

23

23. Doplňky

23.1 Transformace souřadného systému

Transformace souřadného systému je popsána v „**Návodu na programování**“. Na tomto místě se budeme jen zabývat problematikou řízení transformace z PLC programu. Pro další výklad se předpokládá znalost základního návodu pro transformaci souřadnic, který je uveden v samostatné kapitole v návodu na programování.

Systém umožňuje použít nezávisle na sobě 2 rovinné transformace (2D) a jednu prostorovou (3D) transformaci souřadnic. Každá souřadnice může být najednou zařazena jen v jedné transformaci.

Řízení transformace souřadnic je umožněno z NC partprogramu, z PLC programu a také pomocí speciálního dialogového okna.

Pod pojmem **fiktivní prostor** (fiktivní souřadnice) rozumíme ten prostor, ve kterém se pracuje. V tomto prostoru probíhá NC program a v něm také pracuje interpolátor systému. Fiktivní prostor je před transformací totožný s původním prostorem, daným referencí a nulovými body stroje. Systém při transformaci zobrazuje souřadnice podle fiktivních souřadnic.

Pod pojmem **reálný prostor** (reálné souřadnice) rozumíme skutečný prostor, ve kterém se obrábí. Výstup z interpolátoru systému je transformován z fiktivních do reálných přírůstků dráhy. Reálné souřadnice systému je možno vidět jen ve speciální obrazovce pro diagnostiku transformace. Softwarové limitní spínače a tabulkové nelineární korekce pracují podle reálných souřadnic systému.

Protože transformace souřadného systému probíhá až na výstupu z interpolátoru, není transformací ovlivněn průběh NC programu včetně posunutí, délkových a poloměrových korekcí a ani dalších přepočtů souřadnic.

Transformace souřadnic se týká všech druhů pohybu v systému, takže platí pro interpolátor, všechny ruční pohyby (AUTMAN), vlečení os, kopírování a také pro polohovací jednotky řízené z PLC programu.

Transformační parametry pro rovinnou transformaci jsou vypočteny ze zadaného posunutí a úhlu natočení. Pro prostorovou transformaci je nutno zadat tři úhly natočení.

Řízení transformace se skládá ze třech hlavních úkonů:

- Nastavení transformačních parametrů
- Aktivace transformace
- Dezaktivace transformace

V PLC programu je umožněno aktivovat a deaktivovat jednotlivé transformace a řídit zálohování parametrů transformace při vypnutí systému. Dále budou rozebrány některé aspekty povolení pohybu transformovaných os, čtení odměřování fiktivních a reálných souřadnic. V PLC programu se normálně nepředpokládá možnost nastavování parametrů transformace, ale jen řízení její aktivace.

23.1.1 Aktivace a deaktivace transformace z PLC programu

V PLC programu jsou zpřístupněny proměnné:

	bit 2: TR RESTORE	bit 1 : TR SET	bit 0: TR ENABLE	význam
1.rovinná transformace				
TRANS_FLAT1_STATUS	r	r	r	stav
TRANS_FLAT1_COMMAND	w	w	-	povely z PLC
TRANS_FLAT1_AXIS	bitová maska os (r w)			definice os
2.rovinná transformace				
TRANS_FLAT2_STATUS	r	r	r	stav
TRANS_FLAT2_COMMAND	w	w	-	povely z PLC
TRANS_FLAT2_AXIS	bitová maska os (r w)			definice os
prostorová transformace				
TRANS_SPACE1_STATUS	r	r	r	stav
TRANS_SPACE1_COMMAND	w	w	-	povely z PLC
TRANS_SPACE1_AXIS	bitová maska os (r w)			definice os

V tabulce jsou políčka u jednotlivých bitů označena písmeny **r** a **w**. Písmeno **r** znamená, že PLC program může stav bitu jenom číst a písmeno **w** znamená, že do bitu může PLC program jenom zapisovat.

Pro práci s jednotlivými bity musí PLC program použít způsoby složitější adresace bitu.

Význam jednotlivých bitů:

Bit:	Význam
TR_ENABLE	Hodnota log.1 znamená, že příslušná transformace je v systému povolena.
TR_SET	Stav nebo příkaz pro aktivaci příslušné transformace. Hodnota log.1 v případě čtení stavu určuje, že transformace je právě aktivní a v případě příkazu určuje, že transformace má být aktivována.
TR_RESTORE	Stav nebo příkaz pro zrušení příslušné transformace. Hodnota log.1 v případě čtení stavu určuje, že transformace je zrušena a byl obnoven původní stav. V případě příkazu určuje, že transformace má být zrušena.

Příklad:

Aktivace a deaktivace 2.rovinné transformace:

```

FL    1, TRANS_FLAT2_COMMAND.TR_SET      ;aktivace 2D(2)

FL    1, TRANS_FLAT2_COMMAND.TR_RESTORE  ;dezaktivace 2D(2)

```

Test, zda je aktivní prostorová transformace:

```

LDR   TRANS_SPACE1_STATUS.TR_SET         ;načtení stavu transformace
JL1   TR_AKTIVNI                          ;skok, když je aktivní

```

23.1.2 Základní bitové signály v transformaci

Bity pro povolení pohybu **PO_OSx** definované v **PB20** transformace souřadnic **neovlivní**. Když například bude programován pohyb v ose Y, ale následkem transformace pojede také osa X, zůstane v rozhraní nahozen jen bit **PO_OSY**.

Bity pro povolení pohybu **PO_OSxPI** definované v **PB20PI** transformace souřadnic **ovlivní**. Když například bude programován pohyb v ose Y, ale následkem transformace pojede také osa X, budou v rozhraní nahozeny bity **PO_OSXPI** a **PO_OSYPI**. Nastaví se požadavek na pohyb pro všechny osy, které pojedou následkem transformace. Požadavek na pohyb osy, která vstupuje do transformace, způsobí požadavky na pohyb všech os, které tvoří prostoru transformace.

Bity **SM_POxPI** definované v **PB20PIS** určují směr skutečného reálného pojezdu a jsou proto zařazenou transformací ovlivněny.

Bity **MPxPI** definované v **BZH08PI** mohou sloužit pro povolování pohybu, ale PLC program by měl ovládat povolení pohybu stejně pro všechny osy, které jsou v transformaci. Když například má PLC program povolit pohyb v ose X, ale osa X je právě v transformaci s osou Y, musí PLC program povolit současně pohyb v ose Y. **Pokud PLC program zakáže povolení pohybu jenom v některých osách v transformaci, systém sám zakáže povolení pohybu ve všech osách, kterých se transformace týká.**

23.1.3 Reálná a fiktivní poloha pro PLC program

PLC program má k dispozici odměřování systému pro zjištění okamžité polohy stroje. Odměřování je součet bufferu polohy **B_POL** a bufferu inkrementu **B_INK**. Oba buffery jsou typu DWORD v doplňkovém kódu a odměřování je v osminách mikronu. Odměřování je platné od najetí do reference. Přístup k jednotlivým osám je relativní.

Toto prvotní odměřování před aktivací transformace se stane fiktivním odměřováním po zařazení transformace. Kromě toho má PLC program k dispozici reálné odměřování v buňkách **TRANS_ODM**, jedná se o 6 buněk typu DWORD v doplňkovém kódu v osminách mikronu.

Skutečná poloha před transformací	
$(B_POL+4.x) + (B_INK+4.x)$	
Fiktivní poloha v transformaci	Reálná poloha v transformaci
$(B_POL+4.x) + (B_INK+4.x)$	$(TRANS_ODM + 4.x)$

V tabulce x je pořadové číslo osy od nuly: $x = 0,1,2,3,4,5$.

Příklad:

Načtení fiktivní polohy pro 2. souřadnici:

```

LOD    DWORD.B_POL+4      ;buffer polohy pro 2. osu
AD     DWORD.B_INK+4      ;buffer inkrementu pro 2. osu

```

Načtení reálné polohy pro 2. souřadnici:

```

LOD    DWORD.TRANS_ODM+4  ;reálné odměřování pro 2.osu

```

23.1.4 Zálohování parametrů transformace

Systémy řady CNC8x9 nemají paměť CMOS pro zálohování dat, ale mohou využít zálohování na HARDISK. PLC program má možnost zálohovat před vypnutím různé paměťové oblasti a způsob zálohování je popsán v návodu k programování PLC v podkapitole: „Vypínání a zapínání systému řady CNC8x9-DUAL“.

Když nastavení parametrů transformace například z dialogového zadání má být platné po delší dobu obrábění, je potřeba zálohovat při vypnutí i parametry transformace. Aby systém při opětovném zapnutí systému obnovil všechny parametry transformace, je potřeba nastavit **sedmou dekádu strojní konstanty R394** na hodnotu 1.

Nastavení 7.dekády R394 na hodnotu 1.

	8	7	6	5	4	.	3	2	1
R394:	x	1	x	x	x	x	x	x	x

V PLC programu musí existovat „zálohovací mechanismus“ spouštěn například od softwarového tlačítka pro vypínání systému (kód v MATTL je 0E7h) a v něm naprogramovaná žádost o zálohování transformace. Žádost se provede nastavením bitu **REQ_BACKUP_TRANS**. Systém pak uloží parametry transformace na disk do souborů \$BKP_TRN.SYS a \$BKP_TRN.BAK v adresáři SYSFILES. Po zapsání na disk systém bit REQ_BACKUP_TRANS vynuluje.

Příklad:

Žádost a zálohování parametrů transformace v mechanismu:

FL	1,REQ_BACKUP_TRANS	;nastavení žádosti
EX		
LDR	REQ_BACKUP_TRANS	;čekání na zapsání na disk
EX1		

Po zapnutí systému se parametry transformace automaticky obnoví, ale neprovede se aktivace transformace. Samotnou aktivaci je možno provést po nájedzu do reference z NC programu, pomocí dialogového okna nebo z PLC programu.

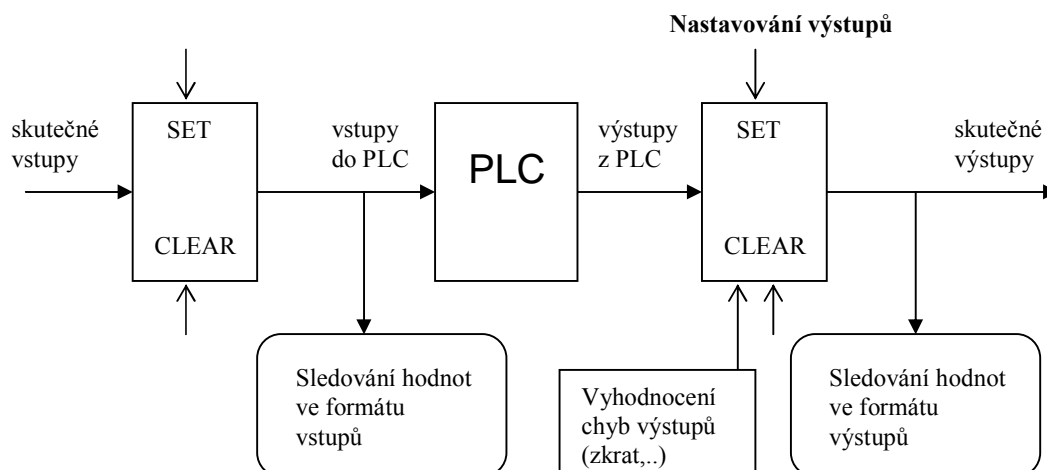
23.2 Ruční ovládání vstupů a výstupů

Možnost nezávislého ručního ovládání binárních vstupů a výstupů přímo z panelu systému.
Platí pro systémy řady CNC8x9 – DUAL od softwarové verze panelu 40.15 a PLC překladače 6.042.

Systémy řady CNC8x9 mají možnost plného ladění a tvorby PLC programu pomocí externího počítače (netbooku) připojeného přes sériový kanál. Softwarové prostředky potřebné pro ladění a tvorbu PLC programu – Wintechnol jsou k dispozici na firemním CD nebo na stránkách: www.mefi.cz. Možnost ručního ovládání vstupů a výstupů je jen doplňková nouzová možnost přímého nastavování vstupů a výstupů přímo z panelu systému bez použití externího počítače.

Ruční řízení výstupů je nezávislé na nastavované hodnotě výstupu z PLC programu a řízení vstupů je nezávislé na fyzické sejmuté hodnotě vstupu. Najednou je možno ručně ovládat libovolný počet vstupů a výstupů.

Blokové schéma pro ruční ovládání vstupů a výstupů:



23.2.1 Aktivace ručního ovládání vstupů a výstupů

Od verze 40.16 je možné použít dva způsoby aktivace ovládání:

- A) Provádí se pomocí tlačítka panelu systému, které aktivuje makro:
3800H povolení / zakázání ručního ovládání vstupů a výstupů

Makro se přiřadí příslušnému tlačítku na panelu systému pomocí souboru **KLAV.KNF** (viz příloha L – tlačítka na panelu systému) a standardně je přiřazeno tlačítku „USER“.

- B) Provádí se automaticky volbou formátů vstupů, výstupů nebo matice panelu stroje pomocí volby indikace – tlačítko **WIN**.


Funkci ručního ovládání možno zablokovat pomocí páté dekády strojní konstanty R89 (**5.R89**):

5.R89 Povolení aktivace ručního ovládání vstupů a výstupů.		
5. dekáda	= 0	Ruční ovládání vstupů a výstupů je zablokováno.
	= 1	Ruční ovládání vstupů a výstupů je povoleno.

Pokud je stisknuto příslušné tlačítko („USER“) a v 5. dekadě strojní konstanty R89 je hodnota 1, systém zobrazí na obrazovce dotaz pro opětovné potvrzení aktivace ručního ovládání.

Po celou dobu aktivace ručního ovládání vstupů a výstupů se na obrazovce systému zobrazuje v horní části červeně značka **I/O**.

Aktivaci ručního ovládání je možno provést kdykoli nezávisle na aktuálních režimech a formátech systému. Po aktivaci se na systému zobrazí speciální menu pro ruční ovládání :

0 nastavení	1 nastavení	x uvolnění	x-all uvolní vše	I/O typ portu	 návrat
-----------------------	-----------------------	----------------------	----------------------------	-------------------------	---

23.2.2 Ruční nastavování vstupů a výstupů

V době, když je na systému aktivní menu pro ruční ovládání vstupů a výstupů, podbarví se v horní části obrazovky místo pro aktuální položku na červenou s nápisem vyvoleného typu portu (OUTPUT, INPUT, MATRIX).

Nastavení se provede v krocích:

1.	Volba typu portu. Typ portu se přepíná 5. softwarovým tlačítkem menu a volí se z možností, které poskytuje aktuální softwarová verze. Ve verzi 40.15 jsou možné volby: OUTPUT nastavování výstupů INPUT nastavování vstupů MATRIX nastavování maticových vstupů
2.	Volba nastavení. Možnosti nastavení se provedou 1. až 3. softwarovým tlačítkem menu. 0 nastavení bitu na hodnotu log.0 1 nastavení bitu na hodnotu log.1 x uvolnění bitu, výstup bude opět ovládán z PLC a vstup bude odpovídat skutečnému vstupu Po volbě nastavení (0,1,x) se v horní části obrazovky pro aktuální položku zobrazí editační část pro zadání portu a váhy bitu: „_._“
3.	Volba portu a váhy bitu. Zadá se dvoudekádové číslo vstupního nebo výstupního portu. Rozsahy zadání jsou (viz „Návod na programování PLC – Řízení binárních vstupů a výstupů“). 01 – 40 vstupní porty 00 – 29 výstupní porty 00 – 15 maticové vstupy Po zadání čísla portu se hned zadává váha bitu v rozsahu 0 – 7. Pro zadávání se používají číslice a tlačítko DEL. Zadání se ukončí tlačítkem ENTER. Tlačítko ENTER způsobí provedení požadované akce.

Ruční ovládání vstupů a výstupů je možno provádět kdykoli nezávisle na aktuálních režimech a formátech systému. Pro opětovné vysvícení speciálního menu pro ruční ovládání se může použít aktivační tlačítko „USER“, ale režim ovládání se při dotazu neodvolá. Po celou dobu aktivace ručního ovládání (i když právě není aktivní menu pro ruční ovládání) se na obrazovce systému zobrazuje v horní části značka **I/O**.

23.2.3 Zobrazování stavu vstupů a výstupů s ohledem na ruční ovládání

Pro zobrazování stavu vstupů a výstupů s ohledem na ruční ovládání slouží standardní formáty pro sledování vstupů, výstupů a matice panelu stroje. Pomocí těchto formátů můžeme získat přehled o všech vstupech a výstupech, které byly přenastaveny pomocí ručního ovládání.

Pokud má zobrazovaný bit políčko podbarveno **zeleně**, je tento bit právě ručně ovládán a jeho hodnota by měla odpovídat našemu ručnímu nastavení. Pokud je bit podbarven zeleně, ale jeho hodnota neodpovídá požadovanému ručnímu nastavení, pravděpodobně není příslušná periferní jednotka nakonfigurovaná (R231). Když se příslušný bit uvolní, zelené podbarvení se musí ztratit.

Pokud má zobrazovaný bit políčko podbarveno **červeně**, vykazuje tento bit právě chybu. Z největší pravděpodobnosti se jedná o chybu zkratu. Pokud chyba nebude potvrzena v příslušném chybovém poli od PLC programu, zůstane bit v chybě až do vypnutí systému.

23.2.4 Dezaktivace ručního ovládání vstupů a výstupů

Dezaktivace režimu ručního ovládání vstupů a výstupů se provádí pomocí stejného tlačítka jako pro aktivaci režimu. Provádí se pomocí tlačítka panelu systému, které aktivuje makro 3800h a standardně je přiřazeno tlačítku „USER“. Po stisku se na obrazovce zobrazí dotaz pro opětovné potvrzení.

Po dezaktivaci ručního ovládání dojde najednou k uvolnění všech nastavených bitů. (Žádný bit nebude podbarven zeleně). V horní části obrazovky se přestane zobrazovat značka **I/O**.

Po ukončení činnosti doporučujeme ruční ovládání vstupů a výstupů zablokovat pomocí strojní konstanty 5.R89: (**5.R89 = 0**).

23.3 Druhá skupina vlečených os

Od verze překladače PLC programu 6.050 je zavedena možnost použít druhou nezávislou skupinu vlečených os. Jedná se o skupinu plně ovládanou z PLC programu.

Pro PLC jsou zpřístupněny struktury pro ovládání vlečení pro každou souřadnici, které mají tvar:

```

;*** Struktura pro řízení vlečení z PLC ***
TRAILS          struc          ;delka 4 BYTE
    TRAIL_ACTIV      DB          0      ;aktivacni a modifikacni signal
    TRAIL_MASTER      DB          0      ;ridici osa 1,2,3,...
    TRAIL_MODIF       DW          0      ;modifikace (rezerva)
TRAILS          ends

```

Implementace (umístění) struktur pro jednotlivé souřadnice je:

	Aktivační a modifikační signál	Definice řídící osy	Offset
Název	TRAIL_ACTIV (1 Byte, offset +0)	TRAIL_MASTER (1 Byte, offset +1)	
TRAIL_X	0, 1, 2	1, 2, 3, 4, 5, 6	+0
TRAIL_Y	0, 1, 2	1, 2, 3, 4, 5, 6	+4
TRAIL_Z	0, 1, 2	1, 2, 3, 4, 5, 6	+8
TRAIL_4	0, 1, 2	1, 2, 3, 4, 5, 6	+12
TRAIL_5	0, 1, 2	1, 2, 3, 4, 5, 6	+16
TRAIL_6	0, 1, 2	1, 2, 3, 4, 5, 6	+20

23.3.1 Aktivační a modifikační signál TRAIL_ACTIV

Prvek struktury o velikosti 1 Byte slouží pro aktivaci vlečení a určuje také způsob vlečení. Může nabývat hodnot:

0	Vlečení není povoleno
1	Vlečení je aktivní pro danou ve stejném směru
2	Vlečení je aktivní pro danou v obráceném směru (zrcadlově)

23.3.2 Definice řídící osy TRAIL_MASTER

Prvek struktury o velikosti 1 Byte slouží pro definici řídící osy. Hodnota by měla být nastavena před aktivací vlečení a může být nastavena trvale i když je vlečení zakázáno. Může nabývat hodnot:

0	Vlečení nemá definovanou řídící osu a neprovede se.
1	Řídící osa pro vlečení je 1. osa (dle strojní konstanty R00, například X)
2	Řídící osa pro vlečení je 2. osa
3	Řídící osa pro vlečení je 3. osa
4	Řídící osa pro vlečení je 4. osa
5	Řídící osa pro vlečení je 5. osa
6	Řídící osa pro vlečení je 6. osa

Pro vlečené osy se nastavují bity PO_OSxPI. Pohyb vlečených os je blokován signály MPxPI.

Příklady:

Nastavení vlečení, kde osa Y bude vléct také 4.osu: Y -> 4.

```

LOD   CNST.2                ;řídící osa Y
STO   BYTE.TRAIL_4.TRAIL_MASTER ;nastaví řídící osu pro 4.
LOD   CNST.1                ;přímé vlečení
STO   BYTE.TRAIL_4.TRAIL_ACTIV  ;aktivace vlečení Y->4

```

Nastavení vlečení, kde osa Y bude vléct 4. a 6. osu: Y -> 4, 6

```

LOD   CNST.2                ;řídící osa 2. (Y)
STO   BYTE.TRAIL_4.TRAIL_MASTER ;nastaví řídící osu pro 4.
STO   BYTE.TRAIL_6.TRAIL_MASTER ;nastaví řídící osu pro 6.
LOD   CNST.1                ;přímé vlečení
STO   BYTE.TRAIL_4.TRAIL_ACTIV  ;aktivace vlečení Y->4
STO   BYTE.TRAIL_6.TRAIL_ACTIV  ;aktivace vlečení Y->6

```

Zrušení vlečení pro 4. a 6.osu:

```

LOD   CNST.0                ;zrušení vlečení
STO   BYTE.TRAIL_4.TRAIL_ACTIV ;zrušení vlečení pro 4.
STO   BYTE.TRAIL_6.TRAIL_ACTIV ;zrušení vlečení pro 6.

```

23.4 Řízení procenta S z PLC programu

Od verze panelu 40.18 DUAL a PLC překladače 6.053 je umožněno ovládat procenta otáček S z PLC programu.

Aktivace ovládání procenta S z PLC programu se provede nastavením bitu **SPEED_OVR** definovaném v BZH_ENH1. Žádaná hodnota pro procenta otáček se zadává pomocí bajtové buňky **SPEED_OVR_EXT**, ve které možno nastavit plynule hodnotu od 0-255.

Příklad:

```

BUN_PROC_S: DS      1          ;buňka pro žádanou hodnotu proc.S

LOD   BUN_PROC_S          ;žádaná hodnota pro procenta S
STO   SPEED_OVR_EXT        ;nastavení žádané hodnoty
FL    1,SPEED_OVR         ;aktivace ovládání procenta S z PLC

```

23.5 Čas a datum pro PLC program

V PLC programu jsou zpřístupněny struktury pro zjišťování času a datumu (požité formáty proměnných jsou standardní například pro MS-DOS):

```
TIME_INFOS  struc
    TIME_MIN  DB      0      ;Minuty
    TIME_HOUR DB      0      ;Hodiny
    TIME_HUND DB      0      ;Setiny sekundy
    TIME_SEC  DB      0      ;Sekundy
TIME_INFOS  ends

DATE_INFOS   struc
    DATE_YEAR DW      0      ;Rok
    DATE_DAY  DB      0      ;Den
    DATE_MON  DB      0      ;Mesic
DATE_INFOS   ends
```

Příklad:

```
LOD  BYTE.SYST_TIME_INFO.TIME_MIN      ;načte minuty
LOD  BYTE.SYST_TIME_INFO.TIME_HOUR     ;načte hodiny
LOD  BYTE.SYST_TIME_INFO.TIME_SEC      ;načte sekundy
LOD  WORD.SYST_DATE_INFO.DATE_YEAR     ;načte rok
LOD  BYTE.SYST_DATE_INFO.DATE_DAY      ;načte den
LOD  BYTE.SYST_DATE_INFO.DATE_MON      ;načte měsíc
```

23.6 Spánkový režim systému a spuštění MS-Windows98

Ve **spánkovém režimu** systému pracuje jen sekundární procesor a v něm je aktivní jen odměřování souřadnic. PLC program je zastaven. Primární procesor je tak uvolněn pro jiné aplikace MS-DOS a také je umožněn přechod do MS-Windows98. Při opětovném startu se startuje jen samotný primární procesor (software panelu), přičemž dojde k **režimu probuzení**. Na systému zůstanou platné míry souřadnic a reference os.

23.6.1 Nastavení spánkového režimu

Na systému je nutno nastavit příslušnou dekádu strojní konstanty **R241** na hodnotu **2**, která slouží pro spouštění externích programů se spánkovým režimem.

Pokud potřebujeme používat MS-Windows98, je nutno nastavit:	
1.	<p>Když systém má k dispozici 128 Mbytů dynamické paměti, umístíme pracovní oblast pro sekundární procesor na adresu 7000000h (cca 117 Mbyte).</p> <p>V souboru C:\PLC\SYSTEM\SYSTECH.KNF změníme hodnotu u klíčového slova AddrSecBase:</p> <p>AddrSecBase = 7000000h</p>
2.	<p>Omezíme paměť systému Windows, aby nezasáhl do pracovní oblasti sekundárního procesoru.</p> <p>Zavedeme nové klíčové slovo MaxPhysPage v souboru C:\WINDOWS\SYSTEM.INI v sekci [386Enh] a nastaví se na 6000h po 1000h bajtech (6000000h).</p> <p>[386Enh] MaxPhysPage=6000</p>
3.	<p>V souboru autoexec.bat se spouští jen sekundární procesor:</p> <pre>cd c:\plc\system call sec.bat if errorlevel 1 goto konec</pre>
4.	<p>Ve Win98 se například vytvoří ikona na ploše pro spouštění panelu systému (probouzení systému). Poklepáním na ikonu se musí spustit dávka pro zavedení DLL knihoven a spuštění panelu systému. Ve vlastnostech spuštění se musí nastavit : režim DOS a použít aktuální konfiguraci</p> <pre>:cykl cd c:\syst call dll.bat call panel31.exe if errorlevel 101 goto konec call c:\ext_prog.bat goto %nav% :konec c:\smonitor c</pre>

23.7 Řešení pro vysekávací lisy

Od verze panelu 40.19 DUAL a PLC překladače 6.201 je systémově zavedena podpora pro vysekávací a niblovací lisy. Pod pojmem **vysekávání** rozumíme způsob, kdy systém popojíždí po stejných úsecích dráhy a zastavuje pro vyseknutí díry. Rozpočítání dráhy na správné úseky vykoná systém automaticky. Při vysekávání se po každém ražení díry vždy znovu spíná spojka pro razící beran a po vyražení díry se spojka vypíná. Při **niblování** je proces děrování podobný, ale spojka beranu je trvale sepnuta. Proto niblování děruje materiál s větší frekvencí.

23.7.1 Programování pro vysekávací lisy

Pro řízení vysekávání a niblování je rezervována 14. skupina M funkcí. Všechny M funkce jsou kontinuální:

Funkce 14.skupiny	Musí být programováno v bloku	Programováno předem – kontinuální funkce	Význam
M20			Základní stav. Systém nerozpočítává dráhu na úseky pro děrování. (Měla by být zařazena do prioritním bloku.)
M22, M24		E	Niblování. PLC program natrvalo sepne spojku pro razící beran. Systém rozpočítává dráhu na úseky dráhy, kterých velikost se zadává pod funkcí E [Exxxxx.xxx (mm)]. Funkce E je v tomto případě kontinuální a může se programovat jen na začátku programu. Rychlost pro přejíždění systém použije pro M22 ze strojní konstanty R14 a pro M24 ze strojní konstanty R15 (rychloposuv pro 5. a 6. souřadnici). Zrychlení pro niblování se použije ze strojní konstanty R237 (zrychlení pro rychloposuv). Úseky jsou přitom zoptimalizovány tak, aby celá dráha bloku byla vždy rovnoměrně rozdělena. Povolení jetí pro další úseky dráhy se vykonává ve spolupráci s PLC programem.
M25, M26, M27		F	Vysekávání děr jen na konci bloku. Systém v tomto případě vykoná normální pohyb bloku a PLC program vysekne díru v závěrečných funkcích bloku.
M25, M26, M27	E		Vysekávání děr se zadáním úseku. Systém rozpočítává dráhu na úseky dráhy, kterých velikost se zadává pod funkcí E (podobně jako u niblování, ale funkce E musí být programována v každém bloku). Rychlost pro přejíždění systém použije standardně z posledně programované funkce F . Zrychlení pro vysekávání se použije ze strojní konstanty R52 .
M25, M26, M27	P		Vysekávání děr se zadáním počtu. Systém rozpočítává dráhu na úseky dráhy, kterých počet se zadává pod funkcí P (funkce P musí být programována v každém bloku). Rychlost pro přejíždění systém použije standardně z posledně programované funkce F .

Příklad:

Příklad pro niblování po úsecích 10mm.

```

N20 X100 M22 E10.000      "začátek niblování
N30 Y100
N40 X0 Y0
N50 X100 Y0 G2 I50 J0
N60 X0 Y0 G1 M20          "odvolání niblování
N70 M30

```

23.7.2 Nastavení pro vysekávací lisy

Pro správnou funkci vysekávacího lisu je nutno navíc nastavit:

parametr	hodnota	Význam
5.dekáda R95	1	Speciální systémové úpravy pro vysekávací lisy (rozpočítávání dráhy,...)
6.dekáda R95	1	Povolení spouštění rychlého modulu PLC programu PIS_FAST v rastru 1ms.
R14	xx.xxx	Rychlost pro niblování M22. Musí být menší než rychloposuv v konstantách R10 a R11.
R15	xx.xxx	Rychlost pro niblování M24. Musí být menší než rychloposuv v konstantách R10 a R11.
R52	xx.xxx	Zrychlení pro základní stav a pro vysekávání M25, M26 a M27.
R237	xx.xxx	Zrychlení pro niblování M22 a M24 (max. 30.000)

23.7.3 PLC program pro vysekávací lisy

Způsob řešení vysekávacích lisů si vyžaduje spolupráci systému a PLC programu při řízení pohybu. Princip řízení úseků dráhy (niblů) je založen na bitech z rozhraní pro povolení pohybu **MPX**, **MPY**,...v **BZH08**. (Pro případ, že by byl nastaven jiný rastr pro interpolátor, než pro řízení servosmyček, je vhodné nastavovat také bity v **BZH08RTM**.)

Řízení úseků dráhy probíhá ve cyklicky dvou krocích:

1.	<p>Systém odjede odpovídající úsek dráhy, sám zastaví příslušnou rampou a vynuluje povolení pohybu:</p> <p style="margin-left: 40px;">BZH08 ← 0 BZH08RTM ← 0</p> <p>Po ukončení úseku dráhy se nastaví signál INPOS na hodnotu 1.</p>
2.	<p>PLC program testuje v rychlé smyčce PLC programu signál INPOS a na základě jeho hodnoty provede vyražení díry. Potom sám povolí pohyb pro další úsek pomocí buněk BZH08 a BZH08RTM, například:</p> <p style="margin-left: 40px;">BZH08 ← 03 BZH08RTM ← 03</p>

Vzorový PLC program je na firemním CD (I_O_TRUM.PLC }.

Příklad:

Mechanismus pro děrování v rychlé smyčce PLC programu

```
MECH_BEGIN   DEROVANI

;CEKAME NA DOJETI SOURADNIC
DERO_NOR:
    LDR      INPOS
    EX0

    ;TEST PODMINEK PRO VYRAZENI DIRY
    ;JE KONEC BLOKU ?
    ;JE STOP BLOKU ?
    ;....

;VYRAZENI DIRY
    FL      1,JEDNA_DIRA    ;MECHANIZMUS PRO DIRU
    EX
    LDR      JEDNA_DIRA
    EX1

;POVOLENÍ POHYBU - DALŠÍ KROK
    CLI
    FL      1,MPX,MPY
    LOD      BZH08
    STO      BZH08RTM
    STI
    EX                      ;PRODLEVA 1 ms

;KONTROLA PREJEZDU NA DALŠÍ KROK A SPUSTENÍ DALŠÍHO CYKLU
    LDR      INPOS
    EX1
    JUM      DERO_NOR      ;OPAKOVANI

MECH_END     DEROVANI
```

23.8 Paměťový osciloskop

Od softwarové verze panelu 40.20 a PLC překladače 6.202 je zpřístupněna možnost použití paměťového osciloskopu. Paměťový osciloskop má možnost sledování maximálně 12 volitelných průběhů a nastavování libovolné podmínky pro trigger.

Sledování grafů je rozděleno do 6 kanálů, ve kterých se vždy zobrazují 2 průběhy. Sledované průběhy mohou být implicitní, to je sledování difference a složkové rychlosti pro jednotlivé osy, nebo se mohou zvolit obsluhou. Volba sledovaných průběhů se může provést výběrem ze standardních prvků nebo přímou volbou zadáním adresy sledovaného prvku.

Pro paměťový osciloskop je nutno nastavit podmínku triggeru. Podmínka se může nastavit jako jednoduchá nebo složená ze dvou podmínek svázaných logickou operací AND a OR. Podmínka může být bitová, kde se zadává test na příchod log.1 nebo log.0, nebo datová relace rovnosti, větší a menší. Proměnná, která vstupuje do podmínky nastavení triggeru, se může provést výběrem ze standardních prvků nebo přímou volbou zadáním adresy sledovaného prvku.

Při příchodu triggeru je zapamatováno vždy všech 12 průběhů, které je možno potom libovolně prohlížet, měnit mírku a znovu kalibrovat. Nejmenější mírka je 25ms/dílek {jeden bod odpovídá 1 ms}.

Pro nastavování parametrů osciloskopu slouží řídicí soubor TABOSCIL.PAR, který se edituje společně s tabulkami parametrů systému. Osciloskop převezme nově nastavené hodnoty po ukončení edice souboru nebo po spuštění syntaktické kontroly řídicího souboru.

Spouštění syntaktické kontroly a převzetí nových parametrů osciloskopu, povolení triggeru a zrychlená volba formátu pro osciloskop se dá provádět i pomocí tlačítek panelu, kterým jsou přiřazena speciální makra.

23.8.1 Řídicí soubor osciloskopu

Aktivní část řídicího souboru osciloskopu **TABOSCIL.PAR** začíná klíčovým slovem **\$OSC** a dále pokračuje seznamem nastavovatelných parametrů.

Od parametru **0** do parametru **13** je oblast pro nastavování triggeru osciloskopu

```
$OSC
00: 0          "Trigger request 0=disable, 1=enable
01: 0.150      "Posttrigger (150)
02: 1          "Trigger AUTO
03: 10         "Trigger relation 1..bit=1 2..bit=0
               " 10..GE 11..LT 12..EQ
04: 0          "Trigger standard address
05: 0          "Trigger weigth bit..(1,2,..80) 0=WORD 09=BYTE 99=DWORD
06: 0.000      "Trigger offset 1-5..offset 6-8..selector
07: 0.000      "Trigger data
08: 0          "Triggers condition 0=2.trigger neni, 1=AND, 2=OR
09: 0          "2.Trigger relation
10: 0          "2.Trigger standard address
11: 0          "2.Trigger weigth
12: 0.000      "2.Trigger offset
13: 0.000      "2.Trigger data
```

R	hodnota	název	význam
00	0 1	Trigger request	- Trigger pro osciloskop není požadován (0) - Trigger pro osciloskop je požadován (1)
01	0.150	Posttrigger	Počet ms, pro které se mají zaznamenat hodnoty ještě po příchodu triggeru.
02	0 1	Trigger AUTO	- Je použit jen manuální trigger (0). - Je použit trigger podle podmínky (1).
03	1; 2 10;11;12	Trigger relation	-Podmínka triggeru je: bit=1 (1), nebo bit=0 (2) -Podmínka triggeru je: obsah buňky >= data (GE..10) obsah buňky < data (LT..11) obsah buňky = data (EQ..12)
04	0 abcd	Trigger standard address	-Adresa buňky pro trigger je zadána přímo podle mapy (0). -Adresa buňky pro trigger je vybrána z předdefinovaných vzorů (abcd), viz tabulka dále.
05	1;2;4.. 0 9 99	Trigger weight	-Buňka triggeru je bit s váhou 1,2,4,8,10,20,40 nebo 80. -Buňka triggeru je hodnota WORD (0). -Buňka triggeru je hodnota BYTE (9). -Buňka triggeru je hodnota DWORD (99).
06	xxx.xxx	Trigger offset	Ručně zadaná adresa buňky pomocí mapy CNC_PIS.MAP offsetem v 1. až 5. dekádě a selektorem v 6. až 8. dekádě. Tabulka selektorů je uvedena dále. Hodnota se musí zadat s desetinnou tečkou dekadicky.
07	xxx.xxx	Trigger data	Data, které vstupují do podmínky triggeru. Data se musí zadat s desetinnou tečkou dekadicky.
08	0 1 2	Triggers condition	2.podmínka triggeru není použita (0). 2.podmínka triggeru svázaná operací AND (1). 2.podmínka triggeru svázaná operací OR (2).

Nastavení parametrů 09 až 13 pro druhou podmínku triggeru je podobné jako pro nastavení první podmínky.

Nastavení parametrů pro první kanál – dva grafy:

```

20: 0      "Channel1 - graph1 request    (0,1)
21: 0      "Channel1 - graph1 weight    bit..(1,2,..80)  0=WORD  09=BYTE
22: 0      "Channel1 - graph1 standard
23: 0.000  "Channel1 - graph1 offset    1-5..offset  6-8..selector
25: 0      "Channel1 - graph2 request    (0,1)
26: 0      "Channel1 - graph2 weight    bit..(1,2,..80)  0=WORD  09=BYTE
27: 0      "Channel1 - graph2 standard
28: 0.000  "Channel1 - graph2 offset    1-5..offset  6-8..selector

```

R	hodnota	název	význam
20	0 1	Channel1 graph1 request	- 1.průběh je implicitní – obsah diferenčního čítače (0). - 1.průběh je zvolen uživatelem (1).
21	1;2;4.. 0 9	Channel1 graph1 weight	-Buňka pro 1.průběh je bit s váhou 1,2,4,8,10,20,40 nebo 80. -Buňka pro 1.průběh je hodnota WORD (0). -Buňka pro 1.průběh je hodnota BYTE (9).
22	0 abcd	Channel1 graph1 standard	-Adresa buňky pro 1.graf je zadána přímo podle mapy (0). -Adresa buňky pro 1.graf je vybrána z předdefinovaných vzorů (abcd), viz tabulka dále.
23	xxx.xxx	Channel1 graph1 offset	Ručně zadaná adresa buňky pomocí mapy CNC_PIS.MAP offsetem v 1. až 5. dekádě a selektorem v 6. až 8. dekádě. Tabulka selektorů je uvedena dále. Hodnota se musí zadat s desetinnou tečkou.

Význam jednotlivých parametrů se opakuje pro 2.průběh a také pro všechny další kanály a je zřejmý z komentářů v řídicím souboru TABOSCIL.PAR.

Předdefinované vzory (standard address) jsou čtyřciferné kódy, kde první dvě cifry jsou přiřazeny určité paměťové oblasti (poli) a druhé dvě cifry určují žádaný prvek. Tabulka předdefinovaných vzorů (ve verzi 6.202):

Předdefinované adresy (standard address)		
0101 - 0140	vstupy PLC: P1IN - P40IN
0200 - 0229	výstupy PLC: P0OUT - P29OUT
0300 - 0315	maticové vstupy: PANEL STROJE
0400 - 0405	odměřování os X,Y,Z,4,5,6
0500 - 0505	diference os X,Y,Z,4,5,6
0600 - 0605	složkové rychlosti os X,Y,Z,4,5,6
0700 - 0701	prostorová rychlost a její derivace

Převzetí nových parametrů pro osciloskop se provede pro syntaktické kontrole řídicího souboru. Syntaktická kontrola se spustí buď standardně v tabulkových režimech parametrů, výběrem souboru TABOSCIL.PAR a stisknutím tlačítka ENTER, nebo pomocí tlačítka panelu systému, kterému pomocí definičního souboru klávesnice KLAV.KNF přiřadíme makro **6500h**.

Možné použití maker pro přiřazení tlačítkům panelu v KLAV.KNF pro řízení osciloskopu	
6300h aktivace a deaktivace triggeru pro osciloskop
6400h zrychlena volba formátu osciloskopu
6500h syntaktická analýza řídicího souboru pro osciloskop

Makro **6300h** pro aktivaci a deaktivaci triggeru řídí, zda je trigger požadován nebo není. Makro má stejný účinek jako nastavení parametru **“00: Trigger request“**, jen jej může měnit dynamicky rychlejším způsobem (nejedná se o odstartování triggeru, to se provede softwarovým tlačítkem).

Makro **6400h** jen urychluje volbu formátu pro osciloskop (není potřeba použít volbu indikace - WIN).

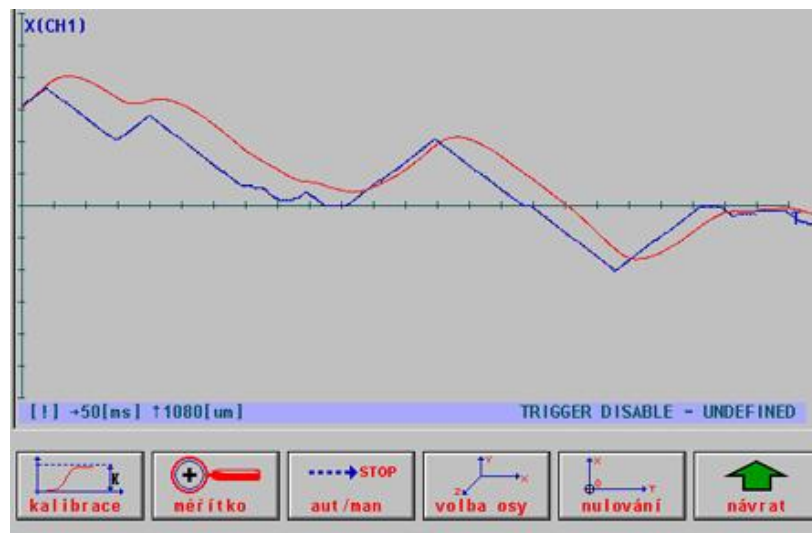
Při zadávání požadované adresy prvku, je potřeba použít mapu CNC_PIS.MAP, která vznikne po překladu PLC programu. Adresa je zadávána dekadicky formou offsetu a selektoru (jedná se o chráněný režim sekundárního procesoru, selektor je pointer do tabulky deskriptorů segmentů). Dále uvádíme tabulku selektorů pro nejpoužívanější oblasti proměnných:

Selektor (dec)	Název segmentu	Popis
048	DATA_COM	Datový komunikační segment (spolupráce primárního a sekundárního procesoru), vstupy, výstupy, oblasti TECHNOL, LABEL_MEM apod. (definováno v DCOM.ASH).
104	DATA1	Datový segment pro systémové proměnné a pro rozhraní PLC-systém. Základní datová oblast PLC (DATA ... globální data).
160	DATA_USR	Pomocný datový segment, většinou pro systémové proměnné.
360	DATA_PLC	Datový segment pro lokální a automatické proměnné PLC programu (DATA_LOCAL).
320	_TEXT32	32 bitový datový segment pro FLAT model (Wintechnol apod.).

23.8.2 Ovládání osciloskopu

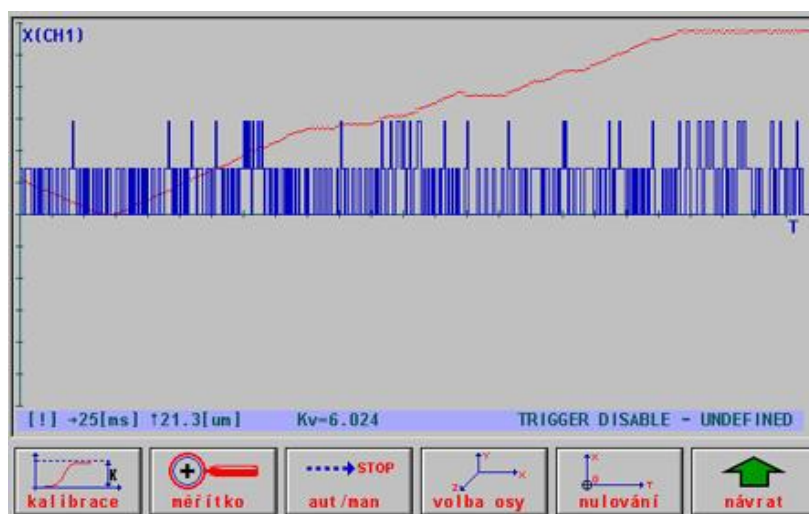
Pro ovládání osciloskopu slouží softwarová tlačítka menu, které se modifikuje v závislosti na tom, zda je trigger povolen nebo zakázán:

Příklady osciloskopu bez nastavení triggeru. Softwarová tlačítka slouží pro kalibraci, nastavení mírky, přepínání automatického a ručního nulování obrazovky, volbu kanálů a ruční nulování obrazovky. Kalibraci možno použít automatickou nebo ruční zadáním hodnoty.



Příklad sledování průběhu bez nastavení triggeru. V řídicím souboru je nastaven jeden průběh na sledování prostorové rychlosti (700) a druhý průběh je ponechán na implicitní nastavení pro sledování diferenčního čítače.

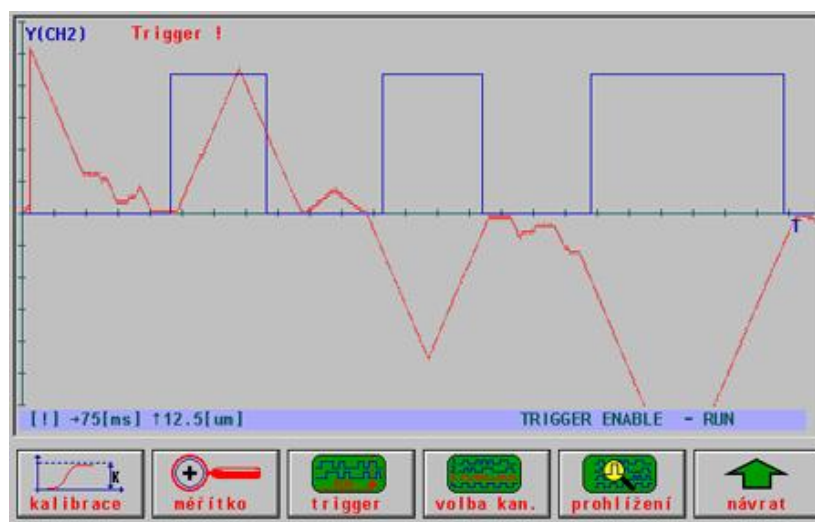
V pravé části informační lišty se zobrazuje nápis „TRIGGER DISABLE“, což znamená, že trigger nebyl aktivován. Aktivace se provede spuštěním syntaktické kontroly řídicího souboru TABOSCIL.PAR, kde je nastaven parametr “00: Trigger request“ na hodnotu 1 nebo pomocí tlačítka panelu systému, které aktivuje makro 6300h.



Příklad sledování průběhu bez nastavení triggeru. V řídicím souboru je nastaven jeden průběh na sledování prostorové rychlosti (700) a druhý průběh na sledování změny prostorové rychlosti (701).

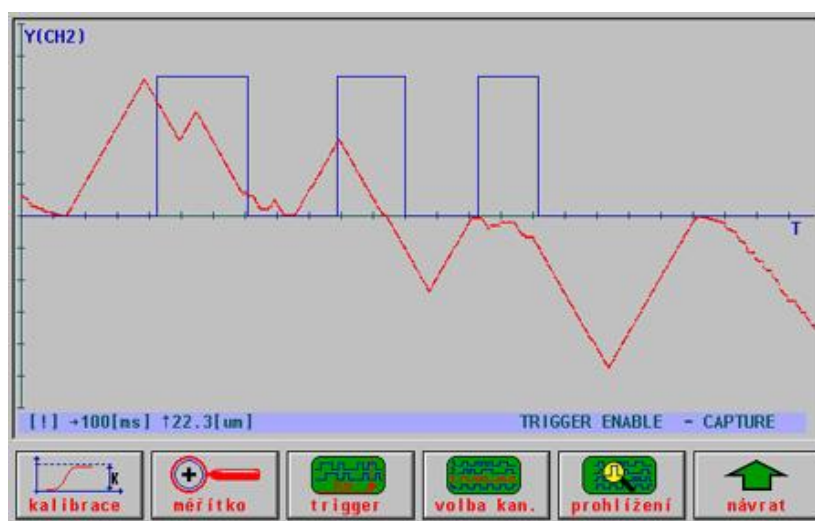
Po aktivaci triggeru se změní menu osciloskopu. První dvě tlačítka jsou stejná jako v předešlém případě. Trigger je povolen, což se hlásí v informační liště nápisem „TRIGGER ENABLE“. Samotné spuštění triggeru se provede třetím softwarovým tlačítkem „TRIGGER“. Po jeho zmáčknutí se v informační liště zobrazí nápis „RUN“ a v horní části obrazovky se zobrazí červený nápis „Trigger !“. Zobrazovaný průběh v tomto stavu je neplatný, je to poslední prohlížený stav z předchozího sejmутí.

Systém čeká na vyhodnocení podmínky triggeru. Pokud chceme použít manuální trigger, čekání přerušíme opětovným stiskem třetího tlačítka „TRIGGER“.



Trigger je povolen a současně spuštěn. Stav čekání na vyhodnocení podmínky triggeru. Čekání možno přerušit opětovným stiskem tlačítka TRIGGER.

Po příchodu triggeru se na obrazovce zobrazí sejmутý průběh. Informační nápis ve stavové liště osciloskopu se změní na „CAPTURE“. Sejmутý průběh je možno kalibrovat, měnit mírku a opětovně prohlížet pomocí pátého softwarového tlačítka „PROHLÍŽENÍ“. Současně je možno přepínat kanály čtvrtým softwarovým tlačítkem a sledovat sejmутou událost ve všech kanálech osciloskopu.



Trigger je povolen a právě byl dosažen splnění podmínky triggeru. Sejmутé snímky ve všech kanálech je možno opětovně prohlížet, kalibrovat a měnit jejich mírku.

23.9 Režim učení „TEACH-IN“

Režim učení (TEACH-IN) platí pro systémy řady CNC8x9 od verze panelu 40.23, 7.4.2003.

Jedná se o možnost tvorby programu podle ručních pojezdů, například v pomocných ručních pojezdech nebo v režimu RUP.

23.9.1 NC program pro režim TEACH-IN

NC program pro režim TEACH-IN může být libovolný program, který obsahuje řídicí hlavičku partprogramů (viz. Příloha M), která pro tento režim musí obsahovat klíčová slova **IGT(par)** a **TIN(par)**. Pomocí tlačítka panelu „SET TEACH-IN“ (popsáno dále) může být NC program pro TEACH-IN automaticky vytvořen.

V klíčovém slově **IGT(par)** parametr *par* je číselný a určuje číslo bloku, od kterého se budou automaticky tvořit bloky při příštím zápisu v režimu TEACH-IN. Číslo bloků se zvětšují po 10. Při zápisu nového bloku v režimu TEACH-IN se vytvoří blok podle parametru (par) a hodnota v klíčovém slově IGT se zvětší o 10. (Pro hledání řádku pro vložení nového bloku musí existovat blok s číslem bloku o 10 menší než je uvedeno v klíčovém slově IGT.)

V klíčovém slově **TIN(par)** parametr *par* je číselný, může obsahovat číslice 1,2,3,4,5,6, které určují pořadové číslo souřadnic, kterých se režim TEACH-IN týká. Pořadí souřadnic je stejné, jako pořadí definice os ve strojních konstantách, nebo pořadí os na obrazovce systému.

Příklad:

```
Bloky se budou tvořit od čísla bloku N5010
Režim TEACH-IN se týká souřadnic 1. 2. a 4.
{
  TIN(124) IGT(5010)
}
%1
N5000 G1 F1000 G90      "Hlavní věta programu
N999999 M30
*
```

23.9.2 Ovládací tlačítka pro režim TEACH-IN

Pro režim TEACH-IN je potřeba na panelu systému zvolit dvě tlačítka.

První tlačítko „SET TEACH-IN“ slouží pro volbu režimu TEACH-IN, volby nebo vytvoření souboru pro TEACH-IN.

Druhé tlačítko „WRITE TEACH-IN“ slouží pro zápis nového bloku.

Volba tlačítek na panelu systému se provádí pomocí přiřazení maker.

Makra se přiřadí příslušnému fyzickému tlačítku v souboru KLAV.KNF. (viz příloha L – tlačítka na panelu systému).

makro	tlačítko	Popis
6D00H	„SET TEACH-IN“	<p>Volba režimu učení. Stisk tlačítka může mít dva různé projevy:</p> <ol style="list-style-type: none"> 1. Pokud je toto tlačítko stisknuto v době, kdy na obrazovce je vykreslen přehled programů (formát DIR po stisku tlačítka PROGRAMY pro práci s pamětí v hlavním menu), označí se vyvolený soubor podle místa kursoru pro režim TEACH-IN. Označení je potvrzeno krátkou zprávou „Provedena volba souboru pro režim TEACH-IN“. Všechny zápisy bloků budou směřovat do tohoto souboru. Předpokládá se, že takto vyvolený soubor obsahuje klíčová slova IGT(..) a TIN(..). 2. Pokud je toto tlačítko stisknuto v jiné době, než je na obrazovce vykreslen formát DIR přehledu programů, bude po dotazu a po zadání jména souboru tento soubor automaticky vytvořen. Po vytvoření se zobrazí potvrzující zpráva o provedení akce. Vytvoření souboru se provede prostým přepokopírováním dat ze vzorového prioritního programu pro TEACH-IN, který má název TEACHCNF.NCP. Tento prioritní program pro TEACH-IN je možno editovat a je vhodné do 1. bloku napsat hlavní větu pro konkrétní stroj (rychlost, otáčky vřetene apod.)
6E00H	„WRITE TEACH-IN“	<p>Zápis nového bloku. Po stisku tlačítka se vytvoří nový blok ve vyvoleném nebo vytvořeném souboru. Číslo bloku se určí podle klíčového slova IGT(..) a zapíšou se souřadnice podle klíčového slova TIN(..). Zapsání nového bloku je potvrzeno krátkou zprávou „Proveden zápis bloku v režimu TEACH-IN“. Systém bloky ukládá jen v lineární interpolaci G1, kterou možno podle potřeby dodatečně změnit na kruhovou interpolaci (změní se G1 na G2,G3 a doplní se rádius R).</p>

23.9.3 Práce v režimu TEACH-IN

Práce v režimu TEACH-IN se skládá ze dvou kroků:

1.	Tvorba nového nebo označení existujícího NC programu pro režim TEACH-IN pomocí tlačítka „SET TEACH-IN“.
2.	Zápis nových bloků pomocí tlačítka „WRITE TEACH-IN“

Pro zápis nových bloků v režimu TEACH-IN je potřeba používat stejné posunutí počátku, jaké bude použito ve vyvoleném programu.

Prioritní soubor pro režim TEACH-IN

Pro každý stroj je potřeba při prvním použití nastavit prioritní soubor **TEACHCNF.NCP**:

```
TEACH-IN
{
  TIN(123) IGT(5010)
}
%1
N5000 G1 F1000 G90      "Hlavní veta
N999999 M30
*
```

23.10 Připojení sériových točíttek

23.10.1 Připojení jednoho sériového točítka s ovládáním z PLC

Platí pro softwarovou verzi sekundárního procesoru 6.209 a panelu 40.22.

Kromě bitů pro externí řízení pomocných ručních pojezdů popsanych v PLC návodu v kapitole „Pomocné ruční pojezdy“, je navíc definováno:

Bit: EXPAN_TOC_DIS

Nastavením bitu EXPAN_TOC_DIS na hodnotu log.1 se zablokuje ovládání ručních pojezdů z externího panýlku s točítkem. Bit je přednastaven na hodnotu log.0.

Bit: NCPAN_TOC_DIS

Nastavením bitu NCPAN_TOC_DIS na hodnotu log.1 se zablokuje ovládání ručních pojezdů z panelu systému. Bit nemá význam nastavovat v případě, že je aktivní externí řízení ručních pojezdů. Bit je přednastaven na hodnotu log.0.

Bit: PRESET_TOC_CLR

Nastavením bitu PRESET_TOC_CLR na hodnotu log.1 se zruší předvolba pro použití točítka. Po zrušení předvolby systém bit sám vrátí do hodnoty log.0.

Byte: PTLTOC

PLC program získá bitový obraz stisků tlačítek panýlku v BYTEové buňce PTLTOC, nebo PTLTOC_P1 atd. (viz dále)

Bitový obraz stisků tlačítek:

symbol tlačítka X bit: 3, váha 08h	symbol tlačítka Y bit: 2, váha 04h	symbol tlačítka Z bit: 1, váha 02h	symbol tlačítka X bit: 0, váha 01h
symbol tlačítka 5 bit: 7, váha 80h	symbol tlačítka + bit: 6, váha 40h	symbol tlačítka - bit: 5, váha 20h	symbol tlačítka ~ bit: 4, váha 10h

Při externím řízení pohybu v pomocných ručních pojezdech možno například použít:

Příklady:

;Aktivace externího řízení:

```

      FL    1,TOC_ENABLE           ;Odblokování točitek
      FL    1,REQ_EXT_CONT_AUTMAN ;Pozadavek na ext.řízení pohybu
;?      FL    1,AUTMAN_REQ         ;Pozadavek na aktivaci ruc.pojezdu

```

;Volba osy v mechanismu:

```

      FL    1,VOLBA_TOC_BUSY       ;priznak rozpracovanosti
      FL    1,REQ_EXT_SELECT_AUTMAN ;zadost o predvolbu
      TIM    -,K2                  ;prodleva
      FL    1,EXM_X0
      TIM    -,K2
      FL    0,EXM_X0
      TIM    -,K2
      FL    0,REQ_EXT_SELECT_AUTMAN
      TIM    -,K2
      FL    0,VOLBA_TOC_BUSY

```

23.10.2 Připojení dvou sériových točítok s ovládáním z PLC

Od softwarové verze sekundárního procesoru 6.206 je umožněno připojení dvou externích panílků s ručním ovládáním a točítkem. Ovládání pohybů je ponecháno částečně na PLC program. Takový způsob připojení panílků s točítkem budeme nazývat: **PLCTOC**.

Nastavení strojních konstant a čtení sejmutých dat:

4. dekáda R53	0	Systém nemá připojen standardní ruční panálek se sériovým točítkem, který plně obsluhuje systém.
3. dekáda R396	0, 1, 2, 3	<p>0: Na systém není připojen panálek typu PLCTOC. 1: Na 1. kanál jednotky CDISTP je připojen panálek typu PLCTOC. 2: Na 2. kanál jednotky CDISTP je připojen panálek typu PLCTOC. 3: Na oba kanály jednotky CDISTP jsou připojeny panálky typu PLCTOC.</p> <p>Sejmutá data z točítok jsou plně k dispozici pro PLC program (včetně točitka): DATA_PLCTOC1 ...pole 4xWORD..... pro 1.panálek DATA_PLCTOC2 ...pole 4xWORD..... pro 2.panálek</p> <p>Rozložení dat: offset +0 offset +1 Wordový čítač točitka offset +3 Bitový obraz stisků tlačítek na panálku (váhy jsou popsány dále)</p> <p>Pokud požadujeme aby obsluha točitka zůstala v režii systému, hodnoty z polí DATA_PLCTOCx nepotřebujeme v PLC programu číst.</p>
4. dekáda R396	0, 1, 2, 3	<p>0: Kolečka panílků PLCTOC systém neobsluhuje 1: Kolečko panílků z 1.kanálu je standardně obsluhováno systémem (přiřazení podle předvolby) 2: Kolečko panílků z 2.kanálu je standardně obsluhováno systémem (přiřazení podle předvolby) 3: Obě kolečka panílků z obou kanálů jsou obsluhovány systémem, přitom 1.kolečko je přiřazeno první souřadnici a 2.kolečko je přiřazeno druhé souřadnici.</p>

		<p>PLC program získá bitový obraz stisků tlačítek panýlků v BYTEových buňkách:</p> <p>PTLTOC_P1Bitový obraz stisků tlačítek 1. panýlku PTLTOC_P1_2 ...Bitový obraz stisků rozšířených tlačítek 1. panýlku</p> <p>PTLTOC_P2Bitový obraz stisků tlačítek 2. panýlku PTLTOC_P2_2 ...Bitový obraz stisků rozšířených tlačítek 2. panýlku</p> <p>(váhy jsou popsány dále)</p>
--	--	--

Bitový obraz stisků tlačítek:

symbol tlačítka X bit: 3, váha 08h	symbol tlačítka Y bit: 2, váha 04h	symbol tlačítka Z bit: 1, váha 02h	symbol tlačítka X bit: 0, váha 01h
symbol tlačítka 5 bit: 7, váha 80h	symbol tlačítka + bit: 6, váha 40h	symbol tlačítka - bit: 5, váha 20h	symbol tlačítka ~ bit: 4, váha 10h

23.10.3 Nastavování kroků točítek

Pro externí nastavování kroků točítek z PLC programu je nutno nastavit bit EXTPAN na hodnotu log.1. Krok točítek se nastavuje pomocí wordových buněk KROK_TOC_EXT a KROK_TOC_EXT2.

Příklad pro nastavení kroku 2. točitka:

```

FL      1,EXTPAN
LOD     CNST.10
STO     KROK_TOC_EXT2

```

Předvolba kroku točitka (5.R396)

Pátá dekáda strojní konstanty R396 určuje předvolbu kroku točitka po zapnutí systému.

5. dekáda R396	hodnota	Předvolba
	0	1 um
	1	5 um
	2	10 um
	3	50 um
	4	100 um
	9	Předvolba podle zapamatované hodnoty před vypnutím systému. Hodnota se ukládá do souboru \$BKP_TOC.SYS v adresáři SYSFILES.

23.10.4 Točítko s displejem

Od softwarové verze sekundárního procesoru 6.211 je možnost připojení k systému dvou externích panílků s ručním ovládáním, s točítkem a s displejem. Ovládání pohybů je ponecháno částečně na PLC program.

Tlačítka z panílků točítka se snímají v PLC programu pomocí buněk PTLTOC_P1, PTLTOC_P1_2 pro první kanál, nebo pomocí buněk PTLTOC_P2, PTLTOC_P2_2 pro druhý kanál. Musí být nastavena 3. 4. dekáda konstanty R396 odpovídajícím způsobem, viz. Připojení dvou sériových točítok s ovládáním z PLC. (4. dekáda konstanty R53 musí být nulová.)

Pro ovládání displeje je pro PLC program definována struktura:

```
;Struktura pro tocitko s displejem
```

```
TOC_DISPLS struc
    TOC_ROW          DB    0      ;radek      (0,1,2,3)
    TOC_COLUMN       DB    0      ;sloupec   (0,1,2,...,19)
    TOC_CHAR1        DB    0      ;1.znak   ASCII hodnota, 0FFh=přeskok
    TOC_CHAR2        DB    0      ;2.znak
    TOC_CHAR3        DB    0      ;3.znak
    TOC_CHAR4        DB    0      ;4.znak
TOC_DISPLS ends
```

Na displej je možno zapsat najednou 4 znaky na pozici danou zvoleným řádkem a zvoleným sloupcem. Znaky se zadávají ASCII hodnotou. Hodnota 0FFh je transparentní znak (výpis pozici přeskočí), to znamená že na dané pozici zůstane svítit minulý zobrazený znak Řádky i sloupce se počítají od nuly.

Pole pro ovládání displeje pro 1.kanál je **DISPTOC1** (8 byte) a pole pro ovládání displeje pro 2.kanál je **DISPTOC2**.

Pro řízení rozpracovanosti slouží bity **BUSY_DISPTOC1** a **BUSY_DISPTOC2**. Bity nastavuje PLC program na hodnotu log.1 v době, když data v polích DISPTOC1 a DISPTOC2 jsou ještě neplatná a způsobí zákaz obsluhy displeje.

Příklad:

Zobrazení znaků Abc v 2.řádku a na 4.sloupci:

```
FL    1, BUSY_DISPTOC1      ;zákaz obsluhy
LOD   CNST.1                ;2. radek
STO   BYTE.DISPTOC1.TOC_ROW
LOD   CNST.3                ;4. sloupec
STO   BYTE.DISPTOC1.TOC_COLUMN
LOD   CNST.'A'              ;A
STO   BYTE.DISPTOC1.TOC_CHAR1
LOD   CNST.'b'              ;b
STO   BYTE.DISPTOC1.TOC_CHAR2
LOD   CNST.'c'              ;c
STO   BYTE.DISPTOC1.TOC_CHAR3
LOD   CNST.0FFh             ;transparent
STO   BYTE.DISPTOC1.TOC_CHAR4
FL    0, BUSY_DISPTOC1      ;povolení obsluhy
```

23.11 Připojení a obsluha CAN-BUSu pro PLC program

verze	popis
6.334	Možnost výhradního použití CAN kanálu pro PLC program
6.388	Možnost sdíleného použití CAN kanálu pro PLC program

Výhradní požití CAN kanálu v PLC programu znamená, že PLC program plně obsluhuje CAN-BUS kanál. Systém tento CAN kanál vůbec nepoužívá, jen provede počáteční inicializaci a aktivaci PNP karty pro zpřístupnění CAN registrů. Systém už neprovede ani inicializaci CAN kontroléru. PLC program sám vysílá a přímá své pakety na CAN-BUS kanál.

Sdílené použití CAN kanálu v PLC programu znamená, že PLC program používá CAN-BUS kanál společně se systémem, nebo alespoň pro jeho řízení používá systémové prostředky. Inicializaci CAN kontroléru a řízení vysílání a příjmu paketů také provádí systém. PLC program jen žádá a zařazení svých paketů do fronty pro vyslání a čte frontu přijmutých paketů.

23.11.1 CAN kanály a použitý hardware

Označení CAN kanálů z logického pohledu (nezávisle na hardware):

pořadí	Logické označení	použití
1	CAN1	Připojení pohonů
2	CAN2	Připojení periférií (vstupy a výstupy)
3	CAN3	Připojení periférií (vstupy a výstupy)

Fyzicky jsou CAN kanály umístěny po dvojicích na PCI kartách. V současné verzi software jsou podporovány karty: PCAN od firmy Peak, MCAN (Tedia) firma Mefi nebo PCAN-Dongle od firmy Peak.

O přiřazení fyzických CAN kanálů k logickým CAN kanálům rozhodují 5. a 6. dekáda strojních konstant **R590** a **R770**. Strojní konstanta R590 nastavuje logický CAN1 kanál pro připojení pohonů a strojní konstanta R770 nastavuje logický CAN2 kanál pro připojení periférií:

5.6. dekáda R590 Připojení CAN-BUS pohonů CAN1	5.6. dekáda R770 Připojení CAN-BUS periférií CAN2
1 = 1.fyzický kanál na PCAN (Peak)	2 = 2.fyzický kanál na PCAN (Peak)
2 = 2.fyzický kanál na PCAN (Peak)	1 = 1.fyzický kanál na PCAN (Peak)
3 = 1.fyzický kanál na MCAN (Mefi, Tedia)	4 = 2.fyzický kanál na MCAN (Mefi, Tedia)
4 = 2.fyzický kanál na MCAN (Mefi, Tedia)	3 = 1.fyzický kanál na MCAN (Mefi, Tedia)

23.11.2 Výhradní přístup pro CAN kanál

Funkce pro výhradní obsluhu CAN-BUSu jsou do PLC programu implementovány jako 6 speciálních instrukcí.

Instrukce pro výhradní přístup	popis
CAN_INIT	Inicializace a módování CAN kontroleru
CAN_READ	Čtení přijatého paketu (zprávy)
CAN_WRITE	Zápis paketu (zprávy) pro vyslání
CAN_STATUS	Zjištění stavu CAN kontroleru
CAN_CLOSE	Dezaktivace CAN kontroleru
CAN_REGISTERMSG	Zaregistrování paketů

Definované symboly pro CAN-BUS

Důležité předdefinované konstanty (konstanty jsou definovány v souboru EXT_05.ASH)

Konstanty pro nastavení Bd rychlosti:		
Symbol	Rychlost	Hodnota (BTR0/BTR1)
CAN_BAUD_1M	1 MBit / s	0014h
CAN_BAUD_500K	500 kBit / s	001Ch
CAN_BAUD_250K	250 kBit / s	011Ch
CAN_BAUD_125K	125 kBit / s	031Ch
CAN_BAUD_100K	100 kBit / s	432Fh
CAN_BAUD_50K	50 kBit / s	472Fh
CAN_BAUD_20K	20 kBit / s	532Fh
CAN_BAUD_10K	10 kBit / s	672Fh
CAN_BAUD_5K	5 kBit / s	7F7Fh

Masky bitů pro vyhodnocování		
Symbol	Maska	Význam
CAN_ERR_OK	00h	Vše v pořádku
CAN_ERR_XMTFULL	01h	Plný buffer pro vysílání
CAN_ERR_RCVEMPTY	02h	Nepřijala se nová zpráva
CAN_ERR_OVERRUN	04h	Přijatá zpráva nebyla přečtena v požadovaném čase
CAN_ERR_BUSERROR	08h	CAN kontroler hlásí přerušení sběrnice
CAN_ERR_BUSOFF	10h	CAN kontroler má vypnutou sběrnici
CAN_ERR_REGTEST	20h	Chyba při testování registrů CAN kontroleru

Symboly možno použít v instrukcích porovnání (EQ, LT, LE,...) a v instrukcích bitových operacích s daty:

Příklad:

```

EQ      CAN_ERR_OK
ANDB    CAN_ERR_RCVEMPTY

```

Formální definice bitů pro vyhodnocení		
Název bitu	Bit	Význam
CAN_XMTFULL	Bit 0	Plný buffer pro vysílání
CAN_RCVEMPTY	Bit 1	Nepřijala se nová zpráva
CAN_OVERRUN	Bit 2	Přijatá zpráva nebyla přečtena v požadovaném čase
CAN_BUSERROR	Bit 3	CAN kontroler hlásí přerušení sběrnice
CAN_BUSOFF	Bit 4	CAN kontroler má vypnutou sběrnici
CAN_REGTEST	Bit 5	Chyba při testování registrů CAN kontroleru

Symboly možno použít v bitových operacích pomocí „složitější adresace bitů“.

Příklad: LDR ERR_CAN.CAN_RCVEMPTY

Struktura zprávy pro vysílání a příjem paketu

```
;CAN-Message
TCANMSGSGS STRUC
    CAN_ID    DW 0           ;11 Bit-ID
    CAN_RTR   DB 0           ;true, if remote request
    CAN_LEN   DB 0           ;Number of valid Data bytes (0..8)
    CAN_DATA  DB 8 DUP (0)   ;Databytes 0..7
TCANMSGSGS ENDS
```

Maximální délka paketu je 12 bajtů a v PLC programu musí být místo pro vysílané nebo přijímané pakety definováno pomocí řetězcové instrukce **STR** (instrukce automaticky vygeneruje pointer na paket).

Příklad:

```
CANBUFF_IN:   STR    12    ;místo pro příjem paketu
CANBUFF_OUT:  STR    12    ;místo pro vyslání paketu
```

Práce s daty v jednotlivých paketech je plně v režii PLC programu, včetně přidělení ID pro jednotlivé periferie. Význam jednotlivých dat se liší podle typu CAN periferie.

Konfigurace jednotlivých CAN periférií se může provést v PLC programu, ale jednodušší je periferie nakonfigurovat mimo systém pomocí dodávaného programu CAN-MONITOR (například CANVIEW od firmy PEAK). Programem se přidělí jednotlivým perifériím 11 bitové ID adresy a každá periferie se nakonfiguruje na požadovaný způsob práce (například automatické vysílání hodnot do systému po 1 ms). Konfigurace se v CAN perifériích automaticky zapisuje do paměti EEPROM nebo FLASH.

Instrukce pro práci s CAN-BUSEm s výhradním přístupem

instrukce	CAN_INIT
------------------	-----------------

funkce **Inicializace a módování CAN kontroleru**

syntax **CAN_INIT lin, port, baud**

Instrukce slouží pro inicializaci a nastavení CAN kontroleru. Instrukce po vykonání vrátí v datovém registru výsledek, který je složen z masek chyb (popsaných dříve). Pokud při inicializaci nevznikla chyba, vrácená hodnota bude CAN_ERR_OK (00h). Při chybě bude vrácená hodnota CAN_ERR_REGTEST (20h) .

Význam parametrů:

Parametr	Hodnota (příklad)	Význam
lin	1, 2	CAN kanál (CAN2)
port	378h (lpt1)	Adresa portu pro připojení CAN kontroleru (PCAN-DONGLE)
	1, 2	1. nebo 2. fyzický CAN kanál karty PCAN (Peak)
	3, 4	1 nebo 2. fyzický CAN kanál karty MCAN (Mefi, Tedia)
baud	CAN_BAUD_500K	Požadovaná rychlost komunikace

Příklad 1:

```

CAN_INIT     2, 4, CAN_BAUD_500K     ;inicializace CAN kontroleru
EQ            CAN_ERR_OK               ;test návratové hodnoty
JL0           CAN_ERROR                ;hlášení chyby

```

Příklad 2:

```

CAN_INIT     2, 2, CAN_BAUD_500K     ;inicializace CAN kontroleru
STO           ERR_CAN                  ;přepis výsledku
LDR           ERR_CAN.CAN_REGTEST     ;prošel test registrů kontroleru ?
JL1           CAN_ERROR                ;hlášení chyby

```

Instrukce se může umístit do modulu PIS_INIT a PIS_CLEAR (MODULE_INIT a MODULE_CLEAR). Pokud inicializace neproběhne správně, nesmí se spustit běžný provoz CAN linky.

Po průběhu instrukce CAN_INIT je CAN kontroler nastaven tak, že přijme pakety s libovolným ID. Pokud by jsme potřebovali omezit příjem paketů jen na pakety s očekávaným ID, je potřeba tyto pakety zaregistrovat pomocí instrukce CAN_REGISTERMSG.

CAN kontroler je připojen na paralelní port. Paralelní port se musí nastavit v SETUPu BIOSu na EPP mód, nebo EPP/ECP mód. Nesmí být nastaven na SPP a NORMAL mód.

instrukce	CAN_READ
------------------	-----------------

funkce Čtení přijatého paketu (zprávy)

syntax **CAN_READ** **lin, poimsg**

Instrukce slouží pro načtení přijatého paketu (zprávy). Instrukce po vykonání vrátí v datovém registru výsledek, který je složen z masek chyb (popsaných dříve). Pokud při čtení nevznikla chyba, vrácená hodnota bude `CAN_ERR_OK` (00h). Při chybě bude vrácená hodnota složena z masek `CAN_ERR_RCVEMPTY` (02h), `CAN_ERR_OVERRUN` (04h), `CAN_ERR_BUSERROR` (08h) a `CAN_ERR_BUSOFF` (10h). Význam jednotlivých bitů byl popsán dříve. Jednotlivé chyby je výhodné testovat jako bity pomocí „složitější adresace bitů“.

CAN kontroler má vestavěný buffer pro příjem 8 paketů, proto se doporučuje při čtení opakovat instrukci `CAN_READ` 8x.

Význam parametrů:

Parametr	Hodnota (příklad)	Význam
lin	1, 2	CAN kanál (CAN2)
poimsg	CANBUFF_IN	Pointer, který ukazuje na vyhrazené místo 12 bajtů. Návěští u instrukce STR, kde je místo pro načtení paketu.

Příklad:

```

CANBUFF_IN:      STR    12                ;místo pro načtení paketu

      CAN_READ    2, CANBUFF_IN          ;Čtení paketu z CAN-BUSu
      EQ          CNST.0                 ;Bylo přečteno ?
      JL1        NOVY                    ;Přišel nový paket
      STO        ERR_CAN                 ;Vrácená hodnota
      LDR        ERR_CAN.CAN_RCVEMPTY    ;Nic se nepřijmulo ?
      JL1        NIC

```

Podrobnější příklad bude uveden v závěru této kapitoly

instrukce	CAN_WRITE
------------------	------------------

funkce Zápis paketu (zprávy) pro vyslání

syntax **CAN_WRITE** **lin, poimsg**

Instrukce slouží pro zápis paketu (zprávy) pro vyslání na CAN-BUS. Instrukce po vykonání vrátí v datovém

registru výsledek, který je složen z masek chyb (popsaných dříve). Pokud při zápisu nevznikla chyba, vrácená hodnota bude CAN_ERR_OK (00h). Při chybě bude vrácená hodnota CAN_ERR_XMTFULL (01h)

Význam parametrů:

Parametr	Hodnota (příklad)	Význam
lin	1, 2	CAN kanál (CAN2)
poimsg	CANBUFF_OUT	Pointer, který ukazuje na vyhrazené místo 12 bajtů. Návěští u instrukce STR, kde je místo pro vysílání paketu.

Příklad:

```

CANBUFF_OUT:      STR    12                ;místo pro vysílání paketu

        CAN_WRITE  1, CANBUFF_OUT        ;Vysílání paketu na CAN-BUSu
        EQ         CNST.0                ;Bylo vysláno ?
        JL1        OK                    ;Vysláno OK
        STO        ERR_CAN               ;Vrácená hodnota
        LDR        ERR_CAN.CAN_XMTFULL   ;Nic se nevyslalo
        JL1        NIC                   ; (ještě je plný buffer)

```

Podrobnější příklad bude uveden v závěru této kapitoly

instrukce	CAN_STATUS
------------------	-------------------

funkce **Zjištění stavu CAN kontroleru**

syntax **CAN_STATUS lin**

Instrukce slouží pro zjištění stavu CAN kontroleru. Instrukce po vykonání vrátí v datovém registru výsledek, který je složen z masek chyb (popsaných dříve). Pokud je kontroler v klidovém stavu, vrácená hodnota bude CAN_ERR_OK (00h). Při chybě bude vrácená hodnota složena z masek CAN_ERR_XMTFULL (01h), CAN_ERR_RCVEMPTY (02h), CAN_ERR_OVERRUN (04h), CAN_ERR_BUSERROR (08h) a CAN_ERR_BUSOFF (10h).

Význam jednotlivých bitů byl popsán dříve. Jednotlivé chyby je výhodné testovat jako bity pomocí „složitější adresace bitů“.

Význam parametrů:

Parametr	Hodnota (příklad)	Význam
lin	1, 2	CAN kanál (CAN2)

instrukce	CAN_CLOSE
------------------	------------------

funkce **Dezaktivace CAN kontroleru**

syntax **CAN_STATUS lin**

Instrukce slouží pro dezaktivaci CAN kontroleru. Instrukce nevrací žádnou hodnotu o provedené akci.

Význam parametrů:

Parametr	Hodnota (příklad)	Význam
lin	1, 2	CAN kanál (CAN2)

instrukce	CAN_REGISTERMSG
------------------	------------------------

funkce **Registrování paketů**

syntax **CAN_REGISTERMSG lin, ID**

Instrukce slouží pro zaregistrování očekávaných paketů pro příjem. Po průběhu instrukce CAN_INIT je CAN kontroler nastaven tak, že přijme pakety s libovolným ID. Pro omezení příjmu paketů jenom na ty které mají očekávané ID, slouží instrukce CAN_REGISTERMSG. Instrukce může být použita vícekrát za sebou pro různé ID.

Parametr	Hodnota (příklad)	Význam
lin	1, 2	CAN kanál (CAN2)
ID	10h	Číslo ID pro registraci

Příklad pro rychlé vstupy a výstupy na CAN periférii s výhradním přístupem

Příklad řeší příjem a vysílání osmi-bitových portů v rastru 1 ms.

Použitá periférie: karta PCI - MCAN, 2. fyzický kanál.

ID adresa periférie je nastavena na 10h.

Periférie je nakonfigurována na automatické vysílání dat po 1 ms.

V deklaraci dat:

```
;Definice chyb
EQUI  ERR_OVERUN,20h    ;Prijmuta zprava nebyla prectena vcas (neni chyba)
EQUI  ERR_BUSERROR,21h  ;CAN kontroler hlasi preruseni zbernice
EQUI  ERR_BUSOFF,22h    ;CAN kontroler ma vypnutou zbernici
EQUI  ERR_CANINIT,23h   ;Chyba konfigurace CAN
EQUI  ERR_XMTFULL,24h   ;Plny buffer pro vysilani (neni chyba)

ERR_CAN_READ:          DS 1          ;definice bitu dle formalni deklarace TCANRSR
ERR_CAN_WRITE:         DS 1          ;definice bitu dle formalni deklarace TCANRSR
CANBUFF_IN:            STR 12        ;misto na prijem paketu
CANBUFF_OUT:           STR 12        ;misto pro vysilani paketu

PAM_CAN:               DFM CAN_OK,,,,,,
ERR_CANBUS_NODATA:     DS 2          ;dgn. citac - nejsou nova data pro prijem
ERR_CANBUS_OVERUN:    DS 2          ;dgn. citac - nevyzvednuta data na prijmu
ERR_CANBUS_XMTFULL:   DS 2          ;dgn. citac - plny buffer pro vysilani
ERR_CANBUS_CYKL:      DS 2          ;dgn. citac - pocet cyklu vybirani paketu
```

V inicializaci:

```
CAN_INIT    2, 4, CAN_BAUD_500K    ;Inicializace CAN kontroleru MCAN
EQ          CAN_ERR_OK
WR          CAN_OK                  ;CAN je OK ?
CA
ESET1      ERR_CANINIT              ;podminene hlaseni chyby
```

V modulu PIS_FAST:

```

        LDR          CAN_OK                ;Je CAN-BUS ?
        JLO          CAN_VYS

;**** Obsluha CAN-BUSu ... cteni rychlych vstupu ****

        LOD          CNST.0
        STO          ERR_CANBUS_CYKL

CAN_RD:                                ;Cyklus
        CAN_READ     2,CANBUFF_IN         ;Cteni paketu z CAN-BUSu
        EQ            CAN_ERR_OK          ;Je novy paket ?
        JL1          CAN_NEW              ;Ano
        STO          ERR_CAN_READ

        FL            0,ERR_CAN_READ.CAN_RCVEMPTY ;CAN_ERR_RCVEMPTY neni error
        LOD          ERR_CAN_READ
        EQ            CNST.0
        JL1          CAN_EEE_N            ;Nic se neprijmulo

;Rozkodovani chyb po prijmu
        LOD          CNST.ERR_BUSERROR
        LDR          ERR_CAN_READ.CAN_BUSERROR ;CAN - preruseni zbernice
        JL1          CAN_ERR
        LOD          CNST.ERR_BUSOFF
        LDR          ERR_CAN_READ.CAN_BUSOFF ;CAN - vypnuta zbernice
        JL1          CAN_ERR
        LDR          ERR_CAN_READ.CAN_OVERRUN ;Zprava neprectena vcas
        JL1          CAN_NEW              ;Nepoklada se za chybu
        JUM          CAN_EEE

CAN_ERR:                                ;Hlaseni chyby
        ESET
        JUM          CAN_EEE

;Test na prijem paketu ID=10h+1, delka=2, Byte1=Opc=1, Byte2=Input
CAN_NEW:
        LOD          WORD.CANBUFF_IN.CAN_ID ;Test ID
        ANDB         CNST.7FFh             ;11 bitove ID
        EQ            CNST.10h+1           ;Odpoved od ID=10h ?
        JLO          CAN_OTH

        LOD          BYTE.CANBUFF_IN.CAN_LEN ;Delka
        EQ            CNST.2               ;Je delka = 2 ?
        JLO          CAN_OTH

        LOD          BYTE.CANBUFF_IN.CAN_DATA+0 ;Opcode
        EQ            CNST.1               ;Opcode 1 pro vstupy ?
        JLO          CAN_OTH

        LOD          BYTE.CANBUFF_IN.CAN_DATA+1 ;Sejmuti vstupnich dat !
        STO          IP_CAN                 ;Misto pro nova data

        LDR          ERR_CAN_READ.CAN_OVERRUN ;Zprava neprectena vcas
        JL1          CAN_EEE_O             ;Nepoklada se za chybu
        JUM          CAN_OTH

```

```
CAN_OTH:                                ;Jiny paket, opakujeme
    LOD      ERR_CANBUS_CYKL            ;pocet cyklu (8)
    INR
    STO      ERR_CANBUS_CYKL
    LE       CNST.8                     ;max 8 paketu
    JL1      CAN_RD                     ;cyklus
    JUM      CAN_EEE                   ;konec snimani

CAN_EEE_O:                              ;Dgn.- nevyzvednuta data
    LOD      ERR_CANBUS_OVERUN         ;citac
    INR
    STO      ERR_CANBUS_OVERUN
    JUM      CAN_OTH

CAN_EEE_N:                              ;Dgn.- nejsou nova data
    LOD      ERR_CANBUS_NODATA         ;citac
    INR
    STO      ERR_CANBUS_NODATA

CAN_EEE:

;**** Obsluha CAN-BUSu ... vysilani rychlych vystupu ****
    LOD      CNST.10h                  ;Vyslani na ID=10h
    STO      WORD.CANBUFF_OUT.CAN_ID
    LOD      CNST.0h                   ;Neni "remote request"
    STO      BYTE.CANBUFF_OUT.CAN_RTR
    LOD      CNST.2                     ;delka paketu
    STO      BYTE.CANBUFF_OUT.CAN_LEN
    LOD      CNST.2                     ;Opcode = 2
    STO      BYTE.CANBUFF_OUT.CAN_DATA+0
    LOD      IP0                        ;data pro vyslani
    STO      BYTE.CANBUFF_OUT.CAN_DATA+1

    CAN_WRITE 2,CANBUFF_OUT            ;Vyslani paketu na CAN-BUS
    EQ        CAN_ERR_OK               ;Je vyslani OK ?
    JL1      CAN_VYS
    STO      ERR_CAN_WRITE

    LDR      ERR_CAN_WRITE.CAN_XMTFULL ;Test zda je plny buffer
    JL1      ERR_CANBUS_FULL           ;Plny buffer - nevyslano
    JUM      CAN_VYS

ERR_CANBUS_FULL:
    LOD      ERR_CANBUS_XMTFULL        ;Dgn.: plny buffer
    INR
    STO      ERR_CANBUS_XMTFULL

CAN_VYS:
```

23.11.3 Sdílený přístup pro CAN kanál

Sdílené použití CAN kanálu v PLC programu znamená, že PLC program používá CAN-BUS kanál společně se systémem, nebo alespoň pro jeho řízení používá systémové prostředky. Inicializaci CAN kontroléru a řízení vysílání a příjmu paketů také provádí systém. PLC program jen žádá a zařazení svých paketů do fronty pro vysílání a čte frontu přijmutých paketů.

Konfigurace CAN kanálu pro sdílený přístup

Konfigurace CAN kanálu pro sdílený přístup se nastavuje pomocí strojních konstant. Strojní konstanta R590 nastavuje logický CAN1 kanál pro připojení pohonů a strojní konstanta R770 nastavuje logický CAN2 kanál pro připojení periférií:

R770 (NASTAVENÍ CAN-BUSU PRO PERIFERIE – CAN2)

Dekáda	Hodnota	Popis	Doporuč.hodnota
1. a 2. dekáda	0	CAN-BUS pro periferie zakázán	1
	1	CAN-BUS pro periferie povolen	
3. a 4. dekáda	0	Rychlost 1 MBd	0
	1	Rychlost 500 kBd	
	2	Rychlost 250 kBd	
	3	Rychlost 125 kBd	
	4	Rychlost 100 kBd	
5. a 6. dekáda	0	Hardware pro CAN-BUS: „Peak Dongle EPP mód“	2, 4
	1	1.fyzický kanál na PCAN (Peak)	
	2	2.fyzický kanál na PCAN (Peak)	
	3	1.fyzický kanál na MCAN (Mefi, Tedia)	
	4	2.fyzický kanál na MCAN (Mefi, Tedia)	
7. a 8. dekáda	0	Obsluha CAN-BUSu po ¼ ms	0
	2	Obsluha CAN-BUSu po 1 ms	

Podle 5. a 6. dekády konstant R590 a R770 se automaticky přednastaví „VENDOR ID“ a „DEVICE ID“. Proto za standardních podmínek strojní konstanty R596 a R597 není potřeba nastavovat.

Masky bitů pro vyhodnocování		
Symbol	Maska	Význam
CAN_ERR_OK	00h	Vše v pořádku
CAN_ERR_XMTFULL	01h	Plná fronta pro vysílání
CAN_ERR_RCVEMPTY	02h	Nepřijala se nová zpráva

Instrukce pro práci s CAN-BUSEm se sdíleným přístupem

Funkce pro sdílenou obsluhu CAN-BUSu jsou do PLC programu implementovány jako 2 speciální instrukce.

Instrukce pro sdílený přístup	Popis
CAN_SEND	Zařazení paketu do fronty pro vysílání na periferii
CAN_RECV	Čtení paketu z příjmové fronty

instrukce CAN_SEND

funkce **Zařazení paketu do výstupní fronty**

syntax **CAN_SEND can, poimsg**

Instrukce slouží pro zařazení paketu do výstupní fronty paketů, které jsou vysílány přes CAN kanál na periferii. Instrukce po vykonání vrátí v datovém registru výsledek, který je složen z masek chyb (popsaných dříve). Pokud při řazení nevznikla chyba, vrácená hodnota bude CAN_ERR_OK (00h). pokud se paket do fronty nepodařilo zařadit (fronta je přeplněna), bude vrácená hodnota CAN_ERR_XMTFULL (01h)

Význam parametrů:

Parametr	Hodnota (příklad)	Význam
can	2	Logický CAN kanál (CAN2)
poimsg	CANBUFF_OUT	Pointer, který ukazuje na vyhrazené místo 12 bajtů. Návěští u instrukce STR, kde je místo pro paket

Příklad:

```

CANBUFF_OUT:      STR    12                ;výstupní paket

      CAN_SEND     2, CANBUFF_OUT          ;Zařazení paketu
      EQ           CAN_ERR_OK              ;Bylo zařazeno ?
      JL1         OK                       ;zařazeno
                                           ;nezařazeno

```

Podrobnější příklad bude uveden v závěru této kapitoly

instrukce	CAN_RECV
------------------	-----------------

funkce Čtení paketu z příjmové fronty

syntax CAN_RECV can, poimsg

Instrukce slouží pro čtení paketu z fronty paketů, které byly přijaty z periférie přes CAN kanál. Instrukce po vykonání vrátí v datovém registru výsledek, který je složen z masek chyb (popsaných dříve). Pokud při čtení nevznikla chyba, vrácená hodnota bude CAN_ERR_OK (00h). pokud se žádný paket pro PLC nepřijal, bude vrácená hodnota CAN_ERR_RCVEMPTY (02h)

Význam parametrů:

Parametr	Hodnota (příklad)	Význam
can	2	Logický CAN kanál (CAN2)
poimsg	CANBUFF_IN	Pointer, který ukazuje na vyhrazené místo 12 bajtů. Návěští u instrukce STR, kde je místo pro paket

Příklad:

```

CANBUFF_IN:      STR    12                ;vstupní paket

                  CAN_RECV    2, CANBUFF_IN    ;Přijem paketu
                  EQ          CAN_ERR_OK      ;Bylo přijato ?
                  JL1         OK              ;přijato
                                          ;nic se nepřijalo

```

Podrobnější příklad bude uveden v závěru této kapitoly

Příklad pro vstupy a výstupy na CAN periférii se sdíleným přístupem

ID adresa periférie je nastavena na 10h.

V deklaraci dat:

```
CANBUFF_IN:      STR 12      ;misto na prijem paketu
CANBUFF_OUT:     STR 12      ;misto pro vysilani paketu
```

V modulu MODULE_MAIN:

```

CAN_RECV 2,CANBUFF_IN          ;Cteni paketu
EQ      CAN_ERR_OK             ;Je novy paket ?
JL0     NIC_NEPRIJATO

;Test na prijem paketu ID=10h+1, delka=2, Byte1=Opc=1, Byte2=Input
LOD      WORD.CANBUFF_IN.CAN_ID      ;Test ID
ANDB     CNST.7FFh                  ;11 bitove ID
EQ       CNST.10h+1                 ;Odpoved od ID=10h ?
JL0      CAN_OTH
LOD      BYTE.CANBUFF_IN.CAN_LEN      ;Delka
EQ       CNST.2                      ;Je delka = 2 ?
JL0      CAN_OTH
LOD      BYTE.CANBUFF_IN.CAN_DATA+0   ;Opcode
EQ       CNST.1                      ;Opcode 1 pro vstupy ?
JL0      CAN_OTH
LOD      BYTE.CANBUFF_IN.CAN_DATA+1   ;Sejmuti vstupnich dat !
STO      IP_CAN                      ;Misto pro nova data
CAN_OTH:                                ;Jiny paket, opakujeme

;**** Obsluha CAN-BUSu ... vysilani vystupu ****
LOD      CNST.10h                    ;Vyslani na ID=10h
STO      WORD.CANBUFF_OUT.CAN_ID
LOD      CNST.0h                      ;Neni "remote request"
STO      BYTE.CANBUFF_OUT.CAN_RTR
LOD      CNST.2                        ;delka paketu
STO      BYTE.CANBUFF_OUT.CAN_LEN
LOD      CNST.2                        ;Opcode = 2
STO      BYTE.CANBUFF_OUT.CAN_DATA+0
LOD      IP0                          ;data pro vyslani
STO      BYTE.CANBUFF_OUT.CAN_DATA+1

CAN_SEND 2,CANBUFF_OUT              ;Zarazeni paketu do fronty
```

23.11.4 Ovládání systémových CAN-BUS periférií

PLC program má možnost číst a nastavovat struktury pro systémové CAN-BUS periferie, jako jsou INOUT08 a KLA50. Kromě toho zůstává možnost použít instrukce CAN_SEND a CAN_RECV pro sdílený přístup.

Nastavení **NODE-ID** pro systémové CAN-BUS periferie:

skupina (group)		Pořadové číslo jednotky ve skupině (board)						
		1	2	3	4	5	6	...
INOUT08	1	21h	22h	23h	24h	25h	26h	20h+board
KLA50	2	41h	42h	43h	44h			40h+board

Funkce pro přístup k systémovým strukturám CAN-BUS periférií jsou do PLC programu implementovány jako 2 speciální instrukce.

Instrukce pro sdílený přístup	Popis
CAN_SYSIO_READ	Čtení prvku z pole systémových struktur pro CAN-BUS periferie
CAN_SYSIO_WRITE	Zápis prvku do pole systémových struktur pro CAN-BUS periferie

Každá periferie má přidělenou jednu strukturu INOUTS. Prvky struktury je možno sledovat také pomocí diagnostické obrazovky „CAN-BUS peripherals diagnostic“. Pomocí této obrazovky je umožněno také vysílat a přijímat SDO pakety.

```

INOUTS  STRUC
        SIZE          DW      0      ;velkost
        PRESENT        DB      0      ;Je jednotka pritomna ?
        MODE           DB      0      ;mod (0=standard, 1=analog, 2=matice)
        VENDORID       DD      0      ;SDO 1018
        DEVICENAME     DD      0      ;SDO 1008
        HWVERSION      DD      0      ;SDO 1009
        SWVERSION      DD      0      ;SDO 100A

        IN0            DB      0      ;vstupy
        IN1            DB      0
        IN2            DB      0
        IN3            DB      0

        OUT0           DB      0      ;vystupy
        OUT1           DB      0
        OUT2           DB      0
                   DB      0

        ANL0           DB      0      ;analog
        ANL1           DB      0
        ANL2           DB      0
        ANL3           DB      0
        ANL4           DB      0
        ANL5           DB      0
        ANL6           DB      0
        ANL7           DB      0

        ERROR          DB      0      ;error registr 1001 - okamzity
        ERR_CODE       DB      0      ;kod chyby ... zapise pri vzniku chyby

        ERR_STAT       DB      0      ;hlaseni chyb - record E_IOR
                                ;bit0 (IO_ER_HLI_ACT) = chyba (INOUT_ERR_CODE)
                                ;bit1 (IO_ER_TM_ACT) = time-out
                                ;bit2 (IO_ER_HLI) = chyba (INOUT_ERR_CODE)
                                ;bit3 (IO_ER_TMOUT) = time-out pamet
                                ;bit4 (IO_ER_NULL) = zadost o nulovani periferii

```



```

CONTROL      DB      0      ;rizeni - record C_IOR
                                ;bit0 (IO_DIS_ERR) = blokovani vypisu chyb
                                ;bit1 (IO_DIS_ERPIS) = blokovani chyb pro PLC
                                ;bit2 (IO_DIS_TMOUT) = blokovani chyb time-out
                                ;bit6 (IO_DIS_NULL) = blokovani nulovani periferii
COUNT       DW      0      ;citac pro time-out

SEND_REQ     DB      0      ;zadost o vyslani SDO paketu INOUT_SEND
RESP_COUNT   DB      0      ;citac prijmu
SHORT_SEND_REQ DB      0      ;zadost o informaci zkratu

SHORT_OUT0    DB      0      ;informace o zkratu
SHORT_OUT1    DB      0
SHORT_OUT2    DB      0

SEND          TCANMSG2 <0> ;paket pro vyslani
RESP          TCANMSG2 <0> ;response paket

RPDO_COUNT   DW      0      ;dgn cistac zarazenych RPDO paketu pro vyslani
TPDO_COUNT   DW      0      ;dgn cistac prijmutych TPDO paketu

PxIN          DW      0      ;cislo vstupniho PLC portu 1=PlIN
PxOUT         DW      0      ;cislo vystupniho PLC portu 1=PlOUT

lpINOUT_PxIN   DW 2 DUP (0) ;pointry na PLC oblasti
lpINOUT_PxOUT   DW 2 DUP (0)
lpINOUT_ERRHI_PxOUT DW 2 DUP (0)
lpINOUT_SETH_PxOUT DW 2 DUP (0)
lpINOUT_SETH_PxIN DW 2 DUP (0)

MODE_ACT      DB      0      ;mod aktual
MODE_FAZE     DB      0

TL0           DB      0      ;tlacitka matice
TL1           DB      0
TL2           DB      0
TL3           DB      0

IRC0          DW      0      ;irc
IRC1          DW      0
INOUTS ENDS

```

instrukce	CAN_SYSIO_READ
------------------	-----------------------

funkce Čtení prvku z pole systémových struktur pro CAN-BUS periferie

syntax CAN_SYSIO_READ can, group, board, item

Instrukce **CAN_SYSIO_READ** slouží pro načtení jednoho prvku z pole struktur INOUTS. Instrukce načte do DR registru bajt, word nebo double-word podle typu zadaného prvku. Pokud se instrukce provede bez chyb (prvek ve struktuře se najde), tak vrací RLO registr nastaven na hodnotu 0. Pokud RLO registr je nastaven na hodnotu 1, potom instrukce skončila s chybou a data v DR registru nejsou platná.

Význam parametrů:

Parametr	Hodnota (příklad)	Význam
can	2	Logický CAN kanál (CAN2)
group	1, 2	Skupina CAN-BUS periferií. 1 = jednotky vstupů a výstupů INOUT08 2 = jednotky maticových vstupů KLA50
board	1, 2, ..., 32	Pořadové číslo jednotky ve skupině. Každá skupina má maximálně 32 jednotek. NODE-ID jednotky musí být nastaveno s ohledem na pořadové číslo jednotky a s ohledem na skupinu.
item	(MODE, IN0, ...)	Identifikátor prvku struktury podle definice struktury INOUTS

Příklad:

Načtení analogového napětí z 3.jednotky INOUT08 (NODE-ID = 23h):

```
CAN_SYSIO_READ    2, 1, 3, ANL0    ;1.skupina,3.jednotka,prvek ANL0
JL1               Error
```

instrukce	CAN_SYSIO_WRITE
------------------	------------------------

funkce **Zápis do prvku v poli systémových struktur pro CAN-BUS periferie**

syntax **CAN_SYSIO_WRITE can, group, board, item**

Instrukce **CAN_SYSIO_WRITE** slouží pro zápis jednoho prvku do pole struktur INOUTS. Instrukce zapíše obsah DR registru do struktury jako bajt, word nebo double-word podle typu zadaného prvku. Pokud se instrukce provede bez chyb (prvek ve struktuře se najde), tak vrací RLO registr nastaven na hodnotu 0. Pokud RLO registr je nastaven na hodnotu 1, potom instrukce skončila s chybou a data se do struktury nezapsala.

Význam parametrů:

Parametr	Hodnota (příklad)	Význam
can	2	Logický CAN kanál (CAN2)
group	1, 2	Skupina CAN-BUS periferií. 1 = jednotky vstupů a výstupů INOUT08 2 = jednotky maticových vstupů KLA50
board	1, 2, ..., 32	Pořadové číslo jednotky ve skupině. Každá skupina má maximálně 32 jednotek. NODE-ID jednotky musí být nastaveno s ohledem na pořadové číslo jednotky a s ohledem na skupinu.
item	(MODE, IN0, ...)	Identifikátor prvku struktury podle definice struktury INOUTS

Příklad:

Namódování 4.jednotky INOUT08 pro snímání analogových vstupů:

```
LOD               CNST.1           ;mode=1 pro analogové vstupy
CAN_SYSIO_WRITE   2, 1, 4, MODE    ;1.skupina,4.jednotka,prvek MODE
JL1               Error
```

23.12 Konstantní řezná rychlost – verze 2

KŘR není omezena na pohybové bloky a také ji možno plně řídit z PLC programu. Systém i PLC mají možnost omezit otáčky vřetene na zadanou hodnotu. KŘR verze 2, je do značné míry kompatibilní se starší verzí, ale poskytuje mnohé vlastnosti navíc. Pokud nebude výslovně upozorněno, platí pro novější KŘR vše, co bylo napsáno v předcházející části. V dalším popisu se zaměříme jen na nové vlastnosti KŘR verze 2.

Novější verzi KŘR možno nastavit od verze software primárního procesoru 40.44 a sekundárního procesoru 6.369. Aktivace se provede nastavením hodnoty **1** do strojní konstanty **R580**.

Pro obvodovou rychlost platí:

$$v = \omega \cdot r = 2 \cdot \pi \cdot n \cdot x$$

x = poloha řídící souřadnice (korigována vzhledem ke korekcím a posunutí)
v = obvodová rychlost (KŘR)
n = otáčky vřetene

Vypočtené napětí, které se zadává pro vřeteno (rozsah 0-7FFFh)

$$U_x = \frac{v}{2 \cdot \pi \cdot x} \cdot U \cdot U_m$$

U_x = vysílané napětí na vřeteno (rozsah 0-7FFFh)
 U = maximální napětí pro vřeteno pro daný převodový stupeň (poměr k max. napětí)
 U_m = maximální napětí (7FFFh)
 P = maximální otáčky dané převodové řady

23.12.1 KŘR-verze 2, řízení z NC programu

Řízení z NC programu je stejné jako pro starší KŘR a platí také stejné modifikace pomocí strojních konstant **R67** a **R329**. Nová vlastnost je, že KŘR není omezena jen na pohybové bloky NC programu. Zařazení KŘR může proto být i v nepohybovém bloku. Také změna délkové korekce a změna posunutí počátku se uplatní okamžitě a nemusí se čekat na pohybový blok. Korekce a posunutí mají vliv na KŘR, protože KŘR se počítá na špičku nástroje a ne na suport (tuto vlastnost možno vypnout – viz dále). KŘR se může také uplatňovat v pomocných ručních pojezdech nebo v jiných druzích pohybu, například vlečení nebo pro pohyby řízené z PLC programu.

Další nová vlastnost je, že aktivní KŘR nikdy nepřepíná převodový stupeň, ale omezí otáčky na maximální otáčky daného převodového stupně (pokud není ještě jiné omezení otáček).

Z NC programu možno kdykoli zadat omezení otáček pro KŘR pomocí funkce **G76** a adresy **S**. Pod adresou **S** se zadávají maximální otáčky v ot/min. Systém omezuje otáčky vzhledem k menší hodnotě z maximálních otáček převodového stupně a ze zadaného omezení pomocí funkce **G76**.

Příklad 1:

```
N100 G96 S500      "zadání KŘR - okamžitý účinek na otáčky vřetene
...
N200 S600           "změna KŘR
```

Příklad 2:

```
N100 G76 S2000      "zadání omezení otáček pro KŘR na 2000 ot/min
```

23.12.2 KŘR-verze 2, řízení z PLC programu

PLC program má k dispozici pole double-wordových hodnot pro sledování stavu KŘR. Na systému je možno jednotlivé hodnoty sledovat v prohlížení paměti systému – paměťová oblast 1.

název	adresa	Popis
AKRR_ACT_V	A178h	Aktuální konstantní řezná rychlost [mm/min] (G96 nebo PLC)
AKRR_ACT_SMAX	A17Ch	Aktuální maximální otáčky [1/1000 ot/min] (G76 nebo PLC)
AKRR_ACT_DIST	A180h	Aktuální korekce poloměru [1/8 μm] (jen PLC)
AKRR_ACT_VCORR	A184h	Aktuální korekce řezné rychlosti [mm/min] (jen PLC)
AKRR_ACT_R	A188h	Aktuální poloměr [1/8μm]
AKRR_ACT_P	A18Ch	Aktuální max.otáčky dle převodové řady [1/1000 ot/min]
AKRR_ACT_U	A190h	Aktuální rozsah napětí dle převodové řady [0-7FFFh]
AKRR_ACT_S	A194h	Aktuální vypočtené otáčky [1/1000 ot/min]
AKRR_ACT_OUT	A198h	Aktuální vypočtené napětí [0-7FFFh]

PLC program má možnost zadávat velikost KŘR, maximální otáčky, korekci poloměru pro výpočet KŘR a korekci řezné rychlosti. Také může KŘR aktivovat a případně modifikovat výpočet. Pro řízení PLC program používá bity umístěné v řídicím bajtu **AKRR_CNTR**.

Název bitu	Váha		popis
EXT_AKRR_V	0	0	Velikost KŘR se řídí z NC programu (G96,G97+ S)
		1	Velikost KŘR zadává PLC program v buňce AKRR_V
EXT_AKRR_SMAX	1	0	Maximální otáčky zadává NC program (G76)
		1	Maximální otáčky zadává PLC program v buňce AKRR_SMAX
EXT_AKRR_DIST	2	0	PLC program nezadává korekci poloměru
		1	PLC program zadává korekci poloměru v buňce AKRR_DIST
EXT_AKRR_VCORR	3	0	PLC program nezadává korekci řezné rychlosti
		1	PLC program zadává korekci řezné rychlosti v AKRR_VCORR
EXT_AKRR_REQ	4	0	Aktivace KŘR se řídí z NC programu (G96,G97)
		1	Aktivace KŘR z PLC programu
EXT_AKRR_SUPORT	5	0	Standardní výpočet KŘR
		1	Výpočet KŘR nezapočítává korekce a posunutí

název	adresa	Popis
AKRR_V	A19Ch	Velikost KŘR zadávaná z PLC [mm/min]
AKRR_SMAX	A1A0h	Maximální otáčky zadávané z PLC [1/1000 ot/min]
AKRR_DIST	A1A4h	Korekce poloměru zadávaná z PLC [1/8 μm]
AKRR_VCORR	A1A8h	Korekce řezné rychlosti zadávaná z PLC [mm/min]