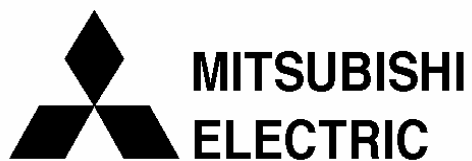


GX IEC Developer

**Programovací a dokumentační
software dle normy IEC 1131.3**



Upozornění

Texty a obrázky uvedené v tomto manuálu slouží pro objasnění struktury programu GX IEC Developer dle normy IEC. Bližší popis instrukcí a technické parametry jednotlivých řad řídicích systémů MITSUBISHI MELSEC jsou uvedeny v samostatných manuálech.

Příklady uvedené v tomto manuálu slouží pouze pro lepší pochopení dané problematiky a nemusí být plně funkční ve všech verzích GX IEC.

MITSUBISHI ELECTRIC si vyhrazuje právo provést změny v technické specifikaci bez předchozího upozornění.

Všechna práva k tomuto manuálu a software jsou vyhrazena firmou MITSUBISHI ELECTRIC a bez jejího souhlasu nesmí být žádné části kopírovány a dále rozšiřovány.

1	Předmluva	1
1.1	Symbole použité v manuálu	2
2	Standard IEC 1131.3	3
2.1	Stavba (struktura) programu	3
2.2	Programovací jazyk	5
2.3	Proměnné	5
3	Základní názvosloví programovacího standardu IEC	6
3.1	Projekt (Project)	6
3.2	Programový modul (Program Organisation Unit –POU)	6
3.3	Programy, funkční bloky a funkce	7
3.4	Parametry (Parameters) a instance	8
3.5	Úlohy (Tasks)	9
3.6	Proměnné (Variables)	10
3.6.1	Definování proměnných (Declaring Variables)	10
3.6.2	Druh (Class)	11
3.6.3	Identifikátory (Identifiers) a absolutní adresy (Absolute Addresses)	11
3.6.4	Datové typy (Data Types)	12
3.6.5	Inicializační hodnota (Initial value)	13
3.6.6	Komentář (Comment)	13
3.6.7	Volba autoextern (Autoextern option)	13
4	Programovací jazyky (Programming Languages)	14
5	Textové editory (Text Editors)	15
5.1	MELSEC instrukční list (MELSEC IL)	15
5.2	IEC instrukční list (IEC IL)	15
5.2.1	Akumulátor	16
5.2.2	Volání funkčních bloků	17
5.2.3	Volání funkce	177
5.3	Strukturovaný text (Structure Text - ST)	158
6	Grafické editory	19
6.1	Kontaktní schéma (Ladder Diagram Language –LD)	19
6.1.1	Volání funkčních bloků	20
6.1.2	Volání funkce	20
6.2	Diagram funkčních bloků	21
6.2.1	Volání funkce a funkčního bloku	21
6.3	Sekvenční funkční diagram editor (Sequential Func. Chart – SFC)	22
6.3.1	Pravidla sekvenčního programu	22
7	Instalace	24
7.1	Hardwarové požadavky pro práci s GX IEC Developerem	24
7.2	Softwarové požadavky pro práci s GX IEC Developerem	24
8	Uživatelské rozhraní	25
8.1	Prvky uživatelského rozhraní	25
8.1.1	Lišta s ovládacím menu (Menu bar)	26
8.1.2	Nástrojová lišta (Toolbar)	26
8.1.3	Okna (Windows)	26
8.1.4	Stavová lišta (Status bar)	26
8.1.5	Navigátor projektem (Project Navigator)	26
8.2	Tabulka deklarácí (Declaration Table)	28
8.2.1	Práce s tabulkou	28
8.2.2	Vymazání celé tabulky nebo vymazání jednotlivých řádků	28
8.2.3	Formátování tabulky	29
8.3	Editory	30
9	Postup při vytváření programu	32
9.1	Vytvoření nového projektu (Creating New Project)	32
9.2	Definování nové úlohy (New Task)	34
9.3	Definování (deklarace) globálních proměnných (global variable)	34
9.4	Vytvoření programového modulu (POU)	36
9.4.1	Vytvoření hlavičky programového modulu (POU)	37
9.4.2	Programování těla (Body) programového modulu (POU)	38

10	Příklady programování.....	39
10.1	Vstupy a výstupy v editoru kontakt. schémat (Ladder diagram – LD).....	39
10.2	Příklady naprogramování funkce součet (SUM) v editoru diagramu funkčních bloků (FBD) ..	41
10.3	Konfigurace parametrů vstupních a výstupních signálů.....	44
10.4	Časovače v editorech LD, FBD a IL	45
10.4.1	Popis proměnných časovače.....	45
11	Příklad časovače.....	46
11.1	Vytvoření programu	46
11.1.1	Naprogramování funkčního bloku SET_RST	46
11.1.2	Definování globálních proměnných	47
11.1.3	Vytvoření nového POU Timer	48
11.1.4	Časový průběh programu	50
11.2	Programování časovačů v editoru diagramu fun. bloků (FBD).....	51
11.3	Programování časovačů v instrukčním listu (IL)	51
11.4	Sekvenční funkční diagram editor.....	52
11.5	Vytvoření programu	53
11.5.1	Vytvoření POU.....	53
11.5.2	Deklarace proměnných v hlavičce	54
11.5.3	Otevření těla	54
11.5.4	Vytvoření sekvence.....	55
11.5.5	Přiřazení jmen ke krokům a návštějí skoku.....	60
11.5.6	Přiřazení přechodových podmínek k přechodům	61
12	Kontrola PLC programu (syntax check)	65
13	Konfigurace tasku	66
13.1	Vytvoření nového tasku	66
13.2	Zařazení POU do tasku	66
13.3	Konfigurace parametrů tasku	67
14	Kompilace projektu.....	68
15	Nastavení komunikačního portu.....	69
16	Nahrání programu do PLC	70
17	Monitorování programu	71
18	Nahrávání programu z PLC	73
19	Vzorový program – Řízení garáží	74
19.1	Zadání	74
19.2	Části programu	74
19.2.1	Vytvoření nového projektu CarPark.....	75
19.2.2	Vytvoření tasků	75
19.2.3	Deklarace globálních proměnných	75
19.2.4	Vytvoření programových modulů (POU).....	76
19.3	Hlavičky programových modulů.....	76
19.3.1	Hlavička programového modulu Control.....	77
19.3.2	Hlavička programového modulu Counter	77
19.4	Těla programových modulů	78
19.4.1	Tělo programového modulu Control.....	78
19.4.2	Tělo programového modulu Counter	79
19.4.3	Tělo programového modulu Door_Control.....	80
19.5	Konfigurace tasků	81
19.5.1	Task Main	81
19.5.2	Task Door_Operate	82
20	Import projektu vytvořeného v MELSEC MEDOCu.....	833

1 Předmluva

Tento manuál...

...je stručným průvodcem software GX IEC Developer (dále jen GX IEC) od prvního spuštění programu až po jeho odladění a nahrání aplikačního programu do řídicího systému. Svým zaměřením je orientován zejména na uživatele, kteří doposud používali starší programovací software MELSEC MEDOC, popř. na uživatele jiných řídicích systémů (softwarů).

V manuálu jsou vysvětleny základní pojmy a koncepce programování dle normy IEC, včetně uvedení jednoduchých příkladů.

Uživatelský manuál...

...na rozdíl od tohoto manuálu není zaměřen na metodiku tvorby programu, ale je orientován na popis příkazů, nabídek a menu v GX IEC.

Nejste-li uživateli MS Windows...

...seznamte se prosím nejprve alespoň se základními funkcemi tohoto operačního systému. Bez těchto základních znalostí bude pro vás práce s GX IEC poměrně obtížná, neboť GX IEC ve velké míře využívá standardní funkce tohoto operačního systému.

Nejste-li seznámeni s normou IEC 1131.3...

...přečtěte si pozorně alespoň kapitolu Standard IEC 1131.3, ve které se seznámíte se základními pojmy a konvencemi tohoto průmyslového standardu.

Jste-li již seznámeni s normou IEC 1131.3...

...můžete výše uvedenou úvodní kapitolu vynechat a rovnou přejít ke vzorovému příkladu, v němž je popsán kompletní postup při vytváření programu.

Dostanete-li se do problémů...

...podívejte se prosím nejprve do tohoto manuálu. V případě, že daný problém není v manuálu zmíněn, použijte kontextový on-line Help, v němž najdete celou řadu dalších rad a informací. Pokud vám ani tento Help nepomůže při řešení vašeho problému, obraťte se na nejbližší zastoupení MITSUBISHI ELECTRIC nebo volejte přímo centrálu v Ratingenu. Telefonní čísla a kontaktní adresy jsou uvedeny na všech firemních materiálech.

1.1 Symboly použité v manuálu

Tento symbol je použit v případě, že příkazy zadané pomocí myši a z klávesnice se liší.



Příklad. Tento symbol je uveden u kapitol s příklady programování.



Upozornění. Tento symbol je umístěn u odstavců hodných zvláštního zřetele (upozornění, užitečné informace, rady).



Varování. Tento symbol je umístěn u příkazů, jejichž aplikace může způsobit ztrátu dat, zničení zařízení nebo jiný vážný problém.



Pro lepší orientaci jsou jména jednotlivých menu, podmenu, příkazů a dialogových oken vytištěna tučně. Např. příkaz **New** v menu **Project**, nebo nabídka **CPU Port** a **Computer Link (AJ71C24)** v dialogovém okně **Transfer Setup**.

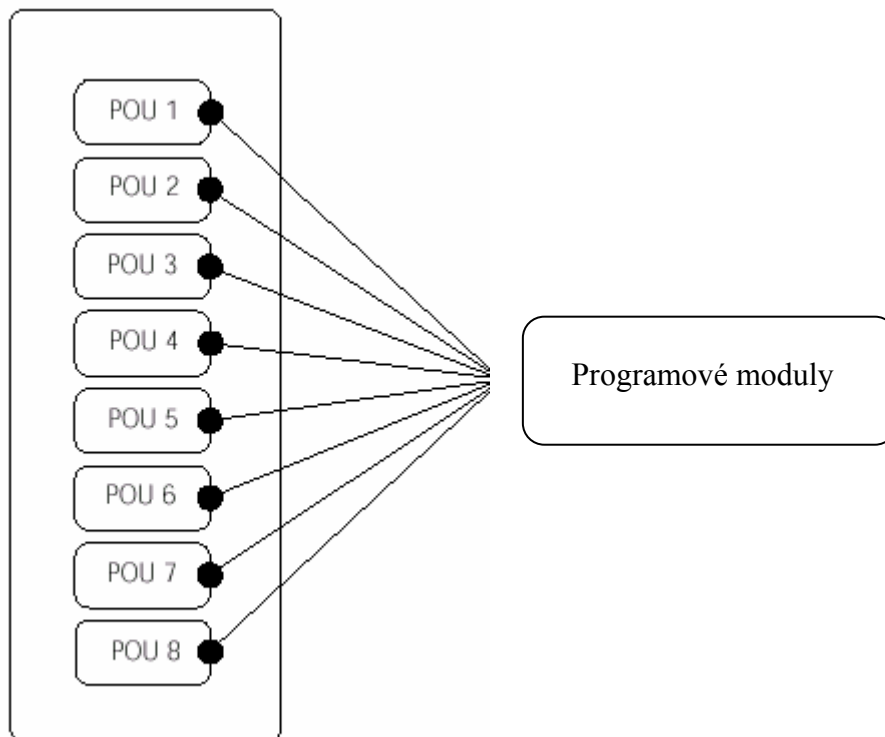
2 Standard IEC 1131.3

IEC 1131.3 je nový mezinárodní standard pro programování PLC systémů definovaný mezinárodní elektrotechnickou komisí (IEC). Tento standard definuje programovací jazyk a strukturu jednotlivých částí programu.

2.1 Stavba (struktura) programu

Standard navzájem přibližuje širokou škálu instrukcí definovaných jednotlivými výrobci PLC a umožňuje podle přesně daných pravidel jejich zápis do programových modulů.

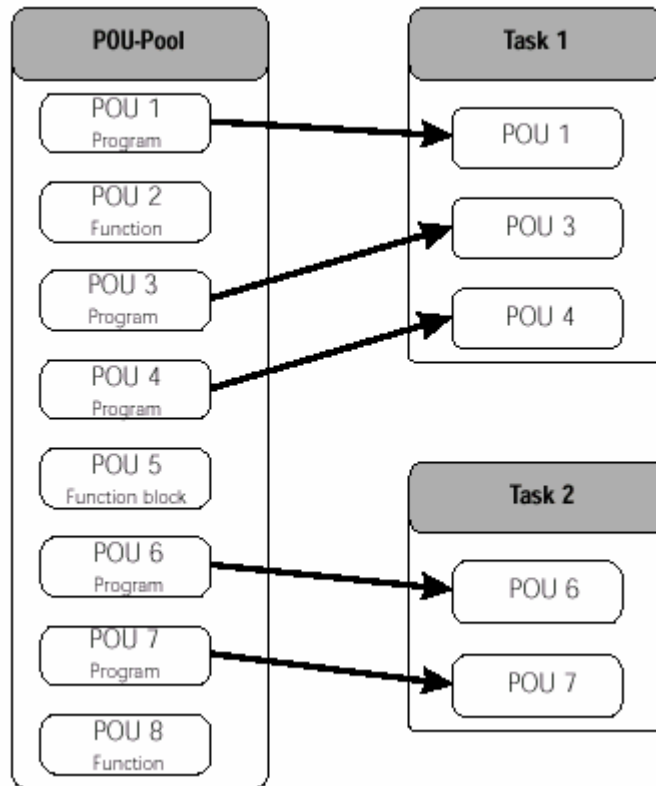
Tyto programové moduly jsou nazývány **POU** (Program Organisation Units) a tvoří základ budoucího programu.



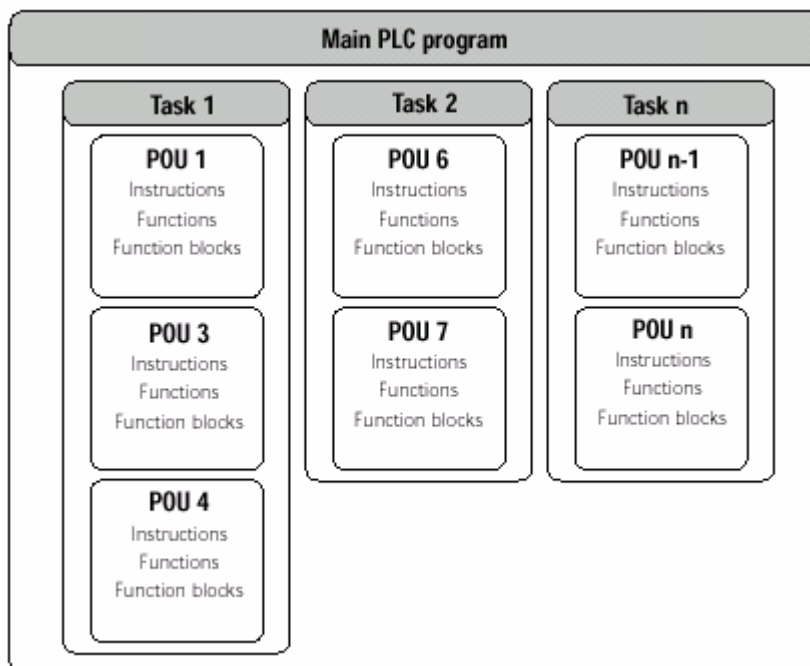
Existují celkem tři různé typy POU, rozdělené podle jejich funkce :

- **Programy**
- **Funkce**
- **Funkční bloky**

Výsledný program vznikne sloučením jednotlivých programových modulů do jedné nebo více úloh (Task).



Všechny úlohy (Tasky) jsou při kompilaci automaticky převedeny do formy skutečného PLC programu (kódu) v takzvaném Task Pool.



2.2 Programovací jazyk

Pro napsání programu je možno použít kterýkoliv z níže uvedených editorů, přičemž nelze jednoznačně stanovit výhodnost toho kterého editoru pro určitou konkrétní úlohu. Proto je výběr závislý zejména na osobě programátora, které ze zobrazení si zvolí.

- **Textové editory :**

Seznam instrukcí (Instruction List – IL)

Strukturovaný text (Structure Text – ST)

- **Grafické editory :**

Kontaktní schéma (Ladder Diagram – LD)

Diagram funkčních bloků (Function Block Diagram – FBD)

Sekvenční funkční diagram (Sequential Function Chart – SFC)

2.3 Proměnné

Použijeme-li v programu jako parametry instrukcí symbolické názvy (např. Motor_stop, Ventil_1 atd.), je nutné tyto symbolické názvy zadefinovat do seznamu proměnných.

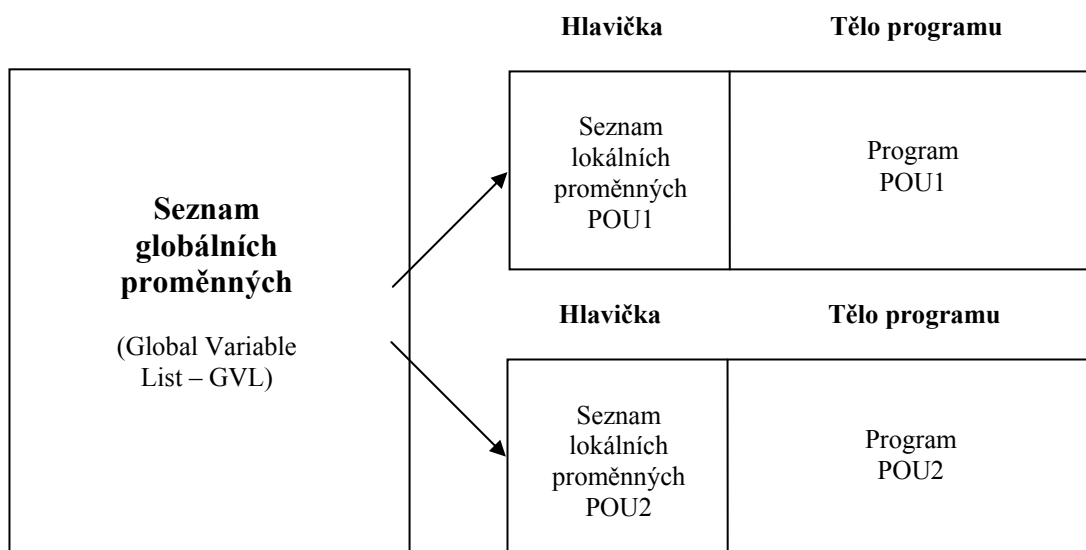
Proměnné mohou být dvojího typu :

Lokální proměnné – definované v seznamu lokální proměnných daného POU

Globální proměnné – definované v seznamu globálních proměnných (Global Variable List)

Globální proměnné je většinou přiřazena přímá adresa (např. X6, Y25, M100 atd.) a v celém programu má stejný význam. Lokální proměnná nemá přiřazenou adresu a v různých POU může mít různý význam (hodnotu).

Vyskytují-li se v programu jako parametry jednotlivých instrukcí pouze přímé adresy (např. X0, M100, Y20 atd.), není nutno žádné proměnné definovat.



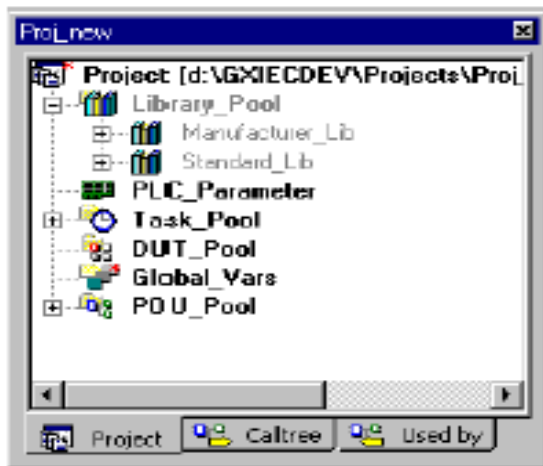
3 Základní názvosloví programovacího standardu IEC

3.1 Projekt (Project)

Každý GX IEC projekt obsahuje následující prvky :

- Skupinu knihoven (Library Pool)
- PLC parametry (PLC Parameters)
- Skupinu úloh (Task Pool)
- Skupinu strukturovaných datových typů (DUT Pool)
- Globální proměnné (Global Vars)
- Skupinu programových modulů (POU Pool)

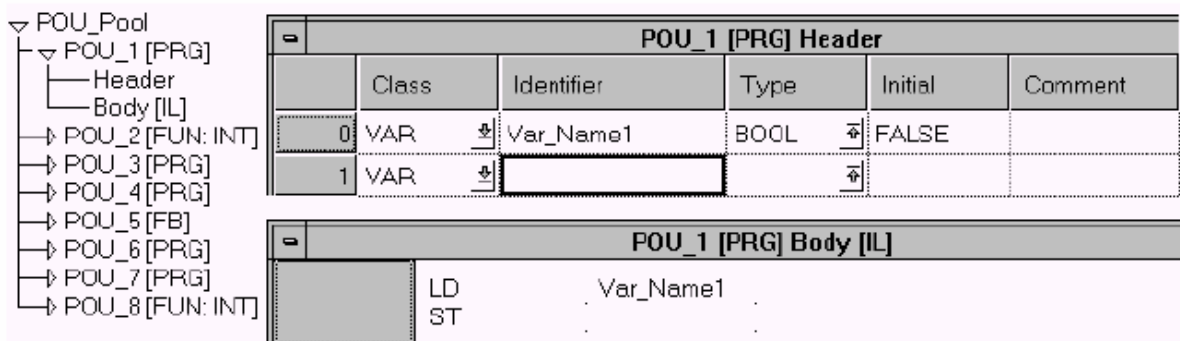
Výše uvedené skupiny prvků jsou zobrazeny v okně navigátoru (Project Navigator window).



3.2 Programový modul (Program Organisation Unit –POU)

Každý programový modul obsahuje **hlavičku** a **tělo**.

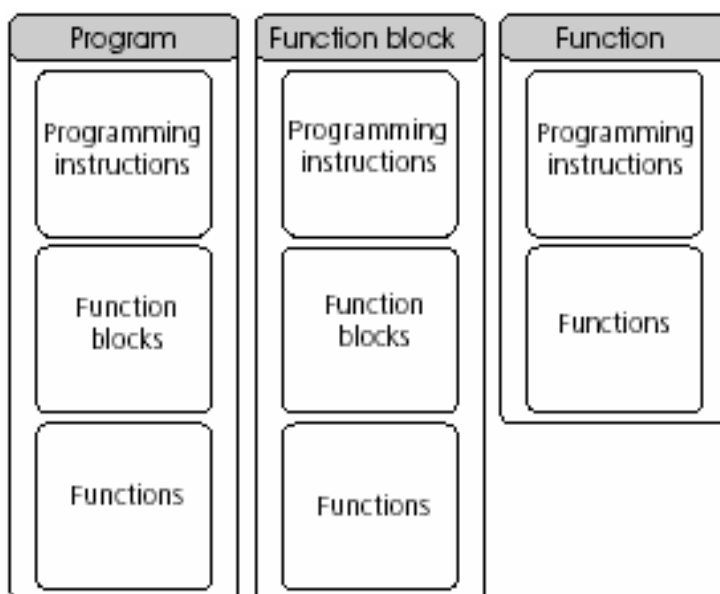
V hlavičce jsou definovány proměnné použité v daném POU, tělo obsahuje vlastní program.



Programové moduly se podle své funkce dělí do tří kategorií :

- **Programy (Programs – PRG)**
- **Funkce (Functions – FUN)**
- **Funkční bloky (Function blocks – FB)**

3.3 Programy, funkční bloky a funkce



Programové moduly (POU) tvoří základní stavební prvek konečného programu. Program může obsahovat instrukce z knihoven, funkce nebo funkční bloky. Způsob a organizace vykonávání jednotlivých POU je řízena pomocí správce úloh (Tasku).

POU definované jako funkce nebo jako funkční blok je nezávislý programový element sdružující větší skupinu samostatných instrukcí do jednoho bloku. Funkce a funkční bloky mohou být následně použity v programových modulech (Program POU) podobně jako samostatné instrukce.



Funkční bloky mohou být volány z programových modulů (Program POU) nebo z jiných existujících funkčních bloků. Nemůže být volán z funkce. Funkční blok může obsahovat instrukce z knihoven, jiné funkční bloky nebo funkce.

Výstupem z funkčního bloku může být více proměnných. Všechny vnitřní hodnoty a hodnoty výstupních proměnných funkčního bloku jsou po zpracování bloku uloženy do vnitřní paměti a při dalším vyvolání bloku jsou následně opět použity. Díky tomu je možné, že **stejný funkční blok zpracovaný dvakrát se stejnými vstupními hodnotami může mít různé výstupní hodnoty.**



Funkce mohou být volány z programových modulů (Program POU), funkčních bloků nebo z jiných existujících funkcí. Funkce může obsahovat instrukce z knihoven nebo jiné existující funkce.

Výstupem z funkce je vždy pouze jedna hodnota a funkce si neuchovává stavy vnitřních proměnných. To znamená, že **při stejných vstupních proměnných je vždy výstupem stejná hodnota**.

Porovnání funkce a funkčního bloku :

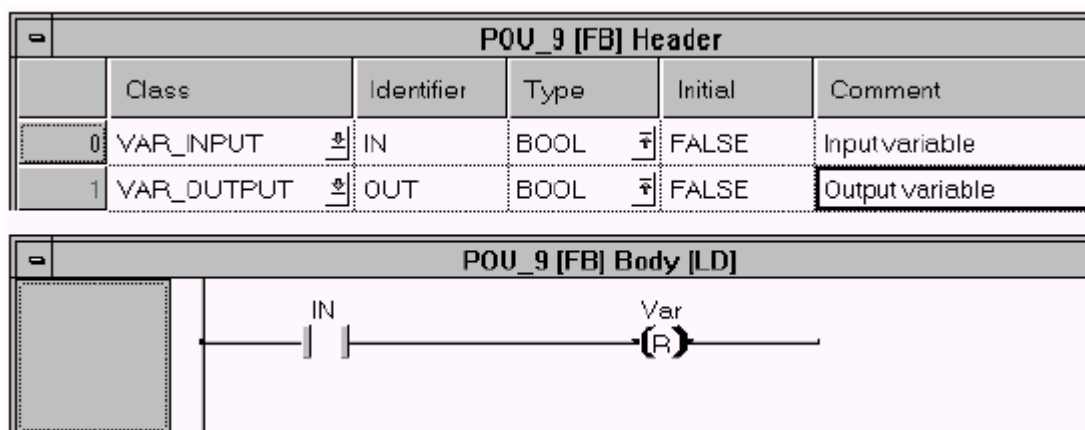
	Funkční blok	Funkce
Uložení vnitřních proměnných	Ano	Ne
Instance	Ano	Ne
Výstupy	Žádný výstup Jeden výstup Několik výstupů	Jeden výstup
Opakované volání se stejnými parametry	Mohou být různé hodnoty výstupních proměnných	Vždy stejná hodnota

3.4 Parametry (Parameters) a instance

Funkce a funkční bloky mohou mít formální nebo skutečné parametry.

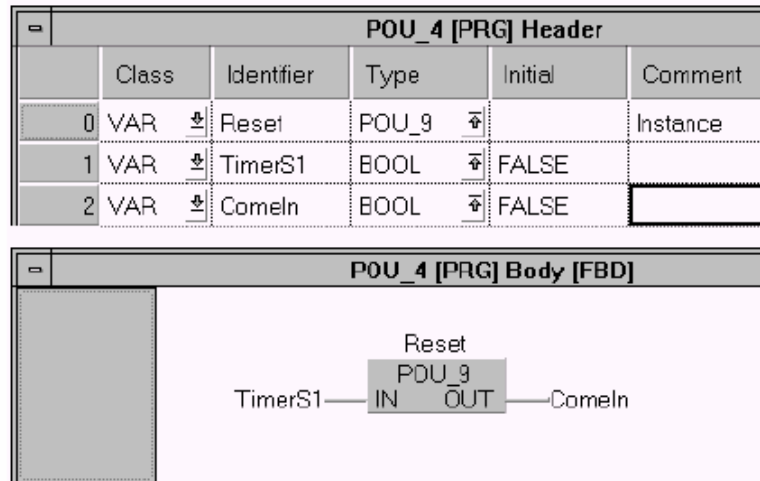
Formální parametry jsou proměnné použité při vytváření funkce nebo funkčního bloku. Formální parametry programových instrukcí ve standardních a firemních knihovnách nejsou zobrazovány.

Skutečné parametry jsou proměnné vkládané do funkce nebo do instance funkčního bloku, jsou-li použity v jiném POU. Jako skutečný parametr může být zdefinována proměnná, hardwarová adresa nebo konstanta.



POU_9 je funkční blok (FB) s proměnnými IN a OUT, definovanými (deklarovanými) v hlavičce. IN a OUT jsou formální parametry.

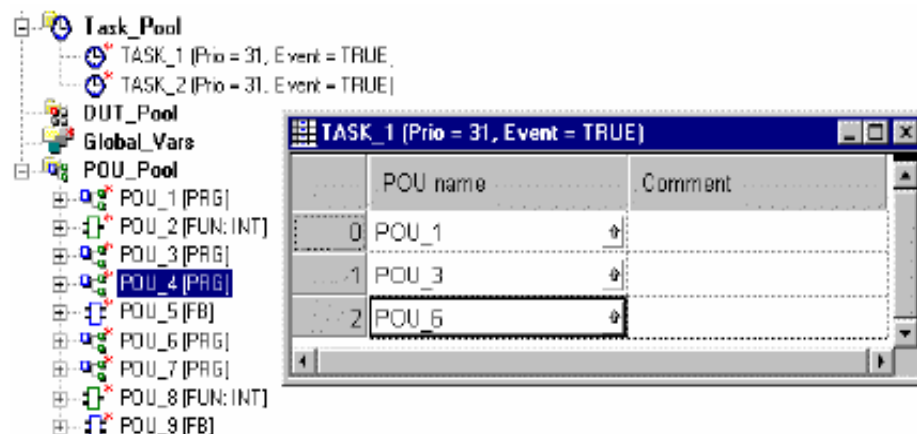
Funkční blok může být volán jen jako instance. Zadání instance je provedeno v hlavičce POU, ve kterém je daný blok použit (viz níže uvedený obrázek). V této hlavičce je funkční blok deklarován jako proměnná a výsledkem procesu instance je přiřazení jména danému bloku. Deklaraci instance lze provést vícenásobně s různými jmény pro stejný funkční blok v rámci jednoho POU. V průběhu zpracování programu v PLC jsou potom instance postupně volány a formální parametry jsou nahrazovány skutečnými parametry. Každá instance může být použita vícenásobně. Podrobnější popis aktivace instance funkčního bloku je uveden v kapitole „Programovací jazyky“.



“Reset“ je instance funkčního bloku POU_9. IN a OUT jsou formální parametry, TimerS1 a ComeIn jsou skutečné parametry instance.

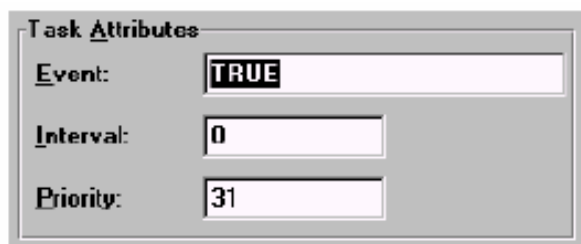
3.5 Úlohy (Tasks)

Úloha obsahuje jeden nebo více POU deklarovaných jako program (PRG). Úloha (Task) řídí zpracování jednotlivých programových modulů (POU) v řídicím systému.



Tento projekt obsahuje dvě úlohy – TASK_1 a TASK_2

Pro každou z úloh lze definovat podmínky pro její vykonání (spuštění) :



The image shows a dialog box titled "Task Attributes". It has three input fields: "Event" with the value "TRUE", "Interval" with the value "0", and "Priority" with the value "31".

- **Event** : spuštění úlohy na nástupnou hranu proměnné (default hodnota TRUE, tj. hodnot vykonává se v každém scanu)
- **Interval** : spuštění v přednastaveném čase
- **Priority** : spuštění s danou prioritou, v případě požadavku na spuštění více úloh současně

3.6 Proměnné (Variables)

Jako proměnné jsou v programu označovány prvky, obsahující hodnoty vstupů, výstupů a vnitřních proměnných v PLC. Na základě rozsahu působnosti v rámci programu se proměnné dělí do dvou kategorií :

- Globální proměnné (Global variable)
- Lokální proměnné (Local variable)

Globální proměnné jsou definovány pro celý projekt, tj. jsou dostupné ze kteréhokoliv POU a ve všech POU mají stejný význam.

Lokální proměnné je možno použít pouze v POU, v jehož hlavičce jsou zadefinovány. Nemohou být použity v jiných POU.

V různých POU mohou být použity lokální proměnné se stejným názvem. **Jejich význam (hodnota) však bude v každém POU různá !**

3.6.1 Definování proměnných (Declaring Variables)

Před zahájením psaní programu je vhodné zadefinovat všechny proměnné, které budete v projektu používat. Tento seznam proměnných lze samozřejmě vytvářet (doplňovat) i v průběhu psaní programu.

Při definování proměnné je nutno zadat tyto údaje :

- Druh (Class)
- Volbu Autoextern (Autoextern option) – pouze pro globální proměnné
- Identifikátor (Identifier)
- Absolutní adresu (Absolute address) – pouze pro globální proměnné
- Datový typ (Data type)

- Inicializační hodnota (Initial value)
- Komentář (Comment) - volitelně

POU_9 [FB] Header						
	Class	Identifier	Type	Initial	Comment	
0	VAR_INPUT	IN	BOOL	FALSE	Input variable	
1	VAR_OUTPUT	OUT	BOOL	FALSE	Output variable	

Příklad deklarace hlavičky lokálních proměnných pro funkční blok POU_9

3.6.2 Druh (Class)

Pomocí klíčového slova uvedeného v položce Druh, jsou jednotlivým proměnným přiřazeny specifické vlastnosti, definující jakým způsobem je proměnná dále použita v projektu.

Druh	Použití v POU			Význam
	PRG	FUN	FB	
VAR	X	X	X	Proměnná použitá pouze v rámci daného POU
VAR_CONSTANT	X	X	X	Lokální proměnná s neměnnou inicializační hodnotou, použitelná pouze v rámci daného POU
VAR_INPUT	-	X	X	Vstupní proměnná do POU, nemůže být v daném POU změněna
VAR_OUTPUT	-	-	X	Proměnná vystupující z POU
VAR_IN_OUT	-	-	X	Lokální proměnná vstupující do POU, v rámci daného POU může být změněna
VAR_EXTERNAL	X	-	X	Globální proměnná použitá v hlavičce POU
VAR_EXTERNAL_CONSTANT	X	-	X	Globální proměnná s neměnnou inicializační hodnotou, použitá v hlavičce POU
VAR_GLOBAL	X	-	X	Globální proměnná definovaná v seznamu globálních proměnných (GVL)
VAR_GLOBAL_CONSTANT	X	-	X	Globální proměnná s neměnnou inicializační hodnotou definovaná v GVL

3.6.3 Identifikátory (Identifiers) a absolutní adresy (Absolute Addresses)

Každá proměnná je definována svou symbolickou adresou – jménem. Tato symbolická adresa se obecně označuje jako **identifikátor** a je reprezentována řetězcem alfanumerických znaků (popř. kombinace znaků a jednoho nebo více podtržíték). Identifikátor musí vždy začínat písmenem nebo podtržítkem. Mezera nebo matematický operátor (např. +, -, * atd.) nejsou přípustné.

Příklady identifikátorů :
 Porucha_motoru
 Ventil_V32_OT
 Max_hladina

Při definování globálních proměnných je nutno uvést rovněž **absolutní adresu**, která dané proměnné přiřadí konkrétní vstup, výstup nebo vnitřní proměnnou v řídicím systému.

Při definování lokálních proměnných není nutno uvádět absolutní adresy – přiřazení jména k vnitřní proměnné se provede automaticky.

Absolutní adresy je možno zadávat pomocí MITSUBISHI syntaxe (MIT-Addr.) nebo syntaxe dle normy IEC (IEC-Addr.). Jakmile vložíte jednu z těchto adres, druhá adresa se přiřadí automaticky. Pokud vložíte adresu do nesprávného pole (např. MITSUBISHI adresu do pole IEC), software tuto chybu identifikuje a automaticky opraví - přesune adresu do správného pole.

Příklad absolutní adresace

IEC adresa	MITSUBISHI adresa	Význam
%QX0	Y0	Výstup Y0
%IX31	X1F	Vstup X1F
%MW0.450	D450	Datový registr D450

Pro zápis absolutních adres se velká písmena. Nelze používat mezery ani matematické operátory (+, -, * atd.).

3.6.4 Datové typy (Data Types)

Datový typ proměnné definuje počet bitů, způsob zpracování a číselný rozsah proměnné. V software MMP je možno používat následující datové typy :

Datový typ		Číselný rozsah	Velikost
BOOL	Bool	0 (FALSE), 1 (TRUE)	1 bit
INT	Celé číslo	-32.768 až 32.767	16 bitů
DINT	Celé číslo s dvojnásobnou délkou	-2.147.483.648 až 2.147.483.647	32 bitů
WORD	16-ti bitový řetězec	0 až 65.535	16 bitů
DWORD	32 bitový řetězec	0 až 4.294.967.295	32 bitů
REAL	Číslo s pohyblivou řádovou čárkou	+/-3.4 x 10 ³⁸ až +/- 1.175 x 10 ⁻³⁸	32 bitů
TIME	Časový údaj	-24d 0h 31m 23s 648.00ms až 24d 20h 31m 23s 647.00ms	32 bitů
STRING (pouze automaty řady Q)	Řetězec znaků	Max. 50 znaků	

3.6.5 Inicializační hodnota (Initial value)

Inicializační hodnota je automaticky nastavena softwarem a nemůže být změněna, resp. inicializační hodnotu je možno zadat (změnit) pouze pro proměnnou typu VAR_GLOBAL_CONSTANT.

3.6.6 Komentář (Comment)

Ke každé proměnné je možno připojit komentář dlouhý max. 64 znaků.

3.6.7 Volba autoextern (Autoextern option)

Globální proměnné označené ve sloupci Autoextern jsou automaticky zapsány do všech existujících POU (pouze programové a funkční bloky). Po vytvoření nového POU se rovněž dříve označené proměnné zapíší do jeho hlavičky.

4 Programovací jazyky (Programming Languages)

GX IEC Developer podporuje pět typů programovacích jazyků (editorů) – dva textové, dva grafické a jeden strukturovaný editor.

Textové jazyky:

Seznam instrukcí (Instruction List - IL; IEC IL a MELSEC IL)
Strukturovaný text (Structure Text – ST)

Grafický jazyk :

Kontaktní schéma (Ladder diagram – LD)
Diagram funkčních bloků (Function Block Diagram – FBD)

Strukturovaný jazyk :

Sekvenční funkční diagram (Sequential Function Chart – SFC)



Upozornění: Po vybrání programovacího jazyka již nelze tento jazyk změnit, resp. změnu lze provést, dojde však ke ztrátě dat (programu) v daném POU!

Pro každé POU může být použit jiný programovací jazyk

Segment (Network)

Ve všech editorech (s výjimkou SFC editoru) je PLC program v POU rozdělen do menších programových sekcí nazývaných **segmenty** (Networks). Každému segmentu může být přiřazeno jméno (network label), které může být současně použito jako cílová adresa pro instrukci skoku (jump - goto).



Každý segment by měl obsahovat pouze jeden logický obvod. Toto pravidlo nemusí být dodrženo; v jistých speciálních případech však může dojít ke zpracování instrukcí v chybném pořadí.

5 Textové editory (Text Editors)

GX IEC Developer podporuje tři typy textových editorů (jazyků) :

- **Seznam instrukcí MELSEC** (MELSEC Instruction List)
- **Seznam instrukcí IEC** (IEC Instruction List)
- **Stukturovaný text** (Structure Text)

Struktura instrukčních jazyků (editorů) je identická. Program vytvořený v některém z těchto editorů se skládá z jednotlivých instrukcí napsaných na samostatných řádcích. Každá instrukce se skládá z kódu instrukce (např. LD, AND, PLS atd.), jeho parametrů a proměnných.

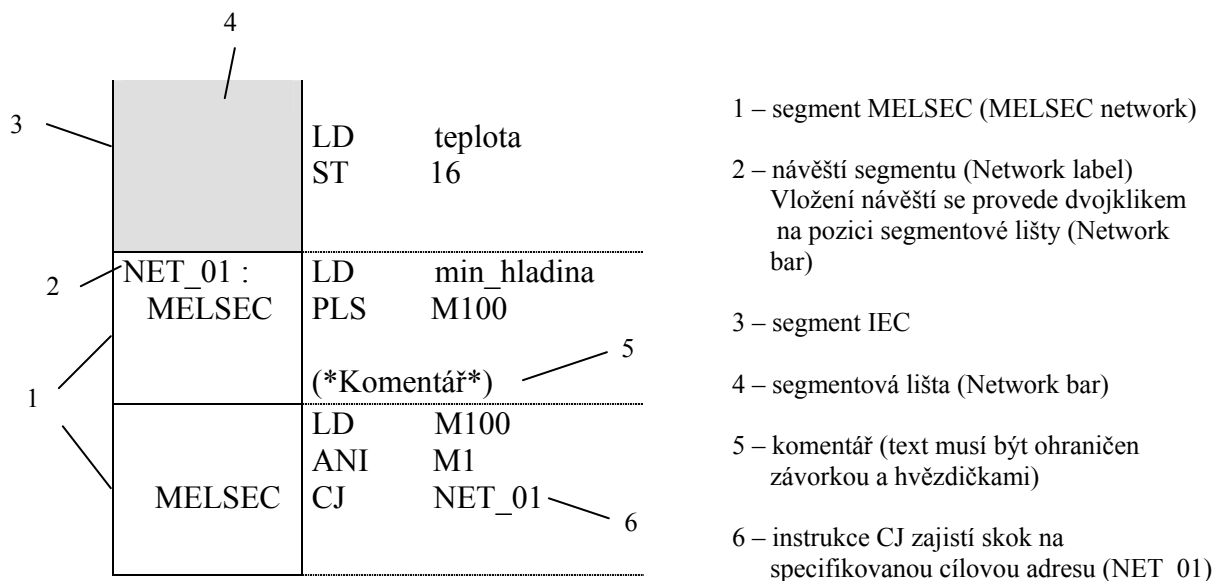
I přes formální podobnost zápisu však existují mezi oběma instrukčními jazyky podstatné rozdíly v tom, jakým způsobem jsou jednotlivé instrukce vykonávány.

5.1 Seznam instrukcí MELSEC (MELSEC IL)

Programy v MELSEC IL jsou psány (organizovány) podle pravidel normy DIN 19239 a pravidel stanovených pro softwarové produkty firmy MITSUBISHI ELECTRIC. V tomto editoru je tedy možno používat pouze MELSEC instrukce (viz. dodatek uživatelského manuálu) a MELSEC segmenty. Instrukce IEC nejsou přípustné.

5.2 Seznam instrukcí IEC (IEC IL)

IEC instrukční list povoluje kombinovat v jednom POU segmenty napsané v MELSEC instrukčním listu se segmenty napsané v instrukčním listu IEC.



5.2.1 Akumulátor

V IEC editoru se výsledek každé operace po jejím ukončení ukládá do akumulátoru. Akumulátor tedy vždy obsahuje výsledek poslední instrukce programu.



Upozornění. V tomto editoru není potřeba programovat vstupní podmínku pro vykonání operace. Vykonání je vždy závislé na obsahu bitového akumulátoru.



Příklad. V následujícím obrázku je znázorněn rozdíl mezi vykonáváním programu v MELSEC a IEC IL.

Chceme naprogramovat součet $D0(5) + D1(10) = D2(15)$ za podmínky sepnutí vstupu X0.

	LD	X0	1	
	JMPCN	Next	2	
			3	
			4	Součet v IEC instrukčním listu
Next :	LD	X3		
	ST	M10		
MELSEC	LD	X0		
	ADD	D0 D1 D2		Součet v MELSEC instrukčním listu

- 1 – Na začátku nového segmentu není definován obsah akumulátoru.
- 2 - V této chvíli akumulátor obsahuje 0 nebo 1 podle stavu vstupu X0.
- 3 – Je-li hodnota v akumulátoru 0, vykoná se instrukce JMPCN (JumpConditionalNot). Instrukce v sekvenci 4 jsou přeskočeny a program přejde do segmentu Next:. Je-li hodnota v akumulátoru 1, instrukce JMPCN je ignorována a instrukce 5,6 a 7 jsou vykonány.
- 5 - Zapiše obsah datového registru D0 (tj. 5) do akumulátoru.
- 6 – Přičte hodnotu registru D0 k hodnotě registru D1. Výsledek součtu (tj. 15) je uložen do akumulátoru.
- 7 - Uloží obsah akumulátoru do registru D2. Akumulátor bude do dalšího přepsání obsahovat poslední vypočtenou hodnotu, tj. 15.

5.2.2 Volání funkčních bloků

Funkční bloky mohou být volány pouze jako instance použitím následujících operátorů:

CAL	(Call)	nepodmíněné volání
CALC	(CallConditional)	podmíněné volání
CALCN	(CallConditionalNot)	podmíněné volání

CAL je vykonáno vždy. CALC a CALCN závisí na stavu bitového akumulátoru. CALC se vykoná, je-li hodnota akumulátoru 1, CALCN se vykoná, je-li hodnota akumulátoru 0.

Jméno instance se bloku přiřazuje v hlavičce. Skutečné parametry se bloku přiřadí v těle (body) programu, v němž je blok použit.

POU_3 [PRG] Header					
	Class	Identifier	Type	Initial	Comments
0	VAR	Reset ❶	POU_9		Instance
1	VAR	TimerS1	BOOL	FALSE	
2	VAR	ComeIn	BOOL	FALSE	

POU_3 [PRG] Body [IL]					
	LD	TimerS1			
	ST	Reset.IN			
	LD	ComeIn			
	ST	Reset.OUT			
	CAL	Reset()			❷
	LD	X0			
	CALC	Reset(IN=TimerS1,OUT:=ComeIn)			❸

Deklarace instance
,RESET' funkčního
bloku POU_9

Skutečné parametry
,TimerS1' a
,ComeIn' jsou
přiřazeny formálním
parametrům ,IN' a
,OUT'.

Ve výše uvedeném příkladu jsou uvedeny dva možné způsoby přiřazení skutečných parametrů parametrům formálním.

5.2.3 Volání funkce

Stejně jako při volání funkčního bloku i v případě volání funkce je třeba nahradit formální parametry parametry skutečnými.

Každé funkci je přiřazeno n-1 skutečných parametrů (n = celkový počet parametrů funkce), neboť první parametr musí být vždy instrukcí LD zapsán do akumulátoru.



Příklad. Použití funkce AVERAGE v IEC instrukčním listu. Funkce má 4 vstupní parametry.

Next :	LD	D0
	AVERAGE	D1,D2,D3

Funkce AVERAGE slouží k vykonání operace $(D0+D1+D2+D3) : 4$.

Po vykonání výpočtu obsahuje akumulátor výslednou hodnotu průměru výše uvedených registrů.

U některých funkcí musí být použita instrukce LD k připojení prvního parametru EN/ENO (např. E_ADD,E_MUL atd.).



Příklad : Použití funkce E_ADD v IEC instrukčním listu.

Next :	LD	X0
	E_ADD	D0,D1,D2

Funkce E_ADD slouží k vykonání operace $D0+D1 = D2$.

Vykonání následující operace je podmíněno stavem bitového akumulátoru, který obsahuje stav ENO výstupu (ENO=Enable Out). ENO má stejný stav (0 nebo 1) jako vstup EN, resp. jako signál připojený na tento vstup (X0).

5.3 Strukturovaný text (ST)

ST je textově orientovaný editor podobný jazyku PASCAL. Podporuje matematické funkce a snadné vytváření algoritmů.

Tento editor je kompatibilní s IEC 61131.3 – podporuje všechny jeho standardní funkce. Rovněž jsou podporovány všechny MELSEC instrukce.

6.1.1 Volání funkčních bloků

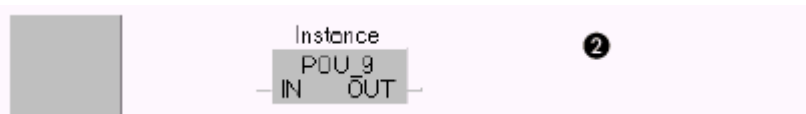
Funkční bloky mohou být volány pouze jako instance. Instance musí být deklarována v hlavičce příslušného POU.

V editoru kontaktních schémat je jméno funkčního bloku zobrazeno uvnitř šedého bloku na pracovní ploše. Jméno instance deklarováno v hlavičce musí být zapsáno přímo nad funkční blok. Skutečné parametry musí být přiřazeny příslušným formálním parametrům, jenž jsou zobrazeny uvnitř bloku.

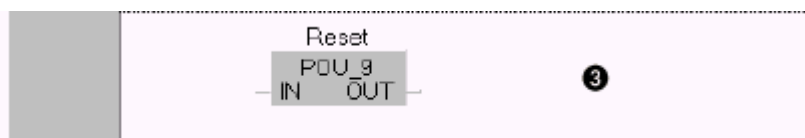
1 – Deklarace ‚RESET‘, instance funkčního bloku POU_9.

	Class	Identifier	Type	Initial	Comment
0	VAR	Reset ❶	POU_9		Instance
1	VAR	TimerS1	BOOL	FALSE	
2	VAR	ComeIn	BOOL	FALSE	

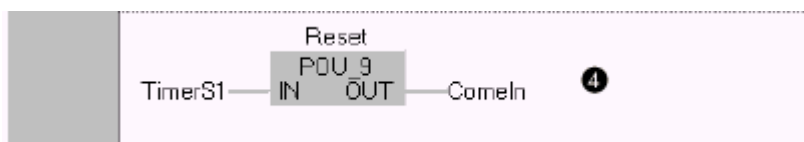
2 – Aktivace funkčního bloku POU_9. Slovo ‚Instance‘ nad šedým blokem znázorňuje požadavek vložení jména instance funkčního bloku.



3 – Vložení jména instance ‚RESET‘.



4 – Přiřazení skutečných parametrů ‚TimerS1‘ a ‚ComeIn‘ formálním parametrům ‚IN‘ a ‚OUT‘.



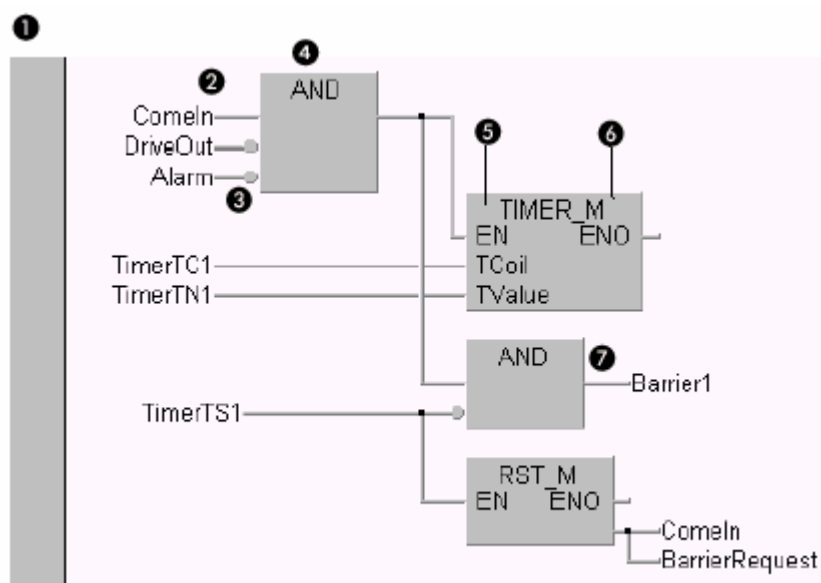
6.1.2 Volání funkce

Při volání funkce je rovněž nezbytné přiřadit skutečné parametry (vně šedého bloku znázorňujícího funkci) parametrům formálním (uvnitř šedého bloku). Viz bod 4 ve výše uvedeném příkladu.

6.2 Diagram funkčních bloků

V tomto grafickém editoru lze rovněž použít veškeré programové instrukce (viz dodatek Uživatelského manuálu). Instrukce jsou znázorněny jako šedé bloky s popisem a jsou propojovány podobně jako v editoru kontaktních schémat pomocí vertikálních a horizontálních čar – mezipropojů. Napájecí lišta (Power bar) se v tomto editoru nevyskytuje.

Některé z bloků mají kromě svých vstupních a výstupních parametrů i binární vstup EN (ENable) a binární výstup ENO (ENable Output).



- 1 – segmentová lišta (Network bar)
- 2 – vstupní proměnná – normální (Input variable - normal)
- 3 – vstupní proměnná – negovaný (Input variable - negation)
- 4 – funkce (Function)
- 5 – vstup EN (EN input)
- 6 – výstup ENO (ENO output)
- 7 – výstupní proměnná (Output variable)

6.2.1 Volání funkce a funkčního bloku

Funkce a funkční bloky jsou volány stejným způsobem jako je popsáno v kapitole editoru kontaktních schémat.

6.3 Sekvenční funkční diagram (Sequential Function Chart – SFC)

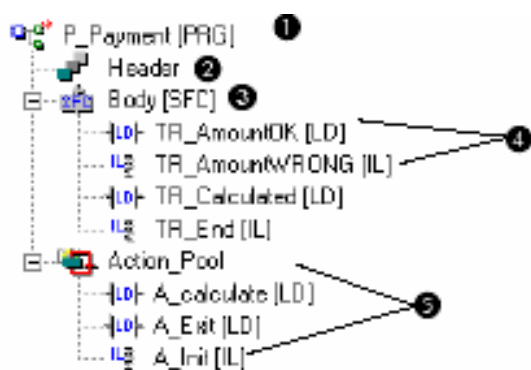
SFC je strukturovaný jazyk, který umožňuje jasné zobrazení komplexního řídicího procesu.

Základními prvky SFC jsou kroky (step) a přechody (transition).

Každému kroku může být přiřazeno 0 až n akcí. Akcí může být binární proměnná nebo PLC program. Tyto programy mohou být napsány ve kterémkoliv z editorů, včetně editoru SFC. Všechny akce jsou přehledně zobrazeny v Action_Pool v okně navigátoru projektem (Project Navigator).

Každému přechodu je přiřazena přechodová podmínka. Přechodová podmínka může být stejně jako akce napsána v jakémkoliv z editorů a je rovněž zobrazena v okně navigátoru projektem.

Přechod umožní zpracování následujícího kroku v programové sekvenci, je-li přechodová podmínka splněna (vyhodnocena jako logická jednička).



1 – ‚P_Payment‘ je programový modul (POU) deklarovaný jako program

2 – Hlavička obsahuje POU proměnné

3 – PLC program je zapsán editorem SFC

4 – Jednotlivé přechody jsou zapsány různými editory

5 – Action_Pool obsahuje jednotlivé akce, které jsou rovněž zapsány různými editory

Přiřazení akcí jednotlivým krokům a přechodových podmínek jednotlivým přechodům se provádí pomocí následujících ikon :



Aktivace akce/přechodové podmínky



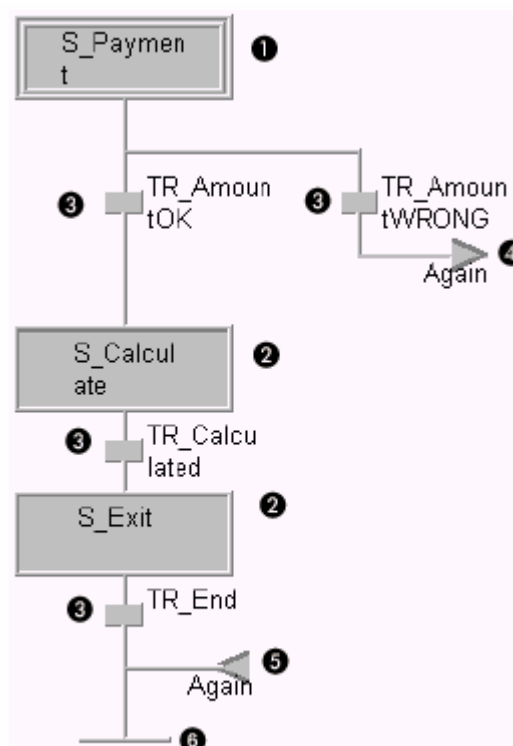
Deaktivace akce/přechodové podmínky

6.3.1 Pravidla sekvenčního programu

Sekvence vždy začíná inicializačním krokem (Initial Step). Inicializační krok nemusí být fyzicky začátkem sekvence, ale může být umístěn na kterémkoliv jiném místě. Inicializační krok je na pracovní ploše označen dvojitým rámečkem.

Kroky jsou zobrazeny jako šedé bloky s uvedením jména. Přechody jsou zobrazeny jako malé šedé obdélníky umístěné přímo na vertikálním propoju mezi jednotlivými kroky.

V jedné chvíli může být aktivován pouze jeden krok. Tato podmínka platí i v případě sekvencí se selektivním větvením. Krok je aktivován ve chvíli, kdy je předchozí krok deaktivován (ukončen) a je splněna přechodová podmínka. Je-li současně splněna podmínka pro více větví v případě selektivního větvení, je postup vykonávání dán prioritou jednotlivých sekvencí. Priorita je definována pořadím bloků směrem zleva doprava – vykoná se pouze sekvence umístěna nejvíce vlevo. To znamená, že pokud jsou současně splněny i podmínky pro další větve, sekvence napravo nebudou vykonány.



1 - Inicializační krok (Initial step)

2 - Krok (Step)

3 - Přechod (Transition)

4 - Skok

5 - Návrat instrukce skoku

6 - Závěrečný krok

V rámci jednoho sekvenčního programu je rovněž možno provádět skoky. Skoky obsahují tzv. odchozí místo (exit point) = instrukce skoku a vstupní místo (entry point) = návěští (label).

Každý krok může být deklarován jako makro. Makro kroky jsou identifikovány dvěma horizontálními čarami v bloku.



Upozornění. Více informací k sekvenčnímu programování je uvedeno v Uživatelském manuálu. Příklad vytvoření jednoduchého programu v SFC editoru je uveden v následujících kapitolách tohoto manuálu.

7 Instalace

7.1 Požadavky na hardware

Minimální hardwarová konfigurace

- Pentium II 350 procesor nebo lepší
- 32MB RAM (Microsoft Windows 95/98/Me)
- 64MB RAM (Microsoft Windows NT Workstation 4.00/ 2000 Professional)
- Hard disk s nejméně 100MB volného místa
- CD-ROM mechanika
- VGA kompatibilní grafický adaptér
- Myš
- RS232 pro komunikaci s PLC
- 17“Monitor

Pro optimální práci je vhodné pracovat s počítačem osazeným procesorem Pentium, 32MB RAM, 100MB volného místa na disku a monitorem 17“.

7.2 Požadavky na software

GX IEC Developer 4.00 je 32-bitový software podporovaný těmito operačními systémy

- Microsoft Windows 95/98/Me
(ServisPack 1)
- Microsoft Windows NT Workstation 4.00
(ServisPack 6)
- Microsoft Windows 2000 Professional
(ServisPack 1)



Důležité upozornění – AUTORSKÁ PRÁVA

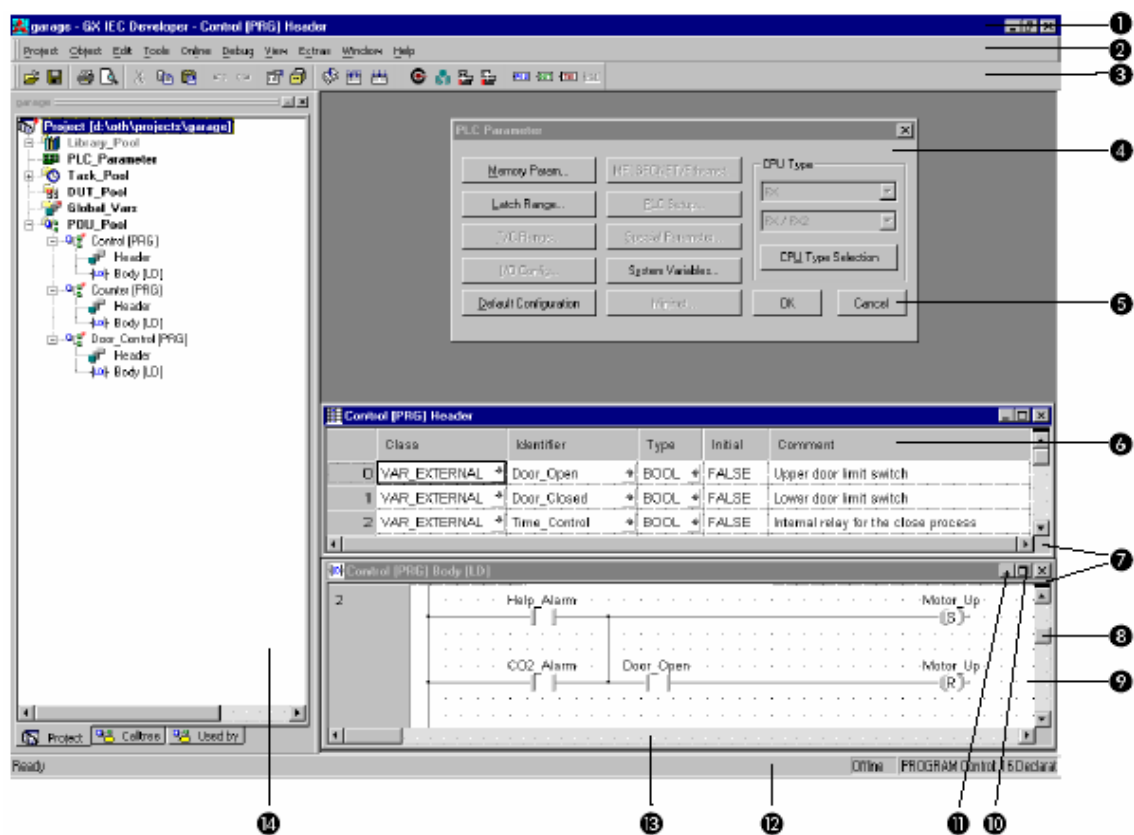
Software je chráněn autorskými právy. Otevřením balíčku se softwarem automaticky souhlasíte s podmínkami licenčního ujednání.

Oprávněnému uživateli je povoleno zhotovit jednu kopii originálního disku za účelem vytvoření záložní kopie.

8 Uživatelské rozhraní

8.1 Prvky uživatelského rozhraní

Po otevření již existujícího projektu nebo po vytvoření projektu nového se na obrazovce objeví kompletní lišta ovládacího menu (Menu bar) a navigátor projektu (Project Navigator). Níže uvedený obrázek ukazuje různé typy oken zobrazovaných v MMP - okno navigátoru projektem (Project Navigator), okno PLC parametrů (PLC parameter) a okno s hlavičkou (Header) a tělem (Body) organizačního modulu (POU). Uživatel může libovolně měnit umístění a velikost jednotlivých oken.



- 1 – Záhloví projektu (Application title bar)
- 2 – Lišta s ovládacím menu (Menu bar)
- 3 – Dialogové okno parametrů (Dialog Box)
- 4 – Nástrojová lišta (Toolbar)
- 5 – Tlačítko (Button)
- 6 – Tabulka deklarací – hlavička (Declaration table - header)
- 7 – Okno objektů (Object window)
- 8 – Vertikální posuvná lišta (Vertical scrollbar)
- 9 – Editor (Editor)
- 10 – Tlačítko pro zvětšení okna (Maximise button)
- 11 – Tlačítko pro zmenšení okna (Minimise button)
- 12 – Stavová lišta (Status bar)
- 13 – Horizontální posuvná lišta (Horizontal scrollbar)
- 14 – Okno navigátoru projektem

8.1.1 Lišta s ovládacím menu (Menu bar)

Lišta s ovládacím menu software GX IEC odpovídá obvyklým konvencím pro všechny programy Windows. Po najetí kurzorem myši na jednu z nabídek v ovládacím menu se rozvine okno s příkazy, přičemž položky, za nimiž je zobrazen symbol šipky, lze dále rozvinout.

Vybráním příkazu se objeví dialogové okno nebo okno pro zapsání údajů (data entry box). Struktura ovládacího menu a všech dostupných nabídek je obsahově sensitivní, tj. mění se v závislosti na právě prováděné činnosti v programu.

Nabídky zobrazené světle šedou barvou nejsou v dané chvíli dostupné.



Upozornění. Seznam všech příkazů je uveden v dodatku Uživatelského manuálu.

8.1.2 Nástrojová lišta (Toolbar)

Nástrojová lišta umožňuje vybrat nejčastěji používané příkazy kliknutím na odpovídající ikonu. Nástrojová lišta je stejně jako ovládací menu obsahově sensitivní, takže se zobrazované ikony mění v závislosti na prováděné činnosti.



Upozornění. Seznam všech příkazů je uveden v dodatku k Uživatelskému manuálu.

8.1.3 Okna (Windows)

Software GX IEC umožňuje současně pracovat s více objekty (např. hlavička, tělo, seznam globálních proměnných apod.) ve stejný okamžik. Pro každý objekt je otevřeno samostatné okno, přičemž uživatel může průběžně měnit polohu i velikost jednotlivých oken. Pokud objekt (okno) obsahuje více informací než je možné v dané chvíli zobrazit, okno je doplněno vertikální a horizontální posuvnou lištou.

8.1.4 Stavová lišta (Status bar)

Stavová lišta umístěná na dolním okraji obrazovky slouží k zobrazení informací o aktuálním stavu projektu. Obsah této lišty je možno uživatelsky konfigurovat (změnit) a rovněž je možné zobrazování této lišty úplně zakázat.

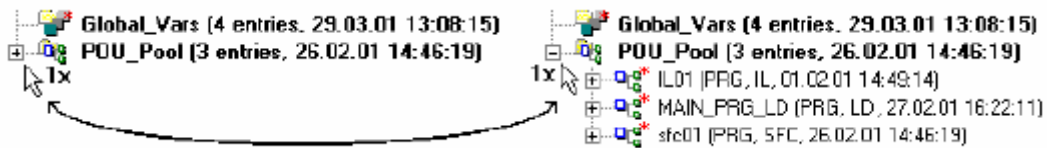
8.1.5 Navigátor projektem (Project Navigator)

Navigátor projektu je ‚řídícím centrem‘ projektu, sloužící k výběru a zpracování objektů použitých v GX IEC. Současně je výchozím místem pro všechny operace prováděné s objekty GX IEC. Okno navigátoru projektem se nezobrazí, dokud není otevřen konkrétní projekt. Uzavřením navigátoru se automaticky uzavře příslušný projekt.

Použití navigátoru

Jednotlivé objekty v navigátoru projektem je možno rozvinout kliknutím na symbol [+] a zobrazit další úroveň v dané větvi. Rozvinuté objekty lze kliknutím na symbol [-] opět sbalit. Větve, které lze rozvinout nebo sbalit, jsou ve stromové struktuře navigátoru označeny symboly [+] a [-]. Rozvinutí nebo sbalení je rovněž možné dvojklikem na symbol větvení. Dvojklikem na objekt v nejnižší úrovni (tj. objekt, jenž již není možno dále rozvinout) se otevře nové okno tohoto objektu.

Příklad práce s objekty (rozvinutí a svinutí) v navigátoru projektem :



S objekty umístěnými v POU_Pool a Task_Pool je možno provádět pouze operace Vyjmutí, Kopírování, Vložení a Mazání. Ve stejnou chvíli je možné kopírovat nebo mazat více objektů. Při výběru více objektů držte zmáčknutou klávesu CTRL a kliknutím myši (levým tlačítkem) označte požadované objekty. Při výběru skupiny objektů seřazených za sebou označte myši první objekt, stiskněte klávesu SHIFT a klikněte na poslední objekt ve skupině.



*Upozornění. Příkazem **Extended Information** v menu **View** lze povolit nebo zakázat další informace k jednotlivým objektům, zobrazované v navigátoru projektem.*

Náhledy navigátoru

Pomocí tlačítek pod oknem navigátoru si lze zvolit jeden ze tří různých náhledů navigátoru:

Project

Tato volba poskytuje celkový přehled o projektu. Obsahuje všechny prvky projektu.

Calltree

Pro tento náhled jsou odpovídajícími kořenovými položkami úlohy (Tasks) a rovněž programové moduly (POU) pokud nejsou vztaheny ke specifickým úlohám. Jako podpoložky jsou všechny použité POU. Navíc si lze v objektu navolit zobrazení použitých globálních proměnných.

Used by

Tento náhled má dvě kořenové položky - POU Pool a Global Vars. Podpoložkami POU a globálních proměnných jsou vždy POU volající respektive používající odpovídající POU nebo globální proměnnou.

Tabulka deklarácí (Declaration Table)

Lokální proměnné programového modulu (POU) jsou deklarovány v hlavičce tohoto POU, globální proměnné jsou definovány v deklarační tabulce globálních proměnných (GVL).

Global Variable List							
	Class	Autoexistem	Identifer	MIT-Addr.	IEC-Addr.	Type	Initial
0	VAR_GLOBAL	#	Door_Open	X0	%X0	BOOL	FALSE
1	VAR_GLOBAL	#	Door_Closed	X1	%X1	BOOL	FALSE
2	VAR_GLOBAL	#	Motor_Up	Y0	%Y0	BOOL	FALSE
3	VAR_GLOBAL	#	Motor_Down	Y1	%Y1	BOOL	FALSE
4	VAR_GLOBAL	#	Enter_Up	X2	%X2	BOOL	FALSE

8.1.6 Práce s tabulkou

Do každé buňky v tabulce je možno přistupovat přímo kliknutím myši. Poté co se objeví kurzor ve vybrané buňce, je možno její obsah editovat, popř. vkládat nové informace. V rámci jedné tabulky je možno pro pohyb mezi buňkami používat následující klávesy.

Šipka nahoru	posun o jeden řádek nahoru
Šipka dolů	posun o jeden řádek dolů
Šipka doprava	posun o jednu buňku vpravo
Šipka doleva	posun o jednu buňku vlevo
Tabelátor	posun o jednu buňku vpravo
SHIFT + Tabelátor	posun o jednu šipku vlevo
SHIFT + ENTER	vložení nového řádku

Nový řádek je rovněž možno vložit také pomocí příkazu Nová deklaráce (**New Declaration**) v menu **Edit**. Nový řádek je možno vložit do kteréhokoliv místa v tabulce (tj. na začátek, doprostřed nebo na konec tabulky).

Poslední možností, jak provést přidání nového řádku, je použití příslušných ikon v liště nástrojů.

8.1.7 Vymazání celé tabulky nebo vymazání jednotlivých řádků

Kliknutím levým tlačítkem myši na číslo označující řádek (tj. sloupec nejvíce vlevo) se provede výběr daného řádku (celý řádek se podbarví tmavě šedou barvou). Kliknutím na horní prázdnou buňku ve sloupci čísel řádků se označí (vyberou) všechny řádky (buňky) v tabulce. Vymazání označených buněk se provede klávesou delete.

Výběr několika řádků seřazených za sebou se provede tak, že označíme první řádek v bloku, stiskneme klávesu SHIFT a označíme kliknutím myši poslední řádek. Výběr několika různých řádků se provede pomocí stisknutím klávesy CTRL a výběrem požadovaných řádků.



Upozornění. Program provádí mazání vybraných údajů (buněk, řádků, tabulky) okamžitě po zmáčknutí klávesy DEL bez předchozí výzvy k potvrzení operace. Provedete-li neúmyslné vymazání údajů, je možné tuto operaci vrátit zpět příkazem Undo v menu Edit. Tento návrat je však možný pouze tehdy, byl-li proveden okamžitě po operaci mazání.

8.1.8 Formátování tabulky

Každý uživatel si může individuálně nastavit šířku jednotlivých sloupců tabulky. Najetím kurzoru myši na čáru oddělující dvě sousední pole (v záhlaví tabulky) se kurzor změní na dvojitou šipku. Tažením této dvojité šipky vlevo nebo vpravo se průběžně mění i šířka sloupce.

8.2 Editory

Každý PLC program je zpravidla rozdělen na několik samostatných programových modulů (POU). Každý z těchto modulů je dále rozdělen na několik menších úloh – segmentů (Networks). Každému segmentu může být přiřazeno jméno (label), které potom může být použito v programu jako cílová adresa instrukce skok (jump). Nový segment lze vytvořit pomocí příslušné ikony v liště nástrojů nebo pomocí příkazu Nový segment (**New Network**) v menu **Edit**. Nový segment lze vložit (vytvořit) ve kterémkoliv místě POU (na začátku, na konci, uprostřed).

Otevření editovacího okna (okna pro psaní programu) se provede dvojklikem myši na tělo (Body) příslušného POU v navigátoru projektem.

Textové editory :

- Seznam instrukcí IEC (IEC Instruction List)
- Seznam instrukcí MELSEC (MELSEC Instruction List)
- Strukturovaný text (Structured Text)

Grafický editor :

- Kontaktní schéma (Ladder diagram – LD)
- Diagram funkčních bloků (Function Block Diagram – FBD)
- Sekvenční funkční diagram (Sequential Function Chart – SFC)
(Pro bližší informace k SFC editoru se podívejte do Uživatelského manuálu)

Použití textových editorů :

Všechny editovací a kurzorové funkce jsou velmi podobné funkcím ve standardních textových editorech (programech). Kromě těchto standardních funkcí je možno dále využít následující doplňkové funkce :

- editace se zahájí kliknutím myši v editačním okně příslušného POU
- každý řádek programu obsahuje jednu řídicí instrukci s následující syntaxí
operátor TABSTOP operand(y)
- operátor a operand(y) musí být vždy odděleny tabelátorem (TABSTOP)
- ke každé části programu je možno zapsat komentář dlouhý jeden nebo několik řádků
- při psaní komentáře **musí** být použita následující syntaxe (* komentář *)
- mezi jednotlivými editačními sloupci a řádky se lze pohybovat pomocí kurzorových kláves
- klávesou F2 nebo kliknutím pravým tlačítkem myši se vyvolá nabídka pro danou pozici kurzoru (seznam dostupných instrukcí nebo proměnných), ze které lze snadno vybrat požadovaný operátor, resp. operand.

Použití grafických editorů :

Práce s grafickým editorem je obdobná jako práce s kterýmkoliv jiným grafickým programem v OS Windows. Nové prvky je možno do jednotlivých programových segmentů vkládat použitím grafických ikon v liště nástrojů nebo pomocí příkazů v menu Tools.

K dispozici jsou následující prvky :

- Kontakt = vstup (Contact) – pouze editor kontaktních schémat
- Cívka = výstup (Coil) – pouze editor kontaktních schémat
- Programová instrukce (Programming instruction)
- Vstupní proměnná (Input variable)
- Výstupní proměnná (Output variable)
- Instrukce návratu (Return instruction)
- Návěští skoku (Jump label)
- Komentář (Comment)

Po vložení prvků do editovacího okna (tj. okna v němž vytváříme program) je nutno tyto prvky mezi sebou spojit pomocí propojovacích čar. Pro usnadnění této činnosti je možno zapnout funkci automatického spojování, tzv. Auto Connect.

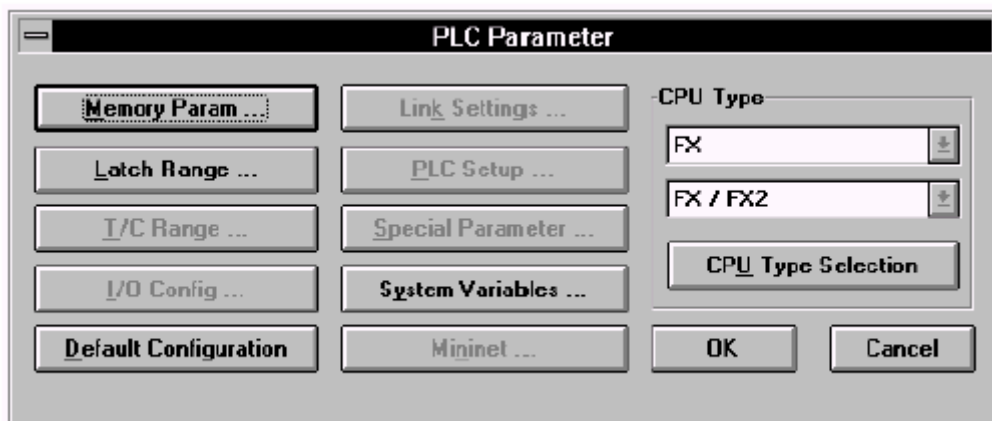
9 Postup při vytváření programu

Následující kapitoly přehledně popisují jednotlivé kroky potřebné k vytvoření PLC programu.

- Vytvoření nového projektu (Creating New Project)
- Vytvoření úlohy (Creating Task)
- Deklarace globálních proměnných (Declaring Global Variables)
- Vytvoření programového modulu – POU (Creating Program Organisation Unit)
- Vyplnění hlavičky POU (Programming POU Header)
- Naprogramování těla POU (Programming POU Body)
 - Příklady programů vytvořených v různých editorech:
 - Vstupy a výstupy v LD editoru
 - Funkce součtu v FBD editoru
 - Konfigurace I/O parametrů
 - Časovače v LD/FBD/IL editoru
 - Sekvenční funkční diagram
- Kontrola programu na syntaktické chyby (Checking PLC Programs – syntax check)
- Konfigurace úloh (Configuring Tasks)
- Kompilace projektu (Compiling Projects)
- Nastavení parametrů komunikačního portu (Communication Port Setup)
- Nahrání programu do PLC (Downloading Program to the PLC)
- Monitorování programu (Monitor Program)
- Nahrání programu z PLC (Uploading from the PLC)

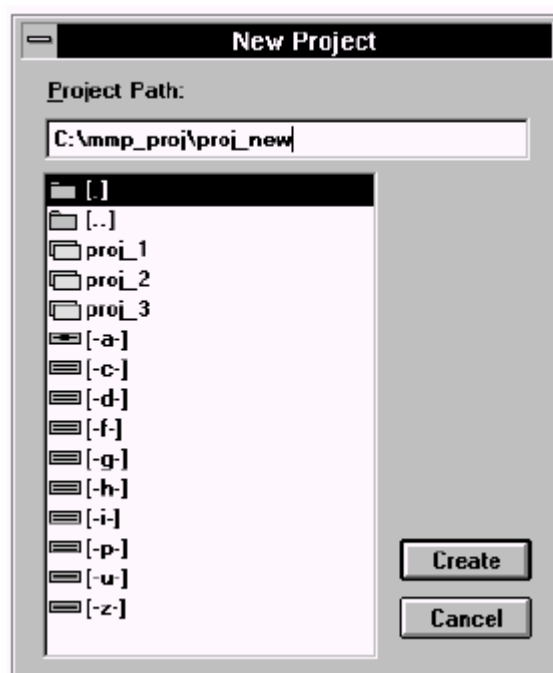
9.1 Vytvoření nového projektu (Creating New Project)

1. Spustíme program GX IEC
2. V menu **Project** vybereme příkaz **New**
3. Na obrazovce se objeví dialogové okno **PLC Parameter**, v němž vybereme vhodný typ PLC (**CPU Type**) a výběr potvrdíme klávesou **OK**.



4. Na obrazovce se objeví dialogové okno **New Project**. Vybereme cestu nebo zadáme novou cestu do kam chceme projekt uložit.
5. Do řádku s vybranou cestou přidáme jméno nového projektu a klikneme na tlačítko **Create**.

V následujícím příkladu je projekt nazvaný PROJ_NEW vytvořen v podadresáři C:\mmp_proj. Zde je třeba zdůraznit, že PROJ_NEW není pouze samostatný název souboru, ale je podadresářem obsahujícím několik souborů tvořících výsledný projekt PROJ_NEW.



Dalším krokem je výběr varianty otevření nového projektu (v automaticky otevřeném okně – GX IEC New Project Startup Options). Klikneme myší na variantu Prázdný projekt (Empty Project) a volbu potvrdíme kliknutím na tlačítko OK.

Po provedení výše uvedeného postupu se na obrazovce se objeví okno navigátoru projektem se stromově seřazenými standardními položkami projektu (Project Name – jméno projektu, Library_Pool – knihovny, PLC_Parameters – parametry PLC, Task_Pool – úlohy, DUT_Pool – strukturované datové typy, Global_Vars – globální proměnné a POU_Pool – programové moduly).

V závorce u každé položky jsou zobrazeny další doplňující informace. Zobrazení nebo zakázání těchto informací je možno provést příkazem **Extended Information** v nabídce **View** (je-li zobrazení informací povoleno, je vedle příkazu Extended Information symbol zatržítka).

Standardní barva pozadí okna GX IEC je bílá. Tuto barvu je možno uživatelsky nastavit příkazem **Colors** v menu **View**.

9.2 Definování nové úlohy (New Task)

1. Vybereme okno navigátoru projektem.
2. Vybereme příkaz New v menu Object a dále vybereme nabídku Task, nebo klikneme na ikonu New Task v liště nástrojů :



Poté se zobrazí dialogové okno nové úlohy :



3. Vložíme jméno úlohy (maximálně 16 znaků) a potvrdíme OK. GX IEC vytvoří novou úlohu a zařadí ji do seznamu úloh (Task Pool) v okně projekt navigátoru.



Upozornění. Přiřazení programových modulů (POU) do Tasků a definice atributů (Tasků) budou vysvětleny v dalších kapitolách tohoto manuálu.

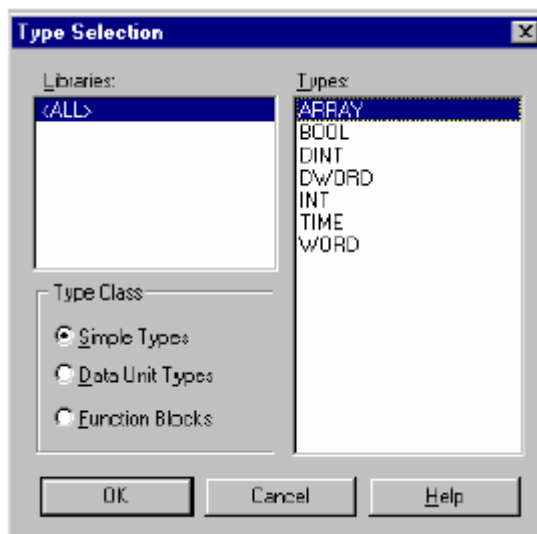
9.3 Definování (deklarace) globálních proměnných (global variable)

1. Dvojklikem na položku Global_Vars v navigátoru projektem se v pravé části obrazovky otevře okno (Global Variable List windows) obsahující tabulku pro zadání údajů.

Global Variable List								
	Class	Autoextern	Identifier	MIT-Addr.	IEC-Addr.	Type	Initial	Comment
	VAR_GLOBAL							

2. Jako přednastavená (default) hodnota v prvním sloupci je VAR_GLOBAL. Pokud chceme změnit typ proměnné, klikneme myší na šipku umístěnou napravo a vybereme jiný typ (VAR_GLOBAL_CONSTANT).
3. Klávesou TAB (tabelátor) se přesuneme do sloupce Identifier (identifikátor), kde zadáme jméno první proměnné.
4. Dalším zmáčknutím klávesy TAB se přesuneme do sloupce MIT-Addr. nebo IEC-Addr., kde zadáme absolutní adresu globální proměnné.
5. Zmáčknutím klávesy TAB se přesuneme do sloupce Type. V okamžiku zapsání absolutní adresy (viz bod 4) se do sloupce Type automaticky zapíše odpovídající typ

proměnné. Pokud chceme tento typ změnit, klikneme na šipku na pravé straně políčka, čímž se zobrazí nové dialogové okno :



Poté vybereme Simple Types z pole Type Class a z pole na pravé straně (types) odpovídající typ proměnné.

6. Inicializační hodnotu ve sloupci Initial je možno zadat pouze v případě, že je navolen typ proměnné VAR_GLOBAL_CONSTANT.
7. Do pole ve sloupci komentář (Comment) je možno zapsat ke každé proměnné samostatný komentář.

Vložení další proměnné :

Je-li editační kurzor v posledním komentářovém poli (barva pole je bílá, kurzor bliká), zmáčknutím tabulátoru dojde ke vložení nového řádku pro deklaraci proměnné



Nový deklarační řádek je možno také vložit tak, že šipkami nebo myší najedeme na poslední existující řádek a zmáčkneme kombinaci kláves SHIFT a ENTER nebo pomocí příkazu **New Declaration** v menu **Edit** a zadáním místa umístění nového řádku výběrem z podmenu (**Top** – na první řádek, **Before** – před aktuální řádek, **After** – za aktuální řádek, **Bottom** – na poslední řádek).

Pro vložení nového řádku je možno rovněž použít příslušné ikony v liště nástrojů.

	Class	Autoextem	Identifer	MIT-Addr.	IEC-Addr.	Type	Initial	Remark
	VAR_GLOBAL							

Poslední možnost, jak zadefinovat novou proměnnou, resp. jak vytvořit nový řádek, je zkopírováním některého ze stávajících řádků přes klipboard (najedeme na řádek který chceme zkopírovat, zmáčkne kombinaci kláves CTRL a C, najedeme na novou pozici a zmáčkne CTRL a V).

8. Po vložení všech potřebných proměnných uložíme provedené změny příkazem **Save** v menu **Object**. Pokud toto uložení neprovedeme, budeme k tomu automaticky vyzváni při pokusu o uzavření tohoto okna.

9.4 Vytvoření programového modulu (POU)

Každý programový modul se skládá ze dvou částí – hlavičky a těla.

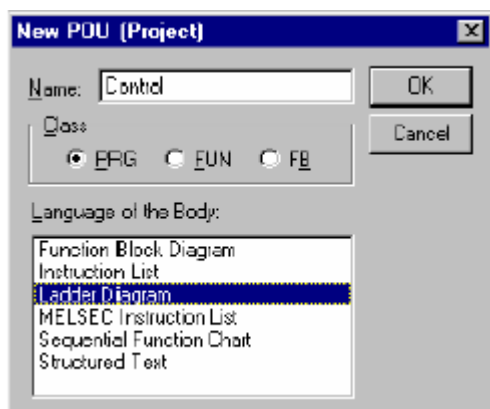
Nový programový modul vytvoříme kliknutím na ikonu nového POU v liště nástrojů :



Pozn. Tato ikona je zobrazena pouze tehdy, pokud je otevřeno okno navigátoru projektem.

Po zobrazení nového dialogového okna specifikujeme, zda nový modul bude vytvořen jako program (PRG), funkce (FUN) a nebo funkční blok (FB). Poté vybereme programovací jazyk (editor) a volbu potvrdíme kliknutím na tlačítko OK. Tím dojde k vytvoření nového POU a jeho vložení do navigátoru projektem.

Na níže uvedeném obrázku je příklad vytvoření nového POU s názvem Control, deklarovaného jako program (PRG) a sestavovaného v editoru kontaktních schémat.



Nové POU vložené do složky POU v navigátoru :



Symbol [+] umístěný před názvem „Control“ signalizuje možnost dalšího rozvinutí dané nabídky, hvězdička signalizuje, že projekt ještě nebyl kompilován.

Dvojklikem na název POU (Control) se POU dále rozvine a zobrazí se hlavička (Header) a tělo (Body) :



9.4.1 Vytvoření hlavičky programového modulu (POU)

Hlavička programového modulu slouží k deklarování proměnných použitých v rámci daného POU. Kromě globálních (externích) a lokálních proměnných, může tato hlavička také obsahovat instance funkčního bloku.

Postup při vytvoření nové hlavičky :

1. Dvojklikem na položku hlavička (Header) příslušného POU v navigátoru projektem otevřeme nové okno obsahující deklarační tabulku pro zapsání lokálních proměnných.
2. Deklarace jednotlivých proměnných se děje stejně jako deklarace globálních proměnných (viz minulé kapitoly) – vložením typu proměnné (class), identifikátoru a datového typu.

Control [PRG] Header					
	Class	Identifer	Type	Initial	Comment
<input type="checkbox"/>	VAR				

3. Po vložení všech proměnných lze data uložit příkazem **Save** v menu **Object**.



*Upozornění. Kopírování proměnných z tabulky globálních proměnných je možno provést kombinací kláves CTRL + C a vložení CTRL + V nebo automaticky zatržením políčka **External**.*

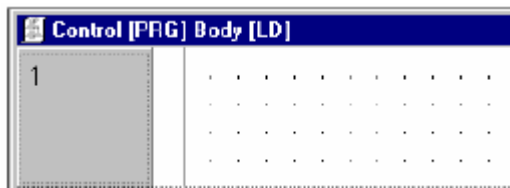
Položky „typ proměnné, identifikátor a datový typ“ jsou vysvětleny v předchozích kapitolách.

9.4.2 Programování těla (Body) programového modulu (POU)

Tělo obsahuje vlastní program (kód) pro PLC. Typ programového jazyka (editoru) je uveden v závorce za názvem projektu v navigátoru projektem.

Postup při programování těla POU :

1. Dvojklikem na položku tělo (Body) příslušného POU v navigátoru projektem otevřeme nové okno pro psaní programu ve vybraném programovacím jazyku (editoru). Okno obsahuje jeden segment (Network).



Smazání rastru zobrazeného na pozadí okna je možno provést zrušením zatržítka u příkazu **Grid** v menu **View**

Velikost rastru (a tím i velikost zobrazení) je možno nastavit příkazem **Environment** v menu **View**. Přitom je však třeba vzít v úvahu, že toto nastavení se netýká pouze daného okna, ale celého otevřeného projektu.

2. Nyní je možno začít psát vlastní program. Příklady napsané v jednotlivých jazycích (editorech) jsou uvedeny na následujících stránkách.
3. Po napsání programu je třeba jej uložit příkazem Save v menu Object.



Upozornění. Velikost editovacího prostoru je možno zvětšit pomocí myši. Najedeme ukazovátkem myši (šipkou) na nejnižší hranu segmentové lišty (levá část okna), čímž dojde ke změně symbolu ukazovátka myši ze šipky na dvojitou šipku. Držením levého tlačítka myši a současným tažením se bude měnit velikost editačního okna. Po dosažení požadované velikosti uvolníme tlačítko myši.

10 Příklady programování

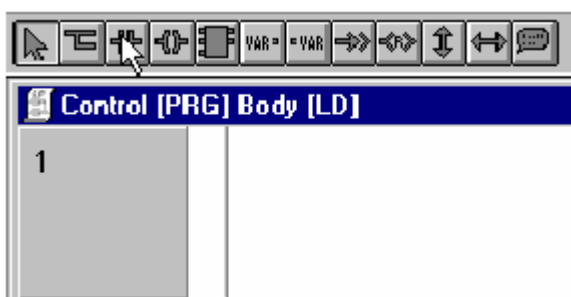
10.1 Vstupy a výstupy v editoru kontakt. schémat (Ladder diagram – LD)



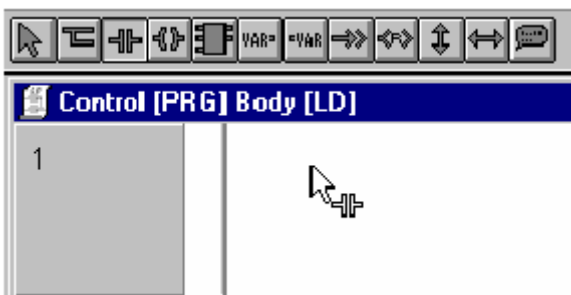
Programování vstupů a výstupů

V okně navigátoru projektem dvojklikněte na tělo (body) příslušného POU definovaného pro editor kontaktních schémat (LD).

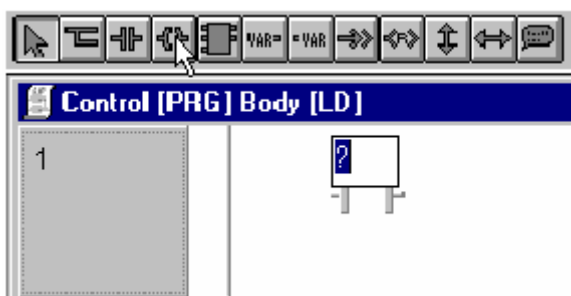
Klikněte na ikonu kontakt (Contact) v liště nástrojů.



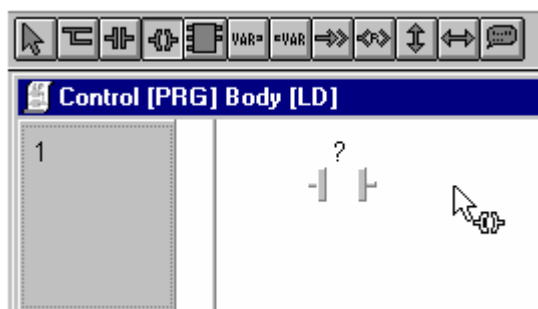
Přesuňte ukazovátko myši na požadovanou pozici v pracovní ploše a stiskněte levé tlačítko myši pro vložení kontaktu.



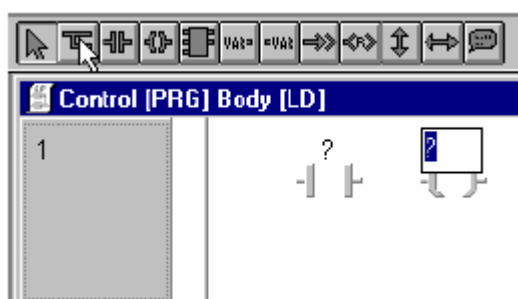
Klikněte na ikonu cívky (Coil) v liště nástrojů.



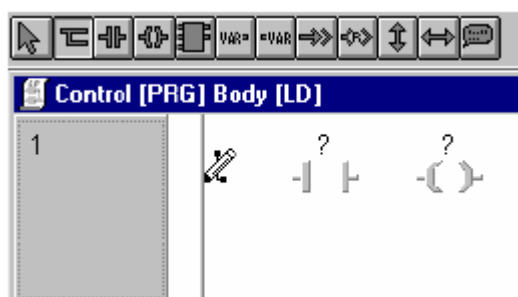
Přesuňte ukazovátka myši na požadovanou pozici v pracovní ploše a stiskněte levé tlačítko myši pro vložení cívky.



Klikněte na ikonu propojování/čára (**Interconnect/Line**) v liště nástrojů nebo klikněte pravým tlačítkem myši a vyberte položku **Line**.

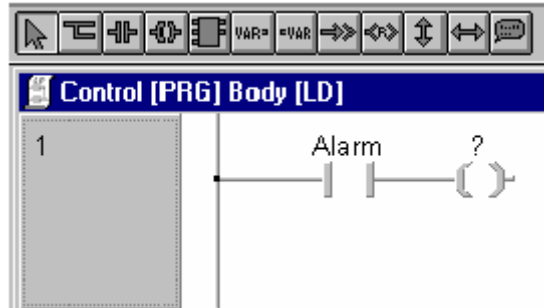


Umístěte ukazovátka myši na svislou přípojnicovou lištu na levé straně okna, stiskněte levé tlačítko myši a přesuňte myš na pozici výstupního kontaktu (cívky). Při této činnosti musí být vypnutá volba automatického propojování - Auto Connect (druhá položka v okně, které se objeví při kliknutí pravým tlačítkem myši na pracovní ploše).



Po vykonání výše uvedeného postupu se na obrazovce objeví níže uvedený obvod. Symboly „?“ umístěné nad vstupním a výstupním kontaktem jsou pouze pracovní systémové symboly, které musí být nahrazeny deklarovanými názvy proměnných nebo absolutními adresami (např. X0, M100, D15 atd.).

Klikněte na ikonu režimu výběru (symbol šipky – viz níže uvedený obrázek) v liště nástrojů, najedte ukazovátkem myši na „?“ umístěný nad vstupním kontaktem a klikněte levým tlačítkem myši. Do políčka, které se tímto otevře, napište název proměnné nebo zmáčkněte klávesu F2 nebo pravé tlačítko myši, čímž se otevře nové okno se seznamem proměnných zadefinovaných v databázi globálních nebo lokálních proměnných. Z tohoto seznamu lze vybrat libovolnou proměnnou a pomocí tlačítka Apply jí vložit do programu.



Upozornění. Pro popisovaný postup nebyla použita volba Auto Connect.

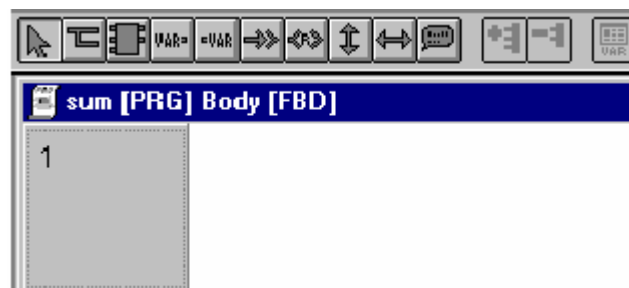
10.2 Příklady naprogramování funkce součet (SUM) v editoru diagramu funkčních bloků (FBD)



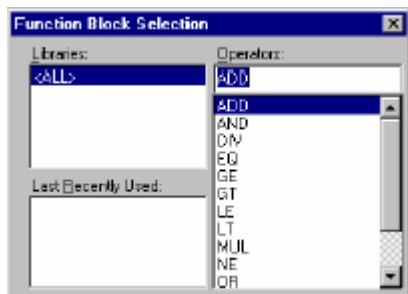
Níže uvedený postup je naprosto stejný pro editor kontaktních schémat (LD) i editor diagramu funkčních bloků (FBD). Oba způsoby programování se v tomto konkrétním případě liší pouze ikonami zobrazovanými v liště nástrojů.

V okně navigátoru projektem dvojklikněte na tělo (body) příslušného POU definovaného pro editor diagramu funkčních bloků (FBD).

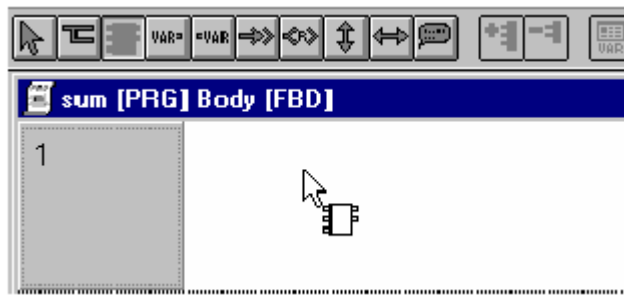
Klikněte na ikonu funkčního bloku umístěnou v liště nástrojů.



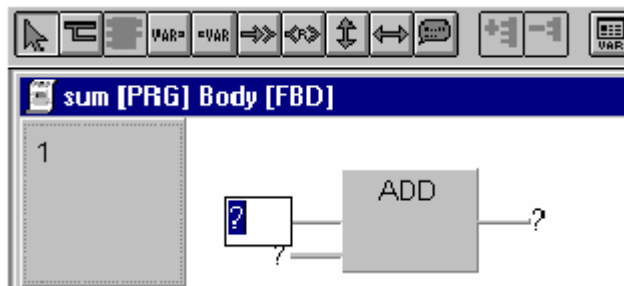
Dvojklikněte na funkci ADD v dialogovém okně výběru funkce (Function Block Selection)



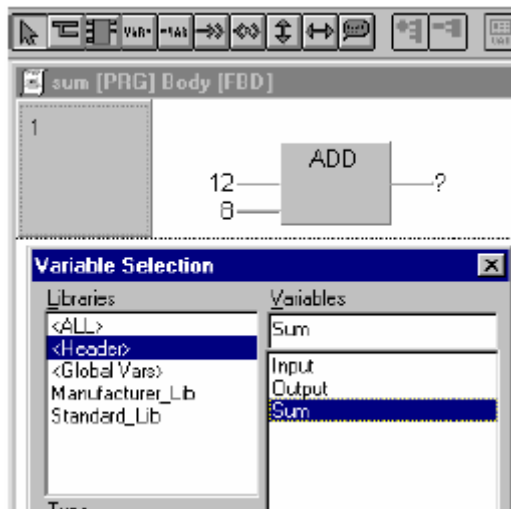
Přesuňte ukazovátko myši na požadovanou pozici v pracovní ploše a stiskněte levé tlačítko myši pro vložení funkčního bloku.



Klikněte na text „?“ první vstupní proměnné a přepište jej hodnotou 12 a druhou proměnnou hodnotou 8.



Klikněte na text „?“ výstupní proměnné a zmáčkněte klávesu F2. Pokud je symbolická proměnná již zadefinována v seznamu lokálních nebo globálních proměnných, je možno jí vybrat a vložit do programu.



Pokud zadefinována není, je možno v této chvíli provést její deklaraci do databáze proměnných.

Dialogové okno Variable Selection se rovněž otevře, pokud do programu napíšeme symbolický název (v našem případě SUM), který není dosud deklarován. V nově otevřeném okně máme možnost okamžitě provést deklaraci, nebo jej zavřít a deklaraci provést dodatečně.

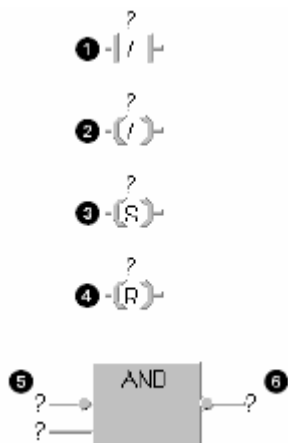
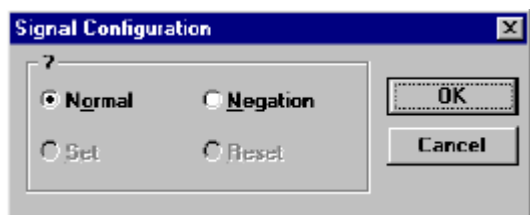


Upozornění. Proměnné se objeví v seznamu operandů až po uložení hlavičky, ve které byly deklarovány.

10.3 Konfigurace parametrů vstupních a výstupních signálů

Dvojklikem na vstupní kontakt, výstupní kontakt nebo spojovací bod proměnné ve funkčním bloku dojde k otevření dialogového okna umožňující konfiguraci signálů.

Po nastavení potřebných parametrů potvrdíme volbu tlačítkem OK.



1. Negovaný vstupní kontakt (pouze editor LD)
2. Negovaný výstup (cívka) (pouze editor LD)
3. Výstup (cívka) je setován (pouze editor LD)
4. Výstup kontakt (cívka) je resetován (pouze editor LD)
5. Negovaná vstupní proměnná (editor FBD nebo LD)
6. Negovaná výstupní proměnná (editor FBD nebo LD)

10.4 Časovače v editorech LD, FBD a IL

10.4.1 Popis proměnných časovače

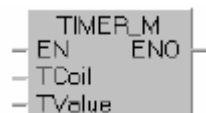
Všechny časovače musí obsahovat čtyři následující prvky :

TValue	přednastavená hodnota (Setpoint value)
TN	skutečná hodnota (Actual value)
TC	výstupní cívka (Output coil)
TS	vstupní kontakt (Input contact) – stav časovače (On/Off)

TN, TC a TS musí být deklarovány v databázi globálních proměnných (GVL).

Global Variable List						
	Class	Autoextem	Identifier	MIT-Addr	IEC-Addr.	Type
0	VAR_GLOBAL		TIMER1C	TC0	%M05.0	BOOL
1	VAR_GLOBAL		TIMER1S	TS0	%M03.0	BOOL
2	VAR_GLOBAL		TIMER1N	TN0	%MW3.0	INT

Hodnota TValue je přiřazena funkci TIMER přímo.



11 Příklad časovače



Následující příklad popisuje postup jak naprogramovat časovač a instanci funkčního bloku v editoru LD, FBD a IL.

Zadání. Je-li sepnut vstup „Input 1“, 100ms časovač „Timer 1“ začne čítat a čítá do doby, než dosáhne hodnoty 100 (tj. uběhne 100ms = 10s). Dále chceme, aby výstup „Output 1“ byl setován po nasetování vstupu „Input 2“ a byl resetován po doběhnutí časovače „Timer 1“ (tj. po uběhnutí 10ms).

Řešení. Nastavení výstupu 1 (Output 1) je závislé na stavu vstupu 2 (Input 2) a časovače 1 (Timer 1). Tuto závislost vyřešíme funkčním blokem SET_RST. Vstup 1 (Input 1) aktivuje časovač, tj. řídí spuštění časovače (aktivaci cívky časovače – Timer1C). Přednastavená hodnota časovače (Tvalue) je 100. Funkce časovače bude realizována standardní funkcí TIMER_M z knihovny .

11.1 Vytvoření programu

11.1.1 Naprogramování funkčního bloku SET_RST

Vytvoříme nový projekt a vytvoříme POU nazvané SET_RST, zdefinované jako funkční blok v kontaktním zobrazení (kliknutím na ikonu POU v liště nástrojů, zadáním jména SET_RST, navolením FB v poli Class a vybráním LD editoru).

Do hlavičky vložíme následující tři proměnné SET, RST a Q (dvojklikem na POU SET_RST v projekt navigátoru se toto POU rozbalí, dvojklikem na položku Header se otevře okno hlavičky).

SET a RST jsou vstupní proměnné (VAR_INPUT), Q je výstupní proměnná (VAR_OUTPUT).

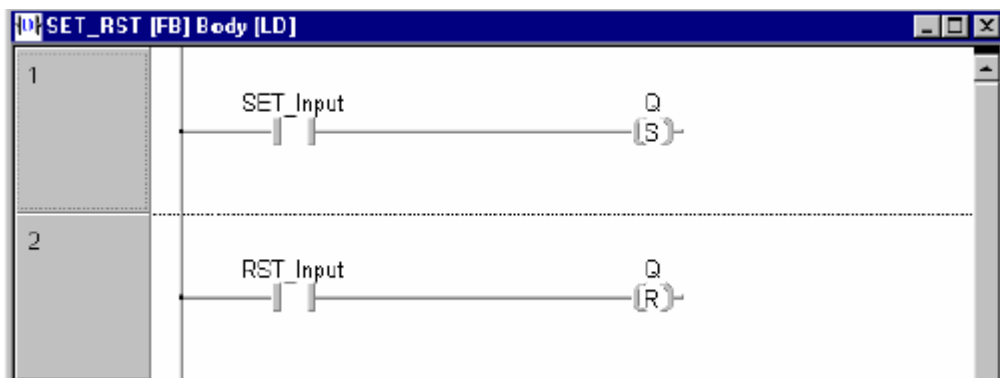
Uložíme hlavičku (Save).

	Class	Identifier	Type
0	VAR_INPUT	SET_Input	BOOL
1	VAR_INPUT	RST_Input	BOOL
2	VAR_OUTPUT	Q	BOOL

V těle (Body) vytvoříme dva segmenty (Network) a zapíšeme do nich níže uvedené obvody.

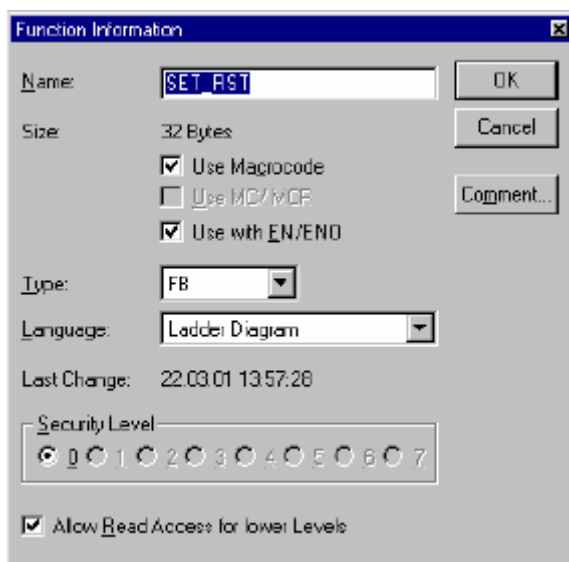
Výstup Q je setován, je-li vstup SET aktivní, a resetován, je-li aktivní vstup RST. Jsou-li aktivní oba signály současně, je výstup Q resetován, neboť funkce R (reset) je umístěna za funkcí S (set), tj. je umístěna blíže ke konci programu.

Přiřazení funkcí set (S) a reset (R) výstupním cívkám uděláme pomocí dvojkliknutí na „normálních“ cívkách vložených do programu.



Příkazem **Save** v menu **Object** vytvořené POU na disk počítače.

V navigátoru projektem najedeme na POU SET_RST a kombinací kláves ALT + ENTER (nebo příkazem **Information** v menu **Object**) otevřeme níže uvedené okno.



Aktivujeme (zatrhneme) volbu EN/ENO pro přidání vstupu Enable input a výstupu Enable Output k našemu bloku a dále volbu Use Macrocode, abychom optimalizovali kód.

11.1.2 Definování globálních proměnných

Jednotlivé prvky časovače (TC, TS a TN) musí být deklarovány v databázi globálních proměnných (Global Variable List). V níže uvedeném příkladu jsou do databáze globálních proměnných rovněž zdefinovány vstupní a výstupní proměnné.

Otevřeme tabulku globálních proměnných (dvojklikem na položku Global_Vars) a zadefinujeme proměnné. Hlavičku uložíme (Save).

Global Variable List							
	Class	Autoextern	Identifier	MIT-Addr	IEC-Addr.	Type	Initial
0	VAR_GLOBAL	X	TIMER1C	T00	%MX5.0	BOOL	FALSE
1	VAR_GLOBAL	X	TIMER1S	T00	%MX3.0	BOOL	FALSE
2	VAR_GLOBAL	X	TIMER1N	T00	%MW3.0	INT	0
3	VAR_GLOBAL	X	Input1	X0	%IX0	BOOL	FALSE
4	VAR_GLOBAL	X	Input2	X1	%IX1	BOOL	FALSE
5	VAR_GLOBAL	X	Output1	Y0	%QX0	BOOL	FALSE

11.1.3 Vytvoření nového POU Timer

Vytvoříme nové POU nazvané DEMO_LD, definované jako program (PRG) v kontaktním schématu (LD).

Otevřeme hlavičku a doplníme níže uvedené proměnné. Tuto činnost je možné zrychlit překopírováním proměnných z databáze globálních proměnných.

DEMO_LD [PRG] Header					
	Class	Identifier	Type	Initial	
0	VAR_EXTERNAL	TIMER1C	BOOL	FALSE	
1	VAR_EXTERNAL	TIMER1S	BOOL	FALSE	
2	VAR_EXTERNAL	TIMER1N	INT	0	
3	VAR_EXTERNAL	Input1	BOOL	FALSE	
4	VAR_EXTERNAL	Input2	BOOL	FALSE	
5	VAR_EXTERNAL	Output1	BOOL	FALSE	
6	VAR	DATA	INT	0	
7	VAR	SET_RST1	SET_RST		

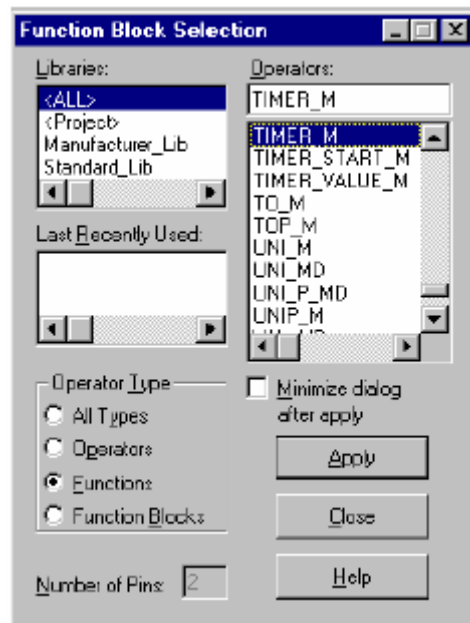
VAR_EXTERNAL použité globální proměnné
 VAR použité lokální proměnné
 SET_RST1 instance funkčního bloku SET_RST

Příkazem **Save** v menu **Object** uložíme hlavičku na disk počítače.

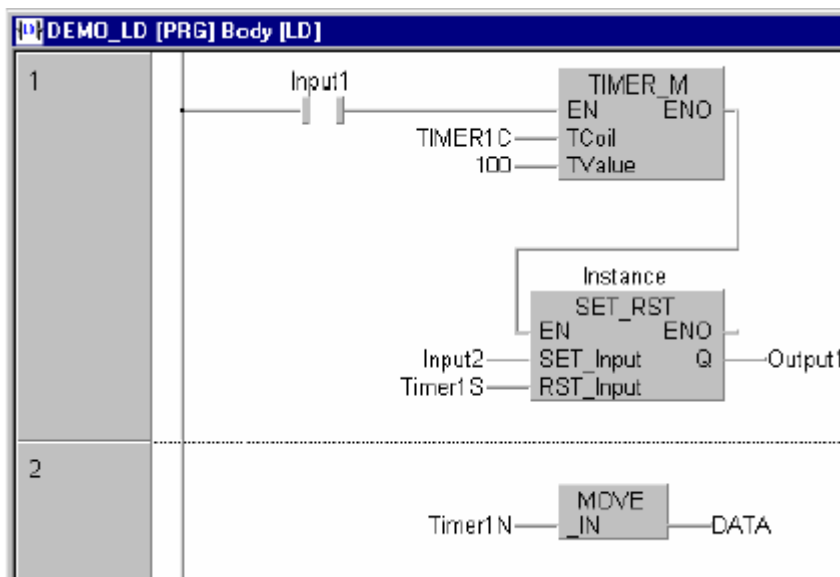
Otevřeme tělo POU DEMO_LD.

Kliknutím na ikonu funkčního bloku v liště nástrojů se otevře okno pro výběr funkčního bloku (Function Block Selection). Zapsáním názvu požadovaného bloku (TIMER_M) do pole Operators nebo pomocí posuvné lišty na pravé straně okna se zobrazí požadovaný blok. Dvojklikem nebo pomocí tlačítka Apply jej vybereme, přemístíme

kurzor myši na pracovní plochu s programem a klikneme levým tlačítkem myši.



Obdobným způsobem vložíme na pracovní plochu následující funkční bloky a pomocí funkce Interconnect/Line je propojíme - vytvoříme následující program.

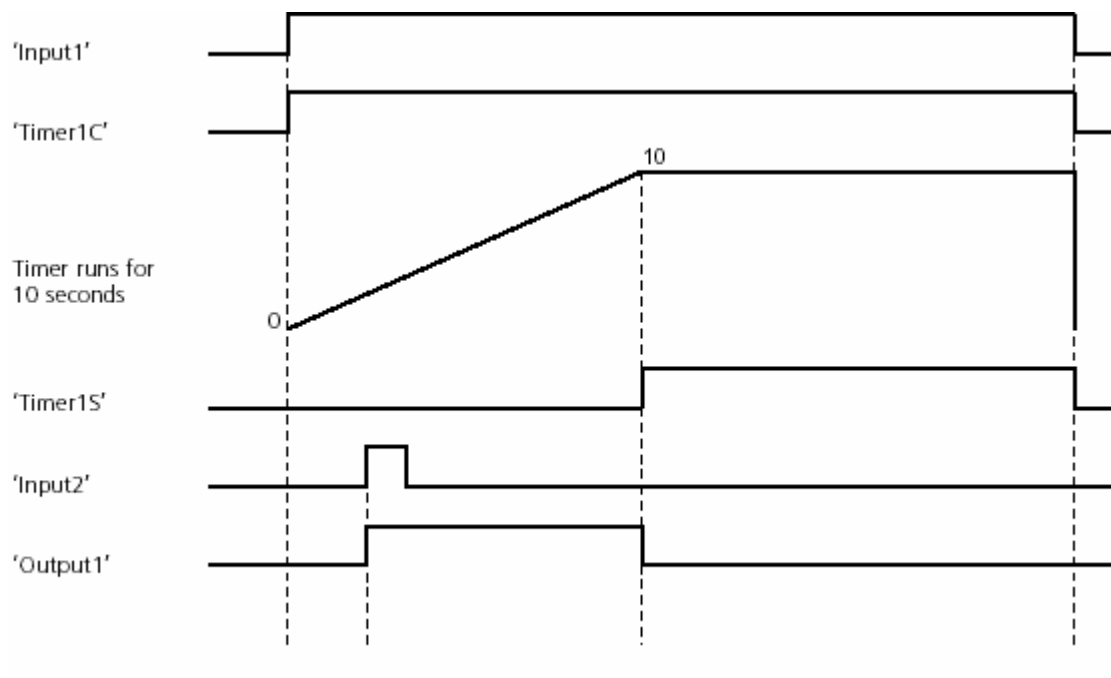


11.1.4 Časový průběh programu

Sekvence časování začíná po nastavení vstupu INPUT 1, který spustí časovač. Je-li nastaven vstup INPUT 2, výstup OUTPUT 1 zapne.

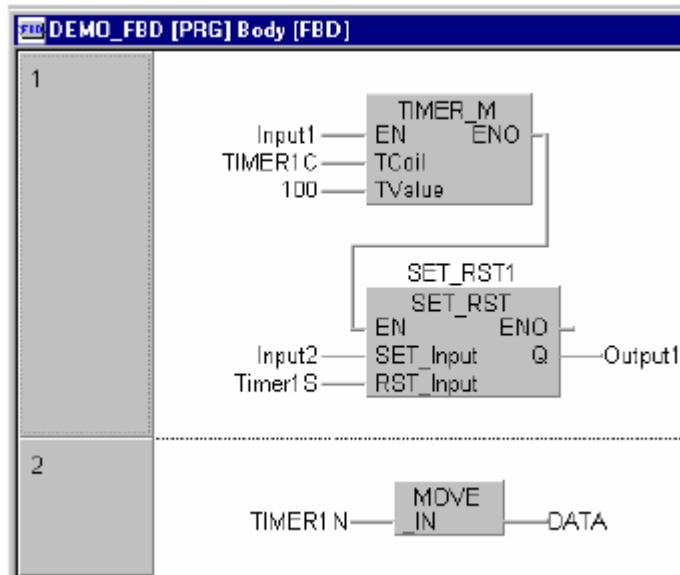
Po uběhnutí přednastaveného času 10s se kontakt TIMER1S sepne a výstup OUTPUT 1 je vypnut.

Programový blok (MOVE) slouží pouze k tomu, aby bylo možno v monitorovacím režimu sledovat časový průběh funkce (průběžný čas).



11.2 Programování časovačů v editoru diagramu fun. bloků (FBD)

Následující obrázek znázorňuje, jak je možno stejný program realizovat v editoru FBD.

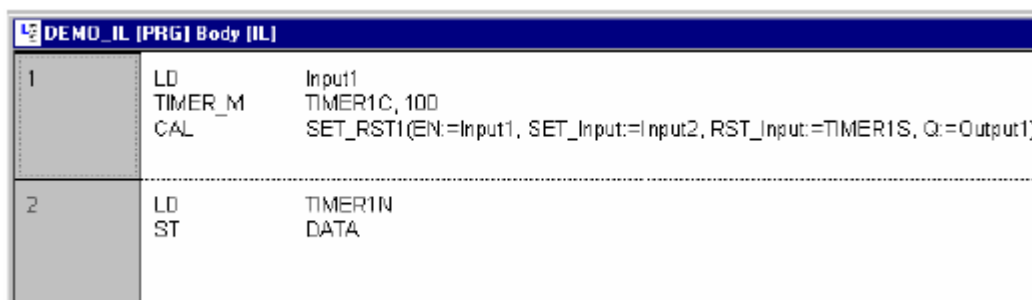


V hlavičce musí být následujícím způsobem zdefinován časovač T2 :

Cívka (Coil)	TIMER2C
Kontakt (Contact)	TIMER2S
Skutečná hodnota (Actual value)	TIMER2N

11.3 Programování časovačů v instrukčním listu (IL)

Následující obrázek znázorňuje jak je možno stejný program realizovat v editoru IL.



Cívka (Coil)	TIMER3C
Kontakt (Contact)	TIMER3S
Skutečná hodnota (Actual value)	TIMER3N



Upozornění. Podívejte se na předchozí kapitoly popisující volání funkcí a vkládání skutečných parametrů v editoru "instrukčního listu".

11.4 Sekvenční funkční diagram (Sequential Function Chart – SFC)

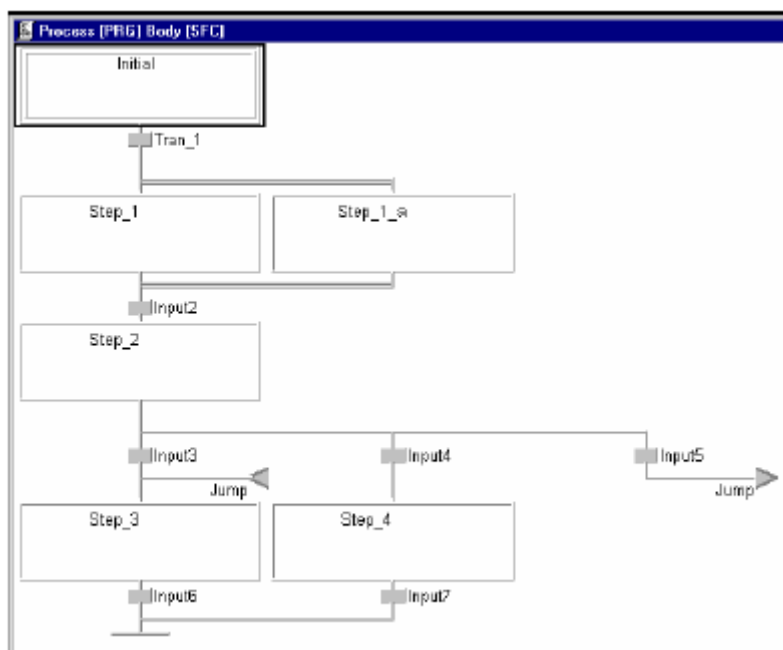
Základní informace o SFC editoru již byly uvedeny v předchozích kapitolách, detailní popis je proveden v Uživatelském manuálu.

Následující programový příklad popisuje krok za krokem, jak postupovat při vytváření programu v SFC editoru pomocí nástrojů GX IEC.



Následující program pojmenovaný Process řeší problém použitím různých druhů větvení a skokových instrukcí.

Zpracování programu a požadovaná funkce v jednotlivých krocích



1. Je-li PLC zapnuto (přepnuto do režimu RUN), zapne se výstup Output 1 (krok Initial).
 2. Přechodovou podmínku „TRAN_1“ tvoří dotaz na stav vstupu „Input1“. Jestliže je sepnut, aktivují se současně kroky „Step_1“ a „Step_1_R“ a (paralelní větve). Výstup „Output2“ bliká a výstup „Output3“ svítí trvale.
 3. Přechodovou podmínkou „Input2“ je přímo stav vstupu „Input2“. Je-li podmínka splněna (vstup „Input2“ je zapnut), začne se vykonávat následující krok „Step_2“ a zapne se výstup „Output4“.
 4. Následuje selektivní rozdělení na tři větve; z podmínek „Input3“, „Input4“ a „Input5“ by měla být splněna pouze jediná.
 5. Jestliže je zapnut vstup „Input3“, přejde se do kroku „Step_3“. Jestliže je zapnut vstup „Input5“, aktivuje se skok „Jump“, přejde se do cílového místa skoku „Jump“ a tím také do kroku „Step_3“. V kroku „Step_3“ se zapne výstup „Output5“.
- Vstup „Input4“ aktivuje pouze krok „Step_4“, ve kterém se sepne výstup „Output6“.

6. Podle toho, která selektivní větev se provádí, vede splnění podmínky „Input6“, resp. „Input7“ na konec programu.

Splnění přechodové podmínky způsobí vždy automaticky deaktivaci předchozího kroku.

11.5 Vytvoření programu

11.5.1 Vytvoření POU

Vytvoříme nové POU (kliknutím na příslušnou ikonu v liště nástrojů), nazveme jej Process a vybereme editor sekvenční funkční diagram (Sequential Function Chart – SFC).

Nově vytvořené POU Process se zobrazí v navigátoru projektem. Na rozdíl od dosud uváděných POU se kromě hlavičky (header) a těla (body) zobrazí ještě skupina akcí (Action_Pool), kde jsou uloženy akce přiřazené jednotlivým krokům daného POU.



11.5.2 Deklarace proměnných v hlavičce

Otevřeme hlavičku a vložíme proměnné použité v POU.

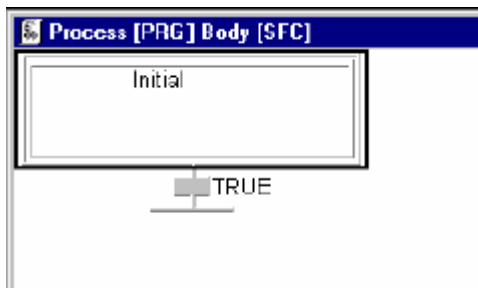
V tomto příkladu jsou použity pouze globální proměnné (kategorie VAR_EXTERNAL).

Process [PRG] Header				
	Class	Identifier	Type	Initial
0	VAR_EXTERNAL	Takt	BOOL	FALSE
1	VAR_EXTERNAL	Input1	BOOL	FALSE
2	VAR_EXTERNAL	Input2	BOOL	FALSE
3	VAR_EXTERNAL	Input3	BOOL	FALSE
4	VAR_EXTERNAL	Input4	BOOL	FALSE
5	VAR_EXTERNAL	Input5	BOOL	FALSE
6	VAR_EXTERNAL	Input6	BOOL	FALSE
7	VAR_EXTERNAL	Input7	BOOL	FALSE
8	VAR_EXTERNAL	Output1	BOOL	FALSE
9	VAR_EXTERNAL	Output2	BOOL	FALSE
10	VAR_EXTERNAL	Output3	BOOL	FALSE
11	VAR_EXTERNAL	Output4	BOOL	FALSE
12	VAR_EXTERNAL	Output5	BOOL	FALSE
13	VAR_EXTERNAL	Output6	BOOL	FALSE
14	VAR_EXTERNAL	Output7	BOOL	FALSE

11.5.3 Otevření těla

Dvojklikem na položku body v navigátoru projektem otevřeme nové okno pro zapsání programu. V nově vytvořeném okně jsou zobrazeny následující elementy :

- inicializační krok (Initial Step)
- přechod TRUE
- koncový krok (Final Step)

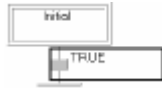




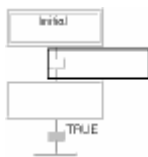
Upozornění. Ikony v liště nástrojů SFC editoru se mění v závislosti na pozici kurzoru, tj. nelze použít ve všech místech programu všechny ikony.

11.5.4 Vytvoření sekvence

Pomocí kurzoru vybereme přechod TRUE



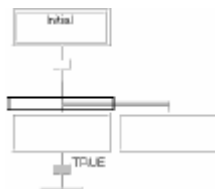
Klikneme na ikonu vložení nového kroku a přechodu.



Vybereme místo do nějž chceme vložit nový blok.



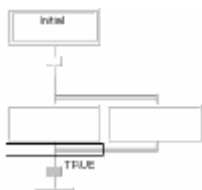
Klikneme na ikonu vložení rozvětvení doprava.



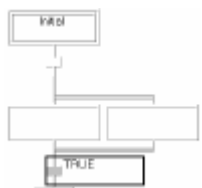
Vybereme přechod TRUE.



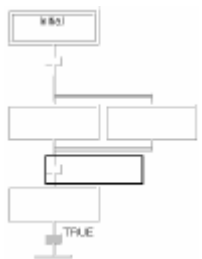
Klikneme na ikonu vložení spojení doprava.



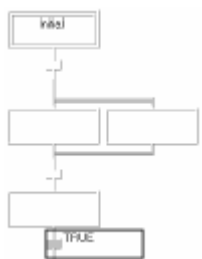
Vybereme přechod TRUE.



Klikneme na ikonu vložení nového kroku a přechodu.



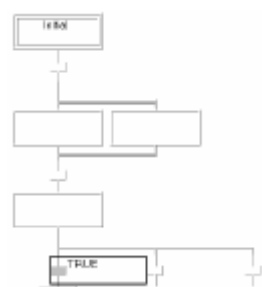
Vybereme přechod TRUE.



Klikneme dvakrát na ikonu vložení rozvětvení doprava.



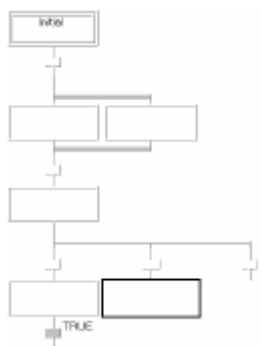
Vybereme přechod TRUE.



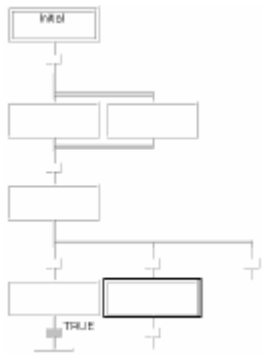
Klikneme na ikonu vložení nového kroku a přechodu.



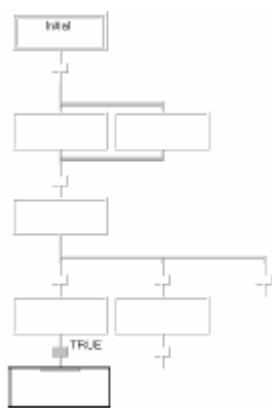
Klikneme na prázdné místo vedle právě vloženého kroku.



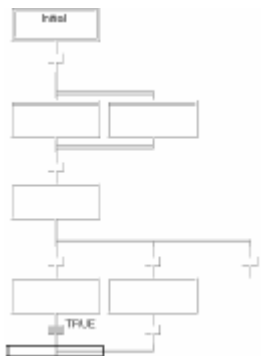
Klikneme na ikonu vložení nového kroku a přechodu.



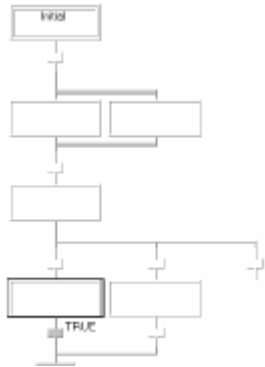
Klikneme na závěrečný krok.



Klikneme na ikonu vložení spojení doprava.



Klikneme na levý krok v dolním řádku.



Klikneme na ikonu vstupní místo skoku (jump entry point).



Klikneme na prázdné místo pod volným přechodem.

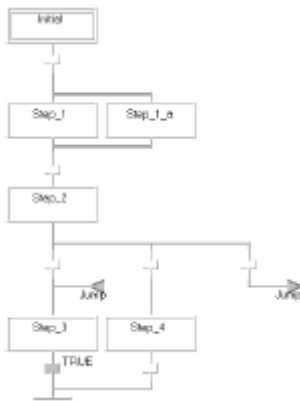


Klikneme na ikonu odchozí místo skoku (jump exit point).



11.5.5 Přřazení jmen ke krokům a návěští skoku

Po vybrání prvku a kliknutí na něj se otevře editační políčko pro zadání jména prvku. Podle níže uvedeného obrázku doplníme jména.



11.5.6 Přiřazení přechodových podmínek k přechodům



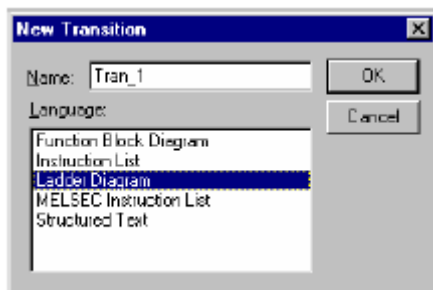
Upozornění. Jako přechodová podmínka může být použita přímo Boolovská (binární) proměnná, TRUE/FALSE (0/1) nebo přechodový program.

Vytvoření a přiřazení přechodového programu

Vybereme a klikneme na přechod, kterému chceme přiřadit jméno, a toto jméno zapíšeme (TRAN_1).



Dvojklikneme na přechod nebo pomocí ikony v liště nástrojů otevřeme dialogové okno pro nový přechod (New Transition) s právě zapsaným jménem (TRAN_1) a vybereme programovací jazyk (v našem příkladu editor kontaktních schémat – LD).

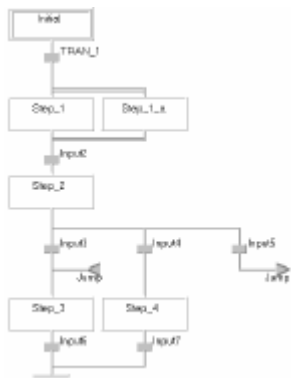


Klikneme na tlačítko OK a na obrazovce se automaticky zobrazí okno pro zapsání přechodového programu, do kterého zapíšeme níže uvedený příklad.




Přiřazení existujících proměnných k přechodům

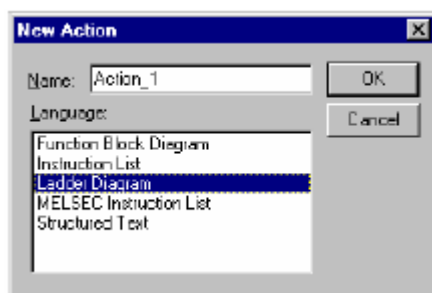
Vybereme a klikneme na jednotlivé přechody a přiřadíme jméno příslušné proměnné dle níže uvedeného obrázku.



Vytvoření akcí

V navigátoru projektem vybereme POU „Process.“

Klikneme na ikonu  v liště nástrojů, čímž se otevře dialogové okno nové akce (**New Action**).



Zapišeme jméno akce (v našem příkladu - Action_1) a vybereme programovací jazyk (editor kontaktních schémát).

Nová akce se objeví ve skupině Action Pool v navigátoru projektem.



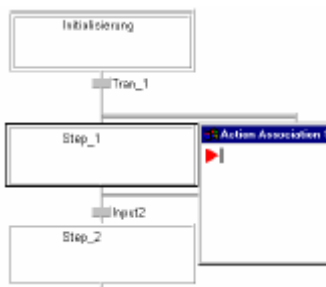
Dvojklikem na Action_1 se otevře okno programového editoru, do kterého zapíšeme níže uvedený program.




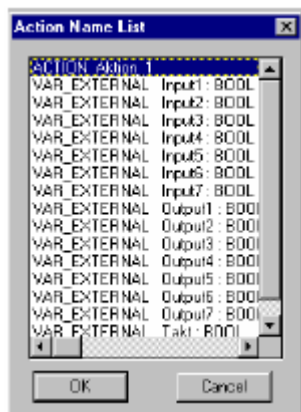
Upozornění. Programy přechodů a akcí (krokové) programy se sestavují (vytvářejí) stejně jako jakékoliv jiné POU. Tyto programy mohou být napsány v editoru LD, IL, nebo FBD. Editor SFC není přístupný.

Přiřazení akcí nebo Boolovských (binárních) proměnných ke krokům

Vybereme krok, kterému chceme přiřadit akci nebo proměnnou (např. krok Step_1).

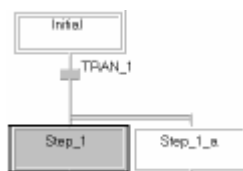


Klikneme na ikonu . Poté se zobrazí nové okno Action Association. Zmáčknutím klávesy F2 se zobrazí seznam akcí (Action Name List) a binárních proměnných, které jsou právě dostupné.




Vybereme příslušné akce a proměnné. Např. Initial = Output1, Step_1 = Action_1, Step_1_a = Output3, Step_2 = Output4, Step_3 = Output5, Step_4 = Output6).

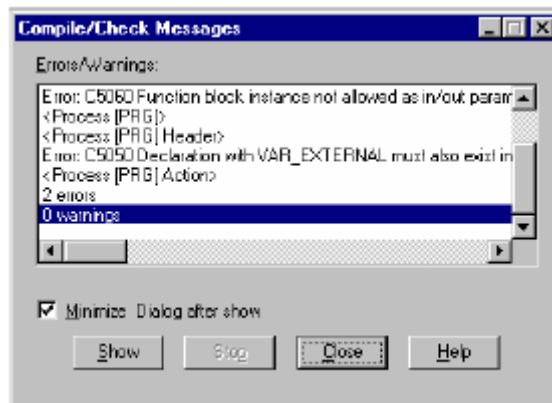
Uzavřeme okno se seznamem akcí.



12 Kontrola PLC programu (syntax check)

V navigátoru projektem vybereme objekt, jenž chceme zkontrolovat, a vybereme příkaz Check v menu Object nebo klikneme na ikonu  v liště nástrojů.

Po ukončení kontroly se objeví dialogové okno s jejím výsledkem.



Pokud byla při kompilaci zjištěná nějaká chyba, objeví se ve výše uvedeném okně její popis. Po najetí na tuto chybu a dvojkliknutí myši nebo zmáčknutím tlačítka Show se zobrazí příslušná část programu s červeným podbarvením chybového úseku.



Upozornění. Kontrolu je možno provádět po částech (po jednotlivých objektech) nebo jako celek. Rovněž je možno spustit pouze kontrolu těla nebo hlavičky jednoho POU. Kontrolu samostatného POU provedeme vybráním daného POU v navigátoru projektem a spuštěním kontroly. Kontrolu celého projektu provedeme kliknutím na první (nejvyšší) řádek v navigátoru projektem a spuštěním kontroly.

13 Konfigurace tasku

V této kapitole je popsán způsob zařazování jednotlivých POU do tasků a konfigurace parametrů tasků.

13.1 Vytvoření nového tasku

V navigátoru projektem najedeme na položku Task_Pool a pomocí ikony TSK v liště nástrojů, nebo příkaz New – Task v menu Object otevřeme nové okno (New Task), v němž zadáme jméno nového tasku.

Po zadání jména se nový task automaticky zařadí do skupiny tasků (Task_Pool) v navigátoru projektem.

13.2 Zařazení POU do tasku

Dvojklikneme na požadovaný task v navigátoru projektem, čímž se na pravé straně obrazovky otevře nové okno, v jehož záhlaví je uvedeno jméno a parametry zvoleného tasku.



	POU name	Comment
0	Control	

Do prvního řádku (0) a sloupce POU-Name můžeme zapsat jméno POU, které chceme do tohoto tasku zařadit. Pokud si nejsme jisti, je možno zmáčknou výběrovou šipku (ve sloupci POU-Name), čímž se otevře okno se seznamem všech použitelných POU.

V tomto okně vybereme POU a volbu potvrdíme OK.



Upozornění. V nabídce jsou uvedeny pouze POU definované jako program (PRG), které ještě nebyly zařazeny do žádného tasku. Tj. jedno POU nemůže být použito ve více tascích současně.

Pomocí tabelátoru je možno přemístit kurzor do sloupce Comment, kde můžeme doplnit popis k danému řádku.

Stejným způsobem pokračujeme při vkládání dalších POU. Vložení nového řádku provedeme pomocí ikony v liště nástrojů nebo pomocí příkazů New Declaration v menu Edit. Pomocí příkazu New Declaration lze specifikovat umístění nového POU v tasku (nahore, před aktuální řádek, za aktuální řádek, dolů), tj. pořadí, v jakém budou jednotlivé POU zpracovávány v PLC.

Nový řádek lze rovněž vložit pomocí klávesy tabelátoru. Pro toto vložení je však nutno být v editovacím režimu ve sloupci Comment (pole je podbarveno tmavě šedou barvou). Po vložení všech požadovaných POU do tasku lze okno zavřít kliknutím na symbol pro uzavírání (x) v pravém horním okně. Po výzvě k uložení nebo zrušení změn (Save changes of editor session) zvolíme Ano.

13.3 Konfigurace parametrů tasku

V navigátoru projektem vybereme task (task se podbarví tmavě šedou barvou) a kombinací kláves Alt + Enter nebo pomocí příkazu Information v menu Object otevřeme dialogové okno Task Information.

V tomto okně je možno nastavit tyto parametry a podmínky pro vykonávání tasku :
Event – událost, od které se úloha zpracuje. Může zde být binární proměnná (vstup, výstup, merkr apod.). Přednastavená hodnota TRUE znamená cyklické (nepodmíněné) zpracování v každém skenu, FALSE znamená časové zpracování.

Interval – pokud je v prvním řádku (Event) navoleno časové zpracování, v tomto poli zadáme časový interval, v jakém se má úloha spouštět.

Priority – je-li od jedné události (proměnné) spouštěno více úloh, je možno v tomto poli zadat prioritu (pořadí) jejich spouštění. Přednastavená hodnota 31 = nejnižší priorita, 1 = nejvyšší priorita.

Name – umožňuje změnu jména vybraného tasku

Velikost (Size), Typ (Type) a údaj o poslední provedené změně (Last Change) není možno editovat. Údaje jsou automaticky přiřazeny programem GX IEC.

Pomocí tlačítka Comment v pravé části okna, se otevře nové editační okno, do něhož lze zapsat další informace o dané úloze.



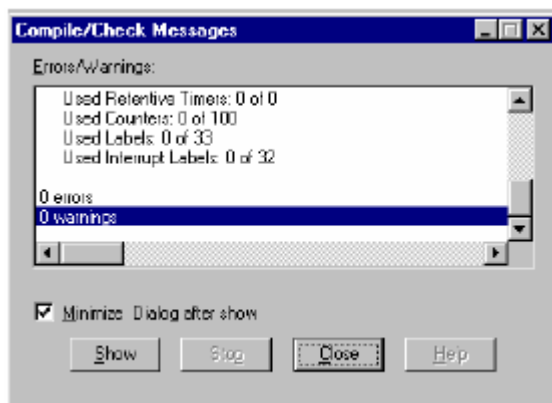
Upozornění. Podrobnější informace o parametrech Tasku jsou uvedeny v Uživatelském manuálu.

14 Kompilace projektu

Před nahráním námi vytvořeného programu do PLC je nutné provést kompilaci projektu. Při této kompilaci dochází k překladi programu z jeho grafické podoby do kódu zpracovatelného řídicím systémem.

Kompilaci projektu je možno provést pomocí ikony v liště nástrojů nebo příkazem Compile Project v menu Project.

Průběh kompilace, popř. chyby odhalené kompilátorem, jsou zobrazovány v samostatném okně.



Upozornění. Při kompilaci ještě nedochází k přenosu programu do PLC. Přenos je realizován samostatnou funkcí Transfer.

Do PLC může být přenášén pouze zkompilovaný program bez chyb.

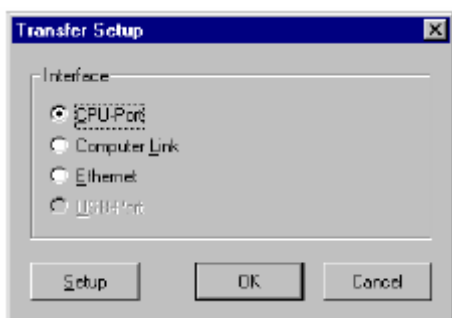
15 Nastavení komunikačního portu

Před nahráním programu do PLC je nutno zkontrolovat, popř. změnit nastavení parametrů komunikačního portu.

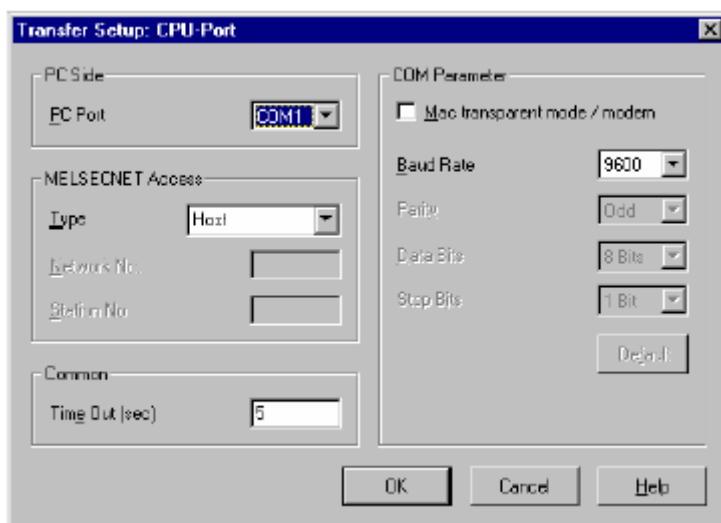
Nejprve vybereme Transfer Setup v menu Online. V menu Transfer Setup vybereme Ports, čímž dojde k otevření níže uvedeného okna.



Upozornění. Vzhled okna se může částečně lišit s ohledem na různé verze software GX IEC.



Myší klikneme na tlačítko Setup.



Vybereme odpovídající komunikační port, na který máme připojený programovací kabel, a volbu potvrdíme OK.

16 Nahrání programu do PLC

Je-li vytvořený program zkontrolován a zkompileván, je možno jej nahrát do řídicího systému.

Připojení k PLC

Připojte programovací kabel k PLC a vašemu počítači. Zkontrolujte nastavení parametrů komunikačního portu.



Upozornění. Možnosti nastavení různých komunikačních parametrů pro různé typy připojení jsou popsány v Uživatelském manuálu.

Nahrání programu

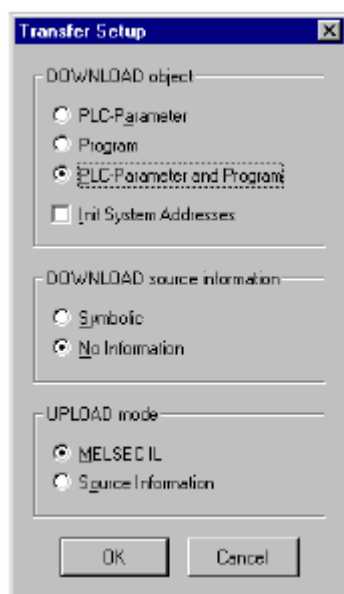
Nahrání programu provedeme volbou **Transfer** (v menu **Project**) a příkazem **MEDOC to PLC**.

Průběh přenosu včetně případných chyb je průběžně zobrazován v samostatném okně.

Při nahrávání programu je možno uživatelsky zvolit několik různých variant. Dialogové okno pro tuto volbu vyvoláme volbou **Transfer Setup** (v menu **Online**) a **Project**.

V tomto okně lze navolit, zda chceme přenášet program s parametry, pouze parametry nebo pouze program (v případě automatů řady FX0S, FX1S, FX0N, FX1N a FX2N nelze tento způsob přenosu navolit).

Dále je možno navolit, zda chceme přenášet program včetně dalších informací (struktura programu v editorech GX IEC) nebo bez informací (pouze „holý“ program). Je-li program nahrán bez informací (No information), je při zpětném přenosu (z PLC do Medocu) možno zobrazit tento program pouze v instrukčním listu.



Upozornění. Pokud se program přenáší do PLC poprvé, je nutno vždy přenést také parametry PLC. PLC řady Q musí být nejdříve zformátovány.

17 Monitorování programu.

V monitorovacím režimu může GX IEC zobrazovat aktuální stavy a změny proměnných uvnitř PLC.



Upozornění. Je možno monitorovat pouze „běžící“ program. Tj. program, který byl zkompilován a nahrán do PLC.

Monitorovací režim se aktivuje příkazem Monitoring Mode v menu Online. Zatřítko před příkazem signalizuje, zda je režim právě aktivní.

V navigátoru projektem vybereme POU které chceme monitorovat a zvolíme příkaz Start Monitoring v menu Online.



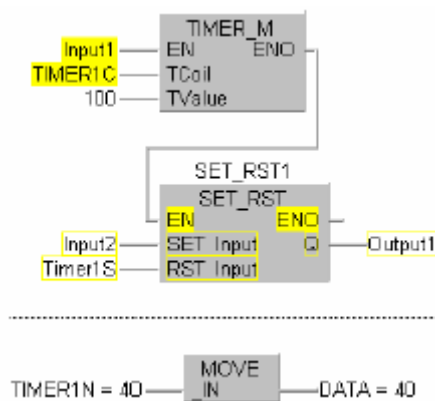
Upozornění. Monitorovací režim lze rovněž aktivovat přímo příkazem Start Monitoring.

Následující příklad znázorňuje způsob zobrazení proměnných v monitorovacím režimu pro různé programovací jazyky (editory).



Upozornění. Více informací je o monitorovacím režimu a způsobech zobrazení je uvedeno v Uživatelském manuálu.

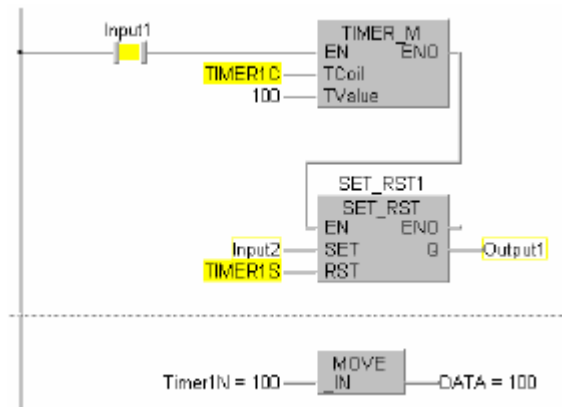
Monitorování v editoru funkčních bloků (FBD)



Vyplněný obdélník – binární proměnná je aktivní (ON)
Prázdný obdélník – binární proměnná není aktivní (OFF)

40 – uplynulý čas (doba běhu časovače) 4,0 vteřiny

Monitorování v editoru kontaktních schémat (LD)



Vyplněné pole mezi kontakty (např. Input1) – podmínka je splněna (v tomto případě – signál je ON)

Vyplněný obdélník – binární proměnná je aktivní (ON)

Prázdný obdélník – binární proměnná není aktivní (OFF)

100 – uplynulý čas (doba běhu časovače) 10,0 vteřin

Monitorování v editoru seznamu instrukcí (IL)

```
LD      Input1
TIMER_M  TIMER1C, 100
CAL     SET_RST1(EN:=Input1, SET:=Input2, RST:=TIMER1S, Q:=Output1)
-----
LD      TIMER1N    100
ST      DATA      100
```

Vyplněný obdélník – binární proměnná je aktivní (ON)

Prázdný obdélník – binární proměnná není aktivní (OFF)

18 Nahrávání programu z PLC

V položce **Transfer** (menu **Project**) vybereme volbu PLC to MEDOC. V nově otevřeném dialogovém okně **Transfer Setup** máme možnost zkontrolovat nebo změnit parametry komunikačního portu. Po provedení kontroly zmáčkeme tlačítko OK, čímž se otevře okno **Transfer Setup**, v němž navolíme způsob přenosu.

Přednastavený režim MELSEC IL přenese program z PLC do GX IEC a umožní tento program zobrazit pouze v editoru seznamu instrukcí (IL).

Režim Source Information přenese program z PLC do GX IEC a umožní následně tento program zobrazit ve struktuře (rozdělení do jednotlivých tasků a POU apod.) a editoru, ve které byl původně vytvořen. Tento způsob přenosu je možno realizovat pouze v případě, že program byl do PLC nahrán s výše uvedenými informacemi, tj. jako Symbolic.

Po navolení požadovaného způsobu přenosu potvrdíme volbu tlačítkem OK a program se začne přenášet. Přitom dojde k přepsání aktuálních parametrů projektu, do nějž program nahráváme a do projektu jsou přidány nové POU (PRG_MAIN) a nový task (MELSEC_MAIN).

Průběh přenosu včetně případných chyb je průběžně zobrazován v samostatném okně.

19 Vzorový program – Řízení garáží

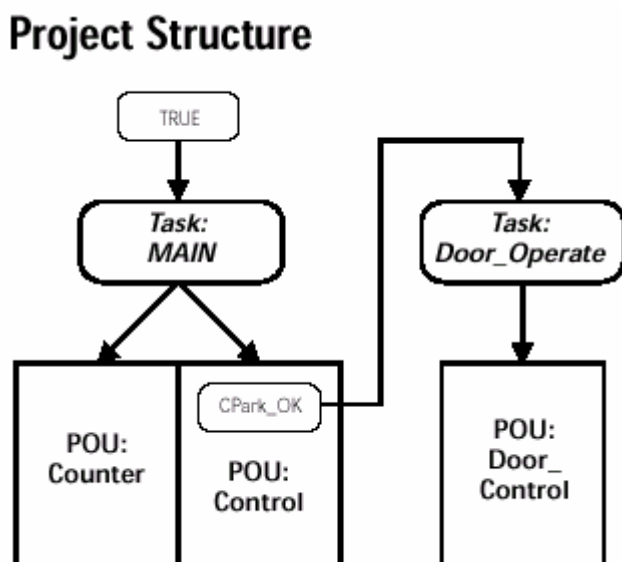


Upozornění. Tento příklad slouží pouze ke znázornění programové struktury a programovací techniky v GX IEC.

Příklad je napsán pro řídicí systém MELSEC FX a je závislý na použité verzi software GX IEC, tj. pro některé z verzí je pro plnou funkčnost programu nutno provést drobné úpravy.

19.1 Zadání

Automatické dveře do podzemních garáží mohou být otevřeny zevnitř nebo zvenku pomocí spínače s klíčem. Program musí dále zajistit uzavření dveří, pokud delší časový interval neprojíždí dovnitř ani ven žádné auto, a hlídání (počítání) počtu aut v garáži. Bezpečnostní funkce obsažená v programu musí zabezpečit automatické otevření dveří v případě alarmu.



19.2 Části programu

Task Main - běží neustále (cyklicky) s maximální prioritou. Tento Task obsahuje programové moduly Control a Counter, které zajišťují následující funkce :

POU Control kontrola stavu garáží
uzavření dveří, neprojelo-li za posledních 60 vteřin žádné auto
otevření dveří v případě spuštění požárního alarmu

POU Counter počítání aut

Task Door_Operate – je spouštěn od události (event-triggered), tj. je aktivován, je-li signál OK pro dveře nastaven (proměnná Cpark_OK). Tento Task obsahuje programový modul Door_Control, který zajišťuje následující funkce :

POU Door_Control otevření dveří, je-li aktivován klíčový přepínač na vnitřní nebo vnější straně dveří
uzavření dveří, aktivovalo-li (projelo) auto optickou závoru



Upozornění. V našem příkladě jsou všechny proměnné známy a deklarovány na začátku projektu. Tento ideální stav se však vyskytuje málokdy a většinou je třeba proměnné měnit a doplňovat až v průběhu psaní programu. Tato činnost je plně podporována software GX IEC, neboť umožňuje korigovat a vkládat proměnné ve kterékoliv fázi vytváření programu, tj. před zahájením programování, v jeho průběhu nebo až po napsání programu.

19.2.1 Vytvoření nového projektu CarPark

Prvním krokem při psaní programu je vytvoření nového projektu s názvem CarPark.

19.2.2 Vytvoření tasků

Vytvoříme task Main a task Door_Operate.

19.2.3 Deklarace globálních proměnných

Deklarujeme globální proměnné zobrazené v dolní tabulce. Texty ve sloupci Comment (komentář) jsou volitelné (nejsou nutné pro činnost programu).

Global Variable List								
	Class	Autostatem	Identifier	MIT-Addr.	IEC-Addr.	Type	Initial	Comment
0	VAR_GLOBAL		Door_Open	X0	%X0	BOOL	FALSE	Upper door limit switch
1	VAR_GLOBAL		Door_Closed	X1	%X1	BOOL	FALSE	Lower door limit switch
2	VAR_GLOBAL		Time_Control	M1	%M0.1	BOOL	FALSE	Internal relay for the close
3	VAR_GLOBAL		Motor_Up	Y0	%Q0	BOOL	FALSE	Motor rolls car park door i
4	VAR_GLOBAL		Motor_Down	Y1	%Q1	BOOL	FALSE	Motor rolls car park door i
5	VAR_GLOBAL		Enter_Up	X2	%X2	BOOL	FALSE	Key switch door open for
6	VAR_GLOBAL		Exit_Up	X3	%X3	BOOL	FALSE	Key switch door open for
7	VAR_GLOBAL		Enter_Car_Gone	X4	%X4	BOOL	FALSE	Photoelectric barrier for c:
8	VAR_GLOBAL		Exit_Car_Gone	X5	%X5	BOOL	FALSE	Photoelectric barrier for c:
9	VAR_GLOBAL		Max_Time_Up_C	T0	%M5.0	BOOL	FALSE	Timer coil: When the time the car park door is close
10	VAR_GLOBAL		Max_Time_Up_S	T5	%M3.0	BOOL	FALSE	Timer contact: When the the car park door is close
11	VAR_GLOBAL		Main_Switch	X6	%X6	BOOL	FALSE	Main switch: Car park doc
12	VAR_GLOBAL		Help_Alarm	X7	%X7	BOOL	FALSE	Alarm switch
13	VAR_GLOBAL		CO2_Alarm	?	X8	BOOL	FALSE	CO2 sensor
14	VAR_GLOBAL		CPark_OK	M0	%M0.0	BOOL	FALSE	OK signal: Car park door
15	VAR_GLOBAL		CPark_OK_Lamp	Y3	%Q3	BOOL	FALSE	OK signal: Car park door
16	VAR_GLOBAL		Cars_Number	D0	%NW0.0	INT	0	Number of cars in the car

19.2.4 Vytvoření programových modulů (POU)

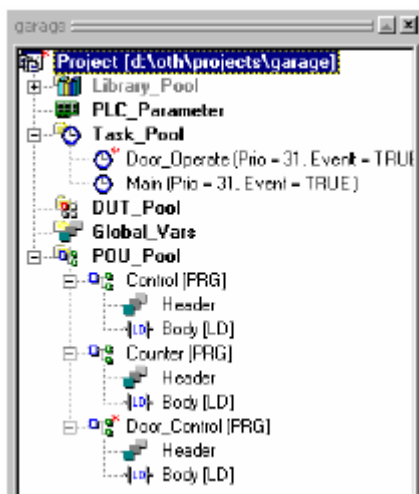
Vytvoříme tři programové moduly (POU) – Control, Counter a Door_Control. Všechny tři POU definujeme jako programy (PRG) a specifikujeme programovací jazyk LD (kontaktní schéma).

Každé POU obsahuje hlavičku a tělo. V hlavičce jsou umístěny proměnné použité v daném POU, tělo obsahuje vlastní PLC program.

Okno navigátoru projektem

Všechny vytvořené tasky a POU jsou automaticky umístěny a zobrazeny v okně navigátoru projektem.

Tasky – Door_Operate a Main
 POU - Control, Counter a Door_Control



19.3 Hlavičky programových modulů

Všechny proměnné použité v tomto příkladu jsou definovány jako globální proměnné. Z databáze globálních proměnných je možné tyto proměnné do jednotlivých hlaviček překopírovat přes clipboard pomocí kláves CTRL+C a vložit CTRL+V, popř. zatrhnout u jednotlivých proměnných v seznamu globálních proměnných volbu Autoextern.



Upozornění. Všechny další kombinace kláves použitelné pro práci s tabulkou (databáze globálních proměnných) jsou uvedeny v Uživatelském manuálu.

19.3.1 Hlavička programového modulu Control

Control (PRG) Header					
	Class	Identifier	Type	Initial	Comment
0	VAR_EXTERNAL	Door_Open	BOOL	FALSE	Upper door limit switch
1	VAR_EXTERNAL	Door_Closed	BOOL	FALSE	Lower door limit switch
2	VAR_EXTERNAL	Time_Control	BOOL	FALSE	Internal relay for the close process
3	VAR_EXTERNAL	Motor_Up	BOOL	FALSE	Motor rolls car park door up
4	VAR_EXTERNAL	Motor_Down	BOOL	FALSE	Motor rolls car park door down
5	VAR_EXTERNAL	Enter_Up	BOOL	FALSE	Key switch door open for entry
6	VAR_EXTERNAL	Exit_Up	BOOL	FALSE	Key switch door open for exit
7	VAR_EXTERNAL	Enter_Car_Gone	BOOL	FALSE	Photoelectric barrier for car entrance
8	VAR_EXTERNAL	Exit_Car_Gone	BOOL	FALSE	Photoelectric barrier for car exit
9	VAR_EXTERNAL	Max_Time_Up_C	BOOL	FALSE	Timer coil: When the time has elapsed the car park door is closed
10	VAR_EXTERNAL	Max_Time_Up_S	BOOL	FALSE	Timer contact: When the time has elapsed the car park door is closed
11	VAR_EXTERNAL	Main_Switch	BOOL	FALSE	Main switch: Car park door is in operation
12	VAR_EXTERNAL	Help_Alarm	BOOL	FALSE	Alarm switch
13	VAR_EXTERNAL	CO2_Alarm	BOOL	FALSE	CO2 sensor
14	VAR_EXTERNAL	CPark_OK	BOOL	FALSE	OK signal: Car park door can be used
15	VAR_EXTERNAL	CPark_OK_Lamp	BOOL	FALSE	OK signal: Car park door can be used

19.3.2 Hlavička programového modulu Counter

Counter (PRG) Header					
	Class	Identifier	Type	Initial	Comment
0	VAR_EXTERNAL	Enter_Car_Gone	BOOL	FALSE	Photoelectric barrier for car entrance
1	VAR_EXTERNAL	Exit_Car_Gone	BOOL	FALSE	Photoelectric barrier for car exit
2	VAR_EXTERNAL	Cars_Number	INT	0	Number of cars in the car park

Hlavička programového modulu Door_Control

Door_Control (PRG) Header					
	Class	Identifier	Type	Initial	Comment
0	VAR_EXTERNAL	Door_Open	BOOL	FALSE	Upper door limit switch
1	VAR_EXTERNAL	Door_Closed	BOOL	FALSE	Lower door limit switch
2	VAR_EXTERNAL	Motor_Up	BOOL	FALSE	Motor rolls car park door up
3	VAR_EXTERNAL	Motor_Down	BOOL	FALSE	Motor rolls car park door down
4	VAR_EXTERNAL	Enter_Up	BOOL	FALSE	Key switch door open for entry
5	VAR_EXTERNAL	Exit_Up	BOOL	FALSE	Key switch door open for exit
6	VAR_EXTERNAL	Enter_Car_Gone	BOOL	FALSE	Photoelectric barrier for car entrance
7	VAR_EXTERNAL	Exit_Car_Gone	BOOL	FALSE	Photoelectric barrier for car exit
8	VAR_EXTERNAL	Max_Time_Up_C	BOOL	FALSE	Timer coil: When the time has elapsed the car park door is closed
9	VAR_EXTERNAL	Max_Time_Up_S	BOOL	FALSE	Timer contact: When the time has elapsed the car park door is closed

19.4 Těla programových modulů

19.4.1 Tělo programového modulu Control

Aktivace řídicího procesu

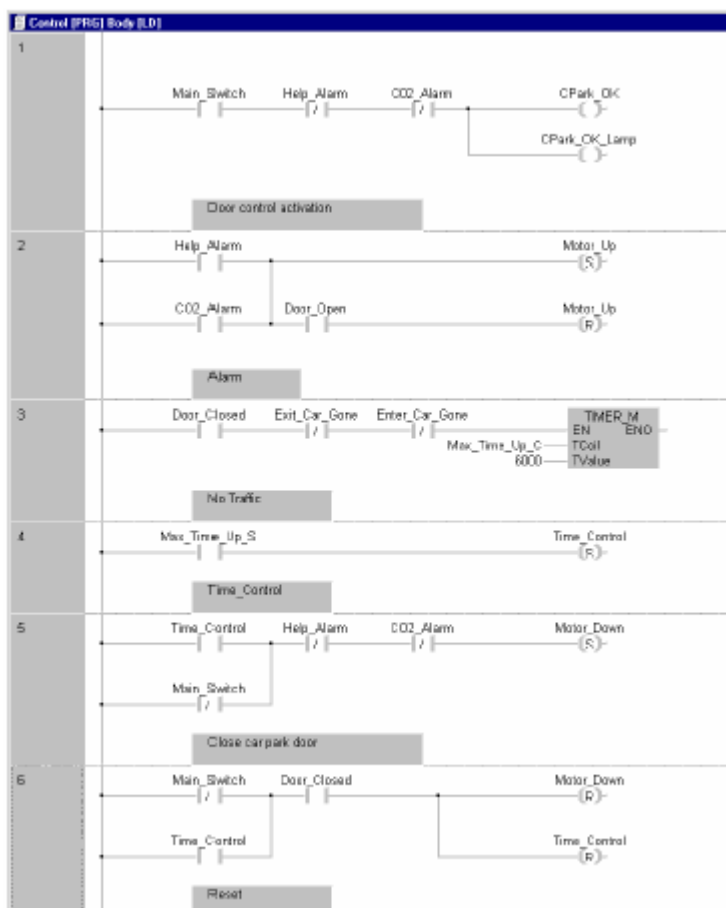
Požadavek pomoci (Help_alarm) nebo požární alarm (CO2_alarm)

Aktivace časovače

Vyhodnocení časovače

Uzavření dveří

Reset



Aktivace řídicího procesu

Je-li zapnut hlavní vypínač a není žádný alarm, je nastaven OK signál pro POU Door_Control a výstup CPark_OK_Lamp je sepnut.

Požadavek pomoci (Help_call) a požární alarm (CO2_alarm)

Je-li zaznamenán některý z těchto alarmů, je okamžitě vydán příkaz k zapnutí motoru zvedajícího dveře. Motor je vypnut (resetován) po otevření dveří (sepnutí horního koncového spínače) – Door_Open.

Aktivace časovače

Jsou-li otevřeny dveře a optická závora u vjezdu (Enter_Car_Gone) a optická závora u výjezdu (Exit_Car_Gone) neregistrují žádné auto, je aktivován časovač Max_Time_Up_C.

Vyhodnocení časovače

Výstup z časovače je aktivován po uplynutí 600 vteřin aktivován.

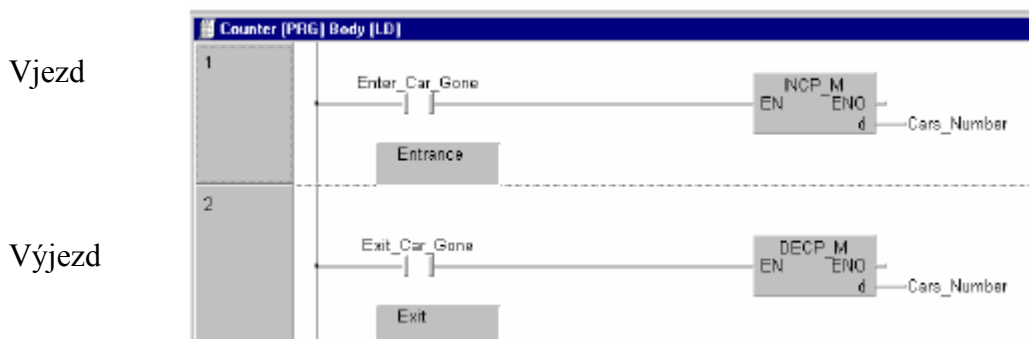
Uzavření dveří

Není-li optickými závorami registrován žádný pohyb nebo je vypnut hlavní vypínač a není žádný alarm, je vydán příkaz k uzavření dveří.

Reset

Není-li registrován žádný pohyb nebo je vypnut hlavní vypínač a dveře se uzavřely (dolní koncový spínač je sepnut), motor i časovač jsou vypnuty (resetovány).

19.4.2 Tělo programového modulu Counter



Vjezd

Program počítá auta vjíždějící do garáže inkrementováním (zvyšováním o jedničku) hodnoty v datovém registru označeného Cars_Number.

Výjezd

Program s každým vyjetým autem dekrementuje (sníží o jedničku) hodnotu v datovém registru Cars_Number.

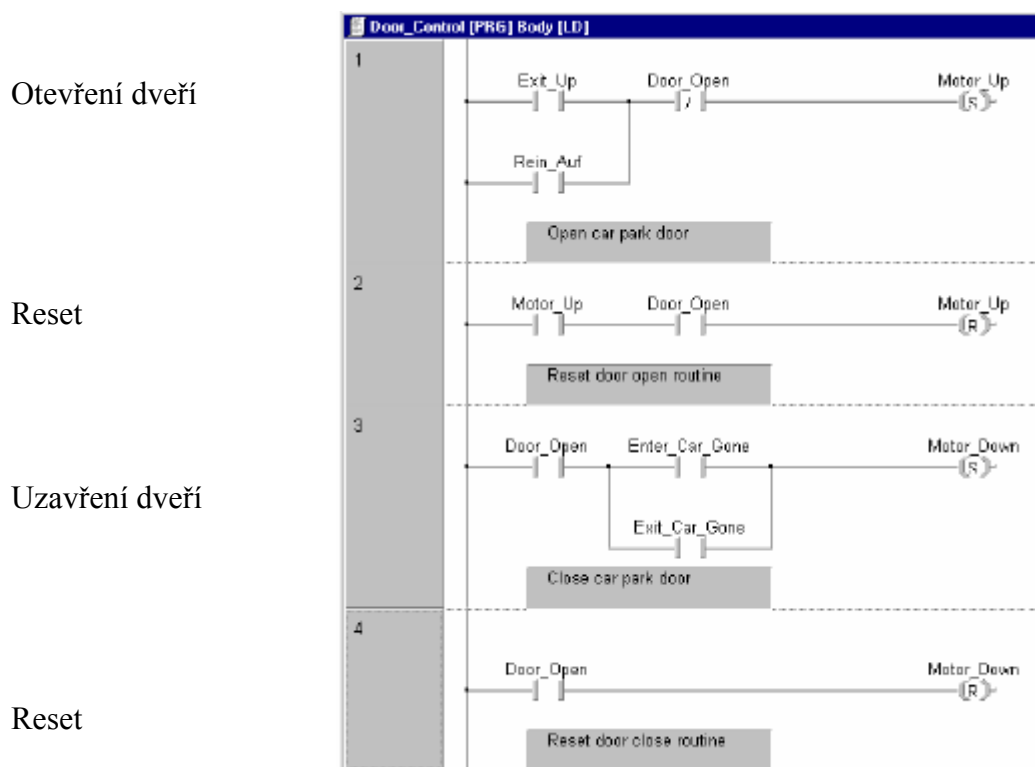
Výsledkem výše uvedených činností je, že v datovém registru je neustále uložen údaj o aktuálním počtu aut v garáži.

19.4.3 Tělo programového modulu Door_Control

Podmínky pro aktivaci programového modulu

Tento programový modul může být vykonáván pouze v případě, že signál CPark_OK je aktivní (setován). Tento signál je aktivní, pokud

- je zapnut hlavní vypínač a
- není registrován alarm požadavek pomoci (Help_call) a
- není registrován požární alarm (CO2 alarm)



Otevření dveří

Jsou-li uzavřeny dveře, je nutno k jejich otevření aktivovat vnitřní klíčový přepínač (Exit_Up) nebo vnější klíčový přepínač (Enter_Up).

Reset

Motor je vypnut (resetován), jsou-li dveře otevřeny (je aktivován horní koncový spínač) – Door_Open.

Uzavření dveří

Dveře se začnou zavírat, aktivuje-li automobil optickou závoru na vjezdu (Enter_Car_Gone) nebo na výjezdu (Exit_Car_Gone).

Reset

Motor je vypnut (resetován) po zavření dveří (je aktivován dolní koncový spínač) – Door_Closed.

19.5 Konfigurace tasků

V úvodu tohoto příkladu jsme vytvořili dva tasky potřebné pro náš program – task Main a task Door_Operate.

Nyní potřebujeme přiřadit jednotlivá POU k těmto taskům, které jsou dosud prázdné. Dvojklikem na jméno tasku v navigátoru projektem se otevře nové okno nazvané Task Main.

19.5.1 Task Main

Main (Prio - 31, Event - TRUE)		
	POU name	Comment
0	Control	
1	Counter	

Kliknutím na šipku ve sloupci POU-Name se zobrazí seznam všech dostupných POU, z něhož vybereme Control a poté Counter.

Po přiřazení POU je nutno k jednotlivým taskům přiřadit parametry. Vybereme task v navigátoru projektem a otevřeme konfigurační tabulku kombinací kláves ALT+ENTER nebo příkazem **Information** v menu **Objekt**.

Task Information

Task Attributes

Event: Garage-OK

Interval: 0

Priority: 31

Name: Door_Operate

Size: 98 Bytes

Type: TASK Timer/Output Control

Last Change: 05.04.2000 15:56:33

Security Level: 0 1 2 3 4 5 6 7

Allow Read Access for lower Levels

Buttons: OK, Cancel, Comment

19.5.2 Task Door_Operate

Task Door_Operate obsahuje pouze jeden programový modul (Door_Control).

Door_Operate (Prio = 31, Event = CPark_OK)	
POU name	Comment
0: Door_Control	

Parametry tasku zadefinujeme v tabulce Task Information.

Task Information

Task Attributes

Event: CPark_OK

Interval: 0

Priority: 31

Name: Door_Operate

Size: 177 Bytes

Type: TASK Timer/Output Control

Last Change: 02.05.01 15:11:57

Security Level: 0 1 2 3 4 5 6 7

Allow Read Access For Lower Levels

OK Cancel Comment...

Event - CPark_OK (úloha se spouští pouze tehdy, je-li tento signál aktivní)

Ukončení projektu

V této chvíli je ukončeno vlastní programování projektu. Před nahráním do řídicího systému je ještě nutné provést kontrolu projektu, jeho kompilaci a kontrolu (popř. změnu nastavení) komunikačního portu.

Po nahrání projektu do řídicího systému je možné zapnout monitorovací režim a zkontrolovat funkčnost programu v reálném čase.

20 Import projektu vytvořeného v MELSEC MEDOCu

Import projektu vytvořeného ve starší verzi MELSEC MEDOC do software GX IEC je možno provést dvěma způsoby :

1. Importování tiskového souboru (print file)
2. Importování nahráním projektu z řídicího systému

1. Importování tiskového souboru

Příprava tiskového souboru v MELSEC MEDOCU

Otevřeme projekt který chceme importovat, vybereme PRINT – Options – Output a zadáme jméno souboru. Přípona .TMP bude přidána automaticky.

Klávesou ESC se vrátíme o jednu úroveň zpět a v nabídce vybereme pro tisk pouze instrukční list (Instruction List) a názvy proměnných (Name List). Hlavička musí být vypnuta (Print without header).

Spustíme tisk do souboru (Go).

Načtení tiskového souboru do GX IEC

Otevřeme tělo (body) existujícího programu vytvořeného v MELSEC instrukčním listě (MELSEC IL) nebo vytvoříme nový projekt a v něm nové POU s programovacím jazykem (editorem) MELSEC IL. POU musí být deklarováno jako program (PRG).

Vybereme příkaz Import MEDOC Network v menu Tools.

V nově otevřeném okně specifikujeme tiskový soubor (.TMP) a cestu k tomuto souboru. Výběr potvrdíme OK.

Tím dojde k otevření dalšího okna, v němž navolíme, zda chceme přenášet program (MEDOC Program = pouze seznam instrukcí) nebo symbolické názvy (MEDOC Symbolic Names) nebo obojí.

Průběh importování je zobrazován v dialogovém okně.

2. Importování nahráním projektu z řídicího systému

Nahrání projektu z MELSEC MEDOCu do řídicího systému.

Nahrání projektu do GX IEC (menu Project-Transfer-PLC to MEDOC).