8

8. ROZHRANÍ CNC SYSTÉM - PLC PROGRAM

Komunikační rozhraní CNC systém - PLC program tvoří bloky:

POVELOVÝ BLOK	povel z CNC systému do PLC programu (bude označován `PBxx`)
BLOK ZPĚTNÉHO HLÁŠENÍ	zpětné hlášení z PLC programu do CNC systému (bude označováno `BZHxx`)
KOMUNIKAČNÍ OBLAST	komunikační oblast mezi primárním a sekundárním procesorem
RTM OBLAST	zveřejněné **s**ystémové proměnné z modulu reálného času

8.1 PLC rozhraní

8.1.1 Odměřování a diference

Systém standardně používá několik transformací, které jsou zařazeny na výstup interpolátora. Jednotlivé transformace oddělují „**prostory**“, ve kterých je definovaná aktuální poloha:

Transformace	**Prostor**
Interpolace (fiktivní poloha)	
	POS0
Programová transformace	
	POS1
Transformace polotovaru	
	POS2
Délková korekce	
	POS3
Posunutí 1	
	POS4
Posunutí 2	
	POS5
Transformace stroje (reálná poloha)	
	POS6

Prostor	pole 6x16xQWORD [1/64000 µm]	pole 6xDWORD [1/8 µm]	Popis
POS0	**B_POL_EX, B_INK_EX**	B_POL, B_INK	poloha v POS0
POS1	**POSITION_POS1_EX**		poloha v POS1
POS2	**POSITION_POS2_EX**		poloha v POS2
POS3	**POSITION_POS3_EX**		poloha v POS3
POS4	**POSITION_POS4_EX**		poloha v POS4
POS5	**POSITION_POS5_EX**		poloha v POS5
POS6			poloha v POS6

Prostor	pole 16xDWORD [µm]		Popis
POS6	**DIFCIT_X**		diferenční čítač

PLC program má k dispozici odměřování systému pro zjištění okamžité polohy stroje. Může jej využít například pro řízení mazání souřadnic od polohy stroje. Odměřování je součet bufferu polohy B_POL a bufferu inkrementu B_INK. Odměřování je platné od najetí do reference. Přístup k jednotlivým osám je relativní.

PLC program nesmí do bufferů polohy jednoducho zapisovat, protože při změně je potřeba vždy provést všechny transformace (inverzní nebo přímé) a nastavit všechny polohy stroje.

Pro nastavení polohy souřadnic (například při PLC referenci), slouží buňka HOMING_REQ, která spustí v systému okamžitý výpočet transformací (viz. popis pro nájezd do reference)

Pro přepínání prostorů z PLC programu například při ručních pojezdech slouží buňky POS_SPACE_PLC a POS_SPACE_MMM (viz návod „Ruční pojezdy“).

Příklad:
Hlídání přetečení diference v ose Y podle limitu LIMIT_Y

```
CLI                     ;zákaz přerušení (ASM86)
LOD   DWRD.(DIFCIT_X+4) ;načte diferenci Y
STI                     ;povolení přerušení (ASM86)
ABS   DWRD              ;absolutní hodnota DR 32 bitů
GE    DWRD.LIMIT_Y      ;porovnání
JL1   ERROR_Y           ;skok na chybu
```

Příklad:
Zjistění polohy v ose Y (- jen kladné míry)

```
CLI                     ;zákaz přerušení (ASM86)
LOD   DWRD.(B_INK+4)    ;načte buffer inkrementu Y
AD    DWRD.(B_POL+4)    ;načte buffer polohy Y
STI                     ;povolení přerušení (ASM86)
RR    CNST.3,DWRD       ;převod na mikrony
STO   DWRD.POLOHA_Y     ;zápis do polohy Y
```

Příklad mazání podle ujeté dráhy je v kapitole "Užitečné příklady."

8.1.2 Otáčky vřetene

Dále je uveden seznam důležitých systémových proměnných souvisejících s problematikou vřetene. Podrobně je problematika vřeten popsána v kapitole: „Rotační osy a vřetena“.

název	velikost	popis
PB11, PB12	2xBYTE	(jen pro čtení) Binární hodnota, která reprezentuje výstupní napětí (0 – 7FFFh), které odpovídá programovaným otáčkám vřetene. Hodnota není zkorigována vzhledem k procentu S a neprojeví se ani řízení v průběhu konstantní řezné rychlosti.
VYSOVERS	WORD	(jen pro čtení) Binární hodnota, která reprezentuje výstupní napětí (0 – 7FFFh), které odpovídá skutečným otáčkám vřetene. Hodnota je zkorigována vzhledem k aktuálnímu stavu procenta S, pokud je tato korekce povolena (není programována G33) . V průběhu konstantní řezné rychlosti je velikost hodnoty řízená od interpolátoru v závislosti na poloze osy X.
REQ_SPEED_PM	DWORD	(jen pro čtení) Požadované otáčky vřetene v tisícinách otáčky za minutu. Hodnota zohledňuje aktuální stav procenta S a konstantní řeznou rychlost. jedná se o výsledné požadované otáčky vřetene. Hodnota buňky je vypočtena z aktuálního stavu VYSOVERS, z aktuálního stavu převodového stupně a maximálního napětí pro daný převodový stupeň.
REQ_SPEED_PT	DWORD	(jen pro čtení) Požadované otáčky vřetene v tisícinách stupně za výpočtový takt systému. Hodnota odpovídá buňce REQ_SPEED_PM, jen je v jiných jednotkách.
ACT_SPEED_PM **ACT_SPEED2_PM** **AVG_SPEED_PM** **AVG_SPEED2_PM**	DWORD	(jen pro čtení) Aktuální otáčky vřetene v tisícinách otáčky za minutu. Hodnota je vypočtena z odměřovacího čidla na vřetenu. Při výpočtu je zohledněn vnitřní inkrement vřetene. K dispozici také filtrovaná (průměrná) hodnota aktuálních otáček AVG_SPEED_PM a také hodnoty pro druhé vřeteno: ACT_SPEED2_PM a AVG_SPEED2_PM
ACT_SPEED_PT **ACT_SPEED2_PT** **AVG_SPEED_PT** **AVG_SPEED2_PT**	DWORD	(jen pro čtení) Aktuální otáčky vřetene v tisícinách stupně za výpočtový takt systému. Hodnota je vypočtena z odměřovacího čidla na vřetenu. Při výpočtu je zohledněn vnitřní inkrement vřetene podle strojní konstanty R60. K dispozici také filtrovaná (průměrná) hodnota aktuálních otáček AVG_SPEED_PT a také hodnoty pro druhé vřeteno: ACT_SPEED2_PT a AVG_SPEED2_PT
SIM_SPEED_PT **(OTACKY_VR)**	DWORD	(pro zápis) Systémová proměnná pro aktuální otáčky vřetene v tisícinách stupně za výpočtový takt systému. Buňka se používá při simulaci otáček vřetene.
REQW_SPEED2_PM	DWORD	(pro zápis) Požadované otáčky pro 2. vřeteno pro výstup na obrazovku.

8.1.3 Aktuální převodový stupeň

```
SPINDLE_GEAR_ACT_1      ....Převodový stupeň pro 1.vřeteno

( WORD čtení/zápis )

REQ_EXT_SPINDLE_GEAR    ....Externí řízení převodů z PLC
REQ_MAN_SPINDLE_GEAR    ....Manuální nastavování převodů z PLC

( bity čtení/zápis )
```

Aktuální převodový stupeň pro 1.vřeteno je v buňce **SPINDLE_GEAR_ACT_1** (WORD 1,2,..,32). Pokud se používají maximálně 4 převodové stupně pomocí funkcí M41-M44, zapíše se do buňky SPINDLE_GEAR_ACT_1 příslušný stupeň (1 až 4) přímo ze systému. V jiných případech musí aktuální převodový stupeň do buňky zapsat PLC program. V tomto případě musí PLC žádat o externí řízení převodových stupňů nastavením bitů REQ_EXT_SPINDLE_GEAR a REQ_MAN_SPINDLE_GEAR. Aktuální převodový stupeň má vliv na výpočet výstupné hodnoty pro zadané otáčky vřetene (viz. „Rotační osy a vřetena“).

8.1.4 Povolení pohybu od PLC

```
MPxPI               ....Povolení pohybu

( 16 bitů, čtení/zápis,  x = X,Y,Z,4,5,6,7,...,15,16 )
```

PLC program může povolit nebo zakázat pohyb pro jednotlivé osy bity **MPxPI,** definované v BZH08PI a BZH08PI2.

- log. 1 povolení pohybu z PLC
- log. 0 zákaz pohybu z PLC

Implicitně jsou bity MPxPI nastaveny na hodnotu log. 1, takže je pohyb povolen. Implicitní nastavení se provede v rámci instrukce MODULE_INIT.

V bloku zpětného hlášení jsou bity MPx, které jsou výsledkem všech podmínek povolení pohybu. Systémový software zapisuje do bitu MPx výsledek součinu povolení pohybu od různých zdrojů. Do tohoto součinu je zařazeno povolení pohybu např. od modulu interpolace, od modulu reálního času - kde jsou zařazené i koncové spínače (i softwarové) a od modulu servosmyčky. Do součinu jsou zahrnuty i bity MPxPI, které umožňují blokovat pohyb z uživatelského PLC programu. Supervizor interfejsu umožní pohyb na základě nastaveného bitu MPx do log.1 a na základě ukončení průchodu modulem přípravných funkcí.

PLC program může zakázat pohyb i počas pohybu. V tomto případě se pohyb zastaví dojezdovou rampou. Při nastavení log.1 do MPxPI bude pohyb povolen a souřadnice se rozjede rozjezdovou rampou.

Skutečné povolení pohybu nastane v případě požadavku na pohyb PO_OSxPI (viz dále), nastavením log.1 v MPxPI a celým průchodem modulu přípravné funkce (viz kapitola "Struktura PLC programu"). Pro případ, že PLC program nepotřebuje blokovat pohyb od poruch (blokování pohybu od průchodu PLC programu přípravnými a závěrečnými funkcemi zabezpečuje supervizor interfejsu), nemusí být bity MPxPI v PLC programu vůbec nastavovány, protože mají implicitní hodnotu log.1. Ovládání bitů MPxPI se doporučuje prakticky jen v případě poruch souřadnic, které vznikly počas pohybu.

V pomocných ručních pojezdech je možno pomocí konfigurace zvolit, zda má být také pohyb povolován od bitů MPxPI nebo od bitů MPxMAN (viz. kapitola: „Pomocné ruční pojezdy“)

Příklad:
Zablokujte pohyb v ose Z od poruchy ERROR_Z .

```
LDR   ERROR_Z          ;načte poruchový signál
FL1   0,MPYPI          ;podmíněný zákaz pohybu
```

8.1.5 Povolení pohybu pro pomocné ruční pojezdy

MPxMAN ...Povolení pohybu pro pomocné ruční pojezdy

(16 bitů, čtení/zápis, x = X,Y,Z,4,5,6,7,...,15,16)

PLC program může povolit nebo zakázat pohyb pro jednotlivé osy v pomocných ručních pojezdech, bity **MPxMAN,** definované v BZH08MAN a BZH08MAN2.

Aktivace bitů musí být povolena v konfiguraci. Podrobně je tato problematika popsána v kapitole "Pomocné ruční pojezdy".

8.1.6 Povolení pohybu celkové

MPx ....Povolení pohybu

(6 bitů, čtení, x = X,Y,Z,4,5,6)

Bity **MPx** jsou výsledkem všech podmínek povolení pohybu pro interpolátor. Systémový software zapisuje do bitu MPx výsledek součinu povolení pohybu od různých zdrojů. Do tohoto součinu je zařazeno povolení pohybu např. od modulu interpolace, od modulu reálného času - kde jsou zařazené i koncové spínače (i softwarové) a od modulu servosmyčky. Do součinu jsou zahrnuty i bity MPxPI, které umožňují blokovat pohyb z uživatelského PLC programu. Supervizor interfejsu umožní pohyb na základě nastaveného bitu MPx do log.1 a na základě ukončení průchodu modulem přípravných funkcí.
PLC program nesmí do bitů MPx zapisovat.

8.1.7 Pohyb v osách

PO_OSxPI ....Pohyb v osách

(16 bitů, čtení, x = X,Y,Z,4,5,6,7,...,15,16)

Požadavek na pohyb v osách. Pro jednotlivé souřadnice je v bitech **PO_OSxPI** v PB20PI a PB20PI2**.**

- log. 1 požadavek na pohyb
- log. 0 není požadavek na pohyb

Bit PO_OSxPI je součtem všech požadavků na pohyb. Na rozdíl od PO_OSx v povelovém bloku PB20 (viz dále) obsahuje i pohyb od polohovacích jednotek, pomocných ručních pojezdů a pod. Po skončení pohybu se bit PO_OSxPI vynuluje.

Příklad:
Nastartujte mechanizmus uvolnění osy X - UVOL_X v přípravných funkcích, když je pohyb programován

```
            LDR   PO_OSXPI          ;načte požadavek na pohyb
            JL0   NENI_POX          ;obskok, když neni pohyb
            FL    1,UVOL_X          ;start mechanizmu uvolnění osy X
            EX                      ;
            LDR   UVOL_X            ;čekání na uvolnění
            EX1                     ;
NENI_POX:                           ;
```

8.1.8 Směr pohybu

SM_POxPI ....Směr pohybu servosmyčky

(16 bitů, čtení, x = X,Y,Z,4,5,6,7,...,15,16)

Směr pohybu pro jednotlivé servosmyčky je v bitech **SM_POxPI**

- log. 0 kladný směr pohybu
- log. 1 záporný směr pohybu

Směr pohybu je platný až po skutečném rozjetí pohybu, to znamená až po průchodu modulem přípravných funkcí. Pro případ, že je potřeba řídit nějakou činnost na základě směru pohybu v přípravných funkcích (například sepnutí směrových spojek), je potřeba spustit mechanizmus od průchodu přípravných funkcí a tak se zabezpečí jeho paralelní chod s pohybem. Tento případ je vysvětlen na následujícím příkladu:

Příklad:
Ovládání směrových spojek stroje pro osu Y:

```
Konec modulu MODULE_BLOCK_INIT:

            LDR   PO_OSYPI          ;načtení požadavku na pohyb
            FL1   1,SPOJKA_Y        ;podmíněná aktivace mechanizmu SPOJKA_Y
MODULE_BLOCK_INIT_END               ;konec přípravných funkcí - nečekáme na
                                    ;dokončení

                                    ;mechanizmus SPOJKA_Y
V modulu MODULE_MAIN definován mechanizmus:
MECH_BEGIN  SPOJKA_Y
            EX                      ;začátek pohybu na 1 průchod
            FL    0,MPYPI           ;zákaz pohybu
            LDR   SM_POYPI          ;načte platný směr pohybu
            WR    SPOJ_MINUS_Y_O    ;nastaví zápornou spojku  log.1=minus
            CA                      ;negace RLO
            WR    SPOJ_PLUS_Y_O     ;nastaví kladnou spojku log.1=plus
            LDR   KSOJ_Y_I          ;kontrolní vstup
            EX0                     ;čekání na sepnutí spojky
            FL    1,MPYPI           ;povolení pohybu
MECH_END    SPOJKA_Y                ;konec mechanizmu
```

8.1.9 Vypínání a zapínání polohové vazby

VAZBA_x ....Polohová vazba

(16 bitů, čtení/zápis, x = X,Y,Z,4,5,6,7,...,15,16)

Vypínání a zapínání polohové vazby se řídí bitem **VAZBA_x** v VAZBAPI a VAZBAPI2**.**

- log. 0 vypnutí polohové vazby
- log. 1 zapnutí polohové vazby

Systém má softwarovou polohovou vazbu. PLC program může polohovou vazbu nejen zapínat a vypínat, ale může také nastavovat parametry polohové servosmyčky. Podrobně je tato problematika rozepsána v kapitole "Nastavení parametrů servopohonů a jejich řízení PLC programem".

Implicitně jsou bity VAZBA_x nastaveny do log.1, to znamená že je polohová vazba zapnuta. Implicitní nastavení se provede v rámci instrukce MODULE_INIT. Pokud PLC program nepotřebuje polohovou vazbu vypínat, nemusí bity VAZBA_x vůbec obsluhovat.

Vypnutí polohové vazby by se mělo provádět, když je souřadnice v klidu. Nastavením log.0 do příslušného bitu VAZBA_x se odpojí účinek diferenčného čítače na vysílání rychlosti pro pohon. Odměřování souřadnice zůstává dál plně v činnosti. Pokud nebudou použity ještě jiné prostředky (například přepnutí do indikace nebo nulování diference), není v tomto případě vhodné, aby souřadnice jezdila. Odměřování je stále v činnosti a ujetá dráha by se kumulovala v diferenčním čítači. Po čase by mohlo dojít k jeho přetečení. Při opětovném zapnutí polohové vazby by pohon najednou dostal velký napěťový skok, protože polohová vazba by se snažila vynulovat diferenční odchylku.

Vypnutí polohové vazby je výhodné použít například pro upínání souřadnic a pro případ, když je víc souřadnic řízeno stejným pohonem.

Příklad:
Zapínání polohové vazby při uvolnění souřadnice Z.

```
V přípravných funkcích:
            LDR   PO_OSZPI    ;požadavek na pohyb
            JL0   POZ_E       ;obskok
            FL    1,UVOL_Z    ;start mechanizmu uvolňování
            EX                ;
            LDR   UVOL_Z      ;čekáme na uvolnění
            EX1               ;
            FL    1,VAZBA_Z   ;zapnutí polohové vazby
POZ_E:                        ;v závěrečných funkcích bude vypnutí vazby
                              ;společně s upnutím souřadnice
```

8.1.10 Polohování vřetene

```
POLOHV_x          ....Polohování vřetene

( 16 bitů, čtení/zápis,  x = X,Y,Z,4,5,6,7,...,15,16 )
```

Příznak pro polohování vřetene je v bitu **POLOHV_x** v POLOHV a POLOHVPI2.

- log. 1 rotační souřadnice je ve stavu nájezdu na nulový puls
- log. 0 rotační souřadnice dosáhla nulový puls

Příznak `POLOHV_x` se používá v režimu přepnutí vřetene na rotační souřadnici (polohování rotační souřadnice). Používá se v součinosti instrukce `SPI_AX_x`, která změní rychlostní vazbu rotační souřadnice na polohovou a tato se začne pohybovat dojížděcím posuvem a zastaví se až po dosažení nulového pulsu. Nájezd na nulový puls trvá určitou dobu, a proto PLC program musí pozastavit vykonávání dalších funkcí do dokončení nájezdu. Pro indikování najetí rotační souřadnice slouží bitové proměnné `POLOHV_x` (viz kapitolu"Popis řízení regulátorů pohonů rotačních os a vřeten").

8.1.11 Ruční pojezdy

Ruční pojezdy jsou podrobně popsány v kapitole „Ruční pojezdy“ a zde uvádíme jen přehled nastavovacích a informačních signálů rozhraní:

Proměnná	typ	akce	popis
ACK_AUTMAN	bit	čtení	Pomocné ruční pojezdy aktivní
INPOS_STOP	bit	čtení	Systém je v poloze posledního stopu
EN_AUTMAN	bit	čtení, zápis	Povolení ručních pojezdů (přednastaven na 1)
AUTMAN_CONT_NC	bit	čtení	Řízení ručních pojezdů z panelu systému
AUTMAN_CONT_TOC	bit	čtení	Řízení ručních pojezdů z panýlku točítka
AUTMAN_CONT_SELECT	bit	čtení	Provádí se předvolba pohybu
AUTMAN_CONT_G00	bit	čtení	Požadován rychloposuv
EXM_x0	bity	zápis	Externí požadavek na pohyb v kladném směru
EXM_x1	bity	zápis	Externí požadavek na pohyb v záporném směru
REQ_EXT_G00_AUTMAN	bit	zápis	Externí požadavek na rychloposuv
REQ_EXT_SELECT_AUTMAN	bit	zápis	Externí požadavek na předvolbu
REQ_EXT_CONT_AUTMAN	bit	zápis	Externí požadavek na řízení pohybu
REQ_EXT_FEED_AUTMAN	bit	zápis	Externí požadavek na zadání rychlosti
REQ_EXT_G00_AUTMAN	bit	zápis	Externí požadavek na zadání rychloposuvu
EXT_FEED_AUTMAN	word	zápis	Externí zadání rychlosti [mm/min]
EXT_FEED_G00_AUTMAN	word	zápis	Externí zadání pro rychloposuv [mm/min]
AUTMAN_REQ	bit	zápis	Externí požadavek na aktivaci ručních pojezdů

8.1.12 Stop z PLC programu

```
STOPPI              ....Stop z PLC programu
STOP_REQ
STOP_EMERGENCY_REQ

( bity, čtení/zápis )
```

Bit **STOPPI** hodnotou log.1 vyvolá STOP obdobně jako stisknutí tlačítka STOP na ovládacím panelu. Nastavením bitu STOPPI se vyvolá stop celého bloku. Systém reaguje na nástupní hranu signálu.
Stav bitu STOPPI je potřeba v PLC programu vrátit do stavu log.0, to se provede například odčasováním. Minimální délka pulsu signálu STOPPI je asi 200 ms.

Bit **STOP_REQ** je systémem potvrzovaný stop. PLC program hodnotou log.1 vyvolá STOP podobně jako u signálu STOPPI, ale systém po převzetí požadavku bit STOP_REQ sám vynuluje. PLC program se nemusí starat o nulování bitu.

Bit **STOP_EMERGENCY_REQ** je systémem potvrzovaný nouzový stop. PLC program hodnotou log.1 vyvolá nouzový STOP a systém po převzetí požadavku bit STOP_EMERGENCY_REQ sám vynuluje. Nouzový stop používá jiné nastavení ramp pro zastavení.

Přehled použitých ramp z konfigurace pro zastavení při stopu v závislosti na druhu provozu:

Provoz	-	STOP	MPxPI	STOP EMERGENCY
Parabolické rampy AccelerationType="1"	ParabAccel LinearAccel	ParabAccelRT LinearAccelRT	okamžité zastavení	ParabAccelEmergency LinearAccelEmergency
Lineární rampy AccelerationType="0"	Acceleration	AccelerationRT	Acceleration Emergency	
Ruční pojezdy	Acceleration	Acceleration	Acceleration Emergency	AccelerationEmergency

Signály používá PLC program, když nastane v průběhu vykonávání bloku chyba. Například při chybě v přípravných funkcích se zastaví další chod pomocí "nekonečné instrukce typu EX", nastavíme bit STOPPI a supervizor interfejsu přeruší vykonávání bloku.

Příklad:
Ošetření chyby v přípravných funkcích:

```
            LDR   TLAK_I                        ;načte vstup o tlaku
            TEX0  CITAC_TL,CAS_ERROR,ERROR,15h  ;časová kontrola tlaku
            ......
            ......
ERROR:      ESET                                ;nastavení chybového hlášení
            FL    1,STOPPI                      ;STOP z PLC programu
            EX
            LDR   CAPI                          ;"nekonečný stav"
            EX0                                 ;zabránění dalšímu výkonu
                                                ;přípravných funkcí
```

8.1.13 Blokování a pozdržení startu

```
START_DISABLE          ....Blokování startu
START_SUSPEND          ....Pozdržení startu

( bit, čtení/zápis )
```

Je-li bit **START_DISABLE** nastaven do log.1, nelze na systému nic odstartovat.

Blokování se týká všech existujících způsobů startu. Bit START_DISABLE se využije například, když je nutno zakázat i malý případný pohyb, který by mohl vzniknout při okamžitém stopu po odstartování bloku.

Po startu systému je bit START_DISABLE přednastaven do log.0.

Při vykonávání STARTu se nejdřív zavolá v PLC programu událostní procedura _ON_EVENT se vstupním kódem PLCEVENT_START_REQ (viz. „Základní instrukce jazyka Technol“) a bit **START_SUSPEND** se přednastaví na log.0. Pokud PLC program potřebuje pozdržet průběh startu systému, tak hned v událostní proceduře nastaví bit START_SUSPEND na hodnotu log.1. Průběh startu bude pokračovat až když PLC program nastaví START_SUSPEND na hodnotu log.0.

8.1.14 Blok rozpracován

```
PO_F                   ....Potvrzení funkcí

( bit, čtení )
```

Bit **PO_F** hodnotou log.1 potvrzuje převzetí funkcí a pohybu. Je řízen supervisorem interfejsu a na panelu systému se podle něj řídí druhá signálka: Funkce rozpracovány. Po vykonání všech funkcí, to znamená na konci bloku, bit PO_F je shozen do log.0. Při stopu, když blok není ukončen zůstává bit PO_F ve stavu log.1. PLC program může bitem PO_F zjišťovat, zda je stroj v klidu a podle toho řídit případné další akce.

8.1.15 Dosažení zadané polohy

```
PO_POL                 ....Potvrzení dosažení polohy

( bit, čtení )
```

Bit **PO_POL** hodnotou log.1 potvrzuje, že byla dosažena zadaná poloha souřadnic. Bit nastavují moduly interpolace v kazetě systému. Bit je nahozen až po dosažení zadané tolerance polohy (MaxDifference). Po stopu pohybu se bit PO_POL nenastaví, protože naprogramovaná poloha nebyla dosažena.

8.1.16 Potvrzení stopu

PO_STOP ....Potvrzení stopu

(bit, čtení)

Bit **PO_STOP** hodnotou log.1 potvrzuje vykonání stopu. Uživatelský PLC program smí tento bit pouze číst, nastavení provádí supervisor PLC programu.
Bit PO_STOP nabývá stavu log.1 jen tehdy, je-li proveden stop běžícího programu a ne je-li pouze stlačeno tlačítko STOP v době, kdy není co stopovat.
Stop může byt vyvolán různým způsobem, například také signálem STOPPI z PLC programu.

Bit PO_STOP se po stopu dostane do stavu log.1 a v něm zůstane až po nový start systému, kdy se vrátí do stavu log.0. Nejedná se o kopírování tlačítka STOP.

Bit je pro PLC program vhodný pro zjištění, zda je systém ve stavu STOP .

8.1.17 Pohyb ukončen

INPOS ....Poloha v toleranci

(bit, čtení)

Je-li bit **INPOS** ve stavu log.1, je poloha os v toleranci. Bit je řízen programem interpolace, není-li v žádné ose polohová odchylka serva větší než je dovoleno pro jednotlivé osy ve konfiguraci (MaxDifference)..

Rozdíl mezi bity PO_POL a INPOS je ten, že u bitu INPOS se netestuje, zda byla dosažena programovaná poloha. To znamená, že bit INPOS se nastaví do log.1 i při stopu pohybu, pokud byla dosažena tolerance polohové odchylky.

Bit INPOS je pro PLC program užitečný, když PLC program musí zjistit, zda se vykonává pohyb os. PLC program nesmí bit INPOS nastavovat.

8.1.18 RTM chyby

BZH13 ....Číslo chyby z RTM

(bajt, čtení)

PLC program může zjistit čísla chyb z reálného času. Jedná se vždy o kritické chyby řízení servosmyčky, chyby odměřování, chyby CAN-BUSu a pod.
PLC program může buňku pouze číst a na základe nenulové hodnoty například deaktivovat pohony.

8.1.19 Řízení procenta rychlosti z PLC

```
FEED_OVR          .....Řízení procenta rychlosti z PLC

( bit, čtení/zápis )


BZH09, BZH10      .....Volba promile pro nastavení rychlosti

( 2 bajty, čtení/zápis )


FEED_OVR_MULTIPLIER ...Násobící koeficient (promile) pro rychlost

( word, čtení/zápis )
```

Bit **FEED_OVR** hodnotou v log.1 provede odstavení potenciometru %F na hlavním ovládacím panelu a informaci o nastaveném procentu rychlosti přebírá z bajtů **BZH09** a **BZH10**.
Do bajtů BZH09 a BZH10 se zapisuje promile rychlosti ve formátu šestnáctibitového slova BIN.
Osa pojede programovanou rychlostí, bude-li v BZH09 a BZH10 hodnota 1000.

Bit FEED_OVR se využije při dálkovém ovládání, například z pomocného ovládacího panelu.

Systém nekontroluje hodnoty zapsané do BZH09 a BZH10, souřadnice však nepojede v žádné ose větší rychlostí než je zadaná v konfiguraci pro rychloposuv.

Použití násobícího koeficientu **FEED_OVR_MULTIPLIER** je jinou možností, jak PLC program může ovlivňovat rychlost. Jedná se o wordovou buňku, přednastavenou na hodnotu 1000. PLC program změnou hodnoty mění rychlost s citlivostí na promile, přitom zůstává funkční i standardní potenciometr %F.

$$F_k = F_z \times \frac{(\%F)}{100} \times \frac{FEED\_OVR\_MULTIPLIER}{1000}$$

F_k ..... rychlost korigovaná
F_z ..... rychlost zadaná

Příklad:
Řízení rychlosti z PLC programu.

```
FL    1,FEED_OVR        ;nastavení požadavku na řízení rychlosti z PLC
LOD   PROMILE           ;promile požadované rychlosti
STO   WORD.BZH09        ;zápis do BZH09 a BZH10
```

8.1.20 Řízení procenta otáček z PLC

SPEED_OVR .....Řízení procenta otáček z PLC

(bit, čtení/zápis)

SPEED_OVR_EXT .....Volba promile pro řízení otáček

(2 bajty, čtení/zápis)

SPEED_OVR_MULTIPLIER ...Násobící koeficient (promile) pro otáčky

(word, čtení/zápis)

Bit **SPEED_OVR** hodnotou v log.1 provede odstavení potenciometru %S na hlavním ovládacím panelu a informaci o nastaveném procentu rychlosti přebírá z bajtů **SPEED_OVR_EXT**.
Do bajtů SPEED_OVR_EXT se zapisuje promile rychlosti ve formátu šestnáctibitového slova BIN.

Použití násobícího koeficientu **SPEED_OVR_MULTIPLIER** je jinou možností, jak PLC program může ovlivňovat otáčky. Jedná se o wordovou buňku, přednastavenou na hodnotu 1000. PLC program změnou hodnoty mění otáčky s citlivostí na promile, přitom zůstává funkční i standardní potenciometr %S.

$$S_k = S_z \times \frac{(\%S)}{100} \times \frac{SPEED\_OVR\_MULTIPLIER}{1000}$$

S_k ..... rychlost korigovaná
S_z ..... rychlost zadaná

8.1.21 Prodleva

PRODLEVA ....Signálka prodleva

(bit, čtení/zápis)

Nastavení bitu **PRODLEVA** do log.1 způsobí rozsvícení signálky "PRODLEVA" na ovládacím panelu. Signálka "PRODLEVA" se rozsvítí, je-li programována časová prodleva pomocí G04.

8.1.22 Restart

```
BLOCK_RESTART     ....Restart bloku

( bit, čtení )


STATUS_RESTART    .... Stav procházeni bloku při Restartu

( bajt, čtení )
```

Pod „restartem bloku“ se rozumí opětovný start bloku po předchozím stopu programu. Stav procházení bloku při restartu může nabývat hodnot:

BLOCK_IDLE	0	Nečinnost
BLOCK_INIT	1	Přípravné funkce
BLOCK_MOVE	2	Pohyb
BLOCK_DELAY	3	Prodleva
BLOCK_DONE	4	Závěrečné funkce

8.1.23 Reference

Problematika referencí je podrobně popsána v kapitole „Způsoby reference“ a zde uvádíme jen přehled nastavovacích a informačních signálů rozhraní:

Proměnná	typ	akce	popis
HOMING	WORD bity os	čtení/zápis	Jednotlivé bity jsou příznaky pro platnou referenci NC os. Bity testuje systém v případě, že NC osa má v konfiguraci nastaveny atributy „NeedRef“ nebo „AutNeedRef“.
HOMING_REQ	WORD bity os	čtení/zápis	Jednotlivé bity jsou žádost o nastavení nulového bodu v prostoru POS5 z PLC. Nulový bod se nastavuje podle atributu „MachineNullPoint“. Systém po nastavení provede všechny zpětné transformace.
REFPOINT_DIST	16x QWORD	čtení/zápis	Posunutí od nulového bodu stroje při zadání HOMING_REQ [1/64000 μm]. Nastavuje PLC.
HOMING_START_REQ	WORD bity os	čtení/zápis	Jednotlivé bity jsou žádost z PLC o vykonání kompletní reference pro jednotlivé osy. Po provedení reference systém bit vynuluje. Reference se provede podle aktuální konfigurace.
HOMING_SINGLE	bit	čtení/zápis	Žádost o referenci jedné osy v závislosti na konfiguraci „RefMethod“. Bit nastavuje systém.
HOMING_PLC	bit	čtení/zápis	Žádost o referenci jedné osy pro PLC. PLC bit pro převzetí vynuluje. Bit nastavuje systém.
HOMING_GROUP	bit	čtení/zápis	Žádost o skupinovou referenci pro PLC. PLC bit pro převzetí vynuluje. Bit nastavuje systém.
SLS_REFER	WORD bity os	čtení/zápis	Jednotlivé bity povolují test na softwarové limitní spínače. Každý bit slouží pro jednu NC osu.
NlcAxisEnable	WORD bity os	čtení/zápis	Jednotlivé bity povolují nelineární softwarové korekce a tepelnou kompenzaci. Každý bit slouží pro jednu řídicí NC osu pro nelineární korekce.

NlcServoEnable	WORD bity serv	čtení/zápis	Jednotlivé bity povolují nelineární softwarové korekce a tepelnou kompenzaci. Každý bit slouží pro jednu kompenzovanou servosmyčku. Systém bity po zapnutí přednastaví na hodnotu log.1.
KRx	16 bitů	čtení/zápis	referenční spínače
KRRx	16 bitů	čtení/zápis	reverzační spínače
ZPRx	16 bitů	čtení/zápis	zpomalovací referenční spínače

8.1.24 Limitní a zpomalovací spínače

```
KHx0              ....Limitní spínače v kladném směru
KHx1              ....Limitní spínače v záporném směru
ZPx0              ....Zpomalovací limitní spínač
ZPx1              ....Zpomalovací limitní spínač

(16 bitů, čtení/zápis,  x = X,Y,Z,4,5,6,7,...,15,16 )
```

Hodnota log.1 v **KHx0** nebo v **KHx1** způsobí zastavení pohybu v dané ose. Zastavení řídí interpolátor dojezdovou rampou podle konfigurace „AccelerationEmergency", nebo v případe parabolických ramp podle sady pro rychloposuv „DynamicControlRT".

Vyjetí z limitního spínače je možné pouze v pomocných ručních pojezdech. V režimu AUT způsobí najetí na jakýkoliv limitní koncový spínač zastavení pohybu. Jestliže by se signál KHx0 nebo KHx1 opět vrátil do stavu log.0, pohyb souřadnice bude pokračovat (není samodrž). Případnou samodrž musí zabezpečit PLC program.
Hodnota log.1 v bitech **ZPx0** a **ZPx1** způsobí přechod na zpomalovací rampu. až pokud nebude dosažena velikost zpomalovacího posuvu. CNC systém u zpomalovacích limitních spínačů nevyhodnocuje směr. Když je potřeba vyhodnocovat směr pohybu, musí to provést PLC program na základě signálů SM_POxPI.

8.2 Povelový blok – kompatibilní verze s CNC8x9

Tato verze povelového bloku je uvedena jen pro usnadnění přechodu ze starších PLC programů pro systém CNC8x9 a CNC8x6 na novější (Windowskou) verzi systému CNC872. Při tvorbě nového PLC programu pro systém CNC872 se **nedoporučuje používat !**

8.2.1 Přehled použitých M funkcí podle skupin

```
Skupina 1:  M00   programový stop   (M00_PID, M01_PID, M02_PID, M30_PID)
            M01   volitelný stop
            M02   konec partprogarmu
            M30   konec partprogramu

Skupina 2:  M03   start vřetene CW             (M03PI, M04PI, M19PI a PB06)
            M04   start vřetene CCW
            M05   stop vřetene
            M19   stop vřetene v orientovaném bodě

Skupina 3:  M41   otáčky vřetene - rozsah 1    (M41PI, M42PI, M43PI, M44PI)
            M42   otáčky vřetene - rozsah 2
            M43   otáčky vřetene - rozsah 3
            M44   otáčky vřetene - rozsah 4
            M40   automatická převodovka

Skupina 4:  ...   Funkce dle konfigurace „MFunctions"

Skupina 5:  M07   zapnutí chlazení 1      (M07PI, M08PI)
            M08   zapnutí chlazení 2
            M09   vypnutí chlazení 1 a 2
            M17   zapnutí chlazení 1 a 2

Skupina 6:  M50   zapnutí chlazení 3      (M50PI, M51PI)
            M51   zapnutí chlazení 4
            M52   zapnutí chlazení 3 a 4
            M53   vypnutí chlazení 3 a 4

Skupina 7:  M10   upnutí obrobku          (M10PID, M11PID)
            M11   uvolnění obrobku

Skupina 8:  M48   zrušení feed-overide
            M49   aktivace feed-overide

Skupina 9:  M06   Výměna nástroje         (M06PID a M60PI)
            M60   Výměna obrobku

Skupina 10: ...   Funkce dle konfigurace „MFunctions" (PB03)

Skupina 11: ...   Funkce dle konfigurace „MFunctions" (PB13)

Skupina 12: ...   Funkce dle konfigurace „MFunctions" (PB14)

Skupina 13: ...   Funkce dle konfigurace „MFunctions" (PB15)

Skupina 14: ...   Ostatní funkce                      (PB16)
```

8.2.2 Režim

```
AUTPI              ....Režim AUT
REFPI              ....Režim nájezdu do reference
CAPI               ....Režim centrální anulace

( bity, čtení )
```

Bit **AUTPI** signalizuje hodnotou log.1, že byl odstartován blok v automatickém režimu (AUT), včetně všech možností modifikace (BB, AVP, M01, ND a LOM). Bit se vysílá jen po startu bloku a není průběžně aktualizován. Případná změna režimu se projeví až novým startem bloku, například odstartováním centrální anulace.

Bit **REFPI** signalizuje hodnotou log.1, že byl odstartován nájezdu do reference. Bit je nastavován při skutečném nájezdu do reference a při pseudoreferenci. V případě simulace reference a nulování reference nastavován není.

Bit **CAPI** signalizuje hodnotou log.1, že byl odstartován blok v režimu centrální anulace. Bit se vysílá jen po startu bloku a není průběžně aktualizován. Případná změna režimu se projeví až novým startem bloku.

8.2.3 Závitování

```
G33PI              ....Závitování

( bit, čtení )
```

Bit **G33PI** indikuje hodnotou log.1, že v bloku je programováno závitování funkcí `G33`. Bit `G33PI` využívají systémové prostředky interpolátoru v RTM.

8.2.4 Rychloposuv

```
G00PI              ....Rychloposuv

( bit, čtení )
```

Bit **G00PI** signalizuje hodnotou log.1, že byl odstartován blok, ve kterém je programován rychloposuv. Bit se vysílá jen po startu bloku a není průběžně aktualizován. Případná změna režimu se projeví až novým startem bloku.
Bit `G00PI` se dá využít například pro uvolnění zpomalovacích limitních spínačů, a tak se zabezpečí, že nebudou reagovat pro pracovní posuv, ale jen pro rychloposuv.

8.2.5 BCD funkce v povelovém bloku

V další části budou popsány funkce, které vystupují v povelovém bloku v BCD tvaru. Jedná se o všechny skupiny M-funkcí a funkce S, H. Změna v BCD hodnotách u těchto funkcí je vždy doprovázena příslušným změnovým bitem.

```
PB03              ....M funkce skupiny 10
PB05              ....H funkce
PB07-PB10         ....T funkce
PB06              ....M funkce skupiny 2
PB13              ....M funkce skupiny 11
PB14              ....M funkce skupiny 12
PB15              ....M funkce skupiny 13
PB16              ....M funkce skupiny 14

( bajty, čtení )
```

V buňkách je BCD kód M funkce ze příslušné skupiny. Funkce sdružené do skupin 10,11,12,13 a 14 jsou určeny k volnému použití. Do příslušné skupiny jsou zařazeny funkce, které jsou uvedeny v konfiguraci podle atributu „GroupNo“:

Příklad:
Zařazení funkce M92 do skupiny 12:

```
<MFunctions>
      <MFunc No="92" GroupNo="12" BreakCont="1"></MFunc>
```

8.2.6 Změnové bity

```
ZMSHPI            ....změnový bit H funkce
ZMSTPI            ....změnový bit T funkce
ZMMxPI            ....změnoví bit M funkce x.skupiny

( bity, čtení,  x = 1,2,3,4,...,14 )
```

Změnové bity se nastavují při změně hodnoty příslušné funkce a zůstanou nastaveny po dobu trvání bloku.
Systém nastavuje změnové signály ve dvou případech:

- Je změna v dané technologické funkci (BCD výstup nebo dekódované funkce). Změna vznikla naprogramováním nové hodnoty, která je jiná než předcházející hodnota.
- Není změna hodnoty v dané technologické funkci, ale tato byla opět v bloku programována.

Změnové signály se v PLC programu testují hlavně v přípravných a závěrečných funkcích a na základě nich se až dekódují BCD výstupy v povelovém bloku nebo dekódované funkce. Aby nedocházelo ke zpomalení průchodu modulem přípravných a závěrečných funkcí, první instrukce typu EX se napíše až za obskokem, jak je patrné na příkladu:

Příklad:
Zapnutí vřetena (pokud je ovládáno pouze z NC programu).

```
LDR   ZMM2PI      ;test změnového signálu 2. skupiny M funkcí
JL0   VRETENO_END ;skok na konec
LDR   M03PI       ;test funkce M03
JL0   SKOKM4      ;skok na další testy
FL    1,CW        ;start mechanizmu CW
EX                ;
LDR   CW          ;čekání na dokončení mechanizmu CW
EX1               ;
JUM   VRETENO_END ;skok na konec
```

8.2.7 Dekódované funkce 1. skupiny M

```
M00_PID           ...Dekódovaná funkce M00
M01_PID           ...Dekódovaná funkce M01
M02_PID           ...Dekódovaná funkce M02
M30_PID           ...Dekódovaná funkce M30

( bity, čtení )
```

Bit **M00_PID** se nastaví do log.1 na dobu jednoho bloku, je-li v bloku programována funkce M00.
Bit **M01_PID** se nastaví do log.1 na dobu jednoho bloku, je-li v bloku programována funkce M01. Jestli se funkce M01 v bloku skutečně uplatní, závisí od toho, zda je aktivní modifikace M01 režimu AUT.
Bit **M02_PID** se nastaví do log.1 na dobu jednoho bloku, je-li v bloku programována funkce M02.
Bit **M30_PID** se nastaví do log.1 na dobu jednoho bloku, je-li v bloku programována funkce M30.

8.2.8 Dekódované funkce 2.skupiny M

```
M03PI             ....Dekódovaná funkce M03
M04PI             ....Dekódovaná funkce M04
M19PI             ....Dekódovaná funkce M19

( bity, čtení )
```

Dekódovaný bit **M03PI** je statický a je určen pro roztočení vřetena ve směru točení hodinových ručiček. Je-li v bloku NC programu programována funkce M03, je nastaven bit M03PI do log.1. Tento stav setrvá do doby, než je jinou funkcí z 2. skupiny M změněn.
Funkce M03 nastaví v BCD tvaru buňku PB06 na hodnotu 03h.

Dekódovaný bit **M04PI** je statický a je určen pro roztočení vřetena proti směru točení hodinových ručiček. Je-li v bloku NC programu programována funkce M04, je na počátku výkonu tohoto bloku nastaven bit M04PI do log.1. Tento stav setrvá do doby, než je jinou funkcí z 2. skupiny M změněn.
Funkce M04 nastaví v BCD tvaru buňku PB06 na hodnotu 04h.

Dekódovaný bit **M19PI** je statický a slouží pro programování zapolohování vřetena. Celé zapolohování vřetena musí provést PLC program. Bit M19PI se nastaví na hodnotu log.1, když je funkce M19 v bloku programována a tento stav trvá do doby, než je jinou funkcí z 2. skupiny M změněn.
Funkce M19 nastaví v BCD tvaru buňku PB06 na hodnotu 19h.

Je-li v bloku NC programu programována funkce M05, mají bity M03PI, M04PI a M19PI hodnotu log.0 a buňka PB06 je nastavena na hodnotu 05h.

8.2.9 Dekódované funkce 3.skupiny M

```
M41PI              ....Dekódovaná funkce M41
M42PI              ....Dekódovaná funkce M42
M43PI              ....Dekódovaná funkce M43
M44PI              ....Dekódovaná funkce M44

( bity, čtení )
```

Dekódované bity **M41PI, M42PI, M43PI a M44PI** jsou statické a nastavují se podle funkcí M41, M42, M43 a M44. Funkce M41 je určena k zařazení prvního převodového stupně vřetene. Funkce M42, M43 a M44 je určena k zařazení druhého, třetího a čtvrtého převodového stupně vřetene.
Je-li v bloku NC programu programována funkce M41, je v bloku nastaven bit M41PI do log.1. Je-li v některém dalším bloku programu programována některá z funkci M42, M43, M44, je v bloku shozen bit M41PI do log.0 a nahozen příslušný bit M42PI, M43PI nebo M44PI podle převodového stupně. CNC systém kontroluje, zda zadané otáčky odpovídají převodovému stupni podle konfigurace .

Funkce M40 je automatická převodovka a způsobuje, že systém automaticky nastavuje proměnné M41PI, M42PI, M43PI, M44PI v závislosti na programované funkci S. Podrobněji je o problematice řízení vřetena pojednáno v kapitole "Zadávání otáček vřetena".

8.2.10 Dekódované funkce 5.skupiny M

```
M07PI              ....Dekódovaná funkce M07
M08PI              ....Dekódovaná funkce M08

( bity, čtení )
```

Funkce	M07PI	M08PI
M07 zapnutí chlazení 1	1	0
M08 zapnutí chlazení 2	0	1
M09 vypnutí chlazení 1 a 2	0	0
M17 zapnutí chlazení 1 a 2	1	1

Dekódovaný bit **M07PI** je statický a je určen pro zapnutí chlazení 1. Je-li v bloku NC programu programována funkce M07 nebo M17, je v bloku nastaven bit M07PI do log.1. Je-li v bloku programována funkce M08 nebo M09, je bit M07PI shozen do log.0.

Dekódovaný bit **M08PI** je statický a je určen pro zapnutí chlazení 2. Je-li v bloku NC programu programována funkce M08 nebo M17, je v bloku nastaven bit M08PI do log.1. Je-li v bloku programována funkce M07 nebo M09, je bit M08PI shozen do log.0.

8.2.11 Dekódované funkce 6.skupiny M

```
M50PI              ....Dekódovaná funkce M50
M51PI              ....Dekódovaná funkce M51

( bity, čtení )
```

Funkce	M50PI	M51PI
M50 zapnutí chlazení 3	1	0
M51 zapnutí chlazení 4	0	1
M52 zapnutí chlazení 3 a 4	1	1
M53 vypnutí chlazení 3 a 4	0	0

Dekódovaný bit **M50PI** je statický a je určen pro zapnutí chlazení 3. Je-li v bloku NC programu programována funkce M50 nebo M52, je v bloku nastaven bit M50PI do log.1. Je-li v bloku programována funkce M51 nebo M53, je bit M50PI shozen do log.0.

Dekódovaný bit **M51PI** je statický a je určen pro zapnutí chlazení 4. Je-li v bloku NC programu programována funkce M51 nebo M52, je v bloku nastaven bit M51PI do log.1. Je-li v bloku programována funkce M50 nebo M53, je bit shozen M51PI do log.0.

8.2.12 Dekódované funkce 7.skupiny M

```
M10PID             ....Dekódovaná funkce M10
M11PID             ....Dekódovaná funkce M11

( bity, čtení )
```

Bity **M10PID** a **M11PID** jsou dynamické a nastavují se jen po dobu trvání jednoho bloku. Nastavením hodnoty log.1 se určuje, že v bloku byla programována funkce upnutí obrobku M10 nebo funkce uvolnění obrobku M11.

8.2.13 Dekódované funkce 9.skupiny M

```
M06PID             ....Dekódovaná funkce M06
M60PI              ....Dekódovaná funkce M60

( bity, čtení )
```

Bity **M06PID a M60PI** jsou dynamické s platností jednoho bloku. Nastavením hodnoty log.1 se určuje, že v bloku je programována funkce pro výměnu nástroje M06 nebo M60.

8.3 Povelový blok – verze pro CNC872

Pro verzi systému CNC872 (Windows) možno použít všechny prvky PLC rozhraní popsány dříve. Místo starší kompatibilní verze povelového bloku se doporučuje používat novější verzi pro CNC872.

8.3.1 Bitové proměnné povelového bloku

Pro načtení bitových proměnných se doporučuje používat instrukce `LDR, LA LO` a `LX` s prefixem „`BLK[xx]`" (viz dále). Všechny bity může PLC program jenom číst.

Název bitu	Popis
M0,M1,...,M14	Změnové bity pro jednotlivé skupiny M funkcí
FirstBlock	První blok NC programu
LastBlock	Poslední blok NC programu
Stop	Programován stop v NC programu (M00)
CondStop	Podmínění stop v NC programu (M01)
Backward	Jízda nazpátek
SelBlock	Volba bloku
Cont	Blok s přískokem z místa mimo dráhu
TchlOff	Blok se má jet bez technologie
Partial	Blok jede odprostřed, (stop a opětovný start)
HProg	Změnový bit pro H funkci
TProg	Změnový bit pro T funkci
RevFeed	Programována rychlost v mm/otáčku (G95)
ContinuousMode	Programována plynulá jízda (G23)
ConstCutSpeed	Programována konstantní řezná rychlost (G96)
AxNC0Move **AxNC1Move** **AxNC2Move,...**	Příznaky pohybu pro jednotlivé NC osy
IPlc0 **IPlc1** **IPlc2,...**	Změnové bity pro celočíselné programované parametry pro PLC
RPlc0 **RPlc1** **RPlc2,...**	Změnové bity pro reálné programované parametry pro PLC

8.3.2 Přístup k prvkům bloku použitím prefixu „BLK[xx]“

Pro zjednodušení přístupu PLC programu k prvkům struktury aktuálního bloku nebo příštích bloků slouží zvláštní prefix **„BLK[xx].“** pro instrukce **LDR,LA,LO** a **LX**. V hranaté závorce prefixu se zadává pořadové číslo z fronty připravených bloků. Pořadové číslo „0“ znamená přístup k aktuálnímu bloku, „1“ je pro první příští blok a pod. PLC program má možnost zjistit aktuální počet připravených bloků v proměnné **„CNT_NEXT_BLOCKS“** (typ WORD), která se aktualizuje při startu průchodu modulu „MODULE_BLOCK_INIT“. V hranaté závorce prefixu může být zadán také index-registr „BX“.

Příklad:
Použití instrukcí LDR, LA, LO a LX s prefixem BLK[xx]:

```
LDR   BLK[0].M3         ;načtení změnového bitu 3.skupiny M
JL1   JE_NOVA

LDR   BLK[0].M2         ;načtení změnového bitu 2.skupiny M
LA    BLK[0].FirstBlock ;jen z 1. bloku programu
JL1   JE_NOVA

LDR   BLK[1].TchlOff    ;Je příští blok s vypnutou technologií ?
JL1   TCH_OFF

LDR   BLK[BX].IPlc1     ;Je změna 2.celočíselného PLC parametru ?
JL1   PLC1_CHANG        ;(ve frontě příštích bloků podle BX)
```

8.3.3 Celočíselné proměnné povelového bloku

Pro práci s celočíselnými (DWORD) proměnnými je možno používat instrukce **LOD, AD, SU, ORB, ANDB, XORB MULB, DIVB, EQ, LE, LT, GE a GT** s prefixem **„BLK[xx]“**. Všechny buňky může PLC program jenom číst. Pro pole je možno použít indexaci, přitom index je číslován po DWORDech od 0 (0,1,2,.. jen pro typ DWORD). U instrukcí je povoleno také používat přetypování, pokud požadujeme jiný typ něž DWORD (přetypování musí být uvedeno jako další prefix za prefixem BLK[xx]).

Název	Popis
M	Programované M funkce jednotlivých skupin, pole 15xDWORD
IPlcParams	Programované celočíselné parametry, pole 5xDWORD
T	Programovaná hodnota adresy T DWORD
AxG	Definice geometrických os (definované jako index do pole NC os) pole 3xDWORD

Příklad:
Použití instrukce `LOD (AD,SU,ORB,ANDB,XORB,MULB,DIVB,EQ,LT,GE,LE,GT):`

```
LDR   BLK[0].M11              ;změnový bit 11.skupiny M
JL0   NENI_NOVA
LOD   BLK[0].M[11]            ;načte hodnotu 11.skupiny M (DWORD)
EQ    CNST.95h                ;test na M95
WR    FUN_95

LOD   BLK[0].IPlcParams[2]    ;načte 3.celočíselný PLC parametr
...
EQ    BLK[0].WORD.T           ;porovnání s hodnotou T jako WORD
...
LOD   BLK[0].BYTE.M[12]       ;načte hodnotu 12.skupiny M (BYTE)
...
```

8.3.4 Reálné proměnné povelového bloku

Pro práci s reálnými proměnnými je možno používat instrukce **`LOD, AD, SU, MULB, DIVB, EQ, LE, LT, GE a GT`** s prefixy **„`BLK[xx]`"** a **„`REAL`"**. Všechny buňky může PLC program jenom číst. Pro pole je možno použít indexaci, přitom index je číslován po reálných typech 0 (0,1,2,.. jen pro typ REAL). U instrukcí je nutno použít přetypování na „`REAL`" a přetypování musí být uvedeno jako další prefix za prefixem `BLK[xx].`

Název	Popis
`H`	Programovaná hodnota adresy H QWORD FLOATING
`RPlcParams`	Programované reálné parametry, pole 5x QWORD FLOATING
`Delay`	Časová prodleva programovaná v bloku QWORD FLOATING
`AxNC`	Absolutní poloha pro NC osy v Pos 0 [mm] pole 16x QWORD FLOATING
`BlockLen`	Délka dráhy bloku v Pos 0 [mm]. Pro přepočet na skutečnou délku dráhy v Pos 5 lze použít měřítko (proměnná Scale). QWORD FLOATING
`Scale`	Měřítko bloku počítané z programové trasformace a transformace polotovaru. Měřítko je platné pouze v případě, že transformace neobsahují změnu měřítka různou v jednotlivých osách. QWORD FLOATING
`FeedProgr`	Programovaná rychlost suportu pro pracovní posuv [mm/min] QWORD FLOATING
`FeedMax`	Maximální rychlost suportu.[mm/min] QWORD FLOATING
`FeedCriterial`	Kriteriální rychlost suportu na začátku bloku. QWORD
`ParabAccel`	Parabolické zrychlení pro dynamické řízení rychlosti QWORD FLOATING
`LinearAccel`	Lineární zrychlen pro dynamické řízení rychlosti QWORD FLOATING
`SpindleSpeed`	Otáčky vřetene [ot/min] QWORD FLOATING
`SpindleSpeedLimit`	Limit otáček vřetene pro konstantní řeznou rychlost [ot/min] QWORD FLOATING
`ConstCuttingSpeed`	Rychlost pro režim konstantní řezné rychlosti [mm/min] QWORD FLOATING

Pro všechny instrukce s reálnými čísly platí, že po operaci zůstává výsledek jak v reálném registru DR_REAL, tak v celočíselném registru DR64. Proto je možno kdykoli pokračovat standardními celočíselnými operacemi. Nelze ale kombinovat celočíselné a reálné operace (viz „Základní instrukce – Reálné operace“).

Příklad:
Použití instrukce `LOD (AD,SU,MULB,DIVB,EQ,LT,GE,LE,GT):`

```
LOD   BLK[0].REAL.H                ;Načte hodnotu funkce H přímo
STO   REAL.rBunH                   ;Zapíše jako reálné číslo

LOD   BLK[0].REAL.RPlcParams[2]    ;Načte hodnotu 3.real.parametru
AD    BLK[0].REAL.RPlcParams[3]    ;Připočte hodnotu 4.real.parametru
...
LOD   BLK[BX].REAL.BlockLen        ;Načte délku bloku v um
...                                ;ve frontě příštích bloků podle BX
```

8.3.5 Konfigurace M funkcí

Přehled použitých M funkcí podle skupin (Význam je popsán v kapitole: „Povelový blok – kompatibilní verze CNC8x9“):

skupina	M funkce
1	M00 M01 M02 M30
2	M03 M04 M05 M19
3	M41 M42 M43 M44 M40
4	
5	M07 M08 M09 M17
6	M50 M51 M52 M53
7	M10 M11
8	M48 M49
9	M06 M60
10	
11	
12	
13	
14	Ostatní funkce

Konfigurace M funkcí se provede v souboru typu „ChannelConfig“

element **MFunctions**			**konfigurace M funkcí**	
	element **MFuncs**		**konfigurace jedné M funkce**	
		atribut **No**	**Číslo M funkce**	
			xx	číslo M funkce (00 – 100)
		atribut **GroupNo**	**Skupina, do které daná M funkce patří**	
			xx	skupina M funkci (1 – 14)
			14	skupina pro ostatní funkce (default)
		atribut **BreakCont**	**Ruší daná M funkce plynulou jízdu na začátku bloku?**	
			0	M funkce neruší plynulou jízdu na začátku bloku (default)
			1	M funkce ruší plynulou jízdu na začátku bloku
		atribut **BreakContEnd**	**Ruší daná M funkce plynulou jízdu na konci bloku?**	
			0	M funkce neruší plynulou jízdu na konci bloku (default)
			1	M funkce ruší plynulou jízdu na konci bloku
		atribut **ChangePos**	**Mění daná M funkce polohu stroje?**	
			0	M funkce nemění polohu stroje (default)
			1	M funkce mění polohu stroje

Obecné pravidla pro definici M funkcí:

- Do všech skupin je možno přidávat nové M funkce.
- Nově přidaným M funkcím je možno libovolně nastavit atributy „BreakCont, BreakContEnd a ChangePos“.
- Pevně přiřazené M funkce není možno ze skupin vyřadit.
- Pevně přiřazeným M funkcím je možno měnit atributy „BreakCont, BreakContEnd a ChangePos“
- Ostatní M funkce, které nejsou definovány, se přiřadí do skupiny 14.

Příklad:

V příklade se definují v 10. skupině funkce M91,M92,M93 a M94 které ruší plynulou jízdu na začátku bloku. V 11. skupině se definují fumkce M95, M96, M97 a M98, které ruší plynulou jízdu na konci bloku. Pro funkci M6 se nastavuje vlastnost, že ruší plynulou jízdu na začátku bloku a že se také mění poloha stroje.

```
<MFunctions>
            <!-- Skupina 9 -->
      <MFunc No="6" GroupNo="9" BreakCont="1" ChangePos="1"></MFunc>

            <!-- Skupina 10 -->
      <MFunc No="91" GroupNo="10" BreakCont="1"></MFunc>
      <MFunc No="92" GroupNo="10" BreakCont="1"></MFunc>
      <MFunc No="93" GroupNo="10" BreakCont="1"></MFunc>
      <MFunc No="94" GroupNo="10" BreakCont="1"></MFunc>
            <!-- Skupina 11 -->
      <MFunc No="95" GroupNo="11" BreakContEnd="1"></MFunc>
      <MFunc No="96" GroupNo="11" BreakContEnd="1"></MFunc>
      <MFunc No="97" GroupNo="11" BreakContEnd="1"></MFunc>
      <MFunc No="98" GroupNo="11" BreakContEnd="1"></MFunc>

</MFunctions>
```

8.3.6 Status

PLC program má možnost číst status z modulu reálného času pomocí bitů:

bit	popis
SF_RUN	"Systém v chodu" (stroj "jede", a/nebo jsou prováděny technologické funkce)
SF_BLOCKINPROGR	Blok rozpracován
SF_BLOCKFINISHED	Blok ukončen
SF_STOP	Všechny druhy stopu (Stop, M00, M01, BB, stop po volbě programu a pod.). Všechny bloky dokončeny, když SF_BLOCKINPROGR není nastaveno, nebo vykonávání přerušeno, když je SF_BLOCKINPROGR nastaveno.
SF_NOTINPOS	Dosud nebylo dosaženo požadované polohy (některá z os jede nebo diference některé z os neklesla pod zadanou mez).
SF_DELAYINPROGR	Časová prodleva - po dobu prodlevy programované funkcí G04.
SF_INDICATIONMODE	Indikační režim
SF_SIMULATIONMODE	Simulační režim ("ježdění bez hardwaru")
SF_HIGHSPEEDMODE	Zrychlený režim
SF_CONDSTOPMODE	Podmíněny stop aktivní (pro bloky s M01)
SF_MANUALMODE	Manuální režim ("pomocné ruční pojezdy")
SF_BLOCKBYBLOCK	Režim blok po bloku
SF_LS	Limitní spínač
SF_SLS	Softwarový limitní spínač
SF_BACKWARD	Couvání
SF_STOPPOSCHANGED	Bude nastaven, když při stopu programu bylo ručně popojeto

8.4 Speciální prvky rozhraní

8.4.1 Nastavení limitu rychlosti

Nastavení maximálního limitu rychlosti (platí i pro parabolické rampy) z PLC programu. Limit se uplatní v automatických režimech a v pomocných ručních pojezdech. Nastavení limitu pro pomocné ruční pojezdy se provede pomocí **REQ_EXT_FEED_AUTMAN** a **EXT_FEED_AUTMAN** (viz kapitolu „pomocné ruční pojezdy“ v „Návodu k programování PLC“.)

buňka	typ	význam
LIMIT_FEED_REQ	bit	1 = Požadavek na aktivaci limitu pracovní rychlosti.
LIMIT_FEED_G00_REQ	bit	1 = Požadavek na aktivaci limitu rychlosti pro rychloposuv.
LIMIT_FEED	DWORD	Nastavení limitu pracovní rychlosti. Nastavuje se 1/64000-tisících μm/ms, to znamená, že když potřebujeme nastavit rychlost v μm/ms nebo v mm/s, stačí nastavit horní word buňky.
LIMIT_FEED_G00	DWORD	Nastavení limitu rychlosti pro rychloposuv. Nastavuje se 1/64000-tisících μm/ms, to znamená, že když potřebujeme nastavit rychlost v μm/ms nebo v mm/s, stačí nastavit horní word buňky.

Příklad:
Zadání limitu rychlosti pro pracovní posun na 1 m/min
1 m/min = 1000 mm/min = 1000/60 mm/s = 17 mm/s

```
LOD   CNST.17
STO   WORD.LIMIT_FEED+2       ;naplní 17 mm/s (horní word)
FL    1,LIMIT_FEED_REQ
```

8.4.2 Čas a datum pro PLC program

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

```
      SYST_TIME_INFO          ....Systémový čas
      SYST_DATE_INFO          ....Systémový datum
      UTC_TIME_INFO           ....Systémový čas UTC
      UTC_DATE_INFO           ....Systémový datum UTC
      FILETIME_INFO           ....Systémový čas UTC ve formátu FILETIME

      (struc, čtení)

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

FILETIME_INFOS  struc
  LOW_FILETIME  DD      0     ;Formát FILETIME
  HIGH_FILETIME DD      0
FILETIME_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
      LOD   BYTE.UTC_TIME_INFO.TIME_HOUR        ;načte hodiny UTC
      LOD   QWRD.FILETIME_INFO                  ;načte formát FILETIME
```

8.4.3 Točítko s displejem

K systému je možnost připojení dvou externích panýlků s ručním ovládáním, s točítkem a s displejem.

Tlačítka z panýlku 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.

Pro ovládání displeje je pro PLC program definována struktura:

```
;Struktura pro tocitko s displejem

TOC_DISPLS  struc
      TOC_ROW           DB    0     ;radek      (0,1,...v pixelech)
      TOC_COLUMN        DB    0     ;sloupec    (0,1,...v pixelech)
      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_SIZE          DB    0     ;velikost
      TOC_INV           DB    0     ;inverse
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

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 na pozici [0,16]:

```
      FL    1, BUSY_DISPTOC1        ;zákaz obsluhy
      LOD   CNST.0                  ;0 pixelů pro řádky
      STO   BYTE.DISPTOC1.TOC_ROW
      LOD   CNST.16                 ;16 pixelů pro sloupce
      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
```

8.4.4 Plynulá jízda

PLC program může testovat, zda se právě jede plynule (G23 aktivní) a také kolik času zbývá do konce plynulého úseku bloků.

G23ACT ....plynulá jízda

(bit, čtení)

TIME_DIST ....aktualizovaný čas do konce plynulého úseku [ms]

(DWORD, čtení)

Hodnota času v buňce **TIME_DIST** je platná jen pokud je nastaven bit **G23ACT** a zmenšuje se reálně jak se blíží konec plynulé jízdy bloků. Ten nastane, když je uhel mezi bloky příliš velký, když je programován rychloposuv, když je nepohybový blok nebo když blok obsahuje technologii která vyžaduje zrušení plynulosti.

8.4.5 Zálohovaná PLC paměť

PLC program má k dispozici zálohovanou paměťovou oblast **PLC_MEM_BACKUP** o velikosti 10000 bajtů (bližší popis viz „Sdílená a zálohovaná paměť“). Při regulérním ukončení systému se automaticky celá oblast uloží na disk. Při startu systému se oblast automaticky načte. PLC program má možnost si vyžádat okamžité uložení oblasti na disk.

REQ_BACKUP_MEM ....žádost o uložení zálohované oblasti

(bit, čtení/zápis)

PLC program nastavením hodnoty log.1 do bitu **REQ_BACKUP_MEM** způsobí okamžité uložení celé paměťové oblasti `PLC_MEM_BACKUP[10000]` na disk. Po uložení zálohované oblasti na disk, systém bit `REQ_BACKUP_MEM` vynuluje.

Příklad:
Uložení oblasti PLC_MEM_BACKUP v mechanizmu s čekáním na provedení akce.

```
FL    1, REQ_BACKUP_MEM       ;žádost o uložení na disk
EX
LDR   REQ_BACKUP_MEM          ;čeká na uložení
EX1
```

8.4.6 Zrychlení a rychlosti

PLC program má možnost zjistit hodnoty zrychlení a rychlostí z konfigurace NC os v souboru typu „ChannelConfig"

Název	Typ	Popis
dwAxNCAcceleration	pole 16xDWORD	Lineární zrychlení posuvu [1/64000 um/T^2] (používá se při referenci, pro synchronní osy, pro ruční pojezdy)
dwAxNCAccelerationRT	pole 16xDWORD	Lineární zrychlení pro rychloposuv [1/64000 um/T^2]
dwAxNCAccelerationEmergency	pole 16xDWORD	Lineární zrychlení pro najetí na limit a „Emergency Stop" [1/64000 um/T^2]
rAxNCParabAccel	pole 16xREAL	Parabolické zrychlení (používá se jen pro synchronní osy)
rAxNCParabAccelRT	pole 16xREAL	Parabolické zrychlení pro rychloposuv
rAxNCParabAccelEmergency	pole 16xREAL	Parabolické zrychlení pro najetí na limit a „Emergency Stop"
dwAxNCLinearAccel	pole 16xDWORD	Lineární zrychlení pro lineární část parabolických ramp (používá se jen pro synchronní osy) [1/64000 um/T^2]
dwAxNCLinearAccelRT	pole 16xDWORD	Lineární zrychlení pro parabolické rampy a rychloposuv [1/64000 um/T^2]
dwAxNCLinearAccelEmergency	pole 16xDWORD	Lineární zrychlení pro parabolické rampy pro najetí na limit a „Emergency Stop"
dwAxNcFeed	pole 16xDWORD	Rychlost [1/64000 um/T] (používá se pro synchronní osy, pro ruční pojezdy)
dwAxNcFeedRT	pole 16xDWORD	Maximální rychlost [1/64000 um/T] (používá se pro ruční pojezdy)
rAxGParabAccel	REAL	Parabolické zrychlení pro geometrické osy. Standardně se zadává v přípravě bloku NC programu. PLC program má možnost zrychlení zadat, když je nastaven bit REQ_PARABACCEL na hodnotu 1. [m/s^3 * 65.536]
dwAxGLinearAccel	DWORD	Lineární zrychlení pro lineární část parabolických ramp pro geometrické osy. Standardně se zadává v přípravě bloku NC programu. PLC program má možnost zrychlení zadat, když je nastaven bit REQ_LINEARACCEL na hodnotu 1. [1/64000 um/T^2]

8.4.7 Licence pro PLC program

PLC program má možnost zjistit stav licencí určených pro volné využití v PLC programu. Licence jsou vázány na hardwarový klíč. Pro zjišťování licencí slouží instrukce **LDR_LIC.**

instrukce	**LDR_LIC**

funkce **čtení stavu licence do RLO**

syntax **LDR_LIC** **feature**

parameter **„feature"** **symbolická konstanta pro určení licence**

Instrukce **LDR_LIC** načte stav některé PLC licence do RLO registru. Instrukce může také zjistit stav připojení hardwarového klíče. Pokud instrukce vrátí hodnotu log.1, tak je licence pro danou vlastnost pořádku.

Licence pro dané vlastnosti PLC programu, jsou určeny symbolickými konstantami:

Licence	Význam
HwKeyConnected	Stav připojení hardwarového klíče
FeaturePlc0	Licence pro uvolnění vlastnosti PLC0
FeaturePlc1	Licence pro uvolnění vlastnosti PLC1
FeaturePlc2	Licence pro uvolnění vlastnosti PLC2
FeaturePlc3	Licence pro uvolnění vlastnosti PLC3
FeaturePlc4	Licence pro uvolnění vlastnosti PLC4
FeaturePlc5	Licence pro uvolnění vlastnosti PLC5
FeaturePlc6	Licence pro uvolnění vlastnosti PLC6
FeaturePlc7	Licence pro uvolnění vlastnosti PLC7
FeaturePlc8	Licence pro uvolnění vlastnosti PLC8
FeaturePlc9	Licence pro uvolnění vlastnosti PLC9

Příklad:

```
      LDR_LIC     FeaturePlc1       ;načíst stav licence PLC1
      JL0         LBL_DISABLE_PLC1  ;skok, když není platná licence
```