěma a více kontakty zapojených za sebou – sériově,
- Logický součet je tvořen dvěma a více kontakty zapojených vedle sebe – paralelně,
- Logická negace je tvořena rozpínacím kontaktem.



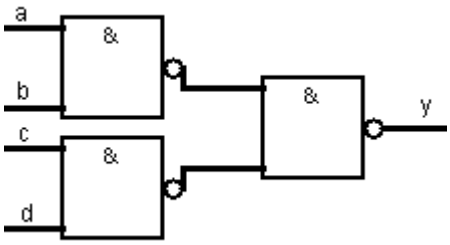
Obrázek 11: Kontaktní schéma

První funkce realizuje y = (I01 + I02) . I03
Druhá funkce realizuje y = $\overline{I04}$

Blokové schéma

Schéma zapojení je tvořeno pomocí elektronických součástek, které plní logické funkce - hradla. Používají se základní logické funkce NOT, AND, OR, XOR a jejich negace. Tyto funkce lze s výhodou vytvořit pouze z hradel NAND, které jsou jednoduché na výrobu. Výhodou je pak to, že je výsledný logický obvod složen jen z jednoho druhu hradel.
Úprava pro logické členy NAND se provádí pomocí De-Morganova zákona dvojí negace na výrazy v DNF formě.

$$y = a.b + c.d = \overline{\overline{a.b + c.d}} = \overline{\overline{(a.b)}.\overline{(c.d)}}$$



Obrázek 12: Logická funkce z hradel NAND

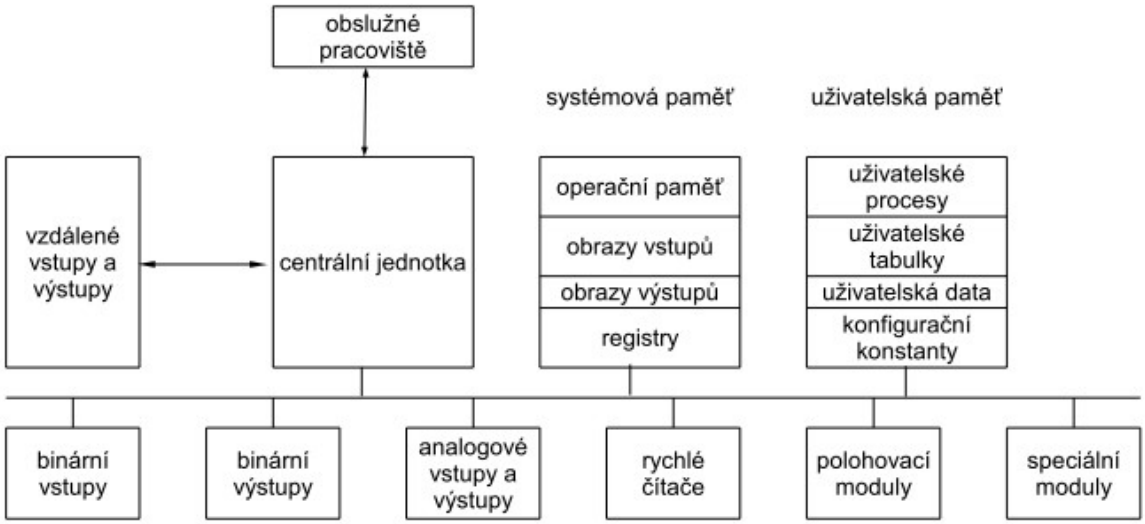
2.2 Programovatelné logické automaty

2.2.1 Co je to programovatelný logický automat

Programovatelný logický automat, označovaný zkratkou PLC (Programmable Logic Controller), je řídicí systém určený pro řízení technologického procesu podle uživatelského programu (např. řízení výrobní linky).

PLC nahrazují počítače pro průmyslové prostředí, protože byly navrženy tak, aby nebyly tolik náchylné k pracovním podmínkám. Mezi hlavní výhody PLC patří vysoká odolnost vůči vibracím, prašnému a vlhkému prostředí, vyšší odolnost proti elektromagnetickým polím a výkyvům napájecího napětí. Vyznačuj se také vysokou spolehlivostí a obsahují vnitřní diagnostické funkce, které kontrolují běh systému a v případě závady ji identifikují a ohlásí.

Vnitřní struktura PLC je založena na sběrnicové architektuře s mikroprocesorem, kde všechny části PLC jsou připojeny na společnou sběrnici.



Obrázek 13: Blokové schéma vnitřní struktury programovatelného automatu

Centrální jednotka
Obsahuje mikroprocesor, který řídí veškerou činnost PLC, vykonává program. Většinou obsahuje i programovací rozhraní pro přenos programu z počítače do PLC a standardizované rozhraní pro sériovou komunikaci. Obvykle je typu EPROM / EEPROM.

Paměť
Je rozdělena na dvě části: uživatelská a systémová. Uživatelská paměť slouží k uchování programu. Systémová paměť uchovává obrazy vstupů a výstupů a slouží jako operační paměť programu. Obvykle je typu RAM.

Vstupy a výstupy
Slouží ke komunikaci PLC s okolím, dnešní PLC mají možnost binárních i analogových vstupů a výstupů.

Binární vstupy slouží pro přivedení dvouhodnotových signálů,jako jsou tlačítka, spínače.

Binární výstupy umožňují dvouhodnotové řízení, jako spínání žárovky, spínání cívek relé. Výstupy mohou být tranzistorové pro spínání menších proudů (řádově desetiny A), nebo reléové pro spínání větších proudů (jednotky A).

Analogové vstupy slouží k přivedení analogové hodnoty do PLC, lze na ně připojit různé snímače s analogovým výstupem, jako např. odporový termočlánek pro měření teploty.

Analogové výstupy umožňují ovládat frekvenční měniče či servopohony.

Speciální moduly
Rozšiřují možnosti a oblast působnosti PLC. Mohou to být moduly realizující PID regulátor, moduly pro řízení hydraulických ventilů, diagnostické moduly, či komunikační moduly umožňující vzájemnou komunikaci mezi jednotlivými PLC.

2.2.2 Typy PLC

Typ PLC volíme podle rozsahu řešené úlohy. Některé moduly nemusí být zahrnuty vůbec, jiné naopak vícekrát. PLC můžeme nakonfigurovat jako čistě výstupní systém (ovladač elektrických spotřebičů), nebo jen jako binární systém obsahující pouze binární vstupy a výstupy (řízení pásového dopravníku).

Mikro PLC
Jedná se o nejmenší a nejlevnější PLC systém. Nabízí pevnou sestavu vstupů a výstupů, obvykle jen v binárním provedení. Uživatel se rozhodne pro jednu sestavu (4/4, 6/6, 8/6 …) vstupů a výstupů, kterou posléze nelze rozšiřovat. Mikro PLC se obvykle používají pro realizaci jednoduché ovládací logiky.

Kompaktní PLC
Nabízí taktéž pevnou sestavu vstupů a výstupů, ale k základnímu PLC je možné připojit rozšiřující moduly – moduly rychlých čítačů, analogové moduly, ovšem tato rozšiřitelnost je omezená.

Modulární PLC
Tento druh PLC se skládá z jednotlivých částí – modulů, které se zasouvají do rámu. PLC systém se pak skládá z napájecího modulu, modulu CPU, vstupních a výstupních modulů, případně speciálních modulů. Moduly zdroje, CPU, přídavné paměti a komunikační moduly mají v rámu pevné umístění, ostatní moduly lze umístit libovolně. Tyto rámy se vyrábějí v různých délkách a lze jich pro jeden systém požít více. Pro tento typ PLC systému existuje nejvíce rozšiřovacích modulů:

- napájecí moduly,
- CPU moduly,
- moduly analogových vstupů a výstupů,
- moduly binárních vstupů a výstupů,
- čítací, polohovací, komunikační moduly,
- operátorské panely,
- a další.

Moduly se mohou umisťovat na větší vzdálenosti (i stovky metrů) a lze tak vytvářet distribuované systémy.

Soft PLC
Je nová kategorie PLC. PLC systém je tvořen centrálním počítačem a sadou vstupních modulů, které jsou spojeny průmyslovou sběrnicí. Centrální počítač provádí veškeré řídicí funkce PLC a navíc umožňuje přímo zadávat, překládat i simulovat program. K napsání programu má programátor vedle tradičních programovacích jazyků PLC navíc služby operačního systému a jeho programové vybavení – databázové operace, programování v jazyce C, simulační a výpočetní programy.
Výhoda tohoto řešení je kompaktní celek zahrnující v sobě funkce PLC, jeho vývojové a vizualizační prostředí a operátorské rozhraní.
Nevýhoda je centralizace funkcí do jednoho počítače, díky čemuž se stává systém zranitelnější a vysoká cena.

2.2.3 Vykonávání programu

Program je soubor instrukcí vykonávající se postupně jedna po druhé. Na rozdíl od jiných systémů se programátor nemusí starat o to, aby program běžel ve smyčce (je to zajištěno automaticky). Ochranu před „zacyklením programu“ zajišťuje tzv. Watchdog – při překročení maximálního povoleného času v jedné programové smyčce je zahlášeno jako „překročení doby cyklu“ a program je zastaven.

Program PLC nepracuje s vlastními vstupy a výstupy, ale s jejich obrazy uloženými v systémové pamětí v oblasti obrazů vstupů a výstupů (registry X a Y). Na začátku každého cyklu se sejme obraz vstupů a dále v programu se pracuje s tímto obrazem, změní-li se hodnota na vstupu v době, kdy ještě nebyl dokončen cyklus, tak se tato změna do chodu programu nepromítne. Při zpracovávání programu se výstupní hodnoty zapisují do oblasti obrazů výstupů, toto je výhodné zejména proto, že dojde-li během jednoho cyklu programu k několika změnám jednoho výstupu, tak se toto „kmitání“ neprojeví na výstupech PLC.

Po zpracování poslední instrukce vlastního programu se na výstupy PLC naráz zapíše obraz výstupu, proběhne režie PLC a pokračuje se další smyčkou cyklu. Režie zajišťuje to, že program běží v nekonečné smyčce, nuluje watchdog, komunikuje po sériové lince.

Obrázek 14: Běh programu v PLC

2.2.4 Programovací jazyky PLC

K napsání programu pro PLC se používají softwarové nástroje – vývojová prostředí, každý výrobce dodává své prostředí. Napsaný program není přenositelný mezi různými výrobci PLC, nemusí být přenositelný ani v rámci jednoho výrobce. Pro napsání program jsou k dispozici čtyři typy programovacích jazyků:

- **grafické programovací jazyky**
  - **LD** – jazyk kontaktních schémat (Ladder Diagram)
  - **FBD** – jazyk funkčních bloků (Function Blok Diagram)
- **textové programovací jazyky**
  - **IL** – jazyk mnemokódů (Instructions List)
  - **ST** – strukturovaný text (Structured Text)

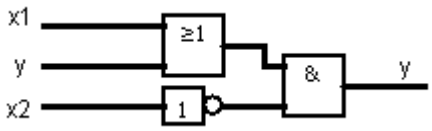
Jazyk kontaktních schémat

Program se kreslí za pomoci spínacích a rozpínacích reléových kontaktů zobrazující se jako dvě závorky (v případě rozpínacího kontaktu přeškrtnuté), reléové cívky symbolizující výstup, funkční bloky (čítače, časovače) jsou kresleny jako obdélníkové značky. Tento jazyk je vhodný pro programování jednodušších úloh. Ocení jej lidé, kteří neovládají programování, tvorba programu je názorná a hledání chyb bývá velmi rychlé.

Obrázek 15: Jazyk LD

Jazyk funkčních bloků

Tento programovací jazyk je podobný kreslení logických schémat. Základní logické operace, aritmetické instrukce i funkční bloky (čítače, časovače) jsou interpretovány jako obdélníkové značky.



Obrázek 16: Jazyk FBD

Jazyk mnemokódů

Jedná se o obdobu Assembleru, tedy každé instrukci PLC odpovídá stejně pojmenovaný příkaz. U tohoto jazyka lze použít veškeré vlastnosti Assembleru, jako pojmenované vnitřní proměnné, konstanty, návěstí používané pro skoky v programu. Program je vykonáván tak, jak jdou instrukce za sebou – výjimku tvoří skoky v programu a volání podprogramu.

```
P 0
ld      x0.0 ;tlačítko start
or      y0
anc     x0.1 ;tlačítko stop
wr      y0
E 0
```

Obrázek 17: Jazyk IL

Strukturovaný text

Je obdoba vyššího programovacího jazyka, např. Jazyk C. Zápis algoritmů je názorný a program lze členit do funkcí.

```
vystup := (start OR vystup) NOT stop;
```

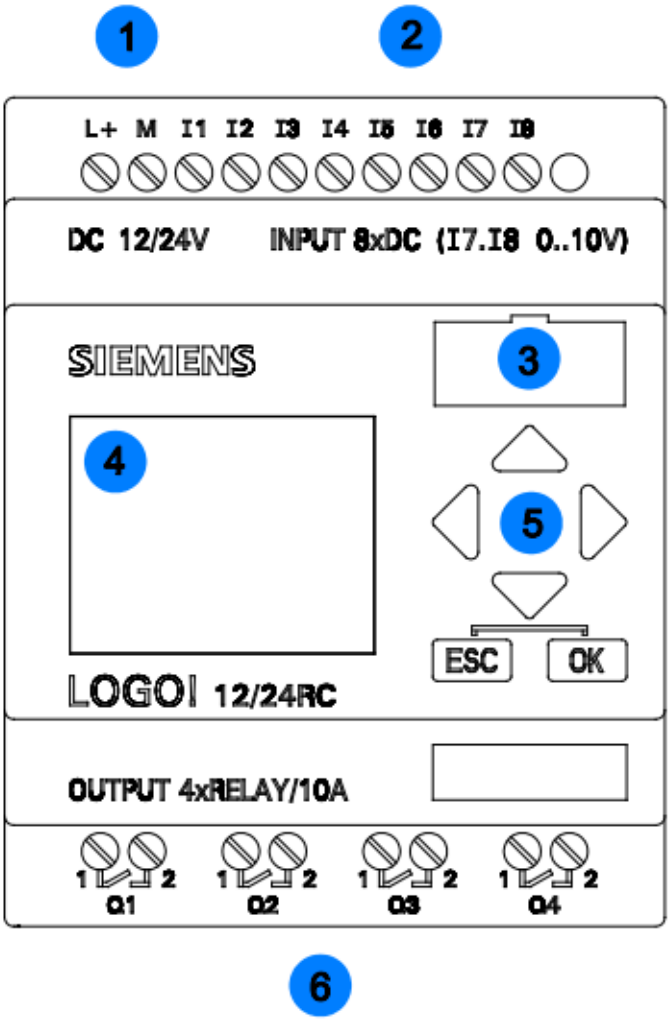
Obrázek 18: Jazyk ST

3 Praktická část

3.1 Logický modul LOGO!

Logický modul LOGO! je universální rozhraní, které zahrnuje napájecí zdroj, řídicí jednotku, digitální vstupy a výstupy a ovládací panel v jednom přístroji. Díky slotu pro paměťový modul lze jednoduchým způsobem zaměnit program za jiný výměnou paměťové karty, nebo použít bateriovou kartu pro zálohování reálného času, či použít kombinovanou kartu spojující obě předcházející v jedinou. Do tohoto slotu se připojuje také programovací LOGO! USB PC kabel. Díky možnosti připojení rozšiřujících modulů si lze přizpůsobit základní konfiguraci své potřebě.

Logický modul LOGO! je výhodná náhrada za řízení pomocí stykačů, kde dokáže nahradit „skříň plnou stykačů“ jediným modulem. Výsledkem je úspora místa v rozvaděči, mnohdy levnější realizace , snazší zapojení úlohy i hledání chyb v logice programu. Využití najde jak v domácím prostředí (řízení zahradního osvětlení, garážových vrat, žaluzií a dalších), tak i v průmyslovém prostředí pro řízení jednodušších procesů (ovládání pásového dopravníku, plnících systémů).



Obrázek 19: Logický modul LOGO!

Základními částmi modulu LOGO! jsou:

- **Zdroj (1)**

pro napájení LOGO! můžeme zvolit napěťovou třídu 1 ≤ 24V (lze použít 12/24 V DC, nebo 24 V AC), nebo třídu 2 > 24 V (lze použít 115/240 V AC/DC).

- **Vstupy (2)**

v základní konfiguraci máme k dispozici 8 digitálních vstupů I1 - I8. Vstupy I7 a I8 lze použít jako analogové vstupy. Vstupy I5 a I6 lze použít jako rychlé digitální vstupy pro signály až 1 KHz.

- **Slot pro modul (3)**

je určený pro vložení paměťového modulu, nebo k připojení LOGO! kabelu.

- **LCD display (4)**

máme na výběr verzi s displejem, nebo bez něj. U verze s displejem jej můžeme využít k zobrazení stavu vstupů/výstupů v režimu RUN. V režimu STOP lze například editovat program.

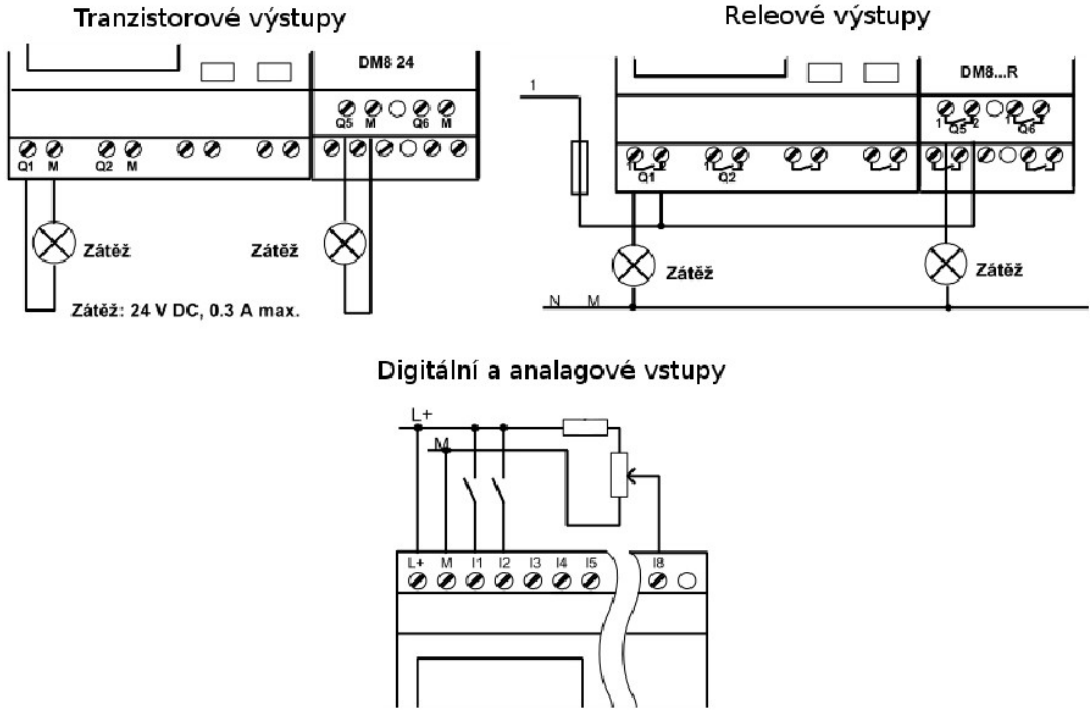
- **Ovládací tlačítka (5)**

pro uvedení modulu do režimu RUN/STOP a pohybu v menu, editaci programu.

- **Výstupy (6)**

v základní konfiguraci máme k dispozici 4 digitální výstupy. Výstup může být reléový (R), nebo tranzistorový (bez R), záleží na naší volbě konfigurace.

Připojení vstupů a výstupů



Obrázek 20: Připojení vstupů a výstupů

3.1.1 Speciální funkce

Na tomto místě bude přiblíženo nastavení speciálních funkcí, které budou využity v laboratorní úloze:

- časovač se zpožděným zapnutím,
- dopředný a zpětný čítač a
- samodržné relé.

U bloků lze nastavit následující vlastnosti:

Remanence

Tato volba zajistí uchování aktuálních dat bloku při výpadku napětí. Hodnoty čítačů a časovačů a stav výstupu budou zachovány i stav samodržného relé bude zachován. Tuto volbu nastavíme ve vlastnostech bloku zatržením položky „Paměť“.

Ochrana parametrů

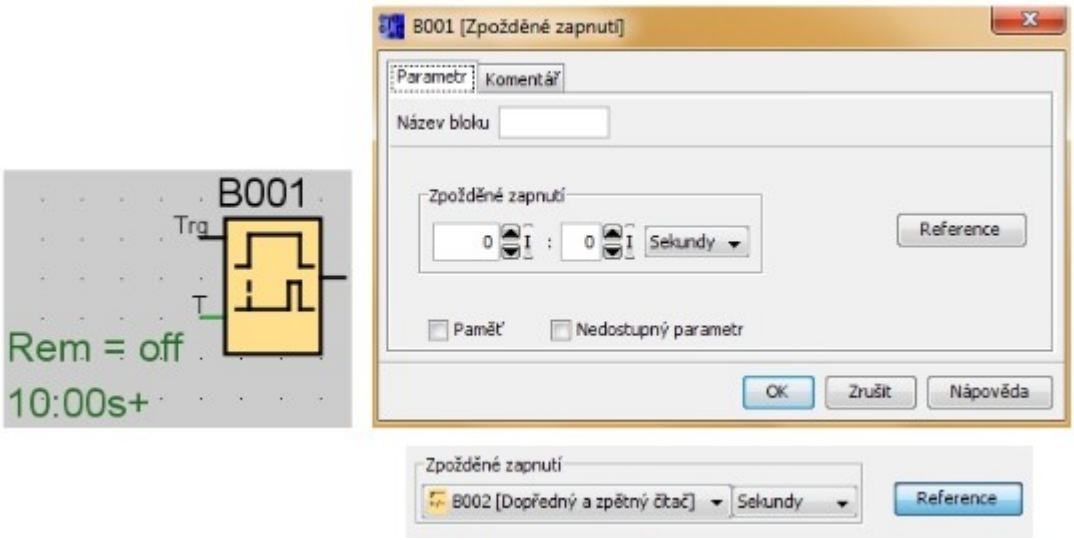
Touto volbou lze povolit/zakázat editaci parametrů v módu přiřazování parametrů přímo na modulu LOGO!.

+: Parametr je přístupný.

-: Parametr není přístupný.

Tuto volbu nastavíme ve vlastnostech bloku zatržením položky „Nedostupný parametr“.

Časovač se zpožděným zapnutím



Obrázek 21: Zpožděné zapnutí

Vstup Trg

Hodnota logická jedna na tomto vstupu spouští časovač.

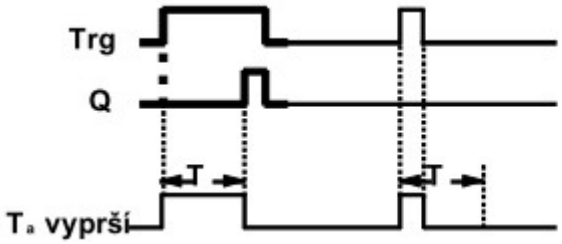
Parametr T

Určuje časový interval, po jehož uplynutí časovač sepne. Časový interval lze zadat v sekundách, minutách, nebo hodinách podle volby z rozbalovacího seznamu.

Časový interval můžeme určit aktuální hodnotou jiné použité funkce (např. čítače). K tomuto nastavení se dostaneme kliknutím na tlačítko „Reference“ a následně zvolíme požadovaný blok ze seznamu.

Indikuje stav remanence.

Časovač se zpožděným zapnutím sepne výstup až po uplynutí časového intervalu. Časování je spuštěno přechodem z logické nuly do logické jedničky na vstupu **Trg**. Pokud se signál na vstupu časovače změní do logické nuly ještě před uplynutím časového intervalu, čas v časovači se resetuje. Po uplynutí zadaného časového intervalu je na výstupu logická jedna tak dlouho, dokud je logická jedna na vstupu **Trg**.



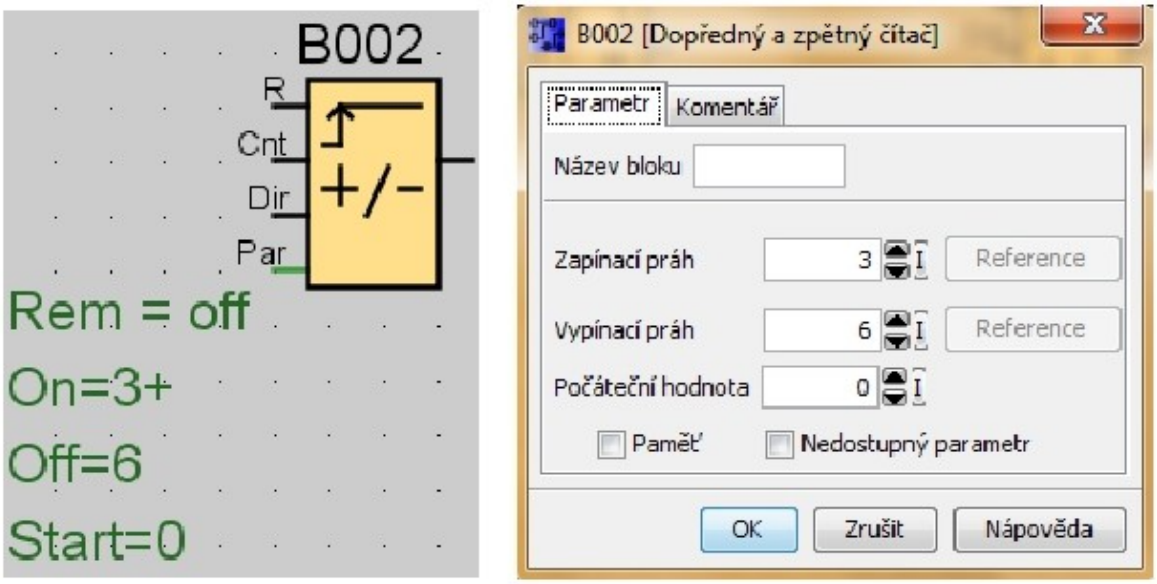
Tučná část časového diagramu je vlastně symbol pro zpožděné zapnutí.

T_a je aktuální čas v LOGO!

T je nastavený časový interval

Obrázek 22: Zpožděné zapnutí – časový diagram

Dopředný a zpětný čítač



Obrázek 23: Dopředný a zpětný čítač

Vstup R

Hodnotou logická jedna vynuluje vnitřní hodnotu čítače.

Vstup Cnt

Na tento vstup se přivádějí počítané impulzy. Počítají se přechody z logické nuly do logické jedničky (náběžné hrany).

Vstup Dir

Určuje směr čítání.

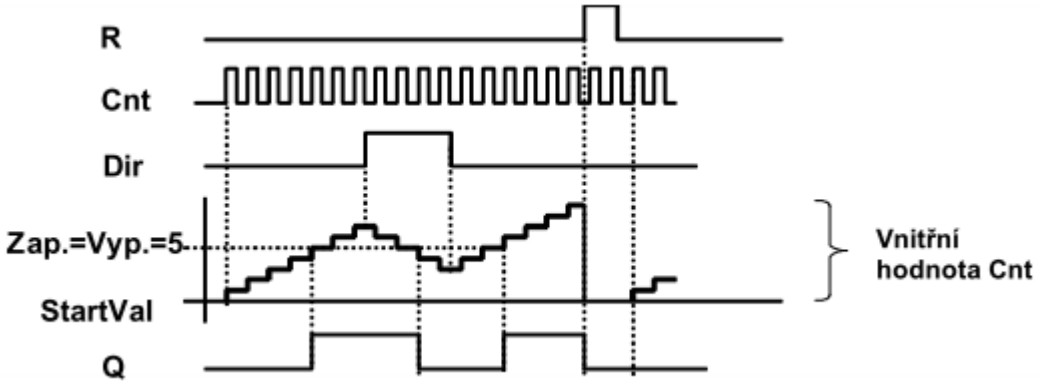
0: vzestupné čítání

1: sestupné čítání

Parametr Par

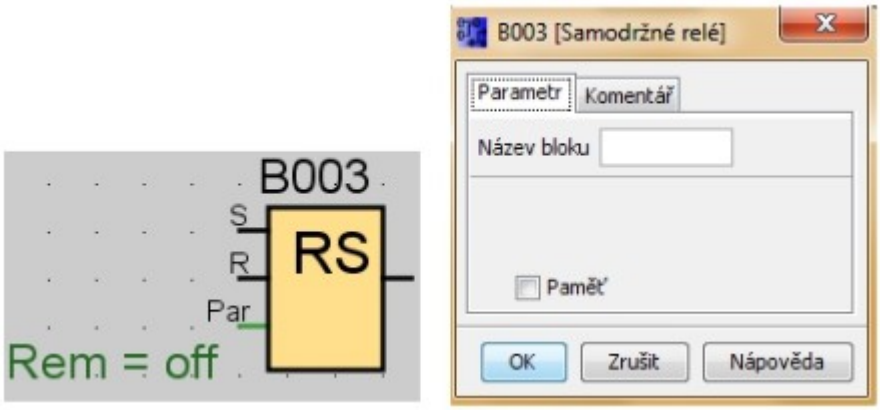
Určuje prahy zapnutí a rozepnutí a počáteční hodnotu, od které se začne čítat.
Indikuje stav remanence.

Čítač počítá každou náběžnou hranu na vstupu **Cnt**. Podle nastavení vstupu **Dir** se buď přičítá, nebo odečítá vnitřní proměnná. Výstup **Q** je spínán/ rozepínán podle nastavení prahových hodnot.
Zapínací práh určuje hodnotu, od které bude výstup sepnut.
Vypínací práh určuje hodnotu, od které bude výstup rozepnut.
Vnitřní stav se nuluje přivedením logické jedničky na vstup **R**.



Obrázek 24: Dopředný a zpětný čítač - časový diagram

Samodržné relé



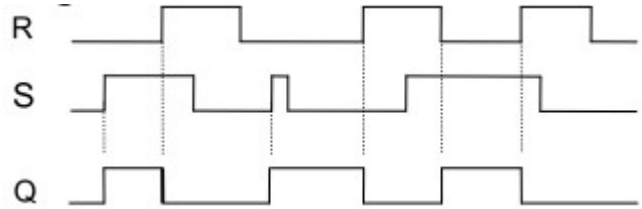
Obrázek 25: Samodržné relé

Vstup R
Hodnota logická jedna rozepíná výstup.
Vstup S
Hodnota logická jedna zapíná výstup.
Parametr Par
Indikuje stav remanence.

Samodržné relé má charakter klopného obvodu RS.

Tabulka 10: Samodržné relé - charakter

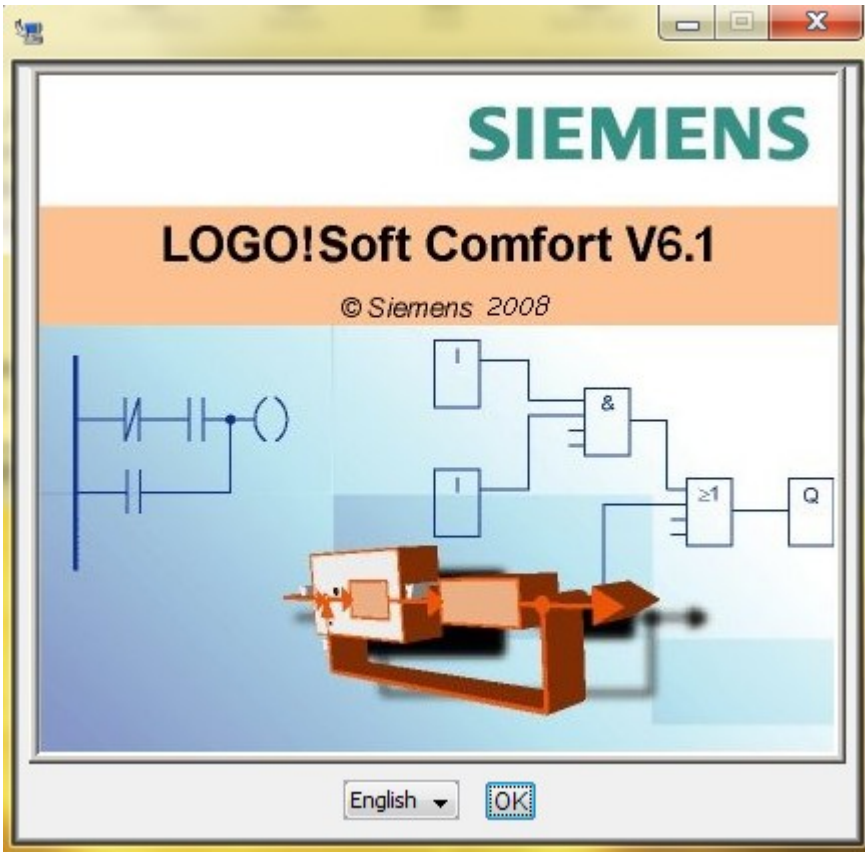
R	S	Q	
0	0	Q	Zůstává původní hodnota
0	1	1	Set – spíná výstup
1	0	0	Reset – rozepíná výstup
1	1	0	Reset – rozepíná výstup (reset má vyšší prioritu)



Obrázek 26: Samodržné relé – časový diagram

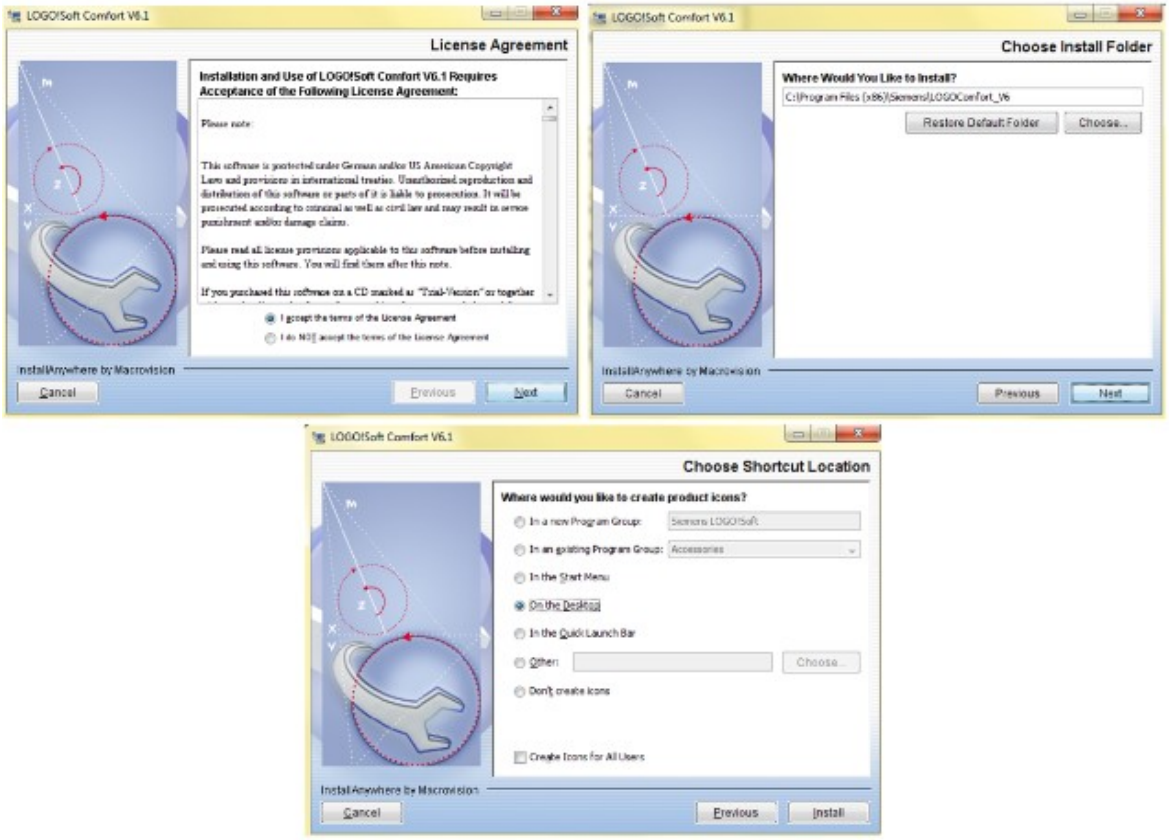
3.2 Vývojového prostředí LOGO!Soft Comfort

3.2.1 Instalace vývojového prostředí LOGO!Soft Comfort



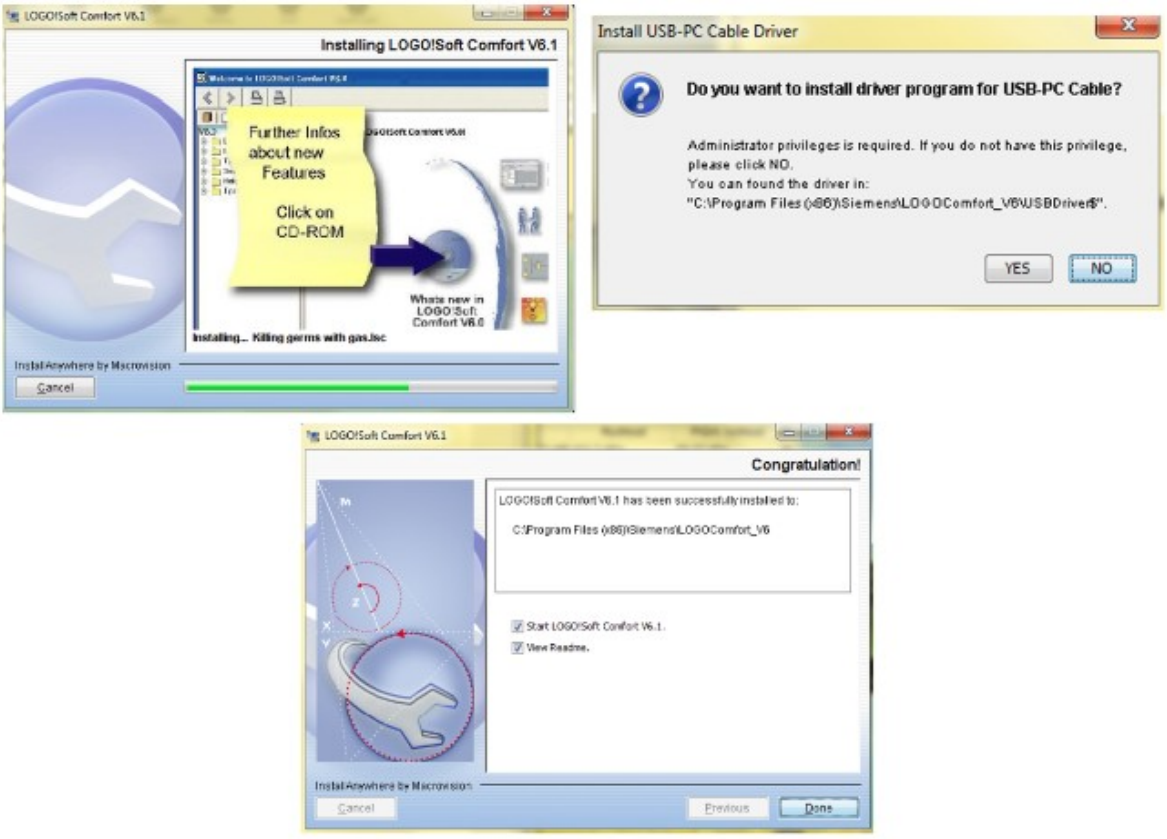
Obrázek 27: Instalace – úvodní okno

Při spuštění instalace se jako první otevře okno s výběrem jazyka instalace, které potvrdíme tlačítkem OK.
Následují tři standardní obrazovky: přijmutí licenčních podmínek, následně volba instalační cesty a nakonec máme možnost vybrat, kde chceme vytvořit ikonu vývojového prostředí.



Obrázek 28: Volby instalace

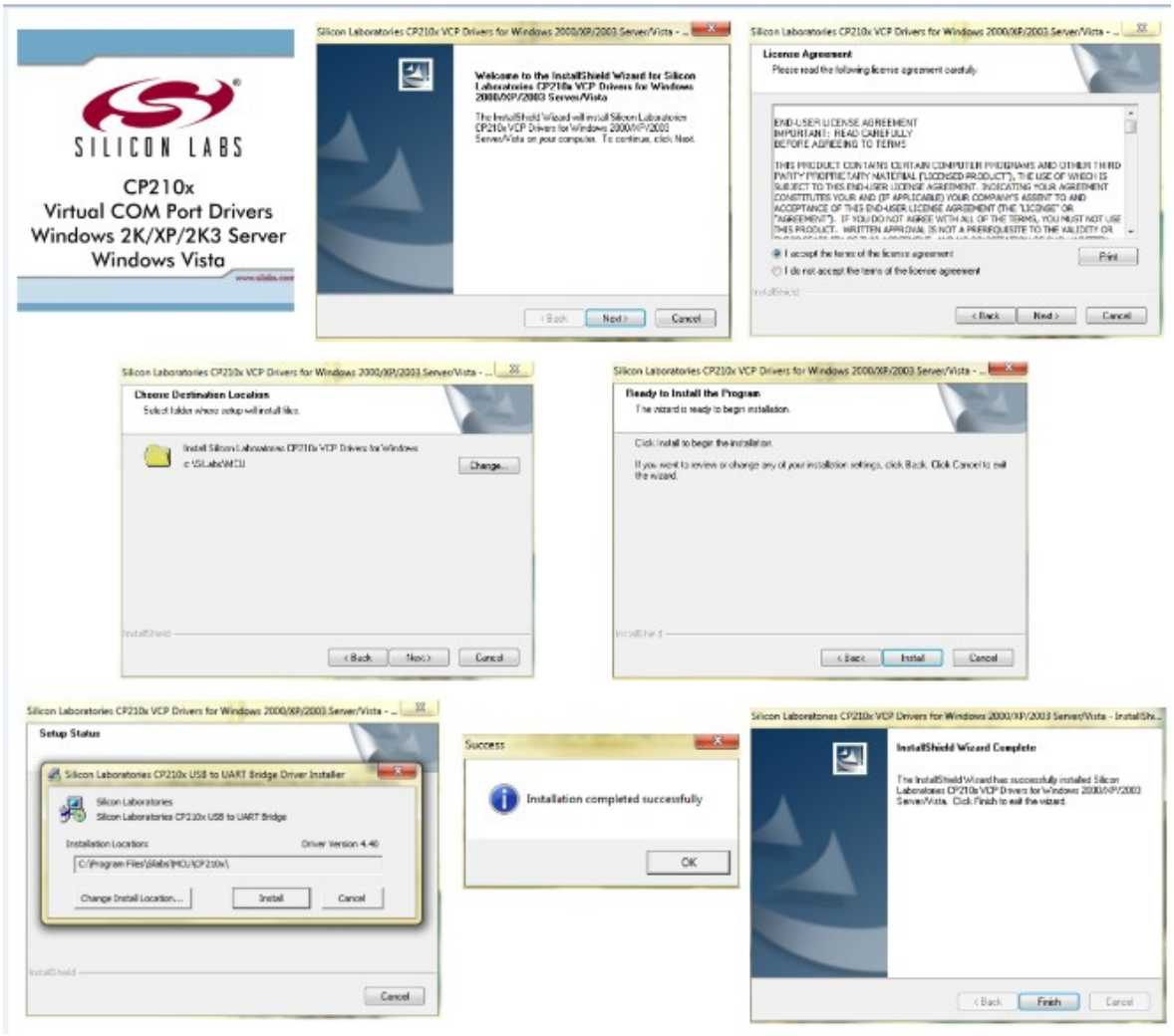
Následuje vlastní instalace vývojového prostředí. Před dokončením instalace se zobrazí žádost o instalaci ovladače k LOGO! USB PC KABELu. Při volbě ano se začne instalovat ovladač k tomuto kabelu, při volbě ne se zobrazí závěrečné okno instalace. Klikneme na tlačítko dokončit a instalace je úspěšně dokončena.



Obrázek 29: Dokončení instalace

3.2.2 Instalace ovladače LOGO! USB PC KABELu

Ovladač pro LOGO! USB PC KABEL je nutné nainstalovat proto, abychom mohli snadno komunikovat s modulem LOGO! pomocí počítače. Výsledný program pak jednoduše nahrajeme do (nebo stáhneme z) modulu LOGO! stisknutím jednoho tlačítka.
Instalace probíhá standardně, potvrdíme všechna okna a instalace je úspěšně dokončena.

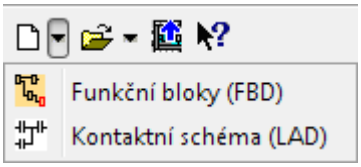


Obrázek 30: Instalace ovladače kabelu

3.2.3 Vytvoření projektu, simulace a přenos programu do modulu LOGO!

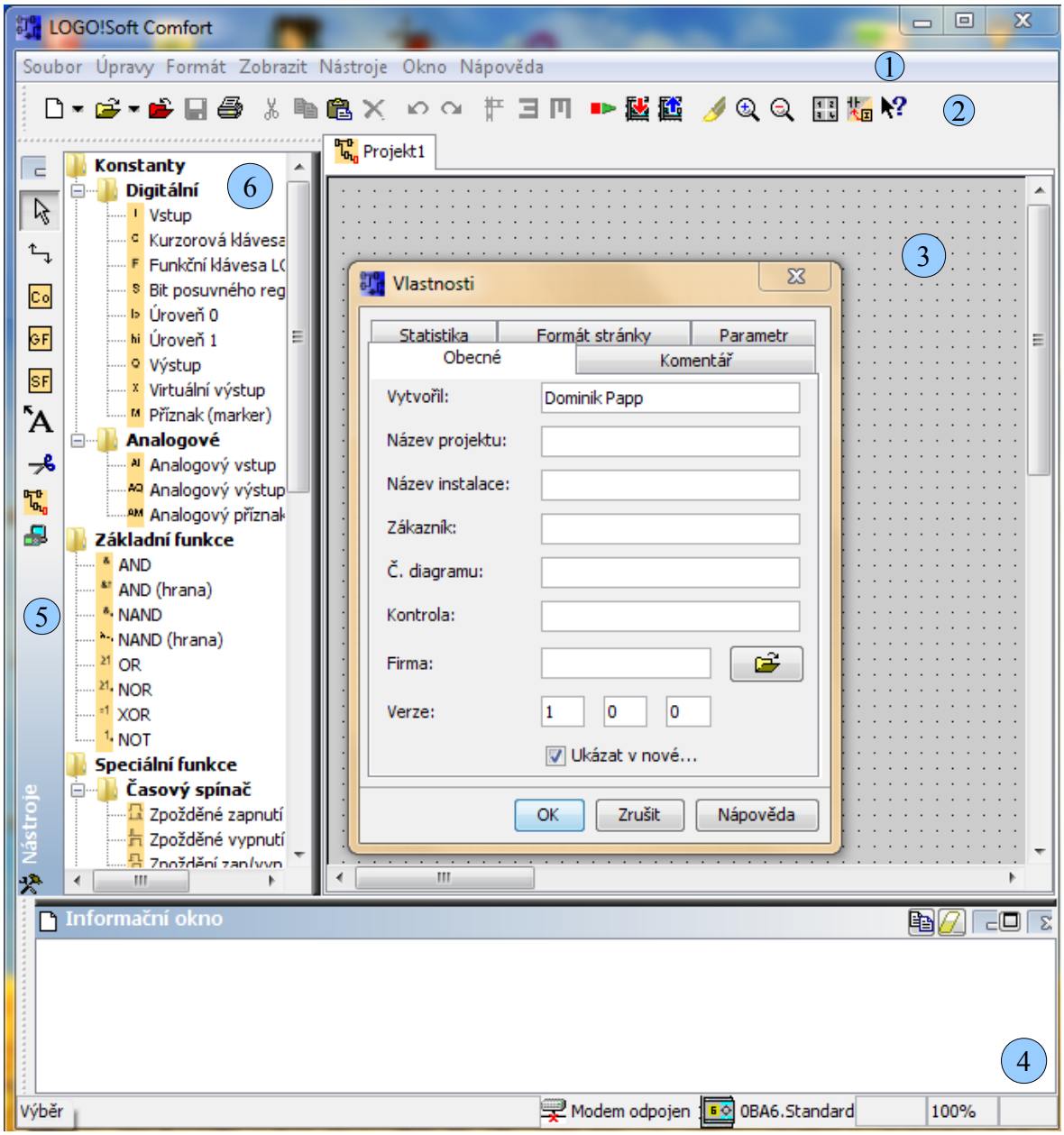
LOGO!Soft Comfort je vývojové prostředí určené pro programování logických modulů LOGO! značky Siemens. K vytvoření programu nabízí dva programovací jazyky, a to jazyk funkčních bloků (FBD), nebo jazyk kontaktních schémat (LD).

Po spuštění programu se otevře prázdné okno, kde máme možnost stáhnout aktuální program přímo z připojeného modulu LOGO!, otevřít rozpracovaný projekt, nebo založit projekt nový. Pro založení nového projektu stačí kliknout na bílou ikonku „Nový“, dojde k vytvoření nového projektu standartě v jazyce FBD. Chceme-li pro náš projekt použít jazyk LD, stačí kliknout na malou šipku po pravé straně ikonky „Nový“, a poté zvolit požadovaný programovací jazyk.



Obrázek 31: LOGO!Soft Comfort – výběr programovacího jazyka

Vlastnosti projektu, kde máme možnost vyplnit si informace o projektu, např. jméno autora, název projektu, číslo verze... Tuto nabídku lze také najít v liště menu Soubor->Vlastnosti. V průběhu vývoje programu se lze mezi oběma jazyky přepínat pomocí ikonky  „Konverze do LAD“ v nástrojové liště, nebo v liště menu Soubor ->„Konverze do LAD“. Dojde k vytvoření nového projektu, kde bude náš dosavadní program převeden do jazyka LD. Konverze zpět do jazyka FBD se prování stejným postupem.



Obrázek 32: LOGO!Soft Comfort – nový projekt

Okno LOGO!Soft Comfort se skládá z následujících oblastí:

- Lišta hlavního menu (1),
- Nástrojová lišta (2),
- Programovací rozhraní (3),
- Informační okno (4),
- Panel programovacích nástrojů (5),
- Seznam programovacích bloků (6).

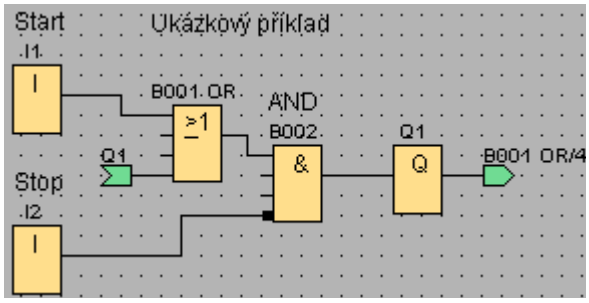
Máme k dispozici bloky vstupů a výstupů, základní logické funkce a speciální funkce.

Bloky vstupů a výstupů
Tyto bloky realizují vstupy a výstupy programu, změnu čísla vstupu/výstupu nalezneme ve vlastnostech bloku (PTM a zvolit „Vlastnosti bloku“, nebo dvakrát poklepat na blok) na kartě „Parametr“. Na kartě „Komentář“ lze připsat komentář. Bloky vstupů mají navíc kartu „Simulace“, kde můžeme nastavit, zda se jedná o přepínač, či spínací, nebo rozpínací tlačítko. Tuto volbu využijeme při simulaci programu.

Základní logické funkce
Zde nalezneme bloky logických funkcí, hradla. Každé hradlo má čtyři vstupy, přičemž nepoužité vstupy není nutné ošetřovat, toto je již zajištěno softwarově. Chceme-li použít negovaný vstup, stačí na zvolený vstupní pin dvakrát poklepat, ten se změní v černý vyplněný čtvereček. Hradla můžeme pojmenovat ve vlastnostech bloku na kartě „Parametr“.

Speciální funkce
Zde nalezneme funkce čítače, časovače, samodržné relé a mnoho dalších užitečných funkcí.

Tvorba programu
Program v prostředí LOGO!Soft Comfort se tvoří velice jednoduše. Stačí si v seznamu bloků (4) vybrat požadovaný blok, kliknout na tento blok a následně kliknutím v programovacím rozhraní (3) jej umístit na požadovanou pozici. Prostředí je nastaveno tak, že umožňuje opakovanou volbu, tedy po umístění bloku do projektu lze dalším kliknutím umístit blok stejného typu. Pro přerušení umisťovaní bloků stačí stisknout klávesu „ESC“, nebo kliknout na ikonu kurzoru v panelu programovacích nástrojů (možnosti tohoto panelu jsou též přístupné po kliknutí pravým tlačítkem myši).
Ke spojení bloků slouží ikona , vlastní spojení se provede tím, že klikneme na požadovaný pin bloku (dojde k jeho zvýraznění modrým čtverečkem) a za stálého držení tlačítka myši najedeme na pin druhého boku. Chceme-li program více zpřehlednit, protože cesta příliš dlouhá, nebo se kříží s jinými cestami, lze tento spoj přerušit. Klikneme na něj pravým tlačítkem myši a zvolit „Rozdělit spoj“, čímž se místo spoje vytvoří zelené bloky symbolizující začátek a konec cesty. Pro opětovné spojení cesty se zvolí „Připojit spoj“.
V programu lze vytvářet komentáře, čímž se lze lépe orientovat ve „shlucích“ bloků našeho projektu. Slouží k tomu ikona a následné kliknutí na vybrané místo. Komentář lze editovat tak, že se namísto volné plochy klikneme na komentář, který chceme upravit.

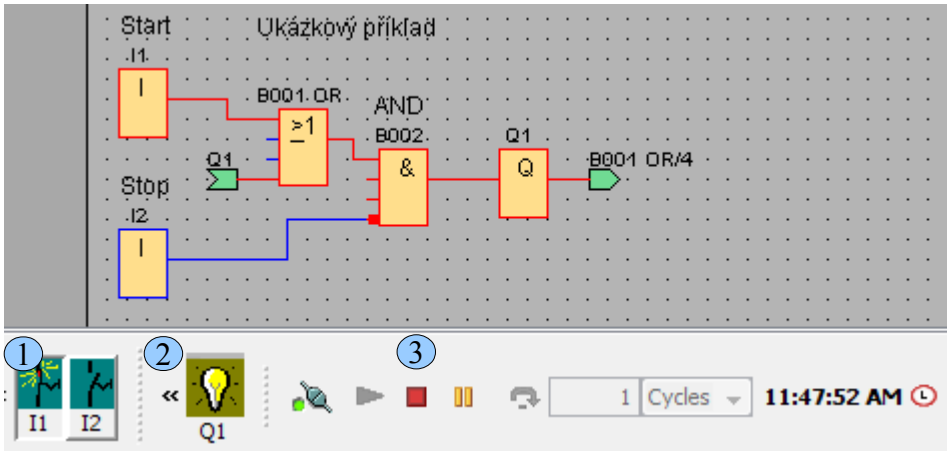


Obrázek 33: Ukázkový příklad

Simulace programu

Vývojové prostředí LOGO!Soft Comfort obsahuje zabudovaný nástroj umožňující simulovat náš program bez nutnosti jeho běhu v zařízení. Můžeme tak program odzkoušet a předejít riziku poškození modulu LOGO!, nebo řízeného stroje vlivem chyby v programu.

Simulaci spustíme v hlavním menu na záložce „Nástroje“ volbou „Simulace“, nebo klávesou „F3“ a ukončíme opětovným kliknutím volbu „Simulace“ (či klávesou „F3“). Cesty zvýrazněné červenou barvou odpovídají stavu logická jedna („1“) a cesty zvýrazněné modrou barvou odpovídají stavu logická nula („0“).



Obrázek 34: Simulace programu

Lišta simulace obsahuje:

- Skupina vstupů (1) – zde nalezneme všechny použité vstupy programu, lze je ovládat kliknutím na vybraný vstup
- Skupina výstupů (2) – zde vidíme stavy na všech použitých výstupech
- Ovládání simulace (3)
  - umožňuje simulovat výpadek napětí, při stisknutí tohoto tlačítka dojde k odpojení všech vstupů a vyplutí výstupů, vynulovány budou také hodnoty čítačů, časovačů i samodržné relé.
  - tlačítka umožňující spustit, zastavit, nebo pozastavit simulaci.
  - je aktivní v režimu pozastavené simulace, umožňuje krokovat simulaci a to buď tak, že se vykoná zadaný počet cyklů, nebo lze z nabídky vybrat délku běhu simulace v sekundách, minutách, nebo hodinách.

- čas v pravé části je reálný čas, lze jej nastavit kliknutím na ikonu  . Toto využijeme při simulaci programu obsahující např. denní časovač.

Přenos programu
Jakmile máme program napsaný a odsimulovaný můžeme jej nahrát do modulu LOGO!. Připojíme modul LOGO! a počítač pomocí USB PC propojovací kabelu. Typ modulu se po připojení nastaví automaticky, případně jej lze určit v záložce „Nástroje“ volbou „Stanovit typ LOGO!“ (klávesová zkratka F2), či volbou „Výběr typu přístroje“ (klávesová zkratka CTRL+H). Možnosti přenosu programu nalezneme v hlavním menu v záložce „Nástroje“ volbou „Přenos“, nebo tlačítky z nástrojové lišty.  Přenos programu je možný, je-li modul LOGO! V režimu STOP.

Lišta přenosu programu obsahuje:

 slouží k přepínání stavu modulo LOGO! Mezi stavy STOP a RUN

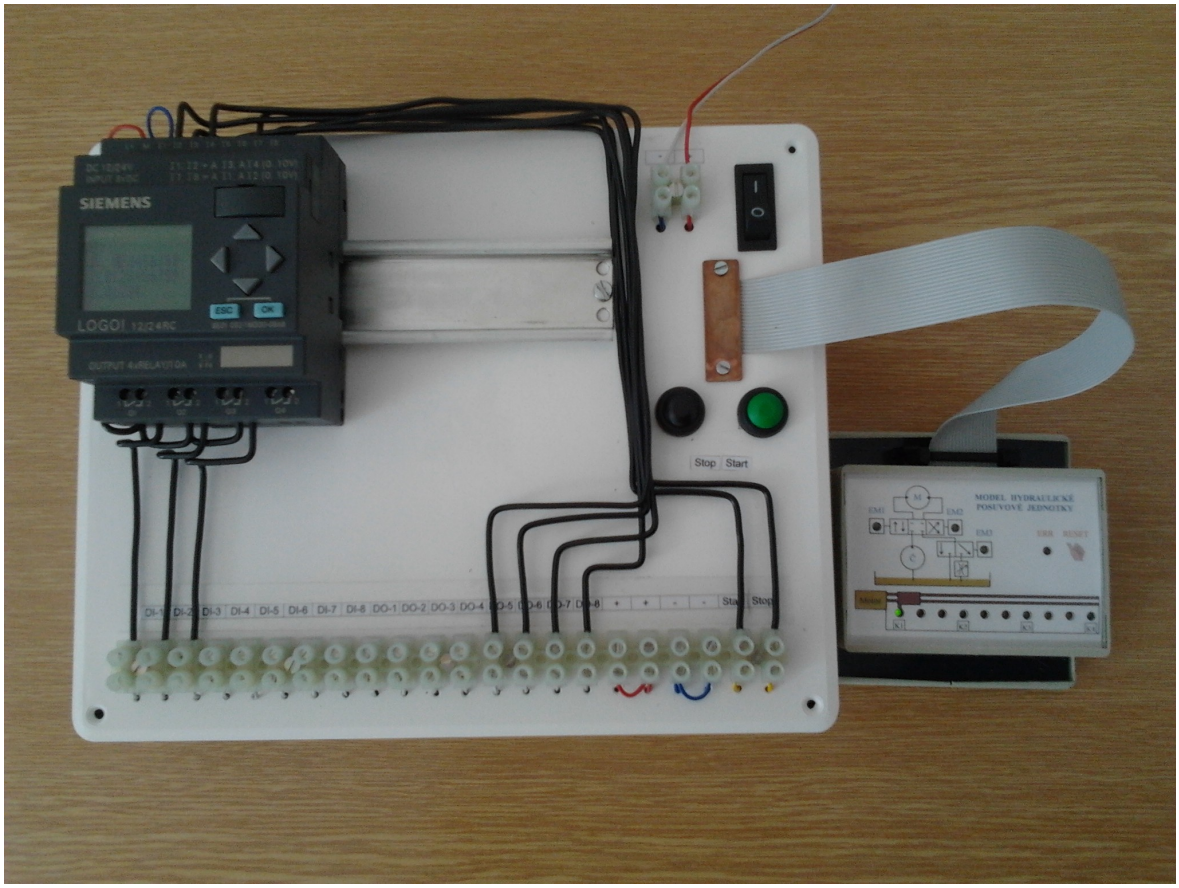
 provede stažení programu z modulu LOGO! do počítače a otevře jej jako nový projekt. Původní program v modulu LOGO! zůstane zachován.

 vyšle náš program do modulu LOGO!. Původní program v modulu bude přepsán tímto programem.

3.3 Laboratorní úloha – Model Hydraulické pohybové jednotky

3.3.1 Modul pro připojení úlohy EDU-mod

Výukový model umožňuje jednoduché propojení logického modulu LOGO! spolu s modely soustav Edu-mod řady 24V. Jednotlivé vstupy a výstupy z modelu jsou vyvedeny na svorkovnici. Svorky označené **DI** jsou vstupy do modelu (řízení motoru Q1), a výstupy z LOGO!. Svorky označené **DO** jsou výstupy z modelu (koncová čidla), tedy jsou vstupy do LOGO!. Na svorkovnici se dále nacházejí dvě dvojice svorek s přivedeným napájecím napětím. Součástí modelu jsou dvě tlačítka použitelná např. jako tlačítka start a stop, jejichž vývody se nalézají na konci svorkovnice.



Obrázek 35: Modul pro připojení úlohy EDU-mod

Přiřazení zdířek svorkovnice propojovacímu kabelu

Tabulka 11: Kabel - svorkovnice

20ti žilový kabel	Svorkovnice	20ti žilový kabel	Svorkovnice
1	GND	11	DI-7
2	+	12	DI-8
3	DI-1	13	DO-1
4	DI-2	14	DO-2
5	DI-3	15	DO-3
6	DI-4	16	DO-4
7	GND	17	DO-5
8	+	18	DO-6
9	DI-5	19	DO-7
10	DI-6	20	DO-8

3.3.2 Model Hydraulické posuvné jednotky

Obrázek 36: Model Hydraulické posuvné jednotky

Jedná se o jednu úlohu řady soustav EDU-mod [7]

Funkce modelu

- Pohyb suportu je simulován pomocí deseti LED diod, z toho čtyři mají zároveň funkci snímačů polohy K1 až K4.
- Model je řízen třemi výstupními signálovými bity. Výstup EM1 řídí pohyb suportu vpřed, EM2 řídí pohyb vzad a EM3 ovládá dvoupolohově rychlost (EM3 = 1 pracovní posuv).

Inicializační stav

- Po zapnutí napájení nebo po restartu (tlačítko RESET) se suport automaticky nastaví do inicializačního stavu - pozice na snímači K1.

Chybová hlášení

Model dokáže vyhodnotit dva druhy funkčních chyb:

- Přejezd krajního snímače (K1, K4)
  - Rozsvítí se červená LED ERR a zelené LED snímačů K1 až K4. Systém se vrátí do výchozího stavu po stisku tlačítka RESET.
- Současné sepnutí EM1 a EM2
  - Signalizační dioda ERR bliká, po odstranění chybového stavu systém bez restartu pokračuje v činnosti.

Přiřazení vstupů a výstupů

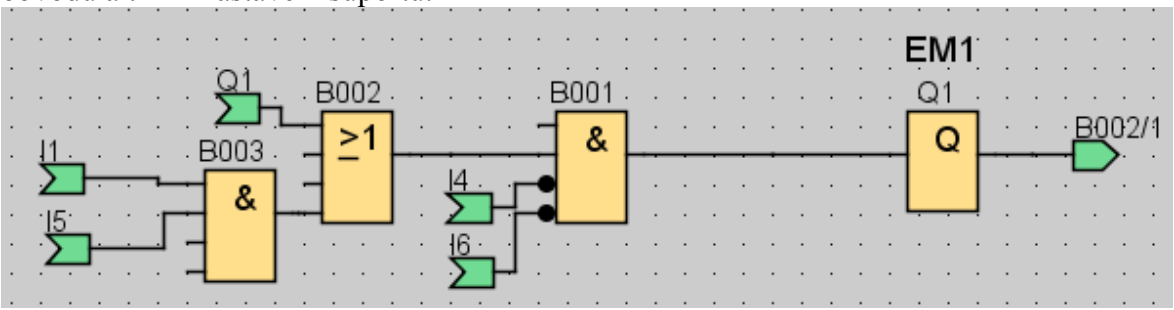
Tabulka 12: Přiřazení vstupů a výstupů

Proměnná	Svorka modelu	Typ
EM1	1	Výstup
EM2	2	Výstup
EM3	3	Výstup
K1	13	Vstup
K2	14	Vstup
K3	15	Vstup
K4	16	Vstup

3.3.3 Laboratorní úlohy

Příklad 1
Vytvořte program pro řízení hydraulické pohybové jednotky podle zadání:
Po stisku tlačítka START se suport rozjede z výchozí pozice na snímači K1 rychloposuvem vpřed (EM1). Zastaví na koncovém snímači K4. Tlačítkem STOP lze pohyb suportu kdykoli zastavit. Přesun do výchozí polohy je zajištěn obsluhou suportu (restart modulu).

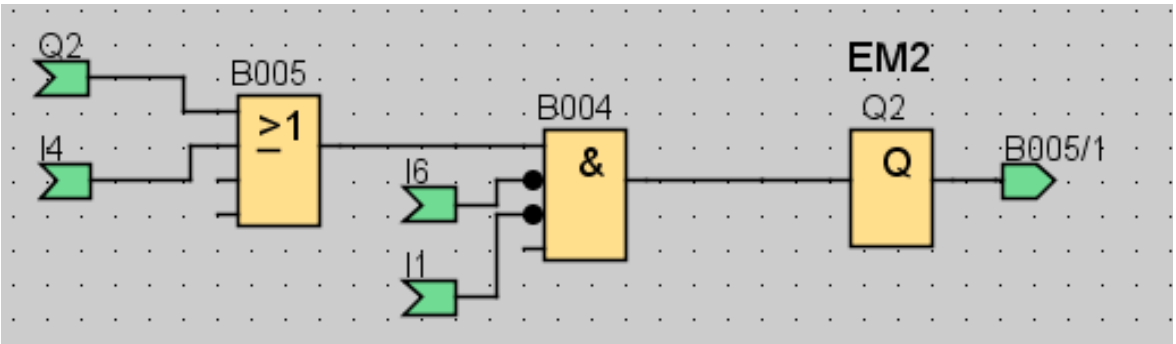
Řešení
Suport se smí rozjet jen tehdy, bude-li se nacházet ve výchozí pozici (snímač K1). Bude-li se zde nacházet a zároveň stiskneme tlačítko START, smí se rozjet vpřed (EM1 = 1). Aby se suport po uvolnění tlačítka START nezastavil vytvoříme smyčku tak, že výstup z EM1 přivedeme spolu se spouštěcí podmínkou do součtového hradla. Pro zastavení suportu na stisk tlačítka STOP, nebo po dojetí na koncové čidlo K4 vložíme před výstup EM1 rozpínací kontakt (negovaný vstup do hradla) těchto prvků, sepne-li se, dojde k přerušení obvodu a tím k zastaveni suportu.



Obrázek 37: Pohyb suportu vpřed

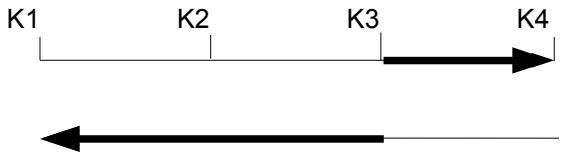
Příklad 2
Pokračujte v prvním příkladě. Po najeti na koncoví snímač (K4) se suport rozjede rychloposuvem vzad (EM2) a zastaví se na počátečním snímači K1. Vytvořte tak základní pracovní cyklus.

Řešení
Pohyb suportu vzad (EM2) realizujeme podobným principem, jako vpřed. Pohyb vzad bude zahájen najetím na snímač K4.



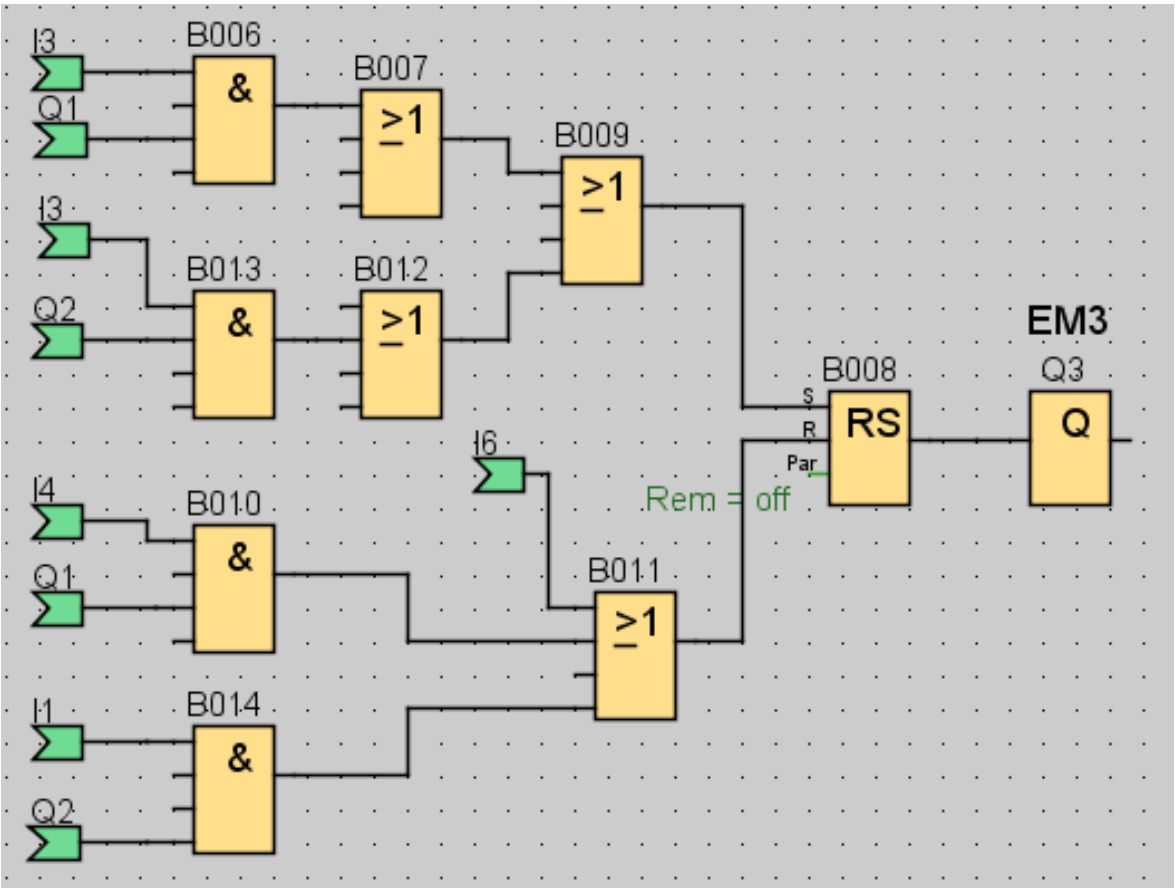
Obrázek 38: Pohyb suportu vzad

Příklad 3
Modifikujte základní pracovní cyklus tak, že při pohybu VPŘED (EM1) bude mezi čidly K3 až K4 prováděn pracovní pohyb (EM3 = 1). Při pohybu VZAD (EM2) bude prováděn pracovní pohyb mezi čidly K3 až K1. Podle diagramu.



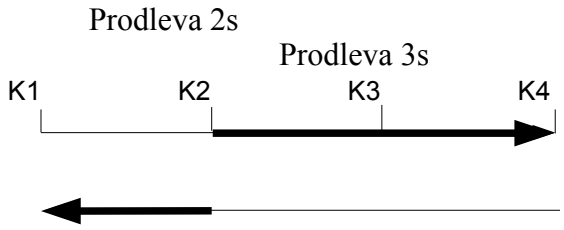
Obrázek 39: Diagram př. 3

Řešení
Spojením obou předchozích příkladů nám vznikne pracovní cyklus.
K ovládání rychlosti pohybu suportu pomocí výstupu EM3 můžeme využít samodržné relé. Nastavení relé se provede při pohybu vpřed a projetí přes snímač K3, nebo při pohybu vzad a projetí přes snímač K3. Tímto se aktivuje pracovní posuv.
Resetování relé se provede při pohybu vpřed a projetí přes snímač K4, nebo při pohybu vzad a projetí přes snímač K1. Tímto se aktivuje rychloposuv.



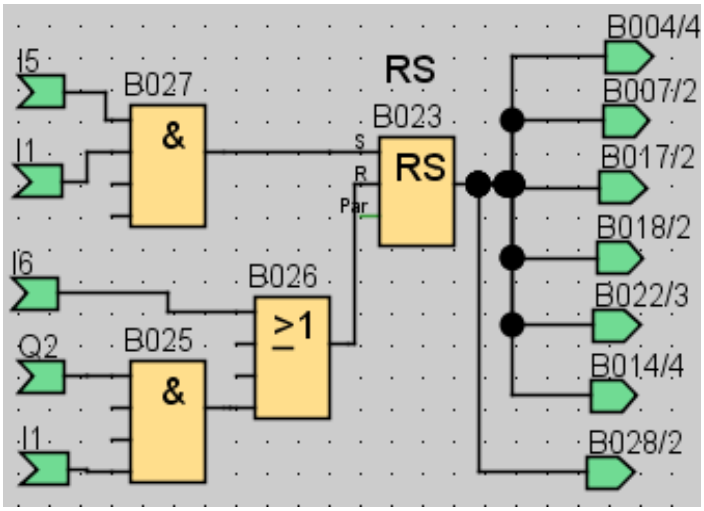
Obrázek 40: Př.3 - ovládání EM3

Příklad 4
Modifikujte základní pracovní cyklus tak, že při pohybu VPŘED (EM1) se na čidlech K1, K2, K3 a K4 suport zastaví na 1, 2, 3 a 4 sekundy. Mezi čidly K2 až K4 je prováděn pracovní pohyb. Při pohybu VZAD (EM2) je prováděn pracovní pohyb (EM3 = 1) mezi čidly K2 až K1. Podle diagramu.



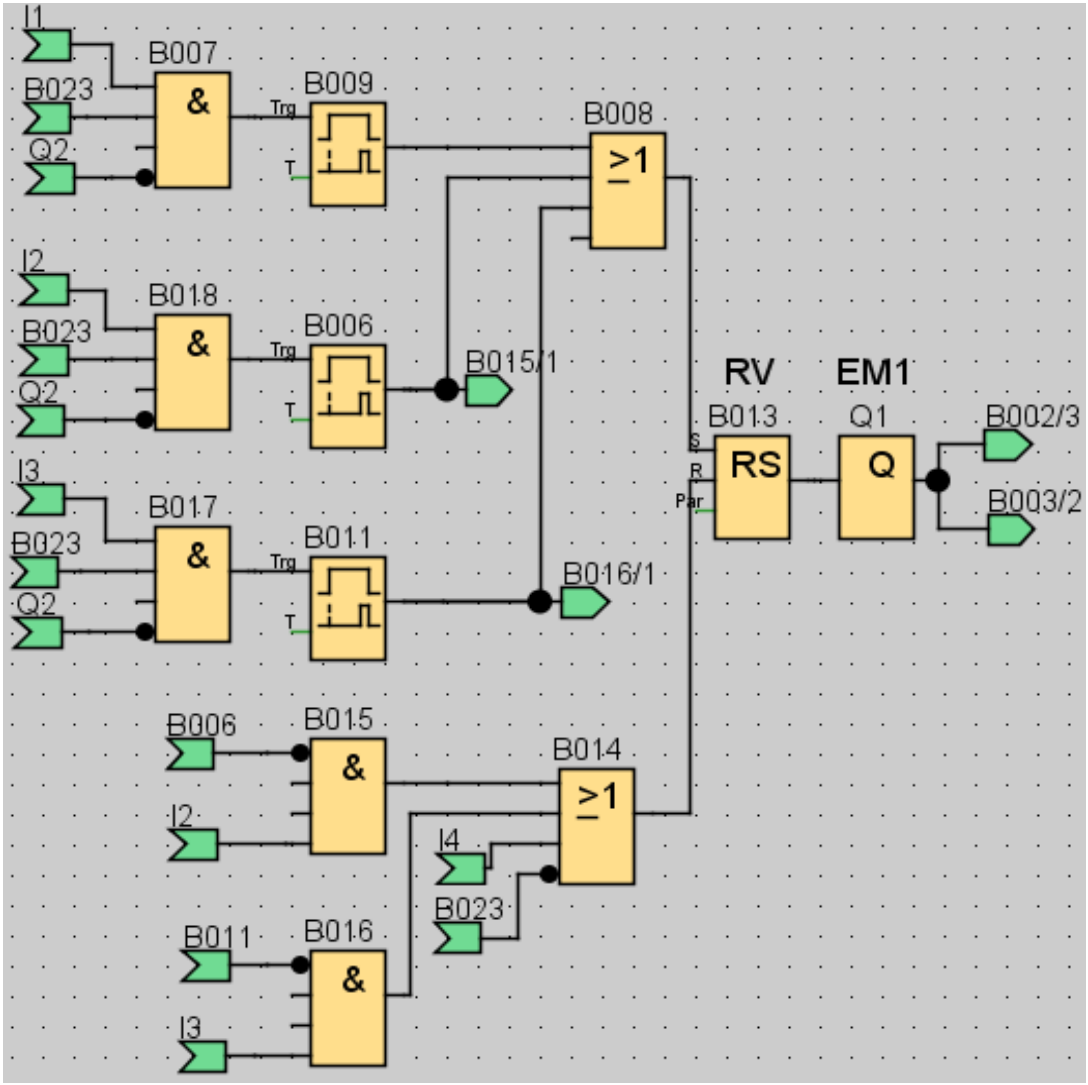
Obrázek 41: Diagram př. 4

Řešení
START a STOP suportu.
Stav suportu (v provozu / zastaven) si zapamatujeme za pomocí samodržného relé RS, který použijeme pro ovládání pohybu. Zamezíme tím spuštění časovačů v případě, že byl suport zastaven právě na jednom ze snímačů.
Rozjetí suportu musí být možné jen z výchozí pozice na snímači K1 a následným stiskem tlačítka START – nastavení RS. Suport lze okamžitě zastavit tlačítkem STOP, nebo po dokončení pracovního cyklu opět na snímači K1 – nulování RS.

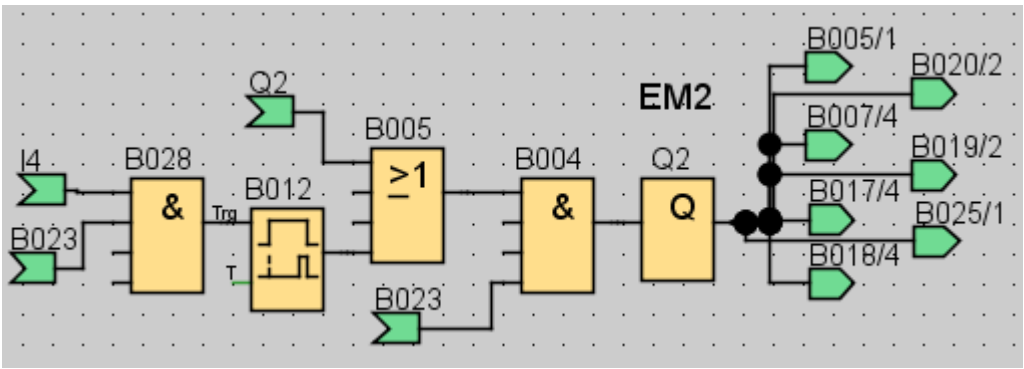


Obrázek 42: Př.4 - START a STOP suportu

Ovládání pohybu VPŘED.
Pohybu vpřed lze realizovat také přes samodržné relé RV. Jeho nastavení, tedy rozjetí suportu, provedou časová relé se zpožděným přítahem, která jsou aktivována podmínkou – suport se nalézá na snímači a zároveň se nepohybuje VZAD a zároveň je nastaveno relé startu programu RS. Na jednotlivých čidlech musí suport zastavit a pokračovat až po uplynutí dané časové prodlevy. K jeho zastavení dojde vynulováním relé RV, tedy bude se nalézat na daném snímači a zároveň nebude sepnuto časové relé, nebo se bude nalézat na koncovém snímači K4. Nulování zajistí též nenastavené relé RS (B023).



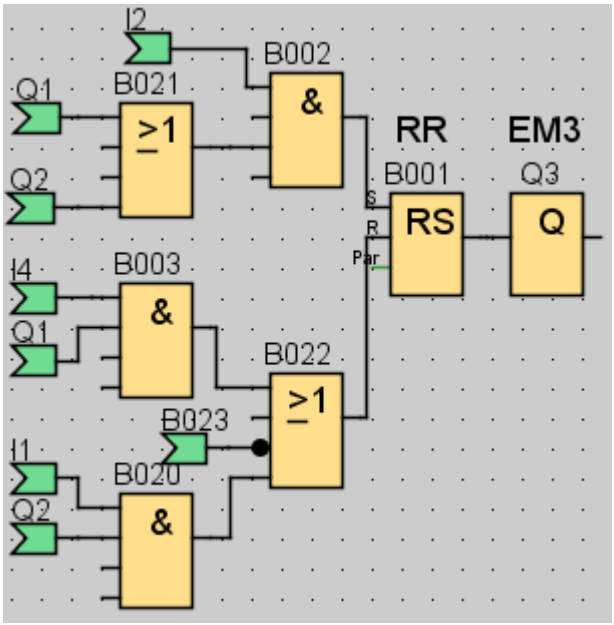
Obrázek 43: Př. 4 - pohyb vpřed



Obrázek 44: Př. 4 - pohyb vzad

Ovládání pohybu VZAD.
Suport se má rozjet zpět po 4 sekundové prodlevě na snímači K4, použijeme časové relé se zpožděným přítahem s nastavenou dobou přítahu po 4 sekundách. Po této době relé sepne a suport se rozjede VZAD, aby se po opuštění čidla K4 nezastavil, neboť také dojde k rozepnutí časového relé, použijeme součtové hradlo na jehož vstupech bude výstup časového relé a výstup EM2. Aby bylo možné suport zastavit, vložíme mezi toto hradlo a výstup EM2 součinové hradlo provádějící součin se stavem relé RS (B023).

Ovládání rychlosti posuvu.
K ovládání rychlosti posuvu použijeme samodržné relé RR, které se nastaví na snímači K2, pohybuje-li se suport VPŘED, nebo VZAD. Vynuluje na snímačí K4, pohybuje-li se suport VPŘED, nebo na snímači K1, pohybuje-li se suport VZAD, nebo bude-li rozepnuto relé KS (použijeme negovaný vstup hradla).



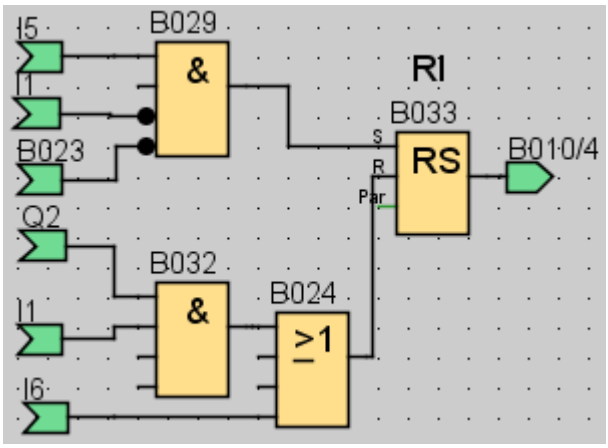
Obrázek 45: Př. 4 - rychlost posuvu

Příklad 5

Pokračujte v příkladě 4. Na stisk tlačítka START suport najede do výchozí pozice na snímači K1. Na další stisk tlačítka START se zahájí pracovní cyklus. Pokud se suport nalézá ve výchozí pozici tak je po stisku tlačítka START ihned zahájen pracovní cyklus.

Řešení

Pro najetí suportu do výchozí pozice na stisk tlačítka START využijeme samodržné relé RI, které se nastaví pouze nebude-li se suport nacházet na snímači K1 při stisku tlačítka START a zároveň nebude-li již nastavené relé RS (suport již v pohybu). Nulování relé RI zajistí najetí na snímač K1 při pohybu VZAD, nebo stisk tlačítka STOP. Neboť je výchozí pozice na snímači K1, který je v levé krajní pozici stačí suport rozjet VZAD. K ovládání pohybu VZAD přidáme součtové hradlo před výstup EM2, kam navíc přivedeme výstup relé RI.



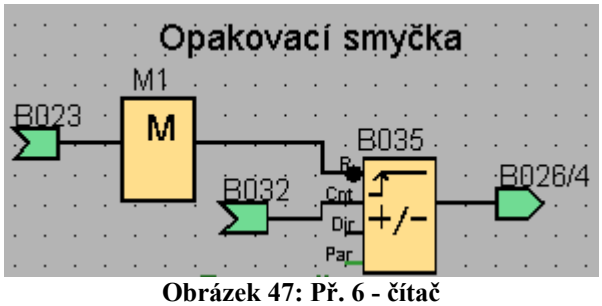
Obrázek 46: Př. 5 - inicializace

Příklad 6

Pokračujte v příkladě 5. Po stisku tlačítka START se pracovní cyklus provede 3-krát po sobě.

Řešení

K zopakování pracovního cyklu použijeme čítač s nastavenou hodnotou zapnutí na 3. Na vstup čítače Cnt přivedeme součin snímače K1 a pohybu VZAD, který přivedeme z inicializačního bloku B032. K resetování čítače použijeme výstup z relé RS přivedený přes příznak M (výstupem příznakového bloku je jeho vstupní signál) na negovaný vstup R. Příznak se musí použít, protože relé RI ovládáme také čítačem (vznikla by rekurze) a prostředí dovoluje rekurze jen přes výstupy, nebo příznaky. Výstupem z čítače nahradíme součin EM2 a K1, který nuluje relé RS. Relé RS bude tedy nulovat výstup čítače, nebo stisk tlačítka STOP.



Obrázek 47: Př. 6 - čítač

4 Závěr

Cílem bakalářské práce bylo vytvořit laboratorní úlohu pro logický modul LOGO! za použití modelu EDU-mod Model hydraulické posuvné jednotky. Ze všeho nejdřív jsem musel prostudovat technickou dokumentaci modulu LOGO!, kde jsem potřeboval zjistit použitelné napájecí napětí modulu a způsob připojení vstupů a výstupů. Poté bylo nutné zjistit napájecí napětí pro modely EDU-mod a případné upravit toto napětí na požadovanou hodnotu. Modely EDU-mod i modul LOGO! používají stejné napájecí napětí (24 V), tudíž přizpůsobení nebylo nutné. Poté bylo potřeba zkonstruovat samotný modul pro připojení soustavy EDU-mod, který zajišťuje snadné propojení logického modulu LOGO! s modely EDU-mod.
Nejprve jsem nainstaloval vývojové prostředí LOGO!Soft Comfort a s ním také ovladač k LOGO! USB PC KABELu, přes který se nahrává program do logického modulu LOGO!. Vývojové prostředí je celkem jednoduché a intuitivní. Prostředí je v českém jazyce a má k dispozici i nápovědu v češtině a uživatel se v něm relativně rychle zorientuje. Programuje se v jazyce FBD, vytvoření programu je tedy relativně snadné. Velmi užitečnou funkcí je simulace programu, kde lze program odladit a otestovat dříve než bude přenesen do logického modulu.

Bakalářská práce je rozdělená do čtyřech kapitol, teoretická a praktická kapitola rozdělena do několika podkapitol.
V teoretické části bakalářské práce je popsána v první podkapitole teorie logického řízení a v druhé podkapitole programovatelné logické automaty. V praktické části věnována první podkapitola vývojovému prostředí LOGO!Soft Comfort, jeho instalaci a základnímu popisu. Ve druhé podkapitole je přiblížen logický modul LOGO!, model hydraulické posuvné jednotky a model pro připojení soustav EDU-mod. Na konci této podkapitoly je uvedena laboratorní úloha se zadáním a s popisem řešení. Tato úloha je rozdělena do šesti částí, jež na sebe navazují.
Programy pro jednotlivé úlohy jsou uloženy na přiloženém CD.

Literatura

[1] ŠMEJKAL, Ladislav a MARTINÁSKOVÁ Marie. *PLC a automatizace: základní pojmy, úvod do programování.* 1. vyd. Praha: BEN - technická literatura, 1999, 223 s. ISBN 80-860-5658-9.

[2] TŮMA, Jiří, WAGNEROVÁ Renata, FARANA Radim a LANDRYOVÁ Lenka. *Základy automatizace* [online]. Ostrava: Vysoká škola báňská - Technická univerzita, 2007, 288 s. [cit. 2012-05-08]. ISBN 978-80-248-1523-7. Dostupné z: http://www.elearn.vsb.cz/archivcd/FS/Zaut/Skripta_text.pdf

[3] *Logické řízení* [online]. 65 s. [cit. 2012-05-08]. Dostupné z: http://autnt.fme.vutbr.cz/lab/a4-603/opory/elr.pdf

[4] ZEZULKA, František, BRADÁČ Zdeněk, FIELDER Petr a kol. *Programovatelné automaty* [online]. Brno, 2003, 79 s. [cit. 2012-05-08]. Dostupné z: http://www.vaeprosys.cz/Dokumentace/Programovatelne_automaty/Programovatelne_automaty-Skripta_FEKT_VUT_Brno.pdf

[5] HERNYCH, Miloš. *Základy logického řízení: část I.* [online]. Liberec. 43 s. [cit. 2012-05-08]. Dostupné z: http://www.fm.tul.cz/~milos.hernych/zlr/ZLR_1.pdf

[6] VONDRA, Zdeněk. *Základy programování PLC* [online]. Vlašim, 2006, 67 s. [cit. 2012-05-08]. Dostupné z: http://www.spsejecna.org/skola/documents/Vysledky_proektu/PLC_zakl.pdf

[7] KOHOUT, Luděk. *Modely EDU-mod* [online]. 2008 [cit. 2012-03-21]. Dostupné z: http://www.edumat.cz/produkty.php?produkt=suport

[8] *SIEMENS. LOGO! Manuál: desáté vydání* [online]. Praha, 2008, 201 s. [cit. 2012-05-08]. Dostupné z: http://www1.siemens.cz/ad/current/content/data_files/automatizacni_systemy/mikrosystemy/logo/zakladni_pristroje/_manualy/manual_logo-0ba6_11-008_cz.pdf